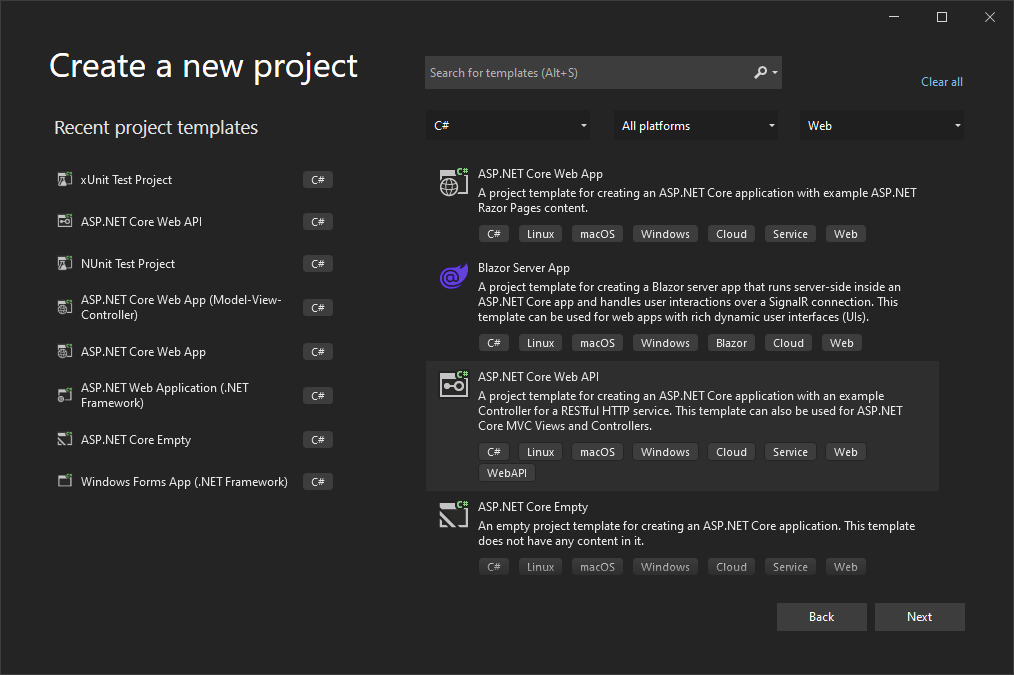
– FINAL PROJECT –

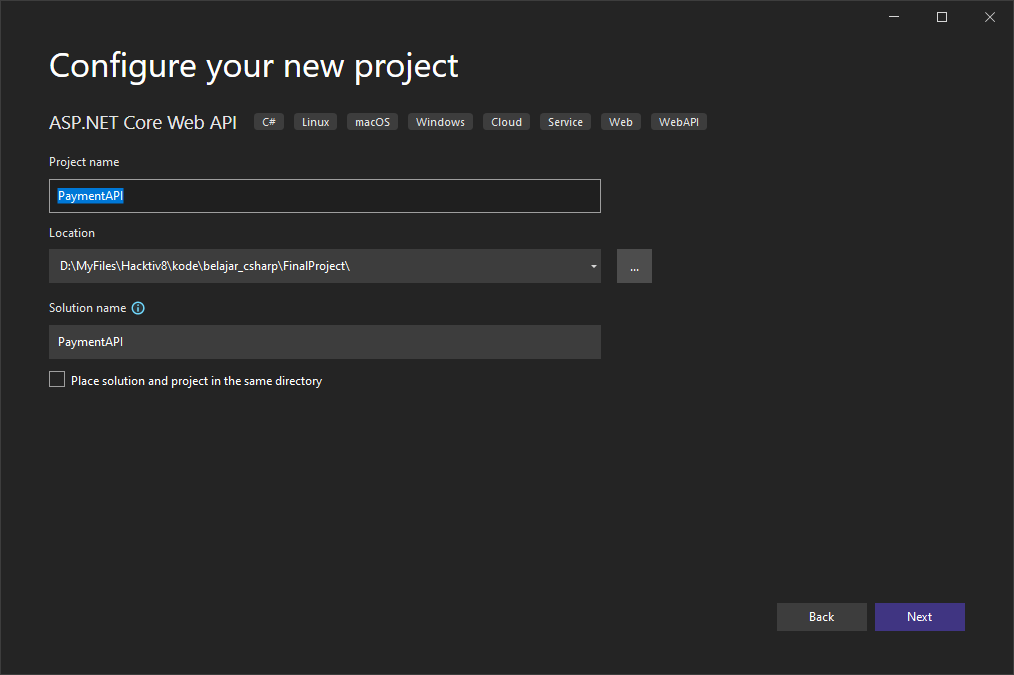
REST API: Online Debit/Credit Card

# Membuat Project Baru Melalui Visual Studio

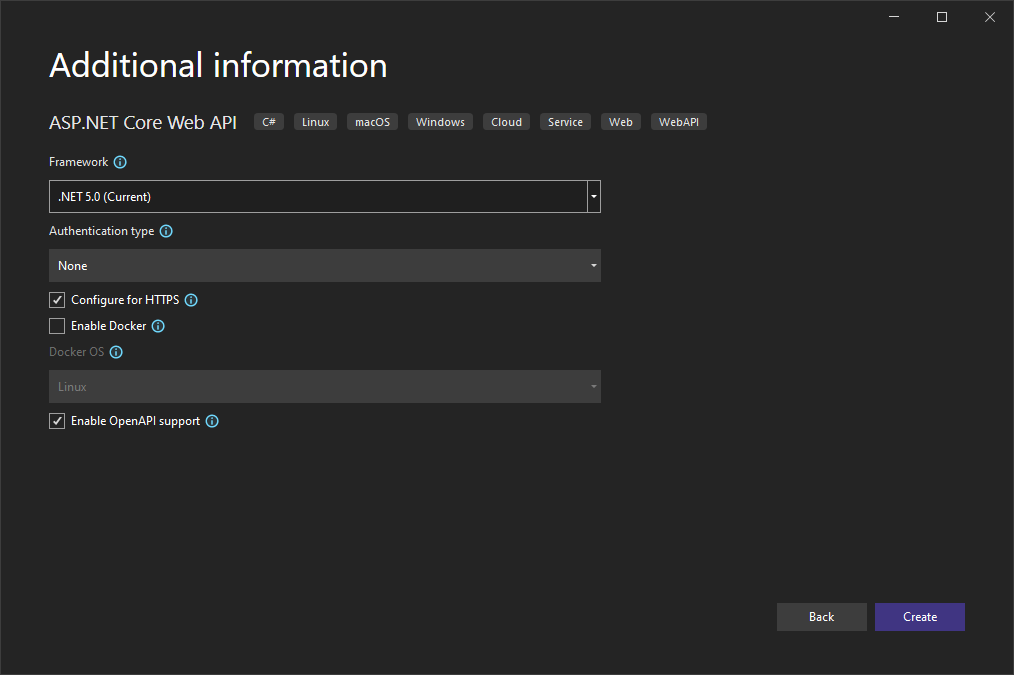
1. Create a New Project > Next



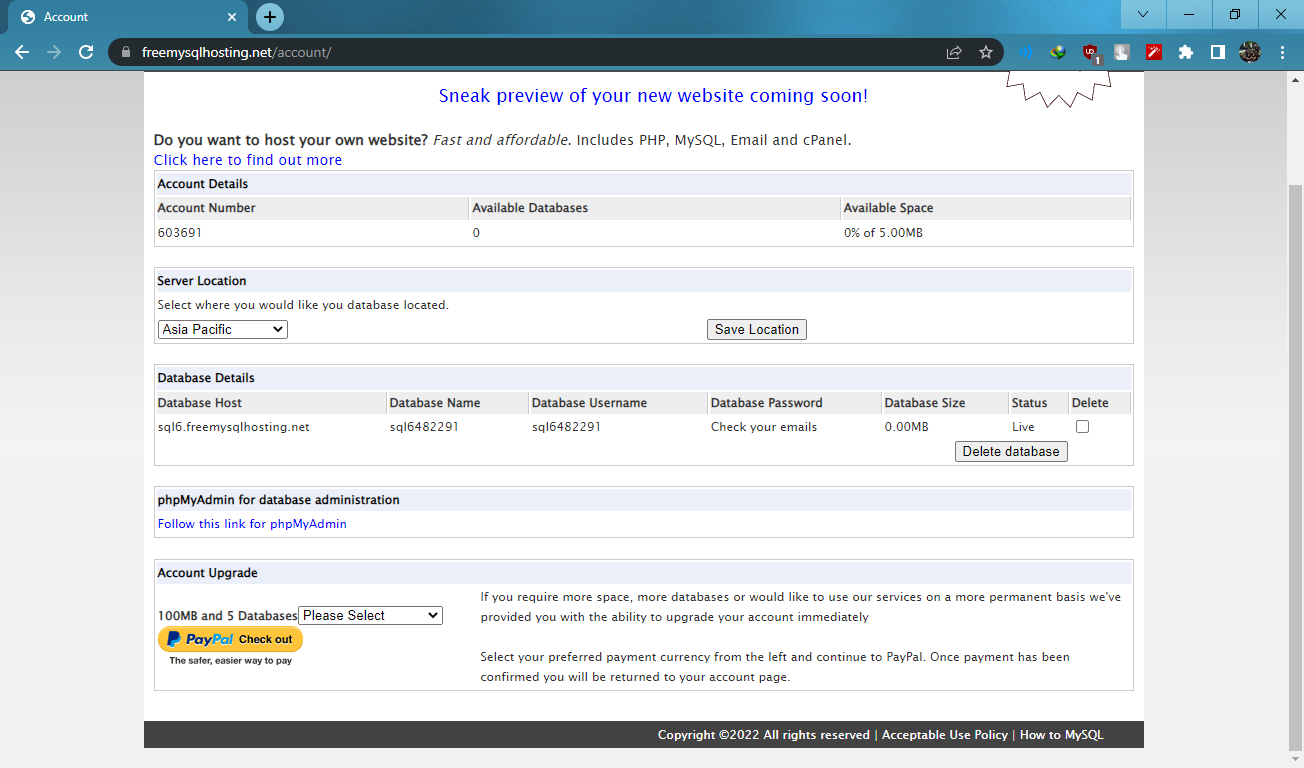
1. Mengisi kolom Project Name menjadi ‘PaymentAPI’ > Pada Kolom Lokasi arahkan pada folder FinalProject > Next



1. Ikuti Konfigurasi > Create

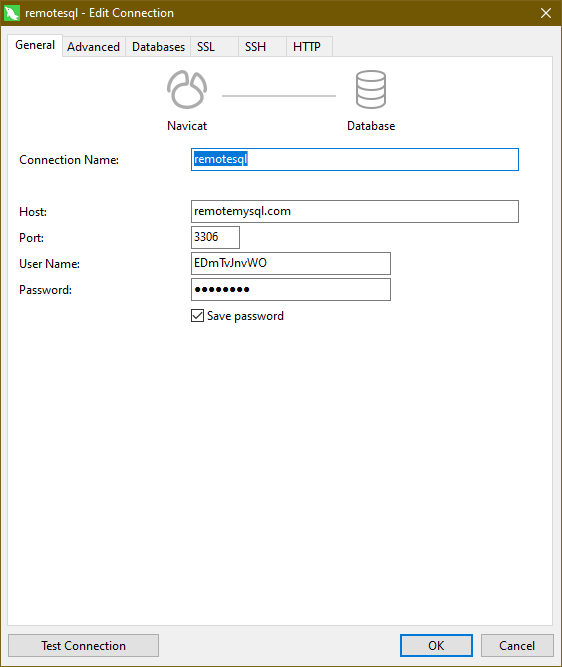


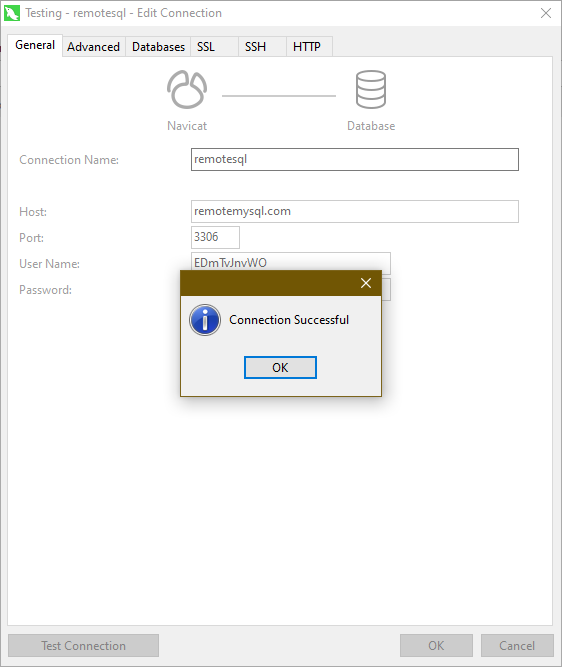
# Membuat Database di Hosting

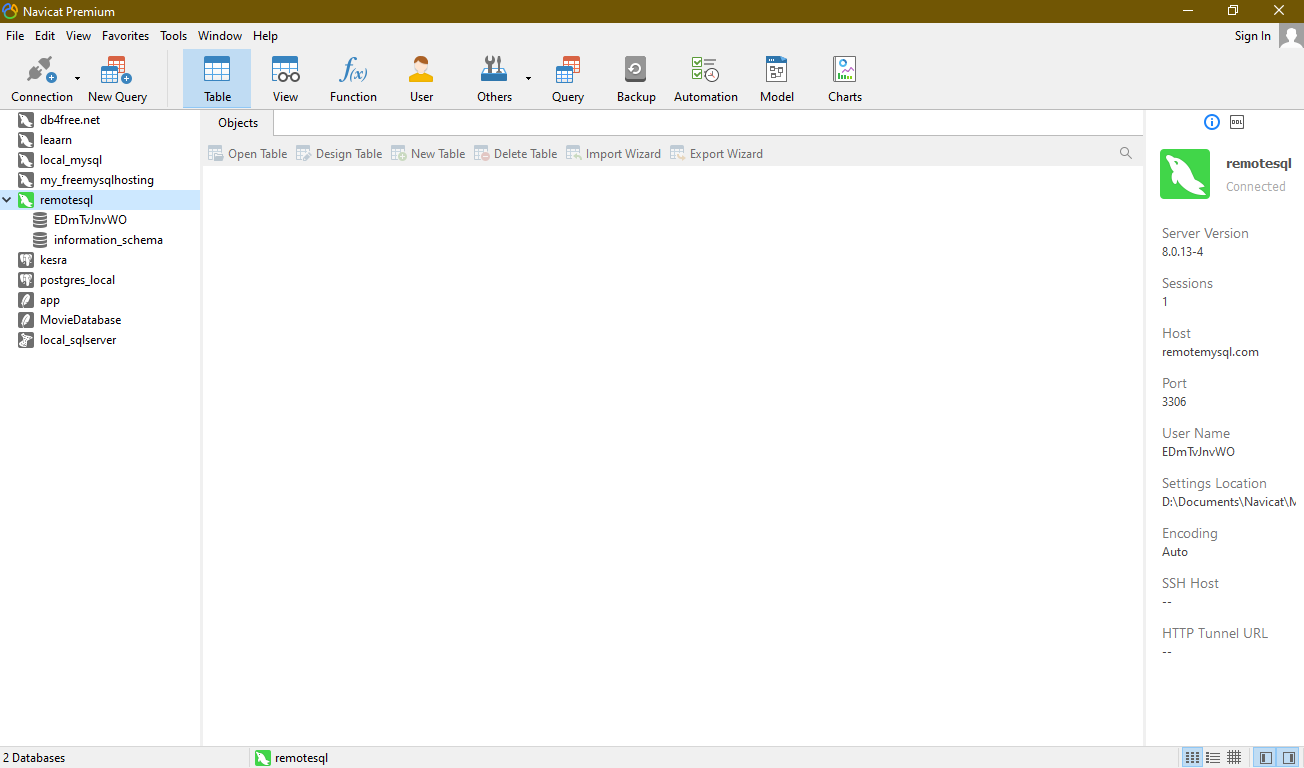


**Server:** [sql6.freemysqlhosting.net](http://sql6.freemysqlhosting.net/" \t "_blank)  
**Name:** sql6482291  
**Username:** sql6482291  
**Password:** \*\*\*\*\*\*\*\*\*  
**Port number:** 3306

**REMOTE DATABASE VIA NAVICAT**



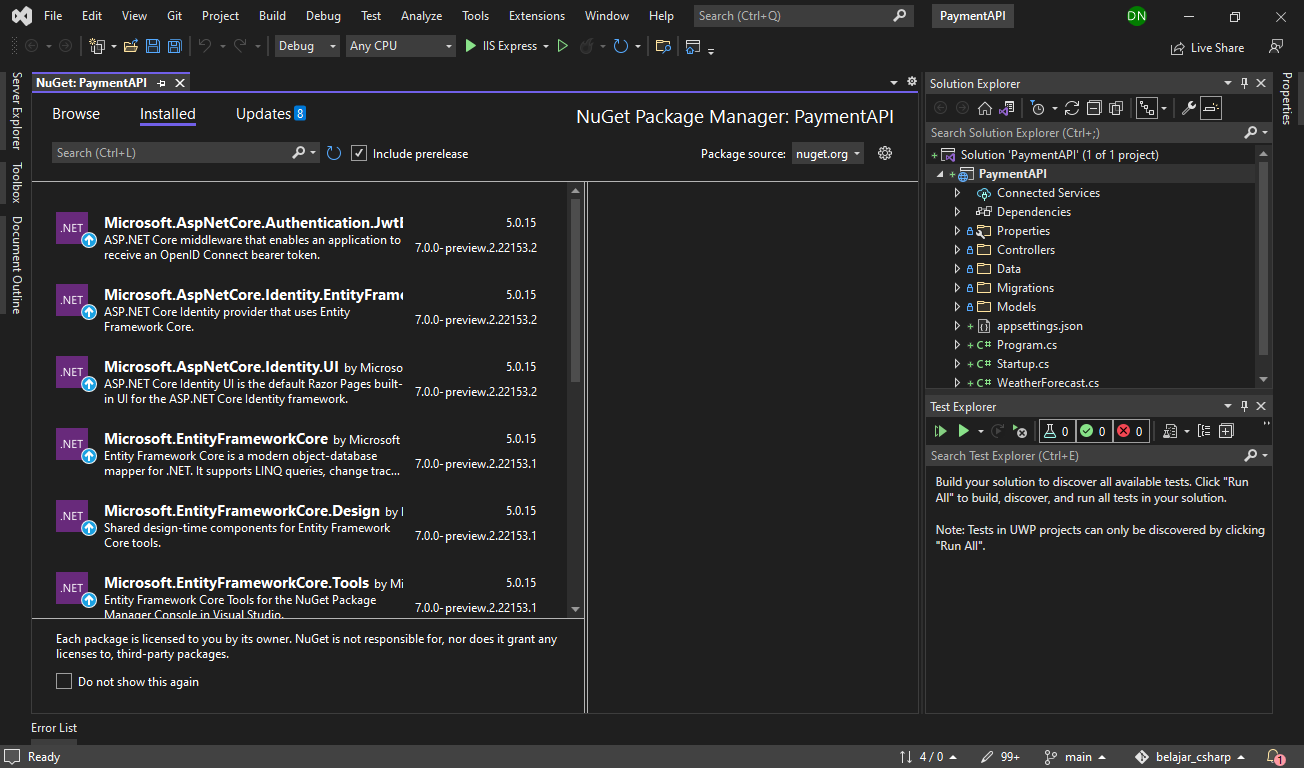


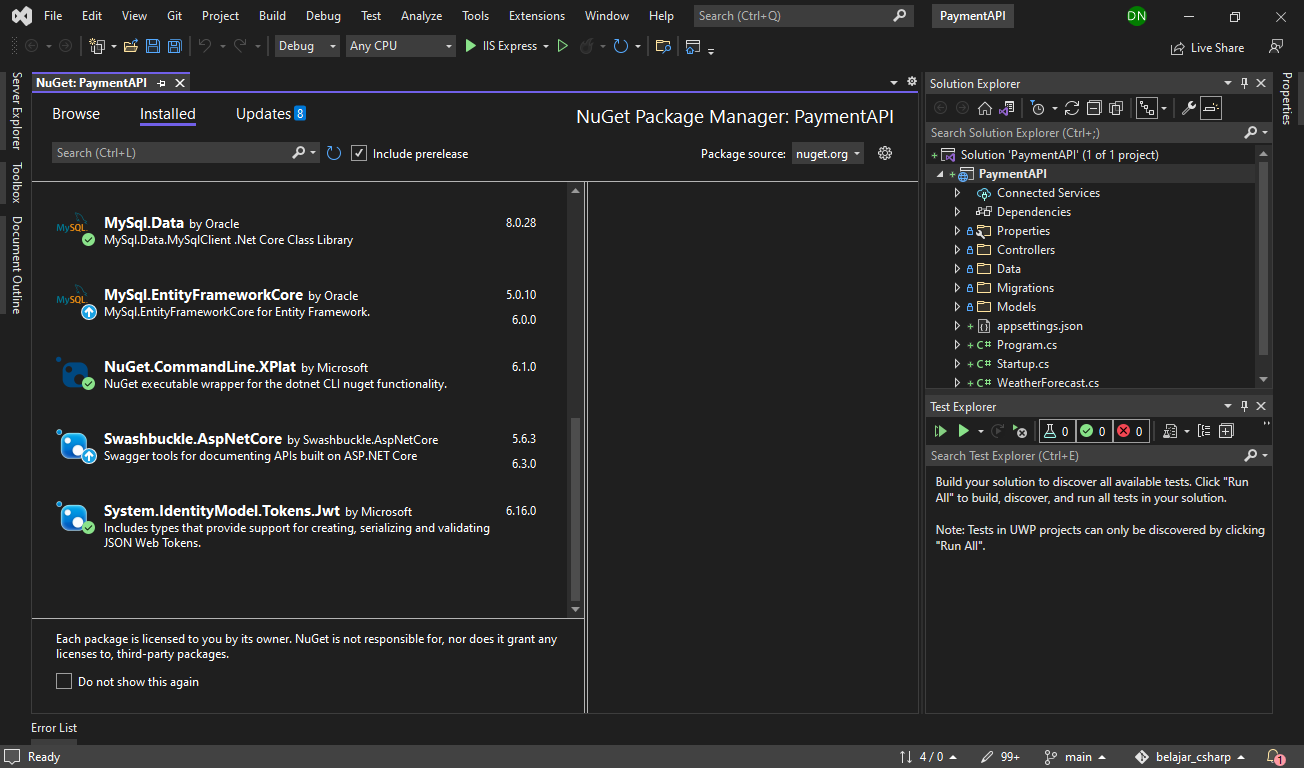


# Install Package via Nugget Package Manager Visual Studio

List package yang perlu diinstall:

|  |  |  |
| --- | --- | --- |
| Nama Paket | Versi | Deskripsi |
| Microsoft.AspNetCore.Authentication.JwtBearer | 5.0.15 |  |
| Microsoft.AspNetCore.Identity.EntityFrameworkCore | 5.0.15 |  |
| Microsoft.AspNetCore.Identity.UI | 5.0.15 |  |
| Microsoft.EntityFrameworkCore | 5.0.15 |  |
| Microsoft.EntityFrameworkCore.Design | 5.0.15 |  |
| Microsoft.EntityFrameworkCore.Tools | 5.0.15 |  |
| MySql.Data | 8.0.28 |  |
| MySql.EntityFrameworkCore | 5.0.10 |  |
| NuGet.CommandLine.XPlat | 6.1.0 |  |
| Swashbuckle.AspNetCore | 5.6.3 | for auto generating your swagger definition |
| System.IdentityModel.Tokens.Jwt | 6.16.0 |  |





# Pembuatan Kelas/Object Payment dan ApiDbContext

1. Membuat Folder Untuk Models
2. Pada ‘PaymentAPI/Models’ menambahkan file PaymentItem.cs sebagai kelas/object data Payment.

------------------------------------------------------------------------------------------

‘PaymentAPI/Models/PaymentItem.cs’

------------------------------------------------------------------------------------------

using System;

using System.ComponentModel.DataAnnotations;

namespace PaymentAPI.Models

{

    public class PaymentItem{

        [Key]

        public int paymentDetailId {get; set;}

        public string cardOwnerName {get; set;}

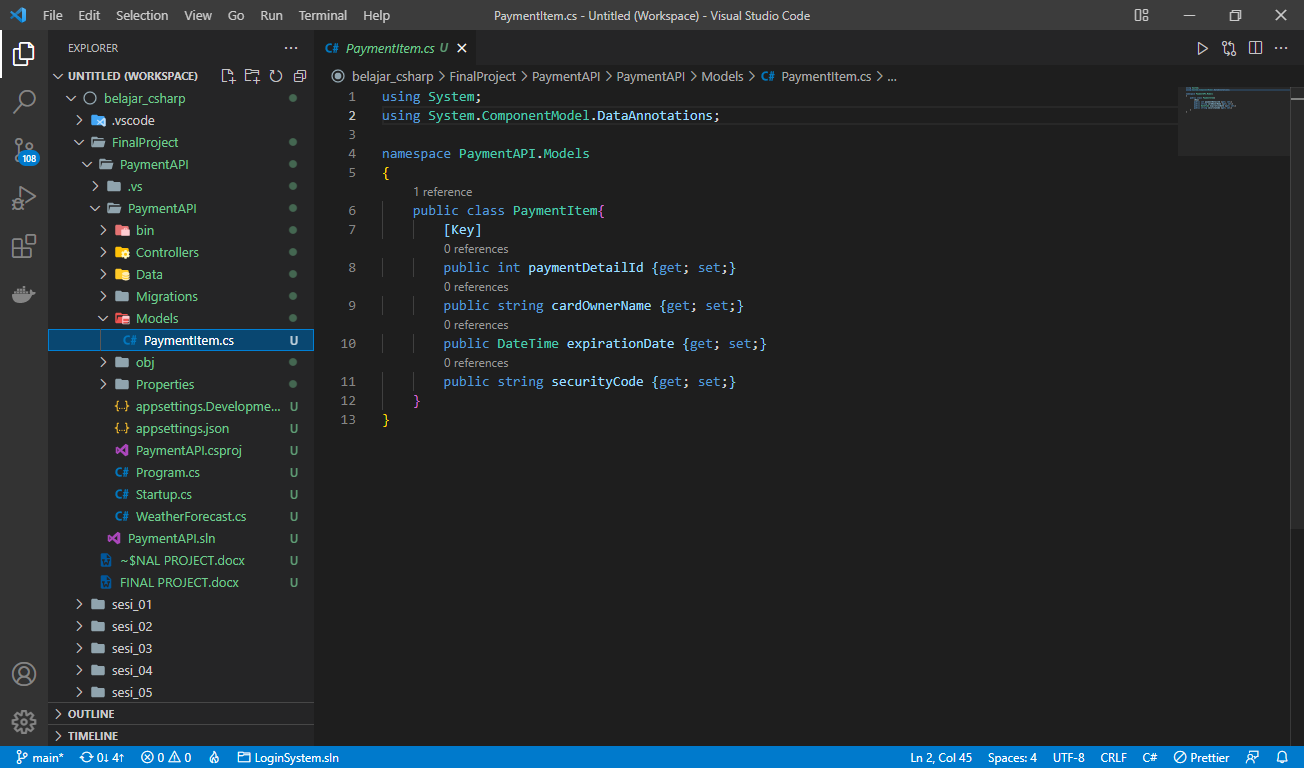
        public DateTime expirationDate {get; set;}

        public string securityCode {get; set;}

    }

}

------------------------------------------------------------------------------------------



1. Membuat Folder Data
2. Pada ‘PaymentAPI/Data’ menambahkan file ApiDbContext sebagai tempat pendefinisian Tabel PaymentDetails.

------------------------------------------------------------------------------------------

‘PaymentAPI/Data/ApiDbContext.cs’

------------------------------------------------------------------------------------------

using Microsoft.EntityFrameworkCore;

using PaymentAPI.Models;

using Microsoft.AspNetCore.Identity.EntityFrameworkCore;

using Microsoft.AspNetCore.Identity;

namespace PaymentAPI.Data

{

    public class ApiDbContext : IdentityDbContext

    {

        public virtual DbSet<PaymentItem> PaymentDetails {get; set;}

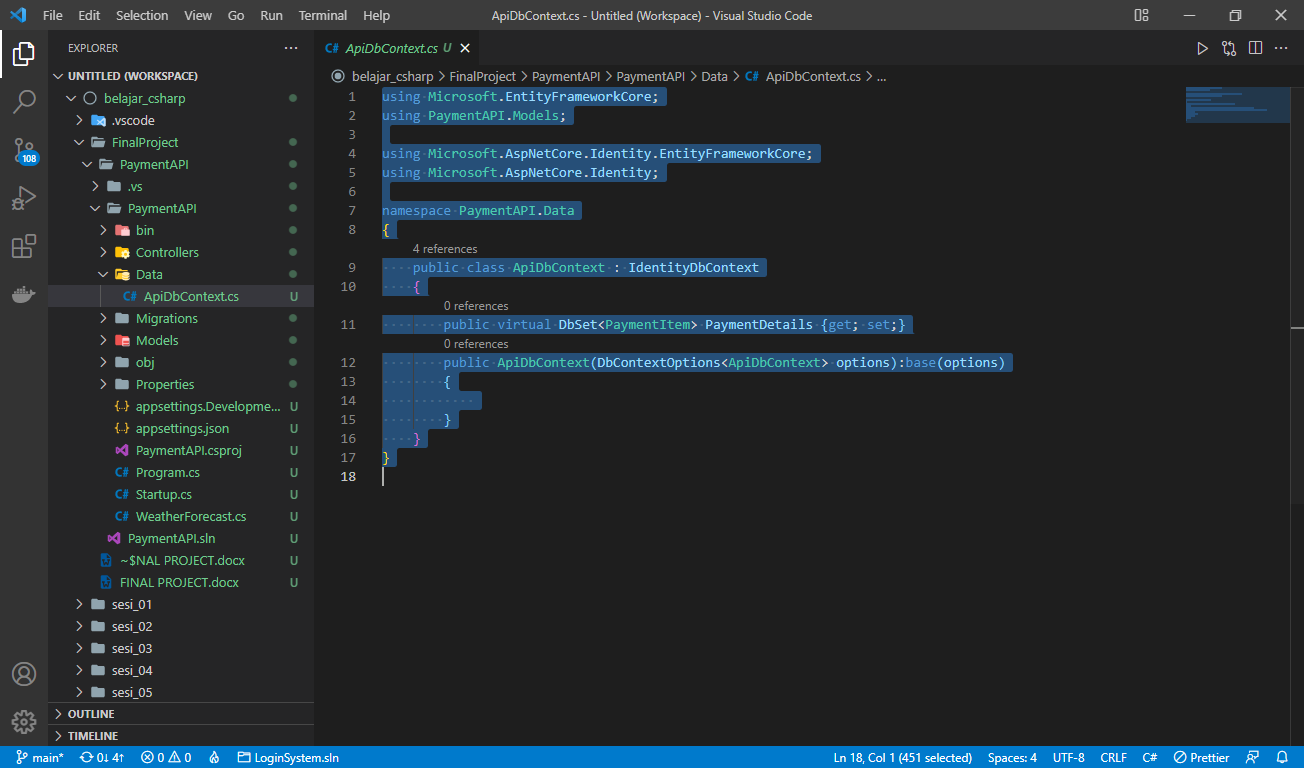
        public ApiDbContext(DbContextOptions<ApiDbContext> options):base(options)

        {

        }

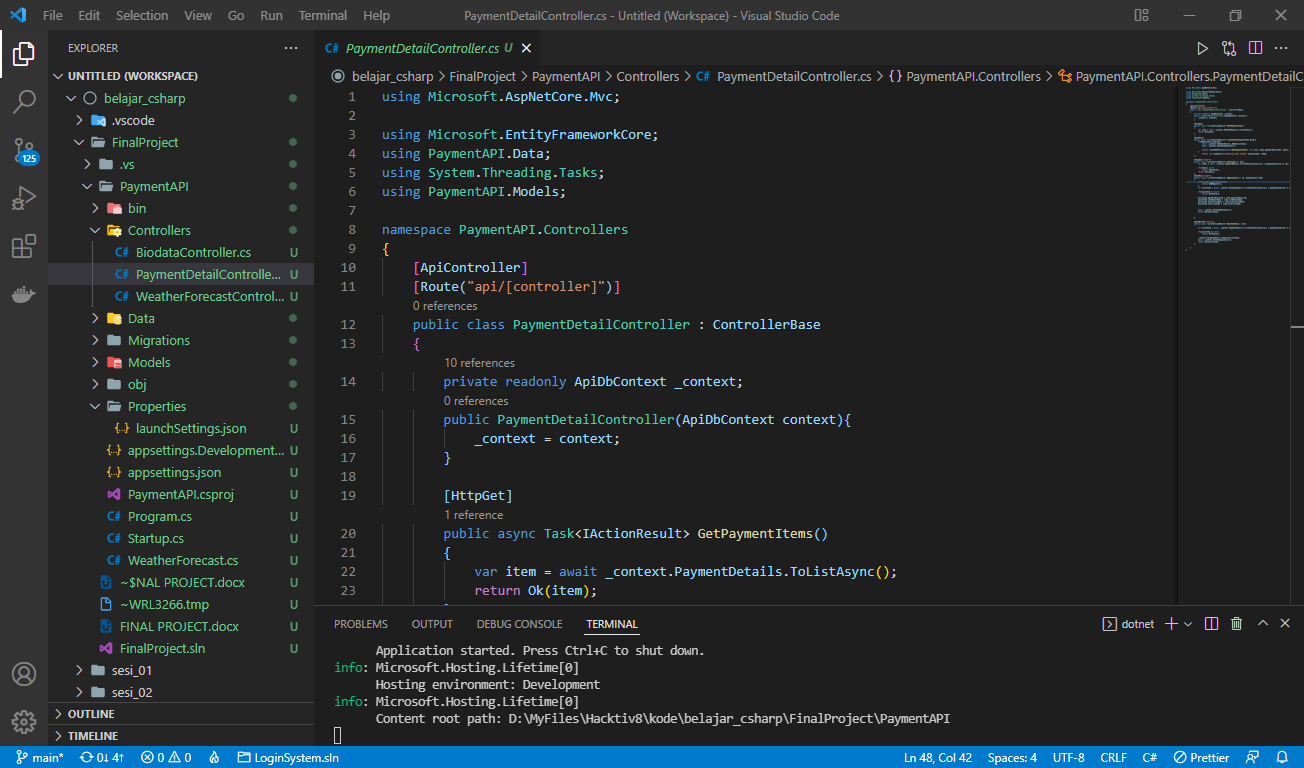
    }

}



# Pembuatan Controller CRUD Payment

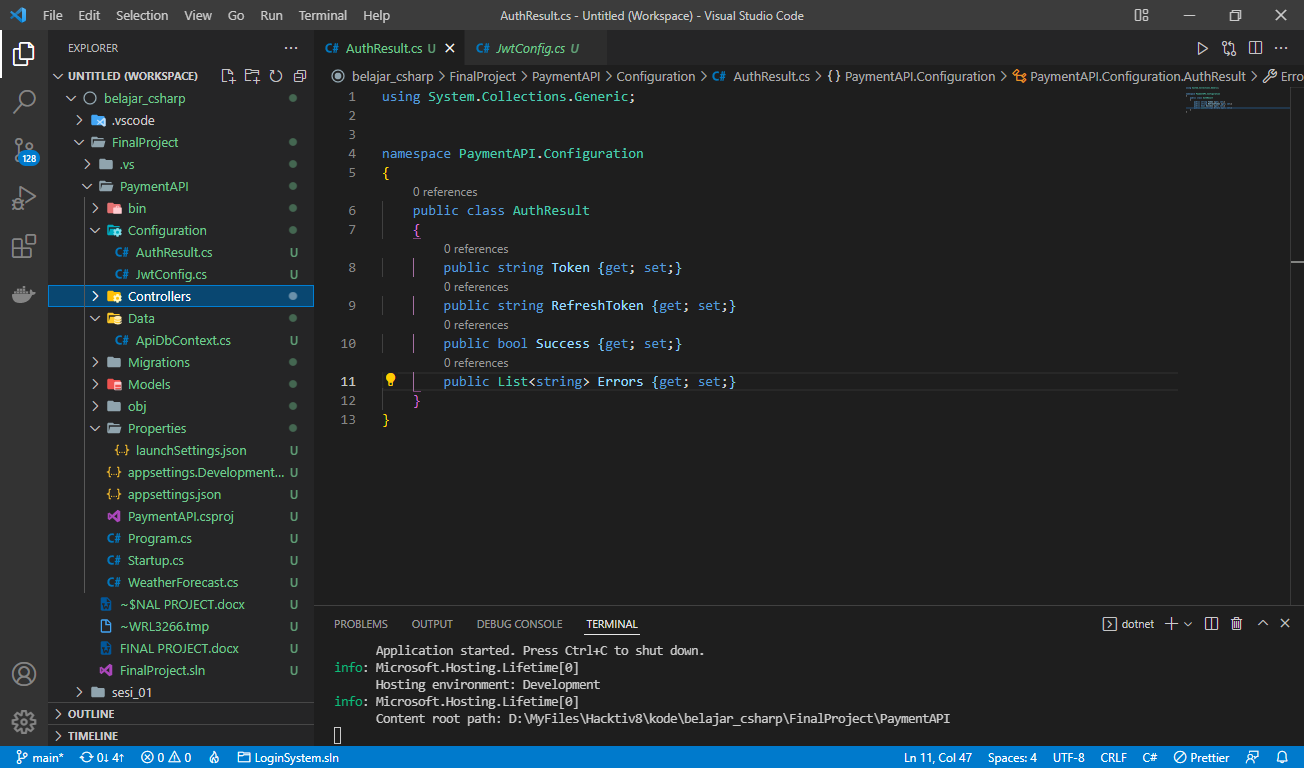
1. Pada ‘PaymentAPI/Controllers’, menambahkan file ‘PaymentDetailController.cs’



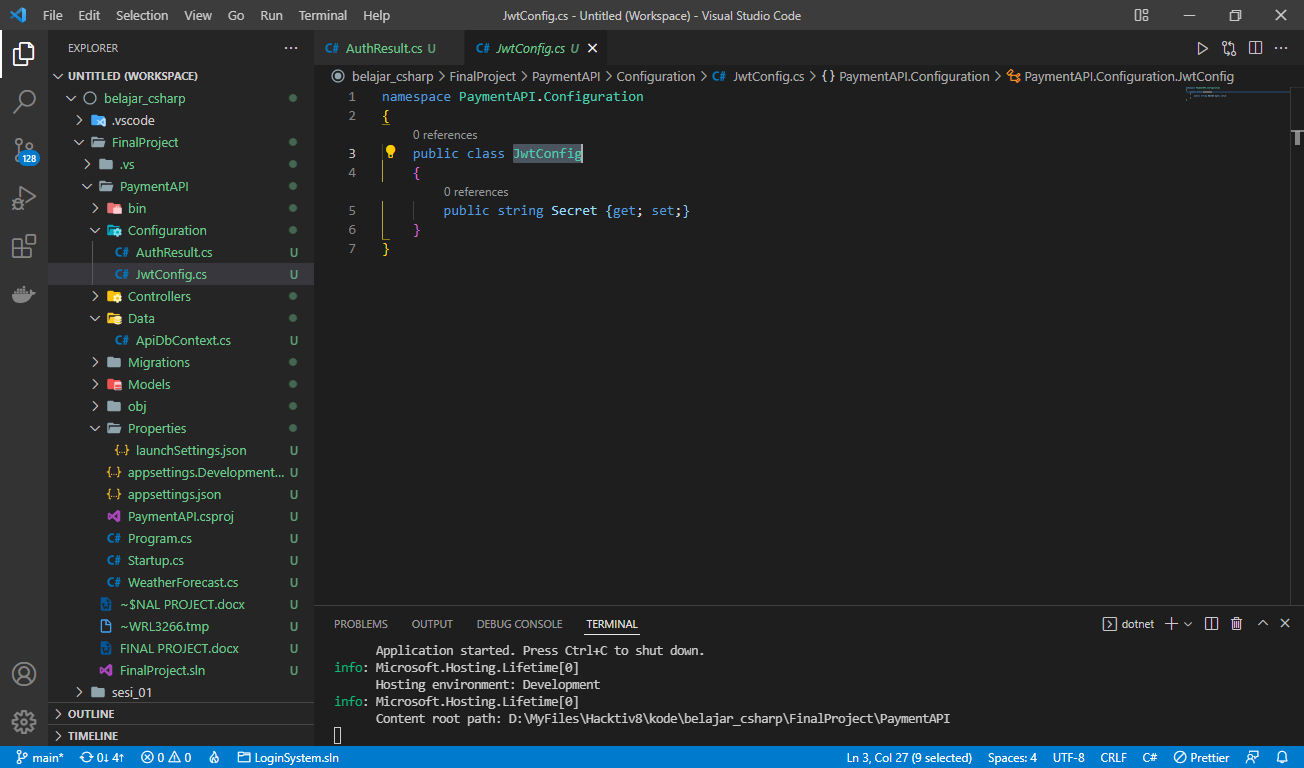
# Penambahan JWT Token dan RefreshToken

1. Pada ‘PaymentAPI’, menambahkan Folder ‘Configuration’
2. Pada ‘PaymentAPI/Configuration’ menambahkan file

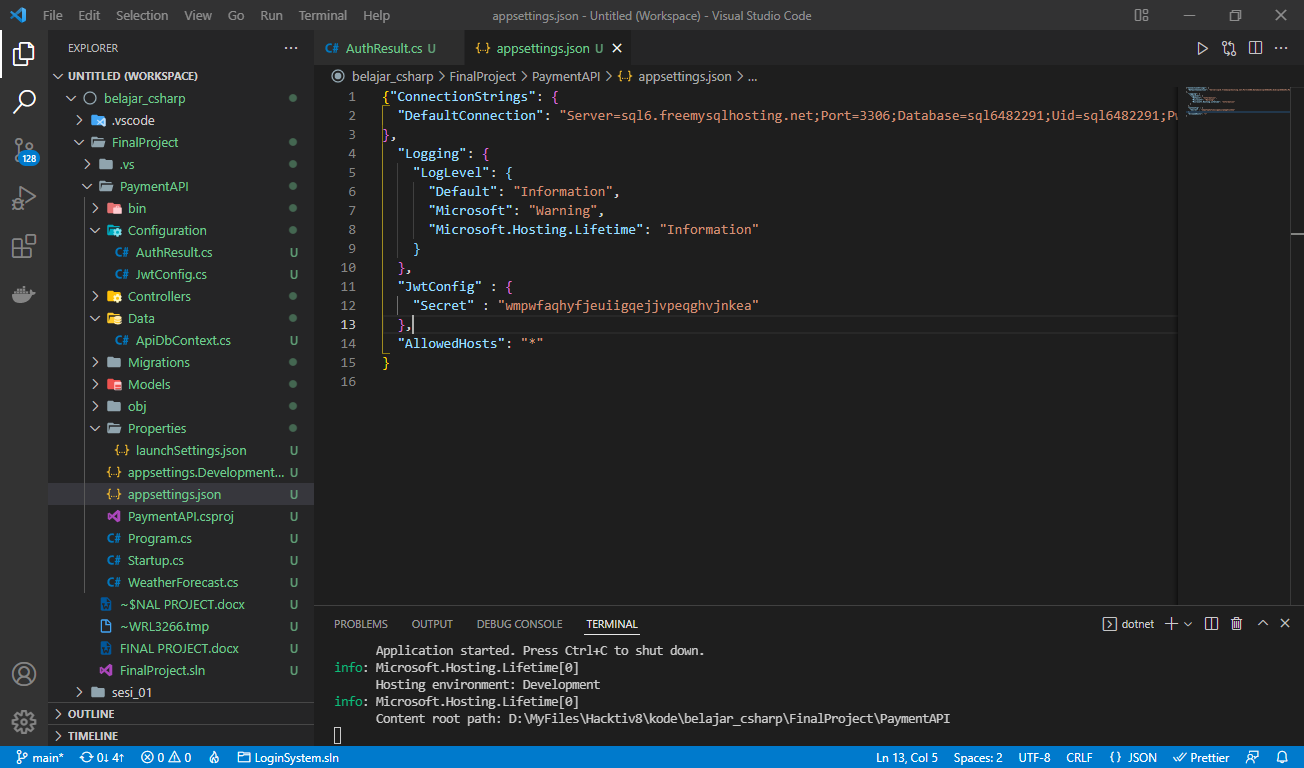
* ‘PaymentAPI/Configuration/AuthResult.cs’



* ‘PaymernAPI/Configuration/JwtConfig.cs’

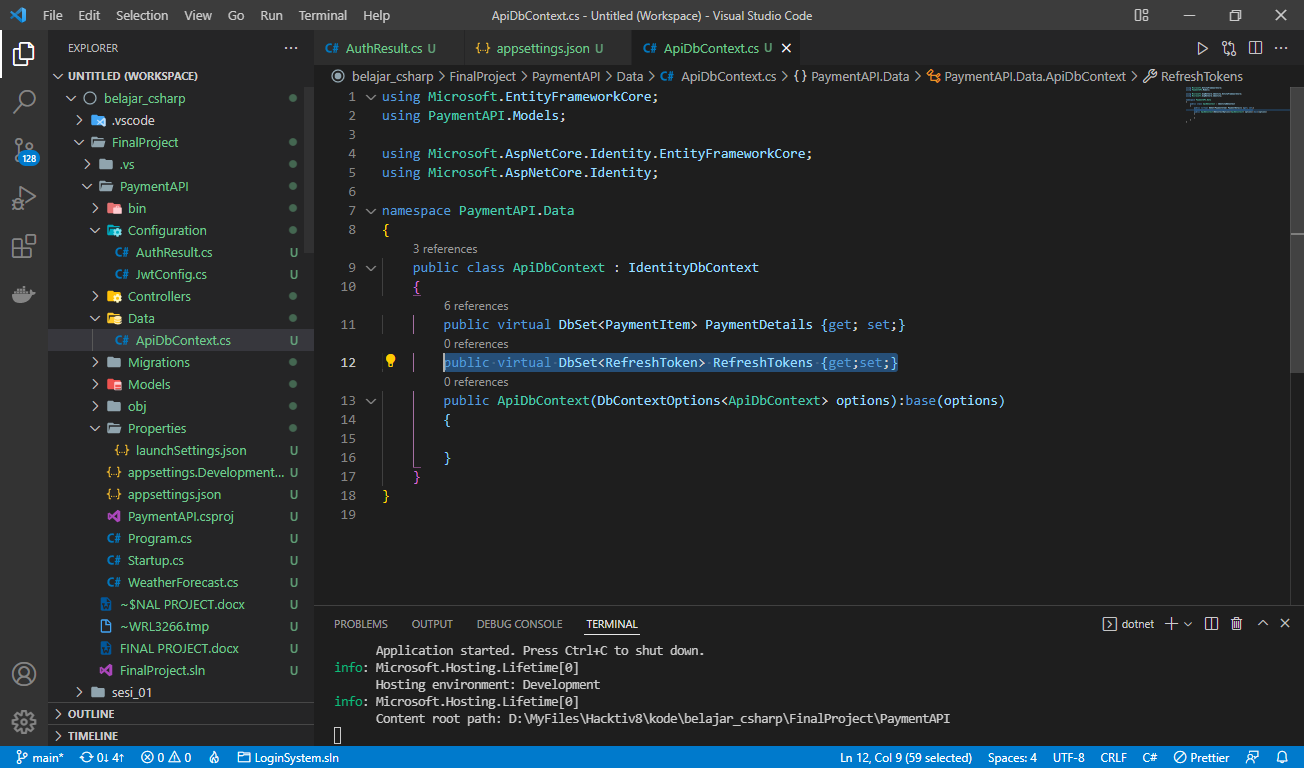


1. Menambahkan JwtConfig.Secrets pada ‘PaymentAPI/appsetting.json’



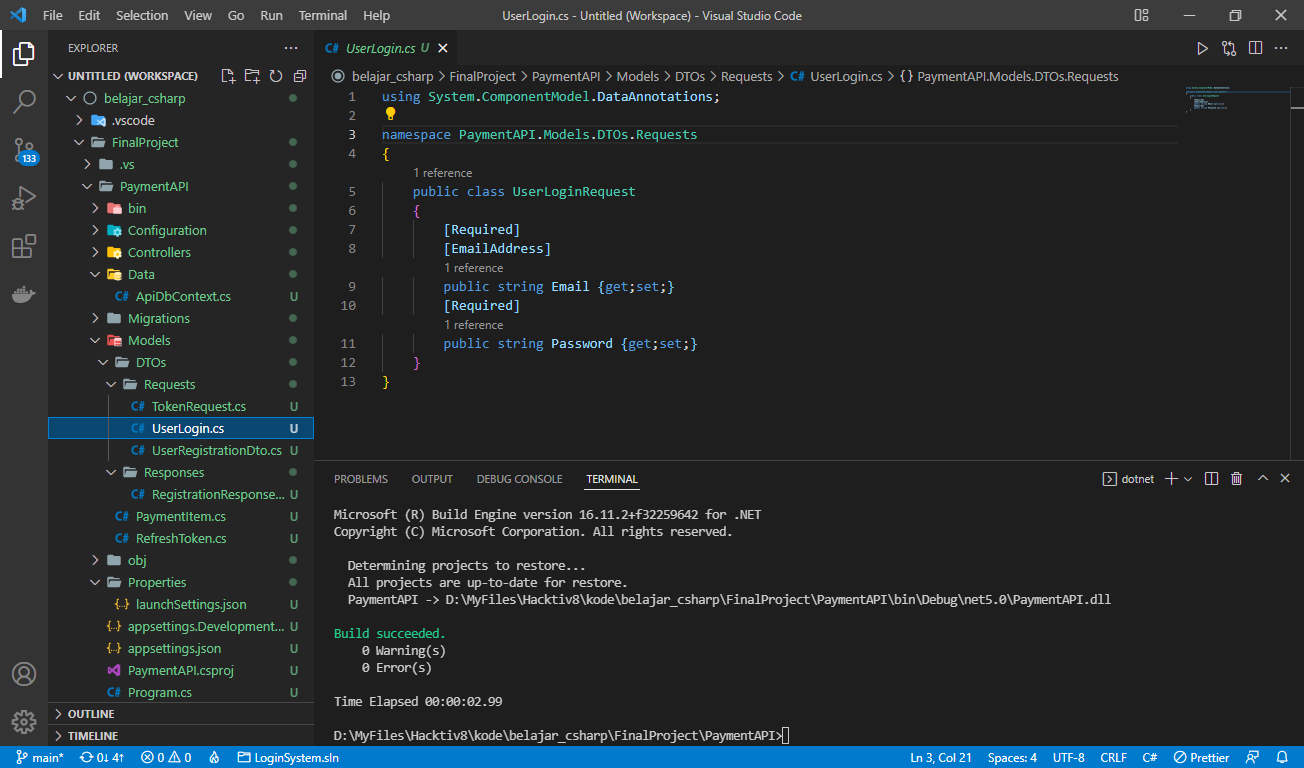
1. Menambahkan DbSet untuk Tabel RefreshToken pada ‘PaymentAPI/Data/ApiDbContext.cs’

public virtual DbSet<RefreshToken> RefreshTokens {get;set;}

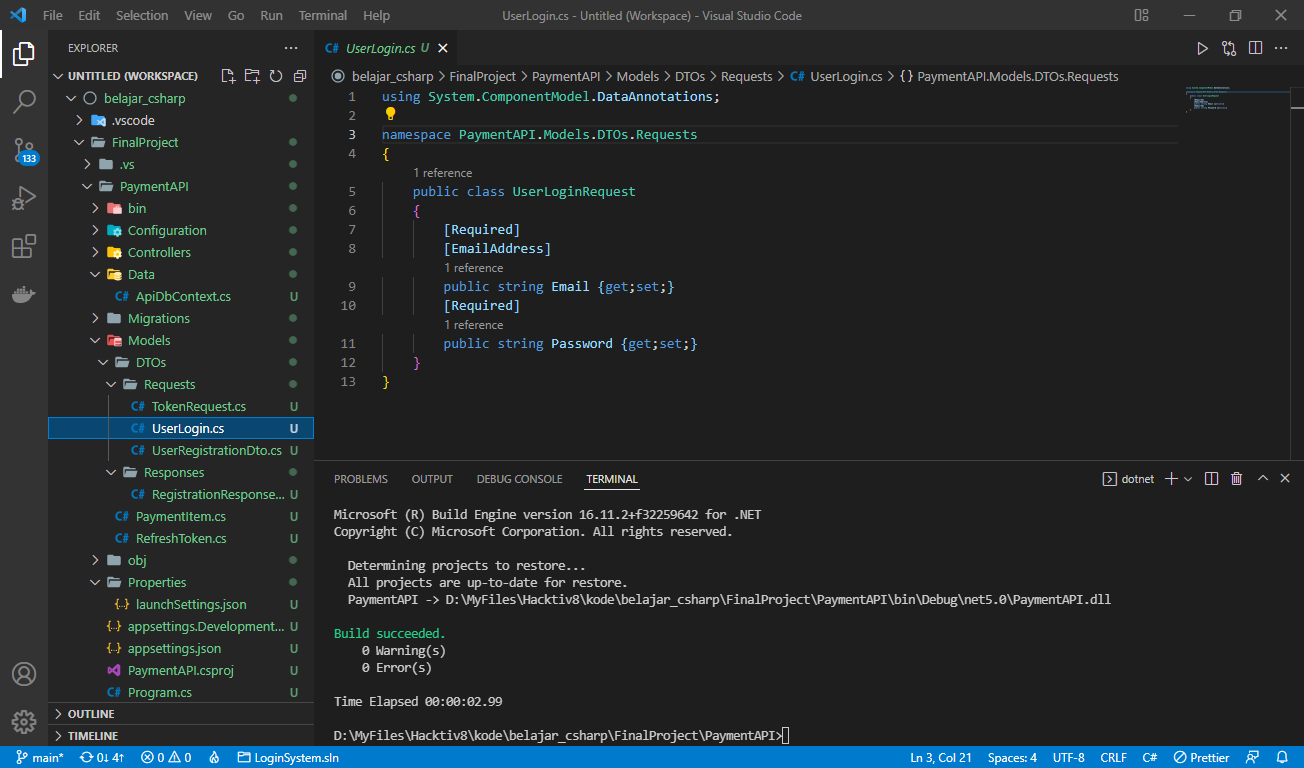


1. Pada ‘PaymentAPI/Models’ menambahkan DTO request dan response pada Models

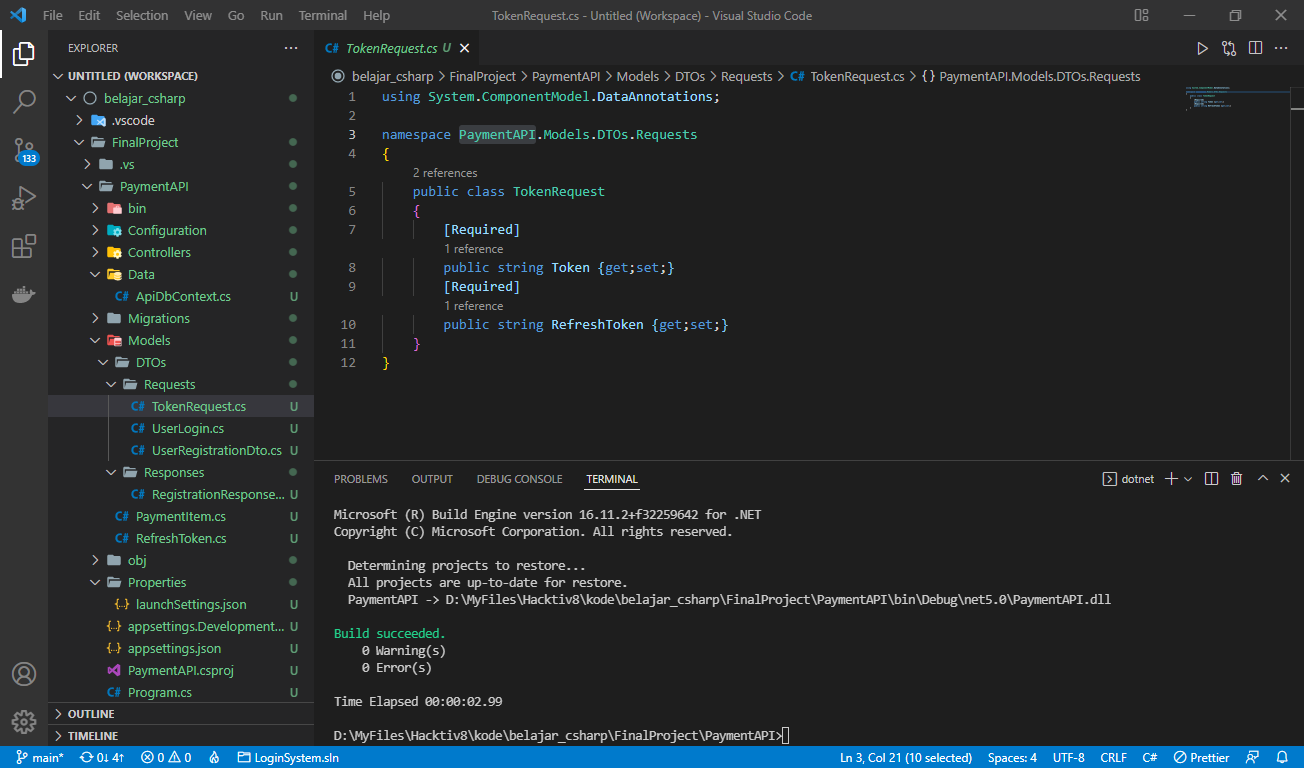
* ‘PaymentAPI/Models/DTOs/Requests/UserRegister.cs’



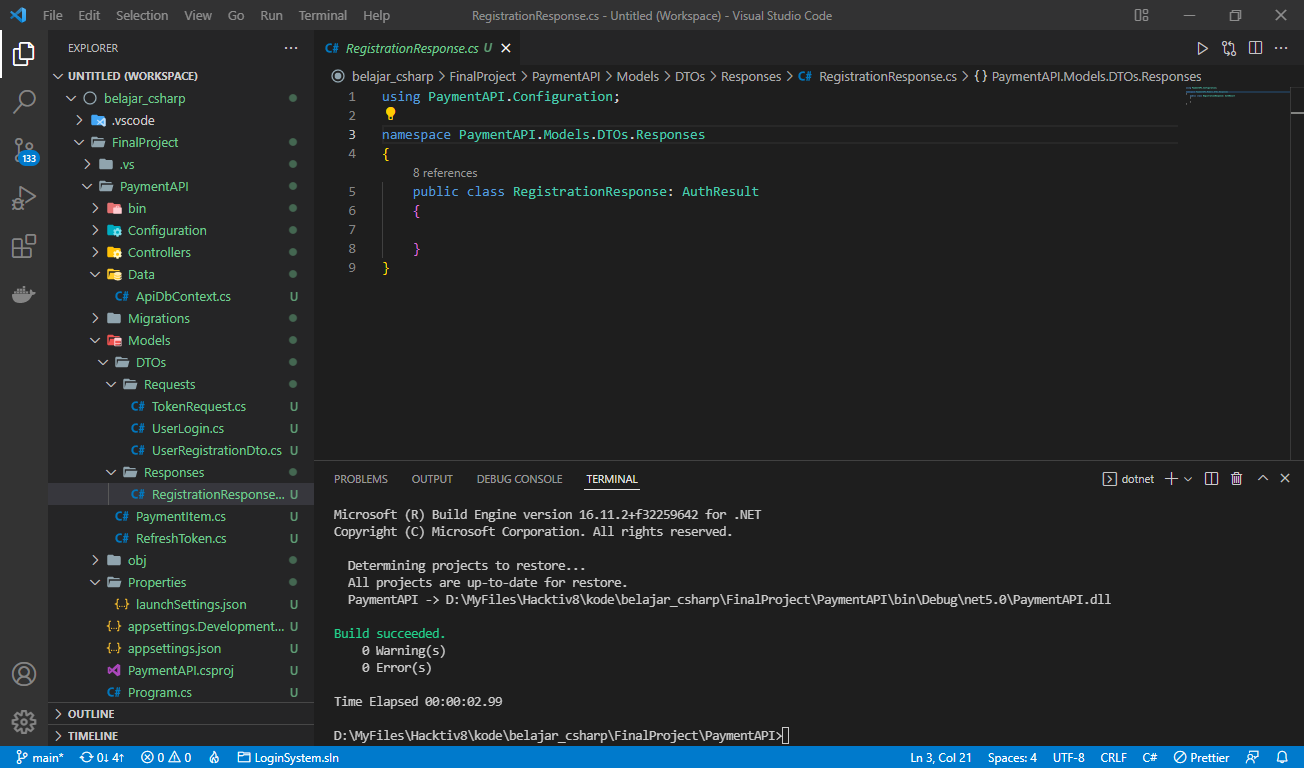
* ‘PaymentAPI/Models/DTOs/Requests/UserLogin.cs’



* ‘PaymentAPI/Models/DTOs/Requests/TokenRequest.cs’

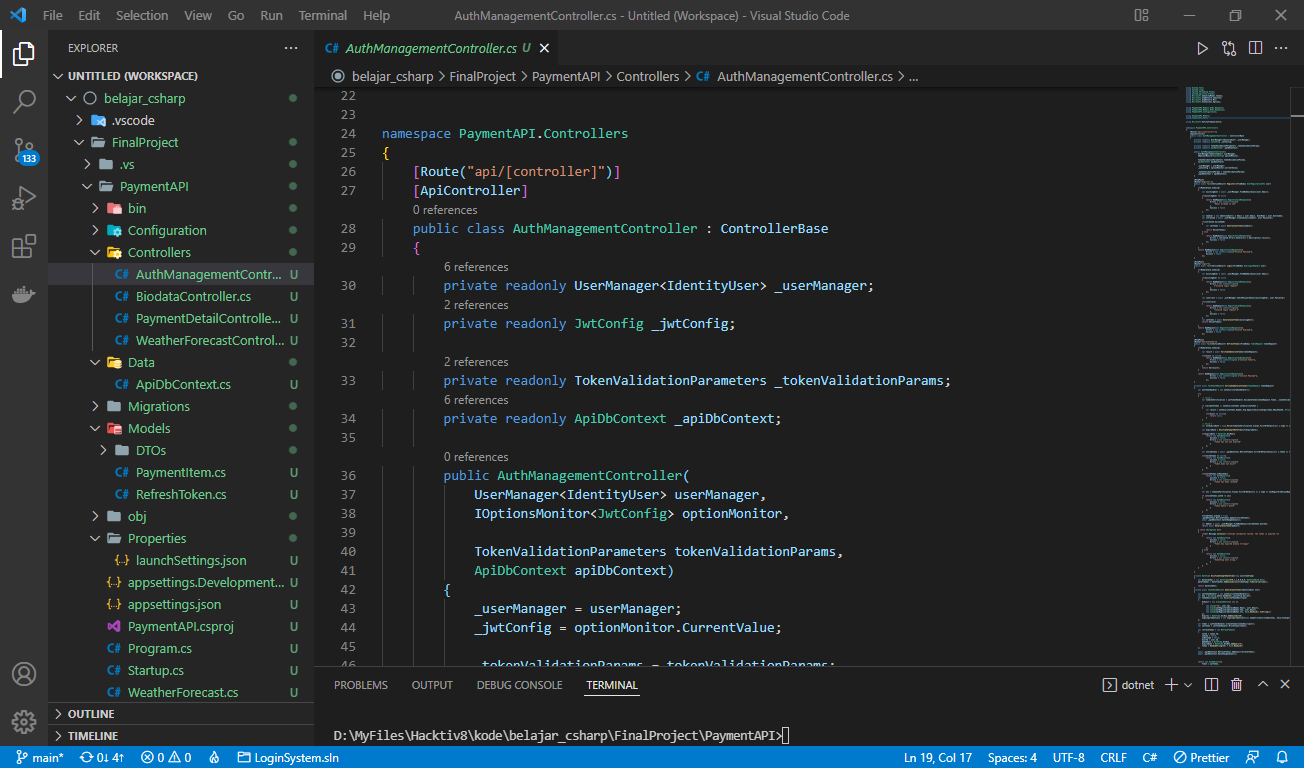


* ‘PaymentAPI/Models/DTOs/Responses/RegistrationResponse.cs’



1. Penambahan Controller untuk AuthManagement

‘PaymentAPI/Controllers\/AuthManagement.cs’



1. Pada ‘PaymentAPI/Startup.cs’, di dalam method ‘ConfigureServices’ setelah ‘services.AddDbContext’ menambahkan baris kode dibawah ini:

services.Configure<JwtConfig>(Configuration.GetSection("JwtConfig"));

            var key = Encoding.ASCII.GetBytes(Configuration["JwtConfig:Secret"]);

            var TokenValidationParameters = new TokenValidationParameters {

                    ValidateIssuerSigningKey = true,

                    IssuerSigningKey = new SymmetricSecurityKey(key),

                    ValidateIssuer = false,

                    ValidateAudience = false,

                    ValidateLifetime = true,

                    RequireExpirationTime = false,

                    ClockSkew = TimeSpan.Zero

                    };

            services.AddSingleton(TokenValidationParameters);

            services.AddAuthentication(options => {

                options.DefaultAuthenticateScheme = JwtBearerDefaults.AuthenticationScheme;

                options.DefaultScheme = JwtBearerDefaults.AuthenticationScheme;

                options.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme;

            } ).AddJwtBearer(jwt=>{

                jwt.SaveToken = true;

                jwt.TokenValidationParameters = TokenValidationParameters;

            });

            // using aspNetCoreUI

            services.AddDefaultIdentity<IdentityUser>(options => options.SignIn.RequireConfirmedAccount = true).AddEntityFrameworkStores<ApiDbContext>();

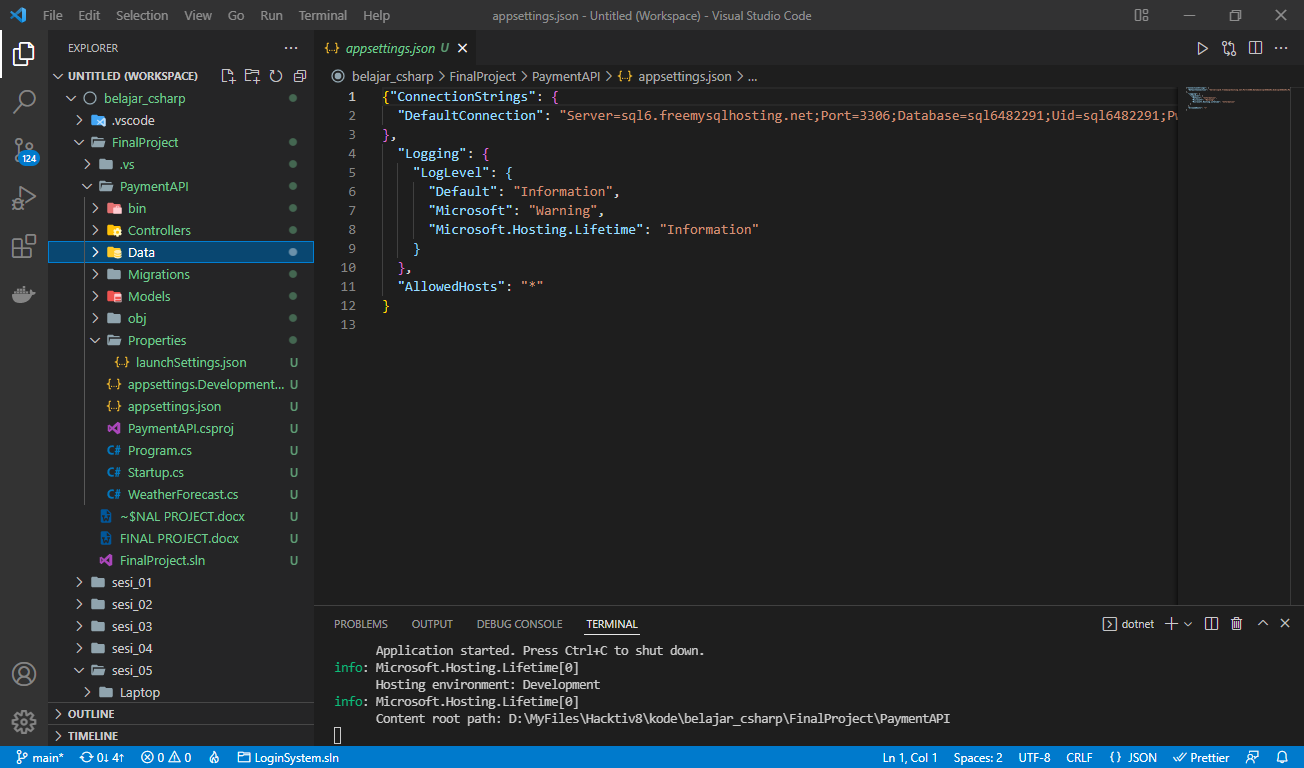
# Migrasi dan Update Database

1. Pada ‘PaymentAPI/appsettings.json’, menambahkan ‘ConnectionStrings.DefaultConnection’

"ConnectionStrings": {

  "DefaultConnection": "Server=sql6.freemysqlhosting.net;Port=3306;Database=sql6482291;Uid=sql6482291;Pwd=;"

}



1. Pada ‘PaymentAPI/Startup.cs’ menambahkan ‘services.AddDbContext’

services.AddDbContext<ApiDbContext>(options =>

                options.UseMySQL(Configuration.GetConnectionString("DefaultConnection")));

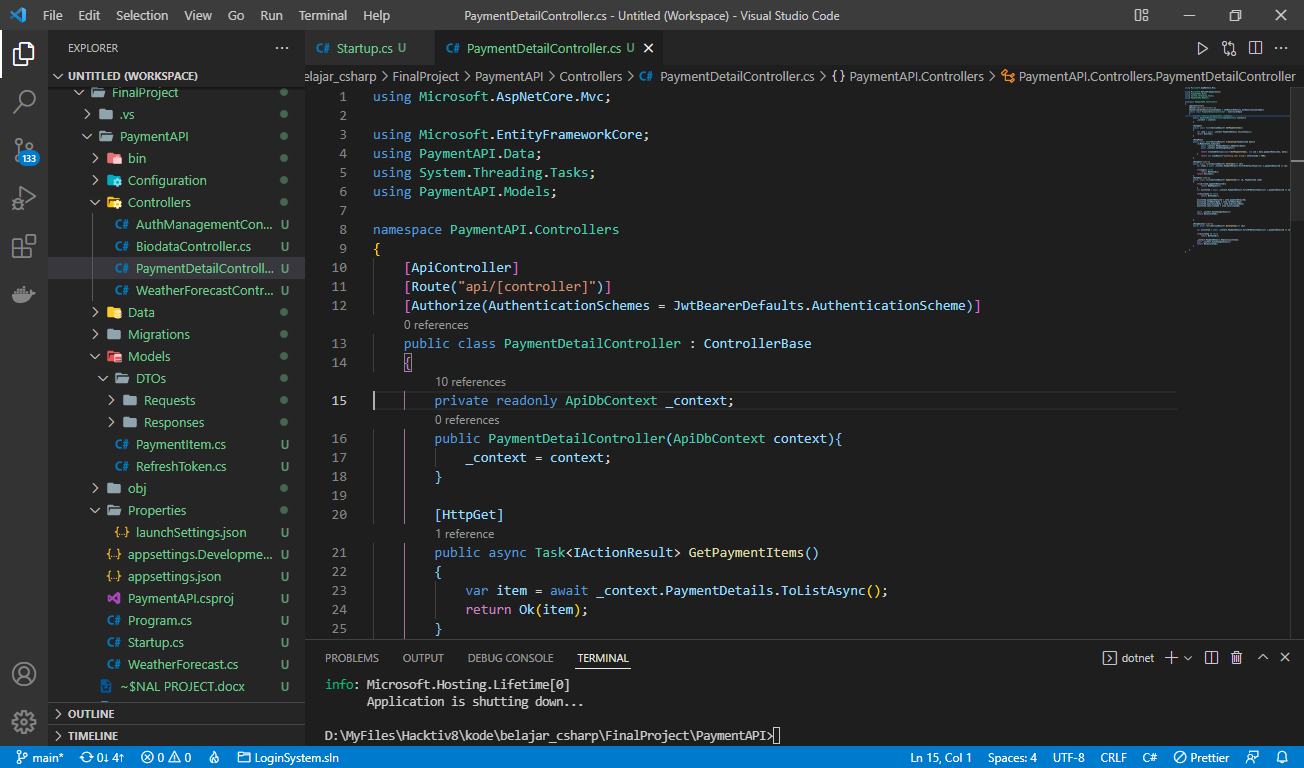
1. Pada ‘PaymentAPI/Controllers/PaymenDetailsController.cs’, pada bagian include (atas) tambahkan library:

using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Authentication.JwtBearer;

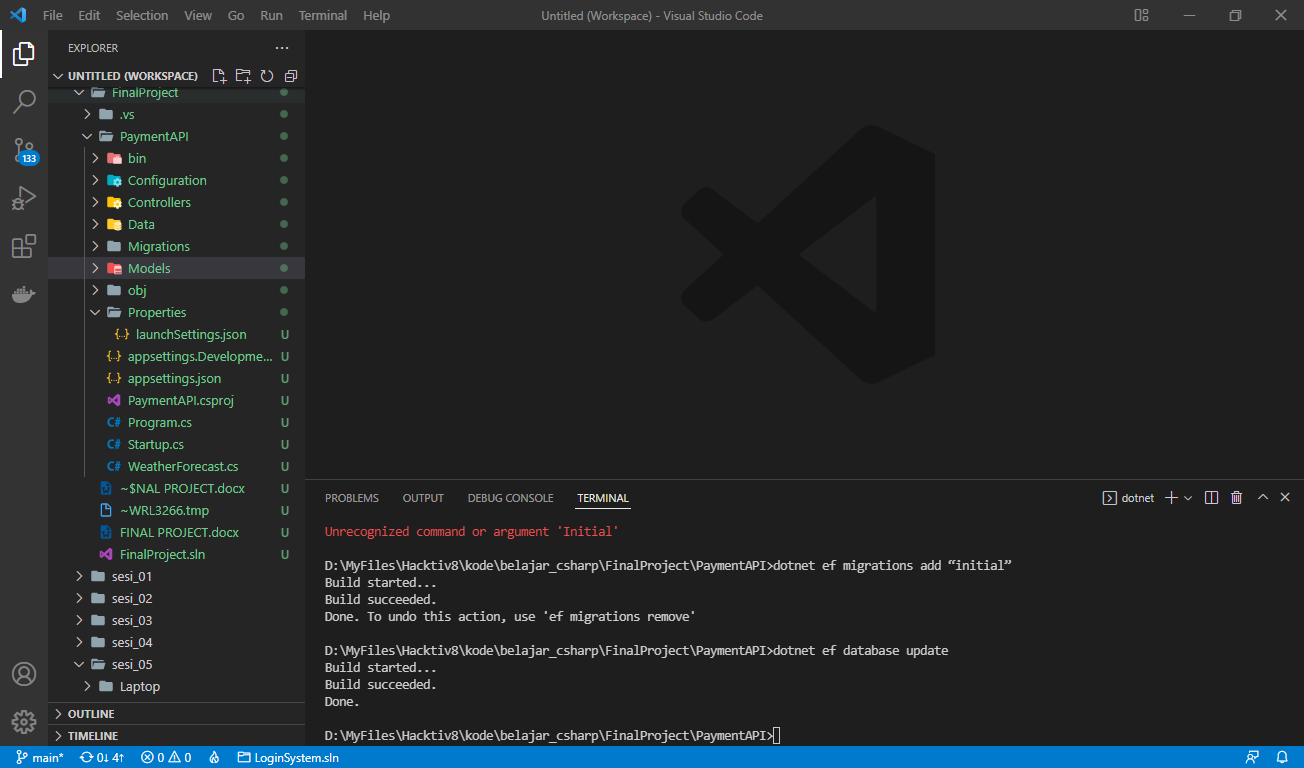
1. bagian setelah [ApiController] tambahakn baris dibawah ini untuk mengaktifkan Authorization.

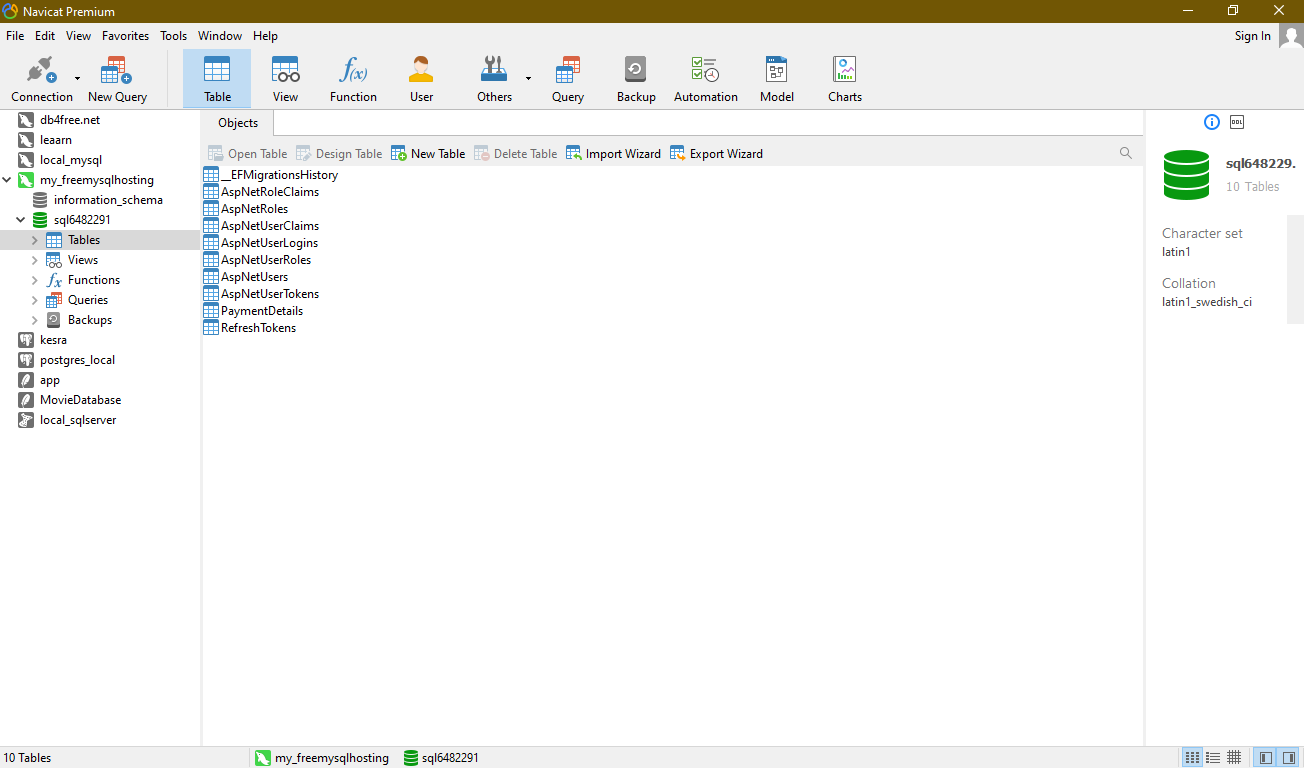
[Authorize(AuthenticationSchemes = JwtBearerDefaults.AuthenticationScheme)]



1. Setelah itu menjalankan pada terminal

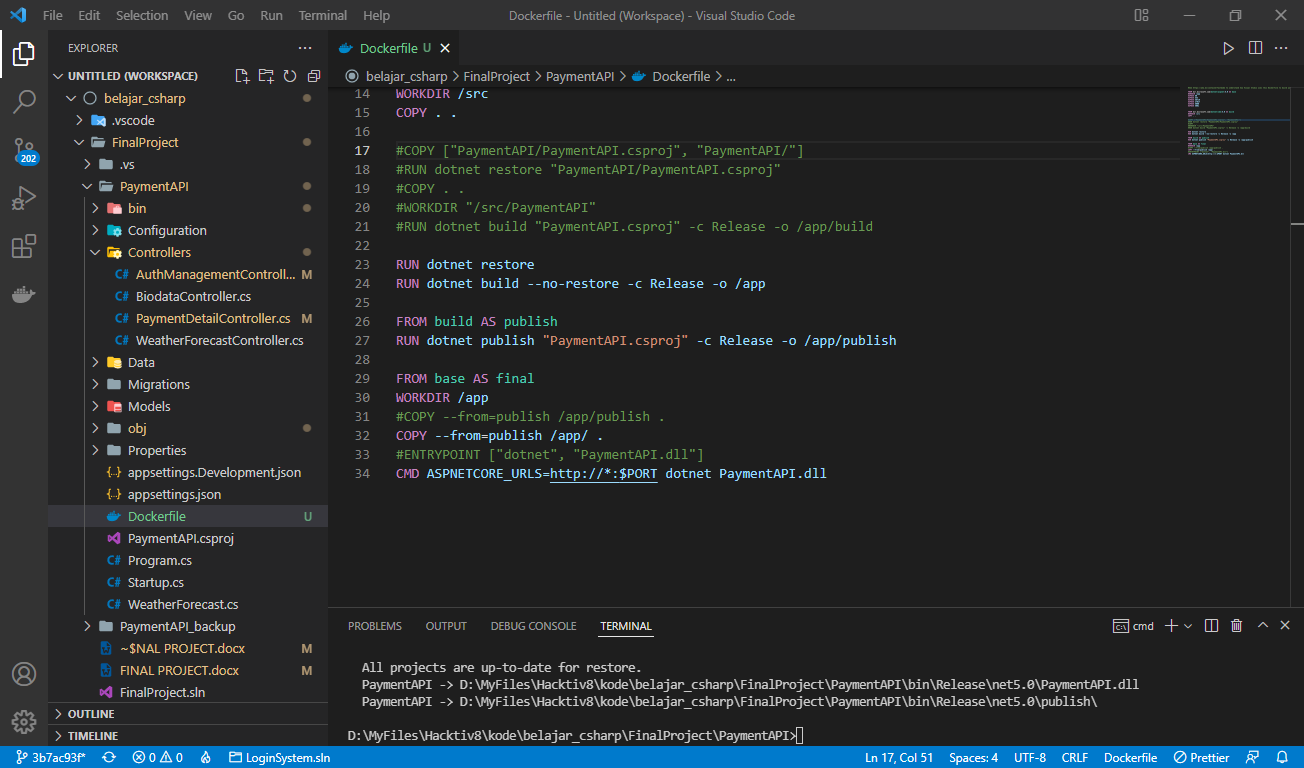
* dotnet ef migrations add “Initial”
* dotnet ef database update





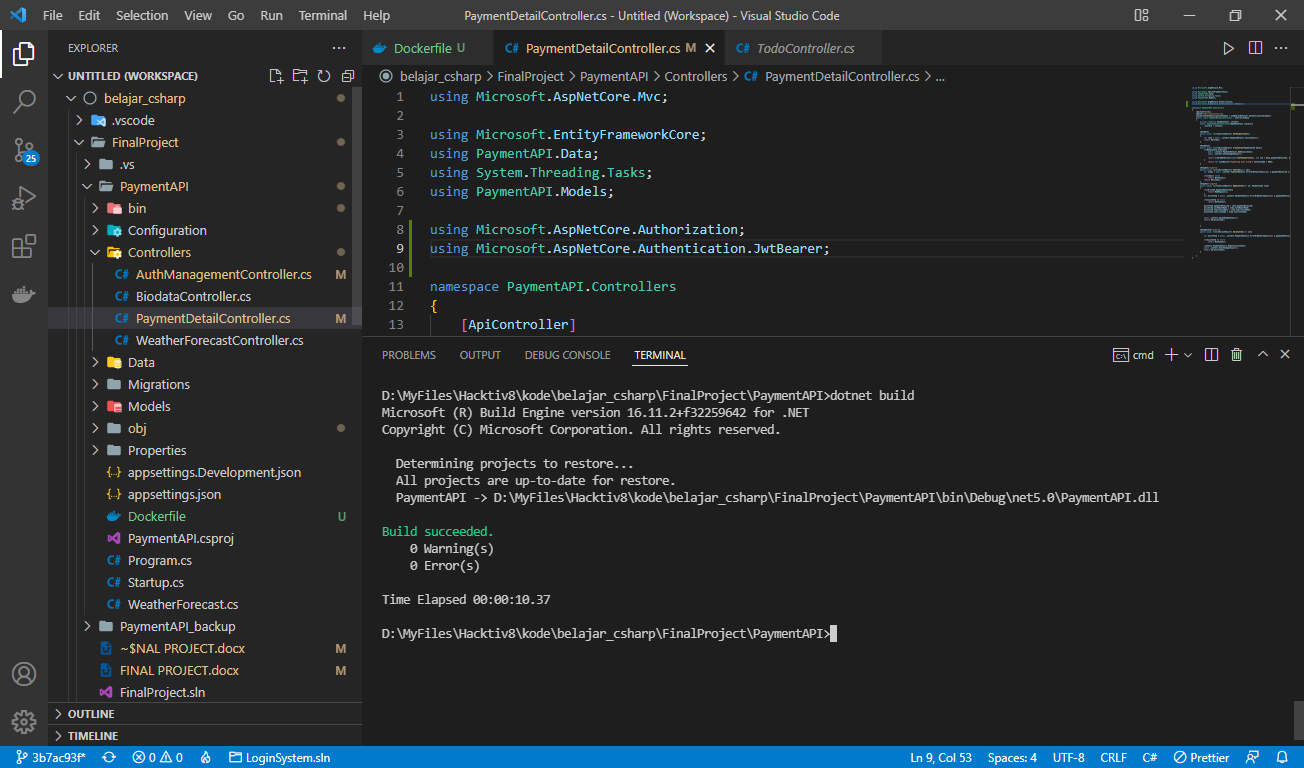
# Docker

1. Pada folder project ‘PaymentAPI’ tambahkan file ‘Dockerfile’



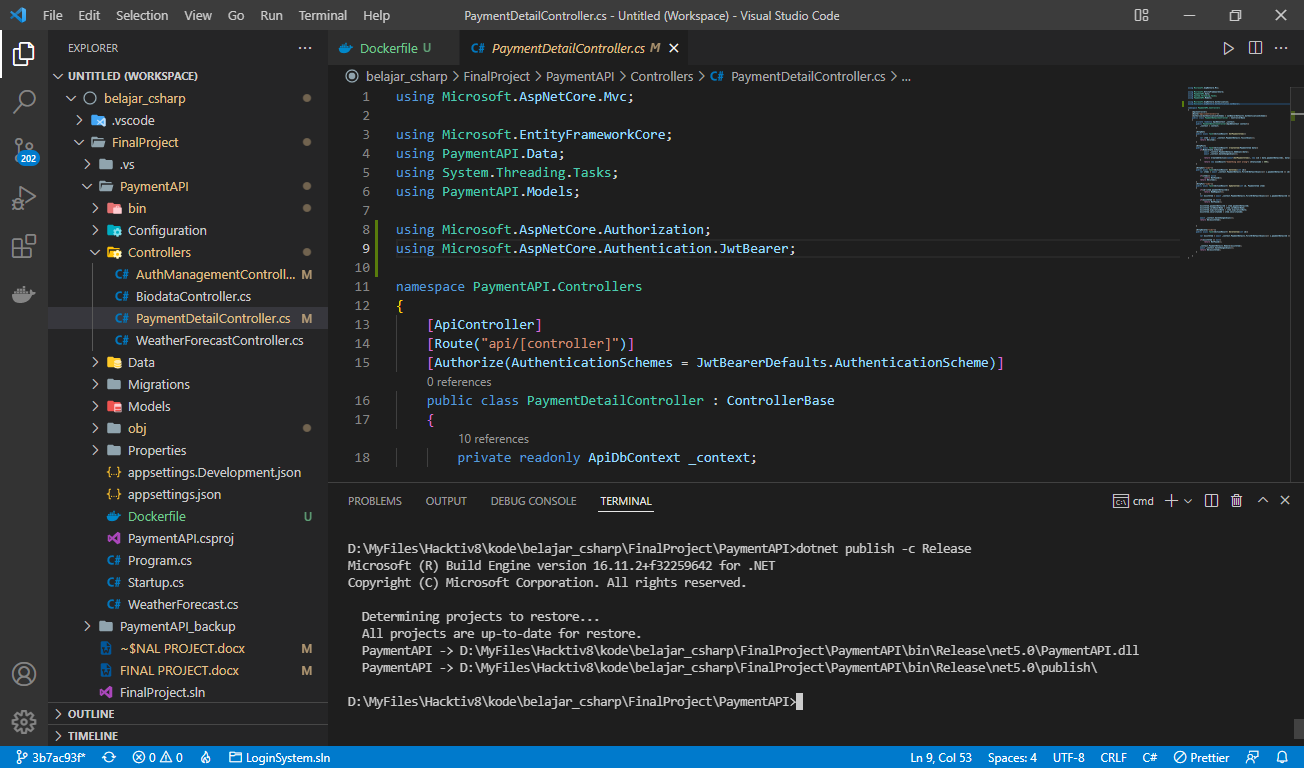
1. Build

dotnet build



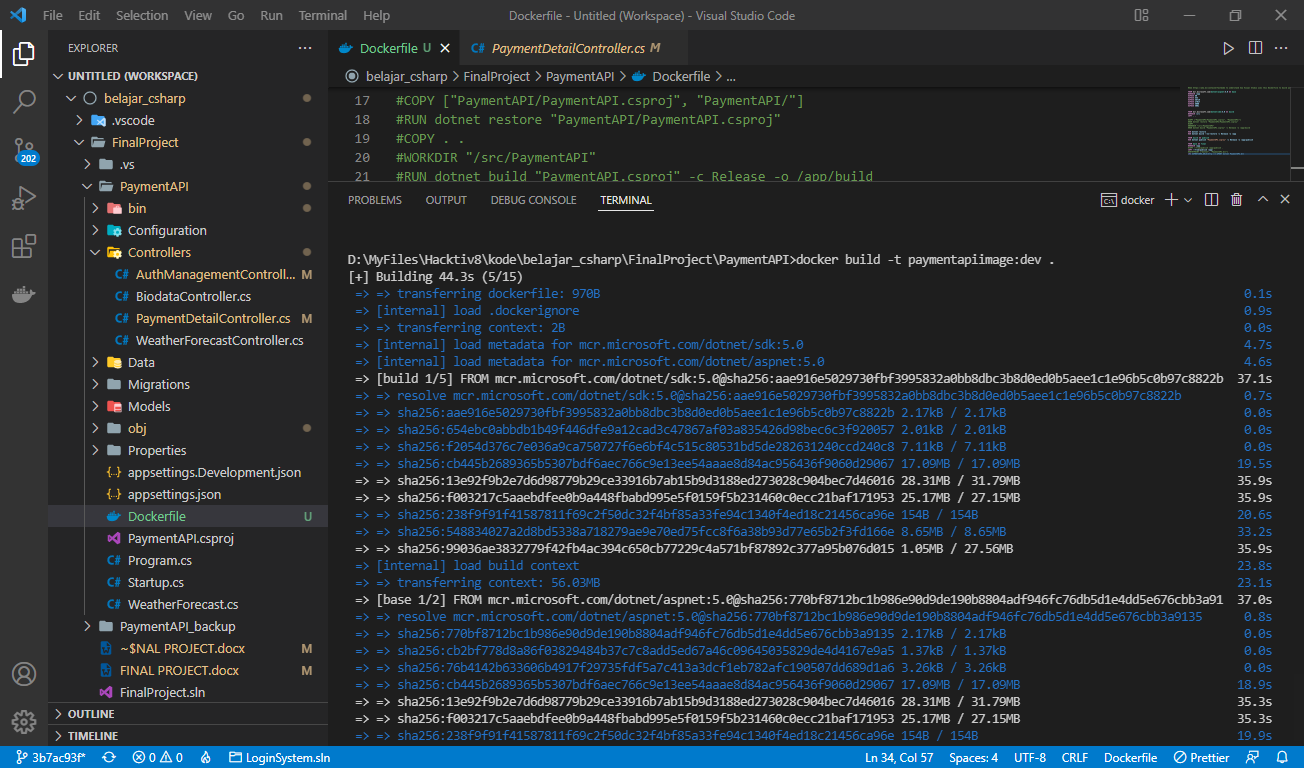
1. Release

dotnet publish -c Release



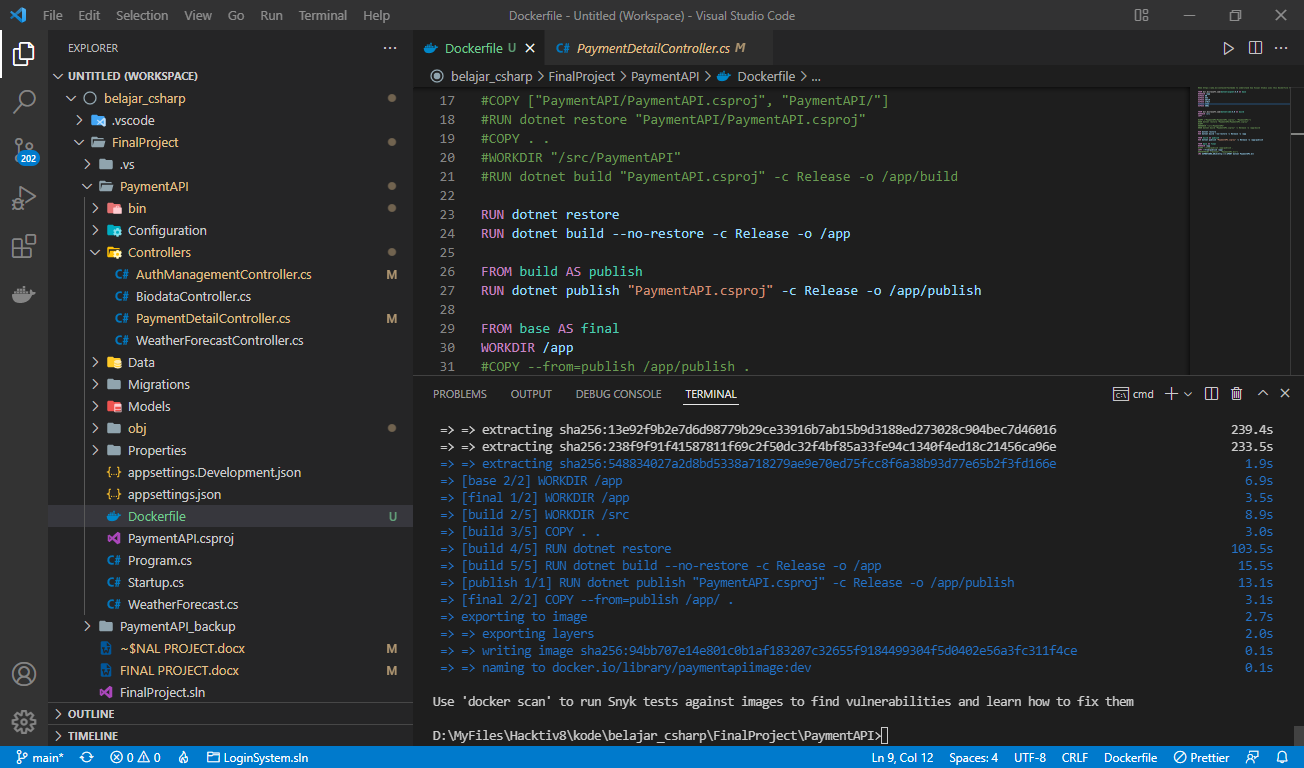
1. Jalankan pembuatan docker image

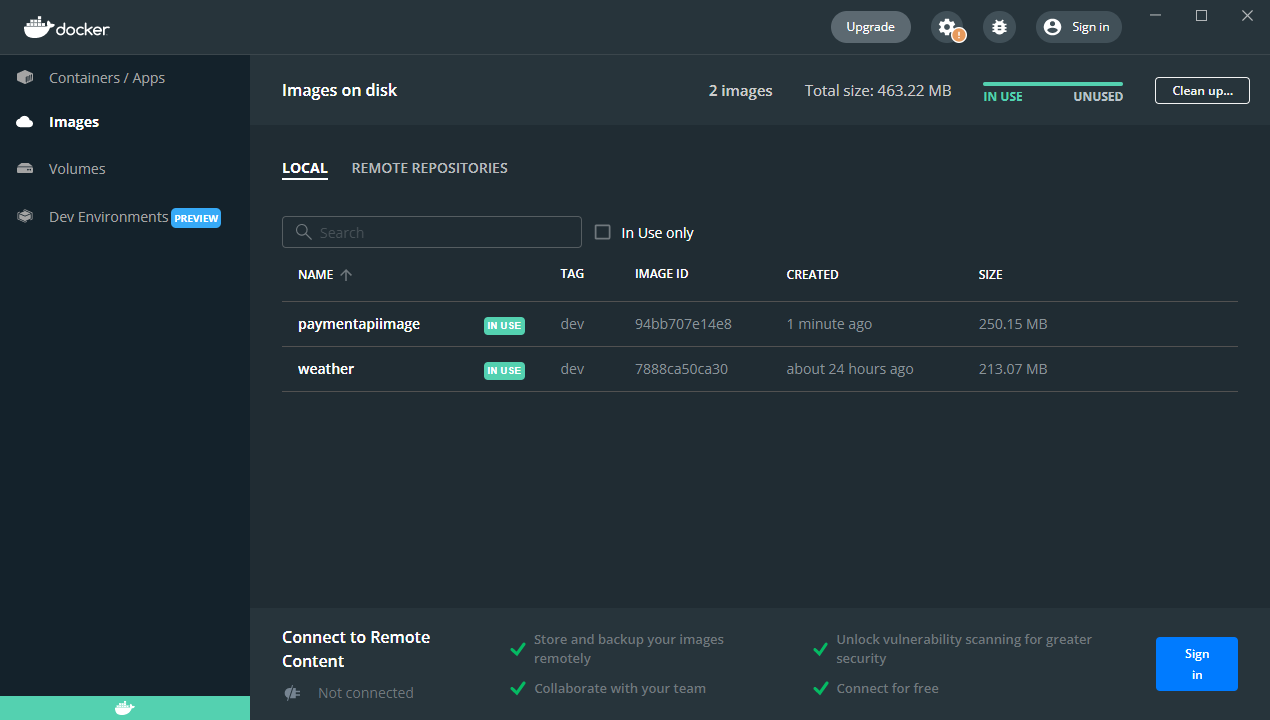
docker build -t paymentapiimage:dev .



note: nama image haru lowercase

Apabila proses berhasil tanpa kendala maka akan seperti ini



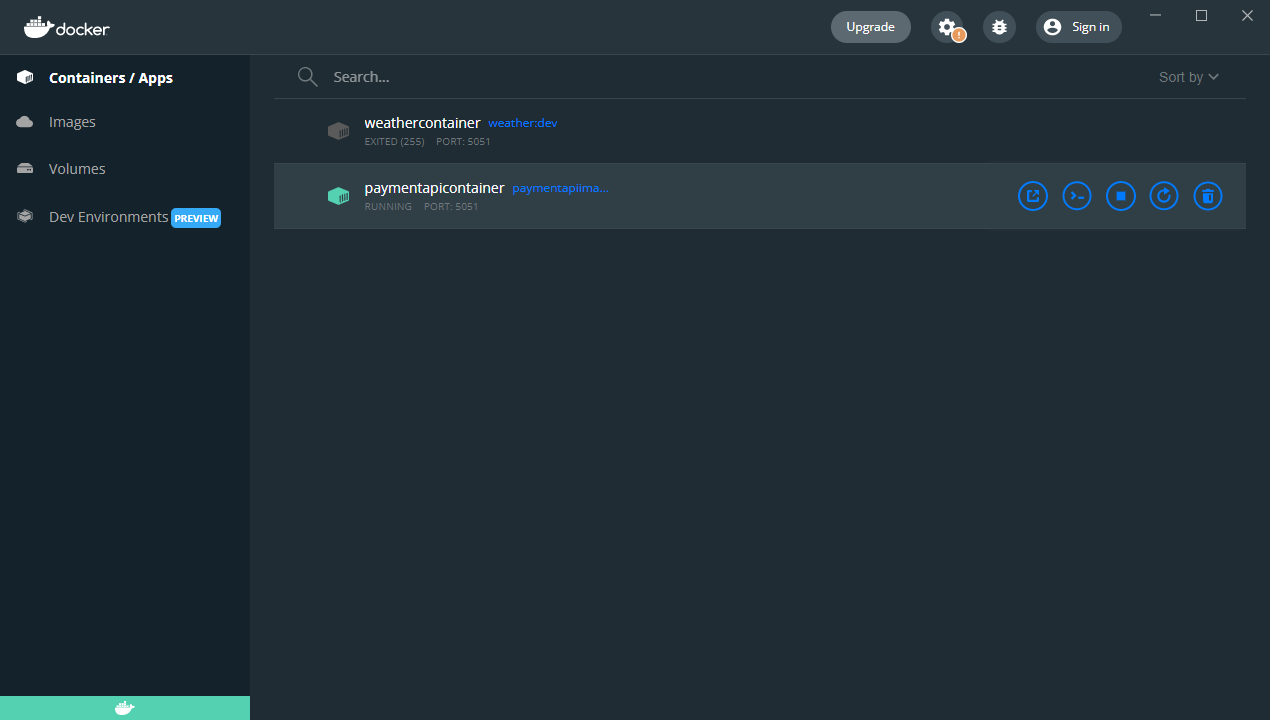


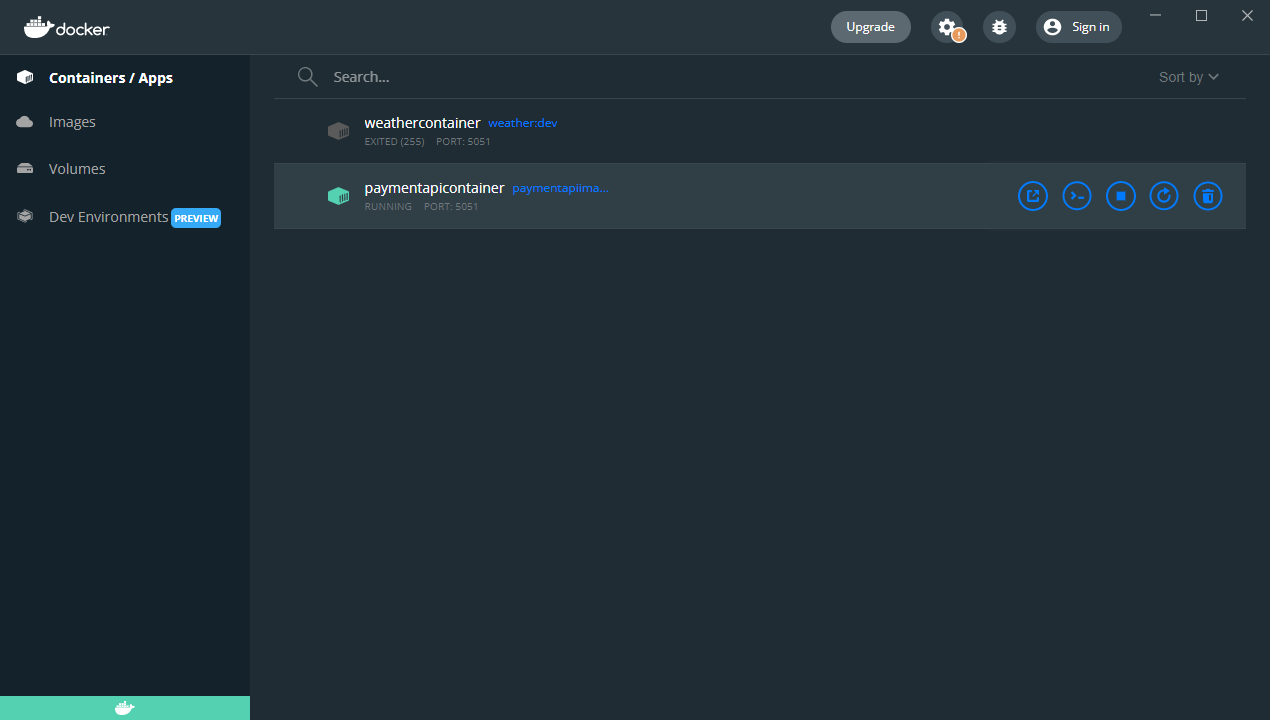
1. Membuat container dengan menjalankan

docker run -d -p 5051:80 --name paymentapicontainer paymentapiimage:dev

Note: aplikasi akan berjalan pada port 5051 saat dilakukan running pada Docker

1. Cek pada Aplikasi Docker, bagian container, akan terdapat ‘paymentapicontainer running’ menandakan bahwa aplikasi telah dijalankan

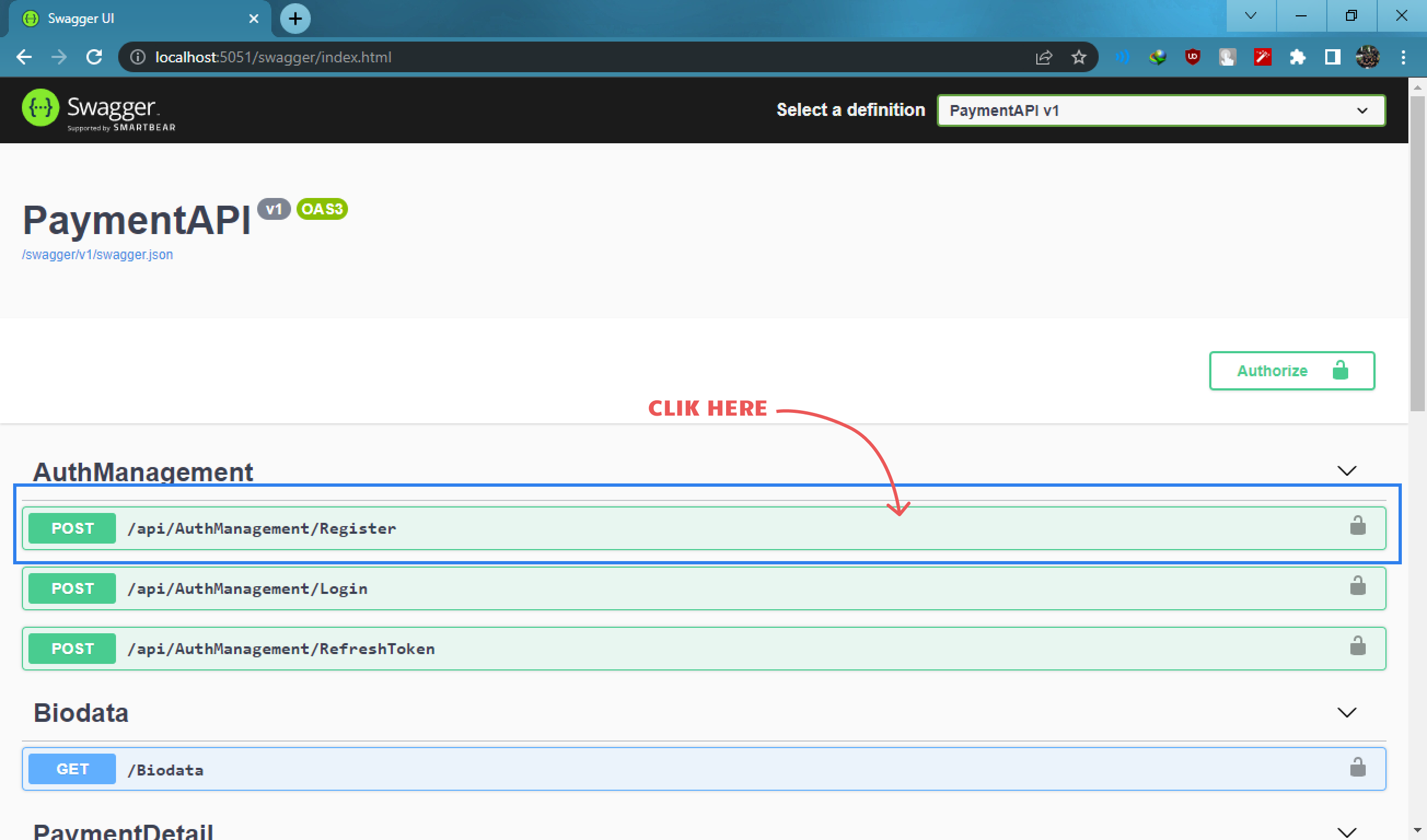


1. Masih pada Docker, silahkan Klik ini  untuk menjalankan melalui browser

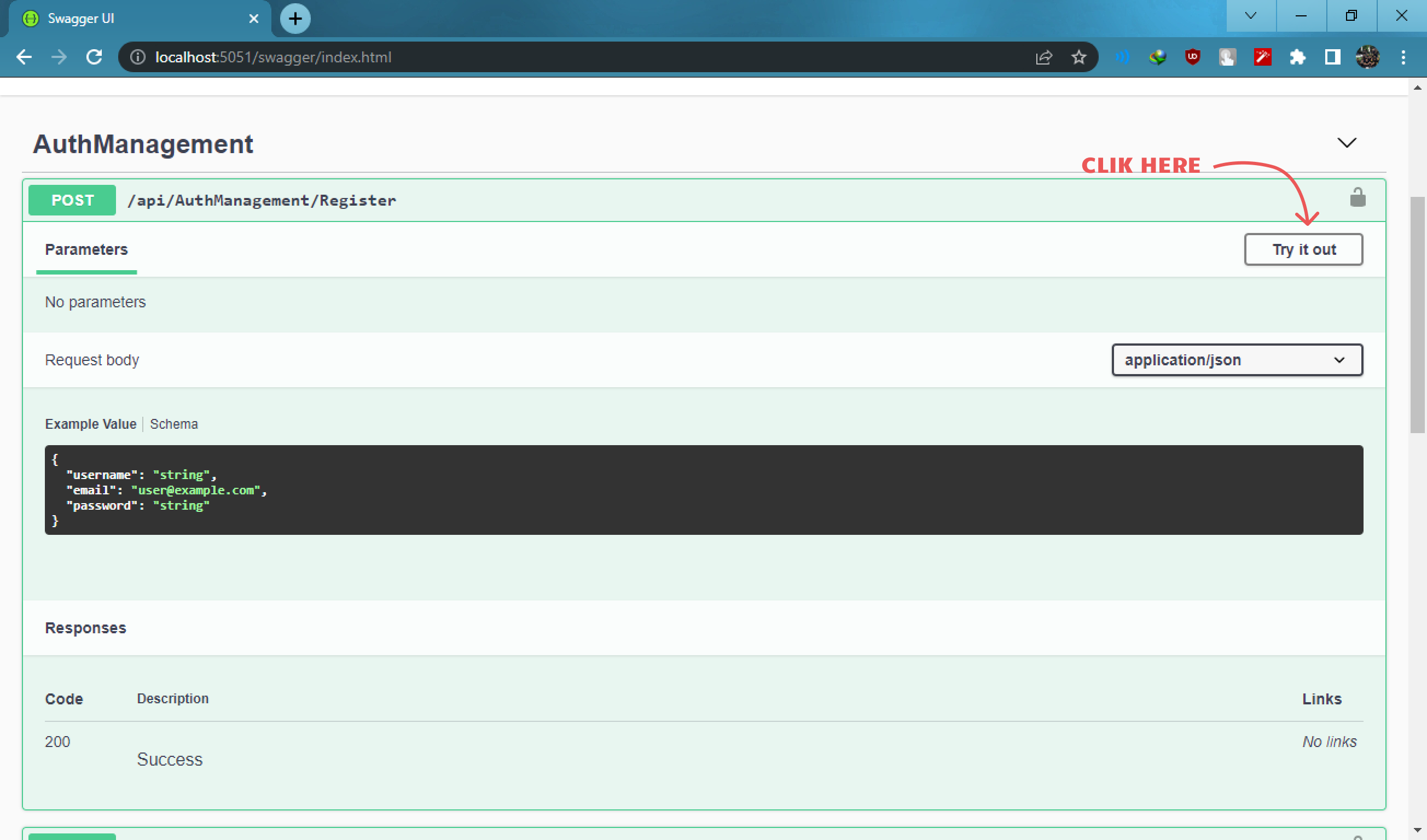
# Test Running App:

## Cara Register

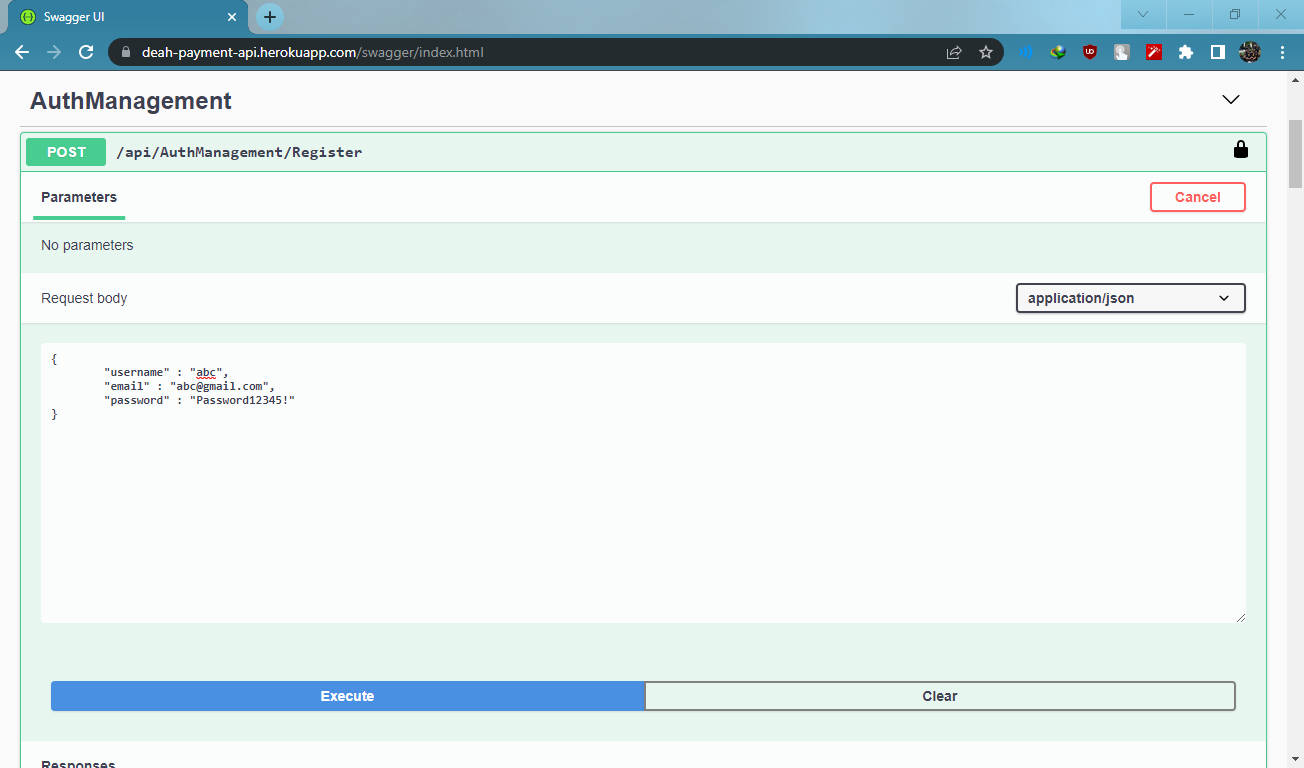
1. Klik Card POST: Register



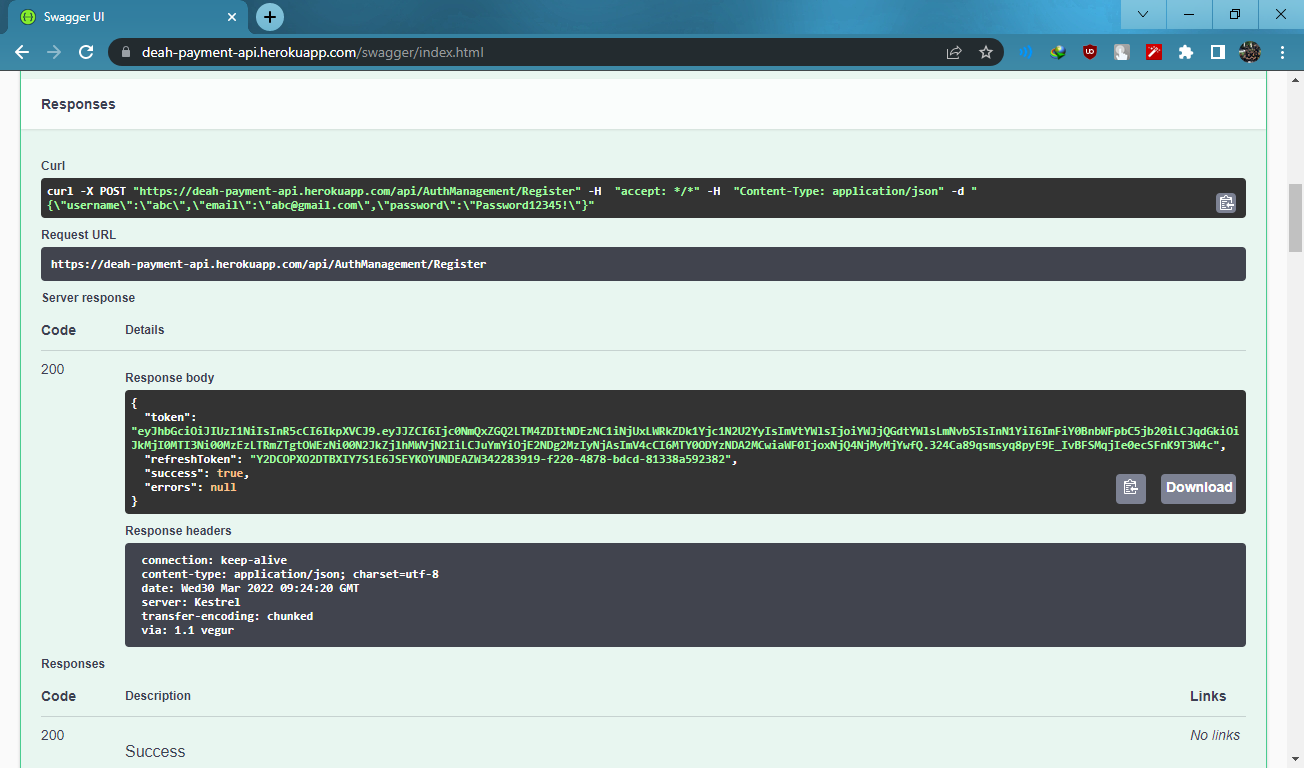
1. Klik ‘try it out’ untuk menggunakan



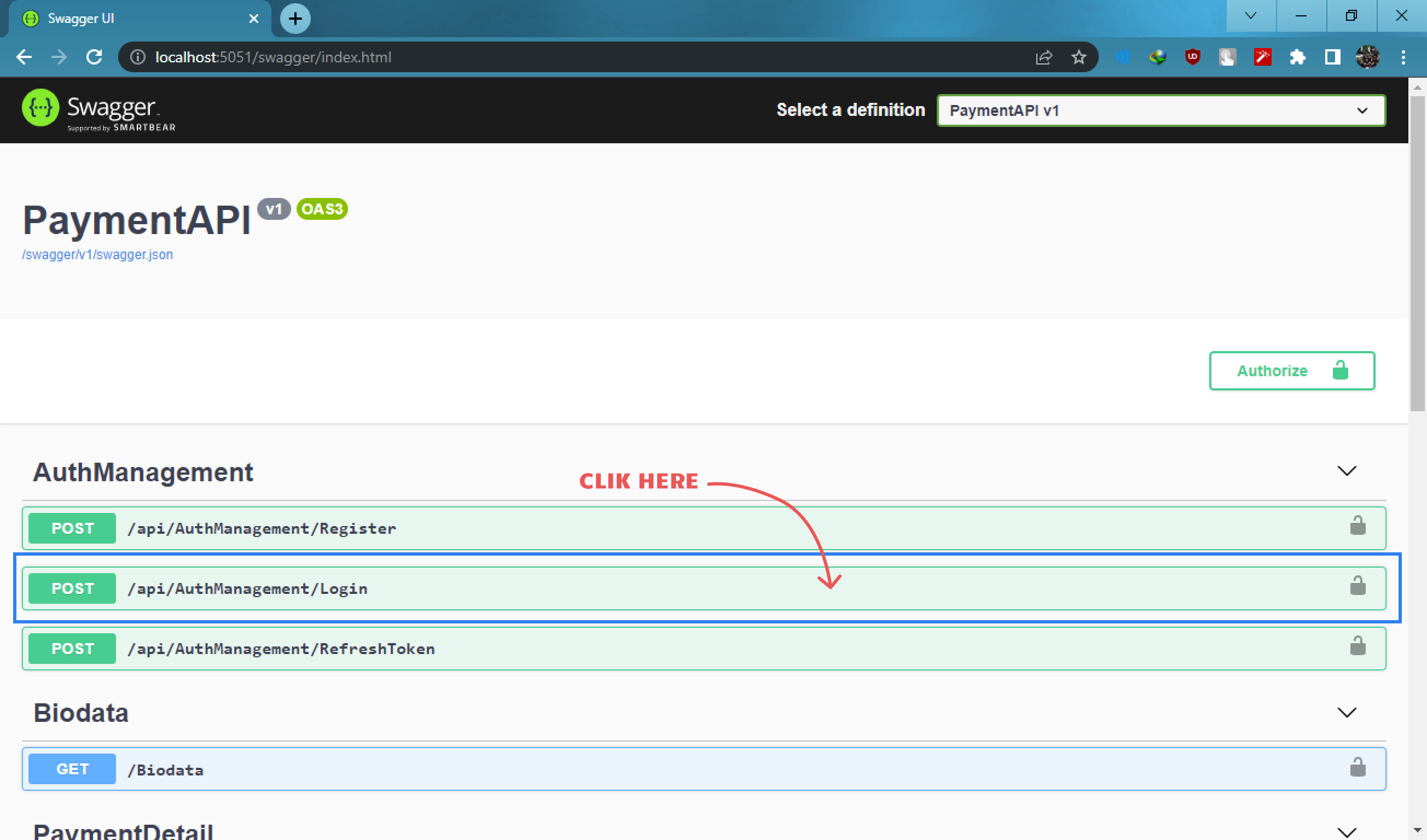
1. Masukan JSON pada Kolom input Request Body > Klik Execute



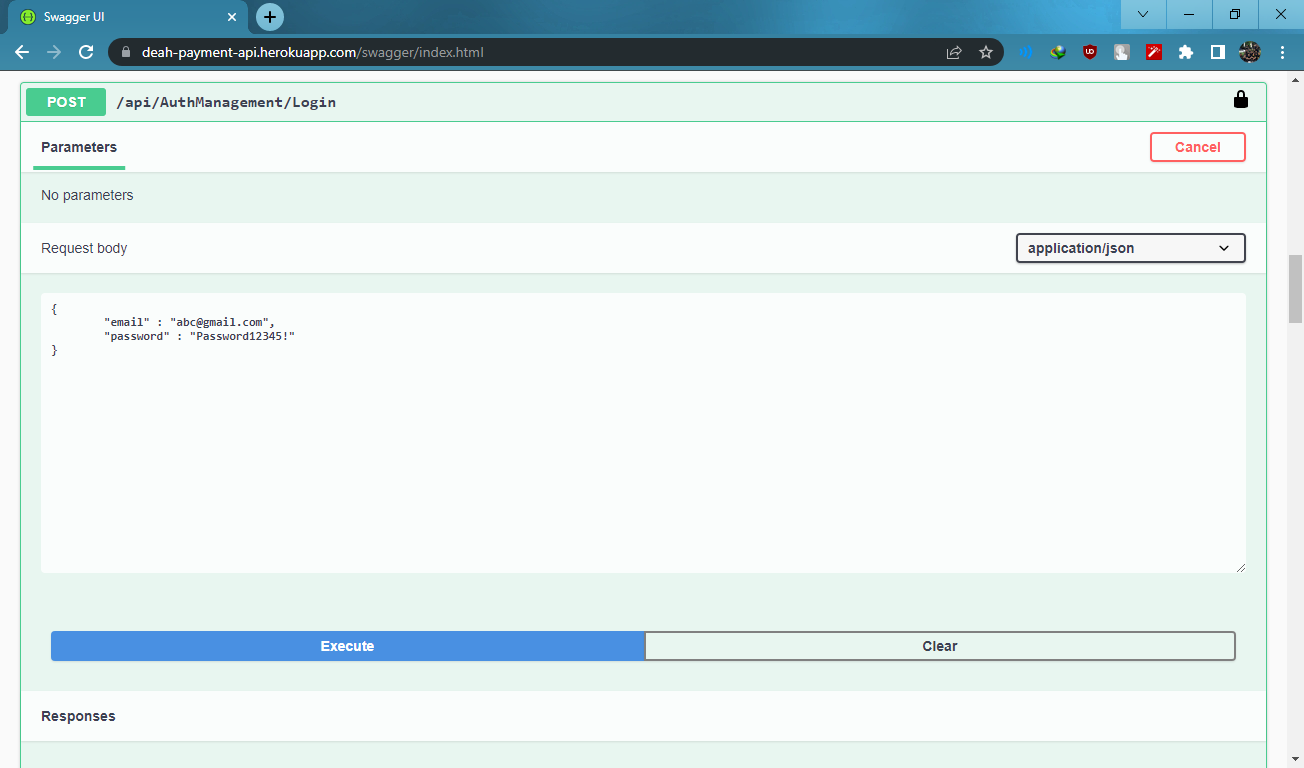
1. Berikut response menandakan bahwa registrasi telah berhasil



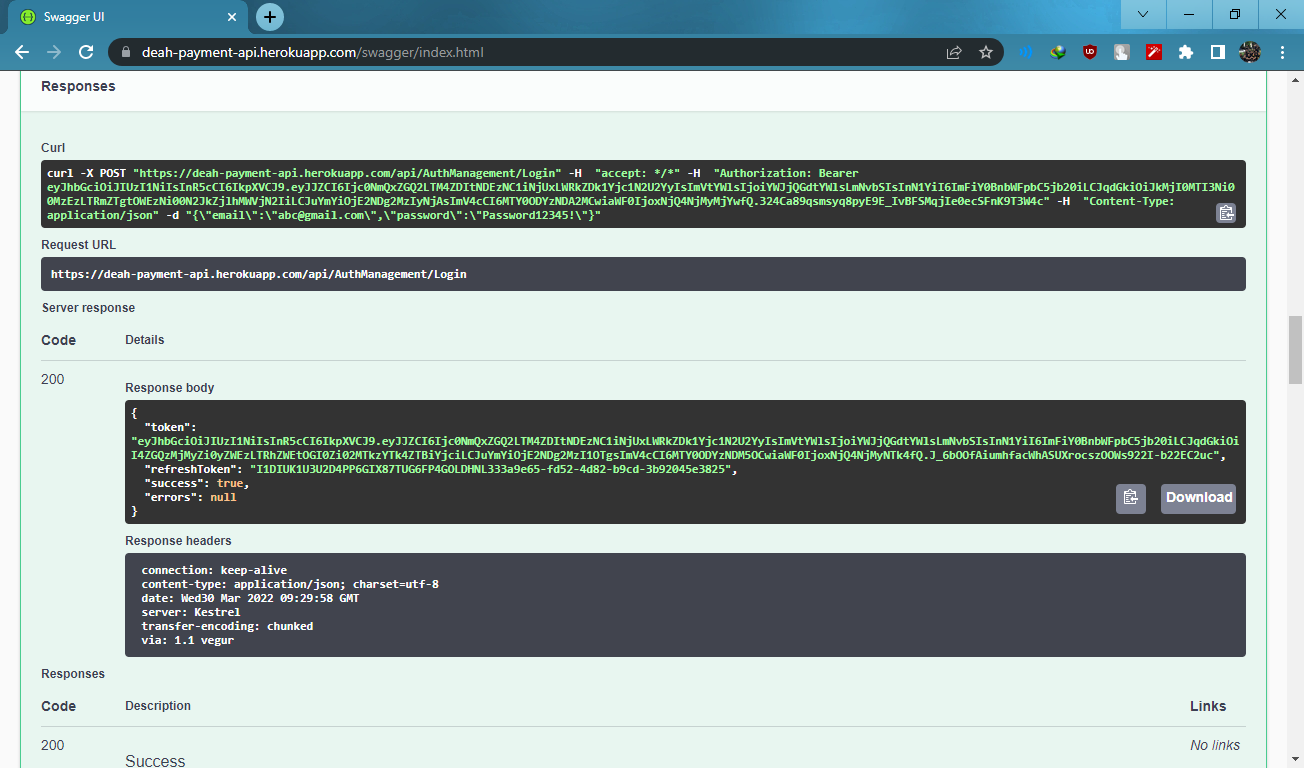
## Cara Login



1. Masukan pada kolom input request body > klik Execute

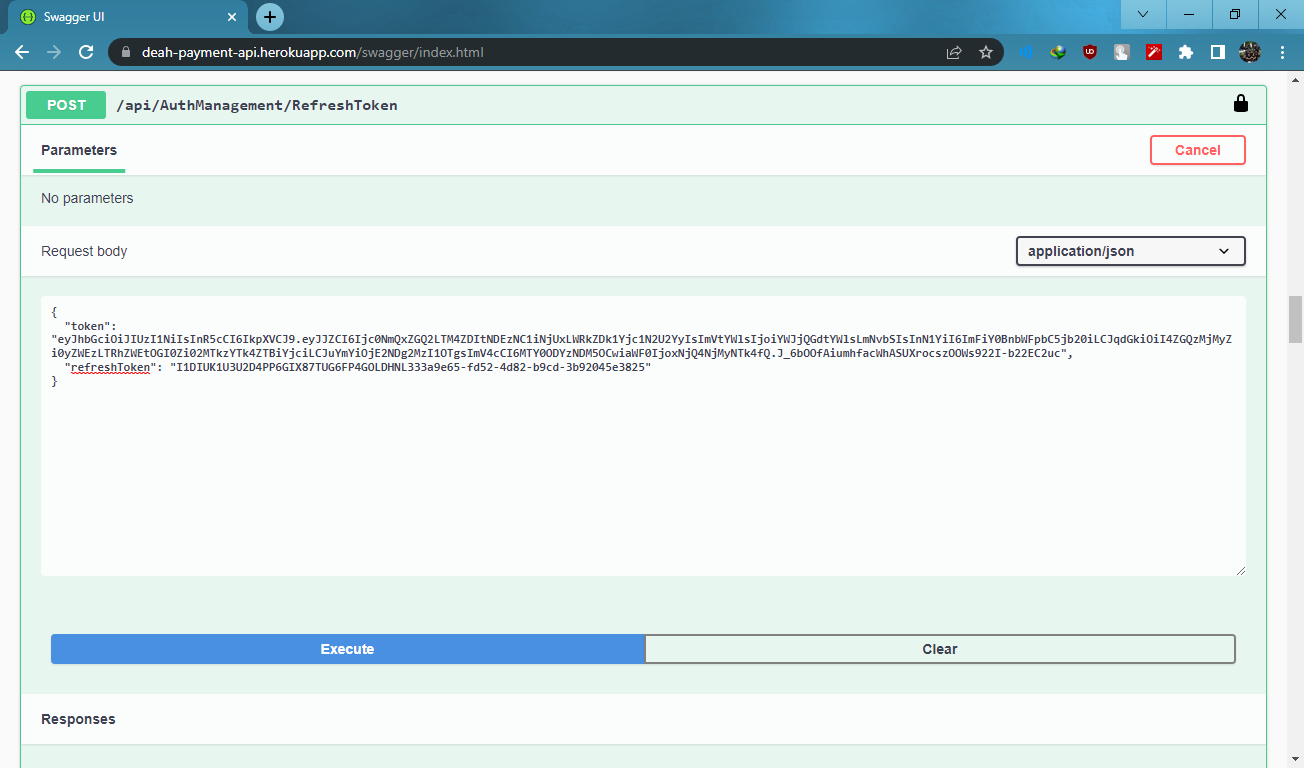


1. Berikut persponse menandakan login berhasil

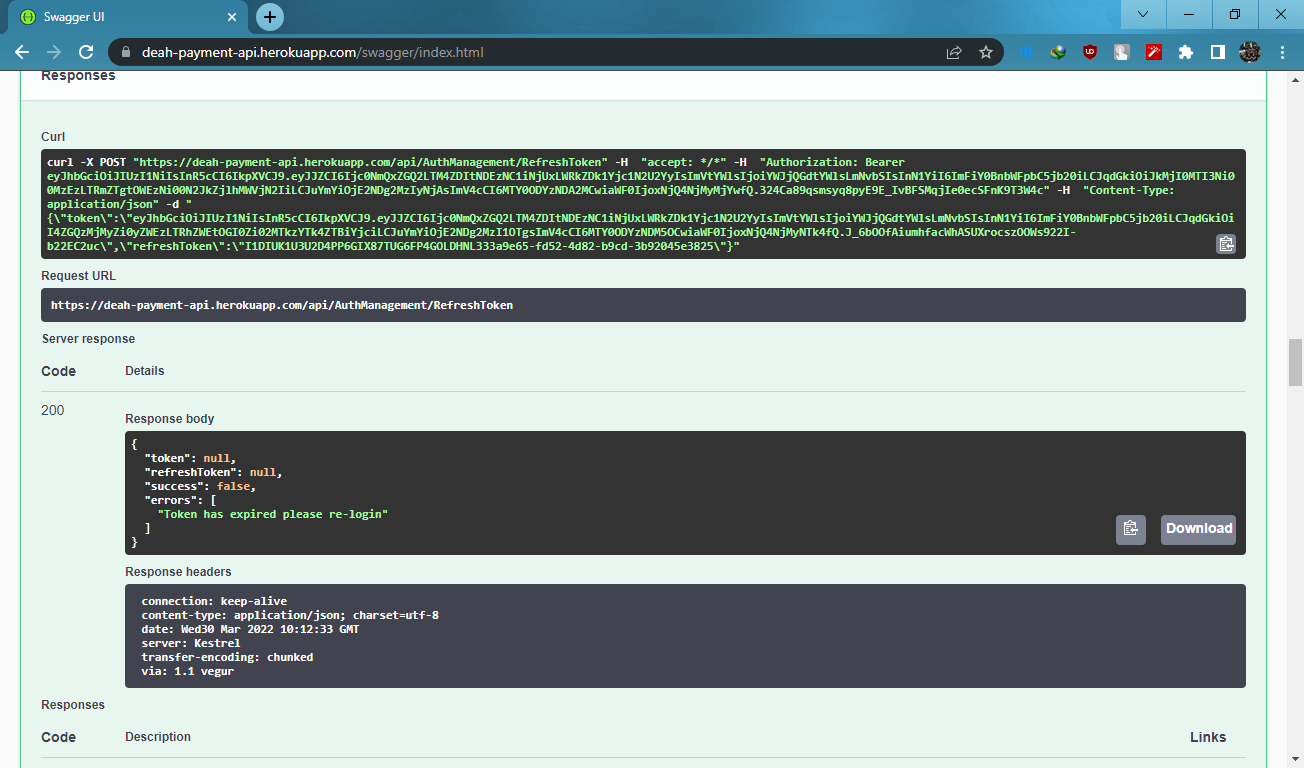


## Refresh Token – pengecekan apakah token masih berlaku atau sudah expired

1. Masukan input token dan refresh token yang didapatkan saat login/register pada kolom input body > klik Execute

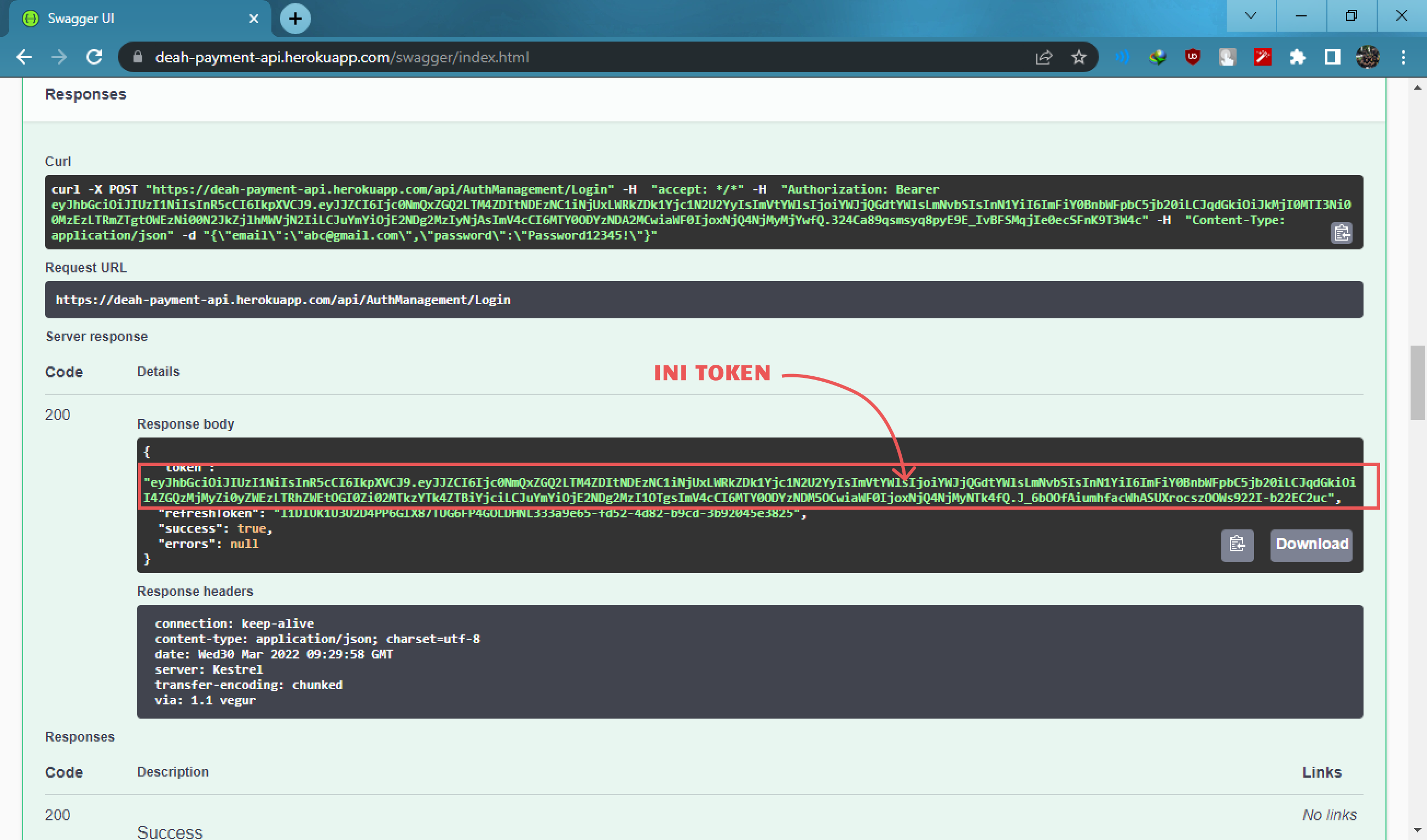


1. Berikut saat token belum expired
2. Berikut hasil saat token expired

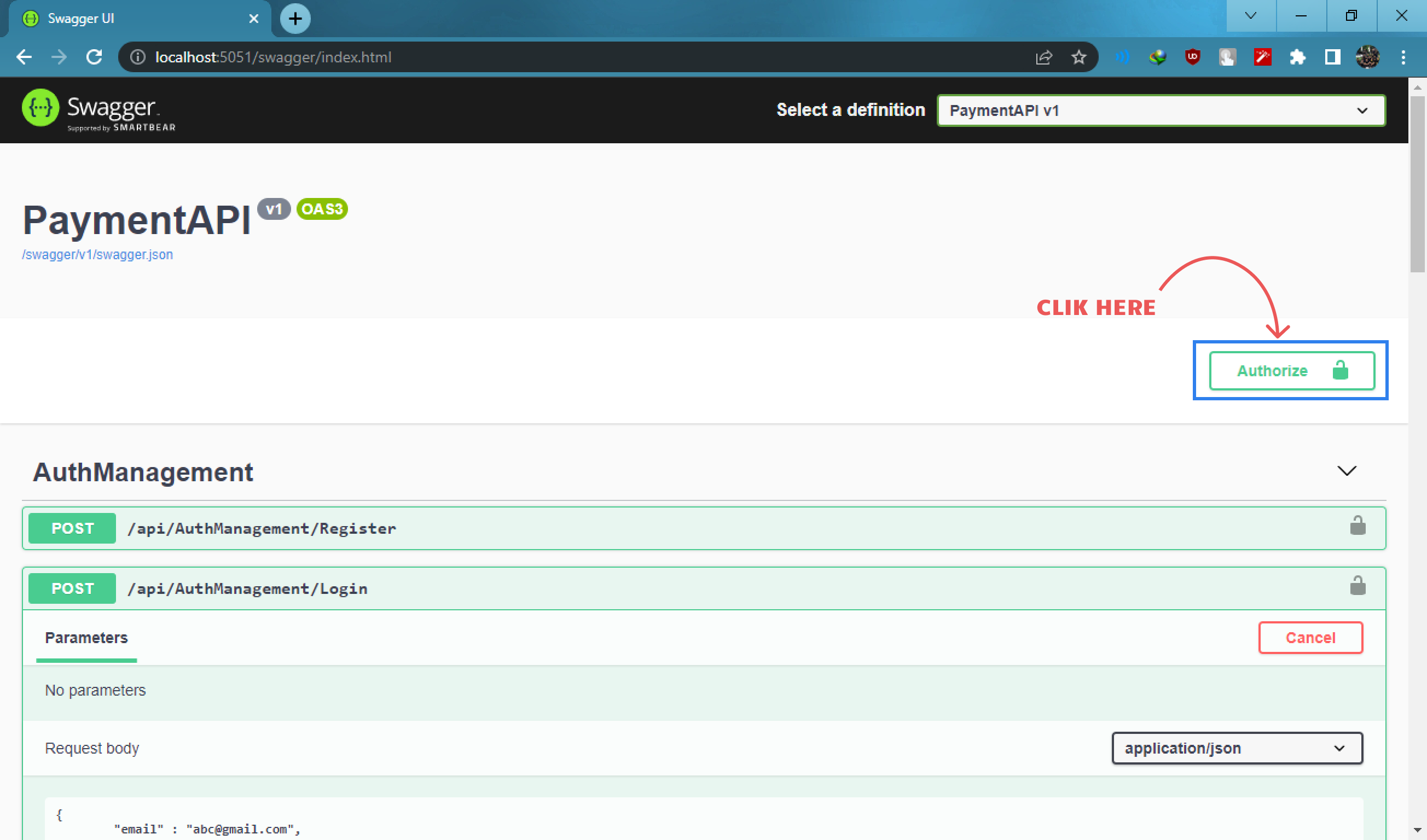


## Authorize: memasukan Token

1. Pada saat login, akan mendapat token yang bisa dipakai



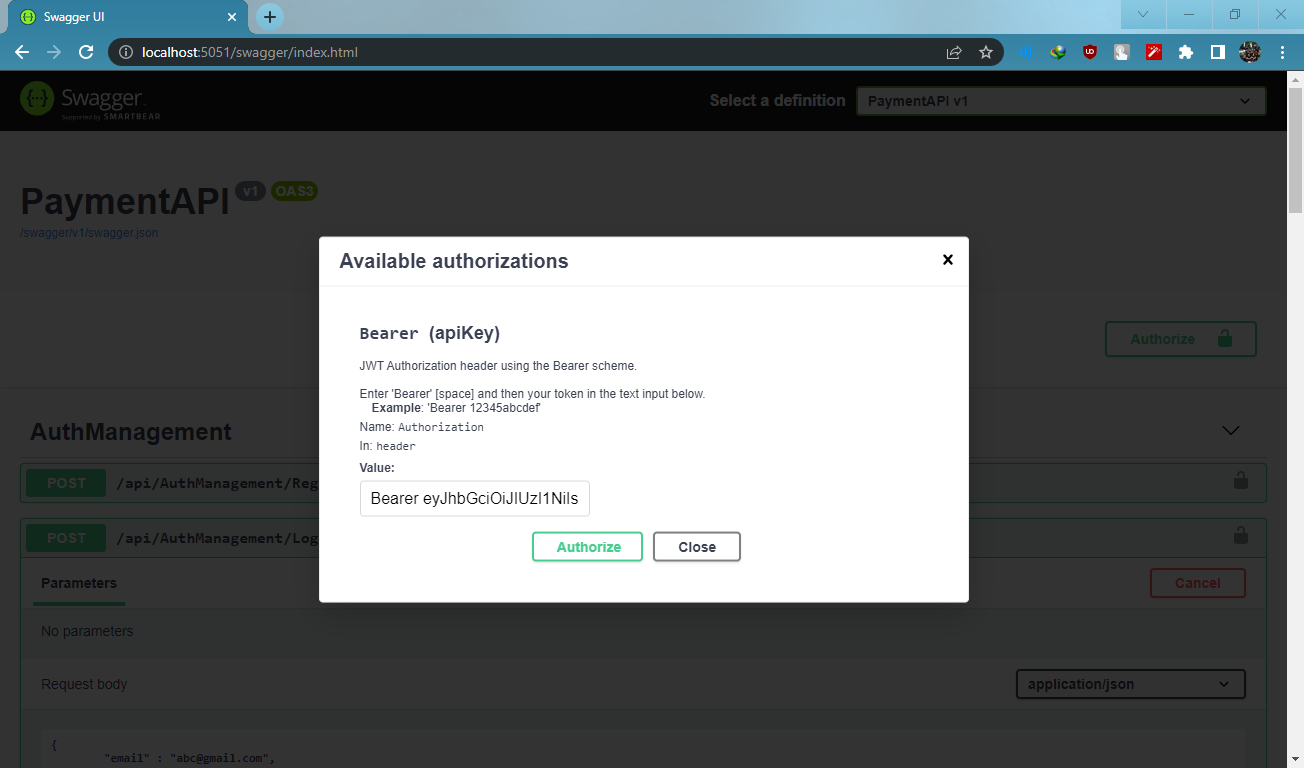
1. Pada pojok kanan atas klik button ‘Authorize’



1. Akan muncul modal seperti dibawah, Pada kolom value masukan

Bearer <token>

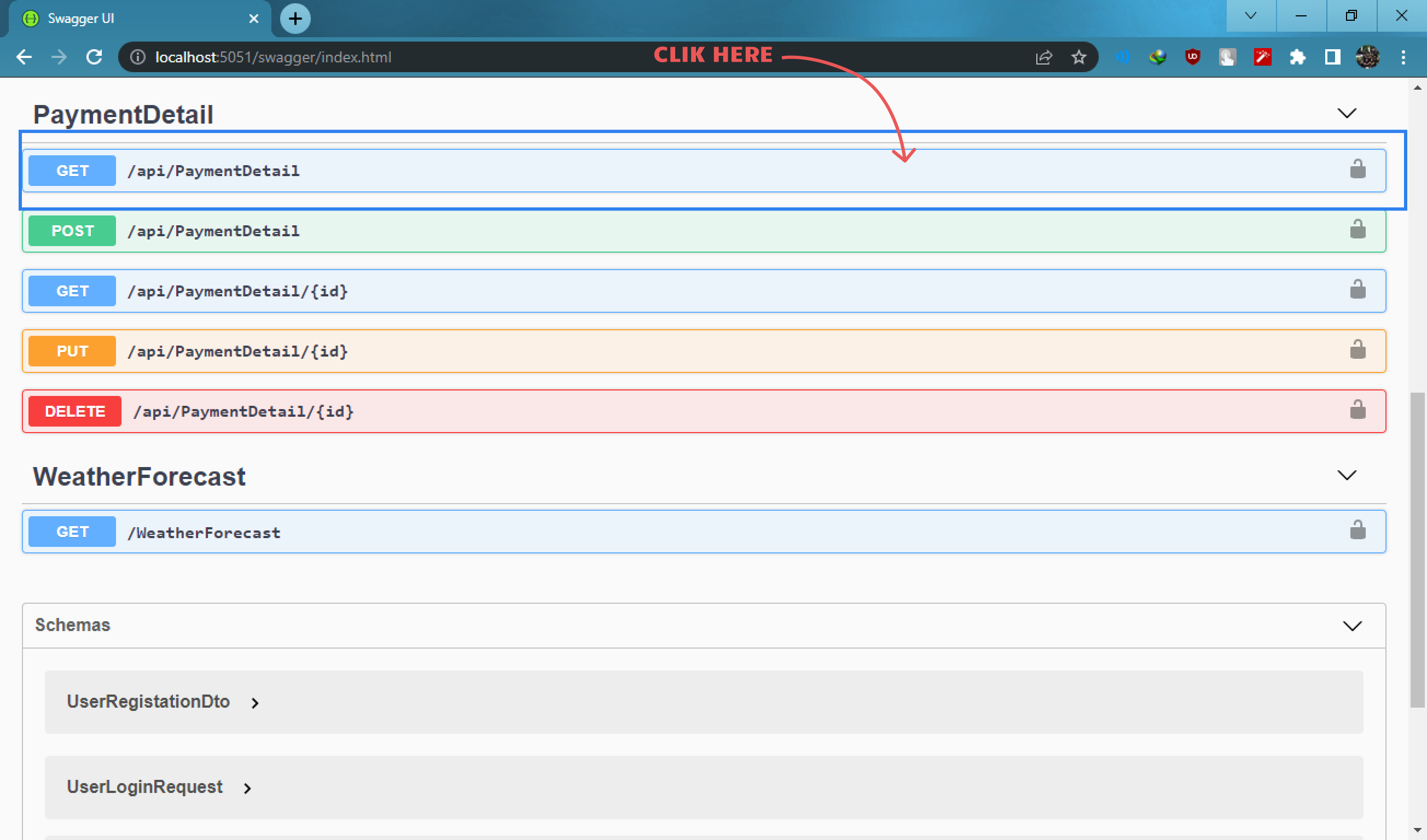
Note: ganti <token> dengan token yang didapatkan saal login



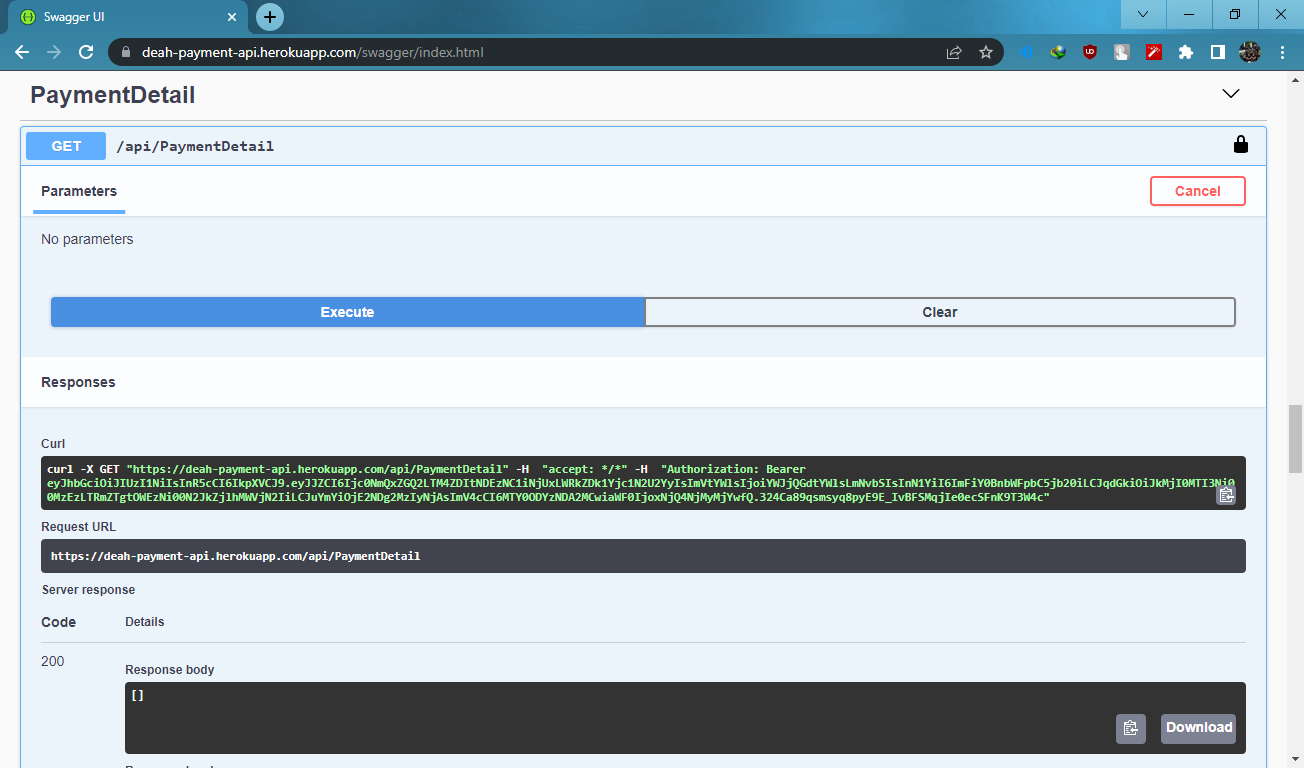
1. Lalu klik button Authorize untuk menerapkan token > lalu close

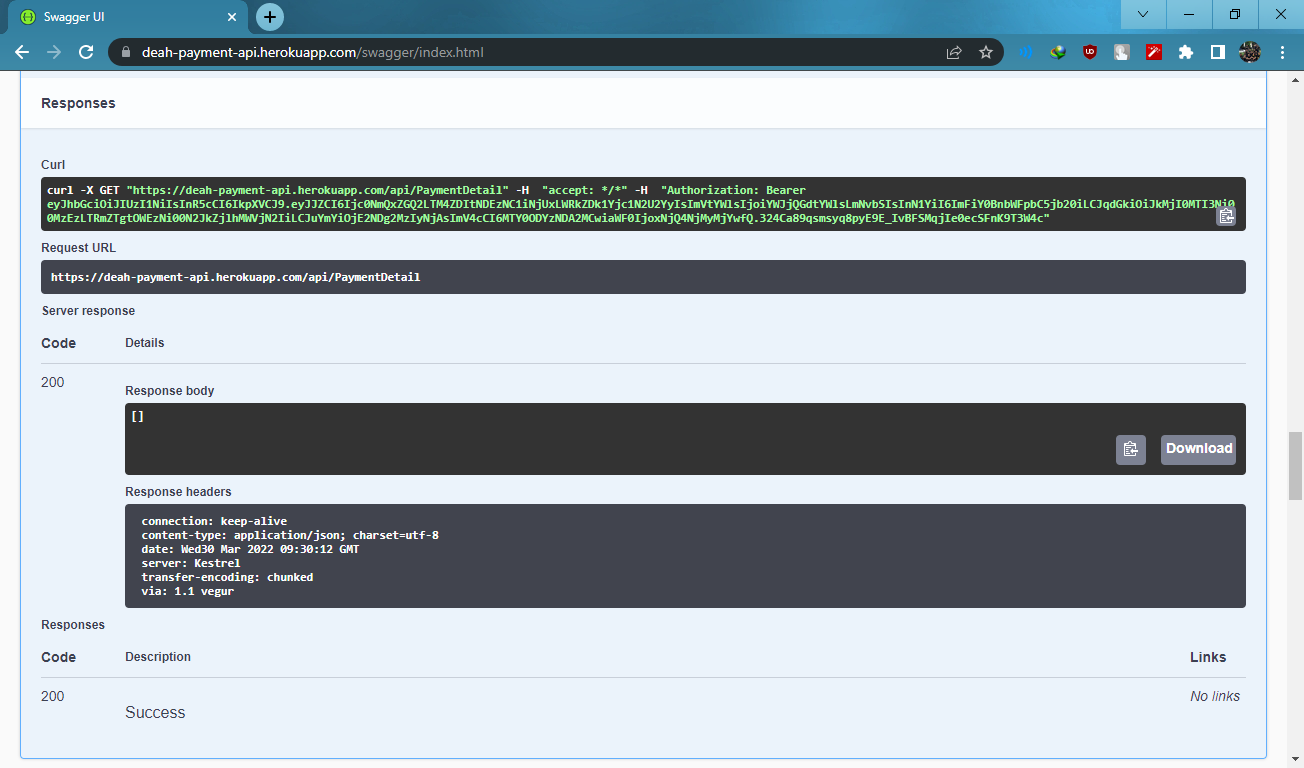
## GET – untuk mendapatkan data dengan jumlah banyak

1. Pastikan sudah memasukan token yang valid di button Authorize
2. Klik card untuk ‘GET: api/paymentDetail’

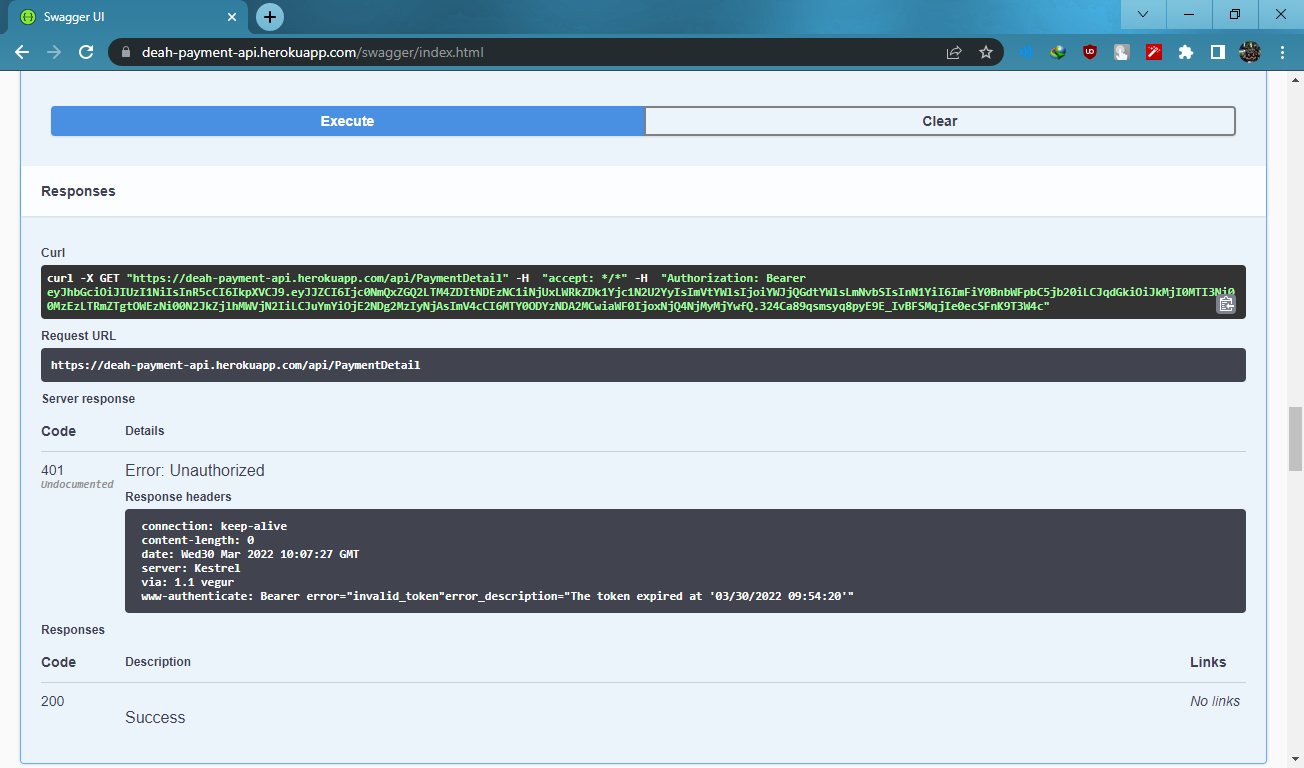


1. Berikut response apabila token valid. Response body berupa array kosong karena Data masih kosong



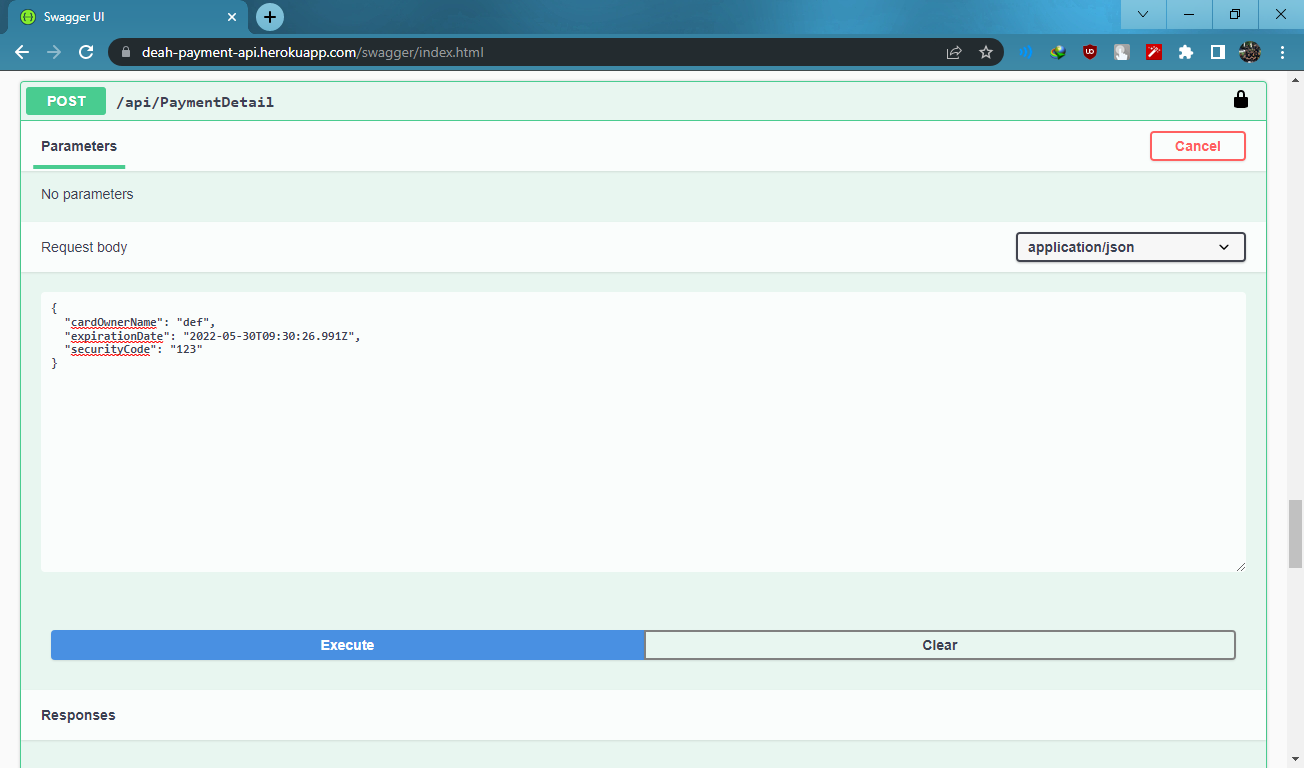


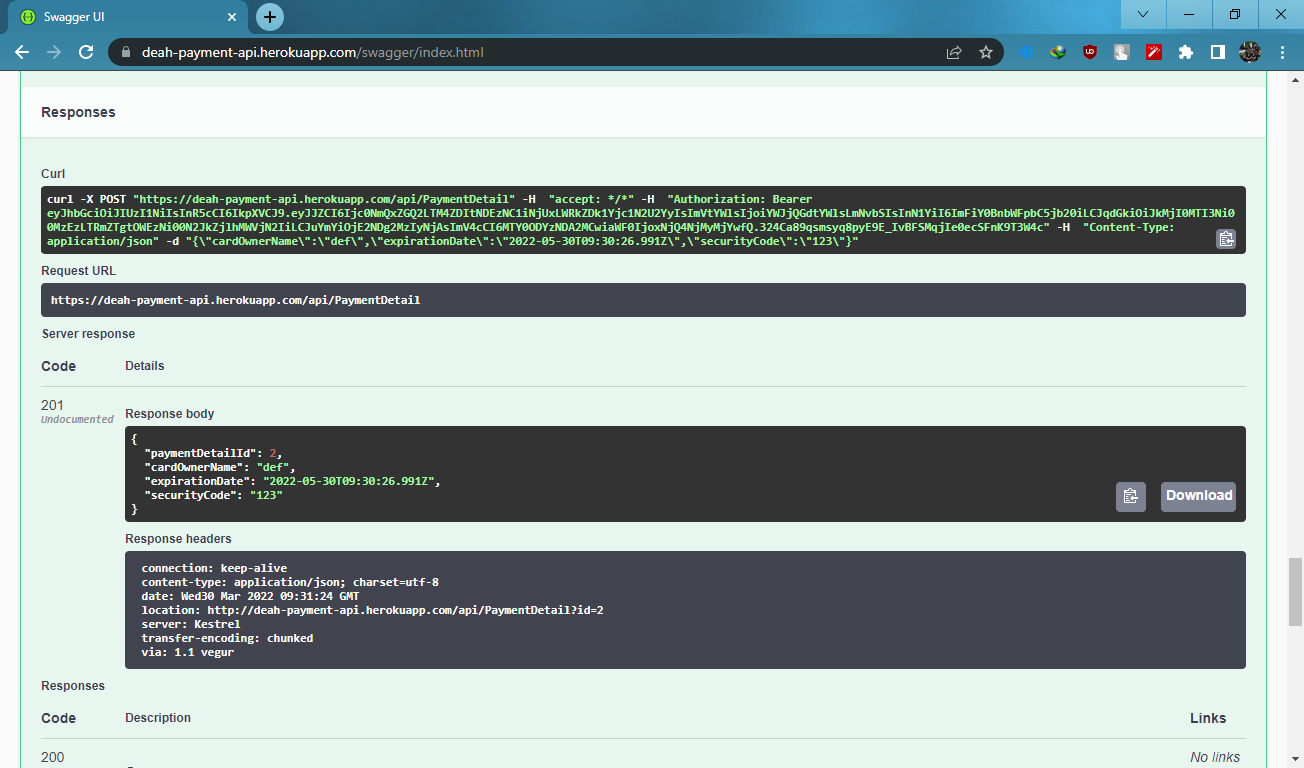
1. Berikut kalau Token tidak valid/expired



## POST – untuk melakukan insert

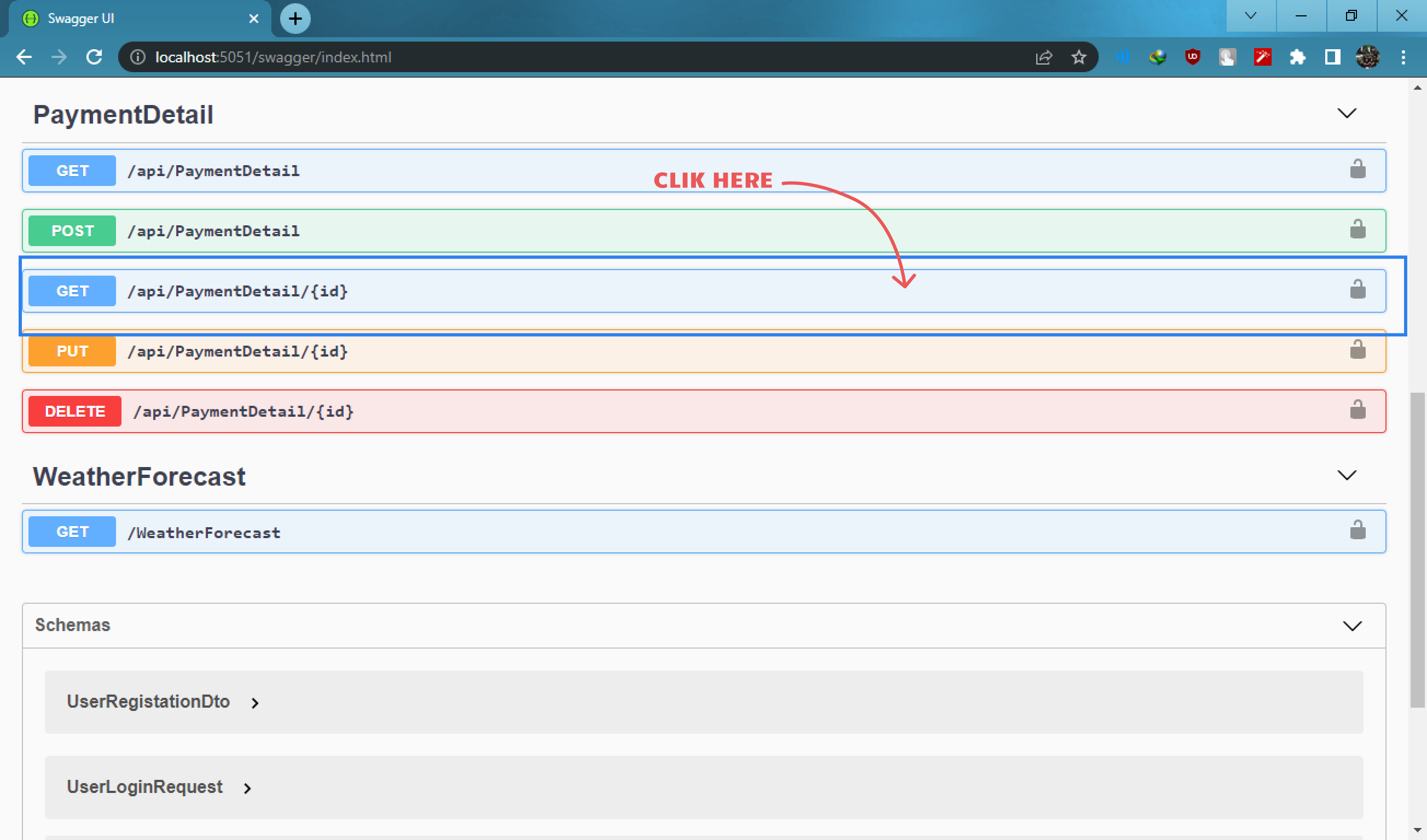
1. Pastikan sudah memasukan token yang valid di button Authorize
2. Klik card untuk ‘GET: api/paymentDetail’



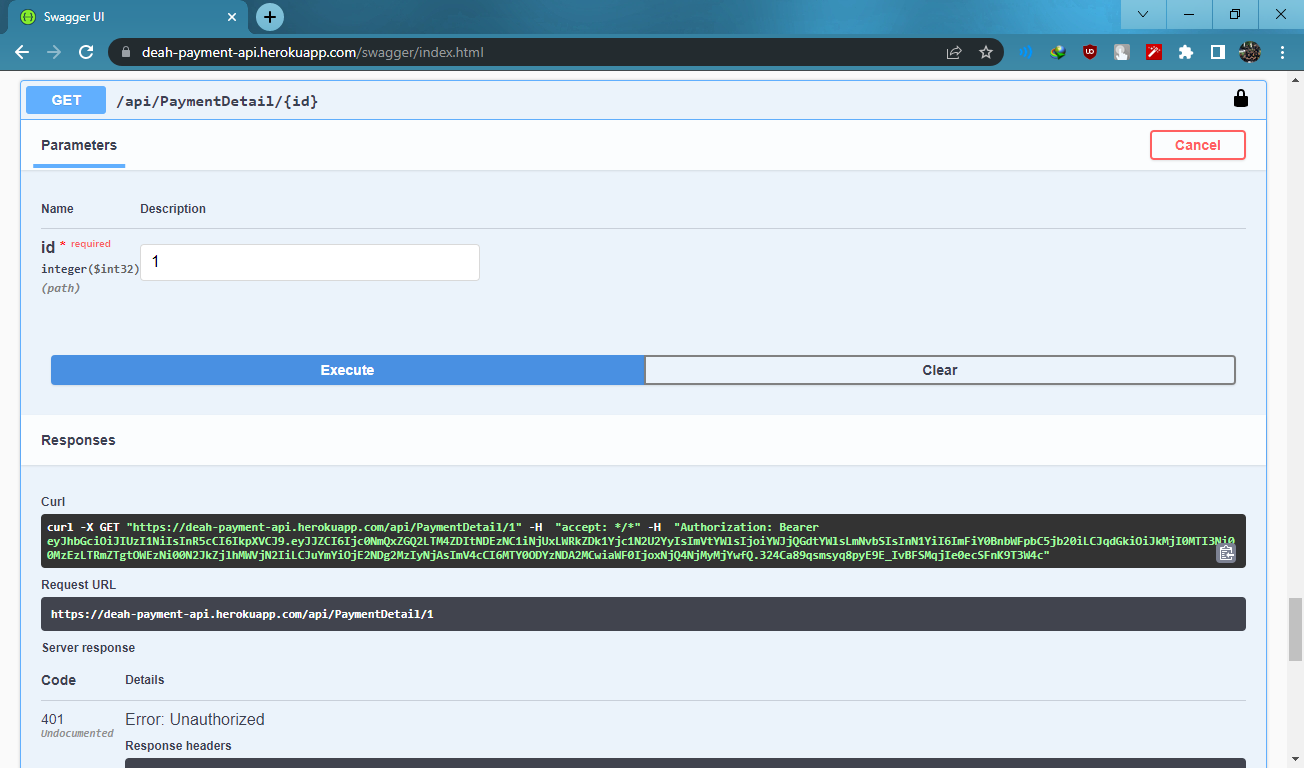


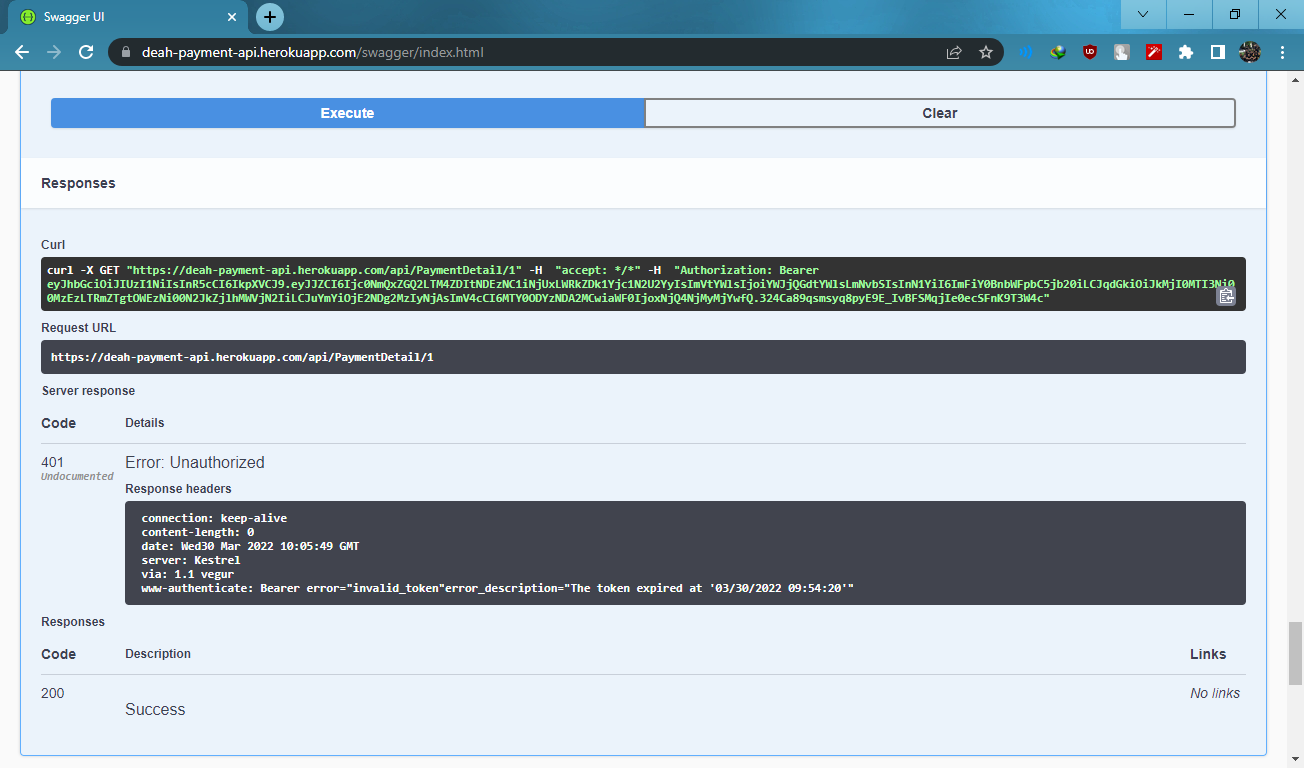
## GET{id} – untuk mendapatkan data by id

1. Pastikan sudah memasukan token yang valid di button Authorize
2. Klik card untuk ‘GET: api/paymentDetail/{id}’



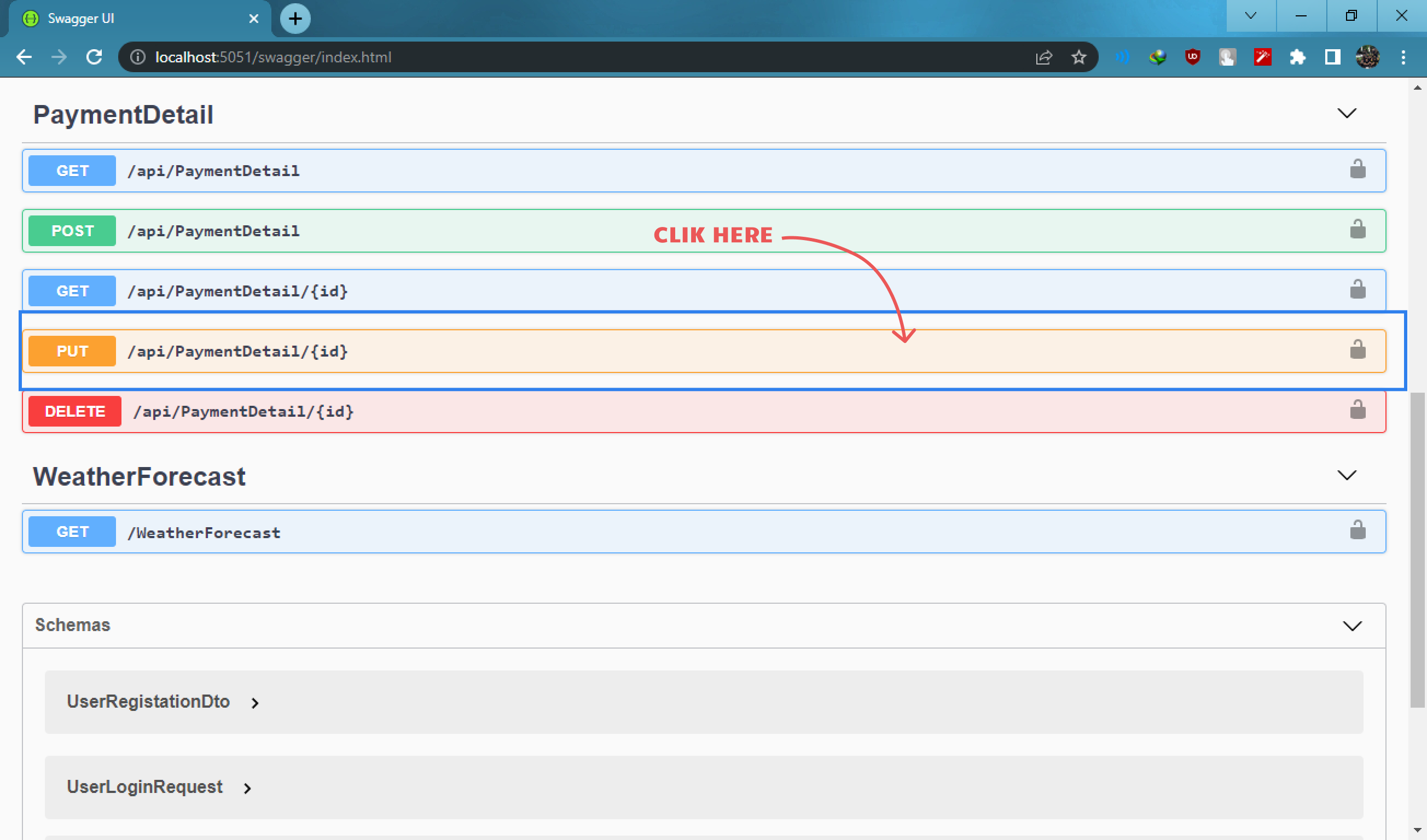
1. Klik try it out > masukan input Id > execute
2. Berikut response apabila token expired



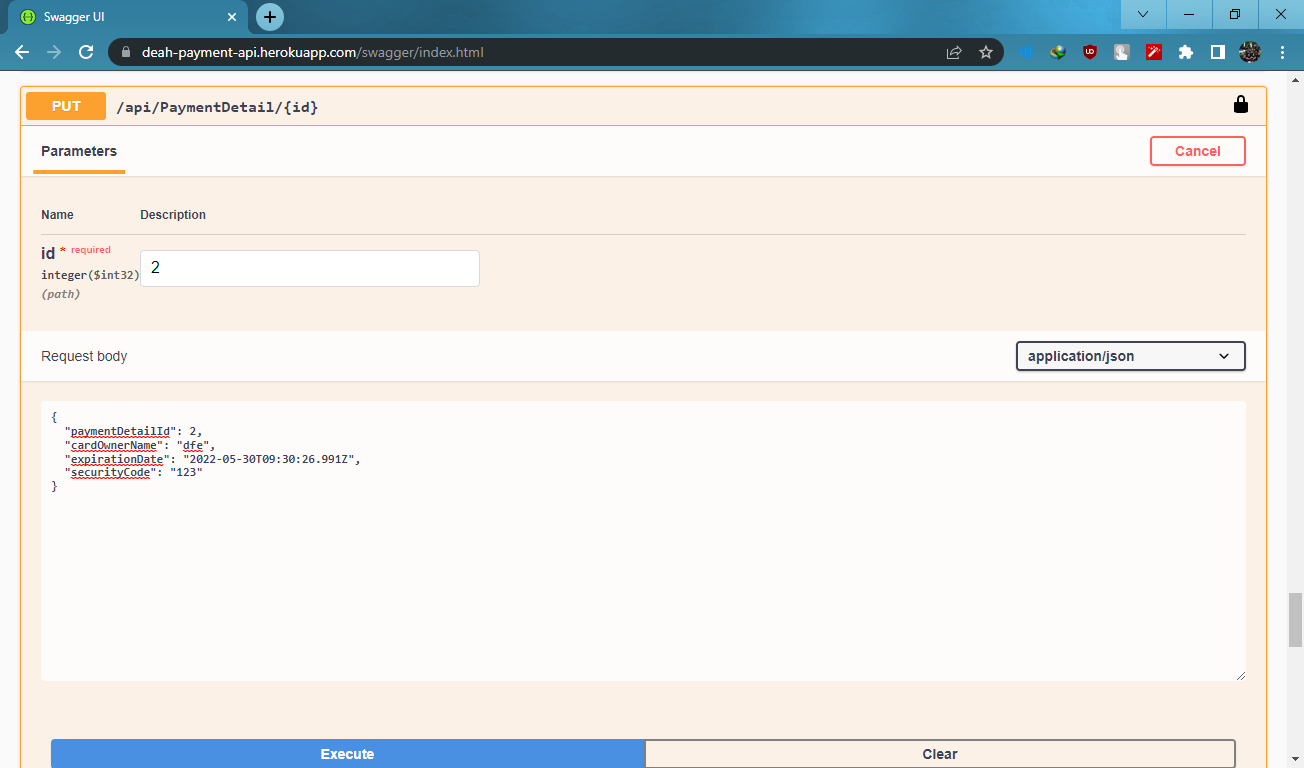


## PUT

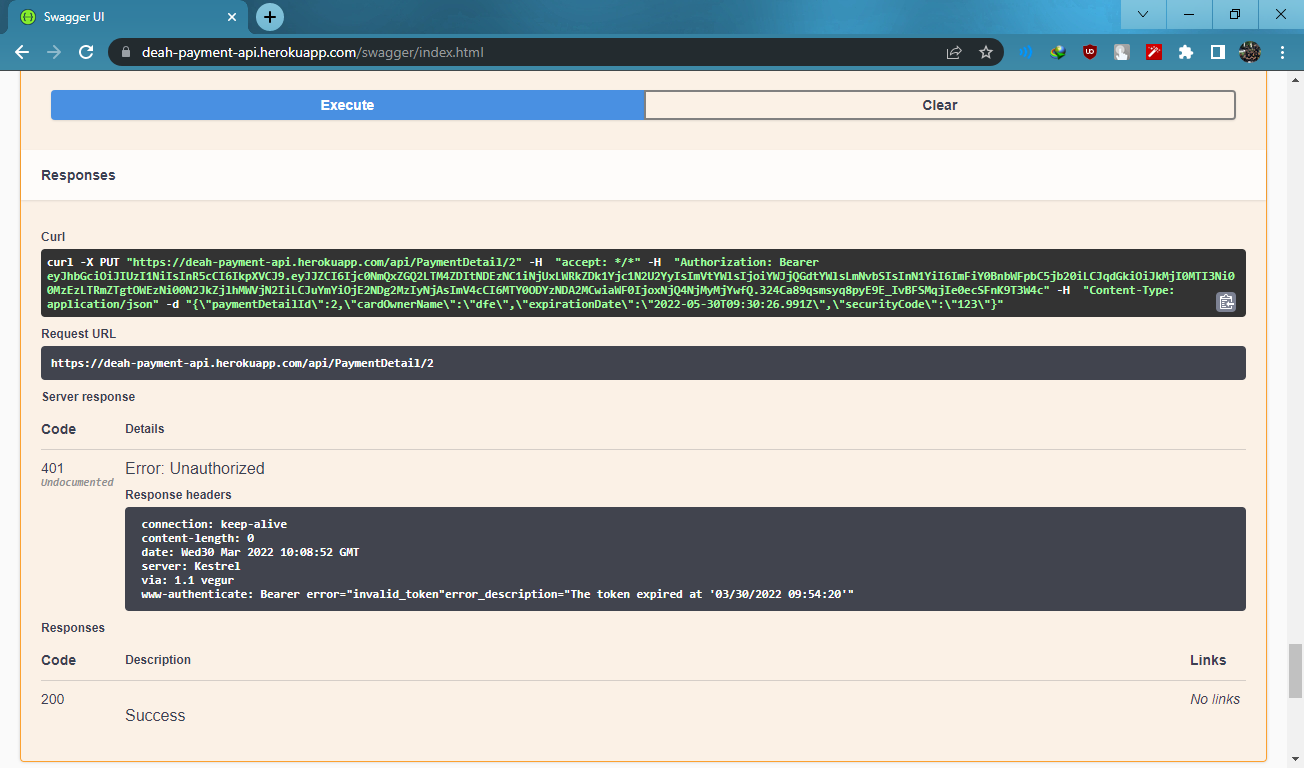
1. Pastikan sudah memasukan token yang valid di button Authorize
2. Klik card untuk ‘PUT: api/paymentDetail/{id}’



1. Masukan ‘value Id’ pada kolom ‘description’ > masukan pula input JSON pada kolom input body> execute

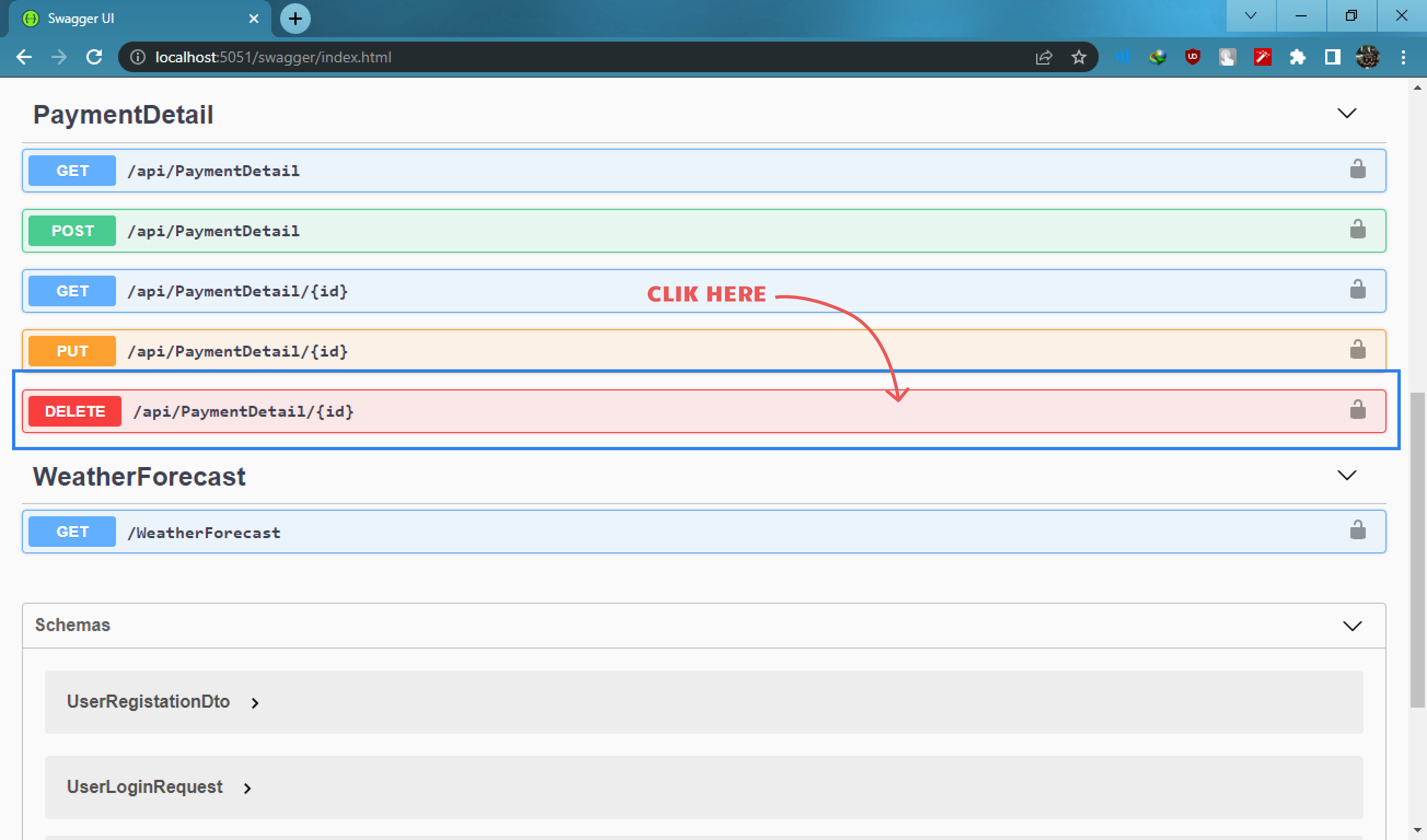


1. Response saat token expired

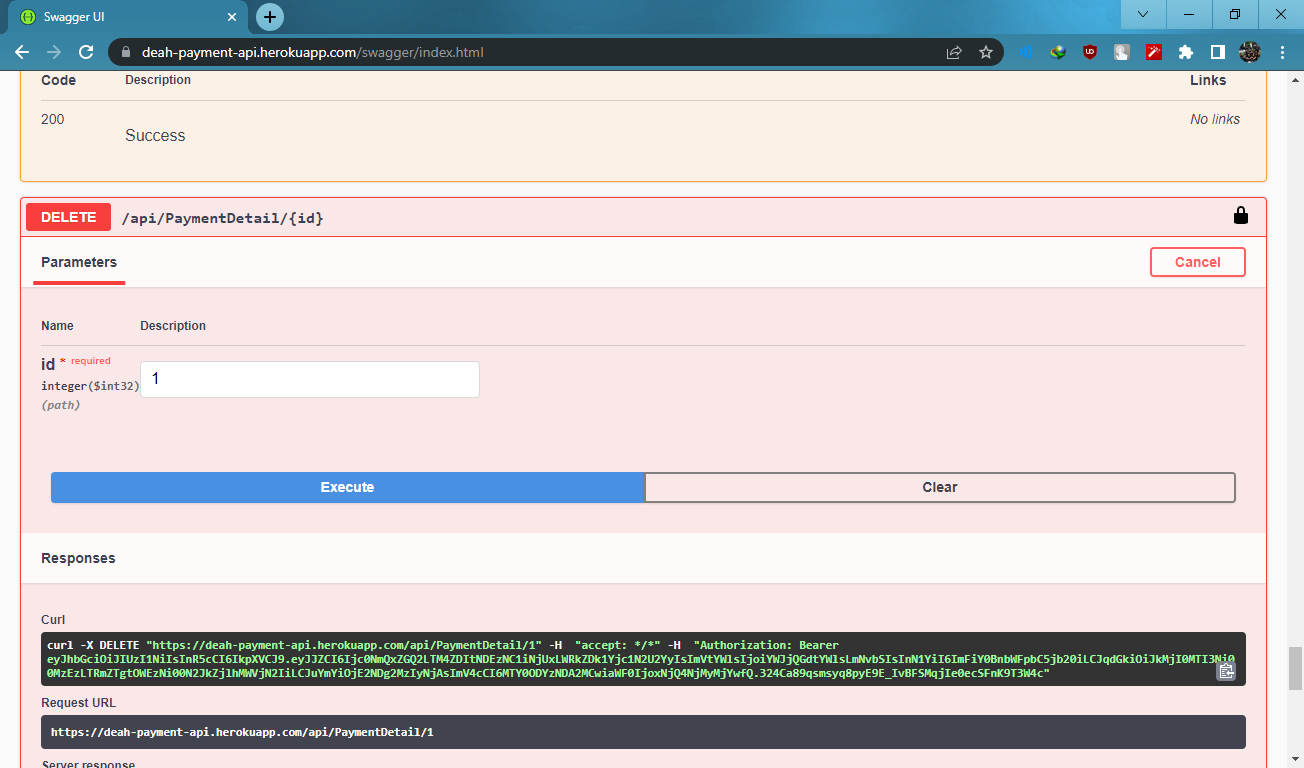


## DELETE

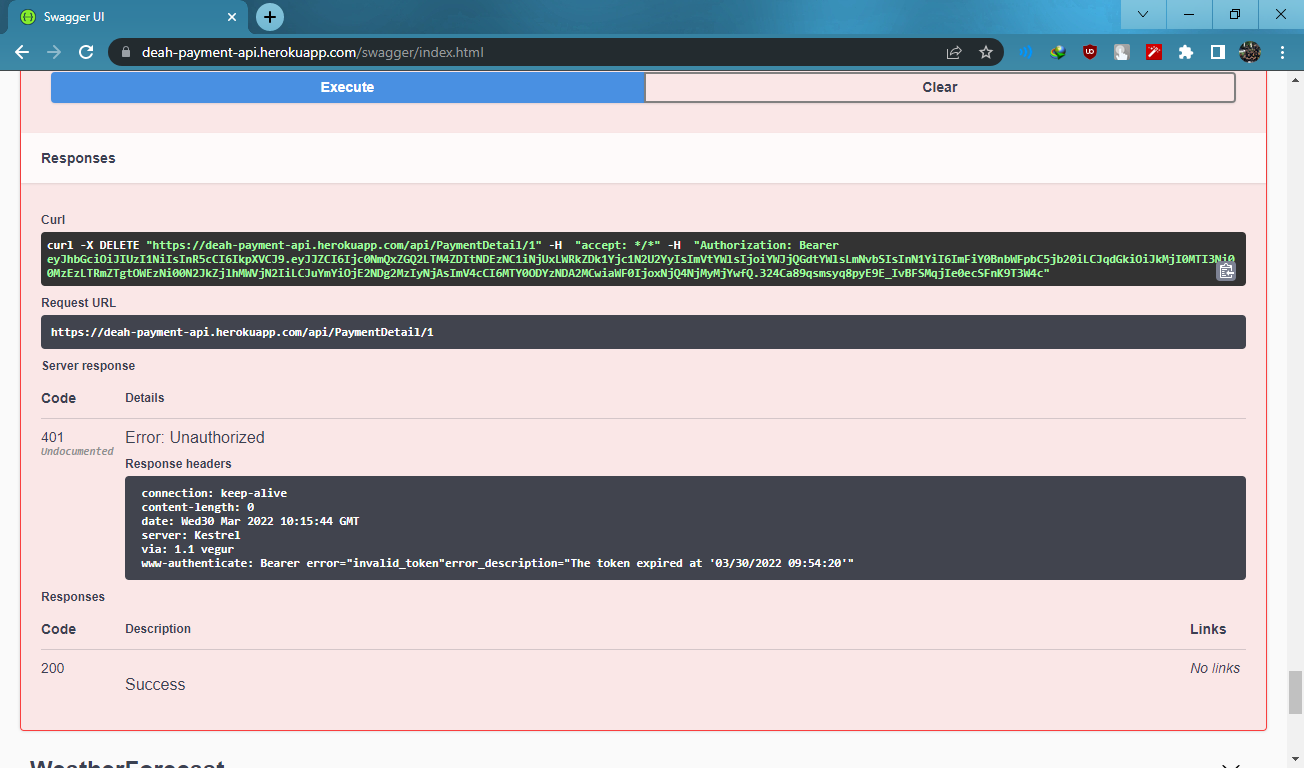
1. Pastikan sudah memasukan token yang valid di button Authorize
2. Klik card untuk ‘DELETE: api/paymentDetail/{id}’



1. Klik try it out > masukan input Id > execute



1. Berikut response saat token expired



# Push Github

1. Melalui Terminal (CMD) arahkan ke dalam folder project

D:

cd D:/<lokasi\_project>

1. Menjalankan peintah git

echo "# test" >> README.md

git init

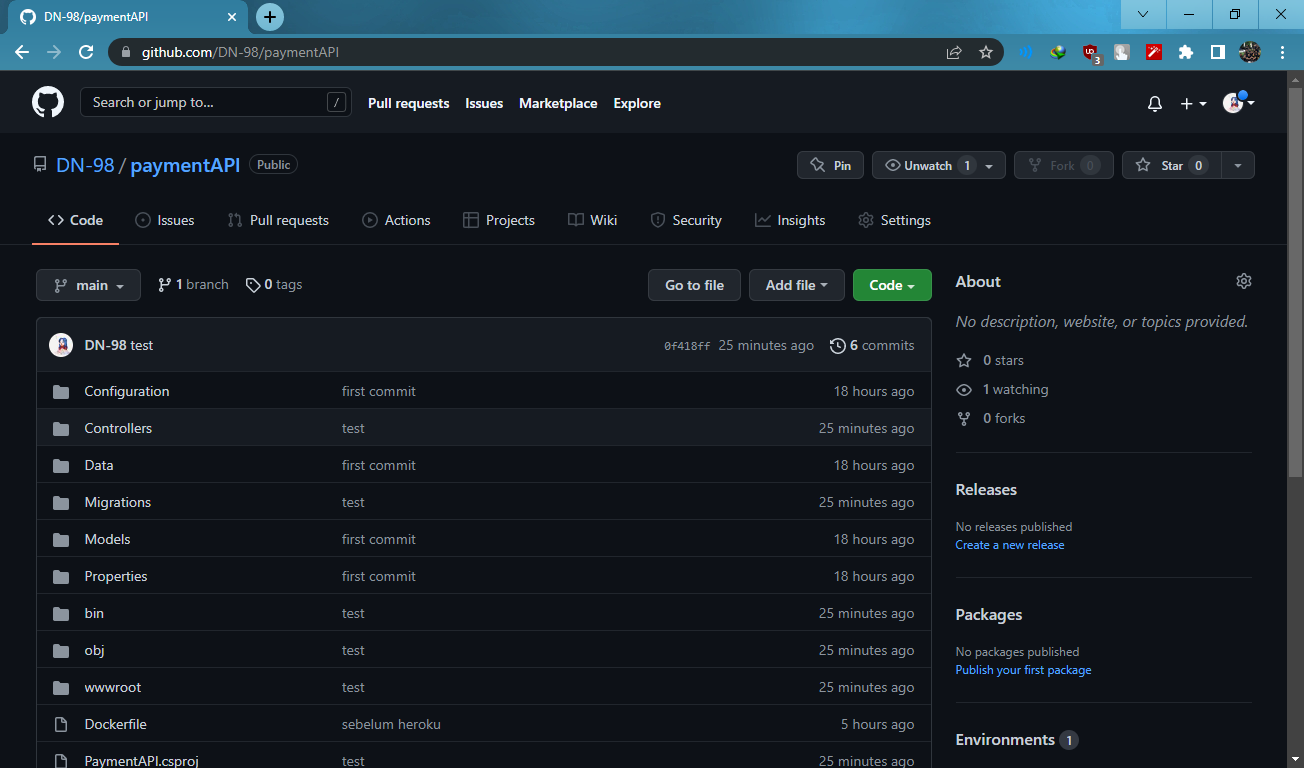
git add .

git commit -m "first commit"

git branch -M main

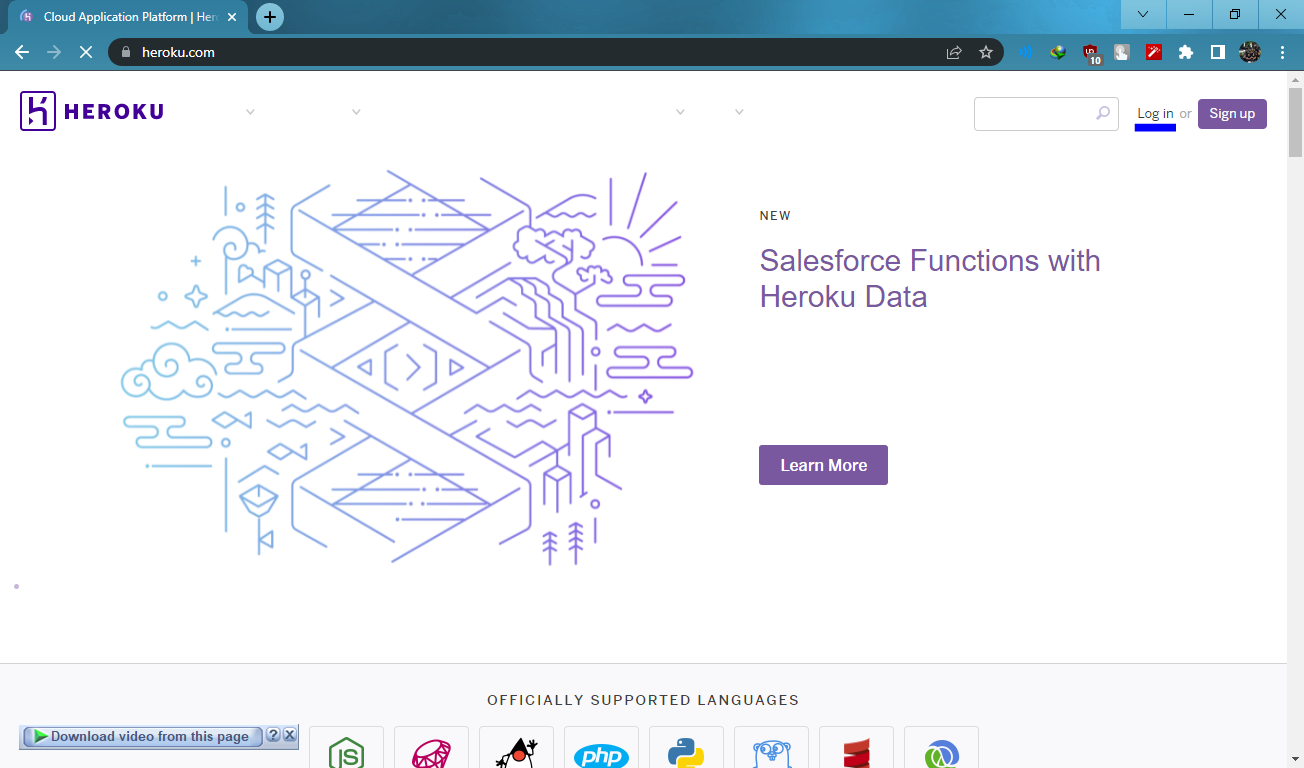
git remote add origin https://github.com/DN-98/paymentAPI.git

git push -u origin main

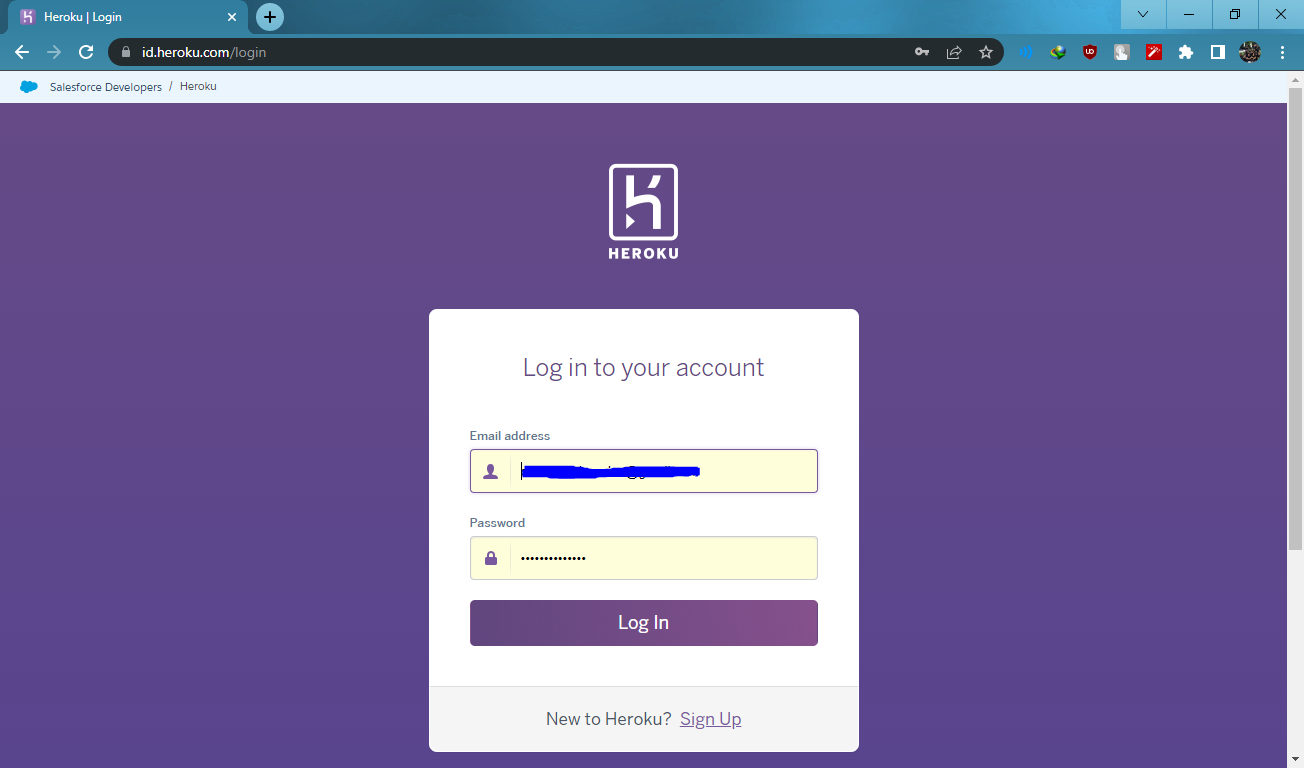


# Deploy Heroku

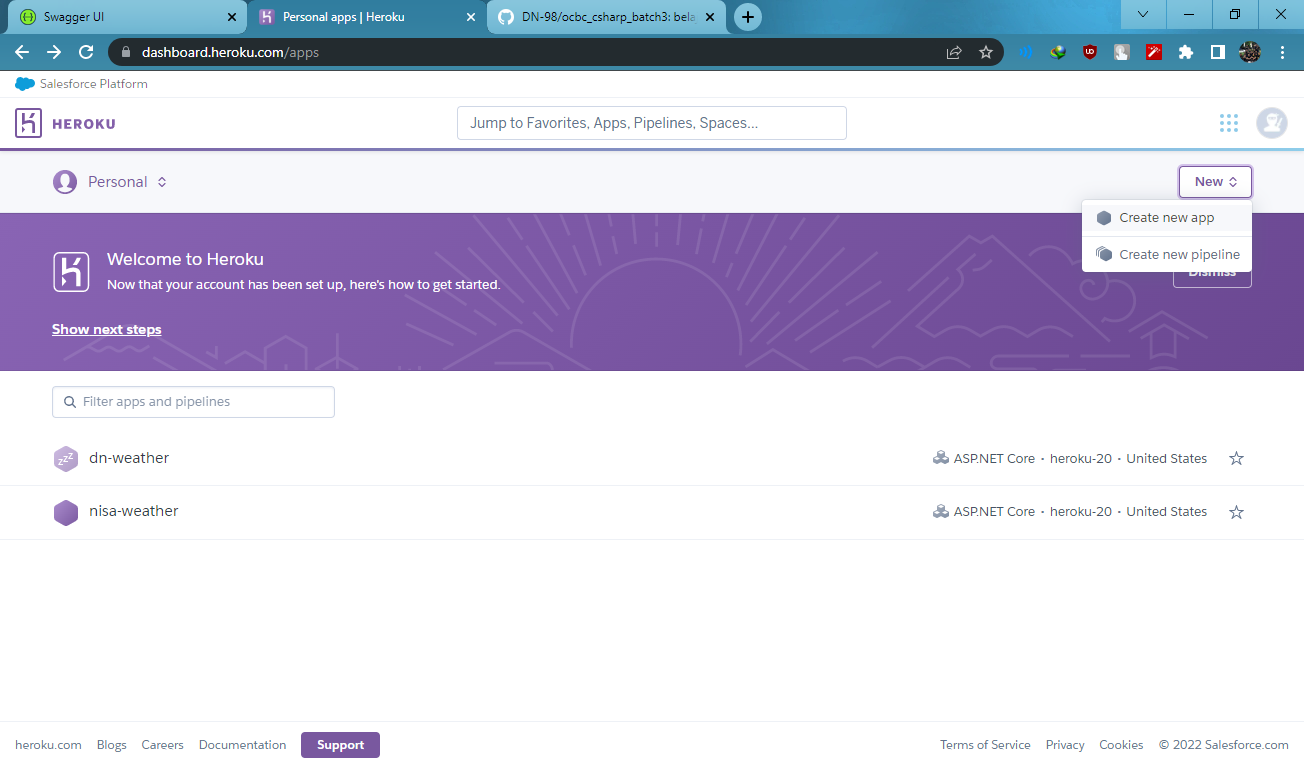
1. Buka heroku: <https://www.heroku.com/> > pada pojok kanan atas klik Log in



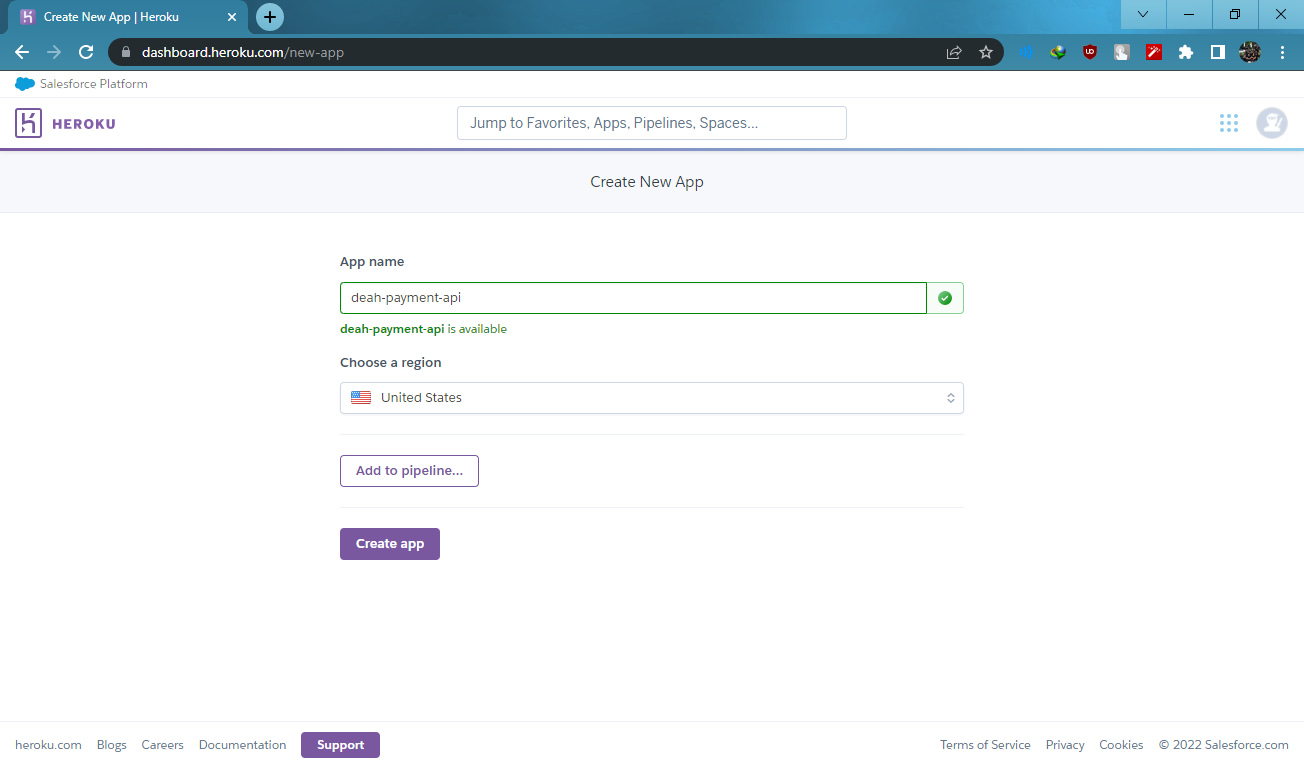
1. Masukan Email Address dan Password > Login



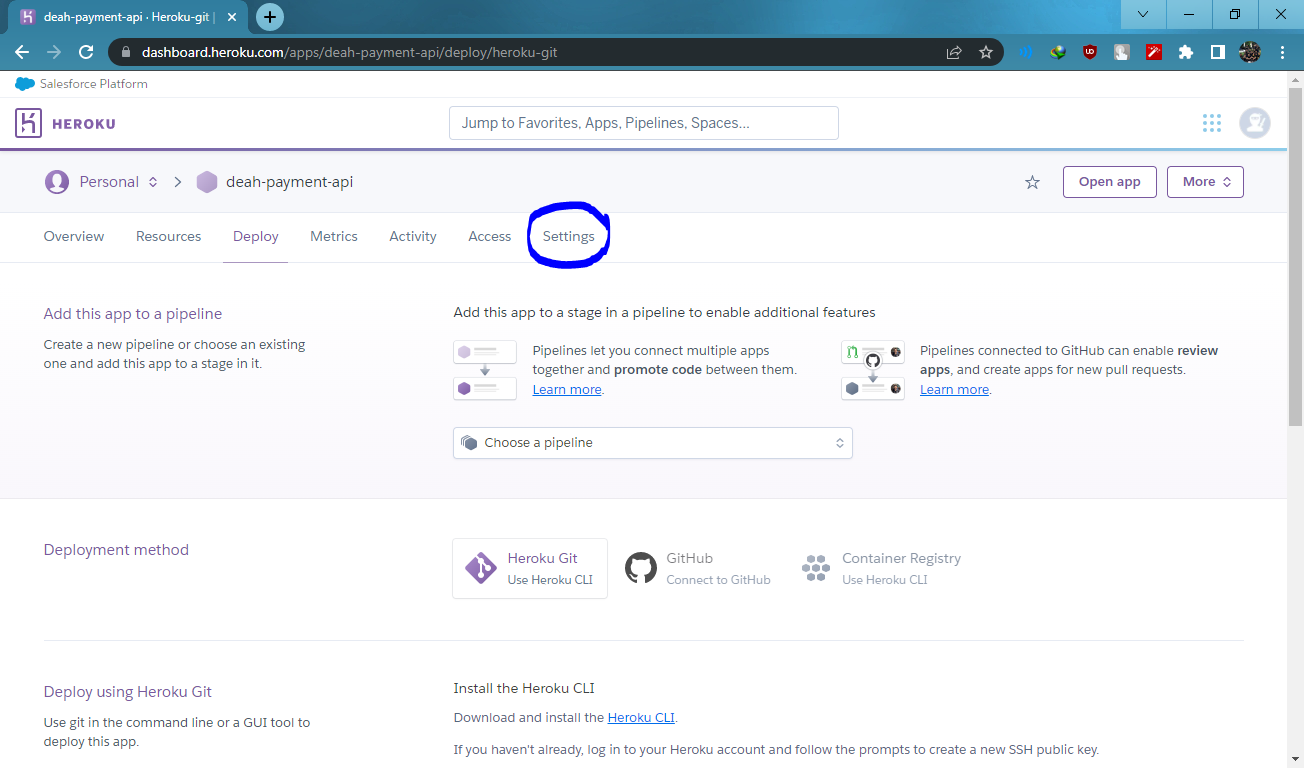
1. Pada pojok kanan atas klik Button New>Create New App



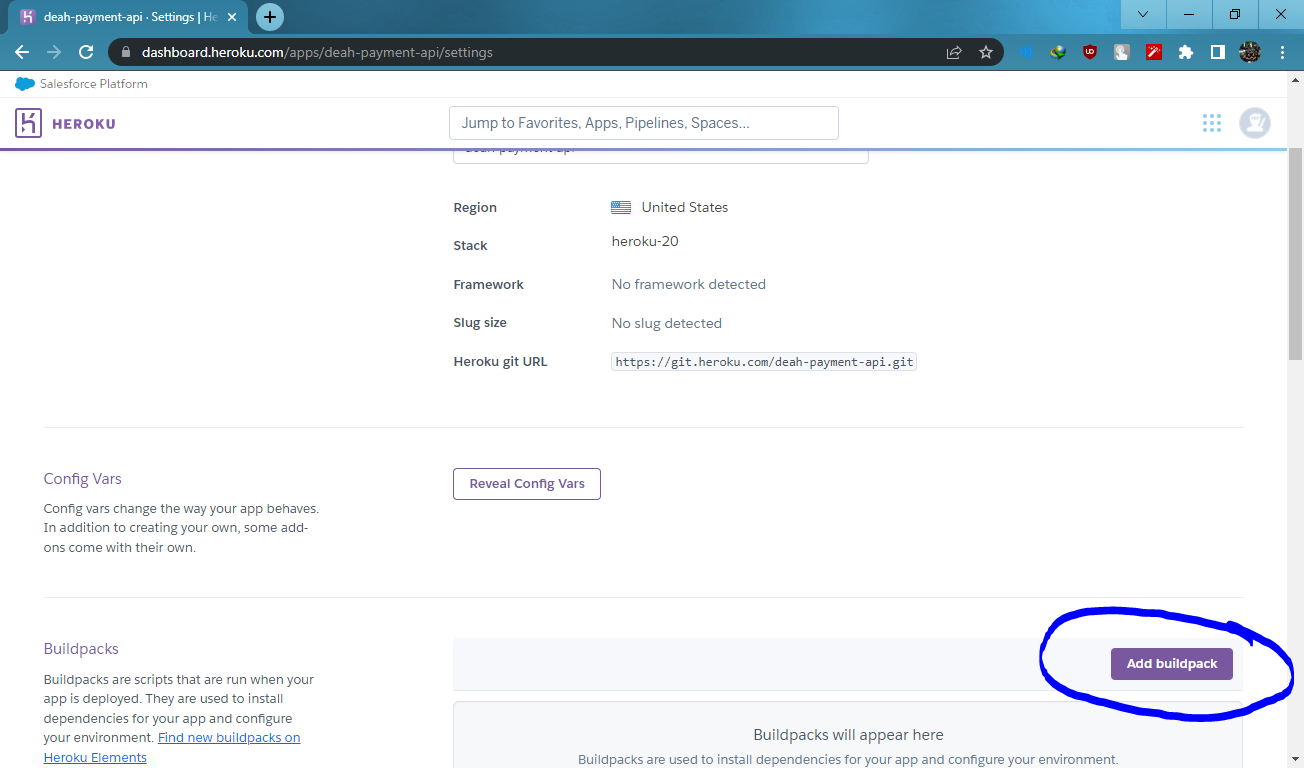
1. Masukan nama App> create app



1. Halaman setelah ‘create app’. Klik ‘Settings’

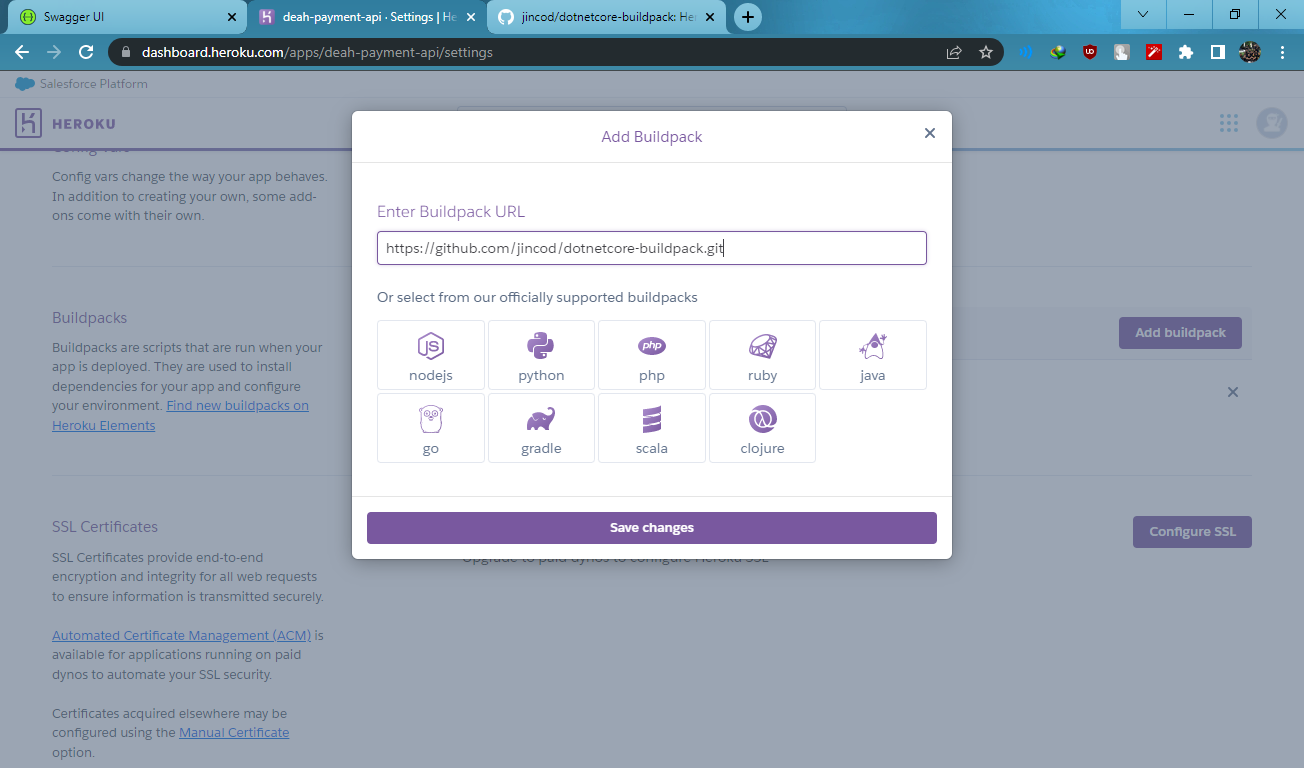


1. Scroll kebawah > klik button ‘Add buildpack’



1. Tambah kan link repo jincod > Save changes

<https://github.com/jincod/dotnetcore-buildpack.git>



1. Maka akan tampil seperti ini:

