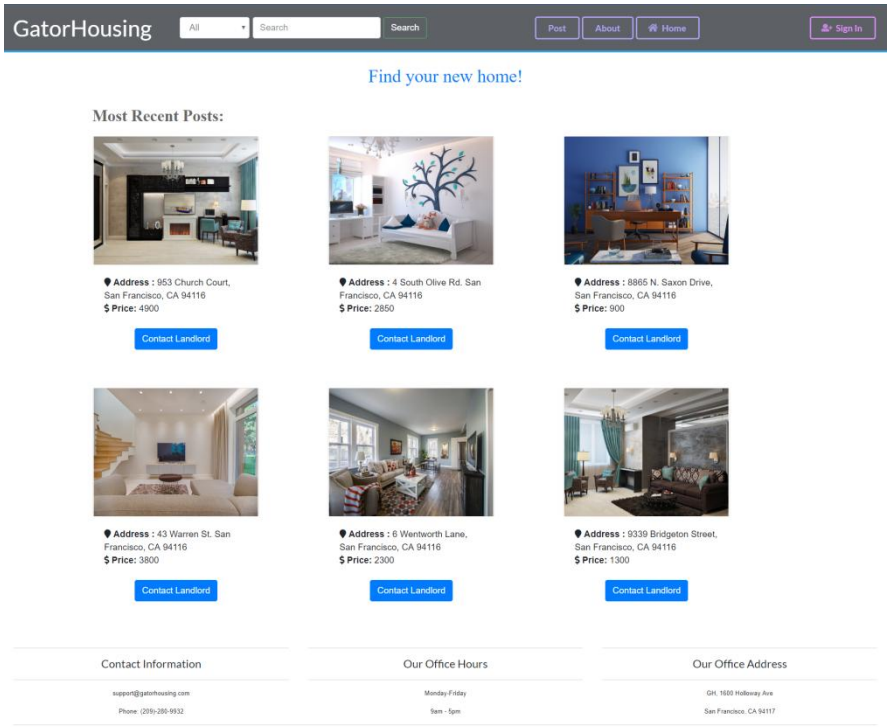
“SW Engineering CSC648/848 Spring 2019”

“GATORHOUSING WEBSITE”

TEAM 03



Dhwan Shah (Team Lead) : dshah2@mail.sfsu.edu

Saad Bouayad (Backend Lead)
Aye Win Sandy,
Rajvi Shah,
Mubarak Akinbola,
Dawit Ayele (Front End Lead),
Ahmad Rangeen (GitHub Master)

MILESTONE 4

Date	Submitted	Time Taken
5/7/2019	Done ☺	7 days
5/15/2019	Done with Changes Given	3 days

1. Product Summary:

"GATORHOUSING" website is basically developed by the engineers for the betterment of SF STATE students where they can rent a property (apartment, condo). This will help SF State students to get cheaper housing and will also give relief from being waitlisted for on-campus housing. SF State students can basically login or register and see the posts posted from landlord. The unique about the website is it's only for SF State students. Priority 1 Functions List:

1) Home Page:
    - The attractive home page shall show the recent posts from database with having search, post, about and login features
2) Login/Registration:
    - Users shall be able to create accounts and log into them, and remained logged in during the duration of session
    - Stored Passwords will be encrypted
3) Search with Results:
    - Users shall be able to search by text(address, city, keywords) or and category and get redirected to results page
4) Post Details:
    - Each post shall have Contact Landlord and Show Details button where the user can click for more details of the post like description of the post or send a message to landlord.
5) Messaging :
    - Users shall send a one-way message to landlord showing the interest in the posted apartment/condo etc
6) Details Page:
    - Users shall redirected to detail page on clicking show details of any interested page where the user can know more about the post
7) Admin Dashboard:
    - The role of admin is to approve the content posted by landlord

URL: http://ec2-54-193-123-249.us-west-1.compute.amazonaws.com/front2/index.php

2. Usability test plan:

Test Objectives:

"GatorHousing" is a rental web application that provides a platform for San Francisco State University students to easily find affordable rent around the campus and nearby neighborhoods. One of the major features GatorHousing web application offers is "the search", and there is no doubt that this is a central feature to the application itself, because it produces the results which then will be used as an asset for different parts of the web application. But also to the student/renter because it helps them browse the specific data they want with the help of the filter which narrows down the listing pool and offers precise search results. So, this usability test raises few questions concerning integrated search: Is it easy to use? Is it efficient? Does it respect Fitt's law?

Test Description:

Our web application is deployed on an AWS EC2 instance that has been formerly set up. The instance includes installation and configuration of PHP, MySQL, Apache and phpMyAdmin to make database manipulation easy and accessible by different members of the team, both frontend and backend teams.

To access GatorHousing web application, users will need a browser (ideally, Chrome, Firefox or IE) and the link to the starting point of our application which is the home page. Users do not need to sign up or log in in order to search for listings, however, they will be prompt to a sign up or log in page in case they want to contact the landlord. Below is the URL to GatorHousing homepage.

URL starting point: http://ec2-54-193-123-249.us-west-1.compute.amazonaws.com/front2/index.php

The test should cover the time taken to send a query and have the correct results displayed in less than 6 seconds, the readability and visibility of all the buttons and text contained in the web page, the alignment of various components of the website and the coherent esthetic traits highlighted in the navigation bar, and the way results display, and finally the grouping of different elements of the search feature(dropdown housing type, search input) and their fulfillment to Fitt's predictive law.

⇨ Usability Task Description:

Before filling out Likert test, the testers should be able to pick a housing type, and enter an input in the search bar, the input can be anything related to the listing, but usually, users tend to use keywords as city, zip code, neighborhood, or phrases like “close to the bart” or “close to the bus stop”. For now, our database includes listings from the following zip codes for test only (94110 and 94116). The testers should pay attention to how much time an average query (with 2 or 3 parameters) take and if the images upload at the same time as the text.

Questionnaire:

1/ The search is visible and its position in the webpage is easily predictable (user friendly).

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
O	O	O	O	O

2/The search is efficient and the results are as per the expectation of user.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
O	O	O	O	O

3/On entering zip 94110 I get all the results having zipcode 94110.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
O	O	O	O	O

4/On mentioning Condo in dropdown/Search I get all the results of Condo.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
O	O	O	O	O

Any Comments :

Comments…

3) QA Test Plan

Test Objectives: Objective for testing is to make sure the search function using the search bar on the Gator-Housing homepage is working so that a first time user should be able to find rooms/apartment they are looking for easily, and that any search functions does not turn up empty or create any kind of error messages, even when the search bar is empty. Also to make sure there are no serious bugs. The testing can be done with either registered or unregistered users, but in this testing we will use an unregistered user as the tester for simplicity. Testers are told specific instructions what to do so that they know if the end result passes or fails.

Hardware and Software setup

HW setup:

1. Computer or mobile connect to the Internet.
2. Open chosen web browser. (The website will support the latest version of Google Chrome as well as mobile.)

SW setup:

1. Go to the following link:
2. Create profile if needed
3. Expand the menu to show search bar
4. Enter input in search bar

Feature to be tested:

We will be testing the search functionality, specifically the results that should be correctly displayed after the search has been executed with the user's input.

⇨ Test Plan :

Test Number	Browser	Description	Input	Expected Output	Pass/Fail
1	Chrome	Search apartments	Select “Apartment” from Dropdown menu	All the results would be having housing type of Apartments	Pass
2	Firefox	Search apartments	Select “Apartment” from Dropdown	All the results would be having housing type of Apartments	Pass
3	Chrome	Test input Validation	“!@#$%^&*”	Form will not submit	Pass
4	Firefox	Test input Validation	“!@#$%^&*”	Form will not submit	Pass
5	Chrome	Test inputlength	Enter 50 instance of ‘A’	Form will not submit	Fail
6	Firefox	Test%likeinsearchfor title field			Pass
7	Chrome	User Registration	User info	Successfully able to create user Account	Pass
8	Chrome	Contact Landlord		Successfully able to contact Landlord	Failed
9	Chrome	Filter ByDistance		Successfully filter properties by distance from SFSU	Passed
10	Chrome	Post Property	Property Description and images	Successfully post property	Passed

4) Code Review:

Code Review Request 2

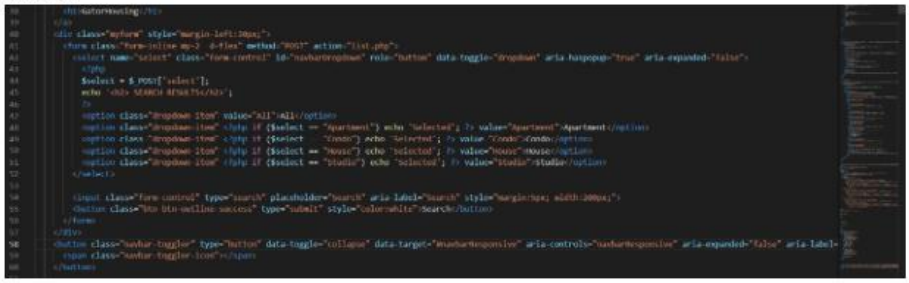
MA

Mubarak Akinbola
Tue 5/7/2019 1:44 PM
Saad Bouayad

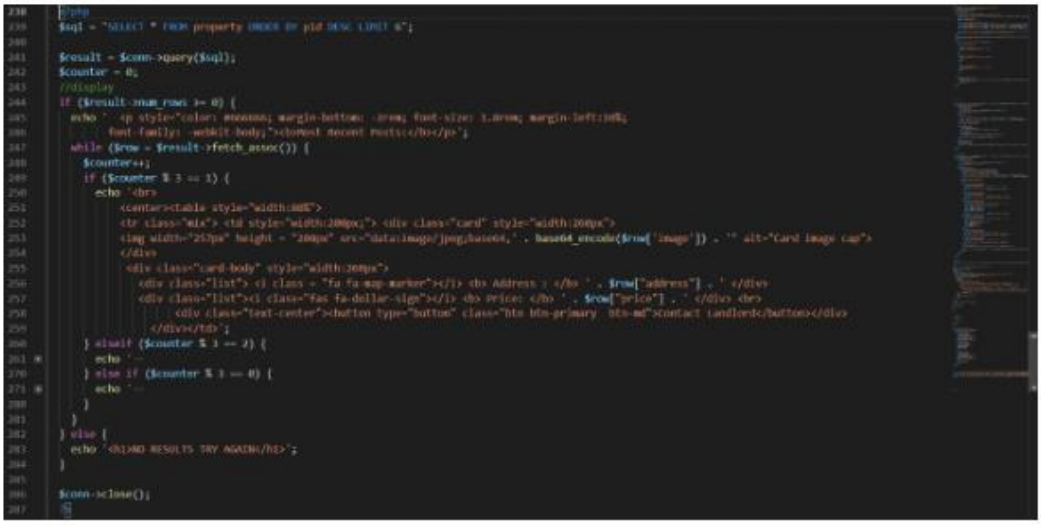
Hello Mr. Bouayad,

Thank you to give me the opportunity to review the Search code of our project. I have to say you did a pretty neat job, I totally no concerns about coding style and how the search was implemented. I like how the code is to understand with your use of descriptive variable names. Moreover, I like how you used modulus to display the result of the search. I would just add some comments in the code, overall it's a job well done.

Search



Post-Search



5) Self-Check On Best Practices for Security:

Assets being protected:

1. Username and password – We are protecting this because many people can end up using the same username and password that they use for other important websites. So, it is very important for us to protect the password so it can't be accessed by anyone but the user. The password is encrypted before it is stored in the database.
2. Posted items and images – We are protecting this because user created posts contain images posted by the user, the images are protected in our database and can't be edited. Additionally, users can only see approved posts, so that any posts containing explicit content will not be approved and not shown to users. Finally, users will only be able to access their own posts on the user dashboard, and not posts created by other users.

Input Validation:

1. Image Protection: We are using BLOBS for protecting the image files.

2. Input Validation: The user is allowed to enter the string of 40 characters in search input box. The user can strictly not exceed the limit of 40 alphanumerical characters because we have mentioned HTML property 'maxlength' for all the inputs of "Search".

6) Non-Functional Requirements:

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). - ON TRACK
2. Application shall be optimized for standard desktop/laptop browsers e.g. must render correctly on the two latest versions of two major browsers – ON TRACK
3. Selected application functions must render well on mobile devices – ON TRACK
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 – ON TRACK
6. Privacy of users shall be protected and all privacy policies will be appropriately communicated to the users. – ON TRACK
7. The language used shall be English. – DONE
8. Application shall be very easy to use and intuitive. – DONE
9. Google analytics shall be added – ISSUE
10. No e-mail clients shall be allowed – DONE
11. Pay functionality, if any (e.g. paying for goods and services) shall not be implemented nor simulated. – DONE
12. Site security: basic best practices shall be applied (as covered in the class) – DONE
13. Before posted live, all content (e.g. apartment listings and images) must be approved by site administrator – DONE
14. Modern SE processes and practices shall be used as specified in the class, including collaborative and continuous SW development – DONE
15. The website shall prominently display the following exact text on all pages "SFSU Software Engineering Project CSC 648-848, Spring 2019. For Demonstration Only" at the top of the WWW page. (Important so as to not confuse this with a real application). – ON TRACK