Developing a web application
by using Visual Studio C#, ASP.Net Core MVC and Microsoft SQL Server

I. Case Study

You have been hired to develop a task management web application for a company.

Following are the excerpts from the ***functional requirements*** of the application and the assumption:

1. The application allows the *Employees* to create tasks.
2. Each task can be assigned for one single employee.
3. Each employee can have more than one task assigned.
4. The application can show all the tasks and also all the tasks for a specific employee.
5. The employee can delete and edit the tasks.

Data:

User table:

UserId	FullName	JobTitle	Password	Email
1000	Jhon Jones	Developer	123456	jhon@gmail.com
1001	Peter Joran	Developer	123456	peter@yahoo.com

Task table:

TaskId	TaskDescription	IsDone	Deadline	Assigned
IMI101	Create Database Tables	False	19/12/2022	1000
IMI102	Create Frontend	False	17/12/2022	1001

II. Technical Requirements

You should use the following technique to develop the application

- .Net Core MVC
  - Creating and Using Models
  - Creating and Using Controllers
  - Creating and Using Views
  - Creating and Using ViewModels
- Using SQL Server Database in .Net Core MVC
- Using **Entity Framework** in .Net Core MVC
- Data Annotations
- Routing in MVC
- Design should be responsive and add some style to pages.

Part 1: Database Design and Implementation

For each table, specify Primary Key, data type. The relationship among the tables is also included.

- All the fields are required
- Task Description must not be longer than 100 chars.

Create the models, the migration and create the database from your models.

Part 2: Login Page

The user can only access the application if logged. Create a login page that will provide the access to the user.

Part 3: Display all the tasks.

The index page should show in a table all the tasks and its information. Each task must have two buttons "Edit" and "Delete"

At the Nav-bar, one new link must be added called "My Tasks", this link will redirect the user to see only his/her tasks, exactly as on index page but only tasks assigned to the logged user will be shown.

Part 4: Create Task Page

Create Task Page must be called when a button named "Create Task" is clicked on the index page. The create task page must have a form with all validations for the task model.

If the form is valid, the user must return to index page, if not valid the user should stay at creation page and see the validation errors.

Part 5: Edit Task

Edit the task page that must be called when the user presses the button "Edit" on the task. The page with a form should be called and the user can edit all the information in a task, but its ID.
If the form is valid, the user must return to the index page; otherwise, the user should be in the creation page and the validation errors should be displayed to him/her.

Part 6: Delete Task

Delete Task page must be called when the user presses the button "Delete" on the task. A page with all the information for that task must be called and a question "Are you sure that you want to delete this task?"
If the user presses "Yes" delete the task and go back to index page, if the user presses "No" go back to the index page too.