

Spring 2006

Volunteer System Project: Regis University Networking Lab Practicum

Desirea Duarte Ulibarri
Regis University

Follow this and additional works at: <https://epublications.regis.edu/theses>



Part of the [Computer Sciences Commons](#)

Recommended Citation

Ulibarri, Desirea Duarte, "Volunteer System Project: Regis University Networking Lab Practicum" (2006). *All Regis University Theses*.
411.
<https://epublications.regis.edu/theses/411>

This Thesis - Open Access is brought to you for free and open access by ePublications at Regis University. It has been accepted for inclusion in All Regis University Theses by an authorized administrator of ePublications at Regis University. For more information, please contact epublications@regis.edu.

Regis University
School for Professional Studies Graduate Programs
Final Project/Thesis

Disclaimer

Use of the materials available in the Regis University Thesis Collection ("Collection") is limited and restricted to those users who agree to comply with the following terms of use. Regis University reserves the right to deny access to the Collection to any person who violates these terms of use or who seeks to or does alter, avoid or supersede the functional conditions, restrictions and limitations of the Collection.

The site may be used only for lawful purposes. The user is solely responsible for knowing and adhering to any and all applicable laws, rules, and regulations relating or pertaining to use of the Collection.

All content in this Collection is owned by and subject to the exclusive control of Regis University and the authors of the materials. It is available only for research purposes and may not be used in violation of copyright laws or for unlawful purposes. The materials may not be downloaded in whole or in part without permission of the copyright holder or as otherwise authorized in the "fair use" standards of the U.S. copyright laws and regulations.

**VOLUNTEER SYSTEM PROJECT:
REGIS UNIVERSITY
NETWORKING LAB PRACTICUM**

Prepared for: Dan Likarish
Regis MSCIT Assistant Professor

Prepared by: Desirea Duarte Ulibarri

April 1, 2006

Acknowledgements Page

The author wishes to express sincere appreciation to Professor Dan Likarish for his assistance and guidance throughout the practicum experience.

In addition, thanks to other practicum team members, Bakir Akhmad and Sharlene Korbson, for consulting on ELMs web hosting and contributing their time, effort, and enthusiasm on getting the DTC NLP groups tasks completed.

Many thanks to Amy Pepper, Pei-Keng Foong, and Patrick Clancy for consulting on the Volunteer System project and providing input on alternative design solutions.

Special thanks to Project CURE for their continued support, vision, and patience on the Volunteer System project.

Acknowledgement to Benjamin Thomas Soloman who created the Legacy Volunteer System.

Abstract

“Volunteer System Project: Regis University Networking Lab Practicum”

By Desirea Duarte Ulibarri

The Regis University Networking Lab Practicum (NLP) allows MSCIT graduate students the opportunity to gain hands-on experience in IT topics in a controlled computer network environment. The NLP also introduces students to a wide range of external IT projects such as the Volunteer System Project. The Volunteer System Project was the construction of a database system for a non-profit organization. The Volunteer System captures all of the personnel data, work time and project affiliation information and is a critical tool in managing and maintaining a successful volunteer workforce. The performance of the original spreadsheet-based Volunteer System was declining and could not support the business needs for time tracking, querying, and reporting functions nor the rapidly increasing growth of the volunteer base. The NLP studied the original Volunteer System and proposed a project concept to reconstruct the system using relational database technology. The project was developed within the scope of the non-profit group's business requirements and implemented using industry-accepted project management processes, all phases of a standard system development lifecycle, and database development research and application. The non-profit organization would only consider the project if there were no budget requirements and only if existing licensed software, such as Microsoft Access, were utilized. Other software alternatives were considered, such as Oracle and MySQL, however the business did not have the budget to procure any new software licenses and would not have adequate support for a non-Microsoft solution after the project concluded. The NLP participated, managed, and

implemented the project with an additional constraint of only one primary project resource. The Volunteer System project provided a sound topic for the NLP student's professional project while benefiting the non-profit group with free IT development. Additionally, the documentation of the implemented project can be used as a project management and development learning tool for future NLP students.

Table of Contents

<i>Certification of Authorship Page</i> _____	<i>ii</i>
<i>Advisor Approval Page</i> _____	<i>iii</i>
<i>Project Paper Revision/Change History Tracking Page</i> _____	<i>iv</i>
<i>Acknowledgements Page</i> _____	<i>v</i>
<i>Abstract</i> _____	<i>vi</i>
<i>List of Illustrations</i> _____	<i>x</i>
<i>Chapter One: Introduction</i> _____	<i>1</i>
Statement of the problem to be investigated and goal to be achieved _____	<i>1</i>
Relevance, significance, and/or need for the project relative to the needs of the Academic Research Network _____	<i>3</i>
Barriers and/or issues _____	<i>4</i>
Elements, hypotheses, theories, or questions to be discussed/answered _____	<i>5</i>
Limitations/scope of the project _____	<i>6</i>
Summary _____	<i>7</i>
<i>Chapter Two: Review of Literature / Research</i> _____	<i>8</i>
Overview of all literature and research on the project _____	<i>8</i>
Literature and research that is specific/relevant to the project _____	<i>10</i>
Project Management _____	<i>10</i>
Systems Development Lifecycle Methodologies _____	<i>15</i>
Microsoft Solutions Framework _____	<i>18</i>
Database Design and Development _____	<i>19</i>
Summary of what is known and unknown about the project topic _____	<i>21</i>
The contribution this project will make to the Academic Research Network _____	<i>21</i>
<i>Chapter Three: Methodology</i> _____	<i>22</i>
Development methods to be used _____	<i>22</i>
Lifecycle models to be followed _____	<i>22</i>
Specific procedures _____	<i>25</i>
Formats for presenting results/deliverables _____	<i>25</i>
Review of the deliverables _____	<i>26</i>
Resource requirements _____	<i>27</i>
Outcomes _____	<i>28</i>
Summary _____	<i>29</i>
<i>Chapter Four: Project History</i> _____	<i>31</i>
How the project began _____	<i>31</i>
How the project was managed _____	<i>34</i>
Significant events/milestones in the project _____	<i>37</i>
Changes to the project plan _____	<i>39</i>
Evaluation of whether or not the project met project goals _____	<i>41</i>
Discussion of what went right and what went wrong in the project _____	<i>42</i>
Discussion of project variables and their impact on the project _____	<i>43</i>
Findings / analysis results _____	<i>44</i>
Summary of results _____	<i>45</i>
<i>Chapter Five: Lessons Learned</i> _____	<i>46</i>
What you learned from the project experience _____	<i>46</i>

What you would have done differently in the project	47
Discussion of whether or not the project met initial project expectations	47
What the next stage of evolution for the project would be if it continued	48
Conclusions / recommendations	49
Summary	51
<i>Annotated Bibliography</i>	52
Microsoft Solutions Framework	52
MSDN Academic Alliance e-Academy License Management System (ELMs)	53
Project Management	54
Regis University Jesuit Philosophy	55
Software Development	56
<i>References</i>	60
<i>Glossary</i>	62
<i>Appendix A: Volunteer System Proposal</i>	65
<i>Appendix B: Volunteer System Statement of Work</i>	70
<i>Appendix C: Volunteer System Project Plan</i>	74
<i>Appendix D: Volunteer System Design Document</i>	91
<i>Appendix E: Volunteer System Test Case Scenarios</i>	130
<i>Appendix F: Volunteer System Training Manual</i>	158
<i>Appendix G: NLP Journal Part 1</i>	188
<i>Appendix H: NLP Journal Part 2</i>	193

List of Illustrations

Figures

<i>Figure 1 - Interaction between Phases (PMBOK® 31)</i>	11
<i>Figure 2 - The Waterfall Methodology (Chapman 1)</i>	16
<i>Figure 3 - The Spiral Model (Chapman 1)</i>	17
<i>Figure 4 - MSF Process Model Phases and Milestones (MSF Process Model 23)</i>	25
<i>Figure 5 - Tradeoff Triangle (MSF Process Model 13)</i>	28

Tables

<i>Table 1 - Volunteer System Milestones/Deliverables</i>	27
---	----

Chapter One: Introduction

Statement of the problem to be investigated and goal to be achieved

The Networking Lab Practicum (NLP) of the Regis University Masters of Science in Computer Information Technology (MSCIT) program gives graduate students the opportunities to participate in hands-on lab exercises at the Academic Research Network (ARN), gain work skills in providing information technology (IT) support and administration, and find and research topics for professional projects. To complete the requirements of the Regis MSCIT graduate program, one student practiced three areas of professional project work while a member of the NLP Group B 2003: (1) participation in the NLP lab sessions and group meetings, (2) administration and support of the MSDN Academic Alliance e-Academy License Management System (ELMs) software program for NLP, and (3) IT work at an external non-profit organization executing the Volunteer System project. The student's NLP experiences and application of support and administration processes are highlighted in an appended journal. This paper discusses the student's activities to research, execute, and complete a professional project called the Volunteer System project.

The NLP enabled the student to learn and gain practice on topics such as LAN configuration and administration, web serving, data backup and retrieval, and workstation and LAN architecture. Such topics were explored in various lab sessions one or two times per week for approximately six months in late 2003. Students who participate in the NLP provide the necessary IT support for the ARN through application of learned topics while serving to complete the requirements of the Regis MSCIT graduate degree programs. The

author's lab sessions and learning topics comprise Part 1 of the NLP journal (*See Appendix G*).

The Regis MSDN Academic Alliance ELMs software program provides students and faculty the benefit to lease major software programs for free or purchase select programs at reduced prices. The author administered this program in NLP for approximately twelve months. The ELMs needed IT support and administration to configure web hosting of the most requested software, maintain and update the software catalog, and track and address user trouble tickets. The configuration and administration of Regis ELMs composes Part 2 of the NLP journal (*See Appendix H*).

The NLP also introduces students to other endeavors outside of the practicum that serve to provide a topic for completion of the Regis MSCIT professional project thesis. The Volunteer System project became available during 2003. The original Volunteer System was a spreadsheet program operated by a non-profit organization called Project CURE. The company had no full-time, paid IT administration; they relied solely on volunteers or short-term, contract consultants. As NLP students are encouraged to volunteer IT consultation to non-profit organizations and benefit from the gained experience and applied project topics, managing and executing the Volunteer System project would prove to be beneficial to both the company and participating students.

The main focus of this paper is the Volunteer System project. The original Volunteer System was a simple spreadsheet program that would take volunteer work time and project data and parse it into categories for sorting, filtering, and reporting. Part of the original system was an identification number assignment algorithm for generating a 16-digit volunteer identification number similar to the logic employed by credit card

companies. This algorithm logic used volunteer information such as name, address, and phone number to generate four separate components, each 4 digits long, for a full 16-digit number. As the numbers of volunteers over the years increased into the thousands and more robust reporting was desired, there arose a business need to convert the spreadsheet program into a more reliable, scalable and functional database solution.

The Volunteer System project encompassed reconstructing the system through application of project management, a systems development lifecycle methodology, and code development and testing. The project also included creating documentation such as user manuals, importing batch data from another database GiftMaker PRO, and providing basic IT consultation. Since, Project CURE relied mainly on volunteered help for IT support and corporate donations of hardware and software, the solution also had to be simple to maintain, easy to change, and cost effective. The goal of the work was to build a user-friendly, small-scale, and inexpensive database solution that also met the business requirements of Project CURE.

Relevance, significance, and/or need for the project relative to the needs of the Academic Research Network

The relevance of the NLP program is to provide an avenue for students to participate, learn, and benefit from research and networking exercises in a lab environment. Students don't often have the option in their day-to-day careers to configure a router, set up a web server, or reconfigure a LAN, even if their occupation is in information technology. Students who participate in the NLP get to complete research in the ARN, gain hands-on experience in networking, and find paper topics for classes and their professional project. The NLP program is beneficial and necessary for the longevity

and maintenance of the ARN. Any learned and applied networking administration and configurations are documented in lab manuals that remain in the NLP program for future students. These lab manuals record ARN enhancements and serve as a baseline for future change.

Undertaking the Volunteer System project would greatly benefit a company who had a critical business need but no budget for IT development. The project was also significant as part of the Regis University graduate program as it gave students the opportunity to provide community service which is encouraged by the Jesuit academic philosophy and included in the Regis University mission statement (Callahan 55). In addition, the project was relevant to the goal of the NLP in that students gained work experience necessary for them to complete their professional projects and lab manuals and project papers for future students. The written experiences of the student working on the Volunteer System project may be used by future NLP and ARN students to help navigate them through their own project experiences and to deal with topics such as project management, systems development lifecycle, business support, etc.

Barriers and/or issues

The informally structured environment of the NLP program requires students to be proactive and self-starters in order to successfully learn and apply topics. Thus, some students may encounter personal barriers in the NLP setting because of lack of networking knowledge and project initiation skills. At onset, this may initially force a student to be more of a lab observer rather than a participant. However, to combat such problems, the organization of the support staff of NLP is tiered. Students with little or no experience could begin at a tier 1 level and gain practical experience by working with

higher levels of support. Also, students could draw on other IT work experiences and apply such knowledge to produce a more meaningful NLP program. For example, students may use their project management skills and compensate for the lack of network administration knowledge with an abundance of project management knowledge. Thus, was the case for the author who participated in the Volunteer System project at Project CURE.

The Volunteer System project initially posed barriers for this NLP student. The topic did not center on networking, rather it focused on a database system solution. The participating NLP student may not have had the initial expertise to provide database support but, along with research and collaboration, it was possible for the student to fulfill the project management role and apply experience, insight, and learning to produce successful results.

Elements, hypotheses, theories, or questions to be discussed/answered

The NLP provides and enables students the opportunities to apply learning topics in a lab setting so that they may gain experience relative to working in an IT support role in the real world. With NLP experience and networking knowledge, NLP students could potentially manage, execute, and complete an IT project using established project management principles, tools, and standards, an accepted system development lifecycle methodology (SDLC), and applied information technology. All of these elements composed the Volunteer System project and provided a sound basis for potential project success.

It was proposed that the Volunteer System project could be successfully designed, developed, and implemented using an inexpensive database solution such as Microsoft

Access. The business requirements for system functionality and features would guide the project design and be met even with the constraints of no project budget and limited resources. The Volunteer System project could be successfully executed using applied project management, SDLC, processes, and with research and collaboration on relational database logic

Limitations/scope of the project

The Volunteer System project was necessary for Project CURE to continue to capture and track volunteer data and properly acknowledge and award their work time. The original system could not: support batch upload and table generation, be configured to capture an electronic card swipe login/logout, track volunteered time, or run queries and reporting with multiple options. The spreadsheet system was also highly subject to data errors, inadvertent changes such as deletions, and could not expand to keep up with the rapid growth of Project CURE's volunteer program.

The scope of the Volunteer System project was to include all the current functionality of the spreadsheet system including the volunteer identification number algorithm and all the above options that the spreadsheet system could not provide. The business also required that the system be accessible by multiple people at the same time and possibly remotely (if the local network would allow and support remote access). The new system was proposed to be reliable, scalable, and built using relational database logic.

Summary

The Volunteer System project was proposed to build functionality that would capture volunteer information, track volunteered time, and provide reporting for Project CURE. The Volunteer System should be able to generate a unique identification and tracking number for every new volunteer entry, log volunteer hours when a user swipes in or out, and store all other attributes of the organization's volunteers. The Volunteer system was required to be built using Microsoft Access and intended to be used to import data from another database called GiftMaker PRO. The project was originally projected to last approximately six months with no expenditures. However as the project progressed, the timeline increased to eighteen months which was necessary for major functional development components to be fully built, tested, and implemented.

The Volunteer system project was constructed with a small scale, low-cost database software solution using Microsoft Access. With Microsoft Access, the Volunteer system would be relatively simple to maintain and change for future enhancements. Project CURE owned multiple licenses for Microsoft Office Suite 2003 and could maintain a Microsoft Access DBMS on their main file server for easy access by all locally connected hosts. The student followed an established system development lifecycle methodology to define, design, develop, test, and implement the reconstruction of the Volunteer System. The student also documented all business requirements, system design, and testing, and provided this documentation in addition to user manuals to Project CURE at the end of the project.

Chapter Two: Review of Literature / Research

Overview of all literature and research on the project

The primary research for the Volunteer System project was centered on project management, system development lifecycle methodologies, Microsoft Solutions Framework (MSF), and database analysis and design. Some of the primary sources of research included project management guidebooks, technical research manuals on Microsoft methodologies, and industry handbooks on software development. Examples of the primary sources were: *A Guide to the Project Management Body of Knowledge, On Time Within Budget, Mastering Project Management, Access 2000 VBA Handbook, and Systems Analysis and Design Methods.*

Project management research helped the student to frame the project in terms of planning, scheduling, documenting, communicating, and prioritizing decisions, processes, resources, and objectives. James P. Lewis, in *Mastering Project Management*, defines project management “as the planning, scheduling, and controlling of activities to meet project objectives.” (61) A sound fundamental knowledge of project management was necessary to even consider analyzing the Volunteer System project and submitting a proposal to the business owners.

A system development lifecycle methodology (SDLC) should be employed to guide a project with an outline of phases of activities, milestones, and deliverables. The development methodology is a documented approach incorporating policies, processes, and procedures for the successful execution of a project to create products or services. Without a system development lifecycle methodology, a project may not be directed by the best intentions of the business sponsor. Rather, without solid business requirements, a

project may end up being built reflecting popular opinion, the latest and trendiest technology, and not work at all according to how the business wanted. All of these potential problems and more pose risks to the success of any project. “The best approach to applying a methodology is to consider it as a means to manage risk.” (Chapman 1). A project may never truly conclude or be fully tested and functional without following a system development lifecycle.

The Volunteer System project was typical of the average software development project. A range of customized services was needed by a customer who could not afford to purchase off-the-shelf software that came already coded with desired functionality. Textbooks such as *Systems Analysis and Design Methods* helped the student to learn more about relational database logic. Structured Query Language (SQL) and Visual Basic programming were studied and code changes and additions were facilitated by online guides and how-to books like *Access 2000 VBA Handbook*, *PHP and MySQL* and *LAN Times Guide to SQL*. Thus, research in database design and development focused on standard relational database logic, construction, and programming languages.

Ancillary research included studying paradigms such as lean programming which prescribe iterative development in software programming based on the principles of total quality management and lean production (Poppendieck 1-2). These principles espouse the concept of worker empowerment to allow for quick, integral decisions without bureaucracy and fear of retribution. This quick, on-the-fly decision-making should support efficient processes such as just-in-time inventory and pull scheduling versus push scheduling (Poppendieck 2-3). All of these production disciplines can also apply to

software programming development and aid in the management of resources, time, and costs in order to execute projects.

Literature and research that is specific/relevant to the project

Project Management

Project management processes should be practiced in conjunction with an established system development lifecycle methodology. The processes may be integrated in various and different phases of system development depending on the chosen lifecycle methodology. However implemented, “project management is a *disciplined thought process*, not to be confused with the content of what is being done” (Lewis 63). The project management processes control the project and may determine the project management activities and tools; the system development lifecycle structures a project with discernable beginning and end points, milestones, and outlines the activities, the schedule of deliverables, and ultimately produces the product.

Project organization and structure are supported by tools, techniques, standards, processes, and theories. However, project management is accomplished based on fundamental processes that every potential project should execute in order to be successful: initiation, planning, executing, controlling, and closing (PMBOK® 30). It is not enough just to throw a design document or a presentation together; a project must be executed with the above fundamentals to measure, communicate, and manage success. The undertaking of any project assumes financial and ownership responsibility by the business sponsors, but also requires performance responsibility by the project manager and project team.

Ideally, in a project-driven business, work is managed and controlled by the people doing the work—not by an executive two or three levels removed from the tasks. People in project groups are able to assume responsibility for and derive satisfaction from their own objectives while continuing to contribute to the larger objectives of the organization as a whole. (Baker and Baker 14)

The five project management processes as defined by the Project Management Body of Knowledge, are necessary to initiate and define an idea as a project, outline the project work and resources, put the project work in motion, manage all phases of execution, minimize project risk, and successfully implement a project solution (*See Figure 1*). This diagram exemplifies how each of the five process groups interacts with each other and is integrated within each system development lifecycle phase.

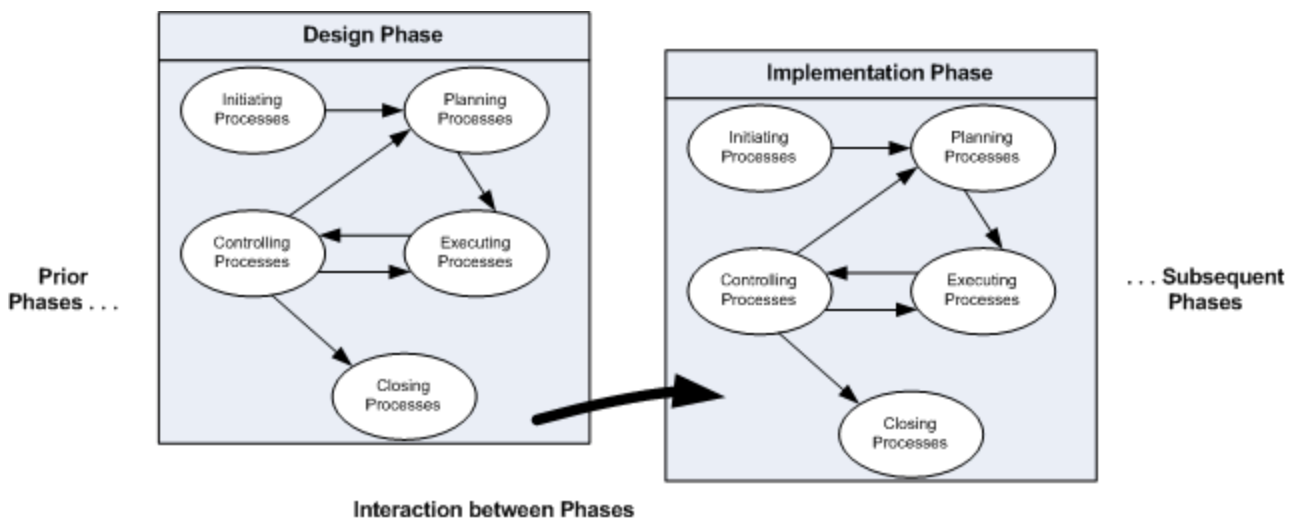


Figure 1 - Interaction between Phases (PMBOK® 31).

Risk analysis and risk management are additional and integral parts of project management and system development. A project concept is usually the result of business operations risk analysis and is often the solution to mitigate and manage such risk. For

example, the problem that was posed by Project CURE was that they did not have a sound tracking mechanism for volunteer data and volunteered time. This problem posed risks to the longevity and integrity of their volunteer program. Unless they could properly track and acknowledge volunteers for their time, effort, and participation, they would potentially lose volunteers which are the majority of their workforce. Project CURE could not maintain a business, albeit non-profit, if they do not have workers to keep them running operationally. The operation risks measured too great to accept avoidance of project development. The only acceptable solution was to analyze the Volunteer System and mitigate these risks with a reconstruction project.

Undertaking (or not) the Volunteer System Project becomes a risk itself. Successfully implementing the project would mitigate the risk to the program, but the process of implementation produces risks to cost, time, and production in each phase of the project. "Project risk is an uncertain event or condition that, if it occurs, has a positive or negative effect on a project objective." (PMBOK® 127). Thus, risk analysis and management must be integrated as part of all the project management processes throughout the project lifecycle.

Projects and project management processes begin with initiation: an idea is posed based on a current problem or identified need. During initiation, a concept is vetted and turned into a problem statement with breadth of scope determination. A feasibility analysis will also be determine if a project can and should indeed be initiated. Feasibility takes into account project cost, time, resources, and technology. The result of a feasibility analysis will determine a go or no-go decision by business stakeholders.

Planning is the next process group by which project management is accomplished (PMBOK® 32). Planning includes finalizing business requirements, outlining work and work-breakdown structures (WBS) for project groups, activity/task scheduling, requirements feasibility, and critical path definition (Baker and Baker 89). The final requirements will define and confine the scope. The scope is broken into work units that drive the statement of work and WBS. All work will produce deliverables that can be further delineated into activities and tasks. Project dependencies, assumptions, and constraints may change or influence what, when, and how tasks must be completed. Activity sequencing is important as some tasks must begin or even fully complete before others can be started. Sequencing and duration estimating will drive the critical path by which the project must maintain in order to implement according to plan (Baker and Baker 149). The schedule should account for these influences and everything should be documented in the project plan.

The executing process group includes the activities that coordinate and perform the work of the project. Leading the project group to accomplish the WBS tasks and performance of those tasks are all part of executing processes. A great degree of interaction with the project team and team development through frequent and meaningful communication, participation, empowerment, and most important leadership also embodies executing. As the project is executed, buy-in from all stakeholders should be maintained and reinforced throughout all process activities. It is not enough to just manage the project but a project manager must also lead, guide, and direct all those involved in order to make progress on project goals. Project tools and documents such as

the project plan guide the project in executing and provide a baseline of measurement for controlling.

Controlling are the processes by which the project makes progress. “If you are going to keep a project on schedule, budget, within scope, and meet quality requirements, you must have a way to measure where are for each variable of interest.” (Lewis 185). Variables that are measured for status on project progress are cost, performance, schedule (time) and scope (Lewis 191). Extension of a project’s scheduled timeline may indicate scope change or scope creep. Scope changes/creep may cause increases in development time which may result in increases in project cost, due to necessary additions in resources. One project variable may be influenced by or cause detrimental changes in another variable. Managing all variables, their impacts upon the critical path and upon each other is the process of controlling in action. Any significant departure from baselines established in the project plan for any variable such as budget or schedule should be reported immediately in order to address, mitigate problems and diminish any negative effect on project success (Baker and Baker 272).

All projects are temporary and must come to a definitive end (PMBOK® 5). If they do not, they are not truly projects but perhaps elements of operational, cyclical work or they are abandoned attempts. The processes of closing include all the activities and deliverables necessary to wrap-up and frame the project as a completed package and hand-off all end results to the business owners. Part of closing is measuring the final product against requirements, risks, and design to determine if the project goals, risk mitigations, and objectives truly came to fruition. As in all the other processes, the business owner is re-engaged and surveyed to measure their satisfaction. The project is

evaluated and scrutinized for learned lessons or examples of successful practices for modeling. Closing may also include maintenance planning for post-project operations or recommendations for future enhancements.

The project management processes outline and guide the practice of project management. They are necessary to know and use for project execution and success. Each of the process groups contain core and additional facilitating processes, but not every process or process tool will be engaged for every project. Large projects may need more planning and management of resources. However, an endeavor with only a few hands producing the work may need less activity monitoring and planning to stay on track, but requires more scrutiny of the output to ensure no critical requisites were missed.

For a small project such as the Volunteer System project, with no budget and limited resources, the process groups were necessary to keep focus on the requirements and project goals. The project encompassed scope planning, time management, and activity estimation. However, planning and measuring only a single person's work required less effort.

Systems Development Lifecycle Methodologies

The waterfall model is often described as a baseline model for systems and software development. Figure 2 is a diagram example of the waterfall model. Activities are executed in a series of phases that are dependent on the preceding phases' results that flow (waterfall) into the next phase. It is a highly structured model that incorporates strict governance, critical path management, tight budgetary controls, and the concept of gating. Gating is dependent on milestone approval. A phase cannot complete (gate

closing) until a phase review is conducted to determine if all deliverables were accomplished and outstanding issues were addressed. Once a phase gate closes, the phase is ended and not revisited unless a formal change request is initiated. Thus, when requirements are finalized and design/development begins, the requirements cannot change. All analysis must be completed in the analysis phase, all development within development phase, etc.

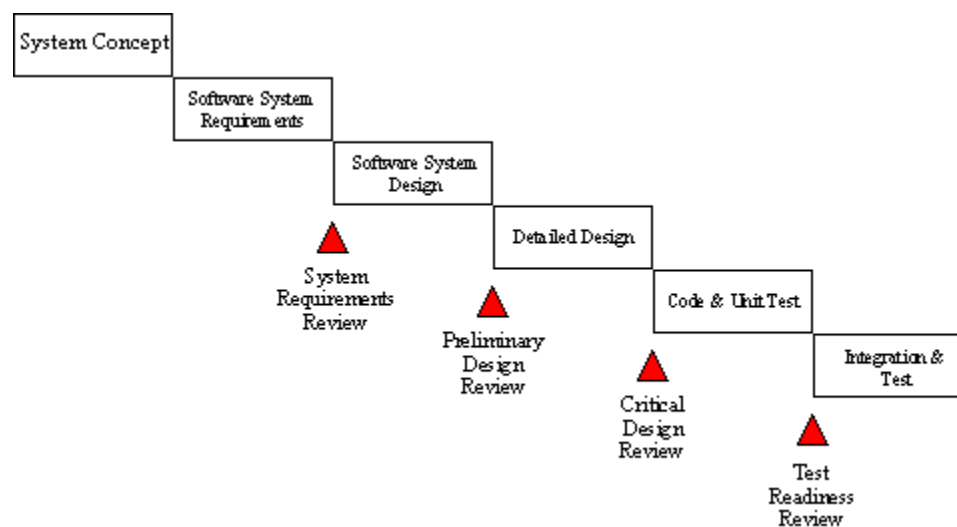


Figure 2 - The Waterfall Methodology (Chapman 1).

Figure 3 is a diagram that is a basic representation of the spiral development model. “The ultimate evolution from the water fall is the spiral, taking advantage of the fact that development projects work best when they are both incremental and iterative, where the team is able to start small and benefit from enlightened trial and error along the way.” (Chapman 1). The spiral method allows for rapid prototyping, concurrent design, and development, and re-entrance into previous phases during every iteration (spiral), even requirements. Iteration of a group of phase activities is shaped by constant business sponsor feedback and a changing vision of the final product/project. The spiral

methodology includes all phases of an SDLC, but if not managed tightly, a project can spiral out of control. A spiral model does not always incorporate clear checkpoints, thus project management processes and SDLC milestones and deliverables have to be planned and documented early and often in order to keep the project moving in a productive yet accountable direction (MSF Process Model 6).

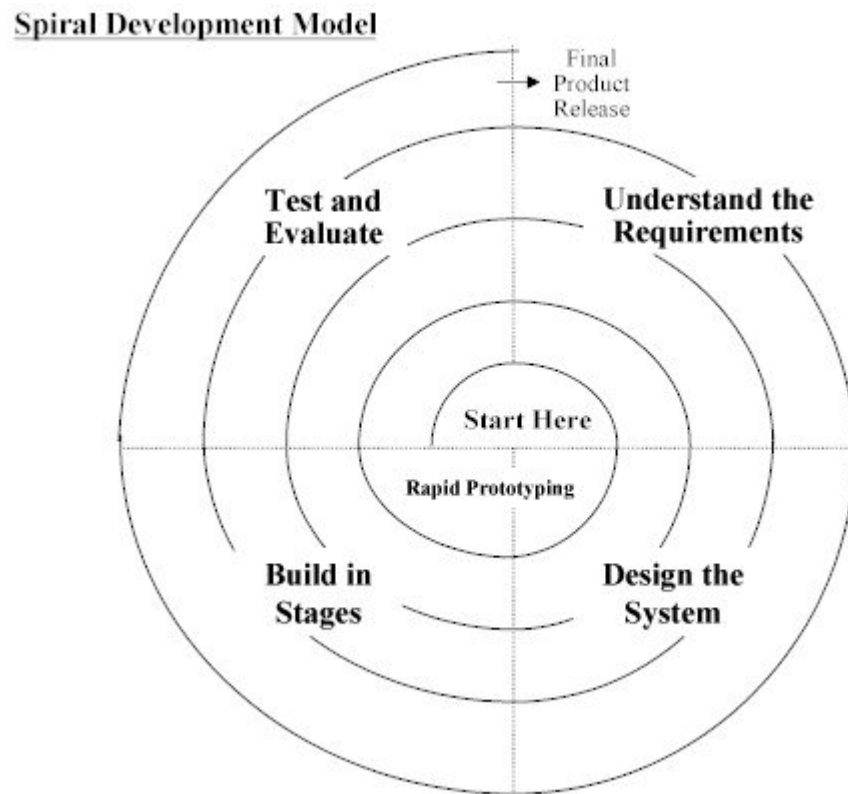


Figure 3 - The Spiral Model (Chapman 1).

The development methodology employed in the Volunteer System project was a variation of the spiral and its practice of rapid prototyping. The applied methodology took proven practices from the spiral but was more aligned with the Microsoft Solutions Framework process model. Microsoft Solutions Framework embodies a combination of the two standard lifecycle models, both the waterfall and the spiral (MSF Process Model

4). *There will be further discussion of the specific Microsoft Solutions Framework process model in Chapter 3.* The usual example of rapid prototyping “iterates in a mini-development phase until a system prototype is developed” (Bennatan 64). In the Volunteer System project, it was necessary to iterate the analysis and requirements phases twice, as the original system had no documentation and had to be continuously researched during the project. It took a couple iterations of analysis of the existing system infrastructure and requirements definition to fully document the original system’s shortcomings and determine the final functional requirements. The design, development, and testing phases were each iterated five full times.

Microsoft Solutions Framework

Microsoft Solutions Framework (MSF) is a highly customizable and scalable, framework of processes, principles, and proven practices for agile and adaptive IT project development (MSF Overview 4). MSF incorporates core disciplines and models of process, team, learning, and vision for project management prescribed within a systems development lifecycle structure. This framework is Microsoft’s attempt to combine processes, methodologies, and software to produce a holistic structure to develop IT applications, products, and services. The development phases of the system, product, or services lifecycle are administered by MSF and the operational and maintenance phases are managed by Microsoft Operations Framework (MOF) which shares the same foundational principles and core disciplines (MSF Overview 7).

MSF evolved from a collection of best practices and development efforts of different Microsoft product and technical groups over 25 years (MSF Overview 6). The framework is based on proven practices and standards and incorporates generally

accepted industry principles for project management and IT development. MSF and MOF integrate many disciplines, such as project management, risk management, readiness management, etc. with core principals into a framework that can be applied to emerging development and operational foundations so that all industry input and output are united in theory, practice, vision, and ultimately business value (MSF Overview 13).

MSF is applied using a learning-centric, team model with a shared vision. “The MSF Team Model is based on the concept of a team of peers and the implied empowered nature of such team members.” (MSF Overview 11). The model defines the roles, responsibilities, goals, and objectives for all team participants yet places equal value on every role of team members (MSF Team Model 10). It also outlines how each role applies the principles that compose MSF and measures accountability for everyone’s goals. The “team of peers” is composed of members enacting the disciplines of project, risk, and readiness management within the framework’s process lifecycle model (MSF Team Model 11). The team model utilizes the principle to learn from all past experiences and build upon these baselines for future success. Thus, it is fluid and can adapt to change in industry, technology, or business (MSF Team Model 10).

Database Design and Development

Research in relational data analysis and database logic aided the student in studying the original system and learning where to design, normalize, optimize, and make changes to the tables, code, forms, and queries. Data analysis is the process of preparing a database for implementation, removing redundancy and complexity from the data through normalization (Whitten and Bentley 408). Normalizing the data and constructing and maintaining entity relationships early in the process helped to curb data redundancy,

track relational keys, table dependencies, and optimize data performance. The logical design model of a database is based on the entity relationships and the constraints and definitions of their interdependence (Chenoweth, Schuff, and St. Louis 94). For example, if two fields of data are defined in a one-to-one relationship between tables, then there could only be one unique representation of the data field in each table; the definition is also a constraint on table construction and any forms servicing those tables.

In the development of a relational database, the goal is to support all the transaction-processing needs of the organization. The focus is on identifying the functions the organization must perform and the data required to perform them. (Chenoweth, Schuff, and St. Louis 94)

Front-end forms serve to provide protected access to users or allow input of data. Research in Visual Basic (VB) programming language taught the student the necessary programming code for forms and front-end interfaces and how to protect code from inadvertent changes or problem breaks. Users of the Volunteer System needed data entry screens that employ point-and-click and are considered graphical user interfaces (GUI). A user should not have to learn programming code or how to input command-line requests just to use the database. Software, such as Microsoft Access, provided the means to create forms that are user-friendly and protect the data from accidental erasure, changes, or additions. These forms are controlled by programming code and macros that provide instructions for every object on the forms including buttons and drop-down menus.

Once the data was in the database, it could be manipulated, retrieved, and used in calculations or reporting. Structured Query Language (SQL) online training and research gave the student the means to create scripts and queries which would move the data from

entry to permanent archive and retrieve the data when reporting and user access were necessary. All high-end relational databases and DBMS engines, such as Microsoft Access, support SQL language standards (Whitten and Bentley 406).

Summary of what is known and unknown about the project topic

The NLP student had extensive knowledge of systems development but only basic knowledge of database design and construction at start of the Volunteer System project. The student had a background in project management, management of technology, networking (system engineering), and some database design/development experience. However, with a strong foundation in project management and systems development, the student would be able to overcome any shortcomings in database development by supplementing with research, consultation, and continuous learning through the Regis MSCIT graduate curriculum, extracurricular training, and online references.

The contribution this project will make to the Academic Research Network

The Volunteer System project encompassed many roles performed by only one resource. To be a designer, programmer, tester, and the project manager required research, documentation, and trial and error. The end product itself, the Volunteer System, may not have been the best or most efficient of the possible solutions for the project; it does however serve as an example of successful project execution through the practices of established project management principles, tools, and standards, an accepted system development lifecycle methodology, and applied information technology. This project can serve as a guidebook for future ARN students to introduce them to these practices and disciplines and set them in the right direction for proposing and completing their own professional projects.

Chapter Three: Methodology

Development methods to be used

Microsoft Solutions Framework Process Model was the development methodology utilized for the Volunteer System project. “The iterative nature of the MSF Process Model requires that a shared vision exist to guide a solution toward the ultimate business result.” (MSF Overview 11). The MSF Process Model was fitting as development of the Volunteer System was iterative, adaptive, involved prototyping, and engaged the business owners in every aspect. Most importantly, the goals and vision of the project were shared between the project team and business owners. This is an integral aspect of the Microsoft Solutions Framework and key to providing business value through project development.

Lifecycle models to be followed

The MSF process model is a combination of the waterfall and spiral (MSF Process Model 5). By combining the best attributes of the two industry standards, the lifecycle of the Volunteer System project benefited from rapid prototyping, milestone/phase reviews, and design/development cycle iterations (MSF Process Model 5). The Volunteer System project did not employ a full spiral as requirements definition did not always occur each iteration; however there were a couple of rotations where requirements were reviewed and refined. “The MSF process model is designed to accommodate changing project requirements [...]” and provided the necessary framework to iterate short phases of design, development, and testing to create or change the working model (MSF Process Model 4). Support of a prototyping process was

necessary as Project CURE wanted to reconstruct an existing database and to review the changes as they were implemented.

An iterative approach such as MSF allows for more flexibility in requirements definition, design, and development of a product as applying constant review and scrutiny can reveal previously unknown needs and prerequisites that may not have been included in the original set of business requirements. While this flexibility allows for a more polished final product, if it is not managed closely scope creep, timeline increases, and even project failure may ensue, as project goals sometimes become moving targets. Early phases produce documents that are maintained and supplemented with each review and help keep the project on track. The documents begin high-level and development starts as soon as the initial iteration of planning and design completes. “By creating and baselining project documents early in the process, team members are empowered to begin development work without the delays that may be incurred in excessive planning.” (MSF Process Model 18). It is important to baseline the project early and constantly update planning and design documents to remain on target to accomplish project goals.

The benefit of using MSF over the spiral model is that the framework incorporates milestone reviews emulating the waterfall gating process. All phases are subject to milestone reviews for each completed cycle. The planning and work completed in the current iteration are reviewed, approved, and closed, creating a baseline for the next iteration. The key difference with the waterfall model is that although a gate may close, multiple iterations allow all phases and milestones to be reviewed many times. The constant review helps to develop a product that evolves more consistently with the final business vision.

The SDLC phases of the MSF process model are Envisioning, Planning, Developing, Stabilizing, and Deploying (*See Figure 4*). The Envisioning phase is where concepts are initiated and analyzed and risks and benefits are assessed and incorporated into a feasibility study. The vision of the project and its business value (worth) are first assessed and documented. The Planning phase is where project scope, solution scope, and design definition and refinement occur. Project scope outlines the work that the project team will perform to deliver the solution; the solution scope describes the specific design features and functionality of the product (MSF Process Model 11). The design captures all the instructions for development. The written deliverables of this phase document the baseline of the project. The Developing stage is where the majority of the development work takes place, such as software coding and infrastructure configuration (MSF Process Model 32). The Stabilizing phase encompasses system and user acceptance testing. This phase measures and prepares the product for delivery into a production environment. The delivery stage is the Deploying phase.

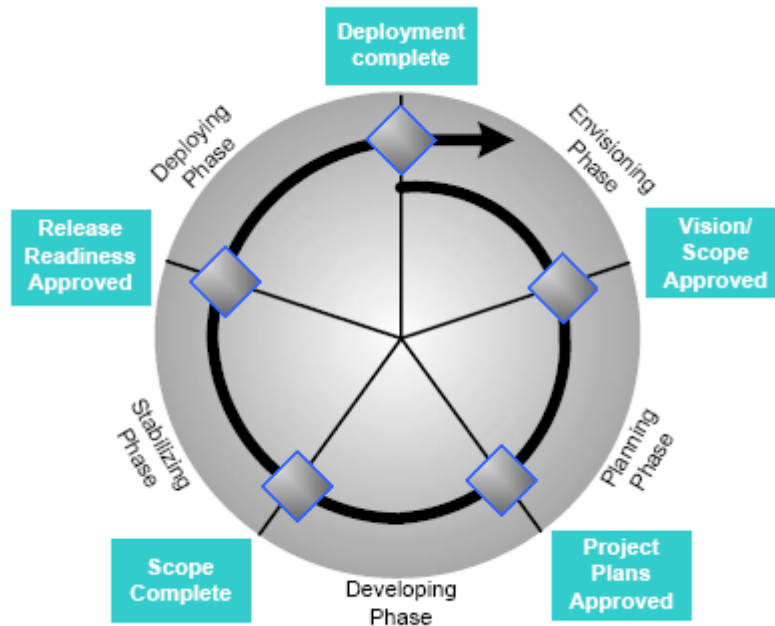


Figure 4 - MSF Process Model Phases and Milestones (MSF Process Model 23).

Specific procedures

Each phase of the MSF Process Model dictates specific milestones, activities, and deliverables but process and procedure may vary depending on the size and complexity of the project. The Volunteer System project was a small project with limited resources and a short list of project stakeholders. Thus, formality in procedure was relaxed. The documentation laid out the course of the project and phase/milestone review meetings were casual and only required two or three people. The communication was very open and mostly accomplished by email, addressing only two or three people at most.

Formats for presenting results/deliverables

The MSF process model provides fundamental theories and principals for system/product development but does not explicitly dictate the project management tools and documentation. Microsoft's website has many sample document templates for each

MSF planning, design, and development deliverable category. However, for the purposes of the Volunteer System project, the deliverables were modified and presented in user-friendly Microsoft Word or Excel documents. They are representative of simple project management documents, but they do not exclusively belong to one methodology or framework.

The main product deliverable was the prototype. For each rotation of development, the completed module was first presented using a prototype demonstration. User acceptance testing would take place after system test and bug fixes. The business was given an opportunity to test the working prototype after each module was built. In this practice of versioning, core functionality was delivered as early as possible while formatting and cosmetic enhancements were delivered last.

Review of the deliverables

Each MSF process model phase concluded with a milestone which marks the passing of the phase when it is accomplished. Below is a table of milestones and the associated deliverables and work timeline for the Volunteer System project. Envisioning began with a study of the Volunteer System and continued with a proposal of project development that would provide Project CURE with a more robust system that would address the business needs and could be simple to maintain. The proposal outlined initial scope and milestones, feasibility, timeline, resource requirements, and was presented to the business. Subsequently, a Statement of Work was agreed to which further documented the project scope and captured the benefits of the project.

For the iterated segments of Planning, Developing, Stabilizing, and Deploying, the deliverables were an up-to-date project plan, design document, a working prototype

of the system software, test cases, and a training manual for all users. Final delivery included the fully implemented software and completed documentation.

<i>MSF Phase/ PMBOK Process</i>	<i>Milestones</i>	<i>Deliverables</i>	<i>Timeline</i>
Envisioning/ Initiating	Vision/Scope Approved	Proposal Statement of Work	2 weeks 4 weeks
Planning/ Planning	Project Plans Approved	Project Plan Design Document Data Dictionary	Each Iteration = 3 to 6 months
Developing/ Executing	Scope Complete	Working Prototype	Each Iteration = 3 to 6 months
Stabilizing/ Controlling	Release Readiness Approved	UAT Test (Use) Case Scenarios	Each Iteration = 3 to 5 weeks
Deploying/ Closing	Deployment Complete	Training Manual Product Delivery	Final Delivery December 2005

Table 1 - Volunteer System Milestones/Deliverables

Each milestone review of the deliverables focused on traceability to monitor and track that the project was on course and every requirement remained in consideration. Thus, the design document details could be mapped back to the requirements and mapped forward to the test cases. “*Traceability* is the ability to determine that each feature has a source in requirements and each requirements has a corresponding implemented feature.” (Bennatan 5). The project plan kept record of this mapping of the Volunteer System project.

Resource requirements

In every project, there is a combination of resources (people and money), scope, and time that produce the deliverables (MSF Process Model 12). This combination is represented by a triangle with a single variable on each side. If the scope increases, so must another variable such as time or resources in order to maintain dependencies of the

combination (MSF Process Model 12). Microsoft calls this balance, the Tradeoff Triangle (See Figure 6). “The key to deploying a solution that matches the customer’s needs when they need it is to find the right balance between resources, deployment date, and features.” (MSF Process Model 12).



Figure 5 - Tradeoff Triangle (MSF Process Model 13).

The Volunteer System project did not have any viable options for acquiring additional resources. Project CURE did not have any additional budget to purchase resources and there was only one full-time NLP resource available during the project. With a fixed set of resources, the other variable inputs to the project were scope and timeline. As Project CURE definitely had no money for off-the-shelf software and could not pare down the solution scope any further, the timeline had to remain a flexible variable. If scope creep or resource constraints occurred, the timeline had to change to accommodate developing and delivering the full solution

Outcomes

The Volunteer System project encompassed a study of the current volunteer tracking processes and tools, definition of business requirements, research, design, development, testing, and implementation of an enhanced database system. The project was completed using the MSF iterative system development process model, a hybrid of

the spiral and waterfall methodologies with prototyping. In order to develop the database in a short timeframe with as much hands-on feedback from the customer, it was important to have a working prototype early and to iterate the design, development, and testing phases of the development lifecycle as many times as necessary to finalize the product.

Using the Volunteer System project as an example of deployed MSF, the result was a successful product delivery, however with necessary expansions to the project timeline. Lack of resources, expertise and budget contributed to timeline increases as well as external factors, such as local area network issues (*See Chapter 4*). Overall, this prototyping process model was more fitting to the project as the project was small in scale, needed a model for refinement and shaping of the business expectations, and had a short list of project stakeholders. Using MSF also allowed for a shorter time to production of a working model.

Summary

The focus of the Volunteer System project was not just to deliver a product, but to deliver business value. The most difficult job role of the project was to consistently maintain buy-in from business owners. The length of time it took to fully implement the project helped erode the initial level of buy-in. However, the last six months of the project included several rounds of robust design, development, and testing. This late flurry of activities involving both the project resource and business sponsors resurrected and highlighted the business value of the project. Buy-in was rebuilt as the final product took shape. When using MSF the focus should always be on the business value. “While many technology projects focus on the delivery of technology, technology is not

delivered for its own sake—solutions must provide tangible business value.” (MSF Overview 13).

Chapter Four: Project History

How the project began

The need for a Volunteer system to track and manage volunteer data and volunteered time began with the rapidly expanding volunteer base at Project CURE. The primary workforce of this non-profit company is composed of volunteers. In order to properly capture personnel data and track and award work time, volunteer information had to be entered, tracked, and reported. The company used a Microsoft Excel spreadsheet program as the system for a few years. However, over time the user experience degraded as the data grew too large to manage manually.

At this time, the operations group of Project CURE reached out to Dan Likarish, MSCIT Assistant Professor, who recommended two Regis NLP students as possible IT volunteers. These NLP students were introduced to Bob Standish and Michelle Sanders at Project Cure. The students agreed to conduct a survey of the original Volunteer System and produce a project proposal for overhauling the Volunteer System from the Microsoft Excel format.

The survey of the original Volunteer System uncovered these needs, assumptions, and constraints:

Needs

1. The Volunteer System needs to track and store volunteer information reliably and permanently.
2. The system needs to support batch and manual entry of data.
3. The system must be user friendly so that a user of any level can create reports and input information easily.
4. The Volunteer System needs to assign and track a unique identification number for each volunteer that can be encoded in a swipe-card.
5. The Volunteer System needs to build/support a mechanism for capturing volunteer time swipe/logging.
6. The Volunteer System needs to support reporting of volunteer data and volunteered time.

Assumptions

1. The Volunteer System will employ the original 16-digit identification number assignment algorithm.
2. The Volunteer System will maintain complete, detailed information on volunteers with the profile reports.
3. The Volunteer System must be available to all site locations at the same time, i.e. Denver, Nashville, Houston, Los Angeles and other sites.

Constraints

1. There is no budget for software or hardware procurement.
2. The Volunteer System will be built using Microsoft Access database software.
3. Project CURE has no paid IT administration or support. Thus, users and administrators would need training on the development changes.

The Volunteer System in Microsoft Excel could not support future growth and did not have the desired database functionalities. Overhaul and reconstruction of the current system would require additional software capabilities and some customization. The company had no paid IT personnel to build a new system and no budget to procure off-the-shelf software for a database solution that would address all their business needs. However, they did have licenses for Microsoft office products. Thus, the operations group decided that any new development should be built using Microsoft Access. This became a constraint on the project as alternatives such as open-source freeware MySQL were not seriously considered. Furthermore, unbeknownst to the operations department, the facilities group at Project CURE had already hired a consultant to turn the Excel process into a database using Microsoft Access. The consultant created an initial version of the Volunteer System but the contract ended before the database was fully functional and documented. It was also unknown to the students that a consultant had already converted the Excel system into a database.

After identifying the business needs, assumptions, and constraints, the two NLP students embarked on gathering a good set of business requirements. The students

submitted a proposal to recreate the Volunteer System according to these business requirements (*See Appendix A*). The proposal originally contained references to another database, GiftMaker PRO, which was also to be modified as part of the entire Volunteer System solution. However, further analysis proved that the license Project CURE held for GiftMaker PRO did not contain allowances for customization. Any additional required functionality would force Project CURE to purchase a new, more expensive license. As a result the business removed any modifications of GiftMaker PRO from of the scope of the Volunteer System project.

Prior to finalizing the business requirements, the NLP students discovered that a consultant had already created an initial version of a database for the Volunteer System in Microsoft Access. However, the developer left no documentation on how to use the database; there was no design document or a training manual. The database was not fully functional as the developer's contract ended before development was completed. At this time, the operations group decided the project should build upon this previous work. This decision added delay to the critical path as the students had to embark on a study of the new database and research the code to discern what functionality was created and what was still left to build before launching the Planning phase. The early, non-working model of the Volunteer System was referred to as the Legacy Volunteer System (LVS). During this second system study, one student left the project and only one resource remained.

To continue the project, the original business requirements from the proposal were mapped to current functionality of LVS and any gaps or non-functional areas were documented as change requirements. The student modified the scope of the original proposal and created a Statement of Work (SOW) that acknowledged the initial version

of the Volunteer System database in Microsoft Access (*See Appendix B*). The SOW outlined an agreement by the NLP student to manage the project and perform the work to enhance LVS according to change requirements rather than starting over again with the original Excel system.

How the project was managed

The Volunteer System project was managed using the PMBOK® defined project management processes and MSF Process Model. The Envisioning phase began with the project concept, continued with two separate studies of different versions of the Volunteer System, produced a project proposal, and concluded with project acceptance. The Envisioning phase aligns with the PMBOK® defined project management process group: initiation. This phase's activities also produced the SOW which documented the project scope and level of effort. The project was set in motion with acceptance of the SOW by the business and one of the NLP students.

Once the SOW was reviewed and accepted, the Planning phased commenced. Planning is one of the five fundamental project management process groups and necessary for a baseline measurement of project deliverables. The NLP student gathered functional change requirements for the LVS and documented and tracked the requirements in the project plan. The project plan outlined the solution scope based on the LVS change requirements (*See Appendix C*). The solution design was separately documented with each new piece of functionality delineated as individual modules of development (*See Appendix D*). Both documents were continuously modified throughout iterations of design, development, and testing.

The Volunteer System project timeline was broken down into milestones and deliverables that were documented in the project plan. Project activities and tasks were documented in a work breakdown structure (WBS) and each task was sequenced. Due to the constraint of only one available resource, activity sequencing did not result in many overlapping tasks as a new task could only begin when the NLP student was finished with the current task. The last planning exercise to complete the critical path was activity duration estimation. The resulting schedule and WBS were captured in the project plan. As there was no budget for additional resources, the timeline had to remain a flexible and changeable variable. The critical path became more of a guideline than a must-adhere-to standard.

The Volunteer System project was initiated with requirements gathering and definition. Through their requirements, the business asked for relational database technology to support transactional processing of volunteer data. From these requirements sprung use cases of the data which shaped the design of the system after a couple of rounds of analysis and planning. The use cases explicitly described how the data enters the system, what data is entered, how the data is used in the system, the major players interacting with the data, and all operations the data supports. The use case scenarios became the basis for each module of design of the system and test cases (Alexander and Maiden, 29).

The Volunteer System design was created during the Planning phase and built one module at a time during the Developing phase. Developing is part of the project management group executing; which accomplishes the majority of the work of the project. The first developed modules were the upload functionality, card swipe, and

Volunteer ID number generation. The last modules to be developed were the time-tracking mechanism and Volunteer System reporting. Many meetings with the business owners to clarify requirements ensued to ensure that the design continue to capture the business vision. Using rapid prototyping and employing iterative phases of design, development, and testing allowed the project owners the opportunity to sample the working model in progress.

The Stabilizing phase was where each developed module was tested in a code-test-build fashion. System testing was executed for every functional piece. In addition, the multiple rounds of user acceptance testing (UAT) that took place in the last iterations of Stabilizing served to train and update the business owners on the product prior to full implementation. Test case scenarios and testing steps were constructed and documented for UAT (*See Appendix E*). The test cases included each step of the design module that was currently being reviewed. UAT encompassed walkthroughs of all implemented functionality that required user interaction.

Stabilizing is the phase where project measurement occurs and scope liability is revealed. Much of the project management process group controlling involves measurement and accountability. Stabilizing is part of this process group. If a test defect is great enough to halt testing and commence a review of the design or further scrutiny of the requirements, then a whole new iteration of MSF Process Model phases may be required and instigated. For example, during the Stabilizing phase, it was discovered that the time calculation functionality only worked for one person at a time, rather than the whole group of volunteers. As a result, a review of the design had to occur to try to pinpoint the source of the problem. This increased the timeline, but not the scope of the

project as the defect was not with the requirements, only with the design and development. The Planning, Developing, and Stabilizing phases were reiterated for this module in order to correct the issue and deploy the functionality.

During the Deploying phase, the Volunteer System was introduced piecemeal into a prototype with completion of each module. Every completed and implemented module became part of the prototype and closed an iteration of planning, design, development and testing. A user training manual update was delivered at the close of each iterated cycle (*See Appendix F*). The project management processes for closing are part of the Deploying phase. After each implemented module, a post-implementation meeting was held with the business. The meetings were very informal and did not uncover any major issues, only questions stemming from the user experiences with the prototype. Once a completed module was deployed, it could be sampled in production but the system was not fully operational until all modules were completed.

Significant events/milestones in the project

The creation of the SOW was the first true valuation of the project scope since the concept had drastically changed from when first proposed. During the first Envisioning milestone review, it was decided to iterate the Envisioning phase with a second study to review the LVS and all changes that were made from the original Excel format. The gaps between the LVS and the previously proposed system solution were evaluated and documented as change requirements that created a new baseline for the Volunteer System project. . The SOW captured the changed level of effort (since project proposal) which was the first significant event to impact the project.

The creation of the project plan defined the solution scope as five modules of development:

1. Batch Import Functionality
 - A newly-built feature that will support importing files of data extracted from GiftMaker PRO.
2. Volunteer User Interface Enhancements
 - New interface to add the uploaded data from GiftMaker PRO and create a unique Volunteer ID number according to the 16-digit algorithm logic from the original Volunteer System.
 - New and changed user forms for separate Volunteer and Management interfaces to look up Volunteer data. Volunteer interfaces will be in add or read-only modes and Management will have the option to add, change, or delete data.
 - Enhanced swipe form that will not only capture a Volunteer ID number but will also capture login and logout time.
3. Volunteer Work Time Tracking Additions
 - Volunteered time will be captured, calculated, tracked, and stored.
4. Volunteer Reporting
 - Robust reporting, including volunteered time by Volunteer ID and date range.
5. Change Requests to New Volunteer System - *These enhancements will not be developed until after the production deployment of the Volunteer System Project.*
 - Change formatted reporting to include “Organization Group”.
 - Change sequence of login/logout user messages and pop-up screens.

Modules 1 through 5 were all developed as separate iterations of planning, developing, and stabilizing. Repository changes, such as new tables and table field additions to both the front-end and back-end database engines, occurred throughout each of these iterations rather than in its own module. Completion of a module prompted a milestone review for Stabilizing phase. If all testing was complete and a module did not need any further iterations of any phase, then it was deployed as part of the prototype.

Completion of a module was a significant event because other than the repository changes, a module could not be started until the previous module development was completed. For example, capturing, storing, and tracking of volunteer work time were built in Module 4 but could have been fully developed and deployed until Module 3 was complete. This is because Module 3 focused on developing the time swipe login. Time

could not be calculated and reported if the database could not yet capture login/logout time. The most significant milestones were not those that concluded each phase, rather the completion of each round of planning, developing, and stabilizing phases for each module.

Changes to the project plan

The project plan was initially created after the identification of the functional gaps in the LVS. The change requirements that covered the identified gaps drove the planning process. However, the initial project plan did not include all final requirements. There were a few late change requests as the project progressed which changed the project plan. A few requirements were necessary to provide full functionality to the modules; they were missed during initial requirements gathering. These items increased the project timeline but were still considered in scope of the project. These items were listed as Module 5 and only agreed to be developed after all previously planned items were completed.

The project plan was arranged by module, each broken out by subject type. The first subject was the import functionality of database. It was first necessary to get data on existing volunteers from GiftMaker PRO into the Volunteer System database. Originally, the proposal was to build an interface directly with GiftMaker PRO, but this piece was dropped by the business as GiftMaker PRO could not support any modification. The LVS was not able to upload data, nor able to create a unique Volunteer ID number, according to the 16-digit algorithm logic, for more than one volunteer at a time. Once this was discovered, there was a change in plan and design early in the project to build import functionality that not only parsed and saved data into permanent tables, but also set the

data in a staging area where a Volunteer ID number could be generated and saved permanently.

The second subject was the swipe functionality. The LVS swipe functionality was not functional but was vital for all other items of development (except upload). Once development ensued to correct the swipe functionality, it was discovered that while a user could login/logout with swipe or manual entry, time was not captured in a manner that would allow a calculation of work hours. Once again, the plan and design changed to include enhancing all time capture functionality in order to accurately calculate volunteer work time.

The third subject was the time calculation function for reporting which was the most difficult module to develop, but posed no additional surprises until testing. During testing, it was discovered that while the functionality worked, it only worked for a single user at any one time. If someone logged in, then another volunteer could not log in until that person logged out! Module 3 had to be reviewed again during the next iteration of planning, development, and testing which increased the project plan timeline.

The last subject was originally planned to be reporting. However before this module could begin, it was revealed that the business wanted a review of the login/logout utility as the users had difficulty navigating through pop-up and information screens. Also, there was a reporting change request from the business to capture additional information from the GiftMaker PRO extracts. Changes to the import file had been previously deemed out of scope of the project as the business had already agreed to a final document of import file fields and that the data accommodated their needs. However, since much of the Volunteer System functionality depended on the batch entry

of volunteer data, and it would only be a minor change to the upload process to import the required field into the Volunteer System, it was agreed to add the reporting change to the project plan and prepare for a fifth module of development. Both of these changes in scope created the need for an additional round of planning, development, and testing to change, review, and complete the necessary code enhancements.

Evaluation of whether or not the project met project goals

Not all the project goals that were defined on the original proposal were completed. Although agreed to by the business, they had to drop any proposed changes to GiftMaker PRO including interface development between that database and the Volunteer System because they did not have any license or contract provisions that would allow customization or changes to the database.

The proposal also included creating a change control process for continued maintenance and support of the system. However, the project started with two student resources and a database consultant but was executed with only one resource directing and representing all areas of the project. As there were no indications that there would be any additional, available student resources after the project was implemented to carry out a change control process, this component was removed from scope and project plan with support and agreement from the business. It did not make business sense to dictate a process that would not be implemented and supported.

The Volunteer System project did accomplish all other goals for system functionality contained in the original proposal, albeit in a longer timeframe than first anticipated. However, the most important achievement was maintaining buy-in from the business that the product was necessary and would add business value. When a project

takes much longer to implement than planned, opportunities and reasons for a business to stop and abandon attempts at completing work tend to arise. Ensuring the project and ultimate product will create business value is the goal of the process of constantly communicating with business stakeholders and highlighting when business objectives are being achieved during the project. Using a rapid prototyping methodology allows showcasing completed functionality of the product in progress which aids in this process.

Discussion of what went right and what went wrong in the project

There are many things that went right with the Volunteer System project. The main functional components were successfully designed, developed, tested, and implemented as specified by the business requirements. The swipe functionality was coded in Visual Basic to accept both login and logout time swipes and calculate hours and minutes worked based on the entered values. The Volunteer system also accepts many details of volunteer data, en masse or one at a time via batch or manual entry. In addition, the system assigns unique volunteer identification numbers according to the original 16-digit algorithm as required by the business.

Further functional successes of the Volunteer System are that it can accept user login entries via a swipe card or by a manual input (in the absence of a card) of the identification number. The Volunteer System tracks user time and provides historical data of previous logins and logout entries. Lastly, the Volunteer System is capable of robust reporting on Project CURE volunteers including work time, availability, and essentially any other attribute of the stored data. The Volunteer System project met the business requirements of Project CURE and provided a user-friendly, automatic, scalable, systematic solution.

Of the items that went wrong with the project, the very obvious issue was the lapse in the original timeline for the project. Originally the project duration was proposed to consist of six months of work. However, the project actually took over eighteen months to complete. A lack of human resources was the major factor that pushed out the delivery date. Initially the project had two NLP students surveying and analyzing the system and producing the project proposal. Soon after, there was only one student available to execute the design, development, testing, and implementation. Lack of knowledge and database experience of the NLP students also caused some delay. To remedy this, a third student from the DBA practicum was brought in to consult on the database and propose alternative solutions. However, it was then decided that the development had progressed too far to feasibly restart the project; any major changes would only add more delay for little gained benefit. The DBA student exited the project after this assessment and the project ended with only one student to see it through to implementation.

Discussion of project variables and their impact on the project

The variable of time was the most impacting on the Volunteer System project. Increases to the timeline were partially caused by the condition of the business's local network of which this project had no control. Project CURE overhauled their local network and went through a twelve-month period of transition during the Volunteer System project. The database initially was housed in a virtual file server managed and supported by a telecommunications vendor. This virtual network could only be accessed via an Internet connection with a login and password. Access was established using Citrix but Project CURE did not have much budget for more than a few portal

connections. With multiple persons using the same login and competing for limited bandwidth, the average user experience was degraded. Major coding development and system testing of the Volunteer System was difficult and sometimes impossible to accomplish over such a connection. These conditions existed for the first nine months of the project and delayed deployment of the first module of development.

Although the project did not create any out-of-pocket expenses for the business, the variable of cost still impacted the project. After Project CURE completed moving all production applications from the virtual network to an onsite, local area network (LAN), there was no longer the capability to make changes to the Volunteer System remotely. For security purposes, lack of IT administration, and budget, the LAN was not enabled with terminal services or a virtual private network (VPN). The business had no funds to hire IT personnel or procure additional software/hardware to manage security and allow remote access. Consequently, all coding had to be uploaded onsite and user acceptance testing could only be completed by the business during regular business hours.

Findings / analysis results

In other projects, the overrun of a project timeline may indicate that the project was not an overall success. However in the Volunteer System project, all other measurements indicated success, including the positive reception and functional acceptance by the business. Project CURE was satisfied with the results and was very happy to achieve such success without monetary costs or having to solicit donations. The time lapse was perceived inconsequential to the business as they received a valuable product at no more of a cost except time itself.

For the business, successful completion of the Volunteer System project meant that they would not have to pay IT consultants or make costly software purchases to fulfill their needs. The new functionality was satisfied with volunteer resources, current software assets, and no out-of-pocket expenses. Project CURE is now able to grow their volunteer program and manage volunteer data with minimal production support and training and without disruption to their production operations.

Summary of results

The Volunteer System project was successful. The enhanced Volunteer system is scalable, sustainable, cost-effective, and user-friendly. The new and changed functionality satisfies the business requirements to upload data, assign a unique volunteer ID number, capture and track volunteer data and work time, and provide volunteer reporting. The project was executed using established project management processes and a system development lifecycle methodology. The project is a good example of application of information technology that satisfies a business's needs and delivers business value.

Chapter Five: Lessons Learned

What you learned from the project experience

During the Volunteer System project, the NLP student participated as a project manager, developer, and all other major positions of a functional project team. Because of the lack of resources, the student thoroughly learned the significance of employing a matrix team with different areas of expertise and authority. The student came to recognize the importance of interdependencies of every position and the value that various expertise provides to project success. It is very difficult to implement a project with so many components with only a single resource. It is the author's opinion that the definition of a project should be enhanced with more emphasis on "collaborative effort" as a project's success depends on so many moving parts working interdependently but coming together in the end to create a holistic solution.

The author also learned that a project needs more than one perspective in order to provide the best possible solution. When different people are analyzing and brainstorming for a solution, all knowledgeable viewpoints together can lead a team to choose the best alternative. When only a single perspective is pursued, it becomes difficult to consider any other alternative as appropriate especially once development begins. Thus was the case with the Volunteer System; during the Developing phase of the project, another consultant studied the system and provided different suggestions for design and development. However, as development was already in advanced stages of progress, it was very difficult to consider starting over and abandoning any previously completed work. Although staying the course turned out to be the best decision,

employing multiple viewpoints earlier in the Envisioning phase would have been a better practice and may have translated into increased business value.

What you would have done differently in the project

If the Volunteer System project were to be executed again, the author would recommend fully deciphering all VB code and SQL scripts before submitting a statement of work. There was code that had to be researched again late in the project prior to completing some modules of development. The level of effort of development continued to increase even after development started because the analysis of the Legacy Volunteer System did not include studying every piece of the code, only those that were deemed relevant. In hindsight, it would have been beneficial to have consulted with a VB expert and created a truer estimate of development for the critical path.

While the student did consult with a database expert and research alternative solutions (other than Access) for the Volunteer System project, this occurred much too late in the project to influence the design. Furthermore, the business was set on using Microsoft Access; thus, some time was unwisely spent reviewing alternative software that would not and could not be employed due to this business constraint. It would have saved time, effort, and provided more value to have considered different alternatives earlier in the Envisioning phase and then focus fully on the chosen solution rather than backtracking in the middle of the project.

Discussion of whether or not the project met initial project expectations

Overall, the Volunteer System project met its initial expectations. The proposal for the Volunteer System project contained this objective: “The proposal is to recreate the

Volunteer System with a more efficient technological solution according to the business requirements of Project CURE” (*See Appendix A*). The project did recreate the system with enhancements to form, functionality, and use with a more efficient technological software solution as required by the business. Also, the project identified and met these business needs: to track and store volunteer information reliably and permanently, to support batch and manual entry of data, to assign and track a unique identification number for each volunteer, to capture volunteer time swipe/logging, to report volunteer data and volunteered time, and to be user friendly. All of this was achieved with a small resource set and no budget. With only an increase in working timeline, all the major components of the project were accomplished and successfully implemented.

What the next stage of evolution for the project would be if it continued

The Volunteer System and all other production applications in use at Project CURE need security enhancements. The next stage of evolution for the Volunteer System project would be to create and implement a security plan. Also needed are backup and recovery processes that should be included as part of the security plan. The plan should be developed in cooperation with executive management at Project CURE and focus on network infrastructure, application software, and remote access. A security plan is necessary to guide future implementation and upgrades, provide processes for operation and intrusion detection/responses, and outline penalties for violation of security rules. Such a plan needs support and vision from the top of the company down to every user of any IT systems at Project CURE to be successful, viable, and beneficial.

During the Volunteer System project, Project CURE upgraded their local area network (LAN) from vendor-managed server hosting to a local server/client, 10/100

Ethernet network. The new LAN employed a star configuration with one switch connected to all local workstations, servers, and a single perimeter-placed router with a firewall connected to a T1. This configuration provides basic network protection from outside intruders using firewall filtering rules. However further future security hardening is recommended on all application servers, web servers, and email server. (At the time of the project implementation, Project CURE had not installed an internal email system; all email was provided through a web-based vendor application). Other Project CURE sites in different states cannot access the Volunteer System without enabling terminal services or installing a virtual private network (VPN) on the LAN; security enhancements will be necessary to safely allow these services. Until security is increased and maintained, terminal services will remain disabled and a VPN is not allowed. Project CURE needs to hire IT personnel to fully maintain and defend such security as it warrants much more time and expertise than an average volunteer could provide.

It was originally proposed to produce a change control process for continued maintenance and support of the Volunteer System. However since there were no other available NLP student resources beyond the timeframe of project launch, it was decided the NLP would not provide ongoing maintenance of the Volunteer system. As such, the student participating on the project removed maintenance and support from the scope and did not dictate an ongoing process rather leaving it to future IT volunteers or staff.

Conclusions / recommendations

The Volunteer System could potentially evolve into a more robust workflow and project tracking system with multiple login sites and reporting centers. To fully support production operations, the Volunteer System should be implemented across multiple sites

in different states. If possible, the backend database should be located on a server connected to the wide area network (WAN) backbone to allow access to all local area networks. At the time of implementation, Project CURE did not have a WAN or a VPN. All site workstations originally had access only to the Internet and users would login and use applications hosted on a vendor server. The scope of the project did not include network changes as the business was already in the process of changing their LAN from the vendor server solution to a local client/server configuration. The business did not have any budget for further LAN changes or the necessary IT support to create a VPN; provisioning across multiple sites was just not feasible at the time of this project.

For the longevity and scalability of the system, it is recommended that Project CURE consider different software for both the backend database engine and front-end interfaces. Alternative solutions were researched and a feasible option would be to allocate a dedicated server and build the database using freeware such as MySQL. The front end GUI could also be built on the server using PHP, also an open source freeware product. Such a solution would provide greater storage capacity, functionality, and allow for multiple sites to easily access the data, without having to purchase new client licenses. A server license for MySQL may not even be necessary as Project CURE is a non-profit group and will not make money from use of the open-source software (Ullman XV).

The Volunteer System in Microsoft Access will be suitable and efficient for a few years, most likely only needing minor upgrades and additional reporting enhancements. The database engine is reliable, but breaks can be easily repaired with a little research using online help guides or a Microsoft Access handbook. Microsoft also offers 24 hour helpline support so no problem should remain an issue for long. With its use and growth

remaining constant or steadily increasing, the Volunteer System will be sustainable for the use of capturing, tracking and reporting of data and work time for many years.

Summary

The Volunteer System project concluded with delivery of final documentation of the project plan, design, and training manuals of the Volunteer System. User acceptance testing served to train Project CURE administration staff on the different uses of the Volunteer System and how to recognize a break or problem. Eventually, the system will need to be learned by a different volunteer or IT staff for long-term support. The design document with data dictionary and entity relationships should serve as a comprehensive guide to all code enhancements, SQL and macro scripts, and table configurations. From these documents, the next administrator, project manager, designer, or developer should be able to discern the new functionality of the Volunteer System and be able to make changes where appropriate.

The Volunteer System project served as one student's example of professional work in project management, system development, IT configuration, administration, and support. The project was a good marriage of all the most important characteristics of implementing a business solution: the study and application of information technology using a methodology of processes and a framework for requirements gathering, development, testing, and business acceptance. As all graduates of the Regis MSCIT program must learn and employ project management at some point in their degree plans, the student used this project to fulfill that role and put theory into action. The result of the Volunteer System project was not only a completed professional project but also an example of providing business value through information technology.

Annotated Bibliography

Microsoft Solutions Framework

Microsoft Corporation. "Microsoft Solutions Framework Version 3.0 Overview."

Microsoft Corporation. 2003. 5 May. 2004

<<http://www.microsoft.com/downloads/details.aspx?FamilyID=50dbfffe-3a65-434a-a1dd-29652ab4600f&DisplayLang=en>>. This is a company white paper that describes the basic components of the Microsoft Solutions Framework systems development methodology. The entire framework is composed of all the major disciplines of IT development, such as project management, risk management, and development lifecycle methodologies. The overview was helpful in assessing and choosing a development methodology for the Volunteer System as it outlined a systematic approach defined through many years of best learned practices and IT experience.

Microsoft Corporation. "MSF Process Model v. 3.1." Microsoft Corporation. 2002. 5 May. 2004

<<http://www.microsoft.com/downloads/details.aspx?FamilyID=e481cb0b-ac05-42a6-bab8-fc886956790e&DisplayLang=en>>. This is a company white paper that discusses the MSF Process Model as a lifecycle methodology for system development. The process model is described as a series of procedures that can be adapted and applied to a variety of project examples with a number of recommended, supporting practices. This model was fitting for the Volunteer System project because it prescribes iterative, business vision-oriented development

Microsoft Corporation. "MSF Project Management Discipline v. 1.1." Microsoft Corporation. 2002. 5 May. 2004

<<http://www.microsoft.com/downloads/details.aspx?FamilyID=9ab963d5-5932-4e22-9209-d0560ba05f19&DisplayLang=en>>. This is a company white paper that describes the specific project management discipline underlying the Microsoft Solutions Framework development methodology. Project management for MSF can be adapted for small to large projects and the paper describes a number of useful tools, processes, and templates that can be used for simple to complex endeavors. The paper was especially useful in describing team organization and project planning process such as work breakdown structures and activity estimating.

Microsoft Corporation. "MSF Team Model v. 3.1." Microsoft Corporation. 2002. 5 May. 2004 <<http://www.microsoft.com/downloads/details.aspx?FamilyID=c54114a3-7cc6-4fa7-ab09-2083c768e9ab&DisplayLang=en>>. This is a company white paper that describes the team-oriented, collaborative approach to product and system development. The paper focuses on the human resource aspect of project management with specifics on team structure, roles, responsibilities, and best management practices.

MSDN Academic Alliance e-Academy License Management System (ELMs)

MSDN Academic Alliance. "ELMS for MSDNAA SetUp Instructions." MSDN Academic Alliance Microsoft. 2003. 5 Jan. 2003 <https://msdn03.e-academy.com/regis_grad/index.cfm?loc=admin/getting_started&parentID=2&CFID=2777670&CFTOKEN=65893639>. This is a manual for ELMs administrators. The manual outlines procedures for web hosting software, upload and management

of user names and passwords, and configuration of storefront. The manual is updated frequently with any changes from MSDN Academic Alliance.

Project Management

Baker, Sunny, and Kim Baker. The Complete Idiot's Guide to Project Management.

Indianapolis: Alpha Books, 2000. This is a project management guidebook with how-to instructions. The basics of the PMBOK® process groups are covered in addition to risk management. Project management planning tools are also covered in detail. This book is useful in touching all the major areas of project planning, but it is very high-level.

Bennatan, E.M. On Time Within Budget. New York: John Wiley & Sons, Inc, 2000. This

book is a summary of practices and techniques for software project management.

The information is focused heavily on system development lifecycle methodologies and industry accepted standards. This book introduces PM planning tools and project management case studies.

Chapman, James PMP. "Software Development Methodology a.k.a. System

Development Life Cycle." Principle Based Project Management Information and

Training Site. 2004. 3 Feb. 2005 <http://www.hyperthot.com/pm_sdm.htm>. This website is a useful index for many project management topics, training, and tools.

The main focus of information is software development. There are available documents and software programs to download with instructions on how to use the materials and how to obtain support. The project management information is based on the PMBOK® standards and other recognized knowledge areas such as

Capability Maturity Model Integration (CMMI). The diagrams are very useful in describing methodology and project lifecycle structure.

Lewis, James P. Mastering Project Management. New York: McGraw-Hill, 1998. This is a self-improvement book with a clear focus on project management and people management practices. The author purports the essence of project management as more than just design and development but as a discipline where expertise is achieved through learning to gain and maintain control and truly manage a project. This book was informative and explained many project management theories with example of real scenarios.

Project Management Institute, Inc. A Guide to the Project Management Body of Knowledge (PMBOK® guide). Newtown Square: Project Management Institute, Inc., 2000. This is a guidebook to the profession and theories, processes, and procedures of project management. The material is an American National Standard and the organization is nationally accredited. The practices included in the PMBOK® are generally accepted, apply, and are useful to a majority of projects across many industries.

Regis University Jesuit Philosophy

Callahan, John J., S.J., "Foundations The Jesuit Tradition at Regis University." School for Professional Studies - Regis University. 1997. 31 Oct. 2005
<<http://www.regis.edu/regis.asp?sctn=ars&p1=asn&p2=spsgform>>. This paper is a series of essays discussing the Jesuit tradition at Regis University and how that tradition is exemplified through history, education, service to others, etc. The paper includes the mission statement of Regis University which begins with, "Regis

University educates men and women of all ages to take leadership roles and make a positive impact in a changing society”. This mission statement was a goal of the Volunteer System project and served to inspire and highlight the human value of the project.

Software Development

Alexander, Ian, and Neil Maiden. “Scenarios, Stories, and Use Cases: The Modern Basis for System Development.” IEE Computing & Control Engineering 15.5 (2004): 24-29. Academic Search Premier (EBSCO). Regis University Lib., Denver. 31 Oct. 2005 <<http://search.epnet.com>>. This article describes use case scenarios and the role they play in systems development, especially during requirements definition and refinement. The article references the main source of this information as the book “Scenarios, Stories, Use Cases: Through the Systems Development Life-Cycle,” which was edited by this article’s authors. The use case scenario method is beneficial in describing user action and interaction of each piece of required functionality to fully explain what a system must do, who is performing the action, and the expected response. Use cases are not only used in requirements definition but also for test case structuring and sometimes in design assessments.

Anderson, Virginia. Access 2002: The Complete Reference. Berkeley:

Osborne/McGraw-Hill, 2001. This book is a reference and programming manual for Microsoft Access 2002. The book includes how-to lessons with steps from creating a database to applying programming code for custom functions and interfaces. The manual was helpful in looking for specific functional topics such as “creating user

forms” or “automating with macros”. However, this book does not contain complex Visual Basic code instructions.

Chenoweth, Tim, David Schuff, and Robert St. Louis. “A Method for Developing Dimensional Data Marts.” Communications of the ACM 46.12 (2003): 93-98. Academic Search Premier (EBSCO). Regis University Lib., Denver. 31 Oct. 2005 <<http://search.epnet.com>>. This article describes the details of construction of a data mart database versus that of a relational database. While focusing primarily on dimensional data marts, the article does delve into the logic and uses of relational databases to contrast the differences between data types, data uses, and operational support requirements for the two database technologies. The logic of the argument that supports the differentiation of two technologies is a good basis for describing how, what, and why relational databases are used and who uses them.

Groff, James R., and Paul N. Weinberg. LAN Times Guide to SQL. Berkeley: Osborne/McGraw-Hill, 1994. This book is a textbook for learning the Structured Query Language (SQL) for application in database construction and querying. The textbook discusses the differences in the multitude of varying SQL languages, but teaches SQL in a basic/universal manner. The textbook was very useful as a reference to create, repair, and change queries using SELECT, MAKE-TABLE, DELETE, and APPEND statements.

Novalis, Susann. Access 2000 VBA Handbook. Alameda: SYBEX Inc., 1999. This book is a developer’s handbook for Visual Basic programming code for applications in Microsoft Access. The handbook provides coding logic, code samples, and essentially teaches a reader how to program specific functions for a variety of uses.

The book also delves into SQL coding and was very applicable and useful for the Volunteer System project.

Poppendieck, Mary. "Lean Programming Part 1 of 2." Software Development. May. 2001. 5 Oct. 2005

<<http://www.sdmagazine.com/documents/s=731/sdm0105e/0105e.htm>>. This article describes the tenets of Total Quality Management and inventory management enhancements of the early 1980's and how the same principles apply to modern practices such as lean programming. The author produces a list of ten best practices of TQM that are applied and espoused by lean programming as a methodology of adaptive software development. Lean programming is very much aligned with adaptive, agile, and iterative development which is fundamental to Microsoft Solutions Framework.

Poppendieck, Mary. "Lean Programming Part 2 of 2." Software Development. Jun. 2001. 5 Oct. 2005

<<http://www.sdmagazine.com/documents/s=730/sdm0106g/0106g.htm>>. This article is a continuation of part 1 and builds upon the origins of TQM and continues the list of the ten best practices of lean programming. Lean programming description is expanded and described for project management, especially in the area of software development.

Whitten, Jeffrey L., and Lonnie D. Bentley. Systems Analysis and Design Methods.

Boston: Irwin/McGraw-Hill, 1998. This is an industry textbook of the fundamentals of system development with emphasis on software engineering. Relational database development is briefly discussed with discussion on entity relationships, data flows,

and use cases. This book is dated, but still very relevant as a guideline to database development. The fundamentals of relational database were incorporated in the Volunteer System.

Ullman, Larry. PHP and MySQL for Dynamic Web Sites. Berkeley: Peachpit Press, 2003. This is a development handbook with scripts and instructions on creating PHP code and developing MySQL databases. The author is presenting the material as a complete solution for back-end database engines with web-based front-end interfaces. This book was useful in comparing alternative solutions to client software such as Microsoft Access for the Volunteer System project.

References

- Alexander, Ian, and Neil Maiden. "Scenarios, Stories, and Use Cases: The Modern Basis for System Development." IEE Computing & Control Engineering 15.5 (2004): 24-29. Academic Search Premier (EBSCO). Regis University Lib., Denver. 31 Oct. 2005 <<http://search.epnet.com>>.
- Baker, Sunny and Kim Baker. The Complete Idiot's Guide to Project Management. Indianapolis: Alpha Books, 2000.
- Bennatan, E.M. On Time Within Budget. New York: John Wiley & Sons, Inc, 2000.
- Callahan, John J., S.J., "Foundations The Jesuit Tradition at Regis University." School for Professional Studies - Regis University. 1997. 31 Oct. 2005 <<http://www.regis.edu/regis.asp?sctn=ars&p1=asn&p2=spsgform>>.
- Chapman, James PMP. "Software Development Methodology a.k.a. System Development Life Cycle." Principle Based Project Management Information and Training Site. 2004. 3 Feb. 2005 <http://www.hyperthot.com/pm_sdm.htm>.
- Chenoweth, Tim, David Schuff, and Robert St. Louis. "A Method for Developing Dimensional Data Marts." Communications of the ACM 46.12 (2003): 93-98. Academic Search Premier (EBSCO). Regis University Lib., Denver. 31 Oct. 2005 <<http://search.epnet.com>>.
- Lewis, James P. Mastering Project Management. New York: McGraw-Hill, 1998.
- Microsoft Corporation. "Microsoft Solutions Framework Version 3.0 Overview." Microsoft Corporation. Jun. 2003. 5 May. 2004 <<http://www.microsoft.com/downloads/details.aspx?FamilyID=50dbfffe-3a65-434a-a1dd-29652ab4600f&DisplayLang=en>>.

Microsoft Corporation. "MSF Process Model v. 3.1." Microsoft Corporation. 2002. 5 May. 2004

<<http://www.microsoft.com/downloads/details.aspx?FamilyID=e481cb0b-ac05-42a6-bab8-fc886956790e&DisplayLang=en>>.

Microsoft Corporation. "MSF Team Model v. 3.1." Microsoft Corporation. 2002. 5 May.

2004 <<http://www.microsoft.com/downloads/details.aspx?FamilyID=c54114a3-7cc6-4fa7-ab09-2083c768e9ab&DisplayLang=en>>.

Poppendieck, Mary. "Lean Programming Part 1 of 2." Software Development. May.

2001. 5 Oct. 2005.

<<http://www.sdmagazine.com/documents/s=731/sdm0105e/0105e.htm>>.

Project Management Institute, Inc. A Guide to the Project Management Body of Knowledge (PMBOK® guide). Newtown Square, PA: Project Management Institute, Inc., 2000.

Ullman, Larry. PHP and MySQL for Dynamic Web Sites. Berkeley, CA: Peachpit Press, 2003.

Whitten, Jeffrey L. and Lonnie D. Bentley. Systems Analysis and Design Methods. Boston: Irwin/McGraw-Hill, 1998.

Glossary

Academic Research Network (ARN): A practice networking lab for the students of Regis's MSCIT graduate program.

Activity duration estimation: Estimating the length of time to complete tasks by a fixed amount of resources.

Activity sequencing: Placing an order to project activities that accounts for dependencies and schedule.

Client/server: A relationship of data exchange between computer components that facilitates workload sharing in a seamless and invisible process to end-users.

Data dictionary: A reference of descriptions of database table data including the types, sizes, and value descriptions.

Entity relationships: The interdependencies of data within and among database tables.

10/100 Ethernet network: A logical and physical network configuration of computer components that are connected to a switch that provides 10MB of bandwidth through each switch port for each separate exchange of data.

Firewall: A logical barrier of entry of a computer network that places restrictions based on rules for the passing, exchange, and transmission of data from both internal and external components.

Graphical User Interface (GUI): The visual interface that an end-user would receive and interact using icons and graphics rather than programming/query language.

Information Technology (IT): The application of technology for managing, sharing, constructing, and processing information.

Legacy Volunteer System (LVS): The initial, non-working model of the Volunteer System constructed in Microsoft Access.

Local Area Network (LAN): A combination of computer components, such as workstations, servers, printers, wiring hubs, etc., that are geographically, logically, and physically connected to interact, share resources, and exchange data and information with one another.

Microsoft Operational Framework (MOF): A set of production best practices for sustainability and manageability of IT applications and products.

Microsoft Solutions Framework (MSF): A cohesive structure of processes, principles, and practices for system development of IT applications and products.

MSDN Academic Alliance e-Academy License Management System (ELMs): A tracking, distribution, and management system that allows purchases and leases of Microsoft software to authorized students and faculty.

Networking Lab Practicum (NLP): A program for Regis's MSCIT graduate students to apply systems engineering knowledge and professional projects in a constructive and supported environment.

Spiral: A methodology of project development that prescribes working all phases of the development lifecycle in iterative cycles that may be repeated for clarity and refinement of requirements; cycles may also be progress concurrently rather than consecutively.

Statement of Work (SOW): A contract of delivery of products or services and the specifics of work that will be performed.

Structured Query Language (SQL): A standardized, non-proprietary, query language that communicates with relational database management systems.

T1: A high-capacity digital transmission circuit comprised of twenty-four 64 Kbps channels for a total bandwidth of 1.544 Mbps which is split up with some channels utilized for data and others for voice.

User Acceptance Testing (UAT): A user perspective testing that allows end-users to experience and test software in a controlled environment for the purpose of formally accepting the deliverables based on a set of pre-defined criteria.

Visual Basic programming language (VB): A programming language that is evolved from BASIC language but can be built visually using events and easily manipulative sub processes for quicker and higher-level code development.

Waterfall: A methodology of project development that prescribes working each phase of the development lifecycle in sequence where the completion and review of one phase initiates the beginning of the next phase.

Work Breakdown Structure (WBS): An itemization of groups of activities and tasks that construct the necessary components for completion of a project's work.

Appendix A: Volunteer System Proposal

Prepared by Desirea Duarte Ulibarri and Amy Pepper

Distributed October 7, 2003

Prepared for Bob Standish

Thesis

Project CURE is a nonprofit organization that provides medical supplies to those in need around the world. Project CURE has many offices in the United States and a regional office in Denver, Colorado. This regional office is in need of volunteers to create and support IT tools for two systems. This proposal presents an IT project of two phases: The first phase is to create and support a database for the Volunteer System and the second phase is to modify and support GiftMaker Pro. This proposal will outline the project's scope, methodology, activities, and deliverables.

Existing Situation

Currently the data for the Volunteer System is in file form, housed in an Excel spreadsheet. All data is entered onto a spreadsheet form and stored within the document. The Volunteer System must be transitioned to a more functional and permanent software medium. Microsoft Access is also used to store the data pertaining to the volunteers. Though Microsoft Access is more advanced than Microsoft Excel, it still will not provide the kind of database Project CURE will need in the future.

Another software package that is currently used for Project CURE is GiftMaker Pro. GiftMaker Pro can track donation and pledge information, create reports and graphs, provide analysis, and organize the tracking system (just to name a few). GiftMaker Pro is not producing these types of reports accurately. At this time, employees are having difficulty working with GiftMaker Pro while creating reports. For example, two different users will try to create the same report, providing the same information, and the deliverables are two completely different reports. This type of inaccuracy causes problems with the data when trying to run a report for specific information. In addition, GiftMaker Pro also does not seem to be user friendly especially to novice users. The credibility of Project CURE is at stake if this type of problem arises.

The proposal is to recreate the Volunteer System with a more efficient technological solution according to the business requirements of Project CURE. Overhaul of the Volunteer System is necessary to sustain information growth and support the business processes of Project CURE. In addition, modification and support of GiftMaker Pro is necessary. A better understanding and knowledge of GiftMaker Pro would be very beneficial to Project CURE.

Problem Solution

WHY

Project CURE will give us, the students at Regis University in the Master's Program, the opportunity to receive hands on experience in the IT world, work with databases and provide IT support. At the same time, Project CURE will adopt a database that will function at a higher level to provide detailed information when needed. The current

system needs to be updated and be able to grow and expand along with the growth and expansion of Project CURE. With the amount of volunteers working with Project CURE, the system needs to be able to run daily, monthly, ad hoc report at any given time.

WHAT

Create a database that will allow Project CURE to organize data and run reports. The first phase will be to create and support the Volunteer System. Phase two is modification and support of GiftMaker Pro, a fundraising software database. At the same time, we will gain experience and knowledge for our Professional Project.

WHO

Students of Regis University Graduate Networking Lab Practicum will provide IT support for Project CURE for the remainder of 2003 and approximately 3 months of 2004. The students' roles are to provide research, analysis and creation of IT tools for general IT support of Project CURE. The students will fulfill research hours for completion of their professional projects with hands-on experience in IT. Creation and support of the Volunteer System is the first phase of the proposed project. Modification and support of the GiftMaker Pro database is the second phase.

HOW

The proposal for the first phase of the project is to study the Volunteer system, gather functional requirements for the system, design, develop, test, implement, and support a new database. The first phase will be completed using an iterative system development methodology, a hybrid of the spiral method and rapid prototyping. In order to develop the database in a short timeframe with as much hands-on feedback from the customer, it is important to have a working prototype early and to iterate the design, development, testing, and implementation phases of the development lifecycle.

The activities for the first phase of the proposed project are listed below. Each completed activity denotes a milestone that will produce the indicated deliverable:

<i>Milestones</i>	<i>Deliverable</i>	<i>Timeline</i>
Project Initiation	Proposal	2 weeks
Requirements	Statement of Work	1 week
Design w/Iteration	Design Document and Entity Relationship Diagram	Each Iteration = 3 to 5 weeks
Development w/Iteration	System Software	
Testing w/Iteration	Working Prototype	
Implementation w/Iteration	Product Delivery	
Training	Training Document	
Maintenance/Support	Change Control Process and User Manual	Three Users = 1 week
		Ongoing until Project Ends July 2004

The student will begin the project with a short study of the Volunteer System and a project proposal. Once the proposal is accepted, the students will compile functional requirements for the design of Volunteer System. A preliminary draft of the requirements is attached (See Appendix A). The requirements drive the scope and level of effort of the first phase of the project. The students will compose a statement of work (SOW) based on the scope and expected level of effort.

For each iterated segment of the design, development, testing, and implementation, the deliverables are a work-in-progress design document with ERD, a working prototype of the system software, and the final product upon the last iteration. Once implemented, the students will create training manual for all users of the Volunteer System. The students will also document all new system functionality and changes and provide a user manual to Project CURE at the end of the project. The students will also create a change control process for continued maintenance and support of the system.

The timeline column marks the time of completion to achieve each milestone. While a few of the iterated activities can occur in parallel, the other activities should occur in sequence whereupon one must complete before the next starts. The entire first phase of the proposed project, to create and support a new database for the Volunteer System, with two full iterations of the design, development, testing and implementation segment will take approximately 14 weeks. The second phase of the project, to modify and support the GiftMaker Pro database, will take approximately 12 weeks. The details of system development (if any) will be outlined at a later time.

Assumptions

The students will not own or maintain any system permanently for Project CURE. Once the project ends, all IT tools are the sole responsibility of Project CURE. The students will use the project and final products as research for development of their professional project papers but retain no rights to the final software/hardware product. The students will keep a copy of the software (without data) for research and documentation.

In addition, students are not responsible for any production process or business functionality. The students are merely creating and supporting IT tools within an acceptable system development lifecycle with no dependencies on any other project or production process. Project CURE will retain all responsibility for the integrity, retention, backup, and recovery of their company data. The students will protect the privacy of such data and not intentionally disclose to anyone outside of Project CURE.

Since Project CURE is not an IT company, many of the employees and volunteers do not have the background to work with databases. The database that is created must be user friendly so that a user of any level can create reports and input information easily. These systems must also be available to all locations simultaneously, with information updated instantaneously.

Appendix 1—Attachments



"Vol System
Req1.xls"

Appendix B: Volunteer System Statement of Work

Prepared by Desirea Duarte Ulibarri

Distributed December 1, 2003

Prepared for Michelle Sanders

High Level Description:

The need for a Volunteer System to track and manage volunteer data and volunteered time began with the rapidly expanding volunteer base at Project CURE. As a result, Project CURE hired a consultant to turn the original Volunteer Management Excel process into a database using Microsoft Access. The consultant created an initial version of the Volunteer system but the contract ended before the database was fully functional. The current state of the Volunteer Management database is that many user forms are not complete and the required card swipe and time tracking functionalities do not yet work as designed. The Volunteer System project will reconstruct the Volunteer Management database according to the business requirements of Project CURE.

Key Benefit:

The Volunteer System project is necessary for Project CURE to continue to track volunteers and properly acknowledge and award their work time. If Project CURE cannot properly track and potentially award volunteers for their time, efforts, and participation, then they could potentially lose volunteers, thus part of their workforce. Project CURE cannot maintain a business, albeit non-profit, if they do not have workers to keep them running operationally.

Scope of Work:

This statement of work outlines the project scope of the Volunteer System project:

1. Regis MSCIT NLP students have already participated in a study of the Volunteer System project. The students completed a survey/analysis of the original Volunteer System (Microsoft Excel program) and subsequently the more recent Microsoft Access database Volunteer System, which will be addressed as the Legacy Volunteer System (LVS) in all future documentation. A formal proposal was produced and presented to Project CURE for initiation of the Volunteer System project.
2. One student resource is assigned to execute the Volunteer System project. The student agrees to manage the Volunteer System project and address the original business requirements with a solution of reconstructing LVS. The original business requirements will be mapped to current functionality of LVS and any gaps or non-functional areas will be documented as change requirements for the reconstructed system. The student will use an established System Development Lifecycle (SDLC) and standard project management processes to execute the project.
3. The student will produce a project plan that will document the business vision, change requirements, change requests, and solution scope. The project plan will outline the milestones, deliverables, and a schedule for each phase of the SDLC.
4. The student will iterate phases of design, development, and testing to produce and evolve a working prototype of the reconstructed Volunteer System. In this manner, business feedback is solicited and welcomed. Requirement changes and enhancements may result based on user-perspective feedback. However, all proposed changes must first be negotiated and agreed between the student and Project CURE before they will be incorporated in the project plan and scheduled.

5. The student will create a solution design that will document all changes, additions, and enhancements to LVS. This design document will include query and macro scripts, VB coding, entity relationship descriptions, and a data dictionary. The solution design will be a document-in-progress until final testing of all changes is complete.
6. The student will create test cases and conduct User Acceptance Testing.
7. The student will create a user training manual that will provide step-by-step instructions on using the Volunteer System. The user training manual will be a document-in-progress until completion of final implementation.

The original proposal contained references to another database GiftMaker PRO that was also to be modified as part of the entire Volunteer System solution. However, further analysis proved that the license Project CURE held for GiftMaker PRO did not contain allowances for customization. Any additional required functionality would force Project CURE to purchase a new, more expensive license. As a result the business removed any modifications to GiftMaker PRO from the scope of the Volunteer system project.

Limitations:

The student is not authorized, thus is not planning to make configuration changes to the local area network infrastructure, virtual private network, or wide area network. Any assumptions or requirements to make the Volunteer System available to multiple locations on or off-site are dependent on the current state of the network and any limitations/allowances of technology that are present at the time of system deployment. Regis MSCIT NLP has expressed interest in consulting on network configuration and security, but has not been permitted for the basis of this project.

The student will not own or maintain any system permanently for Project CURE. Once the project ends, all IT tools are the sole responsibility of Project CURE. The student will use the project and final products as research for development of the professional project paper but retains no rights to the final software/hardware product. The students will keep a copy of the software (without data) for research and documentation.

In addition, students are not responsible for any production processes or business operations. The students are merely creating and supporting IT tools within a standard system development lifecycle with no dependencies on any other project or production process. Project CURE will retain all responsibility for the integrity, retention, backup, and recovery of their company data. The students will protect the privacy of such data by not intentionally disclosing to anyone outside of Project CURE.

Requirements:

Attached are the original business requirements submitted by Bob Standish on October 19, 2003.



"Vol System
Req1.xls"

Estimated Completion Date:

Estimated completion date of the request is July 2004.

Cost:

The Regis University MSCIT NLP will not charge Project CURE for execution of the Volunteer System Project. All participation in the project is on a volunteer-only basis. Neither party is contractually obligated in any manner, financial or otherwise, to initiate, maintain, or complete work and/or consultation.

Terms of Payment: NA

Appendix C: Volunteer System Project Plan

Author: Desirea Ulibarri
Author Position: Project Manager
Date: February 9, 2005

Version: 5.0 Final

PROJECT SUMMARY

This project plan will document the business vision, change requirements, change requests, and solution scope and outline the milestones, deliverables, and a schedule for each SDLC phase of the Volunteer System Project.

Business Vision:

To effectively track, manage and view volunteer information, including hours volunteered, to aid in local and national reporting. In addition, the Volunteer System will serve as a tool to aid in the recognition and reward of key PROJECT C.U.R.E. volunteers.

Project Description:

The need for a Volunteer System to track and manage volunteer data and volunteered time began with the rapidly expanding volunteer base at Project CURE. As a result, Project CURE hired a consultant to turn the original Volunteer Management Excel process into a database using Microsoft Access. The consultant created an initial version of the Volunteer System but the contract ended before the database was fully functional. This original system will be referred to as Legacy Volunteer System (LVS). The current state of LVS is that many user forms are not complete and the required card swipe and time tracking functionalities do not yet work as designed. **The original business requirements from the proposal are mapped to current functionality of LVS and any gaps or non-functional areas are documented as change requirements. The Volunteer System project will reconstruct LVS according to the change requirements of Project CURE.**

Project Scope - High Level:

The student will iterate phases of design, development, and testing to produce and evolve a working prototype of the Volunteer System (reconstructing LVS). In this manner, business feedback is solicited and welcomed. Requirement changes and enhancements may result based on user-perspective feedback. However, all proposed changes must first be negotiated and agreed between the student and Project CURE before they will be incorporated in the project plan and scheduled.

Solution Scope:

1. Batch Import Functionality
 - A newly-built feature that will support importing files of data extracted from GiftMaker PRO.
2. Volunteer User Interface Enhancements
 - New interface to add the uploaded data from GiftMaker PRO and create a unique Volunteer ID number according to the 16-digit algorithm logic from the original Volunteer System.
 - New and changed user forms for separate Volunteer and Management interfaces to look up Volunteer data. Volunteer interfaces will be add or read-only modes and Management will have the option to add, change, or delete data.
 - Enhanced swipe form that will not only capture a Volunteer ID number but will also capture login and logout time.
3. Volunteer Work Time Tracking Additions
 - Volunteered time will be captured, calculated, tracked, and stored.
4. Volunteer Reporting
 - Robust reporting, including volunteered time by Volunteer ID and date range.
5. Change Requests to New Volunteer System - *These enhancements will not be developed until after the production deployment of the Volunteer System Project.*
 - Change formatted reporting to include “Organization Group”.
 - Change sequence of login/logout user messages and pop-up screens.

Project Milestones/Deliverables/Timeline:

<i>MSF Phase</i>	<i>Milestones</i>	<i>Deliverables</i>	<i>Timeline</i>
Envisioning	Vision/Scope Approved	Proposal Statement of Work	2 weeks 4 weeks
Planning	Project Plans Approved	Project Plan Design Document Data Dictionary	Each Iteration = 3 to 6 months
Developing	Scope Complete	Working Prototype	Each Iteration = 3 to 6 months
Stabilizing	Release Readiness Approved	UAT Test (Use) Case Scenarios	Each Iteration = 3 to 5 weeks
Deploying	Deployment Complete	Training Manual Product Delivery	Final Delivery December 2005

See Work Breakdown Structure for full timeline.

Project Schedule:

WORK BREAKDOWN STRUCTURE							
	Volunteer System Project						
	Desirea Duarte Ulibarri						
WBS	TASKS	RESOURCE	PLANNED START DATE	PLANNED FINISH DATE	STATUS	ACTUAL FINISH DATE	NOTES
1	Envisioning Phase		09/01/03	12/08/03			
1.1	Project Concept	Daniel Likarish	09/01/03	09/01/03		09/01/03	
1.2	System Study (Analysis) and Project Proposal	Desirea Ulibarri Amy Pepper	09/01/03	10/07/03	Completed	10/07/03	
1.3	Proposal Review, Business Acceptance, and Business Requirements Definition	Bob Standish Michelle Sanders	10/07/03	10/21/03	Completed	10/19/03	2 weeks
1.4	Business Requirements Review	Desirea Ulibarri Amy Pepper	10/21/03	10/28/03	Completed	10/28/03	1 week
1.5	First Envisioning Milestone Review	Desirea Ulibarri Amy Pepper	10/28/03	11/01/03	Completed	11/01/03	Discovery of LVS -Noted in Project Plan
1.6	Second System Study of Legacy Volunteer System (LVS) and Statement of Work	Desirea Ulibarri Amy Pepper	11/01/03	11/08/03	Completed	11/30/03	Date Slip 3 weeks -Noted in Project Plan
1.7	SOW Review and Business Acceptance	Michelle Sanders	12/01/03	12/01/03	Completed	12/01/03	
1.8	Second Envisioning Milestone Review	Desirea Ulibarri	12/01/03	12/08/03	Completed	12/08/03	
2	Planning Phase		01/01/04	09/30/05			
2.1	LVS Change Requirements Definition	Desirea Ulibarri	01/01/04	01/08/04	Completed	01/08/04	Date slip 3 weeks
2.2	Create Project Plan	Desirea Ulibarri	01/01/04	03/31/04	Completed	03/31/04	
2.3	Create Design Document	Desirea Ulibarri	01/01/04	03/31/04	Completed	03/31/04	
2.4	Design use cases and functional specifications	Desirea Ulibarri	01/01/04	01/31/04	Completed	03/31/04	
2.5	Design code and queries	Desirea Ulibarri	02/01/04	02/28/04	Completed	02/28/04	
2.6	Design GUIs	Desirea Ulibarri	03/01/04	03/31/04	Completed	03/31/04	
2.7	Second Requirements Revision	Desirea Ulibarri Pei-Keng Foong	02/13/04	06/30/04	Completed	06/30/04	Alternative Project Solutions are researched
2.8	Update Project Plan	Desirea Ulibarri	07/01/04	09/30/04	Completed	09/30/04	
2.9	Update Design Document	Desirea Ulibarri	07/01/04	09/30/04	Completed	09/30/04	
2.10	Second Module Design	Desirea Ulibarri	07/01/04	07/31/04	Completed	07/31/04	
2.11	Second Project Plan/Design Doc updates	Desirea Ulibarri	07/01/04	07/31/04	Completed	07/31/04	
2.12	Second Planning Milestone Review	Desirea Ulibarri	08/01/04	08/01/04	Completed	07/31/04	
2.13	Third Module Design	Desirea Ulibarri	10/01/04	10/31/04	Completed	10/31/04	
2.14	Third Project Plan/Design Doc updates	Desirea Ulibarri	10/01/04	10/31/04	Completed	10/31/04	
2.15	Third Planning Milestone Review	Desirea Ulibarri	10/31/04	10/31/04	Completed	10/31/04	
2.16	Fourth Module Design	Desirea Ulibarri	01/01/05	02/28/05	Completed	09/30/05	Date Slip 7 months – critical path lapse
2.17	Fourth Project Plan/Design	Desirea Ulibarri	01/01/05	03/31/05	Completed	09/30/05	

	Doc updates						
2.18	Change Requests Definition	Desirea Ulibarri	06/01/05	09/30/05	Completed	12/31/05	
2.19	Fifth Project Plan/Design Doc Updates	Desirea Ulibarri	06/01/05	09/30/05	Completed	12/31/05	
2.20	Finalize Design and Documentation	Desirea Ulibarri	09/01/05	09/30/05	Completed	12/31/05	
2.21	Final Planning Milestone Review	Desirea Ulibarri	09/30/05	09/30/05	Completed	12/31/05	
3	Developing Phase		04/01/04	12/01/05			
3.1	Review Functional Design	Desirea Ulibarri	04/01/04	04/30/04	Completed	04/30/04	
3.2	Develop Module 1 (code)	Desirea Ulibarri	04/01/04	04/30/04	Completed	04/30/04	
3.3	Create/Enhance GUI or queries	Desirea Ulibarri	05/01/04	05/31/04	Completed	04/30/04	
3.4	Developer testing (primary debugging)	Desirea Ulibarri	06/01/04	06/30/04	Completed	04/30/04	
3.5	Developer Integration Testing (build)	Desirea Ulibarri	06/15/04	06/30/04	Completed	04/30/04	
3.6	First Developing Milestone Review	Desirea Ulibarri	07/01/04	07/01/04	Completed	06/01/04	Completed Mod 1 early by 4 weeks
3.7	Develop Module 2	Desirea Ulibarri	08/01/04	09/30/04	Completed	09/30/04	
3.8	Code/Test/Build	Desirea Ulibarri	08/01/04	09/30/04	Completed	09/30/04	
3.9	Second Developing Milestone Review	Desirea Ulibarri	09/30/04	09/30/04	Completed	09/30/04	
3.10	Develop Module 3	Desirea Ulibarri	11/01/04	12/31/04	Completed	11/30/04	
3.11	Code/Test/Build	Desirea Ulibarri	11/01/04	12/31/04	Completed	11/30/04	
3.12	Third Developing Milestone Review	Desirea Ulibarri	12/31/04	12/31/04	Completed	11/30/04	Completed Mod 3 early by 4 weeks
3.13	Develop Module 4	Desirea Ulibarri	03/01/05	05/31/05	Completed	11/30/05	
3.14	Code/Test/Build	Desirea Ulibarri	04/01/05	05/31/05	Completed	11/30/05	
3.15	Fourth Developing Milestone Review	Desirea Ulibarri	06/01/05	06/01/05	Completed	11/30/05	
3.16	Update Project Plan/Design Docs	Desirea Ulibarri	11/01/05	12/01/05	Completed	02/09/05	Date Slip 5 weeks
3.17	Final Development Milestone Review	Desirea Ulibarri	12/31/05	12/31/05	Completed	02/09/05	
4	Stabilizing Phase		06/01/04	12/31/05			
4.1	Develop UAT test scenarios using Design use cases	Desirea Ulibarri	06/01/04	06/30/04	Completed	06/30/04	
4.2	Test software and GUI to specifications	Desirea Ulibarri	07/01/04	07/31/04	Completed	08/31/04	Mod 1 testing delayed 4 weeks due to network issues
4.3	First User Acceptance Testing	Desirea Ulibarri Michelle Sanders	07/01/04	07/31/04	Completed	09/30/05	Business Owner is not available/able to test (GiftMaker PRO issues) - critical path lapse
4.4	Identify anomalies and Modify Code	Desirea Ulibarri	07/01/04	07/15/04	Completed	09/30/05	Testing timeline pushed out 12 months
4.5	Re-test modified code	Desirea Ulibarri	07/15/04	07/31/04	Completed	09/30/05	
4.6	First Stabilizing Milestone Review	Desirea Ulibarri	07/31/04	07/31/04	Completed	09/30/05	
4.7	Second User Acceptance Testing	Desirea Ulibarri Michelle Sanders	10/01/05	10/31/05	Completed	10/31/05	
4.8	Identify defects and Modify Code	Desirea Ulibarri	10/01/05	10/15/05	Completed	10/31/05	

4.9	Re-test modified code	Desirea Ulibarri	10/16/05	10/31/05	Completed	10/31/05	
4.10	Second Stabilizing Milestone Review	Desirea Ulibarri	10/31/05	10/31/05	Completed	10/31/05	
4.11	Third User Acceptance Testing	Desirea Ulibarri Michelle Sanders	11/01/05	11/30/05	Completed	12/31/05	Date Slip 4 weeks
4.12	Identify defects/Modify/Retest Code	Desirea Ulibarri	11/01/05	11/30/05	Completed	12/31/05	
4.13	Third Stabilizing Milestone Review	Desirea Ulibarri	11/30/05	11/30/05	Completed	12/31/05	
4.14	Fourth User Acceptance Testing	Desirea Ulibarri Michelle Sanders	12/01/05	12/31/05	Completed	01/31/06	Date Slip 4 weeks
4.15	Identify defects/Modify/Retest Code	Desirea Ulibarri	12/15/05	12/31/05	Completed	01/31/06	
4.16	UAT Testing Signoff	Desirea Ulibarri	12/31/05	12/31/05	Completed	01/31/06	
4.17	Update Project Plan/Design Docs	Desirea Ulibarri	12/01/05	12/31/05	Completed	02/09/06	
4.18	Final Stabilizing Milestone Review	Name	12/31/05	12/31/05	Completed	02/01/05	
5	Deploying Phase		06/01/04	01/07/06			
5.1	Deploying Module 1	Desirea Ulibarri	08/01/04	08/08/04	Completed	08/08/04	Deploying to prototype without UAT
5.2	Training Manual Update	Desirea Ulibarri	06/01/04	06/30/04	Completed	05/15/04	
5.3	First Deploying Milestone Review	Desirea Ulibarri	08/08/04	08/08/04	Completed	09/30/05	Reviewed again after UAT
5.4	Deploying Module 2	Desirea Ulibarri	10/31/05	11/07/05	Completed	11/01/05	
5.5	Training Manual Update	Desirea Ulibarri	11/01/05	11/07/05	Completed	11/01/05	
5.6	Second Deploying Milestone Review	Desirea Ulibarri	11/08/05	11/08/05	Completed	11/01/05	
5.7	Deploying Module 3	Desirea Ulibarri	12/01/05	11/30/05	Completed	12/31/05	Date Slip 4 weeks
5.8	Training Manual Update	Desirea Ulibarri	12/01/05	12/01/05	Completed	none	No update needed
5.9	Third Deploying Milestone Review	Desirea Ulibarri	12/01/05	12/01/05	Completed	12/31/05	
5.10	Deploying Module 4	Desirea Ulibarri	12/31/05	01/07/06	Completed	02/08/06	Date Slip 4 weeks
5.11	Training Manual Update	Desirea Ulibarri	01/01/06	01/07/06	Completed	02/09/06	
5.12	Fourth Deploying Milestone Review	Desirea Ulibarri	01/07/06	01/07/06	Completed	02/09/06	
5.13	Final Development Milestone Review	Desirea Ulibarri			Not Started		Scheduled 4/1/06

Requirements Definition:

The business requirements for reconstructing LVS are change requirements from the original proposal. These change requirements should cover the gaps in functionality from the initially proposed system to LVS. The change requirements are the basis for the Volunteer System Project going forward. These requirements were produced by the student with the business acceptance after thorough study of LVS and its shortcomings.

PROJECT REQUIREMENTS

Original Business Requirements from Project Proposal		LVS Gaps	LVS Change Requirements (Functional Requirements for New Volunteer System)		Change Requests to New Volunteer System		Last Update
1	The system must maintain a volunteer list	None - current functionality					
1.1	The system must categorize volunteer by interests and skills	None - current functionality					
2	The system must track a volunteer's availability and when they are available	None - current functionality					
2.1	The system must visibly specify the days and hours each volunteer is available	None - current functionality					
3.1	The system must record work history through batch entry	Does NOT exist	1.1	The system shall support importing extract files of data from GiftMaker PRO.	Import Temp File fields redefined		09/05/2005
		LVS will only generate Volunteer ID for manually entered data.	2.1	The system shall create a unique Volunteer ID number according to the 16-digit algorithm logic from the original Volunteer System for batch imported and manually entered data.			
			2.2	The system shall allow modification of imported batch data.			
4.1	The system must provide reporting of who is available, by dates, interests and skills	Not fully functional - data does exist	2.3	The system shall allow look up all Volunteer data.			

		LVS contains no restrictions for reading, changing, or deleting data based on user level.	2.4	The system will provide different views for volunteer users and management. Volunteers should have read-only access to limited information (TBD by Mgmt). Management should have admin access to all data.			
3	The system must track volunteer work history	Does NOT exist	2.5	The system shall capture login and logout times for each volunteer.	CR# 1-2 Pop-up message only - no screen opening CR# 2-2 Pop-up message with screen opening CR# 3-2 Pop-up error only - No screen openings CR# 4-2 Pop-up only - No screen openings CR# 5-2 Pop-up error only - No screen openings	Change the sequence of login and log-out user messages and pop-up screens: 1-2 Login sequence with no Checked Records. 2-2 Login sequence with Checked Records. 3-2 Login sequence when logging twice in a row. 4-2 Normal Logout sequence. 5-2 Logout sequence when twice in a row.	12/31/2005
3.2	The system must summarize total hours worked automatically	Does NOT exist	3.1	The system shall capture, calculate, track, and store volunteered time.	Review time calculation and revise queries		09/01/2005
4.2	The system must provide reporting of work history by dates, activities and volunteers	Does NOT exist	4.1	The system shall produce robust reporting			
4	The system must provide reporting based on pre-defined or imputed parameters	Does NOT exist	4.2	The system shall produce reporting with the following option parameters: Volunteer Date Range Report that shows the same information on the original report, but allows me to view by project, by site, by city, or by person. Sub-alphabetized by person's last name.	CR# 1-4 - Change base query CR# 2-4 - Change report compile query CR# 3-4 Change form code CR# 4-4 Change report format	The report RPT_DateRange will be enhanced to include the field "Organization Group" as part of the volunteer name header.	12/31/2005

4.3	The system must provide reporting of volunteer profile details	Not fully functional - data does exist	4.3	The system shall provide reporting of all volunteer data by user level. Volunteer users can report on any read-only interface or form to which they have access. Mgmt can report on any interface or form.			
5	The system must be created and supported in Microsoft Access	None - current functionality					
6	The system must provide for input electronically or via agent set	None - current functionality					

PROJECT ASSUMPTIONS

Original Business Assumptions from Project Proposal		LVS Change Requirements		Change Requests to New Volunteer System		Last Update
1	The Volunteer System will facilitate cultivation of volunteer relationships	Assumed				
2	The Volunteer System will reduce data entry by using a preliminary group of available volunteers	Assumed				
3	The Volunteer System will facilitate in rewarding and recognizing volunteers through history analysis and reporting	Assumed				
4	The Volunteer System will facilitate in keeping in contact with volunteers through mailings and email	Assumed				
5	The Volunteer System will maintain complete, detailed information on volunteers with the profile reports	LVS Change Requirement	4.3	The system shall provide reporting of all volunteer data by user level. Volunteer users can report on any read-only interface or form to which they have access. Mgmt can report on any interface or form.		

6	The Volunteer System must be available to all site locations at the same time, i.e., Denver, Nashville, Houston, Los Angeles and other sites	Outside of Scope of Volunteer System Project	The student is not authorized, thus is not planning to make configuration changes to the local area network infrastructure, virtual private network, or wide area network. Any assumptions or requirements to make the Volunteer System available to multiple locations on or off-site are dependent on the current state of the network and any limitations/allowances of technology that are present at the time of system deployment. Regis MSCIT NLP has expressed interest in consulting on network configuration and security, but has not been permitted for the basis of this project.
---	--	--	--

PROJECT CONSTRAINTS

Constraints		LVS Change Requirements		Change Requests to New Volunteer System		Last Update
1	LVS must remain in current operating version of Microsoft Access					
2	There is no budget for software or hardware procurement.					

REQUIREMENTS TRACEABILITY

MODULES 1 AND 2				
Change Requirement #	Design Document #	Design Script/Code/Macro	Test (Use) Case Scenario	Test Description
1.1 The system shall support importing extract files of data from GiftMaker PRO.	Module 1: 1.1.1		0010 - Upload/Data Entry Data will enter the Volunteer Mgmt DB by Upload/Import	Data can be uploaded into the Volunteer Mgmt DB from an Excel, comma-delimited, .CSV, XML, or text file or another database table.
	Module 1: 1.2.1	See Module 1 – Import Macro in VB		
	Module 1: 1.3.1	See Module 1 – Import File Specs		
	Module 1: 1.4.1	See Module 1 – QRY_ImportTemp SQL		
	Module 1: 1.9.1			
	Module 1: 1.7.1	See Module 1 - mFRM_TESTGenUniqueKey VB code with embedded SQL statements	0012 - Upload/Import data is permanently saved in the Volunteer Management system.	Immediately after a unique ID number is generated, VB code will parse and push the uploaded and additional data to populate the VolunteerMasterInformation, PermanentContact, BusinessContact, and BusinessOccupation tables. The inserted data contains the unique primary record number key from GiftMaker PRO as well as the newly assigned Volunteer ID number which now becomes the primary and relational key for all four tables in the Volunteer Mgmt DB.
	Module1: 1.6.1			

	Module 1: 1.7.1	See Module 1 - mFRM_TESTGenUniqueKey VB code with embedded SQL statements		A person can search by Date the record was entered into the table
	Module1: 1.6.1			
	Module 1: 1.7.1	See Module 1 - mFRM_TESTGenUniqueKey VB code with embedded SQL statements		A person can search by GiftMakerPRO Record Number if the record was entered by the Uploaded/Import function.
	Module1: 1.6.1			
	Module 1: 1.7.1	See Module 1 - mFRM_TESTGenUniqueKey VB code with embedded SQL statements		A person can search by Member ID.
	Module1: 1.6.1			
	Module 1: 1.11.1			Once ImportTemp data contains a unique Volunteer ID number and is inserted in the permanent tables, it will no longer be retrieved by the 'Add New Volunteer Master Information' form.
	Module 1: 1.6.1		0013 - The unique primary record number key from GiftMaker PRO should prevent duplicates from being imported and uploaded multiple times, and the	If data has already been saved in the permanent tables, the system will NOT allow other records into these tables with identical GiftMakerPRO Record Number.
	Module 1: 1.6.1		Volunteer ID primary key should prevent duplicate numbers from being used as an ID number.	If data has already been saved in the permanent tables, the system will NOT allow other records into these tables with identical MemberID numbers.
2.1 The system shall create a unique Volunteer ID number according to the 16-digit algorithm logic from the original Volunteer System for batch imported and manually entered data.	Module 1: 1.7.1Module 1: 1.10.1Module 2: 1.2.1	Module 1 - QRY_ImportTempFilter SQL.Module 1 - TEST_GenUniqueKey_frm field properties.	0021 - Create MemberID for upload/import data entries.	The user can select "Generate ID" to create a unique 16-digit volunteer identification number. The fields First Name, Last Name, City, State, and Phone must be populated for the algorithm to produce an ID number, otherwise the user will get an error message.
	Module 2: 1.4.1			
	Module 1: 1.5.1			
	Module 1: 1.6.1 Module 2: 1.4.1			

MODULE 2					
CR Requirement #	Design Document #	Design Script/Code/Macro	Test (Use) Case Scenario	Test Description	
2.2 The system shall allow modification of imported batch data.	Module 2: 1.1.1		0024- Additional volunteer user information details can be added to the upload/import process or manually.	User can add details to uploaded data.	
	Module 2: 1.2.1				
	Module 2: 1.3.1	Module 2 - TEST_GenUniqueKey_frm field properties.		Any data entered on the 'Add New Volunteer Master Information' form AFTER a unique Member ID was generated is NOT entered into the permanent tables.	
	Module 2: 1.4.1				
	Module 2: 1.5.1				
	Module 2: 3.1.1				HOWEVER, any data entered on the Auto-Pop forms IS SAVED in permanent tables even after a unique Member ID is generated.
	Module 2: 3.2.1				
	Module 2: 3.3.1				
	Module 2: 3.5.1	Module 2 – Show Time Instance button VISUAL BASIC code.			
	Module 2: 3.7.1				
	Module 2: 3.4.1	Module 2 – Edit Available Time button VISUAL BASIC code			
	Module 2: 3.7.1				
	Module 2: 4.1.1				
	Module 2: 4.2.1				
2.3 The system shall allow look up all Volunteer data.	Module 2: 4.2.1		0025- There are separate look-up interfaces for volunteers and management.	Users can look up volunteer data in the Volunteer System.	
	Module 2: 4.6.1				
	Module 2: 4.3.1				
	Module 2: 4.4.1				
2.4 The system will provide different views for volunteer users and management. Volunteers should have read-only access to limited	Module 2: 4.5.1			Volunteer information can be added, changed, and/or deleted by management once it is entered or uploaded and saved in the Volunteer System by using interfaces in the dbMGMT database.	
	Module 2: 4.6.1				
	Module 2: 4.7.1				
	Module 2: 5.1.1				
	Module 2: 5.12.1				
	Module 2: 5.4.1	Module 2 - Card Reader Logic VB code.			

information (TBD by Mgmt). Management should have admin access to all data.	Module 2: 5.5.1	Module 2 - Log IN Validation VB code.		
	Module 2: 5.8.1			
2.5 The system shall capture login and logout times for each volunteer.	Module 2: 5.3.1	Module 2 - Swipe Entry Criteria VB code.	0026- The user can login the Volunteer Mgmt DB using a swipe card.	A volunteer with an encoded mag card can swipe a login.
	Module 2: 5.5.1	Module 2 - Log IN Validation VB code.		A user cannot log IN twice in a row.
	Module 2: 6.1.1			
	Module 2: 6.2.1			
	Module 2: 6.3.1			
	Module 2: 6.4.1			
	Module 2: 6.5.1			
	Module 2: 5.5.1		0027 - The user can login the Volunteer Mgmt DB by typing their Member ID number (no card).	A volunteer without an encoded mag card can enter a login.
	Module 2: 5.6.1	Module 2 - Log OUT Validation VB code.		
	Module 2: 5.7.1	Module 2 - Total Hours Worked VB code.		
	Module 3: 1.2.1 and 2.1.1	Module 3 - VolunteerTimeInstanceTable_qry SQL.		
	Module 3: 2.2.1, 2.3.1, and 2.4.1	Module 3 - VolunteerTimeCalculation_qry SQL. Module 3 - VolunteerTimeCalculation2_qry SQL. Module 3 - VolunteerTimeCalculationTable_qry SQL.		
	Module 2: 5.6.1	Module 2 - Log OUT Validation VB code.	0028- The user can logout of the Volunteer Mgmt DB using a swipe card.	A volunteer with an encoded mag card can swipe a <u>logout</u> .
		A user cannot log OUT twice in a row.		

MODULE 3		
CR Requirement #	Design Document #	Design Script/Code/Macro
3.1 The system shall capture, calculate, track, and store volunteered time.	Module 3: 1.1.1	
	Module 3: 1.2.1	
	Module 3: 2.1.1	Module 2 – Total Hours Worked VISUAL BASIC code
	Module 3: 2.1.1	Module 3 – VolunteerTimeInstanceTable_qry SQL.
	Module 3: 2.2.1	Module 3 – VolunteerTimeCalculation_qry SQL.
	Module 3: 2.3.1	Module 3 – VolunteerTimeCalculation2_qry SQL
	Module 3: 2.4.1	Module 3 – VolunteerTimeCalculationTable_qry SQL
	Module 3: 2.5.1	

MODULE 4				
CR Requirement #	Design Document #	Design Script/Code/Macro	Test (Use) Case Scenario	Test Description
4.1 The system shall produce robust reporting	Module 4: 2.1.1		0049- Robust reporting on any attribute of volunteer data housed in the Volunteer System is available.	Formatted reporting is available for users to retrieve volunteer work time data based on entered criteria and summary options. The available criteria are by date range, by project, by site, by city, or by volunteer. The output is summarized by date, by volunteer, and by project and sorted alphabetically by volunteer's last name. Formatted reporting can be easily exported by a user into Excel or Word documents for quick manipulation and reporting.
	Module 4: 2.3.1	Module 4 – QRY_CreateReportReference SQL.		
4.2 The system shall produce reporting with the following option parameters: Volunteer Date Range Report that shows the same information on the original report, but allows me to view by project, by site, by city, or by person. Sub-alphabetized by person's last name.	Module 4: 2.4.1, 2.5.1 and 2.6.1 a	Module 4 - QRY_CompiledReport SQL. Module 4 – RPT_DateRange Module 4 - Calculating above 24 hours SQL Module 4 - MCR_DateRangeALLRunReport in VB.		
	Module 4: 2.6.1 b	Module 4 - MCR_DateRangeFILTERRunReport in VB.		
	Module 4: 2.6.1 c	Module 4 - MCR_ProjectFILTERRunReport in VB.		
	Module 4: 2.6.1 d	Module 4 - MCR_OrgSiteFILTERRunReport in VB.		
	Module 4: 2.6.1 e	Module 4 - MCR_CityFILTERRunReport in VB.		
4.3 The system shall provide reporting of all volunteer data by user level. Volunteer users can report on any read-only interface or form to which they have access. Mgmt can report on any interface or form.	Module 4: 1.1.1			Any retrieved data on all forms and can be easily exported by a user into Excel or Word documents for quick manipulation and reporting.
	Module 4: 1.2.1			
	Module 4: 1.3.1			
	Module 4: 1.4.1 and 1.5.1			
	Module 4: 1.6.1			

MODULE 5				
CR Requirement #	Design Document #	Design Script/Code/Macro	Test (Use) Case Scenario	Test Description
CR# 1-4	Module 5: 1.1.1	CR#1-4 QRY_CreateReportReference SQL	0050- Change formatted reporting to include "Organization Group"	The report RPT_DateRange will be enhanced to include the field "Organization Group" as part of the volunteer name header.
CR# 2-4 CR# 3-4	Module 5: 1.1.2 Module 5: 1.1.3	CR#2-4 QRY_CompileReport SQL CR#3-4 uFRM_CreateReport VISUAL BASIC		
CR# 4-4	Module 5:1.1.4			
CR# 1-2	Module 5: 2.1.1	CR#1-2 mFRM_AutoDisplay VISUAL BASIC	0051 - Change the sequence of login and log-out user messages and pop-up screens:	Login sequence with no Checked Records.
CR#2-2	Module 5: 2.1.2	CR#1-2 mFRM_AutoDisplay VISUAL BASIC		Login sequence with no Checked Records.
CR# 3-2	Module 5: 2.1.3	CR#1-2 mFRM_AutoDisplay VISUAL BASIC		Login sequence when logging twice in a row.
CR#4-2	Module 5: 2.1.4	CR#4-2 mFRM_AutoDisplay VISUAL BASIC		Logout sequence.
CR#5-2	Module 5: 2.1.5	CR#4-2 mFRM_AutoDisplay VISUAL BASIC		Logout twice in a row.

Risk Matrix:

Date Found	Risk Identified	Risk Score	Risk Response	Requirement/Control	Req/Assum Tracking	Issues/Actions	Issue Status
12/08/2003	The available NLP resources have been reduced to 1 student - thus timeline from project proposal must change to avoid jeopardizing entire project.	0.81	Mitigation	Timeline increased by six months to Dec 2004.	Timeline increased for Planning/Developing Phases.		Closed
02/13/2004	There are problems with the desired functionality - specifically with the time calculation - I need assistance from and expert/consultant to remain on track with deliverables	0.63	Mitigation	Will consult with outside resources	Consulting with Pei-Keng Foong from Database practicum and Patrick Clancy.	7-9-04: This risk was accepted with an extension of the timeline for research. Consultation produced alternative programming solutions which are not feasible with the resource constraints and post-implementation support. This project should not be restarted without considerable impact to timeline and business value.	Closed
08/01/2004	The business is not able to test upload functionality due to issues with GiftMaker PRO. This is a risk to timeline and dependent functionality.	0.81	Mitigation	Will deploy without UAT	Project Plan	8/1/04: This risk was accepted with an extension of the timeline for testing. UAT for this module will occur after other modules are developed.	Closed
Risk Score for a Specific Risk							
Probability	Risk Score= P X I						
0.9	0.09	0.45	0.81				
0.7	0.07	0.35	0.63				
0.5	0.05	0.25	0.45				
0.3	0.03	0.15	0.27				
0.1	0.01	0.05	0.09				
	0.1	0.5	0.9				
Impact of Risk Category on Project							

Appendix D: Volunteer System Design Document

Author: Desirea Ulibarri
Author Position: Project Manager
Date: February 9, 2005

Version: 5.0 Final

SOLUTION DESIGN

The Volunteer System is designed as a database with one back-end repository and two front-end engines with graphical user interfaces. This document details the designed additions, changes, and deletions to the existing Volunteer System. The original system will be referred to as Legacy Volunteer System (LVS).

The Volunteer System is a Microsoft Access Database Management System (DBMS or DB for short) comprised of three .mdb instances: fmVM is the “front-end” housing all volunteer user graphical user interfaces (GUIs), VISUAL BASIC program code, macro scripts, and Structured Query Language statements (SQL); dbMGMT is another “front-end” engine which houses the management GUIs, and dbVM is the “back-end” housing all raw table data.

This document outlines the information contained in the Volunteer System, the detailed design of each functional module, and use case scenarios of each module. The use case scenarios serve to describe the design details from a user-friendly perspective, outlining user and systems actions, responses, inputs and outputs. By describing the use cases of each module, the design document can be easily mapped back to business requirements for traceability as well as mapped forward to User Acceptance test cases.

There is no user-level security on either of the front-end or back-end instances. Any volunteer can access the fmVM database at any time. The repository dbVM and dbMGMT both require database passwords.

Addenda to this document are full-length texts of new or altered VISUAL BASIC program code, macros, and Structured Query Language statements (SQL). After completion of the Volunteer System project, all new and changed code, macros, statements can be used, published, or duplicated by Regis NLP as per the agreement terms in the Volunteer System project proposal.

The intended audience of this document is management of Project CURE. This document is an assigned deliverable of the Planning phase as outlined in the Volunteer System Project Plan. Any changes to this document prior to final delivery are owned and managed by Desirea Ulibarri.

Information Design

The Volunteer System is Project CURE's production database of volunteer information, identification, and work time.

The Volunteer System provides:

1. The capability to upload/import volunteer data in batches or by manual data entry from GiftMaker PRO extracts.
2. Mechanisms to assign and track a unique identification number that can be encoded on a swipe card for swiping an electronic card reader device.
3. Tracking of volunteered time (logins/logouts) and reporting of hours worked.
4. Robust reporting and volunteer lookup functionality.
5. A repository of all volunteer personal and professional data.

Information belonging in the Volunteer System includes:

1. Personal and professional volunteer information
 - Contact information, such as: first, middle, and last name, address, city, state, email addresses, phone numbers.
 - Category information, such as: work skills, leisure interests, work title, work location, availability.
2. Volunteer work information
 - Volunteered time.
 - Volunteer work sites.
 - Volunteer projects.

Design Summary

The Volunteer System is designed as five modules of development. Each module will be created in separate iterations of design, development, and testing. Functionality will be implemented and maintained in a prototype. Migration to a live production database will take place at the completion of the Volunteer System project.

The design modules are:

1. Batch Import Functionality
 - There will be a macros, new SQL statements, and new and altered VISUAL BASIC code added to the existing front-end database **fmVM** for importing of volunteer data from GiftMaker PRO extract files.
2. Volunteer User Interface Enhancements
 - There will sections of new and altered VISUAL BASIC code added the existing front-end database **fmVM** to support new and changed graphical user interfaces (Access forms).
 - There will new forms and changed forms added the existing front-end database **fmVM** to support input of data to all back-end tables.
 - All forms and queries will be replicated as part of the management GUIs in the second front-end database **dbMGMT**.
3. Volunteer Work Time Tracking Additions
 - Enhanced swipe form that will not only capture a Volunteer ID number but will also capture login and logout time.
 - There will be new SQL statements added to the existing front-end database **fmVM** for Volunteer work time table population, calculation, and updates.
4. Volunteer Reporting
 - There will be new SQL statements and VISUAL BASIC code added to the existing front-end database **fmVM** to support reporting.
5. Change Requests to New Volunteer System – *These enhancements will not be developed until after the production deployment of the Volunteer System Project: post – December 2005.*
 - Change formatted reporting to include “**Organization Group**”.
 - Change sequence of login/logout user messages and pop-up screens.

Repository Changes:

Repository changes for the Volunteer System include: new tables and table field additions to both the front-end and back-end database engines. The back-end engine **dbVM** contains all permanent tables while **fmVM** contains all the queries and only those tables housing temporary information such as imported data and work time calculations. These changes are identified in Appendix 1. **dbMGMT** contains the same queries and forms as **fmVM**. There are only two existing tables native to **dbMGMT** and are identified, as are all other tables, in the data dictionary in Appendix 3.

Design Detail with Use Cases

Module 1: Batch Import Functionality

Batch Import/Data Entry: Data will enter the Volunteer System by two methods:

1. Data can be imported into the Volunteer System in batches from an Excel, comma-delimited, .CSV, XML, or text file or another database table. The required fields for the import file are located at the end of this document. *See Module 1 – Import File Specs.*¹
 - 1.1 The user opens the database fmVM and the Switchboard form opens automatically.
 - 2.1 The button “[Import File from GiftMaker PRO](#)” is an available option on the Switchboard. Once a user selects this option, the import macro is triggered. *See Module 1 – Import Macro in VISUAL BASIC.*²
 - 3.1 The import script is a macro that first prompts the user to browse and select a file for importing, designate column headings, name the imported file, select whether to create a new table or append to an existing one, and opens [TEST_GenUniqueKey_frm](#) form when upload is finished. The user manual instructs to save the file to a new table, name it “[ImportTemp](#)” and overwrite any existing table. The [ImportTemp](#) table will house imported volunteer information a temporary state until unique Member ID number is assigned
 - 4.1 The [TEST_GenUniqueKey_frm](#) form is populated from the [QRY_ImportTemp](#) query which pulls all data from the newly populated [ImportTemp](#) table. This form retrieves the most recent uploaded data (which has no assigned Member ID numbers). The form opens in edit mode so that the user can add more complete volunteer information or any other missing attribute (if applicable). Until a unique Member ID number is generated and assigned, the user cannot pre-populate any additional forms located as buttons across the form, for example “[Permanent Address](#)”, “[Available Times](#)”, etc. The user manual instructs the user to fill in more master detail and create a Member ID. *See Module 1 – QRY_ImportTemp SQL.*³
 - 5.1 The user can select “[Generate ID](#)” to create a unique, 16-digit Member ID number. The fields “[First Name](#)”, “[Last Name](#)”, “[City](#)”, “[State](#)”, and “[Phone](#)” must be populated for the algorithm to produce an ID number otherwise the user will get an error message. Any missing information can be manually entered. *NOTE: This algorithm is current functionality of LVS and not a new design/developed feature.*
 - 6.1 After a Member ID number is generated, Visual Basic code will automatically parse and push the imported and additional data to populate the [VolunteerMasterInformation](#), [PermanentContact](#), [BusinessContact](#), and [BusinessOccupation](#) tables. The inserted data contains the unique primary key “[RecNo](#)” from GiftMaker PRO as well as the newly assigned Member ID number which now becomes the primary and relational key for all four tables in the Volunteer System. Table settings in the four master tables for “[RecNo](#)” from GiftMaker PRO and the Member ID

should prevent duplicate records from being imported and uploaded multiple times and duplicate numbers from being used as a Member ID. *See Appendix 3 – Volunteer System Data Dictionary with Entity Relationships.*

- 7.1 When the master tables are populated, the [VolunteerMasterInformation](#) table is filled in with basic name and contact information. If the GiftMakerPRO data has an “H” indicating a home address in the “AddrType” field, the [PermanentContact](#) table is populated; if the indicator is “W”, then the [BusinessContact](#) table is populated. The logic may direct insertion into both tables depending on the data. The [BusinessOccupation](#) table is populated if career information is included on the GiftMaker PRO extract files. All of these tables are populated based on the [QRY_ImportTempFilter](#) query. *See Module 1 - mFRM_TESTGenUniqueKey VISUAL BASIC code with embedded SQL statements⁴ and See Module 1 – QRY_ImportTempFilter SQL.⁵*
- 8.1 After a Member ID is assigned, the user may enter additional information by selecting “[Other Volunteer Information](#)”, “[Volunteer Involvement](#)”, or “[Volunteer Interest](#)” categories and clicking on “[Auto Pop Forms](#)”. When the categories “[Permanent Address](#)” and “[Business Address](#)” are chosen, this “[Auto Pop Forms](#)” button which will open, retrieve and pre-populate data in the [vFRM_BussAddr](#) and [vFRM_PermAddr](#) forms for reviewing, adding, or changing address and contact information. The information is saved to the [PermanentContact](#), and [BusinessContact](#) tables. The imported data has been previously saved in these tables based on the Visual Basic code logic and new data is saved when entered on the associated forms.
 - 9.1 The user can select “[Close Form](#)” to exit the interface.
 - 10.1 The [TEST_GenUniqueKey_frm](#) form can be re-entered at any time by selecting the button “[Re-Enter Import Form](#)” from the Switchboard.
 - 11.1 Once the [ImportTemp](#) table record contains a Member ID number and is inserted into the master tables, it can no longer be retrieved by the [QRY_ImportTemp](#) query or the [TEST_GenUniqueKey_frm](#) form.
2. Data can be manually added into the Volunteer System. *NOTE: This is current functionality of LVS and not a new design/developed feature.*
 - 1.1 The user opens the database fmVM and the Switchboard form opens automatically.
 - 2.1 The button “[Add New Volunteer](#)” is an available option on the Switchboard. Once a user selects this option, the [mFRM_GenUniqueKey](#) form is opened in edit mode and able to accept new volunteer information.
 - 3.1 The user inputs volunteer name, city, state, and phone number and can continue to populate more data or select “[Generate ID](#)” to create a unique 16-digit Member ID number.
 - 4.1 The user may input additional information by selecting “[Auto Pop Forms](#)” which will open and pre-populate multiple forms with the new Member ID for entering additional address, contact information, career information,

etc. This additional data is written to tables such as [BusinessContact](#) and [PermanentContact](#).

- 5.1 The user can select “[Close Form](#)” to exit the interface.

Module 2: Volunteer User Interface Enhancements

Volunteer User Interfaces:

- Imported and manually entered data can be used to generate and assign a unique Member ID number that can be encoded on a swipe card and read by an electronic card reader device. Additional volunteer user information details can be added manually at any time.
 - There are separate look-up interfaces for volunteers and management.
 - Volunteers can lookup volunteer data and change their available work days/time.
 - Management can lookup, add, change, or delete data at any time.
 - Volunteers can login and logout by using either an electronic card swipe or manually adding their 16-digit unique identification number.
1. A unique Member ID number can be generated from imported data. After completion of import:
 - 1.1 The user opens the database [fmVM](#) and the [Switchboard](#) form opens automatically.
 - 2.1 The button “[Re-Enter Import Form](#)” is an available option on the [Switchboard](#). Once a user selects this option, the [TEST_GenUniqueKey_frm](#) form is opened in edit mode. The [TEST_GenUniqueKey_frm](#) form is populated from the [QRY_ImportTemp](#) query which pulls all data from the [ImportTemp](#) table. The data is queried by the most recent import date, “[Date Entered](#)”, and only pulls entries that do NOT have an assigned Member ID.
 - 3.1 When the [TEST_GenUniqueKey_frm](#) form is opened, the most recent uploaded data will appear and populate volunteer name, city, state, and phone number. Before a Member ID is generated, a user can add or change data to any volunteer entry regardless of whether it was imported as many of the form fields are mapped to the imported file fields. For example, “[Date Joined](#)” reflects any data in “[Class Date](#)” field from the GiftMaker PRO import files. *See Module 2 - TEST_GenUniqueKey_frm field properties.*⁶
 - 4.1 The user can select “[Generate ID](#)” to create a unique, 16-digit Member ID number. The fields “[First Name](#)”, “[Last Name](#)”, “[City](#)”, “[State](#)”, and “[Phone](#)” must be populated for the algorithm to produce a Member ID number otherwise the user will get an error message. Any missing information can be manually entered.
 - 5.1 Any data entered on the “[TEST_GenUniqueKey_frm](#) form AFTER a Member ID is generated is NOT entered into the master tables. HOWEVER, any data entered on any of the additional forms, such as

“Permanent Address” or “Business Address” IS SAVED in permanent tables even after a Member ID is assigned.

- 6.1 The user can select “Close Form” to exit the interface.
2. A Member ID can be generated from manually entered data. *NOTE: This is current functionality of LVS and not a new design/developed feature.*
 - 1.1 The user opens the database fmVM and the Switchboard form opens automatically.
 - 2.1 The button “Add New Volunteer” is an available option on the Switchboard. Once a user selects this option, the mFRM_GenUniqueKey form is opened in edit mode and able to accept new volunteer information.
 - 3.1 The user inputs volunteer name, city, state, and phone number and can continue to populate more data or select “Generate ID” to create a unique 16-digit Member ID number.
 - 4.1 The user may input additional information by selecting “Auto Pop Forms” which will open and pre-populate multiple forms with the new Member ID for entering additional address, contact information, career information, etc. This additional data is written to tables such as BusinessContact and PermanentContact.
 - 5.1 The user can select “Close Form” to exit the interface.
 3. Users can look up volunteer data in the Volunteer System.
 - 1.1 The user opens the database fmVM and the Switchboard form opens automatically.
 - 2.1 The button “Look Up Volunteer Information” is an available option on the Switchboard. Once a user selects this option, the mFRM_VolunteerLookUp form is opened in read-only mode and able to filter and retrieve volunteer information.
 - 3.1 The user can select the filter function button and populate a single field for querying and lookup. Multiple query criteria can also be entered for a more complex search, such as first name and last name combinations. The user executes the filter from the menu bar by selecting the “Apply Filter” button symbol and data is retrieved. Once data is populated, the user has other button options that can be launched from the retrieved data such as volunteer availability and a retrieval of the volunteer’s previous login/logout entries. A user can change their work day/time availability at any time by selecting the button “Available Times”. This button opens the sFRM_VolunteerAvailableTimes form in edit mode. This form is filtering to only show the specific Member ID data from the VolunteerAvailableTimes table. To populate work availability, a user selects criteria from the drop-down menus. *See Module 2 – Edit Available Time button VISUAL BASIC code⁷*
 - 4.1 A user can review the dates/times that they have logged in and out of the Volunteer System by selecting the button “Volunteer Time Instances”. This button opens the sFRM_VolunteerTimeInstance form in read-only mode. This form is filtering to only show the specific Member ID data

from the [VolunteerTimeInstance](#) table. All previous login times will appear with the latest time swipe appearing first. *See Module 2 – Show Time Instance button VISUAL BASIC code.*⁸

- 5.1 The user can also select “[Swipe Time Instance](#)” in order to enter a login or logout. This button opens the [mFRM_AutoDisplay](#) form. Entries onto this form are saved in the [VolunteerTimeInstance](#) table. *See Module 2 – Time Swipe button VISUAL BASIC code.*⁹
 - 6.1 The user can select “[Close Form](#)” to exit the interface.
4. Volunteer information can be added, changed, and/or deleted by management once it is entered or uploaded and saved in the Volunteer System by using interfaces in the [dbMGMT](#) database.
 - 1.1 The user opens the database [dbMGMT](#) and enters the database password and the Switchboard form opens and available with option buttons.
 - 2.1 The button “[Data Definitions for all Tables](#)” opens the [mFRM_DataDefinitionsUtilities](#) form which allows management to add, change or delete all reference table data such as project names and site locations. The following forms are available as option buttons from this form: [sFRM_Projects](#), [uFRM_DefinitionsCity](#), [uFRM_Nomenclature_VolCode](#), [uFRM_OrganizationSites](#), [uFRM_ValidAffiliations](#), [uFRM_ValidAffiliationType](#), [uFRM_ValidAvailableTimes](#), [uFRM_ValidCountry](#), [uFRM_ValidFavorites](#), [uFRM_ValidFavoriteTypes](#), [uFRM_ValidLevelOfInterest](#), [uFRM_ValidLevelOfSkill](#), [uFRM_ValidMaritalStatus](#), [uFRM_ValidPrefix](#), [uFRM_ValidRelationship](#), [uFRM_ValidSpecializedTraining](#), [uFRM_ValidSuffix](#), [uFRM_ValidWorkCategories](#), [uFRM_ZipCodeCityState](#). *Note: These forms are part of the original LVS.*
 - 3.1 The button “[Change Volunteer Information](#)” opens the [mFRM_GenUniqueKey](#) form which allows management to add, change, or delete any volunteer data such as permanent address, work skills, etc. The following forms are available as option buttons from this form: [sFRM_BC](#), [sFRM_BO](#), [sFRM_PA](#), [sFRM_VolunteerAffiliations](#), [sFRM_VolunteerAvailableTimes](#), [sFRM_VolunteerCountryVisited](#), [sFRM_VolunteerFavorites](#), [sFRM_VolunteerSkills](#), [sFRM_VolunteerSpecializedTraining](#), [sFRM_VolunteerWorkPreferences](#), [sFRM_WorkCategories](#). *Note: These forms are part of the original LVS.*
 - 4.1 The button “[Change Volunteer Logged Time](#)” opens the [sFRM_TotalHoursWorked](#) form which allows management to change or delete any volunteer worked time entry. *This is a new form created specifically for the dbMGMT interface.*
 - 5.1 The button “[Swipe Card Utility](#)” opens the [mFRM_DataManagementUtilities](#) form which allows management to create a table of volunteer names and swipe card utility numbers to export for the creation of volunteer swipe cards. The following forms are available as option buttons from this form: [vFRM_BussAddr](#),

vFRM_PermAddr, vFRM_VolMstInf, vFRM_VolunteerMasterInformation. *Note: These forms are part of the original LVS.*

- 6.1 The user can select “Close Form” on all open screens to exit the interfaces.
 - 7.1 The user can select “Quit Volunteer Management” to exit dbMGMT.
5. The user can login or logout of the Volunteer System using a swipe card.
- 1.1 The user opens the database fmVM and the Switchboard form opens automatically.
 - 2.1 The button “Swipe Card” is an available option on the Switchboard. Once a user selects this option, the mFRM_AutoDisplay form is opened. Entries onto this form are saved in the VolunteerTimeInstance table.
 - 3.1 The user enters the volunteer site location and city as well as the project they are currently working. The user will also enter whether they are logging “IN” or “OUT”. If any of the fields are left empty when moving to the next entry, the system will prompt for population before moving to the next field. *See Module 2 – Swipe Entry Criteria VISUAL BASIC code¹⁰.*
 - 4.1 The user populates their Member ID number by swiping a card through an electronic card reader which allows data encoded on the swipe card to automatically enter the form. *NOTE: This is current functionality of LVS and not a new designed/developed feature. See Module 2 – Card Reader Logic VISUAL BASIC code.¹¹*
 - 5.1 The user must then select the “Press After Swipe” button. If the user is logging “IN”, his/her previous entries in the VolunteerTimeInstance table are queried (using DLookup functionality) to ensure the last card swipe was “OUT” If the user attempts to log “IN” twice in a row, the system will produce an error message reminding the user that they have already logged “IN”. The sFRM_VolunteerTimeInstance form opens after the error occurs. If there is no previous conflicting entry, the system will accept the “IN”, create a date/time stamp entry, and save the entry to the VolunteerTimeInstance table with an “IN” indicator. The sFRM_VolunteerTimeInstance form opens after the entry is saved showing the login. *See Module 2 – Log IN Validation VISUAL BASIC code¹²*
 - 6.1 If the user is logging “OUT”, his/her previous entries in the VolunteerTimeInstance table are queried (using DLookup functionality) to ensure the last card swipe was “IN”. If the user attempts to log “OUT” twice in a row, the system will produce an error and prompt the user to change their entry to “IN”. If there is no previous conflicting entry, the system will accept the “OUT” and create a date/time stamp entry in the VolunteerTimeInstance table with an “OUT” indicator. *See Module 2 – Log OUT Validation VISUAL BASIC code.¹³*
 - 7.1 After the time entry is saved, the system launches a series of SQL queries and VISUAL BASIC code to calculate and store work time in the TotalHoursWorked table. *NOTE: This functionality is detailed in Module*

3. The [mFRM_TotalHoursWorked](#) form immediately opens reflecting the entry in the [TotalHoursWorked](#) table, indicating to the user the current hours:minutes worked since they last logged “IN”. The [sFRM_VolunteerTimeInstance](#) form also opens after the entry is saved showing the logout. *See Module 2 – Total Hours Worked VISUAL BASIC code.*¹⁴
 - 8.1 The user can select “Close Form” on all open screens to exit the interfaces.
6. The user can manually type their 16-digit Member ID number to log work time.
 - 1.1 The user opens the database [fmVM](#) and the Switchboard form opens automatically.
 - 2.1 The button “Swipe Card” is an available option on the Switchboard. Once a user selects this option, the [mFRM_AutoDisplay](#) form is opened. Entries onto this form are saved in the [VolunteerTimeInstance](#) table.
 - 3.1 The user enters the volunteer site location and city as well as the project they are currently working. The user will also enter whether they are logging “IN” or “OUT”. If any of the fields are left empty when moving to the next entry, the system will prompt for population before moving to the next field.
 - 4.1 The user populates their Member ID number by typing in all 16 digits. The form defaults contain the proper input mask for the Member ID field, so there is no need to use dashes or any other special characters.
 - 5.1 When a login entry is successfully captured, entered, and saved, the system opens the [sFRM_VolunteerTimeInstance](#) form showing the login. This indicates a successful entry. The validation logic for “IN” still occurs.
 - 6.1 When a logout entry is successfully captured, entered, and saved, the system opens the [mFRM_TotalHoursWorked](#) form. This indicates a successful entry. The validation logic for “OUT” still occurs. The [sFRM_VolunteerTimeInstance](#) form will also open after a successful “OUT” entry is captured.
 - 7.1 The user can select “Close Form” on all open screens to exit the interfaces.

Module 3: Volunteer Work Time Tracking

As volunteered time is logged in the system, it is saved, tracked, and calculated to produce a table of time worked for all volunteers.

1. Volunteer login/logout swipe and manual entries are saved in the [VolunteerTimeInstance](#) table for tracking.
 - 1.1 When a login entry is successfully captured and passes “IN” logic validation, the system saves the entry in the [VolunteerTimeInstance](#) table populating the date and time of the login and an “IN” indicator in the [VTI_TimeDirectionID](#) field. The city, site, and project code are also saved in the record. There is no calculation of work time until the user enters a logout.
 - 2.1 When a logout entry is successfully captured and passes “OUT” logic validation, the system saves the entry in the [VolunteerTimeInstance](#) table populating the date and time of the logout and an “OUT” indicator in the

[VTI_TimeDirectionID](#) field. The city, site, and project code are also saved in the record.

2. Volunteer work time is calculated based on login/logout entries and saved in a table for reporting
 - 1.1 After an “OUT” entry is saved, VISUAL BASIC code (*See Module 2 – Total Hours Worked VISUAL BASIC code*) invokes the [VolunteerTimeInstanceTable_qry](#) query which filters the [VolunteerTimeInstance](#) table for the records of the user who is logging out. This is a make-table query which stores these filtered records in a new table called [tmpTimeInstanceFilter_tbl](#) table. This is a temporary table; new storage overwrites any previously saved data. *See Module 3 – VolunteerTimeInstanceTable_qry SQL.*¹⁵
 - 2.1 After the [VolunteerTimeInstanceTable_qry](#) completes, the VISUAL BASIC code invokes the [VolunteerTimeCalculationTable_qry](#) which performs a calculation based on the output of a series of queries: These queries are performed in order to produce output that is used in the [VolunteerTimeCalculationTable_qry](#): The [VolunteerTimeCalculation_qry](#) takes the records from the [tmpTimeInstanceFilter_tbl](#) and generates a cross-tab output which creates specific fields of the “IN” and “OUT” indicators. This query filters and matches the last “OUT” entry with the correct “IN” entry. *See Module 3 – VolunteerTimeCalculation_qry SQL.*¹⁶
 - 3.1 The output from the [VolunteerTimeCalculation_qry](#) is used by the next query [VolunteerTimeCalculation2_qry](#). This query subtracts the date/time value for “OUT” from the value for “IN” to calculate and create a value for a new [VTI_TotalHoursWorked](#) field. *See Module 3 – VolunteerTimeCalculation2_qry SQL.*¹⁷
 - 4.1 The [VolunteerTimeCalculation2_qry](#) is then used by the [VolunteerTimeCalculationTable_qry](#) which takes the output and appends the data to the [TotalHoursWorked](#) table. In addition, the [VolunteerTimeCalculation2_qry](#) is also the source for the [mFRM_TotalHoursWorked](#) form. *See Module 3 – VolunteerTimeCalculationTable_qry SQL.*¹⁸
 - 5.1 When a user enters a logout and the city, site, and project differ from the corresponding login entry, then only the city, site, and project designations associated with the “OUT” entry will be saved to the [TotalHoursWorked](#) table.

Module 4: Volunteer Reporting

Robust reporting on any attribute of volunteer data housed in the Volunteer System is available.

1. Any retrieved data on all forms and can be easily exported by a user into Excel or Word documents for quick manipulation and reporting.
 - 1.1 The user opens the database [fmVM](#) and the [Switchboard](#) form opens automatically.

- 2.1 The button “[Look Up Volunteer Information](#)” is an available option on the Switchboard. Once a user selects this option, the [mFRM_VolunteerLookUp](#) form is opened in read-only mode and able to filter and retrieve volunteer information.
 - 3.1 The user can select the filter function button and populate a single field for querying and lookup. Multiple query criteria can also be entered for a more complex search, such as first name and last name combinations. The user executes the filter from the menu bar by selecting the “[Apply Filter](#)” button symbol and data is retrieved.
 - 4.1 Once data is retrieved on this form, the user can select other options that will retrieve the filtered/selected volunteer data, such as previous login/logout time.
 - 5.1 A user can review the dates/times that they have logged in and out of the Volunteer System by selecting the button “[Volunteer Time Instances](#)”. This button opens the [sFRM_VolunteerTimeInstance](#) form in read-only mode. This form is filtering to only show the specific Member ID data from the [VolunteerTimeInstance](#) table. All previous login times will appear with the latest time swipe appearing first.
 - 6.1 The user can select either “[Office Links](#)” or “[Export](#)” from the main file menu at the top of the form to export the filtered data to either Word or Excel. “[Office Links](#)” will immediately open a document and populate the new data, while “[Export](#)” will first prompt the user to save the data to a disk location. Either method is available for saving the data to a preferred user location.
 - 7.1 Any data on any user form is available for export to Word or Excel using these file menu options.
 - 8.1 The user can select “[Close Form](#)” to exit the interface.
2. Formatted reporting is available for users to retrieve volunteer work time data based on entered criteria and summary options. The available criteria are by date range, by project, by site, by city, or by volunteer. The output is summarized by date, by volunteer, and by project and sorted alphabetically by volunteer’s last name. Formatted reporting can be easily exported by a user into Excel or Word documents for quick manipulation and reporting.
 - 1.1 The user opens the database [fmVM](#) and the Switchboard form opens automatically.
 - 2.1 The button “[Create Reports](#)” is an available option on the Switchboard. Once a user selects this option, the [uFRM_CreateReport](#) form is opened.
 - 3.1 The [uFRM_CreateReport](#) is built from the [QRY_CreateReportReference](#) query which joins the [TotalHoursWorked](#), [VolunteerMasterInformation](#), [OrganizationSites](#), [Definitions_City](#), [Projects](#), and [BusinessContact](#) tables. Thus any field or category available on this query can be an attribute on the [uFRM_CreateReport](#) form. Only those volunteers, projects, sites, and cities that have volunteer work hours logged will be retrieved by this query. *See Module 4 – QRY_CreateReportReference SQL.*¹⁹

- 4.1 This form will invoke the [RPT_DateRange](#) which is compiled from the output of the [QRY_CompileReport](#) query. This query is filtered based on criteria selection however each reporting option will include the selected date range. *QRY_CompileReport SQL*.²⁰
- 5.1 The [RPT_DateRange](#) report is alphabetized by volunteer last name and categorizes work hours by each member, date, and project category. Work hours are totaled and displayed for each day, for the total date range (which is displayed in the upper right-hand corner), and for the entire report. The work hour totals for each date are displayed in hours:minutes. An SQL statement in the control source for the totals field converts numerical time into short time. *See Module 4 – RPT_DateRange*²¹ and *Calculating above 24 hours SQL*.²²
- 6.1 A user can select different tabs at the top of the form that represent the available reporting criteria. The tab “[Volunteer Work Time by Date Range](#)” allows Step 1) input of a date range, Step 2) the option to run the report for all volunteers OR Step 3) select alternate criteria on the other tabs.
 - a The date range fields are unbound and default to the current date automatically when the form is opened. When the user clicks the button on this page “[Step 2\) Report for All Volunteers using this date range - Click Here:](#)” Visual Basic code opens the [RPT_DateRange](#) populated by [QRY_CompileReport](#) with no filters other than date range. *See Module 4 - MCR_DateRangeALLRunReport in VISUAL BASIC*.²³
 - b The report is filtered by volunteer name when the user selects the tab “[By Volunteer](#)” and chooses an available volunteer from the drop-down menu. When the user clicks the button “[and Click Here](#)”, Visual Basic opens the report with this filter. *See Module 4 - MCR_DateRangeFILTERRunReport in VISUAL BASIC*.²⁴
 - c The report is filtered by project when the user selects the tab “[By Project](#)” and chooses an available project from the drop-down menu. When the user clicks the button “[and Click Here](#)”, Visual Basic opens the report with this filter. *See Module 4 - MCR_ProjectFILTERRunReport in VISUAL BASIC*.²⁵
 - d The report is filtered by organization site when the user selects the tab “[By Organization Site](#)” and chooses an available site from the drop-down menu. When the user clicks the button “[and Click Here](#)”, Visual Basic opens the report with this filter. *See Module 4 - MCR_OrgSiteFILTERRunReport in VISUAL BASIC*.²⁶
 - e The report is filtered by city when the user selects the tab “[By City](#)” and chooses an available city from the drop-down menu. When the user clicks the button “[and Click Here](#)”, Visual Basic opens the report with this filter. *See Module 4 - MCR_CityFILTERRunReport in VISUAL BASIC*.²⁷
- 7.1 The user can select either “[Office Links](#)” or “[Export](#)” from the main file menu at the top of the form to export the filtered data to either Word or

Excel. “Office Links” will immediately open a document and populate the new data, while “Export” will first prompt the user to save the data to a disk location. Either method is available for saving the data to a preferred user location.

- 8.1 The user can select “Close Form” to exit the interface.

Module 5: Change Requests to New Volunteer System

The change requests to the above design are:

1. Change formatted reporting to include “Organization Group”:
 - 1.1 The query `QRY_CreateReportReference` will be enhanced to include the table `BusinessContact`. See *CR#1-4 QRY_CreateReportReference SQL*.²⁸
 - 1.2 The query `QRY_CompileReport` will be enhanced to include the table `BusinessContact` See *CR#2-4 QRY_CompileReport SQL*.²⁹
 - 1.3 The form `uFRM_CreateReport` will be changed with an additional page tab with the title “By Organization Group”. The page will allow the current reporting described in Module 4 to be filtered by “Organization Group”. See *CR#3-4 uFRM_CreateReport VISUAL BASIC*.³⁰
 - 1.4 The report `RPT_DateRange` will be enhanced to include the field “Organization Group” as part of the volunteer name header.
2. Change the sequence of login/logout user messages and pop-up screens:
 - 1.1 When a user enters a login, a new message “Thank you for logging IN!” should appear with no other screen openings.
 - 1.2 When a user enters a login and selects “Check Records”, a new message “Thank you for logging IN!” should appear and the form `mFRM_VolunteerLookUp` should open and present data about the volunteer who is logging in. No other screen openings should occur.
 - 1.3 When a user enters a login twice in a row, the system will produce an error message reminding the user that they have already logged “IN”. No other screen openings should occur. See *CR#1-2 mFRM_AutoDisplay VISUAL BASIC*.³¹
 - 1.4 When a user enters a logout, the existing message “Thank you for volunteering xx:xx hours/minutes!”. No other screens should open.
 - 1.5 When a user enters a logout twice in a row the existing error message should appear with no other screen openings. See *CR#4-2 mFRM_AutoDisplay VISUAL BASIC*.³²

APPENDIX 1 – Visual Basic code, SQL statements, and file specifications

¹ Module 1 – ImportTemp file specs:

Columns

Name	Type	Size
ID	Long Integer	4
Seq	Double	8
RecNo	Double	8
RecType	Text	255
Person	Double	8
Prefix	Text	255
FirstName	Text	255
MI	Text	255
LastName	Text	255
Suffix	Text	255
Salutation	Text	255
Prefix2	Text	255
FirstName2	Text	255
MI2	Text	255
LastName2	Text	255
Suffix2	Text	255
Salutation2	Text	255
BothName	Text	255
SalutationBoth	Text	255
JobTitle	Text	255
Organization	Text	255
AddrType	Text	255
Address1	Text	255
Address2	Text	255
City	Text	255
State	Text	255
Zip	Double	8
Country	Text	255
Phone1	Text	255
Extn1	Text	255
Phone2	Text	255
Extn2	Text	255
Fax	Text	255
BirthDate	Text	255
UserDef1	Text	255
UserDef2	Text	255
UserDef3	Text	255
UserDef4	Text	255
UserDate	Text	255
UserCode	Text	255
ClassYr	Text	255
Solicitor	Text	255
Classification	Text	255
ClassDate	Date/Time	8
ClassInfo	Text	255
Member	Text	255
Joined	Text	255
Renewed	Text	255
ToExpire	Text	255
Expired	Text	255
Pub_Name	Text	255
Del_Pt	Text	255
Email1	Text	255
Email2	Text	255
Web	Text	255
MbrLevel	Double	8
MbrGroup	Text	255
MbrID	Double	8
MbrSince	Text	255

² Module 1 – Import Macro in VISUAL BASIC:

```

Function ImportTempTbl_mcr()
On Error GoTo ImportTempTbl_mcr_Err
DoCmd.SetWarnings False
' This command will import the table, and prompt where and what to save as.
DoCmd.RunCommand acCmdImport
' This form is run from the 'ImportTemp' table.
DoCmd.OpenForm "TEST_GenUniqueKey_frm", acNormal, "", "", acEdit, acNormal
ImportTempTbl_mcr_Exit:
Exit Function
ImportTempTbl_mcr_Err:
MsgBox Error$
Resume ImportTempTbl_mcr_Exit
End Function

```

³ Module 1 - QRY_ImportTemp SQL:

```

SELECT * FROM ImportTemp WHERE Member is Null;

```

⁴ Module 1 - mFRM TESTGenUniqueKey VISUAL BASIC code with embedded SQL statements:

```

DoCmd.RunCommand acCmdSaveRecord
DoCmd.SetWarnings False
' This statement will insert the necessary fields into the VolunteerMasterInformation and
BusinessOccupation tables.
DoCmd.RunSQL "INSERT INTO VolunteerMasterInformation (VMI_MemberID,
VMI_NamePrefix, VMI_NameLast, VMI_NameMiddle, VMI_NameFirst, VMI_NameSuffix,
VMI_DateJoined, VMI_BirthDate, VMI_MaritalStatus, VMI_VolCode, RecNo) SELECT
Member, Prefix, LastName, MI, FirstName, Suffix, ClassDate, ClassYr, BothName, UserCode,
RecNo FROM QRY_ImportTempFilter; ", -1
DoCmd.RunSQL "INSERT INTO BusinessOccupation (BO_MemberID, BO_JobTitle,
RecNo) SELECT Member, JobTitle, RecNo FROM QRY_ImportTempFilter; ", -1
' This statement will dictate how BusinessContact or Permanent Contact tables will be
populated
LookupVariable = DLookup("[AddrType]", "QRY_ImportTempFilter")
If LookupVariable = "W" Then
DoCmd.RunSQL "INSERT INTO BusinessContact (BC_MemberID, BC_OrgName,
BC_AddrLine1, BC_AddrLine2, BC_AddrCity, BC_AddrState, BC_AddrZipCode,
BC_AddrCountry, BC_ConPhone, BC_ConCell, BC_ConFax, BC_ConEmail, RecNo) SELECT
Member, Organization, Address1, Address2, City, State, Zip, Country, Phone2, Extn2, Fax,
Email2, RecNo FROM QRY_ImportTempFilter; ", -1
DoCmd.RunSQL "INSERT INTO PermanentContact (PC_MemberID, PC_ConPhone,
PC_ConCell, PC_ConEmail, RecNo) SELECT Member, Phone1, UserDef2, Email1, RecNo
FROM QRY_ImportTempFilter; ", -1
Else: DoCmd.RunSQL "INSERT INTO PermanentContact (PC_MemberID,
PC_AddrLine1, PC_AddrLine2, PC_AddrCity, PC_AddrState, PC_AddrZipCode,
PC_AddrCountry, PC_ConPhone, PC_ConCell, PC_ConEmail, RecNo) SELECT Member,
Address1, Address2, City, State, Zip, Country, Phone1, UserDef2, Email1, RecNo FROM
QRY_ImportTempFilter; ", -1
DoCmd.RunSQL "INSERT INTO BusinessContact (BC_MemberID, BC_OrgName,
BC_ConPhone, BC_ConCell, BC_ConFax, BC_ConEmail, RecNo) SELECT Member,
Organization, Phone2, Extn2, Fax, Email2, RecNo FROM QRY_ImportTempFilter; ", -1
End If

' Reset the CheckDone for the next set of data
Forms!TEST_GenUniqueKey_frm![CheckDone] = 0
Exit_CheckDone_Click:

```

Exit Sub

⁵ Module 1 – QRY ImportTempFilter SQL:

```
SELECT *
FROM ImportTemp
WHERE (((ImportTemp.Member)=Forms!TEST_GenUniqueKey_frm!VMI_MemberID));
```

⁶ Module 2 - TEST_GenUniqueKey_frm field properties:

⁷ Module 2 – Edit Available Time button VISUAL BASIC code:

```
Private Sub Button_EditAvaiTime_Click()
On Error GoTo Err_Button_EditAvaiTime_Click
Dim stDocName As String
Dim stLinkCriteria As String
stDocName = "sFRM_VolunteerAvailableTimes"
stLinkCriteria = "[VAT_MemberID]=" & """" & Me![VMI_MemberID] & """"
DoCmd.OpenForm stDocName, , , stLinkCriteria
If IsNull(Forms!sFRM_VolunteerAvailableTimes![VAT_MemberID]) Then
Forms!sFRM_VolunteerAvailableTimes![VAT_MemberID] = Me![VMI_MemberID]
End If
Exit_Button_EditAvaiTime_Click:
Exit Sub
Err_Button_EditAvaiTime_Click:
MsgBox Err.Description
Resume Exit_Button_EditAvaiTime_Click
End Sub
```

⁸ Module 2 – Show Time Instance button VISUAL BASIC code:

```

Private Sub Button_ShowTimeInst_Click()
On Error GoTo Err_Button_ShowTimeInst_Click
    Dim stDocName As String
    Dim stLinkCriteria As String
    stDocName = "sFRM_VolunteerTimeInstance"
    stLinkCriteria = "[VTI_MemberID]=" & "" & Me![VMI_MemberID] & ""
    DoCmd.OpenForm stDocName, , , stLinkCriteria
Exit_Button_ShowTimeInst_Click:
    Exit Sub
Err_Button_ShowTimeInst_Click:
    MsgBox Err.Description
    Resume Exit_Button_ShowTimeInst_Click
End Sub

```

⁹ Module 2 - Time Swipe button VISUAL BASIC code:

```

Private Sub CommandManualTimeSwipe_Click()
On Error GoTo Err_CommandManualTimeSwipe_Click

    Dim stDocName As String
    Dim stLinkCriteria As String

    stDocName = "mFRM_AutoDisplay"

    stLinkCriteria = "[VTI_MemberID]=" & "" & Me![VMI_MemberID] & ""
    DoCmd.OpenForm stDocName, , , stLinkCriteria

    If IsNull(Forms!mFRM_AutoDisplay![VTI_MemberID]) Then
        Forms!mFRM_AutoDisplay![VTI_MemberID] = Me![VMI_MemberID]
    End If

Exit_CommandManualTimeSwipe_Click:
    Exit Sub

Err_CommandManualTimeSwipe_Click:
    MsgBox Err.Description
    Resume Exit_CommandManualTimeSwipe_Click

End Sub

```

¹⁰ Module 2 – Swipe Entry Criteria VISUAL BASIC code:

```

Private Sub Button_EnterTimeSwipe_Click()
    Dim db As Database
    Dim strSQL As String
    Dim currdate As Date
    Dim CardReader As String
    Dim CardID As String
    Dim FirstChar As String
    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim stTimeCalcForm As String
    currdate = Now
    Set db = DBEngine(0)(0)

```

```

If IsNull(Me.CityCode) Then
    MsgBox "Please Enter CityCode"
    Me.CityCode.SetFocus
ElseIf IsNull(Me.SiteID) Then
    MsgBox "Please Enter Site ID"
    Me.SiteID.SetFocus
ElseIf IsNull(Me.TimeDirectionID) Then
    MsgBox "Please indicate whether you are swiping IN or OUT"
    Me.TimeDirectionID.SetFocus
ElseIf IsNull(Me.VTI_MemberID) Then
    MsgBox "Please enter Member ID or swipe card"
    Me.VTI_MemberID.SetFocus
Else

```

¹¹ Module 2 – Card Reader Logic VISUAL BASIC code:

```

CardReader = Me.VTI_MemberID
FirstChar = Mid(CardReader, 1, 1)

    If FirstChar = "%" Then
        CardID = Mid(CardReader, 3, 4) + "-" + Mid(CardReader, 7, 4) + "-" +
Mid(CardReader, 11, 4) + "-" + Mid(CardReader, 15, 4)
    ElseIf FirstChar = ";" Then
        CardID = Mid(CardReader, 2, 4) + "-" + Mid(CardReader, 6, 4) + "-" +
Mid(CardReader, 10, 4) + "-" + Mid(CardReader, 13, 4)
    Else: CardID = Mid(CardReader, 1, 19)
    End If

    If IsNull(CardID) Then
        MsgBox "Invalid/Non-Existant Member ID"
        Me.VTI_MemberID.SetFocus

```

¹² See Module 2 – Log IN Validation VISUAL BASIC code:

```

ElseIf Me.TimeDirectionID = "IN" Then
    LookupVariable = DLast("VTI_TimeDirectionID", "VolunteerTimeInstance",
"[VTI_MemberID] = " & CardID & """)
    If LookupVariable = "IN" Then
        MsgBox "You have already logged IN."
        Me.TimeDirectionID.SetFocus
        stDocName = "sFRM_VolunteerTimeInstance"
        stLinkCriteria = "[VTI_MemberID]=" & """" & CardID & """"
        DoCmd.OpenForm stDocName, , , stLinkCriteria
    Else
        strSQL = "INSERT INTO VolunteerTimeInstance values (" & CardID & ", " &
Me.CityCode & ", " & Me.SiteID & ", " & Me.ProjectID & ", " & currdate & ", " &
Me.TimeDirectionID & ");"
        db.Execute (strSQL)
        db.Close
        stDocName = "sFRM_VolunteerTimeInstance"
        stLinkCriteria = "[VTI_MemberID]=" & """" & CardID & """"
        DoCmd.OpenForm stDocName, , , stLinkCriteria
        ' Check if LookUp records is checked
        If Me.Check_VI_Lookup = -1 Then
            stDocName = "mFRM_VolunteerLookUp"
            stLinkCriteria = "[VMI_MemberID]=" & """" & CardID & """"
            DoCmd.OpenForm stDocName, , , stLinkCriteria

```



```

    End If
  End If

```

¹³ Module 2 – Log OUT Validation VISUAL BASIC code:

```

Elseif Me.TimeDirectionID = "OUT" Then
  LookupVariable = DLast("VTI_TimeDirectionID", "VolunteerTimeInstance",
    "[VTI_MemberID] = " & CardID & """)
  If LookupVariable = "OUT" Then
    MsgBox "You have already logged OUT."
    Me.TimeDirectionID.SetFocus
    stDocName = "sFRM_VolunteerTimeInstance"
    stLinkCriteria = "[VTI_MemberID]=" & "" & CardID & ""
    DoCmd.OpenForm stDocName, , , stLinkCriteria
  Else
    strSQL = "INSERT INTO VolunteerTimeInstance values (" & CardID & ", " & "" &
      Me.CityCode & ", " & "" & Me.SiteID & ", " & "" & Me.ProjectID & ", " & "" &
      Me.TimeDirectionID & ");"
    db.Execute (strSQL)
    db.Close
  End If

```

¹⁴ Module 2 - Total Hours Worked VISUAL BASIC code:

```

    stDocName = "VolunteerTimeCalculationTable_qry"
    stTimeCalcForm = "mFRM_TotalHoursWorked"
    stLinkCriteria = "[VTI_MemberID]=" & "" & CardID & ""
    DoCmd.SetWarnings False
    DoCmd.OpenForm "sFRM_VolunteerTimeInstance", , , stLinkCriteria
    DoCmd.OpenQuery "VolunteerTimeInstanceTable_qry", acNormal, acEdit
    DoCmd.OpenQuery stDocName
    DoCmd.OpenForm stTimeCalcForm
    stDocName = "mFRM_AutoDisplay"
    DoCmd.Close acForm, stDocName
  End If
End If
End Sub

```

¹⁵ Module 3 – VolunteerTimeInstanceTable qry SQL:

```

SELECT VolunteerTimeInstance.VTI_MemberID, VolunteerTimeInstance.VTI_CityCode,
  VolunteerTimeInstance.VTI_SiteID, VolunteerTimeInstance.VTI_ProjectID,
  VolunteerTimeInstance.VTI_Instance, VolunteerTimeInstance.VTI_TimeDirectionID INTO
  tmpTimeInstanceFilter_tbl
FROM VolunteerTimeInstance
WHERE
  (((VolunteerTimeInstance.VTI_MemberID)=Forms!sFRM_VolunteerTimeInstance!VTI_Member
  ID));

```

¹⁶ Module 3 – VolunteerTimeCalculation qry SQL:

```

TRANSFORM Last(tmpTimeInstanceFilter_tbl.VTI_Instance) AS LastOfVTI_Instance
SELECT tmpTimeInstanceFilter_tbl.VTI_MemberID,
  Last(tmpTimeInstanceFilter_tbl.VTI_CityCode) AS VTI_CityCode,
  Last(tmpTimeInstanceFilter_tbl.VTI_SiteID) AS VTI_SiteID,
  Last(tmpTimeInstanceFilter_tbl.VTI_ProjectID) AS VTI_ProjectID
FROM tmpTimeInstanceFilter_tbl
GROUP BY tmpTimeInstanceFilter_tbl.VTI_MemberID
PIVOT tmpTimeInstanceFilter_tbl.VTI_TimeDirectionID;

```


¹⁷ Module 3 – VolunteerTimeCalculation2 qry SQL:

```
SELECT VolunteerTimeCalculation_qry.VTI_MemberID, VolunteerTimeCalculation_qry.[IN],
VolunteerTimeCalculation_qry.OUT, VolunteerTimeCalculation_qry.VTI_CityCode,
VolunteerTimeCalculation_qry.VTI_SiteID, VolunteerTimeCalculation_qry.VTI_ProjectID,
Format([IN]-1-[OUT], "Short Time") AS VTI_TotalHoursWorked
FROM VolunteerTimeCalculation_qry
GROUP BY VolunteerTimeCalculation_qry.VTI_MemberID,
VolunteerTimeCalculation_qry.[IN], VolunteerTimeCalculation_qry.OUT,
VolunteerTimeCalculation_qry.VTI_CityCode, VolunteerTimeCalculation_qry.VTI_SiteID,
VolunteerTimeCalculation_qry.VTI_ProjectID, Format([IN]-1-[OUT], "Short Time");
```

¹⁸ Module 3 – VolunteerTimeCalculationTable qry SQL:

```
INSERT INTO TotalHoursWorked ( VTI_MemberID, INTimeDirection, OUTTimeDirection,
VTI_TotalHoursWorked, VTI_CityCode, VTI_SiteID, VTI_ProjectID )
SELECT VolunteerTimeCalculation2_qry.VTI_MemberID AS VTI_MemberID,
VolunteerTimeCalculation2_qry.[IN] AS INTimeDirection, VolunteerTimeCalculation2_qry.OUT
AS OUTTimeDirection, VolunteerTimeCalculation2_qry.VTI_TotalHoursWorked AS
VTI_TotalHoursWorked, VolunteerTimeCalculation2_qry.VTI_CityCode AS VTI_CityCode,
VolunteerTimeCalculation2_qry.VTI_SiteID AS VTI_SiteID,
VolunteerTimeCalculation2_qry.VTI_ProjectID AS VTI_ProjectID
FROM VolunteerTimeCalculation2_qry;
```

¹⁹ Module 4 – QRY CreateReportReference SQL:

```
SELECT TotalHoursWorked.VTI_MemberID, TotalHoursWorked.VTI_TotalHoursWorked,
TotalHoursWorked.VTI_DateEntered, VolunteerMasterInformation.VMI_NameLast,
VolunteerMasterInformation.VMI_NameMiddle, VolunteerMasterInformation.VMI_NameFirst,
OrganizationSites.OS_SiteName, Projects.P_ProjectName, Definitions_City.DC_Name
FROM (((TotalHoursWorked INNER JOIN VolunteerMasterInformation ON
TotalHoursWorked.VTI_MemberID = VolunteerMasterInformation.VMI_MemberID) INNER
JOIN OrganizationSites ON TotalHoursWorked.VTI_SiteID = OrganizationSites.OS_SiteID)
INNER JOIN Definitions_City ON TotalHoursWorked.VTI_CityCode = Definitions_City.DC_ID)
INNER JOIN Projects ON TotalHoursWorked.VTI_ProjectID = Projects.P_ProjectID) LEFT
JOIN BusinessContact ON VolunteerMasterInformation.VMI_MemberID =
BusinessContact.BC_MemberID;
```

²⁰ Module 4 – QRY CompileReport SQL:

```
SELECT TotalHoursWorked.VTI_MemberID, TotalHoursWorked.VTI_TotalHoursWorked,
TotalHoursWorked.VTI_DateEntered, VolunteerMasterInformation.VMI_NameLast,
VolunteerMasterInformation.VMI_NameMiddle, VolunteerMasterInformation.VMI_NameFirst,
OrganizationSites.OS_SiteName, Projects.P_ProjectName, Definitions_City.DC_Name
FROM (((TotalHoursWorked INNER JOIN VolunteerMasterInformation ON
TotalHoursWorked.VTI_MemberID = VolunteerMasterInformation.VMI_MemberID) INNER
JOIN OrganizationSites ON TotalHoursWorked.VTI_SiteID = OrganizationSites.OS_SiteID)
INNER JOIN Definitions_City ON TotalHoursWorked.VTI_CityCode = Definitions_City.DC_ID)
INNER JOIN Projects ON TotalHoursWorked.VTI_ProjectID = Projects.P_ProjectID) LEFT
JOIN BusinessOccupation ON VolunteerMasterInformation.VMI_MemberID =
BusinessOccupation.BO_MemberID
WHERE (((TotalHoursWorked.VTI_DateEntered) Between
[Forms]![uFRM_CreateReport]![BeginningDate] And
[Forms]![uFRM_CreateReport]![EndingDate]));
```

²¹ Module 4 – RPT DateRange:

Volunteer Work Hours

01/01/2003 through 02/09/2006

MemberID: 3415-3758-7000-6437 Lynne Giddings

Date: **01/31/2006**

<i>Time Worked:</i>	<i>Site:</i>	<i>Project:</i>	<i>City:</i>
6:28	International Headquarters	Attendance	Denver

Date: **02/02/2006**

<i>Time Worked:</i>	<i>Site:</i>	<i>Project:</i>	<i>City:</i>
4:09	International Headquarters	Attendance	Denver

Total Time: **10:37** *Hours:Minutes*

MemberID: 4010-8896-6477-2710 Karen Harmon

Date: **01/31/2006**

<i>Time Worked:</i>	<i>Site:</i>	<i>Project:</i>	<i>City:</i>
1:33	International Headquarters	Attendance	Denver

Date: **02/02/2006**

<i>Time Worked:</i>	<i>Site:</i>	<i>Project:</i>	<i>City:</i>
2:39	International Headquarters	Attendance	Denver

Total Time: **4:12** *Hours:Minutes*

MemberID: 4095-5050-3722-6623 Desirea Duarte Ulibarri

Date: **02/03/2006**

<i>Time Worked:</i>	<i>Site:</i>	<i>Project:</i>	<i>City:</i>
2:58	International Headquarters	Volunteer System Project	Denver
1:41	International Headquarters	Volunteer System Project	Denver
23:57	International Headquarters	Volunteer System Project	Denver
9:10	International Headquarters	Attendance	Denver

Date: **02/04/2006**

<i>Time Worked:</i>	<i>Site:</i>	<i>Project:</i>	<i>City:</i>
3:58	International Headquarters	Volunteer System Project	Denver

Date: **02/08/2006**

<i>Time Worked:</i>	<i>Site:</i>	<i>Project:</i>	<i>City:</i>
0:11	International Headquarters	Volunteer System Project	Denver

Total Time: **41:55** *Hours:Minutes*

Total Report Time: 56:44

Thursday, February 09, 2006

Page 1 of 1

-
- ²² Module 4 – Calculating above 24 hours SQL:
`Int((Sum([VTI_TotalHoursWorked])*24) & ":" & Format(Sum([VTI_TotalHoursWorked]),"nn"))`
- ²³ Module 4 – MCR_DateRangeALLRunReport in VISUAL BASIC:
`Private Sub CMD_DateRangeALLRunReport_Click()
 On Error GoTo Err_CMD_DateRangeALLRunReport_Click
 DoCmd.OpenReport "RPT_DateRange", acViewPreview, "", "", acNormal
Exit_CMD_DateRangeALLRunReport_Click:
 Exit Sub
Err_CMD_DateRangeALLRunReport_Click:
 MsgBox Error$
 Resume Exit_CMD_DateRangeALLRunReport_Click
End Sub`
- ²⁴ Module 4 - MCR_DateRangeFILTERRunReport in VISUAL BASIC:
`Private Sub CMD_DateRangeFILTERRunReport_Click()
 On Error GoTo Err_CMD_DateRangeFILTERRunReport_Click
 DoCmd.OpenReport "RPT_DateRange", acViewPreview, "",
"[TotalHoursWorked].[VTI_MemberID]=[Forms]![uFRM_CreateReport]![MemberIDSecondPage]", acNormal
Exit_CMD_DateRangeFILTERRunReport_Click:
 Exit Sub
Err_CMD_DateRangeFILTERRunReport_Click:
 MsgBox Error$
 Resume Exit_CMD_DateRangeFILTERRunReport_Click
End Sub`
- ²⁵ Module 4 – MCR_ProjectFILTERRunReport in VISUAL BASIC:
`Private Sub CMD_ProjectFILTERRunReport_Click()
 On Error GoTo Err_CMD_ProjectFILTERRunReport_Click
 DoCmd.OpenReport "RPT_DateRange", acViewPreview, "",
"[Projects].[P_ProjectName]=[Forms]![uFRM_CreateReport]![ProjectNameThirdPage]",
acNormal
Exit_CMD_ProjectFILTERRunReport_Click:
 Exit Sub
Err_CMD_ProjectFILTERRunReport_Click:
 MsgBox Error$
 Resume Exit_CMD_ProjectFILTERRunReport_Click
End Sub`
- ²⁶ Module 4 - MCR_OrgSiteFILTERRunReport in VISUAL BASIC:
`Private Sub CMD_OrgSiteFILTERRunReport_Click()
 On Error GoTo Err_CMD_OrgSiteFILTERRunReport_Click
 DoCmd.OpenReport "RPT_DateRange", acViewPreview, "",
"[OrganizationSites].[OS_SiteName]=[Forms]![uFRM_CreateReport]![OrgSiteFourthPage]",
acNormal
Exit_CMD_OrgSiteFILTERRunReport_Click:
 Exit Sub
Err_CMD_OrgSiteFILTERRunReport_Click:
 MsgBox Error$
 Resume Exit_CMD_OrgSiteFILTERRunReport_Click
End Sub`
- ²⁷ Module 4 - MCR_CityFILTERRunReport in VISUAL BASIC:
`Private Sub CMD_CityFILTERRunReport_Click()`

```

On Error GoTo Err_CMD_CityFILTERRunReport_Click
    DoCmd.OpenReport "RPT_DateRange", acViewPreview, "",
    "[Definitions_City].[DC_Name]=[Forms]![uFRM_CreateReport]![CityNameFifthPage]",
    acNormal
Exit_CMD_CityFILTERRunReport_Click:
    Exit Sub
Err_CMD_CityFILTERRunReport_Click:
    MsgBox Error$
    Resume Exit_CMD_CityFILTERRunReport_Click
End Sub

```

²⁸ CR#1-4 QRY CreateReportReference SQL:

```

SELECT TotalHoursWorked.VTI_MemberID, TotalHoursWorked.VTI_TotalHoursWorked,
TotalHoursWorked.VTI_DateEntered, VolunteerMasterInformation.VMI_NameLast,
VolunteerMasterInformation.VMI_NameMiddle, VolunteerMasterInformation.VMI_NameFirst,
OrganizationSites.OS_SiteName, Projects.P_ProjectName, Definitions_City.DC_Name,
BusinessContact.BC_OrgName
FROM BusinessContact INNER JOIN (Projects INNER JOIN (Definitions_City INNER JOIN
(OrganizationSites INNER JOIN (TotalHoursWorked INNER JOIN VolunteerMasterInformation
ON TotalHoursWorked.VTI_MemberID = VolunteerMasterInformation.VMI_MemberID) ON
OrganizationSites.OS_SiteID = TotalHoursWorked.VTI_SiteID) ON Definitions_City.DC_ID =
TotalHoursWorked.VTI_CityCode) ON Projects.P_ProjectID =
TotalHoursWorked.VTI_ProjectID) ON BusinessContact.BC_MemberID =
VolunteerMasterInformation.VMI_MemberID;

```

²⁹ CR#2-4 QRY CompileReport SQL:

```

SELECT TotalHoursWorked.VTI_MemberID, TotalHoursWorked.VTI_TotalHoursWorked,
TotalHoursWorked.VTI_DateEntered, VolunteerMasterInformation.VMI_NameLast,
VolunteerMasterInformation.VMI_NameMiddle, VolunteerMasterInformation.VMI_NameFirst,
OrganizationSites.OS_SiteName, Projects.P_ProjectName, Definitions_City.DC_Name,
BusinessContact.BC_OrgName
FROM (((TotalHoursWorked INNER JOIN VolunteerMasterInformation ON
TotalHoursWorked.VTI_MemberID = VolunteerMasterInformation.VMI_MemberID) INNER
JOIN OrganizationSites ON TotalHoursWorked.VTI_SiteID = OrganizationSites.OS_SiteID)
INNER JOIN Definitions_City ON TotalHoursWorked.VTI_CityCode = Definitions_City.DC_ID)
INNER JOIN Projects ON TotalHoursWorked.VTI_ProjectID = Projects.P_ProjectID) LEFT
JOIN BusinessContact ON VolunteerMasterInformation.VMI_MemberID =
BusinessContact.BC_MemberID
WHERE (((TotalHoursWorked.VTI_DateEntered) Between
[Forms]![uFRM_CreateReport]![BeginningDate] And
[Forms]![uFRM_CreateReport]![EndingDate]));

```

³⁰ CR# 3-4 uFRM CreateReport VISUAL BASIC:

```

Private Sub Button_CloseForm_Click()
On Error GoTo Err_Button_CloseForm_Click
    DoCmd.Close
Exit_Button_CloseForm_Click:
    Exit Sub
Err_Button_CloseForm_Click:
    MsgBox Err.Description
    Resume Exit_Button_CloseForm_Click
End Sub
Private Sub CMD_CityFILTERRunReport_Click()
On Error GoTo Err_CMD_CityFILTERRunReport_Click

```

```
    DoCmd.OpenReport "RPT_DateRange", acViewPreview, "",
"[/Definitions_City].[DC_Name]=[Forms]![uFRM_CreateReport]![CityNameFifthPage]",
acNormal
Exit_CMD_CityFILTERRunReport_Click:
    Exit Sub
Err_CMD_CityFILTERRunReport_Click:
    MsgBox Error$
    Resume Exit_CMD_CityFILTERRunReport_Click
End Sub
Private Sub CMD_DateRangeALLRunReport_Click()
On Error GoTo Err_CMD_DateRangeALLRunReport_Click
    DoCmd.OpenReport "RPT_DateRange", acViewPreview, "", "", acNormal
Exit_CMD_DateRangeALLRunReport_Click:
    Exit Sub
Err_CMD_DateRangeALLRunReport_Click:
    MsgBox Error$
    Resume Exit_CMD_DateRangeALLRunReport_Click
End Sub
Private Sub CMD_DateRangeFILTERRunReport_Click()
On Error GoTo Err_CMD_DateRangeFILTERRunReport_Click
    DoCmd.OpenReport "RPT_DateRange", acViewPreview, "",
"[/TotalHoursWorked].[VTI_MemberID]=[Forms]![uFRM_CreateReport]![MemberIDSecondPage
]", acNormal
Exit_CMD_DateRangeFILTERRunReport_Click:
    Exit Sub
Err_CMD_DateRangeFILTERRunReport_Click:
    MsgBox Error$
    Resume Exit_CMD_DateRangeFILTERRunReport_Click
End Sub
Private Sub CMD_OrgSiteFILTERRunReport_Click()
On Error GoTo Err_CMD_OrgSiteFILTERRunReport_Click
    DoCmd.OpenReport "RPT_DateRange", acViewPreview, "",
"[/OrganizationSites].[OS_SiteName]=[Forms]![uFRM_CreateReport]![OrgSiteFourthPage]",
acNormal
Exit_CMD_OrgSiteFILTERRunReport_Click:
    Exit Sub
Err_CMD_OrgSiteFILTERRunReport_Click:
    MsgBox Error$
    Resume Exit_CMD_OrgSiteFILTERRunReport_Click
End Sub
Private Sub CMD_ProjectFILTERRunReport_Click()
On Error GoTo Err_CMD_ProjectFILTERRunReport_Click
    DoCmd.OpenReport "RPT_DateRange", acViewPreview, "",
"[/Projects].[P_ProjectName]=[Forms]![uFRM_CreateReport]![ProjectNameThirdPage]",
acNormal
Exit_CMD_ProjectFILTERRunReport_Click:
    Exit Sub
Err_CMD_ProjectFILTERRunReport_Click:
    MsgBox Error$
    Resume Exit_CMD_ProjectFILTERRunReport_Click
End Sub
Private Sub CMD_OrgGroupNameFILTERRunReport_Click()
On Error GoTo Err_CMD_OrgGroupNameFILTERRunReport_Click
```

```

    DoCmd.OpenReport "RPT_DateRange", acViewPreview, "",
    "[BusinessContact].[BC_OrgName]=[Forms]![uFRM_CreateReport]![OrgGroupNameSixthPage
    ]", acNormal
Exit_CMD_OrgGroupNameFILTERRunReport_Click:
    Exit Sub
Err_CMD_OrgGroupNameFILTERRunReport_Click:
    MsgBox Error$
    Resume Exit_CMD_OrgGroupNameFILTERRunReport_Click
End Sub

```

³¹ CR#1-2 mFRM_AutoDisplay VISUAL BASIC:

```

Elseif Me.TimeDirectionID = "IN" Then
    LookupVariable = DLast("VTI_TimeDirectionID", "VolunteerTimeInstance",
    "[VTI_MemberID] = " & CardID & """)
    If LookupVariable = "IN" Then
        MsgBox "You have already logged IN."
        Me.TimeDirectionID.SetFocus
        'stDocName = "sFRM_VolunteerTimeInstance"
        'stLinkCriteria = "[VTI_MemberID]=" & "" & CardID & ""
        'DoCmd.OpenForm stDocName, , , stLinkCriteria
    Else
        strSQL = "INSERT INTO VolunteerTimeInstance values (" & CardID & ", " &
        Me.CityCode & ", " & Me.SiteID & ", " & Me.ProjectID & ", " & curdate & ", " &
        Me.TimeDirectionID & ");"
        db.Execute (strSQL)
        db.Close
        MsgBox "Thank you for logging IN!"
        Me.VTI_MemberID = ""
        Me.VTI_MemberID.SetFocus
        ' Check if LookUp records is checked
        If Me.Check_VI_Lookup = -1 Then
            stDocName = "mFRM_VolunteerLookUp"
            stLinkCriteria = "[VMI_MemberID]=" & "" & CardID & ""
            DoCmd.OpenForm stDocName, , , stLinkCriteria
            End If
        'stDocName = "sFRM_VolunteerTimeInstance"
        'stLinkCriteria = "[VTI_MemberID]=" & "" & CardID & ""
        'DoCmd.OpenForm stDocName, , , stLinkCriteria
        stDocName = "mFRM_AutoDisplay"
        'stDocName.SetFocus
        DoCmd.Close acForm, stDocName
    End If

```

³² CR# 4-2 mFRM_AutoDisplay VISUAL BASIC:

```

Elseif Me.TimeDirectionID = "OUT" Then
    LookupVariable = DLast("VTI_TimeDirectionID", "VolunteerTimeInstance",
    "[VTI_MemberID] = " & CardID & """)
    If LookupVariable = "OUT" Then
        MsgBox "You have already logged OUT."
        Me.TimeDirectionID.SetFocus
        'stDocName = "sFRM_VolunteerTimeInstance"
        'stLinkCriteria = "[VTI_MemberID]=" & "" & CardID & ""
        'DoCmd.OpenForm stDocName, , , stLinkCriteria
    Else

```

```
        strSQL = "INSERT INTO VolunteerTimeInstance values ('" & CardID & "', '" &
Me.CityCode & "', '" & Me.SiteID & "', '" & Me.ProjectID & "', '" & currdate & "', '" &
Me.TimeDirectionID & "');"
        db.Execute (strSQL)
        db.Close
        Me.VTI_MemberID = ""
        Me.VTI_MemberID.SetFocus
        stDocName = "VolunteerTimeCalculationTable_qry"
        stTimeCalcForm = "mFRM_TotalHoursWorked"
        Me.VTI_MemberID.SetFocus
        stLinkCriteria = "[VTI_MemberID]=" & "" & CardID & ""
        DoCmd.SetWarnings False
        DoCmd.OpenForm "sFRM_VolunteerTimeInstance", , , stLinkCriteria
        DoCmd.OpenQuery "VolunteerTimeInstanceTable_qry", acNormal, acEdit
        DoCmd.OpenQuery stDocName
        DoCmd.OpenForm stTimeCalcForm
        stDocName = "sFRM_VolunteerTimeInstance"
        DoCmd.Close acForm, stDocName
        stDocName = "mFRM_AutoDisplay"
        DoCmd.Close acForm, stDocName
    End If
End If
```

APPENDIX 2 – Repository Changes

To support development of each module, there will be new tables and table field additions to the existing front-end and back-end database engines. The back-end engine **dbVM** contains all permanent tables while **fmVM** contains only those tables housing temporary information such as imported data and work time calculations. All new and changed tables are identified below:

New/Changed	DB	Enhanced/New Table Names	Table Field Additions	Enhanced/New Queries and Forms	In Design Document?
new	dbVM	TotalHoursWorked table	VTI_MemberID	QRY_CreateReportReference query	yes - SQL
			INTimeDirection		yes - SQL
			OUTTimeDirection	QRY_CompileReport query	yes - SQL
			VTI_TotalHoursWorked		
			VTI_CityCode	uFRM_CreateReport form	yes - VB code
			VTI_SiteID		
			VTI_ProjectID	sFRM_TotalHoursWorked form	
VTI_DateEntered					
new	dbVM	ValidType_TimeInstanceDirection table	VTI_Direction	mFRM_AutoDisplay form	yes - VB code
new	fmVM	ImportTemp table (temporary)	See ImportTemp File Specs	QRY_ImportTemp query	yes - SQL
				TEST_GenUniqueKey_frm form	yes - VB code
				QRY_ImportTempFilter query	yes - file specs
new	fmVM	tmpTimeInstanceFilter_tbl table (temporary)	VTI_MemberID	VolunteerTimeCalculation_qry query	yes - SQL
			VTI_CityCode		
			VTI_SiteID		
			VTI_ProjectID		
			VTI_Instance		
VTI_TimeDirectionID					
change	dbVM	VolunteerTimeInstance table	VTI_TimeDirectionID	mFRM_AutoDisplay form	yes - VB code
				sFRM_VolunteerTimeInstance form	yes - VB code
				VolunteerTimeInstanceTable_qry query	yes - SQL
new	fmVM	Output from VolunteerTimeCalculation_qry query (temporary)	IN	VolunteerTimeCalculation2_qry query	yes - SQL
			OUT		
new	fmVM	Output from VolunteerTimeCalculation2_qry query (temporary)	VTI_TotalHoursWorked	VolunteerTimeCalculationTable_qry query	yes - SQL
				mFRM_TotalHoursWorked form	
new	fmVM	Output from VolunteerTimeCalculationTable_qry query (temporary)	INTimeDirection		yes - SQL
			OUTTimeDirection		

new	fmVM	Output from QRY_CompileReport query	Sum([VTI_TotalHours Worked])	RPT_DateRange report	yes - SQL and report layout
change	dbVM	VolunteerAvailableTimes table	VAT_DateEntered	sFRM_VolunteerAvailableTimes form	yes - VB code
change	dbVM	BusinessOccupation table	RecNo	QRY_ImportTempFilter query	yes - SQL
			BO_DateEntered	TEST_GenUniqueKey_fm form	yes - SQL
change	dbVM	BusinessContact table	RecNo	QRY_ImportTempFilter query	yes - SQL
			BC_DateEntered	TEST_GenUniqueKey_fm form	yes - SQL
change	dbVM	PermanentContact table	RecNo	QRY_ImportTempFilter query	yes - SQL
			PC_DateEntered	TEST_GenUniqueKey_fm form	yes - SQL
change	dbVM	VolunteerMasterInformation table	RecNo	mFRM_VolunteerLookUp form	yes - VB code
			VMI_DateEntered	TEST_GenUniqueKey_fm form	yes - SQL
				REPORT_TotalHoursWorked_qry query	yes - VB code
change	dbVM	DatabaseOperations table	DBO_DateEntered		no - existing
change	dbVM	VolunteerFavorites table	VF_DateEntered		no - existing
change	dbVM	VolunteerCountryVisited table	VC_DateEntered		no - existing
change	dbVM	VolunteerAffiliations table	VA_DateEntered		no - existing
change	dbVM	VolunteerRelationships table	VR_DateEntered		no - existing
change	dbVM	Volunteer Skills table	VS_DateEntered		no - existing

For a full data dictionary with relationships of the Volunteer System: **See Appendix 3 – Volunteer System Data Dictionary with Entity Relationship Descriptions.**

Appendix 3 – Volunteer System Data Dictionary with Entity Relationship Descriptions

dbVM

Name	Type	Size
BC_MemberID	Text	20
BC_OrgName	Text	50
BC_AddrLine1	Text	50
BC_AddrLine2	Text	50
BC_AddrCity	Text	28
BC_AddrState	Text	28
BC_AddrZipCode	Text	10
BC_AddrCountry	Text	28
BC_ConPhone	Text	14
BC_ConFax	Text	14
BC_ConCell	Text	14
BC_ConEmail	Text	50
BC_DateEntered	Date/Time	8
RecNo	Double	8

Relationships

VolunteerMaster Information	BusinessContact
VMI_MemberID	BC_MemberID

Attributes: Unique, Not Enforced, Inherited, Left Join
 RelationshipType: One-To-One (External)

Table: BusinessContact

Table: BusinessOccupation

Name	Type	Size
BO_MemberID	Text	20
BO_JobTitle	Text	50
BO_Notes	Text	255
BO_DateEntered	Date/Time	8
RecNo	Double	8

Relationships

VolunteerMaster Information	BusinessOccupation
VMI_MemberID	BO_MemberID

Attributes: Unique, Not Enforced, Inherited, Left Join
 RelationshipType: One-To-One (External)

Table: DatabaseOperations

Name	Type	Size
DBO_MemberID	Text	20
DBO_Password	Text	8
DBO_DateEntered	Date/Time	8

Table: Definitions_City

Name	Type	Size
DC_ID	Text	4
DC_Name	Text	50

Relationships

Definitions_City	OrganizationSites
DC_ID	OS_SiteID

Attributes: Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

Table: Nomenclature_VolCode

Name	Type	Size
NVC_Code	Text	255
NVC_Level	Double	8
NVC_Description	Text	255

Table: OrganizationSites

Name	Type	Size
OS_CityCode	Text	4
OS_SiteID	Text	4
OS_SiteName	Text	50
OS_AddrLine1	Text	50
OS_AddrLine2	Text	50
OS_AddrCity	Text	28
OS_AddrState	Text	28
OS_AddrZipCode	Text	10
OS_AddrCountry	Text	28
OS_ConPhone	Text	14
OS_ConFax	Text	14
OS_ConCell	Text	14
OS_ConEmail	Text	50
OS_Min	Text	10
OS_Max	Text	10

Relationships

Definitions_City	OrganizationSites
DC_ID	OS_SiteID

Attributes: Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

Table: PermanentContact

Name	Type	Size
PC_MemberID	Text	20
PC_AddrLine1	Text	50
PC_AddrLine2	Text	50
PC_AddrCity	Text	28
PC_AddrState	Text	28
PC_AddrZipCode	Text	10
PC_AddrCountry	Text	28
PC_ConPhone	Text	14
PC_ConFax	Text	14
PC_ConCell	Text	14
PC_ConEmail	Text	50
PC_DateEntered	Date/Time	8
RecNo	Double	8

Relationships

VolunteerMaster Information	PermanentContact
VMI_MemberID	PC_MemberID

Attributes: Unique, Not Enforced, Inherited, Left Join
 RelationshipType: One-To-One (External)

Table: Projects

Name	Type	Size
P_ProjectID	Integer	2
P_ProjectName	Text	50
P_Type	Yes/No	1

Table: TotalHoursWorked

Name	Type	Size
VTI_MemberID	Text	20
INTimeDirection	Date/Time	8
OUTTimeDirection	Date/Time	8
VTI_TotalHoursWorked	Date/Time	8
VTI_CityCode	Text	4
VTI_SiteID	Text	4
VTI_ProjectID	Integer	2
VTI_DateEntered	Date/Time	8

Table: ValidType_Affiliations

Name	Type	Size
VTA_AffiliationType	Text	32
VTA_AffiliationValue	Text	50

Relationships

ValidType_Affiliations	VolunteerAffiliations
VTA_AffiliationValue	VA_AffiliationValue

Attributes: Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

Table: ValidType_AffiliationType

Name	Type	Size
VTAT_AffiliationValues	Text	50

Relationships

ValidType_Affiliation	VolunteerAffiliations
VTAT_AffiliationValues	VA_AffiliationType

Attributes: Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

Table: ValidType_Country

Name	Type	Size
VTC_Country	Text	20

Relationships

ValidType_Country	VolunteerCountryVisited
VTC_Country	VC_Country

Attributes: Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

Table: ValidType_Favorites

Name	Type	Size
VTF_FavoriteType	Text	14
VTF_FavoriteValue	Text	50

Relationships

ValidType_Favorites	VolunteerFavorites
VTF_FavoriteValue	VF_FavoriteValue

Attributes: Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

Table: ValidType_FavoriteTypes

Name	Type	Size
VTFT_TypeValues	Text	14

Relationships

ValidType_FavoriteType	VolunteerFavorites
VTFT_TypeValues	VF_FavoriteType

Attributes: Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

Table: ValidType_LevelOfInterest

Name	Type	Size
VTLOI_Value	Byte	1
VTLOI_Description	Text	28

Relationships

ValidType_LevelOfInterest	VolunteerWorkPreferences
---------------------------	--------------------------

Table: ValidType_AvailableTimes

Name	Type	Size
VTA_TimeCode	Text	4
VTA_AvailableTimes	Text	40

Relationships

ValidType_AvailableTimes	VolunteerAvailableTimes
VTA_TimeCode	VAT_Sun

Attributes: Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

ValidType_AvailableTimes	VolunteerAvailableTimes
VTA_TimeCode	VAT_Mon

Attributes: Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

ValidType_AvailableTimes	VolunteerAvailableTimes
VTA_TimeCode	VAT_Tue

Attributes: Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

ValidType_AvailableTimes	VolunteerAvailableTimes
VTA_TimeCode	VAT_Wed

Attributes: Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

ValidType_AvailableTimes	VolunteerAvailableTimes
VTA_TimeCode	VAT_Thurs

Attributes: Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

ValidType_AvailableTimes	VolunteerAvailableTimes
VTA_TimeCode	VAT_Fri

Attributes: Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

ValidType_AvailableTimes	VolunteerAvailableTimes
VTA_TimeCode	VAT_Sat

Attributes: Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

Table: ValidType_SkillCategories

Name	Type	Size
VTSC_Code	Text	4
VTSC_SkillCategories	Text	70

VTLOI_Value	VWP_LevelOfInterest
Attributes:	Enforced, Inherited, Left Join
RelationshipType:	One-To-Many (External)

Table: ValidType_LevelOfSkill

Name	Type	Size
VTLOS_Value	Byte	1
VTLOS_Description	Text	28

Relationships

ValidType_LevelOfSkill	VolunteerSkills
VTLOS_Value	VS_LevelOfSkill

Attributes: Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

Table: ValidType_Relationship

Name	Type	Size
VTR_Forward	Text	14
VTR_Backward	Text	14

Relationships

ValidType_Relationship	VolunteerRelationships
VTR_Forward	VR_RelationshipType

Attributes: Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

Table: ValidType_MaritalStatus

Name	Type	Size
SMWStatus	Text	8

Relationships

ValidType_MaritalStatus	VolunteerMasterInformation
SMWStatus	VMI_MaritalStatus

Attributes: Not Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

Table: ValidType_Prefix

Name	Type	Size
VTP_Prefix	Text	14

Relationships

ValidType_Prefix	VolunteerMasterInformation
VTP_Prefix	VMI_NamePrefix

Attributes: Not Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

Table: ValidType_SpecializedTraining

Name	Type	Size
VTST_STCode	Text	4
VTST_SpecializedTraining	Text	50

Relationships

ValidType_SpecializedTraining	VolunteerSkills
VTST_STCode	VS_SkillCode

Attributes: Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

Table: ValidType_Suffix

Name	Type	Size
VTS_Suffix	Text	14

Relationships

ValidType_Suffix	VolunteerMasterInformation
VTS_Suffix	VMI_NameSuffix

Attributes: Not Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

Table: ValidType_TimeInstanceDirection

Name	Type	Size
VTI_Direction	Text	50

Table: ValidType_WorkCategories

Name	Type	Size
VTWC_Code	Text	4
VTWC_WorkCategories	Text	70

Relationships

ValidType_WorkCategories	VolunteerWork Preferences
VTWC_Code	VWP_WorkCode

Attributes: Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

Table: VolunteerCountryVisited

Name	Type	Size
VC_MemberID	Text	20
VC_Country	Text	14
VC_VisitYear	Text	4
VC_DateEntered	Date/Time	8

Relationships

ValidType_Country	VolunteerCountryVisited
VTC_Country	VC_Country

Attributes: Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

VolunteerMasterInformation	VolunteerCountryVisit
VMI_MemberID	VC_MemberID

Attributes: Not Enforced, Inherited, Left Join
 RelationshipType: One-To-One (External)

Table: VolunteerFavorites

Name	Type	Size
VF_MemberID	Text	20
VF_FavoriteType	Text	14
VF_FavoriteValue	Text	50
VF_DateEntered	Date/Time	8

Relationships

ValidType_Favorites	VolunteerFavorites
VTF_FavoriteValue	VF_FavoriteValue

Attributes: Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

ValidType_FavoriteTypes	VolunteerFavorites
VTFT_TypeValues	VF_FavoriteType

Attributes: Enforced, Inherited, Left Join
 RelationshipType: One-To-Many (External)

VolunteerMasterInformation	VolunteerFavorites
VMI_MemberID	VF_MemberID

Attributes: Not Enforced, Inherited, Left Join
 RelationshipType: One-To-One (External)

Table: VolunteerAffiliations

Name	Type	Size
VA_MemberID	Text	20
VA_AffiliationType	Text	32
VA_AffiliationValue	Text	50
VA_DateEntered	Date/Time	8

Relationships

ValidType_Affiliations	VolunteerAffiliations
VTA_AffiliationValue	VA_AffiliationValue

Attributes: Enforced, Inherited, Left Join

RelationshipType: One-To-Many (External)

ValidType_Affiliation	VolunteerAffiliations
VTAT_AffiliationValues	VA_AffiliationType

Attributes: Enforced, Inherited, Left Join

RelationshipType: One-To-Many (External)

VolunteerMaster Information	VolunteerAffiliations
VMI_MemberID	VA_MemberID

Attributes: Not Enforced, Inherited, Left Join

RelationshipType: One-To-One (External)

Table: VolunteerAvailableTimes

Name	Type	Size
VAT_MemberID	Text	20
VAT_Sun	Text	4
VAT_Mon	Text	4
VAT_Tue	Text	4
VAT_Wed	Text	4
VAT_Thu	Text	4
VAT_Fri	Text	4
VAT_Sat	Text	4
VAT_DateEntered	Date/Time	8

VolunteerMaster Information	VolunteerAvailableTimes
VMI_MemberID	VAT_MemberID

Attributes: Unique, Not Enforced, Inherited, Left Join

RelationshipType: One-To-One (External)

Table: VolunteerSpecializedTraining

Name	Type	Size
VST_MemberID	Text	20
VST_LevelOfSkill	Byte	1
VST_TrainingCode	Text	4

Table: VolunteerTimeInstance

Name	Type	Size
VTI_MemberID	Text	20
VTI_CityCode	Text	4
VTI_SiteID	Text	4
VTI_ProjectID	Integer	2
VTI_Instance	Date/Time	8
VTI_TimeDirectionID	Text	50

Table: VolunteerMasterInformation

Name	Type	Size
VMI_MemberID	Text	20
VMI_VolCode	Text	6
VMI_DateJoined	Date/Time	8
VMI_NameLast	Text	28
VMI_NameMiddle	Text	28
VMI_NameFirst	Text	28
VMI_NamePrefix	Text	14
VMI_NameSuffix	Text	14
VMI_BirthDate	Date/Time	8
VMI_MaritalStatus	Text	14
VMI_PersonDeceased	Yes/No	1
VMI_PreferenceAddress	Yes/No	1
VMI_PersonGender	Yes/No	1
VMI_PreferenceNewsletter	Yes/No	1
VMI_StatusActive	Yes/No	1
VMI_DateEntered	Date/Time	8
RecNo	Double	8

Table: VolunteerSkills

Name	Type	Size
VS_MemberID	Text	20
VS_LevelOfSkill	Byte	1
VS_SkillCode	Text	4
VS_DateEntered	Date/Time	8

Relationships

ValidType_LevelOfSkills	VolunteerSkills
VTLOS_Value	VS_LevelOfSkill

Attributes: Enforced, Inherited, Left Join

RelationshipType: One-To-Many (External)

ValidType_SpecializedTraining	VolunteerSkills
VTST_STCode	VS_SkillCode

Attributes: Enforced, Inherited, Left Join

RelationshipType: One-To-Many (External)

VolunteerMasterInformation	VolunteerSkills
VMI_MemberID	VS_MemberID

Attributes: Not Enforced, Inherited, Left Join

RelationshipType: One-To-One (External)

Relationships

ValidType_MaritalStatus	VolunteerMasterInformation
SMWStatus	VMI_MaritalStatus

Attributes: Not Enforced, Inherited, Left Join

RelationshipType: One-To-Many (External)

ValidType_Prefix	VolunteerMasterInformation
VTP_Prefix	VMI_NamePrefix

Attributes: Not Enforced, Inherited, Left Join

RelationshipType: One-To-Many (External)

ValidType_Suffix	VolunteerMasterInformation
VTS_Suffix	VMI_NameSuffix

Attributes: Not Enforced, Inherited, Left Join

RelationshipType: One-To-Many (External)

Table: VolunteerMasterInformation CONT.

VolunteerMaster Information	BusinessContact
VMI_MemberID	BC_MemberID
Attributes:	Unique, Not Enforced, Inherited, Left Join
RelationshipType:	One-To-One (External)
VolunteerMaster Information	BusinessOccupation
VMI_MemberID	BO_MemberID
Attributes:	Unique, Not Enforced, Inherited, Left Join
RelationshipType:	One-To-One (External)
VolunteerMaster Information	PermanentContact
VMI_MemberID	PC_MemberID
Attributes:	Unique, Not Enforced, Inherited, Left Join
RelationshipType:	One-To-One (External)
VolunteerMaster Information	VolunteerAffiliations
VMI_MemberID	VA_MemberID
Attributes:	Not Enforced, Inherited, Left Join
RelationshipType:	One-To-One (External)
VolunteerMaster Information	VolunteerAvailableTimes
VMI_MemberID	VAT_MemberID
Attributes:	Unique, Not Enforced, Inherited, Left Join
RelationshipType:	One-To-One (External)
VolunteerMaster Information	VolunteerCountryVisited
VMI_MemberID	VC_MemberID
Attributes:	Not Enforced, Inherited, Left Join
RelationshipType:	One-To-One (External)
VolunteerMaster Information	VolunteerFavorites
VMI_MemberID	VF_MemberID
Attributes:	Not Enforced, Inherited, Left Join
RelationshipType:	One-To-One (External)
VolunteerMaster Information	VolunteerSkills
VMI_MemberID	VS_MemberID
Attributes:	Not Enforced, Inherited, Left Join
RelationshipType:	One-To-One (External)
VolunteerMaster Information	VolunteerWorkPreferences
VMI_MemberID	VWP_MemberID
Attributes:	Not Enforced, Inherited, Left Join
RelationshipType:	One-To-One (External)

Table: VolunteerWorkPreferences

Name	Type	Size
VWP_MemberID	Text	20
VWP_LevelOfInterest	Byte	1
VWP_WorkCode	Text	4

Relationships

ValidType_LevelOfInterest	VolunteerWorkPreferences
VTLOI_Value	VWP_LevelOfInterest
Attributes:	Enforced, Inherited, Left Join
RelationshipType:	One-To-Many (External)
ValidType_WorkCategories	VolunteerWorkPreferences
VTWC_Code	VWP_WorkCode
Attributes:	Enforced, Inherited, Left Join
RelationshipType:	One-To-Many (External)
VolunteerMasterInformation	VolunteerWorkPreferences
VMI_MemberID	VWP_MemberID
Attributes:	Not Enforced, Inherited, Left Join
RelationshipType:	One-To-One (External)

Table: ZipCodeCityState

Name	Type	Size
ZCS_ZipCode	Text	10
ZCS_City	Text	28
ZCS_State	Text	28

Table: VolunteerRelationships

Name	Type	Size
VR_ID1	Text	20
VR_ID2	Text	20
VR_RelationshipType	Text	14
VR_DateEntered	Date/Time	8

Relationships

ValidType_Relationship	VolunteerRelationships
VTR_Forward	VR_RelationshipType
Attributes:	Enforced, Inherited, Left Join
RelationshipType:	One-To-Many (External)

fmVM		Table: tmpTimeInstanceFilter_tbl	
Name	Type	Size	
VTI_MemberID	Text	20	
VTI_CityCode	Text	4	
VTI_SiteID	Text	4	
VTI_ProjectID	Integer	2	
VTI_Instance	Date/Time	8	
VTI_TimeDirectionID	Text	50	

Table: ImportTemp (*See Appendix 1*)

dbMGMT			Table: tmpMagCard			Table: VolTotalInfo		
Name	Type	Size	Name	Type	Size	Name	Type	Size
VMI_MemberID	Text	255	VMI_MemberID	Text	20	VMI_MemberID	Text	20
VMI_NameFirst	Text	255	VMI_NameFirst	Text	28	VMI_VolCode	Text	6
VMI_NameMiddle	Text	255	VMI_NameMiddle	Text	28	VMI_DateJoined	Date/Time	8
VMI_NameLast	Text	255	VMI_NameLast	Text	28	VMI_NameLast	Text	28
VMI_DateJoined	Date/Time	8	VMI_StatusActive	Yes/No	1	VMI_NamePrefix	Text	14
VMI_VolCode	Text	6	PC_AddrLine1	Text	255	VMI_NameSuffix	Text	14
PC_AddrLine1	Text	255	PC_AddrCity	Text	255	VMI_BirthDate	Date/Time	8
PC_AddrCity	Text	255	PC_AddrState	Text	255	VMI_MaritalStatus	Text	14
PC_AddrState	Text	255	PC_AddrZipCode	Text	255	VMI_PersonDeceased	Yes/No	1
PC_AddrZipCode	Text	255	JoinedMonth	Text	255	VMI_PersonGender	Yes/No	1
JoinedMonth	Text	255	JoinedYear	Text	255	VMI_PreferenceAddress	Yes/No	1
JoinedYear	Text	255	CardID	Text	255	VMI_PreferenceNewsletter	Yes/No	1
CardID	Text	255	CardName	Text	255	VMI_StatusActive	Yes/No	1
CardName	Text	255	Track01	Text	255	PC_AddrLine1	Text	50
Track01	Text	255	Track02	Text	255	PC_AddrLine2	Text	50
Track02	Text	255				PC_AddrCity	Text	28
						PC_AddrState	Text	28
						PC_AddrZipCode	Text	10
						PC_AddrCountry	Text	28
						PC_ConPhone	Text	14
						PC_ConFax	Text	14
						PC_ConCell	Text	14
						PC_ConEmail	Text	50
						BC_OrgName	Text	50
						BC_AddrLine1	Text	50
						BC_AddrLine2	Text	50
						BC_AddrCity	Text	28
						BC_AddrState	Text	28
						BC_AddrZipCode	Text	10
						BC_AddrCountry	Text	28
						BC_ConPhone	Text	14
						BC_ConFax	Text	14
						BC_ConCell	Text	14
						BC_ConEmail	Text	50
						BO_JobTitle	Text	50
						BO_Notes	Text	255

Appendix E: Volunteer System Test Case Scenarios

Module 1 and 2:

Test (Use) Case Scenario	Test Description	Test Step	Step Description	Step Expected Results	Test Results	DATA SETUP REQUIRE-MENTS
0010 - Upload/Data Entry Data will enter the Volunteer Mgmt DB by Upload/Import	Data can be uploaded into the Volunteer Mgmt DB from an Excel, comma-delimited, .csv, XML, or text file or another database table.	1	The user opens the database frmVM and the Switchboard form is open and available with option buttons.	Switchboard opens		
		2	The option 'Import File from GiftMakerPRO' is an available option on the Switchboard. Once a user selects this option, the upload/import macro is triggered.	Upload/Import process opens a browse window in order to select a file.		Use TEST_Volunteer.xls

		3	<p>The upload macro is combination of macro and SQL statements:</p> <ol style="list-style-type: none"> 1) User is prompted to browse and select a file for importing - select 'TEST_Volunteer.xls' 2) Designate column headings 'First Row Contains Headings' , click NEXT 3) Select to create a new table, click NEXT 4) Click NEXT through two more screens (accept default population) 5) Name the table 'ImportTemp', CLICK FINISH 6) Select 'YES' to overwrite any existing table (if applicable) 	'Add New Volunteer Master Information' form opens when upload is finished.		
		4	There should be 5 records that can be selected. The form should default to '1 of 5'	At bottom of form, the record selector should indicate '1 of 5'		
		5	Validate the first entry in the form.	The entry name should be 'Sharing Stephenson-Golden':		
		6	Press F11	Nothing - F11 should NOT open the database window		
		7	Close 'Add New Volunteer Master Information' form in upper right-hand corner. Switchboard should still be open.	Switchboard remains open		

		8	Press F11	The database window will open and will default to 'Tables'		
		9	Browse and select 'ImportTemp'	ImportTemp' table should open and present 5 entries.		
		10	Close ImportTemp table in upper right-hand corner. Switchboard should still be open.	Database window and Switchboard should be visible.		
		11	Click on Switchboard. Select 'Quit Volunteer Management'.	The entire Volunteer Management database should close. Microsoft Access is exited.		
0021 - Create MemberID for upload/import data entries.	The user can select "Generate ID" to create a unique 16-digit volunteer identification number. The fields First Name, Last Name, City, State, and Phone must be populated for the algorithm to produce an ID number, otherwise the user will get an error message.	1	The user opens the database frmVM and the Switchboard form is open and available with option buttons.	Switchboard opens		
		2	The option 'Re-Enter Import Form' is an available option on the Switchboard. Once a user selects this option, the 'Add New Volunteer Master Information' form opens.	'Add New Volunteer Master Information' form opens. At bottom of form, the record selector should indicate '1 of 5'.		
		3	Validate the first entry in the form.	The entry name should be 'Stephenson-Golden':		
		4	Select 'Generate ID'	An error message will pop-up that says 'Need to Enter Phone to Generate Unique Key' press OK.		

		5	Enter in phone number 999-999-999, use only 9 digits.	An error message will pop-up that says 'The value you entered isn't appropriate for the input mask'!(999)" "000\0000;0;_' specified for this field' press OK.		
		6	Re-Enter in phone number 999-999-9999 using all 10 digits	The value is accepted with no error.		
		7	Select 'Generate ID'	Member ID will be populated with '6925-1418-5143-5801'		
0012 - Upload/Import data is permanently saved in the Volunteer Management system.	Immediately after a unique ID number is generated, VB code will parse and push the uploaded and additional data to populate the VolunteerMasterInformation, PermanentContact, BusinessContact, and BusinessOccupation tables. The inserted data contains the unique primary record number key from GiftMaker PRO as well as the newly assigned Volunteer ID number which now becomes the primary and relational key for all four tables in the Volunteer Mgmt DB.	1	Press F11	Nothing - F11 should NOT open the database window		
		2	Close 'Add New Volunteer Master Information' form in upper right-hand corner. Switchboard should still be open.	Switchboard remains open		
		3	Press F11	The database window will open and will default to 'Tables'		
		4	Browse and select 'Business Contact'	Business Contact' table should open.		
	A person can search by Date the record was entered into the table	5	Highlight column 'BC_DateEntered', Select 'ZtoA' on the menu bar above in order to sort alphabetically starting with Z.	The first entry that should appear is Member ID '6925-1418-5143-5801'		
		6	The entry should be populated with this information from the original upload	BC_MemberID	BC_OrgName	BC_AddrLine1

			6925-1418-5143-5801	Business in TESTING		
		BC_AddrLine2	BC_AddrCity	BC_AddrState	BC_AddrZipCode	
		BC_AddrCountry	BC_ConPhone	BC_ConFax	BC_ConCell	
		USA				
			BC_ConEmail	BC_DateEntered	RecNo	
	7	Close table in upper right-hand corner		12/02/2005 Current Time PM	99011	
A person can search by GiftMakerPRO Record Number if the record was entered by the Uploaded/Import function.	8	Browse and select 'Permanent Contact'	Permanent Contact' table should open.			
	9	Highlight column 'RecNo', and select the Find function on the menu bar above. The Find function is a pair of binoculars. Enter '99011' in 'RecNo' field. Click FIND	The highlighted entry that should appear is Member ID '6925-1418-5143-5801'			
	10	The entry should be populated with this information from the original upload	PC_MemberID	PC_AddrLine1	PC_AddrLine2	
			6925-1418-5143-5801	3301 Shakertown Ct		
PC_AddrCity		PC_AddrState	PC_AddrZipCode	PC_AddrCountry		
Antioch		TN	37013-			
PC_ConPhone		PC_ConFax	PC_ConCell	PC_ConEmail		
(999) 999-9999						
PC_DateEntered		RecNo				
12/02/2005 Current Time PM	99011					
	11	Close table in upper right-hand corner				
A person can search by unique volunteer ID	12	Browse and select 'Volunteer Master Information'	Volunteer Master Information' table should open.			

number or unique Gift Maker PRO number.	13	Highlight column 'VMI_MemberID', and select the Find function on the menu bar above. The Find function is a pair of binoculars. Enter '6925-1418-5143-5801' in 'VMI_MemberID' field. Click FIND	The highlighted entry that should appear is Member ID '6925-1418-5143-5801'		
	14	The entry should be populated with this information from the original upload	VMI_MemberID	VMI_VolCode	VMI_DateJoined
			6925-1418-5143-5801	INVLUK	01/29/2004
		VMI_NameLast	VMI_NameMiddle	VMI_NameFirst	VMI_NamePrefix
		Stephenson-Golden		Sharing	Ms
		VMI_NameSuffix	VMI_BirthDate	VMI_MaritalStatus	VMI_PersonDeceased
					FALSE
		VMI_PreferenceAddress	VMI_PersonGender	VMI_PreferenceNewsletter	VMI_StatusActive
		TRUE	FALSE	FALSE	FALSE
		VMI_DateEntered	RecNo		
	12/02/2005 Current Time PM	19011			
	15	Close table in upper right-hand corner			
	16	Browse and select 'Business Occupation'	Business Occupation' table should open.		
	17	Highlight column 'RecNo', and select the Find function on the menu bar above. The Find function is a pair of binoculars. Enter '99011' in 'RecNo' field. Click FIND	The highlighted entry that should appear is Member ID '6925-1418-5143-5801'		
	18	The entry should be populated with this information from the original upload	BO_MemberID	BO_JobTitle	
		Close table in upper right-hand corner	6925-1418-5143-5801	Tester	
		BO_Notes	BO_DateEntered	RecNo	
			12/02/2005 Current Time PM	99011	

		19	Database window and Switchboard should be visible. Click on Switchboard.	Switchboard remains open		
	Once ImportTemp data contains a unique Volunteer ID number and is inserted in the permanent tables, it will no longer be retrieved by the 'Add New Volunteer Master Information' form.	20	The option 'Re-Enter Import Form' is an available option on the Switchboard. Once a user selects this option, the 'Add New Volunteer Master Information' form opens.	'Add New Volunteer Master Information' form opens. At bottom of form, the record selector should indicate '1 of 4'.		
0013 - The unique primary record number key from GiftMaker PRO should prevent duplicates from being imported and uploaded multiple times, and the Volunteer ID primary key should prevent duplicate numbers from being used as an ID number.	If data has already been saved in the permanent tables, the system will NOT allow other records into these tables with identical GiftMakerPRO Record Number.	1	Validate the first entry in the table.	The entry name should be:	Mr. Jason Personal	
		2	Close 'Add New Volunteer Master Information' form in upper right-hand corner. Switchboard should still be open.	Switchboard remains open		
		3	Press F11	The database window will open and will default to 'Tables'		
		4	Browse and select 'ImportTemp'	'ImportTemp' table should open and present 5 entries.		
		5	Validate the first entry in the table.	The entry name should be:	Desirea Duarte Ulibarri	
		6	Database window and Switchboard should be visible. Click on Switchboard.	Switchboard remains open		

		7	The option 'Re-Enter Import Form' is an available option on the Switchboard. Once a user selects this option, the 'Add New Volunteer Master Information' form opens.	'Add New Volunteer Master Information' form opens. At bottom of form, the record selector should indicate '1 of 4'.		
		8	Click Next until you see the entry with GiftMakerPRO Record Number '99999'	The chosen record should have user name 'Desirea Duarte Ulibarri'		
		9	Select 'Generate ID'	An ID is generated as '4094-8516-1973-7833'		
		10	Close 'Add New Volunteer Master Information' form in upper right-hand corner. Switchboard should still be open.	Switchboard remains open		
		11	Press F11	The database window will open and will default to 'Tables'		
		12	Browse and select 'Volunteer Master Information'	Volunteer Master Information' table should open.		
	If data has already been saved in the permanent tables, the system will NOT allow other records into these tables with identical MemberID numbers.	13	Highlight column 'VMI_MemberID', and select the Find function on the menu bar above. The Find function is a pair of binoculars. Enter '4094-8516-1973-7833' in 'VMI_MemberID' field. Click FIND	There should be NO entry with this Member ID.		

	14	Highlight column 'RecNo', and select the Find function on the menu bar above. The Find function is a pair of binoculars. Enter '99999' in 'RecNo' field. Click FIND	The highlighted entry that should appear is Member ID '4095-5050-3722-6623'		
	15	The entry should be populated with this information from the original upload	VMI_MemberID	VMI_VolCode	VMI_DateJoined
			4095-5050-3722-6623		07/31/2003
		VMI_NameLast	VMI_NameMiddle	VMI_NameFirst	VMI_BirthDate
		Ulibarri	Duarte	Desirea	07/01/1976
		VMI_MaritalStatus	VMI_PersonDeceased	VMI_PreferenceAddress	VMI_PersonGender
		Married	FALSE	TRUE	FALSE
		VMI_PreferenceNewsletter	VMI_StatusActive	VMI_DateEntered	RecNo
		FALSE	FALSE	11/16/2005 7:09:59 PM	99999
	16	Close table in upper right-hand corner			
	17	REPEAT STEPS 13-16 opening Business Contact, Business Occupation, and Permanent Contact.			
	18	Database window and Switchboard should be visible. Click on Switchboard.	Switchboard remains open		
	19	The option 'Re-Enter Import Form' is an available option on the Switchboard. Once a user selects this option, the 'Add New Volunteer Master Information' form opens.	'Add New Volunteer Master Information' form opens. At bottom of form, the record selector should indicate '1 of 3'.		

	20	Click Next until you see the entry with GiftMakerPRO Record Number '99100'	The chosen record should have user name 'Michael Waiter'		
	21	Select 'Generate ID'	An ID is generated as '7229-3979-7993-6936'		
	22	Close 'Add New Volunteer Master Information' form in upper right-hand corner. Switchboard should still be open.	Switchboard remains open		
	23	Press F11	The database window will open and will default to 'Tables'		
	24	Browse and select 'Volunteer Master Information'	Volunteer Master Information' table should open.		
	25	Highlight column 'RecNo', and select the Find function on the menu bar above. The Find function is a pair of binoculars. Enter '99100' in 'RecNo' field. Click FIND	There should be NO entry with this RecNo.		
	26	Highlight column 'VMI_MemberID', and select the Find function on the menu bar above. The Find function is a pair of binoculars. Enter '7229-3979-7993-6936' in 'VMI_MemberID' field. Click FIND	The highlighted entry that should appear is Member ID '7229-3979-7993-6936' with RecNo as '99052'		
	27	The entry should be populated with this information from the original upload	VMI_MemberID	VMI_VolCode	VMI_DateJoined
			7229-3979-7993-6936	INVLUK	01/30/2004
		VMI_NameLast	VMI_NameMiddle	VMI_NameFirst	VMI_BirthDate
		Waiter		Michael	

		VMI_MaritalStatus	VMI_PersonDec eased	VMI_Prefer enceAddress s	VMI_PersonGende r
			FALSE	TRUE	FALSE
		VMI_PreferenceNew sletter	VMI_StatusActiv e	VMI_DateE ntered	RecNo
		FALSE	FALSE	12/2/05 1:49 PM	99052
28	Close table in upper right-hand corner				
29	REPEAT STEPS 7- 10 opening Business Contact, Business Occupation, and Permanent Contact.				

Module 2:

Test (Use) Case Scenario	Test Description	Test Step	Step Description	Step Expected Results	Test Results	DATA SETUP REQUIREMENTS
0024-Additional volunteer user information details can be added to the upload/import process or manually.	User can add details to uploaded data.	1	Database window and Switchboard should be visible. Click on Switchboard.	Switchboard remains open		
		2	The option 'Re-Enter Import Form' is an available option on the Switchboard. Once a user selects this option, the 'Add New Volunteer Master Information' form opens.	'Add New Volunteer Master Information' form opens. At bottom of form, the record selector should indicate '1 of 2'.		
		3	Click Next until you see the entry with GiftMakerPRO Record Number '99057'	The chosen record should have user name 'Jason Personal'		
		4	Select 'Generate ID'	An ID is generated as '6149-2296-7062-0866'		
		5	Select 'Permanent Address' Select 'Business Address'	There should be no change except the check box is marked.		
		6	Highlight 'Date of Birth' and type '01-01-1901'	01/01/1901' will appear		
		7	Highlight 'Marital Status' and choose 'Single' from the drop-down list	Single' will appear		
		8	Click 'Auto-Pop' button	Permanent Address' and 'Business Contacts' forms open		
		9	On Permanent Address form, add City 'Alamosa', State 'CO' and Zip '81101'	City State and Zip should populate		
		10	Close 'Permanent Address' form in upper right-hand corner. 'Business Contacts' should still be open.	Business Contacts' remains open		
		11	On 'Business Contact' form, add Country 'USA'	Country should populate		
		12	Close 'Business Contact' form in upper right-hand corner. 'Add New Volunteer Master Information' form should still be open.	'Add New Volunteer Master Information' form remains open		
		13	Close 'Add New Volunteer Master Information' form in upper right-hand corner. Switchboard should still be open.	Switchboard remains open		
	Any data entered on the 'Add	14	Press F11	The database window will open and will default to 'Tables'		

New Volunteer Master Information' form AFTER a unique Member ID was generated is NOT entered into the permanent tables.	15	Browse and select 'Volunteer Master Information'	Volunteer Master Information' table should open.			
	16	Highlight column 'VMI_MemberID', and select the Find function on the menu bar above. The Find function is a pair of binoculars. Enter '6149-2296-7062-0866' in 'VMI_MemberID' field. Click FIND	The highlighted entry that should appear is Member ID '6149-2296-7062-0866'			
	17	Review the table entry.				
	18	VMI_Birthdate should NOT be populated. VMI_MaritalStatus should NOT be populated.				
		VMI_MemberID	VMI_VolCode	VMI_DateJoined	VMI_NameLast	
		6149-2296-7062-0866	INVLUK	01/30/2004	Personett	
		VMI_NameMiddle	VMI_NameFirst	VMI_NamePrefix	VMI_NameSuffix	
			Jason	Mr		
		VMI_MaritalStatus	VMI_PersonDeceased	VMI_PreferenceAddress	VMI_DateEntered	
			FALSE	TRUE	12/02/2005 11:34	
		VMI_PersonGender	VMI_PreferenceNewsletter	VMI_StatusActive	RecNo	
		FALSE	FALSE	FALSE	99057	
		VMI_BirthDate				
	19	VMI_Birthdate and VMI_MaritalStatus were entered on the Main form AFTER a unique Member ID was generated thus this data was NOT entered into the permanent tables.	Verify VMI_Birthdate and VMI_MaritalStatus are NOT populated.			
	20	Close table in upper right-hand corner	Table closes.			
HOWEVER, any data entered on the Auto-Pop forms IS SAVED in permanent tables even after a unique Member ID is generated.	21	Database window and Switchboard should be visible. Select 'Tables' on database window.	Tables' are available on database window.			
	22	Browse and select 'Permanent Contact'	Permanent Contact' table should open.			
	23	Highlight column 'VMI_MemberID', and select the Find function on the menu bar above. The Find function is a pair of binoculars. Enter '6149-2296-7062-0866' in 'VMI_MemberID' field. Click FIND	The highlighted entry that should appear is Member ID '6149-2296-7062-0866'			
	24	Review the table entry.				

		25	Data entered on Auto-Pop forms is SAVED in permanent tables even after a unique Member ID is generated. PC_AddrCity should be populated PC_AddrState should be populated PC_AddrZip should be populated	City = 'Alamosa' State = 'CO' Zip = '81101'		
		26	Close table in upper right-hand corner	Table closes.		
		27	Browse and select 'Business Contact'	'Business Contact' table should open.		
		28	Highlight column 'VMI_MemberID', and select the Find function on the menu bar above. The Find function is a pair of binoculars. Enter '6149-2296-7062-0866' in 'VMI_MemberID' field. Click FIND	The highlighted entry that should appear is Member ID '6149-2296-7062-0866'		
		29	Review the table entry.			
		30	Data entered on Auto-Pop forms is SAVED in permanent tables even after a unique Member ID is generated. BC_AddrCountry should be populated	Country = 'USA'		
		31	Close table in upper right-hand corner	Table closes.		
0025- There are separate look-up interfaces for volunteers and management.	Users can look up volunteer data in the Volunteer System.	1	The user opens the database fmVM and the Switchboard form opens automatically.	Switchboard remains open		
		2	The button 'Look Up Volunteer Information' is an available option on the Switchboard. Once a user selects this option, the mFRM_VolunteerLookUp form is opened in read-only mode and able to filter and retrieve volunteer information.	'Volunteer Master Information' form opens. The form default Member ID should read "1032-9721-7260-4004" as this is the first value in the table.		
		3	Type in the Member ID field.	No entry should be captured as this form is read-only. This is NOT a volunteer-edit function.		

	4	The user can select the filter function button and populate a single field for querying and lookup. Click on the 'Filter Data/Find By' button and erase any previously populated data from all fields. Enter "4095-5050-3722-6623" in the Member ID field and select 'Apply Filter' button symbol from the menu bar.	The chosen record should have user name "Desirea Duarte Ulibarri"		
	5	Once data is retrieved on this form, the user can select other options that will retrieve the filtered/selected volunteer data. Click the button 'Volunteer Time Instance'. The uFRM_VolunteerTimeInstance should open in read-only mode. All previous login times will appear with the latest time swipe appearing first.	'Volunteer Time Instance' form opens. The first entry should have a time stamp of "2/10/2006 12:58:54 PM"		
	6	The user can select "Close Form" to exit the interface.	Volunteer Time Instance' form closes.		
	7	A user can change their work day/time availability at any time by selecting the button "Available Times". This button opens the sFRM_VolunteerAvailableTimes form in edit mode. This form is filtering to only show the specific Member ID data from the VolunteerAvailableTimes table. To populate work availability, a user selects criteria from the drop-down menus.	Drop-down menus should allow changing available times. This is a volunteer-edit function.		
	8	The user can select "Close Form" on all open interfaces.	Available Times' form closes. 'Volunteer Look-up' form closes. Switchboard remains open		
Volunteer information can be added, changed, and/or deleted by	9	The user opens the database dbMGMT and enters the database password and the Switchboard form opens and available with option buttons.	Switchboard opens		

management once it is entered or uploaded and saved in the Volunteer System by using interfaces in the dbMGMT database.	10	Click on the button "Data Definitions for all Tables".	The mFRM_DataDefinitionsUtilities form opens and allows management to add, change or delete all reference table data such as project names and site locations.		
	11	Verify the following forms are available as option buttons from this form: Note: These forms are part of the original LVS.	sFRM_Projects, uFRM_DefinitionsCity, uFRM_Nomenclature_Vol Code, uFRM_OrganizationSites, uFRM_ValidAffiliations, uFRM_ValidAffiliationType, uFRM_ValidAvailableTimes, uFRM_ValidCountry, uFRM_ValidFavorites, uFRM_ValidFavoriteTypes, uFRM_ValidLevelOfInterest, uFRM_ValidLevelOfSkill, uFRM_ValidMaritalStatus, uFRM_ValidPrefix, uFRM_ValidRelationship, uFRM_ValidSpecializedTraining, uFRM_ValidSuffix, uFRM_ValidWorkCategories, uFRM_ZipCodeCityState.		
	12	The user can select "Close Form" on all open screens to exit the interfaces.	Switchboard remains open		
	13	Click on the button "Change Volunteer Information".	The mFRM_GenUniqueKey form opens and allows management to add, change, or delete any volunteer data such as permanent address, work skills, etc.		

		14	Verify the following forms are available as option buttons from this form: Note: These forms are part of the original LVS.	sFRM_BC, sFRM_BO, sFRM_PA, sFRM_VolunteerAffiliations, sFRM_VolunteerAvailableTimes, sFRM_VolunteerCountryVisited, sFRM_VolunteerFavorites, sFRM_VolunteerSkills, sFRM_VolunteerSpecializedTraining, sFRM_VolunteerWorkPreferences, sFRM_WorkCategories.		
		15	The user can select "Close Form" on all open screens to exit the interfaces.	Switchboard remains open		
		16	Click on the button "Change Volunteer Logged Time". This is a new form created specifically for the dbMGMT interface.	The sFRM_TotalHoursWorked form opens and allows management to change or delete any volunteer worked time entry.		
		17	Search for Member ID "4095-5050-3722-6623"	Highlight the first entry and press the "X" for delete on the file menu. The entry should be deleted. NOTE: This is a live interface to the TotalHoursWorked table. All entries, changes or deletions will modify live data.		
		18	The user can select "Close Form" on all open screens to exit the interfaces.	Switchboard remains open		
		19	Click the button "Swipe Card Utility". The following forms are available as option buttons from this form:	The mFRM_DataManagementUtilities form opens and allows management to create a table of volunteer names and swipe card utility numbers to export for the creation of volunteer swipe cards.		
		20	Verify the following forms are available as option buttons from this form: Note: These forms are part of the original LVS.	vFRM_BussAddr, vFRM_PermAddr, vFRM_VolMstInf, vFRM_VolunteerMasterInformation.		
		21	The user can select "Close Form" on all open screens to exit the interfaces.	All windows close and Switchboard remains open.		

		22	The user can select "Quit Volunteer Management" to exit dbMGMT.	dbMGMT closes.			
0026- The user can login the Volunteer Mgmt DB using a swipe card.	A volunteer with an encoded mag card can swipe a login.	1	The user opens the database fmVM and the Switchboard form is open and available with option buttons.	Switchboard opens			
		2	The option 'Swipe Card' is an available option on the Switchboard. Once a user selects this option, the 'Card Swipe' form is opened.	Card Swipe' form opens.			
		3	Validate default population.	The default population on the form: City Code = 'Denver' Site ID = 'International Headquarters' Project = "Attendance' Check Records is CHECKED (This will open 'Volunteer Look-up' form			
		4	Highlight Project field and Select 'Volunteer System Project' from the drop-down box.	'Volunteer System Project' is selected.			
		5	Indicate you are swiping 'IN' from the drop-down box.	'IN' is populated.			
		6	Swipe Card	'4095-5050-3722-6623' should populate the ID number field.			Volunteer mag card for 'Desirea Ulibarri'
		7	Click 'Press after swipe' button	'Volunteer Time Instance' form opens.			
		8	Validate login card swipe.	The first entry reports: Member ID = '4095-5050-3722-6623' Project = 'Volunteer System Project' City = 'Denver' Site ID = 'International Headquarters' Time Instance = 'Current' IN or OUT = 'IN' Total Hours is Blank			
		9	Click 'Close Form' button at top-right of 'Volunteer Time Instance' form	'Volunteer Time Instance' form closes. 'Volunteer Look-up' remains open.			
		10	Click 'Close Form' button at top-right of 'Volunteer Look-up' form	Switchboard remains opens			
	A user cannot log IN twice in a row.	11	The option 'Swipe Card' is an available option on the Switchboard. Once a user selects this option, the 'Card Swipe' form is opened.	Card Swipe' form opens.			

		12	Validate default population.	The default population on the form: City Code = 'Denver' Site ID = 'International Headquarters' Project = "Attendance' Check Records is CHECKED (This will open 'Volunteer Look-up' form		
		13	Indicate you are swiping 'IN' from the drop-down box.	'IN' is populated.		
		14	Swipe Card	'4095-5050-3722-6623' should populate the ID number field.		
		15	Click 'Press after swipe' button	An error message will pop-up that says 'You have already logged IN' press OK.		
		16	Click 'OK' on error message	Error message pop-up closes. 'Volunteer Time Instance' form opens.		
		17	NO login is captured. The 'Volunteer Time Instance' form opens to verify that no entry was made.	'Volunteer Time Instance' form opens. The first entry reports: Member ID = '4095-5050-3722-6623' Project = 'Volunteer System Project' City = 'Denver' Site ID = 'International Headquarters' Time Instance = ' NOT Current ' IN or OUT = 'IN' Total Hours is Blank		
		18	Click 'Close Form' button at top-right of 'Volunteer Time Instance' form	'Volunteer Time Instance' form closes. 'Card Swipe' form remains open.		
		19	Click 'Close Form' button at top-right of 'Card Swipe' form	'Card Swipe' form closes.		
0027 - The user can login the Volunteer Mgmt DB by typing their Member ID number (no card).	A volunteer without an encoded mag card can enter a login.	1	The user opens the database fmVM and the Switchboard form is open and available with option buttons.	Switchboard opens		
		2	The option 'Swipe Card' is an available option on the Switchboard. Once a user selects this option, the 'Card Swipe' form is opened.	Card Swipe' form opens.		

		3	Validate default population.	The default population on the form: City Code = 'Denver' Site ID = 'International Headquarters' Project = "Attendance" Check Records is CHECKED (This will open 'Volunteer Look-up' form		
		4	Indicate you are swiping 'IN' from the drop-down box.	'IN' is populated.		
		5	Type in this Member ID: 7229-3979-7993-6936	'7229-3979-7993-6936' should populate the ID number field.		
		6	Click 'Press after swipe' button	'Volunteer Time Instance' form opens.		
		7	Validate login card swipe.	The first entry reports: Member ID = '7229-3979-7993-6936' Project = 'Attendance' City = 'Denver' Site ID = 'International Headquarters' Time Instance = 'Current' IN or OUT = 'IN' Total Hours is Blank		
		8	Click 'Close Form' button at top-right of 'Volunteer Time Instance' form	'Volunteer Time Instance' form closes. 'Volunteer Look-up' form remains open.		
		9	Click 'Close Form' button at top-right of 'Volunteer Look-up' form	Switchboard remains opens		
0028- The user can logout of the Volunteer Mgmt DB using a swipe card.	A volunteer with an encoded mag card can swipe a logout.	1	The option 'Swipe Card' is an available option on the Switchboard. Once a user selects this option, the 'Card Swipe' form is opened.	Card Swipe' form opens.		
		2	Validate default population.	The default population on the form: City Code = 'Denver' Site ID = 'International Headquarters' Project = "Attendance" Check Records is CHECKED (WHEN logging OUT, this will NOT open 'Volunteer Look-up' form		
		3	Highlight Project field and Select 'Volunteer System Project' from the drop-down box.	'Volunteer System Project' is selected.		Volunteer mag card for 'Desirea Ulibarri'
		4	Indicate you are swiping 'OUT' from the drop-down box.	'OUT' is populated.		

	5	Swipe Card	'4095-5050-3722-6623' should populate the ID number field.		
	6	Click 'Press after swipe' button	'Volunteer Time Instance' form opens. A pop-up box opens on top that says 'Thank you for volunteering xx:xx hours/minutes!'		
	7	Close 'Your Volunteered Time' pop-up.	'Volunteer Time Instance' form remain opens.		
	8	Validate logout card swipe.	The first entry reports: Member ID = '4095-5050-3722-6623' Project = 'Volunteer System Project' City = 'Denver' Site ID = 'International Headquarters' Time Instance = 'Current' IN or OUT = 'OUT' Total Hours is Blank		
	9	Click 'Close Form' button at top-right of 'Volunteer Time Instance' form	'Volunteer Time Instance' form closes. Switchboard remains opens		
A user cannot log OUT twice in a row.	10	The option 'Swipe Card' is an available option on the Switchboard. Once a user selects this option, the 'Card Swipe' form is opened.	Card Swipe' form opens.		
	11	Validate default population.	The default population on the form: City Code = 'Denver' Site ID = 'International Headquarters' Project = "Attendance" Check Records is CHECKED (WHEN logging OUT, this will NOT open 'Volunteer Look-up' form		
	12	Highlight Project field and Select 'Volunteer System Project' from the drop-down box.	'Volunteer System Project' is selected.		
	13	Indicate you are swiping 'OUT' from the drop-down box.	'OUT' is populated.		
	14	Swipe Card	'4095-5050-3722-6623' should populate the ID number field.		
	15	Click 'Press after swipe' button	An error message will pop-up that says 'You have already logged OUT'.		

		16	Click 'OK' on error message	Error message pop-up closes. 'Volunteer Time Instance' form opens.		
		17	NO logout is captured. The 'Volunteer Time Instance' form opens to verify that no entry was made.	'Volunteer Time Instance' form opens. The first entry reports: Member ID = '4095-5050-3722-6623' Project = 'Volunteer System Project' City = 'Denver' Site ID = 'International Headquarters' Time Instance = 'NOT Current' IN or OUT = 'OUT' Total Hours is Blank		
		18	Click 'Close Form' button at top-right of 'Volunteer Time Instance' form	'Volunteer Time Instance' form closes. 'Card Swipe' form remains open.		
		19	Click 'Close Form' button at top-right of 'Card Swipe' form	'Card Swipe' form closes.		

Module 4:

Test (Use) Case Scenario	Test Description	Test Step	Step Description	Step Expected Results	Test Results	
0049- Robust reporting on any attribute of volunteer data housed in the Volunteer System is available.	Any retrieved data on all forms and can be easily exported by a user into Excel or Word documents for quick manipulation and reporting.	1	The user opens the database fmVM and the Switchboard form opens automatically.	Switchboard remains open		
		2	The button 'Look Up Volunteer Information' is an available option on the Switchboard. Once a user selects this option, the mFRM_VolunteerLookUp form is opened in read-only mode and able to filter and retrieve volunteer information.	'Volunteer Master Information' form opens. The form default Member ID should read "1032-9721-7260-4004" as this is the first value in the table.		
		3	The user can select the filter function button and populate a single field for querying and lookup. Click on the 'Filter Data/Find By' button and erase any previously populated data from all fields. Enter " 4095-5050-3722-6623" in the Member ID field and select 'Apply Filter' button symbol from the menu bar.	The chosen record should have user name "Desirea Duarte Ulibarri"		
		4	Once data is retrieved on this form, the user can select other options that will retrieve the filtered/selected volunteer data. Click the button 'Volunteer Time Instance'. The uFRM_VolunteerTimeInstance should open in read-only mode. All previous login times will appear with the latest time swipe appearing first.	'Volunteer Time Instance' form opens. The first entry should have a time stamp of "2/10/2006 12:58:54 PM"		

	5	The user selects 'Office Links' from the main file menu at top right of the form to open the filtered data in either Word or Excel. 'Office Links' will immediately open a document and populate the new data. Either method is available for saving the data to a preferred user location. Select 'Analyze it with Microsoft Office Excel'.	The data should all output and immediately open in an Excel spreadsheet and report as below:		
	6	The user can close or save the report in Excel.	VTI_TimeDirectionID	VTI_MemberID	VTI_ProjectID
IN			4095-5050-3722-6623	Attendance	
VTI_CityCode			VTI_SiteID	VTI_Instance	
		Denver	International Headquarters	02/10/2006 12:58	
	7	The user selects 'Export' which will first prompt the user to save the data to a disk location. Select 'Desktop', name the file "Test 0040 Step 7" and format as 'Microsoft Excel 97-2003'	A pop-up box will open and prompt the user to save the data to a location. Click 'Export All' to complete the process.		
	8		The data should immediately open in an Excel spreadsheet and report as below:		
			VTI_TimeDirectionID	VTI_MemberID	VTI_ProjectID
			IN	4095-5050-3722-6623	Attendance
	9	Go to Desktop and open "Test 0040 Step 7.xls"	VTI_CityCode	VTI_SiteID	VTI_Instance
			Denver	International Headquarters	02/10/2006 12:58
		Close the report.	'Volunteer Time Instance' form closes. 'Volunteer Look-up' remains open.		
	10	Click 'Close Form' button at top-right of 'Volunteer Time Instance' form	Switchboard remains open		
	11	Click 'Close Form' button at top-right of 'Volunteer Look-up' form			
	12	The user can select "Quit Volunteer Mgmt" to exit the interface.	Switchboard remains open		
Formatted reporting is available for users to	13	The user opens the database fmVM and the Switchboard form opens automatically.	Create Your Own Report' form opens.		

	<p>retrieve volunteer work time data based on entered criteria and summary options. The available criteria are by date range, by project, by site, by city, or by volunteer. The output is summarized by date, by volunteer, and by project and sorted alphabetically by volunteer's last name. Formatted reporting can be easily exported by a user into Excel or Word documents for quick manipulation and reporting.</p>	<p>14</p>	<p>The button "Create Reports" is an available option on the Switchboard. Once a user selects this option, the uFRM_CreateReport form is opened.</p>	<p>The RPT_DateRange report should automatically open with the heading "Volunteer Work Hours" . The report is alphabetized by volunteer last name and categorizes work hours by each member, date, and project category. Work hours are totaled and displayed for each day, for the total date range (which is displayed in the upper right-hand corner), and for the entire report. The work hour totals for each date are displayed in hours:minutes. The date range should be "01/01/03" to today's date.</p>		
		<p>15</p>	<p>A user can select different tabs at the top of the form that represent the available reporting criteria. The tab "Volunteer Work Time by Date Range" allows Step 1) input of a date range, Step 2) the option to run the report for all volunteers OR Step 3) select alternate criteria on the other tabs. The date range fields are unbound and default to the current date automatically when the form is opened. CHANGE the beginning date to "01/01/03" and CLICK the button on this page "Step 2) Report for All Volunteers using this date range - Click Here:"</p>	<p>The RPT_DateRange report opens and is filtered to only show the selected volunteer name. All summaries pertain to the selected volunteer only.</p>		

		16	Close report in upper right-hand corner and go back to 'Create your own report' form. Click on tab 'By Volunteer' and choose "Desirea Ulibarri" as an available volunteer from the drop-down menu. Click on "and Click Here".	The RPT_DateRange report opens and is filtered to only show the selected project. All summaries pertain to the selected project only.		
		17	Close report in upper right-hand corner and go back to 'Create your own report' form. Click on tab 'By Project' and choose "Volunteer System Project" as available project from the drop-down menu. Click on "and Click Here".	The RPT_DateRange report opens and is filtered to only show the selected organization site. All summaries pertain to the selected organization site only.		
		18	Close report in upper right-hand corner and go back to 'Create your own report' form. Click on tab 'By Organization Site' and choose "Testing Site" as from the drop-down menu. Click on "and Click Here".	The RPT_DateRange report opens and is filtered to only show the selected city. All summaries pertain to the selected city only.		
		19	Close report in upper right-hand corner and go back to 'Create your own report' form. Click on tab 'By City' and choose "Nashville" from the drop-down menu. Click on "and Click Here".			
		20	The user can select either "Office Links" or "Export" from the main file menu at the top of the form to export the filtered data to either Word or Excel. "Office Links" will immediately open a document and populate the new data, while "Export" will first prompt the user to save the data to a disk location. Either method is available for saving the data to a preferred user location.	'Create Your Own Report' form closes. 'Switchboard remains open.		
		21	Click 'Close Form' button at top-right of 'Create Your Own Report' form	'Card Swipe' form closes.		
		22	Click 'Close Form' button at top-right of 'Card Swipe' form			

Module 5:

Test (Use) Case Scenario	Test Description	Test Step	Step Description	Step Expected Results	Test Results
0050- Change formatted reporting to include "Organization Group"	The report RPT_DateRange will be enhanced to include the field "Organization Group" as part of the volunteer name header.	1	The user opens the database fmVM and the Switchboard form opens automatically.	Switchboard remains open	
		2	The button "Create Reports" is an available option on the Switchboard. Once a user selects this option, the uFRM_CreateReport form is opened.	'Create Your Own Report' form opens.	
		3	Close report in upper right-hand corner and go back to 'Create your own report' form. Click on tab 'By Organization Group' and choose "Sprint Nextel" as an available group name from the drop-down menu. Click on "and Click Here".	The RPT_DateRange report should automatically open with the heading "Volunteer Work Hours". The report is alphabetized by volunteer last name and categorizes work hours by each member, date, and project category. Work hours are totaled and displayed for Desirea Duarte Ulibarri with Group Name "Sprint Nextel"	
		4	Close report in upper right-hand corner and go back to 'Create your own report' form. Click 'Close Form' button at top-right of 'Create Your Own Report' form.	'Create Your Own Report' form closes. 'Switchboard remains open.	
0051 - Change the sequence of login and log-out user messages and pop-up screens:	Login sequence with no Checked Records.	1	The option 'Swipe Card' is an available option on the Switchboard. Once a user selects this option, the 'Card Swipe' form is opened.	Card Swipe' form opens.	
		2	Validate default population. UNCHECK "Check Records"	The default population on the form: City Code = 'Denver' Site ID = 'International Headquarters' Project = "Attendance"	
		3	Indicate you are swiping 'IN' from the drop-down box.	'IN' is populated.	
		4	Type in this Member ID: "4095-5050-3722-6623"	'4095-5050-3722-6623' should populate the ID number field.	
		5	Click 'Press after swipe' button	A new message "Thank you for logging IN!" should appear with no other screen openings.	
		6	Click 'OK' button on the pop-up message.	Pop-up closes. Switchboard remains open.	
	Login sequence when logging twice in a row.	7	The option 'Swipe Card' is an available option on the Switchboard. Once a user selects this option, the 'Card Swipe' form is opened.	Card Swipe' form opens.	

	8	Validate default population. Validate "Check Records" is selected.	The default population on the form: City Code = 'Denver' Site ID = 'International Headquarters' Project = "Attendance"	
	9	Indicate you are swiping 'IN' from the drop-down box.	'IN' is populated.	
	10	Type in this Member ID: "4095-5050-3722-6623"	'4095-5050-3722-6623' should populate the ID number field.	
	11	Click 'Press after swipe' button	An error message will pop-up that says 'You have already logged IN' press OK. No other screens open.	
	12	Click 'OK' on error message	Error message pop-up closes.	
Logout sequence.	13	Enter a logout for the same volunteer.	A pop-up box opens on top that says 'Thank you for volunteering xx:xx hours/minutes! No other screen openings should occur.	
	14	Close 'Your Volunteered Time' pop-up.	Switchboard remains open.	
Logout twice in a row.	15	Enter ANOTHER logout for the same volunteer.	An error message will pop-up that says 'You have already logged OUT' No other screen openings should occur.	
	16	Click 'OK' on error message.	'Card Swipe' form remains opens.	
	17	Click 'Close Form' button at top-right of 'Card Swipe' form.	'Card Swipe' form closes. Switchboard remains open.	
Login sequence with Checked Records.	18	The option 'Swipe Card' is an available option on the Switchboard. Once a user selects this option, the 'Card Swipe' form is opened.	Card Swipe' form opens.	
	19	Validate default population. Validate "Check Records" is selected.	The default population on the form: City Code = 'Denver' Site ID = 'International Headquarters' Project = "Attendance" Check Records is CHECKED (This will STILL open 'Volunteer Look-up' form)	
	20	Indicate you are swiping 'IN' from the drop-down box.	'IN' is populated.	
	21	Type in this Member ID: "4095-5050-3722-6623"	'4095-5050-3722-6623' should populate the ID number field.	
	22	Click 'Press after swipe' button	A new message "Thank you for logging IN!" should appear. 'Volunteer Look-up' form will open and present data about the volunteer who is logging in. No other screen openings should occur.	
	23	Click 'Close Form' button at top-right of 'Volunteer Look-up' form	Switchboard remains opens	

Appendix F: Volunteer System Training Manual

Prepared by Desirea Duarte Ulibarri

For Michelle Sanders

Version 5.0

Distributed February 2006

Lesson 1: How to import a file from GiftMaker Pro

Step #1

- Generate an extract in GiftMaker Pro (using a predetermined logic for date parameters).
- The extract must be in a Microsoft Excel or a .csv file format and should contain the following columns: See attached:

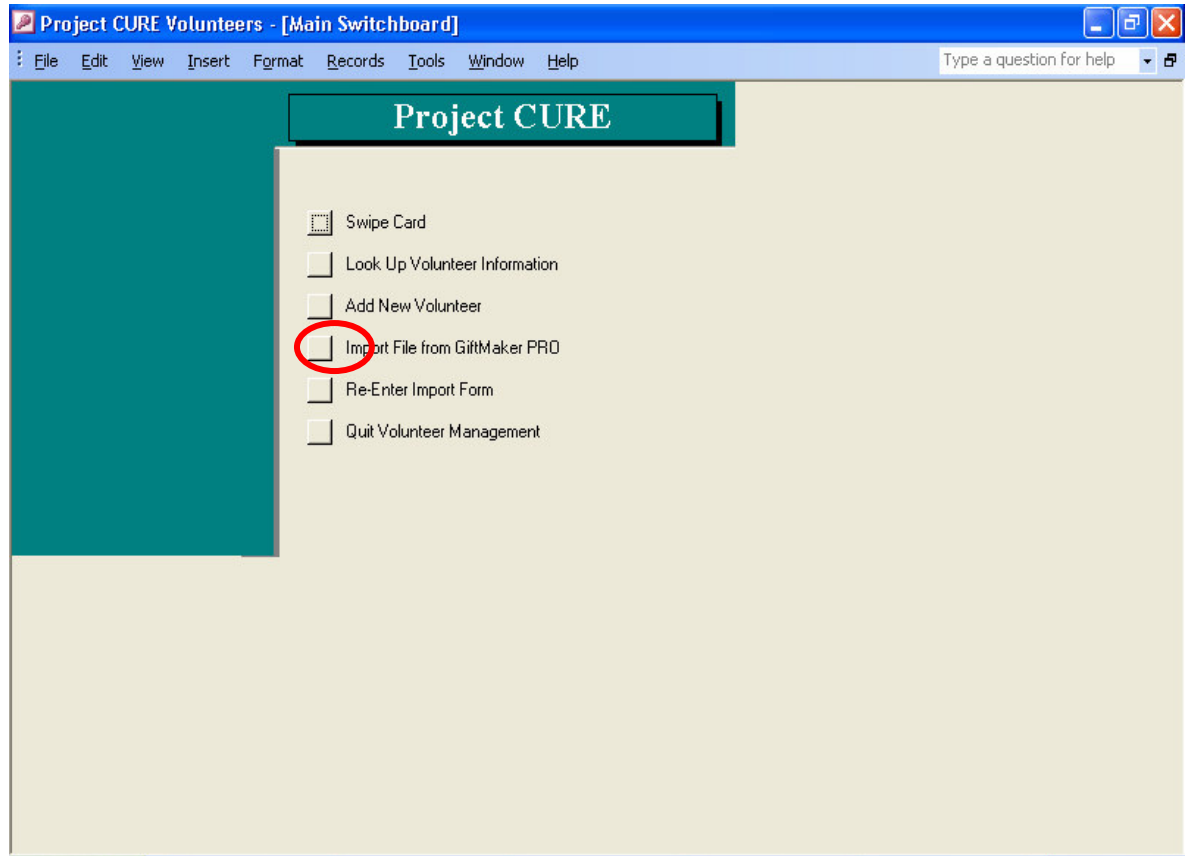


Sample
GiftMakerPRO extract

- Save file to a specific location.

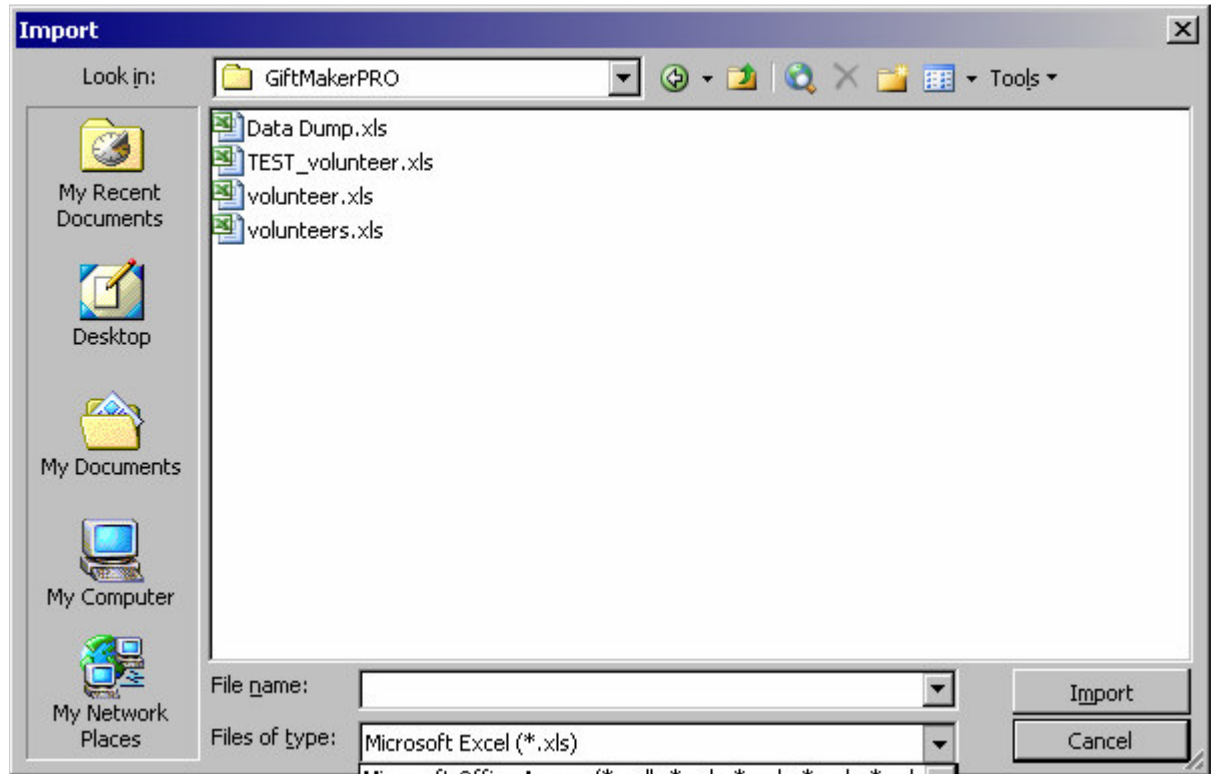
Step #2

- Open the 'Volunteer System' database.
- In the Main Menu, select 'Import File from GiftMaker PRO'.

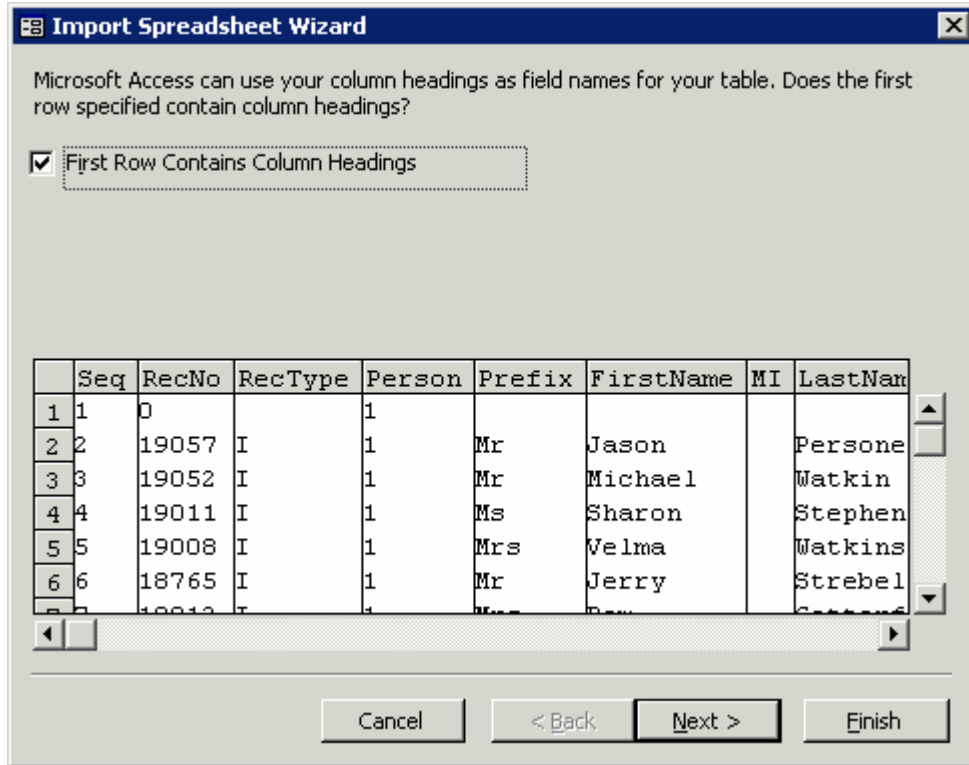


Step #3

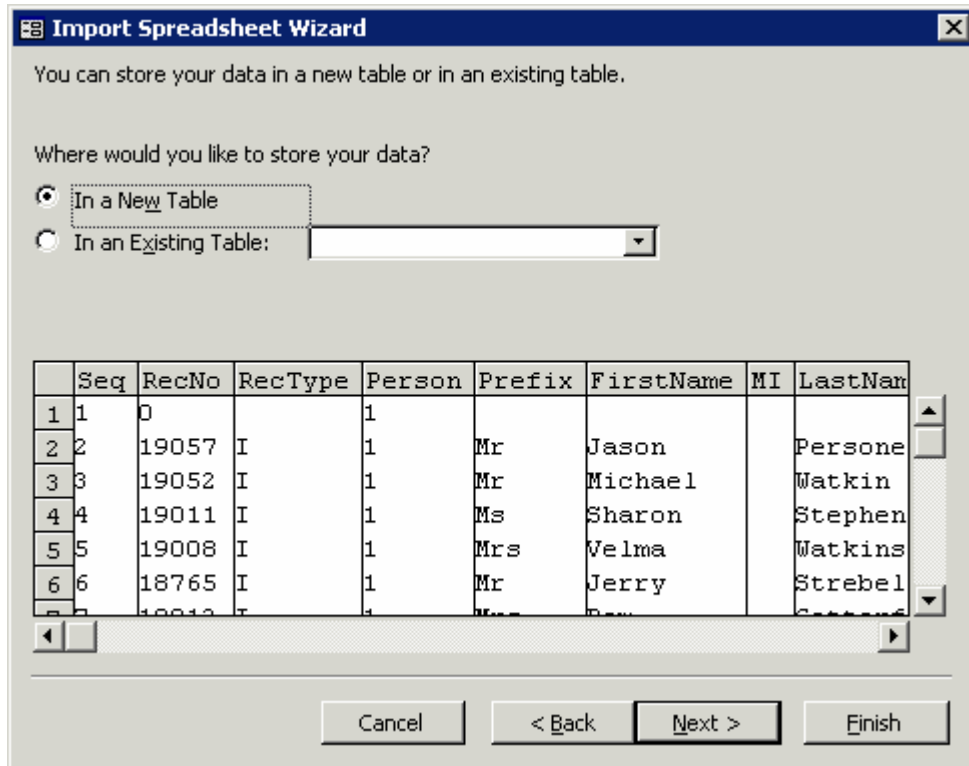
- Follow the wizard for importing the table.
- When browsing for GiftMaker Pro file, make sure that the 'Files of Type' match the file type you are searching (e.g. Text or Microsoft Excel)



- When prompted, select ‘First Row Contains Column Headings’.



- When prompted, select to store the data ‘In a New Table’.



- Click 'Next' through the next two screens:

Import Spreadsheet Wizard

You can specify information about each of the fields you are importing. Select fields in the area below. You can then modify field information in the 'Field Options' area.

Field Options

Field Name: Data Type:

Indexed: Do not import field (Skip)

	Seq	RecNo	RecType	Person	Prefix	FirstName	MI	LastNam
1	1	0		1				
2	2	19057	I	1	Mr	Jason		Person
3	3	19052	I	1	Mr	Michael		Watkin
4	4	19011	I	1	Ms	Sharon		Stephen
5	5	19008	I	1	Mrs	Velma		Watkins
6	6	18765	I	1	Mr	Jerry		Strebel

Cancel < Back **Next >** Finish

Import Spreadsheet Wizard

Microsoft Access recommends that you define a primary key for your new table. A primary key is used to uniquely identify each record in your table. It allows you to retrieve data more quickly.

Let Access add primary key.

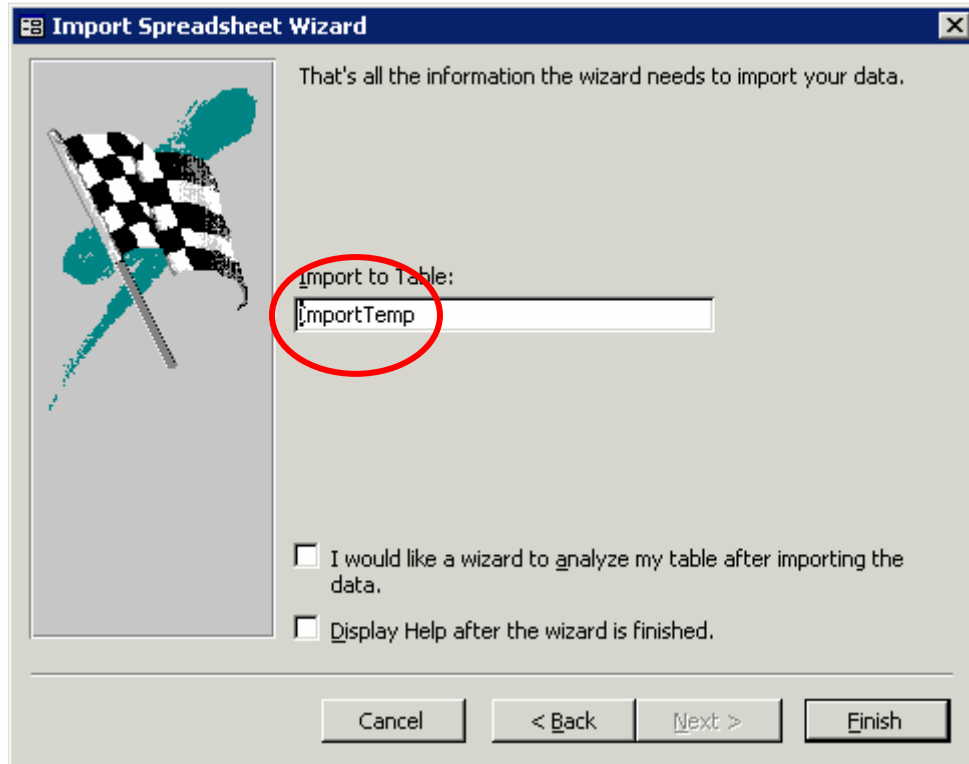
Choose my own primary key.

No primary key.

	ID	Seq	RecNo	RecType	Person	Prefix	FirstName	MI	Le
1	1	1	0		1				
2	2	2	19057	I	1	Mr	Jason		Pe
3	3	3	19052	I	1	Mr	Michael		Wa
4	4	4	19011	I	1	Ms	Sharon		St
5	5	5	19008	I	1	Mrs	Velma		Wa
6	6	6	18765	I	1	Mr	Jerry		St

Cancel < Back **Next >** Finish

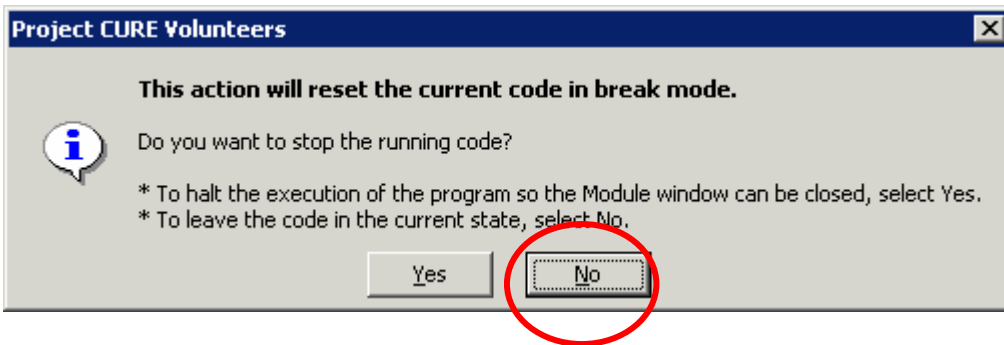
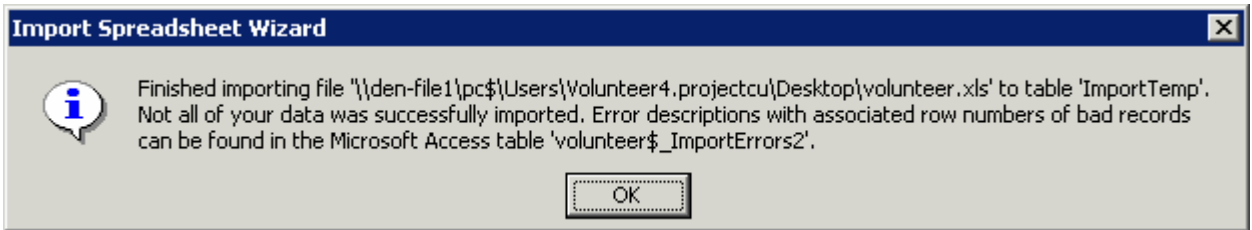
- When prompted, save the import table with the name **ImportTemp**. Do not use any spaces or quotation marks.



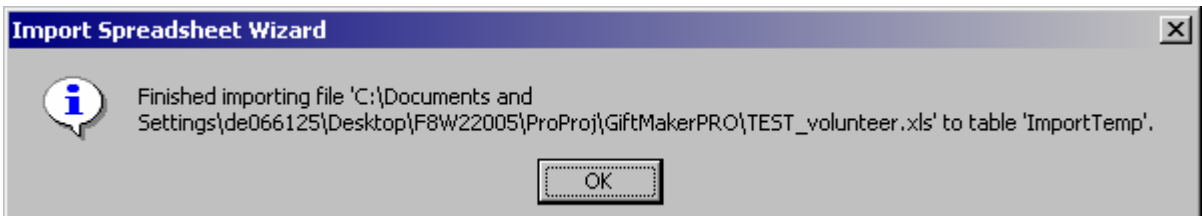
- Click 'Yes' to overwrite any existing table.

NOTE: This will effectively overwrite any temporary information that was previously uploaded. However, entries that have a unique number assignment are complete and in permanent tables.

NOTE: You may periodically get an error message that states there will be a 'Break in Code' due to import errors. This would only be in the case that the data is not complete for every column in the file. Please select 'No' when asked to 'Stop the running code'.



- When you see this message below, select 'OK' as the file has been imported.



Step #4

- Once the table is successfully imported, this screen will open automatically:

Project CURE. Add New Volunteer Information

19057 GiftMaker PRO Record Number Generate ID Auto Pop Forms

Volunteer Master Information

Member ID	Volunteer Code	Date Joined	Date of Birth	Marital Status
19057	INVLUK	01/30/2004	07/01/1976	Married

Prefix: Mr | First Name: Jason | Middle Name: | Last Name: Personett | Suffix: |

City: Anywhere | State: FL | Phone: () . .

Set Up Vol Code

IN	Level 01
VL	Level 02
UK	Level 03

Volunteer Flags

- Gender (Male?)
- Active Volunteer:
- Deceased?
- Business Address
- Email Newsletter:

Other Volunteer Information

- Permanent Address
- Business Address
- Other Business Information
- Special Training Received

Volunteer Involvement

- Available Times
- Select Work Preferences
- Select Volunteer Skills

Volunteer Interest

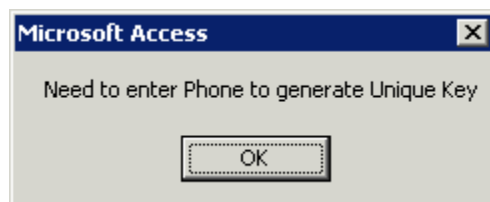
- Select Volunteer Favorites
- Select Volunteer Affiliations
- Select Country Visited

Record: 2 of 155

- The imported information should appear one record at a time. You can add more volunteer information or any other missing attribute (if applicable). Until a unique Volunteer identification number is generated and assigned, you cannot open save information in any accompanying additional forms located as buttons across the form, e.g. Permanent Address, Business Address, etc.
- If the form is populated with 'First Name', 'Last Name', 'City' 'State' and 'Phone' then you can generate a Member ID for each new volunteer by clicking on 'Generate ID'.
- Click on 'Generate ID' for each volunteer. Once a Member ID generates, click on the Next arrow to scroll to the next volunteer entry until all newly added volunteers have Member ID numbers.

The screenshot shows the 'Project CURE Volunteers - [Add New Volunteer Master Information]' window. At the top, there is a 'Generate ID' button circled in red. Below it, the 'Volunteer Master Information' section contains fields for Member ID (19057), Volunteer Code (INVLUK), Date Joined (01/30/2004), Date of Birth (07/01/1976), and Marital Status (Married). The name fields are filled with 'Mr Jason Personett'. The State is 'FL' and the City is 'Anywhere'. The Phone field is empty. A red arrow points to the Phone field. To the right, there are 'Set Up Vol Code' and 'Volunteer Flags' sections. At the bottom, there are 'Other Volunteer Information', 'Volunteer Involvement', and 'Volunteer Interest' sections with checkboxes and buttons.

- NOTE: You will get an error when trying to generate a Volunteer ID if the volunteer does not have 'First Name', 'Last Name', 'City' 'State', and 'Phone'.



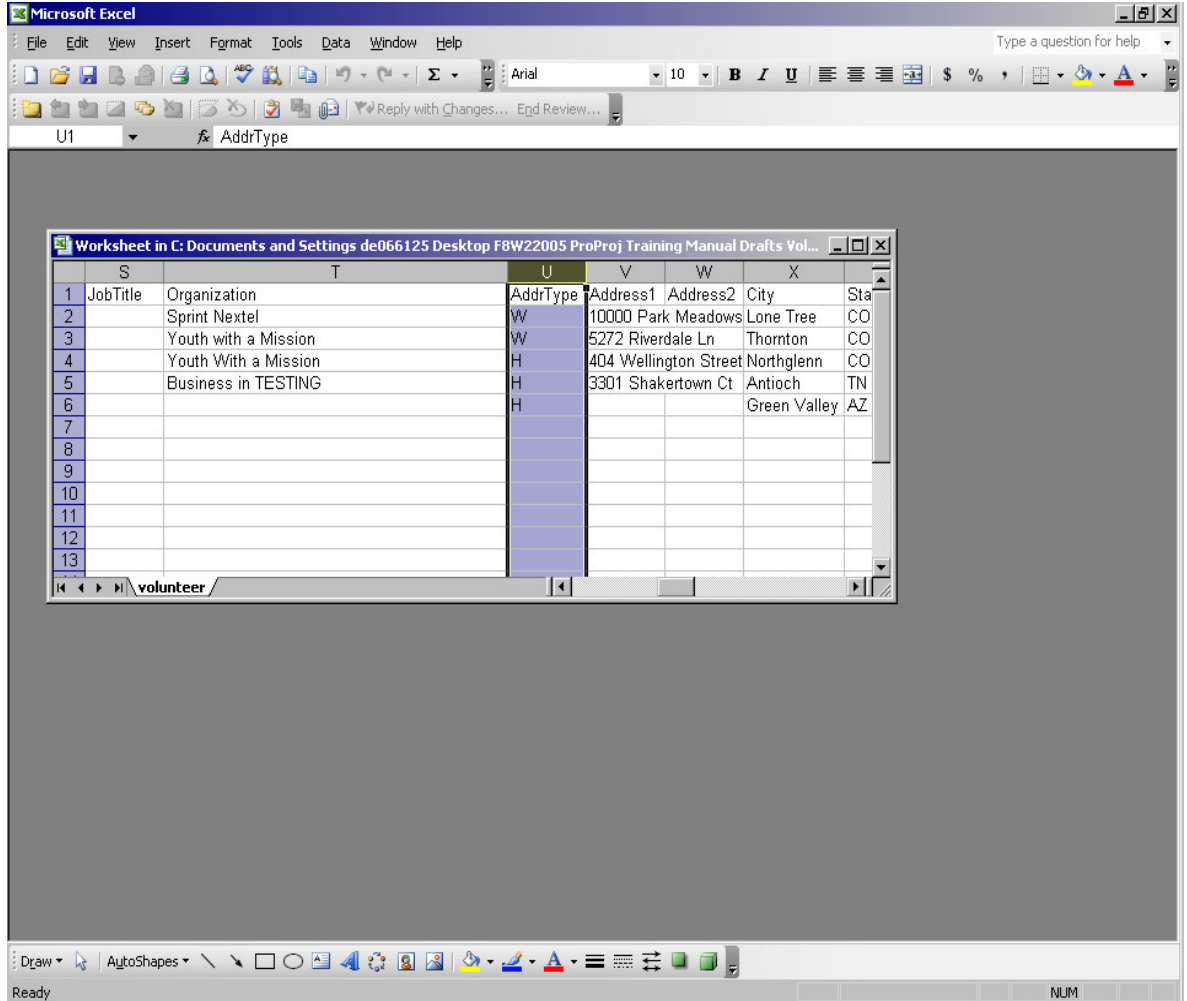
- If the form is NOT populated with 'First Name', 'Last Name', 'City' 'State' or 'Phone' then you MUST manually populate these fields to generate an ID. For these incomplete records, you can type in the required information or skip the

record entirely by clicking on the Next arrow. These records will remain in form view until a Volunteer ID is created.

WARNING: If you import a new batch of volunteer entries from GiftMaker Pro, you will overwrite these incomplete entries. To prevent losing unsaved information, please fill in the necessary information and create a Member ID as soon as possible.

Step #5

- After a Member ID is generated, the database stores information from this form into permanent tables including Permanent Address, Business Address, etc.
- If the GiftMakerPRO data ‘AddrType’ has an “H” indicating a home address, the Permanent Address information is saved; if the indicator is “W”, then the Business Address information is saved. Job and personal details are also saved if the data is available in the GiftMakerPRO file.



- Click on the check boxes next to these categories to select different forms to pre-populate Member ID and retrieve this saved data. Click on 'Auto Pop Forms' to open multiple forms simultaneously or click on the button to the right of each category to open individually.

Project CURE. Add New Volunteer Information

99999 GiftMaker PRO Record Number Generate ID Auto Pop Forms

Volunteer Master Information

Member ID	Volunteer Code	Date Joined	Date of Birth	Marital Status
	INVLUK	01/21/2004		
Prefix	First Name	Middle Name	Last Name	Suffix
	Desirea	Duarte	Ulibarri	
City	State	Phone		
Lone Tree	CO	303-419-4412		

Set Up Vol Code

IN	Level 01
VL	Level 02
UK	Level 03

Volunteer Flags

<input type="checkbox"/>	Gender (Male?)
<input type="checkbox"/>	Active Volunteer?
<input type="checkbox"/>	Deceased?
<input type="checkbox"/>	Business Address?
<input type="checkbox"/>	Email Newsletter?

Other Volunteer Information

<input checked="" type="checkbox"/>	<input type="button" value="↔"/>	Permanent Address
<input checked="" type="checkbox"/>	<input type="button" value="↔"/>	Business Address
<input type="checkbox"/>	<input type="button" value="↔"/>	Other Business Information
<input type="checkbox"/>	<input type="button" value="↔"/>	Special Training Received

Volunteer Involvement

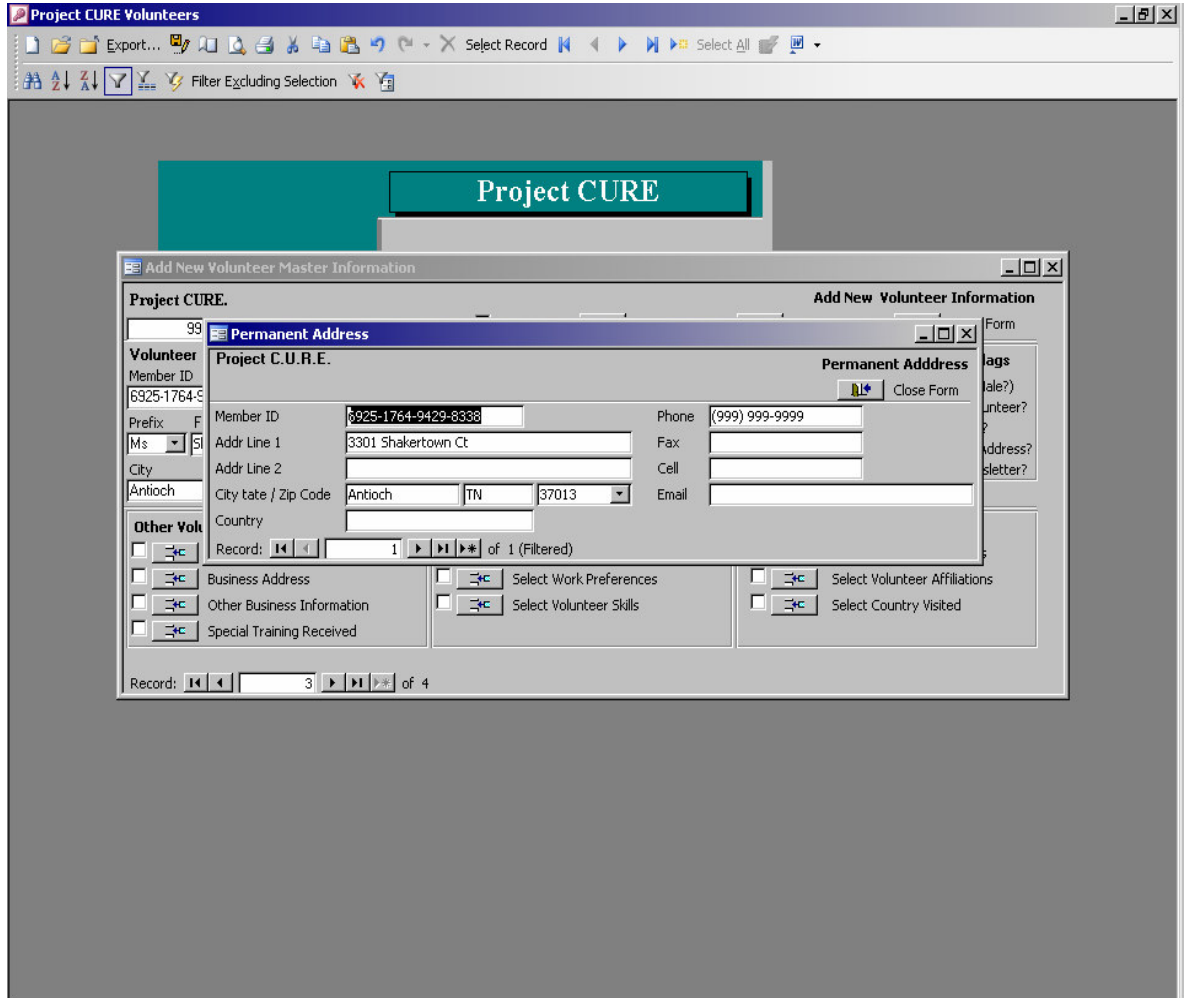
<input type="checkbox"/>	<input type="button" value="↔"/>	Available Times
<input type="checkbox"/>	<input type="button" value="↔"/>	Select Work Preferences
<input type="checkbox"/>	<input type="button" value="↔"/>	Select Volunteer Skills

Volunteer Interest

<input type="checkbox"/>	<input type="button" value="↔"/>	Select Volunteer Favorites
<input type="checkbox"/>	<input type="button" value="↔"/>	Select Volunteer Affiliations
<input type="checkbox"/>	<input type="button" value="↔"/>	Select Country Visited

Record: 1 of 5

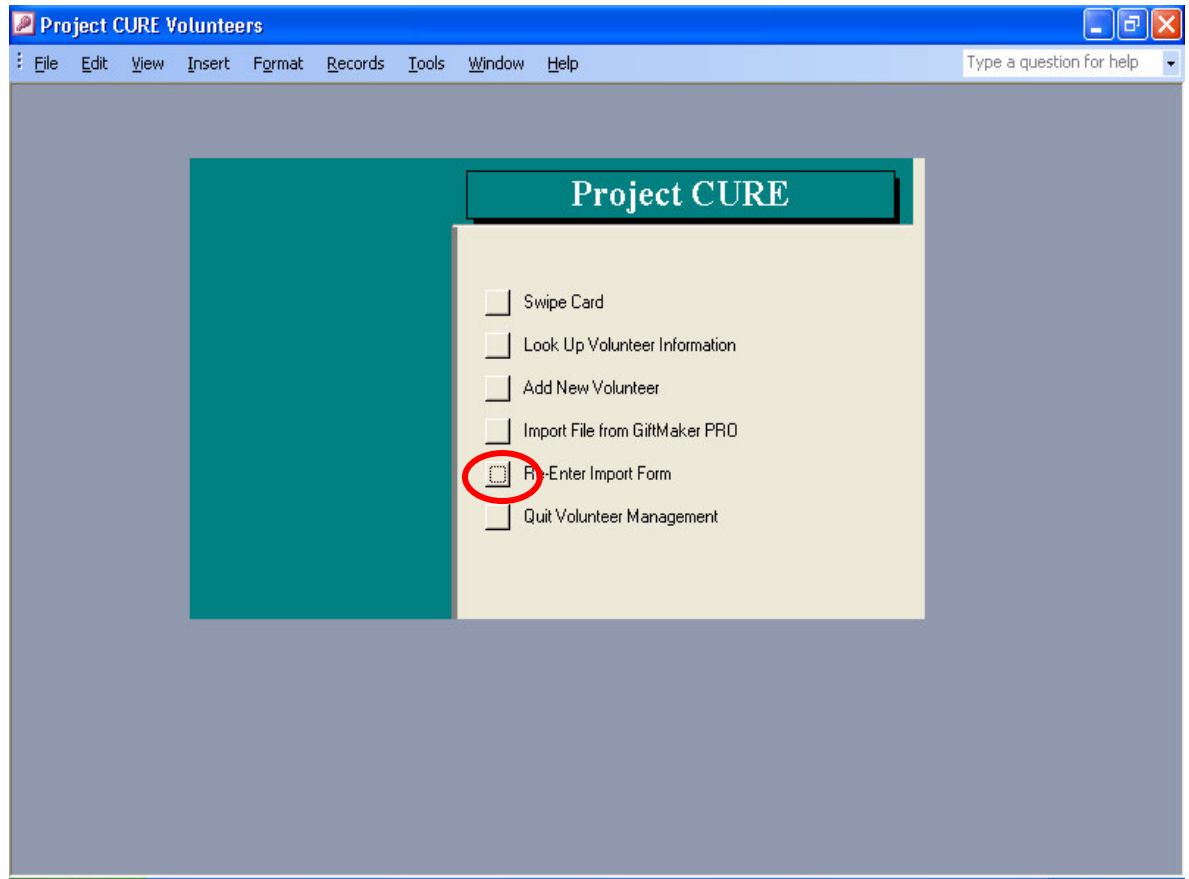
- Although data may be pre-populated, you can also enter additional information on these additional forms at any time.



NOTE: Any data entered on the main 'Add New Volunteer Master Information' form AFTER a Member ID is generated is NOT entered in the master tables. HOWEVER, any data entered on any of the additional forms, such as "Permanent Address" or "Business Address" IS SAVED in permanent tables even after a Member ID is assigned.

Step #6

- Once you exit this form, you can get back to it via the main menu:



Only records that have no assigned Member ID can be retrieved.

Lesson 2: How to swipe card or manually record time

Step #1 – Swipe “IN”

- Open the ‘Volunteer System’ database.
- In the Main Menu, select ‘Swipe Card’.

Step #2

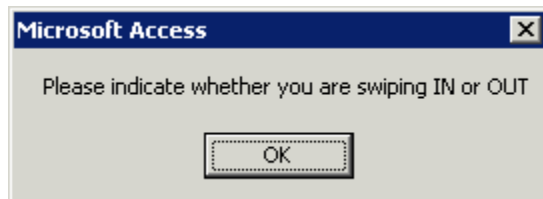
- Select ‘City’ of the location where you volunteer work from the drop-down menu.
- The ‘Site ID’ should default and populate. If it does not, select the Site where you volunteer work.
- The ‘Project’ should default automatically to ‘Attendance’. You may select other projects from the drop-down menu.
- Select ‘IN’.

The screenshot shows a Microsoft Access form titled "Project CURE Volunteers - [Card Swipe]". The form has a menu bar (File, Edit, View, Insert, Format, Records, Tools, Window, Help) and a toolbar. The main area contains the following elements:

- Project CURE** label.
- City Code:** A dropdown menu set to "Denver" with a checkmark icon.
- Site ID:** A dropdown menu set to "International Headquarters".
- Project:** A dropdown menu set to "Attendance".
- Check Records?:** A checkbox that is checked.
- Close Form:** A button with a blue arrow icon.
- Please indicate if you are swiping IN or OUT:** A dropdown menu set to "IN".
- Swipe your card now... or enter your ID#:** A text input field.
- Press after swipe:** A button with a clock icon.

At the bottom of the form, it says "Form View" on the left and "NUM" on the right.

- You cannot leave this field blank or you will receive an error notice.



Step #3

- Swipe your card through the swipe strip reader. IF YOU DO NOT HAVE A SWIPE CARD, SKIP TO STEP #4.
- When your number populates the ID number field, click on 'Press after swipe'.

The screenshot shows the 'Project CURE Volunteers - [Card Swipe]' application window. The menu bar includes File, Edit, Insert, Records, Window, and Help. A search bar on the right contains the text 'Type a question for help'. The form is divided into several sections:

- Project CURE** section: City Code is set to 'Denver' with a dropdown arrow and a checked 'Check Records?' checkbox. Site ID is 'International Headquarters' and Project is 'Attendance', both with dropdown arrows.
- A 'Close Form' button with a paper airplane icon is located to the right of the Project CURE section.
- Please indicate if you are swiping IN or OUT:** A dropdown menu is set to 'IN'.
- Swipe your card now... or enter your ID#:** The text input field contains the number '4095-5050-3722-6623'.
- A 'Press after swipe' button with a clock icon is located to the right of the ID number field.

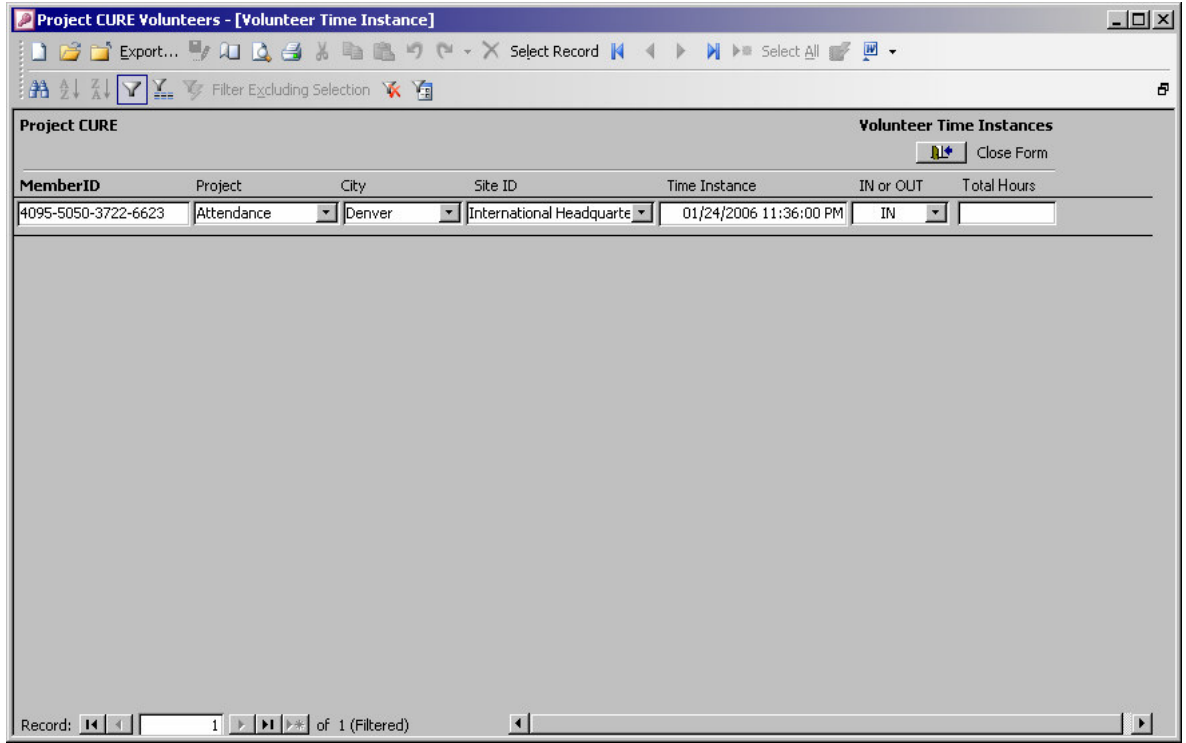
Step #4

- If you do not have a swipe card, type in your 16 digit Member ID number in the ID number field. As you type, you will see the dashes and number signs appear. You do not have to type in any dashes as it will automatically populate during input.

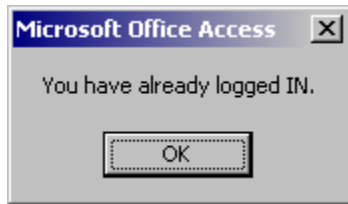
This screenshot shows the same application window as the previous one, but with the ID number field updated. The text input field now contains '4095-5050-####-####', where the dashes and number signs are automatically populated as the user types. The 'Press after swipe' button remains visible to the right of the field.

- When your number populates the ID number field, click on 'Press after swipe'.

- If the transaction successfully completes, a window will open and display your volunteer login/logout information. IF THIS WINDOW DOES NOT DISPLAY, please repeat Steps 1-3.



- If you have already logged 'IN' without logging out, you will receive this error:



Step #5

- When you complete swiping in, click 'Close Form' on all open windows.

The screenshot shows a software window titled "Project CURE Volunteers - [Card Swipe]". The window has a menu bar with options like "Export...", "Select Record", and "Select All". Below the menu bar is a toolbar with various icons. The main area of the window is divided into sections:

- Project CURE** section: Contains dropdown menus for "City Code" (Denver), "Site ID" (International Headquarters), and "Project" (Attendance). There is also a checkbox for "Check Records?".
- Close Form** button: A button with a red 'X' icon and the text "Close Form".
- Swiping Instructions** section: Contains a dropdown menu for "Please indicate if you are swiping IN or OUT:" (IN) and a text input field for "Swipe your card now... or enter your ID#:" (4095-5050-3722-6623).
- Press after swipe** button: A button with a clock icon and the text "Press after swipe".

The bottom half of the window is a large, empty gray area.

Step #1 – Swipe “OUT”

- Open the ‘Volunteer System’ database.
- In the Main Menu, select ‘Swipe Card’.

Step #2

- Select ‘City’ of the location where you volunteer work from the drop down menu.
- The ‘Site ID’ should default and populate. If it does not, select the Site where you volunteer work.
- The ‘Project’ should default automatically to ‘Attendance’.
- Select ‘**OUT**’ from the drop down menu. You cannot leave this field blank or you will receive an error notice.

Project CURE Volunteers - [Card Swipe]

File Edit View Insert Format Records Tools Window Help

Project CURE City Code: Check Records?

Site ID:

Project:

Close Form

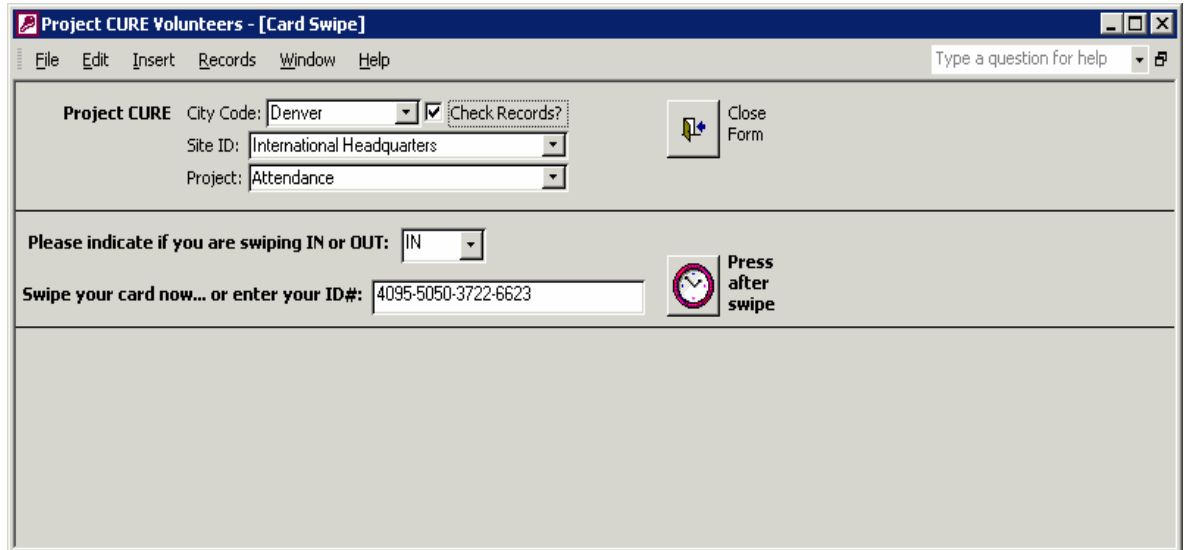
Please indicate if you are swiping IN or OUT:

Swipe your card now... or enter your ID#:

Press after swipe

Step #3

- Swipe your card through the swipe strip reader. **IF YOU DO NOT HAVE A SWIPE CARD, SKIP TO STEP #4.**
- When your number populates the ID number field, click on **'Press after swipe'**.



Project CURE City Code: Denver Check Records?
Site ID: International Headquarters
Project: Attendance

Please indicate if you are swiping IN or OUT: IN

Swipe your card now... or enter your ID#: 4095-5050-3722-6623

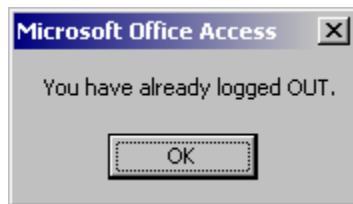
Close Form

Press after swipe

- If the transaction successfully completes, a window will open and display your work time total since you last logged in. This information is for the current work instance only. IF THIS WINDOW DOES NOT DISPLAY, please repeat Steps 1 through 3.



- If you have already logged 'OUT', you will receive this error:



- When you complete swiping out, click ‘Close Form’ on all open windows.

Project CURE Volunteers - [Volunteer Master Information]

Filter Data / Find By Project CURE

Click and populate field(s) below. Then click above on 'Apply Filter/Sort' Close Form

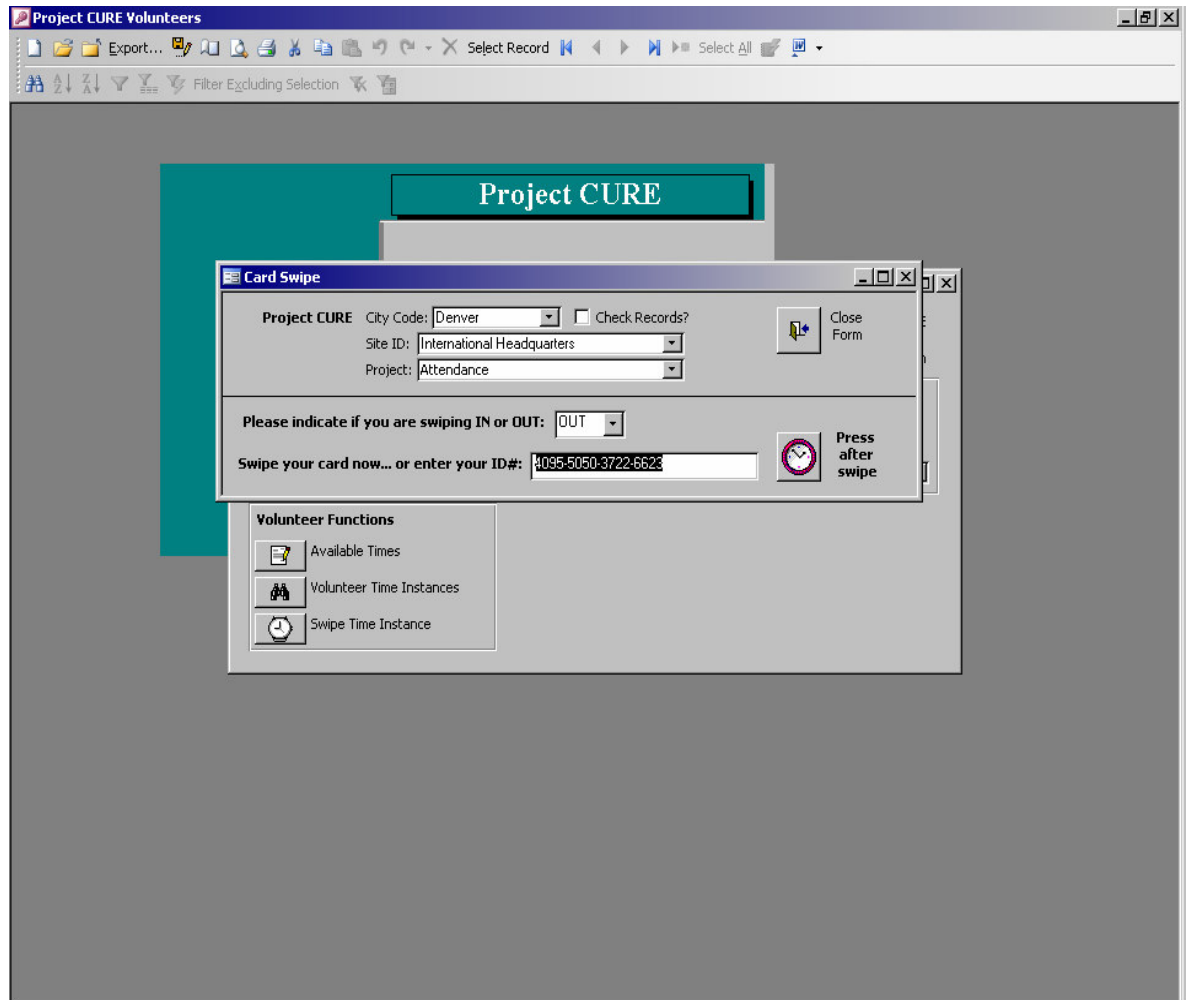
Member ID	Date Joined		
4095-5050-3722-6623	07/31/2003		
First Name	Middle Name	Last Name	Suffix
Desirea	Duarte	Ulibarri	

Volunteer Functions

- Available Times
- Volunteer Time Instances
- Swipe Time Instance

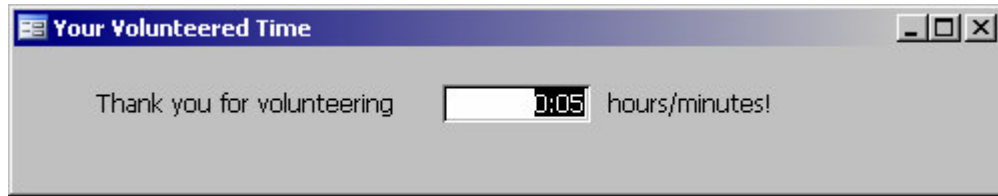
Step #4

- If you do not have a swipe card, type in your 16 digit ID number in the Member ID number field. As you type, you will see the dashes and number signs appear. You do not have to type in any dashes as it will automatically populate during input.

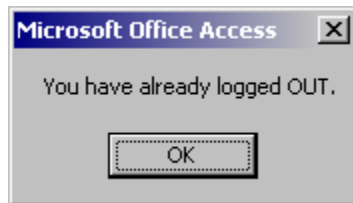


- When your number populates the ID number field, click on 'Press after swipe'.

- If the transaction successfully completes, a window will open and display your work time total since you last logged in. This information is for the current work instance only. IF THIS WINDOW DOES NOT DISPLAY, please repeat Steps 4.

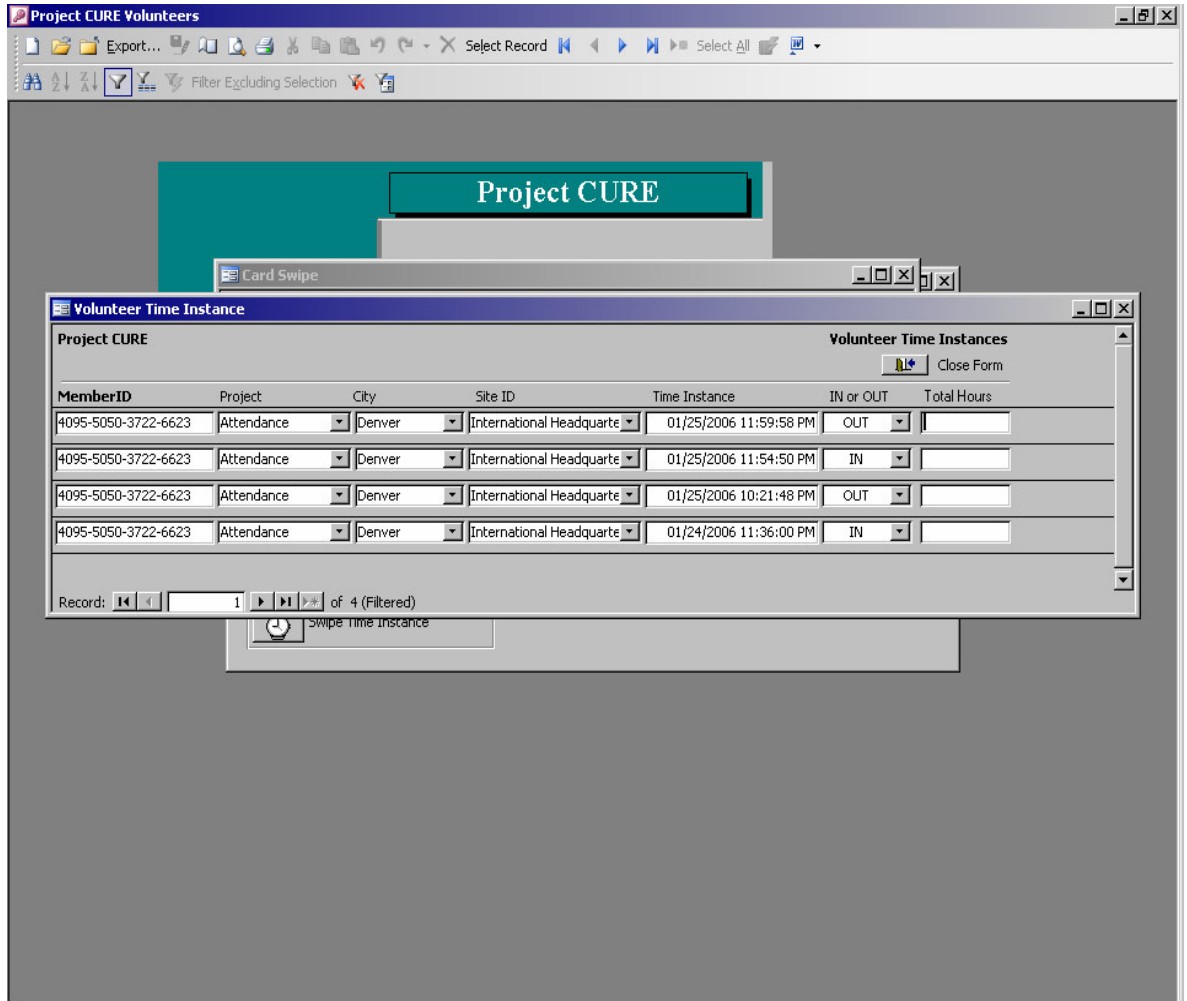


- If you have already logged 'OUT', you will receive this error:



Step #5

- When you complete swiping out, click ‘Close Form’ on all open windows.



Lesson 3: How to Create Reporting

Step #1

- Open the 'Volunteer System' database.
- In the Main Menu, select 'Create Reports'.
- Entered desired "Beginning Date Range" and "Ending Date Range" for the requested report.

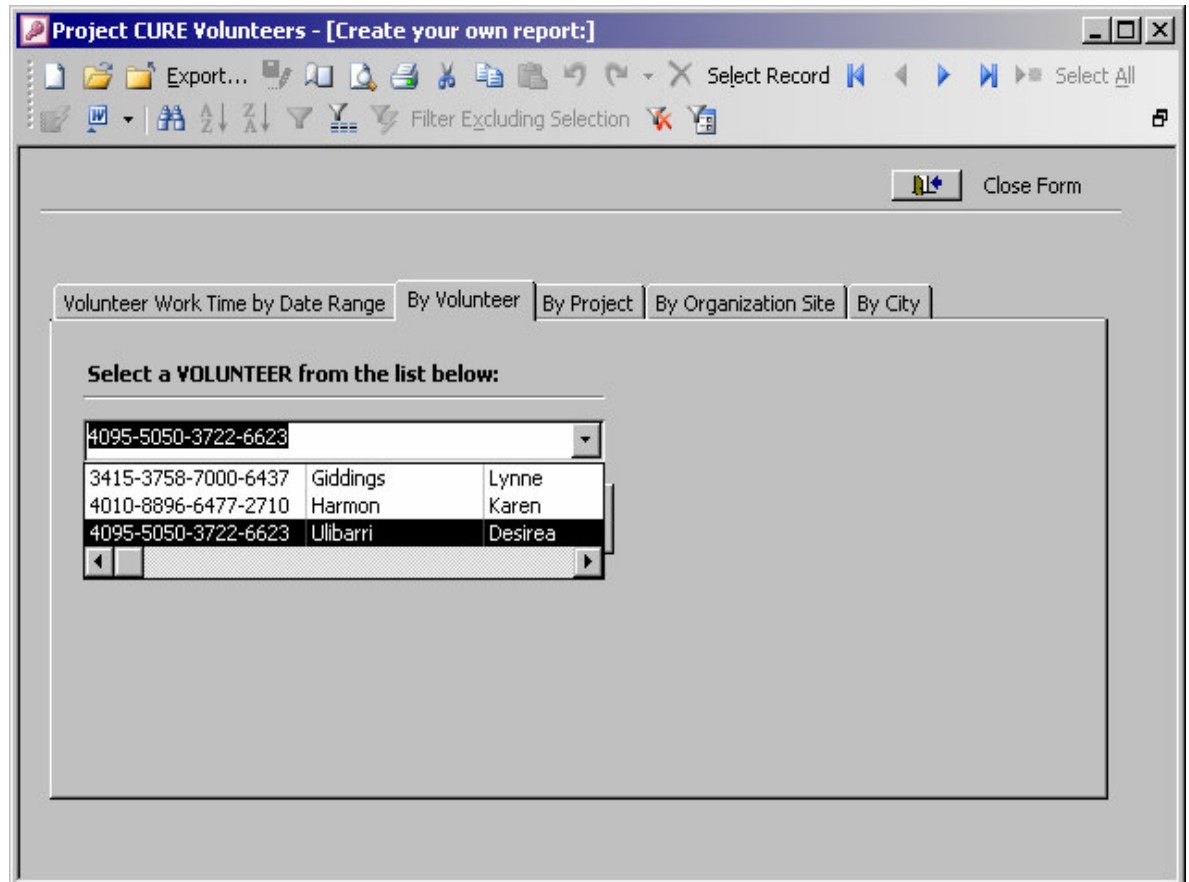
The screenshot shows a software window titled "Project CURE Volunteers - [Create your own report:]". The window has a menu bar with options like "Export...", "Select Record", and "Select All". Below the menu bar, there are several tabs: "Volunteer Work Time by Date Range", "By Volunteer", "By Project", "By Organization Site", and "By City". The "Volunteer Work Time by Date Range" tab is selected. The main content area is titled "Step 1) Enter Desired Date Range". It contains two text input fields: "Beginning Date Range:" with the value "01/01/2006" and "Ending Date Range:" with the value "03/10/2006". To the right of these fields is a blue-bordered box with text: "OR Step 3 - Select above tabs for other reporting options using this date range. For example, select 'By Volunteer Name' to review specific volunteer data only." Below the date range fields, there is a section titled "Step 2) Report for All Volunteers using this date range - Click Here:" followed by a small icon of a document with a magnifying glass, which is circled in red.

Step #2

- Click on the report icon to see all information for all volunteers for the specified date range. Otherwise skip to Step #3.

Step #3

- Select any page tab with the desired query parameter: 'By Volunteer', 'By Project', 'By Organization Site' or 'By City'.
- Once on the requested page, select available data from the drop-down menu(s).



- After choosing a query parameter, click the report icon.

- The report will open. It will report all available time tracking information but be filtered for the selected query parameter:

Volunteer Work Hours**01/01/2006 through 03/10/2006**

MemberID: 4095-5050-3722-6623

Desirea Duarte Ulibarri

Date: 02/03/2006

<i>Time Worked:</i>	<i>Site:</i>	<i>Project:</i>	<i>City:</i>
9:10	International Headquarters	Attendance	Denver
2:58	International Headquarters	Volunteer System Project	Denver
1:41	International Headquarters	Volunteer System Project	Denver
23:57	International Headquarters	Volunteer System Project	Denver

Date: 02/04/2006

<i>Time Worked:</i>	<i>Site:</i>	<i>Project:</i>	<i>City:</i>
3:58	International Headquarters	Volunteer System Project	Denver

Date: 02/08/2006

<i>Time Worked:</i>	<i>Site:</i>	<i>Project:</i>	<i>City:</i>
0:11	International Headquarters	Volunteer System Project	Denver

Total Time: 41:55 ***Hours: Minutes*****Total Report Time: 41:55**

Step #4

- Repeat Steps 1-3 for any other query parameter and date range.

Appendix G: NLP Journal Part 1

The Regis University Academic Research Network (ARN) is wide area network of several on-campus local area networks that are designed and designated to give students the opportunity to explore topics in systems engineering, database administration, etc. and provide hands-on experience in a lab setting. The ARN is constructed, maintained, and administered by tiered levels of IT support. The ARN needs IT support and maintenance to remain functional as a research network for the MSCIT program.

The relevance of the NLP program is to provide an avenue for students to fulfill research hours and benefit from a lab setting. In this manner I participated in the NLP to complete research and gain hands-on experience in networking. As a participant in NLP, I was considered a tier 1 support person as I had no prior working background in systems engineering. My only experience was what I learned in the graduate curriculum of the MSCIT program. My role in NLP was providing support backup to all other higher-tiered persons and to learn the basics of administering and maintaining the ARN. I researched and applied topics in web hosting and administration and am able to leave a document of my experience to benefit future NLP practitioners.

Journal

July 30, 2003: Started rebuilding a client workstation, increasing memory and installing a ghost image of Windows 2003 from the St. Anthony server.

August 4, 2003: Dan moved five additional machines to Arrupe High School. He will be out from Wednesday to Wednesday of this next week, and will get with me about working at Arrupe as well as DTC during August.

August 5, 2003: Finished building box. Dan Rcvd a call from Fr. Planning at Arrupe.

The IT Company that is providing IT support has extended their contract, indefinitely. So, there isn't an immediate need for us to become involved with the High School.

August 8, 2003: Box from NOC to CS on Recovery. Need to get with CS on progress on

ghosting. Get a copy of the NLP roster from Gregg for ELMS/MSDN administration. Mike Becker is a new person in FC. Paul suggests honing in on vendor-specific technical manuals. Cisco white papers are also a good reference. For papers, use everything you have done. EVERYTHING! Maybe I should take on a more management/administrator role in the lab. Research seems to be the most important aspect for projects. Bellarc is a free download. Maybe need to consider how to configure the boxes for the classes that are in there. Send Paul and Jim and Mike ELMS website. Clean up image inventories for ghosting. Need to document hard drives (Paper on the sides). Create IP address just for ghosting. Use for backup? Probably not, probably use just to speed up on imaging. Design, install, test and ghost an imaging process. Test pushing the ghost image back to boxes. Who is doing configuration management at DTC? Send Paul the activities and updates from our lab.

August 12, 2003: Ask Dan about printer 192.168.0.251. Ask about passwords for St.

Anthony, etc. Installed Belarc and PDF Writer. Looked at ELMS site.

August 19, 2003: Added Paul, Jim and Amy as ELMS users. Still need to add Mike

Becker. Need to find license for my computer or I will lose all. I just registered but I will need new Keys for Amy and Lynn's machine. Can't ghost because I do not have a

password for St. Anthony. I used the Admin password for Maxtor file server but I need to learn how to set up a ghost session.

September 1-2, 2003: Dan met with Amy Pepper and I to discuss new project concept

with Project CURE because Arrupe High School lab will be constructed by a contractor company. Amy and I will meet with representatives from Project CURE and obtain a set of high-level requirements for the database project.

September 13, 2003: Tore down DTC lab and replaced low-end Pentium boxes with

donated Pentium II and higher. Upgraded memory in all boxes and ghosted Windows 2000 with Office 2000 in all LAN computers.

September 16, 2003: Problem with Ghosting on LS120 drives. Bakir used Semantec

Ghost wizard to create a bootable floppy in MS-DOS mode with LS120 drivers--

Sematec 7.5. Wizard asks for source drive for files so then use Windows98. Again

we are multicasting multiple ghost images at the same time. Initiate Ghostcaster

server program (St Luke) and name session, find image and accept clients. Use

Semantic floppy to boot program on clients and then initiate ghosting and connect

to server. Hit send once all ghost clients are connected and visible. What is going on

with the Web Server. What is the status of the web server-- For new software--add

to storefront on ELMS. Import new group with Gregg's list.

September 17, 2003: Received business requirements from Project CURE.

October 11, 2003: Project Status: Enterprise Architecture

Centralized documentation: Dan's website: www.dlikarish.com

Mission: Drive people to Dan's website for NLP applications. Service others.

Microsoft Operational Framework. MSF (drives architecture). ITIL, 7799, 8500.

ITIL (IT Infrastructure Library). <http://www.ogc.gov.uk/index.asp?id=2261>.

<http://www.bsi-global.com/indez.xalter>. <http://www.itsmf.com>.

SMS (MS) is a separate server.

Moving from LAN based org to an Enterprise org for ARN.

Dan proposes to adopt MOF, a mgmt standard that combines Microsoft Service.

MSF is a modified waterfall with spiral (working prototype troubleshooting)

iteration. Dave Pultorak, Whitepaper, Microsoft.

THESE ARE NOT OPEN SOURCE STANDARDS like Linux or Unix.

Discuss/research XML to exchange database information over SOAP.

www.ARN-Regis.org. Internal 192.168.1.178. MS Applications.

PortalWorkspaces interfacing with Team Services driving profile with Link to File Store.

SQL Or Security 2000 Or Citrix Etc

November 11, 2003: I need to get new copy of database from Project CURE and Revise report by date range input to see hours by date and time. There is no date associated for when the schedule is valid so I need to look into end-date possibility. Maybe it will have to be a manually applied entry. Need a pop-up for swipe card reader and perhaps need new field "Inactive Date". "Comments" field, activity description, and activity comment do not appear in Volunteer database.

November 25, 2003: According to Bob's new requirements, he wants to query based on historical availability.

September 28, 2005: Converting the query logic into VBL is proving to be very tough. I built the queries to step out each desired effect. TimeCalculationTableQuery points

to TimeCalculationQuery2 which in turn relies on TimeCalculationQuery. It is difficult to pinpoint where to break in the code as queries essentially run backwards. However, because TimeCalculationQuery relies on TimeInstanceCalculationQuery, I feel it would be best to modify TimeInstanceCalculationQuery. I am trying to relearn how to build a query that has filter criteria from an input form. Thus, I can then direct this query to filter by Volunteer ID and data in order to populate TimeCalculationQuery which pivots the IN and OUT values into their own table columns. I am pretty sure that I can accomplish all the queries in one SQL statements, however at this time I will keep the queries intact as it is work already accomplished.

Appendix H: NLP Journal Part 2

The MSDN Academic Alliance ELMs software program enables colleges and universities to allow users to download and check out software from a large selection of approved titles for a lease period. The available software programs may be beta versions, new releases, or even commercially available full-featured programs. The lease period was a pre-defined period determined by the participating university department. Regis had a limit of 50 licenses per software program and a set lease period of six months. The software was only available to students in participating classes and faculty teaching those classes.

There were three ways for students and faculty to obtain software. All titles for each software disk were available for check-out, some software could be downloaded online, and permanently licensed copies of popular titles could be purchased online. Actual hard copies of available software were stored on CD and catalogued at the Regis DTC campus. A student with an ELMs account could access the MSDN Academic Alliance storefront website at https://msdn03.e-academy.com/elms/Security/Login.aspx?campus=regis_grad and select a title from a roster of available software. The website would send the user a product serial/activation key and instruct the student to email the ELMs administrator for the software. The administrator would receive the email and burn a hard copy of selection for the student to pick up or receive by mail. The product serial/activation key was leased, distributed, and tracked by the MSDN Academic Alliance. The ELMs administrator had only a cache of product activation keys for internal lab distribution and use. Below is a sample of the ELMs administrator response for CD check-out:

Fred,

I have received your request for Visual Studio.NET.

I will burn you a copy tonight and leave it at the front desk of DTC.

Please see the receptionist.

In the meantime, please go to https://msdn03.e-academy.com/regis_grad/

Login ID: Fred Red

Password: New2003

Once you are logged in, go to software and select Visual Studio .NET Professional 2003 - CD1 and separately you must also select Visual Studio.NET Professional 2003 - CD2. Once you submit your request, MSDN will send the product key to install the software via email.

If you have any questions, please feel free to email me.

Thanks,

Desirea

The Regis ELMs administrator worked with another practicum student to set up a server at the Regis DTC campus to host titles of ELMs software. The software storefront was hosted by the MSDN Academic Alliance and accessible via the Internet, but select titles of the actual software were loaded on an internal server in the Regis DTC networking practicum lab. These software titles were enabled as a download from the ELMs website. When the download option was enabled and selected for a title, a license key executable would be downloaded from a server at MSDN Academic Alliance. The user would be prompted to save the executable on their local computer. A product serial/activation number would also be provided in a notification response from MSDN Academic Alliance after successful authentication of the user. When the user clicked on the license key executable, the file would route a download request to the URL of the DTC server and the software would come directly from the Regis DTC server, not from

the ELMs site. The executable license key program would install the software only when the correct product serial/activation number was applied. This security mechanism ensured that unauthorized persons could not use any software without a product activation key. If any persons were to compromise the DTC server, at which at time had little or no hardening on any files or directories, the software would be unusable without this key. After the software was successfully authorized, downloaded, and authenticated, it could be unpacked, installed, and launched.

The file extension .sdc is the program that stores the software in compact and locked form. The .sdc format is a wrapper and protects the software from use until a product activation key is used to initiate unpacking and installing. All software files stored on the Regis DTC hosting server were of file extension .sdc and would not work without a product activation key. The files were secure from use in the event of theft as they were unusable in the .sdc format.

Each piece of software that was enabled for download was configured on the ELMs storefront to point to the exact directory and folder on the hosting server. The URL configured on the ELMs storefront was <http://elms.arn-regis.org/files/<softwarename>.sdc>. The exact URL had to be added for each individual piece of software. The management of the ELMs storefront could become time consuming when server directories were moved or folder names were changed. As a result, only the most requested and popular software programs were enabled and configured for download. Microsoft Office, Project 2002 and 2003, Visio 2002 and 2003, Windows XP Professional and Visual Studio.net 2003 were among the chosen software. Below is a sample of the MSDN Academic Alliance notification for software download:

***** Regis
University - GRAD

ELMS for MSDNAA Software Center: This message has been generated automatically from your MSDNAA Online Software System. Please do not reply to this message as you will not receive a response. Please click on the SUPPORT link in your MSDNAA Online Software System for support contact information on your campus.

Regis University - GRAD

Below is a summary of your order from your ELMS for MSDNAA Software Centre and any product installation keys required to install your software. Please keep this mail for future reference.

Customer: Desirea Ulibarri
Username: Desirea.Ulibarri@Nextel.com
Storefront URL: http://msdn03.e-academy.com/regis_grad

Date of Order: 2004-03-28
Invoice Number: E0164015

L/N: 0000-164015

Product: Project Professional 2003
1 Download @ \$0.00 USD = \$0.00 USD

Method of Payment: No Charge

Serial Number
=====

To view serial numbers and any special instructions for the product(s) you purchased, please use the link below and log into the system. Under "My Software" please select the product, you will see serial number and any special instructions corresponding to your purchase.

To view or print an official copy of your order receipt, simply login here:
http://msdn03.e-academy.com/regis_grad
=====

Sincerely,
ELMS for MSDNAA Staff
e-academy Inc.

Also available were discount software programs for purchase. When selected, the student would receive packaged software in the mail. The ELMs storefront handled the sales transaction and the software would come directly from the MSDN Academic Alliance. Regis held no responsibility for purchased software; the ELMs administrator kept no inventory or record of software purchases. The licenses for purchased software did not interfere with the pool of licenses available for free download/checkout.

Administration of the ELMs consisted of keeping up-to-date records of the electronic and physical software inventories, uploading and managing user names and passwords in ELMs website, and configuring the internal web server for hosting software. Electronic software titles were tracked and enabled for check-out and/or download on the administrator area of the ELMs website. In addition user accounts and passwords were also maintained using the ELMs website. All physical software and internal product activation keys were kept in locked cabinets at DTC practicum lab and only the lab leads had access to the disks. When a request would come in for software, the administrator would burn a copy from the master disks and assign an inventory number in order to track how many physical copies were being used at any one time.

In order to host software and support downloading, the router firewall in the DTC practicum lab was configured to allow incoming requests on port 80 to pull program files from a particular directory on the web server. The only legitimate requests on port 80 were those for ELMs software download. At the time, the ELMs web server was the only hosting server in the DTC practicum lab. Thus, only one public IP address was needed and used. All SDC files were stored in directories at http://elms.arn-regis.org/files/*sdc, where * is the name of the software file.