Day 1: SQL Database Operations

Topics:

- Database concepts and normalization
- Creating, altering, and dropping databases
- Creating, renaming, and dropping tables
- Data types and constraints (Foreign Key, Unique, Check)
- Insert, update, and delete operations
- Select statements, sorting, and grouping
- Data retrieval from multiple tables

Lab Problems:

1. **Student Database:** Create a SQL database for a student management system. Include tables for students, courses, and enrollments.
2. **SQL Data Queries:** Write SQL queries to insert student records, update their course enrollments, and delete student entries if they leave the program.
3. **JOIN Queries:** Use JOIN operations to display which students are enrolled in which courses, along with sorting by student names.

Day 2: C# and Object-Oriented Programming (OOP)

Topics:

- Introduction to C# and OOP concepts (class, object, inheritance, polymorphism)
- Conditional and looping constructs
- Exception handling
- File operations
- N-tier architecture and debugging techniques
- Collections and LINQ
- DB Connection Using ADO.NET
- Using SqlConnection, SqlCommand, SqlDataAdapter
- DB Connection Using Entity Framework

Lab Problems:

1. **Library System:** Implement a C# program that models a library with classes for Book, Member, and Transaction. Allow members to borrow and return books.
2. **File Handling:** Create a C# program that reads data from a text file containing student names and writes their attendance status into another file.

3. **LINQ Queries:** Write a C# program to filter and display a list of books whose titles contain the word "Programming" using LINQ.

Day 3: Working with ASP.NET

- What is MVC?
- Hello World with MVC
- View/Partial View in MVC
- Models, View and Controller
- Web API
- Routing in MVC
- View Bag vs View Data

Lab Project

1. Create a MV Project
2. Add /Edit/Delete/Listing Records including Search and Pagination and Sorting

Day 4: Async Programming in .NET Core & ASP.NET Core Basics

- **Topics**:
    1. **Async Programming in .NET Core**:
        - Overview of synchronous vs asynchronous programming.
        - Task-based Asynchronous Pattern (TAP).
        - Implementing async/await in .NET Core.
        - Handling exceptions and cancellation tokens in async operations.
        - Performance considerations with asynchronous programming.
    2. **ASP.NET Core Basics**:
        - ASP.NET Core project structure.
        - The request processing pipeline.
        - Middleware, routing, and services.
        - Dependency Injection (DI) in ASP.NET Core.
        - Creating simple MVC controllers and views.
        - Building basic APIs using controllers and action methods.
- **Lab**:
    - Create an ASP.NET Core application that performs both synchronous and asynchronous operations for HTTP calls and file I/O.
    - Build basic MVC controllers and a simple API endpoint.

Day 5: Entity Framework (EF) Core & Security in .NET Core

- **Topics**:
    1. **Entity Framework Core**:
        - Introduction to EF Core and ORM (Object-Relational Mapping).
        - Code-first and database-first approaches.
        - Configuring DbContext and defining models.
        - LINQ for querying data.
        - Relationships (one-to-one, one-to-many, many-to-many).
        - Migrations and database updates.
    2. **Security in .NET Core**:
        - ASP.NET Core Identity for user authentication.
        - JWT (JSON Web Token) for securing Web APIs.
        - Role-based and policy-based authorization.
        - Protecting data with encryption and Data Protection API.
        - Best practices for securing .NET Core applications (e.g., HTTPS, CORS, CSRF).
- **Lab**:
    - Build a simple CRUD application using EF Core to manage a "Customer" and "Order" system.
    - Implement JWT-based authentication to secure the API.
    - Implement role-based authorization to restrict access to specific actions.

Day 6: Hosting .NET Core Apps with IIS & Docker

- **Topics**:
    1. **Hosting with IIS**:
        - Overview of web hosting and IIS.
        - Setting up IIS for hosting ASP.NET Core applications.
        - Configuring application pools, environment variables, and bindings.
        - Deploying applications to IIS.
    2. **Docker for .NET Core**:
        - Introduction to Docker and its use in microservices architecture.

        - Containerizing .NET Core applications.
        - Creating Dockerfiles and working with .dockerignore.

- **Lab**:
    - Deploy an ASP.NET Core application on IIS, configure the hosting environment, and troubleshoot issues.
    - Dockerize the ASP.NET Core API and run it inside a container