Iverson Associates Sdn Bhd (303330-M)
Suite T113 – T114, 3rd Floor, Centrepoint, Lebuh Bandar Utama
Bandar Utama, 47800 Petaling Jaya, Selangor Darul Ehsan
Tel: 03-7726 2678 Fax: 03-7727 9737 Website: www.iverson.com.my

Module Title : ASP.NET Core 5 for Beginners
Duration : 5 days

WHAT YOU WILL LEARN

ASP.NET Core is a powerful and effective framework that's open source and cross-platforms. It helps you build cloud-ready, modern applications, such as web apps and services.

This course is designed with hands-on exercises and projects. It is an easy-to-follow guide that will teach you how to develop using the ASP.NET Core 5 framework. You'll learn about the framework using C# 8, Visual Studio 2019, Visual Studio Code, Razor Pages, Blazor, IIS, Apache, Docker, Azure and more.

In this course, you'll learn how to write applications, build websites, and deploy your web apps to cloud. You will thoroughly explore your coding environment and recommended best practices, and we'll provide code samples to systematically cover the top scenarios that you'll face in the industry today. By the end of this course, you'll be able to leverage ASP.NET Core 5 to build and deploy web applications and services in a variety of real-world scenarios.

AUDIENCE

This course is for developers who want to learn how to develop web-based applications using the ASP.NET Core framework. Familiarity with the C# language and a basic understanding of HTML and CSS is required to get the most out of this course.

PREREQUISITES

- Basic web related technology knowledge.
- Basic knowledge and skill in C# programming.

METHODOLOGY

This program will be conducted with interactive lectures, PowerPoint presentation, discussions and practical exercise.

COURSE OUTLINE

Module 1: .NET Core Essentials

- The different between .NET and .NET Core
- A brief history about .NET Core
- Coming version of .NET
- The direction of .NET Core

Module 2: Introduction to ASP.NET Core 5

- Technical requirements
- Explaining ASP.NET Core
- Refreshing your C# knowledge
- Understanding websites and web servers
- Exploring Visual Studio Code
- Leveraging Windows Terminal

Module 3: Cross-Platform Setup

- Technical requirements
- Leveraging the .NET Framework
- Getting started on Windows, Linux, and macOS
- Debugging Linux on Windows with Visual Studio 2019

Module 4: Dependency Injection

- Technical requirements
- Learning dependency injection in ASP.NET Core
- Reviewing types of dependency injection
- Understanding dependency injection containers
- Understanding dependency lifetimes
- Handling complex scenarios

Module 5: Razor View Engine

- Technical requirements
- Understanding the Razor view engine
- Learning the basics of Razor syntax
- Building a to-do application with MVC
- Building a to-do app with Razor Pages
- Differences between MVC and Razor Pages

Module 6: Getting Started with Blazor

- Technical requirements
- Understanding the Blazor web framework
- Creating the backend applications

Module 7: Exploring the Blazor Web Framework

- Creating the Blazor Server project
- Creating the Blazor Web Assembly project

Module 8: APIs and Data Access

- Data Access Techniques in brief
- The Model, the role in MVC

Module 9: Working with Identity in ASP.NET

- Technical requirements
- Understanding authentication concepts
- Understanding authorization concepts
- The role of middleware in ASP.NET and identity
- OAuth and OpenID Connect
- Integrating with Azure Active Directory
- Working with federated identity

Module 10: Getting Started with Containers

- Technical requirements
- Overview of containerization
- Getting started with Docker
- Running Redis on Docker
- Running ASP.NET Core in a container

Module 11: Deploying to the Cloud

- Technical requirements
- Overview of cloud computing
- Creating a sample ASP.NET Core web application
- Publishing to Cloud

Module 12: Browser and Visual Studio Debugging

- Technical requirements
- Setting up the sample application
- Using debugging tools in the browser
- Debugging in Visual Studio

Module 13: Build microservice with .NET Core

- Introduction
- What are microservices?
- Build a Dockerfile for your microservice
- Microservices orchestration
- Create a Docker Compose file

Module 14: ASP.NET Core Performance Best Practices

- Cache aggressively
- Avoid blocking calls
- Return large collections across multiple smaller pages
- Minimize large object allocations
- Optimize data access and I/O
- Pool HTTP connections with HttpClientFactory
- Keep common code paths fast
- Complete long-running Tasks outside of HTTP requests
- Minify client assets
- Compress responses
- Minimize exceptions
- Performance and reliability
- Avoid synchronous read or write on HttpRequest/HttpResponse body
- Prefer ReadFormAsync over Request.Form
- Avoid reading large request bodies or response bodies into memory
- Working with a synchronous data processing API
- Do not store IHttpContextAccessor.HttpContext in a field
- Do not access HttpContext from multiple threads
- Do not use the HttpContext after the request is complete
- Do not capture the HttpContext in background threads
- Do not capture services injected into the controllers on background threads
- Do not modify the status code or headers after the response body has started
- Do not call next() if you have already started writing to the response body
- Use In-process hosting with IIS