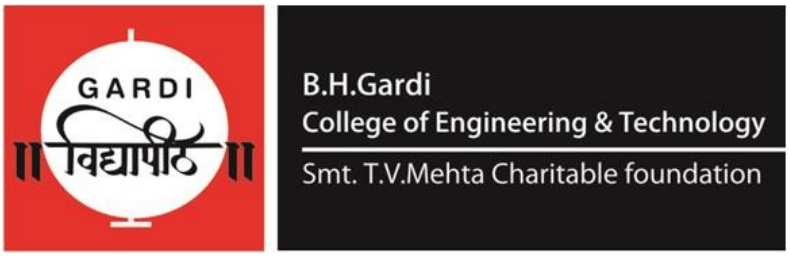
Employee Leave Management System

: Submitted By :

Enrollment No	Name
215030694008	Vadodaria Manali

OF



: Submitted To :

Gujarat Technological University (GTU)

As a partial fulfillment of the degree of
Master of Computer Applications (MCA)

April – 2023

Table of Contents

INTRODUCTION

a. Existing System

- The existing Leave Management system manifests a lot of inefficiency and impotence.
- The system takes a lot of time to compose the employee's leave and the confirmation of leave approval.
- It also delays the leave system. Due to various unforeseen occasions, the payroll is not generated within the leave period thus creating a lot of hassle
- Error is the other major problem with the manual system and even with a repeated cross check some of the other errors surely persist which can cause a lot of problems.
- The existing payroll system manifests a lot of inefficiency and impotence. The system takes a lot of time to compose the employee's leave and the approval or decline of leave.
- It also delays the leave management system. Due to various unforeseen occasions, the leave is not generated within the leave period thus creating a lot of hassle.
- Error is the other major problem with the manual system and even with a repeated cross check some of the other errors surely persist which can cause a lot of problems.
- The existing leave system manifests a lot of inefficiency and impotence. The system takes a lot of time to compose the employee's leave approval
- It also delays the salary decline system. Due to various unforeseen occasions, the leave does not get to afford by the company.
- Error is the other major problem with the manual system and even with a repeated cross check some of the other errors surely
- persist which can cause a lot of problems

A Third-party payroll system is a leave management system that is provided by an external vendor or service provider, rather than being developed in-house by the organization. These systems are often web-based and offer a range of features for managing employee leave, including calculating pending, approval, and decline leave. Third-party leave systems are typically accessed through a software-as-a-service (SaaS) model, where the organization pays a subscription fee to use the system. This model allows organizations to avoid the costs and complexities of developing and maintaining their payroll system

in-house, as well as reducing the need for dedicated IT staff to manage the system.

There are some drawbacks to using a third-party leave system:

- Limited customization: Third-party leave systems may not offer the same level of customization as an in-house system, which can limit the ability to tailor the system to the organization's specific needs.
- Security concerns: Third-party leave systems may raise security concerns, as sensitive employee data is being stored and managed by an external vendor.
- Integration challenges: Third-party leave systems may not integrate seamlessly with other systems used by the organization, such as HR systems or accounting software.

So, third-party payroll systems can be a viable option for organizations looking to streamline their leave management processes, but it is important to carefully leave the features and limitations of any system before making a decision. So we required the new leave system where the company can manage data and system on their own.

b. Need for the new System

- Scalability: As your Vue.js project grows in size and complexity, you may find that your current system is no longer able to handle the increased demands. In this case, you may need to implement a new system that can scale to meet your needs.

- Performance: If your Vue.js project is experiencing performance issues, a new system may be necessary to improve performance. This could involve optimizing your code or implementing a new caching strategy.

- Security: If your Vue.js project is handling sensitive data, you may need to implement a new security system to ensure that the data is properly protected.
- Maintenance: If your Vue.js project is becoming difficult to maintain, a new system may be necessary to simplify the codebase and make it easier to manage.

- New Features: If you want to add new features to your Vue.js project that are not supported by your current system, a new system may be necessary to implement those features.

c. The objective for the new System

- Store Object: In Vue.js, the store object is used to manage the state of your application. It contains the data and logic needed to manage the application's state and keep it in sync with the user interface. You might create a new store object for your system to manage the state of your new features.

- Component Object: A Vue.js component is a self-contained unit of code that defines a custom element or UI widget. You might create a new component object for your system to encapsulate the behavior and appearance of a specific UI element.

- Service Object: A service object in Vue.js is used to abstract the communication with an external API or backend system. You might create a new service object for your system to handle the communication with a specific API endpoint or backend system.

- Middleware Object: A middleware object in Vue.js is used to intercept and modify incoming or outgoing requests or responses. You might create a new middleware object for your system to add custom behavior or validation to incoming or outgoing requests.

- Directive Object: A Vue.js directive is used to add custom behavior or functionality to a specific element in the user interface. You might create a new directive object for your system to add custom behavior to a specific element or group of elements in the user interface.

  These are just a few examples of objects that you might create for a new system in Vue.js. The specific objects you need will depend on the requirements and architecture of your system.

d. Problem Definition

Defining the problem in Vue.js is an important step in developing any application. It involves identifying the issues or challenges that your application needs to address and specifying the requirements and goals that it needs to achieve.

Here are some steps you can take to define the problem in Vue.js:

- Identify the problem: Start by identifying the problem that your application needs to solve. This could be a business problem, a technical challenge, or a user need that you are trying to address.

- Gather requirements: Once you have identified the problem, gather requirements from stakeholders, users, and other sources to define what your application needs to achieve. This could include

functionality, performance, usability, and other factors.

- Define goals: Based on the requirements, define the goals that your application needs to achieve. This could include specific business outcomes, user engagement metrics, or technical performance benchmarks.

- Analyze the problem domain: Analyze the problem domain to identify any constraints, dependencies, or other factors that could impact your solution. This could include technical limitations, regulatory requirements, or user preferences.

- Develop a plan: Based on the problem definition and analysis, develop a plan for how you will build your application. This should include a high-level architecture, a list of key features and functionality, and a timeline and budget for development.

By following these steps, you can define the problem in Vue.js and set clear goals and requirements for your application. This will help ensure that your development efforts are focused on addressing the most important challenges and delivering the best possible solution for your users.

e. Core Component

Vue.js has several core components that are built-in and provide essential functionality for building web applications. Here are some of the most commonly used core components in Vue.js:

- Vue Instance: The Vue instance is the core building block of a Vue.js application. It represents the root of the application and is responsible for creating and managing all the other components in the application.

- Template: The template is the HTML-like syntax used to define the structure and layout of the user interface in a Vue.js application. It includes directives and bindings that allow for dynamic data rendering and user interaction.

- Component: A component is a self-contained unit of code that defines a custom element or UI widget in a Vue.js application. Components are reusable and can be composed together to create complex user interfaces.

- Directives: Directives are special attributes that provide additional functionality to HTML elements in a Vue.js application. They allow you to manipulate the DOM, add or remove elements, and bind data to user interface elements.

- Filters: Filters are used to transform data before it is displayed in the user interface. They allow you to format dates, numbers, and other data types in a variety of ways.

- Mixins: Mixins are a way to share functionality between multiple components in a Vue.js application. They allow you to encapsulate reusable code and apply it to multiple components, improving code reusability and maintainability.

- Plugins: Plugins are external packages that provide additional functionality to a Vue.js application. They can be used to add third-party libraries, integrate with other frameworks, or add custom functionality to the application.

These core components are essential building blocks for creating Vue.js applications, and understanding how to use them effectively is key to building robust, scalable, and maintainable applications.

f. Project Profile

Project Name:	Employee Leave Management System
Type of Application:	Roll base System
Project Description:	The Employee Leave Management System is a software application that helps organizations manage and automate their employee payroll processes. This system helps organizations maintain accurate and up-to-date employee payroll records, calculate leave and conform and decline the leave, and also remark on that.
Team Size:	1
Front End:	HTML, CSS, JavaScript, BootStrap, Vue.js
Back End:	Node js
Database:	Firebase
Tools used:	Microsoft Visual Studio Code

❖ Company Profile

Company Name:	Alian Software
Address:	S1, Shreeji Arcade, Indira Circle, opp. Shasvat Hospital, Anand, Gujarat 388001
Work in Technology:	NodeJS, VueJS, React JS, React Native, Shopify, PHP, Nuxt Js, Next JS, Python, Machine Learning and WordPress
Mentor at Company:	Mr. Shyam Vadaliya (Team Leader of Back End)

g. Assumption and Constraints

Assumptions and constraints are important considerations when building any project, including a leave management system in Vue.js. Here are some examples of assumptions and constraints that you might encounter when developing a leave management system in Vue.js:

- Assumptions:

  - Employees will be able to submit leave requests through the system.
  - Managers will be able to approve or deny leave requests submitted by employees.
  - The system will maintain a record of all leave requests and their statuses.
  - The system will allow for the creation and management of leave policies, including accrual rates, carryover limits, and other policy details.
  - The system will be able to integrate with other HR systems, such as payroll or time and attendance systems.

- Constraints:

  - The system must comply with relevant labor laws and company policies.
  - The system must be secure and protect employee and company data.
  - The system must be scalable and able to handle a growing number of employees and leave requests.
    The system must be user-friendly and intuitive for employees and managers to use.

- The system must be developed within a specified budget and timeline.

  By considering assumptions and constraints when developing a leave management system in Vue.js, you can ensure that your project is focused on delivering the most important features and functionality, while also meeting the needs and expectations of users and stakeholders.

h. Advantages and Limitations of the proposed system

Advantages of the Proposed System in Leave Management System in Vue.js:

1. Increased Efficiency: The proposed leave management system in Vue.js will automate many manual processes, such as leave request submissions, approvals, and tracking. This will increase the efficiency of the leave management process and reduce the workload of HR staff and managers.

2. Improved Accuracy: With a centralized system for managing leave requests, data entry errors and inconsistencies can be minimized. This will improve the accuracy of leave tracking and reduce the risk of errors in payroll and benefits calculations.

3. Better Transparency: The proposed system will provide better visibility into the leave management process, allowing employees and managers to track leave requests and approvals in real-time. This will improve transparency and reduce the potential for misunderstandings

or disputes.

4. Enhanced Security: The proposed system will provide enhanced security measures to protect sensitive employee data, such as social security numbers and bank account information. This will help ensure that employee data is kept confidential and secure.

5. Customizable Policies: The proposed system will allow for the creation and customization of leave policies, including accrual rates, carryover limits, and other policy details. This will enable organizations to create policies that are tailored to their specific needs and requirements.

Limitations of the Proposed System in Leave Management System in Vue.js:

1. Implementation Cost: Developing and implementing a leave management system in Vue.js can be expensive, especially for small businesses with limited budgets. The cost of hardware, software, and employee training can add up quickly.

2. User Training: The proposed system may require extensive user training for HR staff and managers to effectively use the system. This may result in a steep learning curve for some users, which could impact adoption rates.

3. Technical Expertise: Developing and maintaining a leave management system in Vue.js requires technical expertise in web development and programming. This may be a limitation for organizations that do not have the in-house technical expertise or the budget to hire external developers.

4. Integration Challenges: The proposed system may face integration challenges with other HR systems, such as payroll or time and attendance systems. This could result in additional costs and technical challenges for organizations that require seamless integration with

existing HR systems.

5. Internet Connectivity: The proposed system will rely on internet connectivity, which could be a limitation in areas with poor or unreliable internet connectivity. This may impact the availability and accessibility of the system for some users

REQUIREMENT DETERMINATION & ANALYSIS

a. Requirement Determination

Requirement determination is a critical process in the development of a leave management system in Vue.js. It involves identifying and analyzing the needs and requirements of the stakeholders, including employees, managers, and HR staff. Here are some steps to follow for effective requirement determination:

- Identify Stakeholders: The first step is to identify the stakeholders involved in the leave management process. This may include employees, managers, HR staff, and other relevant personnel.

- Gather Information: The next step is to gather information about the current leave management process, including the methods used for requesting, approving, and tracking leave requests.

- Conduct Interviews and Surveys: Conduct interviews and surveys with stakeholders to gather information about their needs and requirements for the proposed system. This can help identify pain points and areas for improvement.

- Analyze Data: Analyze the data collected from stakeholders to identify common themes and areas of focus. This will help you prioritize the requirements for the proposed system.

- Define Requirements: Based on the data analysis, define the requirements for the proposed system. This should include a list of features and functionality that are essential for the system to meet

the needs of stakeholders.

- Prioritize Requirements: Prioritize the requirements based on their importance and impact on the leave management process. This will help you focus on the most critical requirements first.

  Validate Requirements: Validate the requirements with stakeholders to ensure that they accurately reflect their needs and requirements. This can help identify any gaps or misunderstandings in the requirements

By following these steps for requirement determination, you can ensure that the proposed leave management system in Vue.js is aligned with the needs and requirements of stakeholders, and is focused on delivering the most essential features and functionality to improve the leave management process.

b. Target User

The target users for a leave management system can include employees and HR staff.

- Employees: Employees are the primary users of the leave management system as they will use the system to request time off and track their leave balances. They will also need to be able to view their leave history and receive notifications when their leave requests have been approved or denied.

- HR Staff: HR staff is responsible for managing and maintaining the leave management system. They will use the system to monitor leave balances, enforce leave policies, generate reports,

and provide support to employees and managers who may have questions or issues with the system.

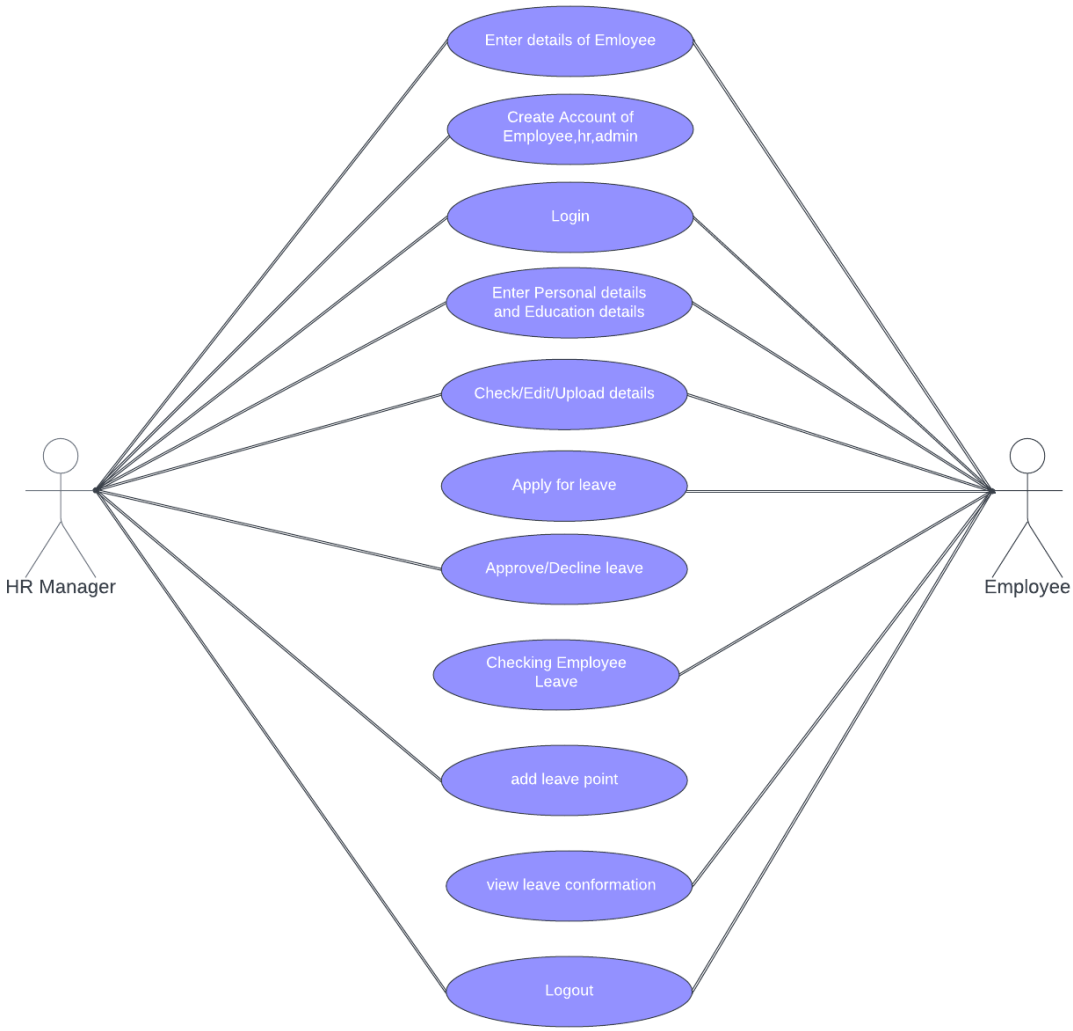
Other potential users of the leave management system may include payroll administrators who need access to leave data to ensure that employees are paid accurately during periods of leave, and senior management who may need to review leave data for strategic planning purposes.

In summary, the target users for a leave management system are employees and HR staff, with potential additional users including payroll administrators and senior management. The system should be designed to meet the specific needs of each user group and should be user-friendly and accessible to ensure adoption and usage.

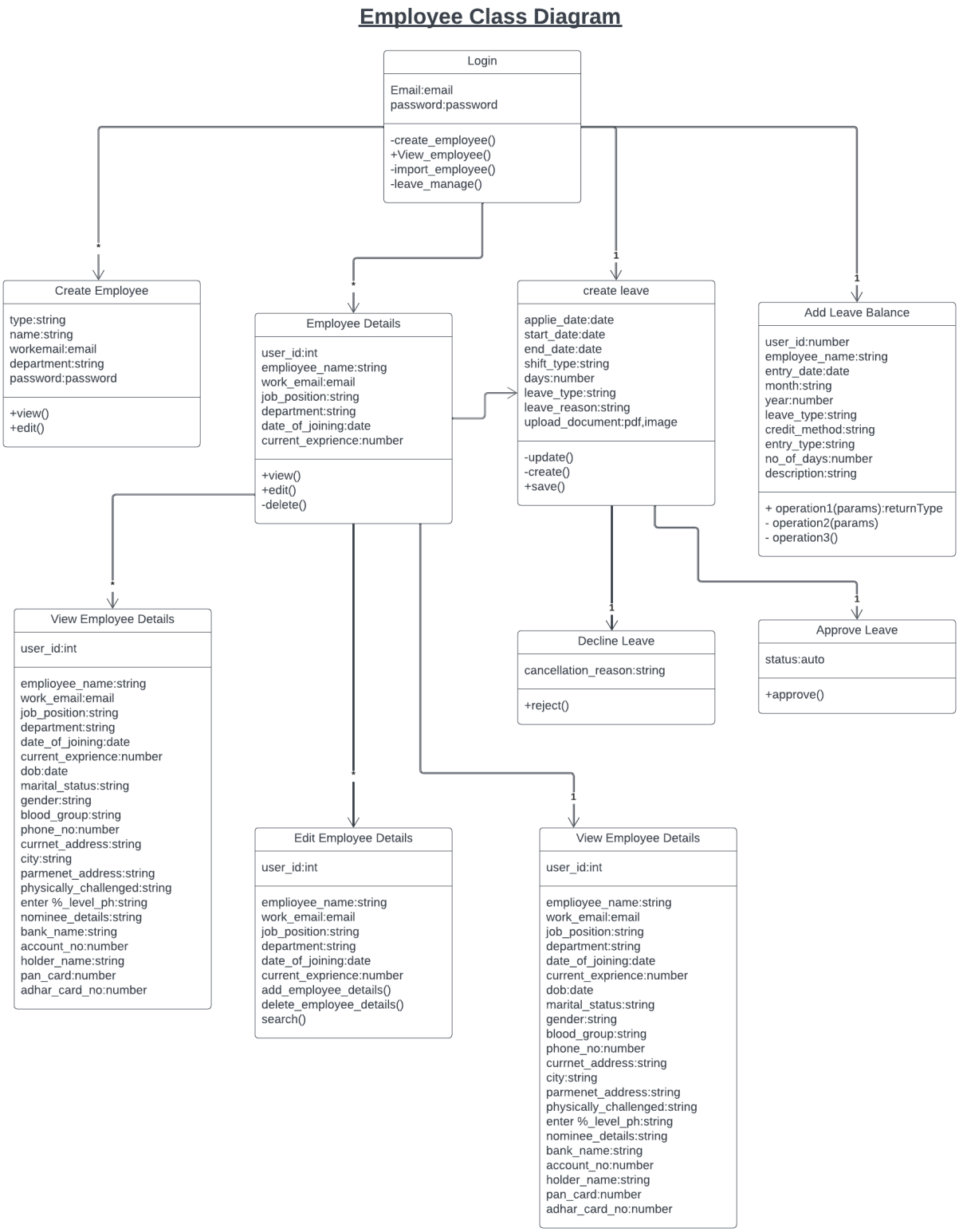
SYSTEM DESIGN

a. Use Case Diagram

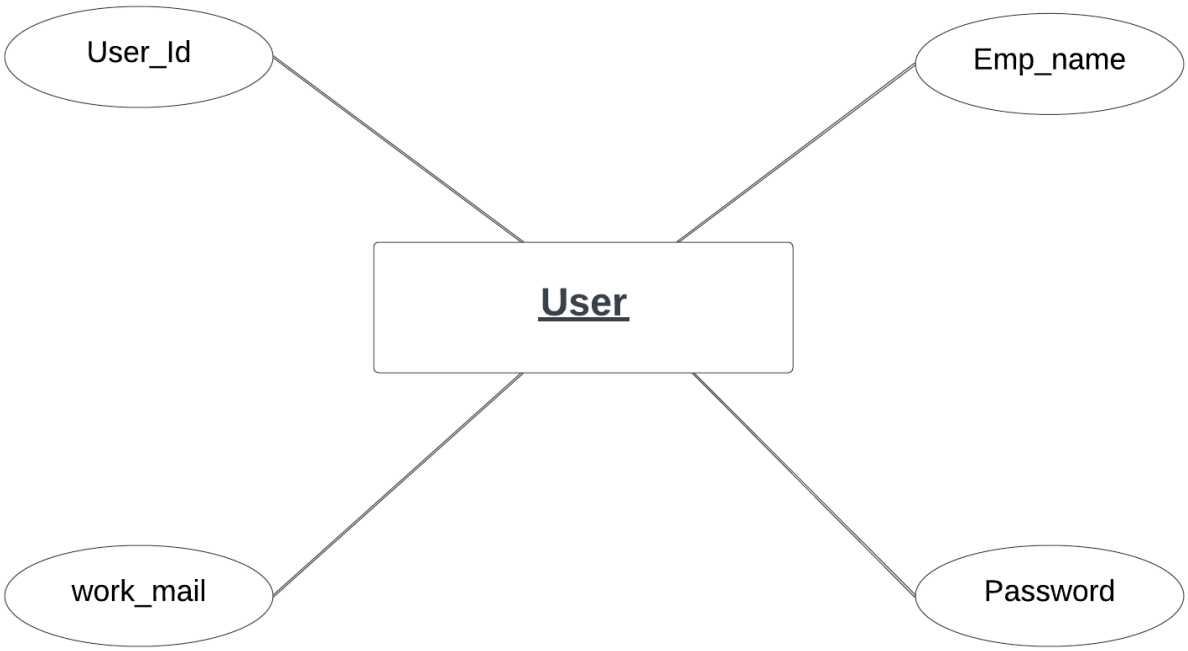
Employee Leave Management System



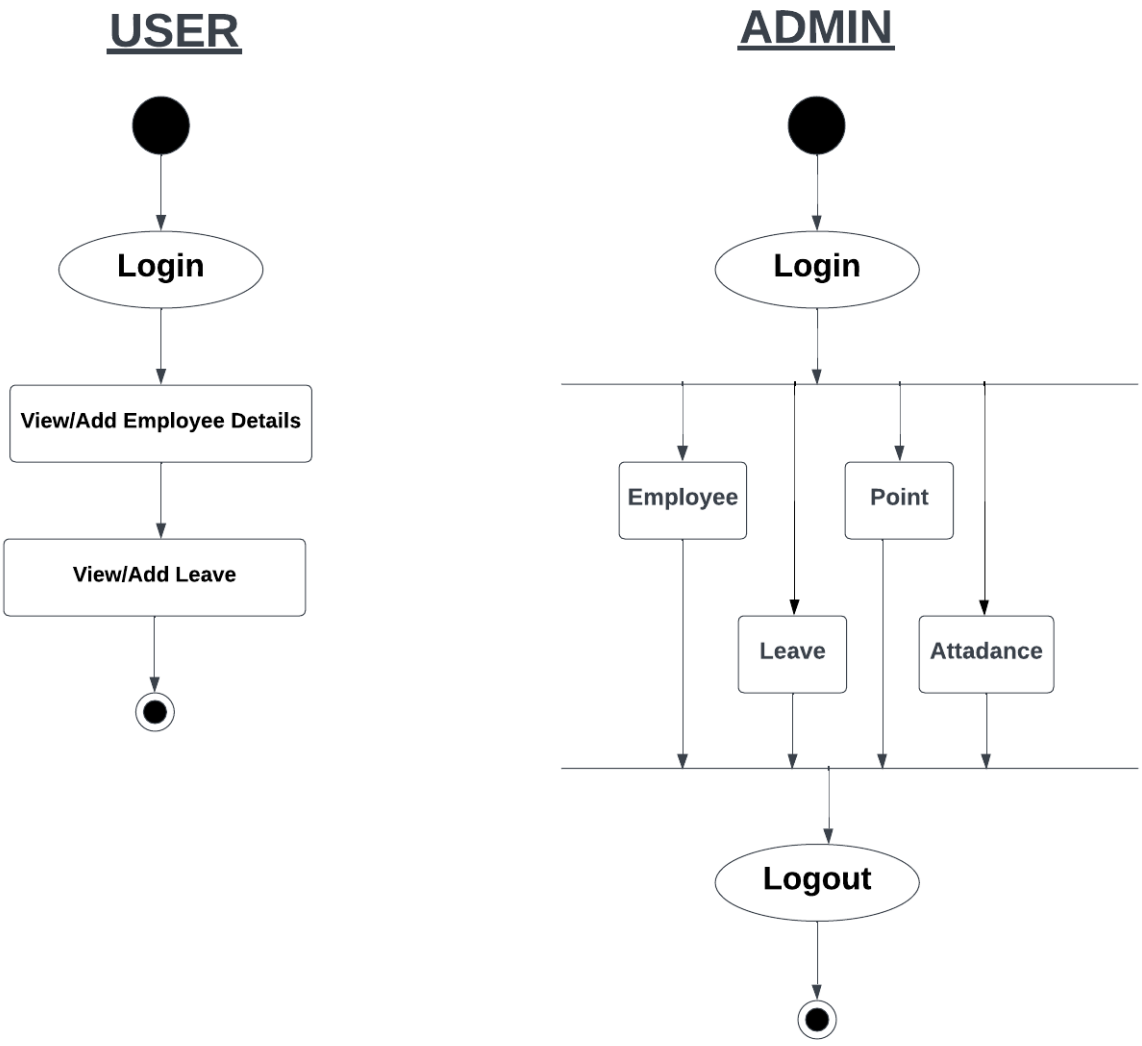
b. Class Diagram



C. Interaction Diagram



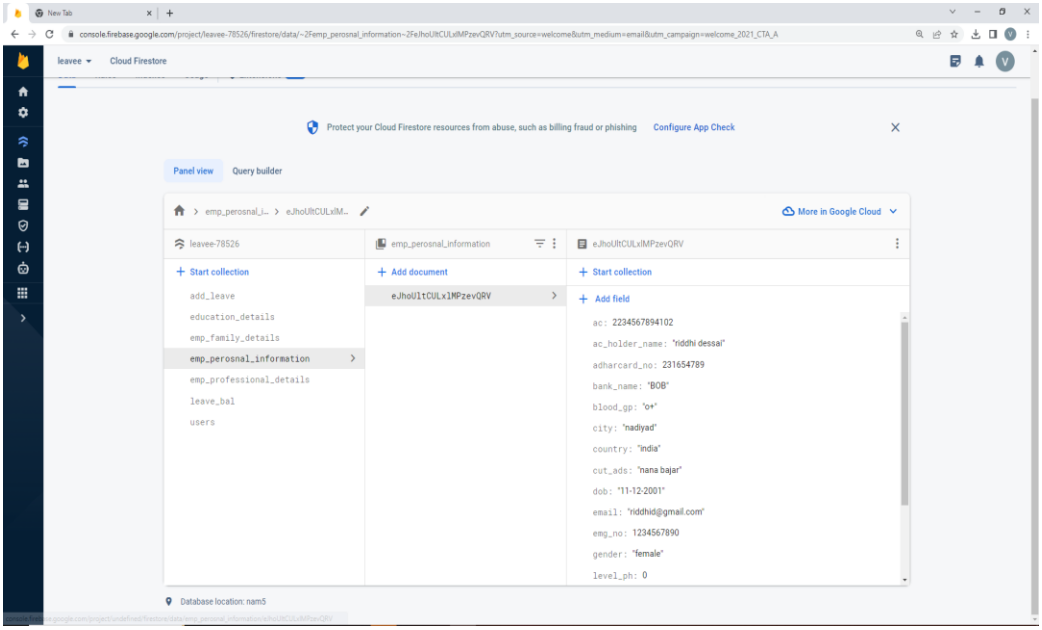
d. Use Case Diagram



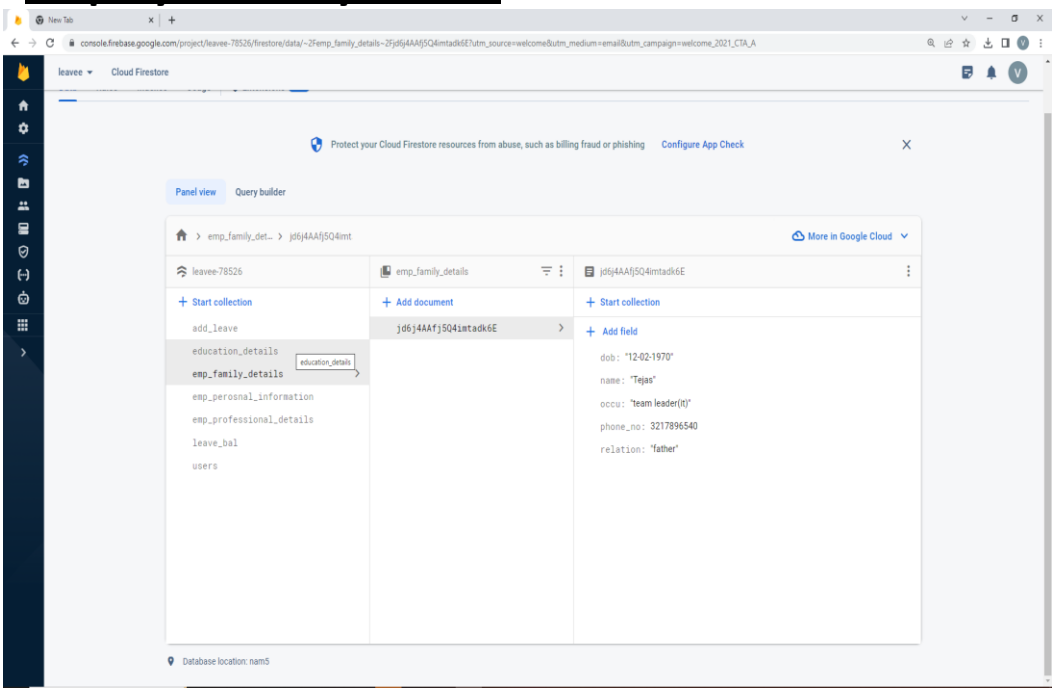
e. Data Dictionary

- Employee ID: A unique identifier for each employee in the system.
- First Name: The first name of the employee.
- Last Name: The last name of the employee.
- Date of Birth: The date of birth of the employee.
- Gender: The gender of the employee.
- Join Date: The date on which the employee was hired.
- CL (Casual Leave): The term 'casual' defines an event or situation that occurs by chance and without any plan.
- PL (Privilege Leave): The purpose of privilege leave is to provide employees with an opportunity to take a break from work and rejuvenate themselves.
- SL (Sick Leave): Sick leave is the time that a person spends away from work because of illness or injury.
- Reject/Approve: Get the reason of reject leave otherwise approve the leave.
- Leave Balance: To add leave balance if not get any leave in month.

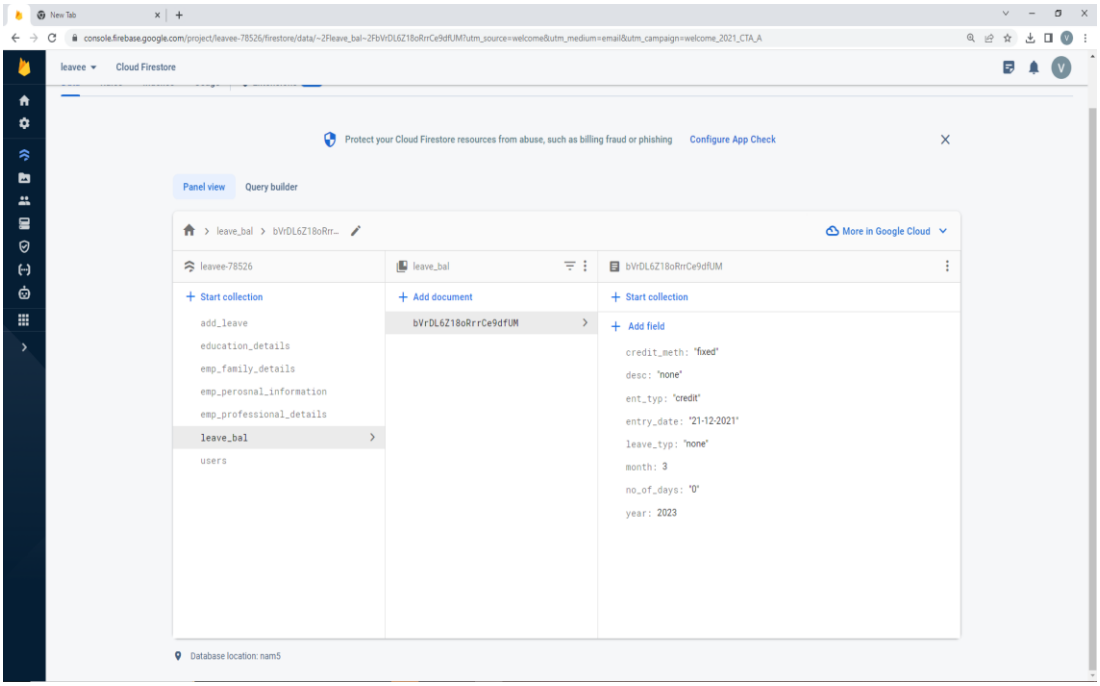
Employee personal details



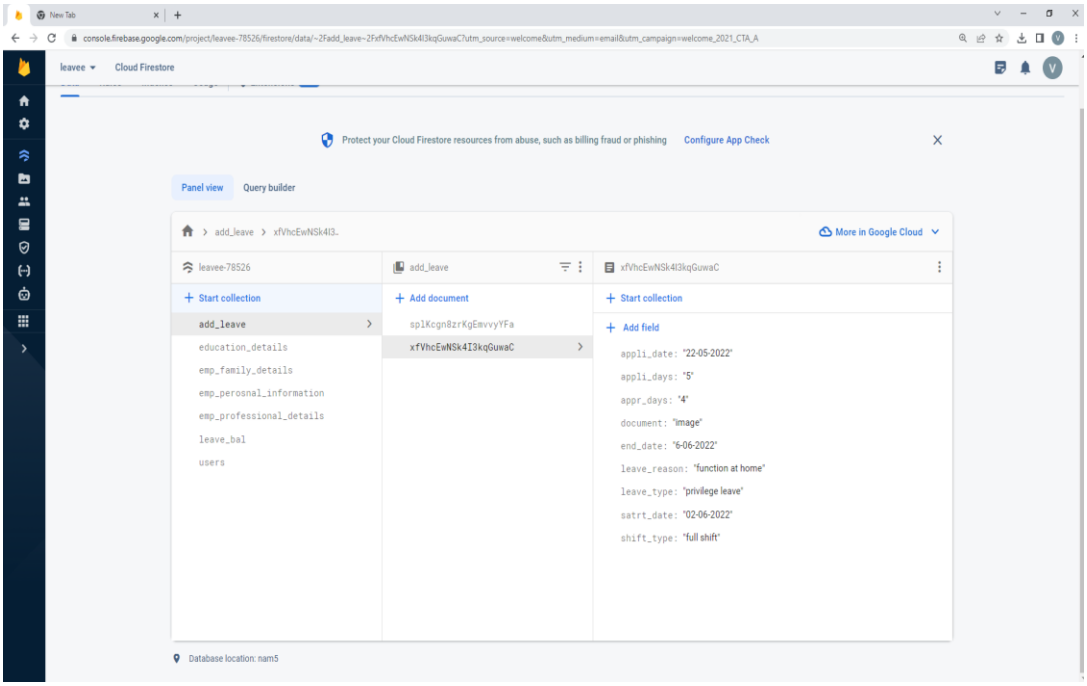
Employee family details



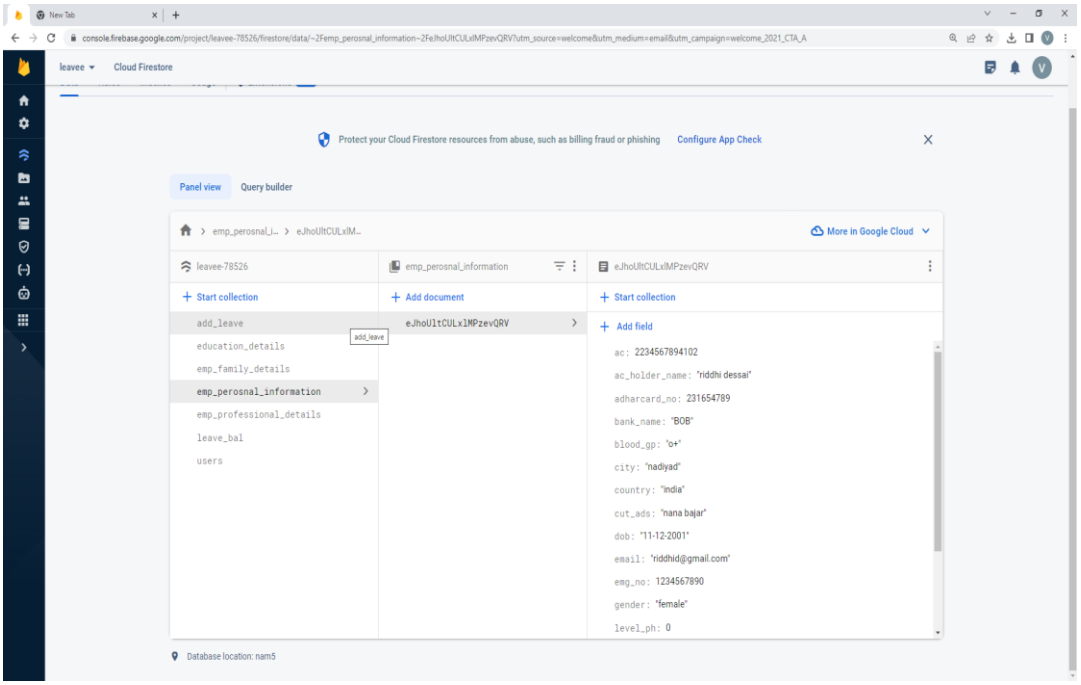
Add leave balance



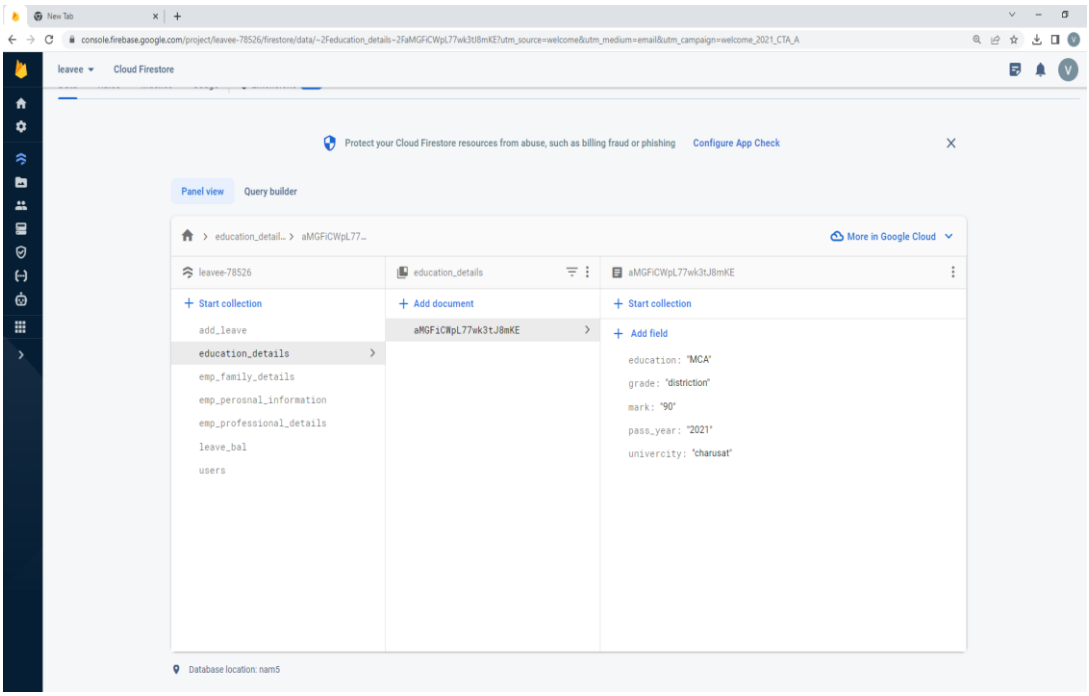
Apply leave



Employee Professional information



Employee Education details



DEVELOPMENT

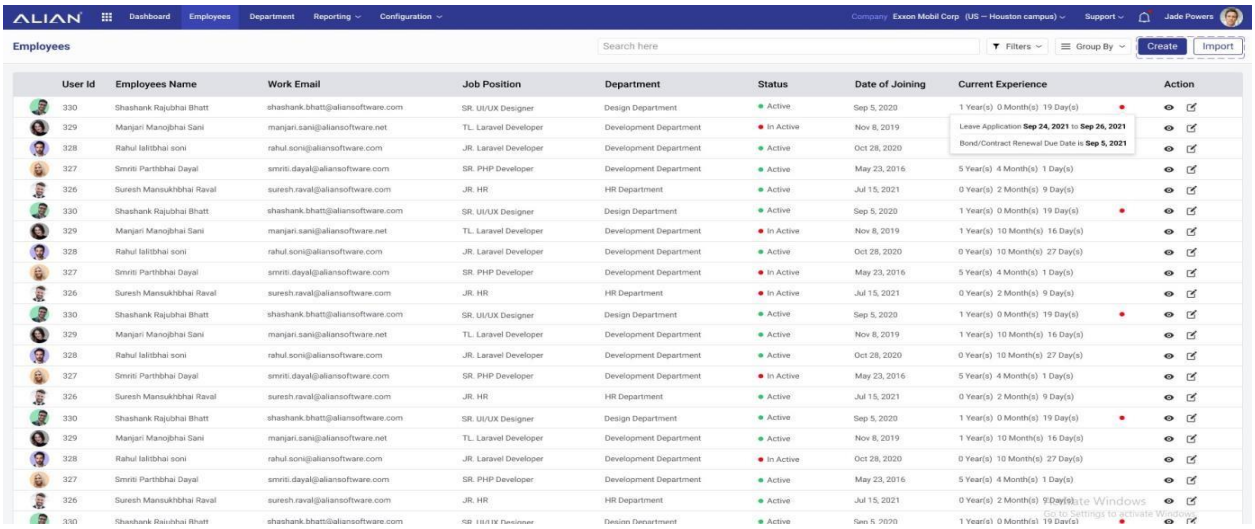
a. Coding Standard

- Use consistent naming conventions for variables, functions, and components. For example, use camelCase for variables and functions and Pascal Case for components.
- Use indentation and spacing consistently to improve readability. For example, use two spaces for indentation and add spaces around operators and after commas.
- Use comments to explain complex or important sections of code. Comments should be concise and explain what the code is doing, not how it's doing it.
- Use modular design principles to break down the code into reusable components. This makes it easier to maintain and scale the codebase over time.
- Use Vue.js specific coding standards such as using v-bind instead of : and v-on instead of @.
- Use ES6 features such as let and const instead of var and arrow functions instead of function expressions.
- Use proper error handling and validation to ensure that the system is robust and secure.
- Use linting tools such as ESLint to ensure code consistency and catch potential errors.
- Write unit tests for critical sections of the codebase to ensure that changes to the code do not break existing functionality.
- By following these coding standards, you can ensure that your employee payroll management system in Vue.js with Firebase is maintainable, scalable, and consistent, making it easier for developers to work on and improve over time

b. Screen Shots

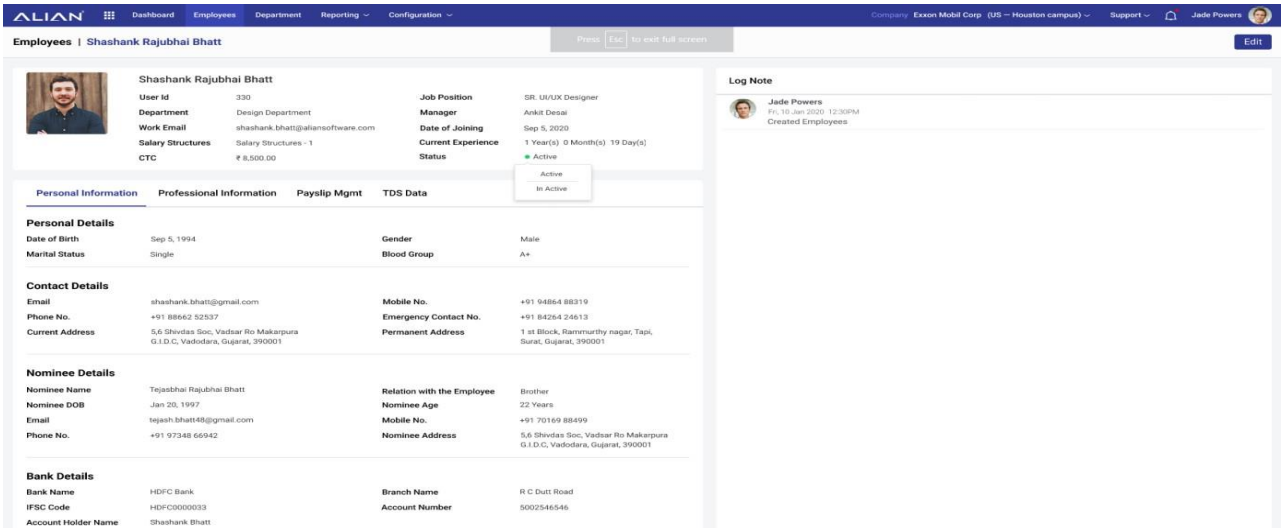
Admin Side

Employee listing page



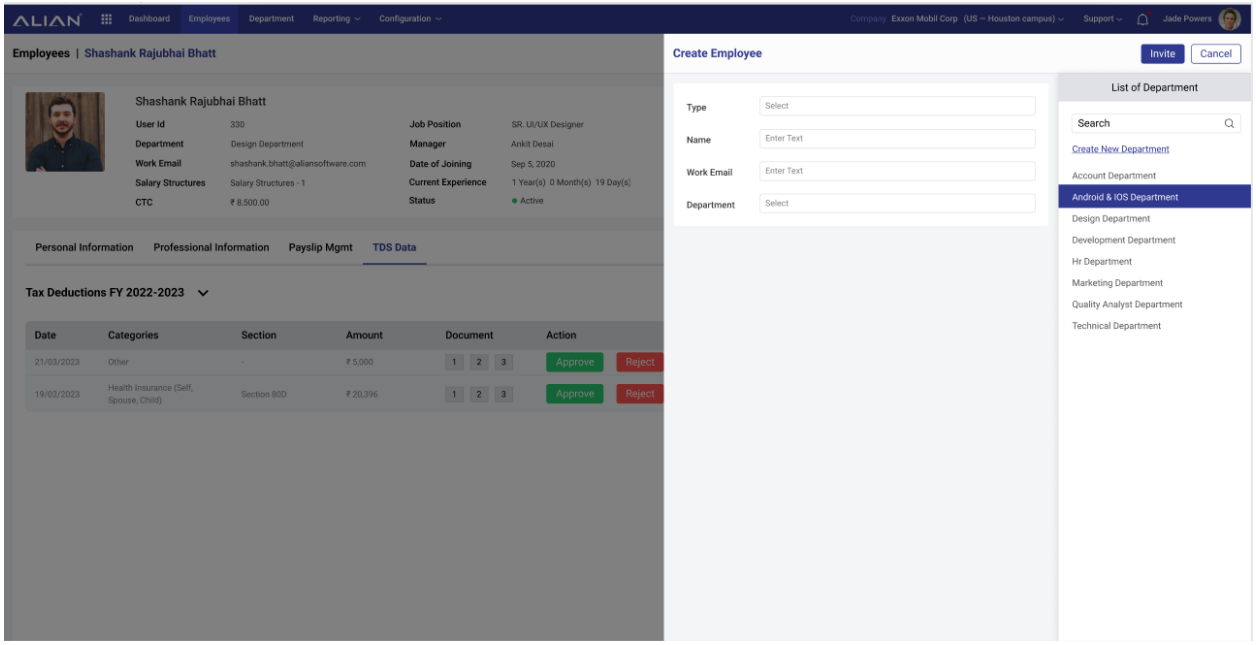
Description: This page describe all employees details, HR can edit employee, create employee and searchemployee.

Employee Details Page



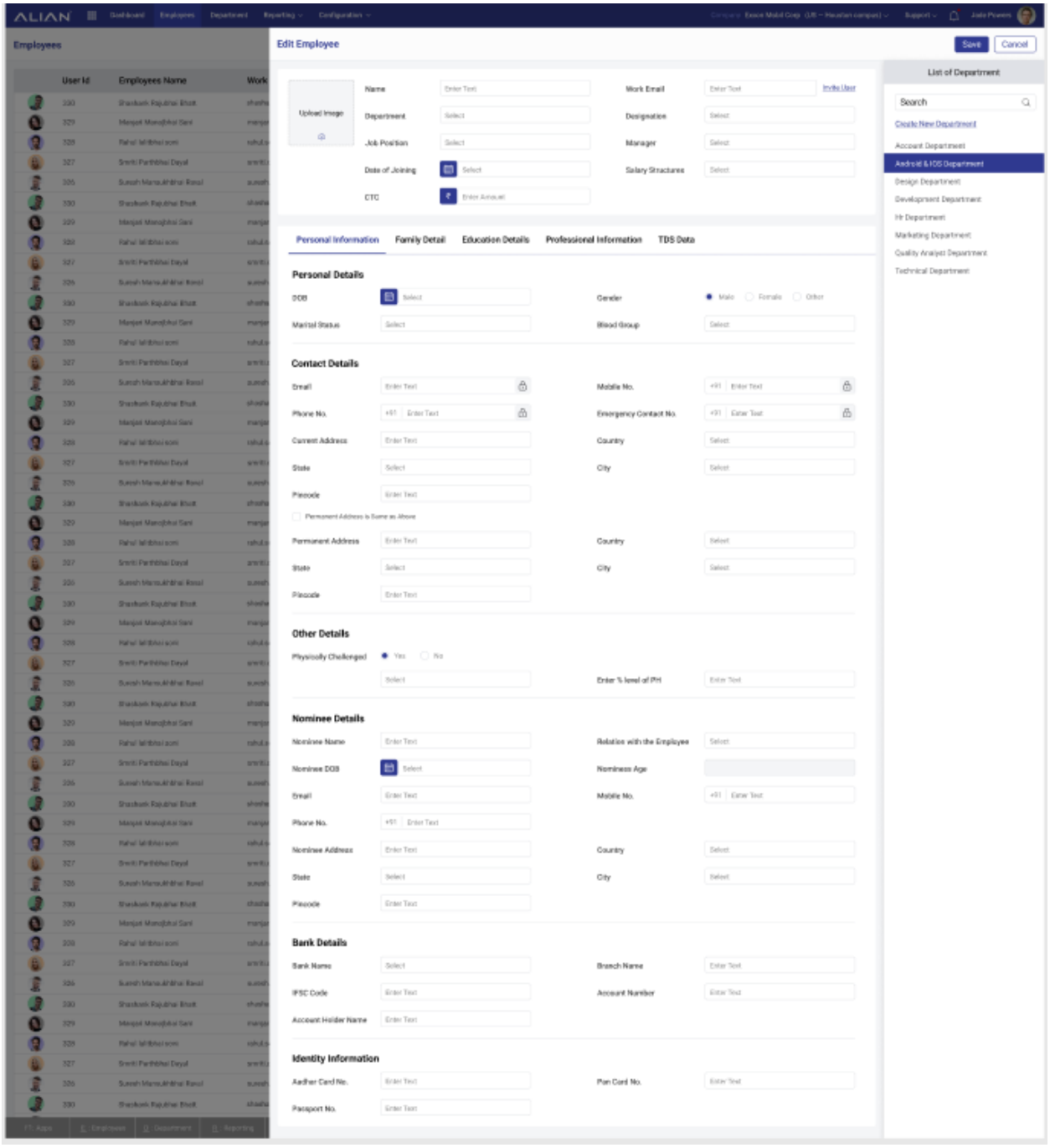
Description: This page describe employee more details

Create model



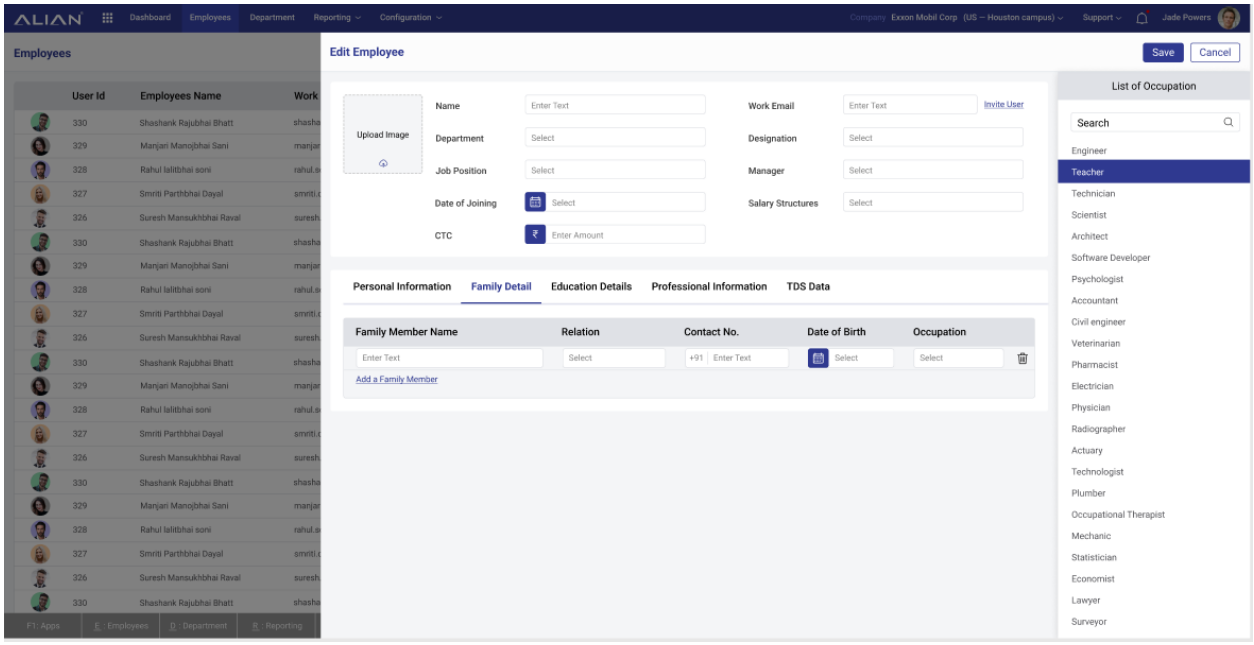
Description: This page describes create model like employee, HR, admin.

Edit personal information



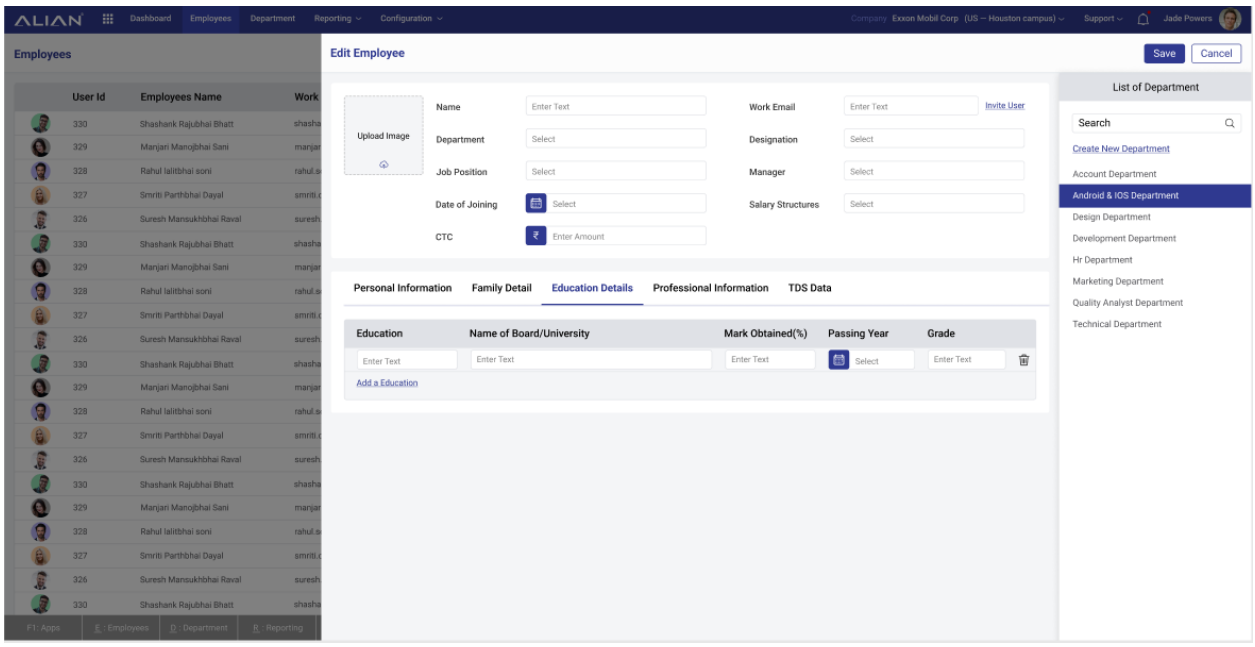
Description: This page describes edit personal information.

Edit family details (user side also)



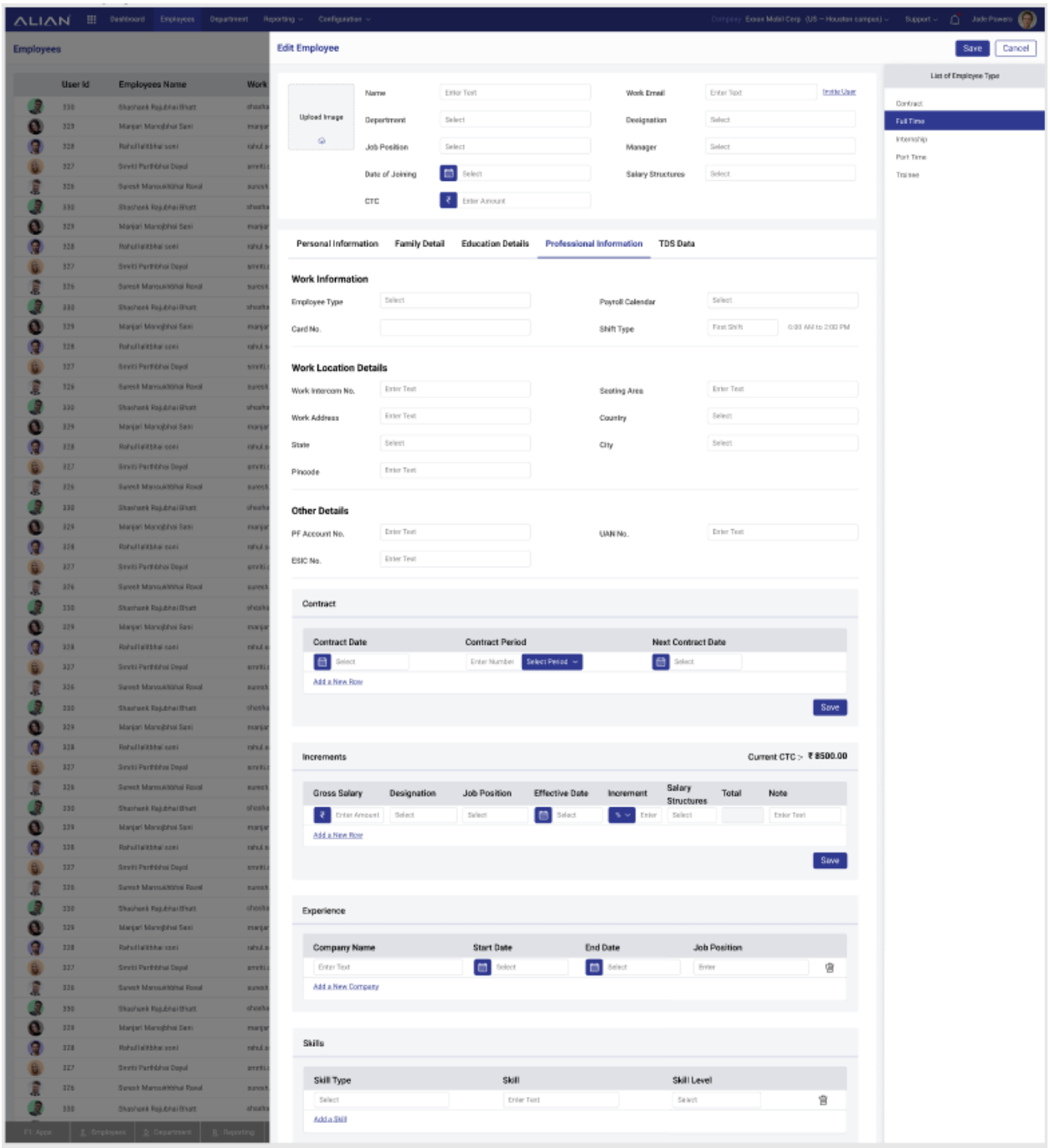
Description: This page describes edit family details.

Edit education details (user side also)



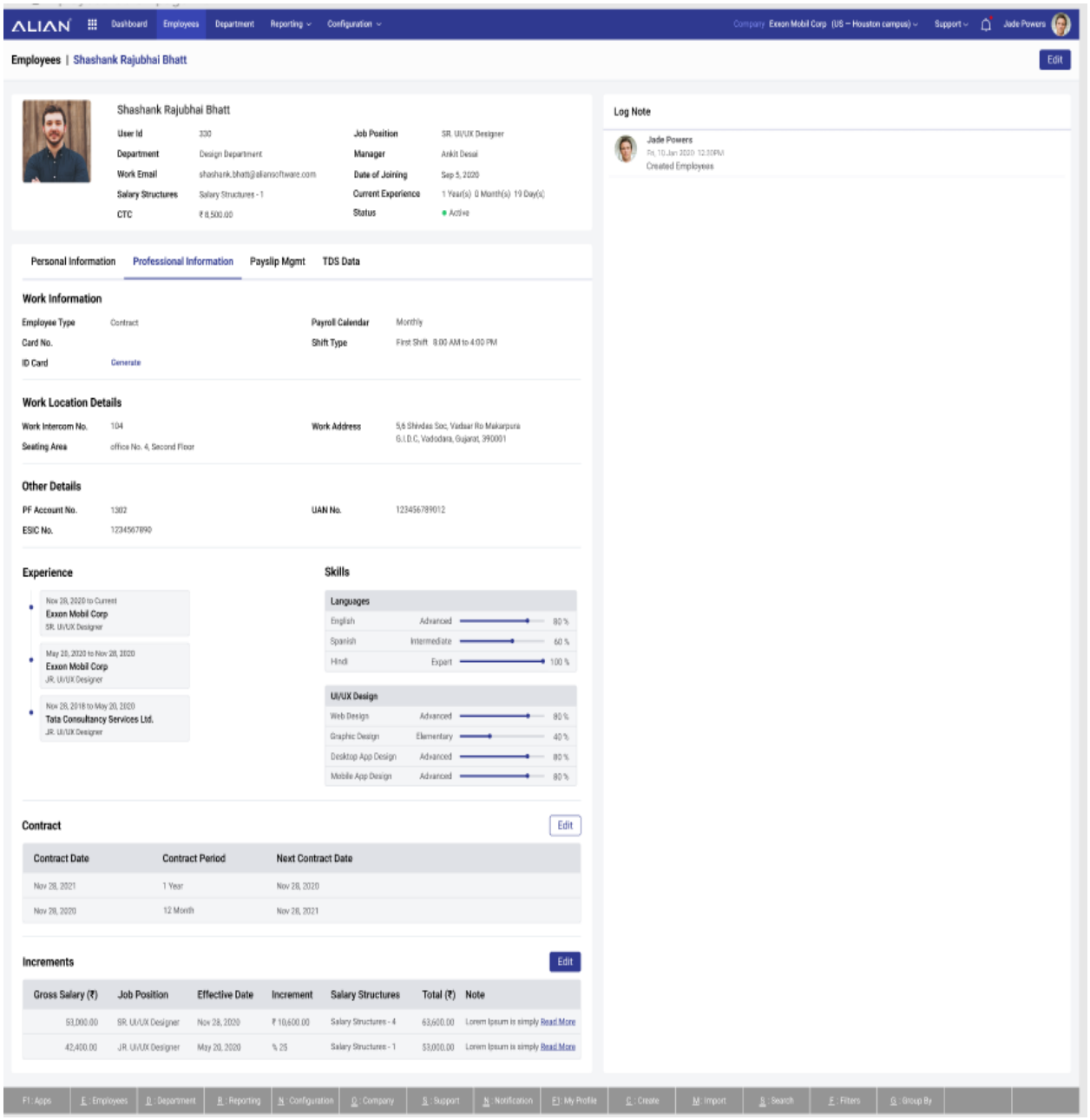
Description: This page describes edit education details.

Edit professional details



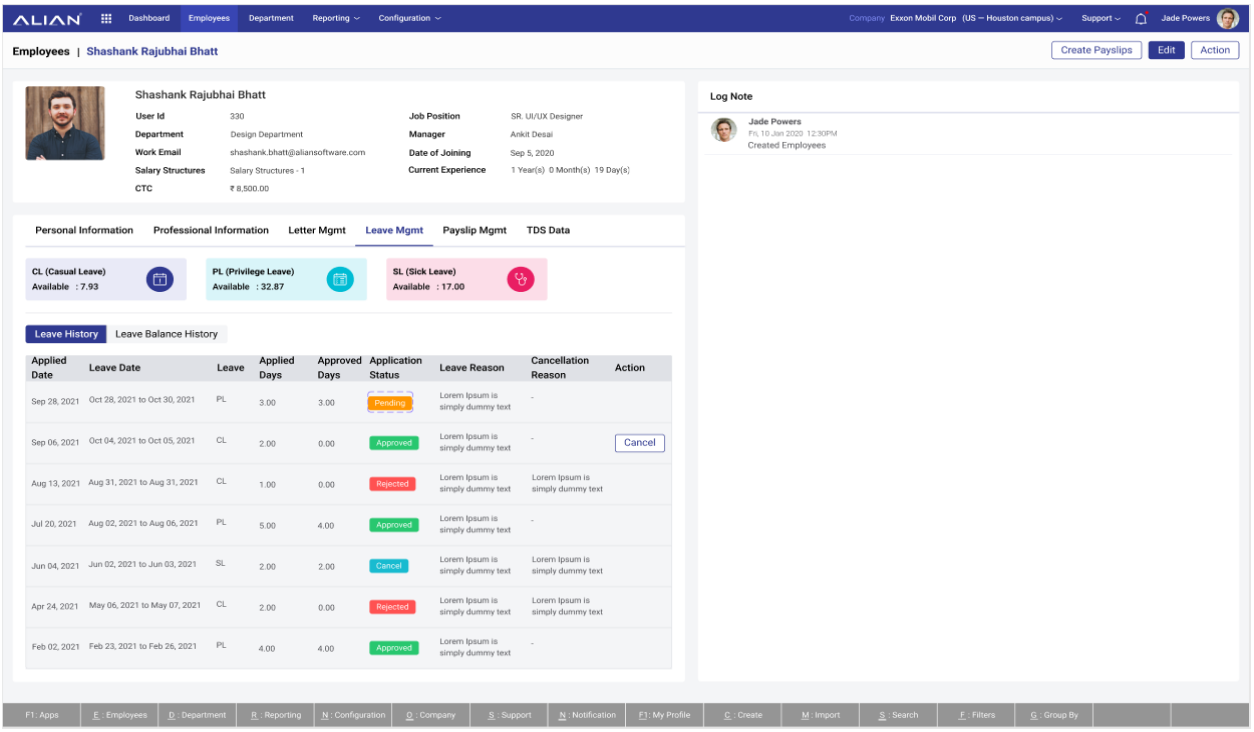
Description: This page describes edit professional details.

View professional details



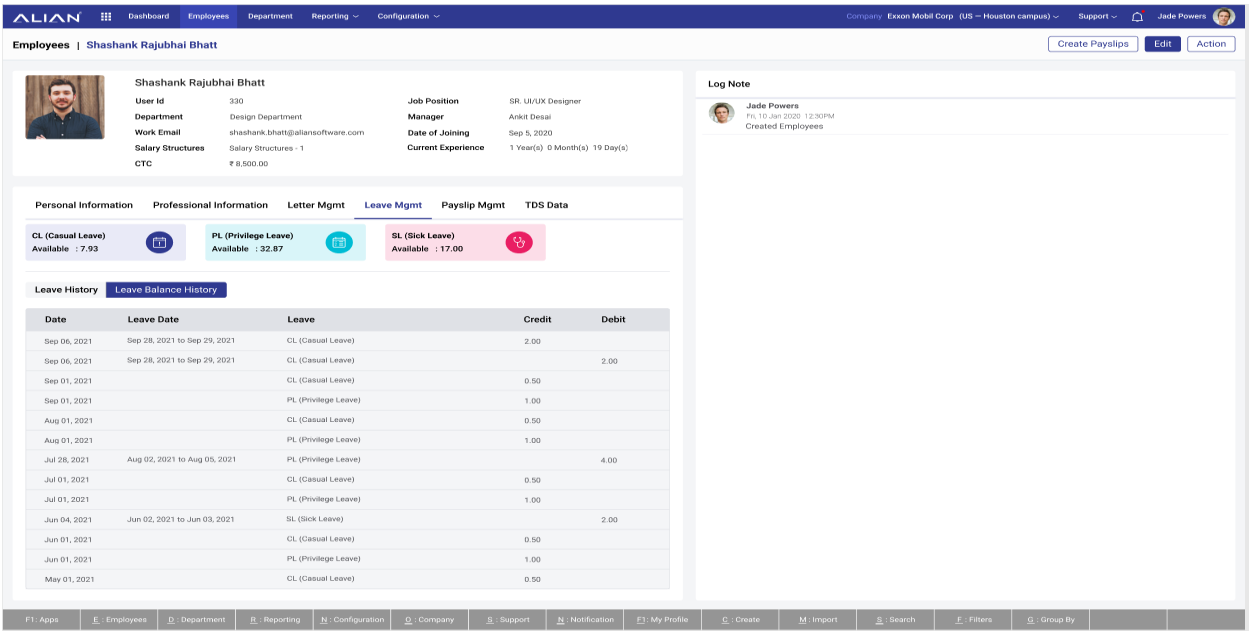
Description: This page describes professional details.

View employee leave details page



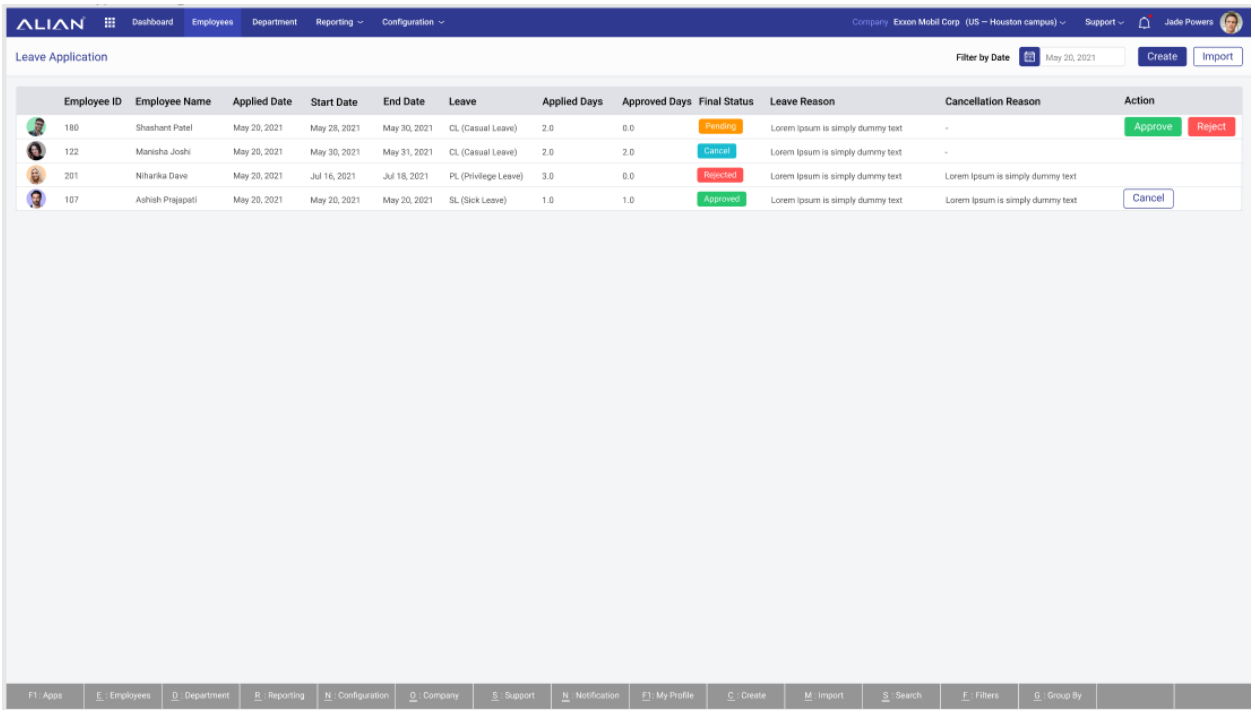
Description: This page describes employee all leave details.

Leave balance history



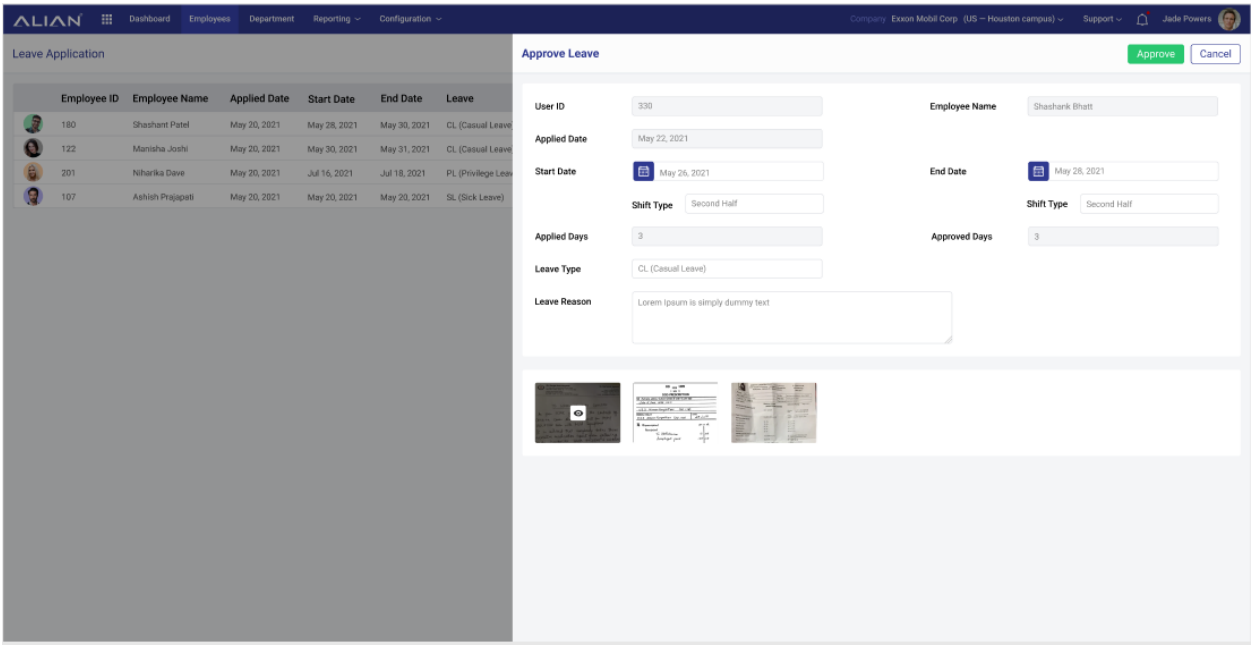
Description: This page describes leave balance history.

All employee leave details



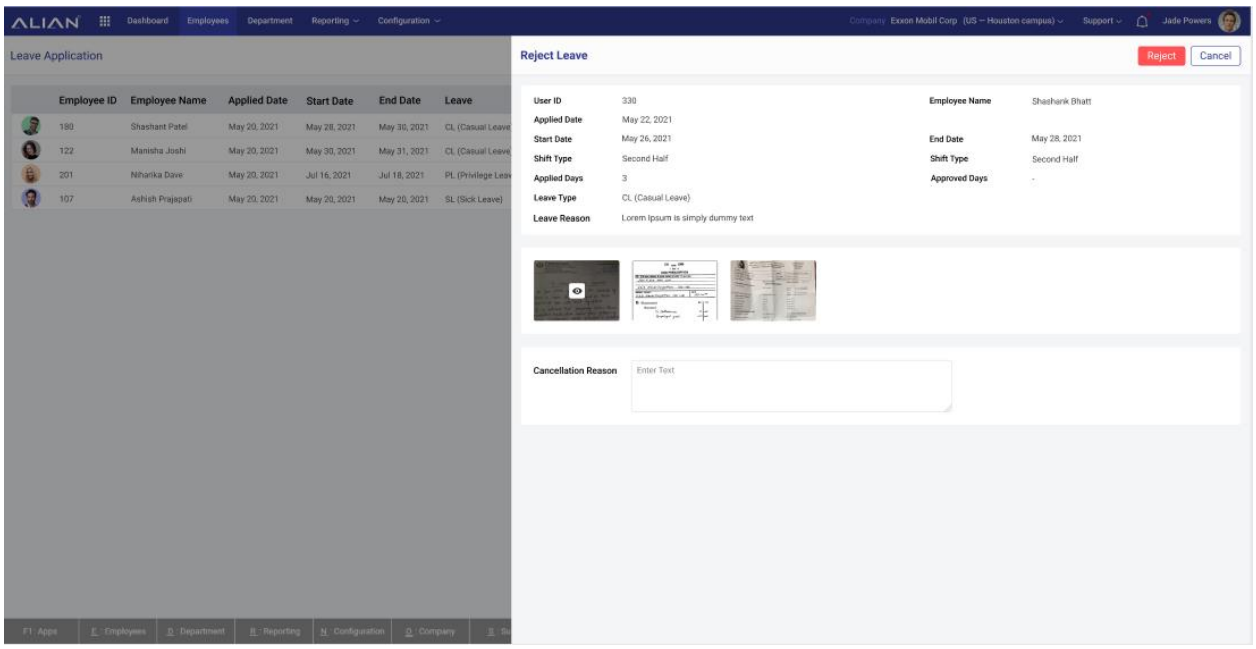
Description: This page describes all employees leave reject or approve.

Approve leave



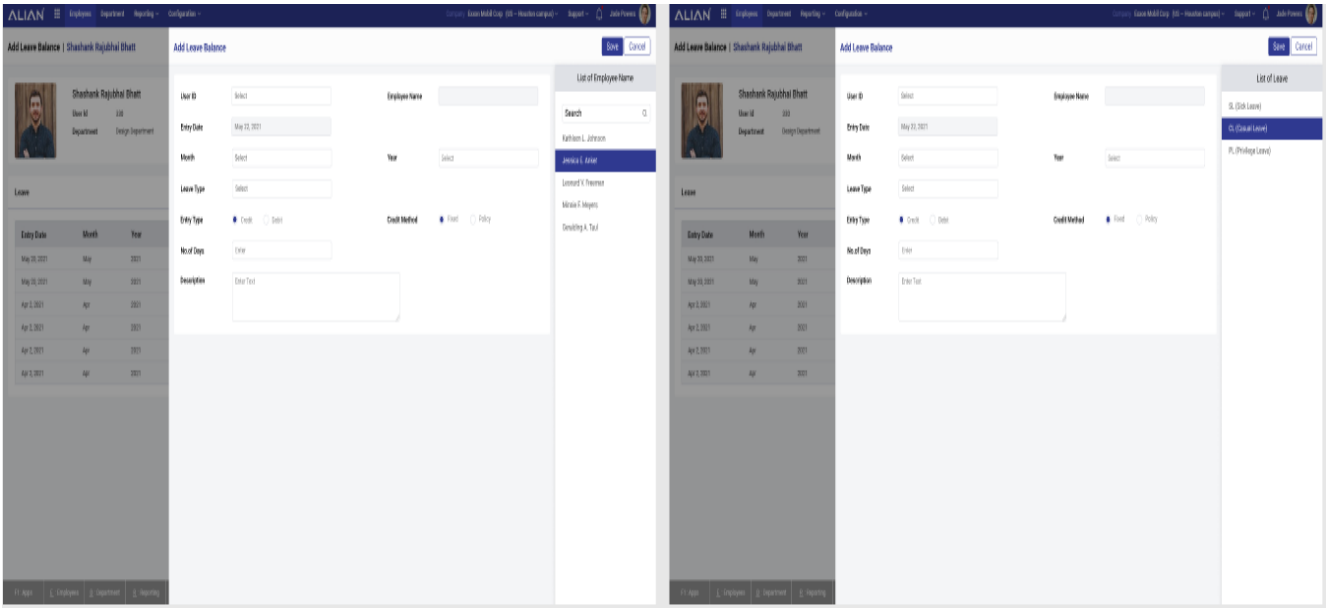
Description: This page describes approve leave of employee.

Reject leave



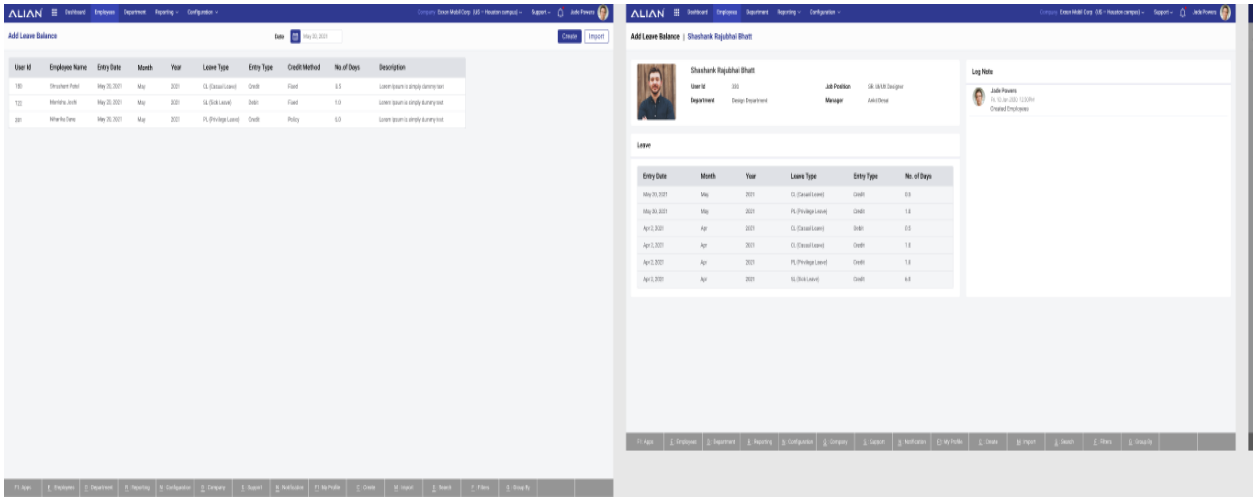
Description: This page describes reject leave of employee.

Add leave balance



Description: This page describes reject leave of employee.

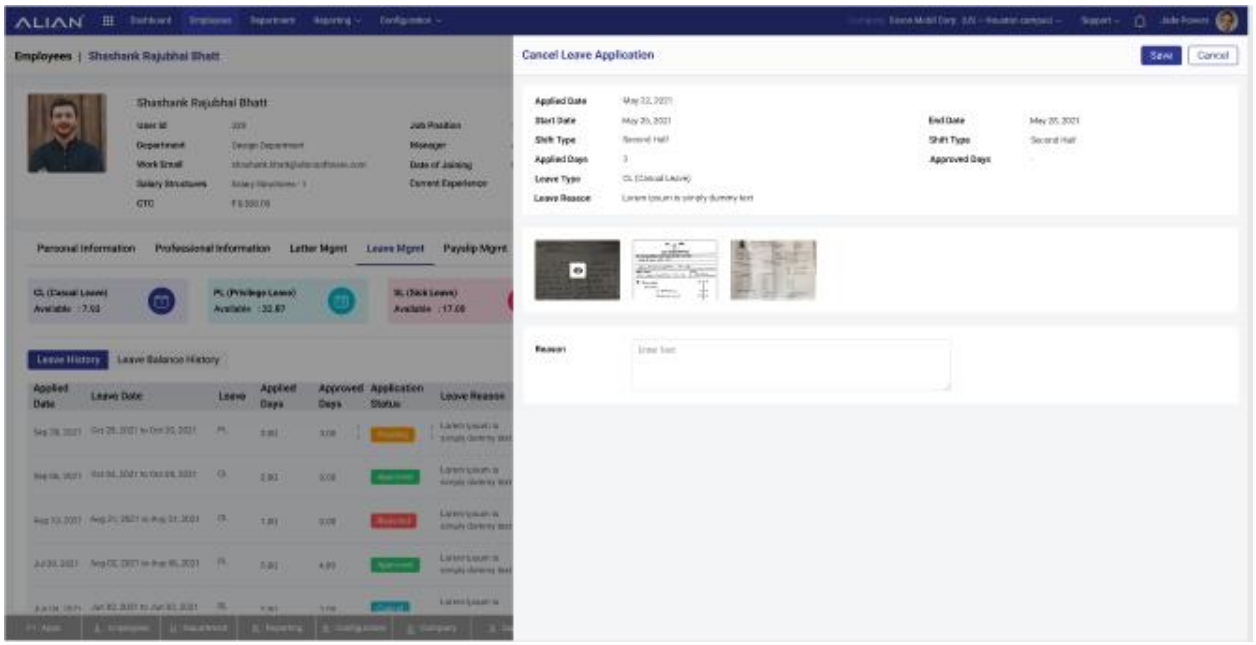
View leave balance

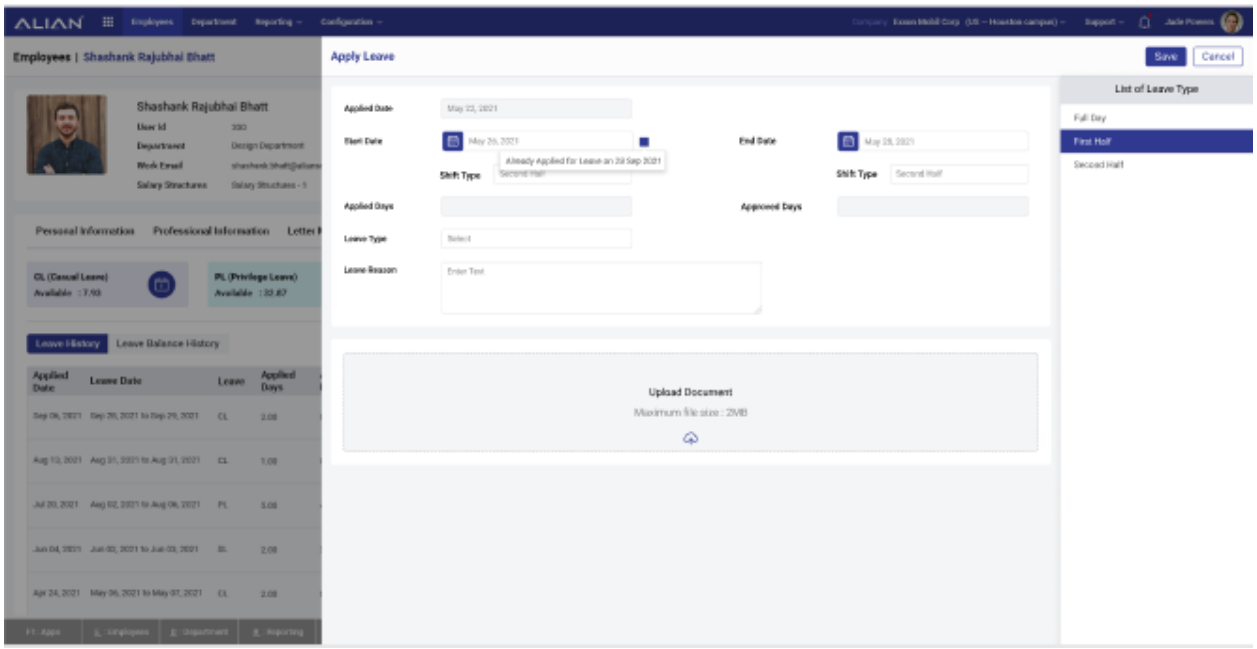


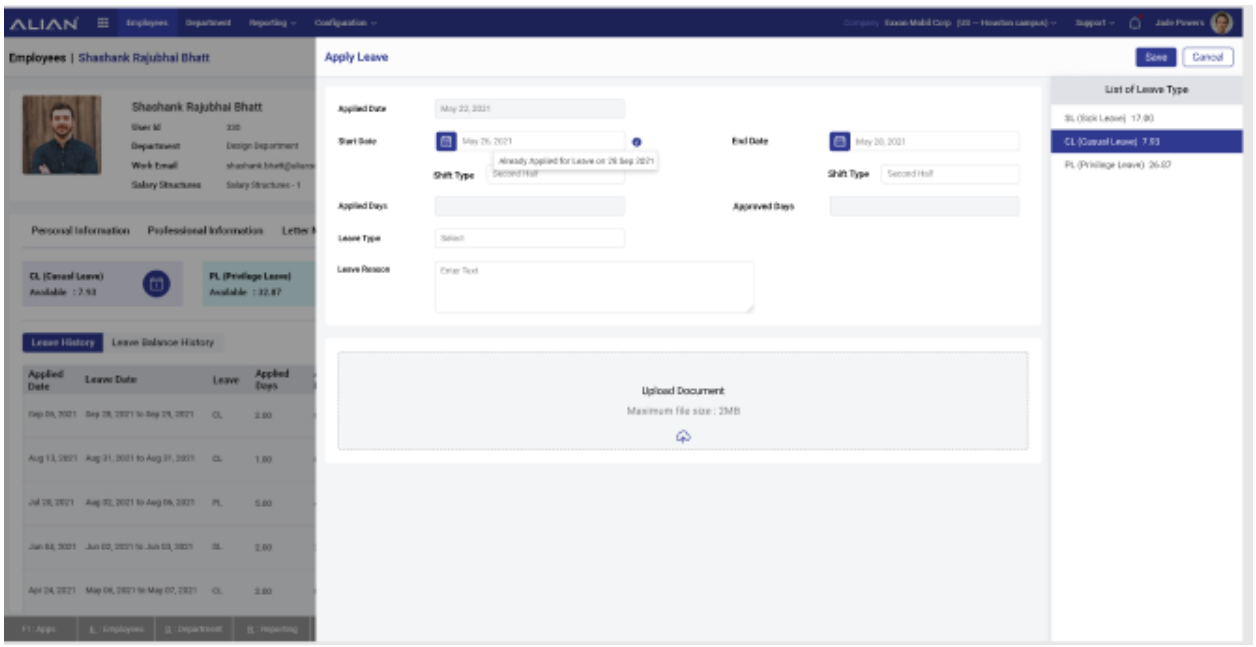
Description: This page describes reject leave of employee.

❖ User Side

Add leave application

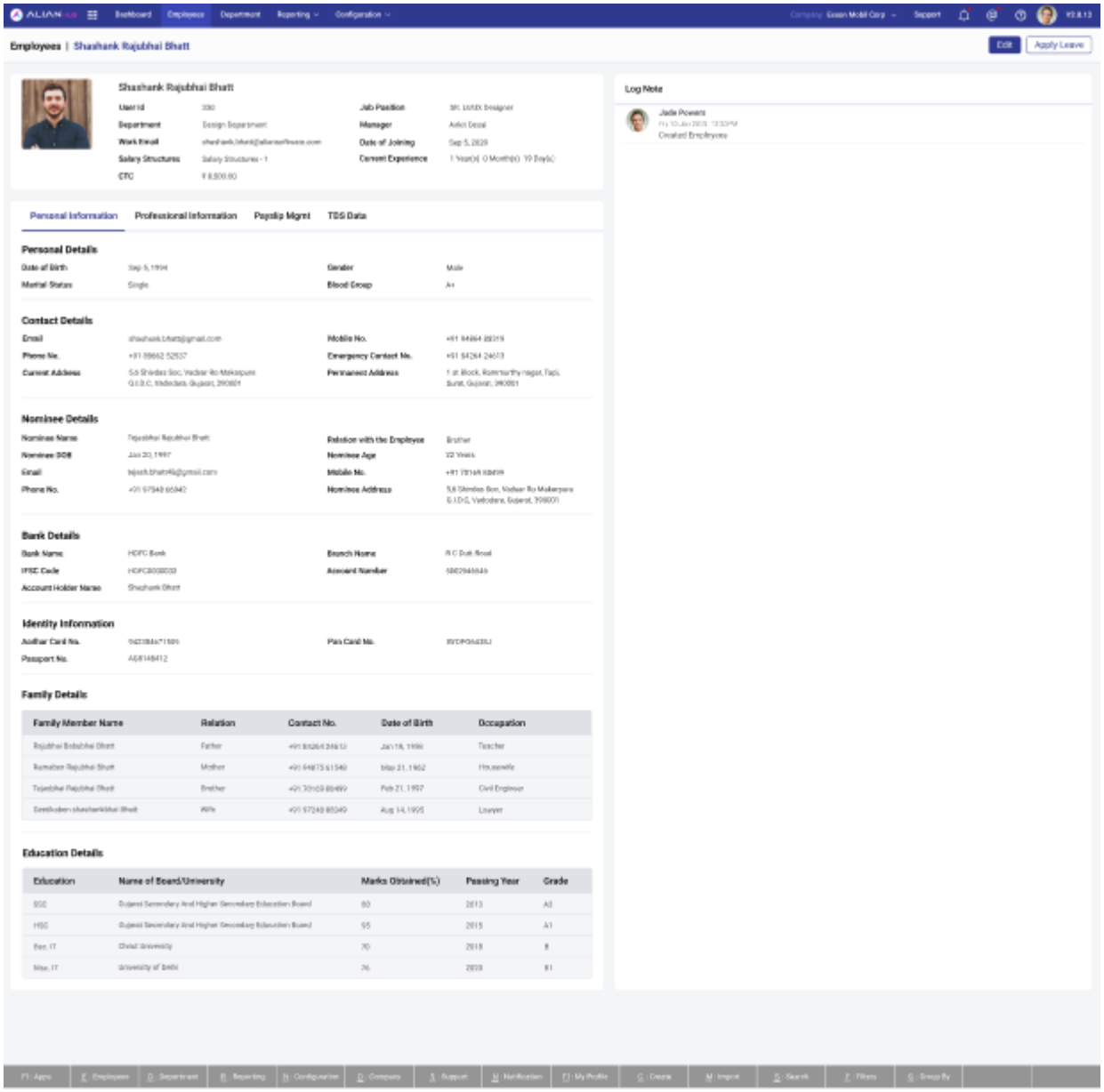






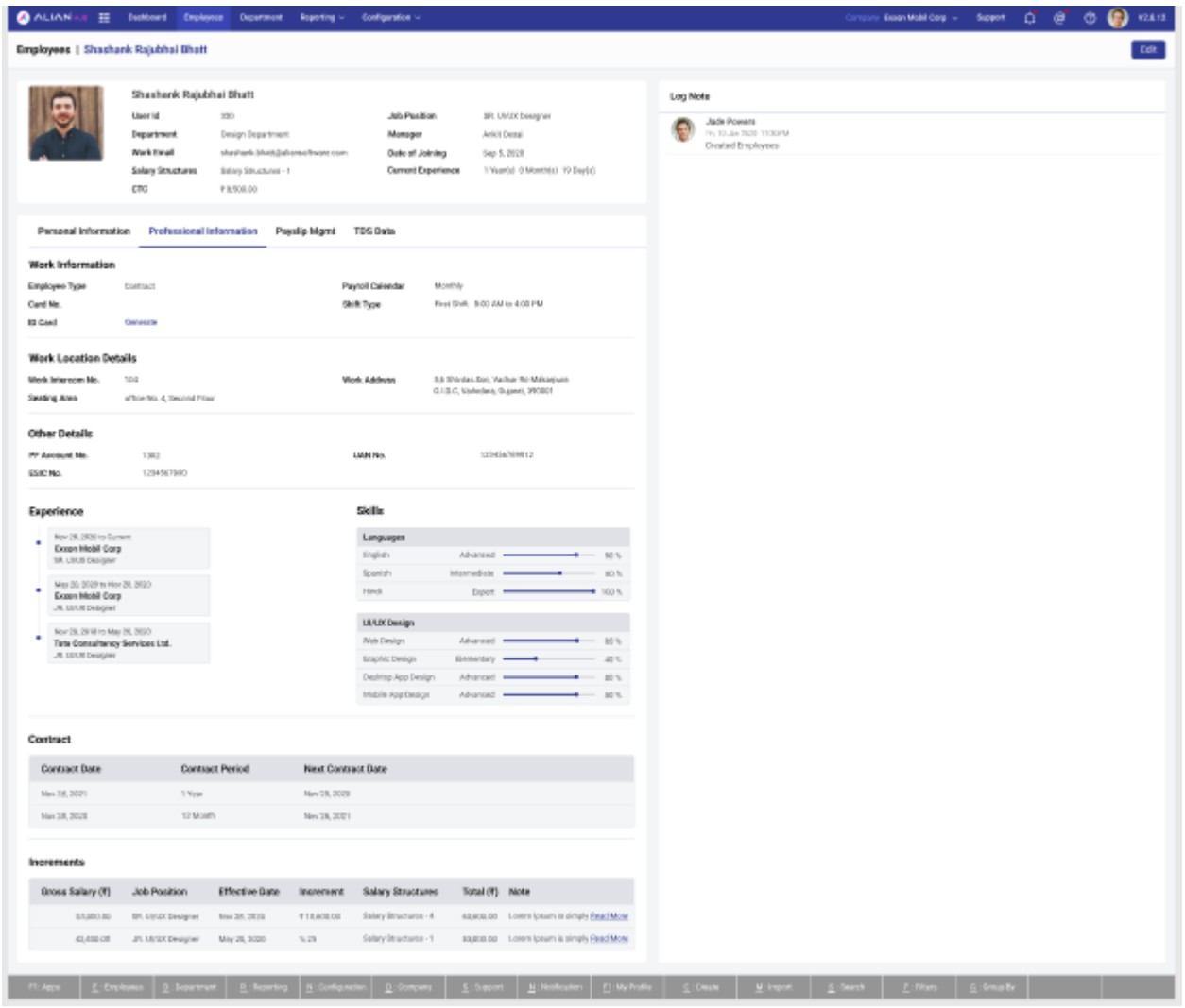
Description: This page describes apply for leaves and view how many days, which type of leave apply.

View personal information



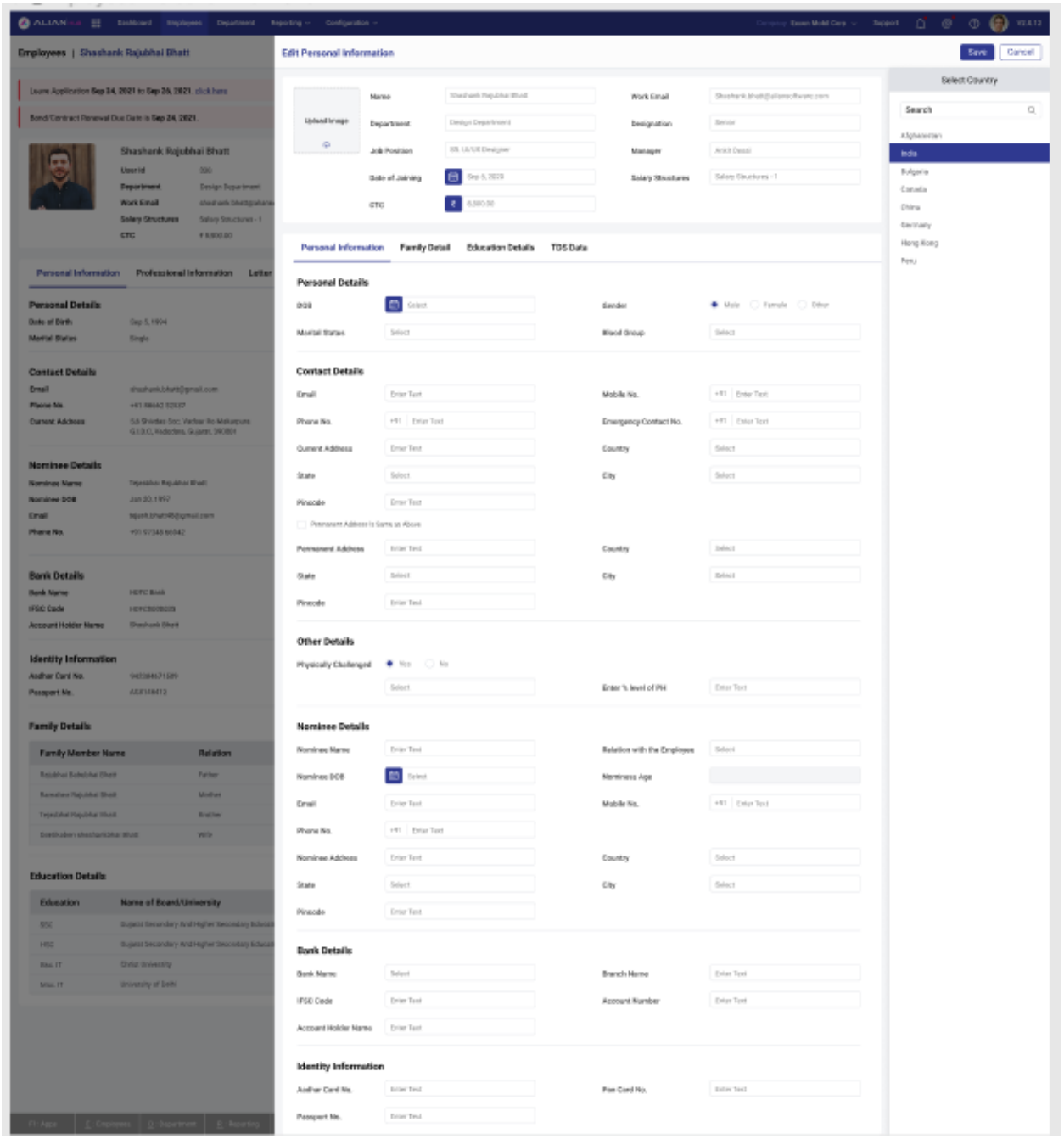
Description: This page describes view personal information.

View professional information



Description: This page describes view personal information.

Edit personal information



Description: This page describes edit personal information.

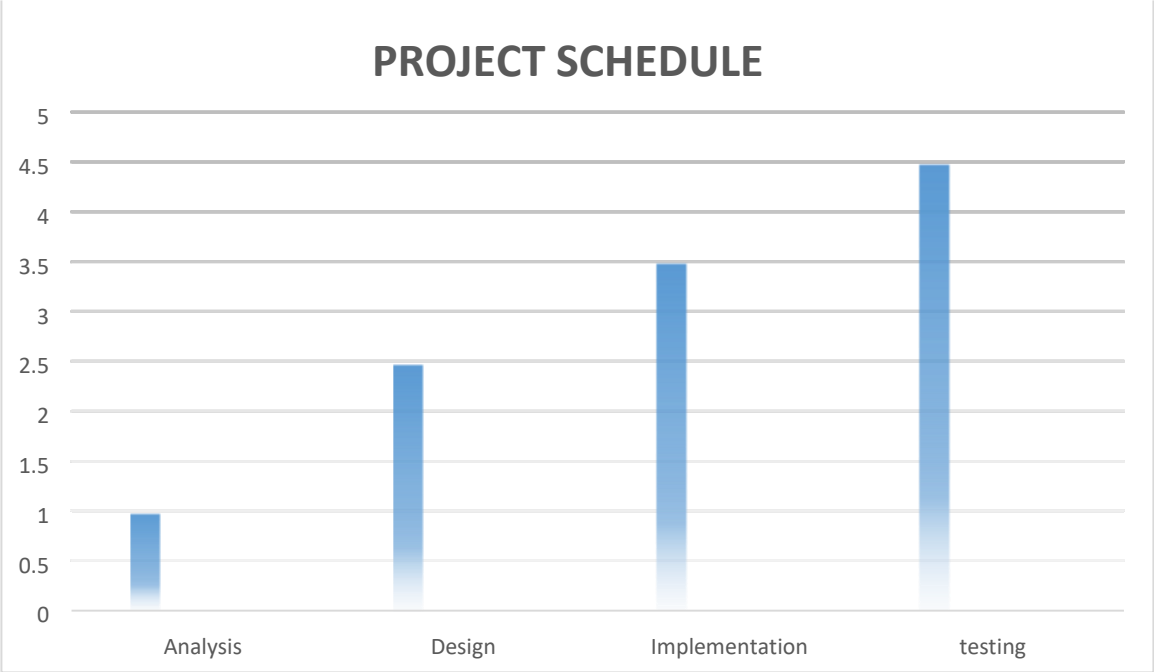
AGILE

DOCUMENTATION

a. Agile project chapter

Project Name	Employee Leave Management System Application
Project User	Customer
Expected Start Date	22 -3- 2023
Project Detail:	
Mission	Design a report module for Employee leave Management System. This report creates for the apply leave and approve or reject leave with reason and view all the employees leave and all information.
Scope	First of all, we create our design how to make our web application more useful and user access easily for that we created user friendly design.

b. Agile roadmap



c. Agile project plan

Task Name	Duration	Start Date	End Date	Status
Requirement Gathering	2	22-3-23	23-3-23	Done
Analysis	5	23-3-23	24-2-23	Done
System Design	20	25-2-23	27-3-23	Done
Implementation	25	28-3-23	-	In Process
Testing	7	--	-	Done

d. Agile user story

User Story ID	As a/an	I want….	So that…
1	HR	Add Admin and Employee	HR Can Add Employee And Admin
2	HR	Add Employee Details and Manages leave	Add Details of Employee and Manages leave
3	HR	Reject Employee Leave	HR Can Reject leave with Reason.
4	Employee	Edit/View Employee Details	Employee Can Manages, View Employee Detail

e. Agile release plan

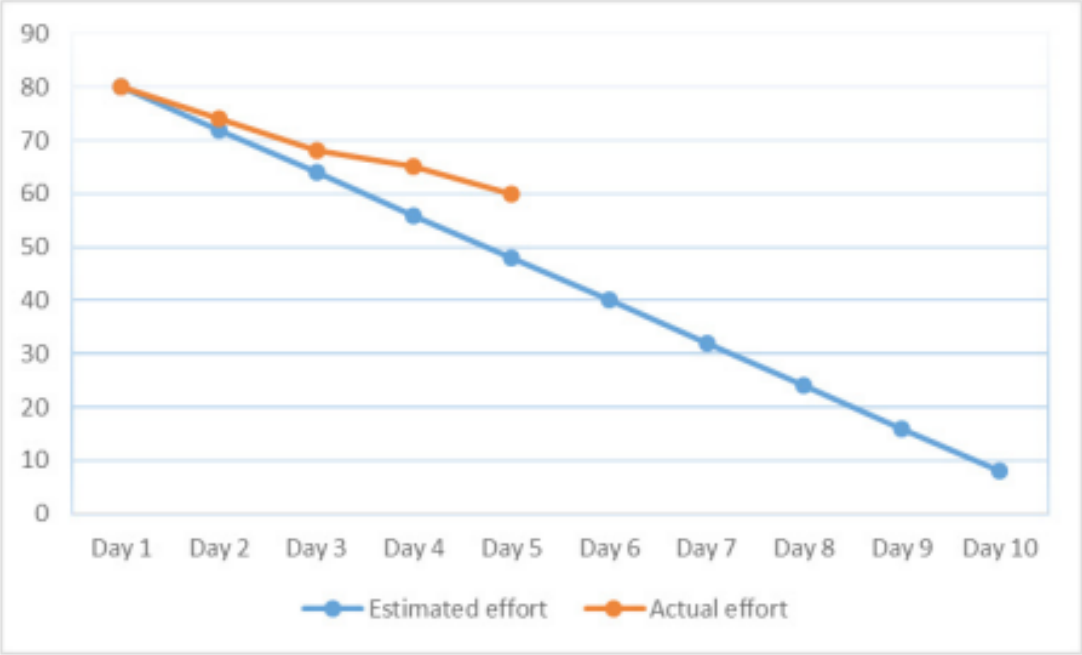
Task Name	Duration	Start Date	End Date	Status	Goal
Leave Questionnaires	3	22-3-23	24-3-23	done	Leave question work properly
Create Model	3	25-3-23	27-3-23	done	create model hr, admin, employee
Edit, view, delete employee data	5	28-4-23	1-4-23	done	View, delete, edit employee data in module.
Add leave point	3	2-4-23	4-4-23	done	Add leave balance credit, debit.
View all leave data approve and reject	4	4-4-23	7-4-23	done	View all data.
User side create	-	10-4-23	-3	Process	-

f. Agile sprint backlog

User	To do	InProgress…	Completed
Create model			Task: created model for hr, admin, employee. Estimate time: 5 days
Edit, view, delete employee data			Task: edit data, delete data, view data from admin side. Estimate time:7days
View all leaves and add balance of leave.		Task: add balance form admin side. Estimate time: 3 days	Task: add balance Estimate time: 3 days
user side create	Task: create user side	Task: create user side. Estimate time: 10 days	-

g. Agile test plan

h. Earned-value and burn charts



PROPOSED ENCASEMENT

- Improved Authentication and Authorization: Implement a more robust authentication and authorization system to ensure that only authorized personnel can access sensitive payroll data. You can use Firebase Authentication for this purpose.

- Real-time Data Synchronization: Use Firebase Realtime Database to synchronize employee data in real-time. This will ensure that any changes made to employee records are immediately reflected in the system and all authorized users have access to the most up-to-date information.
- Advanced Search and Filter Functionality: Add advanced search and filter functionality to the system so that users can quickly find specific employee records based on various criteria such as name, department, salary, etc.

- User-Friendly UI: Make the UI more user-friendly by adding clear navigation, intuitive controls, and easy-to-use forms for data entry and editing. You can use a UI framework like Verify to achieve this.

- Automated Payroll Calculations: Implement a system that automatically calculates employee payrolls based on their hourly rate, hours worked, overtime, deductions, and other relevant factors. You can use a library like Moment.js or Numeral.js to help with this.

- Reporting and Analytics: Add reporting and analytics functionality to the system so that users can generate reports on various payroll-related metrics such as total payroll expenses, average salary, employee turnover rate, etc.

- Mobile Responsiveness: Ensure that the system is mobile-responsive and can be accessed from any device, including smartphones and tablets.

- Role-Based Access Control: Implement role-based access control to ensure that different users have access to different features and functionality based on their roles and permissions.

- Integration with Third-Party Services: Integrate the payroll management system with third-party services such as payment gateways, accounting software, and HR management tools to streamline the payroll process and improve efficiency.

- Multi-Language Support: Add support for multiple languages to the system to make it accessible to a wider audience. You can use a localization library like Vue-i18n to achieve this.

CONCLUSION

In conclusion, developing an employee Leave management system in Vue.js with Firebase offers many benefits, such as real-time data synchronization, advanced search and filter functionality, apply for leave, approve and decline leave, also add leave balance, reporting and analytics, mobile responsiveness, and integration with third-party services. These enhancements can greatly improve the efficiency and accuracy of the leave process, making it easier for businesses to manage their leave and comply with legal requirements. With role-based access control and improved authentication and authorization, the system can also ensure the security of sensitive payroll data. Overall, implementing these enhancements can provide a more user-friendly and streamlined leave management system for businesses, while also offering scalability and flexibility for future growth and development.

BIBLIOGRAPHY/ REFERENCES

- https://www.w3schools.com/
- https://vuejs.org/
- https://firebase.google.com/
- https://www.geeksforgeeks.org/
- https://www.figma.com/
- https://nodejs.org/en