Unit tests (25 points): Include a file/directory named ‘Testing’ in your Git Repository. There should be details (can be in a separate file in the directory) provided by each team member about the module and the functional testing they have done. Each team member picks a module or module and lists the equivalence classes and the test cases selected to cover all equivalence classes

Expandable Search View Testing

To test my expandable search view, I utilized a black box test by giving it specific inputs as the Yelp Fusion API would have to make sure that data was being fed to my lists properly and were being displayed. These specific inputs were 14 strings for the names of 14 restaurants that would be displayed. I tested some combinations of numbers and letters, upper case and lower case, as well as symbols in each of the 14 list items to make sure that there would be no errors in displaying the strings. After this, to ensure that the Yelp Fusion API worked with my list, I tried searching with multiple inputs such as “Food” or “Indian” or “Mexican” or “Hawaiian.” This resulted in a successful test where I received real results such as “Pono Hawaiian Grill” or “Taqueria Los Pericos” or “Mumbai Delights.” I also tested to see if it was indeed an expandable list. I tested this by loading in the specific dummy strings in multiple child list items in multiple parent lists and pressing the tabs that would expand or hide the child list items. The test was successful and the child list items were expanding and hiding appropriately. Additionally, I tested my filtering button which would allow the user to filter out results based on the string they typed in. To test this, I once again used some strings including numbers and letters as well as symbols to ensure that they were all being filtered out correctly.

Preferences Class Testing

To test my Preferences classes, I checked the values every time the Search button was pressed to make sure that the preferences are shown in the Yelp Search. Since there are 4 different values and a total of 16 different combinations possible. I checked them in order by checking the value which pops up when in testing mode is activated. This is used so that I can check and see if the value is passed through and the write string is used. For example to test the equivalence classes:

1. Open the preferences page and choose the first value to test
2. Press back and go do a search
3. The debugging code will pop up and be used and show what value is passed through
4. Check if correct and repeat for all 16 values.

Google Maps URL Testing

To test the Google Maps URL popup, I checked the given Latitude and Longitude given by the maps url pop up compared to the actual location of the restaurant. In this example, there isn’t a finite amount of inputs so the equivalence classes would be tested by checking different inputs on the Search and comparing a wide variety of terms like: Indian, Italian, etc. as well as changing preferences and choosing from that list. For example to test equivalence classes:

1. Send in a Search for the Yelp API
2. Click on Restaraunt

3. Find restaurant location
4. Click on Google Maps URL provided on that screen
5. Compare the restaurant location vs the Google Maps URL

Maps Testing

Tested the the gps by making sure it pointed to the device's current location and update constantly. Moved to different locations with the map open to test location updating, and spoofed gps locations to test gps itself.

Login/Signup Testing

Tested signups by creating accounts using both real and fake email addresses. Email validity turned out to be irrelevant as long as there was an '@' as there was no email verification. Tested logins by attempting to login and with both existing and non-existing accounts.