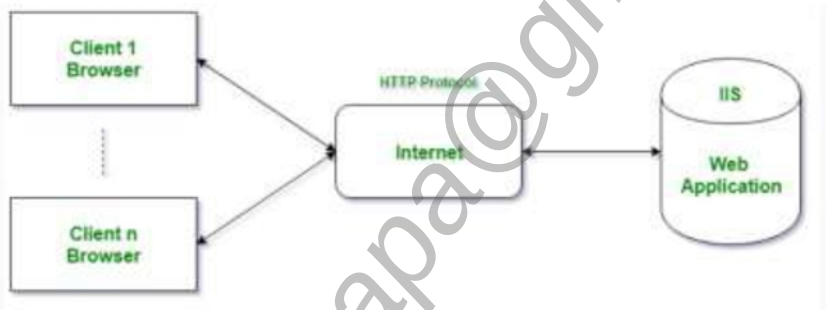
Unit 2
Introduction to ASP.net

Introduction to .NET

- .NET is a free, open-source development platform for building many kinds of apps, such as: Web application, Web services, Desktop application, mobile application, game application, console application etc.
- **.NET** is a software framework which is designed and developed by Microsoft.
- One on side, it is a **developer platform** made up of tools, programming languages, and libraries for creating and building many different types of applications by integrating different programming language. On other side, .NET is an essential component of the operating system, that **helps in executing applications** for **general User.**
- There is a variety of programming languages available on the .Net platform, VB.Net and C# being the most common ones.

ASP.NET

- ASP.NET is open source and a subset of the .NET Framework and successor of the classic ASP(**A**ctive **S**erver **P**ages).
- ASP.NET is a free web framework designed and developed by Microsoft. for building great websites and web applications using HTML, CSS, and JavaScript.
- A web application is an application installed only on the web server which is accessed by the users using a web browser like Microsoft Internet Explorer, Google Chrome, Mozilla FireFox, Apple Safari, etc.



- ASP.NET is a web development platform, which provides a programming model, a comprehensive software infrastructure and various services required to build up robust web applications for PC, as well as mobile devices.
- ASP.NET works on top of the HTTP protocol, and uses the HTTP commands and policies to set a browser-to-server bilateral communication and cooperation.
- When a browser requests an ASP or ASP.NET file, the ASP engine reads the file, executes any code in the file, and returns the result to the browser.
- The web applications which are developed using the .NET framework or its subsets required to execute under the **Microsoft Internet Information Services(IIS)** on the server side.
- ASP.NET applications are compiled codes, written using the extensible and reusable components or objects present in .Net framework.
- These codes can use the entire hierarchy of classes in .Net framework.
- However, the development of ASP.NET 5 was stopped in favor of ASP.NET Core.
- .NET Supports a lot of web development model/Framework
    1. Classic ASP:
        - It is the first server side scripting language developed by Microsoft.
        - ASP (i.e. Classic ASP) was introduced in 1998 as Microsoft's first server side scripting language.
        - Classic ASP pages have the file extension **.asp** and are normally written in VBScript.
        - HTML markup and your code together in the same file

    2. ASP.NET Web Page
        - ASP.NET Web Pages and the Razor syntax provide a fast, approachable, and lightweight way to combine server code with HTML to create dynamic web content.

- Connect to databases, add video, link to social networking sites, and include many more features that help you create beautiful sites that conform to the latest web standards.

3. ASP.NET Web Forms:
    - With ASP.NET Web Forms, you can build dynamic websites using a familiar drag-and-drop, event-driven model.
    - A design surface and hundreds of controls and components let you rapidly build sophisticated, powerful UI-driven sites with data access.
    - These are the event-driven application model which are not considered a part of the new ASP.NET Core.
    - These are used to provide the server-side events and controls to develop a web application.

Features of ASP.NET Web Forms

1. Server Controls: Web Forms provides rich set of server controls. These controls are objects that run when the page is requested and render markup to the browser. Some Web server controls are similar to familiar HTML elements, such as buttons and text boxes. It also provides controls that we can use to connect to data sources and display data.
2. Master Pages: It allows us to create a consistent layout for the pages in our application. This page defines the look and feel and standard behavior that we want for all of the pages in our application. When users request the content pages, they merge with the master page to produce output that combines the layout of the master page with the content from the content page.
3. Working with Data: In an ASP.NET Web Forms application, we use data-bound controls to automate the presentation or input of data in web page UI elements such as tables and text boxes and drop-down lists.
4. Client Script and Client Frameworks: We can enhance the server-based features of ASP.NET by including client-script functionality in ASP.NET Web Form pages. We can use client script to provide a richer, more responsive user interface to the users. We can also use client script to make asynchronous calls to the Web server while a page is running in the browser.
5. State Management: ASP.NET Web Forms includes several options that help you preserve data on both a per-page basis and an application-wide basis.
6. Security: Developing a secure application is most important aspect of software development process. ASP.NET Web Forms allow us to add extensibility points and configuration options that enable us to customize various security behaviors in the application.
7. Performance: Web Forms provides good performance and allows us to modify performance related to page and server control processing, state management, data access, application configuration and loading, and efficient coding practices.
8. Debugging and Error Handling: We can diagnose problems that occur in our Web Forms application. Debugging and error handling are well supported within ASP.NET Web Forms so that our applications compile and run effectively.

4. ASP.NET API:
    - It is the Web Application Programming Interface(API).
    - ASP.NET Web API is a framework that makes it easy to build HTTP services that reach a broad range of clients, including browsers and mobile devices.
    - ASP.NET Web API is an ideal platform for building RESTful applications on the .NET Framework.
    - The ASP.NET Web API is an extensible framework for building HTTP based services that can be accessed in different application on different platform such as web, windows, mobile etc.
    - It is more or less similar to that of ASP.NET MVC except that it sends data as response instead of HTML View.

5. ASP.NET MVC:
    - It is the Model-View-Controller application model which can be merged with the new ASP.NET Core.
    - It is used to build dynamic websites as it provides fast development.
    - ASP.NET MVC gives you a powerful, patterns-based way to build dynamic websites that enables a clean separation of concerns and that gives you full control over markup for enjoyable, agile development.

6. ASP.NET Core:

//Will be discussed in next chapter

.NET CLI: Build, Run, Test and Deploy .Net Core Application

- The .NET command-line interface (CLI) is a cross-platform toolchain for developing restoring, building, running, and publishing .NET applications.
- We created our first .NET application using Visual Studio in the previous chapter but Visual Studio internally uses this CLI to restore, build and publish an application.
- Other higher level IDEs, editors and tools can use CLI to support .NET Core applications.
- The .NET Core CLI is installed with .NET Core SDK for selected platforms. So we don't need to install it separately on the development machine.
- We can verify whether the CLI is installed properly by opening command prompt in Windows and writing dotnet and pressing Enter. If it displays usage and help as shown below then it means it is installed properly.
- **Some** of the basic CLI commands include
    1. new: Creates a new project, configuration file, or solution based on the specified template (Console, MVC etc.) Example:
        1.

```
C:\Users\Bheeshma\Desktop\MyFolder>dotnet new Console --name FirstConsoleApplication
The template "Console Application" was created successfully.

Processing post-creation actions...
Running 'dotnet restore' on FirstConsoleApplication\FirstConsoleApplication.csproj...
  Restoring packages for C:\Users\Bheeshma\Desktop\MyFolder\FirstConsoleApplication\FirstConsoleApplication.csproj...
  Generating MSBuild file C:\Users\Bheeshma\Desktop\MyFolder\FirstConsoleApplication\obj\FirstConsoleApplication.csproj.
nuget.g.props.
  Generating MSBuild file C:\Users\Bheeshma\Desktop\MyFolder\FirstConsoleApplication\obj\FirstConsoleApplication.csproj.
nuget.g.targets.
  Restore completed in 231.47 ms for C:\Users\Bheeshma\Desktop\MyFolder\FirstConsoleApplication\FirstConsoleApplication.
csproj.

Restore succeeded.
```

        This, will create new console application named FirstConsoleApplication in MyFolder directory in Desktop.
        2.

```
C:\Users\Bheeshma\Desktop\MyFolder>dotnet new mvc --name FirstMVCApplication
The template "ASP.NET Core Web App (Model-View-Controller)" was created successfully.
This template contains technologies from parties other than Microsoft, see https://aka.ms/aspnetcore-template-3pn-210 fo
r details.

Processing post-creation actions...
Running 'dotnet restore' on FirstMVCApplication\FirstMVCApplication.csproj...
  Restoring packages for C:\Users\Bheeshma\Desktop\MyFolder\FirstMVCApplication\FirstMVCApplication.csproj...
  Generating MSBuild file C:\Users\Bheeshma\Desktop\MyFolder\FirstMVCApplication\obj\FirstMVCApplication.csproj.nuget.g.
props.
  Generating MSBuild file C:\Users\Bheeshma\Desktop\MyFolder\FirstMVCApplication\obj\FirstMVCApplication.csproj.nuget.g.
targets.
  Restore completed in 1.5 sec for C:\Users\Bheeshma\Desktop\MyFolder\FirstMVCApplication\FirstMVCApplication.csproj.
```

        This, will create new mvc core application named FirstMVCApplication in MyFolder directory in Desktop.

    2. restore: restores the dependencies and tools of a project. The dotnet restore command uses NuGet(Package Manager) to restore dependencies as well as project-specific tools that are specified in the project file. In most

cases, you don't need to explicitly use the dotnet restore command, since a NuGet restore is run implicitly if necessary when we specify the commands like dotnet new, dotnet Build etc.

3. Build: Builds a project and all of its dependencies. The dotnet build command builds the project and its dependencies into a set of binaries. The binaries include the project's code in Intermediate Language (IL) files with a *.dll* extension. Depending on the project type and settings, other files may be included, such as: .exe and other package references like entity, newtonshoft.json in form of .dll etc.
   Example:

```
C:\Users\Bheeshma\Desktop\MyFolder>cd FirstConsoleApplication

C:\Users\Bheeshma\Desktop\MyFolder\FirstConsoleApplication>dotnet build
Microsoft (R) Build Engine version 15.9.20+g88f5fadfbe for .NET Core
Copyright (C) Microsoft Corporation. All rights reserved.

  Restore completed in 52.06 ms for C:\Users\Bheeshma\Desktop\MyFolder\FirstConsoleApplication\FirstConsoleApplication.c
sproj.
  FirstConsoleApplication -> C:\Users\Bheeshma\Desktop\MyFolder\FirstConsoleApplication\bin\Debug\netcoreapp2.1\FirstCon
soleApplication.dll

Build succeeded.
    0 Warning(s)
    0 Error(s)

Time Elapsed 00:00:02.99
```

4. Run: Runs the source code without any explicit compilation or launch command.

```
C:\Users\Bheeshma\Desktop\MyFolder\FirstConsoleApplication>dotnet run
Hello World!
```

5. Publish: Packs the applications and its dependencies into a folder for deployment to a hosting system.
6. Help: display the help on specified command
7. Add package: add package reference to a project
   Example:
   Adding Entity Framework version 2.0.0

   dotnet add package EntityFramework --version 6.4.4

   <Note: Observe the *.csproj file to verify the added dependencies>
8. Remove package: remove package reference from the project
   Example:

   dotnet remove package EntityFramework

   <Note: Observe the *.csproj file to verify the removed dependencies>
9. Version: Provides the current .net core version
   Example:

```
C:\Users\Bheeshma\Desktop\MyFolder\FirstConsoleApplication>dotnet --version
2.1.517
```

Assignment-2

- Write difference between .net framework and .net core.
  1. .NET Framework happened to be released as proprietary and licensed software framework. However, Microsoft released .NET Core as an open-source framework.
  2. Using the .NET framework, it allows developers to build apps for only the Windows platform but .NET Core is cross-platform compatible. It can be easily combined with a variety of operating systems, such as Linux, Windows and OS X.
  3. While installing the .NET framework for Windows, make sure to install it as a single package and runtime environment. When it comes to .NET Core, which is cross-platform compatible, it would need to be packaged and installed independently of any operating system. Here, the developers are responsible for compiling Nuget packages which are included in .NET Core.
  4. With .NET Core, it is easy for developers to create microservice-generated systems as per their choice. But this is not the case with the .NET framework.
  5. You would not find any robust tools or frameworks within the .NET Framework that makes it easier to simplify the mobile app development process. However, unlike the .NET Framework, you will find that .NET Core is easily compatible with Xamarin via the .NET Standard library.
  6. .NET Framework would be more expensive than .NET Core when we compare the performance and scalability of apps. With it, developers can improve the performance of apps without having to deploy any extra resources or hardware.
  7. When deploying web apps (developed using .NET Framework), the developers need to ensure that these apps are deployed on Internet Information Server. However, it is possible that the web apps developed using ASP.NET Core can be easily hosted in many different ways. It is possible for the developers to deploy the ASP.NET Core apps directly to the cloud.

- Differentiate between ASP.NET Web Form and ASP.NET MVC.

Asp.Net Web Forms	Asp.Net MVC
Asp.Net Web Form follows a traditional event driven development model. It is not Open Source.	Asp.Net MVC is a lightweight and follow MVC (Model, View, and Controller) pattern based development model. It's Open Source.
Asp.Net Web Form has built-in data controls and best for rapid development with powerful data access.	Asp.Net MVC is lightweight, provide full control over markup and support many features that allow fast & agile development. Hence it is best for developing interactive web application with latest web standards.
Asp.Net Web Form has server controls and state management (like as view state, session) techniques.	Asp.Net MVC has html helpers but with no automatic state management techniques.
Asp.Net Web Form has file-based URLs means file name exist in the URLs must have its physically existence.	Asp.Net MVC has route-based URLs means URLs are divided into controllers and actions and moreover it is based on controller not on physical file.
In Asp.Net Web Form, Web Forms(ASPX) i.e. views are tightly coupled to Code behind(ASPX.CS) i.e. logic.	In Asp.Net MVC, Views and logic are kept separately.
Asp.Net Web Form has Master Pages for consistent look and feels.	Asp.Net Web Form has Layouts for consistent look and feels.
Asp.Net Web Form has User Controls for code re-usability.	Asp.Net MVC has Partial Views for code re-usability.