SOFTWARE TRACKING SYSTEM

A PROJECT REPORT

Submitted by:

STEPHY JOBOY

LLMC17MCA031

to

The APJ Abdul Kalam Technological University

in partial fulfillment of the requirements for the award of the Degree

of

Master of Computer Applications

Department of Computer Applications

LOURDES MATHA COLLEGE OF SCIENCE AND TECHNOLOGY
KUTTICHAL, THIRUVANANTHAPURAM 695574

JULY 2020

SOFTWARE TRACKING SYSTEM

A PROJECT REPORT

Submitted by:

STEPHY JOBOY

LLMC17MCA031

to

The APJ Abdul Kalam Technological University

in partial fulfillment of the requirements for the award of the Degree

of

Master of Computer Applications

Department of Computer Applications

LOURDES MATHA COLLEGE OF SCIENCE AND TECHNOLOGY
KUTTICHAL, THIRUVANANTHAPURAM 695574

JULY 2020

DECLARATION

I undersigned hereby declare that the project report “SOFTWARE TRACKING SYSTEM”, submitted for partial fulfillment of the requirements for the award of degree of Master of Computer Application of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by me under supervision of Ms. Selma Joseph. This submission represents my ideas in my own words and, I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University.

[illegible]

Thiruvananthapuram

06/7/2020 STEPHY JOBOY

LOURDES MATHA COLLEGE OF SCIENCE AND TECHNOLOGY

(Managed By Archdiocese Of Changanacherry)

(Affiliated To APJ Abdul Kalam Technological University, Kerala)

KUTTICHAL, THIRUVANANTHAPURAM-695574

DEPARTMENT OF COMPUTER APPLICATIONS

CERTIFICATE

This is to certify that the project work entitled "SOFTWARE TRACKING SYSTEM" is a bonafide record of the work done by Ms STEPHY JOBOY, Reg No LLMC17MCA031, student of Department Computer of Applications, Lourdes Matha College of Science & Technology, Kuttichal, Thiruvananthapuram, affiliated to APJ Abdul Kalam Technological University, Kerala during the academic year 2019-2020 from January 2019 to July 2020 in partial fulfilment of the requirements for the award of the degree of Master of Computer Applications from APJ Abdul Kalam Technological University, Kerala.

Prof. Selma Joseph
(Internal Supervisor)

Prof. Justin G Russel
(Project Co-Ordinator)

Prof. Selma Joseph
(Head of the Dept.)

TABLE OF CONTENT

ACKNOWLEDGEMENT

At the outset I like to mention that a project report of this magnitude could not have been possible to make without the support, assistance and guidance of some distinguished personalities.

I am greatly indebted to all those who helped and guided me to make this report without which it would not have been possible for me to do this work.

First of all, I like to express my sincere thanks to **Rev.Dr. Tomy Joseph Padinjareveetil**, Director of Lourdes Matha College of Science and Technology and our principal **Prof. Dr. Mohanlal PP** for granting permission to do this project and giving necessary guidance and assistance.

Prof. Selma Joseph who was my internal supervisor for the project has always been at her best to help and guide me to the right path which was a great factor for the successful completion of this project.

I like to extend my sincere gratitude to **Prof. Selma Joseph, Head** Of the department of Computer Application. she was always approachable and gave correct advice to go ahead with the project.

Let me also take this opportunity to extend gratitude to our esteemed institution Lourdes Matha College of Science and Technology.

My parents who held my hand throughout in my endeavour to do this project need a special mention.

I also like to thank my friends who helped me abundantly in the successful completion of this project report.

Any omission in acknowledgement may be pardoned as it is not intentional.

ABSTRACT

Software Tracking System helps to effectively manage the time and resources of the project as per the requirements of the company. It gives a clear notion of how the employee effort is distributed amongst the ongoing projects. It provides the necessary input for financial and HR accounting and keeps a record of each project's total effort. The main users of this are: clients to whom the completed project is to be given, project leader who coordinates all the activities of the project and developers who helps to develop the project.

ADMINISTRATOR

This is one of the important modules of the Software Tracking System. It basically handles administrative task like addition of a new employee, creation of new project, assigning a project to the available employee, modification of employee details and others. This module is available to the administrator project manager levels only.

DEVELOPER

Developer module contains all the activities of employers working in that company.

When a developer log on to the site, he enters to his menu page. This page will have the following links:

a.My profile
b.Leave details
c.Communication
d.My assignments
e.Log out

PROJECT LEADER

Project leader module contains all the activities that the project leader has to do when he is assigned as the project leader for the project.

When a Project Leader log on to the site, he enters to his menu page. Here there will be a link named “My Task As PL”. By selecting this link he will get a page which contains links for all the works he has to do as a project leader.

CLIENT

A client is an external person who registers in the company site for giving some project to the company to do according to his needs. A client can register in the company’s website by selecting the link given in the main page.

CHAPTER 1
INTRODUCTION

1.1 GENERAL BACKGROUND

The main users of Software Tracking Systems are: clients to whom the completed project is to be given, project leader who coordinates all the activities of the project and developers who help to develop the project.

ADMINISTRATOR

This is one of the important modules of the project management system. It basically handles administrative task like addition of a new employee, creation of new project, assigning a project to the available employee, modification of employee details and others. This module is available to the administrator project manager levels only.

When an administrator log on to the site, he enters to his menu page. This page will have the following links:

a. Employee details
b. Leave details
c. Employee registration
d. List project
e. PL selection for project
f. Edit employee profile
g. Notify client
h. Submitted projects
i. Communication
j. Log out

DEVELOPER

Developer module contains all the activities of employers working in that company. All employers are registered as developer at first. According to their skill set some of them are assigned as project leaders. This is done by the administrator. This is done for the registered projects according to the developer’s skill sets and the requirements of the project.

When a developer log on to the site, he enters to his menu page. This page will have the following links:

a. My profile
b. Leave details
c. Communication
d. My assignments
e. Log out

PROJECT LEADER

Project leader module contains all the activities that the project leader has to do when he is assigned as the project leader for the project.

When a Project Leader log on to the site, he enters to his menu page. Here there will be a link named “My Task As PL”. By selecting this link he will get a page which contains links for all the works he has to do as a project leader.

This page will have the following links:

a. Register project
b. Cost calculation

CLIENT

A client is an external person who registers in the company site for giving some project to the company to do according to his needs. A client can register in the company’s website by selecting the link given in the main page.

When a client log on to the website, he enters to his menu page. It will contain the following links:

a. Register project
a. Communication
b. Modification

1.2 OBJECTIVE OF THE PROJECT

Software Tracking System helps to effectively manage the time and resources of the project as per the requirements of the company. It gives a clear notion of how the employee effort is distributed amongst the ongoing projects. It provides the necessary input for financial and HR accounting and keeps a record of each project’s total effort.

CHAPTER 2
LITERATURE SURVEY

2.1 STUDY OF SIMILAR WORKS

2.1.1 EXISTING SYSTEM

1. Currently all the work allocation is done manually.
2. It’s very difficult to know the progress of the project for the client once it is being started.
3. Cost comparison between the estimated cost and the development cost is very difficult.

2.1.2 DRAWBACKS OF EXISTING SYSTEM

1. Cost Calculation is Very difficult.
2. Time Consuming.
3. Work Assignment is very difficult

CHAPTER 3
OVERALL DESCRIPTION

3.1 PROPOSED SYSTEM

1. Simplifies Project Effort tracking for employees, while ensuring accurate, timely and centralized information gathering
2. Offers enhanced reporting for financial controllers with information about the exact costs allocated to each project
3. Intuitive and efficient effort tracking interface for employees
4. Email notifications for missing timesheet submittal
5. Financial controller reporting for project costs

3.2 FEATURES OF PROPOSED SYSTEM

- Faster and easier to use.
- Report can be generating at any time.
- Work Assignment is very easy

3.3 REQUIREMENT SPECIFICATION

A **System Requirements Specification** (abbreviated SRS when need to be distinct from a Software Requirement Specification SRS) is a structured collection of information that embodies the requirements of a system.

A business analyst sometimes titled system analyst. , is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the system development life cycle domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers.

FUNCTIONAL REQUIREMENTS

Any requirement which specifies **what** the system should do. In other words, a functional requirement will describe a particular behavior of function of the system when certain conditions are met.

Typical functional requirements include:

- Business Rules
- Transaction corrections, adjustments and cancellations
- Administrative functions
- Authentication
- Authorization levels
- Audit Tracking
- External Interfaces
- Certification Requirements
- Reporting Requirements
- Historical Data
- Legal or Regulatory Requirements

NON-FUNCTIONAL REQUIREMENTS

Any requirement which specifies **how** the system performs a certain function. In other words, a non-functional requirement will describe how a system should behave and what limits there are on its functionality.

Non-functional requirements generally specify the system's quality attributes or characteristics, for example: "Modified data in a database should be updated for all users accessing it within 2 seconds."

Typical non-functional requirements include:

- Performance – for example: response time, throughput, utilization, static volumetric
- Scalability
- Capacity
- Availability
- Reliability
- Recoverability
- Maintainability
- Security
- Manageability
- Data Integrity

3.4 FEASIBILITY ANALYSIS

The initial investigation points to be question whether the project is feasible. A feasibility study is conducted to identify the best system that meets all the requirements. This gives an identification description, an evaluation of proposed systems and the selection of the best system for the job.

The result of a feasibility study is a formal proposal. The proposal summarizes what is known and what is going to be done.

It consists of following:

1. Statement of the problem.
2. Summary of the finding and recommendations.
3. Details of finding.
4. Recommendations and conclusion.

The feasibility study determines whether the proposed system is flexible or not. One personal computer, a printer a scanner (or a digital camera) is necessary for the proposed system. The present project will improve quality of the image and reduce cost. There are costs incurred from implementing this project as well as benefits. The requirement of the system is specified with a set of constraints such as system objectives and description of the outputs. It is then the duty of the analyst to evaluate the feasibility of the proposed system to generate the above results. The key factors to be considered during the feasibility study are.

3.4.1 TECHNICAL FEASIBILITY

The main consideration is to be given at this juncture is to the study of the available resources of the organizations where the project is to be developed and implemented. Here the system analyst evaluates the technical merits of the system given emphasis on the performance, reliability, maintainability and productivity. The user some needs a printed or some other hardware to be added to the system that will raise the technical complexity. Then the user has to check whether the system will support the addition or not. This involves financial considerations to accommodate technical enhancements.

3.4.2 OPERATIONAL FEASIBILITY

An estimate should be made to determine how much effort and care is needed in the development of the system that also includes the training given to the user. The user does not require any severe training to use the proposed system, because it just automates the existing system. Hence, the user does not have to deviate much from his line of work, when the new system is installed.

3.4.3 ECONOMICAL FEASIBILITY

Economic analysis is the most important and frequently used method for evaluating the effectiveness of the proposed system. It is very essential because the main goal of the proposed system is to have economically better result along with the increased efficiency. The tool used in cost is cost-benefit analysis. It is the comparative study of the cost versus the benefit and savings that are expected from the proposed system. It reduces unnecessary wastage of human hours and expenses.

3.4.4 BEHAVIORAL FEASIBILITY

Behavioral feasibility determines how much effort will go into educating, selling and training the user staff on a candidate system. People are inherently resistant to change and computing has been known to facilitate change. Since the change is user friendly, user training is a very easy matter.

The key steps to be considered during the feasibility study are:

- Prepare system flowchart.
- Enumerate potential candidate system.
- Describe and identify characteristics of candidate system.
- Determine and evaluate performance and cost effectiveness of each candidate system.
- Weight system performance and cost data.
- Select the best candidate system.
- Report project directive to management.

CHAPTER 4
OPERATING ENVIRONMENT

4.1 HARDWARE REQUIREMENTS

Processor:	i3
Processor Speed:	3 GHz
Hard Disk Space:	320 GB
Main Memory :	2 GB
Mouse :	3 Button
Key Board:	104 Keys Multimedia Keyboard

4.2 SOFTWARE REQUIREMENTS

Operating System:	Windows 10
Front End:	Visual Studio .Net 2010
Back End:	Microsoft SQL Express
Web Browser:	Any web browser

4.3 TOOLS AND PLATFORMS

4.3.1 INTRODUCTION TO C#

C# is a multi-paradigm programming language that encompasses functional, imperative, generic, object-oriented (class-based), and component-oriented programming disciplines. It was developed by Microsoft as part of the .NET initiative and later approved as a standard by Emma (ECMA-334) and ISO (ISO/IEC 23270). C# is one of the programming languages designed for the Common Language Infrastructure.

C# is intended to be a simple, modern, general-purpose, object-oriented programming language. Its development team is led by Anders Hejlsberg, the designer of Borland's Object Pascal language. It has an object-oriented syntax based on C++ and is heavily influenced by Java. It was initially named Cool, which stood for "C-like Object Oriented Language." However, in July 2000, when Microsoft made the project public, the name of the programming language was given as C#. The most recent version of the language is 3.0 which were released in conjunction with the .NET Framework 3.5 in 2007. The next proposed version, 4.0, is in development.

Design goals

The Emma standard lists these design goals for C#:

- C# is intended to be a simple, modern, general-purpose, object-oriented programming language.
- Because software robustness, durability and programmer productivity are important, the language should include strong type checking, array bounds checking, and detection of attempts to use uninitialized variables, source code portability, and automatic garbage collection.
- The language is intended for use in developing software components that can take advantage of distributed environments.
- Programmer portability is very important, especially for those programmers already familiar with C and C++.
- Support for internationalization is very important.
- C# is intended to be suitable for writing applications for both hosted and embedded systems, ranging from the very large that use sophisticated operating systems, down to the very small having dedicated functions.
- Although C# applications are intended to be economical with regard to memory and processing power requirements, the language is not intended to compete directly on performance and size with C.

Features

By design, C# is the programming language that most directly reflects the underlying Common Language Infrastructure (CLI). Most of its intrinsic types correspond to value-types implemented by the CLI framework. However, the language specification does not state the code generation requirements of the compiler: that is, it does not state that a C# compiler must target a Common Language Runtime, or generate Common Intermediate Language (CIL), or generate any other specific format. Theoretically, a C# compiler could generate machine code like traditional compilers of C++ or FORTRAN. In practice, all existing compiler implementations target CIL.

Some notable C# distinguishing features are:

- There are no global variables or functions. All methods and members must be declared within classes. Static members of public classes can substitute for global variables and functions.
- Local variables cannot shadow variables of the enclosing block, unlike C and C++. Variable shadowing is often considered confusing by C++ texts.

- C# supports a strict Boolean data type, bolo. Statements that take conditions, such as while and if, require an expression of a Boolean type.
- In C#, memory address pointers can only be used within blocks specifically marked as unsafe, and programs with unsafe code need appropriate permissions to run. Code that is not marked as unsafe can still store and manipulate pointers through the System.IntPtr type, but it cannot dereference them.
- Managed memory cannot be explicitly freed; instead, it is automatically garbage collected. Garbage collection addresses memory leaks by freeing the programmer of responsibility for releasing memory which is no longer needed.
- Multiple inheritance is not supported, although a class can implement any number of interfaces.
- C# is more typesafe than C++.
- Enumeration members are placed in their own scope.
- C# provides properties as syntactic sugar for a common pattern in which a pair of methods, accessor (getter) and mutator (setter) encapsulate operations on a single attribute of a class.
- Full type reflection and discovery is available.

4.3.2 MICROSOFT .NET

Microsoft released the .NET (pronounced dot net)Framework in February 2002.DOT Net is a revolutionary multi language platform that knits various aspects of application development together with the Internet. It is Microsoft’s new strategy for development and deployment of software. The framework covers all layers of software development above the operating system. The .NET initiative is all about enabling data transfer between networks, PCs and devices seamlessly, independent of the platforms, architecture and solutions .NET is Microsoft’s next-generation platform for building web applications and web services. It is a platform for XML web services areas of Microsoft.

.NET Framework

The .NET Framework includes classes, interfaces and value types that help expedite and optimize the development process and give you access to system functionality. The .NET Framework was designed with three goals in mind. First, it was intended to make Windows applications much more reliable, while also providing an application with greater degree of security .Second, it was intended to simplify the development of Web applications and services that not only work in the traditional sense, but on mobile devices as well. Lastly the framework was designed to provide a single set of libraries that would work with multiple languages.
Main parts of the .NET Framework

1. The Common Language Runtime (CLR)
2. The Framework Class Libraries (FCL)

The Common Language Runtime:

One of the design goals of .NET Framework was to unify the runtime engines so that all developers could work with a set of runtime services. The .NET Framework's solution is called the Common Language Runtime (CLR).The CLR provides capabilities such as memory management, security, and robust error handling to any language that work with the .NET Framework. The CLR enables languages to inter operate with one another. Memory can be allocated by code written in one language and can be freed by code written in another language.

A .NET programming language (C#,VB.NET,VC++.NET)does not compile into executable code that can be directly executed on your computer; instead it compiles into an intermediate code called Microsoft Intermediate Language(MSIL).As a programmer one need not worry about the syntax of MSIL since source code is automatically converted to MSIL.

Common type System (CTS)

The Common Type System (CTS) is a component of the CLR and provides a common set of data types, each having a common set of behaviors. CTS types are divided into two categories: reference types and value types. The CTS eliminates many interoperability problems that exist outside .NET. The benefits of common type system also include a singly rooted object hierarchy and type safety.
In particular, the common type system provides the following foundations for the .NET Framework

- CTS provide a first-class, pure object oriented model supported by all programming languages that have adopted the .Net Framework.
- CTS establish the foundations and reference framework for cross language integration, interoperation, type safety, security and high-performance code execution. CTS defines rules that languages must follow, which helps ensure that objects written in different languages can interact with each other.

The CTS itself is defined in the Common Language Specification. The CLS defines a single set of rules for every .NET compiler ensuring that each compiler will output code that interacts with the common language runtime consistently. One of the CLS requirements is that the compiler must support certain types defined in the CTS.

Class Libraries:

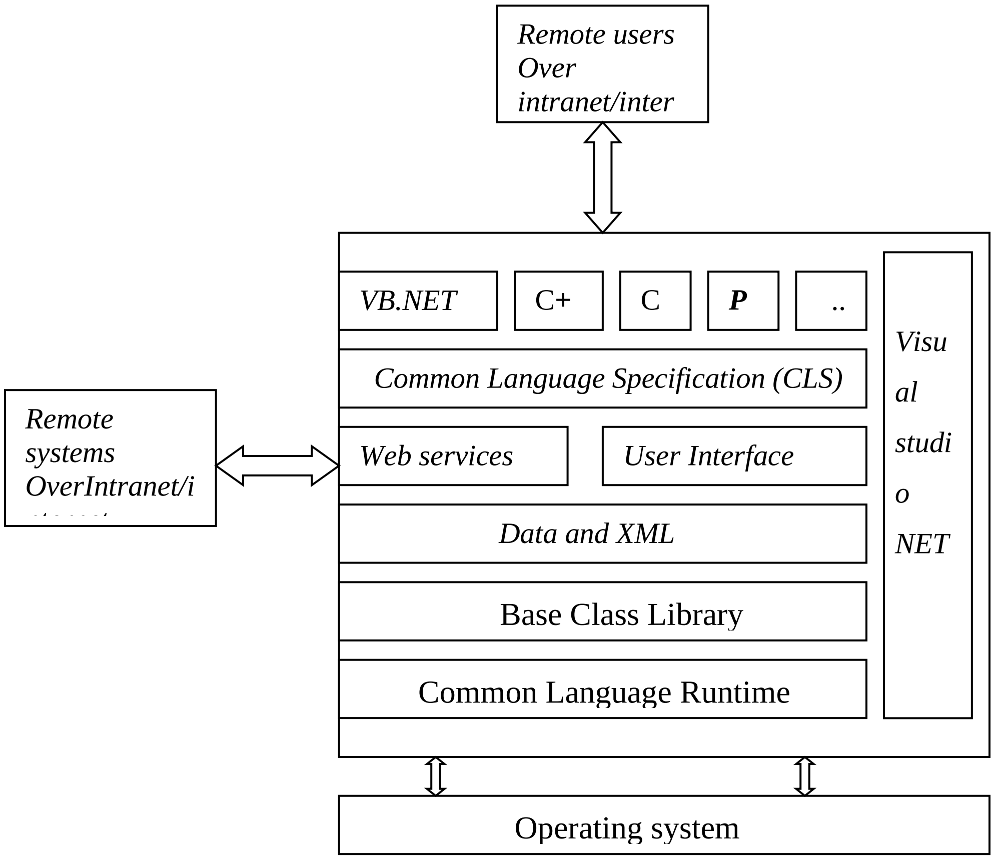
Class Libraries**,** the other main component of the .NET framework, is a comprehensive, object-oriented collection of reusable type that you can use to develop applications ranging from traditional command-line or Graphical User Interface (GUI) applications to applications based on the latest innovations provided by ASP.NET such as Web Forms and XML Web Services.

The .Net framework can be hosted by unmanaged components that load the CLR into their processes and initiate the execution of managed code, there by creating a software environment that can exploit both managed and unmanaged features. The .Net framework not only provides several runtime hosts, but also supports the development of third-party runtime hosts. We can use the .Net framework to develop the following types of application and services.

- Console applications.
- Scripted or hosted application.
- Windows GUI application (windows form).
- ASP.NET applications.
- XML web services.
- Windows Services.

.NET platform architecture

The diagram below gives us an overview of the .NET architecture. At the bottom of the diagram is your operating system, above that sits the .NET Framework that acts as interface to it.



.NET Framework

4.3.3 ASP.NET

ASP.NET is a programming framework built on the common language runtime that can be used on a server to build powerful Web applications. ASP.NET offers several important advantages over previous Web development models. ASP.NET is compiled common language runtime code running on the server. It is interpreted predecessor. ASP.NET can take advantages of early binding, just time compilation, native optimization and catching services right out of the box. ASP.NET gives dramatically better performance to write a line of code. The ASP.NET framework is complemented by a rich tool box and designer in the visual studio integrated development environment:

Structure

Asp.net brings structure back into programming by offering a code behind page, which separates the client-side script and html from the server-side code.

Compiled code

Asp.net solves the problem of running interpreted script by compiling the server side code into intermediate language (it is faster than interpreted script).

Early binding

Asp.net also uses early binding when making calls to com components resulting in faster performance.

Security

Asp.net has an enhanced security infrastructure that can be quickly configured and programmed to authenticate and authorize website users.

Performance

Asp.net contains performance enhancements such as page and data caching.

Diagnostics

Asp.net offers an enhanced tracing and debugging option, which will save time when you are ready to get the system running.

Session state

Asp.net has an improved session object. Session state can be configured to be share among all servers in a web farm.

Language Support

The Microsoft .NET Platform currently offers built-in support for three languages: C#,Visual Basic, and JScript. The exercises and code samples in this tutorial demonstrate how to use C#, Visual Basic, and JScript to build .NET applications.

Power and flexibility

ASP.NET is based on the common language runtime, the power and flexibility of that entire flat form is available to web developers. The .NET framework class library, messaging and data access solutions are all seamlessly accessible from the web.ASP.NET is also language independent, so we can choose that best applies to your application or portion your application across many languages.

Simplicity

ASP.NET makes it easy to perform common tasks, from simple form submission and client authentication to deployment and site configuration.

E.g.: The ASP.NET page framework allows to build user interface that cleanly separate application logic from presentation code and to handle events in a simple, visual Basic – like form processing model. Additionally, the common language runtime simplifies development, with managed code services such as automotive reference counting and garbage collection.

Manageability

ASP.NET employs a text based, hierarchical configuration system, which applies settings to your server environment and web application .The information is stored as plain text, and new setting may be applied with the aid of local administration tools. ASP.NET frame work application is deployed to a server simplicity by the ASP.NET runtime ,so that if one leaks, deadlocks a new process can be created in its place., which helps to keep your constantly available to handle requests.

Scalability and availability

ASP.NET has been designed with scalability in mind, with features specifically tailored to improve performance in clustered and multiprocessor environment. The processor are closely monitored and managed by the ASP.NET runtime, so that if one leaks, deadlocks a new process can be created in its place, which helps to keep your application constantly available to handle requests.

Customizability and Extensibility

ASP.NET delivers a well factored architecture that allows the developers to “plug-in”, their code at appropriate level. Then it is possible to replace any subcomponent of the ASP.NET runtime with your own customer- when component implementing custom authentication or state services has never been easier.

Security

To built in windows authentication and per application configuration can be assured on our applications are secure.

ASP.NET Web Forms

The ASP.NET Forms page frame work is scalable common language runtime programming model that can be used on the server to dynamically generate web pages. The ASP.NET web forms frameworks has been specifically designed to address a number of deficiencies.

The ASP.NET server controls

The ASP.NET server controls using <% %> code blocks to perform dynamic
Content, as ASP.NET page developers can use ASP.NET server controls to program web pages server controls are declared with in an .aspx files using system tags or intrinsic HTML tags are handled by one of the control in the system. Web.UI.HTML control name space. Any tag does not explicitly map to one of the control is assigned the type of system.Web.UI.HTML controls .At runtime these server controls automatically generate HTML content.

The server controls automatically maintain any client entered values between round trips to the server. Each ASP.NET server control is capable of exposing an object model containing properties, methods and events.ASP.NET developers can use this project model to cleanly modify and interact with page.

Validation Controls:

The ASP.NET web forms page frame work provides a set of validations server control that provides an easy to use but powerful way to check input forms for errors and if necessary to display messages to the user. Validation controls are added to an ASP.NET page like other server controls.

Code Behind Web Forms:

ASP.NET supports two methods of authoring dynamic pages. The first is the methods shown in the preceding samples where the page code is physically declared within the originated .aspx file.

Performing Page Navigation:

The navigation scenarios are initiated through hyper link on the client. Client side page redirect or navigates can also be initiated from the server by an ASP.NET page developer by calling the Response.Redirect(url) methods.

Web Services:

Web sites that just deliver user interface pages to browsers to a next generation of programmable web sites that directly link organizations, applications, services and devices with one another. These programmable web sites become more than passively accessed sites – they become reusable, intelligent web services. The resulting model is both scalable and extensible, and embraces open internet standards so that it can be accessed.

4.3.4 SQL EXPRESS

SQL Express is the database used in this project. SQL (Structured Query Language) is a database computer language designed for the retrieval and management of data in relational database management systems (RDBMS), database schema creation and modification, and database object access control management.

SQL is a querying language for querying and modifying data and managing databases. SQL was standardized first by the ANSI and later by the ISO. Most database management systems implement a majority of one of these standards and add their proprietary extensions. SQL allows the retrieval, insertion, updating, and deletion of data. A database management system also includes management and administrative functions. Most – if not all – implementations also include a command-line interface (SQL/CLI) that allows for the entry and execution of the language commands, as opposed to only providing an application programming interface (API) intended for access from a graphical user interface (GUI).

The first version of SQL was developed at IBM by Andrew Richardson, Donald C. Miserly and Raymond F. Boyce in the early 1970s. Subsequent versions of the SQL standard have been released by ANSI and as International Organization for Standardization (ISO) standards.

Originally designed as a declarative query and data manipulation language, variations of SQL have been created by SQL database management system (DBMS) vendors that add procedural constructs, flow-of-control statements, user-defined data types, and various other language extensions.

Common criticisms of SQL include a perceived lack of cross-platform portability between vendors, inappropriate handling of missing data (see Null (SQL)), and unnecessarily complex and occasionally ambiguous language grammar and semantics.

Procedural extensions

SQL is designed for a specific purpose: to query data contained in a relational database. SQL is a set-based, declarative query language, not an imperative language such as C or BASIC. However, there are extensions to Standard SQL which add procedural programming language functionality, such as control-of-flow constructs.

Language elements

The SQL language is sub-divided into several language elements, including:

- Clauses, which are in some cases optional, constituent components of statements and queries.
- Expressions which can produce either scalar values or tables consisting of columns and rows of data.
- Predicates which specify conditions that can be evaluated to SQL three-valued logic (3VL) Boolean truth values and which are used to limit the effects of statements and queries, or to change program flow.
- Queries which retrieve data based on specific criteria.
- Statements which may have a persistent effect on schemas and data, or which may control transactions, program flow, connections, sessions, or diagnostics.
- SQL statements also include the semicolon (";") statement terminator. Though not required on every platform, it is defined as a standard part of the SQL grammar.
- Insignificant whitespace is generally ignored in SQL statements and queries, making it easier to format SQL code for readability.

Queries

The most common operation in SQL databases is the query, which is performed with the declarative SELECT keyword. SELECT retrieves data from a specified table, multiple related tables in a database or the result of an expression. While often grouped with Data Manipulation Language (DML) statements, the standard SELECT query is considered separate from SQL DML, as it has no persistent effects on the data stored in a database. There are some platform-specific variations of SELECT that can persist their effects in a database, such as the SELECT INTO syntax that exists in some databases.

SQL queries allow the user to specify a description of the desired result set, but it is left to the devices of the database management system (DBMS) to plan, optimize, and perform the physical operations necessary to produce that result set in as efficient a manner as possible. An SQL query includes a list of columns to be included in the final result immediately following the SELECT keyword. An asterisk ("*") can also be used as a "wildcard" indicator to specify that all available columns of a table (or multiple tables) are to be returned. SELECT is the most complex statement in SQL, with several optional keywords and clauses, including:

- The FROM clause which indicates the source table or tables from which the data is to be retrieved. The FROM clause can include optional JOIN clauses to join related tables to one another based on user-specified criteria.
- The WHERE clause includes a comparison predicate, which is used to restrict the number of rows returned by the query. The WHERE clause is applied before the GROUP BY clause. The WHERE clause eliminates all rows from the result set where the comparison predicate does not evaluate to True.
- The GROUP BY clause is used to combine, or group, rows with related values into elements of a smaller set of rows. GROUP BY is often used in conjunction with SQL aggregate functions or to eliminate duplicate rows from a result set.
- The HAVING clause includes a comparison predicate used to eliminate rows after the GROUP BY clause is applied to the result set. Because it acts on the results of the GROUP BY clause, aggregate functions can be used in the HAVING clause predicate.
- The ORDER BY clause is used to identify which columns are used to sort the resulting data, and in which order they should be sorted (options are ascending or descending). The order of rows returned by an SQL query is never guaranteed unless an ORDER BY clause is specified.

a) Data manipulation

First, there are the standard Data Manipulation Language (DML) elements. DML is the subset of the language used to add, update and delete data:

- INSERT is used to add rows (formally tuples) to an existing table
- UPDATE is used to modify the values of a set of existing table rows
- DELETE removes zero or more existing rows from a table
- MERGE is used to combine the data of multiple tables. It is something of a combination of the INSERT and UPDATE elements.

b) Data definition

The second group of keywords is the Data Definition Language (DDL). DDL allows the user to define new tables and associated elements. Most commercial SQL databases have proprietary extensions in their DDL, which allow control over nonstandard features of the database system. The most basic items of DDL are the CREATE, ALTER, RENAME, TRUNCATE and DROP statements:

- CREATE causes an object (a table, for example) to be created within the database.
- DROP causes an existing object within the database to be deleted, usually irretrievably.
- TRUNCATE deletes all data from a table in a very fast way. It usually implies a subsequent COMMIT operation.
- ALTER statement permits the user to modify an existing object in various ways -- for example, adding a column to an existing table.

c) Data control

The third group of SQL keywords is the Data Control Language (DCL). DCL handles the authorization aspects of data and permits the user to control who has access to see or manipulate data within the database. Its two main keywords are:

- GRANT authorizes one or more users to perform an operation or a set of operations on an object.
- REVOKE removes or restricts the capability of a user to perform an operation or a set of operations.

CHAPTER 5
DESIGN

5.1 SYSTEM DESIGN

System design is the process by which an agent creates a specification of a software artifact, intended to accomplish goals, using a set of primitive components and subject to constraints. Software design may refer to either "all the activities involved in conceptualizing, framing, implementing, commissioning, and ultimately modifying complex systems" or "the activity following requirements specification and before programming, as a stylized software engineering process.

Software design usually involves problem solving and planning a software solution. This includes both low-level component and algorithm design and high-level, architecture design. Software design is the process of implementing software solutions to one or more set of problems. One of the important parts of software design is the software requirements analysis (SRA). It is a part of the software development process that lists specifications used in software engineering. If the software is "semi-automated" or user centered, software design may involve user experience design yielding a storyboard to help determine those specifications. If the software is completely automated (meaning no user or user interface), a software design may be as simple as a flow chart or text describing a planned sequence of events. There are also semi-standard methods like Unified Modeling Language and Fundamental modeling concepts. In either case, some documentation of the plan is usually the product of the design. Furthermore, a software design may be platform-independent or platform-specific, depending upon the availability of the technology used for the design.

Software design can be considered as creating a solution to a problem in hand with available capabilities. The main difference between software analysis and design is that the output of a software analysis consists of smaller problems to solve. Also, the analysis should not be very different even if it is designed by different team members or groups. The design focuses on the capabilities, and there can be multiple designs for the same problem depending on the environment that solution will be hosted. They can be operations systems, webpages, mobile or even the new cloud computing paradigm. Sometimes the design depends on the environment that it was developed for, whether it is created from reliable frameworks or implemented with suitable design patterns.

Software design is both a process and a model. The design process is a sequence of steps that enable the designer to describe all aspects of the software to be built. It is important to note, however, that the design process is not simply a cookbook. Creative skill, past experience, a sense of what makes "good" software and an overall commitment to quality are critical success factors for a competent design. The design model is the equivalent of an architect’s plans for a house. It begins by representing the totality of the thing to be built (e.g., a three-dimensional rendering of the house) and slowly refines the thing to provide guidance for constructing each detail (e.g., the plumbing layout). Similarly, the design model that is created for software provides a variety of different views of the computer software. Basic design principles enable the software engineer to navigate the design process.

5.1.1 DATA FLOW DIAGRAM

Data flow diagrams are commonly used to get a pictorial representation of the various processes occurring within the system which shows the minimum content of data stores. Each data stores should contain all the data elements that flow in and out.

The following are some advantages:

- Provide an overview of DFD and the transformation of data.
- Act as a good communication tool with users. Several methods have been devised to control processing activities. One such is the batch processing. In addition to the batch controls, several other programmed checks can use for testing the data.
- Completeness checks ensure that all the fields in a record are present and read in a proper sequence.
- A consistency check refers to the relevance to the one type of data or another.
- A sequence check verifies that data records are in the sequence prior to processing.

DFD show the flow of data from external entities into the system, showed how the data moved from one process from to another as well as its logical storage. There are only four symbols.

External Entity

An external entity is a source or destination of a data flow. This is outside the area of study. The symbol used in an oval containing a meaningful and unique identifier.

Process

A process shows a transformation for manipulation of data flows within the system. The symbol used is a rectangular box which contains three descriptive elements. An identification number appears in the upper left hand corner. This is allocated arbitrarily at the top level and serves as a unique reference. A location appears to the right of the identifier and describes where in the system that process takes place.

Data Flow

A data flow shows the flow of information from its source to its destination. A data flow is represented by a line, with arrow heads showing the direction of flow. Information always flows to or from a process and may be written, verbal or electronic. Each data flow may be referenced by the process or data stores at its head and tail, or by a description of its contents.

Data Store

A data store is a holding place for information within the system .it is represented by an open ended narrow rectangle. A DFD consists of a series of bubbles joined by lines. The bubble represents data transformation and line represents flow in the system. In the normal convention a DFD has four major symbols.

Square: which defines source or destination of data

Arrow: which shows data flow.

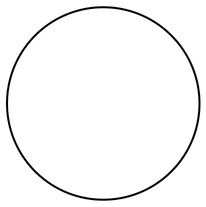
Circle: which represent a process that transforms incoming data in to outgoing flow

Open Rectangle: which shows data store

- **Entities:**

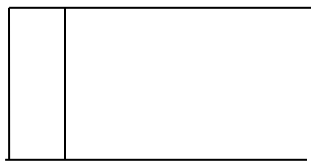
External entities represent the sources if data that enter the system or the recipients of data that leave the system.

- **Process:**



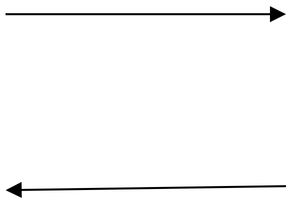
Processes represent activities in which data is manipulated by being stored or retrieved or transformed in some way. A circle represents it. The process will show the data transformation or change.

- **Databases:**



Databases represent storage of data within the system.

- **Data flow:**



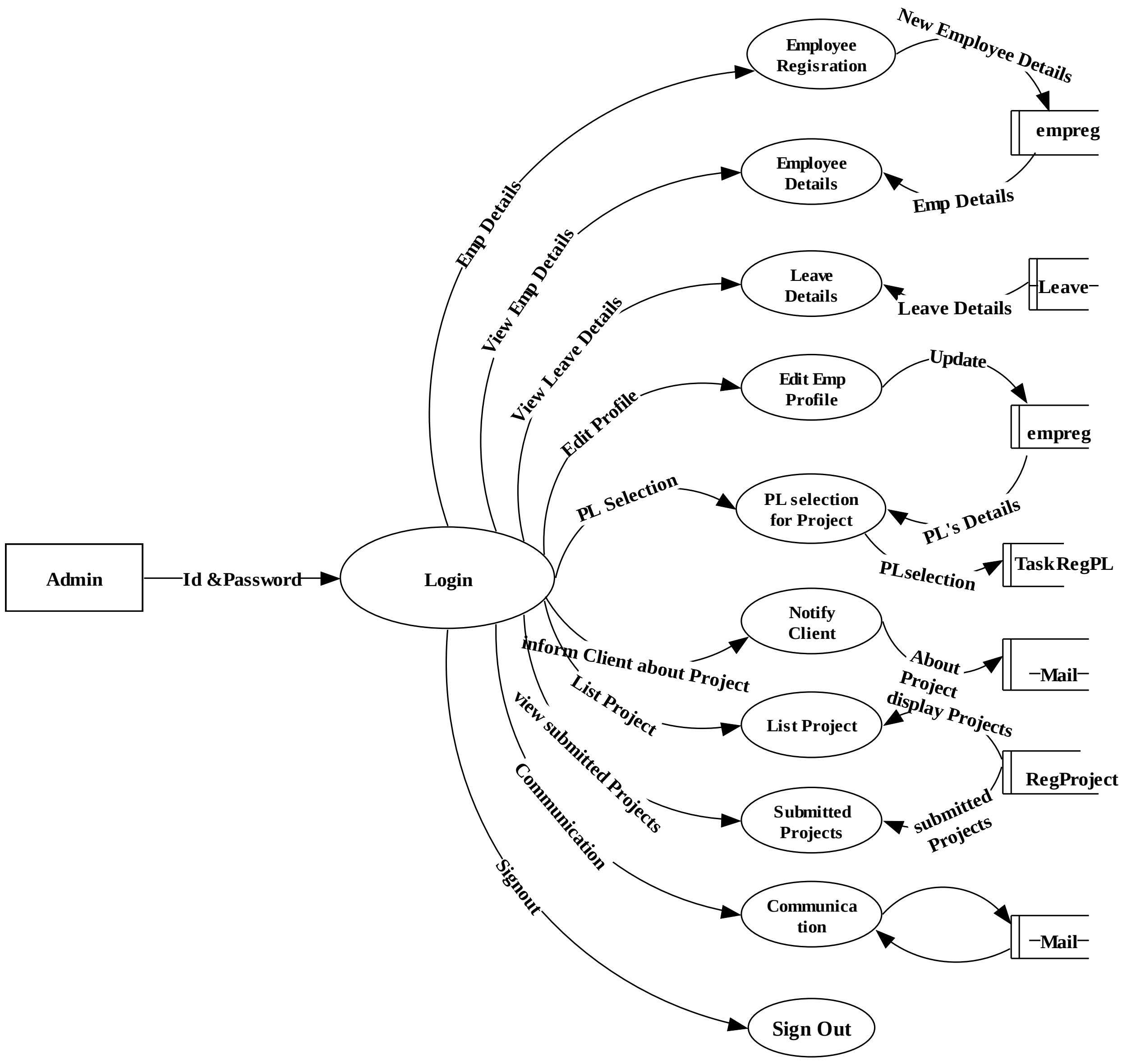
A data flow shows the flow of information from its source to its destination. A line represents a data flow, with arrow heads showing the direction of flow.

Bubbles or curved rectangles represent the process and arrows indicate dataflow. External entities are represented by rectangles and are outside the systems such as users or customer with whom the system interacts. They either supply or consume data. Entities supplying data are known as source and those that consume data are called links. Data are stored in a data store by a process in the system. Each component in a DFD is labeled with descriptive name. Process name are further identified with number. Context level DFD is draw first. Then the process are decomposed in to several elementary levels and represented in the order of importance

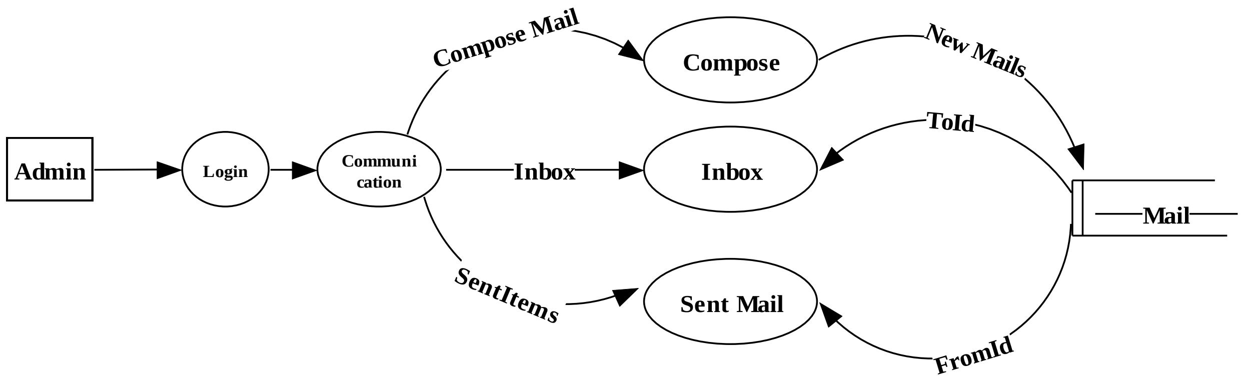
A DFD describes what data flow (logical) rather than how they are processed, so it does not dependent on hardware, software and data stricture or file organization. A DFD methodology is quite effective; especially when the required design is clear and the analyst need a notation language for communication. A DFD is easy to understand after a brief orientation.

5.1.2 PROJECT DFD

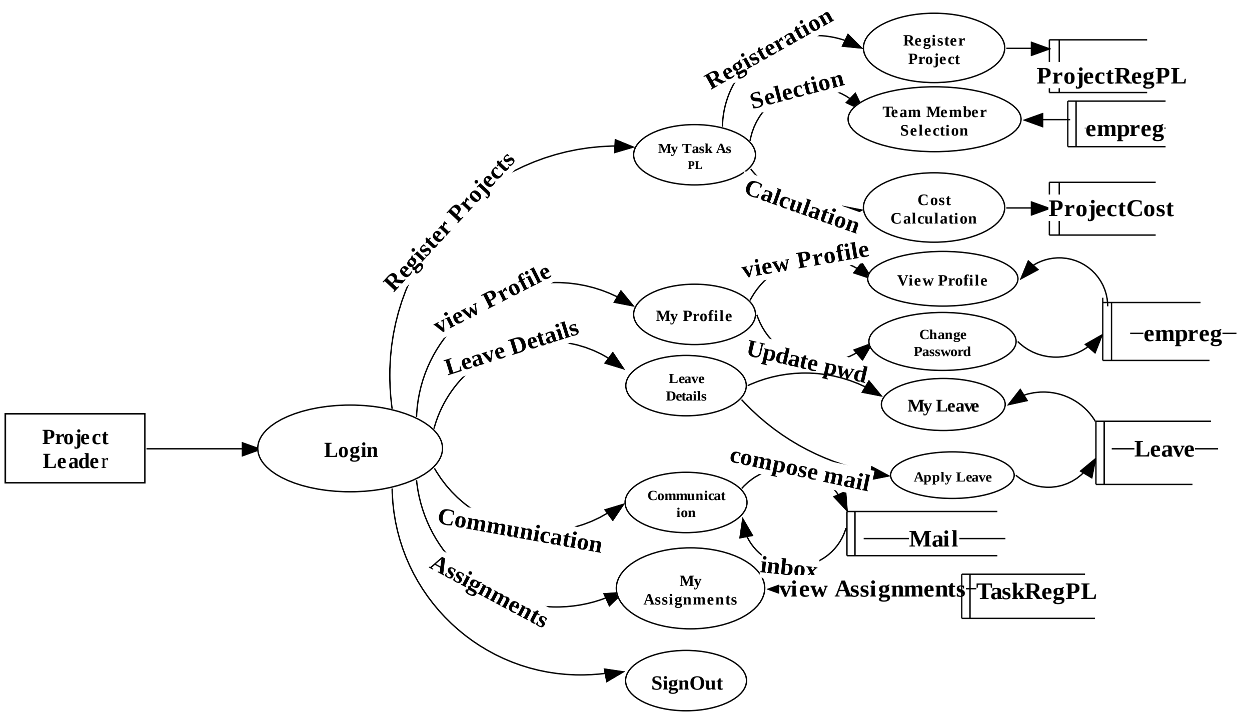
Level 1 For Admin



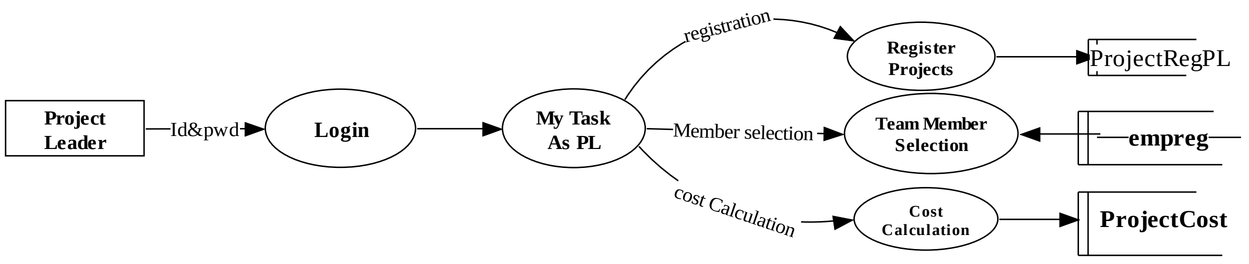
Level1.1 for Admin



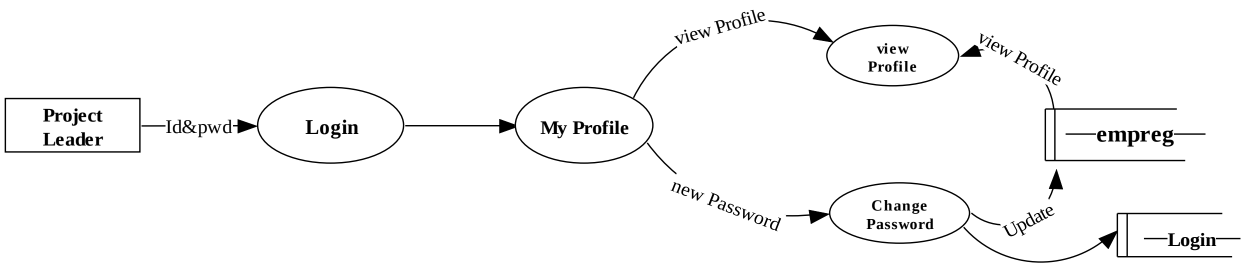
Level1 for Project Leader



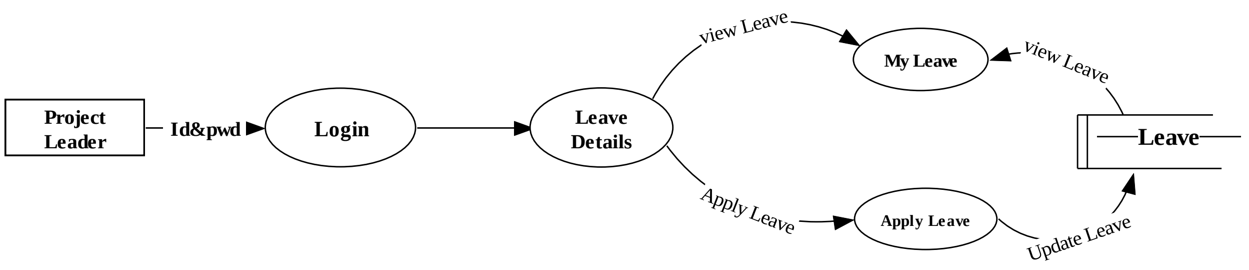
Level 1.1 for Project Leader



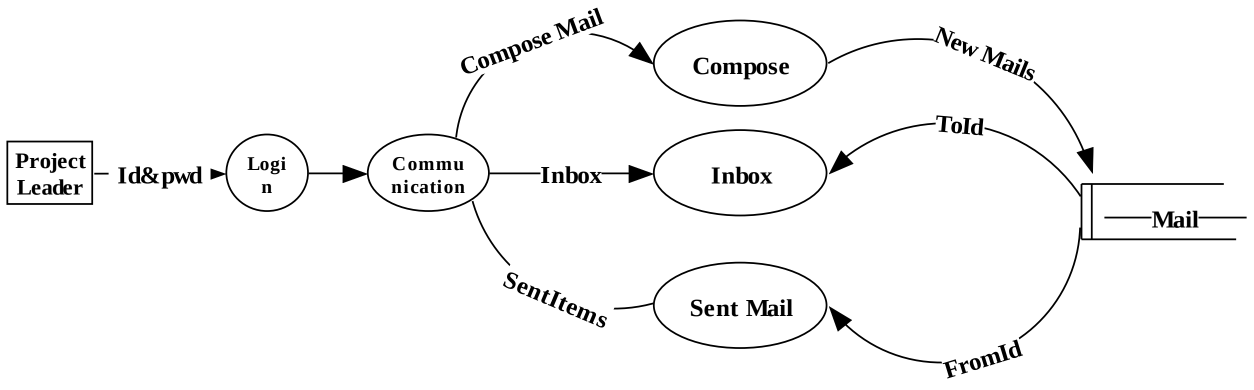
Level1.2 for Project Leader



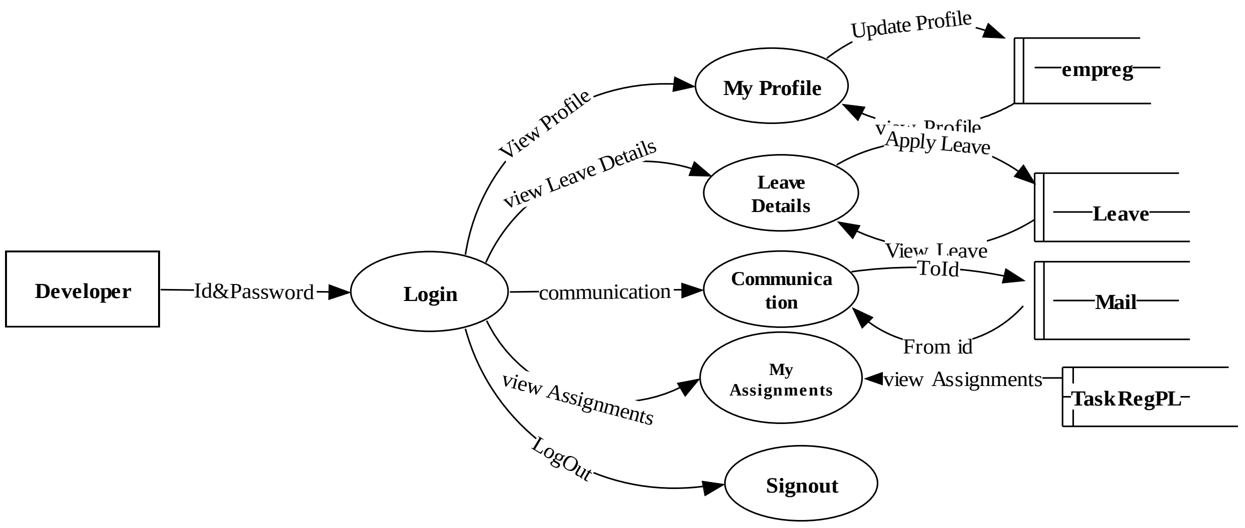
Level 1.3 for Project Leader



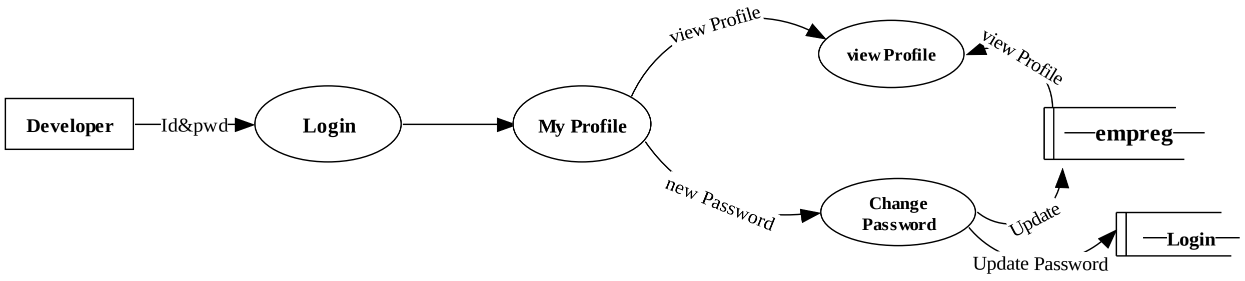
Level 1.4 for Project Leader



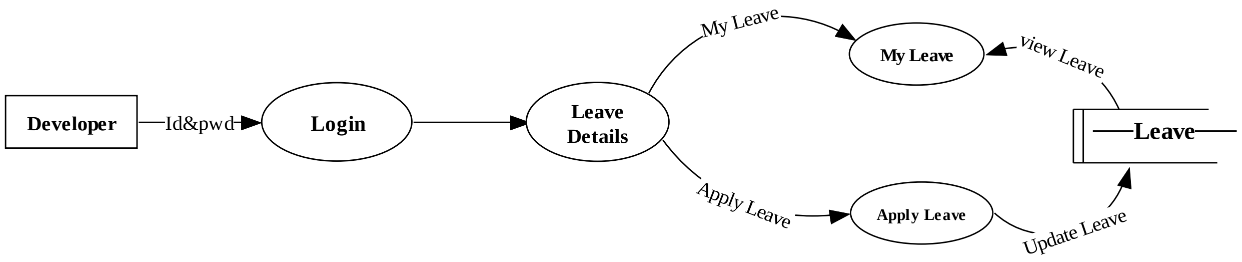
Level1 For Developer



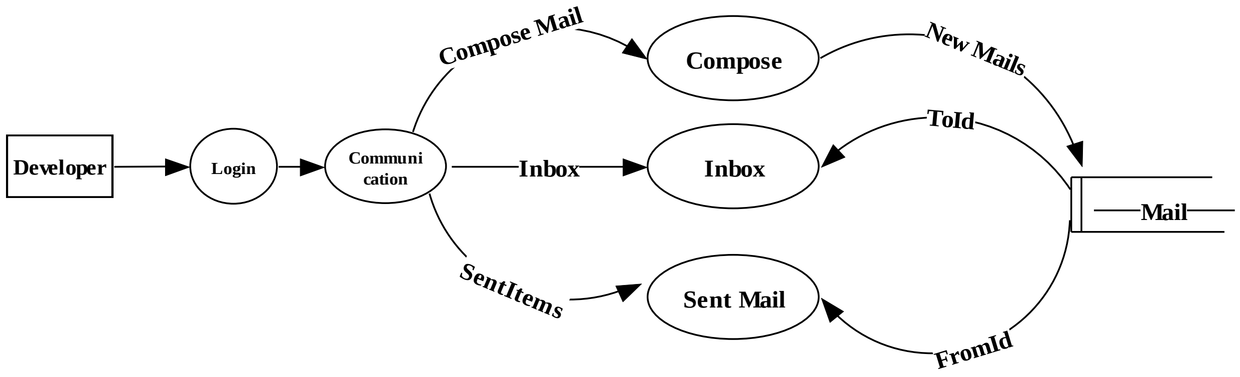
Level 1.1 for Developer



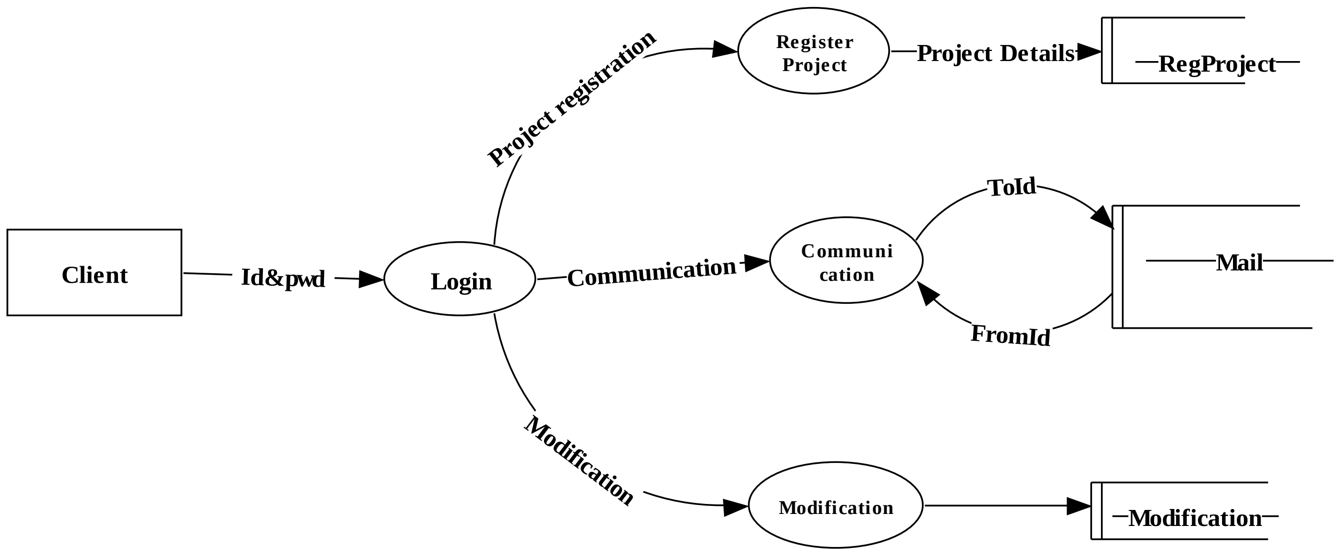
Level 1.2 for Developer



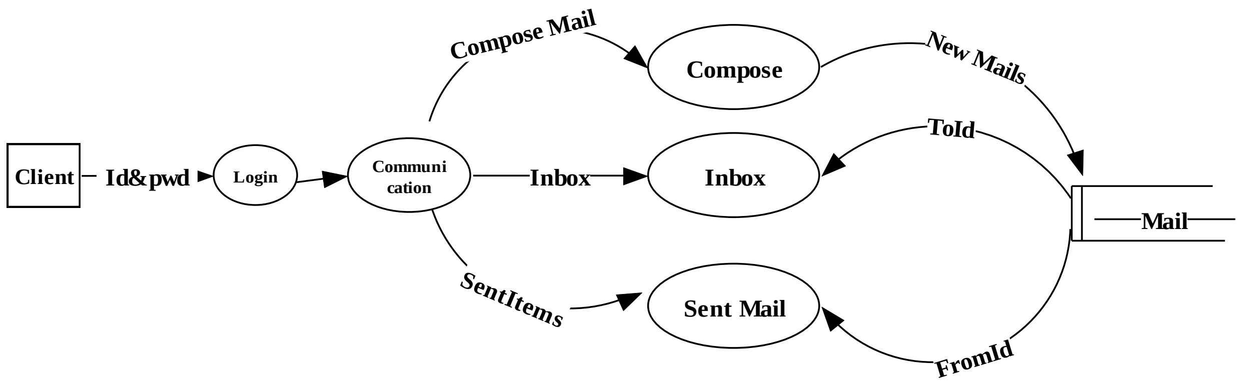
Level 1.3 for Developer



Level1 for Client



Level1.1 for Client



5.2 DATABASE DESIGN

Database Concepts:

A database is a logically coherent collection of data with some inherent meaning, representing some aspect of real world and which is designed, built and populated with data for a specific purpose.

It is a collection of programs that enables user to create and maintain a database. In other words it is general-purpose software that provides the users with the processes of defining, constructing and manipulating the database for various applications.

Normalization

Normalization is a design technique that is widely used as a guide in designing relational databases. There are two goals of the normalization process:

- Eliminate redundant data (for example, storing the same data in more than one table) and
- Ensure data dependencies make sense (only storing related data in a table).

The database community has developed a series of guidelines for ensuring that databases are normalized. These are referred to as normal forms and are numbered from one (the lowest form of normalization, referred to as first normal form or 1NF) through five (fifth normal form or 5NF). In practical applications, you'll often see 1NF, 2NF, and 3NF along with the occasional 4NF.

Constraints in creating a table:

- Candidate Key: A candidate key is a combination of one or more columns, the value of which uniquely identifies each row of a table.
- Primary Key: The primary key of a table is one of the candidate keys that you give some characteristics. We can have only one primary key, and a primary key column cannot contain NULLs.
- Foreign Key: A foreign key is a combination of columns with values based on primary key values from another table. A foreign key constraint, also known as Referential Integrity constraint, specifies that the values of the foreign key correspond to actual values of the primary key in the other table.
- CHECK Constraint: Many columns must have values that are within a certain range or that satisfy certain conditions. With a CHECK constraint, we can specify an expression that must always be true for every row in the table.

Rules of Data Normalization:

1NF : Eliminate Repeating Groups

- Make a separate table for each set of related attributes, and give each table a primary key.

2NF **:** Eliminate Redundant Data

- If an attribute depends on only part of a multi-valued key, remove it to a separate table.

3NF **:** Eliminate Columns Not Dependent On Key

- If attributes do not contribute to a description of the key, remove them to a separate table.

5.3 INPUT DESIGN

The Input design is the link that lays the information system to the world of users. In Input design user defined inputs are converted into computer-based format. Input design involves determining the record media, method of Input, speed of capture and entry to the system. The data is fed into the system using simple interactive forms. Input design consists of developing specification and procedures for data preparation, steps necessary to put transaction data in a form that is unable for computer proceeding. The data is validated wherever it requires in the project. This ensures only correct data is entered to the system.

The goal of designing input data is to make the automation as easy and free from errors as possible for providing a good design for the application easy data input and selection features adopted. Input design involves determining the record media, method of input, speed of capture and entry to the system.
GUI is the interface used in input design.
The major input designs in these modules are:

- Login Form
- Client Registration form
- Employee Registration Form
- Add new Projects

User Friendly

User is never lift in confusion as to what is happening. Instead appropriate error messages and acknowledgements are displayed to the users.

Interactive dialog

The system engages the user in and interactive dialog. The system us able to extract missing information from the user by directing through appropriate messages that are displayed.

5.4 OUTPUT DESIGN

It generally refers to the results and the information that are generated by the system. Effective, descriptive

and useful design will improve the relationship with the user and the system because it is the direct source of information to the user.
In the output design it is determined how the information is to be displayed for immediate need and also hard copy output.

The objective of the output design is to convey the information of all the past activities, current status and to emphasize important events. Outputs from the computers are required primarily to communicate the results of processing to the users. They also used to provide a permanent copy of these results for later consultation.

The major outputs designs are:

- Employee Details
- Leave Details
- Project Details

5.5 PROGRAM DESIGN

A design methodology combines a systematic set of rules for creating a program design with diagramming tools needed to represent it. Procedural design is best used to model programs that have an obvious flow of data from input to output. It represents the architecture of a program as a set of interacting processes that pass data from one to another.

Transforms structural elements of the architecture into a procedural description of software construction.

CHAPTER 6
FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS

A **System Requirements Specification** (abbreviated SRS when need to be distinct from a Software Requirement Specification SRS) is a structured collection of information that embodies the requirements of a system.

A business analyst sometimes titled system analyst. , is responsible for analysing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the system development life cycle domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers.

6.1 FUNCTIONAL REQUIREMENTS

Any requirement which specifies **what** the system should do. In other words, a functional requirement will describe a particular behavior of function of the system when certain conditions are met.

Typical functional requirements include:

- Business Rules
- Transaction corrections, adjustments and cancellations
- Administrative functions
- Authentication
- Authorization levels
- Audit Tracking
- External Interfaces
- Certification Requirements
- Reporting Requirements
- Historical Data
- Legal or Regulatory Requirements

6.2 NON-FUNCTIONAL REQUIREMENTS

Any requirement which specifies **how** the system performs a certain function. In other words, a non-functional requirement will describe how a system should behave and what limits there are on its functionality.

Non-functional requirements generally specify the system's quality attributes or characteristics, for example: "Modified data in a database should be updated for all users accessing it within 2 seconds."

Typical non-functional requirements include:

- Performance – for example: response time, throughput, utilization, static volumetric
- Scalability
- Capacity
- Availability
- Reliability
- Recoverability
- Maintainability
- Security
- Manageability
- Data Integrity
- Usability

It is important to correctly state non-functional requirements since they'll affect your users'experience when interacting with the system.

CHAPTER 7
TESTING

7.1 TESTING STRATEGIES

The software, which has been developed, has to be tested to prove its validity. Testing is considered to be the least creative phase of the whole cycle of system design. In the real sense it is the phase, which helps to bring out the creativity of the other phases, and makes it shine. That is testing represent the ultimate review of specification, design and code.

Any software has to be tested with pre-planned strategies. As Roger Pressman states, the preparation for testing should start as soon as the design of the system starts. To carry out the testing in an efficient manner certain amount of strategic planning has to be done. Any testing strategy must incorporate test planning, test case design, test execution and the resultant data collection and evaluation.

Testing is a process of executing a program with intend of finding an error. A good test case is one that has a high probability of finding yet undiscovered errors.

7.2 UNIT TESTING

This is the first level of testing. In this different modules are tested against the specifications produced during the design of the modules. Unit testing is done for the verification of the code produced during the coding of single program module in an isolated environment. Unit testing first focuses on the modules independently of one another to locate errors.

After coding each dialogue is tested and run individually. All unnecessary coding were removed and it was ensured that all the modules worked, as the programmer would expect. Logical errors found were corrected. So, by working all the modules independently and verifying the outputs of each module in the presence of staff was concluded that the program was functioning as expected.

7.3 INTEGRATION TESTING

Data can be lost access an interface, one module can have as adverse effort on another sub functions when combined, may not produce the desired major functions. Integration testing is a systematic testing for constructing the program structure, while at the same time conducting tests to uncover errors associated within the interface. The objectives are to take unit tested as a whole. Here correction is difficult because the vast expenses of the entire program complicate the isolation of causes. Thus in the integration testing step, all the errors uncovered are corrected for the next testing steps.

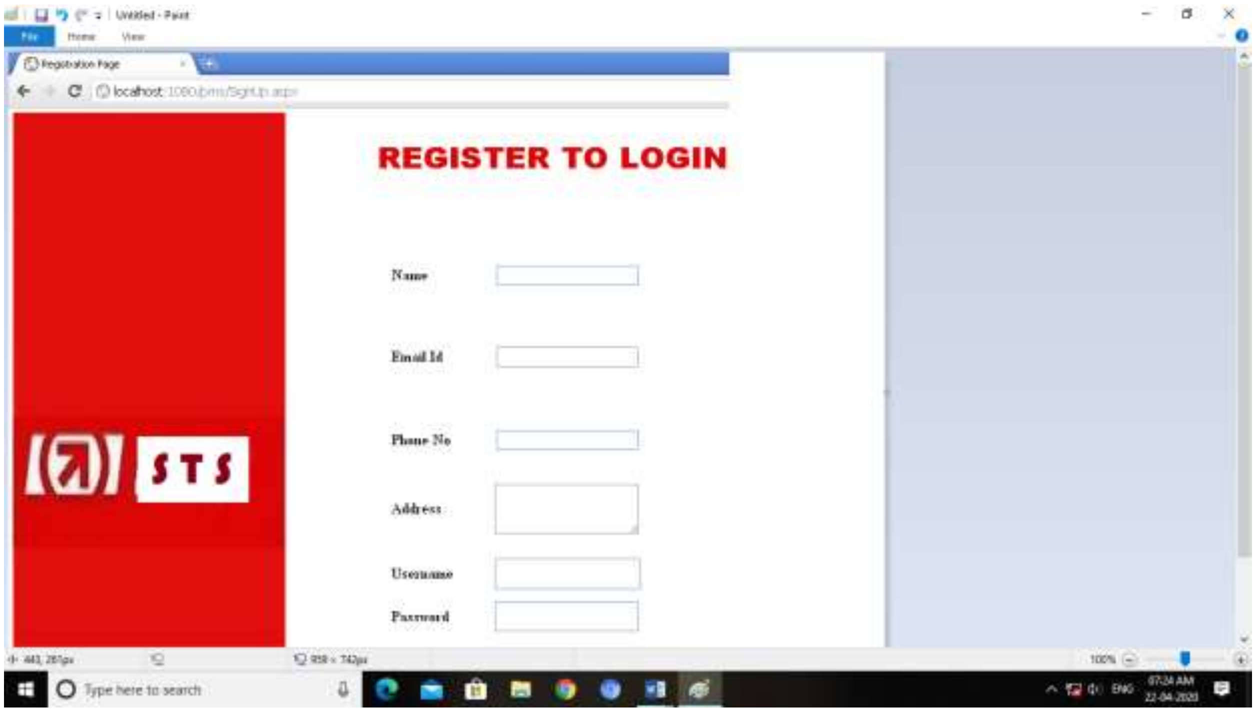
7.4 SYSTEM TESTING

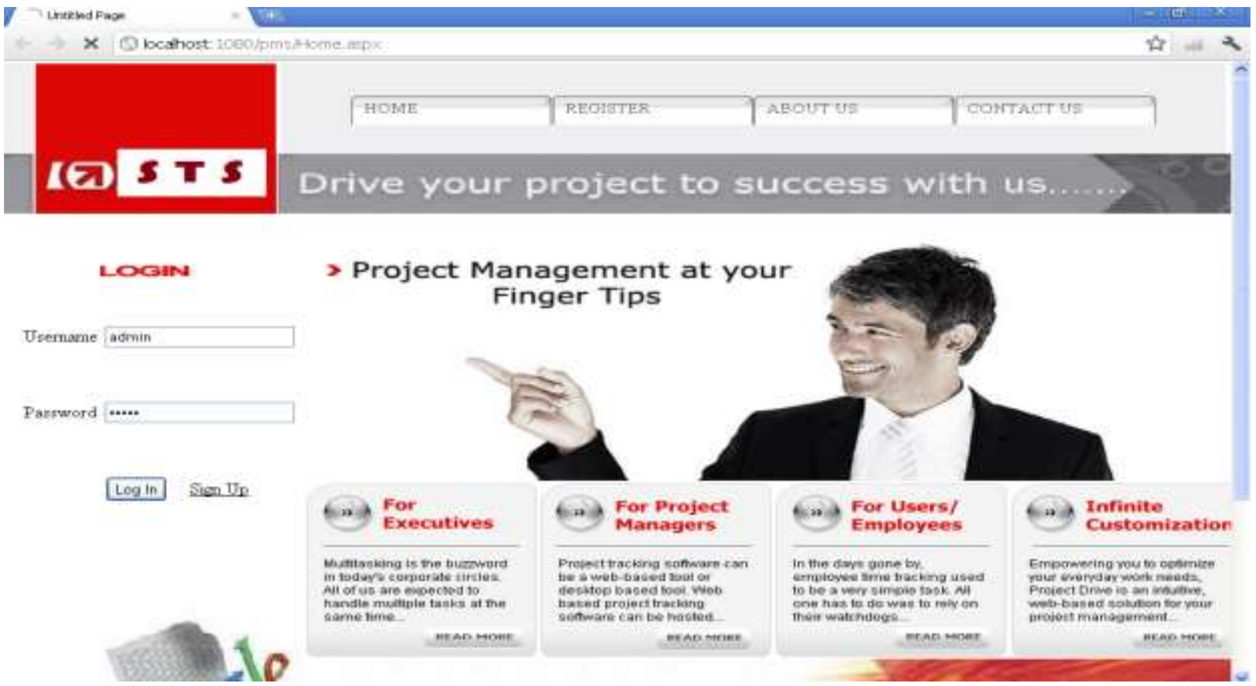
7.5 TESTING RESULTS

Sl no	Test case	Input	Expected Result	Current Result
1	Registration page	Invalid data type	Enter a valid name	Successful
2	Registration page	Invalid e-mail id	Enter a valid e-mail id	Successful
3	Registration page	Existing username	Username already existed	Successful
4	Login page	Incorrect username	Invalid username or password	Successful
5	Login page	Incorrect password	Invalid username or password	Successful
6	Login Page	No. of characters is more than the limit	The maximum number of digit is limited to 10	Unsuccessful
7	Registration page	Pin code contain at least 6 digit	Enter 6 digit pin code	Successful
8	Registration page	Retype the password	Please confirm the password given	Successful
9	Communication page	To\id is blank	Please select the email id	Successful
10	Project Registration	Didn’t type the module name	Please enter the module name	Successful
11	Project cost calculation	If the project name is not selected	Please select the project name	Successful

CHAPTER 8
RESULTS AND DISCUSSION

8.1 SCREEN SHOTS



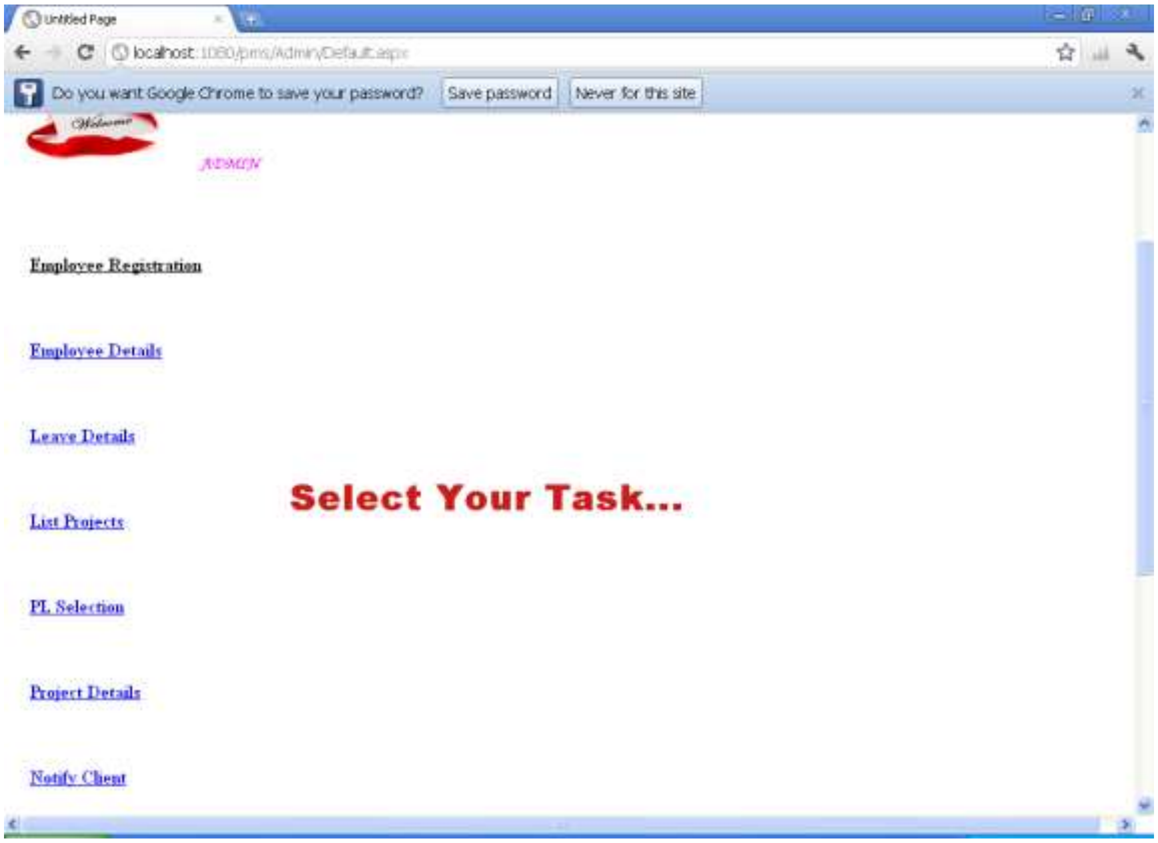


Multitasking is the buzzword in today's corporate circles. All of us are expected to handle multiple tasks at the same time...

Project tracking software can be a web-based tool or desktop based tool. Web based project tracking software can be hosted...

In the days gone by, employee time tracking used to be a very simple task. All one has to do was to rely on their watchdogs...

Empowering you to optimize your everyday work needs, Project Drive is an intuitive, web-based solution for your project management...



Untitled Page

localhost:1080/pms/Admin/Default.aspx

Do you want Google Chrome to save your password? Save password Never for this site

ADMIN

Employee Registration

Employee Details

Leave Details

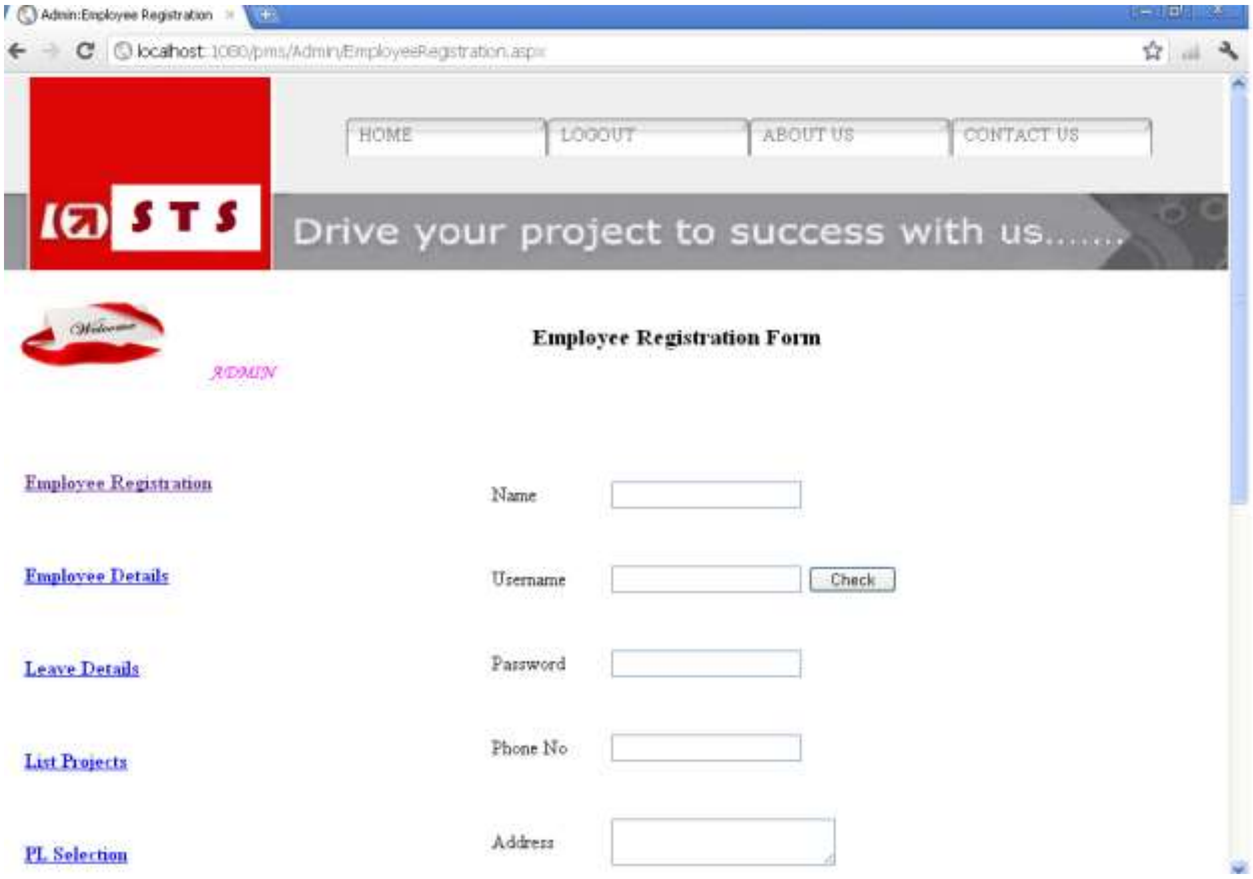
List Projects

PL Selection

Project Details

Notify Client

Select Your Task...



Admin:Employee Registration

localhost:1080/pms/Admin/EmployeeRegistration.aspx

HOME LOGOUT ABOUT US CONTACT US

STS Drive your project to success with us.......

ADMIN

Employee Registration Form

Employee Registration

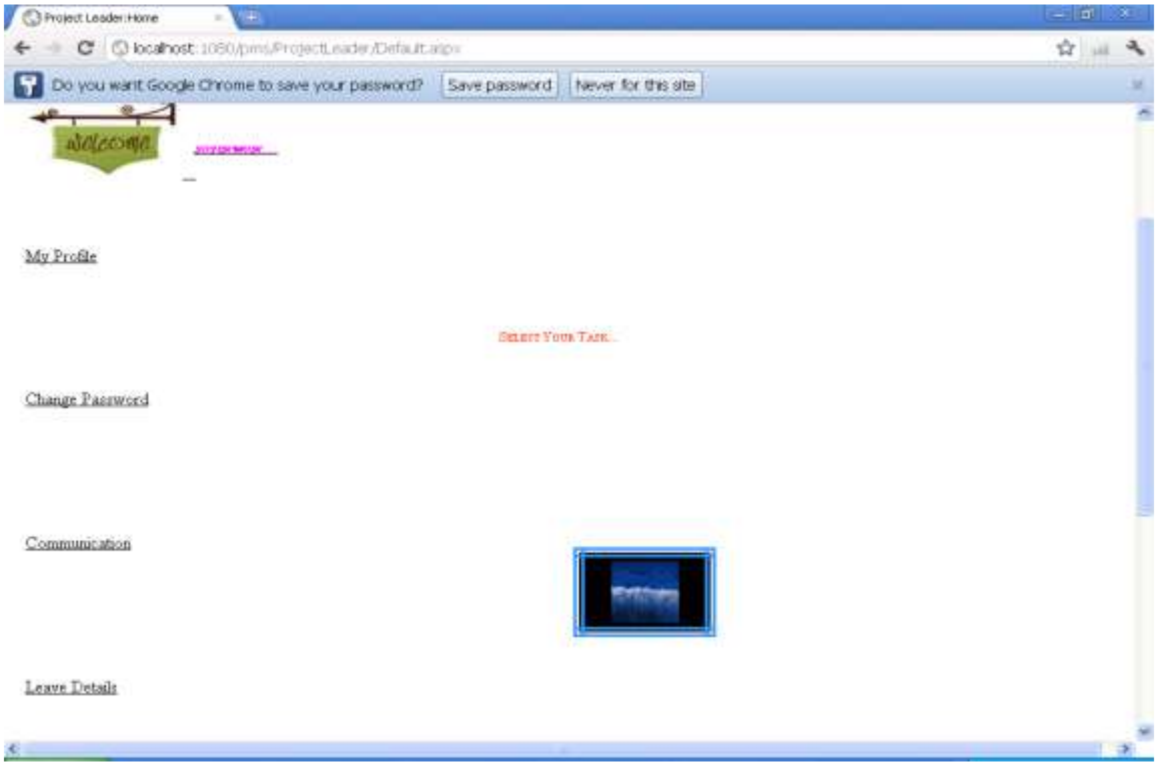
Employee Details

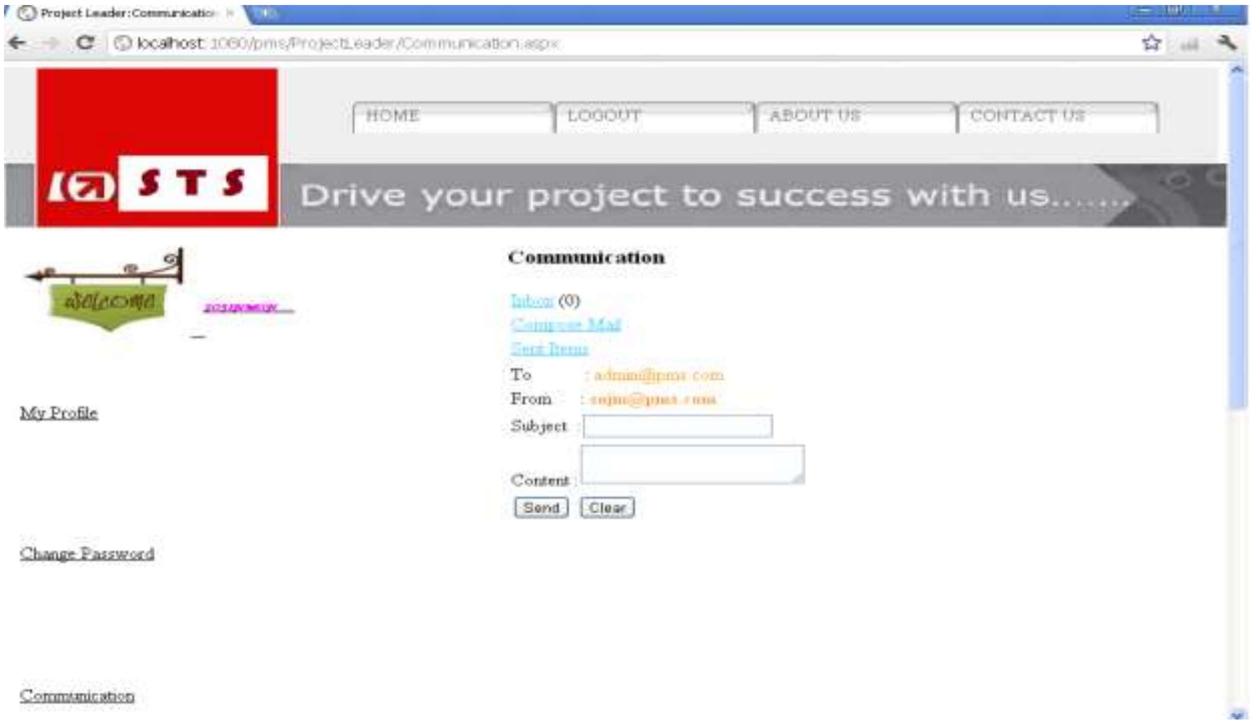
Leave Details

List Projects

PL Selection

Name	
Username	Check
Password	
Phone No	
Address	



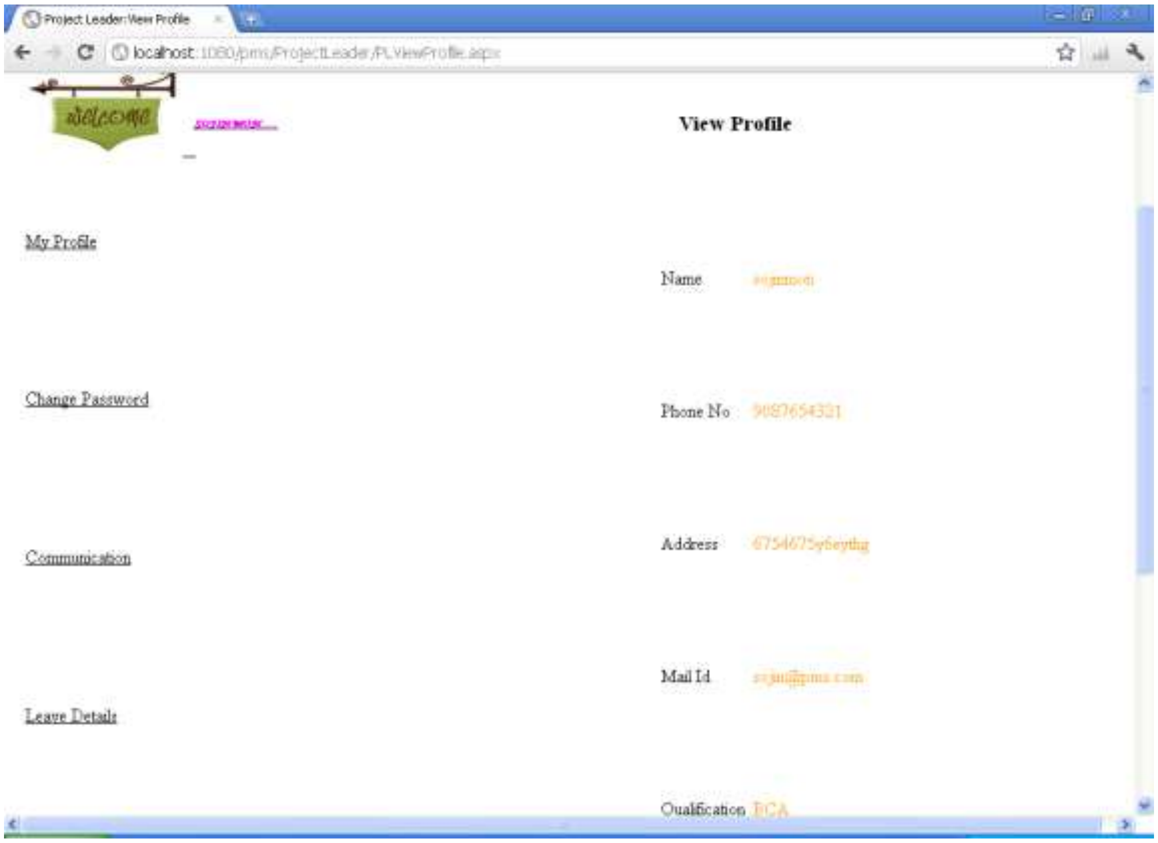

Project Leader:Communication
localhost:1080/pms/ProjectLeader/Communication.aspx
HOME
LOGOUT
ABOUT US
CONTACT US
STS
Drive your project to success with us.......
welcome
My Profile
Change Password
Communication

Communication

Inbox (0)
Compose Mail
Sent Items

To	: admin@pms.com
From	: sujin@pms.com
Subject	:
Content	:

Send Clear


Project Leader:View Profile
localhost:1080/pms/ProjectLeader/PLViewProfile.aspx
welcome
[illegible]

View Profile

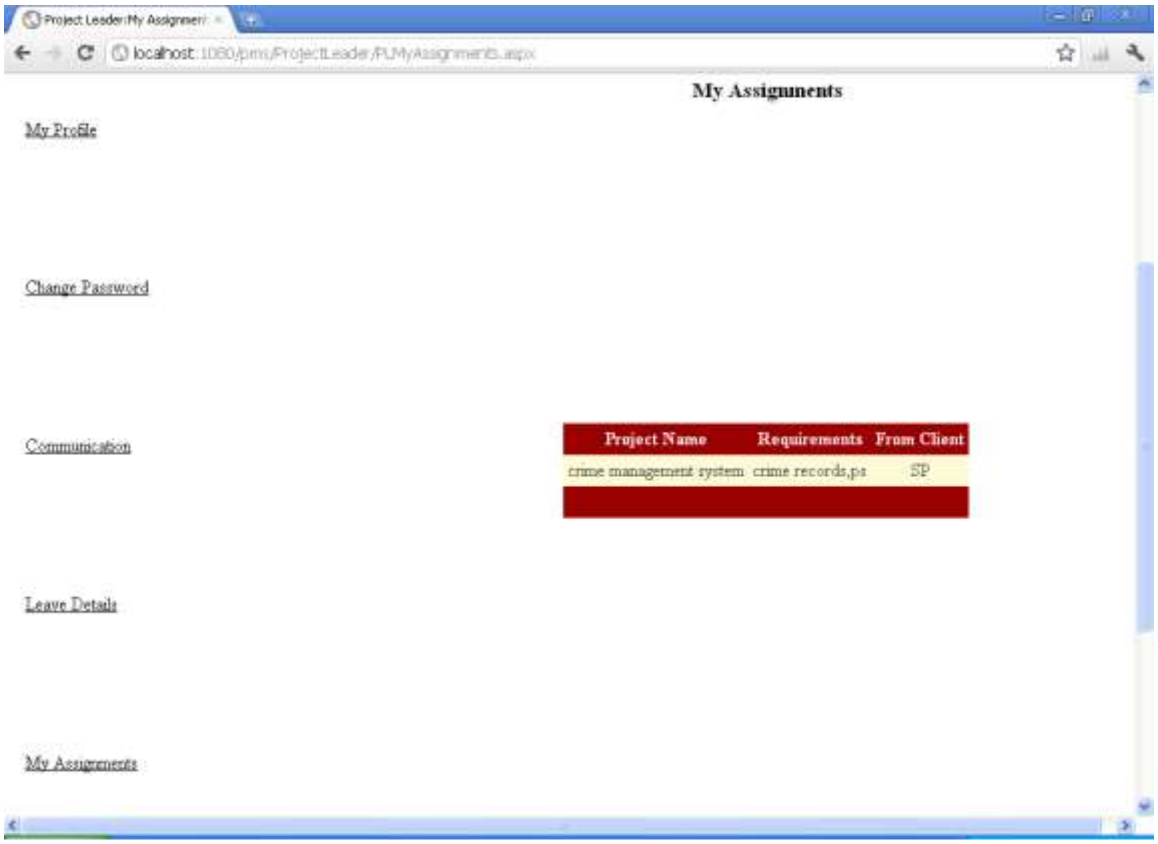
My Profile

Change Password

Communication

Leave Details

Name	sujanmon
Phone No	9087654321
Address	6754675y6eythg
Mail Id	sujan@pms.com
Qualification	BCA



Project Leader:My Assignment
localhost:1080/pms/ProjectLeader/PLMyAssignments.aspx

My Assignments

My Profile

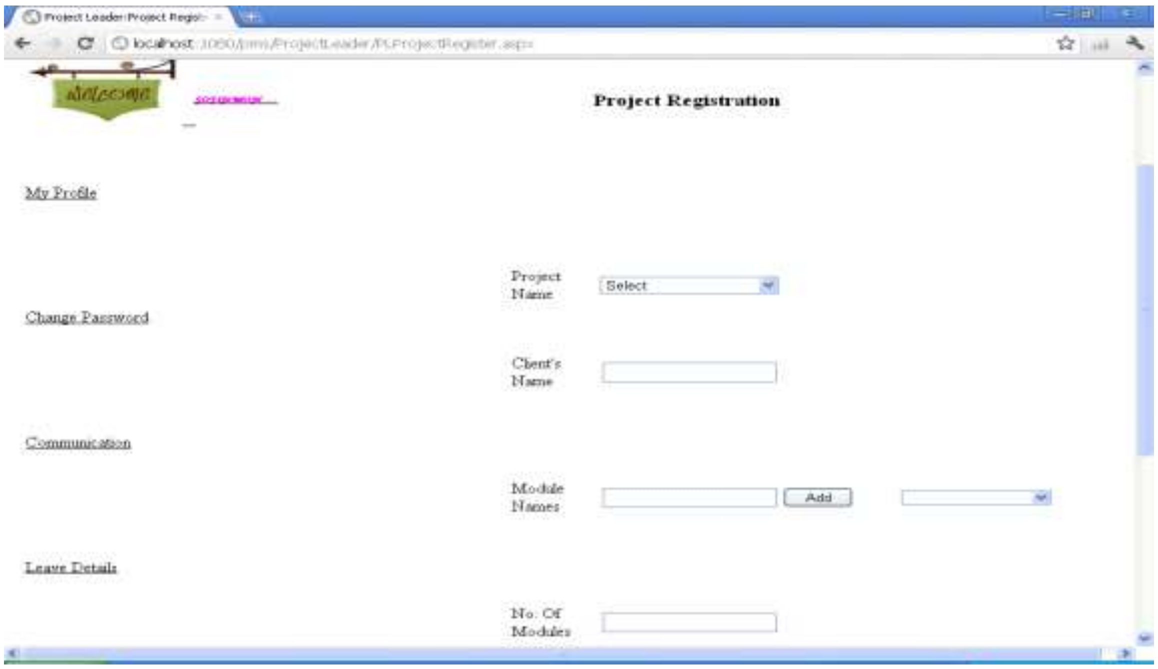
Change Password

Communication

Leave Details

My Assignments

Project Name	Requirements	From Client
crime management system	crime records,ps	SP



Project Leader:Project Regist
localhost:1080/pms/ProjectLeader/PLProjectRegister.aspx
Welcome
Add

Project Registration

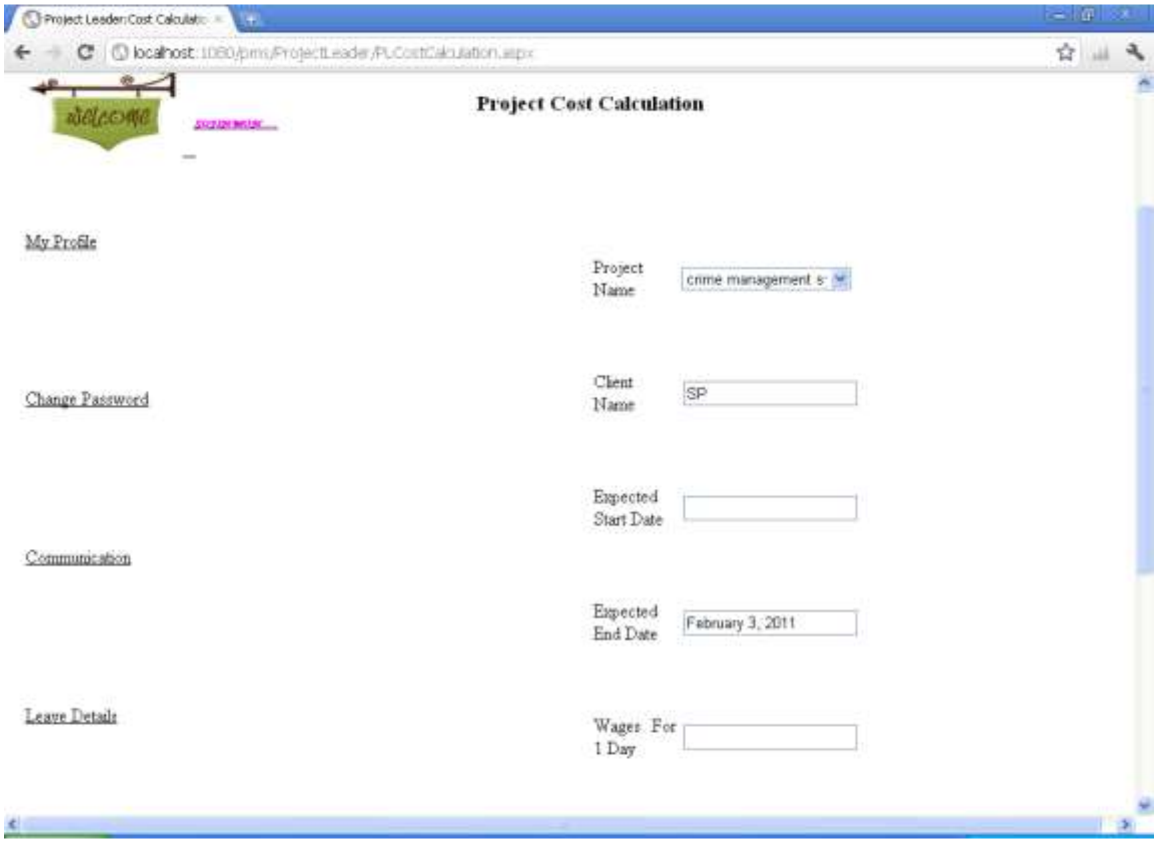
My Profile

Change Password

Communication

Leave Details

Project Name	Select
Client's Name	
Module Names	
No. Of Modules	


Project Leader:Cost Calculatio

localhost:1080/pms/ProjectLeader/PLCostCalculation.aspx

Project Cost Calculation

My Profile

Change Password

Communication

Leave Details

Project Name	crime management s
Client Name	SP
Expected Start Date	
Expected End Date	February 3, 2011
Wages For 1 Day	

CHAPTER 9
CONCLUSION

9.1 SYSTEM IMPLEMENTATION

Implementation is the stage in the project where the theoretical design is turned into a working system. The implementation phase constructs, installs and operates the new system. The most crucial stage in achieving a new successful system is that it will work efficiently and effectively.

MODULE DESCRIPTION

ADMINISTRATOR

This is one of the important modules of the project management system. It basically handles administrative task like addition of a new employee, creation of new project, assigning a project to the available employee, modification of employee details and others. This module is available to the administrator project manager levels only.

When a administrator log on to the site, he enters to his menu page. This page will have the following links:

k. Employee details
l. Leave details
m. Employee registration
n. List project
o. PL selection for project
p. Edit employee profile
q. Notify client
r. Submitted projects
s. Communication
t. Log out

DEVELOPER

Developer module contains all the activities of employers working in that company. All employers are registered as developer at first. According to their skill set some of them are assigned as project leaders. This is done by the administrator. This is done for the registered projects according to the developer’s skill sets and the requirements of the project.

When a developer log on to the site, he enters to his menu page. This page will have the following links:

a. My profile
b. Leave details
c. Communication

d. My assignments
e. Log out

PROJECT LEADER

Project leader module contains all the activities that the project leader has to do when he is assigned as the project leader for the project.

When a Project Leader log on to the site, he enters to his menu page. Here there will be a link named "My Task As PL". By selecting this link he will get a page which contains links for all the works he has to do as a project leader.

This page will have the following links:

a. Register project
b. Cost calculation
c. Change password
d. Communication
e. New Assignment
f. My Assignment
g. My Task

CLIENT

A client is an external person who registers in the company site for giving some project to the company to do according to his needs. A client can register in the company's website by selecting the link given in the main page.

When a client log on to the website, he enters to his menu page. It will contain the following links:

c. Register project
a. Communication
b. Modification

IMPLEMENTATION DETAILS

Implementation is the stage of the project where the theoretical design is turned into a working system. At this stage the main work load, the greatest upheaval and the major impact on the existing system shifts to the user department. If the implementation is not carefully planned and controlled, it can cause chaos and confusion.

Implementation includes all those activities that take place to convert from the old system to new system. The new system may be totally new, replacing an existing manual or automated system or it may be a major modification to an existing system. Proper implementation is essential to provide a reliable system to meet the organization requirements. Successful implementation may not guarantee improvement in the organization using the new system, but improper installation will prevent it.

The implementation stage involves the following tasks.

- Careful planning.
- Investigation of system and constrains.
- Design of methods to achieve the changeover.
- Training of staff in the changeover phase.
- Evaluation of the changeover method.

The method of implementation and the time scale to be adopted are found out initially. Next the system is tested properly and the same time users are trained in the new procedures.

Implementation Procedures

Implementation of software refers to the final installation of the package in its real environment, to the satisfaction of the intended users and the operation of the system. In many organizations someone who will not be operating it, will commission the software development project. In the initial stage, they doubt about the software but we have to ensure that the résistance does not built up as one has to make sure that

- The active user must be aware of the benefits of using the system.
- Their confidence in the software is built up.
- Proper guidance is imparted to the user so that he is comfortable in using the application.

Before going ahead and viewing the system, the user must know that for viewing the result, the server program should be running in the server. If the server object is not up running on the server, the actual processes won't take place.

Implementation Plan

Implementation includes all those activities that take place to convert from the old system to the new one. The new system may be totally new, replacing an existing manual or automated system. Proper implementation is essential to provide a reliable system to meet organization requirements. Successful implementation may not guarantee improvement in the organization using the new system, but improper installation will prevent it.

The process of putting the developed system in actual use is called system implementation. This includes all those activities that take place to convert from the old system to the new system. The system can be implemented only after thorough testing is done and if it is found to be working according to the specifications. The system personnel check the feasibility of the system. The implementation stage involves following tasks.

- Investigation of system and constraints.

- Design of methods to achieve the changeover.
- Training of the staff in the changeover phase.
- Evaluation of the changeover method.

The newly proposed system is implemented after the successful testing of the system.

Post Implementation Review

The final step of the systems approach recognizes that an implemented solution can fail to solve the problem for which it was developed. The results of implementing a solution should be monitored and evaluated. This is called post implementation review process, since the success of a solution is reviewed after it is implemented. The focus on this step, i.e. to determine if the implemented solution has indeed helped the firm and selected business units, meet their system objectives

9.2 CONCLUSION

Anything cannot be ended a single step. It is the fact that nothing is permanent in this world. Almost every project is subjected to change; there is always a scope for further enhancements. The system and the architecture of the proposed system is a compatible one, so addition of new modules can be done without much difficulty. Since the module has its unique properties it can extend further to make this system a complete one.

The aim and objective within this project can all somehow associate and work for the base of the main question i.e., how to manage the Project in an Organization. The software was implemented and tested and found to be error free. Also, the system is protected from any unauthorized changes. All the necessary validations are carried out on this project. The system is designed and developed in Asp.Net platform with C#. Database is needed in this project. The database used is MS SQL Express. This is also cheaper than any other similar system.

9.3 FUTURE ENHANCEMENT

Software product enhancement may involve providing new functional capabilities, improving user displays and modes of integration, updating external documents and internal documentation or updating the performance characteristics of a system.

The Scope of the future development is endless as we would keep updating the project with latest technologies available.

The software that we have developed is compatible with every .NET platform, so any user can perform further development on it. **Software Tracking System** is simple to use and it is possible to add additional features to the system. Now the system is developed in windows based and in future it can be developed in Linux based.

REFERENCES

Hand Books

1 Professional C# 2010, Wrox publication.

2 Programming in C#.Net, Julia Case Bradely

3 System Analysis and Design, Awad, Elias M, Galgothia
   Publications (P) Ltd, 2010.

4 Software Engineering- A Practical Approach, Pressman, Roger S,
   McGraw-Hill Publishing Company Ltd, Ed.5

5 Visual C#.Net, Elliotte Rusty Haold

Websites

1 www.csharpcorner.com

2 www.codeproject.com

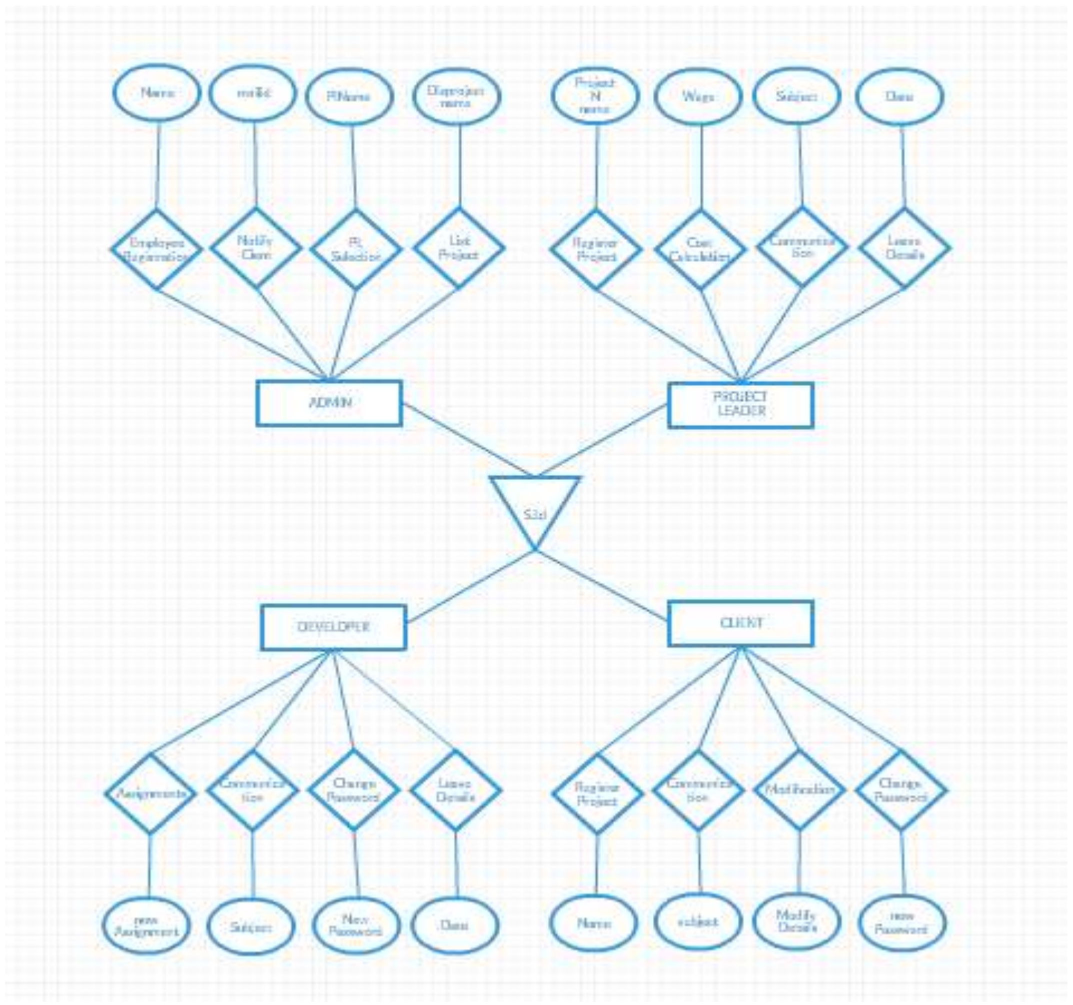
3 www.thescripts.com

4 www.samplesgotdotnet.com

5www.msdn2.microsoft.com

APPENDICES

1. SCRUM BOARD



2. LIST OF TABLES

1. Table Name : ClientReg

Primary Key : ClientId

Purpose : Stores the Client Registration

Field Name	**Type**	**Size**	**Description**
ClientId	int	4	Client Id
ClientName	Varchar	25	Client Name
ClientMailId	Varchar	25	Mail Id
ClientCompanyId	Varchar	25	CompanyId
ClientPhoneNumber	Varchar	25	Phone Number
ClientAddress	Varchar	25	Address
UserName	Varchar	25	Username

Client Registration

2. Table Name : empreg

Primary Key : EmpId

Purpose : Stores the details of employee

Field Name	**Type**	**Size**	**Description**
EmpId	Int	4	Employee Id
EmpName	Varchar	25	Employee Name
EmpAddress	Varchar	MAX	Address
EmpDob	Datetime	8	Date of Birth
EmpPhoneNumber	Varchar	25	Phone Number
EmpQualification	Varchar	25	Qualification
EmpDesignation	Varchar	25	Designation
EmpSkills	Varchar	25	SkillSet

EmpPic	Varchar	MAX	Picture
EmpUsername	Varchar	25	UserName
EmpStatus	Varchar	25	Status
EmpCompanyMailId	Varchar	25	CompanyMailId
ProjectDeadLine	Datetime	8	ProjectDeadLine

Employee Profile

3. Table Name : EmpStatus

Foreign Key : EmpId

Purpose : Stores the status of the Employee

Field Name	**Type**	**Size**	**Description**
EmpId	Int	4	Employee Id
EmpName	Varchar	25	Employee Name
ProjectName	Varchar	25	Project Name
ModuleName	Varchar	25	Module Name
StartDate	datetime	8	Start Date
EndDate	Datetime	8	End Date
CurrentDate	Datetime	8	Current Date
To	Varchar	25	To
Status	Varchar	25	Status

Employee Status

4. Table Name : Leave

Primary Key : Slno

Foreign Key : EmpId

Purpose : Stores the Leave Details

Field Name	**Type**	**Size**	**Description**

Slno	Int	4	Serial Number
EmpId	Int	4	Employee Id
UserName	Varchar	25	UserName
EmpName	Varchar	25	Employee Name
MailId	Varchar	25	Mail Id
Days	Int	4	No: of Days
Frm	Datetime	8	From
Status	Varchar	25	Status

Leave

5. Table Name : Login

Primary Key : UserName

Purpose : stores the data for login

Field Name	**Type**	**Size**	**Description**
Username	Varchar	25	UserName
Password	Varchar	25	Password
Designation	Varchar	25	Designation

Login

6. Table Name : MailDetails

Primary Key : Id

Purpose : Stores the Mail details

Field Name	**Type**	**Size**	**Description**
Id	Int	4	Employee Id
FromId	Varchar	25	From Id
ToId	Varchar	25	To Id
Subject	Varchar	MAX	Subject

Contents	Varchar	MAX	Content
MDate	Datetime	8	Date
MailStatus	Bit	_	Status

MailDetails

7. Table Name : ModuleNames

Primary Key : Id

Purpose : Stores the Modulenames

Field Name	**Type**	**Size**	**Description**
Id	Int	4	Employee Id
ProjectName	Varchar	25	Project Name
ModuleName	Varchar	25	Module Name

ModuleNames

8. Table Name : ProjectRegPL

Primary Key : Id

Purpose : Stores the Project registration details of PL

Field Name	**Type**	**Size**	**Description**
Id	Int	4	Id
ProjectName	Varchar	25	Project Name
ClientName	Varchar	25	Client Name
PLName	Varchar	25	Project Leader Name
NoOfModules	Int	4	No Of Modules
NoOfPeople	Int	4	No Of People

Projectregpl

9. Table Name : RegProject

Primary Key : **Id**

Foreign Key : **ClientId**

Purpose : **Stores the Project Registration Details**

Field Name	**Type**	**Size**	**Description**
ProjectId	Int	4	Id
ProjectName	Varchar	25	Project Name
CUserName	varchar	25	User Name
ClientCompanyId	Varchar	25	ClientCompanyId
EndDate	Datetime	8	End Date
Requirements	Varchar	MAX	Requirements
Status	Varchar	25	Status

Regproject

10. Table Name : TaskRegPL

Primary Key : Id

Foreign Key : EmpId

Purpose : stores the task of the Project Leader

Field Name	**Type**	**Size**	**Description**
Id	Int	4	Id
ProjectName	Varchar	25	Project Name
EmpUserName	Varchar	25	User Name
ClientName	Varchar	25	Client Name
EndDate	Datetime	8	End Date
Requirements	Varchar	MAX	Requirements

Taskregpl

11. **Table Name : ProjectCost**

Primary Key : Id

Purpose : strores the cost of the project

Field Name	**Type**	**Size**	**Description**
Id	Int	4	Id
ProjectName	Varchar	25	Project Name
ClientName	Varchar	25	Client Name
StartDate	Datetime	8	Expected Start Date
EndDate	Datetime	8	Expected End Date
TeamSize	Int	4	No Of Workers
TotalCost	Varchar	25	Total Cost

ProjectCost

12. **Table Name : ClientModification**

Primary Key : Id

Foreign Key : clientId

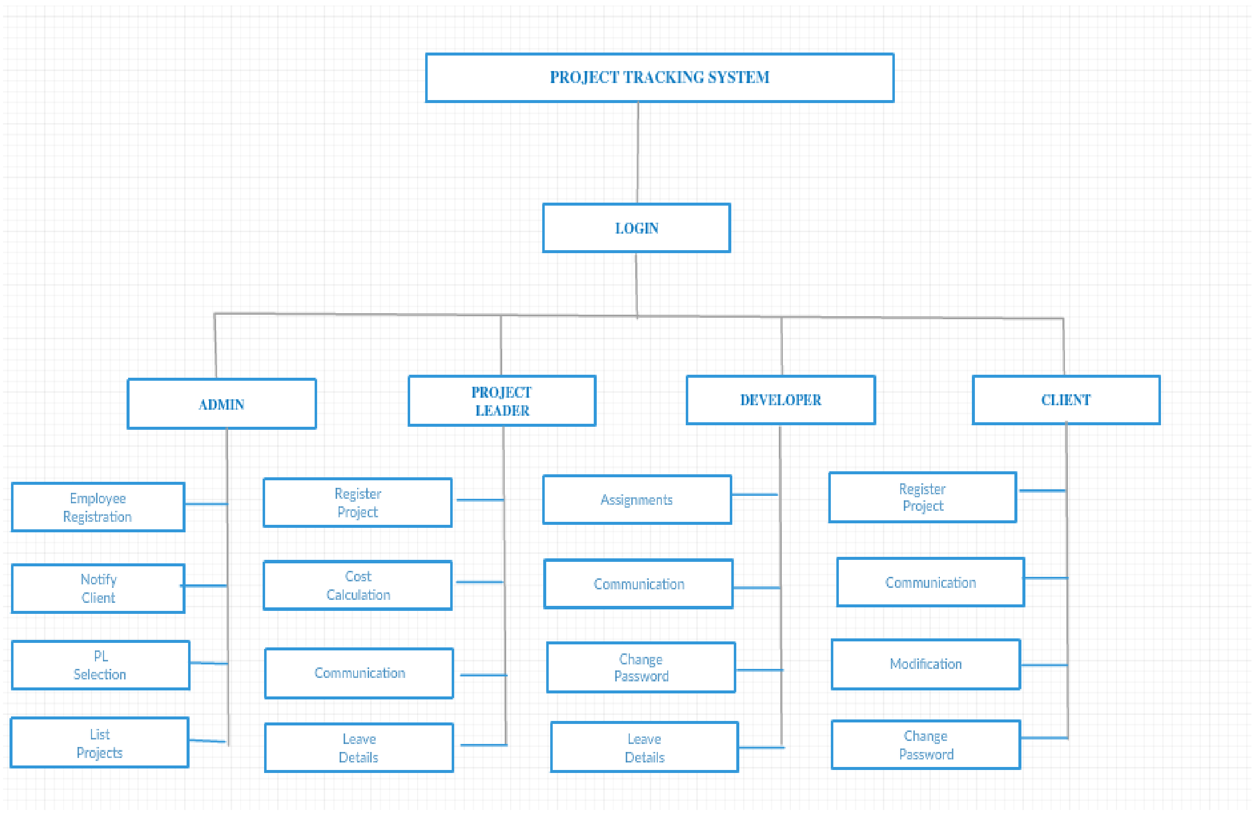
Purpose : stores the modifications of the project

Field Name	**Type**	**Size**	**Description**
Id	Int	4	Id
ClientId	Varchar	25	Client Id
ClientName	Varchar	25	Client Name
ProjectName	Datetime	8	Project Name
NewFeatures	Datetime	8	New Features
Username	Varchar	25	UserName

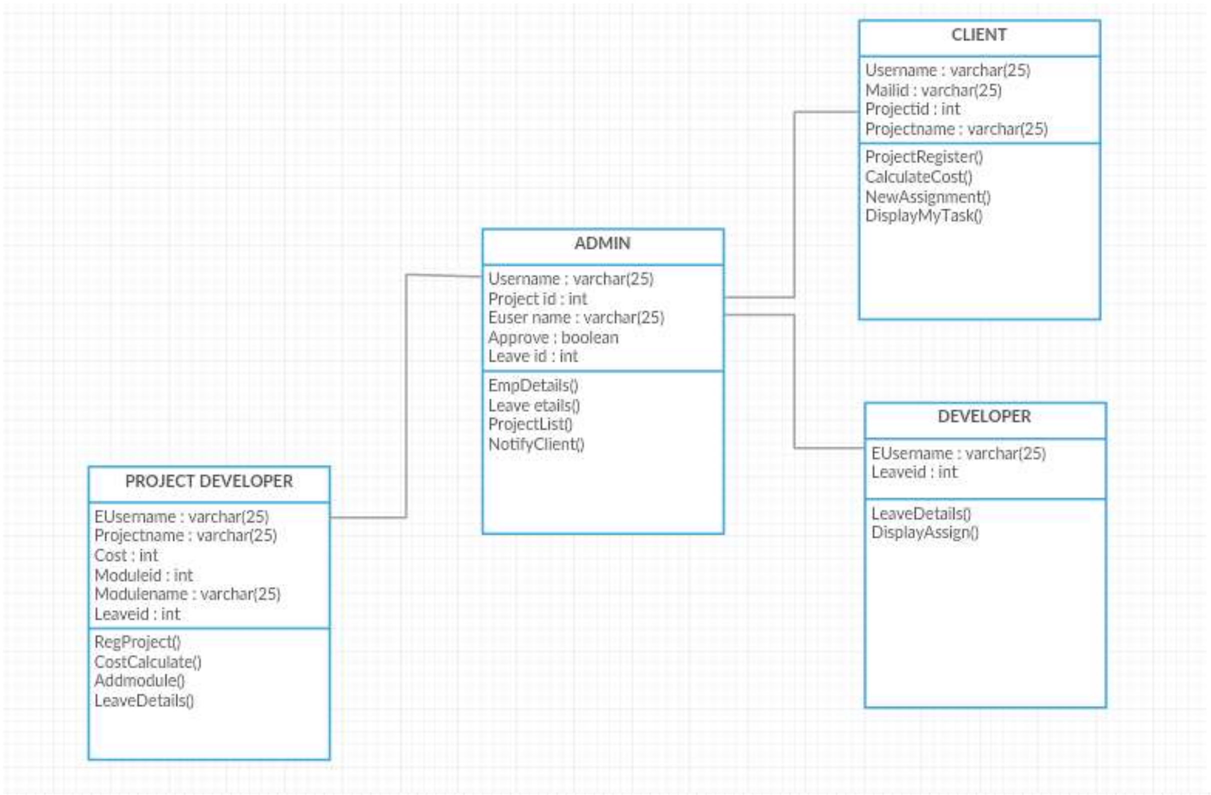
ClientModification

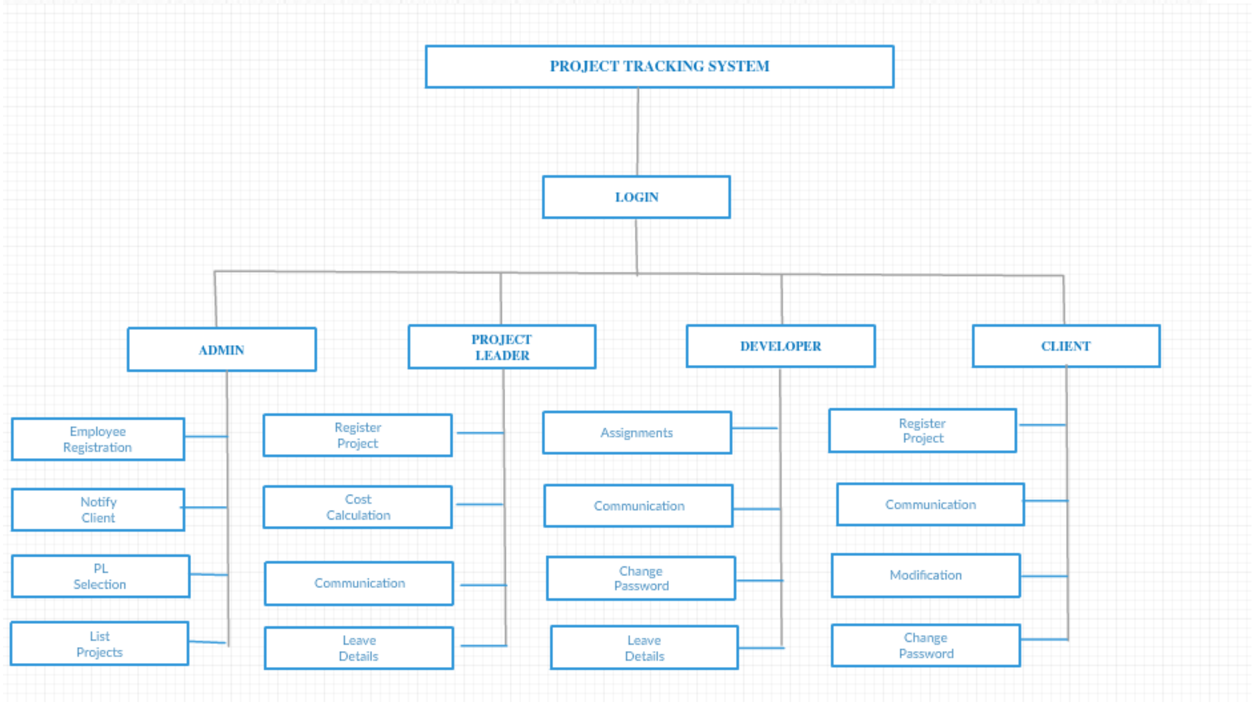
3. LIST OF FIGURES

MENU TREE

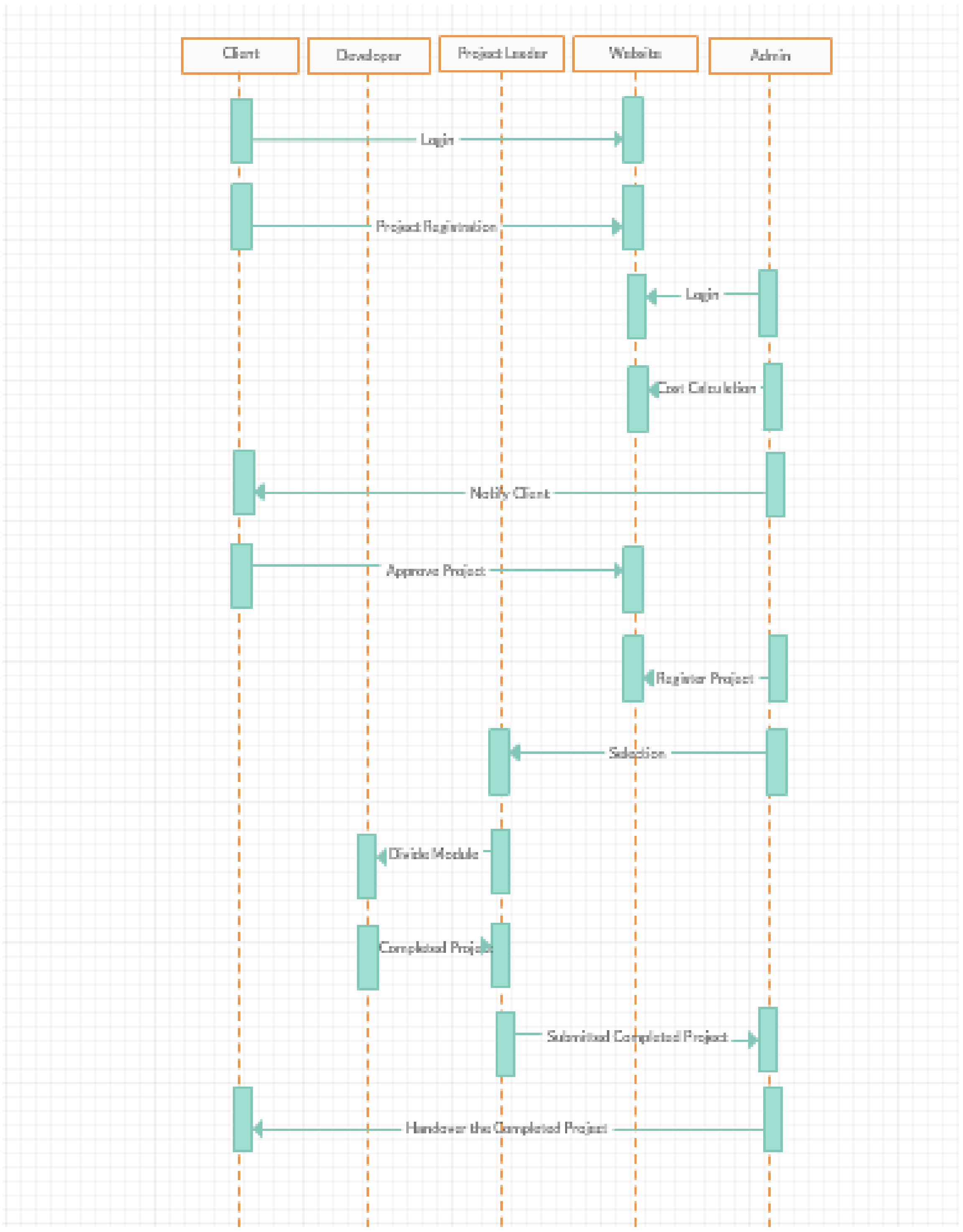


CLASS DIAGRAM





SEQUENCE DIAGRAM



4. CODING

CLIENTREGISTRATION.ASPX.CS

```
Databaseclass con = new Databaseclass();
SqlDataReader rdr;
   protected void btnavilability_Click(object sender, EventArgs e)
   {
       //To check UserName is avilabile or not
       if (String.IsNullOrEmpty(txtusername.Text))
       {
           lblavilabile.Visible = true;
           lblavilabile.ForeColor = System.Drawing.Color.Red;
           lblavilabile.Text = "Enter User Name";
       }
       else
       {
           string sql2 = "select UserName from Login where UserName='" + txtusername.Text + "'";
           rdr = con.select(sql2);
           if (rdr.Read())
           {
               lblavilabile.Visible = true;
               lblavilabile.ForeColor = System.Drawing.Color.Red;
               lblavilabile.Text = "UserName already exists......";
           }
           else
           {
               lblavilabile.Visible = true;
               lblavilabile.ForeColor = System.Drawing.Color.Red;
               lblavilabile.Text = "UserName AVAILABLE.....";
           }
       }

    }
   protected void btnSubmit_Click(object sender, EventArgs e)
   {
       try
       {
           string sql2 = "select UserName from Login where UserName='" + txtusername.Text + "'";
           rdr = con.select(sql2);
           if (rdr.Read())
           {
               lblavilabile.Text = "UserName already exists......";
           }
           else
           {
               if (String.IsNullOrEmpty(txtname.Text) || String.IsNullOrEmpty(txtemailid.Text) ||
String.IsNullOrEmpty(txtphoneno.Text) || String.IsNullOrEmpty(txtaddress.Text) ||
String.IsNullOrEmpty(txtusername.Text) || String.IsNullOrEmpty(txtpassword.Text) ||
String.IsNullOrEmpty(txtreenterpassword.Text))
               {
                   lblerrormessage.ForeColor = System.Drawing.Color.Red;
                   lblerrormessage.Text = "* Required Fields";
               }
               else
               {
                   string sql = "insert into ClientReg
(ClientName,ClientMailId,ClientCompanyId,ClientPhoneNo,ClientAddress,CUserName) values ('" +
txtname.Text + "','" + txtemailid.Text + "','" + txtusername.Text + "@pms.com" + "','" + txtphoneno.Text + "','"
+ txtaddress.Text + "','" + txtusername.Text + "')";
                   con.iducommand(sql);
                   string sql1 = "insert into Login values('" + txtusername.Text + "','" + txtpassword.Text + "','CLNT')";
                   con.iducommand(sql1);
                   Response.Redirect("RegCompletedSucessfully.aspx");
               }
           }
       }
       catch (Exception ex)
       {
           Response.Write(ex.Message);
       }
       finally
       {

       }
   }

   protected void btnReset_Click(object sender, EventArgs e)
   {
       //To Clear all the Fileds
       txtname.Text = " ";
       txtemailid.Text = " ";
       txtphoneno.Text = " ";
       txtaddress.Text = " ";
       txtusername.Text = " ";
       txtpassword.Text = " ";
       txtreenterpassword.Text = " ";
       lblavilabile.Visible = false;

   }
   protected void btnCancel_Click(object sender, EventArgs e)
   {
       //Go back to login page
       Response.Redirect("Login.aspx");
   }
```

LOGIN.ASPX.CS

```
Databaseclass con = new Databaseclass();
    SqlDataReader rdr;
    string sql;

    protected void btnsignup_Click(object sender, EventArgs e)
    {
        Response.Redirect("Signup.aspx");
    }
    protected void btnlogin_Click(object sender, EventArgs e)
    {
        try
        {
            //to check username and password
            sql = "select Designation from Login where UserName='" + txtloginusername.Text + "' and Password='"
+ txtloginpassword.Text + "'";
            rdr = con.select(sql);
            if (rdr.Read())
            {
                string Desi = rdr.GetString(0);
                //store username is session
                Session["username"] = txtloginusername.Text;
                if (Desi == "ADMIN")
                {
                    Response.Redirect("~/Administrator/AdminHome.aspx");
                }
                else if (Desi == "CLNT")
                {
                    Response.Redirect("~/Client/ClientHome.aspx");
                }
                else if (Desi == "PL")
                {
                    Response.Redirect("~/ProjectLeader/PLHome.aspx");
                }
                else if (Desi == "DVPL")
                {
                    Response.Redirect("~/Developer/DeveloperHome.aspx");
                }

            }
            else
            {
                //to change the forecolor of label
                Label1.ForeColor = System.Drawing.Color.Red;
                Label1.Text = "Invalid Username or Password";
            }

        }
        catch (Exception ex)
        {
            //To display System defined error
            Response.Write(ex.Message);

        }
        finally
        {

        }
    }
```

REGISTERPROJECT.ASPX.CS

```
Databaseclass con=new Databaseclass();
    SqlDataReader rdr;
    string sql;
    protected void Page_Load(object sender, EventArgs e)
    {
        //To Display username nad company id directly
        txtusername.Text = Session["username"].ToString();
       sql = "Select ClientName,ClientCompanyId from ClientReg where CUserName='" + txtusername.Text    +
"'";
            rdr = con.select(sql);
            if (rdr.Read())
            {
                txtclientname.Text = rdr.GetString(0);
                txtclientcompanyid.Text = rdr.GetString(1);
        }
    }
    protected void btnsubmit_Click(object sender, EventArgs e)
    {
        //To check the filed null or not
        if (String.IsNullOrEmpty(txtusername.Text) || String.IsNullOrEmpty(txtclientname.Text) ||
String.IsNullOrEmpty(txtclientcompanyid.Text) || String.IsNullOrEmpty(txtprojectname.Text) ||
String.IsNullOrEmpty(txtenddate.Text) || String.IsNullOrEmpty(txtrequirments.Text))
        {
            lblrequired.Visible = true;
            lblrequired.ForeColor = System.Drawing.Color.Red;
            lblrequired.Text = "* Are reqired fileds";
        }
        else
        {
            string sql2 = "select ProjectName from RejProjects where ProjectName = '" + txtprojectname.Text + "' ";
            rdr = con.select(sql2);
            if (rdr.Read())
            {
                lblpronameerror.Visible = true;
                lblpronameerror.ForeColor = System.Drawing.Color.Red;
                lblpronameerror.Text = "Project Name Already Exist Pls Select Another Name";
            }
            else
            {
                try
                {
                    //To check entered date is previous or current
                    string date = DateTime.Today.ToString();
                    DateTime dtStart = DateTime.Parse(txtenddate.Text);
                    DateTime dtEnd = DateTime.Parse(date);
                    if (DateTime.Compare(dtStart, dtEnd) == 0)
                    {
                        lbldatemsg.Visible = true;
                        lbldatemsg.ForeColor = System.Drawing.Color.Red;
                        lbldatemsg.Text = "End date cannot be Today";
                    }
                        // To compare start date and end date
                    else if (DateTime.Compare(dtStart, dtEnd) < 0)
                    {
                        lbldatemsg.ForeColor = System.Drawing.Color.Red;
                        lbldatemsg.Text = "End date cannot be A Previous Date";
                    }
                    else
                    {
                        lblpronameerror.Visible = false;
                        string sql1 = "insert into RejProjects
(ProjectName,CUserName,ClientCompanyId,EndDate,Requirements,Status) values ('" + txtprojectname.Text +
"','" + txtusername.Text + "','" + txtclientcompanyid.Text + "','" + txtenddate.Text + "','" + txtrequirments.Text +
"','Pending')";
                        con.iducommand(sql1);
                        Response.Redirect("RegisterProjectSucessfully.aspx");
                    }
                }
                catch (Exception ex)
                {
                    Response.Write(ex.Message);
                    //lblrequired.ForeColor = System.Drawing.Color.Red;
                    //lblrequired.Text = "Operation failed try again";
                }
                finally
                {
                }
            }
        }
    }
    protected void btncancel_Click(object sender, EventArgs e)
    {
        Response.Redirect("ClientHome.aspx");
    }
```

EMPLOYEEREGISTRATION.ASPX.CS

```
    Databaseclass con = new Databaseclass();
    SqlDataReader rdr;
    string picturepath;
    protected void Page_Load(object sender, EventArgs e)
    {


    }
    protected void btnClear_Click(object sender, EventArgs e)
    {
        txtname.Text = " ";
        txtusername.Text = " ";
        txtpassword.Text = " ";
        txtphoneno.Text = " ";
        txtaddress.Text = " ";
        txtdob.Text = " ";
        lblavilabile.Visible = false;
    }
    protected void btnsubmit_Click(object sender, EventArgs e)
    {
        try
        {
            if (String.IsNullOrEmpty(txtname.Text) || String.IsNullOrEmpty(txtusername.Text) ||
String.IsNullOrEmpty(txtpassword.Text) || String.IsNullOrEmpty(txtphoneno.Text) ||
String.IsNullOrEmpty(txtaddress.Text) || String.IsNullOrEmpty(txtdob.Text) ||
String.IsNullOrEmpty(drpqualification.Text) || String.IsNullOrEmpty(drpdesg.Text) ||
String.IsNullOrEmpty(drpskillset.Text))
            {
                lblrequired.Visible = true;
                lblrequired.ForeColor = System.Drawing.Color.Red;
                lblrequired.Text = "* Required Fields";
            }
        else
```

```
{
string sql = "select UserName from Login where UserName='" + txtusername.Text + "'";
rdr = con.select(sql);
if (rdr.Read())
{
   lblavilabile.ForeColor = System.Drawing.Color.Red;
   lblavilabile.Text = "UserName already exists......";
}
else
{
   if (txtusername.Text == "")
   {
      lblavilabile.Visible = true;
      lblavilabile.ForeColor = System.Drawing.Color.Red;
      lblavilabile.Text = "Enter Username...";
   }
   else
   {
      lblavilabile.Visible = true;
      lblavilabile.ForeColor = System.Drawing.Color.GreenYellow;
      lblavilabile.Text = "UserName Avilabile.....";
      //string pic = FileUpload1.FileName.ToString();
      //FileUpload1.SaveAs(Server.MapPath("~/Images/Employees/" + pic));
      //string picture = "~/Images/Employees/" + pic;
      pictureupload();
      string sql1 = "insert into
EmployeeReg(EmpName,EmpAddress,EmpPic,EmpDob,EmpPhoneNo,EmpQualification,EmpDesignation,Em
pSkills,EmpUserName,EmpStatus,EmpCompanyMailId) values ('" + txtname.Text + "','" + txtaddress.Text +
"','"+picturepath+"','" + txtdob.Text + "','" + txtphoneno.Text + "','" + drpqualification.SelectedItem.ToString() +
"','" + drpdesg.SelectedItem.ToString() + "','" + drpskillset.SelectedItem.ToString() + "','" + txtusername.Text +
"','Avilabile','" + txtusername.Text + "@pms.com" + "')";
      con.iducommand(sql1);
      string sql2 = "insert into Login values('" + txtusername .Text + "','" + txtpassword .Text +
"','"+drpdesg.SelectedItem.ToString ()+"')";
      con.iducommand(sql2);
```

```
                Response.Redirect("EmployeeRegSucessfully.aspx");

            }
        }


        }


    }
    catch (Exception ex)
    {
        Response.Write(ex.Message);
    }
    finally
    {
    }
}
private void pictureupload()
{
    String path = Server.MapPath("~/Images/Employees/");
    if (FileUpload1.HasFile)
    {
        String extension = System.IO.Path.GetExtension(FileUpload1.FileName).ToLower();
        if (extension == ".gif" || extension == ".png" || extension == ".jpg")
        {
            FileUpload1.SaveAs(path + FileUpload1.FileName);
            picturepath = "~/Images/Employees/" +  FileUpload1.FileName;
        }
        else
            Response.Write("Only upload gif,png or jpg file");
    }
}

protected void btnavilabile_Click(object sender, EventArgs e)
{
    string sql = "select UserName from Login where UserName='" + txtusername.Text + "'";
```

```
        rdr = con.select(sql);
        if (rdr.Read())
        {
            lblavilabile.Visible = true;
            lblavilabile.ForeColor = System.Drawing.Color.Red;
            lblavilabile.Text = "UserName already exists......";
        }
        else
        {
            if (txtusername.Text == "")
            {
                lblavilabile.Visible = true;
                lblavilabile.ForeColor = System.Drawing.Color.Red;
                lblavilabile.Text = "Enter Username...";
            }
            else
            {
                lblavilabile.Visible = true;
                lblavilabile.ForeColor = System.Drawing.Color.GreenYellow;
                lblavilabile.Text = "UserName Avilabile.....";
            }
        }
    }
```

EMPLOYEEDETAILS.ASPX.CS

```
Databaseclass con = new Databaseclass();
    SqlDataReader rdr;
    string sql;
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            empdetails();
        }
    }
    private void empdetails()
    {
        sql = "select EmpUserName,EmpName,EmpAddress,EmpPic,convert(varchar(10),EmpDob,103) as
EmpDob,EmpPhoneNo,EmpQualification,EmpDesignation,EmpSkills,EmpStatus from EmployeeReg";
        rdr = con.select(sql);
        grdempdetails.DataSource = con.displaydata(sql);
        grdempdetails.DataBind();

        sql = "select Empusername,EmpPic from EmployeeReg";
        rdr = con.select(sql);
        GridView1.DataSource = con.displaydata(sql);
        GridView1.DataBind();

    }

    protected void grdempdetails_RowCommand(object sender, GridViewCommandEventArgs e)
    {
        if (e.CommandName == "deleterow")
        {
            sql = "delete from EmployeeReg where EmpUserName='" + e.CommandArgument.ToString() + "'";
            con.iducommand(sql);
            empdetails();
            Response.Redirect("EmployeeDetails.aspx");
        }
    }
```

LISTPROJECTS.ASPX.CS

```
Databaseclass con = new Databaseclass();
    SqlDataReader rdr;
    string sql;
    protected void Page_Load(object sender, EventArgs e)
    {
        sql = "select ProjectName,CUserName,ClientCompanyId,EndDate,Requirements,Status from RejProjects";
        rdr = con.select(sql);
        grdprojectlist.DataSource = con.displaydata (sql);
        grdprojectlist.DataBind();
    }
PLSELECTION.ASPX
Databaseclass con = new Databaseclass();
    string sql,sql1;
    protected void Page_Load(object sender, EventArgs e)
    {
        sql = "select count(*) from EmployeeReg where EmpDesignation='PL' and EmpStatus='Avilabile' ";
        int count = con.scalar(sql);
        if (count > 0)
        {
            sql1 = "select * from EmployeeReg where EmpDesignation='PL' and EmpStatus='Avilabile' ";
            grdpllist.DataSource = con.displaydata(sql1);
            grdpllist.DataBind();
        }
        else
        {
            lblerror.Visible = true;
            lblerror.ForeColor = System.Drawing.Color.Red;
            lblerror.Text = "No Project Leaders Available";
        }
    }
    protected void grdpllist_SelectedIndexChanging(object sender, GridViewSelectEventArgs e)
    {

        Session["empname"] = grdpllist.Rows[e.NewSelectedIndex].Cells[0].Text;
        Response.Redirect("PlTaskSet.aspx");
    }
```

PROJECTDETAILS.ASPX.CS

```
Databaseclass con = new Databaseclass();
    SqlDataReader rdr;
    string sql, sql1;
    protected void Page_Load(object sender, EventArgs e)
    {
        lblclientname.Visible = false;
        lblprojectleader.Visible = false;
        lblmsg.Visible = false;
         if (!IsPostBack)
        {
            sql = "select  ProjectName from PlTaskReg";
            drpprojectname.Items.Clear();
            rdr=con.select(sql);
            drpprojectname.Items.Add("select");

            if(rdr.HasRows)
            {
                while (rdr.Read())
                {
                    drpprojectname.Items.Add(rdr["ProjectName"].ToString());

                }
                rdr.Close();
            }
            else
            {
                lblmsg.Text = "No Project Details from Project Leaders";
            }
        }
    }
    protected void drpprojectname_SelectedIndexChanged(object sender, EventArgs e)
    {
        sql = "select ClientName,EmpName from PlTaskReg where ProjectName='" + 
drpprojectname.SelectedItem.ToString() + "'";
        rdr = con.select(sql);
        if (rdr.Read())
        {
            lblclientname.Visible = true;
            lblprojectleader.Visible = true;
            lblclientname.Text = rdr.GetString(0);
            lblprojectleader.Text = rdr.GetString(1);
        }
        sql1 = "select DeveloperUserName from ModuleNames where ProjectName='" + 
drpprojectname.SelectedItem.ToString() + "'";
        ListBox1.Items.Clear();
        rdr = con.select(sql1);
        if (rdr.HasRows)
        {
            while (rdr.Read())
            {
                ListBox1.Items.Add(rdr["DeveloperUserName"].ToString());

            }
            rdr.Close();
        }
    }
```

NOTIFYCLIENT.ASPX.CS

```
Databaseclass con = new Databaseclass();
SqlDataReader rdr;
string sql;
    protected void Page_Load(object sender, EventArgs e)
    {
        pnlcompose.Visible = true;
        lblcomposefrom.Text = "admin@pms.com";
        clientnamedisplay();
    }
    private void clientnamedisplay()
    {
        sql = "select UserName from Login where Designation ='CLNT' ";
        rdr = con.select(sql);
        while (rdr.Read())
        {
            drpcomposeto.Items.Add(rdr["UserName"].ToString() + "@pms.com");
        }
    }
    protected void btnclear_Click(object sender, EventArgs e)
    {
        txtcomposecontent.Text = "";
        txtcomposesubject.Text = "";
        drpcomposeto.SelectedValue = "Select";
        lblmsg.Visible = false;
        lblmsgto.Visible = false;
    }
    protected void btnsent_Click(object sender, EventArgs e)
    {
        if (String.IsNullOrEmpty(txtcomposesubject.Text) || String.IsNullOrEmpty(txtcomposecontent.Text))
        {
            lblmsg.Visible = true;
            lblmsg.ForeColor = System.Drawing.Color.Red;
            lblmsg.Text = "* Required Fields";
        }
        else if (drpcomposeto.SelectedValue == "Select")
        {
            lblmsgto.Visible = true;
            lblmsgto.ForeColor = System.Drawing.Color.Red;
            lblmsgto.Text = "Pls Select ToId";
        }
        else
        {
            string sent = "insert into MailDetails(FromId,ToId,Subject,Contents,MDate,MailStatus) values('" +
lblcomposefrom.Text + "','" + drpcomposeto.SelectedItem.ToString() + "','" + txtcomposesubject.Text + "','" +
txtcomposecontent.Text + "',getdate(),'1') ";
            con.iducommand(sent);
            lblmsg.Visible = true;
            txtcomposecontent.Text = "";
            txtcomposesubject.Text = "";
            drpcomposeto.SelectedValue = "Select";
            lblmsg.ForeColor = System.Drawing.Color.Green;
            lblmsg.Text = "Mail Sent";
        }
    }
```

COMMUNICATION.ASPX.CS

```
Databaseclass con = new Databaseclass();
    SqlDataReader rdr;
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            pnlinbox.Visible = true;
            pnlsent.Visible = false;
            pnlcompose.Visible = false;
            pnlview.Visible = false;
            inbox();
        }
    }
    void inbox()
    {
        //103 for only display date
        string inbo = "select Id,FromId,ToId,Subject,Contents,convert(varchar(10),MDate,103) as
MDate,MailStatus from MailDetails where ToId='admin@pms.com'";
        //rdr = con.select(inbo);
        grdinbox.DataSource = con.displaydata (inbo);
        grdinbox.DataBind();
        string count = "select count(*) from MailDetails where  ToId='admin@pms.com' and MailStatus=1 ";
        lblinboxstatus .Text = "(" + con.scalar (count).ToString() + ")";

    }
    void sentmail()
    {
        string sent = "select Id,FromId,ToId,Subject,Contents,convert(varchar(10),MDate,103) as MDate,MailStatus from  MailDetails where FromId='admin@pms.com'";
        grdsentbox.DataSource = con.displaydata(sent);
        grdsentbox .DataBind();
        string count = "select count(*) from MailDetails where  FromId='admin@pms.com'";
        lblsentstatus .Text = "(" + con.scalar(count).ToString() + ")";
    }

    protected void lnkbtnsentmail_Click(object sender, EventArgs e)
    {
        pnlsent.Visible = true;
        pnlinbox.Visible = false;
        pnlcompose.Visible = false;
        pnlview.Visible = false;
        sentmail();

    }
    protected void lnkbtninbox_Click(object sender, EventArgs e)
    {
        pnlinbox.Visible = true;
        pnlsent.Visible = false;
        pnlcompose.Visible = false;
        pnlview.Visible = false;
        inbox();
    }
    protected void btnreply_Click(object sender, EventArgs e)
    {
        pnlcompose.Visible = true;
        pnlview .Visible = false ;
        pnlsent.Visible = false;
        pnlinbox.Visible = false;
        lblcomposefrom.Text = "admin@pms.com";
        drpcomposeto.Items.Clear();
        drpcomposeto.Items.Add("Select");
        drpcomposeto.Items.Add(lblviewfrom.Text);

    }



    protected void grdinbox_RowCommand(object sender, GridViewCommandEventArgs e)
    {
        pnlview .Visible = true;
        pnlcompose .Visible = false;
        pnlsent.Visible = false;
        pnlinbox.Visible = false;
        //commmandArgument for check the id is matching or not
        string inbox = "select * from MailDetails where Id=" + int.Parse(e.CommandArgument.ToString());
        rdr = con.select (inbox);
        if (rdr.Read())
        {
            lblviewsubject .Text = rdr["Subject"].ToString();
            lblviewdate .Text = rdr["MDate"].ToString();
            lblviewfrom .Text = rdr["FromId"].ToString();
            lblviewto .Text = rdr["ToId"].ToString();
            lblviewcontent .Text = rdr["Contents"].ToString();
            btnreply .Visible = true;
            //update MailStatus=0 to
            string update = "update MailDetails set MailStatus=0 where Id=" +
int.Parse(e.CommandArgument.ToString());
            con.iducommand(update);
        }

    }
    protected void btnsent_Click(object sender, EventArgs e)
    {
        if (String.IsNullOrEmpty(txtcomposesubject.Text) || String.IsNullOrEmpty(txtcomposecontent.Text))
        {
            lblmsg.Visible = true;
            lblmsg.ForeColor =System.Drawing.Color.Red;
            lblmsg.Text = "* Required Fields";
        }
        else if (drpcomposeto.SelectedValue == "Select")
        {
            lblmsgto.Visible = true;
            lblmsgto.ForeColor = System.Drawing.Color.Red;
            lblmsgto.Text = "Pls Select ToId";
        }
        else
        {
            string sent= "insert into MailDetails(FromId,ToId,Subject,Contents,MDate,MailStatus) values('" +
lblcomposefrom.Text + "','"+drpcomposeto.SelectedItem.ToString ()+"','"+txtcomposesubject.Text
+"','"+txtcomposecontent.Text +"',getdate(),'1') ";
            con.iducommand (sent );
            lblmsg.Visible = true;
            txtcomposecontent .Text = "";
            txtcomposesubject .Text = "";
            drpcomposeto.SelectedValue = "Select";
            lblmsg.ForeColor = System.Drawing.Color.Green;
            lblmsg.Text = "Mail Sent";
        }
    }
    protected void btnclear_Click(object sender, EventArgs e)
    {
        lblmsg.Text = "";
        txtcomposecontent.Text = "";
        txtcomposesubject.Text = "";
        drpcomposeto.SelectedValue = "Select";
    }
    protected void lnkbtncomposemail_Click(object sender, EventArgs e)
    {
        drpcomposeto.Items.Clear();
        drpcomposeto.Items.Add("Select");
        drpcomposeto.SelectedValue = "Select";
        pnlcompose .Visible = true;
        pnlsent.Visible = false;
        pnlinbox.Visible = false;
        pnlview.Visible = false;
        lblcomposefrom.Text = "admin@pms.com";
        //To display all th username !admin
        string s = "select UserName from Login where UserName !='admin' ";
        rdr = con.select (s);
        while (rdr .Read())
        {
            drpcomposeto.Items.Add(rdr["UserName"].ToString()+"@pms.com");
        }
    }
    protected void grdsentbox_RowCommand(object sender, GridViewCommandEventArgs e)
    {
        pnlview.Visible = true;
        pnlcompose.Visible = false;
        pnlsent.Visible = false;
        pnlinbox.Visible = false;
        string inbox = "select * from MailDetails where Id=" + int.Parse(e.CommandArgument.ToString());
        rdr = con.select(inbox);
        if (rdr.Read())
        {
            lblviewsubject.Text = rdr["Subject"].ToString();
            lblviewdate.Text = rdr["MDate"].ToString();
            lblviewfrom.Text = rdr["FromId"].ToString();
            lblviewto.Text = rdr["ToId"].ToString();
            lblviewcontent.Text = rdr["Contents"].ToString();
            btnreply.Visible = false ;
            string update = "update MailDetails set MailStatus=0 where Id=" +
int.Parse(e.CommandArgument.ToString());
            con.iducommand(update);
        }
    }
    protected void grdinbox_RowDataBound(object sender, GridViewRowEventArgs e)
    {
```

```
    if (e.Row.RowType == DataControlRowType.DataRow)
    {
        string status = ((Label)e.Row.FindControl("lblstatus")).Text;
        if (status == "True")
            e.Row.BackColor = System.Drawing.Color.ForestGreen;
        else
        {
            e.Row.BackColor = System.Drawing.Color.DimGray;
        }

    }
}
protected void chkinboxselect_CheckedChanged(object sender, EventArgs e)
{
    foreach (GridViewRow row in grdinbox.Rows)
    {
        if (chkinboxselect.Checked)
            ((CheckBox)row.FindControl("chkinboxselect")).Checked = true;
        else
            ((CheckBox)row.FindControl("chkinboxselect")).Checked = false;
    }
}
protected void lnkinboxdelete_Click(object sender, EventArgs e)
{
    foreach (GridViewRow row in grdinbox.Rows)
    {

        CheckBox check = (CheckBox)row.FindControl("chkinboxselect");
        if (check.Checked)
        {
            Label lbl = (Label)row.FindControl("lblId");
            string sent = "Delete from MailDetails where Id=" + int.Parse(lbl.Text);
            con.iducommand(sent);
            inbox();
        }
    }
}
protected void lnksentdelete_Click(object sender, EventArgs e)
{
    foreach (GridViewRow row in grdsentbox.Rows)
    {

        CheckBox check = (CheckBox)row.FindControl("chksentmail");
        if (check.Checked)
        {
            Label lbl = (Label)row.FindControl("lblId");
            string sent = "Delete from MailDetails where Id=" + int.Parse(lbl.Text);
            con.iducommand(sent);
            sentmail();
        }

    }
}
protected void chksentselect_CheckedChanged(object sender, EventArgs e)
{
    foreach (GridViewRow row in grdsentbox .Rows)
    {
        if (chksentselect.Checked)
            ((CheckBox)row.FindControl("chksentmail")).Checked = true;
        else
            ((CheckBox)row.FindControl("chksentmail")).Checked = false;
    }
}
```

APPROVEPROJECTS.ASPX

```
Databaseclass con = new Databaseclass();
    SqlDataReader rdr;
    string sql, sql1,sql2,sql3;
    int count;
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            sql = "select count(*) from RejProjects where Status='Started' ";
            count = con.scalar(sql);
            if (count > 0)
            {
                sql1 = "select ProjectName from RejProjects where Status='Started' ";
                rdr = con.select(sql1);
                while (rdr.Read())
                {
                    drpprojname.Items.Add(rdr["ProjectName"].ToString());
                }
            }
            else
            {
                lblmsg.Text = "No Projects Started";
            }
        }
    }
    protected void btnsubmit_Click(object sender, EventArgs e)
    {
        if (drpprojname.SelectedValue != "Select" && drpstatus.SelectedValue != "Select")
        {
            sql = "update RejProjects set Status='" + drpstatus.SelectedValue.ToString() + "' where ProjectName='"
+ drpprojname.SelectedValue.ToString() + "' ";
            con.iducommand(sql);
            if (drpstatus.SelectedItem.ToString() == "Completed")
            {
                sql ="select m.DeveloperUserName FROM ModuleNames AS m INNER JOIN EmployeeReg AS e
ON m.DeveloperUserName = e.EmpUserName WHERE (m.ProjectName = '" +
drpprojname.SelectedItem.ToString() + "')";
                rdr = con.select(sql);
                while (rdr.Read())
                {
                    sql1 = "update EmployeeReg set EmpStatus='Avilabile' where EmpUserName='" + rdr.GetString(0)
+ "'";
                    con.iducommand(sql1);
                }
                sql2 = "SELECT p.PlUserName FROM PlTaskReg AS p INNER JOIN EmployeeReg AS e ON
p.PlUserName = e.EmpUserName WHERE (p.ProjectName ='" + drpprojname.SelectedItem.ToString() + "')";
                rdr = con.select(sql2);
                if (rdr.Read())
                {
                    sql3 = "update EmployeeReg set EmpStatus='Avilabile' where EmpUserName='" + rdr.GetString(0)
+ "'";
                    con.iducommand(sql3);
                }
            }


            drpprojname.SelectedValue = "Select";
            drpstatus.SelectedValue = "Select";
            lblmsg.Text = "Project Status Updated";
        }
        else
        {
            lblmsg.Text = "Invalid Selection";
        }
    }
```

COSTCALCULATION.ASPX.CS

```
Databaseclass con = new Databaseclass();
    SqlDataReader rdr;
    string sql;
    int a, b, c,d;
    string days;
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            sql = "select ProjectName from RejProjects where Status='Pending' ";
            rdr=con.select(sql);
            while (rdr.Read())
            {
                drpprojname.Items.Add(rdr.GetString(0));
            }
            rdr.Close();
        }
    }
    protected void drpprojname_SelectedIndexChanged(object sender, EventArgs e)
    {
        txtclientname.Text = "";
        sql = "select CUserName from RejProjects where ProjectName= '"+ drpprojname.SelectedItem.ToString()
+ "'";
        string cname = con.Scalar1(sql);
        txtclientname.Text = cname ;

    }
    protected void btncalculate_Click(object sender, EventArgs e)
    {
        lblproerror.Text = "";
        if (drpprojname.SelectedValue == "Select")
        {
            lblpronameerror.Visible = true;
            lblpronameerror.ForeColor = System.Drawing.Color.Red;
            lblpronameerror.Text = "Select a Project Name";
        }
        else if (DateTime.Parse(txtexstartdate.Text) < DateTime.Now)
        {
            lbldateerror.Visible = true;
            lbldateerror.ForeColor = System.Drawing.Color.Red;
            lbldateerror.Text = "Start Date Can't Be Set To An Earlier Time";
        }
        else
        {
            TimeSpan diffe;
             DateTime Startdate = DateTime.Parse(txtexstartdate .Text);
            DateTime Enddate = DateTime.Parse(txtexenddate .Text);
            if (DateTime.Compare(Startdate, Enddate) == 0)
            {
                lbldateerror.Visible = true;
                lbldateerror.Text = "Both Dates Are Same";
            }
            else if (DateTime.Compare(Enddate, Startdate) < 0)
            {
                lbldateerror.Visible = true;
                lbldateerror.Text = "End Date Less Than Start Date";
            }
            else if (DateTime.Compare(Enddate, Startdate) > 0)
            {
                //Subtract btwn startdate and enddate
                diffe = Enddate.Subtract(Startdate);
                days = diffe.TotalDays.ToString();
                a = Convert.ToInt32(days.ToString());
                b = Convert.ToInt32(txtonedaywage.Text);
                c = Convert.ToInt32(txtteamsize.Text);
                d = a * b * c;
                txttotalcost.Text = d.ToString();
            }
        }
    }
    protected void btnsubmit_Click(object sender, EventArgs e)
    {
        if (String.IsNullOrEmpty(txtclientname.Text) || String.IsNullOrEmpty(txtexstartdate.Text) ||
String.IsNullOrEmpty(txtexenddate.Text) || String.IsNullOrEmpty(txtonedaywage.Text) ||
String.IsNullOrEmpty(txtteamsize.Text) || String.IsNullOrEmpty(txttotalcost.Text))
```

```
        {
            lblproerror.Visible = true;
            lblproerror.ForeColor = System.Drawing.Color.Red;
            lblproerror.Text = "* Required Fileds";
        }
        else
        {
            sql = "insert into ProjectCost(ProjectName,ClientName,StartDate,EndDate,TeamSize,TotalCost) values
('" + drpprojname.SelectedItem.ToString() + "','" + txtclientname.Text + "','" + txtexstartdate.Text + "','" +
txtexenddate.Text + "','" + txtteamsize.Text + "','" + txttotalcost.Text + "')";
            con.iducommand(sql);
            Response.Write("Cost Sucessfully added");
        }
    }
```

PLTEAMSELECTION.ASPX.CS

```
Databaseclass con = new Databaseclass();
    string sql,sql1;
    int count;
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            sql1 = "select count(*) from EmployeeReg where EmpDesignation='DVPL' and EmpStatus='Avilabile' ";
            count = con.scalar(sql1);
            if (count > 0)
            {
                sql = "select * from EmployeeReg where EmpDesignation='DVPL' and EmpStatus='Avilabile' ";
                grddevselection.DataSource = con.displaydata(sql);
                grddevselection.DataBind();
            }
            else
            {
                lblerrormsg.Visible = true;
                lblerrormsg.ForeColor = System.Drawing.Color.Red;
                lblerrormsg.Text = "No Developers are now Available";

            }
        }
    }
    protected void grddevselection_SelectedIndexChanging(object sender, GridViewSelectEventArgs e)
    {
        Session["empusername"] = grddevselection.Rows[e.NewSelectedIndex].Cells[1].Text;
        Response.Redirect("SelectRegTask.aspx");
    }
    protected void btnviewselectedteam_Click(object sender, EventArgs e)
    {
        Response.Redirect("PLViewSelectedTeam.aspx");
    }
```

PLVIEWSELECTEDTEAM.ASPX.CS

```
Databaseclass con = new Databaseclass();
    string sql, sql1;
    protected void Page_Load(object sender, EventArgs e)
    {
        displayteam();
    }
    private void displayteam()
    {
        sql = "SELECT e.EmpUserName,e.EmpName,e.EmpPic,e.EmpSkills FROM PlTaskReg AS p INNER
JOIN ModuleNames AS m ON p.ProjectName = m.ProjectName INNER JOIN RejProjects AS r ON
r.ProjectName = p.ProjectName AND r.Status <> 'Completed'INNER JOIN EmployeeReg AS e ON
m.DeveloperUserName = e.EmpUserName WHERE (e.EmpDesignation = 'DVPL')AND (p.PlUserName =
'"+Session["username"]+"')";
        grdselectedteam.DataSource = con.displaydata(sql);
        grdselectedteam.DataBind();
    }
    protected void grdselectedteam_SelectedIndexChanged(object sender, EventArgs e)
    {
        Session["empusername"] = grdselectedteam.SelectedDataKey.Value.ToString();
        sql="update ModuleNames set DeveloperUserName=NULL where DeveloperUserName='"+Session["empusername"].ToString()+"'";
        con.iducommand(sql);
        sql1 = "update EmployeeReg set EmpStatus='Avilabile' where EmpUserName='" + Session["empusername"].ToString() + "' ";
        con.iducommand(sql1);
        Response.Redirect("PLViewSelectedTeam.aspx");
        displayteam();
    }
```

CHANGEPASSWORD.ASPX.CS

```
Databaseclass con = new Databaseclass();
    SqlDataReader rdr;
    protected void Page_Load(object sender, EventArgs e)
    {
        txtusername.Text = Session["username"].ToString();
    }
    protected void btnsubmit_Click(object sender, EventArgs e)
    {
        try
        {
            if (String.IsNullOrEmpty(txtoldpwd.Text) || String.IsNullOrEmpty(txtnewpwd.Text) || String.IsNullOrEmpty(txtconfirmpwd.Text))
            {
                lblerror.Visible = true;
                lblerror.ForeColor = System.Drawing.Color.Red;
                lblerror.Text = "* Required Fileds";
            }
            else
            {
                string sql = "select Password from Login where UserName='" + txtusername.Text + "'";
                string passwd = con.Scalar1(sql);
                if (txtoldpwd.Text != passwd)
                {
                    lblerror.Visible = true;
                    lblerror.ForeColor = System.Drawing.Color.Red;
                    lblerror.Text = "Wrong Password";
                    txtoldpwd.Focus();
                }
                else
                {
                    string sql1 = "update Login set Password='" + txtnewpwd.Text + "' where UserName ='" +
txtusername.Text + "' ";
                    con.iducommand(sql1);
                    lblerror.Visible = true;
                    lblerror.ForeColor = System.Drawing.Color.Green;
                    lblerror.Text = "Password Change sucessfully";
                }
            }

        }
        catch (Exception Ex)
        {
            lblerror.ForeColor = System.Drawing.Color.Red;
            lblerror.Text = "Wrong Password";
            txtoldpwd.Focus();
        }
        finally
        {
        }
    }
    protected void btnclear_Click(object sender, EventArgs e)
    {
        txtoldpwd.Text = "";
        txtnewpwd.Text = "";
        txtconfirmpwd.Text = "";
        lblerror.Visible = false;
    }
DATABASECLASS.CS
using System.Data.SqlClient ;

public class Databaseclass
{
    public SqlConnection Databaseconnection()
    {
        SqlConnection ob = new SqlConnection("Data Source=.\\SQLEXPRESS;AttachDbFilename=|DataDirectory|\\PMSS.mdf;Integrated Security=True;user Instance=True");

        ob.Open ();
        return ob;
    }
    public void iducommand(string str)
    {
      SqlCommand cmd=new SqlCommand (str,Databaseconnection ());
      cmd.ExecuteNonQuery ();
    }
    public SqlDataReader select(string str)
    {
        SqlCommand  cmd=new SqlCommand  (str,Databaseconnection ());
        return cmd.ExecuteReader();
    }
   public DataSet displaydata(string str)
   {
       SqlDataAdapter da=new SqlDataAdapter (str,Databaseconnection ());
       DataSet ds=new DataSet ();
       da.Fill(ds);
       return ds;
   }
    public int scalar(string str)
    {
        SqlCommand cmd = new SqlCommand(str,Databaseconnection ());
        return (Convert.ToInt32(cmd.ExecuteScalar()));

    }
    public string Scalar1(string str)
    {

        SqlCommand cmd = new SqlCommand(str, Databaseconnection  ());
        return (Convert.ToString(cmd.ExecuteScalar()));
    }
}
```