CALIFORNIA STATE UNIVERSITY
FULLERTON™

Department of Computer Science

This project has been satisfactorily demonstrated and is of suitable form.

This project report is acceptable in partial completion of the requirements for the Master of Science degree in Computer Science.

Rental-Insight

Project Title (type)

Hung Chau

Student Name (type)

Dr. Shawn X. Wang

Advisor's Name (type)

Advisor's signature   Date

Reviewer's name

Reviewer's signature   Date

Table of Content

Abstract

As demand for rental housing increases, there is a need for better tools to help simplify the search process. Current web-based solutions only offer standard information focused on the rental unit itself such as rent, location, bedroom capacity, bathroom capacity, and a brief description. Although these information are important, they do not provide a complete picture of a rental unit, which may lead to ill-informed decision making. Rental-Insight is a hybrid mobile application that offer users a neighborhood profile in addition to standard information about a rental unit. This neighborhood profile include information about local crimes, nearby grocery stores and schools. Users could also survey a neighborhood without being there physically. Rental-Insight is a proof-of-concept that a neighborhood profile along with standard information can help renters make better decisions when searching for rental housing.

Keywords: Rental housing, neighborhood profile, hybrid mobile application, Firebase, Ionic Framework

Introduction

Increasingly more people are opting to rent as the cost of owning a home has increased in recent years (Zillow, 2015). According to the U.S. Census Bureau, the national rental vacancy rate of 7.0% was 1.2% lower in the fourth quarter of 2014 compared to the same quarter in 2013, and 0.4% lower than the previous quarter (Callis & Kresin, 2015). From the beginning of the 2009 recession to the end of 2014, homeownership in the U.S. has dropped from 67.3% to 64% (Callis & Kresin, 2015). Furthermore, the Washington Post reported that the majority of Millennials, those born between 1980 and 2000, plan to rent as opposed to buying a home in the near future (Goldfarb, 2015). In other words, demands for rental housing has increased significantly and is likely to continue to grow.

Moreover, people today relying more on the Internet as a tool to search for places to live. A joint study by the National Association of Realtors (NAR) and Google reported that real estate related searches increased 22% from 2011 to 2012 (NAR & Google, 2013). In addition, about one fifth of those searches occurred on a mobile device. This represents a 120% growth in real estate related mobile searches from 2011 to 2012 (NAR & Google, 2013). Therefore, a hybrid mobile application was developed to address these growing trends in rental housing and mobile usage.

Problem Statement

Currently, there exist several popular ways for one to find rental housing. A nontechnical approach is to read the classify section of a newspaper. The problem with this approach is that a classify usually does not contain much information about the rental. Typically, they do not include pictures that show what the rental unit looks like, or a map to show exactly where the property is located. This lack of detail is well addressed by popular web-based solutions such as Forrent.com, Apartment.com, Craigslist, and Zillow. These web solutions allow renters to see where a rental unit is located on a map, images of the rental unit, and basic details about prices, requirements, restrictions, amenities, etc. Without questions, these information are very helpful when a renter is looking for a new home. However, they do not offer a complete picture to aid renting decisions.

Renters often consider the neighborhood where the rental unit is located. They may consider the safety of a neighborhood when deciding where to live. Is the neighborhood prone to robbery, vandalism, sexual assault, etc? Renters without a vehicle may also consider how convenient it is to purchase essential goods like groceries. Renters with children may consider the education system in a neighbor. Are there quality schools within walking distance for their children, or will their children have to take a 45 minute bus ride to school elsewhere? In addition, renters may also consider how a neighborhood fit in with their lifestyle. Are there a lot of “mom's and pop's” stores around, or are there a lot of bars and night clubs? Can they eat out at local restaurants, or will they have to primarily cook and eat at home? Such environmental factors about a neighbor are often overlooked by current solutions.

Objectives

The goal of the current project is to create a hybrid mobile application that provides a neighborhood profile along with standard information to help renters find a suitable housing. Like current solutions, registered users will be able to post rental housing that include images and information about unit type (e.g., apartment, studio, room, etc.), location, prices, and any additional information pertinent to the unit. Unlike current solutions, the current application will allow any users to view a neighborhood profile for a selected rental unit. This neighborhood profile shall include recent crimes, information about nearby grocery stores and schools. The application will also provide a street view to allow users to tour the neighborhood without being there physically.

Relevance and Significance

As demands for rental housing increases, there will be more needs for better tools to help renters find suitable homes. Moreover, the growth of web-capable devices in the last decade have lead more people to take their “apartment hunt” to their smart devices. The current hybrid mobile application is relevant because it leverages the growing trend in mobile usage to address growing demands in the rental housing market.

Furthermore, the application is significant because it provides a solution not yet provided by any known rental search tools. Current tools typically only provide information specific to the rental unit. They do not provide any information about the surrounding neighborhood. Nevertheless, the neighborhood is an important factor when considering where to live. Often, one would have to drive to the property to survey the surrounding neighborhood. This is not always feasible however, if one currently live in a different city or state. For example, college students who attend school out of town are not able to easily visit the rental unit and survey the neighborhood before they move there. Also, those who have to relocate to a different city for work also face the same challenge. It is undesirable to be bound to a long contract in a neighborhood that is unsafe and inconvenient. It is possible to find neighborhood specific information by searching various sources such as the city's police department for crime rates, the local school district website for information about schools, and Yelp for local businesses. However, this is very inconvenient and time consuming. The current application simplifies and expedite this process by providing the above information in one convenient place.

Functional Requirements

The current application the following functional requirements :

- Public and registered users shall be able to perform a general search for rental housing based on zip code or city.
- Public and registered users shall be able to search for a listing according to price, type, bedrooms, bathrooms, and proximity to a target location.
- Public and registered users shall be able to view the location of a rental unit on a map.
- Public and registered users shall be able to view a crime heatmap for a specified location.
- Public and registered users shall be able to view posts containing details about a rental unit.
- Public and registered users shall be able to view a neighborhood profile for a given rental unit.
- Neighborhood profiles shall include recent crime data, ratings for local grocery stores and schools along with their proximity to the rental unit, and a street view of the surrounding neighborhood.
- Registered users shall be able to post renting housing containing relevant information about a unit they wish to rent out.
- Registered users shall be able to delete their post.
- Registered users shall be able to include images in their posts directly from their mobile device.

Use Cases

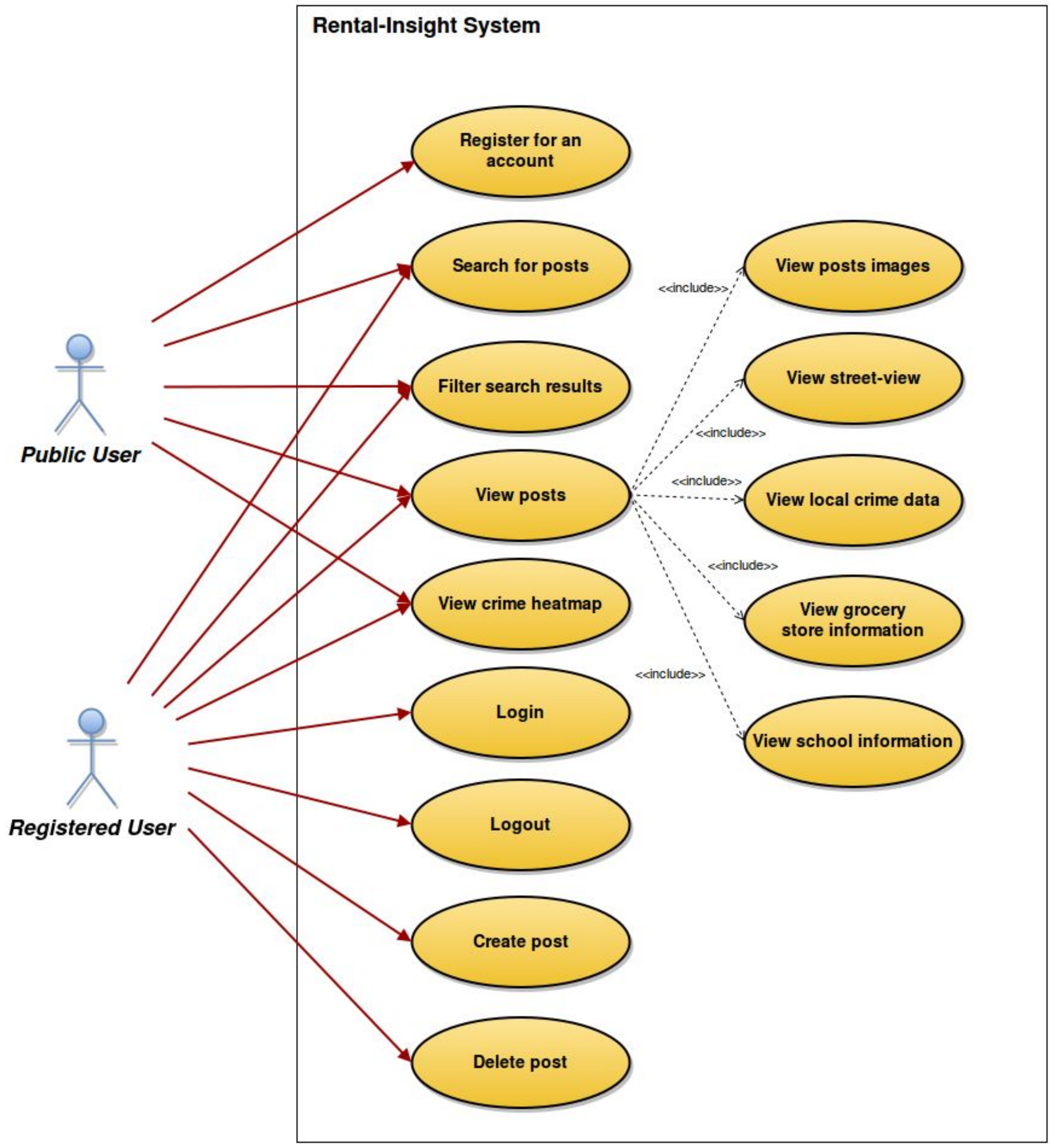


Figure 1. Usecase Diagram

Figure 1 shows the main use cases for the current system. Public users are only allowed to view data in the system, that is, they may only search and view rental unit information. They may also only view a crime heatmap for a specified location. In order to add or modify data in the system, they must first register for an account. In addition to what public users are allowed to do in the system, registered users may also create and remove posts. The following are further details about each use cases.

UC01: Register for an Account

Use Case:	Register for an account.
Actors:	Public user

Pre-conditions		Internet access exist
Post-conditions:		Actor successfully registered for an account and becomes a registered user.
Use Case Main Scenario (Basic Flow)		
Steps	**Actor Actions**	**System Responses**
1.	User clicks on “menu” button.	
2.		System displays side menu.
3.	User clicks on “Login” button.	
4.		System displays login form.
5.	User clicks on “Create New Account” link.	
		System displays registration form.
6.	User enters their email.	
7.	User enters their password.	
8.	User clicks “Register” button.	
9.		System register the account displays success message.
Extensions (Alternative Flows)		
6.a	User enters invalid email.	
6.a.1		System highlight email field and displays error message “invalid email”.
7.a	User enters a password that is too short.	
7.a1		System highlight password field and displays error message “password too short”.
8.a	User does not enter email and or password.	
8.a.1		System displays “missing fields” error message and highlights required fields.
8.b	User enters an email that already exist in the system.	
8.b.1		System displays “email is already taken” error message.

Special Requirements:	
Open Issues:	

UC02: Login

Use Case:		Log into an account
Actors:		Registered user
Pre-conditions		UC01
Post-conditions:		Actor successfully log into their account.
Use Case Main Scenario (Basic Flow)		
Steps	**Actor Actions**	**System Responses**
1.	User clicks on “menu” button.	
2.		System displays side menu.
3.	User clicks on “Login” button.	
4.		System displays login form.
5.	User enters their email.	
6.	User enters their password.	
7.	User clicks “Login” button.	
8.		System logs user into their account and redirect user back to homepage.
9.		System displays success message.
Extensions (Alternative Flows)		
7.a	User enters invalid email password combination.	
7.a.1		System displays “invalid email-password combination” error.
Special Requirements:		None
Open Issues:		None

UC03: Create Post

Use Case:		Create a post
Actors:		Registered user
Pre-conditions		UC01, UC02
Post-conditions:		Actor successfully created a post.
Use Case Main Scenario (Basic Flow)		
Steps	**Actor Actions**	**System Responses**
1.	User clicks “menu” button.	
2.		System displays side menu.
3.	User clicks on “Post” button.	
4.		System displays post form.
5.	User enters title.	
6.	User enters street of rental unit.	
7.	User enters city of rental unit.	
8.	User enters state of rental unit.	
9.	User enters zip code of rental unit.	
10.	User enters rent amount for rental unit.	
11.	User enters a phone number for contact.	
12.	User enters an email for contact.	
13.	User selects a date when the rental unit will be available.	
14.	User selects unit type.	

15.	User selects the number of bedrooms in the unit.	
16.	User selects the number of bathroom in the unit.	
17.	User enters in a description of the unit and any other additional details.	
18.	User clicks “submit” button	
19.		System stores post to database and displays success message.
Extensions (Alternative Flows)		
18.a	User add images by taking a picture using device camera.	
18.a.1	User clicks “camera” button.	
18.a.2		System displays device camera interface.
18.a.3	User clicks on “capture” button	
18.a.4		System append image to post data and redisplay camera interface.
18.a.5	User clicks “close” button	
18.a.6		System closes camera interface and redisplay post form.
18.b	User add images from device storage.	
18.b.1	User clicks on “upload” button	
18.b.2		System displays device album navigator interface.
18.b.3	User navigates through albums and selects desired images.	
18.b.4		System append image to post data and redisplay album interface.

18.b.5	User clicks "close" button	
18.b.6		System closes album interface and redisplay post form.
18.(a or b).3.1	User exceed maximum 6 images limit	
18.(a or b).3.2		System display "6 image maximum" error messsage.
18.c	User does not enter all required fields (title, street, city, state, zip code, bedrooms, bathroom, rent, availability date, type, and description)	
18.c.1		System displays "one or more required field is missing" error message.
Special Requirements:		Device must have camera hardware in order to include image from camera.
Open Issues:		None

UC04: Delete Post

Use Case:		Delete a post
Actors:		Registered user
Pre-conditions		UC01, UC02, UC03
Post-conditions:		Actor successfully deletes a post.
Use Case Main Scenario (Basic Flow)		
Steps	**Actor Actions**	**System Responses**
1.	User clicks "menu" button.	
2.		System displays side menu.
3.	User clicks on "My Posts" button.	
4.		System displays user's post.
5.	User clicks "Delete" button next to target post.	
6.		System displays confirmation modal.

7.	User clicks "OK."	
8.		System closes confirmation modal.
9.		System removes post from view and database and displays success message.
Extensions (Alternative Flows)		
7.a	User changed their mind about deleting post.	
7.a.1	User clicks "cancel" button.	
7.a.2		System closes confirmation modal and ignores delete request.
Special Requirements:		None
Open Issues:		None

UC05: Search

Use Case:		Search for a rental unit.
Actors:		Public or registered users.
Pre-conditions		Actor is on the homepage.
Post-conditions:		Rental units are displayed on a map relative to the location specified by the user.
Use Case Main Scenario (Basic Flow)		
Steps	**Actor Actions**	**System Responses**
1.	User input zipcode or city into search bar.	
2.		System displays all rental units within a 20 mile radius of specified location on a map.
Extensions (Alternative Flows)		
1.a	User does not specify any location.	
1.a.1		System displays error message "query cannot be empty."
2.a	There are no results for the specified location	

2.a.1		System display error message “no results found”
Special Requirements:		None
Open Issues:		None

UC06: Filtered Search

Use Case:		Search for rental units based on filter parameters.
Actors:		Public or registered users.
Pre-conditions		Actor is on the homepage.
Post-conditions:		Rental units matching the filter parameters are displayed on a map on the homepage according to the specified location.
Use Case Main Scenario (Basic Flow)		
Steps	**Actor Actions**	**System Responses**
1.	User clicks on “Filter” button.	
2.		System displays filter form.
3.	User enters required zipcode or city.	
4.	User enters optional price range.	
5.	User selects optional search radius.	
6.	User selects optional unit type.	
7.	User selects optional number of bedrooms.	
8.	User selects optional number of bathrooms.	
9.	User clicks “Search” button.	
10.		System directs user back to map on homepage.
11.		System displays all rental units matching the specified filter parameters on a map
Extensions (Alternative Flows)		

3.a	User does not enter zipcode or city.	
3.a.1	User clicks “Search” button.	
3.a.2		System displays error message, “A zipcode or city is required.”
3.a.3		System highlights required location field.
11.a	There are no results matching the specified parameters.	
11.a.1		System displays “no results found” error message.
Special Requirements:		None
Open Issues:		None

UC07: View Crime Heatmap

Use Case:		View crime heatmap
Actors:		Public or registered user.
Pre-conditions		UC05 or UC06 User is on the homepage
Post-conditions:		User successfully able to view crime heatmap for a given location.
Use Case Main Scenario (Basic Flow)		
Steps	**Actor Actions**	**System Responses**
1.	User clicks on “crime” button.	
2.		System displays a crime heatmap overlay on search result map.
Extensions (Alternative Flows)		
None		
Special Requirements:		None
Open Issues:		None

UC08: View Post

Use Case:		View post from search result
Actors:		Public or registered user
Pre-conditions		UC05 or UC06
Post-conditions:		User is able to view post details.
Use Case Main Scenario (Basic Flow)		
Steps	**Actor Actions**	**System Responses**
1.	User clicks on a marker representing a rental unit on the main map.	
2.		System displays an information window containing quick details about the rental unit's rent, bedroom/ bathroom capacity, and address.
3.	User clicks on the modal.	
4.		System directs the user to a different page containing additional information about the rental unit.
Extensions (Alternative Flows)		
3.a	User wish to select a different unit.	
3.a.1	User clicks anywhere on the map	
3.a.2		System closes the information window.
3.a.3	User clicks on a different marker	
Special Requirements:		None
Open Issues:		None

UC09: View Post Images

Use Case:	View Post Images
Actors:	Public or registered user
Pre-conditions	UC08 Post contains 1 or more images.

Post-conditions:		User is able to view enlarged images
Use Case Main Scenario (Basic Flow)		
Steps	**Actor Actions**	**System Responses**
1.	User clicks on image slideshow.	
2.		System displays slides of enlarged images.
Extensions (Alternative Flows)		
None		
Special Requirements:		Device must have touch screen in order to navigate between images.
Open Issues:		None

UC10: View Street-view

Use Case:		View rental unit’s street-view
Actors:		Public or registered user
Pre-conditions		UC08
Post-conditions:		User is able to see the street-view where the rental unit is located.
Use Case Main Scenario (Basic Flow)		
Steps	**Actor Actions**	**System Responses**
1.	User swipes on the street-view panel.	
2.		System updates the street-view according to the new orientation.
Extensions (Alternative Flows)		
1.a	User wish to view the street-view in fullscreen.	
1.a.1	User clicks on the “expand” button.	
1.a.2		System expands the street-view to fullscreen.
1.a.3	User	

1.a.4	User swipes in desired direction on the screen.	
1.a.5		System updates the street-view according to the new orientation.
Special Requirements:		Device must have touch screen.
Open Issues:		None

UC11: View Local Crimes

Use Case:		View rental unit's local crimes
Actors:		Public or registered user
Pre-conditions		UC08
Post-conditions:		User is able to view local crime data.
Use Case Main Scenario (Basic Flow)		
Steps	**Actor Actions**	**System Responses**
1.	User clicks on a pie chart slice.	
2.		System displays the crime charge description and the number of occurrences in the last 30 days.
Extensions (Alternative Flows)		
None		
Special Requirements:		None
Open Issues:		None

UC12: View Local Grocery Store Information

Use Case:	View local grocery store information
Actors:	Public or registered user
Pre-conditions	UC08
Post-conditions:	User is able to view information about local grocery stores.
Use Case Main Scenario (Basic Flow)	

Steps	Actor Actions	System Responses
1.	User clicks on a marker on the “grocery map”	
2.		System displays information window containing the store’s name, rating, and distance to the rental unit.
Extensions (Alternative Flows)		
2.a	User wishes to view additional information beyond what was provided in the information window.	
2.a.1	User selects/clicks on a grocery store from the “grocery stores list”	
2.a.2		System displays an in-app browser connecting to the store’s Yelp page.
Special Requirements:		None
Open Issues:		None

UC13: View Local School Information

Use Case:		View local school information
Actors:		Public or registered user
Pre-conditions		UC08
Post-conditions:		User is able to view information about local schools.
Use Case Main Scenario (Basic Flow)		
Steps	**Actor Actions**	**System Responses**
1.	User clicks on a marker on the “school map”	
2.		System displays information window containing the school’s name, rating, and distance to the rental unit.
Extensions (Alternative Flows)		
2.a	User wishes to view additional information beyond what was	

	provided in the information window.	
2.a.1	User selects/clicks on a school from the “school list.”	
2.a.2		System displays an in-app browser connecting to the school’s Yelp page.
Special Requirements:		None
Open Issues:		None

UC14: Logout

Use Case:		User logs out
Actors:		Registered user
Pre-conditions		UC02
Post-conditions:		User successfully logs out.
Use Case Main Scenario (Basic Flow)		
Steps	**Actor Actions**	**System Responses**
1.	User clicks on “menu” button.	
2.		System displays side menu.
3.	User clicks “logout” button.	
4.		System redirect user to homepage and display success message.
Extensions (Alternative Flows)		
None		
Special Requirements:		None
Open Issues:		None

Development Environment

Hardware

Device	Yoga 2 Pro
Memory	7.7 GB
Processor	Intel® Core™ i5-4200U CPU @ 1.60GHz × 4
Graphics	Intel® Haswell Mobile
Disk	235.2 GB

Software

Operating System	Ubuntu 14.10 (64-bit)
Languages	JavaScript, HTML5, CSS
Frameworks/ Libraries	Ionic, AngularJS, Node.js, Geofire, Cordova
Database	Firebase

The application was developed on Ubuntu 14.10 operating system using Ionic and AngularJS framework. Ionic is a powerful HTML5 framework that helps developers build native-feeling hybrid mobile applications using HTML, CSS, and JavaScript. Its primary benefit is the simplification of front-end user interface development. AngularJS is a framework that helps manage views and models. The main benefit of AngularJS two-way data binding, that is, the view will automatically gets updated when the model change, and the model will automatically gets updated when the view change. Firebase is a real-time cloud platform that allows applications to store and sync data instantly across all devices. It also provides authentication services, which simplifies the registration/ login process. Such features greatly expedite back-end development and increases scalability. Geofire is a Javascript library that works with Firebase to provide geofencing services. Cordova is a wrapper that allows applications built using web technologies to run natively on mobile devices.

Software Design

Context Diagram

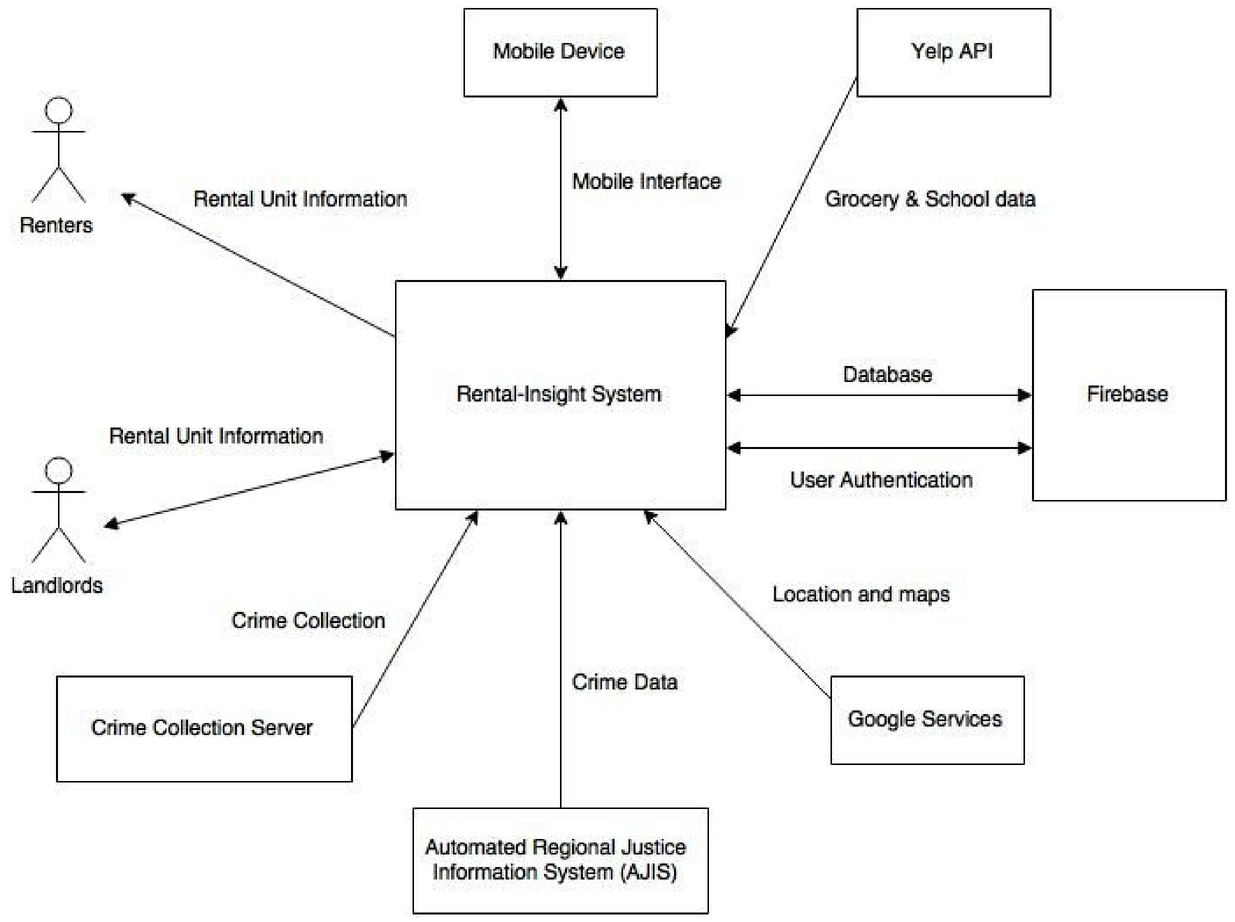


Figure 2. Context Diagram

The context diagram depicted in Figure 2 shows a high-level view of the current system. This diagram shows the boundaries and scope of the system. The following element catalog explains the role of each element and their relationship to the current system.

Element Catalog

Rental-Insight System: The system built to provide services to renters and landlords as specified in the functional requirements.

Renter: A public user of the system. This user will consume information and services provided by the system.

Landlord: A registered user of the system. This user provides rental unit information to the system.

Mobile Device: A web-enabled device with touch screen and camera capability such as a smartphone or tablet. This device serves as the user's main interface to the system.

Yelp API: An external service that provides information about grocery stores and schools.

Firebase: An external platform that serves as the backend of the system and a real-time database. This allow all users to have access to the same data in real-time. This platform is also responsible for user authentication.

Google Services: An external collection of services that provides map related services such as geolocation coordinates and map visualization. This collection also provides the streetview interface.

Automated Regional Justice Information System (AJIS): An external system that provides crime data in the San Diego region.

Crime Collection Server: An external server that is responsible for collection crime data from AJIS and storing it to Firebase.

Domain Model

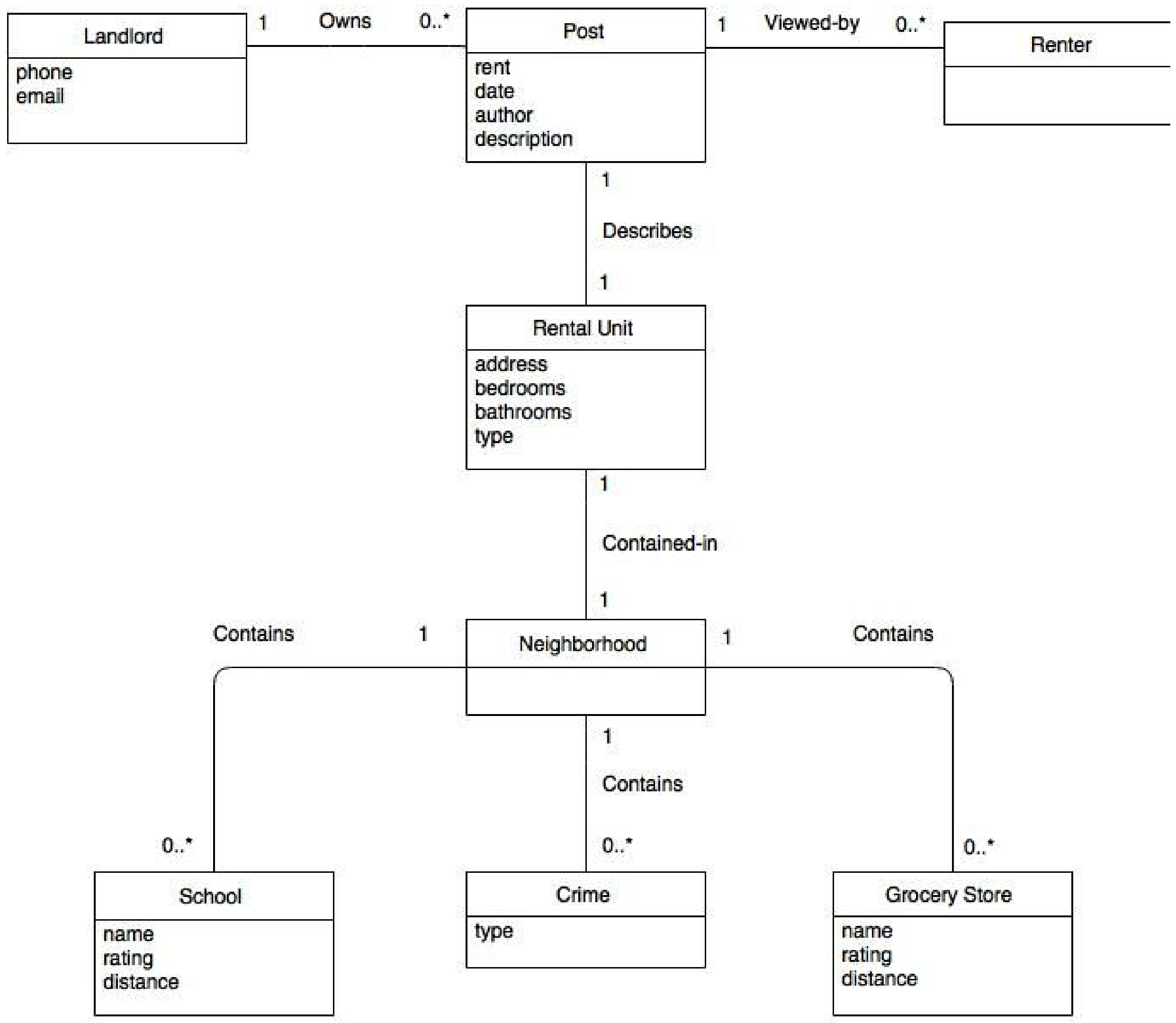


Figure 3. Domain Model

The domain model shown in Figure 3 is a conceptual representation of various real-world objects and concepts that compose the system. In the system, there will be landlords that will post rental listings to the system, where each post belongs to exactly one landlord. A post may be viewed by many renters, people who seek to rent out a rental unit. Also, each post will describe exactly one rental unit. A rental unit is contained in a neighborhood, a geographical area defined by a one mile radius around a rental unit. Each neighborhood may contain zero to many schools, grocery stores, and crime. This model is useful in understand the boundaries of our system and its major entities. It lays the foundation for the software classes developed.

Architectural Pattern

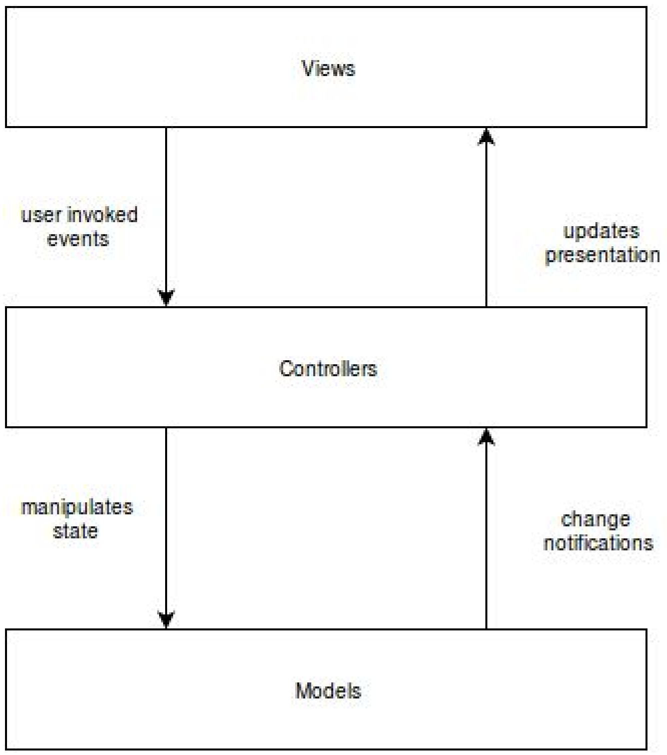


Figure 4. MVC Architectural Pattern

The current application was built following the MVC architectural pattern as shown in Figure 4. The figure shows a high-level view of the main types of components in the system and how they work together. As shown, the pattern decomposes the software into three main types of components -- models, views, and controllers. Each component is responsible for a different aspect of the software. View components are user interfaces that are responsible for accepting user inputs and displaying the state of various models. Model components are representations of the application state or data. It contains logic specific to the application. Controller components are responsible for facilitating interactions between views and models. They translate user actions into changes in models, or changes in models into changes in views. Controllers contain the business logic of the application. A key reason why this pattern was chosen is because it allows for separation of responsibilities. This approach allow the application can be developed in a modular way. Future changes to one area of the application would have minimal effects on other area of the application.

Class Diagram

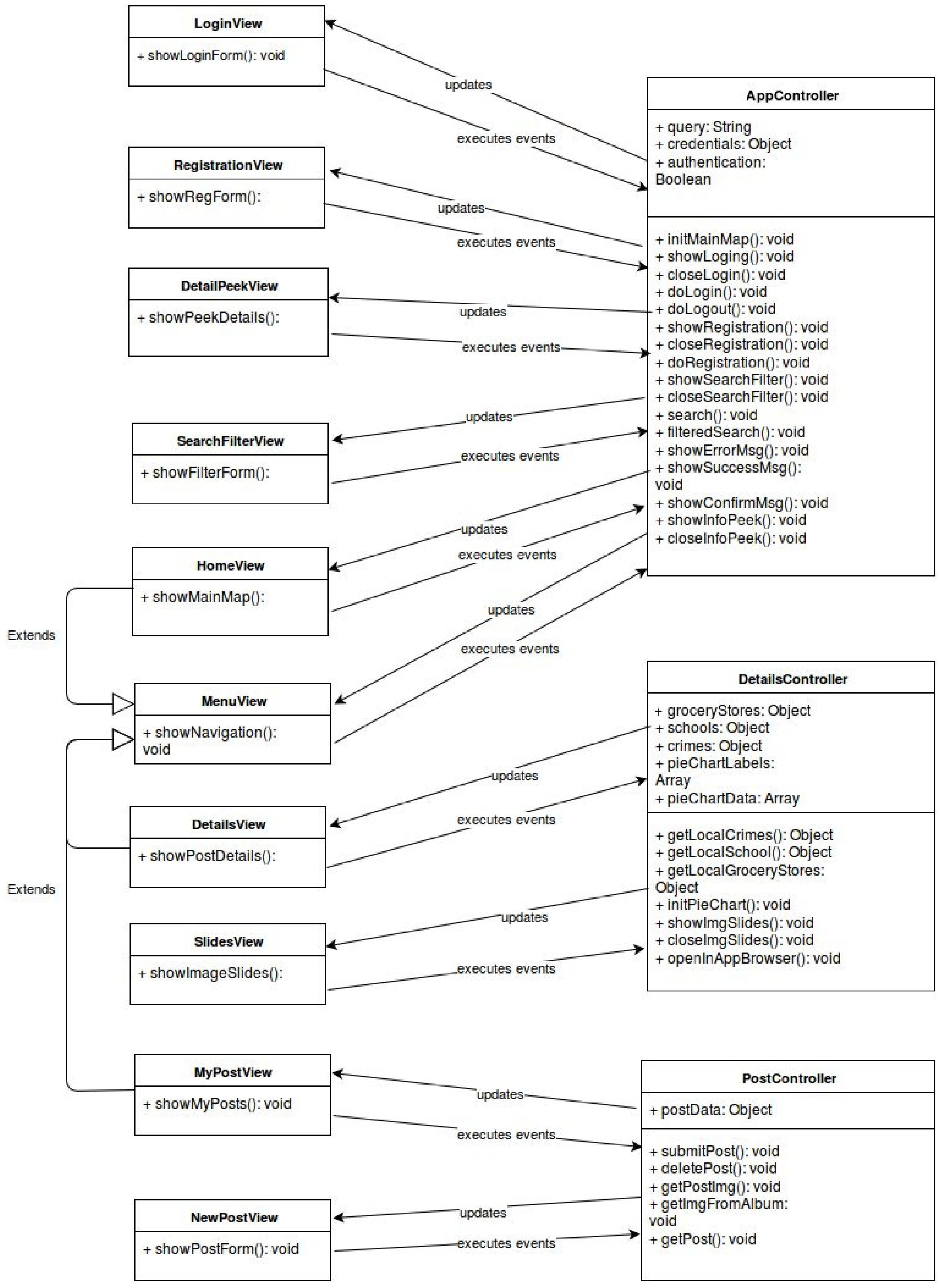


Figure 5. Class Diagram: Views and Controllers

Figure 5 is a software class diagram which shows responsibilities assigned to each view and controller components, as well as, their relations to one another. As described in the architectural pattern depicted in Figure 4, views execute events triggered by the user and controllers will update the views based on changes in the application's states and data. Figure 5 shows that HomeView, DetailView, and MyPostView extends the MenuView. This indicate that those views are constructed from the MenuView template, that is, users will have access to the main menu from views that extends MenuView.

Moreover, views are assigned to be handled by a controller based on related business logic. For example, AppController is the main controller responsible for general business logic that supports the main goal of the application. Thus, it handles events and presentation for LoginView, RegistrationView, DetailPeekView, SearchFilterView, HomeView, and MenuView. PostController is responsible for handling business logic directly related to posts. Events invoked within the NewPostView and MyPostView are sent to PostController for processing. DetailsController is responsible for presenting details about a rental unit. This controller handles requests from DetailsView and SlidesView for information about a rental unit and its neighborhood.

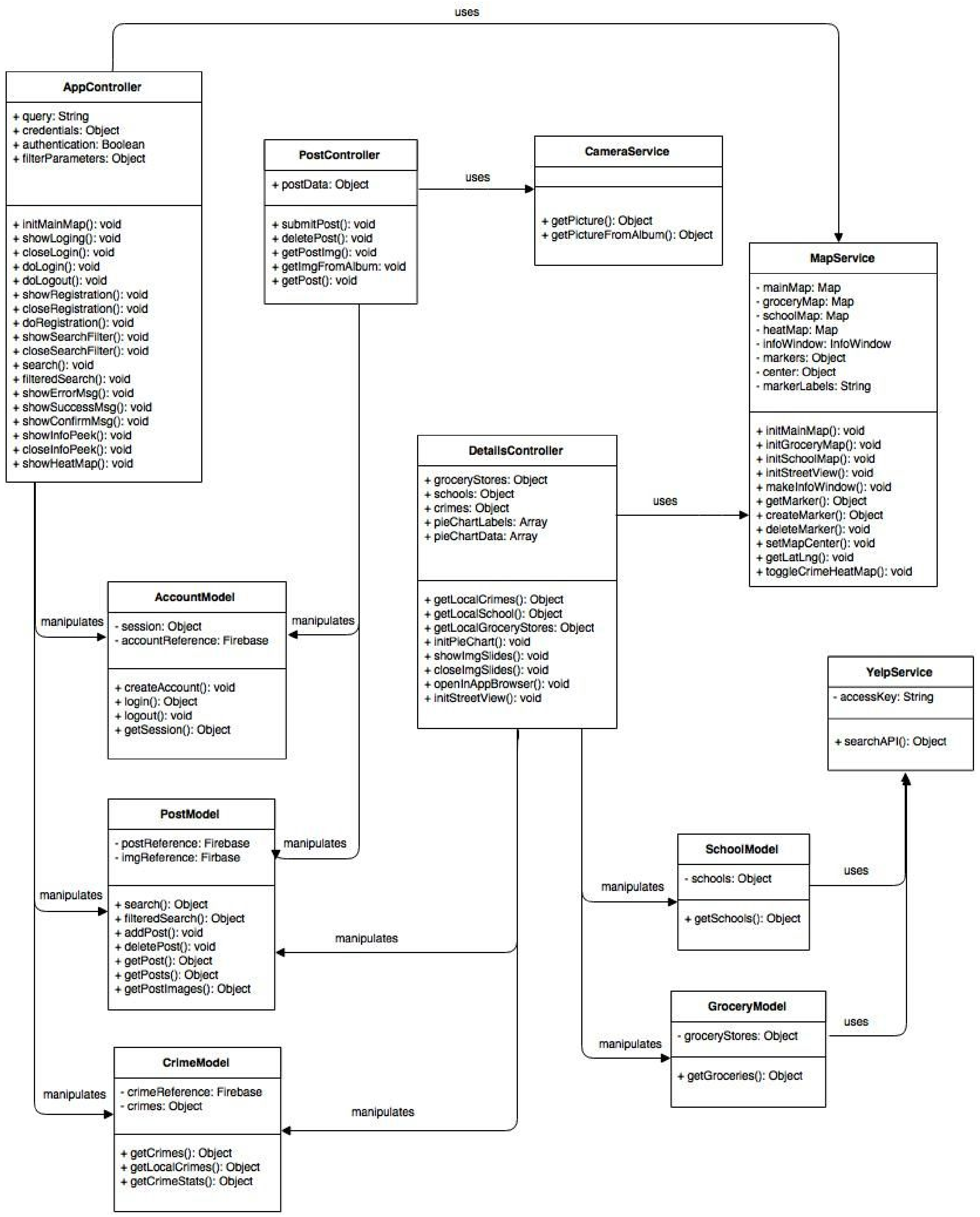


Figure 6. Class Diagram: Controllers, Models, and Services

Figure 6 is a software class diagram that shows the relations and responsibilities of our controllers, models, and services. AppController manipulates AccountModel, PostModel, and CrimeModel because it is responsible for account access, searches, and displaying the crime heatmap. It also uses the MapService to determine where to place search results on the main map and how to draw the crime heatmap overlay. PostController manipulates the PostModel because it is responsible for business logic directly related to a post. It also manipulates the AccountModel because it needs to specify who owns a post. It also needs to use the CameraService to allow users to attach images to their post. DetailsController works with PostModels, CrimeModel, SchoolModel, and GroceryModel to assemble a profile when a user wants to view information about a rental unit. SchoolModel and GroceryModel uses the YelpService in order to obtain the necessary data because those information are not stored in our system.

Data Model Diagram

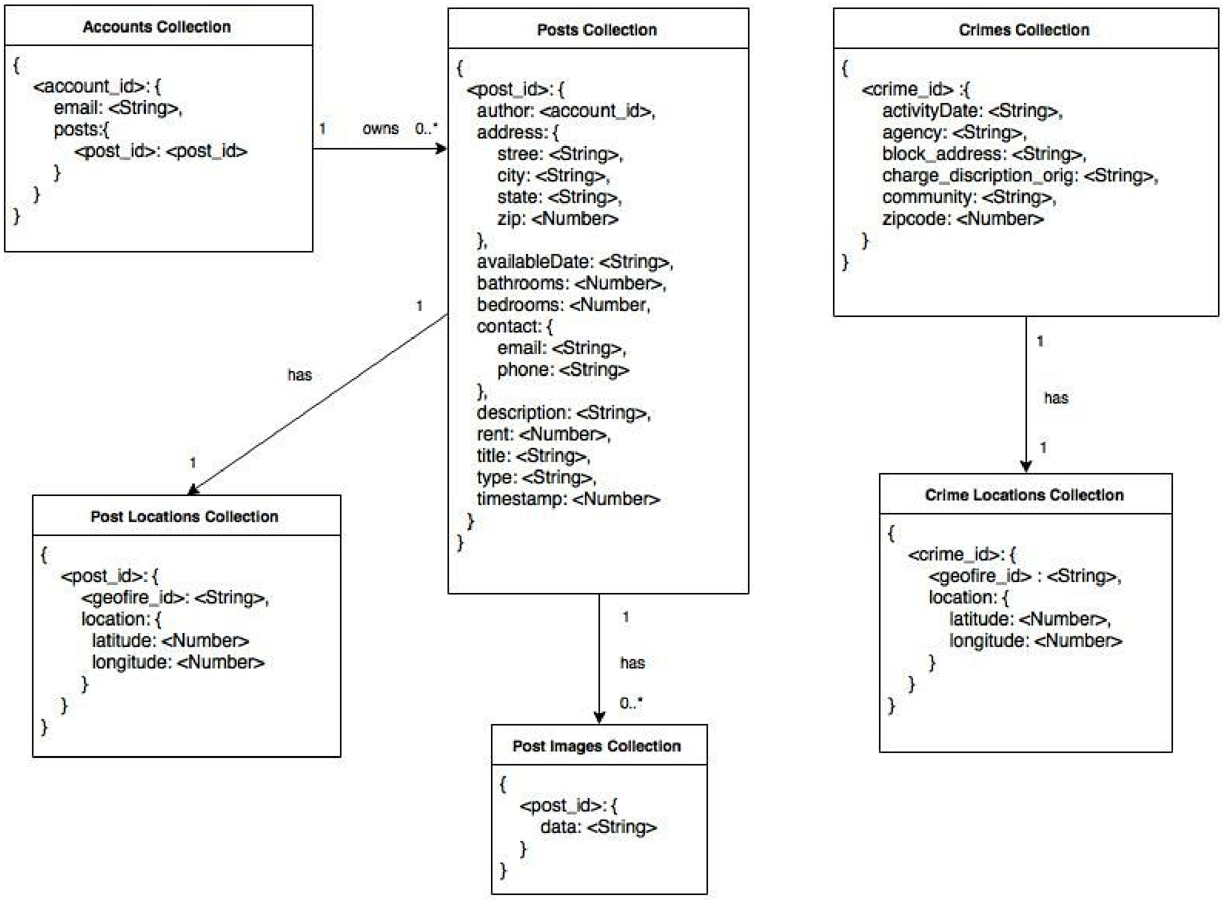


Figure 7. Firebase Data Model

The current application was built using Firebase's NoSQL database. Since Firebase is a NoSQL database, it does not have tables as seen in traditional Relational Databases (RDB). Firebase stores documents in collections, which are analogous to tables in RDB. Each document, which are analogous to rows or entries in RDB, are stored as a JSON-like object. Each document is an object defined by a set of key value pair. Values with angle brackets represent the expected data type. Keys with angle brackets represents unique keys generated by Firebase. Proper indexing of keys enable quick and efficient searches. Instead of looking through an entire collection for a document, the key tells us exactly where the document is located. For example, *"<post_id>"* is a unique key that Firebase generates whenever a post is added to the Post Collection. Using the same key in multiple collections tells us exactly where each related documents in across multiple collections. Figure 7 shows the schema for documents in the current application. A user represented by an Account document object may contain zero or more posts. A post may have zero more images attached. It will also be associated with exactly one location. Similarly, a crime will be associated with exactly one location.

Installation

Installing from Android Google Play Store

The current application can be easily installed on most Android devices from the Google Play store. Perform the following steps to install:

1. From your Android device, launch the “Play Store” application.
2. In the search field, search for “RentalInsight.”
3. Locate and click on the entry titled “RentalInsight” by “hchau619.”
4. Click “INSTALL” to install the application.
5. Locate the application drawer and click on the icon.

Only Android devices are supported at the moment. Support for iOS devices will come in future iterations.

Installing from Source Code

Linux installation:

1. Download the source code from GitHub.

   ```
   git clone https://github.com/hchau619/RentalInsight
   ```

2. Navigate to the folder we have just downloaded, “RentalInsight.”
3. Install Ionic and Cordova.

   ```
   npm install -g ionic cordova
   ```

4. Launch the application in the browser.

   ```
   ionic serve
   ```

5. If a browser does not automatically pop up, open a browser and navigate to *localhost:8100/#/app/home.*

User Manual

Register for an Account

Steps:

1. At the homepage as shown in Figure 8, click on the menu button on the top right corner to reveal the side menu.
2. Click the "LOGIN" button located on the side menu as shown in Figure 9.
3. Click the "Create New Account" link as shown in Figure 10.
4. Fill out the required fields on Registration page as shown in Figure 11.
5. Finally, click the "Register" button.

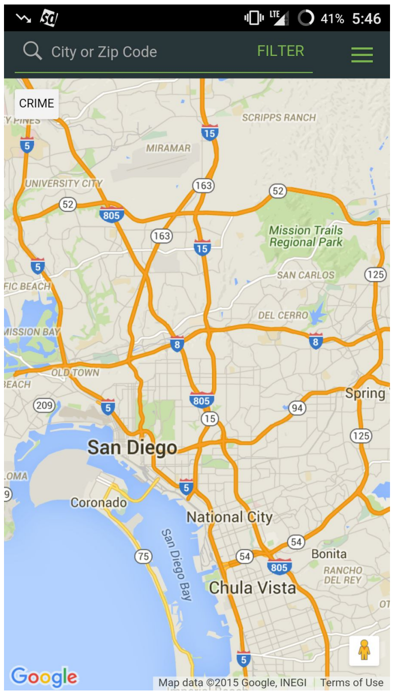


Figure 8. Homepage

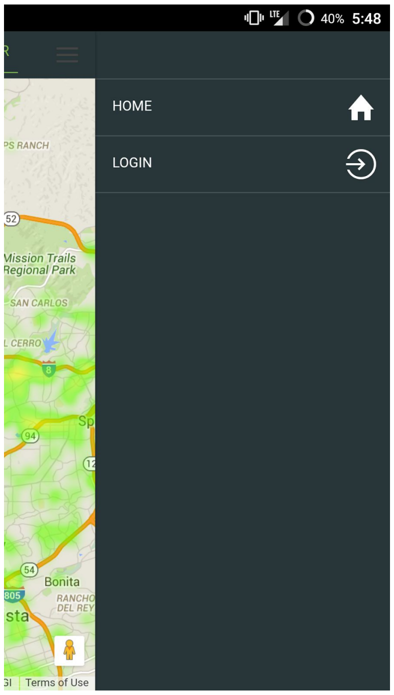
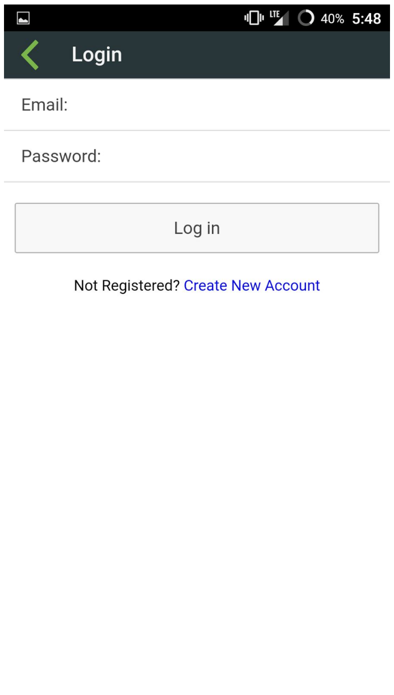


Figure 9. Public Side Menu



Not Registered? Create New Account

Figure 10. Login Page

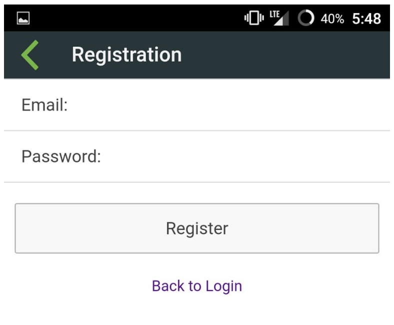
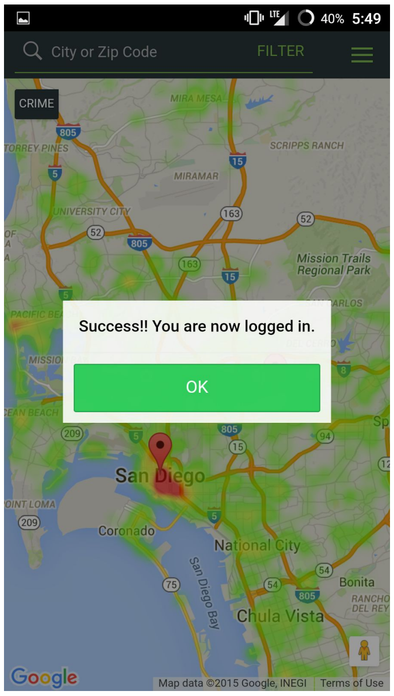


Figure 11. Registration Page

Login

Steps:

1. From the homepage as shown on Figure 8, click the menu button located in the top right corner.
2. Click the “LOGIN” button on the side menu as shown in Figure 9.
3. Enter a valid email and password in the available fields.
4. Finally, click the “Login” button. If successful, you will be directed back to the homepage with a success message as shown in Figure 12.



Success!! You are now logged in.

Figure 12. Login Success

Create a Post

Steps:

1. First, login with valid credentials you have created.
2. From the homepage, click the menu button located on the top right corner to reveal the registered user side menu as shown in Figure 13. Note that the registered user side menu is different from the public side menu.
3. Click the "POST" button located on the side menu to reveal the post form.
4. Fill out all fields shown on Figure 14, 15 and 16.
5. Optionally, you may attach up 6 images by using either the camera method or album method.
    a. Camera Method
        i. Click the "CAMERA" button as shown in Figure 16.
        ii. Your device's default camera application shall launch. This interface will vary depending on your device and the camera application installed.
        iii. Click the the capture button to take a picture.
        iv. Click the check button to save the image, else, click the "X" button to return to the post form.
    b. Album Method
        i. Click the "UPLOAD" button as shown in Figure 16.
        ii. Your device's default album navigator will launch. Figure 17 shows the interface for Android devices.
        iii. Find and click you the images you wish to include in the post.
        iv. Click the hardware back button to return to the post form.
6. Finally, click on the "SUBMIT" button as shown in Figure 16 to submit your post.

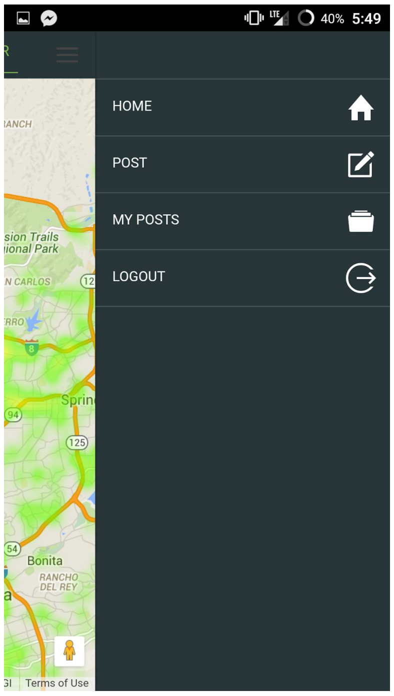


Figure 13. Registered User Side

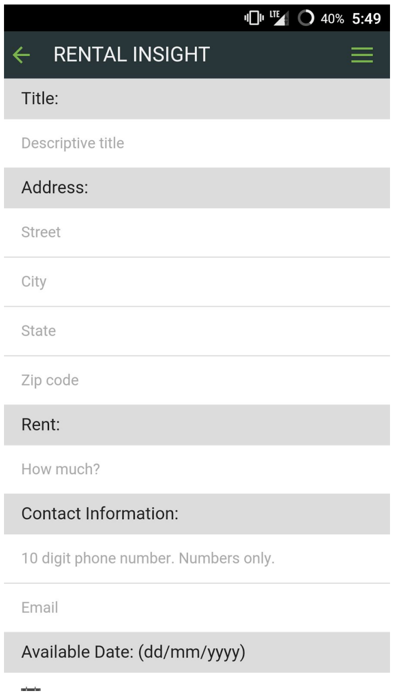


Figure 14. Post Form 1

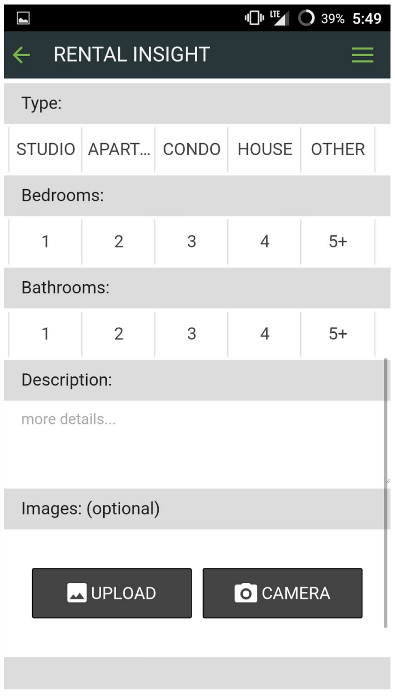


Figure 15. Post Form 2

Menu

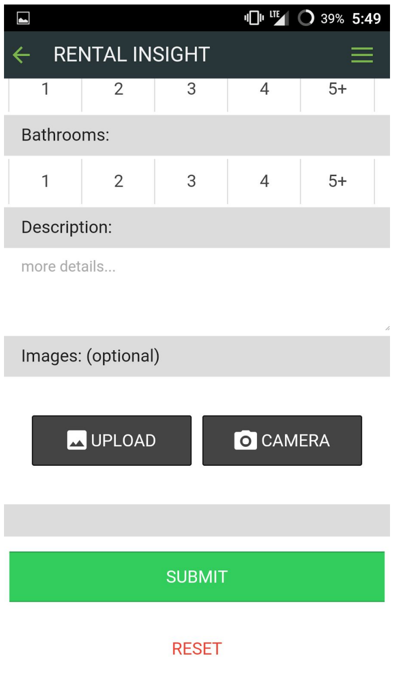


Figure 16. Post Form 3

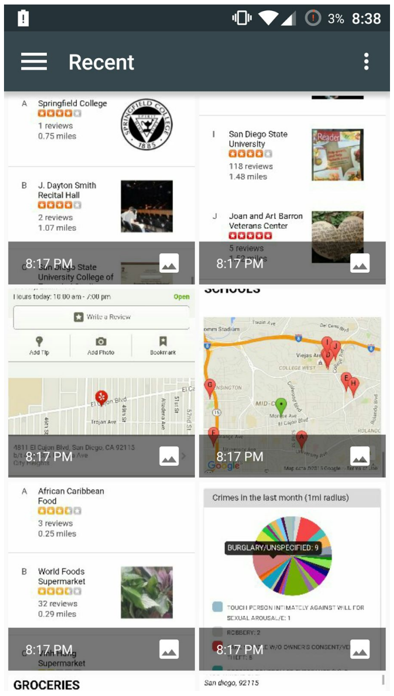


Figure 17. Android Album

Delete a Post

Note: User must have a valid account and have posted one more more post.

Steps:

1. Login into account with valid credentials.
2. From the homepage as shown in Figure 14, click the menu button in the top right corner.
3. Click the “MY POST” button as shown in Figure 13.
4. Click on the circular “X” button on the top right side of each card as shown in Figure 18.
5. Click “OK” to confirm delete as shown on Figure 19.

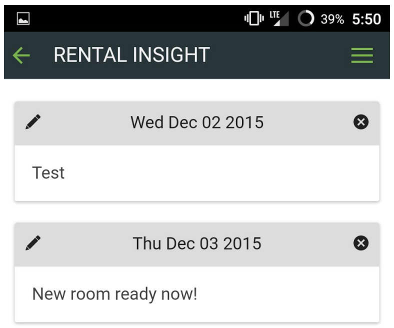


Figure 18. My Post Page

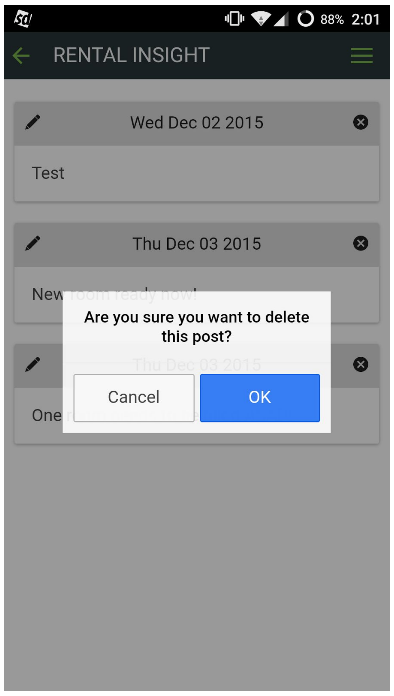


Figure 19. Confirm Delete

Search

Steps:

1. Enter a zip code or a city on the search bar located at the top of the homepage as shown in Figure 8.
2. Hit “enter” on your device virtual keyboard or click the magnifying glass located on the left of the search bar. Posts within a 20 mile radius of you queried location will appear on the main map as shown in Figure 20.

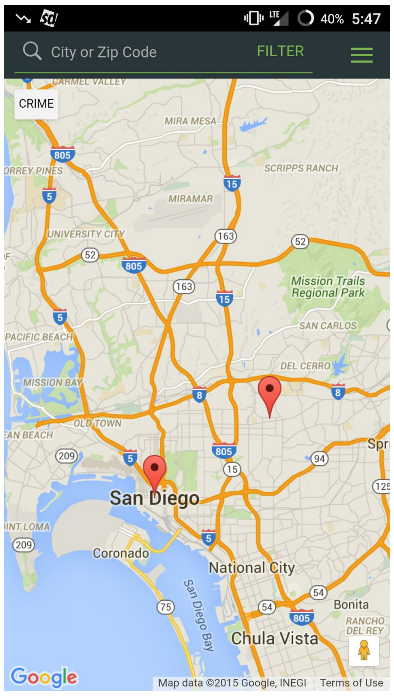


Figure 20. Search Results

Filtered Search

Steps:

1. Click on the “FILTER” button location at the top on the homepage as shown in Figure 8.
2. Enter in zip code or a city in the search bar located at the top of the filter form as shown in Figure 21.
3. Optionally, fill in remaining fields you wish to be used to filter results.
4. Finally, click the “Search” button located at the bottom of the filter form as shown on Figure 21.

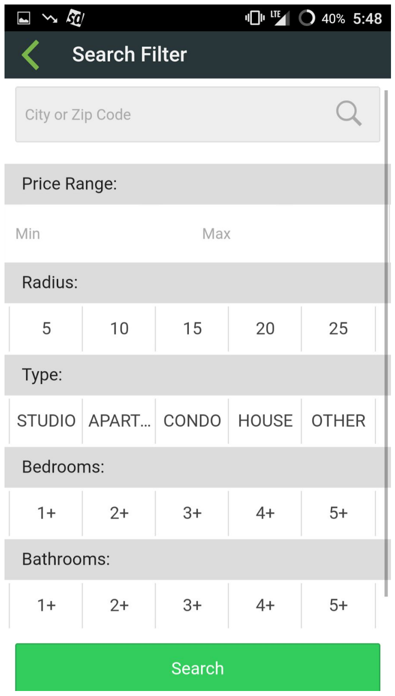


Figure 21. Search Filter Form

View Crime Heatmap

Steps:

1. Perform a search.
2. Click on the "CRIME" button located on the top left of the main map on the homepage as shown in Figure 20. Figure 22 shows a heatmap where areas that are red indicates high volume of crime and areas that are green indicates low volume of crime. Areas that are not colored has no reported crimes.

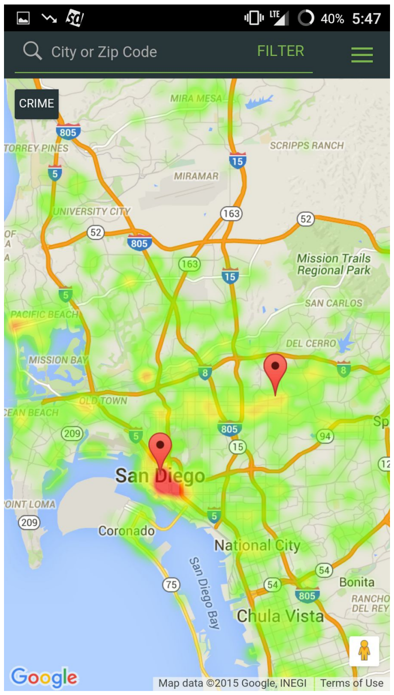


Figure 22. Crime Heatmap

View Post

Steps:

1. Perform a search.
2. Click on any marker on the main map located on the homepage as shown in Figure 20. A peek modal will appear at the bottom of the screen.
3. Click on the peek modal and be directed to that post’s detail page as shown in Figure 24.

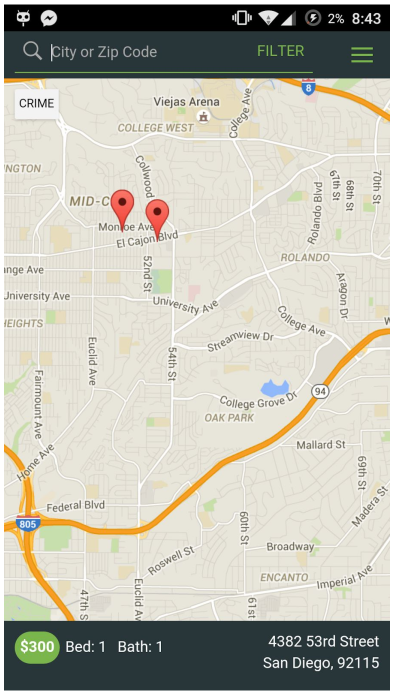
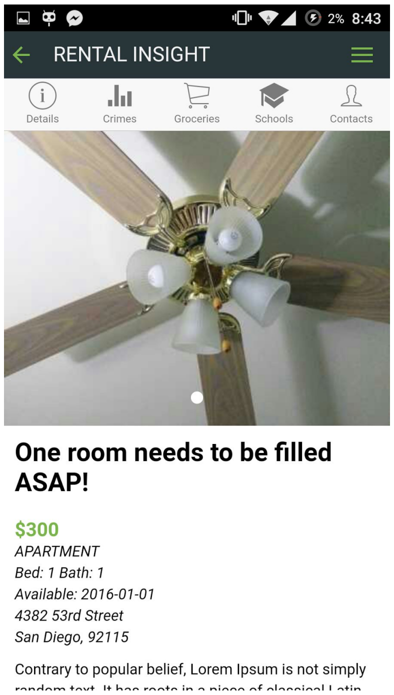


Figure 23. Peek Modal



One room needs to be filled ASAP!

$300
APARTMENT
Bed: 1 Bath: 1
Available: 2016-01-01
4382 53rd Street
San Diego, 92115

Contrary to popular belief, Lorem Ipsum is not simply

Figure 24. Post Detail Page

View Post Images

Note: Post selected must have one or more images.

Steps:

1. Perform View Post steps.
2. Click on the image slideshow located at the top of the Post Details Pages as shown in Figure 24. A new window will pop up containing enlarged images as shown in Figure 25.
3. Navigate between images by swiping left or right.
4. Click the back arrow button located at the top left corner to return to the Post Detail Page.

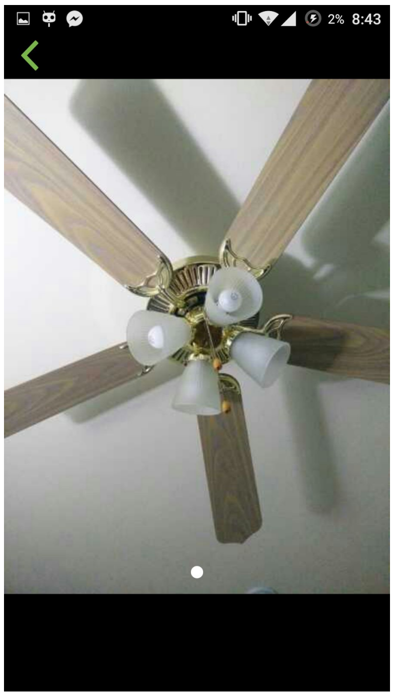
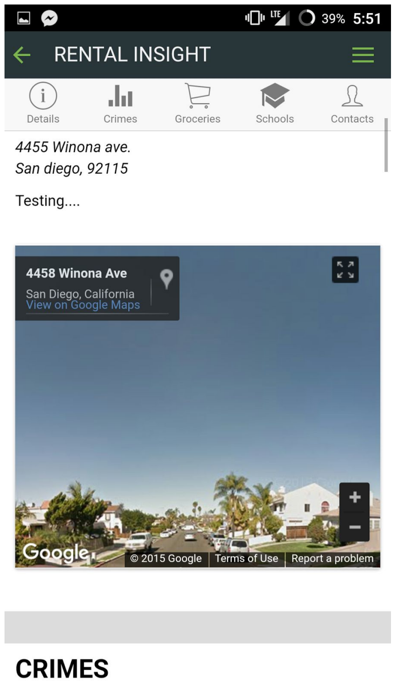


Figure 25. Image Slides

View Street View

Steps:

1. Perform View Post steps.
2. From the top of the Post Detail page, scroll down to the street view panel as shown in Figure 26.
3. Swipe in any direction within the panel to change the view orientation.
4. Optionally, click on the expand button located in the top right corner of the street view panel as shown in Figure 26 to go to full screen.



4455 Winona ave.
San diego, 92115

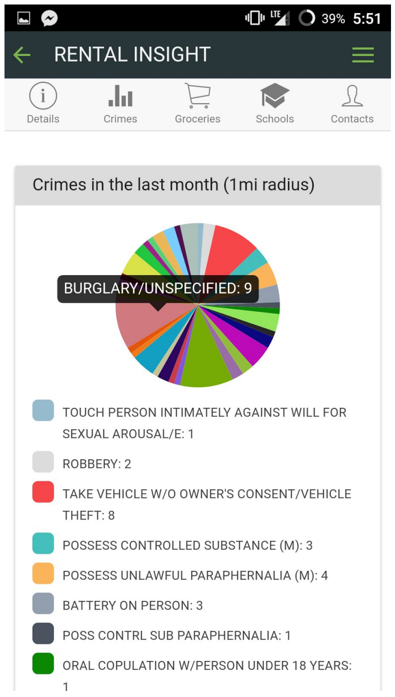
Testing....

Figure 26. Street View

View Local Crimes

Steps:

1. Perform View Post steps.
2. Scroll down to the Crime section or click on the “Crimes” tab in the top navigation.
3. Click anywhere on the crime pie chart to see various types of crimes and their number of occurrences in the past month as shown in Figure 27.



Crimes in the last month (1mi radius)

🟦 TOUCH PERSON INTIMATELY AGAINST WILL FOR SEXUAL AROUSAL/E: 1

⬜ ROBBERY: 2

🟥 TAKE VEHICLE W/O OWNER'S CONSENT/VEHICLE THEFT: 8

🟦 POSSESS CONTROLLED SUBSTANCE (M): 3

🟧 POSSESS UNLAWFUL PARAPHERNALIA (M): 4

⬜ BATTERY ON PERSON: 3

⬛ POSS CONTRL SUB PARAPHERNALIA: 1

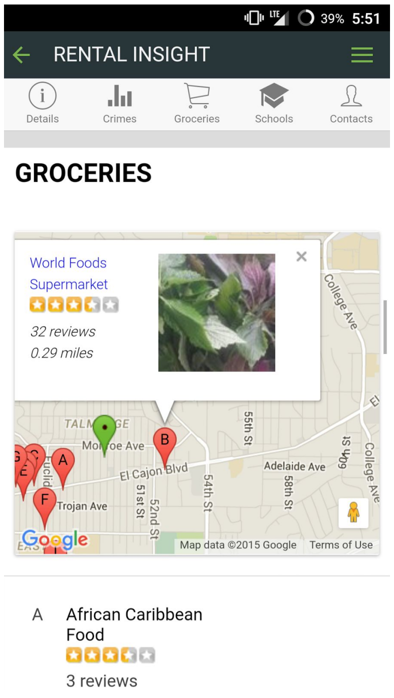
🟩 ORAL COPULATION W/PERSON UNDER 18 YEARS: 1

Figure 27. Crime Pie Chart

View Local Grocery Stores

Steps:

1. Perform View Post steps.
2. Scroll down to the Grocery section or click on the "Groceries" tab in the top navigation.
3. View grocery map.
    a. Click on any red marker to open an information window containing details about a grocery store as shown in Figure 28.
4. View grocery list.
    a. Click on any row to open an in-app browser which links to the store's Yelp page as shown in Figure 29 and 30.
    b. Close the in-app browser by using one of the following method:
        i. Clicking on the "X" button located on the top right corner as shown in Figure 30.
        ii. Clicking the hardware back button.

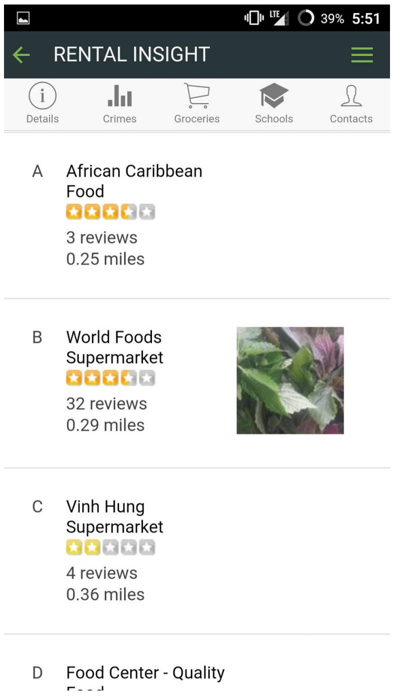


GROCERIES

World Foods Supermarket
32 reviews
0.29 miles

A African Caribbean Food
3 reviews

Figure 28. Grocery Map



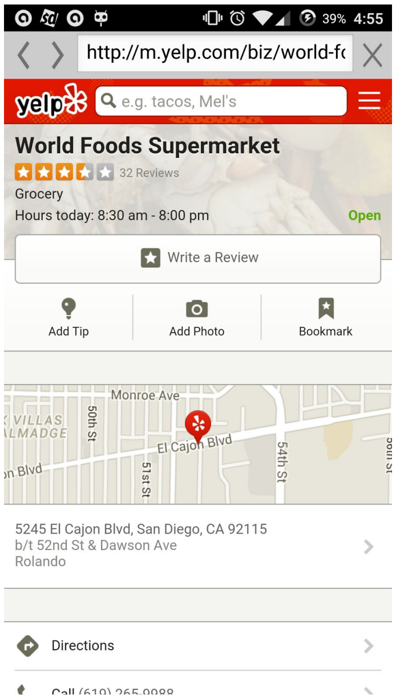
A African Caribbean Food
3 reviews
0.25 miles

B World Foods Supermarket
32 reviews
0.29 miles

C Vinh Hung Supermarket
4 reviews
0.36 miles

D Food Center - Quality

Figure 29. Grocery List



World Foods Supermarket
32 Reviews
Grocery
Hours today: 8:30 am - 8:00 pm Open

5245 El Cajon Blvd, San Diego, CA 92115
b/t 52nd St & Dawson Ave
Rolando

Figure 30. In-app Browser

View Local School

Steps:

1. Perform View Post steps.
2. Scroll down to the School section or click on the “Schools” tab in the top navigation.
3. View school map.
    a. Click on any red marker to open an information window containing details about a school as shown in Figure 31.
4. View school list.
    a. Figure 32 shows the school list. Click on any row to open an in-app browser which links to the school’s Yelp page as shown in Figure 30.
    b. Close the in-app browser by using one of the following method:
        i. Clicking on the “X” button located on the top right corner as shown in Figure 30.
        ii. Clicking the hardware back button.

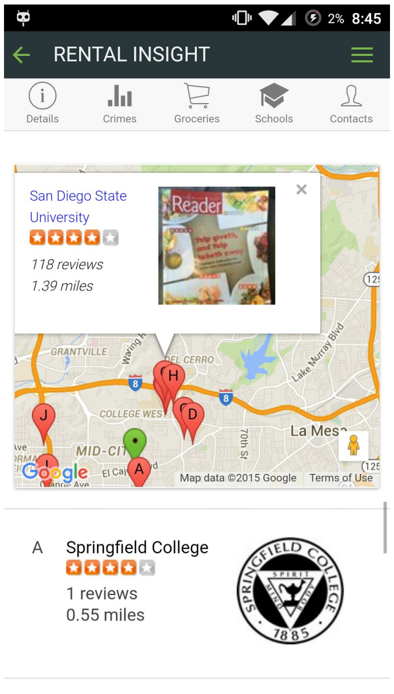


Figure 31. School Map

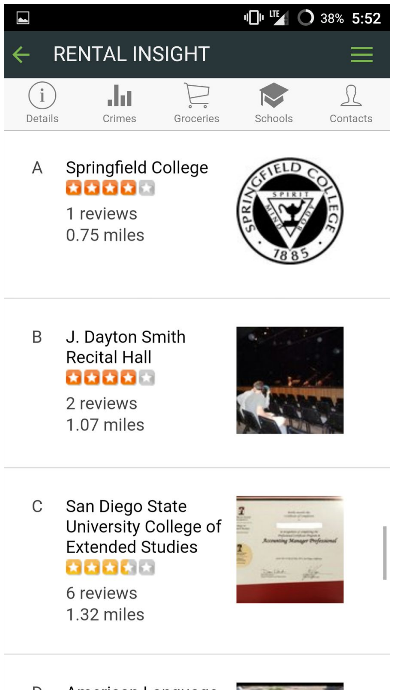


Figure 32. School List

Logout

Note: User must be logged in already.

Steps:

1. From the homepage, click on the menu button location in the top right corner to reveal the side menu as shown in Figure 13.
2. Click the “LOGOUT” button.

Discussion

Technologies

Developing Rental-Insight was a challenging task. The application was built using some of the latest technologies in hybrid application development. Hybrid applications differ from native applications in that they uses web technologies like Javascript, HTML, and CSS instead of using platform specific languages — Java for android, or Objective C for iOS — to build applications for mobile devices. This approach was taken because it allows the current application to be developed once and run on both Android and iOS devices. Nevertheless, there are drawbacks to this approach due to the relative infancy of these new technologies.

Ionic is a new framework for hybrid application development that has gained a lot of popularity since its launch in 2013. As a new framework, there are still some issues with some of its components. For example, performance issues are visible when transitioning from one page to the next. Moreover, documentation for the framework is also lacking. However, it does possess a large community of developers that provide technical support through forums.

Firebase is another new technology used to build the current application. Launched in 2012, the cloud-based platform provides backend services such as user authentication and real-time database. The real-time nature of Firebase is useful in the current project because it allows for a post to be immediately available to all users in the system when it is created. Users are able to get the latest data as soon as it enters the system. One feature that seems to be lacking with Firebase is the ability to search for a document based on multiple parameters. This was a challenge when implementing the filtered search functionality. Filtering had to be done client-side, which may affect performance if there are excessively large amount of posts.

Limitations and Future Works

To limit the scope of the current project, crime data services are limited to the San Diego region. Future work could expand crime coverage by collecting crime data from other cities. Moreover, a separate service can be develop to provide additional crime insights such as trends, comparisons, or prediction.

Furthermore, the neighborhood profile could also be improved. Future work could be to provide additional details about local businesses and neighborhood events on top of current features. Such information may be useful to help the user evaluate their lifestyle fit to a neighborhood.

Lastly, future work could also focus on collecting rental listings from other major platforms such as Craigslist, Apartment.com, Forrent.com, etc. Landlords with properties in crime stricken neighborhood may be less inclined to post on Rental-Insight. By aggregating listings from different sites into Rental-Insight, we give users a complete image of the rental market in a region with all the tools they will need to make an informed decision.

Conclusion

Renting a place to live could be a long and frustrating process. This may be especially true with renters who do not live in the same city as the unit the wish to rent. For example, students attending college in a different state, or a person who need to relocate for work. With Rental-Insight, users can explore their potential neighborhood without ever being there physically. As demands for rental housing increases, there is a real need for a solution that can simplify the searching process. With the latest technologies in hybrid mobile application, Rental-Insight provides proof of concept that a neighborhood profile can help simplify and expedite the search for rental housing. By providing a neighborhood profile, Rental-Insight is a solution that help renters make better and more informed renting decisions.

References

Callis, R., Kresin, M. (2015). Residential vacancies and homeownership in the fourth quarter 2014. U.S. Census Bureau, CB15-08. Retrieved from: http://www.census.gov/housing/hvs/files/currenthvs press.pdf

Goldfarb, Z. (2015). 8 things millennials want—and don't want—show how different they are from their parents. The Washington Post. Retrieved from: http://www.washingtonpost.com/blogs/wonkblog /wp/2015/02/28/8-things-millennials-want-and-dont-want-basically-anything-their-parents-wante d/

The Nation Association of REALTORS & Google. (2013). The digital house hunt: Consumer and market trends in real estate. Retrieved from: http://www.realtor.org/sites/default/files/reports/2013/Study-Digital House-Hunt-2013-01.pdf

Zillow. (2015). United States home prices & values. Retrieved from: http://www.zillow.com/home-values/