CSC 648-848 Software Engineering Summer 2020
Milestone 4

Freshwater Rentals

Rent, sell, buy housing/apartment website

Available on desktop and mobile.

Team 1 : San Francisco State Coders (SFSC)

Luis Chumpitaz	**Team Lead**	**lchumpit@mail.sfsu.edu**
Garrett Peuse	**Backend Lead**	
Gouri Jamakhandi	**Frontend Lead**	
Katthak Shah	**Github Master**	
Nathan Hildum		
Zachary Ma		
Xinwei Fan		

Submitted: Aug 4, 2020

1) Product summary

With the increasing number of SFSU students and the decreasing availability of living spaces, finding a place to live in San Francisco has never been more critical for students. Our website FreshWater Rentals aims to provide users the ability to find housing opportunities in our current fast paced and competitive market. Registered users on our site can advertise living spaces available for sale or for rent. Other features include a detailed map to campus.

Name of Product: Freshwater Rentals
Our system will allow users to:

- Search and filter listings by price, number of rooms, pets allow
- Post listing
- Message to landlords
- Review their listing and messages on the dashboard
- Sign up by email address, username and password
- Login by username and password

Our system will allow administrators to:

- Review posting
- Delete malicious or irrelevant posting
- Delete malicious message

Our website URL: **http://freshwater.city/**

2) Usability test plan – max 2 pages

-Test Objectives

On this test we shall test the functionality of the homepage and its main features, such as looking for housing listings, the search, filtering and sorting functionalities in which this depends on. This test will ensure that each of the components work as intended and to fix potential issues as they arise on the evaluations.

-Test background and setup

The set up is first to deploy the web application into our production environment, the cloud, and make sure it runs similarly to our local environment. Making sure our web and proxy server communicate to each other by routing each request to the proper API. We need to ensure that our web server is linked to our url domain http://freshwater.city using NGINX and proper configurations for it to display the right application rendered in our browser. Our intended users are property owners in the San Francisco Bay Area and college students and faculty from San Francisco State University. The purpose of this user testing is to ensure the end user accomplishes the tasks he/she/them wants to achieve by using our application, and that we provide the right functionalities for this to happen, from loading the page, exploring different listings, narrow down a search result and explore results of interest of the user.

-Usability Task description

Tasks:

*Login to the freshwater.city and explore the current postings

*Search for a specific type of house/apartment

*Sort postings by your main concern when looking for a place, price, bedrooms, distance from sfsu, etc.

*Mix sorting and filters as you may wish.

We would measure the effectiveness of these tasks by checking that components work, links work, there is no crash or unwanted page displayed. Make sure that the results done by the seach, filters and sorting return the proper values of listings.

We would measure the efficiency of our application by checking how fast a user is able to familiarize with the application. The speed in which the different queries are performed, and the accessibility of this from the user perspective.

Questions:

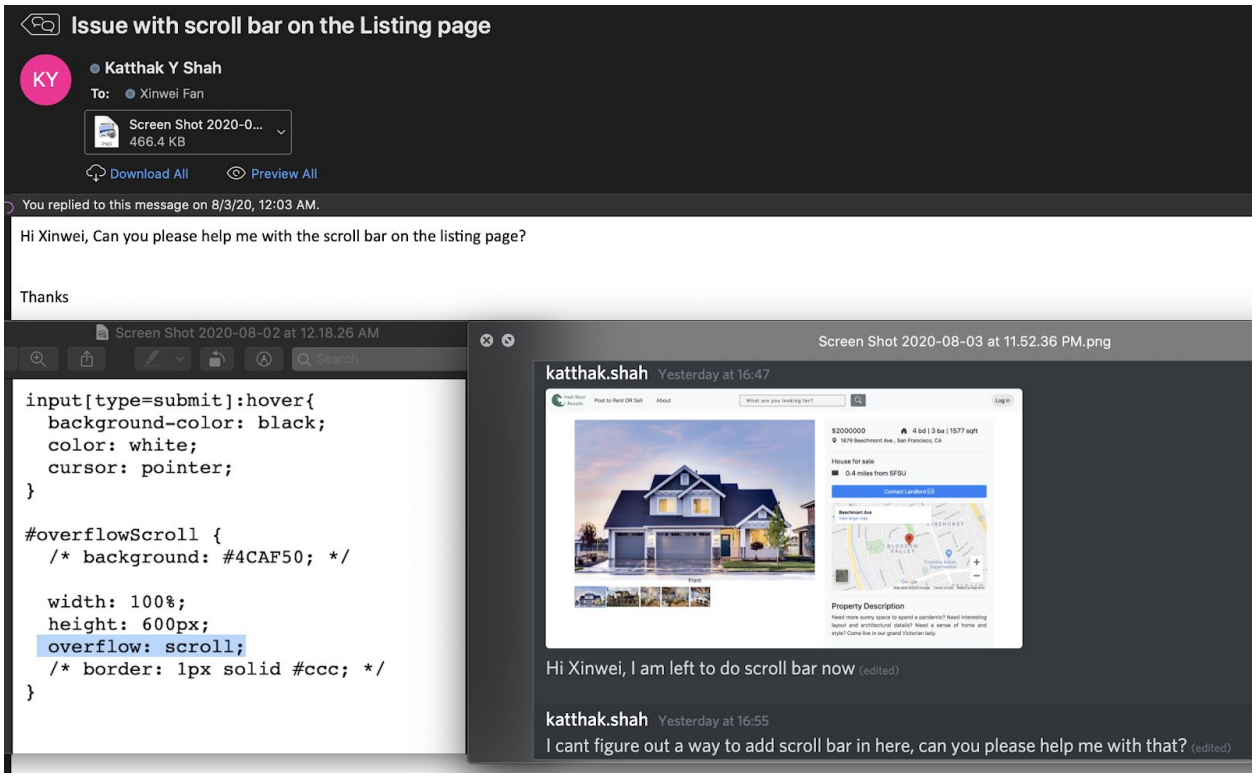
- How easy was it to accomplish each of the tasks.
- Do the current functionalities help you find a desired type of housing?
- How does this functionality perform compared to previous websites you used?

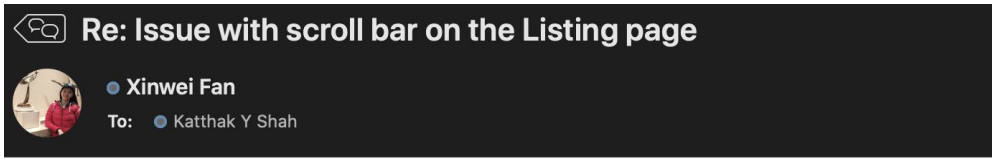
3) QA test plan - max 2 pages

URL: http://freshwater.city/

Test #	Test Title	Description	Test Input	Expected Correct Output	Test Results
1	Search field on Each page	Type % like in search field	Type “nice” in search field	Get 2 results, both of them have “nice” in their description	**Pass**
2	Filters on Home Page	Select different combination of the filters available	Select “House” and “Rent” filters	Get 1 result, which is a House to Rent	**Pass**
3	Search and Filter combination on Home Page	Select different combination of the filters and search	Type “nice” in search field and select “Apartment” to “Rent” in filter	Get 1 result for Apartment to Rent, which has “nice” in the description	**Pass**
4	Sort Functionality Test	Choose different sorting options available	Select “Distance to SFSU” sorting option	All the 5 results should be sorted by Distance to SFSU, shortest distance listing being first	**Pass**
5	Listing more info	Click on one of the listings to check out more info about that listing	Click on “Apartment to Rent” listing with 800 sqft	The more information about this listing should open with correct info. About this listing such as Rent($3000/mo), Distance(1.0 mile from SFSU), map, street address	**Pass**

4) Code Review:





Hey Katthak,
Thanks for your UI code review. The scroll bar is working now. I fixed the CSS and rewrote the HTML code. :)

```
        <lingallery  :addons="{ enableLargeView: true }" :mob
    </b-col>

    <b-col sm="5">
      <!-- scroll bar -->
    <div id="overflowScroll">
      <b-card border-variant="light" bg-variant="light">
      <!-- <h4 >Property Details</h4> -->
      <b-row>
        <b-col cols="6">
```

```
#overflowScroll {
  /* background: #4CAF50; */

  width: 100%;
  height: 600px;
  overflow: scroll;
  /* border: 1px solid #ccc; */
}
```

Best
Xinwei

5) Self-check on best practices for security

Assets we are protecting are:

- User Information via
    - Encrypted Passwords
    - Website Restriction
- General Server Protection via
    - Mysql injection attacks
    - Protection from File uploads

For password encryption we are using Python Flask's Security framework along with Bcrypt for hashing. This process is dual encryption, one on the frontend side and one on the backend side with of course the second hashing of the password stored in the Mysql database. Passwords must be at least 6 characters long upon creation and email must end with @sfsu.edu, @mail.sfsu.edu or @sfsu.com. Also to control the flow of data from authorized users we are deploying session handling that tracks users behavior and restricts users without the correct credentials to certain sites.

For mysql injection attacks we are using sqlAlchemy framework to protect from any for sql injection attack.

For a file uploading, we limit the size and file type to only 'png', 'jpg', 'jpeg', 'gif' to limit malicious script being sent to the server.

6) Self-check: Adherence to original Non-functional specs – performed by team leads

Original Non-Functional Specs	Status
1. Application shall be developed, tested and deployed using tools and servers approved by Class CTO and as agreed in M0 (some may be provided in the class, some may be chosen by the student team but all tools and servers have to be approved by class CTO).	**DONE**
2. Application shall be optimized for standard desktop/laptop browsers e.g. must render correctly on the two latest versions of two major browsers.	**DONE**
3. Selected application functions must render well on mobile devices	**DONE**
4. Data shall be stored in the team's chosen database technology on the team's deployment server.	**DONE**
5. No more than 50 concurrent users shall be accessing the application at any time.	
6. Privacy of users shall be protected and all privacy policies will be appropriately communicated to the users.	**DONE**
7. The language used shall be English (no localization needed)	**DONE**
8. Application shall be very easy to use and intuitive.	**DONE**
9. Google analytics shall be used.	
10. No e-mail clients shall be allowed. Interested users can only message to sellers vis in-site messaging. One round of messaging (from user to seller) is enough for this application	
11. Pay functionality, if any (e.g. paying for goods and services) shall not be implemented nor simulated in UI.	**DONE**
12. Site security: basic best practices shall be applied (as covered in the class) for main data items	**DONE**
13. Media formats shall be standard as used in the market today	**DONE**

14. Modern SE processes and practices shall be used as specified in the class, including collaborative and continuous SW development	
15. The website shall prominently display the following exact text on all pages "SFSU Software Engineering Project CSC 648-848, Summer 2020. For Demonstration Only" at the top of the WWW page. (Important so as to not confuse this with a real application)	**DONE**