**Online Book Reselling**

SOLUTION

# Tools and Technologies

1. .NET Core 5.0
2. Angular
3. Visual Studio 2019, Visual Studio Code
4. SQL Server

**Data (Infrastructure)**

# Architecture

Repository

Unit of Work

EF DB Context

**.Net Core API**

Controllers

Angular

SPA

**Service**

DB SQL

**Core**

Entities & Intefaces

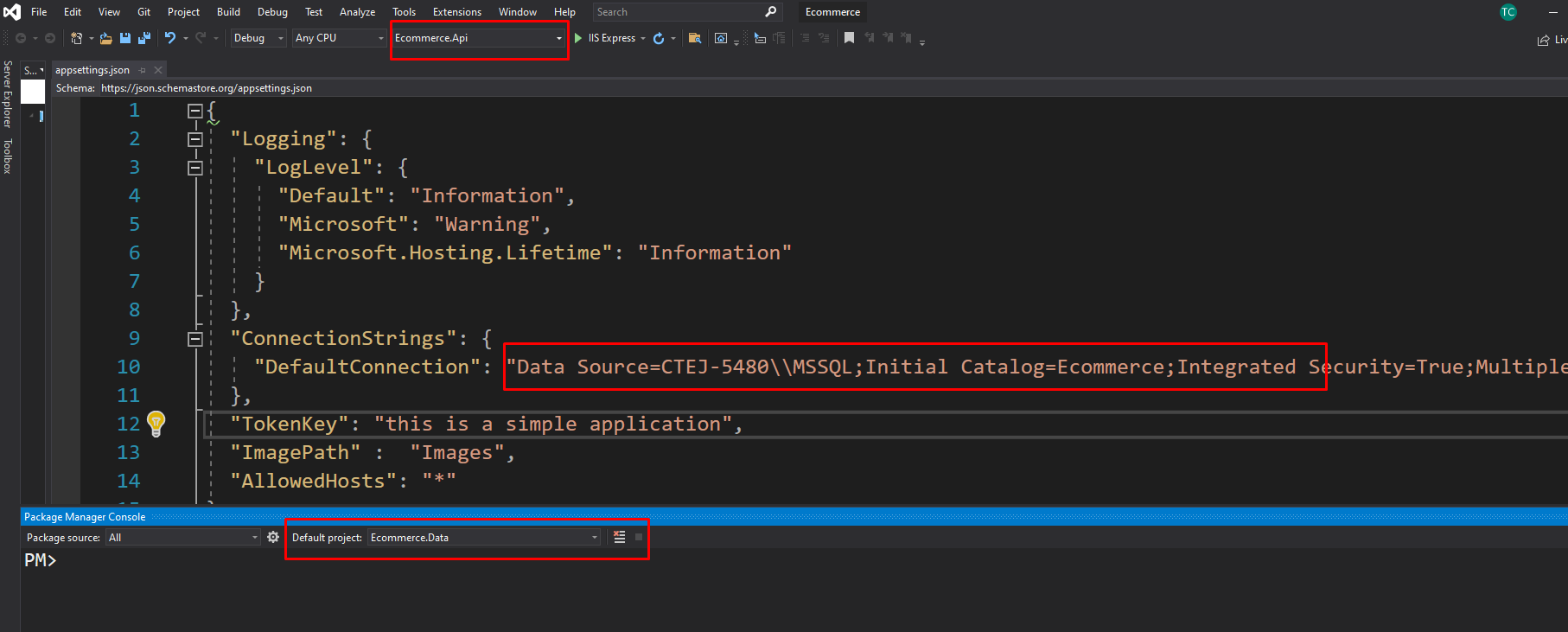
# Setting up the code

1. Clone the repository <https://github.com/charantejg/Ecommerce> on your workstation
2. Open the solution in Visual Studio and once all the Nuget dependencies are restored open the appsettings.json present in Ecommerce.Api project
3. Change the connection string “DefaultConnection” to your SQL server, a new database name can be provided.
4. Open Package Manager Console (PMC nugget) terminal and add the migration,. Ensure the startup project is set to Ecommerce.Api and default project in NPM window should be set to Ecommerce.Data
5. Execute commands

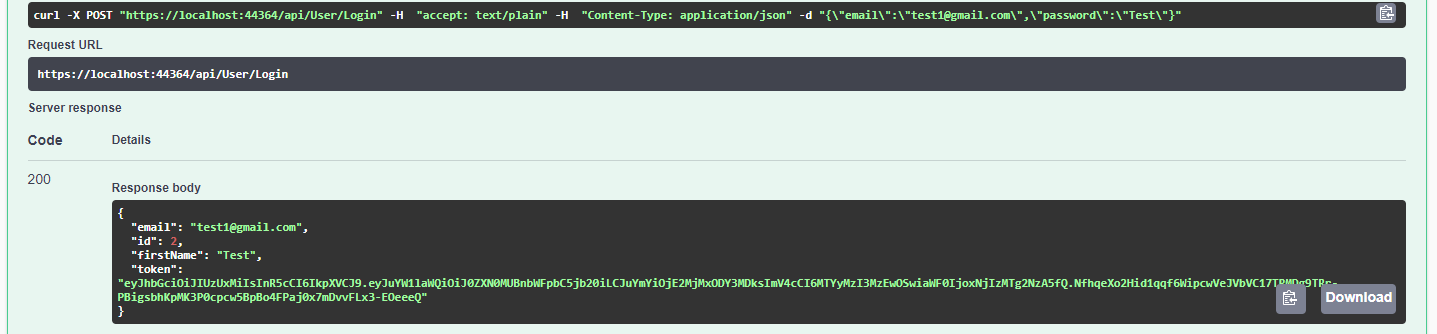
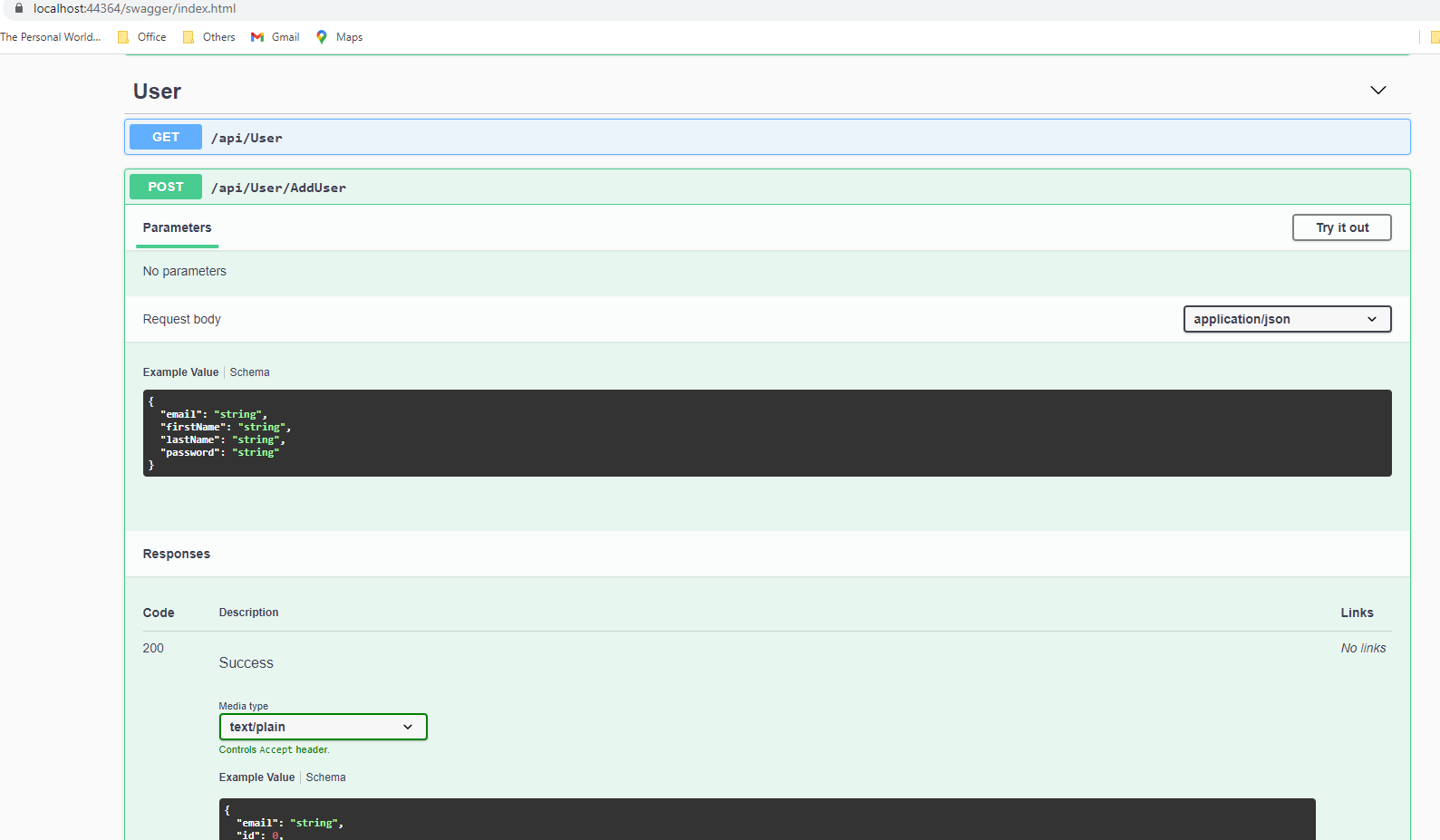
Add-migration initialmigration

Update-database

Note: If any migration error related to decimal conversion can be ignored.



1. Run the Ecommerce.Api project in debug mode from Visual studio. The application has a swagger API page loaded by default
2. It is required to create users from API to login to the application.
3. Go to the /api/User/AddUser in the swagger and seed the data manually.



In the body enter the email Id and password, in the place of “string” and execute.

{

"email": "string",

"firstName": "string",

"lastName": "string",

"password": "string"

}

Sample Data:

{

"email": "test1@gmail.com",

"firstName": "Test",

"lastName": "Last Name",

"password": "Test"

}

Once the 200 result is in place, you have successfully created user.

Note: Since this was a coding round exercise I have not created User Creation pages. Its just a hacky way creating user from API call .

Note: I rather using a **hardcoded user table** with plain password, implemented a custom logic to hash and salt the password. Also implemented JWT token for authorizing subsequent API calls.

1. **Seeding the Books metadata**: As this was a time based excerise I have not created UI admin modules to upload data or have not written any json data seed using the Dbcontext in c# class.

Please open the SQL management studio, connect to the the database created by Code First approach, after the migration.

Run these sql statements.

INSERT INTO [dbo].[Categories] ([CategoryName]) VALUES ('Default')

INSERT INTO [dbo].[Books] ([BookTitle] ,[CategoryId] ,[ThumbnailPath],[Price],[Description]) VALUES ('Java Programmnig Volume 1 ', 1, 'Images\java.png', 250.00, 'Java Book OOP concepts for Engineering Sem 2')

INSERT INTO [dbo].[Books] ([BookTitle] ,[CategoryId] ,[ThumbnailPath],[Price],[Description]) VALUES ('C# Programming Language book', 1, 'Images\csharp.png', 300.00, 'C# Programmming for Dummies')

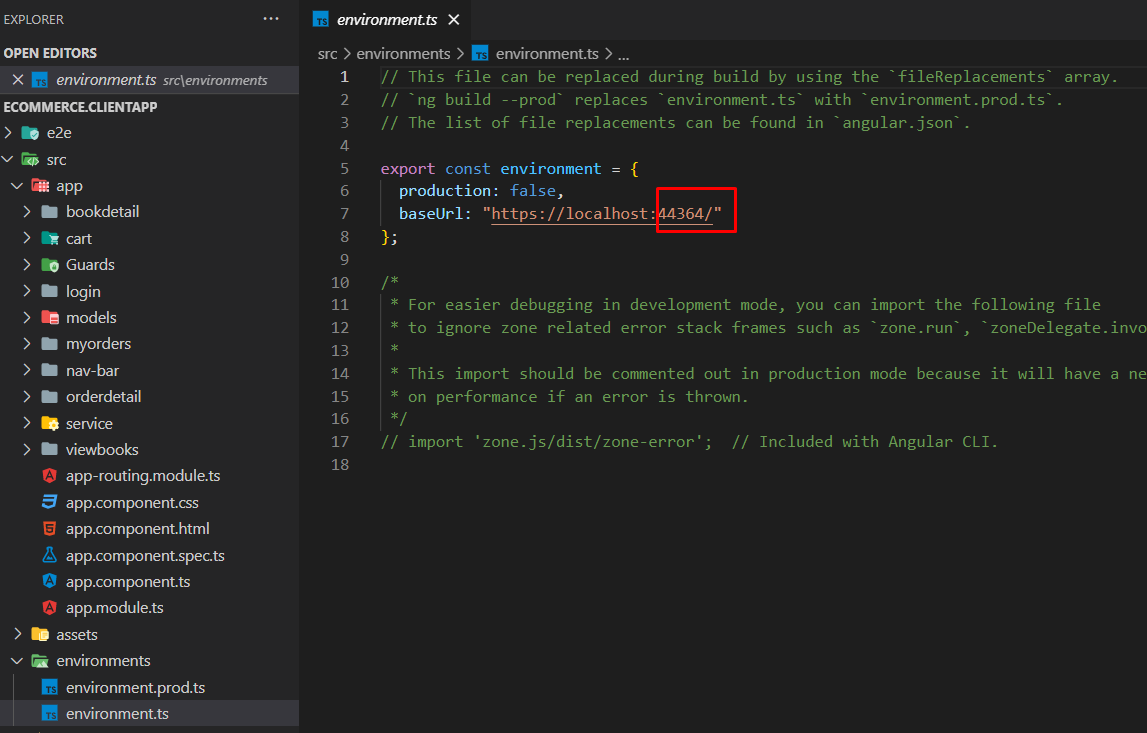
INSERT INTO [dbo].[Books] ([BookTitle] ,[CategoryId] ,[ThumbnailPath],[Price],[Description]) VALUES ('Python Learning guide', 1, 'Images\python.png', 150.00, 'Python is an interpreted, object-oriented, high-level programming language with dynamic semantics')

INSERT INTO [dbo].[Books] ([BookTitle] ,[CategoryId] ,[ThumbnailPath],[Price],[Description]) VALUES ('Machine Language', 1, 'Images\ml.png', 500.00, 'Learn Machine language in 24 hours')

INSERT INTO [dbo].[Books] ([BookTitle] ,[CategoryId] ,[ThumbnailPath],[Price],[Description]) VALUES ('Computer Graphics', 1, 'Images\cg.png', 350.00, 'Lorem Lepsom')

INSERT INTO [dbo].[Books] ([BookTitle] ,[CategoryId] ,[ThumbnailPath],[Price],[Description]) VALUES ('Azure Services', 1, 'Images\Ad.png', 450.00, 'Lorem Lepsom')

1. Open the folder Ecommerce.ClientApp in Visual studio code. Open environment.ts file and ensure the baseUrl is pointed to the same port of the web API application



1. After the npm install (downloaing all dependencies) , perform ng build / ng server and open the default Url - localhost:4200
2. Login with the Email and Password you have created with help of Swagger / Postman API console.
3. Angular Application - Screens

* Login – Only logged in user can access the system.
* View Books – Book Gallery from DB
  + Actions – Add to Cart, View Book Details
* Cart – Displays customer added books in order
  + Quantity can be modified and reflects the sub total
  + Place Order – Creates a new Order
* Order – Once order is placed it can be viewed in My Orders
  + Order Details can be fetched by clicking on an individual order