Table of Contents

[Basics 3](#_Toc121692733)

[Current Resources with Changes 3](#_Toc121692734)

[New Resources 3](#_Toc121692735)

[Build Angular 3](#_Toc121692736)

[Controllers 3](#_Toc121692737)

[FallbackController.cs 3](#_Toc121692738)

[Program.cs 3](#_Toc121692739)

[Build after Angular 4](#_Toc121692740)

[Docker 4](#_Toc121692741)

[Download / Install 4](#_Toc121692742)

[Important Features 4](#_Toc121692743)

[Account 4](#_Toc121692744)

[PostgreSQL Container 4](#_Toc121692745)

[PostgreSQL 5](#_Toc121692746)

[Viewing Postgres database in VS Vode 5](#_Toc121692747)

[Drop SQLLite DB 5](#_Toc121692748)

[Add PostgresSQL EF Package 6](#_Toc121692749)

[Update Connection String to PostgreSQL 6](#_Toc121692750)

[Update ServiceExtension to use PostgreSQL 7](#_Toc121692751)

[/Core/Extensions/ServiceExtensions.cs 7](#_Toc121692752)

[Update Migrations 7](#_Toc121692753)

[Delete Current Migrations 7](#_Toc121692754)

[Create new clean Migrations 7](#_Toc121692755)

[Run App 7](#_Toc121692756)

[EnableLegacyTimestampBehavior in DBContext 7](#_Toc121692757)

[Check the Database 8](#_Toc121692758)

[Dockerizing 9](#_Toc121692759)

[Dockerfile 9](#_Toc121692760)

[.dockerignore 10](#_Toc121692761)

[Terminal Commands – Creating the Image 10](#_Toc121692762)

[Successful Build 10](#_Toc121692763)

[Run Image 10](#_Toc121692764)

[Error 10](#_Toc121692765)

[Clean DB and Run Image again 11](#_Toc121692766)

[Run App 11](#_Toc121692767)

[Push Image to DockerHub 11](#_Toc121692768)

[Option 1 11](#_Toc121692769)

[Option 2 11](#_Toc121692770)

[Fly.io Hosting Service 12](#_Toc121692771)

# Basics

|  |  |
| --- | --- |
| WorkingFolder | Copy the content of “Site-08-PhotoManagement” in “Site-09-Publishing” and issue   * dotnet build |
| Cloudinary Setting | appsetting.json is not checked in so make sure to bring in this from “Site-01-Basics” |

# Current Resources with Changes

1. Program.cs
2. /Core/Extensions/ServiceExtensions
3. /Core/DB/DataContext

# New Resources

1. /Controllers/FallbackController
2. Docker
3. PostgreSQL

# Build Angular

Go to Angular app [View 0040 Angular .. doc] and follow the build procedure.

# Controllers

## FallbackController.cs

Create a mvc controller

using System.IO;

using Microsoft.AspNetCore.Mvc;

namespace MSC.Api.Controllers;

public class FallbackController : Controller

{

    public ActionResult Index()

    {

        //get the index.html from wwwwroot folder

        var indexHtml = Path.Combine(Directory.GetCurrentDirectory(), "wwwroot", "index.html");

        return PhysicalFile(indexHtml, "text/HTML");

    }

}

# Program.cs

Use middle war to instruct use default files and static files. This will go in between UseAuthorization and MapControllers.

//CUSTOM: start

//serve static files as well. In this case we are building the angular prod app inside the MySocialConnect-API\MSC.Api folder

app.UseDefaultFiles(); //will fish out the index.html file from wwwwroot folder

app.UseStaticFiles(); //will be looking for the wwwroot folder

//CUSTOM: END

Then specify fallback controller inside the following custom section.

//CUSTOM: Start

//ordering is important here. UseCors before UseAuthentication and UseAuthentication before UseAuthorization

app.UseCors(myAllowSpecificOrigins);

app.UseAuthentication();

//tell routing about a hub end point and provide a route for accessing PresenceHub

app.MapHub<PresenceHub>("hubs/presence");

app.MapHub<MessageHub>("hubs/message");

//map to fallback controller after angular build and wwwroot folder getting created inside the MVC.Api folder

app.MapFallbackToController("Index", "Fallback");

//CUSTOM: End

# Build after Angular

Once the angular app has been built then wwwroot folder should show in the MSC.Api folder [View 0040 Angular .. doc]. At this time go to the MSC.Api and issue command dotnet run. The link for the angular should display.

Text

Description automatically generated

Go to <https://localhost:5000/> and you should see your app running.

# Docker

## Download / Install

Download and install docker: <https://www.docker.com/>

Look at the requirements here: <https://docs.docker.com/desktop/install/windows-install/>

### Important Features

Following should be turned on

Graphical user interface, text, application

Description automatically generated

## Account

If you are not registered with docker then do create a free account. This will be need to post the app image to docker hub.

## PostgreSQL Container

* In command prompt go to your root folder for the project and create a postgres container.
* Keep the password simple since it will be used on the local machine
* Also provide ports as 5432 for both internal and external ports
* Run it in detatched mode
* And provide the latest version for the postgres
* docker run --name postgres -e POSTGRES\_PASSWORD=postgrespw -p 5432:5432 -d postgres:latest

b06c2bdb15612f98c9c381eda19cc6f39c9c41d122b167ecf45310206a7e8900

Above is postgres sql server running. Go to docker and you’ll see the container as well.

Graphical user interface, text, application, email

Description automatically generated

# PostgreSQL

## Viewing Postgres database in VS Vode

|  |  |
| --- | --- |
| Go to extensions and install the PostgresSQL  Graphical user interface, application  Description automatically generated | This should add an icon  A screenshot of a cell phone  Description automatically generated with low confidence |

## Drop SQLLite DB

* dotnet ef database drop

Build started...

Build succeeded.

The Entity Framework tools version '6.0.6' is older than that of the runtime '6.0.10'. Update the tools for the latest features and bug fixes. See https://aka.ms/AAc1fbw for more information.

info: Microsoft.EntityFrameworkCore.Infrastructure[10403]

Entity Framework Core 6.0.10 initialized 'DataContext' using provider 'Microsoft.EntityFrameworkCore.Sqlite:6.0.6' with options: None

Are you sure you want to drop the database 'main' on server 'Core/DB/MySocialConnect.db'? (y/N)

y

info: Microsoft.EntityFrameworkCore.Infrastructure[10403]

Entity Framework Core 6.0.10 initialized 'DataContext' using provider 'Microsoft.EntityFrameworkCore.Sqlite:6.0.6' with options: None

Dropping database 'main' on server 'Core/DB/MySocialConnect.db'.

Successfully dropped database 'main'.

## Add PostgresSQL EF Package

* Npgsql.EntityFrameworkCore.PostgreSQL by Shay Rojan v7

Graphical user interface, text, website

Description automatically generated

Update package Microsoft.EntityFrameworkCore.Design to version 7.0

* Microsoft.EntityFrameworkCore.Design

Graphical user interface, website

Description automatically generated

## Update Connection String to PostgreSQL

Go to appsettings.Development.json

|  |  |
| --- | --- |
| From | To |
| {    "Logging": {      "LogLevel": {        "Default": "Information",        "Microsoft.AspNetCore": "Warning"      }    },    "ConnectionStrings": {      "DefaultConnection": "Data source=Core/DB/MySocialConnect.db"    },    "AllowSpecificOrigins": ["https://localhost:4200","http://localhost:4200"],    "TokenKey": "7NCyQkWBsqV3bZsT4qShUN6qzpWUjmRs"  } | {    "Logging": {      "LogLevel": {        "Default": "Information",        "Microsoft.AspNetCore": "Warning"      }    },    "ConnectionStrings": {      "DefaultConnection": "Server=localhost; Port=5432; User Id=postgres; Password=postgrespw; Database=MySocialConnect"    },    "AllowSpecificOrigins": ["https://localhost:4200","http://localhost:4200"],    "TokenKey": "7NCyQkWBsqV3bZsT4qShUN6qzpWUjmRs"  } |

* In the postgresql connection, the user id is by default. We didn’t set this up while setting up [postgreSQL docker container](#_PostgreSQL_Container). Also the port and password are the same as during setup.
* Make sure that the postgreSql container is running by going to Docker and looking at containers.

## Update ServiceExtension to use PostgreSQL

### /Core/Extensions/ServiceExtensions.cs

    public static void RegisterDBContext(this IServiceCollection services, IConfiguration config)

    {

        services.AddDbContext<DataContext>(options =>

        {

            //options.UseSqlite(config.GetDefaultConnectionString());

            options.UseNpgsql(config.GetDefaultConnectionString());

        });

    }

## Update Migrations

### Delete Current Migrations

Migrations have been created with “Sqlite”. For this delete the full folder manually

* /Core/DB/Migrations

### Create new clean Migrations

* dotnet ef migrations add PostgreSQLInitial -o Core/DB/Migrations

Graphical user interface, text, application, chat or text message

Description automatically generated

## Run App

* dotnet run

If UTC datetime error shows then do following

### EnableLegacyTimestampBehavior in DBContext

#### Option1: /Core/DB/DataContext.cs

    public DataContext(DbContextOptions options) : base(options)

    {

        AppContext.SetSwitch("Npgsql.EnableLegacyTimestampBehavior", true);

    }

#### Option2: Convert to UTC

1. ensuring all DateTime.Now as well as DateTime.Today were changed to DateTime.UtcNow
2. updating the seed data to use UTC/ISO dates. For this, append "T00:00:00Z" to each timestamp in the seed data json file w/ the following find-replace regex in vs code:

find: "(\d{4}-\d{2}-\d{2})"

replace: "$1T00:00:00Z"

#### Option3: Convert to UTC when seeding /Core/DB/Seed.cs

Convert the dates read from the json to utc

        //add password to the users, make username lower case and track users

        foreach (var user in users)

        {

            user.UserName = user.UserName.ToLowerInvariant();

            user.CreatedOn = DateTime.SpecifyKind(user.CreatedOn, DateTimeKind.Utc);

            user.UpdatedOn = DateTime.SpecifyKind(user.UpdatedOn, DateTimeKind.Utc);

            user.LastActive = DateTime.SpecifyKind(user.LastActive, DateTimeKind.Utc);

## Check the Database

Go to postgre and create a connection. Keep note that postgre is running in a container

A screenshot of a computer

Description automatically generated with medium confidence

Type

* localhost and hit enter
* postgres for the user name and hit enter
* postgrespw for the password and hit enter
* 5432 for the port and hit enter
* Select “standard connection” and hit enter
* Show all databases, select MySocialConnect

Full connection we have already placed in the appSetting.Development.json so follow that for the prompts

   "DefaultConnection": "Server=localhost; Port=5432; User Id=postgres; Password=postgrespw; Database=MySocialConnect"

You should see the database now

Text

Description automatically generated

# Dockerizing

Add Docker Extension

Graphical user interface, application

Description automatically generated

At this point should see a docker icon

Text

Description automatically generated

## Dockerfile

Right click on the api folder and create a new file with name “Dockerfile”

Since I have .net 6 installed so referencing that below. This important the use the version that is installed on machine.

# build environment

FROM mcr.microsoft.com/dotnet/sdk:6.0 AS build-env

# working dir

WORKDIR /app

# copy csproj and restor as distinct layers => inside working directory

COPY \*.csproj ./

# restore the dependencies listed

RUN dotnet restore

# copy every thing else and build

COPY . ./

# publish

RUN dotnet publish -c Release -o out

# build run time image - sdk is quite large and we do not need that inside our container

FROM mcr.microsoft.com/dotnet/aspnet:6.0

WORKDIR /app

# copy every thing from the out to the root of the container

COPY --from=build-env /app/out .

ENTRYPOINT [ "dotnet", "MSC.Api.dll" ]

## .dockerignore

Right click in the api folder and create a new file with name “.dockerignore”

# ignore following since these will be restored

\*\*/bin

\*\*/obj

## Terminal Commands – Creating the Image

* Stay inside your project folder
* You must be signed into Docker Desktop
* Run the following command. It will take some time to complete the first time.
* “tahirjadoon” is any meaningful name that you want to provide and then the second part the site name.
  + It has to be lower case
* The . at the end is important
* docker build -t tahirjadoon/mysocialconnect .

### Successful Build

You should see your image in vscode and docket desktop

|  |  |
| --- | --- |
| Graphical user interface, application  Description automatically generated |  |

## Run Image

* docker run --rm -it -p 8080:80 learning/mysocialconnect:latest

## Error

Need to update the config since the hosting environment is production.

Text

Description automatically generated

Move the connection string to the app.settings.json. We are running from inside the docker image so will need to update the host part. Later we’ll do something different since the app.settings.json is not getting pushed to the repository.

,

  "ConnectionStrings": {

    "DefaultConnection": "Server=host.docker.internal; Port=5432; User Id=postgres; Password=postgrespw; Database=MySocialConnect"

  }

Build the image and run docker again.

* docker build -t tahirjadoon/mysocialconnect .
* docker run --rm -it -p 8080:80 tahirjadoon/mysocialconnect:latest

## Clean DB and Run Image again

* dotnet ef database drop

Build started...

Build succeeded.

The Entity Framework tools version '6.0.6' is older than that of the runtime '7.0.0'. Update the tools for the latest features and bug fixes. See https://aka.ms/AAc1fbw for more information.

Are you sure you want to drop the database 'MySocialConnect' on server ''? (y/N)

y

Dropping database 'MySocialConnect' on server ''.

info: Microsoft.EntityFrameworkCore.Database.Command[20101]

Executed DbCommand (114ms) [Parameters=[], CommandType='Text', CommandTimeout='30']

REVOKE CONNECT ON DATABASE "MySocialConnect" FROM PUBLIC;

SELECT pg\_terminate\_backend(pg\_stat\_activity.pid) FROM pg\_stat\_activity WHERE datname = 'MySocialConnect';

Successfully dropped database 'MySocialConnect'.

Create Image again

* docker run --rm -it -p 8080:80 tahirjadoon/mysocialconnect:latest

## Run App

Take look at the run image port, it is at 8080 so need to run app via this port.

* <http://localhost:8080/>

# Push Image to DockerHub

## Option 1

Timeline

Description automatically generated

## Option 2

You must be in the api project folder

* docker push tahirjadoon/mysocialconnect:latest

Graphical user interface, text

Description automatically generated

Go to dockerhub / repositories and you should see the pushed image.

# Fly.io Hosting Service

<https://fly.io/docs/languages-and-frameworks/dockerfile/>

Will host the site on fly.io. We have already created the docker image and it got pushed to docker hub as well.

Follow the steps as on following link

<https://fly.io/docs/hands-on/>

## Install Flyctl

**Important: use Windows Terminal App**

Common commands for flyctl. Fly.io commands Run in Windows Terminal App

* iwr https://fly.io/install.ps1 -useb | iex
* flyctl auth signup
* flyctl auth login
* fly launch --image tahirjadoon/mysocialconnect:latest
* fly apps list
* fly destroy yyz-mysocialconnect -y
* fly secrets list
  + this will give the postgresql password

Run the following using PowerShell (admin mode) on windows or follow the above hands on for linux and mac

* iwr https://fly.io/install.ps1 -useb | iex

Text

Description automatically generated

You should see fly.exe and flystl.exe

Graphical user interface, application, Word

Description automatically generated

## SignUp & SignIn

* flyctl auth signup
* flyctl auth login

## Launch App

Go to the folder where your Api solution is. It will be one level up from where the project folder is.

We can type flyctl or just fly as well.

* fly launch --image tahirjadoon/mysocialconnect:latest

Creating app in C:\XXXX\MySocialConnect-API

Using image XXXX/mysocialconnect:latest

? Choose an app name (leave blank to generate one): yyz-mysocialconnect

? Choose an app name (leave blank to generate one): yyz-mysocialconnect

automatically selected personal organization: xxx@gmail.com

? Choose a region for deployment: Singapore, Singapore (sin)

Created app yyz-mysocialconnect in organization personal

Admin URL: https://fly.io/apps/yyz-mysocialconnect

Hostname: yyz-mysocialconnect.fly.dev

Wrote config file fly.toml

? Would you like to set up a Postgresql database now? Yes

? Select configuration: Development - Single node, 1x shared CPU, 256MB RAM, 1GB disk

Creating postgres cluster in organization personal

Creating app...

Setting secrets on app yyz-mysocialconnect-db...

Provisioning 1 of 1 machines with image flyio/postgres:14.4

Waiting for machine to start...

Machine 0e2861d5c17286 is created

==> Monitoring health checks

Waiting for 0e2861d5c17286 to become healthy (started, 3/3)

Postgresql connection information removed from here and put in project folder.

? Would you like to deploy now? No

Your app is ready! Deploy with `flyctl deploy`

This will create the fly.toml file.

## fly.toml

Open fly.toml

### [[service]]

* internal\_port = 8080 is very important

### [env]

Update the section [env], this is for production.

[env]

# important to expose port 8080 in docker file and also no need to put https here.

# important for health checks

ASPNETCORE\_URLS="http://+:8080"

# cloudinay config from the appsettings.json. cloudName and ApiKey are not secret so can put here.

# apiSecrect will not be put here. Keep note that there are two under scores

CloudinarySettings\_\_CloudName="dj7i1ncqg"

CloudinarySettings\_\_ApiKey="588574281751962"

## Setting Secrets via Command Prompt

Not every thing goes in the [env] file. Set the secrets like following

### Setting Cloudinary Secret

* fly secrets set CloudinarySettings\_\_ApiSecret=mtFXXXXXXXXXXXXXXXXXXXXXX80

Secrets are staged for the first deployment

### Setting TokenKey Secret

* fly secrets set TokenKey=7NXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXRs

Secrets are staged for the first deployment

### Getting secrets list

* fly secrets list

**NAME DIGEST CREATED AT**

CloudinarySettings\_\_ApiSecret bXXXXXXXXXXc 4m6s ago

DATABASE\_URL eXXXXXXXXX9 41m12s ago

TokenKey dXXXXXXXXXXa 1m22s ago

## Expose the Port on Docker File

Docker file is in the project folder so add EXPOSE 8080

# build environment

FROM mcr.microsoft.com/dotnet/sdk:6.0 AS build-env

# working dir

WORKDIR /app

EXPOSE 8080