Developing ASP.NET Core MVC Web Applications

20486D - Version: 1

5 days Course

Description:

In this course, students will learn to develop advanced ASP.NET Core MVC applications. The focus will be on coding activities that enhance the performance and scalability of the Web site application.

Intended audience:

This course is intended for professional web developers who use Microsoft Visual Studio in an individual-based or team-based, small-sized to large development environment. Candidates for this course are interested in developing advanced web applications and want to manage the rendered HTML comprehensively. They want to create websites that separate the user interface, data access, and application logic.

Prerequisites:

Experience with Visual Studio 2017.
Experience with C# programming, and concepts such as Lambda expressions, LINQ, and anonymous types.
Experience in using the .NET Framework.
Experience with HTML, CSS and JavaScript.
Experience with querying and manipulating data with ADO.NET.
Knowledge of XML and JSON data structures.

Objectives:

Describe the Microsoft Web Technologies stack and select an appropriate technology to use to develop any given application.
Design the architecture and implementation of a web application that will meet a set of functional requirements, user interface requirements, and address business models.

Configure the pipeline of ASP.NET Core web applications using middleware, and leverage dependency injection across MVC application.
Add Controllers to an MVC Application to manage user interaction, update models, and select and return Views.
Develop a web application that uses the ASP.NET Core routing engine to present friendly URLs and a logical navigation hierarchy to users.
Create Views in an MVC application that display and edit data and interact with Models and Controllers.
Create MVC Models and write code that implements business logic within Model methods, properties, and events.
Connect an ASP.NET Core application to a database using Entity Framework Core.
Implement a consistent look and feel across an entire MVC web application.
Write JavaScript code that runs on the client-side and utilizes the jQuery script library to optimize the responsiveness of an MVC web application.
Add client side packages and configure Task Runners
Run unit tests and debugging tools against a web application in Visual Studio 2017.
Write an MVC application that authenticates and authorizes users to access content securely using Identity.
Build an MVC application that resists malicious attacks.
Use caching to accelerate responses to user requests.
Use SignalR to enable two-way communication between client and server.
Describe what a Web API is and why developers might add a Web API to an application.
Describe how to package and deploy an ASP.NET Core MVC web application from a development computer to a web server.

Topics:

Exploring ASP.NET Core MVC

- Overview of Microsoft Web Technologies
- Overview of ASP.NET 4.x
- Introduction to ASP.NET Core MVC

Designing ASP.NET Core MVC Web Applications

- Planning in the Project Design Phase
- Designing Models, Controllers and Views
- Designing ASP.NET Core MVC Web Applications

Configure Middleware and Services in ASP.NET Core

- Configuring Middleware
- Configuring Services
- Configuring Middleware and Services in ASP.NET Core

Developing Controllers

- Writing Controllers and Actions
- Configuring Routes
- Writing Action Filters
- Developing Controllers

Developing Views

- Creating Views with Razor Syntax
- Using HTML Helpers and Tag Helpers
- Reusing Code in Views
- Developing Views

Developing Models

- Creating MVC Models
- Working with Forms
- Validating MVC Application
- Developing Models

Using Entity Framework Core in ASP.NET Core

- Introduction to Entity Framework Core
- Working with Entity Framework Core

- Using Entity Framework Core to Connect to Microsoft SQL Server
- Using Entity Framework Core in ASP.NET Core

Using Layouts, CSS and JavaScript in ASP.NET Core MVC

- Using Layouts
- Using CSS and JavaScript
- Using jQuery
- Using Layouts, CSS and JavaScript in ASP.NET Core MVC

Client-Side Development

- Applying Styles
- Using Task Runners
- Responsive Design
- Client-Side Development

Testing and Troubleshooting

- Testing MVC Applications
- Implementing an Exception Handling Strategy
- Logging MVC Applications
- Testing and Troubleshooting

Managing Security

- Authentication in ASP.NET Core
- Authorization in ASP.NET Core
- Defending from Attacks
- Managing Security

Performance and Communication

- Implementing a Caching Strategy
- Managing State

- Two-Way Communication
- Performance and Communication

Implementing Web APIs

- Introducing Web APIs
- Developing a Web API
- Calling a Web API
- Implementing Web APIs

Hosting and Deployment

- On-premise hosting and deployment
- Deployment to Microsoft Azure
- Microsoft Azure Fundamentals
- Hosting and Deployment