ITIS/ITCS 4180/5180 Mobile Application Development
Final Exam

Basic Instructions:

1. This is the Final Exam.
2. In every file submitted you MUST place the following comments:
    a. Your Full Name.
3. This Final Exam is an individual effort. Each student is responsible for her/his own Final Exam and its submission.
4. Once you have picked up the exam, you may not discuss it in any way with anyone until the exam period is over.
5. During the exam, you are allowed to use the course videos, slides, and your code from previous homeworks and in class assignments. You are NOT allowed to use code provided by other students or from other sources.
6. Answer all the exam parts, all the parts are required.
7. Please download the support files provided with the Final Exam and use them when implementing your project.
8. Your assignment will be graded for functional requirements and efficiency of your submitted solution. You will loose points if your code is not efficient, does unnecessary processing or blocks the UI thread.
9. Export your Android project and create a zip file which includes all the project folder and any required libraries. The file name is very important and should follow the following format: **Lastname_Final.zip.** Submit the exported file using the provided canvas submission link.
10. **Failure to do the above instructions will result in loss of points.**
11. **Any violation of the rules regarding consultation with others will not be tolerated and will result disciplinary action and failing the course.**

Final (100 Points)

In this assignment you will develop a Trip Planner App. There you can add new Trips, add new places in the trips, delete places and display a map of the trip. We will be using Firebase as the for data storage and also you will be using the Google Places API.

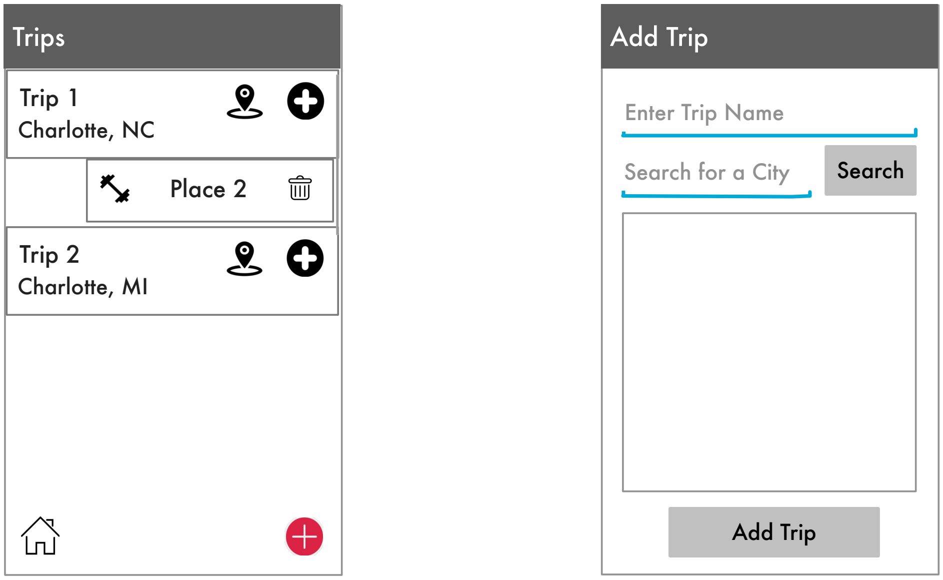
API Details:

1. Get an API Key for the Google places API. https://developers.google.com/places/web-service/get-api-key

Search for Cities:	Enables you to search for a list of cities matching the "input" parameter
Endpoint Url:	https://maps.googleapis.com/maps/api/place/autocomplete/json
Parameters: **GET Parameters**	• key=YOUR_KEY • types=(cities) • input=charlotte,nc or you can provide any other city which is provided by the user
Response:	Returns a list of cities (places) which match the entered input field.

Get Geo Coordinates	Enables you to retrieve the geo coordinates (latitude,longitude) of a given city using the place_id.
Endpoint Url:	https://maps.googleapis.com/maps/api/place/details/json
Parameters: **GET Parameters**	• key=YOUR_KEY • placeid=Value retrieved from the search cities API
Response:	Will return details for the provided placeid. This enables us to get the latitude and longitude of the provided placeid.

Get Places Close to Location	Enables you to retrieve the places close to the selected location, which is the latitude and longitude provided for the location attribute.
Endpoint Url:	https://maps.googleapis.com/maps/api/place/nearbysearch/json
Parameters: **GET Parameters**	• key=YOUR_KEY • location=latitude,longitude • radius=1000
Response:	Returns a list of places which are within 1000 meters of the provided location. Please note that In addition to the place information, the result includes the latitude and longitude information is provided for each place in the list.



(a) Trips Screen **(b) Add Trip Screen**

Figure 1: Trip App

<u>Part 1 (Trips Screen, 35 points):</u>

1. This is the launcher screen for the application. This screen will display the saved trips, which should be retrieved from Firebase. Each trip should have a trip name, and city information. In addition, each trip should also maintain a list of places that will be included in the trip.
2. Clicking on the bottom add icon (red) should show the Add Trip Screen.
3. The screen should use ListView/RecyclerView to display the trips stored on Firebase. For each trip item:
   a) Under each trip you should show the list of places added to this trip as shown in Figure 1(c). Note that Place 2 has been added to Trip 1.
      - For each place under a trip row item if the user clicks on the delete button it should delete this place from the associate trip and should reload the list to display the updated trip information and places.

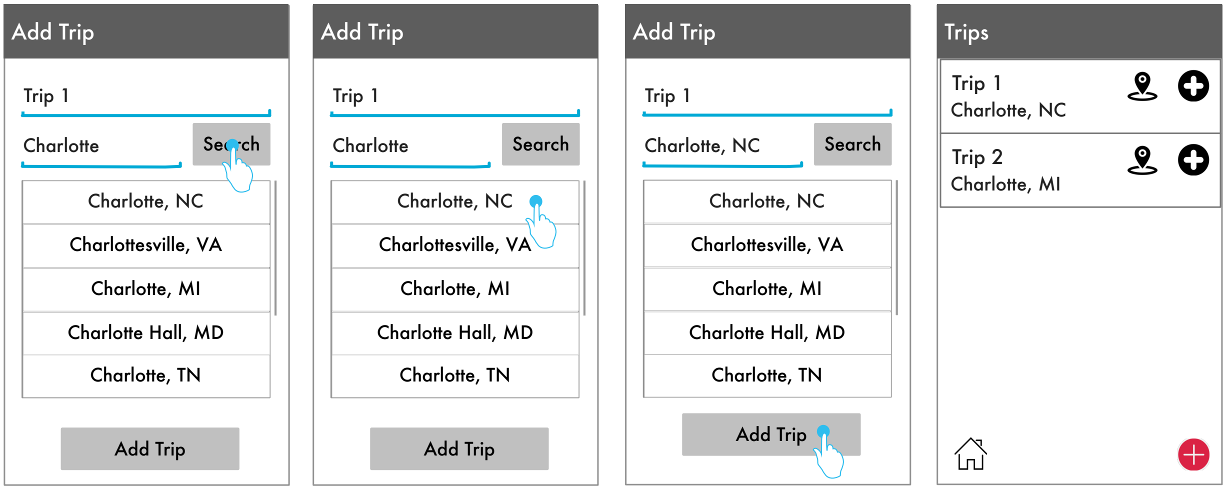
   b) Clicking on the add icon on any trip item should show the Add Places Screen, the Trips Screen should pass the required trip information to the Add Places Screen.
   c) Clicking on the location icon on any trip item should show the Trip Map Screen. The Trips Screen should pass the required trip information not the Trip Map Screen.

<u>Part 2 (Add Trip Screen, 40 points):</u>

1. This screen allows the user to create a new trip by providing a trip name and a city to search for that will be associated with this new trip.
2. The user should enter the city to search for in the “Search for City” EditText, which should be used to search for a city using the “Search for Cities” api, which is described in the API details on Page 2. The “Search for City” EditText input should

be used as the “input” parameter for the API, please note that the format is City,State for example Charlotte,NC.

3. The API results should be parsed and you are required to retrieve for each place item return the “description” and “place_id”. Display the description for each place in the result using a ListView/RecyclerView as shown in Figure 2(a).
4. Clicking on a list item should select the clicked city and should replace the EditText with the description of the selected item. As shown in Figure 2(c).
5. Upon clicking the “Add Trip” button, you should perform the following:
   a) Retrieve the geo-location of the selected City by using the “Get Geo Coordinates” API provided in the API details provide on Page 2.
   b) When the “Get Geo Coordinates” API is successfully retrieved and the results are parsed, then you should save the selected trip, city information and geo location (lat/lng) in Firebase.
   c) Display the Trips Screen as shown in Figure 2(d).

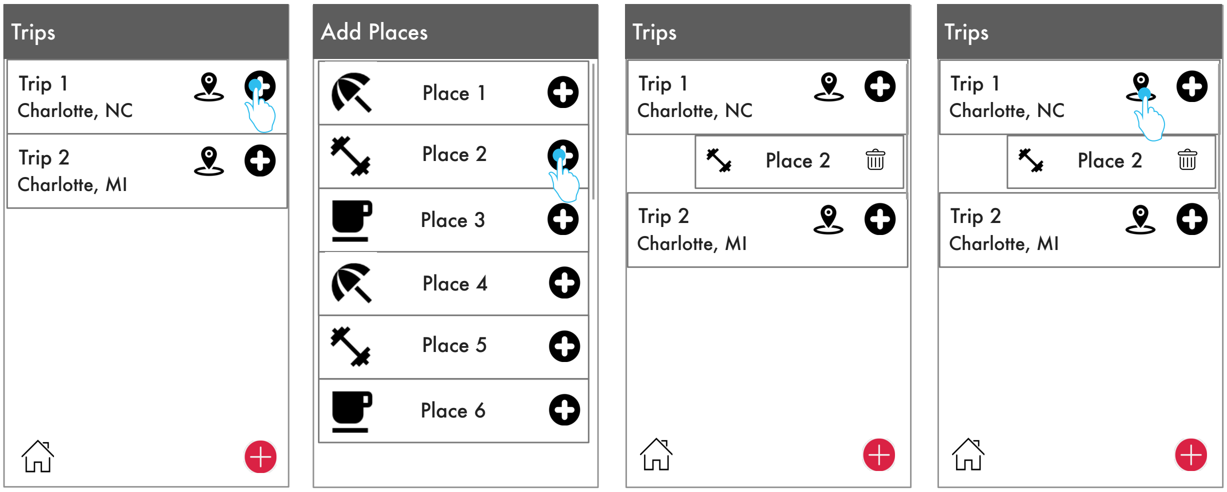


(a) Searching city **(b) Select City from List** **(c) Add Tip to Firebase** **(d) Trips Screen**

Figure 2: Add Trip screen

Part 3 (Add Places Screen, 15 points):

1. This screen will display the list of places which are close to the selected trip’s city location, and will enable the user to select a place to be added to the trip.
2. The list of places close to the trip’s city should be retrieved using the “Get Places Close to Location” API description that is provided on Page 2. Use the lat/lng location for the selected trip’s city when calling the API.
3. Each list item should display the place name and icon. You should also retrieve the latitude and longitude for each place, which is also provided in the JSON response of the API.
4. Clicking on a list item should:
   a) Store the selected place information in Firebase under the selected trip.
   b) Dismiss the current screen and show the Trips Screen. Which should be refreshed to display the updated trip information.



(a) Add places to Trip **(b) Add Places screen** **(c) Added place to Trip** **(d) Click on Map icon**

Figure 3: Trips Screen

Part 4 (Trip Map Screen, 10 points):

1. This screen should show a map that shows markers for all the places included in the specific trip selected in the Trips Screen. The map zooming should be automatically adjusted to display all the markers related to the trip.
2. Clicking on a marker should display the name of the place.

Part #	Points	Description
Part 1	20	Retrieve list of trips and display list of trips and places under trips.
Part 1	10	Delete place from trip and reload the list after delete.
Part 1	5	Add trip, add place, show map icons and their touch events.
Part 2	15	City Search API Call and Parsing, also input validation.
Part 2	10	City Search result display and events handling
Part 2	10	City Get Geo coordinates API call and parsing
Part 2	5	Save trip, city and validation.
Part 3	5	Google place API call and parsing.
Part 3	5	Display the list and user interaction.
Part 3	5	Save place to trip and validation
Part 4	8	Display map, markers and screen resizing to fit all the points on the screen.
Part 4	2	User interaction with markers to display popup when user clicks on a marker.

Grading Key