

Fall 11-1-2007

Software Project Management System

Arun Kumar Nampally
Dakota State University

Follow this and additional works at: <https://scholar.dsu.edu/theses>

Recommended Citation

Nampally, Arun Kumar, "Software Project Management System" (2007). *Masters Theses*. 138.
<https://scholar.dsu.edu/theses/138>

This Thesis is brought to you for free and open access by Beadle Scholar. It has been accepted for inclusion in Masters Theses by an authorized administrator of Beadle Scholar. For more information, please contact repository@dsu.edu.

Software Project Management System



A graduate project submitted to Dakota State University in partial fulfillment of the requirements for the degree of

Master of Science

In

Information Systems

Nov, 2007

By

Arun Kumar Nampally

Project Committee:

Stephen Krebsbach, Ph.D. Project Advisor

Mark Moran, Ph.D., Committee Member

Ronghua Shan, Ph.D., Committee Member

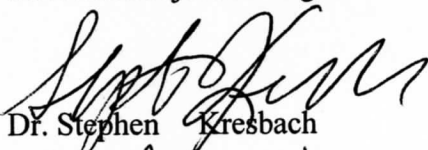


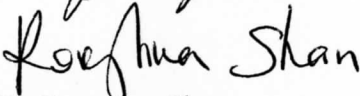
I. PROJECT APPROVAL FORM

We certify that we have read this project and that, in our opinion, it is satisfactory in scope and quality as a project for the degree of MSIS Master of Science in Information Systems.

Student Name: Arun Kumar Nampally

Master's Project Title: Software Project Management Tool

Supervisor:  Dr. Stephen Kresbach Date: 11/30/07

Committee member:  Dr. Ronghua Shan Date: 11/30/07

Committee member:  Dr. Mark Moran Date: 11/30/07

Acknowledgment

I would like to express my gratitude to all those who gave me the possibility to complete this project work. I want to thank my committee members Dr. Stephen Krebsbach, Dr. Ronghua Shan and Dr. Mark Moran who encouraged me constantly through out my project.

I have further more to thank all the teaching staff and non- teaching staff who contributed in one way or the other towards successful completion of my Master's program.

I deeply indebted to our dean of graduate studies Dr.Omar El-Gayar whose help, stimulating suggestions and encouragement helped me in all the times of my Master's program.

Finally I would like to give my special thanks to my family members for their support and also thank my fellow students who made my stay at DSU memorable.

Abstract

Project resource management has become very important in this techno world for a company.

The benefit of having fast and accurate data for a project manager in a company helps to complete the project in a better organized way. There is a need to develop a tool for a project manager who can organize, schedule, maintain, manage, monitor and estimate different modules of a project. In order to meet this requirement SPMT (Software Project Management Tool) was developed.

The SPMT has Client registration, View registered clients, Project registration and View Projects registered, View Employees, View Technologies in a project. with these items in the tool helps a project manager to know who the clients are, what projects undergoing for a particular client, schedule and assign employees for a new project, scheduled and assigned employees for a existing project, monitor a project, manage all the projects, know the current status of a project at any point of time, Expected start date, Expected project delivery date and thus better organization of project resources.

Declaration

I hereby certify that this project constitutes my own product, that where the language of others is set forth, quotation marks so indicate, and that appropriate credit is given where I have used the language, ideas, expressions or writings of another.

I declare that the project describes original work that has not previously been presented for the award of any other degree of any institution.

Signed,

__Arun Kumar Nampally__

Table of Contents

Project Approval Form -----	II
Acknowledgement -----	III
Abstract -----	IV
Declaration -----	V
Table of Contents -----	VI
List of Tables -----	VII
List of Figures -----	VII
Introduction -----	1
System Analysis -----	3
Background of the Problem -----	3
Existing System -----	3
Proposed System -----	4
Metrics -----	4
Merits of Proposed System -----	6
System Design -----	9
System Overview -----	12
UML -----	16
JSP -----	44
Testing -----	55
Conclusion -----	69
Reference -----	70

List of Tables

Table 1: Interface Summary -----	46
Table 2: Class Summary -----	47
Table 3: Exception Summary -----	48
Table 4: Interface Summary -----	48
Table 5: Class Summary -----	49
Table 6: Client Login Table -----	59
Table 7: Project Table -----	59
Table 8: Employee Table -----	60
Table 9: Project Employee -----	61
Table 10: Project Technologies -----	61
Table 11: Technologies -----	61
Table 12: Project Status Table -----	62

List of Figure

Figure 1: Level Zero -----	13
Figure 2: Level One -----	14
Figure 3: Level Two -----	15
Figure 4: Use Case Diagrams -----	25
Figure 5: Sequence Diagram -----	26
Figure 6: Collaboratation Diagram -----	27
Figure 7: Activity Diagram -----	28
Figure 8: Login Page -----	63
Figure 9: Client Registration -----	64
Figure 10: Registered Clients -----	65
Figure 11: Project Registration -----	66
Figure 12: Registered Projects -----	67
Figure 13: Registered Employee -----	68

INTRODUCTION

PROJECT PROFILE

Software Project Management System , provides need has been felt to effectively monitor the activities of the project for giving better services through a Computerized Software Management System using Java.

Problem Definition:

In the earlier days, the project is managed using manual systems for control of Projects and Manager Records, Clients and employees on duty, Company history record. Due to manual systems the data is not available readily at the time of need.

A Software using Java, which is Operating System platform independent, to cover all the activities related to Project is proposed to have on-line enquiry of various details and build a database for generating reports. The Software system proposed will improve the efficiency in managing the Project as compared to existing manual system.

The Software mainly consists of

- Facility to extend application execution environment in different platforms using
 - Advanced Java
 1. JSP
 2. Apache Web Server
 3. JDBC

Expected benefits:

The expected benefits of the Software are:

- Reduction of time in retrieving the Data.
- Cost reduction and Cost control
- Maintenance of Project record.
- A Comprehensive database of all activities.

SYSTEM ANALYSIS

Introduction:

System Analysis is the first and most important step in the process of the development of a project, where we get a general idea about the needs of customers or end-users through interactions with them.

A system is a collection of functionally dependent components interactively working together to attain the goal.

Analysis is the process of diagnosing situations, done with a defiant aim, keeping in view the boundaries of the system, to produce a report based on the findings.

It is a fact-finding technique where in, System Requirements Analysis and Feasibility Analysis are performed. A system requirement analysis is performed to find out the requirements of the system and feasibility analysis is performed to find out if it is worth implementing the project, if it can be.

Existing system:

In the analysis of a system, the first thing that hits the developer's mind is, whether there exists any system similar to this one, already in existence. If such a system exists, a careful study of that system is made. If possible, it is studied thoroughly. In this study, the developer notes down both the merits as well as the demerits of that system, and then finds out why this project is being taken up by his customer i.e., why does the customer want it re-done.

In the case of this project, there is no other system already in existence. But a doubt certainly arises that, when a web-application like JSP is already being used,

what is the need of this application. To understand the need of this project, we need to analyze the merits and demerits of the JSP web-application.

Proposed system:

The Software Project Management System (SPMT) is designed for the “COHORTS” to replace their existing manual, paper based system. The new system is to control the following information, Project information, staff schedules and Staffing details i.e. Employees, Clients, and Managers. These services are too provided in efficient manner, with the goal of reducing the time and resources currently required for such tasks.

Metrics:

- ✓ Provide a fairly good user-friendly user interface Jsp forms.
- ✓ Increase Staff productivity and efficiency
- ✓ Standardizing data in fewer corrections and significantly lowering the incidence of missing or incorrect data.
- ✓ Ensuring data integrity and providing a data base for future statistical and management reporting.
- ✓ Reduces errors made in scheduling and making the system more reliable.

- ✓ More reliable system

Requirement Analysis:

The requirements analysis is done keeping in view, the system definition and is concerned with the study of the requirements of the customer. Requirements Definition is concerned with identifying the basic functions that are supposed to be performed by the software. This is done to find out what exactly the user expects from the project. This stage defines what we should do for the completion of the project to the user’s satisfaction. The output of this stage is the Software Requirements Specification.

In brief, the requirements for this project can be stated as follows:

- ✓ It should provide good user interface forms.
- ✓ The following functionality of COHORT should be met
 - Project Management
 - Staff Management
- ✓ Application deployment should be platform independent.

Feasibility Analysis:

Feasibility analysis determines whether a proposed solution is possible within the boundaries that are determined by the constraints. A solution is considered to be feasible if all the requirements can be satisfied within the constraints of time, resources and technology. Feasibility analysis conducted for this project was as follows:

Economic Feasibility

Is it cost effective to implement this project? The cost of implementing this project is weighed against the benefits you get of the project. It is economically feasible to implement this project because

- ✓ There is no need for any changes in the existing system.
- ✓ There is no need for new investments to complete the project.
- ✓ Automated tasks save a lot of time of Project Management personnel.

Technical Feasibility – Do we have the technical expertise?

This relates to the availability of both the technology as well as qualified personnel to implement this project and use it. It is technically feasible to implement this project because

- ✓ The tools required for this are available.

- ✓ The users of this application are professionals in the field relating to Java Server Pages and Oracle technologies.

User Characteristics:

This project is being developed as a web project. There is a need for the user to understand this application, i.e., get used to this application it doesn't take much time for the user to get used to the interface and the application. The interface has been developed carefully so that the user doesn't get confused with the functionalities it provides. If the user is familiar with Java Server Pages then he will get accustomed to this application maximum by the first two uses. The user is expected to be familiar with Java Server Pages.

Merits of the Proposed System:

This system is being developed to fulfill all the requirements specified above. The merits of this system can be stated as follows:

- Reduction of time in retrieving the Data.
- Cost reduction and Cost control
- Maintenance of Projects, Employees, Clients record.
- User friendly entry screens

Life Cycle Model:

For any project to be implemented successfully, it is considerably important that you choose the right kind of life cycle. The

life cycle you choose for the project, guides you through the process of development of the project. The project can some times become a success even when the chosen life cycle doesn't suit it. But this happens very rarely. The life cycle you choose controls all the activities of the project. It defines (and controls) the situations and the order in which various activities are to be performed. Of course, every project is feasible when provided with unlimited resources.

The Prototype model: – This model emphasizes on the sources of product change requests. A prototype is a model of the actual software being developed. This is created to demonstrate the features of the software and get the response of the client. This helps the software engineer in getting a better understanding of the user's requirements.

The reasons for developing a prototype could be many. The major factor being, to display the various input screens and formats of reports and outputs to the client. This is done to get the opinion and permission of the client to proceed with the development. This is some times also used to check if what we are trying to do can be technically improved. The other reason for doing this is because of the difficulties in following phased life cycle model.

The phased life cycle model is used only when it is possible to get a complete set of specifications at the very beginning of the project. The phased life cycle model is suitable in situations where the development team already has some experience in developing similar type of software.

The life cycle model chosen for this project is the prototype model. The reasons for this were those that were specified above. The advantage of using this model can be fully exploited when not all the requirements of the project are completely defined; or otherwise when the requirements of the project are expected to change. This model allows the software to show the various components that have been developed to get the nod of the client and to find out if all the requirements for that component have been fulfilled

Hardware and Software Requirements

This application is expected to run on systems with the following configuration:

Client Requirements:

Processor	Intel Pentium
RAM	64 MB RAM
Hard Disk	20 GB
Operating System	Any Windows OS with MFC support or UNIX.

Web Server Requirements:

Processor	Pentium IV
RAM	1 GB RAM
Application Server	TOMCAT
OS	Windows NT/2000
Database	Microsoft SQL Server2000

SYSTEM DESIGN

Introduction:

To design is to conceive and plan an outline of the final product and decide how to achieve the aim. This stage is concerned with identifying the components of the software, and how the software should be broken up. This stage is also concerned with the identification of data stores, functions, and relationships between the components.

Design is the bridge between customer requirements and the actual implementation of the project. To design is to plan before starting to code an application. The activity that any programmer takes up before writing the code is to think, how to solve the problem. The solution for the problem is first thought out, and then if any alternatives exist, they are explored. The application code is then written for the best solution possible for that situation.

Software design can be classified as follows:

- ✓ External Design

- ✓ Internal Design
 - Architectural Design
 - Detailed Design

External Design:

The external design is concerned with how to present the software to the user. It deals with the looks of the software i.e., the user interfaces, reports, output formats, data storage formats etc.

Internal Design:

The internal design is concerned with the internal structure and internal processing details of the project. It defines clearly, how many components the project will be made up of and the techniques that will be used to create them. In this stage, the test plan is created and all the design decisions are recorded, including the reasons why some alternatives (if any) have been discarded or chosen. Internal design deals with the algorithms required, and give the vague decisions and data structures, a concrete shape. The interconnections between the functions and the data structures are clearly specified.

- Architectural design is concerned with defining the components, the interconnections, the data structures, and the data stores. In this stage, concrete decisions are not made so as to make it possible to change design specifications (if necessary).
- On the other hand, detailed design takes care of “how” to create the components, i.e., the code units, the data structures, and the interconnections. Detailed design is very closely related to implementation. The programming language that is going to be used definitely influences the detailed design process, but not to an extent where it becomes difficult to revert back and change the programming language or make some other design changes. The outputs of the detailed design include the details of the algorithms and data structures.

Some of the fundamental concepts of design include the following:

Abstraction:

It is a tool that allows us to deal with concepts without any considerations for how it will be developed. This tool allows us to postpone the structural and

algorithmic considerations. This reduces much of the complexity involved in the design of a component.

Information hiding:

The approach used to hide the complex internal structures of the functions and make them available to the outer world through well-defined interfaces is that of information hiding. This is used effectively where design decisions are hard to make and/or changes to design decisions are expected.

Modularity:

A module is a collection of code and / or data. Modules are well-defined and manageable units with well-defined interfaces among the units. Modularity enhances the clarity of the design, which in turn helps in implementation and testing stages.

Concurrency:

Software systems can be classified in two ways:

Sequential – Where only one portion of the system is active at a time.

Concurrent – Where several independent processes can be active at a Time.

Along with the advantages that follow with concurrency, there are some problems too. Deadlock, mutual exclusion and synchronization of processes are some of the problems associated with it.

System Overview

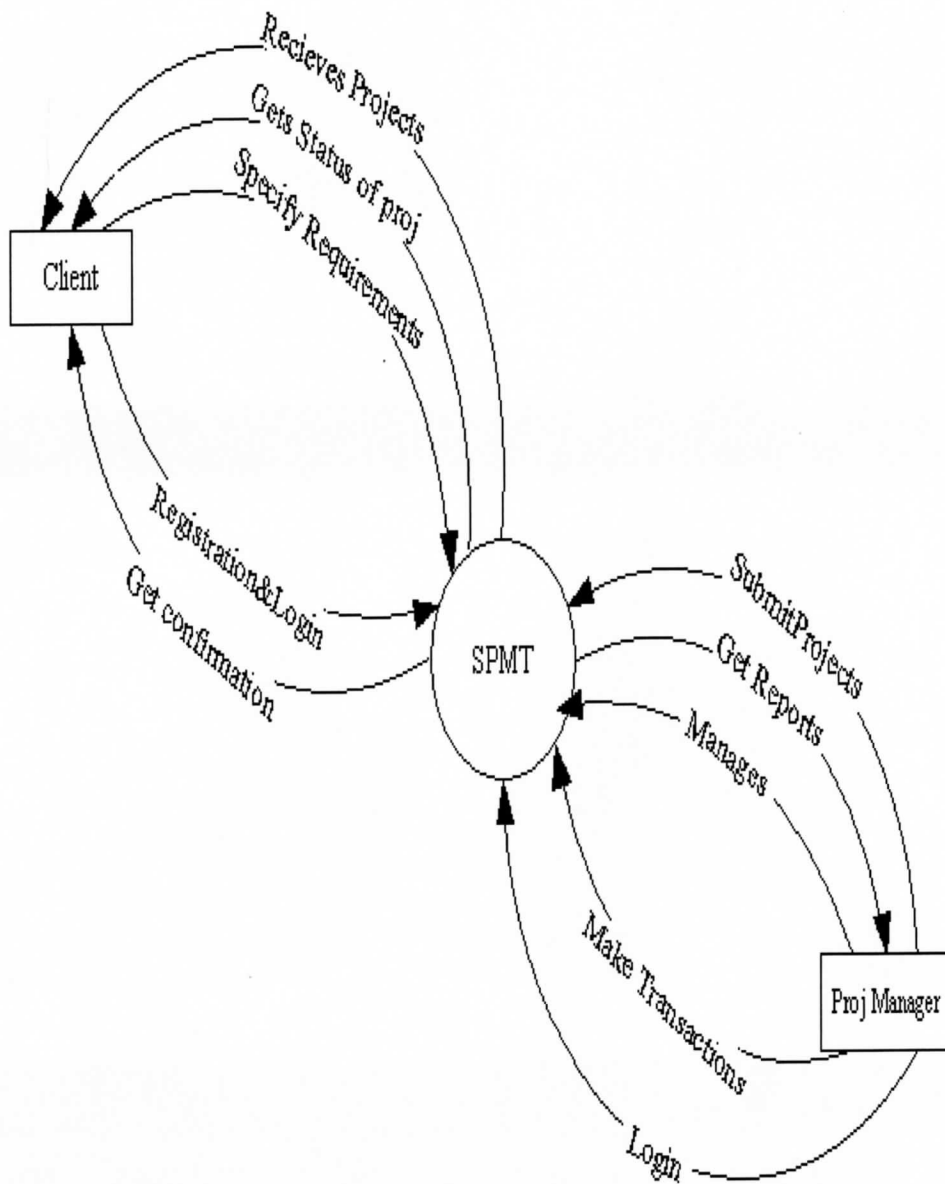
Software organizations undertake huge number of projects, there is a reflective need to control projects and resources better. Organization has to manage the projects effectively and efficient way and all the projects are completed on time.

Project management software gives you the ability to meet many of those demanding project needs. Project management software can make projects more efficient, cost effective and help meet project deadlines. In addition, project management software can streamline project processes and help you utilize resources to achieve the project goals.

Project management software is software that allows us to monitor and manage the progress throughout the life cycle of a project. Project software comes in handy when trying to control large projects. In addition, we can expect to improve team productivity, optimize communications between all team members, identifying problems, and scale to any size projects.

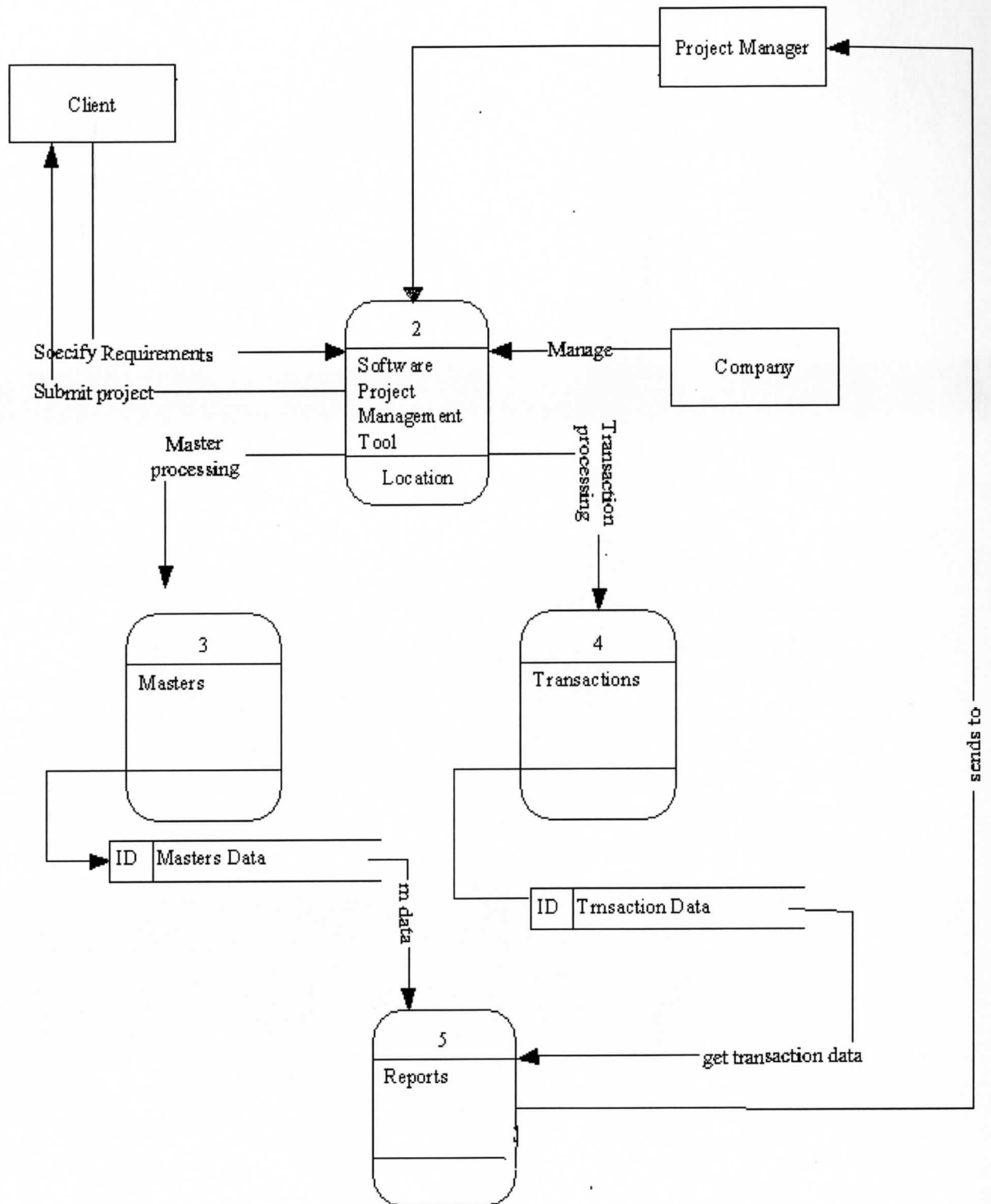
DATA FLOW DIAGRAMS

LEVEL ZERO:



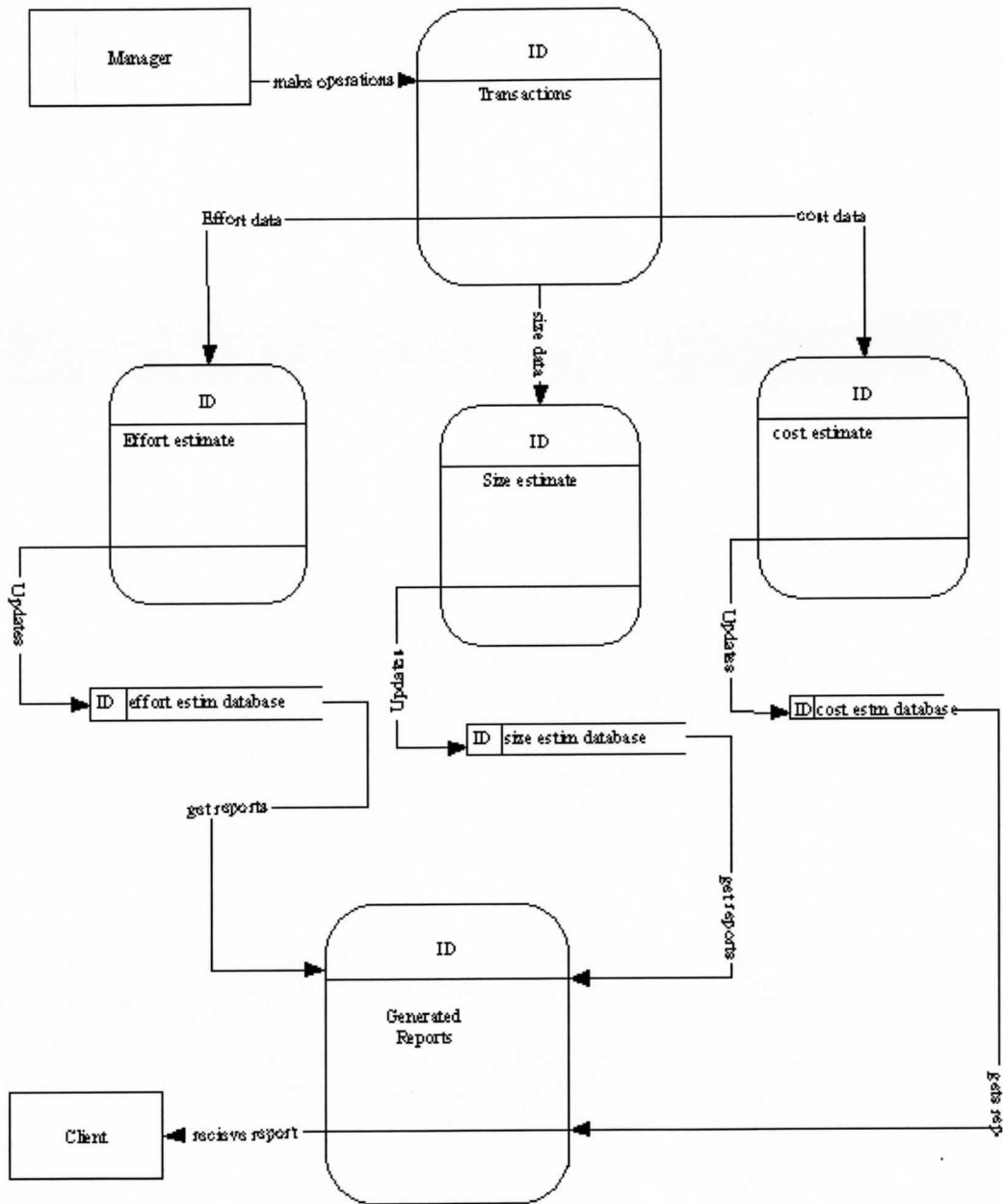
LEVEL1:

Level 1



LEVEL2:

Transaction Module level 2



UNIFIED MODELLING LANGUAGE

An Overview of UML

UML is a Language for

- Visualizing
- Specifying
- Constructing
- Documenting

These are the artifacts of software – intensive system.

A conceptual model of UML:

The three major elements of UML are

The UML's basics building blocks.

The rules that dictate how those building blocks may be put together.

Some common mechanisms that apply throughout the UML.

Basic building blocks of UML

- Things
- Relationships
- Diagrams

Things are the abstractions that are first-class citizens in a model;

Relationships tie these things together;

Diagrams group the interesting collection of things.

Things in UML: There are four kinds of things in UML

1. Structural things.
2. Behavioral things
3. Grouping things
4. Annotational things

These things are the basic object oriented building blocks of UML. They are used to write well-formed models.

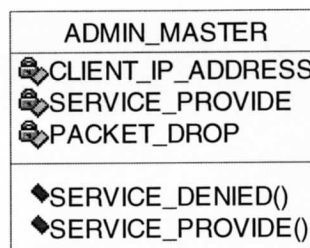
STRUCTURAL THINGS :

Structural things are the nouns of UML models. These are mostly static parts of the model, representing elements that are either conceptual or physical. In all, there are seven kinds of Structural things.

Class:

A class is a description of a set of objects that share the same attributes, operations, relationships, and semantics. A class implements one or more interfaces.
” A class refers to a plain class, class utility, parameterized class or utility, or instantiated class or utility”.

Graphically a class is rendered as rectangle, usually including its name, attributes and operations as shown below.



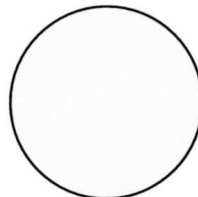
Interface

An interface is a collection of operations that specify a service of class or component. An interface describes the externally visible behavior of that element. Graphically the interface is rendered as circle together with its name.

“A declaration of a collection of operations that may be used for defining a service offered by an instance of a class.” An interface typically specifies only a limited part of the behavior of a class or component.

An interface typically specifies only a limited part of the behavior of a class or component.

Collaboration



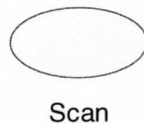
Collaboration defines an interaction and is a society of roles and other elements that work together to provide some cooperative behavior that's bigger than the sum of all the elements.

Graphically, collaboration is rendered as an ellipse with dashed lines, usually including only its name as shown below.



Use Case

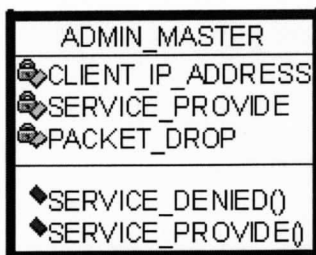
Use case is a description of a set of sequence of actions that a system performs that yields an observable result of value to a particular things in a model. Graphically, Use Case is rendered as an ellipse with dashed lines, usually including only its name as shown below.



Active Class

An Active class is a class whose objects own one or more processes or threads and therefore can initiate control activity.

Graphically, an active class is rendered just like a class, but with heavy lines usually including its name, attributes and operations as shown below.



Component

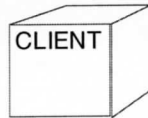
Component is a physical and replaceable part of a system that conforms to and provides the realizations of a set of interfaces.

Graphically, a component is rendered as rectangle with tabs, usually including only its name as shown below

Node

A Node is a physical element that exists at run time and represents a computational resource, generally having at least some memory and often , processing capability.

Graphically, anode is rendered as a cube, usually including only its name, as shown below.



BEHAVIORAL THINGS

Behavioral things are the dynamic parts of UML models. These are the verbs of a model, representing behavior over time and space.

Interaction

An interaction is a behavior that comprises a set of messages exchanged among a set of objects within a particular contest to accomplish a specific purpose.

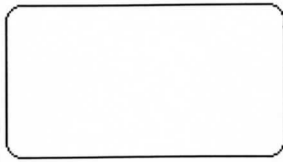
Graphically, a message is rendered as a direct line, almost always including the name of the operation, as shown below.



State Machine

A State machine is a behavior that specifies the sequence of states an object or an interaction goes through during its lifetime on response to events, together with its responses to those events.

Graphically, a state is rendered as a rounded rectangle usually including its name and its sub-state, if any, as shown below.

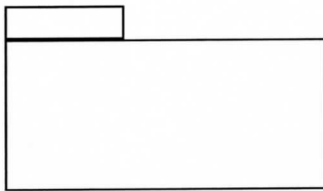


GROUPING THINGS

Grouping things are the organizational parts of the UML models. These are the boxes into which a model can be decomposed.

Package

- A package is a general purpose mechanism for organizing elements into groups.



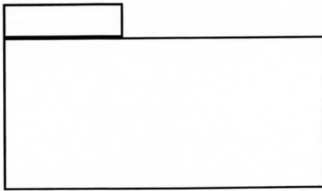
ANNOTATIONAL THINGS

Annotational things are the explanatory parts of the UML models.

Note:

A note is simply a symbol for rendering constraints and comments attached to an element or a collection of elements.

Graphically, a note is rendered as a rectangle with dog-eared corners together, with textual or graphical comment, as shown below.



RELATIONSHIPS IN THE UML

There are four kinds of relationships in UML

1. Dependency
2. Association
3. Generalization
4. Realization

CLASS DIAGRAMS:

Class diagrams are the most common diagrams found in modeling object-oriented systems. A class diagram shows a set of classes, interfaces, and collaboration and their relationships. Graphically a class diagram is a collection of vertices and arcs.

Contents

Class diagrams commonly contain the following things:

- Classes
- Interfaces
- Collaborations
- Dependency, Generalization and association relationships.

USE CASES:

Use case diagrams are one of the five diagrams in the UML for modeling the dynamic aspects of systems (Activity diagrams, Sequence diagrams, State chart diagrams and Collaboration diagrams are the four other kinds of diagrams in the UML for modeling the dynamic aspects of systems). Use Case diagrams are central to modeling the behavior of the system, a sub-system or a class. Each one shows a set of use cases and actors and relationships.

Common properties:

A Use Case diagram is just a special kind of diagram and shares the same common properties, as do all other diagrams – a name and graphical contents that are a projection into the model. What distinguishes a use case diagram from all other kinds of diagrams is its particular content.

Contents:

Use Case diagrams commonly contain:

Use Cases

Actors

Dependency, generalization and association relationships.

Like all other diagrams, use case diagrams may contain notes and constraints. Use Case diagrams may also contain packages, which are used to group elements of your model into larger chunks. Occasionally, you will want to place instances of use cases in your diagrams, as well, especially when you want to visualize a specific system.

INTERACTION DIAGRAMS:

An interaction diagram shows an interaction, consisting of a set of objects and their relationships, including the messages that may be dispatched among them. Interaction diagrams are used for modeling the dynamic aspects of the system. A Sequence diagram is an interaction diagram that emphasizes the time ordering of the messages. Graphically, a sequence diagram is a table that shows objects arranged along the X-axis and messages, ordered in increasing time, along the Y-axis.

Contents:

Interaction diagram commonly contains:

- Objects
- Links
- Messages

Like all other diagrams, interaction diagrams may contain notes and constraints.

SEQUENCE DIAGRAMS:

A Sequence diagram is an interaction diagram that emphasizes the time ordering of the messages. Graphically, a Sequence diagram is a table that shows objects arranged along the X-axis and messages, ordered in increasing time, along the Y-axis.

Typically you place the object that initiates the interaction at the left, and increasingly more sub-routine objects to the right. Next you place the messages that these objects send and receive along the Y-axis, in order of increasing time from top to bottom. This gives the reader a clear visual cue to the flow of control over time.

Sequence diagrams have two interesting features:

1. There is the object lifeline. An object lifeline is the vertical dashed line that represents the existence of an object over a period of time. Most objects that appear in the interaction diagrams will be in existence for the duration of the interactions, so these objects are all aligned at the top of the diagram, with their lifelines drawn from the top of the diagram to the bottom.
2. There is a focus of control. The focus of control is tall, thin rectangle that shows the period of time during which an object is performing an action, either directly or through the subordinate procedure. The top of the rectangle aligns with the action; the bottom is aligned with its completion.

ACTIVITY DIAGRAM:

An Activity Diagram is essentially a flow chart showing flow of control from activity to activity. They are used to model the dynamic aspects of a system. They can also be used to model the flow of an object as it moves from state to state at different points in the flow of control.

An Activity is an ongoing non-atomic execution within a state machine. Activities ultimately result in some action, which is made up of executable atomic computations that result in a change in state of the system or the return of a value.

Contents

Activity diagrams commonly contain:

- Activity states and action states

- Transitions
- Objects

Like all other diagrams Activity diagrams may contain notes and constraints.

STATECHART DIAGRAMS:

A State chart diagram shows a state machine, emphasizing the flow of control from state to state. A state machine is a behavior that specifies the sequence of states an object goes through during its lifetime in response to events, together with its responses to those events.

A state is a condition or situation in the life of an object during which it satisfies some condition, performs some activity, or waits for some event. An event is the specification of a significant occurrence that has allocation in time and space. Graphically, a state chart diagram is a collection of vertices and arcs.

Contents

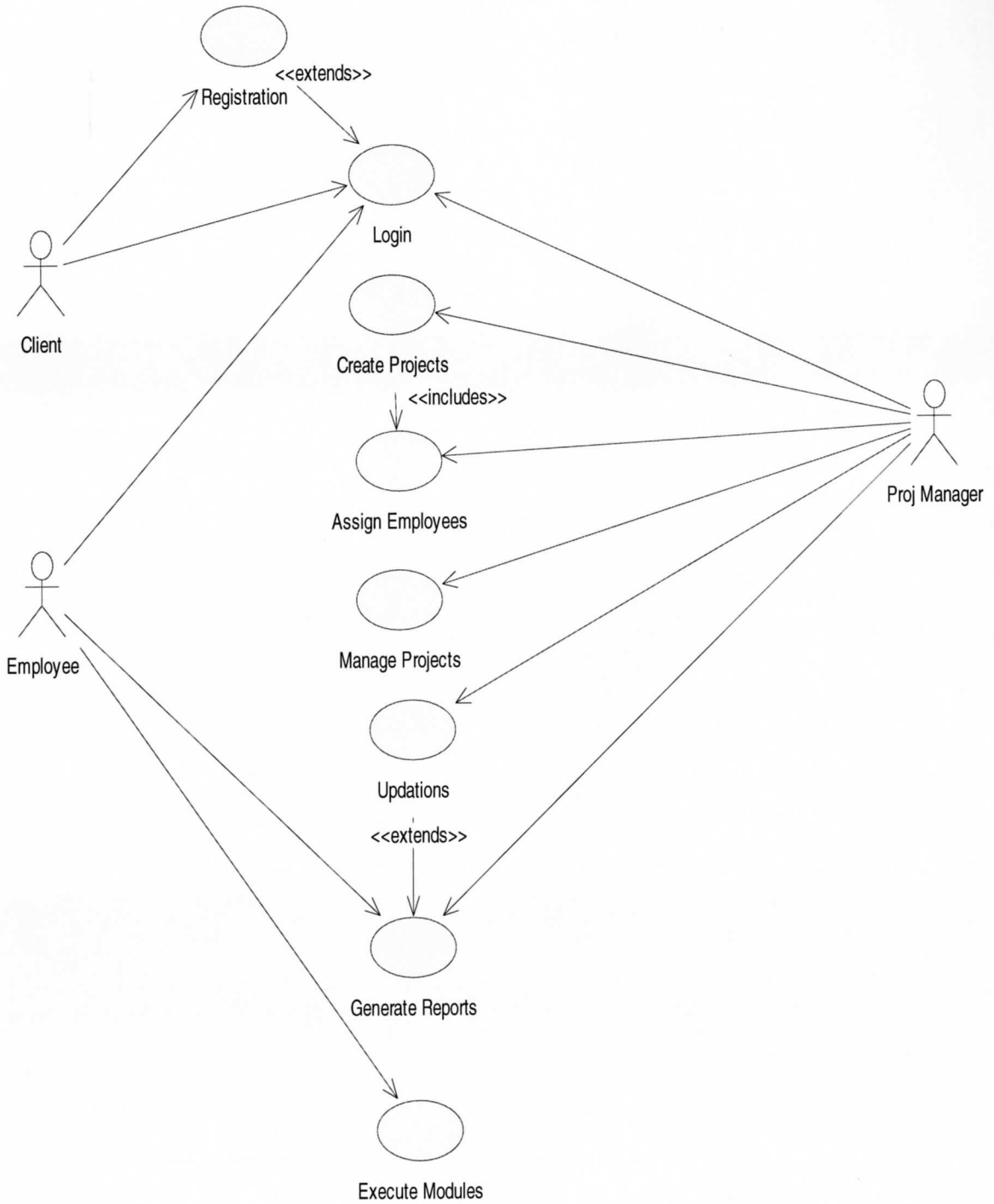
State chart diagrams commonly contain

- Simple states and Composite states
- Transitions, including events and actions

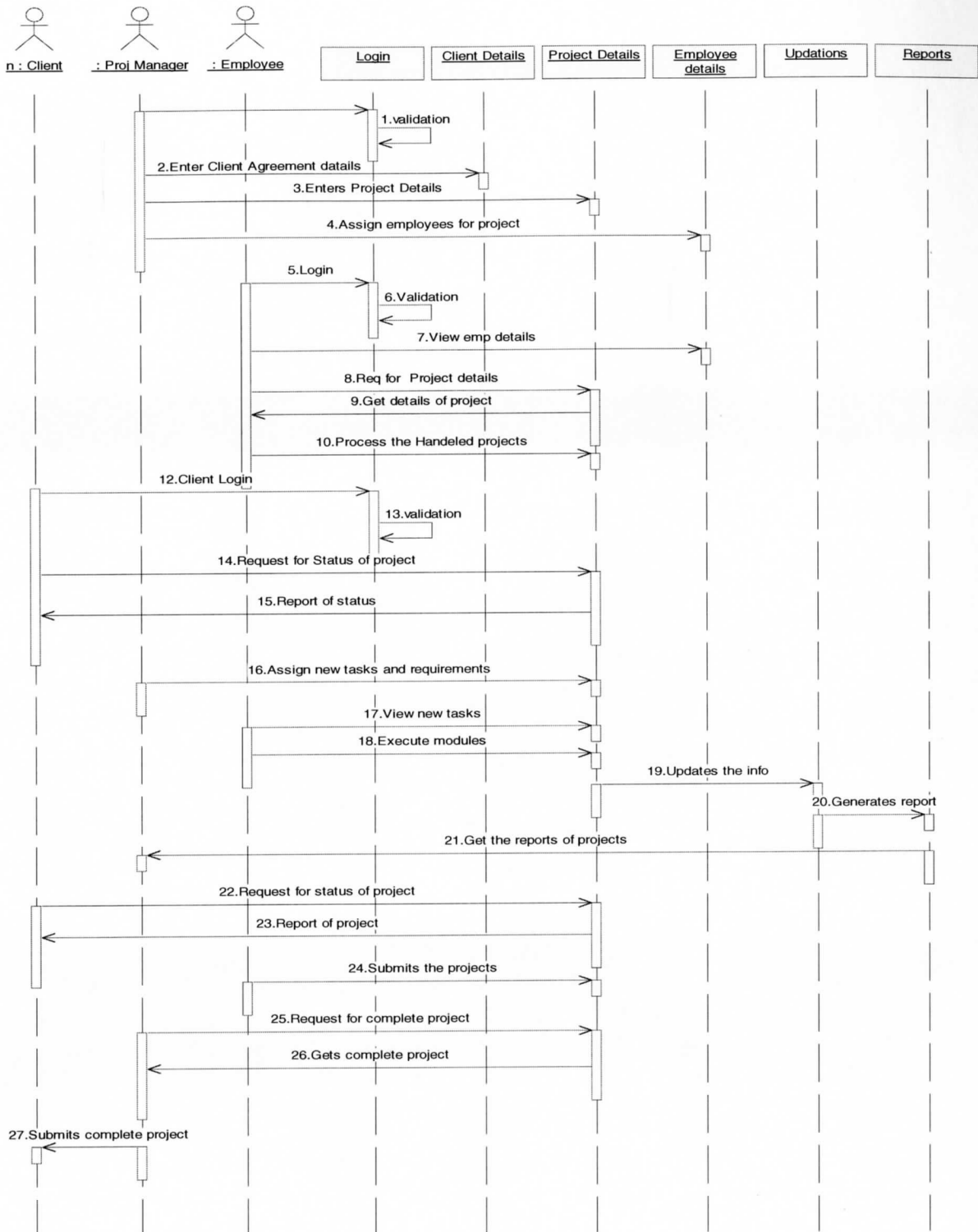
Like all other diagrams, state chart diagrams may contain notes and constraints.

UML DIAGRAMS

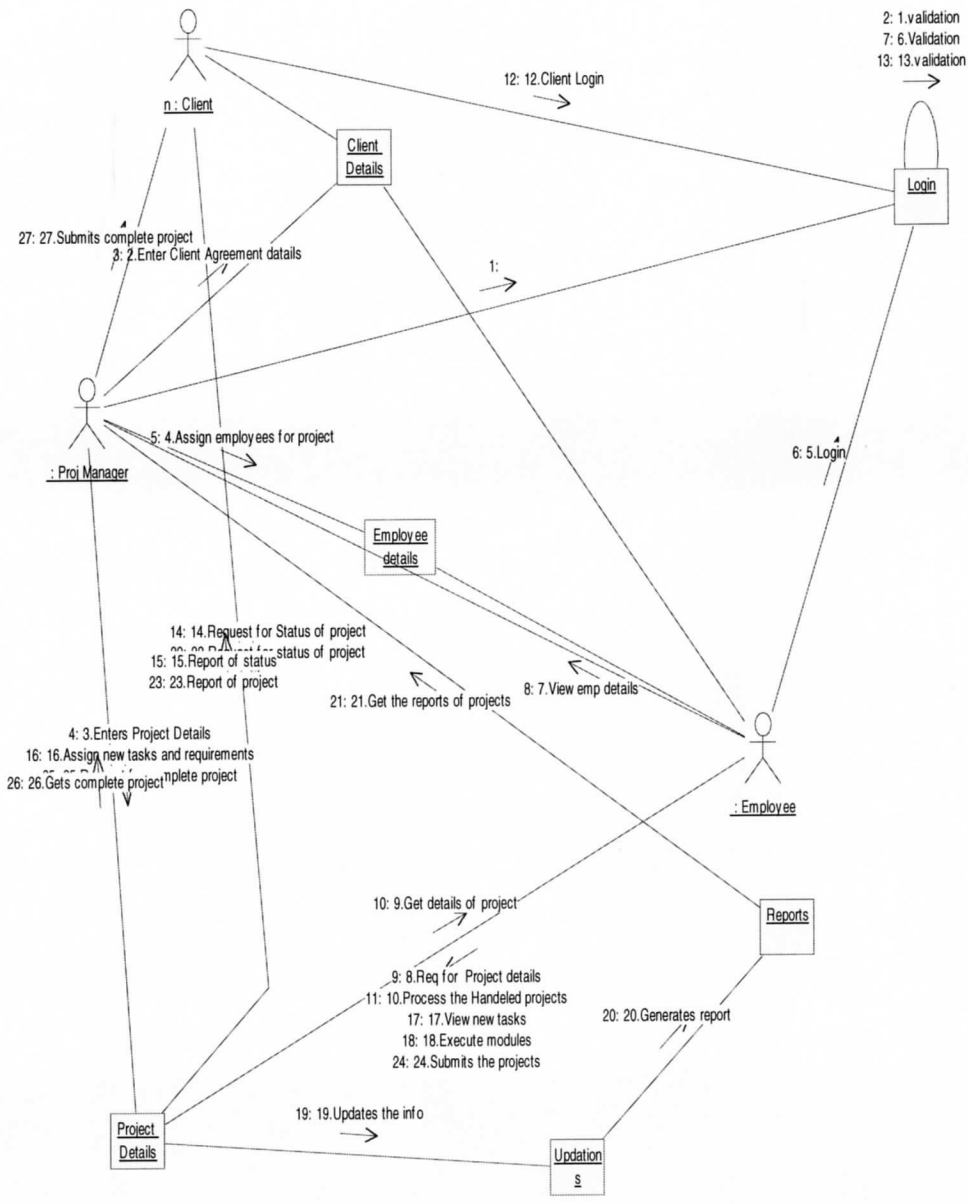
USECASE DIAGRAM:



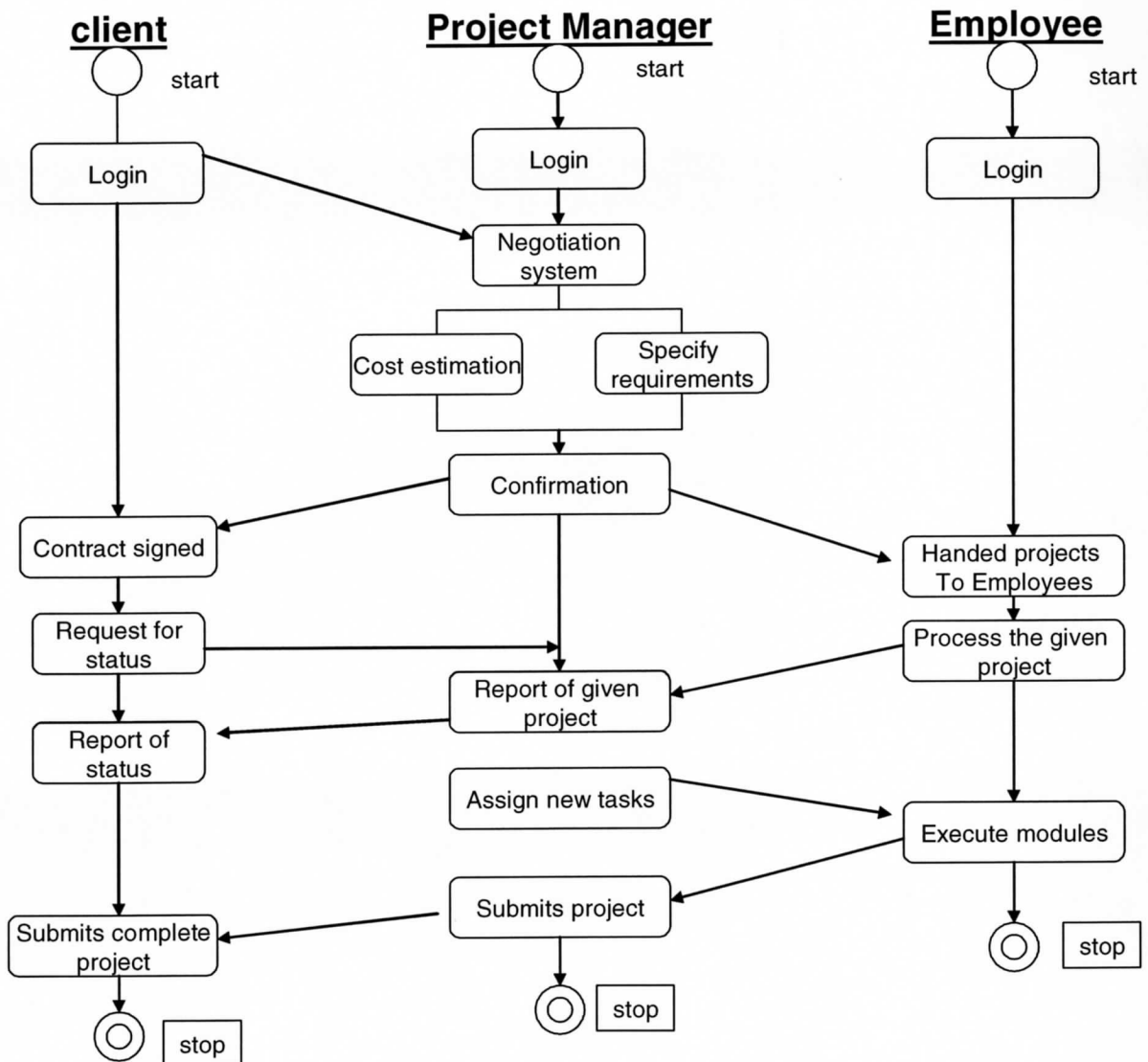
SEQUENCE DIAGRAM:



COLLABORATION DIAGRAM:



ACTIVITY DIAGRAM:



Introduction to Database Management System

(DBMS):

A Database is an integrated collection of user related data stored with minimum redundancy, serves many users/application quickly and efficiently.

A database system is basically a computerized record keeping system, i.e. it is a computerized system whose overall purpose is to maintain information and make that information available on demand.

DBMS is a collection of inter-related data and set of programs that allows several users to access and manipulate data. Its main purpose is to provide users with and with an abstract view of the data, i.e. the system hides certain details of how the data is stored and maintained.

Database Management System is dividing into 4 main components

- Database
- Hardware
- Software
- User

Database: It consists of collection of persistent data that is used by the application system.

Hardware: The processors and associated main memory that are used to support the execution of database system software.

Software: The layer between the physical database and the users that handles all requests from the user for access to the database.

User: There are three types of users

- Application Programmers
- End User
- Database Administrator(DBA)

TYPES OF DBMS:

There are four major categories of DBMS data models.

- Hierarchical
- Network
- Inverted
- Relational

RELATIONAL DATABASE MANAGEMENT SYSTEMS:

Database Management System has evolved from hierarchical to network to relational models. Today, the most widely accepted database model is the relational model. The relational database management system uses only its relational model. The relational database management system uses only its relational capabilities to manage the information stored in the database. The relational model has three different aspects.

- Structures
- Operation
- Integrity rules

Structures:

They are well-defined objects that store the data of a database structure and the data contained within them can be manipulated by operations.

Operations:

They are clearly defined actions that allow users to manipulate the data and structures of a database. The operations on a database must adhere to a predefined set of integrity rules.

Integrity rules:

They are laws that govern which operations are allowed on the data and structures of a database. Integrity rules protect the data and the structures of a database.

A DBMS can be regarded as relational only if it obeys a set of 12 rules formulated by Dr.E.F.Codd. These rules define the scope and functionality of a relational database. There is a single overall rule called “Zero Rule” which foundation rule that covers all other rules. It states that

“Any truly relational database must be manageable entirely through its relational capabilities”.

TWELVE RULES ARE AS FOLLOWS:

- Rule 1: The information rules
- Rule 2: The guaranteed access rule
- Rule 3: The systematic treatment of null values
- Rule 4: The database description rule.
- Rule 5: The comprehensive sub-language rule.
- Rule 6: The view update rule.
- Rule 7: The insert, update and delete rule
- Rule 8: The physical data independence rule
- Rule 9: The logical data independence rule
- Rule 10: The integrity independence rule.
- Rule 11: The distribution independence rule.
- Rule 12: The no subversion rules.

ADVANTAGES OF RDBMS:

- Data redundancy can be reduced
- Data Independence
- Ensure access and integrity
- Sharing data
- Recovery from failure
- Uniform security and privacy through password, view and automatic backup.

DISADVANTAGES OF RDBMS:

- Cost of software /hardware
- Problems associated with centralization
- Complexity of backup and recovery

COMPARISION OF DBMS AND RDBMS:

SNO	DBMS	RDBMS
1	The relation between two files has to be maintained programmatically, if required.	Relation can be specified While creating structure itself.
2	Client/Server architecture not normally supported	Most RDBMS support Client/ Server architecture.
3	Distributed database is not Supported.	Supports distributed databases.

4	Least Security: anyone can Access records.	Multiple level of security maintained Login security Command level security Object level security.
5	Doesn't follow Codd's Rules.	Follows many of the Codd's rules.

STRUCTURED QUERY LANGUAGE:

SQL stands for Structured Query Language which is used to communicate with relational database, which are in turn a set of related information stored in the form of tables.

SQL is a non-procedural language because it processes sets of records rather than just one data at a time and also provides automatic navigation to the data. Here one can manipulate a set or rows rather than one at a time. SQL commands accept a set or rows rather than one at a time. SQL commands accept a sets or rows as input and return sets as outputs. The set property of SQL allows the results of one SQL statement to be used as input to another. Here one need not specify the access method of the data.

SQL is utilized as the communication language with the database among the database users like database administrators, Systems administrators, and security administrators and applications programmers.

SQL provides commands for a variety of tasks including:

Querying data:

- Inserting, updating, and deleting rows in an object.
- Creating, replacing, altering and dropping objects.
- Controlling access to the database and its objects.
- Guaranteeing database consistency and integrity.

The SQL is subdivided according to their functions as follows:

- Data Definition Language (DDL)

Or

- Schema Definition Language (DDL)
- Data Manipulation Language (DML)
- Data Control Language (DCL)

SQL*PLUS is a program or a tool available for working with an ORACLE database. It allows the user to:

- Create tables in a database
- Stores information in the tables
- Changes information in the tables
- Retrieve the information in the form we choose, performing calculations on it and combining it and combining it in new ways.
- Maintain the data itself.

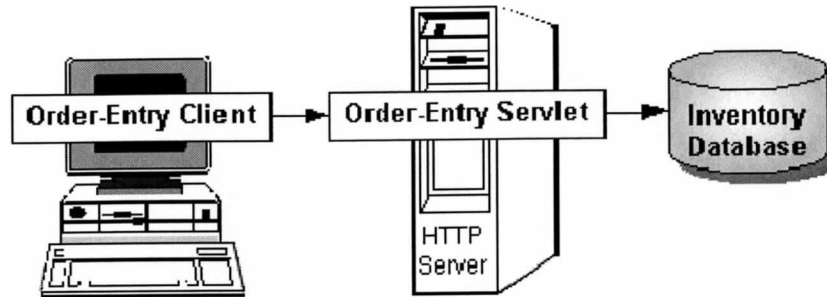
Even though the architecture of RDBMS is same but features are different from different RDBMS. Oracle, Sybase, Informix. As per the features concerned all the RDBMS can be related as it. But with the easiness, demand it is better to go for Oracle.

Because the professional availability in Oracle is high compared to other RDBMS professionals. Because the availability is high the man power cost will be less compared to Sybase professionals, Informix professionals. Even the support from the vendors is also high in Oracle environment rather than other RDBMS.

Servlets

Servlets are modules that extend request/response-oriented servers, such as Java-enabled web servers. For example, a servlet might be responsible for taking data in an

HTML order-entry form and applying the business logic used to update a company's order database.



Servlets are to servers what applets are to browsers. Unlike applets, however, servlets have no graphical user interface.

Servlets can be embedded in many different servers because the servlet API, which you use to write servlets, assumes nothing about the server's environment or protocol. Servlets have become most widely used within HTTP servers; many web servers support the Servlet API.

The ServletRequest Interface

The `ServletRequest` interface allows the servlet access to:

- Information such as the names of the parameters passed in by the client, the protocol (scheme) being used by the client, and the names of the remote host that made the request and the server that received it.
- The input stream, `ServletInputStream`. Servlets use the input stream to get data from clients that use application protocols such as the HTTP POST and PUT methods.

Interfaces that extend `ServletRequest` interface allow the servlet to retrieve more protocol-specific data. For example, the `HttpServletRequest` interface contains methods for accessing HTTP-specific header information.

The ServletResponse Interface

The `ServletResponse` interface gives the servlet methods for replying to the client. It:

- Allows the servlet to set the content length and MIME type of the reply.
- Provides an output stream, `ServletOutputStream`, and a `Writer` through which the servlet can send the reply data.

Interfaces that extend the `ServletResponse` interface give the servlet more protocol-specific capabilities. For example, the `HttpServletResponse` interface contains methods that allow the servlet to manipulate HTTP-specific header information

Java and Internet

Java is strongly associated with Internet and known as Internet programming language. Internet users can use java to create applet programs and run them locally using java enabled browser search as hot java. Applets can be downloaded from remote machine via Internet and run it on local machine.

Java and World Wide Web

World wide web is an open-ended information retrieval system designed for the use in the distributed environment. This system contains web pages that provide both information and controls. We can navigate to a new web page in any direction. This is made possible with HTML. Java was meant to be used in the distributed environment such as Internet. So java could be easily incorporated into the web system and is capable of supporting animation graphics, games and other special effect. The web has become more dynamic and interactive with support of java. We can run a java program on remote machine over Internet with the support of web.

HTML:

Hyper Text Markup Language the language of the web page. World Wide Web better known as www, HTML allows the user to produce those web pages. It includes the text, graphics and pointers to other web pages.

HTML (hyper text markup language) is a language used to create hypertext documents that have hyper links embedded in them. It consists of tags embedded in the text of a document with HTML. We can build web pages or web documents. It is basically a formatting language and not a programming language. The browser reading the document interprets mark up tags to help format the document for subsequent display to a reader. HTML is a language for describing structured documents. HTML is a platform independent. WWW (world wide web) pages are written using HTML. HTML tags control in part the representation of the WWW page when view with web browser. The browser interprets HTML tags in the web document and displays it. Different browsers show data differently. Examples of browsers used to be web pages include:

- Netscape
- Internet Explorer

HTML is not a programming language but it is an application of ISO standard 8879, SGML (Standard Generalized Markup Language), but specialized to Hypertext and adapted to the web. The idea behind Hypertext is that instead of reading text in rigid linear structure it can easily jump from one web page to another web page. We can navigate through the information based on our interest and preference. A mark up language simply a series of elements, each delimited with special characters, that define how text and other items enclosed within the element should be displayed. Hyperlinks are underlined or emphasized words that load to other documents or some portions of the same documents.

HTML can be used to display any type of document on the host computer, which can be geographically at the different location. It is a versatile language and can be used on any platform on desktop.

HTML provides tags (special codes) to make the document look attractive. HTML tags are not case sensitive. Using graphics, fonts, different sizes, color etc. can enhance the presentation of the document. Anything that is not in the tag is part of the document itself.

```
<html>
<head><title>Welcome to Intranet Mailing System</title></head>
<body bgcolor=#40004>
<pre>
<center><h1><font color=white size=+3 face=mscomic></font></center></h1>
<center><h1><font color=white size=+3 face=mscomic>Intranet Mailing
System</font></center></h1>
<center></center>
<center><h1><u><a href="loginsc.html"><font color=yellow>-----Enter into the WebPage-
-----</h1></a></center></font>
</pre>
</body>
</html>
```

Advantages:

- A HTML document is small and hence easy to send over the Internet. It is small because it does not include any type of formatted information.
- HTML is a platform Independent.
- HTML tags are not Case sensitive.
- HTML is good in the part of designing a Web page with graphics, color, font, and other style programming.

JAVA SCRIPTS:

Java Scripts was developed by Netscape Communication Corp, the makers of Netscape Navigator Web browser, in the year 1995.

Java script was the first web programming language to be introduced and it is by far the most popular.

Java script was originally called *Live Script* and renamed as the JavaScript to indicate its relationship with java. JavaScript supports the development of both client and server Components of Web based applications. Even though JavaScript supports both client and server Web programming, we prefer JavaScript at client side programming since most of the browsers supports it.

JavaScript is almost as easy to learn as HTML, and JavaScript statements can be included in HTML documents by enclosing the statement between <SCRIPT>..... </SCRIPT> tags pair.

```
<SCRIPT LANGUAGE=" JavaScript">
```

```
JavaScript statements </SCRIPT>
```

In JavaScript we can:

- Add scrolling or changing messages to the browsers status line.
- Validate the contents of the form and make calculations.
- Animate images or rotate images that change when we move the mouse over them.
- Detect the browser in use and display different content for different browsers.
- Detect installed pug-ins and notify the user if a plug-in is required.

Advantages:

- JavaScript can be used for both Client and Server side scripting.
- JavaScript is the default scripting language at client side since all the browsers supports it.
- It is more flexible than VBscripts.

JDBC:

Java Database Connectivity is better known as the JDBC.

The JDBC provides developer with a way to connect to relational data from within java code. Using the JDBC developers can create a Client which can be anything from an applet to an EJB that can be connected to a Database, executes Structured Query Language (SQL) statements, and process the result of those statements.

JDBC generalizes the most common database access functions by abstracting the vendor-specific details of a particular database. This result is a set of classes and interfaces, placed in the "java.sql." package, which can be used with any database that has an appropriate JDBC Driver. This allows JDBC connectivity to be provided in a consistent way for any Database.

It is important while working with JDBC to know about its drivers. JDBC has four different Drivers. They are as Follows:

Type-1:

JDBC-ODBC Bridge Driver.

- It is used only in the stand alone application and non portable i.e. used only in one machine
- It does not support the Internet and Intranet database application access.
- It provides the bridge between the JDBC API and ODBC API. The bridge translates the standard JDBC calls to corresponding ODBC calls, and send them to ODBC data source via ODBC libraries.
- Driver name: "sun.jdbc.odbc.JdbcOdbcDriver"

Type-2:

Part Java, Part Native Driver

- It has the same properties as type-1.
- Implementations are available in .lib files (library).
- This is platform Dependent.
- Type -2 drivers are more efficient than the type-1 drivers.
- Driver name: "oracle.jdbc.Driver.OracleDriver"

Type-3:

Intermediate Database Access Server

- Type-3 drivers use an intermediate (Middleware) database server that has the ability to connect multiple java clients to multiple database servers.
- It is used for the Intranet access but not Internet access.
- It is platform Independent.
- To use this driver Web server is necessary one.
- Driver name: "oracle.jdbc.Driver.OracleDriver"

Type-4:

Pure Java Drivers

- Type-4 drivers are pure java alternate to Type-2 drivers.
- It uses the thin layer of Network.
- It passes the query through network and can be used for internet access.
- It has an access to Internet.
- Driver name:"oracle.jdbc.Driver.OracleDriver"

A list of available all JDBC drivers, is available
"http://industry.java.sun.com/products/jdbc/drivers/."

Interfaces of JDBC:

- **Connection**
- **Statement**
- **Result Set**
- **Prepared Statement**
- **Callable Statement**
- **Batch Update**

Connection:

- Connection is to establish the connection with the Data base.
- If the connection is established then the get connection return connection object, if error occurs it throws SQL exception.
- To establish connection we write:
- `Connection=DriverManager.getConnection("url", id,pwd);`

Statement:

This interface lets u execute SQL statements over the underline connection and access the results.

Result set:

Result set is an interface used to execute the query or SQL Statement in the program and return will be stores in the result set object.

Prepared Statements:

JDBC prepared statement address the following requirements:

- Creating parameterized statement such that data for parameters can be substituted dynamically
- Creating statements involving data values that can not always be represented as character strings.

- Precompiling SQL statements to avoid repeated compiling of the same SQL statements.

Callable Statements:

- It is used to call the Store Procedures in the java and used for calling precompiled SQL statements.
- **Java.sql.CallableStatement** interface can be used for the creating statement for executing stored procedures.

Batch Update:

A part from the Scrollable Result set the JDBC API supports the batch update. This feature allows the multiple update statements like Insert, Delete, or Update in a single request to the database. Batching large numbers of statement can result in significant performance gains.

Steps To Perform Data Base Interactions:

- Load the Driver
- Establish Connection with the Database
- Perform Transactions
- Close the Connection with the Database.
- Exit

JSP

JSP better known as “Java Server Pages”.

JSP is the new kind of Server side technique to develop dynamic web pages in the Website. JSP is the technology that provides server side dynamic web pages. Generally Dynamic web pages means a web pages which will change its contents differently for different users or a web page after display may be changing its content, format of display style of display because of events generated by the user interaction.

JSP can be designed and developed less like the programs and more like web pages. JSP pages are ideal when we need to display markup with the embedded dynamic content.

JSP can use java beans with the specified *scope* or *tag extension* to achieve a clean separation of static content and the java code that produce dynamic web application. The JSP page and dependent files are together known as the translation unit. The first time JSP engine intercepts a requests for a JSP, it compiles the translation unit into a servlet.

It is a two stage process. First the JSP source code is converted into the Servlet and this Servlet is then compiled. The reference implementation stores both the servlet source and compiled code.

The first time a JSP is loaded by the JSP container also called as the JSP engine, the servlet code necessary to implement the JSP tags is automatically generated, compiled, and loaded into the servlet container.

At the first stage the browser request .jsp file the request goes to JSP engine. The file .jsp in JSP engine will be converted into .java file and will be sent it to Servlet engine for the execution. After Execution the .class file will be sent into the browser.

JSP Directives:

- Output Directives: for getting the output

<%=.....%>

- Declarative Directives: for Declaring Variables

<%!.....%>

- Scriptlets: to write java codes

<%.....%>

- Page url: to import java API

<%@.....%>

- JSP comment: to write comments

<%!.....%>

JSP ACTION TAGS:

<jsp: plugin> : used to call applets in Jsp
<jsp: include> : to include the Jsp page or an html page
<jsp: forward> : to forward the Jsp page or an html page
<jsp: usebean> : to pass the values to bean class
<jsp: setProperty>: .jsp usebean for processing
<jsp: getProperty>: jsp usebean for the processing.

Variables declared in one Scriptlet can not be access in the other scriptlet so variable in jsp must be declared only in declarative directives. Objects are used in Scriptlet only.

Advantages of JSP:

- JSP will be compiled by Web Server automatically.
- Html and java code can be separated but they work together
- JSP execution is faster than the Servlets.
- JSP is very easy to code when compare to Servlets.
- JSP provides Model Architecture
- All the objects are inbuilt object.

In JSP for the exceptions we use “error page”. Error page can be once written and saved and then it can be called.

CLASSES USED:

Package javax.servlet

Interface Summary	
<u>RequestDispatcher</u>	Defines an object that receives requests from the client and sends them to any resource (such as a servlet, HTML file, or JSP file) on the server.
<u>Servlet</u>	Defines methods that all servlets must implement.
<u>ServletConfig</u>	A servlet configuration object used by a servlet container to pass information to a servlet during initialization.
<u>ServletContext</u>	Defines a set of methods that a servlet uses to communicate with its servlet container, for example, to get the MIME type of a file,

	dispatch requests, or write to a log file.
<u>ServletContextAttributeListener</u>	Implementations of this interface receive notifications of changes to the attribute list on the servlet context of a web application.
<u>ServletContextListener</u>	Implementations of this interface receive notifications about changes to the servlet context of the web application they are part of.
<u>ServletRequest</u>	Defines an object to provide client request information to a servlet.
<u>ServletRequestAttributeListener</u>	A ServletRequestAttributeListener can be implemented by the developer interested in being notified of request attribute changes.
<u>ServletRequestListener</u>	A ServletRequestListener can be implemented by the developer interested in being notified of requests coming in and out of scope in a web component.
<u>ServletResponse</u>	Defines an object to assist a servlet in sending a response to the client.

Class Summary

<u>GenericServlet</u>	Defines a generic, protocol-independent servlet.
<u>ServletContextAttributeEvent</u>	This is the event class for notifications about changes to the attributes of the servlet context of a web application.
<u>ServletContextEvent</u>	This is the event class for notifications about changes to the servlet context of a web application.
<u>ServletInputStream</u>	Provides an input stream for reading binary data from a

	client request, including an efficient <code>readLine</code> method for reading data one line at a time.
<u>ServletOutputStream</u>	Provides an output stream for sending binary data to the client.
<u>ServletRequestAttributeEvent</u>	This is the event class for notifications of changes to the attributes of the servlet request in an application.
<u>ServletRequestEvent</u>	Events of this kind indicate lifecycle events for a <code>ServletRequest</code> .

Exception Summary

<u>ServletException</u>	Defines a general exception a servlet can throw when it encounters difficulty.
<u>UnavailableException</u>	Defines an exception that a servlet or filter throws to indicate that it is permanently or temporarily unavailable.

Package javax.servlet.http

Interface Summary

<u>HttpServletRequest</u>	Extends the <code>ServletRequest</code> interface to provide request information for HTTP servlets.
<u>HttpServletResponse</u>	Extends the <code>ServletResponse</code> interface to provide HTTP-specific functionality in sending a response.

<u>HttpSession</u>	Provides a way to identify a user across more than one page request or visit to a Web site and to store information about that user.
<u>HttpSessionActivationListener</u>	Objects that are bound to a session may listen to container events notifying them that sessions will be passivated and that session will be activated.
<u>HttpSessionAttributeListener</u>	This listener interface can be implemented in order to get notifications of changes to the attribute lists of sessions within this web application.
<u>HttpSessionBindingListener</u>	Causes an object to be notified when it is bound to or unbound from a session.
<u>HttpSessionContext</u>	Deprecated. <i>As of Java(tm) Servlet API 2.1 for security reasons, with no replacement.</i>
<u>HttpSessionListener</u>	Implementations of this interface are notified of changes to the list of active sessions in a web application.

Class Summary

<u>Cookie</u>	Creates a cookie, a small amount of information sent by a servlet to a Web browser, saved by the browser, and later sent back to the server.
<u>HttpServlet</u>	Provides an abstract class to be subclassed to create an HTTP servlet suitable for a Web site.
<u>HttpSessionBindingEvent</u>	Events of this type are either sent to an object that implements <u>HttpSessionBindingListener</u> when it is bound or unbound from a session, or to a <u>HttpSessionAttributeListener</u> that has been

	configured in the deployment descriptor when any attribute is bound, unbound or replaced in a session.
<u>HttpSessionEvent</u>	This is the class representing event notifications for changes to sessions within a web application.

My SQL Introduction:

Versions of MySQL and the mm.mysql JDBC driver when have been reported to work:

- MySQL 3.23.47, MySQL 3.23.47 using InnoDB, MySQL 4.0.1alpha
- mm.mysql 2.0.14 (JDBC Driver)

Please let us know if you have tested the new MySQL mm.mysql 3.0 driver.

1. MySQL configuration

Ensure that you follow these instructions as variations can cause problems.

Create a new test user, a new database and a single test table. Your MySQL user **must** have a password assigned. The driver will fail if you try to connect with an empty password.

```
mysql> GRANT ALL PRIVILEGES ON *.* TO javauser@localhost
-> IDENTIFIED BY 'javadude' WITH GRANT OPTION;
mysql> create database javatest;
mysql> use javatest;
mysql> create table testdata (
-> id int not null auto_increment primary key,
-> foo varchar(25),
-> bar int);
```

Note: the above user should be removed once testing is complete!

Next insert some test data into the testdata table.

```
mysql> insert into testdata values(null, 'hello', 12345);
Query OK, 1 row affected (0.00 sec)
```

```
mysql> select * from testdata;
+-----+-----+
| ID | FOO   | BAR   |
+-----+-----+
|  1 | hello | 12345 |
+-----+-----+
1 row in set (0.00 sec)

mysql>
```

2. server.xml configuration

Configure the JNDI DataSource in Tomcat by adding a declaration for your resource to `$CATALINA_HOME/conf/server.xml`.

Add this in between the `</Context>` tag of the examples context and the `</Host>` tag closing the localhost definition.

```
<Context path="/DBTest" docBase="DBTest"
  debug="5" reloadable="true" crossContext="true">

  <Logger className="org.apache.catalina.logger.FileLogger"
    prefix="localhost_DBTest_log." suffix=".txt"
    timestamp="true"/>

  <Resource name="jdbc/TestDB"
    auth="Container"
    type="javax.sql.DataSource"/>

  <ResourceParams name="jdbc/TestDB">
    <parameter>
      <name>factory</name>
      <value>org.apache.commons.dbcp.BasicDataSourceFactory</value>
```

```
</parameter>

<!-- Maximum number of dB connections in pool. Make sure you
      configure your mysqld max_connections large enough to handle
      all of your db connections. Set to 0 for no limit.
      -->
<parameter>
  <name>maxActive</name>
  <value>100</value>
</parameter>

<!-- Maximum number of idle dB connections to retain in pool.
      Set to 0 for no limit.
      -->
<parameter>
  <name>maxIdle</name>
  <value>30</value>
</parameter>

<!-- Maximum time to wait for a dB connection to become available
      in ms, in this example 10 seconds. An Exception is thrown if
      this timeout is exceeded. Set to -1 to wait indefinitely.
      -->
<parameter>
  <name>maxWait</name>
  <value>10000</value>
</parameter>

<!-- MySQL dB username and password for dB connections -->
<parameter>
  <name>username</name>
  <value>javauser</value>
</parameter>
<parameter>
  <name>password</name>
  <value>javadude</value>
</parameter>
```

```

<!-- Class name for mm.mysql JDBC driver -->
<parameter>
  <name>driverClassName</name>
  <value>org.gjt.mm.mysql.Driver</value>
</parameter>

<!-- The JDBC connection url for connecting to your MySQL dB.
The autoReconnect=true argument to the url makes sure that
the
mm.mysql JDBC Driver will automatically reconnect if mysqld
closed the
connection. mysqld by default closes idle connections after
8 hours.
-->
<parameter>
  <name>url</name>
<value>jdbc:mysql://localhost:3306/javatest?autoReconnect=true</value>
</parameter>
</ResourceParams>
</Context>

```

3. web.xml configuration

Now create a WEB-INF/web.xml for this test application.

```

<?xml version="1.0" encoding="ISO-8859-1"?>
  <!DOCTYPE web-app PUBLIC
  "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
  "http://java.sun.com/dtd/web-app_2_3.dtd">

```

```
<web-app>
  <description>MySQL Test App</description>
  <resource-ref>
    <description>DB Connection</description>
    <res-ref-name>jdbc/TestDB</res-ref-name>
    <res-type>javax.sql.DataSource</res-type>
    <res-auth>Container</res-auth>
  </resource-ref>
</web-app>
```


TESTING

Testing Techniques:

Testing is a process in which we create a series of test cases that are intended to “demolish”. The software that has been built. In fact testing is the one step in the software engineering process that could be viewed as destructive rather than constructive

A good test case is one that has a high probability of finding as-yet undiscovered error.

White box Testing:

This is performed knowing an internal workings of a product, tests are conducted to ensure that “all gears mesh,” i.e. that internal operation performs according to specification and all internal component have been adequately exercise.

Using the white box test we can derive test cases:

- Guarantee that all independent paths with in a module have been exercised at least once.
- Exercised all logical decisions on their true and false sides.
- Executes all loop at their boundaries and with in their operational bounds and exercise internal data structures to assure their validity.

Black Box Testing :

Knowing the specified function that a product has been designed to perform test can be conducted that demonstrate each function is fully operational.

It attempts to find errors in the following:

- Incorrect or missing functions
- Interface errors
- Errors in the data structure or external database access.
- Performance errors.
- Initialization and terminal errors.

The different test strategies used are:

- **Unit testing:** It focuses verification effort on the smallest unit of the system design module.
- **Integration testing:** it is a systematic technique for construction the programs structure while conduction test too uncover errors associate with interface.
- **Validation testing:** Validation succeeds when system functions in a manner that can be a reasonable expected by end user. There are two tests for system validation they are *alpha & beta testing*.
- **System testing:** This testing is actually a series of different test whose primary purpose to fully exercise the computer basis system.

SYSTEM TESTING AND MAINTAINANCE:

There are three steps involved with the system testing and maintenance they are as follows:

- Testing with sample data
- Testing with Actual data
- System Maintenance

DATA DICTIONARY:

FEATURES OF DATA DICTIONARV

The Volume of data in most Information system applications is substantially more than a single user can easily keep track of data dictionaries are an integral component of structural analysis, since data flow diagrams by themselves do not fully describe the subject of investigation. The data dictionary provides additional information about the system.

What is Data Dictionary?

A Data Dictionary a repository of the elements in a system. As the name suggests, these elements center on data and the way they are structured to meet user requirements and organization needs. In a data dictionary we will find a list of ali elements composing the data flowing through a system. The major elements are data flows, data stores and process the Data Dictionary stores details and descriptions of these elements.

Why is a data dictionary important?

Analysts use data dictionaries for five important reasons.

1. To manage the detail in large systems.
2. To communicate a common meaning for ali elements.
3. To document the features of the system.
4. To facilitate analysts of details in order to evaluate characteristics and determine where system changes would be made.
5. To locate errors and omissions in the system.

Manage Detail

Large systems have huge volumes of data flowing them in the form of documents, reports and even. Similarly many different activities take place that use existing data or create new details. The best organized or most effective analysis use automated data dictionaries designed specifically for systems analysis and design.

Communicate Meaning

Data dictionaries assist in ensuring common meanings for system elements and activities. Data dictionaries record additional details about the data flow in a system; so that the persons who are involved may look up the description of data flows, data stores and processes.

Document System Features

Documenting the features of an information system is the third reason for using the data dictionary systems. Features include the parts or components and the characteristics that distinguish each. Having to formally describe, system features will produce a more complete understanding.

Facilitate Analysis

The Fourth reason for using data dictionaries is to determine whether new features are needed in a system or whether change of any type is in order. For any situation, system analysts will typically focus on the following system characteristics.

- a) Nature of transaction
- b) Inquiries
- c) Output and Report generation

CLIENT LOGIN DATABASE: this table gives the information about the client, employee, manager of username, and password.

FIELD NAME	DATATYPE	SIZE
CLIENTID	Integer	10
NAME	Varchar	20
ADDRESS	Varchar	200
CONTACT_PERSON	Varchar	20
PHNO	Integer	10
LOGIN_NAME	Varchar	20
PASSWORD	Varchar	20

PROJECT DATABASE: This table gives total information about the projects which are undertaken by the organization.

FIELD NAME	DATA TYPE	SIZE
PROJECT ID	Integer	10
PNAME	Varchar	20
START_DATE	DATE	
TENTATIVE_END_DATE	DATE	
COMPLEXITY	Varchar	10
END_DATE	DATE	

COMPLEXITY	Varchar	10
MODULE_COUNT	Integer	10
SPECS	Varchar	50
ESTIMATED_MAN_HOURS	Integer	30
STATEID	Integer	30
CLIENTID	Integer	10
PROJECTMANAGER_ID	Integer	10

EMPLOYEE DATABASE: This table gives the information about Employees in the organization.

FIELD NAME	DATA TYPE	SIZE
EMP_ID	Varchar	10
NAME	Varchar	20
JOB	Varchar	
HIREDATE	DATE	20
QUALIFICATION	Varchar	30
E_MAIL ID	Varchar	30
ADDRESS	Varchar	40
PH_NO	Integer	16

LOGIN_NAME	Varchar	40
PASSWORD	Varchar	10

PROJECT EMPLOYEE: This table gives the information about the Employees of a particular Project.

FIELD NAME	DATA TYPE	SIZE
PROJECTID	Varchar	20
EMPLOYEEID	Varchar	10
FROM_DATE	DATE	
TO_DATE	DATE	

PROJECT TECHNOLOGIES: table gives the information about technologies used by the Project.

FIELD NAME	DATA TYPE	SIZE
PID	Integer	20
TECHID	Integer	20

TECHNOLOGIES: This table gives the information about the existing technologies.

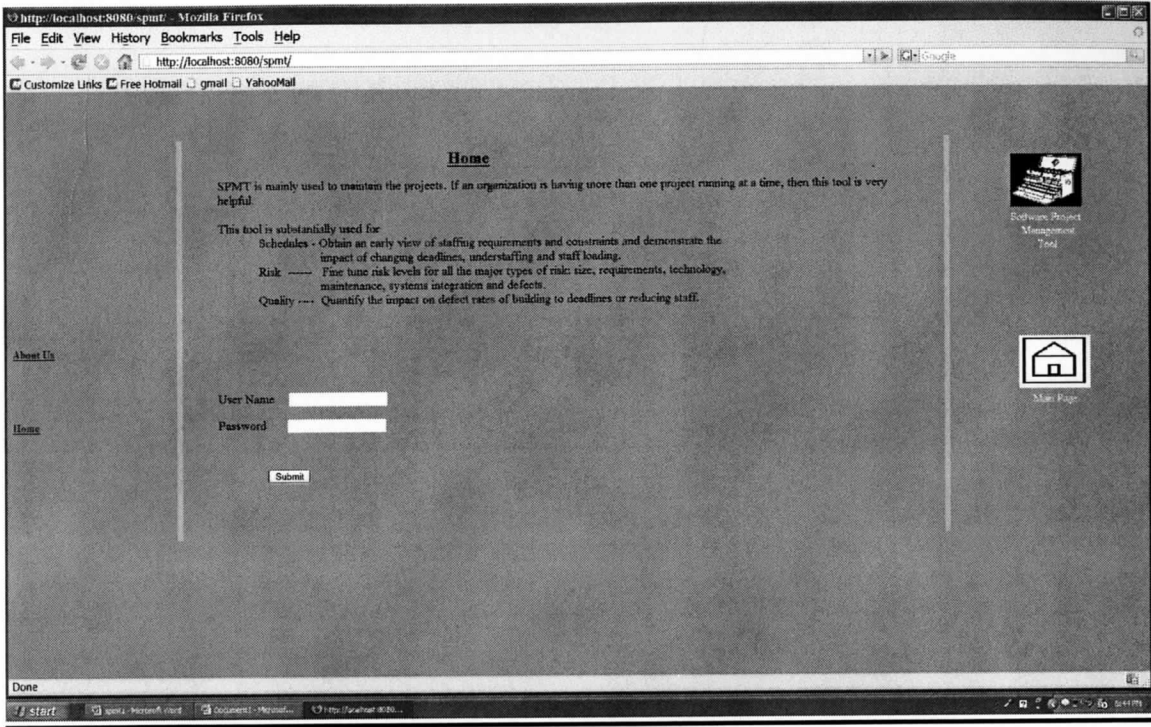
FIELD NAME	DATA TYPE	SIZE
TECHID	Integer	20
TNAME	Varchar	20

PROJECT STATUS:

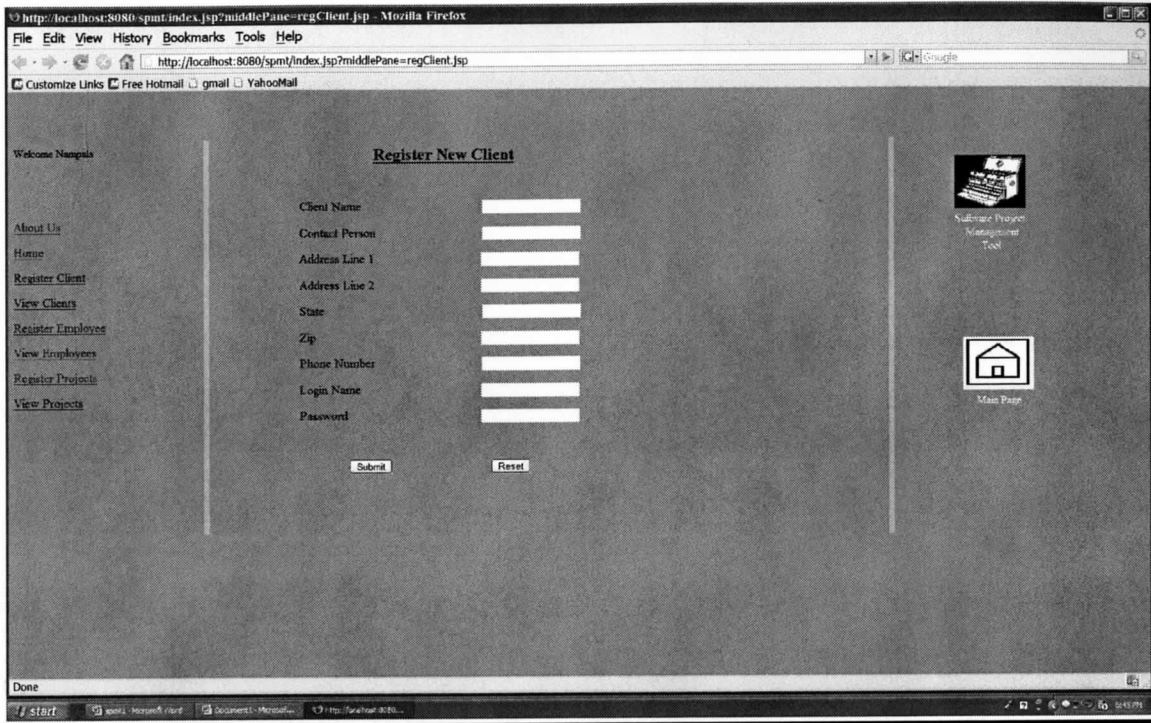
FIELD NAME	DATA TYPE	SIZE
STATEID	Integer	10
STATE	Varchar	20

Output Screens

LOGIN PAGE:



CLIENT REGISTRATION:



REGISTERED CLIENTS:

The screenshot shows a Mozilla Firefox browser window displaying a web application. The address bar shows the URL: `http://localhost:8080/spmt/index.jsp?middlePane=viewClient.jsp`. The page title is "Registered Clients".

On the left side, there is a navigation menu with the following links: [Welcome Nanipala](#), [About Us](#), [Home](#), [Register Client](#), [View Clients](#), [Register Employee](#), [View Employees](#), [Register Projects](#), and [View Projects](#).

The main content area is titled "Registered Clients" and features a search bar labeled "Select a Client" with the text "Sanyam Computers Services Ltd" entered and a "Go" button. Below the search bar is a table with the following data:

Name	Sellootania XP
Contact Person	Ramulu.P
Address	Princeton, NJ
Phone Number	609-750-9069
Login Name	sronas

On the right side of the page, there are two icons: one for "Sanyam's Project Management Tool" and another for "Main Page".

PROJECT REGISTRATION:

http://localhost:8080/spmt/index.jsp?middlePane=regProject.jsp - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://localhost:8080/spmt/index.jsp?middlePane=regProject.jsp

Customize Links Free Hotmail gmail YahooMail

Register New Project

Welcome Namada

[About Us](#)
[Home](#)
[Register Client](#)
[View Clients](#)
[Register Employee](#)
[View Employees](#)
[Register Projects](#)
[View Projects](#)

[Software Project Management Tool](#)

[Main Page](#)

Project Name

No. of Modules

Project Status

Complexity

Man Hours

Requirements

Technologies

Client Organization

Assign Employees
Sandy -- Programmer
Raman -- Module Lead
Kashore -- Tech Lead

Start Date (YYYY-MM-DD)

End Date (YYYY-MM-DD)

Expected End Date (YYYY-MM-DD)

Done

start | http://localhost:8080

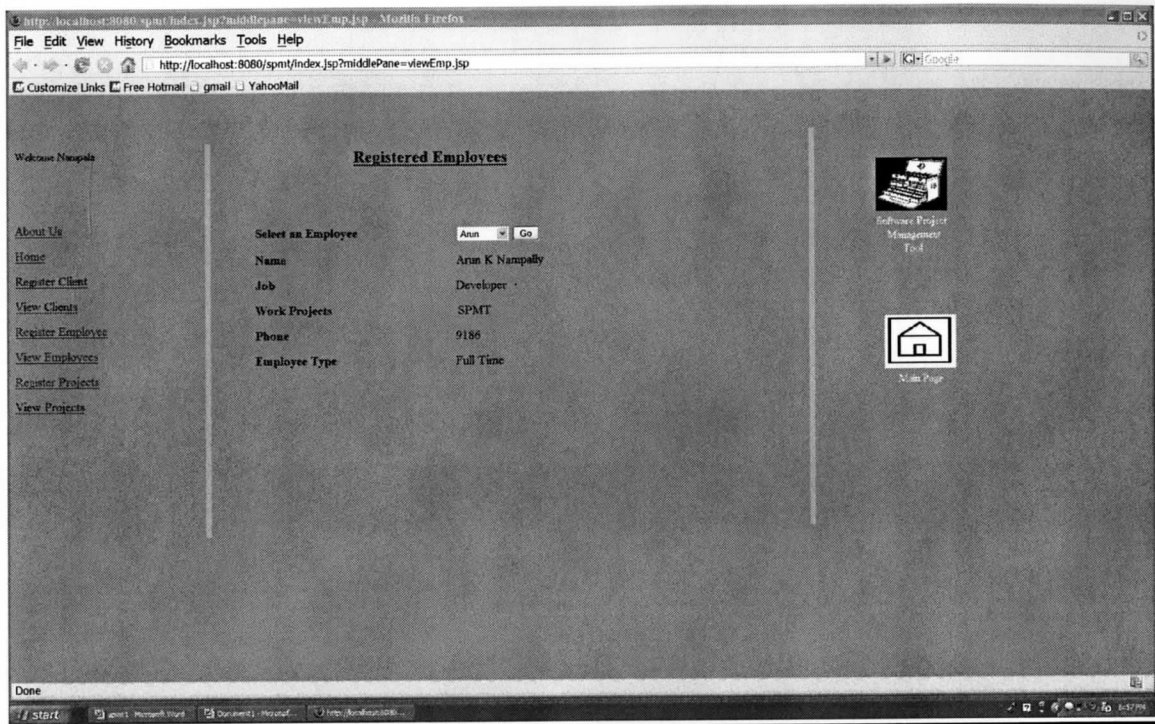
REGISTERED PROJECTS:

The screenshot shows a web browser window with the following content:

- Browser Title:** http://localhost:8080/sfmt/index.jsp?middlePane=viewProject.jsp - Mozilla Firefox
- Browser Address Bar:** http://localhost:8080/sfmt/index.jsp?middlePane=viewProject.jsp
- Page Title:** Registered Projects
- Left Navigation Menu:**
 - Welcome Nampala
 - About Us
 - Home
 - Register Client
 - View Clients
 - Register Employee
 - View Employees
 - Register Projects
 - View Projects
- Project Details Table:**

Project Name	Sellarama XP
No. of Modules	6
Project Status	Current
Complexity	Medium
Man Hours	500
Requirements	Undefined
Technologies	JAVA
Client Organization	Sellarama XP
Assign Employees	Kumar, Vidhya
Start Date	2007-11-26
End Date	2008-02-28
Expected End Date	2008-01-10
- Right Sidebar:**
 - Software Project Management Tool
 - Home Page

REGISTERED EMPLOYEE:



CONCLUSION

The project “**Software Project Management System**” has been designed and implemented using active components and passive components under real time environment.

The project has enabled to simplify the Maintenance of Projects by following certain simple methodologies and techniques as mentioned in the report. This project keeps track of information of each Project, Employee, and Client in the Cohort.

This project also helps the Project administrator to identify the problems and solve them and also plan for future development.

