****

**Multi-user Blogging Platform**

**Introduction**

You are tasked with building a multi-user blogging platform where users can create accounts, write and publish blog posts, view posts by other users, and leave comments.

Assignment Duration: 3 days

**Day 1: Database and Backend (22)**

Task 1.1: Setup the Database **(5)**

Create a database named "BlogDB". Create four tables: Users, Posts, Comments, and Categories.

Task 1.2: Stored Procedures and Queries **(5)**

Write SQL stored procedures for each CRUD operation on the Users, Posts, Comments, and Categories tables. Also, develop complex queries for fetching posts with their corresponding comments, fetching posts by category, and fetching posts by user.

Task 1.3: ADO.NET **(2)**

Use ADO.NET to establish a connection from your ASP.NET application to your SQL Server database. Use this connection to execute your stored procedures and manipulate the database from your application.

Task 1.4: Web APIs **(10)**

Develop Web APIs to expose the functionality of your application to the frontend. You will need APIs for user registration and login, creating, reading, updating, and deleting posts, creating and deleting comments, and viewing posts by category or user.

**Day 2: Frontend Development and Integration (15)**

Task 2.1: HTML/CSS **(5)**

Create the frontend for your application using HTML and CSS. You will need pages for user registration and login, viewing all posts, viewing a single post with its comments, creating a new post, and editing a post.

Task 2.2: jQuery **(5)**

Make your frontend dynamic and interactive with jQuery. Use Ajax to communicate with your APIs, dynamically update the DOM to display the fetched data, and handle user interactions such as form submissions and button clicks.

Task 2.3: Frontend and Backend Integration **(5)**

Connect your frontend and backend by using jQuery to call your Web APIs, passing the appropriate data and handling the responses. Make sure all functionalities of the application are working correctly.

**Day 3: Advanced Concepts and Testing (13)**

Task 3.1: Implementing Advanced SQL Queries **(3)**

Develop advanced SQL queries to provide complex functionality such as searching posts by keywords, sorting posts by date or popularity (number of comments), and recommending similar posts to users.

Task 3.2: Applying OOP Concepts **(2)**

Ensure that your C# code follows OOP principles. This includes proper use of classes and objects, inheritance, encapsulation, abstraction, and polymorphism. For example, you could create classes for User, Post, Comment, and Category with appropriate properties and methods.

Task 3.3: Data Structures **(3)**

Incorporate data structures where they would be beneficial. For instance, you could use a Dictionary to map categories to posts for efficient lookup, or a List to store and manipulate a collection of Comment objects in a Post.

Task 3.4: Performance Optimization **(2)**

Optimize your application for better performance. This could involve optimizing your SQL queries, minimizing the amount of data transferred between the frontend and backend, or improving the efficiency of your C# code.

Task 3.5: Comprehensive Testing **(3)**

Finally, thoroughly test all aspects of your application, from individual functionalities to the overall user experience. Ensure that the application handles incorrect input gracefully, displays useful error messages, and doesn't crash under unexpected conditions.

This project will provide you with a comprehensive exercise on various aspects of web development, including database design, backend development with ASP.NET and ADO.NET, frontend development with HTML, CSS, and jQuery, Web APIs, OOP principles, data structures, and testing.