Company Management System (MVC)

This document provides an overview of the Company Management System built using ASP.NET Core MVC. The system facilitates employee, department, project, user, and service management, streamlining operations for HR and managers.

# Team Members:

* + Mohamed Yasser Mohamed Selem
  + Mostafa Elmahdy Atwa
  + Mostafa Yasser Abd El Hameed Eid
  + Mohamed Hassan Elanwar Mohamed Elkhodary Mohamed
  + Mahmoud Ahmed Mahmoud Amin

# Supervisor Name: AST

# System Architecture

The system follows a 3-tier architecture, composed of the following layers:

1. Data Access Layer (DAL): Responsible for interacting with the database, using models to represent employees, departments, projects, users, services, and roles.

2. Business Logic Layer (BLL): Encapsulates the core functionality and business rules. It handles data operations using repositories, unit of work, and generic repository patterns.

3. Presentation Layer (PL): Manages controllers to handle user requests and return appropriate views.

# Controllers and Actions

The system implements several controllers, each responsible for managing different aspects of the company. Below are the key controllers and their actions:

## EmployeeController

* AddEmployee(): Adds a new employee.
* UpdateEmployee(): Edits employee details.
* DeleteEmployee(): Removes an employee.
* ViewEmployee(): Displays employee details.

## DepartmentController

* AddDepartment(): Adds a new department.
* UpdateDepartment(): Edits department details.
* DeleteDepartment(): Removes a department.
* AssignHead(): Assigns a head of the department.

## ProjectController

* AddProject(): Adds a new project.
* UpdateProject(): Edits an existing project.
* DeleteProject(): Removes a project.
* ViewProject(): Displays project details.

## UsersController

* AddUser(): Adds a new user.
* UpdateUser(): Edits user information.
* DeleteUser(): Removes a user.
* ViewUser(): Displays user details.

## ServicesController

* AddService(): Adds a new service.
* UpdateService(): Edits service details.
* DeleteService(): Removes a service.
* ViewService(): Displays service details.

# View Models and Views

The View Model serves as an intermediary between the data and the view, handling data representation, command handling, and validation. Views are implemented to display various data such as employees, departments, users, services, and projects.

# Document Setting

**Upload**

1. Get Located Folder Path

2. Get File Name and Make it Unique

3. Get File Path [Folder Path + FileName]

4. Save File As Streams

5. Return File Name

**Delete**

1. Get File Path

2. Check if File Exists Or Not

If Exists Remove It

# Email Settings

#### SMTP Server

Configure the SMTP server details, including hostname, port, and encryption settings.

#### Authentication

Define authentication credentials, such as username and password, for sending emails.

#### Email Behavior

Set options to customize email sending behavior, like specifying the sender address or enabling email logging.