Chapter 1: System Introduction

Chapter -1

Systems Introduction

Description of Organization

Introduction – **Human Resource Management**

1.1 History of Organization

HCL INFOSYSTEMS LTD. is one of the pioneers in the Indian IT market, with its origins in 1976. For over quarter of a century, we have developed and implemented solutions for multiple market segments, across a range of technologies in India. We have been in the forefront in introducing new technologies and solutions.

HCL INFOSYSTEMS LTD. is India's premier hardware, services and ICT systems Integration Company offering a wide spectrum of ICT products that includes Computing, Storage, Networking, Security, Telecom, Imaging and Retail. HCL is a one-stop-shop for all the ICT requirements of an organization. India's leading System Integration and Infrastructure Management Services Organization, HCL has specialized expertise across verticals including Telecom, BFSI, e-Governance & Power. HCL has India's largest distribution and retail network, taking to market a range of Digital Lifestyle products in partnership with leading global ICT brands, including Apple, Cisco, Ericsson, Kingston, Kodak, Konica Minolta, Microsoft, Nokia, Toshiba, and many more.

1.2 Objective of Organization

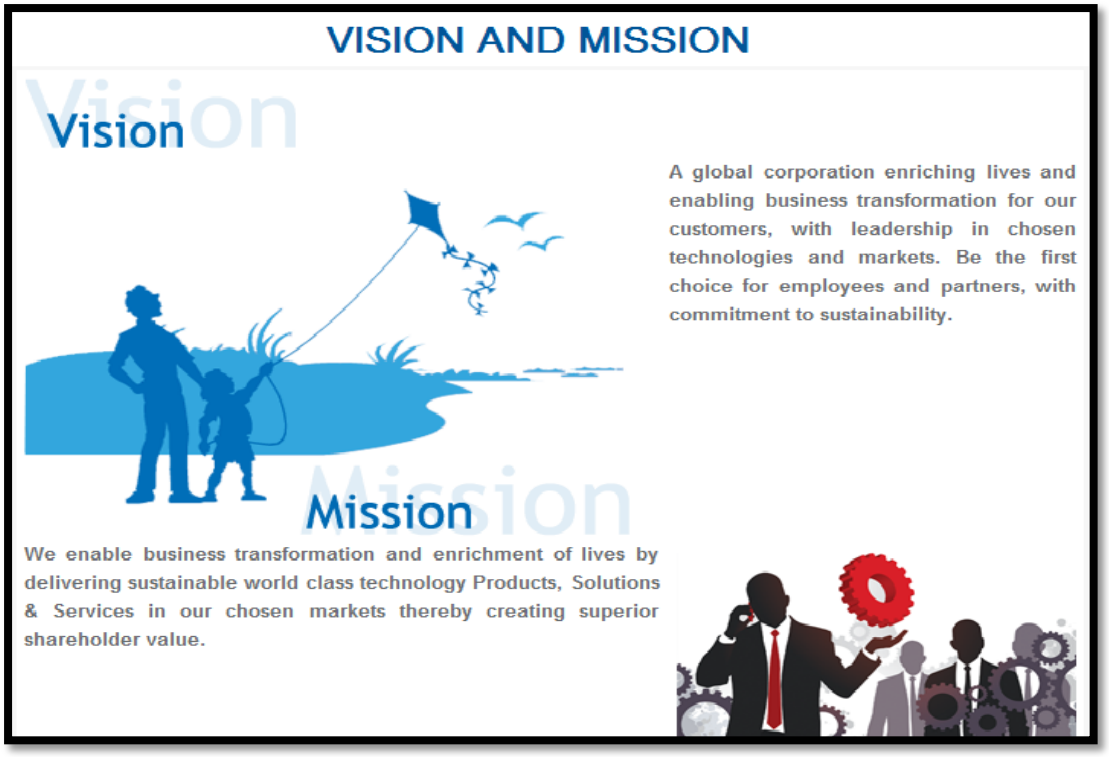

Figure 1.1

1.3 Organizational Structure

In a world where the right technology infrastructure is a prerequisite, we offer a reliable IT backbone to our partners. HCL combines technical innovation with built-in reliability to keep your business running. We provide a one stop shop for meeting end-to-end IT requirements, thus offering a smooth ICT management. Additionally, we offer industry leading technology, designed to deliver a price to performance advantage to help you provide increased benefits to your customers. Our high-quality products and services give you means to work in a smarter way and be more productive and competitive.

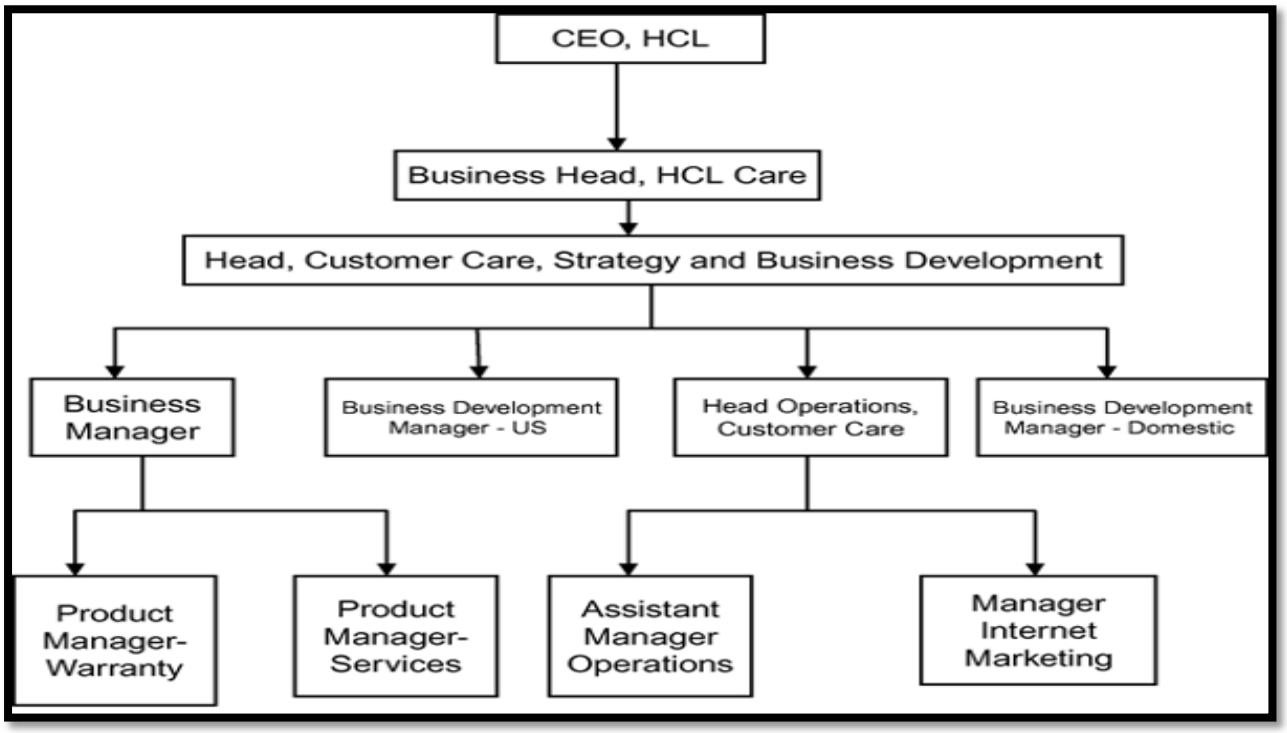

Figure 1.2

Extensive Marketing Support

HCL has closely seen the IT industry rise from scratch, and has actively participated in its progress. We have picked up valuable marketing lessons in serving the IT needs of the Indian customers. You can combine your individual strengths and reputation with the power of a global brand.

We can help you to focus on some of the most critical marketing needs facing your business. Additionally, we can provide you a set of proven sales and marketing tools designed to help you generate new leads, increased demands for products and services and help you reach your business goals.

1.4 Key Result Areas

Facilitation and Coordination of Skill Development Track under CSR:

1. Complete co- ordination and implementation of project- Udaan a National Skills Development Corporation and Ministry of Home Affairs awarded skill development project in the state of Jammu and Kashmir. The key responsibilities includes complete end to end liaison with government, proposal development, grant release, marketing and mobilization for project implementation.

2. Co – ordination with Telecom Sector skill council in development of national occupation standards for telecom sector.

Core Corporate Social Responsibility Function:

1. Instrumental in forming CSR committee and framing CSR policies for HCL Info systems.
2. Engaged with NGO's working with disabled population to ensure equal access to technology.
3. Community engagement in Delhi NCR and in all other HCL Info systems locations in partnership with Key Non Profit Organizations.
4. Documentation and presentation of Quarterly newsletters and annual report on CSR and sustainability practices of the organization.

Training and development in HCL Info systems:

Facilitated training to L1, L2 and L3 staff on HCL one training focusing on professional, behavioral and soft skills.

Chapter 2: System Analysis

Chapter – 2

System Analysis

Software Requirement Specifications

2.1 Introduction

The following subsections of Software Requirement Specifications Document should facilitate in providing the entire overview of the Information system "Human Resource Management" under development. This document aims at defining the overall software requirements for your end users. Efforts have been made to define the requirements of the Information system exhaustively and accurately.

2.1.1 Purpose

The main purpose of Software Requirement Specifications Document is to describe in a precise manner all the capabilities that will be provided by the Software Application "Human Resource Management". It also states the various constraints which the system will be abide to. This document further leads to clear vision of the software requirements, specifications and capabilities. These are to be exposed to the development, testing team and end users of the software.

2.1.2 Scope

The future looks quite good for organizations but it has been true, it will be good for enhanced productivity fully utilizing the combined talent and skills of the entire workforce of an organization. HRM must today reinvent itself to cope with the demands of a global economy.

2.1.3 Definition, acronyms, abbreviations

DFD: - Data Flow Diagram.
ER Diagram: - Entity Relationship Diagram.
Admin: - Administrator.
Emp: - Employee.
Employ: -Employee.
HR: - Human Resources.

2.1.4 References

i. http://en.wikipedia.org/wiki/Enterprise_resource_planning
ii. http://en.wikipedia.org/wiki/Human_resource_management
iii. http://humanresources.about.com/od/glossaryh/f/hr_management.htm
iv. http://bookboon.com/en/hrm-ebooks

2.1.5 Overview

This document has been prepared in accordance with the IEEE standard 30/1998, IEEE recommended practice for software requirement specification. It provides the information of product perspective, product function, user characteristics, constraints, assumptions and dependencies and specific requirements.
The rest of this SRS document describes the various system requirements, interfaces, features and functionality in detail.

2.2 Overall description of proposed system

2.2.1 Product Perspective

The application will be windows-based, self contained and independent software product.

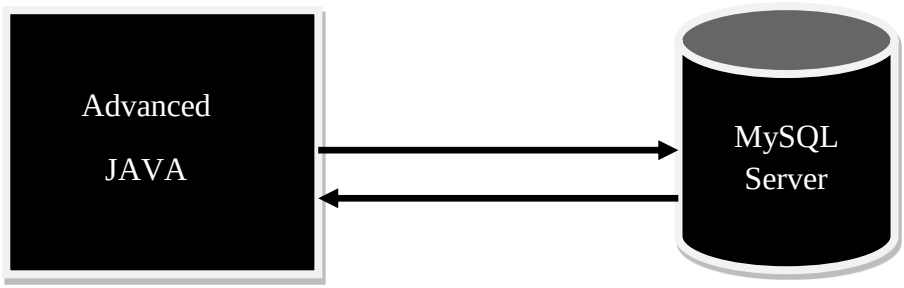

Figure 2.1

2.2.1.1 System Interfaces

The system will be complete window based operating system with configuration as:
OS: XP/VISTA/7/8
Processor: Intel Core 2 duo, Dual Core etc.

2.2.1.2 Interfaces

The application will have a user friendly and menu based interface. Following screens will be provided.

i. A **Login Screen** for entering username, password and role (Administrator, User) will be provided. Access to different screens will be based upon the role of the user. The Login Screen for the user and administrator should be different .So the user can't interfere in working of administrator.
ii. The **Modification Screen** will be provided to the DBA. If the admin wants to make any modification in details of the spiritual places whether it is updating or deletion or modification, the DBA can do this with the help of Modification screen.

iii. The **Report Screen** will be provided to the user. If the user wants to search a particular spiritual place, the user can view the report. This screen is mainly for the DBA or other users.

2.2.1.3 Hardware Interfaces

Processor : X86 Compatible processor with 1.7 GHz Clock speed

RAM : 512 MB or more

Hard disk : 20 GB or more

Monitor : VGA/SVGA

Keyboard : 104 Keys

Mouse : 2 buttons/ 3 buttons

2.2.1.4 Software Interfaces

Operating System : Windows 2000/7/8/XP/VISTA
Front end : JSP
Back end : MySQL

2.2.1.5 Communication Interfaces

The e-store system shall use the HTTP protocol for communication over the internet and for the intranet communication will be through TCP/IP protocol suite.

2.2.1.6 Memory Constraints

RAM : 512 MB or more

Hard disk : 20 GB or more

2.2.1.7 Operations

This product will not cover any automated housekeeping aspects of database. The DBA at client site will be manually deleting old/ non required data. Database backup and recovery will also have to be handled by DBA.

2.2.1.8 Site Adaptation Requirement

The terminals at client side will have to support the hardware and software interfaces specified.

2.2.2 Product functions

The system will allow access only to authorized users with specific roles (Administrator, Operator). Depending upon the user's role, he/she will be able to access only specific modules of the system.

A summary of the major functions that the software will perform:

i. **A Login facility for enabling only authorized access to the system**: Access control is the selective restriction of access to a place or other resource. The act of accessing may mean consuming, entering, or using. Permission to access a resource is called authorization. Locks and login credentials are two analogous mechanisms of access control

ii. **Users (with role operator) will add/update/delete the stored information and so on:** In computer programming, create; read, update and delete (CRUD) (Sometimes called SCRUD with an "S" for Search) are the four basic functions of persistent storage. Sometimes CRUD is expanded with the words retrieve instead of read, modify instead of update, or destroy instead of delete. It is also sometimes used to describe user interface conventions that facilitate viewing, searching, and changing information; often using computer-based forms and reports. The term was likely first popularized by James Martin in his 1983 book managing the Data-base Environment. The acronym may be extended to CRUDL to cover listing of large data sets which bring additional complexity such as pagination when the data sets are too large to hold easily in memory. CRUD is also relevant at the user interface level of most applications. For example, in address book software, the basic storage unit is an individual contact entry. As a bare minimum, the software must allow the user to:

    1. Create or add new entries
    2. Read, retrieve, search, or view existing entries
    3. Update or edit existing entries
    4. Delete/deactivate existing entries

Without at least these four operations, the software cannot be considered complete. Because these operations are so fundamental, they are often documented and described under one comprehensive heading, such as "contact management", "content management" or "contact maintenance" (or "document management" in general, depending on the basic storage unit for the particular application).

2.2.3 User Characteristics

1. Educational Level: At least graduate and should be comfortable with English language.

2. Technical Expertise: Should be a high or middle level employee of the organization comfortable with using general purpose applications on a computer.

2.2.4 Constraints

None

2.2.5 Assumptions and Dependencies

Assumptions: The code should be free with all possible errors and the product must have an interface which is simple enough to understand.

Dependencies: All necessary hardware and software are available for implementing and use of the tool. The proposed system would be designed, developed and implemented based on the software requirements specifications document. Users should have basic knowledge of computer and we also assure that the users will be given software training documentation and reference material.

2.2.6 Apportioning Requirement

Not Required

2.3 Specific Requirements

This section contains the software requirements to a level of detail sufficient to enable designers to design the system, and testers to test the system.

2.3.1 External Interfaces

2.3.1.1 User Interfaces

Refer 2.2.1.2

2.3.1.2 Hardware Interfaces

Refer 2.2.1.3

2.1.1.1 Software Interfaces

Refer 2.2.1.4

2.1.1.2 Communication Interfaces

None

2.1.2 System Features

Module Names:

i. Administrator

a. Login
b. Create Employee
c. Update Employee Records
d. Delete Employee
e. Project Allotment
f. Update Project Allotment
g. Delete Project
h. View Employee Record
i. Pay Slip Generation
j. Reset password
k. Logout

ii. Employee

a. Login
b. Set Attendance
c. View Payslip
d. View Project
e. Reset Password
f. Logout

Description:

i. <u>Administrator</u>

a. Login

In the admin table, we have two attributes User Id and Password wherein the admin can login. He can view and edit student as well as trainer profile. He can update, add modify etc. in student as well as trainer profile.

Validity Checks

1. Email id
2. Password

Sequencing Information/Exception handling

First admin will enter the email id and password. If in case of admin login, he gives wrong email ID then he get an error page.

b. Create Employee:

Admin can create record of new employees recruited in the firm, i.e. he can add the employee in his organization.

c. Update Employee Record:

Admin can update the employee record by adding employee, deleting any employee record etc.

d. Delete Employee:

In this module admin can delete the record of that employee who leaves the firm.

e. Project Allotment:

In this module admin allot the project to their employees.

f. Update Project Allotment:

In this module admin update the projects allotted to his/her employee, in updating admin can add more new modules in the project, or he also can delete the modules that are not required in project.

g. Delete Project:

In this module admin can delete the record of that project that leaves the firm.

h. View Employee Record

This is a module which shows the updated list of the entire employee with in a firm.

i. Pay Slip Generation:

The payslip generation control goes to only admin. Only admin can generate their employees on the basis of no. of days employee have attended.

j. Reset Password:

Admin can reset the password of its account.

k. logout

In this module the first screen will open and other screen will closed

ii. <u>Employee</u>

Employee plays an important role in any organization, here employee can set his/her attendance in which he can check how many days in a month He/she was present, check payslip, and reset the password of their account.

Validity Checks

1. Set Attendance.
2. View Payslip.

Sequencing Information/Exception handling

If employ try to enter previous day attendance its gives an error page.

a. Set Attendance:

Employee can set the attendance example if he/she is present in the organization he can set the present.

b. Reset Password:

Employee can reset the password of his/her account.

c. Check Payslip:

Employee can check the payslip of his/her salary.

d. Logout

In this module the first screen will open and other screen will closed.

2.1.3 Performance Requirements

None

2.1.4 Logical Database Requirements

The proposed information system contains the following data tables in its database collection.

1) Login
2) Employ
3) Project
4) Attendance
5) Payslip

2.1.5 Design Constraints

i. The system runs under Windows XP and Windows latest versions.
ii. The system requires regular yearly updating.

2.1.6 Software System Attributes

Reliability

This application is a reliable product that produces fast and verified output of all its processes.

Availability

This application will be available to use for your end users and help them to carry out their operations conveniently.

Security

The application will be password protected. User will have to enter correct username, password and role in order to access the application.

Maintainability

The application will be designed in a maintainable manner. It will be easy to incorporate new requirements in the individual modules.

Portability

The application will be easily portable on any windows-based system that has oracle installed.

2.1.7 Other Requirements

None

3. Methodologies for Data Collection

3.1 Primary Data Collection

Some projects involve going into workplaces and asking workers questions. Researchers who do this have specific work-health questions in mind that they'd like answered. The answers or data collected from the responses are called primary data. An advantage of using primary data is that researchers are collecting information for the specific purposes of their study. In essence, the questions the researchers ask are tailored to elicit the data that will help them with their study. Researchers collect the data themselves, using surveys, interviews and direct observations (such as observing safety practices on a shop floor)

3.2 Secondary Data Collection

Other projects involve using data that has already been gathered by someone else, such as survey information from the Canadian Census. Researchers then examine this information in a different way to find a response to their question. This data are called secondary data. Secondary data sources can be very useful resources for research. These data sources are collected by organizations to conduct their business. These types of data generally provide savings in cost and time and reduction in respondent burden as compared to direct surveying or interviewing.

Both primary data and secondary data have their pros and cons. Primary data offers tailored information but tends to be expensive to conduct and takes a long time to process. Secondary data is usually inexpensive to obtain and can be analysed in less time. However, because it was gathered for other purposes, you may need to tease out the information to find what you are looking for. For secondary level data collection I gathered other necessary details from:

i. www.sap.com
ii. http://www.hrmguide.net/
iii. http://www.orangehrm.com/

4. Methodology used for Analysis, Design and Development

The basic idea here is that instead of freezing the requirements before a design or coding can proceed, a throwaway prototype is built to understand the requirements. This prototype is developed based on the currently known requirements. By using this prototype, the client can get an "actual feel" of the system, since the interactions with prototype can enable the client to better understand the requirements of the desired system. Prototyping is an attractive idea for complicated and large systems for which there is no manual process or existing system to help determining the requirements. The prototype is usually not complete systems and many of the details are not built in the prototype. The goal is to provide a system with overall functionality. Prototype Model places more effort in creating the actual software instead of concentrating on documentation. This way, the actual software could be released in advance. Prototyping requires more user involvement and allows them to see and interact with a prototype allowing them to provide better and more complete feedback and specifications. The presence of the prototype being examined by the user prevents many misunderstandings that occur when each side believe the other understands what they said. The final product is more likely to satisfy the user's desire for look, feel and performance.

4.1 Stages of Model

i. Requirement analysis and specification phase:

The goal of this phase is to understand the exact requirement of the customer and to document them properly. The requirement describes the "what" of a system not the "how". This phase produce a large document written in a natural language, contains a description of what a system will do without describing how it will be done.

ii. Design:

The goal of this phase is to transform the requirements specification into a structure that is suitable for implementation in some programming language. Here, overall software architecture is defined, and the high level and detailed designed work is performed. This work is documented and known as software design description (SDD) document**.**

iii. Implementation and unit testing phase:

During this phase design is implemented. If the SDD is complete, the implementation and coding phase proceeds smoothly, because all the information needed by the software developers is contained in SDD.

iv. Integration and system testing phase:

This is very important phase the purpose of unit testing is to determine that each independent module correctly implemented, to determine that the interface between modules is also correct, and for this reason integration testing is performed. System testing involves the testing of the entire system, whereas software is a part of system.

v. Operation and maintenance phase:

Software maintenance is a task that every development group has to face. When the software is delivered to the customer site, install and is operational. Therefore the release of the software inaugurates the operations and maintenance phase of the lifecycle.

4.2 Block Diagram of Model

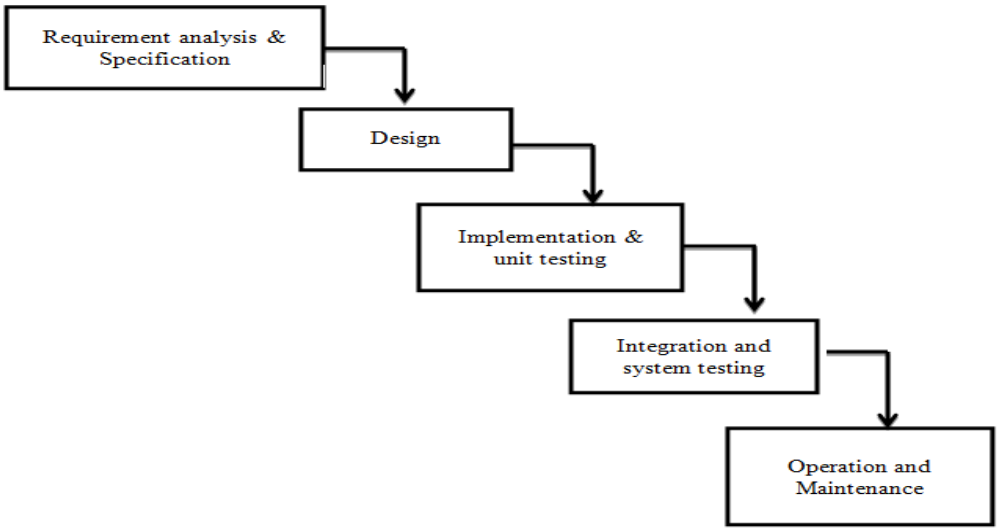

Figure 2.2

4.3 Reasons for choosing Model

1. Users are actively involved in the development
2. Since in this methodology a working model of the system is provided, the users get a better understanding of the system being developed.
3. Errors can be detected much earlier.
4. Quicker user feedback is available leading to better solutions.
5. Missing functionality can be identified easily
6. Confusing or difficult functions can be identified

Requirements validation, Quick implementation of, incomplete, but functional, application.

5. Project Planning Gantt chart

A Gantt chart is a type of bar chart that illustrates a project schedule. A Gantt chart illustrates the start and finish dates of the terminal elements and summary elements of a project. Some Gantt Charts also show the dependency (i.e. precedence network) relationships between activities. Gantt Charts can be used to show current schedule status using percent-complete shadings and a vertical "TODAY" line as shown here. Gantt chart for Human Resource Management is shown here.

	Task Name	Duration	Start	Finish	Resource Names
1	**Human Resoursce Management**	**36 days?**	**Wed 7/2/14**	**Sat 8/9/14**	
2	Finalisation of Topic	1 day?	Wed 7/2/14	Wed 7/2/14	
3	Submission of Synopsis	1 day?	Fri 7/4/14	Fri 7/4/14	
4	**Chapter 1 : Introduction**	**7 days?**	**Sat 7/5/14**	**Mon 7/14/14**	
5	1.1 Description of Organisation	3 days?	Sat 7/5/14	Tue 7/8/14	Rakesh Sahani
6	1.2 Data Collection	4 days?	Tue 7/8/14	Fri 7/11/14	Ankush Sharma
7	1.3 Project Planning	2 days?	Fri 7/11/14	Mon 7/14/14	Himanshi
8	**Chapter 2 : System Analysis**	**10 days?**	**Tue 7/15/14**	**Thu 7/24/14**	
9	**1 Description Of Information Sy**	**8 days?**	**Tue 7/15/14**	**Tue 7/22/14**	
10	2.1.1 Identify the Process	2 days?	Tue 7/15/14	Wed 7/16/14	Rakesh Sahani
11	2.1.2 Identify the input to and ou	2 days?	Thu 7/17/14	Fri 7/18/14	Ankush Sharma
12	2.1.3Identify the data elements	2 days?	Sat 7/19/14	Sun 7/20/14	Himanshi
13	2.1.4 Identify the procedures	2 days?	Mon 7/21/14	Tue 7/22/14	Sachin Gogna
14	2 Conceptual Design	2 days?	Wed 7/23/14	Thu 7/24/14	Rakesh Sahani
15	**Chapter 3 : System Requirement Sf**	**11 days?**	**Fri 7/25/14**	**Mon 8/4/14**	
16	**3.1 Physical Specification**	**2 days?**	**Fri 7/25/14**	**Sat 7/26/14**	**Rakesh Sahani**
17	3.1.1 ERD	2 days?	Fri 7/25/14	Sat 7/26/14	
18	3.2Output Specification	3 days?	Sun 7/27/14	Tue 7/29/14	Ankush Sharma
19	3.3Input Specification	1 day?	Thu 7/31/14	Thu 7/31/14	Rakesh Sahani
20	3.4 Validate Specification	2 days?	Fri 8/1/14	Sat 8/2/14	Himanshi
21	3.5 Database Specification	2 days?	Sun 8/3/14	Mon 8/4/14	Sachin Gogna
22	Final Report Submission	5 days?	Tue 8/5/14	Sat 8/9/14	

Figure 2.3

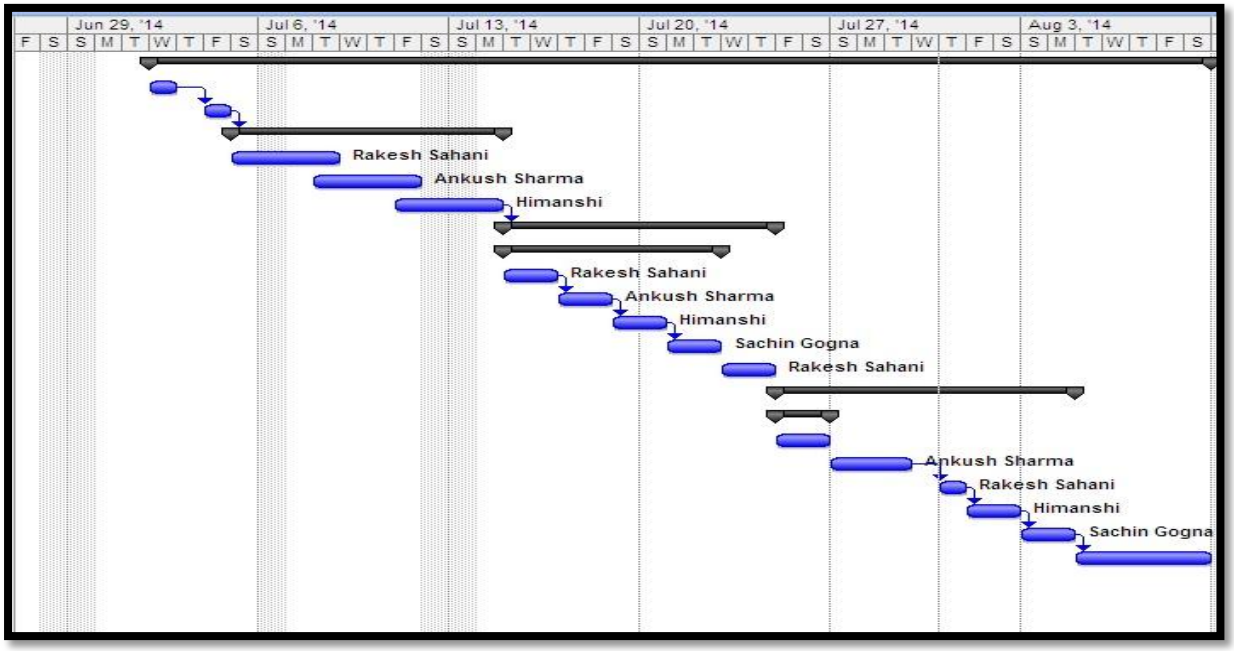

Figure 2.4

Chapter 3: System Design

Chapter – 3

System Design

1. Physical Design

Physical design of Spiritual search engine will provide all the mandatory functions which should be beneficial for the organization. In this, we completed the technical blueprints for the system, based on the implementation platform. Physical design is the generation of tables, indexes, defaults and the checking of restraints.

Block Diagram

Block diagram is a diagram of a system, in which the principal parts or functions are represented by blocks connected by lines that show the relationships of the blocks. They are heavily used in the engineering world in hardware design, electronic design, software design, and process flow diagrams.
The block diagram is typically used for a higher level, less detailed description aimed more understanding the overall concepts and less at understanding the details of implementation.

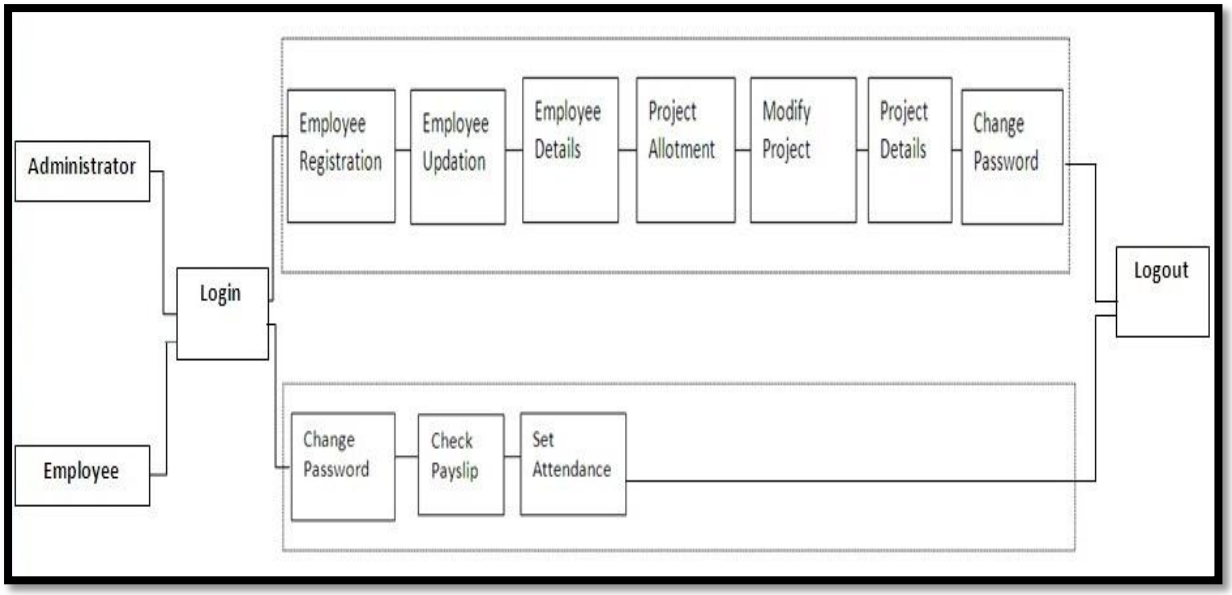

Figure 3.1

2. Use Case

USE-CASE diagrams are the pictorial representation of entities and their functions. There are two components of a use-case diagram – actors and use cases. It represents what happens when an actor interacts with a system. Hence, a use case diagram captures the functional aspects of a system. The system is shown as a rectangle with the name of the system (or subsystems) inside, the actors are shown as stick figures (even the non-human ones), the use cases are shown as solid bordered ovals labeled with the name of the use case, and relationships are lines or arrows between actors and use cases and/or between the use cases themselves. These components are given below:

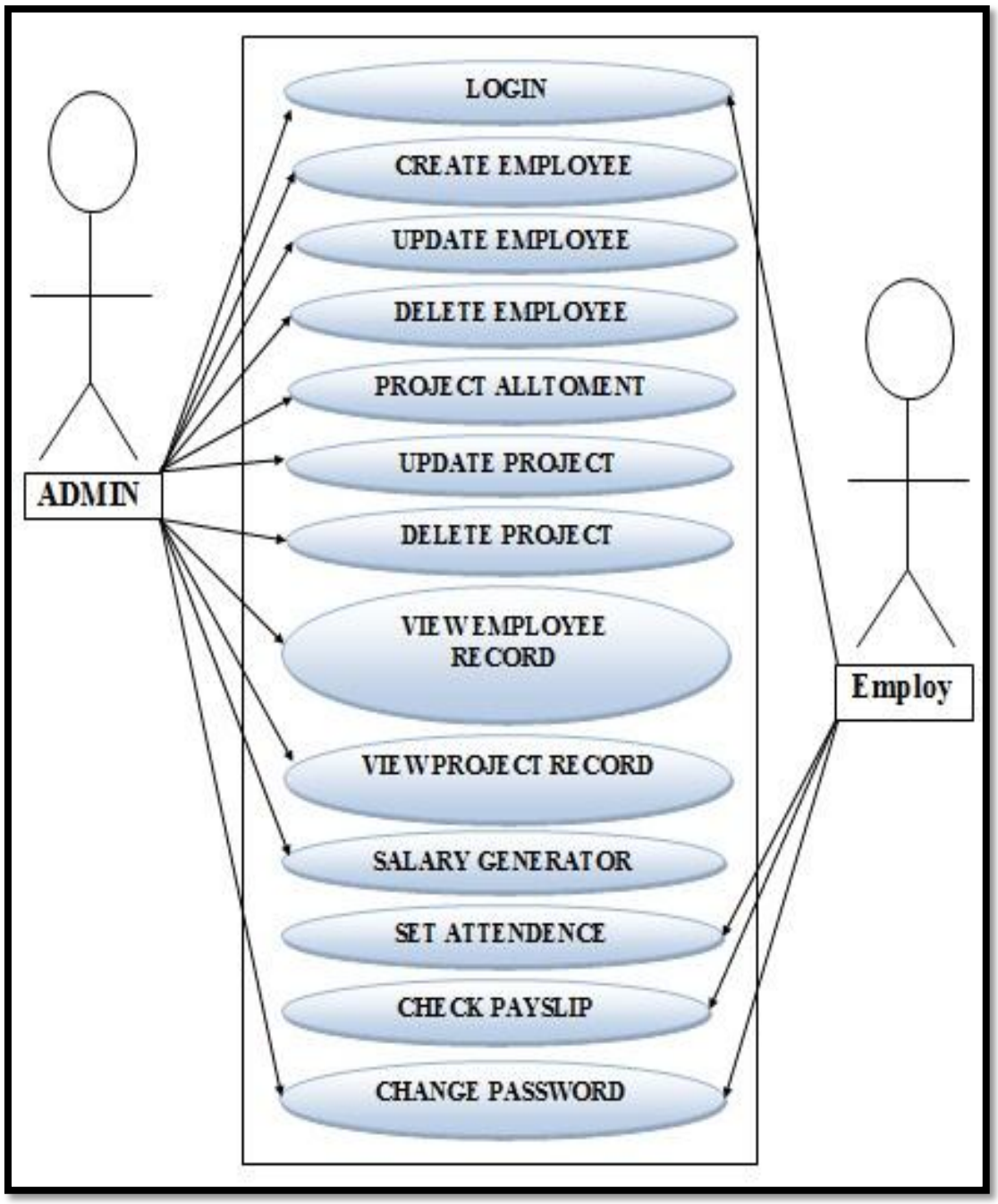

Figure 3.2

3. DFD

A DFD is a graphical representation that depicts information flow and the transforms that are applied as data moves from input to output. The basic form of a DFD is also known as a Data Flow graph or a Bubble Chart. DFD may be used to represent a system or software at any level of the transaction process. DFD's can be partitioned into levels that represent increasing information flow and functional details.

1). Level 0

The next stage is to create the Level 1 Data Flow Diagram. This highlights the main functions carried out by the system. As a rule, we try to describe the system using between two and seven functions - two being a simple system and seven being a complicated system. This enables us to keep the model manageable on screen or paper.

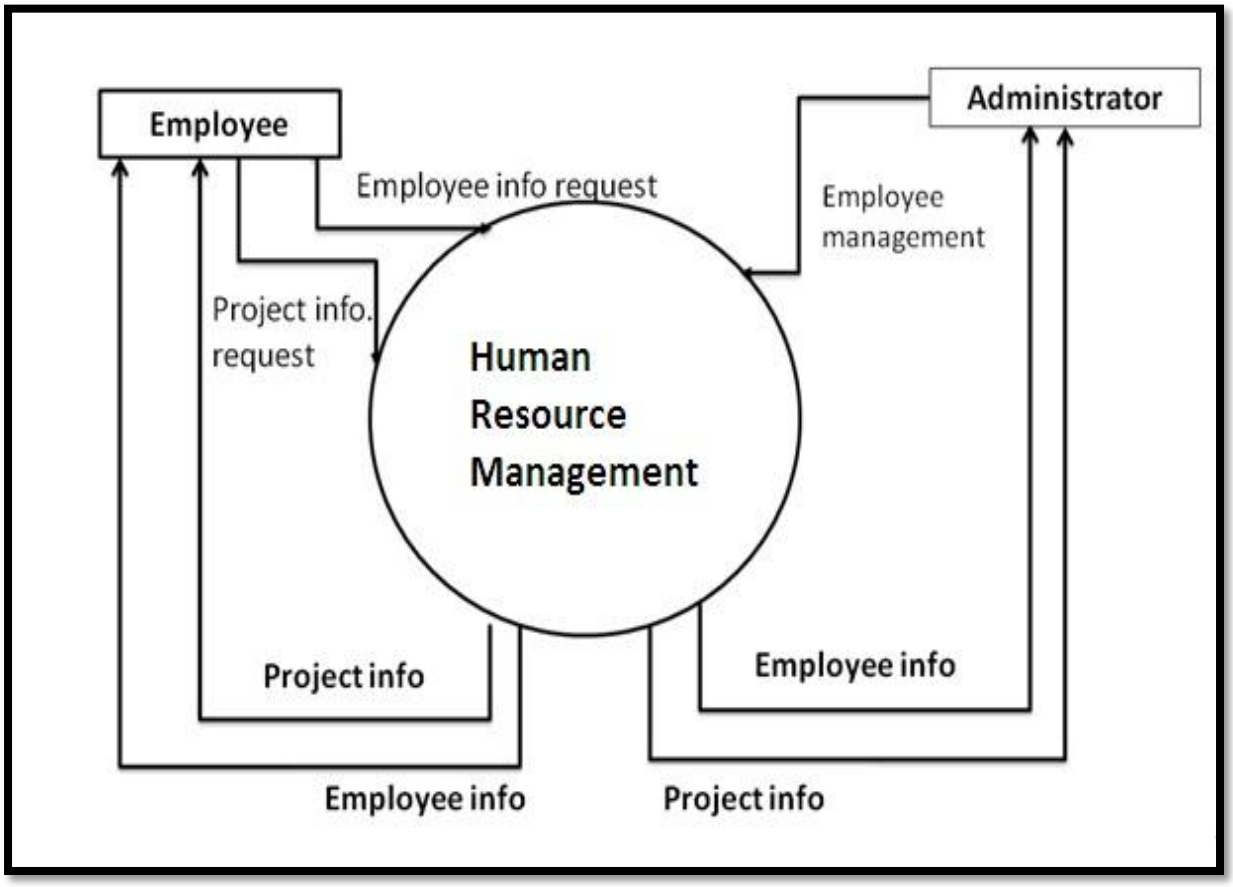

Figure 3.3

2). Level 1

Level Two DFD shows a detailed working of each module that is shown in the Level One DFD. These are as follows:

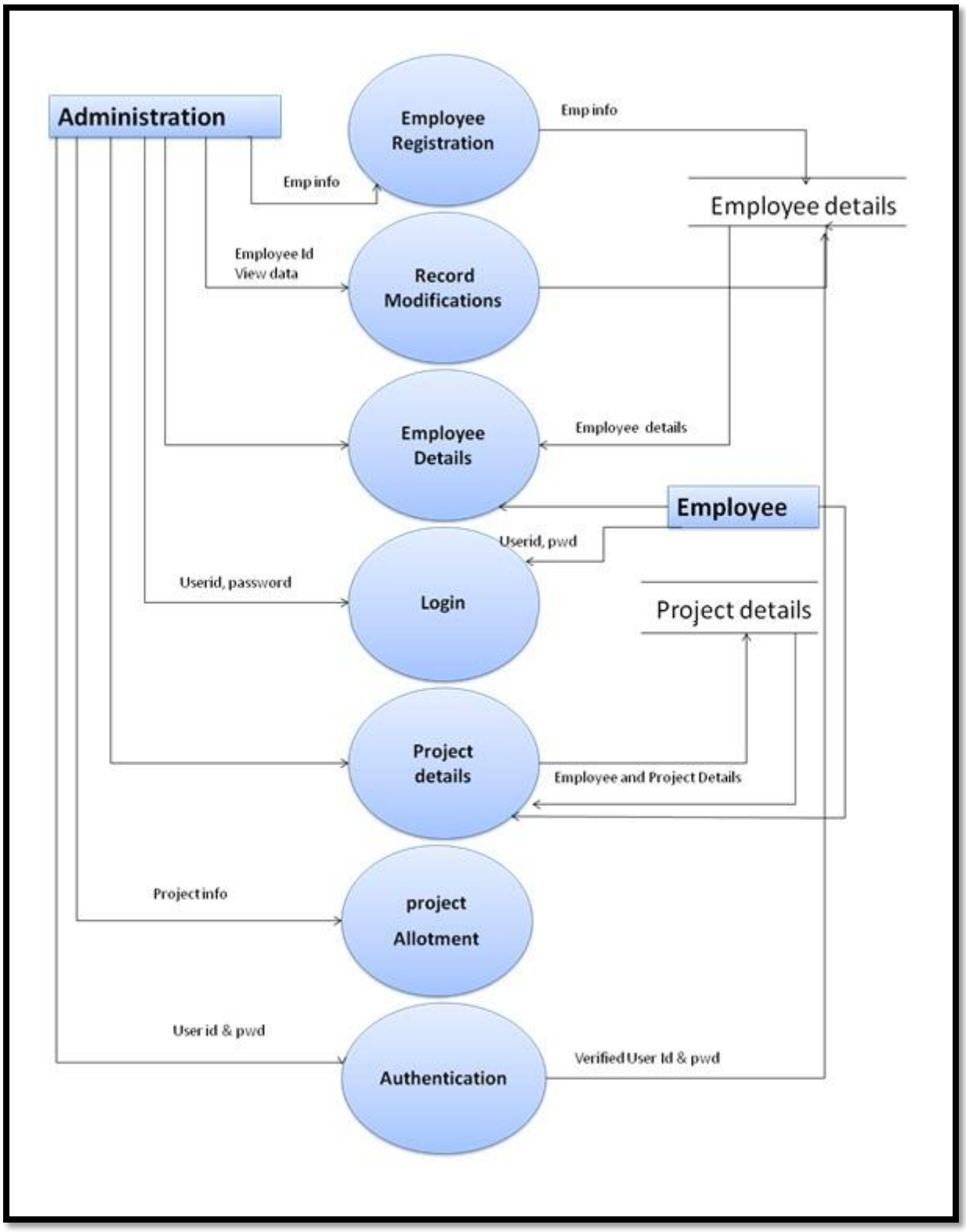

Figure 3.4

4. ER Diagram

An ER diagram depicts the relationship between data objects. The object-relationship pair can be represented graphically using the Entity-Relationship Diagram. A set of primary components is identified for the ER Diagram: Data Objects, attributes, relationships and various type of indicators. The primary purpose of an ER Diagram is to represent data objects and their relationships.

Data Objects, Attributes and Relationships

Data Objects: A data object is a representation of almost any composite information that must be understood by software. By composite information we mean something that has a number of different properties or attributes.

Attributes: They define the properties of a data object and take on one of the three different characteristics. They can be used to name an instance of the data object, describe the instances or make reference to another instance in another table.

Relationship: Data objects are connected to one another in a variety of different ways. We can define a set of objects-relationships pairs that define the relevant relationship. Object-relationship pairs are bi-directional. Different data objects and their attributes are described in the data dictionary and the relationship between these data objects are given in the ER diagram in the next section.

Cardinality: The data model must be capable of representing the number of occurrences of objects in a given relationship. The cardinalities of an object-relationship are:

1). One to One (1:1) – an occurrence of object 'A' can relate to one and only on occurrence of object 'B' and the occurrence of 'B' can relate to only one occurrence of 'A'.
2). One to Many (1:N) – one occurrence of object 'A' can relate to one or many occurrences of object 'B' but an occurrence of 'B' can relate to only one occurrence of 'B'.
3). Many to Many (M:N) – an occurrence of object 'A' can relate to one or more occurrences of 'B' while an occurrence of 'B' can relate to one or more occurrence of 'A'.

Cardinality defines "the maximum number of object-relationships that can participate in a relationship".

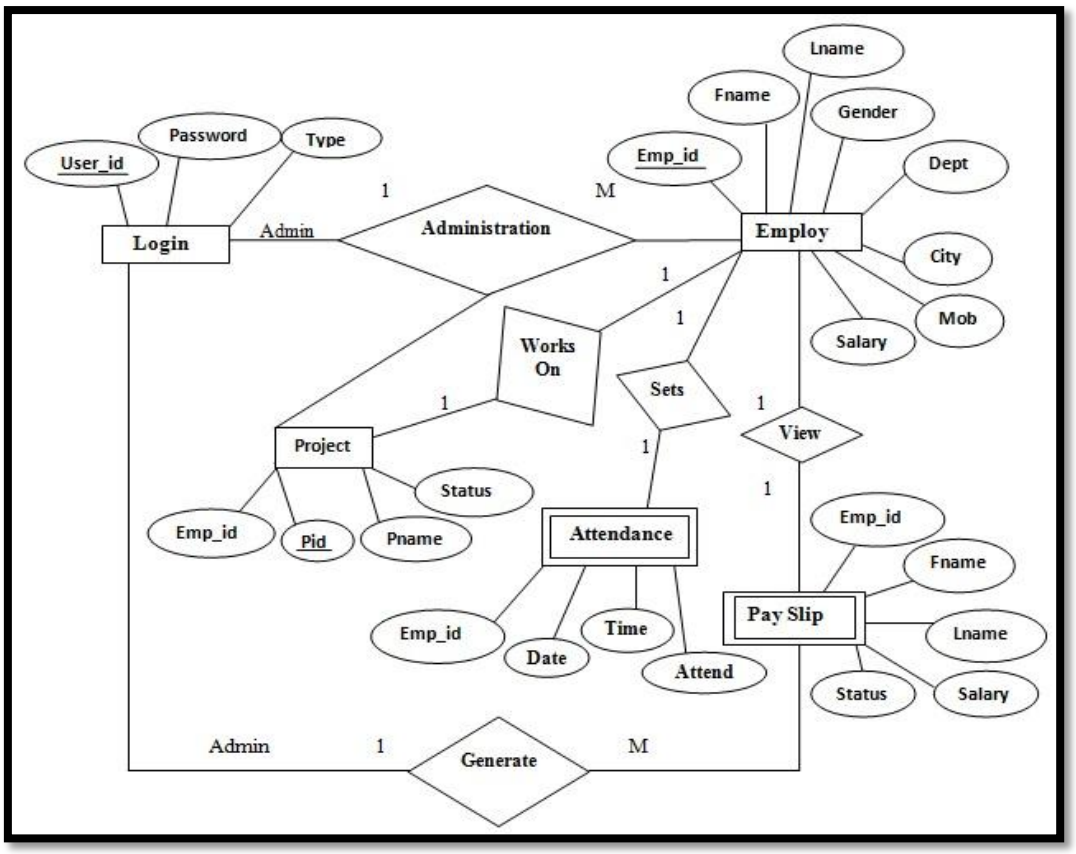

Figure 3.5

5. Database Design

The information system of "Human Resource Management" performs its function with the help of the data store in certain repositories called Databases of the system. Detailed descriptions of the various databases included in the information systems are tabulated as follows:

Login:

S.NO.	Field Name	Field Type	Field Size	Constraint	Description
1.	User Id	Varchar	20	Primary	Enter the user id
2.	Password	Varchar	20	-	Enter his password
3.	Type	Varchar	20	-	Choose the User Type

Employ:

S.NO.	Field Name	Field Type	Field Size	Constraint	Description
1.	Emp_id	Varchar	20	Primary	Enter The Employ Id.
2.	Fname	Varchar	20	-	Enter The Employ First Name.
3.	Lname	Varchar	20	-	Enter The Employ Last Name.
4.	Gender	Varchar	20	-	Enter The Gender Male/Female
5.	Dept	Varchar	20	-	Enter The Department.
6.	City	Varchar	20	-	Enter The City.
7.	Mob	Varchar	20	-	Enter The Mobile No.
8.	Salary	Number	20	-	Enter The Salary.

Project:

S.NO.	Field Name	Field Type	Field Size	Constraint	Description
1.	Emp_id	Varchar	20	-	Enter The Employ Id.
2.	Pid	Number	11	Primary	Enter The Project Id.
3.	Pname	Varchar	20	-	Enter The Project Name.
4.	Status	Varchar	20	-	Enter The Project Status

Attendance:

S.NO.	Field Name	Field Type	Field Size	Constraint	Description
1.	Emp_id	Number	20	-	Enter The Employ Id.
2.	Date	Varchar	20	Primary	Enter The Date.
3.	Time	Varchar	20	-	Enter The Time.
4.	Attend	Varchar	20	-	Enter The Attendance.

Payslip:

S.NO.	Field Name	Field Type	Field Size	Constraint	Description
1.	Emp_id	Varchar	20	Primary	Enter The Employ Id.
2.	Fname	Varchar	20	-	Enter The Employ First Name.
3.	Lname	Varchar	20	-	Enter The Employ Last Name.
4.	Salary	Number	20	-	Enter The Salary.
5.	Status	Varchar	20	-	Enter The Payslip Status.

6. Site Map

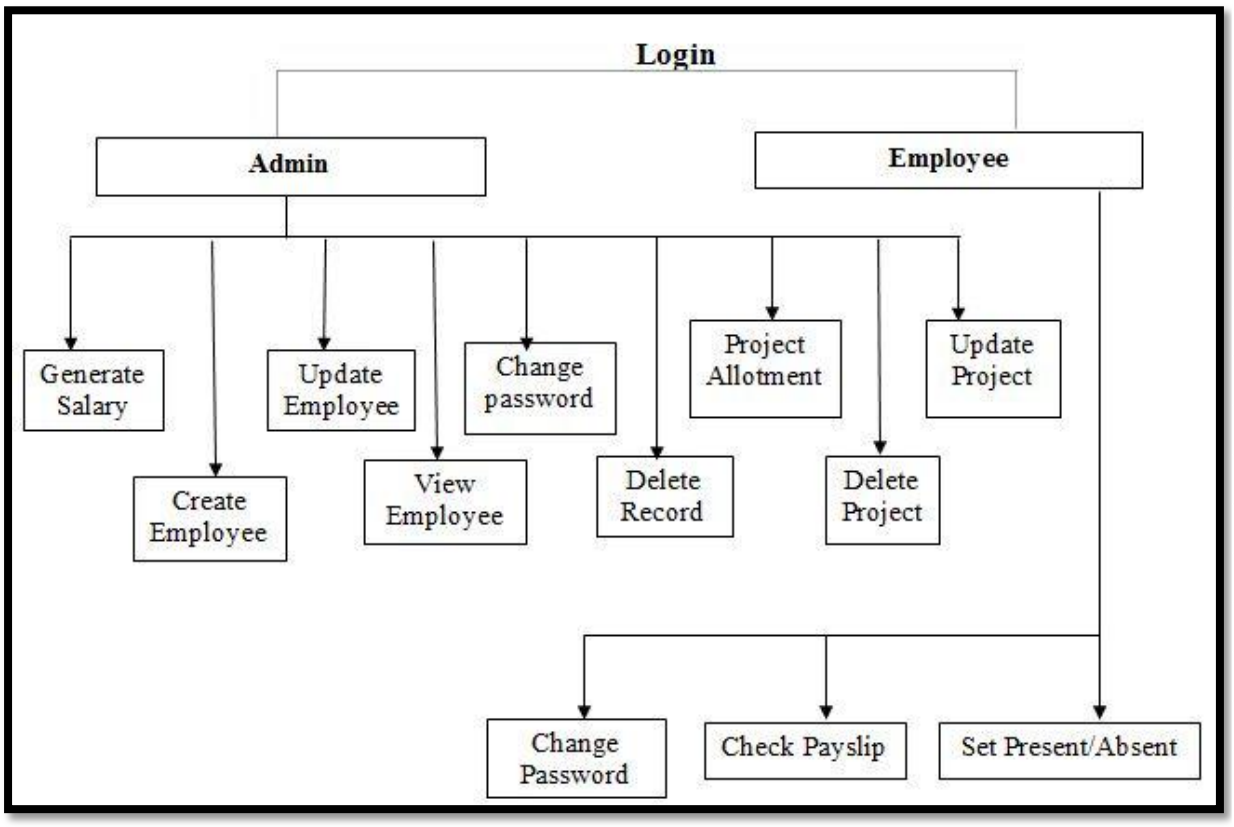

Figure 3.6

3 Interface Design

The interface design consists of the input and output source layouts. i.e. the input forms and screens and the report layouts that form as a source of outcome and income in the design and implementation of the information system under study

3.1 Input Design

The input specifications of the existing information system include the illustration of the detailed characteristics of contents included in each Input Screen and documents. The description for each graphical user interface has been mentioned.

EXISTING SYSTEM DESIGN (Graphical User Interface)

i. Main Form

Figure 3.7

Description

This is the first page that should be first displayed in which we have two hyper references login and Contact.

ii. Admin/Employ login Form

Figure 3.8

Description

This is the admin or employ login page in which admin or employ can easily login with two fields first one is email id and second one is password and choose the user type.

iii. Create Employ page

Figure 3.9

Description

This is the employ registration page in which admin can registered easily with the help of all fields like name, mobile, city etc.

iv. Update Employ Record

Figure 3.10

Figure 3.11

Description

In this page admin can update information of employee.

v. Delete Employ Record

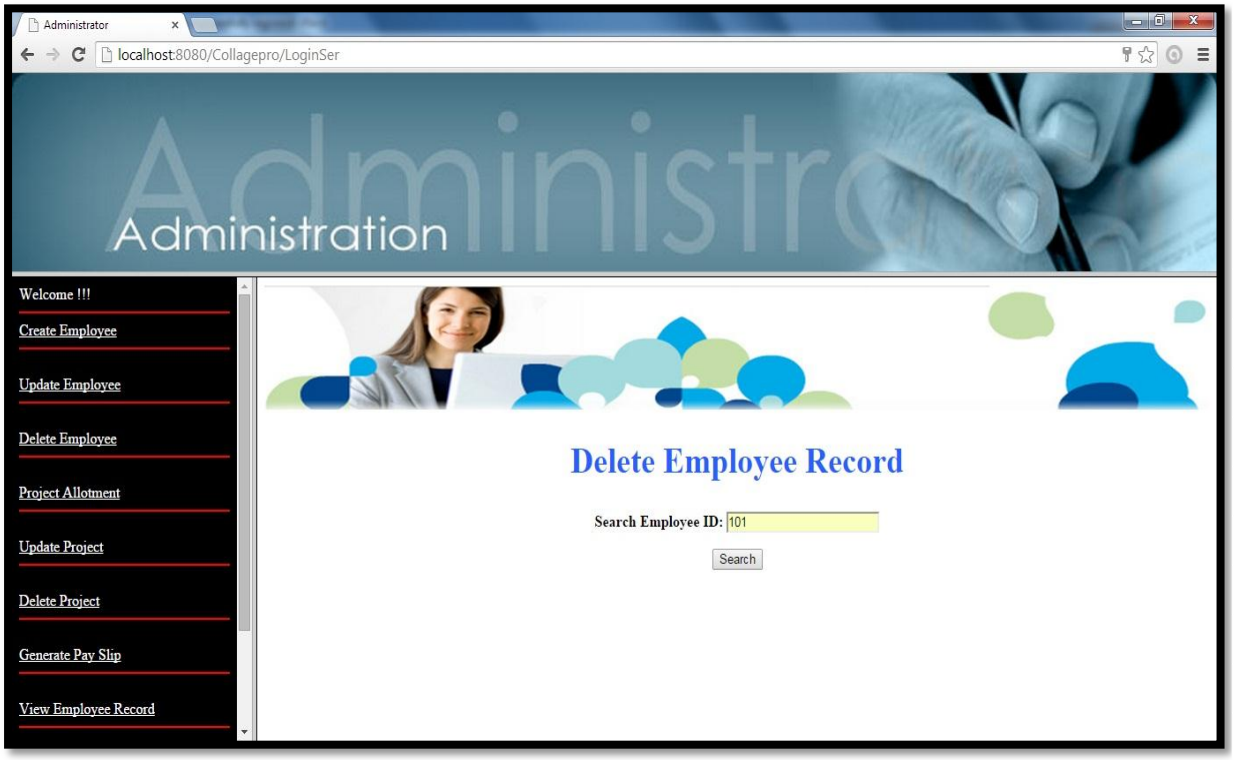

Figure 3.12

Figure 3.13

Description

In this page admin can delete the record of his employee in case employee leaves the firm.

vi. Project Allotment

Figure 3.14

Description

In this page admin can assign the project of his employee in the firm.

vii. Update Project

Figure 3.15

Figure 3.16

Description

In this page admin can update the assigned project of his employee in the firm.

viii. Delete Project

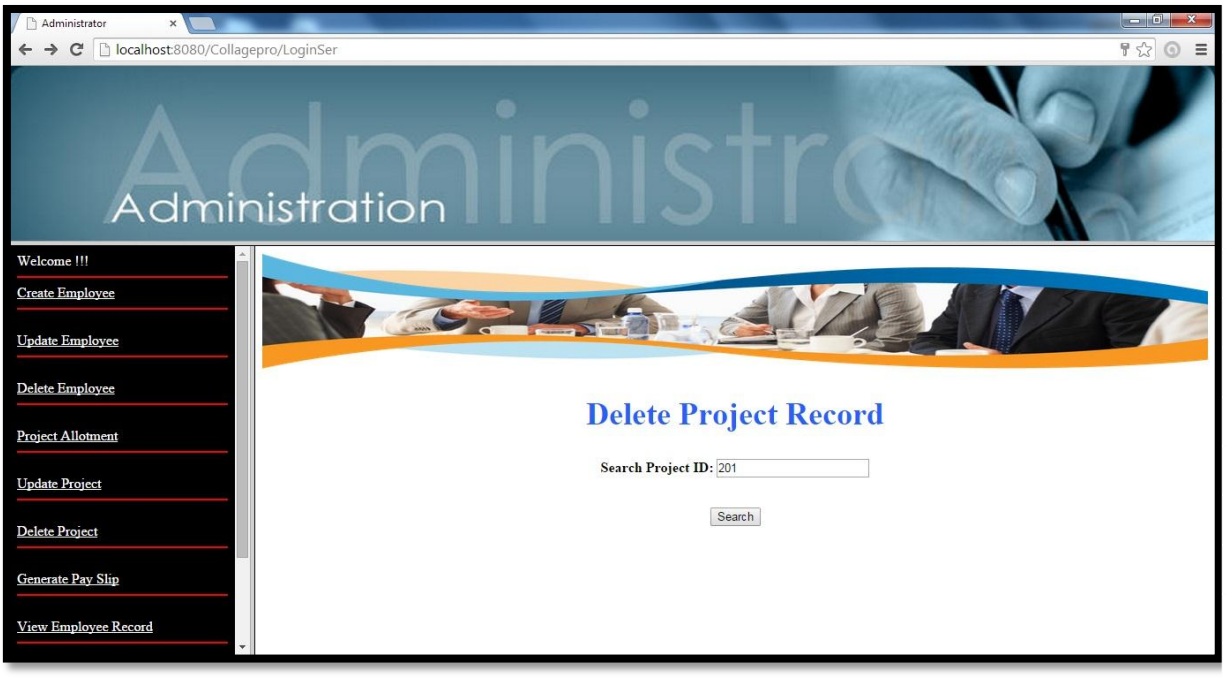

Figure 3.17

Figure 3.18

Description

In this page admin can delete the record of project completed or drop the firm.

ix. Generate PaySlip

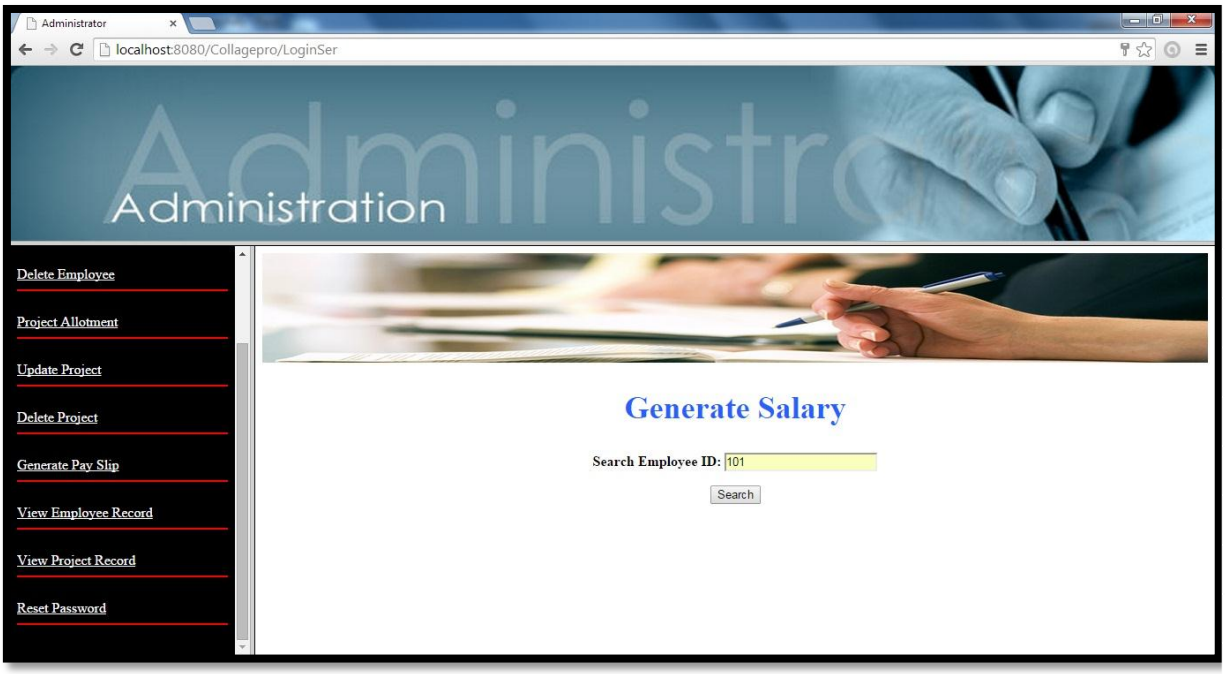

Figure 3.19

Figure 3.20

Description

In this page admin can generate the salary of his employ.

x. View Employ Record

Figure 3.21

Description

In this page admin can view the record of his/her employee working in his/her firm.

xi. View Project Record

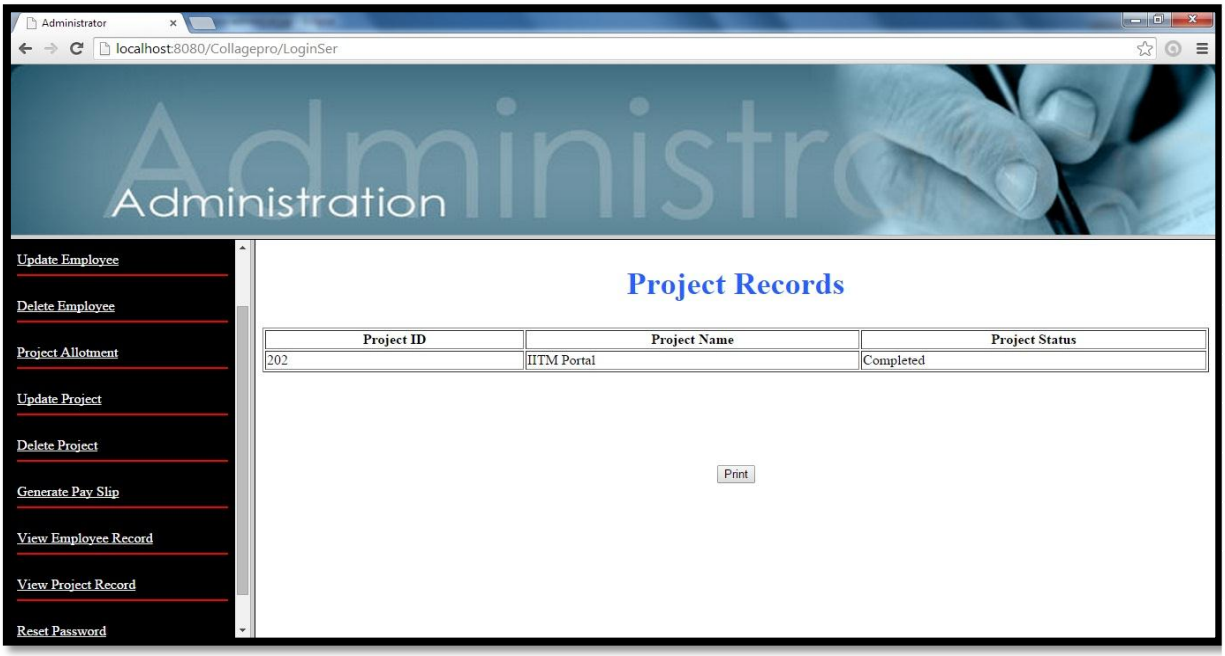

Figure 3.22

Description

In this page admin can view the project record.

xii. Change Password

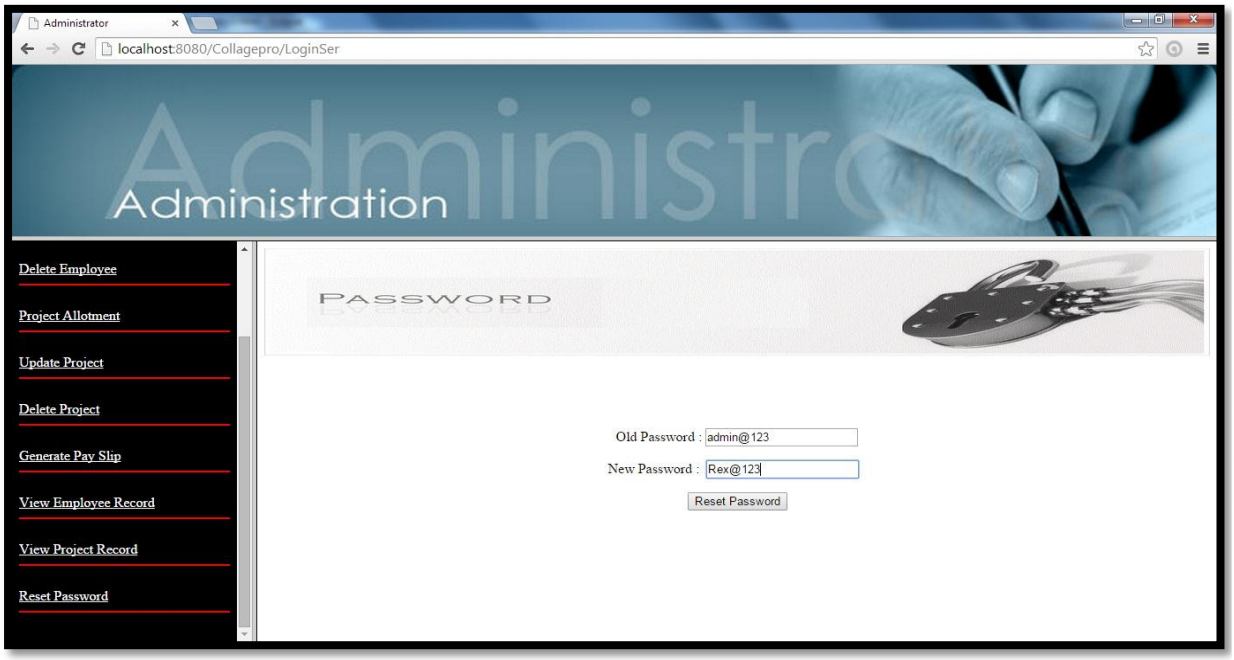

Figure 3.23

Description

In this page is meant for changing the current password in case admin want to change his password.

xiii. Employee

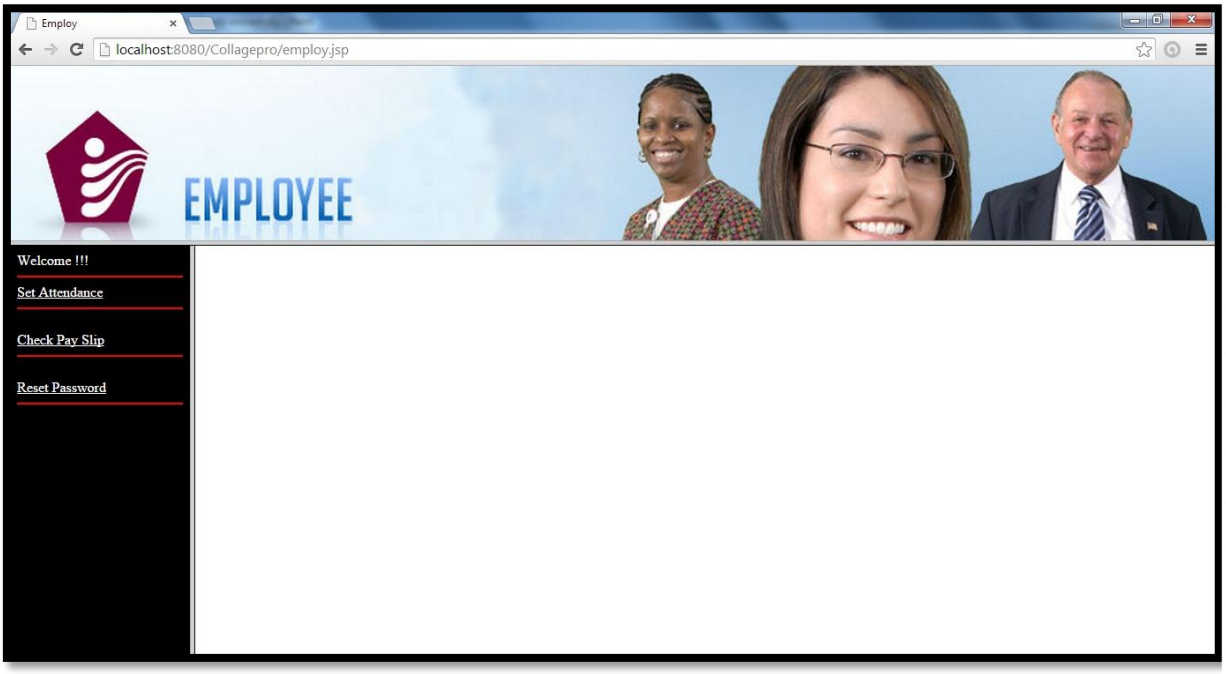

Figure 3.24

Description

After login by employee, employee can select any option shown in the left hand side of a screen these options are as set attendance, check for pay slip and reset password.

xiv. Set Attendance

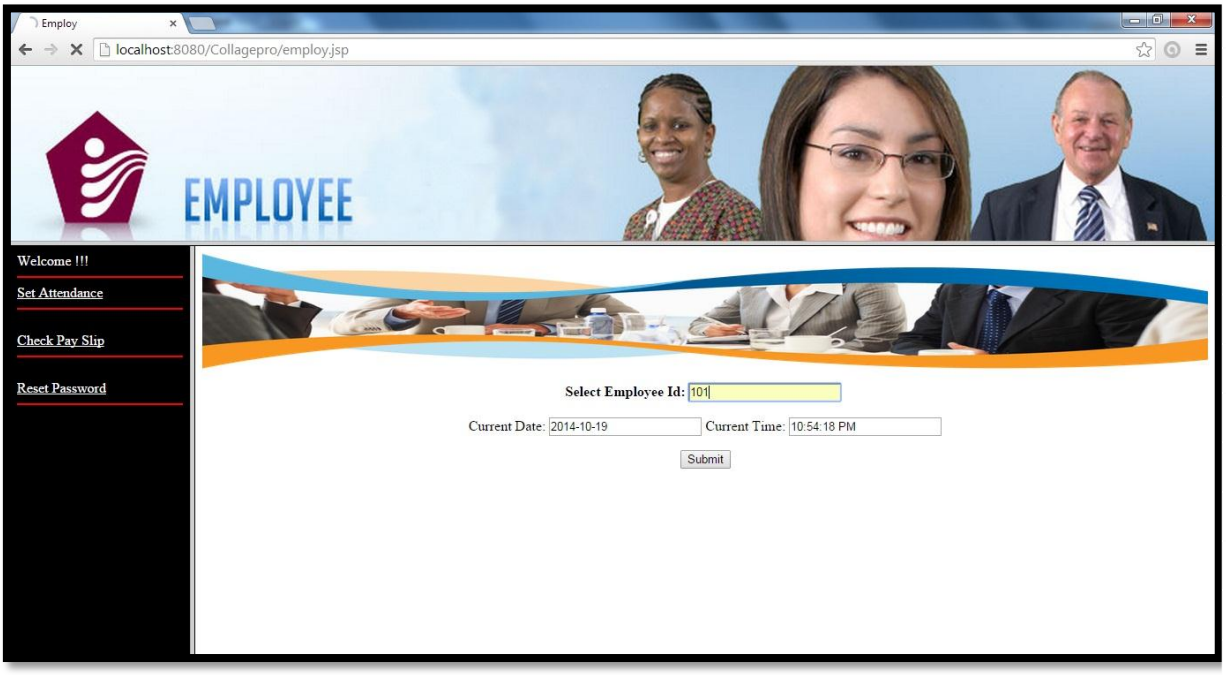

Figure 3.25

Description

In this page employee can set his attendance.

xv. Check Payslip

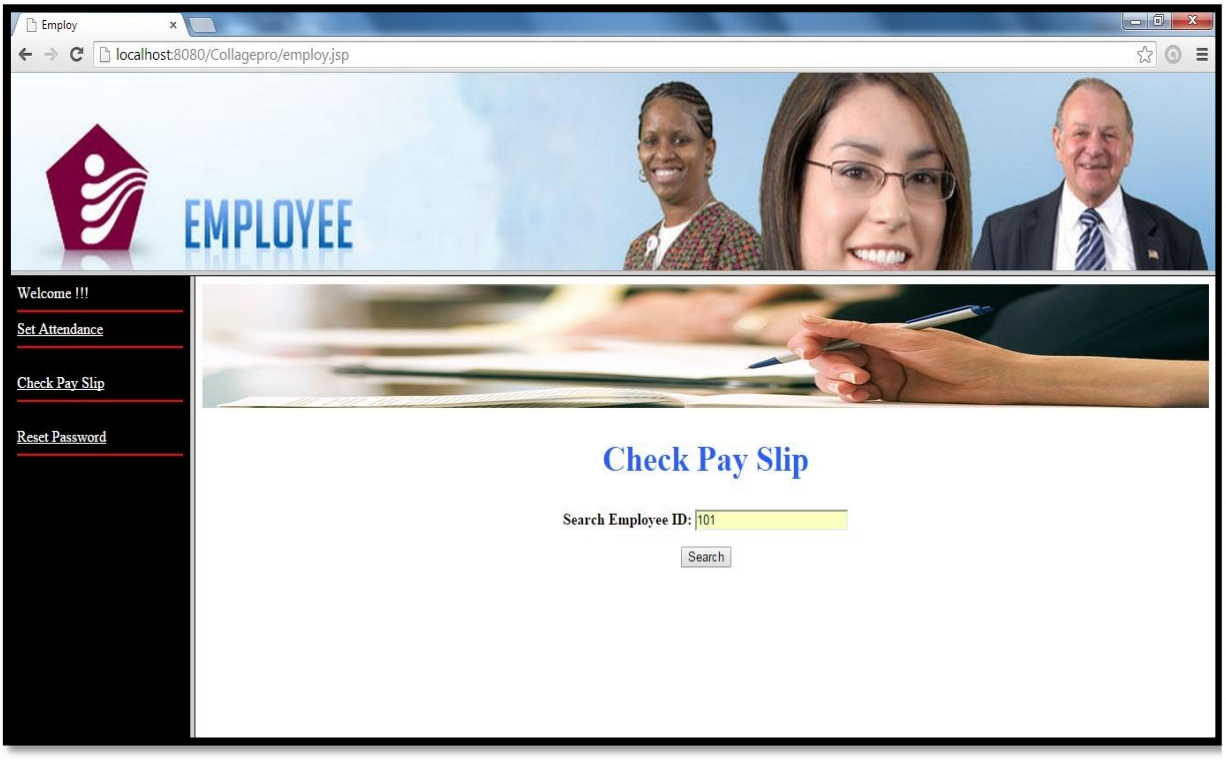

Figure 3.26

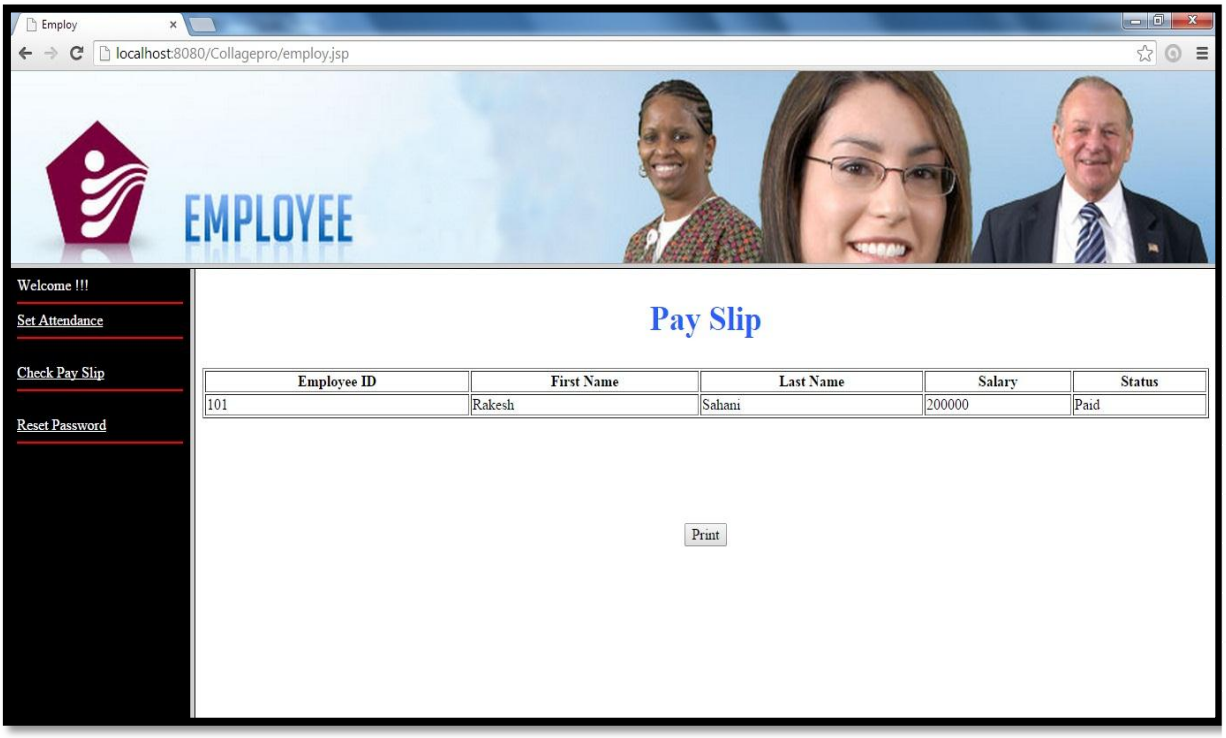

Figure 3.27

Description

In this page employee can check the payslip.

xvi. Change Password

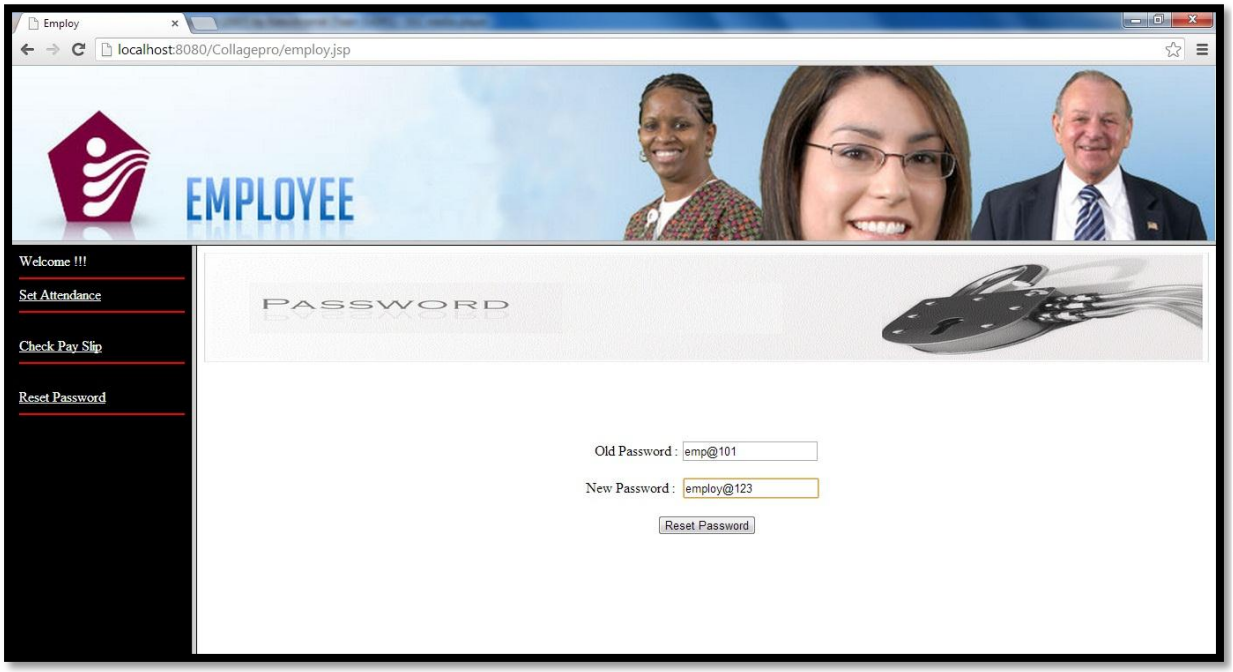

Figure 3.28

Description

In this page is meant for changing the current password in case admin want to change his password.

3.2 Output Design

i. Employ Registered

Figure 3.29

ii. Employ Updated

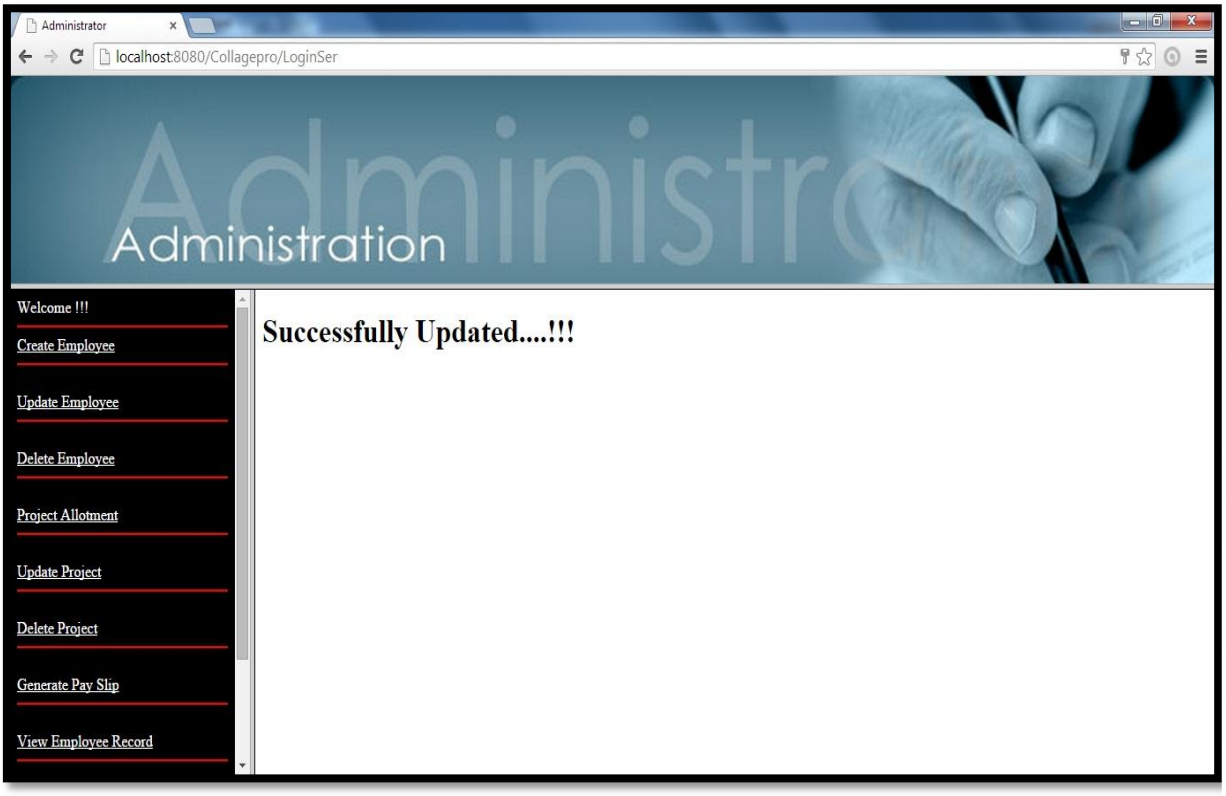

Figure 3.30

iii. Employ Deleted

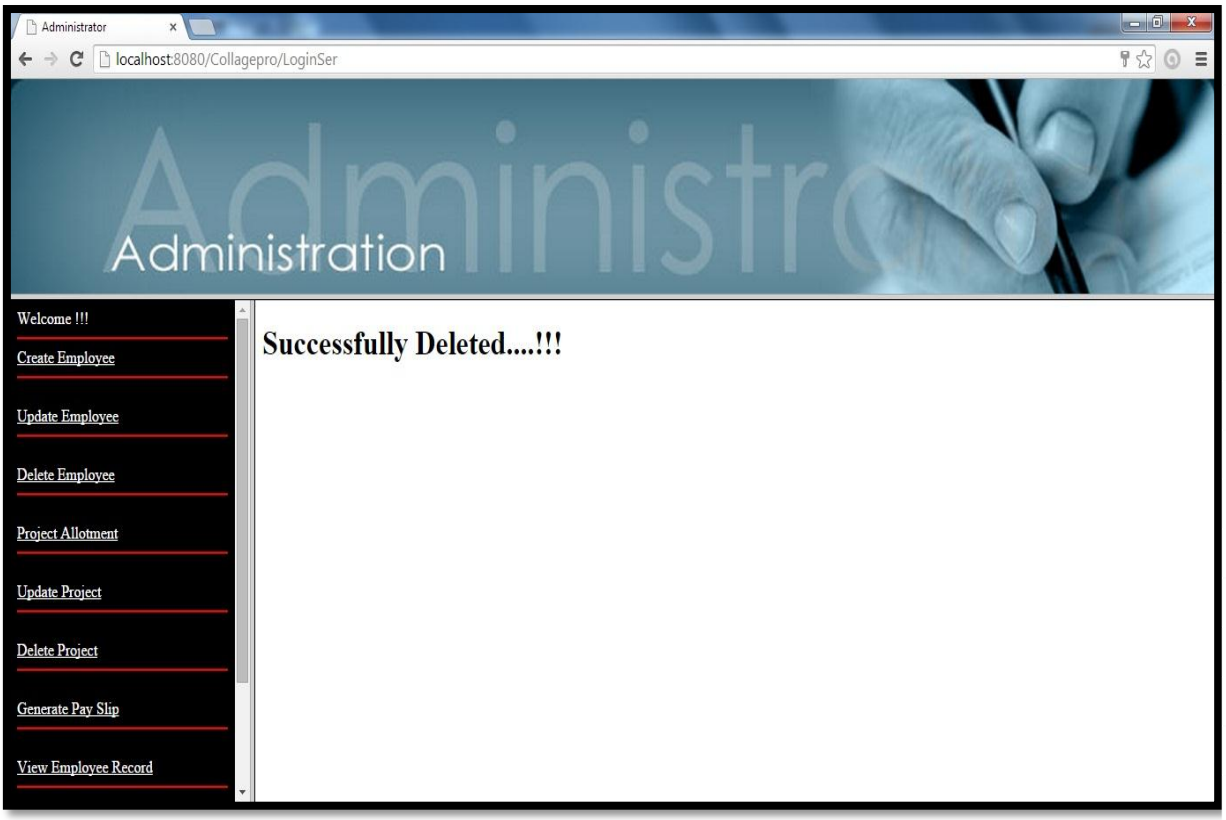

Figure 3.31

iv. Project Registered

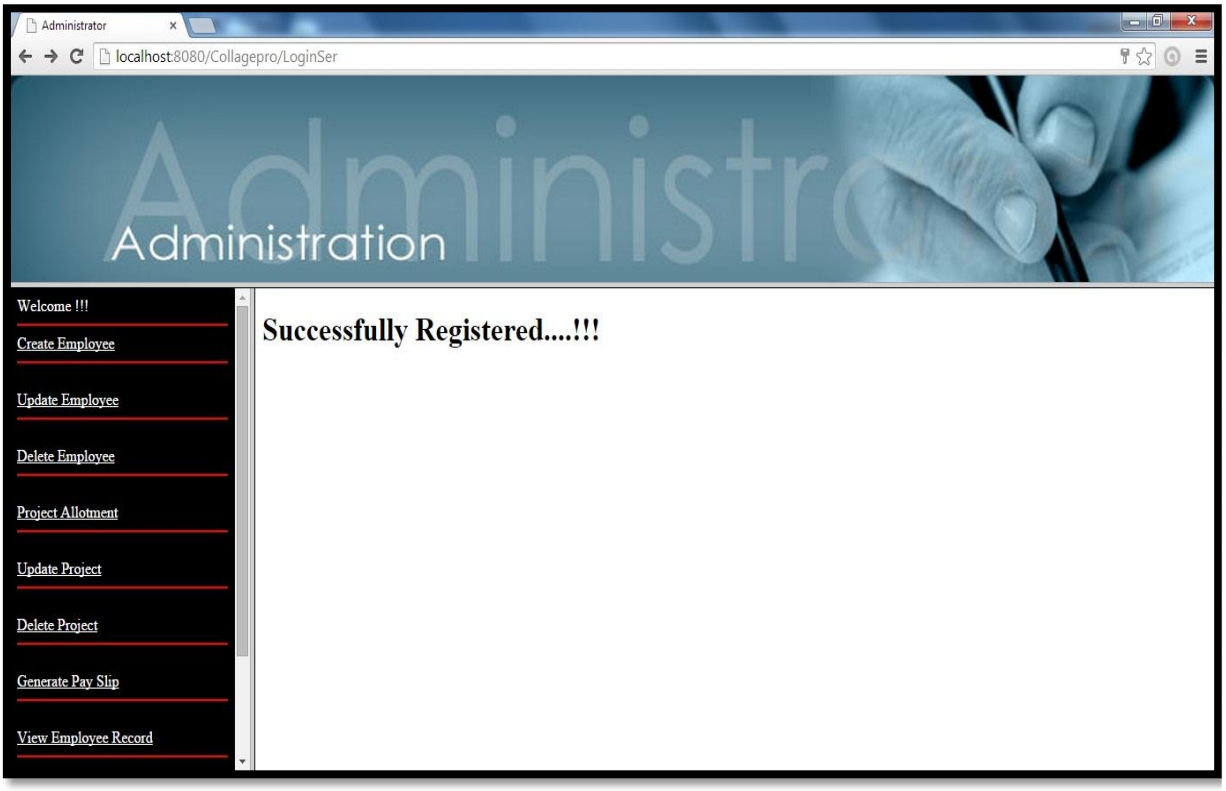

Figure 3.32

v. Project Updated

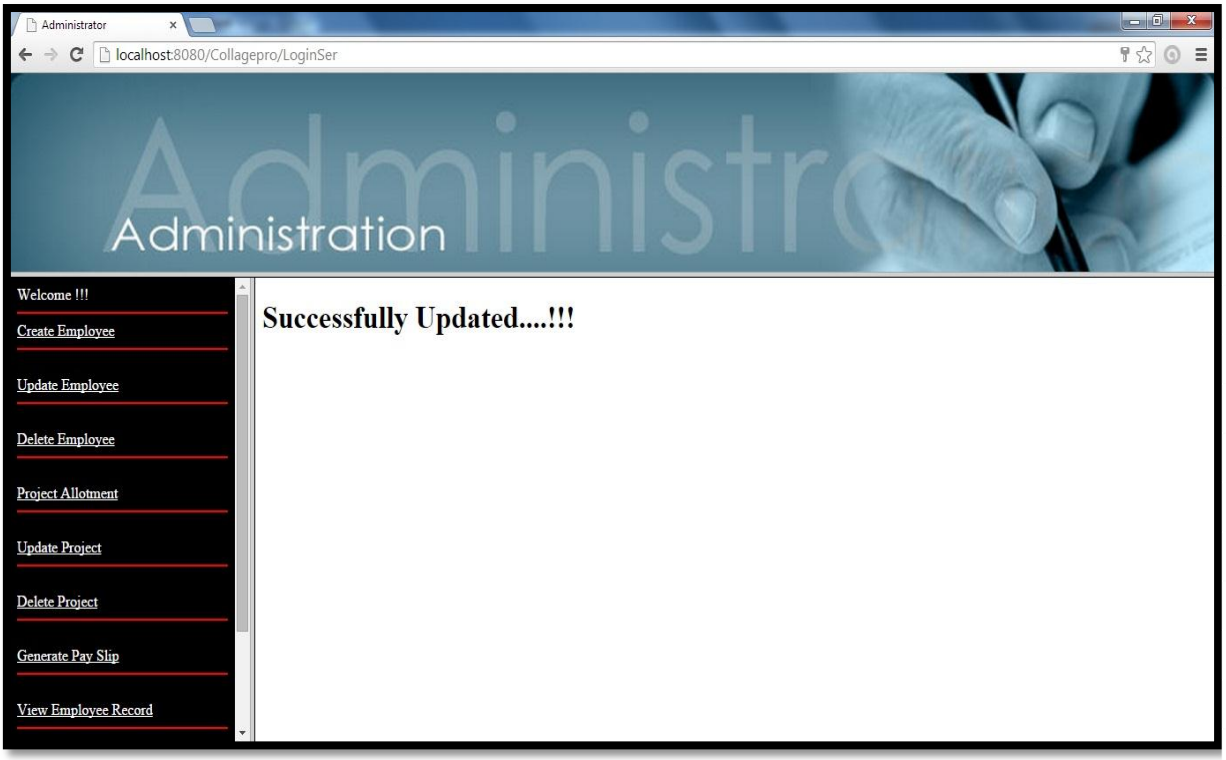

Figure 3.33

vi. Project Deleted

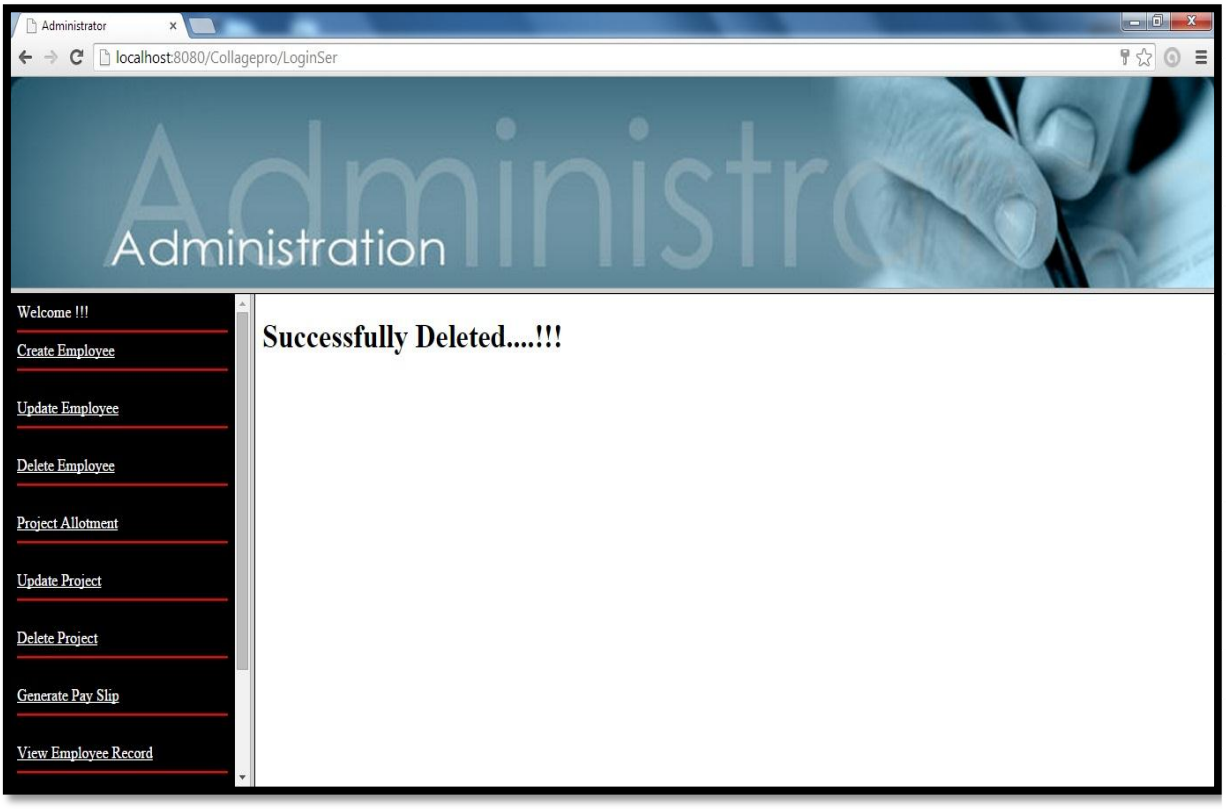

Figure 3.34

vii. Salary Generated

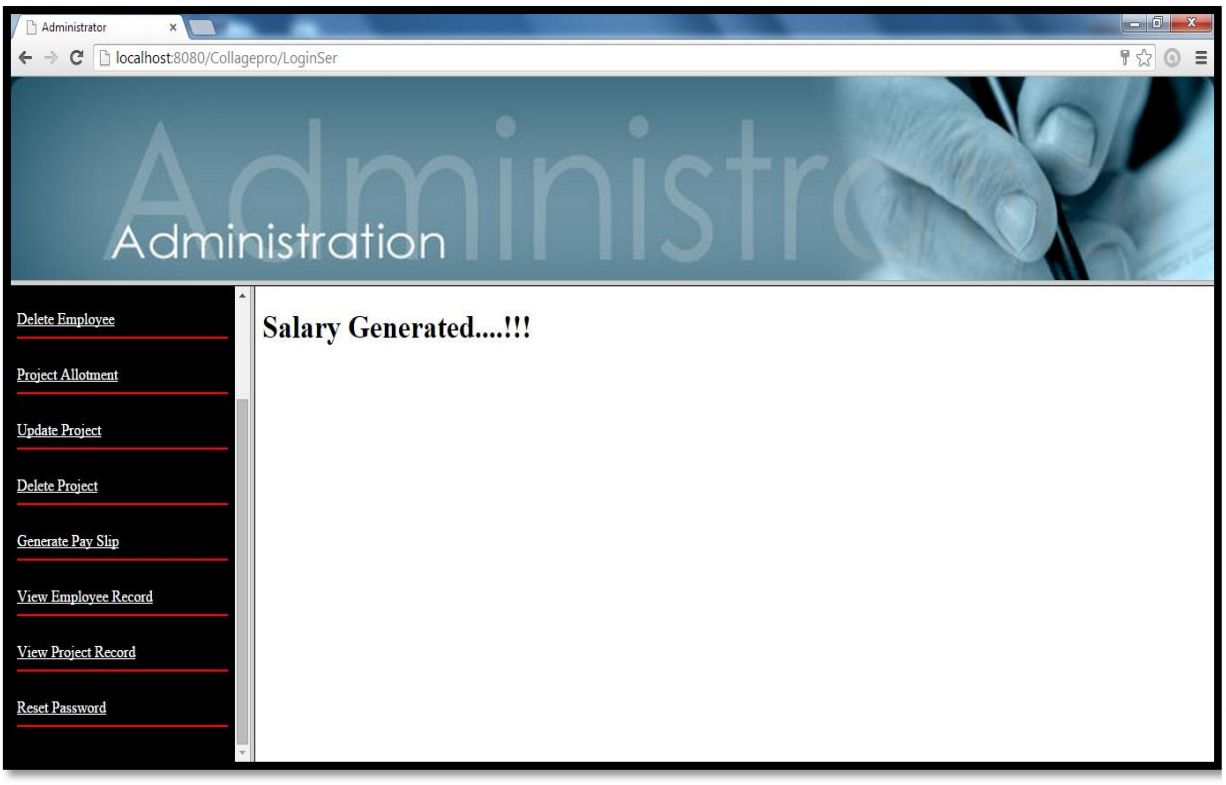

Figure 3.35

viii. Attendance Marked

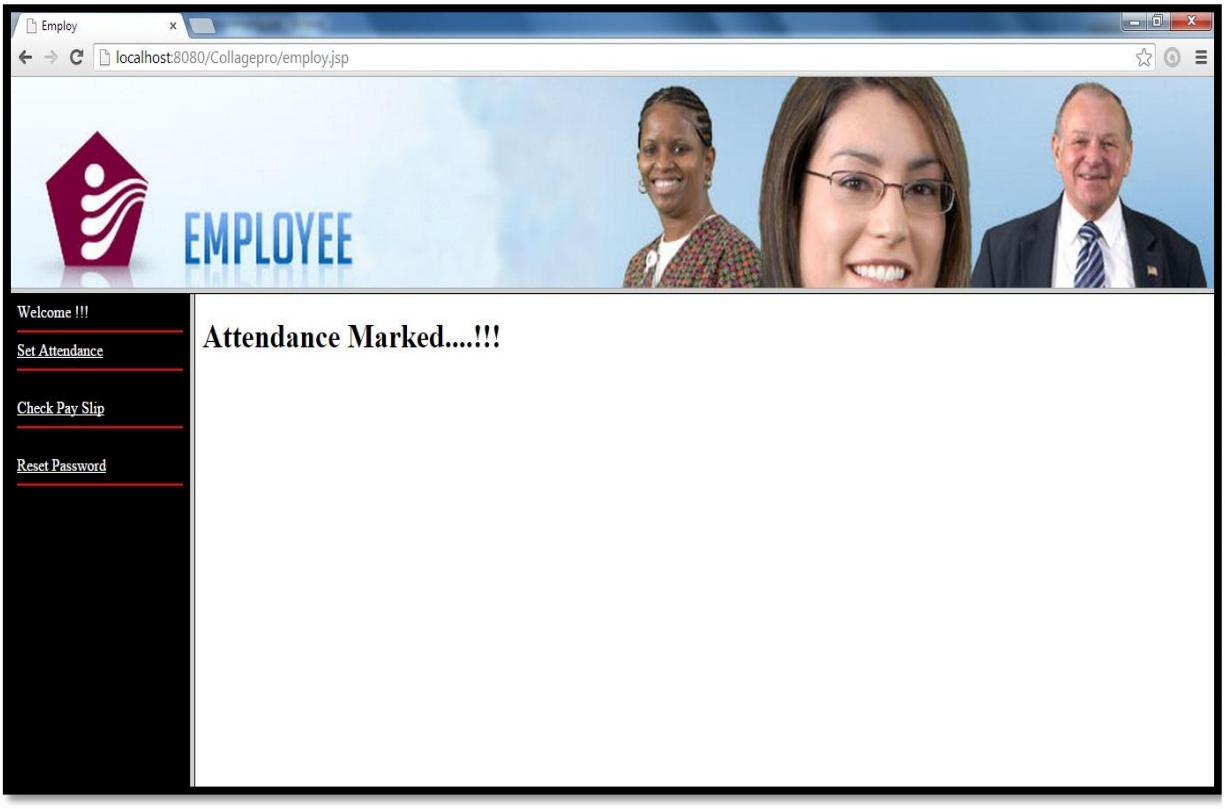

Figure 3.36

Chapter 4: System development and Implementation

Chapter – 4

System development and implementation

4.1 Introduction:

Software development is the computer programming, documenting, testing and bug fixing involved in creating and maintaining applications and frameworks involved in a software release life cycle and resulting in a software product. The term refers to a process of writing and maintaining the source code, but in a broader sense of the term it includes all that is involved between the conception of the desired software through to the final manifestation of the software, ideally in a planned and structured process. Therefore, software development may include research, new development, prototyping, modification, reuse, re-engineering, maintenance, or any other activities that result in software products.

Software can be developed for a variety of purposes, the three most common being to meet specific needs of a specific client/business (the case with custom software), to meet a perceived need of some set of potential users (the case with commercial land open source software), or for personal use (e.g. a scientist may write software to automate a mundane task). **Embedded software development**, that is, the development of embedded software such as used for controlling consumer products, requires the development process to be integrated with the development of the controlled physical product. System software underlies applications and the programming process itself, and is often developed separately.

Interface Design

The interface design consists of the input and output source layouts. i.e. the input forms and screens and the report layouts that form as a source of outcome and income in the design and implementation of the information system under study

Input Design

The input specifications of the existing information system include the illustration of the detailed characteristics of contents included in each Input Screen and documents. The description for each graphical user interface has been mentioned.

EXISTING SYSTEM DESIGN (Graphical User Interface)

i. Home page

Figure 4.1

Code:-

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Home</title>
</head>
<body>
<table width="100%" border="1" bgcolor = "skyblue" bordercolor = "white" >
<tr><th scope="col"><span class="style1"><a href = HomePage.html>Home</a>
</span></th>
<th scope="col"><span class="style1"><a href = About.html>About Us
</a></span></th>
<th scope="col"><span class="style1"><a href =Contact.html>Contact Us
</a></span></th>
<th scope="col"><span class="style1"><a href = Login.jsp>Login</a></span></th>
</tr>
</table>
<img src="images\banner3 codpy.jpg" width="100%" height="270" />
```

<marquee direction="left" height="50" loop="100" > </marquee>

Rex Pvt. Ltd is an ISO certified company providing its services in software and hardware.
This firm was registered at NYC (New York stock exchange) in 2008 to raise it funds from foreign investors.
Current turnover of the company for the FY 2012-12 was 10.4 billon. Yearly revenue recorded for 2012 was 4 billon with 2.3 of profit.
Rex Pvt. Ltd is among ten It Company of Asia. Currently 20k employees are working with Rex Group. Company is planning to expand its BPO services after a huge mark able achievement of its BPO in Surat.

</body>
</html>

ii. Admin/Employ login Form

Figure 4.2

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
   pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Login</title>
</head>
<body >
<pre>                                                    <img src="images/Logo1.jpg"
width="469" height="195"></pre>
   <h1 align="center" ><font color="blue"> REX HR Solutions Pvt. Ltd </font></h1>
<marquee direction="right" ><font color="darkblue"><font size =
15><b><i>Developing Innovations</i></b></font></marquee>
<h2 align="center">Human Resource Management System </h2>
<form  action="LoginSer"  method="post" >
<font color="black"/></font>
<center>
<input type="radio" name="admin" value="admin" />Administrator
<input type="radio" name="admin"  value="employ"/>Employee
<table border=10 bgcolor="SKYBLUE" >
<tr>
  <td>
```

```
User ID:    &amp;nbsp;&amp;nbsp;
   <input type="text"  name="user" value=""></td></tr><tr><td>
Password: <input type="password" name="pwd" ></td></tr>
<tr><td><center/><input type= "hidden"  name="page" value = "login"/>
<input type= "Submit" value = "Login"></center></td></tr>
</table>
</center>
</form>
</body>
</html>
```

Java Class File Code:-LoginDB.java

```
package com;
import java.sql.*;
public class LoginDB {
    Connection con;
    Statement st;
    ResultSet rs;
    public LoginDB()
    {
            try{
                Class.forName("com.mysql.jdbc.Driver");
                con=DriverManager.getConnection("jdbc:mysql://localhost:3306/hr","roo
                t","1234");
                }
            catch(Exception e)
            {
            System.out.println(e);
            System.out.println("connection not build");
            }
}

boolean vloginDB(String user,String pass,String type)
{
    try{
            st =con.createStatement();
            ResultSet res=st.executeQuery("select * from login where user_id='"+user+ "'
            and password='"+pass+"' and type = '"+type+"'");
            if(res.next())
            {
                    return true;
            }
        }
      catch(Exception e)
      {
            System.out.println(e);
            System.out.println("connection not build");
        }
            return false;
    }
}
```

iii. Admin Page

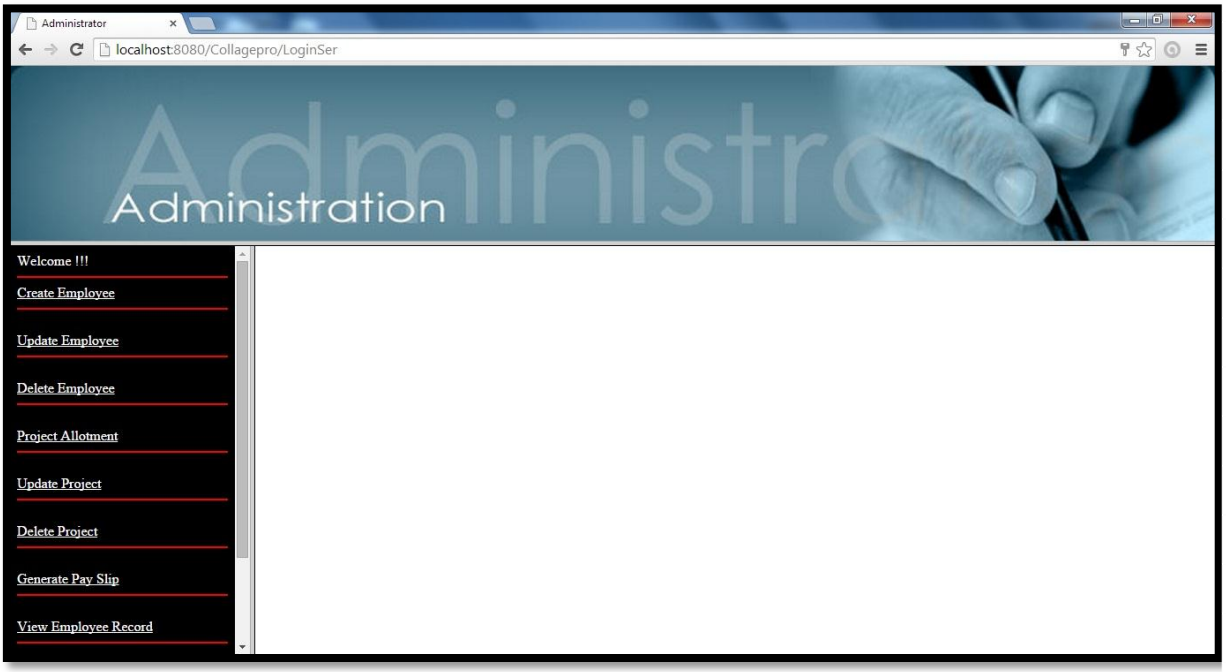

Figure 4.3

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Administrator</title>
</head>
<frameset rows="30%,70%">
<frame src="Topimg1.html" noresize="noresize" >
<frameset cols="20%,80%">
<frame  src="adminList.jsp" noresize="noresize">
<frame  src="" name ="right" noresize="noresize">
</frameset>
</frameset>
</body>
</html>
```

Code:- AdminList.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"import="java.util.*,java.sql.*"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Administrator Functions List</title>
</head>
<body bgcolor="#000000" text = "white">
<%out.println("Welcome !!!");%><br><hr color="red"/>
<a href="createEmploy.jsp" target="right"style="color:white">Create Employee </a><hr
color="red"/><br>
<a href="updateEmp.jsp" target="right"style="color:white">Update Employee</a><hr
color="red"/><br>
<a href="Delete.jsp" target="right"style="color:white">Delete Employee</a><hr
color="red"/><br>
<a href="LoginSer?page=prc" target="right"style="color:white">Project
Allotment</a><hr color="red"/><br>
<a href="updateProject.jsp" target="right"style="color:white">Update Project</a><hr
color="red"/><br>
<a href="deleteProject.jsp" target="right"style="color:white">Delete Project </a><hr
color="red"/><br>
<a href="Payslip.jsp" target="right"style="color:white">Generate Pay Slip</a><hr
color="red"/><br>
<a href="ViewEmploy.jsp" target="right"style="color:white">View Employee
Record</a><hr color="red"/><br>
<a href="ViewProject.jsp" target="right"style="color:white">View Project
Record</a><hr color="red"/><br>
<a href="updatePasswordAdmin.jsp" target="right"style="color:white">Reset
Password</a><hr color="red"/><br>
</body>
</html>
```

iv. Create Employ page

Figure 4.4

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Create Employ</title>
<style type="text/css">
<!--
.style1 {
      font-size: 36px;
      font-weight: bold;
      color: #2A5FFF;
}
-->
</style>

</head>
<body >
<jsp:useBean id="cemp" class="com.CreateEmploy" scope="request"/>
<jsp:setProperty property="*" name="cemp"/>
```

```
<%
if(cemp.getFname()!=null){
%>
<jsp:forward page="LoginSer"/>
<%} %>
<form name="frm" method="get" >
<img src="images\hrms.png" width="100%" height="124">
<h2 align="center" class="style1"> Create Employee Record </h2>
<p align="left">Employ ID :   
  <input type= "text" name = "empid" size = "4">
<p align="center">First Name :  <input type= "text" name = "fname" size =
"20">
               
          
     
Last Name : <input type= "text" name = "lname" size = "20">
<p align="center">Gender : <input type= "radio" name="gender" value= "male"> Male
  <input type= "radio" name="gender"  value= "female">Female
<p align="center">Department :   
  <select name="dept" >
    <option>IT</option>
    <option>Staffing</option>
    <option>Programming</option>
    <option>Testing</option>
   </select>
<p align="center">       City
:           
  <input type= "text" name = "city"size = "20" />
<p align="center">     Mobile No. :
  <input type= "text" name = "mob" size = "10"/></p>
<p align="center">     Salary
:        
  <input type= "text" name = "salary" size = "10"/></p>
<p align="center">User Name :  
  <input type= "text" name = "user" >
               
             
  
Password :
<input type= "text" name = "pwd"size = "10"> </p>
<p align="center">
<br>Choose User Type:   <select name="type" >
 <option>Admin</option>
 <option>Employ</option> </select>
```

```
 <br> <br>
 <input type= "hidden" name = "page" value = "create" />
  <input type= "Submit" name = "submit" value = "Registered"/>
 </p>
</form>
</body>
</html>
```

Java Class File Code:- CreateEmployDB.java

```
package com;
import java.sql.*;
public class CreateEmployDB {
Connection con;
Statement st;
ResultSet rs;

void create(CreateEmploy em)
{
    try{
            Class.forName("com.mysql.jdbc.Driver");
            con=DriverManager.getConnection("jdbc:mysql://localhost:3306/hr","root","1
            234");
            PreparedStatement ps=con.prepareStatement("insert into employ
            values(?,?,?,?,?,?,?,?)");
            ps.setInt(1, em.getEmpid());
            ps.setString(2, em.getFname());
            ps.setString(3, em.getLname());
            ps.setString(4, em.getGender());
            ps.setString(5, em.getDept());
            ps.setString(6, em.getCity());
            ps.setString(7, em.getMob());
            ps.setInt(8, em.getSalary());
            ps.executeUpdate();
            ps=con.prepareStatement("insert into login values(?,?,?)");
            ps.setString(1, em.getUser());
            ps.setString(2, em.getPwd());
            ps.setString(3, em.getType());
            ps.executeUpdate();
    }
    catch(Exception e)
    {
            System.out.println(e);
            System.out.println("connection not build");
    }
 }
}
```

Java Class File Code:- CreateEmploy.java

```
package com;
public class CreateEmploy {
    int empid,salary;
    String fname ,lname,gender,city,mob,dept,user,pwd,type;
    public String getType() {
            return type;
    }
    public void setType(String type) {
            this.type = type;
    }
    public String getUser() {
            return user;
    }
    public void setUser(String user) {
            this.user = user;
    }
    public String getPwd() {
            return pwd;
    }
    public void setPwd(String pwd) {
            this.pwd = pwd;
    }
    public int getEmpid() {
            return empid;
    }
    public void setEmpid(int empid) {
            this.empid = empid;
    }
    public int getSalary() {
            return salary;
    }
    public void setSalary(int salary) {
            this.salary = salary;
    }
    public String getFname() {
            return fname;
    }
    public void setFname(String fname) {
            this.fname = fname;
    }
```

```
    public String getLname() {
            return lname;
    }
    public void setLname(String lname) {
            this.lname = lname;
    }
    public String getGender() {
            return gender;
    }
    public void setGender(String gender) {
            this.gender = gender;
    }
    public String getCity() {
            return city;
    }
    public void setCity(String city) {
            this.city = city;
    }
    public String getMob() {
            return mob;
    }
    public void setMob(String mob) {
            this.mob = mob;
    }
    public String getDept() {
            return dept;
    }
    public void setDept(String dept) {
            this.dept = dept;
    }

}
```

v. Update Employ Record

Figure 4.5

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"import="java.util.*,java.sql.*"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Search Update Employ</title>
<style type="text/css">
<!--
.style1 {
      font-size: 36px;
      font-weight: bold;
      color: #2A5FFF;
}
-->
</style>
</head>
<body >
<form  action="LoginSer"  method="get" >
<img src="images\hrms.png" width="100%" height="124">
```

```
<h2 align="center" class="style1"> Update Employee Record </h2>
<p align="center"><strong>Search Employee ID: <input type= "text" name = "emp">
</strong></p>
<p align="center">
<input type= "hidden" name = "page" value = "updateSearch" />
<input type= "Submit" value = "Search" />
</p>
</form>
</body>
</html>
```

Figure 4.6

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"import="java.util.*,java.sql.*"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Update Employ</title>
<style type="text/css">
<!--
.style1 {
  font-size: 36px;
  font-weight: bold;
  color: #2A5FFF;
}
-->
</style>
</head>
<body >
<%
ResultSet rs;
%>
<form  action="LoginSer"  method="GET">
```

```
<%
try{
rs=(ResultSet)request.getAttribute("up1");
while(rs.next()){%>
<img src="images\hrms.png" width="100%" height="124">
<h2 align="center" class="style1"> Update Employee Record </h2>
<p align="center"><strong>Employee ID: <input type= "text" name =
"emp"readonly="readonly" value="<%out.println(rs.getString("Emp_Id"));%>"/>
</strong>
<p align="center">First Name :  <input type= "text" name = "fname" size
= "20"readonly="readonly" value="<%out.println(rs.getString("fname"));%>">
               
                
Last Name : <input type= "text" name = "lname" size = "20"readonly="readonly"
value="<%out.println(rs.getString("lname"));%>">
<p align="center">Department :       
  <select name="dept" size="1">
    <option>IT</option>
    <option>Staffing</option>
     <option>Programming</option>
    <option>Testing</option>
  </select>
<p align="center">City  :     
     
  <input type= "text" name = "city"  value="<%out.println(rs.getString("city"));%>"/>
</p>
<p align="center">Mobile No. :
  <input type= "text" name = "mob"
value="<%out.println(rs.getString("mob"));%>"/></p>
<p align="center">Salary :        
  <input type= "text" name = "salary"
value="<%out.println(rs.getString("salary"));%>"/>
  <br>
  <br>
<input type="hidden" name="page" value="update">
<input type="submit" value="Update">
<%} %>
  </form>
<%
}catch(Exception e){
  e.printStackTrace();
}
%>
</body>
</html>
```

Java Class File Code:- UpdateEmploy.java

```
package com;
import java.sql.*;
public class updateEmpDB {
Connection con;
PreparedStatement ps;
Statement st;
ResultSet rs;
public updateEmpDB()
{
    try{
            Class.forName("com.mysql.jdbc.Driver");
            con=DriverManager.getConnection("jdbc:mysql://localhost:3306/hr","root","1
            234");
            }
        catch(Exception e)
        {
            System.out.println(e);
            System.out.println("connection not build");
            }
    }

void updateEmploy(int id,String dept,String city ,String mob,int salary)
{
    try{
            ps=con.prepareStatement("update employ set dept ='"+dept+"',city
            ='"+city+"',mob='"+mob+"',salary = "+salary+" where emp_id="+id+"");
            ps.executeUpdate();
        }
      catch(Exception e)
      {
            System.out.println(e);
            System.out.println("connection not build");
    }
}
```

```
ResultSet USearch(int empid)
{
    ResultSet rs=null;
    try{
            st=con.createStatement();
            String query ="select * from employ where emp_id=" + empid;
            rs=st.executeQuery(query);
        }
      catch(Exception e)
       {
            System.out.println("Exception+" +e);
        }
        return rs;
    }
}
```

vi. Delete Employ Record

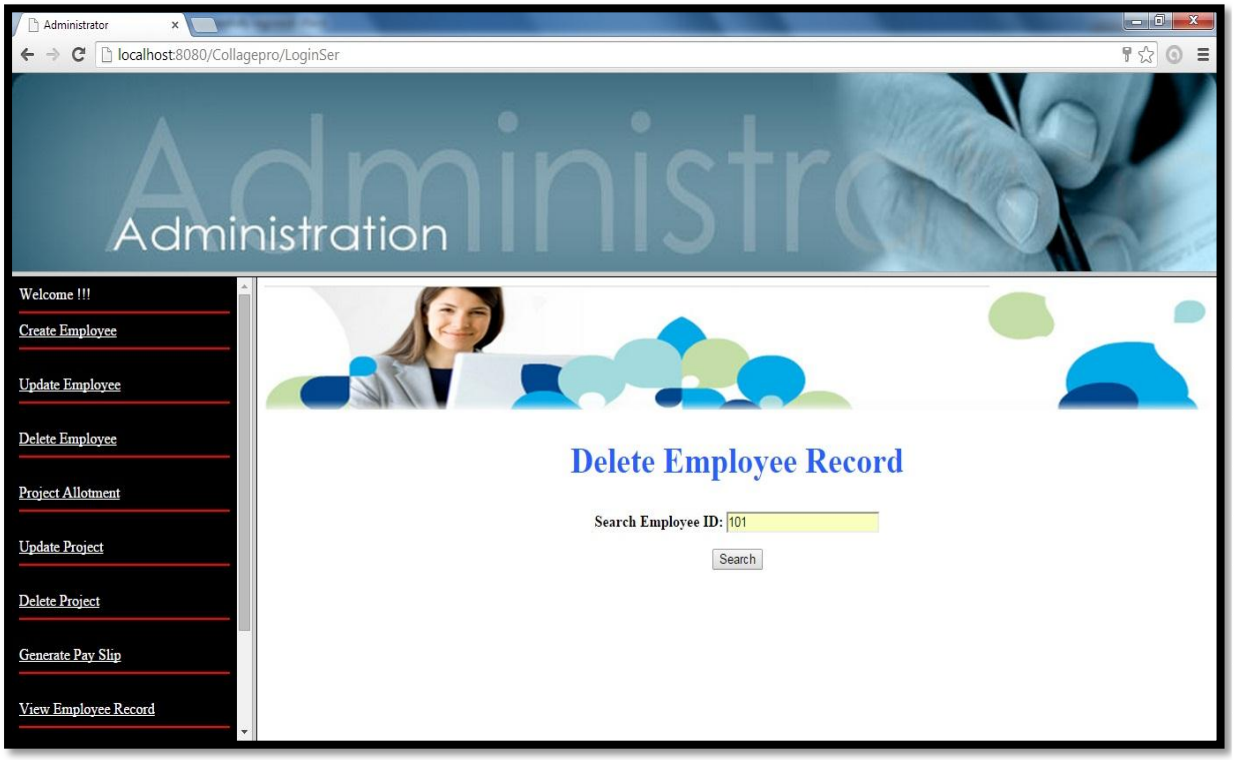

Figure 4.7

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
   pageEncoding="ISO-8859-1"import="java.util.*,java.sql.*"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Search Delete Employ</title>
<style type="text/css">
<!--
.style1 {
     font-size: 36px;
     font-weight: bold;
     color: #2A5FFF;
}
-->
</style>
</head>
<body>
<form  name="frm" action="LoginSer"  method="GET">
<img src="images\hrms.png" width="100%" height="124">
```

```
<h2 align="center" class="style1"> Delete Employee Record </h2>
<p align="center"><strong>Search Employee ID: <input type= "text" name = "emp">
</strong></p>
<p align="center">
   <input type= "hidden" name = "page" value = "SDelete" />
<p align="center"><input type="submit" name="Submit" value="Search"
align="middle"/>
</p>
</form>
</body>
</html>
```

Figure 4.8

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"import="java.util.*,java.sql.*"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Delete Employ</title>
<style type="text/css">
<!--
.style1 {
      font-size: 36px;
      font-weight: bold;
      color: #2A5FFF;
}
-->
</style>
</head>
<body >
<%
ResultSet rs;
%>
```

```
<form  action="LoginSer"  method="GET">
<%
try{
rs=(ResultSet)request.getAttribute("DE");
while(rs.next()){%>
<img src="images\hrms.png" width="100%" height="124">
<h2 align="center" class="style1"> Delete Employee Record </h2>
<p align="center"><strong>Employee ID: <input type= "text" name =
"emp"readonly="readonly" value="<%out.println(rs.getString("Emp_Id"));%>"/>
</strong>
<p align="center">First Name :  <input type= "text" name = "fname" size =
"20"readonly="readonly" value="<%out.println(rs.getString("fname"));%>">
               
                
Last Name : <input type= "text" name = "lname" size = "20"readonly="readonly"
value="<%out.println(rs.getString("lname"));%>">
<p align="center">Department :<input type= "text" name = "dept" size =
"20"readonly="readonly" value="<%out.println(rs.getString("dept"));%>">
<p align="center">City  :     
     
  <input type= "text" name = "city" readonly="readonly"
value="<%out.println(rs.getString("city"));%>"/>
</p>
<p align="center">Mobile No. :
  <input type= "text" name = "mob"readonly="readonly"
value="<%out.println(rs.getString("mob"));%>"/></p>
<p align="center">Salary :        
  <input type= "text" name = "salary"
readonly="readonly"value="<%out.println(rs.getString("salary"));%>"/>
  <br>
  <br>
<input type="hidden" name="page" value="Deleted">
<input type="submit" value="Delete">
<%} %>
    </form>
<%
}catch(Exception e){
    e.printStackTrace();
}
%>
</body>
</html>
```

Java Class File Code:- DeleteEmploy.java

```
package com;
import java.sql.*;
public class Delete {
Connection con;
PreparedStatement ps;
ResultSet rs;
Statement st;
public Delete()
{
    try{
            Class.forName("com.mysql.jdbc.Driver");
            con=DriverManager.getConnection("jdbc:mysql://localhost:3306/hr","root","1
            234");
            }
    catch(Exception e)
    {
            System.out.println(e);
            System.out.println("connection not build");
    }
}

ResultSet DeleteEmpId(int empid)
{
    ResultSet rs=null;
    try
    {
            st=con.createStatement();
            String query ="select * from employ where emp_id=" + empid;
            rs=st.executeQuery(query);
    }
    catch(Exception e)
    {
            System.out.println("Exception+" +e);
    }
    return rs;
}
```

```
    void del(String id)
    {
        try
        {
                ps=con.prepareStatement("delete from employ where emp_id = "+id+"");
                ps.executeUpdate();
        }
        catch(Exception e)
        {
                System.out.println(e);
                System.out.println("connection not build");
        }
    }
}
```

vii. Project Allotment

Figure 4.9

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"import="java.util.*,java.sql.*"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Create Project</title>
<style type="text/css">
<!--
.style1 {
                font-size: 36px;
                font-weight: bold;
                color: #2A5FFF;
}
-->
</style>
```

```
</head>
<body>
<%!ResultSet rst;
String s;
%>
<form  name="frm" action="LoginSer"  method="GET">
<img src="images/sp-project-management-img.png" width="100%" height="131"/>
<h2 align="center" class="style1"> Project Allotment </h2>
<p align="center"><strong>Select Employee ID:    </strong>
  <select name="emp"  >
  <%rst=(ResultSet)request.getAttribute("cpr");
  while(rst.next()){
                    s=rst.getString(1);
 %>
     <option value=<%=s %>><%out.println(s);%></option>
    <%} %>
    </select>
<p align="center">Project ID:     
  <input type= "text" name = "pid" value = "" />
<p align="center">Project Name:
  <input type= "text" name = "pname" value = "" /></p>
<p align="center">Project Status:
    <select name="status" size="1">
  <option>Pending</option>
   <option>Working</option>
    <option>Completed</option>
  </select>  </p>
<div align="center">
  <input type= "hidden" name = "page" value = "prjct" />
  <input type="submit" name="Submit" value="Create Project" align="middle"/>
</div>
</form>
</body>
</html>
```

Java Class File Code:- CreateProject.java

```
package com;
import java.sql.*;
public class creProject {
Connection con;
Statement st;
ResultSet rsa;
public creProject() {
try{
		Class.forName("com.mysql.jdbc.Driver");
		con=DriverManager.getConnection("jdbc:mysql://localhost:3306/hr","root",
		"1234");
		}
catch(Exception e)
{
		System.out.println(e);
		System.out.println("connection not build");
}
}

ResultSet proEmpId()
{
		String query="select Emp_Id from Employ ";
		try {
			rsa=con.createStatement().executeQuery(query);
		}
		catch (SQLException e)
		{
			e.printStackTrace();
		}
		return rsa;
}
```

```
void crprct(int emp,int pid,String pname,String status)
{
    try{
            PreparedStatement ps=con.prepareStatement("insert into project
            values(?,?,?,?)");
            ps.setInt(1, emp);
            ps.setInt(2, pid);
            ps.setString(3, pname);
            ps.setString(4, status);
            ps.executeUpdate();
    }
    catch(Exception e)
    {
            System.out.println(e);
            System.out.println("connection not build");
    }
  }
}
```

viii. Update Project

Figure 4.10

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"import="java.util.*,java.sql.*"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Search Update Project</title>
<style type="text/css">
<!--
.style1 {
                font-size: 36px;
                font-weight: bold;
                color: #2A5FFF;
}
-->
</style>
```

```
</head>
<body >
<form  action="LoginSer"  method="GET">
<img src="images/sp-project-management-img.png" width="100%" height="131" />
<h2 align="center" class="style1"> Update Project Record </h2>
<p align="center"><strong>Search Project ID:
  <input type= "text" name = "pid" value = "" />
</strong> </p>
  <p align="center"><br>
    <input type= "hidden" name = "page" value = "updateProject" />
    <input type= "Submit" name = "submit" value = "Search" />
  </p>
</form>
</body>
</html>
```

Figure 4.11

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"import="java.util.*,java.sql.*"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Update Project</title>
<style type="text/css">
<!--
.style1 {
        font-size: 36px;
        font-weight: bold;
        color: #2A5FFF;
}
-->
</style>
</head>
<body>
<%
ResultSet rs;
%>
```

```
<form  action="LoginSer"  method="GET">
<%
try{
rs=(ResultSet)request.getAttribute("uup");
while(rs.next()){%>
<img src="images/sp-project-management-img.png" width="100%" height="131" />
<h2 align="center" class="style1"> Update Project Record </h2>
<p align="center">Project ID:     
  <input type= "text" name = "pid"  readonly="readonly"
value="<%out.println(rs.getString("pid"));%>"/>
  <p align="center"> Employee ID: <input type= "text" name =
"emp"value="<%out.println(rs.getString("Emp_Id"));%>"/>
<p align="center">Project Name: <input type = "text" name = "pname" value =
"<%out.println(rs.getString("pname"));%>"/>
<p align="center">Project Status:
   <select name="status" size="1">
  <option>Pending</option>
   <option>Working</option>
    <option>Completed</option>
  </select>
<p align="center">
  <input type= "hidden" name = "page" value = "updateproj" />
  <input type="submit" name="Submit" value="Update Project" align="middle"/>
<%} %>
      </form>
<%
}catch(Exception e){
      e.printStackTrace();
}
%>
</body>
</html>
```

Java Class File Code:- UpdateProject.java

```
package com;
import java.sql.*;
public class updatProject {
Connection con;
PreparedStatement ps;
ResultSet rs;
Statement st;
public updatProject() {
try{
                    Class.forName("com.mysql.jdbc.Driver");
                    con=DriverManager.getConnection("jdbc:mysql://localhost:3306/hr","root",
                    "1234");
        }
        catch(Exception e)
        {
                System.out.println(e);
                System.out.println("connection not build");
        }
   }

ResultSet PSearch(int pid){
        ResultSet rs=null;
        try{
                 st=con.createStatement();
                String query ="select * from project where pid =" + pid;
                rs=st.executeQuery(query);
                }
        catch(Exception e)
        {
                System.out.println("Exception+" +e);
        }
        return rs;
        }
```

```
void updateProject(int pid,int id,String pname,String status){
try{
        ps=con.prepareStatement("update project set  emp_id
        ="+id+",pname='"+pname+"',status = '"+status+"' where pid="+pid+"");
        ps.executeUpdate();
        }
    catch(Exception e)
    {
            System.out.println(e);
            System.out.println("connection not build");
    }
  }
}
```

ix. Delete Project

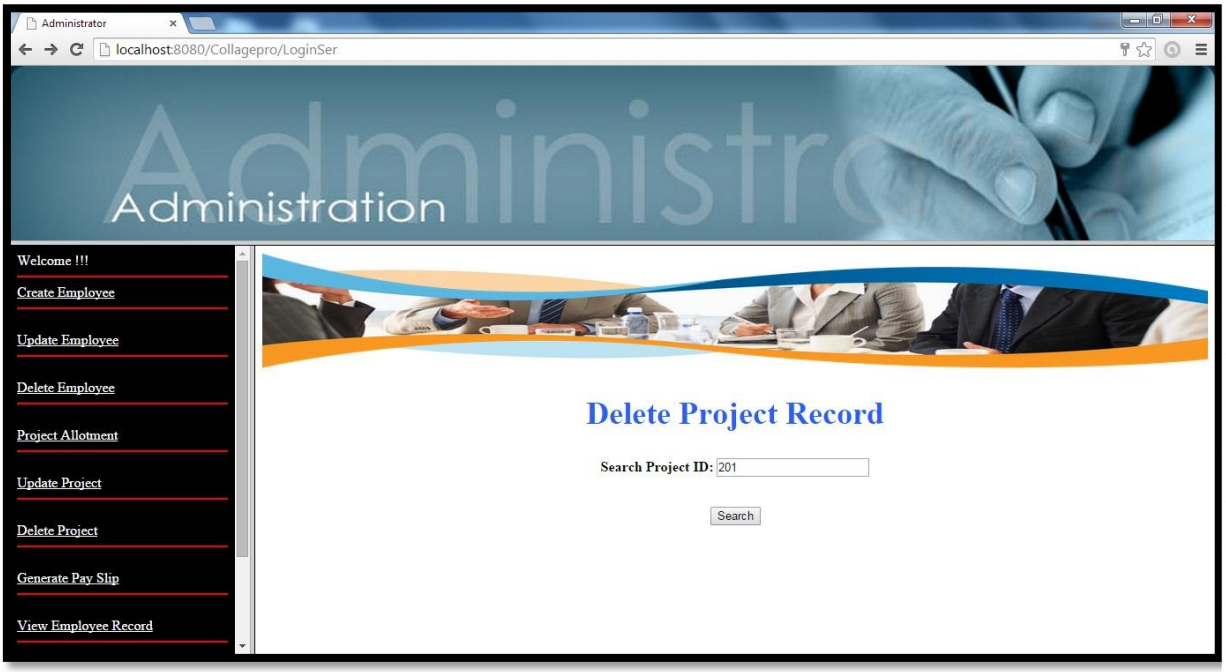

Figure 4.12

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"import="java.util.*,java.sql.*"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Search Delete Project</title>
<style type="text/css">
<!--
.style1 {
      font-size: 36px;
      font-weight: bold;
      color: #2A5FFF;
}
-->
</style>
</head>
<body >
<form  action="LoginSer"  method="GET">
<img src="images/sp-project-management-img.png" width="100%" height="131"/>
<h2 align="center" class="style1"> Delete Project Record </h2>
<p align="center"><strong>Search Project ID:
```

```
  <input type= "text" name = "pid" value = "" />
</strong> </p>
  <p align="center"><br>
    <input type= "hidden" name = "page" value = "DelProject" />
    <input type= "Submit" name = "submit" value = "Search" />
  </p>
</form>
</body>
</html>
```

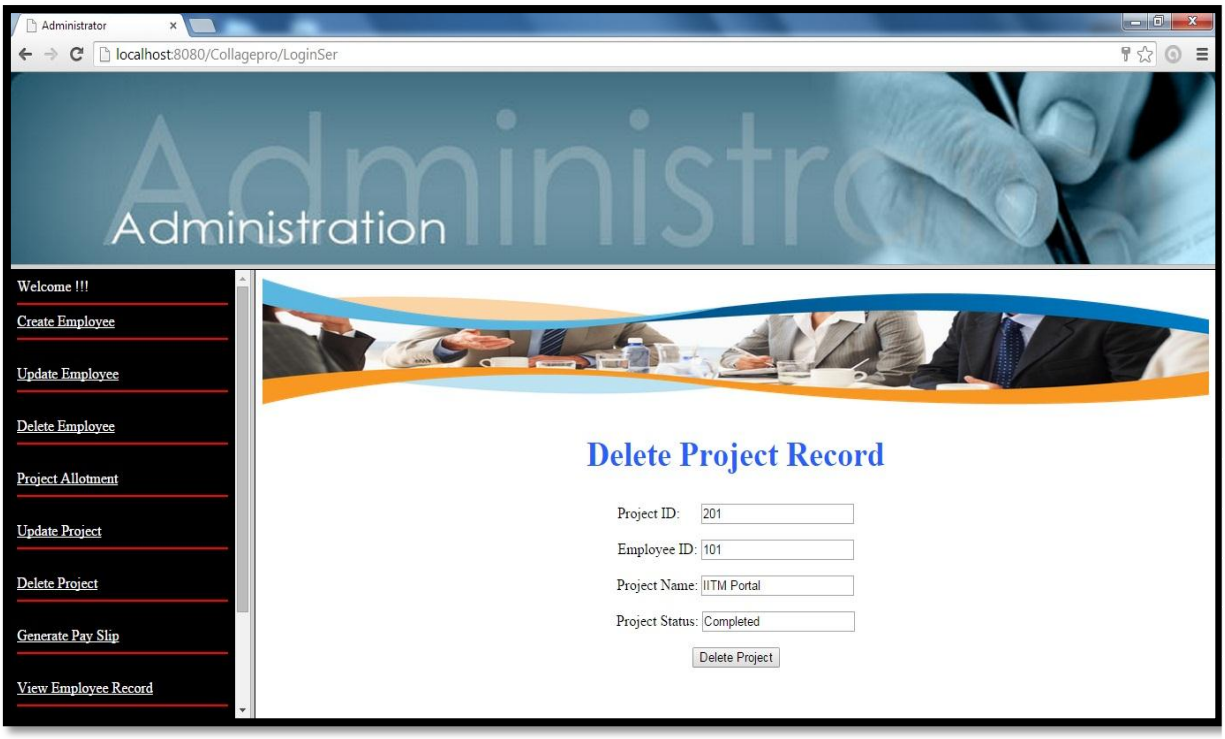

Figure 4.13

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
   pageEncoding="ISO-8859-1"import="java.util.*,java.sql.*"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Delete Project</title>
<style type="text/css">
<!--
.style1 {
     font-size: 36px;
     font-weight: bold;
     color: #2A5FFF;
}
-->
</style>
</head>
<body>
<%
ResultSet rs;
%>
```

```
<form  action="LoginSer"  method="GET">
<%
try{
rs=(ResultSet)request.getAttribute("dp");
while(rs.next()){%>
<img src="images/sp-project-management-img.png" width="100%" height="131" />
<h2 align="center" class="style1"> Delete Project Record </h2>
<p align="center">Project ID:     
  <input type= "text" name = "pid"  readonly="readonly"
value="<%out.println(rs.getString("pid"));%>"/>
  <p align="center"> Employee ID: <input type= "text" name =
"emp"value="<%out.println(rs.getString("Emp_Id"));%>"/>
<p align="center">Project Name: <input type = "text" name = "pname" value =
"<%out.println(rs.getString("pname"));%>"/>
<p align="center">Project Status: <input type = "text" name = "status" value =
"<%out.println(rs.getString("status"));%>"/>
<p align="center">
  <input type= "hidden" name = "page" value = "DeleteProject" />
  <input type="submit" name="Submit" value="Delete Project" align="middle"/>
<%} %>
    </form>
<%
}catch(Exception e){
    e.printStackTrace();
}
%>
</body>
</html>
```

Java Class File Code:- DeleteProject.java

```
package com;
import java.sql.*;
public class deleteProject {
    Connection con;
    Statement st;
   ResultSet rsa;
   public deleteProject() {
    try{
            Class.forName("com.mysql.jdbc.Driver");
            con=DriverManager.getConnection("jdbc:mysql://localhost:3306/hr","root","1
            234");
    }catch(Exception e)
    {
            System.out.println(e);
            System.out.println("connection not build");
    }
    }

    ResultSet PSearch(int pid){
    ResultSet rs=null;
    try{
            st=con.createStatement();
            String query ="select * from project where pid =" + pid;
            rs=st.executeQuery(query);
            }
    catch(Exception e)
    {
            System.out.println("Exception+" +e);
    }
    return rs;
    }
```

```
    void del(int pid){
    try{
            PreparedStatement ps=con.prepareStatement("delete from project where pid =
            '"+pid+"'");
            ps.executeUpdate();
     }catch(Exception e)
    {
            System.out.println(e);
            System.out.println("connection not build");
    }
    }
}
```

x. Generate PaySlip

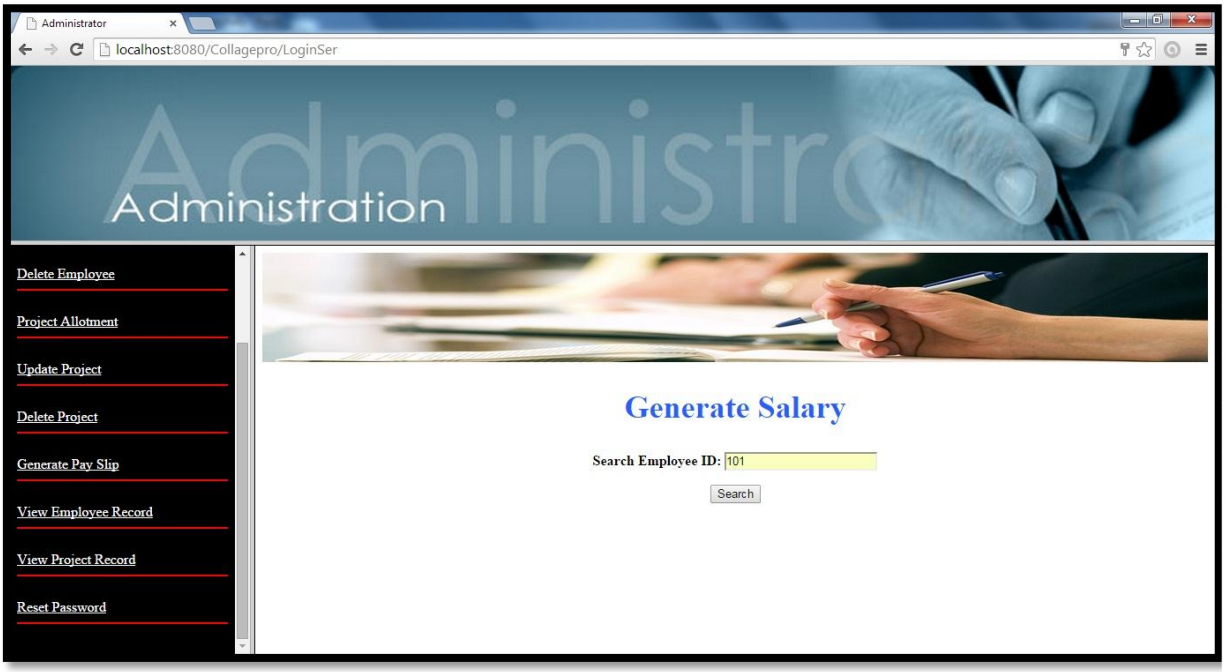

Figure 4.14

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1" import="java.util.*,java.sql.ResultSet"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Search Pay Slip</title>
<style type="text/css">
<!--
.style1 {
      font-size: 36px;
      font-weight: bold;
      color: #2A5FFF;
}
-->
</style>
</head>
<body>
<img src="images\cga-banner-img-hand-960-2.jpg" width="100%" height="124">
<%!ResultSet rsa;
String s;
%>
```

```
<form  action="LoginSer"  method="GET" >
<h2 align="center" class="style1"> Generate Salary </h2>
<p align="center"><strong>Search Employee ID: <input type= "text" name = "emp">
</strong>
<p align="center">
  <input type= "hidden" name = "page" value = "Payslip" />
  <input type="submit" name="Submit" value="Search" align="middle"/>
</form>
</body>
</html>
```

Figure 4.15

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1" import="java.util.*,java.sql.ResultSet"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Generate Pay Slip</title>
<style type="text/css">
<!--
.style1 {
      font-size: 36px;
      font-weight: bold;
      color: #2A5FFF;
}
-->
</style>
</head>
<body>
<%
ResultSet rs;
%>
```

```
<form  action="LoginSer"  method="GET">
<%
try{
rs=(ResultSet)request.getAttribute("pay");
while(rs.next()){%>
<img src="images\cga-banner-img-hand-960-2.jpg" width="100%" height="124">
<h2 align="center" class="style1"> Generate Salary </h2>
<p align="center">Employee ID:
  <input name="emp" type="text" readonly="readonly"
value="<%out.println(rs.getString("Emp_Id"));%>"></p>
<p align="center">First Name :
  <input name="fname" type="text" readonly="readonly"
value="<%out.println(rs.getString("fname"));%>"/>  </p>
<p align="center">Last Name :
  <input name="lname" type="text" readonly="readonly"
value="<%out.println(rs.getString("lname"));%>"/>  </p>
<p align="center">Salary :
  <input name="salary" type="text" value="<%out.println(rs.getString("salary"));%>"/>
</p>
<p align="center">Salary Status :
  <select name="status" >     <option>Paid</option>     <option>Unpaid</option>
</select>
</p>
<p align="center">
  <input type= "hidden" name = "page" value = "PayslipG" />
  <input type="submit" name="Submit" value="Generate" align="middle"/>
</p>
<%} %>
</form>
<%
}catch(Exception e){
     e.printStackTrace();
}
%>
</body>
</html>
```

Java Class File Code:- Payslip.java

```
package com;
import java.sql.*;
public class payslip {
Connection con;
PreparedStatement ps;
Statement st;
ResultSet rs;
public payslip() {
try{
        Class.forName("com.mysql.jdbc.Driver");
        con=DriverManager.getConnection("jdbc:mysql://localhost:3306/hr","root","1
        234");
    }catch(Exception e)
    {
        System.out.println(e);
        System.out.println("connection not build");
    }
}
void PayEmploy(int empid,String fname,String lname ,int Salary,String Status){
try{
        ps=con.prepareStatement("insert into payslip values(?,?,?,?,?)");
        ps.setInt(1, empid);
        ps.setString(2, fname);
        ps.setString(3, lname);
        ps.setInt(4, Salary);
        ps.setString(5, Status);
        ps.executeUpdate();
    }catch(Exception e) {
        System.out.println(e);
        System.out.println("connection not build");
        }
    }
```

```
    ResultSet PaySearch(int empid){
    ResultSet rs=null;
    try{
            st=con.createStatement();
            String query ="select * from employ where emp_id=" + empid;
            rs=st.executeQuery(query);
         }catch(Exception e)
        {
            System.out.println("Exception+" +e);
         }
         return rs;
    }

ResultSet CheckSearch(int empid){
ResultSet rs=null;
try {
            st=con.createStatement();
            String query ="select * from payslip where emp_id=" + empid;
            rs=st.executeQuery(query);
    }
    catch(Exception e)
    {
            System.out.println("Exception+" +e);
    }
    return rs;
    }
}
```

xi. View Employ Record

Figure 4.16

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
   pageEncoding="ISO-8859-1" import="java.sql.* "%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Employ Record List</title>
<style type="text/css">
<!--
.style1 {
     font-size: 36px;
     font-weight: bold;
     color: #2A5FFF;
}
-->
</style>
</head>
<body >
<form method="GET" action = "LoginSer">
<h2 align="center" class="style1"> Employee Records</h2>
       <% Class.forName("com.mysql.jdbc.Driver");
```

```
        Connection con;
con = DriverManager.getConnection("jdbc:mysql://localhost:3306/hr","root","1234");
        Statement stmt = con.createStatement() ;
        ResultSet rs = stmt.executeQuery("select * from employ") ;
      %>
      <table border="1" width="100%" align="center">
        <TR>
          <TH>Employee ID</TH>
          <TH>First Name</TH>
                        <TH>Last Name</TH>
                        <TH>Gender</TH>
                        <TH>Department</TH>
                        <TH>City</TH>
                        <TH>Mobile</TH>
          <TH>Salary</TH>
        </TR>
        <% while(rs.next()){ %>
        <TR>
          <TD> <%= rs.getString(1) %></tD>
          <TD> <%= rs.getString(2) %></TD>
          <TD> <%= rs.getString(3) %></TD>
                        <TD> <%= rs.getString(4) %></TD>
                        <TD> <%= rs.getString(5) %></TD>
                        <TD> <%= rs.getString(6) %></TD>
                        <TD> <%= rs.getString(7) %></TD>
                        <TD> <%= rs.getString(8) %></TD>
        </TR>
        <% } %>
        </TABLE>
<br> <br> <br> <br> <br> <br>
    <center><input  name="button" type="button" class="style2"
onClick="window.print()" value="Print">
    </center>

</form>
</body>
</html>
```

xii. View Project Record

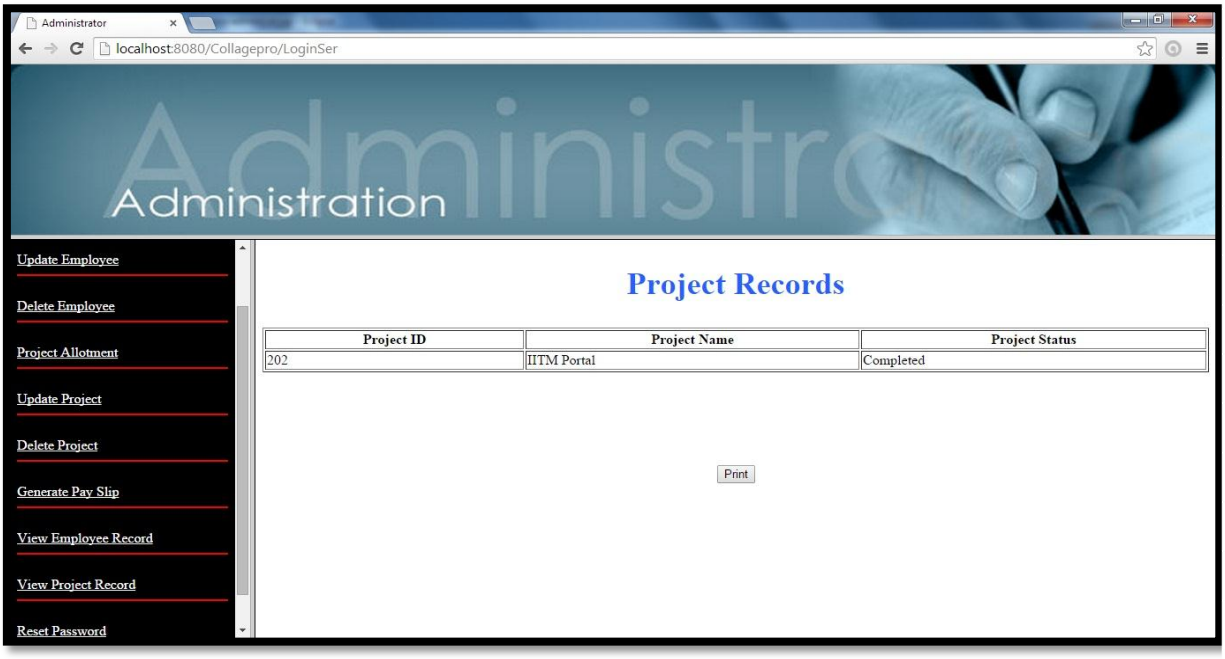

Figure 4.17

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1" import="java.sql.* "%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>View Project</title>
<style type="text/css">
<!--
.style1 {
      font-size: 36px;
      font-weight: bold;
      color: #2A5FFF;
}
-->
</style>
</head>
<body >
<form method="GET" action = "LoginSer">
<h2 align="center" class="style1"> Project Records</h2>
        <% Class.forName("com.mysql.jdbc.Driver");
           Connection con;
```

```
con = DriverManager.getConnection("jdbc:mysql://localhost:3306/hr","root","1234");
        Statement stmt = con.createStatement() ;
        ResultSet rs = stmt.executeQuery("select pid,pname,status from project") ;
     %>
    <table border="1" width="100%" align="center">
        <TR>
           <TH>Project ID</TH>
           <TH>Project Name</TH>
           <TH>Project Status</TH>
        </TR>
        <% while(rs.next()){ %>
        <TR>
           <TD> <%= rs.getString(1) %></TD>
           <TD> <%= rs.getString(2) %></TD>
           <TD> <%= rs.getString(3) %></TD>
        </TR>
        <% } %>
        </TABLE>

 <br> <br> <br> <br> <br> <br>
    <center><input  name="button" type="button" class="style2"
onClick="window.print()" value="Print">
    </center>
</form>
</body>
</html>
```

xiii. Change Password

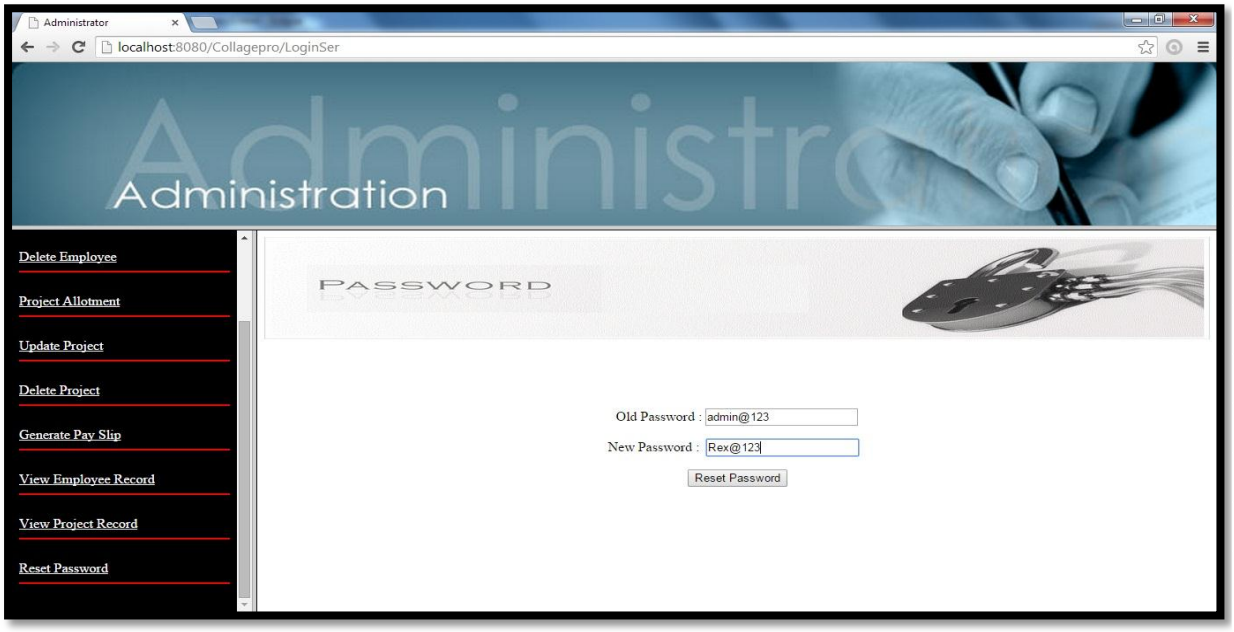

Figure 4.18

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Update Password Administrator</title>
</head>
<body>
<form  action="LoginSer"  method="GET" >
<img src="images\password.jpg" width="100%" height="124" />
<p> </p><p> </p>
<p align="center">  Old Password : <input type="text" name="old" />
<p align="center">  <label>New Password</label>  :   <input type="text"
name="new" />   
<p align="center">
<input type= "hidden" name = "page" value = "ResetPasswordAdmin" />
<input type="submit" name="Submit" value="Reset Password" />
</p>
<p>  </p>
</form>
</body>
</html>
```

Java Class File Code:- resetPassword.java

```
package com;
import java.sql.*;
public class resetPassword {
Connection con;
PreparedStatement ps;
void reset(String user,String oldpass,String newpass)
{
    try{
            Class.forName("com.mysql.jdbc.Driver");
            con=DriverManager.getConnection("jdbc:mysql://localhost:3306/hr","root","1
            234");
            ps=con.prepareStatement("update login set password ="'+newpass+"'where
            password="'+oldpass+'" and user_id = "'+user+"'");
            ps.executeUpdate();
        }
    catch(Exception e)
    {
            System.out.println(e);
            System.out.println("connection not build");
    }
  }
}
```

xiv. Employee

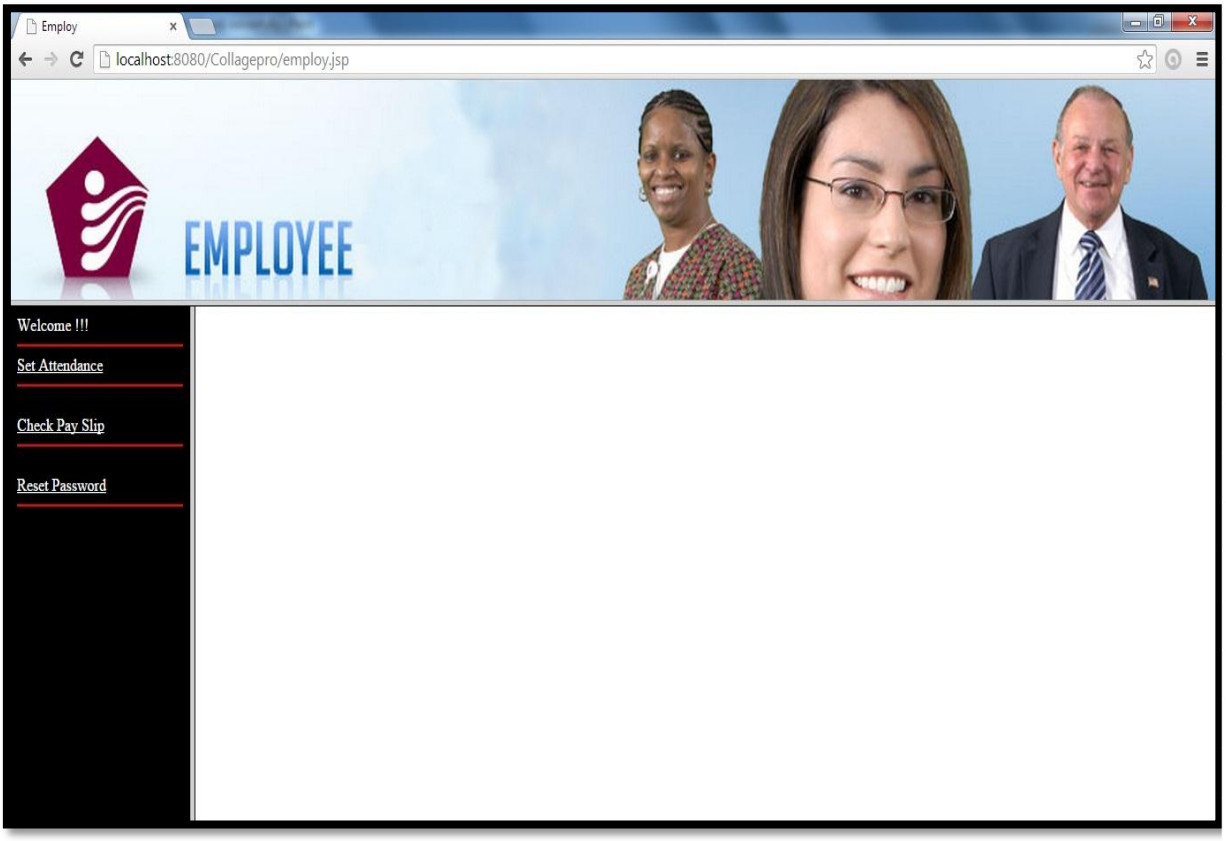

Figure 4.19

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Employ</title>
</head>
<frameset rows="30%,70%">
<frame src="Topimg.html" noresize="noresize" >
<frameset cols="15%,85%">
<frame  src="employList.jsp" noresize="noresize">
<frame  src="" name ="right" noresize="noresize">
</frameset>
</frameset>
<body>
</body>
</html>
```

Code:- EmployList.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Employ Functions List</title>
</head>
<body bgcolor="#000000" text = "white">
<%out.println("Welcome !!!");%><br><hr color="red"/>
<a href="empatt.jsp" target="right"style="color:white">Set Attendance</a><hr
color="red"/><br/>
<a href="CheckPayslip.jsp" target="right"style="color:white">Check Pay Slip</a><hr
color="red"/><br/>
<a href="updatePasswordEmploy.jsp" target="right"style="color:white">Reset
Password</a><hr color="red"/><br/>
</body>
</html>
```

xv. Set Attendance

Figure 4.20

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1" import="java.util.*,java.sql.*,java.io.*"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Employ Attendances</title>
</head>
<body>
<%!ResultSet rs;
String s;
%>
<form  action="LoginSer"  method="GET" >
<img src="images/sp-project-management-img.png" width="100%" height="131" />
<p align="center"><strong>Select Employee Id:
 </strong><input type= "text" name = "emp">
 <%response.setIntHeader("Refresh", 5); %>
 <%
   java.text.DateFormat date = new java.text.SimpleDateFormat("yyyy-MM-dd");
   java.text.DateFormat time = new java.text.SimpleDateFormat("hh:mm:ss a");
 %>
```

```
 <br> <br>Current Date: <input type="text" name="mydate" readonly="readonly"
value="<%= date.format(new java.util.Date()).toString() %>"/>

  Current Time:
  <input type="text" name="mytime" readonly="readonly" value="<%= time.format(new
java.util.Date()).toString()%>"/>
   </p>
 </p>
 <p align="center">
 <input type= "hidden" name = "page" value = "attendence" />
  <input type="submit" name="Submit" value="Submit" align="middle"/>
</p>
</form>
</body>
</html>
```

Java Class File Code:- Attendance.java

```
package com;
import java.sql.*;
public class attendence {
Connection con;
Statement st;
ResultSet rs;
PreparedStatement ps;
public attendence() {
try{
            Class.forName("com.mysql.jdbc.Driver");
            con=DriverManager.getConnection("jdbc:mysql://localhost:3306/hr","root","1
            234");
    }
    catch(Exception e)
    {
        System.out.println("ReINSERT"+e);
        System.out.println("connection not build");
    }
}

ResultSet retriveEmpId(){
try{
            String query="select Emp_Id from Employ";
            rs=con.createStatement().executeQuery(query);
    }
    catch(Exception e)
    {
        System.out.println(e);
        System.out.println("connection not build");
    }
    return rs;
}

void atten(String empid,String date,String time,String attend){
System.out.println("empid="+empid+"date="+date+"attaend="+attend);
try{
      ps=con.prepareStatement("insert into attendence values(?,?,?,?)");
      ps.setString(1, empid);
      ps.setString(2,date);
      ps.setString(3,time);
      ps.setString(4, attend);
      ps.executeUpdate();
    }
```

```
        catch(Exception e)
         {
             System.out.println("ReINSERT"+e);
             System.out.println("connection not build");
         }
    }
}
```

xvi. Check Payslip

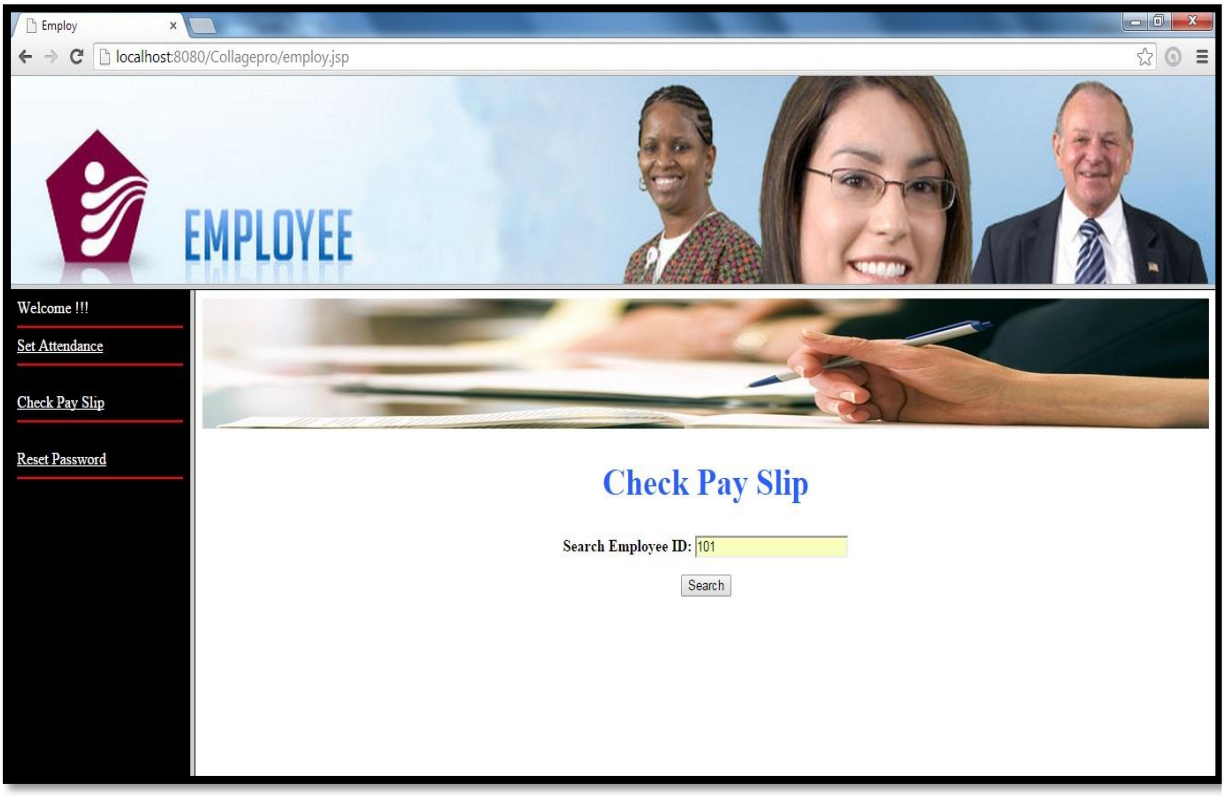

Figure 4.21

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1" import="java.util.*,java.sql.ResultSet"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Search Pay Slip</title>
<style type="text/css">
<!--
.style1 {
      font-size: 36px;
      font-weight: bold;
      color: #2A5FFF;
}
-->
</style>
</head>
<body>
<img src="images\cga-banner-img-hand-960-2.jpg" width="100%" height="124">
```

```
<%!ResultSet rsa;
String s;
%>
<form  action="LoginSer"  method="GET" >
<h2 align="center" class="style1"> Check Pay Slip </h2>
<p align="center"><strong>Search Employee ID: <input type= "text" name = "emp">
</strong>
<p align="center">
  <input type= "hidden" name = "page" value = "CheckPayslip" />
  <input type="submit" name="Submit" value="Search" align="middle"/>
</form>
</body>
</html>
```

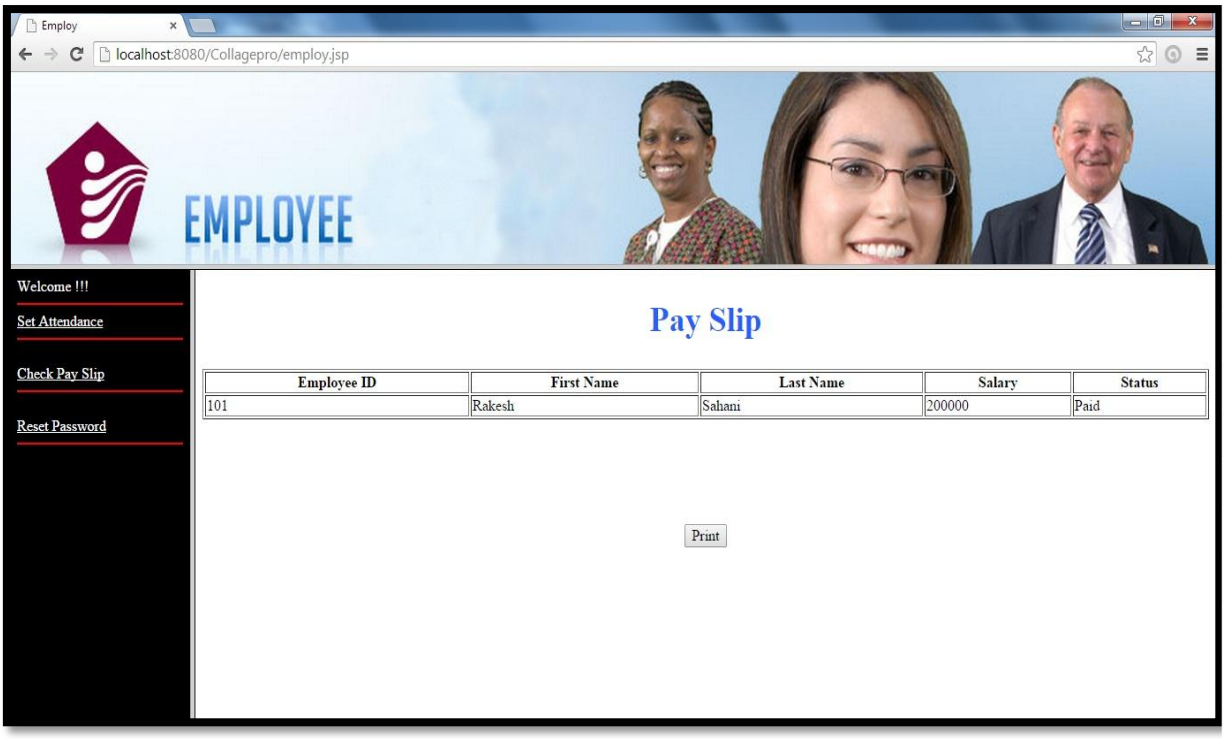

Figure 4.22

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
   pageEncoding="ISO-8859-1" import="java.util.*,java.sql.ResultSet"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Search Pay Slip</title>
<style type="text/css">
<!--
.style1 {
     font-size: 36px;
     font-weight: bold;
     color: #2A5FFF;
}
.style2 {
     font-size: 16px;
     font-family:"Times New Roman", Times, serif;
     color: #000000;
}
-->
</style>
```

```
</head>
<body>
<%
ResultSet rs;
%>
<form  action="LoginSer"  method="GET">
<%
try{
rs=(ResultSet)request.getAttribute("checkpay");%>
<h2 align="center" class="style1"> Pay Slip </h2>
 <table border="1" width="100%" align="center">
        <TR>
           <TH>Employee ID</TH>
           <TH>First Name</TH>
                        <TH>Last Name</TH>
                        <TH>Salary</TH>
                        <TH>Status</TH>
        </TR>
        <% while(rs.next()){ %>
        <TR>
           <TD> <%= rs.getInt(1) %></tD>
           <TD> <%= rs.getString(2) %></TD>
           <TD> <%= rs.getString(3) %></TD>
                        <TD> <%= rs.getInt(4) %></TD>
                        <TD> <%= rs.getString(5) %></TD>
                </TR>
 </TABLE>

</form>
   <%} %>
   <br> <br> <br> <br> <br> <br>
     <center><input  name="button" type="button" class="style2"
onClick="window.print()" value="Print">
     </center>
     <%
}catch(Exception e){
     e.printStackTrace();
}
%>
</body>
</html>
```

Java Class File Code:- Payslip.java

```
package com;
import java.sql.*;
public class payslip {
Connection con;
PreparedStatement ps;
Statement st;
ResultSet rs;
public payslip() {
try{
        Class.forName("com.mysql.jdbc.Driver");
        con=DriverManager.getConnection("jdbc:mysql://localhost:3306/hr","root","1
        234");
    }catch(Exception e)
    {
        System.out.println(e);
        System.out.println("connection not build");
    }
}
void PayEmploy(int empid,String fname,String lname ,int Salary,String Status){
try{
        ps=con.prepareStatement("insert into payslip values(?,?,?,?,?)");
        ps.setInt(1, empid);
        ps.setString(2, fname);
        ps.setString(3, lname);
        ps.setInt(4, Salary);
        ps.setString(5, Status);
        ps.executeUpdate();
    }catch(Exception e)
    {
        System.out.println(e);
        System.out.println("connection not build");
    }
}
```

```
    ResultSet PaySearch(int empid){
    ResultSet rs=null;
    try{
                st=con.createStatement();
                String query ="select * from employ where emp_id=" + empid;
                rs=st.executeQuery(query);
        }catch(Exception e)
        {
                System.out.println("Exception+" +e);
        }
        return rs;
    }

    ResultSet CheckSearch(int empid){
    ResultSet rs=null;
    try {
                st=con.createStatement();
                String query ="select * from payslip where emp_id=" + empid;
                rs=st.executeQuery(query);
        }catch(Exception e)
        {
                System.out.println("Exception+" +e);
        }
        return rs;
    }
}
```

xvii. Change Password

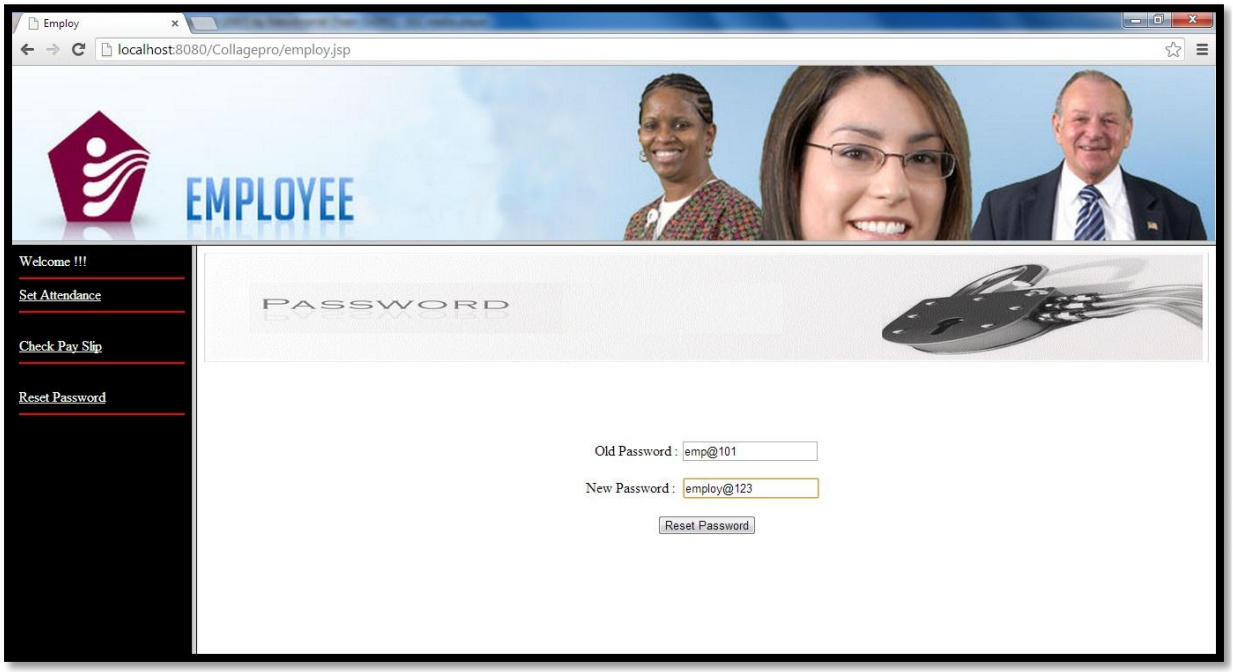

Figure 4.23

Code:-

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Update Password Employ</title>
</head>
<body>
<form  action="LoginSer"  method="GET" >
<img src="images\password.jpg" width="100%" height="124" />
<p> </p><p> </p>
<p align="center">  Old Password : <input type="text" name="old" />
<p align="center"> <label>New Password</label>  :   <input type="text"
name="new" />   
<p align="center">
<input type= "hidden" name = "page" value = "ResetPasswordEmploy" />
<input type="submit" name="Submit" value="Reset Password" />
</p><p>  </p>
</form>
</body>
</html>
```

Java Class File Code:- resetPassword.java

```
package com;
import java.sql.*;
public class resetPassword {
Connection con;
PreparedStatement ps;
void reset(String user,String oldpass,String newpass)
{
    try{
            Class.forName("com.mysql.jdbc.Driver");
            con=DriverManager.getConnection("jdbc:mysql://localhost:3306/hr","root","1
            234");
            ps=con.prepareStatement("update login set password ="'+newpass+"'where
            password="'+oldpass+'" and user_id = "'+user+"'");
            ps.executeUpdate();
        }
    catch(Exception e)
    {
            System.out.println(e);
            System.out.println("connection not build");
    }
  }
}
```

Java Servlet File Code:- LoginSer.java

```
package com;
import java.io.IOException;
import java.sql.*;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletConfig;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
public LoginSer()
{
      super();
}

public String user;
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
            String pagename = request.getParameter("page");
            RequestDispatcher rd;
            CreateEmployDB Empdb=new CreateEmployDB();
            CreateEmploy e=(CreateEmploy)request.getAttribute("cemp");
            System.out.println(pagename);

            //Create Employ Record
            if(pagename.equals("create"))
            {
            request.getAttribute("cemp");
            Empdb.create(e);
            rd=request.getRequestDispatcher("Success.html");
            rd.forward(request, response);
            }

            //Update Employ Record
            else if(pagename.equals("updateSearch"))
            {
                  updateEmpDB upEmp=new updateEmpDB();
                  int empid=Integer.parseInt(request.getParameter("emp"));
                  ResultSet rs=upEmp.USearch(empid);
                  System.out.println("in servlet empid34="+empid+"");
                  request.setAttribute("up1", rs);
                  rd=request.getRequestDispatcher("updateEmp1.jsp");
                  rd.forward(request, response);
            }
```

```
else if(pagename.equals("update"))
{
        updateEmpDB upEmp=new updateEmpDB();
        int empid=Integer.parseInt(request.getParameter("emp"));
        String Dept=request.getParameter("dept");
        String City=request.getParameter("city");
        String Mob=request.getParameter("mob");
        int Salary=Integer.parseInt(request.getParameter("salary"));
        upEmp.updateEmploy(empid, Dept, City, Mob, Salary);
        System.out.println("in servlet empid12="+empid+"");
        rd=request.getRequestDispatcher("Success1.html");
        rd.forward(request, response);
}
```

//Reset Password Admin

```
else if(pagename.equals("ResetPasswordAdmin"))
{
        String old=request.getParameter("old");
        String newp=request.getParameter("new");
        resetPassword rp = new resetPassword();
        rp.reset(user,old,newp);
        rd=request.getRequestDispatcher("Success5.html");
        rd.forward(request, response);
}
```

//Reset Password Employ

```
else if(pagename.equals("ResetPasswordEmploy"))
{
        String old=request.getParameter("old");
        String newp=request.getParameter("new");
        resetPassword rp = new resetPassword();
        rp.reset(user,old,newp);
        rd=request.getRequestDispatcher("Success5.html");
        rd.forward(request, response);
}
```

```
//Set AttenDence
else if(pagename.equals("attendence"))
{
	attendence at=new attendence();
	String empid=request.getParameter("emp");
	String mdate=request.getParameter("mydate");
	String mtime=request.getParameter("mytime");
	String x = null;
	if((empid!=null)&&(mdate!=null))
	{
		x="Present";
	}
		else
		{
			x="Absent";
		}
	System.out.println("in servlet
	empid="+empid+"date="+mdate+"attaend="+x);
	at.atten(empid, mdate, mtime, x);
	rd= request.getRequestDispatcher("Success3.html");
	rd.forward(request, response);
}

//Delete Employee
else if(pagename.equals("SDelete"))
{
	Delete DE = new Delete();
	int empid=Integer.parseInt(request.getParameter("emp"));
	ResultSet rs=DE.DeleteEmpId(empid);
	System.out.println("in servlet empid34="+empid+"");
	request.setAttribute("DE", rs);
	rd=request.getRequestDispatcher("DeleteEmp.jsp");
	rd.forward(request, response);
}

else if(pagename.equals("Deleted"))
{
	Delete dt=new Delete();
	String empid=request.getParameter("emp");
	dt.del(empid);
	rd=request.getRequestDispatcher("Success2.html");
	rd.forward(request, response);

}
```

```
//Project
else if(pagename.equals("prc"))
{
        System.out.println("DONE");
        creProject cp=new creProject();
        ResultSet rst=cp.proEmpId();
        request.setAttribute("cpr", rst);
        rd= request.getRequestDispatcher("createProject.jsp");
        rd.forward(request, response);
}

else if(pagename.equals("prjct"))
{
        creProject crp=new creProject();
        int empi=Integer.parseInt(request.getParameter("emp"));
        int pid=Integer.parseInt(request.getParameter("pid"));
        String pname=request.getParameter("pname");
        String status=request.getParameter("status");
        crp.crprct(empi, pid, pname,status);
        System.out.println("in servlet empid="+empi+"");
        rd=request.getRequestDispatcher("Success.html");
        rd.forward(request, response);

}

//Update Project
else if(pagename.equals("updateProject"))
{
        System.out.println("DONE");
        updatProject upp = new updatProject();
        int pid=Integer.parseInt(request.getParameter("pid"));
        ResultSet rs=upp.PSearch(pid);
        System.out.println("in servlet pid34="+pid+"");
        request.setAttribute("uup", rs);
        rd=request.getRequestDispatcher("updateProject1.jsp");
        rd.forward(request, response);
}
```

```
else if(pagename.equals("updateproj"))
{
        updatProject upp=new updatProject();
        int empid=Integer.parseInt(request.getParameter("emp"));
        int pid=Integer.parseInt(request.getParameter("pid"));
        String pname=request.getParameter("pname");
        String status=request.getParameter("status");
        upp.updateProject(pid, empid, pname, status);
        System.out.println("in servlet empid="+pid+"");
        rd=request.getRequestDispatcher("Success1.html");
        rd.forward(request, response);
}
```

//Delete Project

```
else if(pagename.equals("DelProject"))
{
        System.out.println("DONE");
        deleteProject dp=new deleteProject();
        int pid=Integer.parseInt(request.getParameter("pid"));
        ResultSet rs=dp.PSearch(pid);
        System.out.println("in servlet pid34="+pid+"");
        request.setAttribute("dp", rs);
        rd=request.getRequestDispatcher("deleteProject1.jsp");
        rd.forward(request, response);
}

else if(pagename.equals("DeleteProject"))
{
        deleteProject dpt=new deleteProject();
        int pid=Integer.parseInt(request.getParameter("pid"));
        dpt.del(pid);
        System.out.println("in servlet empid="+pid+"");
        rd=request.getRequestDispatcher("Success2.html");
        rd.forward(request, response);

}
```

```
//PaySlip
else if(pagename.equals("Payslip"))
{
        payslip pay=new payslip();
        int empid=Integer.parseInt(request.getParameter("emp"));
        ResultSet rs=pay.PaySearch(empid);
        System.out.println("in servlet empid34="+empid+"");
        request.setAttribute("pay", rs);
        rd=request.getRequestDispatcher("Payslip1.jsp");
        rd.forward(request, response);
}

else if(pagename.equals("PayslipG"))
{
        System.out.println("DONE");
        payslip pay1=new payslip();
        int empid=Integer.parseInt(request.getParameter("emp"));
        String fname=request.getParameter("fname");
        String lname=request.getParameter("lname");
        int Salary=Integer.parseInt(request.getParameter("salary"));
        String Status=request.getParameter("status");
        pay1.PayEmploy(empid, fname, lname, Salary, Status);
        rd= request.getRequestDispatcher("Success4.html");
        rd.forward(request, response);
}

else if(pagename.equals("CheckPayslip"))
{
        payslip pay=new payslip();
        int empid=Integer.parseInt(request.getParameter("emp"));
        ResultSet rs=pay.CheckSearch(empid);
        System.out.println("in servlet empid34="+empid+"");
        request.setAttribute("checkpay", rs);
        rd=request.getRequestDispatcher("CheckPayslip1.jsp");
        rd.forward(request, response);
}
}
```

```
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
		// TODO Auto-generated method stub
		String pagename = request.getParameter("page");
		RequestDispatcher rd;
		System.out.println(pagename);
		String password,type;
		user=request.getParameter("user");
		password=request.getParameter("pwd");
		type=request.getParameter("admin");
		if(pagename.equals("login"))
		{
			LoginDB db=new LoginDB();
			if(type.equals("admin"))
			{
				if(db.vloginDB(user,password,type))
				{
					rd= request.getRequestDispatcher("Admin.jsp");
					rd.forward(request, response);
				}
				else
				{
					rd= request.getRequestDispatcher("Error.jsp");
					rd.forward(request, response);
				}
			}
			else
			{
				if(db.vloginDB(user,password,type))
				{
					rd= request.getRequestDispatcher("employ.jsp");
					rd.forward(request, response);
				}
				else
				{
					rd= request.getRequestDispatcher("Error.jsp");
					rd.forward(request, response);
				 }
			}
		}
	}

}
```

Chapter 5: System Testing

Chapter – 5

System Testing

5.1 Introduction:

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic.

As a rule, system testing takes, as its input, all of the "integrated" software components that have passed integration testing and also the software system itself integrated with any applicable hardware system(s). The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together (called *assemblages*) or between any of the *assemblages* and the hardware. System testing is a more limited type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole.

Types of system testing :

5.1.1 Unit testing

Unit testing is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures are tested to determine if they are fit for use. Intuitively, one can view a unit as the smallest testable part of an application.

5.1.2 Integration testing

Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.

5.1.3 Automation testing

In software testing, test automation is the use of special software (separate from the software being tested) to control the execution of tests and the comparison of actual outcomes with predicted outcomes. Test automation can automate some repetitive but necessary tasks in a formalized testing process already in place, or add additional testing that would be difficult to perform manually.

5.1.4 Regression testing

Regression testing is a type of software testing that seeks to uncover new software bugs, or *regressions*, in existing functional and non-functional areas of a system after changes such as enhancements, patches or configuration changes, have been made to them.

The intent of regression testing is to ensure that changes such as those mentioned above have not introduced new faults. One of the main reasons for regression testing is to determine whether a change in one part of the software affects other parts of the software.

Test case generation

To generate a set of test cases, test designers attempt to cover all the functionality of the system and fully exercise the GUI itself.

Test cases

Test case 1

Chapter 6: Future scope and Conclusion

6.1 Future scope

One of the ugliest things about the airline pilot profession is the fact that we usually are handcuffed to our ultimate employer for our entire career, and if our employer goes out of business, we're forced to start over as if you're a brand new pilot fresh out of flight school.

There are many good things that come with being an airline pilot....

Pilots Love Their Job

First of all; most pilots are very passionate about their profession and love flying airplanes. Of everyone I know in all kinds of different professions, no one enjoys their job as much as I do. And likely, if you do meet your career aspirations of becoming an airline pilot, you will realize the same thing. There are few things better in life than going to work and actually enjoying what you do. The job is always different, it's stimulating, interesting, and can be extremely rewarding.

Schedule

The schedule flexibility, especially the flexibility afforded to senior pilots, can be extremely beneficial Airline pilots don't work the typical 9 to 5 schedule that many other professionals work. Very often, pilots have groups of days where they are "ON" and have groups of days off where they are "OFF." The quantity and the quality of these ON/OFF days is usually determined by one thing- seniority. A moderately senior pilot can have sometimes 18 days off, with these days off grouped together in a manner that would allow weekends and holidays off, or perhaps long stretches of time off by grouping "OFF" days together. Of course, you have to be senior enough to take advantage of these scheduling abilities.

Travel

If you love to travel, then this job is for you. Not only will you have the opportunity to "see the world" on your company's dime as you "work for the man" as an airline pilot, but you also will enjoy travel benefits, like inexpensive space available seating to wherever your airline flies, or discounted airline tickets for you, your family, and your parents. Now I'll be the first to tell you that the travel benefits aren't as good as they used to be just 10 years ago, but for the most part they are usable if you travel smart. And if you're traveling alone as a pilot, you'll have access

to the jump seat in the cockpit of both your airline and other airlines, usually for free. With this benefit, you can travel virtually anywhere in the world on your own.

People

Just as pilots usually love their jobs, you'll find that the other professionals you work with enjoy theirs, too. You'll meet many different people, cultures, and their associated ideas. There are few things more enjoyable than flying with a group of people who love their jobs and the airline biz.

Live Wherever You Want

Since pilots can fly very inexpensively on their own airline, or use the pilot-exclusive cockpit jump seats on their own carrier or just about any other carrier for free, many pilots choose to live outside of the city they are based in with their airline.

6.2 Conclusion

In this project there are 3 main modules (admin, student and trainer). The entire modules are connected to respective tables and when admin use any module then all the functions are performed directly on the tables. For making project more accurate some test case are done and all the results from those test cases are included in documentation. At last we fulfill our goal by making software which is less prone to errors and tired to make it user friendly to the extent we could.

Bibliography

Books:

1. Software engineering (Third edition K.K.Aggarwal and Yogesh Singh 3rd edition)
2. Software Testing(Paul C. Jorgensen 6th edition)
3. DBMS(Ramez Elmasri and Shamkant B.Navathe 5th edition)
4. Web technology
5. Java 2EE

Search Engine:

1. Google
2. Yahoo

Sites we referred:

1. www.wikipedia.org
2. www.w3schools.com/html
3. www.roseindia.net/jsp/implement-javascript-with-jsp.shtml
4. http://www.sloanehelicopters.com/helicopter_flying_school/private_pilot_licence