Mobile Computing Report

WeatherGetWeb

Student Ionescu Iulian Ştefan

1. About the project

My project is about obtaining real time weather forecast using two kind of sources (Web and Arduino microcontroller) and two API services.
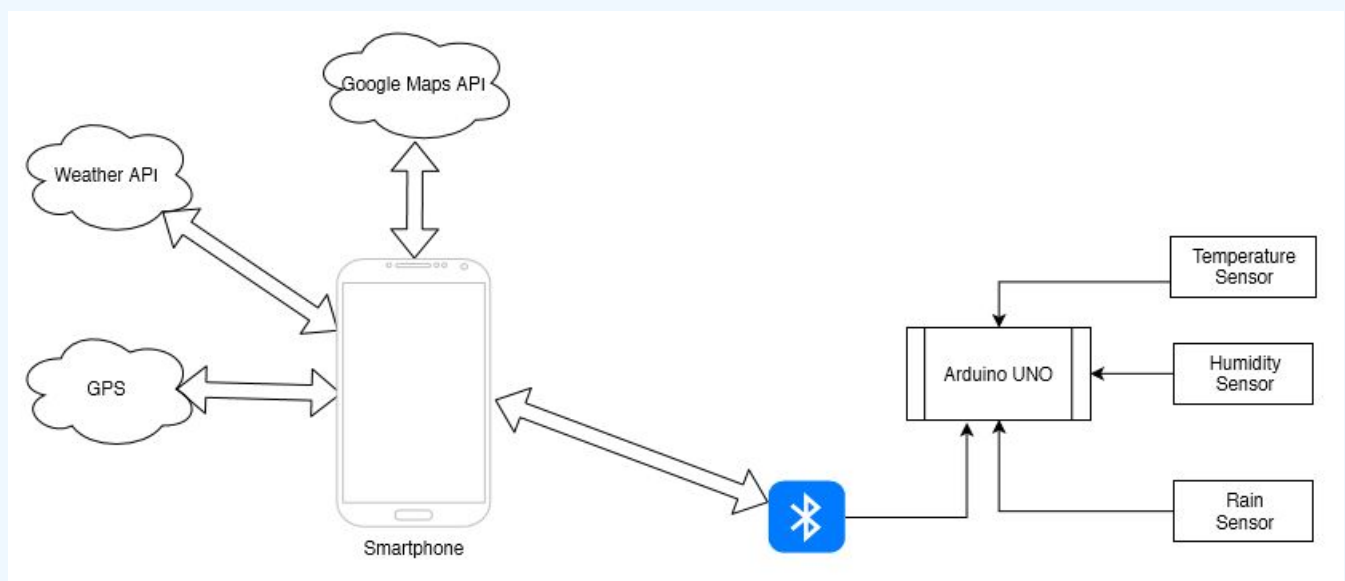
Github link: https://github.com/IulianIonescu/MobileComputing.git

The location data (latitude,longitude) can be retrieved by GPS or by a Google Maps API :
https://maps.googleapis.com/maps/api/geocode/json?address=Craiova,+Romania&key=AIzaSyBnxx_P3fB-7V1ViyvtiCqoAMeZXiZjyRE
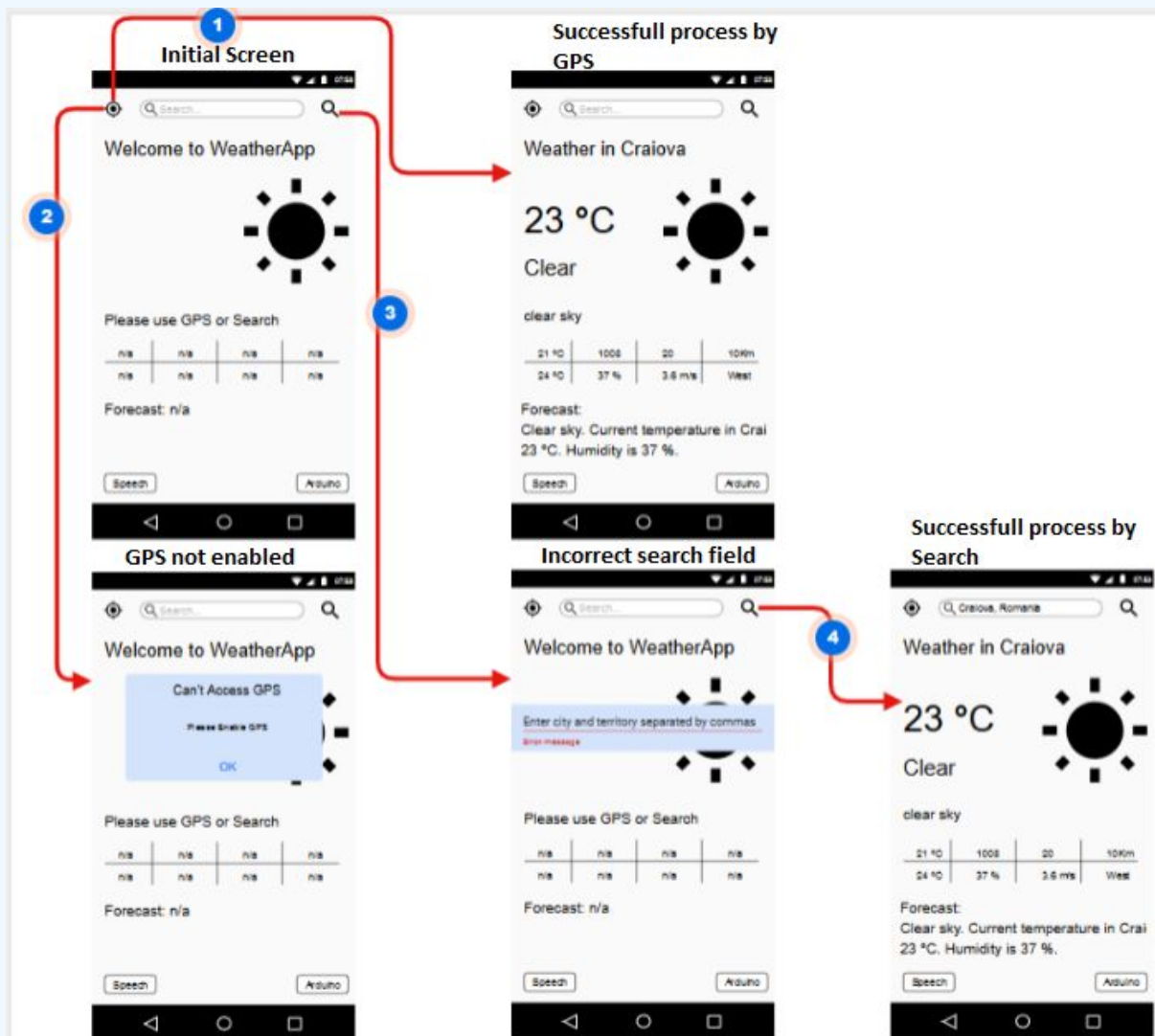
The weather data is retrieved by the following API:
http://api.openweathermap.org/data/2.5/weather?lat=44.32308&lon=23.8103&appid=d5bfbd6d08ee736dad9dd034b772665b.
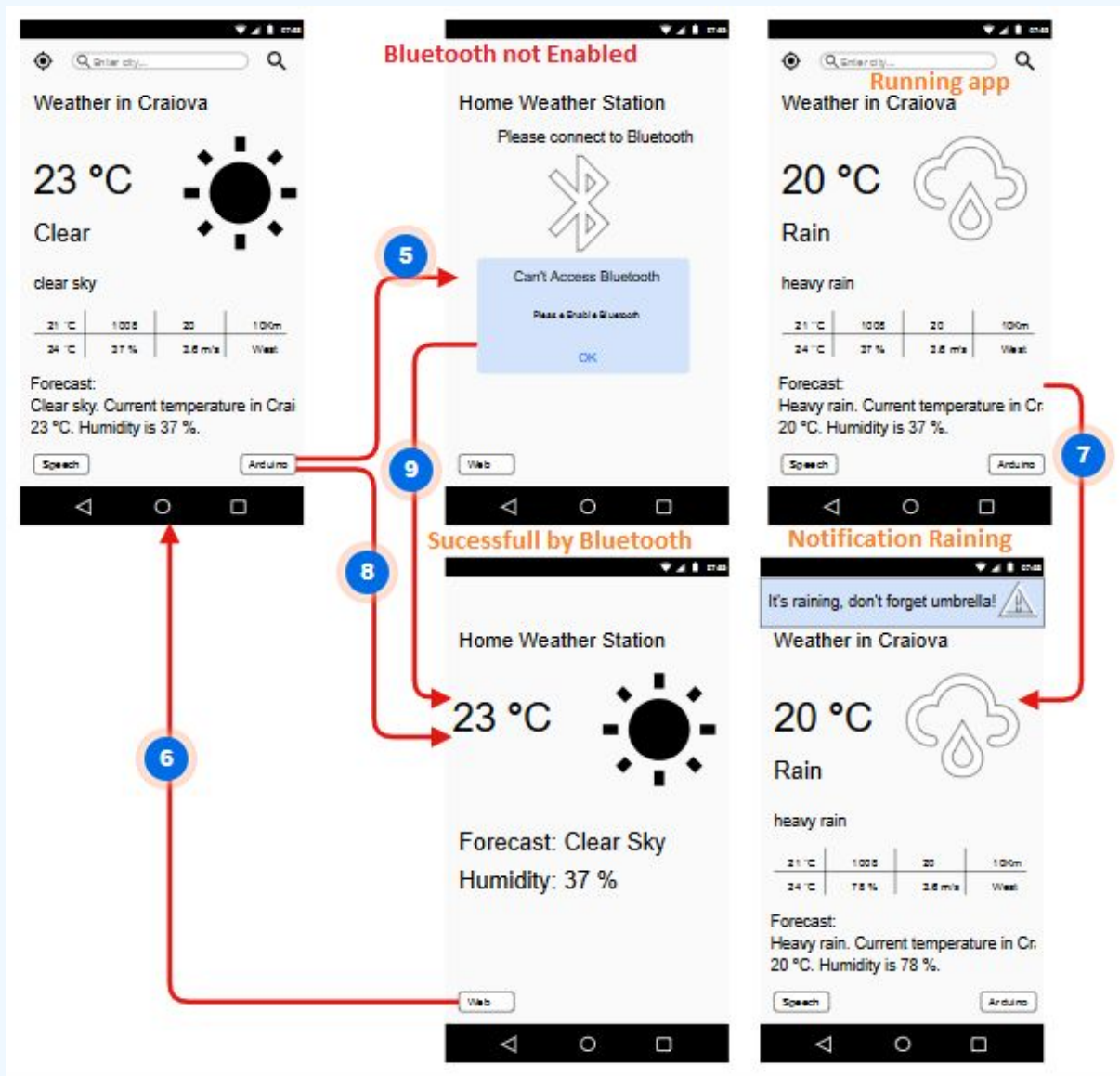
A small amount of weather data (temperature, humidity, rain status) will be retrieved using the microcontroller Arduino UNO along with the respective sensors. The connection between smartphone and Arduino is made by Bluetooth. The purpose of embedded system usage is for more trustable weather info, when user is at home.

Devices block diagram:

2. Application wireframe

Bluetooth not Enabled

Home Weather Station

Please connect to Bluetooth

Can't Access Bluetooth

Please Enable Bluetooth

OK

Running app

Weather in Craiova

20 °C

Rain

heavy rain

21 °C	1005	20	100m
24 °C	37 %	3.6 m/s	West

Forecast:
Heavy rain. Current temperature in Cr. 20 °C. Humidity is 37 %.

Speech   Arduino

Weather in Craiova

23 °C

Clear

clear sky

21 °C	1005	20	100m
24 °C	37 %	3.6 m/s	West

Forecast:
Clear sky. Current temperature in Crai 23 °C. Humidity is 37 %.

Speech   Arduino

Sucessfull by Bluetooth

Home Weather Station

23 °C

Forecast: Clear Sky
Humidity: 37 %

Web

Notification Raining

It's raining, don't forget umbrella!

Weather in Craiova

20 °C

Rain

heavy rain

21 °C	1005	20	100m
24 °C	78 %	3.6 m/s	West

Forecast:
Heavy rain. Current temperature in Cr. 20 °C. Humidity is 78 %.

Speech   Arduino

Arrow 1 : When GPS button it's pressed application launches the built-in GPS in order to receive the latitude and longitude. The Weather API call will include this location information in order to retrieve the weather data and display on application.
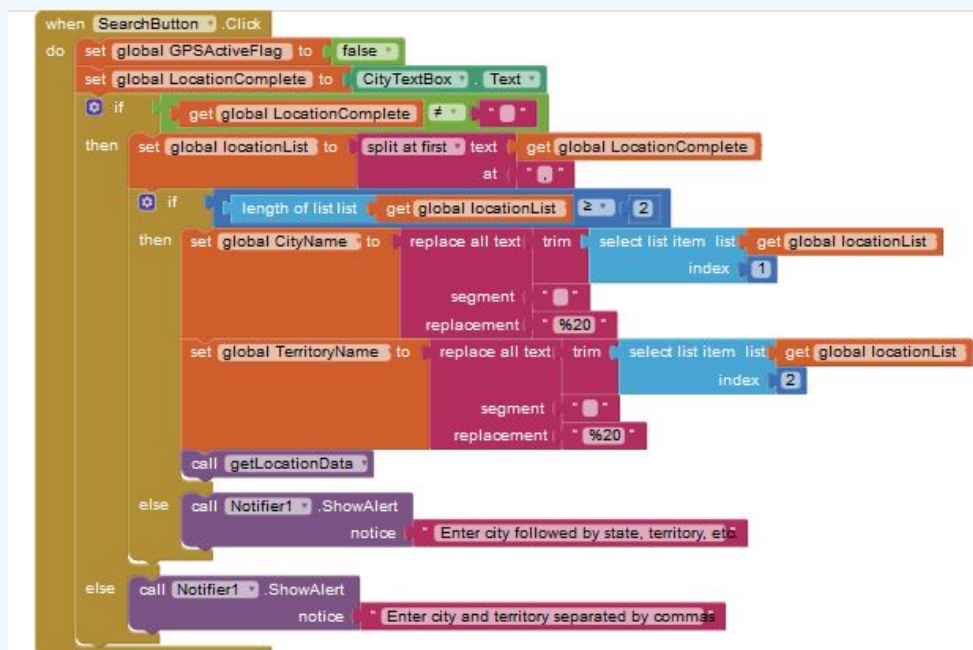
Arrow 2 : When GPS button it's pressed but GPS is not enabled, an error window pops up. By clicking 'OK', user will be driven to the location settings in order to enable GPS.
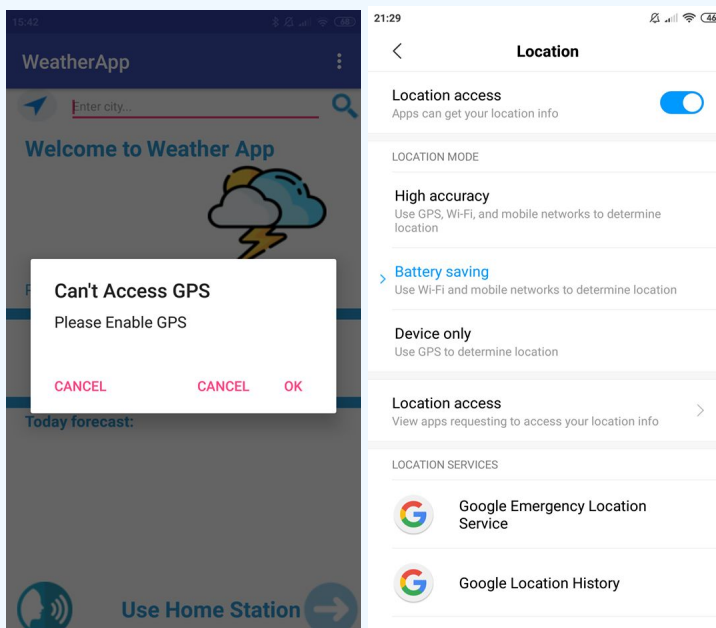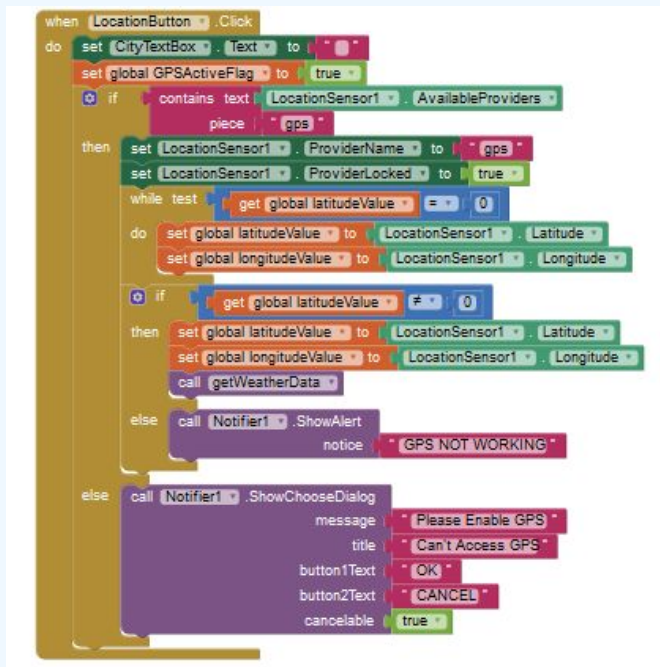
Arrow 3: When Search button is pressed, but textbox fields is not filled or has incorrect format, a helper message pops up.
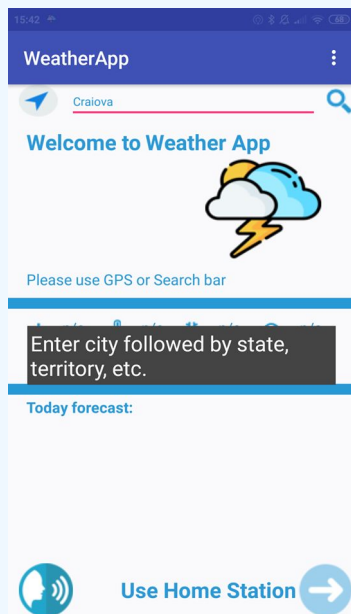

Arrow 4: When Search button it's pressed and preconditions are fulfilled, a call is made to Google Maps API in order to obtain the location data for the searched city. After location is retrieved the Weather API is called and weather data is displayed.
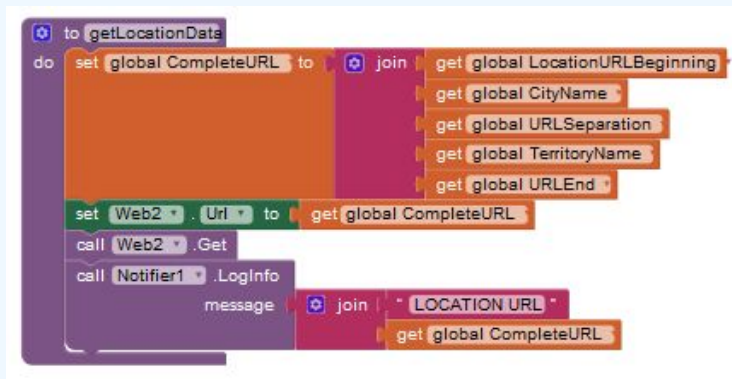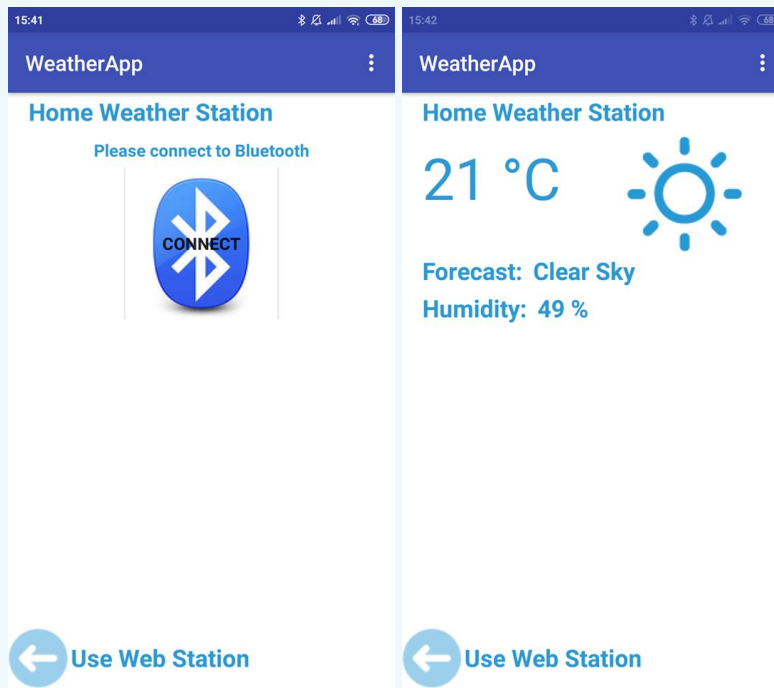
Arrow 5: When Home Weather Station button it's pressed, Screen 2 it's initialized. If Bluetooth is not enabled, an error window pops up. By clicking 'OK', user will be driven to the bluetooth settings in order to enable Bluetooth.
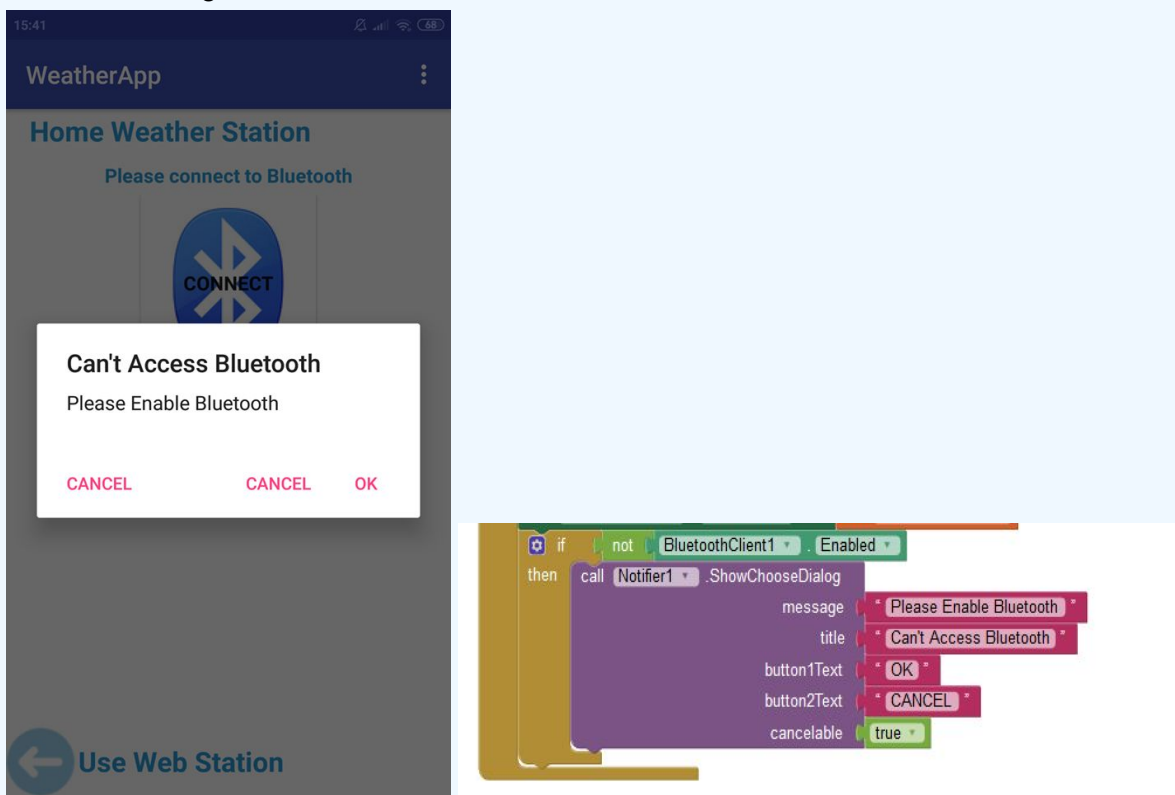
Arrow 8: When Home Weather Station button it's pressed and Bluetooth is enabled and connected to Bluetooth device, application reads the sent bytes, separates the elements by a defined separator character, and displays the info.
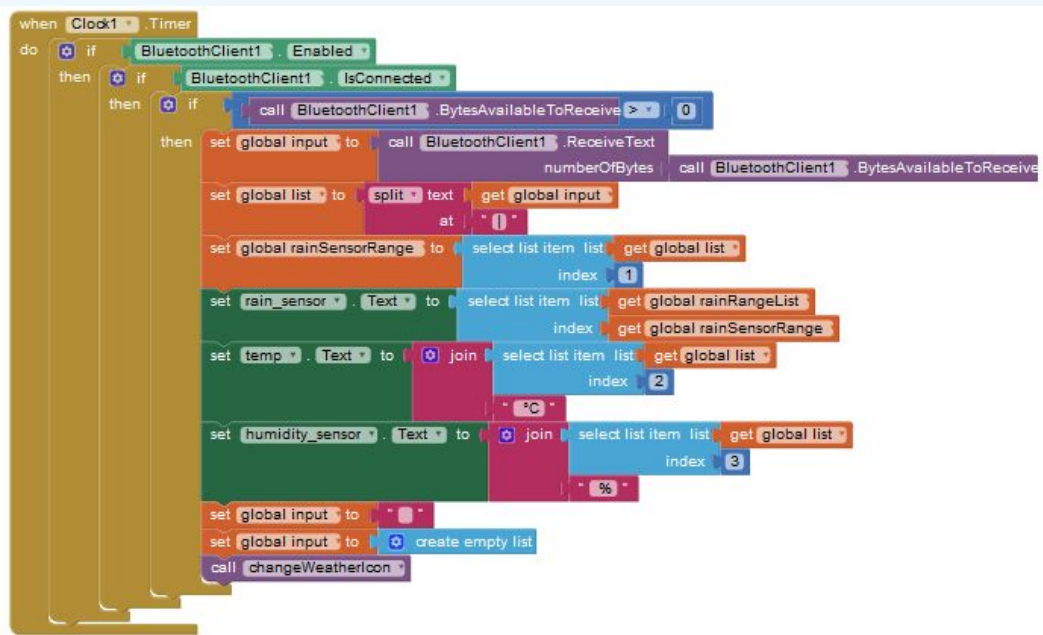
Arrow 6: When Web Station button it's pressed, the screen is changed to the main screen.

Arrow 7: When the forecast (either by Web or Arduino) announces rain, a popup notification will be shown, used to alert the user.
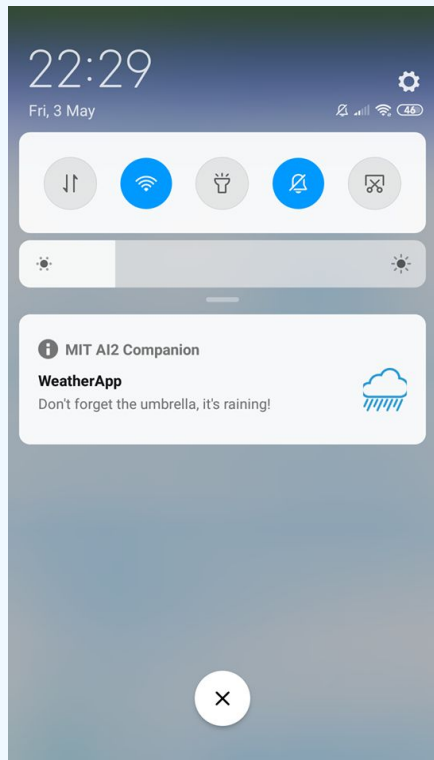
Another processes:
- the weather image is changed according to weather forecast and day time.

- A forecast report speech can be trigger by pressing the button

```
to changeWeatherIcon
do  set global HourTime to   call Clock1 .Now
    if    contains text   get global feed
                  piece      " "Snow" "
    then  set CurrentWeatherIconImage . Picture to    " Snow.png "
    else if   contains text   get global feed
                  piece      " "Rain" "
    then  set CurrentWeatherIconImage . Picture to    " Rain.png "
          call TaifunNotification1 .Send
                      seconds   0
                        title   " WeatherApp "
                         text   " Don't forget the umbrella, it's raining! "
                    startText   " It's raining! "
          set TaifunNotification1 . Icon to   " Rain.png "
    else if   contains text   get global feed
                  piece      " "Thunderstorm" "
    then  set CurrentWeatherIconImage . Picture to    " Rain.png "
          call TaifunNotification1 .Send
                      seconds   0
                        title   " WeatherApp "
                         text   " It's raining! "
                    startText   " Don't forget the umbrella, it's raining! "

    else if   contains text   get global feed
                  piece      " "Clear" "
    then  if    call Clock1 .Hour                          > 19
                      instant   get global HourTime
          then  set CurrentWeatherIconImage . Picture to    " Night.png "
          else  set CurrentWeatherIconImage . Picture to    " Sunny.png "
    else if   contains text   get global feed
                  piece      " "Mist" "
    then  set CurrentWeatherIconImage . Picture to    " Foggy.png "
    else  if    call Clock1 .Hour                          > 19
                      instant   get global HourTime
          then  set CurrentWeatherIconImage . Picture to    " CloudyNight.png "
          else  set CurrentWeatherIconImage . Picture to    " CloudyDay.png "
```
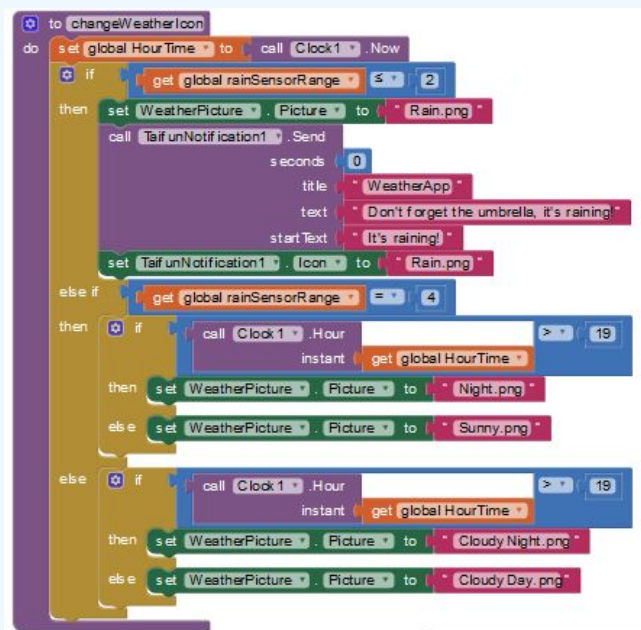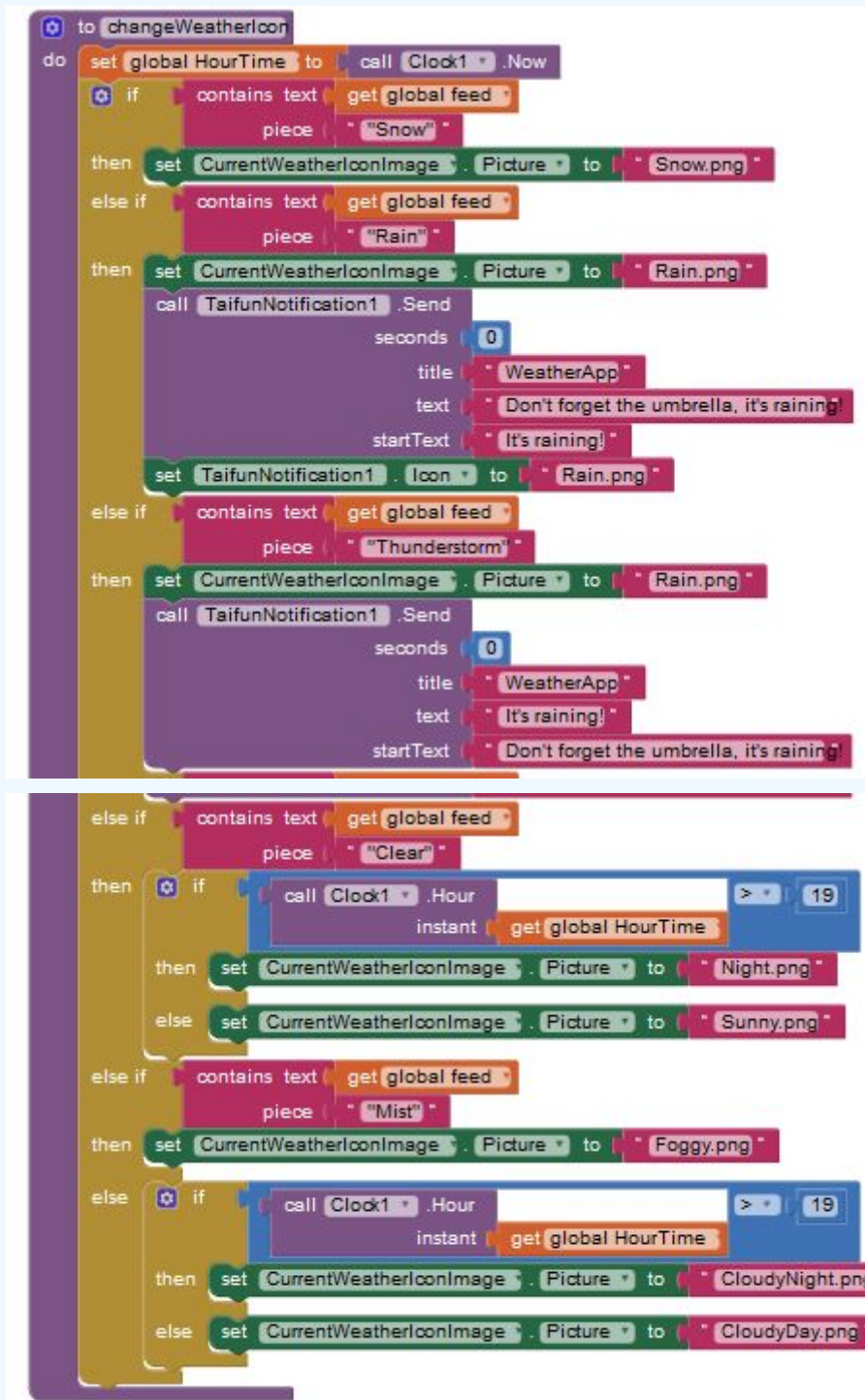
9

# 3. Server Side Functionality

As described above, WeatherGetWeb application data is retrieved by using two API services. First API it's used to retrieve location data, in the case when user do no use GPS. The second API it's used to gather weather data.

Let's describe the handling for the Google Maps API.

The API call will include the city and country typed in Search Box. After the data it's received, it will be broken in JSON format in order to be select the relevant info (latitude,longitude).
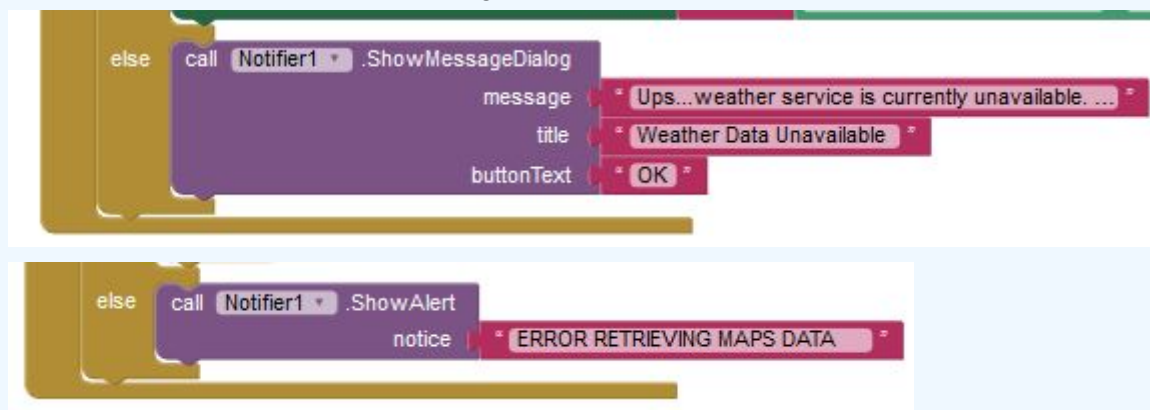

With latitude and longitude obtained, the second API will be called, with the same handling (broke lists until I reach the desired value).
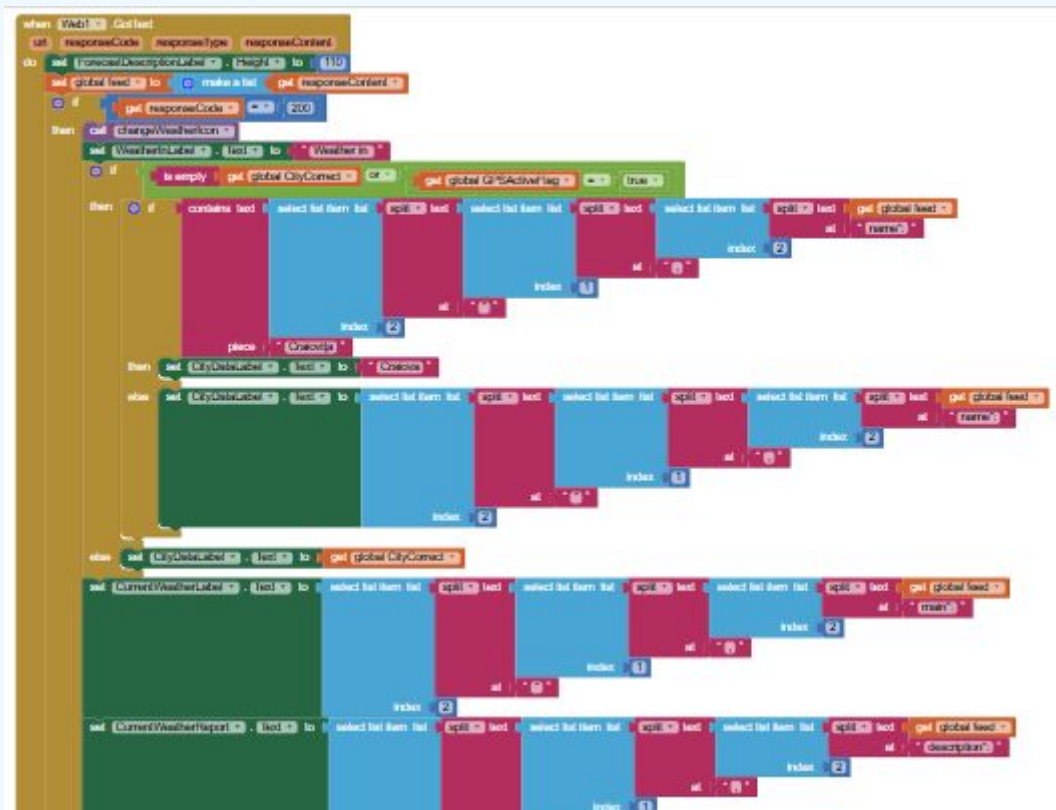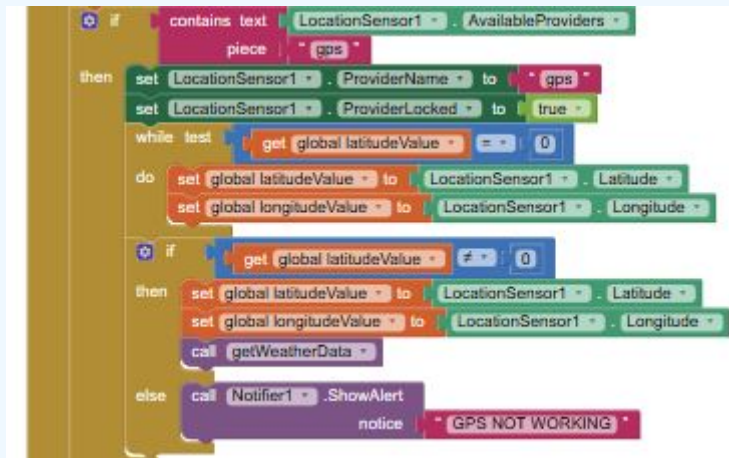
When weather data is retrieved, multiple operations are made in order to display the info within application UI.

If API calls have issues, error messages are called in order to know the root cause.

If GPS location data is not relevant (eg lat=0, long=0) API calls are made until data is relevant.

Fun fact: when API call was made with (0,0) the retrieved location is Earth.

4. Arduino side

Using Arduino UNO and 3 sensors (temperature, humidity, rain detection), application can retrieve real time data. Communication is made my Bluetooth.

HW Components:
- Arduino UNO
- Sensor DHT22 (temperature,humidity)
- Sensor for Raindrops detection
- Bluetooth HC-05

After sensors are read, the values are sent in a specific format which includes a separator, which will be used by application to broke again the bytes in pieces.

```cpp
void ReadSensor_Temperature()
{
  humidity = dht.readHumidity();
  temperature = dht.readTemperature();
}

void ReadSensor_Rain()
{
  int rainSensorReading = analogRead(RAIN_SENSOR);

  //Map sensor rain for easier values handling
  rainRange = map(rainSensorReading,sensorRainMin, sensorRainMax,0,4) + 1;

}
```

```
void loop() {

//Read temperature and humidity
ReadSensor_Temperature();

//Read rain sensor
ReadSensor_Rain();


Serial.print(rainRange);
Serial.print("|");
Serial.print((int)temperature);
Serial.print("|");
Serial.print((int)humidity);
Serial.print("|");


delay(2000);
}
```
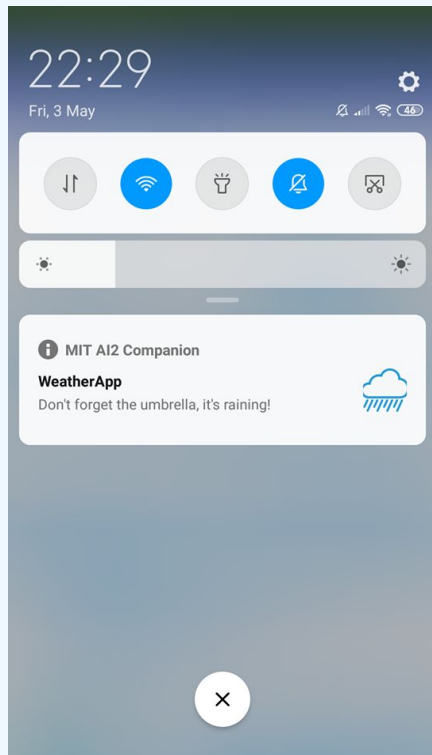
5. Notifications

The only notification used in application it's triggered when weather forecast or sensor announces rain.

Note: App Inventor 2 has some restrictions regarding notifications. The app must be running in order to trigger the notification. For devices with MIUI firmware (sadly mine) the notification it's triggered when app is running in background.

6. Maps and Directions

The only service related to maps is the call of Google Maps API, described above on Server Side chapter.