**Hands on Day -6**

**INTCDE21ID008**

**C N S Varun**

**916214**

**ASP.NET Core**

It is actually a cross-platform, open-source framework for the development of internet-connected, cloud-based modern web applications. Through this framework, it is quite possible to create more effective web apps as well as services along with mobile app backends and even IoT apps. In much simpler terms, it is a significant redesign of the ASP.NET framework. It comes with a plethora of benefits like improved performance, tighter security, lesser coding and so on.

A lot of companies have duly adopted this latest technology in order to create their own applications. While ASP.NET remains to be a great combination of different web development models which consists of all of the required services in order to create robust web applications for different kinds of businesses, but here we will be discussing in detail about the various benefits of ASP.NET Core in order to develop robust web applications. It has led to a massive surge of **dot net programming**.

**Prerequisites**

* Basic knowledge of C#
* Visual Studio
* Object Oriented Programming concepts

## Improved Performance

The most obvious and important benefit of ASP.NET Core framework is its higher performance. With the new enhancements as well as upgrades, the code actually gets much more optimized that results into improved performance. However, this isn’t the best part of it. The most significant part is that you don’t actually require to change the code. The ASP.NET Core’s compiler will eventually optimize the entire code whenever the code is re-compiled utilizing the ASP.NET Core framework. ASP.NET Core’s actual performance is multiple times more than any of the famous framework implementations. It clearly shows that Microsoft has quite a long-term plan with this ASP.NET Core technology.

## 2. Support of Cross-platform

When it comes down to web application development, it is essential to ensure that the application actually supports all of the platforms. The latest ASP.NET Core is actually cross-platform that allows you to easily create web applications which run on Windows, Linux, and Mac. In much simpler terms, the entire backend will be using the same C# code. For instance, utilizing Xamarin, a business can easily create an iOS app and then eventually use the same given code for creating an Android app too. You can [**hire dot net developer**](https://www.cisin.com/service/microsoft-development.htm) who can utilize this cross-platform feature.

## 3. Lesser Code

The latest technology actually demands less coding, meaning that the developers can easily optimize the code-structure by means of writing much lesser statements. As coding is less, the less amount of hours are required in order to create an application which makes ASP.NET Core much more cost-effective.

## 4. Easier Maintenance

Whenever there is lesser code, it becomes much easier to maintain automatically. While it might not actually be easier for any new developer to easily understand this pattern, but a much-experienced developer needs to know how to actually optimize the entire code in ASP.NET Code with much lesser statements. It essentially means that it not only takes the lesser amount of code to create a web application but it is quite easier to effectively manage and maintain it. It improves the quality of Microsoft app development.

## 5. Cloud-Based Web Application Development Support

In case you have a business, it is quite a better option to create a cloud-based application in the current modern era. The major reason for this particular things is that the ASP.NET Core actually offers different types of web app development as well as mobile backend along with IoT application development. This means ASP.NET Core is the best solution for the business requirements of the current times. Also, ASP.NET Core can easily help in developing great and robust web applications.

### 1. Model

A model is data used by a program. This may be a [database](https://techterms.com/definition/database), [file](https://techterms.com/definition/file), or a simple object, such as an [icon](https://techterms.com/definition/icon) or a character in a video game.

2. View

A view is the means of displaying objects within an application. Examples include displaying a [window](https://techterms.com/definition/window) or buttons or text within a window. It includes anything that the user can see.

3. Controller

A controller updates both models and views. It accepts [input](https://techterms.com/definition/input) and performs the corresponding update. For example, a controller can update a model by changing the attributes of a character in a video game. It may modify the view by displaying the updated character in the game.

The three parts of MVC are interconnected (see diagram). The view displays the model for the user. The controller accepts user input and updates the model and view accordingly. While MVC is not required in application design, many [programming languages](https://techterms.com/definition/programming_language) and [IDEs](https://techterms.com/definition/ide) support the MVC architecture, making it a common choice for developers.

**Folders Structure**

**wwwroot**

The wwwroot folder in the ASP.NET Core application is treated as the web root folder and this folder or directory should be present in the root project folder.  In ASP.NET Core Application, the Static files can be stored in any folder under the web root folder and can be accessed with a relative path to that root. Static content is hosted in this folder like CSS, JS files, Bootstrap, jQuery libraries and images.

**Models**

A model is a class with .cs (for C#) as an extension having both properties and methods. Models are used to set or get the data.  If your application does not have data, then there is no need for a model. If your application has data, then you need a model. The Models in ASP.NET Core MVC contains a set of classes that are used to represent the domain data (you can also say the business data) as well as it also contains logic to manage the domain/business data.

**Controllers**

A Controller is used to group actions i.e. Action Methods. The Controller is responsible to handle the incoming HTTP Request. The Mapping of the HTTP Request is done using Routing. That is for a given HTTP Request, which action method of which controller is going to invoke is handled by the Routing Engine.’ Many important features such as Caching, Security, etc. can be applied to the controller.

**Views**

the View is the component that contains logic to represent the model data (the model data provided to it by a controller) as a user interface with which the end-user can interact. The Views in MVC are HTML templates with embedded Razor mark-up which generate content that sends to the client. We display information about the website on the browser using the views only.  A user generally performs all the actions on a view such as a button click, form, list, and other UI elements.

**Dependencies**

It is the place where the necessary dll.s for the applications are stored.

**Properties-launchSettings.json**

It describes how a project can be launched, whether the browser should be opened, and which environment variable should be set.

**appsettings.json**

It stores the information of connection strings or application settings.  It is similar to Web.config

**Startup.cs**

Startup. cs file contains Startup class which triggers at first when application launches and even in each HTTP request/response. Startup. cs file is entry point of application level it handles request pipeline. Startup class include two public methods Configure and ConfigureServices.

**ConfigureServices()**

The ConfigureServices method is a place where you can register your dependent classes with the built-in IoC container. After registering dependent class, it can be used anywhere in the application. You just need to include it in the parameter of the constructor of a class where you want to use it.

**Configure()**

The Configure method is a place where you can configure application request pipeline for your application using IApplicationBuilder instance that is provided by the built-in IoC container. The configure method includes three parameters IApplicationBuilder, IHostingEnvironment, and ILoggerFactory by default.

**CODE**

using System;

namespace FirstH

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Hello World!");

//Console.Read();

}

}

}

**OUTPUT**

