SW Engineering CSC 648/848 Section 02 Fall 2017

Dream Home

Team 05

Milestone 2:

Detailed Requirements, Specs, Architecture, UI mock-ups, and a vertical SW prototype

- Sophie Tait (stait@mail.sfsu.edu)
- Bravolly Pich
- James Hinds
- Supritha Amudhu
- Saengduean (Em) Calderaz
- Brendan Kelly
- Steve Cardenas

History

Date	Description
10/26/2017	Created first draft
11/02/2017	Revised draft, froze draft for grading

1. Data Definitions

- **Unregistered User**: A user who does not have an account or is not currently logged in. They use the website to browse listings and to create an account.
- **Registered User**: A user who has an account and is currently logged in. They use the website to browse listings and contact sellers.
- **Administrator**: A user who has elevated privileges including access to the website database. They use the website to perform administrative tasks and to moderate listings and users.
- **Seller**: A user who is a real estate agent. They have an account and are logged in. They use the website to browse listings, post listings, and receive messages from registered users.
- **Listing**: The entity that represents a property for sale. Listings contain information on location, property amenities, number of bedrooms, number of bathrooms, price, and seller. Listings are posted by sellers and available to users. Listings may be moderated at the discretion of the administrator.
- **User Dashboard**: A page that includes all listings that the user has contacted the seller about.
- **Seller Dashboard**: A page that includes all listings that the user has posted. Seller Dashboard contains all users that have contacted the seller about a particular listing.

2. Functional Requirements

Priority 1

1. **Unregistered User**
    a. Shall be able to browse through the listings
    b. Shall be able to view house listing details
    c. Shall be able to search listing
    d. Shall be able to sort listing
    e. Shall be able to register an account
2. **Registered User**
    a. Shall be able to browse through the listings
    b. Shall be able to view house listing details
    c. Shall be able to search listing
    d. Shall be able to sort listing
    e. Shall be able to login to their account
    f. Shall be able to log out of their account
    g. Shall be able to contact seller
3. **Seller**
    a. Shall be able to browse through the listings
    b. Shall be able to view house listing details
    c. Shall be able to search listing
    d. Shall be able to sort listing
    e. Shall be able to login to their account
    f. Shall be able to log out of their account
    g. Shall be able to view their own Seller Dashboard
    h. Seller Dashboard shall show their own listings of houses that they had created
        i. Shall be able to add a house listing
        ii. Shall be able to remove their house listing
        iii. Shall be able to edit their house listing
    i. Seller Dashboard shall be able to view Registered Users that had contacted them
        i. Shall be able to contact these Registered Users
4. **Administrator**
    a. Shall be able to access the database
    b. Shall be able to remove a house listing
    c. Shall be able to ban an users for misappropriate use

Priority 2

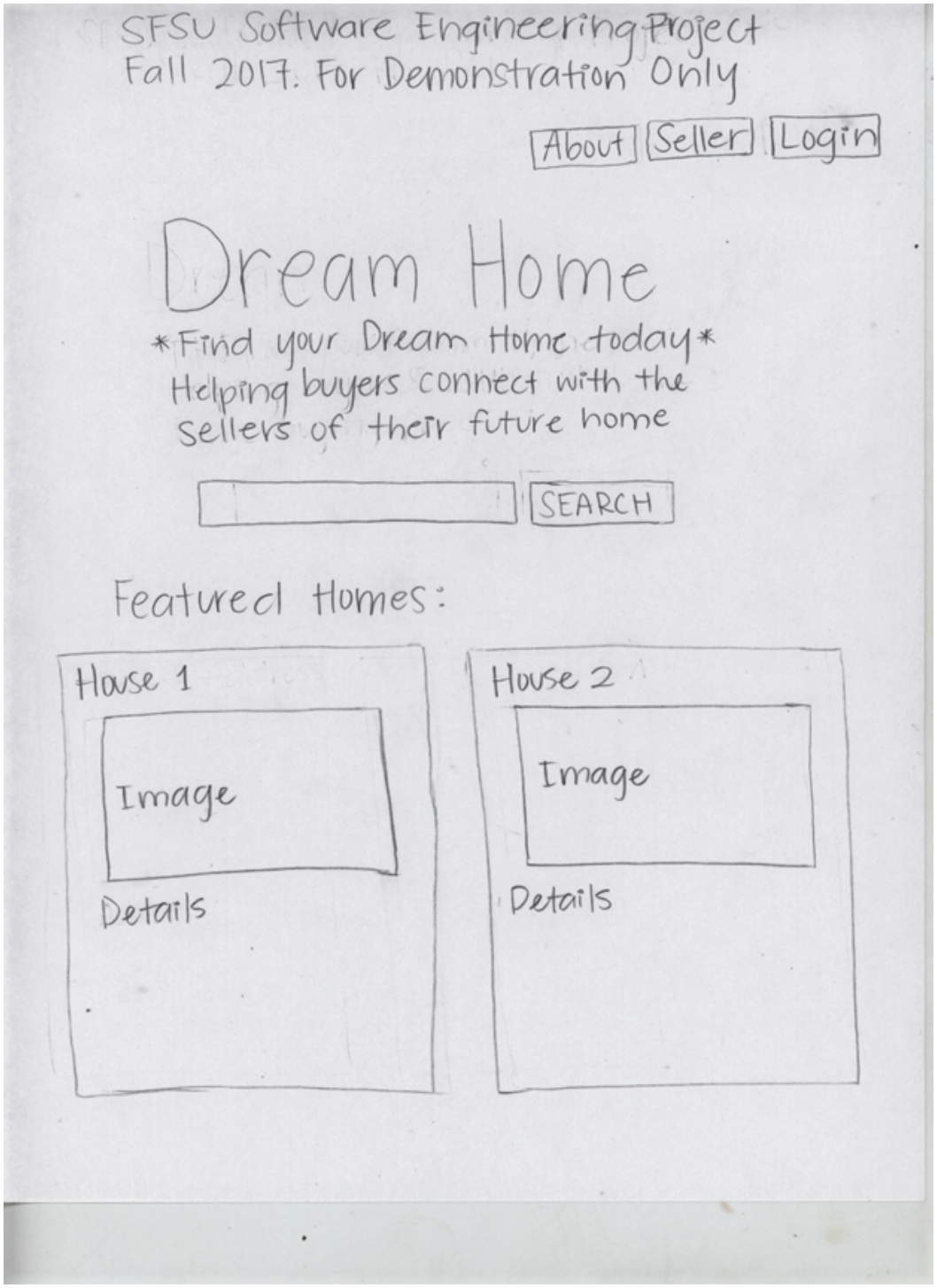
1. **Unregistered User**
2. **Registered User**
    a. Shall be able to view their own Registered User Dashboard
    b. Registered User Dashboard shall show sellers that they have been in contact with
    c. Registered User Dashboard shall show house listing that they are interested in
    d. Registered User Dashboard shall be able to remove house listing that they are no longer interested in
3. **Seller**
4. **Administrator**
    a. Shall be able to view website analytics

Priority 3

1. **Unregistered User**
2. **Registered User**
3. **Seller**
4. **Administrator**

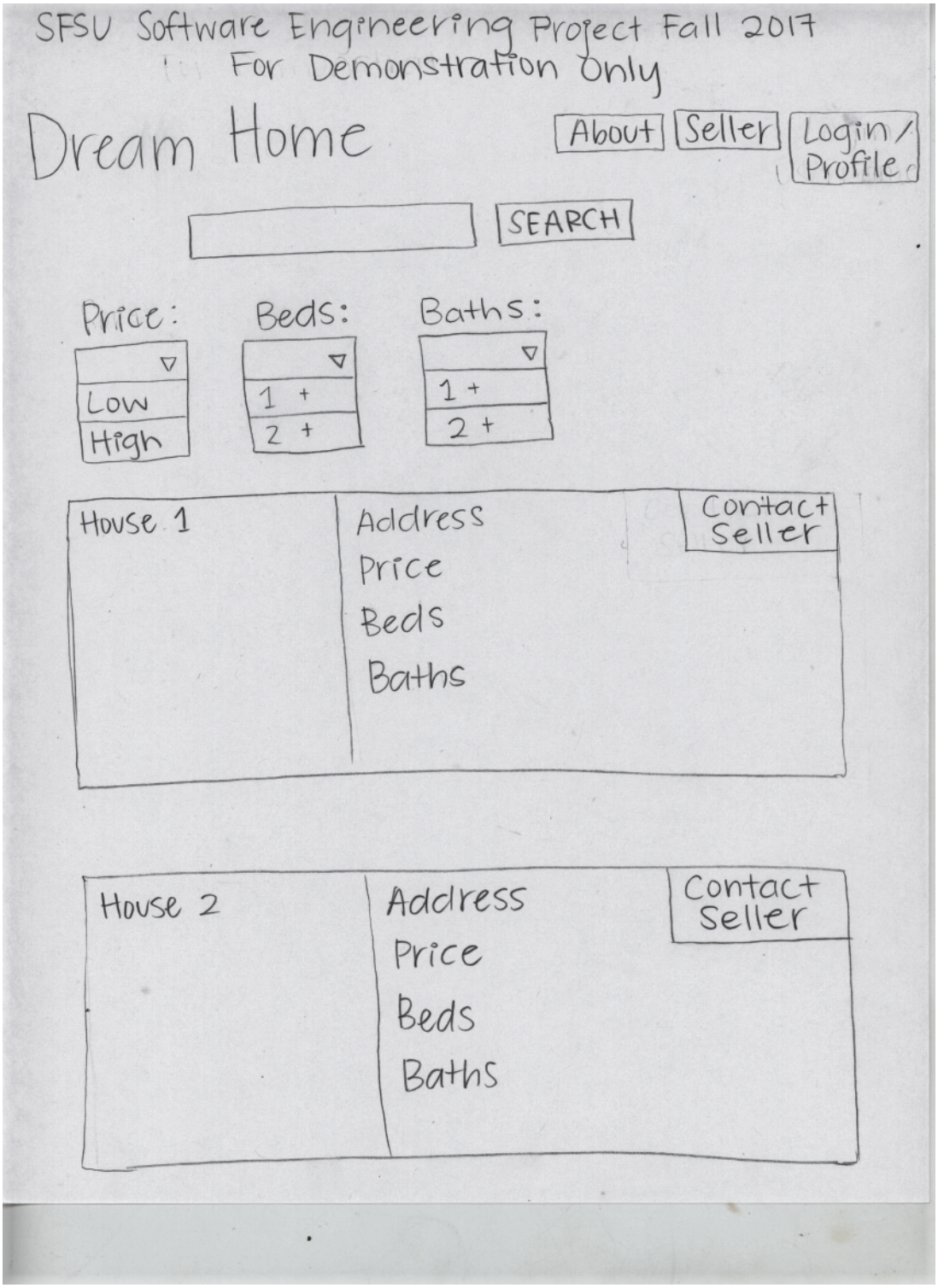
3. UI Mockups and Storyboards

Homepage



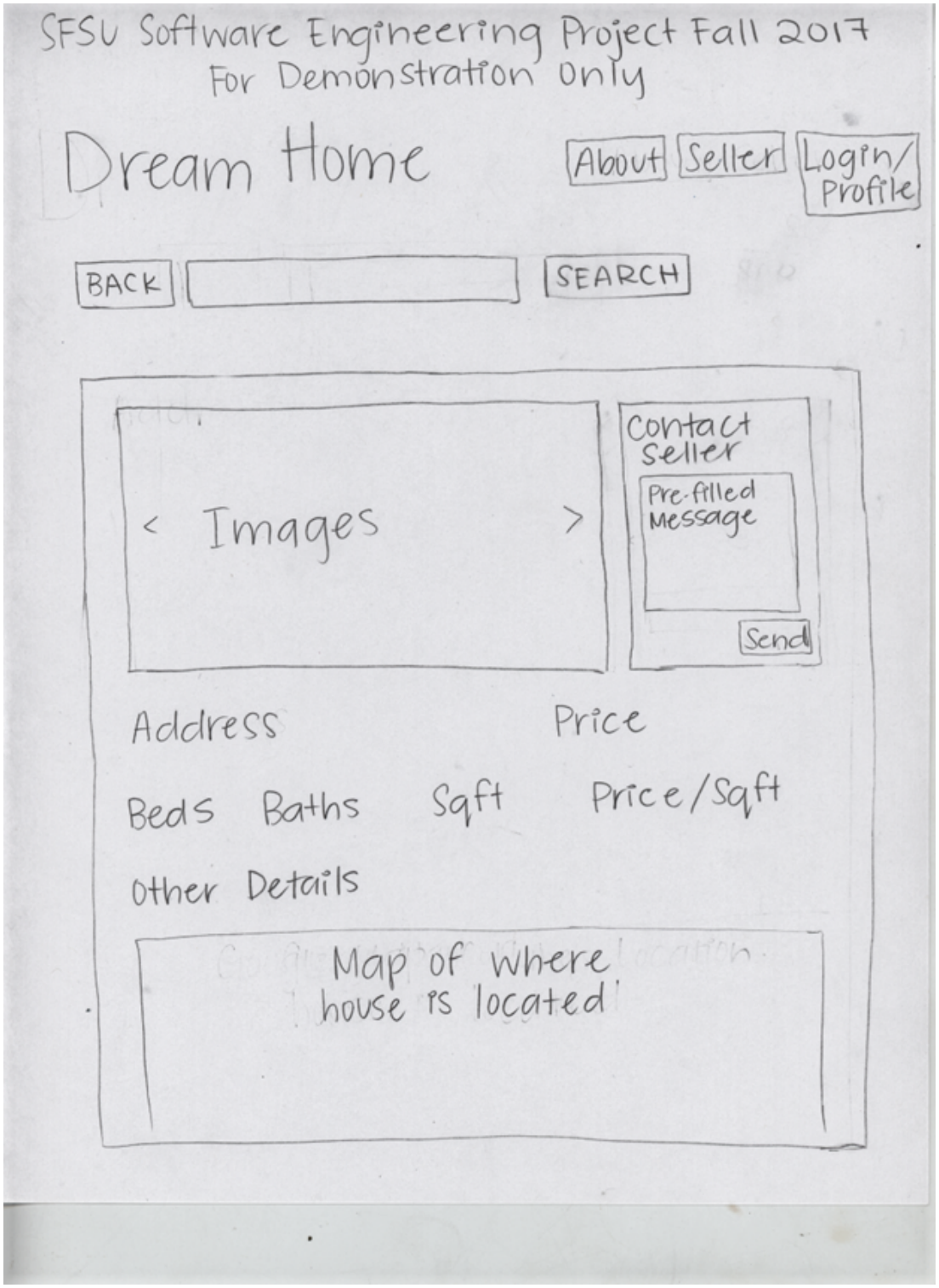
Search Results

If no results found, notify user and display other home suggestions

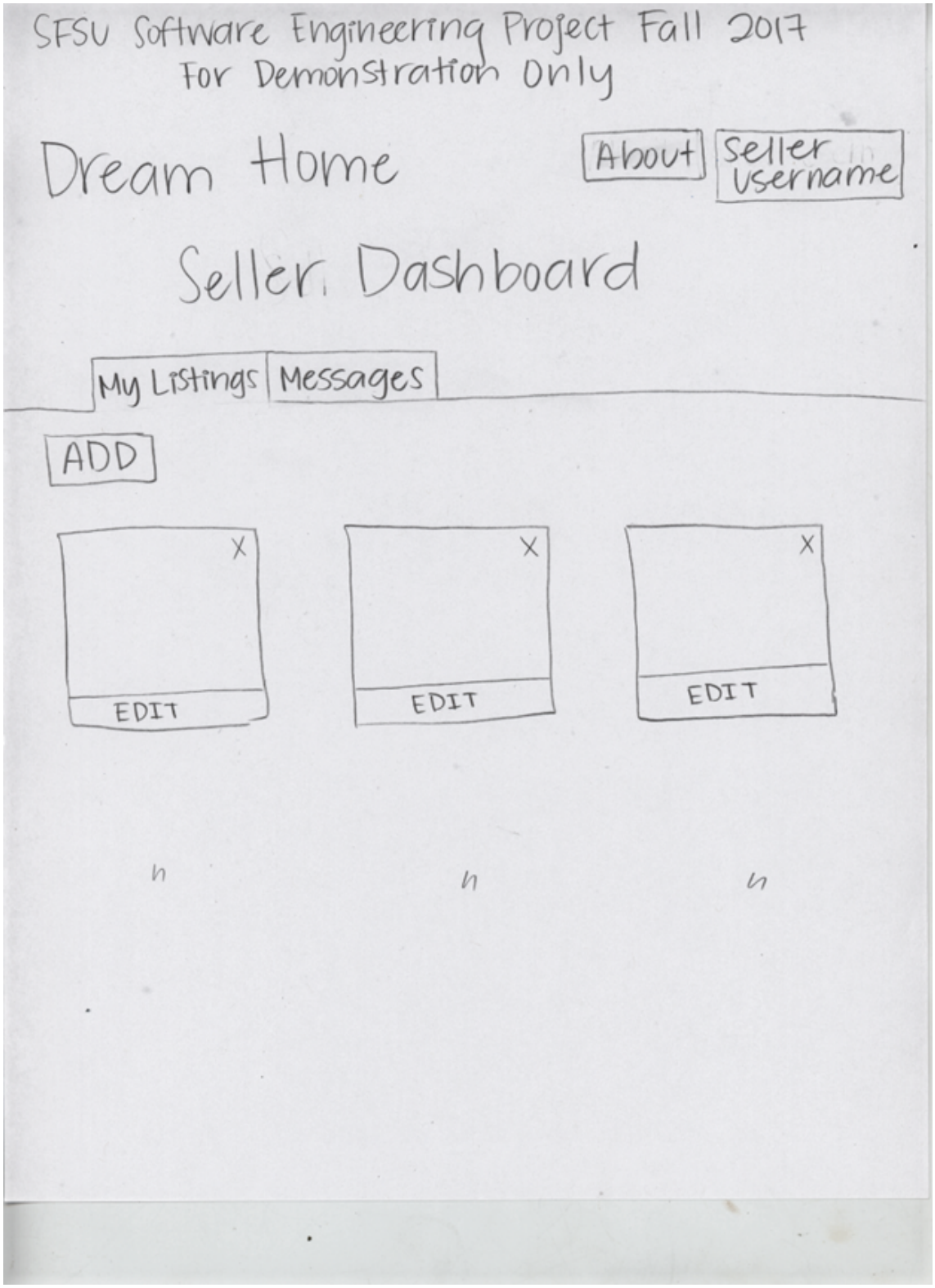


Listing of House

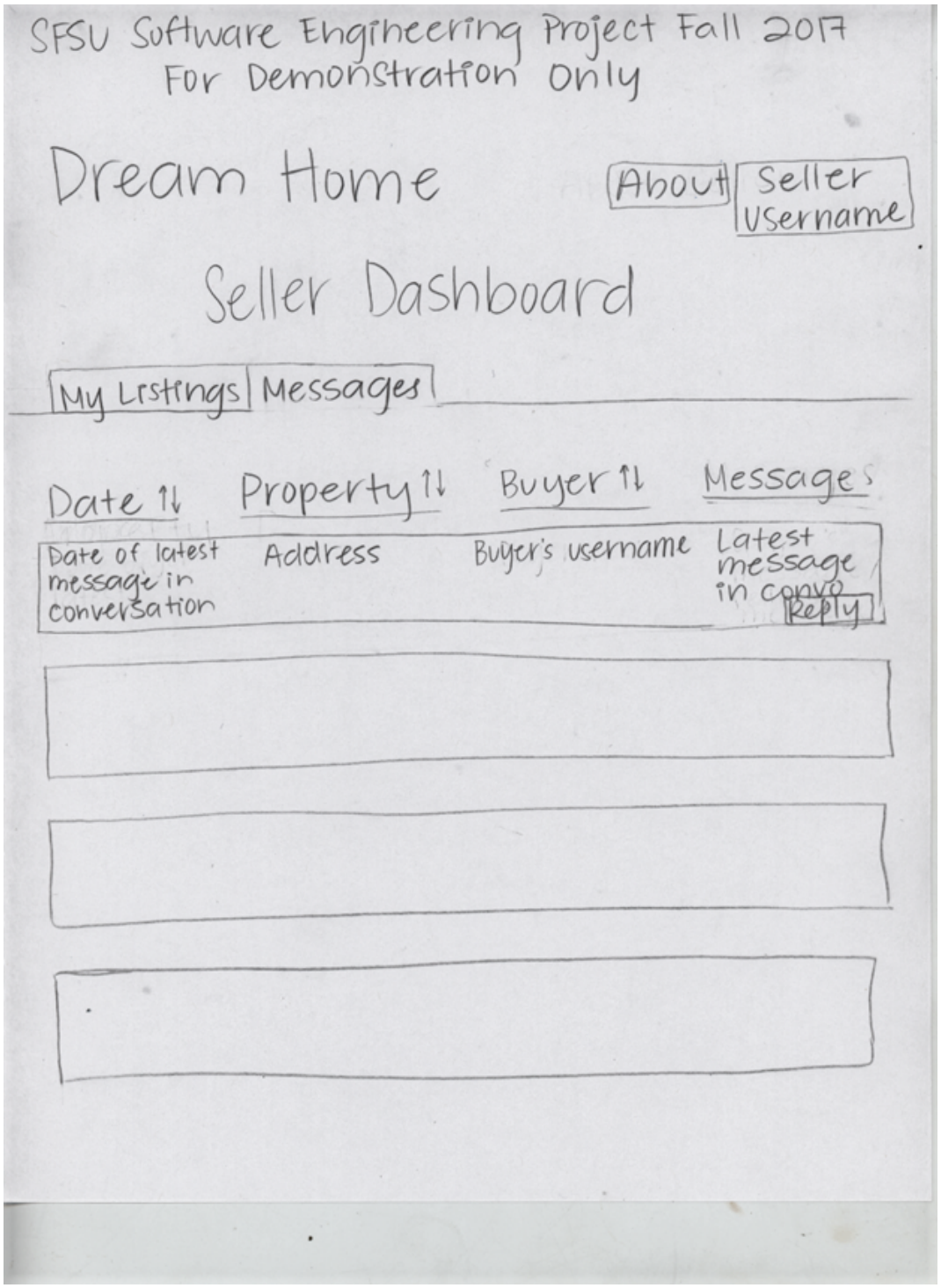
For Seller’s edit page, add edit buttons for house fields and images



Dashboard for listings



Dashboard for Messages



4. High Level Architecture, Database Organization

High Level Architecture:

- Each model will control access to their respective database items. The models will take requests from the controllers and return the appropriate data from the database. Models will access the database through a database helper module that controls the connection to the database and contains generic database functions. There will be a model to represent the objects represented in our application such as listing and user objects.

- All page requests will be processed/handled by the controllers. The controller will do and logic and processing of data needed to render a view. Controllers will access the database indirectly through requests to the appropriate models and forward data to the views. The controllers will process and validate any user input submitted through the views. There will be a controller to handle the different types of requests to our application such as listing, user, and search requests.

- The views will only be concerned with displayed webpages. Views will be provided data from the controllers and choose how to display that data in a user friendly format. There will be different views for each type of page in our application such as about, home, search, listing, and user pages/views.

Database Organization:

- User table:

Column	Type	Default Value	Nullable
user_id	int(11)		NO
user_type	tinyint(1)		NO
first_name	varchar(30)		NO
last_name	varchar(30)		NO
email	varchar(40)		NO
user_password	varchar(15)		NO
phone	int(11)		NO
i_agree	tinyint(1)		NO

- Listing table:

Column	Type	Default Value	Nullable
listing_id	int(11)		NO
address	varchar(100)		NO
state	varchar(25)		NO
city	varchar(25)		NO
zipcode	int(10)		NO
price	float		NO
posted_on	date		NO
bedroom_count	float		NO
bathroom_count	float		NO
pool	tinyint(1)		YES
ac	tinyint(1)		YES
floor_size	int(5)		NO
parking	tinyint(1)		YES
seller_id	int(11)		YES
image	longblob		YES
thumbnail	longblob		YES
price_per_square_f...	int(5)		YES
heater	tinyint(1)		YES

- Contacted_Listing table:

Column	Type	Default Value	Nullable
buyer_id	int(11)		NO
listing_id	int(11)		NO
message	varchar(500)		YES

Media Storage:

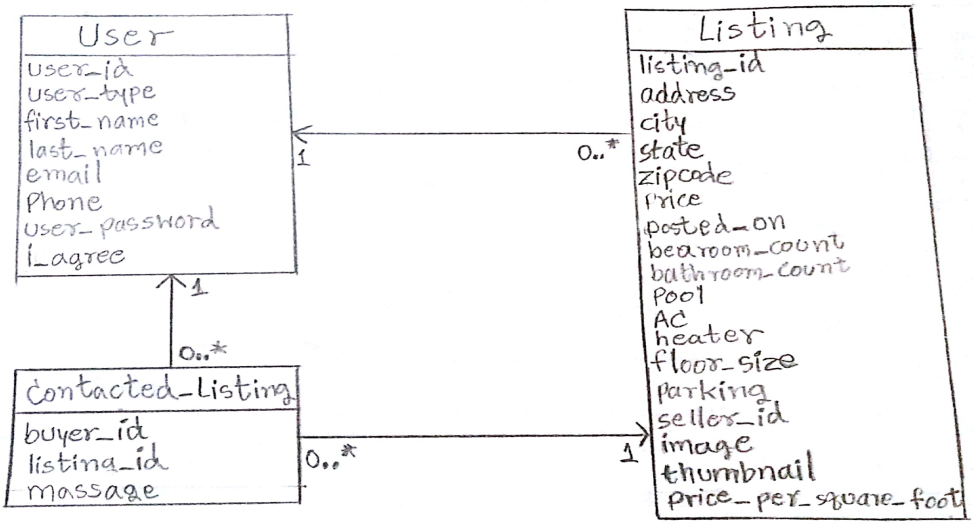
The images and thumbnails in our application will be stored as BLOBs in the database.

Search/filter architecture and implementation:

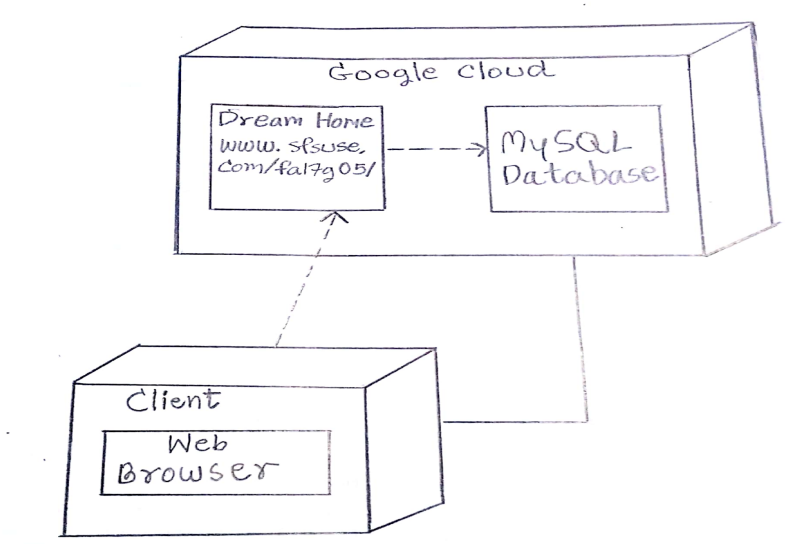
- If a user searches for no specific term, search results display all the listings from the database.
- If the user searches using pin code, state, city or address line, search results that are like the search term gets listed. We have used a ‘LIKE’ operator here which returns search results by comparing the database values with the search text.
- If a user wants to filter houses based on the price, number of bedrooms, number of bathrooms, availability of pool, heater_cooler, parking and floor_size, search result will be aligned according to the filter applied.
- If the user wants to list house based on Price Low-High, or High-Low, the search results will be sorted in that order.

5. High Level UML Diagrams

UML Class Diagrams



UML Deployment Diagrams



6. Key Risks

1. Skills Risks
    - All group members may not have all the right skills for our application, since every team member's skills vary. While we are all senior undergraduate students or graduate students, we might not have the in depth knowledge necessary for our application. While we might not know how to do a particular skill, we are all motivate students. We will seek out the help, either from the internet, our fellow students, or our professor, to learn the particular skill and continue developing our application.
2. Technical Risks
    - Whenever using new or unfamiliar technology, there is always a risk of the technical issues arising. For our application, we are going to implement BLOBs in our database, which is a difficult and unfamiliar technology to our team. To ensure all technical issues with BLOBs are resolved early, we have already begun working with BLOBs in our database. By fixing all technical issues early, we are setting up our application for success and avoiding any devastating last minute technical issues.
3. Teamwork Risks
    - Due to the size of our team and each team member's busy schedule, we definitely have scheduling issues. The entire team is only available to meet three hours a week, which means that our team might fall behind schedule or team members might not get the required help from their teammates. To fix this issue, we are constantly communicating to the entire team via Slack. We have also decided to meet in person in smaller groups (front-end, back-end, and leads).