Topics that we will Cover:

* Asp.net MVC building blocks,
* CRUD operation with Entity FrameWork Core,
* Build RestFul Services,
* Asp.net Identity,
* Paypal integration
* Azure Development
* Model Binding

By the end of this course I will learn how to use asp.net frame work and use it in a personal and commercial webapplication

Step by step what we’ll be learning and doing :

1. Whatis MVC ? (Learn about how to set up the views model and controllers)
2. DBContext file which is used to interact with database
3. CRUD data using EFCore
4. Services ( to work with the data)
5. ViewComponents ( To add and edit the shopping cart )
6. Authentication and Authorization.

Other topic include:

* PayPal SDK for online payments,
* Dependency Injections,
* Model Binding,
* Routing
* Model Validations
* Tag helpers

**MVC ( Model View Controller)**

A model is just a C# class that represent server in the sql format:

* It servers as a data blue print,
* And also help to find the data relation.

Eg : Authors, main characters and anything related to to database we need to create the tables and to create the tables we need to create the models,

**View**

is what user get to see it is a file that represents user interface

* It have cshtml ( cshtml is razor meaning the combination of html and c#)
* It is used to trigger the event.

**Controller:**

* It receives the event,
* Returns the data to the view,]
* A controller is just a C# class that inherits from the Controller base class,
* Through this course we are going to create, author controller, main character controller, manga controller.

**Web Applications**

How does it work ?

A computer screen with arrows pointing to a server

Description automatically generated

From the above picture

1. First the browser sends the request to the server,
2. Server then look the request in the Database and get the item that it wants,
3. DataBase then returns the item to the server,
4. Server then send the item to the browser

**MVC**

**A screen shot of a computer

Description automatically generated**

From above picture:

1. View first sends the request to the controller,
2. Controller then sends the request to the Model which in this case is a database
3. Model then analyze the request and sends the data that needed to be send to the controller,
4. Controller and then send data to the view, which is later on displayed to the client.

Inside the Microsoft visual studio code

A screenshot of a computer

Description automatically generated

* Inside the connected services we have the we can service like wcf, authentication with azure and many more
* Inside Dependencies, we have analyzer and the framework.
* Inside properties we have launch setting of Json file where our app is going to run
  + There are profile inside the properties where we can define command names, .net messages, the launch browser (the application url ) which is important, and environment type
* There is root section next in which we have static files which include css, js, and custom libraries
* Controllers, inside we have the C# class which is inherited from the base class controller
* Models,
* Views , here we have the home file because we have the home controller, so each test controller will have the test folder in the views, the shared contains the layout of the application which contains the razor view]
  + - There we have the header for the navigation bar
    - We have class container and the renderbody() is used for the rendering all of the views that we have
    - Down we have the footer and some scripts
  + Anything else are the default files created by the system, we will continue if there is need to talk about them or else we can just leave them as it is
* In the appsetting.json file is called configuration files because here we store the data like configuration string, the loggin related configuration,
* Other is the program.cs which is the main file

**Application Model Overview,**

Application Features,

* Create, Read, Update and Delete (Except Manga)
  + Main characters,
  + Authors,
  + Studio
  + Movies
* Adding items to a cart
* Orders
* Authentication

We are going to store the data in the database so we are going to create the respective models

Initial Models

* Main characters
* Authors,
* Studio,
* Manga.

For there we are going to:

* Add the relation to these Models
* Configure these to SQL data base,
* Also add some test data to the data base.

Inside the Microsoft visual studio 2022,

We created a folder called data, where we are going to put the data related stuffs like: enum, servies, view models etc

**Adding the DBContext file?**

What is Dbcontext file?

* A computer screen shot of a file

  Description automatically generatedTranslator file between the model class and the database,
* Here if we have to send data from the model to the database and receive data from the database, we need dBcontext file cause database contains the SQL language, to understand the C# code be understood by the database and vice versa we need dBcontext file which will work as a translator,
* We can create an custom Dbcontext file from inheriting from the base of Dbcontext file.

We need to add the dbcontext confriguration in program.cs like builder.Services.AddDbContext<AppDbContext>();

A diagram of a movie

Description automatically generated

Here in the diagram we have the joint table called character\_movies, cause there was a many to many relationships between the mangas and the character, we are splitting many to many to relationships into two relationships so there will be 2 one to many relationships.