**C# 12 ASP.CORE 8**

**Total Duration: 4 Days**

**Prerequisites:**

**Proficiency in C# (at least up to C# 9).**

**Solid understanding of .NET fundamentals and object-oriented programming principles.**

**Familiarity with basic asynchronous programming concepts.**

**Experience with LINQ and basic multithreading.**

**Course Outcomes:**

**By the end of this course, participants will be able to:**

1. **Understand and apply advanced features of C# 12.**
2. **Utilize generics, delegates, and events effectively in applications.**
3. **Implement and debug asynchronous patterns in .NET.**
4. **Use PLINQ to perform parallel data processing.**
5. **Apply parallel programming techniques for improved performance.**
6. **Understand and diagnose concurrency issues and improve thread safety.**

**Lab Setup:**

1. **Software Requirements:**
   * **IDE: Install Visual Studio 2022 or Visual Studio Code with .NET Core 8 SDK.**
   * **Database: Optional, depending on your case study approach (SQL Server, SQLite, etc.).**
   * **Testing Tools: Install Postman or Swagger for API testing.**
   * **Version Control: Set up Git for source code management.**
2. **Creating a Project:**
   * **Start a new .NET console or ASP.NET Core project:**

**bash**

**dotnet new console -n AdvancedCSharp12**

1. **Dependencies:**
   * **Install necessary NuGet packages for:**
     + **Entity Framework Core (if you plan to interact with a database).**
     + **NLog or Serilog for logging.**

**Day 1**

**Identifying Business Case and Segmentation**

**C#.Net Advanced**

1.Generics (Hands on for all topics relevant to identified use case except why Generics)

* Generic Types
* Why Generics Exist
* Generic Methods
* Declaring Type Parameters
* typeof and Unbound Generic Types
* The default Generic Value
* Generic Constraints
* Subclassing Generic Types
* Self-Referencing Generic Declarations
* Static Data
* Type Parameters and Conversions
* Covariance
* Contravariance
* Tuples

2.Delegates (Hands on for all topics relevant to identified use case)

* Writing Plug-In Methods with Delegates
* Instance and Static Method Targets
* Multicast Delegates
* Generic Delegate Types
* The Func and Action Delegates
* Delegates Versus Interfaces
* Delegate Compatibility

3. Events (Hands on for all topics relevant to identified use case publisher and subscriber with Kafka)

* Standard Event Pattern
* Event Accessors
* Event Modifiers

Day 2

4. Lambda Expressions (Hands on for all topics relevant to identified use case CRUD with SQL Server)

* Explicitly Specifying Lambda Parameter and Return Types
* Capturing Outer Variables
* Lambda Expressions Versus Local Methods
* Anonymous Methods
* Extension Method

**LINQ Queries** (Hands on for all topics relevant to identified use case CRUD with SQL Server)

* Fluent Syntax
  + Chaining Query Operators
  + Composing Lambda Expressions
  + Natural Ordering
  + Other Operators
* 2. Query Expressions
  + Range Variables
  + Query Syntax Versus SQL Syntax
  + Query Syntax Versus Fluent Syntax
  + Mixed-Syntax Queries
* 3. Deferred Execution
  + Reevaluation
  + Captured Variables
  + How Deferred Execution Works
  + Chaining Decorators
  + How Queries Are Executed
* 4. Subqueries
  + Subqueries and Deferred Execution
* 5. Composition Strategies
  + Progressive Query Building
  + The into Keyword
  + Wrapping Queries
* 6. Projection Strategies
  + Object Initializers
  + Anonymous Types
  + The let Keyword
* 7. Interpreted Queries
  + How Interpreted Queries Work
  + Combining Interpreted and Local Queries
  + AsEnumerable

**Day 3**

**Concurrency and Asynchrony** (Hands on for all topics relevant to identified use case Thread, Multithread, Locks, Sleep and Join, Thread Pool, Synchronization, Producer and Consumer with RabbitMQ, Thread Pool and Tasks)

* Introduction
* Threading
  + Creating a Thread
  + Join and Sleep
  + Blocking
  + Local Versus Shared State
  + Locking and Thread Safety
  + Passing Data to a Thread
  + Exception Handling
  + Foreground Versus Background Threads
  + Thread Priority
  + Signaling
  + Threading in Rich Client Applications
  + Synchronization Contexts
  + The Thread Pool
  + Atomic Methods Thread Safety and Race Conditions in C#
* Tasks
  + Starting a Task
  + Returning values
  + Exceptions
  + Continuations
  + TaskCompletionSource
  + Task.Delay
  + Principles of Asynchrony
  + Synchronous Versus Asynchronous Operations
  + What Is Asynchronous Programming?
  + Asynchronous Programming and Continuations
  + Why Language Support Is Important
* Asynchronous Programming in C#
  + Awaiting
  + Writing Asynchronous Functions
  + Asynchronous Lambda Expressions
  + Asynchronous Streams
  + Asynchronous Methods in WinRT
  + Asynchrony and Synchronization Contexts
  + Task-based asynchronous pattern (TAP).
  + Exception handling in async methods.
  + Optimizations

Day 4

* Async Debugging Techniques
  + Techniques for debugging asynchronous code.
  + Understanding the call stack in async methods.
  + Common pitfalls: deadlocks and race conditions.
  + Tools in Visual Studio for async debugging.
  + Hands-On Exercise: Debug a sample application with intentional async bugs and resolve them.

**Parallel Programming** (Hands on for all topics relevant to identified use case with Large Files)

* PLINQ
  + Parallel Execution Ballistics
  + PLINQ and Ordering
  + LINQ Limitations
  + Example: Parallel Spellchecker
  + Functional Purity
  + Setting the Degree of Parallelism
  + Cancellation
  + Optimizing PLINQ
* The Parallel Class
  + Parallel.Invoke
  + Parallel.For and Parallel.ForEach
* Task Parallelism
  + Creating and Starting Tasks
  + Waiting on Multiple Tasks
  + Canceling Tasks
  + Continuations
  + Task Schedulers
  + TaskFactory

**Capstone Project and Case Study**

**Integrating Concepts (4 hours)**

* Review and integration of Generics, Delegates, Events, Async Programming, and PLINQ.
* Discuss real-world applications and considerations.
* **Hands-On Exercise:** Plan a small project or application that utilizes multiple concepts learned.

**Session 8: Suggested Case Study Development (4 hours)**

* **Case Study:** Develop a **Multi-User Task Management API**.
  + **Requirements:**
    - User authentication (JWT).
    - Task Create, Read, Update, Delete (CRUD) operations.
    - Implement asynchronous data access with Entity Framework Core.
    - Use PLINQ for analytics/reporting features.
    - Register events for critical updates (e.g., task status changes).
    - Utilize generics for handling diverse data types.
* Teams will work on the project throughout the session, culminating in demonstrations or presentations.

**Suggested Case Study: Multi-User Task Management API**

* **Objective:** Build a robust web API for managing tasks across multiple users.
* **Key Features:**
  + User registration and authentication.
  + Task management (CRUD operations).
  + Implementation of asynchronous patterns for data-fetching and updates.
  + Use of PLINQ to allow users to filter, sort, and analyze tasks.
  + Events for notifying users about task updates.
  + Implement generics to allow diverse task types and interfaces.