2002-09

Web based database processing for Turkish Navy Officers in USA

Ozkan, Gokhan.

Monterey, Calif. Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

WEB BASED DATABASE PROCESSING FOR TURKISH NAVY OFFICERS IN USA

by

Gokhan Ozkan

September 2002

Thesis Advisor:	William J. Haga
Second Reader:	Gary Porter

Approved for public release; distribution unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2002	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE: Web Based Database Processing for Turkish Navy Offices in USA			**5. FUNDING NUMBERS**
6. AUTHOR(S) **Gokhan OZKAN**			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A			**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution unlimited.			**12b. DISTRIBUTION CODE**

13. ABSTRACT ***(maximum 200 words)***

This thesis reports the client server architecture choices (2- tier versus 3 tier) and details the supporting web server and database server choices. It then presents a prototype of a web-based database system to speed and simplify tracking of academic and personal information on naval officers attending graduate school in another country. The result will be better manpower data for naval headquarter with fewer errors at the lower cost. The difficulties implementing client server systems are discussed along with suggestions for better management of information technology.

14. SUBJECT TERMS Information technology, Database, Web-database Connectivity, Three tier architecture, Two tier architecture, UML			**15. NUMBER OF PAGES** 83
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution unlimited.

WEB BASED DATABASE PROCESSING FOR TURKISH NAVY OFFICERS IN USA

Gokhan Ozkan
Lieutenant Junior Grade, Turkish Navy
B.S., Turkish Naval Academy, 1995

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN INFORMATION TECHNOLOGY MANAGEMENT

from the

NAVAL POSTGRADUATE SCHOOL
September 2002

Author: Gokhan Ozkan

Approved by: William J. Haga
Thesis Advisor

Gary Porter
Associate Advisor

Dan Boger, Chairman
Information Systems Academic Group

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This thesis reports the client server architecture choices (2- tier versus 3 tier) and details the supporting web server and database server choices. It then presents a prototype of a web-based database system to speed and simplify tracking of academic and personal information on naval officers attending graduate school in another country. The result will be better manpower data for naval headquarter with fewer errors at the lower cost. The difficulties implementing client server systems are discussed along with suggestions for better management of information technology.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

The author would like to acknowledge Prof. Haga and Prof. Porter for their continuous support throughout all the phases of the thesis.

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

This thesis reports the client server architecture choices (2- tier versus 3 tier) and details the supporting web server and database server choices. It then presents a prototype of a web-based database system to speed and simplify tracking of academic and personal information on naval officers attending graduate school in another country. The result will be better manpower data for naval headquarter with fewer errors at the lower cost. The difficulties implementing client server systems are discussed along with suggestions for better management of information technology.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. BACKGROUND

1. Problem

With the help of a reliable information system, Turkish Navy Headquarters in Ankara, Turkey can have real time access to the administrative and educational information of the Turkish Navy Officers who are studying for master degrees in USA.

Using the information system proposed in this thesis, Turkish Navy officers in USA will be able to do simple, repetitive tasks through the web by themselves. The system will provide a central location for officers to update their required personal data that would otherwise require tedious time, energy, man power and filing for both the Turkish Military attaché in Washington D.C. and The Turkish Headquarter in Ankara, Turkey. (HQ)

2. Solution Proposed by This Thesis

All the officers must prepare paper-based educational, administrative and personal reports according to DKY-179-1 regulation. The most senior officer in that school files the reports, and is responsible for sending them to the HQ through the attaché.

A solution is a web-based database for Turkish Navy Officers, which will provide access, read, update and delete capability for all information on a need-to-know basis. All the users, such as the liaison officer in Turkish Navy Headquarter in Ankara, the attaché and the officers will have access to the system, based on their privileges.

3. Consequence If The Problem is Not Solved

- The right answer delivered too late becomes the wrong answer.
- The Officers, attaché and liaison officer who are geographically dispersed from each other will not have real time access to the information.
- In order to make a simple change, many interactions are necessary.
- In order to manage simple repetitive tasks, too much manpower, time and effort is wasted.

4. Specific Case

According to DKY-179-1 regulation, officers are responsible to inform HQ about their current address, and phone number. Most officers are move frequently and change phone numbers frequently. To handle these changes on time requires a lot of paperwork, time and effort.

Officers must prepare and submit their grade point average to Headquarters via the attaché at the end of each quarter. This procedure, too, takes a lot of time and effort.

Officers have opportunities to go to other US states or other countries during their vacations. Within a reasonable amount of time all of their phone number information of their vacation destinations. And then, they are required to get permission for the trip from the HQ via the attaché. These vacation plans are used to reach the officers in an emergency. However, providing accurate addresses and phone numbers within a reasonable amount of time is difficult.

The system will be implemented as a web based system. The users will be able to access the system through their browser anywhere and anytime and perform most of their own administrative and educational responsibilities. With the proposed system, most of the current difficulties will be eliminated. Real time information will be provided to the users. This will save a considerable amount of time and effort.

B. OBJECTIVES

The goal of this thesis is to design, implement a web based administrative and educational database system for the Officers. In keeping with this goal, the current architecture types, development tools, database technologies will be discussed and the best configuration will be selected. Additionally, according to the selection, a prototype will be designed and implemented.

C. METHODOLOGY

The thesis research will cover the following phases:

- Planning and design: Requirement of the application will be defined
- Web based database architectures
- Web based database application development tools

- Web server database server connection programs
- Database management systems
- Application design and implementation

THIS PAGE INTENTIONALLY LEFT BLANK

II. PLANNING AND DESIGN

There is an old saying: “Don’t try to fix it unless you understand it.” With those words of wisdom, the author first details the project’s mission, vision, values and goals in the planning process; later, defines the requirements of the system by using use cases as the first step in the design of the application. Rational Rose Unified Modeling Language (UML) (For more information about UML see Appendix A) use case views are used to visualize the functionality the system should deliver, as perceived by external actors.

A. HIGH LEVEL PERSPECTIVE

1. Plan

The correct and timely management of information is a critical success factor on administrative applications. The increase in the use of the Internet as a mean of communication (via electronic mail and bulletin boards) will continue to expand and to influence complex military administrative applications. Services and applications provided by the Internet will the users in real time. Reducing the paperwork, and delay while enhancing service and convenience.

a. Mission

Based on the defined requirements in the DKY-179-1 research, analyze, design, implement and document a secure, standard, centralized, time saving, modern, integrated web based administrative and educational database system. This will be easy to access and use at anytime and at anywhere for all the users.

b. Vision

The vision of this system is to provide a convenient worldwide communication environment to perform the users responsibilities without delay and paperwork.

c. Value

The application will conform the highest standards of honesty, integrity, accuracy and professionalism.

d. Goals

The goals of this application can be listed as:

- Adding value by enhancing quality.
- Providing timely information, delivered to right person at the right time.
- Simplifying and automating processes, tasks and transaction to reduce cost and delay.
- Integrating of all the users over large geographical areas.

B. REQUIREMENTS

The basic requirements in the project planning and design process are broken down by reference number, function and category type.

Reference #	Function	Category
R.1	Intrusion and authentication	
R.1.1	Allow Officers to *Input User Name.*	Evident
R.1.2	Allow officers to Input Password	Evident
R.2	Add information	
R.2.1	Add personal information	Evident
R.2.2	Add grade information	Evident
R.3	Update information	
R.3.1	Update personal information	Evident
R.3.2	Update grade information	Evident
R.4	Display Information	
R.4.1	Display personal information	Evident
R.4.2	Display grade information	Evident

Reference #	Function	Category
R.5	Maintenance	
R.5.1	Delete the account of graduated officers	Evident
R.5.2	Set officers privileges	Evident

C. THE USE CASE DIAGRAM

The main Use Case Diagram is presented below in Figure 1. The chart summarized the processes that the application performs. The actors will: authenticate themselves to the system according to their privileges the application will let them add, update, and/or display information.

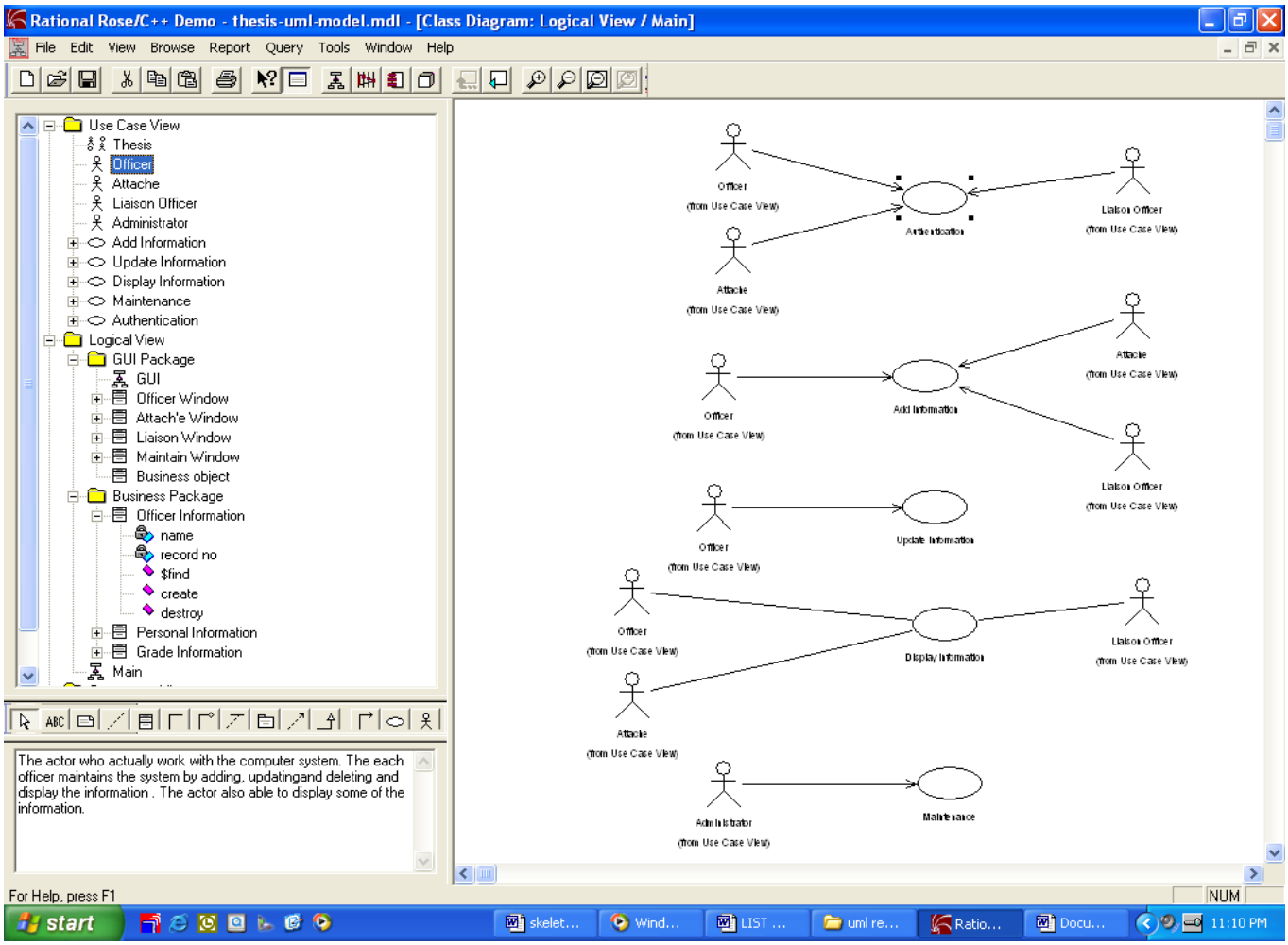


Figure 1. Use Case Diagram

D. THE USE CASES

The use cases identified in the previous diagram are expanded blow.

USE CASE	Authentication
TYPE	Primary, Essential
ACTOR	Officer, Attaché, Liaison Officer, Administrator
PURPOSE	To introduce and authenticate the actor
OVERVIEW	The authentication begins when the user want to access to the application through the web browser by using specific IP address.
CROSS REFERENCE	Functions: R.1.1, R1.2

TYPICAL COURSE OF EVENTS	
ACTOR ACTION	**SYSTEM RESPONSE**
1. The user reads warning page and hits the **login** button	**2.** Intrusion and authentication message box pops up
3. The user inputs his/her user ID and password in the message box and the OK button is pressed	**4.** System validates the provided password and user ID
	5. System assigns the user privileges according to pre assigned NTFS permissions.
	6. User specific web page pops up
Alternative Courses:	
Line 4. A dialog box prompting "You Are Not Authorized To See This Page" pops up.	

USE CASE	Add Information
TYPE	Primary
ACTOR	Officer, Attaché, Liaison Officer
PURPOSE	To add personal or grade information of the actor
OVERVIEW	Enable the user to add own personal and grade information in the database.
CROSS REFERENCE	Functions: R.2.1, R2.2

TYPICAL COURSE OF EVENTS	
ACTOR ACTION	**SYSTEM RESPONSE**
	1. This use case begins once the system has authenticated the user
2. The user selects the ***add option***	**3.** ***Add*** web site comes to the screen
4. The user will add his/her personal or grade information into the proper places on the web site and press submit button.	5. The system validates the data type accuracy of the provided data at the browser side.
	6. Send the data to the database to the application.
7. Confirmation page pops up.	
Alternative Courses:	
Line 5. A dialog box prompting, "The property type for the provided information is not correct, please correct it and submit again."	

USE CASE	Update Information
TYPE	Primary, Essential
ACTOR	Officer
PURPOSE	To update personal or grade information of the actor
OVERVIEW	Enable the user to update own personal and grade information in the database.
CROSS REFERENCE	**Functions:** R.3.1, R3.2

TYPICAL COURSE OF EVENTS	
ACTOR ACTION	**SYSTEM RESPONSE**
	1. This use case begins once the system has authenticated the user
2. The user selects the ***update option***	**3.** ***Update*** web site comes to the screen
4. The user will update his/her personal or grade information into the proper places on the web site and press submit button.	**5.** The system validates the data type accuracy of the provided data at the browser side.
	6. Send the data to the database to the application.
7. Confirmation page pops up.	
Alternative Courses:	
Line 5 A dialog box prompting, "The property type for the provided information is not correct, please correct it and submit again."	

USE CASE	Display Information
TYPE	Primary, Essential
ACTOR	Officer, Attaché, Liaison Officer
PURPOSE	To display personal or grade information of the actor(s)
OVERVIEW	Enable the user to retrieve the information from the database and display.
CROSS REFERENCE	**Functions:** R.4.1, R4.2

TYPICAL COURSE OF EVENTS	
ACTOR ACTION	**SYSTEM RESPONSE**
	1. This use case begins once the system has authenticated the user
2. The user selects the ***display option***	**3.** ***Display*** web site comes to the screen and shows the information according to the users privileges.

USE CASE	Maintenance
TYPE	Secondary
ACTOR	Administrator
PURPOSE	To add new accounts and set their privileges or removes graduated officer's accounts from the system.
OVERVIEW	Maintain the system.
CROSS REFERENCE	**Functions:** R.5.1, R5.2

TYPICAL COURSE OF EVENTS	
ACTOR ACTION	**SYSTEM RESPONSE**
	1. This use case begins once the system has authenticated the user
2. The user selects the either of the two: • ***Add new account option*** • ***Remove account option***	
<u>If the user selects the Add new account option:</u>	
	3. ***Add new account*** web site comes to the screen asking for the following: • **The name of the new user,** • **The last name of the new user,** • **Temporary password of the user**

ACTOR ACTION	SYSTEM RESPONSE
4. The user will selects the name, last name text boxes and input the new users name, last name and temporary password and press submit button.	**5.** The password is matched with existing password list. If it exists, the user is prompted with the line 3 options after displaying an error message "Change The Password"
<u>If the user selects remove account option:</u>	
	3. *Remove account* web site comes to the screen asking for the following: • The name of the user you want to remove • The last name of the user you want to remove
4. The user inputs the name and/or last name of the user and press the submit button.	**5.** Find the account in the database and display account information asking confirmation to remove the account.
6. The user press the confirms button	**7.** Account is removed from the system.

THIS PAGE INTENTIONALLY LEFT BLANK

III. WEB BASED DATABASE ARCHITECTURES

In this chapter client server computing with it's components, and two tiered and three tiered web database architectures will be covered.

A. CLIENT SERVER COMPUTING

Internet provides communication services to computers by using standard Terminal Control Program/ Internet Protocol (TCP/IP) communication protocol. This communication service can be divided into two logical units: a client and a server. In order to initiate communication, client unit (web browser refers to the client) makes request. The server unit then provides a response to the client's needs. This communication is called client/server computing.

The forms of Client/Server systems in use today are 1 tiered, 2 tiered, 3 tiered and N tiered architectures. All of these Client/Server systems have these properties (Orfali, 1999):

- ***Service:*** Client/server is primarily a relationship between processes running on separate machines. The server process is a provider of services. The client is a consumer of services. In essence, client/server provides a clean separation of function based on the idea of service.
- ***Shared resources:*** *A* server can service many clients at the same time and regulate their access to shared resources.
- ***Asymmetrical protocols:*** There is a many-to-one relationship between clients and server. Clients always initiate the dialog by requesting a service. Servers are passively awaiting requests from the clients. Note that in some cases a client may pass a reference to a *callback* object when it invokes a service. This lets the server call back the client. So the client becomes a server.
- ***Transparency of location:*** The server is a process that can reside on the same machine as the client or on a different machine across a network. Client/server software usually masks the location of the server from the clients by redirecting the service calls when needed. A program can be a client, a server, or both.
- ***Mix-and-match:*** The ideal client/server software is independent of hardware or operating system software platforms. You should be able to mix-and-match client and server platforms.

- ***Message-based exchanges:*** Clients and servers are loosely coupled systems that interact through a message-passing mechanism. The message is the delivery mechanism for the service requests and replies.
- ***Encapsulation of services:*** The server is a "specialist." A message tells a server what service is requested, it is up to the server to determine how to get the job done. Servers can be upgraded without affecting the clients as long as the published message interface is not changed.
- ***Scalability:*** Client/server systems can be scaled horizontally or vertically. Horizontal scaling means adding or removing client workstations with only a slight performance impact. Vertical scaling means either migrating to a larger and faster server machine or distributing the processing load across multiple servers.
- ***Integrity:*** The server code and server data is centrally managed, which results in cheaper maintenance and the guarding of shared data integrity. At the same time, the clients remain personal and independent.

The client/server characteristics described here allow intelligence to be easily distributed across a network.

B TWO TIERED WEB DATABASE ARCHITECTURE

In 2-tiered web database architecture, a client talks directly to the database server (Figure 2). Simplicity is the biggest factor driving the popularity of 2-tier client/server. An advantage of 2-tier is the ability to create applications quickly using visual builder tools. Typically, these are departmental applications, such as decision support and small-scale groupware, or simple Web publishing applications. (Orfali, 1999)

Many system solution models made by 2-tier architectures worked very well, however, in the large-scale real world implementations most of them failed. In 2-tiered database architectures client has to manage:

- User interface
- Data validation
- Post requests from clients
- Execute database retrievals and updates
- Manage data integrity
- Control transactions.

Most of these functions require some programming capabilities to handle.

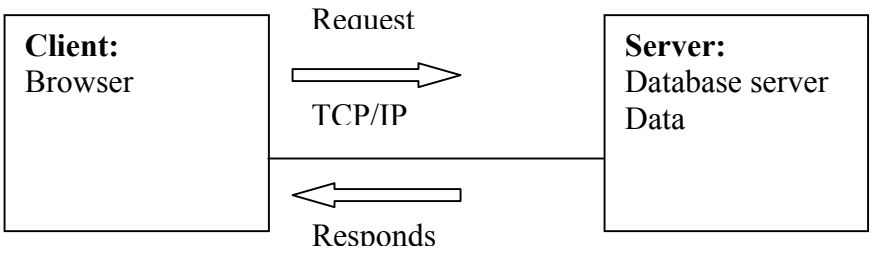


Figure 2. Two-Tier Architecture

C. THREE TIER WEB DATABASE ARCHITECTURE

In the 3-tiered database applications, the web server tier is inserted between the web browser and the database server. (Figure 3)

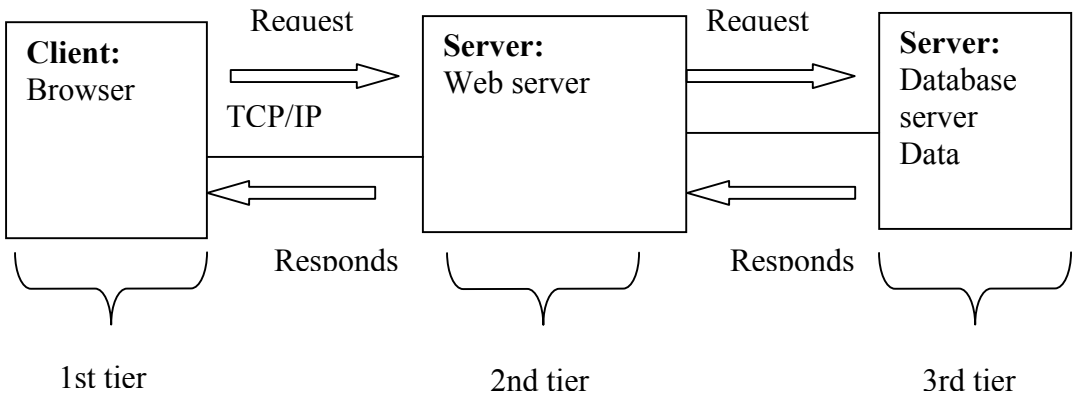


Figure 3. Three Tier web database architecture

Data flow in the 3-tier architecture is provided in the Figure 4.

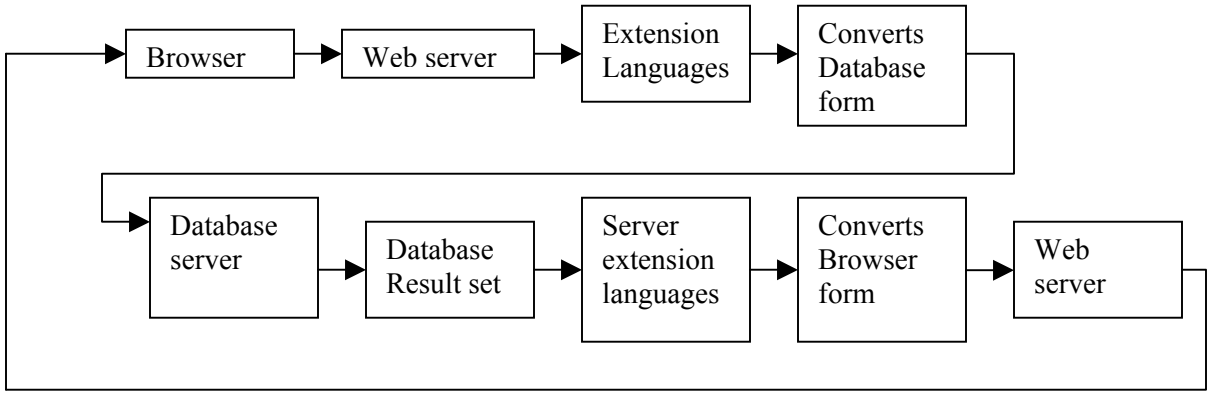


Figure 4. Data flow in the three-tier architecture

A 3-tier architecture extends the 2-tier architecture to allow additional processing to occur before the Web server responds to the web client's request. Higher-order architectures, that is, those that have more than three tiers, are usually called n-tier architectures. The third tier usually includes software applications that supply information to the Web server. The Web server can then use the output of these software

applications when responding to client requests instead of just looking up a Web page or other type of file on its disk drive. Architectures that have four, five, or even more tiers include software applications (just as the three-tier systems), but they also include the databases and database management programs that work with the software applications to generate information that the Web server can turn into Web pages, which it then sends to the requesting client. (Schneider, 2002)

The additional tier provides more functionality and helps to meet the requirements of large-scale applications. The functionalities that are provided by this additional tier can be summarized as follows:

- Deal with the database server and it's data schema. Database server deals only with the web server.
- Components and different data formats can be reused for many applications in the database server
- Provides access to different types of database management systems (DBMS) without client cooperation
- Adds flexibility to database server to add or upgrade different DBMSs with minor adjustments only in the web server
- Database server's processing speed only depends upon web server's processing speed.
- Embedded authorizations on the web server provide better security.

Table 1 shows a detailed comparison of two tier and three tier applications. (Orfali, 1999)

COMPARISON OF TWO-TIER AND THREE-TIER ARCHITECTURE			
No	**Property**	**Two-tier**	**Three-tier**
1	System administration	Complex (more logic on the client to manage)	Less complex (the application can be centrally managed on the server)
2	Security	Low (data-level security)	High (fine tuned at the service or method level)
3	Encapsulation of data	Low (data tables are exposed)	High (the client invokes services or methods)
4	Performance	Poor (many SQL statements are sent over the network; selected data must be downloaded for analysis on the client)	Good (only service requests and responses are sent between the client and server)
5	Scale	Poor (limited management of client communications links)	Excellent (concentrates on incoming sessions; can distribute loads across multiple servers)
6	Application reuse	Poor (monolithic application on client)	Excellent (can reuse services and objects)
7	Ease of development	High	Getting better
8	Server to server infrastructure	No	Yes (via server side middleware)
9	Legacy application integration	No	Yes (via gateways encapsulated by services or objects)
10	Internet support	Poor (internet bandwidth limitations make it harder to download fat clients and exacerbate the already noted limitations)	Excellent (thin clients are easier to download as applets or beans even as HTML pages; remote service invocations distribute the application load to the server)
11	Heterogeneous database support	No	Yes (three-tier applications can use multiple databases within the same business transactions)
12	Rich communication choices	No (only synchronous, connection-oriented calls) Yes	(supports connection oriented calls, connectionless messaging, queued delivery, publish-and-subscribe, and broadcast)
13	Hardware architecture flexibility	Limited (one has a client and a server)	Excellent (all three tiers may reside on different computers, or the second and third tiers may both reside on the same computer; with component-based environments, you can distribute the second tier across multiple servers as well)
14	Availability	Poor (cannot fail over a backup server)	Excellent (can restart the middle tier components on other servers)

Table 1 The Comparison of Two-Tier and Three-Tier Architectures (From Orfali, 1999)

D. CONCLUSION

Information systems are trying to make our life easy and provide solutions for today and possible future problems. These are the driving factors for the technology. Each single item in the information system should be open to the benefits of the provided technological developments. The suggested information system in this thesis will benefit only the officers in USA for administrative and personal record keeping areas. However, in the future, the variety of users and the problem solution areas can be enlarged. In order to provide a scaleable framework, 3-tier architecture is selected for this implementation.

IV. WEB BASED DATABASE APPLICATION DEVELOPMENT TOOLS

A. SERVER SIDE TECHNOLOGIES

The Web server performs three major functions. First, it is an HTTP server meaning that it processes the HTTP protocol, receiving requests and generating responses in HTTP format. The second function of the Web server is to host scripting applications to help developers write code in languages such as VBScript and JavaScript to be used on the web server. The third and final function in the database application of the Web server is to create, read, update, and delete view instances.

In the network environment, by the help of server's functions, the network database applications can be published in three different ways according to the client request.

- *Static Report Publishing*: In this type of publishing, a client sends a request to a server and server sends a static form, or query response in HTML format, which is generated by the database application. (Figure 5) Notice that the transmission traffic in static report publishing is one-way. The user provides no data; all of the data flows from the server to the user at the user's request. Because of this, no data about the state of the interaction between the client and the network server need be maintained by either. (Kroenke, 2000)

Figure 5. Static Report Publishing

- *Database Query Publishing:* In this type of publishing client sends a HTML request form that has text boxes. The client defines the query criteria in this text boxes. (Figure 6) Unlike static report publishing, the data transmission for query publishing is two-way. The client and the server interaction need to be maintained.

Figure 6. Database Query Publishing

- *Application Publishing:* In this kind of publishing, applications provide many data entry forms and reports to the client. (Figure 7) They support transaction logic, provide concurrency control over database changes, and have all of the other database application characteristics. (Kroenke, 2000)

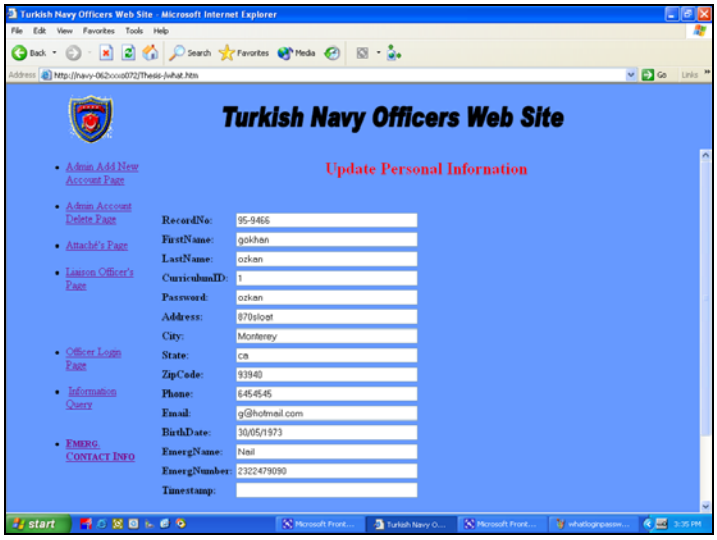


Figure 7. Application Publishing

The popular server-side technologies in 2002 were Common Gateway Interface (CGI), Web Server Application Programming Interfaces (API), Active Server Pages (ASP), Java Servlets and Java Server Pages (JSP). (Topuz, 2002)

The focus of this chapter and the implementation of our application will be Common Gateway Interface (CGI), Web Server Application Programming Interfaces (API), and Active Server Pages (ASP).

1. Common Gateway Interface (CGI)

The Common Gateway Interface (CGI) is a standard for interfacing external applications with information servers, such as HTTP or Web servers. A plain HTML document that the Web user retrieves is static, which means it exists in a constant state: a text file that doesn't change. In real time a CGI program is executed in real-time, so that it can output dynamic information.

Because of its superior string processing capabilities, the language Perl is often used with CGI. (Kroenke, 2000)

The greatest disadvantage of CGI is that web server creates an entirely new process for each client request with its own variables. This reduces the process and response speed of the web server.

2. Web Server APIs (ISAPI, NSAPI)

The disadvantage of CGI's creating an entirely new process for each client request with its own variables is eliminated in Microsoft's Internet Server Application Program Interface (ISAPI), and Netscape's Server Netscape Application Program Interface NSAPI.

API server extensions provide an alternative to the use of CGI applications for Internet servers. Unlike CGI applications, APIs run in the same address space as the HTTP server and have access to all the resources available to the HTTP server. APIs have lower overhead than CGI applications because they do not require the creation of additional processes and do not perform time-consuming communications across process boundaries. Both extension and filter dynamic-link libraries (DLL) may be unloaded if the memory is needed by another process.

API allows multiple commands in one DLL, implemented as member functions of the **CHttpServer** object in the DLL. CGI requires a separate name and URL mapping to a separate executable file for each task. Every new CGI request launches a new process, and each different request is contained in its own executable file, loaded and unloaded on each request, so overhead is higher than for APIs. (MSDN, 2001)

3. Active Server Processor (ASP)

When received with the suffix .asp web pages interfaced by the Internet processing servers, ASP processes the entire web page.

Active Server Pages use pre-defined object models to process the web pages to access and manipulate information from an HTML pages. These object models of ASP usage include:

- Request — Receives requests from end users in the browser.
- Response — Sends information to the browser in order to display it to the user.
- Session — Maintains information about the current user session and stores and retrieves state information.

- Application — Manages state that is shared across multiple webclass instances.
- Server — Creates other objects and determines server-specific properties that might influence the webclass's processing.
- BrowserType — Determines the capabilities of the user's browser and makes processing decisions based on that information.

The Active Server Processor in HTTP server is important to database applications because it provides a means to maintain state over the otherwise stateless HTTP protocol, it hosts a server scripting environment that can be used to perform all of the functions of a database application, and it publishes style sheets and Extensible Style Language documents. (Kroenke, 2000)

B. CONCLUSION

In 2002, users want to do their simple tasks through their browser, like buying books from the web. In order to manage these types of transactions, a web server plays a middleman role between the browser and the database server. The server side technology provides a dynamically published web page with a higher speed process and response time.

The application of this thesis is to maintain total maximum efficiency. This is done by the use of Microsoft's predefined ASP libraries with the interface of ISAPI in IIS 5.1 as HTTP server.

THIS PAGE INTENTIONALLY LEFT BLANK

V. DATABASE CONNECTION PROGRAMS

A. OPEN DATABASE CONNECTIVITY STANDARD (ODBC)

There are many types of database applications such as Relational databases, Oracle, SQL Server, Access, DB2. These applications have different purposes, different database management systems (DBMS), and different interfacing units. Web server application developers have to learn each individual DBMS product to use in their web applications. In order to incorporate and access multiple database applications made in different DBMSs, Open Database Connectivity (ODBC) interface is created.

Open Database Connectivity (ODBC) interface is a C programming language interface that makes it possible for applications to access data from a variety of database management systems (DBMSs). The ODBC interface permits maximum interoperability—an application can access data in diverse DBMSs through a single interface. Furthermore, that application will be independent of any DBMS from which it accesses data. Users of the application can add software components called drivers, which interface between an application and a specific DBMS. (MSDN, 2001)

ODBC consists of a set of standards by which SQL statements also can be issued and results and error messages returned. (Kroenke, 2000)

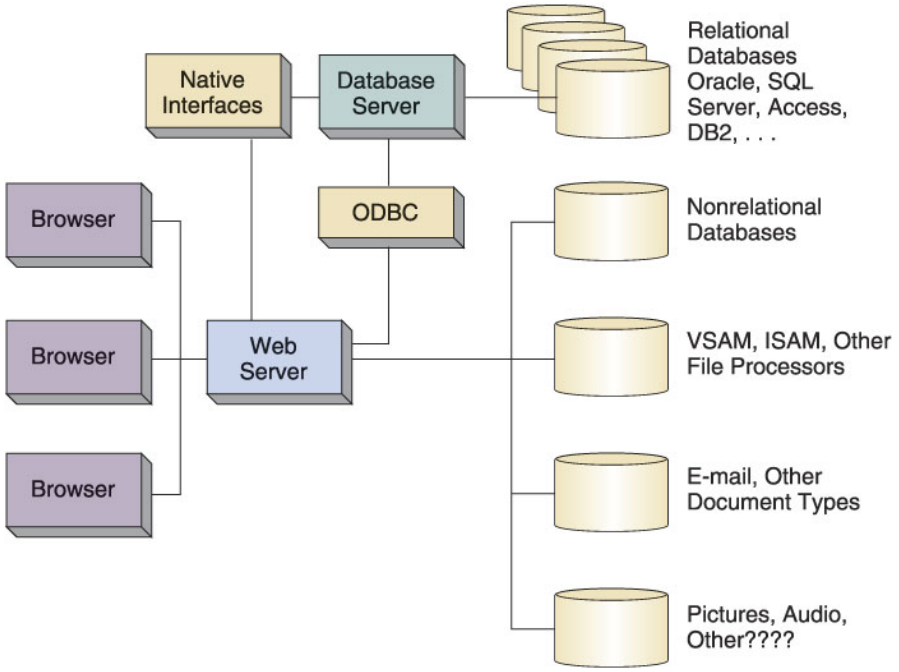


Figure 8. ODBC Role (From Kroenke, 2000)

B. OBJECT LINKING AND EMBEDDIGN DATABASE (OLE DB) CONNECTIVITY

ODBC has been a tremendous success and greatly simplified some database development tasks. As you will learn, it has a substantial disadvantage that was addressed by Microsoft when they developed OLE DB. Figure 9 shows the relationship of OLE DB, ODBC, and other data types. OLE DB provides an object-oriented interface to data of almost any type. DBMS vendors can wrap portions of their native libraries in OLE DB objects to expose their product's functionality through this interface. OLE DB can also be used as an interface to ODBC data sources. Finally, OLE DB was developed to support the processing of non-relational data as well. (Kroenke, 2000)

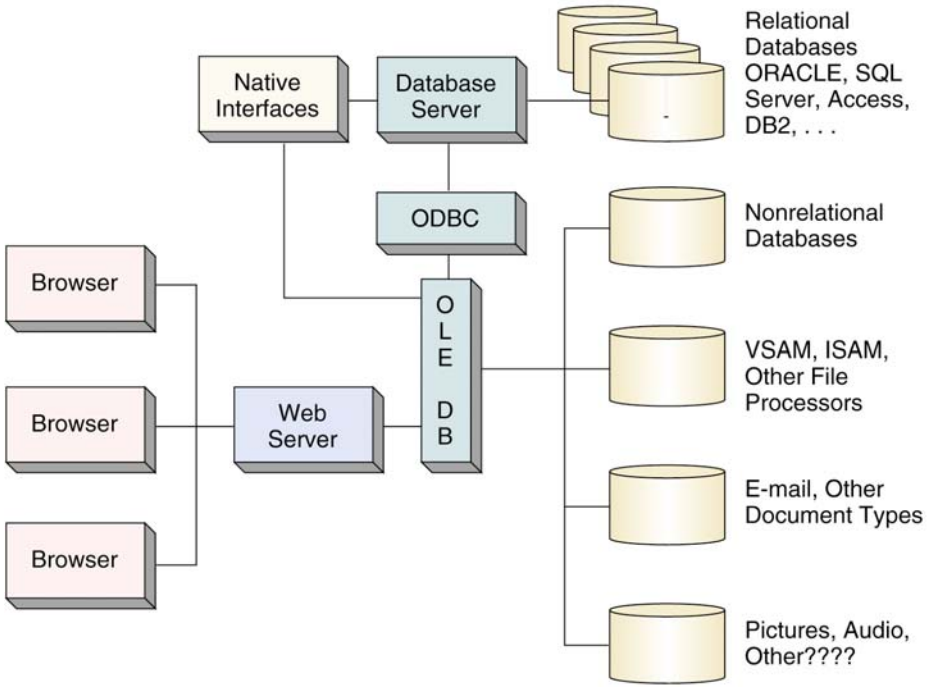


Figure 9. OLE Role (From Kroenke, 2000)

C. ACTIVEX DATA OBJECTS (ADO) CONNECTIVITY

ActiveX Data Objects (ADO) provide the mechanism for us to connect to a variety of different databases and data services. ADO is referred to as a thin layer on top of OLE DB. OLE DB is a data service on top of ODBC. What that all means is that if your database has an ODBC driver, you can access it through ADO. What ADO provides on top of OLE DB are methods that make accessing and working with data in your database simple. (Bugzek, 2000)

Because OLE DB is an object-oriented interface, it is particularly suited to object-oriented languages like C++. Most database application developers, however, program in Visual Basic, or scripting languages such as VBScript and JScript. Consequently, Microsoft edited ADO as a cover over OLE DB objects (see Figure 10), ADO enables programmers in almost any language to be able to access OLE DB functionality. Additionally, the ADO interface is designed to work in conjunction with Remote Data Services objects. These objects can be used to cache and process data on client computers. The objects shown in Figure 10 are interfacing with ADO objects on the Web server.

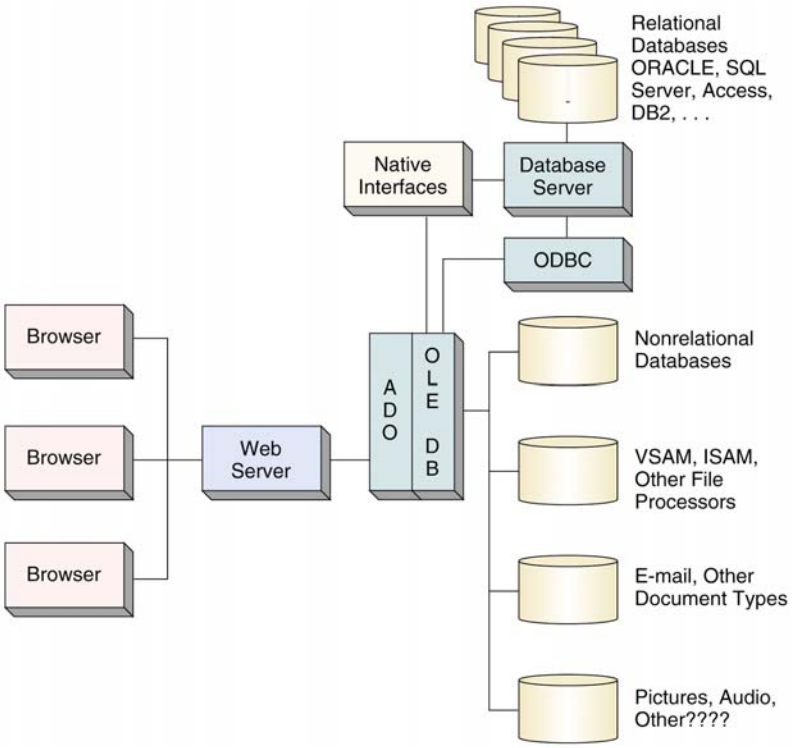


Figure 10. ADO Role (From Kroenke, 2000)

You may feel uncomfortable with the strong Microsoft presence in this discussion. Both OLE DB and ADO were developed and promulgated by Microsoft, and even ODBC received prominence in large measure because of support from Microsoft. In fact, other vendors and standards committees did propose alternatives to OLE DB and ADO, but because Microsoft Windows resides on nearly 90 percent of the world's desktops, it is difficult for others to promulgate opposing standards. (Kroenke, 2000)

D. CONCLUSION

In order to provide database connectivity to web applications, ODBC was developed by a committee of industry experts from the X/Open and SQL Access Group committees. ODBC was not supporting object oriented database applications. Therefore Microsoft developed OLE DB.

ADO was created to allow VBScript, Jscript, Visual Basic, Java and C++ to interface OLE DB to use object oriented database applications and ODBC.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. DESIGN AND IMPLEMENTATION

The implementation was based on the defined requirements of the project in the Chapter II. In order to provide the defined requirements, the following procedures are followed:

- Built the architecture of the system
- Built the database system for the project
- Defined site hierarchy and built html and asp web pages
- Screen shots of the web pages
- Set the security functions of the web pages
- Adaptation to web based information technology

A. THE ARCHITECTURE OF THE SYSTEM

Three-tier client server architecture is going to be used based on the defined benefits in the Chapter III. For the client side of the application, Microsoft Internet explorer, Mosaic, and Netscape Navigator browsers will be supported.

Internet Information Server (IIS) 5.1 will be used as the HTTP server in the middle tier. In the IIS 5.1, ISAPI will interface the active server pages. Application service provider will execute the VBScripts and Java Scrips and send SQL commends to the database server in the form that database server interpret. Microsoft Access 2000 will be used in the database server as the third tier. With the growth of the requirements, oracle or SQL server can be implemented to the project easily; because three-tier architecture provides high interoperability capability to the project. TCP/IP protocol will be used in the client – server interactions. ODBC, ADO and OLE DB will be used for the database server-web server interactions. Figure 11 shows an overview of information flows in the application architecture. Numbers on the flow arrows indicate the order in which the messages flow over the indicated paths.

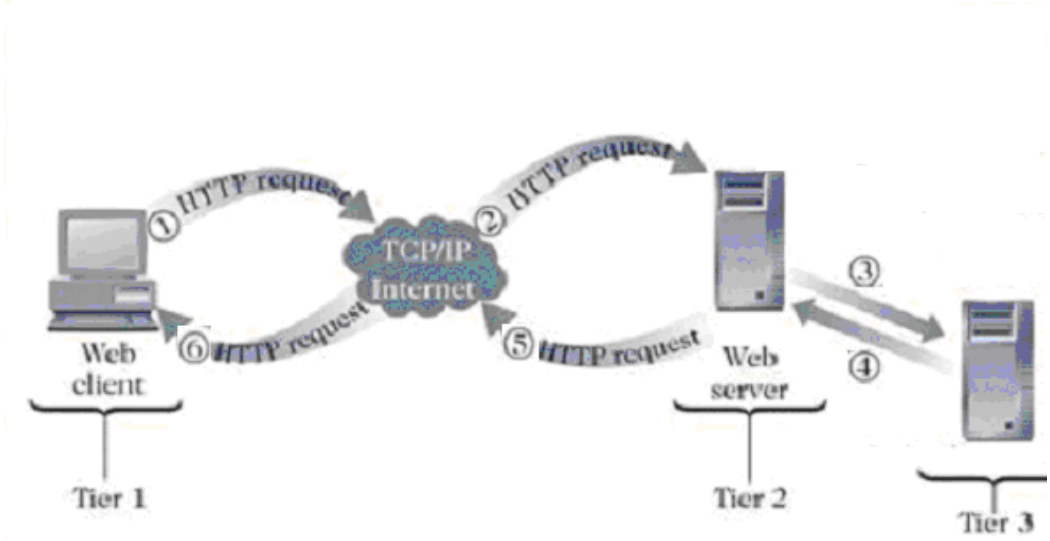


Figure 11. System Architecture (After Schneider, 2002)

B. THE DATABASE SYSTEM

Relational database and normalization is used to generate the tables used in the project. Tables and their relations are shown in the Figure 12. (See Appendix B for Database Management systems)

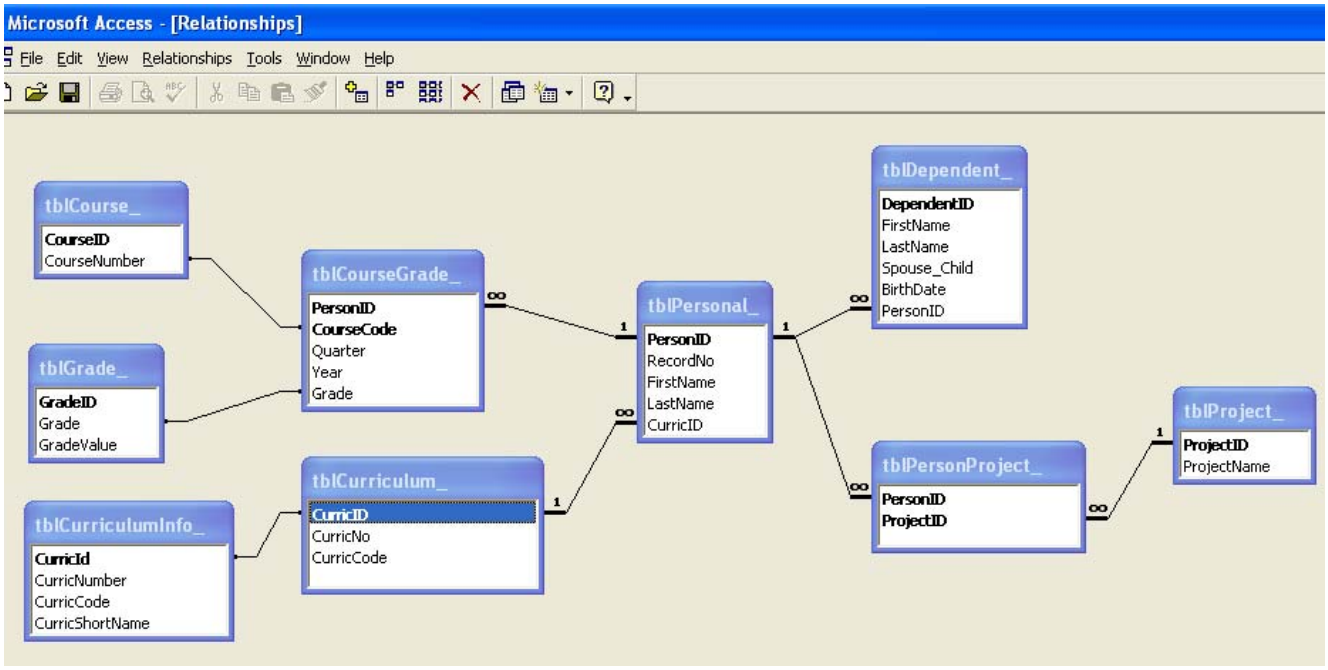


Figure 12. Relationships of the tables

Network computing system is shown in Figure 13.

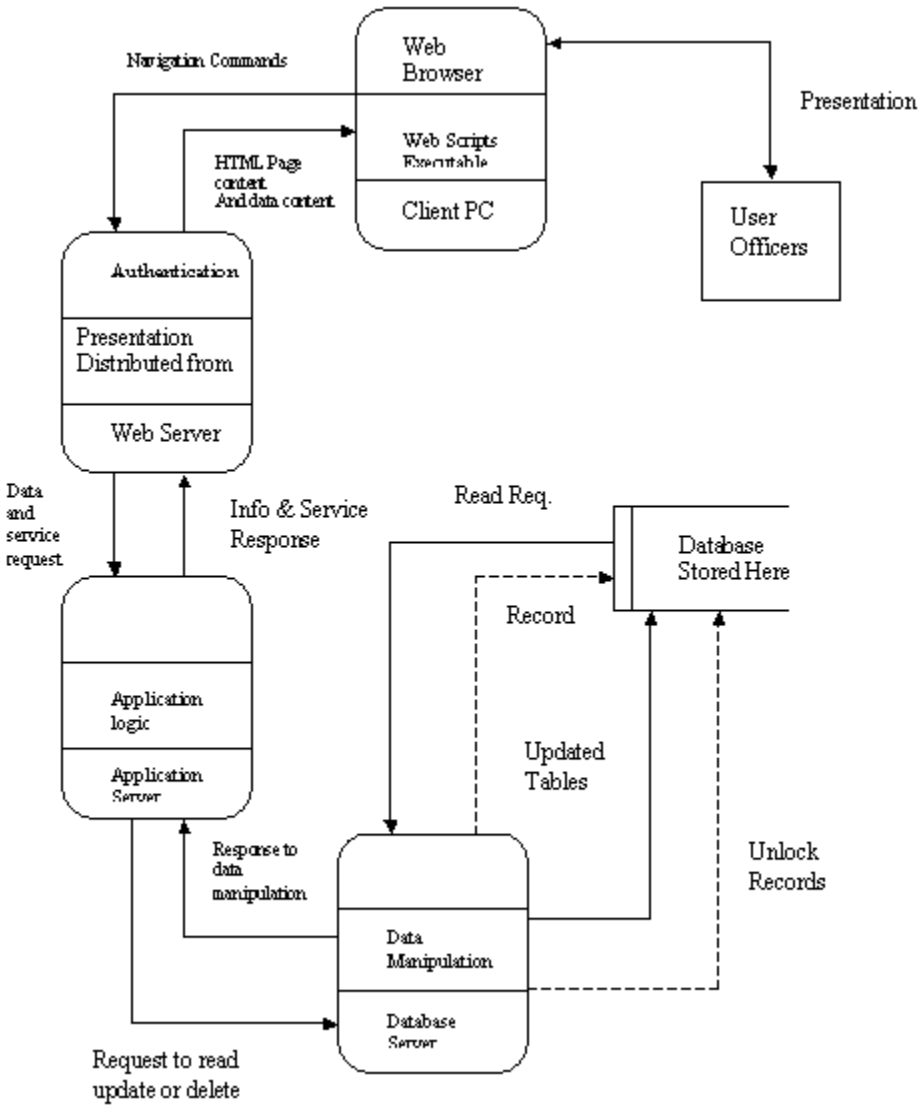


Figure 13. Network Computing System

C. SITE HIERARCHY

Each of the boxes in the Figure 14 represents one of the pages in the site. All the pages are Active Server pages, except the warning page, which contains HTML only. In order to provide good interfaces, web design criteria and for security issues US Navy Web Policy (R211930Z) are taken into consideration.

According to US Navy Web policy, sites must have:

- Privacy and security notice
- Web master contact info in home page
- "Approved by" statement in home page
- Links to www.navy.mil and www.navyjobs.com
- Notice stating it is an official Navy web site.

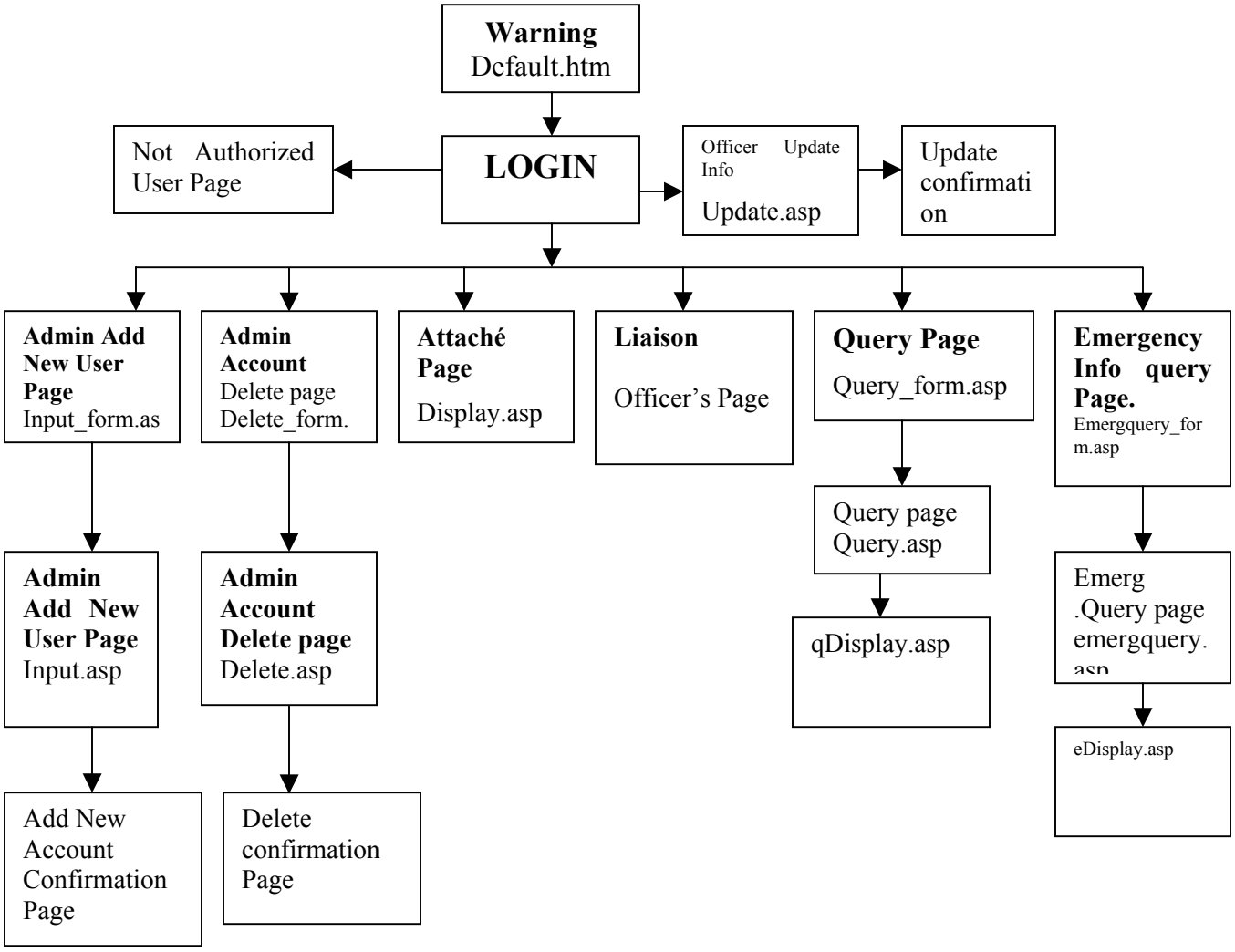


Figure 14. Web Pages Hierarchy

D. SCREEN SHOTS OF THE SYSTEM

In this section some of the screen shots for each user of the system are presented.

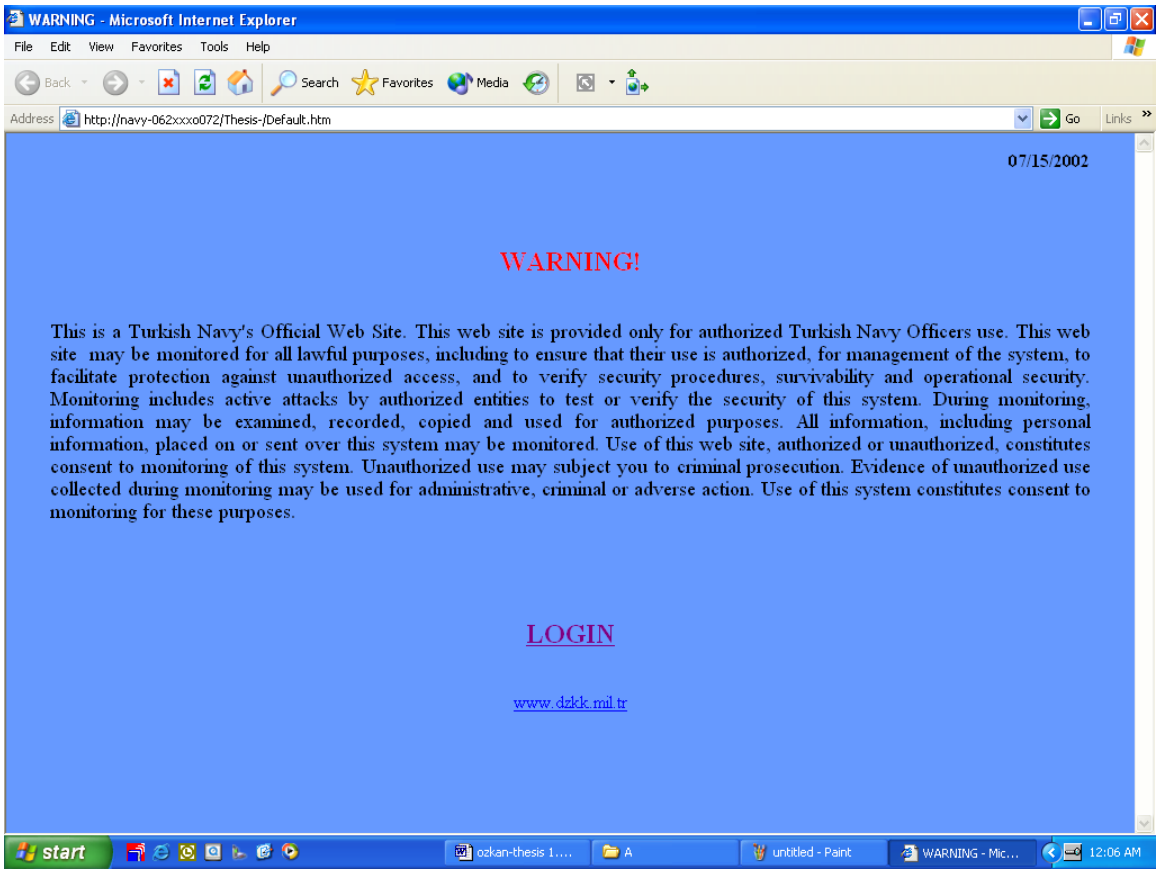


Figure 15. Warning page

This is the default page for the project. All the other pages will be accessed after this page.

Figure 16. Login page

In order to prevent officers to see other officer's personal records, additional password is provided for the officers.

Figure 17. Personal record update page for the officers

Officers will be able to update their personal information in this page and all the information will be updated in the database.

Figure 18. Personal information display in tabular form for attaché and liaison officer

Liaison officer and the attaché will be able to see all the information in this page.

Figure 19. Personal information display in list form for attaché and liaison officer

Attaché and the liaison officers will be able to see all the information of the officers individually.

Figure 20. Adding new user by the administrator

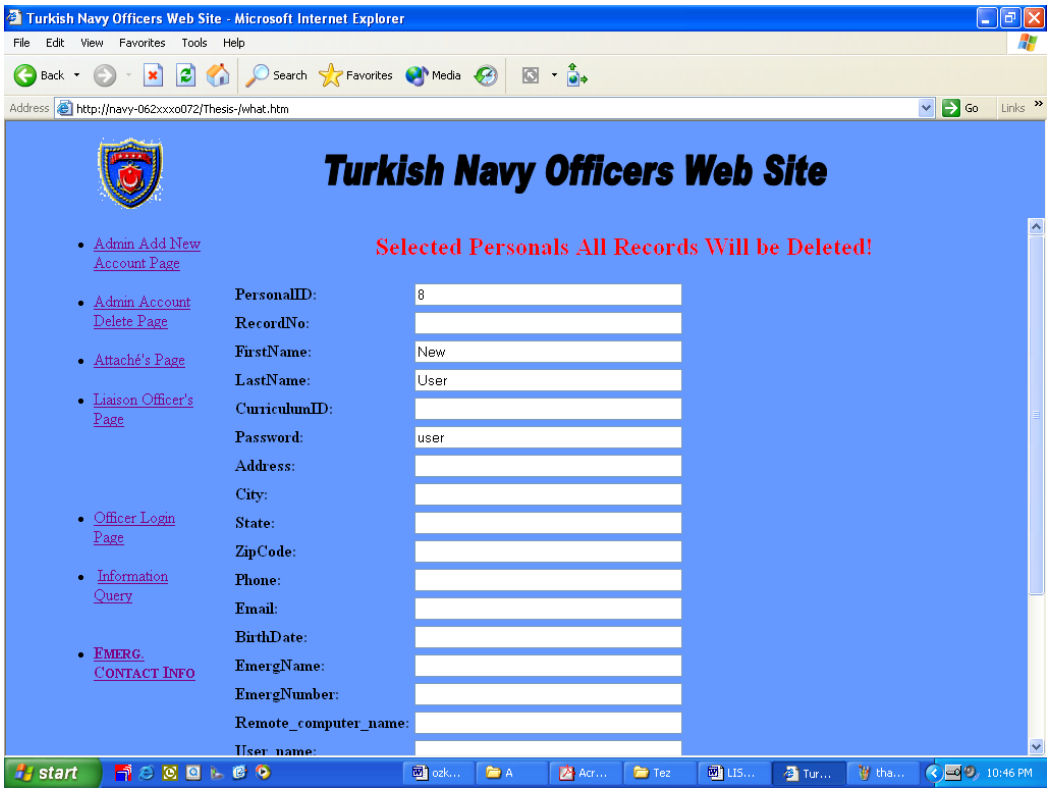


Figure 21. Deleting account by Administrator

Figure 22. Account delete confirmation

Figure 23. Query for the officers phone numbers

Officers can get other officers phone numbers by using query functions in this page

Figure 24. Emergency contact information query

Emergency contact information of the officers' can be queried here.

E. SECURITY FUNCTIONS OF THE WEB PAGES

IIS 5.1 supports the US government security standard, commonly called Fortezza. This standard satisfies the Defense Message System security architecture with a cryptographic mechanism that provides message confidentiality, integrity, authentication, and access control to messages, components, and systems. These features can be implemented both with server and browser software. (Microsoft .com) NTFS and IIS security functions are used for each web page to restrict the unauthenticated users to access and control the web pages. In order to provide this, an account for each user is provided and Officers, Attaché, Liaison, and Administrator groups are created in Microsoft Management Console (MMC). See Figure 25, Figure 26 and Figure 27

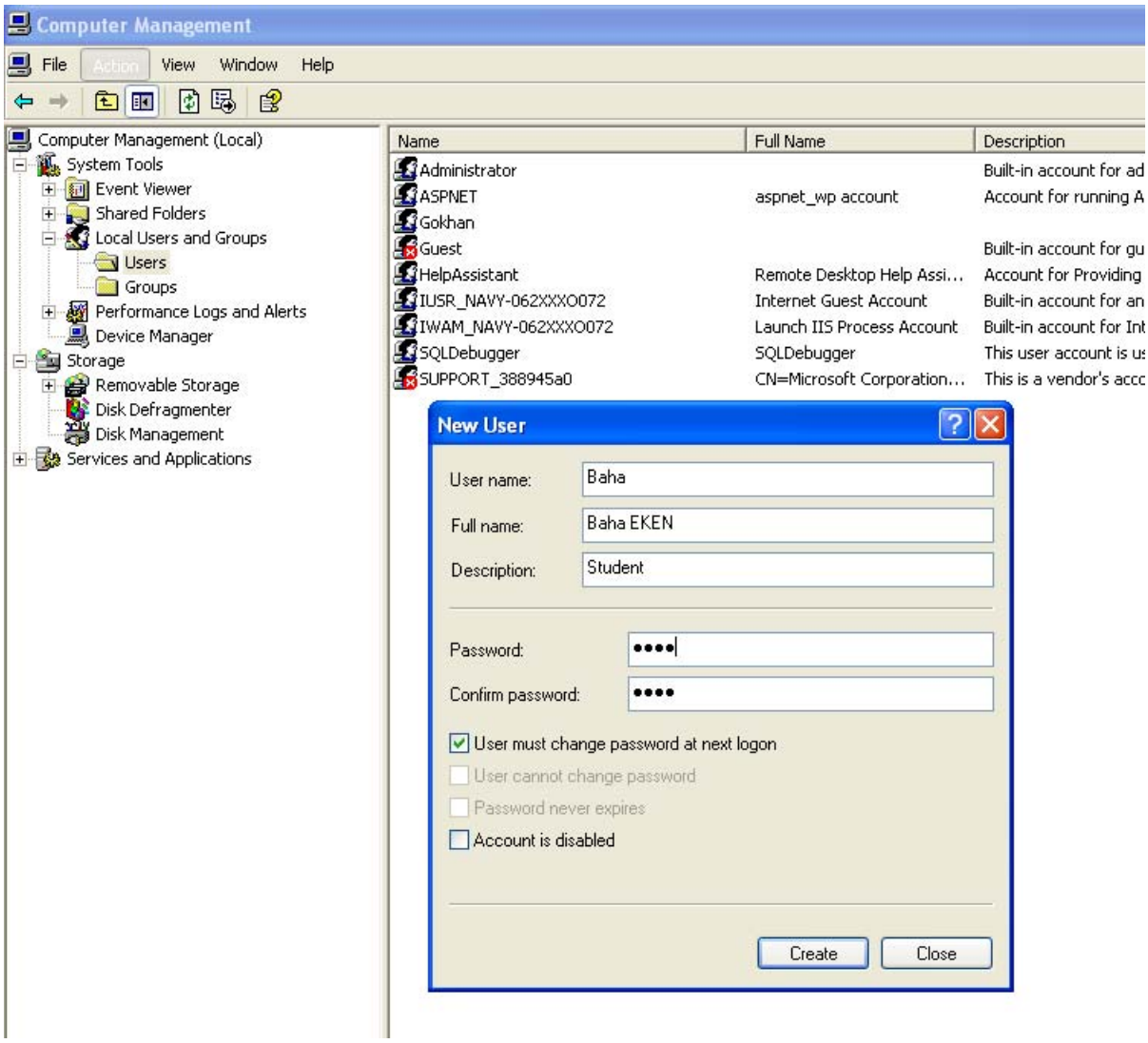


Figure 25. Create an account in Microsoft Management Console

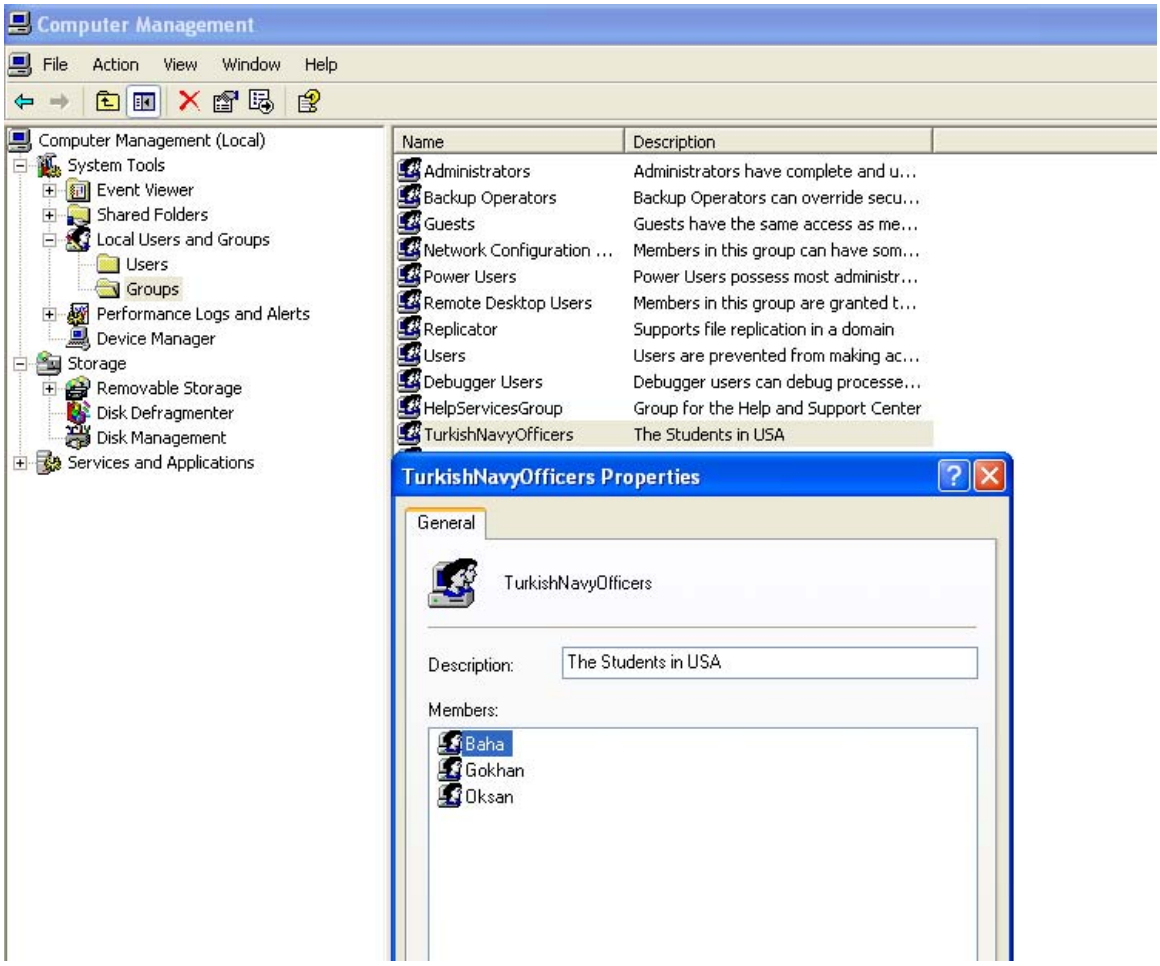


Figure 26. Assigning accounts into groups in MMC

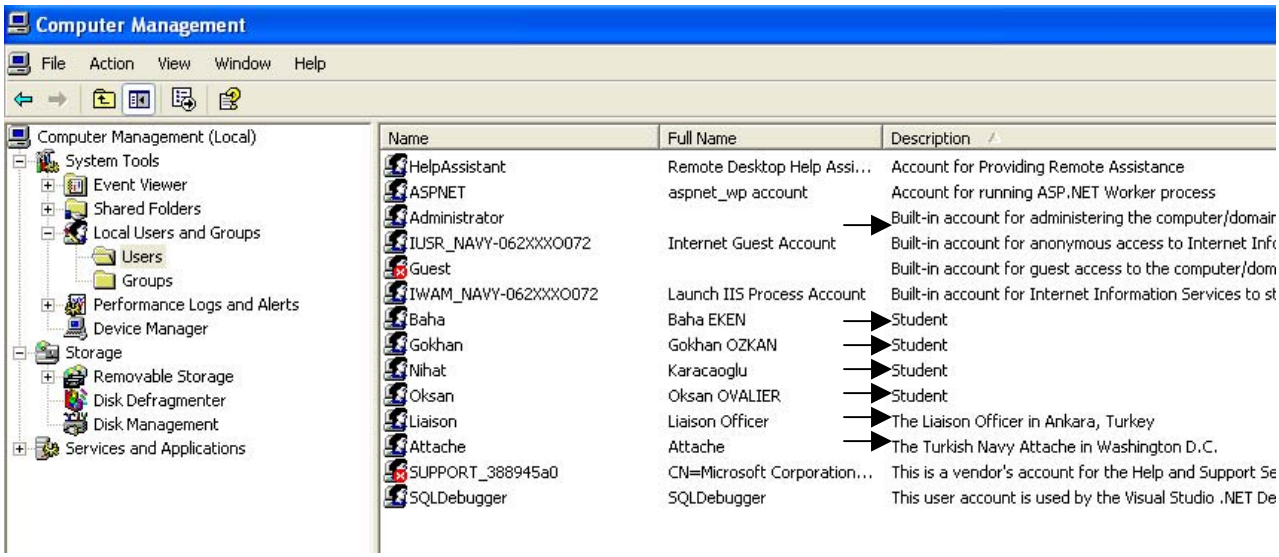


Figure 27. The users and their groups

Then, each user groups are assigned to specific web pages. Figure 28

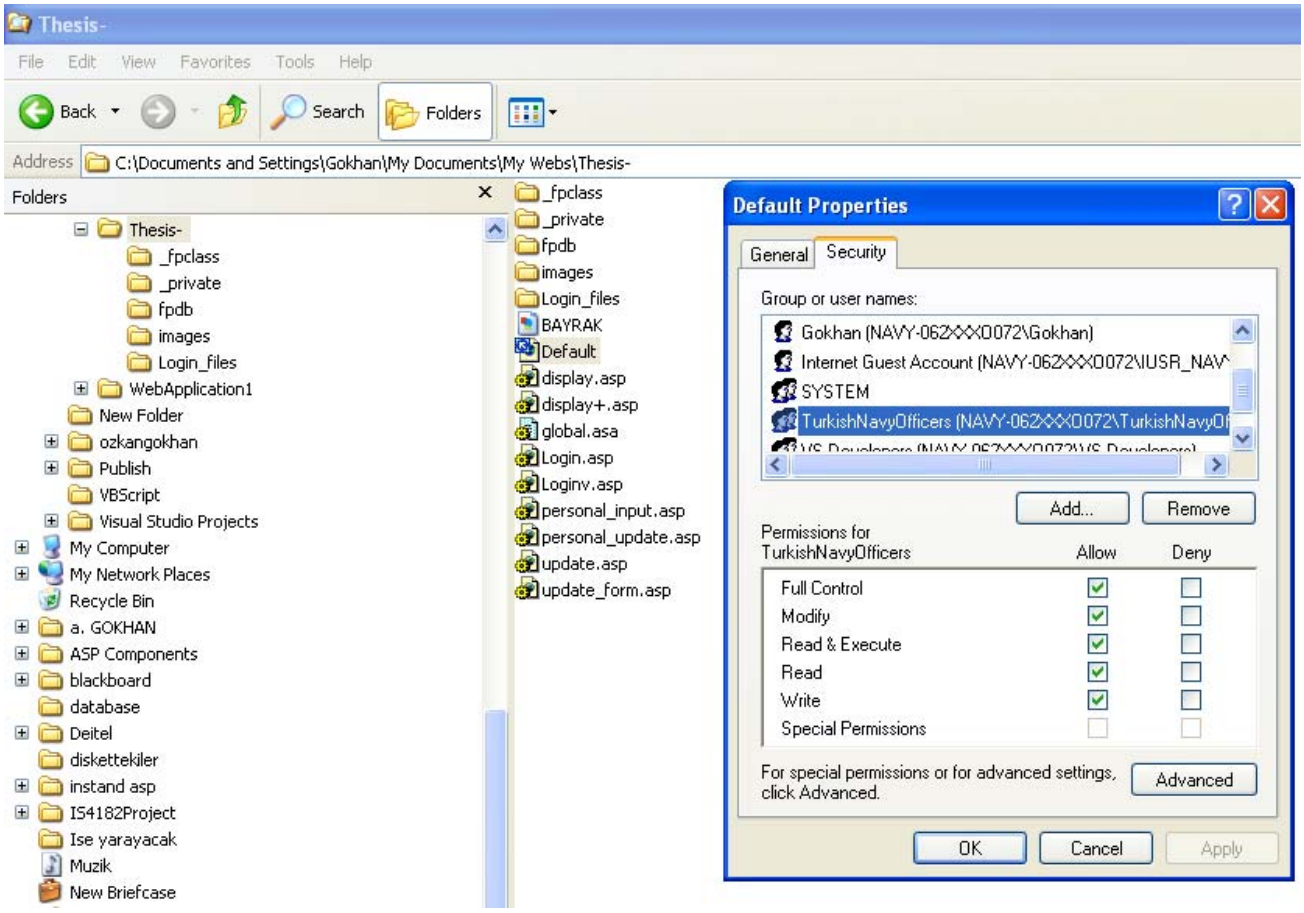


Figure 28. Assigning groups into specific web pages and specifying permissions

Officers have to provide both server password and database passwords in order to make changes in their data. All the attempts to log in the pages and the activities will be recorded in the server log files. (Figure 29 and Figure 30)

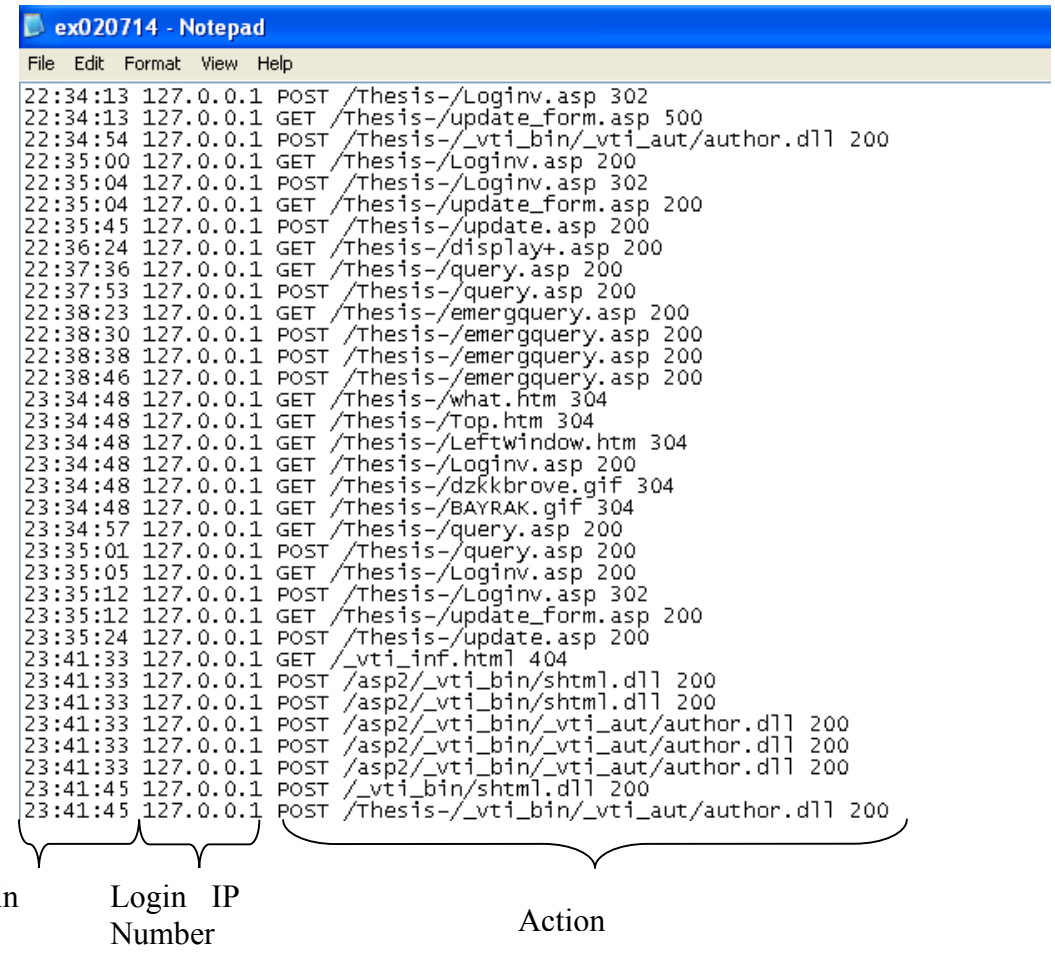


Figure 29. Web Server Log files

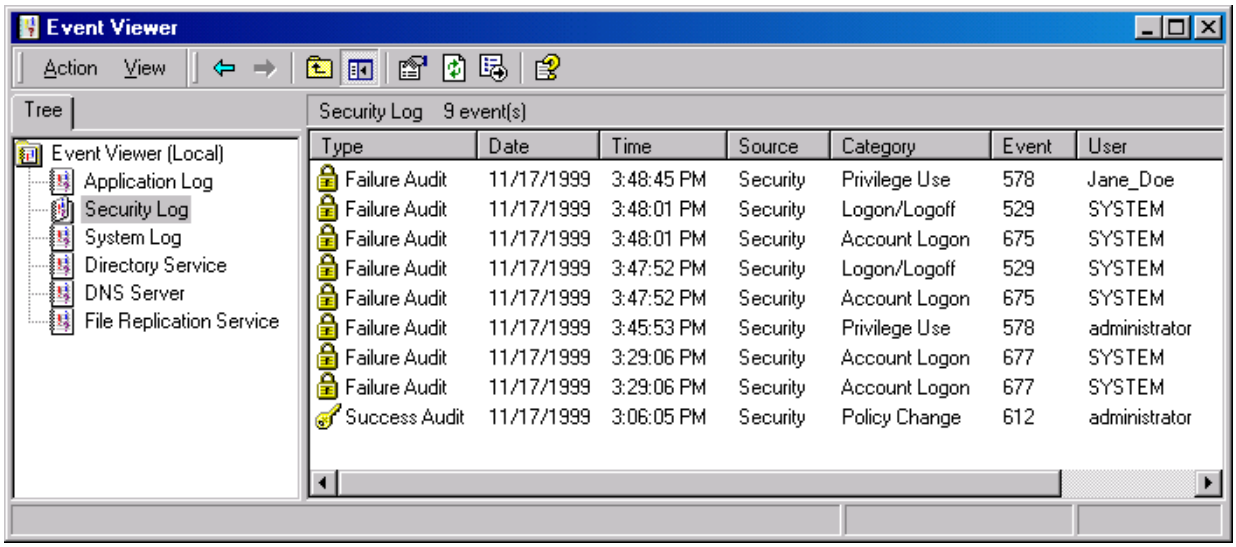


Figure 30. Web server log files (from Microsoft Windows 2000 Server, 2000)

Default port number for the web pages is port 8080. However, in IIS MMC panel, any port number can be assigned for the web pages. Assigning a port number different from the default one enhanced the security of the project. (Figure 31)

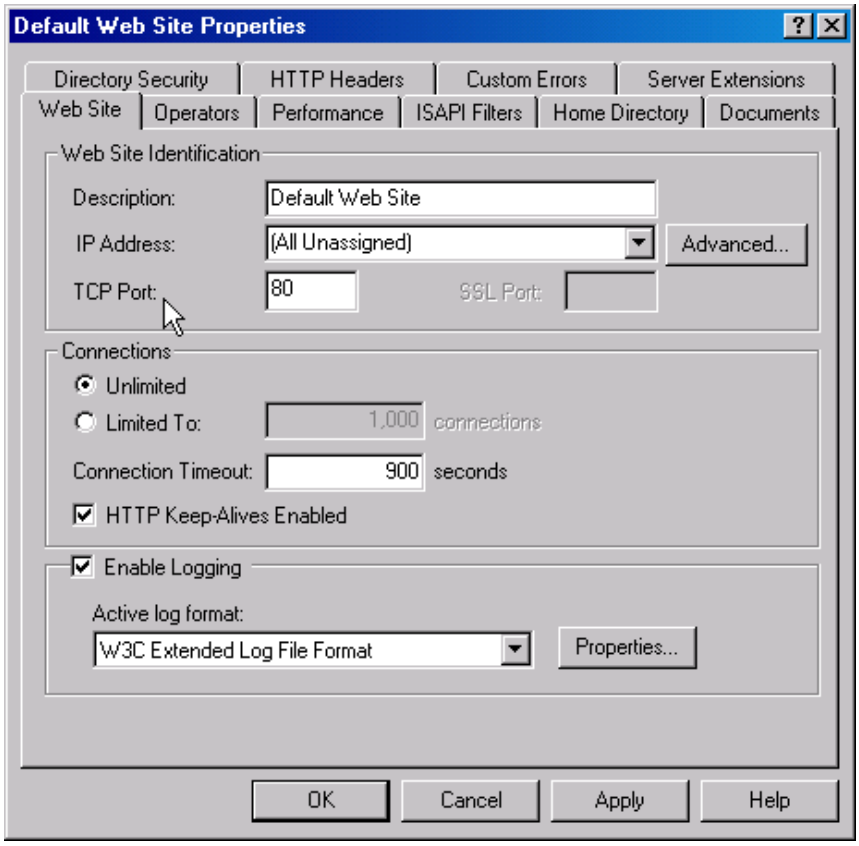


Figure 31. Assigning Port number to the web pages

F. ADAPTATION TO CLIENT SERVER BASED MANAGEMENT

Many companies are trying to implement their business models to client server architecture due to the benefits of the web-based applications defined in Chapter III. According to Peter and Chengalur-Smith (1998), only 16% of all the client server applications are successful.

According to the data collected from 350 US nationwide client server applications, Peter and Chengalur-Smith (1998) defines three problem areas. These problem areas are:

1. Computer Architecture

- Inadequate internal skill set
- Unanticipated extra costs

- Supporting multiple vendor products
- Continual troubleshooting activities
- Performance degradation as the number of components increased
- Interoperability of components
- Quality and reliability of middleware
- Quality and reliability of network

2. Management and Organization

- Corporate policies and politics
- Inadequate user involvement
- Senior management's inability to create an integrated technology strategy
- Organizational change brought about by the new system
- Inadequate management support
- Management partnership between IS and functional managers
- Other business activities
- IS group's inability to be user oriented

3. Conversion and Maintenance

- Inadequate disaster recovery
- Inadequate security
- Conversion of mainframe applications
- Knowledge of corporate data availability

In order to mitigate many management and organizational problems, Peter and Chengalur-Smith (1998) suggests companies to:

- Ensure that senior managers establish the business direction, align the IT strategy with it, and make client/server applications an integral part of the IT strategy.
- Encourage IS managers to be more business oriented and less systems oriented, help business functions conceptualize client/server applications, and integrate those applications with the IT strategy.
- Make IS management responsible for planning the company's computing infrastructure based on an understanding of business function requirements and company strategy.
- Give functional managers a role in developing IT strategy and specifying client/server requirements.

- Form a project team with technically skilled people and have them create a detailed migration plan for all of the network, systems, management, and organizational changes required by the application.
- Provide the resources needed for hardware, software, and network requirements, as well as for education, training, product support, and systems operation activities, and budget them accordingly. (Of total systems costs, those for client hardware and software are 9%; for servers, printers, and communications, 9%; and for relational database and systems management, 2%. These costs can be shared by the affected functional areas.)
- • Evaluate client/server projects for their strategic fit with the business as well as for their expected benefits and costs.

Execution can be anticipated to be difficult, or it can be approached as a creative opportunity to improvise, experiment, and prototype to discover the best way to accomplish strategic moves. (Boar, 2001)

In the implementation of this project, the same organizational and management problems may occur in small scale. Before implementing large-scale client server applications, this kind of projects presented in this thesis can be used to define the problems and to resolve with the suggested solutions. In this way, client server systems can reach higher success ratio.

G. FUTURE APPLICATION AREAS OF THIS SYSTEM

If security issues related with the Internet could be handled, this system would be applied to all Navy personnel as platform-independent system solutions for Human Resources. Providing the capability for each navy personal to maintain his or her own information through the web will bring a better manpower data for the headquarters with the fewer errors at a lower cost.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A

UNIFIED MODALING LANGUAGE

The Unified Modeling Language (UML) is the industry-standard modeling language for specifying and documenting both data and processes in software systems. The UML was released in November 1997 by a consortium of companies led by Rational Software Corporation. (MSDN, 2001)

UML has a broad spectrum of usage. It can be used for business modeling, software modeling in all phases of development and for all types of systems, and general modeling of any construction that has both a static structure and a dynamic behavior. In order to achieve these wide-ranging capabilities, the language is defined to be extensive and generic enough to allow for the modeling of such diverse systems, avoiding the too specialized and too complex.

The general parts of UML:

- Views: Views show different aspects of the system that are modeled. A view is not a graph, but an abstraction consisting of a number of diagrams. Only by defining a number of views, each showing a particular aspect of the system, can a complete picture of the system be constructed. The views also link the modeling language to the method/process chosen for development.
- Diagrams: Diagrams are the graphs that describe the contents in a view.
- Model elements: The concepts used in the diagrams are model elements that represent common object-oriented concepts such as classes, objects, and messages, and the relationships among these concepts including association, dependency, and generalization. A model element is used in several different diagrams, but it always has the same meaning and symbol.
- General mechanisms: General mechanisms provide extra comments, information, or semantics about a model element; they also provide extension mechanisms to adapt or extend the UML to a specific method/process, organization, or user. (Eriksson, 1998)

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B

DATABASE MANAGEMENT

A. DATABASE MANAGEMENT SYSTEM

A database management system (DBMS) is specialized computer software available from computer vendors that is used to create, access, control, and manage the database. The core of the DBMS is often called its database engine. The engine responds to specific commands to create database structures and then to create, read, update, and delete records in the database. (Whitten, 2001)

Marakas (1999) explains the general functions of DBMS are as follows:

Data Definition

- Provides a data definition language (DDL) that allows users to describe the data entities and their associated attributes and relationships
- Allows for the interrelation of data from multiple sources

Data Manipulation

- Provides the user with a query language to interact with the database
- Allows for capture and extraction of data
- Provides rapid retrieval of data for ad hoc queries and reports
- Allows for the construction of complex queries for retrieval and data manipulation

Data Integrity

- Allows the user to describe rules (integrity constraints) to maintain the integrity of the database
- Assists in the control of erroneous data entry based on the defined integrity constraints

Access Control

- Allows identification of authorized users
- Controls access to various data elements and data manipulation activities within the database
- Tracks usage and access to data by authorized users

Concurrency Control

- Provides procedures for controlling simultaneous access to the same data by more than one user
- Transaction Recovery
- Provides a mechanism for restart and reconciliation of the database in the event of hardware failure
- Records information on all transactions at certain points to enable satisfactory database restart

Database engineers define database types according to the way they structure the records, and these are:

- Relational,
- Hierarchical,
- Network,
- Object oriented,
- Object relational model.

Early database management systems organized records in hierarchies or networks implemented with indexes and linked lists. But today, most successful database management systems are based on relational technology. (Whitten, 2001)

B. RELATIONAL DATABASE MANAGEMENT SYSTEM

The relational model is important for two reasons. First, because the constructions of relational model are broad and general, it can be used to express DBMS-independent database designs. Second, the relational model is the basis for almost all DBMS products. (Kroenke, 2000)

Relational databases implement data in a series of two-dimensional tables that are "related" to one another via foreign keys. Each table (sometimes called a *relation)* consists of named columns (which are fields or attributes) and any number of unnamed rows (which correspond to records).

Figure 32 illustrates a logical data model. Figure 33 is the physical, relational database implementation of that data model (called a **schema).** In a relational database,

files are seen as simple two-dimensional tables, also known as relations. The rows are records. The columns correspond to fields.

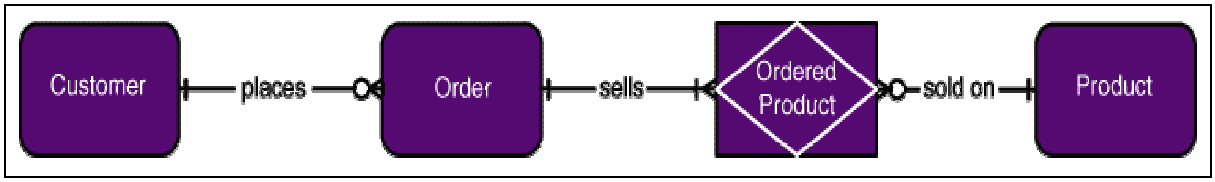
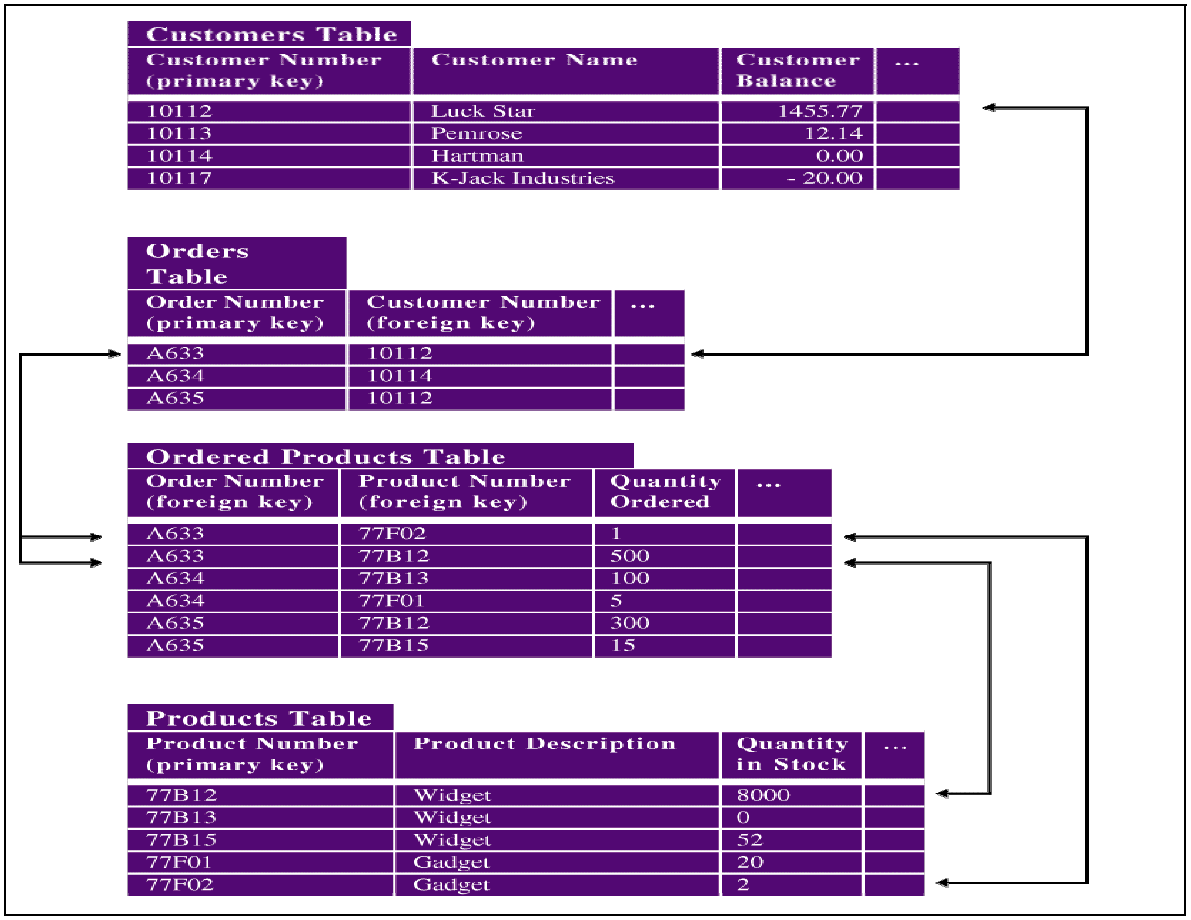


Figure 32. A Simple Logical Data Model (From Whitten, 2001)



Customers Table

Customer Number (primary key)	Customer Name	Customer Balance	...
10112	Luck Star	1455.77	
10113	Penrose	12.14	
10114	Hartman	0.00	
10117	K-Jack Industries	- 20.00	

Orders Table

Order Number (primary key)	Customer Number (foreign key)	...
A633	10112	
A634	10114	
A635	10112	

Ordered Products Table

Order Number (foreign key)	Product Number (foreign key)	Quantity Ordered	...
A633	77F02	1	
A633	77B12	500	
A634	77B13	100	
A634	77F01	5	
A635	77B12	300	
A635	77B15	15	

Products Table

Product Number (primary key)	Product Description	Quantity in Stock	...
77B12	Widget	8000	
77B13	Widget	0	
77B15	Widget	52	
77F01	Gadget	20	
77F02	Gadget	2	

Figure 33. A simple Physical Database Schema (From Whitten 2001)

The following shorthand notation for tables is commonly encountered in systems design and database books.

CUSTOMERS (CUSTOMER-NUMBER. CUSTOMER-NAME, CUSTOMER-BALANCE, . . .)
ORDERS (ORDER-NUMBER. CUSTOMER-NUMBER (FK), . . .)

ORDERED-PRODUCTS (ORDER-NUMBER (FK:), PRODUCT-NUMBER (FK), QUANTITY-ORDERED,)

PRODUCTS (PRODUCT-NUMBER. PRODUCT-DESCRIPTION, QUANTITY-IN-STOCK, . .)

Both the DDL and DML of most relational databases is called **SQL** (pronounced *S-Q-L* by some and *sequel* by others). SQL supports complete database creation, maintenance, and usage. To access data in tables and records, SQL provides the following basic commands:

- SELECT specific records from a table based on specific criteria (e.g., SELECT CUSTOMER WHERE BALANCE > 500.00).
- PROJECT out specific fields from a table (e.g., PROJECT CUSTOMER TO INCLUDE ONLY CUSTOMER-NUMBER, CUSTOMER-NAME, BALANCE).
- JOIN two or more tables across a common field—a primary and foreign key (JOIN CUSTOMER AND ORDER USING CUSTOMER-NUMBER).

When used in combination, these basic commands can address most database requirements. A fundamental characteristic of SQL is that commands return a set of records, not necessarily just a single record (as in nonrelational database and file technology). SQL databases also provide commands for creating, updating, and deleting records, as well as sorting records. (Whitten, 2001)

LIST OF REFERENCES

Orfali, R. H., Edwards, J., *The Essential Client/Server Survival Guide*, Third Edition, John Wiley & Sons, 1999

Kroenke, David M., *Database Processing: Fundamentals, Design, And Implementation*, Prentice-Hall, 2000

Topuz, Rasim, *Web-based database applications: An educational, Administrative Management System for Military Academies*, NPS, 2002

Bulzeg, Greg, *Instant ASP Scripts*, Mc Graw Hill, 2000

April 2001 release of the MSDN Library

Boar, Bernard, *The Art of Strategic Planning for Information Technology*, Second Edition, John Wiley & Sons, 2001

Peter, Duchessi and InduShobha Chengalur-Smith, *Client/Server Benefits, Problems, Best Practices*, Communication of the ACM, May 1998/Vol.41.No.5

Schneider, Gary P., *Electronic Commerce*, Third Annual Edition, Thomson Learning, 2002

Microsoft, *Windows 2000 Server Academic Learning Series*, Microsoft, 2000

Reeves, Scott and Kalinda Reeves, *IIS 5 On Site*, Coriolis, 2001

Evans, Jessica and Roger Hayen, *Microsoft FrontPage 2002*, Thomson Learning, 2002

Whitten, J.L., Bently, L. D., Dittman, K. C., *System Analysis and Design Methods,* McGraw-Hill, fifth edition, 2001.

Eriksson, H., Penker, M., *UML Toolkit*, John Wiley & Sons, 1998

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. DEFENSE TECHNICAL INFORMATION CENTER
   Ft. Belvoir, Virginia

2. DUDLEY KNOX LIBRARY
   Naval Postgraduate School
   Monterey, California

3. DENIZ KUVVETLERI KOMUTANLIGI
   Personel Daire Baskanligi
   Bakanliklar 06410
   Ankara, TURKEY

4. DENIZ KUVVETLERI KOMUTANLIGI
   Kutuphanesi
   Bakanliklar 06410
   Ankara, TURKEY

5. DENIZ HARP OKULU
   Kutuphanesi
   Tuzla 81740
   Istanbul, TURKEY

6. KARA HARP OKULU
   Kutuphanesi
   Bakanliklar 06410
   Ankara, TURKEY

7. HAVA HARP OKULU
   Kutuphanesi
   Yesilyurt
   Istanbul, TURKEY

8. DAN BOGER
   Naval Postgraduate School
   Monterey, California

9. WILLIAM J. HAGA
   Naval Postgraduate School
   Monterey, California

10. GARY PORTER
    Naval Postgraduate School
    Monterey, California

11. LTJG.GOKHAN OZKAN
    Envanter Kontrol Merkezi Komutanligi
    Golcuk 41650
    Kocaeli, TURKEY