

EMPLOYEE MANAGEMENT SYSTEM

A MINI PROJECT REPORT

SUBMITTED BY

YESHWANTH KUMAR P	221701063
PUGAZHENTHI	221701043
MADHESH	221701035
GANESH	221701015

In partial fulfillment for the award of the degree of

**BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE AND DESIGN**

**RAJALAKSHMI ENGINEERING COLLEGE
THANDALAM
CHENNAI – 602105**



ANNA UNIVERSITY: CHENNAI 600625

BONAFIDE CERTIFICATE

Certified that this project report “ **EMPLOYEE MANAGEMENT SYSTEM**” is the bonafide work of

**“YESHWANTH KUMAR (221701063), PUGAZHENTHI (221701043),
MADHESH (221701035), GANESH (221701015)”** who carried
out the project work under my supervision.

SIGNATURE

Dr.P.Kumar M.E., Ph.D.,
Professor and Head,
Computer Science and Engineering,
Rajalakshmi Engineering College,
Thndalam, Chennai – 602105.

SIGNATURE

Mr.Vijaykumar M.Tech.,
Asst. Professor (SS),
Computer Science and Engineering,
Rajalakshmi Engineering College,
Thandalam, Chennai – 602105.

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We are highly obliged in taking the opportunity to thank our Chairman **Mr. S. Meganathan**, Chairperson **Dr.Thangam Meganathan** and our Principal **Dr.S.N.Murugesan** for providing all the facilities which are required to carry out this project work.

We are ineffably indebted to our H.O.D **Dr.P.Kumar M.E., Ph.D.**, for his conscientious guidance and encouragement to make this project a recognizable one.

We are extremely thankful to our faculty **Mr.Vijaykumar M.Tech.**, for his valuable guidance and indefatigable support and extend our heartfelt thanks to all the teaching and non-teaching staff of **Computer Science department** who helped us directly or indirectly in the completion of this project successfully.

At last but not least gratitude goes to our friends who helped us compiling the project and finally to god who made all things possible.

Any omission in this brief acknowledgement doesn't mean lack of gratitude.

YESHWANTH KUMAR P(221701063)

PUGAZHENTHI (221701043)

MADHESH (221701035)

GANESH (221701015)

ABSTRACT

The Employee Management System (EMS) is a web-based application developed to simplify employee data management. Built using HTML, CSS, and JavaScript for the frontend, MongoDB for the database, Express.js for database connectivity, and Node.js for server communication, the EMS offers a streamlined and secure solution for handling employee records.

The system features an admin login page that ensures secure access. Upon logging in, administrators can perform core operations such as adding, deleting, updating, and displaying employee details through a user-friendly interface. MongoDB provides scalable and efficient data storage, while Express.js and Node.js facilitate seamless interaction between the frontend and backend.

The EMS eliminates manual errors, automates routine tasks, and organizes employee information effectively, making it a practical tool for businesses of any size. Its lightweight design ensures fast performance and ease of use.

In summary, the Employee Management System is a simple yet efficient application for managing employee records, ensuring data security, and improving operational efficiency.

TABLE OF CONTENTS

1. INTRODUCTION	
1.1 INTRODUCTION	[6]
1.2 SCOPE OF WORK	[6]
1.3 PROBLEM STATEMENT	[6]
1.4 AIM AND OBJECTIVE	[6]
2. SURVEY OF TECHNOLOGIES	
2.1 SOFTWARE DESCRIPTION	[7]
2.2 LANGUAGES	[8]
3. REQUIREMENT AND ANALYSIS	
3.1 REQUIREMENT SPECIFICATION	[9]
3.2 HARDWARE AND SOFTWARE SPECIFICATION	[10]
4. PROGRAM CODE	[11]
5. RESULT AND DISCUSSION	
5.1 FUNCTIONALITY OF THE PROJECT	[15]
5.2 USER FEEDBACK	[22]
5.3 CHALLENGES FACED DURING DEVELOPMENT	[22]
6. CONCLUSION	[24]
7. REFERENCES	[25]

CHAPTER 1

1.INTRODUCTION

An Employee Management System (EMS) is a software application designed to manage and streamline employee-related processes within an organization. It helps simplify operations by automating employee data management, reducing manual errors, and ensuring accurate and timely information for decision-making. Effective employee management is essential for improving productivity, ensuring compliance, and enhancing employee satisfaction..

2.SCOPE OF WORK

The scope of this project covers the development of an Employee Management System that enables organizations to manage their workforce efficiently. The system will support core functionalities such as adding, updating, and deleting employee records; managing payroll; processing leave requests; and displaying employee data in an organized format. The system will be designed to support organizations of all sizes, from small businesses to large enterprises.

3.PROBLEM STATEMENT

Many organizations face challenges in managing employee data effectively, leading to issues such as payroll errors, mismanagement of leave records, and inefficient workflows. These challenges can result in increased operational costs, reduced employee satisfaction, and administrative inefficiencies. Traditional, manual methods of employee management are prone to errors, time-consuming, and lack scalability for modern organizational needs.

4.AIM AND OBJECTIVE

The aim of this project is to develop a user-friendly Employee Management System that simplifies employee data tracking, streamlines workflows, and ensures data accuracy. This system aims to support organizations in optimizing their operations, reducing manual errors, and improving efficiency.

Objectives:

1. Provide a welcome page for user navigation.
2. Enable employees to log in and view their profiles securely.
3. Allow admins to add, update, delete, and display employee records.
4. Minimize manual errors and improve operational efficiency.

CHAPTER 2

2.1 SOFTWARE DESCRIPTION

The Employee Management System software is a web-based application designed to provide a comprehensive solution for managing employee data. The software is developed using various programming languages, tools, and technologies to ensure its efficiency, scalability, and security. The following is a description of the software used in the development of the Employee Management System:

HTML: HTML is a markup language used to structure the content of the Employee Management System. It defines the layout and structure of the user interface and is used in conjunction with CSS and JavaScript to create interactive and user-friendly web pages.

CSS: CSS is a style sheet language used to design the visual appearance of the Employee Management System. It ensures consistency in layout, color schemes, and overall design, providing an aesthetically pleasing user interface.

JavaScript: JavaScript is a client-side scripting language that is used to handle user interactions and dynamic behaviors within the Employee Management System. It enables features such as form validation and interactive components.

Node.js: Node.js is a JavaScript runtime environment used for server-side development. It ensures efficient handling of server-side operations and enables seamless communication between the client and server in the Employee Management System.

Express: Express is a web application framework for Node.js that simplifies server-side development. It handles routing and server logic, ensuring smooth operation and scalability of the system.

MongoDB: MongoDB is a NoSQL database used to store and manage employee data in the Employee Management System. It is highly scalable and suitable for handling large datasets, making it ideal for dynamic and flexible data management.

These software components are integrated and tested to ensure the proper functioning of the Employee Management System. The software is designed to be scalable, efficient, and secure, with regular updates and maintenance to ensure its continued performance.

2.2 LANGUAGES

2.2.1 JAVASCRIPT

JavaScript is the core programming language used for both client-side and server-side development in the Employee Management System. It enables dynamic functionality, handles user interactions, and manages communication between the front end and back end. JavaScript ensures responsiveness and interactivity across the platform.

SQL-LIKE QUERIES MONGODB

MongoDB employs a query language similar to SQL for managing database operations in the Employee Management System. This language is used for performing CRUD (Create, Read, Update, Delete) operations, ensuring that employee records are efficiently stored, retrieved, and updated as needed.

CHAPTER 3

3.1 REQUIREMENT SPECIFICATION

3.1.1 Functional Requirements:

1. User Management:

- Admin should be able to create, update, and delete employee records.
- Users should be able to view their profiles and update limited personal information.

2. Employee Data Management:

- Admin should be able to add, update, and delete employee details.
- Each employee record should include attributes such as name, ID, department, designation, salary, and contact details.

3. Payroll Management:

- The system should calculate salaries based on predefined rules, including allowances, deductions, and taxes.
- Payroll records should include employee name, ID, salary breakdown, and payment date.

4. Leave Management:

- Employees should be able to request leaves through the system.
- Admin should have the ability to approve or reject leave requests and track leave balances..

5. Report Management:

- The system should generate detailed reports on employee data, payroll summaries, and leave statistics.
- Reports should provide insights for decision-making processes.

6. Dashboard:

- Same for Admin dashboard should display key metrics like the total number of employees, pending leave requests, and payroll activities.
- Employees should have personalized dashboards summarizing their profiles, leave balances, and payroll details.

3.1.2 Non-Functional Requirements:

1. Security:

- The system should include secure login and authentication mechanisms for all users.
- Role-based access control should ensure only authorized users can access specific functionalities.

2. Performance:

- The system should be responsive and handle multiple users efficiently.
- Reports should be generated quickly, even with large datasets.

3. Usability:

- User interface should be intuitive and easy to navigate for both admins and employees.
- Consistent layout and design across all pages for better user experience.

3.2 HARDWARE AND SOFTWARE REQUIREMENT

3.2.1 Hardware Requirements:

- **Server:**
 - Processor: Quad-Core CPU
 - RAM: 8 GB or higher
 - Storage: 500 GB SSD or higher
 - Network: High-speed internet connection
- **Client Devices:**
 - Any device capable of running a modern web browser (desktop, laptop, tablet, or smartphone)
 - Screen resolution: 1024x768 or higher

3.2.2 Software Requirements:

- **Server:**
 - Operating System: Linux (preferred), Windows Server, or macOS
 - Web Server: Node.js or Nginx
 - Database Server: MongoDB
 - Programming Framework: Express.js
- **Client Devices:**
 - Web Browser: Latest versions of Google Chrome, Mozilla Firefox, Microsoft Edge, or Safari

CHAPTER 4

P R O G R A M M I N G C O D E W E L C O M E P A G E :

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport"
content="width=device-width, initial-
scale=1.0">
  <script
src="https://cdn.tailwindcss.com"></scr
ipt>
  <title>Welcome - Employee
Management System</title>
</head>
<body class="bg-gray-100">
  <header class="bg-slate-800 text-
white p-4 flex justify-between">
    <h1 class="text-2xl font-
bold">Employee Management System</h1>
    <nav>
      <a href="welcome.html"
class="mx-2 hover:underline">Home</a>
```

```
      <a href="employee-
login.html" class="mx-2
hover:underline">Employee Login</a>
      <a href="admin-login.html"
class="mx-2 hover:underline">Admin
Login</a>
    </nav>
  </header>
  <main class="text-center p-10">
    <h2 class="text-3xl font-
bold">Welcome to the Employee
Management System</h2>
    <p class="text-gray-700 mt-
4">Navigate through the system using
the links above.</p>
  </main>
</body>
</html>
```

EMPLOYEE LOGIN PAGE:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport"
content="width=device-width, initial-
scale=1.0">
  <script
src="https://cdn.tailwindcss.com"></scr
ipt>
  <title>Employee Login - Employee
Management System</title>
</head>
<body class="bg-gray-100">
  <header class="bg-slate-800 text-
white p-4 flex justify-between">
    <h1 class="text-2xl font-
bold">Employee Management System</h1>
    <nav>
      <a href="welcome.html"
class="mx-2 hover:underline">Home</a>
      <a href="employee-
login.html" class="mx-2
hover:underline">Employee Login</a>
      <a href="admin-login.html"
class="mx-2 hover:underline">Admin
Login</a>
    </nav>
  </header>
  <main class="flex justify-center
items-center min-h-screen">
    <div class="bg-white p-6
shadow-lg rounded-lg max-w-sm w-full">
      <h2 class="text-xl font-
bold mb-4">Employee Login</h2>
      <form>
        <div class="mb-4">
```

```

        <label
for="employee-email" class="block text-
sm font-bold">Email</label>
        <input type="email"
id="employee-email" class="w-full px-4
py-2 border rounded" placeholder="Enter
your email">
    </div>
    <div class="mb-4">
        <label
for="employee-password" class="block
text-sm font-bold">Password</label>
        <input
type="password" id="employee-password"
class="w-full px-4 py-2 border rounded"
placeholder="Enter your password">
    </div>
    <button type="button"
onclick="employeeLogin()" class="bg-
blue-500 text-white py-2 px-4 rounded
w-full">
        Login
    </button>
</form>
</div>
</main>
<script>
    const employeeDatabase = {
        "yeshwanth@gmail.com": {
name: "Yeshwanth", password:
"yeshwanth123" },
        "pugazh43@example.com": {
name: "Pugazhenth", password:
"pugazh123" },
    };

    function employeeLogin() {
        const email =
document.getElementById("employee-
email").value;
        const password =
document.getElementById("employee-
password").value;

        if (email in
employeeDatabase &&
employeeDatabase[email].password ===
password) {
            alert(`Welcome,
${employeeDatabase[email].name}!`);
            window.location.href =
"employee-dashboard.html"; // Redirect
to employee dashboard
        } else {
            alert("Invalid
credentials. Please try again.");
        }
    }
</script>
</body>
</html>

```

```

</div>                                </div>
<?php include_once('layouts/footer.php'); ?>

```

ADMIN LOGIN PAGE:

```

<?php
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <script src="https://cdn.tailwindcss.com"></script>
    <title>Employee Dashboard</title>
</head>
<body class="bg-gray-100">
    <header class="bg-slate-800 text-white p-4 flex justify-between">
        <h1 class="text-2xl font-bold">Employee Management System</h1>
        <nav>
            <a href="welcome.html" class="mx-2 hover:underline">Home</a>
            <a href="employee-login.html" class="mx-2
hover:underline">Logout</a>
        </nav>
    </header>
    <main class="flex flex-col items-center justify-center min-h-screen">
        <div class="bg-white p-6 shadow-lg rounded-lg max-w-lg w-full text-
center">
            <h2 class="text-2xl font-bold mb-4">Welcome, <span id="employee-
name">Employee</span>!</h2>
            <p class="text-gray-700 mb-4">
                Below are your details. Please ensure they are accurate.
            </p>
            <div class="text-left space-y-2">
                <p><strong>Name:</strong> <span id="emp-name"></span></p>
                <p><strong>Email:</strong> <span id="emp-email"></span></p>
                <p><strong>Occupation:</strong> <span id="emp-
occupation"></span></p>
            </div>
        </div>
    </main>

    <script>
        // Retrieve employee details from session storage
        const employeeName = sessionStorage.getItem("empName");
        const employeeEmail = sessionStorage.getItem("empEmail");
        const employeeOccupation = sessionStorage.getItem("empOccupation");

        // Update the page content dynamically
        if (employeeName && employeeEmail && employeeOccupation) {
            document.getElementById("employee-name").textContent = employeeName;
            document.getElementById("emp-name").textContent = employeeName;
            document.getElementById("emp-email").textContent = employeeEmail;
            document.getElementById("emp-occupation").textContent =

```

```
employeeOccupation;  
    } else {  
        // If session storage is empty, redirect to login  
        alert("Unauthorized access. Please log in.");  
        window.location.href = "employee-login.html";  
    }  
    </script>  
</body>  
</html>
```

CHAPTER 5

RESULT AND DISCUSSION

5.1 Functionality of the Project

The Employee Management System was developed to streamline and automate the management of employees, their records, and user roles within an organization. The system includes several key functionalities:

1. WELCOME PAGE:

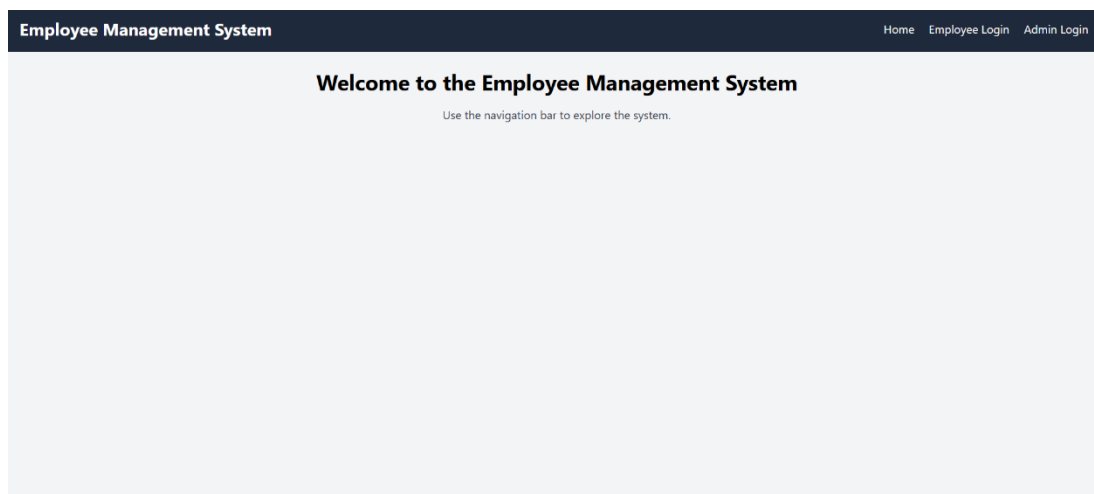
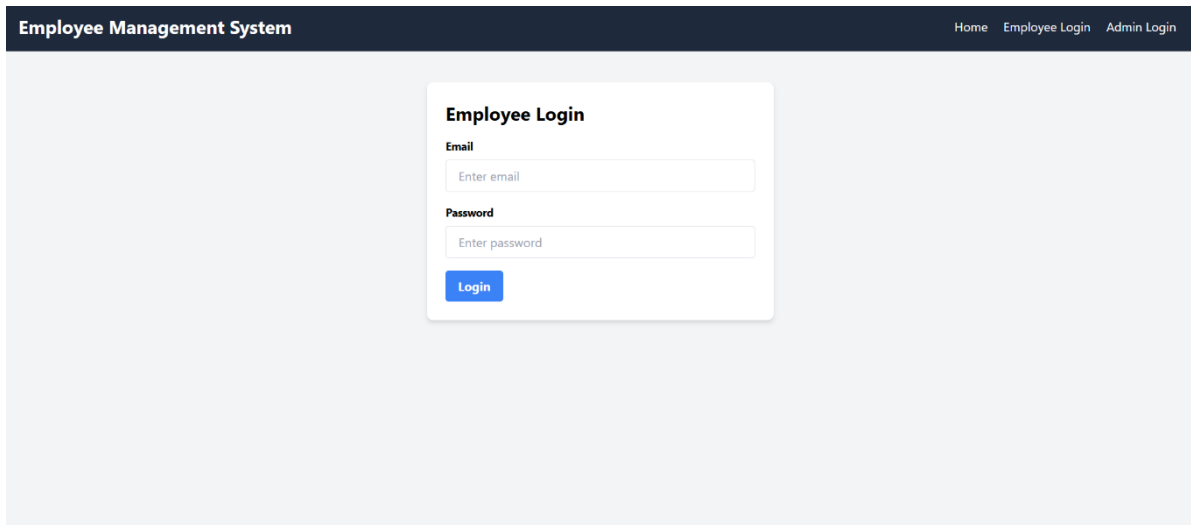


Fig 5.1.1 – Welcome Page

- The welcome page serves as the entry point for the system.
- It features navigation links for employees and admins to access their respective login pages.
- The page is user-friendly and responsive, providing an overview of the Employee Management System's purpose.s.

2. Employee Login:

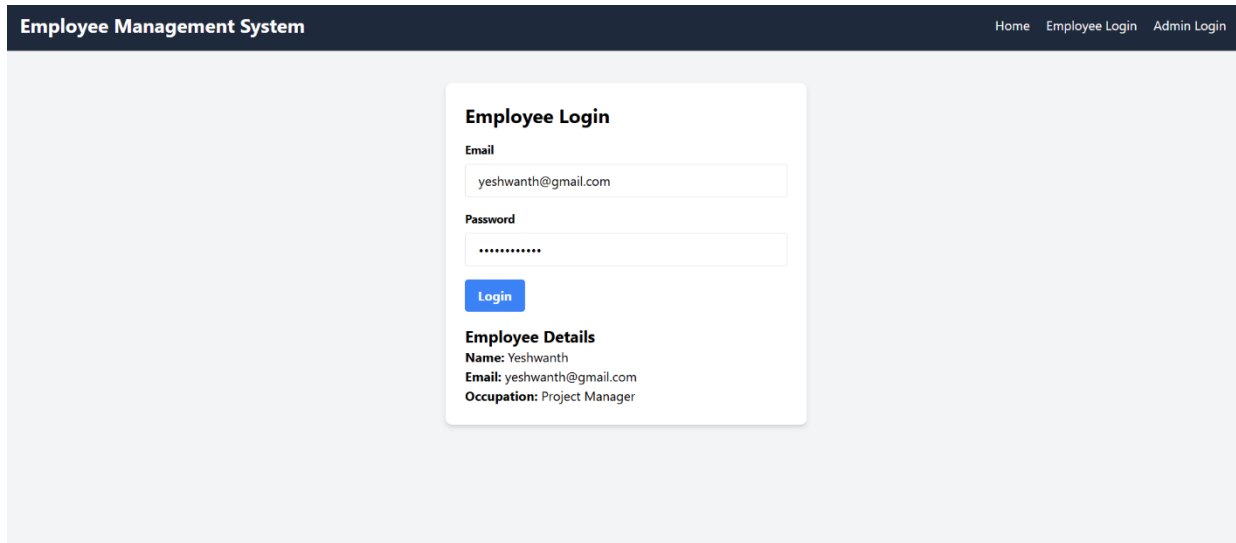


The screenshot displays the 'Employee Login' interface within an 'Employee Management System'. At the top, a dark blue header bar contains the system name on the left and navigation links for 'Home', 'Employee Login', and 'Admin Login' on the right. The main content area is light gray and features a white login form. The form is titled 'Employee Login' and includes two input fields: 'Email' with the placeholder text 'Enter email' and 'Password' with the placeholder text 'Enter password'. Below these fields is a blue 'Login' button.

Fig 5.1.2 – Employee Login

- Employees can securely log in using their credentials.
- The system verifies the email and password with the database for authentication.
- Upon successful login, employees are redirected to their profile/dashboard.category

3. Employee Profile:

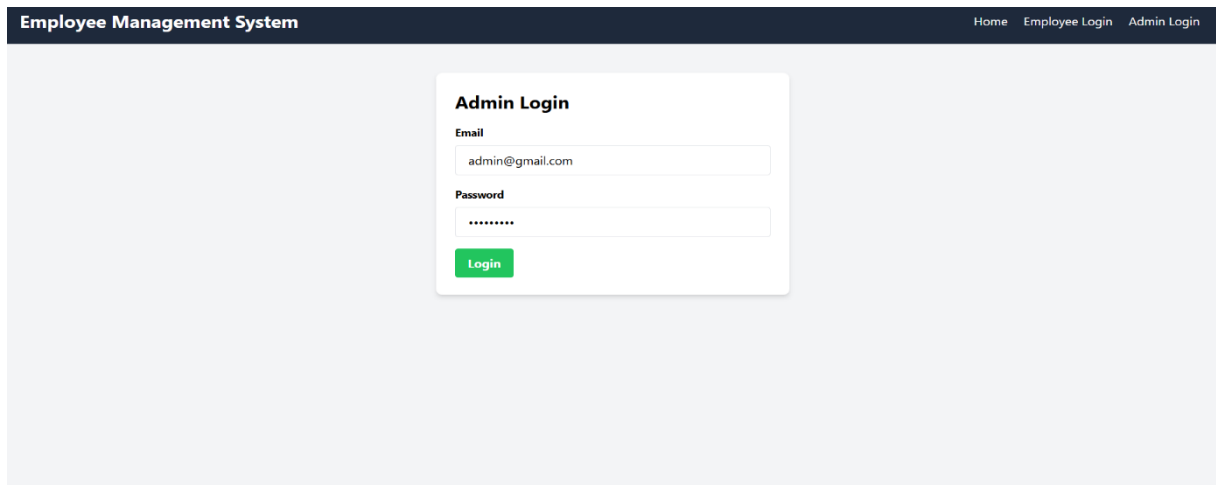


The screenshot displays the 'Employee Management System' interface. At the top, a dark blue header bar contains the system name on the left and navigation links 'Home', 'Employee Login', and 'Admin Login' on the right. The main content area is light gray and features a white login card. The card has a title 'Employee Login' and two input fields: 'Email' (containing 'yeshwanth@gmail.com') and 'Password' (masked with dots). A blue 'Login' button is positioned below the password field. Underneath the login section, the 'Employee Details' are listed: 'Name: Yeshwanth', 'Email: yeshwanth@gmail.com', and 'Occupation: Project Manager'.

Fig 5.1.3 – Employee Profile

- After login, employees can view their profile details, including their name, email, and occupation.
- The profile page ensures that employees have access to relevant and accurate data.
- The design is intuitive, allowing employees to focus on their personal information.files.

4. Admin Login:



The screenshot displays the 'Admin Login' page of an 'Employee Management System'. The page features a dark blue header with the system name on the left and navigation links ('Home', 'Employee Login', 'Admin Login') on the right. The main content area is light gray and contains a white login form. The form is titled 'Admin Login' and includes two input fields: 'Email' (containing 'admin@gmail.com') and 'Password' (displayed as dots). A green 'Login' button is positioned at the bottom of the form.

Fig 5.1.4 – Admin Login

- Admins log in using a dedicated interface with their credentials.
- The system provides secure access to the admin dashboard after verifying login details.
- Robust authentication ensures unauthorized users are restricted.automatically

5. Admin Dashboard:

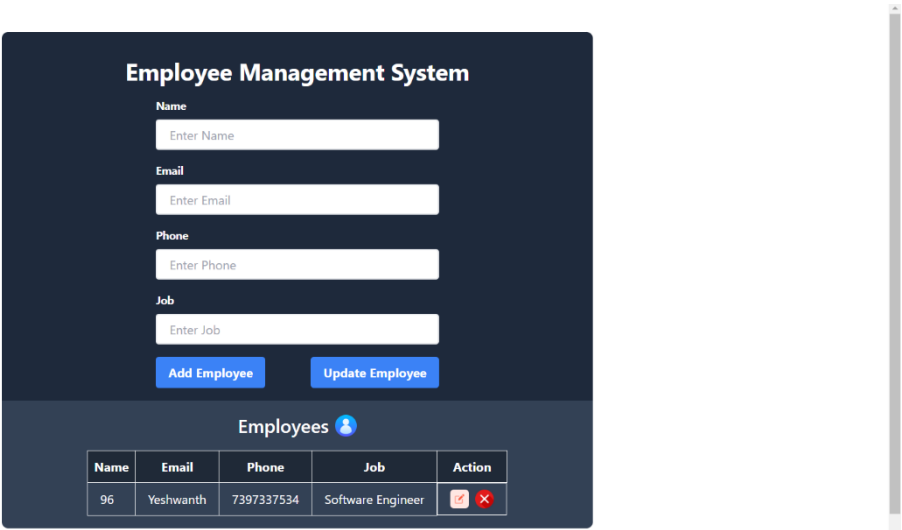


Fig 5.1.5 – Admin Dashboard

- The admin dashboard is the central control panel for managing the system.
- Admins can add, update, and delete employee records.
- Key features include access to employee data, managing user roles, and generating detailed reports.

5.2 User Feedback

During the testing phase, the Employee Management System was evaluated through user feedback to assess its performance and usability. Here are some key insights from the feedback:

1. **Ease of Use:** Users found the system intuitive and straightforward. The interface design was appreciated for its clean and responsive layout, enhancing the overall user experience.
2. **Functionality:** Users valued the system's features, particularly the secure employee and admin login functionalities and the ability to manage employee records efficiently.
3. **Performance:** The system demonstrated quick response times and stable operation under various scenarios, ensuring reliable data handling.
4. **Security:** Users expressed confidence in the security measures, including robust authentication protocols and role-based access control, which safeguarded sensitive data effectively.

.

5.3 Challenges Faced During Development

Developing the Employee Management System involved addressing several challenges through strategic planning and solutions:

1. Database Design:

- **Challenge:** Designing a flexible and efficient database structure to manage employee information and interactions.
- **Solution:** Employing MongoDB to create a scalable and dynamic schema that allowed for seamless data updates and retrieval.

2. User Authentication and Authorization:

- **Challenge:** Implementing a secure authentication mechanism and managing different access levels for employees and administrators.
- **Solution:** Using Node.js and Express with robust middleware to establish secure login systems and enforce role-based access control.

3. Front-End Design:

- **Challenge:** Developing a user-friendly and responsive interface compatible with various devices and screen sizes.
- **Solution:** Leveraging modern web design practices with HTML, CSS,

and JavaScript to ensure a visually appealing and functional layout.

4. Integration of Admin and Employee Portals:

- **Challenge:** Seamlessly connecting the admin and employee portals for efficient data flow and access management.
- **Solution:** Implementing a structured routing system in Express.js to handle distinct functionalities without conflict.

5. Error Handling and Validation:

- **Challenge:** Ensuring robust error handling and validation mechanisms for data integrity.
- **Solution:** Developing comprehensive validation checks for input fields and implementing exception-handling logic in both the front-end and back-end.

Overall, the development process provided valuable learning experiences. Through continuous iteration and feedback, the final system met its objectives, delivering a reliable and effective solution for employee management.

CONCLUSION

In conclusion, the Employee Management System project successfully achieved its goal of providing a comprehensive and efficient platform for managing employee records. By incorporating features such as secure authentication, role-based access control, and streamlined data management processes, the system addressed key challenges faced by organizations in workforce management.

The use of modern technologies like Node.js, Express, and MongoDB for back-end development, combined with HTML, CSS, and JavaScript for the front-end, ensured a robust, responsive, and user-friendly application.

During development, challenges such as database design, user authentication, and error handling were overcome through strategic solutions and iterative improvements. User feedback highlighted the system's ease of use, performance, and security, validating its effectiveness and usability.

Overall, the project not only provided a practical solution for employee management but also served as a learning experience in software development, database management, and user interface design. The resulting system reflects the team's dedication and effort in creating a reliable and efficient platform.

REFERENCES

- a. Official Node.js Documentation: <https://nodejs.org/en/docs/>
- b. Official MongoDB Documentation: <https://www.mongodb.com/docs/>
- c. MDN Web Docs for HTML: <https://developer.mozilla.org/en-US/docs/Web/HTML>
- d. MDN Web Docs for CSS: <https://developer.mozilla.org/en-US/docs/Web/CSS>
- e. MDN Web Docs for JavaScript: <https://developer.mozilla.org/en-US/docs/Web/JavaScript>
- f. Express.js Documentation: <https://expressjs.com/>
- g. W3Schools for Web Development Basics: <https://www.w3schools.com/>
- h. Stack Overflow for Development Support: <https://stackoverflow.com/>
- i. Visual Studio Code Documentation: <https://code.visualstudio.com/docs>
- j. GitHub for Version Control: <https://github.com/>