Microservices using ASP.NET Core Web API

New Batch: 20th February 2025, 9 PM to 10.30 PM IST.
Registration Link: https://forms.gle/7yiT3tZHrhMbWDTE9
Telegram Group: https://telegram.me/microservicesusingaspnetcore
Contact us through Our WhatsApp Number: +91 7021801173
Call us for More Details: +91 7021801173
Duration: 3 Months
Demo Sessions: 20, and 21 February, 2025, from 9 PM to 10.30 IST

Detailed Course Syllabus

Module 1: Fundamentals & Architectural Patterns

Chapter 1: Introduction to Monolithic vs. Microservices Architectures

Objective: Students will understand the evolution of application architectures from monolithic systems to distributed microservices and learn the benefits, trade-offs, and drivers behind architectural decisions.

Topics Covered:

- **Monolithic Applications: Definition, structure, benefits, and drawbacks**
- **Containerizing & Scaling Monoliths: A high-level overview**
- **Introduction to Microservices: Characteristics and motivations**
- **Comparing Architectures: Key differences, trade-offs, and migration drivers**

Chapter 2: Microservices Design Principles & Patterns

Objective: Students will learn core design principles for building robust microservices, including the fundamentals of Domain-Driven Design (DDD), common microservices patterns, and practices that support scalability and maintainability.

Topics Covered:

- **Characteristics, benefits, and drawbacks of microservices**
- **Design Principles: Loose coupling, bounded contexts, and independent deployability**
- **Domain-Driven Design (DDD) Fundamentals**
- **Common Patterns: Service discovery, API Gateway, and inter-service communication**
- **Overview of Advanced Patterns: Brief introduction to CQRS, Event Sourcing, and Composite UI.**

Chapter 3: Data Management & Communication in Microservices

Objective: Students will explore strategies for managing data and communication across distributed services, including the distinctions between synchronous and asynchronous methods and techniques for ensuring data consistency.

Topics Covered:

- **Data Management Strategies: Database per service, eventual consistency, and distributed transactions**
- **Synchronous Communication: REST/HTTP and gRPC**
- **Asynchronous Communication: Message brokers, event-driven design, and saga patterns**
- **Integration Patterns: Event sourcing and change data capture.**

Module 2: Building Microservices with ASP.NET Core Web API

Chapter 4: Developing Microservices with ASP.NET Core Web API

Objective: Students will gain practical experience building a microservice using ASP.NET Core Web API from setting up the solution and project structure to implementing CRUD operations and documenting APIs.

Topics Covered:

- **Solution & Project Layout**
- **Implementing CRUD Operations: Domain models, controllers, and repository pattern**
- **Data Context & Seeding Techniques**
- **API Documentation: Using Swagger/Swashbuckle**

Chapter 5: API Gateways for Microservices

Objective: Students will understand the role of API gateways in a microservices ecosystem, focusing on routing, security, and centralized access.
Topics Covered:

- **Role and Benefits of API Gateways**
- **Implementation Options: Using Ocelot and an introduction to YARP (Yet Another Reverse Proxy)**
- **Securing Incoming Requests and Service-to-Service Communication**

Chapter 6: Building a User Interface for Microservices

Objective: Students will learn how to create a front-end application that consumes multiple microservices, integrating back-end services into a cohesive user experience.
Topics Covered:

- **ASP.NET Core MVC/Razor Pages: Project setup and structure**
- **Developing Models & Service Classes**
- **Building Controllers & Views**
- **Integrating Multiple Microservices in the UI**

Module 3: Containerization, Orchestration & Cloud Deployment

Chapter 7: Introduction to Containers and Docker

Objective: Students will understand the fundamentals of containerization, the differences between containers and virtual machines (VMs), and the benefits of Docker in modern development and production environments.
Topics Covered:

- **Containers vs. VMs: Key differences and advantages**
- **Docker Fundamentals: Architecture, images, and containers**
- **Benefits in Development & Production**

Chapter 8: Building and Publishing Docker Images for .NET Core

Objective: Students will gain hands-on experience in creating optimized Docker images for .NET Core applications using best practices such as multi-stage builds.
Topics Covered:

- **Writing Dockerfiles: Best practices and multi-stage builds**
- **Building & Inspecting Images: Understanding image layers and optimization techniques**
- **Hosting ASP.NET Core Applications in Containers**

Chapter 9: Docker Compose and Multi Container Applications

Objective: Students will learn to orchestrate multi-container environments using Docker Compose, focusing on service networking, environment variables, and configuration management.
Topics Covered:

- **Docker Compose Basics: Understanding the structure of docker-compose.yml**
- **Networking Between Containers**
- **Managing Environment Variables & Configuration Files**
- **Common Docker Compose Commands**

Chapter 10: Hosting Microservices with Docker & Cloud Platforms

Objective: Students will explore strategies for containerizing an entire microservices solution including databases and caches and deploying it on cloud platforms (e.g., Azure), with an introduction to Kubernetes concepts.
Topics Covered:

- **Containerizing the Entire Solution: Creating Dockerfiles for all services**

- **Multi-Container System Design: Integrating services, databases, and caching solutions**
- **Environment Configuration: Managing connection strings and secrets**
- **Deployment Options: Azure App Services and an introduction to Kubernetes with Azure Kubernetes Service (AKS)**

Module 4: Data, Messaging & Real-Time Communication

Chapter 11: Caching Strategies with Redis

Objective: Students will understand the importance of caching in distributed systems and learn how to implement Redis to reduce latency and improve performance.

Topics Covered:

- **Caching Fundamentals in Microservices**
- **Redis Architecture & Use Cases**
- **Implementing Redis in ASP.NET Core**
- **Cache Invalidation Strategies**

Chapter 12: Messaging and Event Driven Microservices

Objective: Students will explore asynchronous communication patterns and real-time communication technologies essential for decoupled microservices.

Topics Covered:

- **Messaging Fundamentals: Synchronous vs. asynchronous communication**
- **RabbitMQ: Architecture, producing/consuming messages, and error handling**
- **Kafka: Overview, message production/consumption, and event streaming use cases**
- **WebSockets: Implementing real-time communication (e.g., chat, notifications)**

Chapter 13: Advanced Messaging with MassTransit

Objective: Students will understand advanced messaging capabilities using MassTransit to simplify and standardize message-based communication between microservices.

Topics Covered:

- **Introduction to MassTransit and Its Benefits**
- **Setting Up and Configuring MassTransit**
- **Advanced Messaging Patterns and Monitoring**

Module 5: Security & Advanced Architectural Patterns

Chapter 14: Security and Identity in Microservices

Objective: Students will learn the fundamentals of securing microservices by implementing authentication and authorization mechanisms and ensuring secure service-to-service communication.

Topics Covered:

- **Authentication & Authorization Fundamentals**
- **Securing Inter-Service Communication**
- **Introduction to Identity Microservices and Identity Server**

Chapter 15: Advanced Authentication with OAuth2, OpenID Connect & JWT

Objective: Students will master modern authentication protocols to secure microservices, including detailed workflows and token-based authentication practices.

Topics Covered:

- **OAuth2 & OpenID Connect: Protocols and workflows**
- **Implementing JWT-Based Authentication**
- **Best Practices for Securing APIs**

Chapter 16: Advanced Patterns – CQRS and Event Sourcing

Objective: Students will understand and implement the CQRS pattern along with event sourcing to separate command and query responsibilities, thereby enhancing scalability and maintainability.

Topics Covered:

- **Fundamentals of CQRS: Concepts and benefits over traditional CRUD**
- **Implementing Command-Query Separation**

- **Event Sourcing: Principles, trade-offs, and best practices**

Chapter 17: Clean Architecture & Domain-Driven Design

Objective: Students will learn how to structure microservices for high maintainability and testability by applying Clean Architecture principles alongside Domain-Driven Design.

Topics Covered:

- **Overview of Clean Architecture: Separation of concerns and layering**
- **Implementing Boundaries in ASP.NET Core**
- **Integrating Domain-Driven Design Principles**
- **Benefits, Common Pitfalls, and Best Practices**

Chapter 18: SteelToe in Microservices

Objective: Students will explore Steeltoe and its pivotal role in building microservices within .NET applications by understanding its core concepts, benefits, and the rationale behind its adoption, setting up the necessary prerequisites, and installing essential Steeltoe packages in a .NET Core Web API project.

Topics Covered:

- **What is Steeltoe?**
- **Why use Steeltoe in .NET applications?**
- **Benefits of using Steeltoe in Microservices**
- **Prerequisites and Setup**
- **Installing Steeltoe packages in a .NET Core Web API project**
- **Creating a simple microservice using Steeltoe**
- **Understanding Program.cs and dependency injection**
- **Running a basic Steeltoe application**
- **Centralized configuration in microservices**
- **Using Spring Cloud Config Server with Steeltoe**
- **Fetching configuration dynamically**
- **Environment-based configuration handling**

Module 6: Resilience, Observability & Performance

Chapter 19: Resilience and Failure Handling

Objective: Students will be equipped with strategies to handle failures in distributed systems by implementing retries, circuit breakers, and exploring chaos engineering principles.

Topics Covered:

- **Strategies for Handling Partial Failures and Graceful Degradation**
- **Implementing Retries & Circuit Breakers (using Polly)**
- **Introduction to Chaos Engineering**

Chapter 20: Observability and Monitoring

Objective: Students will learn to build observable microservices by implementing centralized logging, distributed tracing, and metrics collection to maintain high availability and rapid issue resolution.

Topics Covered:

- **Observability Fundamentals: Logging, tracing, and metrics**
- **Centralized Logging Tools (e.g., Serilog, ELK Stack)**
- **Distributed Tracing: Using OpenTelemetry or Jaeger**
- **Metrics & Dashboards: Prometheus and Grafana**
- **Alerting & Incident Response Strategies**

Chapter 21: Performance Optimization

Objective: Students will learn techniques for identifying and mitigating performance bottlenecks through load testing, profiling, and various optimization strategies for both applications and databases.

Topics Covered:

- **Performance Testing Tools: Apache JMeter, k6, etc.**
- **Application Profiling & Tuning: Identifying bottlenecks in ASP.NET Core**

- **Advanced Caching Strategies**
- **Database Optimization: Indexing, query tuning, and best practices**

Chapter 22: CI/CD Pipeline with Azure DevOps

Objective: Students will automate the build, test, and deployment processes for a complete microservices solution using Azure DevOps (or GitHub Actions), streamlining continuous integration and delivery.

Topics Covered:

- **CI/CD Fundamentals: Benefits of automation in microservices**
- **Azure DevOps Overview: Repos, Pipelines, and Artifacts**
- **Pipeline Setup: Automating builds, tests, and deployments**
- **Integrating Source Control (Azure Repos or GitHub)**
- **Continuous Deployment Strategies for Azure App Services or AKS**
- **Integrating Monitoring into CI/CD Workflows**

Chapter 23: Advanced Deployment Strategies

Objective: Students will explore cutting-edge deployment techniques including container orchestration, service meshes, and serverless microservices to further enhance scalability and operational management.

Topics Covered:

- **Kubernetes Architecture & Deployments (e.g., Azure Kubernetes Service)**
- **Introduction to Service Mesh Technologies (e.g., Istio, Linkerd)**
- **Serverless Microservices: Using Azure Functions or AWS Lambda**
- **GraphQL Integration for Efficient Data Retrieval**

Note: If we missed any topics, if any new features are introduced, or if you want to learn any concepts not in this Microservices using ASP.NET Core Web API Syllabus, please let us know, and that will also be included in this course. If you have any questions, please contact us.