Software Engineering CSC 648/848 Spring 2021

Gator Connection

A One Stop Website for SF State Gators

Team 05

Team Lead/Github Master: Alec Stephen Tenefrancia alectene@mail.sfsu.edu

Frontend Lead/Document Master: Lakshita Chugh

Backend Lead: Angelo Gloria Reyes

Frontend member: Bikram Tamang

Backend member: Benjamin Patrick Kao

Frontend member: Carmen Denisse Paisano

Backend member: Jiaxin Yu

Milestone 2

Milestone/Version	Date
M2V2	04/24/21
M2V1	04/01/21
M1V2	03/09/21
M1V1	03/04/21

Table of Contents

Data Definitions V2

---

1. **Unregistered User/ Guest**

These are the people who aren't registered with the school system yet. They will be treated as guests, and would not be eligible for any of our student/staff services. This is an administrative task.

2. **Registered User**

A person who is registered with us using their email would have an account with us. He/She will follow the Create Account path. His/Her data is already registered with us, he/she will have the authority to create their password here.

3. **Account**

An account contains identification data about the registered user that created the account. The data identifies what type of registered user the user is, as well as helps authentication registered users for restricted features such as posting announcements.

a) **Student:** A student account gives registered users the ability to search housing listings, post restaurant reviews, and make purchase requests for posted items and housing listings.
b) **Admin:** An admin account gives registered users the power to post public announcements. Admin accounts are split into more specific roles as to more easily group announcements together when filtering/searching.
    i) **Sports/Athletics:** A registered user who has this type of account is in charge of a specific sport that he/she registered with.
    ii) **Organization/Club:** A registered user who has this type of account is in charge of a specific organization that he/she registered with.
    iii) **School Department:** A registered user who has this type of account is a part of SFSU staff.
c) **Super User:** A super user account gives registered users the power to approve restaurant additions, administrative account creation requests, and registered user account upgrade requests.

4. Notifications

Receiving notifications about housing, submissions, etc.

a) **Housing Notifications:** Registered users will receive email notifications about housing such as if they receive a request for a listing.
b) **Submission Notifications:** Registered users will receive email notifications about submissions such as if their submission has been approved or denied

5. Rate

This gives users the ability to rate various student organizations, restaurants. It'll prove to be a useful tool for fellow users.

a) **Student Organization Rating:** Ratings that users can give organizations from 1-10. The average of all ratings will be shown on the organization.
b) **Restaurants Rating:** Ratings that users can give restaurants from 1-10. The average of all ratings will be shown on the restaurants.

6. Review

Posts shown under restaurants made by registered users and above. Unregistered users can view these but they cannot rate or post a review.

7. Email

Being able to send emails to all of the users that are registered. This would be students and administration.

a) **Housing emails:** A User who posted a listing of a house for sale will receive emails from people who are interested to purchase the listing.
b) **Registration Verification email:** A User who creates an account will receive an email to their SFSU email saying that their account has been created
c) **Reset password email:** An email which a user can request to reset their password

8. Announcements

Section designated to announcements that different departments make, along with the health centers, athletics, and organizations/clubs. These announcements are directly related to SFSU campus and only administrative users can post announcements.

a) **Athletic Announcements:** Announcements made from athletic directors
b) **Organization Announcements:** Announcements made from organizations

c) **School Announcements:** Announcements made from the school

9. Restaurants

Section dedicated to restaurants around campus. Users can view, rate, and write reviews for that specific restaurant. Unregistered users can view these but they cannot rate or review.

10. Ecommerce/Listing

The main section where users can view the selected items for sale or housing available and make purchase requests for them. Registered users can post things for sale. Unregistered guests cannot view these items.

a) **Items:**
   The item(s) listed for sale in the Ecommerce/Listing by a registered seller or for purchase by a registered buyer. Users can message the seller if they are interested in the item.

b) **Housing:**
   The listing(s) of housing listed for sale in the Housing section. Users interested in a listing can fill out a form which includes their name, school year, SFSU email, and a little information about themselves. After filling out the form, the system will send out an email to the person who posted the listing, and it will be up to them if they wish to respond back.

Functional Requirements V2

Priority 1 Requirements:

1. Guest User
    - 1.1. A guest user shall be able to search for restaurants by title.
    - 1.2. A guest user shall be able to sort restaurant listings by rating.
    - 1.3. A guest user shall be able to navigate through announcements.
    - 1.4. A guest user shall be able to create an account using his/her unique SFSU email account for a student account or a unique email account for an admin or super user account.
    - 1.5. A guest user shall be able to navigate to a map of SFSU.
    - 1.6. A guest user shall be able to search for items on sale by preferred payment.
    - 1.7. A guest user shall be able to search for items on sale by price.
    - 1.8. A guest user shall be able to search for items on sale by title.
2. Registered User
    - 2.1. A registered user shall have all of the same permissions as guest users.
    - 2.2. A registered user shall be able to log in to his/her account using his/her unique SFSU email.
    - 2.3. A registered user shall be able to log in to his/her account using many devices.
    - 2.4. A registered user shall be able to log out of the website.
    - 2.5. A registered user shall be able to stay logged in if they have not logged out.
    - 2.6. A registered user shall be able to post reviews of restaurants/food.
    - 2.7. A registered user shall be able to make purchase requests for items.
    - 2.8. A registered user shall be able to search for items by preferred payment, price, etc.
    - 2.9. A registered user shall be able to see and search housing listings.
    - 2.10. A registered user shall be able to make a request for a housing listing.
    - 2.11. A registered user shall be able to edit a housing listing title he/she posted.
    - 2.12. A registered user shall be able to edit a housing listing description he/she posted.
    - 2.13. A registered user shall be able to edit a housing listing price he/she posted.
    - 2.14. A registered user shall be able to change a housing listing image he/she posted.
    - 2.15. A registered user shall be able to close a housing listing that he/she posted.
    - 2.16. A registered user shall be able to post many pending items for sale with pictures.
    - 2.17. A registered user shall be able to edit a post about an item for sale that he/she posted.
    - 2.18. A registered user shall be able to take down an item that he/she put up for sale.

2.19. A registered user shall be able to create a restaurant request to add a restaurant to the restaurants listing by providing the name of the restaurant and location.
2.20. A registered user shall be able to sort and search for restaurants by ratings/reviews.
2.21. A registered user shall be notified if their restaurant location has been approved or denied, with a message explaining why.
2.22. A registered user shall receive a notification about a house listing being closed if he/she had made a request to the house lister.
2.23. An upgraded registered user shall receive a notification when their administration privileges are taken away and why.
2.24. A registered user shall have a unique registered id.
2.25. A registered user shall be able to fill out a form for a housing request.
2.26. A registered user shall receive a notification if their housing post has been approved by email.
2.27. A registered user shall receive a notification if their housing post has been denied by email.
2.28. A registered user shall receive a notification if their post has been approved by email.
2.29. A registered user shall receive a notification if their post has been denied by email.
2.30. A registered user shall be able to request to upgrade his/her account to an administrative account with a valid organization, role, and organization by email.
2.31. A registered user shall receive a notification about purchase requests made on his/her account by email.
2.32. A registered user shall receive a notification about purchase requests made on items he/she posted for sale by email.
2.33. A registered user shall receive a notification about a house listing request for a house listing that he/she posted that includes information about the person who requested, such as his/her name and email address.
2.34. A registered user who requested an account upgrade shall receive a notification when the request has been approved and the date of when their administration privileges will be taken away by email.
2.35. A registered user who requested an account upgrade shall receive a notification when the request has been denied by email.

3. Admin User
   3.1. An administrative user shall have all of the same permissions as registered users.
   3.2. An administrative user shall be able to post announcements.
   3.3. An administrative user shall be able to remove an announcement he/she had posted.
   3.4. An administrative user shall have a unique admin id.

4. Super User
    4.1. A super user shall be able to log into the website.
    4.2. A super user shall be able to log out of the website.
    4.3. A super user shall have all of the same permissions as administrative users.
    4.4. A super user shall be able to approve/deny and close account upgrade requests,
    4.5. A super user shall be able to approve/deny and close administrative account creation requests.
5. Sale Item
    5.1. A sale item shall be posted by a registered user.
    5.2. A sale item shall have a unique item id.
    5.3. A sale item shall have a title for the item.
    5.4. A sale item shall have a message describing the item.
    5.5. A sale item shall have one picture attached to it.
    5.6. A sale item shall be purchased by a registered user.
    5.7. A sale item shall be taken down by the registered user that posted it for sale.
    5.8. A sale item shall have a price.
    5.9. When a sale item is taken down, all registered users who made a request for the item shall be notified of the closure by email.
6. Housing Listing
    6.1. A housing listing shall be posted by a registered user.
    6.2. A housing listing shall have a unique housing id.
    6.3. A housing listing shall have a title.
    6.4. A housing listing shall have a message describing the item.
    6.5. A housing listing shall have a price.
    6.6. A housing listing shall have at least 1 picture attached to it.
    6.7. A housing listing shall be taken down by the registered user that posted it.
    6.8. A housing listing shall be requested by many registered users.
    6.9. A housing listing shall have a form to fill out for interested users.
    6.10. A housing listing shall have a situation(available/close).
    6.11. When a housing listing is taken down, all registered users who made a request shall be notified of the closure by email.
7. Housing Listing Form
    7.1. A housing listing form shall be filled out by a registered user.
    7.2. A housing listing form shall fill out their full name.
    7.3. A housing listing form shall fill out their school year.
    7.4. A housing listing form shall fill out their school email.
    7.5. A housing listing form shall fill out their phone number.
    7.6. A housing listing form shall have a section dedicated to “an about” me.
    7.7. A housing listing form shall have an expected day they want to move in.

8. Restaurant
    8.1. A restaurant shall have a name.
    8.2. A restaurant shall have a location.
    8.3. A restaurant shall have a unique restaurant id.
    8.4. A restaurant shall have a rating.
9. Restaurant Request
    9.1. A restaurant request shall be created by one and only one registered user and above.
    9.2. A restaurant request shall have the name of the restaurant.
    9.3. A restaurant request shall have the location of the restaurant.
    9.4. A restaurant request shall be confirmed/denied and closed by one and only one super user.
    9.5. When a restaurant request is confirmed/denied the registered user who made the request will be notified by email.
10. Organization
    10.1. An organization shall be able to post announcements.
11. Announcement
    11.1. An announcement shall be posted by one and only one administrative user or organization.
    11.2. An announcement shall be able to be viewed by one or many users.
    11.3. An announcement shall be able to be taken down by the one who posted it.

Priority 2 Requirements:

12. Guest User
    12.1. A guest user shall be able to make item purchase requests after filling out an identification form and completing a Captcha.
13. Registered User
    13.1. A registered user shall be able to rate organizations.
    13.2. A registered user shall be able to change their rating of the organization.
    13.3. A registered user shall be able to post reviews under event announcements.
    13.4. A registered user shall be able change any reviews they have under event announcements.
    13.5. A registered user shall be able to rate a housing listing.
    13.6. A registered user shall be able to rate any of the item listings on sale.
    13.7. A registered user shall be able to change their rating of the item.
    13.8. A registered user shall be able to report housing listings and provide reasoning for the report.
    13.9. A registered user shall be able to report restaurant reviews.
    13.10. A registered user shall be able to report announcements

14. Admin User
    14.1. An admin user shall be able to temporarily take down posts that he/she finds false and give a reason why.
15. Super User
    15.1. A super user shall receive notifications of posts that admin users take down.
16. Sale Item
    16.1. A sale item shall be able to be taken down by a super user.
17. Housing Listing
    17.1. A housing listing shall be able to be taken down by a super user.
18. Housing Listing Form
    18.1. A housing listing form shall be able to be saved by the creator in their notifications page.
19. Restaurant
    19.1. A restaurant shall be able to have their rating changed.
    19.2. A restaurant shall have many reviews written by registered users.
20. Restaurant Request
    20.1. A restaurant request shall also allow users to request to update incorrect information about a restaurant.
21. Organization
    21.1. An organization shall be able to be rated by many registered users.
22. Announcement
    22.1. An announcement shall be able to be taken down by a super user.
    22.2. An announcement shall be able to be rated.

Priority 3 Requirements:

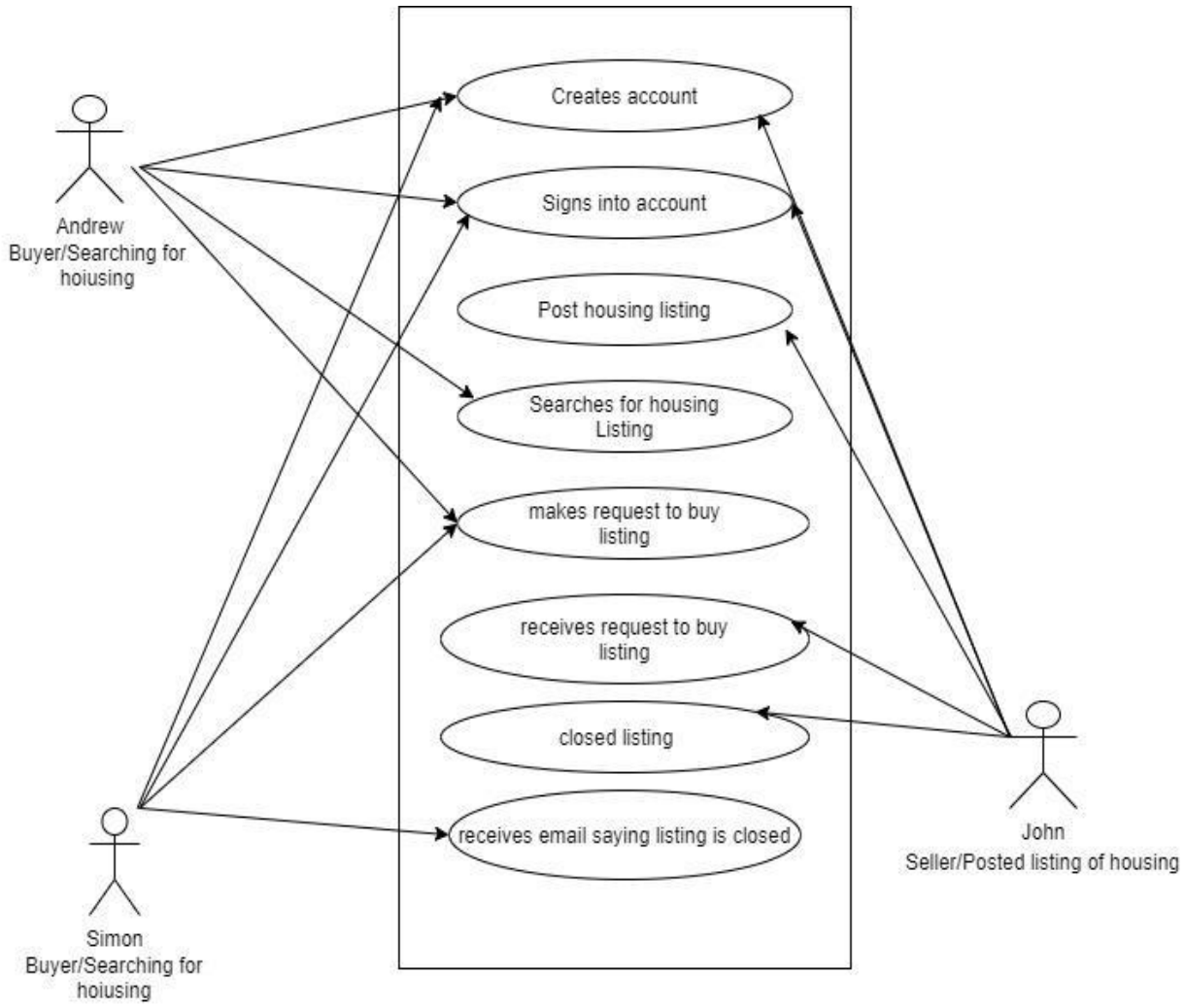
23. Guest User
    23.1. A guest user shall be able to make housing purchase requests after filling out an identification form and completing a Captcha.
24. Registered User
    24.1. A registered user shall be able to fill out a potential roommate form when posting a housing listing.
    24.2. A registered user shall be able to fill out a potential roommate form when searching through housing listing posts.
    24.3. A registered user shall be able to receive suggested housing listings where both the poster and searcher are ideal roommates.
    24.4. A registered user shall be able to subscribe to organizations and departments.
    24.5. A registered user shall be able to report item and housing listings for false/fake information.

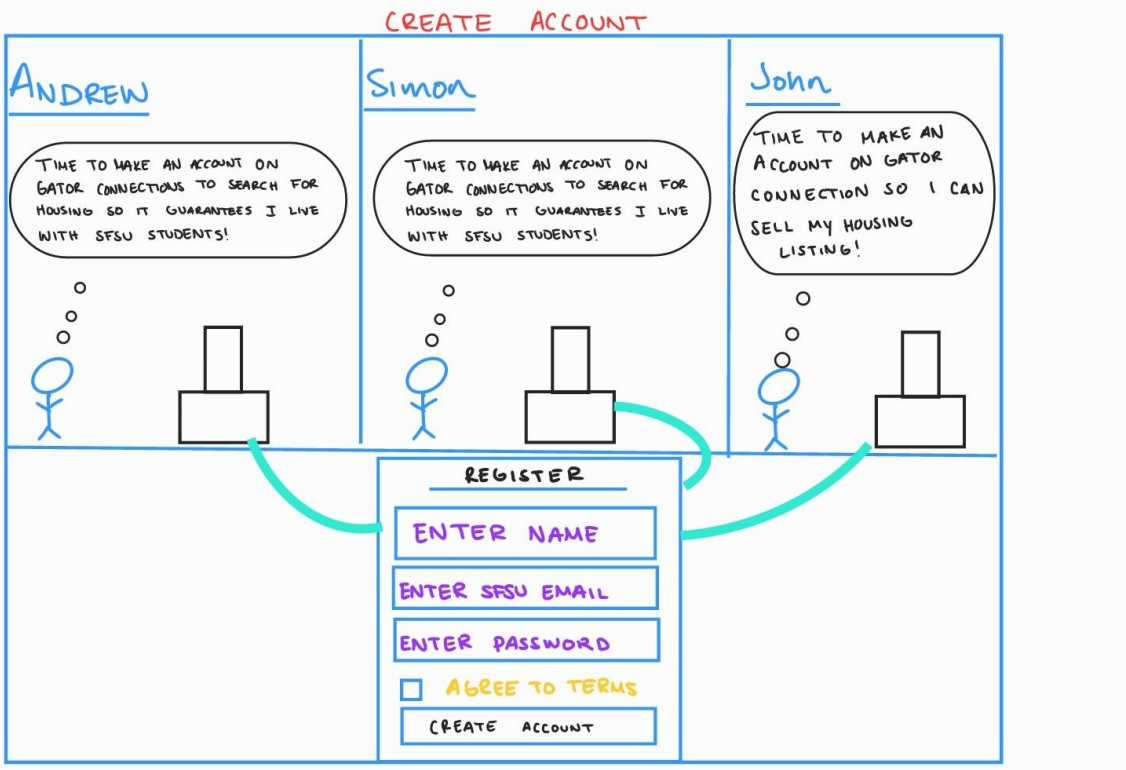
24.6. A registered user shall be able to be banned/restricted to only guest user privileges if he/she has a certain number of reports on posts.

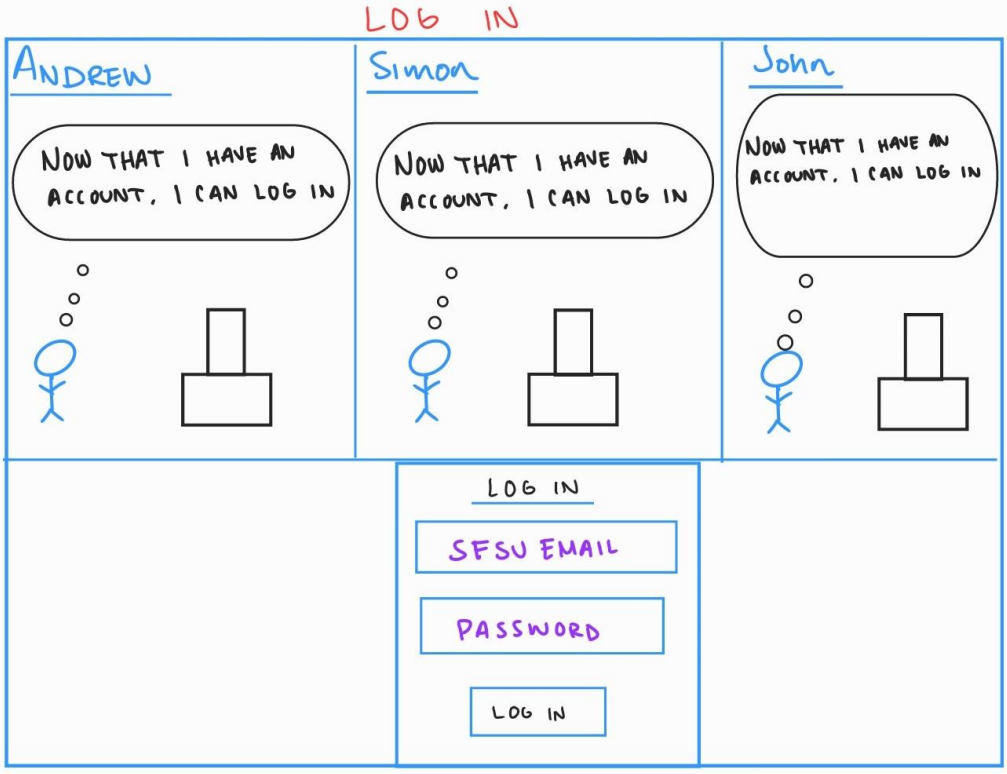
25. Admin User
    25.1. An admin user shall be able to be banned/restricted to only guest user privileges if he/she has a certain number of reports on announcements.
26. Super User
    26.1. A super user shall receive notifications of taken down posts such as announcements, housing listings, and items.
    26.2. A super user shall receive notifications of banned users.
    26.3. A super user shall be able to unban banned users.
27. Sale Item
    27.1. A sale item shall be automatically hidden from users when a certain number of reports is reached.
28. Housing Listing
    28.1. A housing listing shall be automatically hidden from users when a certain number of reports is reached.
    28.2. A housing listing shall be shown on a virtual map.
    28.3. A housing listing shall show the distance from SF State.
    28.4. A housing listing shall show how long it takes for a person to get to SF State.
29. Housing Listing Form
    29.1. A housing listing form shall be able to be customized for a housing listing post by registered users.
30. Restaurant
    30.1. A restaurant shall be shown on a virtual map.
    30.2. A restaurant shall show the distance from SF State.
31. Restaurant Request
    31.1. A restaurant request shall be able to be deleted by the one requested it.
32. Organization
    32.1. An organization shall be able to send emails to subscribed registered users about announcements.
33. Announcement
    33.1. An announcement shall be able to be mass sent to registered users.
    33.2. An announcement shall be automatically hidden from users when a certain number of reports is reached.

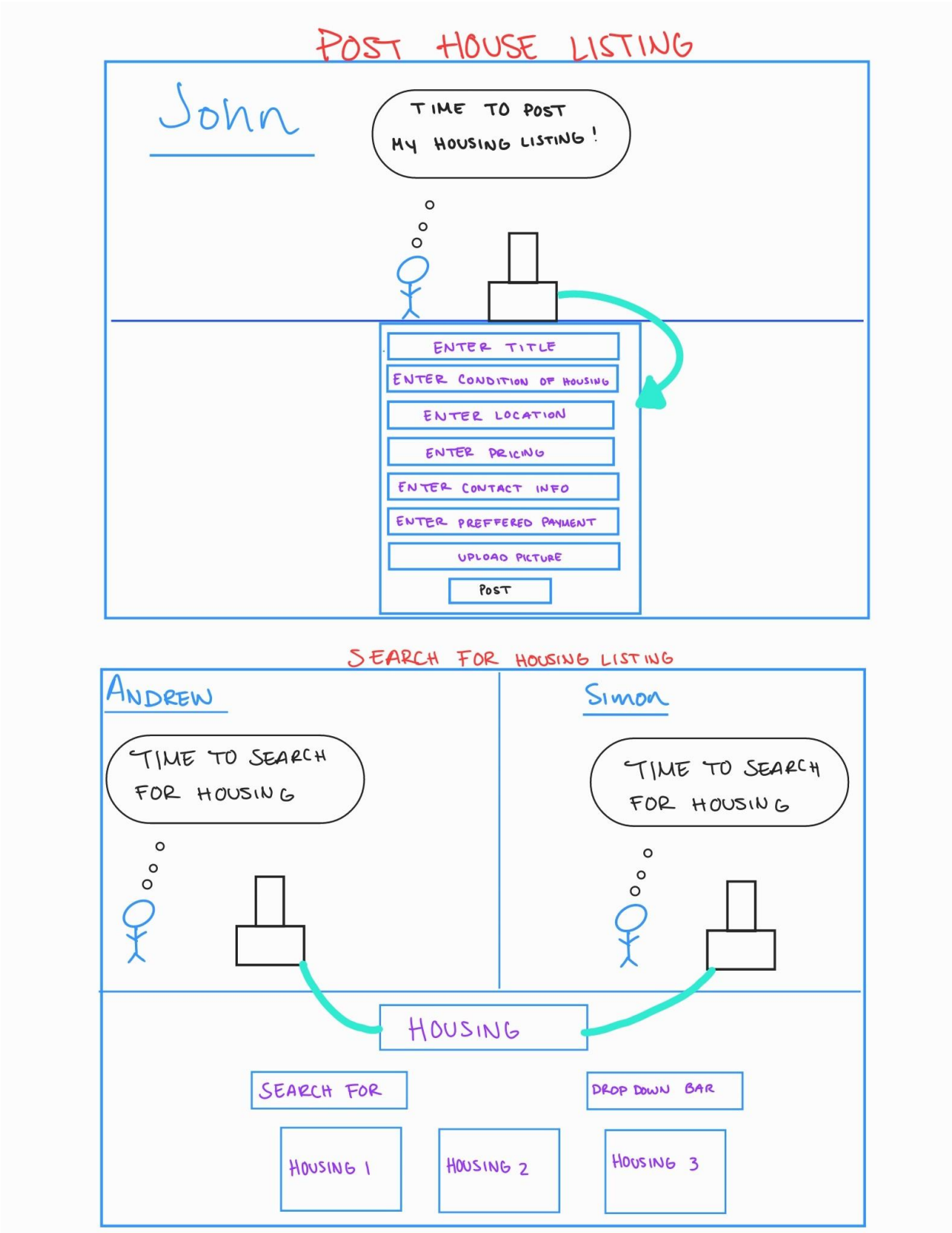
UI Mockups and Storyboards

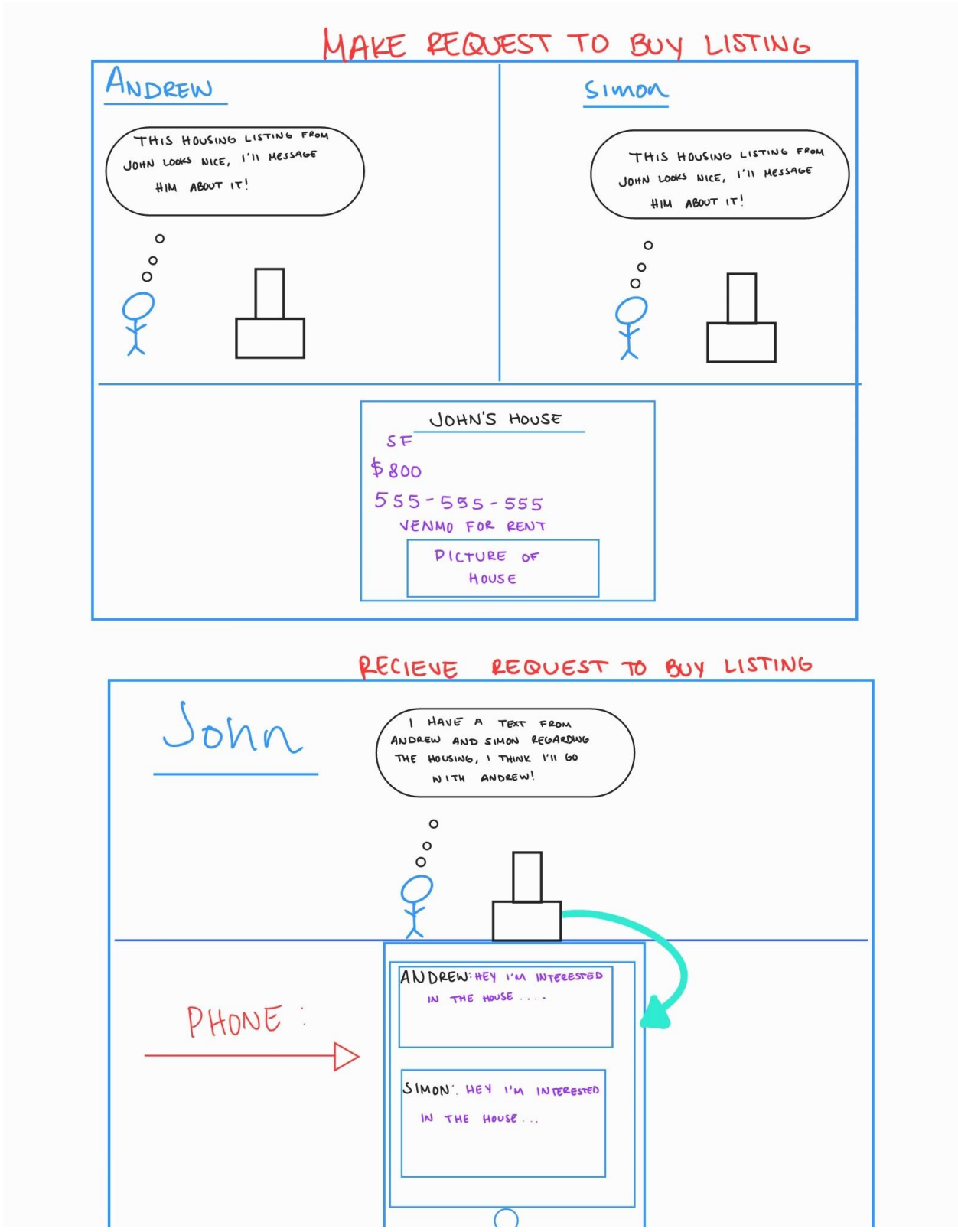
Case 1: Freshman Student looking for housing at SFSU
Actors: Andrew(Buyer, Searching for housing), Simon (Buyer, Searching for housing), John(Seller, Posted listing of housing)

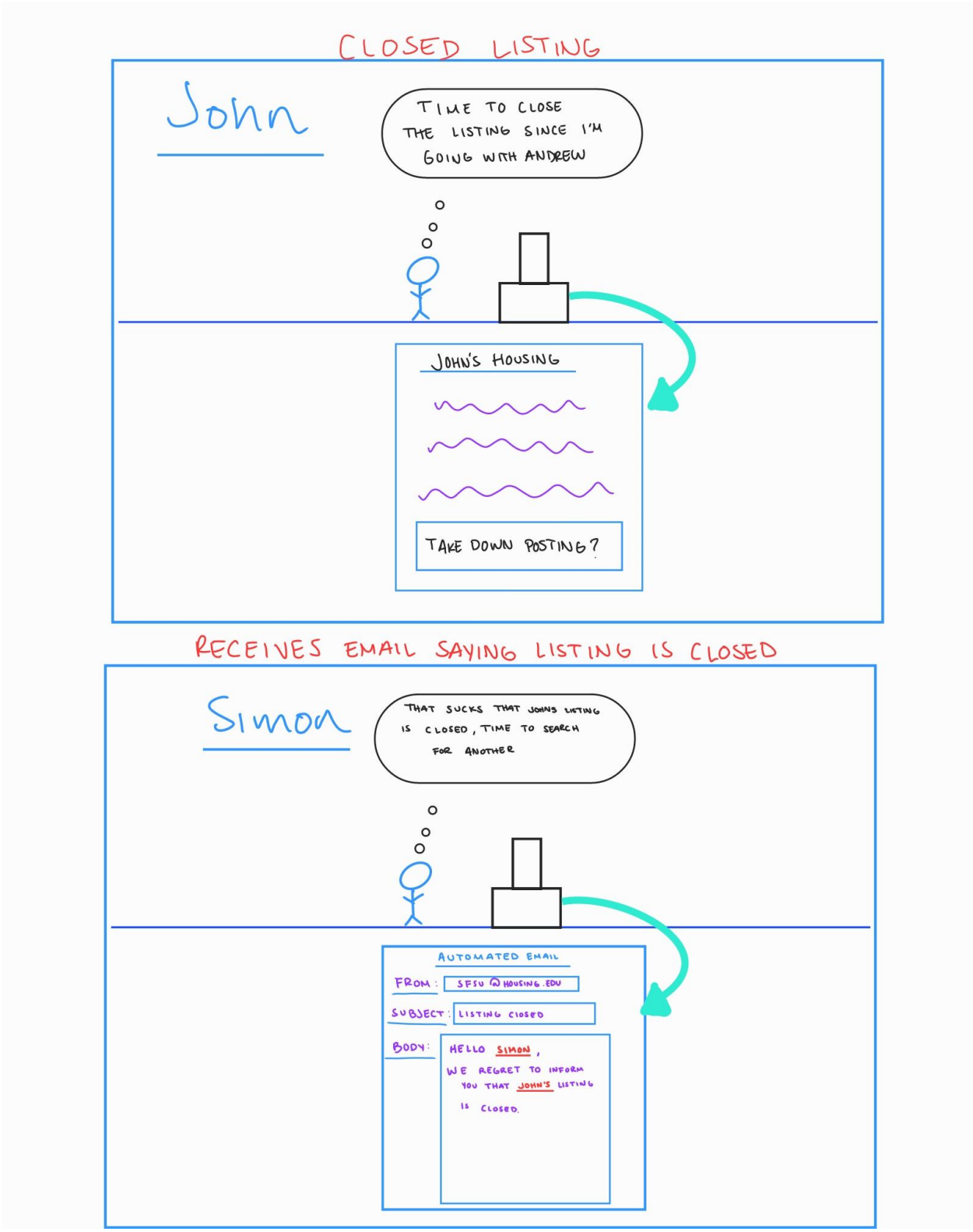



CREATE ACCOUNT
ANDREW
TIME TO MAKE AN ACCOUNT ON GATOR CONNECTIONS TO SEARCH FOR HOUSING SO IT GUARANTEES I LIVE WITH SFSU STUDENTS!
Simon
TIME TO MAKE AN ACCOUNT ON GATOR CONNECTIONS TO SEARCH FOR HOUSING SO IT GUARANTEES I LIVE WITH SFSU STUDENTS!
John
TIME TO MAKE AN ACCOUNT ON GATOR CONNECTION SO I CAN SELL MY HOUSING LISTING!
REGISTER
ENTER NAME
ENTER SFSU EMAIL
ENTER PASSWORD
AGREE TO TERMS
CREATE ACCOUNT

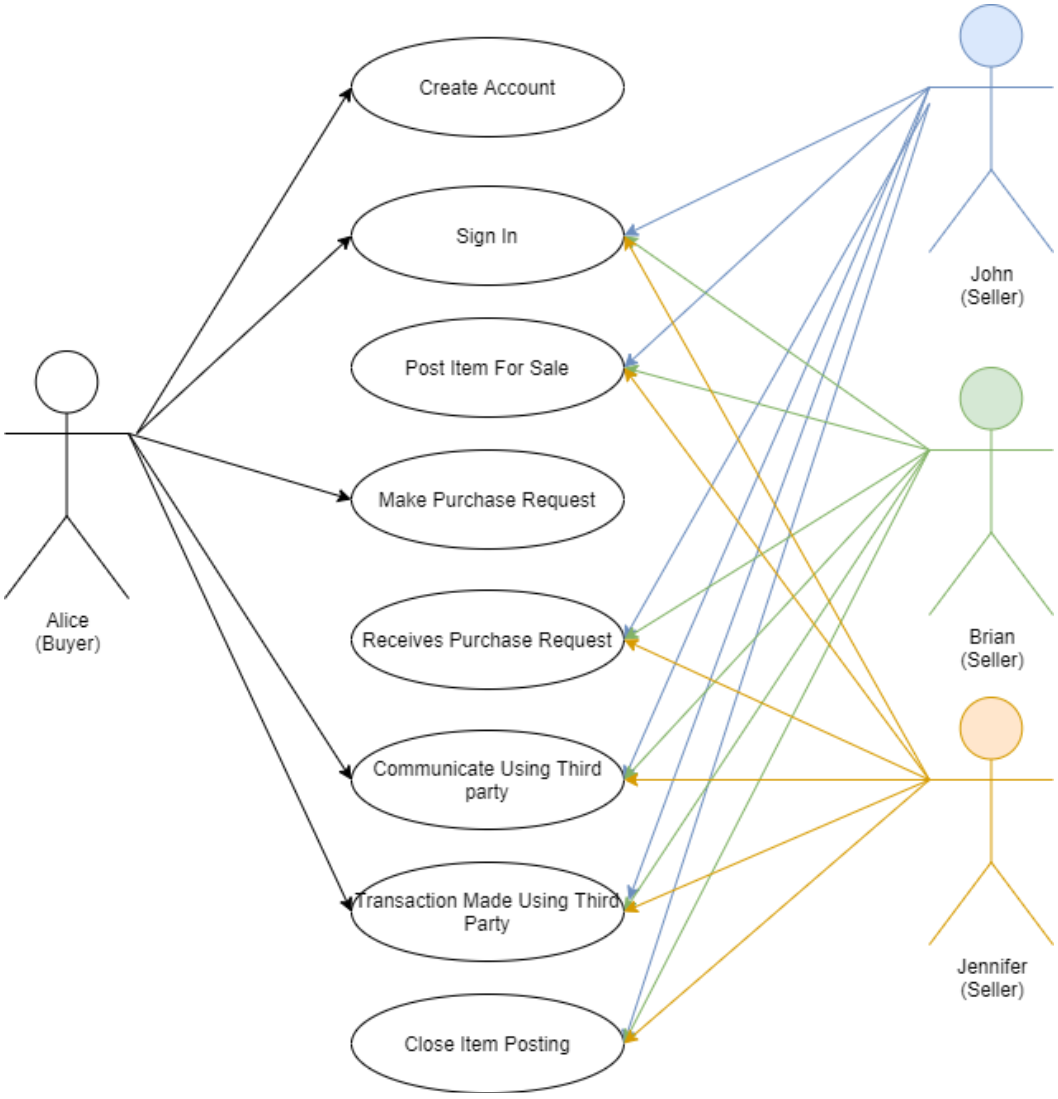

LOG IN
ANDREW
NOW THAT I HAVE AN ACCOUNT, I CAN LOG IN
Simon
NOW THAT I HAVE AN ACCOUNT, I CAN LOG IN
John
NOW THAT I HAVE AN ACCOUNT, I CAN LOG IN
LOG IN
SFSU EMAIL
PASSWORD
LOG IN

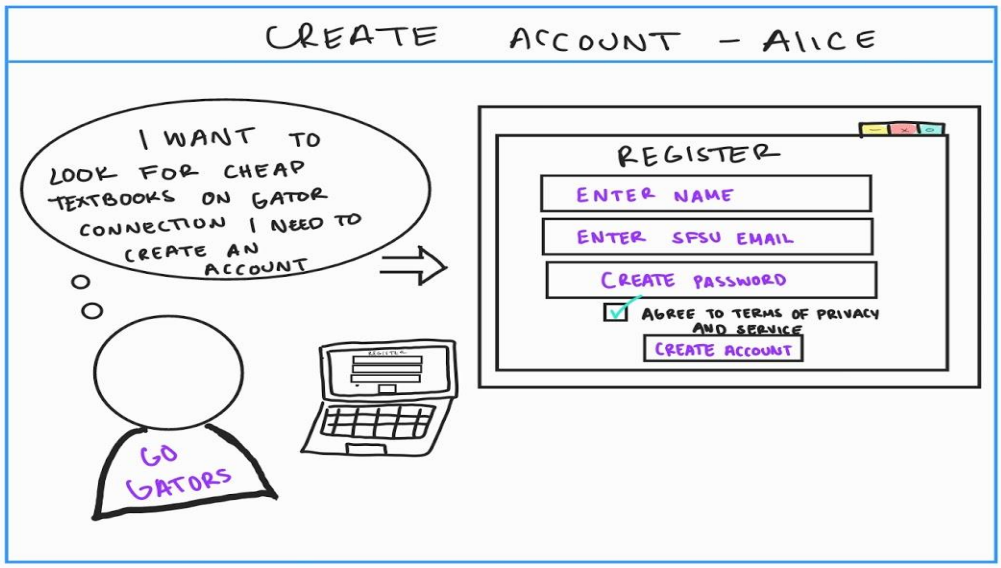

POST HOUSE LISTING
John
TIME TO POST MY HOUSING LISTING!
ENTER TITLE
ENTER CONDITION OF HOUSING
ENTER LOCATION
ENTER PRICING
ENTER CONTACT INFO
ENTER PREFFERED PAYMENT
UPLOAD PICTURE
POST
SEARCH FOR HOUSING LISTING
ANDREW
TIME TO SEARCH FOR HOUSING
Simon
TIME TO SEARCH FOR HOUSING
HOUSING
SEARCH FOR
DROP DOWN BAR
HOUSING 1
HOUSING 2
HOUSING 3

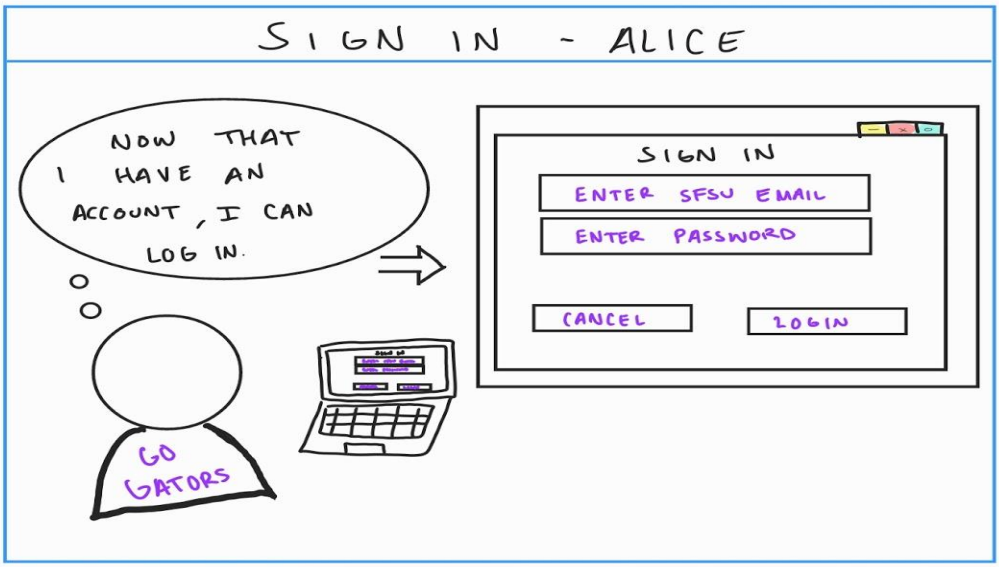

MAKE REQUEST TO BUY LISTING
ANDREW
THIS HOUSING LISTING FROM JOHN LOOKS NICE, I'll MESSAGE HIM ABOUT IT!
SIMON
THIS HOUSING LISTING FROM JOHN LOOKS NICE, I'll MESSAGE HIM ABOUT IT!
JOHN'S HOUSE
SF
$800
555-555-555
VENMO FOR RENT
PICTURE OF HOUSE
RECIEVE REQUEST TO BUY LISTING
John
I HAVE A TEXT FROM ANDREW AND SIMON REGARDING THE HOUSING, I THINK I'll GO WITH ANDREW!
PHONE
ANDREW: HEY I'M INTERESTED IN THE HOUSE....
SIMON: HEY I'M INTERESTED IN THE HOUSE...

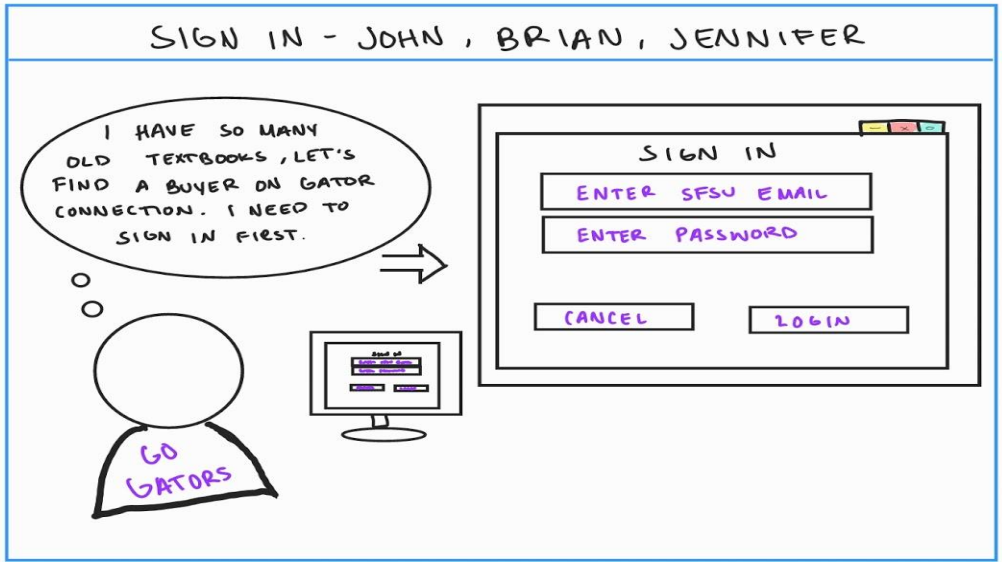

CLOSED LISTING
John
TIME TO CLOSE THE LISTING SINCE I'M GOING WITH ANDREW
JOHN'S HOUSING
TAKE DOWN POSTING?
RECEIVES EMAIL SAYING LISTING IS CLOSED
Simon
THAT SUCKS THAT JOHNS LISTING IS CLOSED, TIME TO SEARCH FOR ANOTHER
AUTOMATED EMAIL
FROM: SFSU @ HOUSING.EDU
SUBJECT: LISTING CLOSED
BODY:
HELLO SIMON,
WE REGRET TO INFORM YOU THAT JOHN'S LISTING IS CLOSED.

Case 2: Student at SFSU looking for cheaper textbooks
Actors: Alice(Buyer), John(Seller), Brian(Seller), Jennifer(Seller)

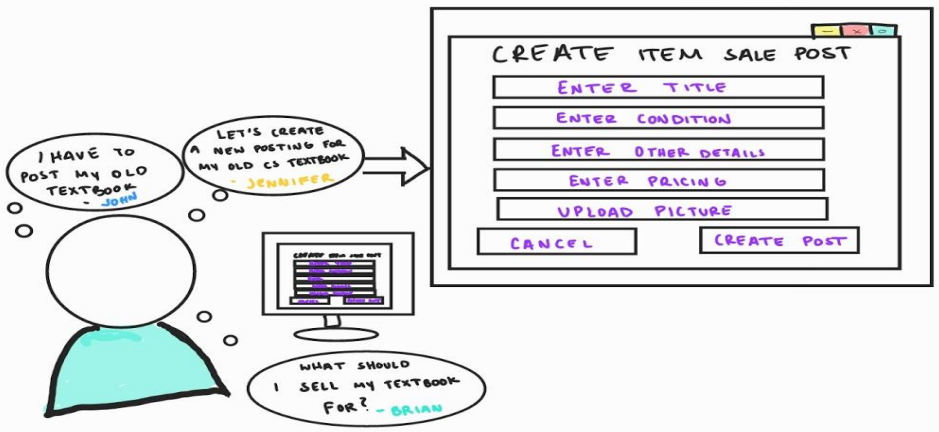



CREATE ACCOUNT - ALICE
I WANT TO LOOK FOR CHEAP TEXTBOOKS ON GATOR CONNECTION I NEED TO CREATE AN ACCOUNT
REGISTER
ENTER NAME
ENTER SFSU EMAIL
CREATE PASSWORD
AGREE TO TERMS OF PRIVACY AND SERVICE
CREATE ACCOUNT
GO GATORS

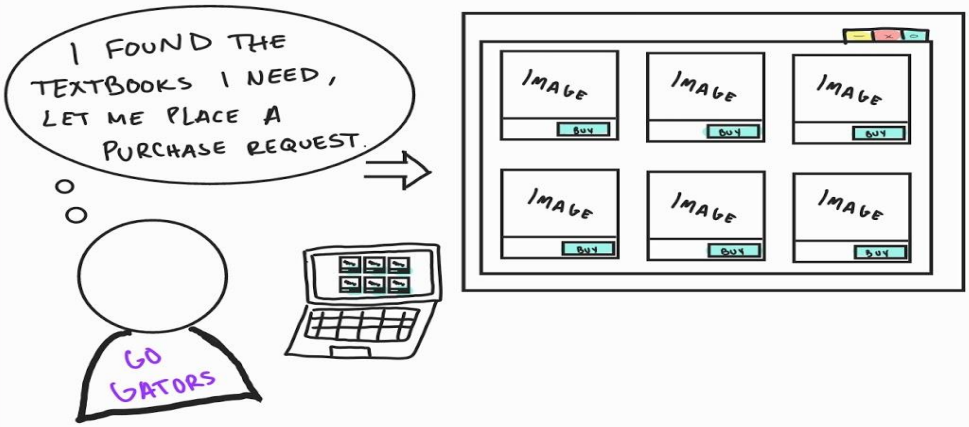

SIGN IN - ALICE
NOW THAT I HAVE AN ACCOUNT, I CAN LOG IN.
SIGN IN
ENTER SFSU EMAIL
ENTER PASSWORD
CANCEL
LOGIN
GO GATORS


SIGN IN - JOHN, BRIAN, JENNIFER
I HAVE SO MANY OLD TEXTBOOKS, LET'S FIND A BUYER ON GATOR CONNECTION. I NEED TO SIGN IN FIRST.
SIGN IN
ENTER SFSU EMAIL
ENTER PASSWORD
CANCEL
LOGIN
GO GATORS

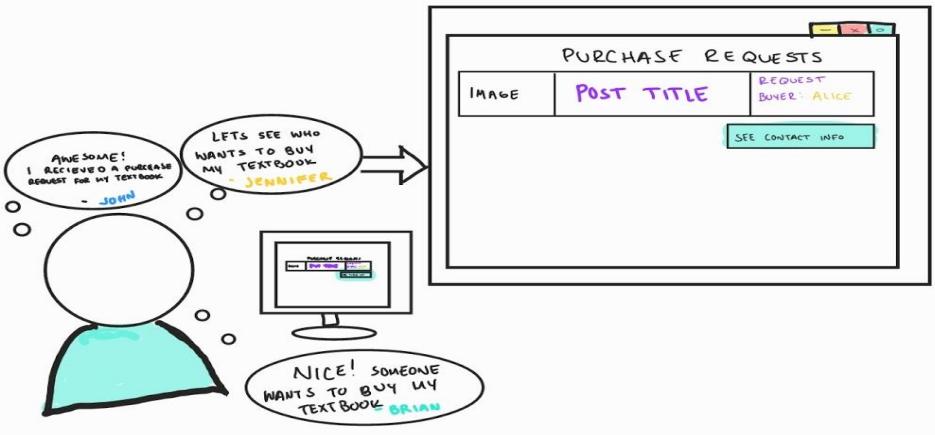
POST ITEM - JOHN, BRIAN, JENNIFER

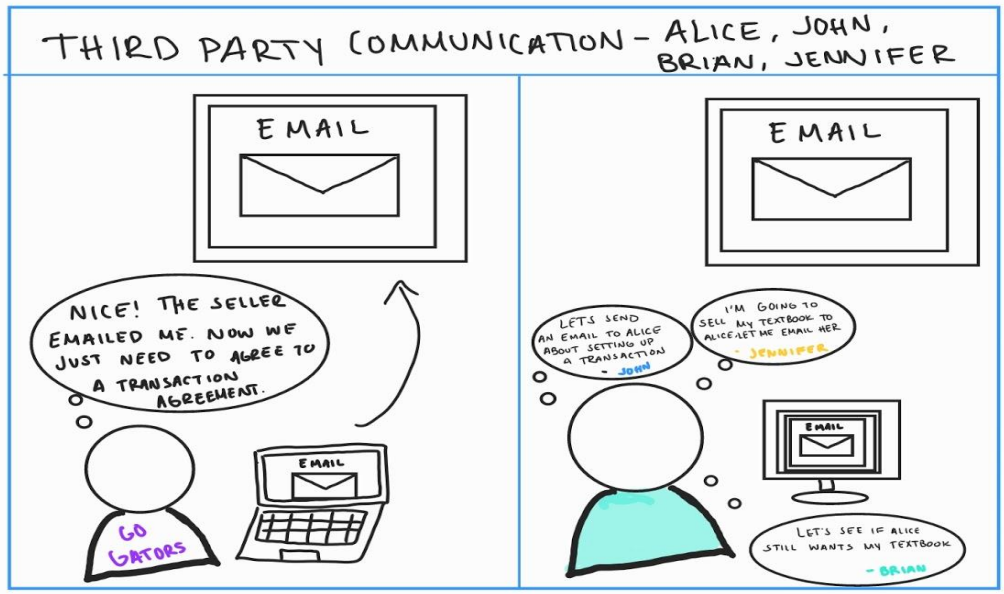


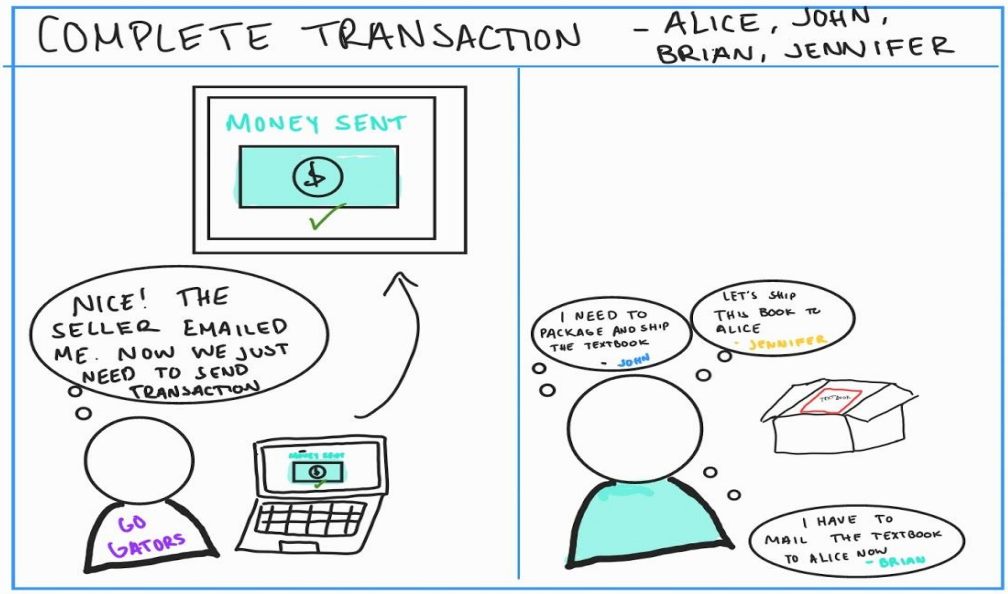
MAKE PURCHASE REQUEST - ALICE

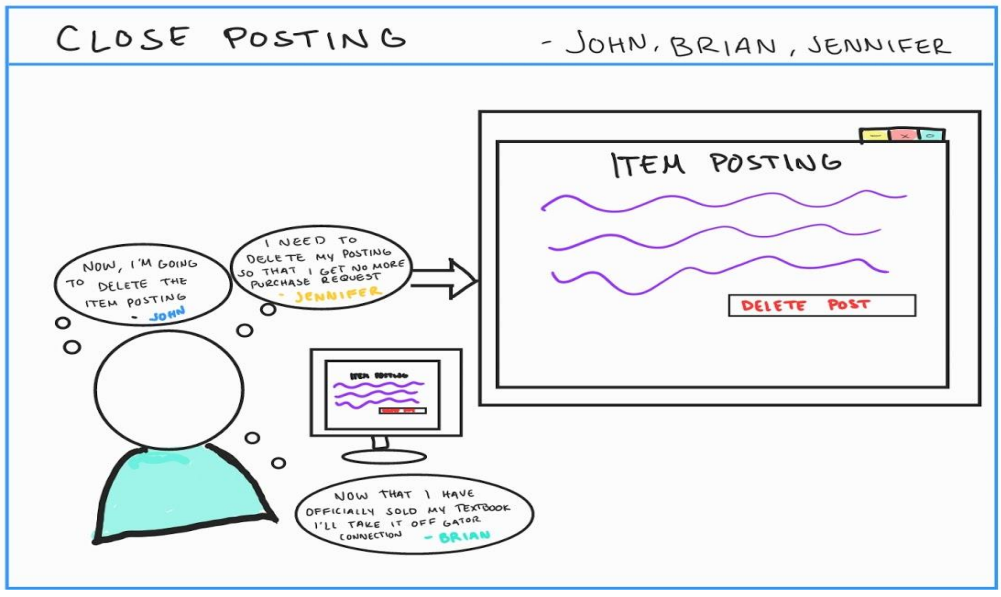


RECIEVING PURCHASE REQUEST - JOHN, BRIAN, JENNIFER

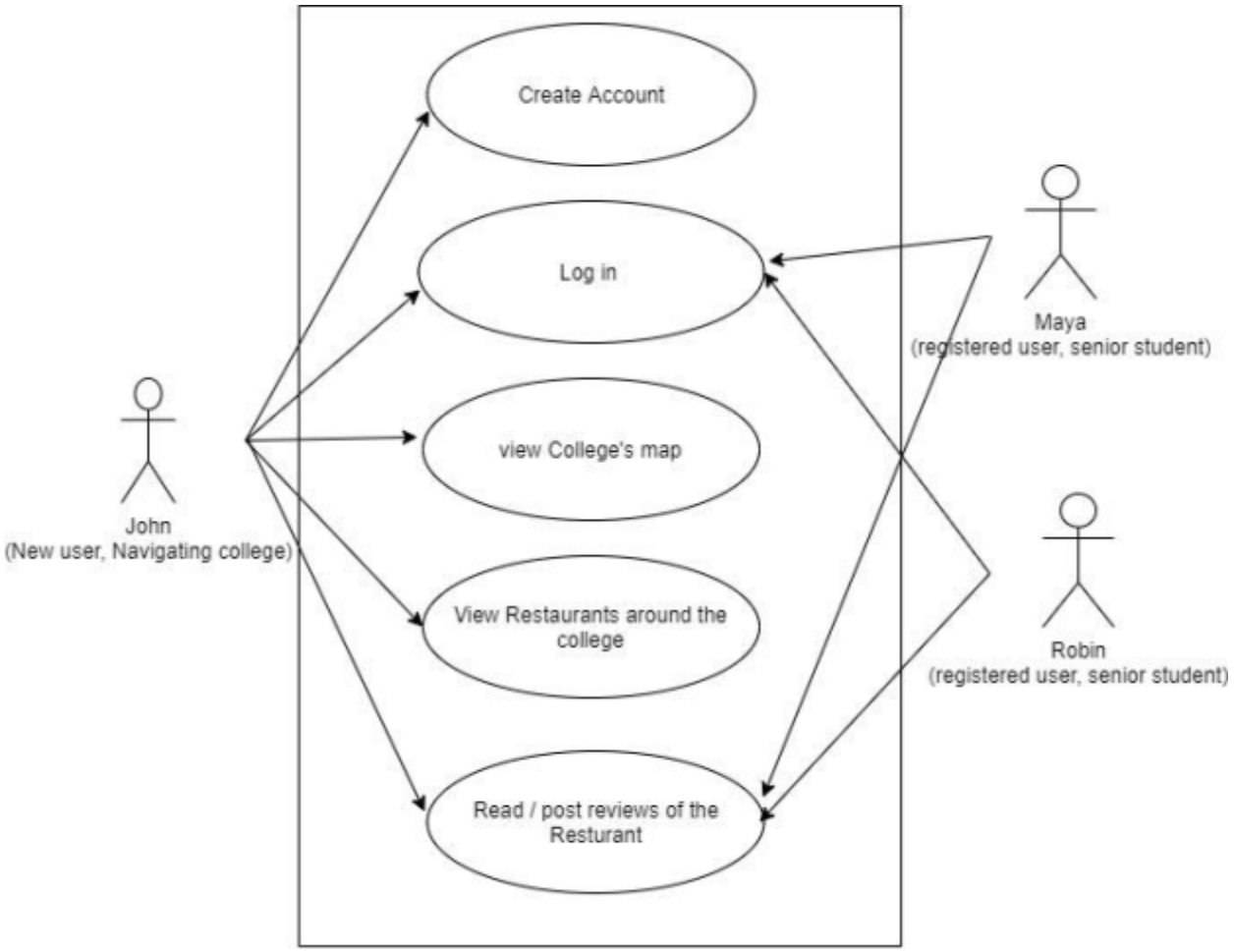


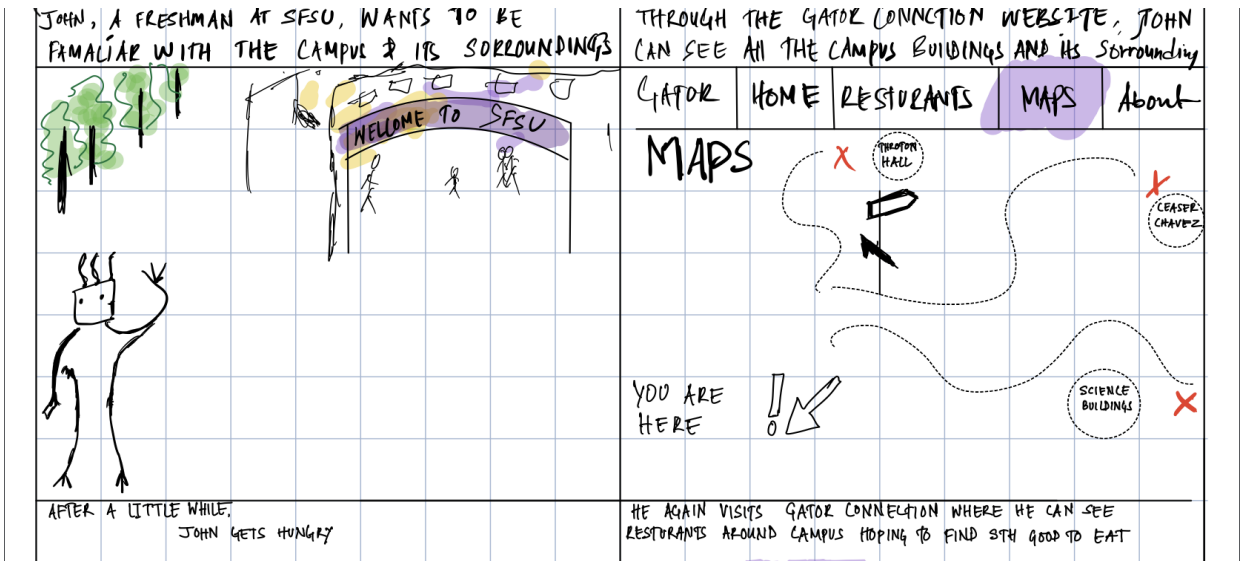

THIRD PARTY COMMUNICATION - ALICE, JOHN, BRIAN, JENNIFER
EMAIL
NICE! THE SELLER EMAILED ME. NOW WE JUST NEED TO AGREE TO A TRANSACTION AGREEMENT.
EMAIL
GO GATORS
EMAIL
LETS SEND AN EMAIL TO ALICE ABOUT SETTING UP A TRANSACTION - JOHN
I'M GOING TO SELL MY TEXTBOOK TO ALICE LET ME EMAIL HER - JENNIFER
EMAIL
LET'S SEE IF ALICE STILL WANTS MY TEXTBOOK - BRIAN

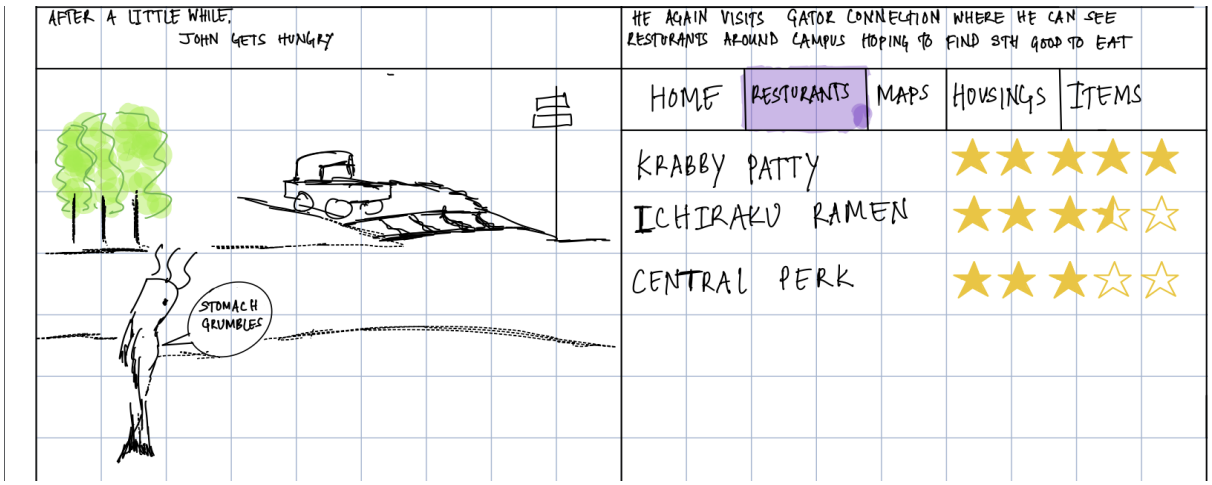

COMPLETE TRANSACTION - ALICE, JOHN, BRIAN, JENNIFER
MONEY SENT
NICE! THE SELLER EMAILED ME. NOW WE JUST NEED TO SEND TRANSACTION
GO GATORS
I NEED TO PACKAGE AND SHIP THE TEXTBOOK - JOHN
LET'S SHIP THIS BOOK TO ALICE - JENNIFER
I HAVE TO MAIL THE TEXTBOOK TO ALICE NOW - BRIAN

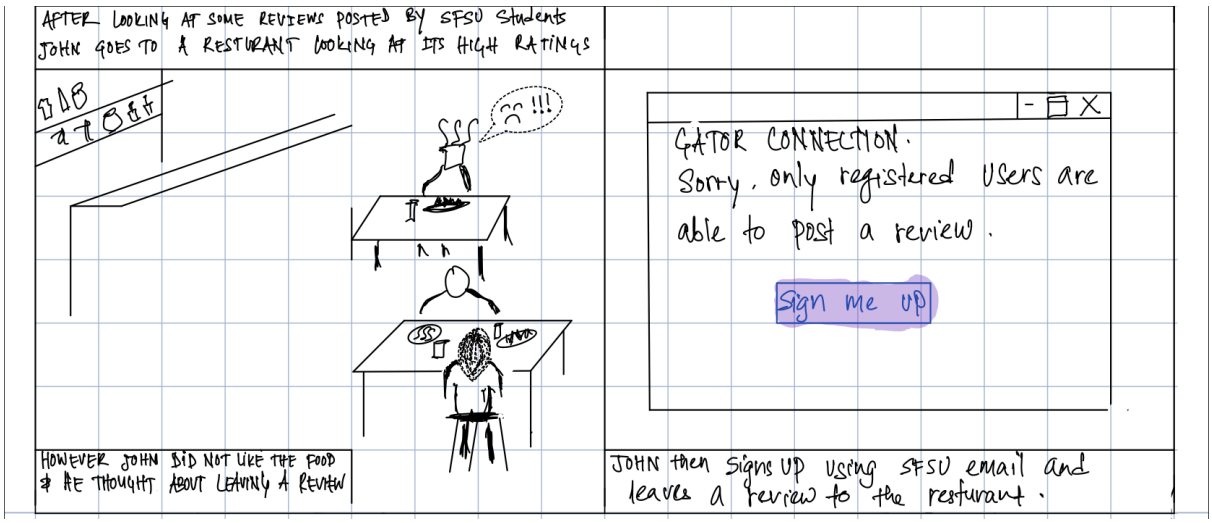

CLOSE POSTING - JOHN, BRIAN, JENNIFER
ITEM POSTING
DELETE POST
NOW, I'M GOING TO DELETE THE ITEM POSTING - JOHN
I NEED TO DELETE MY POSTING SO THAT I GET NO MORE PURCHASE REQUEST - JENNIFER
NOW THAT I HAVE OFFICIALLY SOLD MY TEXTBOOK I'LL TAKE IT OFF GATOR CONNECTION - BRIAN

Use Case 3: Student at SFSU looking for food on campus, with his two friends outside SFSU accompanying him
Actors: John(New User), Maya(Unregistered User), Robin(Unregistered User)

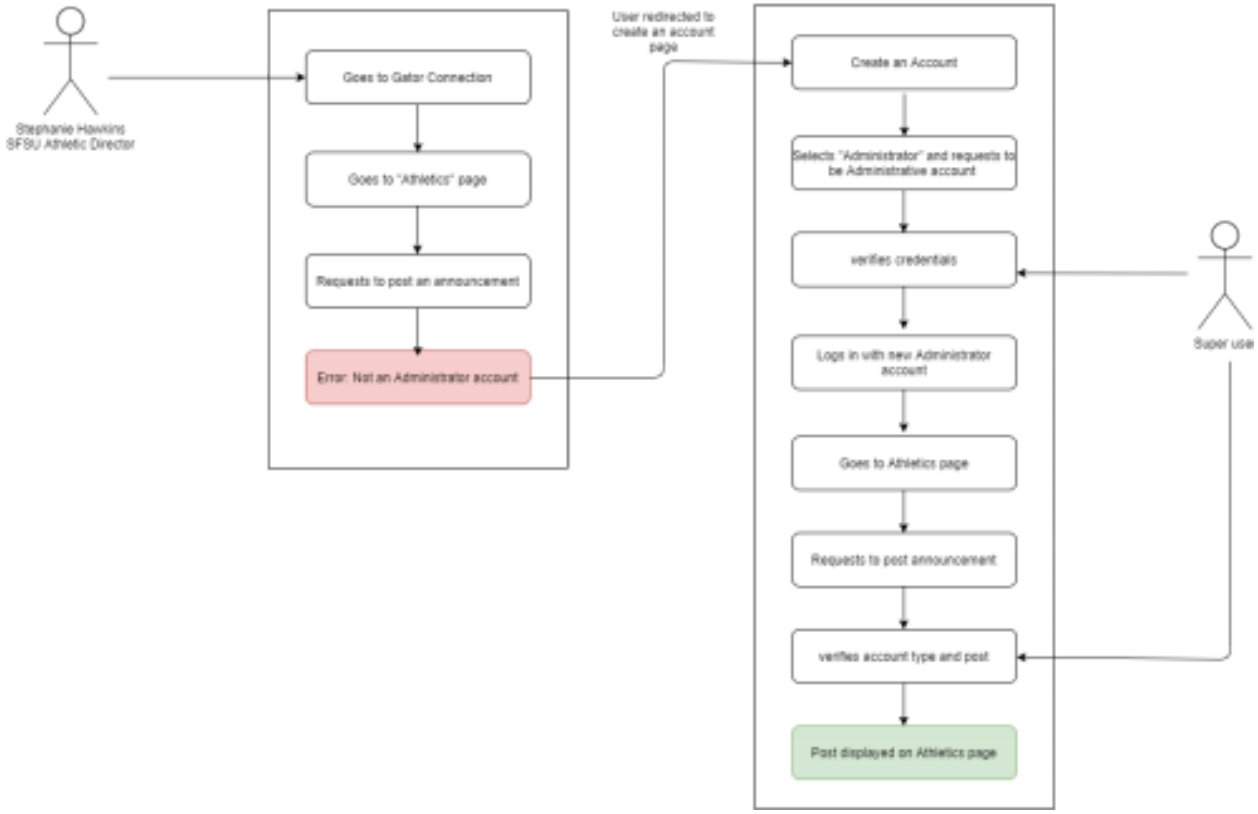



JOHN, A FRESHMAN AT SFSU, WANTS TO BE FAMALIAR WITH THE CAMPUS & ITS SORROUNDINGS
WELCOME TO SFSU
THROUGH THE GATOR CONNCTION WEBSITE, JOHN CAN SEE ALL THE CAMPUS BUILDINGS AND ITS Sorrounding
GATOR
HOME
RESTURANTS
MAPS
About
MAPS
THORTON HALL
CESAR CHAVEZ
YOU ARE HERE
SCIENCE BUILDINGS
AFTER A LITTLE WHILE,
JOHN GETS HUNGRY
HE AGAIN VISITS GATOR CONNECTION WHERE HE CAN SEE RESTORANTS AROUND CAMPUS HOPING TO FIND STH GOOD TO EAT

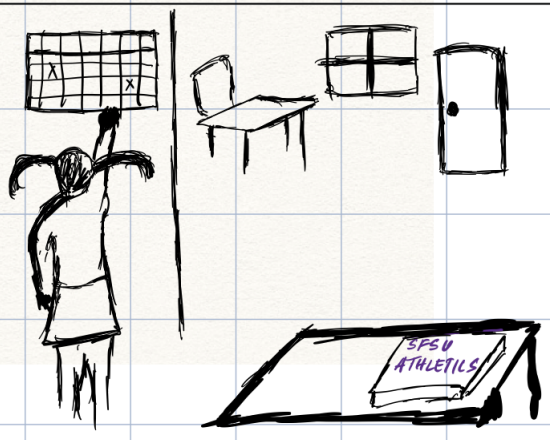

AFTER A LITTLE WHILE,
JOHN GETS HUNGRY
STOMACH GRUMBLES
HE AGAIN VISITS GATOR CONNECTION WHERE HE CAN SEE RESTORANTS AROUND CAMPUS HOPING TO FIND STH GOOD TO EAT
HOME
RESTURANTS
MAPS
HOUSINGS
ITEMS
KRABBY PATTY
ICHIRAKU RAMEN
CENTRAL PERK


AFTER LOOKING AT SOME REVIEWS POSTED BY SFSU Students JOHN GOES TO A RESTURANT LOOKING AT ITS HIGH RATINGS
!!!
HOWEVER JOHN DID NOT LIKE THE FOOD & HE THOUGHT ABOUT LEAVING A REVIEW
GATOR CONNECTION.
Sorry, only registered Users are able to post a review.
Sign me up
JOHN then signs UP using SFSU email and leaves a review to the resturant.

Case 4: Athletic Director at SFSU wants to announce upcoming events
Actors: Stephanie(User, Athletic Director)



Stephanie Hawkins is a athletics director in the SFSU



SHE WANTS TO ANNOUNCE the UPCOMING Basketball GAME hosting at SFSU

She realized Most of the students use GATOR connection & decides to post the announcement there.

GATOR CONNECTION

ANNOUNCEMENTS

MAKE NEW ANNOUNCEMENT

EDIT ANNOUNCEMENTS

DELETE ANNOUNCEMENTS

However, making announcements takes special privilege And she gets an error.

GATOR CONNECTION

ANNOUNCEMENTS

Sorry, but announcements can be made only with administrative account if you are a SFSU director please create a administrative account.

Create a administrative account

Stephanie Sucessfully creates an administrative account and could Sucessfully post an announcement & everyone can see it.

GATOR CONNECTION

HOME | SHOP | ANN

ANNOUNCEMENTS

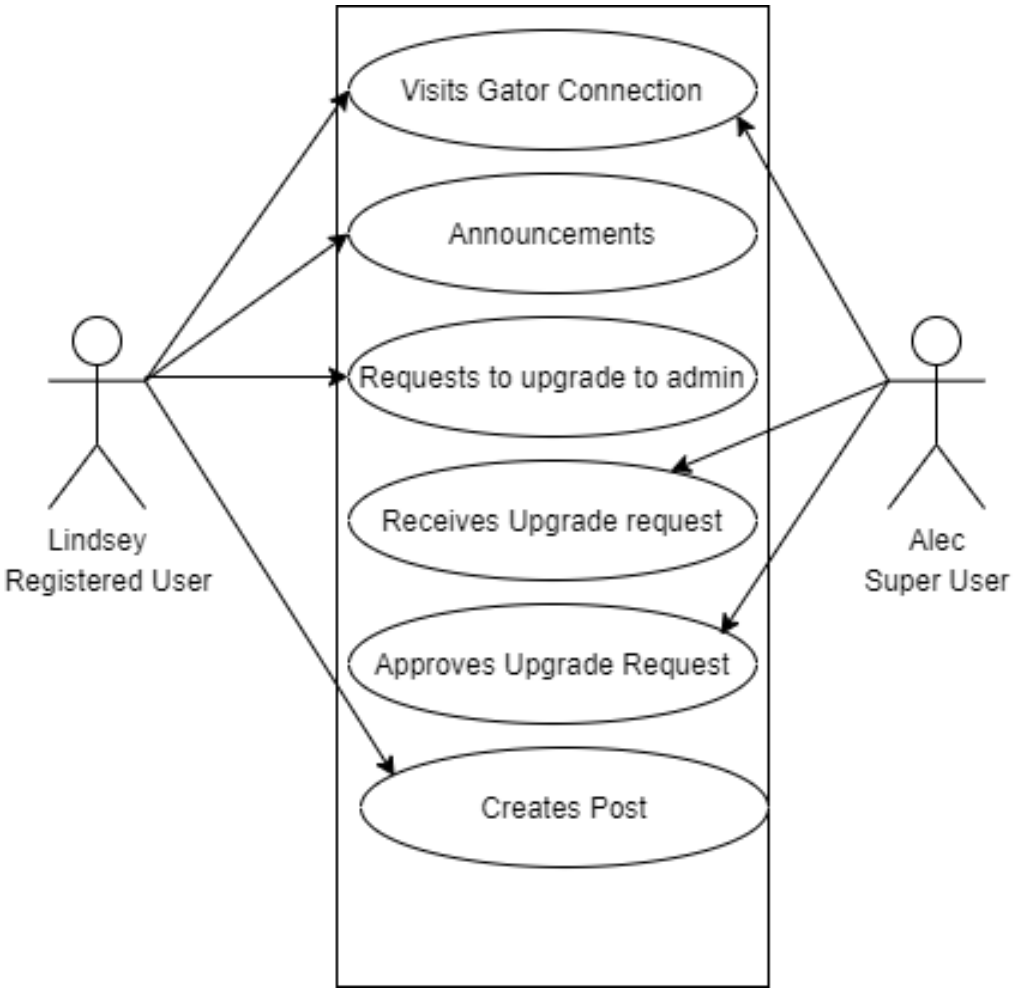
...

ADVERTISEMENTS

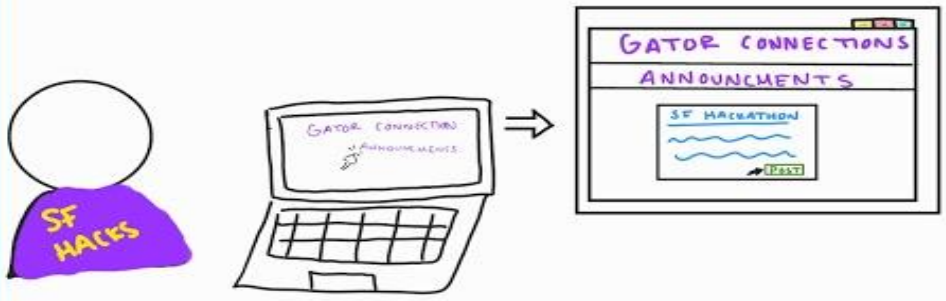
...

Stephanie's announcement is shown on the home page Carasoul.

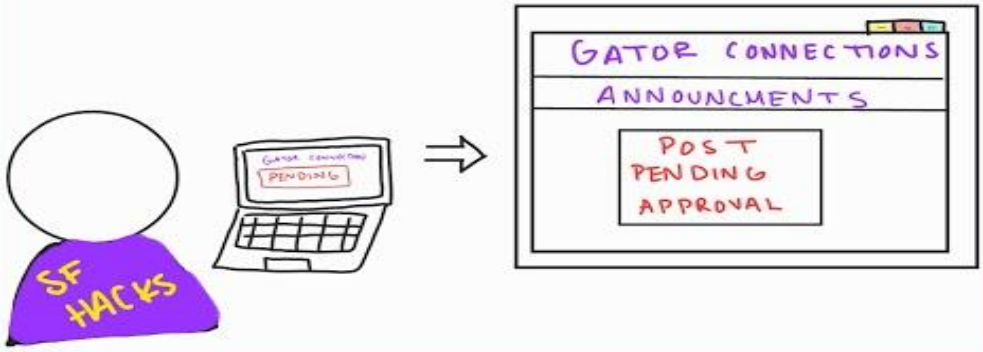
Case 5: President of SF Hacks wants to post about her upcoming events
Actors: Lindsey(Registered User), Alec(Super User)



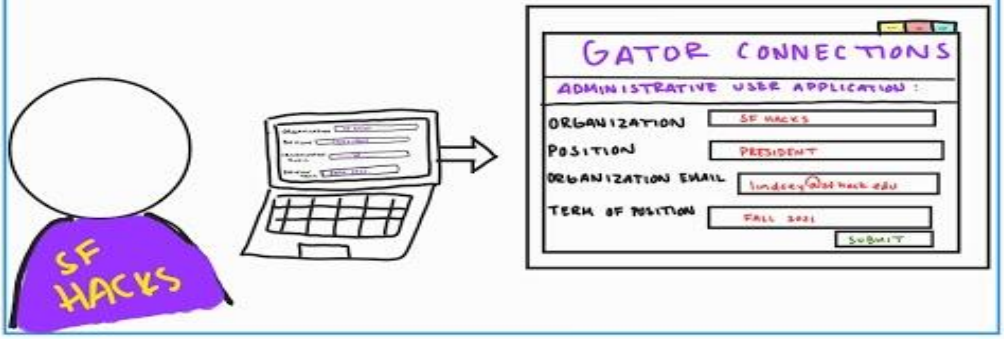
LINDSEY WANTS TO POST SF HACK'S SOCIAL MEDIA, ANNOUNCEMENTS, AND FUTURE EVENTS ON GATOR CONNECTIONS.



LINDSEY TRIES TO POST AN ANNOUNCMENT BUT SHE REALIZES SHE DOESN'T HAVE FULL ACCESS AS ADMINISTRATIVE USER, THEREFORE HER POST IS PENDING.

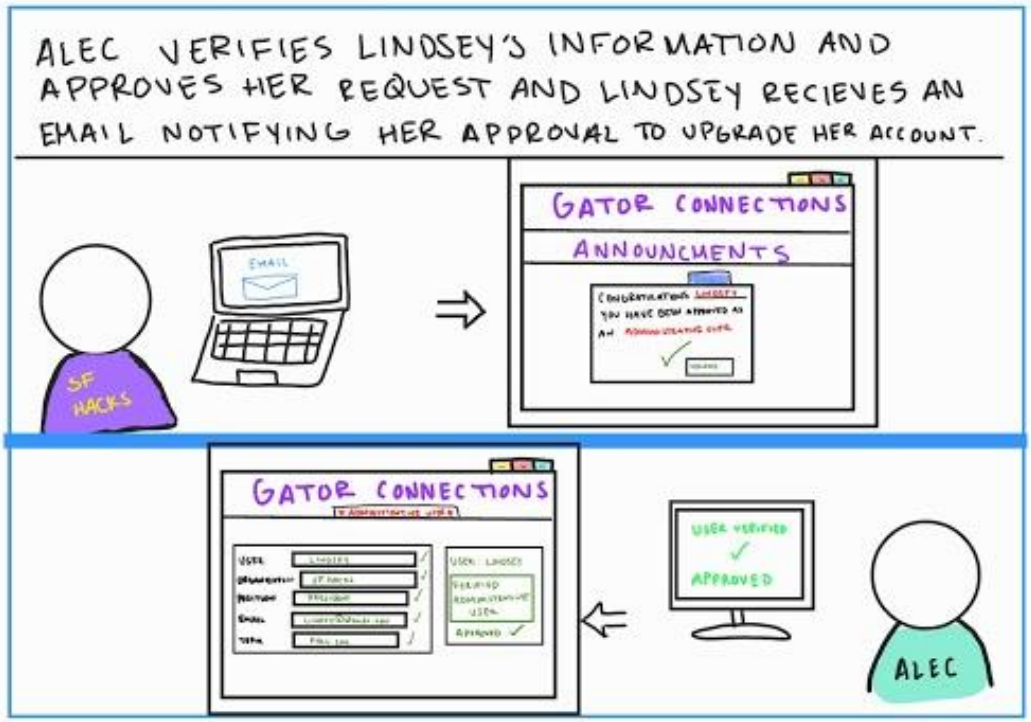


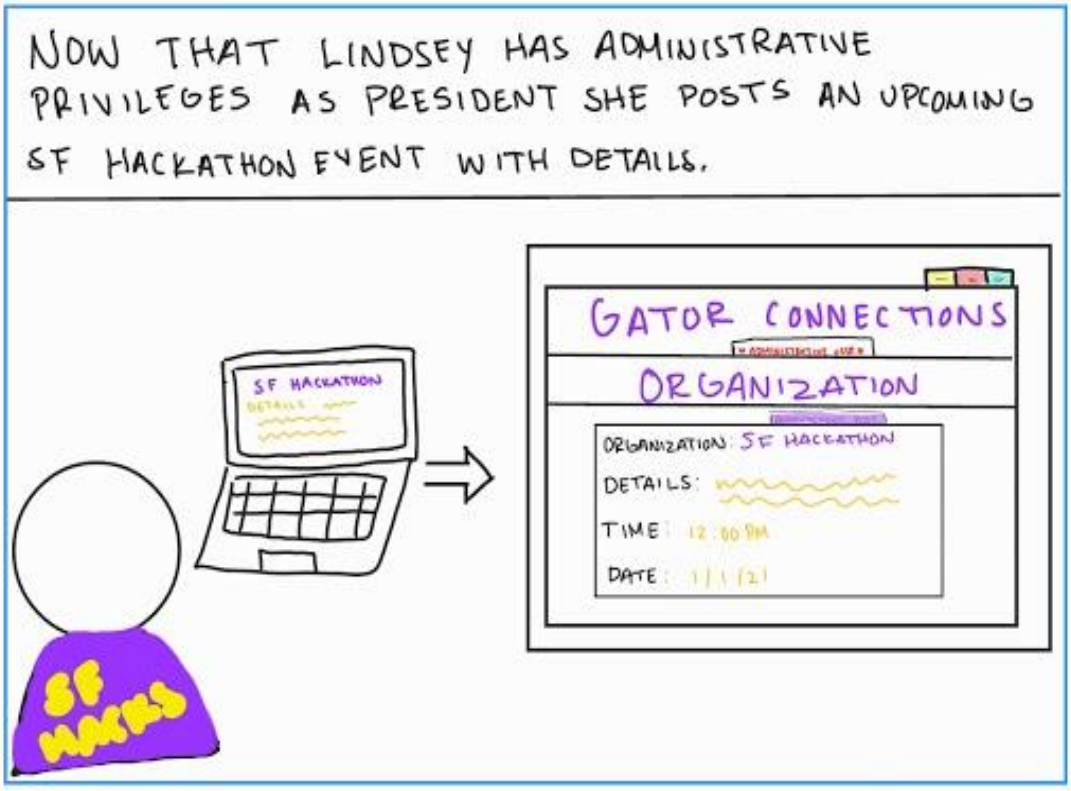
LINDSEY DECIDES THAT SINCE SHE IS PRESIDENT OF SF HACKS SHE CAN APPLY AS AN ADMINISTRATIVE USER SO HER POSTS DON'T HAVE TO GO THROUGH A PENDING PROCCESS

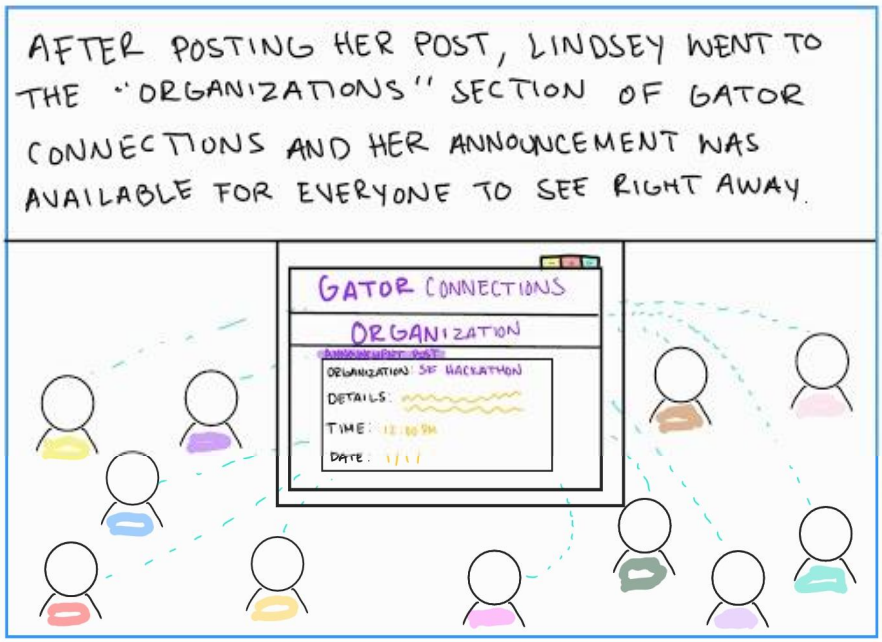


WHILE LINDSEY IS WAITING FOR OFFICIAL CONFIRMATION, SUPER USER ALEC HAS RECEIVED HER REQUEST.

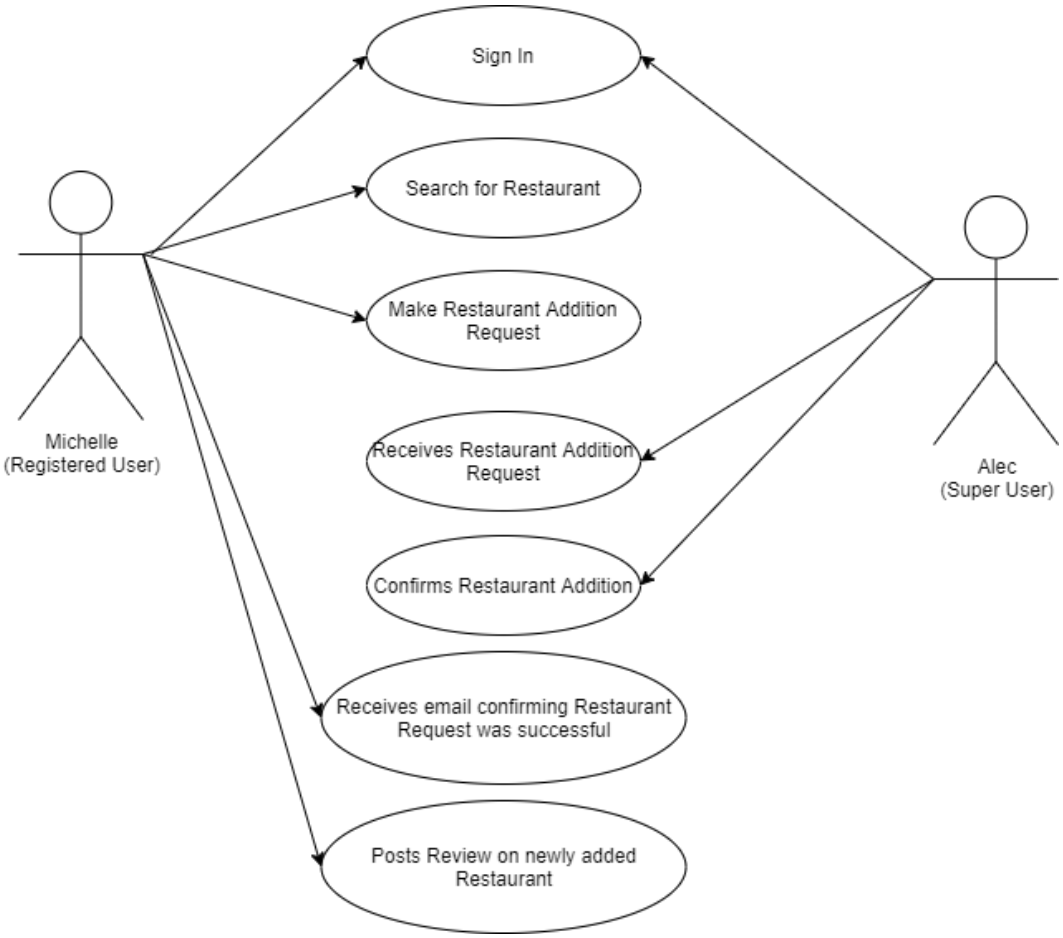
ALEC VERIFIES LINDSEY'S INFORMATION AND APPROVES HER REQUEST AND LINDSEY RECIEVES AN EMAIL NOTIFYING HER APPROVAL TO UPGRADE HER ACCOUNT.

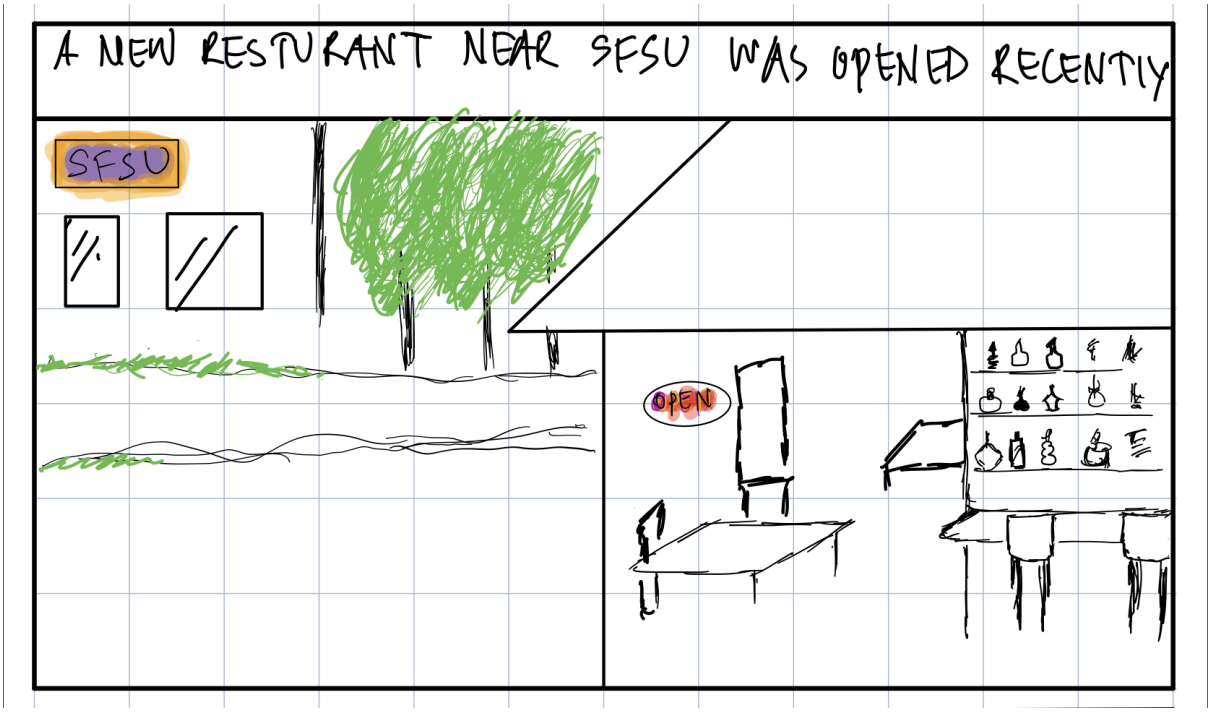


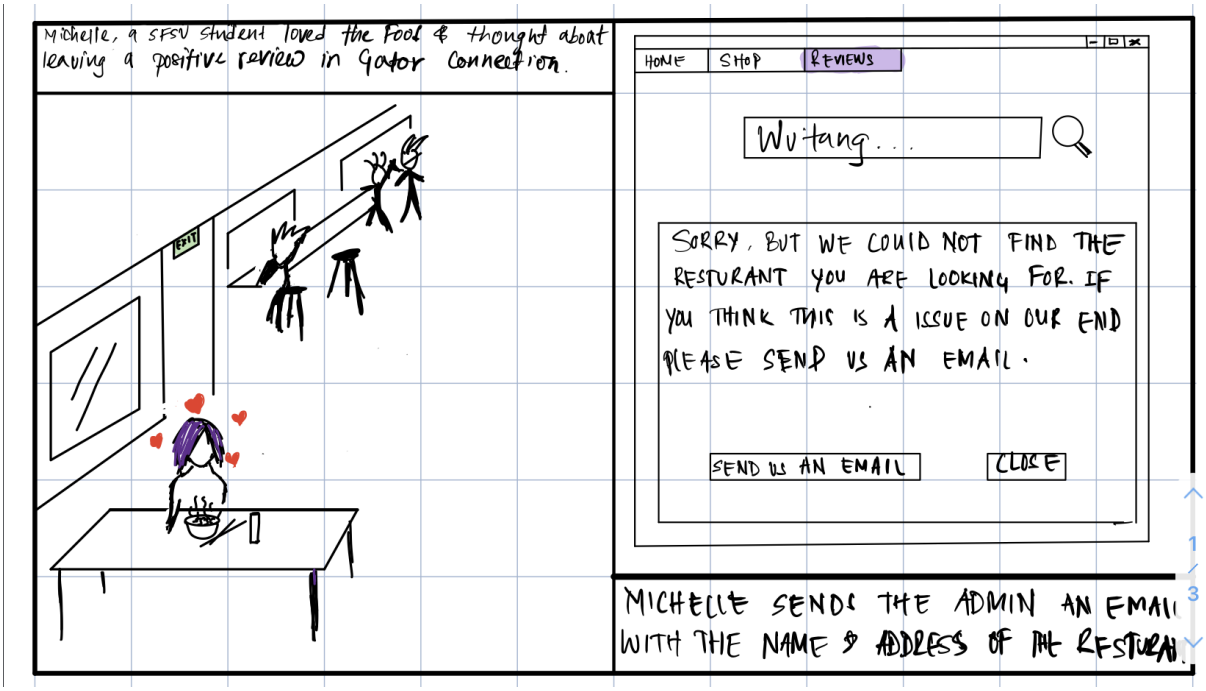

NOW THAT LINDSEY HAS ADMINISTRATIVE PRIVILEGES AS PRESIDENT SHE POSTS AN UPCOMING SF HACKATHON EVENT WITH DETAILS.
SF HACKATHON
DETAILS:
GATOR CONNECTIONS
ORGANIZATION
ORGANIZATION: SF HACKATHON
DETAILS:
TIME:
DATE:
SF HACKS

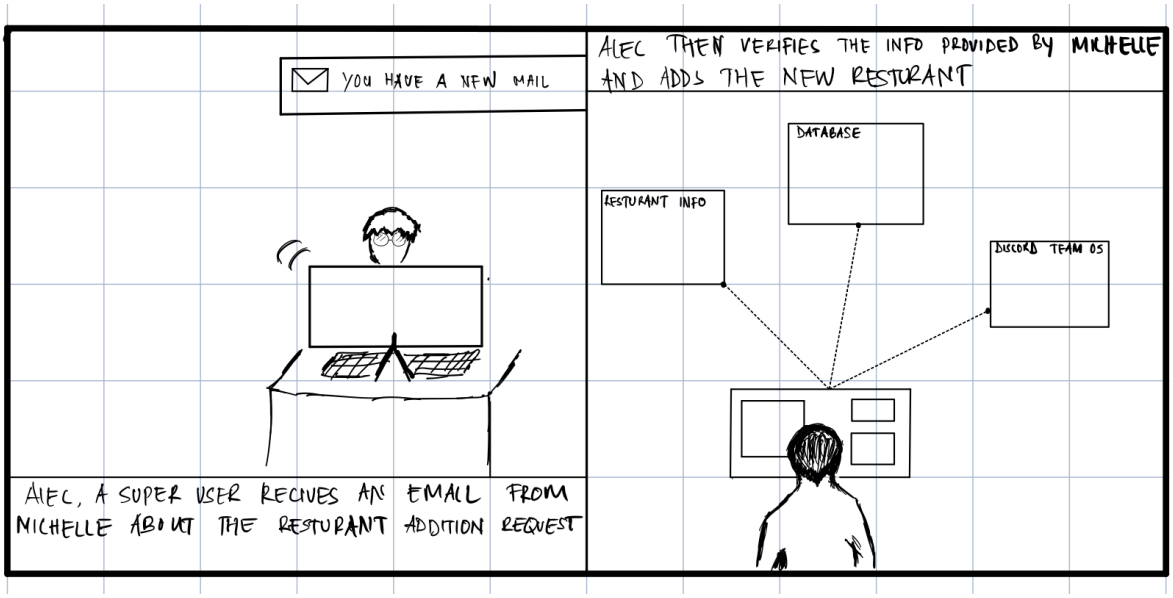

AFTER POSTING HER POST, LINDSEY WENT TO THE "ORGANIZATIONS" SECTION OF GATOR CONNECTIONS AND HER ANNOUNCEMENT WAS AVAILABLE FOR EVERYONE TO SEE RIGHT AWAY.
GATOR CONNECTIONS
ORGANIZATION
ORGANIZATION: SF HACKATHON
DETAILS:
TIME:
DATE:

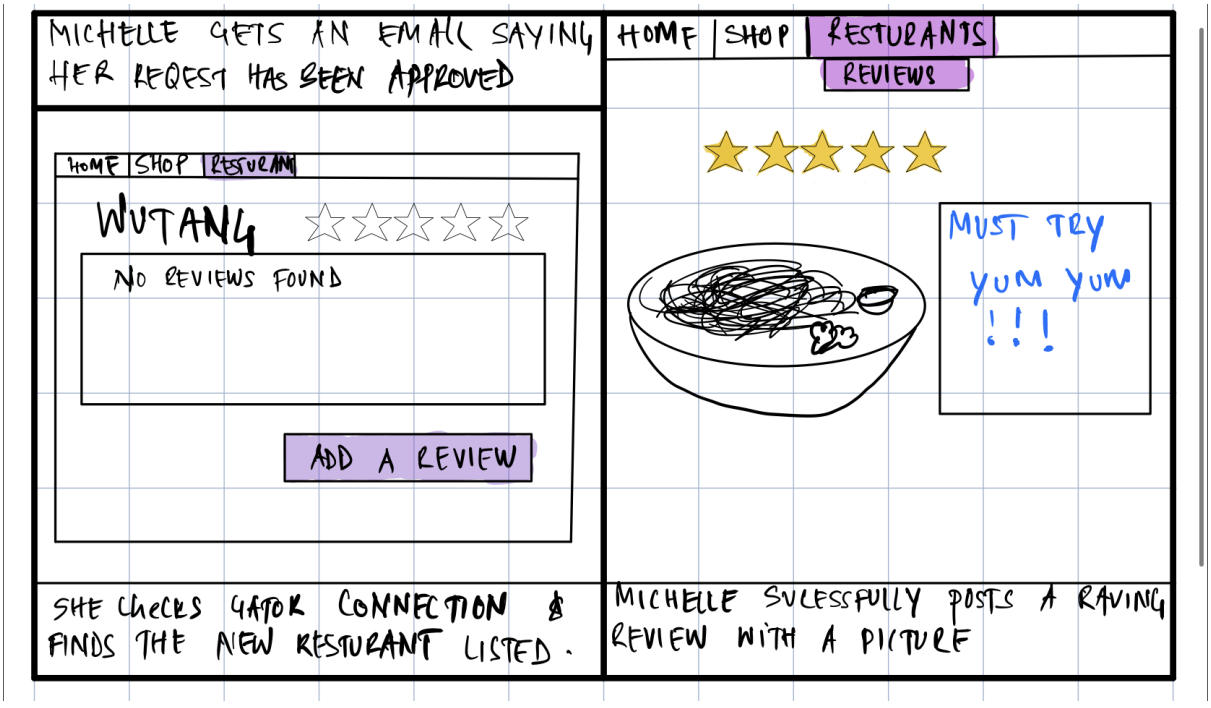
Case 6: Student at SFSU wants to write a review about a restaurant she liked
Actors: Michelle (Registered User), Alec (Super User)



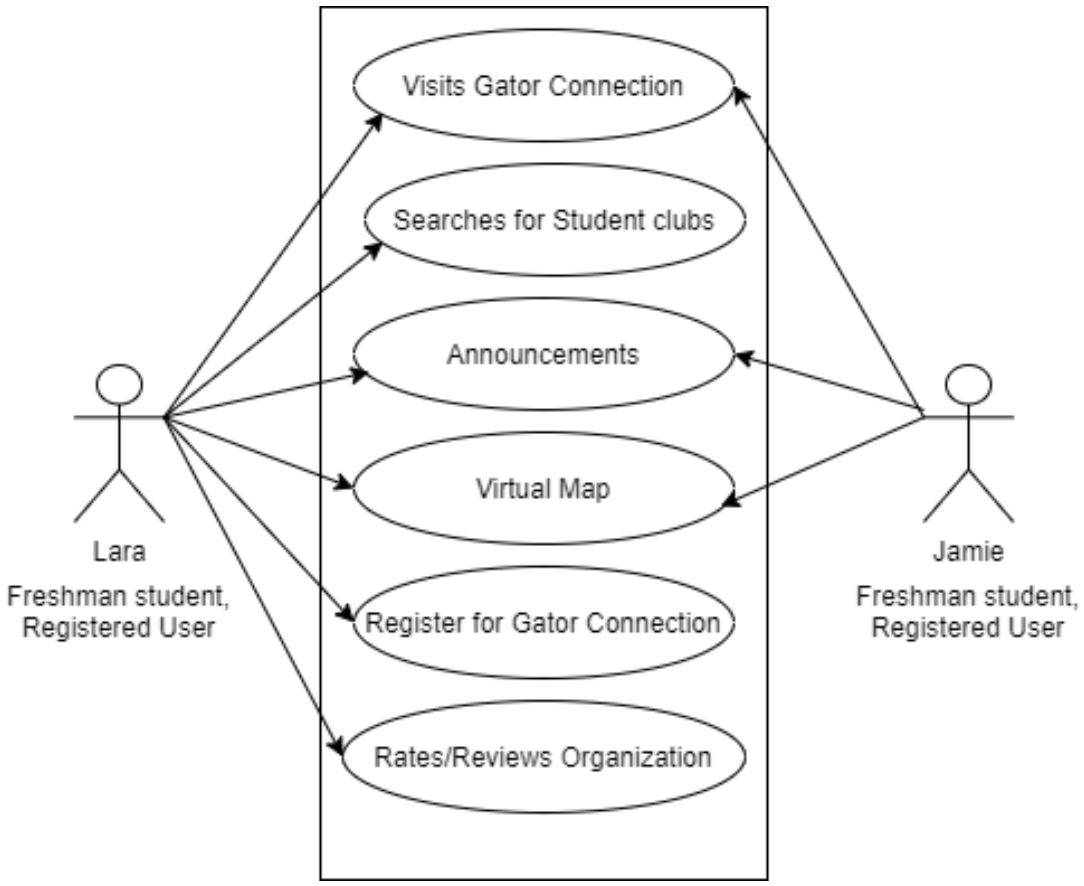

A NEW RESTURANT NEAR SFSU WAS OPENED RECENTLY
SFSU
OPEN


Michelle, a SFSU student loved the food & thought about leaving a positive review in Gator Connection.
EXIT
HOME
SHOP
REVIEWS
Wu-tang...
SORRY, BUT WE COULD NOT FIND THE RESTURANT YOU ARE LOOKING FOR. IF YOU THINK THIS IS A ISSUE ON OUR END PLEASE SEND US AN EMAIL.
SEND US AN EMAIL
CLOSE
MICHELLE SENDS THE ADMIN AN EMAIL WITH THE NAME & ADDRESS OF THE RESTURAN

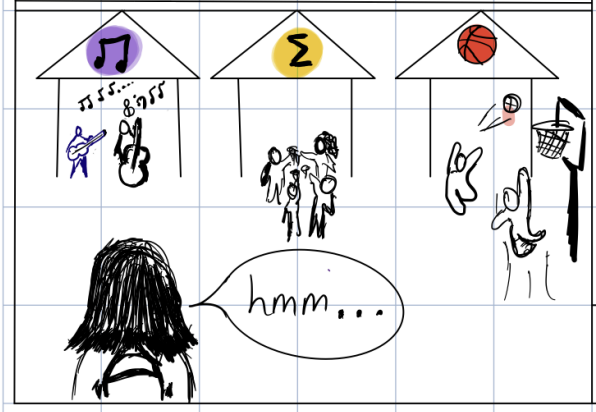

YOU HAVE A NEW MAIL
ALEC, A SUPER USER RECIVES AN EMAIL FROM MICHELLE ABOUT THE RESTURANT ADDITION REQUEST
ALEC THEN VERIFIES THE INFO PROVIDED BY MICHELLE AND ADDS THE NEW RESTURANT
DATABASE
RESTURANT INFO
DISCORD TEAM 05

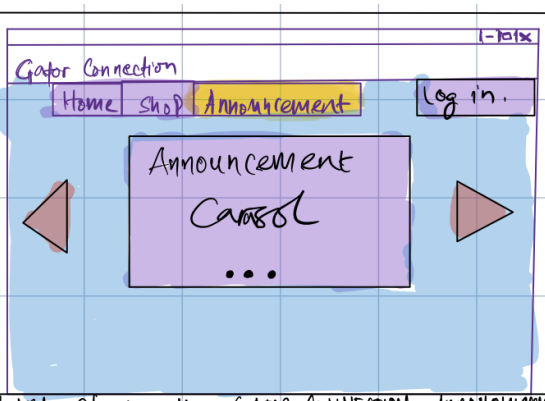

MICHELLE GETS AN EMAIL SAYING HER REQEST HAS BEEN APPROVED
HOME SHOP RESTURANTS
WUTANG
NO REVIEWS FOUND
ADD A REVIEW
SHE CHECKS GATOR CONNECTION & FINDS THE NEW RESTURANT LISTED.
HOME SHOP RESTURANTS
REVIEWS
MUST TRY
YUM YUM
!!!
MICHELLE SUCESSFULLY POSTS A RAVING REVIEW WITH A PICTURE

Case 7: Two Freshman at SFSU want to join a club but want more information
Actors: Lara(Registered User), Jamie(Registered User)



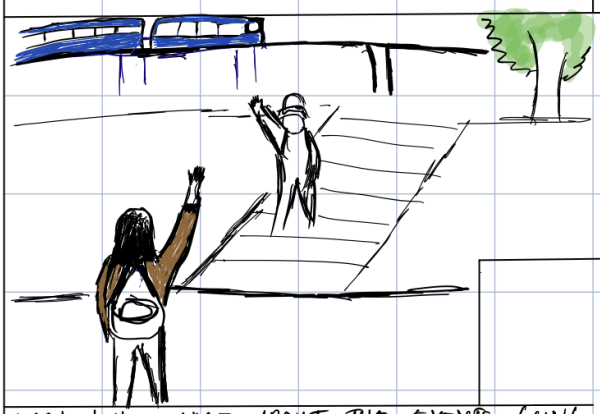
LARA, a freshman at SFSU is deciding which club to join at the campus.





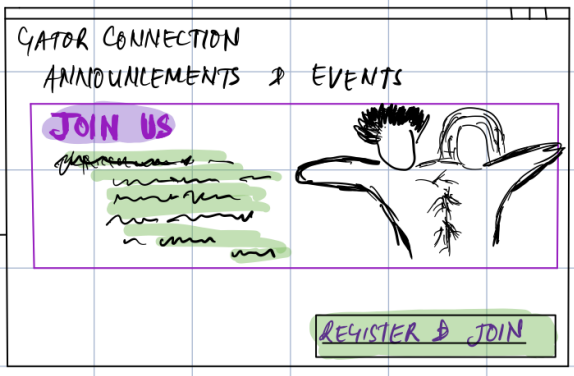
LARA checks the GATOR CONNECTION ANNOUNCEMENT to see if there is any event that may peek her intrest.

LARA MEETS HER HIGH SCHOOL FRIEND JAIME WHO IS ALSO ATTENDING SFSU

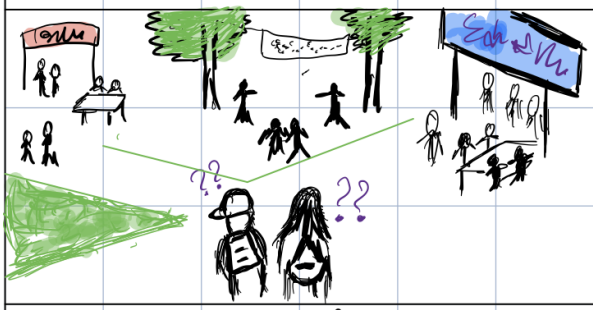


LARA tells JAIME ABOUT THE EVENTS GOING AND THEY BOTH DECIDE TO JOIN A EVENT TOGETHER

JAIME ALSO VISITS GATOR CONNECTION AND THEY BOTH REGISTER FOR AN EVENT THEY LIKE

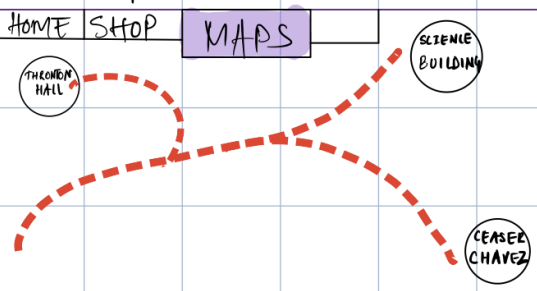


AFTER REGISTERING AND JOINING THE EVENT BOTH GET EMAIL CONFIRMATION



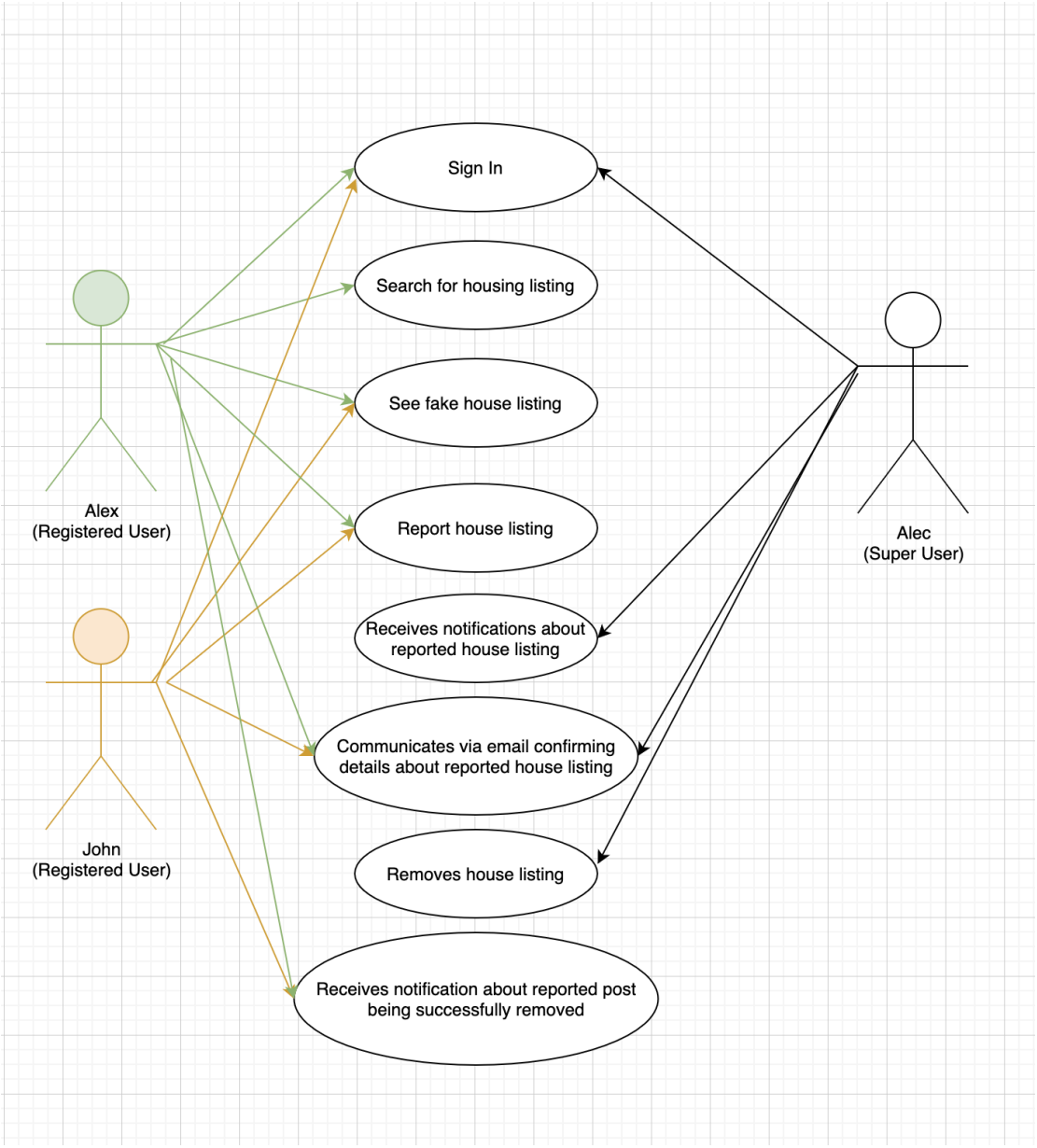
HOWEVER THEY DON'T KNOW WHERE THE EVENT IS HOSTED.

THEY GO TO GATOR CONNECTION ONCE MORE WHERE THEY CAN SEE THE MAP OF THE CAMPUS

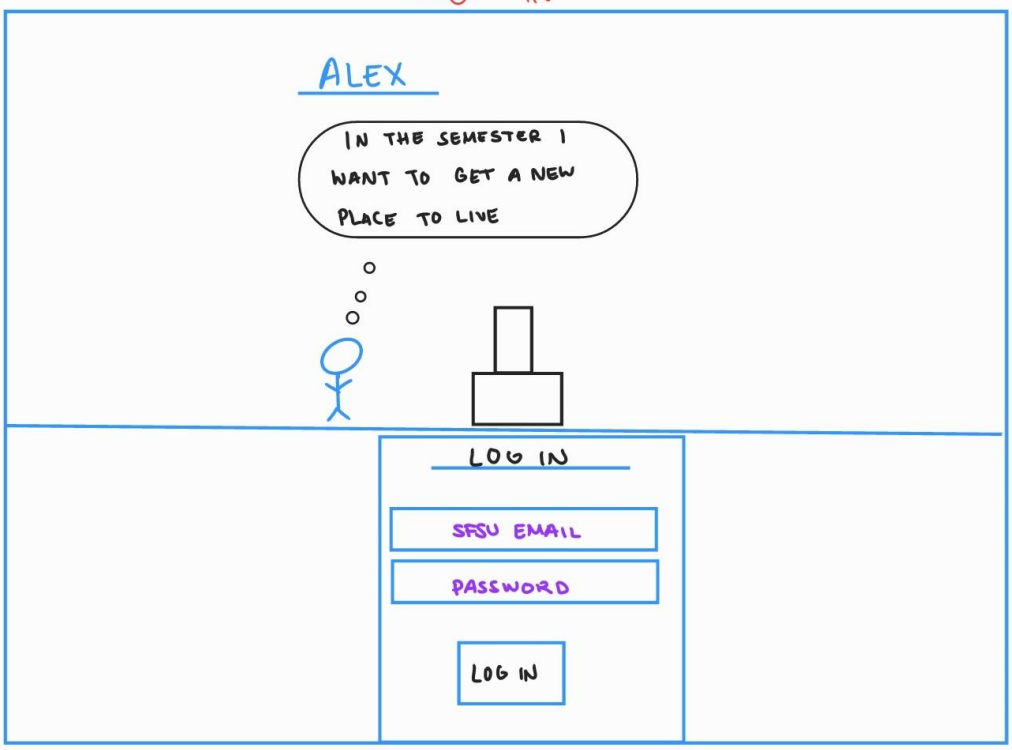


BOTH SUCESSFULLY ATTENDS THE EVENT

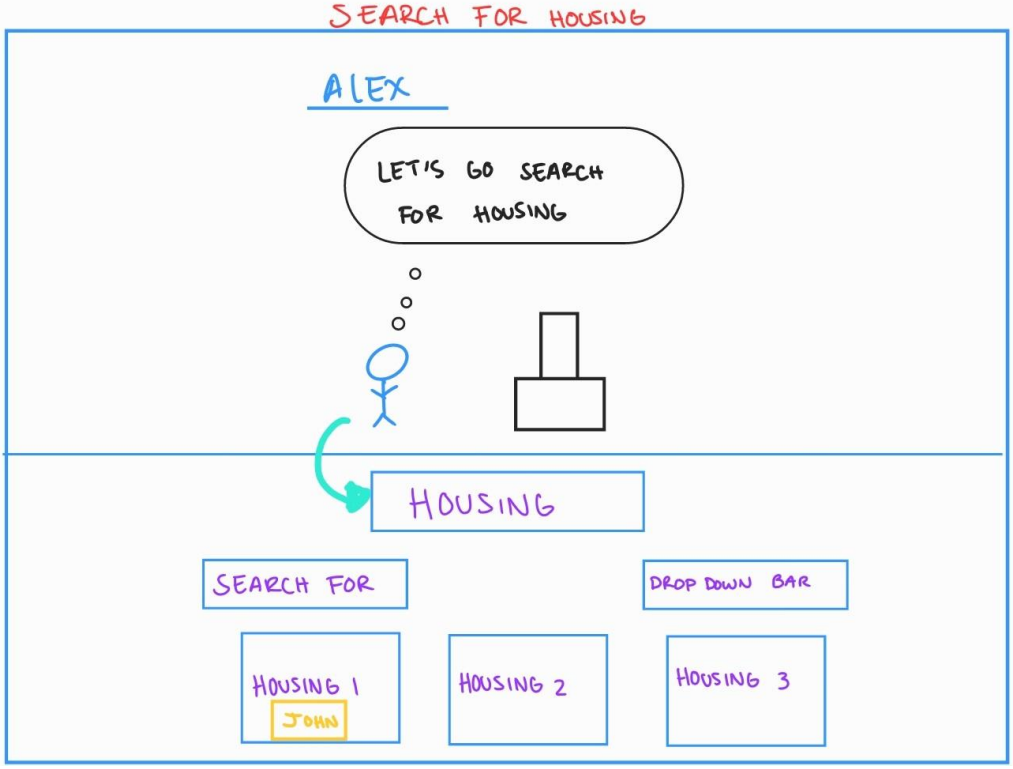
Case 8: A registered user reports a false housing listing
Actors: John (Registered User), Alex (Registered User), Alec (Super User)

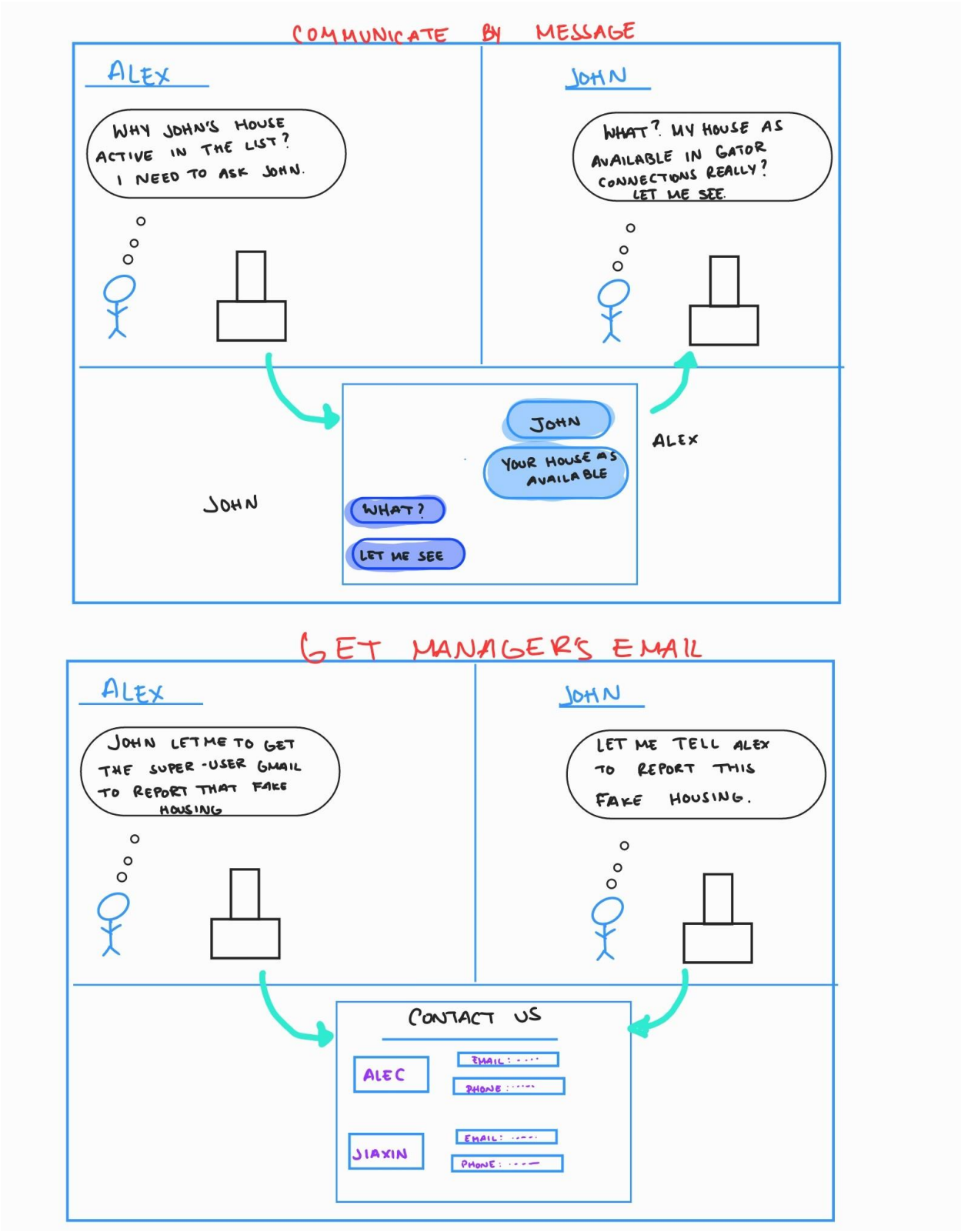


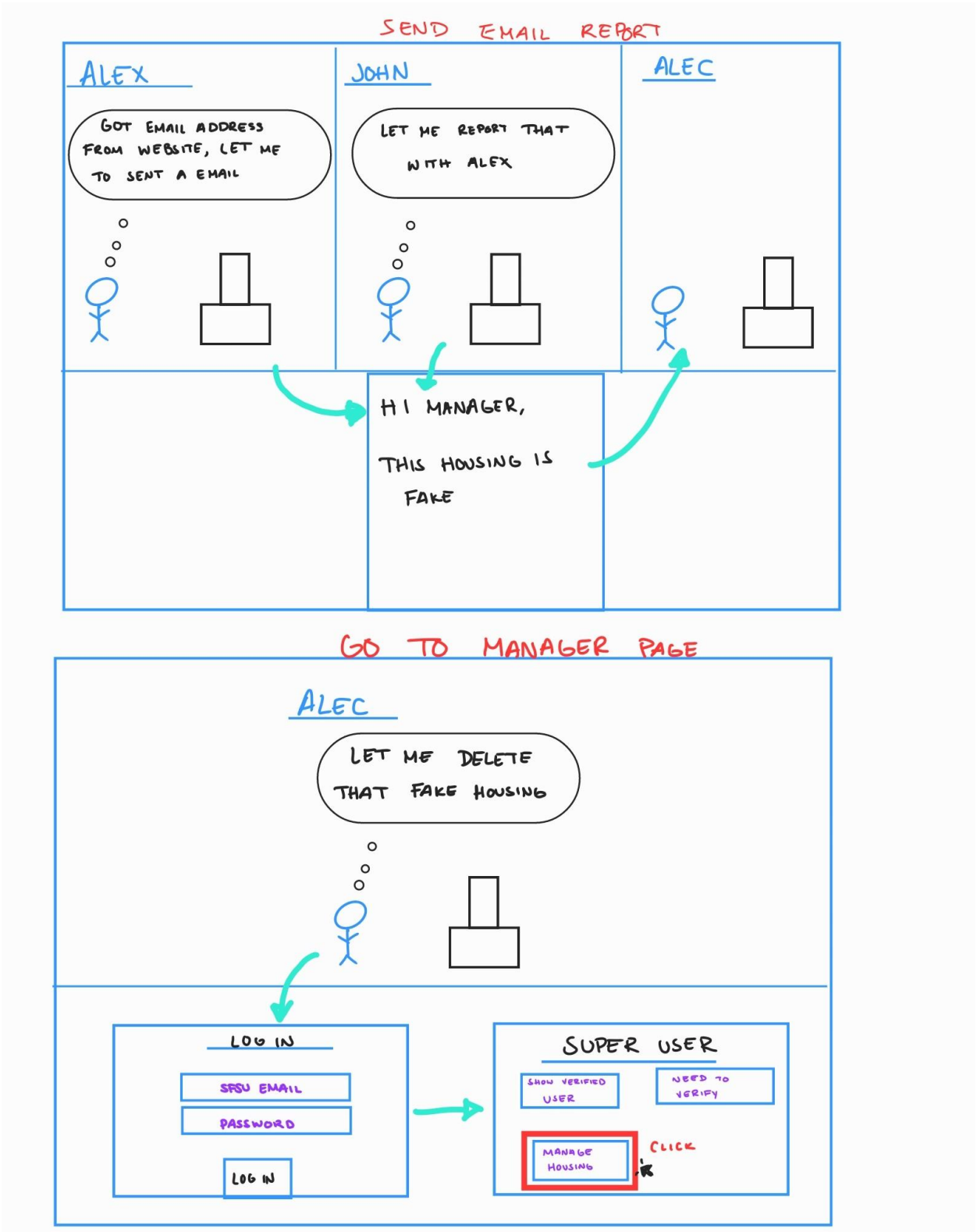
LOG IN

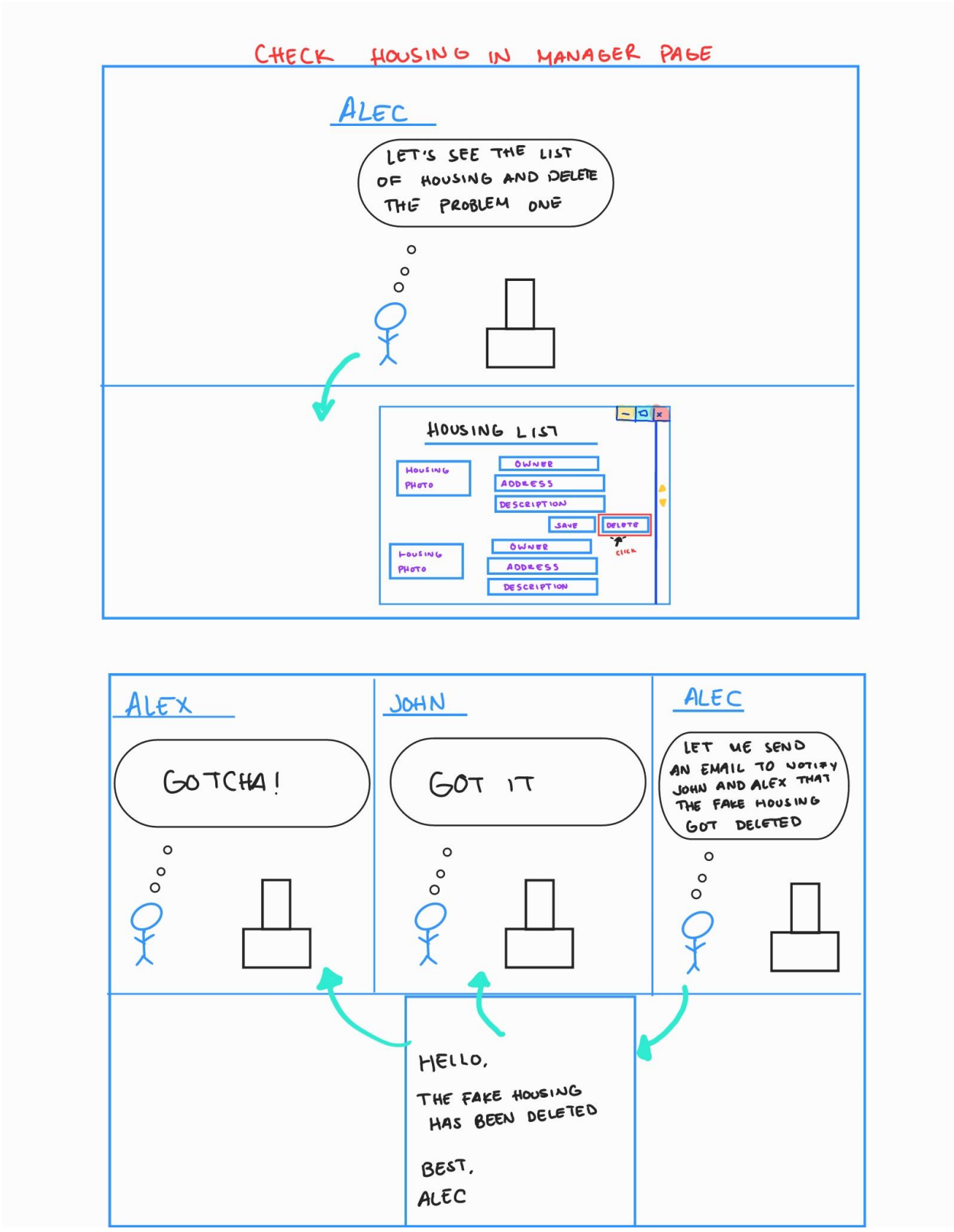


SEARCH FOR HOUSING




COMMUNICATE BY MESSAGE
ALEX
JOHN
WHY JOHN'S HOUSE ACTIVE IN THE LIST? I NEED TO ASK JOHN.
WHAT? MY HOUSE AS AVAILABLE IN GATOR CONNECTIONS REALLY? LET ME SEE.
JOHN
YOUR HOUSE AS AVAILABLE
WHAT?
LET ME SEE
ALEX
JOHN
GET MANAGER'S EMAIL
ALEX
JOHN
JOHN LETME TO GET THE SUPER-USER GMAIL TO REPORT THAT FAKE HOUSING
LET ME TELL ALEX TO REPORT THIS FAKE HOUSING.
CONTACT US
ALEC
EMAIL: ....
PHONE: .....
JIAXIN
EMAIL: ....
PHONE: ....


SEND EMAIL REPORT
ALEX
JOHN
ALEC
GOT EMAIL ADDRESS FROM WEBSITE, LET ME TO SENT A EMAIL
LET ME REPORT THAT WITH ALEX
HI MANAGER,
THIS HOUSING IS FAKE
GO TO MANAGER PAGE
ALEC
LET ME DELETE THAT FAKE HOUSING
LOG IN
SFSU EMAIL
PASSWORD
LOG IN
SUPER USER
SHOW VERIFIED USER
NEED TO VERIFY
MANAGE HOUSING
CLICK


CHECK HOUSING IN MANAGER PAGE
ALEC
LET'S SEE THE LIST OF HOUSING AND DELETE THE PROBLEM ONE
HOUSING LIST
HOUSING PHOTO
OWNER
ADDRESS
DESCRIPTION
SAVE
DELETE
CLICK
HOUSING PHOTO
OWNER
ADDRESS
DESCRIPTION
ALEX
GOTCHA!
JOHN
GOT IT
ALEC
LET ME SEND AN EMAIL TO NOTIFY JOHN AND ALEX THAT THE FAKE HOUSING GOT DELETED
HELLO,
THE FAKE HOUSING HAS BEEN DELETED
BEST,
ALEC

High Level Database Architecture and Organization

Business Rules

1. Registered User
    a. A registered user shall be able to create zero or many housing listing requests.
    b. A registered user shall have zero or many housing listing request notifications.
    c. A registered user shall be able to fill out one housing listing form for one housing listing request.
2. Housing Listing Request
    a. A housing listing request shall be created by one and only one registered user.
    b. A housing listing request shall be for one housing listing post.
    c. A housing listing request shall create one housing listing request notification.
    d. A housing listing request shall have one housing listing form.
    e. A housing listing request shall have a date of when the request was made.
3. Housing Listing Request Notification
    a. A housing listing request notification shall have one housing listing form.
    b. A housing listing request notification shall notify one and only one registered user.
    c. A housing listing request notification shall have one and only one housing listing request.
    d. A housing listing request notification shall have one and only one unique notification id.
    e. A housing listing request notification shall have a date of when the notification was created.
4. Housing Listing Post
    a. A housing listing post shall be posted by one and only one registered user.
    b. A housing listing post shall have a unique housing id.
    c. A housing listing post shall have a title.
    d. A housing listing post shall have an item description.
    e. A housing listing post shall have a price.
    f. A housing listing post shall have many images.
    g. A housing listing post shall be able to be removed by one and only one registered user that posted the housing listing.
    h. A housing listing post shall be requested by one or many registered users.
    i. A housing listing post shall have one housing listing form.

5. Housing Listing Form
    a. A housing listing form shall be for one housing listing post.
    b. A housing listing form shall be filled out by one and only one registered user.
    c. A housing listing form shall have the registered user's full name.
    d. A housing listing form shall have an about me about the registered user.
    e. A housing listing form shall have an expected day they want to move in.
6. Image
    a. An image shall belong to one and only one post.
    b. An image shall have a file path to the image.

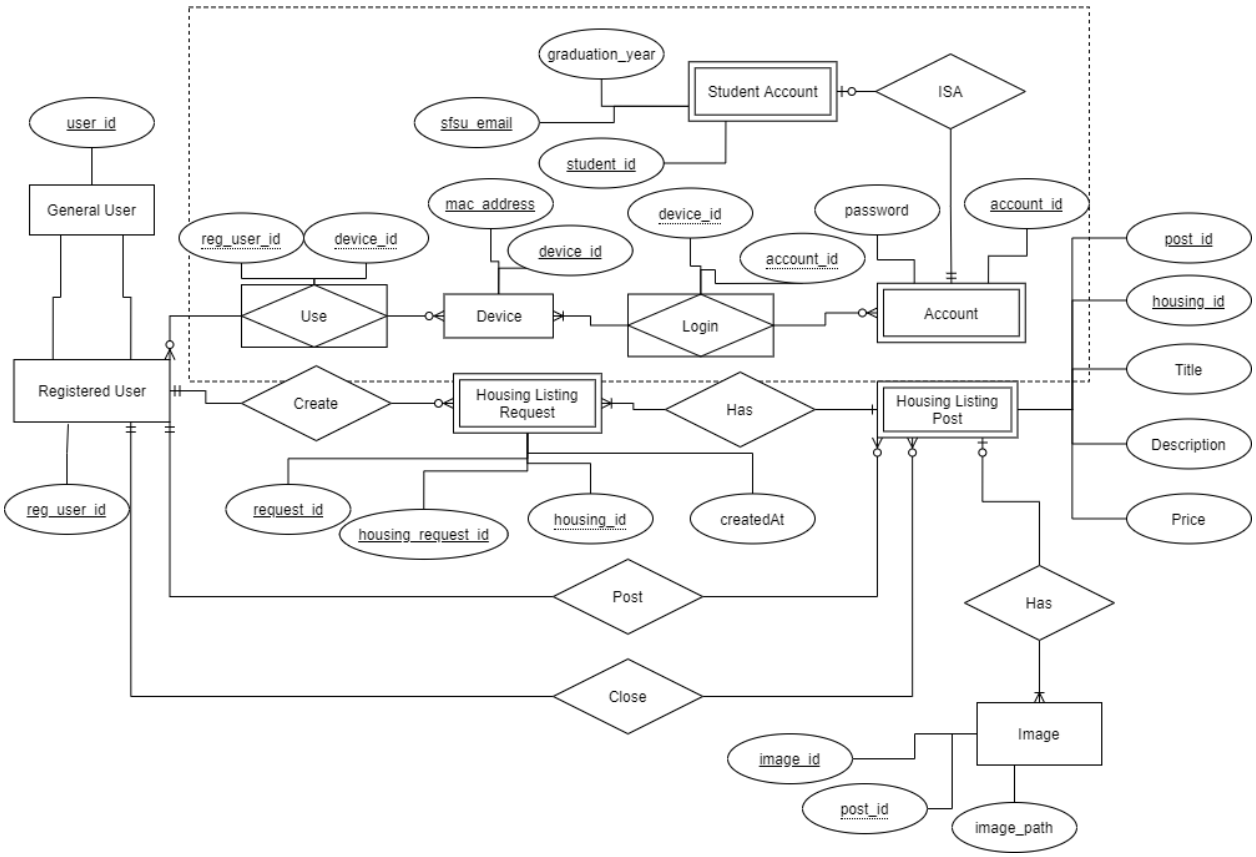
Entity, Attribute, Relationship, Domain Descriptions

1. General User (Strong)
    a. user_id: key, numeric
2. Registered User (Strong)
    a. reg_user_id: key, numeric
3. Image (Strong)
    a. image_id: key, numeric
    b. post_id: key, numeric
    c. image_path: alphanumeric
4. Registered User Device (Weak)
    a. reg_user_device_id: key, numeric
    b. reg_user_id: key, numeric
    c. device_id: key, numeric
    d. createdAt: DateTime
5. Device Session (Weak)
    a. device_session_id: key, numeric
    b. device_id: key, numeric
    c. account_id: key, numeric
    d. createdAt: DateTime
6. Account (Weak)
    a. account_id: key, numeric
    b. full_name: composite, alphanumeric
    c. password: alphanumeric
7. Student Account (Weak)
    a. student_id: key, numeric
    b. sfsu_email: key, numeric
    c. graduation_year: numeric
8. Housing Listing Request (Weak)
    a. housing_request_id: key, numeric
    b. request_id: key, numeric
    c. housing_id: key, numeric
    d. createdAt: DateTime
9. Housing Listing Post (Weak)
    a. post_id, key, numeric
    b. housing_id: key, numeric
    c. title: alphanumeric
    d. description: alphanumeric
    e. price: numeric

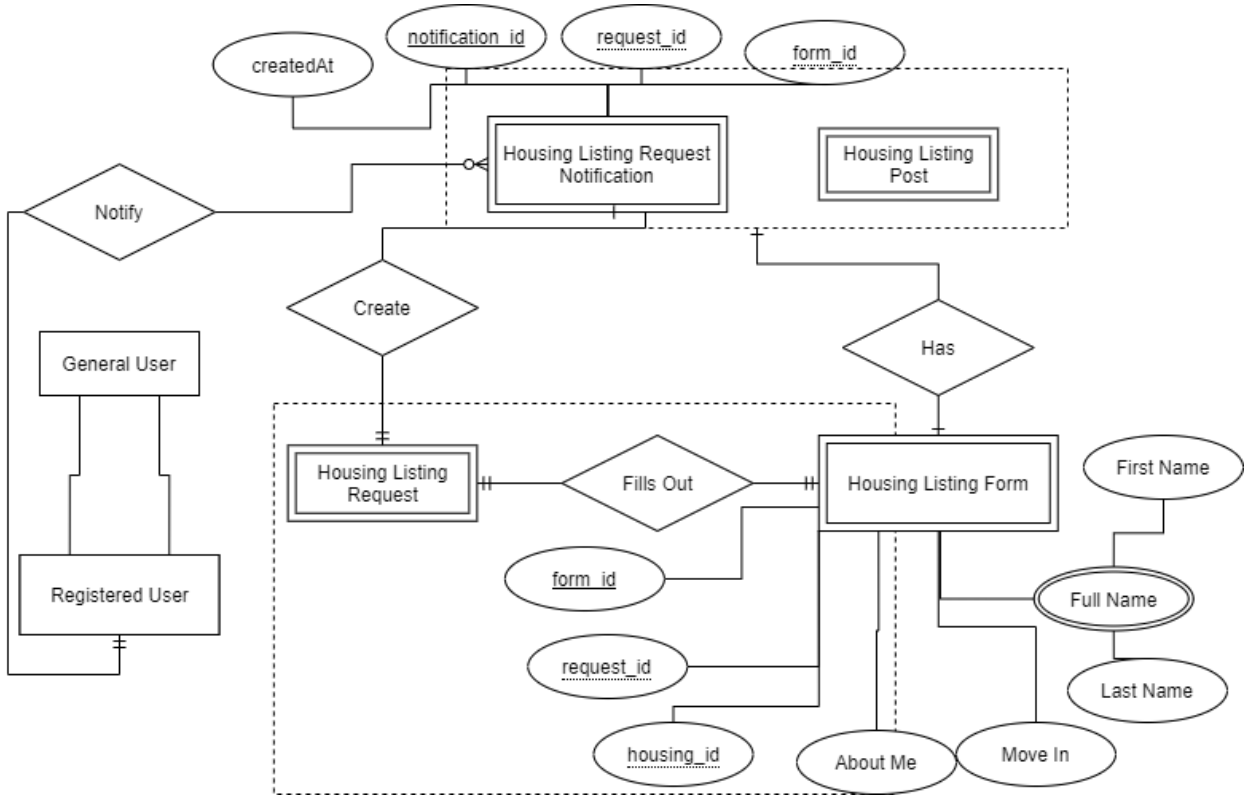
10. Housing Listing Form (Weak)
    a. form_id, key, numeric
    b. request_id, key, numeric
    c. housing_id: key, numeric
    d. full_name: composite, alphanumeric
    e. about_me: alphanumeric
    f. move_in: Date
11. Housing Listing Request Notification (Weak)
    a. notification_id: key, numeric
    b. form_id: key, numeric
    c. request_id: key, numeric
    d. createdAt: DateTime

Entity Relationship Diagram (ERD)

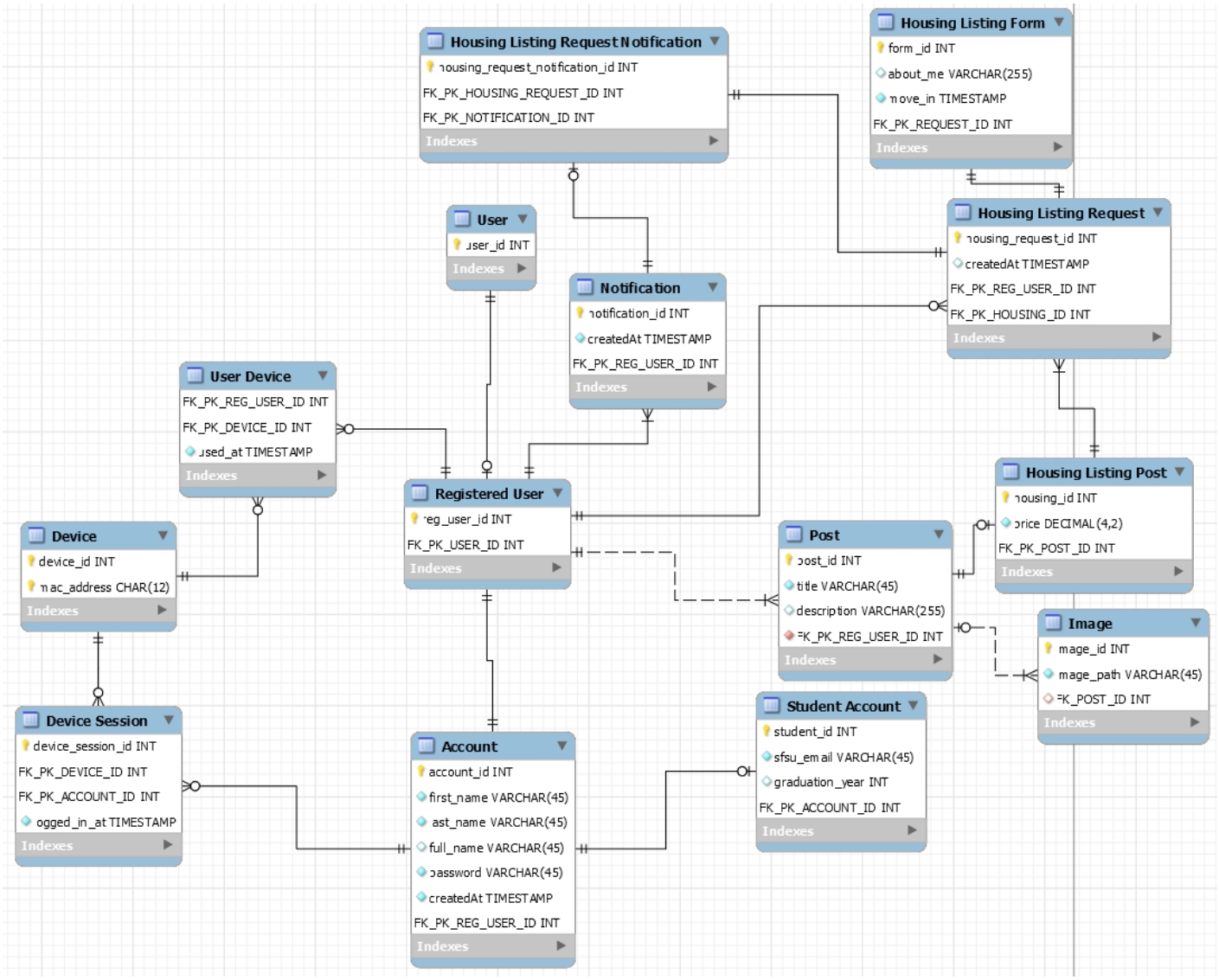
Shows login portion of ERD and relationships between Registered User, Housing Listing Request, and Housing Listing Post



Shows notification portion of ERD and relationships between notifications, Housing Listing Request, Housing Listing Form, and Housing Listing Post.



Database Model/Enhanced Entity Relationship (EER)



Database Management System (DBMS)

The DBMS that we are using to create the database is MySQL because we are creating a web application that mainly only deals in data transactions and simple queries. Another option we were considering was PostgreSQL. However, we concluded that MySQL was suitable for our application in the long run, and PostgreSQL, even though it supports more complex features, does not provide any significant advantages or extra utility for our web application needs.

Media Storage

Images will be kept in the file system in the EC2 instance because storing the images as blobs in our database does not provide any significant performance advantages. On top of this, storing the images in the EC2 instance keeps our database size small.

Search/Filter Architecture and Implementation

For our basic search functionality, we will be taking advantage of MySQL's InnoDB engine Full Text Search feature. In our database tables such as items, housing, restaurants, etc., we will create full text indexes based off of the columns in the table we want searched. Full text search allows us to search multiple columns in tables for relevant text strings due to word stemming which means our search functionality will be flexible in the results it finds. Also, we can have MySQL automatically sort search results by most relevant text-wise, which SQL's LIKE operator does not do. On top of this, based on our research, Full Text Search, on average, is faster than using SQL's LIKE operator. Therefore, our main search implementation method will be using MySQL's InnoDB engine Full Text Search. However, we will still look into SQL's LIKE operator and compare the performance and results of individual queries. Therefore, some sections of our search implementation will be decided on a case to case basis. We will create full text indexes for the combinations of table columns that we need. These full text indexes also have the added benefit of taking care of most of our filtering.

High Level APIs and Main Algorithms

AUTHENTICATION

We are creating our own authentication API instead of using similar APIs such as Django's built-in authentication API. We chose to create our own API because from our research, Django's authentication API only supports one type of user with one type of account. However, for our application, we have many different types of accounts such as Student, Admin, and Super User. This means Django's API will not be compatible with our application when it comes to dealing with user permissions, creating/registering different types of accounts, and logging into different types of accounts. By creating our own authentication API, we will be able to support multiple types of accounts and deal with user permissions as well. Our authentication API will handle registering new users by checking if users have a SFSU email. Login will be done by searching for the email that the user enters and comparing the password that the user enters with the encrypted password stored in our database.

1) **Password Encryption:** We will be using the popular password encryption algorithm, BCrypt to encrypt passwords in our database. We chose to use BCrypt instead of creating our own password encryption algorithm because the encryption is much more secure than what we can provide in our current timeframe. On top of this, the development costs of creating a new password encryption algorithm from scratch would outweigh the very minimal benefits we might get from creating our own algorithm. BCrypt's algorithm works by adding randomized data to the end of a given password, and encrypting the combined result many times.

2) **Email Verification:** We will be creating our own email verification API because all of Django's premade email verification modules use Django's authentication user model. Because we will be creating our own authentication API since Django's authentication user model is not compatible with our application, we must also create our own email verification API. However, we will be using Django's built-in email backend as the foundation for our email verification API. When a user creates an account, the account will be created in our database, but this account will have an extra piece of information indicating whether or not the user has verified his/her account and a unique code. Also, a verification email will be sent to the user's provided email. This email will contain a link with the unique code appended to the end of the link and when the user clicks the link, his/her account will automatically be verified in our database.

3) **Email Verification Link Token Generator:** We will be using Python's built-in secrets library to create URL safe tokens that will be appended to the unique link found in the verification email. Python's secrets URL safe tokens algorithm works by taking random

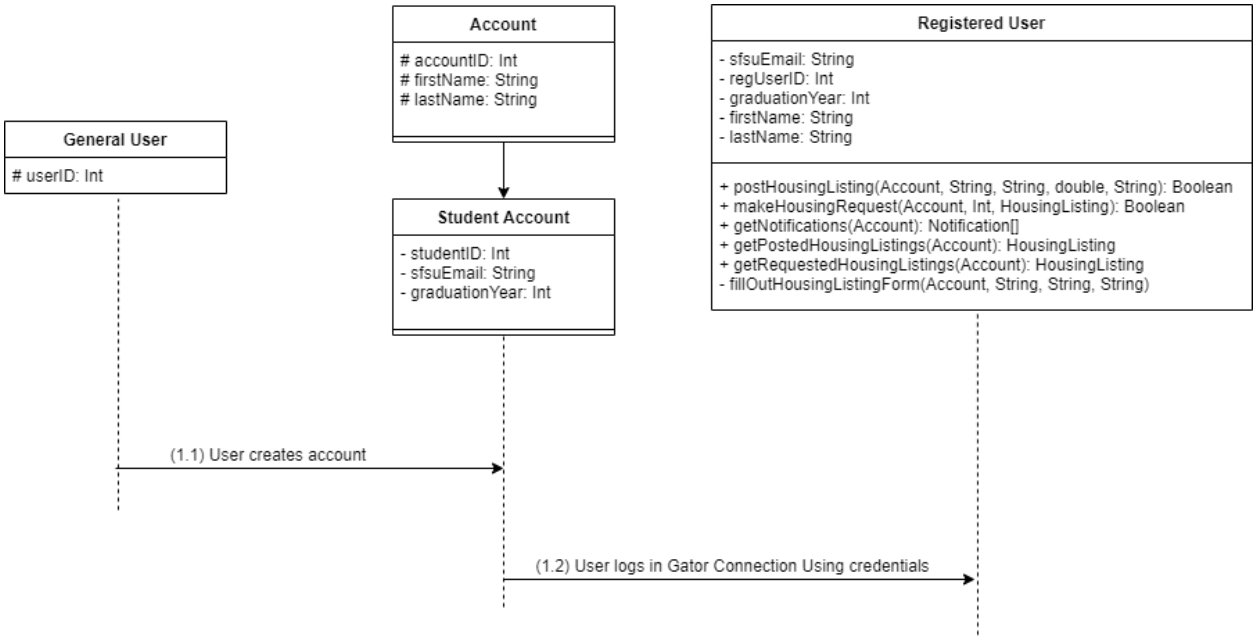
data and encoding it. Then the algorithm returns this encoded random data as the unique token.

NOTIFICATIONS

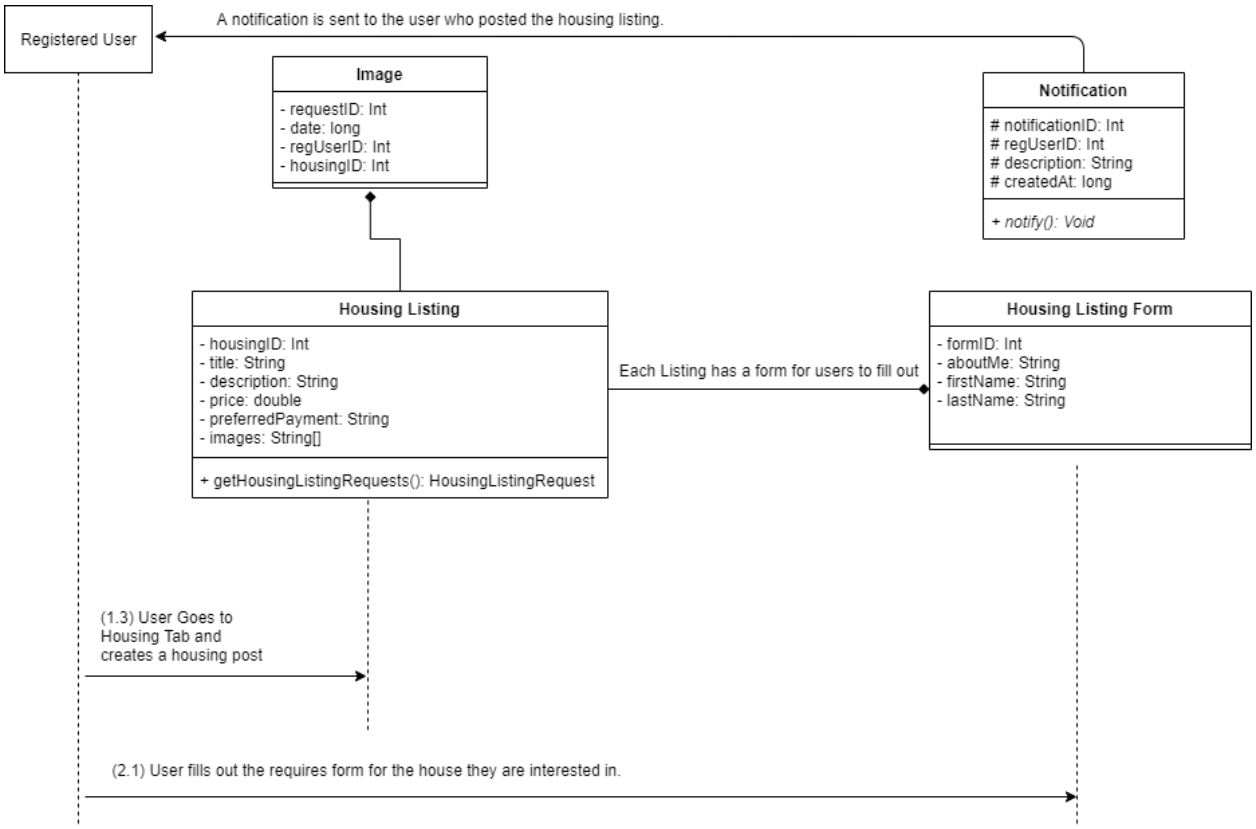
We will be creating our own notification API because we could not find an equivalent Django notification module. However, similar to our verification email API, we will be using Django's email backend as the foundation of our notification API. Our API will be designed so that whenever a user does an action that should notify others such as closing a posting or sending a buy request, we will create notifications for each user that must be notified in the database. Also, we will send notification emails of the event that occurred that created the notification. Because this email notification API is similar to the email verification API we will be creating, creating the notification API will not affect the development cost of our product greatly.

High Level UML Diagrams

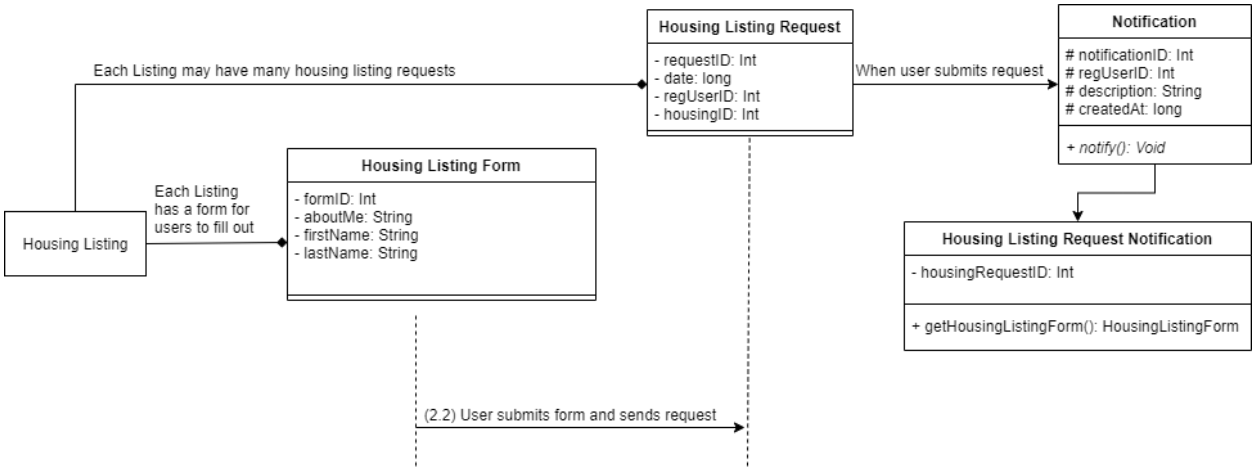
Association and Relationships between General User, Registered User, Account, and Student Account.



Associations and Relationships between Registered User, which was already defined above, Housing Listing, Image, Housing Listing Form, and Notification.

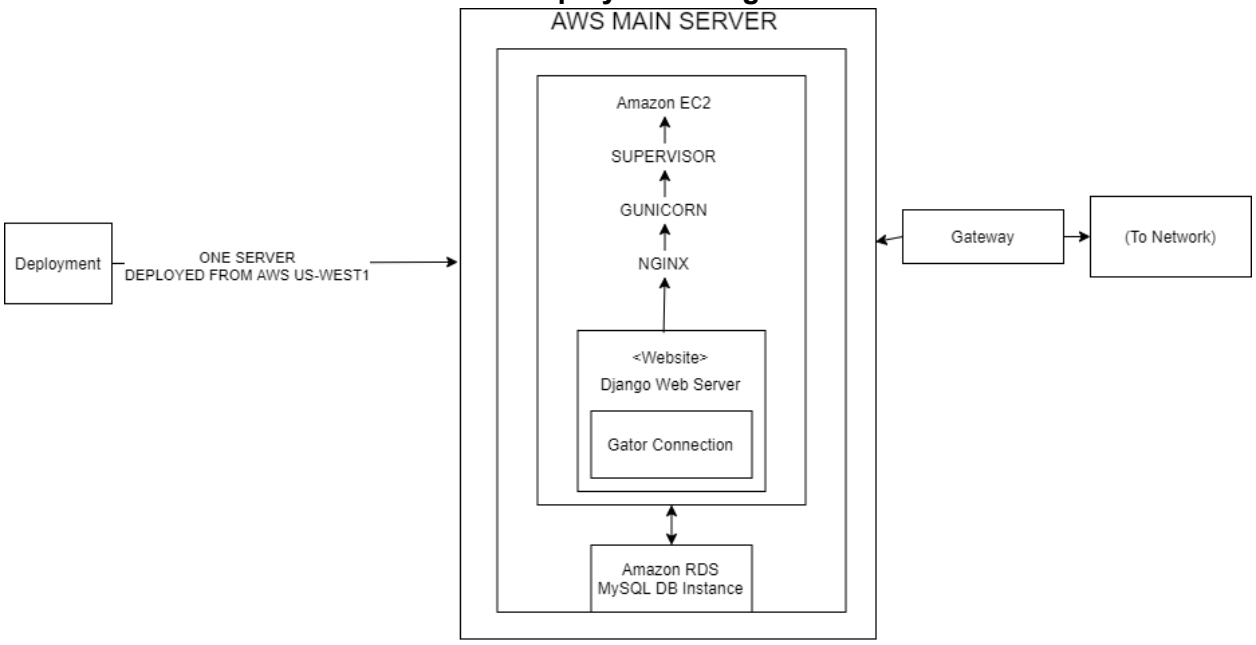


Associations and relationships between Housing Listing, which was already defined above, Housing Listing Form, Housing Listing Request, Notification, and Housing Listing Request Notification.

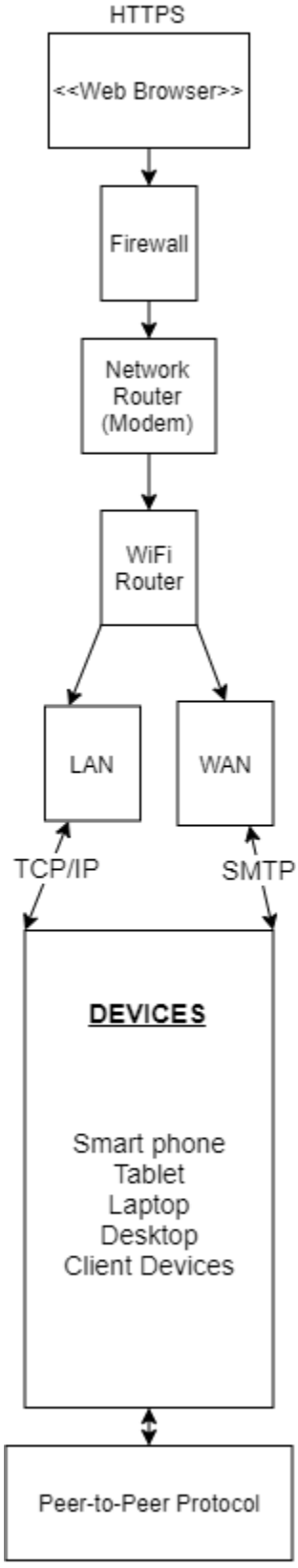

Housing Listing Request
- requestID: Int
- date: long
- regUserID: Int
- housingID: Int
Notification
notificationID: Int
regUserID: Int
description: String
createdAt: long
+ notify(): Void
Each Listing may have many housing listing requests
When user submits request
Housing Listing Form
- formID: Int
- aboutMe: String
- firstName: String
- lastName: String
Each Listing has a form for users to fill out
Housing Listing
Housing Listing Request Notification
- housingRequestID: Int
+ getHousingListingForm(): HousingListingForm
(2.2) User submits form and sends request

High Level Application Network and Deployment Diagrams

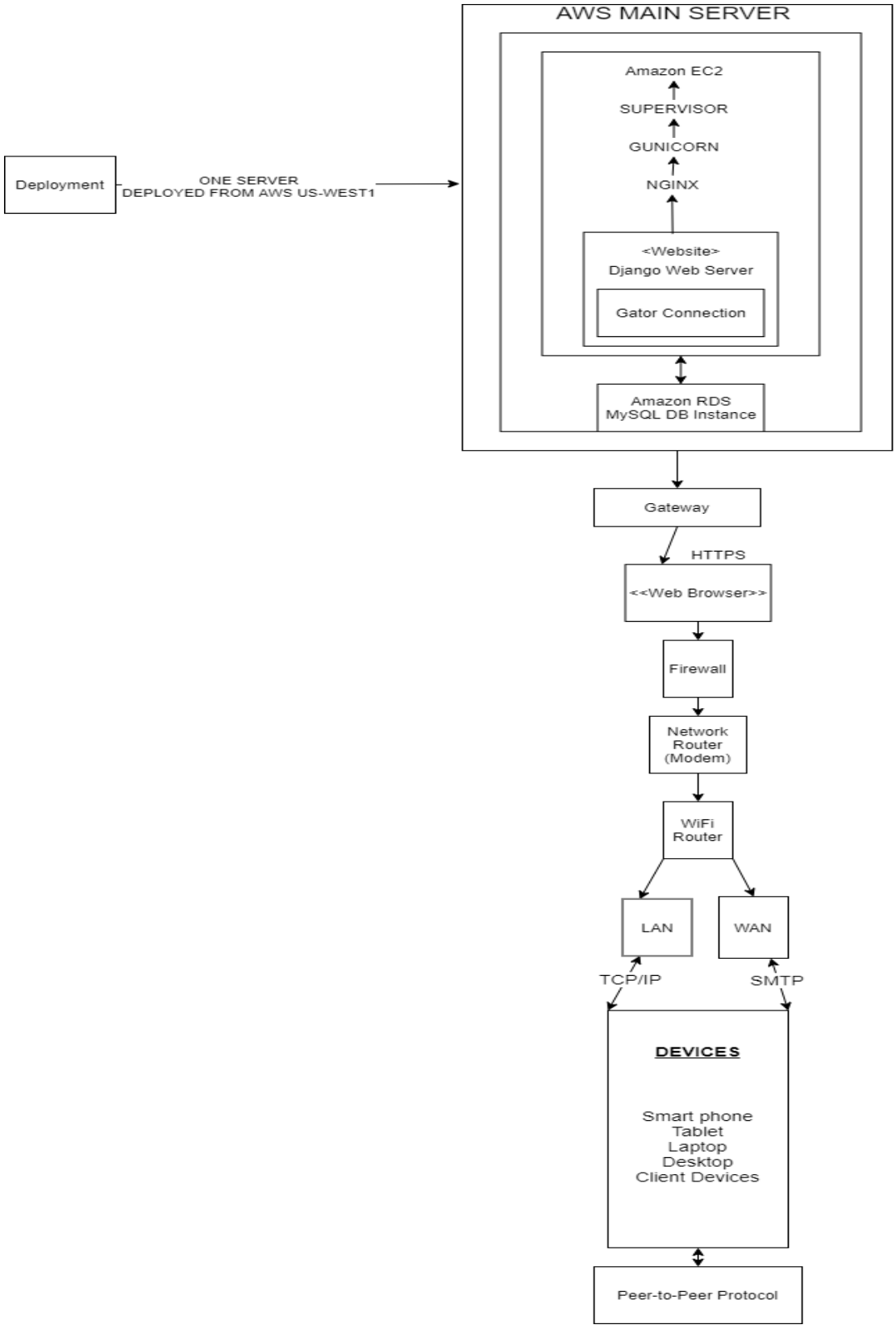
Deployment Diagram:



Application Network Diagram:



Application Network Diagram with Application Network Diagram:



Project Risks

Schedule risk

Due to everyone's schedules, it's hard to get a meeting time for everyone to be present. However, with having a trello account set up for project management for all team members, it has been easier to convey task distribution. Furthermore, we have been able to meet by separating our meetings into front-end and back-end meetings, and recapping the team members who can not attend.

Skill risk

Our team has development experience, however, none of us had experience designing a website. Furthermore, while some of us were familiar with the django web framework, it was unfamiliar to most of the team, with most of us having to learn it on the fly. In addition, Python is also unfamiliar to some of the team, so some of us are still in the process of understanding it. Lastly, most of us have never done front end design at this large of a scale, so everyone on the team is learning it as we go along as well.

Technical risk

One technical risk was associated with our github repo. What was at risk was our git branches, specifically with push/pull and merge conflicts. How we keep track of merge conflicts is with the multiple branches we have for different tasks and having a Github master who handles the merge conflicts. Another technical risk was the issue of our MySQL database because we wanted the team to connect to the database locally so we would all work with the same data. Connecting to the database was challenging for the team, but we had managed to all do it. However, one thing was that we had to do was make our database public, which could expose information. However, we fixed this problem by creating security rules that allowed only the team's and EC2 instance's IP addresses to access the database. One more technical risk was how everyone has different operating systems, such as Windows, Mac, etc. In the end, we decided to use Ubuntu for our project, which is generally easier for people with Apple products. For the team members who had a Windows Operating system, they had to learn Linux commands to get used to the environment for our project.

Teamwork risk

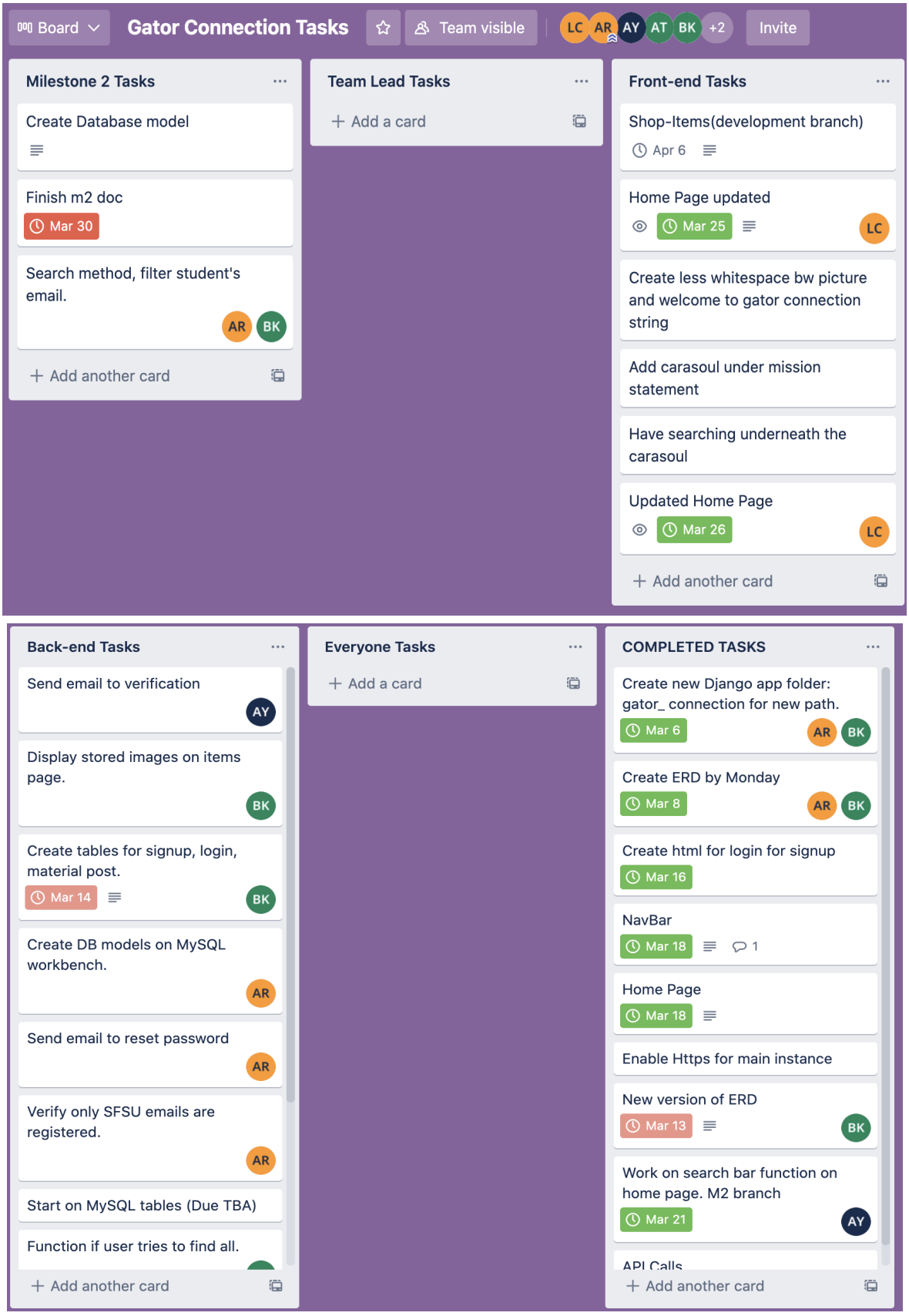
One major teamwork risk is having good communication between the backend and frontend teams due to the team's scheduling issues. However, with our project management being handled by Trello and the team lead posting updates on the discord channel and attending every meeting to be the middleman between the front and backend, everyone is up to date on what needs to be done and what each team is working on.

Legal/Content risk

Our legal risks may vary depending on the end product of the deployment. On our server, we will have control and ownership of images on the site. With that, one risk associated with us is that we may encounter people who upload images that do not belong to them. Another risk is that we will have external links to outside resources for services we do not provide. This means that if anything happens to the user after he/she clicks on an external link, we might be held responsible. To resolve these risks, when a user registers their account, they must agree to a terms and conditions form. These terms and conditions would ensure that everything they upload belongs to them and that the user understands that we are not responsible for anything that occurs after the user exits our site via any external links. We will also warn users when they click on an external link that they are leaving our site.

Project Management

In our first couple of team meetings for M2, we addressed the sections that we believed had to be done with everyone together such as setting the priorities of our functional requirements, enhancing our data definitions, and examples of how we wanted to format our UI mockups and storyboards. Afterwards, our next team meeting, we set up Trello as our main project management tool with tasks split up into frontend tasks, backend tasks, team tasks, and completed tasks. By keeping track with Trello, backend and frontend team members would create rough drafts of the sections assigned to them. Once those team members completed the rough drafts and notified the team on Trello and Discord, either the frontend lead or backend lead would give feedback depending on which team was in charge of which section and if needed, set up meetings with the team members that created the rough drafts and edited the sections together. Otherwise, the team members would incorporate the feedback and continue the cycle of getting feedback from a frontend lead or backend lead. Once the frontend lead or backend lead deemed the section was completed, the team lead did the final quality check. The team lead also made sure to check in and help with all sections that team members were working on and gave intermittent feedback as well. For future tasks, we will continue to use Trello and Discord to keep track of tasks and our progress. Most importantly, we will continue to continue the cycle we were practicing of creating rough drafts, receiving feedback from a team lead member, and incorporating the feedback.


Board
Gator Connection Tasks
Team visible
LC AR AY AT BK +2
Invite
Milestone 2 Tasks
Create Database model
Finish m2 doc
Mar 30
Search method, filter student's email.
AR BK
Add another card
Team Lead Tasks
Add a card
Front-end Tasks
Shop-Items(development branch)
Apr 6
Home Page updated
Mar 25
LC
Create less whitespace bw picture and welcome to gator connection string
Add carasoul under mission statement
Have searching underneath the carasoul
Updated Home Page
Mar 26
LC
Add another card
Back-end Tasks
Send email to verification
AY
Display stored images on items page.
BK
Create tables for signup, login, material post.
Mar 14
BK
Create DB models on MySQL workbench.
AR
Send email to reset password
AR
Verify only SFSU emails are registered.
AR
Start on MySQL tables (Due TBA)
Function if user tries to find all.
Add another card
Everyone Tasks
Add a card
COMPLETED TASKS
Create new Django app folder: gator_ connection for new path.
Mar 6
AR BK
Create ERD by Monday
Mar 8
AR BK
Create html for login for signup
Mar 16
NavBar
Mar 18
1
Home Page
Mar 18
Enable Https for main instance
New version of ERD
Mar 13
BK
Work on search bar function on home page. M2 branch
Mar 21
AY
API Calls
Add another card

List of Contributions

---

1) Data Definitions V2:
During a team meeting, the team looked at the data definitions we had, and decided on which terms we needed to expand upon. After deciding, Alec added the extra terms to explain certain subsections, with the team revising what should be said to explain them.

2) Functional Requirements V2:
During a team meeting, the team looked at the functional requirements we had. During this time, with suggestions from other members, namely Angelo and Benjamin, we were able group up our functional requirements into the specified three priority groups, with Priority 1 being the most important, to Priority 3 being opportunistic.

3) UI Mockups and StoryBoards:
During a team meeting, Alec tasked the group to do a rough draft of a UI mockup of the Use Case they had made. After receiving the rough draft, Alec looked over the styles everyone had submitted and decided to have the Front End members, namely Carmen and Bikram to revise everyone's rough draft into a final draft version.

4) High level database architecture and organization:
During a team meeting, Angelo started the rough draft of the entity relationship diagram (ERD) for our special feature. After feedback from Jiaxin and Benjamin, the ERD was revised to be more accurate of the special feature we had chosen for our project and to follow more closely along the lines of the business rules. From the business rules, Angelo was able to extract the entities and describe them at a deeper level. On the database model, Benjamin designed the database model that we would be using for the special feature based off of the ERD. On this part, we also decided that we would use MySQL for our Database Management System(DBMS) and our images would be kept in the file systems of our EC2 instance, with Benjamin filling this part out. Furthermore, on another team meeting, Benjamin showed the team the Full Text Search feature, which we would decide to use for our search/filter architecture.

5) High Level APIs and Main Algorithms:
Alec listed down certain API’s that the team was going to eventually use for the project, such as POST calls for logging in and registering, GET calls for getting user information, etc. After feedback from the team, the API’s were more fleshed out to describe the functionality.

6) High Level UML Diagrams:
Benjamin was tasked to complete the UML diagram for the special feature. After a meeting and some feedback for the first version from Angelo, Benjamin was able to revise the UML diagram to fit the standards of our special feature.

7) High Level Application Network and Deployment Diagrams:

Alec started a rough draft of the Application Network diagram. After a class lecture, the team, most notably Benjamin, helped revise the Application Network diagram to portray more info of what the professor had explained about the diagram. After clarification from the professor, Alec found out that the High Level Application Network and Deployment Diagrams could be combined into one, and revised it as such.

8) Identify actual key risks for your project at this time:

During a team meeting, the team helped suggest things that were considered as risks for the project. After jotting them down, Alec organized them into 5 sections, which were schedule, skill, technical, teamwork, and legal/content risks.

9) Project management:

During a team meeting, Angelo set up Trello and invited everyone on the team so we could start assigning tasks. On Trello, Alec delegated tasks to the team, such as having UI mockups done by a certain date.

10) Detailed list of contributions:

Alec filled out this part. After assigning the tasks to everyone, Alec looked over what everyone did to accurately fill out this section.

11) Milestone2 Editor:

After volunteering to edit the document, Lakshita organized the Milestone2 document to make it ready to be turned in.

Vertical Prototype Contributions

Backend:

For the backend portion of our vertical prototype, Angelo helped set up basic API features, such as registering and logging in for users. After a revision from Benjamin for the login portion, the registering and logging in features was complete. After the team learned how to use the login feature, which would allow us to restrict certain actions from someone if they were not logged in, Jiaxin was tasked to start implementing the search function. With that, Angelo expanded upon the search feature which would then allow users to search for a user based by their email. Alec then helped on the searching feature, so users would then be able to filter by sfsu email, and by the first or last name associated with that email. Benjamin then improved upon Alec's filtering feature by having the feature use Full Text Search to improve upon the searching feature. After improving upon the filtering feature, Benjamin then reorganized our backend code structure to make everything easier to find. Angelo then set up the DB table "student" on MySQL workbench. When a user would register, their info would be stored there. Furthermore, on the registering and logging in functions, Alec added error handling to make sure that the program would continue in case of an error.

Frontend:

The front end team was in charge of designing the vertical requirement home page. Lakshita, Carmen, and Bikram added features such as having a mission statement at the top of the screen, and a carousel slide at the bottom to show SFSU pictures. Other features implemented were having certain features only appear if a user is logged in, having messages appear to notify a user that once they have completed a certain action, such as logging in or registering. Furthermore, these messages have replies for actions that fail or succeed. In the case of an action succeeding, they will be greeted with a green box saying something like "Thank you for registering". On the flip side, if they fail, they will be greeted with something like "Please try again". Another feature added was structuring how the data coming from the database would look like on the page. On the navbar features, Alec designed the rough prototype, then had Bikram implement features on the navbar such as the navbar would highlight a specific section when it was on that page. Furthermore, Carmen designed our logo which would show on the top left of the page. On the design of the login and registering boxes, Alec designed the rough prototype for them, and after feedback from the team, he was able to revise it to make it look much cleaner than it was before.