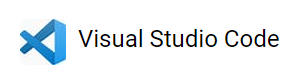
DEPLOYMENT +

GIT EXPLORATION

Muhammad Arif Rahman - FSDO003ONL004 – Dokumentasi Sesi 13



1. Aplikasi persiapan
   1. Visual Code
   2. Browser
   3. Xampp
   4. Docker
   5. VS Code
   6. SQLyog
   7. Postman

<https://docs.google.com/forms/u/0/d/e/1FAIpQLSca9QDX4Ufwr7lqvRxVqIg4nQYUQpjA9_fzoRY6nok7-FnM8Q/formResponse?pli=1>

=======================(terminal)===================================================

dotnet new webapi -n "TodoApp" -lang "C#" -au none

dotnet build

dotnet publish -c Release

docker build -t weatherimage:dev .

docker run -d -p 5051:80 --name weathercontainer weatherimage:dev

(buat file - DDockerfile - copy paste)

#See https://aka.ms/containerfastmode to understand how Visual Studio uses this Dockerfile to build your images for faster debugging.

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 ["TodoAppTest/TodoAppTest.csproj", "TodoAppTest/"]

#RUN dotnet restore "TodoAppTest/TodoAppTest.csproj"

#COPY . .

#WORKDIR "/src/TodoAppTest"

#RUN dotnet build "TodoAppTest.csproj" -c Release -o /app/build

RUN dotnet restore

RUN dotnet build --no-restore -c Release -o /app

FROM build AS publish

RUN dotnet publish "TodoAppTest.csproj" -c Release -o /app/publish

FROM base AS final

WORKDIR /app

#COPY --from=publish /app/publish .

COPY --from=publish /app/ .

#ENTRYPOINT ["dotnet", "TodoAppTest.dll"]

CMD ASPNETCORE\_URLS=http://\*:$PORT dotnet TodoAppTest.dll

================================================================

(aktifin swagger)

public void Configure(IApplicationBuilder app, IWebHostEnvironment env)

{

if (env.IsDevelopment())

{

app.UseDeveloperExceptionPage();

}

app.UseSwagger();

app.UseSwaggerUI(c => c.SwaggerEndpoint("/swagger/v1/swagger.json", "WeatherTest v1"));

app.UseHttpsRedirection();

app.UseRouting();

app.UseAuthorization();

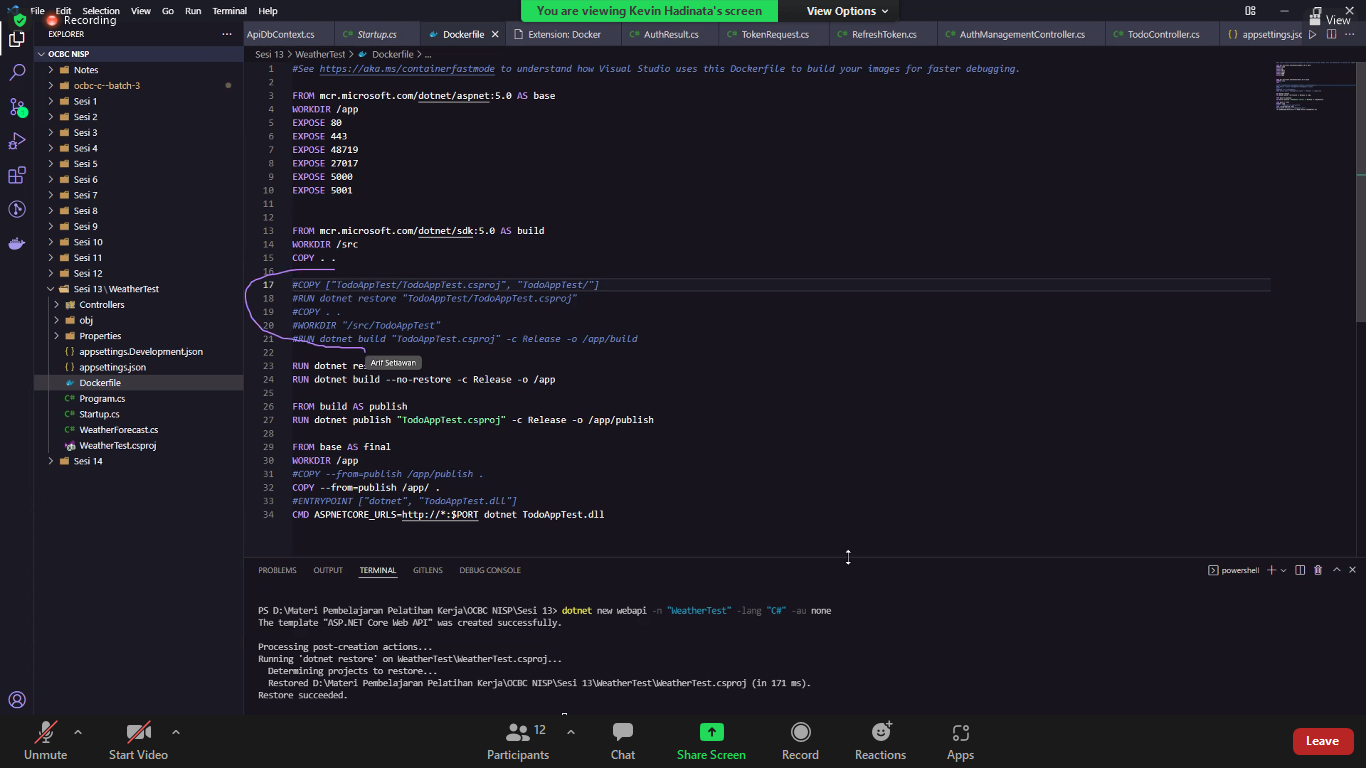
app.UseEndpoints(endpoints =>

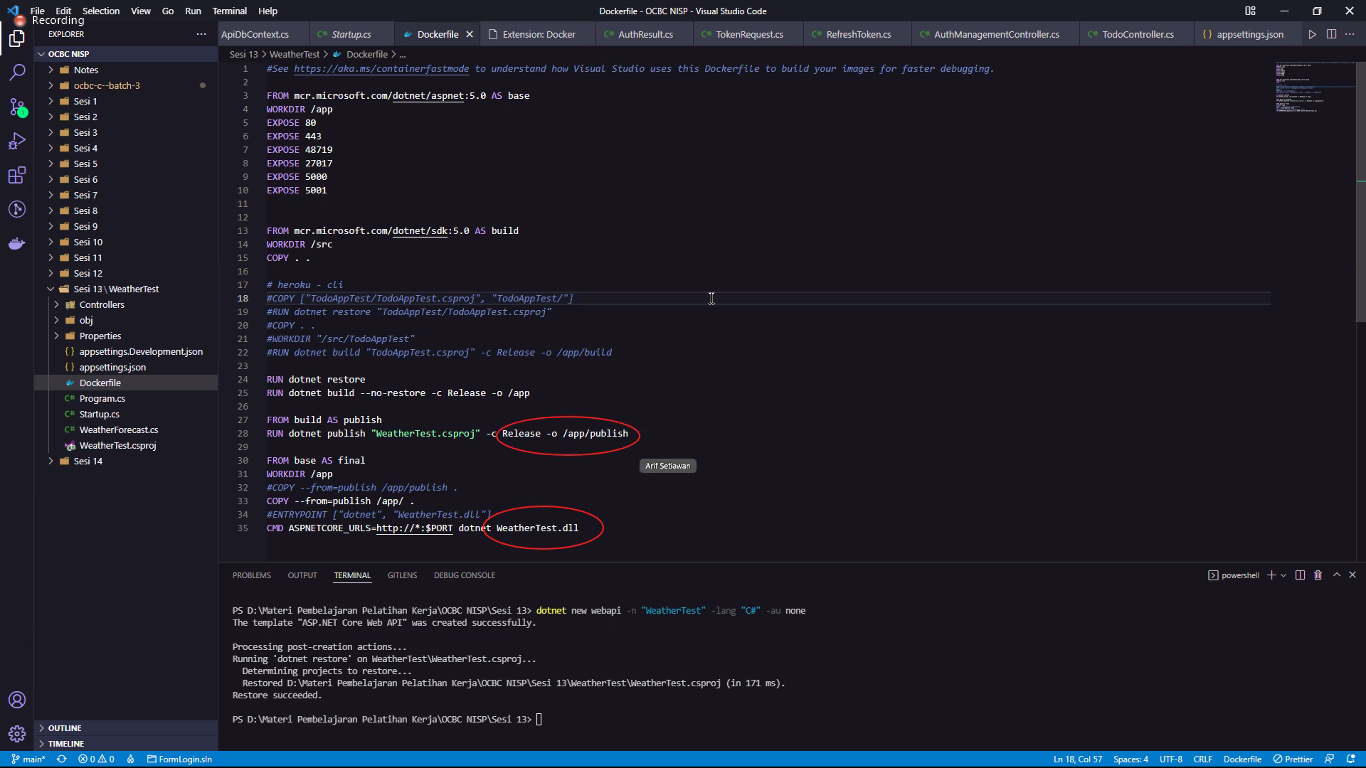
{

endpoints.MapControllers();

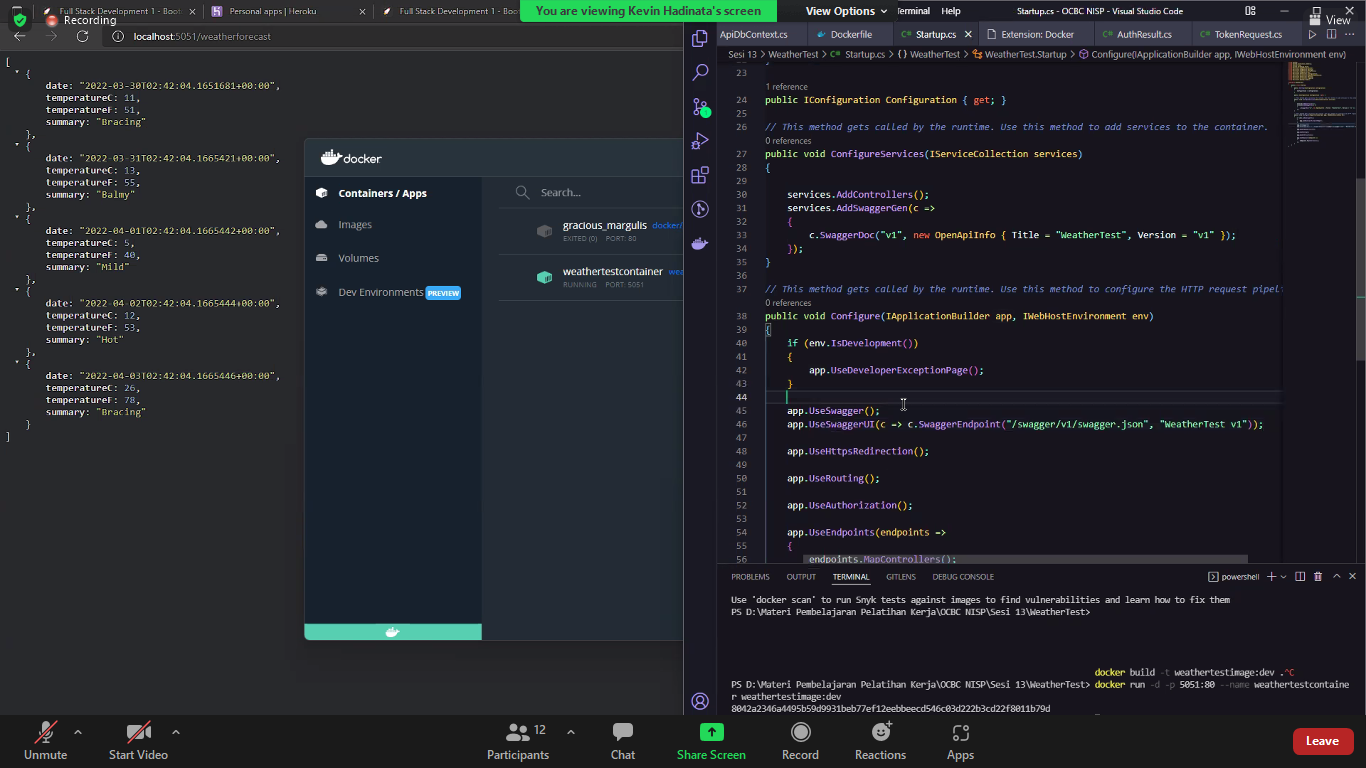
});

}

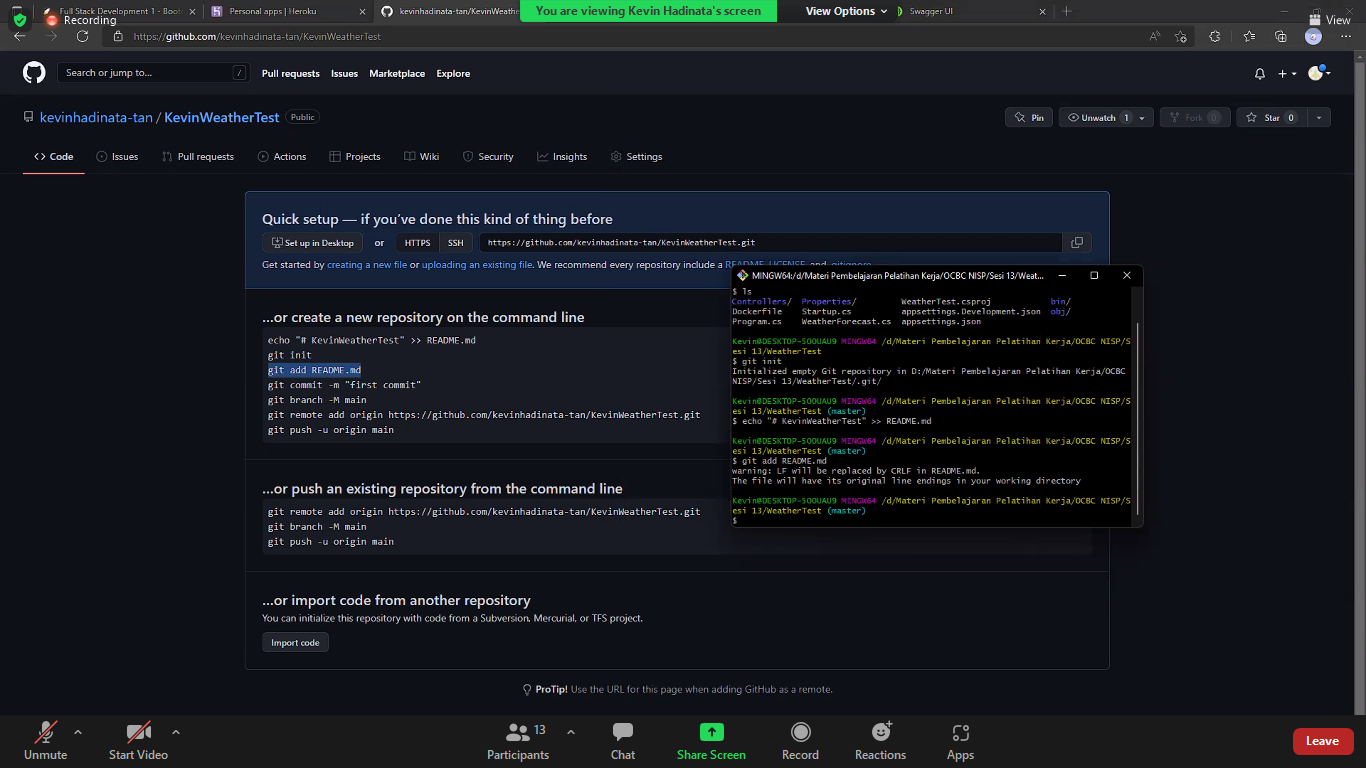


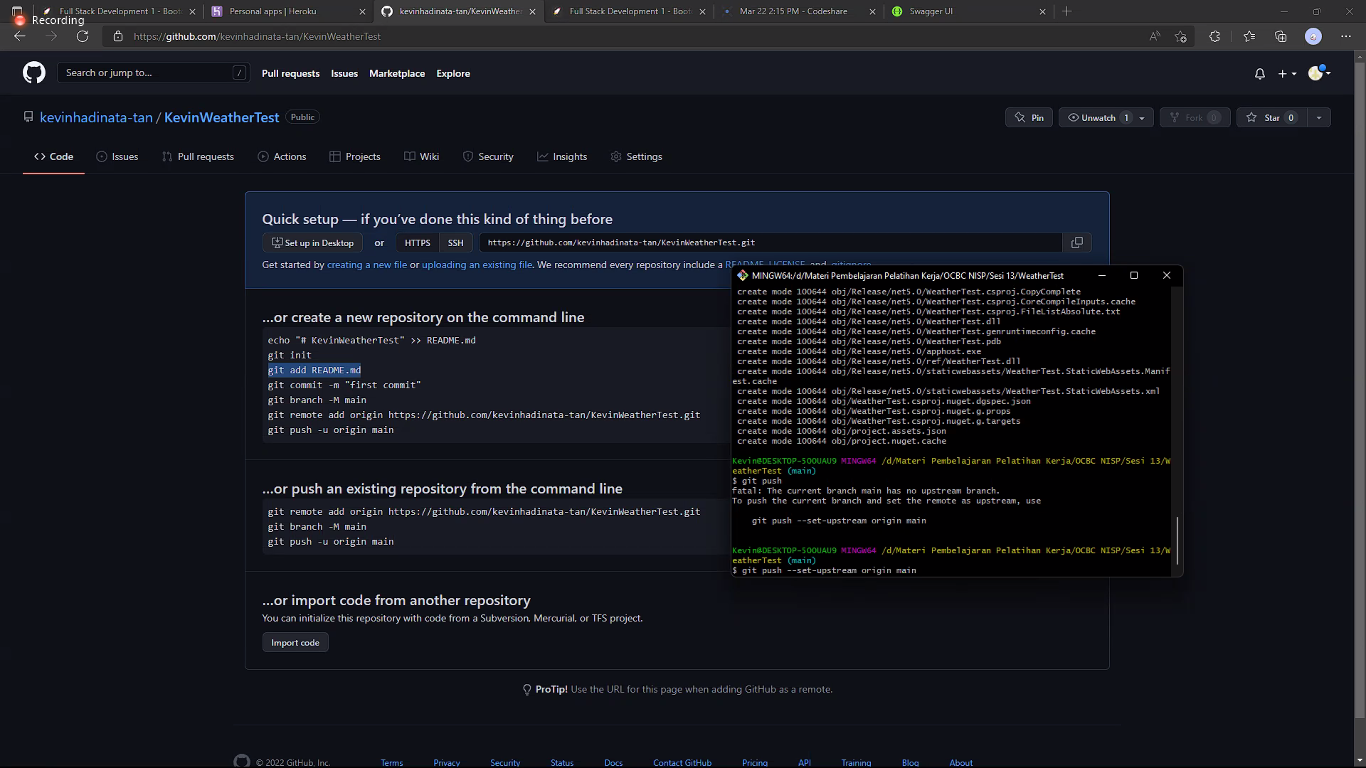
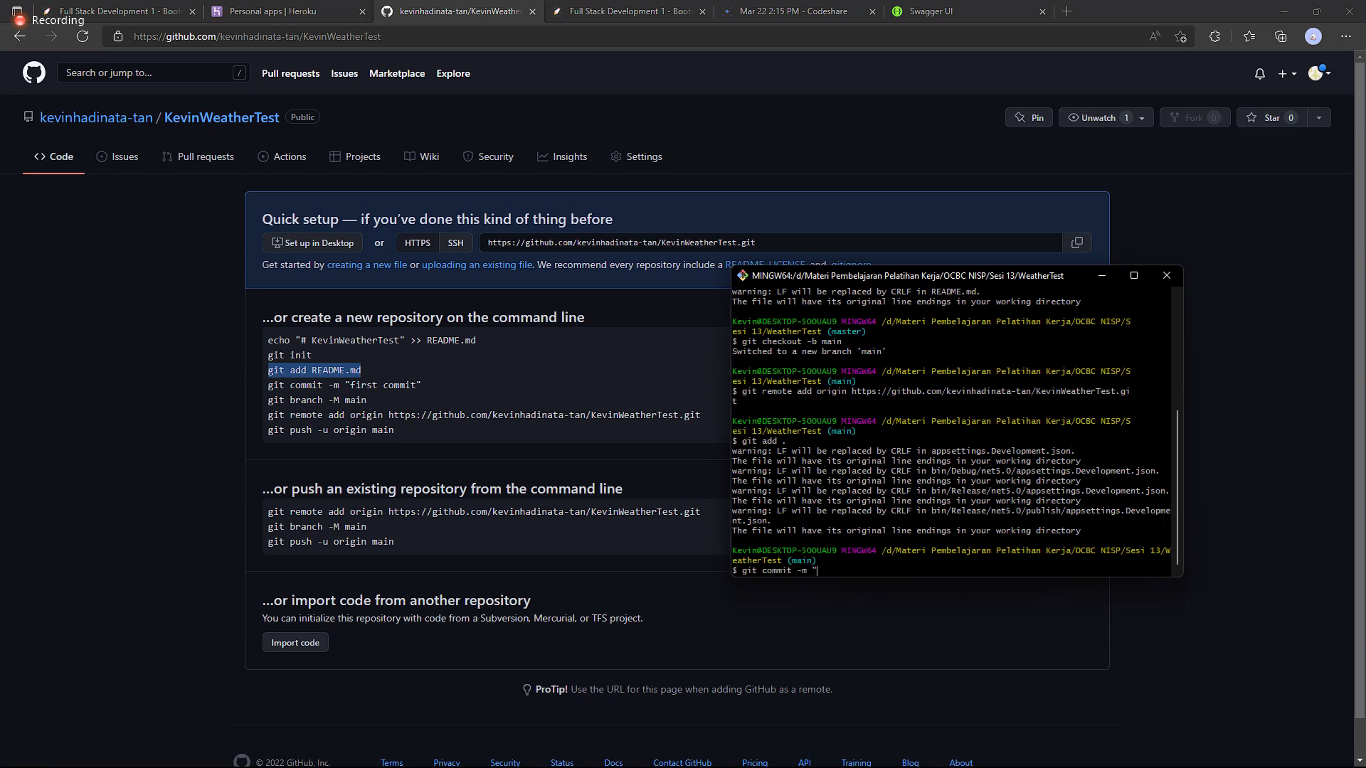


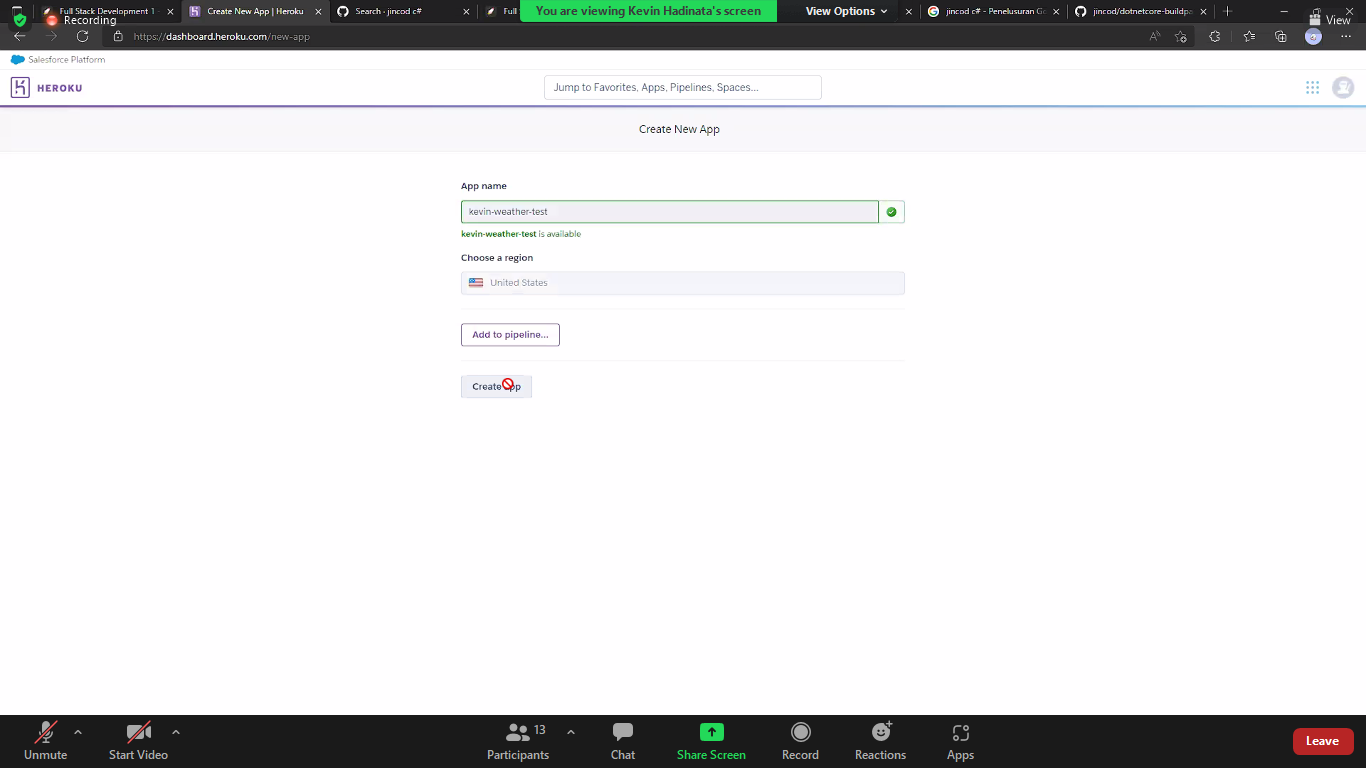
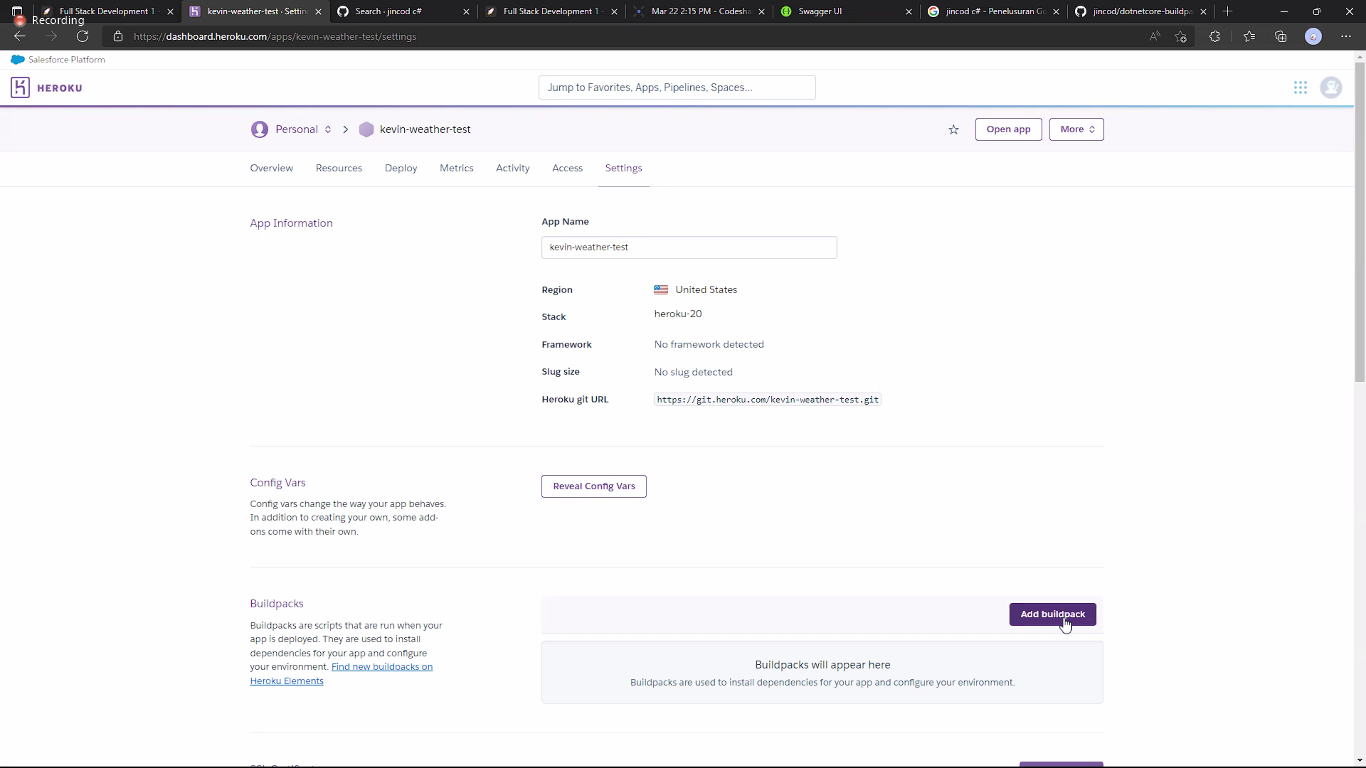
Untuk versi production untuk menampilkan swagger butuh dikeluarckan code di baris 45

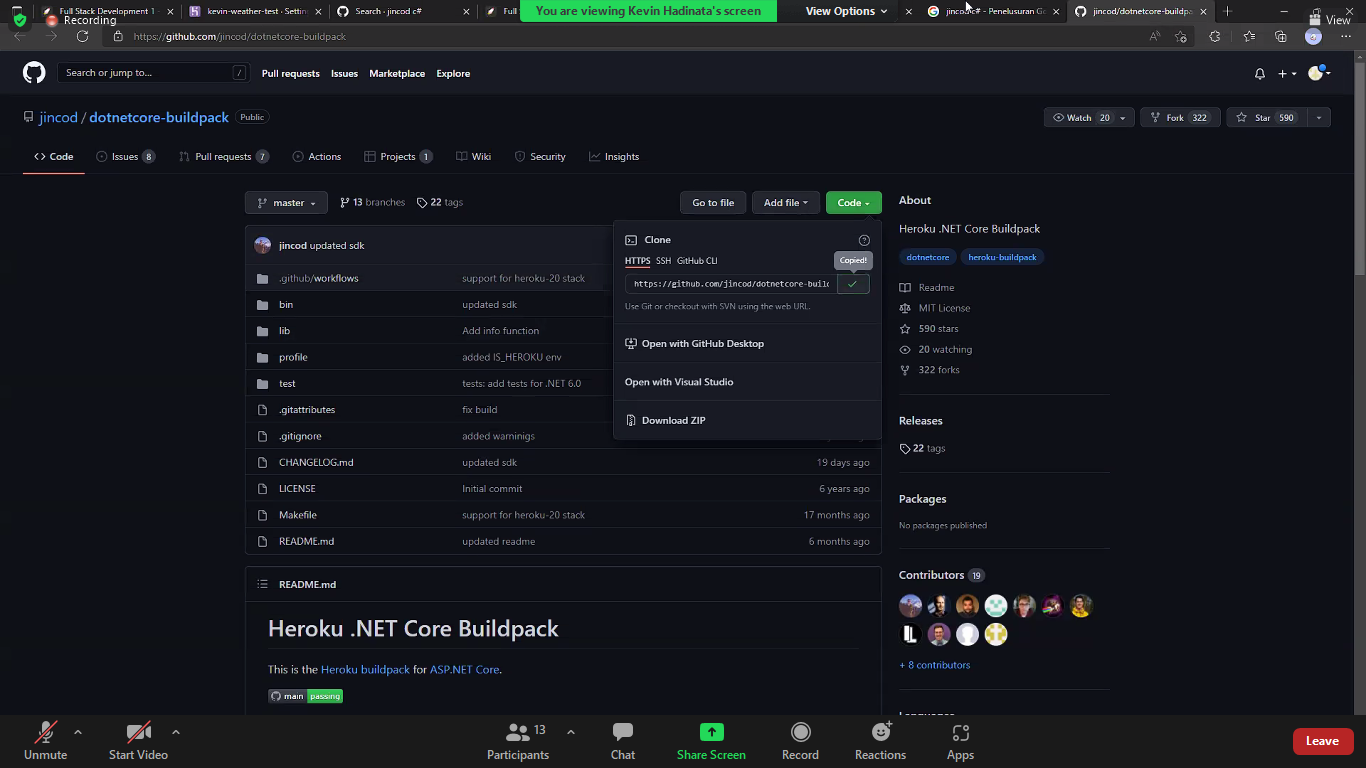


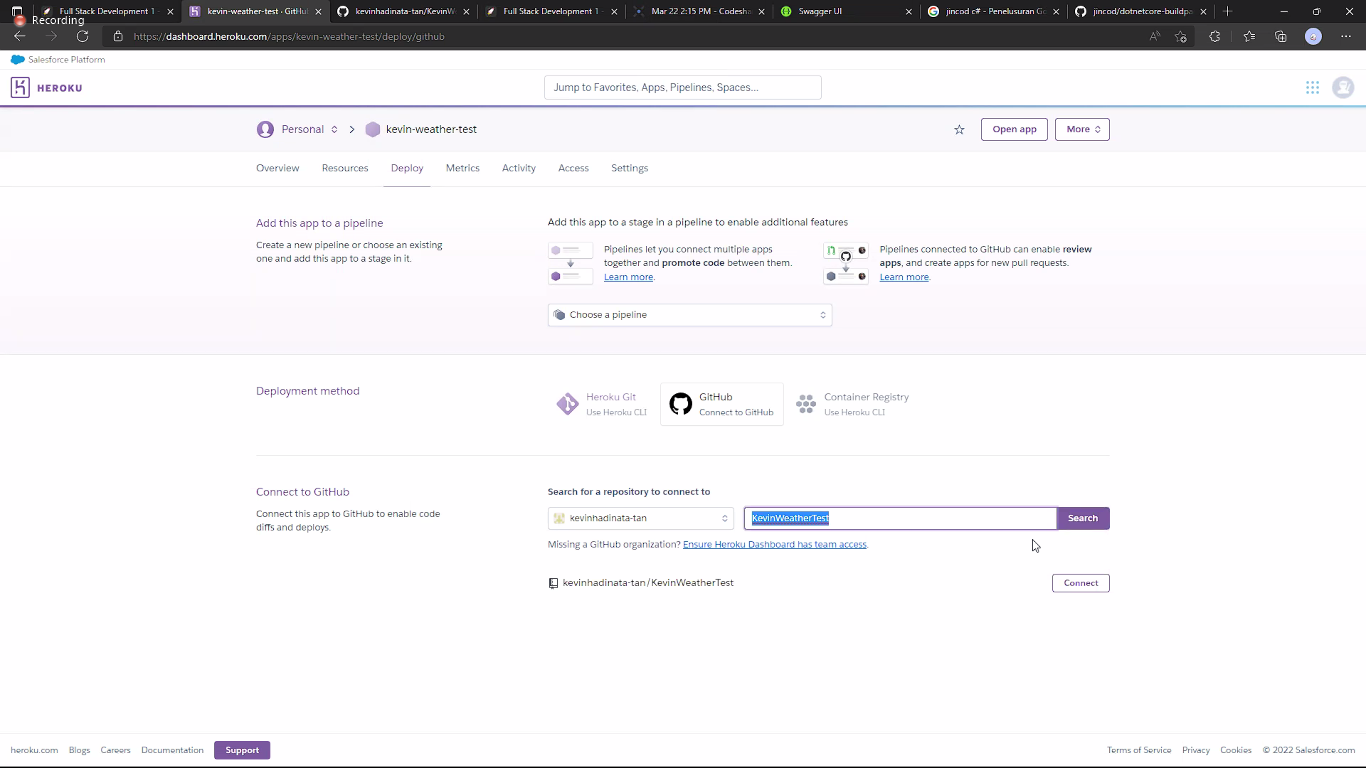
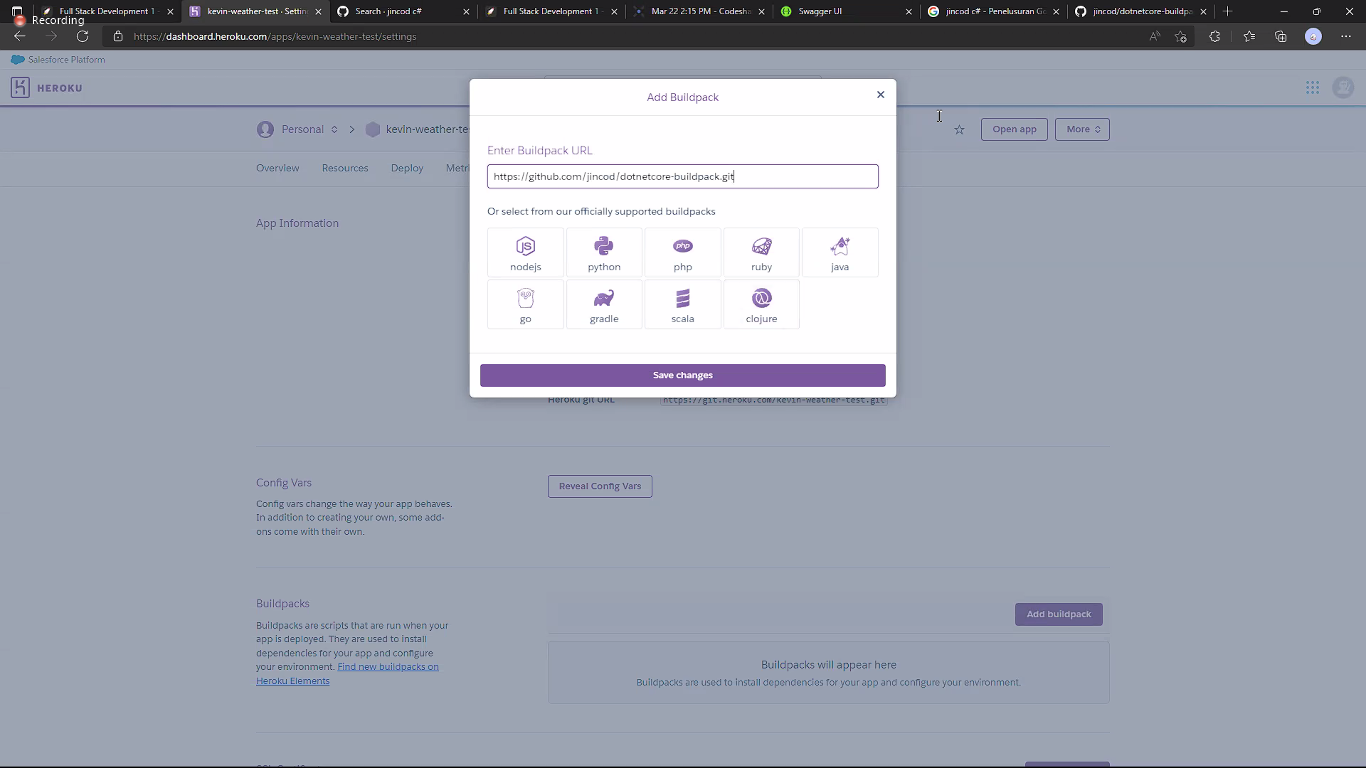
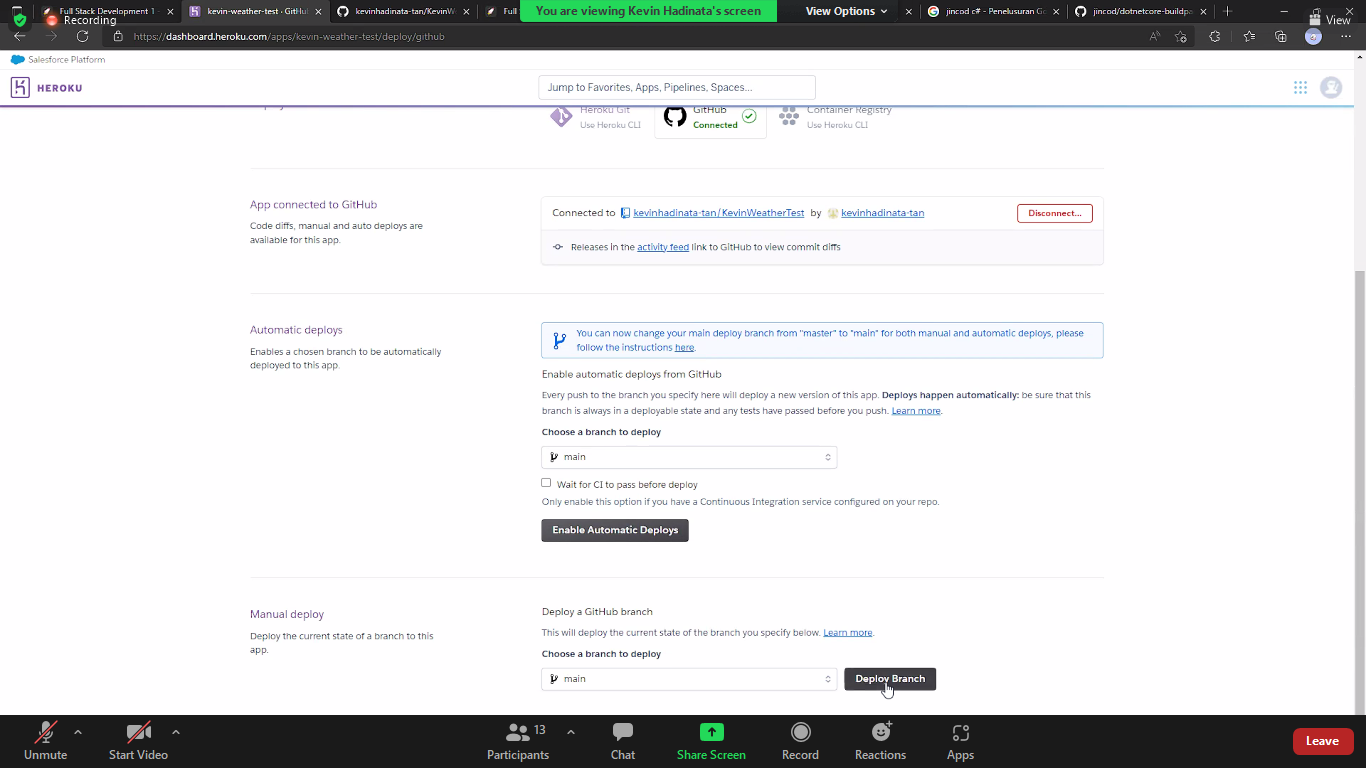
Weather container harus dihhapus

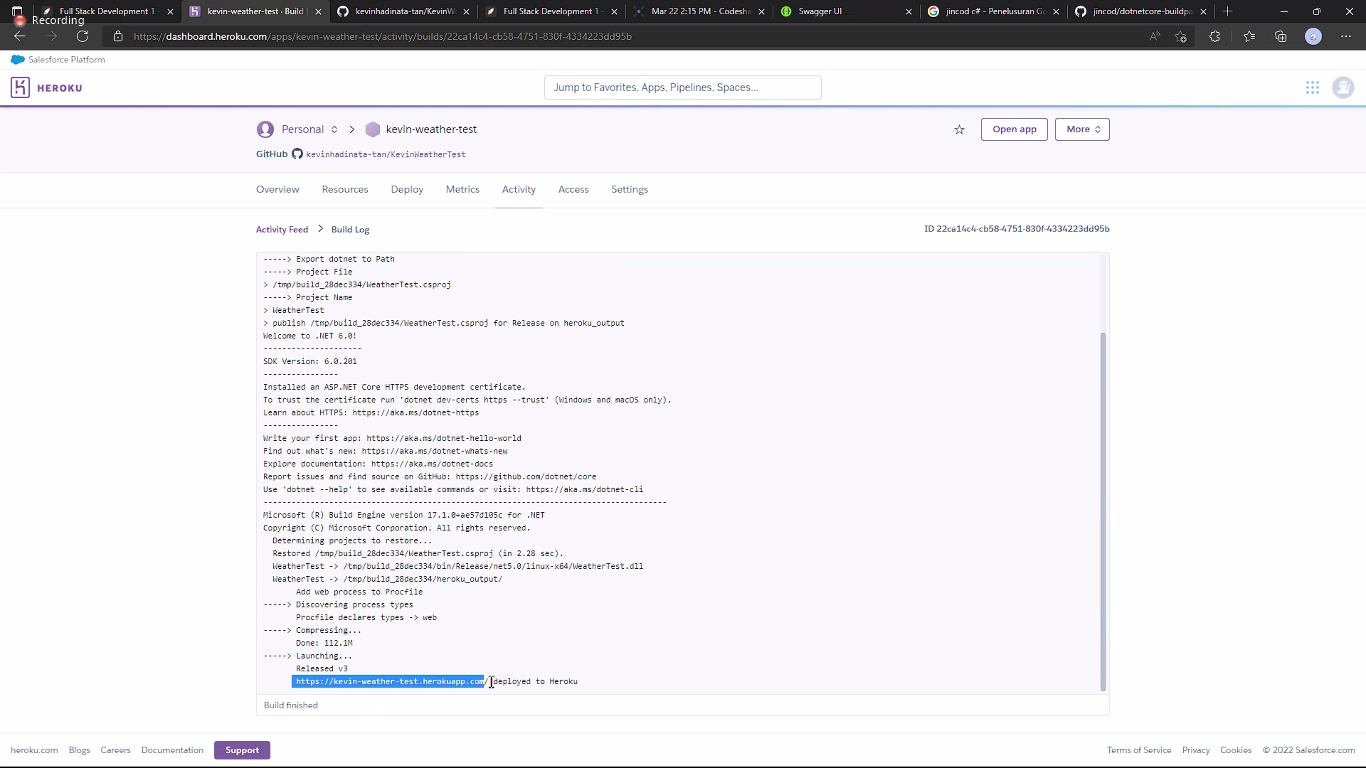
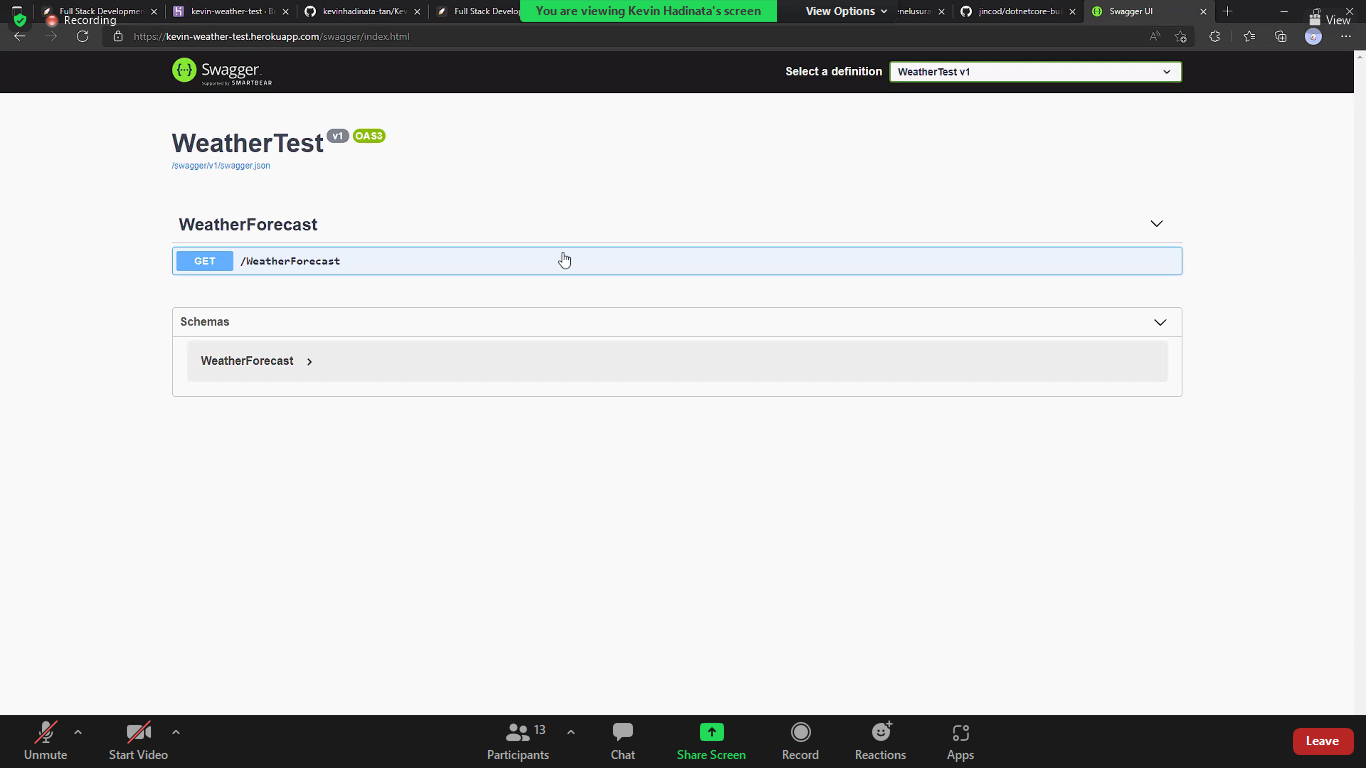


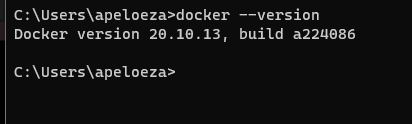
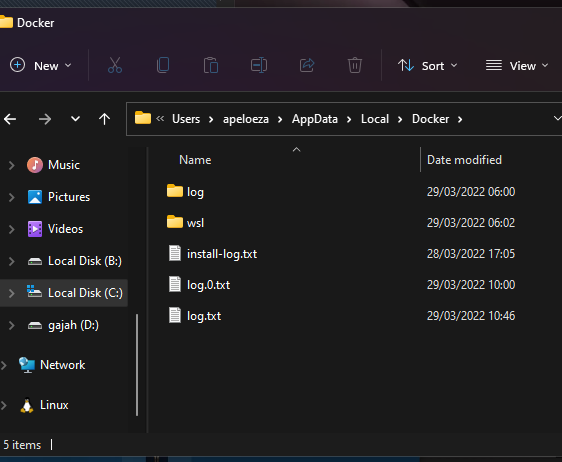
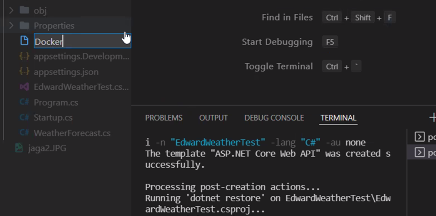
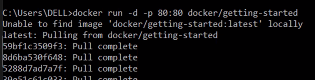


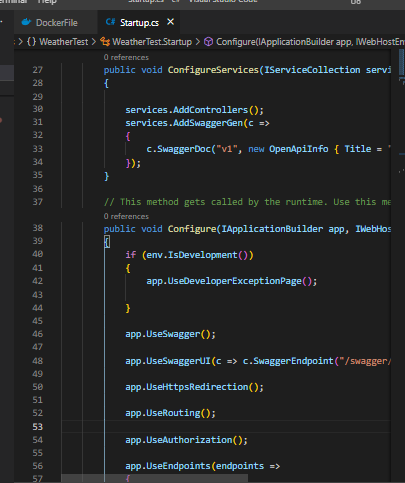
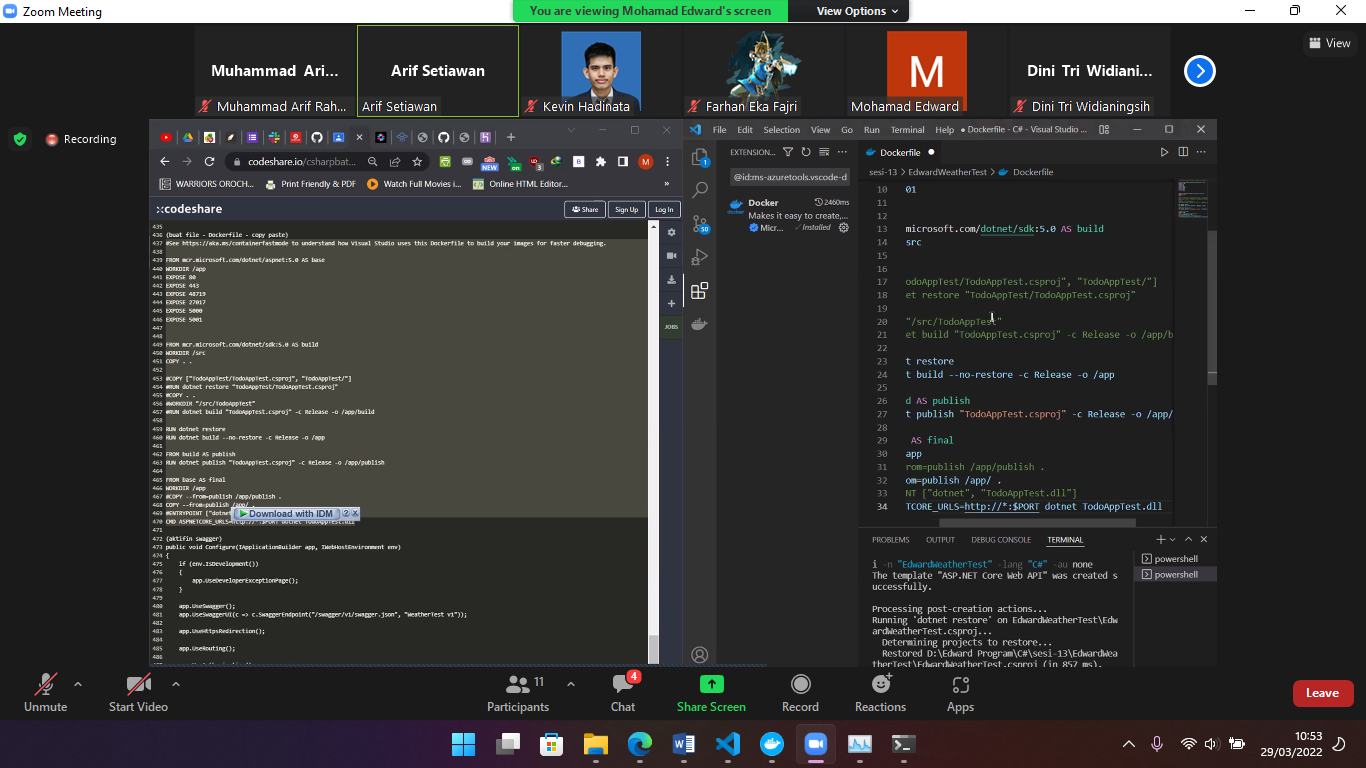
 



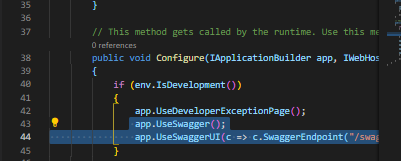
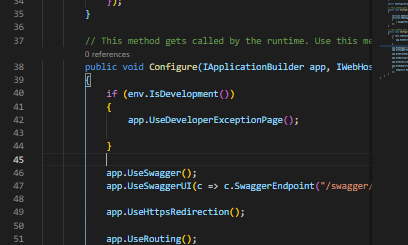


Install docker desktop

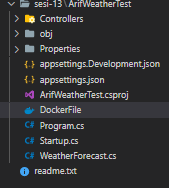
Run cmd lalu “ C:\Users\apeloeza>docker run -d -p 80:80 docker/getting-started”

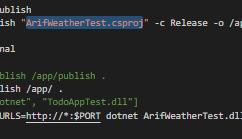
Run cmd in vs code to sesi-13 “ dotnet new webapi -n "WeatherTest" -lang "C#" -au none”

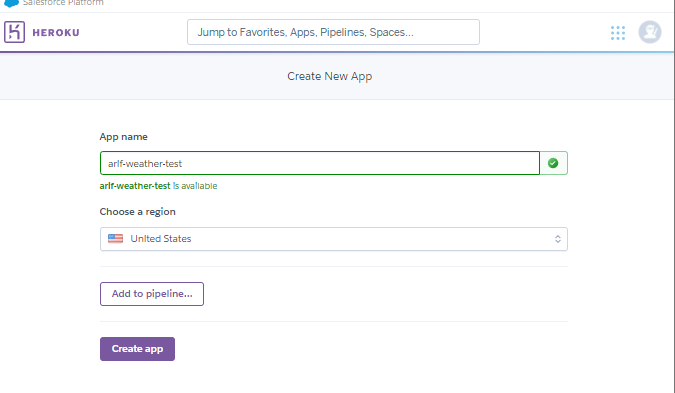
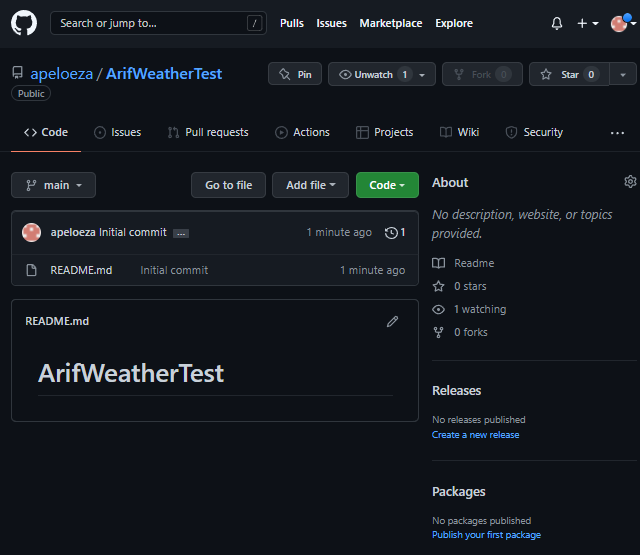
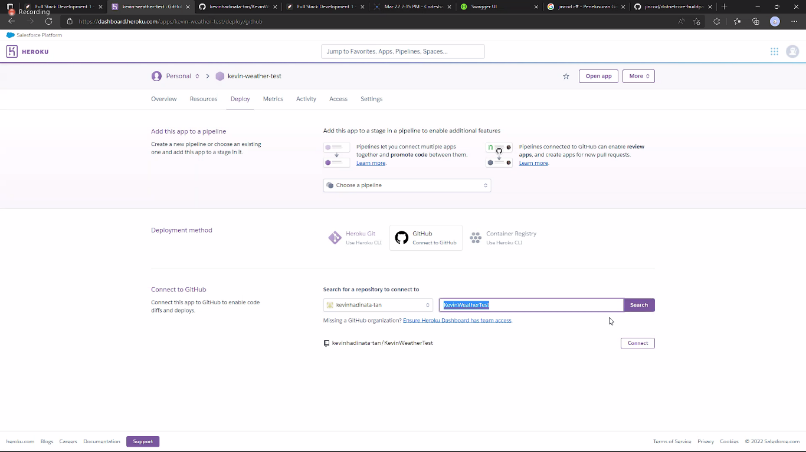
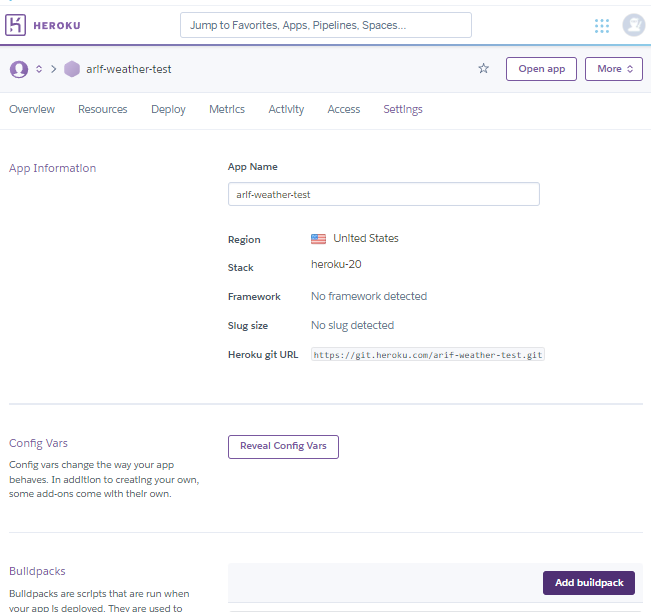
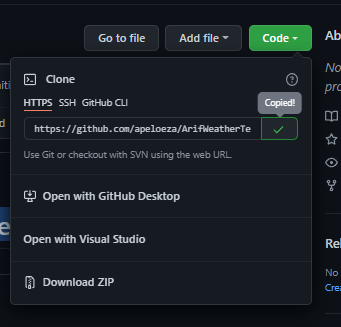
Lalu open startup.cs lalu keluarkan seperti gambar dibawah

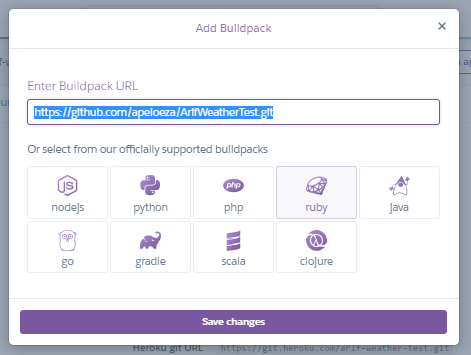
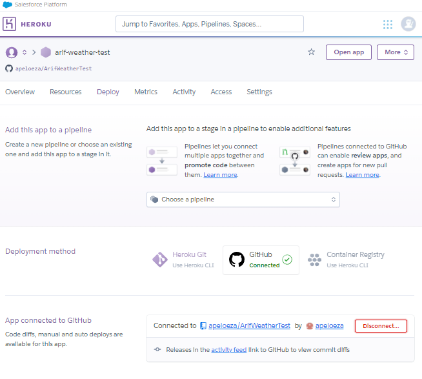
🡺 

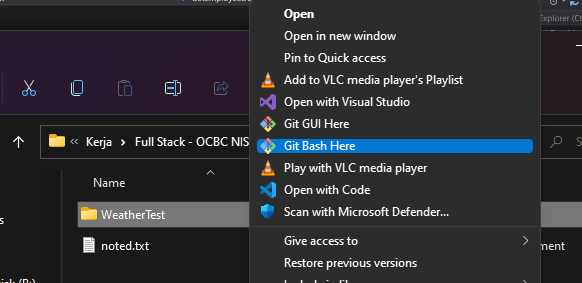
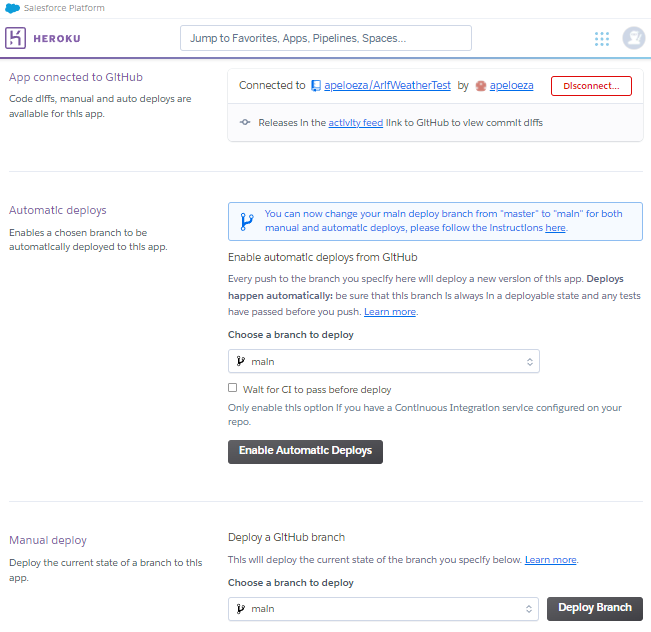
Selanjutnya buat fie DockerFile

 lalu pastekan sourcecode dan sesuaikan



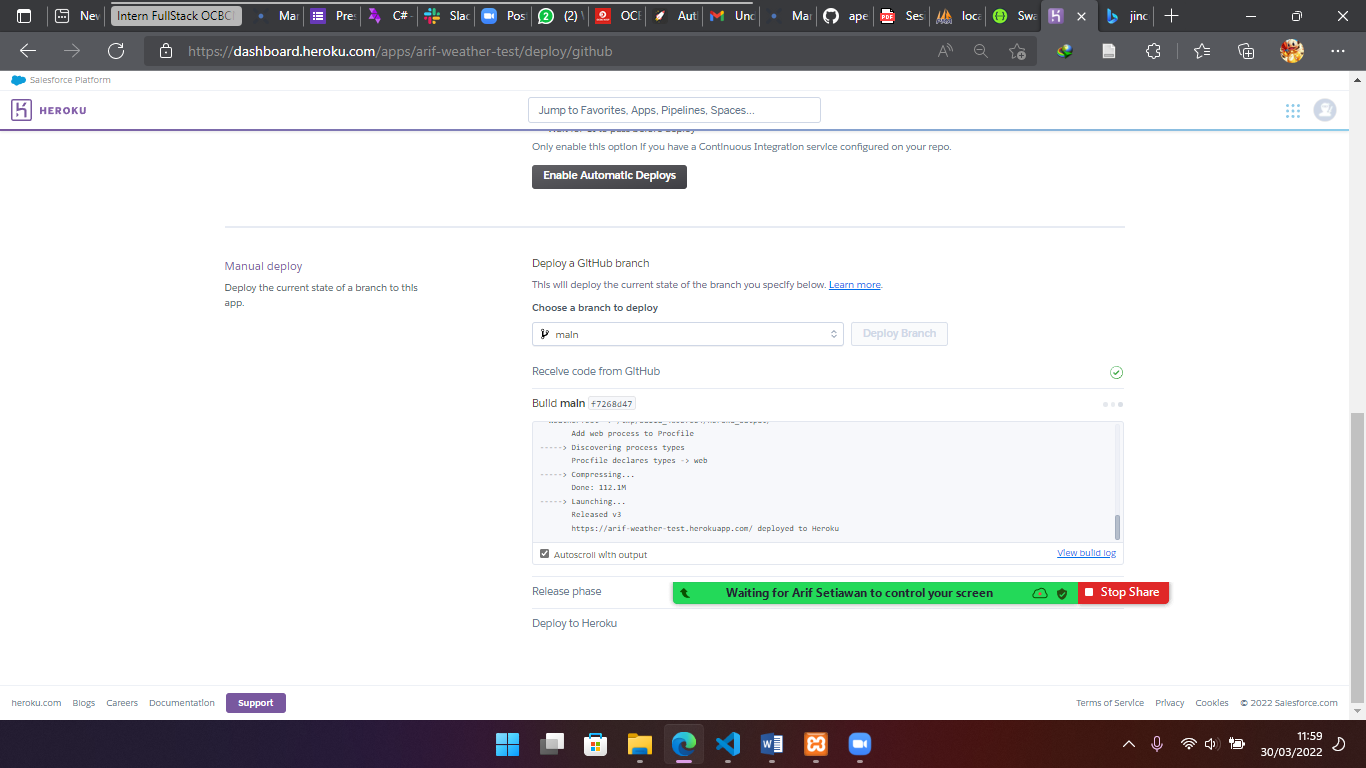
  

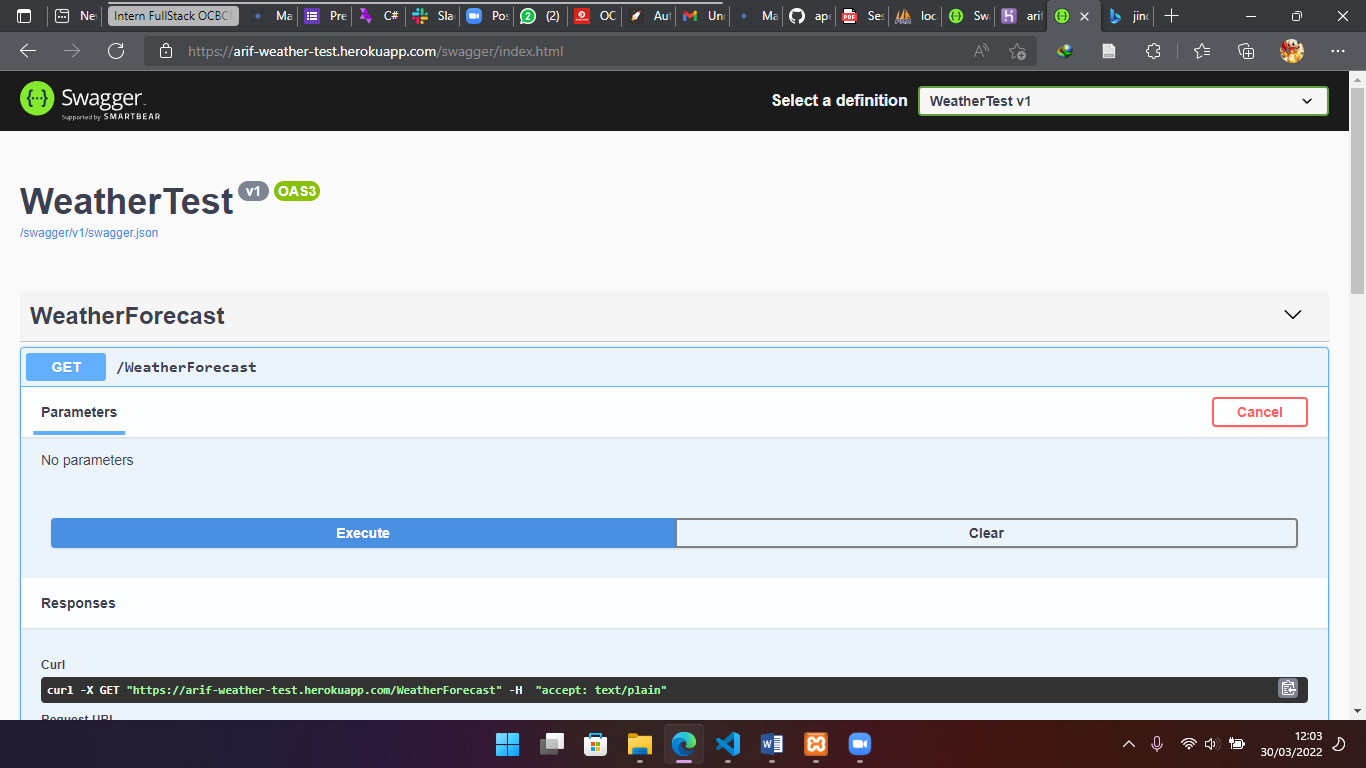




services.AddDbContext<ApiDbContext>(options => options.UseMySQL(Configuration.GetConnectionString("DefaultConnection")));

you should just simply run these two lines of commands inside powerShell: First Exit docker hub desktop, then run these commands in PowerShell:wsl --unregister docker-desktopwsl --unregister docker-desktop-data





git initgit remote add origin https://github.com/Leonuch/flex.gitgit add README.mdgit commit -m "first commit"git checkout -b maingit pull origin branchname --allow-unrelated-historiesgit push -u origin main