KASPER – ID3 PROJECT DOCUMENTATION

SOFTWARE DEVELOPMENT TEAM:

Project Manager: Tushita Patel

Dev Lead: Kristof Mercier, Dylan Prefontaine

Test Lead: Jeremy Liau

Build Manager: Christopher Mykota-Reid (ChrisMR)

Developers: Gaurav Arora, Haotian (Justin) Ma, Melody (Tian) Zhao

Test Team: Christopher May (ChrisJ), Ryan Tetland

Documentation: Arianne Butler

Contents

1.0 Requirements Document ID3 3
1.1 Front-end Requirements 3
1.2 Back-end Requirements 3
1.2 Mini Milestones for ID3 5
2.0 Time Estimations 6
3.0 Design 7
3.1 API Document 7
3.2 Updated Data Structures ID3 8
3.3 Development Unit Tests 9
4.0 Testing Document 9
4.1 Test Plan 9
4.2 Test Report 11
4.3 Defect Report 12
4.4 State Transition Diagrams 16
4.5 Critical Use Cases 18
5.0 Coding Style Guide 21
6.0 Build Report 21
7.0 Upcoming Requirements 23
7.1 ID4 Requirements 23
7.2 Future Requirements 23
8.0 Triage Meeting ID3 24

1.0 Requirements Document ID3

1.1 Front-end Requirements

Priority (Not Finished in ID2)

- Receive notifications regarding Favourites
  - Price changes
  - Listing removed/edited
- Seller Upload images

Front-end Requirements Changes

This section highlights the changes to our UI since ID2. These changes were discussed and agreed upon during both the current and previous ID's, and have been implemented during ID3.

Finalized changes to the UI for ID3 can be found at the following link:

> https://github.com/CMPT371Team1/Documentation/blob/master/ID3-Documentation/Other/ID3-UI-Changes.pdf

1.2 Back-end Requirements

Priority (Not Finished in ID2)

- Email verification (not done)
- Forgot password (not done)
- Change password
- Get all Listings for Browse page (removed)
- Get filtered Listings
- Edit Listings
- Like/Dislike
- Edit Account
- Get Favourites Listings

Back-end Requirements

System Design:
The back-end system implementation is separated into two main modules – User accounts and Listings information. The account module handles user Sign-in, Sign-out, Sign-up, email verification, forgotten passwords, and resetting passwords. The Listings module defines a set of data related to a listed property, such as its location, price, description, images, etc. It includes creating new Listings, getting filtered Listings, and edit existing Listings. To start the back-end, the http server is initialized, which calls all system modules before serving user requests. Thus, it has complete control over all parts of the system, and can decide to close any aspect should an issue arise.

System Requirements:

Fundamental aspects of the back-end behaviour can be defined by the following set of requirements:

Functional Requirements:

1. The back-end must gather data sent from devices and store it in the database for future reference.
2. User requests must be handled appropriately, and relevant information stored in the database must be sent to the device interface for display.
3. The system should be capable of recovering from failures and crashes whilst maintaining the integrity of any stored data.

Non-functional Requirements:

1. The back-end system should be responsive to user requests, so that delays in displaying data are minimized.
2. Data integrity and error correction mechanisms should be implemented so that no erroneous data is stored in the database.
3. The system should send informative error messages to the client about the source of error.
4. The system should provide an appropriate debugging environment, in which new code can be easily integrated, tested, and checked for errors.

Software:

The back-end system is implemented in Python and uses several external sources for specific implementations:

1. Google App Engine
2. NoSQL
3. Google Datastore NDB Client Library
4. Webapp2: a lightweight Python web framework

Back-end Requirements Changes

This section highlights the changes to our back-end since ID2. These changes were discussed and agreed upon during both the current and previous ID's, and have been implemented during ID3.

- Remove cursor and lastListing from the getListings API call
- Allow optional parameters to be passed to the getListings API call
  - Filter data structure
    - If the filter is not given, a default filter will be applied
  - "included fields"
    - Tells the API which fields of the selected listings should be returned.
    - Include a "featureImage" option which will return the single featured image as opposed to all of the images
  - "limit"
    - The number of listings to be returned
    - By default, only returns id's

- EditListing call must update modified date and return it in the response JSON
- signInWithToken and signIn need to update the "lastSeen" field in the user table.
- The server must be updated if a seller's listing is modified on the device, and the device must be updated when a favourited Listing is changed on the server. Check for validity must be done on front and back-end. If the back-end is passed an invalid listing, it should not be accepted, and the device's Listing entry should be reverted by force. This will only happen with API misuse, so this behaviour is acceptable.
- A user specific rate limit should be enforced to ensure the API is being used correctly, and to defend against brute force password cracking. There is a default overall rate limit, but we should look into implementing a user specific one as well.

- Add a boolean “verified” field in the user table to indicate if the user has verified their email. This must be updated when they change their email, or when they click an activation link.
- "signInWithToken" should return everything that "signIn" returns.

1.2 Mini Milestones for ID3

Development:

- Finish all new requirements for front and back-end on Trello boards
- Have the server up on a USASK machine ✓
- Have the front-end communicating with the back-end (server)
    - This has been achieved with Sign in and Sign up only ✓
- Make two test accounts, and dummy Listings ✓

Testing:

- Incorporate expect statements into End-to-End tests ✓
- Resolve "similar buttonText" element ID issue ✓
- Show test team how to test on Android device ✓
- Hold bug-fixing sessions with developers ✓
- Test plan for future ID's ✓
- Manual testing on Android device and Browser ✓
- Separate Android, Browser, and Bug-Party bugs into categories ✓
- Update End-to-End tests, test matrix, defect report ✓
- Merge all End-to-End tests together, run all tests in parallel in Firefox ✓
- Use case and flow diagram update ✓

Build:

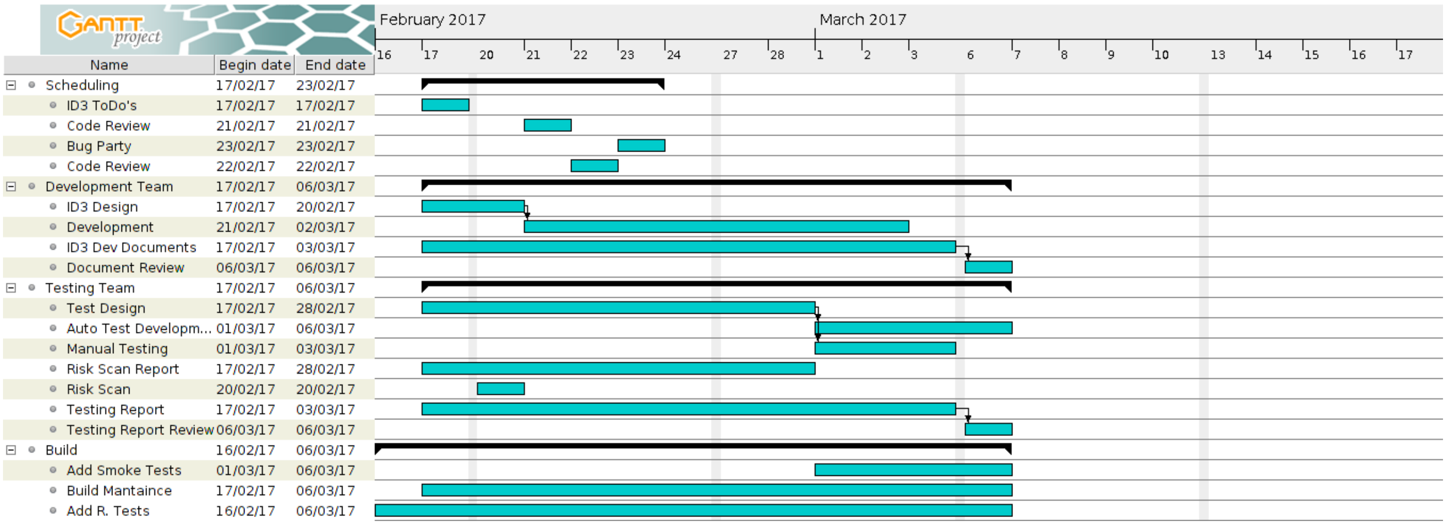
- Implement front-end smoke tests ✓
- Implement back-end smoke tests ✓
- Set up multi-stage build

Documentation:

- Organize, compile, and edit all documentation ✓

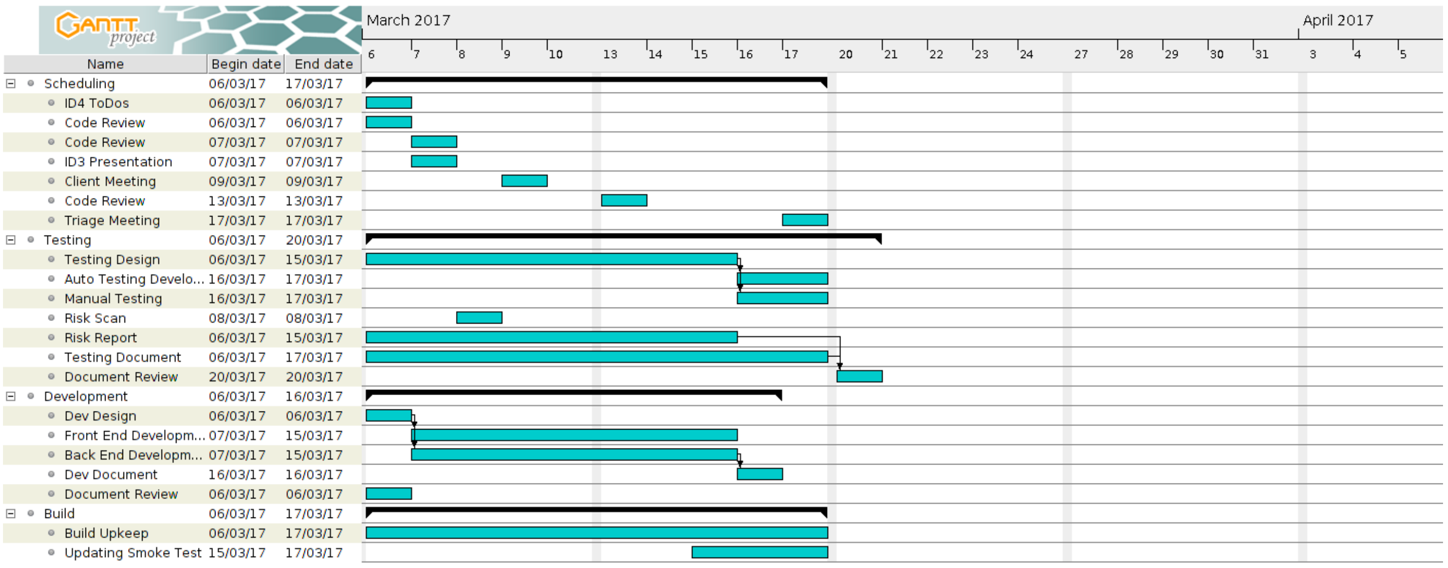
2.0 Time Estimations

Gantt Diagram ID3:



Name	Begin date	End date
Scheduling	17/02/17	23/02/17
ID3 ToDo's	17/02/17	17/02/17
Code Review	21/02/17	21/02/17
Bug Party	23/02/17	23/02/17
Code Review	22/02/17	22/02/17
Development Team	17/02/17	06/03/17
ID3 Design	17/02/17	20/02/17
Development	21/02/17	02/03/17
ID3 Dev Documents	17/02/17	03/03/17
Document Review	06/03/17	06/03/17
Testing Team	17/02/17	06/03/17
Test Design	17/02/17	28/02/17
Auto Test Developm...	01/03/17	06/03/17
Manual Testing	01/03/17	03/03/17
Risk Scan Report	17/02/17	28/02/17
Risk Scan	20/02/17	20/02/17
Testing Report	17/02/17	03/03/17
Testing Report Review	06/03/17	06/03/17
Build	16/02/17	06/03/17
Add Smoke Tests	01/03/17	06/03/17
Build Mantaince	17/02/17	06/03/17
Add R. Tests	16/02/17	06/03/17

Gantt Diagram ID4:

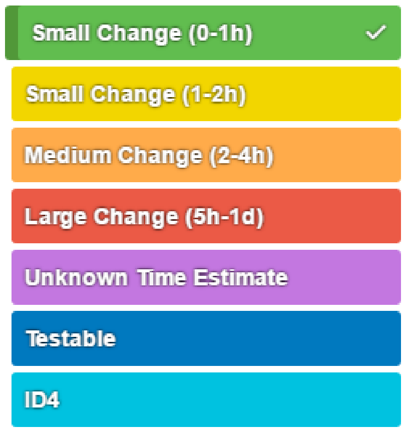


Name	Begin date	End date
Scheduling	06/03/17	17/03/17
ID4 ToDos	06/03/17	06/03/17
Code Review	06/03/17	06/03/17
Code Review	07/03/17	07/03/17
ID3 Presentation	07/03/17	07/03/17
Client Meeting	09/03/17	09/03/17
Code Review	13/03/17	13/03/17
Triage Meeting	17/03/17	17/03/17
Testing	06/03/17	20/03/17
Testing Design	06/03/17	15/03/17
Auto Testing Develo...	16/03/17	17/03/17
Manual Testing	16/03/17	17/03/17
Risk Scan	08/03/17	08/03/17
Risk Report	06/03/17	15/03/17
Testing Document	06/03/17	17/03/17
Document Review	20/03/17	20/03/17
Development	06/03/17	16/03/17
Dev Design	06/03/17	06/03/17
Front End Developm...	07/03/17	15/03/17
Back End Developm...	07/03/17	15/03/17
Dev Document	16/03/17	16/03/17
Document Review	06/03/17	06/03/17
Build	06/03/17	17/03/17
Build Upkeep	06/03/17	17/03/17
Updating Smoke Test	15/03/17	17/03/17

To see time estimations on all individual tasks, please follow the link in Process Documentation, section 2.0 Activity Log, and click on the bottom tab labelled "Individual Activity Log".

For ID3, development time estimations have been added for each assigned task on Trello. Trello cards now contain a coloured label indicating the approximate estimation of each task. These estimates were approximated by the dev team.

The following image shows the colour coded time estimation scale with which each trello card will be rated.



3.0 Design

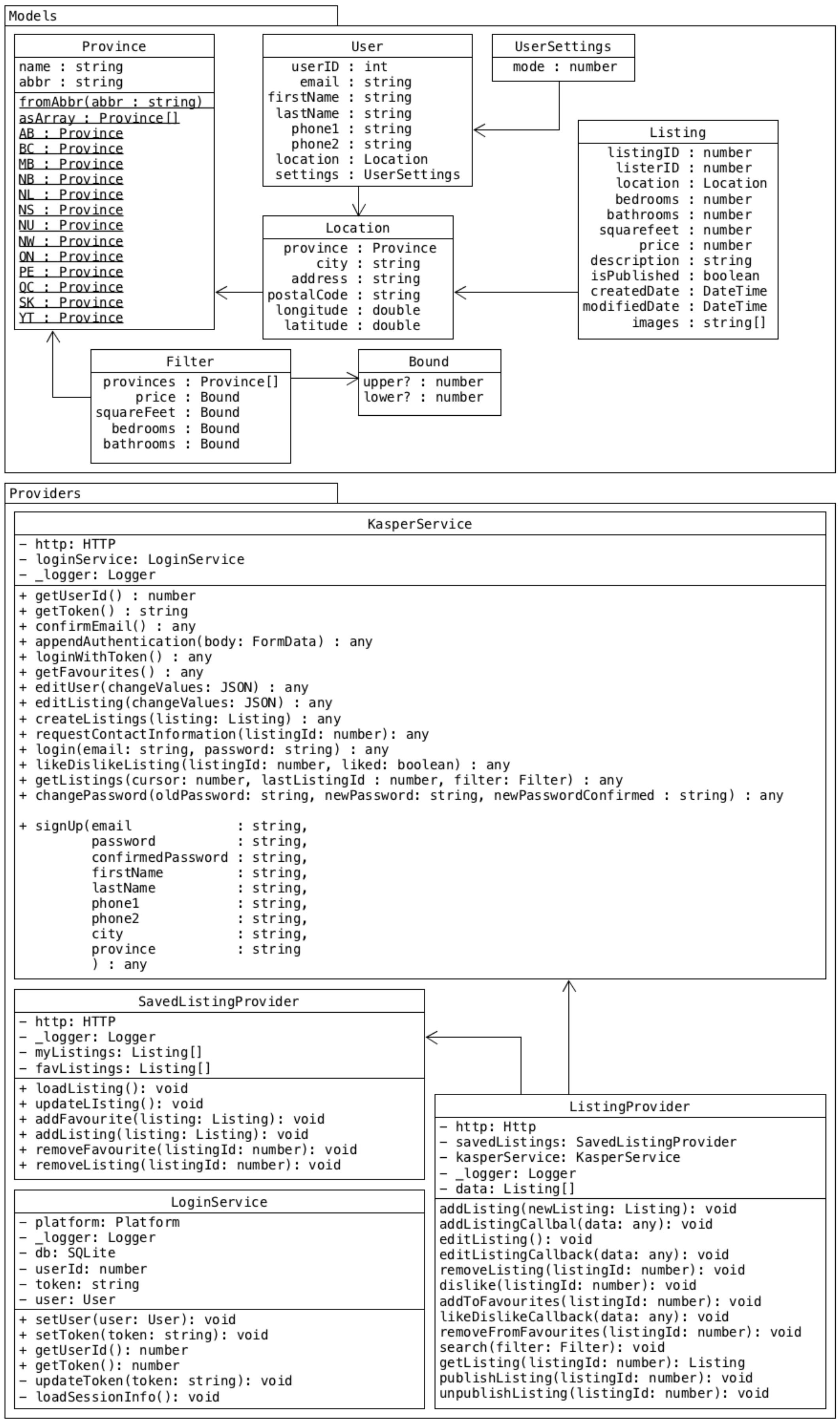
3.1 API Document

The API document outlines the client server communication of our system. Our project will contain sixteen API calls (calls to the server), eleven of which are currently implemented, and two of which are in full communication with the server. These calls send a JSON body with relevant data. The server receives, parses, and processes the JSON using the database. When finished, it will reply to the sender with a token containing the reply status and a JSON body containing the requested information. The following link contains our updated ID3 API document and details on the various calls to our database.

Note: We are currently experiencing difficulty with this link. Please copy and paste it into your web browser.

https://docs.google.com/document/d/1N4jt1_PgxPhXwdc1TcT7TBjFNOZqYO5L10ha3bpO5M8/edit#heading=h.1skatsa28we

3.2 Updated Data Structures ID3



3.3 Development Unit Tests

The procedure to be followed by all developers for writing both front and back-end unit tests respectively, can be found at the following links:

Front-end Guide for Unit Testing:

https://github.com/CMPT371Team1/Project/wiki/Coding-Style-Example-(JavaScript)

Back-end Guide for Unit Testing:

https://github.com/CMPT371Team1/Project/wiki/Coding-Style-Guide-(Python)

4.0 Testing Document

4.1 Test Plan

4.1.1 Test Strategy

Critical Use-Cases ID3:

- Register New User, Filter Listings, Add Listing

Test Matrix:

- Improving our test matrix was one of the test teams top priorities for ID3. A better description was written to explain the challenges the test team faced during both manual and automatic testing of the application.

Manual Testing (Android and Browser):

- The test team prioritized testing on mobile devices for ID3 over browser testing. The application is expected to be more commonly used on a mobile device. The Browser is currently limited to end-to-end tests.

End-to-end testing:

- End-to-end tests were broken during ID3 due to the front-end requirements changes. Ryan and Chris May worked on updating tests to work correctly with the changes.

4.1.2 Training Requirements

- Testers learned to run the application on their Android device for testing. Android SDK was downloaded and the application was created using the “ionic run” option, rather than using “ionic serve” to run on the browser.
- Testers continued to practice with Protractor, learning to select elements that we were previously unable to select due to lack of experience. Testers also cleaned up the test code for readability.

4.1.3 Documentation

Test Matrix:

The X-axis represents the functionality of the app, while the Y-axis represents the tests. The green boxes represent the areas where the tests are accessing and testing functions. The red boxes are the unimplemented/incomplete functions of the app, which will be added to the regular testing schedule once they are complete. The Favourites, Favourites Filter and Browse Listings Filter functionalities are mostly untested, due to the application using fake, hardcoded data. Once the developers have created real Listings in the database, these will become a critical focus of testing. Facebook Sign-In, Publish/Save Listings are currently unimplemented.

	App Sign-In	Facebook Sign-In	Sign-Up	Browse Listings Filter	Publish New Listing	Save W/O Publishing Listing	My Profile	Browse	Favourites	Favourites Filter	My Listings	Change Password
Sign In with existing user	🟩	🟩	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜
Add a Listing (Signed In)	🟩	🟩	⬜	⬜	🟩	🟩	⬜	⬜	⬜	⬜	🟩	⬜
Registering a new user	⬜	⬜	🟩	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜
Browse Listings (Signed In)	🟩	🟩	⬜	🟩	⬜	⬜	⬜	🟩	⬜	⬜	⬜	⬜
Edit account information	🟩	🟩	⬜	⬜	⬜	⬜	🟩	⬜	⬜	⬜	⬜	🟩
Register an existing user	⬜	⬜	🟩	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜
Invalid Sign-In credentials	🟩	🟩	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜
Change account password	🟩	🟩	⬜	⬜	⬜	⬜	🟩	⬜	⬜	⬜	⬜	🟩
Like/Dislike a listing	🟩	🟩	⬜	🟩	⬜	⬜	⬜	🟩	🟩	⬜	⬜	⬜
Weak password given	⬜	⬜	🟩	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	🟩

Navigation while signed In	⬜	⬜	⬜	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Navigation while not signed in	🟩	🟩	🟩	🟩	⬜	⬜	⬜	🟩	⬜	⬜	⬜	⬜
Edit Favourites	🟩	🟩	⬜	⬜	⬜	⬜	⬜	⬜	🟩	🟩	⬜	⬜

4.1.4 Schedule

ID1: Testing was mostly done manually on the front-end of the application. The test team began creation of automated tests using Protractor. Tests and functionality were recorded in the test matrix, and all bugs were listed in detail in the defect report.

ID2: The Test Team attempted to integrate automated end-to-end tests with TravisCI, updated the defect report, updated the flow charts to better reflect the new requirements/current state of the system, and re-created, simplified, and updated the test matrix.

ID3: The first bug party was held. The test team continued to update the test matrix, defect report, and flow charts. Testing on Android devices was successful, and brought new, device specific, defects to the defect report. The test matrix was updated and made more thorough by including a better description of the state of our tests. End-to-end tests had to be updated again, due to the front-end requirements changes and other changes. The flow charts have been significantly updated to show the access to new functionality, as well as areas of the program not previously implemented. Protractor was successfully integrated with TravisCI, which was an unresolved issue from ID2.

ID4 and Future ID's:

- Continue updates on the test matrix, defect report, and flowcharts
- Favourites screen, Like/Dislike functionality, Browsing/Favourites filter testing (once developers have added real Listings to the database)
- Create better end-to-end tests with more thorough testing. Our tests have evolved slowly due to requirements changes in the last two ID's.
- Hold Bug Party 2 (closer to the end of ID4).

4.2 Test Report

The following summarizes various observations and measures taken by the Test Team in ID3:

- Once the front and back-end of the application were connected, the test team found many more defects
- The test matrix greatly increased in size as more functionality was added, with respect to the ID2 test matrix which was only applicable to the front-end
- Protractor was integrated with TravisCI by working with build manager Chris Mykota-Reid
- Manual testing on Android devices was prioritized over testing with Chrome/Firefox browsers

4.3 Defect Report

4.3.1 Bugs found after Bug Party

Bug:	Like/Dislike on “Browse” page breaks cycling of Listings on details page
Reproducibility:	100%
Steps taken:	I. Like/Dislike any listing by swiping on the “Browse” page II. View detailed information of any listing, and attempt to cycle listings with arrow buttons
Variations:	None
Resolved:	No
Screenshot:	None

Bug:	Slide bars change size when switching from numbers to “No max” or “No min”
Reproducibility:	100%
Steps taken:	I. Navigate to filter screen and move slide bars to boundaries
Variations:	None
Resolved:	No
Screenshot:	None

Bug:	Pop-up saying a list was liked or disliked pops up multiple times per swipe
Reproducibility:	100%
Steps taken:	II. Like/Dislike a listing on Browse page by swiping, either direction
Variations:	Swipe left for reproducing dislike pop-up Swipe right for reproducing like pop-up
Resolved:	No
Screenshot:	None

Bug:	While signed in to any account, being on the “My Profile” tab and selecting “My Profile” tab again will show the sign-in screen.
Reproducibility:	100%
Steps taken:	I. Sign in on any account II. Navigate to “My Profile” tab III. Select “My Profile” tab
Variations:	None
Resolved:	No
Screenshot:	None

Bug:	Liking/Disliking a listing on the Listing’s detailed info page doesn’t remove it from the list
Reproducibility:	100%
Steps taken:	I. Navigate to detailed info page of any listing II. Like/Dislike the listing
Variations:	Liking/Disliking on the “Browse” page correctly removes the listing.
Resolved:	No
Screenshot:	None

Bug:	Selecting a listing on the “Browse” page allows the user to navigate to the “My Listings tab” without being required to sign in first
Reproducibility:	100%
Steps taken:	I. Select any listing on the “Browse” page to navigate to the detailed info page II. Navigate to “My Listings” tab
Variations:	None
Resolved:	No
Screenshot:	None

4.3.2 Bugs found during Bug Party 1 (Feb 22nd)

Bug:	No error prompt when login information is incorrect
Reproducibility:	100%
Steps taken:	I. Navigate to "Sign In" page II. Attempt to sign in with invalid credentials
Variations:	None
Resolved:	No
Screenshot:	None

Bug:	Provinces in My Profile not properly aligned
Reproducibility:	100%
Steps taken:	I. Navigate to My Profile tab while signed in
Variations:	None
Resolved:	No
Screenshot:	None

Bug:	[Android] While registering new user, input fields scroll out of view while typing
Reproducibility:	100%
Steps taken:	I. Select "Register Here" on "My Profile" tab II. Select on any input field
Variations:	None
Resolved:	No
Screenshot:	None

Bug:	[Android] In "My Listings", Add Listing button is incorrectly on top right corner instead of top left
Reproducibility:	100%
Steps taken:	I. Select "Register Here" on "My Profile" tab II. Select on any input field
Variations:	None
Resolved:	No
Screenshot:	None

Bug:	[Android] On "Add Listing" page, selected Province(s) don't appear when selected
Reproducibility:	100%
Steps taken:	I. Navigate to "Add Listing" page II. Select Province list III. Add any province
Variations:	None
Resolved:	Yes
Screenshot:	None

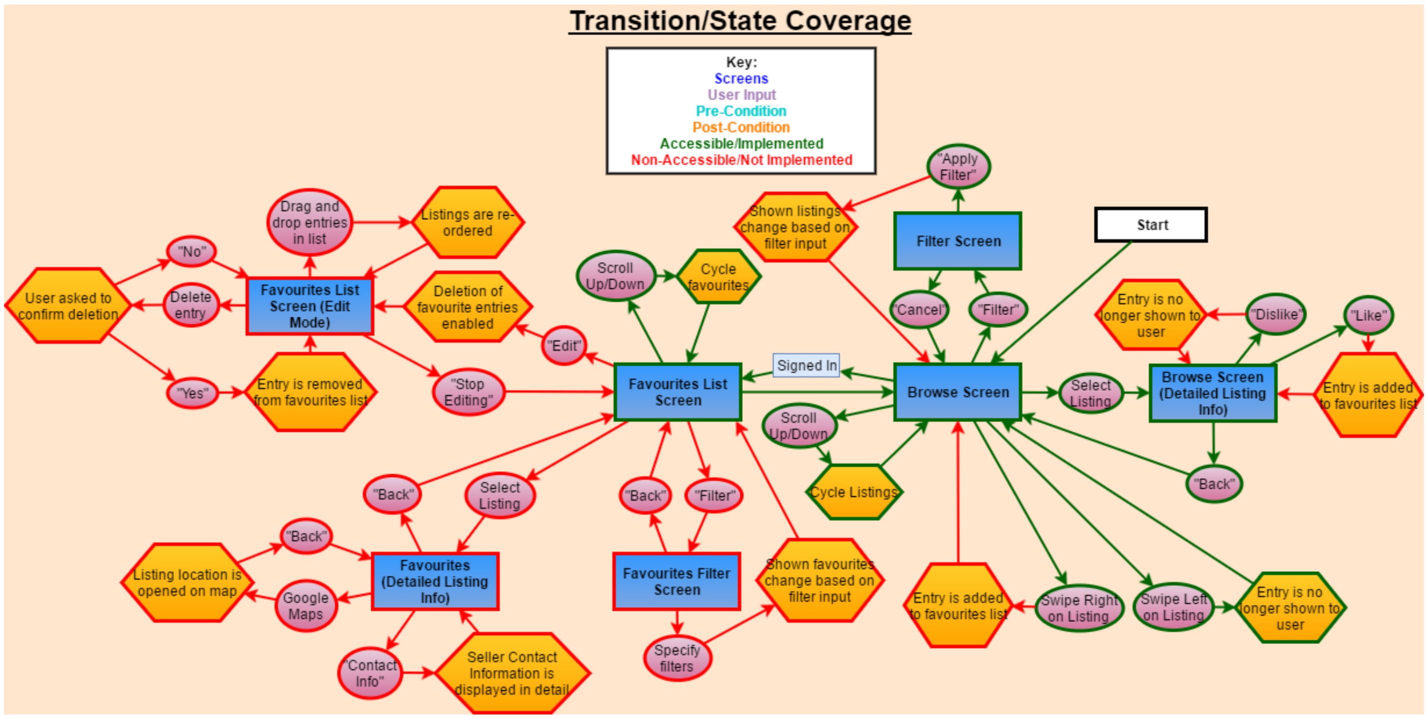
Bug:	Cannot cancel registration
Reproducibility:	100%
Steps taken:	I. Select "Register Here" on "My Profile" tab II. Attempt to go back to "Sign In" page
Variations:	None
Resolved:	Yes
Screenshot:	None

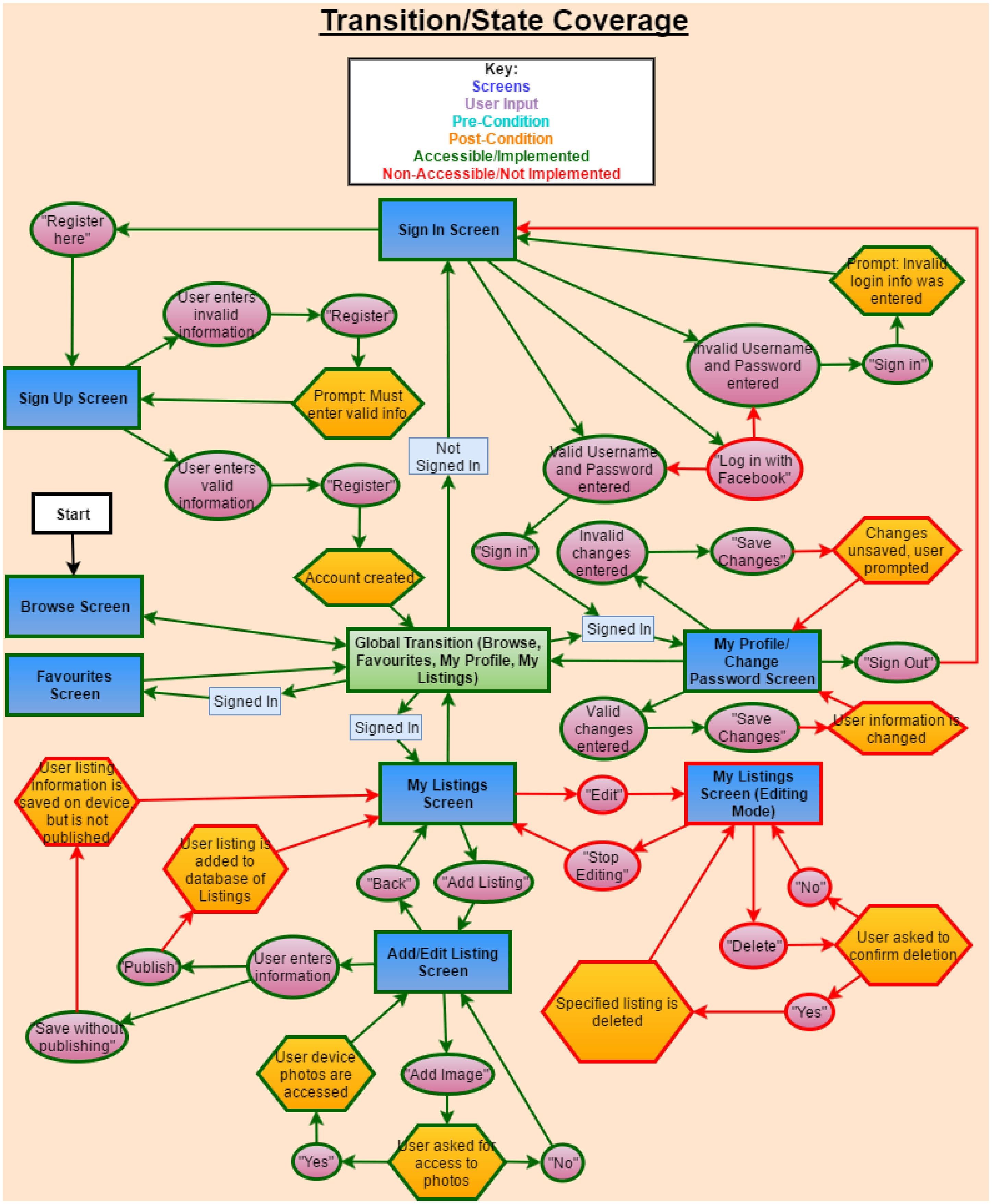
Bug:	If E-mail is incorrect format on registration page, it still proceeds
Reproducibility:	100%
Steps taken:	I. Enter an email of incorrect format in e-mail field
Variations:	None
Resolved:	Yes
Screenshot:	None

Bug:	Registration requires last name
Reproducibility:	100%
Steps taken:	I. Attempt to register a new account II. Attempt to proceed without entering a last name
Variations:	None
Resolved:	Yes
Screenshot:	None

Bug:	If password != confirmed password in registration page, it still proceeds
Reproducibility:	100%
Steps taken:	I. Attempt to register a new account II. Enter different passwords in "Password" and "Confirmed Password" fields
Variations:	None
Resolved:	Yes
Screenshot:	None

4.4 State Transition Diagrams


Transition/State Coverage
Key:
Screens
User Input
Pre-Condition
Post-Condition
Accessible/Implemented
Non-Accessible/Not Implemented
Start
Signed In
Browse Screen
Filter Screen
"Apply Filter"
"Cancel"
"Filter"
Shown listings change based on filter input
Select Listing
Browse Screen (Detailed Listing Info)
"Dislike"
"Like"
"Back"
Entry is no longer shown to user
Entry is added to favourites list
Scroll Up/Down
Cycle Listings
Swipe Right on Listing
Swipe Left on Listing
Favourites List Screen
Cycle favourites
"Edit"
"Stop Editing"
Favourites List Screen (Edit Mode)
Drag and drop entries in list
Listings are re-ordered
Deletion of favourite entries enabled
Delete entry
User asked to confirm deletion
"No"
"Yes"
Entry is removed from favourites list
"Back"
"Filter"
Favourites Filter Screen
Specify filters
Shown favourites change based on filter input
Favourites (Detailed Listing Info)
Google Maps
Listing location is opened on map
"Contact Info"
Seller Contact Information is displayed in detail


Transition/State Coverage
Key:
Screens
User Input
Pre-Condition
Post-Condition
Accessible/Implemented
Non-Accessible/Not Implemented
Sign In Screen
"Register here"
Sign Up Screen
User enters invalid information
"Register"
Prompt: Must enter valid info
User enters valid information
"Register"
Account created
Start
Browse Screen
Favourites Screen
Global Transition (Browse, Favourites, My Profile, My Listings)
Not Signed In
Signed In
Signed In
Signed In
Prompt: Invalid login info was entered
Invalid Username and Password entered
"Sign in"
Valid Username and Password entered
"Log in with Facebook"
"Sign in"
Invalid changes entered
"Save Changes"
Changes unsaved, user prompted
My Profile/ Change Password Screen
"Sign Out"
Valid changes entered
"Save Changes"
User information is changed
My Listings Screen
"Edit"
My Listings Screen (Editing Mode)
"Stop Editing"
"No"
"Delete"
User asked to confirm deletion
"Yes"
Specified listing is deleted
User listing information is saved on device but is not published
User listing is added to database of Listings
"Back"
"Add Listing"
Add/Edit Listing Screen
Publish
User enters information
"Save without publishing"
User device photos are accessed
"Add Image"
User asked for access to photos
"Yes"
"No"

4.5 Critical Use Cases

The following critical use cases outline the low level details of the most important features of our application:

Use Case Component	**Description**
Use Case Name	ID3: Attempt to register a new user
Primary Actor	New User
Pre-conditions	At entry point of app, on main browse screen
Success End Conditions	Either the new user is added to the database or it is identified that the e-mail is already in the database
Failed End Conditions	User is not registered
Main Success Scenario A: Actor S: System	I. A: Selects “My Profile” tab II. A: Selects “Register Here” button III. A: Inputs “testEmail” in email field IV. A: Inputs “Password123” in password field V. A: Inputs “Password” in confirm password field VI. A: Selects “Next” button VII. S: Moves to next info screen VIII. A: Inputs “John” in first name field IX. A: Selects “Smith” in last name field X. A: Selects “3065551234” in phone number field XI. A: Selects “Next” button XII. S: Moves to next info screen XIII. A: Selects "Province" drop down list and selects "Alberta" XIV. A: Selects "OK" button XV. A: Inputs “Edmonton” into city field XVI. A: Clicks "finish" button XVII. S: Adds the new user to database if it is not already
Extensions	IV. A: Inputs "weakpassword" into password field VI. A: Inputs weakpassword
Priority	Critical
Response Time	< 1 second
Notes	The user is will only be registered if the specific test email has not already been used. If it has been used, it will be tested in the next part of the tests to make sure it can be logged into.

Use Case Component	**Description**
Use Case Name	ID3: Filter Listings
Primary Actor	Browser (Can be signed in, or not signed in)
Pre-conditions	On “Browse” page
Success End Conditions	Listings on “Browse” page are filtered to specifications provided
Failed End Conditions	Listings on “Browse” page are not filtered to specifications provided
Main Success Scenario A: Actor S: System	I. A: Selects “Filter” button II. A: Selects “Provinces” drop down III. A: Selects “Alberta” and “Saskatchewan” and selects the “OK” button IV. A: Drags the “Price” slider between “100 000 and 1 600 000” V. A: Drags the “Square feet” slider between “100 and 10 000” VI. A: Drags the “Bedrooms” slider between “2 and 8” VII. A: Drags the “Bathroms” slider between “3 and 7” VIII. A: Selects the “Apply Filter” button IX. S: Displays the listings from the database within the filter criteria
Extensions	I. - VIII. A: Selects Cancel Button I. – VIII. A: Clicks outside of Filter pop up
Priority	Critical
Response Time	< 1 second
Notes	

Use Case Component	**Description**
Use Case Name	ID3: Add Trailer Listing in Saskatchewan
Primary Actor	Seller
Pre-conditions	User is logged in
Success End Conditions	Listing is added to browse page
Failed End Conditions	Listing is not added to browse page
Main Success Scenario A: Actor	I. A: Selects “My Listings” tab II. A: Selects “+” button to add a listing

S: System	III. A: Selects “Province” dropdown list IV. A: Selects “Saskatchewan” province field V. A: Selects “OK” button VI. A: Inputs “Saskatoon” in city field VII. A: Inputs “123 First Street East” VIII. A: Inputs “fls 9u8” into Postal Code field IX. A: Inputs “130000” into price field X. Inputs “600” in Sq. Feet field XI. A: Inputs “5” in Bed Rooms field XII. A: Inputs “2” in Bath Rooms field XIII. A: Inputs “Nice trailer with a little dust. Fixer upper.” XIV. A: Adds 1 Image to images XV. A: Selects “Publish” XVI. S: Attempts to Adds the listing to database
Extensions	
Priority	Critical
Response Time	< 1 second
Notes	Disallow < 0 in Price, Bedroom, Bathroom, Sq. Ft fields. Uploading images can only be done on mobile device.

Use Case Component	**Description**
Use Case Name	ID3: Browse screen
Primary Actor	Browser
Pre-conditions	User is logged in
Success End Conditions	Some of the listings being browsed are saved to favourites
Failed End Conditions	Nothing is added to favourites
Main Success Scenario A: Actor S: System	I. A: Selects "Browse” tab II. A: Selects on Image for listing III. A: Goes through listings, favouriting every second listing and disliking the others
Extensions	
Priority	Critical
Response Time	< 1 second
Notes	

5.0 Coding Style Guide

The development team has put together a set of guidelines to serve our purposes. These guidelines highlight the salient features of the coding style to be followed by developers. Useful examples are provided for quick referencing.

For front-end development using Ionic, the coding guide and sample can be found at the following wiki page:

https://github.com/CMPT371Team1/Project/wiki/Coding-Style-Example-(JavaScript)

For back-end development using Python, the coding guide and sample can be found at the following wiki page:

https://github.com/CMPT371Team1/Project/wiki/Coding-Style-Guide-(Python)

6.0 Build Report

Smoke Test Status:

Front-end tests are now implemented using protractor and are currently passing. Back-end smoke tests are also implemented and passing.

Build Status:

The builds for iOS and Android are running and simulating correctly. Both the Linux build and remote server are currently undergoing smoke tests. The server is now integrated with the front-end for two of our API calls. End-to-end smoke tests are being conducted using Protractor on the Firefox browser. The back-end is being tested with Python scripts. Until now, all testing has been done on the browser version of our app; smoke tests are currently not in place for the other platforms. Upon successful completion of the smoke tests, the build should push the source code to a branch dedicated to more rigorous forms of manual testing. This feature is currently in Beta.

SDKs, Packages, and Tools:

All SDKs, packages, and tools employed in our build, as well as their version number, are subject to change. These frameworks are still in question due to lack of experience. These decisions will be made final once the build manager has a firm understanding of automated testing, deployment, server builds, and system builds.

Current list of SDK’s, Packages, and Tools:

- Cordova CLI: 6.5.0
- Ionic Framework Version: 2.0.0-rc.5
- Ionic CLI Version: 2.2.1
- Ionic App Lib Version: 2.2.0
- Ionic App Scripts Version: 1.0.0
- npm: 3.10.10

- jdk: 1.8.0_121
- nvm: 0.32.0
- node: 6.9.4
- packages listed in package.json
- plus ~400 other Ionic dependency packages

- Android:
    - SDK Platform Android 7.1.1, API 25, revision 3
    - Android SDK Tools, revision 25.2.5
    - Android SDK Build-tools, revision 25.0.1
    - Android SDK Platform-tools, revision 25.0.3
    - Google Repository, revision 42
    - Android Support Repository, revision 42

- iOS:
    - OS: OS X El Capitan
    - Xcode version: Xcode 7.3.1 Build version 7D1014Server:

- Server:
    - Python 2.7
    - Google Cloud sdk v143.0.1
    - Python Extension for google cloud v1.9.50
    - Python Extension (Extra Libs) v1.9.49

- End-to-End Tests
    - Protractor v5.0.0
    - Firefox v47.0.1
    - Selenium v3.1

Corodova 6.5.0 requires both jdk 1.8 (or higher), as well as npm v2.2.1 and node v4.0.0. We will be using the most recent version of node and npm to reduce version conflicts. Google Cloud is required for the Google App engine; the platform our servers are built upon. The server uses Python 2.7, the most recent version of Google Cloud SDK, and Python extensions for Google Cloud SDK. Our end-to-end tests are driven by Protractor, which sends to Ionic through the Selenium server. We are currently testing our system on Firefox v47.0.1, because of compatibility issues with newer versions. All developers and testers have been set up with the latest versions of the required tools. We are using Xcode 7.3.1 for the time being. The Ionic dependencies are extensive, and can be viewed in further detail at the following link:

https://ionicframework.com/docs/

Releases for our build can be found at the link below:

https://github.com/CMPT371Team1/Project/tree/develop/releases/ID3

7.0 Upcoming Requirements

7.1 ID4 Requirements

Front-end Requirements

- Form control validation for all remaining pages
- Replace descriptive text with icons
- Fix description box in Add Listings
- Add Google Maps section to Detail page
- Display currently added images
- Get a legitimate Kasper icon
- Finalize method of ordering images
- Make Favourites page the same style as Browse page
- Get front-end fully integrated with back-end
- Finalize and complete UI flow

Priority Back-end Requirements

- Email verification (not done for ID3)
- Forgot password (not done for ID3)
- Confirm email
- Logout
- Contact Seller
- Sign in / Sign up using Facebook
- Delete Listing
- Get the front-end fully integrated with the back-end for all API calls

Other Back-end Requirements

- Thorough review of test cases for all back-end code
- Push notifications if something changes regarding a Listing saved in Favourites (a change in the database triggers a notification)
- Integration with Google Maps

7.2 Future Requirements

This section outlines requirements identified as “nice-to-have’s” for upcoming ID’s. Certain client requirements have been identified as too ambitions for the allotted time frame, and are not expected to be completed.

- Send a message to the seller including contact information
- Users can sign up to receive “hot list” notifications
  - Feature Listing (paid for by Sellers)

  - Newly added Listings
  - Price changes on Favourites
  - Based on previous search history
- Sellers receive notifications regarding personal Listings:
  - When a Listing is saved to Favourites
  - When someone requests a viewing
  - When someone sets a price watch
- Super Admin User
  - Log in as Super Admin
  - Add new Listing under any user
  - Edit any Listing
  - Remove any Listing

8.0 Triage Meeting ID3

Date: Saturday, March 4th /17
Start Time: 1:35pm
End Time:
Location: Spinks360

Members Present: Tushita Patel (Project Lead), Kristof Mercier and Dylan Prefontaine (Dev leads), Jeremy Liau (Test Lead), Chris Mykota-Reid (Build Manager)

Summary:

1) **Front-end:**
The application was redesigned after requirements changes – changes are shown using images from our new UI in section 1.1 Front-end Requirements. Some changes, such as integration with Google Maps, Detail Page, and certain features of Browse, will be carried over to ID4.

2) **Back-end**
The following tasks were completed for ID3:
   - Implemented eleven out of sixteen API calls, two of which have been tested. For the tested API calls we have achieved an average of 82.9% line coverage in our unit tests.

   The following tasks were not completed for ID3:
   - Delete Listings *NEW*
   - Sign in / Sign up with Facebook
   - Confirm email
   - Logout *NEW*
   - Contact seller *NEW*

3) **Testing**
The following test related tasks were completed for ID3:
   - Manual testing in Browser and Android

- Use cases
- Updated test matrix
- Updated path coverage diagrams
- End-to-end tests
- Testing Documentation
- Implemented Smoke Tests
- Updated defect report – with android, browser test, bugs found in bug party

The following test related tasks were completed for ID3:

- Some bugs have not been fixed – Test and Dev Teams will work together to fix all remaining bugs as quickly as possible.

4) Build

The following build related tasks were completed for ID3:

- Smoke tests are passing on the front-end in the Firefox web browser
- Smoke tests are passing on the back-end
- Multi stage development is working

The following build related tasks were not completed for ID3:

- GitHub releases
- Smoke tests for iOS and Android
- Browser smoke tests for Chrome