Dean H - Employee Performance Management System Design Document

Introduction **2**
- **Key Features** **2**
- Tools and Technologies 2

System Architecture **3**
- Presentation Layer 3
- Application Layer 3
- Data Layer 3
- Scalability and Performance Considerations 3

SQL Database Creation **4**
- Table Creation 4
- Inserting Data 5

Components **7**

Note - This project is not completed yet and I will still be working on this documentation as I produce the product. So there will be pseudocode to help structure the project and might change when I finish. But hopefully this will give you an idea of where I intend the project to go.

Introduction

The Employee Performance Management System is an ASP.NET Core MVC application designed to streamline the management of employee records and performance evaluations within an organization. By leveraging Microsoft SQL Server, Entity Framework Core, and SSRS, the system provides a comprehensive solution for tracking employee data, conducting performance reviews, and generating insightful reports. This application aims to enhance HR processes and decision-making by offering an intuitive interface for data management and analysis.

Key Features

- The Employee Records Management feature utilizes SQL Server to manage employee details and departmental assignments. This information is dynamically displayed on Razor pages, providing a user-friendly interface for viewing team details.

- The Performance Reviews feature utilizes SQL Server to record and manage employee performance evaluations. This information is dynamically displayed on Razor pages, providing a user-friendly interface for tracking and analyzing employee performance over time.

- The Department Management feature utilizes SQL Server to manage department details and performance metrics. This information is dynamically displayed on Razor pages, providing a user-friendly interface for monitoring and analyzing department performance.

- The Data Security and Role-Based Access Control feature ensures secure access to sensitive data by implementing role-based access control (RBAC). It utilizes SQL Server to enforce access permissions and maintain data integrity, adhering to proper database design principles. This ensures that only authorized users can view and modify sensitive data, promoting data integrity and consistency.

Tools and Technologies

ASP.NET Core MVC - For building the web application.
Entity Framework Core - For database operations.
Razor - For rendering dynamic HTML content.
SQL Server 2022 Developer Edition - For database management.
SQL Server Reporting Services (SSRS) - For creating and displaying reports.

System Architecture

The Employee Performance Management System follows a three-tier architecture, comprising the presentation layer, the application layer, and Data Layer. Each layer plays a distinct role in ensuring the functionality and performance of the system.

Presentation Layer

The presentation layer encompasses the user interface components responsible for displaying employee data, performance reviews, and department information. This layer includes web pages, views, and client-side scripts that facilitate user interaction with the application. Designed to be intuitive and user-friendly, the interface allows users to easily navigate through employee records and performance metrics.

Application Layer

Serving as the intermediary between the presentation layer and the data layer, the application layer handles business logic and data processing tasks. It comprises controllers, business logic components, and service layers responsible for processing user requests, fetching data from the data layer, and orchestrating the flow of information within the application. This layer ensures seamless integration of employee data and performance metrics retrieved from the data layer.

Data Layer

The data layer is responsible for storing and managing persistent data used by the application. It typically consists of a relational database management system (RDBMS), such as SQL Server, which stores employee records, performance reviews, department information, and other relevant data. The data layer ensures data integrity, consistency, and security through proper database design, including the definition of tables, relationships, and constraints. It also provides mechanisms for data retrieval, modification, and deletion, enabling efficient access to and manipulation of data by the application layer.

Scalability and Performance Considerations

The system architecture of the application is designed to be scalable and performant, capable of handling increasing loads and user interactions. Horizontal scalability is achieved through load balancing and auto-scaling mechanisms, allowing the application to dynamically allocate resources based on demand. Performance optimizations, such as efficient database queries and caching strategies, are implemented to minimize latency and improve responsiveness. Additionally, asynchronous processing and parallelization techniques are employed to enhance throughput and resource utilization, ensuring optimal performance under varying workloads.

SQL Database Creation

Table Creation

CreatingDatabaseT...S4PTHQ\Dean (55)) | InsertingTableData...0S4PTHQ\Dean (70))*

```sql
CREATE SCHEMA Employee;
go

CREATE SCHEMA Department;
go

-- Create EmployeeInfo Table
CREATE TABLE Employee.EmployeeInfo (
    employee_id INT IDENTITY (1,1) PRIMARY KEY,
    first_name VARCHAR(255) NOT NULL,
    last_name VARCHAR(255) NOT NULL,
    position VARCHAR(255) NOT NULL,
    hire_date DATE NOT NULL
);

-- Create PerformanceReview Table
CREATE TABLE Employee.PerformanceReview (
    review_id INT IDENTITY (1,1) PRIMARY KEY,
    employee_id INT NOT NULL,
    review_date DATE NOT NULL,
    score INT NOT NULL,
    comments VARCHAR(255),
    CONSTRAINT FK_Employee_PerformanceReview FOREIGN KEY (employee_id)
        REFERENCES Employee.EmployeeInfo(employee_id) ON DELETE CASCADE
);

-- Create DepartmentInfo Table
CREATE TABLE Department.DepartmentInfo (
    department_id INT IDENTITY (1,1) PRIMARY KEY,
    department_name VARCHAR(255) NOT NULL
);

-- Create EmployeeDepartment Table
CREATE TABLE Department.EmployeeDepartment (
    connection_id INT IDENTITY (1,1) PRIMARY KEY,
    employee_id INT NOT NULL,
    department_id INT NOT NULL,
    CONSTRAINT FK_Employee_EmployeeDepartment FOREIGN KEY (employee_id)
        REFERENCES Employee.EmployeeInfo(employee_id) ON DELETE CASCADE,
    CONSTRAINT FK_Department_EmployeeDepartment FOREIGN KEY (department_id)
        REFERENCES Department.DepartmentInfo(department_id) ON DELETE CASCADE
);
```

Inserting Data

InsertingTableData...0S4PTHQ\Dean (70))*

```
use EmployeePerformanceDB;
go


SET IDENTITY_INSERT Employee.EmployeeInfo ON;
go

INSERT INTO Employee.EmployeeInfo (employee_id, first_name, last_name, position, hire_date)
VALUES
--Team A
(1, 'John', 'Doe', 'Tech Lead', '2022-01-15'),
(2, 'Jane', 'Smith', 'Software Developer', '2022-03-22'),
(3, 'Alice', 'Johnson', 'Software Developer', '2022-06-10'),
(4, 'Bob', 'Williams', 'Software Developer', '2022-08-05'),
(5, 'Charlie', 'Brown', 'Software Developer', '2022-10-18'),
(6, 'David', 'Jones', 'Software Developer', '2023-02-14'),
(7, 'Eva', 'Miller', 'Software Developer', '2023-04-30'),
(8, 'Frank', 'Davis', 'Software Developer', '2023-07-21'),

--Team B
(9, 'Grace', 'Hall', 'Tech Lead', '2023-09-12'),
(10, 'Hannah', 'Moore', 'Software Developer', '2023-09-12'),
(11, 'Isaac', 'Clark', 'Software Developer', '2024-01-15'),
(12, 'Jack', 'Lewis', 'Software Developer', '2024-01-20'),

--HR
(13, 'Kate', 'Martin', 'HR Lead', '2023-05-09'),
(14, 'Liam', 'Walker', 'HR', '2023-07-22'),

--Support Team
(15, 'Mia', 'Harris', 'Support Team Lead', '2023-06-13'),
(16, 'Noah', 'Clarkson', 'Support Team', '2023-08-25');
GO

SET IDENTITY_INSERT Employee.EmployeeInfo OFF;
go


INSERT INTO Employee.PerformanceReview (employee_id, review_date, score, comments)
VALUES
-- Reviews for Team A
(1, '2023-01-15', 85, 'Good leadership skills'),
(2, '2023-03-22', 78, 'Consistent performance'),
(3, '2023-06-10', 82, 'Great improvement'),
(4, '2023-08-05', 74, 'Needs improvement in code quality'),
(5, '2023-10-18', 80, 'Solid performance'),
(6, '2023-12-14', 54, 'Average performance'),
(6, '2024-03-03', 24, 'Needs improving'),
(7, '2024-04-30', 88, 'Outstanding contribution'),
(8, '2024-07-21', 81, 'Good team player'),

-- Reviews for Team B
(9, '2024-09-12', 89, 'Excellent leadership'),
(10, '2024-09-12', 77, 'Good coding skills'),
(11, '2024-01-15', 83, 'Very reliable'),
(12, '2024-01-20', 79, 'Consistent worker'),
```

```
-- Reviews for HR
(13, '2024-05-09', 87, 'Strong management'),
(14, '2024-07-22', 75, 'Dependable'),

-- Reviews for Support Team
(15, '2024-06-13', 86, 'Great customer service'),
(16, '2024-08-25', 78, 'Good problem-solving skills');
GO
SET IDENTITY_INSERT Department.DepartmentInfo ON;
go

INSERT INTO Department.DepartmentInfo (department_id, department_name)
VALUES
(1, 'HR'),
(2, 'Main Team A'),
(3, 'Secondary Team B'),
(4, 'Support Team');
go

SET IDENTITY_INSERT Department.DepartmentInfo OFF;
go


INSERT INTO Department.EmployeeDepartment (employee_id, department_id)
VALUES
-- Team A in Software Development
(1, 2),
(2, 2),
(3, 2),
(4, 2),
(5, 2),
(6, 2),
(7, 2),
(8, 2),

-- Team B in Software Development
(9, 2),
(10, 2),
(11, 2),
(12, 2),

-- HR Department
(13, 1),
(14, 1),

-- Support Team
(15, 3),
(16, 3);
GO
```

Components

The