Test Document

After a series of actions, we have considered the following (system level actions with input and output) a series of steps that a user would follow to determine that each user story has been completed.

1. In order to determined if a user would like a table that lists how much their friends or they have owed, the user would sign in to their account and redirect themselves to "View Log". In "Sign In" >"View Log", the user is able to view their list of friends that they or their friends owe.

2. Another user story that has been implemented would be the main page, that sees the overall functionality of the application. When downloading the "Driving_Tracker.apk", the user would be able to open up the project and be directed to the main page; where there they can set up an account or sign in if they have an existing account.

3. After testing these user stories, we have also fulfilled another functionality where a user may be able to create an account and store their information on a database. If the user were to click the button "Create Account", then they would be redirected to the "Create Account" page where then they would choose a username and password to hold on our database.

4. Another test that can determine the success of our user story would be having a running database that allows the update information and puling of information from our application.To test this, a user would contact the Driving Tracker admin and they would have access to the user's information which is held on Parse.

5. With all these test, we still had more user stories embedded in our application. For the user to test another user story, "would like to link the UI and database", they would test it in the following way. First, they would create an account, this would push the data up on to the database. Then they would be redirected to "Sign In" page and now they sign in where they would pull their username from the database. If successful they would be sign in on only their account.

6. The following user story would be testable if the user were to sign in to their account: have a nice and easy to use layout to all the android pages. On our

application we believe simplicity helps with the user friendliness; therefore using the application should be self explanatory.

7. The next user story that would be testable would be the list of events that a user has been invited to. Also the user would be able to accept or decline that specific event. To test this, the user would first create an account or sign in if their account exist. Then in their account page, the user would click, “View Events”, where they would be shown the events that they were invited to and be able to accept or decline those events.

8. Following these user stories, the next user story to test would be the ability to create an event. To do this, the user would be logged in and redirect themselves to “Create an Event” page. There they would add their event by invited their friends and inserting the times they drove.

9. The last user story, the ability to change a user’s own password, is able to be tested when the user is logged in. Then they redirect themselves to the “Change password” page, and there they would be able to rewrite their password and sign in using their new password.

As we developed our Android Studio project over the following months, these are a series of tests that came up during testing:

1. **Create Account**, when a user deletes an account on parse, the new user can’t create an account under that deleted name, unless they log in someone else’s account, sign out, and go create that account name.
2. **Create Account**, click on password, then click back to main menu, then click back on create account, it starts on password instead of username.
3. C**reate Event**, anything but integers will crash the app.
4. **Create Event**, the user must not be able to update until the other user accepts the invitation.
5. **Create Event**, can add yourself on passenger.
6. **View Events**, needs a toast when user accepts the event.
7. **Longer toasts** are requested if the text is longer.
8. **Create Account**, Log in to an account, then sign out. Go to create account and then create that new account. Then if you click back, you can end up on the first logged in account.

9. **Change Password**, needs confirmation just in case a user types really fast and mistypes their specified password.
10. **Change Password**, a check to make sure password is actually entered and not left blank.
11. **SQL Injection** on any inserted text field.
12. **Create Event**, when creating an event, if the user inserts the passenger twice the application crashes.
13. **Create Event,** when creating an event, the update time does not delete the event created if the user clicks back to create event.

The following are a list of known problems/issues that remain to be fixed:

1. **Add Friend,** when adding a friend, the post to Parse database is not consistent and sometimes our application crashes.

2. **Account Page,** if for whatever reason the application crashes, the application refreshes and displays the account page, however the user is not signed in to any account; therefore they must sign out and sign back in.