AWD

DOT NET CORE API **TOPICS**

C# - Topics

- Data Type
- Variables
- Console Output/Input
- Decision Statements (if, Switch)
- Ternary Operator
- Loops (for, while, do....while)
- Break/Continue

- Array (one dimensional, two dimensional)
- Foreach loop
- Type Casting
- Boxing and Unboxing
- Exception Handling
- Enum

C# OOPS Concepts

- Introduction to OOPS
- Classes & Objects
- Functions
- Constructors
- Inheritance & Its types
- Interface
- Polymorphism
- Method Overloading & Overriding
- Abstraction
- Encapsulation

Non-Generic Collections

- Array List
- Sorted List
- Stack
- Queue

- Hash table

Generic Collections

- List
- Dictionary
- Sorted List
- Queue
- Stack
- Introduction to dependency injection
- Introduction to GIT

INTRODUCTION TO .NET CORE API

- Overview of .NET Core and its features.
- Differences between .NET Framework and .NET Core.
- Installing .NET Core SDK.
- Creating a new .NET Core Web API project using Visual Studio or the CLI.
- Project structure and files.
- Understanding routing in .NET Core.
- Creating controllers and defining actions.
- Attribute routing vs. convention-based routing.

MODEL BINDING , VALIDATION AND DEPENDENCY INJECTION (DI)

- Binding parameters from the request.
- Data annotations for model validation.
- Custom validation attributes.
- Built-in Dependency Injection in .NET Core.
- Registering services in the Startup class or Program.cs .
- Scoped, transient, and singleton life cycle methods of dependency injection.

Entity Framework Core , Dapper ,View Models and Data Transfer Objects

- Setting up EF Core in a .NET Core project.
- Creating DbContext and entity classes.
- -Creating API with Entity Framwork Core
- -Creating API with Dapper
- Performing CRUD operations or POST, GET, PUT, and DELETE.
- Migrations and database updates.
- -Data Seeding
- -DataModels and View Models
- -Introduction to Auto Mapper for object to object maping with data transfer objects

AUTHENTICATION AND AUTHORIZATION

- Overview of authentication and authorization.
- Implementing JWT authentication.
- Role-based and policy-based authorization.

Error Handling and Logging

- Global exception handling middleware.
- Custom error responses.
- Logging with built-in logging providers.

Middleware

- Understanding the middleware pipeline.
- Creating custom middleware.
- Using built-in middleware for common tasks .

Versioning

- Strategies for API versioning.
- Implementing versioning using URL, query string, or headers.

Testing

- Unit testing controllers and services.
- Using mocking frameworks.

Documentation with Swagger/Open API

- Setting up Swagger in a .NET Core API.
- Customizing Swagger documentation.

- Generating client code from Swagger.

Performance Optimization

- Best practices for optimizing API performance.

- Caching strategies.

- Asynchronous programming with async/await.

Security Best Practices

- Securing API endpoints.

- HTTPS and data encryption.

Deployment

- Hosting .NET Core API .

- Continuous Integration/Continuous Deployment (CI/CD) with Azure DevOps or GitHub Actions.

- Monitoring and diagnostics.

Advanced Topics

- Real-time communication with SignalR.

Consuming APIs in Blazor

Introduction to Blazor

- Overview of Blazor and its hosting models (Blazor Server vs. Blazor WebAssembly).

- Setting up a Blazor project.

HTTP Client in Blazor

- Configuring HttpClient in Blazor.
- Dependency injection for HttpClient.

Making API Calls

- GET, POST, PUT, DELETE requests.
- Handling JSON data.
- Async programming with HttpClient.

Authentication and Authorization

- Authenticating users in Blazor.
- Passing authentication tokens in API requests.
- Implementing role-based and policy-based authorization.

State Management

- Managing application state in Blazor.
- Sharing state between components.
- State persistence strategies.

Error Handling

- Handling HTTP errors in Blazor.
- Displaying user-friendly error messages.

Data Binding and Component Interaction

- Binding data to UI components.
- Component-to-component communication.
- Parent-child component relationships.

Forms and Validation

- Building forms in Blazor.
- Form validation using data annotations.
- Custom validation logic.

Routing and Navigation

- Defining routes in Blazor.
- Navigating between components.
- Passing parameters in routes.

Real-time Updates with SignalR

- Integrating SignalR in Blazor for real-time updates.
- Implementing server-side notifications.

File Uploads and Downloads

- Handling file uploads in Blazor.
- Downloading files from API endpoints.

Testing Blazor Components

- Unit testing Blazor components.
- Integration testing with bUnit or other testing frameworks.

Optimizing Blazor Applications

- Performance optimization techniques.
- Lazy loading of components.
- Minimizing HTTP requests.

Deployment of Blazor Applications

- Deploying Blazor WebAssembly and Blazor Server apps.
- Configuring hosting environments.