KASPER – ID1 PROJECT DOCUMENTATION

SOFTWARE DEVELOPMENT TEAM:

Project Manager: Tushita Patel

Developing Lead: Dylan Prefontaine

Testing Lead: Jeremy Liau

Build Manager: Christopher Mykota-Reid

Developers: Gaurav Arora, Arianne Butler, Haotian Ma, Kristof Mercier, Melody Zhao

Test Team: Christopher May, Ryan Tetland

Contents

[1.0 Requirements Document 3](#_Toc473837582)

[1.1 Requirements ID-1: 3](#_Toc473837583)

[1.2 Platform Description (iOS, Android, and Web Browser) 4](#_Toc473837584)

[1.3 GUI Mock Ups: 5](#_Toc473837585)

[1.4 Use Cases: 7](#_Toc473837586)

[1.5 Sequence Flow Diagrams: 11](#_Toc473837587)

[2.0 Design 13](#_Toc473837588)

[3.0 Testing Plan 13](#_Toc473837589)

[4.0 Test Report 13](#_Toc473837590)

[5.0 Style Guide 13](#_Toc473837591)

[6.0 Build Report 13](#_Toc473837592)

[7.0 Defect Report 14](#_Toc473837593)

[8.0 Mini Milestones for ID1 14](#_Toc473837594)

[9.0 Upcoming ID-2 Requirements 15](#_Toc473837595)

[User Requirements: 15](#_Toc473837596)

[System Requirements 16](#_Toc473837597)

[10.0 Future Requirements (Nice to Haves) 16](#_Toc473837598)

# 1.0 Requirements Document

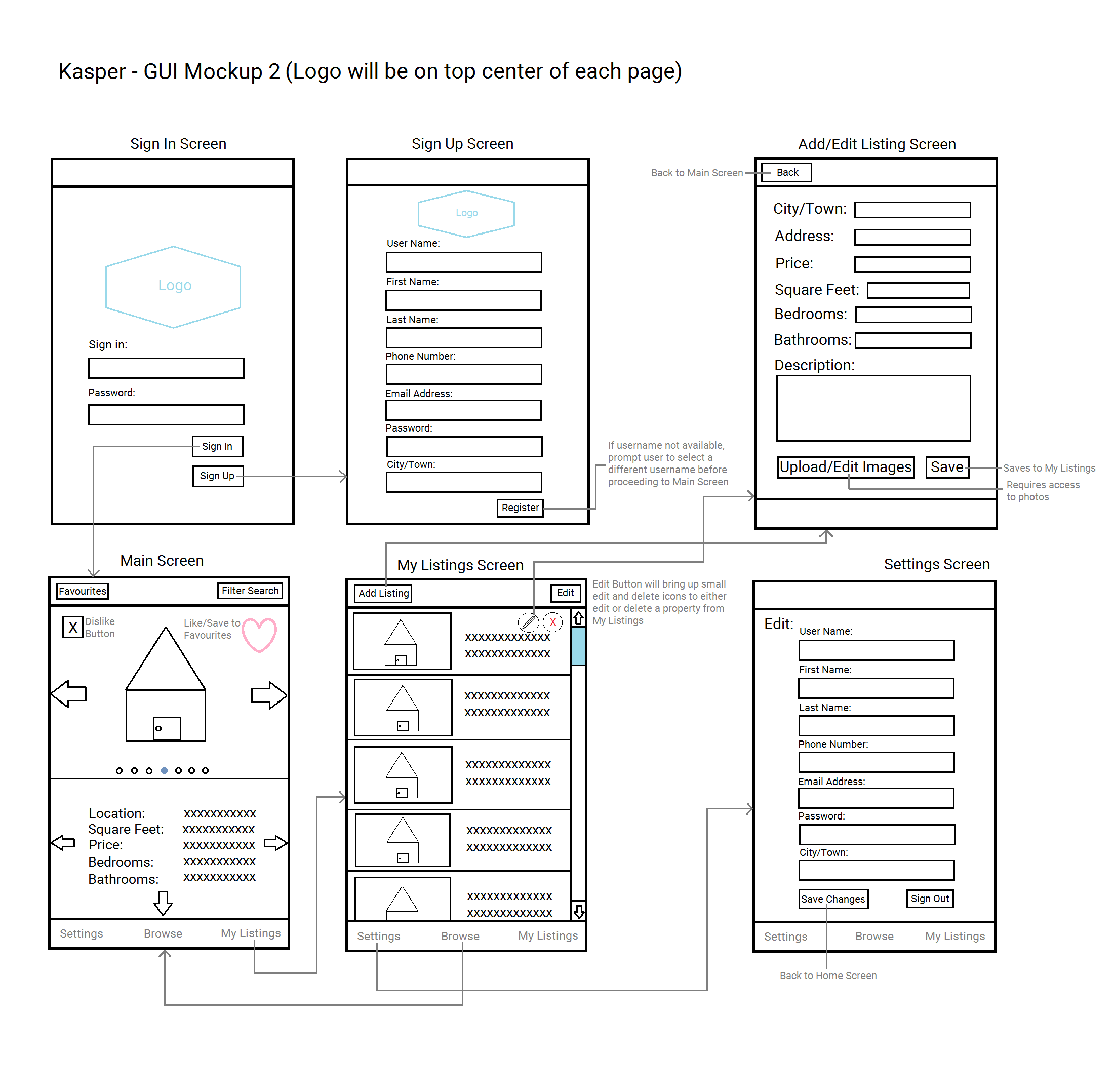
## 1.1 Requirements ID-1:

* Basic User Interface
  + Buttons
  + Text boxes
  + Range sliders
  + Menus
  + Links between screens
  + Image swipe feature
* Generate Dummy Data Structures
  + User Structure:
    - Integer 🡪 ID
    - String 🡪First Name
    - String 🡪 Last Name
    - String 🡪 Email Address
    - String 🡪 Password
    - String 🡪 Salt
    - String 🡪 PhoneNum (x3)
    - The ID 2 stuff looks good
    - Future stuff looks good as well
    - Location 🡪 Location
    - DateTime 🡪 Registered
    - DateTime 🡪 LastSeen
  + Listing Structure Fields:
    - Integer 🡪 ListingID
    - Integer 🡪 ListerID
    - Float 🡪 Price
    - Integer 🡪 SquareFeet
    - Integer 🡪 Bedrooms
    - Float 🡪 Bathrooms
    - Location 🡪 Location
    - String 🡪 Description
    - Boolean 🡪 isPublished
    - DateTime 🡪 dateCreated
    - DateTime 🡪 lastModified
    - Array 🡪 Thumbnail Images
    - Array 🡪 Images
  + Location Structure (to be determined):
    - Province 🡪 Province
    - String 🡪 City
    - String 🡪 Address
    - String 🡪 Postal Code
    - Double 🡪 Latitude
    - Double 🡪 Longitude
  + Province Structure:
    - String 🡪 Name
    - Char[2] 🡪 Abbrev

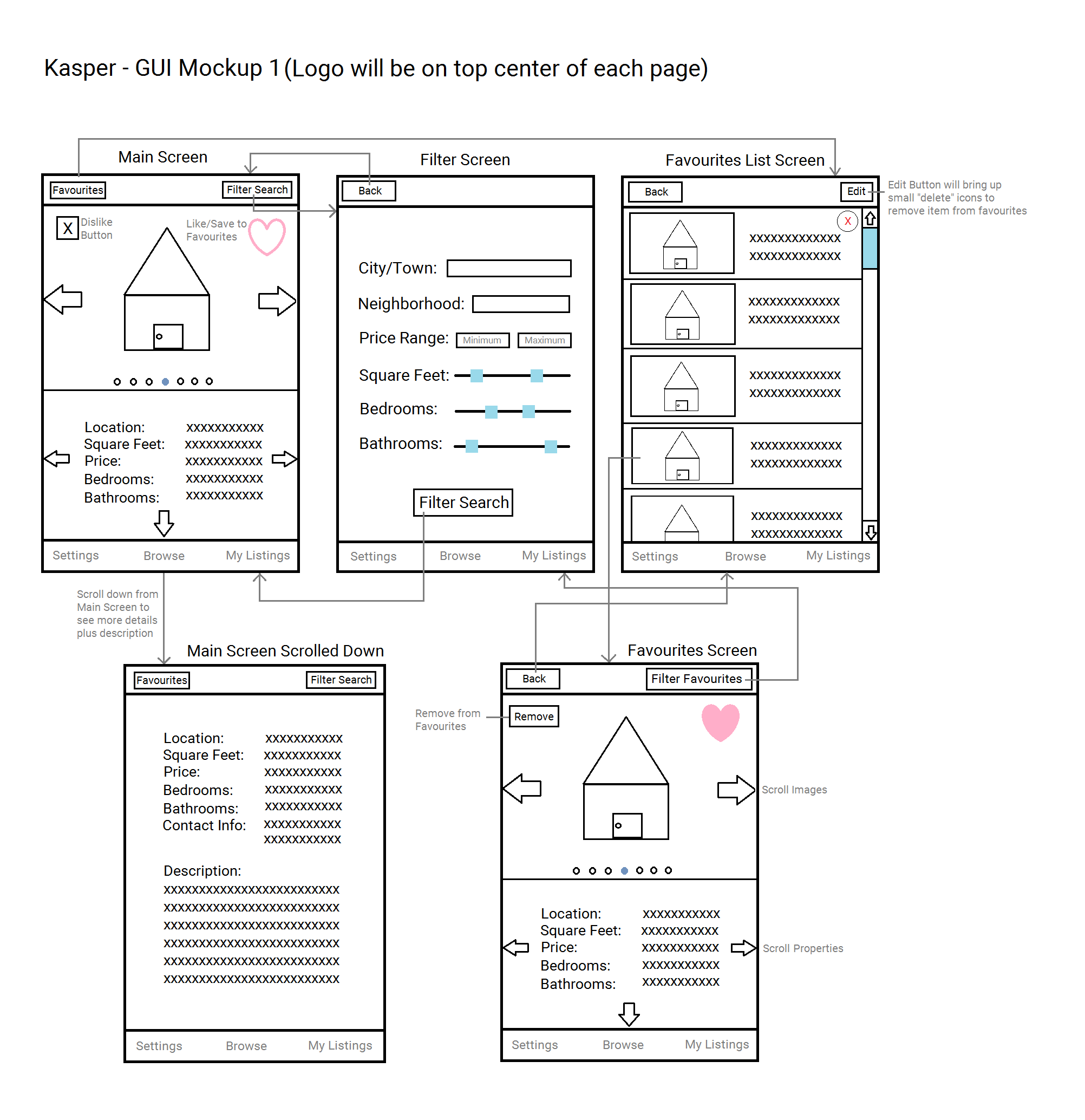
## 1.2 Platform Description (iOS, Android, and Web Browser)

* Insert Platform Description by Tushita

## 1.3 GUI Mock Ups:

GUI Mockup showing Sign In, Sign Up, Main Screen, Listings, Add/Edit Listings, and Settings

The above diagram shows the flow of events when a user chooses to either Sign in or Sign up. When a user signs in or signs up successfully, they will be taken to the Main Screen, where they can browse through an unfiltered set of property Listings. Users who wish to post their own Listings may do so by choosing the My Listings option from the bottom tab bar. Clicking on My Listings will take the user to the My Listings screen, where they can Edit, Remove, and Add New Listings. Choosing the Add Listing button from the top tab bar will navigate to the Add/Edit Listing screen, where the user can set up their new Listing, add pictures, and provide a description. Lastly, users may edit their personal profile in the Settings screen.

GUI Mockup showing Main Screen (with scroll down), Filter Search, Favourites List, and Favourites ****

The two large sideways arrows on the Main Screen indicate the ability to scroll through the provided images, while the two smaller sideways arrows below indicate the ability to swipe on to a new Listing. The small arrow on the bottom shows the ability to swipe down for more information as shown. Users will have the option to filter a search and be shown only those Listings which match their filter criteria. Users can tap the heart button to save a Listing to Favourites, and all Listings saved to Favourites can be removed by choosing the Edit button on the Favourites List Screen. To view the Listing of an item saved to Favourites, the user must click on the desired Listing in the Favourites List screen, which will bring up their selected Listing in the Favourites Screen. Users may also remove a Listing from Favourites by hitting remove on the Favourites screen.

## 1.4 Use Cases:

Sign-in:

* Actor(s):
  + User, Database
* Precondition(s):
  + app is open
* Basic Flow:
  1. User enters Username and password
  2. User clicks Sign-in
  3. User’s info is verified with the Database
  4. If password is correct the User is sent to Main Screen
* Alternate Flow:

4\*. Username/password is incorrect and User re-enters

* Post Condition(s):
  + The app is on the Main Screen

Sign-up:

* Actor(s):
  + User, Database
* Precondition(s):
  + App is open
* Basic Flow:
  1. User selects sign up button
  2. User enters information and selects register
  3. The User’s information is checked if correct and added to the Database
  4. User signs in
* Alternate Flow:

4\*. There is an issue with the info and User is prompted to fix issue

5\*. User fixes info and submits info

* Post Condition(s):
  + User has an account

Filter:

* Actor(s):
  + User
* Precondition(s):
  + User is either on the Main Screen or Favourites
* Basic Flow:
  1. User selects filter button
  2. User selects specific filters and clicks filter search button
  3. The entries are updated with search results
* Alternate Flow:

2\*. User selects back and returns back to Main Screen

* Post Condition(s):
  + Entries change based on filter input

Swipe through Listings:

* Actor(s):
  + User, Database
* Precondition(s):
  + User is on the Main Screen
  + Listings Buffer contains five Listings loaded from the Database (relevant to design in ID-2)
* Basic Flow:
  + User swipes Left or Right
  + Next Listing is loaded from the Database
* Alternate Flow: none
* Post Condition(s):
  + Next Listing is visible on the Main Screen

Add Listing to Favourites:

* Actor(s):
  + User, Database
* Precondition(s):
  + User is on the Main Screen
* Basic Flow:
  1. User selects Favourite Button on a particular Listing
  2. The Listing is added to user Favourites
* Alternate Flow: none
* Post Condition(s):
  + Listing is added to user Favourites

Dislike a Listing:

* Actor(s):
  + User, Database
* Precondition(s):
  + User is on the Main Screen
* Basic Flow:
  1. User selects the Dislike Button on a particular Listing
  2. The property is blocked from being re-added to the Listings buffer
* Alternate Flow:

3\*. The user hits the Dislike Button again and the Listing is unblocked from re-entering the buffer

* Post Condition(s):
  + The Listing is blocked from re-entering the Listings buffer for that user

Settings:

* Actor(s):
  + User, Database
* Precondition(s):
  + User is on the Main Screen
* Basic Flow:
  1. User selects Settings Button
  2. User edits Settings and clicks the Save Changes button
  3. The User’s new info is confirmed and updated in the Database
* Alternate Flow:

2\*. User clicks Sign-out button and is sent to the Sign-in Screen

* Post Condition(s):
  + Users info is successfully updated

Add Listing:

* Actor(s):
  + User, Photo Directory, Camera, Database
* Precondition(s):
  + User is on the Main Screen
* Basic Flow:
  1. User selects the Add Listing Button
  2. User fills in the Listing information and may upload photos
  3. User clicks the Save Button
* Alternate Flow: none
* Post Condition(s):
  + Listing is added to My Listings and the Data Base and is made available for other users to see

Edit Listing:

* Actor(s):
  + User, Photo Directory, Camera, Database
* Precondition(s):
  + User is on the Main Screen and has at least one Listing in My Listings
* Basic Flow:
  1. User selects listings button
  2. User selects Edit from the Listings Screen
  3. User submits the edited info
  4. The new listing info is verified and then updated in the entries Database
* Alternate Flow: none
* Post Condition(s):
  + Listing modified in My Listings and the Database

View Favourites List:

* Actor(s):
  + User
* Precondition(s):
  + User is on the Main Screen
* Basic Flow:
  + User selects Favourites Button
* Alternate Flow: none
* Post Condition(s):
  + The user is on the Favourites List Screen

View a Favourite Listing:

* Actor(s):
  + User
* Precondition(s):
  + User is on the Main Screen
* Basic Flow:
  + User selects a Listing from the Favourites List
* Alternate Flow: none
* Post Condition(s):
  + The user is on the Favourites Screen of the particular Listing

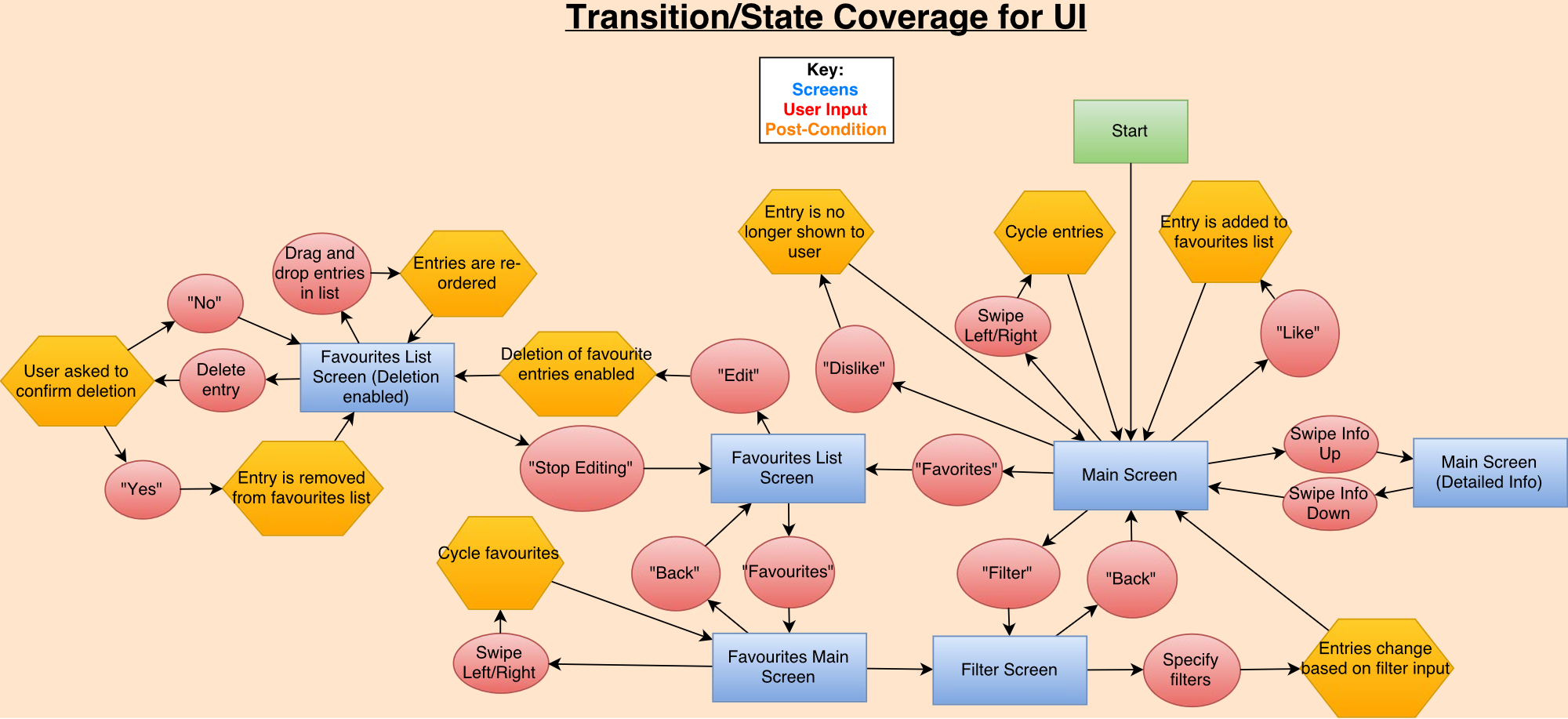
Sign-up:

* Actor(s):
* Precondition(s):
* Basic Flow:
* Alternate Flow:
* Post Condition(s):

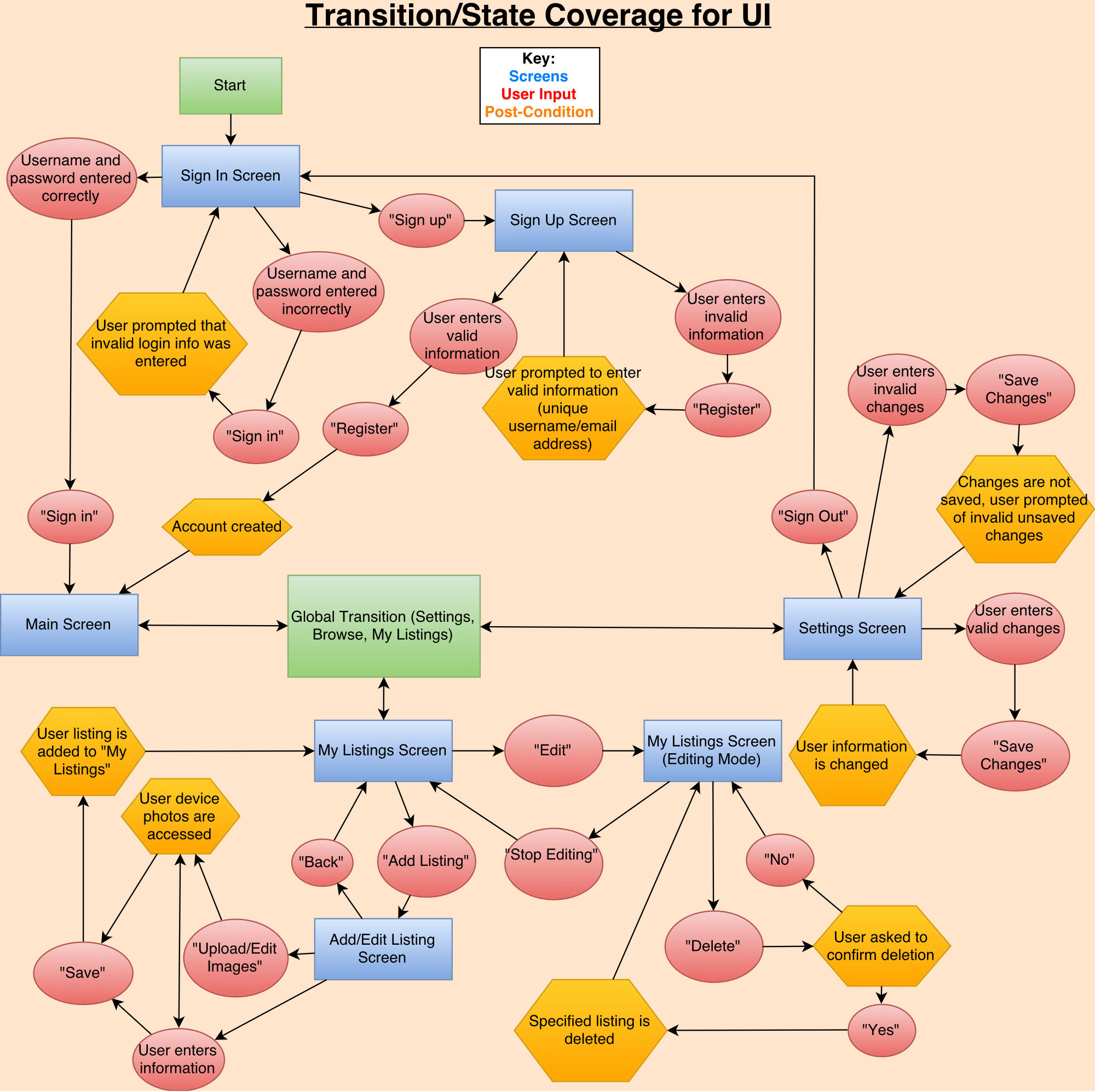
Remove Listing from Favourites:

* Actor(s):
  + User, Database
* Precondition(s):
  + User is on Favourites List Screen
* Basic Flow:
  + User selects the Edit Button
  + User selects the Remove Button
  + User confirms removal of Listing
* Alternate Flow:
* Post Condition(s):
  + Listing is removed from Favourites List

## 1.5 Sequence Flow Diagrams:

****

* Jeremy will replace these and test team will provide brief description

****

# 2.0 Design

* UML Diagrams -> DYLAN
* Architecture Diagram -> GAURAV
* Architecture Description / Defense -> KRISTOF

SELECT \* FROM (table) WHERE (current range\*) ORDER BY (chosen field), id LIMIT 100

\*this range will be stored on the device to resume the search later.

* when you get to 100, call the same sort (sorted by filter) again on the database and skip to the stored cursor position

Architecture is 3-Tiered such that tier 1 is View/Model components, tier 2 is Controller, tier 3 is the server. MVC is used by ionic.

# 3.0 Testing Plan

* Jeremy and Test Team

# 4.0 Test Report

* Jeremy and Test Team

# 5.0 Style Guide

# 6.0 Build Report

Smoke Test and CI

* Chris MR

# 7.0 Defect Report

# 8.0 Mini Milestones for ID1

**Dev Team:**

* Every developer familiar with Ionic2, CSS, HTML, typed Script and Angular2 ✔
* Each developer has completed at least one App Page
* Unit Testing of GUI’s where possible (not much to test in ID-1)
* Decide on System Architecture ✔
* Finalize Design for the System
* Implement GUI Designs

**Test Team:**

* Visual testing of the GUI’s implemented by Dev Team
* Testing reports complete
* All testers familiar with testing frameworks ✔

**Build:**

* Set up Travis with Ionic
* Verify code style (low priority)
* Set up Django, goDaddy and server

**Documents:**

* Testing Plan Document
* Risk Report
* Requirements Document
* Design Document
* All documents edited and compiled into one fluid document

**Other:**

* Each Developer and Tester has completed a pair programming session
* Get server to host Django
* Learn how to host Website on goDaddy.com
* All group member’s familiar with Trello, GitHub, Slack, and Google Documents ✔
* Finish GUI Design and Mock-ups for ID1 ✔
* Triage meeting
* Meet with Upper Management
* Meet with client weekly ✔
* Every member familiar with the Schedule as given by project manager ✔

# 9.0 Upcoming ID-2 Requirements

## User Requirements:

* Generate Dummy Data Structures:
  + UserVotes:
    - Integer 🡪 UserID
    - Integer 🡪 ListingID
    - Boolean 🡪 isLiked
  + Filter:
    - Location 🡪 Location
    - Float 🡪 minPrice
    - Float 🡪 maxPrice
    - Integer 🡪 minBedroom
    - Integer 🡪 maxBedroom
    - Integer 🡪 minFeet
    - Integer 🡪 maxFeet
    - Boolean 🡪 isFavourites
  + BrowseList:
    - Integer 🡪Cursor
    - Integer[100] 🡪 ListingIDs
    - Listings[5] 🡪 curLoaded
  + ListingImages:
    - Integer 🡪 ListingID
    - Image 🡪 Image
    - Integer 🡪 OrderDisplayed
* Generate Mock Classes:
  + ListingProvider:
    - addListing(ListingToJSON: String)
    - editListing(ListingToJSON: String)
    - removeListing(ListingID: Integer)
    - dislike(ListingID: Integer)
    - addToFavourites(ListingID: Integer)
    - removeFromFavourites(ListingID: Integer)
    - search(Filter: Filter) 🡪 Filter can be Null
  + LoginProvider (to be determined)

General User Requirements:

* Log on
* Sign up
* Edit settings and User Info
* View all property listings
* Listing swipe feature (cursor)

Buyer Specific Requirements:

* Filter Search Based on:
  + City/Town
  + Address
  + Category (house, condo, building, etc.)
  + Square Feet
  + Price range
  + Number of bedrooms
  + Number of bathrooms
* View all Listing Info (including pictures, descriptions, and seller contact info)
* Save Listing to Favourites
* Browse Favourites
* Remove Listing from Favourites
* Receive notifications regarding Favourites
  + Price changes
  + Listing removed/edited

Seller Specific Requirements:

* View personal listings
* Edit personal listings (text fields, description, and images)
* Add new listings (contact info can be automatically added via sign-up info)
  + Upload images
  + Input:
    - City/Town
    - Address
    - Category (house, condo, building, etc.)
    - Square Feet
    - Price
    - Number of bedrooms
    - Number of bathrooms
* Remove Listings

## System Requirements

* SSL certificate
* Server for data storage
* Domain access

# 10.0 Future Requirements (Nice to Haves)

* Book a viewing feature
* Sign in using Facebook feature
* Set price watch on a given Listing
* Users can sign up to receive “hot list” notifications
  + Feature Listings (paid for by Sellers)
  + Newly added Listings
  + Price changes on Favourites
  + Based on previous search history
* Push notifications if something changes regarding a Listing saved in Favourites (change in database triggers notification)
* Sellers receive notifications regarding personal listings:
  + When a listing is saved to Favourites
  + When someone requests a viewing
  + When someone sets a price watch
* Integration with Google Maps
* Super admin User:
  + Log in as Super Admin
  + Add new listing under any user
  + Edit any listing
  + Remove any listing
* Future System Requirements
  + Tian