Progressive Web App Development Using Entity Framework Core and Blazor Training (3 Days)

This Blazor training class will provide students with an understanding of the principles of using Blazor components and templated components, service and dependency injection, layout, and routing. Students will also learn how to enable the progressive web apps functionality in existing and new Blazor applications by creating service workers to support push notifications and access storage using Web Storage API and IndexedDB.

Course Benefits:

- Install and use the Entity Framework package to work with databases on the server-side
- Install packages required to develop the client-side Blazor applications
- Use components, service & dependency injection, layout, and routing
- Enable the Progressive Web Apps features in an application by creating service workers
- Use the Progressive Web Apps features, such as push notifications and storage with IndexedDB

Prerequisites:

Experience in the following *is required* for this ASP.NET class:

- Familiarity with .NET and C#.

Experience in the following *would be useful* for this ASP.NET class:

- Basic knowledge of JavaScript is useful but not necessary.

Course Outline:

Introduction to Blazor, Entity Framework Core, and Progressive Web Apps
- What is Microsoft Blazor
- What is Entity Framework Core
- What is Progressive Web Apps (PWAs)
- Relationship between C#, LINQ, Entity Framework Core, JavaScript, HTML, and Razor
- Comparing Blazor to Angular and React

Installing the Packages and Working with Git
- Blazor
- Entity Framework Core
- ODP.NET for Oracle
- Working with Git on Azure DevOps Services

Getting Started with Entity Framework Core
- Entity Framework Core Overview
- Installing Entity Framework Core
- Modelling using Code First
- Modelling using Database First
- Using Fluent API

Querying and Saving Data with Entity Framework Core
- Basic Queries with LINQ
- Calling Stored Procedures
- Using Include and Then Include
- Controlling the Tracking Behaviour
- Saving changes
- Performing Bulk Insert
- Implementing Transactions

Getting Started with Blazor
- Creating a Blazor Hello World Application
- Understanding the Razor fundamentals

Working with Blazor Components
- Working with Components in Blazor
- Using C# in Components
- Parameterize Components
- Understanding Component Life Cycle
- Binding: One-way data binding
- Binding: Two-way data binding
- Binding: Event binding

Advanced Blazor Component Concepts
- Render Raw HTML
- Render Child Content
- Using RenderFragment
- Using RenderTreeBuilder
- Using &key and &attribute

Layout and Routing
- Creating a Master Layout
- Implementing Routing
- Using URL Helpers

Forms and Validation
- Using EditForm
- Implementing Form Validation
- Using &ref
- Routing

Dependency Injection & JSInterop
- Understanding Dependency Injection (DI)
- DI with Default Service
- DI with Custom Service
- Call a JavaScript Function
- Call C# from JavaScript

Debugging & Deployment
- Debugging a Blazor Application
- Deploying a Blazor Application
- Upgrading a Blazor Application

Securing a Blazor Application
- Blazor Authentication
- Using ASP.NET Core Identity and JWT

Best Practices
- Project Structure
- Optimizing the Startup Time
- Optimizing Rendering Performance
- Optimizing the Application Download Size

Progressive Web Apps (PWAs) Overview
- Understanding the features of a progressive web app
- Getting started with a Blazor PWA
- When to create offline apps
- Using Service Workers
- Customizing the application appearance
- Adding Push Notifications
- Controlling Caching