.Net Core Framework 8 with C# 10.0

Pre-requisites

- Participants should be well versed in any Object Oriented Programming language
- Good if they have idea of C# beforehand
- Participants should be aware of database management system

Objective:

Participants will be able to understand

- .NET Core framework
- Different applications that can be created
- CLI tool
- Difference between .NET Framework and .NET Core and realize the future of .NET which is .NET 8
- C# language and create applications
- Access data from application using FE Core
- Test application using xUnit and many other testing tools

Software and Hardware:

- Please find the details in a separate document provided for the same

Duration:

- 5 full days

Course Content:

Day-1:

.NET Core

Module-1: Introduction to .NET Core

- What is .NET Core?
- .NET Core vs. .NET Framework
- How to create an application from scratch
    - Note: Console application will be created as sample
- Modularity in .NET Core
- NuGet Packages and Metapackages
- NO Base Class Library – welcome to CoreFx
- Deep-dive:
    - Understand the details of application and its execution
    - Different files that are required for running

    - Dependency configuration file, Runtime configuration file etc.
    - Using external packages in your application
    - Configure additional probing path
    - Understand execution in detail
    - Where is my Cross-platform advantage?
    - JIT compilation technique
    - Core CLR
    - Roslyn compiler
    - .NET CLI
    - dotnet.exe and it's commands
- .NET Standard: is it a framework?
- Understanding .NET Native
- Hosting .NET Core application: Really Host is required for non-web applications too?
- .NET Core Versions: 1.0, 2.0, 2.2, 3.0,3.1, 5.0, 6.0, 7.0 and 8.0
- Future: .NET 8

Module-2: Installation and Tools

- Installation of .NET Core framework package
- .NET Core Command-Line Interface (CLI)
- Visual Studio and Visual Studio Code
- New templates of VS 2022

Module-3: Applications in .NET Core

- Creating a Console Application
- Cross-Compiling
- Deployment in .NET Core
- UWP and .NET Core
- Portable Class Library (PCL)
- .NET Standard Use Case
- .NET Portability Analyzer

C# .NET Core (C# 10):

Module 01: C# Language Fundamentals

- Structure of a C# Program
- Basic Input/Output Operations
- Commenting a Program
- Recommended Practices

Module 02: Using Value-Type Variables in C#

- Naming Variables
- Best Practices for Naming Conventions
- Using Built-In Data Types
- Creating User-Defined Data Types
- Converting Data Types

- Typecasting
- Boxing and Un-boxing data types
- Nullable types

Module 03: C# Statements

- Introduction to Statements
- Using Selection Statements
- Using Iteration Statements
- Using Jump Statements
- Using Conditional Statements
- Applications Based on All Statements

Module 04: String and Arrays in C#

- String Handling
  - The String and String Builder Class
  - Different Methods and Properties of String and String Builder Class
- Arrays
  - Overview of Arrays
  - Creating and using Single Dimension and Multi Dimension Arrays
  - Jagged Arrays
  - Using foreach Loop, Param Keyword

Module 05: Methods and Parameters using C#

- Using Methods
- Using Parameters
  - Passing value type parameters
  - Passing Reference types(string, Array, object) as parameters
- Passing Parameters using Ref and Out keyword
- Static and Instance Members
- Explaining Constant and ReadOnly

Day-2:

Module 06: OOP in C#

- Defining Classes
- Instantiating and Working with Objects
- Difference between Abstraction and Encapsulation
- Understanding and Implementing Encapsulation
- Defining Object-Oriented Systems
- Understanding Namespaces
- Understanding Scope Resolution

Module 07: Creating Objects in C#

- Using Constructors
- Using Initializer Lists
- Initializing Data

Module 08: Properties and Indexers in C#

- Data Fields
- Properties
- Using Indexers
- Compare and Contrast between Properties and Indexers

Module 09: Inheritance in C#

- Deriving classes
- Understanding Type Hierarchy
- Hiding Base Class Member in Derived Class
- Implementing Multi Level Inheritance
- Using base keyword
- Using Static Classes

Module 10: Access Modifiers and Constructor

- Access Modifiers in C#
- Default Accessibility Level for Class, Methods and Structures
- Constructor Execution Sequence in Inheritance Scenario
- Default Constructor and Parameterized Constructor
- Constructor in Structure in C#
- Calling Base Class Constructor in Derived Class Constructor
- Using this keyword to Call One Constructor by another Constructor
- Discussing Public, Private and Protected Constructor
- Understanding Static Constructor
- Differentiating Static Constructor and Instance Constructor by call Mechanism
- Implementing Singleton Design Pattern and Understanding Static Class
- Partial types

Module 11: Polymorphism in C# and interface

- Polymorphism Using Methods
  - Overloading a Method
  - Overriding Virtual Method
- Abstract Class and Abstract Method
- Interface Implementation
- Interface Inheritance and Implementation

- Using Sealed Class and Sealed Method
- Discussed Inheritance and Interface Implementation in Structure
- Discussion to Differentiate Virtual Method, Abstract Method and Interface

Day-3:

Module 12: Exception Handling

- Checked and Unchecked Statements
- Try, Catch and Finally
- Do's and Don'ts of Exception Handling

Module 13: Collection Classes in C#

- Understanding Collection
- Using Different Collections viz. ArrayList, Stack, Queue, SortedList
- Understanding and Implementing Different Interface viz. IEnumerable, IEnumerator, IComparable, IComparer, IList
- Hashing Mechanism
- Generic Collection Classes such as List<T>, Stack<T>, Queue<T>, Dictionary<TK,TV>, SortedList<TK,TV>, HashSet<T> and interfaces such as IList<T>, IEnumerable<T>, IComparable<T>, IComparer<T> etc.
- Performance Improvement using Generic Collection over Non Generic version
- Generic class, generic methods, generic interfaces

Module 14: Operators and Equality Comparison

- Introduction to operators
- Operator overloading
- Equality Comparison Operators and Methods
  - Comparing Value Equality
  - Comparing Reference Equality
  - Using ==, Equals, ReferenceEquals, CompareTo
  - Comparison by GetHashCode Method
  - Overriding Methods and Operators for Equality Comparison
  - Overriding ToString Method

Module 15: Delegates and Events in C#

- Creating and using Delegates
  - Multicast Delegates
  - Anonymous Method
- When to Use Delegates, Events and Interfaces
- Generic Delegates

Module 16: Working with Files and Streams and Serialization

- Managing the filesystem
- Handling cross-platform environments and filesystems
- Managing drives, directories, files and paths
- Getting file information
- Controlling how you work with files
- Reading and writing with streams
- Writing to text streams
- Writing to XML streams
- Disposing of file resources
- Compressing streams
- Asynchronous streams
- Serialization Scenarios
- Serialization Attributes
- Object Graph
- Serialization Process
- Serialization Example
- Deserialization Example

Module 17: Language Enhancements

- Implicitly typed local variables
- Anonymous Types
- Extension Methods
- Object and Collection Initializer
- Lambda Expressions
- Query Expressions
- Named and Optional Parameters
- Co and Contra variance
- Dynamic Typing and Late Binding
- Parallel Programming TPL (asynchronous and Task<T>)
- Asynchronous members
- Caller info attributes
- Static imports
- Exception filters
- Property initializers
- Expression bodied members
- Null propagator
- String interpolation
- nameof operator
- Dictionary initializer

- out variables
- Tuples
- Discards
- Pattern Matching
- Local functions
- Generalized async return types
- Readonly members
- Default interface methods
- Pattern matching enhancements:
- Switch expressions
- Property patterns
- Tuple patterns
- Positional patterns
- Using declarations
- Static local functions
- Disposable ref structs
- Nullable reference types
- Asynchronous streams
- Asynchronous disposable
- Indices and ranges
- Null-coalescing assignment
- Unmanaged constructed types
- Stackalloc in nested expressions
- Enhancement of interpolated verbatim strings

Day-4:

Module 18: Threading and Asynchronous Programming in C#

- Threading & Synchronization
- Life cycle of a thread
- Different Thread Methods and Properties
- Synchronizing critical data using Synchronization objects
- Thread Pool
- How bad is it?
- Business Scenario and Problem Statement
- Solution to the Synchronous Problem
- Asynchronous Patterns
- Asynchronous Programming Model Pattern
- Event Based Asynchronous Pattern
- Task based Asynchronous Pattern
- C# 5.0 async and await based Asynchronous Pattern
- Problem with older Asynchrnous Patterns
- Business Scenario

- Task Parallel Library in C#
- What is Parallel Programming?
- Data Parallelism
- Task Parallelism

Part-3: EF Core and LINQ:

Module-1: Querying and Manipulating Data Using LINQ

- Writing LINQ queries
- Working with sets using LINQ
- Using LINQ with EF Core
- Sweetening LINQ syntax with syntactic sugar
- Using multiple threads with parallel LINQ
- Creating your own LINQ extension methods
- Working with LINQ to XML

Module-2: Working with Databases Using Entity Framework Core

- Understanding modern databases
- Setting up EF Core
- Defining EF Core models
- Querying EF Core models
- Loading patterns with EF Core
- Manipulating data with EF Core

Part-4: Testing

- Introduction
- Unit Testing
- xUnit
- MS Test

Day-5:

Part-5: .NET Core Web API

- Introduction to Web Service with Demo
- Introduction to WCF Service with Demo
- Introduction to Web API
- Difference between Web Service, WCF Service and Web API
- Web API features
- HTTP Web Services
- Web API Introduction
- Middleware
- Web API Routing
- Configuring WebApi

- Web API Parameter Biding
- Action Return Type
- WebApi Filters
- Content Negotiation
- Create CRUD WebApi
- Consume WebApi
- Working with swagger
- Postman Utility

Part-6: Open AI and .NET Core

- What is Open AI?
- Capabilities
- The technology used to build Chat GPT
- Consume ChatGPT Open AI API Inside .Net Core