KASPER - ID ONE

Software Development Team:

Gaurav Arora, Arianne Butler, Jeremy Liau, Haotian Ma, Christopher May, Kristof Mercier, Christopher Mykota-Reid, Tushita Patel, Dylan Prefontaine, Ryan Tetland, and Melody Zhao

Contents

[Requirements - ID 1 3](#_Toc473377316)

[Platform Description (iOS, Android, and Web Browser) 4](#_Toc473377317)

[GUI Mock Ups: 4](#_Toc473377318)

[Sequence Flow Diagrams: 6](#_Toc473377319)

[Design 8](#_Toc473377320)

[Smoke Test Status 8](#_Toc473377321)

[Mini Milestones for ID1 8](#_Toc473377322)

[Risk Report 8](#_Toc473377323)

[Activity Log 9](#_Toc473377324)

[Inspections Report 9](#_Toc473377325)

[Client Meetings 9](#_Toc473377326)

[Testing Results and Defect Reports 9](#_Toc473377327)

[Upcoming ID 2 Requirements (remove all that are done for ID1) 9](#_Toc473377328)

[User Requirements: 9](#_Toc473377329)

[System Requirements 11](#_Toc473377330)

[Future Requirements (Nice to Haves) 11](#_Toc473377331)

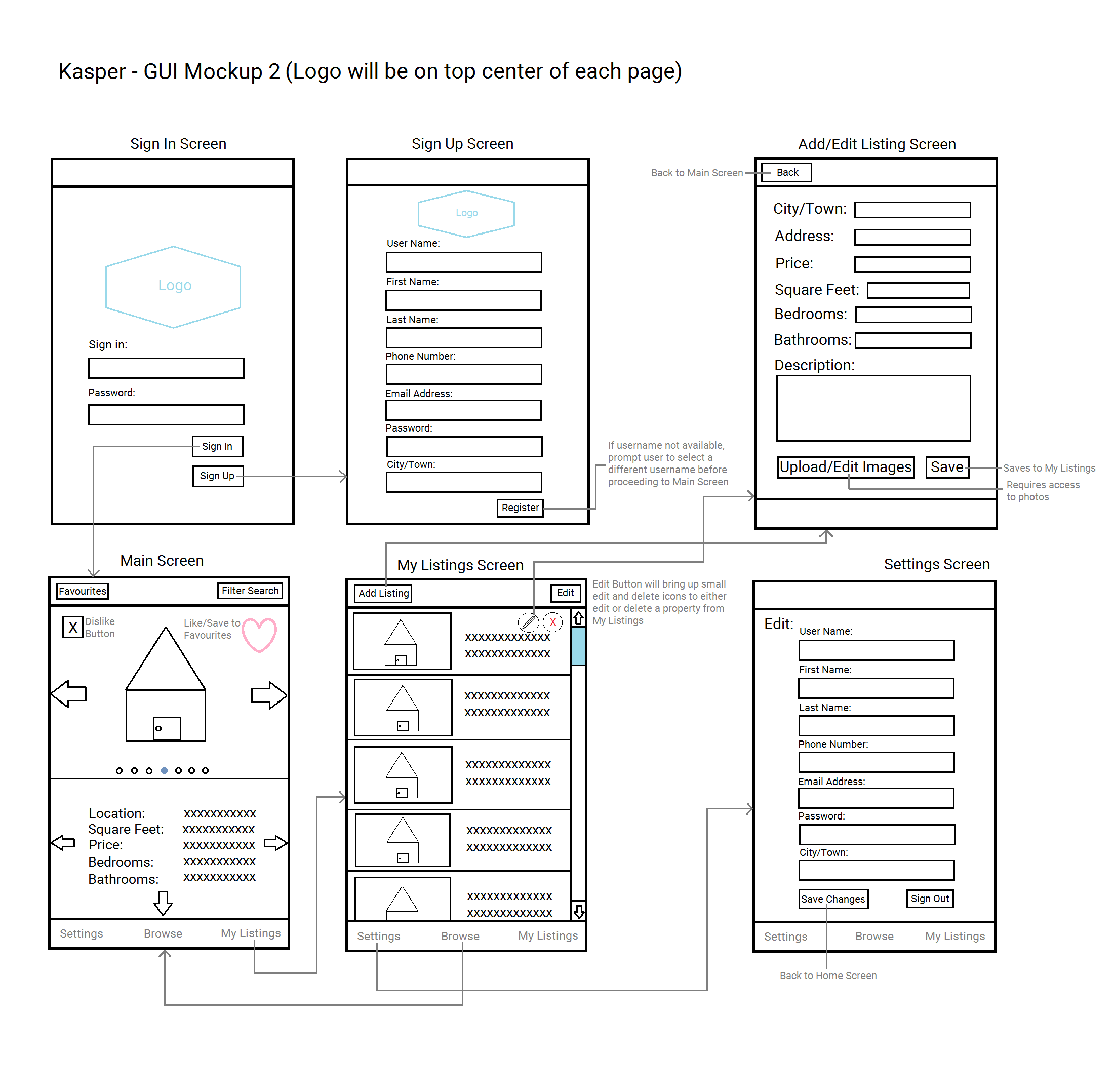
Requirements - ID 1

* 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 🡪Phone Number
    - String 🡪 Email Address
    - String 🡪 Password
    - String 🡪 Salt
    - String 🡪 PhoneNum (x3)
    - 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
    - Image 🡪 Thumbnail
  + 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

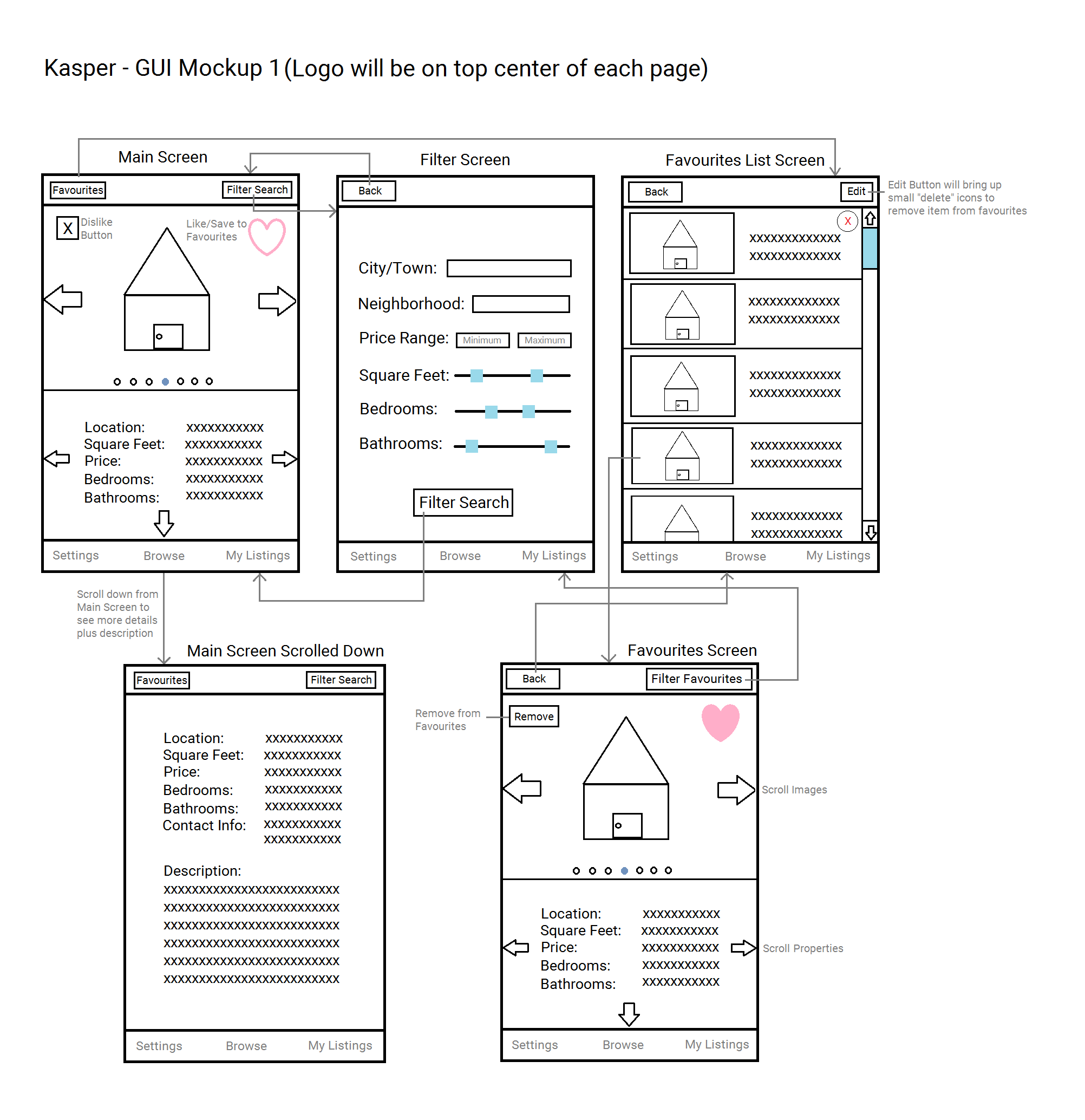
## Platform Description (iOS, Android, and Web Browser)

* Insert Platform Description

## GUI Mock Ups:

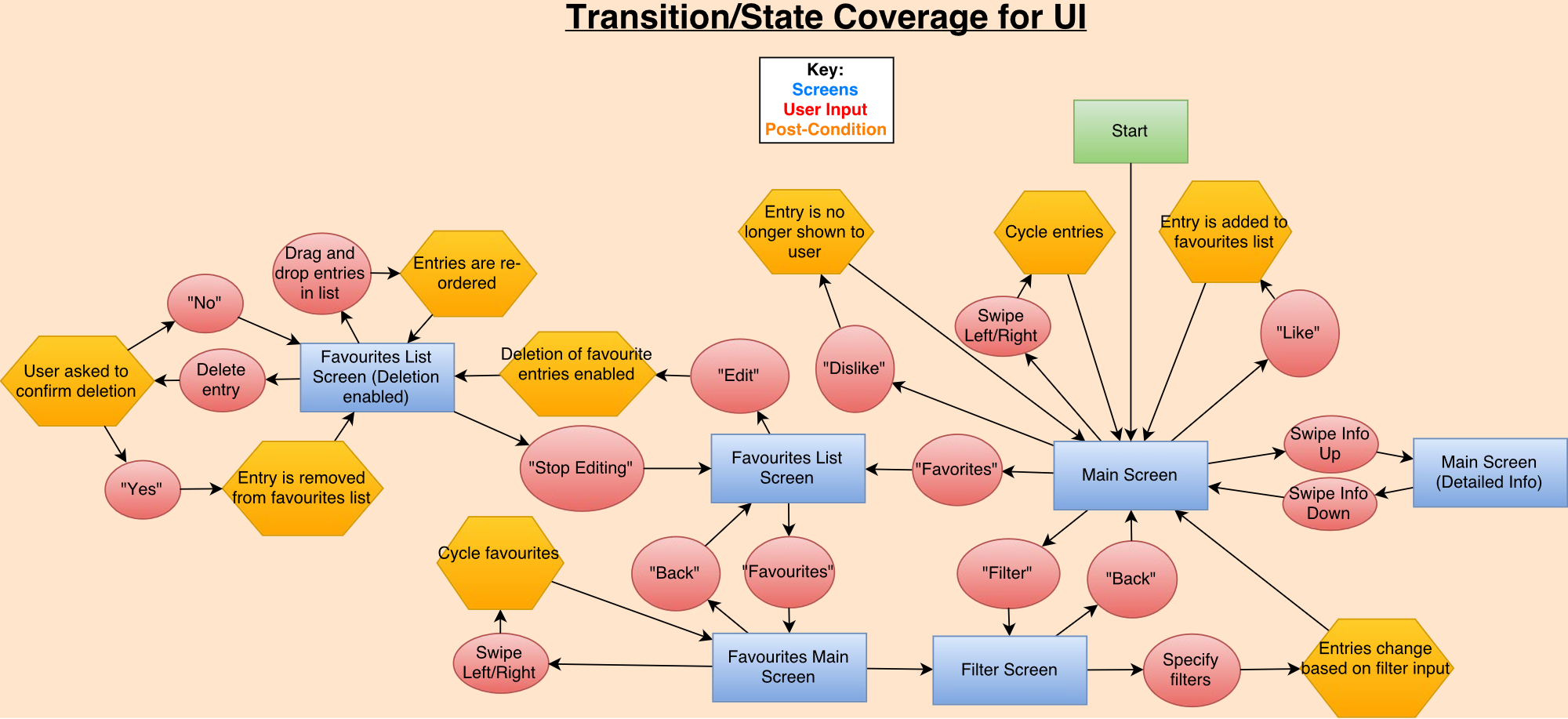
GUI Mockup showing Sign In, Sign Up, Main Screen, Listings, Add/Edit Listings, and Settings

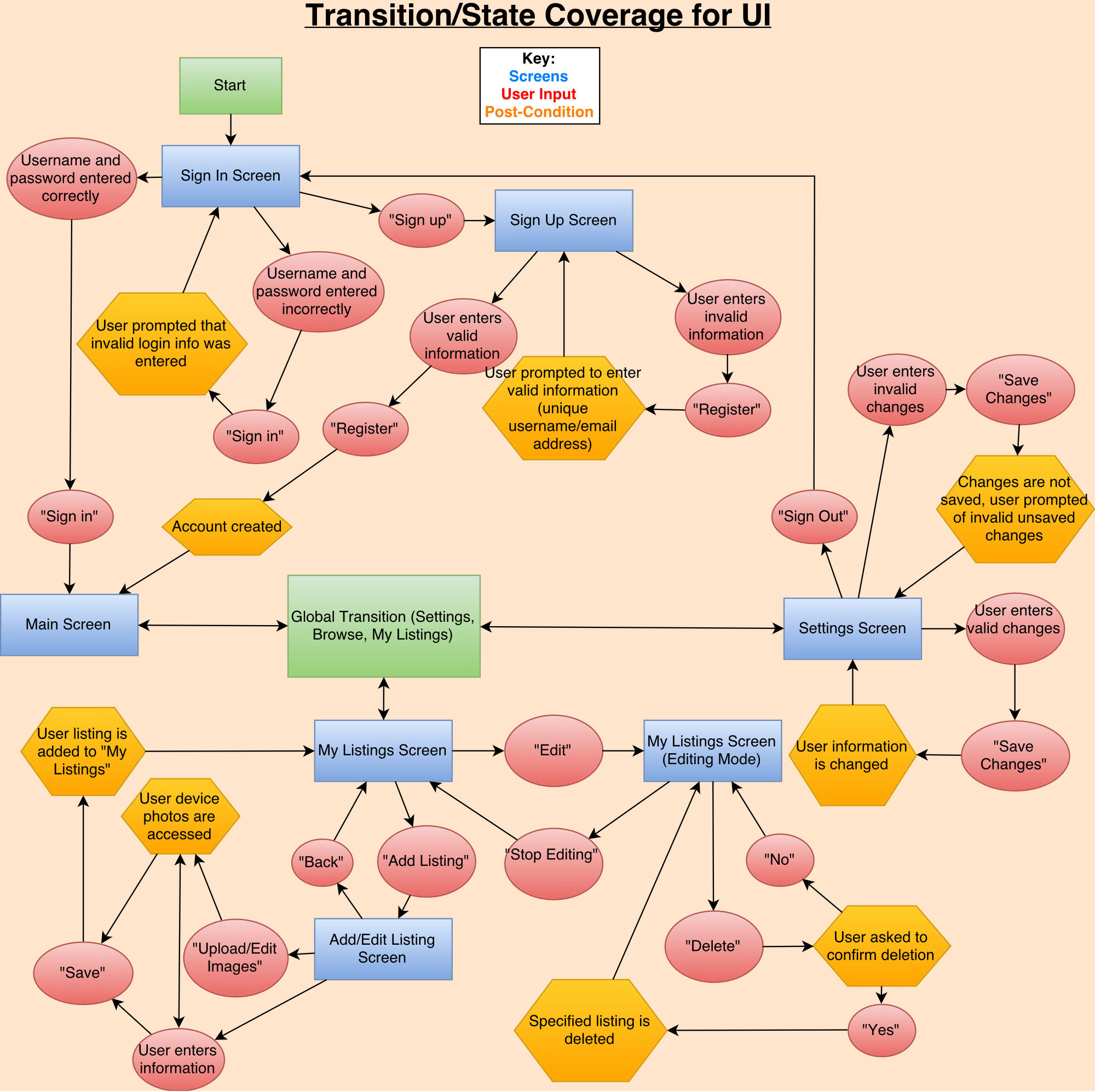
The above diagram shows the flow of events when a user chooses to either Sign in or Sign up. When a user signs in or signs up successfully, they will be taken to the Main Screen, where they can browse through an unfiltered set of property Listings. Users who wish to post their own Listings may do so 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.

## Sequence Flow Diagrams:

****

****

# Design

* uml plus short description

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.

# Smoke Test Status

# Mini Milestones for ID1

* All group members’ familiar with Trello, GitHub, Slack, and Google Docs ✔
* All developers familiar with Ionic2, CSS, HTML, Typed Script and Angular2 ✔
* All testers familiar with testing frameworks
* Finish GUI Design and Mock-ups for ID1 ✔
* Decision on System Architecture ✔
* Finalizing the Design for the System
* All developers are assigned at least one App Page
* Implement all GUI’s
* Unit testing for all ID1 elements
* Build master has Travis with Ionic setup
* Each Dev and Test has a pair programming session
* Testing Plan Document
* Risk Report
* Learn how to host Website on goDaddy.com
* Getting a server to host Django Server
* Put together all reports into final ID1 Document

# Risk Report

# Activity Log

# Inspections Report

# Client Meetings

# Testing Results and Defect Reports

# Upcoming ID 2 Requirements (remove all that are done for ID1)

## 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

# 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