EVENTME

*Testing Document*

Alice Han hanalice@usc.edu

Oscar Hong othong@usc.edu

Yoonsoo Nam yoonsoon@usc.edu

CSCI 310 Team 30

*November 21, 2022*

# Contents

[**Contents**](#_546krc2tqt0v) **2**

[**Preface**](#_loj20wo0ofaa) **3**

[**Instruction**](#_7293286obfkg) **4**

[1.1 How to execute test cases](#_entrk7msrixq) 4

[**Black-box Testing**](#_iffyg6tj6bul) **5**

[**White-box Testing**](#_w5edjo2otz1y) **8**

[1.1 Test Cases](#_nm6evjgseimk) 8

[1.2 Coverage](#_hbq3nkup5ktc) 10

# Preface

As the global pandemic winds down and the weather is getting better, people have a need to attend different in-person events. But students and other individuals who may be financially insecure or money-conscious don’t particularly want to spend money on every single event being offered. Hence, EventMe is an application that will help users find free and/or cheap local events and register for them.

# 

# Instruction

### *1.1 How to execute test cases*

Right-click on each test class and click Run.

# Black-box Testing

1. Use without login (Continue as guest)
   1. **Location**: java/com/example/eventmegroup/LogInEspTest.java: test\_a\_guest()
   2. **Description**: It tries to click on continue as guest and see if the app allows the user to continue as guest
   3. **Rationale**: We need to have the user be able to continue as a guest because it was one of the client’s requests
   4. **Bug** **Fixed**: There was no bug related to the test
2. Putting in invalid/wrong password for login
   1. **Location**: java/com/example/eventmegroup/LogInEspTest.java: test\_b\_wrong\_password\_login()
   2. **Description**: It enters the wrong password for the user to see if it will lead the user to the next page
   3. **Rationale**: We cannot let users enter whatever they want and lead them to the next page
   4. **Bug** **Fixed**: There was no bug related to the test
3. Putting in correct log in and navigating to the main activity
   1. **Location**: java/com/example/eventmegroup/LogInEspTest.java: test\_c\_log\_in\_success()
   2. **Description**: It tries to see if the right signin values would lead to the next page (the main activity) for the user to navigate as a logged in user
   3. **Rationale**: The user must be logged in with the correct id and password to navigate to the next page
   4. **Bug** **Fixed**: There was no bug related to the test
4. Changing name in the profile
   1. **Location**: java/com/example/eventmegroup/SetUpEspTest.java: Test\_Changing\_name()
   2. **Description**: The user must be able to change the name
   3. **Rationale**: It is one of the requirements from the client
   4. **Bug** **Fixed**: We could not individually change the name without changing other inputs so we fixed the bug by changing out the logic for changing profile inputs. More specifically, we bulk upload the picture, birthday, and the username at the same time to the backend. Therefore, we could not change the username without changing the profile picture. This was detected during the testing and we were able to fix it by checking whether or not each input has changed to upload it with the old values, if old values existed.
5. Changing birthday in the profile
   1. **Location**: java/com/example/eventmegroup/SetUpEspTest.java: Test\_Changing\_Bday()
   2. **Description**: The user must be able to change the birthday in their profile without changing other inputs
   3. **Rationale**: Client’s requirements
   4. **Bug** **Fixed**: We could not individually change the name without changing other inputs so we fixed the bug by changing out the logic for changing profile inputs. More specifically, we bulk upload the picture, birthday, and the username at the same time to the backend. Therefore, we could not change the username without changing the profile picture. This was detected during the testing and we were able to fix it by checking whether or not each input has changed to upload it with the old values, if old values existed.
6. Clicking the event in the list returned on Explore
   1. **Location**: java/com/example/eventmegroup/ExploreActivityAndroidTestEspresso.java: Test\_ClickEvent()
   2. **Description**: The user should be able to click on the event to see the event details to register.
   3. **Rationale**: Client requirements
   4. **Bug** **Fixed**: Specific event information was not populating
7. Clicking the “All” button after another filter button (i.e. “Shop”)
   1. **Location**: java/com/example/eventmegroup/ExploreActivityAndroidTestEspresso.java: Test\_AllButton()
   2. **Description**: The user should be able see all events after selecting a different filter button beforehand.
   3. **Rationale**: Client requirements, the listview should also be changing.
   4. **Bug** **Fixed**: None
8. Clicking the “Cost” button
   1. **Location**: java/com/example/eventmegroup/ExploreActivityAndroidTestEspresso.java: Test\_CostButton()
   2. **Description**: Should be able to correctly click and as one of the sorting buttons for the Explore function, should return a different ordered list of events per sorting requirement
   3. **Rationale**: Client requirements
   4. **Bug** **Fixed**: None
9. Clicking the “Party” button
   1. **Location**: java/com/example/eventmegroup/ExploreActivityAndroidTestEspresso.java: Test\_PartyButton()
   2. **Description**: Should be able to correctly click one of the filtering buttons for the Explore function, should return a smaller changed list of events per filtering requirement
   3. **Rationale**: Client requirements
   4. **Bug Fixed**: None
10. Clicking the “Proximity” button after clicking another sort button
    1. **Location**: java/com/example/eventmegroup/ExploreActivityAndroidTestEspresso.java: Test\_ProximityButton()
    2. **Description**: Should be able to correctly click and as one of the sorting buttons for the Explore function, should return a different ordered list of events per sorting requirement
    3. **Rationale**: Client requirements
    4. **Bug** **Fixed**: None
11. Clicking the home button in bottom nav bar
    1. **Location**: java/com/example/eventmegroup/NavBarBlackTest.java:Test\_Home()
    2. **Description**: The user clicks the home button on the bottom nav bar to return home
    3. **Rationale**: It was a requirement from the user; used to navigate between pages
    4. **Bug** **Fixed**: There was no bug related to the test
12. Clicking the map button in bottom nav bar
    1. **Location**: java/com/example/eventmegroup/NavBarBlackTest.java:Test\_Map()
    2. **Description**: The user clicks the map button on the bottom nav bar to return to map
    3. **Rationale**: It was a requirement from the user; used to navigate between pages
    4. **Bug** **Fixed**: There was no bug related to the test
13. Clicking the search button in bottom nav bar
    1. **Location**: java/com/example/eventmegroup/NavBarBlackTest.java:Test\_Search()
    2. **Description**: The user clicks the search button on the bottom nav bar to return to search
    3. **Rationale**: It was a requirement from the user; used to navigate between pages
    4. **Bug** **Fixed**: There was no bug related to the test
14. Clicking a pin in the map view
    1. **Location**: java/com/example/eventmegroup/MapBlackTest.java:Test\_Pin\_Click()
    2. **Description**: The user clicks the pin on the Google Map to reveal event data
    3. **Rationale**: It was a requirement from the user; used to see all events on the map
    4. **Bug** **Fixed**: There was no bug related to the test
15. Moving location in map view
    1. **Location**: java/com/example/eventmegroup/MapBlackTest.java:Test\_Pin\_Click()
    2. **Description**: The user clicks the pin on the Google Map to reveal event data
    3. **Rationale**: It was a requirement from the user; used to see all events on the map
    4. **Bug** **Fixed**: There was no bug related to the test

# White-box Testing

### *1.1 Test Cases*

1. Test validation functionality for the login inputs for good email and password format
   1. **Location:** java/com/example/eventmegroup/SignInUnitTest.java: testGoodValidation()
   2. **Description:** This function tests whether the good input format for email and password would return 1 in the check function under the SignInVal class, which would then be used to sign in the user in SignInActivity
   3. **Test Result:** Passed
   4. **Bug Fixed:** None
2. Test validation functionality for the login inputs for bad email format
   1. **Location:** java/com/example/eventmegroup/SignInUnitTest.java: testBadEmail()
   2. **Description:** This tests whether or not the check function will return the correct error code. The function returns 2 for empty email and 5 for malformatted email.
   3. **Test Result:** Passed
   4. **Bug Fixed:** The regex of the email checking had errors. I changed the regex checking string to fix the bug.
3. Test validation functionality for the loging inputs for bad password format
   1. **Location:** java/com/example/eventmegroup/SignInUnitTest.java: testBadPass()
   2. **Description:** This tests whether or not the password failure would indeed lead to correct error codes. In our case, we return 3 for empty password and 4 for passwords less than 7.
   3. **Test Result:** Passed
   4. **Bug Fixed:** None
4. Test event registration checking function under User class for not registered events
   1. **Location:** java/com/example/eventmegroup/EventRegUnitTest.java: testNotRegistered()
   2. **Description:** If the user is not registered for an event, the User class has a registeredEvent method which would return 1 or 2. 1 is returned if the user is not registered for any events and 2 for user not registered for the specified event.
   3. **Test Result:** Passed
   4. **Bug Fixed:** None
5. Test event registration checking function under User class for not registered events
   1. **Location:** java/com/example/eventmegroup/EventRegUnitTest.java: testRegistered()
   2. **Description:** If the user is registered for the event, the method registeredEvent in the User class would return 2.
   3. **Test Result:** Passed
   4. **Bug Fixed:** None
6. Test event arraylist population via Firebase storage for Explore
   1. **Location:** java/com/example/eventmegroup/ExploreUnitTest.java: allEvents\_isPresent()
   2. **Description:** The number of events populated in the list should match the number of events stored in the database
   3. **Test Result:** Passed
   4. **Bug Fixed:** None
7. Test initialization of arraylist for Explore
   1. **Location:** java/com/example/eventmegroup/ExploreUnitTest.java: allEvents\_isInit()
   2. **Description:** There should be no events populated when Explore is initialized
   3. **Test Result:** Passed
   4. **Bug Fixed:** None
8. Test correct initialization of search bar widget
   1. **Location:** java/com/example/eventmegroup/ExploreUnitTest.java: shopEvents\_isCorrect()
   2. **Description:** The search bar should return false when initialized as there are no inputs in the EditTextView
   3. **Test Result:** Passed
   4. **Bug Fixed:** None
9. Test correct sorting by date
   1. **Location:** java/com/example/eventmegroup/ExploreUnitTest.java: dateSort\_isCorrect()
   2. **Description:** As one of the sorting buttons, it should correctly sort events by date
   3. **Test Result:** Passed
   4. **Bug Fixed:** None
10. Test correct filtering of all events
    1. **Location:** java/com/example/eventmegroup/ExploreUnitTest.java: all\_isCorrect()
    2. **Description:** As one of the filtering buttons, it should correctly filter events by type all.
    3. **Test Result:** Passed
    4. **Bug Fixed:** None
11. Test correct / accurate functionality of home button
    1. **Location:** java/com/example/eventmegroup/NavBarWhiteTest.java:Test\_Home\_Button()
    2. **Description:** If the user clicks home button, they should be directed to correct home page, given their sign in status
    3. **Test Result:** N/A
    4. **Bug Fixed:** N/A
12. Test correct / accurate functionality of map button
    1. **Location:** java/com/example/eventmegroup/NavBarWhiteTest.java:Test\_Map\_Button()
    2. **Description:** If the user clicks map button, they should be directed to correct page
    3. **Test Result:** N/A
    4. **Bug Fixed:** N/A
13. Test correct / accurate functionality of search button
    1. **Location:** java/com/example/eventmegroup/NavBarWhiteTest.java:Test\_Search\_Button()
    2. **Description:** If the user clicks search button, they should be directed to correct search page
    3. **Test Result:** N/A
    4. **Bug Fixed:** N/A
14. Test correct / accurate functionality of map pin details on click
    1. **Location:** java/com/example/eventmegroup/MapWhiteTest.java:Test\_Pin\_Register()
    2. **Description:** If the user clicks Google Maps pin, then the correct corresponding details shows up
    3. **Test Result:** N/A
    4. **Bug Fixed:** N/A
15. Test correct / accurate functionality of map event details when pin pop up is clicked
    1. **Location:** java/com/example/eventmegroup/MapWhiteTest.java:Test\_Pin\_Register()
    2. **Description:** If the user clicks Google Maps pin AND then the correct corresponding details shows up, then the correct event register page shows up
    3. **Test Result:** N/A
    4. **Bug Fixed:** N/A

### *1.2 Coverage*

For User functions, all functions were covered through white-box testing. Every function related to sign-in were able to be covered, but sign-up activity was not completely covered via white-box testing as a majority of the test cases were focused on sign-in. Most functions in Explore were covered via testing as the multiple filtering and sorting functions relied on similar syntax and logic, so it was able to be tested by selecting a few representative functions to test the filtering and sorting capabilities. Map coverage is pending due to some delay, but most of its functionality was able to be tested via black-box testing. In the end, we weren’t able to cover components that heavily relied on the UI via white-box testing (i.e. changing the profile picture). Taking all factors into account, our coverage level ended up being around 78%.