# Final Project for SW Engineering Class
# CSC 648-848 Spring 2019

---

## GatorHouse

## Team 14

Davis Hoang ( Davis.hoang2015@gmail.com ) ( Team Lead / Github Master )

### Front-End

Jonathan Fox ( Front-End Lead )
Kevin Mitsuda
Jarrett Lee

### Back-End

Hashim Jacobs ( Back-End Lead )
Eric Chen
Shalaka Aigal

DEMO URL : http://ec2-52-53-199-247.us-west-1.compute.amazonaws.com

05/21/2019

## Product summary

Gatorhouse is a service that anyone can use for hassle-free house-hunting or house-selling. If you are a student, faculty member, or just someone searching for a new living situation near San Francisco State University, Gatorhouse is your new best friend.

1. **Simple Search**

   A simple search and a couple of clicks will get you that much closer to acquiring your ideal living space. With Gatorhouse, you're able to connect with multiple landlords with a high degree of simplicity in just minutes.

2. **Upload Property**

   If you're a seller, post your property with brief details, sit back, and watch the buyers/renters flood in. Gatorhouse is here to do the hard work so that everyone can live their best life.

3. **Admin control**

   Gatorhouse enforces reliable house listings. Our admin will be able to approve or reject uploaded listing types before they go live within 24 hours.

**You can reach us at:** http://ec2-52-53-199-247.us-west-1.compute.amazonaws.com

# SW Engineering CSC648/848 Spring 2019

---

## SFSU House Me


Team 14

Davis Hoang ( Davis.hoang2015@gmail.com ) ( Team Lead / Github Master )

Front-End

Jonathan Fox ( Front-End Lead )
Kevin Mitsuda
Jarrett Lee

Back-End

Hashim Jacobs ( Back-End Lead )
Eric Chen
Shalaka Aigal


## Milestone 1

02/07/2019

| Revision Table | | |
|---|---|---|
| 03/07/2019 | 1.0 | Initial Document |
| 03/18/2019 | 2.0 | Revised Document |

## Executive Summary

The lack of access to affordable housing and the increase in housing instability in the bay area has caused many San Francisco State University students to struggling to secure a place to live. According to the president and chief financial officer at San Francisco State, about 3,000 students are waitlisted for room and board every Fall semester. San Francisco State will be unable to accomodate room and board to every student because of the insufficient number of available spaces. This will cause many students to look at other possibilities for housing.

We are looking to develop a housing website for SFSU students to help ease the process of securing a place to live. Students will be able to quickly browse through available housing spaces in the bay area. Students will have the option to search by price or by distance or by housing type. Student will be able to browse through the our website without having to sign in. If a student wishes to contact the landlord then they will be prompted to input their login credentials. Furthermore, our web application will allow homeowners and landlords to list their available housing spaces. They will be able to upload photos, videos, and information of their vacant homes. Homeowners and landlords will have the options to accept or decline inquires about their available spaces from users.

Our site aims to serve a variety people, students who are looking for housing or homeowners who are looking to lease their house. The use of our site will be perfect for students who need an easier process to secure housing. In addition, students will be able to find more affordable housing through homeowners in the bay area.

## Personae and main User Cases

### Mary: Student looking for room

About Mary:

- Very busy
- Is not local
- Familiar with WWW tools and apps.
- Interested in the city
- Looking to meet new people

Goals and scenario:

- Freshman/ new to the area, looking for a place to rent that makes it convenient to attend SFSU.
- Looking for a web app that clean shows available rooms and makes it easy to find apartments in her parameters.
- If she finds an issue on the site, she is likely to report it to the administration of the site.

Use case:

Non Registered User Browsing Site

1. Mary wants to quickly find a room near SFSU to rent out for the year. So she enters the site and searches by distance, cost, and type and sees a number of listings that fulfill such requirements. She is then able to browse the listings with additional filtering options.

Non Registered User Contacting Poster

2. Mary has found a listing/ room that she would like to get more information from the poster, she will then be prompted to register before she can message the poster.

## Carol: Student looking for roommates

About Carol:

- Very busy
- Familiar with the area
- Familiar with WWW tools and apps
- Loves the city
- Outgoing

Goals and scenario:

- Looking for a fellow student that shares similar hobbies to share a room with/ split the bill.
- Is very busy, so will not be monitoring the posting frequently

Use case:

Local Student Creates a Listing to Find Roommates

3. Carol will want to post a listing onto the site, indicating that they are looking for a roommate for a certain apartment. They will be prompted to register or log into the site. From there, there will be a form to input information such as location and price for the location along with an option indicating that the poster is looking for roommates.

## Harold: Landowners looking for renters


About Harold:

- Lots of Free time
- Unfamiliar with WWW apps/ tools
- Owns multiple homes

Goals and scenario:

- Looking for someone to rent out unused rooms/ apartments to.
- Harold is unfamiliar with WWW apps, so he will require a web app that makes it simple to post listings without much effort.
- Spend less time having to manage their postings on the site.

Use case:

Landowner Posts Listings to Site

4. Harold will want to post their unused rooms onto the site. There will be an option to post listings onto the site with a simple form. This form will require basic information about the listing and images of the property. Before the form is complete they will be required to register onto the site, i.e. lazy registration.

## Administration: Admin for site

About Administration:

- Working on or maintaining the site
- Familiar with WWW tools
- Will only be able to work for a few hours a day

Goals and scenario:

- Looking for a UI that allows easy maintenance and monitoring of the site.
- Listings need to be processed in a way to allow them to be approved and easily taken down by a single user/ admin.

Use case:
Admin's Unique Page to Quickly Maintain Site Needs

5. Administration will want to log into the site and be shown a page that includes a list of pending listings that have yet to go live and reported listings from users. From there, they will be able to take down or approve certain listings quickly.

## List of main data items and entities

- Unregistered User : The unregistered user will be able to browse through all the property listings but will not be allowed to post any listing.

- Registered User : The registered user will have a user account on the website will be able to view all the property listings and can also post the listing provided they are logged in on the site. Registered Users can be both Landlord/Seller and Tenant.

- Landlord/Seller: A registered user who has a property to rent and can post the property on the website.

- Tenant: A SFSU student looking for a place to rent.

- Administrator : Users that have special privileges, and have the ability to remove posts from the site, remove items, issue warnings and bans from the site, and generally enforce the Code of Conduct for the site. Administrators also are responsible for helping users when needed.

- Property : The main data item is property listed for sale, or for purchase on the site. The following are the types of property which can be listed on the site.
    - House : Individual house with 1 or more rooms to be rented.
    - Apartment: Apartment with 1 or more rooms to be rented.
    - Room: Single or shared room to rent.

- Posting, registration, and search shall be main function.
- Registration details: User's name, email, contact number will be collected from the users during registration.

# Functional Requirements

## Unregistered Users:

1.1 Unregistered users shall be able to view housing types on the home page.
1.2 Unregistered users shall be able to sort by house type, price and distance.
1.3 Unregistered users shall be able to see a photos and description of the house.
1.4 Unregistered users shall be able to sign up with email and password.
1.5 Unregistered users shall be prompted to sign up before contacting homeowners.

## Registered Users:

2.1 Registered users shall have the same functionality as an unregistered user.
2.2 Registered users shall be able to login with their email and password.
2.3 Registered users shall be able to contact homeowners about housing inquiry.
2.4 Registered users shall be able to post listing of their housing type.
2.5 Registered users shall be able to post photos, descriptions and videos of the house

## Admin:

3.1 Admin users shall have the same functionality as a registered user
3.2 Admin users shall review the content and description of all uploaded house listings.
3.3 Admin users shall be able to add, delete and edit house listings.
3.4 Admin users shall be able to delete or block registered users.

## House listings:

4.1 House listings shall be able to upload photos and videos.
4.2 House listings shall be able to add a description about the house.
4.3 House listing shall show the owner of the listing.
4.4 House listing shall be viewed by all users.
4.5 House listing shall be viewed on the home page.

## 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).
2. Application shall be optimized for standard desktop/laptop browsers (Google Chrome and Mozilla Firefox)
3. Selected application functions must render well on mobile devices
4. Data shall be stored in the team's chosen database technology: Database: PostgreSQL 10.12-10.14
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.
7. The language used shall be English.
8. Application shall be very easy to use and intuitive.
9. Google analytics shall be added
10. No e-mail clients shall be allowed
11. Pay functionality, if any (e.g. paying for goods and services) shall not be implemented nor simulated.
12. Site security: basic best practices shall be applied (as covered in the class)
13. Before posted live, all content (e.g. apartment listings and images) must be approved by site administrator
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, Spring 2019. For Demonstration Only"* at the top of the WWW page. (Important so as to not confuse this with a real application).

Competitive Analysis

| Feature | Zillow | Facebook Marketplace | Craigslist | Our Product |
|---|---|---|---|---|
| Text Search | + | + | + | - |
| Tag Filter | - | - | - | + |
| Housing List View | + | ++ | + | ++ |
| Housing Map View | + | - | - | - |
| List Home for Rent/Sale | + | ++ | + | + |
| Distance/Time to Campus | - | - | - | + |

Zillow, Facebook Marketplace, and Craigslist are several examples of other products that provide a similar service to what we are providing. The advantage of our website is mainly focused on providing housing services for SFSU students. Zillow has a large map view that takes up a large amount of screen space. SFSU students are generally looking in the same area, so that kind of map view is unnecessary for us. The housing view of Facebook Marketplace is very simplistic and straightforward, without leaving out any important information. It is much more streamlined than Zillow's and Craigslist's housing views. We should build ours in a similar way to make the website easy to use. The other websites use a text search, but we will use a tag filter because it accomplishes a similar task with less work involved.

Our website will have features that specifically benefit SFSU students. We can show a listing's distance from campus, so that students can easily see how long it would take them to get to school.

# High-level system architecture and technologies used

## Deployment Platform

- Server Host: Google Compute Engine
- Operating System: Linux  Ubuntu 18.10 x86_64
- Database: PostgreSQL 10.12-10.14
- Web Server: Node.js v11.9.0
- Server-Side Language: Javascript

## Frameworks:

- Web Server: Node.js v11.9.0
- Server-Side Language: Javascript
- Web Framework: Reactstrap / Bootstrap 4

## Browsers

Our site will be supported on browsers such as Chrome (version 50.0.2661 +) and Safari (version 6 +)

## Team 14

- Davis Hoang ( Team Lead / Github Master / Document Master )
- Jonathan Fox ( Front-End Lead )
- Kevin Mitsuda
- Jarrett Lee
- Hashim Jacobs (  Back-End Lead )
- Eric Chen
- Shalaka aigal

## Checklist

- Team found a time slot to meet outside of the class : **DONE**
- Github master chosen : **DONE**
- Team decided and agreed together on using the listed SW tools and deployment server : **DONE**
- Team ready and able to use the chosen back and front end frameworks and those who need to learn are working on learning and practicing : **DONE**
- Team lead ensured that all team members read the final M1 and agree/understand it before submission : **DONE**
- Github organized as discussed in class (e.g. master branch, development branch, folder for milestone documents etc.) : **DONE**

# SW Engineering CSC648/848 Spring 2019

---

## GatorHouse


Team 14 (Local)

Davis Hoang ( Davis.hoang2015@gmail.com ) ( Team Lead / Github Master )

Front-End

Jonathan Fox ( Front-End Lead )
Kevin Mitsuda
Jarrett Lee

Back-End

Hashim Jacobs ( Back-End Lead )
Eric Chen
Shalaka Aigal


## Milestone 2

3/22/2019

| Revision Table | | |
|---|---|---|
| 03/22/2019 | 1.0 | Initial Document |
| 4/25/2019 | 2.0 | Revised Document |

**Data Definitions:**

- Unregistered User : The unregistered user will be able to browse through all the property listings but will not be allowed to post any listing.

- Registered User : Registered user will have the have the same rights as unregistered users. The registered user will have a user account on the website will be able to view property listings and can also post the listing provided they are logged in on the site. Registered Users can be both Landlord/Seller and Tenant.

- Landlord/Seller: A registered user who has a property to rent and can post the property on the website. Registered user can view property listing.

- Administrator : Users that have special privileges, and have the ability to remove posts from the site, remove items, issue warnings and bans from the site, and generally enforce the Code of Conduct for the site. Administrators also are responsible for helping users when needed.

- Property : The main data item is property listed for rent or sublease. The following are the types of property which can be listed on the site.
    - House : Individual house with 1 or more rooms to be rented.
    - Apartment: Apartment with 1 or more rooms to be rented.
    - Room: Single or shared room to rent.
    - Price: Price of the property
    - Description: The property will have a description of the house listing.

- Registration: The user will have to register in order to post. The registration details will collect the User's name, email, contact number will be collected from the users during registration.

- Message : Registered user can message landlords about house listings. Registered users will be able click a button that sends a message to the homeowner/landlord that they are interested renters.

- Dashboard : Registered users
    - Shall be able to see house listings
    - Shall be able to see their post
    - Shall be able to see messages about house inquires

**Data Definitions (continue):**

- Dashboard : Admin users
    - Shall be able to approve or deny posting
    - Shall be able to see all postings

## Functional Requirements:

Priority 1:

Unregistered Users:

1.1 Users shall be able to view housing types on the home page.
1.2 Users shall be able to sort by house type, price and distance.
1.3 Users shall be able to see a photos and description of the house.
1.4 Users shall be able to sign up with email and password.
1.5 Users shall be prompted to sign up before contacting homeowners.

Registered Users:

2.1 Users shall have the same functionality as an unregistered user.
2.2 Users shall be able to login with their email and password.
2.3 Users shall be able to contact homeowners about housing inquiry.
2.4 Users shall be able to post listing of their housing type.

Admin:

3.1 Users shall have the same functionality as a registered user.
3.2 Users shall review the content and description of all uploaded house listings and approve it before going live.
3.3 Users shall be able to delete or block registered users.

Priority 2:

Users:

4.1 Unregistered Users shall be able to view a video of the house listing.
4.2 Registered Users shall be able to sort housing post by the number of allowed occupants.

Priority 3:

Users:

5.1 Users shall be able to use an Interactive Map to see the surrounding neighborhood.
5.2 Registered Users Shall be able to compare and contrast up to 5 at once posts.

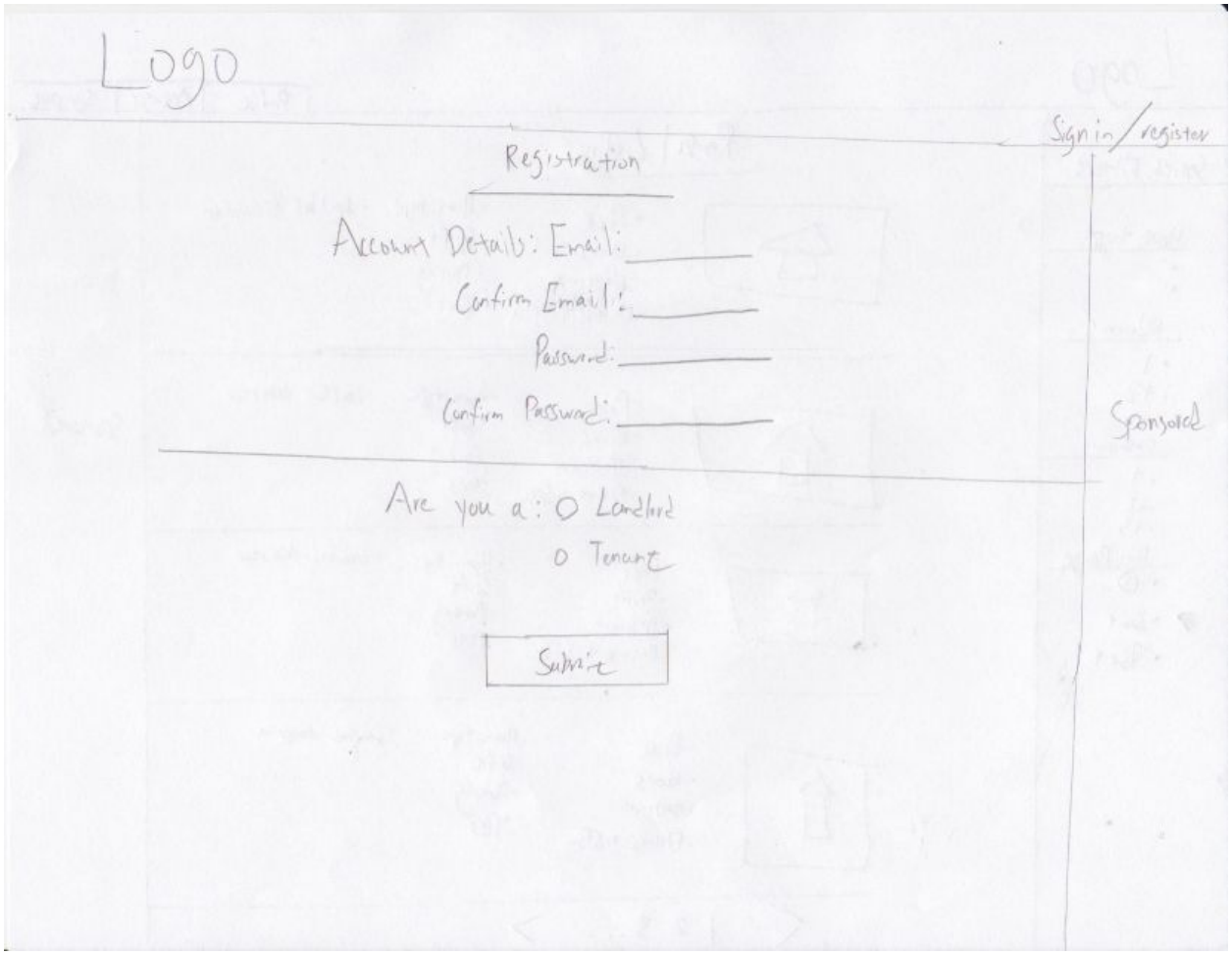
## UI Mockups and Storyboards

Home Page Listings:

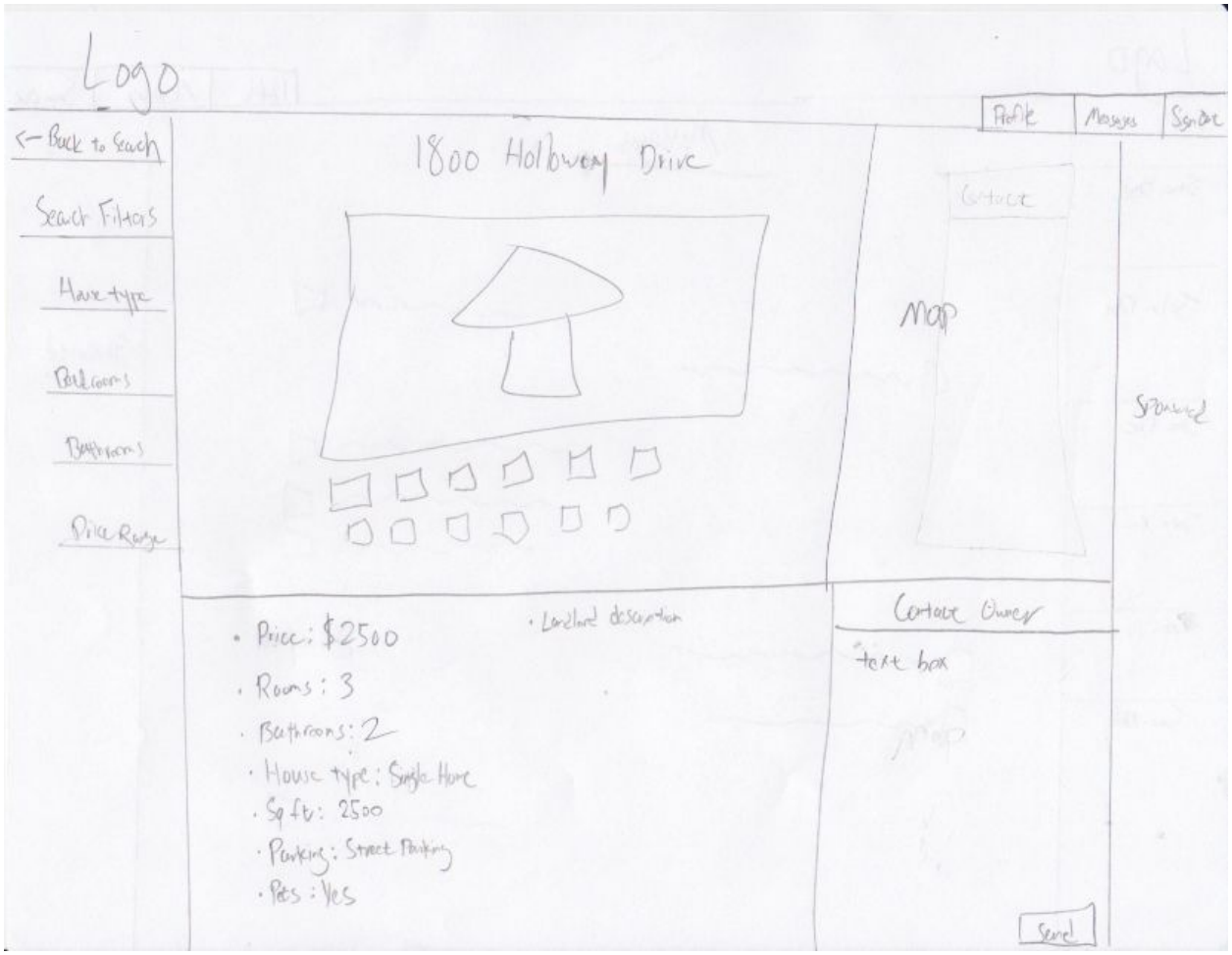

Use Case 1:

Joe is a college students. He is currently a incoming freshman who works part time at starbucks. Joe was planning on moving into the dorms but got waitlisted and is 100th on the waitlist. At this point it seems unlikely that joe will not find housing on campus. Joe decides to take action himself to look for a place before school starts. Joe does onto our app as an unregistered users and browses through available options. Joe finally finds a room he would like to rent out and would like to contact the homeowner. He tries to contact the homeowner and was prompted to register or log-in. Once joe has his own account he will be able to contact the homeowner.

Registration:



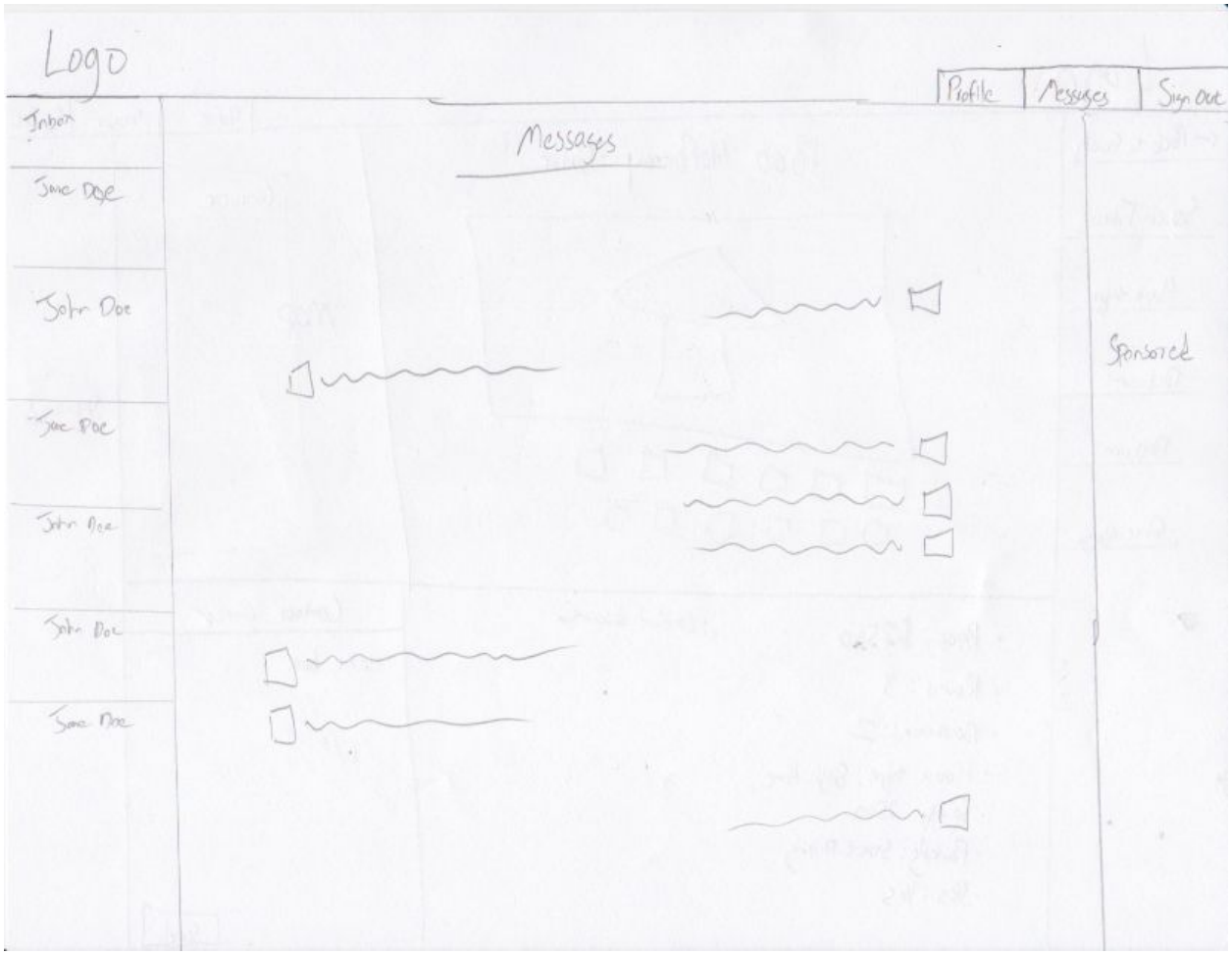
Detailed view of rental property:



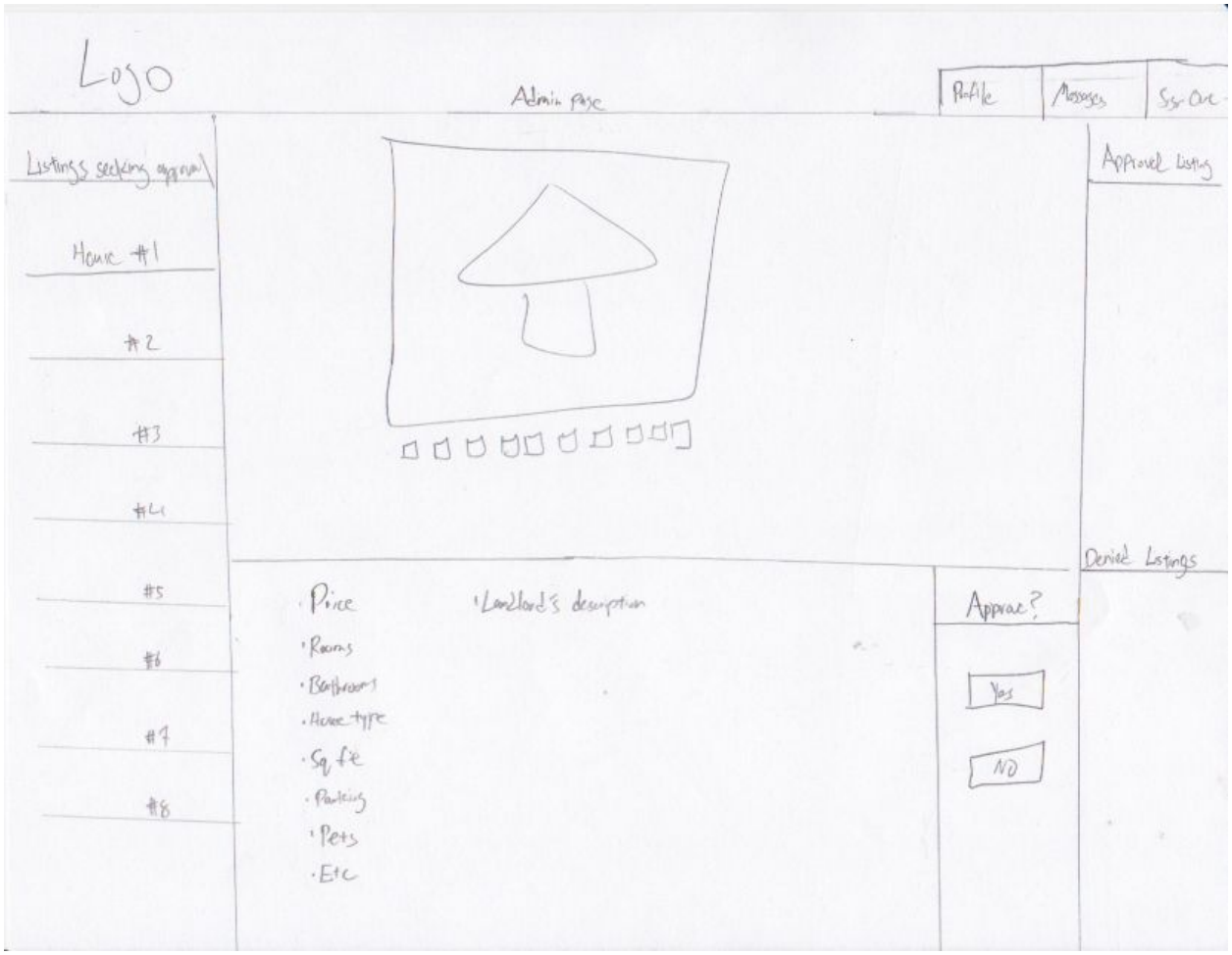

Use Case 2:

Adam is a homeowner. Adam is a 54 year old man who flips houses for a living. He would like to rent out one of his houses he just remodeled. Adam goes into SFSU house me website and registers. Once adam registers, he will then be able to post photos, videos and a description of his house to rent. After adam post a listing of his house he will be able to see any inquiry of registered users who are interested in his house listing on his dashboard.

Messages View:



Admin Page for approving listing:




Use Case 3:

Jasmin is an admin. When she starts her day at work, she opens the homepage of SFSU house me and logs into her account. She then navigates to the house approval section. She the reviews each house listing and its description before deciding to approve or decline the request. She will be able to review accounts to check to for integrity and ethics. If an account does not uphold standards of SFSU house me their account will be deleted or modified.

## High level Architecture, Database Organization

· *DB organization*

PostGreSQL is the database which will be used for the database storage.
Following are the high level tables :

Table Users: userId:int (PK),
userName:String,
email:String,
password: String,
address: String

Table Property: postId:int (PK),
noOfRooms:int,
address:String,
zipcode:int
description:String,
price:double,
imagePath:String,
postDate: DateTime,
userId:int (FK references Users),
propertyId:int(FK references Property)

· *Media storage*:

- File System is the the media storage option . Any Image, Audio or Video files will be placed in a file system and be retrieved from there to display on the webpage.

· *Search/filter architecture and implementation*:

- The header section of our website will have a search box where the user can search for some listing by searching for some keywords. There will be filter criteria like a drop down for Property Type like House, Apartment, Room.

## Database Organization (Continue)

The implementation for the filter section:

- By default none would be checked so we would display some results based on recent posting. When a filter is checked or chosen by the user, the database SQL query will be modified to put the WHERE clause for all the filter and the result will be sent to the frontend to be displayed.

The implementation for the search section:

- The SQL %Like query will be used to implement the search section by querying the database for the search keywords entered by the user.
- Our Homepage will include a search box where if a user just clicks on the search box, then all the listings will be displayed on /search. The user can search by some keywords such by the city, by zip code, etc., then only the specific listings according to the keyword will be displayed using %LIKE SQL query.

Your own APIs (if any): Describe and define at high level any major APIs that you will create other than standard ones provided by tools and frameworks you use
Google Maps Platform:

- Routes - Directions API: Provides directions for transit, biking, driving, and walking between locations.
- Routes - Distance Matrix API: Calculate travel times for multiple destinations.

Describe any significant non-trivial algorithm or process if any (like rating, ranking, automatic prioritizing of items etc.)

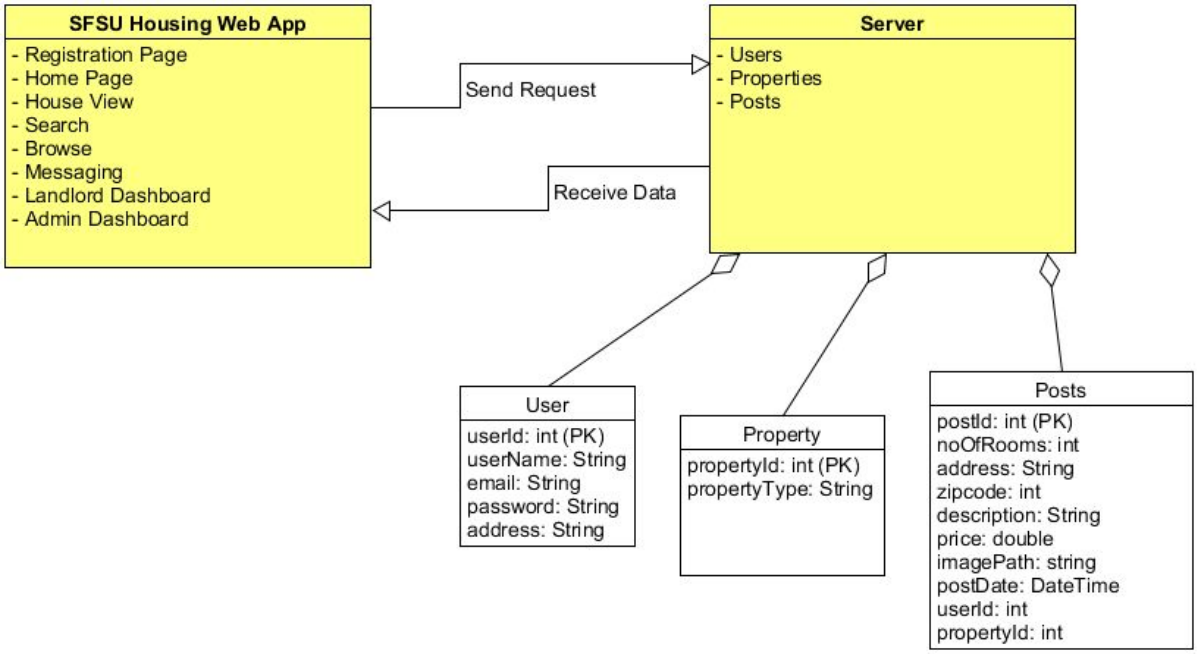
- No non-trivial algorithm for search and process expected at this particular time.

If you have changed SW tools and frameworks or added any new one please describe it. Any new SW or framework you will be using has to be approved by CTO in writing by this time.

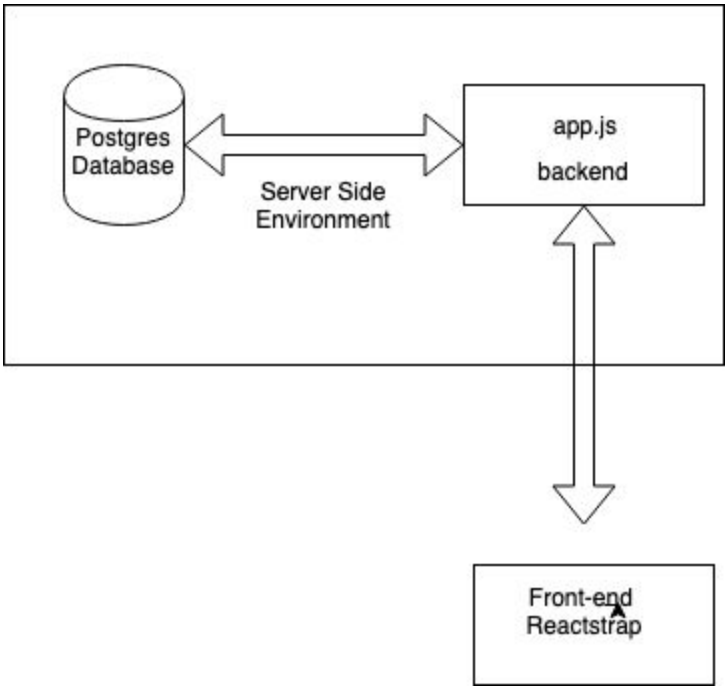
- We have changed the server service from AWS to Google App Engine.

# UML Diagrams

Class UML:




Deployment Diagram :



## Risk

Skills: Medium Risk. We are still learning many of the tool that we will be implementing. However, We are working together and sharing our knowledge of current tools to minimize our risk.

Schedule: No risk at the moment. We are currently working together inside and outside of class.

Technical: No risk at this time. We have switched servers from AWS to Google computing engine and have resolved technical issues.

Teamwork: No risk at this time. Teamwork is productive.

Legal: No risk at this time. We have not use any products that require legal or proper licensing/ copyright.

## Project management

Our team will be managed using trello and google docs. In M2, our team used google documents to split up and assign specific task to each team member. As well, each team member was informed of the deadline of their assigned task. After assigning each task, each group members accepted the milestone 2 document via google docs and started working towards completing their task. Once each group member completed their assigned task, they will inform the document master to check their work. Moving forward onto milestone 3 to milestone 5 we will be using trello to assign task.

# Milestone 3: Review Summary

Spring 2018

Section: 02 Team: 14 Date: 05/03/2019

1. **UI and functionality feedback (P1 functions only)**
    - Home Page
        - Search
            - Have the search bar at the top of the page
            - Have a filter next to the search bar to the left
            - Have a search button to the right of the search box
        - Nav Bar
            - Have Navigation bar under search bar
            - Have About page under search bar
            - Have Account management to the right of the search bar
        - Listings
            - Have two columns of listing after search
            - Create a contact button on the listing card
            - Create a new tab for each listing instead of a popup.
    - Dashboard
        - Message
            - Show listing, messages, user and date
            - Create a messaging system that allows users and homeowners to see their messages
            - Show list of messages or no messages
        - Posting
            - Show number of listings
            - Show current house posting of the homeowners
            - Show no postings if there are no posting on the dashboard
    - Admin Page
        - Drop admin page
        - Use PGAdmin instead
    - Database
        - Create a user database
        - Have about 20 house listing
    - Github
        - GitHub comments should be more descriptive

***2.*** **Feedback on coding style, github, database etc.**

- Git/Github organization (e.g. organization of branches)
    - GH organization looks fines, some branches have confusing names, not sure what scheme you have for branches.

- Git/Github usage: Comments on positing; Number of posting to github; Appr. even distribution of submissions among team members (check github post stats for all members)

Git messages look fine,

38 Hashim Jacobs

13 jfox

11 Davis

11 DavisTH127

11 Shalaka Aigal

10 jarrettlee

6 jfox16

5 Eric Chen

3 Jarrett Lee

3 csc648848Instr

2 Anthony

2 Davis Hoang

2 Kevin Mitsuda

2 kmitsuda9

1 Anthony J Souza

- Code documented (header, in code) with good coding style
    - Little to no comments, couldn't find any header comments.

- MVC/OO patterns followed up
    - MVC framework followed

- Frameworks (back end front end) deployed correctly
    - Express framework used and deployed.

- Database organization (tables, naming...)
    - Database looks fine, only contains two tables

- Blobs being used? If so, is it working?
    - Image paths are being used

- Adherence to best practices of security (PW encrypted; search inputs verified etc.)
    - Passwords are encrypted, could not find any input validation.

- Efficiency (proper use of image thumbnails, efficient search eytc.)
    - Could not find thumbnails,

- Other

# SW Engineering CSC648/848 Spring 2019

---

## GatorHouse


Team 14 (Local)

Davis Hoang ( Davis.hoang2015@gmail.com ) ( Team Lead / Github Master )

Front-End

Jonathan Fox ( Front-End Lead )
Kevin Mitsuda
Jarrett Lee

Back-End

Hashim Jacobs ( Back-End Lead )
Eric Chen
Shalaka Aigal


## Milestone 4
## 05/08/2019

| Revision Table | | |
|---|---|---|
| 05/07/2019 | 1.0 | Initial Document |
| 05/20/2019 | 2.0 | Revised Document |

## Product summary

Gatorhouse is a service that anyone can use for hassle-free house-hunting or house-selling. If you are a student, faculty member, or just someone searching for a new living situation near San Francisco State University, Gatorhouse is your new best friend.

4. **Simple Search**

   A simple search and a couple of clicks will get you that much closer to acquiring your ideal living space. With Gatorhouse, you're able to connect with multiple landlords with a high degree of simplicity in just minutes.

5. **Upload Property**

   If you're a seller, post your property with brief details, sit back, and watch the buyers/renters flood in. Gatorhouse is here to do the hard work so that everyone can live their best life.

6. **Admin control**

   Gatorhouse enforces reliable house listings. Our admin will be able to approve or reject uploaded house listing before they go live within 24 hours.

**You can reach us at:** http://ec2-52-53-199-247.us-west-1.compute.amazonaws.com

# Usability Test Plan

**Test objectives**:

Purpose: Verify that GatorHouse is usable to average users and that users would use GatorHouse for renting housing online.

Problem Statement: Is the main function of GatorHouse, (housing rental), easy for any average user to use?

**Usability Task description**:

Users will be able to locate and search for housing. Users will be able to view specifics of the housing such as number of rooms, cost and type(studio,apartment,or house).
If a user is interested in housing they do not need to register and can just use the contact information listed on the page to contact the landlord to rent the housing.

-User Profile: College student of the bay area, conversational level of English, ages 20 to 30. Tests will be presented to voluntary participants on the test monitor's personal computer

-Method and test design: Users will be asked to perform a simple search for a type of housing (studio,apartment,or house) that meets the criteria (lowest cost,number of rooms) and view the contact details of said housing.

-Test environment and equipment:

Hardware Setup- Website on Amazon Web Services running on a MAC Machine

Software Setup- GatorHouse default homepage on Chrome browser of a MAC Machine. 12 different types of housing are present in the postgres database.

-URL: http://ec2-52-53-199-247.us-west-1.compute.amazonaws.com
-Test monitor role: The test monitor will simply silently observe and time the amount of time it takes for the User to reach the details page of a house that meets the criteria of the user.

-Legal issues: Test is up to the participants and it totally voluntary. Users who test will not be asked their name or any other criteria. The only information we are keeping from the users will be their experience with the website itself. No personal information will be kept because Users will not be entering any information themselves, but if users do enter personal information accidentally, all of that information will be deleted.

-Report: The final report will contain information based on how quickly users found it to navigate through GatorHouse to find an apartment that meets their criteria.

**Questionnaire**:

Task 1: Locate the type of housing that is on 526 Font Blvd

| | |
|---|---|
| Task | Find the type of the housing on the address 526 Font Blvd |
| Machine State | Home page of GatorHouse:<br><br>http://ec2-52-53-199-247.us-west-1.compute.amazonaws.com |
| Success Criteria | Seeing the type of the house on 526 Font Blvd |
| Benchmark | 13 seconds |

Circle the statement that best describes how much you agree with each statement

-It was clear where to type the address on the page

| | | | | |
|---|---|---|---|---|
| Strongly Agree | Agree | Neutral | Disagree | Strongly Disagree |

Comments:

-The search was quick and displayed the correct result

| | | | | |
|---|---|---|---|---|
| Strongly Agree | Agree | Neutral | Disagree | Strongly Disagree |

Comments:

-After locating the house the type of housing at the address was clearly indicated

| | | | | |
|---|---|---|---|---|
| Strongly Agree | Agree | Neutral | Disagree | Strongly Disagree |

Comments:

Task 2: Locate the price of any house in “Daly City”

| | |
|---|---|
| Task | Find the price of a studio in Daly City |
| Machine State | Home page of GatorHouse: http://ec2-52-53-199-247.us-west-1.compute.amazonaws.com |
| Success Criteria | User can see the price of houses on St Francis Heights, Daly City |
| Benchmark | 30 seconds |

Circle the statement that best describes how much you agree with each statement

-It was clear where to type the address on the page

| | | | | |
|---|---|---|---|---|
| Strongly Agree | Agree | Neutral | Disagree | Strongly Disagree |

Comments:

-The search was quick

| | | | | |
|---|---|---|---|---|
| Strongly Agree | Agree | Neutral | Disagree | Strongly Disagree |

Comments:

-Displayed the correct result

| | | | | |
|---|---|---|---|---|
| Strongly Agree | Agree | Neutral | Disagree | Strongly Disagree |

Comments:

-The prices of the houses on Bark Madly Court are clearly displayed and obvious

| | | | | |
|---|---|---|---|---|
| Strongly Agree | Agree | Neutral | Disagree | Strongly Disagree |

Comments:

## QA test plan

Test objectives: QA testing is required in software in order to confirm that the software meets requirements.Further testing is then required to ensure that all features of priority one are completed and tested to meet specifications previously mentioned.

HW setup: Using Amazon Web Services, running on a Linux Machine
SW setup: Home page of GatorHouse on a Chrome browser of Windows 10.
URL: http://ec2-52-53-199-247.us-west-1.compute.amazonaws.com
Feature to be tested: Search function
Housing can be located by entering the housing’s address by name, price, housing number, and type.

**QA TEST PLAN**

| Number | Description | Test input | Expected output | PASS/FAIL |
|---|---|---|---|---|
| 1 | Test % like in search for name field | Type in “Townhouse” in the search bar | Check that results and see 1 listing is displayed | PASS |
| 2 | Test if search returns all listings when given no input | “ “ | Get all available listings | PASS |
| 3 | Test for searching by filtering type | Change dropdown value to “room” | Get all listings that are labeled as room | PASS |
| 4 | Test for searching by price | “3400” | Get all listings that are listed at $1,600 | PASS |
| 5 | Test for search by housing number | “526” | Get all listings that have a housing number of 777 | PASS |

## Code Review:

The code format that we use for Gatorhouse includes the following:

- Include header comments for all files
- Function usage comments are not necessary if name is adequately intuitive
- Two-space indentation for nested blocks
- Function and variable naming will be in camel case
- Functions and variables must be named with emphasis on purpose. For example, if the function searches for and returns rows in a table called “listings” that includes keywords from user input, it should be written in the following manner:

```
searchPropertiesByKeywords(searchInput){
  …
  res.send(properties);
}
```

- We use the ‘prettier’ javascript package, so formatting is done automatically to guarantee consistency.

## Self-check on best practices for security

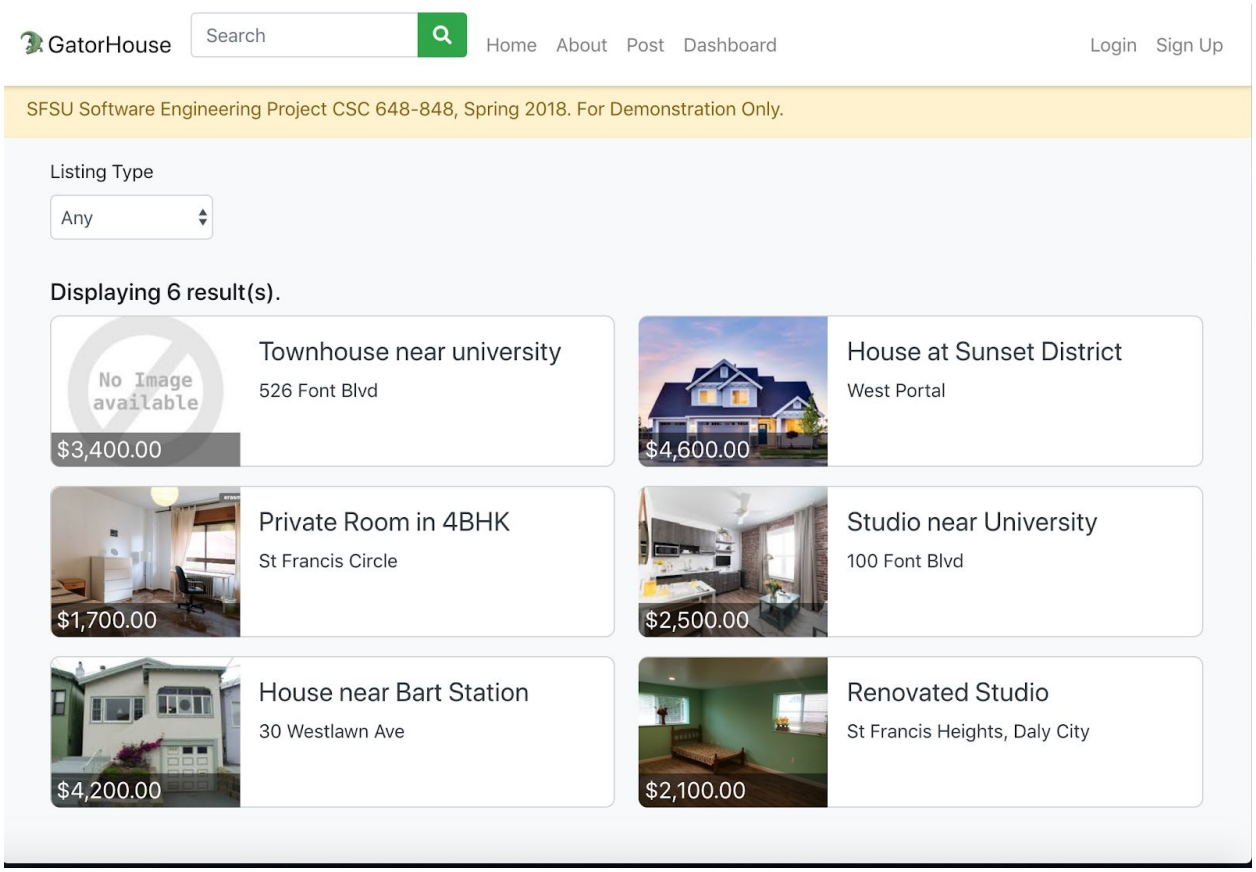
- Major assets being protected include email address and password.
- emails stored in secure DB
- Password in the database have been encrypted using the bcrypt hashing function
- Input validation is presented in login and registration.
- validate search text up to 40 alphanumeric characters
- Store images as files

## Self-check: Adherence to original Non-functional specs

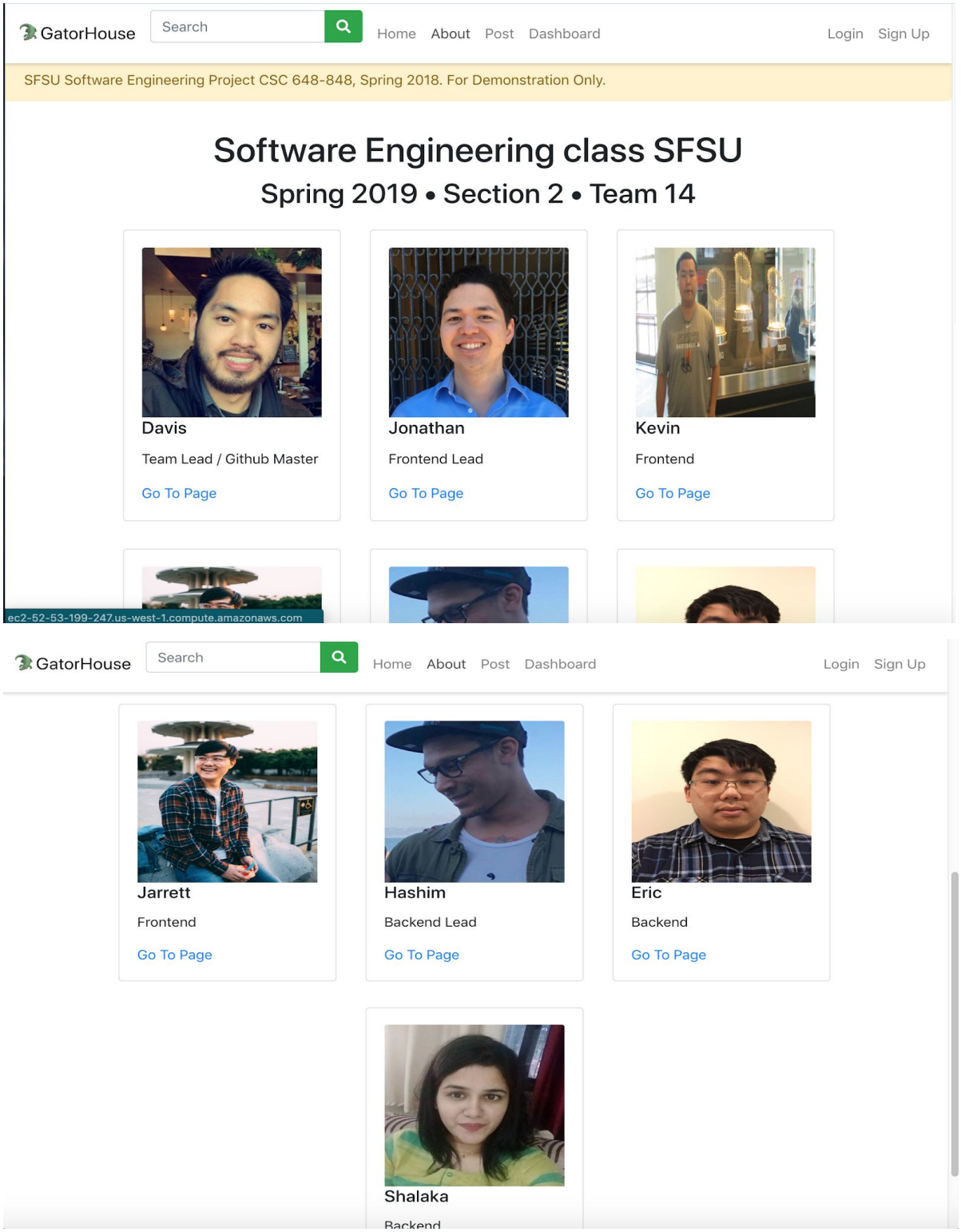
1. Application shall be developed, tested and deployed using tools and servers approved by Class CTO and as agreed in M0 (some may be provided in the class, some may be chosen by the student team but all tools and servers have to be approved by class CTO) (DONE)
2. Application shall be optimized for 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 ( ISSUE - Images are not properly aligned, Cannot click on Navbar dropdown menu)
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 ( Done)
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. (Done)
8. Application shall be very easy to use and intuitive. (Done)
9. Google analytics shall be added (ISSUE - URL contains a fragment)
10. No e-mail clients shall be allowed ( On Track)
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) (On Track)
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). (Done)

# Product Screen Shots:

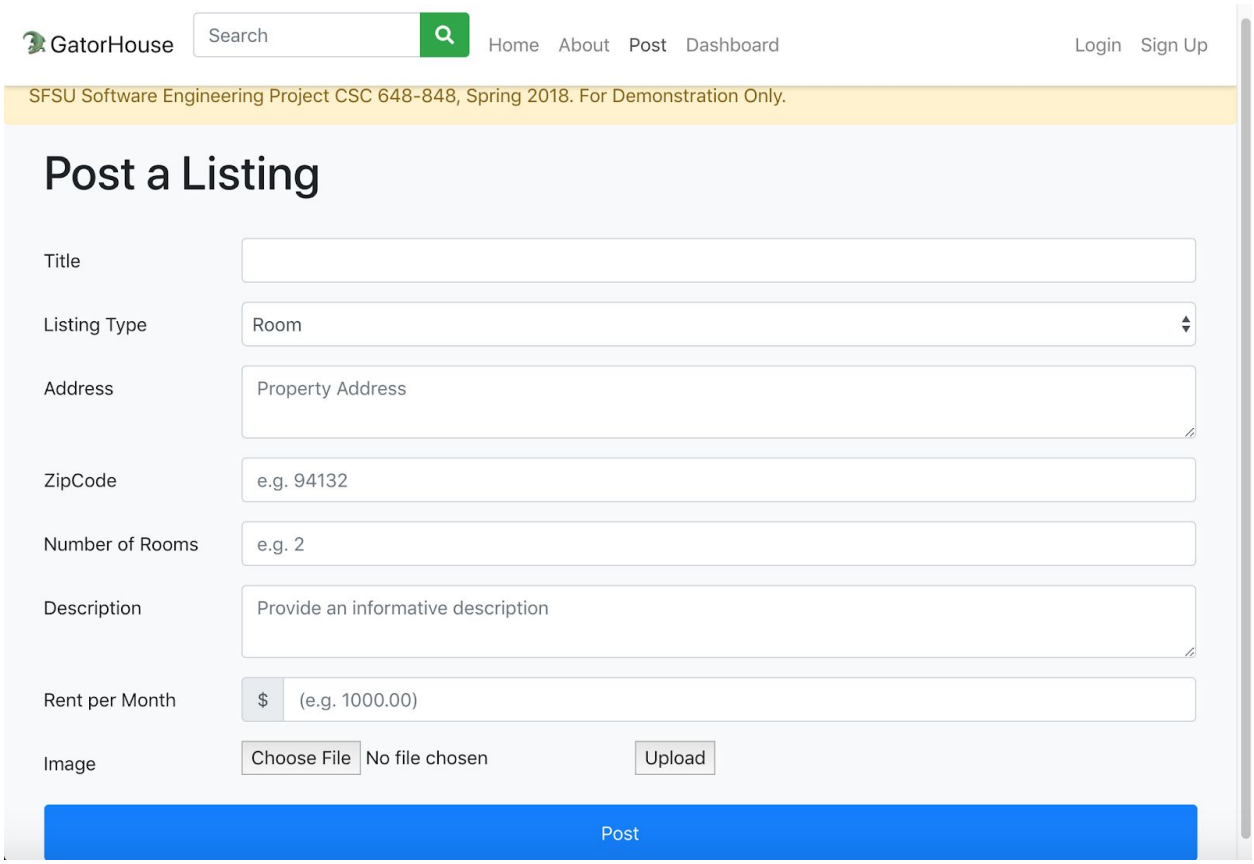
(Home Page)



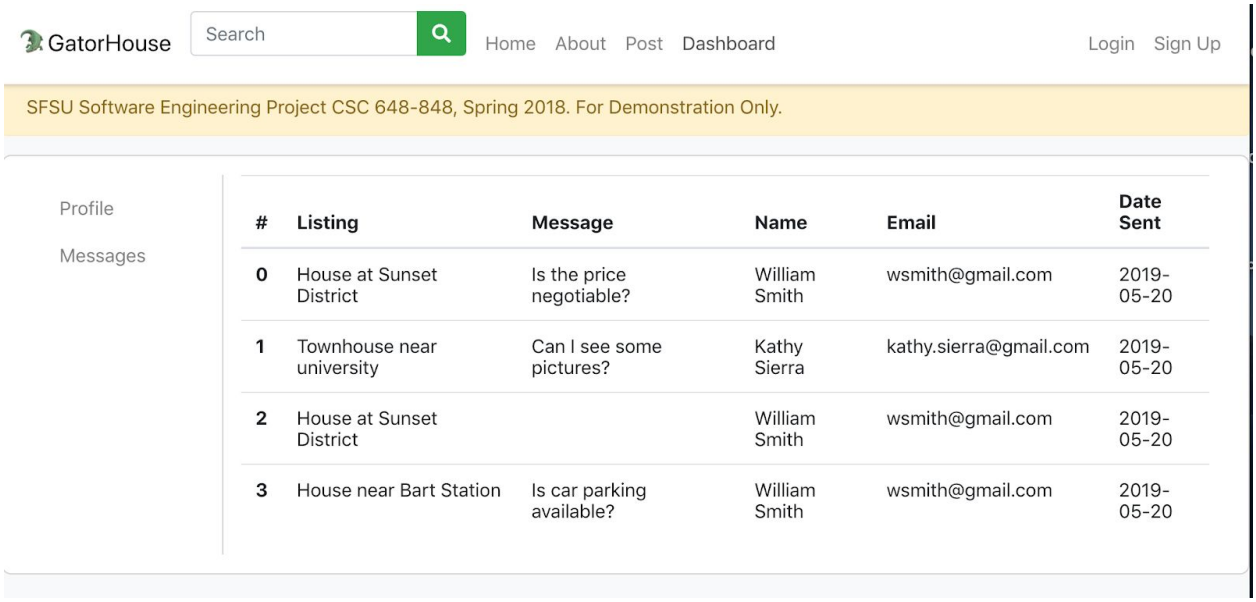
(About Page)


GatorHouse
Search
Home
About
Post
Dashboard
Login
Sign Up
SFSU Software Engineering Project CSC 648-848, Spring 2018. For Demonstration Only.
Software Engineering class SFSU
Spring 2019 • Section 2 • Team 14
Davis
Team Lead / Github Master
Go To Page
Jonathan
Frontend Lead
Go To Page
Kevin
Frontend
Go To Page
ec2-52-53-199-247.us-west-1.compute.amazonaws.com
GatorHouse
Search
Home
About
Post
Dashboard
Login
Sign Up
Jarrett
Frontend
Go To Page
Hashim
Backend Lead
Go To Page
Eric
Backend
Go To Page
Shalaka
Backend

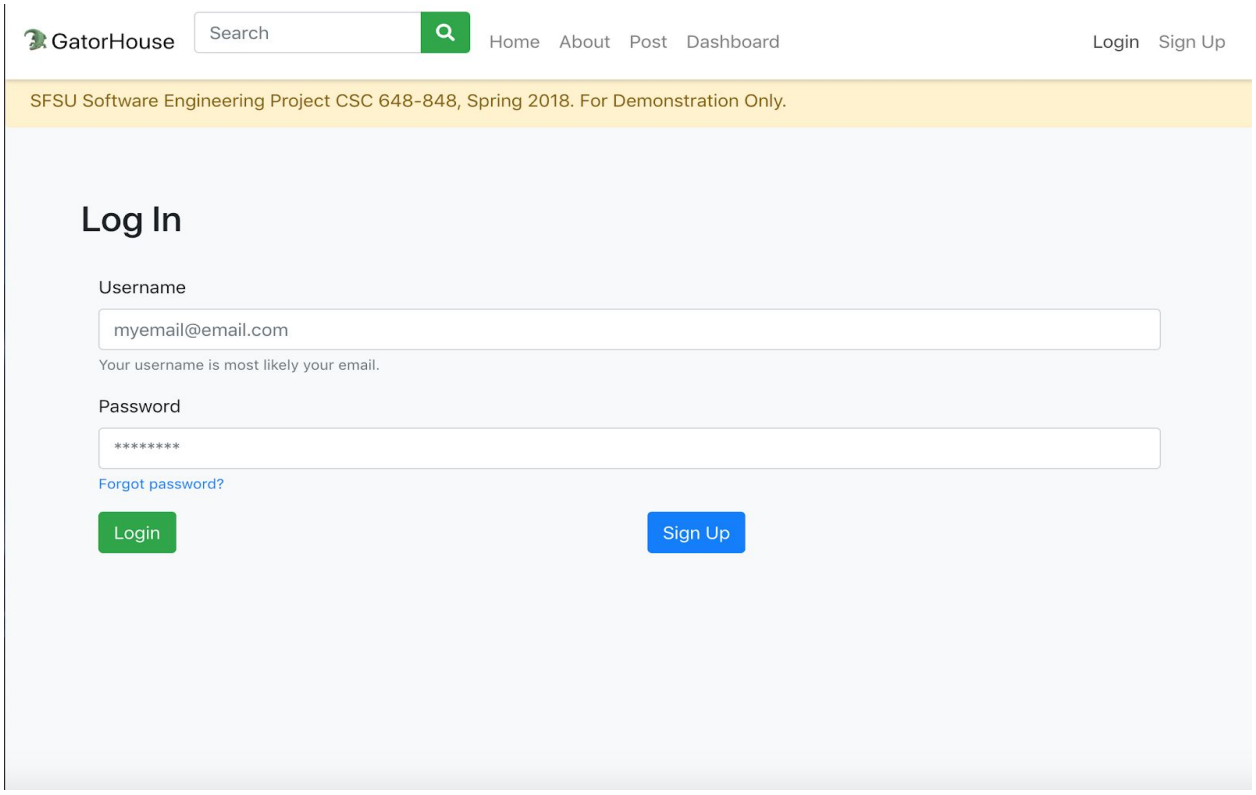
(Posting Page)



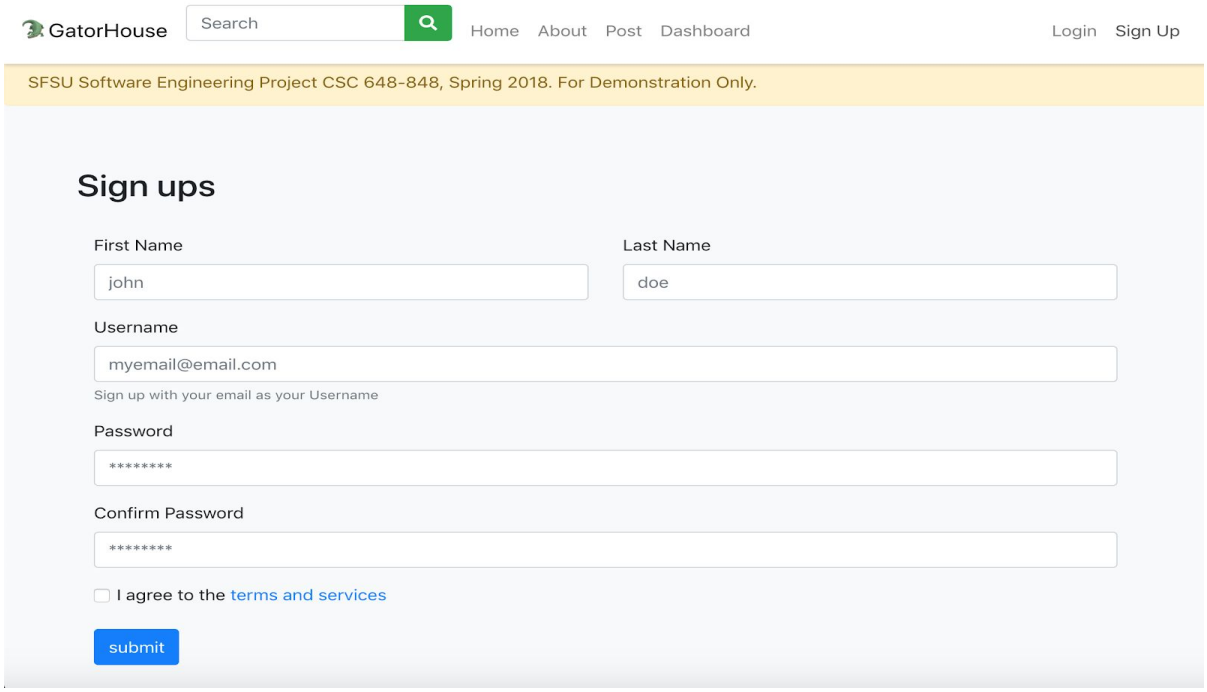
(Dashboard Page)



(Login Page)

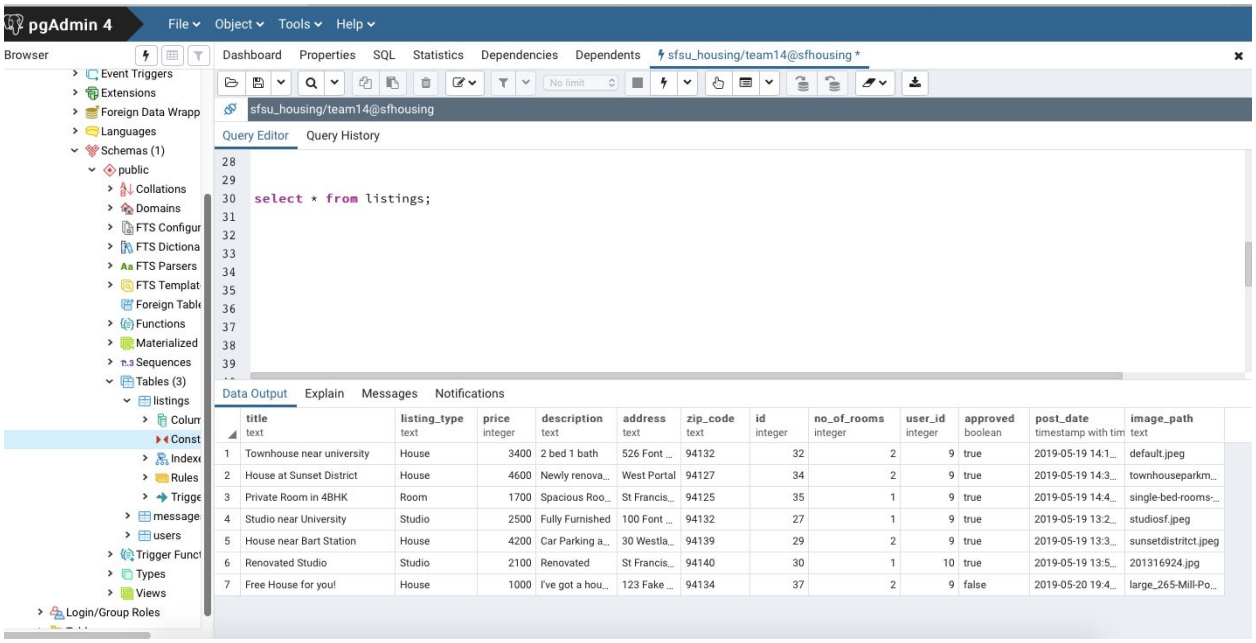

GatorHouse
Search
Home
About
Post
Dashboard
Login
Sign Up
SFSU Software Engineering Project CSC 648-848, Spring 2018. For Demonstration Only.
Log In
Username
myemail@email.com
Your username is most likely your email.
Password
********
Forgot password?
Login
Sign Up

(Signup Page)

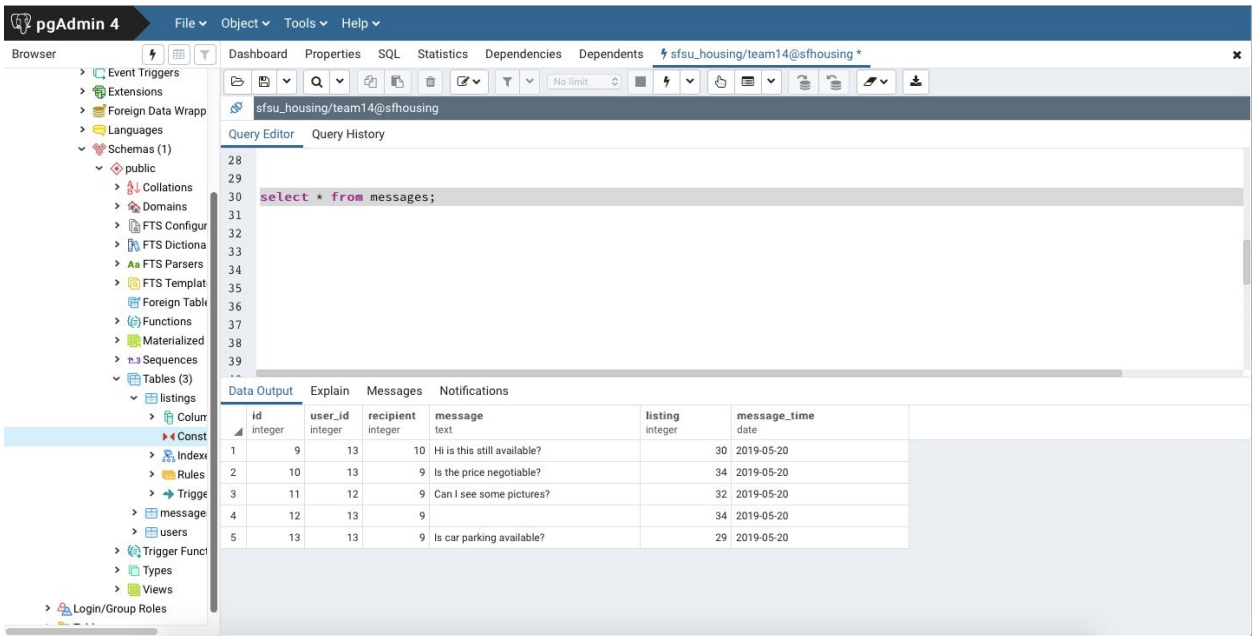

GatorHouse
Search
Home
About
Post
Dashboard
Login
Sign Up
SFSU Software Engineering Project CSC 648-848, Spring 2018. For Demonstration Only.
Sign ups
First Name
john
Last Name
doe
Username
myemail@email.com
Sign up with your email as your Username
Password
********
Confirm Password
********
I agree to the terms and services
submit

# Database Screen Shots:

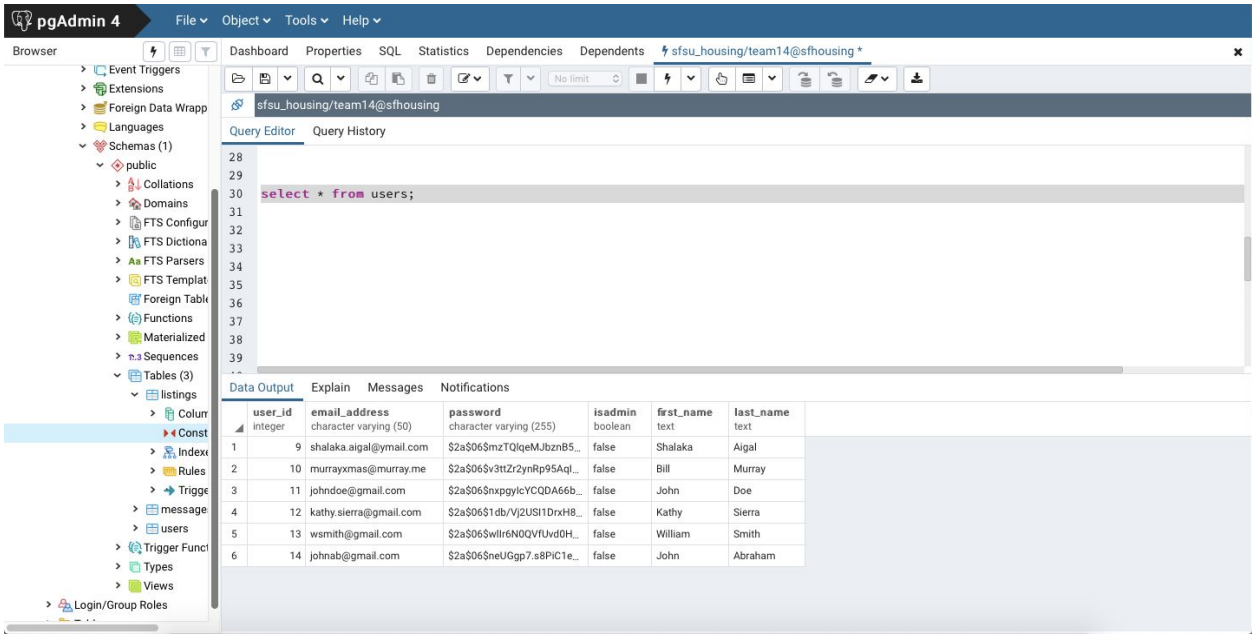
(Listing Display DB)



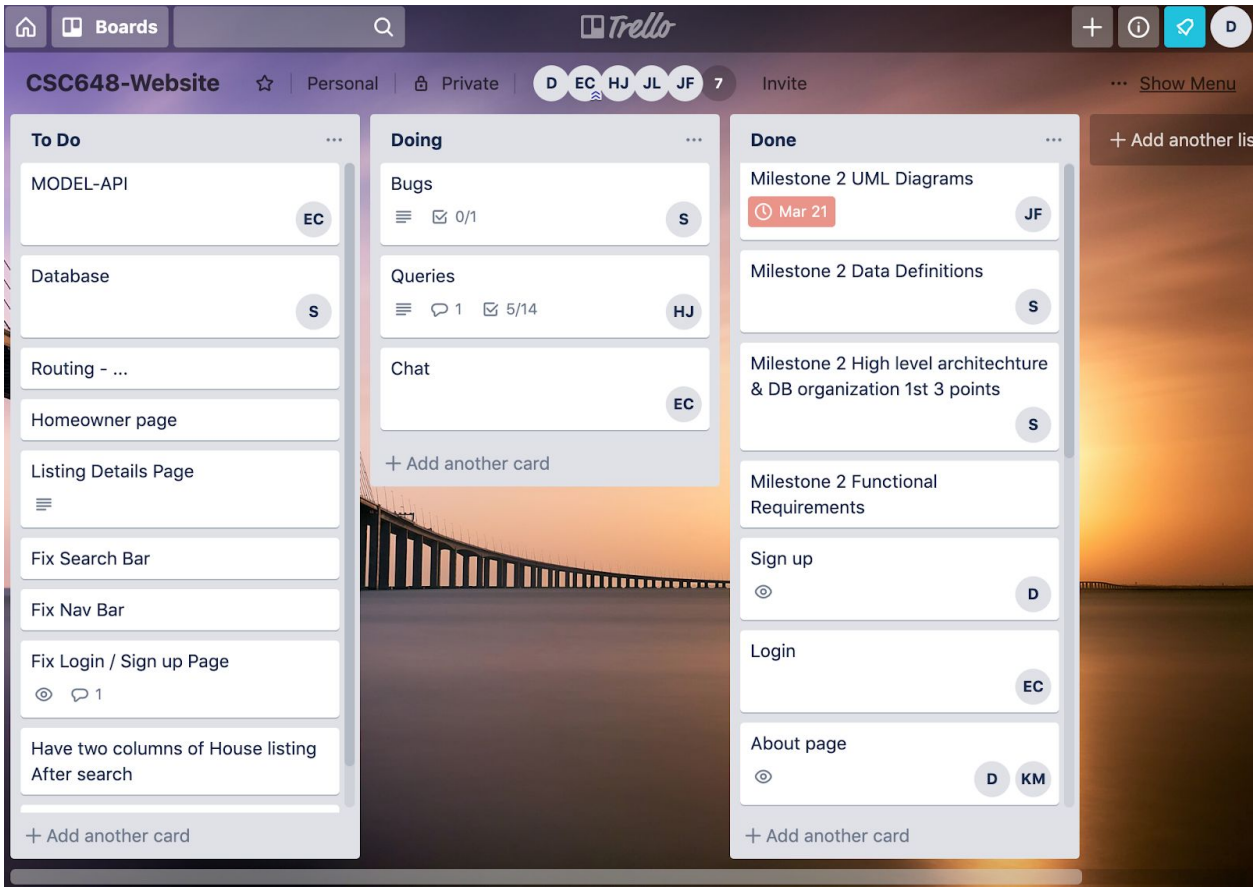

(Messages DB)



(Users DB)

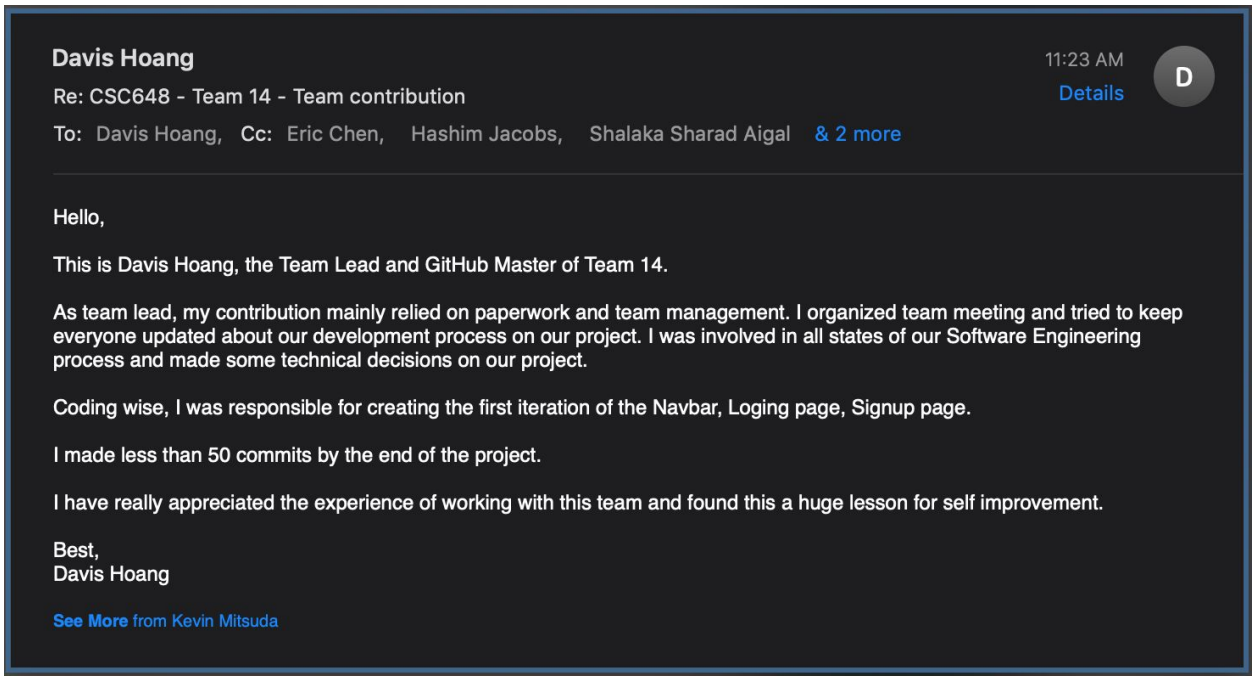

pgAdmin 4
File
Object
Tools
Help
Browser
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
sfsu_housing/team14@sfhousing *
sfsu_housing/team14@sfhousing
Query Editor
Query History
select * from users;
Data Output
Explain
Messages
Notifications
user_id integer
email_address character varying (50)
password character varying (255)
isadmin boolean
first_name text
last_name text
1 9 shalaka.aigal@ymail.com $2a$06$mzTQlqeMJbznB5... false Shalaka Aigal
2 10 murrayxmas@murray.me $2a$06$v3ttZr2ynRp95AqI... false Bill Murray
3 11 johndoe@gmail.com $2a$06$nxpgylcYCQDA66b... false John Doe
4 12 kathy.sierra@gmail.com $2a$06$1db/Vj2USI1DrxH8... false Kathy Sierra
5 13 wsmith@gmail.com $2a$06$wIlr6N0QVfUvd0H... false William Smith
6 14 johnab@gmail.com $2a$06$neUGgp7.s8PiC1e... false John Abraham

Task Management:



Team Member Contribution:

(Team Lead)



Davis Hoang

11:23 AM

Details

D

Re: CSC648 - Team 14 - Team contribution

To: Davis Hoang, Cc: Eric Chen, Hashim Jacobs, Shalaka Sharad Aigal & 2 more

Hello,

This is Davis Hoang, the Team Lead and GitHub Master of Team 14.

As team lead, my contribution mainly relied on paperwork and team management. I organized team meeting and tried to keep everyone updated about our development process on our project. I was involved in all states of our Software Engineering process and made some technical decisions on our project.

Coding wise, I was responsible for creating the first iteration of the Navbar, Loging page, Signup page.

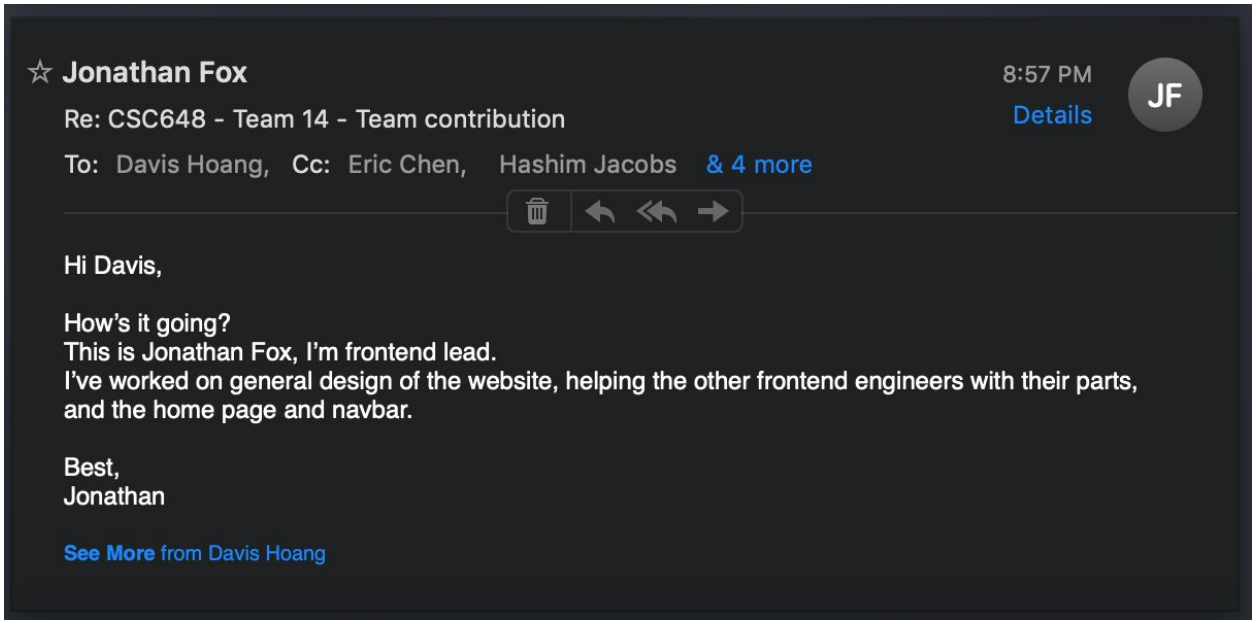
I made less than 50 commits by the end of the project.

I have really appreciated the experience of working with this team and found this a huge lesson for self improvement.

Best,
Davis Hoang

See More from Kevin Mitsuda

(Front End Lead)



☆ Jonathan Fox

8:57 PM

Details

JF

Re: CSC648 - Team 14 - Team contribution

To: Davis Hoang, Cc: Eric Chen, Hashim Jacobs & 4 more

Hi Davis,

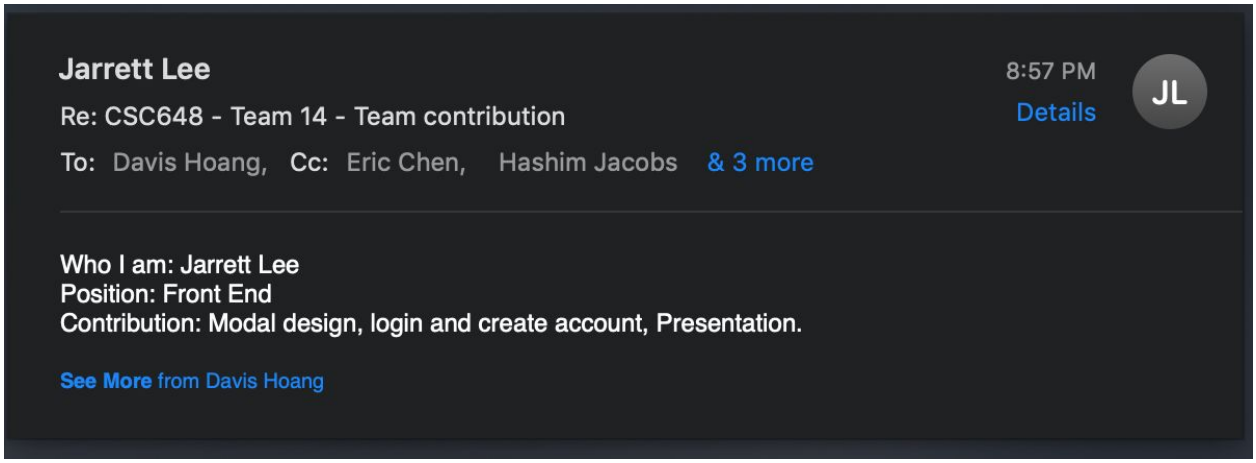
How's it going?
This is Jonathan Fox, I'm frontend lead.
I've worked on general design of the website, helping the other frontend engineers with their parts, and the home page and navbar.

Best,
Jonathan

See More from Davis Hoang

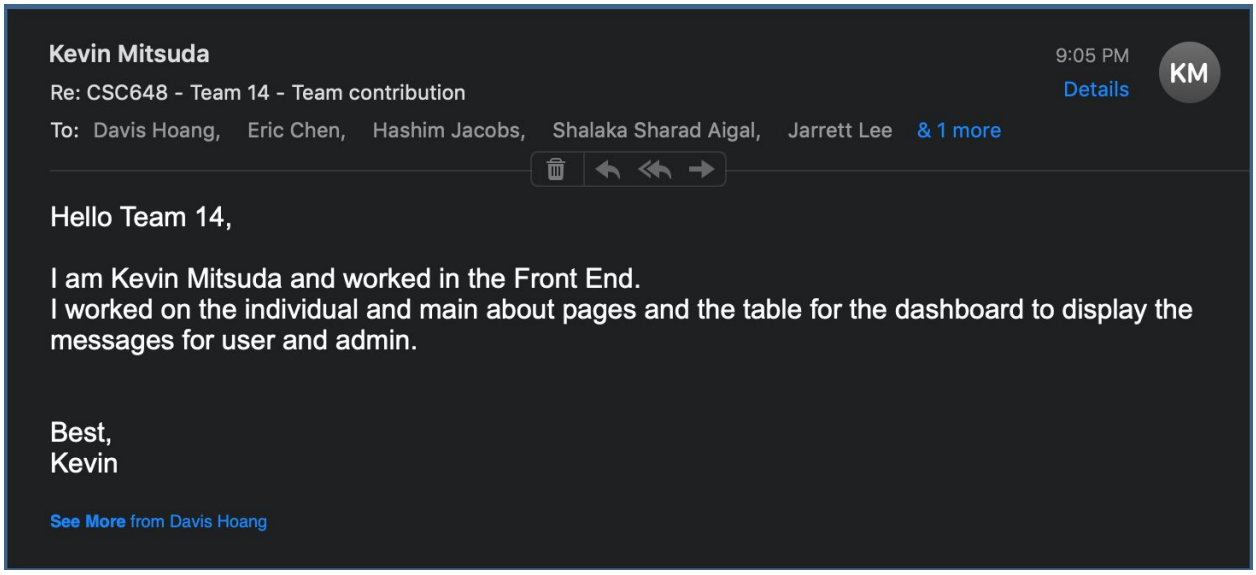
*Note Jonathan contributed less than 60 commits*

(Front End)




*Note Jarrett contributed less than 40 commits*

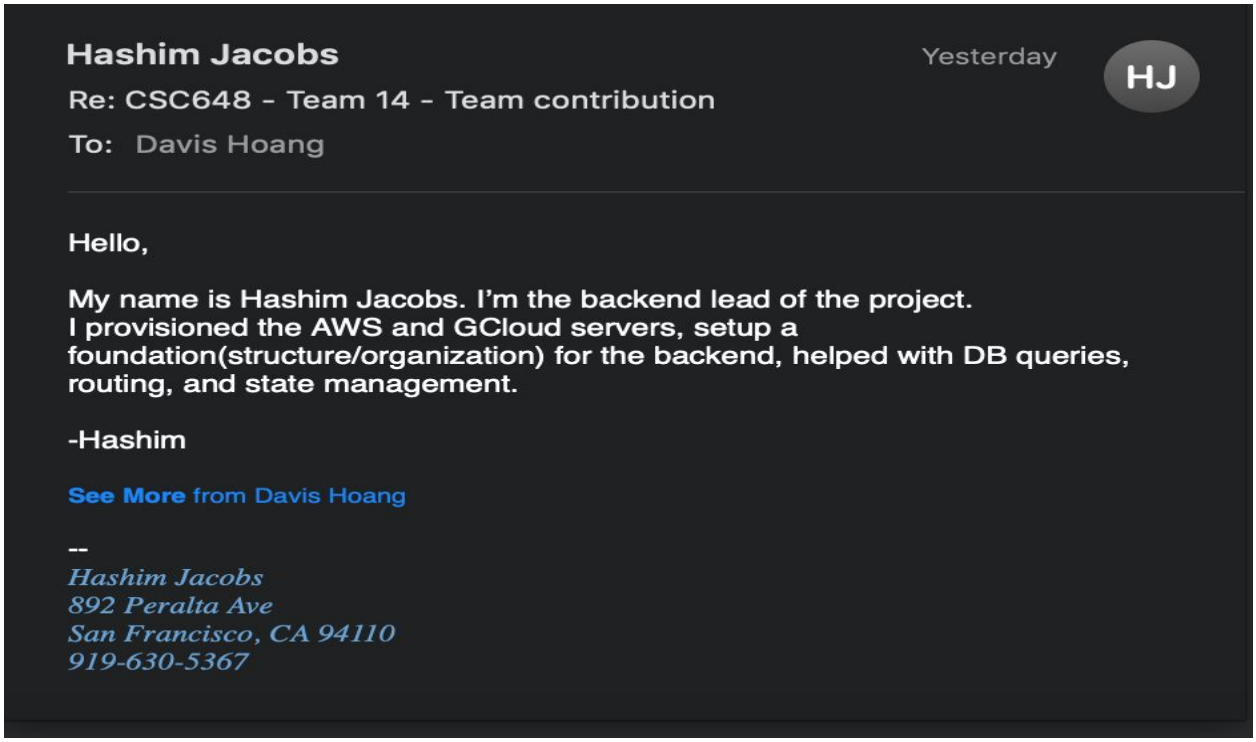
(Front End)




*Note Kevin contributed less than 10 commits*

(Back End Lead)

I



**Hashim Jacobs** Yesterday HJ

Re: CSC648 - Team 14 - Team contribution

To: Davis Hoang

Hello,

My name is Hashim Jacobs. I'm the backend lead of the project.
I provisioned the AWS and GCloud servers, setup a
foundation(structure/organization) for the backend, helped with DB queries,
routing, and state management.

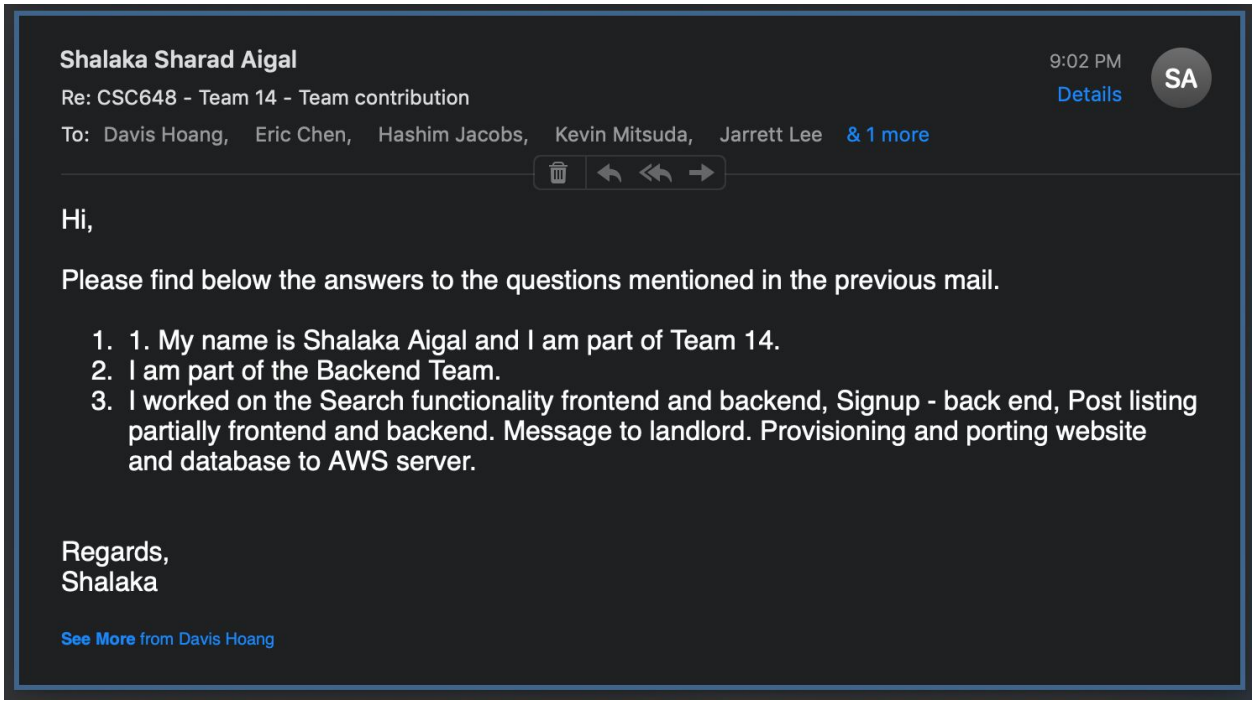
-Hashim

See More from Davis Hoang

--
*Hashim Jacobs*
*892 Peralta Ave*
*San Francisco, CA 94110*
*919-630-5367*

*Note Hashim contributed less than 70 commits*

(Back End)



**Shalaka Sharad Aigal** 9:02 PM Details SA

Re: CSC648 - Team 14 - Team contribution

To: Davis Hoang, Eric Chen, Hashim Jacobs, Kevin Mitsuda, Jarrett Lee & 1 more

Hi,

Please find below the answers to the questions mentioned in the previous mail.

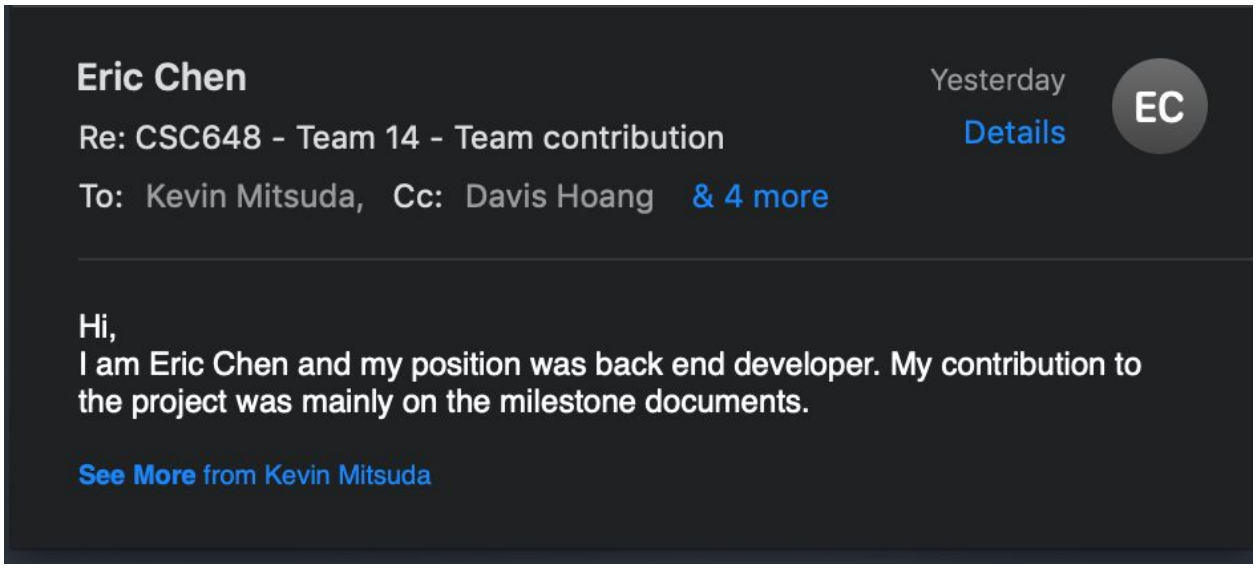
1. 1. My name is Shalaka Aigal and I am part of Team 14.
2. I am part of the Backend Team.
3. I worked on the Search functionality frontend and backend, Signup - back end, Post listing partially frontend and backend. Message to landlord. Provisioning and porting website and database to AWS server.

Regards,
Shalaka

See More from Davis Hoang

*Note Shalaka contributed less than 55 commits*

(Back End)




*Note Eric contributed less than 10 commits*

## Team Lead Post Analysis:

As we come to the end of the semester we reflect on the many lessons we learned while completing this project. After a long semester we are proud of the work we have put in to produce a website that was truly built from the ground up. Although you see the finalized website now, the semester brought many challenges and lessons that we will all be able to apply in the future.

The project definitely challenged us all in many ways. We collectively found one of the most challenging aspects to be getting everyone up to date with the current problems and issues. It was also difficult to learn the specific frameworks we chose to use. This is attributed to the fact that none of our teammates had exposure or experience with any of the frameworks. For example reactstrap, was a very new tool that none of us had worked with before. We were all learning our chosen as we worked through the project which is where most of our problems began to occur. Specifically for the backend team, they had a very hard time with getting the search bar to display the data from the database. This caused a delay in the progress of our website and ultimately was the largest road block in completing the website. This was a prevalent struggle that could have been avoided had we all been working on it together rather than just a couple of the teammates.

If I were to change how I lead the team I would communicate the issues we were having more effectively with all the teammates. I would often times take on the issues we were facing myself and try to fix it solely without communicating the problems to the entire team. I learned is not the best approach while developing a website in a group and next time I lead a team, I will apply this knowledge and yield a better outcome. Another challenge was communication between the front end and back end teams. As a team lead I personally found it difficult to constantly get my teammates up to date with our current issues. A lot of the times we were not all on the same page which lead to major problems with the development process and truly slowed down the work.

Though the semester brought us many challenges we ultimately all learned a big lesson in the development process of software engineering. I am very proud of team 14 and all the hard work they put in this semester to produce our final website.