# **ASP.NET**

Dependency injection concept

It start from main method 🡪 HostBuilder 🡪 StartUp 🡪 startup have a two method 🡪 Configuration service & configuration 🡪 in configuration, declare specific own pipeline or create pipeline by use “USE” keyword.

IIS Express 🡺 It is a light weight server.

IIS 🡺 It is Heavy weight compare to IIS Express.

**MVC (Model View Controller):**

ASP.NET Core MVC is a rich framework for building web apps and APIs using the Model-View-Controller design pattern.

Create asp.net web(model,view,controller)🡺 to start create project.

Then create new repositary to store code into GitHub and create programing what we want then press F5 to run through the browser directly.

**Controller:**

The controller handles and responds to user input and interaction.

Localhost:2353/helloworld

Controllers folder – HelloworldController

[---It is default way to display the output

Localhost : 2353/{controller=Home}/{action=Index}/{id?}---]

**Ex:**

app.UseEndpoints(endpoints =>

{

endpoints.MapControllerRoute(

name: "default",

pattern: "{controller=Home}/{action=Index}/{id?}");

});

**Model:**

Business logic should be encapsulated in the model, along with any implementation logic for persisting the state of the application.

**View:**

A view is a component involved in the application user interface. This displays the required data or collects data from users.

**Advantages of MVC Framework:**

* Separation of concern means we divide the application Model, Control  and View.
* We can easily maintain our application because of separation of concern.
* In the same time we can split many developers work at a time. It  will not affects  one developer work to another developer work.
* It supports TTD (test-driven development). We can create an application with unit test. We can write won test case.
* Latest version of MVC Support default responsive web site and mobile templates.
* We can create own view engine. It is syntax is very easy compare to traditional view engine.

**View Data:**

1. ViewData transfers data from the Controller to View, not vice-versa.
2. ViewData is a dictionary type.
3. ViewData's life only lasts during the current HTTP request. ViewData values will be cleared if redirection occurs.
4. ViewData value must be typecast to an appropriate type before using it.
5. ViewBag internally inserts data into ViewData dictionary. So the key of ViewData and property of ViewBag must **NOT** match.

**ViewBag:**

ViewBag only transfers data from controller to view, not visa-versa. ViewBag values will be null if redirection occurs.

**ActionFilters:**

Action filter executes before and after an action method executes. Action filter attributes can be applied to an individual action method or to a controller. When an action filter is applied to a controller, it will be applied to all the controller's action methods.

**RedirectToAction:**

In HomeController.cs rediect the particular method to another use key word “**RedirectToAction”**

**Synatx:** *return RedirectToAction("Index", "Department", null);*

In Layout.cs add navigation item into the list

<li class="nav-item">

<a class="nav-link text-dark" **asp-area**="" **asp-controller**="Department" **asp-action**="Index">Department</a>

</li>

Name of the Middelware – Static Files, Self Contain Mode and Portable Mode.

When sharing the .net cor we want .net standard, Internal Webserver is Castreol and Exteranal web server is IISExpress and IIS

**Startup.cs 🡪 configure service**

**Transient** – Each request have create a new obj. same or diff client took request to asp.net core app. Obj create based on req

Syntax: services.Add(new ServiceDescriptor(typeof(ILog),new MyConsoleLogger()));

Eg.. News Article

**Scoped** – Every client create each one obj. Not create new in each req. Obj create based on client

Syntax: services.Add(new ServiceDescriptor(typeof(ILog), typeof(MyConsoleLogger), ServiceLifetime.Transient));

Eg.. Banking Application, Amazon, G-Mail

**Singleton** – Only one obj is create even so many no of client req. Helper Class

Eg.. Obj Reuseability

Syntax: services.Add(newServiceDescriptor(typeof(ILog),typeof(MyConsoleLogger), ServiceLifetime.Scoped));

**Entity FrameWork:**

Entity Framework (EF) Core is a **lightweight, extensible, open source and cross-platform version** of the popular Entity Framework data access technology.