**A**

**PROJECT REPORT**

on

**Human Resource Management System**

Submitted in partial fulfillment of the requirements for the degree of

**BACHELOR OF TECHNOLOGY**



**Session: - Jan-May 2024**

Under Guidance of Submitted by :

Mr. Aaditya Maheshwari Vineet Agarwal(20ETCCS114)

Assistant Professor CSE Sudhanshu Dengra(20ETCCS106)

Techno India NJR Sanjay Menaria(20ETCCS098)

Institute of Technology

VIII semester,CSE

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**TECHNO INDIA NJR INSTITUTE OF TECHNOLOGY, UDAIPUR-313001**

**MAY – 2024**



Department of Computer Science and Engineering

Techno India NJR Institute of Technology, Udaipur-313001

**Certificate**

This is to certify that project work titled Human Resource Management System by Vineet Agarwal was successfully carried out in the Department of Computer Science and Engineering, TINJRIT and the report is approved for submission in the partial fulfillment of the requirements for award of degree of Bachelor of Technology in Computer Science and Engineering.

Mr. Aaditya Maheshwari Dr. Rimpy Bishnoi

Assistant Professor CSE Head of Department

Techno India NJR Dept. of CSE TINJRIT, Udaipur

Institute of Technology

Date: Date:



Department of Computer Science and Engineering

Techno India NJR Institute of Technology, Udaipur-313001

**Certificate**

This is to certify that project work titled Human Resource Management System by Sudhanshu Dengra was successfully carried out in the Department of Computer Science and Engineering, TINJRIT and the report is approved for submission in the partial fulfillment of the requirements for award of degree of Bachelor of Technology in Computer Science and Engineering.

Mr. Aaditya Maheshwari Dr. Rimpy Bishnoi

Assistant Professor CSE Head of Department

Techno India NJR Dept. of CSE TINJRIT, Udaipur

Institute of Technology

Date: Date:



Department of Computer Science and Engineering

Techno India NJR Institute of Technology, Udaipur-313001

**Certificate**

This is to certify that project work titled Human Resource Management System by Sanjay Menaria was successfully carried out in the Department of Computer Science and Engineering, TINJRIT and the report is approved for submission in the partial fulfillment of the requirements for award of degree of Bachelor of Technology in Computer Science and Engineering.

Mr. Aaditya Maheshwari Dr. Rimpy Bishnoi

Assistant Professor CSE Head of Department

Techno India NJR Dept. of CSE TINJRIT, Udaipur

Institute of Technology

Date: Date:



Department of Computer Science and Engineering

Techno India NJR Institute of Technology, Udaipur-313001

**Examiner Certificate**

This is to certify that the following student **Vineet Agarwal** of final year B.Tech. (Computer Science and Engineering), was examined for the project work titled “***Human Resource Management System”*** during the academic year 2023 – 2024 at Techno India NJR Institute of Technology, Udaipur.

**Remarks:**

**Date:**

Signature Signature

(**Internal Examiner**) (**External Examiner**)

Name :- Name :-

Designation:- Designation:-.

Department: - Department: -

Organization:- Organization:-



Department of Computer Science and Engineering

Techno India NJR Institute of Technology, Udaipur-313001

**Examiner Certificate**

This is to certify that the following student **Sudhanshu Dengra** of final year B.Tech. (Computer Science and Engineering), was examined for the project work titled “***Human Resource Management System”*** during the academic year 2023 – 2024 at Techno India NJR Institute of Technology, Udaipur.

**Remarks:**

**Date:**

Signature Signature

(**Internal Examiner**) (**External Examiner**)

Name :- Name :-

Designation:- Designation:-.

Department: - Department: -

Organization:- Organization:-



Department of Computer Science and Engineering

Techno India NJR Institute of Technology, Udaipur-313001

**Examiner Certificate**

This is to certify that the following student **Sanjay Menaria** of final year B.Tech. (Computer Science and Engineering), was examined for the project work titled “***Human Resource Management System”*** during the academic year 2023 – 2024 at Techno India NJR Institute of Technology, Udaipur.

**Remarks:**

**Date:**

Signature Signature

(**Internal Examiner**) (**External Examiner**)

Name :- Name :-

Designation:- Designation:-.

Department: - Department: -

Organization:- Organization:-

***PREFACE***

This report has been prepared to provide a comprehensive overview of the design, implementation, and outcomes of the Human Resource Management System developed using the MERN stack. This project was undertaken as part of an educational assignment aimed at applying theoretical knowledge from various computer science and information technology courses to solve real-world business problems efficiently through the use of technology.

The primary motivation behind this project was to address the challenges faced by HR departments in managing employee data, recruitment processes, leave requests, and other administrative functions. By automating these processes, our goal was to showcase how a well-designed software system can significantly contribute to organizational efficiency.

We would like to express our gratitude to our professor and mentors for their invaluable guidance, and to our peers for their insights and suggestions. Their support was crucial in overcoming the challenges we encountered during the development of this system.

This preface, along with the subsequent sections, outlines the efforts and insights garnered throughout the course of this project, presenting both the technical and practical aspects of our work. We hope that this report will serve as a helpful resource for anyone interested in developing or studying similar systems.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

TECHNO INDIA NJR INSTITUTE OF TECHNOLOGY, UDAIPUR-313001

***ACKNOWLEDGMENT***

We take this opportunity to record our sincere thanks to all who helped us to successfully complete this work. Firstly, we are grateful to our **supervisor Mr. Aaditya Maheshwari** for his invaluable guidance and constant encouragement, support and most importantly for giving us the opportunity to carry out this work.

We would like to express our deepest sense of gratitude and humble regards to our

**Head of Department Dr. Rimpy Bishnoi** for giving invariable encouragement in our endeavors and providing necessary facilities for the same. Also, a sincere thanks to all faculty members of CSE, TINJRIT for their help in the project directly or indirectly.

Finally, we would like to thank my friends for their support and discussions that have proved very valuable for us. We are indebted to our parents for providing constant support, love and encouragement. We thank them for the sacrifices they made so that we could grow up in a learning environment. They have always stood by us in everything we have done, providing constant support, encouragement and love.

**Vineet Agarwal, 20ETCCS114**

**Sudhanshu Dengra, 20ETCCS106**

**Sanjay Menaria, 20ETCCS098**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

TECHNO INDIA NJR INSTITUTE OF TECHNOLOGY, UDAIPUR-313001

# ***CONTENTS***

|  |  |
| --- | --- |
| **Name of Contents** | **Page no.** |
| **Certificates** | 1-3 |
| **Examiner Certificates** | 4-6 |
| **Preface** | 7 |
| **Acknowledgement** | 8 |
| **Contents** | 9-10 |
| **List of Figures** | 12 |
| **List of Tables** | 13 |
| [**CHAPTER -1: Introduction**](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.r9vtdlagspzr)  [1.1. Problem Statement](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.5fdjbu3s7g0p)  [1.2. Objectives and Scope](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.wxtt9auwqpyr)  [1.3. Working of HRMS](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.lm5h9fiveiwa)  [1.4. Methodology](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.roy7rhqtty04)  [1.5. Resources and Limitations](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.b5e454od3vks)  [1.6. Conclusion](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.xpfa6is3jj3w) | 14  14  14  14  14  15  15 |
| [**CHAPTER - 2: Tools & Technologies Consumed**](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.o0mfw750x3j9)  [2.1. Code Editor: Microsoft VS Code](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.8mm4re945ehz)  [2.2. Version Control: Git, GitHub](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.ybdxl2dfgrk7)  [2.3. APIs Testing (RESTful): Postman](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.kaxh9culr2v7)  [2.4. Frontend: HTML, CSS, React](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.1vfv3bz46cey)  [2.5. Backend: Node, Express](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.hotneac747ka)  [2.6. Database: MongoDB](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.3yus2utwg0uy)  [2.7. Web Automation using Chrome Extension](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.qc2tmwqgj18o) | 16  16  16  17  17  18  18  19 |
| [**CHAPTER - 3: Project Structure**](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.msm3p1gc8nkd)  3.1. Repository Structure: Monolithic  [3.2. MERN Application: Structure](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.x8m8jx4yt3cu)  [3.2.1. Client-side (React.js)](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.vcraw42ad07n)  [3.2.2. Server-side (Node.js with Express.js)](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.oztqv737fwyb)  [3.3. List of packages used in project](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.5sgplowor0vb)  3.3.1. Client-side packages (React.js)  [3.3.2. Server-side (Node.js with Express.js)](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.q8u99vr3dx8t)  3.4. MongoDB Models (Database Structure) | 20  20  20  20  23  24  24  26  27 |
| [**CHAPTER - 4: Features & Functionality**](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.q5gfp6i9p25k)  [4.1. User Authentication](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.d43yzy6bn49)  [4.2. Employee Management](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.kgpnq67o1iyu)  [4.3. Payroll System](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.y267jkxxh3x)  [4.4. Leave, Loan, and Bonus Request Management](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.3l3ucmj8fk19)  [4.5. Ticket Management](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.ena4s0r2ezs6)  [4.6. Secure and Efficient Database Management](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.pfezjw8z4rmi)  [4.7. Statistical Analysis](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.jnujlitp8a16)  [4.8. Google Calendar Integration](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.j0uydfw9f7r4)  [4.9. News Section](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.t09yp689yd7d) | 29  29  30  31  32  32  33  34  35  35 |
| [**CHAPTER - 5: Future Scopes & Possibilities**](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.w2fzhwkqiwjv)  [5.1. Mobile Application Development](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.ohtupzgzihtt)  [5.2. Chrome Extension for Attendance Management](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.enqxaodrx3gy)  [5.3. Enhanced Data Analytics](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.lwqnv8gb30h3)  [5.4. Enhanced User Interface](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.lt0vk0shgbyf)  [5.5. Third-Party APIs Integration](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.njk1wr7z056j)  [5.6. Hiring & User Onboarding Management](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.yx6k2j9hkrob) | 37  37  37  37  38  38  38 |
| [**Project Timeline**](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.zfz19rfsvd8q) | 39 |
| [**Conclusion**](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.u1ada9lk5dgh) | 40 |
| [**References/Bibliography**](https://docs.google.com/document/d/1agLAHIJi1yeHXjrpt5nP-qu0CEL_m7nQ/edit#heading=h.wvjuacuj021n) | 41 |

# ***List Of Figures***

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Figure Name** | **Page No.** |
| Fig 1 | Client Directory Structure | 21 |
| Fig 2 | Server Directory Structure | 21 |
| Fig 3 | Login Page | 29 |
| Fig 4 | Admin dashboard to see employees | 29 |
| Fig 5 | Employee Management | 30 |
| Fig 6 | Payroll Management | 30 |
| Fig 7 | Employee Request | 31 |
| Fig 8 | Employee Tickets | 32 |
| Fig 9 | Database | 33 |
| Fig 10 | Admin Dashboard | 33 |
| Fig 11 | Google Calendar | 34 |
| Fig 12 | Employee News | 35 |
| Fig 13 | Gantt Chart | 39 |

# ***List Of Tables***

|  |  |  |
| --- | --- | --- |
| **Table No.** | **Table Name** | **Page No.** |
| 1. | Client-Side implemented Packages | 24 - 25 |
| 2. | Server-Side implemented Packages | 25 – 26 |
| 3. | Models in Database | 26 – 27 |

# ***List of Abbreviations***

|  |  |
| --- | --- |
| **Abbreviations** | **Full form** |
| RBAC | Role Based Access Control |
| AES | Advanced Encryption Standard |
| HTTP | HyperText Transfer Protocol |
| API | Application Program Interface |
| UI | User Interface |
| JSON | JavaScript Object Notation |
| HRMS | Human Resource Management System |
| DOM | Document Object Model |
| VS | Visual Studio |
| NPM | Node Package Manager |
| DBMS | Database Management System |
| UUID | Universally Unique Identifier |
| CORS | Cross Origin Resource Sharing |
| MERN | Mongo Express React Node |
| CRUD | Create Read Update Delete |
| JWT | JSON Web Token |
| UX | User Experience |
| OAuth | Open Authorization |

# CHAPTER -1: ***INTRODUCTION***

## Problem Statement

Managing human resources in organizations involves a multitude of tasks ranging from employee management to payroll processing and ticket handling. However, the reliance on manual methods often results in inefficiencies and errors, posing significant challenges to HR departments. The absence of a centralized system exacerbates these issues, making it difficult to maintain accuracy and consistency across various HR processes.

## Objectives and Scope

The primary objective of this project is to address the shortcomings of manual HR management by developing a comprehensive Human Resource Management System (HRMS). This system will automate and streamline key HR processes, including user authentication, employee data management, payroll integration, leave and bonus applications, and ticket handling. By doing so, it aims to enhance the efficiency and effectiveness of HR operations within organizations of varying sizes and structures.

## Working of HRMS

Our HRMS functions as a centralized platform that automates essential HR tasks, such as user authentication to ensure data security, employee management for maintaining accurate personnel records, payroll processing for timely and accurate compensation, leave and benefits management to facilitate employee requests, and ticket handling for resolving inquiries and concerns. Moreover, the integration with Google Calendar allows for seamless scheduling and reminders, further enhancing organizational efficiency and productivity.

## Methodology

The project adheres to a structured methodology encompassing various stages of development, including requirements analysis, system design, database design, development, testing, and deployment. This approach ensures that the HRMS is systematically designed, implemented, and validated to meet the specified requirements and standards. Furthermore, it enables iterative improvements and refinements based on feedback and testing results.

## Resources and Limitations

The successful implementation of the HRMS relies on the availability of adequate hardware and software resources. Hardware resources include a web-hosted server to facilitate access to the application and user devices for interaction. Software components comprise technologies such as ReactJs for front-end development, NodeJs for server-side scripting, MongoDB for database management, and Postman for API testing.

## Conclusion

In conclusion, the HRMS represents a significant advancement in HR management, offering a robust solution to streamline processes, improve accuracy, and enhance efficiency. By automating critical tasks and centralizing HR operations, it empowers organizations to optimize resource utilization, minimize errors, and focus on strategic initiatives. With its user-friendly interface, scalability, and adaptability, the HRMS is poised to revolutionize HR operations and drive organizational success in the digital age.

# CHAPTER - 2: ***TOOLS & TECHNOLOGIES CONSUMED***

## Code Editor: Microsoft VS Code

* **Overview:** Microsoft Visual Studio Code (VS Code) is a versatile and feature-rich code editor that facilitates efficient development workflows.
* **Use Case:**
  + **Code Editing:** VS Code provides an intuitive interface for writing and editing code, with features such as syntax highlighting, code completion, and automatic formatting.
  + **Integrated Terminal:** It includes a built-in terminal that allows developers to execute commands and run scripts easily.
  + **Version Control Integration:** VS Code seamlessly integrates with version control systems like Git, providing tools for managing repositories, reviewing changes, and resolving conflicts.
  + **Extensions:** Its vast library of extensions enhances productivity by offering additional functionalities for language support, debugging, and project management.

## Version Control: Git, GitHub

* **Overview:** Git is a distributed version control system that enables collaborative development and version tracking of source code. GitHub is a web-based platform built on top of Git, providing hosting and collaboration features for Git repositories.
* **Use Case:**
  + **Version Tracking:** Git allows developers to track changes to their codebase over time, facilitating collaboration and ensuring version control. It enables developers to create branches, commit changes, and merge code seamlessly.
  + **Collaboration:** GitHub serves as a centralized platform for hosting Git repositories, allowing multiple developers to work on the same codebase simultaneously. It provides features such as pull requests, code reviews,issue tracking, promoting collaboration & transparency.
  + **Remote Repository Management:** GitHub enables developers to host their Git repositories remotely, providing backup and accessibility from anywhere with an internet connection. It offers features for repository management, such as branching, tagging, and access control.

## APIs Testing (RESTful): Postman

* **Overview:** Postman is a comprehensive API development and testing platform that simplifies the process of designing, testing, and documenting APIs. It offers a user-friendly interface and a range of features for creating and managing API requests.
* **Use Case:**
  + **Request Building:** Postman enables developers to create and send HTTP requests to RESTful APIs effortlessly. Its intuitive interface allows for easy configuration of request parameters such as headers, body, and authentication.
  + **Testing Workflows:** It supports the creation of test suites and scripts for automating API testing workflows. Developers can define assertions to verify API responses and ensure functionality and reliability.
  + **Collaboration:** Postman provides features for team collaboration, allowing multiple team members to work on API development and testing simultaneously. It facilitates sharing of collections, environments, and test scripts, promoting collaboration and knowledge sharing.
  + **Documentation:** Postman offers tools for generating comprehensive API documentation from collections, making it easier for developers to understand and use APIs.

## Frontend: HTML, CSS, React

* **Overview:** HTML, CSS, and React collectively serve as the cornerstone of contemporary web development, facilitating the crafting of immersive, dynamic, and responsive user interfaces. While HTML establishes the structural backbone, CSS injects aesthetic appeal and visual finesse. React, with its robust component-based architecture, not only elevates interactivity but also fosters the creation of reusable UI elements, thereby revolutionizing user experiences and streamlining development workflows.
* **Use Case:**
  + **HTML (Hypertext Markup Language):** HTML is used for structuring web pages, defining the layout, and organizing content elements such as headings, paragraphs, and lists.
  + **CSS (Cascading Style Sheets):** CSS is responsible for styling web pages, controlling visual aspects such as colors, fonts, and layout. It allows for customization and branding, enhancing the visual appeal and user experience.
  + **React(version > 17.0.0):** React is a JavaScript library for building user interfaces, utilizing a component-based architecture. It enables the creation of reusable UI components, simplifying development and maintenance. React's virtual DOM optimizes rendering performance, ensuring smooth and efficient updates to the UI.

## Backend: Node, Express

* **Overview:** Node.js and Express form a powerful combination for server-side development, allowing developers to build scalable and performant web applications and APIs. Leveraging non-blocking I/O and middleware support, this duo empowers developers to create efficient and robust server architectures, facilitating seamless handling of concurrent requests and optimizing performance. Chrome extensions offer a versatile platform for developers to innovate and deliver tailored solutions to address specific user needs or preferences.
* **Use Case:**
  + **Node.js(version > 17.0.0):** Node.js enables server-side JavaScript execution, providing an event-driven, non-blocking I/O model. It allows developers to build lightweight and efficient server applications capable of handling concurrent requests.
  + **Express(version > 4.19.0):** Express is a minimalist web framework for Node.js, providing essential features for routing, middleware integration, and request handling. It simplifies the process of building RESTful APIs and web servers, reducing boilerplate code and promoting code maintainability.

## Database: MongoDB

* **Overview:** MongoDB is a flexible and scalable NoSQL database management system designed for modern application development. It stores data in a document-based format, offering flexibility and scalability. With its dynamic schema and powerful querying capabilities, MongoDB empowers developers to adapt quickly to evolving data requirements and efficiently manage complex data structures.
* **Use Case:**
  + **Document Storage:** MongoDB stores data in flexible JSON-like documents, allowing for the storage of complex and hierarchical data structures. It accommodates dynamic schemas, facilitating agile development and iteration.
  + **Scalability:** MongoDB supports horizontal scaling through sharding, allowing developers to distribute data across multiple servers to accommodate growing workloads. It ensures high availability and fault tolerance, enhancing application scalability and resilience.
  + **Querying and Aggregation:** MongoDB provides powerful querying and aggregation capabilities, allowing developers to perform complex data operations efficiently. It supports a rich query language and aggregation framework, enabling flexible data analysis and reporting.

## Web Automation using Chrome Extension

* **Overview:** Chrome extensions are small software programs that customize the browsing experience and add functionality to the Google Chrome web browser. They provide a lightweight and platform-independent solution for automating repetitive tasks or workflows within the web browser environment, offering advantages over standalone automation tools or browser-specific scripting languages.
* **Use Case:**
  + **Web Scraping Tool:** A Chrome extension designed for web scraping allows users to extract specific data from web pages and store it locally or send it to a server for further processing.
  + **Productivity Tracker:** A Chrome extension focused on productivity tracking enables users to monitor their browsing habits and analyze their time spent on different websites.
  + **Content Recommendation System:** A Chrome extension implementing a content recommendation system provides users with personalized recommendations based on their browsing history and preferences.

# CHAPTER - 3: ***Project Structure***

## Repository Structure : Monolithic

The monolithic repository combines frontend and backend code in one place, simplifying version control and promoting collaboration. However, it can become complex and challenging to scale as the project grows. Agile methodology is suitable here for iterative development and quick adaptations to changes. It facilitates continuous collaboration and allows for incremental improvements, aligning well with the monolithic repository's collaborative nature.

## MERN Application: Structure

In a MERN (MongoDB, Express.js, React.js, Node.js) stack application, the frontend and backend are decoupled, communicating with each other via RESTful APIs.

### Client-side (React.js):

* **Working overview:** The client-side of the application is built using React.js. It typically consists of components that render the UI and handle user interactions.
  + When a user interacts with the UI, such as clicking a button or submitting a form, React.js components trigger events or state updates.
  + These events or state updates may result in HTTP requests being sent to the server to perform actions like fetching data, submitting data, updating data, or deleting data.
  + React.js components may use libraries like axios or the built-in fetch API to make HTTP requests to the server's RESTful APIs.
  + Once the server responds to the HTTP requests, React.js components update the UI accordingly, displaying fetched data or providing feedback to the user.
* **Directory Structure:** Let's break down the project folder structure for a React-based application along with the purpose of each directory and file:
  + ***build(\*folder):*** This directory typically contains the production-ready code generated by the build process. It's not typically present in the initial project setup but gets created when you build your React app for deployment.
  + ***node\_modules(\*folder):*** This directory contains all the dependencies (libraries) your project utilizes. It's managed by npm (Node Package Manager) or Yarn and is not usually included in version control because it's generated based on the dependencies listed in your package.json file.
  + ***public(\*folder):*** This directory contains static assets such as HTML files, images, fonts, etc. The index.html file in this directory is the entry point of your React application. It's where you typically link your bundled JavaScript files.
  + ***src(\*folder):*** This is where most of your development work will happen. It contains the source code of your React application.
    - ***components(\*folder):*** This directory holds reusable React components. Organizing your components into a separate directory helps keep your project structured and makes it easier to manage and reuse components across your application.
    - ***app.js(\*file):*** This file typically serves as the main entry point for your React application. It might contain the root component of your application, where you render other components.
    - ***index.js(\*file):*** This file is usually the entry point for Webpack or other bundlers. It's where you typically render your root component (e.g., App) into the DOM.
  + ***.eslintcache(\*file):*** This file contains cached data generated by ESLint, a tool for identifying and reporting on patterns found in ECMAScript/JavaScript code. It's used to improve performance when running ESLint.
  + ***package.json:*** This file is a manifest for your project and lists the project's dependencies, scripts, and other metadata. It's also used by npm or Yarn to install dependencies and manage scripts for tasks like building, testing, and running your application.

That's a basic overview of the project structure for a React-based application. It's a common setup that provides a foundation for building scalable and maintainable React applications. As your project grows, you might introduce additional directories or files to accommodate specific needs or patterns.

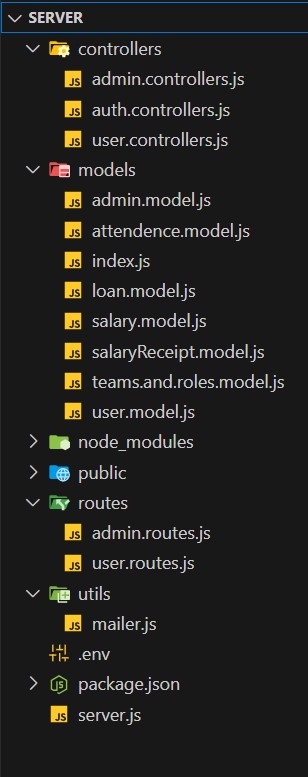
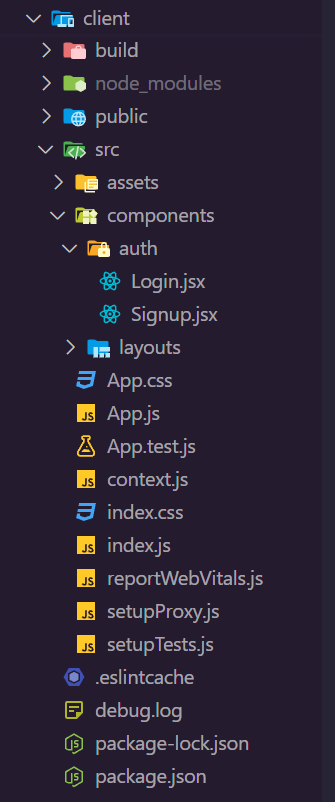
****

Figure : 1 Figure : 2

### Server-side (Node.js with Express.js):

* **Working Overview:** The server-side of the application is built using Node.js with Express.js, a web application framework for Node.js.
  + Express.js handles incoming HTTP requests from the client and routes them to the appropriate handlers (controllers) based on the request method and URL.
  + Controllers (located in the controllers directory) contain the business logic of the application. They interact with the database (MongoDB) through models (located in the models directory) to perform CRUD operations.
  + When a client makes a request to the server, Express.js routes the request to the corresponding controller, which processes the request, interacts with the database if necessary, and sends a response back to the client.
  + The server typically exposes RESTful endpoints (routes) for various operations such as fetching data, creating data, updating data, and deleting data. These endpoints are accessed by the client-side of the application via HTTP requests.
* **Directory Structure:** Let's delve into the folder structure for a Node.js-based application and the purpose of each directory and file:
  + ***controllers(\*folder):*** This directory typically contains the controller files, which handle the business logic of your application. Controllers are responsible for processing incoming requests, interacting with models, and sending responses back to clients.
  + ***models(\*folder):*** This directory holds the model files, which represent the structure and behavior of your application's data. Models often interact with a database or other data sources to perform CRUD (Create, Read, Update, Delete) operations.
  + ***node\_modules(\*folder):*** Just like in the React project structure, this directory contains all the dependencies (Node.js modules) required by your application. It's managed by npm or Yarn.
  + ***public(\*folder):*** This directory stores static assets such as HTML files, CSS stylesheets, client-side JavaScript files, images, etc. These assets

are served directly to clients and can be accessed publicly.

* + ***routes(\*folder):*** This directory houses the route files, which define the endpoints of your application's API. Routes handle incoming requests from clients and route them to the appropriate controllers for processing.
  + ***utils(\*folders):*** This directory contains utility files or helper functions that are used across your application. Utilities can include functions for data validation, formatting, error handling, etc.
  + ***.env(\*folder):*** This file is typically used to store environment variables such as API keys, database connection strings, and other configuration values. Environment variables keep sensitive information separate from your codebase and are accessed using libraries like dotenv.
  + ***app.js(\*file):*** This file is the main entry point of your Node.js application. It typically sets up the Express.js application instance, middleware, routes, and other configurations necessary for your application to run.
  + ***server.js(\*file):*** This file is responsible for starting the server and listening for incoming requests. It usually imports the app.js file and starts the server on a specified port.
  + ***package.json(\*file):*** Just like in the React project structure, this file is a manifest for your project and lists its dependencies, scripts, and other metadata. It's also used by npm or Yarn to install dependencies and manage scripts for tasks like running your application.

This folder structure provides a solid foundation for building Node.js applications, separating concerns like routing, business logic, and data modeling. As your application grows, you may organize your files further or introduce additional directories based on the specific needs of your project.

## List of packages used in project

### Client side packages (React.js)

|  |  |
| --- | --- |
| **Package Name** | **Uses & Functionality** |
| @testing-library/jest-dom | Testing DOM elements. Provides custom jest matchers for testing DOM elements. Simplifies testing by offering custom matchers for DOM. |
| @testing-library/react | Provides Unit and integration testing library for React components. Encourages realistic testing scenarios by simulating user interactions |
| @testing-library/user-event | Provides utilities for simulating user events like typing, clicking, etc.  Enables more realistic testing scenarios by simulating user interactions accurately |
| axios | Promise-based HTTP client for making requests. Communicating with backend APIs. |
| bcryptjs | Library for securely hashing passwords for storage in databases. Provides a secure way to hash passwords with bcrypt algorithm. |
| bootstrap | Framework for building responsive and mobile-first websites.  Provides Prototyping or building UIs for web apps. |
| chart.js | Library for creating interactive and customizable charts and graphs,Visualizing data in web applications. |
| classnames | Utility for conditionally joining CSS class names,Dynamic styling of React components. Simplifies conditionally applying CSS classes to elements. |
| http-proxy-middleware | Middleware for proxying HTTP requests in development, Proxying requests during development. Facilitates development by allowing easy proxying of requests to backend servers. |
| react | JavaScript library for building user interfaces. Developing web applications with React. |
| react-calendar | React component for displaying calendars.  Displaying and managing dates in applications. |
| react-chartjs-2 | React wrapper for Chart.js library. Integrating Chart.js with React components. |
| react-dom | Entry point for DOM-specific methods in React applications. Rendering React components in the DOM. |
| react-router-dom | Declarative routing for React applications. Handling navigation in single-page apps. |
| react-scripts | Command-line tools and scripts for creating, building, and running React applications.  Helps in managing React application development. |
| react-spring | Animation library for React. Provides smooth, performant animations with a natural feel, supports complex animation effects |
| react-tooltip | Tooltip library for React components. Simplifies adding tooltips to React components, customizable tooltip appearance and behavior. |
| toasted-notes | Notifying users about events or alerts. Provides a simple API for displaying toast notifications in React applications |
| uuid | Creating unique identifiers in applications. Provides a reliable way to generate universally unique identifiers (UUIDs). |
| web-vitals | Library for measuring and reporting core web vital metrics. Provides tools for measuring and analyzing core web vital metrics, helps identify areas for optimization. |
| yarn | Managing project dependencies. Provides fast and reliable package installation and management, deterministic dependency resolution. |

Table - 1: Packages of client side application

### Server-side (Node.js with Express.js)

|  |  |
| --- | --- |
| **Package Name** | **Uses & Functionality** |
| axios | Promise-based HTTP client. Making HTTP requests & communicating with backend APIs. |
| bcryptjs | Securely hashing passwords for storage in databases. Provides a secure way to hash passwords with bcrypt algorithm. |
| cors | Middleware for enabling Cross-Origin Resource Sharing in Web Applications. Allows controlled access to resources from different origins. |
| dotenv | Simplifying configuration management by loading environment variables from a file. Managing application configuration settings. |
| express | Providing robust features for building web applications. Helps in developing server-side applications with Node.js and Express.js. |
| jsonwebtoken | Enables secure authentication and authorization using JSON Web Tokens. |
| mongoose | Simplifies interaction with MongoDB databases, provides schema-based solution for data modeling. |
| multer | Simplifies handling of multipart/form-data requests, supports file uploads in Node.js applications. |
| nodemailer | Providing a simple API for sending emails from Node.js applications. |
| nodemon | Utility for automatically restarting Node.js apps. Facilitates development by automatically restarting the server when changes are made to files. |

Table - 2: Packages of server side application

## MongoDB Models(Database Structure)

|  |  |
| --- | --- |
| **Model** | **Structure** |
| Loan Model | {  reqId: { type: String },  empId: { type: String },  date: { type: String },  empName: { type: String },  gender: { type: String },  empRole: { type: String },  empTeam: { type: String },  empEmail: { type: String },  loanNote: { type: String },  loanReason: { type: String },  modeOfRepayment: { type: String },  timePeriod: { type: String },  amount: { type: String },  loanRepaid: { type: Boolean },  } |
| Salary Model | {  empId: { type: String },  empName: { type: String },  basicPay: { type: String },  totalLeaves: { type: String },  travelAllowance: { type: String },  medicalAllowance: { type: String },  bonus: { type: String },  salary: { type: String },  } |
| Admin Model | {  email: { type: String, required: true, unique: true },  password: { type: String, required: true, minlength: 5 },  name: { type: String },  role: { type: String },  leaveRequests: { type: Array },  bonusRequests: { type: Array },  loanRequests: { type: Array },  } |
| Salary Receipt Model | {  empId: { type: String },  empName: { type: String },  currentSalary: { type: String },  monthlyReceipts: [{ type: String }],  } |
| Teams & Roles Model | {  teamNames: [{ type: String }],  roleNames: [{ type: String }],  } |
| User Model | {  email: { type: String, required: true, unique: true },  password: { type: String, required: true, minlength: 8 },  name: { type: String },  gender: { type: String },  address: { type: String },  phoneNo: { type: String },  role: { type: String },  team: { type: String },  objective: { type: String },  skills: { type: String },  doj: { type: String },  notification: [],  alert: [],  active: { type: Boolean, default: true },  } |

Table -3 : Models in database

# CHAPTER - 4: ***Features & Functionality***

Traditional HR processes often involve manual tasks, paper-based documentation, and fragmented systems, leading to inefficiencies, errors, and delays. To address these challenges and meet the evolving needs of modern workplaces, the integration of technology into HR management has become imperative.

Our MERN stack-based HR management system represents a forward-thinking approach to HR administration, leveraging cutting-edge technologies to streamline processes, enhance data security, and provide actionable insights for informed decision-making. Built on the MERN (MongoDB, Express.js, React, Node.js) stack, our system offers a comprehensive suite of features designed to meet the diverse needs of HR departments and employees alike.

In this report, we will explore the key features and functionalities of our MERN stack-based HR management system, highlighting its capabilities in areas such as user authentication, employee management, payroll processing, leave and benefits administration, ticket management, and more. By understanding the intricacies of each feature and its potential impact on HR operations, stakeholders can gain valuable insights into the benefits and advantages offered by our innovative solution.

Through the seamless integration of technology, our MERN stack-based HR management system aims to revolutionize HR administration, empower HR professionals to focus on strategic initiatives, and create a more efficient, transparent, and engaging workplace environment for all employees.

## User Authentication

* **Description:** User authentication ensures that only authorized individuals can access the system by verifying their identity through unique credentials.
* **Implementation:** Utilize authentication protocols such as JWT (JSON Web Tokens) or OAuth to securely manage user sessions and authenticate requests. Employee bcrypt for password hashing to enhance security.
* **Benefits:**
  + Ensures data privacy and system integrity by preventing unauthorized access.
  + Provides a seamless and secure login experience for employees and administrators.

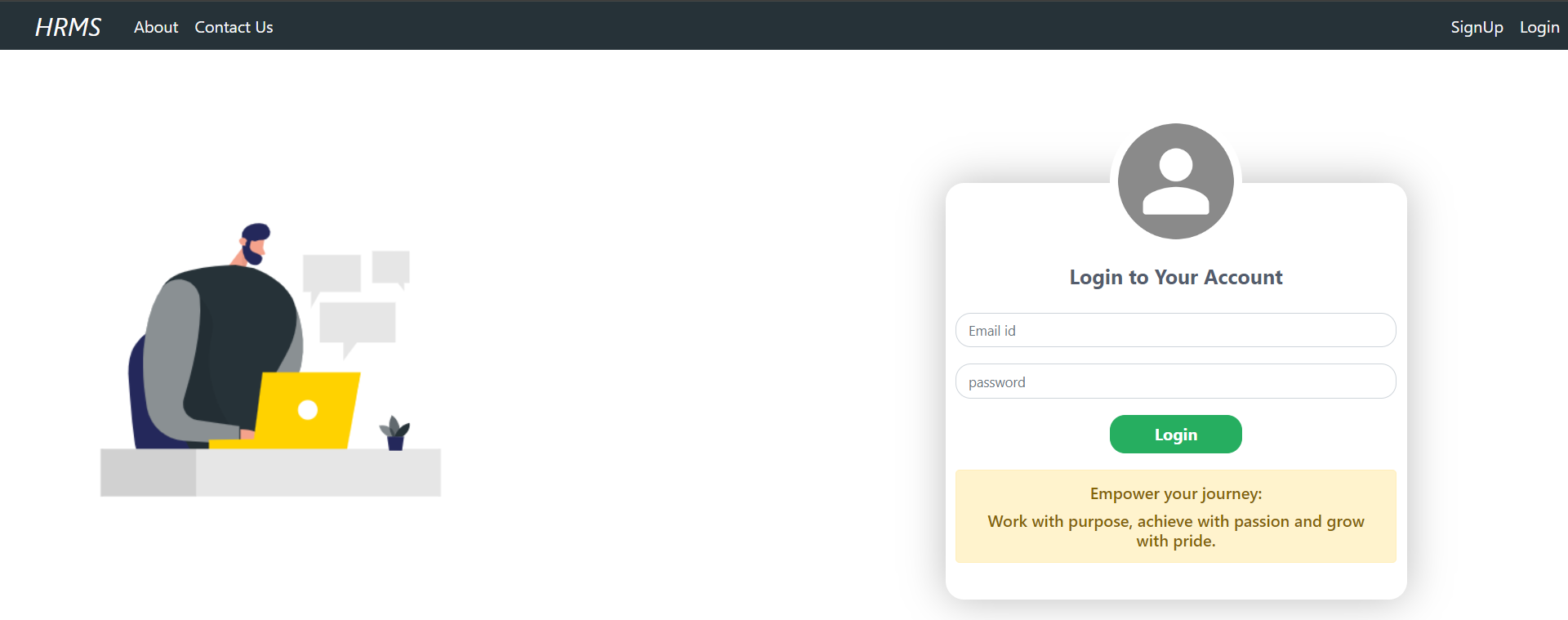


Figure: 3

## Employee Management

* **Description:** Employee management functionality enables the addition, editing, and deletion of employee records, maintaining a centralized database of workforce information.
* **Implementation:** Develop CRUD (Create, Read, Update, Delete) operations to interact with the employee database. Utilize form validation to ensure data accuracy and integrity.
* **Benefits:**
  + Facilitates efficient workforce management by centralizing employee information.
  + Streamlines HR processes such as recruitment, onboarding, and performance evaluation.

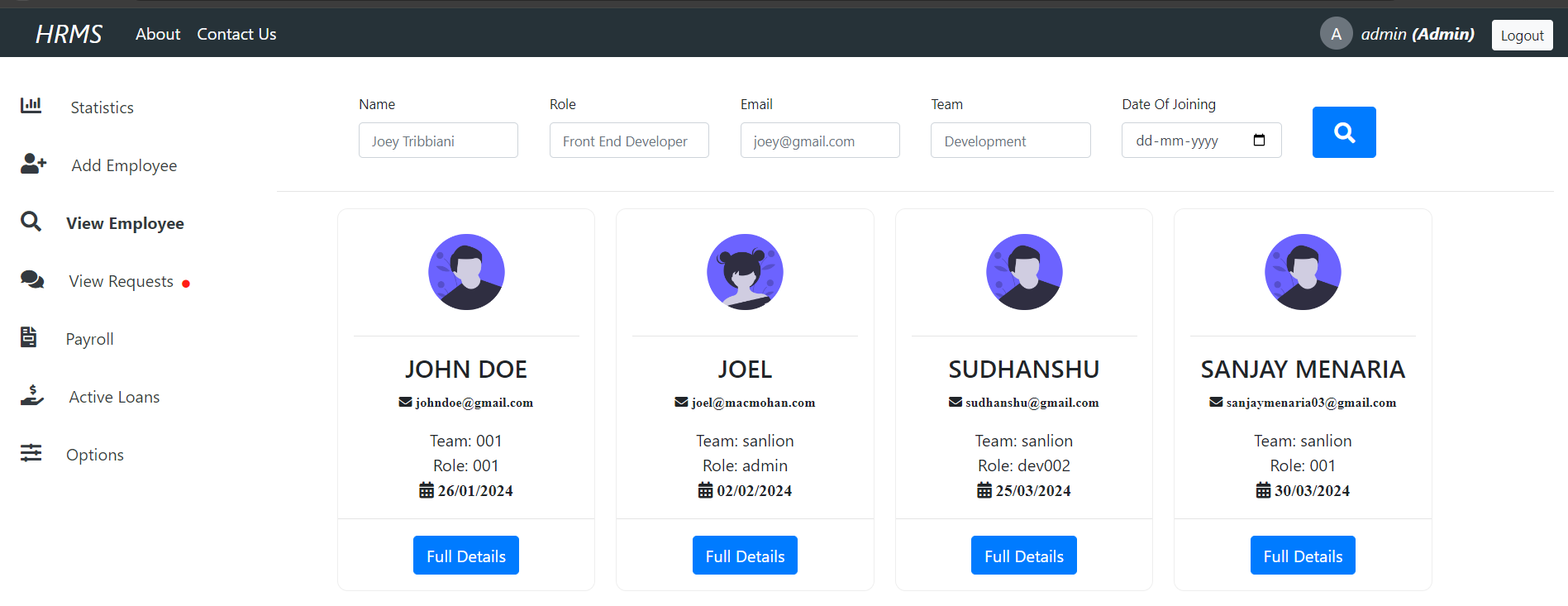


Figure: 4

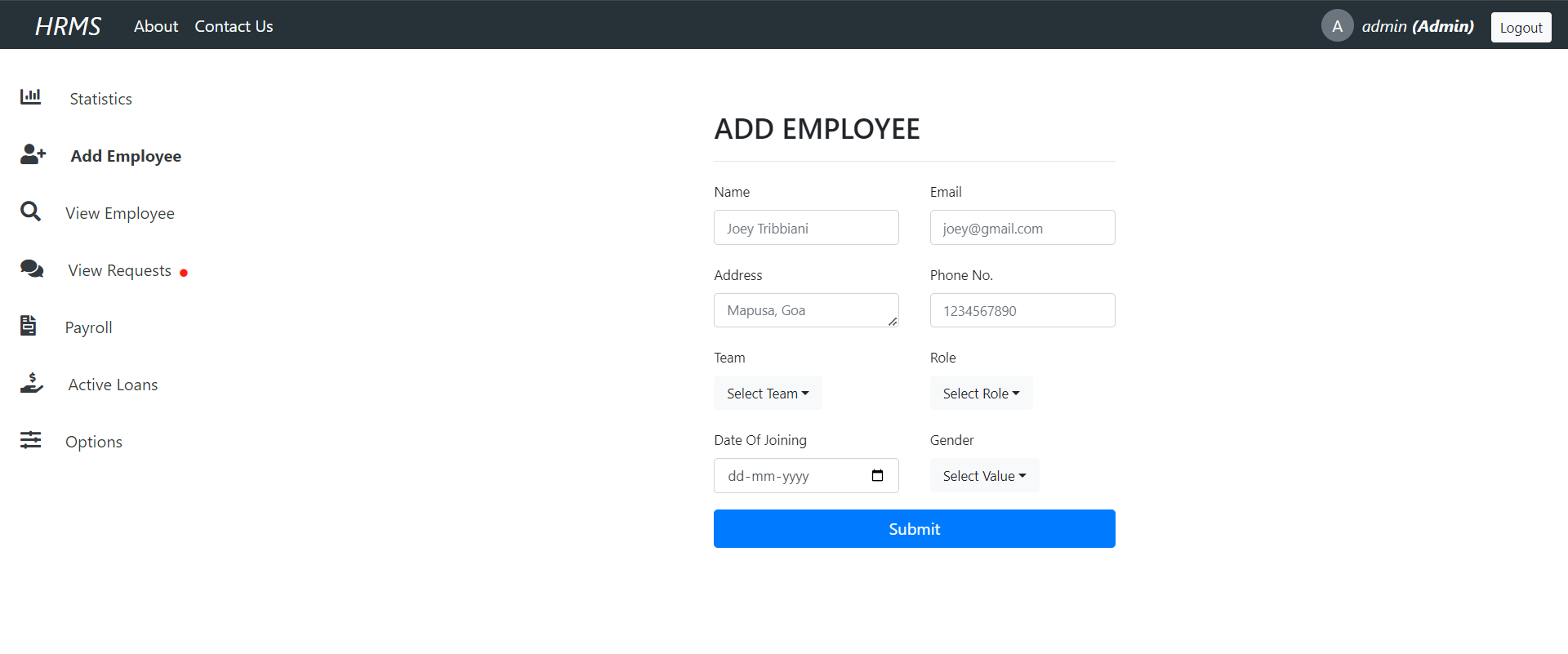


Figure: 5

## Payroll System

* **Description:** The payroll system automates salary processing, ensuring accurate and timely payment to employees. Also it provides admin facility to manage salaries and their calculations based on leaves and bonus requests to make final salary receipt.
* **Implementation:** Integrate payroll software or develop custom payroll modules to calculate salaries based on attendance, deductions, and bonuses. Implement role-based access control to restrict access to sensitive payroll data.
* **Benefits:**
  + Reduces manual errors and processing time associated with manual payroll management.
  + Enhances employee satisfaction by timely & accurate salary payments.



Figure: 6

## Leave, Loan, and Bonus Request Management

* **Description:** Leave, loan, and bonus management streamlines processes related to leave requests, loan applications, and bonus disbursements, providing employees with a centralized platform to manage these aspects of their employment.
* **Implementation:** Develop forms or workflows for employees to submit leave, loan, or bonus requests within the HRMS. Implement approval workflows and notifications to notify relevant stakeholders of pending requests for leaves, loans, or bonuses.
* **Benefits:**
  + Improves efficiency in request management by automating approval workflows for leave, loan, and bonus requests.
  + Enhances employee satisfaction by providing transparent and accessible processes for managing leaves, loans, and bonuses.

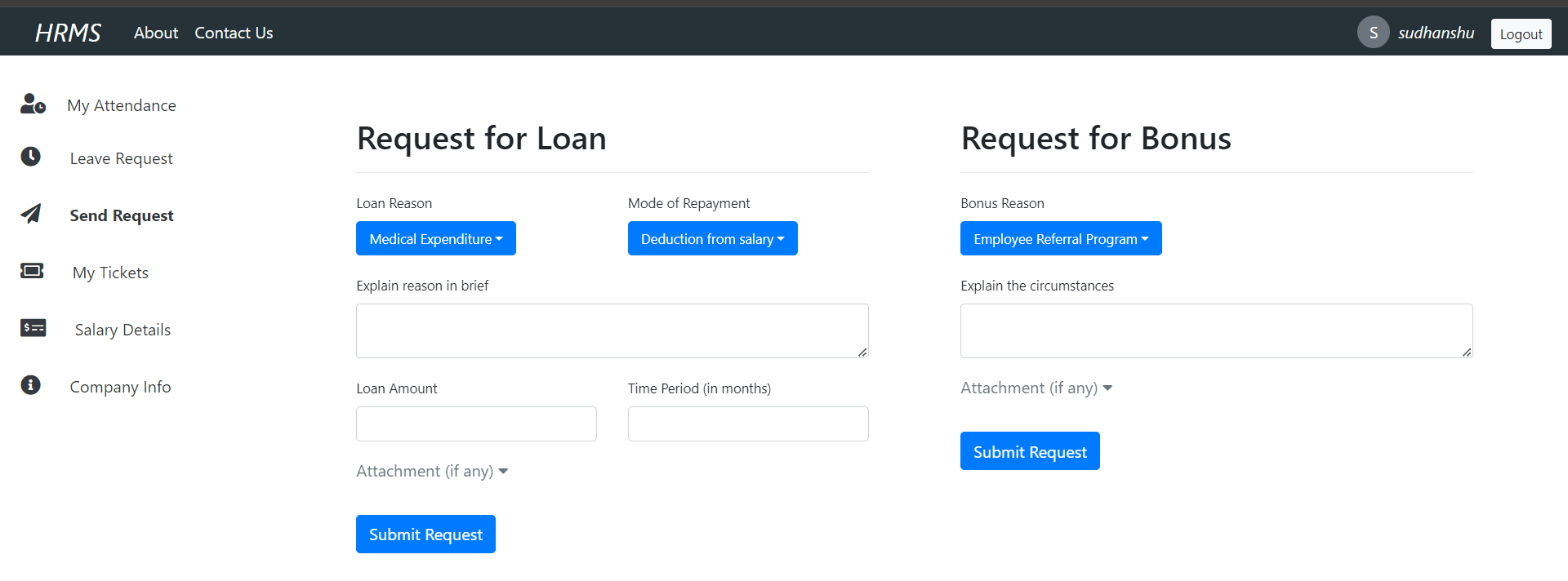


Figure: 7

## Ticket Management

* **Description:** Ticket management provides a centralized system for employees to submit queries or requests for HR-related assistance, including issues related to leave, loans, and bonuses.
* **Implementation:** Develop a comprehensive ticketing system within the HRMS with functionalities for ticket creation, assignment, tracking, and resolution. Implement email notifications to keep users informed about the status updates of their tickets, including those related to leave, loan, and bonus inquiries.
* **Benefits:**
  + Ensures timely resolution of employee queries and issues, including those related to leave, loans, and bonuses.
  + Provides visibility into HR support processes and performance metrics, allowing stakeholders to track and analyze the resolution of leave, loan, and bonus-related queries effectively.

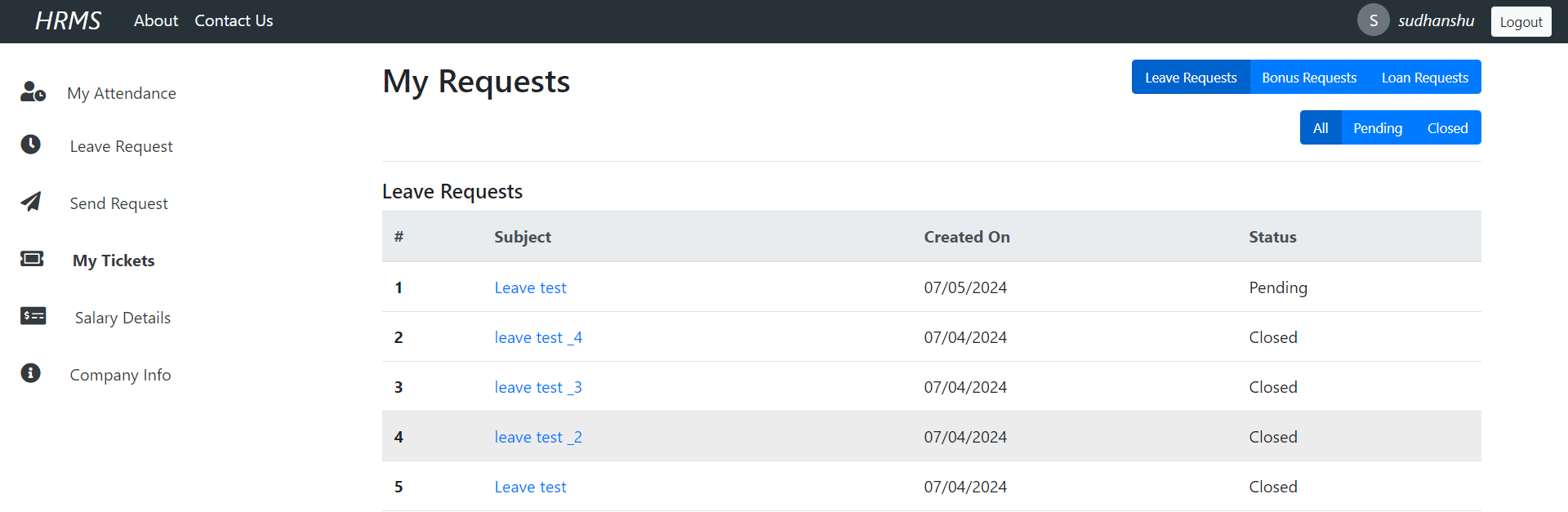


Figure : 8

## Secure and Efficient Database Management

* **Description:**  Ensures data security and optimizes access to employee records, safeguarding sensitive information while enhancing operational efficiency
* **Implementation:**
  + Secure Database Management involves utilizing AES encryption for robust data security, implementing RBAC for granular access control, and maintaining comprehensive audit logs for enhanced transparency and accountability.
  + Efficient Data Management is achieved through the development of intuitive search, filter, and pagination features, enabling quick and seamless access to relevant employee records.
* **Benefits:**
  + Enhanced security and compliance are ensured by AES encryption and RBAC, which not only mitigate breaches but also ensure regulatory compliance, fostering trust and confidence in the HRMS system.
  + Improved Efficiency and User Experience are realized as search, filter, and pagination functionalities streamline data access, empowering users with efficient tools for data retrieval and manipulation, ultimately enhancing productivity and user satisfaction.

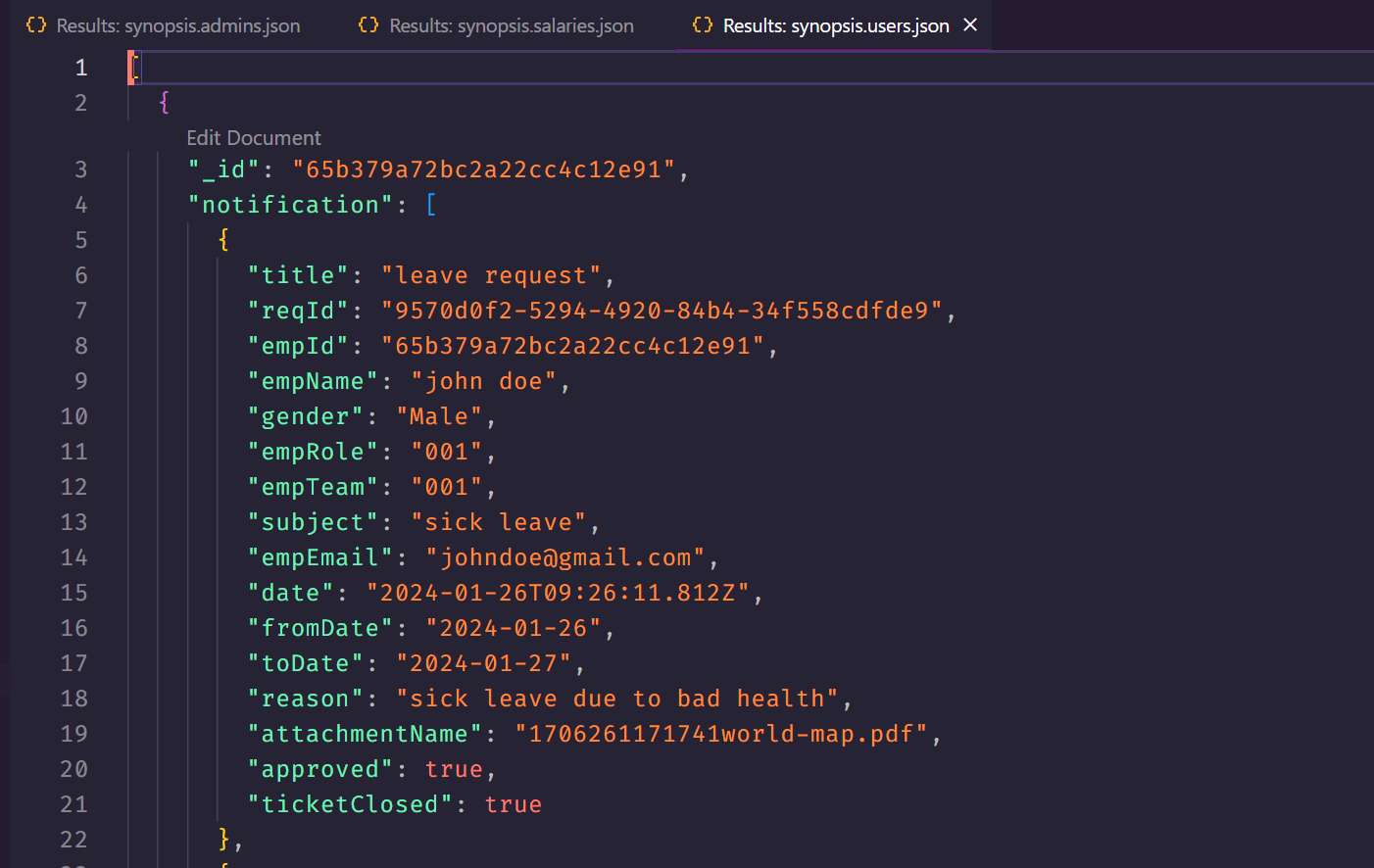


Figure : 9

## Statistical Analysis

* **Description:** Statistical analysis provides insights into various HR aspects such as employee attendance, leave patterns, and system usage.
* **Implementation:** Develop dashboards and reports to visualize HR statistics using charts, graphs, and tables. Utilize data analysis techniques to identify trends, patterns, and anomalies in HR data.
* **Benefits:**
  + Aids informed decision-making by providing actionable insights into HR metrics and trends.
  + Facilitates continuous improvement and strategic planning based on data-driven analysis.

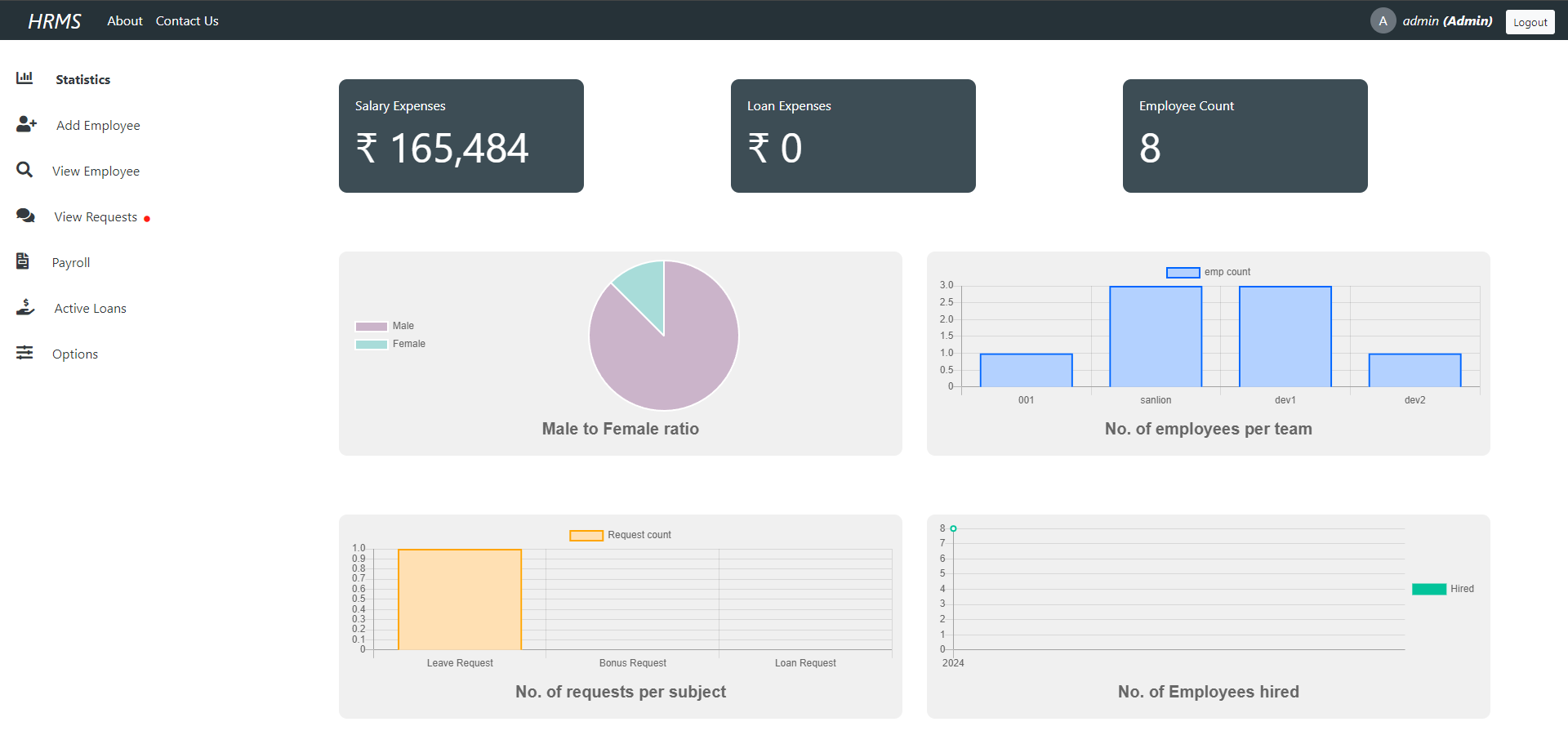


Figure: 10

## Google Calendar Integration

* **Description:** Google Calendar integration allows users to synchronize HR tasks and events with their Google Calendar, enabling them to set reminders and receive notifications.
* **Implementation:** Utilize Google Calendar APIs to integrate HR tasks with users' Google Calendar accounts. Implement event scheduling and reminder functionalities to notify users of upcoming HR-related events.
* **Benefits:**
  + Improves user productivity by synchronizing HR tasks with users' existing calendar systems.
  + Enhances user engagement and participation in HR-related activities by providing timely reminders and notifications.
  + By implementing these features, your MERN stack-based HR management system can effectively streamline HR processes, enhance data security and integrity, and provide valuable insights for informed decision-making and strategic planning.

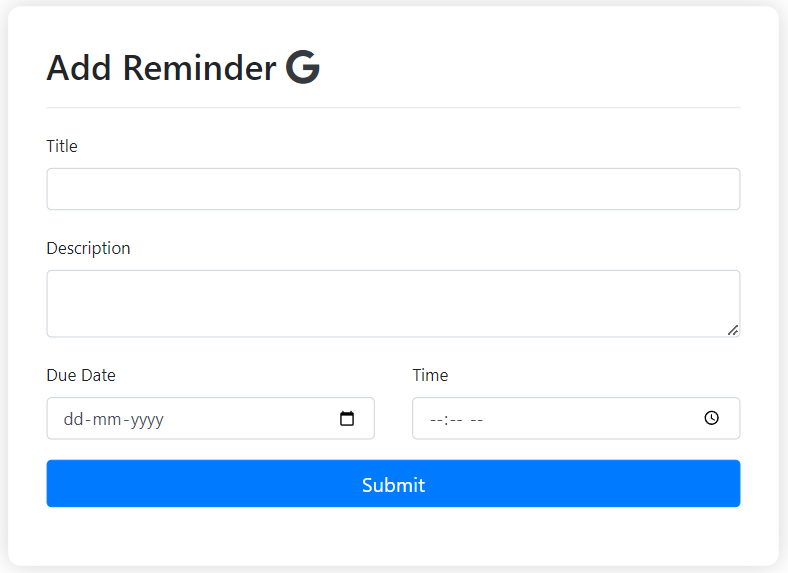


Figure: 11

## News Section

* **Description:** The Latest News Section on the employee dashboard provides curated news updates relevant to employees, covering industry trends, company announcements, and professional development insights.
* **Implementation:** Utilize news aggregation APIs to fetch articles from reputable sources and display them in a user-friendly format on the dashboard. Implement categorization and filtering options to allow users to explore news articles based on their interests.
* **Benefits:**
  + Stay Informed: Keep employees updated on industry developments, company news, and career-related insights to foster continuous learning and professional growth.
  + Engagement: Encourage employee engagement by providing relevant and interesting content directly within the HR management system, increasing user interaction and participation.
  + Knowledge Sharing: Facilitate knowledge sharing and discussions among employees by providing a centralized platform for accessing and discussing news articles and relevant topics.

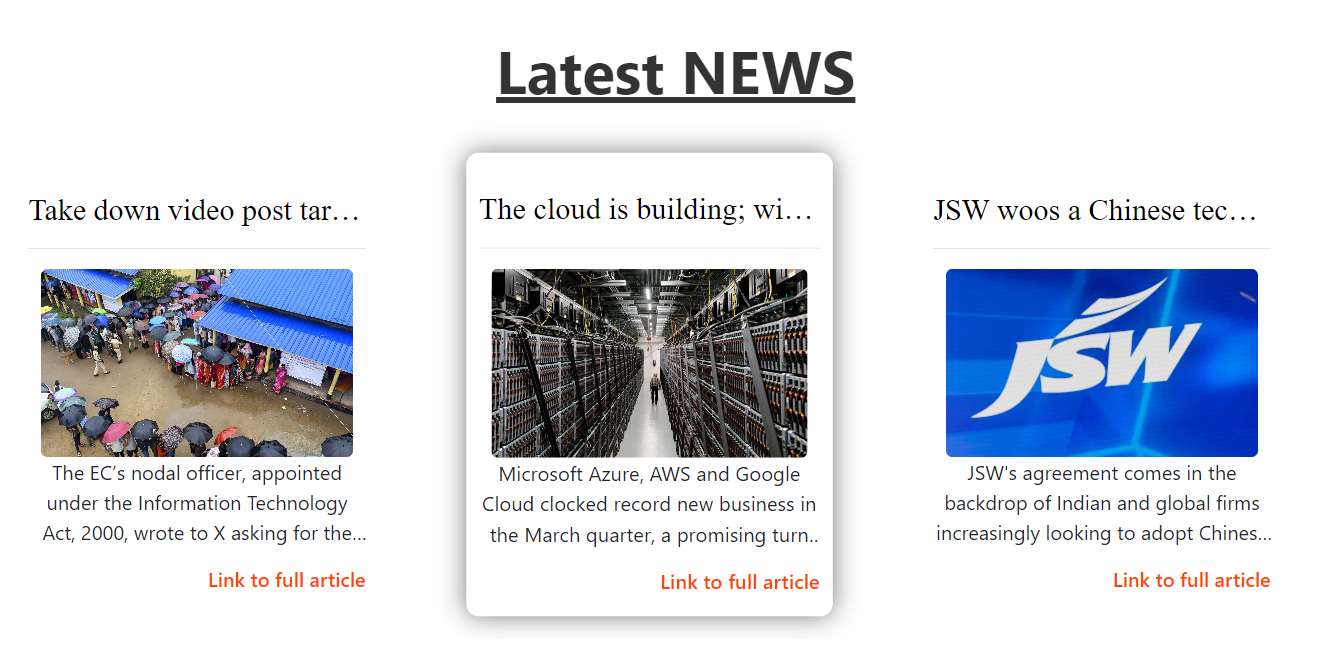


Figure: 12

By implementing these features, your MERN stack-based HR management system can effectively streamline HR processes, enhance data security and integrity, and provide valuable insights for informed decision-making and strategic planning.

# CHAPTER - 5: ***Future Scopes & Possibilities***

## Mobile Application Development

* Develop custom mobile applications for iOS and Android platforms tailored to the HRMS functionalities.
* Features may include:
  + Secure login for employees to access their profiles and HR services.
  + Submission of leave requests with real-time status updates.
  + Participation in company surveys, training programs, and events.
  + Integration with push notifications to alert employees about important HR updates or upcoming deadlines.
  + Collaboration with HR administrators to ensure seamless integration with existing HR systems and databases.

## Chrome Extension for Attendance Management

* Design and deploy a Chrome extension specifically for attendance management within the HRMS ecosystem.
* Features may include:
  + Clock-in/out functionality for employees to record their work hours.
  + Real-time attendance tracking to monitor employee presence and absence.
  + Automated generation of attendance reports for payroll processing.
  + Integration with HRMS dashboard for administrators to oversee attendance data and trends.
  + Collaboration with IT and HR departments to ensure compatibility with existing systems and security protocols.

## Enhanced Data Analytics

* Implement advanced data analytics techniques to leverage the wealth of HR data captured within the HRMS.
* Utilize machine learning algorithms for:
  + Employee sentiment analysis to gauge overall satisfaction and identify potential retention risks.
  + Predictive analytics for talent management, including identifying high-potential employees and predicting future staffing needs.

## Enhanced User Interface

* Enhance the user interface (UI) of the HRMS to improve usability and user satisfaction.
* Conduct user research and usability testing to identify pain points and areas for improvement in the current UI design.
* Redesign UI elements, navigation menus, and workflows to prioritize user needs and streamline interactions.
* Incorporate responsive design principles to ensure compatibility and optimal user experience across various devices and screen sizes.
* Implement personalized dashboards and customizable settings to empower users to tailor their HRMS experience according to their preferences and job roles.

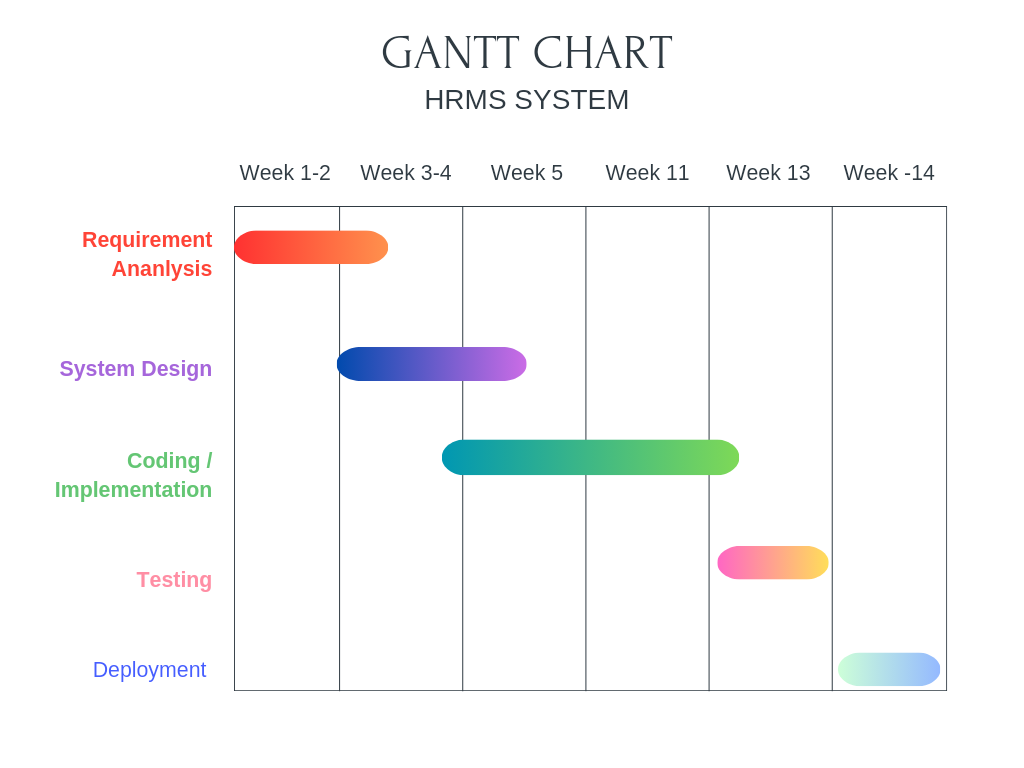
## Third-Party APIs Integration

* Integrate third-party APIs to expand the functionality and capabilities of the HRMS and enhance organizational productivity.
* Identify key areas for integration, such as:
  + Integration with LinkedIn or other job boards for seamless talent acquisition and recruitment processes.
  + Integration with communication platforms like Slack or Microsoft Teams for enhanced collaboration and employee engagement.
  + Integration with project management tools such as Trello or Asana to streamline task assignment and tracking within cross-functional teams.

## Hiring & User Onboarding Management

* Develop features within the HRMS to streamline the hiring and user onboarding processes, from job posting to employee integration.
* Implement automation for repetitive tasks, such as:
  + Posting job openings on company career portals and external job boards.
  + Screening and shortlisting candidates based on predefined criteria.
  + Generating offer letters, contracts, and new employee documentation.
  + Incorporate self-service onboarding modules for new hires to complete paperwork, set up accounts, and access training materials prior to their start date.

# ***Project Timeline***

  
 Figure: 13

# ***Conclusion***

The Human Resource Management System (HRMS) represents a significant leap forward in modernizing HR operations, leveraging automation and technology to streamline key processes such as employee management, payroll processing, and ticket handling. With robust features including user authentication, data management, and Google Calendar integration, the HRMS offers a comprehensive solution for organizations seeking to optimize their HR functions.

The immediate focus of the project is on delivering a scalable and user-friendly solution that can adapt to the diverse needs of organizations across various industries. By prioritizing thorough analysis, systematic design, and iterative development, the project aims to ensure the creation of a robust and reliable system that meets the evolving requirements of HR professionals and employees.

Looking ahead, the HRMS has ambitious plans for future enhancements, including advanced analytics, mobile integration, and comprehensive reporting capabilities. These additions will further elevate decision-making processes and provide deeper insights into workforce dynamics and organizational performance.

Throughout the development process, rigorous testing and deployment procedures are followed to guarantee the system's reliability, security, and performance. The system relies on essential hardware resources such as web servers and user-accessible computers to ensure seamless operation and accessibility for users.

In summary, the HRMS is poised to revolutionize HR operations, offering organizations a powerful tool to optimize their HR functions with precision, efficiency, and adaptability. It stands as a valuable asset for organizations looking to streamline their HR operations and enhance overall organizational effectiveness in today's competitive business landscape.

# ***References/Bibliography***

* **Reactjs**: <https://react.dev/>
* **Nodejs**: <https://nodejs.org/>
* **MongoDB:** [https://www.mongodb.com/docs/](http://www.mongodb.com/docs/)
* **Expressjs:** <https://expressjs.com/>
* **VS code Editor:** <https://code.visualstudio.com/docs>
* **Bootstrap:** <https://getbootstrap.com/>
* **Javascript:** <https://developer.mozilla.org/en-US/docs/Web/JavaScript>