HOMEWORK ASSIGNMENT 4

CSCI 571 – Fall 2024

Abstract

JSON, Node.js, MongoDB and the Tomorrow.io API on Android

This content is protected and may not be shared, uploaded,

Assignment 4: Android Weather App

Table of Contents

1. Objectives

- Become familiar with Java, JSON, Android Lifecycle and Android Studio for Android app development.
- Learn the essentials of Google's Material design rules for designing Android apps
- Learn to use the Google Maps APIs and Android SDK.
- Get familiar with third party libraries like Picasso, Glide and Volley.

The objective is to create an Android application as specified in the rest of this document.

2. Background

2.1 Android Studio

Android Studio is the official Integrated Development Environment (IDE) for Android application development, based on IntelliJ IDEA - a powerful Java IDE. On top of the capabilities you expect from IntelliJ, Android Studio offers:

- Flexible Gradle - based build system.
- Build variants and multiple apk file generation.
- Code templates to help you build common app features.
- Rich layout editor with support for drag and drop theme editing.
- Lint tools to catch performance, usability, version compatibility, and other problems.
- ProGuard and app-signing capabilities.
- Built-in support for Google Cloud Platform, making it easy to integrate Google Cloud Messaging and App Engine.

More information about Android Studio can be found at:

http://developer.android.com/tools/studio/index.html

2.2 Android

Android is a mobile operating system initially developed by Android Inc.; a firm purchased by Google in 2005. Android is based upon a modified version of the Linux kernel.

The Official Android home page is located at:

http://www.android.com

The Official Android Developer home page is located at:

http://developer.android.com

3. Prerequisites

This homework requires the use of the following components:

- Download and install Android Studio. Technically, you may use any other IDE other than Android Studio such as Eclipse, but the latest SDKs may not be supported with Eclipse. We will not be providing any help on problems arising due to your choice of alternate IDEs.
- You must use the emulator. Everything should just work out of the box.
- If you are new to Android Development, Hints are going to be your best friends!

4. High Level Design

This homework is a mobile app version of Assignment 3 with new features.

In this exercise, you will develop an Android application, which allows users to search for cities to see weather summary, look at detailed information about them, pin those cities to favorites and post on Twitter about the weather.

You should reuse the Node.js backend service you developed in Assignment 3 and follow the same API call requirements. In case you need to change something in Node, make sure you do not break your Angular assignment (or deploy a separate copy) as the grading will not be finished at least until 2 weeks later.

PS: This app has been designed and implemented in a **Pixel 5 emulator running Android 15**. It is highly recommended that you use the same virtual device to ensure consistency.

Recorded demo video will be on an emulator, no personal devices allowed.

5. Implementation

5.1 App Icon and Splash Screen

To get this icon/image, go to the icon's web page specified in the hints. Using advanced export option, set the colors and a correct size to download the PNG icon.

The app begins with a welcome screen (Figure 2) which displays the icon downloaded above.
This is also where we credit Tomorrow.io for using their APIs and data. This screen is called Splash Screen and can be implemented using many different methods. The simplest is to create a resource file for launcher screen and adding it as a style to AppTheme.Launcher (see hints).

This image is also the app icon as shown in Figure 1

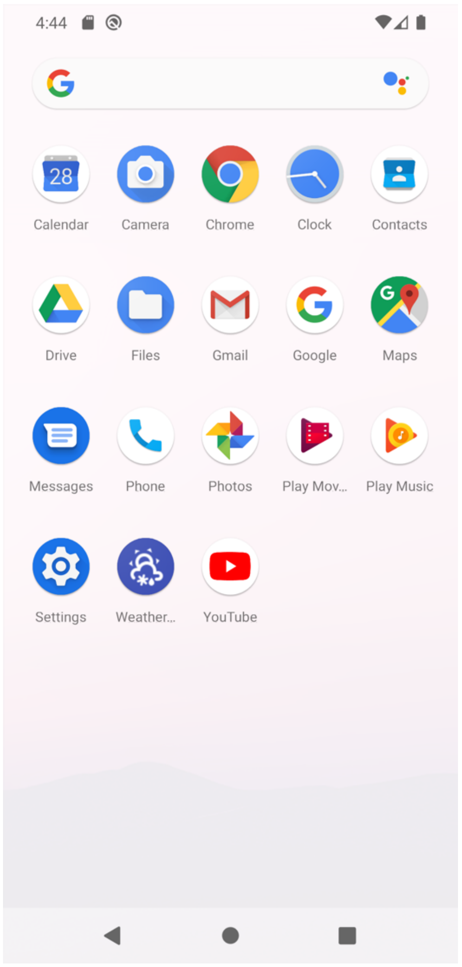


Figure 1: App Icon

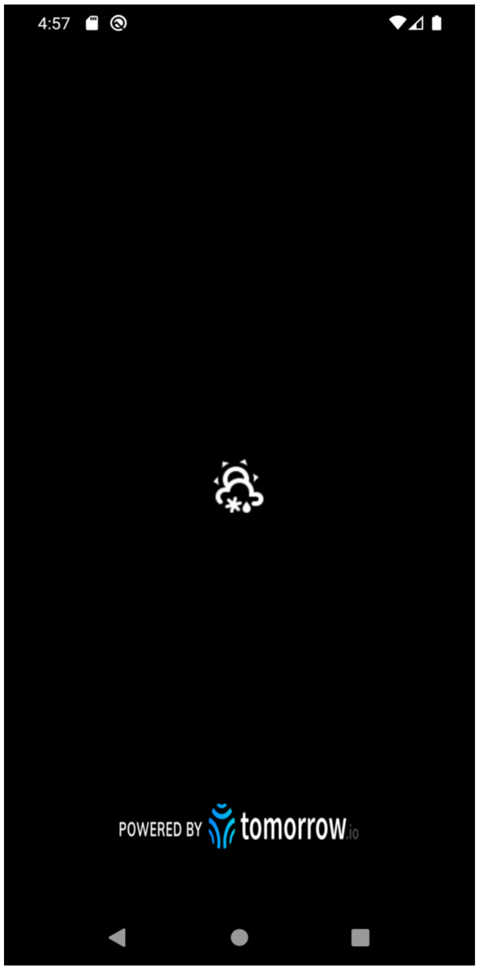
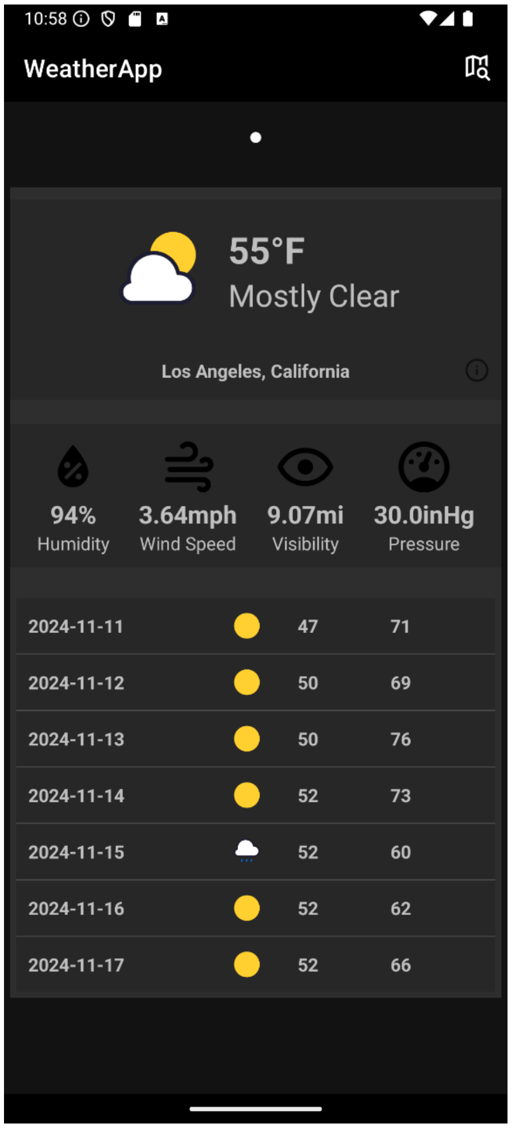


Figure 2: Splash Screen

5.2 Home Screen / Current Location View

As soon as you open the app, the summary view of the current location is displayed. The tabs on the right swipe are described in a later section.
The location can either be fetched by using the IPinfo API.

Note: The current location tab always stays and cannot be deleted/removed by any means.



55°F
Mostly Clear

Los Angeles, California

94% Humidity
3.64mph Wind Speed
9.07mi Visibility
30.0inHg Pressure

2024-11-11	●	47	71
2024-11-12	●	50	69
2024-11-13	●	50	76
2024-11-14	●	52	73
2024-11-15	☁	52	60
2024-11-16	●	52	62
2024-11-17	●	52	66

Figure 3: The Home page for the app

This view is also called the Home screen view. The Home screen view has a selected amount of information which might be most useful for a user. It has the following fields:

Card 1:

- Icon: Weather Status icon corresponding to the weather code for the current weather condition (same as in previous assignments).
- Temperature: The current temperature. Make sure that the temperature is rounded off to nearest integer.
- Summary: The current weather conditions obtained using the weather code.
- City: Name of the city to which this card belongs. You can use either the autocomplete suggestion or the location information returned in the IPinfo API call.
- On click: Clicking this card will open a new "detailed weather information view" described in section 5.4.

Card 2:

This card shows 4 selected values from the "currently" property. i.e., humidity, windspeed, visibility, pressure. Make sure all floats are terminated to exactly 2 digits after the decimal point. Also ensure that the correct icons are used, mapping is given in the hints.

- "humidity" is returned between 0 and 1 by the API - convert it to % by either mathematical operations or by dropping everything before "." followed by a % symbol.
- "windSpeed" property from currently JSON is used. The unit is mph.
- "visibility" property from "currently" JSON is used. The unit is mu.
- "pressure" property from "currently" JSON is used. The unit is inHg.

Card 3:

This card is used to display a quick overview of the next week's temperature. The predictions are available inside the "data" array of "daily" JSON. Each row in the table consists of:

- Date: "time" property converted to a YYYY-MM-DD format.
- Icon: based on the "weathercode" property.
- Minimum temperature: "temperatureLow" property in the JSON, rounded off to integer.
- Maximum temperature: "temperatureHigh" property in the JSON, rounded off to integer.
- This must display 5-6 days of information. You can use a ScrollView to achieve this table look. See hints.

Notes:

- JSON property names are written for reference. For exact structure please refer to HW5
- If any of the details are missing, you can use a default value of 0 or "N/A".
- All floats are rounded to exactly 2 digits after the decimal point.
- All temperatures must be rounded off to nearest integers. See hints.
- String manipulation in Java.

5.3 Searching for a new city

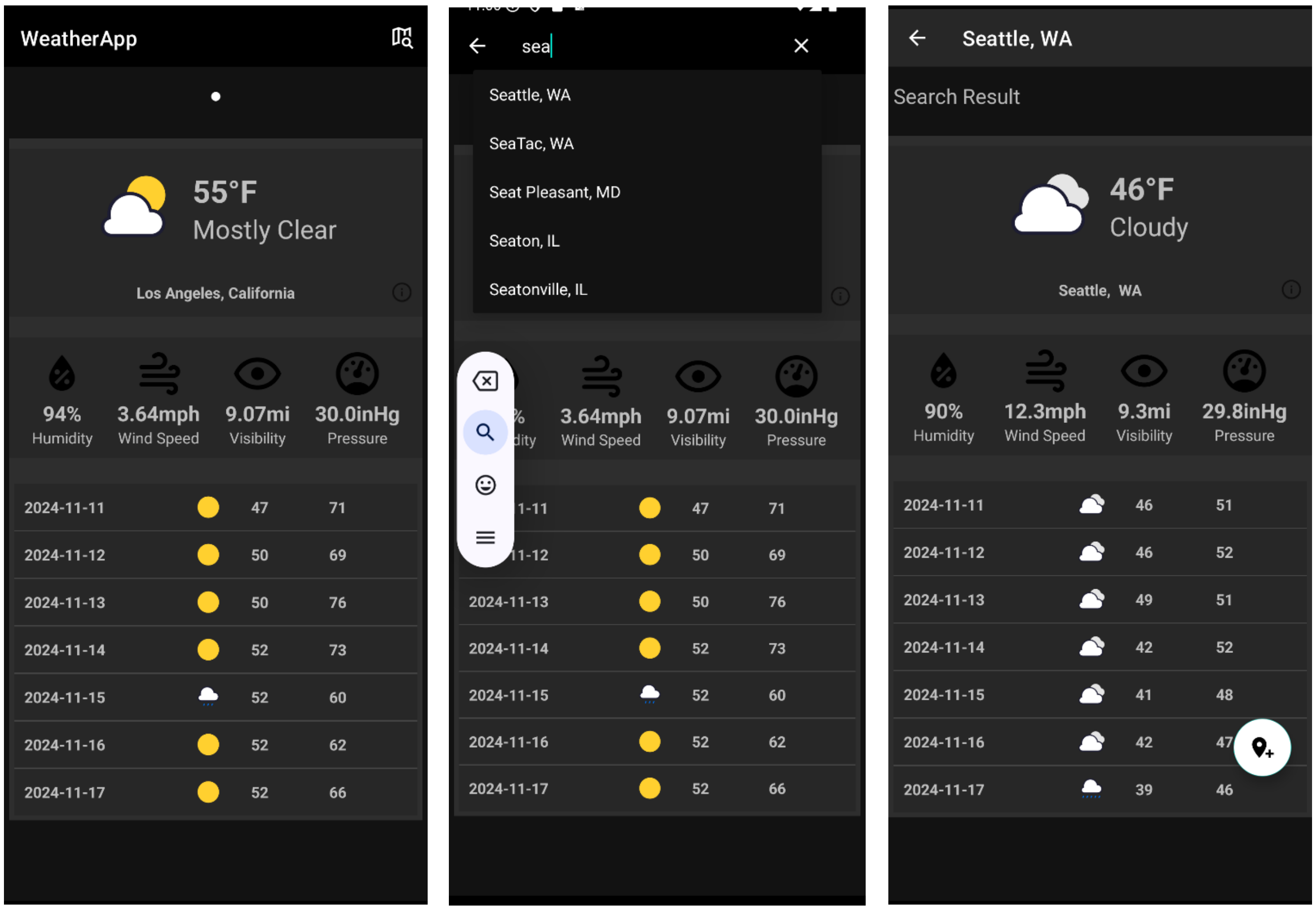
- In a typical weather app, we need to provide only the city name to search. Our app will be a Google recommended "Searchable" app.
- On top right side, there will be a search button which opens a textbox where the user can type name of a city. See hints for icon.
- The user is provided with suggestions of city names using the autocomplete API used in Assignment 3.
- When the user taps on a suggestion, it is filled inside the search box and clicking enter/next takes the user to the next page.
- Before you get the data from your backend server, a progress bar should display on the screen as indicated in Figure 12.
- On the next page, the user will then be redirected to a new page/activity which will show the Home Screen view (described above in section 5.2).

This component involves 2 things:

- Implementing a searchable – see hints.
- Implementing auto-complete – see hints.

Notes:

- If your Node backend from Assignment 3 returns only the city name and not the state and country, you can still use that.
- If you still must change Node backend, make sure you don't break the Angular app (if grading is still in progress) or deploy a newer instance of Node.
- Reusing the Home Screen view and the corresponding logic to dynamically set the value will make it much faster to implement this.
- Do not forget the static "Search Results" label on top.
- Favorites button: This button will add/remove a city to/from the favorites. This can be implemented using a Floating Action Button. The icon of the button should also change based on whether the city currently belongs to favorites or not. For more details see video and implementation hints.
- Clicking on the top card, also opens the same details view as described in section 5.4.



WeatherApp

55°F
Mostly Clear

Los Angeles, California

94% Humidity | 3.64mph Wind Speed | 9.07mi Visibility | 30.0inHg Pressure

2024-11-11	☀	47	71
2024-11-12	☀	50	69
2024-11-13	☀	50	76
2024-11-14	☀	52	73
2024-11-15	🌧	52	60
2024-11-16	☀	52	62
2024-11-17	☀	52	66

sea

Seattle, WA
SeaTac, WA
Seat Pleasant, MD
Seaton, IL
Seatonville, IL

3.64mph Wind Speed | 9.07mi Visibility | 30.0inHg Pressure

1-11	☀	47	71
1-12	☀	50	69
2024-11-13	☀	50	76
2024-11-14	☀	52	73
2024-11-15	🌧	52	60
2024-11-16	☀	52	62
2024-11-17	☀	52	66

Seattle, WA

Search Result

46°F
Cloudy

Seattle, WA

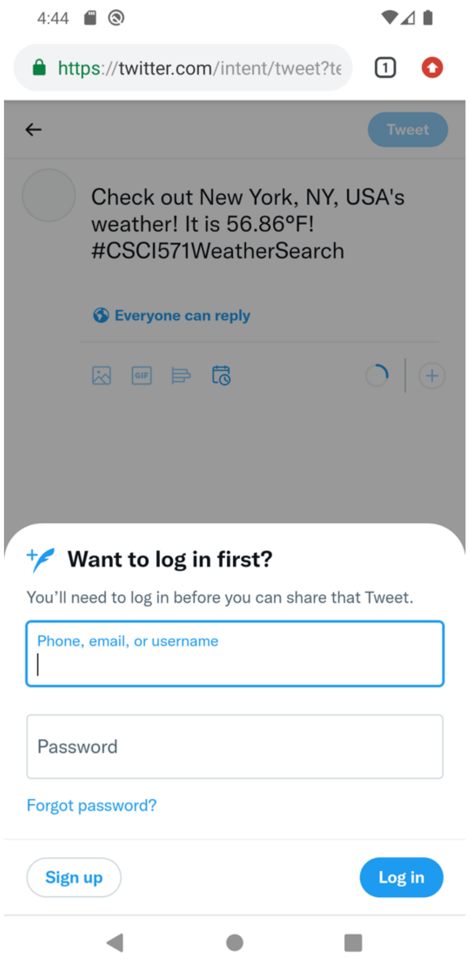
90% Humidity | 12.3mph Wind Speed | 9.3mi Visibility | 29.8inHg Pressure

2024-11-11	☁	46	51
2024-11-12	☁	46	52
2024-11-13	☁	49	51
2024-11-14	☁	42	52
2024-11-15	☁	41	48
2024-11-16	☁	42	47
2024-11-17	🌧	39	46

Figure 4: Flow for searching a city

5.4 Detailed Weather Information view

- Notice the information icon on card 1 in the Home Screen view. When the user clicks on this card, a detailed view about the corresponding city is displayed.
- Add a ripple effect to the card.
- The detailed view has 3 tabs. These tabs are called "Fragments". They can be implemented either with a tabbed activity or generated programmatically (see hints).
- The tabs are called "Today", "Weekly" and "Weather Data".
- There is a X (Tweet) button in the Action Bar click which opens a browser with the X (Twitter) intent like Assignment 3. See hints.
- The Tweet should be of the form: (Check Out CITY's Weather! It is X°F! #CSCI571WeatherSearch)


4:44
https://twitter.com/intent/tweet?te
Tweet
Everyone can reply
Want to log in first?
You'll need to log in before you can share that Tweet.
Phone, email, or username
Password
Forgot password?
Sign up
Log in

Check out New York, NY, USA's weather! It is 56.86°F! #CSCI571WeatherSearch

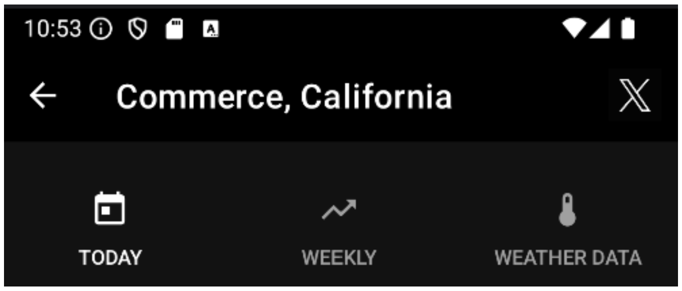


Figure 5: Tab view with back and X (Twitter) buttons and city name

Figure 6: X (Twitter) intent

Notes:

- You can try to pass the weather JSON from the Home Screen view to the Detailed View. You can also choose to query Tomorrow.io again. In any case, there should be a progress bar every time you are waiting for data.
- The default tab to be opened is the Today tab.

- The back button takes the user back to the Home Screen view. See hints.
- The app bar must also contain the name of the city that the current view is for.

5.4.1 Today tab

The Today tab contains information from the "currently" property in JSON. It contains 9 cards with a gray background an icon, description and the value of that parameter. See hints.
The 9 cards in this tab indicate:

1. Wind Speed: "windSpeed" property of the "now" JSON. Unit: mph/
2. Pressure: "pressureSeaLevel" property of the "now" JSON. Unit: inHg/
3. Precipitation: "precipitationIntensity" property of the "now" JSON. Unit: %/
4. Temperature: "temperature" property of the "now" JSON. Unit: F/
5. Icon: Same as above.
6. Humidity: "humidity" property of the "now" JSON. Unit: %.
7. Visibility: "visibility" property of the "now" JSON. Unit: mi.
8. Cloud Cover: "cloudCover" property of the "now" JSON. Unit: %.
9. UvIndex: "unIndex" property of the "now" JSON.

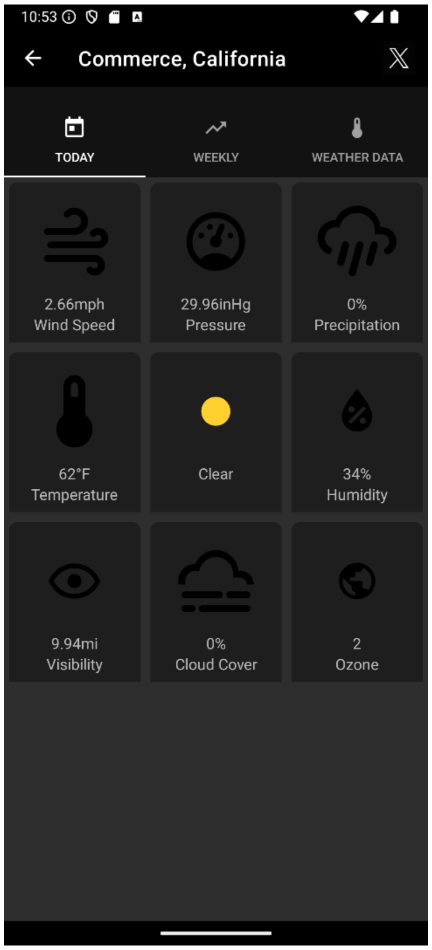


Figure 7: Today tab

Notes:

- Handle any missing values, otherwise the app can crash.
- Make sure all floating values are exactly 2 digits after the decimal point. Temperature must be rounded off to the nearest integer. See hints.
- If a value is not present/malformed you can choose to display 0 or NA.
- Make sure that the cards are well-styled and have appropriate sizes with centered images/texts.

5.4.2 Weekly tab

This tab indicates what the weather will look like next week. The data belongs to "daily" API call.

Graph view:

- This view uses Highcharts Android library. You must add this as a dependency.
- This chart contains two LineCharts i.e. AreaRange Chart.
- Extract the "temperatureLow" and "temperatureHigh" for each day from the "data" array and display the Lines in given color.
- Make sure you change the Legend, font and line colors.
- There are 3 hints which should be enough to implement this graph view.

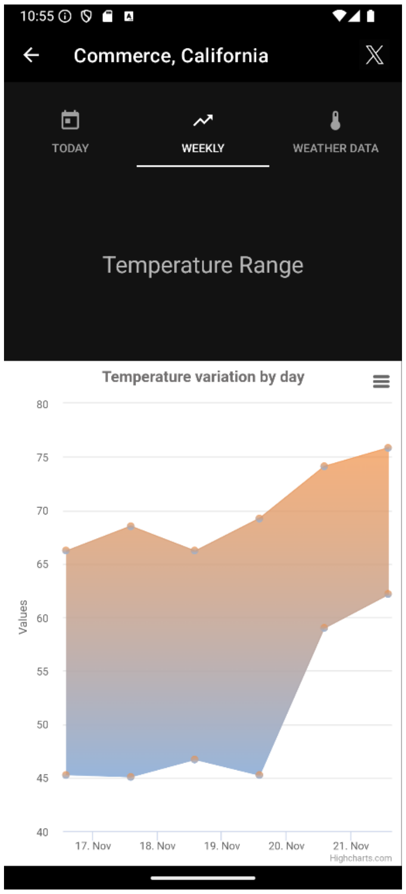


Figure 8: Weekly Tab

5.4.3 Weather Data tab

This tab indicates what the weather will look like in the coming week. The data belongs to "daily" properties' "data" array. There are 2 components to this page:

Graph view:

- This view uses HighCharts 3rd party library. You must add this as a dependency.
- This chart contains 3 data: Cloud Cover, Precipitation and Humidity.
- Extract the "Cloud Cover," "Precipitation" and " Humidity" from now and enter the "data" array and display the chart in given colors.
- Make sure you change the style of the chart as referred in the video.
- There are 3 hints which should be enough to implement this graph view.

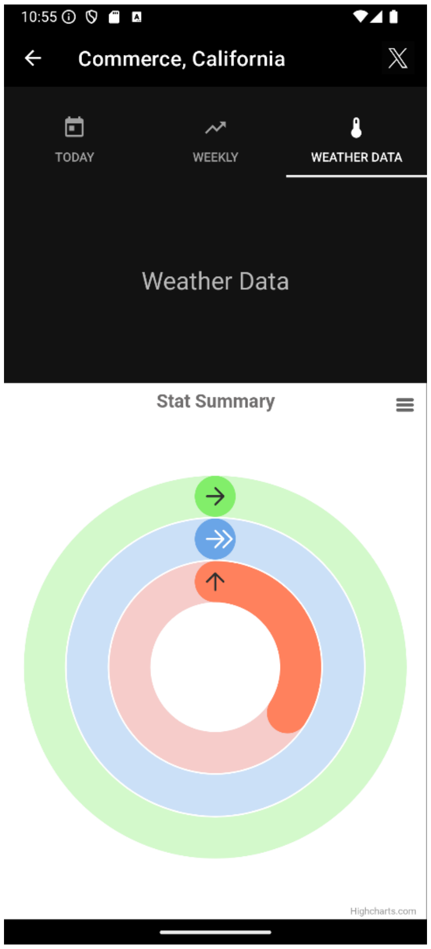
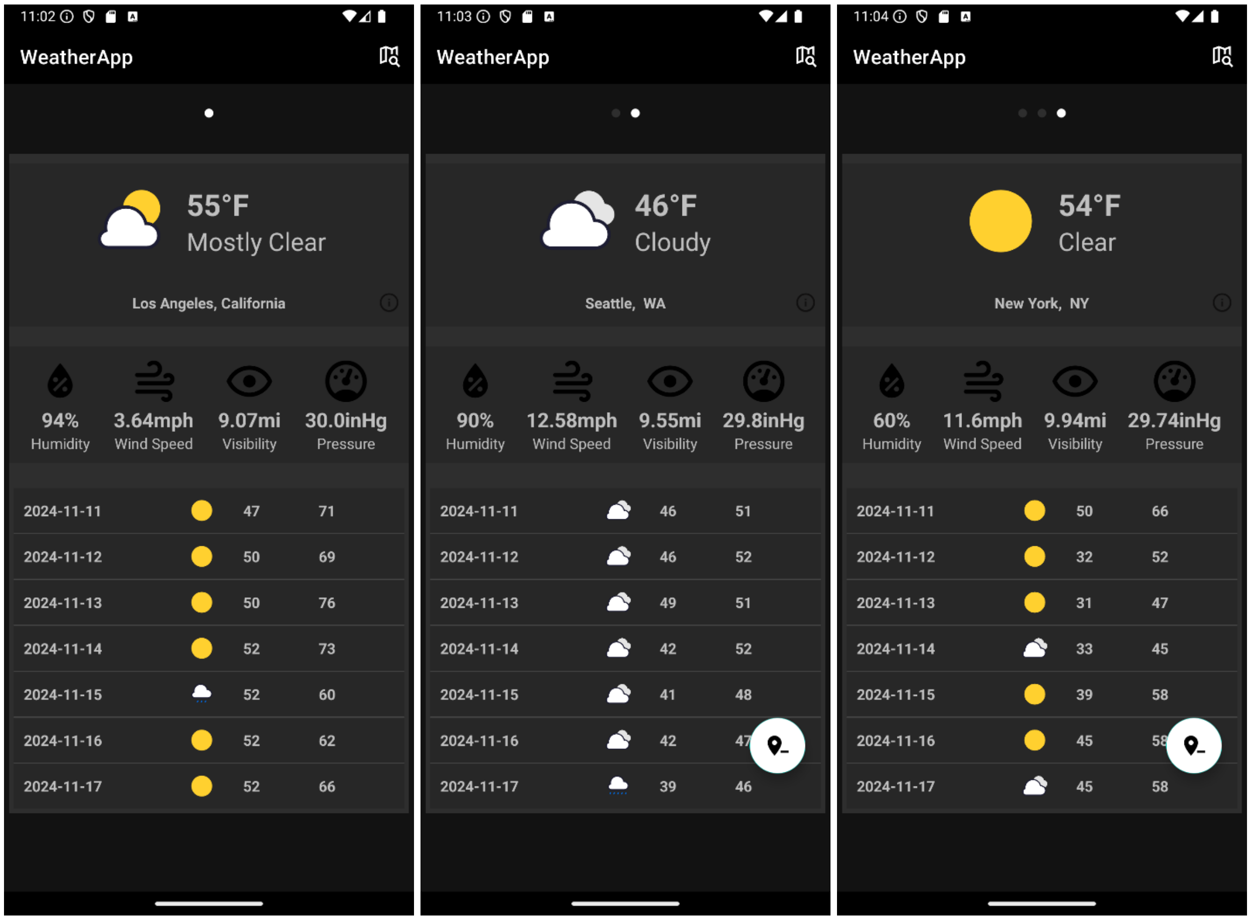


Figure 9: Weather Data tab

5.5 Favorite cities

This will be one of the most complicated parts of the assignment. The idea is to give users an ability to add a city to Favorites. The favorite cities persist even after the user has closed the app. The favorite cities are added as a Dynamic page/tab/fragment on the home page.

Again, consider reusing the view and the code that you wrote for search and the home location tabs. The view, onclick behaviors are all the same with additional buttons.



55°F
Mostly Clear
Los Angeles, California

94%	3.64mph	9.07mi	30.0inHg
Humidity	Wind Speed	Visibility	Pressure

2024-11-11	☀	47	71
2024-11-12	☀	50	69
2024-11-13	☀	50	76
2024-11-14	☀	52	73
2024-11-15	🌧	52	60
2024-11-16	☀	52	62
2024-11-17	☀	52	66

46°F
Cloudy
Seattle, WA

90%	12.58mph	9.55mi	29.8inHg
Humidity	Wind Speed	Visibility	Pressure

2024-11-11	⛅	46	51
2024-11-12	⛅	46	52
2024-11-13	⛅	49	51
2024-11-14	⛅	42	52
2024-11-15	⛅	41	48
2024-11-16	⛅	42	47
2024-11-17	🌧	39	46

54°F
Clear
New York, NY

60%	11.6mph	9.94mi	29.74inHg
Humidity	Wind Speed	Visibility	Pressure

2024-11-11	☀	50	66
2024-11-12	☀	32	52
2024-11-13	☀	31	47
2024-11-14	⛅	33	45
2024-11-15	☀	39	58
2024-11-16	☀	45	58
2024-11-17	⛅	45	58

Figure 10: Favorite cities – Left to right: Default location without remove button, two cities added to favorites. Also notice the dot indicators on top

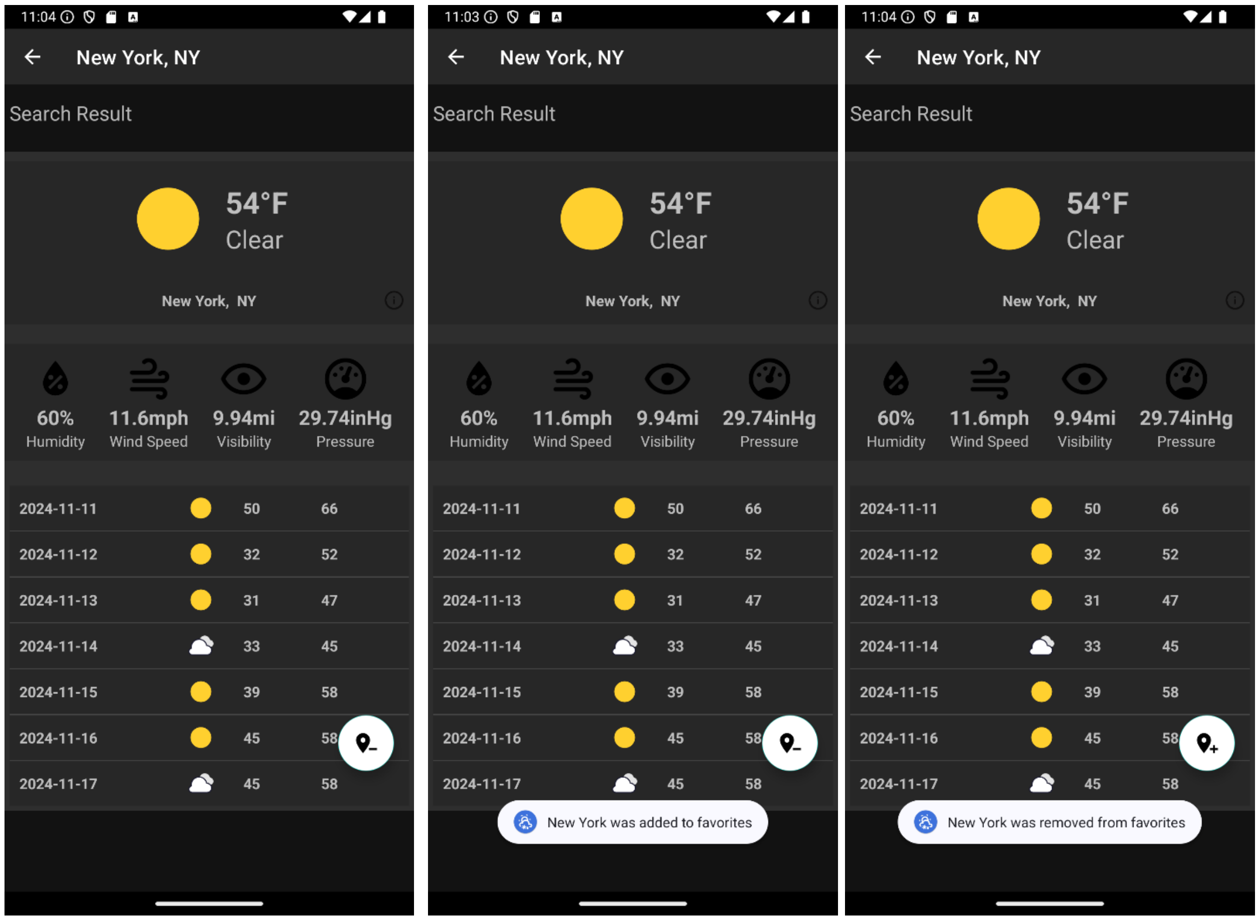
5.5.1 Adding to Favorites

Once a user has searched for a city, the summary view contains a FloatingActionButton which indicates "add to favorites" – icon of a map pin with a plus sign.

5.5.2 Removing from Favorites

"Remove from favorites" is the same FloatingActionButton as add, but with a different icon of marker with a minus sign. This button appears on 2 places:

- On the corresponding dynamically added page on the home screen
- On the search result page when the city is searched again.



11:04

New York, NY

Search Result

54°F
Clear

New York, NY

60%	11.6mph	9.94mi	29.74inHg
Humidity	Wind Speed	Visibility	Pressure

2024-11-11	50	66
2024-11-12	32	52
2024-11-13	31	47
2024-11-14	33	45
2024-11-15	39	58
2024-11-16	45	58
2024-11-17	45	58

11:03

New York, NY

Search Result

54°F
Clear

New York, NY

60%	11.6mph	9.94mi	29.74inHg
Humidity	Wind Speed	Visibility	Pressure

2024-11-11	50	66
2024-11-12	32	52
2024-11-13	31	47
2024-11-14	33	45
2024-11-15	39	58
2024-11-16	45	58
2024-11-17	45	58

New York was added to favorites

11:04

New York, NY

Search Result

54°F
Clear

New York, NY

60%	11.6mph	9.94mi	29.74inHg
Humidity	Wind Speed	Visibility	Pressure

2024-11-11	50	66
2024-11-12	32	52
2024-11-13	31	47
2024-11-14	33	45
2024-11-15	39	58
2024-11-16	45	58
2024-11-17	45	58

New York was removed from favorites

Figure 11: Toast messages on adding/removing a city from favorite

<u>Notes:</u>

- Display "CITY was added/removed to/from favorites" using a Toast Hints.
- Make sure that the add/remove icon toggles correctly.
- The favorite cities must persist across sessions i.e., opening and closing should retain the view. You should use the data stored in MongoDB Atlas.

- The favorite cities are also indicated with dots to indicate the active tab. See hints.
- The favorites tab must update dynamically when the user comes back to home screen from a search query. E.g. If the favorites are empty and the user searches for New York, from the search Home Screen view if the user clicks the "add to favorite" button and navigates back to the home screen, instead of just current location there should now also be the New York tab! And similarly, removing a favorite city must delete the tab.
- The dynamic tabs can be implemented in multiple ways. See hints.

5.6 Progress bar

Every time the user has to wait before user can see the data, you must display a progress bar. The progress bar is constant for every screen and just says "Fetching Weather/Loading". One idea would be to display the progress bar by default (where needed) and then hide it as soon as the data is received/view is prepared.
See hints for progress bar related ideas and styling.

Note:
Based on your implementation, you might need to put progress bars on different places. During demo, there should NOT be any screen with default/dummy/placeholder values or an empty screen.

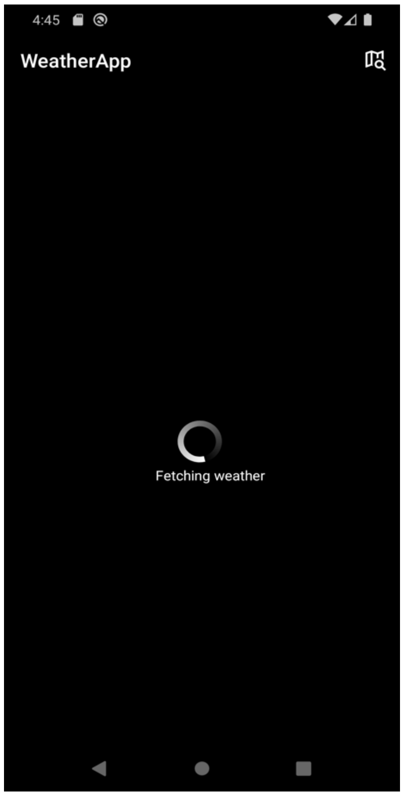


Figure 12: Progress bar

5.7 Summary of detailing and error handling

1. Make sure there are no conditions under which the app crashes.
2. All floating-point numbers must have EXACTLY 2 digits after the decimal point. All temperatures must be rounded off to nearest integer. See hints.
3. Make sure the units are correct.
4. Make sure all icons and texts are correctly positioned/centered as in the video/screenshots.
5. Make sure the graph legends and axes are correctly displayed.
6. Make sure there is a progress bar every time there is nothing to show.
7. The favorites button work as expected and must dynamically update the tabs on the home screen
8. Make sure you use the right icons as listed in the hints.

5.8 Additional Info

For things not specified in the document, grading guideline, or the video, you can refer Piazza. But keep in mind about the following points:

- Always display a proper message and don't crash if an error happens.
- Always display a loading message if the data is loading.
- You can only make HTTP requests to your backend Node.js on AWS/GAE/Azure.
- All HTTP requests should be asynchronous and should not block the main UI thread. You can use third party libraries like Volley to achieve this in a simple manner.

6. Implementation Hints

6.0 Structuring your app

There are many ways you could do this. The easiest way would be to create following Activities:
(this is just a suggestion and NOT a mandate):

- Main
- Searchable
- Details

And the following Fragments:

- Favorites
- Tabs

Try to reuse as much of your code as you can - there are multiple places which share the same logic – Home view, favorites view and search view are the same. All the details pages are essentially the same.

All the hints below are just pointers, feel free to research and find a different way of doing things. You do NOT have to stick to these resources.

6.1 Icons

6.1.1 Icons

The images used in this homework are taken from https://materialdesignicons.com/ .

You can choose to work with xml/png/jpg versions. We recommend using xml as it is easy to modify colors by setting the Fill Colors.

Icon Name	Usage
weather-partly-snowy-rainy	App Icon and Splash screen
Tomorrow.io attribution logo	https://github.com/Tomorrow-IO-API/tomorrow-weather-codes/blob/master/powered-by-tomorrow/Powered_by_Tomorrow-Black.png
weather-windy	Wind Card
gauge	Pressure Card
weather-pouring	Rain Card
thermometer	Temperature Card
water-percent	Humidity card
eye-outline	Visibility card
weather-fog	Cloud Cover Card
earth	UV card
information-outline	Information icon for card on summary views
map-marker-plus	Add to favorites
map-marker-minus	Remove from favorites
map-search-outline	Search bar icon
calendar-today	For “Today” tab
trending-up	For “Weekly” tab
thermometer	For “Weather Data” tab
twitter	Tweet Icon

6.2 Getting current location

To get the current location, you can use either IPinfo API or Android location services. Refer to assignment 3 for more details on IPinfo. You can perform this API call locally on device.

6.3 Third party libraries

Sometimes using 3rd party libraries can make your implementation much easier and quicker. Some libraries you may have to use are:

6.3.1 Google Play services

You will need this for various features like getting the current location and using Google Maps in your app.

You can learn about setting it up here:

https://developers.google.com/android/guides/setup

6.3.2 Volley HTTP requests

Volley can be helpful with asynchronously http request to load data. You can also use Volley network ImageView to load photos in Google tab. You can learn more about them here:

https://developer.android.com/training/volley/index.html

6.3.3 HighChart Android

In order to create graphs, we will use the HighChart Android. Documentation for Highchart Integration in Android can be found here: https://www.highcharts.com/blog/tutorials/highcharts-android-wrapper-tutorial/

For our purposes following these links should suffice:
Creating a AreaRange Chart: https://www.highcharts.com/demo/android/arearange
Creating Gauge Chart: https://www.highcharts.com/demo/android/gauge-activity

6.4 Adding tab indicators for favorite cities

https://stackoverflow.com/questions/20586619/android-viewpager-with-bottom-dots
https://stackoverflow.com/questions/38459309/how-do-you-create-an-android-view-pager-with-a-dots-indicator

6.5 Working with action bars and menus

https://developer.android.com/training/appbar/setting-up
https://stackoverflow.com/questions/38195522/what-is-oncreateoptionsmenumenu-menu

6.6 Finding resource IDs dynamically

This can be helpful in setting the 5 days table in the summary view.
https://stackoverflow.com/questions/4427608/android-getting-resource-id-from-string

6.7 Converting number to date in Java

https://stackoverflow.com/questions/535004/unix-epoch-time-to-java-date-object

6.8 Adding Ripple effect on click of a card

https://stackoverflow.com/questions/26942434/ripple-effect-on-android-lollipop-cardview

6.9 Displaying ProgressBars

https://stackoverflow.com/questions/5337613/how-to-change-color-in-circular-progress-bar

6.10 Dynamically changing ImageView

This will be used for icon fields based on summary property.
https://stackoverflow.com/questions/2974862/changing-imageview-source

6.11 SearchBar and AutoCompleteTextView

To implement the search functionality, these pages will help:
https://www.youtube.com/watch?v=9OWmnYPX1uc
https://developer.android.com/guide/topics/search/search-dialog

Working with the AutoCompleteTextView to show the suggestions might be a little challenging. This tutorial goes over how it is done so that you get an idea of how to go about it.
https://www.truiton.com/2018/06/android-autocompletetextview-suggestions-from-webservice-call/

In order to link your Search Bar with autocomplete suggestions, these links might help:
https://www.dev2qa.com/android-actionbar-searchview-autocomplete-example/https://stackoverflow.com/questions/34603157/how-to-get-a-text-from-searchview

6.12 Implementing Splash Screen

There are many ways to implement a splash screen. This blog highlights almost all of them with examples:
https://android.jlelse.eu/the-complete-android-splash-screen-guide-c7db82bce565

6.13 Adding the App Icon

https://dev.to/sfarias051/how-to-create-adaptive-icons-for-android-using-android-studio-459h

6.14 String manipulation in Java

https://www.guru99.com/java-strings.html

6.15 Adding ellipsis to long strings

https://stackoverflow.com/questions/6393487/how-can-i-show-ellipses-on-my-textview-if-it-is-greater-than-the-1-line

6.16 Floating Action Buttons

https://developer.android.com/guide/topics/ui/floating-action-button

6.17 Adding tabs in Android

https://www.spaceotechnologies.com/create-multiple-tabs-using-android-tab-layout/

6.18 Adding a button to ActionBar

https://developer.android.com/training/appbar/actions
https://stackoverflow.com/questions/12070744/add-back-button-to-action-bar
https://stackoverflow.com/questions/38195522/what-is-oncreateoptionsmenumenu-menu

6.19 Adding Toasts

https://stackoverflow.com/questions/3500197/how-to-display-toast-in-android

6.20 To create the weekly table

https://stackoverflow.com/questions/4427608/android-getting-resource-id-from-string
https://stackoverflow.com/questions/535004/unix-epoch-time-to-java-date-object
https://developer.android.com/reference/android/widget/ScrollView

6.21 To implement favorites

- Please read about how ViewPagers work: https://developer.android.com/training/animation/screen-slide
- Set OffScreenLimit to 10: https://stackoverflow.com/questions/11650152/viewpager-offscreen-page-limit
- And then follow these links:
  - https://medium.com/@rajatsingh1695/making-dynamic-tabs-and-fragments-in-android-using-JSON-data-70a59eb6f0e0
  - https://stackoverflow.com/questions/34306476/dynamically-add-and-remove-tabs-in-tablayoutmaterial-design-android
  - https://stackoverflow.com/questions/10396321/remove-fragment-page-from-viewpager-in-android

6.22 Passing variables to intent

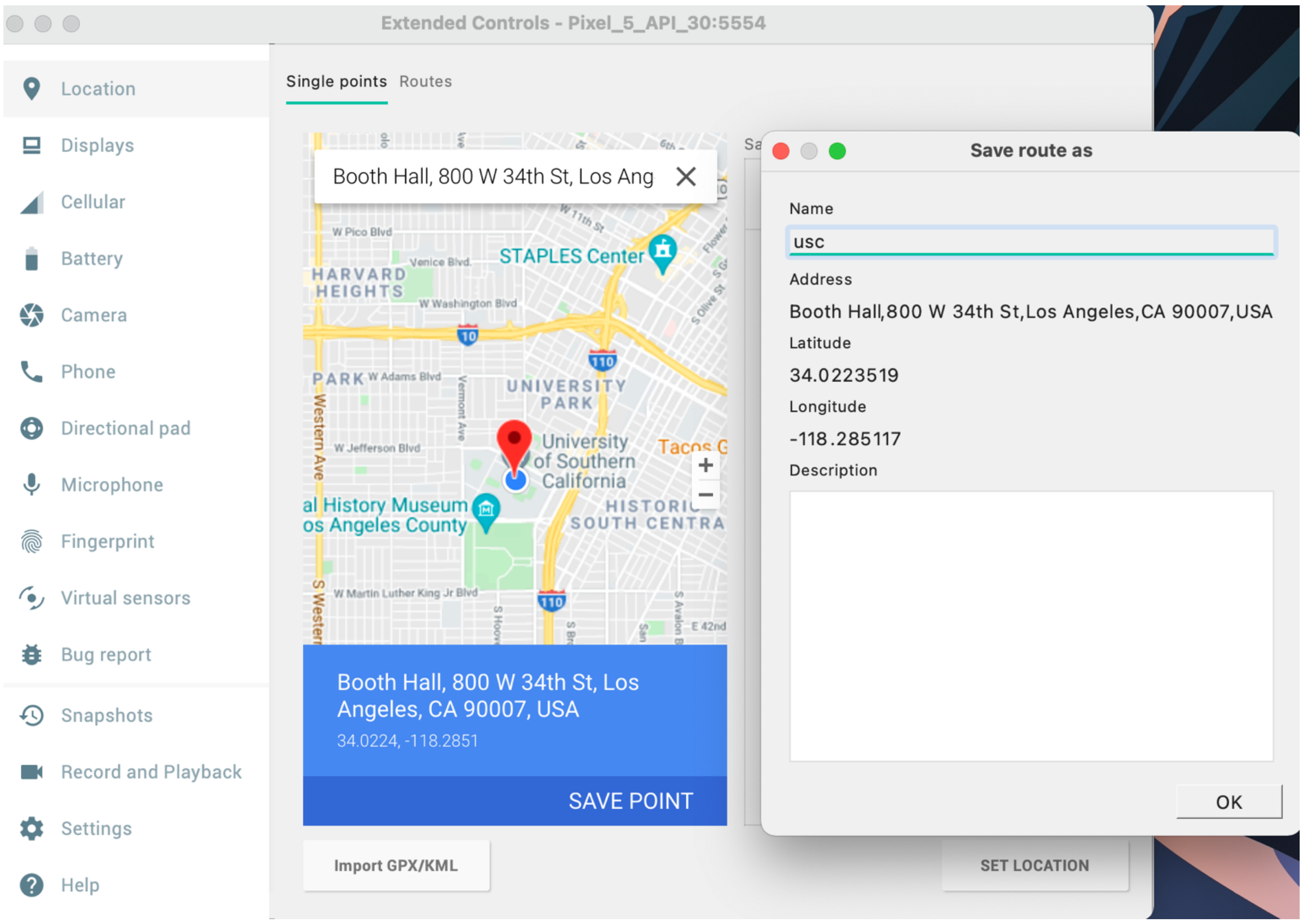
https://stackoverflow.com/questions/2405120/how-to-start-an-intent-by-passing-some-parameters-to-it

6.23 Rounding off to integers

https://stackoverflow.com/questions/2654839/rounding-a-double-to-turn-it-into-an-int-java

6.24 Location using Emulator Settings

If you choose to use Android location services to get the current location, then you can use the emulator settings.



Booth Hall, 800 W 34th St, Los Angeles, CA 90007, USA
34.0224, -118.2851

Name
usc
Address
Booth Hall,800 W 34th St,Los Angeles,CA 90007,USA
Latitude
34.0223519
Longitude
-118.285117
Description

Figure 14: Location Setting of Emulator

7. Material you need to Submit

You will have to demo your submission using a video recorded with **Zoom**. Details are included in a Final Presentation document on D2L Brightspace. **Demo is done on a MacBook using the simulator, and not a physical mobile device.**

You should run **Android Studio** → **Code** → **Code Cleanup** → **Whole Project**, to remove all compiled object files.

Then generate the following files:

1. **Your mobile source code** as a **ZIP file**. This file should contain your front-end code (Java or Kotlin), by zipping the top Android folder containing your project; name your zip file **Assignment4-<username>_mobile.zip**.
2. Your **back-end source code** (NodeJS files, possibly the same as Assignment #3), as a **ZIP file** of the NodeJS top folder; name your zip file **Assignment4-<username>_backend.zip.**
3. Your **video file** in **.mp4 format**, created using Zoom and the *Final Presentation* guidelines available on D2L Brightspace. Name the recording as **Assignment4-<username>.mp4**.

Then click **File Upload**. Note: make sure that BOTH the 2 .ZIP files and the .MP4 file (3 files) are **uploaded on the SAME submission**, as every submission will overwrite the previous one.

IMPORTANT

All videos, all discussions and explanations on Piazza related to this assignment are part of the assignment description and will be accounted into grading. So please review all Piazza threads before finishing the assignment.