SW Engineering CSC648/848 Section 01 Fall2016

Gator Lodge

Team 7
Lijie Zhou (Team Lead)
Anna Sever(Tech Lead)
Theofanis Koutoulas(Front End)
Ivan Marchenko(Full Stack)
Logan Figgins(Back End)
Hao Xian Zheng(Back end)

"Milestone 2"

10/24/16
Version: 01

1. Use Cases

Four primary use cases for Gator Lodge are a browser, landlord, tenant, and administrator. A browser is someone who is looking for an apartment listing site to commit to. A landlord is trying to rent out property to a student. Tenant's are people who have accounts and are students trying to find a place to live. Lastly, the administrator is making money off of GatorLodge and running the website.

Browser

Timothy is an incoming student to SFSU and needs a place to live. He has a reasonable computer skill level so he is comparing different apartment search sites to find his home away from home. His parents are aiding him in his apartment search and he is looking for a website that can cater to all their needs in his apartment search. Timothy is able to browse through listings based on a budget, look up the location of an apartment, and view pictures of an apartment he's interested in renting. He is able to save apartment units he's interested in living in so he can discuss them with his parents.

Landlord

Ms. Watson has been living in her duplex for thirty years. She is looking to rent out the upper unit and is looking for an easy-to-use website to showcase her unit and rent it out. She uses an easy system that lets her post pictures of her apartment, fill out availability dates, set the location, and manage payments. Ms. Watson does not have a great relationship with technology but she feels obligated to post her unit online in order for people to see it. She is hoping to find something that is easy for her post and manage her listing while searching for a tenant and to let her manage rent easily without leaving her confused.

Tenant

Joffrey is a returning San Francisco State University student who is looking for a different place to live. He's very comfortable with using computers and all sorts of technology, especially user friendly web sites. Since he's a returning student, Joffrey has an idea of where he wants to live. He is able to search for apartments by using the zip code that school has, which helps with his search. Joffrey being technology sound is able to browse listings and add those listings to his favorites, check reviews, and also look at pictures. He also compares prices so he can find the best price for his apartment, since his parents are paying for his school and board. Seeing how he is a student and school starts at a certain date he is able to check the availability of the listings.

Administrator

Anastasia a former San Francisco State University student is able to manage most of the tasks with running the apartments site. Having basic computer skills she is able to log into the site and manage everyday things. She is able to see listings, censor listings, pictures, and also users. She uses the database using the Workbench interface to manage all the data that is provided. Anastasia is making the site a nice and friendly place for students to come and find the apartment right for them. With that she is able to ban accounts, reviews, pictures and listings. She is also managing the money that is being used on the site, since she is making money from the site. Anastasia is able to do all these things in a timely fashion, not spending hours trying to log in and check the site.

2. Data Definition

Apartment Listing:

All users shall be able to browse through listings, (Guest and Member), the apartment listing will be able to display what Member users uploaded as photos to the apartment.

Apartment Listing Database

Data Name	userId	photo	listingId	listingsTitle	listingsDetail
Data Type	INT	FILE	INT	STRING	STRING

Member_user:

Able to post listings/photos and able to rent. Needs to login/register

Member User Database

Data Name	userId	userFirstName	userLastName	userEmail
Data Type	INT	STRING	STRING	STRING

Photos:

Uploaded by registered members and manageable by admin

Photos Database

Data Name	userId	photo
Data Type	INT	FILE

Administrator:

Access to all user data, able to add or delete listing, spam control, able to login
Has full access to database of members and data

Guest_user:

Can only browse site, cannot post or contact

The apartment listings can be viewed by any user, and the administrator has permission to remove the listing if considered to be spam. Unregistered users are browsers, people who has permission to browse through the listings available. The Member Users will also be able to browse listings, Member users are able to login, post listings and upload content(photos) about the listing. Admin shall be able to login and browse all user profiles and their postings then has the power to ban users and add or remove listings. All registered users has permission to upload photos and post listings. Admin would have final say in whether user uploads has permission to uploads the photos or to post listings.

3. Functional Specifications

1. High Priority - mandatory for project completion.
2. Medium Priority - can be removed if completing them would jeopardize releasing the project on time.
3. Low Priority - features that can be worked on after Priority 1 and 2 requirements are complete.

- **<u>All Users</u>**
  Priority 1:
    1. All Users shall have the option to browse listings with or without logging in
    2. All users shall have the option of registering without facebook becoming a Member User
    3. All Users shall have the ability to search through listings with various filters
    4. All Users shall have the option to sort listings on the screen based on various conditions
    5. All Users shall be able to view the website on various size screens easily due to the site's use of media queries to determine the screen size being used
    6. All Users shall be able to view the location of the listing using embedded google maps services

  Priority 3:
    1. All users shall have the ability to authenticate their identity using facebook login

- **<u>Member users</u>**
  Priority 1:
    1. Member Users shall be able to create listings on the websites
    2. Member Users shall have the option to upload an arbitrary amount of images to listings they own
    3. Member Users shall be able to edit their listings at any time
    4. Member Users shall be able to leave written reviews that can be viewed by other users in addition to a star rating if desired

  Priority 2:
    1. Member Users shall have option to add potential listings to their favorites list so that they can be easily viewed at a later date

  Priority 3:

1. Member Users shall be able to review listings they have previously lived in with 1 to 5 star rating

- **Administrators**
  Priority 1:
  1. Administrators shall be able to remove images deemed inappropriate
  2. Administrators shall be able to ban user accounts who fail to adhere to the terms of service agreements

4. Non-Functional Specifications

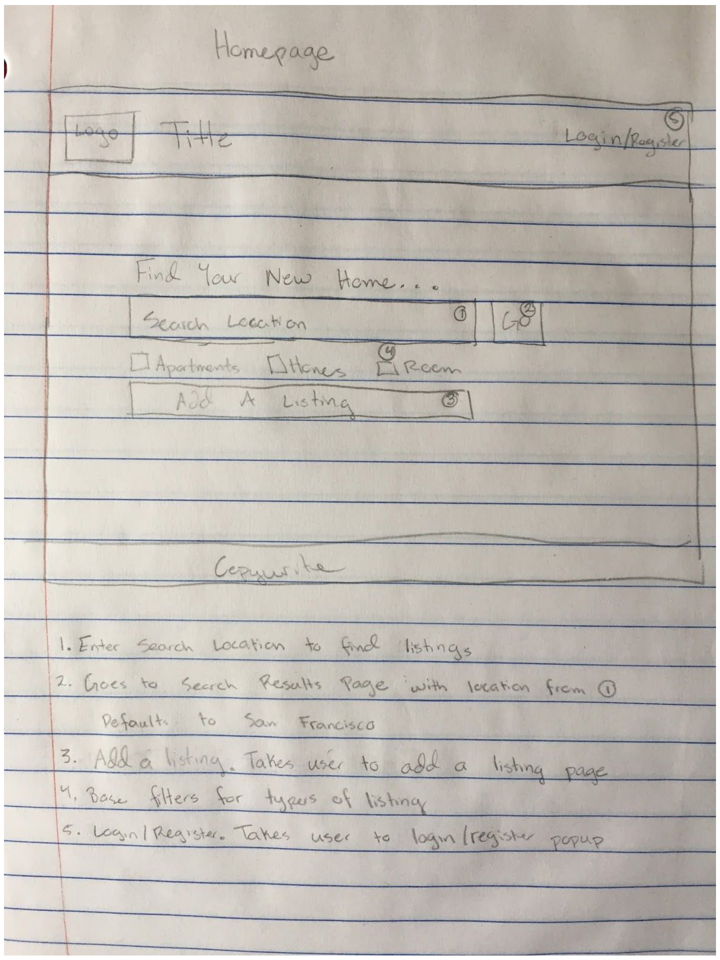
1. System shall respond visually within 5 seconds.
2. File size in no time shall exceed 2 Mbytes.
3. Users with high-school diploma, after 1 hour training, shall complete the task in 5 minutes with no more than 2 errors.
4. Each WWW page shall have official company logo in upper left corner.
5. The following user data shall be collected and stored; the data shall be used ONLY for record keeping.
6. Application shall be developed using class provided LAMP stack.
7. Application shall be developed using pre-approved set of SW development and collaborative tools provided in the class. Any other tools or frameworks shall be explicitly approved by Marc Sosnick on a case by case basis.
8. Application shall be hosted and deployed on Amazon Web Services as specified in the class.
9. Application shall be optimized for standard desktop/laptop browsers, and shall render correctly on the two latest versions of all major browsers: Mozilla, Safari, Chrome. It shall degrade nicely for different sized windows using class approved programming technology and frameworks so it can be adequately rendered on mobile devices.
10. Data shall be stored in the MySQL database on the class server in the team's account.
11. Application shall be served from the team's account.
12. No more than 50 concurrent users shall be accessing the application at any time.
13. Privacy of users shall be protected and all privacy policies will be appropriately communicated to the users.
14. The language used shall be English.
15. Application shall be very easy to use and intuitive. No prior training shall be required to use the website.
16. Google analytics shall be added for major site functions.
17. Messaging between users shall be done only by class approved methods to avoid issues of security with e-mail services.
18. Pay functionality (how to pay for goods and services) shall be simulated with proper UI, no backend.
19. Site security: basic best practices shall be applied (as covered in the class)
20. Modern SE processes and practices shall be used as specified in the class, including collaborative and continuous SW development, and only the tools and practices approved by instructors
21. The website shall prominently display the following text on all pages "SFSU/FAU/Fulda Software Engineering Project, Fall 2016. For Demonstration Only". (Important so as to not confuse this with a real application).

5. User Interface and Mockup

Major use cases for GatorLodge are of a landlord looking to rent out their property to a student and a student looking to rent an apartment. The student looking to rent an apartment branches off into those with and without an account. All three will begin at the homepage. The student looking to rent an apartment will have a similar experience whether logged in or not. The user who is not logged in will not have access to the contact property owner options and will be able to save favorite listings and refer to them later on a favorites page. The student who is logged in will have access to all features. The logged in student and landlord will be referred to as member users and the browsing student will be considered a guest user.

Member User (landlord)

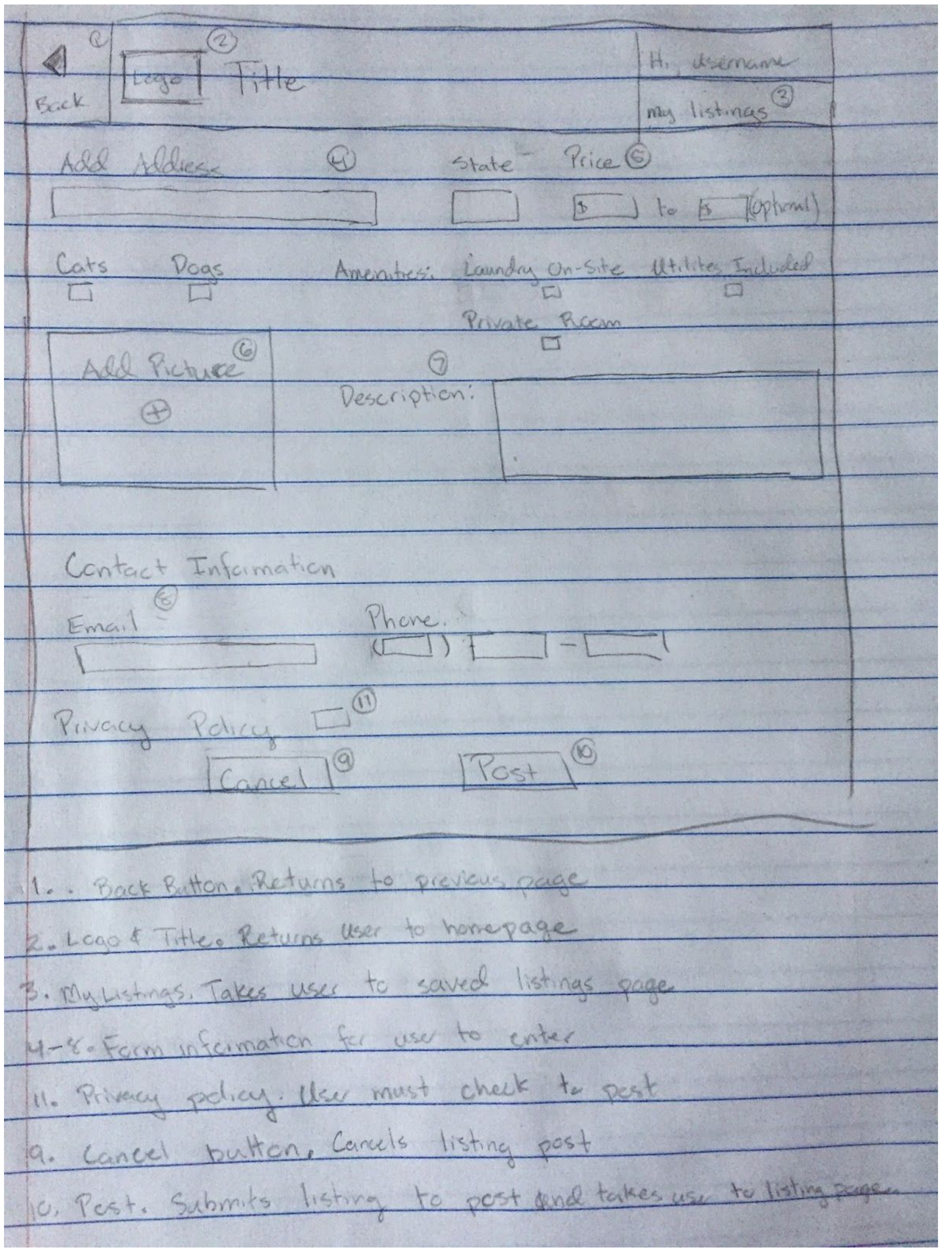
The landlord will begin on the homepage (Figure 1) and click the add a listing button (3) to proceed to the add listing page.



1. Enter Search Location to find listings
2. Goes to Search Results Page with location from ①
   Defaults to San Francisco
3. Add a listing. Takes user to add a listing page
4. Base filters for types of listing
5. Login/Register. Takes user to login/register popup

Figure 1: Homepage

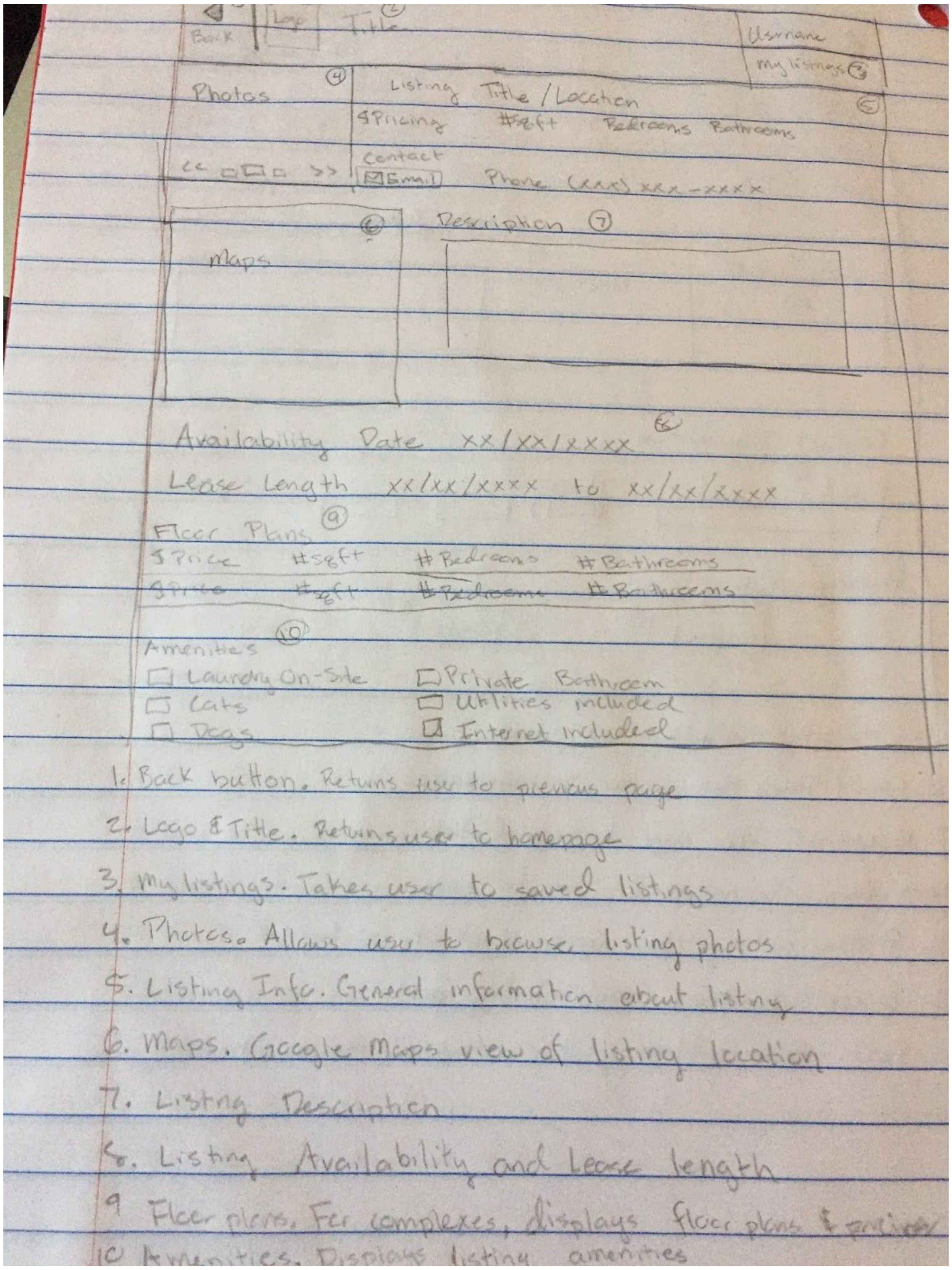
After the user clicks on the add a listing button they will be taken to the Add Listing Page (Figure 2). The landlord will fill out a form (items 4-8) for their listing then press the post button (10). Once the post button is pressed, the site will check if the user is logged in or not. If the user is logged in then the login popup will present and ask for credentials. If the user is logged in then hitting the post button will take the user to the listing they created.



1. Back Button. Returns to previous page
2. Logo & Title. Returns user to homepage
3. My Listings. Takes user to saved listings page
4-8. Form information for user to enter
11. Privacy policy. User must check to post
9. Cancel button. Cancels listing post
10. Post. Submits listing to post and takes user to listing page.

Figure 2: Add a Listing Page

Once the listing has been posted the landlord will be taken to the posted listing page to view it. The posted listing page will display all the listing information for the landlord.

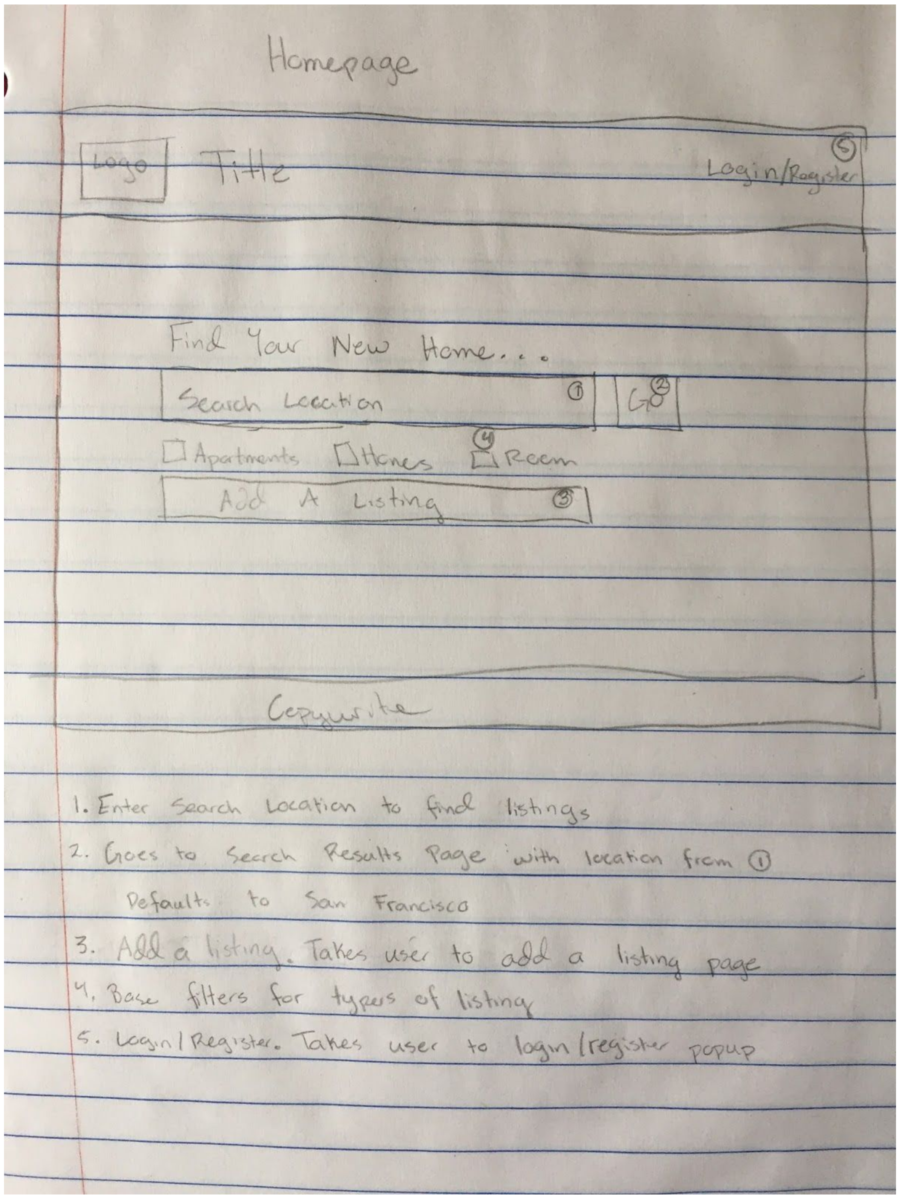


1. Back button. Returns user to previous page
2. Logo & Title. Returns user to homepage
3. My listings. Takes user to saved listings
4. Photos. Allows user to browse listing photos
5. Listing Info. General information about listing
6. Maps. Google Maps view of listing location
7. Listing Description
8. Listing Availability and Lease length
9. Floor plans. For complexes, displays floor plans & pricing
10. Amenities. Displays listing amenities

Figure 3: Detailed Listing Page

Member User (student)

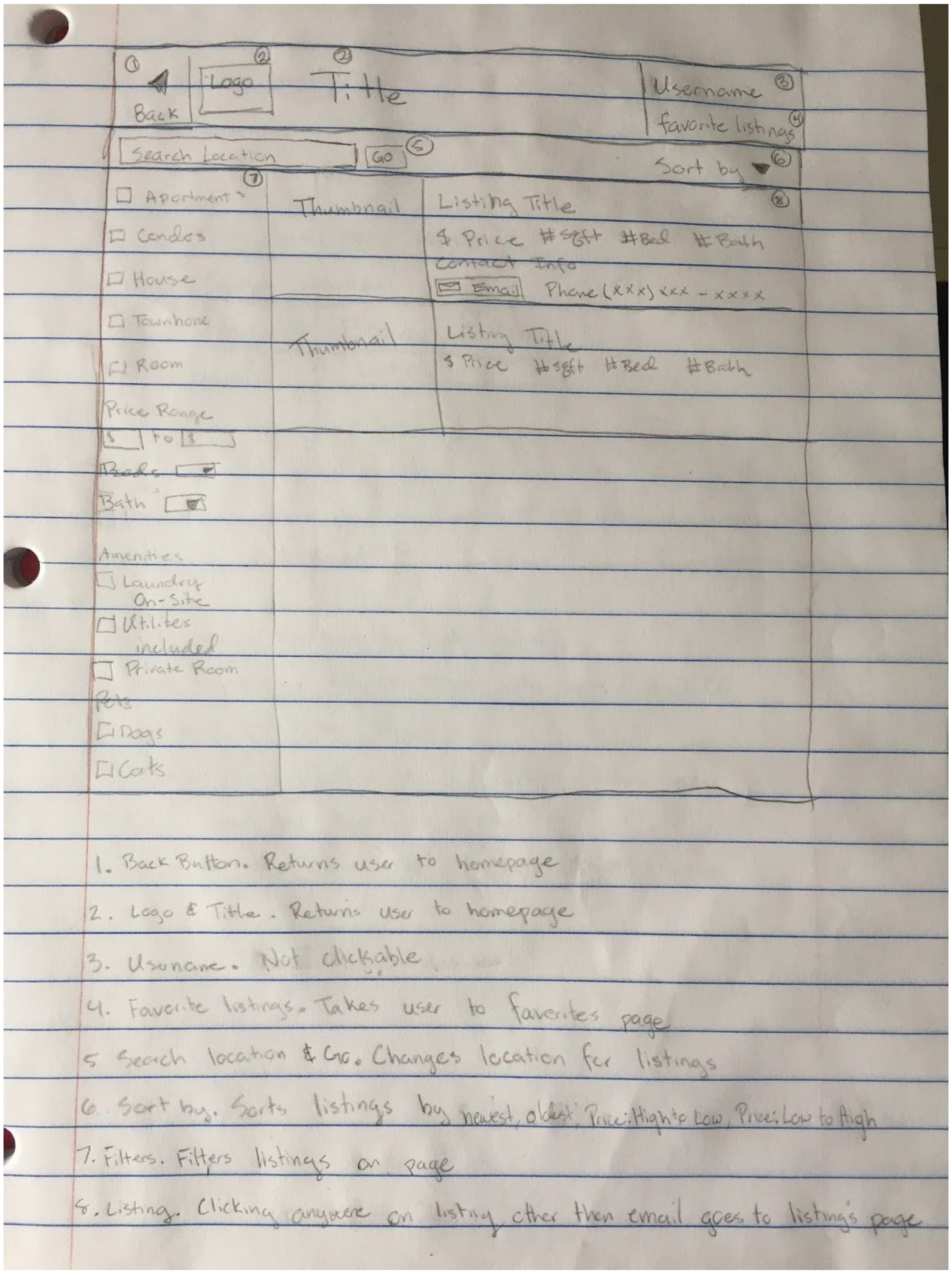
The student will begin on the homepage (Figure 3) much like the landlord. Instead of clicking the add a listing button (3), the user will enter a location in the search location bar (1) and then search using the search button (2). The search button will direct the user to Search Results Page.



1. Enter Search Location to find listings
2. Goes to Search Results Page with location from ①
   Defaults to San Francisco
3. Add a listing. Takes user to add a listing page
4. Base filters for types of listing
5. Login/Register. Takes user to login/register popup

Figure 3: Homepage (note: identical to Figure 1)

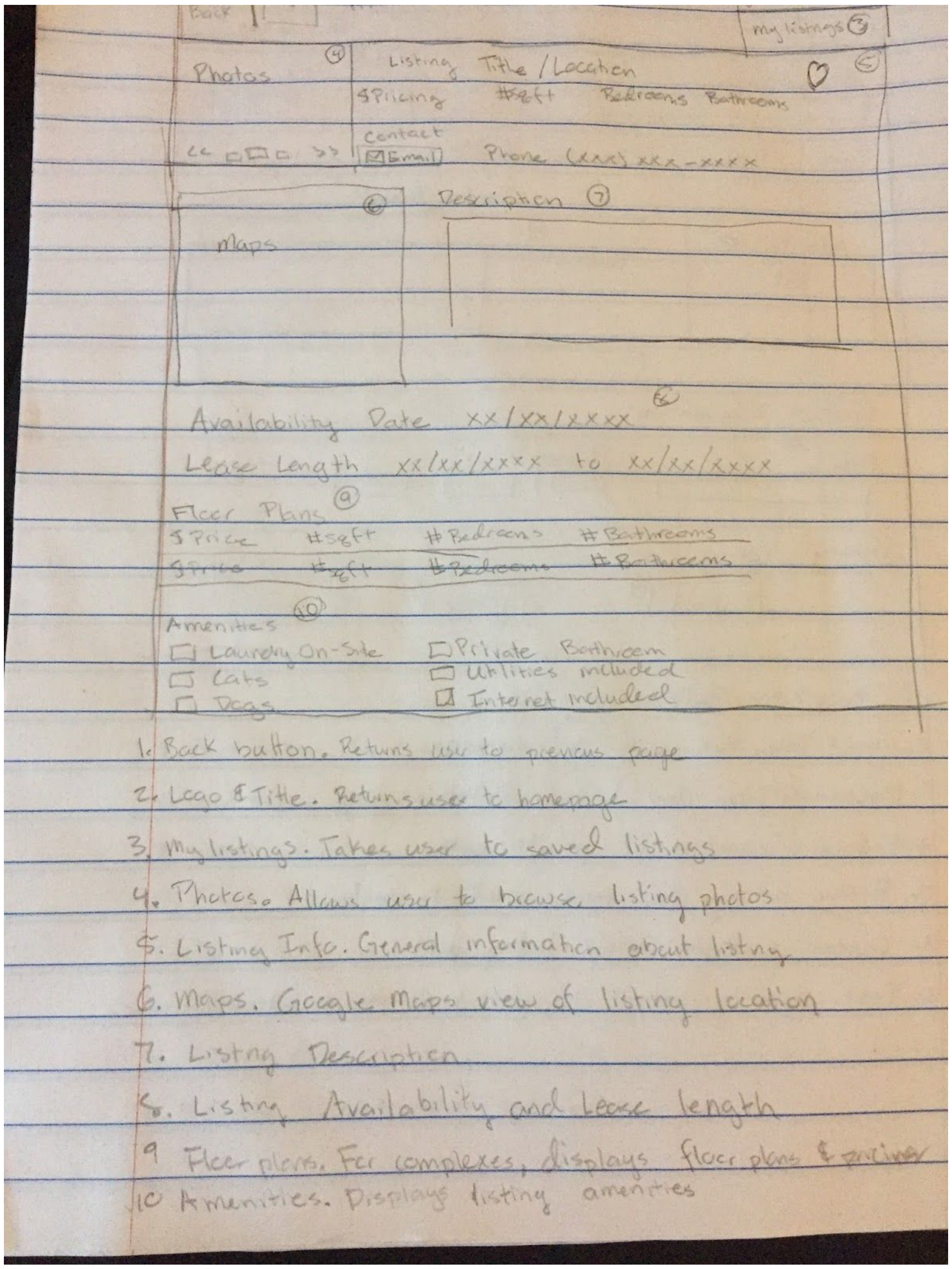
Once the member user reaches the Search Results Page (Figure 4), they are able to change the filters with (7), browse listings, and change the sorting method (6). The user may open a detailed listing page by clicking on an individual listing (8).



1. Back Button. Returns user to homepage
2. Logo & Title. Returns user to homepage
3. Usename. Not clickable
4. Favorite listings. Takes user to favorites page
5 Search location & Go. Changes location for listings
6. Sort by. Sorts listings by newest, oldest, Price: High to Low, Price: Low to High
7. Filters. Filters listings on page
8. Listing. Clicking anywere on listing other then email goes to listing's page

Figure 4: Search Results Page

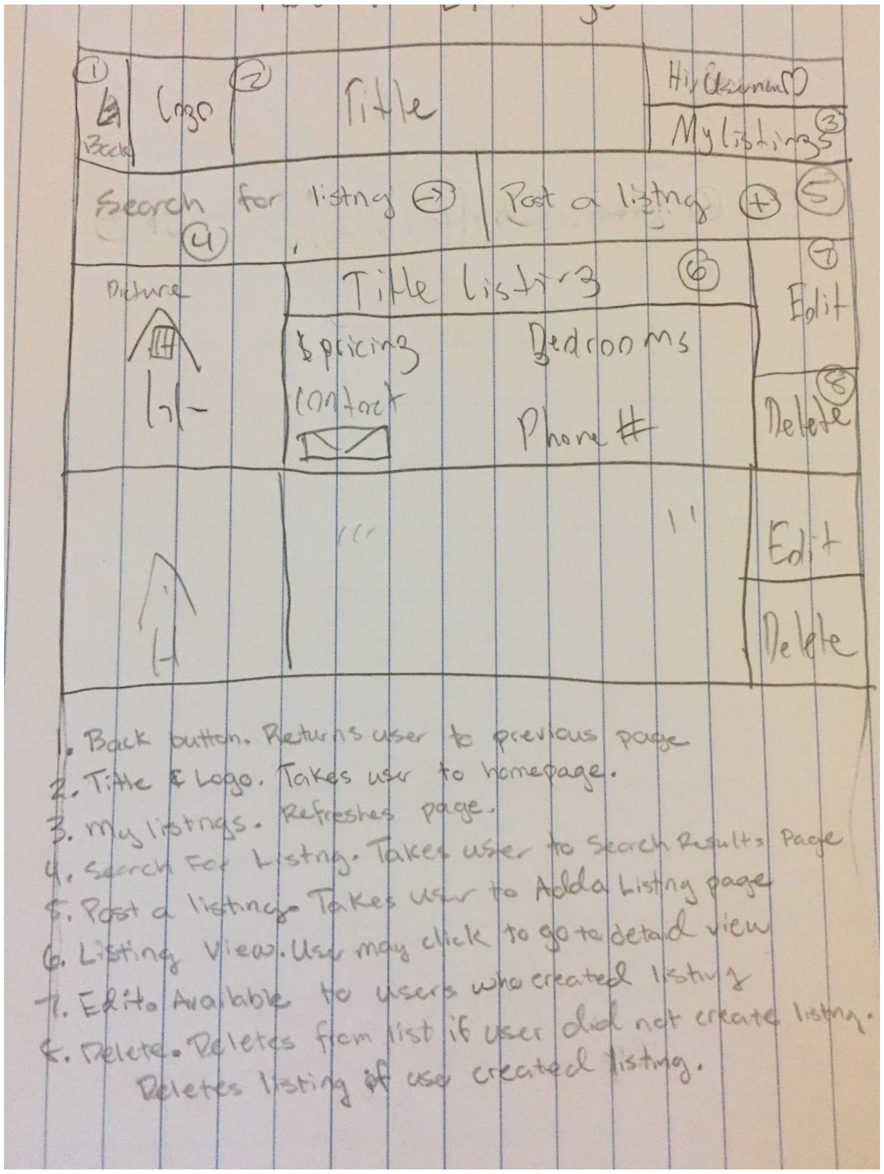
The user is able to view further details about a specific listing in the Detailed Listing Page (Figure 5). This page has contact information (5), a maps view (6), and a description (7). Should the listing be part of a complex the page will display information for the units in a complex (9). The member user can either favorite the listing by clicking on the heart in (5), contact the property owner, or return to the Search Results Page (Figure 4). To create an example, the member user clicks the heart in (5) then clicks on My Listings in (3) to go to view their saved listings.


Back
My listings ③
Photos ④
<< □□□ >>
Listing Title / Location ⑤
$Pricing #sqft Bedrooms Bathrooms
Contact
Email
Phone (xxx) xxx-xxxx
⑥ Maps
Description ⑦
⑧
Availability Date xx/xx/xxxx
Lease Length xx/xx/xxxx to xx/xx/xxxx
Floor Plans ⑨
$Price #sqft #Bedrooms #Bathrooms
$Price #sqft #Bedrooms #Bathrooms
Amenities ⑩
Laundry On-Site
Cats
Dogs
Private Bathroom
Utilities included
Internet included

1. Back button. Returns user to previous page
2. Logo & Title. Returns user to homepage
3. My listings. Takes user to saved listings
4. Photos. Allows user to browse listing photos
5. Listing Info. General information about listing
6. Maps. Google Maps view of listing location
7. Listing Description
8. Listing Availability and Lease length
9 Floor plans. For complexes, displays floor plans & pricing
10 Amenities. Displays listing amenities

Figure 5: Detailed Listing Page

The My Listings Page (Figure 6) display favorited listings and posted listings. The user will be able to delete any favorited listings using the Delete button (8). Should the user have posted listings a Edit button (7) will be present above the Delete button (8). Clicking on the Edit button (7) will take the user to the Post listing page where they will be able to edit the listing information. The member user will be able to click on a listing in (6) and be able to go to the Detailed Listing View (Figure 5) for that page.

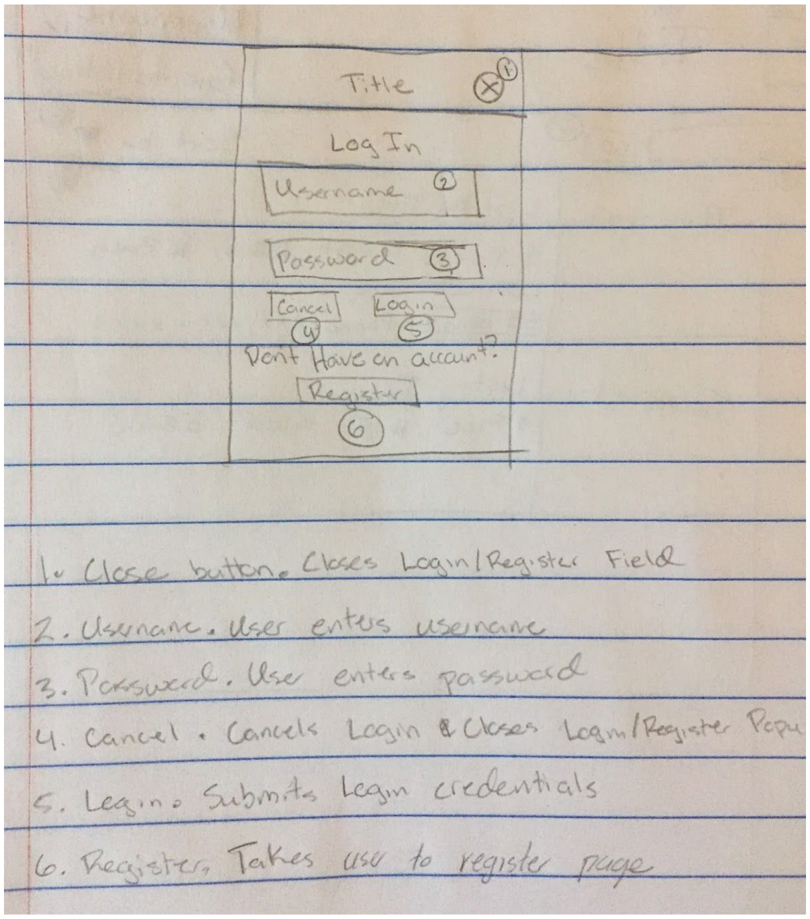


1. Back button. Returns user to previous page
2. Title & Logo. Takes user to homepage.
3. My listings. Refreshes page.
4. Search For Listing. Takes user to Search Results Page
5. Post a listing. Takes user to Add a Listing page
6. Listing View. User may click to go to detail view
7. Edit. Available to users who created listing
8. Delete. Deletes from list if user did not create listing. Deletes listing if user created listing.

Figure 6: My Listings Page

Guest User

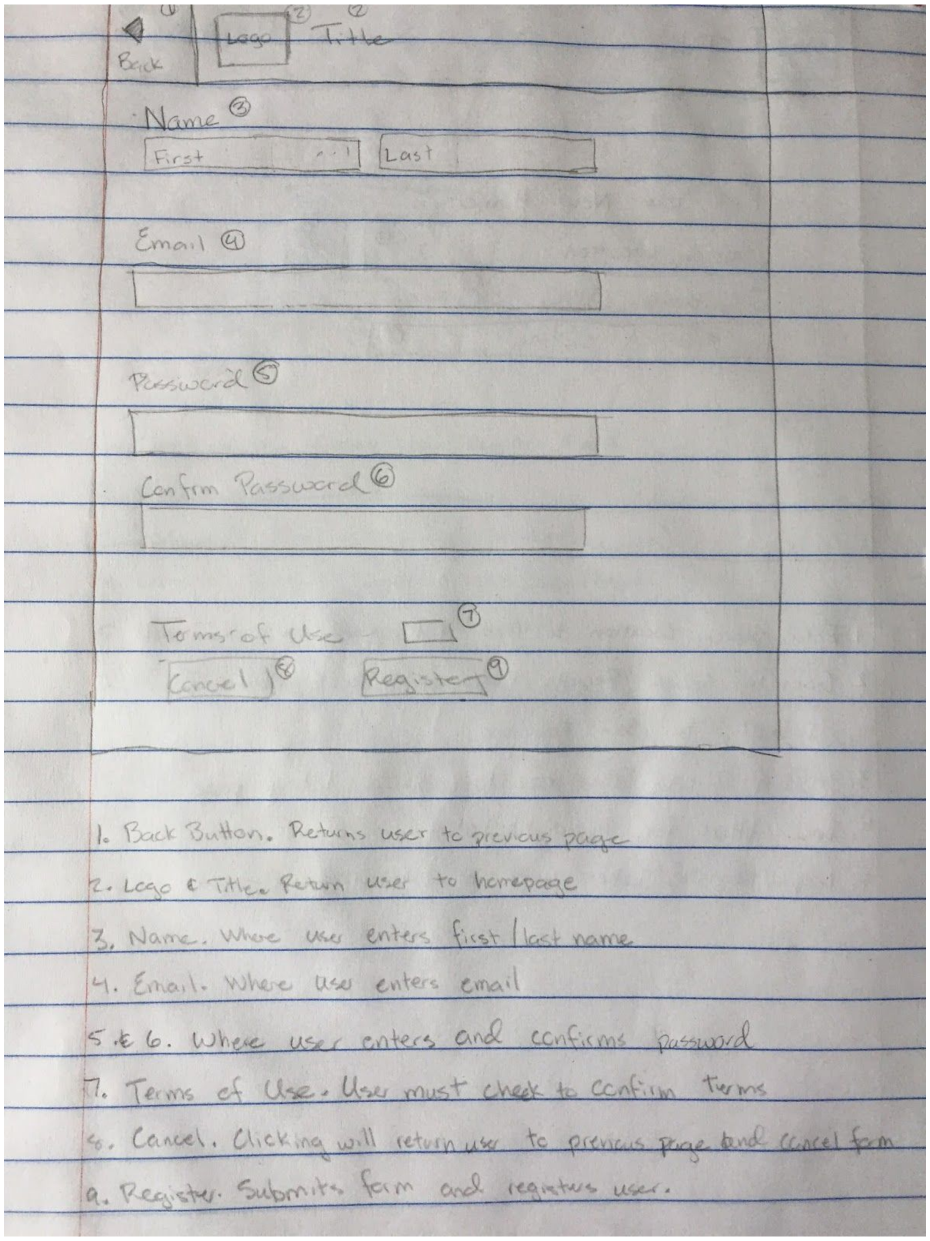
The guest user will begin on the homepage (Figure 1). They will then proceed in a similar fashion to the student who is a member user. The guest user will browse listings in a location in the Search Results Page (Figure 4) and be able to view a detailed view of a listing as in Figure 5. If the guest user should at any time try to favorite a listing, post a listing, or attempt to contact a property the guest user will be presented with a login popup (Figure 7). As the user is a guest user, they will click on the register button (6) and be directed to the Register Page (Figure 8).



1. Close button. Closes Login/Register Field
2. Username. User enters username
3. Password. Use enters password
4. Cancel. Cancels Login & Closes Login/Register Popu
5. Login. Submits Login credentials
6. Register, Takes user to register page

Figure 7: Login/Register Popup

The Registration Page (Figure 8) will present a form for the guest user to fill out to register. The user will be able to go back to their previous page should they choose not to register with the Back Button (1). The guest user will fill out a name, email, and password. The user must check the Terms of Use (7) in order to register an account. The guest user will register with the Register button (9) and the system will verify the form information and Terms of Use agreement before creating an account. Once an account has been created then the user is logged in and is taken back to their previous page prior to registration.



1. Back Button. Returns user to previous page
2. Logo & Title. Return user to homepage
3. Name. Where user enters first/last name
4. Email. Where user enters email
5. & 6. Where user enters and confirms password
7. Terms of Use. User must check to confirm terms
8. Cancel. Clicking will return user to previous page and cancel form
9. Register. Submits form and registers user.

Figure 8: Registration Page

6. High-level system architecture & Database Organization

6.1 LAMP stack

- The application shall be hosted on Amazon Web Services (AWS) and shall run on Ubuntu Linux version 16.04 (or the current version installed on AWS).
- All web pages shall be served by Apache version 2.4.10.
- The application shall use MySQL version 5.7.12 to store all data.
- The application shall use PHP version 7.0.4 to serve dynamic web pages.

6.2 Front-end

- The application shall use HTML, CSS and Javascript for front-end. Additionally, JQuery shall be user for HTML manipulation,
- The application shall use Bootstrap to handle the responsive design.

6.3 Back-end.

- MINI PHP framework shall be used for the back-end.

6.4 Development Environment.

- The project shall use Gitlab to store the code base.

6.5 Database Organization

Apartment Listing Database

Data Name	userId	photo	listingId	listingsTitle	listingsDetail
Data Type	INT	FILE	INT	STRING	STRING

Member User Database

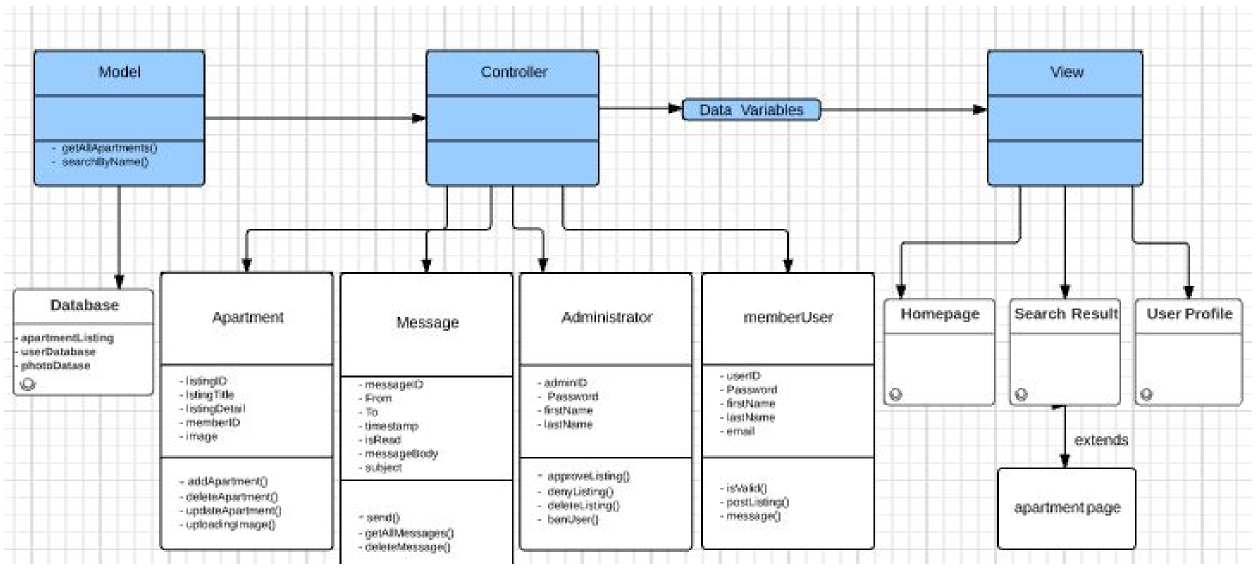
Data Name	userId	userFirstName	userLastName	userEmail
Data Type	INT	STRING	STRING	STRING

Photos Database

Data Name	userId	photo
Data Type	INT	FILE

7. High Level UML

Below is Gator Lodge’s high level UML. We also provide a URL below:
https://www.lucidchart.com/invitations/accept/d566c319-225d-4f95-b8ff-c12278f8154b



8. High Level Application Program Interfaces

The *Apartments* class will be responsible for the API calls related to getting information about the apartment. /searchresults/ will make a call to the database and fetch the apartments the falls into the search category, only the basic information on the apartment listing will be returned. /searchresult/ {id_number} will make another call to the database fetching all the details of the apartment given the apartment id. When data is fetched from the database it will be converted into JSON format.

The *memberUsers* class will have access to API calls related to posting listings and messaging other member user. When boolean variable logged in is true , member user will be allowed to post listings and access to the /postlisting/ URL allowing user to POST to the apartments database.

The *messages* class will be there for member users to view messages between user A and user B. Input currently active user and message from second member user to view messages between the two

API URL	Method	Input	Output
/searchresult	GET	{search_term} - string	JSON Array: apartment_id, name, location,zip
/searchresult/{id}	GET	{apartment_id} - int	JSON Array: apartment_id, name, zip, listing details
/postlistings	POST	{apartment_details} - string (multiple strings)	-
/messages	GET	{member_id} - int	JSON Array: all messaged between logged in user and another user
/user	GET	{user_id} - int	JSON array: member user details

9. Key Risks

	Key Risks	Response
Skills Risk	Team members may not have experience with some Technologies, for example, PHP and MINI framework,	Learn basic PHP from Khan Academy. Also, organize team training session to help each other to learn together.
Schedule Risk	All team members have very packed schedule.	Team lead will organize weekly team meeting. Also, check-in often for progress.
Technical Risk	User information (username and password) need to be stored securely. Also, there are several common attacks that need to be taken into consideration, for example: sql injection, session hijack, etc	All username and password are going to be hashed/salted before storing in the database. Need to use HTTPS instead of HTTP.
Content Risk	There might be miscommunication between team members.	Team lead and tech lead need to check occasionally to make sure the team is on the same page during weekly check-in.
Legal Risk	The side may use some third party APIs that do need explicitly marked.	Check the documentation of all the third party APIs before use.

10. Team Members

Team Lead: Lijie Zhou
Tech Lead: Anna Sever
Team members:

Frontend: Theofanis Koutoulas, Anna Sever
Backend: Logan Figgins, Hao Xian Zheng, Lijie Zhou
Full-stack: Ivan Marchenk