Task Managment System

Project Documentation

1. Technologies Used

- **C# (.NET Core 8)**: The primary programming language used to build the backend of the application.
- **ASP.NET MVC**: Used to implement the Model-View-Controller pattern, which separates the concerns of the application.
- **Entity Framework Core**: An ORM (Object-Relational Mapper) used for database access and management.
- **SQL Server**: The database system used to store task data, users, and project details.
- **Identity Framework**: Used to handle user admin authorization.
- **Bootstrap**: A CSS framework used to design the frontend and create responsive UI components. [I have used a template from bootswatch webiste]
- **Razor Pages**: Used for rendering views on the server side.

2. Installation Instructions

2.1 Prerequisites

Before running the project, ensure the following are installed on your machine:

- **Visual Studio 2022** (or a compatible version)
- **.NET SDK** (version 8.0 or higher)
- **SQL Server**

2.2 Setting up the Database

Option 1: Attach Database Files

1. Locate the `.mdf` and `.ldf` files for the database.
2. Open **SQL Server Management Studio (SSMS)** and attach the `.mdf` file.
3. Ensure the **connection string** in the project's configuration file matches your SQL Server instance.
4. The application should now be able to connect to the attached database.

Option 2: Using Entity Framework Core Migrations

If you're unable to attach the database files, you can create the database from scratch using migrations:

1. **Review the Server Name**: Ensure the connection string in `appsettings.json` matches your SQL Server instance.
2. Open Visual Studio, and open the **Package Manager Console** (`Tools > NuGet Package Manager > Package Manager Console`).
3. Run the following command to apply the migrations and create the database schema: `Update-Database`

4. This will set up the database schema. Since the database is newly created, you will need to manually enter test data.

2.3 Running the Application

1. Once the database is set up (via migration or attached files), you can run the application .
2. Navigate to the task list and use various filters to search tasks by users, projects, or due dates.

2.4 Admin Access and API

- **Admin Login**: Use the following credentials to log in as an admin:
  - **Email**: admin123@gmail.com
  - **Password**: Admin@123

If you create a new database, you'll need to manually insert a new admin user into the `AspNetUsers` table, as the registration view is hidden by default.

Alternatively, to enable registration from the login page:

1. Go to the login view (`Areas > Identity > Pages > Account > Login.cshtml`).
2. Uncomment the code for the registration button.
3. Once you navigate to the admin page, use the button to register a new admin.

Overdue Task API

You can retrieve overdue tasks using the API:

https://localhost:7240/api/taskapi/overdue/{Count}

Replace `{Count}` with the number of overdue tasks you want to retrieve.