ASP.NET Core:-

 it is a infrastructure means set of interface and classes for building cross platform server side application using .NET . it has it own server where application is run that is called Kestrel.

1. Request Pipeline

request will be send to Kesteral server that request is sent to ASP.NET core that consist on middleware or chain on delegate (It is also called request pipeline)

 first middlware is RoutingMiddleware 2nd is OtherMiddleware and last will be always End Point Middle Ware.

Eeach Middleware (Delegate) will do the process and would call next delegate.Ecah delegate would perform some operation on the request

to write middleware  class there is 2 things must be there first is constructor which take argument type of (RequestDelegate next)

  and your middleware class must contain invoke method and it takes one parameter of (HttpContext) Type  and it must return task

for Example Authorization and Authetication , useStaticFile, useExceptionHandler(“/Home/Error”) is middleware if we define in our application then definetly it would execute

1. Service Container:- it creates service class object and inject it whenever we are required.

**Middleware:-**Asp.Net core introduced a new concept called Middleware . A middleware is nothing but a component that is class which executed on every request in ASP.Net core application. it is a software component that is define in request pipline that handle request and response.

It is also used to handle exception. Middleware is also used in authenticate and authorization. Order of middleware is very important.

**Routing Middle ware:-** This is the first middleware in request pipeline . Routing middleware handle the request from the client . whatever the request comes through url routing middleware select the right resources which is mapped to resources. basically it select the Controller and action method , handler method.

Conventional Based Routing:-

Program.cs

Builder.Service.AddControllerWithViews()

App.MapControllerRoute(); // by default it will call home controller and Index action method

App.MapControllerRoute(“default”,”{controller=Home}/{action=Index}/{id?}” id is optional

Attribute Based routing:- In attribute base Routing we use [Route] attribute is used to define the routes.

Builder.Service.AddControllerWithViews() we have to register this service to make mvc application

App.MapControllers() we have to use this middle ware to implement attribute Routing

Above action method [Route(“ ”)] [Route(“ Home”)] [Route(“ Home/Index”)] [Route(“ Home/Index/{id}”)]

By all request url same action method execute if we have multiple Route in same action method.

[Route(“ Home/Index/{id?}”)] public int Details(int id) if id not passed then 0

[Route(“ Home/Index/{id?}”)] public int Details(int? id) if id not passed then null id return id ?? :1 ;

Return(“~/Views/Home/Index.cshtml”);

Route on Controller and action method on controller Route[“Home”] ,on action Route[“Index”]

Route[“~/”] on vacant url this action would me executed

[Route(“[controller]/[action]”)] this can be implement on class so that it act asa place holder

Routing With Http methods:- any request comes from browser it defenitly comes with url and any one http methods.

http methods can be get , post , delete and update .

app.Map:- if we are not sure that request can be get, post , delete, update . this method would execute

app.MapGet(() =>{ }):

aap.MapPost(() =>{ }):

aap.MapUpdate(() =>{ }):

aap.MapDelete(() =>{ }):

but if have to write multiple line and we have to get some object from form and we have to define some custome logic so we will use dfiffere method for that

we have to define middleware app.UseRouting();

then we have to define EndPoints like that

app.UseEndPoints(endpoint=>

{

Endpoint.MapGet(“/Home”, ( async context)=>{ await multiple line and custome logiv});

Endpoint.MapPost(“/Home”, (context)=>{multiple line and custome logiv});

Endpoint.MapDelete(“/Home”, (context)=>{multiple line and custome logiv});

Endpoint.MapUpdate(“/Home/{id}”, (context)=>{multiple line and custome logiv});

}

All the methods is used to define endpoints. endpoints checks url and http method and respective it calls methds

**Controller:-** Controller layer act as an interface between Model and View component . in controller layer we write business logic .controller layer contact to database and it would fetch the data from database and update ,delete etc.

**Controller to View data passing:-**

1. View Data
2. View Bag
3. Temp Data
4. Strongly Typed Views

View Data:- View data is a object if we have to send data from controller to View then we need to send this object. ViewData [“key”]=value (that can be of any data type)

View Bag :- ViewBag.<PropertyName>=Value of any data Type

Where Property is a string value that represents a property of ViewBag. In View Bag we use property so when we get data from collection we not required to typecast. This is and advantage of View bag as respective to Vie data.

View Data can be access by View Bag. Whatever we have key in View data in View page we as a propertname.and vice versa can be access

View Bag is a dynamic data property of the base class of all the controllers , which is the cotrollerBase class.

TempData:-

It passes data from controller to View .