**Assignment Task: Create a Todo Application**

**Objective**

The goal of this assignment is to apply the concepts learned in the recent training on Microservices, ASP.NET Core 6 +, and related technologies to build a functional Todo application. This will include implementing API endpoints, managing data flow, and ensuring secure communication.

**Requirements**

* **Platform**: ASP.NET Core 6
* **API Specification**: Open API (Swagger)
* **Security**: Secure API’s by using JWT Token’s.
* **Duration: 3 days**

**Application Features**

1. **User Authentication**:
   * Implement user authentication using ASP.NET Identity.
   * Users should be able to register, log in, and manage their session.
2. **Todo Management**:
   * **Create Todo**: Users can add new todos with details such as title, description, and due date.
   * **Read Todo**: Users can view all their todos in a list, as well as view details of a specific todo.
   * **Update Todo**: Users can modify the details of existing todos.
   * **Delete Todo**: Users can remove todos that are no longer needed.

**Development Tasks**

1. **Environment Setup**:
   * Set up an ASP.NET Core 6 project with the necessary configurations and dependencies.
2. **API Development**:
   * Define and implement routes using ASP.NET Core routing features.
   * Create controllers for handling the HTTP verbs: GET, POST, PUT, and DELETE.
   * Ensure your API handles different HTTP status codes appropriately.
3. **Data Handling**:
   * Implement the Repository Pattern for data access.
   * Use Entity Framework for ORM, focusing on asynchronous methods for database operations.
4. **Security Implementations**:
   * Configure SSL for HTTPS.
   * Secure your application using the ASP.NET Identity for user authentication and manage secure sessions.
5. **Testing and Documentation**:
   * Document your API using Swagger to provide a clear overview of API capabilities and endpoints.

**Deliverables**

* **Source Code**: A GitHub repository or Project solution with the complete source code.
* **API Documentation**: A link to the Swagger UI where the API documentation is accessible.

**Evaluation Criteria**

* **Functionality**: The application meets all the functional requirements.
* **Code Quality**: Code is well-organized, properly commented, and adheres to best practices.
* **Error Handling**: Appropriate handling of different error scenarios, including network issues and server errors.

This assignment will help you solidify your understanding of building modern web APIs and handling real-world scenarios in web development. Good luck!