# Program Structure

**General:**

**Live Teaching -**

- Live **Teaching** sessions from Monday – Saturday from 08:45 AM to 05:45 PM.

**Graded Assessments**

- For the **Milestone graded assessment (quiz and project)**, the submission timeline will be
  - Graded Quiz- 45 minutes
  - Graded Project- 1 hour 20 minutes

| Courses | Day no | Date | Modules | SubTopics |
|---|---|---|---|---|
| Introduction to .Net & C# | Day 1 | 27-01-2025 | Introduction & Basic Programming Concepts | |
| | | | Getting started with .NET platform | Installation of Visual Studio 2019 |
| | | | | Introduction to .NET framework |
| | | | | .NET framework architecture |
| | | | | Common type system |
| | | | | Common language specification |
| | | | | Common language runtime |
| | | | | Base class library |
| | | | Types in C# | Introduction to Classes |
| | | | | Reading and Writing in the console |
| | | | | Assembly |
| | | | | Private |
| | | | | Shared |
| | | | | Types |
| | | | | Value type and Reference type |
| | | | | Boxing and Unboxing |
| | | | | Strings |
| | | | | String Builder |
| | | | | Difference between String and StringBuilder |
| | | | | Nullable types |
| | Day 2 | 28-01-2025 | Core C# Programming | |
| | | | Arrays in C# | Array |
| | | | | Single array |
| | | | | Multi-dimension |
| | | | | Jagged array |
| | | | Looping Construct | Loops and Statement |
| | | | | IF statement |
| | | | | For loop |
| | | | | Foreach loop |
| | | | | Switch statement |
| | | | | While loop |
| | | | | Do-while loop |
| | Day 3 | 29-01-2025 | Getting Started with properties and functions | |
| | | | Access modifiers in C# | Access modifiers |
| | | | | Private |
| | | | | Public |
| | | | | Protected |
| | | | | Internal |
| | | | | Protected internal |
| | | | Methods in C# | Methods |
| | | | | Void method |
| | | | | With parameter |
| | | | | Return type method |
| | | | | Static method |
| | | | | Instance method |
| | | | | Namespaces |
| | | | | Unit testing |
| OOPS using C# | Day 4 | 30-01-2025 | Getting started with Inheritance for code reusability | |
| | | | Inheritance in C#.NET | Object Oriented Concept |
| | | | | Inheritance |
| | | | | Single |
| | | | | Multilevel |
| | | | | Multiple (using Interface) |
| | | | | Hierarchical |
| | | | | Hybrid |
| | | | OOPs principles ie Polymorphism in Action | |
| | | | Polymorphism in C#.NET | Polymorphism |
| | | | | Method override |
| | | | | Method overload |
| | | | | Encapsulation |
| | | | | Abstraction |
| | | | Object types in C#.NET | Properties |
| | | | | Static |
| | | | | Static class |
| | | | | Static method |
| | | | | Static constructor |
| | | | User define types in C#.NET | Structs |
| | | | | Interface |
| | | | | Abstract |
| | | | | Enumeration |
| | | | | Difference between Interface and Abstract |
| | | | | Problem with multiple class inheritance |
| | | | | Solution for multiple class inheritance using interface. |
| | | | | Exception handling |
| Data Handling using C# | Day 5 | 31-01-2025 | File Handling in C#.NET | |
| | | | File Handling in C#.NET | File I/O |
| | | | | Create |
| | | | | Write |
| | | | | Append |
| | | | | Delete |
| | | | | Copy |
| | | | Collections in C#.NET: Generic and Non Generic | Collections |
| | | | | Non generic |
| | | | | Array list |
| | | | | Hash table |
| | | | | Sorted list |
| | | | | Stack |
| | | | | Queue |
| | | | | Generic |
| | | | | List |
| | | | | Dictionary |
| | | | | Sorted list |
| | | | | Stack |
| | | | | Queue |
| | | | Advance Topics in C# | |

| | | | | |
|---|---|---|---|---|
| Advsnced C# | Day - 6 | 01-02-2025 | Exception Handling in C# | Introduction to Exception Handling<br>Types of Exceptions in C#, The try, catch, finally Blocks<br>Common Exception Classes<br>Creating Custom Exceptions<br>Exception Filters (C# 6.0 and Later)<br>Exception Handling Best Practices |
| | | | Latest C# 7.0 features | Extension Methods |
| | | | | Tuples and Deconstruction, Pattern Matching, Local Functions,Out Variables, Ref Locals and Returns, Discards (_) & Expression-bodied Members |
| | | | Prominent feartures till C# 12 | Nullable Reference Types, Async Streams, Ranges and Indices, Switch Expressions, Default Interface Methods, Pattern Matching Enhancements, Using Declarations & Readonly Members, Required Members, Primary Constructors, Collection Expressions, Lambda Expression Improvements, Improved Interpolated ,Using Static Directives for Accessing Static MembersStrings, Generic Math Support etc |
| | | | Searching and Sorting Data Structure | Importance of searching in data structures |
| | | | | Types of searching algorithms (Linear vs Binary) |
| | | | | Types of sorting algorithms (Comparison-based vs Non-comparison-based) |
| | | | | Bubble Sort, Selection Sort , Insertion Sort |
| | | | | Efficiency comparison among Bubble, Selection, and Insertion Sort<br>Real-world scenarios for each sorting algorithm |
| | | | Indexers and Attributes in C# | Indexer |
| | | | | Difference between index and properties |
| | | | | Attributes |
| | Day-7 | 03-02-2025 | Delegates & Reflection | |
| | | | Delegates  in C#.NET | Delegates |
| | | | | Single cast |
| | | | | Multi cast |
| | | | Standard C# Features | Anonymous method |
| | | | | Lambda expression |
| | | | | Func |
| | | | | Action |
| | | | | Predicate |
| | | | | Events |
| | | | | INtroduction to Reflection and its practical use |
| | | | | Generics |
| | | | | Generic class |
| | | | | Generic field |
| | | | | Generic method |
| | | | | Advantage of generics |
| | | | | Threading |
| | | | | Ref and Out keyword |
| | | | | Async & Await |
| | Day 8 | 04-02-2025 | Unit Testing & Test-Driven Development **(using  Xunit, Moq, Fluent Assertions)** | |
| | | | Introduction to Unit Testing | What is unit testing? |
| | | | | Benefits of unit testing |
| | | | | Basic concepts and terminology |
| | | | | Testing Frameworks |
| | | | Introduction to Testing Frameworks | Introduction to popular testing frameworks (NUnit, MSTest, XTest) |
| | | | | Setting up a testing framework in a project |
| | | | | Understanding test projects and test classes |
| | | | Writing Test Cases | Anatomy of a unit test |
| | | | | Creating test methods |
| | | | | Organizing test classes and test suites |
| | | | Assertions and Test Data | Using assertions to validate test results |
| | | | | Common assertion methods (e.g., Assert.AreEqual, Assert.IsTrue) |
| | | | | Test data setup and teardown |
| | | | Testing Techniques , Execution & Reporting | Test-driven development (TDD),Testing different scenarios (positive, negative, edge cases) |
| | | | | Running tests using the testing framework, analyzing and understanding test report |
| | | | .NET Core Fundamentals | |
| | | | Introduction to .NET Core | DOT NET CORE |
| | | | | What is .NET Core |
| | | | | Benefits of .NET core |
| | | | | What is new in .NET Core |
| | | | | .NET Core vs .NET Framework |
| | | | | First .NET Core Application. |
| | | | | Building .NET Core Applications |
| | | | Middleware and Static Files | Understanding middleware in ASP.NET Core |
| | | | | Configuring and using middleware components |
| | | | | Serving static files (HTML, CSS, JavaScript) |
| | | | | Security considerations with static files |
| | | | Introduction to Razor Pages | Overview of Razor Pages architecture |
| | | | | Advantages of Razor Pages over traditional MVC |
| | | | | Creating and configuring Razor Pages in a project |
| | | | | Folder structure and naming conventions |
| | | | Razor Syntax and Page Model | Razor syntax basics and directives |
| | | | | Mixing HTML and server-side code |
| | | | | Understanding the PageModel class |
| | | | | Property binding and handling requests |
| Advanced C# & Introduction to .Net | Day 9 | 05-02-2025 | **.NET Security & Reliability** | |
| | | | .NET Security & Reliability | Understanding security in .NET applications |
| | | | | Common security practices (Authentication, Authorization, Encryption) |
| | | | | Secure coding (Best practices) |
| | | | | Using .NET libraries for encryption, secure communication, and secure storage |
| | | | | Building Reliable Applications |
| | | | | Designing for reliability |
| | | | | Exception handling strategies |
| | | | | Error Handling & Logging |
| | | | | Implementing error handling best practices |
| | | | | Logging errors and monitoring application health |
| | Day 10 | 06-02-2025 | SOLID Principles & Design Patterns | |
| | | | SOLID Principles | Introduction to SOLID Principles |
| | | | | Overview of SOLID principles (SRP, OCP, LSP, ISP, DIP) |
| | | | | Practical Application of SOLID |
| | | | | Refactoring code to adhere to SOLID principles |
| | | | Design Patterns | Introduction to Design Patterns |
| | | | | Understanding the importance of Design Patterns in software design |
| | | | | Overview of common design patterns (Creational, Structural, Behavioral) |
| | | | Introduction, Basic Concepts, and DML Commands | |
| | | | Introduction to Databases | Installation of SQL Server Management Studio |
| | | | | Introduction to basic database concepts. |

|  |  |  |  |  |
|---|---|---|---|---|
| SQl & microsoft SQl server | Day 11 | 07-02-2025 |  | Advantage of DBMS |
|  |  |  | Keys, Operators & DML Commands | Introduction to RDBMS |
|  |  |  |  | Creating Tables |
|  |  |  |  | Relationship between tables. |
|  |  |  |  | Primary keys, Foreign keys, Unique keys |
|  |  |  |  | SQL operators (Arithmetic, Comparison, Logical) |
|  |  |  |  | DML Commands |
|  |  |  |  | CRUD operations |
|  |  |  | DDL, DCL, and TCL Commands |  |
|  |  |  | SQL Fundamentals | SQL commands |
|  |  |  |  | Constraints |
|  |  |  |  | Primary key |
|  |  |  |  | Foreign key |
|  |  |  |  | Types of contraints - Not null, Check, Unique. |
|  |  |  |  | DDL Commands |
|  |  |  |  | DCL & TCL Commands |
|  |  |  |  | Grant & Revoke, Commit & Rollback |
|  |  |  | Advanced Queries, Functions, and Joins |  |
|  | Day 12 | 08-02-2025 | Function in SQL Server | Functions |
|  |  |  |  | Built-in Functions |
|  |  |  |  | Scalar functions (e.g., LEN, ROUND, GETDATE) |
|  |  |  |  | Aggregate functions (e.g., SUM, AVG, COUNT) |
|  |  |  |  | User-defined Functions |
|  |  |  | Working with Queries | Aggregate functions |
|  |  |  |  | Joins |
|  |  |  |  | Inner join |
|  |  |  |  | Left join |
|  |  |  |  | Right join |
|  |  |  |  | Self-join |
|  |  |  |  | Full outer join |
|  |  |  |  | Cross join |
|  |  |  |  | Set operators |
|  |  |  |  | Union |
|  |  |  |  | Intersect |
|  |  |  |  | Minus |
|  |  |  | Advanced Queries, Functions, and Joins |  |
|  |  |  | Normalizations | Normalizations |
|  |  |  |  | First normal form |
|  |  |  |  | Second normal form |
|  |  |  |  | Third normal form |
|  |  |  |  | Benefits of normalization and when to apply it |
| HTML CSS JS | Getting started with HTML /CSS /JavaScript |  |  |  |
|  | Day 13 | 10-02-2025 | HTML and CSS | Basic Structure of an HTML Page |
|  |  |  |  | Various Tags and Attributes |
|  |  |  |  | Inline and Block Elements |
|  |  |  |  | Forms |
|  |  |  |  | Semantic elements |
|  |  |  |  | HTML 5 Features |
|  |  |  |  | Paper based Layout |
|  |  |  |  | Hero unit |
|  |  |  |  | Audio Video Players |
|  |  |  |  | Semantic Markup |
|  |  |  |  | Geo Location, A11Y (Accessibility) |
|  |  |  |  | Intro to CSS |
|  |  |  |  | CSS Properties and their syntaxes |
|  |  |  |  | Various Selectors in CSS and their preference orders |
|  |  |  |  | Inline , Internal and External Styling |
|  |  |  |  | Box Model |
|  |  |  |  | CSS 3 |
|  |  |  |  | Advance Unites like rem, em and vh vw |
|  |  |  |  | New elements added like Round Corners |
|  |  |  |  | Working with flex box |
|  | Day 14 | 11-02-2025 | Bootstrap<br>Basics of JavaScript | Working with Bootstrap |
|  |  |  |  | Cards, Tables, and Lists |
|  |  |  |  | Develop a static webpage,Forms and Form Validations |
|  |  |  |  | Navs and Navbar,Pagination and Progress |
|  |  |  |  | Navigation and Pagination |
|  |  |  |  | Basics of JavaScript - Data type, variables, |
|  |  |  |  | Type Conversions, Basic Operators, |
|  |  |  |  | Branching and Looping Statements, Functions, Arrays and Strings |
|  |  |  | Advanced Java Script | Date and Time Objects, Closures, DOM Objects, ES6+ Features, Promises and Async Programming, Closures and Scoping, |
|  |  |  |  | Functional Programming, Error Handlingclient side scripting, |
|  |  |  |  | JSON, JSON - Objects, arrays etc, JSON-Ajax Advanced DOM Manipulation, |
|  |  |  |  | jQuery Plugins, Event Delegation, Ajax and Deferred Objects, Optimization and Performance |
| Type Script |  |  | SPA & Working with NPM |  |
|  | Day 15 | 12-02-2025 | Type Script | Var-hoisting<br>Functional Score<br>Classes & Methods & Constructor |
|  |  |  |  | Maps<br>Iterators<br>Interfaces<br>Declartions and Annotations |
|  |  |  |  | Anytype<br>Enumeration<br>Decorator , Arrays & tuples |
| NPM & Node JS | Day 16 | 13-02-2025 | Working with NPM<br>& Node JS | Introduction to NPM |
|  |  |  |  | Initializing a project with npm init. |
|  |  |  |  | Installing, updating, and removing packages: |
|  |  |  |  | npm install, npm update, npm uninstall. |
|  |  |  |  | Understanding package.json and package-lock.json. |
|  |  |  |  | reating a simple Node.js application, Installing, updating, and removing packages: |
|  |  |  |  | Node.js Core Modules : fs (File System),http/https (HTTP/HTTPS Servers and Clients)<br>"path (File and Directory Paths)<br>events (EventEmitter), Understanding package.json and package-lock.json." |
|  |  |  |  | "Asynchronous Programming in Node.js : Callback Functions , Promises , Async/Await , More Built in Modules :Events and Errors, What is Express.js?<br>Why use Express.js? Features and advantages<br>Getting started with Express.js"<br>"Creating a new Node.js project<br>Installing Express.js in your project<br>Basic Routing : Understanding routes in Express.js<br>Creating routes for different HTTP methods (GET, POST, PUT, DELETE),Handling dynamic routes and parameters, Using route middleware" |
|  |  |  | React Basics and Component Architecture |  |
|  |  |  | Understanding React as a Framework: | Introduction to React |
|  |  |  |  | Key features and advantages of React |

|  |  |  |  |  |
|---|---|---|---|---|
| React | Day 17 & 18 | 14-02-2025 | Understanding React as a Framework: | Use cases and scenarios where Angular or React is preferred |
|  |  |  | Component Lifecycle and Basic Building Blocks of SPA Framework: | Overview of component lifecycle in React |
|  |  |  |  | Understanding the role of components in SPA development |
|  |  |  |  | Basic building blocks such as components, modules, services, and routing in Angular and React applications |
|  |  |  | Components, States, and Props | Creating functional and class components |
|  |  | 15-02-2025 |  | Managing state within components |
|  |  |  |  | Passing data between components using props |
|  |  |  | Events and Passing Props | Handling events in React (e.g., onClick, onChange) |
|  |  |  |  | Passing functions as props to manage state in child components |
|  |  |  |  | Practical exercises: Building interactive components with event handling |
|  |  |  | React Router and More | Introduction to React Router for client-side navigation |
|  |  |  |  | Setting up basic routing in a React application |
|  |  |  |  | Practical exercises: Creating a multi-page React application |
| Advanced React | Day 19 & 20 | 17-02-2025 | Advanced React Concepts |  |
|  |  |  | Refactoring with Hooks and Context API | Introduction to React Hooks (useState, useEffect, useContext) |
|  |  |  |  | Refactoring class components to functional components with hooks |
|  |  |  |  | Using Context API for global state management |
|  |  |  |  | Practical exercises: Refactor an existing application using hooks |
|  |  | 18-02-2025 | Using React Hooks | Deep dive into advanced hooks (useReducer, useMemo, useCallback) |
|  |  |  |  | Custom hooks: When and how to create them |
|  |  |  |  | Practical exercises: Implement custom hooks in a React application |
|  |  |  | Building Server-side Rendered React Apps | Introduction to server-side rendering (SSR) with React |
|  |  |  |  | Setting up SSR with frameworks like Next.js |
|  |  |  |  | Benefits of SSR for performance and SEO |
|  |  |  |  | Practical exercises: Create a basic server-side rendered React application |
| Introduction to Microservices | Day 21 | 19-02-2025 | Introduction to REST & Microservice Architecture using .NET Core |  |
|  |  |  | Introduction to REST and its Architecture | Overview of RESTful services and principles |
|  |  |  |  | RESTful API design best practices in ASP.NET Core |
|  |  |  |  | Understanding statelessness and resource-based routing |
|  |  |  | Introduction to Microservices Architecture | Introduction to microservices architecture |
|  |  |  |  | Comparing microservices with monolithic architecture |
|  |  |  |  | Benefits and challenges of microservices |
|  | Day 22 | 20-02-2025 | **MileStone 1 --**<br>**C# ( Arrays  & Strings)** |  |