Final project Demo : https://bulky.azurewebsites.net/
Content course : https://www.dotnetmastery.com/Home/Details?courseId=9
Github material: https://github.com/bhrugen/Bulky_MVC

N-Tire Architecture: means dividing our pages into distinct layers and responsibilities are also divided.
Add important features of .net core here!

Dependency files:
If we want to use another project in our current project we need to define it as a dependency.
More packages We'll add to this part when we need more functionalities like using databases, authentication and …
Global variables are defined in properties and asp environment → "ASPNETCORE_ENVIRONMENT": "Development"

Wwwroot:
All the static content of your .net project is there → NuGet packages, CSS, JavaScript, images, files, etc. They do not have html codes.

appsettings.json:
here you will be the host for all your connection strings. Not all of them but the secret keys that you need for your application.
Example of, in databases, you'll be adding your Azur account here or for email, you'll be adding that EmailGrid you are using or any specific keys for showing that it is for db or for email.
So whenever you think about adding or editing the connections, you will go through appsetting.

Program.cs:
It is your main file. Two important things :
1_we need to add some services
2_configure the request pipeline:
Pipeline means when a request come to your app then your application how should handle or process it. In the pipeline we have app.Environment.IsDevelopment() , this development is the global variable which we defined in the Dependency file.
The other lines are like app.UseSth()asks that call the controllers, statics files, etc. to start.
Here is the place where you want to define and edit the middleware.
MapControllerRoute = Remember two keys which is necessary for the controller
1_ action
2_ controller
Now let's talk more about the MVC frameworks
1_ Model: Represent the shape of the data. Mean if you have any classes or tables. For example, in the e-commerce application, you have the class of productions, orders, details, shopping cart and etc. The model of your application has all of these classes.
2_View: it is the user interface, it handles the html files of your project.
3_controller: it is between the model and the view.
Total work of its: first user requests to the controller, it goes to the model and fetches data then pass it to the view.

```
app.MapControllerRoute(
   name: "default",
   pattern: "{controller=Home}/{action=Index}/{id?}");
```

this line of code shows the route of our work. Here we define the default route which said that if nothing is called then go to the home controller

Routing pattern in MVC:
http:/ourWebDomain/{controller}/{action}/{id?}
action= index, edit … id= it is optional.

For each page, we have a separate controller, but the model is optional. For view, we have separated folders with the exact name of our controllers.

The URL shows the route for our controller: https://localhost:7241/Home/Privacy. In this example, the controller is "Home". However, it also works if instead of "Privacy", we write "Index" (thanks to the default mapController setting). After specifying the controller, we need to call the action. For this, we use the following code to determine what should be returned and shown to the user from View().

If there is nothing inside View(), it will display the content of the action method. In this case, the "Index" is the one that represents the home page. However, if we were to specify the name of a specific controller, like View("Privacy"), it would display the content of that page.

```
public IActionResult Index()
{
    return View();
}
```