System and Unit Test Report
Public Defender
6 angry men
6/6/2017

System Tests:

From Sprint 1 user story:

As a user I want to be able to create an account so that I can use this service

Scenario:

1. Start Public Defender App; select 'Sign In'
    - Follow login prompt using a google account for credentials
    - Click 'Next'
      App will now allow for feature access

From Sprint 2 user stories:

As a user I want to be able to record my interactions with police (authority) officers to hold them accountable

1. Start Public Defender App and sign in as above is no cached sign in is available
2. Press Record
3. Say something

As a user I want a button to signal the end of my interaction with the police to notify my community that this interaction is complete and they no longer have to participate

1. Press "Stop Recording"
2. Navigate to Current Events
   Your event will not be listed anymore

From Sprint 3 user stories

As a user I want to be aware of interactions with police in my community, in order to participate in the gist of the app

1. Select 'Current Events'
   A List of events that are broadcasting will appear

As a user I want to be able to see incidents projected onto a geographical map so that I can understand where incidents occur

1. Select 'Current Events'
2. Select an event from the list
   A map activity will appear showing the event selected as well as your location

Unit Tests:
To run our Android Unit Tests:

1. Run the configuration: “PD_with_tests”
2. Tests will execute automatically as the app compiles
3. The code is kept in PDUnitTests.java

These tests:

- Verify file creation
- Verify the correct header data has been generated during the conversion from the PCM file format to the WAV file format.

Backend Server Tests:
To run these tests you need a REST client that can read .http files or they need to be recreated using curl or similar request tool.

There are:

- get_all.http - tests all the database fields/relationships are working as well as can be used for other debugging purposes.
- get_nearby.http - Tests the ‘current events’ section of the app where you can see where other events are occurring.
- get_user_events.http - tests the association of events and users.
- get_user.http - tests user database (gets single user)
- get_users.http - tests user database (gets all users)
- stream_test.http - tests listening to other events streams (didn’t end being implemented on app).