SW Engineering CSC648-848 Summer 2021
dropsell.gq, Jose's Angels

Team 03

Name	Email	Roles
Mitchel Baker	mbaker3@mail.sfsu.edu	Team lead
Krina Bharatbhai Patel	kpatel11@sfsu.edu	Frontend lead
Charmaine Eusebio	ceusebio1@mail.sfsu.edu	Frontend engineer
Rowena Elaine Echevarria	rechevarria@mail.sfsu.edu	Frontend engineer
Michael Schroeder	mschroeder@mail.sfsu.edu	Backend lead, Github master
Kenneth N Chuson	kchuson@mail.sfsu.edu	Backend engineer
Jamie Dominic Walker	jwalker5@mail.sfsu.edu	Backend engineer

Milestone 2
July 8, 2021

History Table

Version	Date	Notes
M2V1	07/08/2021	

Table of Contents

1. Data Definitions

1. Unregistered User: A user who can visit the website, explore the products displayed in the marketplace, and checkout any public facing pages. Unregistered users need to register to use all the features of the website related to viewing auctions, commenting and rating products, messaging buyers/sellers, and purchasing products.
    a. A general user shall be able to create an account.
    b. A general user shall choose the option of creating an account as buyer/seller/both.

2. Registered User: A registered user is able to access the website with all features. Registered users need to login to the website for buying or selling products.
    a. User login
        i. username; type VARCHAR(45), NN
            1. Must enter username to login
        ii. password; CHAR(90), NN
            1. Must enter a valid password, encrypted with Bcrypt library

    b. User account details
        i. user_id; type INT, PK, NN, AI
        ii. email; type VARCHAR(90), NN
        iii. first_name; type VARCHAR(45), null
        iv. last_name; type VARCHAR(45), null
        v. phone; type VARCHAR(15), null
        vi. street; type VARCHAR(90), null
        vii. city; type VARCHAR(45), null
        viii. state; type VARCHAR(45), null
        ix. zip; type VARCHAR(15), null
        x. created_at; type DATETIME, NN
        xi. is_active; BOOL, NN
        xii. Payment details
            1. Attributes
                a. CC number
                b. Expiration date
                c. 3 digit code
                d. Zip
            2. Payment details should be handled by Stripe.js. We may store payment details in the database for multiple payment options in the future.
        xiii. User is a buyer? Seller?
            1. is_buyer; type BOOL, NN
            2. is_seller; type BOOL, NN

    c. User conversations
        i. conversation_id; type INT, PK, NN, AI
        ii. sending_user_id (FK to sending user); type INT, NN
        iii. receiving_user_id (FK to receiving user); type INT, NN
        iv. Messages
            1. message_id; type INT, PK, NN, AI
            2. conversation_id (FK to conversation id); type INT, NN

3. message_timestamp; type DATETIME, NN

d. User meetups
   i. meetup_id; type INT, PK, NN, AI
   ii. buyer_id (FK to buyer id); type INT, NN
   iii. seller_id (FK to seller id); type INT, NN
   iv. meetup_time; type DATETIME, NN
   v. meetup_location; type VARCHAR(180), NN

3. User session
   a. Users should have an active session created in order to keep track of the date and time they logged in. This information will be used to keep track of the length of the user's session.
   b. When the user logs in, their session data should be updated in the database.
   c. If the user's session has expired, then the user should be required to log in again.
   d. If, for example, the user's session is still active and they decide to refresh the page, then the user should stay logged in.
   e. Attributes
      i. session_id; type INT, PK, NN
      ii. session_expires; type DATETIME, NN
      iii. session_data; type VARCHAR(180), NN

4. Buyers: can browse products and buy them.
   a. shopping_cart
      i. shopping_cart_id; type INT, PK, NN, AI
      ii. buyer_id (FK to buyer id); type INT, NN
      iii. subtotal; type VARCHAR(45), NN
      iv. shopping_cart_products
         1. product_id; type INT, PK, NN, AI
         2. shopping_cart_id (FK to shopping cart id); type INT, NN
         3. title; type VARCHAR(250), NN
         4. price; type VARCHAR(90), NN
         5. quantity; type VARCHAR(45), NN

5. Sellers: can upload product information and sell them.
   a. Seller Ratings
      i. seller_rating_id; type INT, PK, NN, AI
      ii. seller_id; type INT, NN
      iii. seller_rating (5-star rating system, from 1-5); type VARCHAR(10), NN

   b. Products: The items which are uploaded by sellers, and purchased by buyers.
      i. product_id; type INT, PK, NN, AI
      ii. seller_id; type INT, NN
      iii. title; type VARCHAR(250), NN
         1. At Least 5 word Title or name of the product
      iv. description; type VARCHAR(500), NN
         1. At Least 10 word Description of the product
      v. image; type VARCHAR(100), NN
         1. At least 3 images of size (600x600 or 2x2) for product visibility

vi. price; type VARCHAR(90), NN
    1. The price for product (For sale and auction only)
vii. category; type VARCHAR(90), NN

c. Product Comments
    i. product_comment_id; type INT, PK, NN, AI
    ii. product_id (FK to product id); type INT, NN
    iii. creator_id (FK to user id); type INT, NN
    iv. comment_timestamp; type DATETIME, NN
    v. comment; type VARCHAR(500); NN

d. Product Ratings
    i. product_rating_id; type INT, PK, NN, AI
    ii. product_id (FK to product id); type INT, NN
    iii. creator_id (FK to user id); type INT, NN
    iv. product_rating (5-star rating system, from 1-5); type VARCHAR(10), NN

e. Product Refunds
    i. product_refund_id; type INT, PK, NN, AI
    ii. product_id (FK to product id); type INT, NN
    iii. buyer_id (FK to buyer id); type INT, NN
    iv. seller_id (FK to seller id); type INT, NN
    v. refund_amount; type VARCHAR(90), NN

6. Auction Products
    a. product_id; type INT, PK, NN, AI
    b. seller_id (FK to seller id); type INT, NN
    c. starting_bid; type VARCHAR(90), NN
    d. auction_duration: type VARCHAR(90), NN

7. Top-Purchased Products
    a. product_id; type INT, PK, NN, AI
    b. seller_id (FK to seller id); type INT, NN
    c. total_purchased; VARCHAR(90), NN
    d. added_at; type DATETIME, NN

8. Daily Deal Products
    a. product_id; type INT, PK, NN, AI
    b. seller_id (FK to seller id); type INT, NN
    c. deal_duration: type VARCHAR(90), NN

9. Shipping Products
    a. product_id; type INT, PK, NN, AI
    b. buyer_id (FK to buyer id); type INT, NN
    c. seller_id (FK to seller id); type INT, NN
    d. shipping_from; type VARCHAR(180), NN
    e. shipping_to; type VARCHAR(180), NN
    f. transaction_total; type VARCHAR(90), NN

10. Redux related data definitions
    a. Login
        i. Actions
            1. setUsername(username)
                a. Action type: ‘USER_SET_USERNAME’
                b. Datatype: String
            2. setPassword(password)
                a. Action type: ‘USER_SET_PASSWORD’
                b. Datatype: String
            3. loginUser()
                a. userData(username, password)
            4. redirectUserAfterLogin(loggedIn)
                a. Action type: ‘USER_IS_LOGGEDIN’
                b. Datatype: String
        ii. Reducer
            1. INITIAL_LOGIN_STATE
                a. Username; datatype String
                b. Password; datatype String
                c. loggedIn; datatype Boolean

    b. Register
        i. Actions
            1. setUsername(username)
                a. Action type: ‘USER_SET_USERNAME’
                b. Datatype: String
            2. setPassword(password)
                a. Action type: ‘USER_SET_PASSWORD’
                b. Datatype: String
            3. setConfirmPassword(confirmPassword)
                a. Action type: ‘USER_SET_CONFIRM_PASSWORD’
                b. Datatype: String
            4. createUser()
                a. userData(username, password, confirmPassword)
            5. redirectUser(registered)
                a. Action type: ‘USER_IS_REGISTERED’
                b. Datatype: Boolean
        ii. Reducer
            1. INITIAL_REGISTER_STATE
                a. Username; datatype String
                b. Password; datatype String
                c. confirmPassword; datatype String
                d. Registered; datatype Boolean

    c. Products
        i. Actions
            1. setTitle(title)
                a. Action type: ‘PRODUCT_SET_TITLE’
                b. Datatype: String
            2. setDescription(description)

        a. Action type: ‘PRODUCT_SET_DESCRIPTION’
        b. Datatype: String
    3. setPrice(price)
        a. Action type: ‘PRODUCT_SET_PRICE’
        b. Datatype: String
    4. setImage(image)
        a. Action type: ‘PRODUCT_SET_IMAGE’
        b. Datatype: String
    5. setSuccess(isSuccess)
        a. Action type: ‘PRODUCT_SET_SUCCESS’
        b. Datatype: Boolean
    6. setCategory(category)
        a. Action type: ‘PRODUCT_SET_CATEGORY’
        b. Datatype: String
    7. setCategories(categories)
        a. Action type: ‘SET_CATEGORIES’
        b. Datatype: Boolean
    8. changeDropdownText(text)
        a. Action type: ‘CHANGE_DROPDOWN_TEXT’
        b. Datatype: String
    9. createProduct()
        a. formData(title, description, price, category, file)
    10. getProducts(products)
        a. Action type: ‘GET_PRODUCTS’
        b. Datatype: Array
- ii. Reducer
    1. INITIAL_PRODUCT_STATE
        a. Title; datatype String
        b. Description; datatype String
        c. Price; datatype String
        d. Category; datatype String
        e. File: datatype String
        f. filePreview; datatype null/URL object
        g. isSuccess; datatype, null/boolean
        h. Products; datatype Array
        i. Categories; datatype Boolean
        j. dropdownText; datatype String

11. Search data definitions
    a. Query; datatype URL object
    b. searchQuery/setSearchQuery(); datatype URL object/empty String
    c. filterProducts(products, query); datatype Array

2. Prioritized Functional Requirements

Priority 1

Marketplace

1. Users should be able to interact with a search bar for looking up new products.
2. Create an auction, buy it now, or best offer options for listing.
    a. More options for sellers to earn money.
3. Users should be offered a base price for automatic reject or accept
4. Buyers should be able to interact with sellers using the google maps api
    a. Buyers should have a clear indication of their general location in correlation to the sellers. They should also be provided with travel times
5. Users should be shown ads of similar items on the product page
    a. To attract more customers to buy products.
6. Users should be shown a shipping fees table
    a. Breakdown of the shipping fees for time efficiency
7. Order status: confirmed, processing, shipped, returned. This will allow the user to track the status of their order/s.
    a. The user should be able to get immediate updates on the status of their order. They should also be able to check their order history, and then check on specific orders to see its processing, shipping, or return details.
8. Users should be able to track the delivery of their package.
    a. This will allow users to confirm that they have received their order.
9. Display “out of stock” for empty selling items.
    a. This will let users know that the item is no longer available; they will then have the option of being contacted by email when the item comes back in stock.
10. International marketplace- buyers can buy products wherever they are in the world.
11. International marketplace- sellers can sell products wherever they are in the world.

Website Features

12. Consignment operations, either for having items delivered or for meetup with the seller, or from a designated pickup location.
    a. Freedom to choose buyer’s delivery options.
13. Price matching, for items either already posted on our app or compared to other items on other websites (amazon, craigslist, eBay)
14. Daily deals which will highlight the sales going on and attract potential buyers to products.
15. Users should able to zoom in or zoom out of an item with a magnifying glass feature

16. Tax calculated automatically based on buyer origin. This tax will be reflected at the checkout process page.

Buyers

17. Purchase agreement to be signed by every user intending to make a purchase on the app. The buyer will agree to conduct business according to laid out rules.
18. Buyers should be able to seamlessly send an inquiry of interest or any questions to the seller of their choice
19. All of the user's selected items to be purchased will be added to their shopping cart.
20. Buyers should be able to remove an item from their shopping cart, and the total list of items should be updated. Buyers should also be able to return to the main shopping menu to look for other items if they are not done.
21. Cancel/modify order option. This will allow buyers to cancel their order or make changes in case they'd like to add more items to their cart.
22. Buyers will receive a detailed receipt of their purchase with tax and total after checking out through email.
23. Buyers will receive a confirmation email of their purchase with shipping information.
24. Buyers should be able to add their receipt information after checking out with their cart for the first time.
    a. If they are new users, have them update their receipt info. If they are returning users, then they should be able to choose their existing receipt info or add new credentials.
25. Buyer payment option, (if user is buying for the first time, then have them update their payment option, if they have opted to remember this information, then payment will already be there)
26. The buyer should be notified by final invoice of their purchase's pickup time/delivery time, expected time of arrival, name of seller, location to meet them at, their contact info, and the final total to be paid.
27. Buyers should be able to rate/star products to keep track of them.
28. Buyers should be able to choose delivery options.
29. Buyers should be able to interact with the seller's review page
30. Product reviews on product listing page. The buyer will be able to submit a review for purchased items.
31. Watch item feature/list. If a product is in an auction, the buyer will be able to keep track of the most recent updates to the item.
32. When a buyer purchased an item, it will show some top-rated items that are related to that buyer's purchase item.
33. Email order confirmation to both buyer and seller. The email will contain information of the product sold, time it was sold, and who bought it.

Sellers

34. Sellers should be able to create a new listing with a name, description, image, etc.
    a. Each listing also needs their own respective ID
35. Sellers should be able to edit the description, title, and price of their items. This will allow the seller to advertise their products better.
36. Seller contract to be signed by all users who wish to sell. This will provide sellers with an outline of how to sell on the app and rules they must follow.
37. Seller listing fees for each product that is sold. This will include a percentage of each sale being taken as profit for the app.
38. Sellers should be able to adjust the quantity of an item that is available for sale.
39. Sellers should be able to delete an item. This will allow sellers to delete any items they think that will not be able to be sold.
    a. Ability to have multiple select delete items.
40. Under the seller section of the user's profile information, users should be able to keep track of all the items they have posted.
41. Seller tool - relist item. This will provide sellers with the option to easily relist their product being up for sale.
42. Sellers will get email confirmation that one of their products has sold.

Auction

43. Filtering for item searching, min/max price, location filtering (within 50 miles), type of shipping (pickup, delivery. This will help buyers and sellers to choose the type of delivery options.
44. Sellers should be able to have full control over the settings of their auction.
    a. They should be able to set the average starting bid, the total duration of the auction (1, 3, 5, 7 days), and the time when the auction begins.
45. When an auction is posted for bidding, users should be able to see the current price it is set at, the amount of time remaining in the auction, and any bids from others.
    a. To make auctions real time, we will have to set up some sort of WebSocket lobby where everyone partaking in the auction has access to the auction's stats.
46. Buyer can bid on products with one click.
47. Buyers can see how many bids the product has.
48. Buyers/sellers can see the live countdown of the auction.
49. Sellers can set the auction to go as long as 30 days, or as short as 1 hour.
50. Buyers cannot retract their bid.
51. If a user wins an auction, they should be notified immediately, either by text or by email. (email seems more doable at this point, texts with Twilio are added cost)

Messaging

52. Contact the seller directly on the website.
53. Contact the seller via email.

Priority 2

Marketplace

54. Users should see an advanced search bar for more detailed searches.
    a. Users should be able to choose from a broad range of details from size, price range, color, etc.
55. If a seller has different styles of their product, i.e. colors or fabrics, then users should be able to choose their preferred style
56. Buyers/sellers should be able to schedule/edit meetup times if they agree with each other
57. If a product is purchased multiple times by the same user and it happens to become sold out, then the user should be notified and refunded if necessary
58. Share listing feature to share products with friends, through email, social media, or a link
59. Flagging system for reporting suspicious items. Users can either flag the item as inappropriate, a scam, or they can continue the process further by submitting a ticket to us for further review.
60. Users should be able to go back to items they have recently viewed.
61. Must show the maximum time length shipping delivery item.
62. Must show the minimum time length shipping delivery item.
63. User monitoring so that we can display to the user the posts they recently interacted with. I.e., keep track of the IDs of the posts they click on, and before redirecting to another portion of the app, send a post request to the server which will add these items to the database for interaction later.
64. International marketplace- currency converter

Website Features

65. Sourcing form (daily item forecast), profitability test, how much it costs to ship the product as opposed to being shipped on other services.
66. Users should be able to take advantage of an algorithm which keeps track of how many interactions their product have
    a. There will be functions attached to each listing which will send/receive data from the node API to create these real-time updates
67. Item hashtags to help with categorization. Sellers will be able to use hashtags to categorize their products. All users will be able to search products using hashtags.
    a. To see the best deals.
68. Buyers should be shown items which are similar in price or category to the item being viewed. A comparison between the item specs could be displayed as well.

    a. As an example, let us say that you are looking to buy an USB-C adaptor for connecting a monitor to your computer. When clicking on one USB-C item, users should have the option to compare the price, item features/details, and specs in depth to USB-C adaptors of similar brands so that they can make the most accurate decision for what they are looking for.

69. Sellers should be able to see the statistics related to how many buyers have purchased their listings
70. The seller should be able to keep track of the median price ranges of items similar to the one they are products
    a. For example, Dropsell has a few tables listed in its marketplace which have been set to an average price of $500. If a seller lists a table and he/she sets the price to $100, then Dropsell would notify the seller that they may be able to get a better profit by increasing their price.

Buyers

71. Buyers should be able to click on the product listing images to zoom in and see more details of the product.
72. Purchase/order history for buyers to keep track of their past purchases.
73. When a user clicks on an item they would like to see more of, display not only the photo slider but also the ability of messaging/sending an inquiry to the seller.
74. Favorite's list - the buyer will be able to add what they are interested into an ongoing list.
75. Wishlist. This will allow the buyer to save the items they would like to buy in the future.
76. Discount codes for frequent buyers. The codes will have expiration dates and will attract business by providing special discounts for potential buyers.
77. Coupon/promo code box for the buyer to enter the code and have the discount applied to their shopping cart.
78. Generate a referral URL which existing users can send to their friends.
79. Users should be able to subscribe to a specific seller so that they can be notified if the seller adds new items, adds updates, or makes any changes to the current price of an item.
80. Buyers should be notified about the items they viewed previously, and they should be asked if they are still interested in these items.

Sellers

81. Storefront profile page for sellers. This will showcase some products that the seller has up for sale, along with some details about who the seller is or what their shop is about.
82. Seller tool - send offers to specific buyer/buyers. This will allow sellers to gain potential business from buyers who are interested in their products.

83. Each seller’s product should have its own review page
    a. To provide buyers with insight into the functionality or state of the products being bought on our site.
84. Each seller should have their own review page/star rating
    a. Proving seller’s credibility
85. If the seller's item(s) is/are illegal, then it is the seller’s responsibility to display “illegal item”, otherwise the seller can have a chance to get into trouble.
86. While displaying “illegal item”, then the seller must explain the illegal item.
87. Seller’s product listing will reflect when an item is almost out of stock.
    a. “Only a few left in stock” or “Last one available”
88. Striking system for users for wrongdoing. This will allow the app to prevent users from continuing misconduct or violations of the terms of service. Perhaps 3 strikes on a user account will lead to a ban.
    a. User ban/blacklist

Auction

89. Buyers can bid on the products as many times as possible.
90. Buyers can watch products for auction.
91. Buyers can see how many other buyers are bidding on the same product.
92. Buyers will get notified when the auction is 5 minutes from being over.
93. Product listing will have time until the auction is over.
94. After the user wins an auction, they should be notified and then provided the option of choosing from multiple payment options.
    a. The user can pay with credit card, Paypal, or ApplePay/GooglePay (if time)

Priority 3

Marketplace

95. Wedding registry for those who wish to create a Wishlist for their special day. This will allow users to add products from the app into a list that they want to share with others.
96. Show people who have the same purchase as you.
    a. When a customer purchases an item, the feature will pop up the list of customers that have the same purchase and a customer will be able to message or call one of the customers about relating to the similar purchase item.
97. If the buyer's and seller's home location are closer around 1 or 2 miles, then the algorithm will hide the time length of the shipping delivery item.

Buyers

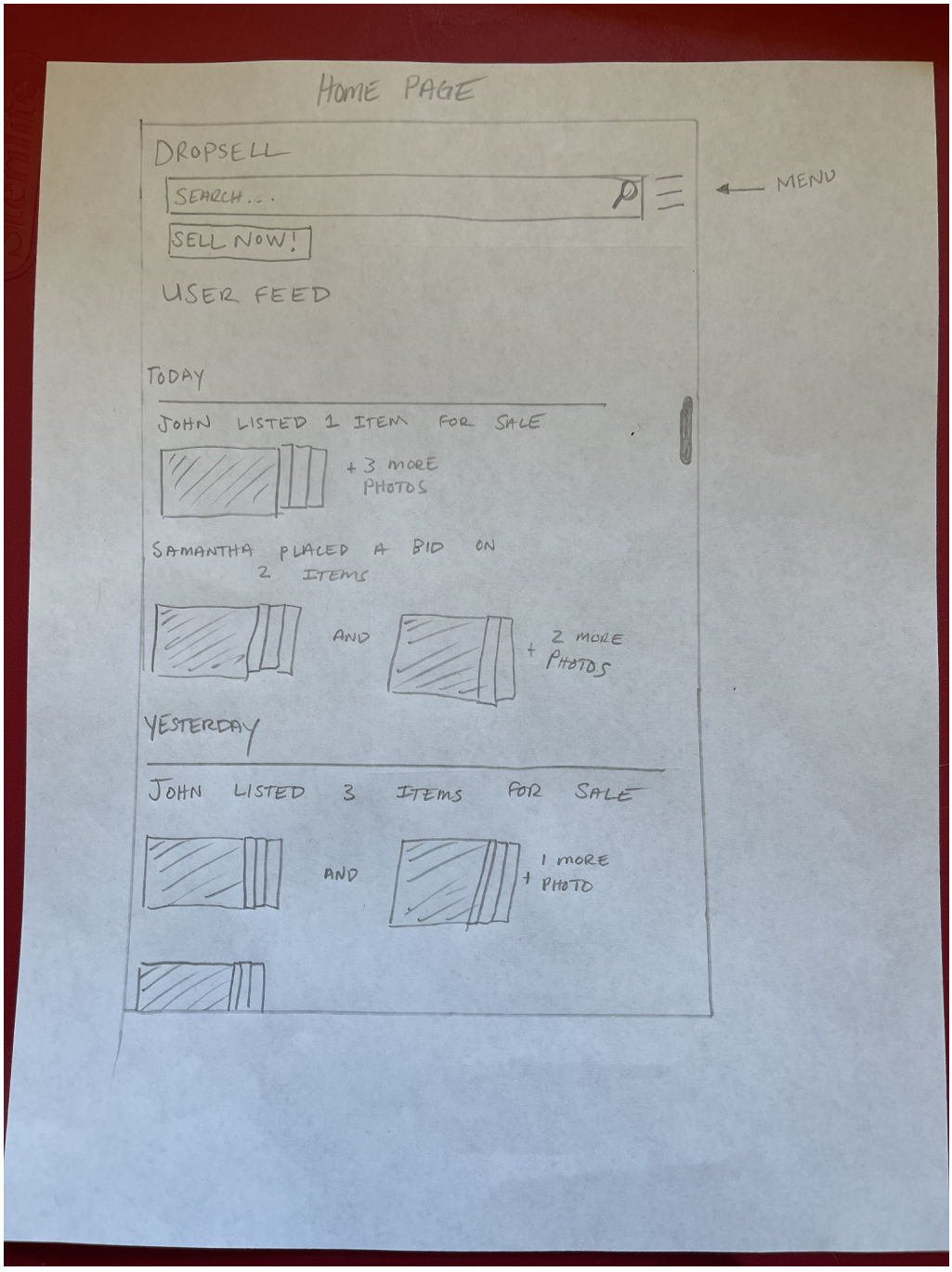
98. Buyers can have an award item.
    a. If a buyer bought some lucky items, then the buyer will have one of the free award items.
99. Birthday promo code / random award item.
    a. If a buyer's birthday is today, then the buyer will choose between a Promo code or an award item from a random item generated algorithm.

Sellers

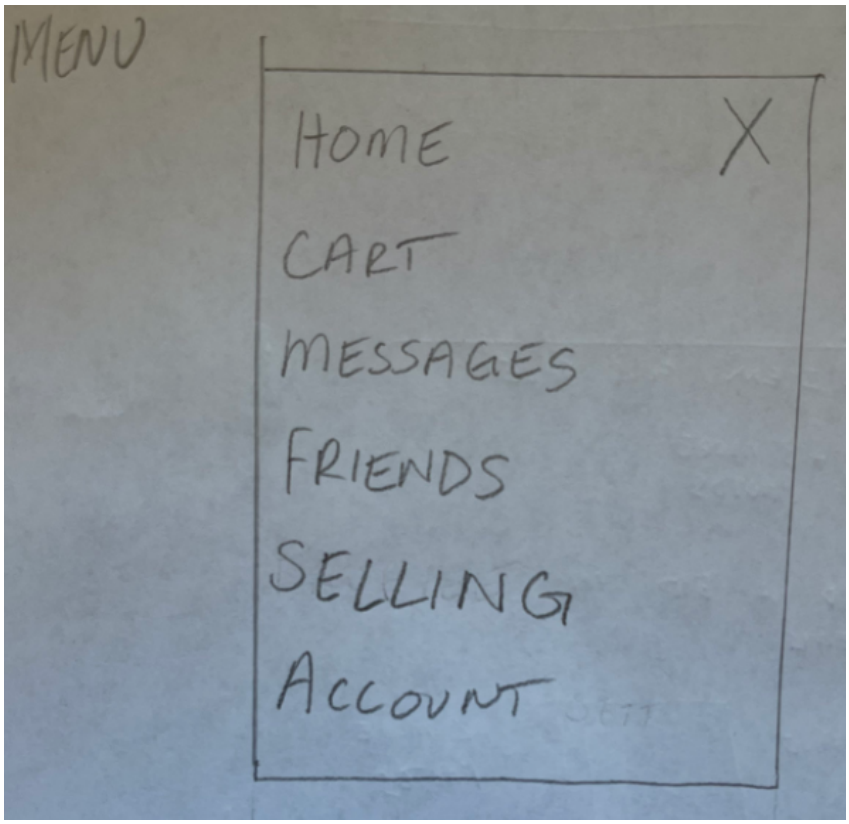
100. Seller background checks for all users intending to sell on the app. This will help to fight against scammers potentially.
101. If one of the items has very good ratings, then the feature can hide ratings.

3. UI Mockups and Storyboards

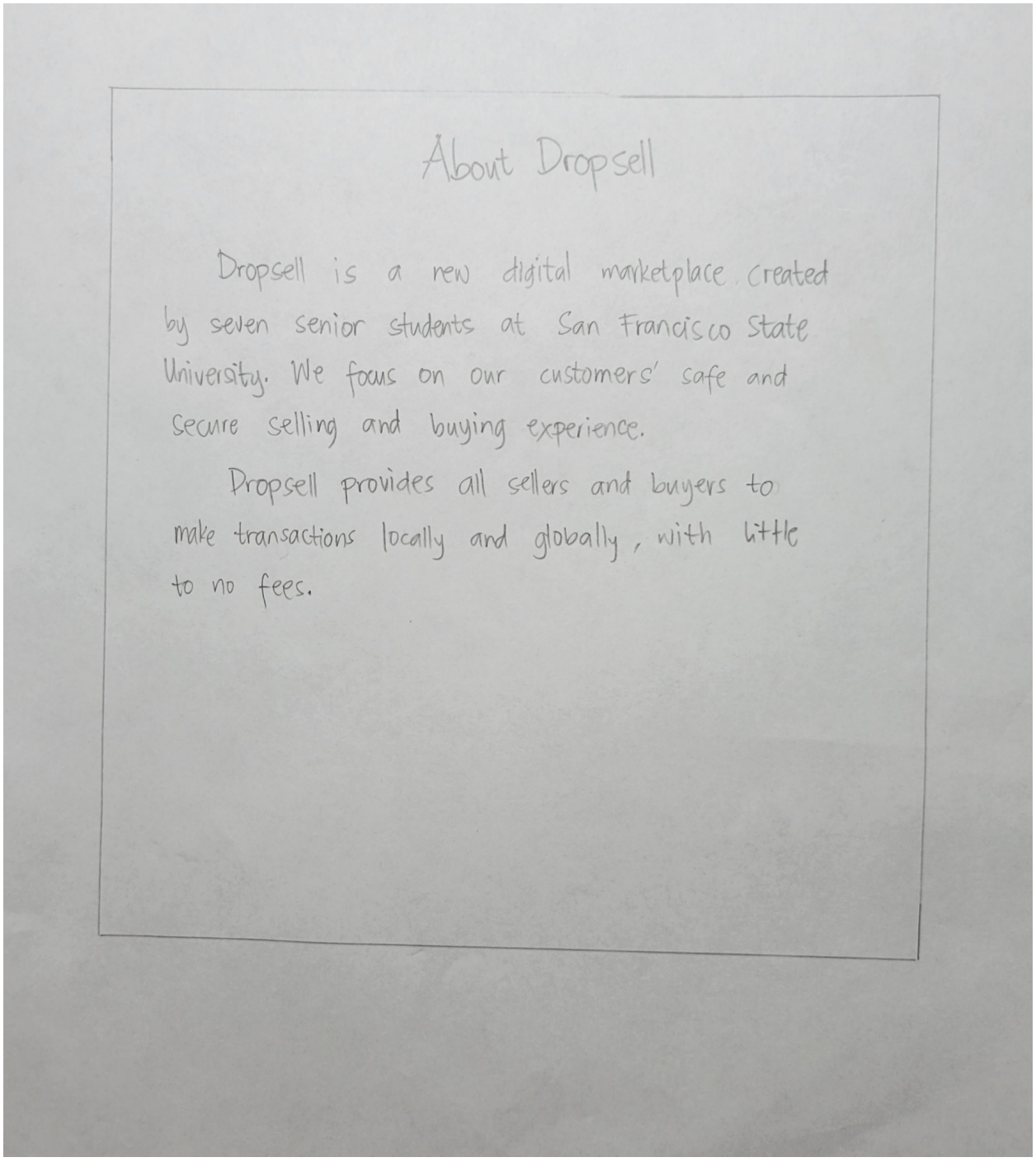
Home Page



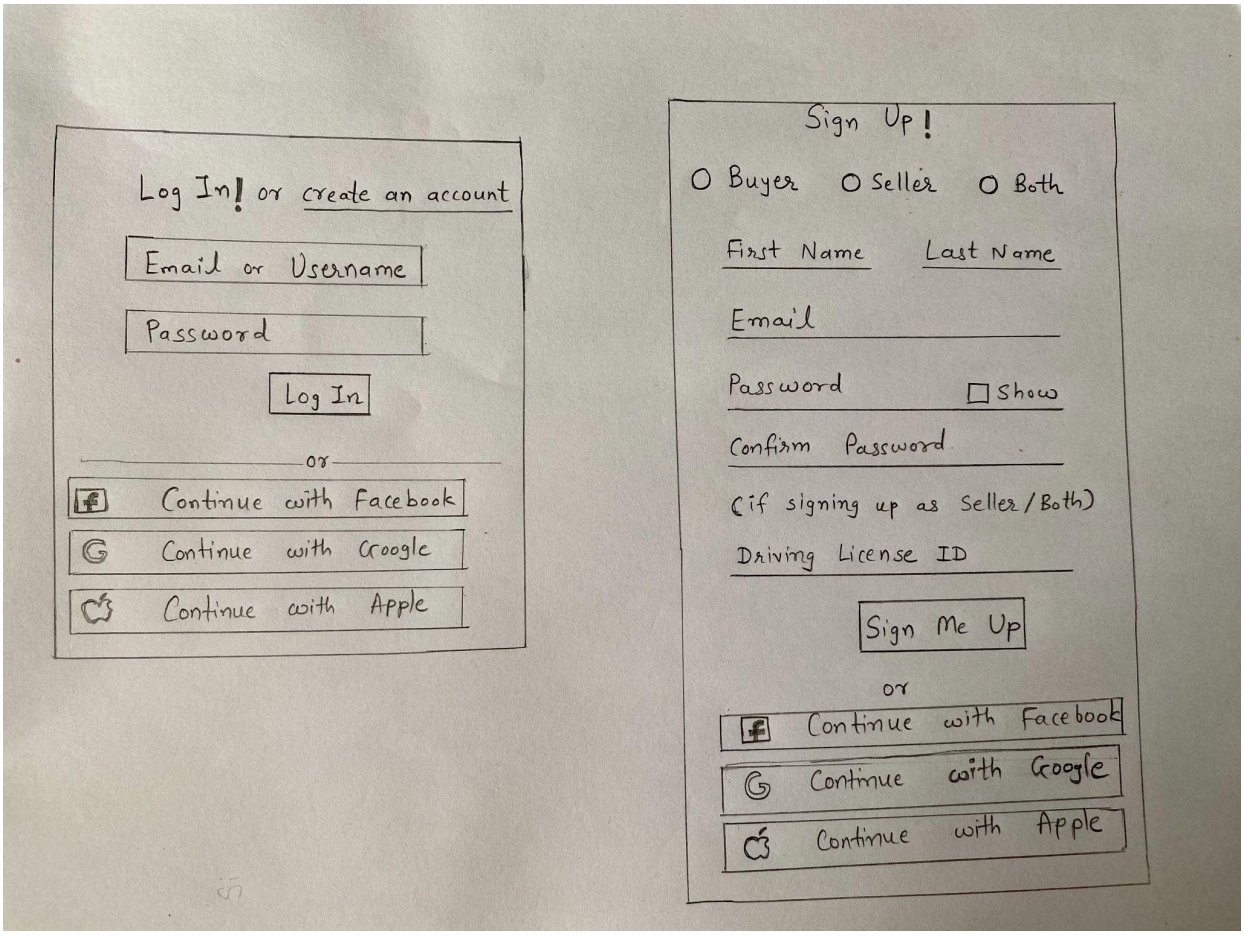
Menu



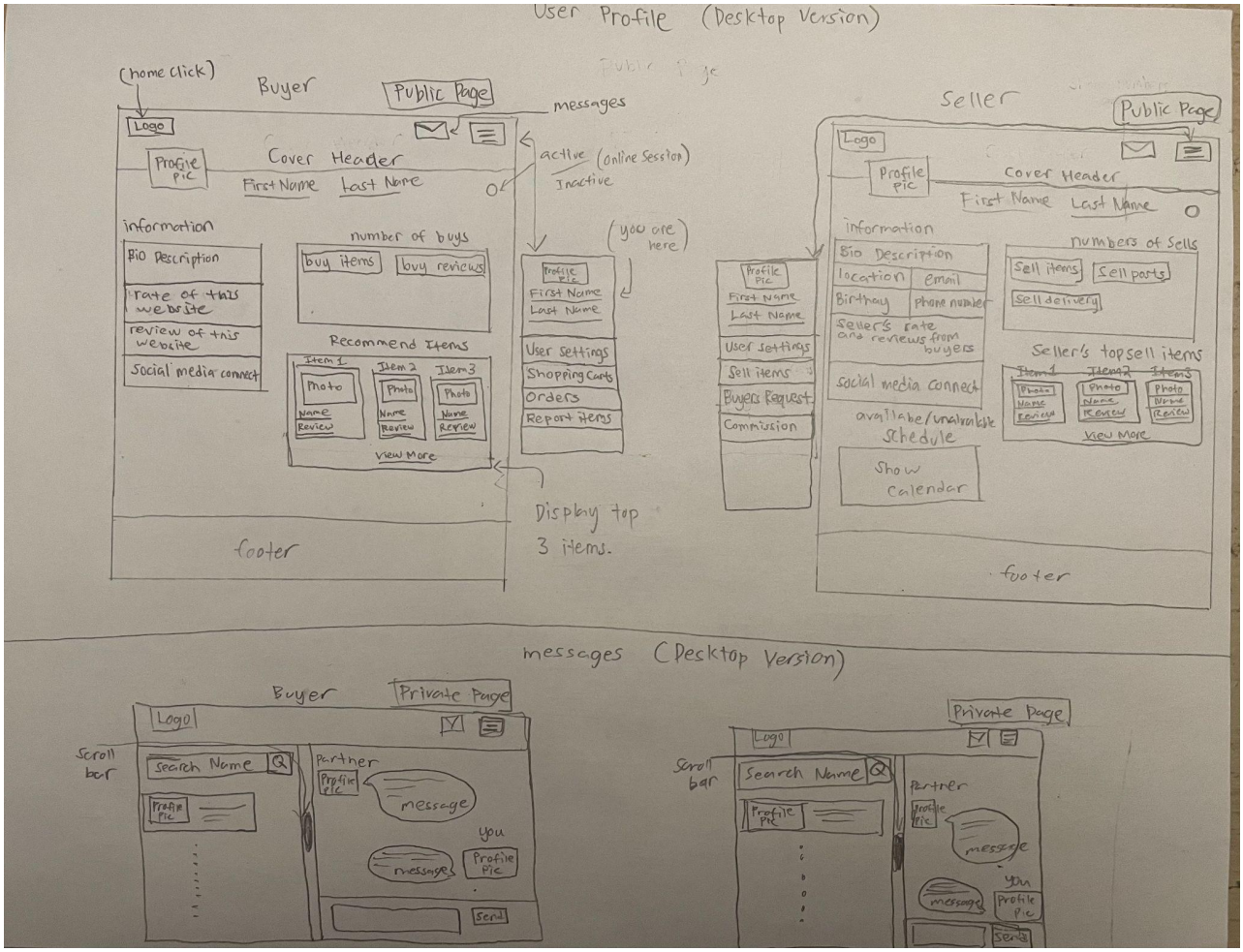
About Page


About Dropsell
Dropsell is a new digital marketplace created by seven senior students at San Francisco State University. We focus on our customers' safe and secure selling and buying experience.
Dropsell provides all sellers and buyers to make transactions locally and globally, with little to no fees.

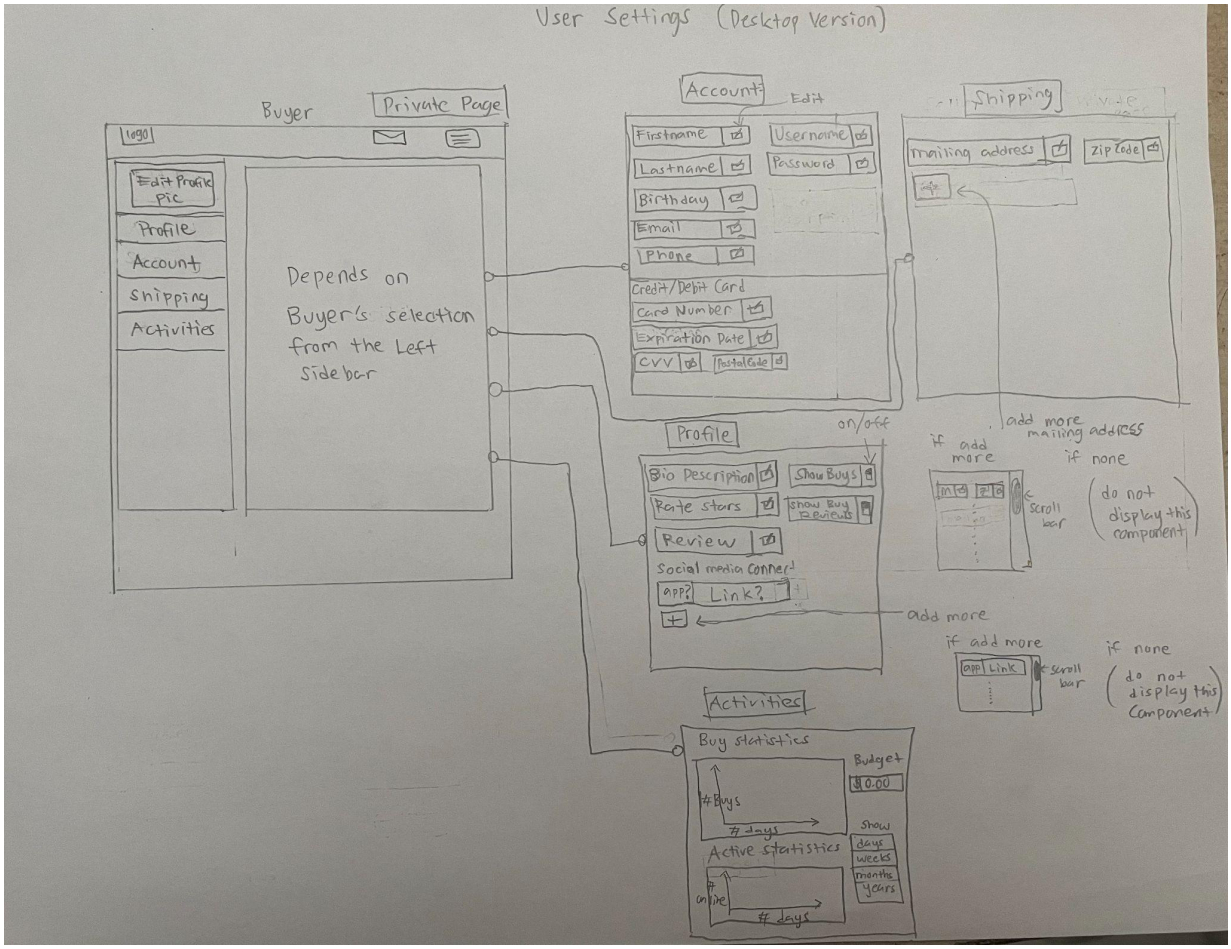
User Sign up / Login



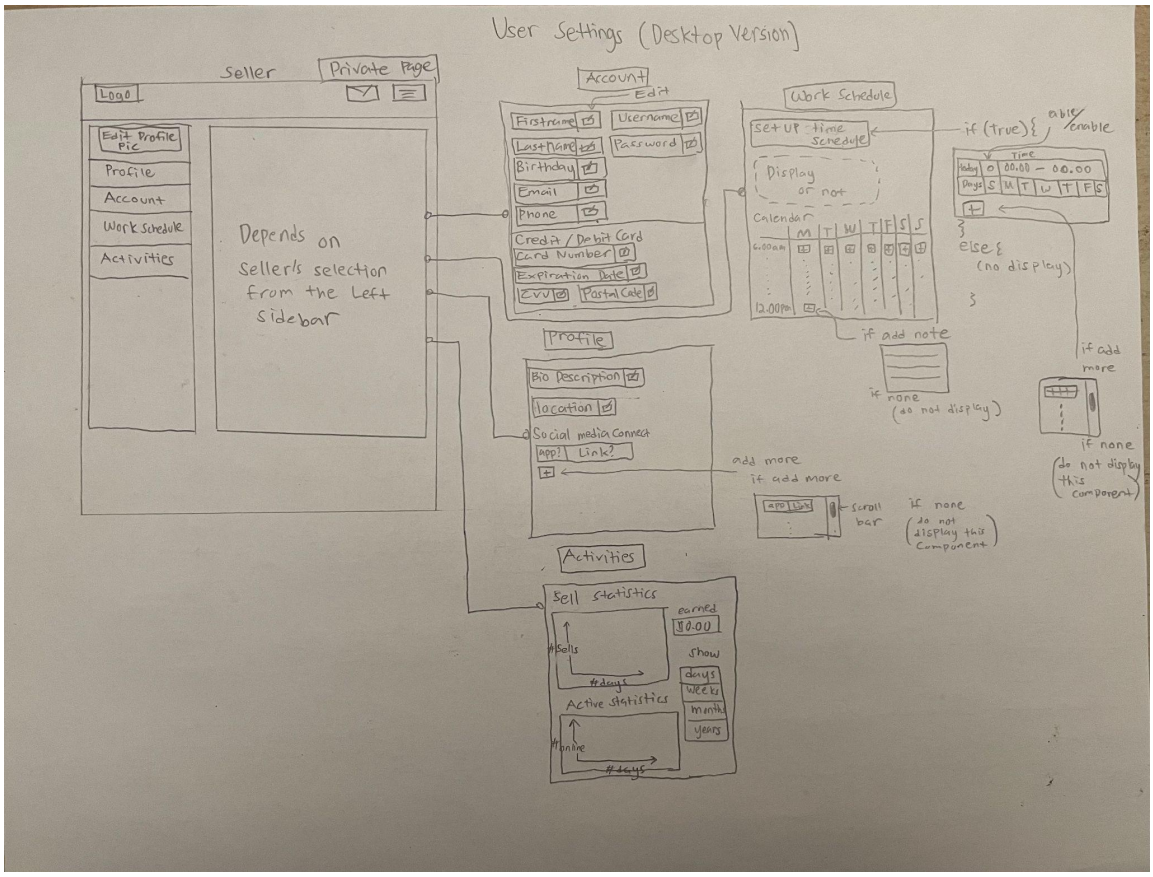
User Profile



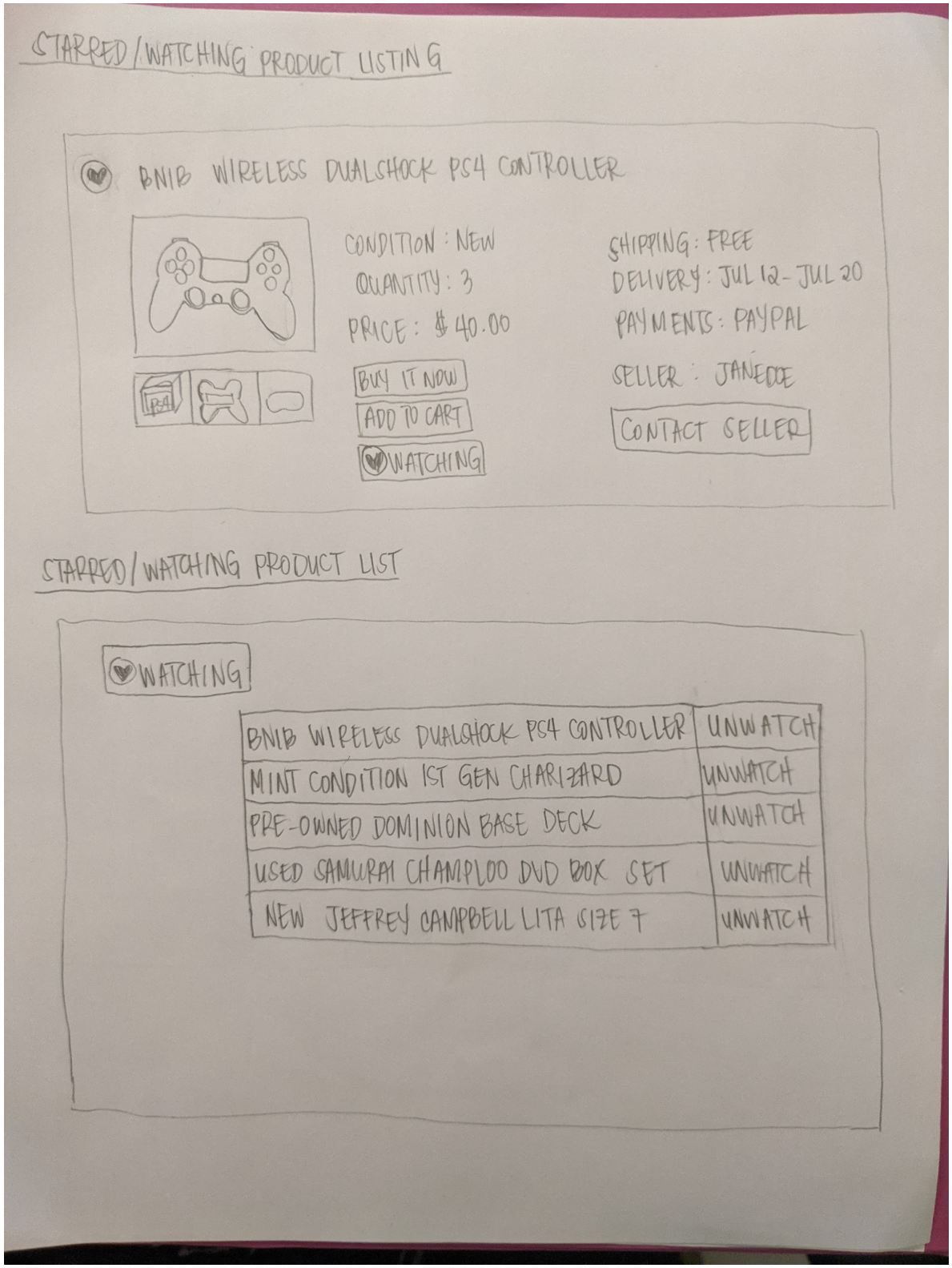
Buyers Settings



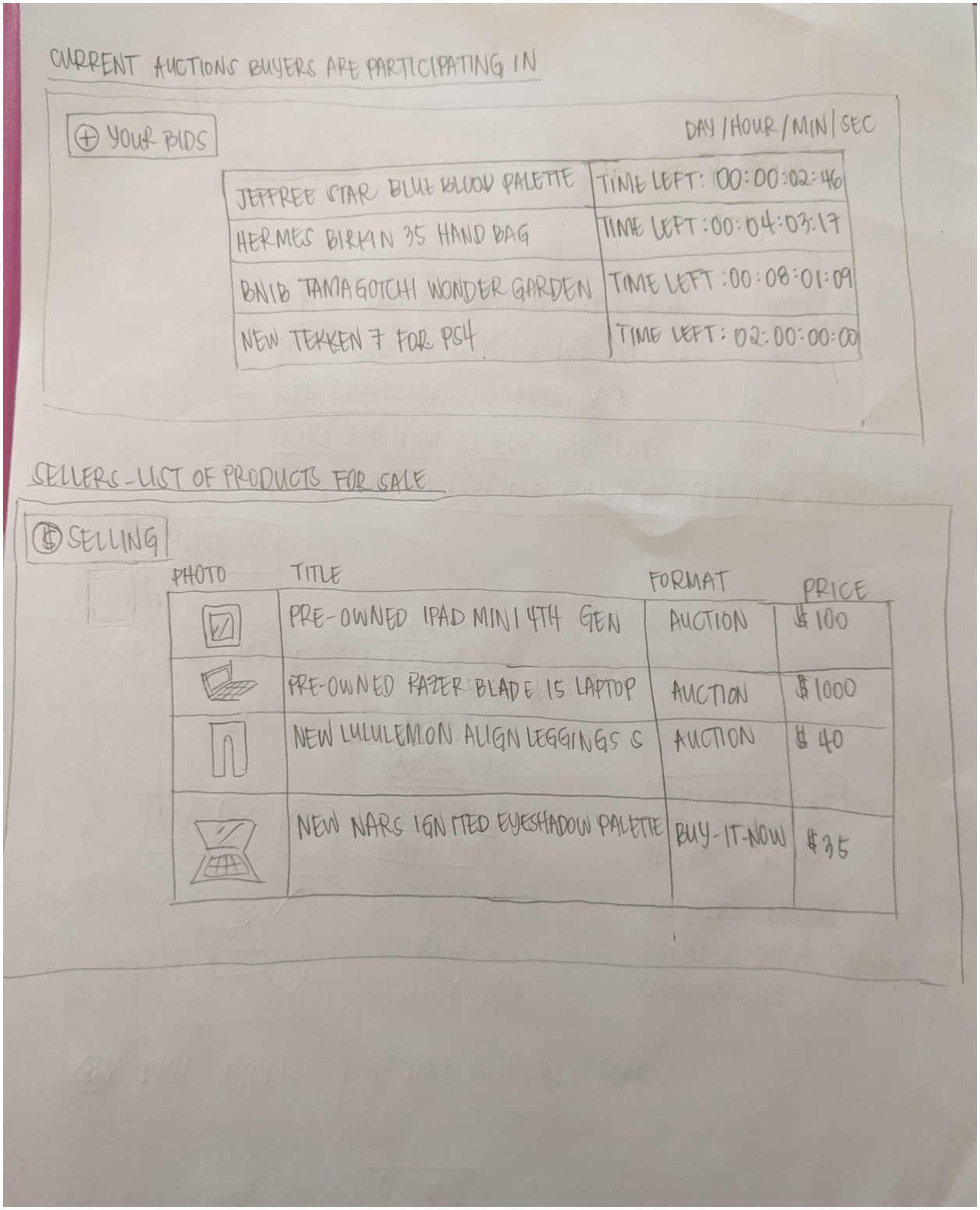
Sellers Settings



Starred/Watching Product Listing, Starred/Watching Product List



List of Auctions and Products for Sale



CURRENT AUCTIONS BUYERS ARE PARTICIPATING IN

⊕ YOUR BIDS

DAY / HOUR / MIN / SEC

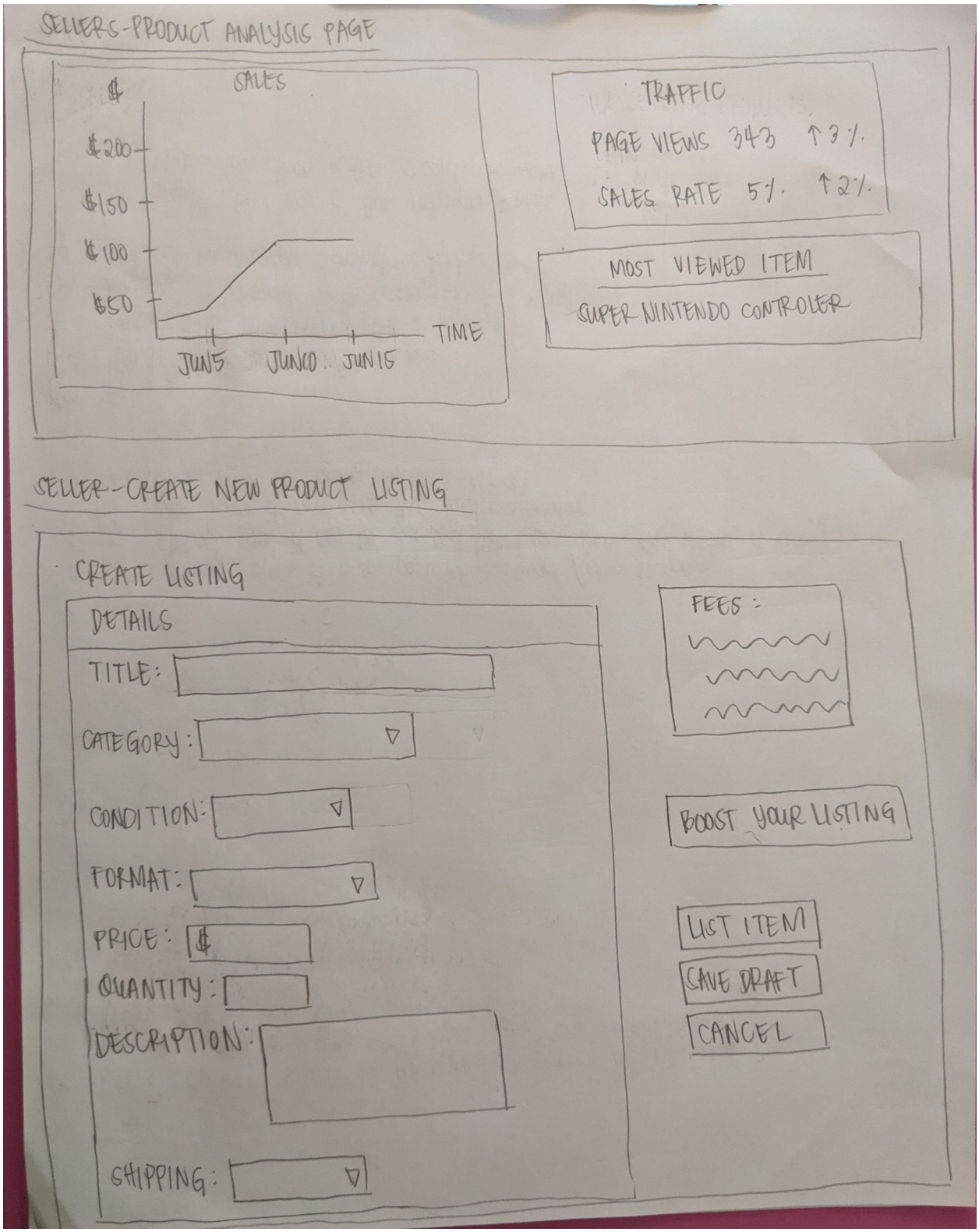
JEFFREE STAR BLUE BLOOD PALETTE	TIME LEFT: 00:00:02:46
HERMES BIRKIN 35 HAND BAG	TIME LEFT: 00:04:03:17
BNIB TAMAGOTCHI WONDER GARDEN	TIME LEFT: 00:08:01:09
NEW TEKKEN 7 FOR PS4	TIME LEFT: 02:00:00:00

SELLERS - LIST OF PRODUCTS FOR SALE

$ SELLING

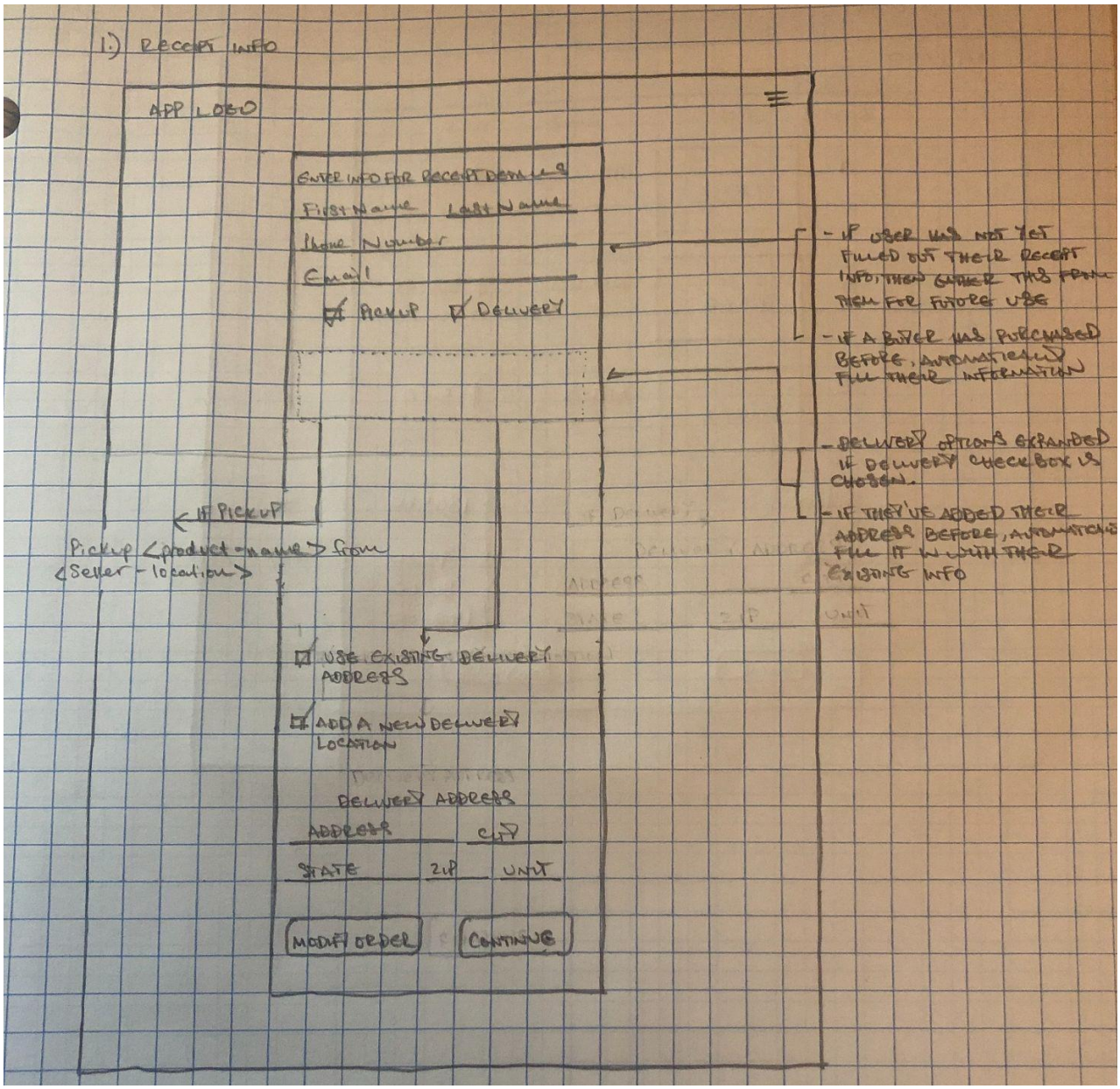
PHOTO	TITLE	FORMAT	PRICE
	PRE-OWNED IPAD MINI 4TH GEN	AUCTION	$ 100
	PRE-OWNED RAZER BLADE 15 LAPTOP	AUCTION	$ 1000
	NEW LULULEMON ALIGN LEGGINGS S	AUCTION	$ 40
	NEW NARS IGNITED EYESHADOW PALETTE	BUY-IT-NOW	$ 35

Seller Analytics and Create Product Listing Page

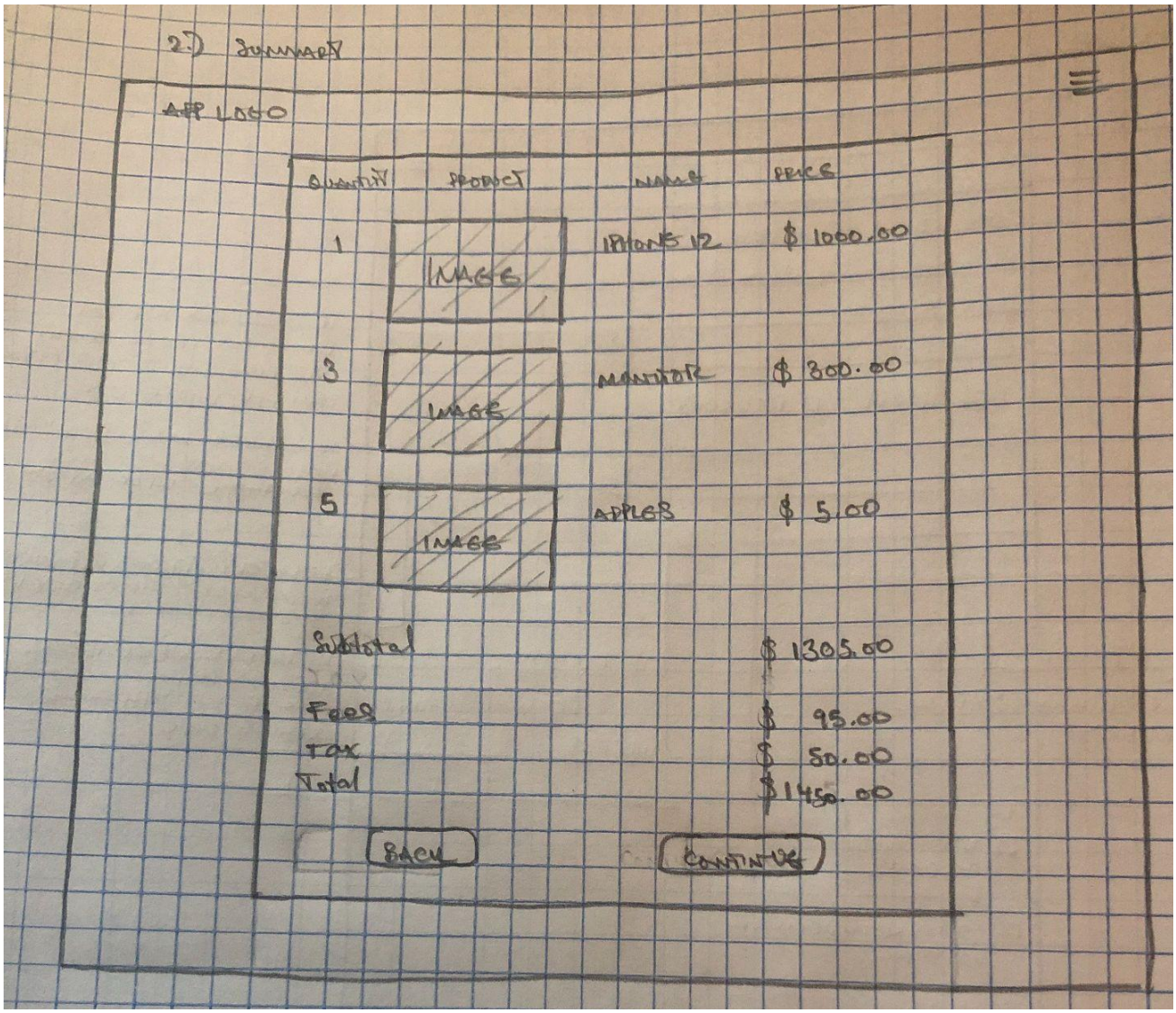


User Checkout Experience

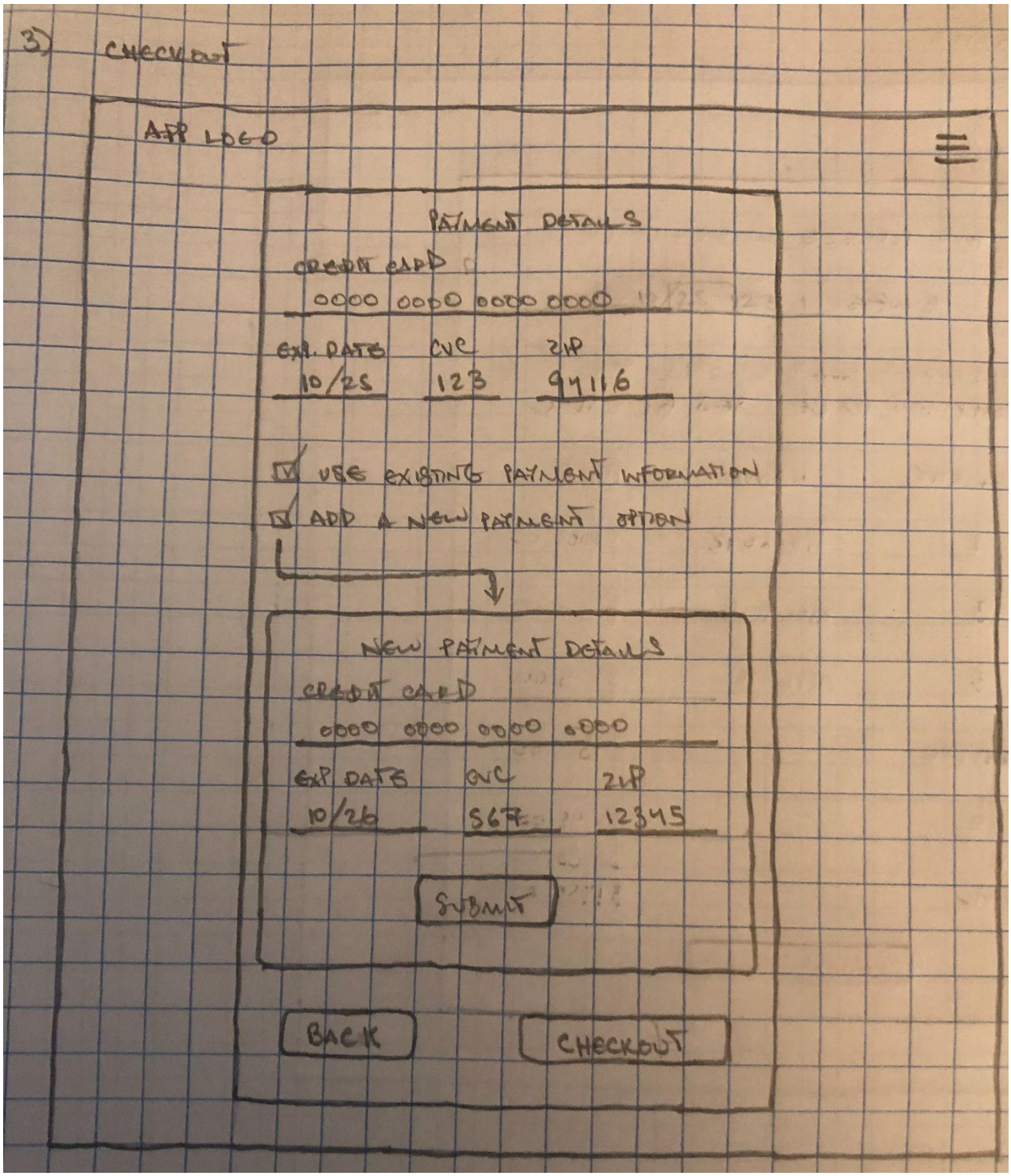
Receipt Info



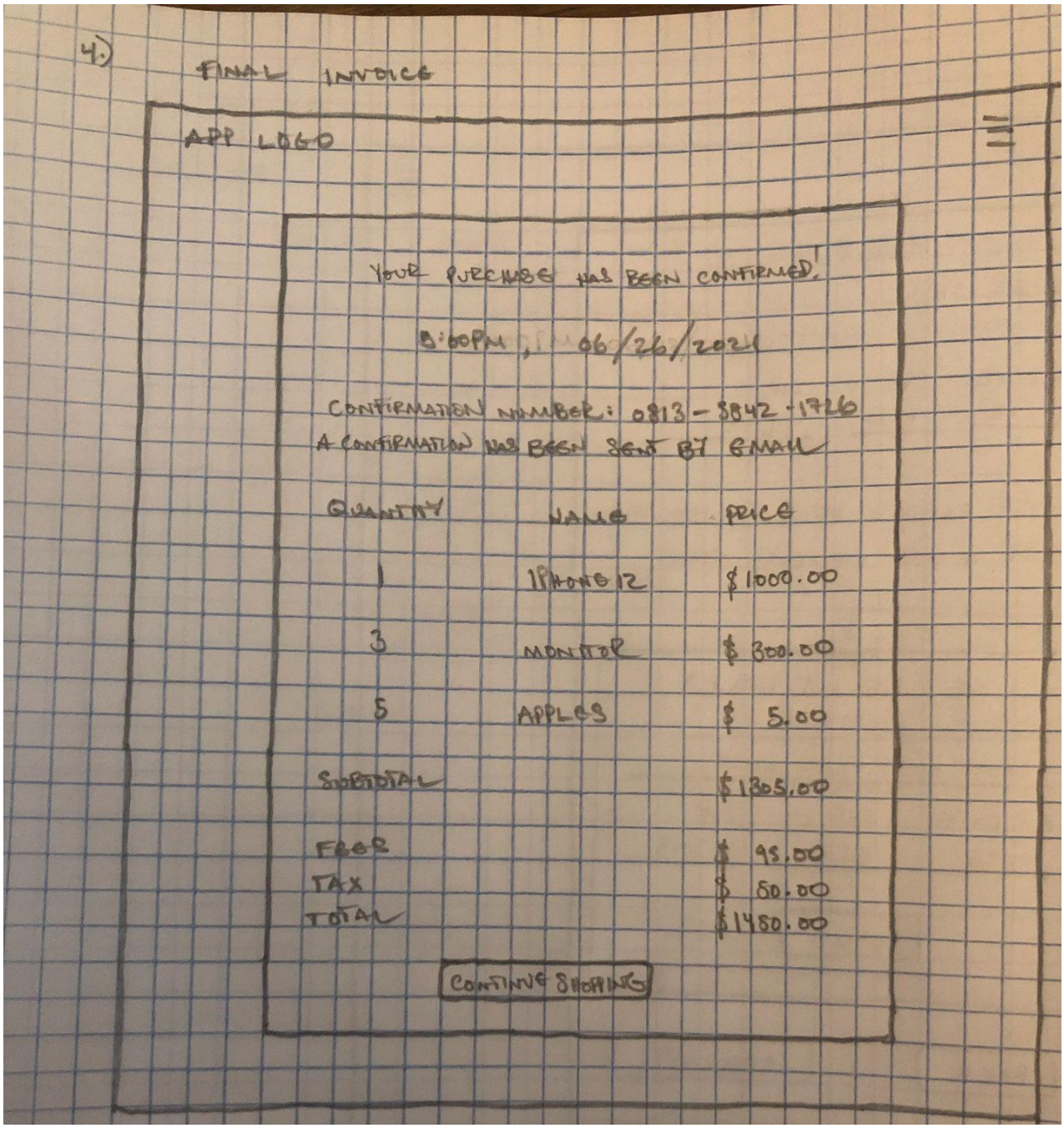
Summary


2.) SUMMARY
APP LOGO
QUANTITY
PRODUCT
NAME
PRICE
1
IMAGE
IPHONE 12
$ 1000.00
3
IMAGE
MONITOR
$ 300.00
5
IMAGE
APPLES
$ 5.00
Subtotal
$ 1305.00
Fees
$ 95.00
Tax
$ 50.00
Total
$ 1450.00
BACK
CONTINUE

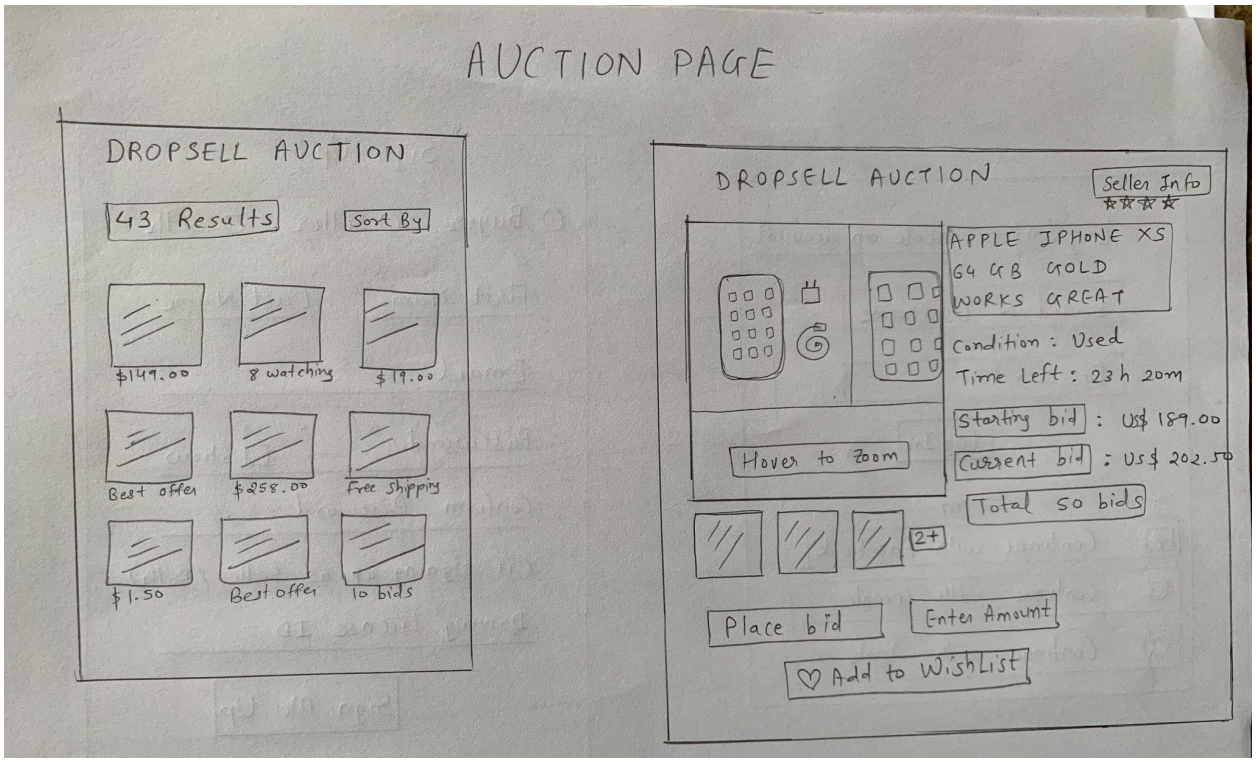
Checkout


3) CHECKOUT
APP LOGO
PAYMENT DETAILS
CREDIT CARD
0000 0000 0000 0000
EXP. DATE
10/25
CVC
123
ZIP
94116
USE EXISTING PAYMENT INFORMATION
ADD A NEW PAYMENT OPTION
NEW PAYMENT DETAILS
CREDIT CARD
0000 0000 0000 0000
EXP DATE
10/26
CVC
567
ZIP
12345
SUBMIT
BACK
CHECKOUT

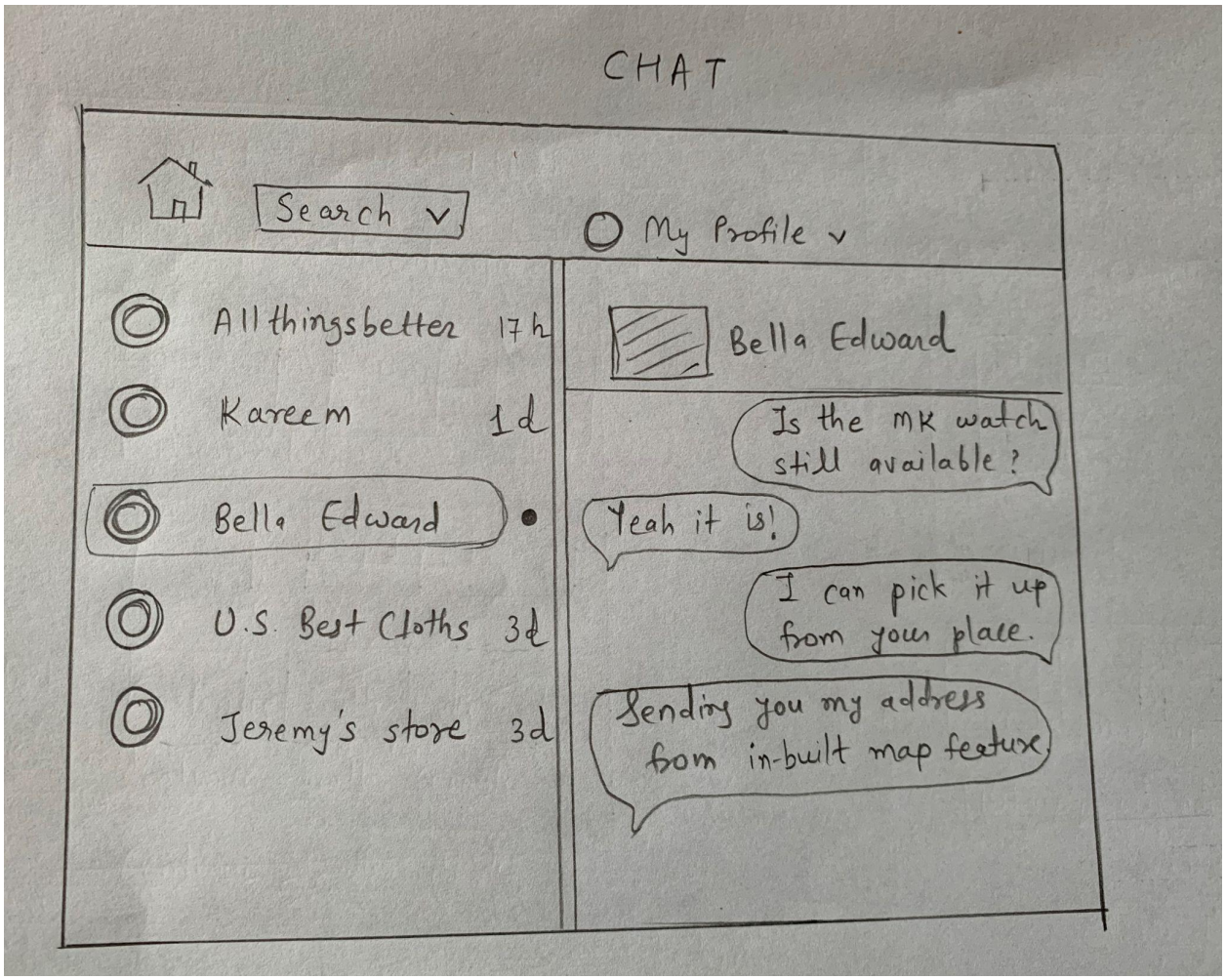
Final Invoice



Auction

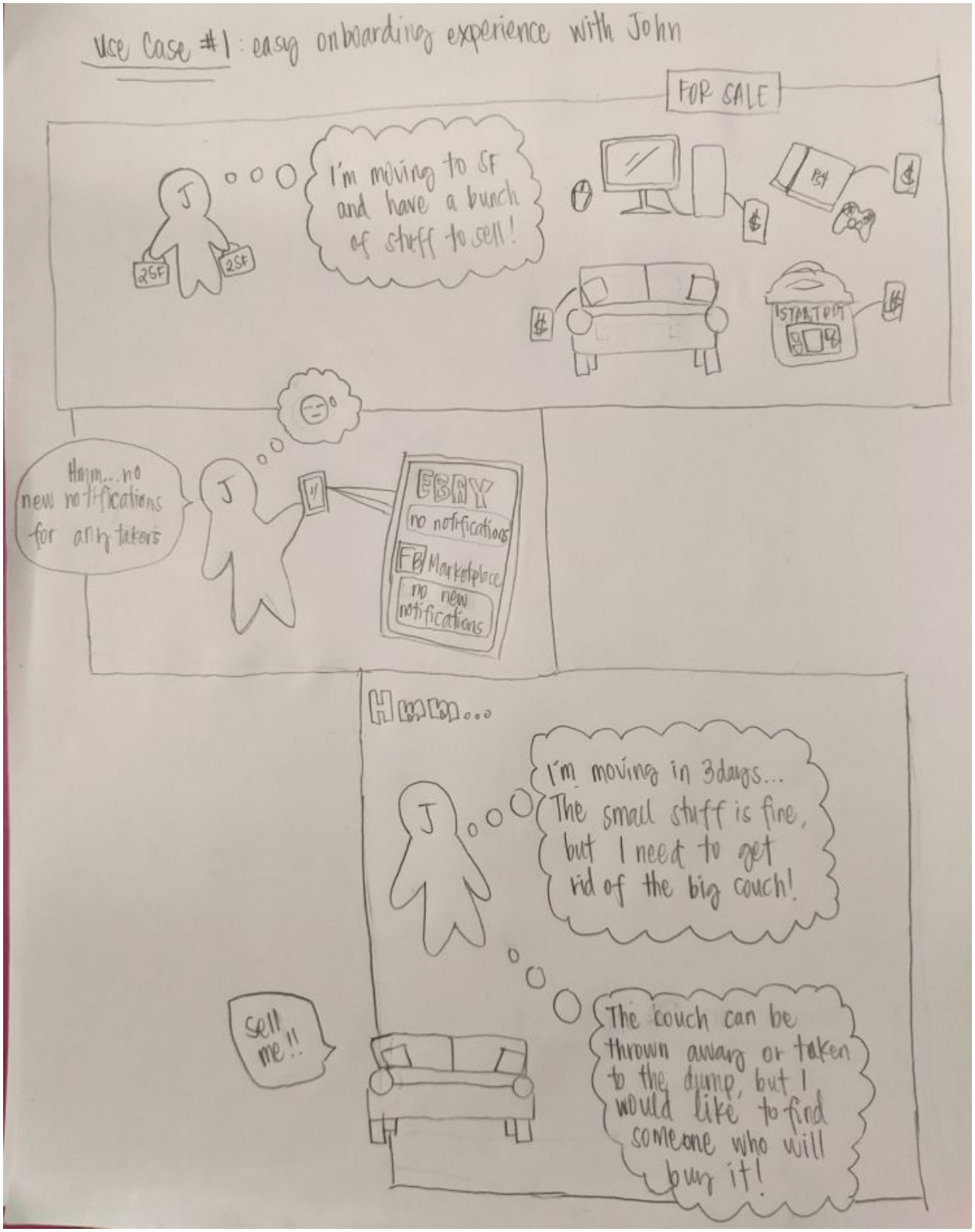

AUCTION PAGE
DROPSELL AUCTION
43 Results
Sort By
$149.00
8 watching
$19.00
Best offer
$258.00
Free shipping
$1.50
Best offer
10 bids
DROPSELL AUCTION
Seller Info
APPLE IPHONE XS
64 GB GOLD
WORKS GREAT
Condition : Used
Time Left : 23h 20m
Starting bid : US$ 189.00
Current bid : US$ 202.50
Total 50 bids
Hover to Zoom
2+
Place bid
Enter Amount
Add to WishList

Buyer/Seller Chat

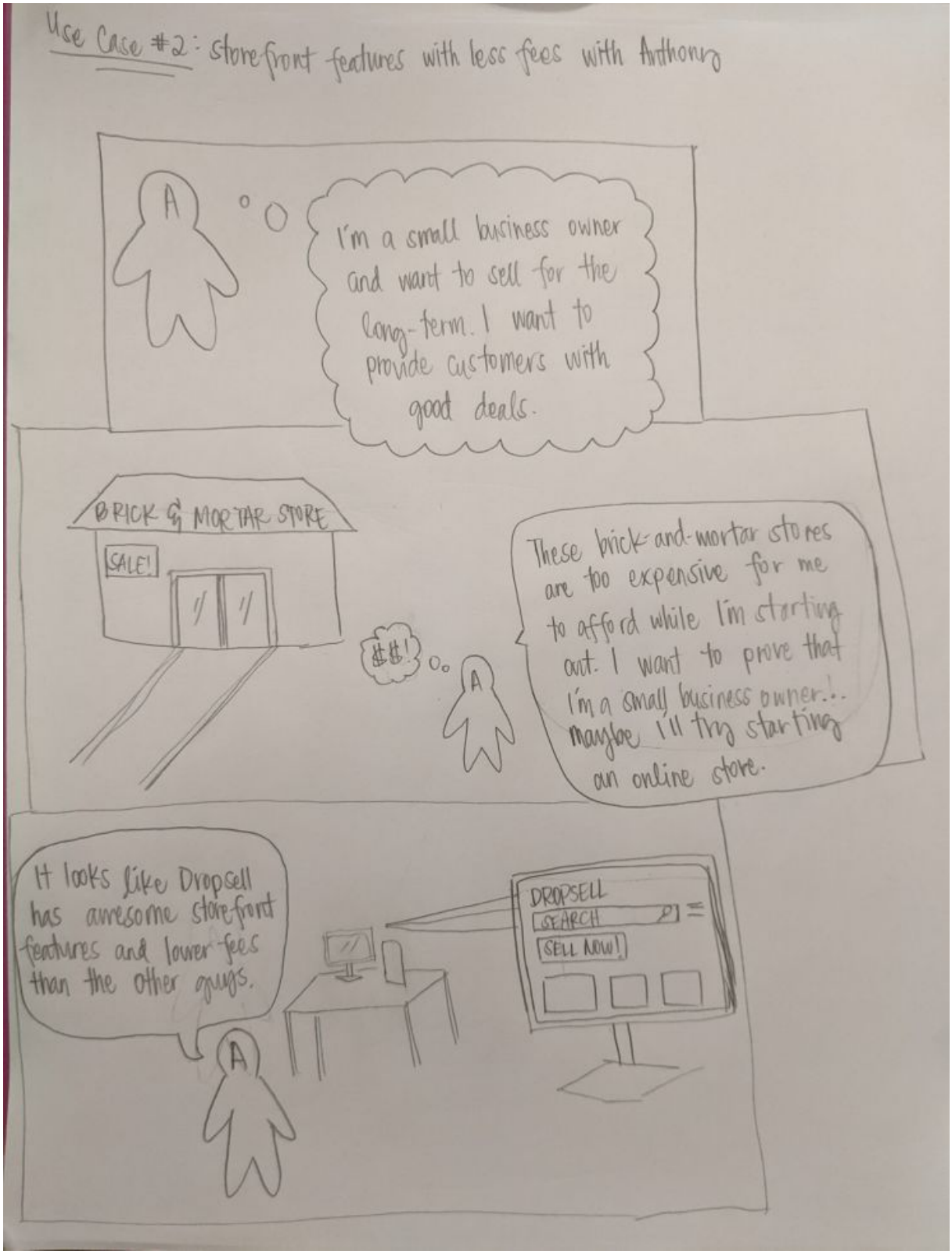


Storyboards

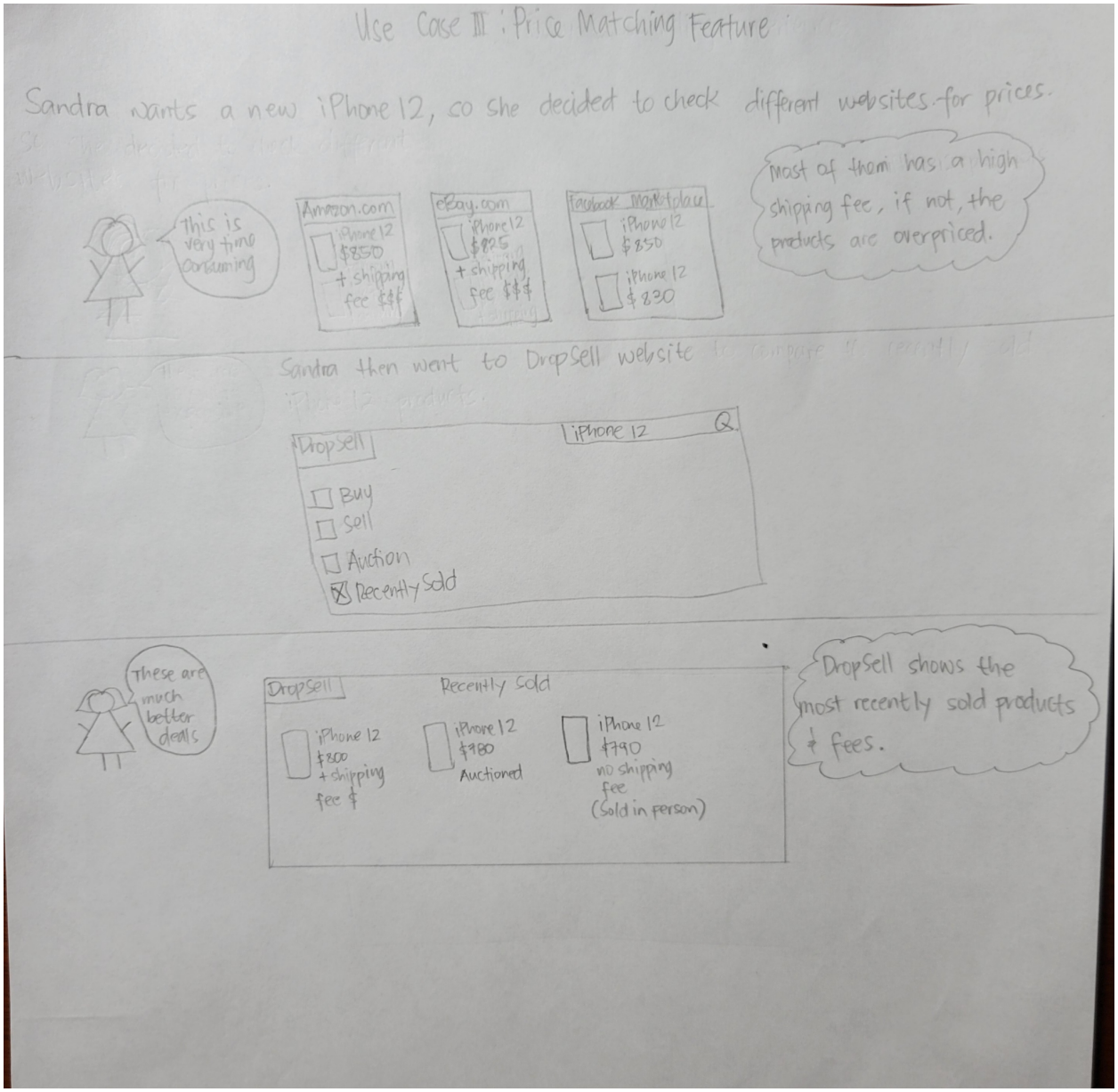
Use Case #1



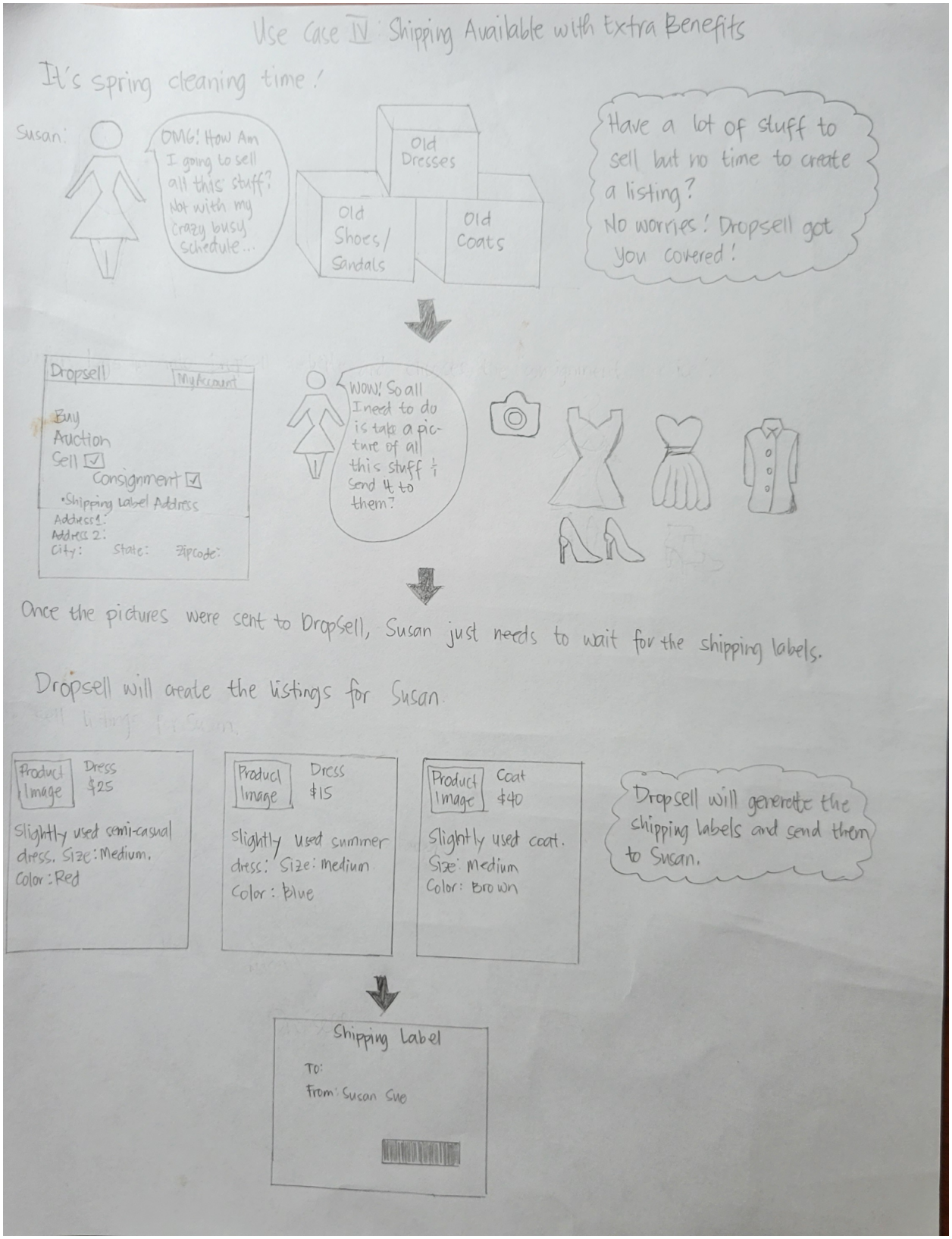
Use case #2


Use Case #2: storefront features with less fees with Anthony
A
I'm a small business owner and want to sell for the long-term. I want to provide customers with good deals.
BRICK & MORTAR STORE
SALE!
#$!
A
These brick-and-mortar stores are too expensive for me to afford while I'm starting out. I want to prove that I'm a small business owner... maybe I'll try starting an online store.
It looks like Dropsell has awesome storefront features and lower fees than the other guys.
A
DROPSELL
SEARCH
SELL NOW!

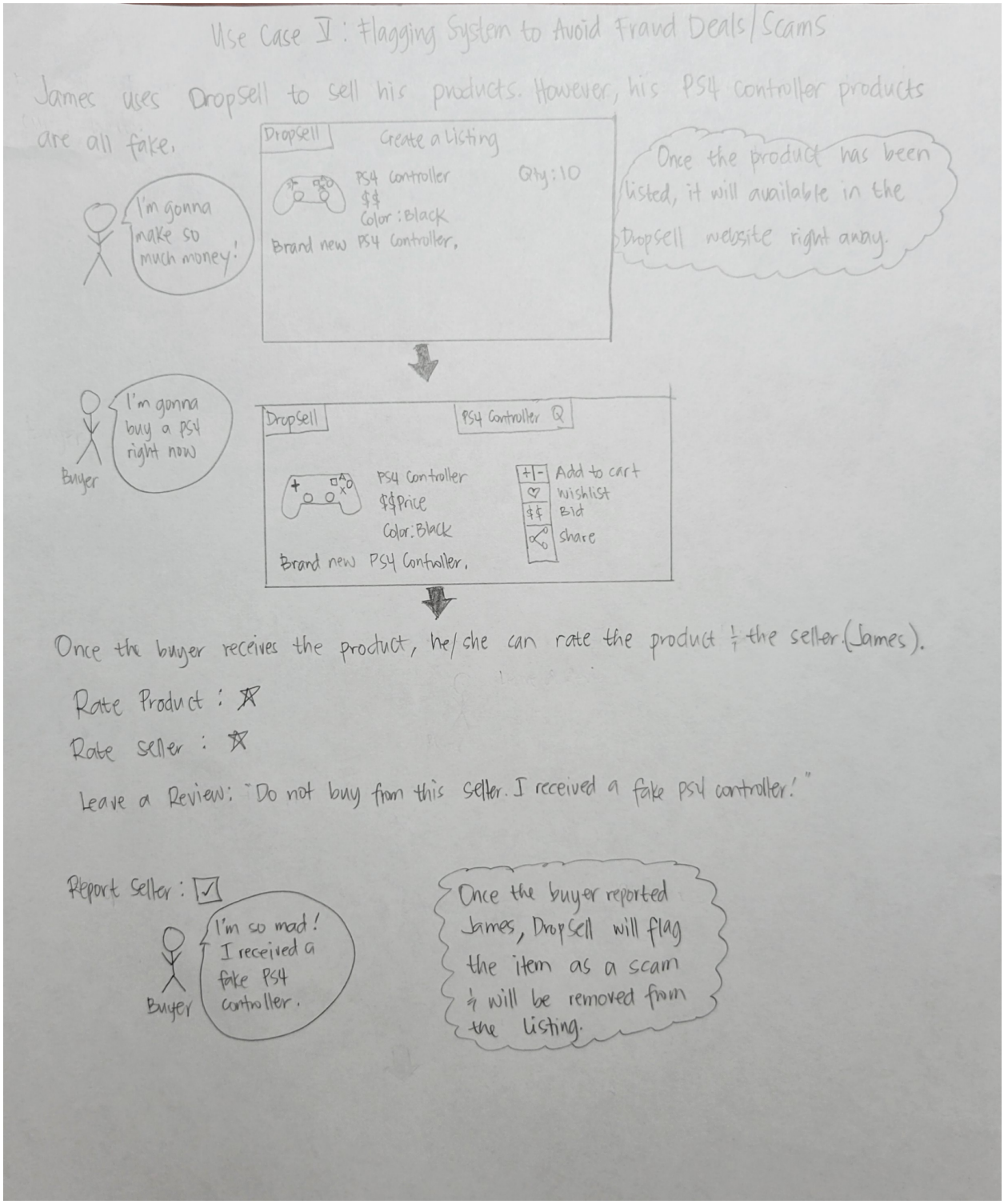
Use case #3


Use Case III : Price Matching Feature
Sandra wants a new iPhone 12, so she decided to check different websites for prices.
this is very time consuming
Amazon.com
iPhone 12
$850
+ shipping fee $$$
eBay.com
iPhone 12
$825
+ shipping fee $$$
Facebook Marketplace
iPhone 12
$850
iPhone 12
$830
Most of them has a high shipping fee, if not, the products are overpriced.
Sandra then went to DropSell website to compare the recently sold iPhone 12 products.
DropSell
iPhone 12
Buy
Sell
Auction
Recently Sold
These are much better deals
DropSell
Recently Sold
iPhone 12
$800
+ shipping fee $
iPhone 12
$780
Auctioned
iPhone 12
$790
no shipping fee
(Sold in person)
DropSell shows the most recently sold products & fees.

Use case #4


Use Case IV: Shipping Available with Extra Benefits
It's spring cleaning time!
Susan:
OMG! How Am I going to sell all this stuff? Not with my crazy busy schedule...
Old Dresses
Old Shoes/ Sandals
Old Coats
Have a lot of stuff to sell but no time to create a listing? No worries! Dropsell got you covered!
Dropsell
My Account
Buy
Auction
Sell
Consignment
•Shipping Label Address
Address 1:
Address 2:
City: State: Zipcode:
WOW! So all I need to do is take a picture of all this stuff & send it to them?
Once the pictures were sent to Dropsell, Susan just needs to wait for the shipping labels.
Dropsell will create the listings for Susan.
Product Image
Dress $25
Slightly used semi-casual dress. Size: Medium. Color: Red
Product Image
Dress $15
Slightly used summer dress: Size: Medium Color: Blue
Product Image
Coat $40
Slightly used coat. Size: Medium Color: Brown
Dropsell will generate the shipping labels and send them to Susan.
Shipping Label
To:
From: Susan Sue

Use case #5


Use Case V: Flagging System to Avoid Fraud Deals/Scams
James uses Dropsell to sell his products. However, his PS4 controller products are all fake.
I'm gonna make so much money!
Dropsell
Create a Listing
PS4 Controller
Qty: 10
$$
Color: Black
Brand new PS4 Controller.
Once the product has been listed, it will available in the Dropsell website right away.
Buyer
I'm gonna buy a PS4 right now
Dropsell
PS4 Controller Q
PS4 Controller
$$Price
Color: Black
Brand new PS4 Controller.
+|-
Add to cart
wishlist
$$
Bid
share
Once the buyer receives the product, he/she can rate the product & the seller (James).
Rate Product : ☆
Rate seller : ☆
Leave a Review: "Do not buy from this seller. I received a fake ps4 controller!"
Report Seller : ☑
Buyer
I'm so mad! I received a fake PS4 controller.
Once the buyer reported James, Dropsell will flag the item as a scam & will be removed from the listing.

4. High Level Database Architecture and Organization

Business Rules

1. The database should be oriented around the user. By referencing a row inside the user table in mysql, the id column can then be used to join other tables back to the user. As a result, we are able to create inheritance relationships.
2. Upon account creation users should be set to the status of buyer, which can be checked in the ‘is_buyer’ column of the user table. Buyers should have access to a shopping cart feature which allows them to securely store their data in the database and as an authenticated user with a valid session.
3. The express-sessions library for node.js is purposefully created so that user sessions are stored in the server’s memory. If you refresh the page, all of the data will be gone. This is a persistence issue, which is why we will have to use express-mysql-sessions to store session data in a mysql table. The schema for sessions should have the columns ‘session_id’, ‘session_expires’, and ‘session_data’.
4. When a user creates an account, their password should be hashed with 10 salt rounds. If they’ve input data which the client and server both validate, then the user’s account is created in the users table. The initial, default columns to fill in are username, encrypted password, the date user was created, is_active (0/1), and buyer, seller or both (0/1).
5. Once a user is a buyer, they can automatically interact with a product feed, profile, and any account settings. They can update their account settings with new values, and also keep track of data such as products sellers publish onto Dropsell.
6. A user may also apply to become a seller, which allows them to publish new products. Once a user becomes a seller, the ‘is_seller’ column for the user will change to 1. Sellers can choose the title, description, price, and image of their products. They can update these values if necessary. We also plan to incorporate product statistics into the seller’s dashboard.
7. Sellers should have ratings, these ratings are provided by buyers who purchase and interact with the seller’s products. Products should also have discussions, ratings, and refunds. Discussions bring a social aspect of the application, where both buyers and sellers can discuss the quality of products posted onto Dropsell. This feature would also include a rating system for products, and if necessary, refunds are an option if user’s decide to return their product. Refunds and payment credentials will be handled by the Stripe.js API.
8. Sellers can decide to either post their products in the marketplace or put their products up for auction. Within the marketplace, buyers can interact with top-purchased and daily deal content. In auctions, sellers can set the starting bid and duration of their product which buyers can then bid on.

Entities, attributes, relationships, domains

1. Registered user
    a. Attributes
        i. user_id
        ii. username
        iii. email
        iv. password
        v. first_name
        vi. last_name

vii. phone
viii. street
ix. city
x. state
xi. zip
xii. is_active
xiii. created_at
xiv. is_buyer
xv. is_seller

b. Relationships (Other tables)
i. sessions
ii. shopping_cart, shopping_cart_products
iii. conversations, messages
iv. meetups
v. seller_ratings
vi. products
vii. product_comments
viii. product_ratings
ix. product_refunds
x. auction_products
xi. top_purchased_products
xii. daily_deal_products
xiii. shipping_products

2. Sessions
   a. Attributes
      i. session_id
      ii. session_expires
      iii. session_data
   b. Relationships
      i. users
3. Conversations
   a. Attributes
      i. conversation_id
      ii. sending_user_id
      iii. receiving_user_id
   b. Relationships
      i. users
      ii. messages
4. Messages
   a. Attributes
      i. message_id
      ii. conversation_id
      iii. message_timestamp
   b. Relationships
      i. Conversations
5. Meetups
   a. Attributes
      i. meetup_id
      ii. buyer_id

   iii. seller_id
   iv. meetup_time
   v. meetup_location
- b. Relationships
   i. users

6. Shopping cart
   - a. Attributes
      i. shopping_cart_id
      ii. buyer_id
      iii. subtotal
   - b. Relationships
      i. shopping_cart_products
7. Shopping Cart Products
   - a. Attributes
      i. product_id
      ii. shopping_cart_id
      iii. title
      iv. price
      v. quantity
   - b. Relationships
      i. shopping_cart

8. Products
   - a. Attributes
      i. product_id
      ii. seller_id
      iii. title
      iv. description
      v. price
      vi. images
      vii. category
   - b. Relationships
      i. users
      ii. product_comments
      iii. product_ratings
      iv. product_refunds
9. Product comments
   - a. Attributes
      i. product_comment_id
      ii. creator_id
      iii. product_id
      iv. comment_timestamp
      v. comment
   - b. Relationships:
      i. products
      ii. users
10. Product ratings
   - a. Attributes
      i. product_rating_id

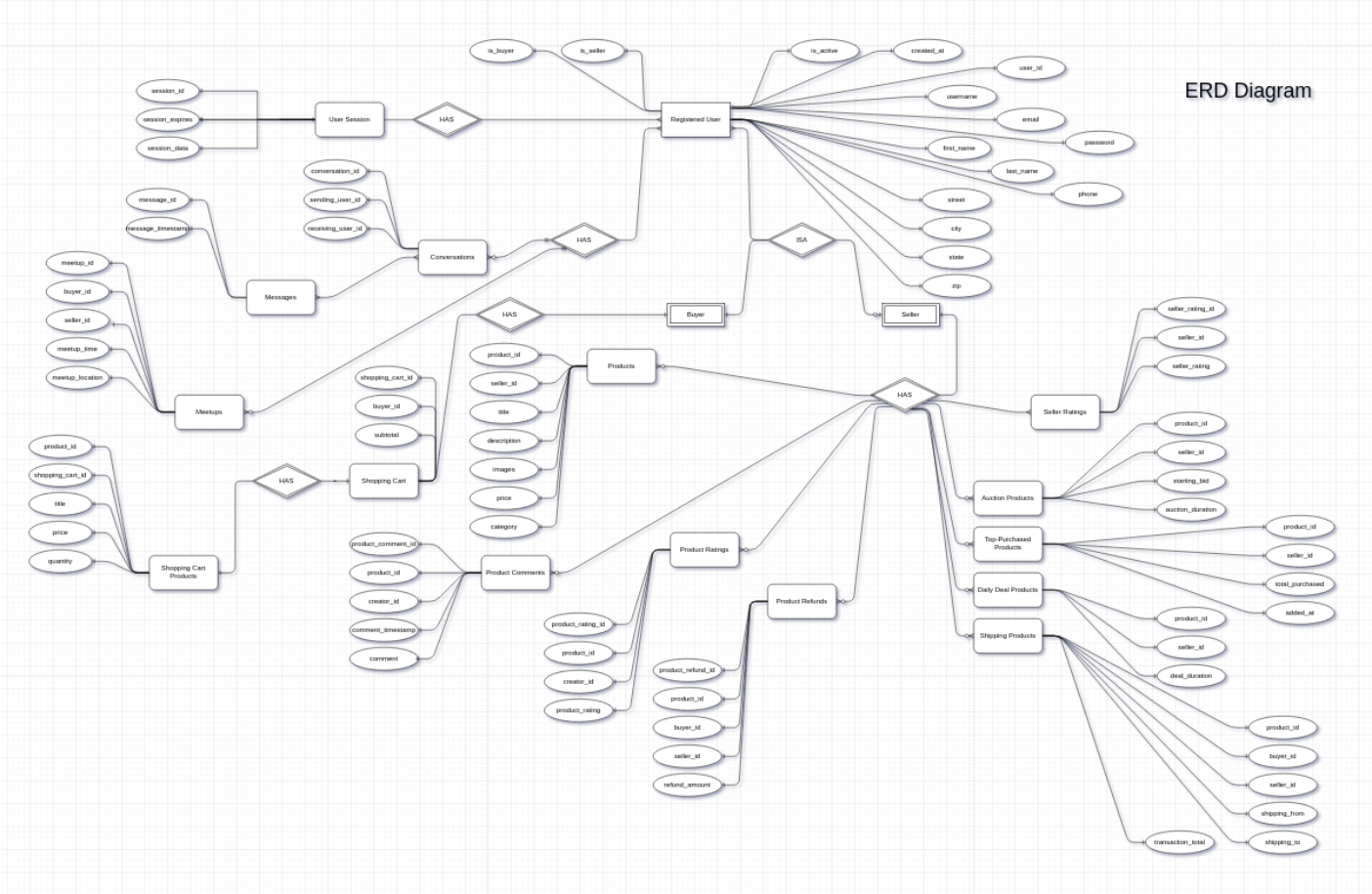
    ii. product_id
    iii. creator_id
    iv. product_rating
  b. Relationships
    i. products
    ii. users
11. Product refunds
  a. Attributes
    i. product_refund_id
    ii. product_id
    iii. buyer_id
    iv. seller_id
    v. refund_amount
  b. Relationships
    i. products
    ii. users
12. Auction products
  a. Attributes
    i. product_id
    ii. seller_id
    iii. starting_bid
    iv. auction_duration
  b. Relationships
    i. products
    ii. users
13. Top-purchased products
  a. Attributes
    i. product_id
    ii. seller_id
    iii. total_purchased
    iv. added_at
  b. Relationships
    i. products
    ii. users
14. Daily deal products
  a. Attributes
    i. product_id
    ii. seller_id
    iii. deal_duration
  b. Relationships
    i. products
    ii. users
15. Shipping products
  a. Attributes
    i. product_id
    ii. buyer_id
    iii. seller_id
    iv. shipping_from
    v. shipping_to

vi. transaction_total

b. Relationships
   i. products
   ii. users

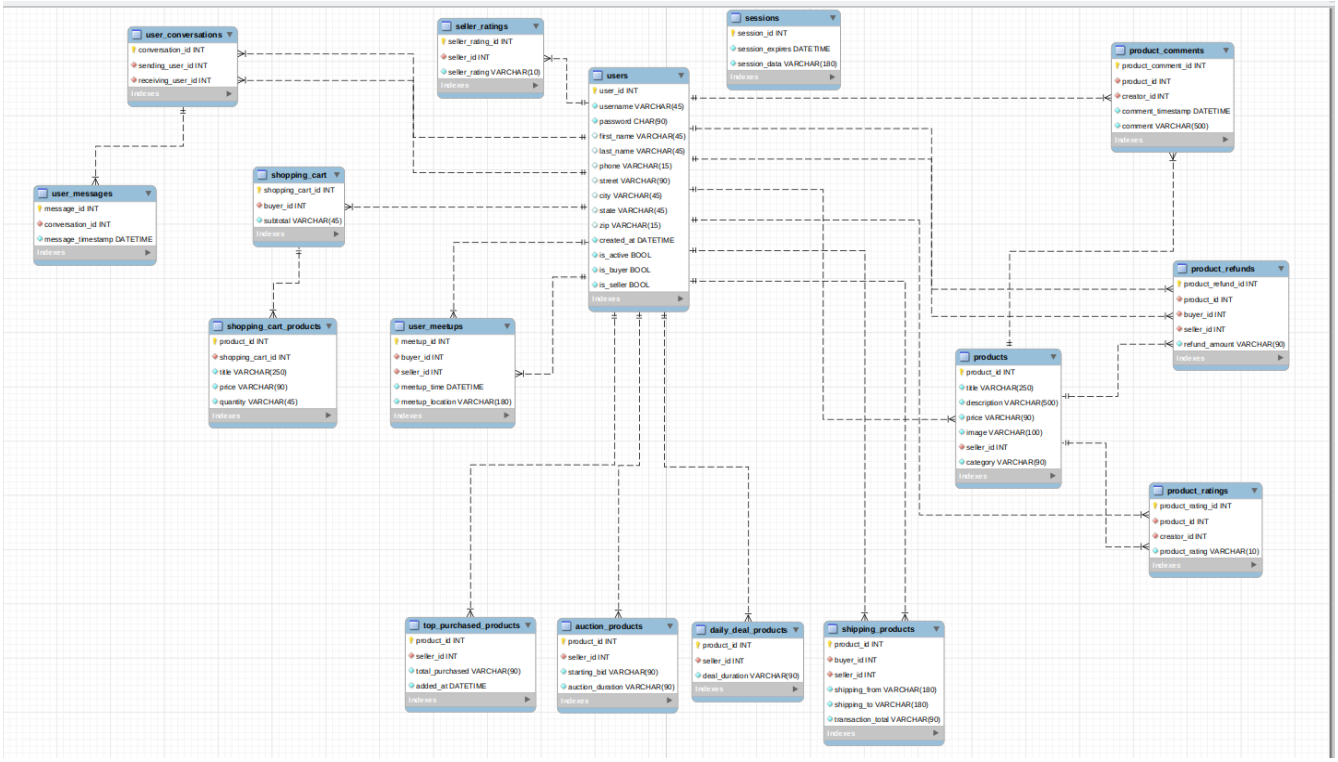
Entity Relationship Diagram

1. https://drive.google.com/file/d/1OmkYbqwzamnWdM2J0xKMSGNLJ2dQBWrc/view?usp=sharing (top of document)



Database Model

1. https://github.com/sfsu-joseo/csc648-848-sw-engineering-sum21-T03/blob/development/application/server/db/mysql_ja_model.mwb



DBMS Decision

1. We will be using MySQL Workbench as our DBMS, since the software comes out of the box with features such as writing custom SQL statements, and creating/managing models and schemas. MySQL Workbench is also an easy-to-use interface which makes creating new tables, setting public/foreign key relationships, and creating new columns simple to explain to other team members.

Media Storage

1. Images and video/audio files will be stored in our project's file system. We will only be storing the image and video/audio file names in our database in order to save space. Additionally, we have decided to store these media files in our project's file system because we can keep them in one consistent location. As a result, this makes lookup times and references to these media files easy and efficient. Since we are storing only the media file names in the database, we can simply prepend the proper path behind the image when we have to display these media files to the client. A good example of prepending the proper path would be with an <img/> tag: src={`/uploads/${product.image}`}. When we load the src attribute, we can add /uploads to the front of the file name and we're done.

Search/filter architecture and implementation

1. Our search algorithm involves the use of a URLSearchParams object which is created when the user clicks the ‘search’ button of the search bar. We have implemented a filterProducts(products, query) function, which takes the current Array of products and the search query parameter to filter with. filterProducts() then returns an updated Array with the products corresponding to the specified search parameter. By default, our Home page will load all of the products saved in the database. This is done with a ‘SELECT * FROM products’ SQL query.
2. In order to implement the category filtering for our search bar, we have implemented an initial array of categories to choose from: Clothes, Shoes, and Electronics. When a user opens the category dropdown menu, they can choose from these options to filter products. When they click one of the category options, an axios request is sent to our Node API which takes the category they clicked as a query parameter. On the backend, the ‘/api/product-categories’ route will fetch all products from the database which correspond to the category the user clicked on. This is done with a ‘SELECT * FROM products WHERE category = ?” statement, where ? is filled in with the specified category.
3. Products will be created dynamically inside our ProductCreationForm.js component. With this component, users can specify the title, description, price, category, and image of their new product. After a user creates a new product, the product will be displayed on the Home page along with any other products.

5. High Level APIs and Main Algorithms

a. API's
   i. Creation of accounts(adding to DB)
      1. The user inputs all of their new account information, such as: username, email, password, and confirm password.
      2. Once the user passes all of the client-side validation, they can then register their account with the credentials they have provided.
      3. When the user clicks the register button, this will trigger a post request to our Node API server. The user's credentials will be sent in this post request using the request body object.
      4. After the post request is sent to the server, there is server-side validation performed, which checks that all of the user's credentials are valid.
      5. After the user's credentials are valid, this data will be added into an async/await call or a Promise object which will trigger the SQL statement which adds the new user into the database.
      6. If there were no errors when creating the new user, then the user should be directed to the login screen to login.
   ii. Logging into accounts(accessing DB/verification)
      1. Input username/password, after clicking login then send post request to node api server, check to see if the user exists in the database, if it doesn't return error message to user, if it does, create a new session for the user, and then redirect them to their Home page
   iii. Similar products
      1. When a user is browsing the application, store a history of the last 10 products they viewed. Use this information to display suggestions to the user.
   iv. Buyers reviews, ratings.
      1. When users create reviews for products that review is stored in the database and linked to the appropriate item
      2. Ratings are based on a 5 star system.
         a. As a registered user rates another registered user they leave a review that gets stored in the database and linked to the registered user.
         b. Registered users can also leave a rating between 1-5 for that registered user that will be stored in the database linked to the appropriate user.
         c. All reviews and ratings will be attached to profiles and when any user loads a profile this information will be retrieved from the database and displayed.
   v. Shopping cart
      1. When a user adds an item to their shopping cart that item is reserved for them for a period of 10 minutes(can be adjusted or tied to a user's session).
         a. OPTION A, when placed in a shopping cart that item is reserved to that user for a period of time so that another user can't accidently buy the product before them if they check out faster.
         b. OPTION B, we inform the user how many users concurrently have that item in their shopping cart and establish(and

importantly inform the user) that their item is not secured until they start the actual check out/payment process. This means another user can accidently steal another usersitem if they check out first.

vi. Messaging.
    1. Array of objects, with each object containing the ID of the message poster, ID of the message itself, and the time the message was sent.

vii. Tag system/price checking
    1. When an item is created the seller will enter all appropriate tags to the item in a fashion similar to hashtags. For example a PS5 might have the tags; #Electronics,#Game,#Sony(# used for reference).
    2. Using tags we can program different algorithms such as daily deals, categories, price checking.
    3. This tag will be stored with the item in the database,
    4. Users must enter a minimum number of tags.

viii. Listing products and retrieving product information: title, description, image, price, brand, condition(new/used/etc), misc/notes, and importantly any tags that apply to the product.
    1. Creating a new product, after checking that all required sections are filled in(title, description, image, price, brand, condition, misc/notes, and tags.), create a database entry linked to the user.
    2. If the request to see a product is valid, retrieve the product information and display it in the appropriate manner(ie seller/buyer history, auction listing, etc)
    3. When a product is created for sale, before submitting the item to the database/market place run the price checking algo and display suggested prices and similar product information.

ix. Buyer Receipt details for delivery or checkout (first name, last name, phone, email)
    1. If user elected for peer-to-peer deliver simply store the transaction information in the users history
    2. If a user opted for a consignment service then, like the peer-to-peer service, the transaction information will be stored in the user's history. Then a shipping label will be provided to the buyer.

x. Inappropriate language detection
    1. Maintain a list of words/phrases that are deemed inappropriate or that violate terms and conditions.
    2. When a user creates content(i.e. comments on a post, leaves review, creates a product, etc) parse the text into tokens and check against our list of inappropriate language.
    3. Handle the inappropriate comment based on our terms/condition.

b. Significant non-trivial Algorithm/process
    i. Price Matching against similar products(Superior service)
        1. Using the Tag api to get product tags.
        2. Compare the tags listed on the product with similar tagged items
            a. When comparing tags we will look at similar items that have been listed in the past(time period to be adjusted, up to 3 months).

   b. Based on the tags of the item compared to current and past tags of similar items we generate a list of information and present that to the user.
   c. This information will show simple statistics of sales related to the item.
   d. Display the picture of the item so the seller can see how relatable their item is to the suggested ones.
3. When a seller posts an item our price checker will display listings that have sold in the past, are currently listed, and products that are not selling and all associated price points of those objects.
4. This price checking allows a new or inexperienced seller to get a feel for the correct amount they should list their item for.
   a. Example 1, a user wants to sell an item fast.
      i. They list the item, our price checking algorithm displays history of sells and current listings
      ii. The user sees that the item averages 100$ over the past 3 months.
      iii. The seller sees that current listings of that item range from 85-90$ and that items priced over 105$ generally stay on the marketplace for 2 weeks.
      iv. Seller values time over efficiency and lists the product for 75$ to make the fastest money.
   b. Example 2, a user wants the most money and is patient,
      i. User has an item sitting around in their storage and wants to make the most money they can, and are willing to wait
      ii. They list the item, our price checking algorithm displays history of sales and current listings.
      iii. The user sees that the item averages 500$ over the past 3 months
      iv. The user sees that over the past 3 months items placed over 575 take an average of 3 weeks-1 month to sell.
      v. The user also sees that items placed over 600$ rarely sell.
      vi. The user is patient and lists the item for 580$ to get the biggest return.
   c. Example 3, a user is new and has an unique item they don't know how to price.
      i. The user has never sold anything second hand before and has a high quality painting they wish to sell.
      ii. They list the item, our price checking algorithm displays history of sells and current listings
      iii. The user sees similar paintings for this artist but does not see the exact match for his painting.
      iv. Based on the information return the user can see an average cost of items of similar nature.
      v. Using this information he knows that if he lists the painting for under 200$ he will most likely be losing

money and that if he lists the painting for over 700$ he may never sell it.

vi. The user lists the painting for 400$.

ii. Ordering reviews and rates

1. This will help to organize reviews and rates

i. When buyers purchase their items, they are able to review and rate their items and post them on a public page.

ii. After their review and rate items end, then it will post them to the public in alphabetical order to make sure it is easy to find and organize for reviews and rates items.

iii. Alerting if most of the items have bad reviews and rates

1. Let the sellers know that their selling items must be satisfied to the buyers.

i. This can have feedback from the buyers, and making sure that when the sellers sell their items, they have to improve and make better items of what they are selling.

2. If the sellers cannot resolve this issue, then let the business management know about this issue.

i. Having bad reviews and rates can cause down business selling items.

ii. Sellers should be very responsible to make sure that their items are good enough.

iv. Detecting inappropriate comments

1. When the buyers put comments on selling products, this algorithm will avoid inappropriate comments as possible.

i. This is useful because when young children visit this site, then they should NOT read any inappropriate comments.

2. Use API Calls from the list of inappropriate words, to see if one or more words matches the comments, then the comments should not be allowed to post them in public.

v. Recommend best meetup time

1. When the buyers and sellers decide to schedule meetup times, then recommend the best meetup times will show the best of what days and times to meet.

i. Just in case the buyers or sellers are too lazy to set up their scheduled meetings. They can just click “Show best meeting times”.

2. Use a graph algorithm to optimize the best meetup times between the seller and buyer by analyzing their schedule planner.

vi. Calculate the mean and standard deviation of the number of selling, buying, refund, and return products.

1. Keeping track of a fair number of selling, buying, refund, and return products.

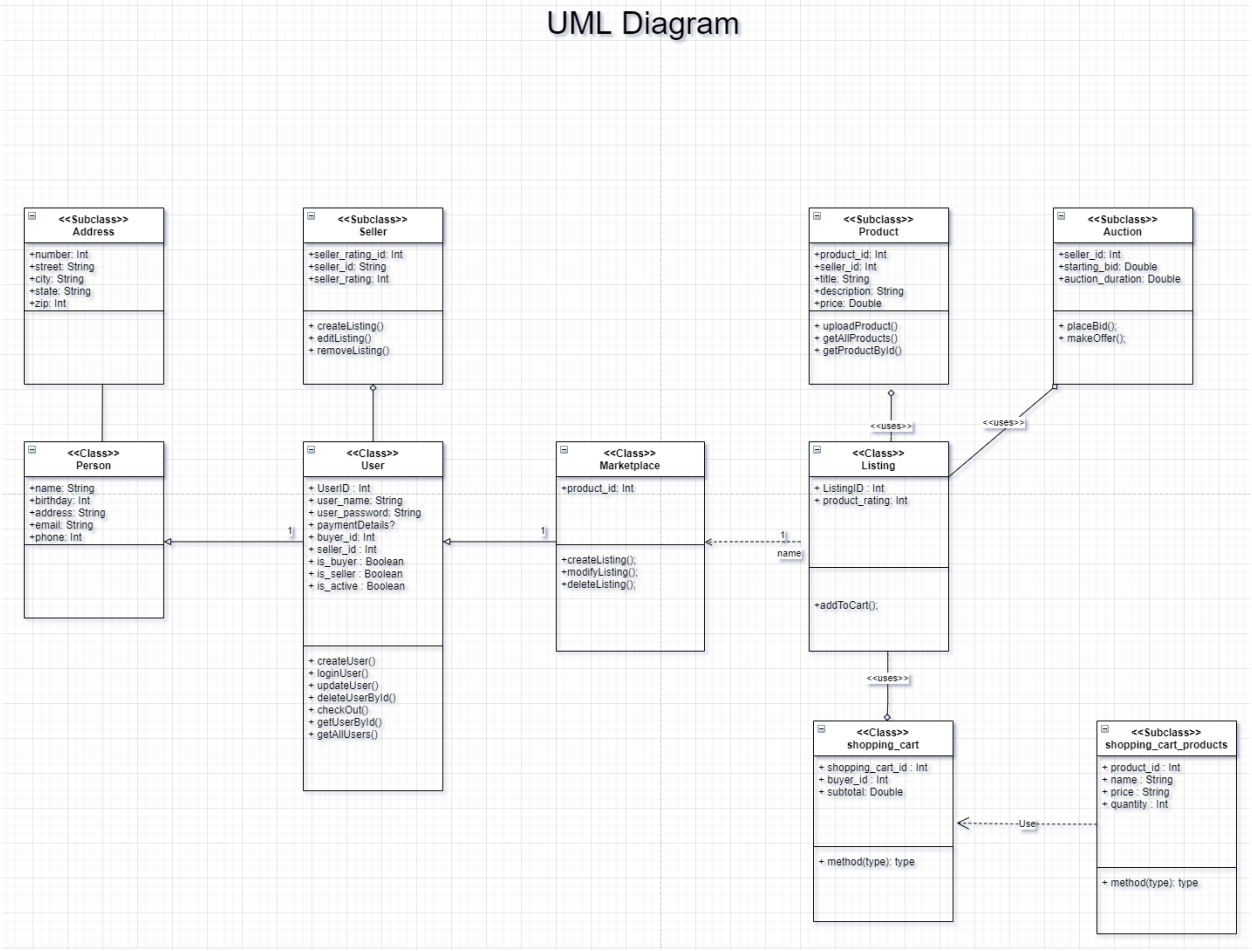
2. This will help to manage business and keep track of customers and sellers actions.

vii. Alerting if most of the items have many reports and complaints from buyers.
    1. If most of the buyers report their items, then the sellers and business management let know that they should be aware to resolve this issue.

viii. Match categories for all products
    1. Making sure what kind of products belong to which categories
        i. For example, if the seller sells a pencil and the categories are school, work, home, and etc. The algorithm will figure out what pencil does relate to one of these categories. Pencil item should be in the school category.
    2. This helps to organize and search for buyers' wanted items.

6. High Level UML Diagrams

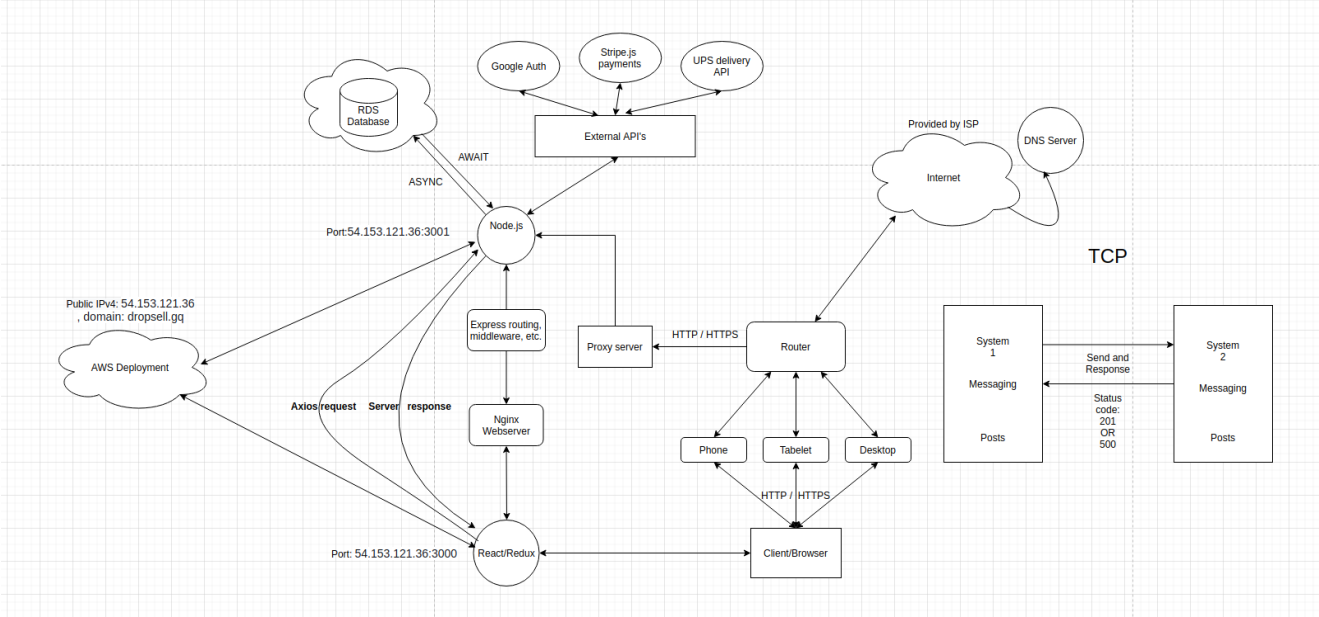
1. https://drive.google.com/file/d/1OmkYbqwzamnWdM2J0xKMSGNLJ2dQBWrc/view?usp=sharing (Bottom of document)



7. High Level Application Network and Deployment Diagrams

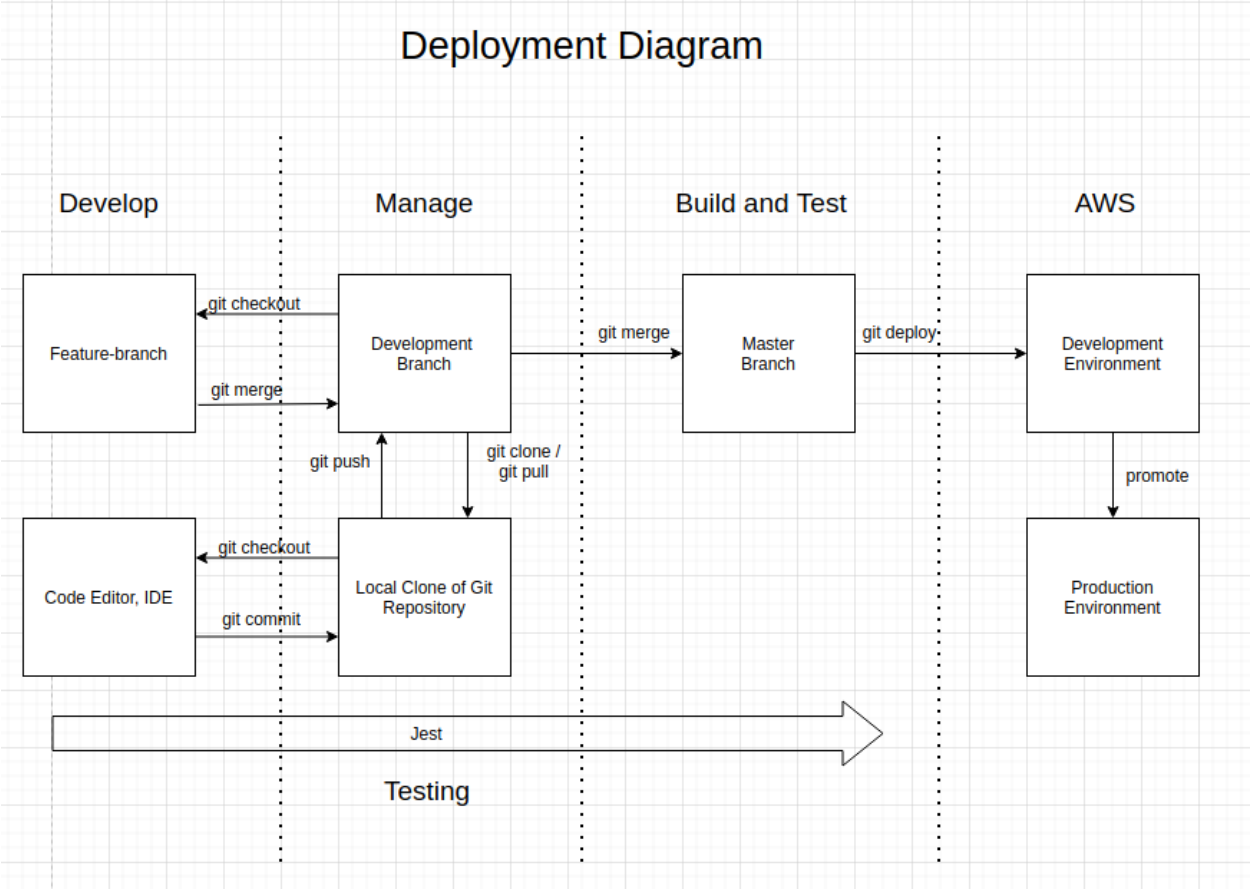
Application Network

1. https://drive.google.com/file/d/1XLbonkJqFnO7ZeOMQqstFPeZm8cr7UxX/view?usp=sharing (Top of document)



Deployment diagrams

1. https://drive.google.com/file/d/1XLbonkJqFnO7ZeOMQqstFPeZm8cr7UxX/view?usp=sharing (Bottom of document)



8. Identify actual key risks for your project at this time

c. Schedule risks
    i. There are conflicting schedules when trying to set up group meetings. Our group will mitigate this by working both asynchronous and synchronous when possible.
d. Legal risks
    i. This is related to either copyright laws when saving images or copying another company's idea. We will resolve this by either making our content as unique as possible, or obtaining the appropriate copyrights and/or licensing.
e. User risks
    i. This is related to scammers who wrongfully use the application. To resolve this issue, we will add security such as captcha, email verification, two-factor authentication, and a report button which users will be able to utilize accordingly.
f. Selling items risk
    i. This unique risk comes with many businesses set up for users to sell their products. Some products can possibly be dangerous and illegal to sell. We will resolve this by labeling such products as dangerous and/or illegal.
g. Skills risk
    i. As with any group work, there will be different skill levels amongst the members. We will mitigate this by sharing information, teaching each other, and working together in a way that best benefits what each member is the most capable of doing.

9. Project Management

As a team, our group decided the best way to complete this milestone was to take a divide and conquer approach. We worked together synchronously as much as possible, asynchronously when we were not able to meet, and communicated through Discord throughout. During meetings, we discussed how we would prioritize our functional requirements and categorized them accordingly. We split up the UI mockups and storyboards initially, then split up work on the database, UML and network diagrams, storyboards and editing the milestone document. We messaged each other in Discord concerning updates. We had set meetings for any questions, and discussed backlogs (upcoming lists of tasks we want done), new features, running tasks (tasks under development, tasks that were fixed and/or upgraded, tasks ready to move to production, and tasks that were labeled as done and/or deployed.) We set up and utilized the Trello application for managing all these tasks as a group. Overall, the methods that our group has been using to complete the milestones have worked really well so far. We will continue to use Trello, have scheduled meetings to manage current and future tasks, as well as maintain daily communication on Discord.

10. Detailed list of contributions

Student Name	Contributions
Mitchel Baker	Mitchel started off M2 working on the UI mockups for our user checkout experience, which included: receipt info, summary, checkout, and final invoice pages. He collaborated with Michael to complete the high level database architecture requirements, ranging from our business rules, ERD diagram, database model, DBMS of choice, media storage decision, and the search/filter architecture. Mitchel also worked with Kenneth in order to research/design our application network diagram, while also creating the deployment diagram, list of contributions, search/filter implementation, and the vertical SW prototype. Lastly, Mitchel wrote up the data definitions and filled in necessary details.
Charmaine Eusebio	Charmaine created a few of the most important UI/UX web pages for our site. She wrote up from scratch the seller analytics (key feature), the create product page, list of auctions, products for sale, starred/watching product and starred/watching list. She also played a pivotal role with the storyboards which reflected our use cases 1 and 2. Lastly, Charmaine identified many of the key risks which she found sufficient solutions for, while also discussing our project management style.
Kenneth N Chuson	Kenneth did a tremendous job with the amount of detail he put into the profile and user settings pages. You can see the amount of work he put into them as a result of his mockup's quality. Kenneth also played a huge role with the write up of our high level API's and main algorithms. He has brought up

	numerous ideas involving data analytics and machine learning which will really make our application stand out. Lastly, Kenneth conducted the majority of the research for creating our application network diagram.
Krina Bharatbhai Patel	Krina jumpstarted our progress on Milestone 2 by creating our initial set of data definitions. She laid the groundwork for the future additions added into our data definitions section. Krina also demonstrated her creativity with the login, register, auction, and chat mockups. Her mockups display all of the key features which we want to implement for our application, while showing off a solid understanding of UI/UX fundamentals. Krina was also responsible for prioritizing our functional requirements. Despite not being an easy task, she took to it quickly and showed that she can deliver quality work with efficiency. Lastly, Krina was also a big part of creating the vertical prototype by implementing a couple of the main UI components for our home page.
Michael Schroeder	Michael was assigned the creation of our initial home and menu page mockups. He took to these tasks with diligence and demonstrated his ability as a quick learner. Michael also collaborated with Mitchel on the ERD diagram portion of the high level database architecture requirements, while also implementing from scratch the UML diagram for our project. Despite not having prior experience writing UML diagrams, he demonstrated an eagerness to learn new skills which is a rare skill to find in people.
Rowena Elaine Echevarria	Rowena was in charge of writing up the about page for our UI mockups. She also did a tremendous job with the creation of her storyboards for use cases 3, 4, 5. She incorporated a ton of

	detail into her storyboards, which demonstrates her attention to detail, especially in regards to UI/UX. As a result of the detail put into her storyboards, our Milestone 2 gives readers the opportunity to see first-hand examples of how Dropsell solves problems for users who are looking to either buy or sell products online.
Jamie Dominic Walker	Jamie was in charge of the high level API and main algorithms section. He demonstrated an eagerness to learn more about API's by researching and sharing videos/articles about the topic. Jamie is someone who knows how to apply his critical thinking skills to real-world applications. When it comes to explaining his thoughts and ideas, Jamie takes an implementation-based approach to explaining our main algorithms which was extremely helpful. He explored various edge cases related to our API implementations, while also bringing up thought-provoking analysis during our meeting discussions.