Project Proposal: Employee Management System

Contents

[1. Introduction 2](#_Toc172494480)

[2. System Requirements 2](#_Toc172494481)

[2.1 Functional Requirements 2](#_Toc172494482)

[Employee Management: 2](#_Toc172494483)

[Attendance Tracking: 2](#_Toc172494484)

[Leave Management: 2](#_Toc172494485)

[2.2 Non-Functional Requirements 3](#_Toc172494486)

[3. System Architecture 3](#_Toc172494487)

[3.1 Front-End (Javascript): 3](#_Toc172494488)

[3.2 Back-End (C#): 3](#_Toc172494489)

[3.3 Database (MSSQL ): 3](#_Toc172494490)

[4. System Features 4](#_Toc172494491)

[4.1 Employee Management 4](#_Toc172494492)

[4.2 Attendance Tracking 4](#_Toc172494493)

[4.3 Leave Management 4](#_Toc172494494)

[5. Implementation Plan 5](#_Toc172494495)

[5.1 Phase 1: 5](#_Toc172494496)

[5.2 Phase 2: 5](#_Toc172494497)

[5.3 Phase 3: 5](#_Toc172494498)

[6. Project Team 5](#_Toc172494499)

[7. Timeline and Budget 5](#_Toc172494500)

[8. Conclusion 6](#_Toc172494501)

# 1. Introduction

This proposal outlines the development of a comprehensive employee management system leveraging Javascript for the front-end, C# for the back-end, and a relational database (MSSQL) to store data. The system aims to provide a user-friendly platform for managing employee performance, attendance, leave requests, and communication.

# 2. System Requirements

## 2.1 Functional Requirements

Employee Management:

o Add, edit, and delete employee profiles

o Manage employee roles and permissions

o Track employee performance metrics (e.g., tasks completed, project contributions, deadlines met, manager rating)

o Set performance goals and track progress

### Attendance Tracking:

o Record employee attendance (check-in/check-out)

o Calculate hours worked

o Generate reports based on hours worked

### Leave Management:

o Apply for leave (different leave types: sick, vacation, etc.)

o Track leave days due and taken

o Approve/reject leave requests

o Send notifications to relevant employees upon leave approval/rejection

o Remind employees about upcoming leave dates

## 2.2 Non-Functional Requirements

• Security: Secure user authentication and authorization

• Performance: Efficient data processing and fast loading times

• Scalability: Ability to handle increasing data volume and user base

• Usability: Intuitive and user-friendly interface

• Accessibility: Comply with accessibility standards for all users

# 3. System Architecture

### 3.1 Front-End (Javascript):

• Framework: React (for its component-based architecture and ease of development)

• UI Design: Material Design or similar, emphasizing clean and modern aesthetics

• Data Fetching and Manipulation: Use Fetch API or Axios for communication with the backend

### 3.2 Back-End (C#):

• Framework: ASP.NET Core (for its robust features, security, and performance)

• Database Interaction: Entity Framework Core for efficient database operations

• API Development: Design RESTful APIs to expose data and functionality to the front-end

• Security: Implement authentication and authorization mechanisms using JWT or similar

### 3.3 Database (MSSQL ):

• Schema Design: Define tables and relationships to store employee data, attendance records, leave requests, and communication details

• Data Integrity: Implement constraints and triggers to ensure data accuracy and consistency

# 4. System Features

### 4.1 Employee Management

• Profile Creation: Users can create detailed profiles including personal information, contact details, employment history, and performance goals.

• Role Management: Different user roles (e.g., employee, manager, administrator) with specific permissions.

• Performance Tracking: Managers can assign tasks, track progress, and provide feedback on employee performance. Performance metrics include whether or not tasks where completed before their deadlines, rating from managers.

• Reporting: Generate reports on employee performance, including individual and team performance metrics.

* Notifications: Allow for the sending of notifications before performance reviews and deadlines.

### 4.2 Attendance Tracking

• Time Tracking: Employees can clock in and out using a web-based interface or mobile application.

• Attendance Reporting: Generate reports on attendance patterns, including hours worked, overtime, and absenteeism. Reports generated weekly, monthly and quarterly.

• Leave Management: Employees can apply for different types of leave, including sick leave, vacation, and unpaid leave. Managers can review and approve/reject leave requests. Employees can also view leave days available to them.

### 4.3 Leave Management

• Leave Request: Employees can submit leave requests with detailed information (dates, reason).

• Approval Process: Managers can review and approve/reject leave requests.

• Notifications: Automated notifications will be sent to relevant employees (e.g., manager, team members) upon leave approval/rejection.

• Reminders: The system will automatically remind employees about approaching leave dates.

# 5. Implementation Plan

### 5.1 Phase 1:

• Front-End Development: Create basic UI components for employee management, attendance tracking, and leave request submission.

• Back-End Development: Set up ASP.NET Core project, implement basic API endpoints, and create database schema.

• Database Integration: Connect front-end components to back-end APIs for data retrieval and submission.

### 5.2 Phase 2:

• Feature Implementation: Develop additional features like performance management, communication module, and leave notification system.

• Security Implementation: Integrate user authentication and authorization mechanisms for secure access.

• Testing and QA: Conduct comprehensive testing of all features to ensure functionality and security.

### 5.3 Phase 3:

• Deployment: Deploy the application to a suitable hosting environment (cloud or on-premises).

• User Training and Documentation: Provide user training materials and documentation for effective usage.

• Maintenance and Support: Ongoing support for bug fixes, feature enhancements, and user feedback.

# 6. Project Team

• Front-End Developer: Experienced in Javascript, React, and UI/UX design

• Back-End Developer: Experienced in C#, ASP.NET Core, and database development

• Database Administrator: Experienced in managing MSSQL databases

• Project Manager: Responsible for overall project management, planning, and execution

# 7. Timeline and Budget

• Timeline: 16 weeks (depending on complexity and resources)

• Budget: Varies based on team size, development tools, and hosting costs.

# 8. Conclusion

This proposal outlines a comprehensive plan for developing an employee management system to enhance efficiency, communication, and performance tracking within an organization. The proposed system will be user-friendly, scalable, and secure, offering a comprehensive platform for managing all aspects of employee activity. The project team possesses the necessary expertise to successfully deliver a high-quality system that meets the needs of the organization.