**Tenant Direct**

Jeet Nair

Samuel Mason

Jue Wang

Zihao Huang

**Project Name:**

Tenant Swap

**Team member:**

Jeet Nair

Samuel Mason

Jue Wang

Zihao Huang

**The description of the project:**

The purpose of the project to build a marketplace online for departing tenants to swap furniture with incoming tenants.

Departing tenants can list their items and their minimum asking price.

All apartment and tenant information provided by landlord when they sign up.

Landlord directs tenants to website or extension from landlords website.

Separate layouts for departing tenants (sellers) and incoming tenants (buyers).

Landlords sign up and provide basic information (building name, floors, unit numbers). Tenants can then sign up and choose their building name, floor and unit number. Tenants can then go select to either go to the buying side or selling side of the webpage. There will be three levels of users (Tenants, Landlords, Administrators).

**Languages/Platform/Libraries:** Java, HTML, Javascript

**Backend:**

* System for users to communicate
* Central Server for Listings
* Clock System

**Fronted:**

* Secure Login
* Page Design

**Both Front and Back End:**

* Accepting Images to be Reuploaded
* Buyer/Seller GUIs

**Screen Sketches:**

**Actors (Users):**

* Administrators (People employed by the company)
  + Able to view information of all Users, Clients, and Administrators
  + Able to view listings of every User
    - Able to delist all items posted by users
  + Able to remove any user
* Clients (Landlords that pay for company services. They provide basic building information about their properties and a list of tenants that live there)
  + Able to see user information of all users assigned to their building
  + Sees all listings of users assigned to their building
    - Able to delist an item from a user assigned to their building
  + Able to remove a user that is assigned to their building
* Users (Tenants in buildings)
  + See their information only
  + List their items
  + View items listed in their current/future building

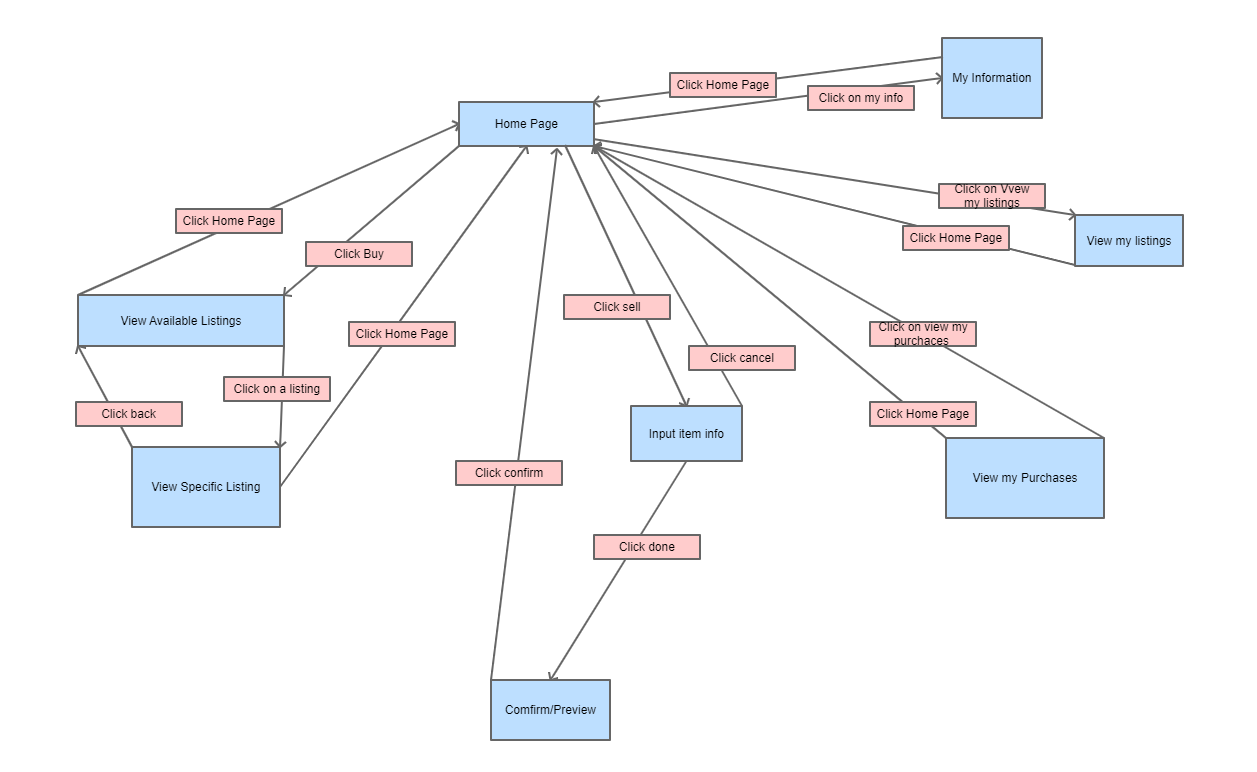
**Non-Function Reqs:**

* Performance
  + Application must have minimal response wait time
  + Layout must be very intuitive to the user
* Maintainability
  + Must be able to be maintained with minimum outages
    - New feature should be able to be added with minimal impact to existing features
  + Data Security and Storage
* Scalability
  + Be able to meet data growth as demand expands
  + Each new client (landlord) could bring in thousands of new tenants.
    - Be able to quickly add a lot of new data at once
* Portability
  + Webpage
    - Perhaps could be a direct extension of the client’s webpage

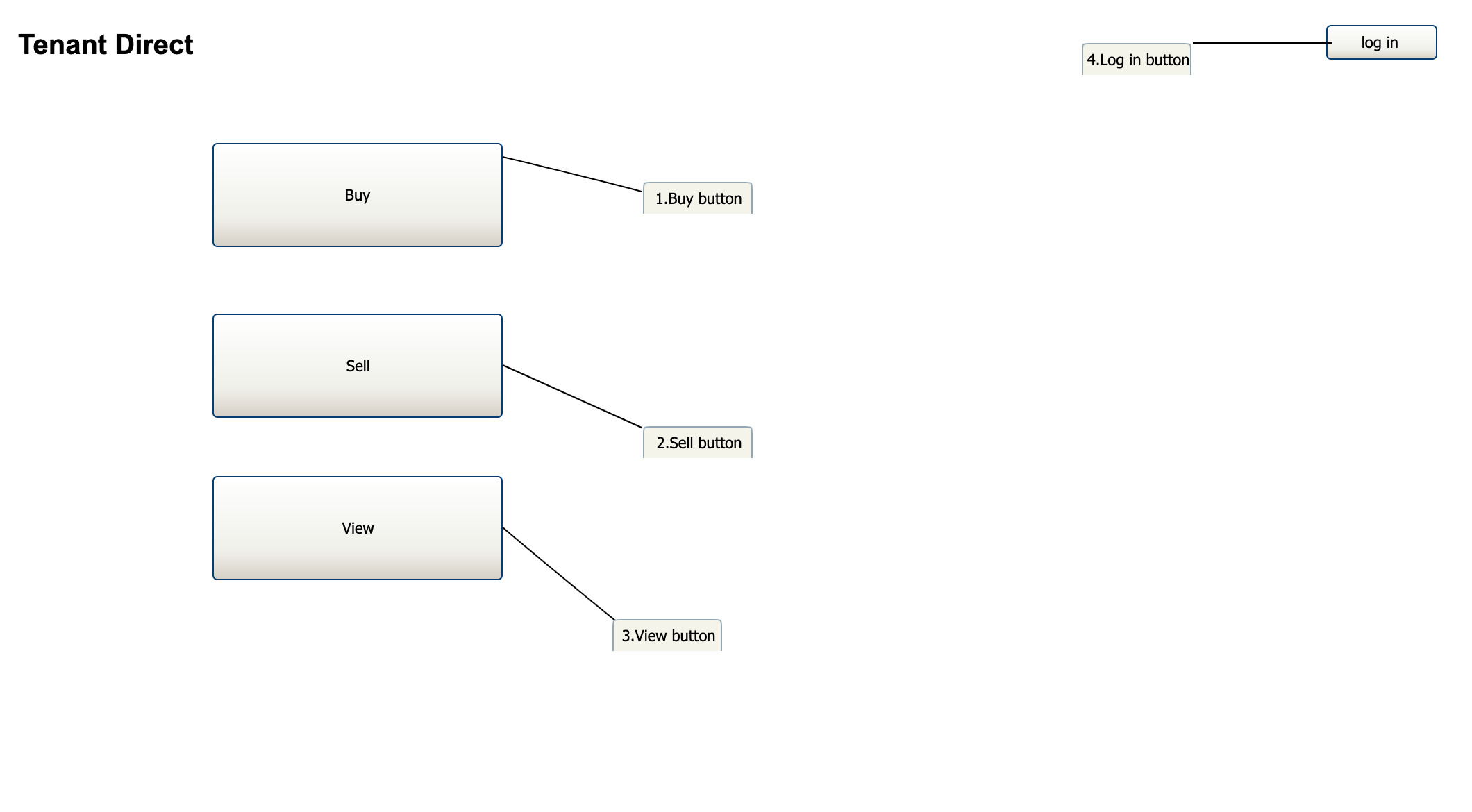
**Tables and fields:**

* User data
  + Username
  + Password
* Tenant data
  + Name
  + Phone number
  + Landlord
  + Building
  + Floor
  + Unit
* Item data
  + Type of item
  + Who it is available to (users in the same building)
  + Description
  + Pictures
  + Moving fee
  + Tax
  + Transaction fee
  + Price
    - Auction
      * Starting bid
      * Current bid
      * End date
      * Minimum increment
      * Optional best offer
    - Straight Listing
      * Price
* Homepage
  + Buy
  + Sell

**Screen flow diagram:**



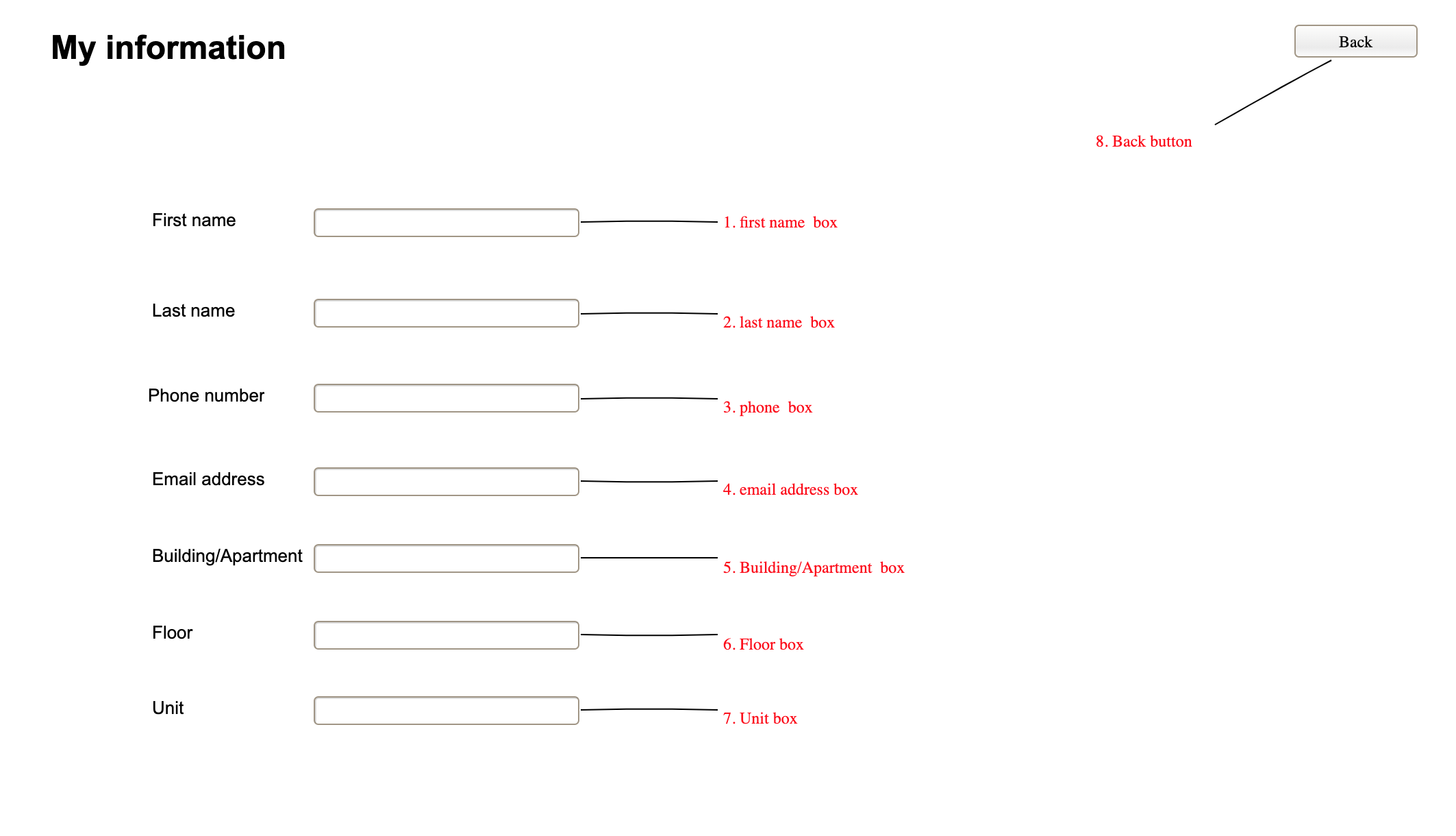
**Home page:**



Description:

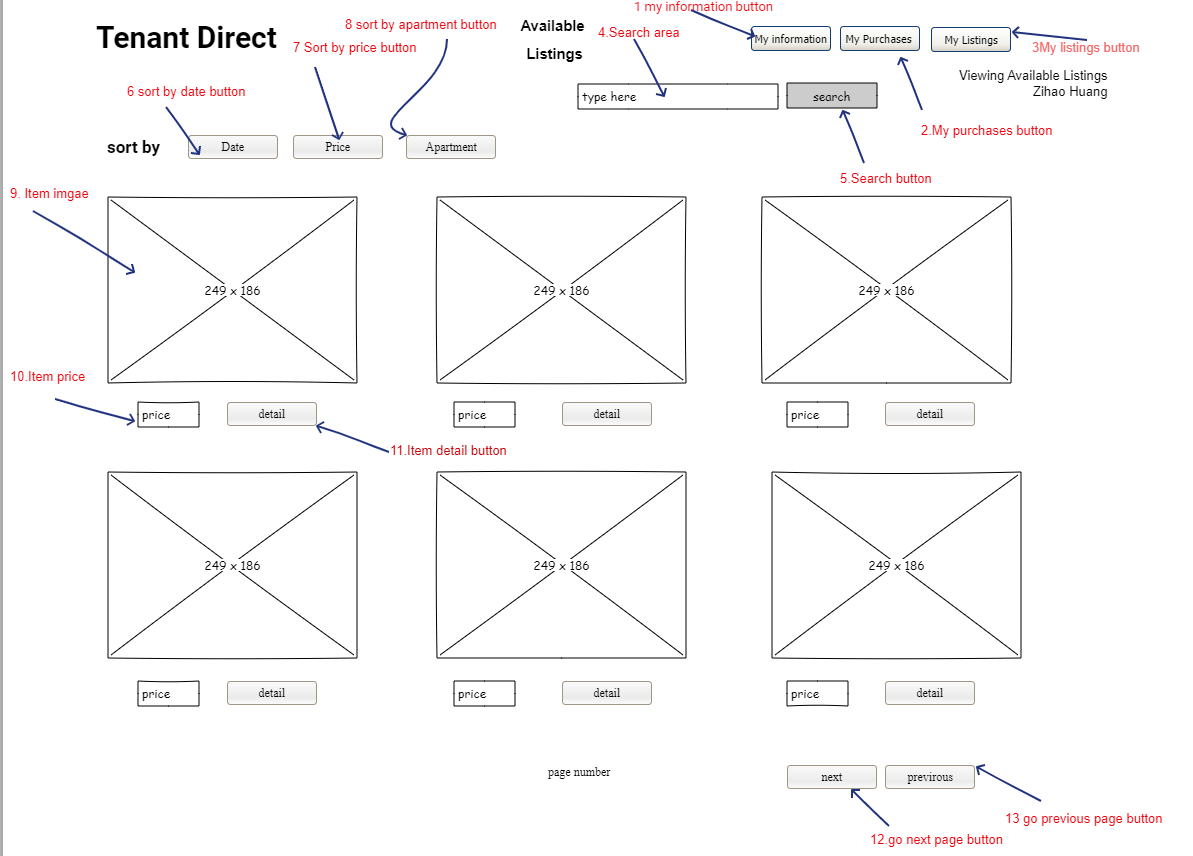
There are three buttons on the home page: Buy button(1), Sell button(2), and View button(3). After clicking the Buy button(1), you will see the items listed in the Buy page.

After clicking the Sell Button(2), you will go to the sell page where you can put the items you would like to sell. The last button is View Button(3), you can see the items you are sgoing to sell, or the items you have already purchased. There is another login button (4)where you can log in your account.



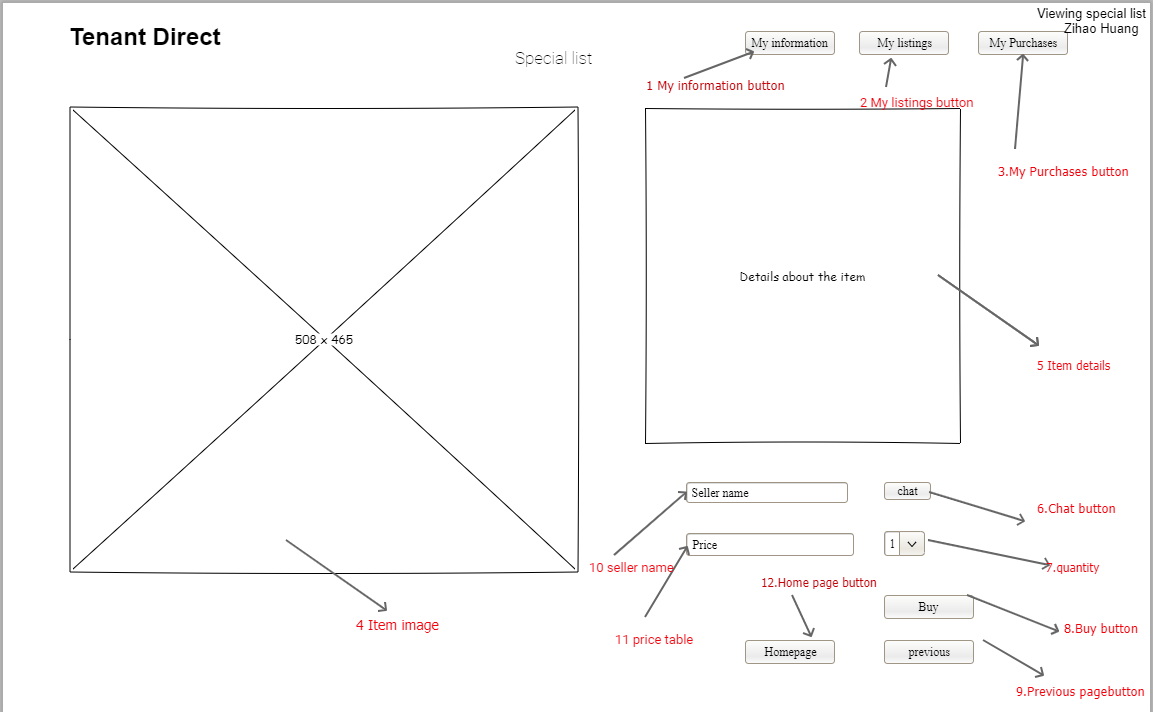
Description:

This is the page where you have all your personal information: your name(1) (2), your phone number(3), the email address(4), the building name or the apartment(5), the floor(6) and the Unit(7). You can edit your information here, the buyer will see these information after they decide to buy the items you listed. There is also a Back button(8), you can go back to the home page.

Description

The Available Listings webpage let users look through the items which are available at this time. Users could see the image of Item (9) and the price of Item(10). If the user wants to know more about the Item , they could click on the detail button and go to the special list page.

Users could see what they sell and purchases by clicking on my listing button(3) and my purchase button(2),They could also edit their information by clicking on the my information button(1).If the users want to search for special items ,they can type in table(4) and click search button(5). They could also let the item sort by date (6) by price (7) or by apartment(8). The users could go to next page (12) or previous page(13)



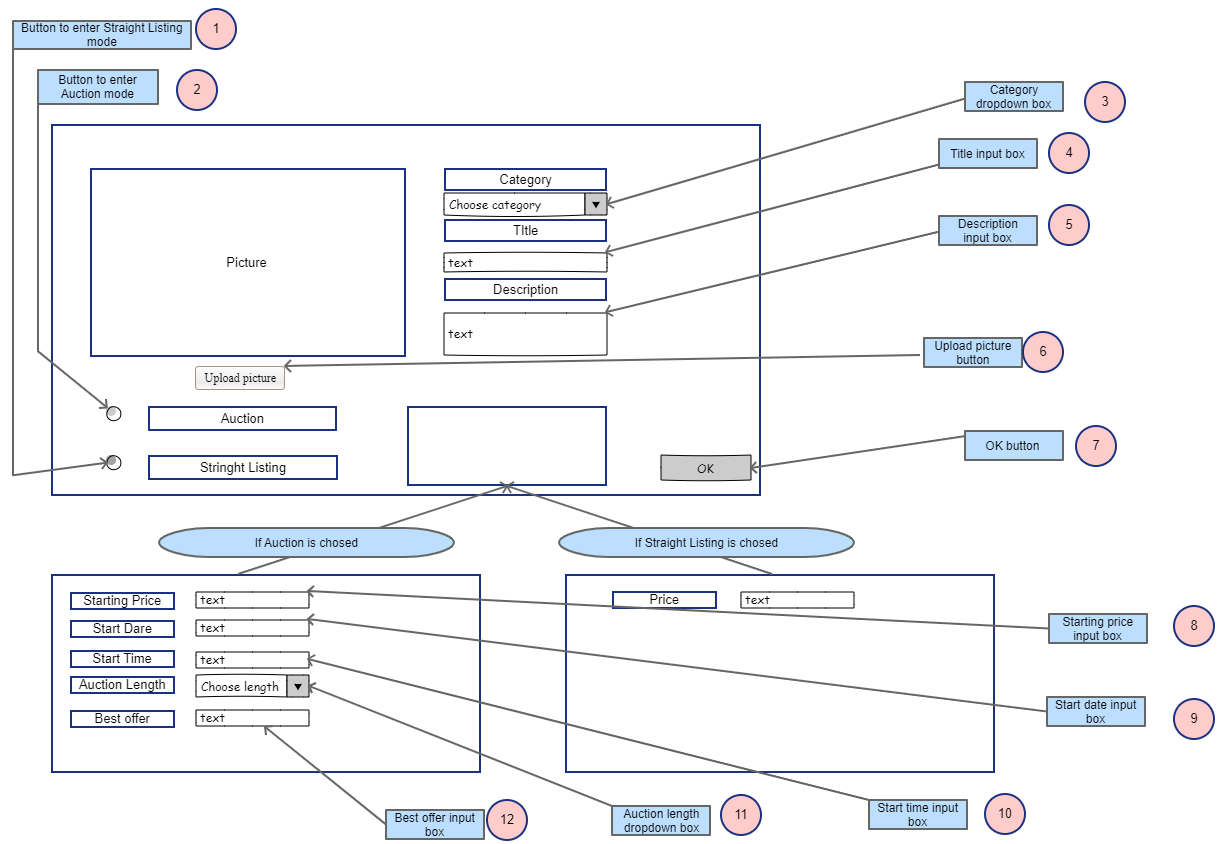
Description

The special list web page allows the user to look through the details of the item they want to buy. For (1)(2)(3) these button have same function as avaliable list. The user could see a larger image (4)than the available list.They could also see the details about the item(5),for example how long did the seller use it,where did they buy it.

The users could see the seller’s name (10) and could chat with the seller if they want to bargain by clicking on the chat button(6) If the user decided to buy the item, they could choose the quantity (7) and could see the price at the table(11) and then click on buy button (8) If the user change their mind, they could go to previous page (9) or homepage（12）

**Selling page:**

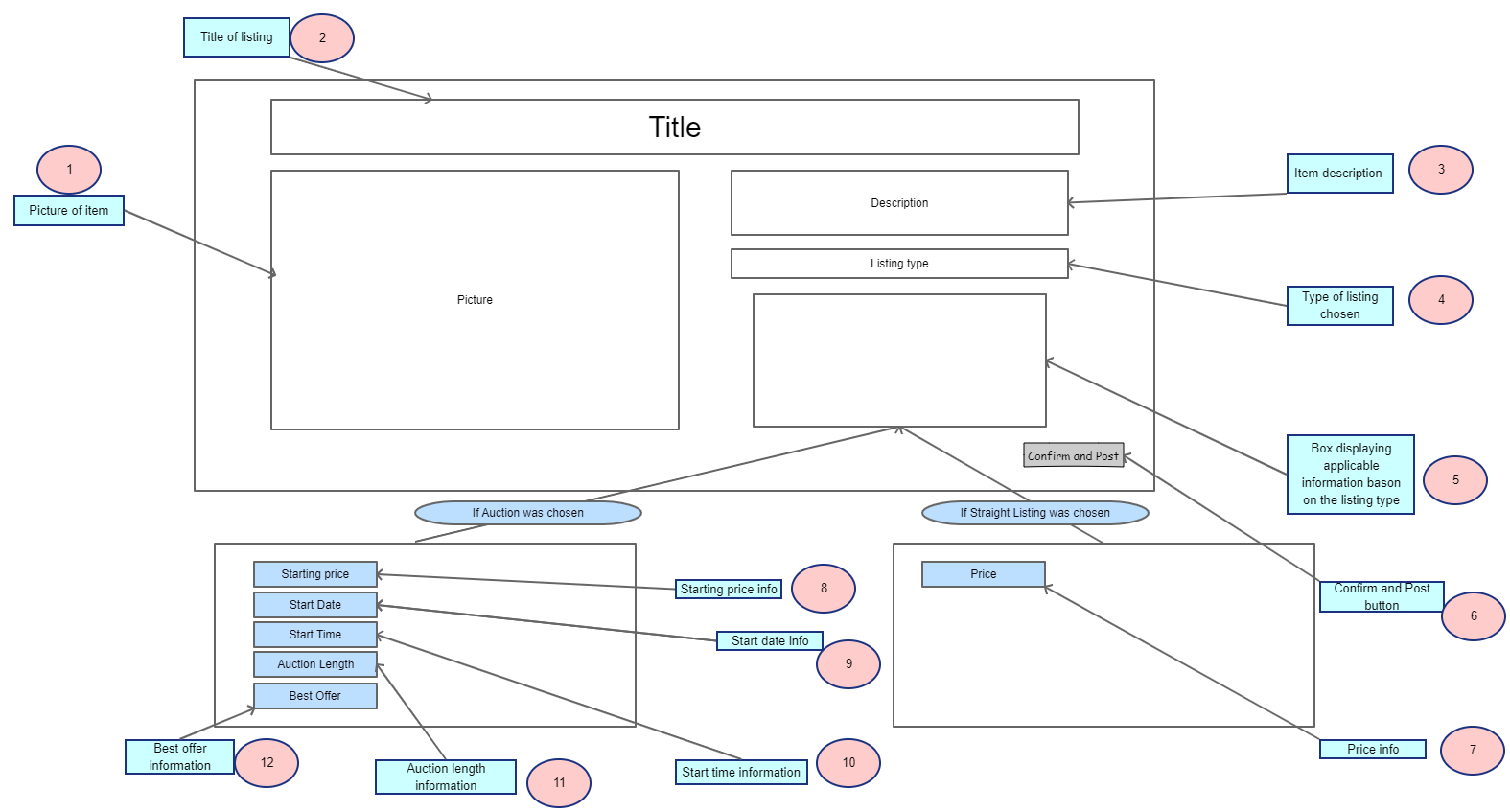
Input item info page



Description:

Upon clicking the sell button from the homepage, the user is taken to the page shown above. They can upload pictures using a button (6) and can view their picture(s) in a large window. They will choose a category (3), make a title (4), and add a description (5). They can then choose if they would like this listing to be an auction or a straight listing (1 or 2) and based on that selection they will enter additional information (8,9,10,11,12). Once this information is provided the user will be able to click the OK button to advance to a preview before officially posting the listing.

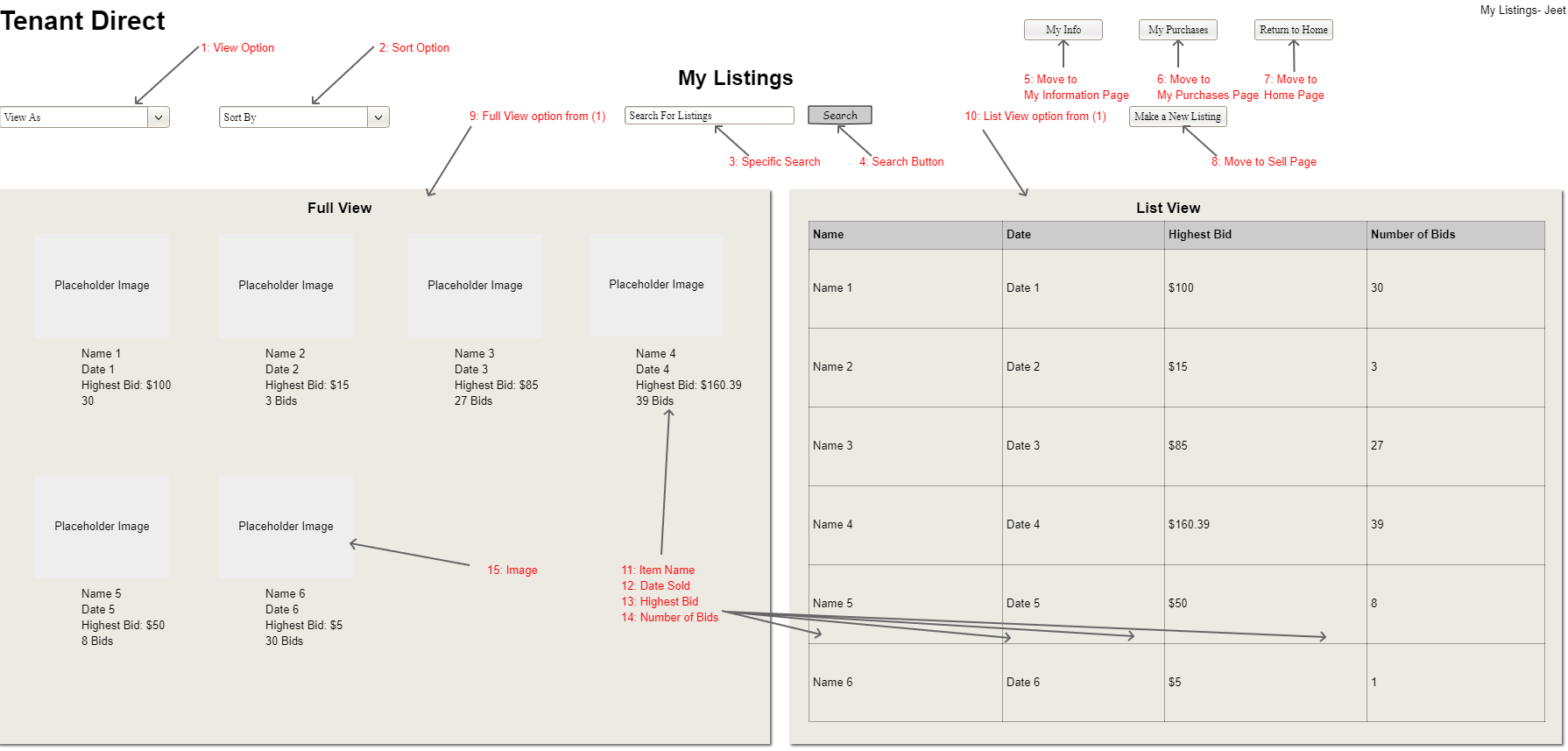
**Confirm/Preview page:**



Description:

After clicking the OK button the user will be given a preview of their listing. They will be shown the pictures of the item that they submitted (1). They will also be able to see the title (2), the item description (3), the type of listing they have chosen (4). They will also see a section (5) with the information specific to the type of listing they have chosen (7,8,9,10,11,12). When they have confirmed all the information they will be able to post the listing by clicking the Confirm and Post button (6).

**View:**

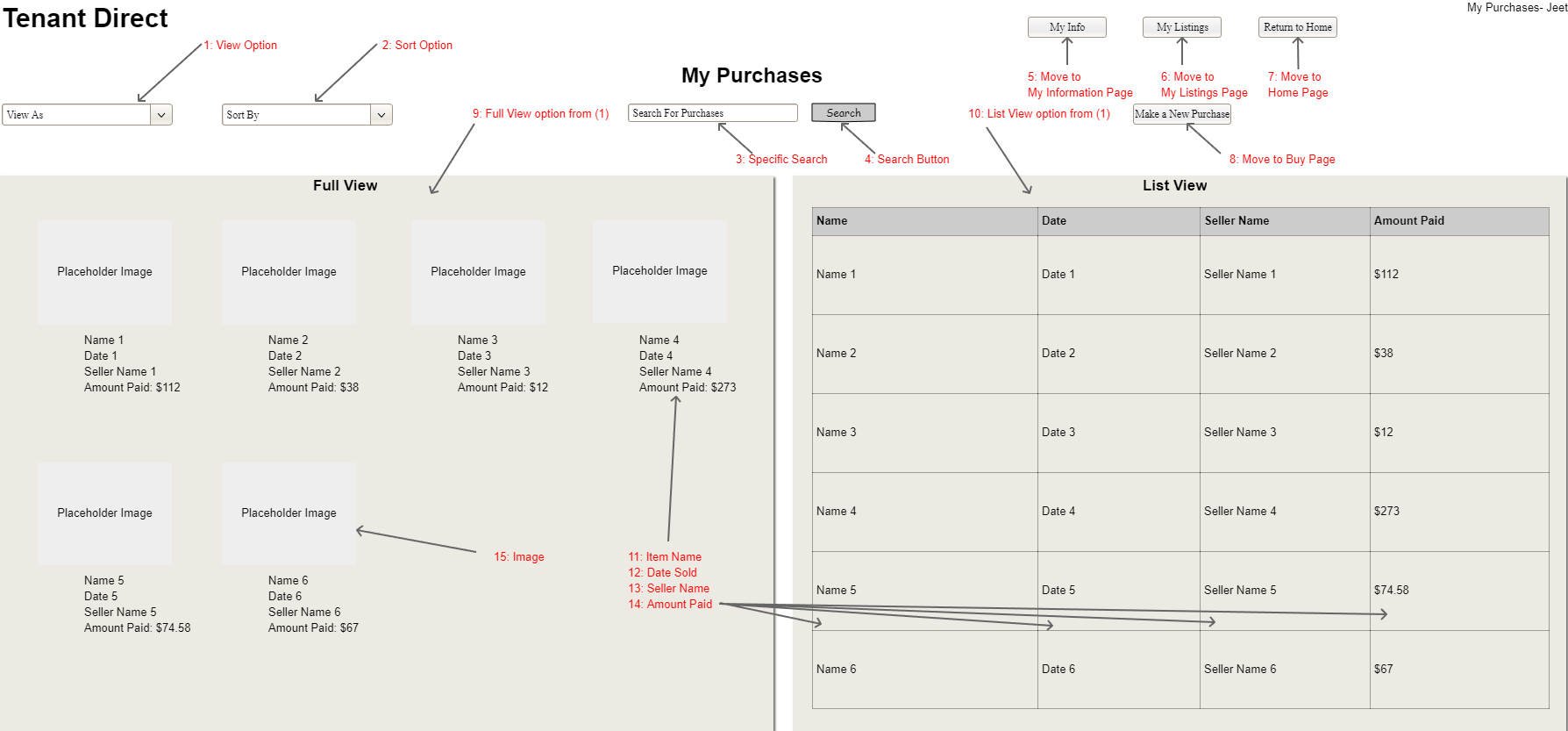
Description: The ‘My Listings’ page is used in order for the user to view all of the items that they currently have up for sale. They may utilize the ‘View’ selector (1), in the top left of the screen, in order to switch between a ‘Full View’ (9) and a ‘List View’ (10). Both views who the name of the listed item (11), the date the listing was made (12), the highest bid currently on said item (13), and the number of bids that have been made thus far (14). The ‘Full View’ has an additional feature which shows the image used for the listing (15) whereas the ‘List View’ does not and opts for a spreadsheet version. The ‘Sort’ selector (2), which is located to the right of the ‘View’ selector, allows you to modify the order of the listings by alphabetical A-Z or Z-A, earliest or latest date listed, highest or lowest bid, and highest or lowest bidders. If the user wishes to search for a specific listing, they can use the search bar (3) and the search confirm button(4) which are located at the top middle of the page. Finally, the ‘My Info’ (5), ‘My Purchases’ (6), ‘Return to Home’ (7), and ‘Make a New Listing’ (8) buttons do the following actions:

My info takes the user to their info page

My Purchases takes the user to their My Purchases page

Return to Home returns the user to the Home portal page

Make a New Listing takes the user to the sell page to put up a new listing

Description: The ‘My Purchases’ page is used in order for the user to view all of the items that they have purchased. They may utilize the ‘View’ selector (1), in the top left of the screen, in order to switch between a ‘Full View’ (9) and a ‘List View’ (10). Both views who the name of the listed item (11), the date the item was purchased (12), the name of the seller (13), and the amount that was paid for the item (14). The ‘Full View’ has an additional feature which shows the image used for the listing (15) whereas the ‘List View’ does not and opts for a spreadsheet version. The ‘Sort’ selector (2), which is located to the right of the ‘View’ selector, allows you to modify the order of the listings by alphabetical item name A-Z or Z-A, earliest or latest date purchased, alphabetical seller name A-Z or Z-A, and highest or lowest amount paid. If the user wishes to search for a specific item they have purchase, they can use the search bar (3) and the search confirm button (4) which are located at the top middle of the page.Finally, the ‘My Info’ (5), ‘My Listings’ (6), ‘Return to Home’ (7), and ‘Make a New Purchase’ (8) buttons do the following actions:

My info takes the user to their info page

My Listings takes the user to their My Listings page

Return to Home returns the user to the Home portal page

Make a New Purchase takes the user to the buy page to view the listings currently available for them to purchase.

**Design description:**

On the client side are using HTML with Thymeleaf to parse it. The spring boot application serves the webpages by serving up the view that it rendered in the browser. The user interacts with these web pages by using form to post information back to the server, clicking on links and buttons to navigate around the layout. Multiple clients can access the site at the same time. Each client will connect to the server through the Tomcat server through the HTTP rest controllers.

The user signs up by filling out the form that is served up when they navigate to the register section of the sign in page. This information is transferred to the server via the RegisterController which receives the post request mapping. RegisterController then calls the RegisterService method that takes the information (name, email, phone number, password, ect) and using the UserDatabase interface puts the information into the User table of the SQL database.

The user can then login with the username and password that they chose previously. When this information is posted the LoginController will receive the post request and will call a method in LoginService that will check the email and password that was entered in the login page match an existing email and password that is already in the database. If there is a match the user will be redirected to their home page.

Once in the home page the user can view all their information, their listings, their purchases, and can view a list of all users. The user can click on List and Item and will be able to post an item in the same fashion as the Register Process. The items will go in an Item table in the SQL database. The user can upload images for their item that they are listing. The file will be assigned a number (likely the id of item in the Item table) and will be written to a directory on the server (not in the folder in the spring boot project where static images are kept) and this file path will be saved.

When the user clicks on “see available items” from their homepage, they will see a list of available items. The server will simply give Thymeleaf the list of Java objects and the Thymeleaf parser will be capable of enumerating these. The user can click on any item in this list to view that item. They will be redirected to view this item and can see any pictures displayed alongside the item information. They also can view the name of the seller that listed this item and can click a link to chat with the user. The chat will be implemented using websockets.

The view users functionality comes with sorting. The user can choose to view other users based on Email, Address, Phone, Building, ect. The sorting function is in the Userservice. The sorting takes place and the list of User objects is given to Thymeleaf to enumerate and display.

**Link of the video to show how our project works:**

<https://youtu.be/_r8ByWBOKkg>