

**Web API Development with ASP.NET Core 6**

**Course Number:** ASPNC-122  
**Duration:** 4 days

**Overview**

This ASP.NET Core 6 training course teaches attendees how to design and build high-performance, secure, well-architected REST APIs that can be consumed by a variety of clients. This course focuses specifically on APIs and does not include coverage of web UI topics such as Views and Razor Pages. An introduction to Blazor is included but it is not covered in-depth.

**Note:** This 4-day training course can be extended to five days with additional topics related to the C# programming language or Entity Framework Core.

**Prerequisites**

* Previous experience developing web-based applications with C#
* Some familiarity with HTML, CSS, and JavaScript

**Materials**

All ASP.NET Core training students receive comprehensive courseware.

**Software Needed on Each Student PC**

* Windows 10 or later with at least 8 GB RAM
* Visual Studio 2022
* .NET 6.0 SDK
* LocalDB or another version of SQL Server
* If you have purchased this class, please contact us for more detailed setup specifications

**Objectives**

* Understand the goals and benefits of ASP.NET Core 6.0
* Learn to make good decisions about application architecture and data access technology
* Use ASP.NET’s routing system to achieve a REST-style architecture
* Gain experience building a service that makes data available via a modern web API
* Learn best practices for employing unit testing, logging, and error handling
* Understand different authentication choices for securing a web API
* Get an introduction to Blazor and gRPC
* Understand the different cross-platform deployment options available including via Docker containers

**Outline**

* Introduction
  + Evolution of .NET and .NET Core
  + .NET SDKs and Runtimes
  + Visual Studio and Visual Studio Code
* .NET 6.0 SDK
  + Installation
  + Version Management
  + Command-Line Interface (CLI)
* What’s New in C#
  + Record Types
  + Init Only Setters
  + Nullable Reference Types
  + Global Using Directives
  + File-Scoped Namespace Declarations
  + Top-Level Statements
* ASP.NET Core Application Architecture
  + NuGet Packages
  + Application Startup
  + Hosting Environments
  + Middleware and the Request Pipeline
  + Services and Dependency Injection
* Application Configuration
  + Configuration Providers and Sources
  + Configuration API
  + Options Pattern
  + HTTPS and HTTP/2
* Request Routing
  + RESTful Services
  + Endpoint Routing
  + Route Templates
  + Route Constraints
  + Route Template Precedence
  + Attribute-Based Routing
* Models
  + Persistence Ignorance
  + Dependency Inversion
  + Asynchronous Data Access
  + Object-Relational Mapping
  + Entity Framework Core
  + Dapper ORM
* Controllers
  + Responsibilities
  + Requirements and Conventions
  + Dependencies
  + Action Results
  + ApiController Attribute
* Web APIs
  + API Controllers
  + Minimal APIs
  + OpenAPI / Swagger
  + Testing APIs
  + Content Negotiation
  + CRUD Operations
  + Patch Requests
  + Microservice Architecture
  + API Gateway Pattern
  + Cross-Origin Resource Sharing (CORS)
* Data Validation
  + Introduction
  + Data Annotations
  + Model Binding
* Error Handling
  + Best Practices
  + HTTP Error Status Codes
  + Exception Handling Middleware
* Logging
  + Configuration
  + ILogger
  + Serilog and Seq
* Testing
  + Unit Testing
  + xUnit
  + Testing Controllers
  + Integration Testing
* Security
  + Authentication
  + ASP.NET Identity
  + Authorization
  + Web API Authentication
  + JSON Web Tokens (JWT)
  + OAuth 2.0 and OpenID Connect
  + Secrets Management
* Remote Procedure Calls (gRPC)
  + Introduction
  + Protobuf
  + Server
  + Client
  + Limitations
* Blazor
  + Razor Components
  + Blazor Server
  + Blazor WebAssembly
* Deployment
  + dotnet publish
  + Kestrel
  + IIS
  + Docker
* Conclusion