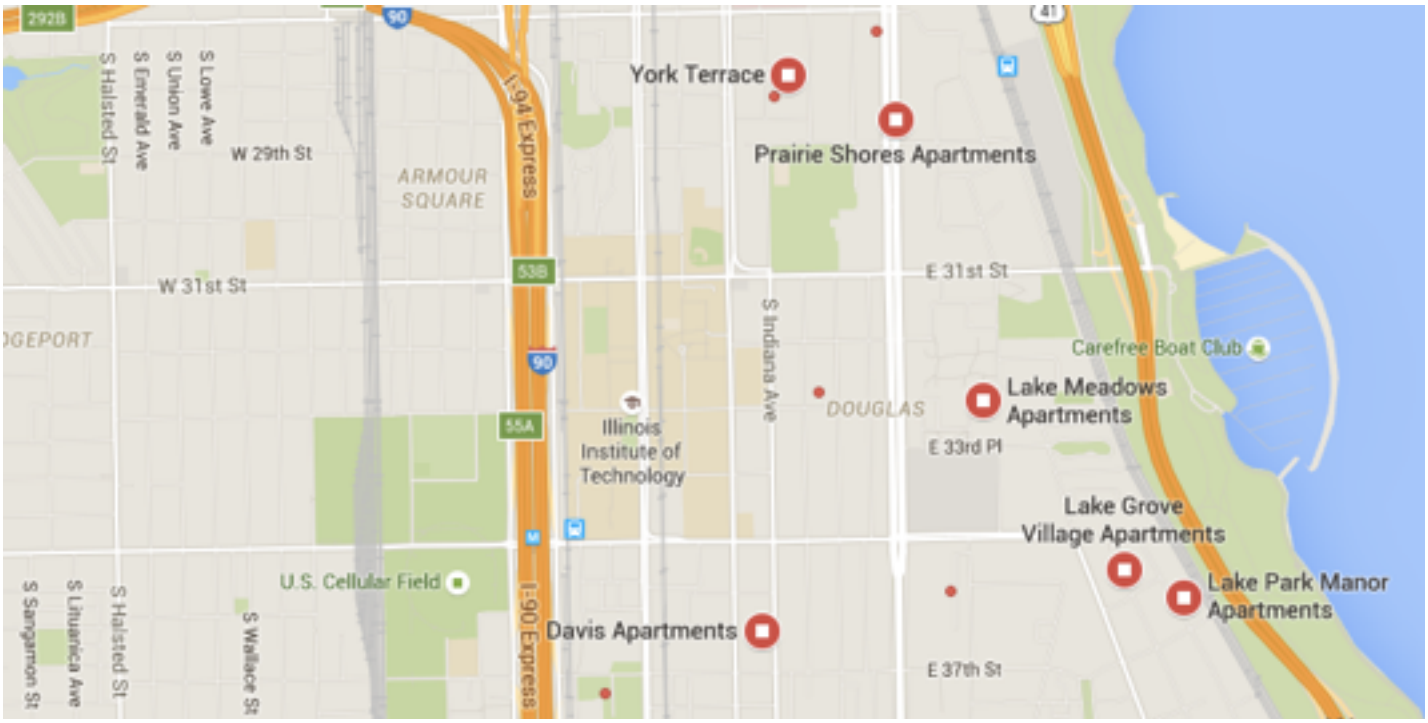
Apartment Finder — eHome

Final Project Report



Team 15

Thomas Molenhouse (A20251438) — Phase Manager

Gan Xu (A20338959)

Priyank Shah (A20344797)

Vedant Godhamgaonkar (A20353267)

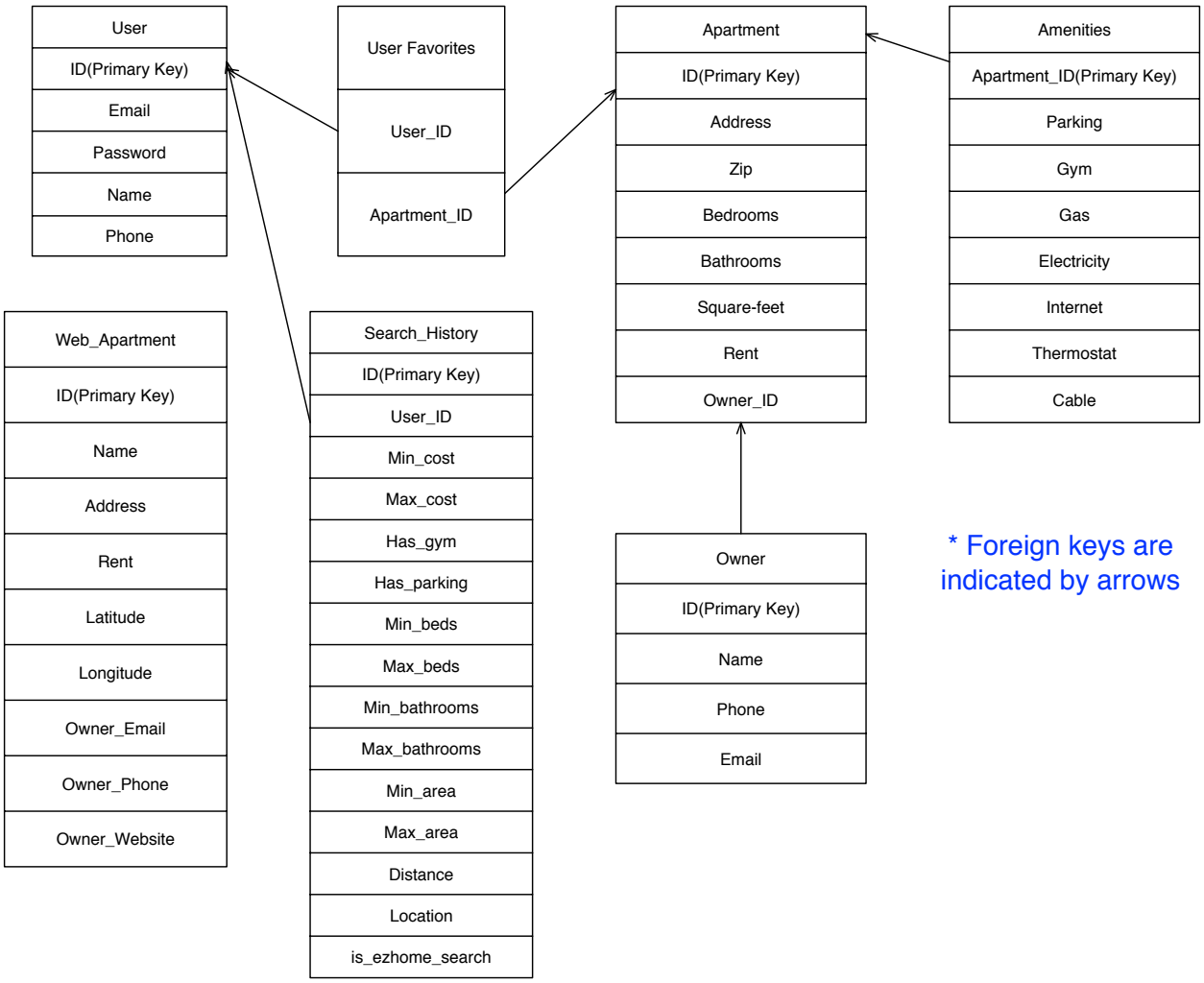
Contents

1. Data structure and list of class
   1.1 Data structure
   1.2 List of classes
2. Design contents: requirements and features
   2.1 Use case diagram
   2.2 System transition diagram
   2.3 requirements and features
3. Test cases
4. User manual
5. Marketing
6. Individual contribution
7. Revision history

1. Data Structure and list of classes

1.1 Data Structure

The database includes seven tables: User, Apartment, Web_Apartment, Owner, User_Favorites, Searching_History. Their fields and relationships are described in the following diagram:



Web_Apartment is an isolated table for OnlineSearch Fragment, the other tables are linked by foreign keys. Details are in app/assets/create.sql and app/assets/offline_apartments.db.

1.2 List of classes

1. Model

This folder consists of five classes: **Amenity.java, Apartment.java, Owner.java, User.java and WebApartment.java**. They are encapsulated java classes for respective tables in database with all data fields, mutators and other methods for future use.

2. Util

- **ApartmentDatabaseHelper.java**

This class extends the SQLiteOpenHelper class. It includes default methods for a DatabaseHelper class and multiple methods for adding, updating, deleting and querying operations in database for future use(e.g. addUser, updateUser, removeFavorite, getSearchFilter, etc).

- **ApartmentSearchFilter.java**

This class is designed for generating correct SQL query sentence when the user provides different requirements for searching apartments. We used string appending method in case the user doesn't provide specific requirements on some fields (e.g. didn't set minimum bedrooms, unsure if parking is mandatory)

- **Chicago.java** (**Interface**)

This interface stores parameters for SearchOnlineFragment.

- **ImageAdapter.java**

This class is a customized PaperAdapter.

- **PasswordDialog.java**

This class provides proper dialogs for different situations in user login. For example, if user enters a wrong password, "The pass word was incorrect. Please try again" will popup.

- **SavedLogin.java**

This class applies SharedPreferences to restore user's login status if the user closes the app and then opens it again. It's created when user successfully login and cleared when user chooses to logout.

- **SimpleTextWatcher.java**

This class implements TextWatcher where beforeTextChanged and onTextChanged are already defined to do nothing.

- **Validation.java**

This class applies android.util.Patterns to check if the email, name, phone number and password provided by the user are valid or not.

3. Main activities and fragments

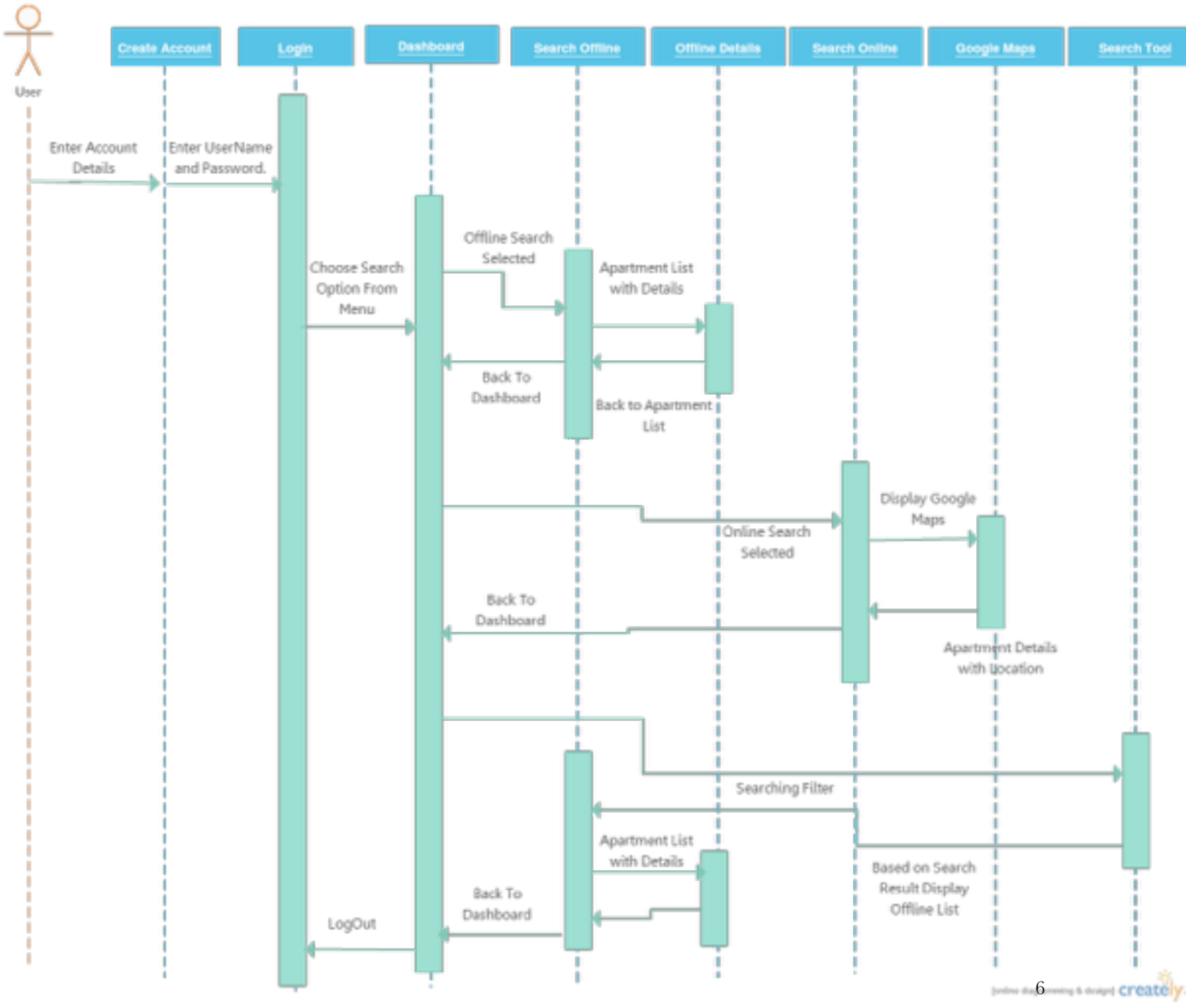
- **LoginActivity.java**
- **RegisterActivity.java**
- **MainActivity.java**
- **DashboardFragment.java**
- **EzHomeSearchFragment.java**
- **EzHomeSearchOptionsActivity.java**
- **EzHomeSearchDetailsActivity.java**
- **SearchHistoryFragment.java**
- **SearchOnlineFragments.java**
- **SearchOnlineOptions.java**

- **AccountSettingFragment.java**

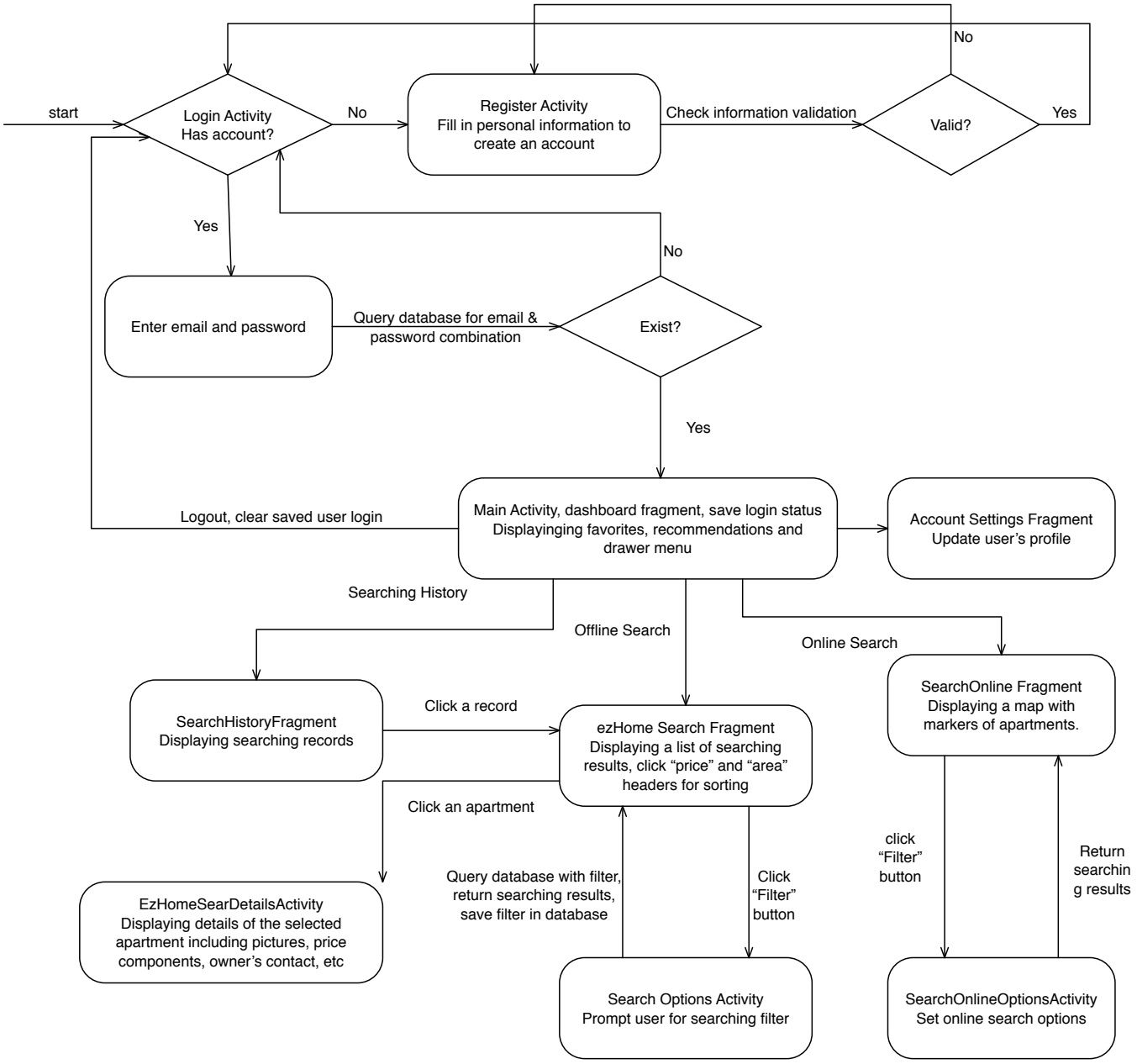
Details of these classes will be discusses in the requirement and design part.

2.Design Contents: Requirements and Features

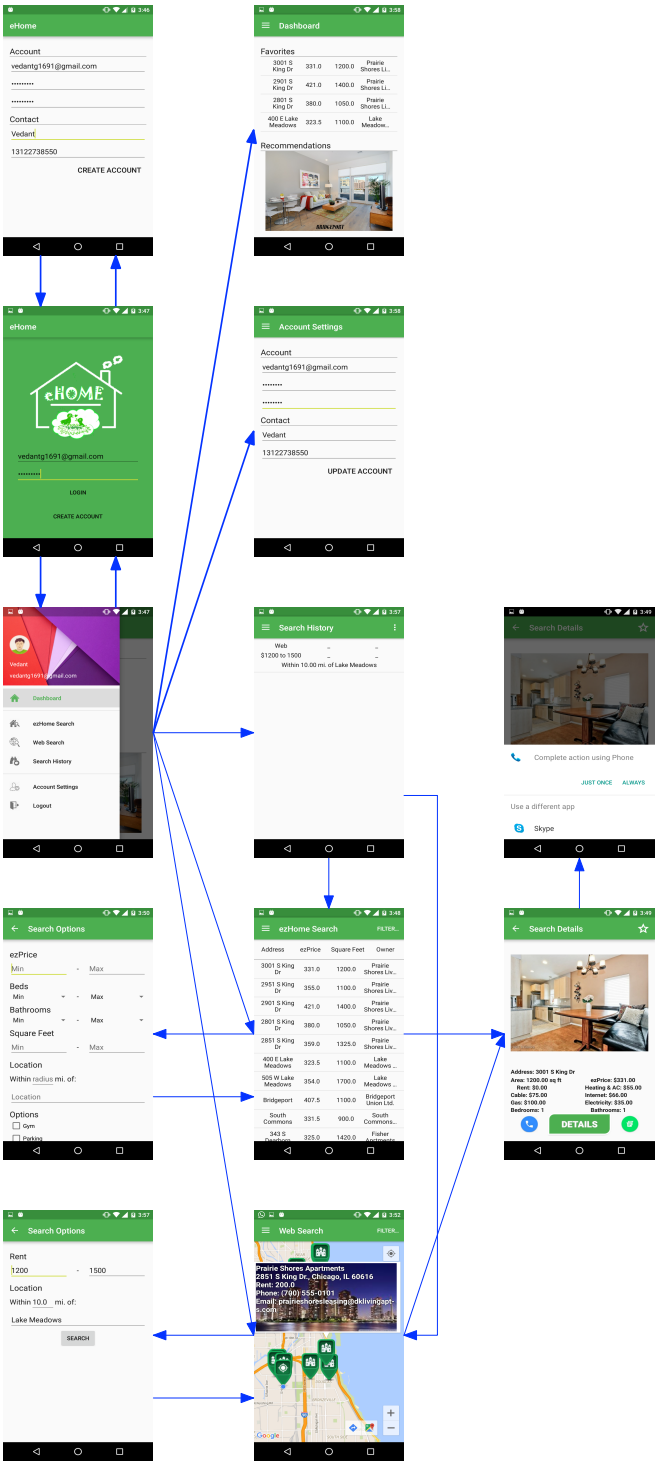
2.1 Use Case Diagram



2.2 System Transition Diagram



Transition diagram with screenshots:



These two diagrams depict an overall view of the application consisting of user's interaction with different modules and system transition in different situations.

2.3 Requirements and Features

- **Version Compatibility**

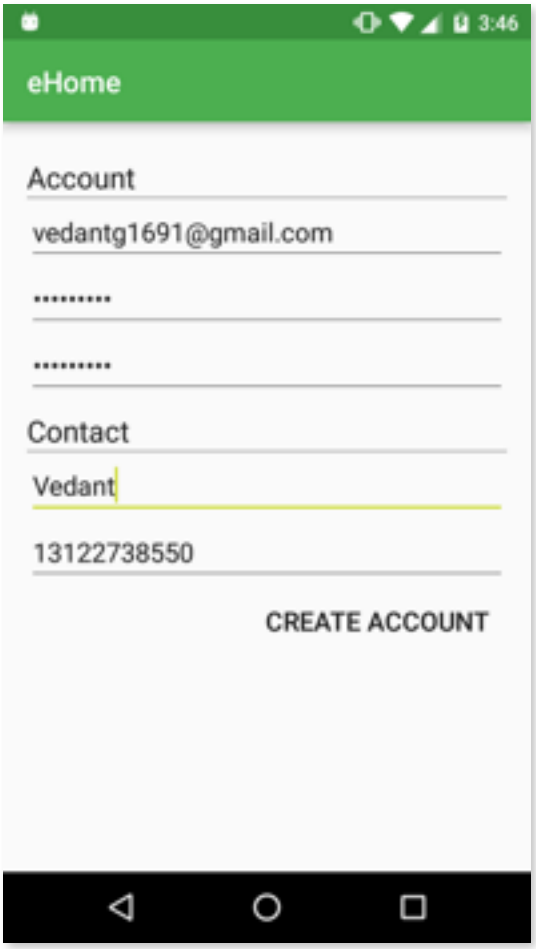
  Version code is 1, Minimum version is API 17 and Compiled on API 23. The size of the .apk file is approximately 4MB.

- **Non-Functional Requirements**
  - **Performance**: The application should have short response time.
  - **Capacity and Scalability**: The application should guarantee sufficient storage space for future expansion.
  - **Usability**: The application should be user-friendly, especially to elders and those who're inexperienced in using smart phones.
  - **Security:** The application should avoid user profile from leaking.
  - **Data Integrity**: The application should provide reliable and up-to-date data.
  - **Reusability:** Modules can be used in other places and add new features without much modification.
  - **Recoverable:** The application should be able to recover from unexpected termination.

- **Functional Requirements & Features**

1) RegisterActivity

- Autofill phone field with user's current phone number.
- Notify the user with empty fields while clicking CREATE ACCOUNT button.
- Check if Confirm Password matches Password and Email, Name, Phone Number conform to android.util.Patterns. If not, alert message will appear on the screen to prompt the user for correct values.
- Create a new User and insert into the user table in database if every information is valid.
- Autofill email and password with the information provided in the previous RegisterActivity when returning to LoginActivity.

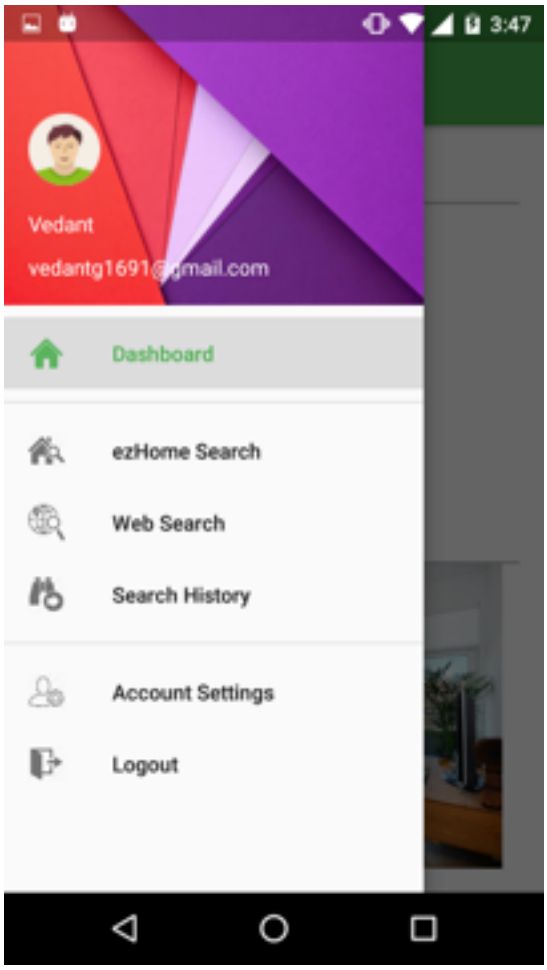


2) LoginActivity

- Notify the user when Email or Password is empty while clicking LOGIN button.
- Query the user table in database for email and password combination entered by user. If not exist, the system will ask the user to enter correct combination. Otherwise, save user's login status in SavedLogin class and switch to MainActivity.
- Clear SavedLogin when user chooses Logout in MainActivity.

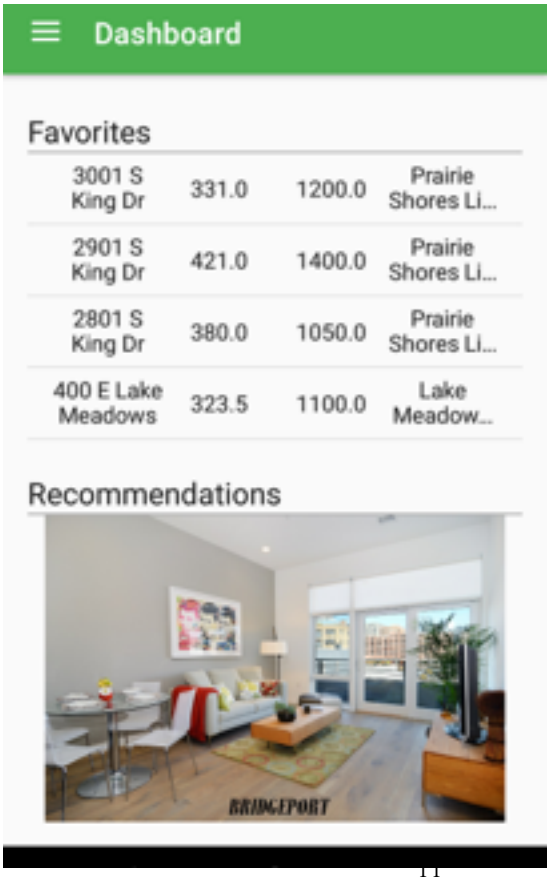
3) MainActivity

- The MainActivity consists of five fragments: Dashboard, ezHome Search, WebSearch, Search History and Account Settings. User can switch between these fragments with the navigation drawer menu on the left.
- The default fragment would be the dashboard.
- Display user's name and email on the header of drawer.



4) DashboardFragment

- Query User_Favorites table in database and display current user's favorite apartments.
- Display recommended apartments in a swipefragment.
- Click an apartment in favorites or recommendations will switch the app to EzHomeSearchDetailsActivity to view details of selected apartment.

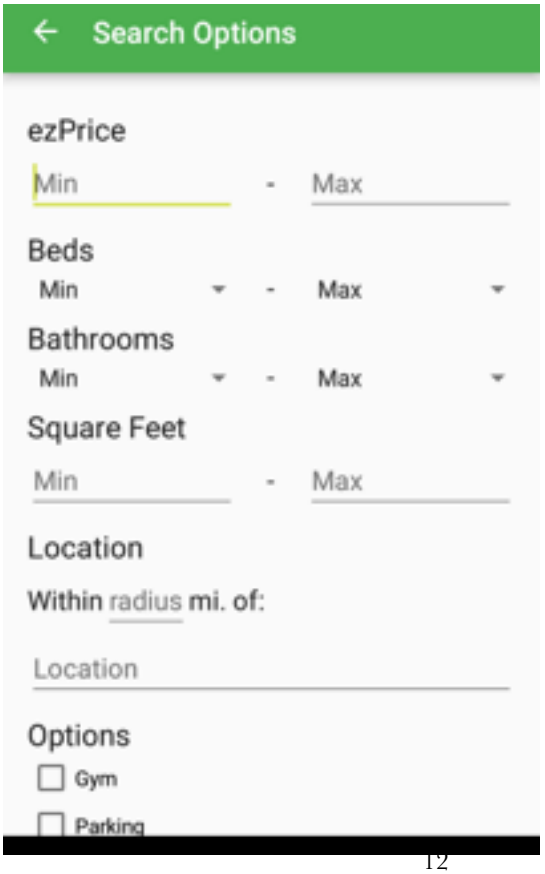


5) EzHomeSearchFragment

- Display a scrollable list of apartments from user's last search, if the user has no searching history, display all the apartments in database.
- Sum up rent and all amenities cost in ezPrice.
- Click ezPrice and Square Feet TextView to sort ascending or descending.
- Click a row in the list to see details of the clicked apartment in EzHomeSearchDetailsActivity.
- Click FILTER in the upper right to set searching filters in EzHomeSearchOptionsActivity.
- If no apartments are found, a message will show up instead of the list view.

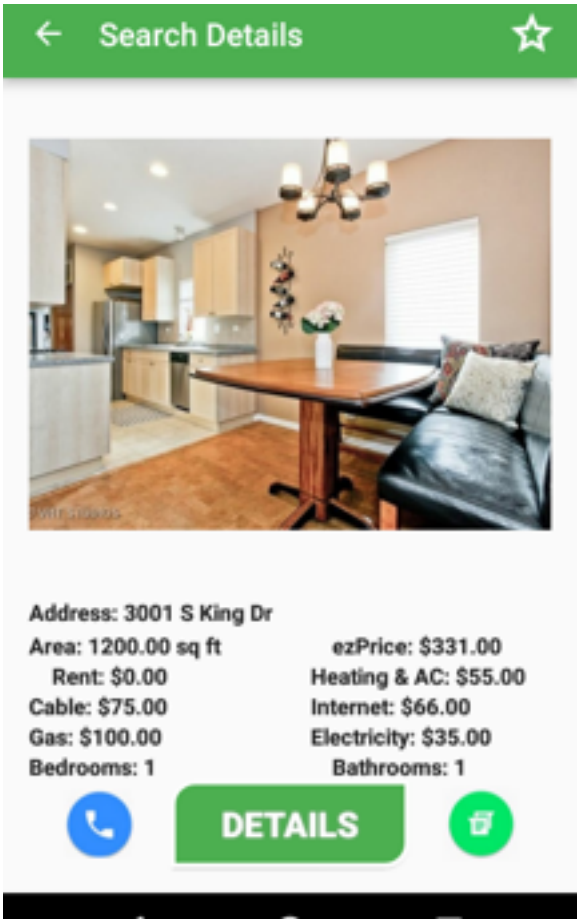
6) EzHomeSearchOptionsActivity

- Permit the user to set searching filters in spinners and checkboxes on multiple data fields.
- Autofill with user's last searching options saved in Searching_History table in database.
- Show alert message if Min value is larger than Max value.
- Generate correct SQL query language in util/ ApartmentSearchFilter.java and return to EzHomeSearchFragment with searching results when user clicks SEARCH button.
- Save searching options in Search_History table in database while clicking SEARCH button.



7) EzHomeSearchDetailsActivity

- Query database for a specific apartment id then display pictures and details of the apartment.
- Click the phone or email icons in the bottom right to make a phone call or send an email to the apartment owner
- Click the pentagon in the upper right to add this apartment to user's favorite list, then it will be listed in the dashboard.

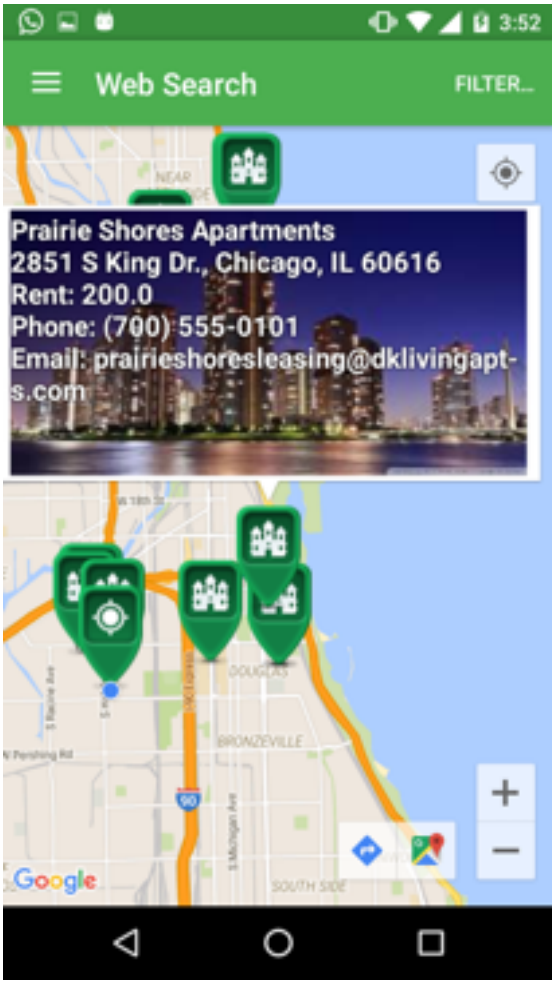


8) SearchHistoryFragment

- Query Search_History table in database, display previous search options of current user in a list.
- Click an offline search record to view search result in EzHomeSearchFragment, click an online search record to view search result in WebSearchFragment.
- Show a "No History" message if current user doesn't have any search history.

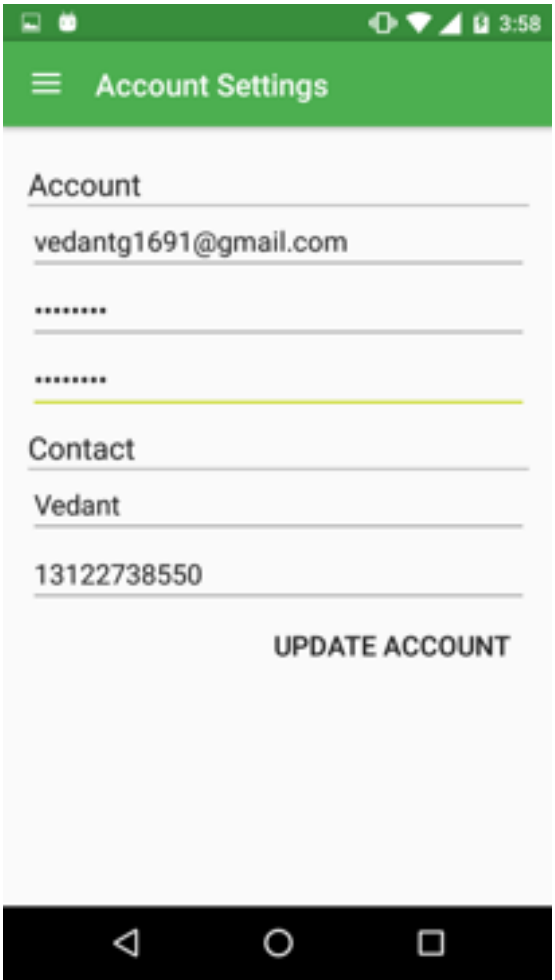
9) SearchOnlineFragment

- Use Google Map to search apartment online.
- Marks some apartments on the map, show an info windows with details when clicking these marks.
- Zoom in on user's location or Chicago
- Click FILTER button to set filters for online search, including rent range and location range.
- Save search history in database.



10) AccountSettingsFragment

- Auto-fill current user's email, name and phone number.
- Notify the user if email, name or phone is empty when click UPDATE ACCOUNT button.
- Check if new Email, Name and Phone Number conform to android.util.Patterns. If not, alert message will appear on the screen to prompt the user for correct value.
- Require the user to enter current password when the user clicks UPDATE ACCOUNT button.
- Update the user table in database and SavedLogin if every information is valid.



3. Test Cases

3.1 Registration, Login and Settings

Test case #1:

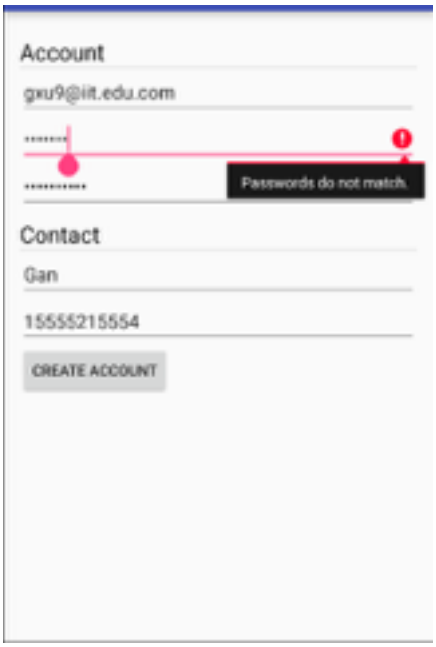
Description: In RegisterActivity, ConfirmPassword doesn't match password.

Data: Password:12345678

Confirm Password:1234567890

Expected Result: Deny registration, show error message.

Actual Result: As expected.



Test case #2:

Description: In RegisterActivity, user enters a password that isn't long enough or invalid email and phone number.

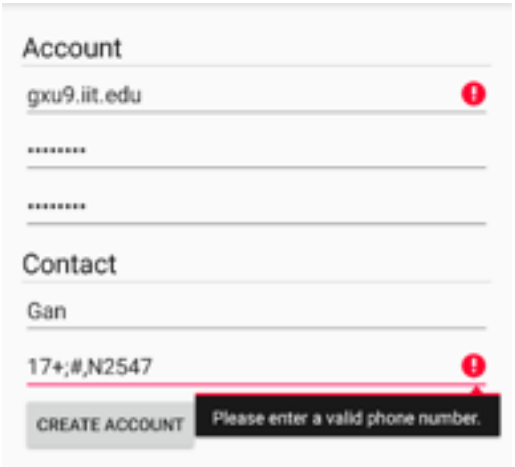
Data: Email: gxu9.iit.edu

Phone: 17+;#,N2547

Password: 1234

Expected Result: Deny registration, show error message.

Actual Result: As expected.



Test case #3:

Description: In RegisterActivity, provide valid information on every field.

Data: Email: gxu7@hawk.iit.edu

Password:12345678

Confirm Password:12345678

Contact: Gan

Phone: 15555215554

Expected Result: Successfully created an account.

Actual Result: As expected.

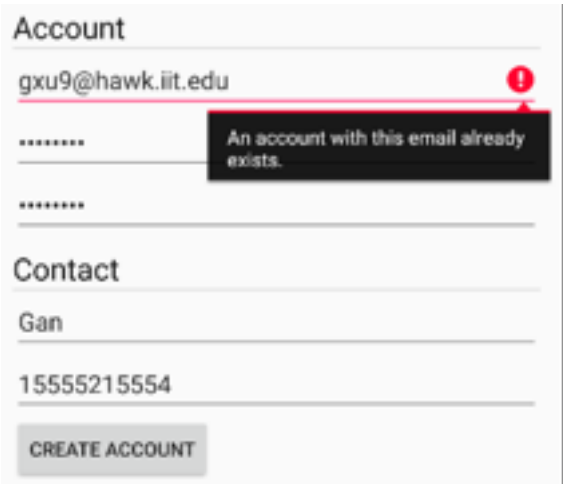
Test case #4:

Description: In RegisterActivity, register with already-existed email.

Data: Email: gxu9@hawk.iit.edu

Expected Result:Deny registration, show error message.

Actual Result: see picture.



Test case #5:

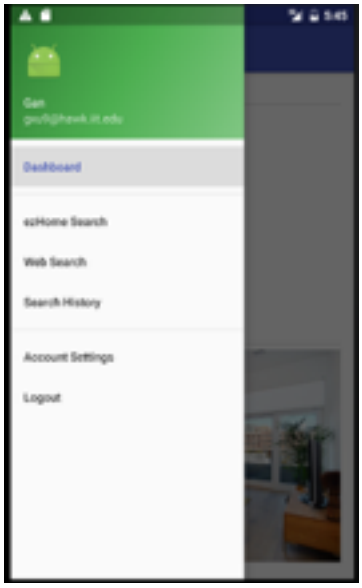
Description: In LoginActivity, enter incorrect email & password combination

Data: Email: gxu100@hawk.iit.edu

Password:12345678

Expected Result:Login failed, show error message.

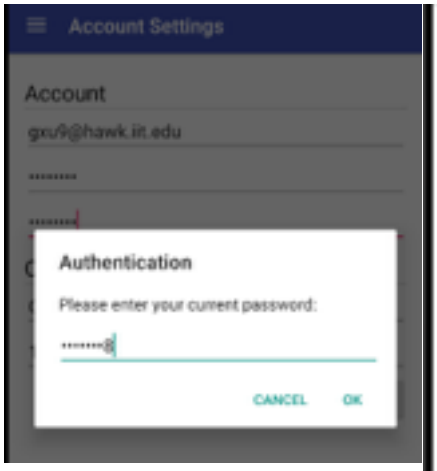
Actual Result: see picture.

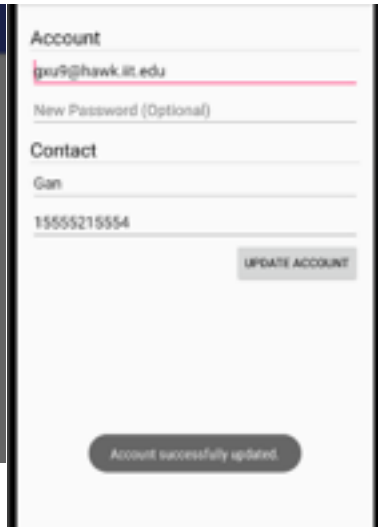


Test case #6:
Description: After successfully login, exit the app and then open it again.
Data: N/A
Expected Result: Stay login status.
Actual Result: As expected, see picture.

Test case #7:
Description: After successfully login, click Logout in drawer menu.
Data: N/A
Expected Result: Return to login page, clear SavedLogin status
Actual Result: As expected.

Test case #8:
Description: In AccountSettings fragment, try to change password.
Data: New Password:11111111
Current Password:12345678
Expected Result: An additional line will show up to confirm password and prompt for current password to verify identity.
Actual Result: As expected.





Test case #9:

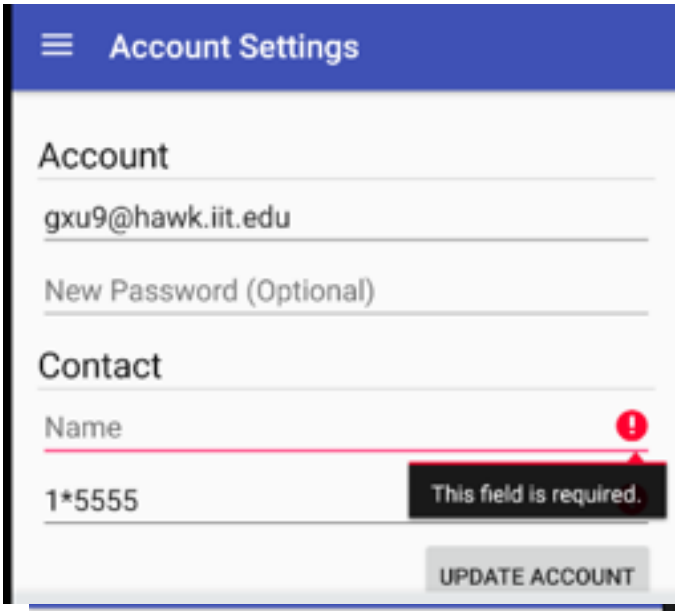
Description: In AccountSettings fragment, try to update name and phone with invalid value.

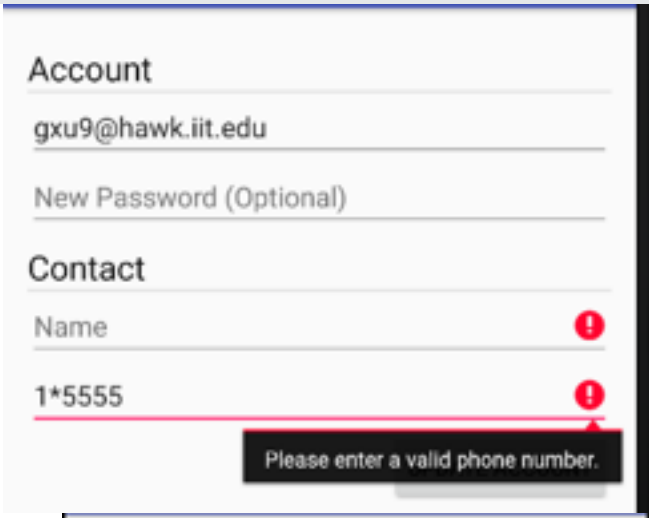
Data: Name: keep blank, null.

Phone: 1*5555

Expected Result: Reject update, show error message

Actual Result: As expected.





Test case #10:

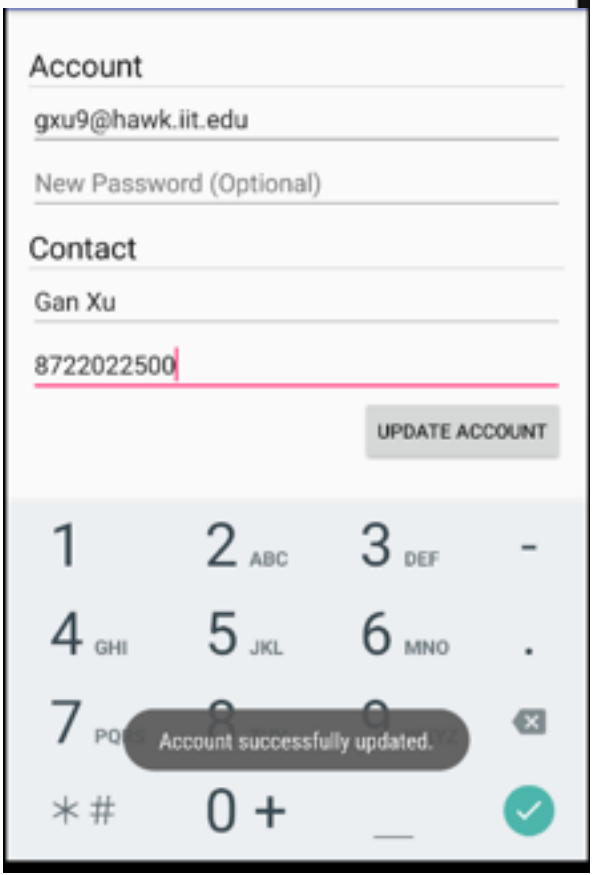
Description: In AccountSettings fragment, try to update name and phone.

Data: New Name: Gan Xu

New Phone 8722022500

Expected Result: Requires the user to enter correct password, then update user profile.

Actual Result: As expected.



3.2 Offline search

Test case #11:

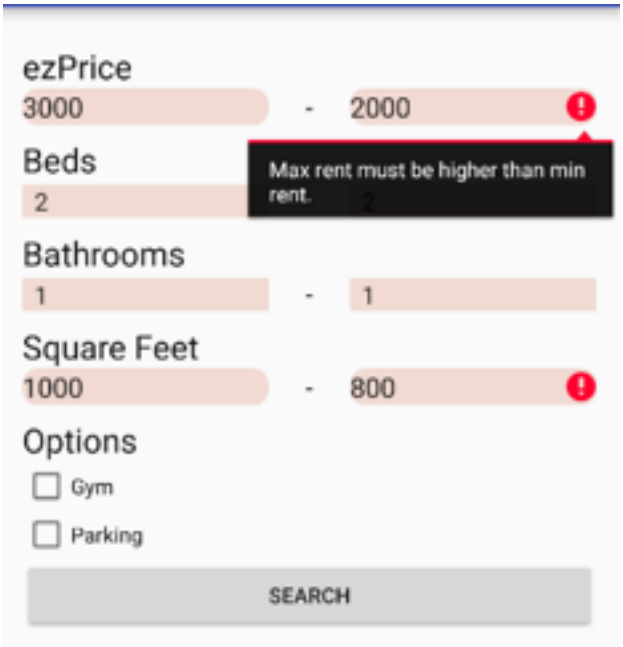
Description: Enter improper data in Search Options

Data: min rent = 3000, max rent =2000

min square feet =1000, max sure feet = 800

Expected Result: Show error message

Actual Result: As expected.



Test case #12:

Description: Leave every field blank in Search Options

Data: N/A

Expected Result: Return to ezHomeSearchFragment, show all apartments in database

Actual Result: As expected.

—>

Test case #13:

Description: In ezHomeSearchFragment, click ezPrice and Square Feet to sort.

Data: N/A

Expected Result: Show sorted results.

Actual Result: As expected.

Address	ezPrice	Square Feet	Owner
All Zero	0.0	0.0	TEST DATA
Invalid Apartment Va...	4.0	-1.0	TEST DATA
Invalid Amenity Values	995.0	500.0	TEST DATA
2801 S King Dr	1280.0	1050.0	Prairie Shores Livings
South Commons	1311.5	900.0	South Commons Pr...
400 E Lake Meadows	1343.5	1100.0	Lake Meadows Complex
3001 S King Dr	1411.0	1200.0	Prairie Shores Livings
585 N State St	1576.0	1200.0	Mr. Adam Smith
343 S Dearborn	1585.0	1420.0	Fisher Aprtments
2951 S King Dr	1655.0	1100.0	Prairie Shores Livings
Bridgeport	1747.5	1100.0	Bridgeport Union Ltd.
2851 S King Dr	1859.0	1325.0	Prairie Shores Livings
2901 S King Dr	2521.0	1400.0	Prairie Shores Livings
505 W Lake Meadows	2654.0	1700.0	Lake Meadows Complex
This apartment	1.4999999E7	9999999.0	TEST DATA

Address	ezPrice	Square Feet	Owner
This apartment has a very lon...	1.4999999E7	9999999.0	TEST DATA
505 W Lake Meadows	2654.0	1700.0	Lake Meadows Complex
343 S Dearborn	1585.0	1420.0	Fisher Aprtments
2901 S King Dr	2521.0	1400.0	Prairie Shores Livings
2851 S King Dr	1859.0	1325.0	Prairie Shores Livings
3001 S King Dr	1411.0	1200.0	Prairie Shores Livings
585 N State St	1576.0	1200.0	Mr. Adam Smith
400 E Lake Meadows	1343.5	1100.0	Lake Meadows Complex
2951 S King Dr	1655.0	1100.0	Prairie Shores Livings
Bridgeport	1747.5	1100.0	Bridgeport Union Ltd.
2801 S King Dr	1280.0	1050.0	Prairie Shores Livings
South Commons	1311.5	900.0	South Commons Pr...
Invalid Amenity Values	995.0	500.0	TEST DATA
All Zero	0.0	0.0	TEST DATA
Invalid	4.0	-1.0	TEST DATA

Test case #14:

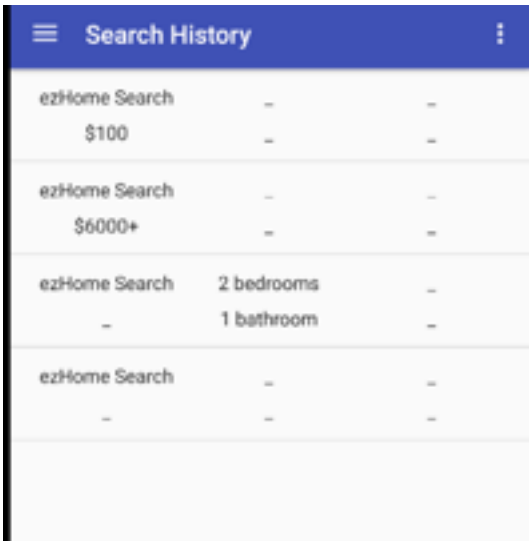
Description: In search Options, set some limit so that no apartments can be found.

Data: min rent = 1, max rent =1

Expected Result: Return to ezHomeSearchFragment, show message.

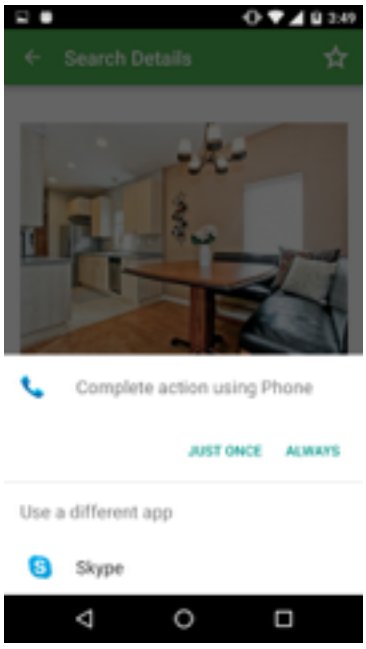
Actual Result: As expected.

Test case #15:
Description: In SeachHistoryFragment, click an item.
Data: N/A
Expected Result: Return to ezHomeSearchFragment, show search results.
Actual Result: As expected.



click the 3rd history

Test case #16:
Description: In EzHomeSearchDetailsActivity, click the pentagon to add favorites.
Data: N/A
Expected Result: See the apartment in dashboard's favorite list.
Actual Result: As expected.



Test case #17:
Description: In SeachHistoryFragment, click the phone and email icons.
Data: N/A
Expected Result: Make phone calls and send emails to the owner.
Actual Result: As expected.

3.3 Online search

Test case #18:

Description: In SearchOnlineOptionsActivity, set a range for location, then return to SearchOnlineFragment

Data: N/A

Expected Result: Only show apartments in than range on Google map.

Actual Result: As expected.

Test case #19:

Description: In SeachHistoryFragment, click an OnlineSearch Record.

Data: N/A

Expected Result: Switch to SearchOnlineFragment, show search results on Google Map.

Actual Result: As expected.

3.4 System Tests

Test case #20:

Description: Turn off internet and pull out SIM card

Data: N/A

Expected Result: The app works normally.

Actual Result: As expected.

4. User Manual

- If you already have an account, please login with your email and password. Otherwise, please click "CREATE ACCOUNT" to create an account with your personal information.
- After logging, you can always switch among dashboard, offline search, online search, search history and account settings in the navigation drawer menu in the upper-left corner.
- In the dashboard, you can view your favorites apartments and popular apartments recommended by us, you can click an apartment to view the details.
- In the EzHomeSearch fragment, you can search our offline database with your customized filter — just click the "FILTER" button in the upper-right corner. You can click the headers "EzPrice" and "Area" to sort searching results can click an apartment in the result list to view the details. Besides, we will automatically save your searching history in our database, you can view them in the SearchHistory fragment.
- In the OnlineSearch fragment, you can see some apartments with markers in Google Map, you can also choose to view apartments in a selected area by clicking the "FILTER" button in the upper-right corner. Again, we will store your searching history automatically.
- When you are viewing the details of an apartment, if you'd like to contact the owner, simply click the phone or email icons!
- You can view your past searching history in the SearchingHistory fragment and click one of them to view the results in offline search or online search fragment.
- You can update your profile in the AccountSettings fragment, password required.
- We'll save your login status automatically so that you don't need to login again when you use our app in the next time. But if you don't want to save login status, please click "log out" in the drawer menu before you quit the app.

5. Marketing

- Easy navigation through various features
- Gives wide variety of options for apartment search
- Searching can be done in either ways: Online or Offline
- One-click calling & SMS
- Online Search within specified radius
- Version Compatibility

6. Individual Contribution

1. Thomas Molenhouse (A20251438)

- Setup Git repository
- Worked on Login, Account Creation/Settings, Dashboard and application navigation modules and database
- Fixed bugs and made improvements to other team members modules

2. Vedant Godhamgaonkar (A20353267)

- Worked on Online Search Module
- Worked on documentation and UML diagrams

3. Priyank P Shah (A20344797)

- Worked on Search Details Module
- Worked on database and UI

4. Gan Xu (A20338959)

- Worked on Offline Search Module
- Worked on final project report and diagrams

7. Revision history

- Updated UI and screenshots
- Added version compatibility, transition diagram with screen shots, user manual, marketing and individual contribution
- Added descriptions for new classes and features implemented in this phase