

Nama : Dini Tri Widianingsih

No. Peserta : FSDO003ONL007

Link Github : [dinni26/DiniFinalProject_PaymentAPI \(github.com\)](https://github.com/dinni26/DiniFinalProject_PaymentAPI)

Link Heroku : [Biodata \(dini-finalproject.herokuapp.com\)](https://biodata.dini-finalproject.herokuapp.com/)

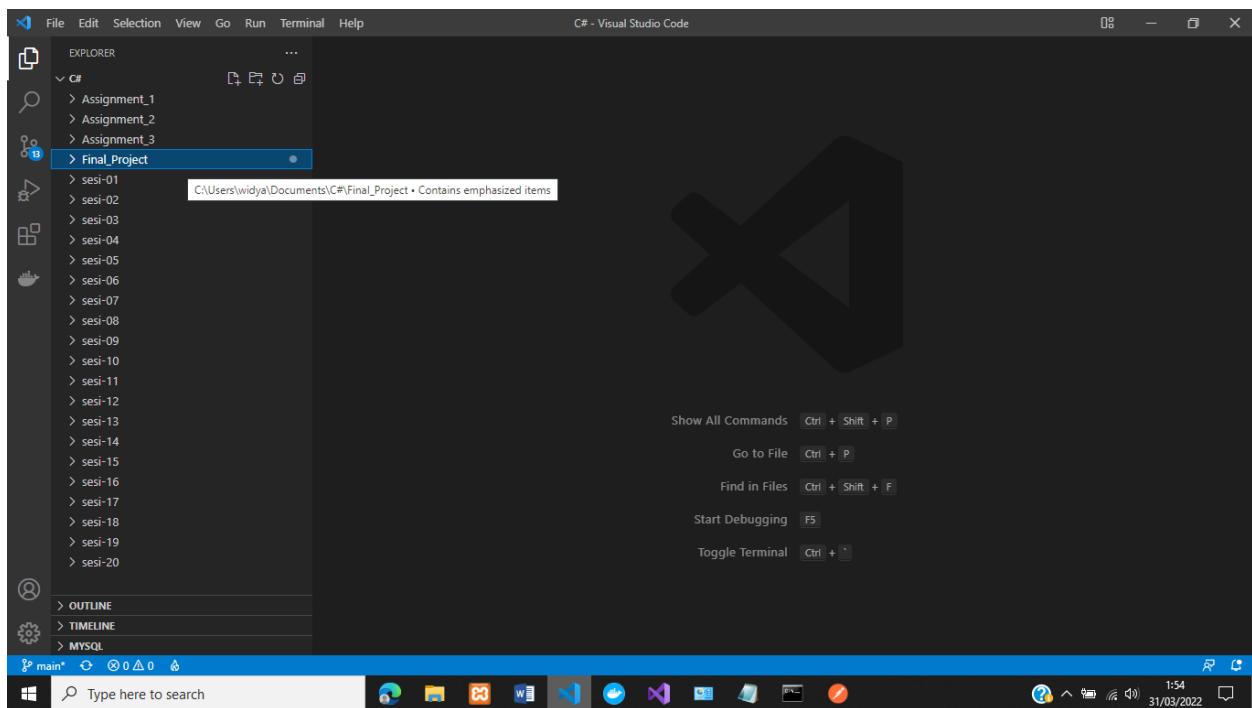
PANDUAN PEMBUATAN FINAL PROJECT

CREATE DATABASE

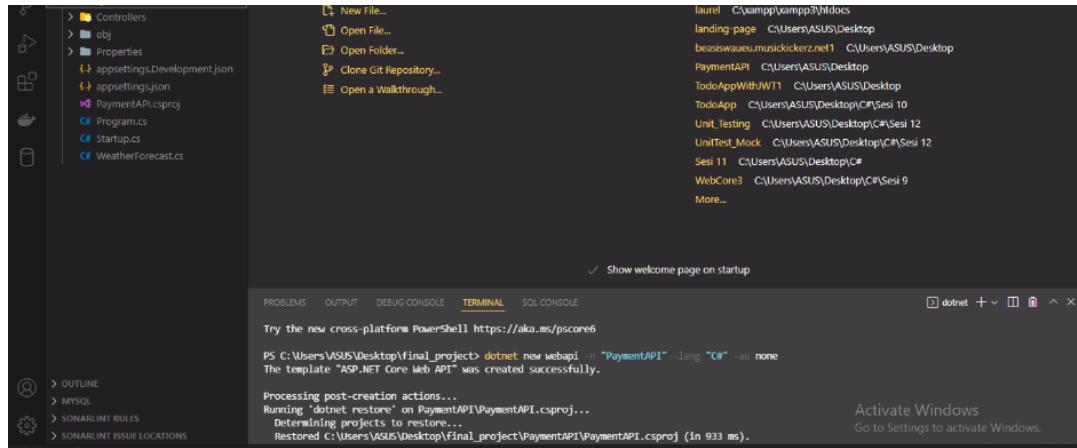
- Belum Online > Jalankan via cmd
CREATE database payment;
- Jika online, Silahkan buka link <https://remotemysql.com/phpmyadmin/>
Lalu masukan :
 - "server=remotemysql.com; port=3306; database=qfqRbkF42v;
username=qfqRbkF42v; password=ob08WIfgL9"

PEMBUATAN

- Buka Visual Studio Code > Create folder Final_Poject

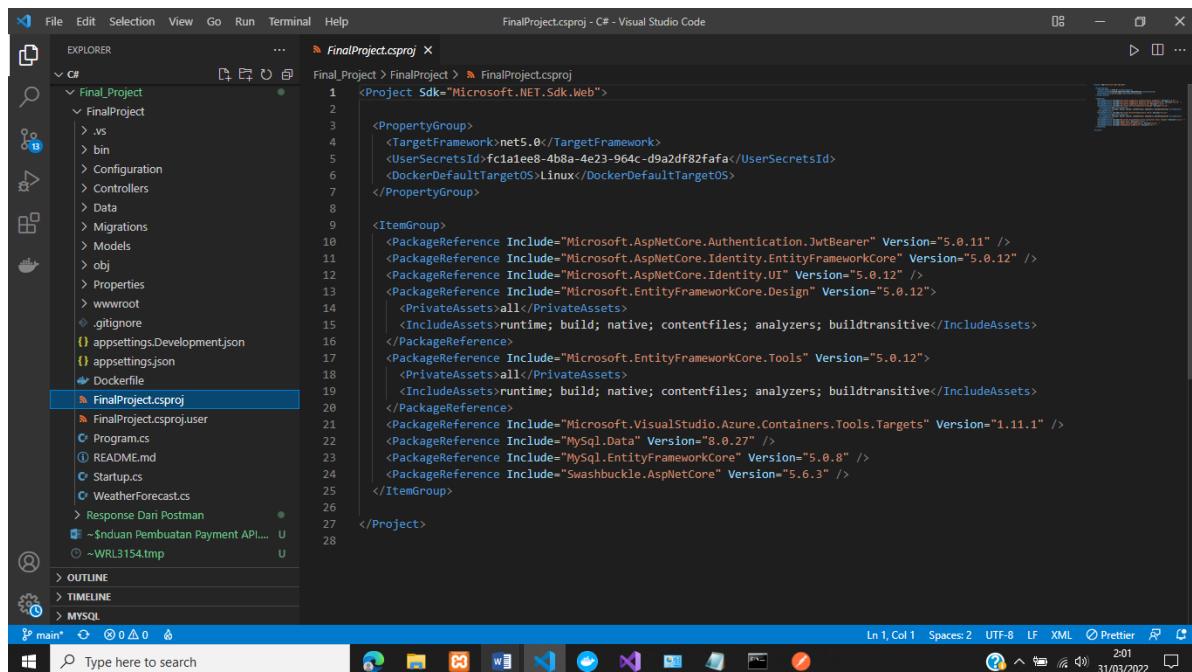


- Inisiasi project di dalam folder tersebut > **dotnet new webapi -n "Final_Project" -lang "C#" -au none** lalu akan mucul folder – folder baru

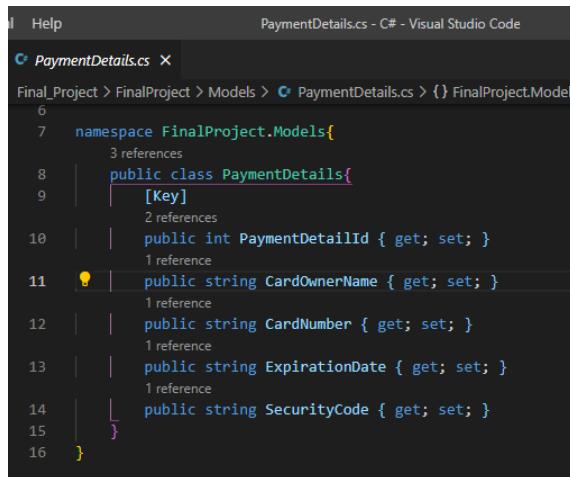


- Install package yang dibutuhkan

- dotnet add package NuGet.CommandLine.XPlat --version 5.11.0
- dotnet add package Microsoft.EntityFrameworkCore.Tools --version 5.0.11
- dotnet add package Microsoft.EntityFrameworkCore.Design --version 5.0.11
- dotnet add package Microsoft.AspNetCore.Authentication.JwtBearer --version 5.0.11
- dotnet add package Microsoft.AspNetCore.Identity.EntityFrameworkCore --version 5.0.11
- dotnet add package Microsoft.AspNetCore.Identity.UI --version 5.0.11
- dotnet add package Microsoft.MySql.Data --version 8.0.28
- dotnet add package Microsoft.MySql.EntityFrameworkCore --version 5.0.10



- Buat folder Models, lalu tambahkan file PaymentDetails.cs di dalam folder tersebut
Location : Final_Project/Models/PaymentDetails.cs



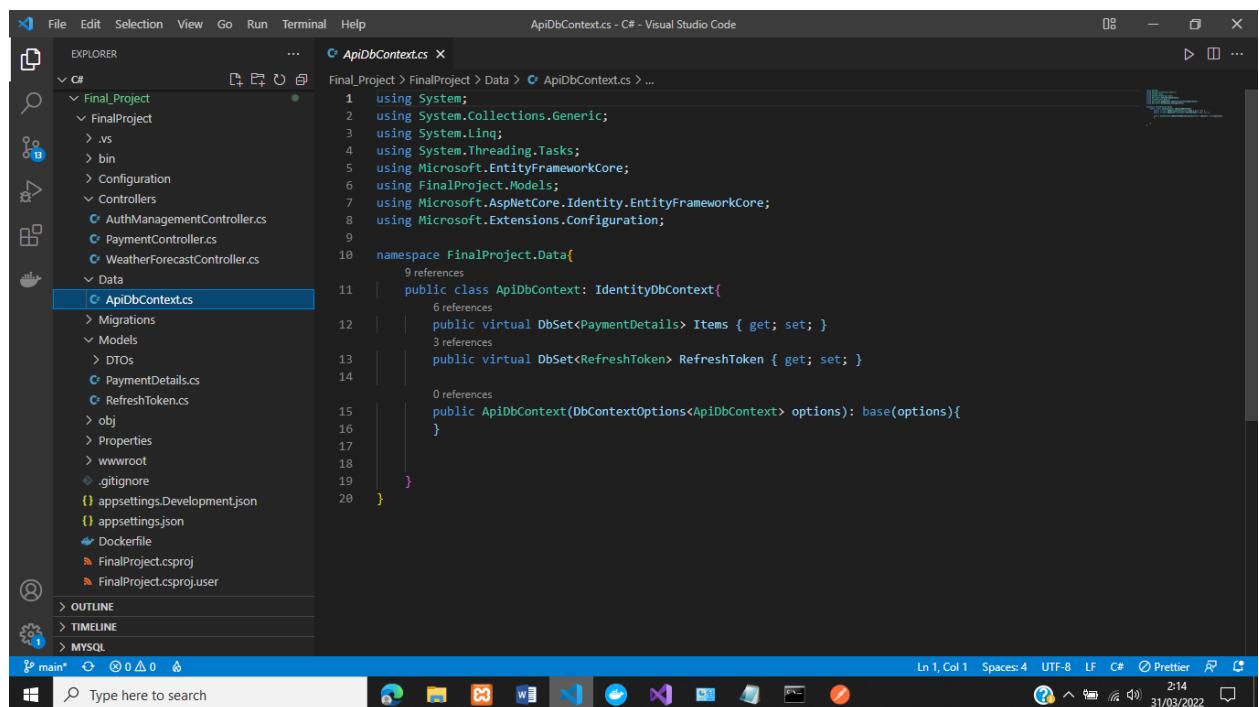
```

PaymentDetails.cs - C# - Visual Studio Code
PaymentDetails.cs

Final_Project > FinalProject > Models > PaymentDetails.cs > {} FinalProject.Models
6
7     namespace FinalProject.Models{
8         3 references
9             public class PaymentDetails{
10                 [Key]
11                 2 references
12                     public int PaymentDetailId { get; set; }
13                     1 reference
14                     public string CardOwnerName { get; set; }
15                     1 reference
16                     public string CardNumber { get; set; }
17                     1 reference
18                     public string ExpirationDate { get; set; }
19                     1 reference
20                     public string SecurityCode { get; set; }
21             }
22     }
23

```

- Buat folder Data, lalu tambahkan file ApiDbContext
Location : Final_Project/Data/ApiDbContext.cs



```

File Edit Selection View Go Run Terminal Help
ApiDbContext.cs - C# - Visual Studio Code
EXPLORER
Final_Project > FinalProject > Data > ApiDbContext.cs > ...
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Threading.Tasks;
5  using Microsoft.EntityFrameworkCore;
6  using FinalProject.Models;
7  using Microsoft.AspNetCore.Identity.EntityFrameworkCore;
8  using Microsoft.Extensions.Configuration;
9
10 namespace FinalProject.Data{
11     9 references
12     public class ApiDbContext: IdentityDbContext{
13         6 references
14             public virtual DbSet<PaymentDetails> Items { get; set; }
15             3 references
16             public virtual DbSet<RefreshToken> RefreshToken { get; set; }
17             0 references
18             public ApiDbContext(DbContextOptions<ApiDbContext> options): base(options){
19             }
20     }

```

- Buat folder Controllers, lalu lanjutkan build PaymentDetailController untuk method post, get, put, delete

Location : Final_Project/Controllers/PaymentController.cs

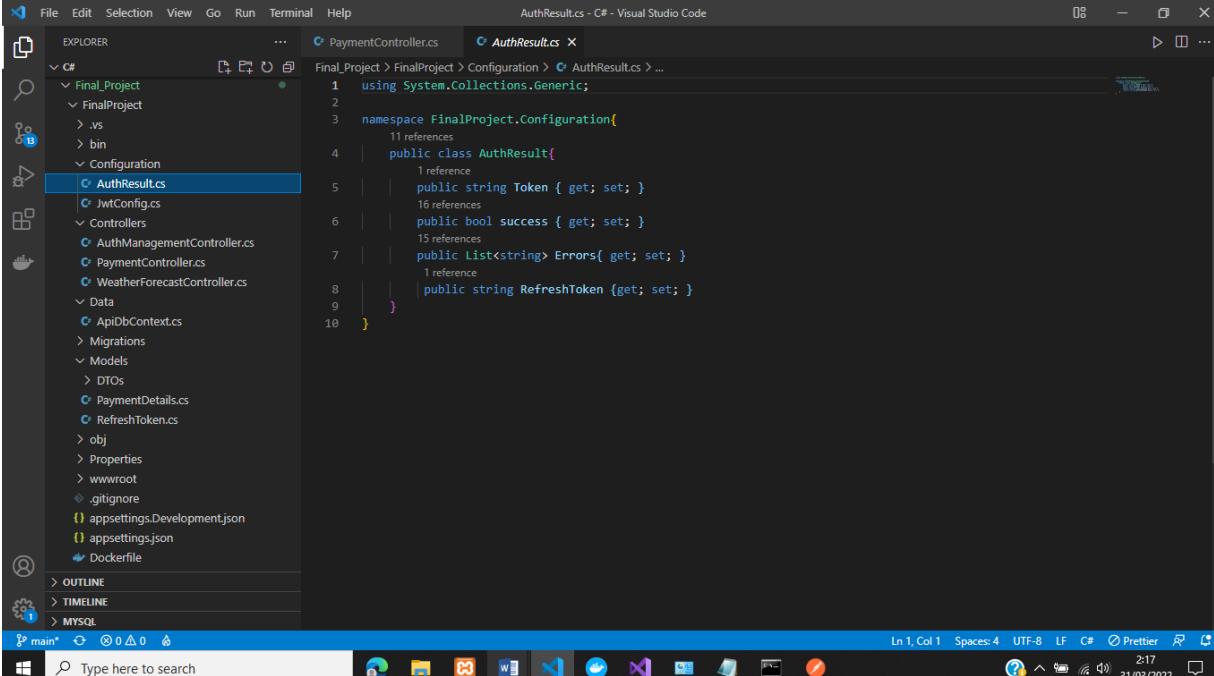
The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer (Left):** Shows the project structure under "Final_Project". The "Controllers" folder is expanded, showing "AuthManagementController.cs", "PaymentController.cs", and "WeatherForecastController.cs". Other files like "ApiDbContext.cs", "PaymentDetails.cs", and "RefreshToken.cs" are also visible.
- Code Editor (Center):** Displays the "PaymentController.cs" file content. The code defines a class "PaymentController" that inherits from "ControllerBase". It includes methods for "GetItem" (HttpGet) and "CreateItem" (HttpPost). The "CreateItem" method uses ModelState.IsValid to validate data before adding it to the database context and saving changes.
- Bottom Status Bar:** Shows "Ln 1, Col 1" and "Spaces: 4" along with other standard status bar icons.

```
PaymentController.cs - C# - Visual Studio Code
Final_Project > FinalProject > Controllers > PaymentController.cs > ...
21
22 namespace FinalProject.Controllers{
23
24     [Route("api/[controller]")]
25     [ApiController]
26     [Authorize(AuthenticationSchemes = JwtBearerDefaults.AuthenticationScheme)]
27
28     public class PaymentController: ControllerBase{
29         private readonly ApiDbContext _context;
30
31         public PaymentController(ApiDbContext context){
32             this._context = context;
33         }
34
35         [HttpGet]
36         public async Task<IActionResult> GetItem(){
37             var items = await _context.Items.ToListAsync();
38             return OK(items);
39         }
40
41         [HttpPost]
42         public async Task<IActionResult> CreateItem(PaymentDetails data){
43             if(ModelState.IsValid){
44                 await _context.Items.AddAsync(data);
45                 await _context.SaveChangesAsync();
46             }
47             return CreatedAtAction("GetItem", new {data.PaymentDetailId}, data);
48         }
}
```

Penambahan JWT Token dan Refresh Token

- Buat folder Configuration, lalu tambahkan file AuthResult.cs dan JwtConfig.cs:
Location : Final_Project/Configuration/AuthResult.cs



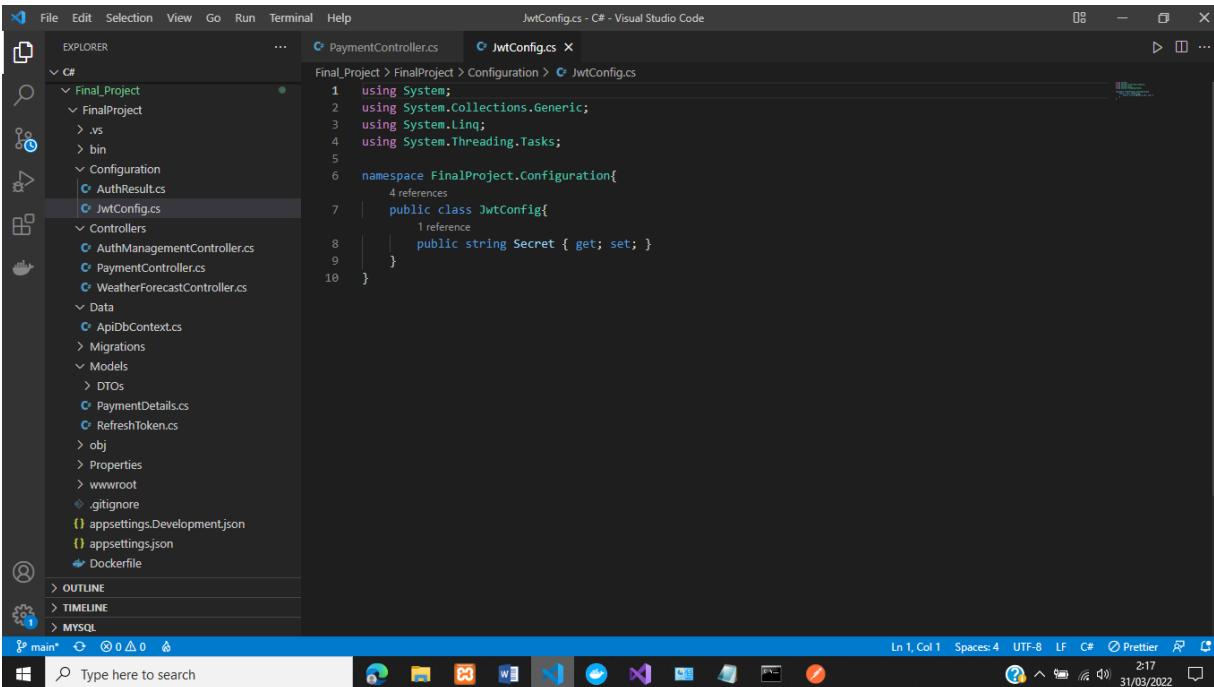
The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure under "Final_Project". The "Configuration" folder is expanded, showing "AuthResult.cs" and "JwtConfig.cs".
- Editor:** The "AuthResult.cs" file is open, displaying the following C# code:

```
1 using System.Collections.Generic;
2
3 namespace FinalProject.Configuration{
4     public class AuthResult{
5         public string Token { get; set; }
6         public bool success { get; set; }
7         public List<string> Errors{ get; set; }
8         public string RefreshToken { get; set; }
9     }
10 }
```

The status bar at the bottom indicates "Ln 1, Col 1" and "217 31/03/2022".

Location : Final_Project/Configuration/JwtConfig.cs



The screenshot shows the Visual Studio Code interface with the following details:

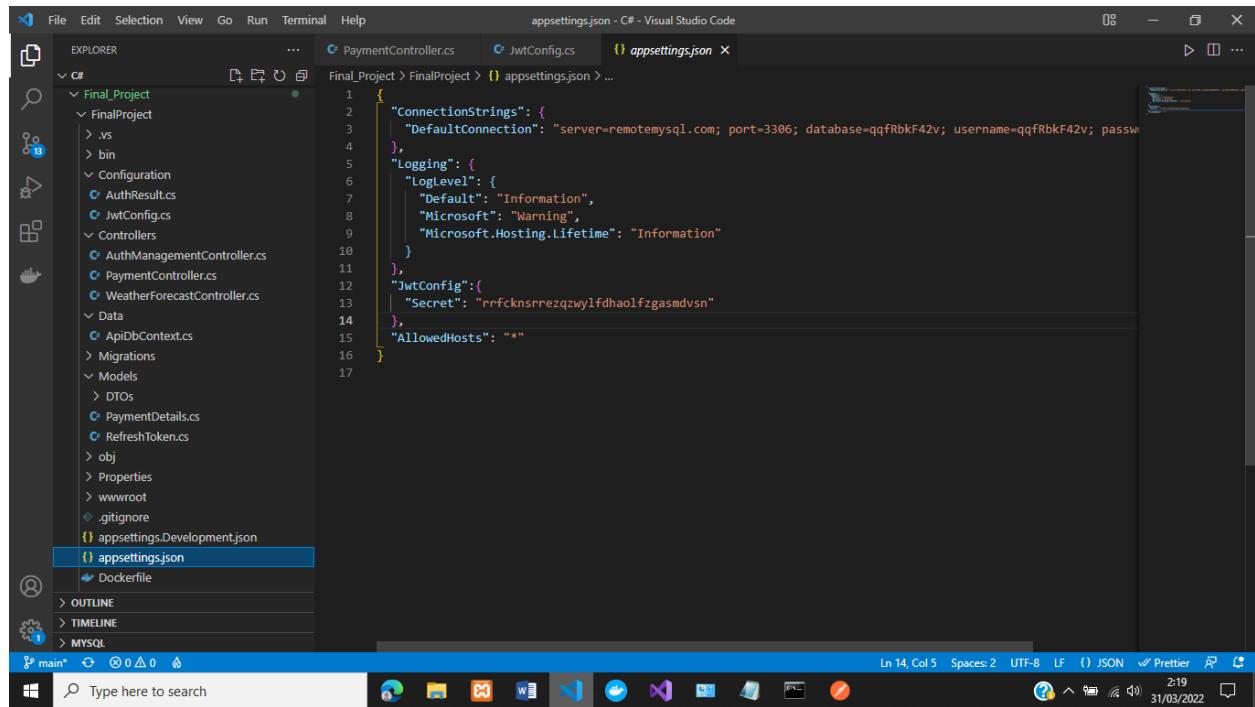
- File Explorer:** Shows the project structure under "Final_Project". The "Configuration" folder is expanded, showing "AuthResult.cs" and "JwtConfig.cs".
- Editor:** The "JwtConfig.cs" file is open, displaying the following C# code:

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Threading.Tasks;
5
6 namespace FinalProject.Configuration{
7     public class JwtConfig{
8         public string Secret { get; set; }
9     }
10 }
```

The status bar at the bottom indicates "Ln 1, Col 1" and "217 31/03/2022".

- Pada file appsettings.json tambahkan JwtConfig.Secrets

Location : Final_Project/appsettings.json



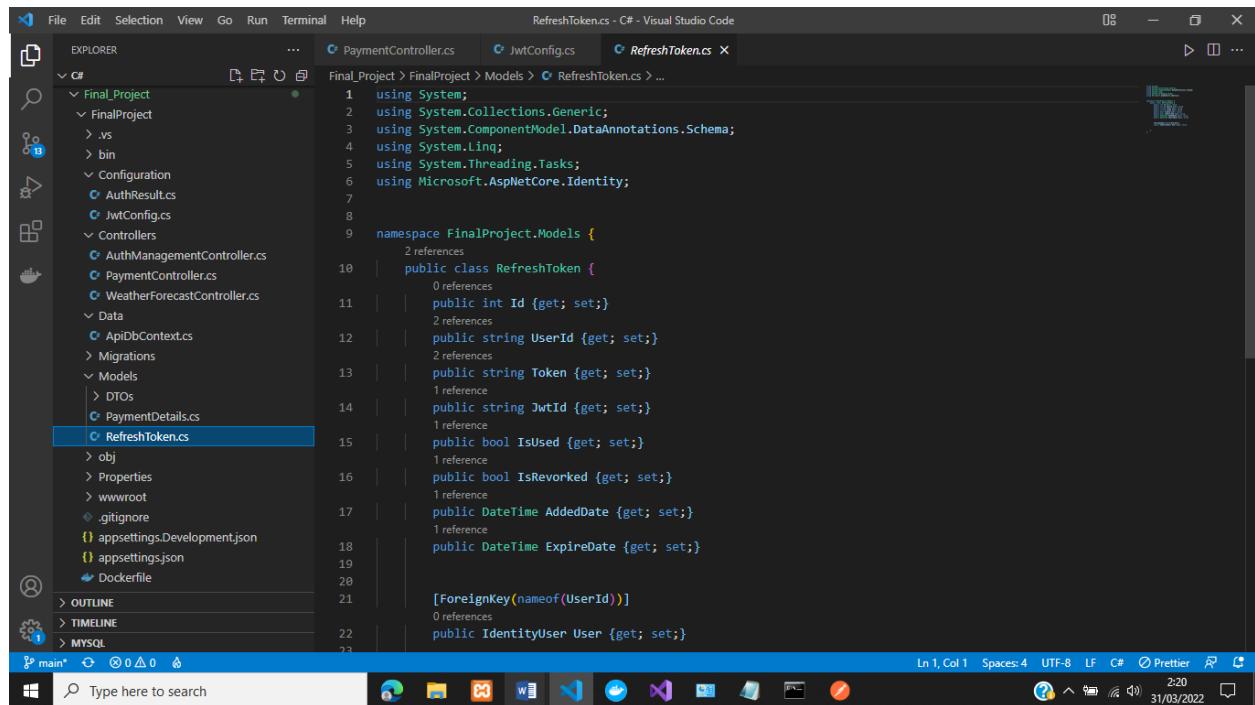
```

{
  "ConnectionStrings": {
    "DefaultConnection": "server=remotemysql.com; port=3306; database=qqrRbkF42v; username=qqrRbkF42v; password=qqrRbkF42v"
  },
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft": "Warning",
      "Microsoft.Hosting.Lifetime": "Information"
    }
  },
  "JwtConfig": {
    "Secret": "rrfcKnsrrezqzwylfdhaolfgzgasmdvsn"
  },
  "AllowedHosts": "*"
}

```

- Buat file baru RefreshToken.cs yang terdapat pada folder Models

Location : Final_Project/Models/RefreshToken.cs



```

using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Identity;

namespace FinalProject.Models {
  public class RefreshToken {
    public int Id { get; set; }
    public string UserId { get; set; }
    public string Token { get; set; }
    public string JwtId { get; set; }
    public bool IsUsed { get; set; }
    public bool IsRevoked { get; set; }
    public DateTime AddedDate { get; set; }
    public DateTime ExpireDate { get; set; }

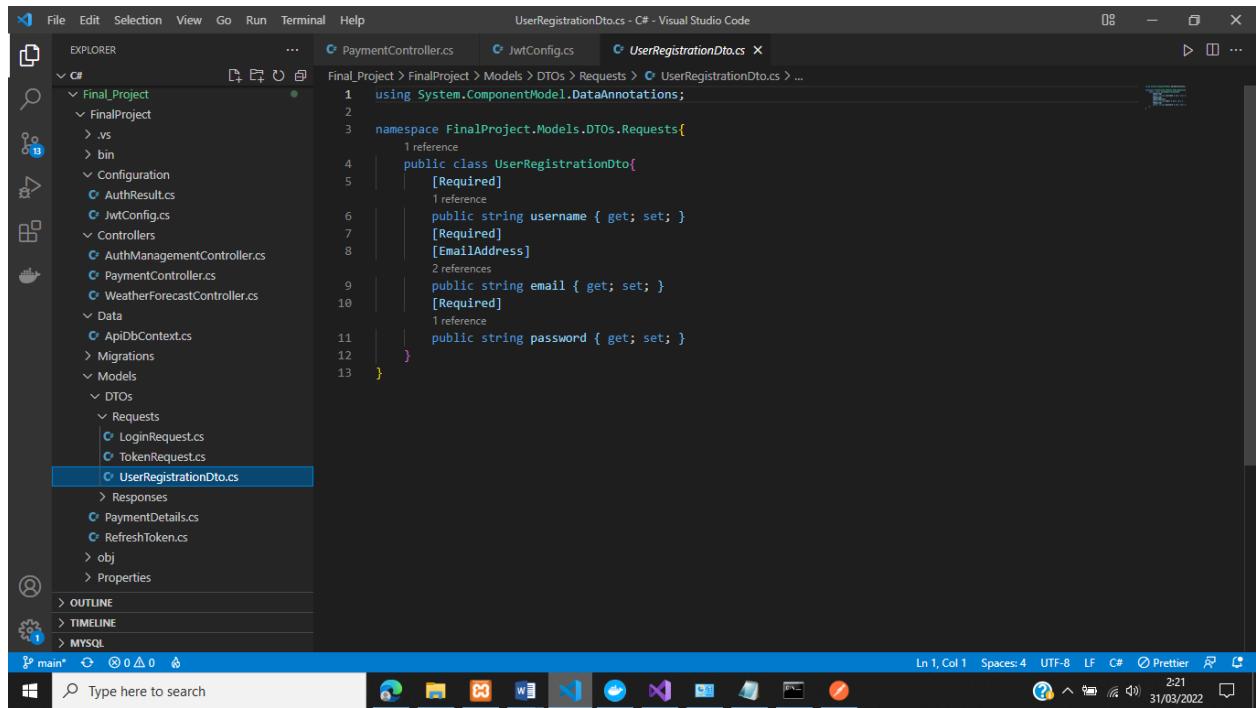
    [ForeignKey(nameof(UserId))]
    public IdentityUser User { get; set; }
  }
}

```

- Buat folder baru bernama DTOs yang di dalamnya terdapat juga folder Request & Response

- **Folder Request**

Location : Final_Project/Models/DTOs/Requests/UserRegistrationDto.cs

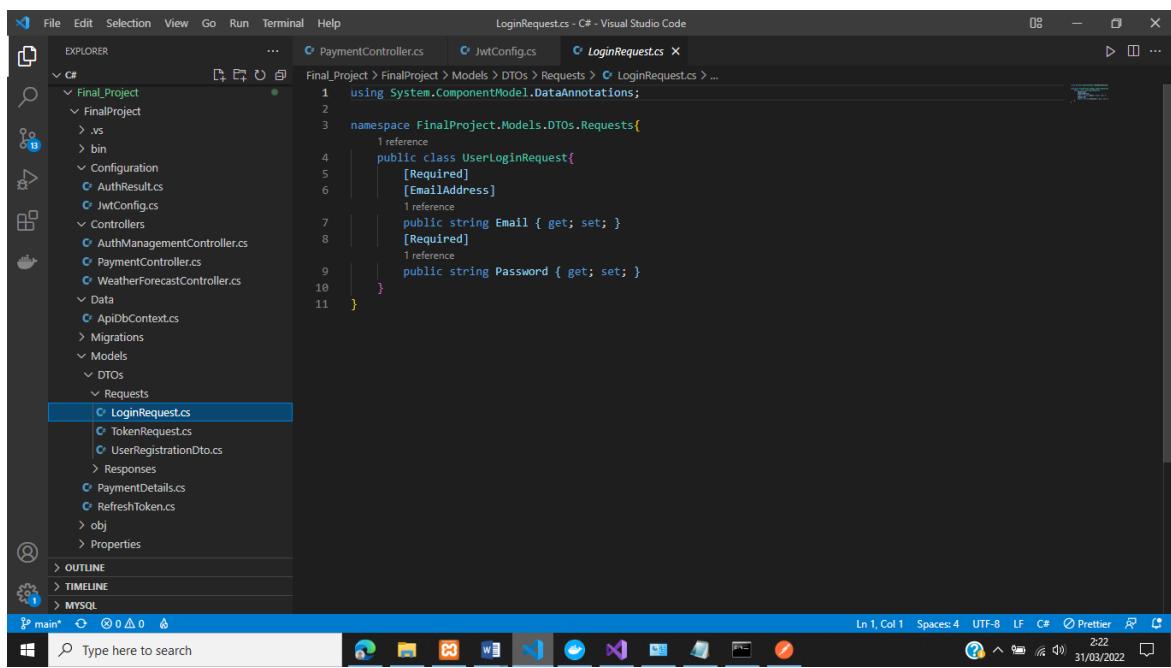


```

File Edit Selection View Go Run Terminal Help UserRegistrationDto.cs - C# - Visual Studio Code
EXPLORER Final_Project > FinalProject > Models > DTOs > Requests > UserRegistrationDto.cs > ...
Final_Project > FinalProject > Models > DTOs > Requests > UserRegistrationDto.cs > ...
1 using System.ComponentModel.DataAnnotations;
2
3 namespace FinalProject.Models.DTOs.Requests{
4     public class UserRegistrationDto{
5         [Required]
6             public string username { get; set; }
7             [Required]
8                 [EmailAddress]
9                     2 references
10                    public string email { get; set; }
11                     [Required]
12                         1 reference
13                             public string password { get; set; }
}

```

Location : Final_Project/Models/DTOs/Requests/LoginRequest.cs

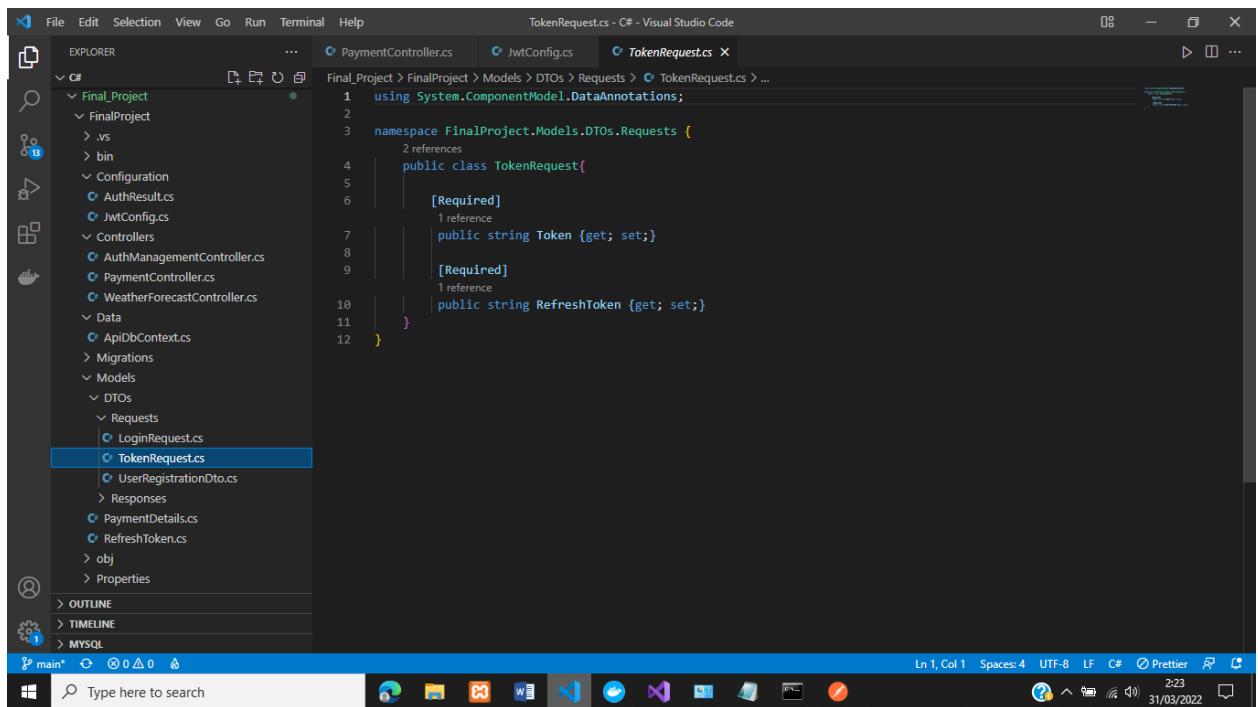


```

File Edit Selection View Go Run Terminal Help LoginRequest.cs - C# - Visual Studio Code
EXPLORER Final_Project > FinalProject > Models > DTOs > Requests > LoginRequest.cs > ...
Final_Project > FinalProject > Models > DTOs > Requests > LoginRequest.cs > ...
1 using System.ComponentModel.DataAnnotations;
2
3 namespace FinalProject.Models.DTOs.Requests{
4     public class UserLoginRequest{
5         [Required]
6             [EmailAddress]
7                     1 reference
8                     public string Email { get; set; }
9                     [Required]
10                        1 reference
11                             public string Password { get; set; }
}

```

Location : Final_Project/Models/DTOs/Requests/TokenRequest.cs



The screenshot shows the Visual Studio Code interface with the following details:

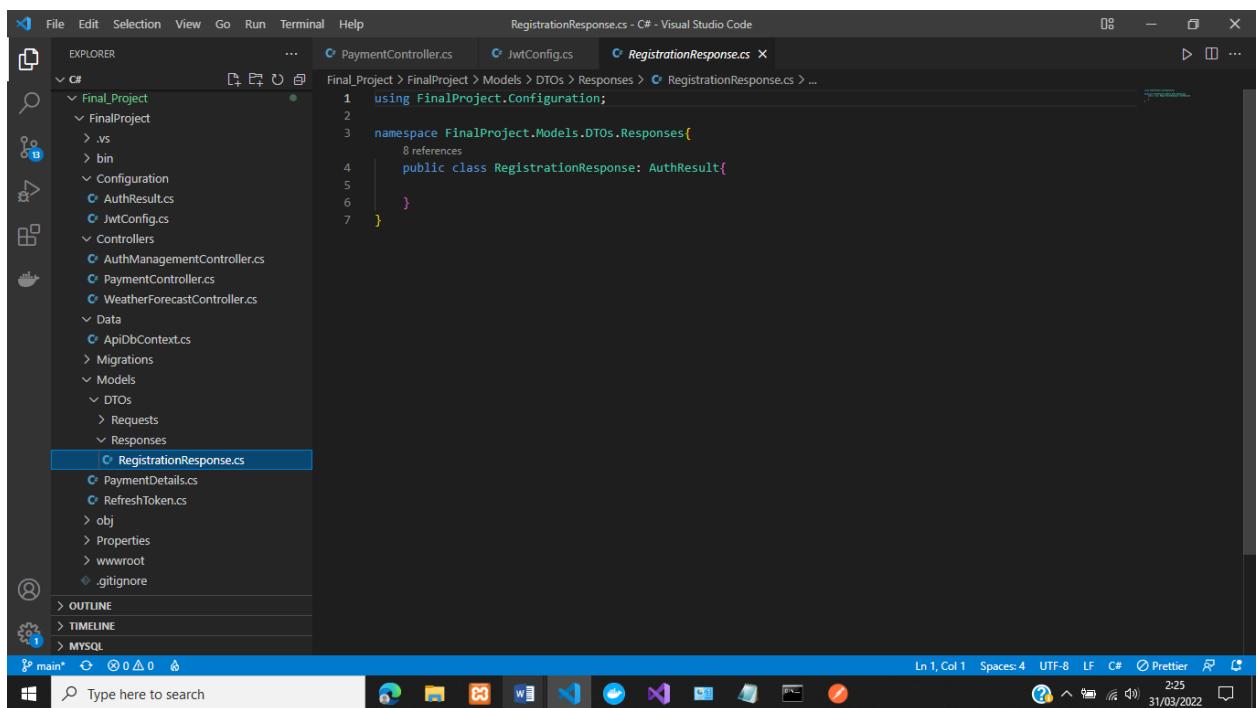
- File Path:** Final_Project/FinalProject/Models/DTOs/Requests/TokenRequest.cs
- Code Content:**

```
1 using System.ComponentModel.DataAnnotations;
2
3 namespace FinalProject.Models.DTOs.Requests {
4     public class TokenRequest{
5         [Required]
6             public string Token {get; set;}
7
8         [Required]
9             public string RefreshToken {get; set;}
10    }
11 }
```

- Explorer View:** Shows the project structure with the current file selected in the Requests folder.
- Bottom Bar:** Includes icons for file operations, search, and various extensions like Prettier.

- **Folder Response**

Location : Final_Project/Models/DTOs/Response/RegistrationResponse.cs



The screenshot shows the Visual Studio Code interface with the following details:

- File Path:** Final_Project/FinalProject/Models/DTOs/Responses/RegistrationResponse.cs
- Code Content:**

```
1 using FinalProject.Configuration;
2
3 namespace FinalProject.Models.DTOs.Responses{
4     public class RegistrationResponse: AuthResult{
5     }
6 }
```

- Explorer View:** Shows the project structure with the current file selected in the Responses folder.
- Bottom Bar:** Includes icons for file operations, search, and various extensions like Prettier.

- Tambahkan file AuthManagement pada folder Controllers
Location : Final_Project/Controllers/AuthManagementController.cs

```
File Edit Selection View Go Run Terminal Help AuthManagementController.cs - C# - Visual Studio Code

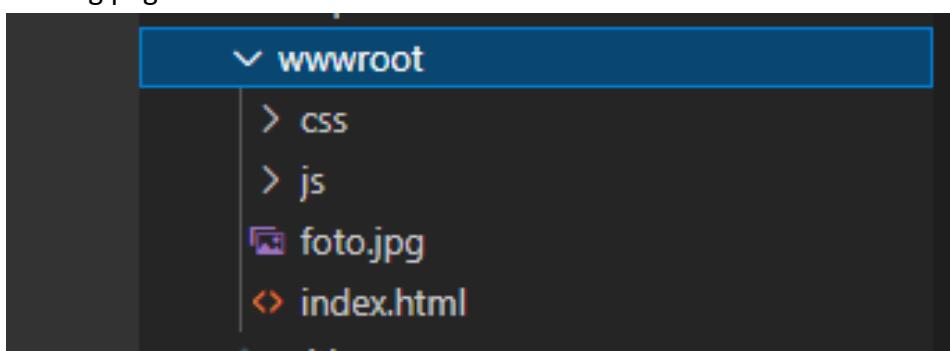
EXPLORER PaymentController.cs JwtConfig.cs AuthManagementController.cs
Final_Project FinalProject > Controllers AuthManagementController.cs > () FinalProject.Controllers > FinalProject.Controllers.AuthManagementController > Gener...
Final_Project vs bin Configuration AuthResult.cs JwtConfig.cs Controllers AuthManagementController.cs PaymentController.cs WeatherForecastController.cs Data ApiDbContext.cs Migrations Models DTOs Requests Responses RegistrationResponse.cs PaymentDetails.cs RefreshToken.cs obj Properties wwwroot .gitignore
OUTLINE
TIMELINE
MYSQL

1  using System;
2  using System.Collections.Generic;
3  using System.IdentityModel.Tokens.Jwt;
4  using System.Linq;
5  using System.Security.Claims;
6  using System.Text;
7  using System.Threading.Tasks;
8  using Microsoft.AspNetCore.Identity;
9  using Microsoft.AspNetCore.Mvc;
10 using Microsoft.EntityFrameworkCore;
11 using Microsoft.Extensions.Options;
12 using Microsoft.IdentityModel.Tokens;
13 using FinalProject.Configuration;
14 using FinalProject.Data;
15 using FinalProject.Models;
16 using FinalProject.Models.Dtos.Requests;
17 using FinalProject.Models.Dtos.Responses;
18
19 namespace FinalProject.Controllers{
20
21     [Route("api/[controller]")]
22     [ApiController]
23     public class AuthManagementController: ControllerBase{
24         private readonly UserManager<IdentityUser> _userManager;
25         private readonly JwtConfig _jwtConfig;
26         private readonly TokenValidationParameters _tokenValidationParams;
27         private readonly ApiDbContext _apiDbContext;
28
29     }
30
31     [HttpPost("register")]
32     public async Task<ActionResult<RegistrationResponse>> Register([FromBody] RegisterRequest request)
33     {
34         var user = new IdentityUser{...}
35         ...
36         var result = await _userManager.CreateAsync(user, "password");
37         if(result.Succeeded)
38         {
39             var token = _jwtConfig.CreateToken(result);
40             return Ok(new RegistrationResponse{...});
41         }
42         return BadRequest(result.Errors);
43     }
44
45     [HttpPost("login")]
46     public async Task<ActionResult<RegistrationResponse>> Login([FromBody] LoginRequest request)
47     {
48         var user = await _userManager.FindByNameAsync(request.Username);
49         if(user != null && _userManager.CheckPasswordAsync(user, request.Password).Result)
50         {
51             var token = _jwtConfig.CreateToken(user);
52             return Ok(new RegistrationResponse{...});
53         }
54         return Unauthorized();
55     }
56
57     [HttpGet("refresh")]
58     public IActionResult RefreshToken()
59     {
60         var token = HttpContext.Session.GetString("RefreshToken");
61         if(token != null)
62         {
63             var claims = _jwtConfig.GetClaimsFromToken(token);
64             var principal = new ClaimsPrincipal(claims);
65             var result = await _userManager.CreateAsync(principal);
66             if(result.Succeeded)
67             {
68                 var newToken = _jwtConfig.CreateToken(result);
69                 HttpContext.Session.SetString("RefreshToken", newToken);
70                 return Ok(new RegistrationResponse{...});
71             }
72         }
73         return Unauthorized();
74     }
75
76     [HttpGet("user")]
77     public async Task<ActionResult<IdentityUser>> GetUser()
78     {
79         var user = await _userManager.GetUserAsync(User);
80         if(user != null)
81         {
82             return Ok(user);
83         }
84         return Unauthorized();
85     }
86
87     [HttpDelete("user")]
88     public async Task<ActionResult<IdentityUser>> DeleteUser()
89     {
90         var user = await _userManager.GetUserAsync(User);
91         if(user != null)
92         {
93             await _userManager.DeleteAsync(user);
94             return Ok(user);
95         }
96         return Unauthorized();
97     }
98 }
```

- JWT dengan batas waktu token 30 menit
Set waktu expired selama 30 menit pada fungsi GenerateJwtToken di AuthManagementController.cs

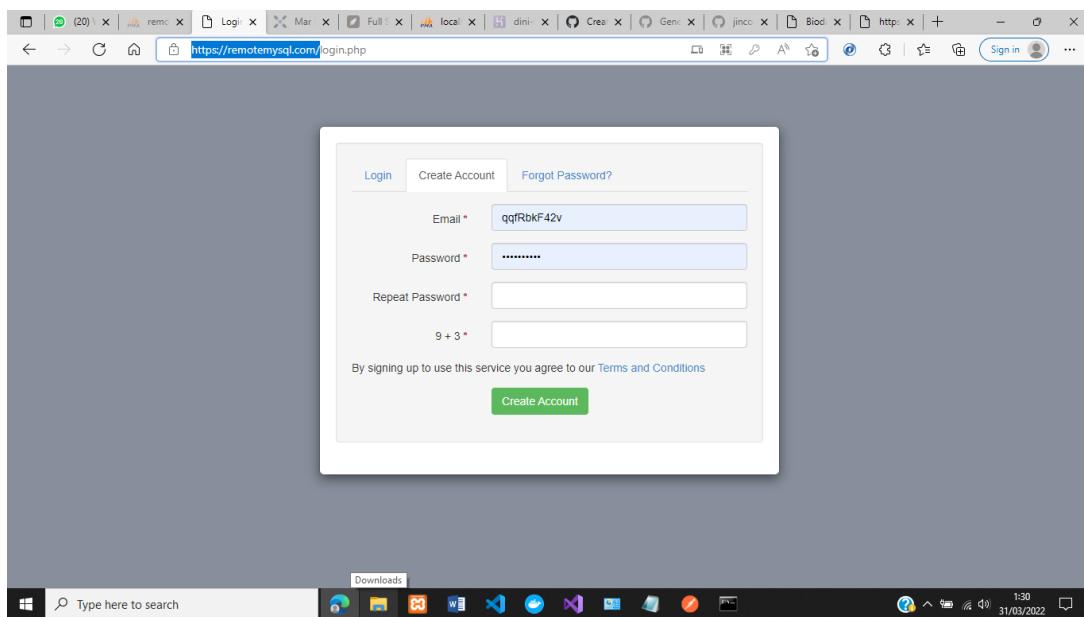
```
        },
        Expires = DateTime.UtcNow.AddMinutes(30),
        SigningCredentials = new SigningCredentials(new SymmetricSecurityKey(key), SecurityAlgorithm),
    };
}
```

- Pembuatan Landing page
Pada project Final_Project buat file “wwwroot” yang berisi file index.html sebagai landing page.

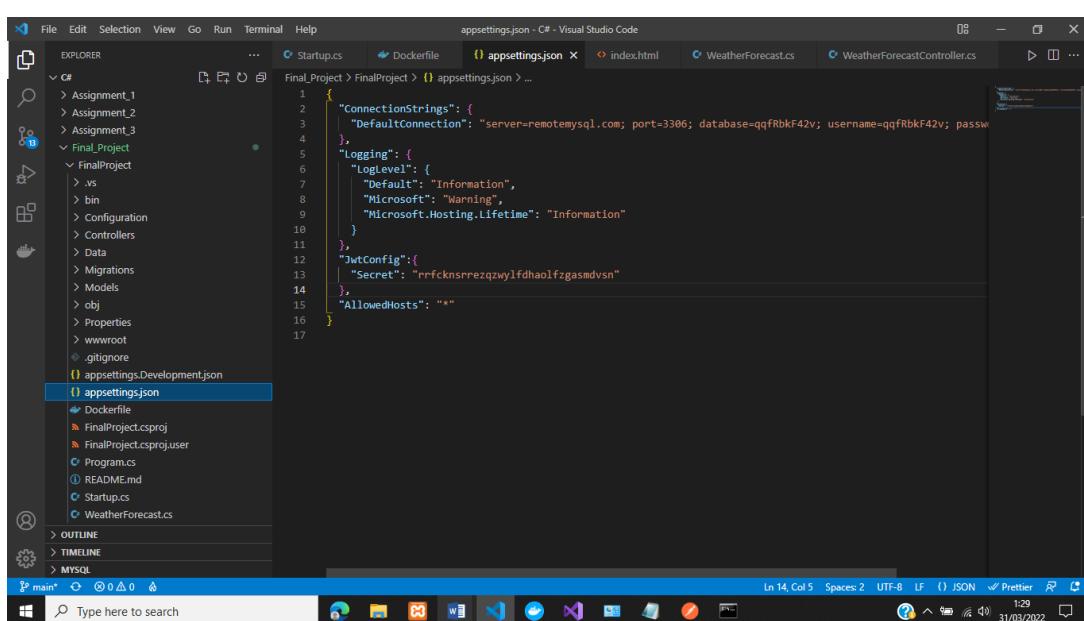


DATABASE

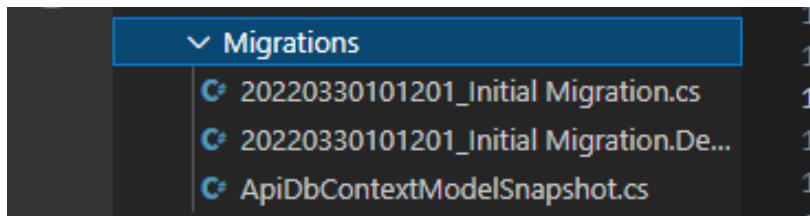
- Buat Akun di <https://remotemysql.com/>



- Setelah masuk dashboard, klik link : <https://remotemysql.com/databases.php?action=new> untuk membuat database
((disini kita akan menerima username, password, dll yang kemudian akan disesuaikan pada file appsettings.json))
- Sesuaikan database didalam appsettings.json



- Lakukan Migrasi **dotnet ef migrations add "Initial Migrations"**
Maka akan muncul File Migrations di dalam Folder Final_Project



- Lakukan Database Update.

DOCKER

- Buat file Dockerfile di dalam folder Final_Project

```

File Edit Selection View Go Run Terminal Help Dockerfile - C# - Visual Studio Code
EXPLORER Dockerfile appsettings.json index.html WeatherForecast.cs WeatherForecastController.cs
Assignment_1 Assignment_2 Assignment_3 Final_Project FinalProject
FinalProject FinalProject.csproj FinalProject.csproj.user Program.cs README.md Startup.cs WeatherForecast.cs
.gitignore appsettings.Development.json appsettings.json
Dockerfile
FROM mcr.microsoft.com/dotnet/aspnet:5.0 AS base
WORKDIR /app
EXPOSE 80
EXPOSE 443
EXPOSE 48719
EXPOSE 27017
EXPOSE 5000
EXPOSE 5001
FROM mcr.microsoft.com/dotnet/sdk:5.0 AS build
WORKDIR /src
COPY . .
#COPY ["CoreWebApi/CoreWebApi.csproj", "CoreWebApi/"]
#RUN dotnet restore "CoreWebApi/CoreWebApi.csproj"
#COPY . .
#WORKDIR "/src/CoreWebApi"
#RUN dotnet build "CoreWebApi.csproj" -c Release -o /app/build
RUN dotnet restore
RUN dotnet build --no-restore -c Release -o /app
FROM build AS final
WORKDIR /app
#COPY --from=build /app/publish .
COPY --from=build /app/
#ENTRYPOINT ["dotnet", "CoreWebApi.dll"]

```

The screenshot shows the Visual Studio Code interface with the Dockerfile open in the editor. The Dockerfile defines a multi-stage build process. It starts with a base image `mcr.microsoft.com/dotnet/aspnet:5.0` and sets the working directory to `/app`. It then exposes ports 80, 443, 48719, 27017, 5000, and 5001. The second stage, `build`, uses the `mcr.microsoft.com/dotnet/sdk:5.0` image, also with a working directory of `/src`. It copies the project files and restores the dependencies. The third stage, `final`, uses the `build` stage's image and copies the published files from the build stage's `/app/publish` directory to the final image's `/app` directory. Finally, it sets the entry point to `dotnet CoreWebApi.dll`.

- Lakukan build dengan perintah dotnet build pada terminal

- Lakukan Release dengan perintah ***dotnet publish -c release***

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer (Left):** Shows the project structure with files like `Startup.cs`, `appsettings.json`, `index.html`, `ApiDbContext.cs`, `WeatherForecast.cs`, and `WeatherForecastController.cs`. It also lists `.vs`, `bin`, `Configuration`, `Controllers`, `Data`, `Migrations`, `Models`, `obj`, `Properties`, `wwwroot`, `.gitignore`, `appsettings.Development.json`, and `appsettings.json`.
- Code Editor (Center):** Displays the `index.html` file content, which includes Bootstrap CSS and JavaScript imports.
- Bottom Status Bar:** Shows the current file is `index.html - C# - Visual Studio Code`, with status indicators for line 84, column 61, spaces, tabs, and encoding.

- Pembuatan docker image dengan syntax **docker build -t paymentapiimage:dev** .

The screenshot shows the Visual Studio Code interface. The code editor displays the `index.html` file with some HTML code. The terminal below shows the command line output of the build process:

```
[+] Building 58.2s (9/15)
-> => transferring context: 28
-> [internal] load metadata for mcr.microsoft.com/dotnet/aspnet:5.0
-> [internal] load metadata for mcr.microsoft.com/dotnet/sdk:5.0
-> [build 1/5] FROM mcr.microsoft.com/dotnet/sdk:5.0@sha256:aae1e5029730fbf3995832a0bb8dbc3b8d0ed0b5aee1c1e96b5c0b97c8822b
-> [base 1/2] FROM mcr.microsoft.com/dotnet/aspnet:5.0@sha256:770bf871bc1b986e90d9de190b8804adf946fc76db5d1e4dd5e676ccb3a9135
-> [internal] load build context
-> => transferring context: 62.41MB
-> CACHED [build 2/5] WORKDIR /src
-> [build 3/5] COPY .
-> [build 4/5] RUN dotnet restore
```

The screenshot shows the Visual Studio Code interface. The code editor displays the `index.html` file. The terminal below shows the command line output of the build process and the creation of the Docker image:

```
[+] Building 58.2s (9/15)
-> => transferring context: 28
-> [internal] load metadata for mcr.microsoft.com/dotnet/aspnet:5.0
-> [internal] load metadata for mcr.microsoft.com/dotnet/sdk:5.0
-> [build 1/5] FROM mcr.microsoft.com/dotnet/sdk:5.0@sha256:aae1e5029730fbf3995832a0bb8dbc3b8d0ed0b5aee1c1e96b5c0b97c8822b
-> [base 1/2] FROM mcr.microsoft.com/dotnet/aspnet:5.0@sha256:770bf871bc1b986e90d9de190b8804adf946fc76db5d1e4dd5e676ccb3a9135
-> [internal] load build context
-> => transferring context: 62.41MB
-> CACHED [build 2/5] WORKDIR /src
-> [build 3/5] COPY .
-> [build 4/5] RUN dotnet restore
```

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
PS C:\Users\Widya\Documents\C#\Final_Project\FinalProject>

((Apabila berhasil, akan muncul paymentapiimage pada Docker))

Images on disk

3 images Total size: 283.09 MB IN USE UNUSED Clean up...

LOCAL REMOTE REPOSITORIES

NAME	TAG	IMAGE ID	CREATED	SIZE
docker/getting-started	latest	bd9a9ff733898	about 2 months ago	28.79 MB
paymentapiimage	dev	3becbf500566	less than a minute ago	246.58 MB
weatherimagedini	dev	d25326c76b00	about 19 hours ago	213.08 MB

Connect to Remote Content

Not connected ✓ Store and backup your images remotely ✓ Unlock vulnerability scanning for greater security
✓ Collaborate with your team ✓ Connect for free Sign in

- Membuat container dengan menjalankan perintah **docker run -d -p 5051:80 --name paymentapicontainer paymentapiimage:dev** (aplikasi akan berjalan pada port 5051)

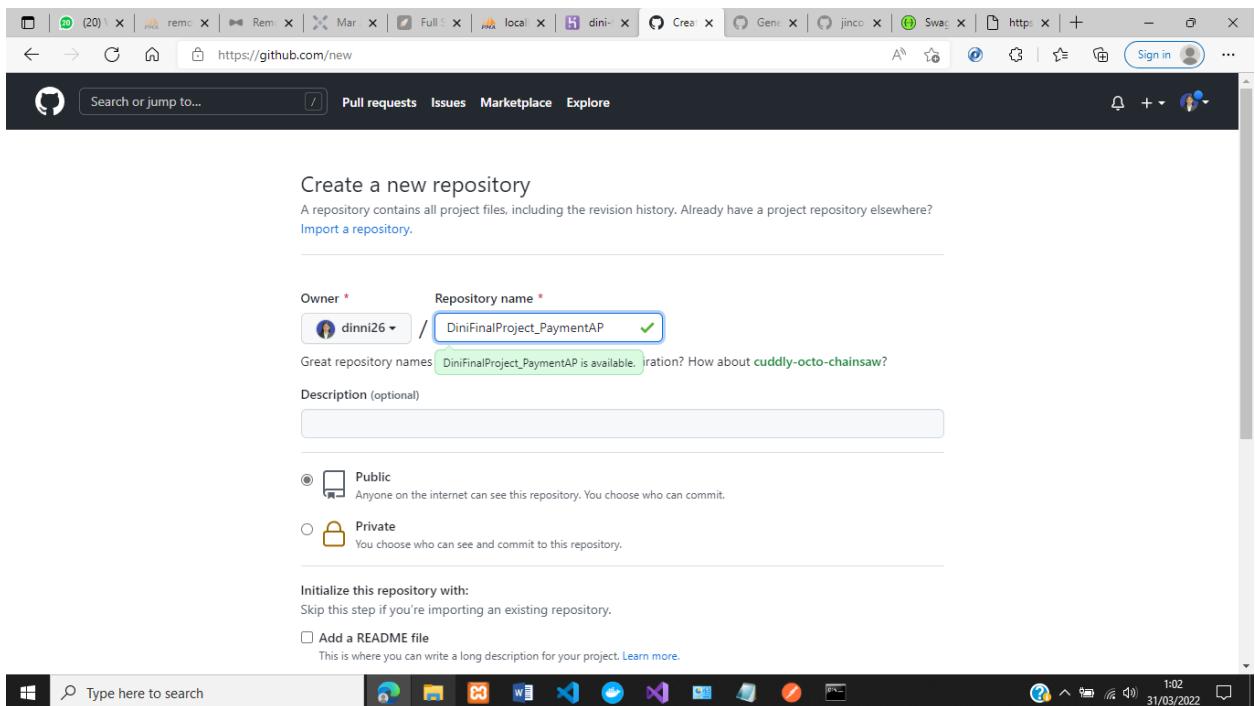
Containers / Apps

Search...

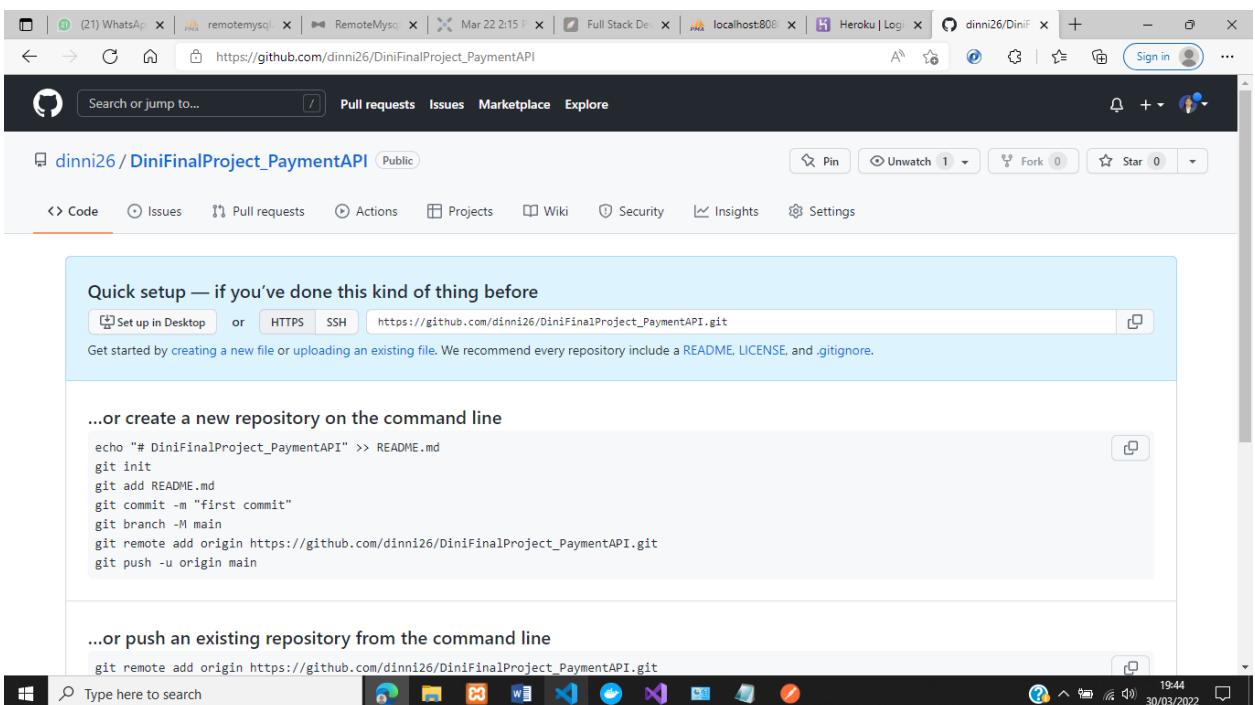
NAME	STATUS	PORTS
weathercontainererdini	EXITED (137)	PORT: 5051
elastic_almeida	EXITED (255)	PORT: 80
paymentapicontainer	RUNNING	PORT: 5051

PEMBUATAN REPO BARU DI GITHUB

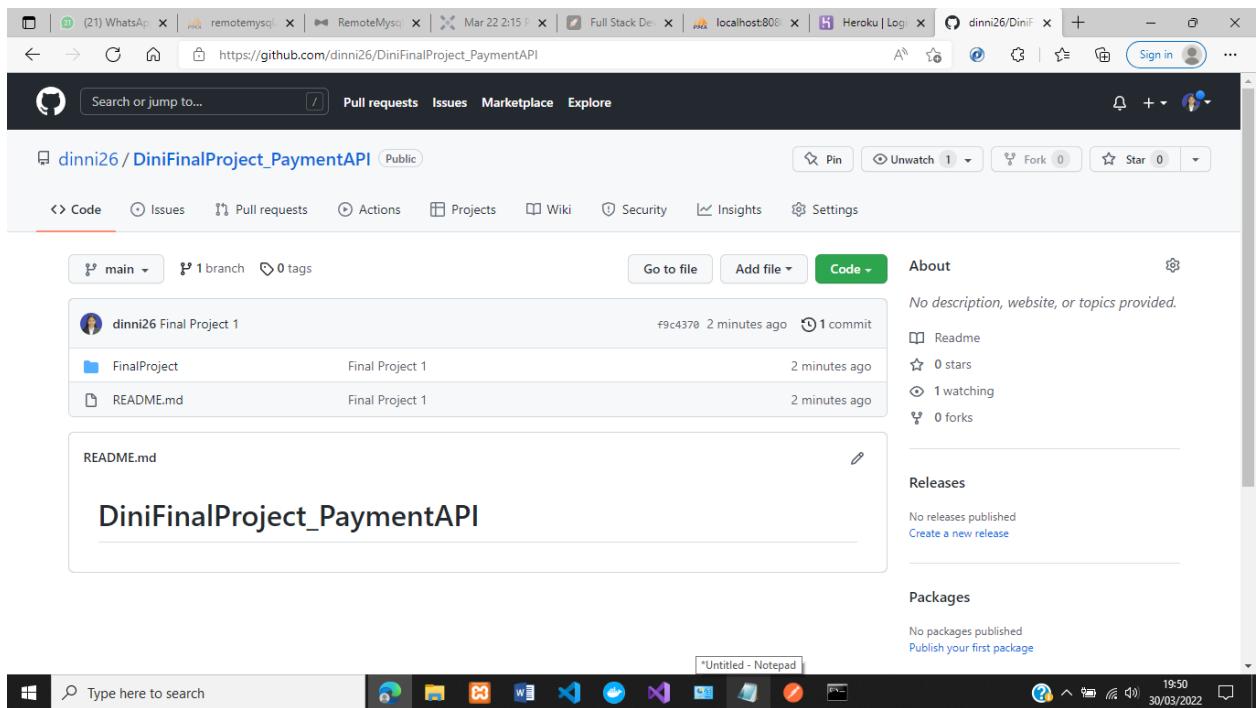
- Create new repo lalu beri nama repo yang baru dibuat



- Setelah terbuat, akan muncul code yang akan kita jalankan di cmd

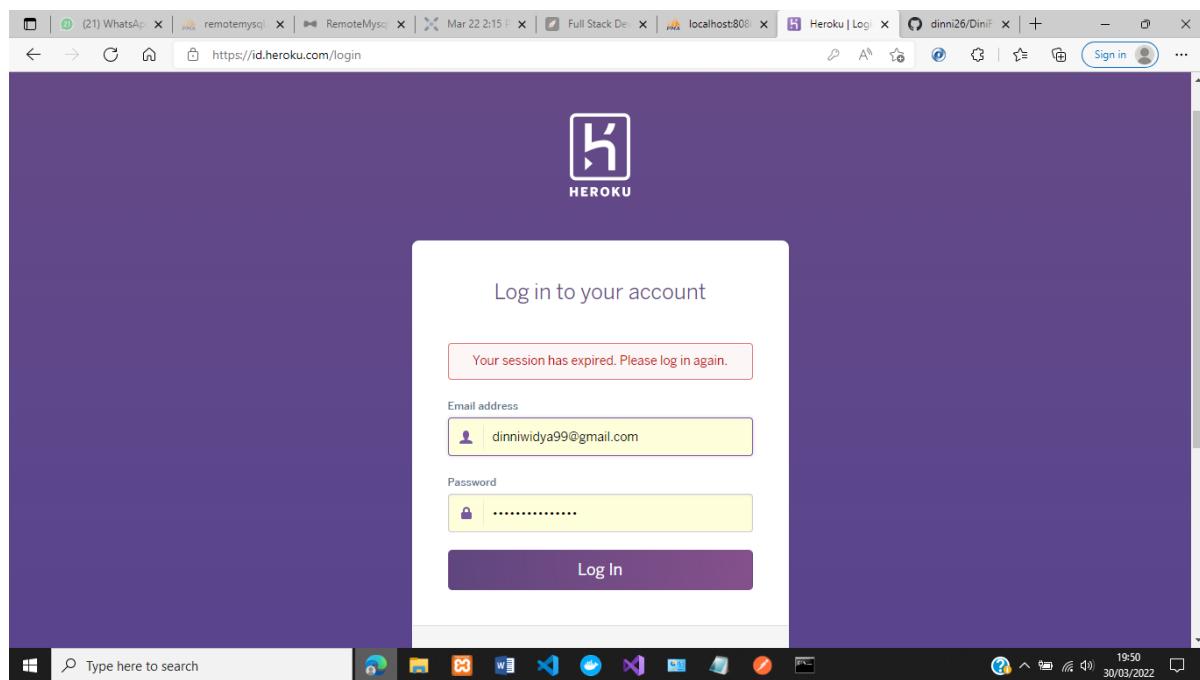


- Setelah dijalankan melalui cmd, maka repository baru akan tampil folder FinalProject yang sudah dibuat

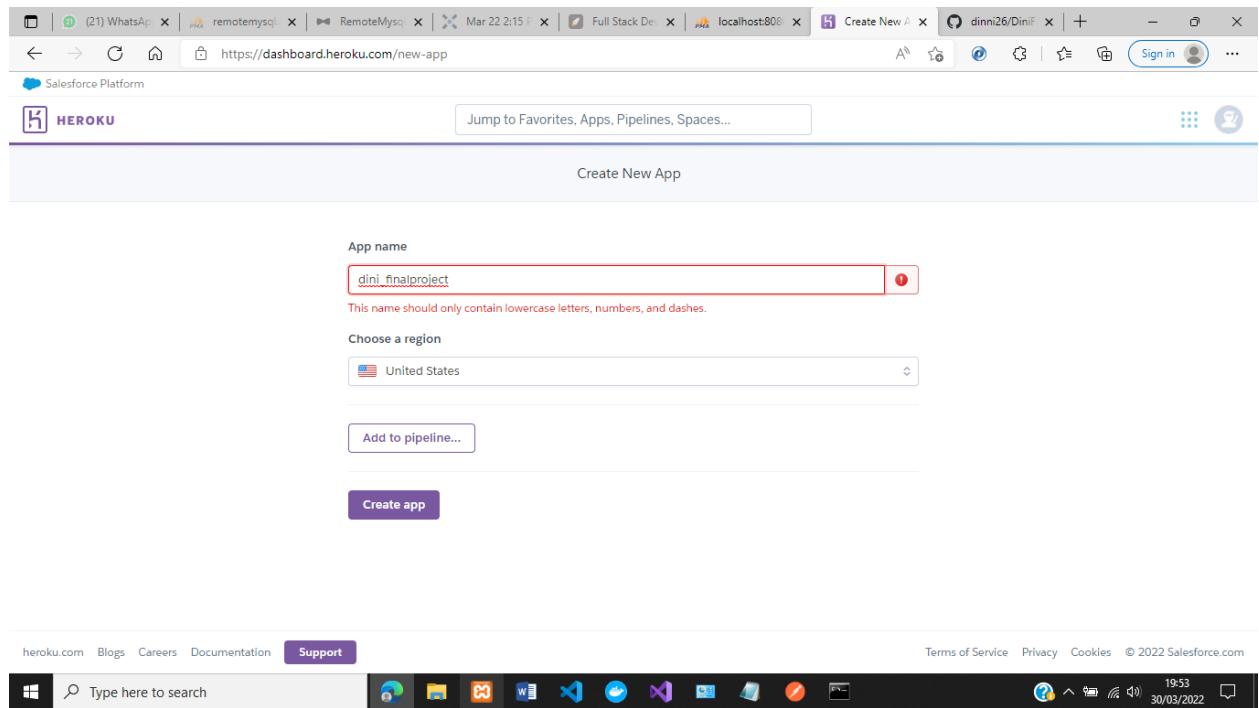


HEROKU

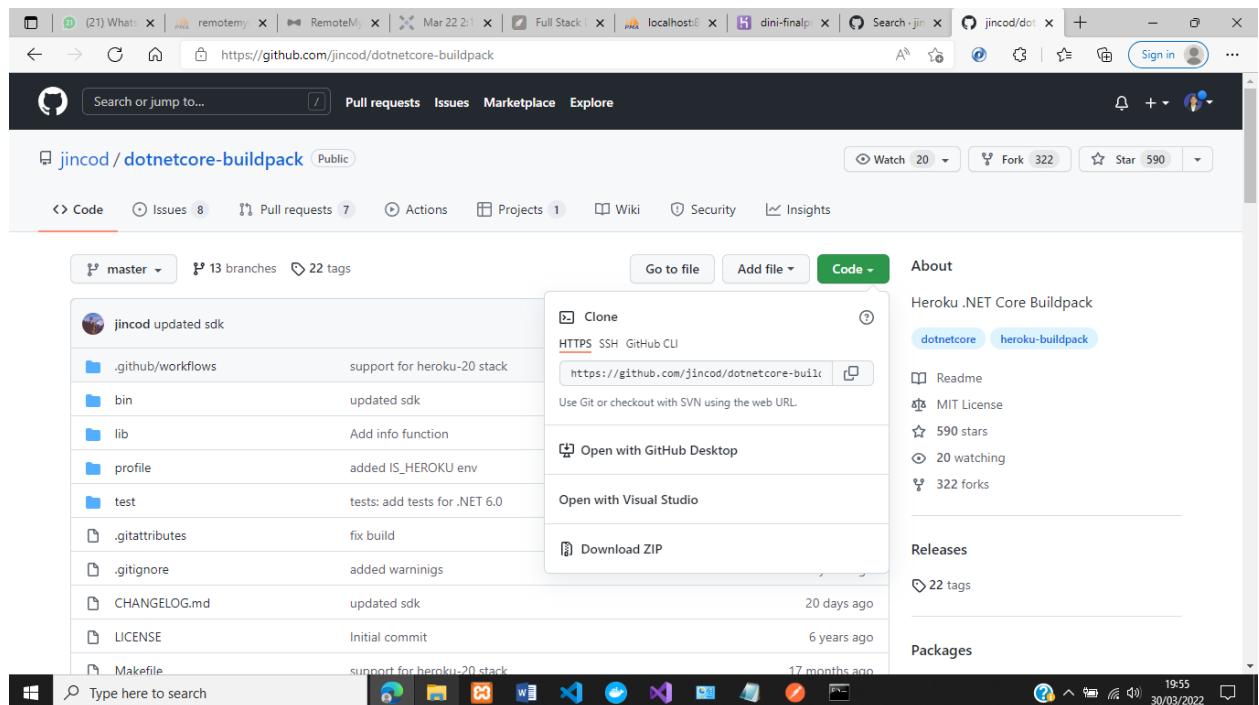
- Login Akun Heroku



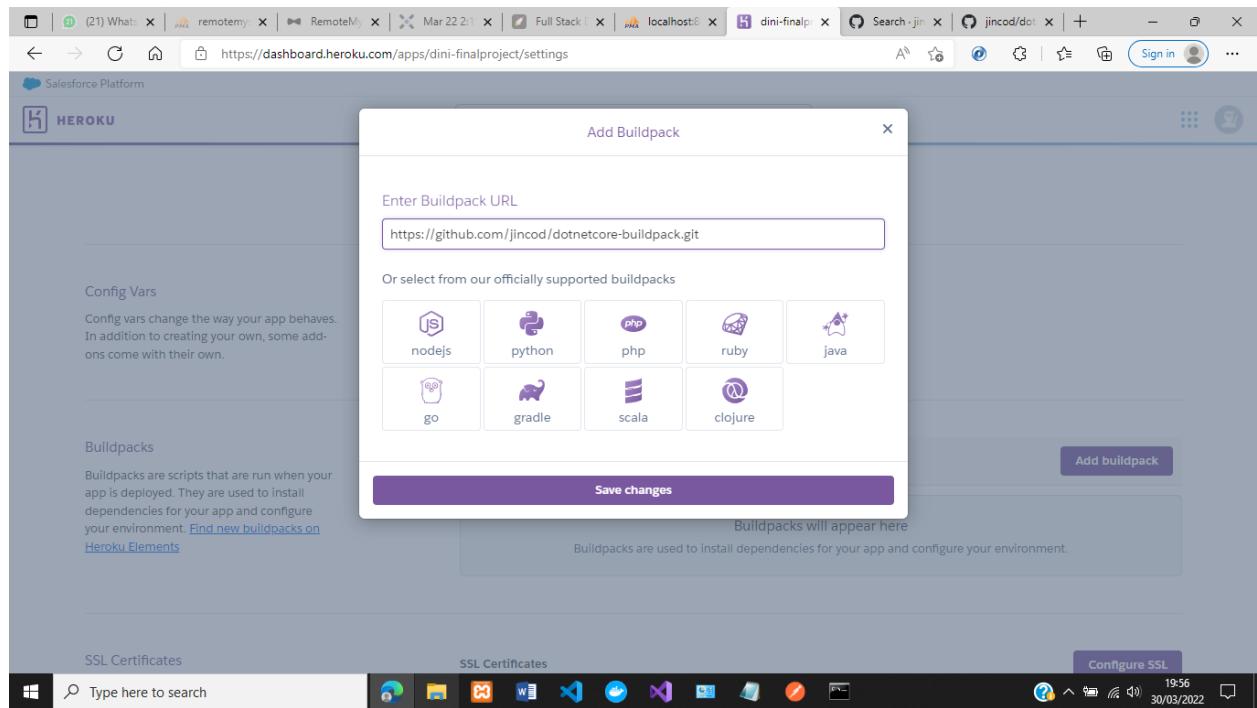
- Create New App lalu masukan nama app yang akan dibuat lalu create app.



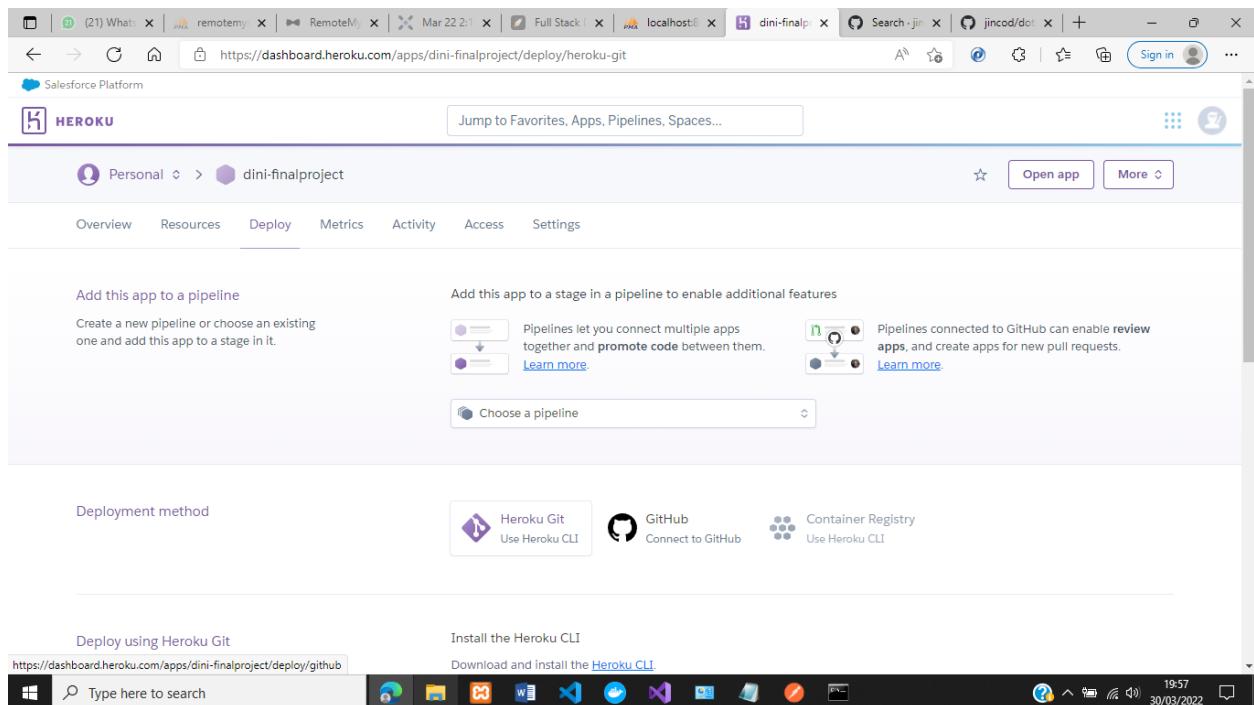
- Setelah Create App > Settings > Add Buildpack > Copy Link Jincod dari github



- Paste link Jincod Kedalam Buildpack URL



- Lalu Ke Deploy > Connect to Github



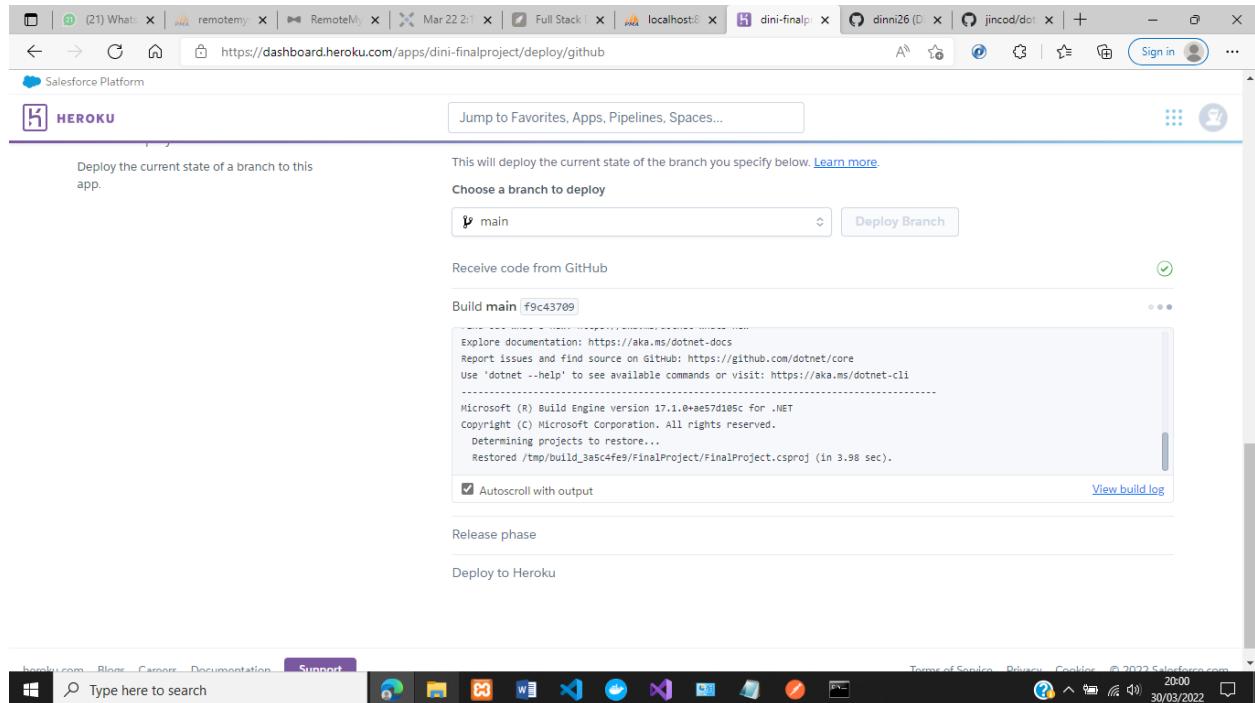
- Masukan Repo Github lalu Connect

The screenshot shows the Heroku Dashboard at <https://dashboard.heroku.com/apps/dini-finalproject/deploy/github>. At the top, there's a navigation bar with tabs like 'What's New', 'remotely...', 'RemoteMy...', 'Mar 22 2:1', 'Full Stack', 'localhost:80', 'dini-final...', 'dinni26...', 'jincod...', and a 'Sign in' button. Below the navigation is the Heroku logo and a search bar labeled 'Jump to Favorites, Apps, Pipelines, Spaces...'. A 'Choose a pipeline' dropdown is present. Under 'Deployment method', there are three options: 'Heroku Git' (using Heroku CLI), 'GitHub Connect to GitHub' (selected), and 'Container Registry' (using Heroku CLI). The main section is titled 'Connect to GitHub' with the sub-instruction 'Connect this app to GitHub to enable code diffs and deploys.' It features a search bar where 'dinni26' is typed and 'DiniFinalProject_PaymentAPI' is selected from the dropdown. A 'Search' button is next to the dropdown. Below the search bar, it says 'Missing a GitHub organization? [Ensure Heroku Dashboard has team access.](#)'. A 'Connect' button is at the bottom right. The bottom of the screen shows a Windows taskbar with icons for File Explorer, Task View, Start, Task Manager, and others, along with a system tray showing the date and time as '30/03/2022 19:58'.

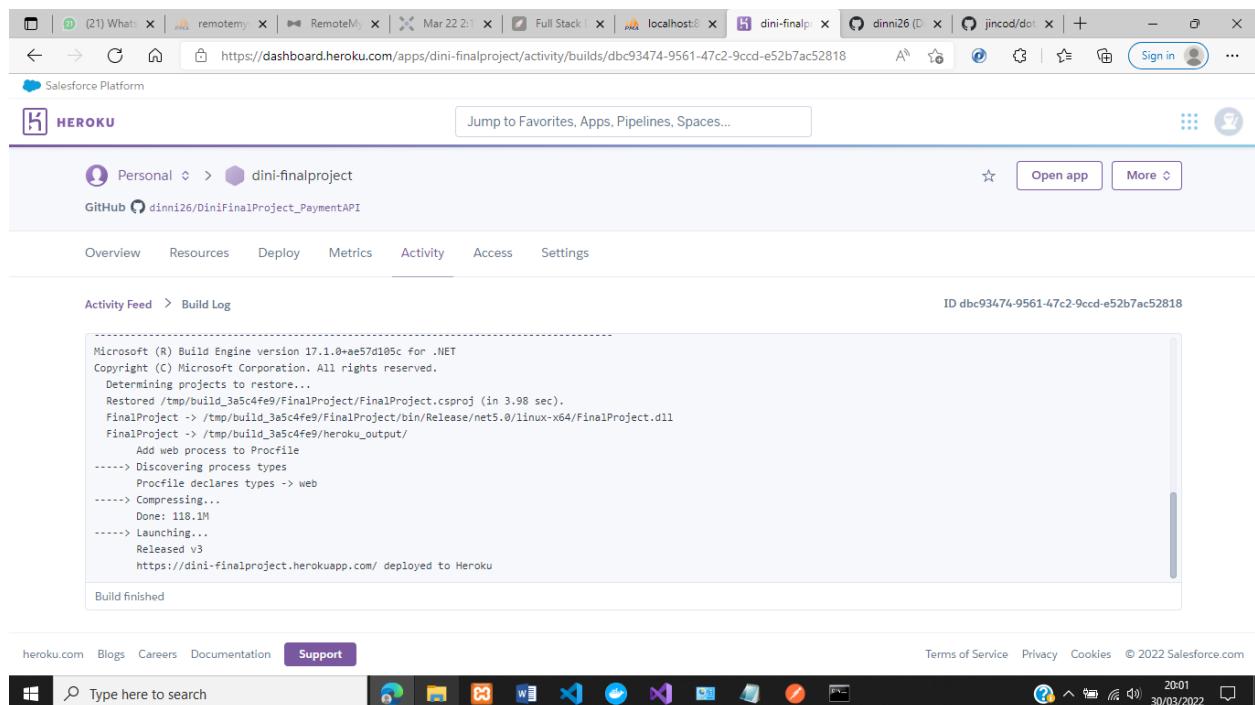
- Setelah itu scroll ke bawah lalu klik "Deploy Branch"

The screenshot shows the Heroku Dashboard at <https://dashboard.heroku.com/apps/dini-finalproject/deploy/github>. The interface is similar to the previous screenshot but with different content. It shows the 'Enable automatic deploys from GitHub' section with the instruction 'Enables a chosen branch to be automatically deployed to this app.' Below this is a 'Choose a branch to deploy' dropdown set to 'main'. There's a checkbox for 'Wait for CI to pass before deploy' with the note 'Only enable this option if you have a Continuous Integration service configured on your repo.' A large 'Enable Automatic Deploys' button is centered below the dropdown. The bottom half of the screen shows the 'Manual deploy' section with a 'Deploy a GitHub branch' button and a note about deploying the current state of a branch. It also shows a 'Choose a branch to deploy' dropdown set to 'main' and a 'Deploy Branch' button. The bottom of the screen shows a Windows taskbar and system tray identical to the first screenshot.

- Tunggu sampai proses selesai



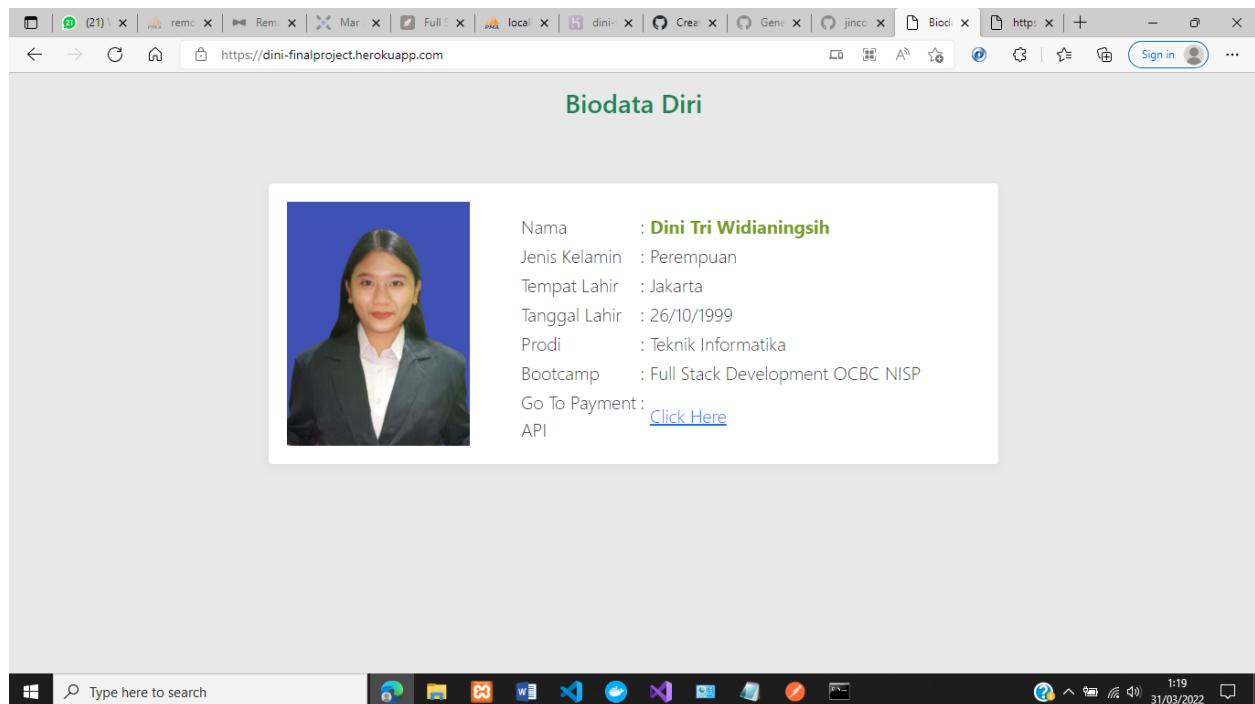
- Setelah selesai maka akan ada link heroku yang nantinya akan kita jalankan untuk penggunaan ProjectAkhir



- Salin Link > Browser untuk menjalankannya.

Link : [Biodata \(dini-finalproject.herokuapp.com\)](https://dini-finalproject.herokuapp.com)

Tampilan Awal



((Lanjut ke panduan penggunaan aplikasi))

Nama : Dini Tri Widianingsih

No. Peserta : FSDO003ONL007

Link Github : [dinni26/DiniFinalProject_PaymentAPI \(github.com\)](https://dinni26/DiniFinalProject_PaymentAPI)

Link Heroku : [Biodata \(dini-finalproject.herokuapp.com\)](https://dini-finalproject.herokuapp.com/)

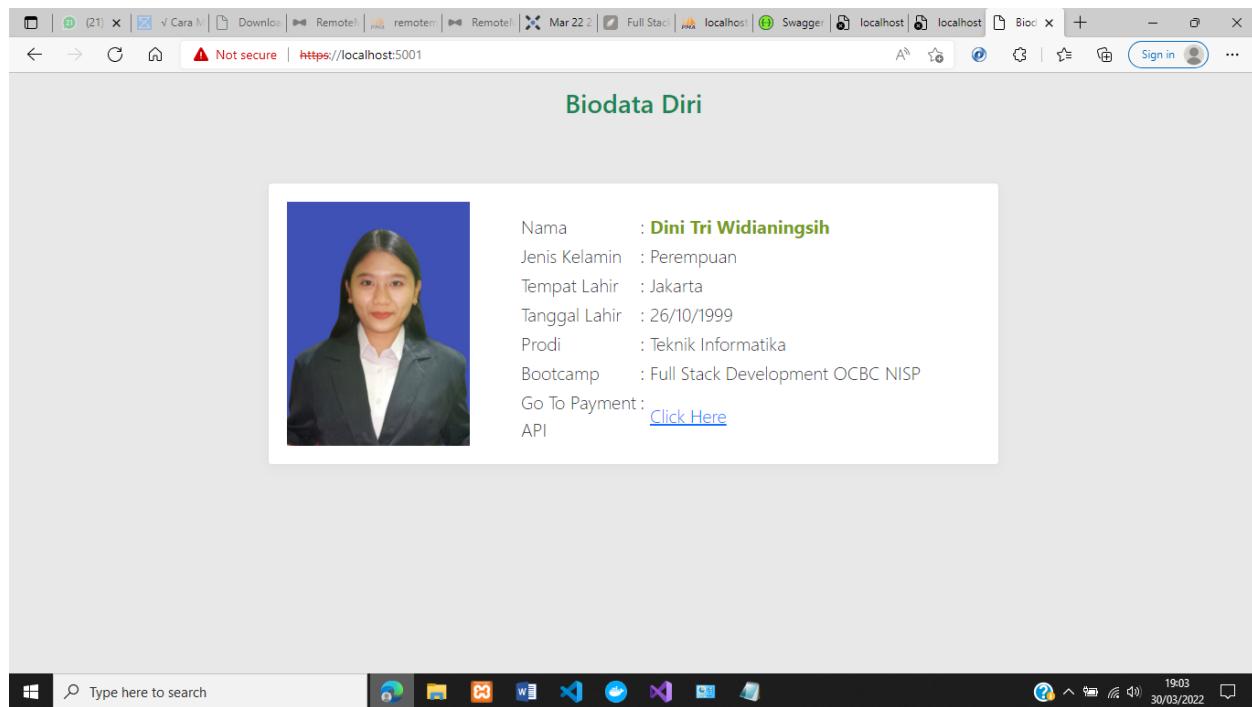
PANDUAN PENGGUNAAN FINAL PROJECT

Uji Coba Menggunakan Swagger dan Postman

Jalankan Link Berikut : <https://dini-finalproject.herokuapp.com/>

SWAGGER

1. Tampilan Awal Saat di Jalankan



((Click Here Untuk Beralih Ke Laman Swagger))

Halaman Swagger

The screenshot shows the Swagger UI interface for a Payment API version 1. The top navigation bar includes tabs for 'Select a definition' (PaymentAPI v1) and 'Sign in'. Below the header, the title 'PaymentAPI' is displayed with 'v1' and 'OAS3' badges. A link to '/swagger/v1/swagger.json' is shown. On the right, there is a green 'Authorize' button with a lock icon.

AuthManagement

- POST /api/AuthManagement/Register** (green button)
- POST /api/AuthManagement/Login** (green button)
- POST /api/AuthManagement/RefreshToken** (green button)

Payment

- GET /api/Payment** (blue button)
- POST /api/Payment** (green button)
- GET /api/Payment/{id}** (blue button)
- PUT /api/Payment/{id}** (orange button)
- DELETE /api/Payment/{id}** (red button)

The screenshot shows the Swagger UI interface for a Payment API version 1. The top navigation bar includes tabs for 'Select a definition' (PaymentAPI v1) and 'Sign in'. Below the header, the title 'Payment' is displayed with 'v1' and 'OAS3' badges. A link to '/swagger/v1/swagger.json' is shown. On the right, there is a blue 'Authorize' button with a lock icon.

Payment

- GET /api/Payment** (blue button)
- POST /api/Payment** (green button)
- GET /api/Payment/{id}** (blue button)
- PUT /api/Payment/{id}** (orange button)
- DELETE /api/Payment/{id}** (red button)

WeatherForecast

- GET /WeatherForecast** (blue button)

Schemas

- UserRegistrationDto** >
- UserLoginRequest** >

The screenshot shows the Swagger UI interface for a Payment API version 1. The top navigation bar includes tabs for 'Select a definition' (PaymentAPI v1) and 'Sign in'. Below the header, the title 'Payment' is displayed with 'v1' and 'OAS3' badges. A link to '/swagger/v1/swagger.json' is shown. On the right, there is a blue 'Authorize' button with a lock icon.

Payment

- GET /api/Payment** (blue button)
- POST /api/Payment** (green button)
- GET /api/Payment/{id}** (blue button)
- PUT /api/Payment/{id}** (orange button)
- DELETE /api/Payment/{id}** (red button)

WeatherForecast

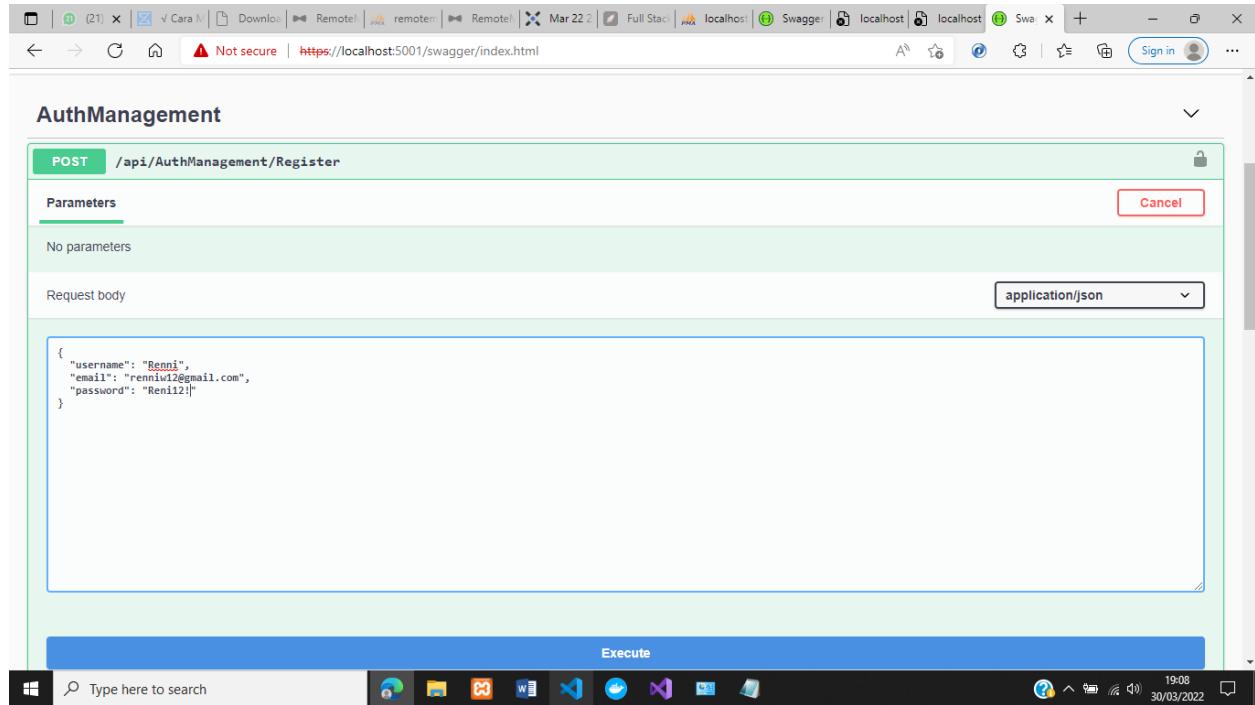
- GET /WeatherForecast** (blue button)

Schemas

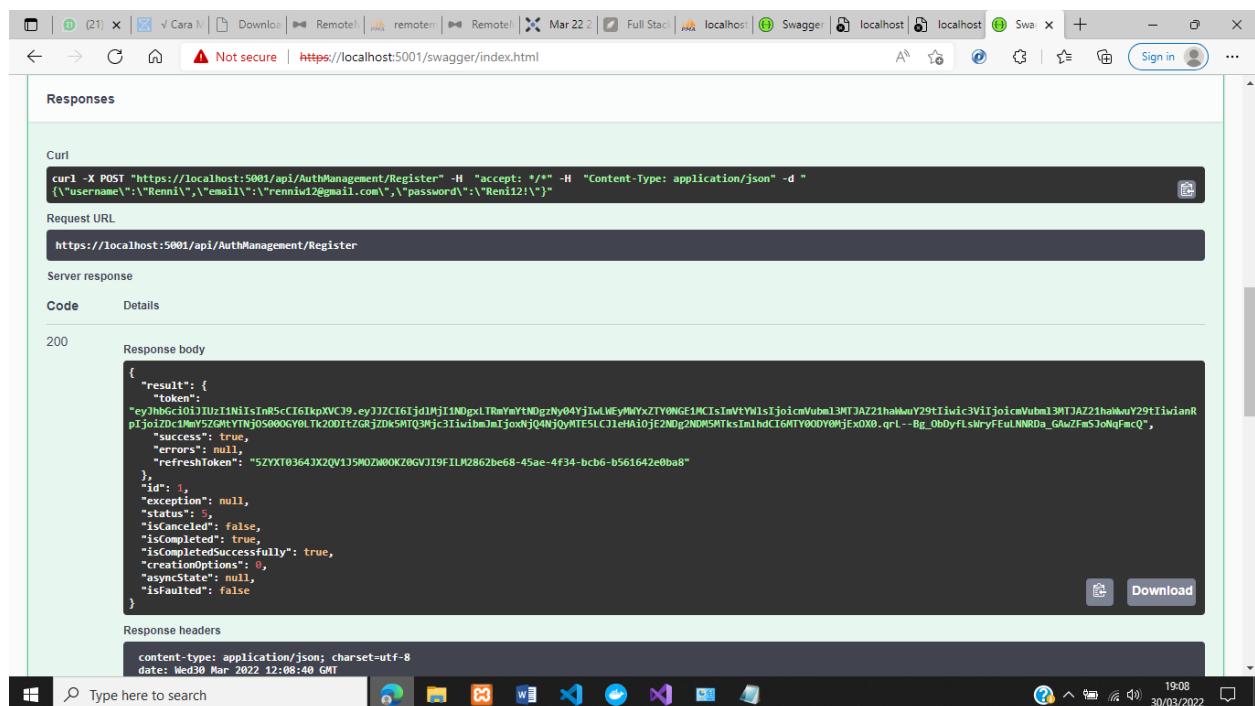
- UserRegistrationDto** >
- UserLoginRequest** >

Register

- Lakukan Register dengan menekan tombol “Try It Out”.
 - Register dengan memasukan username, email, dan password baru.



Execute, lalu respon akan menampilkan token.



Login

- Login menggunakan akun yang sebelumnya sudah didaftarkan dengan memasukan email dan password. Perlu di ingat, token hanya bisa digunakan dalam waktu 30 menit saja.

The screenshot shows the Swagger UI interface for a POST request to the endpoint `/api/AuthManagement/Login`. The request body is defined as follows:

```
{ "email": "renniw12@gmail.com", "password": "Reni12!" }
```

The 'Execute' button is visible at the bottom of the request panel. The browser's address bar shows `https://localhost:5001/swagger/index.html`. The Windows taskbar at the bottom indicates the date and time as 30/03/2022 19:09.

The screenshot shows the Swagger UI interface displaying the response details for the login request. The response code is 200, and the response body is a JSON object containing the following data:

```
{ "result": { "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJJZC16Ijd1MjI1NDgxLTNmYtNDgNy04YjIwLWEyMjYzTjY0NGE1MCIsImVtYWIslJoinvubm13MTjAZ21haMuY29tIiwiic3ViIjoicmVubm13MTjAZ21haMuY29tIiwanrijoInNzEyN2l3UmItZjpMy00MDjkIlgzZETn2E3mlJ00GE4ZjgziwibmJmIjoxNjQ4njyMtg4LC1leHAIoIE2NDg2NDM500gsImhdI6MTY0ODY0MjE4OH0.hMaX1BF7kx8NkZB4rTjy80zikJiwb--d1mhPIQesw", "success": true, "errors": null, "refreshToken": "GX3QJ3Y24CDZSC4F9UK7KE1X6017T31L02U5ec974f-c950-4131-aadf-46b5ad78c146" }, "id": 2, "exception": null, "status": 5, "isCancelled": false, "isCompleted": true, "isCompletedSuccessfully": true, "creationOptions": 0, "asyncState": null, "isAudited": false }
```

The response headers section shows:

```
content-type: application/json; charset=utf-8
date: Wed, 30 Mar 2022 12:09:49 GMT
server: Kestrel
```

The browser's address bar shows `https://localhost:5001/swagger/index.html`. The Windows taskbar at the bottom indicates the date and time as 30/03/2022 19:10.

Refresh Token

- Percobaan Refresh Token sebelum 30 Menit

POST /api/AuthManagement/RefreshToken

Parameters

No parameters

Request body

application/json

```
{
  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ2Cl6IjdmJi1NDgxLTRmYmYtNDg2Ny04Yj1wLjEwMjYxTY0NGE1MCisImVtYnlsIjoicmVubm13MTJAZ21haWuY29tIiwi3ViijoicmVubm13MTJAZ21haWuY29tIiwiianRpIjoInzEyN2U3NmItZjgwly00MDJkLTgtZTET2E3yJmU00GE42jgzLiwiJmJoxhj04NjQyMTg4LC1eHA0jE2NDg2ND05DgsImIhdC16MTY00DY0MjE40H0.hMax1BF7ku8NkZB4rTjY80zikIwb--dImhPIOPesu",
  "refreshToken": "GX3Q3Y2MC0ZC4F9UK7KEIX6D1T731L02U15ec974f-c950-4131-aadf-46b5ad78c146"
}
```

Execute Clear

Responses

Curl

```
curl -X POST "https://localhost:5001/api/AuthManagement/RefreshToken" -H "accept: */*" -H "Content-Type: application/json" -d "{
  \"token\": \"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ2Cl6IjdmJi1NDgxLTRmYmYtNDg2Ny04Yj1wLjEwMjYxTY0NGE1MCisImVtYnlsIjoicmVubm13MTJAZ21haWuY29tIiwi3ViijoicmVubm13MTJAZ21haWuY29tIiwiianRpIjoInzEyN2U3NmItZjgwly00MDJkLTgtZTET2E3yJmU00GE42jgzLiwiJmJoxhj04NjQyMTg4LC1eHA0jE2NDg2ND05DgsImIhdC16MTY00DY0MjE40H0.hMax1BF7ku8NkZB4rTjY80zikIwb--dImhPIOPesu\",
  \"refreshToken\": \"GX3Q3Y2MC0ZC4F9UK7KEIX6D1T731L02U15ec974f-c950-4131-aadf-46b5ad78c146\"
}"
```

Request URL

<https://localhost:5001/api/AuthManagement/RefreshToken>

Server response

Code Details

200 Response body

```
{
  "token": null,
  "success": false,
  "errors": [
    "Token has not yet expired"
  ],
  "refreshToken": null
}
```

Download

Response headers

```
content-type: application/json; charset=utf-8
date: Wed, 30 Mar 2022 12:10:42 GMT
server: Kestrel
```

Responses

Code Description

200 Success

Links

No links

((Muncul pesan bahwa token masih bisa digunakan))

- Percobaan melakukan Refresh Token Setelah 30 Menit

The screenshot shows a browser window with multiple tabs open. The active tab is 'Swagger UI' at <https://localhost:5001/swagger/index.html>. A warning message 'Not secure' is visible above the address bar. The main content area displays a 'Responses' section for a 'RefreshToken' endpoint. It includes a 'Curl' command, a 'Request URL' of <https://localhost:5001/api/AuthManagement/RefreshToken>, and a 'Server response' block. The response code is 200, and the response body contains JSON indicating the token has expired:

```
{
  "token": null,
  "success": false,
  "errors": [
    "Token has expired, please re-login"
  ],
  "refreshToken": null
}
```

Below the response, there are 'Download' and 'Copy' buttons. The status bar at the bottom right shows the date as 28/03/2022 and the time as 16:47.

((terdapat pesan bahwa token sudah expired dan harus melakukan login kembali))

AUTHORIZE

Masukan Bearer lalu disambungkan dengan token yang didapatkan untuk bisa lanjut ke proses CRUD

The screenshot shows the 'PaymentAPI v1' Swagger UI interface. A modal dialog titled 'Available authorizations' is open, specifically for the 'Bearer (apiKey)' option. It contains instructions to enter a valid token, an example token ('Bearer eyJhbGciOiJIUzI1NiIsInR5cI6IkpxVCJ0'), and fields for 'Name', 'Authorization', 'In: header', and 'Value'. The value field contains the token 'Bearer eyJhbGciOiJIUzI1NiIsInR5cI6IkpxVCJ0'. At the bottom of the dialog are 'Authorize' and 'Close' buttons. The background shows the API documentation for the 'AuthManagement' section, including a 'POST /api/AuthManagement/Register' endpoint. The status bar at the bottom right shows the date as 30/03/2022 and the time as 19:11.

PaymentAPI v1 OAS3

AuthManagement

POST /api/AuthManagement/Register

Parameters

No parameters

Request body

application/json

Logout Close

Value: ****

Authorizations

Bearer (apiKey)

Authorized

Enter 'Bearer' [space] and then your valid token in the text input below.
Example: "Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9"
Name: Authorization
In: header
Value: ****

Authorize

PAYMENT POST

Membuat detail payment baru dengan memasukan data yang diminta. (Masukan 3 data)

Payment

GET /api/Payment

POST /api/Payment

Parameters

No parameters

Request body

application/json

```
{ "paymentDetailId": 1, "cardHolderName": "Reni Widiastuty", "cardNumber": "54453", "expirationDate": "2022/12/23", "securityCode": "Renii123" }
```

((Setelah berhasil, maka data yang sudah di input akan masuk ke dalam database))

GET PAYMENT Untuk mengecek data yang sudah di input.

The screenshot shows a browser window with the address bar displaying "Not secure | https://localhost:5001/swagger/index.html". The main content is a Swagger UI interface for a .NET Core API. The top navigation bar includes tabs for "Payment", "Full Stack", "localhost", "Swagger", and "localhost". The "Payment" tab is active.

Payment

GET /api/Payment

Parameters

No parameters

Responses

curl

```
curl -X GET "https://localhost:5001/api/Payment" -H "accept: */*" -H "Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cGkiXVCj9.eyJJZC161jd1Mj1NDgxLTrYmYTNgdNy04YjIwLWEyMNYxZTY0NGE1MCIsImVtYmIsIjoicmVubm13MTJAZ21haWwvZ9tiiwic3ViIjoicmVubm13MTJAZ21haWwvY29tiiwianRpIjoInzEyNzUNmtfjgWky00MDjkLtgzZTETn2E3mU0GE4ZjgzIiwbmJmIjoxhjQ4NjQyMtg4LC1leHA10jE2NDg2NDM50DgsImhC16MTY00D0Mje40H0.hMAx1BF7kux8NkZB4rIjY80zIkIwb--d1mhPIQew"
```

Request URL

```
https://localhost:5001/api/Payment
```

Server response

Code **Details**

200 Response body

```
[ { }
```

Request URL
<https://localhost:5001/api/Payment>

Server response

Code	Details
200	Response body <pre>[{ "paymentDetailId": 1, "cardOwnerName": "Reni Widiastuty", "cardNumber": "54453", "expirationDate": "2022/12/23", "securityCode": "Renil23" }, { "paymentDetailId": 2, "cardOwnerName": "Bayu Widi Prayogo", "cardNumber": "65543", "expirationDate": "2022/11/10", "securityCode": "Bayul23" }, { "paymentDetailId": 3, "cardOwnerName": "Miwik Supiyati", "cardNumber": "2345", "expirationDate": "2022/09/21", "securityCode": "Miwil123" }]</pre> Response headers <pre>content-type: application/json; charset=utf-8 date: Wed, 30 Mar 2022 12:15:24 GMT server: Kestrel</pre>

Responses

Code	Description
GET /api/Payment/{id}	

((Data sudah masuk ke dalam database))

GET PAYMENT BERDASARKAN ID

Co: Mamanggil ID = 2

GET /api/Payment/{id}

Parameters

Name	Description
id <small>required</small>	integer(\$int32) (path)

Execute Clear

Responses

Curl

```
curl -X GET "https://localhost:5001/api/Payment/2" -H "accept: */*" -H "Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.ejJZC16j1Mj1N0gxL1RwYmTN0g2Ny04YjIwLWeY/WtYzT0NGE1MCisImVtYmIsIjoicmVubm13MTJAZ21haWwY29tliwi3ViijoicmVubm13MTJAZ21haWwY29tlixiianRpIjoINzEyN2l3NmtZjpdMy00MDjkLTgzZTEtn2E3YmU0GE47jgzIiwbmJmjoxNjQ4NjQyMtg4lC1eHa0jE2NDg2N0M500gsImhdC16MTY0ODY0MjE4OH0.hMaX1BF7kux8NkZB4rTjy80zikJiwb--d1mhPIQPesw"
```

Request URL
<https://localhost:5001/api/Payment/2>

Server response

Code	Details
200	Response body

((Maka akan muncul data dengan ID = 2))

PUT / UPDATE

Disini, kita akan mengubah data berdasarkan ID. Co. ID = 2 Lalu masukan data terbaru.

The screenshot shows a browser window with the address bar displaying <https://localhost:5001/swagger/index.html>. A warning icon indicates the connection is not secure.

The main content is a Swagger UI interface for a **PUT** request to `/api/Payment/{id}`. On the left, under **Parameters**, there is one parameter listed:

Name	Description
<code>id</code> <small>* required</small>	<code>integer(\$int32)</code> (path)

The value for `id` is set to `2`.

Below the parameters, the **Request body** section is shown. The **Content-Type** is set to `application/json`. The request body contains the following JSON object:

```
{  
    "paymentDetailId": 2,  
    "cardOwnerName": "Dini Tri Widianingsih",  
    "cardNumber": "2221-1234-5678-9012",  
    "expirationDate": "2022/10/26",  
    "securityCode": "Dini123"  
}
```

At the top right of the interface, there is a **Cancel** button.

Screenshot of a browser showing the Swagger UI for a REST API. The URL is <https://localhost:5001/swagger/index.html>. The page displays a "Responses" section for a PUT request to "/api/Payment/2". It includes a "Curl" command, a "Request URL" (<https://localhost:5001/api/Payment/2>), and a "Server response" section. The server response shows a 200 status code with a "Response body" containing "Data Updated!" and a "Download" button. Below the response is a "Response headers" section with content-type: text/plain; charset=utf-8, date: Wed 30 Mar 2022 12:17:59 GMT, and server: Kestrel. The "Responses" section also lists a 200 status code with a "Description" of "Success" and a "Links" section indicating "No links".

((Data Berhasil Di Update))

DELETE

Disini kita akan menghapus data berdasarkan id. Co ID = 1

Screenshot of a browser showing the Swagger UI for a REST API. The URL is <https://localhost:5001/swagger/index.html>. The page displays a "DELETE /api/Payment/{id}" operation. It includes a "Parameters" section with a parameter "id" (required, integer(Sint32), path) set to 1. Below the parameters is an "Execute" button and a "Clear" button. The "Responses" section for a 200 status code shows a "Curl" command, a "Request URL" (<https://localhost:5001/api/Payment/1>), and a "Server response" section. The server response shows a 200 status code with a "Response body".

Curl

```
curl -X DELETE "https://localhost:5001/api/Payment/1" -H "accept: */*" -H "Authorization: Bearer eyJhbGciOiJzIz1N1iisInRcC16IkXVCj9.eyJzC16Ijd1Mj1INDgxLTrwYmYnGzNy04YjIwLWeYmYxZTy0NGE1MCIsImVtYmIsIjoicmVubm13MTJAZ21haWwuY29tIiwiic3ViIiijoiMcVubm13MTJAZ21haWwuY29tIiwiianRpIjoiNzEyN2l3Nmtfzjpw0y0MDJkltgzTETn2E3YmU0GE47jgzIiwbmJmIjoxnjq4NjQyMtg4LC1leHaiOjE2NDg2NDM5ODgsImIhdC16MTY0ODY0Mj40H0.hMAx1BF7kux8NkZB4rTjY80zikjIwbd--dImhPIQPesw"
```

Request URL

<https://localhost:5001/api/Payment/1>

Server response

Code	Details
200	<p>Response body</p> <pre>{ "paymentId": "1111", "cardHolderName": "Reni Widiastuty", "cardNumber": "54453", "expirationDate": "2022/12/23", "securityCode": "Ren123" }</pre> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Wed, 30 Mar 2022 12:18:53 GMT server: Kestrel</pre>

Responses

Code	Description	Links
200	Success	No links

((Respons data telah berhasil dihapus))

CEK KEMBALI MELALUI GET PAYMENT

The screenshot shows a browser window with the address bar displaying "Not secure | https://localhost:5001/swagger/index.html". The main content area is titled "Payment". It shows a "GET /api/Payment" operation. Under "Parameters", it says "No parameters". Below that are "Execute" and "Clear" buttons. Under "Responses", there is a "Curl" section with a command line, a "Request URL" section with "https://localhost:5001/api/Payment", and a "Server response" section. The "Code" section shows a status code of 200 and a "Response body" section containing an empty JSON object [].

Curl

```
curl -X GET "https://localhost:5001/api/Payment" -H "accept: */*" -H "Authorization: Bearer eyJhbGciOiJIUzI1NiIsInRS...[redacted]
```

Request URL

<https://localhost:5001/api/Payment>

Server response

Code	Details
200	Response body <pre>[{ "paymentDetailId": 2, "cardOwnerName": "Dini Tri Widianingsih", "cardNumber": "22113", "expirationDate": "2022/10/26", "securityCode": "Dini123" }, { "paymentDetailId": 3, "cardOwnerName": "Miwik Supiyati", "cardNumber": "22345", "expirationDate": "2022/09/21", "securityCode": "Miwik123" }]</pre> <p>Download</p> Response headers <pre>content-type: application/json; charset=utf-8 date: Wed, 30 Mar 2022 12:19:46 GMT server: Kestrel</pre>

((ID = 1 telah berhasil dihapus))

SWAGGER/WEATHERFORECAST

- Klik Try It Out lalu Execute

WeatherForecast

GET /WeatherForecast

Parameters

No parameters

Execute

Responses

Curl

```
curl -X GET "https://localhost:5001/WeatherForecast" -H "accept: text/plain" -H "Authorization: Bearer eyJhbGciOiJIUzI1NiIsInRS...[redacted]
```

Request URL

<https://localhost:5001/WeatherForecast>

Server response

Code	Details
200	Response body <pre>[{ ... }]</pre>

Not secure | <https://localhost:5001/swagger/index.html>

Server response

Code Details

200 Response body

```
[{"date": "2022-03-31T19:20:23.2328599+07:00", "temperatureC": 3, "temperatureF": 37, "summary": "Hot"}, {"date": "2022-04-01T19:20:23.2332403+07:00", "temperatureC": -2, "temperatureF": 29, "summary": "Balmy"}, {"date": "2022-04-02T19:20:23.2332427+07:00", "temperatureC": 44, "temperatureF": 111, "summary": "Sweltering"}, {"date": "2022-04-03T19:20:23.2332429+07:00", "temperatureC": 4, "temperatureF": 39, "summary": "Cool"}, {"date": "2022-04-04T19:20:23.2332433+07:00", "temperatureC": 50, "temperatureF": 122}],
```

Download

Response headers

```
content-type: application/json; charset=utf-8
date: Wed 30 Mar 2022 12:20:22 GMT
server: Kestrel
```

Responses

Type here to search

19:20 30/03/2022

Not secure | <https://dini-finalproject.herokuapp.com/weatherforecast>

```
[{"date": "2022-03-31T16:35:42.5180397+00:00", "temperatureC": -12, "temperatureF": 11, "summary": "Freezing"}, {"date": "2022-04-01T16:35:42.5182624+00:00", "temperatureC": 10, "temperatureF": 49, "summary": "Mild"}, {"date": "2022-04-02T16:35:42.5182639+00:00", "temperatureC": 50, "temperatureF": 121, "summary": "Sweltering"}, {"date": "2022-04-03T16:35:42.5182642+00:00", "temperatureC": 47, "temperatureF": 116, "summary": "Bracing"}, {"date": "2022-04-04T16:35:42.5182645+00:00", "temperatureC": 7, "temperatureF": 44, "summary": "Hot"}]
```

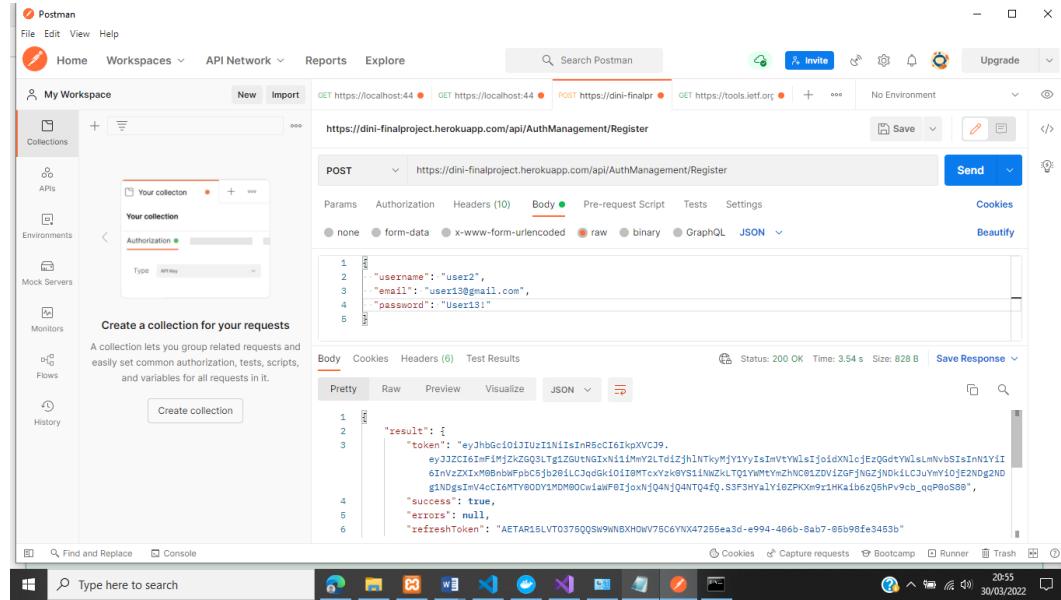
23:35 30/03/2022

POSTMAN

REGISTER

Link : <https://dini-finalproject.herokuapp.com/api/AuthManagement/Register>

- Masukan username, email, dan password baru



The screenshot shows the Postman interface with a POST request to <https://dini-finalproject.herokuapp.com/api/AuthManagement/Register>. The request body is set to JSON and contains the following data:

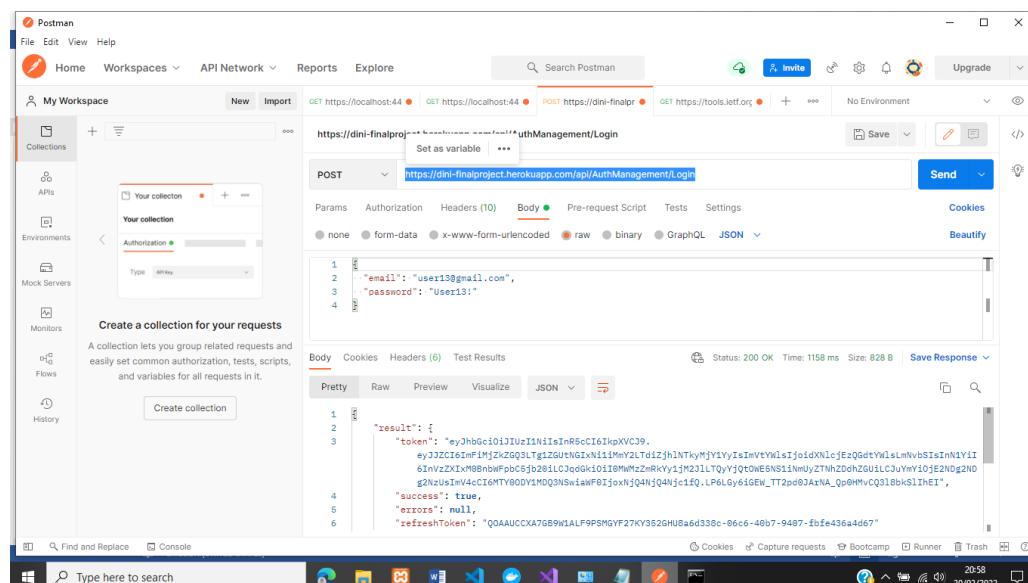
```
1 "username": "userx2",
2 "email": "user13@gmail.com",
3 "password": "User13!"
```

The response status is 200 OK, and the response body is a large JSON token.

LOGIN

- Silahkan login menggunakan email dan password yang telah dibuat.
- Lalu akan didapatkan token untuk melakukian autentikasi dimana token hanya berlaku selama 30 menit.

Link : <https://dini-finalproject.herokuapp.com/api/AuthManagement/Login>



The screenshot shows the Postman interface with a POST request to <https://dini-finalproject.herokuapp.com/api/AuthManagement/Login>. The request body is set to JSON and contains the following data:

```
1 "email": "user13@gmail.com",
2 "password": "User13!"
```

The response status is 200 OK, and the response body is a large JSON token.

REFRESH TOKEN

Link : <https://dini-finalproject.herokuapp.com/api/AuthManagement/RefreshToken>

Percobaan melakukan refresh token sebelum 30 menit akan menampilkan pesan gagal.

The screenshot shows the Postman application interface. A collection named 'Your collection' is selected. A POST request is made to <https://dini-finalproject.herokuapp.com/api/AuthManagement/RefreshToken>. The response status is 200 OK, time 1532 ms, size 277 B. The response body is:

```
1 "token": null,
2 "success": false,
3 "errors": [
4     "Token has not yet expired"
5 ],
6 "refreshToken": null
```

AUTHORIZE

Melakukan autorisasi dengan menambahkan key baru “Authorization” dan diisi dengan Bearer key dimana key adalah token yang didapatkan dari hasil login.

Link : <https://dini-finalproject.herokuapp.com/api/Payment>

The screenshot shows the Postman application interface. A collection named 'My first collection' is selected. A POST request is made to <https://dini-final-project.herokuapp.com/api/AuthManagement/Login>. The response status is 200 OK, time 2.13 s, size 833 B. The response body is:

```
1 "result": {
2     "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
3         eyJzIzC161jd1Mj1lNDgxLTRmYyNDgNy04Yj1wMwYyZTY0NGE1MCIsImVtYWhsIjoicmVubml3MTJAZ21haWwUY29tIiwic3V
4         iijoiicVubm13MTJAZ21haWwUY29tIiwanRoiIoinMw1NTi4MTMtnjYxNC00NDU4LwI4YjgtYmY3ZWRI0dg0MTVkiIwibmJiIoxkj
5         Q4NjUSMjMyLC1ehA10je2N0g2NjEwMzIsImlhdCI6MTY0ODY1OTizMn0.pk9hGvHkznmBGGNuObkLex0X71ZYNQHs5n19mMsq8",
6     "success": true,
```

PAYMENT POST

Membuat data payment baru dengan mengisi id, nama, tanggal expired, dan kode keamanan.

Link : <https://dini-finalproject.herokuapp.com/api/payment>

The screenshot shows the Postman application interface. In the center, there is a request configuration window for a POST method to the URL <https://dini-finalproject.herokuapp.com/api/payment>. The 'Body' tab is selected, showing a JSON payload:

```
1   ...
2   ...
3   ...
4   ...
5   ...
6   ...
7   ...
8   ...
9   ...
10  ...
11  ...
12  ...
13  ...
14  ...
15  ...
16  ...
17  ...
18  ...
19  ...
```

The payload contains fields: cardOwnerName, cardNumber, expirationDate, and securityCode. The response status is 201 Created, and the body of the response is identical to the request payload.

PAYMENT GET

Mendapatkan Semua data yang ada didalam database.

Link : <https://dini-finalproject.herokuapp.com/api/payment>

The screenshot shows the Postman application interface. In the center, there is a request configuration window for a GET method to the URL <https://dini-finalproject.herokuapp.com/api/payment>. The 'Body' tab is selected, showing a JSON response:

```
1  [
2    {
3      "paymentDetailId": 2,
4      "cardOwnerName": "Dini Tri Widianingsih",
5      "cardNumber": "22213",
6      "expirationDate": "2022/10/26",
7      "securityCode": "Dini123"
8    },
9    {
10      "paymentDetailId": 3,
11      "cardOwnerName": "Wiwik Supiyati",
12      "cardNumber": "23456",
13      "expirationDate": "2022/09/25",
14      "securityCode": "Wiwik123"
15    },
16    {
17      "paymentDetailId": 4,
18      "cardOwnerName": "Dicky Sastro",
19      "cardNumber": "22234",
```

The response status is 200 OK, and the body contains a list of payment details, each with an ID, owner name, card number, expiration date, and security code.

PAYMENT GET WHERE ID

Mendapatkan data payment details dengan ID yang dimasukan.

Link : <https://dini-finalproject.herokuapp.com/api/payment/2>

The screenshot shows the Postman application interface. In the center, there is a request card for a GET request to `https://dini-finalproject.herokuapp.com/api/payment/2`. The response status is 200 OK, and the response body is displayed in JSON format:

```
1
2   "paymentDetailId": 2,
3   "cardOwnerName": "Dini Tri Widianingsih",
4   "cardNumber": "22213",
5   "expirationDate": "2022/10/26",
6   "securityCode": "Dini123"
7
```

PAYMENT PUT

Mengubah data dengan ID tertentu lalu menampilkan respons perubahan.

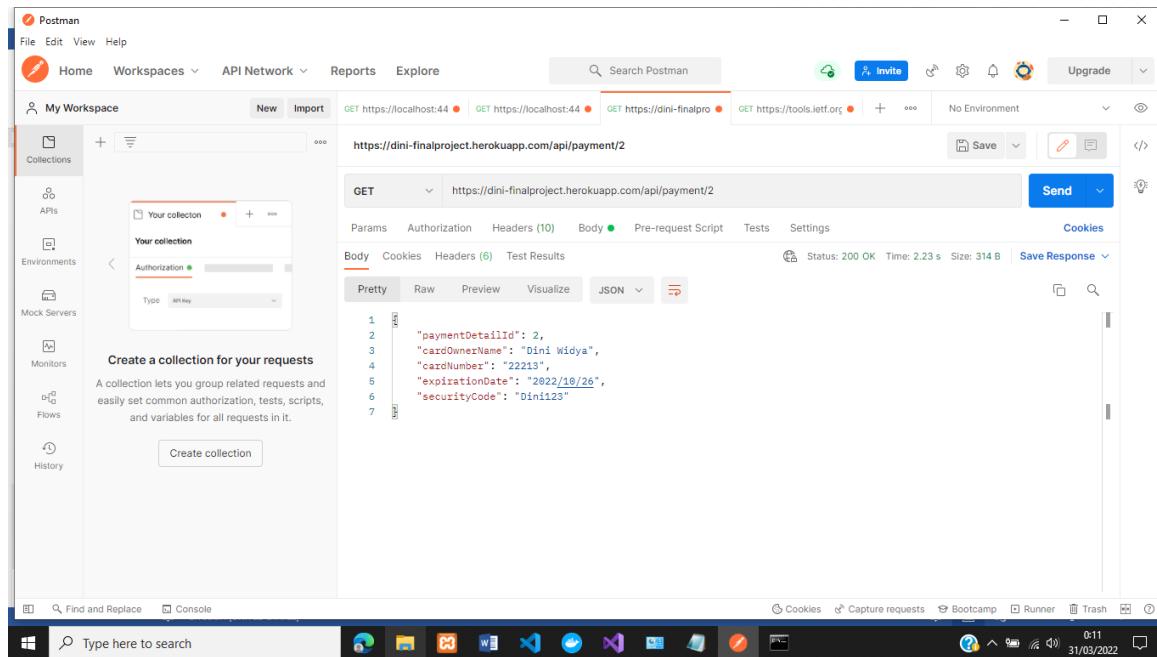
Link : <https://dini-finalproject.herokuapp.com/api/payment/2>

The screenshot shows the Postman application interface. In the center, there is a request card for a PUT request to `https://dini-finalproject.herokuapp.com/api/payment/2`. The response status is 200 OK, and the response body is displayed in JSON format:

```
1
2   "paymentDetailId": 2,
3   "cardOwnerName": "Dini Widya",
4   "cardNumber": "22213",
5   "expirationDate": "2022/10/26",
6   "securityCode": "Dini123"
7
```

Below the response, the text "Data Updated!" is visible.

((Cek Get Payment Kembali))



The screenshot shows the Postman application interface. On the left, the sidebar includes sections for Collections, APIs, Environments, Mock Servers, Monitors, Flows, and History. The main workspace displays a collection named "Your collection". A specific request is selected, showing a GET method directed at <https://dini-finalproject.herokuapp.com/api/payment/2>. The response status is 200 OK, with a response body containing JSON data:

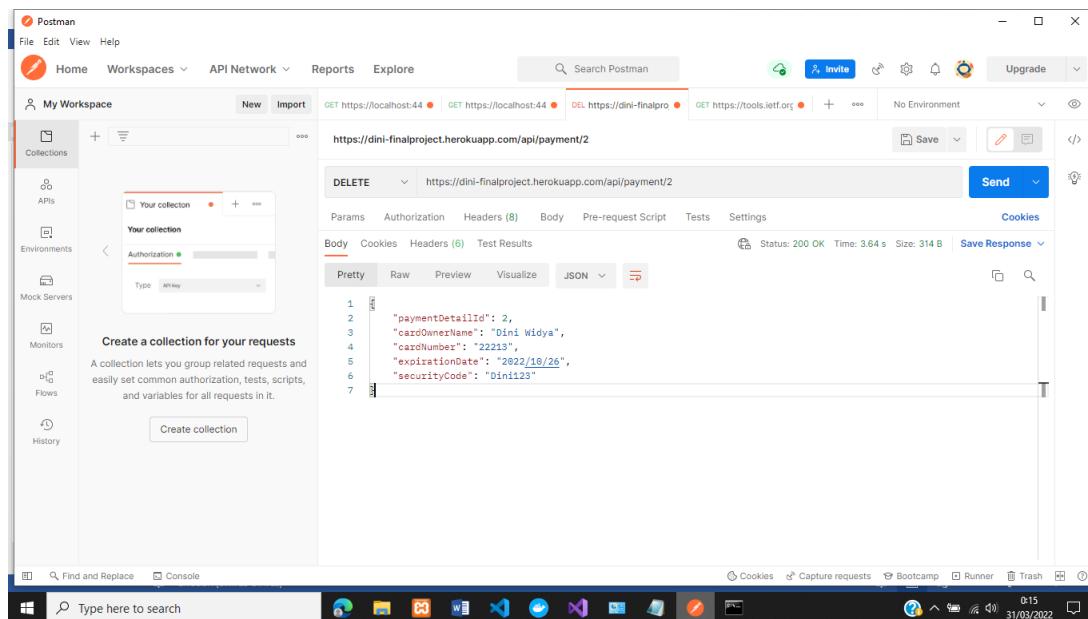
```
1  {
2      "paymentDetailId": 2,
3      "cardOwnerName": "Dini Widya",
4      "cardNumber": "22213",
5      "expirationDate": "2022/10/26",
6      "securityCode": "Dini123"
7 }
```

Data berhasil di ubah.

DELETE PAYMENT

Menghapus data dengan ID tertentu dan menampilkan response berupa data yang berhasil dihapus.

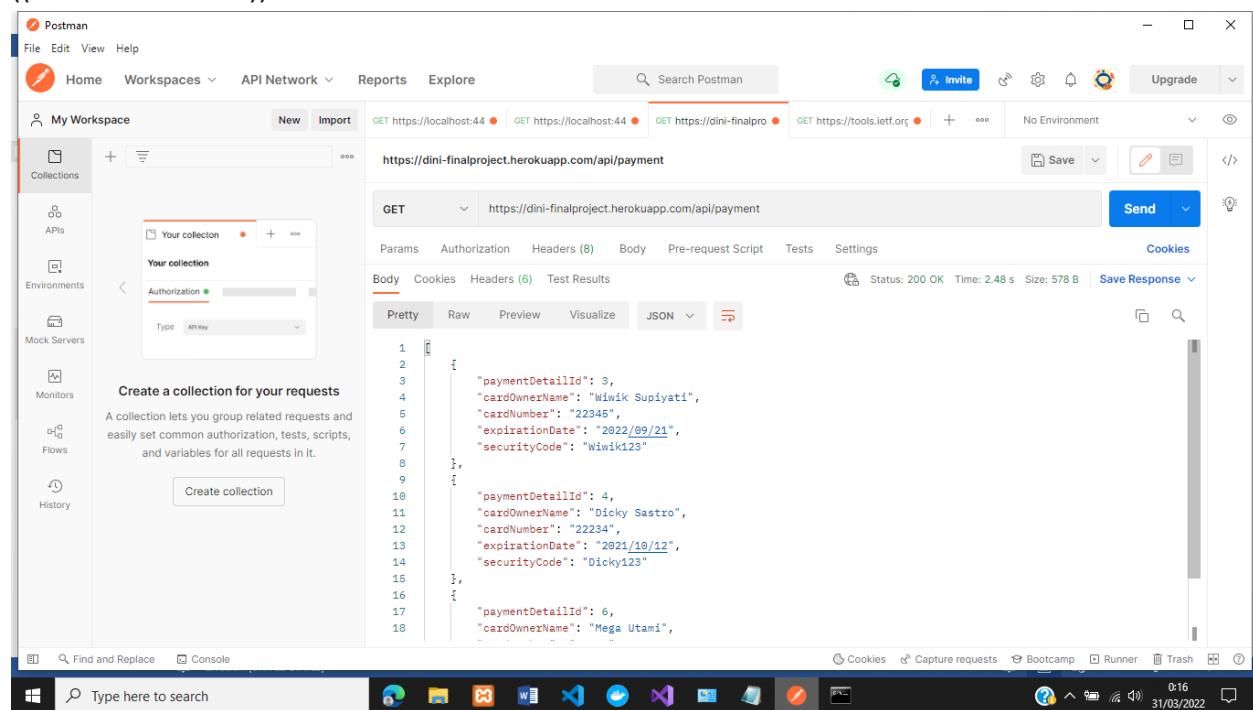
Link : <https://dini-finalproject.herokuapp.com/api/payment/2>



The screenshot shows the Postman application interface. The sidebar and collection structure are identical to the previous screenshot. A new request is selected, showing a DELETE method directed at <https://dini-finalproject.herokuapp.com/api/payment/2>. The response status is 200 OK, with a response body containing the same JSON data as the previous screenshot:

```
1  {
2      "paymentDetailId": 2,
3      "cardOwnerName": "Dini Widya",
4      "cardNumber": "22213",
5      "expirationDate": "2022/10/26",
6      "securityCode": "Dini123"
7 }
```

((Cek GET Kembali))



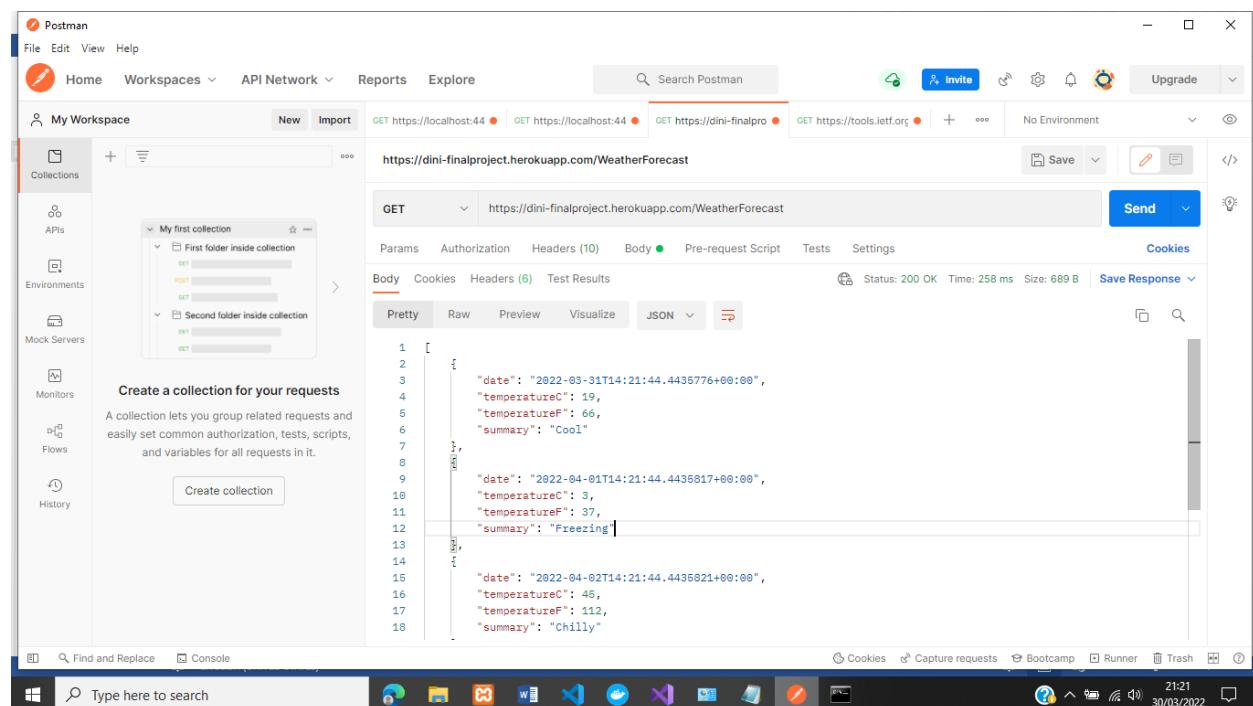
The screenshot shows the Postman application interface. On the left, there's a sidebar with options like Home, Workspaces, API Network, Reports, and Explore. The main area shows a collection named "Your collection" with a single item labeled "Your collection". A modal window is open for a GET request to "https://dini-finalproject.herokuapp.com/api/payment". The response status is 200 OK, and the response body is displayed in JSON format:

```
1  {
2   "paymentDetailId": 3,
3   "cardOwnerName": "Wiwik Supiyati",
4   "cardNumber": "22348",
5   "expirationDate": "2022/09/21",
6   "securityCode": "Wiwik123"
7 },
8 {
9   "paymentDetailId": 4,
10  "cardOwnerName": "Dicky Sastro",
11  "cardNumber": "22234",
12  "expirationDate": "2021/10/12",
13  "securityCode": "Dicky123"
14 },
15 {
16  "paymentDetailId": 6,
17  "cardOwnerName": "Mega Utami",
18 }
```

ID = 2 sudah tidak ada di dalam database.

GET WEATHERFORECAST

Link : <https://dini-finalproject.herokuapp.com/WeatherForecast>



The screenshot shows the Postman application interface. On the left, there's a sidebar with options like Home, Workspaces, API Network, Reports, and Explore. The main area shows a collection named "My first collection" with two folders: "First folder inside collection" and "Second folder inside collection", each containing several requests. A modal window is open for a GET request to "https://dini-finalproject.herokuapp.com/WeatherForecast". The response status is 200 OK, and the response body is displayed in JSON format:

```
1 [
2   {
3     "date": "2022-03-31T14:21:44.4435776+00:00",
4     "temperatureC": 19,
5     "temperatureF": 66,
6     "summary": "Cool"
7   },
8   {
9     "date": "2022-04-01T14:21:44.4435817+00:00",
10    "temperatureC": 3,
11    "temperatureF": 37,
12    "summary": "Freezing"
13  },
14  {
15    "date": "2022-04-02T14:21:44.4435821+00:00",
16    "temperatureC": 45,
17    "temperatureF": 112,
18    "summary": "Chilly"
19  }
20 ]
```

SAVE HASIL CRUD DARI POSTMAN

 1. Response Register	31/03/2022 0:19	JSON Source File	1 KB
 2. Response Login	31/03/2022 0:20	JSON Source File	1 KB
 3. Response Refresh Token	31/03/2022 0:22	JSON Source File	1 KB
 4. Response Post Payment	30/03/2022 23:59	JSON Source File	1 KB
 5. Respons Get Payment	31/03/2022 0:02	JSON Source File	1 KB
 6. Response Get Payment Where ID	31/03/2022 0:05	JSON Source File	1 KB
 7. Response Put Payment	31/03/2022 0:10	JSON Source File	1 KB
 8. Respons Delete Payment	31/03/2022 0:14	JSON Source File	1 KB
 9. Response WeatherForecast	31/03/2022 0:17	JSON Source File	1 KB