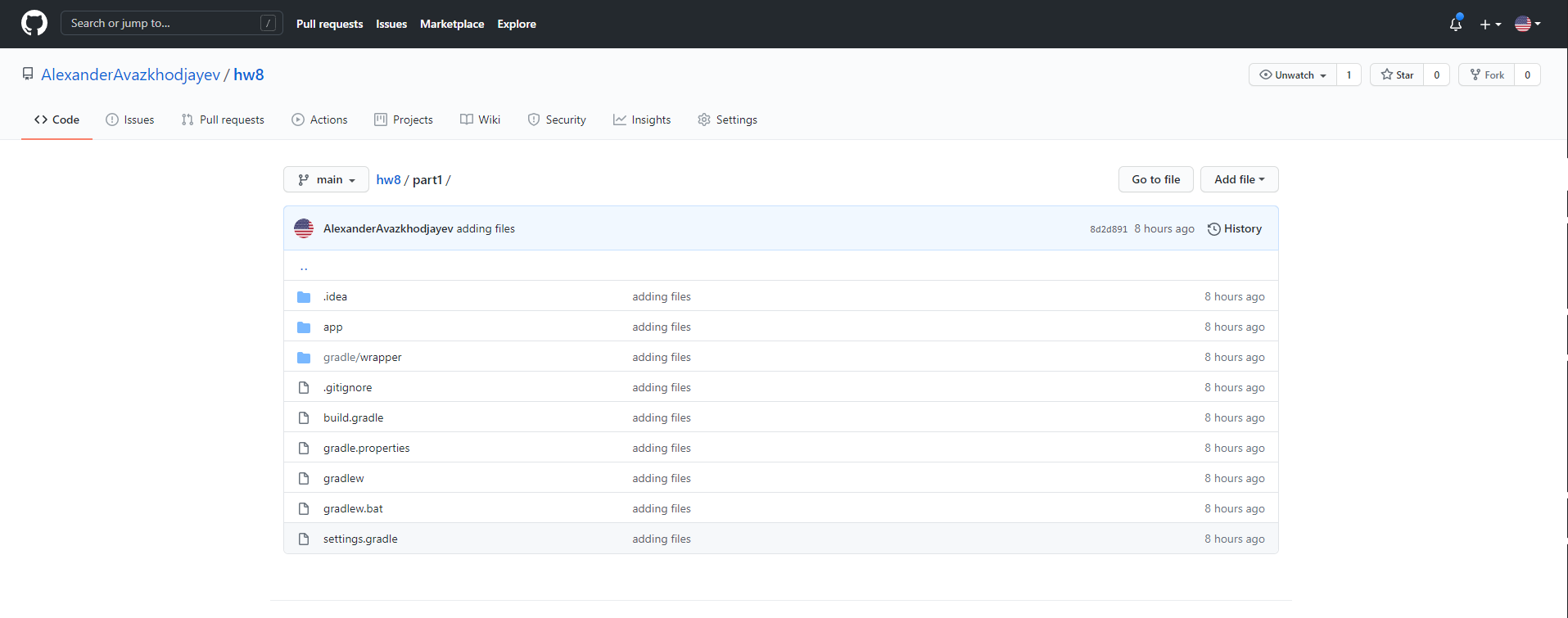
Akbarhon Avazkhodjayev

Mobile Apps

Homework 8

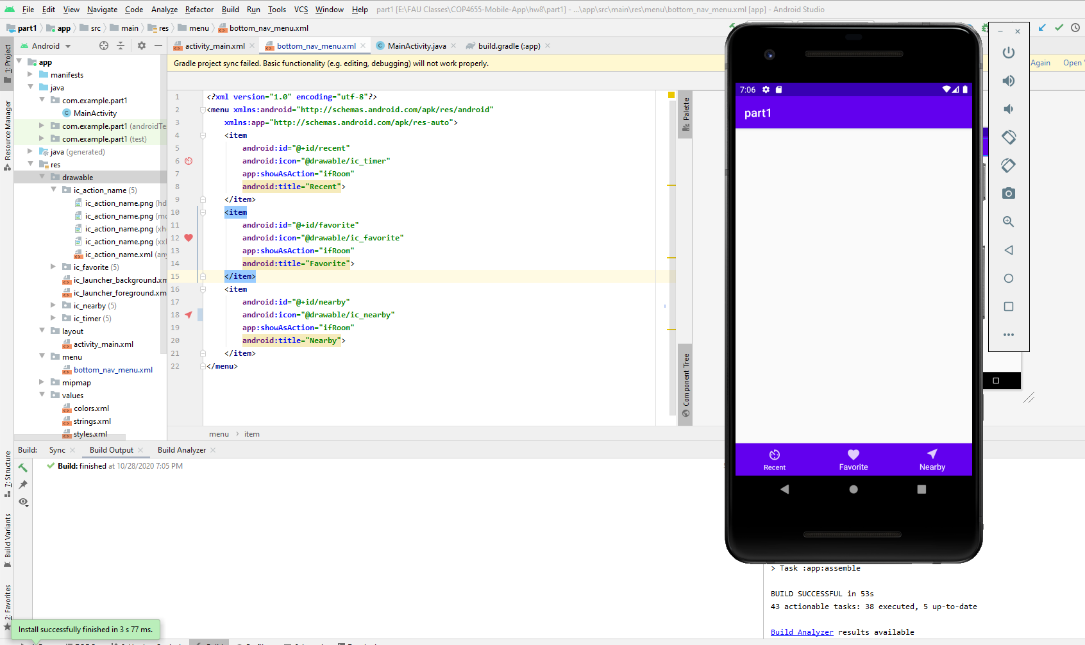
10/30/2020

GitHub with Android Studio Guide (10 points)

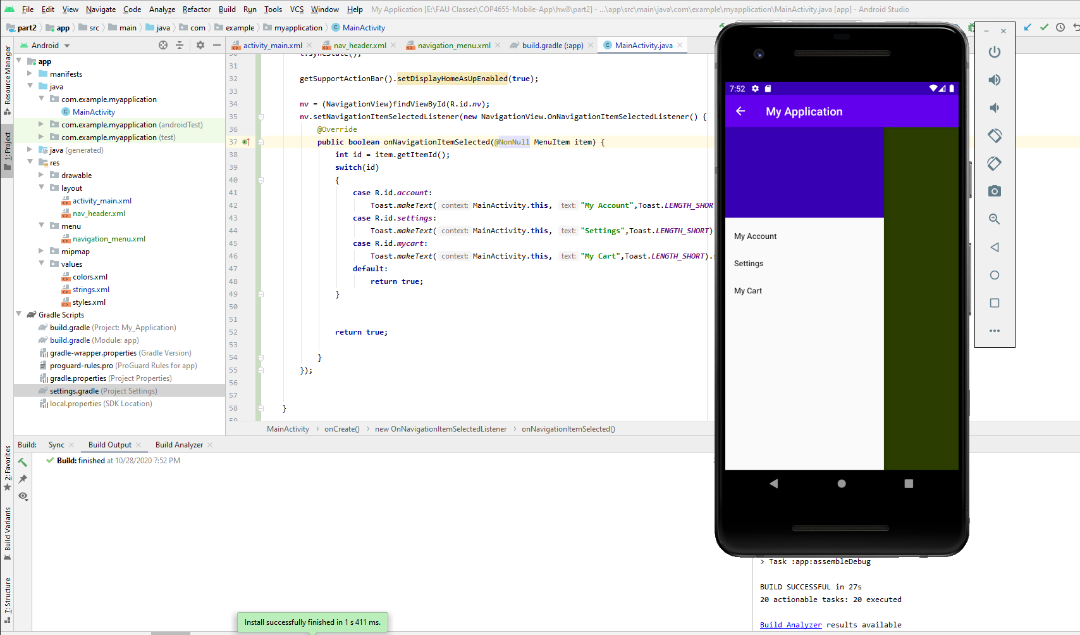


Navigation Guides (20 points / 10 points each)

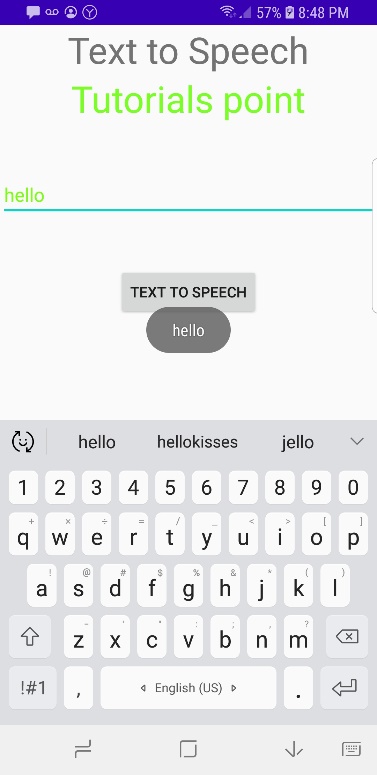
Part One



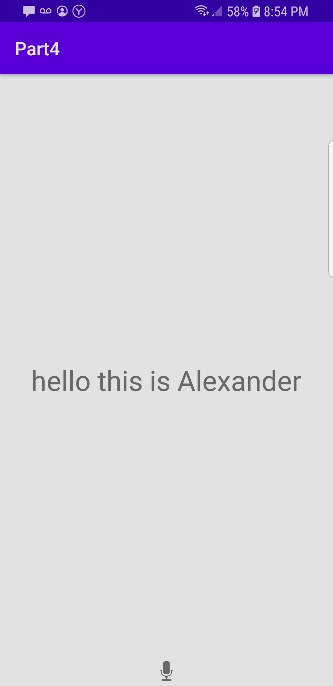
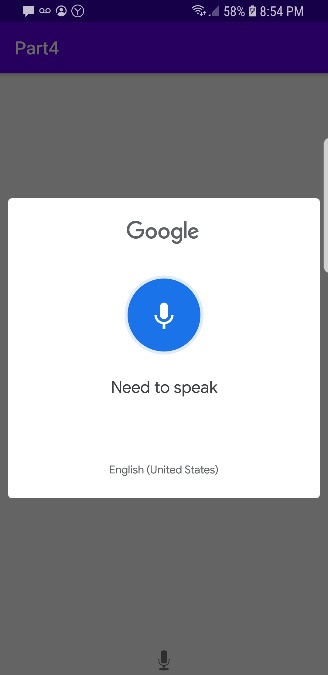
Part Two



Text to Speech App Guide (10 points)



Speech to Text App Guide (10 points)



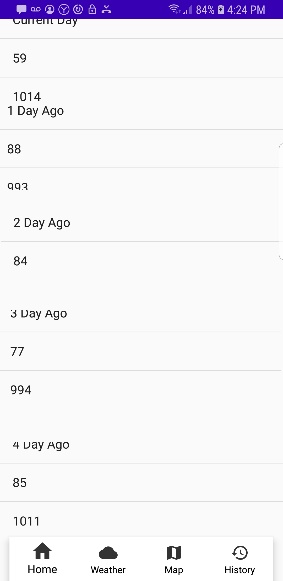
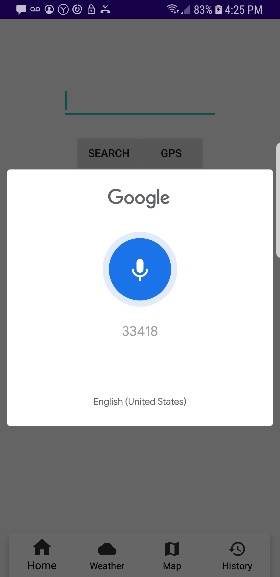
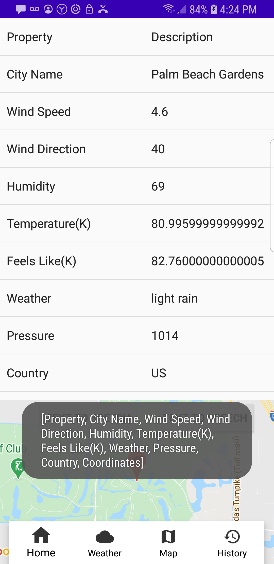
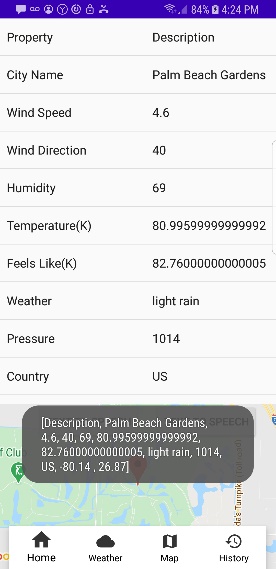
Conclusion

Parts 1-4 were easy and just a matter of following instruction.

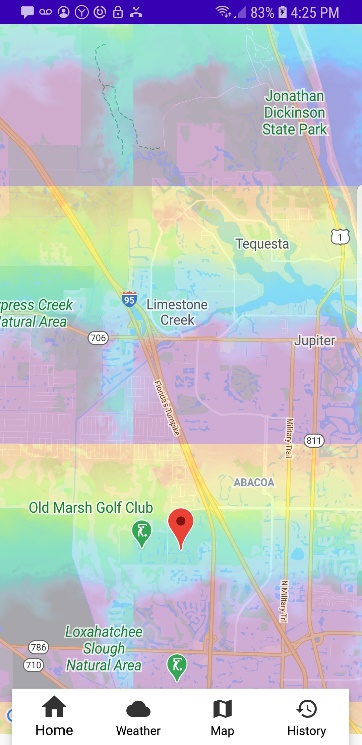
**Multi-Activity App (50 points / 10 points for GitHub / 10 points each for 4 Activities)**

**IMAGES**

History Page Speech-To-Text Text-to-Speech(Left) Text-to-Speech(Right)

 ** ** 

Maps Button



Actual Code – On GitHub

**Conclusion /w Errors and Solutions**

Issue 1: 5 Day History Solutions

**long** oneDay = 24 \* 60 \* 60;  
**long** currentDay = System.*currentTimeMillis*() / 1000L;  
**long** fourthDay = currentDay - oneDay;  
**long** thirdDay = currentDay - 2 \* oneDay;  
**long** secondDay = currentDay - 3 \* oneDay;  
**long** firstDay = currentDay - 4 \* oneDay;

Solution: Create a variable onDay which is the amount of seconds in a day as shown above. Then set the current day to the current time in UNIX as shown above. Then create 4 different variables for the previous four days.

Issue 2: Fixing the public static class

Problem: In order to get the data saved in the class, you need an if statement as shown below to make sure the data isn’t null. If the data shows NULL then the app will crash

Solution:

WeatherData w = weather.*getWeatherData*();  
**if**(w != **null**) {  
 **lat\_value** = Double.*parseDouble*(w.**Lat**);  
 **long\_val** = Double.*parseDouble*(w.**Long**);  
}**else** {  
  
}

Issue 3: Saving the Lat and Long variable to public static class to access for map and history

Problem: In order to access the lat and long value from different activities you can implement the solution below. Or you can call the GPS Tracker and rewrite all the code which would be unnecessary and clutter the code.

Solution:

In WeatherData.java file-

**public class** WeatherData {  
 **public** String **cityName**;  
 **public** String **windSpeed**;  
 **public** String **windDirection**;  
 **public** String **humidity**;  
 **public** String **temperature**;  
 **public** String **feelsLike**;  
 **public** String **Lat**;  
 **public** String **Long**;  
 **public** String **coords**;  
  
}

In Main Activity file-

**public static** WeatherData *WeatherData*;  
**public static** WeatherData getWeatherData() {  
 **return** *WeatherData*;  
}

*WeatherData*.**Lat** = String.*valueOf*(json.getJSONObject(**"coord"**).getString(**"lat"**));  
*WeatherData*.**Long** = String.*valueOf*(json.getJSONObject(**"coord"**).getString(**"lon"**));

Issue 4: Maps not working

Solution 1: First Check if your API is in the meta tag as shown below. This meta tag should be in the manifest file.

<**meta-data  
 android:name="com.google.android.geo.API\_KEY"  
 android:value="API KEY HERE NOT ABOVE "** />

Solution 2: Check if your ID to get the fragment is correct.

SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()  
 .findFragmentById(R.id.***map2***);  
mapFragment.getMapAsync(**this**);

Solution 3: Make sure the lat and long values are saved properly in the onMapReady function.

**public void** onMapReady(GoogleMap googleMap) {  
 **mMap** = googleMap;  
 **double** lat\_value = 0;  
 **double** long\_val = 0;  
 *// Add a marker in Sydney and move the camera* **if**(*WeatherData* != **null**) {  
 lat\_value = Double.*parseDouble*(*WeatherData*.**Lat**);  
 long\_val = Double.*parseDouble*(*WeatherData*.**Long**);  
 }**else** {  
 }  
 LatLng TutorialsPoint = **new** LatLng(lat\_value, long\_val);  
 **mMap**.addMarker(**new** MarkerOptions().position(TutorialsPoint).title(**"Tutorialspoint.com"**));  
 **mMap**.moveCamera(CameraUpdateFactory.*newLatLng*(TutorialsPoint));  
 **mMap**.animateCamera(CameraUpdateFactory.*newLatLngZoom*(TutorialsPoint,14));  
  
}

Issue 5: Basic Git Commands instead of going through Android Studio

Solution:

Step One: Go to where the git directory is.

Step Two: Track all files in your file type out the following: Git add .

Step Three: Commit the files and changes type out the following: Git commit -m “YOUR COMMENT HERE”

Step Four: Update changes to your remote repository type out the following: Git push