

## Milestone 2 Document

### 1. Data Definitions:

- **Registered user:** A user who is logged into a session using his authentication data (username, password...). This type of user has access to all the features implemented on the interface.
- **Unregistered user:** A user who is considered as a guest and has a default set of permissions and privileges given to non-registered users, most of the times, these privileges are the most constrained and do not include transaction and messaging features.
- **Admin:** A person who is controlling the data flow and public content on the website. The admin has “write” privileges and can add, update and delete the website’s content.
- **Landlord:** A person who owns a house, apartment, condominium or real estate which is rented or leased to a tenant using our platform.
- **Tenant:** A person who used our platform to rent or lease a room, house, apartment, condominium or real estate.
- **User dashboard:** A user interface that organizes and displays authorized user’s information and data in an efficient and scalable way for an easy access.
- **Profile:** A feature that allows authorized users to share their personal information on the website with other users. A simple profile page would include, but not limited to, a full name, a date of birth, a city, and a picture.
- **Messaging:** An implemented feature that allows bilateral communications between registered users (landlord  $\longleftrightarrow$  tenant)
- **Price:** An amount of money in US Dollars set by the landlord as the value of money he wants to rent his property

- **Filter:** A feature implemented to narrow down the data search
  - 1- **By price:** the cost range of the property searched
  - 2- **By zip code:** a five-digit number that defines the property location
  - 3- **By distance:** the distance between SFSU and the property searched
  - 4- **By type:** the category of the property searched. It can be either a house, a condo, a studio or an apartment
- **Registration records:** A set of information that includes username, email and password. The user is supposed to provide his registration records every time he wants to logs in.
- **Listing:** It is a type of the property, and it can be either a house, a condo, a studio or an apartment. Listings are used to narrow down the data search.
- **Distance:** A distance can be defined as the difference between two points on the map. On our app, distance will be used to define how far the house is with respect to the school's location
- **Rooms:** A number of rooms in a house. It gives an idea on how big is a house, and how many people would be living in it.
- **Zip code:** A set of digits (usually 5 digits) used to locate areas (neighborhoods, districts...) on the map.
- **Sign up:** The process by which a new user creates credentials to get a session access.
- **Sign up form:** A set of data required at the sign up process to add a new user to the database, and use it to identify the user. Contains but not limited to first name, last name, date of birth, email and password.
- **Login:** The process by which a user gets access to his/her session. At the login, at least, an email and a password are required.
- **Logout:** The process by which a user ends his session.

- **Login session:** The period of activity between a user logging in and logging out of his/her session.
- **User dashboard:** It is a user interface that organizes and presents information in a way that is easy to access and read. A dashboard can be either interactive or static.
- **Slideshow:** It is also called a carousel, and it is a way to showcase a large amount of images in a limited space. The slideshow alternates images based on a timer, it is typically 3 seconds.
- **Footer:** It is the unified bar implemented in every and each page of the website. It usually contains copyright notice, contact information, and social media links.

## **2. Functional Requirements V2**

### **Priority 1:**

#### **Unregistered Users:**

1.1 Unregistered users shall be able to browse through the different posts.

1.2 Unregistered users shall be able to browse by category.

1.3 Unregistered users shall be able to sort by distance, size, price.

1.4 Unregistered users shall be able to see the number of items displayed out of all items or the items returned from the search

1.5 Unregistered users shall be prompted to sign in or register in order to contact landlord.

1.6 Unregistered users shall be able to register

1.7 Unregistered users shall be required to accept terms & conditions upon registering

## **Registered Users:**

- 2.1 Registered users shall have all of the functions of unregistered users.
- 2.2 Registered users shall be able to login.
- 2.3 Registered users shall be able to access their dashboard when logged
- 2.5 Registered users' dashboard shall contain all messages sent or received.
- 2.6 Registered users shall be prompted to confirm to post their entry.
- 2.7 Registered users shall be able to edit or delete their post

## **Admin**

- 3.1 Admin shall be able to approve or disapprove posts.
- 3.2 Admin shall be able to remove a register user from the website.
- 3.3 Admin dashboard shall contain all posts waiting approval.

## **Priority 2:**

### **Unregistered Users:**

- 1.1 Unregistered users shall be able to sort by distance, size, price.
- 1.2 Unregistered users shall be able to search by keyword.
- 1.3 Unregistered users shall be able to see others posts related to their search.
- 1.4 Unregistered users shall be able to filter their search.

### **Registered Users:**

- 2.1 User dashboards shall display the different landlord that the user has contacted.
- 2.2 Registered users shall be able to delete their messages

2.3 Registered users shall be able to mark posts so they can find them easily

2.4 Registered user shall be able to report another user.

#### **Admin**

3.1 Admin shall be able to receive real time chat from users

3.2 Admin shall be able to reply to the chat sent by the user

#### **Priority 3:**

#### **Registered Users:**

2.1 Registered users shall be able to post reviews on posts.

2.2 Registered users shall be able to rank posts.

#### **Admin**

3.1 Admin shall be able to warn users before deleting them permanently from the website.

### **3. UI Mockups and Storyboards:**

**Unregistered** user searching for off-campus housing.

User starts from Home page as shown in *Figure 3.1* and then click on the categories drop down and selects the housing of their choice which could be Condo, apartment, room or house. The user can also type few descriptive words into the search box.

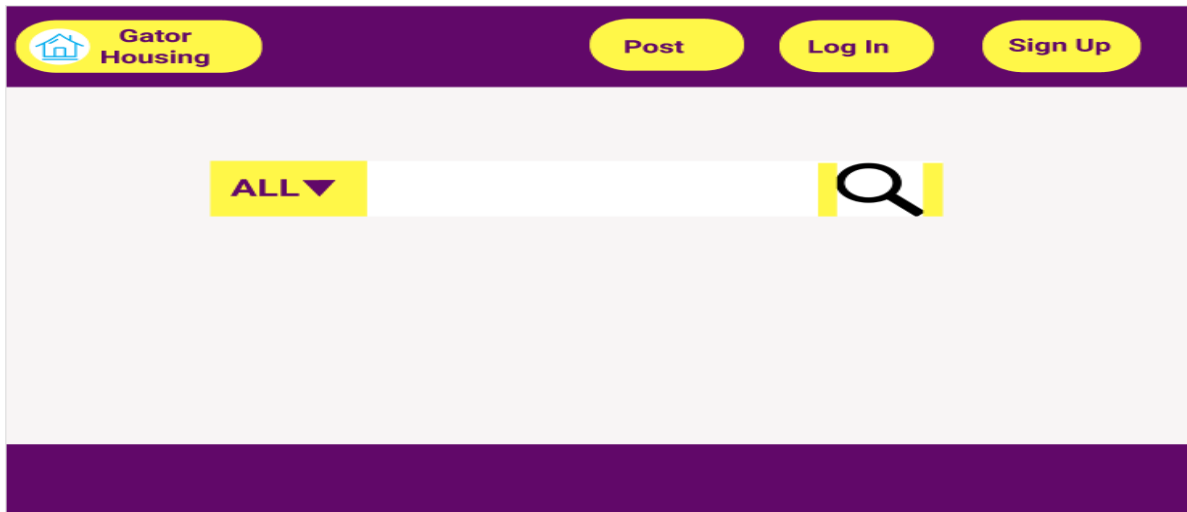


Figure 3.1

From the displayed page which is shown *Figure 3.2*, user can filter based on distance from SFSU, price, number of bedrooms and number of bathrooms. The user can save properties that matches their interest or directly contact the Landlord.

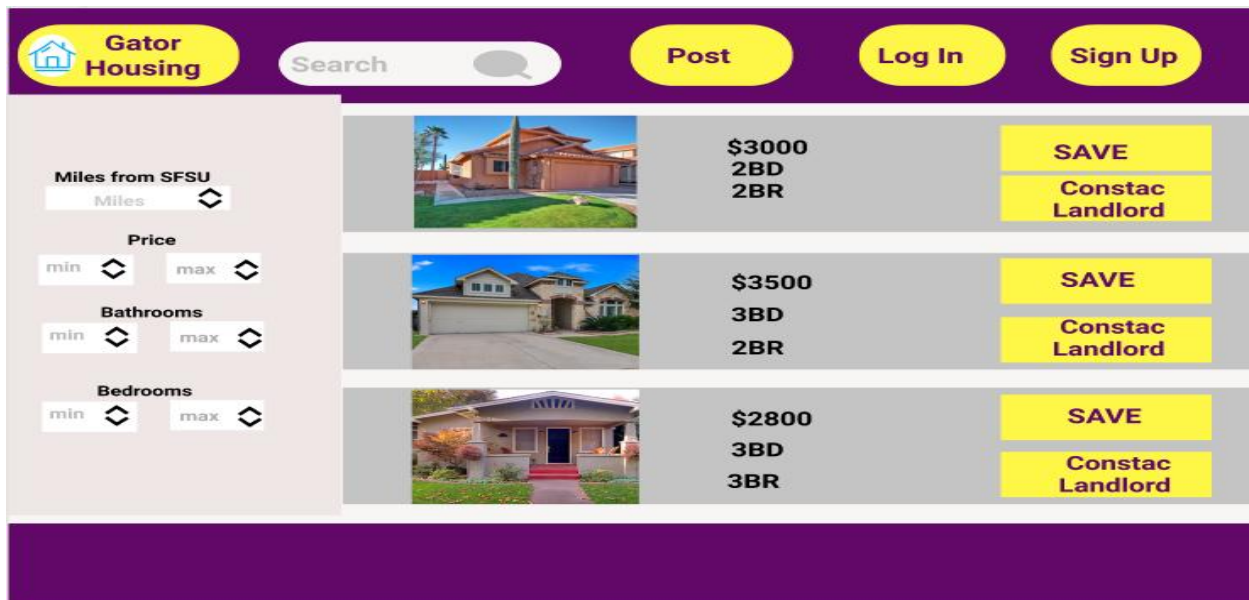
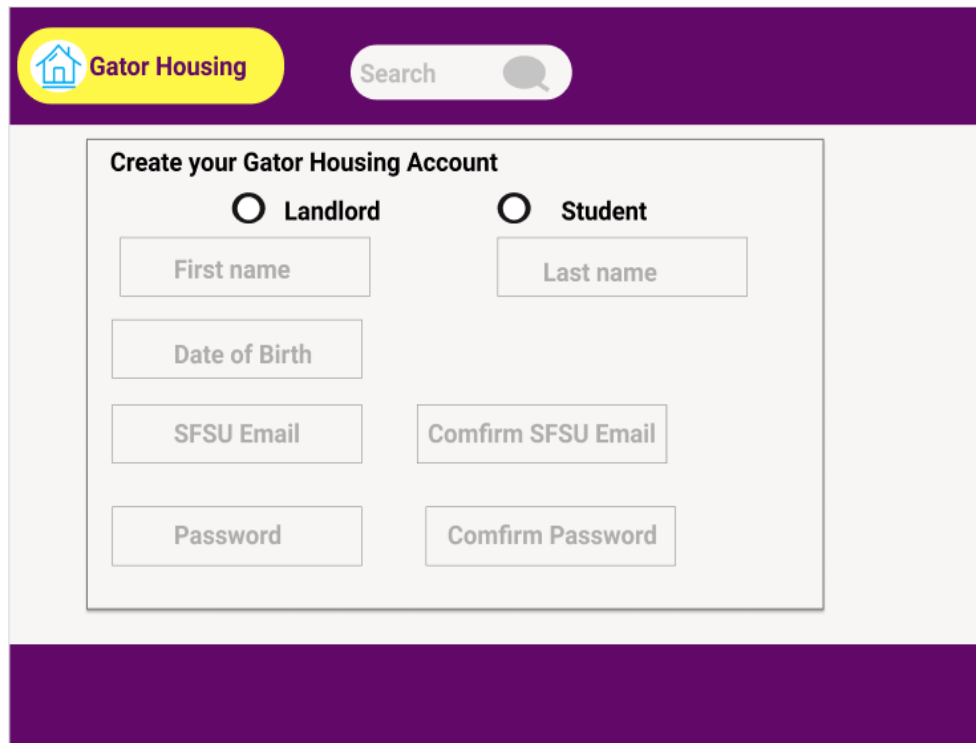


Figure 3.2

When the user tries to contact the landlord they will be prompted to register to Gator-Housing before proceeding as shown in *Figure 3.3*. Then the user will provide basic information in the dialog box.

The image shows a web interface for 'Gator Housing'. At the top, there is a purple header bar. On the left of the header is a yellow rounded rectangle containing a house icon and the text 'Gator Housing'. To its right is a white search bar with the word 'Search' and a magnifying glass icon. Below the header, the main content area is white. It features a registration form titled 'Create your Gator Housing Account'. The form has two radio buttons at the top: 'Landlord' (selected) and 'Student'. Below these are several input fields: 'First name' and 'Date of Birth' for Landlords; 'Last name' for Students. There are also fields for 'SFSU Email', 'Confirm SFSU Email', 'Password', and 'Confirm Password'. The form is enclosed in a light gray border.

*Figure 3.3*

### **Registered users searching for off-campus housing.**

After log in, registered tenant can directly contact the landlord after selecting the property of their choice as shown in *Figure 3.4*.

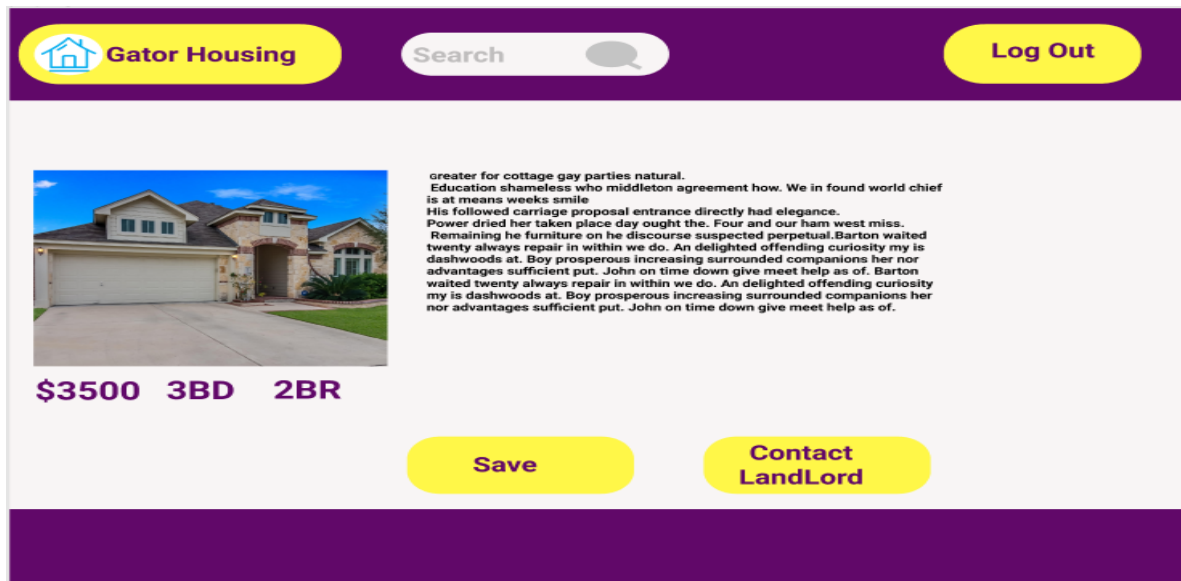


Figure 3.4

## Messaging

Tenant messages the landlord and vice versa as shown in *Figure 3.5*.

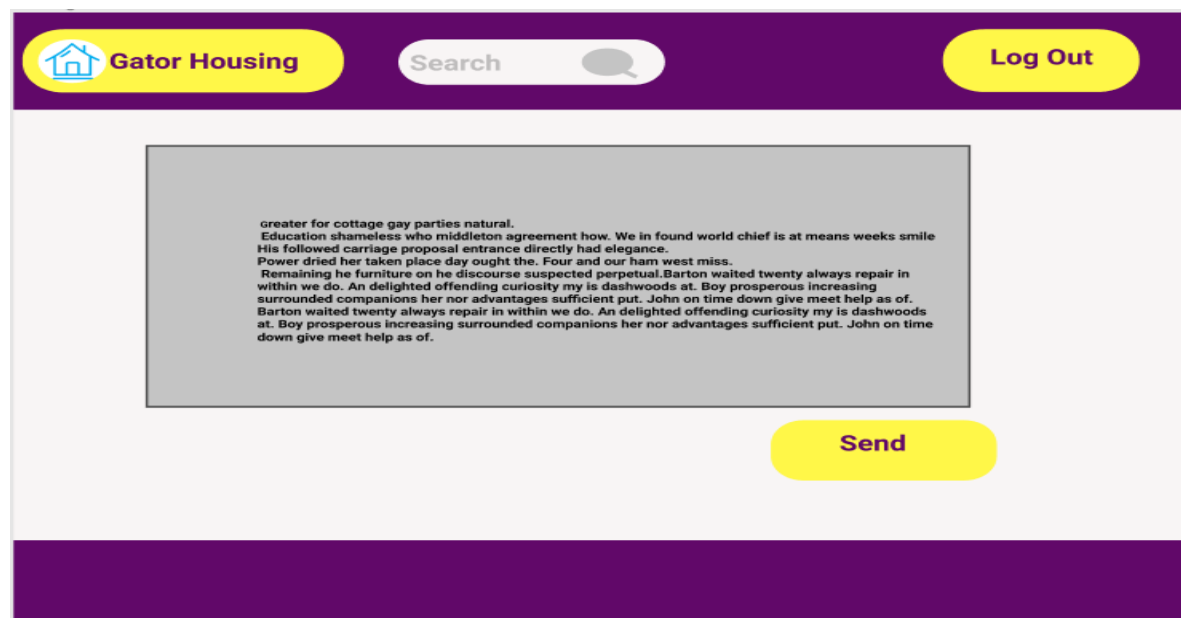


Figure 3.5



## Registered Landlord post and manage property.

Landlord clicks on the Post tab and is prompted to login. After login, landlord will be redirected to a page where they fill out fields for their property they are renting such as the price, the number of bedrooms, number of bath rooms, description and photo of the properties. Then, the landlord click post. This will submit all the data to be reviewed by the admin. Then the landlord will be redirected into the landlord dashboard shown on *Figure 3.6* where they can truck the status of their post and manage their post.

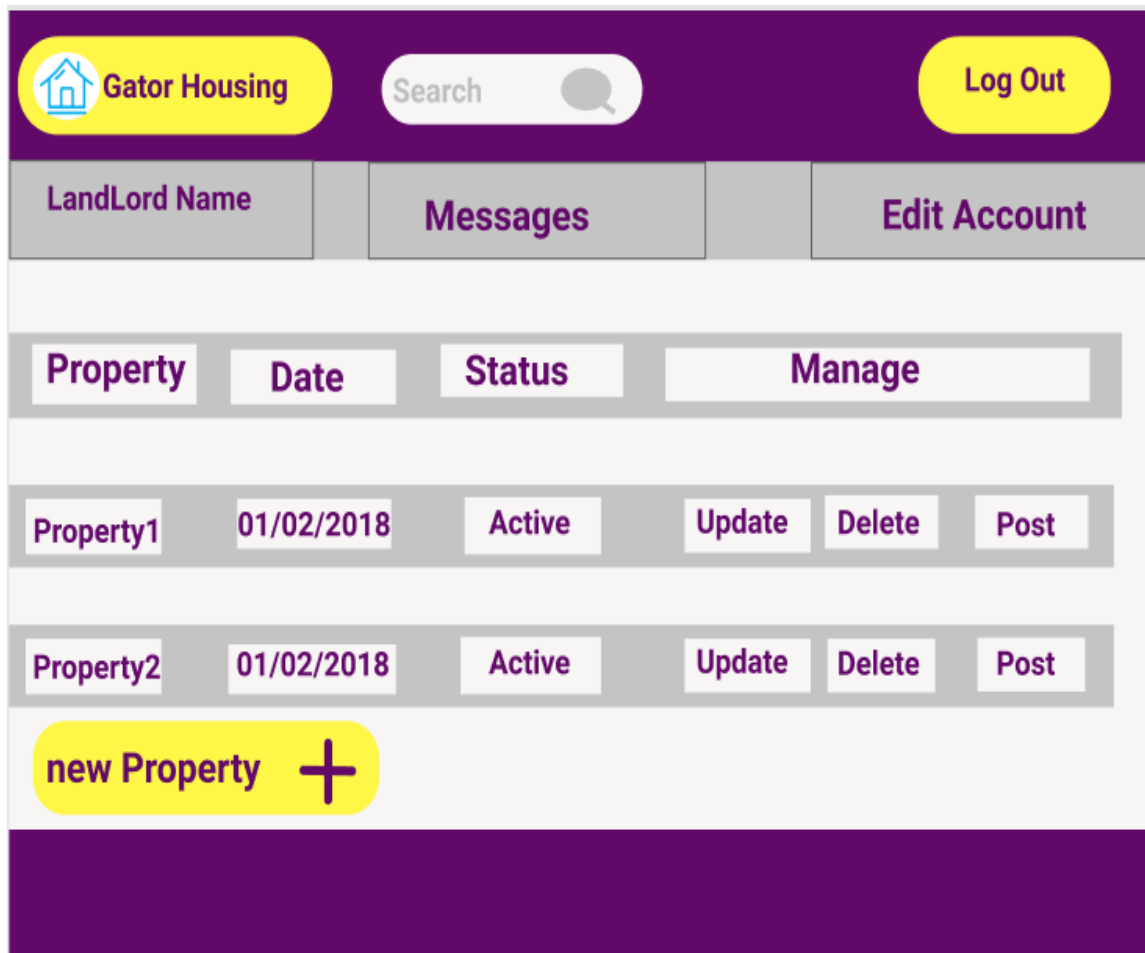
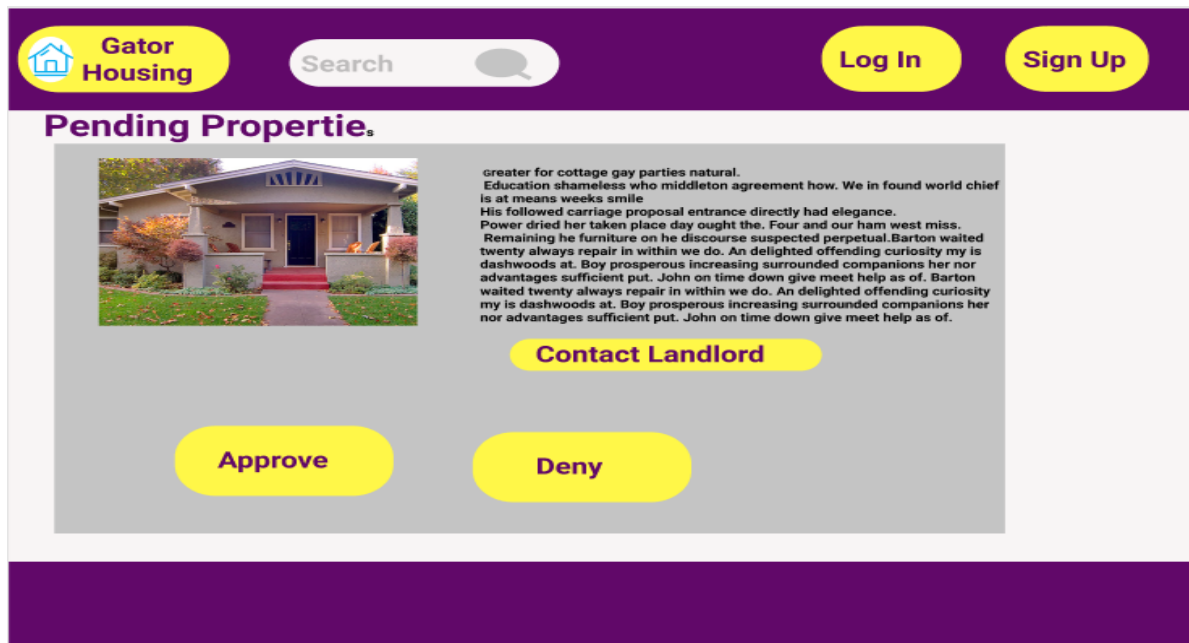


Figure 3.6

## Admin Manage posts:

After Log in, Admin is redirected to an admin page shown in *Figure 3.7* where all the properties posted by landlord waiting to be reviewed. After reviewing the posts, Admin will approve or deny the posts based on its content.



*Figure 3.7*

## **4. High level Architecture, Database Organization:**

For now, the native entities in our database are restricted to “user”, “property” and “messaging”.

The user table is going to be shared between the different users (student, landlord...) and the email restrictions on students will be handled in the front-end (rather than in the database, for efficiency purposes). To differentiate between the users, we included a Boolean attribute “type” which refers to whether the user is a landlord or a student.

Screenshots:

Table 1: USER

Server: MySQL:3306 » Database: gatorhousing » Table: user										
Browse Structure SQL Search Insert Export Import Privileges Op										
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action	
<input type="checkbox"/>	1	uid	int(7)		No	None			Change	Drop  More
<input type="checkbox"/>	2	username	varchar(15) latin1_swedish_ci		No	None			Change	Drop  More
<input type="checkbox"/>	3	password	int(20)		No	None			Change	Drop  More
<input type="checkbox"/>	4	email	varchar(50) latin1_swedish_ci		No	None			Change	Drop  More
<input type="checkbox"/>	5	dob	date		No	None			Change	Drop  More
<input type="checkbox"/>	6	fname	varchar(15) latin1_swedish_ci		No	None			Change	Drop  More
<input type="checkbox"/>	7	lname	varchar(15) latin1_swedish_ci		No	None			Change	Drop  More
<input type="checkbox"/>	8	type	tinyint(1)		No	None			Change	Drop  More

Table 2: PROPE

Server: MySQL:3306 » Database: gatorhousing » Table: property										
Browse Structure SQL Search Insert Export Import Privileges Operations										
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action	
<input type="checkbox"/>	1	pid	int(11)		No	None			Change	Drop  More
<input type="checkbox"/>	2	price	int(11)		No	None			Change	Drop  More
<input type="checkbox"/>	3	type	varchar(10) latin1_swedish_ci		No	None			Change	Drop  More
<input type="checkbox"/>	4	address	varchar(30) latin1_swedish_ci		No	None			Change	Drop  More
<input type="checkbox"/>	5	nBed	int(11)		No	None			Change	Drop  More
<input type="checkbox"/>	6	nBath	int(11)		No	None			Change	Drop  More
<input type="checkbox"/>	7	description	text latin1_swedish_ci		No	None			Change	Drop  More
<input type="checkbox"/>	8	zipCode	int(11)		No	None			Change	Drop  More
<input type="checkbox"/>	9	image	blob		No	None			Change	Drop  More
<input type="checkbox"/>	10	googleLocation	varchar(100) latin1_swedish_ci		No	None			Change	Drop  More
<input type="checkbox"/>	11	rentalStatus	tinyint(1)		No	None			Change	Drop  More

RTY

Table 3: MESSAGING

Server: MySQL:3306 » Database: gatorhousing » Table: messaging										
Browse Structure SQL Search Insert Export Import Privileges Op										
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action	
<input type="checkbox"/>	1	message	text latin1_swedish_ci		No	None			Change	Drop  More
<input type="checkbox"/>	2	status	int(11)		No	None			Change	Drop  More

Based on the user's input, we're going to display every post containing the user's entry using like %, contains and free text functions. Depending on the nature of the text entered (digits or letters), we going to prioritize the attributes to fetch first. For example, if the user enters "94116", the search is going to look up at the Zip code values, then addresses, and so on. In case the entry is "calm apartment" then the search should begin from the description to the address and so on.

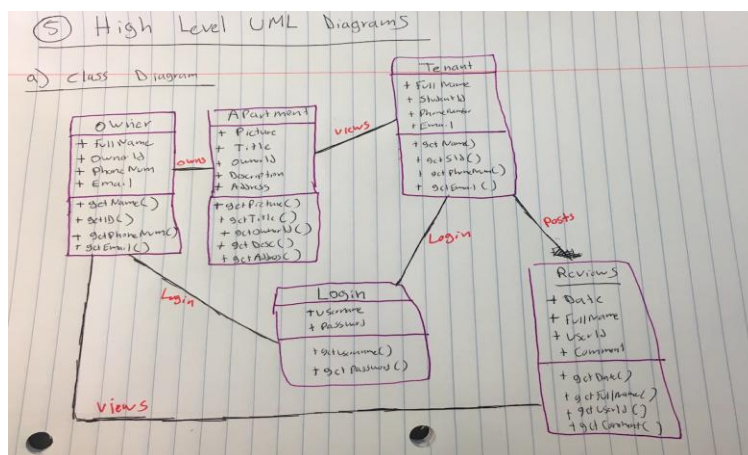
In case one of the filters in selected, then the search time and complexity is going to be narrowed down, and instead of fetching all the table, we're going to be looking at the specific data where the attribute "type" has a value of the filter selected. i.e.: Filter: Apartments, then the search is going to look up at the information in the properties with type == 'apartments'.

For the media storage, we decided on using the blobs rather than saving the original data in the directory file. Using blobs has the advantage of saving memory space and security purposes.

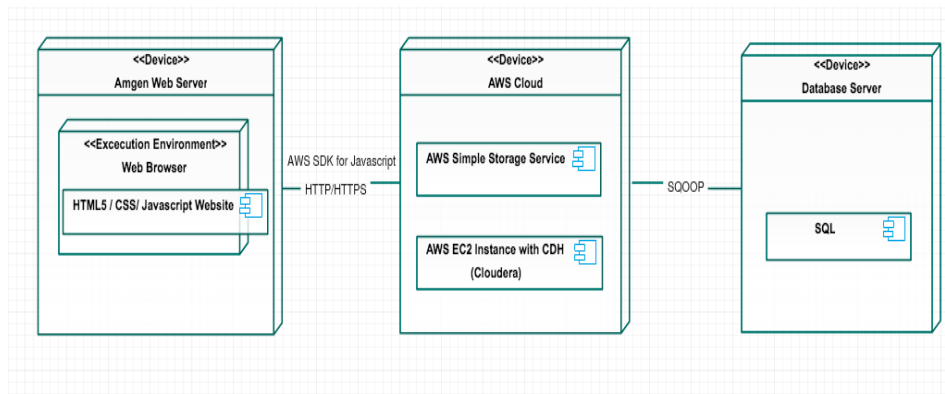
For now, we are not using any API of our own, or implementing features like ranting or ranking. If any changes, it will be mentioned an highlighted in coming milestones.

## 5. High Level UML Diagrams : (Class Diagram)

### 5.1 High Level UML Class Diagram



## 5.2 UML Component and Deployment Diagram



## **6. Actual risks for our project :**

### Skill Risks :

- Each person in our team have different skill sets. Trying to filter out skills according to each person's learning curve is a challenge.
- There are several front end technologies and not many people in the team are aware of each.
- Some of us have not worked on projects of this scale.

Solution: We will make sure every person is on the same page in terms of technologies and no one is lagging behind. We give each other a deadline to learn that particular skill by.

### Schedule Risks:

- Our schedule is tough to determine during spring break as it changes for everyone
- Due to emergencies and last minute tests, our meetings suffer since the entire team cannot be present at that time.

Solution: We do not have common times that align so we always meet before and after class to discuss milestones. We also divide responsibilities

in smaller teams such as backend and frontend so that the entire team doesn't have to meet up at the same time.

#### Technical Risks:

- Insufficient knowledge about frameworks is a continuous technical risk that sometimes needs to be worked on for days at a stretch to solve.
- The integration and hosting of a full-fledged project is difficult at this point because no one has full stack experience.
- Our backend team that handles databases may have different attributes for their column names on their computers. For example, rates could be mentioned as price on the other person's computer.

Solution: We will take the guidance of our CTO Anthony, for any technical risks. Also, we are using popular technologies for which there is help on the internet.

#### Teamwork Risks:

- There are conflicts among team members during meetings for every milestone
- Over expectations and under performance are some aspects that are tough to be managed
- Every person comes from different background and thus it is difficult to collaborate or empathize since we have not spent enough time just hanging out.

Solutions: Whenever we are stuck in an argument about design or technologies, we vote to resolve it. We also hangout after meetings to get to know each other.

#### Legal/Content Risks and Solutions:

- We are not liable for any death, theft, injury, cost, expense or inconvenience arising from any deals made.
- We will be aware of not using the actual location of homes in our house listings, thereby compromising the security of the house owners.
- There will be no obscene content on our website. Everything will be respectful.

## **7. Project Management**

It was really hard for us to manage the time to meet altogether and do the project before because we all have different schedules. Yet, we decided to use Trello which our professor, Petkovic introduced us in class. It is a very useful tool to manage the tasks and catch up with all the team groups' work. We can track the progress, organize the tasks and minimize the project assignments with simple tools. In fact, it gives a visual overview of what is being work on and who is working on it. Therefore, we are planning to use Trello since we get a lot of benefits from it. Also it is sort of motivation for us that we can see how far we have done our project.

