All Assignments are Mandatory

Home Assignment 1 – Basic Task Manager Credits 10

Objective

This assignment is designed to evaluate fundamental development skills in both backend and frontend technologies. It aims to assess the candidate's ability to build a simple full-stack application using C# (.NET 8) and React with TypeScript.

Functional Requirements

- Display a list of tasks
- Add a new task with a description
- Mark a task as completed or uncompleted
- Delete a task

Backend Requirements (C# .NET 8)

- Implement a RESTful API using .NET 8 Core
- Use in-memory data storage (no database required)
- Define a `TaskItem` model with the following properties:

```
public class TaskItem {
public Guid Id { get; set; }
public string Description { get; set; }
public bool IsCompleted { get; set; }
}
```

Expose the Following Endpoints

- `GET /api/tasks`
- `POST /api/tasks`
- `PUT /api/tasks/{id}`
- `DELETE /api/tasks/{id}`

Frontend Requirements (React + TypeScript)

- Implement a single-page application using React

- Display all tasks in a list
- Provide UI for:

  - Adding a task
  - Toggling completion status
  - Deleting a task
  - Use Axios or Fetch for API integration
  - Use React Hooks for state management

Time Estimate

3–6 hours

Enhancements

Task filtering (All / Completed / Active)
Basic design using a framework such as Bootstrap or Tailwind
Save tasks in localStorage

Home Assignment 2 – Mini Project Manager — Credits 20

Objective

This advanced assignment assesses a candidate's ability to implement a more comprehensive full-stack web application, including user authentication, entity relationships, routing, and modular code structure.

Business Context

Build a minimal project management system where users can register, log in, create projects, and manage tasks within those projects.

Core Features

Authentication

- User registration and login using JWT (JSON Web Tokens)
- After login, users can access only their own data

Projects

Each user can manage multiple projects. A project includes:

- Title (required, 3–100 characters)

- Description (optional, up to 500 characters)
- Creation date (set automatically)

Tasks

Each project can have multiple tasks. Each task includes:

- Title (required)
- Due date (optional)
- Completion status
- A reference to its parent project

Backend Requirements (C# .NET 8)

- Build a REST API with .NET 8 Core and Entity Framework Core
- Use either in-memory storage or SQLite
- Implement authentication using JWT
- Use DataAnnotations for input validation
- Apply separation of concerns (e.g., DTOs, services, models)

Endpoints:

Auth:

- `POST /api/auth/register`
- `POST /api/auth/login`

Projects:

- `GET /api/projects`
- `POST /api/projects`
- `GET /api/projects/{id}`
- `DELETE /api/projects/{id}`

Tasks:

- `POST /api/projects/{projectId}/tasks`
- `PUT /api/tasks/{taskId}`
- `DELETE /api/tasks/{taskId}`

---

pathlock

Frontend Requirements (React + TypeScript)

- Implement a web application with at least the following pages:

. Login/Register
. Dashboard (list of projects)
. Project details (including task list)

- Functionality includes:

. Create and delete projects
. Add, update, and delete tasks
. Toggle task completion
. Form validation and error handling
. Store and reuse JWT for authenticated requests
. Use React Router for navigation

Time Estimate

8–12 hours

Required Enhancements Linked with Mini Project Manager

Smart Scheduler API Credits 10

- **Design and implement an endpoint that helps users plan their work automatically.**

Endpoint Example:

POST /api/v1/projects/{projectId}/schedule

Input Example:

{"tasks": [{ "title": "Design API", "estimatedHours": 5, "dueDate": "2025-10-25", "dependencies": [] },{ "title": "Implement Backend", "estimatedHours": 12, "dueDate": "2025-10-28", "dependencies": ["Design API"] },{ "title": "Build Frontend", "estimatedHours": 10, "dueDate": "2025-10-30", "dependencies": ["Design API"] },{ "title": "End-to-End Test", "estimatedHours": 8, "dueDate": "2025-10-31", "dependencies": ["Implement Backend", "Build Frontend"] }] }

Output Example:

{"recommendedOrder": ["Design API","Implement Backend","Build Frontend","End-to-End Test"],"

- **Loading indicators and user feedback**
- **Mobile-friendly design**
- **Deployment (e.g., backend on Render, frontend on Vercel)**

***Important Note:** Submission should not be done **via a .zip archive with the code**. It should be in a repo hosted in GitHub/GitLab with clear code structure etc.
There should be a note about a bonus task in the assignment which instructs the candidates to deploy their apps and share a link to that deployment.