Current Trends in the IT Industry

* Linux Adoption : Reduce the Cost.
* JS Frameworks: Performance
* Microservice Architecture: Independently developed and deployed.

Assign more resources to specific feature.

* Cloud : Dynamic Scalability.
* Containerization : Docker Or K8 : Reduces the Cost , by reducing the amount of resources required.

Microsoft .Net : 20

* .Net 1.0..1.1 : Console App
* .Net 2.0 : Win Forms, WebForms
* .Net 3.0
* .Net 3.5 : Linq
* .Net 4.0
* .Net 4.5 : Async .. Await

Microsoft Rearchitected :

* .Net Core 1.0
* .Net Core 2.0
* .Net Core 3.1 == This
* .Net Core 5.0 : VB and Support EF6 removed

What is .Net Core

* Framework to Build Cross Platform Applications
* Cloud Ready
* Integrated with JS frameworks
* Easily create MS and Containerize

What Languages Support

* C#, F#, C++ as of 3.1

Application we can Create

* Windows Forms
* WPF
* Console Application
* Web Application
* gRPC
* Blazor = Angular Or React

How come .Net Core is Cross Platform

* JVM was present for every environment
* Core CLR Present for every environment

What is Requirements

* Visual Studio 2019 : Free
* SQL Server 2012 and Above OR SQL Express(Free)
* Postman : Free

Real World Application : Administrator to maintain Product inventory

[swpnlgkwd30/BoaSessionFour (github.com)](https://github.com/swpnlgkwd30/BoaSessionFour)

Lab1 : Create sample application

Project Structure

* Program.cs : Entry Point of your application.
* Startup.cs : Configure your application.

Configure : Responsible for Handling Request and response . It contains Middlewares.

ConfigureService : Responsible for Creating Service Objects. DB Objects, File Objects , Security Object .

* appSetting.json : Settings of your application.
* launchSettings.json : Modify port number or application url Or set up environmental variable.

Startup.cs

Program.cs

EF

Objects : DB, File

Model

1. Configure Service

2. Configure

Req

Browser

Ctrl

View

.cshtml

UseEndpoint

Routing

Middleware

Middleware

Razor Syntax : @

MVC

Pipeline

ASP.Net Core Platform

Middleware

- Section of Code that Modifies the Request and Response.

- Middleware is software that's assembled into an app pipeline to handle requests and responses.

- Middleware takes Request Delegate as Parameter

- Takes 2 Parameter

I .Context : Current Request and Response Object

ii. Next : Next middleware to call

Use : Takes Request Delegate as a parameter. Context , next .. if you don’t use next, next middleware will

Not execute . This is called as Short Circuiting the Request

Run : Use to Short circuit the Request. IT doestnot have next parameter

Types of Middleware

Built In : MS Provided

* DeveloperExceptionPage
* Routing
* UseEndpoint
* UseStaticFiles

Custom : You have to create your own

* Our own Middleware

Problem

* I have some CSS, JS , HTML in my project . I want to make it available to Browser whenever Requested

Solution : Whenever you want to make available Such Static files to the browser. You need to use Middleware : UseStatic

Static files, such as HTML, CSS, images, and JavaScript, are assets an ASP.NET Core app serves directly to clients by default.

Problem : wwwroot : Content from this folder is available without any auth . Important Static docs cant be stored here.

Solution : Create your own folder and Keep those docs inside this. Expose this folder from Pipeline and configure it to make available only after authentication.

Web application

* ASP

:- Runtime Error

: = Interpreted language

* ASP.Net Web Forms

Adv

* Faster Development
* Rich controls
* Compiled

Problem

* Testable
* Tightly coupled
* Don’t have control over HTML its going to produce.

ASP.Net MVC 1.0

* MVC Convention based not Configuration based.

Action Name

Controller Name

Model

Request : http://localhost:1385/Home/Index

Controller

Views

UI

View

Model : Holds Data and business logic

View : User interface logic

Controllers : handle the request.

Adv

1. Separation of Concern
2. Unit testable
3. Maintainable

2.0

3.0

4.0

MVC 5.0 = Develop and Deployed only on Windows

Problem

* Doesn’t have Rich Server Controls

ASP.Net MVC Core

* Build Cross Platform Web Applications.

Conventions

1. Controller should be present in Controllers folder
2. UI logic should be present in Views Folders
3. Every Controller name has to end with Controller word.

Problem : Home Page : Home, About Us and Contact Us. How to do this in MVC

Solution

* Home Controller

1. 3 Action Methods

Controller

* Handle The Requests
* Connect the model if required and pass data to View.
* Inherited with Controller class

:- It gives the methods required to return different types . View, Component, Partial View,JSON.

IActionResult

* Base Type for the all the other types
* Json, View , PartialView

Configuring Services

* Configure the Services.
* How many such object created is depend on the lifetime used.

Lifetime

1. AddTransient :

* Within a Request whenever Its resolved . One object created.
* Small Calculation logic

RandomService Object

HomeCtrl

Request 1

RandomService Object

RandomWrapperService

1. AddScoped : Per Request one Object is Created . DB Object or File Object

HomeCtrl

Random Service Object

RandomSWrapper

Request1

Random Service Object

Request2 ..n

1. AddSingleton : All Request one object is Created . Sharing Some Db .. in memory db

Problem : Pass Data from Controller to View

1. View Data Or ViewBag

* Problem

1. Doesn’t give Intellisense
2. If you make typo , you might lose data on screen

Drop down : selectedCategory = Electronics

1. Model

* Intellisense
* Reduce the Errors

Application : CRUD Operations Product Admin.

1. Controller : Home Controller --Done
2. Model :- Product Model --Done
3. Services that can perform this operation.
4. Consistent Look and Feel for all the Pages. -- Done

Problem : Consistent look and feel for all the Pages

Solution : Layout : MasterPage

Services : How to Create

* AddProduct
* DeleteProduct
* UpdateProduct
* GetProduct
* GetProductByID

1. Action Method
2. Action Method will call Service

HomeController

ProductInMemory

IStoreRepository

Service : Done

Problem : Craete a Link => Page that will display options to add new Product

Or Forms . We are not suppose to use HTML code directly

* <a href="Home/Create"> Add Product</a> : Not a good way

ASP.Net MVC 5

1. HTML Helper classes : Well integrated with ASP.Net MVC Framework

@Html.ActionLink("Add Product","Create","Home")

@Html.TexxtboxFor("Add Product","Create","Home")

@Html.Checkbox("Add Product","Create","Home")

Problem

* Anyone who is not C# guy , it will difficult to understand
* Design Web Page => HTML

MVC Core

1. TagHelpers

* Syntax is more like HTML. Designer can use & write it.
* Use to generate HTML elements on View
* Its also well integrated with ASP.NET MVC Framework.

1. Add Package for TagHelpers
2. Enable TagHelper for your application.

Random Service Object

Request1

Request2 ..n

Lab2 : How to Create Services and Configure the lifetime of the Same.

* Create RandomService : Going to Generate Random Nos

i. IRandomService --

ii. Implement --

* Random Service = > Controller Class

Singleton

1. Dependency Inject

Scoped

Transient

Random

HomeController

DI Syntax

IRandom

Service : Logic (DB)

Why it should be implemented as Interface .

* Loosely coupled.

OracleDatabaseService

GetData :

IDBInterface

GetData

Ctrl

IDBInterface I ;

i.GetData();

SQL

MySQLDtabaseService

-GetData

MySQL

Request => HomeCtrl => I Product :Oracle DB Service

Request => CustomerCtrl => Customer : MySQLDB Service

Empty Project

* launchSet,prope,appsetting, etc

gRPC : Web Service

Services.AddgRPC : Classses

Blazor = MS Version of Angular

Services.AddBlazor : Component

ASP.NEt Core Web App : services.AddController

Pass Data from View to Controller

1. Asp-route : Add Additonal Parameters to route and pass the data
2. Submit Action to Pass Data

Problem

* Number of Products are really huge and doesn’t fit into a single Page

Solution

* Pagination
* View having Links

1. Razor
2. TagHelpers

Custom TagHelpers

* Create a TagHelper which will work as Pagination.

1. Need a model that can store the information about the Current Page
2. Rewrite the query which can retrive the data based on pageSize

ViewModels

* Whenever we want to have multiple models within a single view . We should create viewModels.
* Class that represent the model of view

Routing Feature : http://localhost/Product/Page3

http:// localhost/Chess : Give me all records related to chess

<https://localhost/Cricket> : Give me all record related to Cricket

<http://localhost/chess/Page1>

Problem : User Should be able to navigation to these links

Solution : Create Links on the Left Section so that user can select particular categories.

Create Numbers of Categories and Generate the Links.

View Component

* Similar to Partial views
* Render chunk
* Business logic
* Invoked from layout page

<https://docs.microsoft.com/en-us/aspnet/core/mvc/views/view-components?view=aspnetcore-5.0>

Real world Situation

1. Dynamic Nav Menu etc

Problem : Project is Ready and deployed. We get complaint from the Client that something is not working.

Solution: Logging .

How

* ASP.Net core supports logging but doesnot support logging in files.

Third Party logging providers

* NLog : <https://github.com/nlog/nlog/wiki>

: NLog is a flexible and free logging platform for various .NET platforms

* Serilog

Steps to use NLog

1. Download the library
2. Create Config file
3. Enable the Nlog for application
4. DI
5. Log it

Problem : Interact with FS

Solution : File Providers

<https://docs.microsoft.com/en-us/aspnet/core/fundamentals/file-providers?view=aspnetcore-5.0#:~:text=the%20directory%20folder.-,ASP.NET%20Core%20abstracts%20file%20system%20access%20through%20the%20use,Providers%20to%20locate%20static%20files>.

ASP.NEt Abstracts file system through File Providers.

Internally , Webhosting Environment and static files use File Providers.

PhysicalFileProviders: Wraps System.IO files and provide access to Physical file system

IFileProvider

File System

Application

System.IO

Routing

* Responsible for matching HTTP Request and dispatching it to the executable endpoints.
* Endpoints are the apps unit of executable Request handling code.
* Endpoints can also extract the values from the Request URL.

Middlewares

1. Router : app.UseRouting

* It looks for the set of defined endpoints and select best match based on the request.

1. Endpoints : app.UseEndpoints

* Configuration of Endpoints

Home

Index : Pageno

https://local/Customer/index/1

UseEndpoints

Routing

Create

Customer

ASP.Net MVC 5 :Previous : Separate file use to exist : routeconfig.cs : Routing related logic

<https://docs.microsoft.com/en-us/aspnet/core/fundamentals/routing?view=aspnetcore-5.0>

Types of Routing

1. Conventional

* Routing logic in the Startup.cs.
* Endpoints

1. MapGet : Route that is not associated with any of the controllers. Execute some logic based on the route.

endpoints.MapGet("/authorize/{username}", async context =>

{

// read this route price

var userName = context.Request.RouteValues["username"];

await context.Response.WriteAsync("Hello : " + userName);

});

endpoints.MapGet("/productinfo/{price}", async context =>

{

// read this route price

var price = context.Request.RouteValues["price"];

await context.Response.WriteAsync("Hello : " + price);

});

1. MapControllerToRoute : Request Navigate to Specific controller.

endpoints.MapControllerRoute("catpage", "{category}/Page{productPage}",

new { controller = "Home", action = "Index", productPage = 1 });

// Route : http://localhost:5000/Cricket

endpoints.MapControllerRoute("cateogry", "{category}",

new { controller = "Home", action = "Index", productPage = 1 });

//Product/Page3

endpoints.MapControllerRoute("pagination", "Product/Page{productPage}",

new { controller = "Home", action = "Index" });

1. MapControllerToDefaultRoute : Request navigated to Default Route. Controller : Home, action =-index

<http://localhost/customer/index>

* First Part : Controller
* Second Part : Action

If we don’t pass anything by default values will be : Home and Index

1. MapAreaControllerToRoute = Application into Area.

Routing Constraints

* Add a validation into the Route Parameters

endpoints.MapGet("/authorize/{username:minlength(4)}", async context =>

{

// read this route price

var userName = context.Request.RouteValues["username"];

await context.Response.WriteAsync("Hello : " + userName);

});

endpoints.MapGet("/productinfo/{price:int}", async context =>

{

// read this route price

var price = context.Request.RouteValues["price"];

await context.Response.WriteAsync("Hello : " + price);

});

// Route /Chess/Page2

endpoints.MapControllerRoute("catpage", "{category}/Page{productPage:int:min(1)}",

new { controller = "Home", action = "Index", productPage = 1 });

1. Attribute : ASP.NET MVC5

* Routing logic in the specific files.
* Route attribute to specify route to specific action.
* Primary Adv

1. It allows you to define your routes in the same file as a controller so it make sure your startup.cs is more clean.

[Route("Account")]

public class AccountController : Controller

{

[Route("Login/{username:minlength(4)}/{password:minlength(4)}")]

public IActionResult Login(string userName,string password)

{

return Content("Login Called");

}

Problem

* Able to add Empty product into my DB

Solution

* Model Validation

Validation of ASP.Net MVC5 and MVC Core = Same

* Data Annotations

ViewModel

Create

Name of Product

Valid

Model : Product

Valid

Update

Product Name

Index

ProductID

Views tightly attached to model. So if anything changed in the model, every view which is using it

Will be impacted.

One View: Name : Price

Second View : Name :Category

By default this is Server side validation.

Remote Attribute

* Implements Client Side Validation that requires calling a method from the server side.
* Expects JSON response.

1. True : data is valid
2. False : Data is invalid

Real World Example

* User enters username in textbox, we have to validate whether username is already taken or not.

Logic

swapnil

Json : true Or false

Category : Cricket , Chess and Soccer : ServerSide Logic which will validate this.

AutoMapper

* Problem : As a Best Practice when we introduce ViewModels in our project we have to convert VM=> M and Model=>ViewModel

Solution : AutoMapper

* Package the Converts One object to another object.
* AutoMapper is a popular object-to-object mapping library

1. Install the Package—Automapper and DI
2. Create a class that should be inherited from Profile class
3. Constructor write a Mapping Logic
4. Use IMapper interface
5. Configure Service to find Mapping logic in the current Assembly.
6. Use Map Method.

Developer Exception Page

* Contains lot of sensitive information about your code. So it should be available only during development and not on Production.

Solution

* It should Create Custom Error Page.

: Controller

: Index View

Status Code

* An Additional method of handling errors.
* UseStatusCodePages Middleware
* Use to Capture html status codes between 400 to 599
* Display short explanation of error code to the user.

Areas

Problem

* Naukri.com : Application is used by 3 Users

1. Employer : Area

* Set of Function
* 10 Controllers \* Models , \*ViewModels

1. JobSeeker : Area

* Set of functions
* 20 Controllers

1. Administrator : Area

* Set of function
* 30 Controllers
* ASP.NEt MVC5 == ASP.NEt Core
* Organize Related functionality into Group as a Separate.
* Partition an ASP.Net Core Web app into small functional group

Functional Group

1. Pages
2. Controllers
3. Views
4. Master Pages
5. Application
6. Developer : Linq
7. User

When to go for Areas

* App made up of multiple high level functional components that can be logically separated.
* Partition an app so that each functional area can work independently.

Problem

1. We have two users i. Admin ii. Customer

Solution

* Areas

I .Admin

1. Customer

Problem

* Data is Coming from the in Memory Collection.

Solution

* Connect to Database.

Entity Framework.

Problem : Data Access Code before EF : ADO.Net : cmd. commandExcuteQuery(“select \* from Product”)

* Lot of Runtime Errors
* No Rich intellisense.
* Lot of Code.

Object Relational Mapping : EF

Shopping

Sql Server : Relational

ShoppingContext

* ConnectionString

DbSet<Product>PRoducts

Objects

Tracks

Products

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name | Cost | Category |
|  |  |  |  |

C# Application

Class Product

ID,

Name,

Cost,

Category

EDMX

Table = Class

Column = Properties

Database : DataContext

Advantage

1. Developer have to write less code.
2. Intellisense
3. Less Run time Errors

What is EF

* Object Relational Mapping Framework use to Connect to Database.

Different Versions

1. 5.0 : .NET Framework
2. 6.0 : .Net Framework

EF Core

* Lightweight, Open Source and Cross Platform Version of EF.
* EF Core 3.1 runs on .NET Core and .NET Framework, through the use of .NET Standard 2.0. However, EF Core 5.0 does not run on .NET Framework.

1. .Net Framework Application can use EF Core 3.1
2. Application that can have both EF6 and EF Core 3.1
3. ASP.Net Core cant use EF6.0 after 6.3 you can run EF on ASP.Net Core

<https://docs.microsoft.com/en-us/ef/efcore-and-ef6/>

Compare EF6 with EF Core

1. EF Core does not have Graphical Visualization
2. Ef6 Supports EDMX file that is not available with EF Core.

How to Integrated EF Core in ASP.Net Core Application

1. Install libraries :

* EFCore
* EF.Design
* EF.SqlServer

1. Build Context Class :
2. Configure and Read ConnectionString

* IConfiguration

1. Enable Context and Pass the ConnectionString from Startup.cs
2. Data Access Code :
3. Change the Ref to enable SQL Repo
4. Migrations : Generate necessary tables and Database Required by your application.

PM> dotnet ef migrations add Initial

No project was found. Change the current working directory or use the --project option.

PM> dotnet ef migrations add Initial --project sample-app

Build started...

Build succeeded.

Done. To undo this action, use 'ef migrations remove'

PM> dotnet ef database update --project sample-app

Build started...

Build succeeded.

Problem Whenever any sales happens the Inventory Quantity should be reduced by 1. Both operations should be atomic.

* Sales
* Inventory

Transaction support of EF Core.

* Commit, Rollback.

Problem

* Existing .Net Application : EF 6
* Performance : EF Core

Solution :

* .Net Framework => EF Core

Database first approach for EF Core.

* Scaffold-DbContext "Server=Swapnil-PC\SQLEXPRESS;Database=BoaSessionFourSat;Trusted\_Connection=True;" Microsoft.EntityFrameworkCore.SqlServer

Web API

Need

* When 2 application wants to communicate with each other on Internet.

App2

App1

Solution

1. Remoting : .Net => .Net , Binary , Fastest
2. Web Services : .Net => Java, SOAP , Slow
3. WCF : SOAP,HTTP,Faster . Problem : Lot OF Configuration
4. Web API : HTTP based Service, Format : JSON
5. GRPC : Performance wise 7..10 times better than Web API.

Rest Service

* Standard for interactive applications that uses Web Services
* Provides web resources in a textual representation to allow them to read and modified with a

Stateless protocol and predefined operations.

Web API

* Accept the Request and Generate Responses that contain data
* Provide access to application data to Client side applications (Angular, React Ember etc)

<http://localhost/product/Getproduct>

HTTPMethod + HTTPVerb

Application 2

* Web API

GetProduct

AddProduct

DeleteProduct

UpdateProduct

Application 1

<http://localhost/product/Getproduct>

-Angular , React,ASP.Net Core Web

HTTPVERB: GET PUT,POST ,DELETE

JSON format + Status Code

Project Structure is same as ASP.Net Core

1. Controller : ControllerBase

* Access to features from MVC framework

I .HttpContext,ModelState,Request,Response,RouteData and User

* Makes Sure you cant return view from APIS.

1. Controller Attributes

* Route : [Route("[controller]")] = Enabling Attribute Routing
* [HttpGet] HttpPOST,HTTPDELETE, HTTPPUT

1. Binding Source
2. FromBody : Request Body
3. FromQuery : QueryString
4. FromRoute: Route Parameter
5. FromHeader : Header Information

How to Connect to Database and Get the Data for Web API.

* Libraries --
* Context --
* ConnectionString and Read it
* Enable the Context in startup.cs
* Use Context in Controller

Testing APIS

1. Browser
2. Postman : Useful During Development API Testing
3. Swashbuckle: GUI that you can add for Testing your api.

Best Practice

1. Create Asynchronous Action Methods : Wherever Async method is der use it.

User1

R1 : T2,

:.. 1 min

R2 :.. 2 min

R3 : T3 : 3 min

R4 : T1

1. Pass Valid Status codes to Client Application

Let say Get Product that return null in this case we need to send Not Found that is 204.

1. Ok : 200 : optional Object in the response body
2. BadRequest: 400 model state error
3. NotFound: 404
4. Validate Data through Data annotations
5. Project Selected Properties
6. Use Swashbuckle : Open API Specification that will automatically generate description of web Service.
7. Install the Library

dotnet add package Swashbuckle.AspNetCore --version 5.0.0-rc2

1. Enable Swashbuckle for your application.
2. Content Formatting

* Web API always returns a data in JSON format. But web api is capable of sending multiple format based on the configuration.
* Content Format Selected based on

1. Format the client accept.
2. Format that application can produce.
3. Content Policy Set by the action method.

Content Negotiation

* Most clients include an Accept header in a request which specifies the set of formats that they are willing to receive in the response.

Respecting Headers

* Whenever you asked for a format that doesn’t supported by the application. Web API will always give you JSON format.
* Ideally in this case application should get 406 error that is format is not available

1. Adding Produces and Consume Type

Type : GET/PUT

URL : Where

Success : function

Failure : function

datatype:json

Consuming Web API

App

C#, Angular , React

Web API

GetProducts

GetProductId

HTTP Library

* Http
* Axios
* Ajax
* HttpClient
* jQuery => Ajax : Web API Call

Enable Cors

* Browser security prevents a web page from making requests to a different domain than the one that served the web page. This restriction is called the *same-origin policy*. The same-origin policy prevents a malicious site from reading sensitive data from another site

ASP.Net Core Identity

* What is Identity
* Enable Identity
* Login,Logout, Register --
* Json Authorization
* AAD
* Membership system for web application
* API the supports login, logout and Register functionality.
* Manage user , passwords , profile data
* Identity can be configure to use SQL server DB or Azure TableStorage.

Microsoft.AspNetCore.Identity.EFCore

SignInManager :Login and Logout

UserManager : Managing users.

IDentityUser : Represent user

IdentityRole : Represent Role

SQL

Identity Tables

Application

Login

Logout

Register

User

* User is Represented using IdentityUser class.

How to Enable Identity

1. Install the library : Identity.EfCore
2. Create User class : IdentityUser
3. DataContext class should be inherited from IdentityContext.
4. Enable Identity in Startup.cs
5. Lets create default roles for the user.

Registration

* Controller : AccountController :--
* Views : Register :--
* UserManager

Login and Logout

* Controller : AccountController
* Views :
* SignInManager

Problem : Allow Access to user only if he is logged in.

Solution : Filter : Authorize

Problem : Login, Logout and Register all 3 options are always available on our screen.

Solution :

1. Layout hide the links based on the users logged in status.