Startup.cs

Program.cs

Model

IServiceCollection

IApp

2.Configure

1.ConfigureService

HostinEnv

Req

HomeController

Browser

Endpoint

Routing

Custom Middleware

Static Files Middleware

Razor Syntax : @

View

.cshtml

MVC Architecture

Request Pipeline

ASP.Net Core Platform

**What is MVC**

* **ASP.NEt Core Platform Supports MVC Architecture**

**It divides your application into 3 parts**

1. **Model : Class that represents data of an app. Typically is retirve and store model state of data**
2. **View : Display in UI**
3. **Controller : Handles the browser Request. Retrives a model and call view that returns a response**

**Adv**

1. **Separate different aspect of the app so it makes your application more testable**
2. **Helps to manage the complexity**

**ASP.NET Core Or MVC Framework is Convention based**

**Controllers : Handles the browser Request and contains application logic.**

1. **Each controller is derived from controller class. This controller class is capable of returning views.**
2. **Controller consists of multiple action methods**
3. **Generally Action Method return type is IActionResult. As it can return multiple types**
4. **Name of the action method is very imp. Because Framework try to find view with that name and return it to the user.**

**Issues**

1. **MVC is not enabled/ Controller and View not supported**
2. **Routing is not configured**

**Problem : All of My Pages , Index, Aboutus and contact . Should have consistent look and feel.**

**Solution : MasterPage/Layout/Common**

**Layouts**

* **Consistent and look and feel in my application**

1. **Adding \_Layout.cshtml inside Shared folder**
2. **Child pages will be loaded inside** @RenderBody()

**Problem : Index Page I want to display Product Information to the user.**

1. Product

**Solution:**

ProductInMemoryRepo

* Products
* AddProduct(p)
* GetProduct(id)

DI

Index Page

IStoreRepository

Controller

ProductSQL

**Problems : Index View @model .. Class name is too long**

**Solution : \_viewImport.cshtmlfile : and all the namespaces you need to put here**

**Problem : Only Display information. Add more functionality like , Delete, update, Adding new Product**

**Problem :**

1. **Link which user can click—**

* **i. HTML Helper**

**: Generate HTML content : Links, Forms control : Textbox , Labels …**

**: Methods**

* **ii. HTML Controls : Please don’t use it**

**:Its not well integrated with the ecosystem of ASP.Net MVC**

**Gives Better Options : Don’t use HTML Helper also**

1. **TagHelpers**

* **Problem with HTML Helper : Syntax**
* **HTML Helper syntax is not at all close to HTML syntax that most of the developers or designer understands.**
* **Syntax is bit complex and needs to be learned.**
* **TagHelpers syntax is same as HTML that’s why any front end designer can easily use TagHelper syntax to design page.**
* **Add Library : Microsoft.AspNetCore.Mvc.TagHelpers**

1. **On Clicking the link , Form will be presented to the user -- TagHelpers**
2. **User will fill the form and Submit the information -- Done Model Binding**

**Update Product**

* **Presented with the form.. It will have the data that I had selected to update**
* **I will update the data and Submit**

**Model Binding**

* **View Data is Directly associated to the Parameters of the action methods**

**Problem : Details View: When I click on ProductName .. I want to See More Details information about the Product.**

**CRUD Operation using ASP.NEt Core Application…. with Collections**

**Problem : So Many Product on same page**

**Solution : Pagination .. ☹ don’t have controls in ASP.NET MVC**

**Third Party : Kendo.. Very Expensive**

**Custom Pagination Control :**

* **1 2 3 4… Pagination Links : HTML Links I want to generate so that when I click on this link**

**Data wwill be loaded pagewise**

**Solution : Custom TagHelpers(PaginationTagHelper)**

1. **We need a model that stores a state of Total Items, ItemsPerPage, CurrentPage, TotalNoPages**
2. **Create class: PageLinkTagHelper : TagHelper**
3. **Use IUrlHelperFactory To Generate Urls inside this class**
4. **Use Process Method inside this class to write the behaviour code for this Pagelinks**

**View Models**

* **IS Responsible for Passing multiple models into a single page**

**Problems :** [**http://localhost:5000/?productpage=2**](http://localhost:5000/?productpage=2) **=> Userfriendly URL which will be more informative descritpi**

[**http://localhost:5000/products/page1**](http://localhost:5000/products/page1)

[**http://localhost:5000/products/page2**](http://localhost:5000/products/page2)

[**http://localhost:5000/chess/page2**](http://localhost:5000/chess/page2)

[**http://localhost:5000/Soccer/page2**](http://localhost:5000/Soccer/page2)

[**http://localhost:5000/chesss**](http://localhost:5000/chesss)

[**https://localhost:5000/Soccer**](https://localhost:5000/Soccer)

[**http://localhost:5000/Cricket**](http://localhost:5000/Cricket)

**Problem : Type URL : Click on the Links**

**Link :**

**Chess => Product**

**Cricket =>**

**Soccer =>**

**Solution : Use View Components : ASP.Net Core**

* **Inherit :** ViewComponent
* **Invoke function which is automatically**

**Solution**

**Routing**

* **Routing is responsible for matching incoming Request and dispatching it to executable endpoints**
* **Endpoints a: Request handling code**
* **Endpoints defined in the app and configure in the app when it starts**
* **Endpoint matching system that can extract values from URLS**
* **ASP.Net core uses Routing middleware to match URLS to actions.**

Endpoints

http://localhost/home/index

Actions:

Code

EP1

Routing

Code

EP2

1. **Routing : Route Matching . it looks for the set of endpoints defined in the app and Select best match URL.**
2. **UseEndPoints : Adds Endpoint . Its runs the Delegate associated with Endpoint**

**Route Constraints :**

* **Add Some validation or constraint to the Pattern of URL**

**Endpoints has Methods : Configure the Endpoints**

1. **mapGet : If you want to create test urls.**
2. **mapControllerToRoute**
3. **mapcontrollertodefaultRoute**

**Routing**

1. **Conventional**

* **Writing Route Code inside your Startup.cs and Use Endpoint**

1. **Attribute : ASP.NET MVC 5**

* **It allows you to write routing code in the respective controller files.**
* **Adv**

1. **It makes your Configure OR Startupcs much less complex**

**Attribute : Route,**

**Problem : User is able to add anything into the form and pollute the database with wrong data**

**Model Validation**

* **Validating the Input Controls**

ViewModel

Product

* ID
* NAME
* PRice

VL ProductEditModel

View : Needs Validation

ProdutViewModel

View : Doesn’t Need Validation Or may need different Set of Validation

**Best Practice : Validation Logic : Should be on ViewModel**

**VL Can be implemented using Attributes : Annotations**

**Annotations :**

* **By default work at the server side .. you need to click on submit button**

**Client Side JavaScripting**

* **Jquery**
* **jqueryValidation**
* **jqueryObstrusuve**

**Problem : I have a textbox : User is going to enter username. I want to check whether username is already taken or not**

* **DB :=> Method : Check User exists or not => Returns : username is already taken**
* **Need to call Server side Method to Check the Existence of the user**

**Solution : Remote Validation : Implements client side validation that requires calling a method on the server side to determined the input field is valid or not.**

**User: Should only add :Cricket,Chess and Soccer**

**True: Correct**

**Jquery Validateion Remote method**

* **Expects**
* **i. True: input data is valid**
* **ii. False : invalid**

**False: Incorrect Date**

**Problem : We don’t have annotation that can go to the sever and check whether username is taken or not OR Category information is correct or not**

**We need to call = Method of C# that has validation logic for the same**

**Solution : Remote Validation logic : To Validate we are calling Server Side C# method .**

1. **Create Server SideMethod that Validates the Category:**

* **VeirfyCAtegory : He will user… Cricket/Chess /Soccer**

1. **Include this Server Side Method on top of Field Where you want to apply it**

**Problem : Built in Attributes : StringLength , Required, Range …**

**Scenarion which is not Served by this built in attributes. I have to use it throughout the application**

* **Mfg Date Should not be less than the current year.**

**Solution : Create your own attributes : Custom Attributes**

1. **Inehrited : ValidationAttribute**
2. **It should implement isValid Method : that accepts value and validation information as aparameter**

**Problem : We have developed an application and deployed it. Application crashes. How we will know what was the issue**

**Solution : Logs**

**We keep those logs in some files**

**Microsoft have the logging interface. But it cant store logs in files.**

**Third Party Logging System : NLog, Serilog etc**

1. **Add Library for NLog**
2. **Create Configuration file : nlog.config : contains all the configuration related to logging**
3. **Configure your application to use NLog**
4. **Add Logging Capability to Respective part of the application throught ILogger<T>**

File System

NLog

Appliction

Application Insights

**Trends in the market**

**- JS Frameworks**

**- Performance**

**- Cloud**

**- Scalability**

**- Docker and K8s**

**- Reduce the infra cost**

**- Linux Adotopion : 70%**

**Where .NEt Stands**

**- .NET 2.0**

**- .Net 3.0**

**- 3.5**

**- 4.0**

**- 4 .5**

**Why .Net Core**

* **- Cross Platform**
* **- Easily Containerize and use with Orchestrator like K8**
* **- Cloud Ready : Easily deployable to cloud**
* **- Well Integrated with JS frameworks like Anngular and React**
* **- modular and lean : High Performance**

**Languages**

**- C#, F#, C++**

**Templates**

**- ASP.NET Core Web APP , Web API, Blazor, gRPC**

**How .NEt Now Cross Platform**

Native Machine Code

JVM / Core CLR : Linux/MacOs/Windows

IL: exe Or DLL

C# Compiler

C#

**Demo : Create Project**

**Project Structure**

1. **Program.cs**

* **Main Entry of the application**
* **Set up an application. By Triggerting Startup class**

1. **Startup.cs**

* **Setting up Request Pipeline**
* **Configuring Service Objects using couple of methods present inside startup.cs**

1. **Configure**

* **Request Pipeline : Requests past through this pipeline.**

1. **ConfigureServices**

* **Service Objects: DB Objects, Authentication**

1. **Appsettings**

* **Configuration Settings . Db ConnectionString, logging settings, Auth Settings**

1. **launchSettings : Any Project Start up settings can go here. Port number or Application Url Setting can go here**

**Why configureservice called first**

* **Whatver object Configure Or Application Methods needs in the Request pipeline is provided by ConfigureServices**

EmployeeRepository

Configure Service

Application

Service Objects

HomeController

RandomWrapper Service

Complex Logic

Lifespan

Scoped

Transient

Singleton

**Singleton**

* **Only one instance is created for all the request**

**Scoped**

* **Instance created per request**

**Transient**

* **Everytime Dependency is resolved Object Created.**
* **Within a Request , everytime that object is resolved. New object Created**

**RandomService : Generate the Random Nos**

**Consumed by HomeController and RandomWrapper**

HomeController

* IRandomService

Dependency Injection

Ctor Injection

Random Service

-IRandomService

-RandomService

RandomWrapper

* IRandomService

**Dependency Injection :**

* **Whatever Application needs will be passed as ctor injecton rather than creating pure object of it.**

**Dependency Inversion Principle.. Whatver u r providing try to use DI**

* **It make code unit testable and more mtaintable**

**Transient : Calculation logic ..**

Home

New

RandomService

New

RandomWrapper

Home

**Per request**

Single Object

**Scoped**

Current Request

RandomService

RandomWrapper

Second Object

**Singleton**

HomeCtrl

Single Object

RandomWrapper

**Class HomePage**

**{**

**HomePage(IEmployeeInter Emp) {**

Services.Add<IEmployeeInte, **EmployeeRepisotoryOracle > () .. DI**

**}**

**Emp.GetEmployee();**

**}**

**Interface IEmployeeInter**

**{**

**GetEmployee();**

**}**

**// Repsonsible for bringling data from SQL**

**Class EmployeeRepository**

**{**

**GetEmployee() { bring the data from SQL Database)**

**}**

**// Lets change Database from SQL Oracle**

**Classs EmployeeRepisotoryOracle**

**{**

**GetEmployee{ bring data from Oracle}**

**}**

**Class EmployeeXML**

**{**

**// Code to get dat for XML**

**}**

**Dependy Injection Comes :**

**Middleware**

* Segment of code**, that get executed when the Request pass through the pipeline**
* **Purpose of the middleware is to Modify Request or Response.**
* **Middleware takes parameter. Next ..**
* **Next : Next middleware to be called**

1. **Built in Middlewares**

* **Built by MS**

1. **StaticFile : Give Response for HTML Css JS**

* **Wwwroot folder files are available to the user without authentication**

1. **StatusCodePages**
2. **Authentication**
3. **Authorization**
4. **Routing**
5. **Endpoints**
6. **Custom Middlewares**

* **Built your own middleware**

**How To Create Your own middleware**

**Real World Example**

**Application**

* **Use : Can be use to Add Certain Code inside it.**
* **Run : Circuit Breaker . Stop the Execution**

Context

Middleware

Understand the Current Culture and Set it in the next middleware

Response : HelloWorld .FM, SM

Request Object : en-US Or fr-FR : Culture information

1. **Create a class “RequestCultureMiddleware” : Read the Culture Code Coming from the browser**

**Set the culture to Culture code.**

* **RequestDelegate Object as a parameter : that is responsible for calling next middleware**
* **InvokeAsyncMethod : Which is automatically called and contains a code for this middleware**

1. **Best Practice : Expose this Class through Extension Methods**

* **Use : UseMiddleWare To Return the Middleware**

1. **Add RequestCultureMiddleware this middleware inside Request Pipeline at appropriate place : Ordering is important**