

Bachelor of Computer Applications (BCA) Programme

Project Report

BCA Sem VI

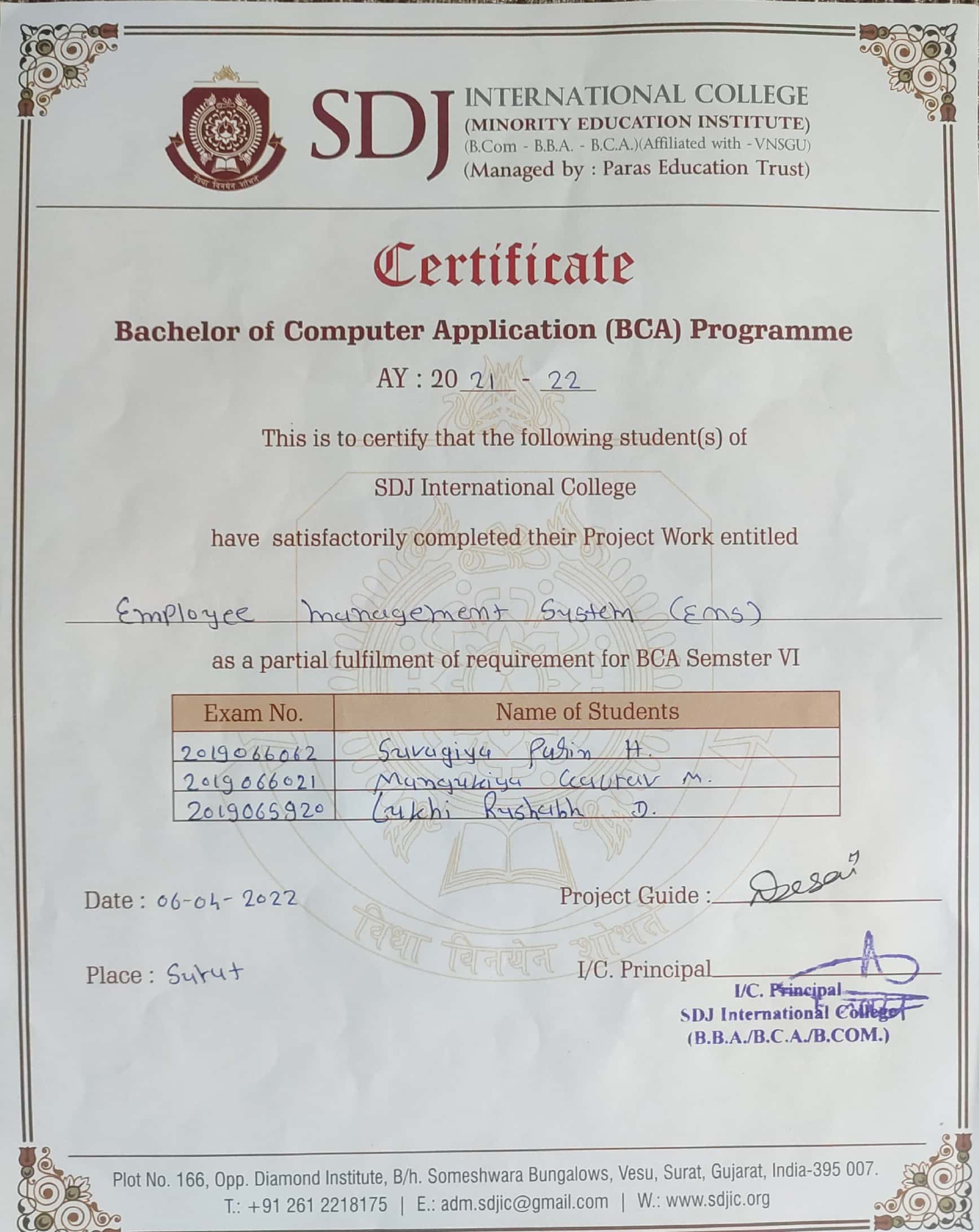
AY 2021-22

Employee Management System(EMS)

*by*

|  |  |
| --- | --- |
| Roll No. | Name of Student |
| 2019066062 | Suvagiya Parin Hareshbhai |
| 2019066021 | Mangukiya Gaurav Mukeshbhai |
| 2019065920 | Lukhi Rushabh Damjibhai |

|  |  |
| --- | --- |
| Project Guide by: | |
| Prof. Nidhi Desai |  |



**ACKNOWLEDGEMENT**

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals. we would like to extend our sincere thanks to all of them.

We are highly indebted to our **I/c Principal Dr. Aditi Bhatt**, **Head of Department Dr. Vaibhav Desai**, **Project guide Prof. Nidhi Desai** and all other Assistant professors of SDJ International College for their guidance and constant supervision as well as for providing necessary information regarding the project and also for their support in completing the project. Their constant guidance and willingness to share their vast knowledge made us understand this project and its manifestation in great Depth and helped us to complete the assigned task on time.

I would like to express my gratitude to SDJ international college for the kind support and encouragement.

Our thanks and appreciation also go to our colleagues in developing this project and people who have willingly helped me out with their abilities.

Thank you very much,

2019066062 Suvagiya Parin

2019066021 Mangukiya Gaurav

2019065920 Lukhi Rushabh

*I N D E X*

|  |  |  |
| --- | --- | --- |
| **Sr No** | **Description** | **Page No.** |
| 1 | Introduction |  |
|  | 1.1 Project Summary | 1 |
|  | 1.2 Project Technical Profile | 2 |
| 2 | Scope & Planning |  |
|  | 2.1 Requirement Analysis | 3 |
|  | 2.2 Feasibility Study | 6 |
|  | 2.3 Timeline Chart | 7 |
|  | 2.4 Future Development | 9 |
|  | 2.5 Technology Details | 9 |
| 3 | Designing |  |
|  | 3.1 Data Flow Diagram | 10 |
|  | 3.2 Use Case Diagram | 13 |
|  | 3.3 ER Diagram | 14 |
|  | 3.4 Database Design | 15 |
|  | 3.5 User Interface | 20 |
| 4 | Testing |  |
|  | 4.1 Unit Testing | 37 |
|  | 4.2 Navigation Testing | 49 |
|  | 4.3 Functional Testing | 49 |
|  | 4.4 Environment Testing | 50 |
| 5 | Conclusion | 51 |
| 6 | Bibliography | 52 |

1. **Introduction**

**1.1 Project Summary**

Employees Management System is a web application for effectively managing employees. In companies handling employee data is the most important factor, so EMS provides functionalities like scheduling important meetings, events and managing employees' leave, salary, attendance, working hours with effective sorting and filtering techniques. This system provides a portal for employees to interact with the company and track their daily data like attendance, leave, salary, working hours. EMS frontend is built using Reactjs, which is a javascript library. EMS Backend provides an Application programming interface (API) that is built using Nodejs, Expressjs, MongoDB. When user requests for EMS frontend will load on the web browser window and API will provide data on user needs.

**1.2 Project Technical Profile**

|  |  |
| --- | --- |
| Project Name: | EMS (Employee Management System) |
| Project Definition/Aim: | The main aim of the project is to provide better functionalities for managing employees' data. |
| Develop for: | Small companies who can not afford to make software to manage their employees. |
| Front End: | JavaScript ReactJS&Redux |
| Backend: | API in Backend |
| Other Technologies: | Bootstrap 5, Material UI, fullcalendar |
| Documentation tools: | Google docs |
| IDE: | Visual studio code |
| Software type: | Web Application |
| Project Guide: | Prof. Nidhi Desai |
| Submitted By: | 1. Suvagiya Parin H. 2. Mangukiya Gaurav M. 3. Lukhi Rushabh D. |

1. **Scope & Planning**

**2. Scope**

* The scope of this system is very large, it can be used in multiple companies. Using EMS they can manage their employee data and they do not need to set up this system because it maintains multiple companies' data at a time through a web interface.–

**2.1 System Functional Requirements**

**Home page**

R1: Provide an interactive home page with redirect links to different pages.

**For companies or company portal**

R1: LOGIN, REGISTER with OTP, and JWT token authentication

* Company can create an account and by login get access to other functionalities for managing employees.

R2: Dashboard

* Provide statistics of employees on the dashboard.

R3: View and Update profile and password

* Company can view their profile and also can update it.
* If the company updates the email reverify the email.

R4: Manage Departments

* Company can view, add, update, and delete departments in this module.

R5: Manage Designations

* Company can view, add, update, and delete designations in this module.

R6: Manage Employees

View employees

- shows a list of employees with search and sorting functionality.

Add employees

* The company can add employees by login into the company portal as soon as the company adds a new employee. Employees must get credentials for the employee portal.

Update employees

* Company has access to update employee details.

leaving employees

* When employees leave provide a field to add leaving date.

Replace manager

* If the manager leaves a company, an old manager can be replaced with a new manager.

R6: Scheduler

* Provide calendar UI and by clicking date company can add events and employees must be able to view events added by the company.
* The company can edit and delete events.

R7: Attendance

* Provide functionality to filter employees and start their shift and also view old attendance records.
* Provide a list of employees who have started a shift today and as well as provide a list of employees who have not started the shift today.

R8: Mange leaves

* Employees can request by login into the employee portal. Show a leaves list and provide the functionality to accept and reject requests.
* Show a list of employees who are on leave today.
* Provide functionality to view old leaves records of particular employees.

R9: Salary

* Filter employees and calculate the salary of employees between two dates by shift calculated by the company, and employee.

**For employees and employee portal**

R1: Login

* When a company adds employees. The employee will get auto generated id and password for the employee portal.

R2: view company profile

* Employees can view the company profile.

R3: view my profile and update password

* View employee detail and their manager detail

R3: Scheduler

* View events added by the company.
* Employees also can add public and private events.

R4: Attendance

* Employees also can start their shift.
* Show list of attendance records

R5: leaves

* Request for leave
* View old leave records

R6: Salary

* Calculate salary according to shift stared by company or employee.

**2.1 Other Non-Functional Requirements**

**Performance and scalability.** How fast does the system return results? How much will this performance change with higher workloads?

**Portability and compatibility.** Which hardware, operating systems, browsers, and their versions does the software run on? Does it conflict with other applications and processes within these environments?

**Reliability, availability, maintainability.** How often does the system experience critical failures? and how much time is it available to users against downtimes?

**Security.** How are the system and its data protected against attacks?

**Localization.** Does the system match local specifics?

**Usability.** How easy is it for a customer to use the system?

**2.2 Feasibility study**

This system is designed to help small companies to manage their employees with functionalities like adding, and managing employee data like leave, salary, attendance, and event calendar after the technical, economic, operational, legal, and schedule evolution of the project we have found the risk of this project and decided to spend time on this project or not, But this project is very useful for small businesses with current functionality. These all factors can be handled so we are going to work on this project.

**Technical Feasibility**

In the technical feasibility study we have found the requirement of hardware and software that can be fulfilled by us.

**Economic Feasibility**

In the economic feasibility study we analyse the cost of the project and in the first phase, all requirements can be fulfilled with a minimal amount.

**Operational Feasibility**

In the Operational Feasibility study we analyse some factors like can we maintain this project operation easily and after compilation of the project will it be easy to maintain.

**2.3 Timeline chart**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Work task** | **Dec** | | **Jan** | | | | **Feb** | | | | **March** | | | |
| Week | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| **1 Requirement Gathering & Analysis** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 1.1 Learn Javascript |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 1.2 requirement gathering |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 1.3 requirement analysis |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **2 Modelling** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2.1 identify proposed project profile |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2.2 identifying objectives |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2.3 scope definition |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **3 System design** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.1 Design system flow |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.2 database design |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.3 company portal design |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.4 employee portal design |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **4 coding** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4.1 API architecture setup |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4.2 API coding |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4.3 API integration with frontend and implementation of cicd pipeline GitHub |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **5 Testing** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5.1 unit testing |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5.2 navigational testing |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5.3 functional testing |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5.4 environmental testing |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

**2.4 Future development**

* Verification with OTP and token
* Employee salary reports
* Employee leave and attendance reports
* Take attendance by face recognition

**2.5 Description of Tools and Technology Used**

1. **RecatJs**

* React. js is an open-source JavaScript library that is used for building user interfaces specifically for single-page applications. It's used for handling the view layer for web and mobile apps. React also allows us to create reusable UI components.

1. **Redux**

* React-Redux is the official React binding for Redux. It allows React components to read data from a Redux Store, and dispatch Actions to the Store to update data. Redux helps apps to scale by providing a sensible way to manage the state through a unidirectional data flow model. React Redux is conceptually simple.

1. **NodeJs**

* Node. js (Node) is an open-source development platform for executing JavaScript code server-side. Node is useful for developing applications that require a persistent connection from the browser to the server and is often used for real-time applications such as chat, news feeds, and web push notifications.

1. **ExpressJs**

* Express is a web application framework for Node. js that allows you to spin up robust APIs and web servers in a much easier and cleaner way. It is a lightweight package that does not obscure the core Node.

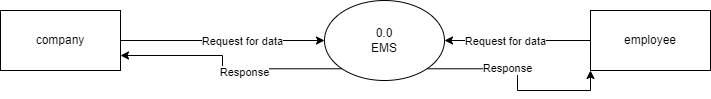
1. **What is API?**

* An API (Application Programming Interface) is a set of functions that allows applications to access data and interact with external software components, operating systems, or micro services. To simplify, an API delivers a user response to a system and sends the system's response back to a user.

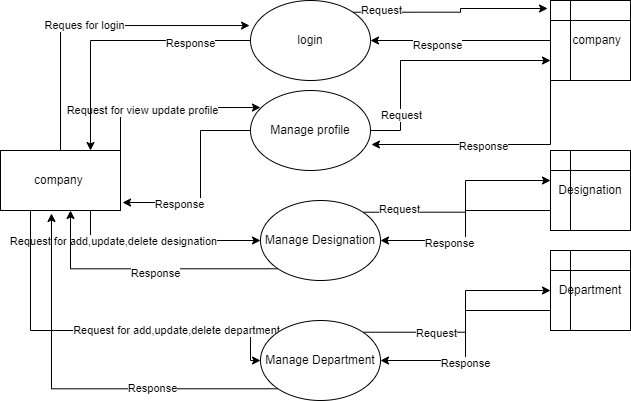
1. **Designing**

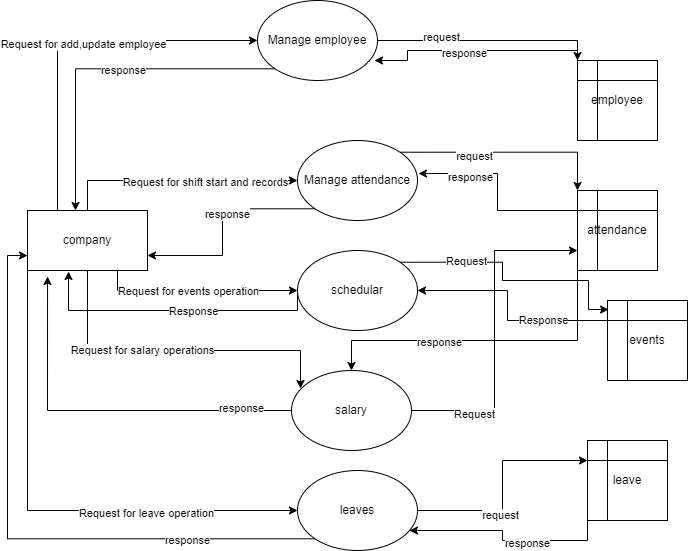
**3.1 Data flow diagram**

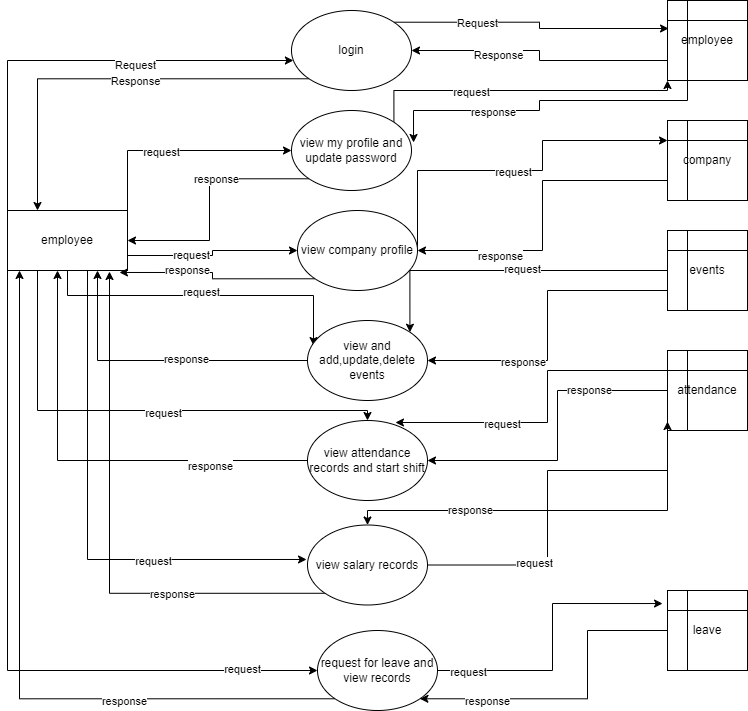
**Context level DFD**



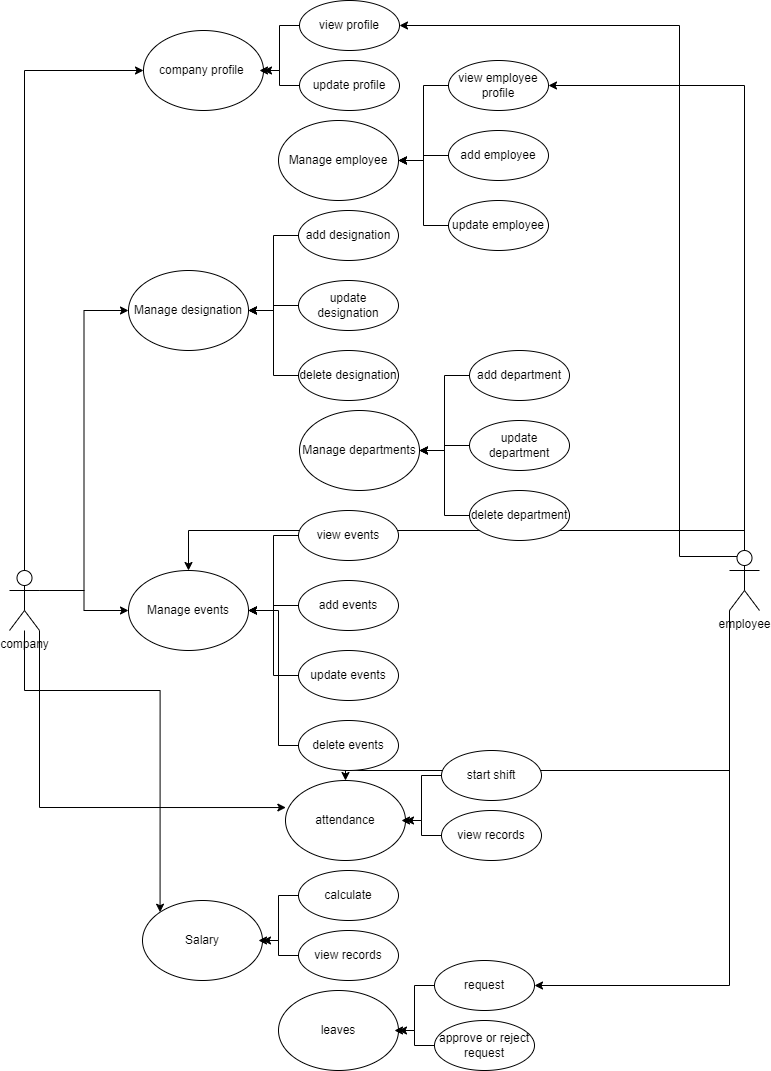
**First level DFD**



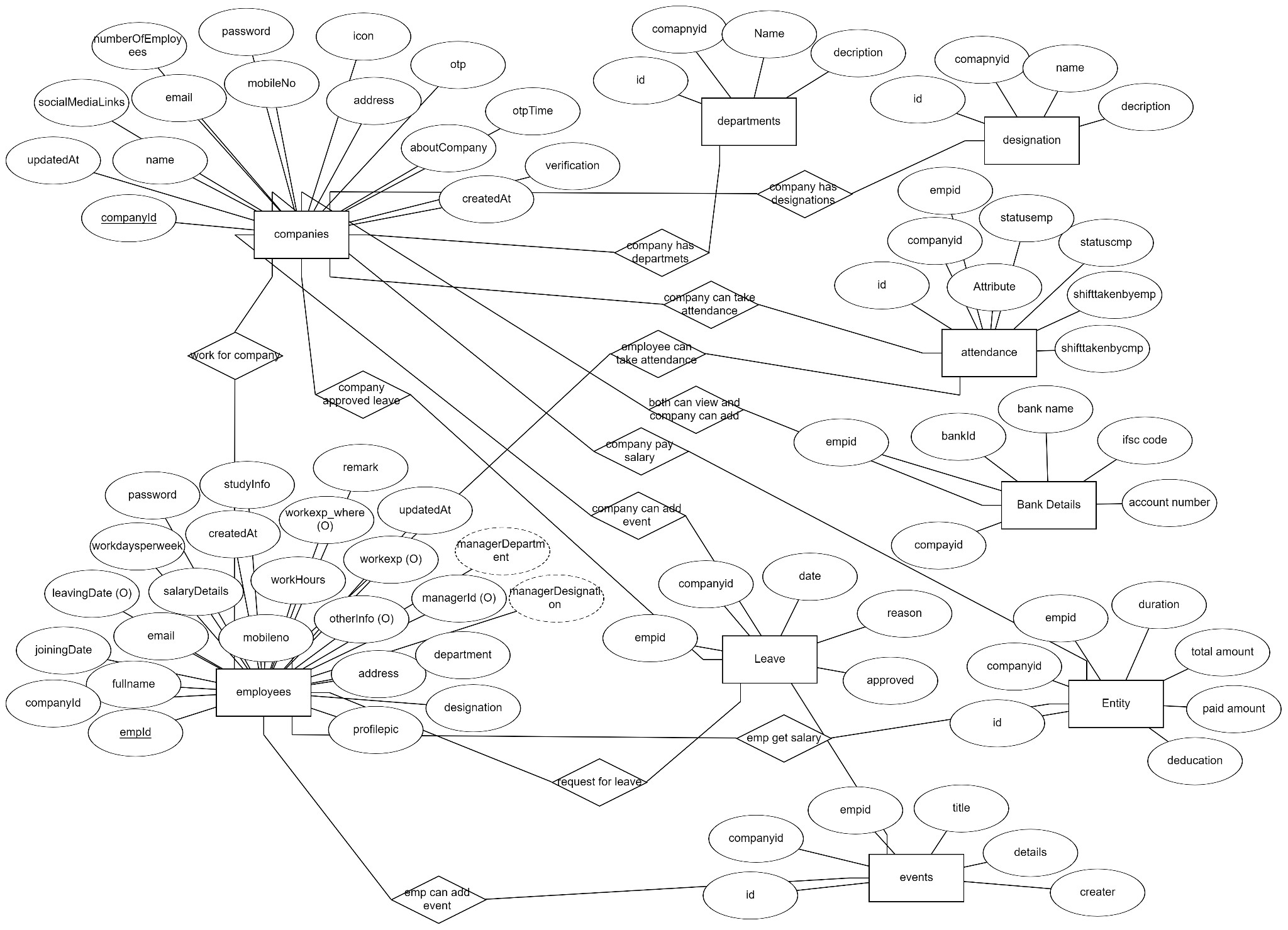




**3.2 Use Case Diagram**



**3.3 ER Diagram**



* 1. **Database design**

1. Companies

|  |  |  |  |
| --- | --- | --- | --- |
| **NAME** | **DATATYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **id** | ObjectID | PK | Company id |
| **name** | string | Required | Company name |
| **email** | string | Required | Company email |
| **mobileno** | number | Required | Company mobileno |
| **address** | string | Required | Company address |
| **password** | string | Required | Company password |
| **about company** | string | Required | Company information |
| **icon** | string | Required | Company icon |
| **numberOfemployee** | number | Required | Number of employees |
| **links** | Object Array |  | Company links |
| **CreatedAt** | date | Required | Creation time |
| **UpdatedAt** | date |  | Updation time |
| **Registrationverification** | boolean | Required | Verification status |
| **otptime** | number | Required | Otp generation time |
| **OTP** | number | Required | Otp is sent to user |

2. Employees

|  |  |  |  |
| --- | --- | --- | --- |
| **NAME** | **DATATYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **id** | ObjectID | PK | Employee id |
| **fullname** | string | Required | Employee name |
| **email** | string | Required | Employee email |
| **mobileno** | number | Required | Employee mobileno |
| **address** | string | Required | Employee address |
| **profilepic** | string | Required | Employee picture |
| **designation** | ObjectID | FK | Reference designations |
| **department** | ObjectID | FK | Reference departments |
| **workexp** | number |  | Number of months |
| **workexp\_where** | string |  | Details about where emp worked |
| **studyinfo** | string | Required | Study information |
| **remark** | string |  | Remark if any |
| **otherdetails** | string |  | Other details |
| **companyid** | ObjectID | FK | Reference companies |
| **createdAt** | date | Required | Creation time |
| **updatedAt** | date |  | Updation time |
| **joiningdate** | date | Required | Joining date |
| **leavingdate** | date |  | Leaving date |
| **password** | string | Required | password |
| **workhours** | number | Required | Working hours daily |
| **hourlysalary** | number | Required | Per hour salary |
| **currency** | string | Required | Currency |
| **gender** | string | Required | Gender |
| **attendancestatus** | boolean | Required | Status of shift |

3. Departments

|  |  |  |  |
| --- | --- | --- | --- |
| **NAME** | **DATATYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **id** | ObjectID | PK | department Id |
| **Companyid** | ObjectID | FK | Reference companies |
| **departmentname** | string | Required | Department name |
| **description** | string | Required | Department description |

4. Designations

|  |  |  |  |
| --- | --- | --- | --- |
| **NAME** | **DATATYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **id** | ObjectID | PK | Designation Id |
| **Companyid** | ObjectID | FK | Reference companies |
| **Designationname** | string | Required | Designation name |
| **description** | string | Required | Designation description |

5. Attendance

|  |  |  |  |
| --- | --- | --- | --- |
| **NAME** | **DATATYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **id** | ObjectID | PK | Attendance Id |
| **Employeeid** | ObjectID | FK | Reference employees |
| **Companyid** | ObjectID | FK | Reference companies |
| **Date** | date | Required | Attendance date |
| **Statuscmp** | Boolean |  | Status shift |
| **Statusemp** | Boolean |  | Status shift |
| **shiftyTakenByEmployee** | Object array |  | Object array of shifts |
| **shiftyTakenByCompany** | Object array |  | Object array of shifts |

6. Events

|  |  |  |  |
| --- | --- | --- | --- |
| **NAME** | **DATATYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **id** | ObjectID | PK | Event Id |
| **Companyid** | ObjectID | FK | Reference companies |
| **Employeeid** | ObjectID | FK | Reference employees |
| **Title** | string | Required | Event title |
| **Start** | date | Required | Event start time |
| **End** | date | Required | Event end time |
| **allDay** | Boolean |  | All day event |
| **Url** | string |  | Redirect link |
| **Backgroundcolor** | string |  | Color |
| **type** | string | Required | Event type public or private |
| **Iscompany** | Boolean | Required | Added by a company or not |
| **Bordercolor** | string |  | Color |

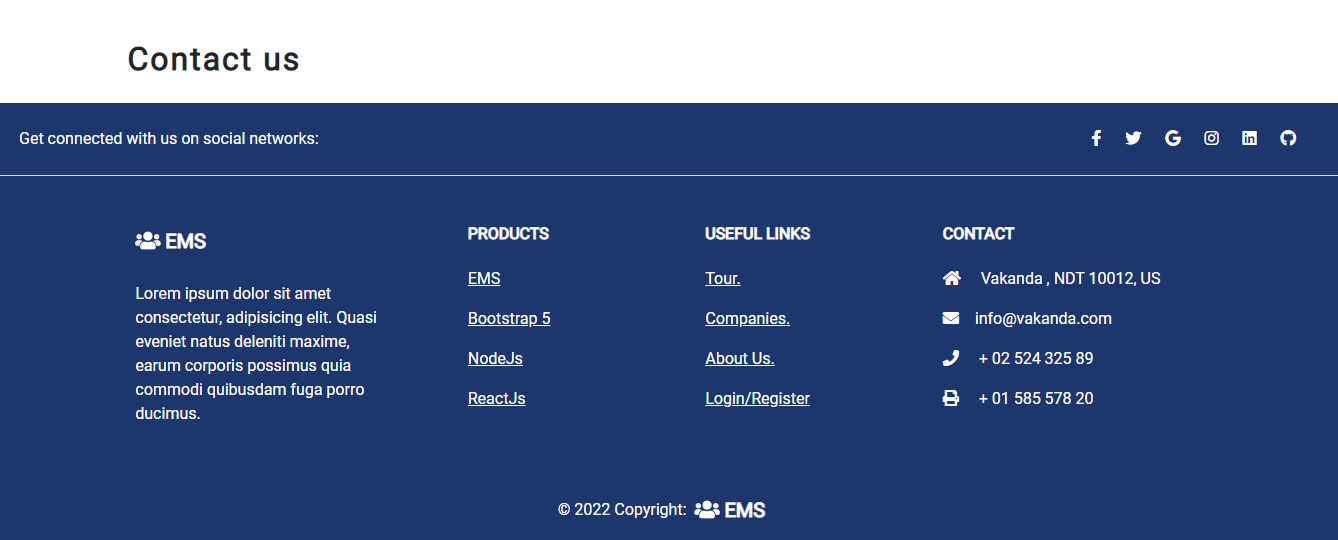
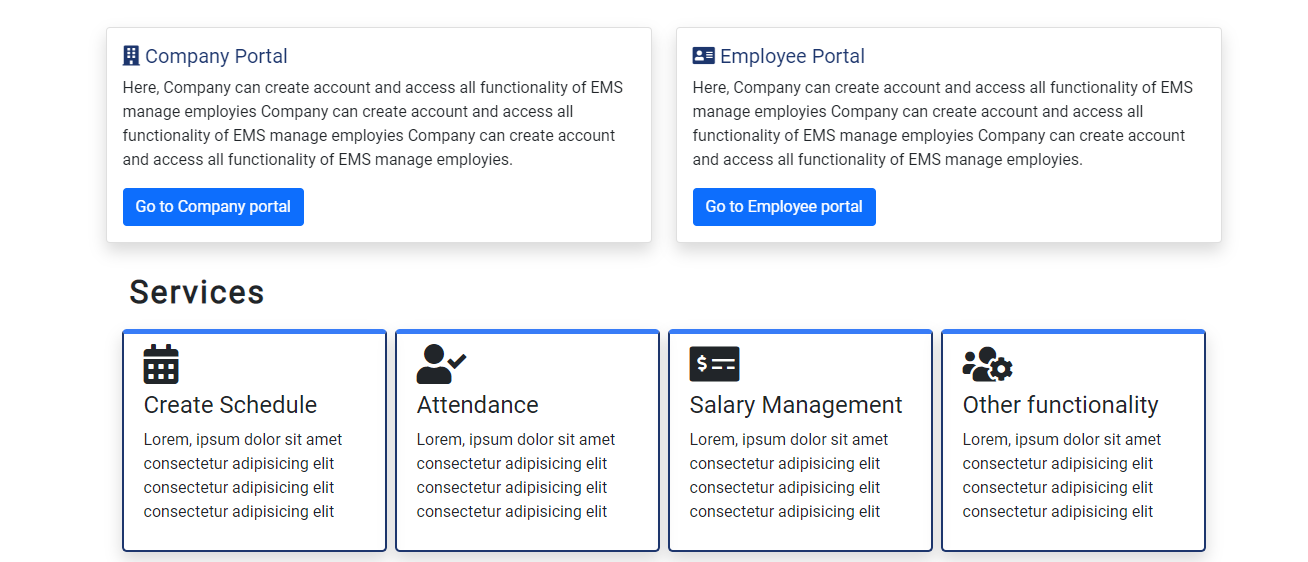
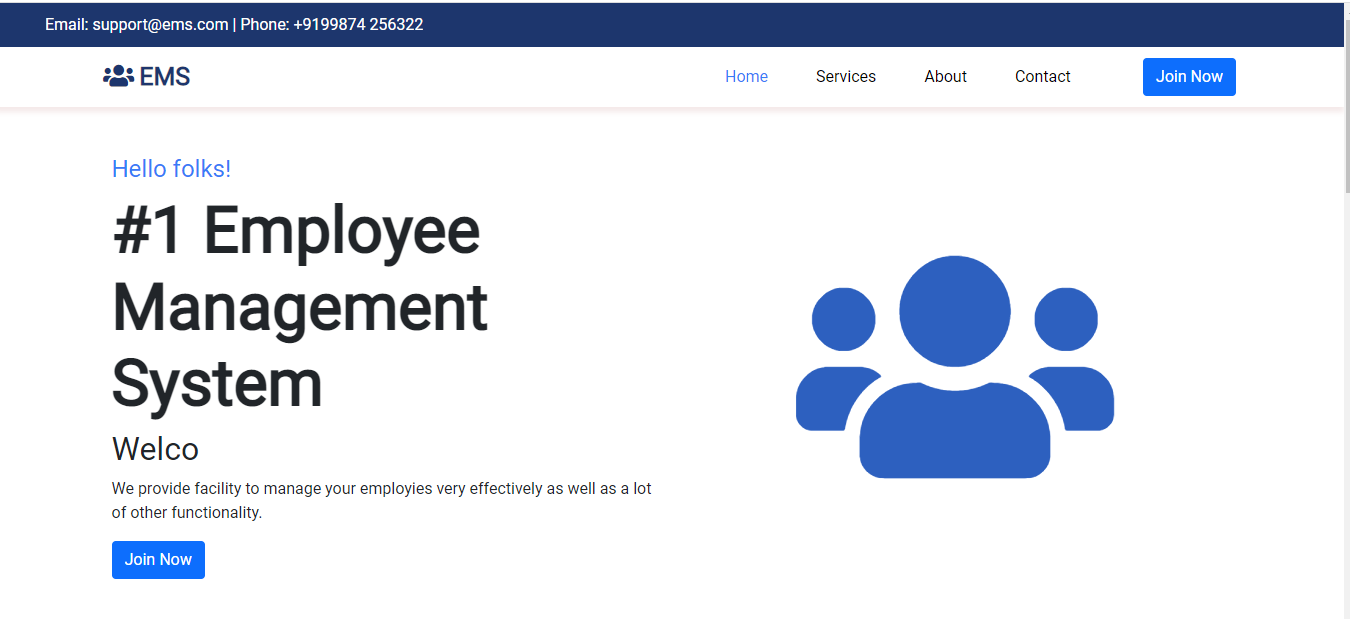
7. Eventtype

|  |  |  |  |
| --- | --- | --- | --- |
| **NAME** | **DATATYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **id** | ObjectID | PK | Eventtype Id |
| **Companyid** | ObjectID | FK | Reference companies |
| **Type** | String | Required | Event type holiday or meeting |
| **Backgroundcolor** | String | Required | Color |

8. Leaves

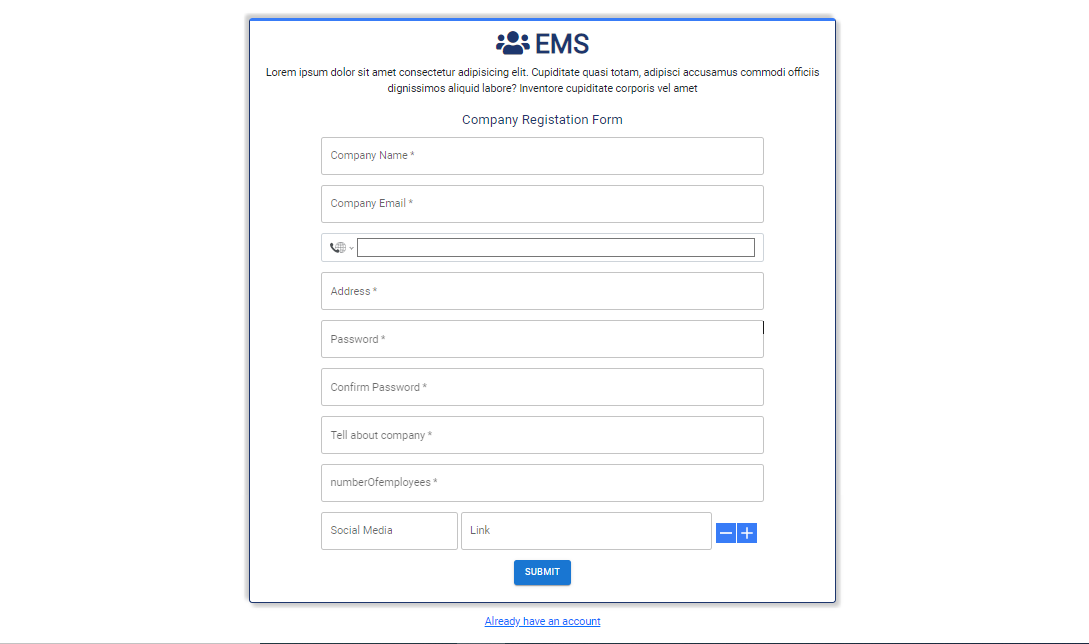
|  |  |  |  |
| --- | --- | --- | --- |
| **NAME** | **DATATYPE** | **CONSTRAINT** | **DESCRIPTION** |
| **id** | ObjectID | PK | Leave Id |
| **Companyid** | ObjectID | FK | Reference companies |
| **Employeeid** | ObjectID | FK | Reference employees |
| **Startdate** | Date | Required | Start date |
| **Enddate** | Date | Required | End date |
| **Halfday** | Number | Required | Number of half-day |
| **Fullday** | Number | Required | Number of full-day |
| **Reason** | String | Required | Reason for leave |
| **Status** | Boolean | Required | Leave status |

**3.5 User interface**



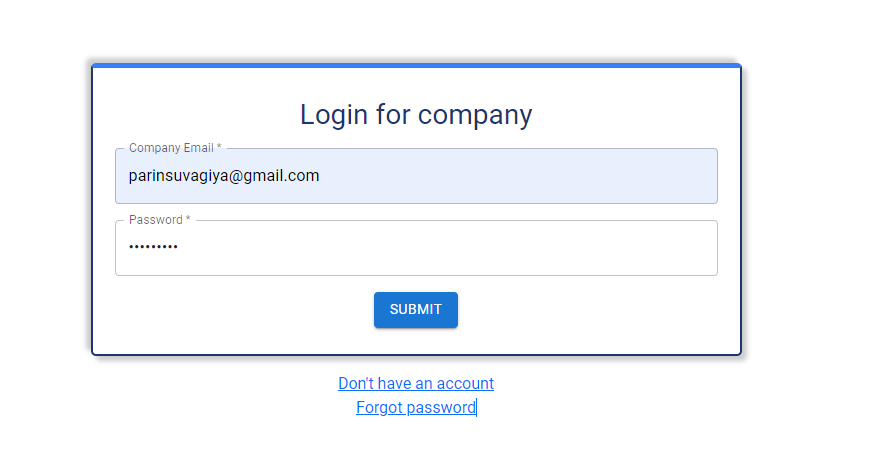
3.5.1 Home page

* This is the home page of ems.



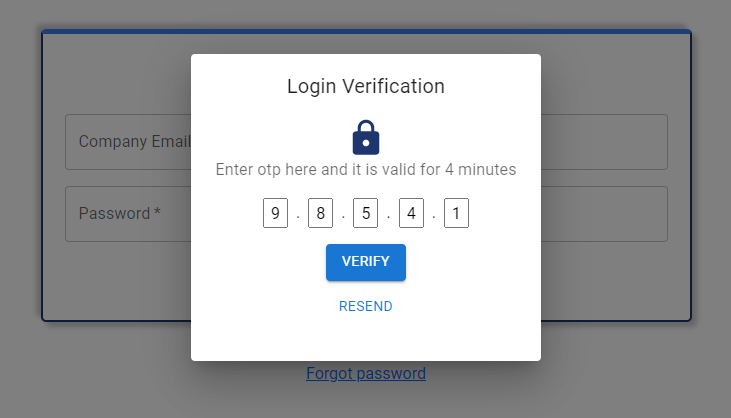
3.5.2 Company registration page

* Company registration form with validation feature



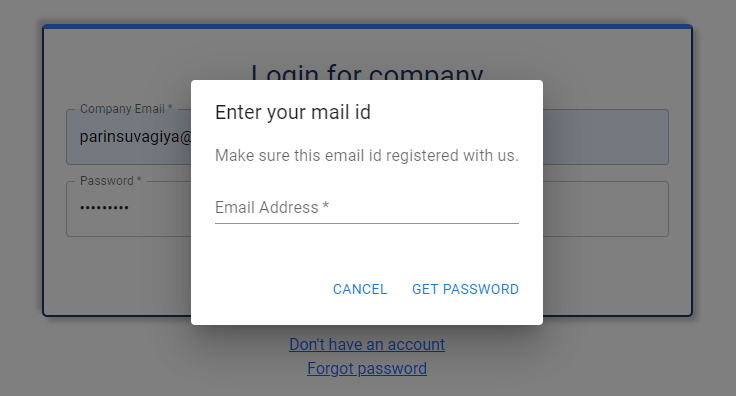
3.5.3 Company Login

* Company login form with validation and authentication



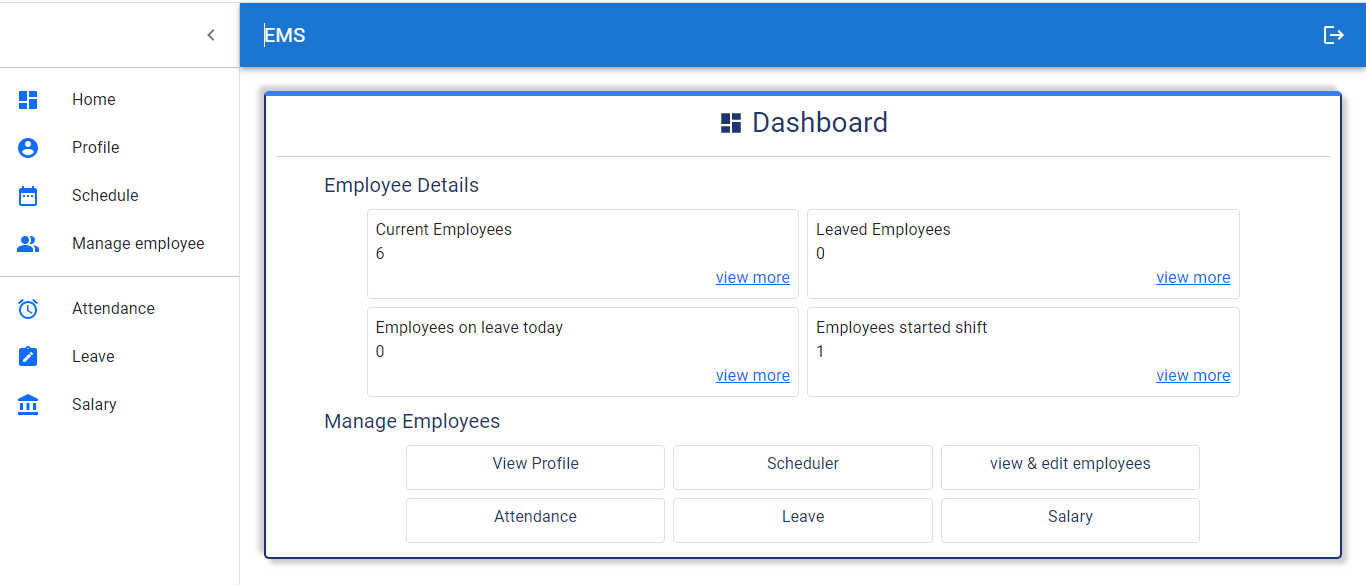
* + 1. verify OTP

- Verify OTP popup with resend OTP functionality

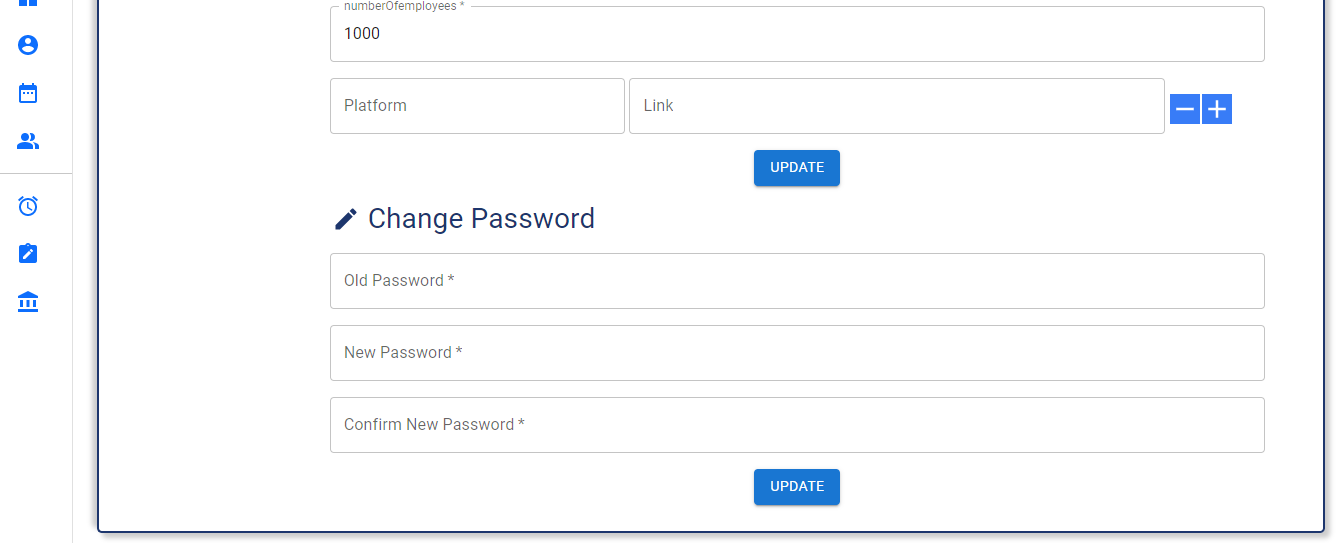
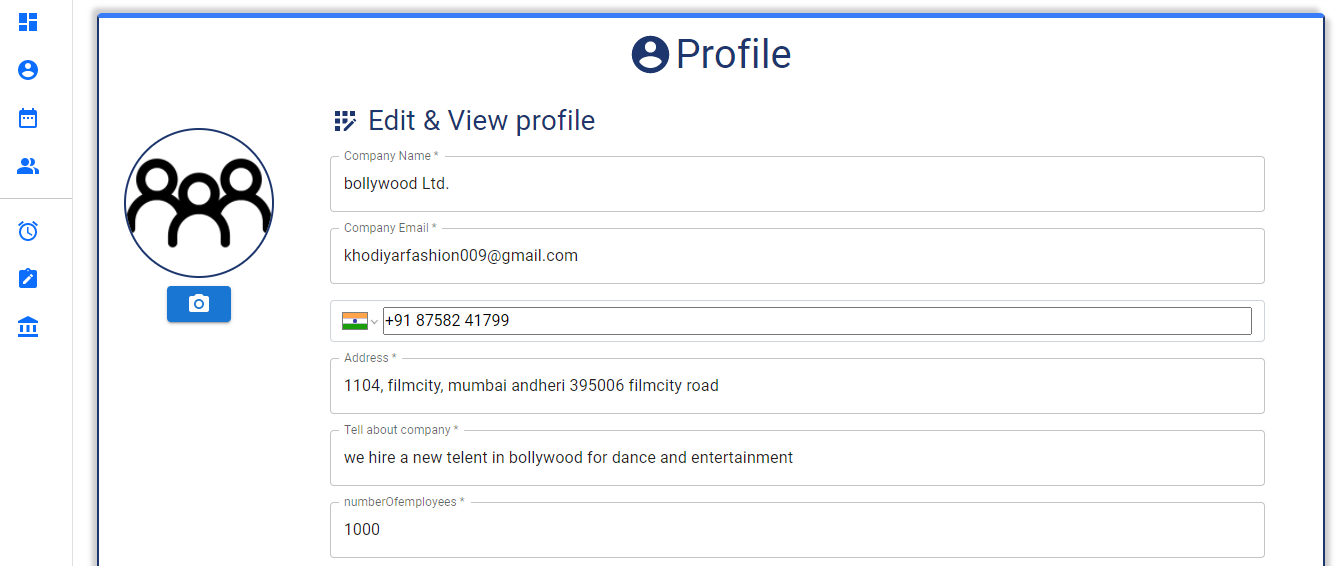


3.5.4 forgot password

* Forgot password page

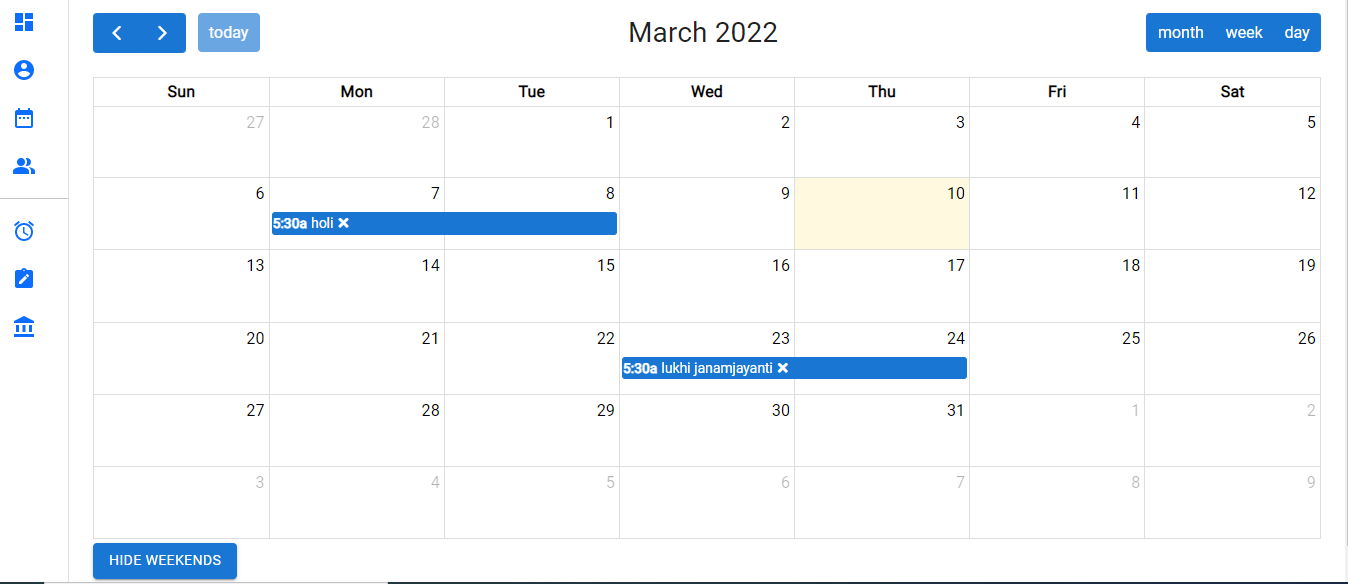


* + 1. company dashboard
* Company dashboard page it shows some statistics and redirects links.



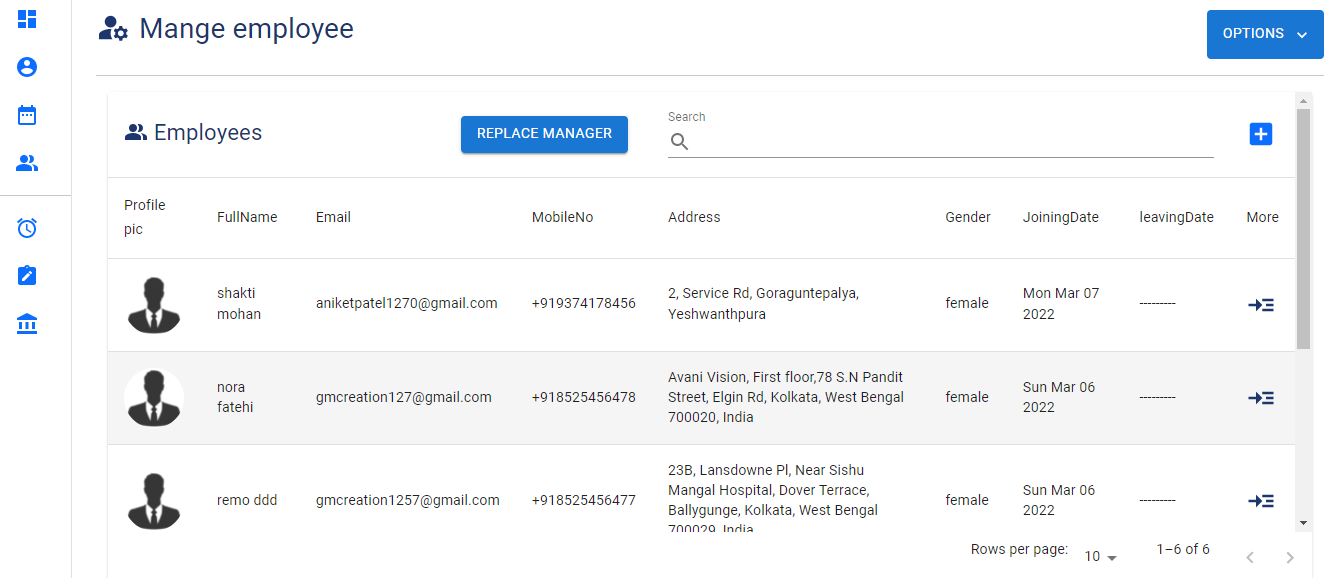
3.5.6 company profile & edit profile

* Company profile page view, update company profile



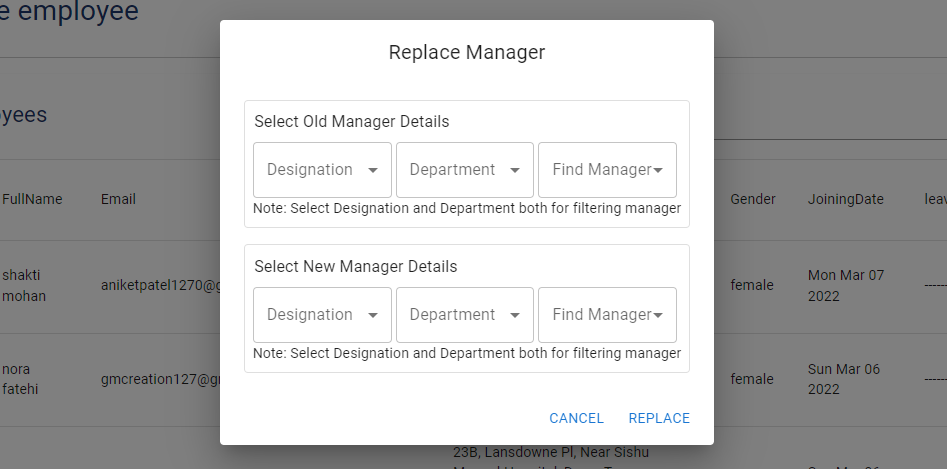
3.5.7 Scheduler

* Company scheduler add, update, delete event and set public events for employees



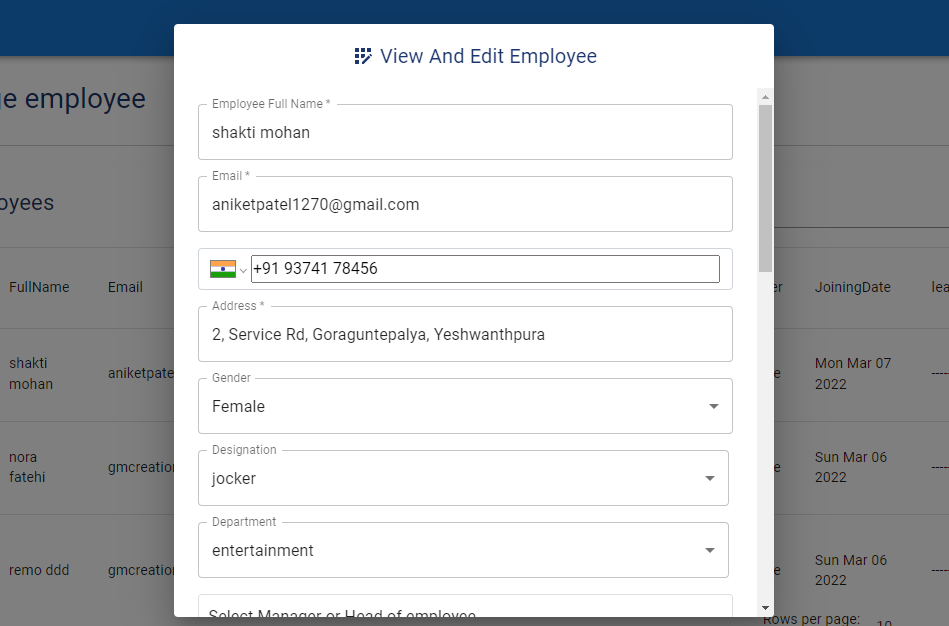
3.5.8 View employees

* View, edit employee profile



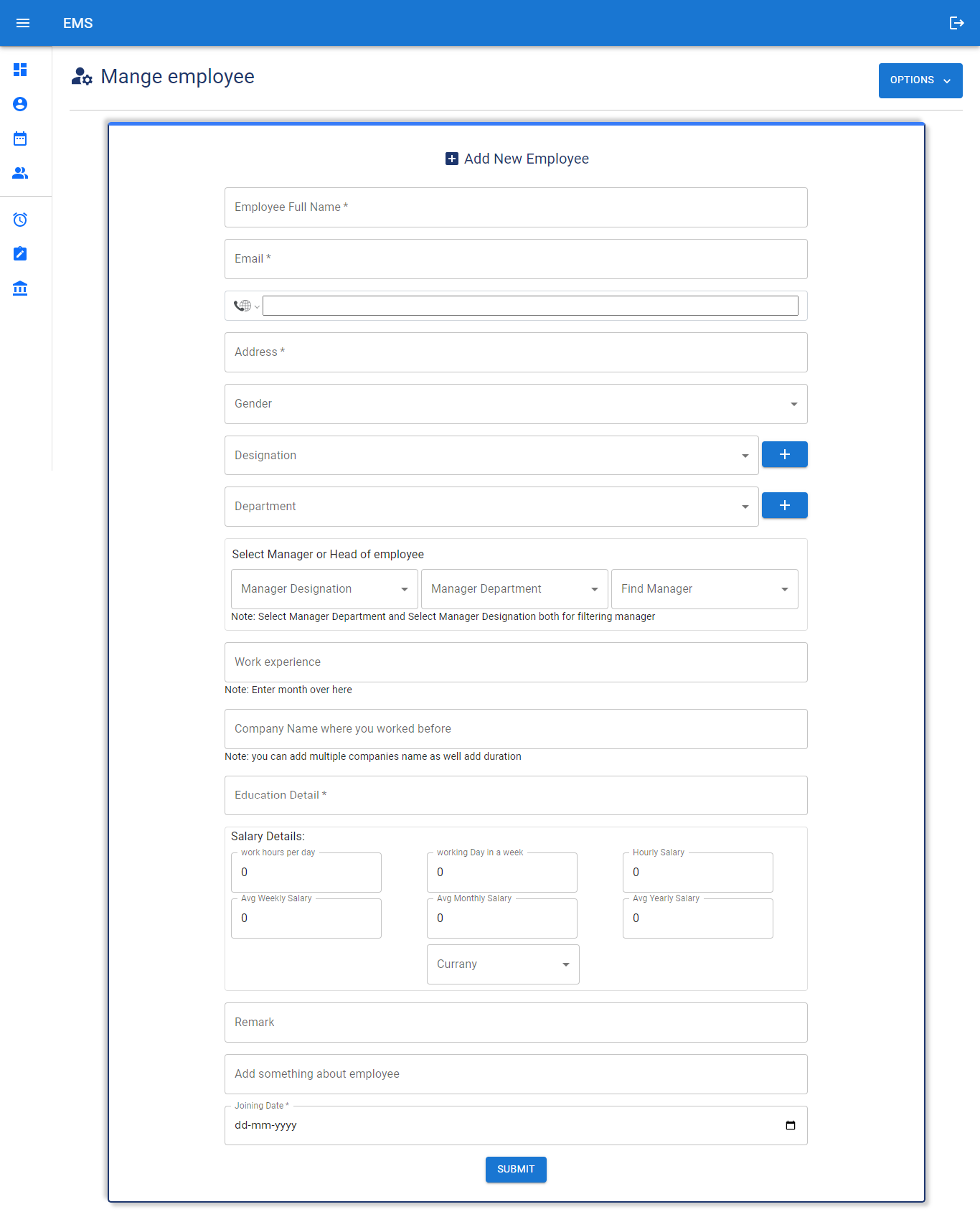
3.5.9 Replace manager

- Update manager of employees



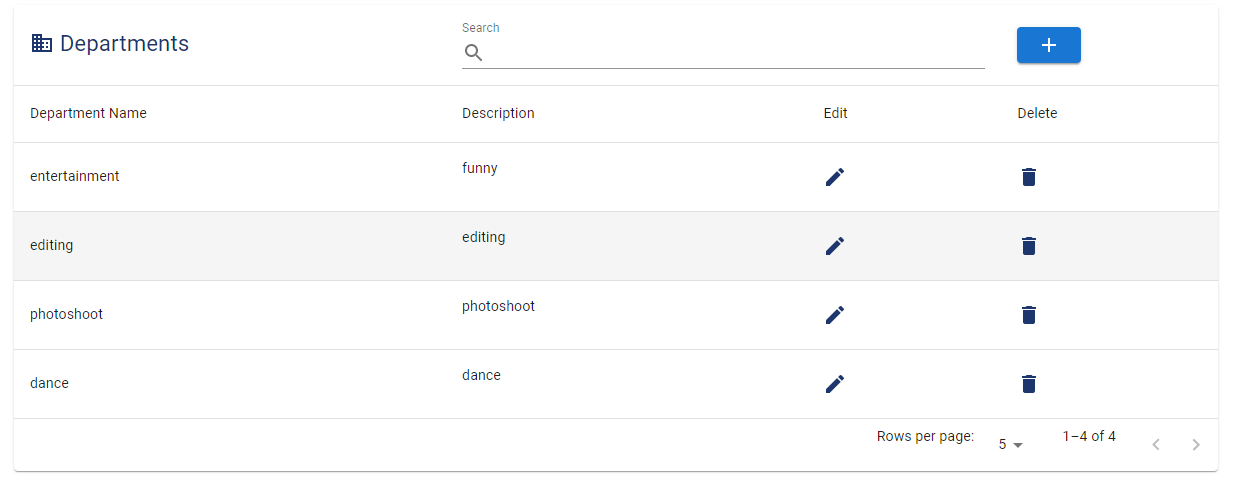
3.5.10 Edit employee

* Edit employee profile

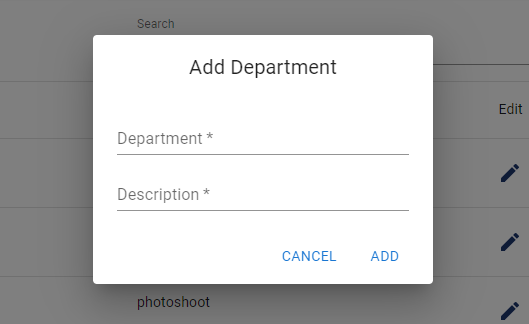


3.5.11 Add employee

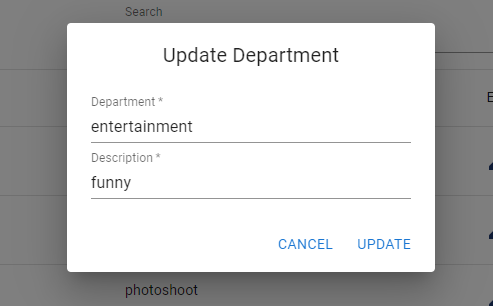
- Add new employee



3.5.12 view Departments

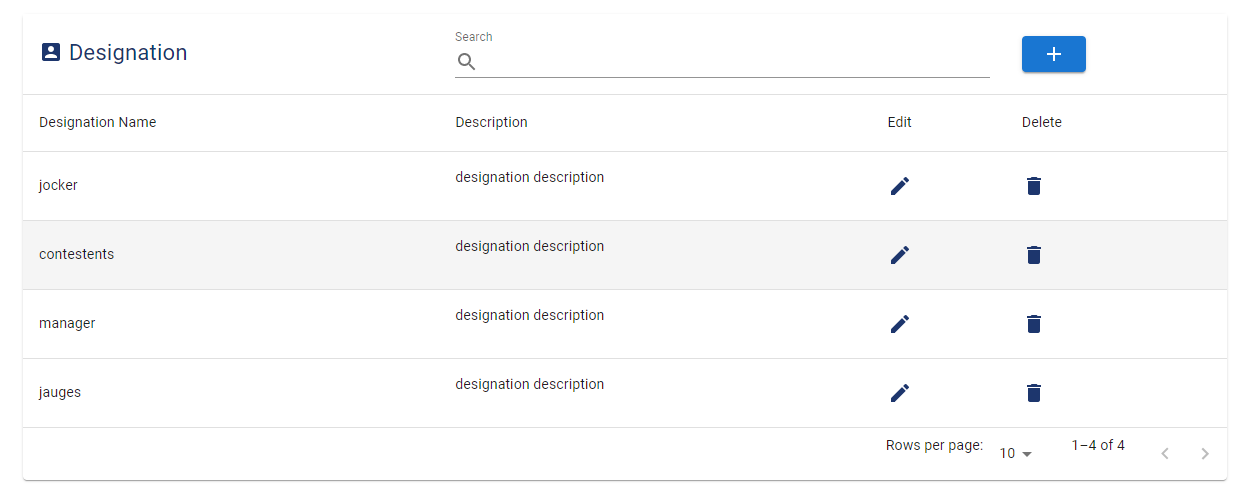


3.5.13 Add Departments

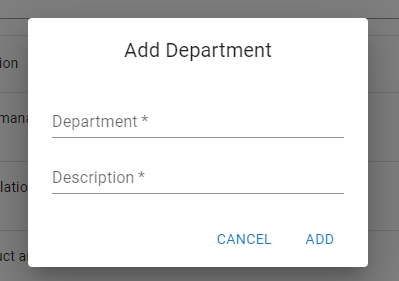


3.5.14 Update Department

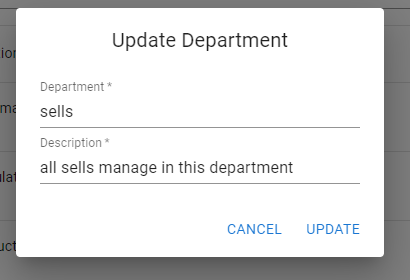
- Manage Designation functionalities like view, add, delete designation



3.5.15 view designations

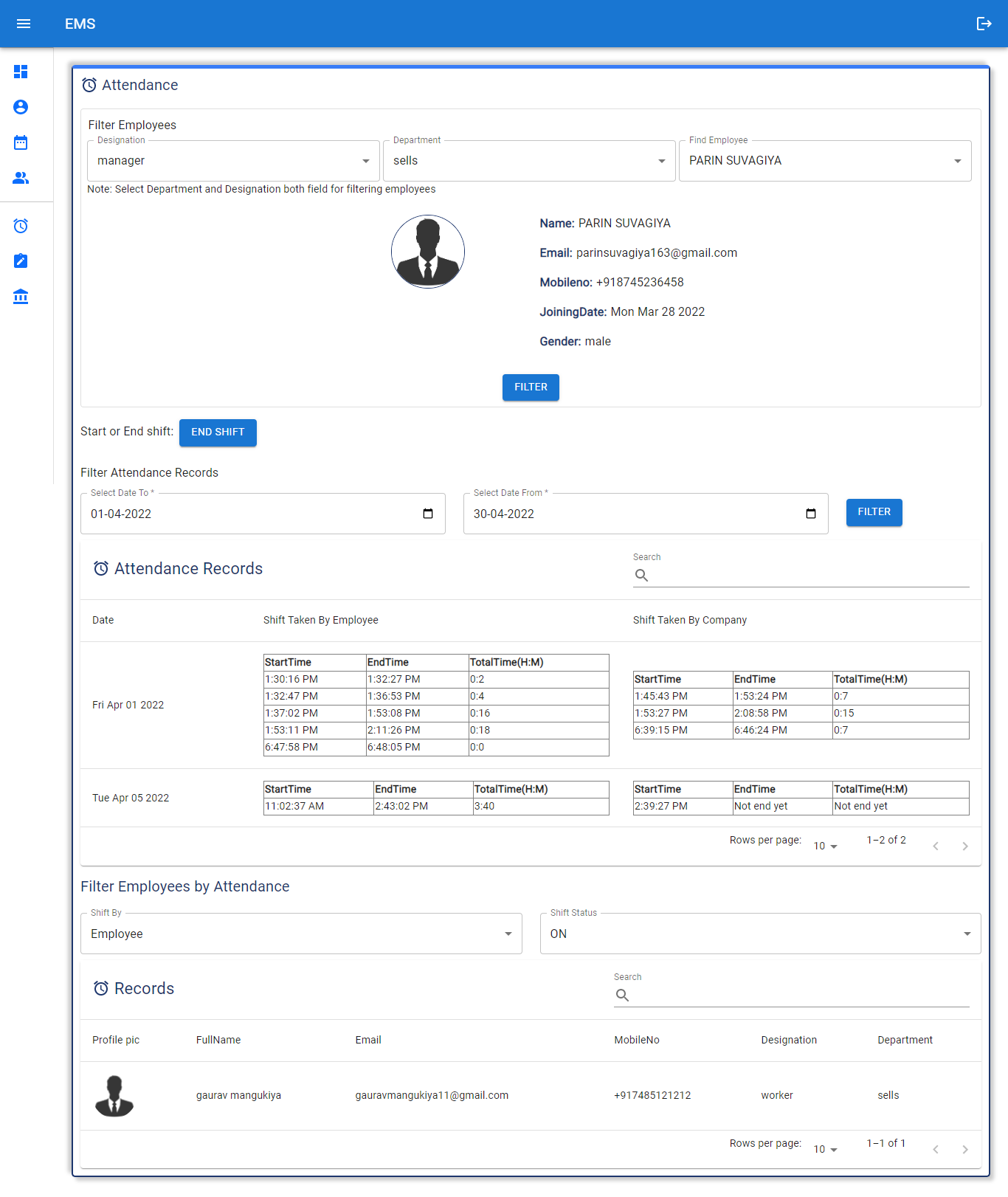


3.5.16 Add designations

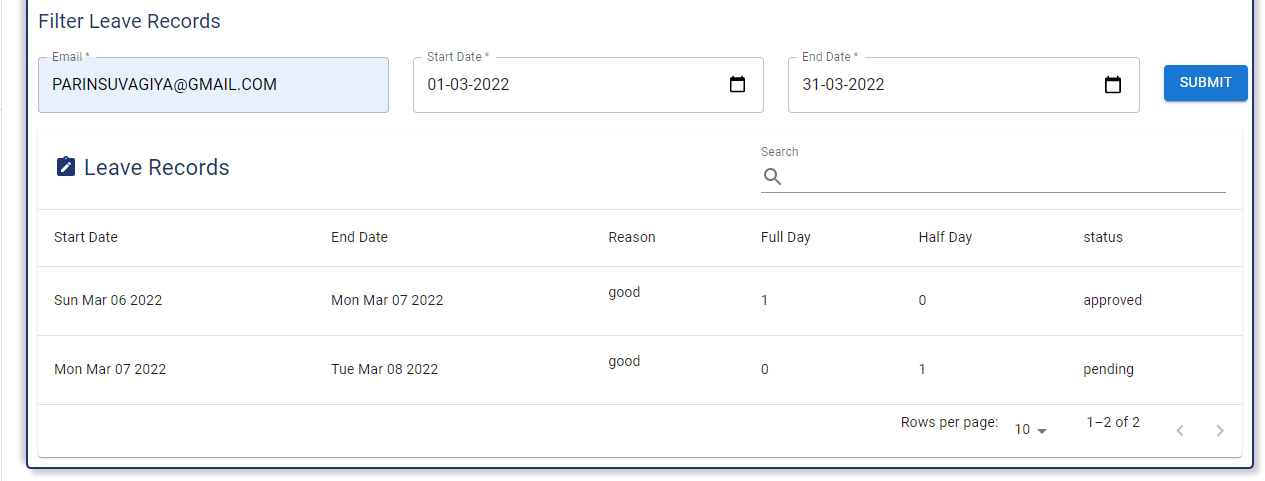
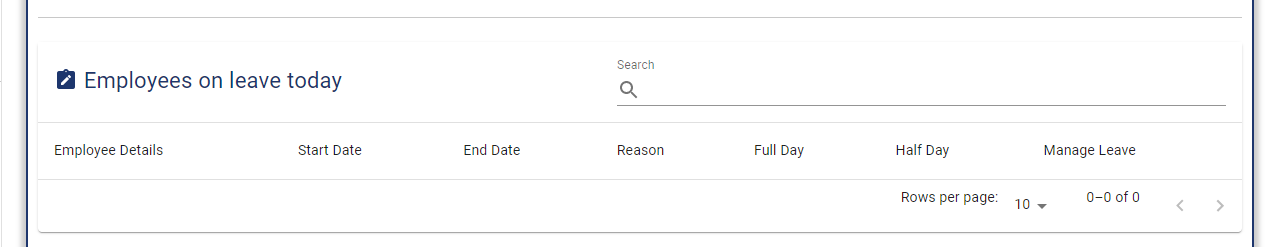
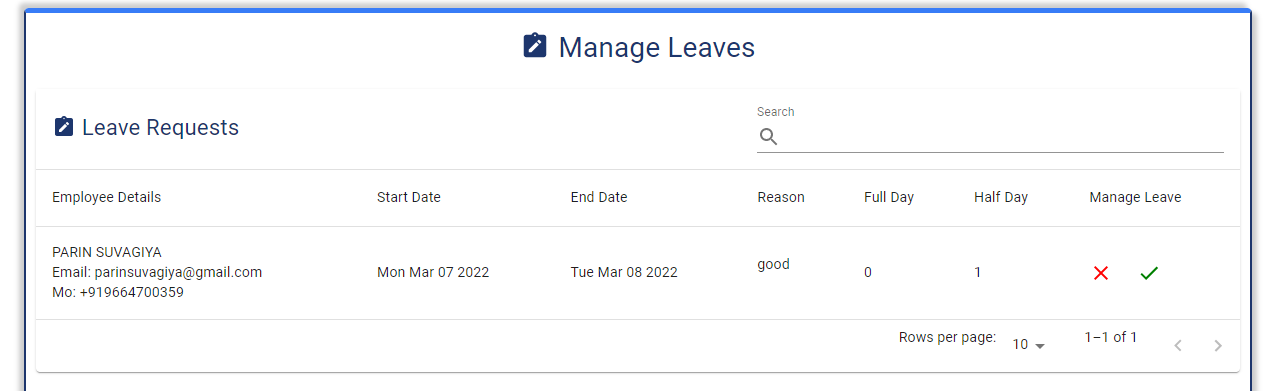


3.5.17 Update Designations

- Manage Departments functionalities like view, add, delete department

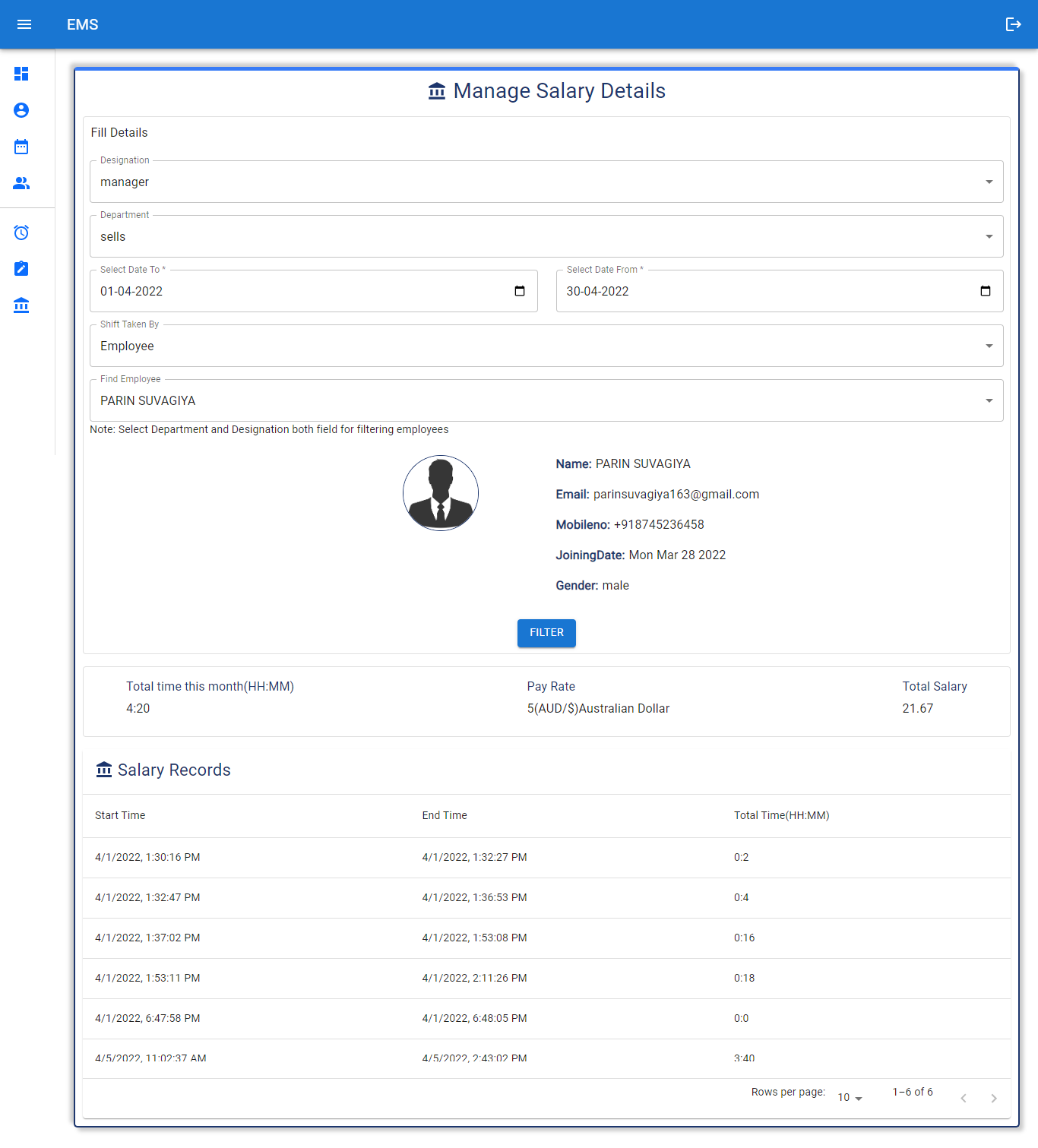
3.5.18 Manage attendance

* Manage attendance and view old records and view employees list who started shift

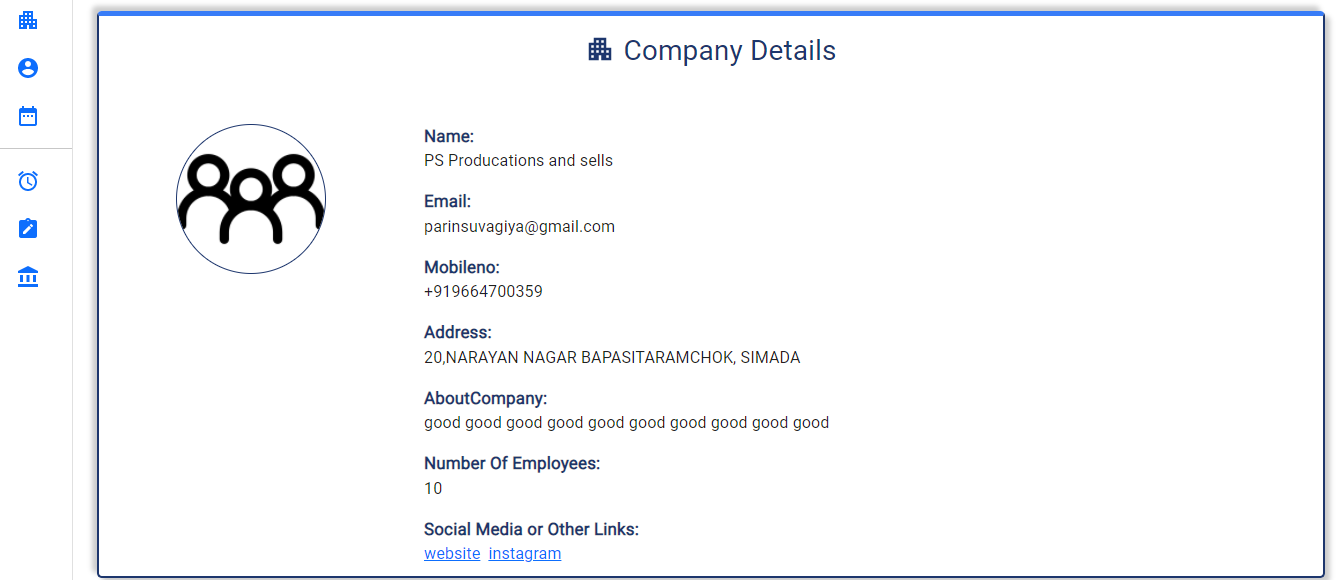


3.5.19 Manage leaves

* Manage employees leave and view their records

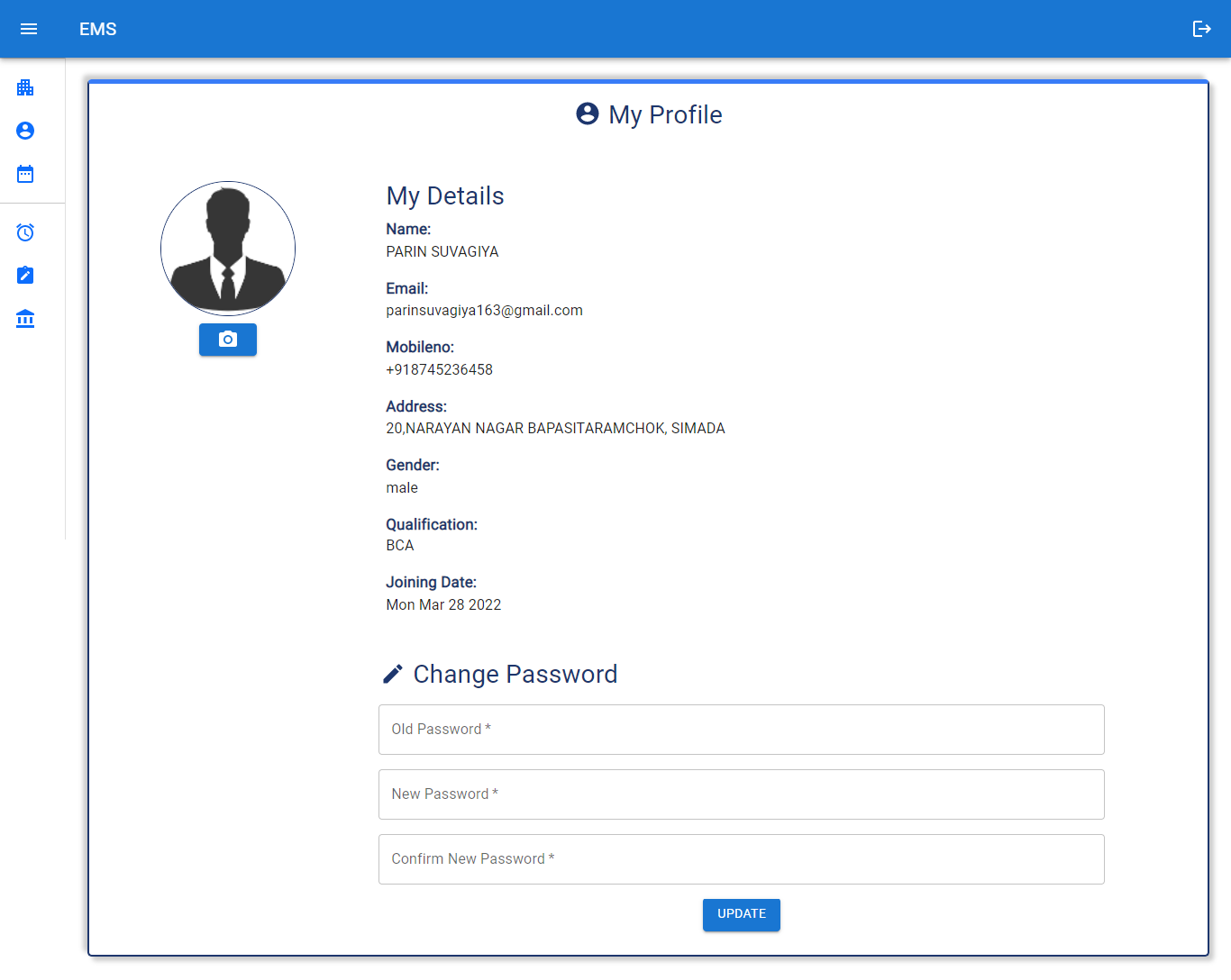
3.5.20 Manage salary

* Calculate salary between two dates

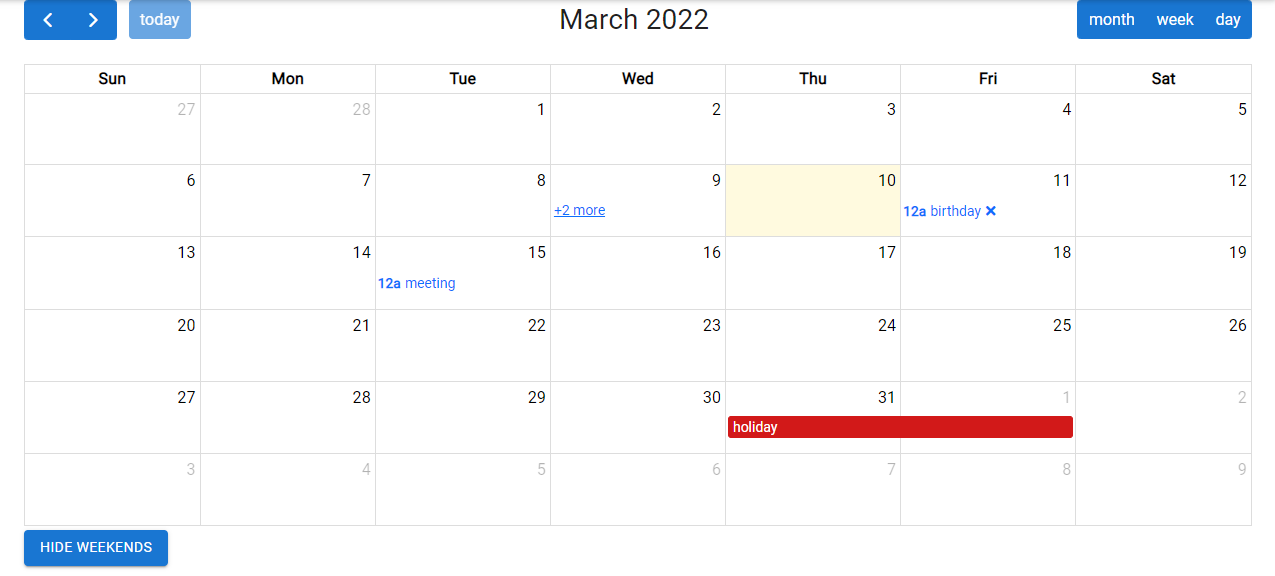


3.5.21 Employee view company profile

* View company profile

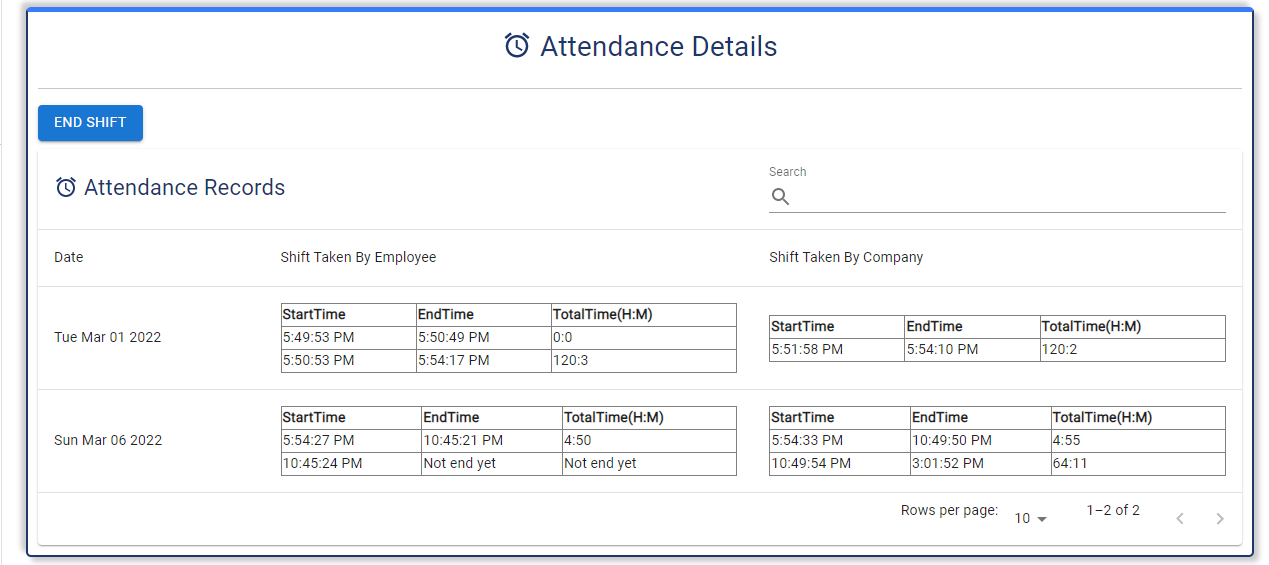
3.5.22 Employee views my profile

* View my profile and edit password



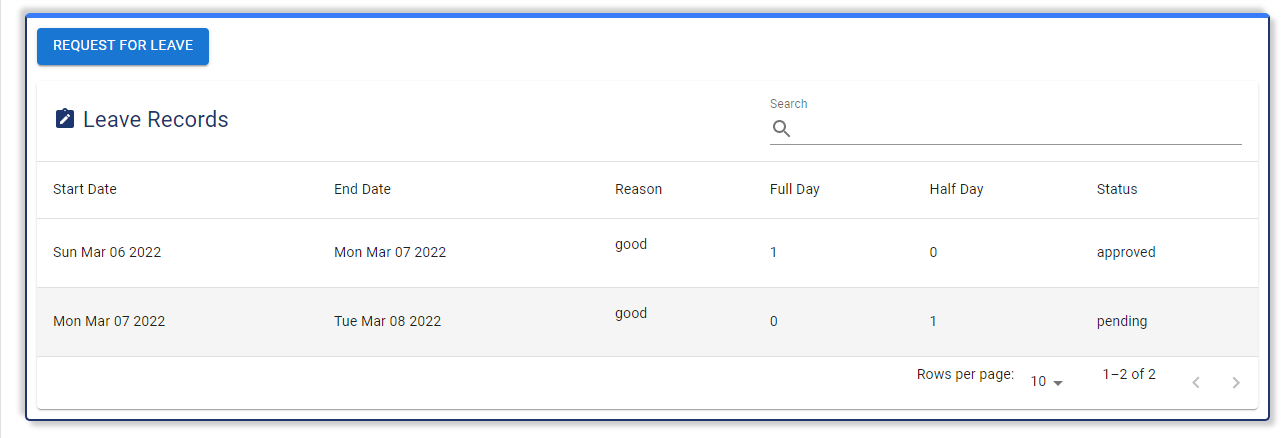
3.5.23 employee Scheduler

* View events and add personal and public events



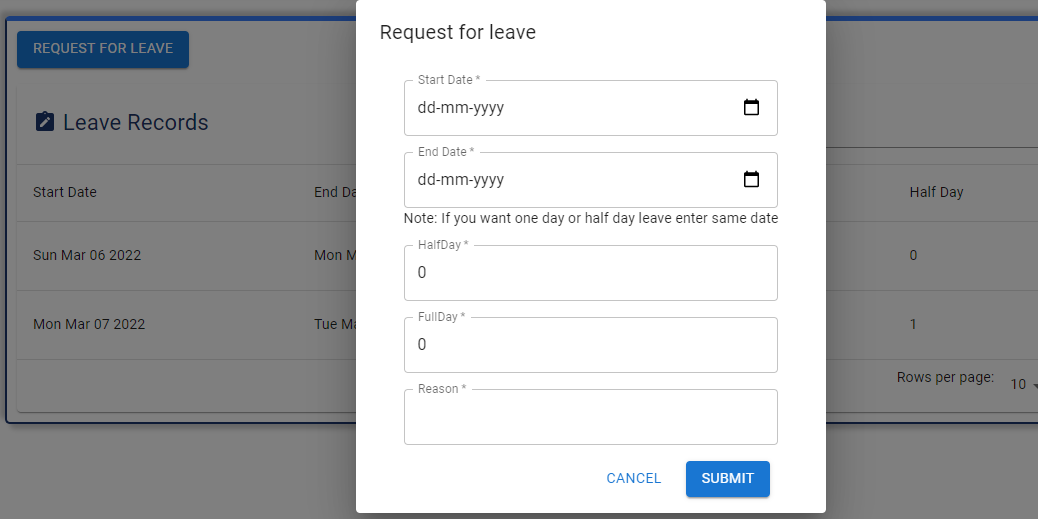
3.5.24 Attendance

- Track working hours & start and end shift



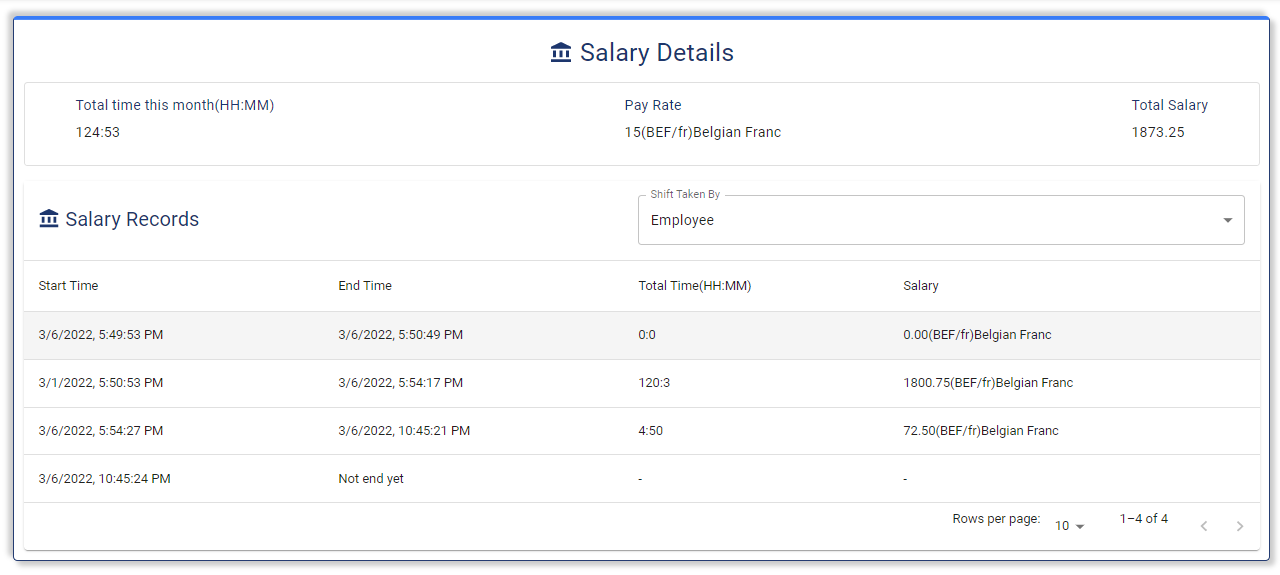
3.5.25 leave history

* View leave history



3.5.26 send leave request

* Request for leave and view old records



3.5.27 Salary

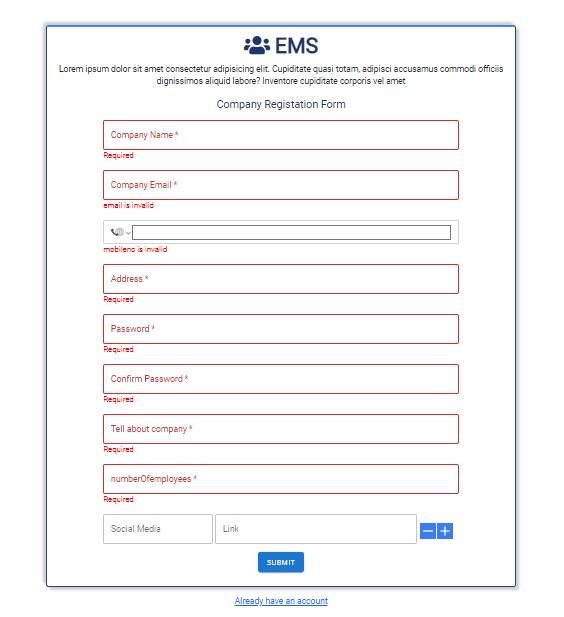
* View salary details daily basis

**4. Testing**

**4.1 Unit testing**

**Test case for company registration**

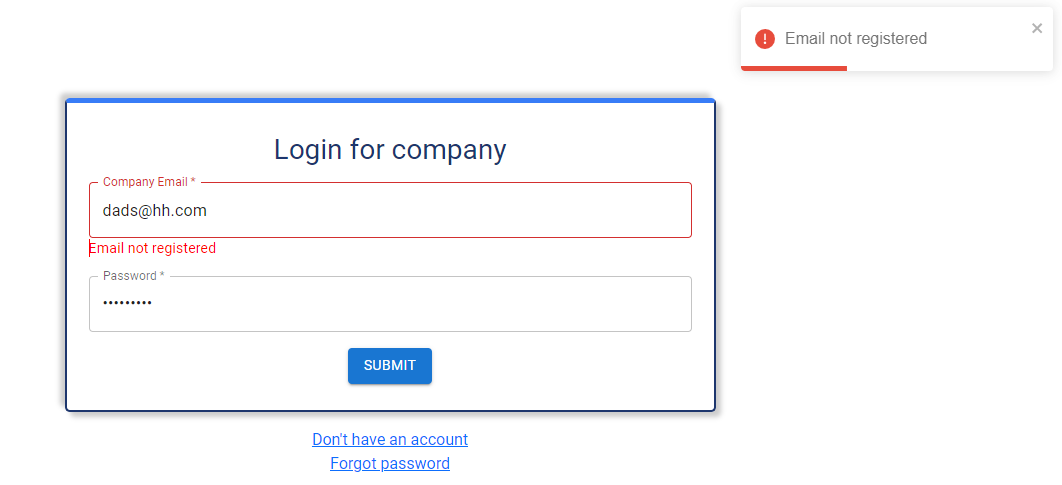
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test id** | **Test field** | **Stop execute** | **Executed result** | **Actual result** |
| 1 | Company name | Required | minimum 5 character required | Expected |
| 2 | Email | Valid email | email is invalid | Expected |
| 3 | Mobile no | Required | mobile no is invalid | Expected |
| 4 | Address | Required | Required,  minimum 30 character | Expected |
| 5 | password | Required | minimum 8 character required | Expected |
| 6 | Confirm password | Required | Does not match with password | Expected |
| 7 | About the company | Required | minimum 50 character required | Expected |
| 8 | Number of employees | Not less than 1 | must have valid digits | Expected |
| 9 | Social media link | Empty | No error | Expected |



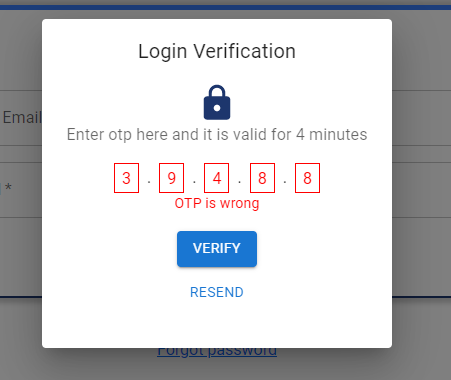
4.1.1 company registration error view

**Test case for the company and employee login**

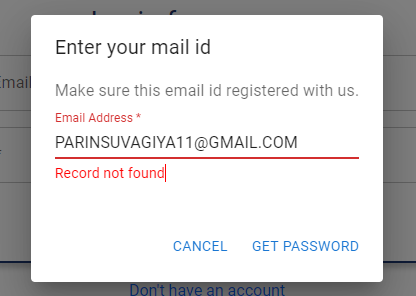
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test id** | **Test field** | **Stop execute** | **Executed result** | **Actual result** |
| 1 | Email | Valid email | email is invalid,  Email not registered | Expected |
| 2 | Password | Required | minimum 8 character | Expected |

****

4.1.2 company login error view

****

4.1.3 OTP error view

****

4.1.4 forgot password error view

**Test case for add employee**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test id** | **Test field** | **Stop execute** | **Executed result** | **Actual result** |
| 1 | Name | Required | minimum 5 characters required | Expected |
| 2 | Mobile no | Required | mobile no is invalid | Expected |
| 3 | Email | Required | email is invalid | Expected |
| 4 | Address | Required | Required | Expected |
| 5 | Gender | Required | Required | Expected |
| 6 | Designation | Required | Required | Expected |
| 4 | departments | Required | Required | Expected |
| 8 | Manager | Required | Required | Expected |
| 9 | Workexp | Not required |  | Expected |
| 10 | Company where worked | Not required |  | Expected |
| 11 | Education details | Required | Required | Expected |
| 12 | Salary | Required | Required | Expected |
| 13 | Remark | Not required |  | Expected |
| 14 | About employees | Not required |  | Expected |
| 15 | Joining date | Required | Required | Expected |

**Test case for designation**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test id** | **Test field** | **Stop execute** | **Executed result** | **Actual result** |
| 1 | Designation name | Required | Required | Expected |
| 2 | Description | Required | Required | Expected |

**Test case for the department**

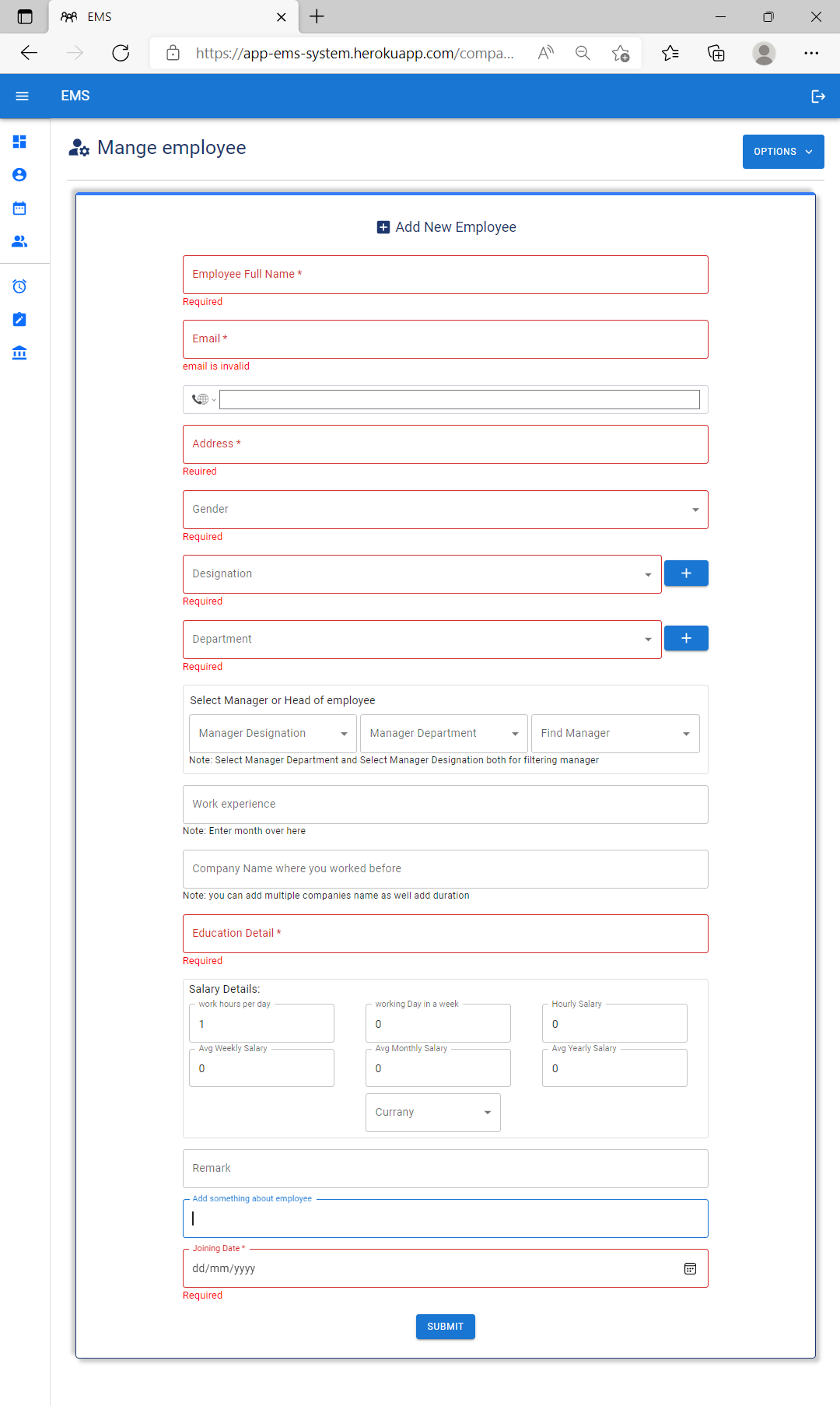
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test id** | **Test field** | **Stop execute** | **Executed result** | **Actual result** |
| 1 | Department name | Required | Required | Expected |
| 2 | Description | Required | Required | Expected |

**Test case for events**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test id** | **Test field** | **Stop execute** | **Executed result** | **Actual result** |
| **1** | **Tital** | **Required** | Required | Expected |
| **2** | **Strat date** | **Required** | Required | Expected |
| **3** | **End date** | **Required** | Required | Expected |
| **4** | **All day** |  |  | Expected |
| **5** | **Category** | **Required** | Required | Expected |
| **6** | **Type** | **Required** | Required | Expected |
| **7** | **Link** | **Not required** | **No error** | **Expected** |

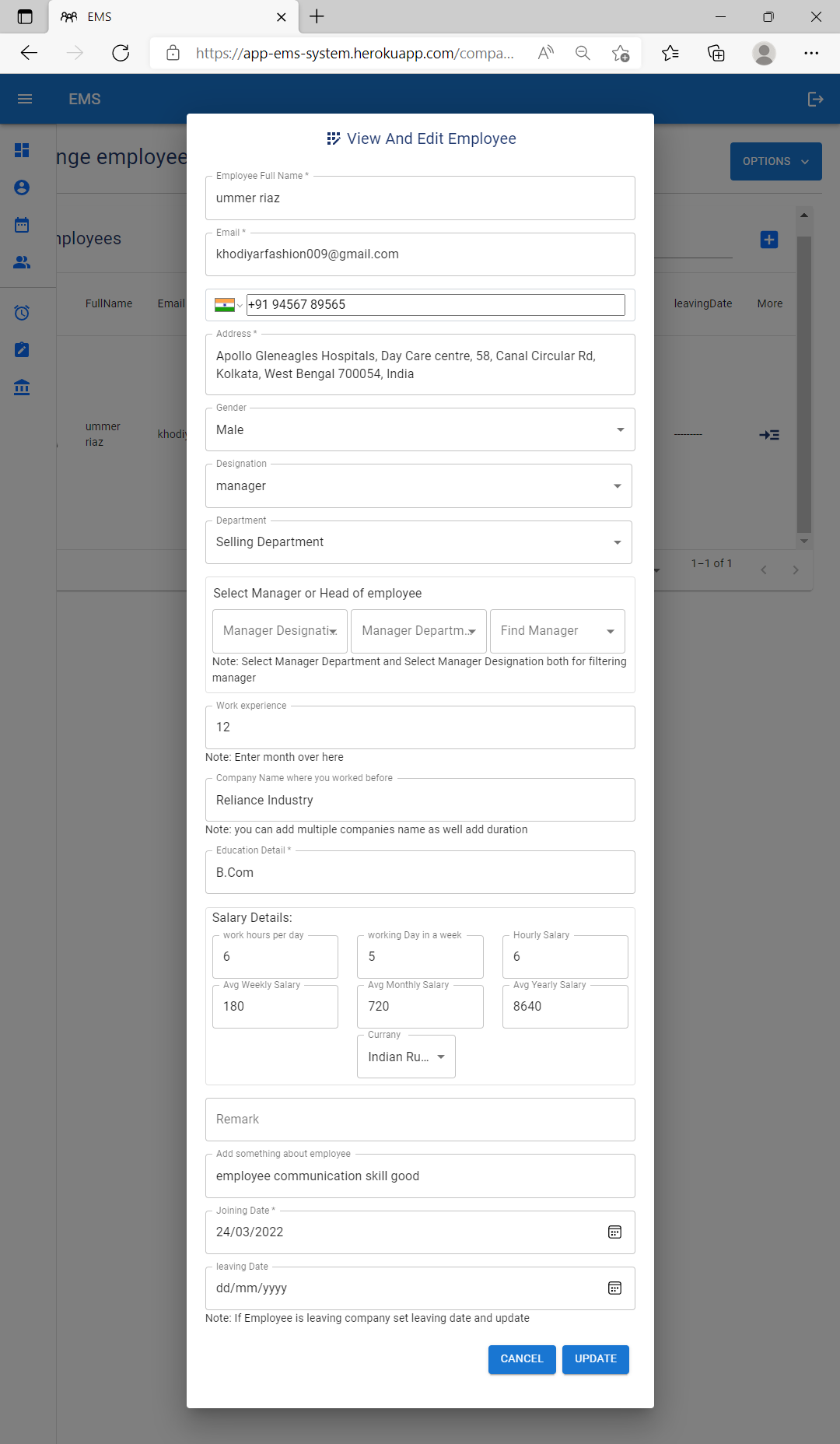
* We have performed other test cases and found expected results.

Add employees



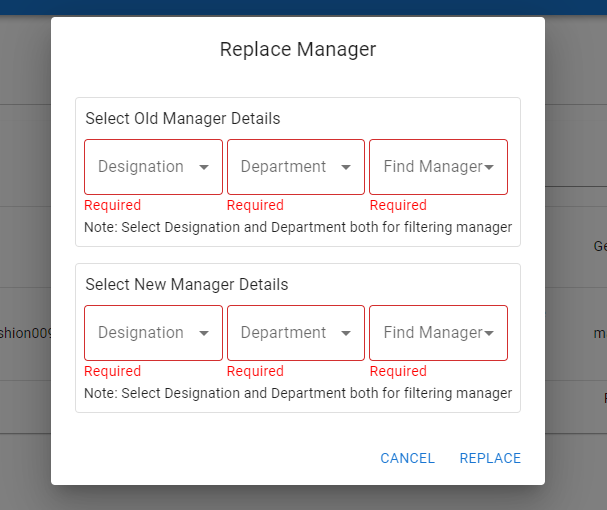
4.1.5 Add employee error view

Edit employees



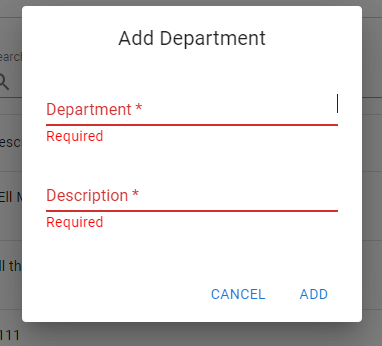
4.1.6 update employee view

Replace Manager



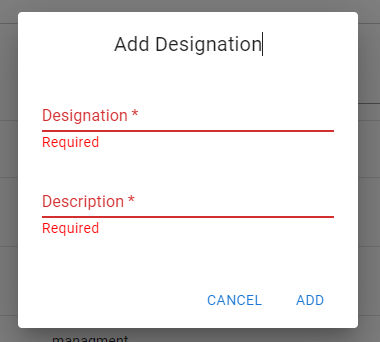
4.1.7 replace manager error view

Departments



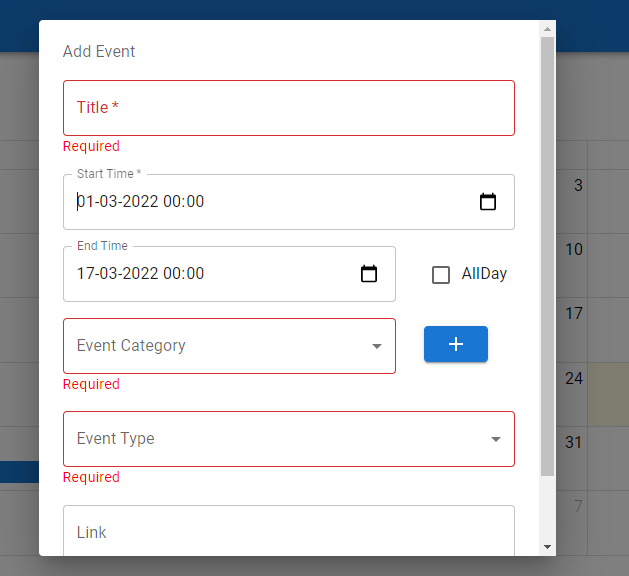
4.1.8 add department error view

Designation



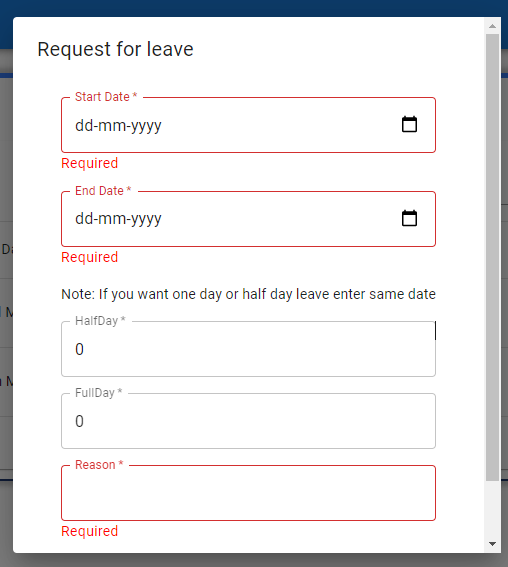
4.1.9 add designation error view

Event



4.1.10 add event error view

Request for leave



4.1.11 leave request error view

* Shift Manager, Salary counter, leave manager all modules are working as we expected.

**4.2 Navigation Testing**

* All navigation links are redirecting users to the perfect page and protected routes are not allowing an unauthorized user to view the page.

**4.3 Functional Testing**

* Company login & registration are working fine with validation and authentication
* Employee login working fine
* Manage employees working fine
* Department manager working as we expected
* Designation manager working as we expected
* Shift manager is working fine
* Salary manager is working fine
* Event module is also working fine
* Leave manager is working file
* Automatic mail sending functionality is working

**4.4 Environment Testing**

Firefox browser, Internet Explorer and chrome consider testing for environment operability of software.

Web server – created using nodejs

Database – MongoDB

OS – Windows

Browser – Firefox/Internet Explorer/Chrome

Postman is used for API environment testing

**5.** **Conclusion**

The purpose of this website is to provide functionality for managing employees. EMS is a web application where multiple companies can create their account and get access to a lot of functionality for managing employees. EMS has API which separates the frontend and backend and also helps to integrate with other websites. EMS provides functionalities like scheduling important meetings, and events and managing employees' leave, salary, attendance, and working hours with effective sorting and filtering techniques. This system provides a portal for employees to interact with the company and track their daily data like attendance, leave, salary, and working hours. EMS frontend is built using Reactjs, which is a javascript library. EMS Backend provides an Application programming interface (API) that is built using Nodejs, Expressjs, and MongoDB. When user requests for EMS frontend will load on the web browser window and API will provide data on user needs.

[code and some project guide](https://github.com/parin-2002/EMS)

**6.** **Bibliography**

**Websites**

<https://reactjs.org/docs/getting-started.html>

<https://nodejs.org/en/docs/>

<https://mongoosejs.com/docs/api.html>

<https://mui.com/>

<https://getbootstrap.com/docs/5.0/getting-started/introduction/>