Microservices with .NET and ASP.NET Core

Create independently deployable, highly scalable, and resilient services using the free and open-source .NET platform.

Prerequisites

- Familiarity with command-line based applications.
- Basic understanding of application and system architecture
- Familiarity with basic Docker concepts and Azure Development.
- Azure free subscription is required
- Experience writing C# at the beginner level

Duration: 4 days

Module 1: Introduction to Microservices

In this module you will learn the basic concepts and principles of microservices architecture, which is a modern way of designing and developing software applications as a collection of loosely coupled, independent, and scalable services.

- What are microservices?
- Microservices vs monolithic architecture
- Advantages and disadvantages of microservices
- Overview of .NET Core 6

Module 2: Designing Microservices Architecture

In this module, you will learn about the principles of microservices design. You will also learn about domain-driven design (DDD) and bounded contexts, which are essential for identifying and modeling the domains and subdomains of your application. You will also learn how to define service boundaries and interfaces, which are the contracts between your services and their consumers. You will also learn how to deal with data consistency and transactions across microservices, which are challenging due to the distributed nature of microservices

- Principles of microservices design
- Domain-driven design (DDD) and bounded contexts
- Service boundaries and interfaces
- Data consistency and transactions

Module 3: Building Microservices with .NET Core 6

Building Microservices with .NET Core 6 is a module that aims to teach the practical skills and techniques of developing microservices using .NET Core 6.

- Setting up a new .NET Core 6 project
- Implementing the service layer with MVC
- Using Entity Framework Core to connect to a database
- Testing individual microservices

Module 4: Communication between Microservices

In this module, you will learn about the communication patterns between microservices, such as synchronous and asynchronous, request-response and event-driven, and point-to-point and publish-subscribe. You will also learn about RESTful APIs and HTTP requests, which are the most common ways of implementing synchronous communication between microservices. You will also learn how to use the HttpClient class in .NET Core, which is a modern and efficient way of sending and receiving HTTP requests in C#. You will also learn how to version and document your APIs, which are important for maintaining compatibility and usability of your microservices.

- Communication patterns between microservices
- RESTful APIs and HTTP requests
- Using the HttpClient class in .NET Core
- Versioning and documentation of APIs

Module 5: Overview of microservice with .NET

Microservice applications are composed of small, independently versioned, and scalable customer-focused services that communicate with each other over standard protocols with well-defined interfaces. Each microservice typically encapsulates simple business logic, which you can scale out or in, test, deploy, and manage independently. Smaller teams develop a microservice based on a customer scenario and use any technologies that they want to use. This module will teach you how to build your first microservice with .NET.

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

Module 6: Decompose a monolithic application into a microservices architecture

Improve development agility, deployment, fault tolerance, and scalability by modernizing your application with a microservices architecture.

- Introduction
- Monolithic and microservice architectures
- Exercise - Deploy a monolithic application on App Service
- Performance constraints of a monolithic application
- Analyze an application and identify decomposition boundaries
- Exercise - Refactor a service within the monolith as a microservice

Module 7: Deploy a .NET microservice to Kubernetes

Microservice applications are composed of small, independently versioned, and scalable customer-focused services. Microservices applications deployed in containers make it possible to scale out apps, and respond to increased demand by deploying more container instances, and to scale back if demand is decreasing. In complex solutions of many microservices the process of deploying, updating, monitoring, and removing containers introduces challenges. This module explains some of those challenges and shows how Kubernetes can help.

- Introduction
- What are orchestrators?

- Exercise - Push a microservice image to Docker Hub
- Exercise - Deploy a microservice container to Kubernetes
- Exercise - Scale a container instance in Kubernetes
- Exercise - Prove microservice resilience in Kubernetes

Module 8: Create and deploy a cloud-native ASP.NET Core microservice

Create and deploy an ASP.NET Core microservice to AKS.

- Introduction
- Review the solution architecture
- Exercise - Deploy the application
- Exercise - Verify the deployment
- Review the coupon service design
- Exercise - Add the coupon service
- Exercise - Build and deploy the changes

Module 9: Implement resiliency in a cloud-native ASP.NET Core microservice

Learn how to make your cloud-native ASP.NET Core microservices app fault-tolerant with minimal impact on the user.

- Introduction
- Exercise - Set up the environment
- Review resiliency concepts
- Exercise - Verify deployment and test the app
- Exercise - Implement code-based resiliency
- Exercise - Implement infrastructure-based resiliency

Module 10: Instrument a cloud-native ASP.NET Core microservice

Learn how to instrument your cloud-native ASP.NET Core microservices app to diagnose problems and monitor performance.

- Introduction
- Exercise - Set up the environment
- Review logging and monitoring concepts
- Exercise - Implement Application Insights
- Exercise - Monitor Application Insights
- Exercise - Implement Azure Monitor for Containers

Module 11: Implement feature flags in a cloud-native ASP.NET Core microservices app

Implement a feature flag in your cloud-native ASP.NET Core microservices app to enable or disable a feature in real time.

- Introduction
- Exercise - Set up the environment
- Review app configuration concepts
- Exercise - Implement the Feature Management library
- Exercise - Implement the Azure App Configuration service

Module 12: Use managed data stores in a cloud-native ASP.NET Core microservices app

Modify a cloud-native ASP.NET Core microservices app to use managed data stores in Azure.

- Introduction
- Exercise - Set up the environment
- Review managed data stores in Azure
- Exercise - Verify deployment and test the app
- Exercise - Implement Azure Cache for Redis
- Exercise - Implement Azure Cosmos DB

Module 13: Understand API gateways in a cloud-native ASP.NET Core microservices app

- Introduction
- Exercise - Set up the environment
- Exercise - Verify deployment and test the app
- Understand API gateways and Backends for Frontends
- Implement a new Backend for Frontend
- Understand Kubernetes ingress controller concepts
- Exercise - Implement a load balancer with Azure Application Gateway

Module 14: Deploy a cloud-native ASP.NET Core microservice with GitHub Actions

Use a GitHub Actions CI/CD pipeline to build a container image and deploy it to Azure Kubernetes Service (AKS).

- Introduction
- Exercise - Set up the environment
- Exercise - Configure GitHub Actions permissions and secrets
- Exercise - Create a GitHub action to build a container image
- Exercise - Create a GitHub action to deploy to AKS