**Software Requirements Specification**

**Version 2**

**CareAxiom HRMS**

**Client: CareAxiom**

**Table of Contents**

[**Introduction**](#_ibfuw0titv5s) **5**

[Product](#_4mxo2drhb0ya) 5

[Scope](#_m6nvxarwuj0y) 5

[Business Goals](#_kngzxbwoeu7j) 5

[Document Conventions](#_xp0dzovqt25x) 5

[References](#_azm9s9drdryn) 5

[**Overall Description**](#_rsz1s9fw0zuf) **6**

[Product Features](#_i6yn28uu7e9i) 6

[User Classes and Characteristics](#_5dhl2bah1ni6) 6

[Operating Environment](#_2ngr5ht8iqqj) 6

[Design and Implementation Constraints](#_99t81uhnkhc4) 6

[Assumptions and Dependencies](#_99s1imt7mcjq) 7

[**Functional Requirements**](#_zc6f4zsumgvu) **8**

[Add User](#_isbhi6afapuj) 9

[Edit Profile](#_i4mi3vbahal4) 11

[Delete user](#_1k9w3htkjpbv) 12

[View Profile](#_lyiuybyx2c8j) 12

[Apply for leave](#_lsmlycoa25lr) 13

[Apply for work from home](#_3vmd18o035hk) 14

[Approve or reject leave/work from home application](#_bbuf549axij7) 15

[View Calendar](#_z3izl88whkoa) 16

[View Directory](#_fek9l55coa2d) 17

[View Modules](#_cy9gq7gemx01) 18

[Add Module](#_pvy8bbnij6pf) 19

[Edit Module](#_jpykuh6zvtgm) 20

[View Organizational Structure](#_ons7ten8hmpb) 21

[View Employee’s Remaining Leaves](#_e9yz4uv1t2jj) 22

[View Employee’s Current Salary](#_qa4j3pb6oqum) 23

[View Employee’s Salary History](#_jq94e73wnb9c) 24

[Edit Employee’s Salary](#_fmtq25m551fq) 26

[Performance Requirements](#_g2ng49va9brr) 27

[Security Requirements](#_8rz1o5xttwhm) 27

[Other Requirements](#_mbmtpnaspide) 27

[**Appendix A: Glossary**](#_a1qdn1asmyq0) **28**

[**Appendix B: Analysis Models**](#_xlv4dikfnacv) **29**

[**Appendix D: Screenshots**](#_3vozr6jtobah) **39**

[**Appendix C: Design Models**](#_hpjt1c24gfo2) **43**

[**Appendix E: Test Cases**](#_vkdfy03j87mp) **44**

[**Appendix F: IV & V Report**](#_vmibp4bjndui) **45**

[**Appendix G: Risk Report**](#_fliqemhtb2i2) **47**

[**Appendix H: Activity Timesheet**](#_h912fmhmpq3q) **47**

[**Project Manager**](#_32a59gne28cm) **49**

# 

# 

# **1.0 Introduction**

## **1.1 Product**

CareAxiom HRMS is a Human Resource Management System specified to the needs of CareAxiom – a SaaS (Software as a Service) based healthcare communication platform. The HRMS includes complete user profiles, salary and leaves history, an integrated calendar and a dynamic organizational structure. It gives users different rights and provides them with different views of the web-based software to cater for their needs.

## **1.2 Scope**

CareaxiomHRMS is a basic level management system. Our main focus shall be on very core functionalities an HRMS can provide. They include maintaining a profile. A module to enable employee to apply for leaves and work from home, and maintain a history of projects. It shall also provide some views for the company to display its organizational structure.

## **1.3 Business Goals**

This project aims to assist the company by:

● Automating their attendance system

● Giving compact view of their organizational structure

● Allowing users to manage their resources through a single interface

All in all, our product will help their management team get rid of excel sheets and trello card.

## **1.4 Document Conventions**

Some conventions followed by the system are:

● Buttons/Tabs names will be written in single quotes

● Major functionalities will be written in bold

● Specified terms by company will be written in italics, for example *public views, engineering owner*

Other than these, we shall refer to team member as *engineer* and team lead as *engineering owner*.

Employee is a general term that can be used for *admin, engineering owner* or *engineer.*

## 

# **2.0 Overall Description**

## **2.1 Product Features**

Our product shall provide minimal core level functionalities. First one shall be **Automated Attendance System**. This feature shall allow the engineer to request for leave, accept leave, reject leave and keep track of remaining and currently availed leaves.

Another feature shall be **Resource Management**. This shall deal with the engineer’s data. It shall be personal such as profile. It shall also include data that deals with the career, like salary.

Some *public views* shall be available to outside viewers including **Organizational View** and **Company Portfolio**. Former one shall give a hierarchical structure of company’s engineers and lateral one shall display a collection of the company’s modules/projects.

In addition to above features, engineers shall be able to edit their personal data, specify the engineering owner they are reporting to. Admin shall be able to increment salary and allot bonus.

Our HRMS shall also maintain a history of salary and projects for engineers.

## **2.2 User Classes and Characteristics**

We have simply one user category that is *engineer*. This *engineer* can then have two rights, i.e. it can have *admin* rights or *engineering owner* rights.

*Admins* have the most privileges. They can manage salary, leaves. They can add or edit a module.

*Engineering Owner*, owns module to which it has a team reporting him. He also has the ability to accept or reject a leave requested by an *engineer* under his team.

## **2.3 Operating Environment**

Our product is a web based solution. It does not require any specific machine to run. However, for the first release, we shall focus on desktop based view. This means that it will not be mobile friendly.

For development purposes, we shall make use of node package manager (npm) to install and keep track of node modules. Bower will also assist us in handling front end packages. Our server will be deployed on Linux.

## **2.4 Design and Implementation Constraints**

Our color theme will be based upon the company’s logo, i.e. will use the same colors that company makes use of in their brand selling.

Our design shall be inspired by certain HRMS applications already available in market. These include BambooHR and WorkPlaceHR.

We’ll be using Node.js for implementation because of requirement from the client.

## **2.5 Assumptions and Dependencies**

We shall be incorporating Google Calendar API. For that reason we assume that its full support for incorporating is available in the technology we are using. We are also dependent upon functionality and flexibility of its API.

# 

# 

# **3.0 Functional Requirements**

Functional requirements are as follows:

● Login

● Add User

● Edit Profile

● Delete Profile

● View Profile

● Apply for leave/work from home

● Approve leave/work from home application

● View calendar

● View directory

● View modules

● Add module

● Edit module

● View organizational structure

● View salary history

● Edit salary

● View leaves

**Login**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-1 | |
| **Purpose** | | User can login to system | |
| **Priority** | | High | |
| **Actors** | | Engineer, Admin, Engineering Owner | |
| **Pre-conditions** | | User account must exist | |
| **Post-conditions** | | User is redirected to Homepage | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Enter username and password | |  |
| **2** | Clicks ‘Login’ button | |  |
| **3** |  | | Verifies credentials |
| **4** |  | | Redirects to *profile page* |
| **Alternate Course of Action (Invalid Username)** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **3** |  | | Display error message ‘Invalid username’ |
| **Go to step 1** | | | |
| **Alternate Course of Action (Invalid Password)** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **3** |  | | Display error message ‘Invalid Password’ |
| **Go to step 1** | | | |
|  |  |  |  |

Table 1: UC-1

## **Add User**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identifier** | | | UC-2 | |
| **Purpose** | | | Admin can add new user | |
| **Priority** | | | High | |
| **Actors** | | | Admin | |
| **Pre-conditions** | | | Admin must be logged in | |
| **Post-conditions** | | | Admin successfully added new user | |
| **Typical Course of Action** | | | | |
| **S#** | | **Actor Action** | | **System Response** |
| **1** | | Clicks on ‘Register’ tab | |  |
| **2** | |  | | Displays Registration page |
| **3** | | Enters name, address, contact number, designation, ppPath join\_date, increament\_per\_year, email | |  |
| **4** | | Enters current project and project lead | |  |
| **5** | | Assigns rights i.e. admin or engineering owner | |  |
| **6** | | Clicks ‘Add User’ button | |  |
| **7** | |  | | Checks the fields |
| **8** | |  | | System adds new user data to database |
| **9** | |  | | Displays feedback message |
| **Alternate Course of Action (Duplicate Email)** | | | | |
| **S#** | | **Actor Action** | | **System Response** |
| **8** | |  | | If the fields are invalid, displays error message, “Please check the entered fields”. |
| **Goto step 3** | | | | |
| **Alternate Course of Action (Skip rights)** | | | | |
| **S#** | **Actor Action** | | | **System Response** |
| **5** | Does not assign any rights | | |  |
|  |  |  |  |  |

Table 2: UC-2

## **Edit Profile**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-3 | |
| **Purpose** | | User can edit his profile | |
| **Priority** | | High | |
| **Actors** | | Engineer, Engineering Owner, Admin | |
| **Pre-conditions** | | User is logged in, User is viewing profile | |
| **Post-conditions** | | System displays the updated information | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Clicks ‘Edit’ button | |  |
| **2** |  | | Enables fields to edit |
| **3** | Enters name, address, contact number,designation, ppPath join\_date,increament\_per\_year,  email | |  |
| **4** | Enters current project and project lead | |  |
| **5** | Assigns rights i.e. admin or engineering owner | |  |
| **5** | Click on ‘Save’ button | |  |
| **6** |  | | Checks the fields |
| **7** |  | | Saves the information. |
|  |  |  |  |

Table 3: UC-3

## **Delete user**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-4 | |
| **Purpose** | | Admin can delete the user | |
| **Priority** | | High | |
| **Actors** | | Admin | |
| **Pre-conditions** | | Admin must be logged in. User’s profile already exists. | |
| **Post-conditions** | | Admin successfully deleted the user | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Click on ‘Delete user’ tab. | |  |
| **2** |  | | Redirect to ‘profiles page’. |
| **3** | Search the user by email. | |  |
| **4** |  | | Display the Result. |
| **5** | Clicks on ‘Delete’ button | |  |
| **6** |  | | Delete the user from system. |
| **7** |  | | Display message “Deleted Successfully” |
|  |  |  |  |

Table 4: UC-4

## **View Profile**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-5 | |
| **Purpose** | | User can view his profile. | |
| **Priority** | | High. | |
| **Actors** | | Admin, Engineer, Engineering owner. | |
| **Pre-conditions** | | User is logged in. | |
| **Post-conditions** | | Profile shown to user. | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Clicks on ‘My Profile’ tab | |  |
| **2** |  | | Display user Profile |
|  |  |  |  |

Table 5: UC-5

## **Apply for leave**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-6 | |
| **Purpose** | | User has to submit an application before he or she can take a leave from work | |
| **Priority** | | High | |
| **Actors** | | Admin, Engineering Owner, Engineer | |
| **Pre-conditions** | | User is logged in | |
| **Post-conditions** | | Application is submitted | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Clicks on ‘Calendar’ tab | |  |
| **2** |  | | Redirects to calendar page |
| **3** | Clicks on leave button | |  |
| **4** |  | | Shows a popup of application form |
| **5** | Mentions start date | |  |
| **6** | Mentions end date | |  |
| **7** | Mentions reason in description | |  |
| **8** | Clicks on ‘Submit’ button | |  |
| **9** |  | | Checks fields |
| **10** |  | | Records the application |
| **11** |  | | Display message “Application submitted” |
| **12** |  | | Closes popup form |

Table 6: UC-6

## **Apply for work from home**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-7 | |
| **Purpose** | | User has to submit an application before he or she can work from home | |
| **Priority** | | High | |
| **Actors** | | Admin, Engineering Owner, Engineer | |
| **Pre-conditions** | | User is logged in | |
| **Post-conditions** | | Application is submitted | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Clicks on calendar tab | |  |
| **2** |  | | Redirects to calendar page |
| **3** | Clicks on ‘Work From Home’ button | |  |
| **4** |  | | Shows a popup of application form |
| **5** | Mentions start date | |  |
| **6** | Mentions end date | |  |
| **7** | Mentions reason in description | |  |
| **8** | Clicks on ‘Submit’ button | |  |
| **9** |  | | Checks fields |
| **10** |  | | Records the application |
| **11** |  | | Display message “Application submitted” |
| **12** |  | | Closes popup form |

Table 7: UC-7

## **Approve or reject leave/work from home application**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-8 | |
| **Purpose** | | Applications for leave/work from home need to be approved. | |
| **Priority** | | High | |
| **Actors** | | Admin, Engineering Owner | |
| **Pre-conditions** | | User is logged in, User is viewing calendar | |
| **Post-conditions** | | Status of application is changed to approved or rejected | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **3** | Clicks on ‘Pending Applications’ button | |  |
| **4** |  | | Redirects to pending applications page |
| **5** | Clicks on an application | |  |
| **6** |  | | Application pops up |
| **7** | Clicks on ‘Approve’ | |  |
| **8** |  | | Records the approval |
| **9** |  | | Sends application status to applicant |
| **10** |  | | Closes popup |
| **Alternate Course of Action (No pending applications)** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **4** |  | | No pending applications |
| **5** |  | | Terminate |
| **Alternate Course of Action (Reject application)** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **7** | Clicks on ‘Reject’ | |  |
| **8** |  | | Records the rejection |
|  |  |  |  |

Table 8: UC-8

## **View Calendar**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-9 | |
| **Purpose** | | Users can view calendar in order to see leaves history | |
| **Priority** | | High | |
| **Actors** | | Admin, Engineering Owner, Engineer | |
| **Pre-conditions** | | User is logged in | |
| **Post-conditions** | | Shows calendar | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Clicks on calendar tab | |  |
| **2** |  | | Redirects to calendar page |
|  |  |  |  |

Table 9: UC-9

## **View Directory**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-10 | |
| **Purpose** | | Show the directory structure of the organization. | |
| **Priority** | | High | |
| **Actors** | | Admin, Engineering Owner, Engineer | |
| **Pre-conditions** | | User is logged in | |
| **Post-conditions** | | Show the directory view | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Request to show directory of the organization. | |  |
| **2** |  | | Checks the information of employees |
| **3** |  | | Shows the directory |
| **Alternate Course of Action (Empty Directory)** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **3** |  | | If no employee found, displays message “No employee found”. |
|  |  |  |  |

**Table 10: UC-10**

## **View Modules**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-11 | |
| **Purpose** | | Show modules | |
| **Priority** | | High | |
| **Actors** | | Admin, Engineering Owner, Engineer | |
| **Pre-conditions** | | User is logged in. | |
| **Post-conditions** | | Shows the modules. | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Request to show the modules. | |  |
| **2** |  | | Checks all the modules of organization. |
| **3** |  | | Shows the modules of the organization. |
| **Alternate Course of Action (No Modules)** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **2** |  | | If no module exists, show message “No module exists”. |
|  |  |  |  |

**Table 11: UC-11**

## **Add Module**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-12 | |
| **Purpose** | | Add a new module in the system. | |
| **Priority** | | High. | |
| **Actors** | | Admin, Engineering Owner | |
| **Pre-conditions** | | User is logged in. | |
| **Post-conditions** | | New module added | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Requests for the “add module” page. | |  |
| **2** |  | | Open the page. |
| **3** | Enters module title | |  |
| **4** | Enters module description | |  |
| **5** |  | | Checks the entered fields. |
| **6** |  | | Add the module to the system. |
| **7** |  | | Shows the message “Module Added”. |
| **Alternate Course of Action (Missing Fields)** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **5** |  | | If any field is empty, system shows “Please enter all fields”. |
| Goto 3 | | | |
|  |  |  |  |

**Table 12: UC-12**

## **Edit Module**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-13 | |
| **Purpose** | | Edit a module. | |
| **Priority** | | Medium. | |
| **Actors** | | Admin, Engineering Owner | |
| **Pre-conditions** | | User is logged in. Module already exists. | |
| **Post-conditions** | | Module edited. | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Request a module to edit. | |  |
| **2** |  | | Open the requested module in editing mode. |
| **3** | Edit the title and description | |  |
| **4** |  | | Checks the updated entered fields. |
| **5** |  | | Update the module in the system. |
| **6** |  | | Show the message “Module Updated”. |
| **Alternate Course of Action (Invalid Fields)** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **5** |  | | If any field is invalid, prompts to re-enter the fields. |
| **6** | Enter the fields again. | |  |
|  | Go to 4 | | |
|  |  |  |  |

**Table 13: UC-13**

## **View Organizational Structure**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-14 | |
| **Purpose** | | Show the organizational hierarchy. | |
| **Priority** | | Low. | |
| **Actors** | | Admin, Engineering Owner, Engineer. | |
| **Pre-conditions** | |  | |
| **Post-conditions** | | Shows the organizational structure. | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Requests to show the organizational structure. | |  |
| **2** |  | | Compiles the organizational structure. |
| **3** |  | | Shows the organizational structure. |
| **Alternate Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
|  |  |  |  |

**Table 14: UC-14**

## **View Employee’s Remaining Leaves**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-15 | |
| **Purpose** | | Allowing employee to keep track of leaves | |
| **Priority** | | High | |
| **Actors** | | Admin, Engineering Owner, Engineer | |
| **Pre-conditions** | | User is logged in | |
| **Post-conditions** | | ... | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Clicks on ‘Employees’ | |  |
| **2** |  | | Redirects to ‘Employees’ page |
| **3** | Clicks on ‘Leaves’ tab | |  |
| **4** |  | | Displays rows populated with user data, including leaves |
| **5** | Uses filter option to search for specific employee | |  |
| **6** |  | | Filters rows |
| **7** | Sees the remaining leaves of desired user | |  |
| **Alternate Course of Action (No Records)** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **6** |  | | User data not present |
| **7** |  | | Terminate |
|  |  |  |  |

**Table 15: UC-15**

## **View Employee’s Current Salary**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-16 | |
| **Purpose** | | Allows employee to view current salary | |
| **Priority** | | High | |
| **Actors** | | Admin, Engineering Owner, Engineer | |
| **Pre-conditions** | | User is logged in | |
| **Post-conditions** | | Shows current salary | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Clicks on ‘Employees’ | |  |
| **2** |  | | Redirects to ‘Employees’ page |
| **3** | Clicks on ‘Salary’ tab | |  |
| **4** |  | | Displays rows populated with user data, including current salary |
| **5** | Uses filter option to search for specific employee | |  |
| **6** |  | | Filters rows |
| **7** | Sees current salary of desired user | |  |
| **Alternate Course of Action (No Records)** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **6** |  | | User data not present |
| **7** |  | | Terminate |
|  |  |  |  |

**Table 16: UC-16**

## **View Employee’s Salary History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-17 | |
| **Purpose** | | Allows employee to keep track of salary | |
| **Priority** | | High | |
| **Actors** | | Admin, Engineering Owner, Engineer | |
| **Pre-conditions** | | User is logged in | |
| **Post-conditions** | | Shows salary history of employee | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Clicks on ‘Employees’ | |  |
| **2** |  | | Redirects to ‘Employees’ page |
| **3** | Clicks on ‘Salary’ tab | |  |
| **4** |  | | Displays rows populated with user data, including current salary |
| **5** | Uses filter option to search for specific employee | |  |
| **6** |  | | Filters rows |
| **7** | Sees current salary of desired user | |  |
| **8** | Clicks on ‘History’ button | |  |
| **9** |  | | Shows salary history of employee. |
| **10** | View salary history | |  |
| **Alternate Course of Action (No Records)** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **6** |  | | User data not present |
| **7** |  | | Terminate |
| **Alternate Course of Action (No History)** | | | |
| **9** |  | | No history present |
| **10** |  | | Terminate |
|  |  |  |  |

**Table 17: UC-17**

## **Edit Employee’s Salary**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-18 | |
| **Purpose** | | Allows admin to add increments or bonuses to employee’s salary | |
| **Priority** | | High | |
| **Actors** | | Admin | |
| **Pre-conditions** | | User is logged in | |
| **Post-conditions** | | Salary is updated | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Clicks on ‘Employees’ | |  |
| **2** |  | | Redirects to ‘Employees’ page |
| **3** | Clicks on ‘Salary’ tab | |  |
| **4** |  | | Displays rows populated with user data, including current salary |
| **5** | Uses filter option to search for specific employee | |  |
| **6** |  | | Filters rows |
| **7** | Sees salary info of desired user | |  |
| **8** | Clicks on ‘Edit’ button | |  |
| **9** |  | | Displays a pop up |
| **10** | Enters month,year,bonus,bump,salary,  Status | |  |
| **11** | Clicks ‘Save’ button | |  |
| **12** |  | | Verifies values and make changes accordingly |
| **Alternate Course of Action 1 (No Records)** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **6** |  | | User data not present |
| **7** |  | | Terminate |
| **Alternate Course of Action 2 (Incorrect Details)** | | | |
| **12** |  | | Prompts user to retype incorrectly provided values |

**Table 18: UC-18**

**4.0 Nonfunctional Requirements**

## **4.1 Performance Requirements**

4.1.1 System shall makes use of Asynchronous paradigm.

4.1.2 System shall makes AJAX request to import employee salary history.

4.1.3 System shall display employee records within 3 seconds, provided a stable internet connection.

## **4.2 Security Requirements**

4.2.1 System shall redirect an unauthorized user to login page.

## **4.3 Other Requirements**

# **Appendix A: Glossary**

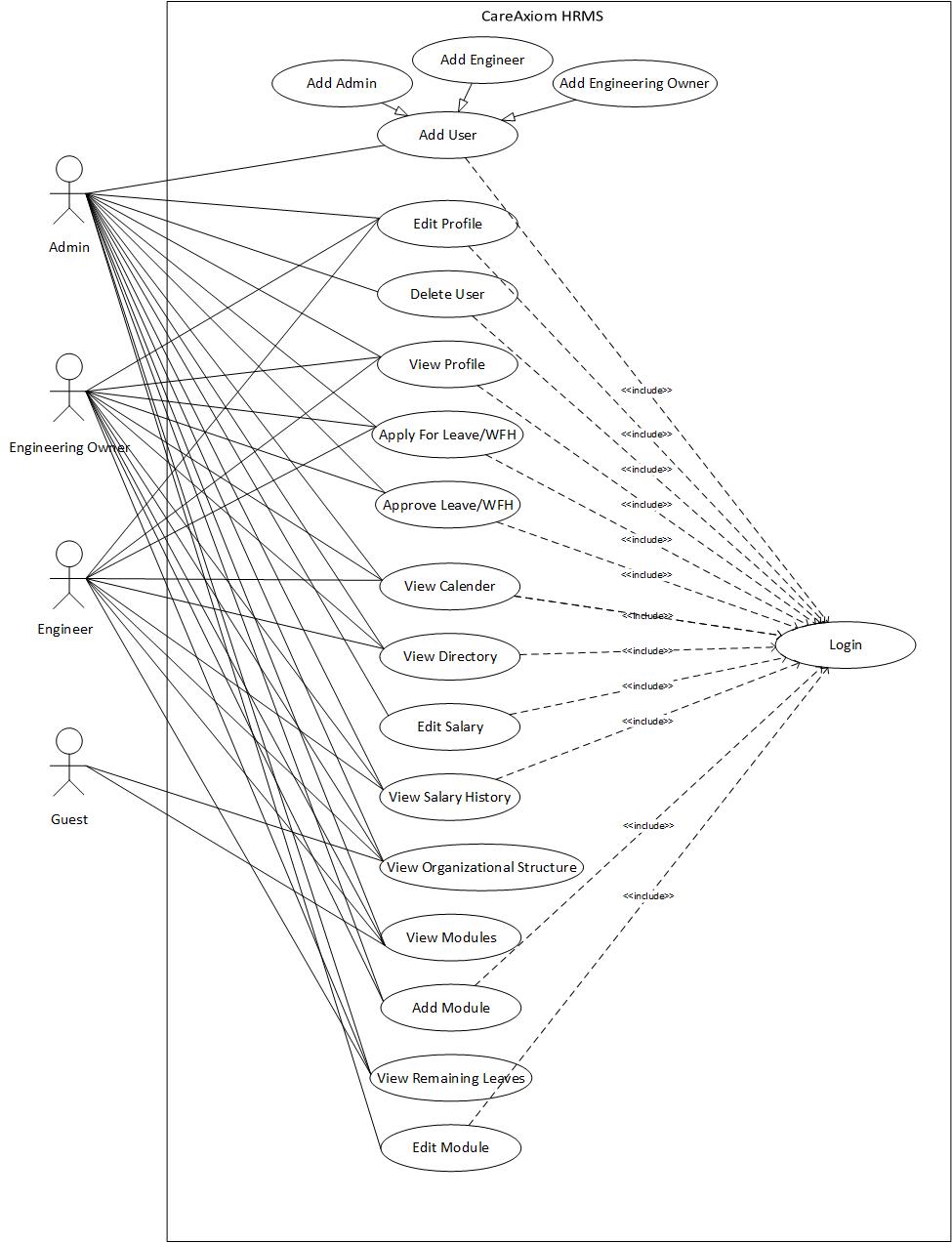
|  |  |
| --- | --- |
| **Term** | **Definition** |
| Admin | Person with complete rights responsible for managing the system. |
| Engineering Owner | Person who owns one or more module, each having a team reporting to Engineering Owner. |
| SaaS | Software as a Service is a software licensing and delivery model in which software is licensed on a subscription basis and is centrally hosted. |
| User | Anyone who makes use of the system. In this case, employees are the users. |
| Modules | A module is a self sustainable component of a complete product. |

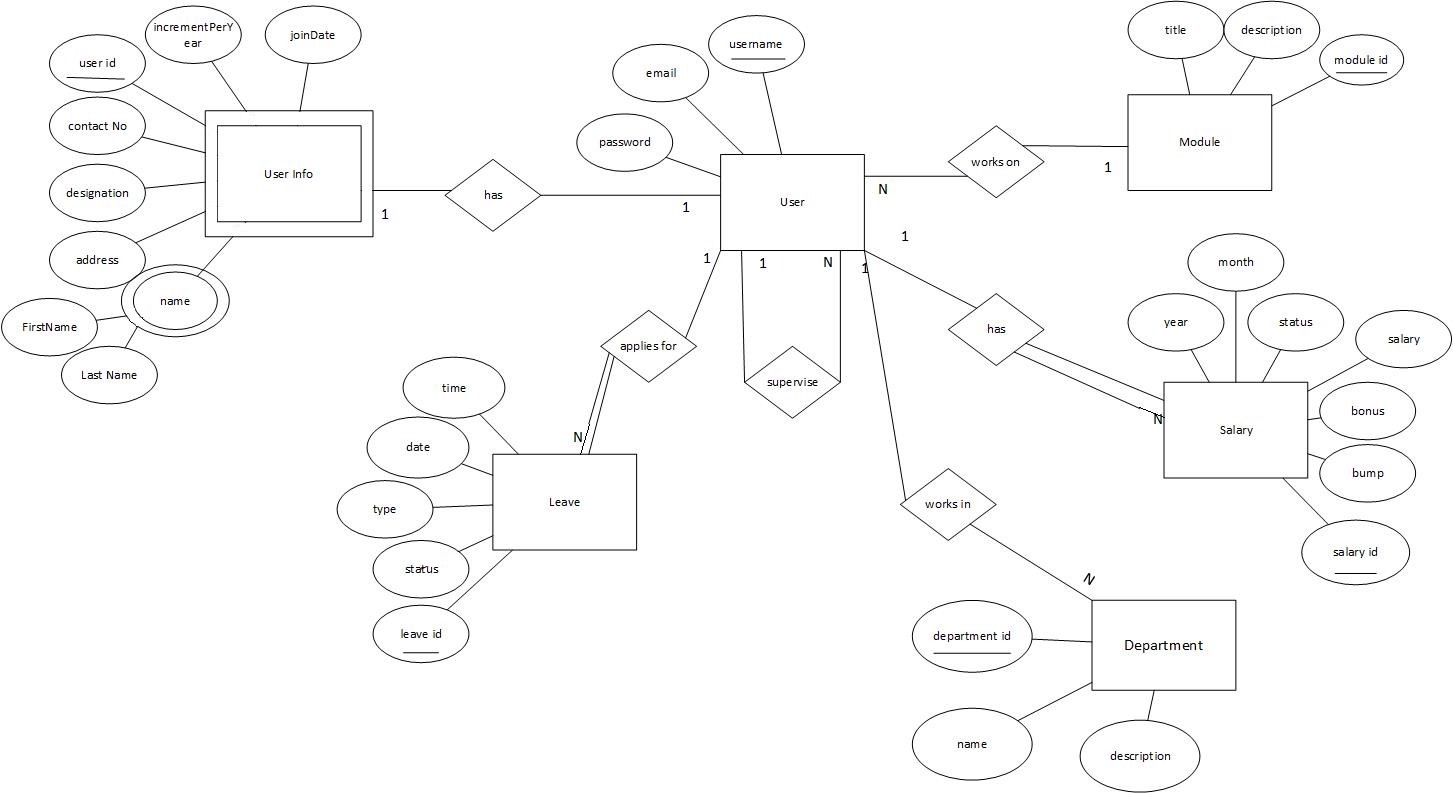
# 

# 

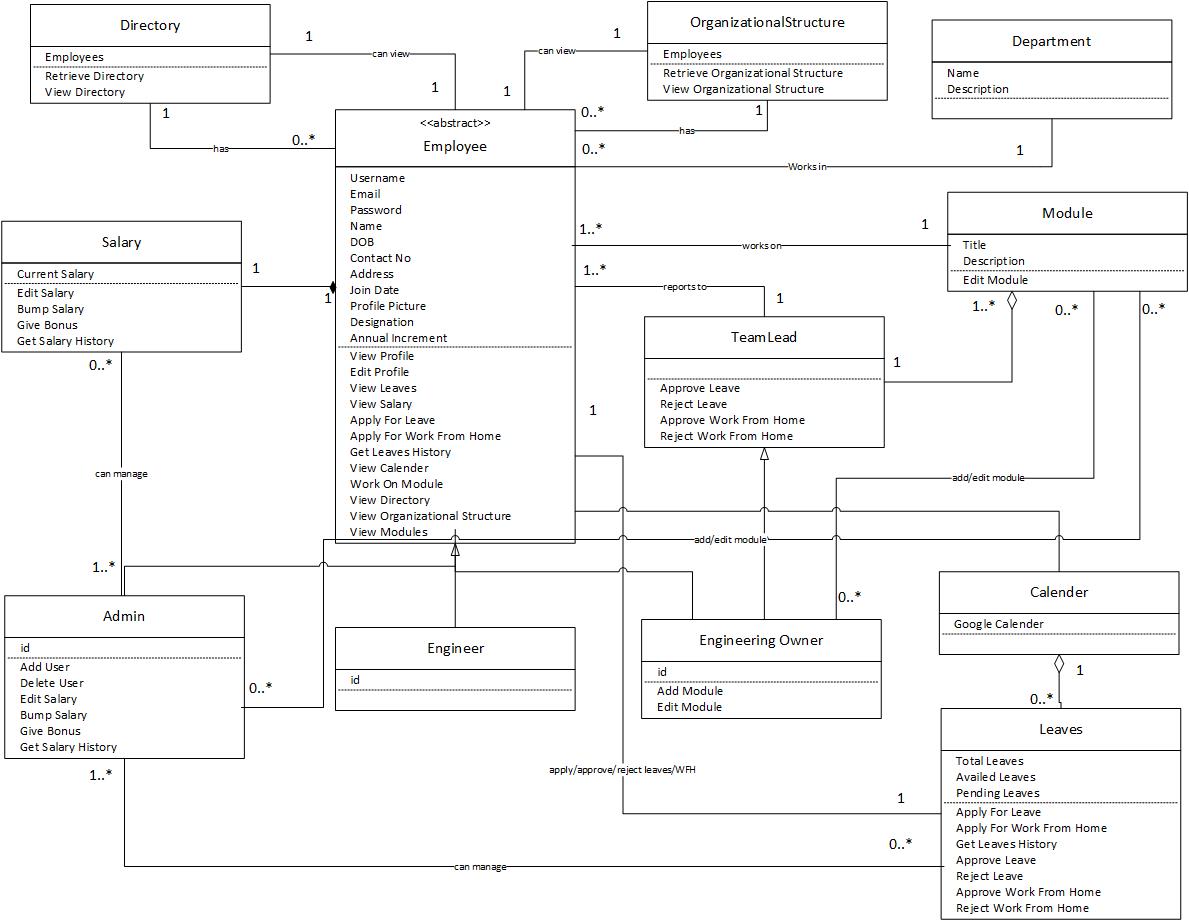
# **Appendix B: Analysis Models**

**Use-Case Diagram:**

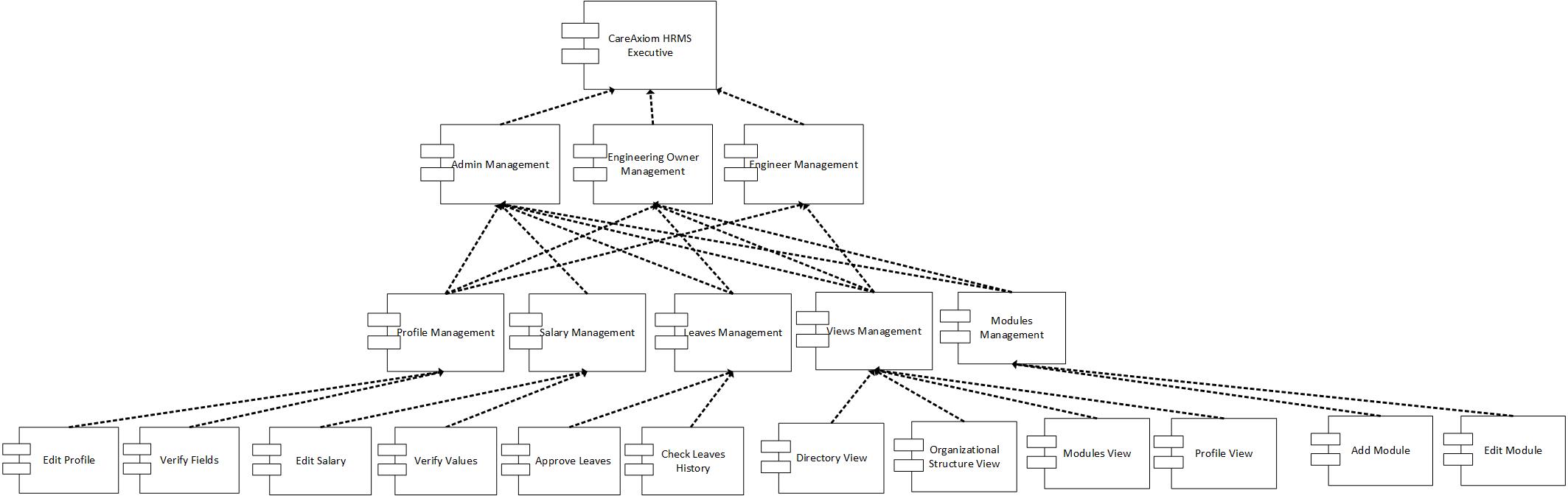


**Entity-relationship Diagram:**

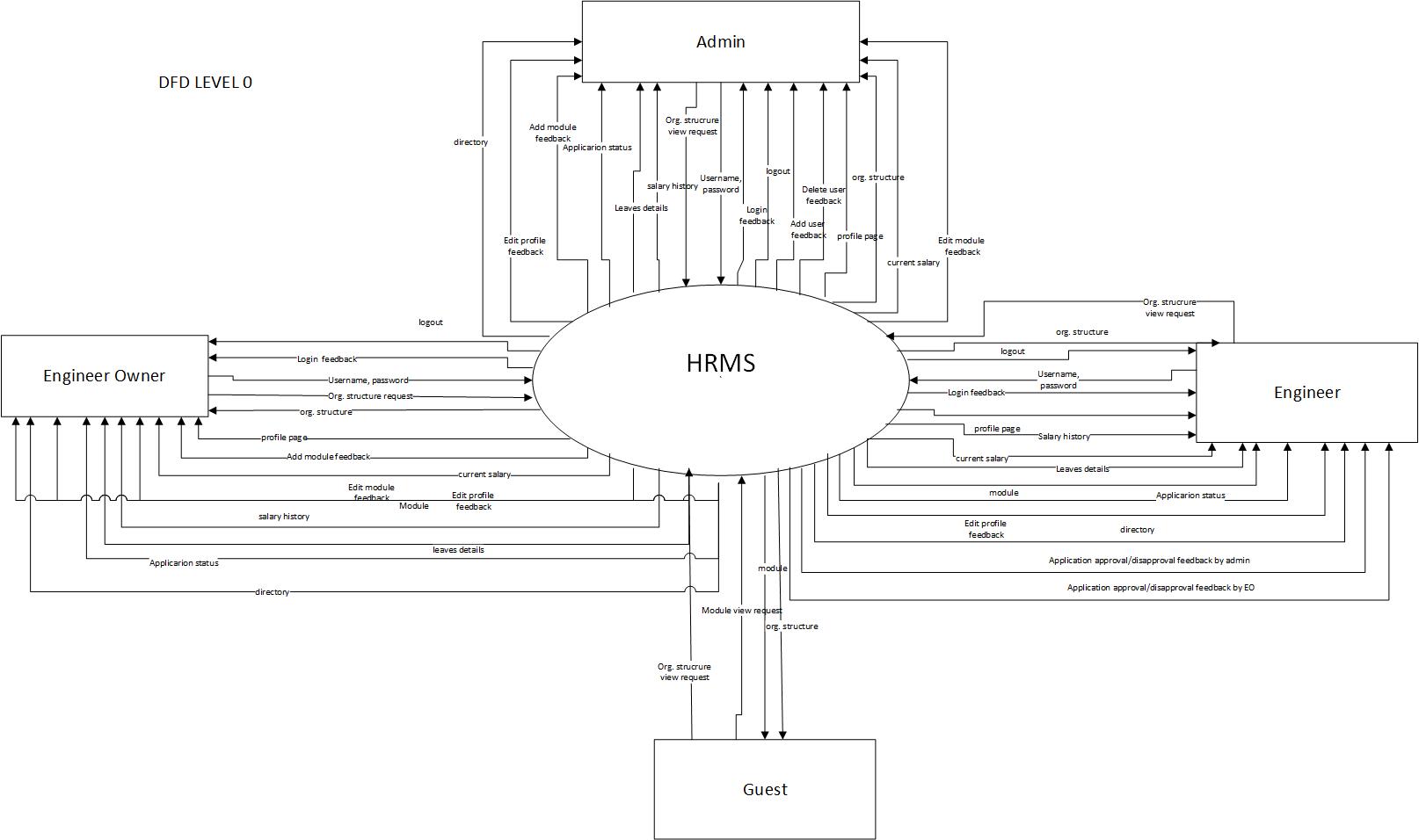
**Analysis Class Diagram:**

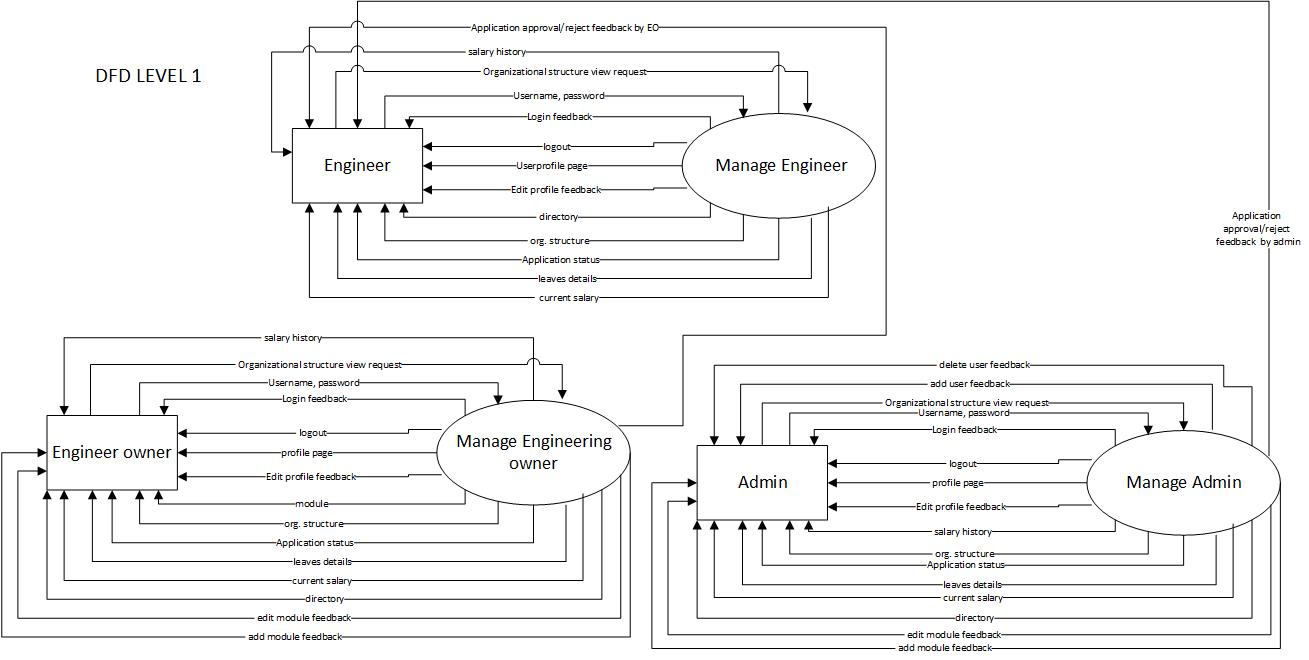


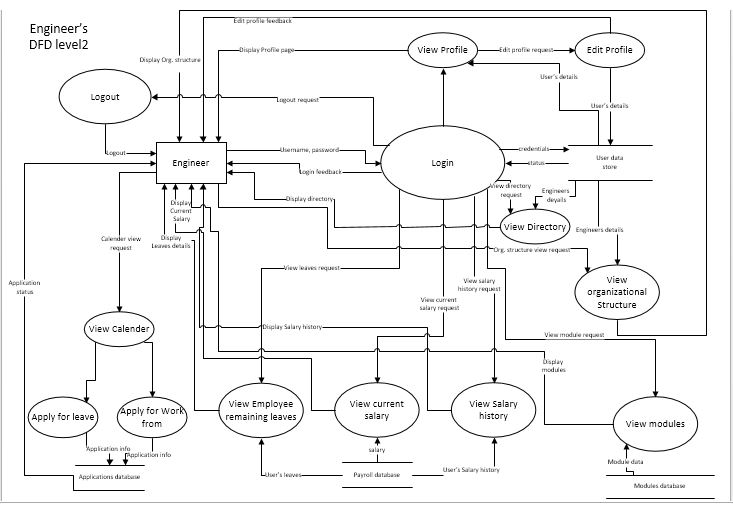
**Component Diagram:**

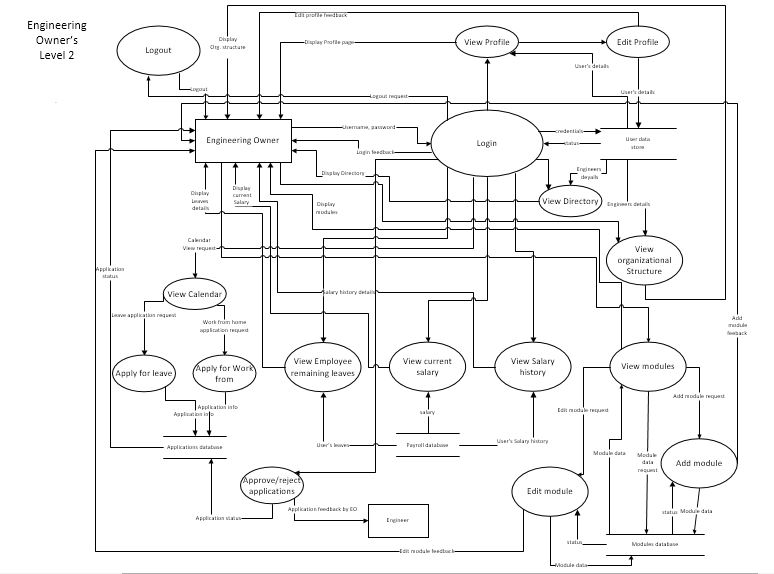


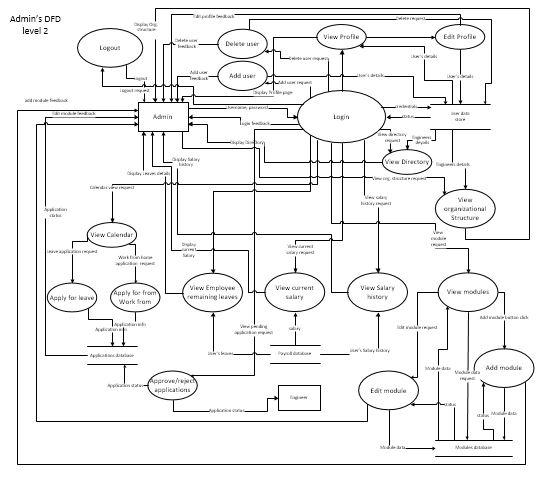
**Data Flow Diagram:**

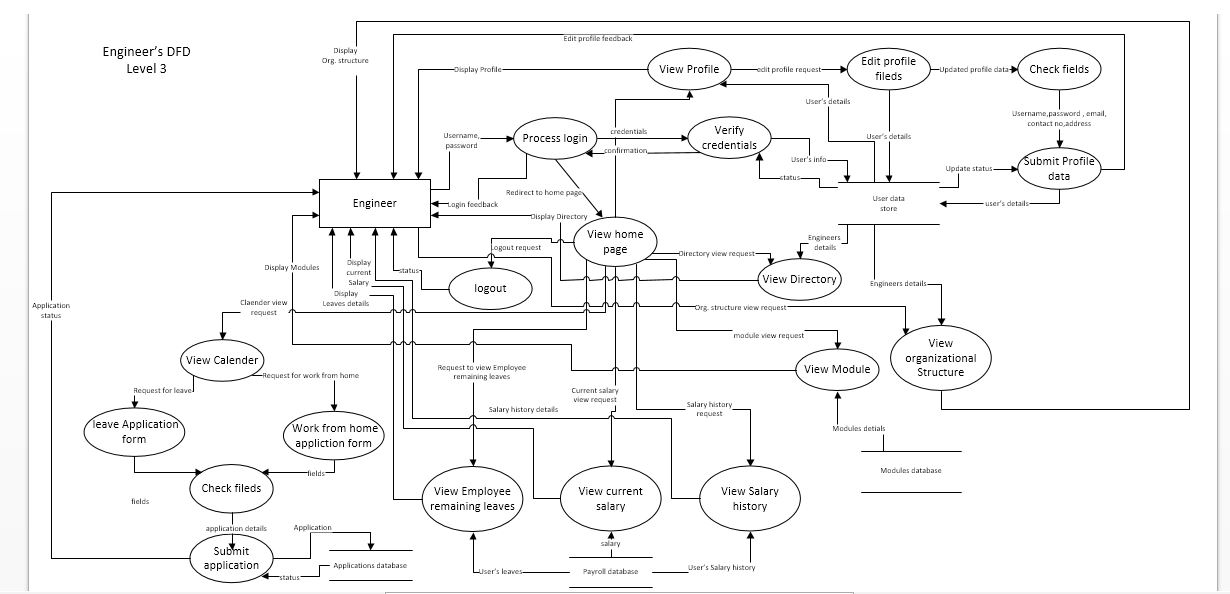
****

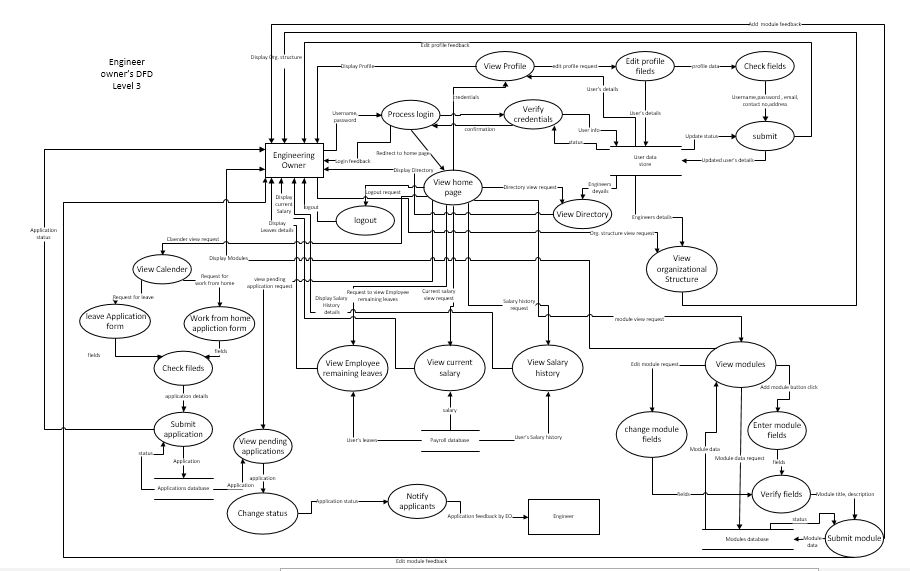
****









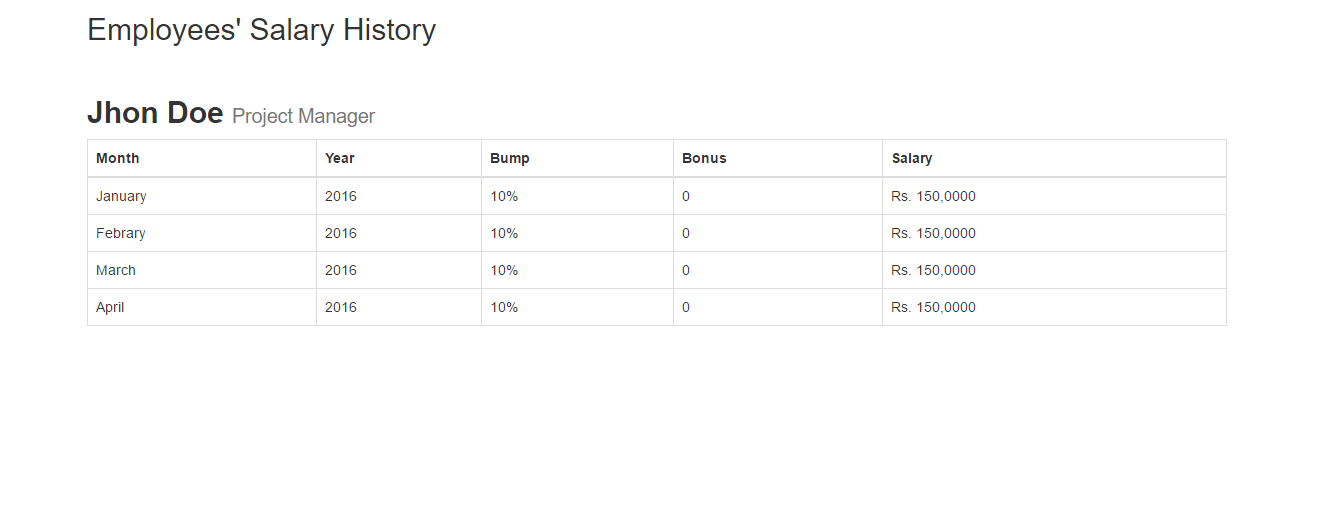
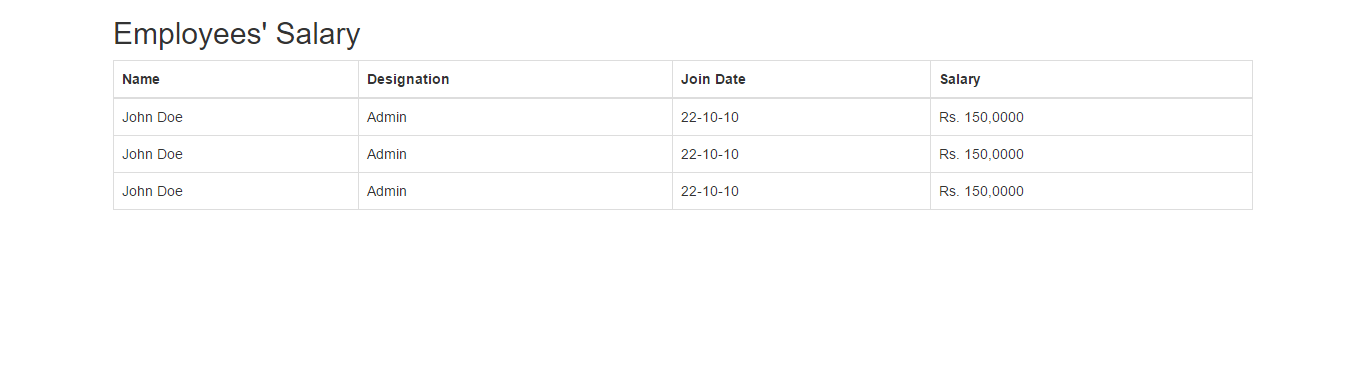
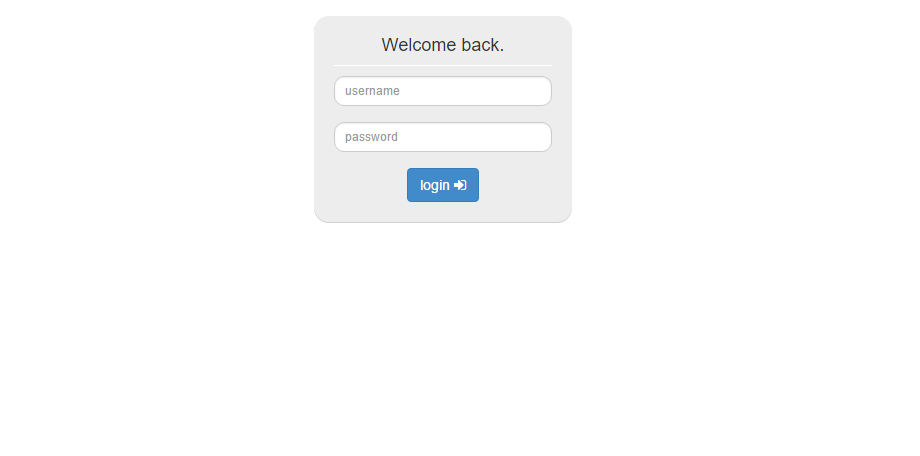


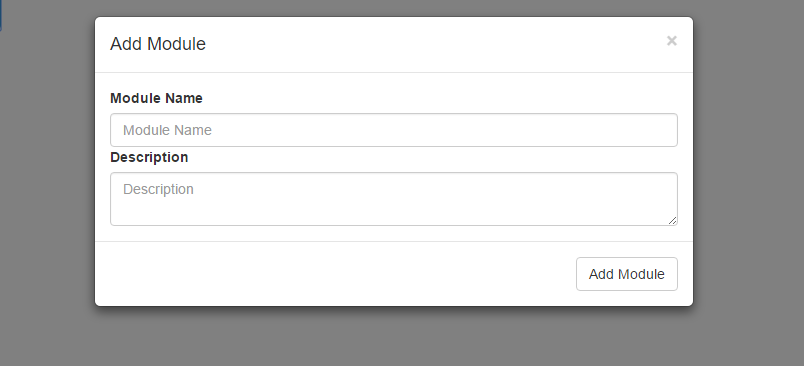
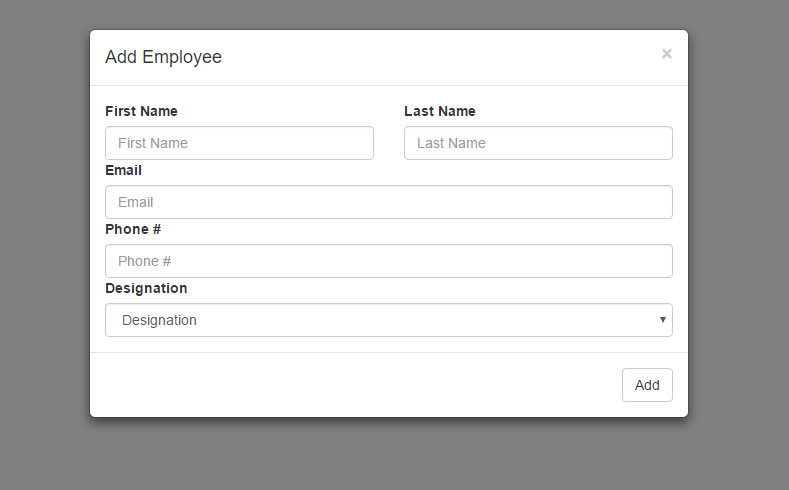
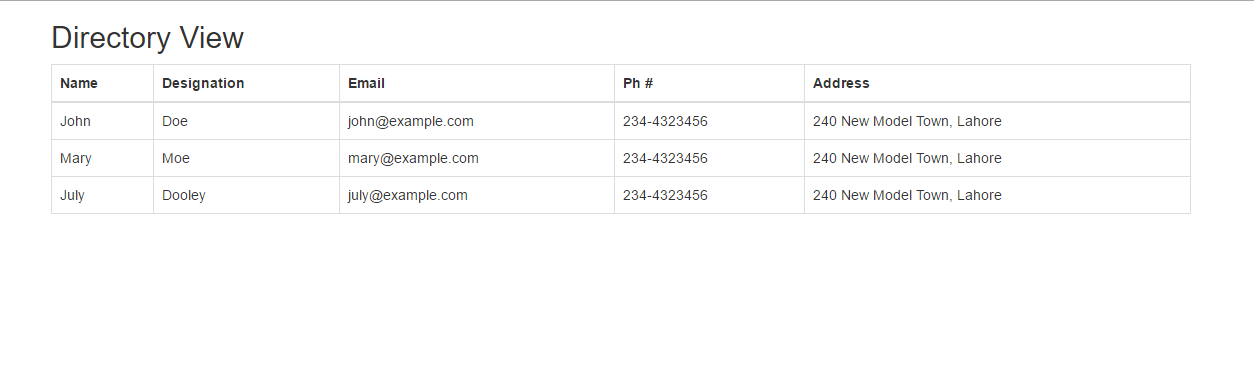
# **Admin level 3.JPG**

# 

# 

# **Appendix D: Screenshots**

**



# 

# 

# 

# 

# **Appendix C: Design Models**

# 

# 

# **Appendix E: Test Cases**

|  |  |
| --- | --- |
| **Identifier** | TC-1 |
| **Priority** | High |
| **Related requirements(s)** | UC-1 |
| **Short description** | Check login feature. |
| **Pre-condition(s)** | User account must exist. |
| **Input data** | Username=”ahsen”,  password=”12345” |
| **Detailed steps** | 1. Enter the above username.  2. Enter the above password.  3. Login. |
| **Expected result(s)** | User is redirected to profile page. |
| **Post-condition(s)** | User is logged in. |

|  |  |
| --- | --- |
| **Identifier** | TC-2 |
| **Priority** | High |
| **Related requirements(s)** | UC-1 |
| **Short description** | Check login feature. |
| **Pre-condition(s)** | User account must exist. |
| **Input data** | Username=”ali”,  password=”12345” |
| **Detailed steps** | 1. Enter the above username.  2. Enter the above password.  3. Login. |
| **Expected result(s)** | Display error message “Invalid username”. |
| **Post-condition(s)** | User is not logged in. |

|  |  |
| --- | --- |
| **Identifier** | TC-3 |
| **Priority** | High |
| **Related requirements(s)** | UC-1 |
| **Short description** | Check login feature. |
| **Pre-condition(s)** | User account must exist. |
| **Input data** | Username=”ahsen”, password=”54321” |
| **Detailed steps** | 1. Enter the above username.  2. Enter the above password.  3. Login. |
| **Expected result(s)** | Display error message “Invalid password”. |
| **Post-condition(s)** | User is not logged in. |

|  |  |
| --- | --- |
| **Identifier** | TC-4 |
| **Priority** | High |
| **Related requirements(s)** | UC-2 |
| **Short description** | Check add user feature. |
| **Pre-condition(s)** | Admin must be logged in. |
| **Input data** | First Name = “Farooq”,Last Name=”Khan”  Username=”farooq”,  password=”12345”,  email=”farooqkhan@gmail.com”,Address = “Lahore”, Phone no = “0300-1234567”, Designation = “Software Developer”,Rank=”Engineer”,  Salary=”10000” |
| **Detailed steps** | 1. Click on register tab.  2. Enter the above First name,Last Name,email,address, phone no,Designation,Rank,Salary.  3.click Add user. |
| **Expected result(s)** | Display feedback message “User added” |
| **Post-condition(s)** | New user is added to database. |

|  |  |
| --- | --- |
| **Identifier** | TC-5 |
| **Priority** | High |
| **Related requirements(s)** | UC-2 |
| **Short description** | Check add user feature. |
| **Pre-condition(s)** | Admin must be logged in. |
| **Input data** | First Name = “Farooq”,Last Name=”Khan”  Username=”farooq”,  password=”12345”,  email=”farooqkhan@gmail.com”,Address = “Lahore”, Phone no = “0300-1234567”, Designation = “Software Developer” |
| **Detailed steps** | 1. Click on register tab.  2. Enter the above First name,Last Name,email,address, phone no,Designation,Rank.  3.Click Add user. |
| **Expected result(s)** | Display feedback message “Please check the entered fields”. |
| **Post-condition(s)** | New user is not added to database. |

|  |  |
| --- | --- |
| **Identifier** | TC-6 |
| **Priority** | High |
| **Related requirements(s)** | UC-3 |
| **Short description** | Check edit profile feature. |
| **Pre-condition(s)** | User must be logged in. |
| **Input data** | First Name = “Farooq”,Last Name=”Khan”  Username=”farooq”,  password=”12345”,  email=”farooqkhan@gmail.com”,Address = “Lahore”, Phone no = “0300-1234567”, Designation = “Software Developer”,Salary=”10000”,  Upcoming bump=”0”,upcoming bonus=”1000” |
| **Detailed steps** | 1. Click on edit button.  1. Click on register tab.  2. Enter the above First name,Last Name,email,address, phone no,Designation,Rank,Salary,upcoming bump,upcoming bonus.  3. Save. |
| **Expected result(s)** | Information is saved. |
| **Post-condition(s)** | Profile is updated in database. |

|  |  |
| --- | --- |
| **Identifier** | TC-7 |
| **Priority** | High |
| **Related requirements(s)** | UC-4 |
| **Short description** | Check delete user feature. |
| **Pre-condition(s)** | Admin must be logged in and user’s profile must exist. |
| **Input data** | username = “ali” |
| **Detailed steps** | 1. Click on delete user tab.  2.Enter username.  3. Click Delete. |
| **Expected result(s)** | Display feedback message “Deleted successfully” |
| **Post-condition(s)** | User is removed from current users database. |

|  |  |
| --- | --- |
| **Identifier** | TC-8 |
| **Priority** | High |
| **Related requirements(s)** | UC-5 |
| **Short description** | Check view profile feature. |
| **Pre-condition(s)** | User must be logged in. |
| **Input data** |  |
| **Detailed steps** | 1. Click on “My Profile” tab. |
| **Expected result(s)** | User profile is displayed. |
| **Post-condition(s)** | User profile is displayed |

|  |  |
| --- | --- |
| **Identifier** | TC-9 |
| **Priority** | High |
| **Related requirements(s)** | UC-6 |
| **Short description** | Check apply for leave feature. |
| **Pre-condition(s)** | User must be logged in. |
| **Input data** | Apply for=”full day”,Date = “20/12/16”,  Available Leaves =”30”, Description = “Sick” |
| **Detailed steps** | 1. Click on “Apply for Leave” tab.  2. Click on leave button.  3. Enter the above date.  4. Enter the above available leaves.  5. Enter the above apply for.  6. Enter the above reason in description. |
| **Expected result(s)** | Display message “Application submitted”. |
| **Post-condition(s)** | Application saved to database. |

|  |  |
| --- | --- |
| **Identifier** | TC-10 |
| **Priority** | High |
| **Related requirements(s)** | UC-6 |
| **Short description** | Check apply for leave feature. |
| **Pre-condition(s)** | User must be logged in. |
| **Input data** | Apply for=”full day”,Date = “40/12/16”,  Available Leaves =”30”, Description = “Sick” |
| **Detailed steps** | 1. Click on “Apply for Leave” tab.  2. Click on leave button.  3. Enter the above date.  4. Enter the above available leaves.  5. Enter the above apply for.  6. Enter the above reason in description.  7.Enter Apply button. |
| **Expected result(s)** | Display message “Please check the entered dates”. |
| **Post-condition(s)** | Application not saved to database. |

|  |  |
| --- | --- |
| **Identifier** | TC-11 |
| **Priority** | High |
| **Related requirements(s)** | UC-6 |
| **Short description** | Check apply for leave feature. |
| **Pre-condition(s)** | User must be logged in. |
| **Input data** | Apply for=”full day”,Date = “40/12/16”,  Available Leaves =”30”, Description = “” |
| **Detailed steps** | 1. Click on “Apply for Leave” tab.  2. Click on leave button.  3. Enter the above date.  4. Enter the above available leaves.  5. Enter the above apply for.  6. Enter the above reason in description.  7.Enter Apply button. |
| **Expected result(s)** | Display message “Description cannot be left empty”. |
| **Post-condition(s)** | Application not saved to database. |

|  |  |
| --- | --- |
| **Identifier** | TC-12 |
| **Priority** | High |
| **Related requirements(s)** | UC-7 |
| **Short description** | Check apply for work from home feature. |
| **Pre-condition(s)** | User must be logged in. |
| **Input data** | Apply for=”Work from Home”,Date = “30/12/16”,  Description = “Sick” |
| **Detailed steps** | 1. Click on “Apply for leave/WFH” tab.  2. Click on work from home button.  3. Enter the above start date.  4. Enter the above end date.  5. Enter the above reason in description. |
| **Expected result(s)** | Display message “Application submitted”. |
| **Post-condition(s)** | Application saved to database. |

|  |  |
| --- | --- |
| **Identifier** | TC-13 |
| **Priority** | High |
| **Related requirements(s)** | UC-7 |
| **Short description** | Check apply for work from home feature. |
| **Pre-condition(s)** | User must be logged in. |
| **Input data** | Apply for=”full day”,Date = “40/12/16”,  Available Leaves =”30”, Description = “Sick” |
| **Detailed steps** | 1. Click on “Apply for leave/WFH” tab.  2. Click on work from home button.  3. Enter the above start date.  4. Enter the above end date.  5. Enter the above reason in description |
| **Expected result(s)** | Display message “Please check the entered dates”. |
| **Post-condition(s)** | Application not saved to database. |

|  |  |
| --- | --- |
| **Identifier** | TC-14 |
| **Priority** | High |
| **Related requirements(s)** | UC-7 |
| **Short description** | Check apply for work from home feature. |
| **Pre-condition(s)** | User must be logged in. |
| **Input data** | Apply for=”Work from home”,Date = “40/12/16” |
| **Detailed steps** | 1. Click on “Apply for leave/WFH” tab.  2. Click on work from home button.  3. Enter the above start date.  4. Enter the above end date.  5. Enter the above reason in description  6. Click apply button. |
| **Expected result(s)** | Display message “Please enter all fields”. |
| **Post-condition(s)** | Application not saved to database. |

|  |  |
| --- | --- |
| **Identifier** | TC-15 |
| **Priority** | High |
| **Related requirements(s)** | UC-8 |
| **Short description** | Check approve applications feature. |
| **Pre-condition(s)** | User must be logged in. |
| **Input data** |  |
| **Detailed steps** | 1. Click on pending applications button.  2. Click on an application.  3. Approve. |
| **Expected result(s)** | Records application status and notifies applicant. |
| **Post-condition(s)** |  |
|  |  |
| **Identifier** | TC-16 |
| **Priority** | High |
| **Related requirements(s)** | UC-8 |
| **Short description** | Check reject applications feature. |
| **Pre-condition(s)** | User must be logged in. |
| **Input data** |  |
| **Detailed steps** | 1. Click on pending applications button.  2. Click on an application.  3. Reject. |
| **Expected result(s)** | Records application status and notifies applicant. |
| **Post-condition(s)** |  |

|  |  |
| --- | --- |
| **Identifier** | TC-17 |
| **Priority** | High |
| **Related requirements(s)** | UC-9 |
| **Short description** | Check view calendar feature. |
| **Pre-condition(s)** | User must be logged in. |
| **Input data** |  |
| **Detailed steps** | 1. Click on “Calendar” tab. |
| **Expected result(s)** | Displays calendar page. |
| **Post-condition(s)** |  |

|  |  |
| --- | --- |
| **Identifier** | TC-18 |
| **Priority** | High |
| **Related requirements(s)** | UC-10 |
| **Short description** | Check view directory feature. |
| **Pre-condition(s)** | User must be logged in. |
| **Input data** |  |
| **Detailed steps** | 1. Click on “Directory” tab. |
| **Expected result(s)** | Displays directory page. |
| **Post-condition(s)** |  |

|  |  |
| --- | --- |
| **Identifier** | TC-19 |
| **Priority** | High |
| **Related requirements(s)** | UC-11 |
| **Short description** | Check view modules feature. |
| **Pre-condition(s)** | User must be logged in. |
| **Input data** |  |
| **Detailed steps** | 1. Click on “Modules” tab. |
| **Expected result(s)** | Displays all modules of the organization. |
| **Post-condition(s)** |  |

|  |  |
| --- | --- |
| **Identifier** | TC-20 |
| **Priority** | High |
| **Related requirements(s)** | UC-12 |
| **Short description** | Check add module feature. |
| **Pre-condition(s)** | User must be logged in. |
| **Input data** | Title = “Web Service”, Description = “Connect with the Services” |
| **Detailed steps** | 1. Click on “Add module”.  2. Enter the above title.  3. Enter the above description. |
| **Expected result(s)** | Display message “Module added”. |
| **Post-condition(s)** | Module added to database. |

|  |  |
| --- | --- |
| **Identifier** | TC-21 |
| **Priority** | High |
| **Related requirements(s)** | UC-12 |
| **Short description** | Check add module feature. |
| **Pre-condition(s)** | User must be logged in. |
| **Input data** | Description = “Connect with other Services” |
| **Detailed steps** | 1. Click on “Add module”.  2. Enter the above description. |
| **Expected result(s)** | Display message “Please enter all fields”. |
| **Post-condition(s)** | Module not added to database. |

|  |  |
| --- | --- |
| **Identifier** | TC-22 |
| **Priority** | High |
| **Related requirements(s)** | UC-12 |
| **Short description** | Check add module feature. |
| **Pre-condition(s)** | User must be logged in. |
| **Input data** | Title = “Web services” |
| **Detailed steps** | 1. Click on “Add module”.  2. Enter the above title. |
| **Expected result(s)** | Display message “Please enter all fields”. |
| **Post-condition(s)** | Module not added to database. |

|  |  |
| --- | --- |
| **Identifier** | TC-23 |
| **Priority** | Medium |
| **Related requirements(s)** | UC-13 |
| **Short description** | Check edit module feature. |
| **Pre-condition(s)** | User must be logged in. |
| **Input data** | Title = “Mobile Service”, Description = “connecting with mobile devices” |
| **Detailed steps** | 1. Select a module to edit.  2. Click on “Edit module”.  3. Enter the above title.  4. Enter the above description.  5. Save. |
| **Expected result(s)** | Display message “Module Updated”. |
| **Post-condition(s)** | Module updated in database. |

|  |  |
| --- | --- |
| **Identifier** | TC-24 |
| **Priority** | Low |
| **Related requirements(s)** | UC-14 |
| **Short description** | Check view organizational structure feature. |
| **Pre-condition(s)** |  |
| **Input data** |  |
| **Detailed steps** | 1. Click on “Organizational Structure” tab. |
| **Expected result(s)** | Displays the organizational structure. |
| **Post-condition(s)** |  |

|  |  |
| --- | --- |
| **Identifier** | TC-25 |
| **Priority** | High |
| **Related requirements(s)** | UC-15 |
| **Short description** | Check view employee’s remaining leaves feature. |
| **Pre-condition(s)** | User must be logged in. |
| **Input data** | username = “ahsen” |
| **Detailed steps** | 1. Click on “profile” tab.  2.view remaining leaves. |
| **Expected result(s)** | Display employee’s remaining leaves. |
| **Post-condition(s)** |  |

|  |  |
| --- | --- |
| **Identifier** | TC-26 |
| **Priority** | High |
| **Related requirements(s)** | UC-16 |
| **Short description** | Check view employee’s current salary feature. |
| **Pre-condition(s)** | User must be logged in. |
| **Input data** | username = “ahsen” |
| **Detailed steps** | 1. Click on “profile” tab.  2. View salary |
| **Expected result(s)** | Display current salary of employee. |
| **Post-condition(s)** |  |

|  |  |
| --- | --- |
| **Identifier** | TC-27 |
| **Priority** | High |
| **Related requirements(s)** | UC-17 |
| **Short description** | Check view employee’s salary history feature. |
| **Pre-condition(s)** | User must be logged in. |
| **Input data** |  |
| **Detailed steps** | 1. Click on “Employee” tab.  2. Click on “Salary” tab.  3. Use filter option to search for specific employee.  4. Click on “History” button. |
| **Expected result(s)** | Display salary history of employee. |
| **Post-condition(s)** |  |

|  |  |
| --- | --- |
| **Identifier** | TC-28 |
| **Priority** | High |
| **Related requirements(s)** | UC-18 |
| **Short description** | Check edit employee’s salary feature. |
| **Pre-condition(s)** | Admin must be logged in. |
| **Input data** | bonus = “2000”,  bump = “1000”,  salary = “10000” |
| **Detailed steps** | 1. Click on “Edit profile” tab.  2. Enter the above bonus, bump, salar  6. Save button. |
| **Expected result(s)** | Employee’s salary is updated. |
| **Post-condition(s)** | Updated salary is saved in database. |

# **Non-Functional Requirements:**

|  |  |
| --- | --- |
| **Identifier** | TC-29 |
| **Priority** | High |
| **Related requirements(s)** | Performance requirement 4.1.1 |
| **Short description** | Check for use of Asynchronous paradigm. |
| **Pre-condition(s)** | Asynchronous language is used for implementation |
| **Input data** | User request |
| **Detailed steps** | 1. Multiple requests can be handled |
| **Expected result(s)** | Handled all concurrent requests with high throughput on a single thread. |
| **Post-condition(s)** | Executed all concurrent requests successfully. |

|  |  |
| --- | --- |
| **Identifier** | TC-30 |
| **Priority** | High |
| **Related requirements(s)** | Performance requirement 4.1.2 |
| **Short description** | Check for AJAX request to import employee salary history. |
| **Pre-condition(s)** | Dynamic web application development technology used in implementation |
| **Input data** | Employee ID=”” |
| **Detailed steps** | 1. Enter the employee ID  2. Click on the search button. |
| **Expected result(s)** | Show the specific employee salary |
| **Post-condition(s)** | Employee salary history is displayed on the page. |

|  |  |
| --- | --- |
| **Identifier** | TC-31 |
| **Priority** | Medium |
| **Related requirements(s)** | Performance requirement 4.1.3 |
| **Short description** | Display employee records within 3 seconds response time. |
| **Pre-condition(s)** | System should have employee’s data in the Database, stable internet connection |
| **Input data** | Employee ID=”” |
| **Detailed steps** | 1. Send Request  2. Click on the search button. |
| **Expected result(s)** | Display all results within 3 seconds. |
| **Post-condition(s)** | Employee’s record displayed on the page. |

|  |  |
| --- | --- |
| **Identifier** | TC-32 |
| **Priority** | High |
| **Related requirements(s)** | Performance requirement 4.2.1 |
| **Short description** | Test whether the system is vulnerable to attack |
| **Pre-condition(s)** | System does not accept special characters |
| **Input data** | Special Characters,SQL queries in text fields. |
| **Detailed steps** | 1.Enter SQL query in any text field |
| **Expected result(s)** | Display message “Invalid input”. |
| **Post-condition(s)** | Unauthorized user could not login to the system. |

# 

# **Appendix F: IV & V Report**

**IV & V Resource**

Name : Anum Shahbaz Roll #: L13-5865 Signature:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S#** | **Defect Description** | **Origin Stage** | **Status** | **Fix Time** | |
| **Hours** | **Minutes** |
| 1 | Input data for current project and project lead missing in TC-4 | Implementation and Testing | Closed |  | 5 |
| 2 | Incorrect expected result in TC-7 | Implementation and Testing | Closed |  | 5 |
| 3 | Test case in case of invalid input fields missing for UC-18 | Implementation and Testing | Open |  |  |
| 4 |  | Implementation and Testing |  |  |  |
| 5 |  | Implementation and Testing |  |  |  |
| 6 |  |  |  |  |  |
| 7 |  |  |  |  |  |
| 8 |  |  |  |  |  |
| 9 |  |  |  |  |  |
| 10 |  |  |  |  |  |
| 11 |  |  |  |  |  |
| 12 |  |  |  |  |  |

**Table 3: List of non-trivial defects**

# 

# 

# **Appendix G: Risk Report**

**[1]Project Risks**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Risk Description** | **Impact**  **(1 – 10)** | **Probability**  **(0 – 1)** | **[2]Risk**  **Exposure** | **Weeks Active** | **Mitigation Strategy** |
| Possible workload due to other courses projects and assignments | 7 | 0.8 | 5.6 | 2 | Proper time management work distribution. |
| Possible unavailability of a group member for meeting | 5 | 0.6 | 3.0 | 2 | Taking notes during meeting to help left out member know what we discussed |
| Possible data loss. | 8 | 0.25 | 4 | 2 | Online backup |
| Possible sickness of somebody | 6 | 0.25 | 1.5 | 2 | Working in pairs |

# 

# **Appendix H: Activity Timesheet**

|  |  |  |
| --- | --- | --- |
| **Activity** | **Time** | |
| **Hours** | **Minutes** |
| Requirements Engineering | 30 | 30 |
| Analysis and Design | 65 | 40 |
| Implementation |  |  |
| Testing |  |  |
| Deployment |  |  |
| Project Management | 1+3 | 40+0 |
| IV & V | 5 | 20 |

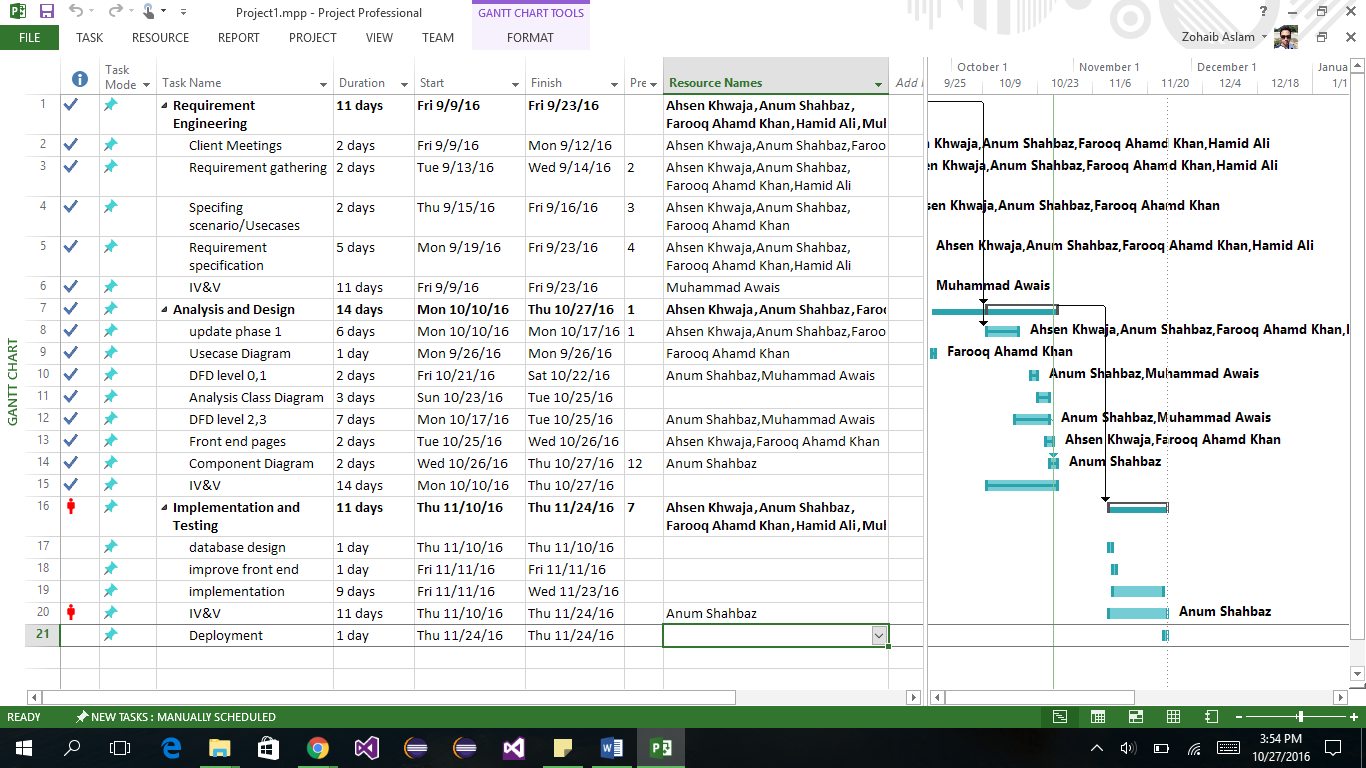
[1] Risks should be sorted in descending order of risk exposure.

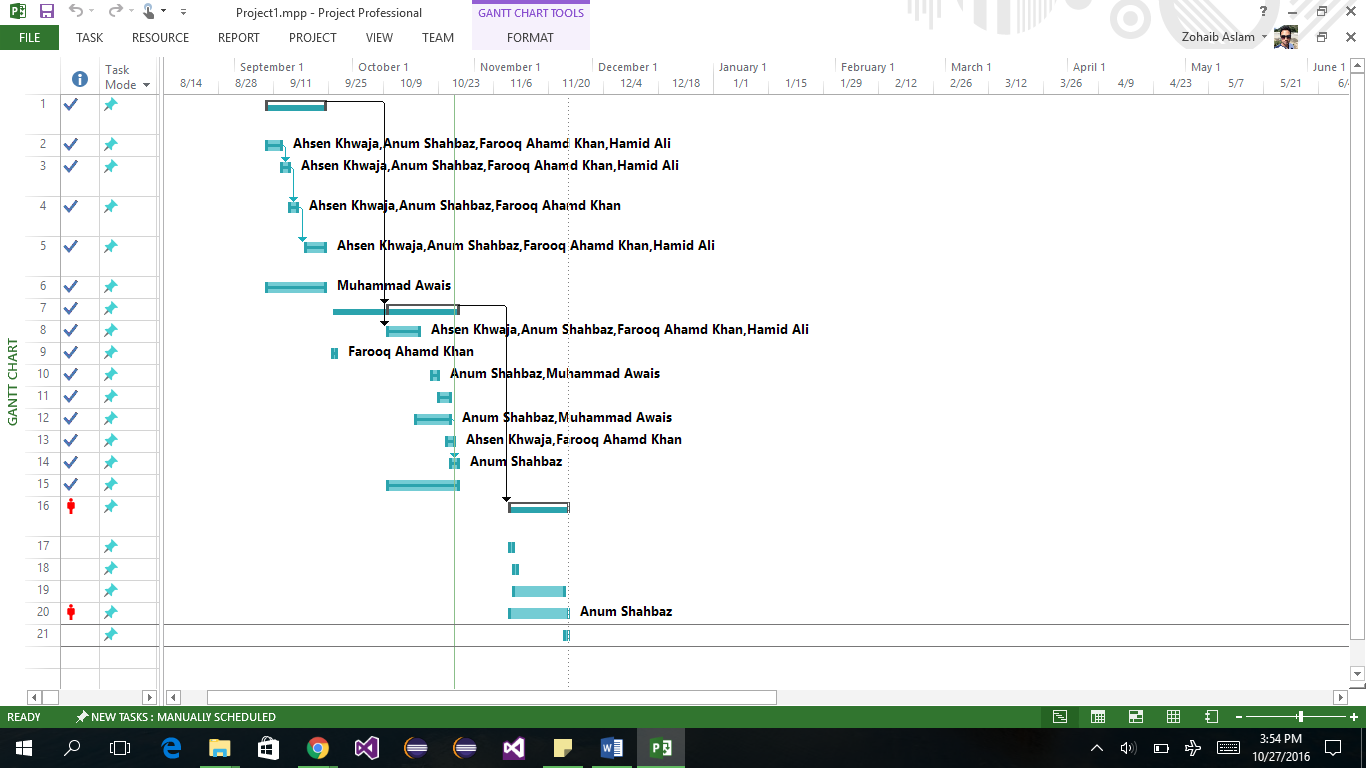
[2] Risk Exposure = Risk Impact x Risk Probability

# **Project Manager**

Name : Farooq Ahmad khan Roll # : L13-4022 Signature:

**Appendix I: Updated Project Plan**

****

****