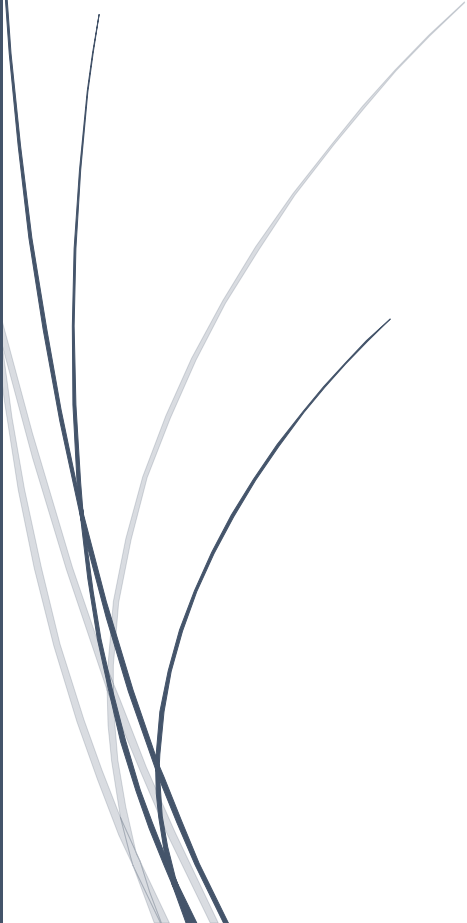


A dark blue vertical bar runs down the left side of the page. A blue arrow points to the right from this bar, containing the date.

10/9/2018

EAD2 REPEAT

Gary Clynch

Several thin, curved lines in dark blue and light grey originate from the bottom left corner and sweep upwards and to the right.

Kadrieh Mohamadzadeh – X00114185
ITT DUBLIN

Contents

Weather API	2
Figure 1	2
Figure 2	3
Figure 3	3
Figure 4 data retrieved from the database	3
Figure 5 database created in azure with the weather table	4
Figure 6 query the weather table retrieving all the data back from the database	4
Figure 7 query the database using filters	4
Azure deployment settings	5
Azure SQL	5
Weather App	5
Figure 8	5
Figure 9	6
Figure 10	6
Figure 11	7
Figure 12	7
Tests	8
Figure 13	8

Weather API

API link hosted on azure. Must use swagger to try out the get method. Type in a city name and it should return the data stored in azure SQL database.

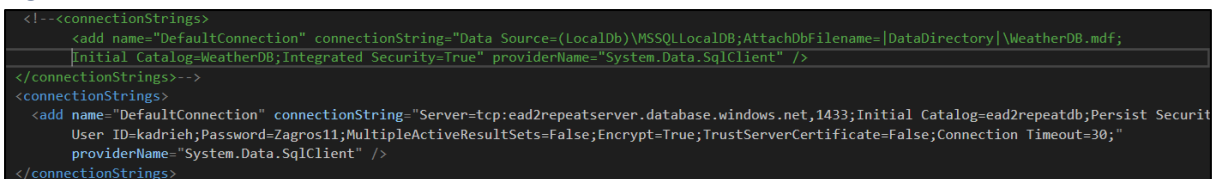
City names: Galway, Sligo, Wicklow, Dublin, Kildare, Cork, Limerick, Donegal, Meath, Waterford

<https://ead2repeat.azurewebsites.net/>

<https://ead2repeat.azurewebsites.net/swagger/ui/index>

The web API was created using a code first approach. The weather class was created first and its attributes declared. I have created the database context class in my models' folder and it must inherit the DbContext class. Then I have declared the name of the connection string that my database is going to use and located in the web.config file. In the web.config file I have two connection strings, one was to test my database locally and the other is my website in azure on the server. I also have declared my table that I want to be created in the database class.

Figure 1



```
<!--connectionStrings-->
  <add name="DefaultConnection" connectionString="Data Source=(LocalDb)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\WeatherDB.mdf;Initial Catalog=WeatherDB;Integrated Security=True" providerName="System.Data.SqlClient" />
</connectionStrings-->
<connectionStrings>
  <add name="DefaultConnection" connectionString="Server=tcp:ead2repeatserver.database.windows.net,1433;Initial Catalog=ead2repeatdb;Persist Security ID=kadrieh;Password=Zagros11;MultipleActiveResultSets=False;Encrypt=True;TrustServerCertificate=False;Connection Timeout=30;" providerName="System.Data.SqlClient" />
</connectionStrings>
```

Once this was done I have then enabled migration to give me the configuration class where then I have created my data inside it. Within this configuration class I have declared a list of weather objects and stored the information about 10 cities.

Because the required option was to have a get operation only in the API, then I have created the data in the configuration class and stored this information in the database so that whenever the client searches for a city this can be achieved by the get method and returned to the client.

I have then added a migration against the configuration class which then creates another class which I have named it Initial create and this has two methods inside it, the up and down method. It's from these methods that the database essentially gets created. I have then used the update database command and its only then that my database gets created and seeded. To check if our data did get created we will be checking the app_data folder for our tables in the database.

Figure 2

```

public override void Up()
{
    CreateTable(
        "dbo.Weathers",
        c => new
        {
            ID = c.Int(nullable: false, identity: true),
            CityName = c.String(nullable: false),
            WindDirection = c.Int(nullable: false),
            WindCondition = c.Int(nullable: false),
            MaxTemp = c.Int(nullable: false),
            MinTemp = c.Int(nullable: false),
            WindSpeed = c.Int(nullable: false),
            NextDayWeather = c.String(),
            WeatherDate = c.DateTime(nullable: false),
        })
    .PrimaryKey(t => t.ID);
}

```

Deployed the app to azure using visual studio to publish to the server.

Figure 3

Figure 4 data retrieved from the database

Server response	
Code	Details
200	<p>Response body</p> <pre> { "ID": 2, "CityName": "Galway", "WindDirection": "West", "WeatherCondition": "Drizzle", "MaxTemp": 15, "MinTemp": 13, "WindSpeed": 75, "NextDayWeather": "Rainy", "WeatherDate": "2018-10-02T23:17:07.98" } </pre>

Figure 5 database created in azure with the weather table

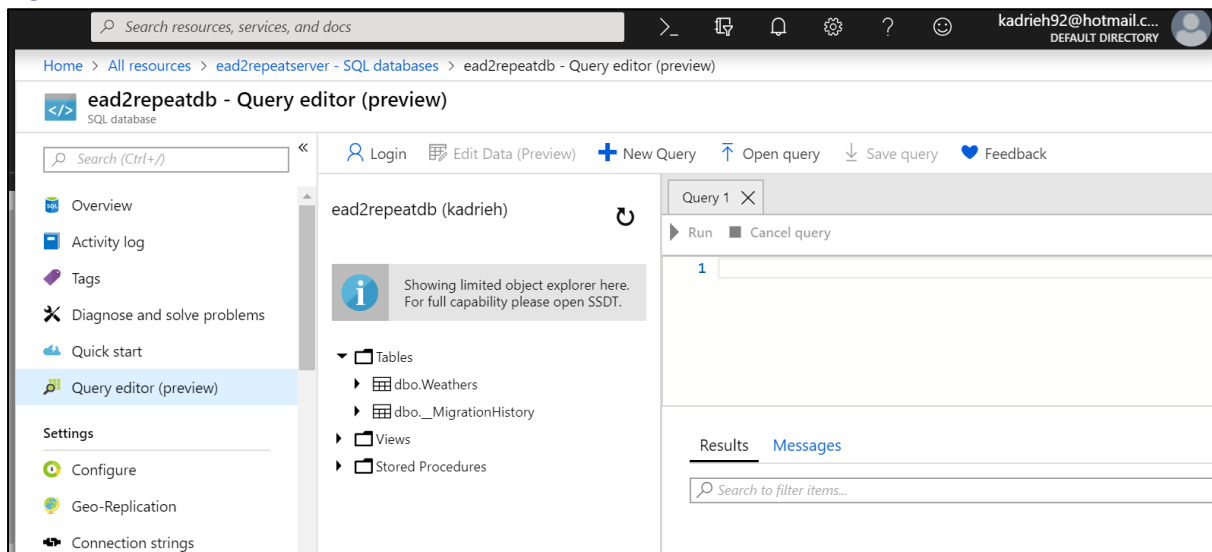


Figure 6 query the weather table retrieving all the data back from the database

Run Cancel query				
1 select * from weathers				
Results Messages				
ID	CITYNAME	WINDDIRECTION	MAXTEMP	MINTEMP
1	Dublin	North	17	13
2	Galway	West	15	13

Figure 7 query the database using filters

Run Cancel query				
1 select * from weathers				
2 where cityname = 'kildare' OR cityname = 'wicklow'				
Results Messages				
ID	CITYNAME	WINDDIRECTION	MAXTEMP	MINTEMP
6	Kildare	West	15	13
9	Wicklow	North	17	13

Azure deployment settings

I created a new web app in azure portal. I open the azure app service where I can view my web app. I click on the web app and download the publish profile. I then right click on my project in visual studio and click the publish option. In the pick a publish target dialog I click on the import profile. I imported the file I downloaded earlier from the web app in azure. I then modify the publish setting to suit my azure account and then click publish.

Azure SQL

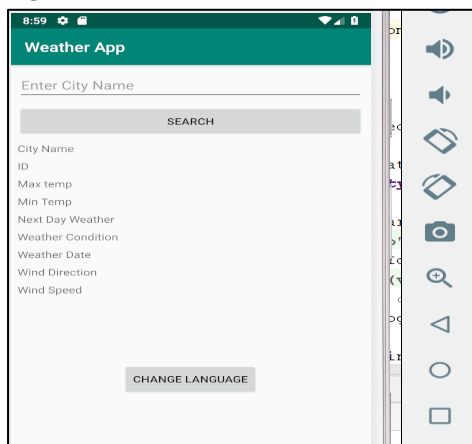
From the azure portal, I clicked into create resources and selected the SQL Database and selected create. I then configured the SQL settings to suit my web app. Username and password credentials are then applied. From here is where our connection string gets created and then we later make use of it in the config.file to be able to deploy the data locally to the azure SQL Database on the server

Weather App

I have created an android app to make use of the web API created in azure using asp.net MVC framework. The aim of this app is to retrieve the data created in the database and display on a screen to the client.

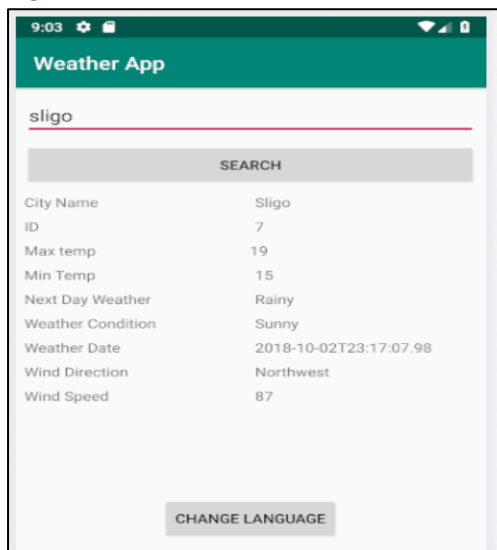
The figure bellow shows the app when run on the emulator.

Figure 8



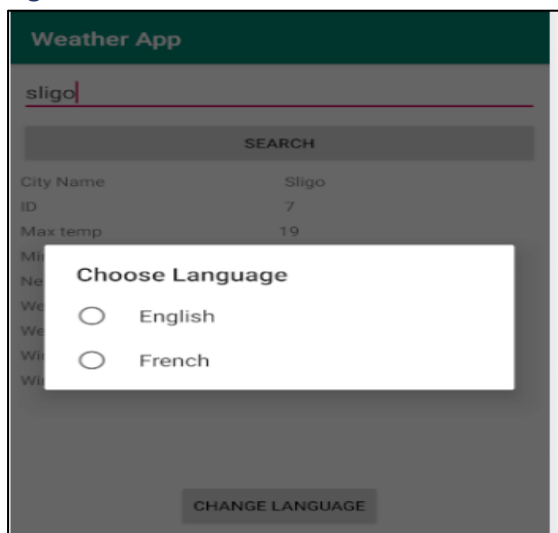
When the client searches for a name of a city it will display the city name and the weather information for that city as shown in the figure bellow.

Figure 9



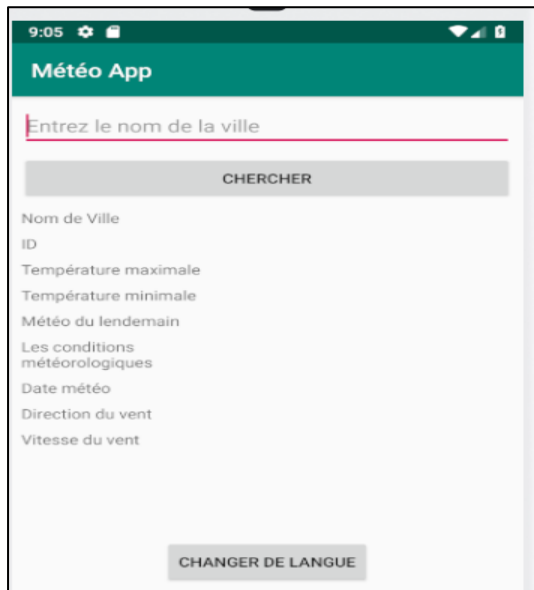
The change language button allows the client to change from one language to another, e.g. English to French as shown in the figure.

Figure 10



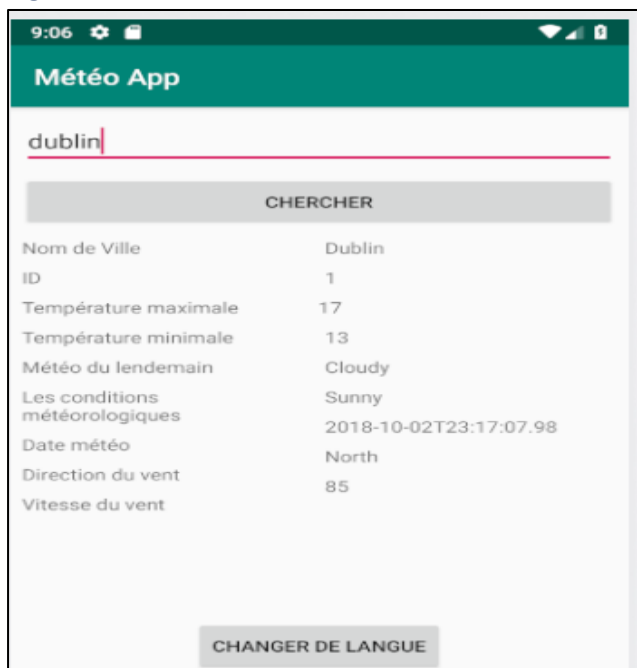
When the client selects the French language the app changes its name and the rest of the information.

Figure 11



The client can search a city and the information is displayed on the screen.

Figure 12



I have created 4 test cases based on the values controller and all test cases have passed. I could not test the weathers controller class as there was only a get method.

Tests

Figure 13

