# ASPNETCORE + ANGULAR ECOMMERCE DOCUMENTATION

Requirements / Software

[.NET CORE SDK](https://dotnet.microsoft.com/download)

[Postman](https://www.postman.com/downloads/)

[VS Code](https://code.visualstudio.com/download)

[Git](https://git-scm.com/downloads)

Creating a Solution

$ ls

$ mkdir ecommerce

$ cd ecommerce

$ dotnet –help

$ dotnet new –h

$ dotnet new sln

$ dotnet new webapi –o API

$ dotnet sln –h

$ dotnet sln add API

$ dotnet sln list

$ code .

Managing the SSL status

1. In launchSettings.json :

“launchBrowser” : false

1. Commands to know/do
   1. $ dotnet run --------To run-------
   2. $ dotnet watch run --------To run and watch on save------
   3. $ dotnet dev-certs https
   4. $ dotnet dev-certs https –t

Adding Own Controller

Extension to add: C# Extensions by jchannon

* Right click on Controller: New C# Class> **ProductsController.cs**

**ProductsController.cs**

namespace API.Controllers

{

[ApiController]

[Route("api/[controller]")]

public class ProductsController : ControllerBase

{

[HttpGet]

public string GetProducts()

{

return “Get json of all products”;

}

[HttpGet("{id}")]

public string GetProduct(int id)

{

return “Get single producs”;

}

}

}

Create a Entity Class

* Create Folder “**Entities**”
* Create class “**Product.cs**”

public class Product

{

public int Id { get; set; }

public string Name { get; set; }

}

Setting an Entity Framework

* Add Extension : **Nuget Package Manager**
* **Ctrl+Shift+P**
* Write nuget and select to add package
* Write Microsoft.EntityFrameworkCore
* Choose Microsoft.EntityFrameworkCore
* Select version: My is 3.1.8
* Choose Microsoft.EntityFrameworkCore.Sqflite

Adding a Connection String

* Open **appSettings.Development.json** and add the line
  + “ConnectionStrings”: {“DefaultConnection”: “Data source=ecommerce.db”}
* Open Settings and search for **private** then in C# extension add “\_”. Then search for this and untick the setting that is there.
* Open startup.cs
  + //////////
  + Generated after Ctrl+. And selected “Initialize field from parameter”
    - private readonly IConfiguration \_config;
  + Replace the old code with this
  + public Startup (IConfiguration config) {
  + \_config = config;
  + }
  + ////////////
  + Services.AddControllers();
  + services.AddDbContext<StoreContext>(x=>x.UseSqlite(\_config.GetConnectionString("DefaultConnection")));

Adding and Entity Framework Migration

* $ dotnet –info
* $ dotnet tool install –global dotnet-ef –version 3.1.402
* $ dotnet ef migrations add InitialCreate –o Data/Migrations
  + This could show error
  + Add package from **nugget package manager** & add **Microsoft.EntityFramework.Design** of version 3.1.8 i.e based on your sdk version or hosting
* $ dotnet ef migrations add InitialCreate –o Data/Migrations

Updating the database

* $ dotnet ef database update
* Extension Required: **Sqflite**
* The file “ecommerce.db” should be generated
* To open that file:
  + **Ctrl+Shift+P**
    - **sqlite: Open Database**
    - Choose **ecommerce.db**
* You can find SQFLITE in the explorer area.
* Expand to find the table
* In Product table, Right click and Select **New Query(Insert)**
  + **Write & Select all the below query and run query by right click.**

--SQlite

INSERT INTO ‘Products’ (Id, Name)

VALUES (1, “Apple”), (2,”Banana”),(3,”Mango”);

Reading Data from the database

* **ProductsController.cs**
* **Replace code and do necessary imports**

[HttpGet]

public async Task<ActionResult<List<Product>>> GetProducts()

{

var products = await \_context.Products.ToListAsync();

return Ok(products);

}

[HttpGet("{id}")]

public async Task<ActionResult<Product>> GetProduct(int id)

{

return await \_context.Products.FindAsync(id);

}

PostMan Collection

{{url}}/api/products

Creating the additional projects

* API $ cd ..
* ecommerce $ dotnet new classlib –o Core
* ecommerce $ dotnet new classlib –o Infrastructure
* $ dotnet sln add Core/
* $ dotnet sln add Infrastructure/
* $cd API
* API $ dotnet add reference ../Infrastructure
* $ cd ..
* $ cd Infrastructure
* Infrastructure $ dotnet add reference ../Core
* $ cd ..
* $ dotnet restore
* Then Move the folder from-to
  + Entities--- > Core
  + Data ----- > Infrastructure
* The fix the namespaces and the using statements
* $ dotnet restore
* $ dotnet build

Hiding obj and bin folder

* Open Settings
* Search **exclude**
  + In **Add** **Pattern** add the following
  + \*\*/obj
  + \*\*/bin

Saving to Git

* $ git status
* $ git init
* Create file “.gitignore”
  + obj
  + bin
  + appSettings.json
  + .db
* $ git add
* $ git commit –m “Initial Commit”
* Create a new repo as <github.com/new>
* $ git remote add origin <https://_____________>
* $ git push –u origin master