# .NET

## 1. Introduction to .NET Framework

- Understand what is .NET Framework
- Familiarize yourself with the components of the .NET Framework, such as Common Language Runtime (CLR), Base Class Library (BCL), and Application Domain
- Understand the architecture of the .NET Framework
- Understand the benefits of using .NET Framework

### Tasks

1. Understand what is .NET Framework:
   - ☐ Read an overview of .NET Framework from the Microsoft documentation
   - ☐ Watch a video explaining what is .NET Framework
   - ☐ Search for articles or blogs discussing .NET Framework and read a few to get an idea of what it is
2. Familiarize yourself with the components of the .NET Framework, such as Common Language Runtime (CLR), Base Class Library (BCL), and Application Domain:
   - ☐ Read about CLR and its role in .NET Framework
   - ☐ Familiarize yourself with the BCL and its most commonly used classes
   - ☐ Learn what Application Domain is and how it can be used in .NET applications
3. Understand the architecture of the .NET Framework:
   - ☐ Study the different layers of the .NET Framework architecture
   - ☐ Understand how the layers interact with each other
   - ☐ Search for diagrams or visual aids to help you better understand the architecture
4. Understand the benefits of using .NET Framework:
   - ☐ Read about the advantages of using .NET Framework for application development
   - ☐ Study the performance benefits of using .NET Framework
   - ☐ Look for examples of companies or products that use .NET Framework to understand its real-world applications

Completing these tasks should give you a good understanding of what .NET Framework is, its components, architecture, and the benefits of using it.

## 2 .NET Class Library

- Learn about the .NET Class Library and its importance in .NET development
- Familiarize yourself with commonly used namespaces, such as System, System.IO, System.Collections, and System.Threading
- Understand how to use classes and interfaces from the .NET Class Library in your application

**Tasks**

1. Learn about the .NET Class Library and its importance in .NET development:
   - ☐ Read an overview of the .NET Class Library from the Microsoft documentation
   - ☐ Watch a video explaining the importance of the .NET Class Library
   - ☐ Study the most commonly used classes in the .NET Class Library
2. Familiarize yourself with commonly used namespaces, such as System, System.IO, System.Collections, and System.Threading:
   - ☐ Read about the purpose and usage of the System namespace
   - ☐ Study the System.IO namespace and its commonly used classes
   - ☐ Understand the usage of the System.Collections namespace and its classes
   - ☐ Study the System.Threading namespace and its commonly used classes
3. Understand how to use classes and interfaces from the .NET Class Library in your application:
   - ☐ Learn how to add references to the .NET Class Library in your project
   - ☐ Study how to instantiate objects from the .NET Class Library
   - ☐ Understand how to use methods and properties from the .NET Class Library classes in your application
   - ☐ Practice creating a simple application that uses classes and interfaces from the .NET Class Library

Completing these tasks should give you a good understanding of the .NET Class Library, commonly used namespaces, and how to use classes and interfaces from the library in your application.

## 3. ASP.NET

- Get an introduction to ASP.NET, a popular web application framework for .NET
- Understand the architecture of ASP.NET
- Learn how to create web applications using ASP.NET and C#
- Understand the concept of server controls and data binding in ASP.NET

**Tasks**

1. Get an introduction to ASP.NET, a popular web application framework for .NET:
   - ☐ Read an overview of ASP.NET from the Microsoft documentation
   - ☐ Watch a video explaining what ASP.NET is and its benefits
   - ☐ Search for articles or blogs discussing ASP.NET and read a few to get an idea of what it is
2. Understand the architecture of ASP.NET:
   - ☐ Study the different components of the ASP.NET architecture, such as the HTTP runtime, HTTP modules, and the page framework
   - ☐ Understand the role of the ASP.NET worker process and application domains in handling requests
   - ☐ Search for diagrams or visual aids to help you better understand the ASP.NET architecture
3. Learn how to create web applications using ASP.NET and C#:
   - ☐ Study the basics of creating an ASP.NET web application, including setting up a project, creating a web form, and adding server controls
   - ☐ Learn how to create a simple web application that displays data from a database using ASP.NET and C#
   - ☐ Study how to handle user input and respond with dynamic content using ASP.NET and C#
4. Understand the concept of server controls and data binding in ASP.NET:
   - ☐ Learn what server controls are and how they are used in ASP.NET
   - ☐ Study the different types of server controls available in ASP.NET, such as text boxes, labels, and buttons
   - ☐ Learn how to use data binding to populate server controls with data from a database
   - ☐ Practice creating a simple application that uses server controls and data binding in ASP.NET

Completing these tasks should give you a good understanding of ASP.NET, its architecture, and how to create web applications using ASP.NET and C#. You should also have an understanding of server controls and data binding in ASP.NET.

## 4. Advanced Topics in .NET

- Learn about advanced topics such as LINQ, Entity Framework, and MVC
- Understand the benefits and usage of LINQ for querying data in .NET
- Learn about Entity Framework and its importance in .NET development
- Understand the Model-View-Controller (MVC) pattern and how to implement it in .NET applications

**Tasks**

Sure! Here are some tasks you could complete for Day 4:

1. Learn about advanced topics such as LINQ, Entity Framework, and MVC:
    - ☐ Read an overview of each topic from the Microsoft documentation
    - ☐ Watch videos that provide an introduction to each topic and explain their importance in .NET development
    - ☐ Search for articles or blogs discussing each topic and read a few to get an idea of what they are and how they are used
2. Understand the benefits and usage of LINQ for querying data in .NET:
    - ☐ Study the syntax of LINQ queries and how they are used to query data in .NET
    - ☐ Learn how to use LINQ to query databases, collections, and arrays
    - ☐ Practice writing LINQ queries for different types of data sources
3. Learn about Entity Framework and its importance in .NET development:
    - ☐ Study the basics of Entity Framework, including its architecture and how it is used to access databases in .NET applications
    - ☐ Learn how to use Entity Framework to create, read, update, and delete data in a database
    - ☐ Practice using Entity Framework in a simple application to get a better understanding of how it works
4. Understand the Model-View-Controller (MVC) pattern and how to implement it in .NET applications:
    - ☐ Study the basics of the MVC pattern and how it is used to organize code in .NET applications
    - ☐ Learn how to create an MVC application in .NET, including creating models, views, and controllers
    - ☐ Practice creating a simple application using the MVC pattern in .NET

Completing these tasks should give you a good understanding of advanced topics such as LINQ, Entity Framework, and MVC. You should also have an understanding of the benefits and usage of LINQ for querying data, Entity Framework and its importance in .NET development, and the Model-View-Controller (MVC) pattern and how to implement it in .NET applications.

## 5. .NET Core

- Learn about .NET Core, the cross-platform version of .NET
- Understand the differences between .NET Framework and .NET Core
- Learn how to create and run .NET Core applications
- Familiarize yourself with ASP.NET Core, the cross-platform version of ASP.NET

**Tasks**

1. Learn about .NET Core, the cross-platform version of .NET:
   - ☐ Study the history and purpose of .NET Core and why it was created
   - ☐ Learn about the features and benefits of .NET Core compared to .NET Framework
   - ☐ Read the official documentation to get a better understanding of .NET Core
2. Understand the differences between .NET Framework and .NET Core:
   - ☐ Study the major differences between .NET Framework and .NET Core, including their architecture and target platforms
   - ☐ Learn about the differences in performance, features, and tooling between .NET Framework and .NET Core
   - ☐ Study the migration path from .NET Framework to .NET Core, including the challenges and benefits of the transition
3. Learn how to create and run .NET Core applications:
   - ☐ Install the .NET Core SDK and learn how to use the command-line interface (CLI) to create and run .NET Core applications
   - ☐ Learn about the different types of .NET Core projects, including console applications, web applications, and class libraries
   - ☐ Practice creating and running .NET Core applications using the CLI and Visual Studio
4. Familiarize yourself with ASP.NET Core, the cross-platform version of ASP.NET:
   - ☐ Study the architecture of ASP.NET Core and how it differs from ASP.NET Framework
   - ☐ Learn how to create and run ASP.NET Core web applications, including creating controllers and views
   - ☐ Practice creating and running a simple ASP.NET Core web application to get a better understanding of how it works

Completing these tasks should give you a good understanding of .NET Core, the differences between .NET Framework and .NET Core, how to create and run .NET Core applications, and how to create and run ASP.NET Core web applications.

## Projects

**Console Application** Build a simple console application that uses the .NET Framework's Base Class Library (BCL). You could create a program that generates a random password, or converts temperatures from Celsius to Fahrenheit.

**Web Page** Build a simple web page using HTML, CSS, and JavaScript. You could create a todo application or a recipe management app.

**Windows Forms Application** Build a simple desktop application using Windows Forms. You could create a text editor or a timer.

**Mobile App** Build a simple mobile app using Xamarin. You could create a tip calculator app, or a weather app.

**API** Build a simple RESTful API using ASP.NET Web API. You could create an API for a to-do list, a weather service, or a simple quiz game.

Remember, the goal of these projects is to apply what you've learned and to have fun while doing it. Don't worry too much about making something complex or perfect – focus on building something useful and learning along the way.

## Resources

There are many free resources available to help you learn about .NET. Here are some of the best ones:

**Official documentation** The official .NET documentation is an excellent resource that covers all aspects of the .NET platform, including the .NET Framework, .NET Core, and ASP.NET. It's comprehensive, up-to-date, and easy to navigate.

**Microsoft Learn** is a free, interactive learning platform that provides hands-on learning experiences with various Microsoft products, including .NET. It offers guided learning paths, tutorials, and exercises to help you learn at your own pace.

**Pluralsight** is a popular online learning platform that offers many courses on .NET development. It offers a free trial, and some courses are available for free without a subscription.

**YouTube** There are many YouTube channels that offer tutorials and courses on .NET development. Some popular channels include Microsoft Visual Studio, Tim Corey, and IAmTimCorey.

**FreeCodeCamp** is a non-profit organization that offers free coding lessons and projects on various programming languages, including C# and .NET. It offers a comprehensive curriculum that covers everything from the basics to advanced topics.

**Books** There are many excellent books available on .NET development, some of which are available for free. Some popular titles include "C# Yellow Book" by Rob Miles, "C# Fundamentals for Absolute Beginners" by Bob Tabor, and ".NET Core Succinctly" by Giancarlo Lelli.

Using these resources in combination with the learning plan we've created should provide you with a well-rounded education on .NET development.