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 3
July 22, 2021

History Table

Version	Date	Notes
M3V2	08/02/2021	
M3V1	07/22/2021	
M2V2	07/20/2021	
M2V1	07/08/2021	
M1V2	07/02/2021	
M1V1	06/22/2021	

Table of Contents

1. Data Definitions **3**

2. Functional Requirements **11**
Priority 1 11
Priority 2 13
Priority 3 16

3. Wireframes Based on Mockups/Storyboards **17**

4. High Level Database Architecture and Organization **39**
Entity Relationship Diagram (User) 43
Database Model 44

5. High Level Diagrams **46**
UML Diagram 46
Application Network Diagram 47
Deployment diagram 48

6. List of Contributions **50**

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
            Must enter username to login
        ii. password; CHAR(90), NN
            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
            Attributes
                bCC number
                Expiration date
                3 digit code
                Zip
            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?
            is_buyer; type BOOL, NN
            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
            message_id; type INT, PK, NN, AI
            conversation_id (FK to conversation id); type INT, NN

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
         product_id; type INT, PK, NN, AI
         shopping_cart_id (FK to shopping cart id); type INT, NN
         title; type VARCHAR(250), NN
         price; type VARCHAR(90), NN
         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
         At Least 5 word Title or name of the product
      iv. description; type VARCHAR(500), NN
         At Least 10 word Description of the product
      v. image; type VARCHAR(100), NN
         At least 3 images of size (600x600 or 2x2) for product visibility

        vi. price; type VARCHAR(90), NN
            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
            setUsername(username)
                Action type: ‘USER_SET_USERNAME’
                Datatype: String
            setPassword(password)
                Action type: ‘USER_SET_PASSWORD’
                Datatype: String
            loginUser()
                userData(username, password)
            redirectUserAfterLogin(loggedIn)
                Action type: ‘USER_IS_LOGGEDIN’
                Datatype: String
        ii. Reducer
            INITIAL_LOGIN_STATE
                Username; datatype String
                Password; datatype String
                loggedIn; datatype Boolean

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

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

- Action type: ‘PRODUCT_SET_DESCRIPTION’
  - Datatype: String
- setPrice(price)
  - Action type: ‘PRODUCT_SET_PRICE’
  - Datatype: String
- setImage(image)
  - Action type: ‘PRODUCT_SET_IMAGE’
  - Datatype: String
- setSuccess(isSuccess)
  - Action type: ‘PRODUCT_SET_SUCCESS’
  - Datatype: Boolean
- setCategory(category)
  - Action type: ‘PRODUCT_SET_CATEGORY’
  - Datatype: String
- setCategories(categories)
  - Action type: ‘SET_CATEGORIES’
  - Datatype: Boolean
- changeDropdownText(text)
  - Action type: ‘CHANGE_DROPDOWN_TEXT’
  - Datatype: String
- createProduct()
  - formData(title, description, price, category, file)
- getProducts(products)
  - Action type: ‘GET_PRODUCTS’
  - Datatype: Array

ii. Reducer

- INITIAL_PRODUCT_STATE
  - Title; datatype String
  - Description; datatype String
  - Price; datatype String
  - Category; datatype String
  - File: datatype String
  - filePreview; datatype null/URL object
  - isSuccess; datatype, null/boolean
  - Products; datatype Array
  - Categories; datatype Boolean
  - dropdownText; datatype String

d. Seller Settings

i. Actions

1. updateFirstName(firstName)
   - a. Action type: ‘USER_UPDATE_FIRSTNAME’’
   - b. Datatype: String
- updateLastName(lastName)
  - a. Action type: ‘USER_UPDATE_LASTNAME’
  - b. Datatype: String
- updateBirthday(birthday)
  - a. Action type: ‘USER_UPDATE_BIRTHDAY’
  - b. Datatype: Date

4. updateEmail(email)
    a. Action type: ‘USER_UPDATE_EMAIL’
    b. Datatype: String
5. updatePhone(phone)
    a. Action type: ‘USER_UPDATE_PHONE’
    b. Datatype: Character
6. updateUserName(userName)
    a. Action type: ‘USER_UPDATE_USERNAME’’
    b. Datatype: String
7. updatePassword(password)
    a. Action type: ‘USER_UPDATE_PASSWORD’’
    b. Datatype: String
8. updateCardNumber(cardNumber)
    a. Action type: ‘USER_UPDATE_CARDNUMBER’
    b. Datatype: Integer
9. updateExpirationDate(cardExpiration)
    a. Action type: ‘USER_UPDATE_CARDEXPIRATION’
    b. Datatype: Date
10. updateCVV(cardCVV)
    a. Action type: ‘USER_UPDATE_CARDCVV’
    b. Datatype: Integer
11. updatePostalCode(postalCode)
    a. Action type: ‘USER_UPDATE_POSTALCODE’
    b. Datatype: Integer
12. updateBioDescription(bioDescription)
    a. Action type: ‘USER_UPDATE_BIODESCRIPTION’
    b. Datatype: String
13. updateLocation(location)
    a. Action type: ‘USER_UPDATE_LOCATION’
    b. Datatype: String
14. updateSocialMedia(socialMedia)
    a. Action type: ‘USER_UPDATE_LOCATION’
    b. Datatype: String
15. updateNoteSchedule(noteSchedule)
    a. Action type: ‘USER_UPDATE_NOTESCHEDULE’
    b. Datatype: Array

ii. Reducer

1. INITIAL_SELLER_SETTINGS_STATE

    firstName; datatype String
    lastName; datatype String
    birthday; datatype Date
    email; datatype Integer
    phone; datatype Character
    userName; datatype String
    password; datatype String
    cardNumber; datatype Integer
    cardExpiration; datatype Date
    updateCVV; datatype Integer

updatePostalCode; datatype Integer
bioDescription; datatype String
location; datatype String
socialMedia: datatype String
noteSchedule; datatype Array

e. Buyer Settings

i. Actions

1. updateFirstName(firstName)
   a. Action type: ‘USER_UPDATE_FIRSTNAME’’
   b. Datatype: String
2. updateLastName(lastName)
   a. Action type: ‘USER_UPDATE_LASTNAME’
   b. Datatype: String
3. updateBirthday(birthday)
   a. Action type: ‘USER_UPDATE_BIRTHDAY’
   b. Datatype: Date
4. updateEmail(email)
   a. Action type: ‘USER_UPDATE_EMAIL’
   b. Datatype: String
5. updatePhone(phone)
   a. Action type: ‘USER_UPDATE_PHONE’
   b. Datatype: Character
6. updateUserName(userName)
   a. Action type: ‘USER_UPDATE_USERNAME’’
   b. Datatype: String
7. updatePassword(password)
   a. Action type: ‘USER_UPDATE_PASSWORD’’
   b. Datatype: String
8. updateCardNumber(cardNumber)
   a. Action type: ‘USER_UPDATE_CARDNUMBER’
   b. Datatype: Integer
9. updateExpirationDate(cardExpiration)
   a. Action type: ‘USER_UPDATE_CARDEXPIRATION’
   b. Datatype: Date
10. updateCVV(cardCVV)
    a. Action type: ‘USER_UPDATE_CARDCVV’
    b. Datatype: Integer
11. updatePostalCode(postalCode)
    a. Action type: ‘USER_UPDATE_POSTALCODE’
    b. Datatype: Integer
12. updateBioDescription(bioDescription)
    a. Action type: ‘USER_UPDATE_BIODESCRIPTION’
    b. Datatype: String
13. updateRateStars(RateStars)
    a. Action type: ‘USER_UPDATE_RATESTARS’
    b. Datatype: Integer
14. updateReview(review)
    a. Action type: ‘USER_UPDATE_REVIEW’

        b. Datatype: String

15. updateSocialMedia(socialMedia)
    a. Action type: ‘USER_UPDATE_SOCIALMEDIA’
    b. Datatype: String
16. updateShowBuys(showBuys)
    a. Action type: ‘USER_UPDATE_SHOWBUYS’
    b. Datatype: Boolean
17. updateShowBuysReviews(showBuysReviews)
    a. Action type: ‘USER_UPDATE_SHOWBUYSREVIEWS’
    b. Datatype: Boolean
18. updateMailingAddress(mailAddress)
    a. Action type: ‘USER_UPDATE_MAILADDRESS’
    b. Datatype: String
19. updateZipCode(zipCode)
    a. Action type: ‘USER_UPDATE_ZIPCODE’
    b. Datatype: Integer

ii. INITIAL_BUYER_SETTINGS_STATE

firstName; datatype String
lastName; datatype String
birthday; datatype Date
email; datatype Integer
phone; datatype Character
userName; datatype String
password; datatype String
cardNumber; datatype Integer
cardExpiration; datatype Date
updateCVV; datatype Integer
updatePostalCode; datatype Integer
bioDescription: datatype String
rateStars: datatype Integer
review: datatype String
socialMedia; datatype String
showBuys: datatype Boolean
showBuysReviews: datatype Boolean
socialMedia: datatype String
showBuys; datatype Boolean
showBuysReviews; datatype Boolean
mailAddress: datatype String
zipCode; datatype Integer

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

2. Functional Requirements

Priority 1

Marketplace

1. Unregistered and registered users shall be able to query the database for products by interacting with a search bar
2. Sellers shall be able to put their products up for auction or list them in the marketplace
3. Buyers shall be able to propose buy it now or best offer options for a seller's product
4. Buyers shall be able to filter the products they are searching for, based on minimum/maximum price, location filtering, or type of shipping (pickup/delivery)
5. Buyers shall be able to check their account settings to determine if a product they've purchased has been confirmed, processed, shipped, or returned
6. Buyers shall be given a tracking number to stay up to date with their product's delivery status

Website Features

7. Sellers shall be provided with consignment operations to get their products delivered to buyers
8. Buyers shall be given a price matching tool, which compares products either already posted on our app or compares other products from other websites (amazon)

Admin

9. Admins shall be able to delete products from the admin page
10. Admins shall be able to see all product discussions for a specific product
11. Admins shall be able to delete product discussions for a specific product
12. Admins shall be able to delete price matching products
13. Admins shall be able to search for a specific user
14. Admins shall be able to view all data related to a specific user
15. Admins shall be able to see the shopping cart data of a specific user
16. Admins shall be able to ban a user from Dropsell
17. Admins shall be able to restrict certain features of Dropsell from a specific user
18. Admins shall have a live chat feature they can join to communicate directly with users

Buyers

19. Buyers shall sign a purchase agreement; they must agree to conduct business according to Dropsell's rules prior to using the marketplace
20. Buyers shall be able to send inquiries of interest to a product's seller
21. Buyers shall save products they wish to purchase by adding them into a shopping cart

22. Buyers shall be able to remove an item from their shopping cart and the total list of products should be updated accordingly
23. Buyers shall be able to return to the main shopping menu to look for other products if they are not finished shopping
24. Buyers shall have the option of canceling or modifying their order
25. Buyers shall receive a detailed receipt of their purchase through email after checking out
26. Buyers shall be able to add their full name, email, phone number, and delivery address if they haven’t inputted this data prior to checking out
27. Buyers shall have the option of choosing an existing payment option or adding a new one
28. Buyers shall be notified by final invoice of their purchase’s pickup/delivery time, expected time of arrival, name of seller, location to meet them at, their contact info, and the grand total to pay

Sellers

29. Sellers shall sign a contract before being granted the privilege of posting products on Dropsell
30. Sellers shall agree to a small listing fee for each product they sell
31. Sellers shall be able to create new products with a title, description, price, category, and image
32. Sellers shall be able to edit the title, description, price, category, and image of their products
33. Sellers shall be able to adjust the quantity of a product they’ve listed on Dropsell
34. Sellers shall be able to delete products they’ve listed
35. Sellers shall be shown all their listed products under the seller settings section of their profile

Auction

36. Sellers shall have full control over their auction settings, from choosing the average starting bid, the time when the auction begins, to the total duration of the auction
37. Sellers shall be able to set the duration of their auction for as long as 30 days or as short as 1 hour
38. Buyers shall be displayed the current price of an auction, the amount of time remaining, and any bids from other buyers
39. Buyers shall be able to bid on products with one click.
40. Buyers and sellers shall see a live countdown of the auction’s remaining time

Messaging

41. Buyers shall have the ability to contact sellers directly through the Dropsell website
42. Buyers shall have the ability to contact sellers via email or phone, if they choose to do so

Priority 2

Marketplace

43. Buyers shall be given an advanced search bar for detailed searches, including product sizes, price ranges, and colors
44. Buyers shall be refunded in full if they purchase multiple products which causes the product to be sold out
45. Products shall have their reviews displayed when listed in the marketplace
46. Products shall be displayed as out of stock if the supply has run out
47. Sellers shall be able to publish different styles of their products, such as different colors or fabrics
48. Unregistered and registered users shall be able to share products with friends through email, social media, or a shareable link
49. Registered users shall have the option of flagging products as inappropriate, a scam, or submit a ticket to the Dropsell team for further review
50. Buyers and sellers shall be able to schedule and edit meetup times
51. Users should be offered a base price for automatic reject or accept
52. Buyers shall be able to access the location information of sellers through an interactive map
53. Unregistered and registered users shall be shown ads of products similar to their search histories
54. Buyers shall be given a clear breakdown of shipping options to choose from
55. Buyers shall be contacted by email if they opt to be notified when a product comes back in stock
56. Unregistered and registered users shall be able to go back to products they have viewed previously on their user feed
57. The marketplace shall show the maximum time length for shipping a product
58. The marketplace shall show the minimum time length for shipping a product
59. The marketplace shall perform monitoring on unregistered and registered users
60. The marketplace shall keep track of the ID’s of posts unregistered and registered users click on which will be saved in the database under most-interacted products
61. The marketplace shall contain a currency converter for entering international markets

Website Features

62. Sellers shall be able to see statistics related to how many buyers have purchased their products
63. Sellers shall be provided with an algorithm which keeps track of how many interactions their products have
64. The checkout page shall calculate tax automatically based on the buyer’s location
65. Buyers shall be provided with daily deals on the home page
66. Sellers shall be provided with a profitability test for determining how much it costs to ship a product on Dropsell as opposed to other ecommerce services

67. Users shall be able to zoom in or out of product images with a magnifying glass feature
68. Sellers shall be provided with a daily product forecast
69. Products shall contain hashtags to help with categorization
70. Unregistered and registered users shall be able to search products using hashtags
71. Buyers shall be shown products which are similar in price or category to the product they are currently viewing
72. Buyers shall be shown comparisons between product specifications
73. Sellers shall be able to keep track of the median price ranges of products they are viewing

Buyers

74. Buyers shall receive shipping information through email if their purchase involves shipping products to their specified address
75. Buyers shall be able to rate and star products
76. Buyers shall be able to interact with all of their starred products in their buyer settings
77. Buyers shall be able to choose from various UPS delivery methods
78. Buyers shall be able to view all data related to a seller's review page
79. Buyers shall add auction products to their watch list to keep track of the auction's most recent updates
80. Top-rated products shall be displayed to buyers after they purchase a similar product
81. Buyers shall be permitted to submit reviews of products they purchase
82. Buyers shall be able to click on product listing images to zoom in and see more details
83. Buyers shall be provided with a list of their purchase histories
84. Buyers shall be able to save products they'd like to buy in the future to a wishlist
85. Buyers shall be rewarded with discount codes if they are frequent buyers
86. Buyers shall be provided with a promotion code box for entering their discount code when checking out
87. Buyers shall be able to generate a referral URL which they can send to friends
88. Buyers shall be able to subscribe to a specific seller so they can be notified when a product's price is updated, the product has been restocked, or if the product has been removed
89. Buyers shall be notified about products they have viewed previously
90. Buyers shall be asked if they are still interested in the products they've previously viewed

Sellers

91. Sellers shall get an email confirmation after one of their products has sold y
92. Sellers shall allow their data such as user details, products, profile picture, to be publicly displayed to buyers
93. Sellers shall have the option of selecting multiple products to delete

94. Sellers shall have the option of relisting their products for sale
95. Sellers shall be able to advertise their products to specific buyers with a send offer feature
96. Seller products shall have a review page where buyers send feedback about its quality
97. Sellers shall have a review page where buyers can provide structured feedback
98. Sellers shall have a rating system based on a five-star system
99. Sellers shall be required to display “illegal item” on illegal products
100. Sellers shall be required to explain their illegal item and their reasons for listing it
101. Seller products shall be marked as out of stock if its supply runs out
102. Seller products shall be marked as “Last 1 Available” so buyers know that it is the last product remaining
103. Sellers shall participate in a striking system if they choose to perform misconduct or violate the terms of service.
104. Sellers shall have 3 strikes before their account faces a possible suspension or blacklist

Auction

105. Buyers shall not be able to retract a bid they place on a product
106. Buyers shall be notified immediately by email that they’ve won an auction
107. Buyers shall bid on products as many times as possible before the remaining time runs out
108. Buyers shall be able to keep track of auction statistics
109. Buyers shall be able to see how many other buyers are bidding on the same product
110. Buyers shall be notified with a 5 minute warning before an auction finishes
111. Buyers shall be notified when the auction finishes
112. Buyers shall be notified after they win an auction
113. Buyers shall have the option of choosing from multiple payment options, such as Paypal, ApplePay, or GooglePlay

Priority 3

Marketplace

114. Sellers shall be able to create a wedding registry with a product wishlist for their special day
115. Buyers shall be shown other buyers who have made the same purchases
116. Buyers and sellers shall have the time length of shipping a product hidden if their location is closer than 1 or 2 miles

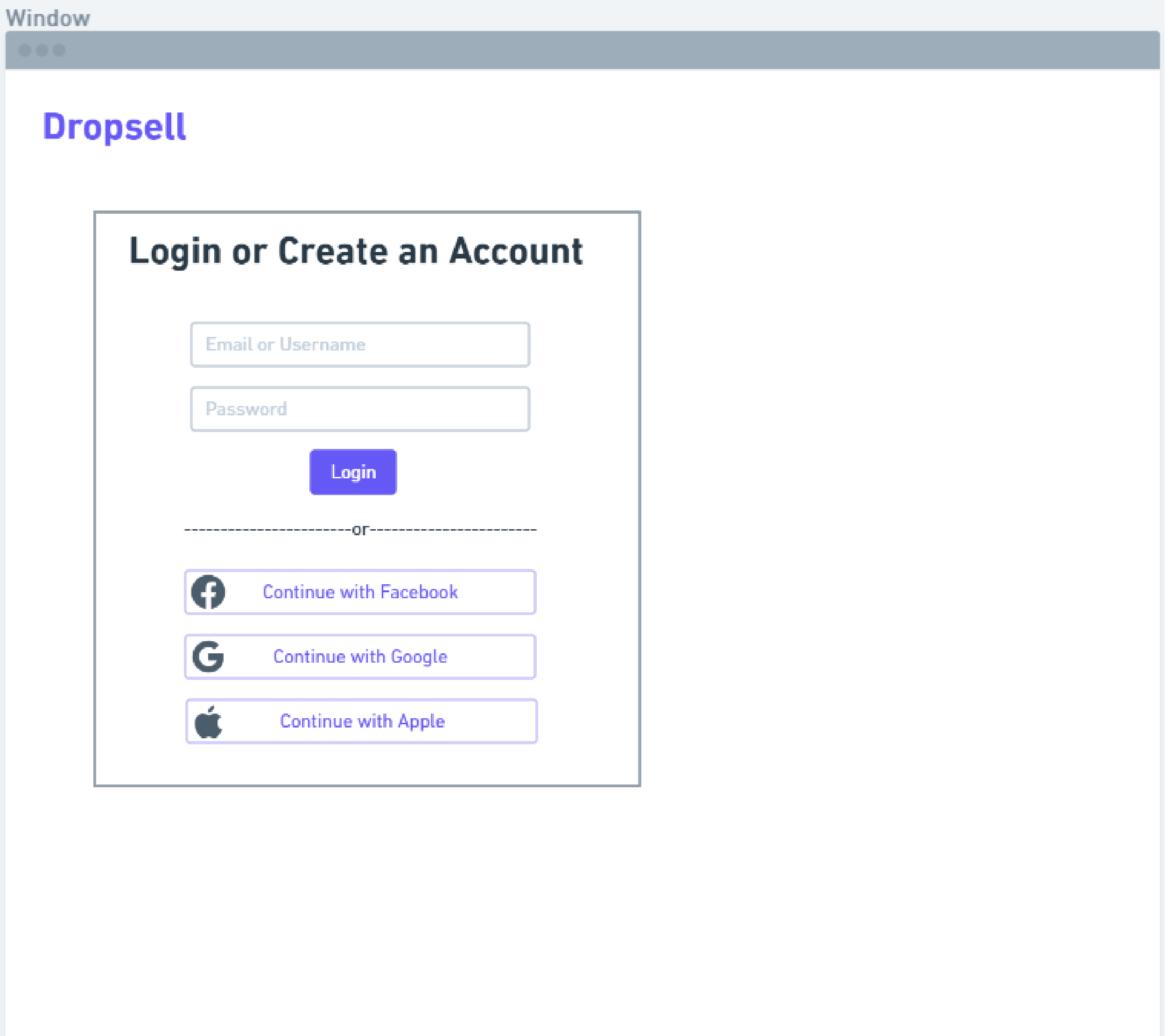
Buyers

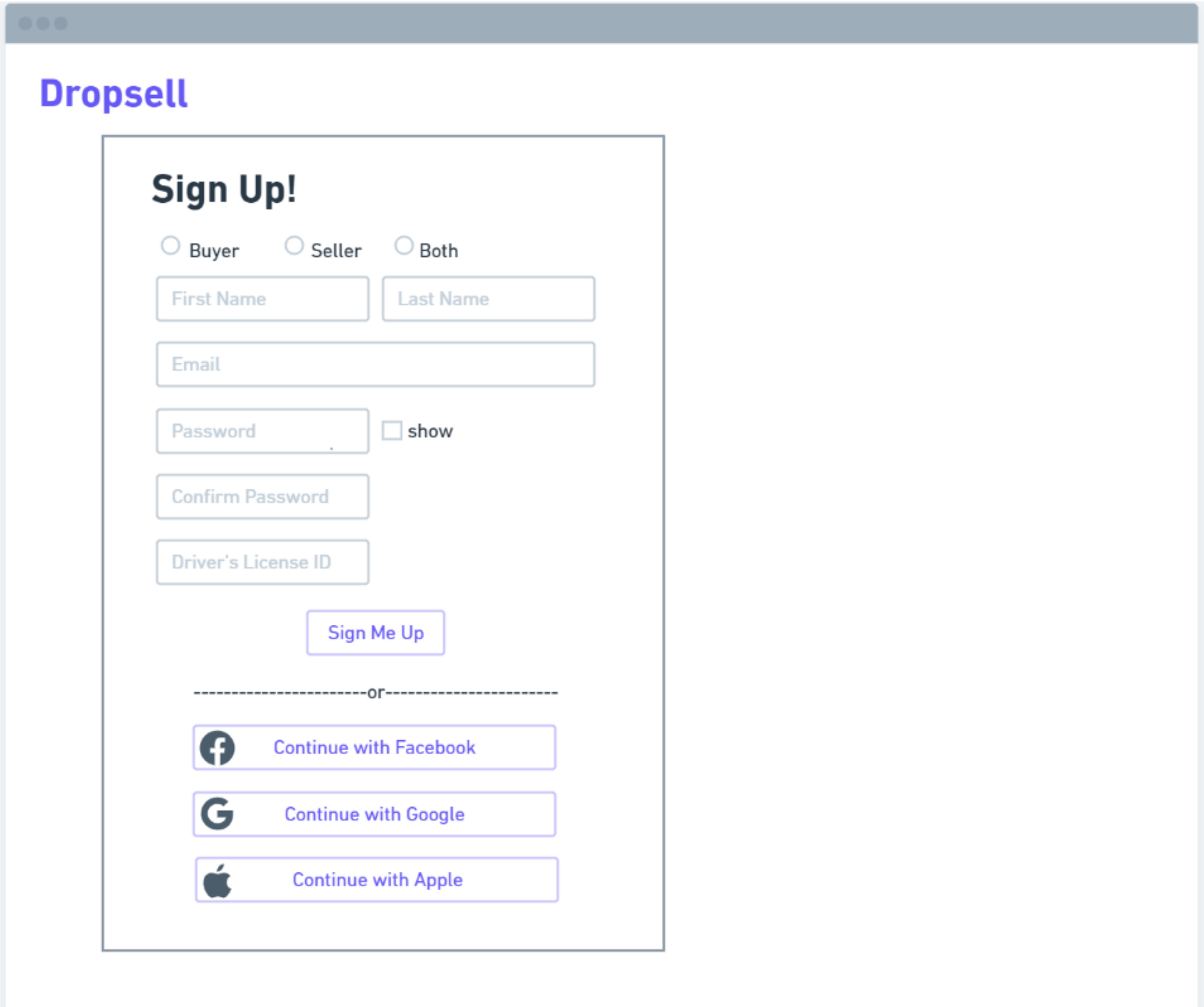
117. Buyers shall be awarded with a random product if they purchase products deemed as lucky
118. Buyers shall be rewarded with a promotion code or a random product for their birthday

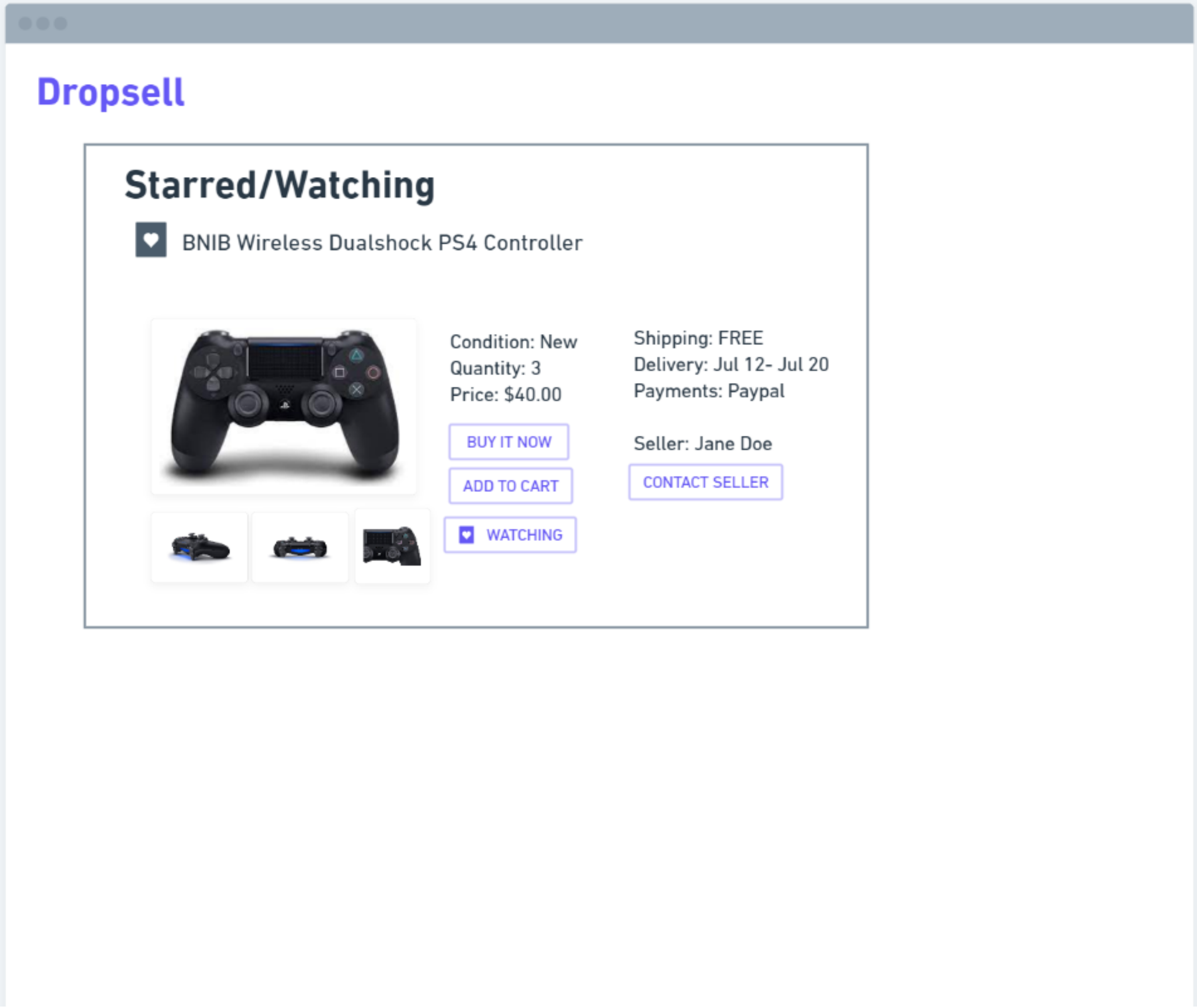
Sellers

119. Sellers shall have background checks if they intend to sell on Dropsell
120. Sellers shall have their products hide ratings if they have 5 stars

3. Wireframes Based on Mockups/Storyboards




Dropsell
Sign Up!
○ Buyer ○ Seller ○ Both
First Name
Last Name
Email
Password
□ show
Confirm Password
Driver's License ID
Sign Me Up
----------------------or----------------------
Continue with Facebook
Continue with Google
Continue with Apple



Dropsell

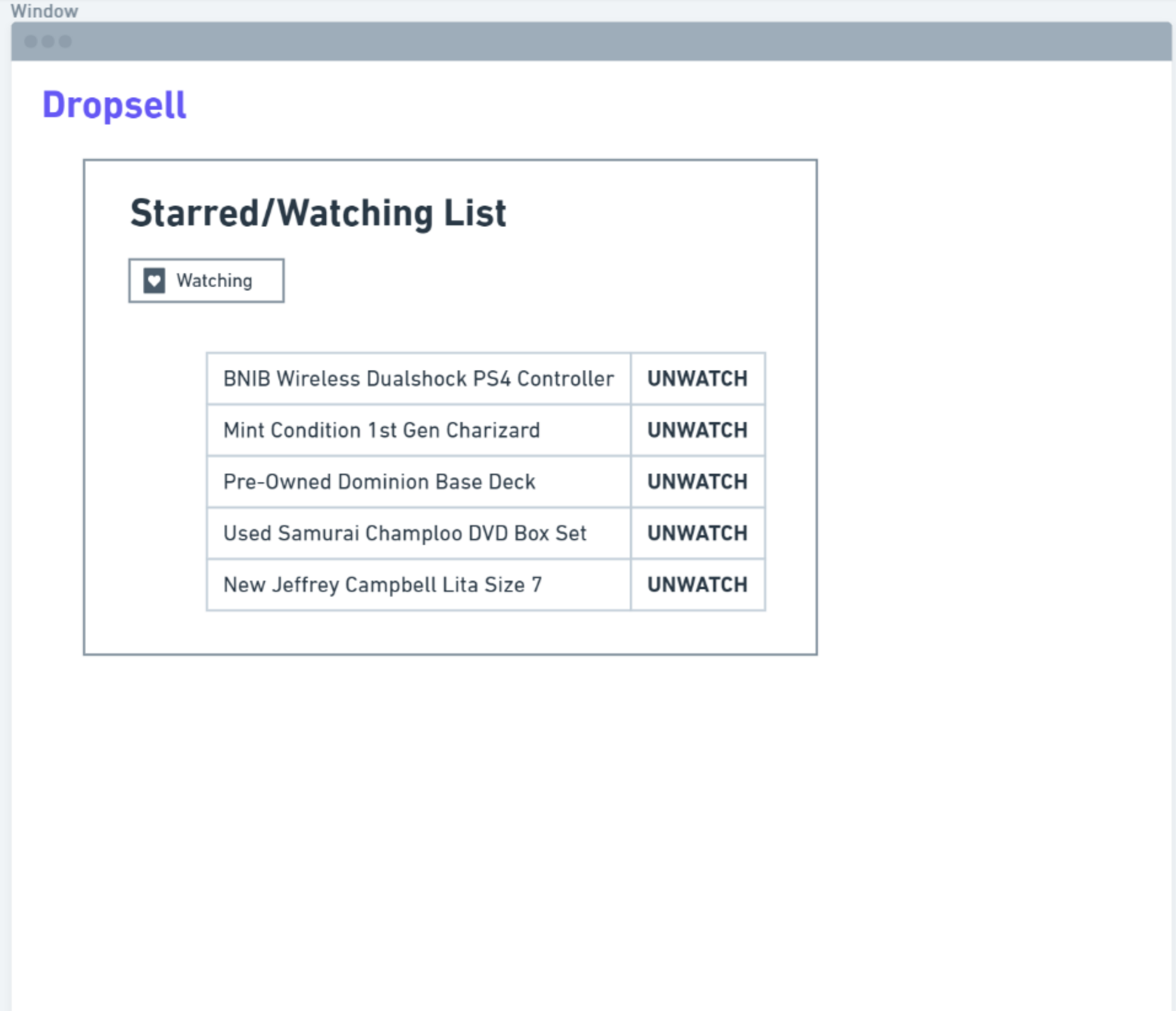
Starred/Watching

BNIB Wireless Dualshock PS4 Controller

Condition: New
Quantity: 3
Price: $40.00

Shipping: FREE
Delivery: Jul 12- Jul 20
Payments: Paypal

Seller: Jane Doe

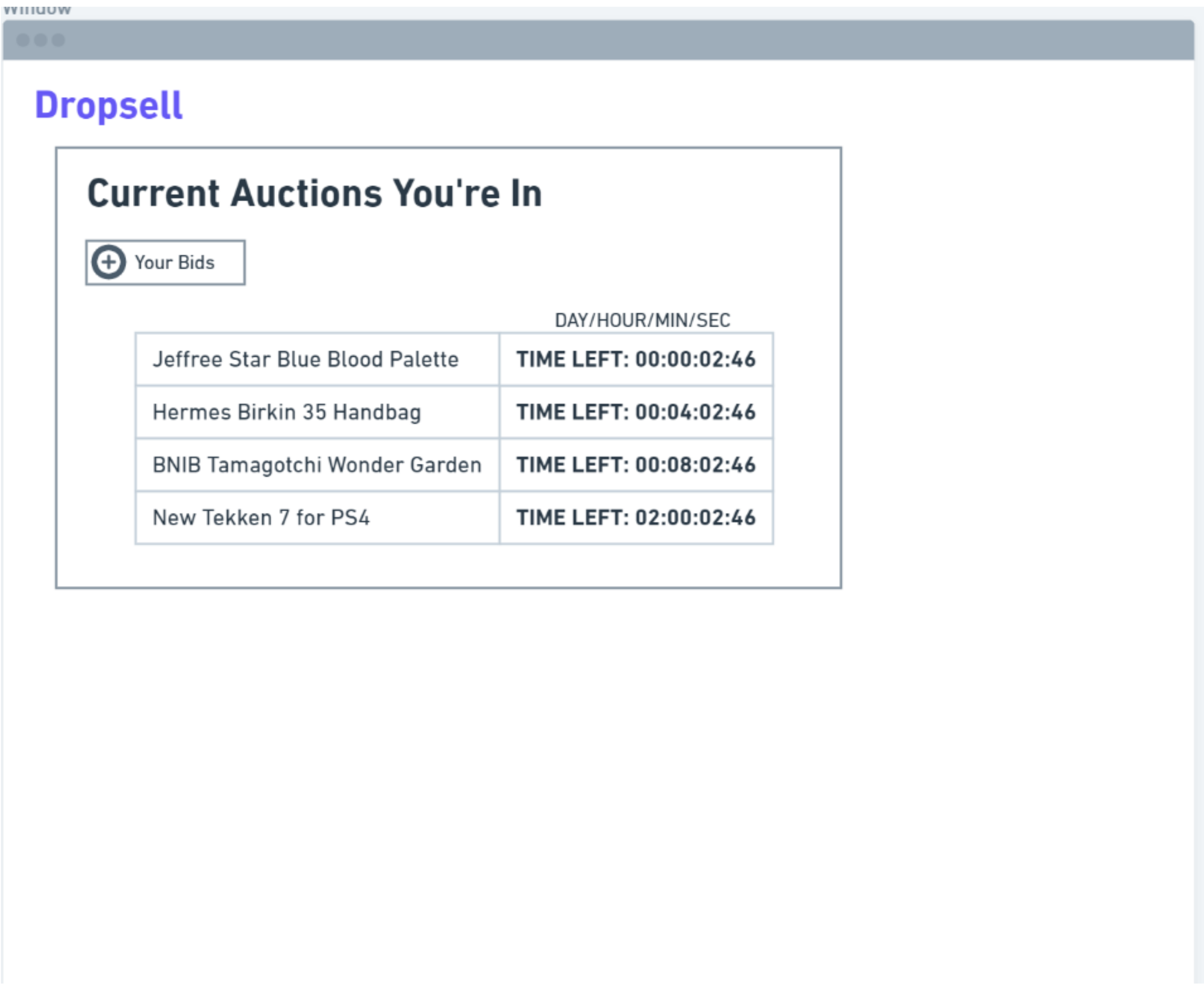


Dropsell

Starred/Watching List

♥ Watching

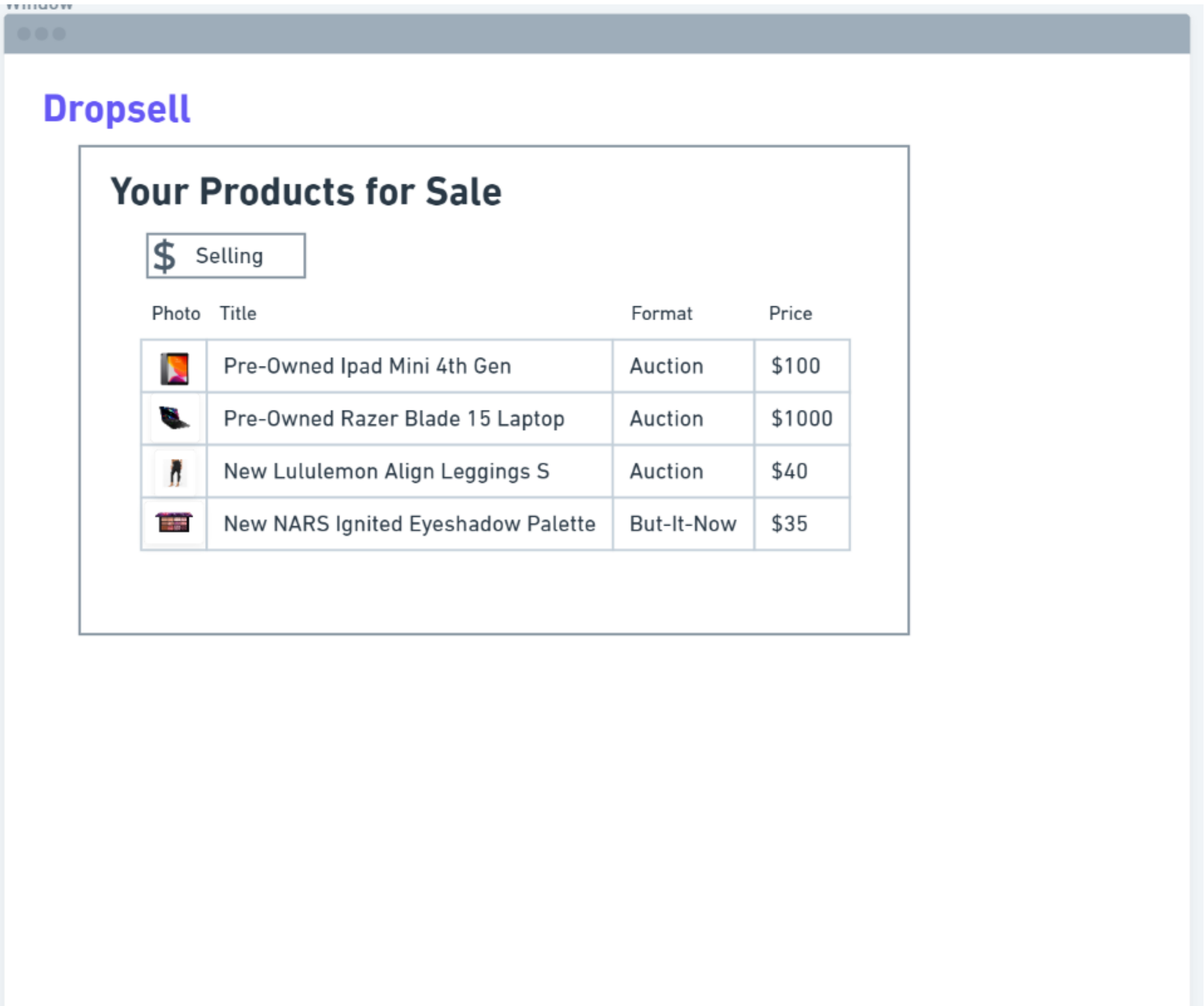
BNIB Wireless Dualshock PS4 Controller	UNWATCH
Mint Condition 1st Gen Charizard	UNWATCH
Pre-Owned Dominion Base Deck	UNWATCH
Used Samurai Champloo DVD Box Set	UNWATCH
New Jeffrey Campbell Lita Size 7	UNWATCH



Current Auctions You're In

DAY/HOUR/MIN/SEC

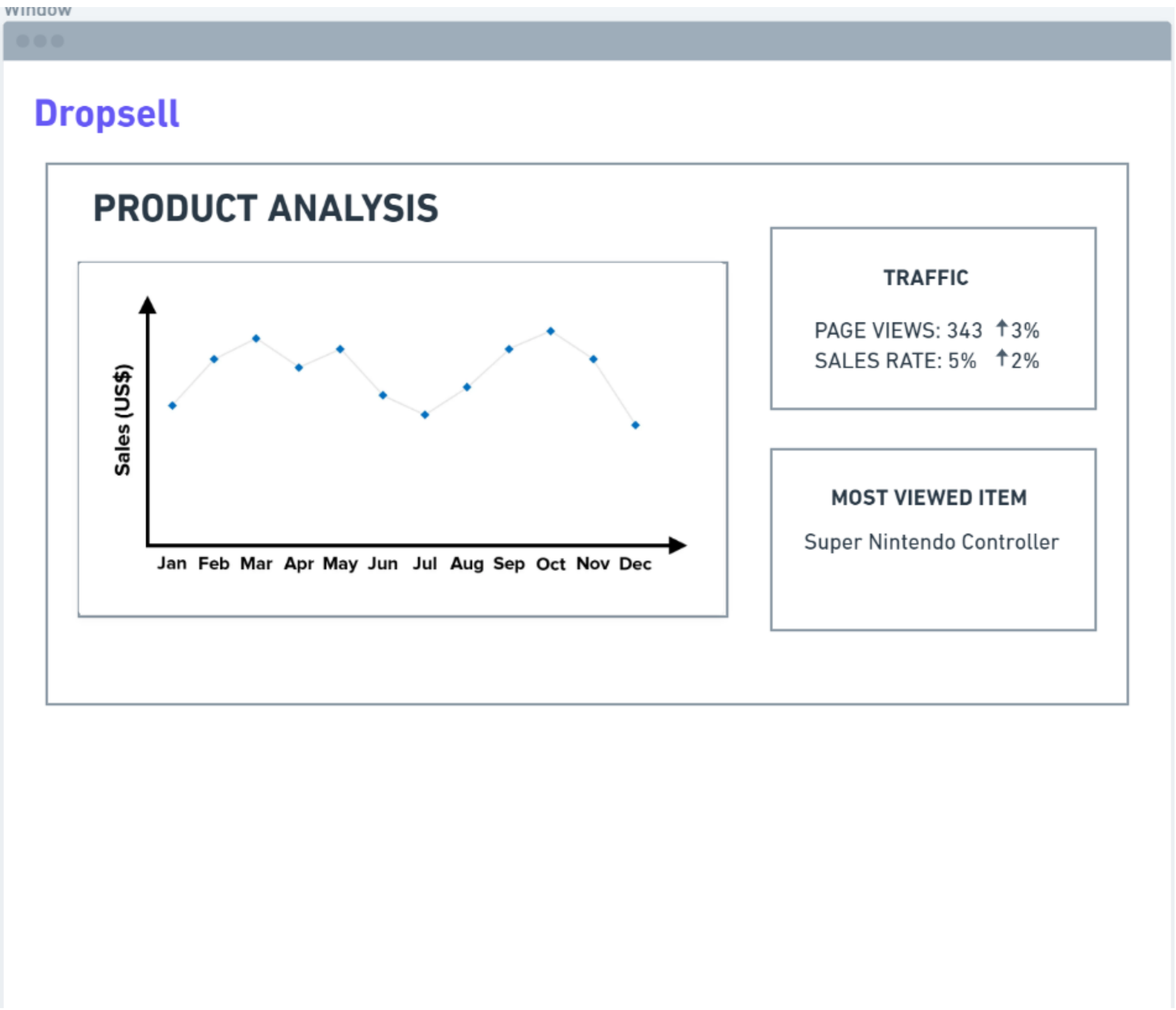
Jeffree Star Blue Blood Palette	**TIME LEFT: 00:00:02:46**
Hermes Birkin 35 Handbag	**TIME LEFT: 00:04:02:46**
BNIB Tamagotchi Wonder Garden	**TIME LEFT: 00:08:02:46**
New Tekken 7 for PS4	**TIME LEFT: 02:00:02:46**



Dropsell

Your Products for Sale

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	But-It-Now	$35



Dropsell

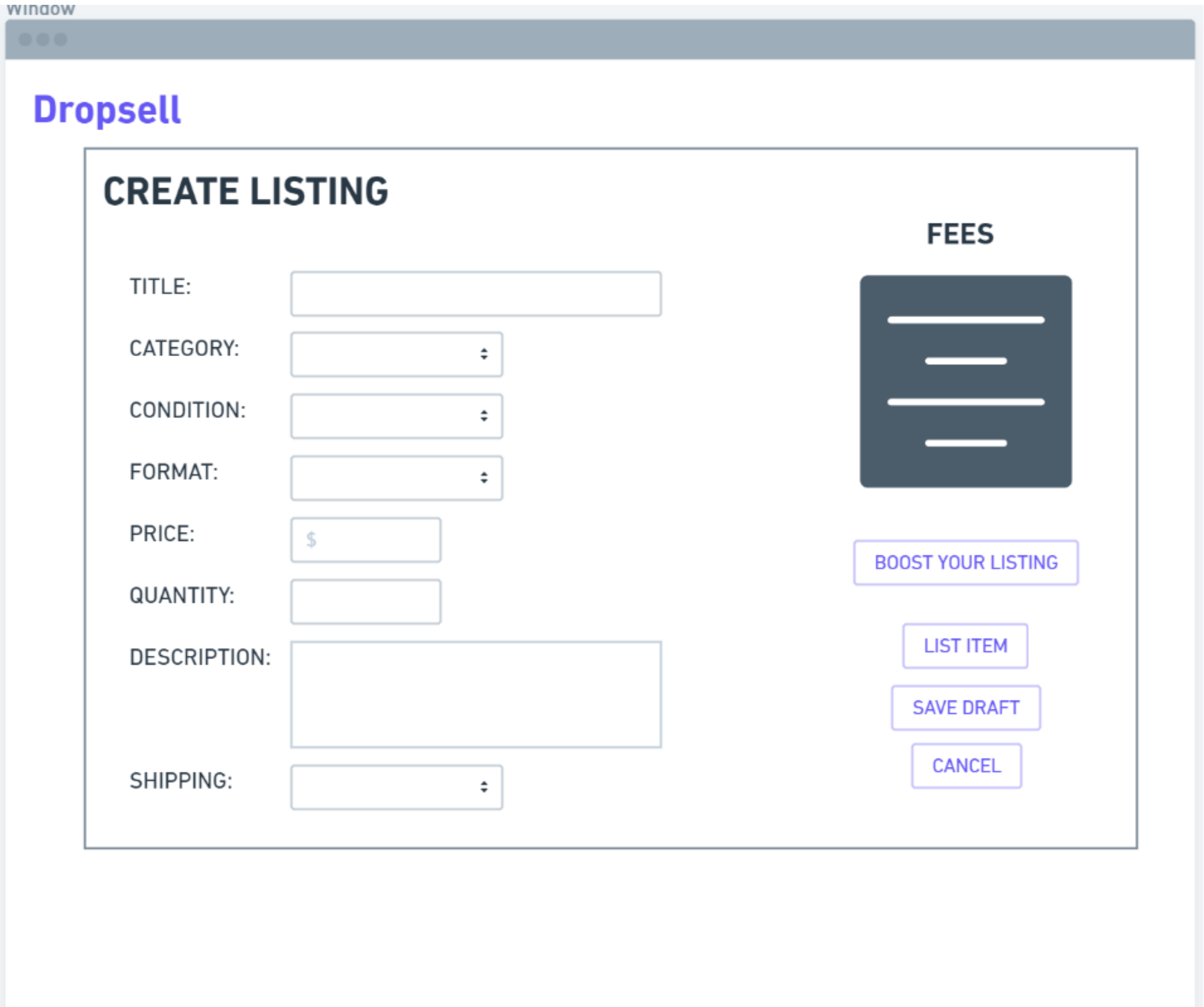
PRODUCT ANALYSIS

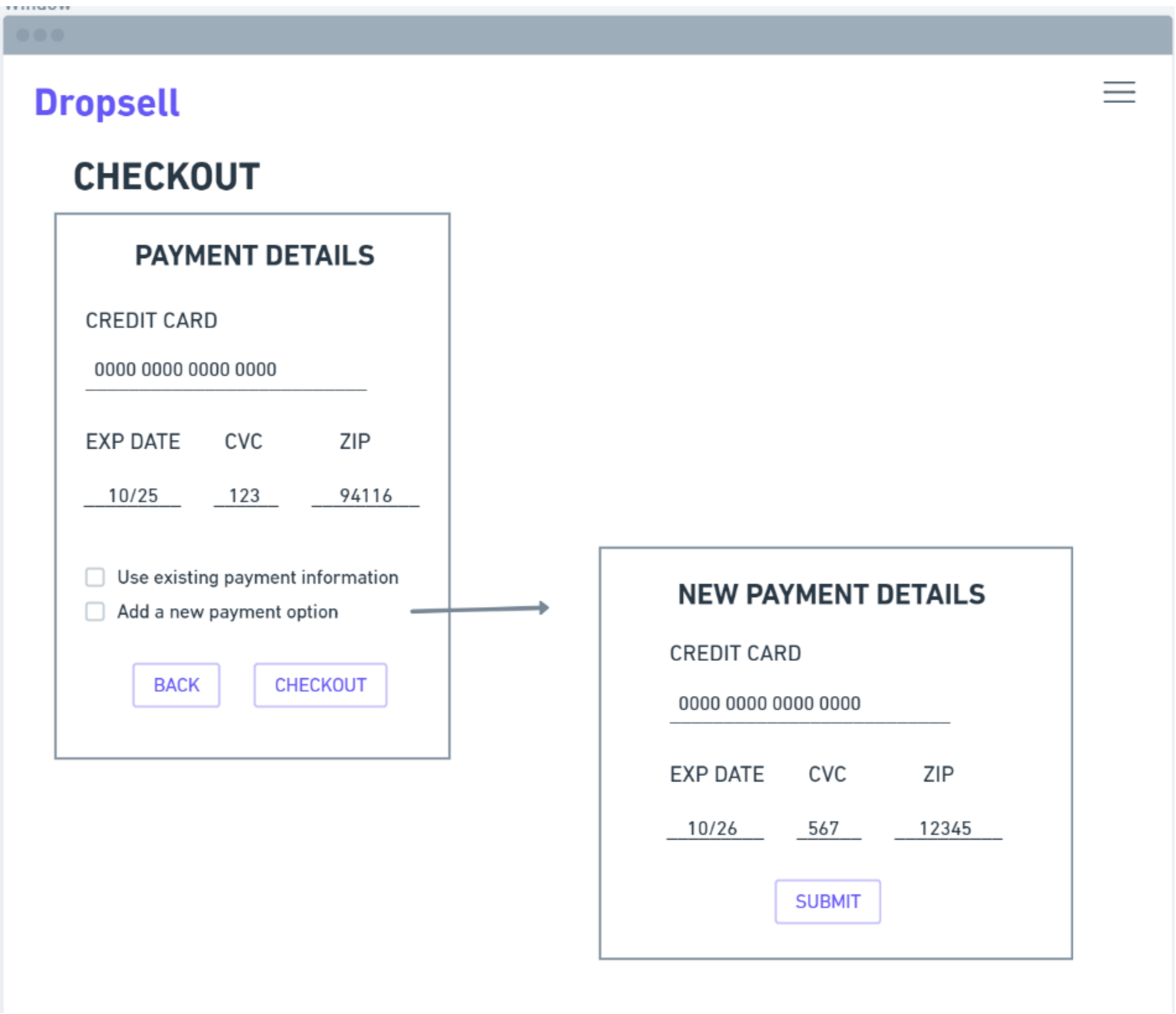
TRAFFIC

PAGE VIEWS: 343 ↑3%
SALES RATE: 5% ↑2%

MOST VIEWED ITEM

Super Nintendo Controller


Window
Dropsell
CREATE LISTING
TITLE:
CATEGORY:
CONDITION:
FORMAT:
PRICE:
$
QUANTITY:
DESCRIPTION:
SHIPPING:
FEES
BOOST YOUR LISTING
LIST ITEM
SAVE DRAFT
CANCEL



Dropsell

CHECKOUT

PAYMENT DETAILS

CREDIT CARD

0000 0000 0000 0000

EXP DATE	CVC	ZIP
10/25	123	94116

□ Use existing payment information

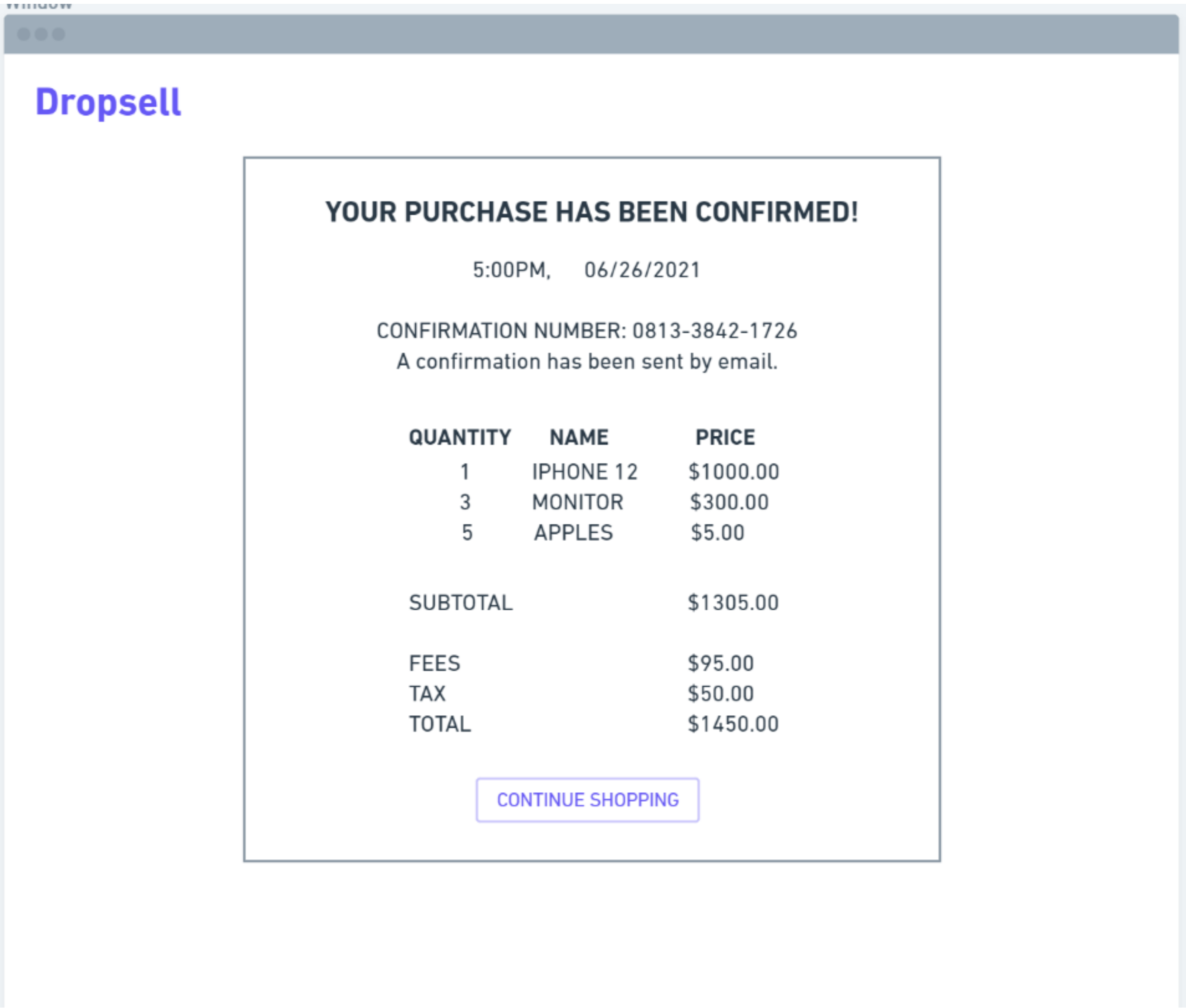
□ Add a new payment option

NEW PAYMENT DETAILS

CREDIT CARD

0000 0000 0000 0000

EXP DATE	CVC	ZIP
10/26	567	12345

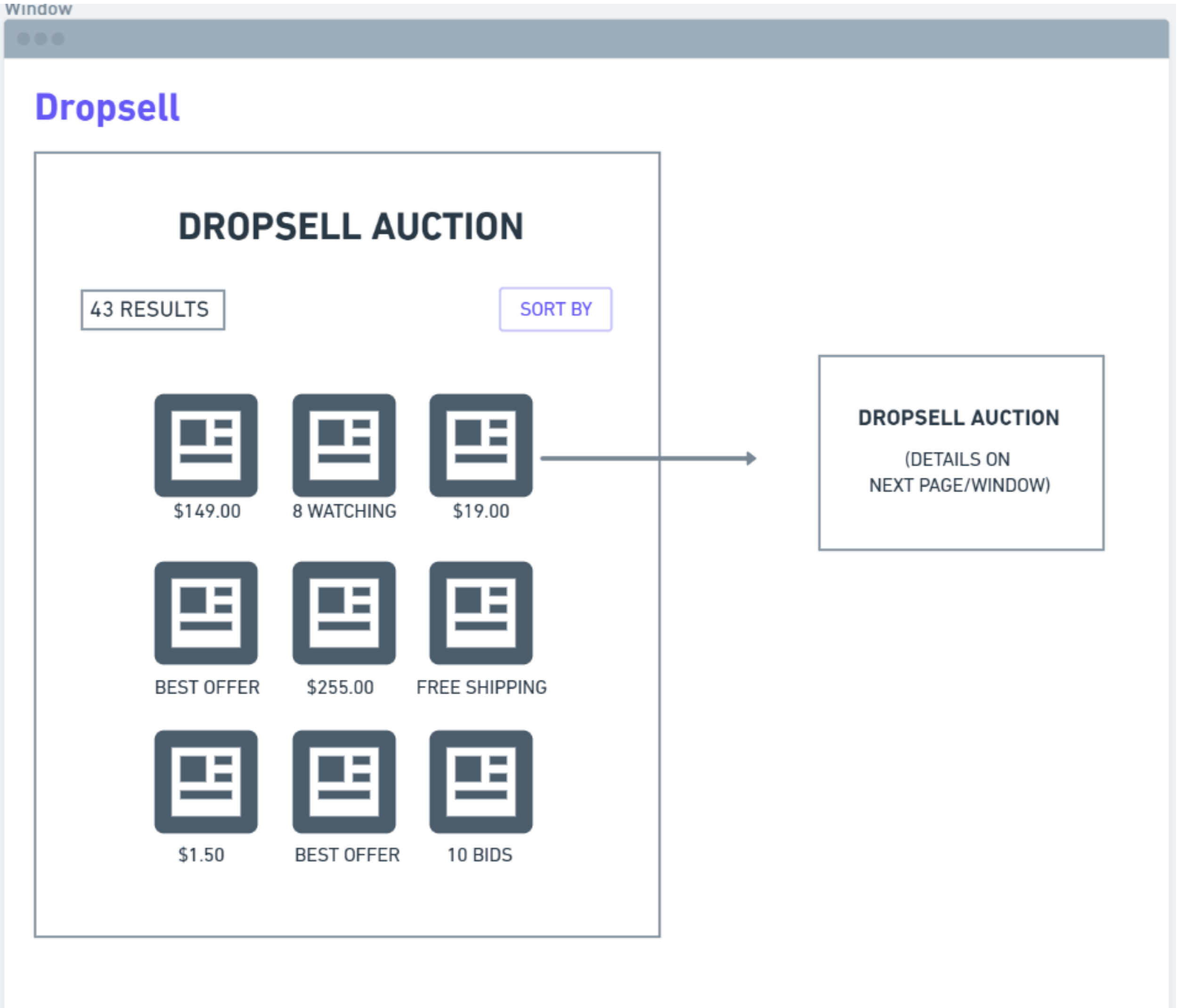


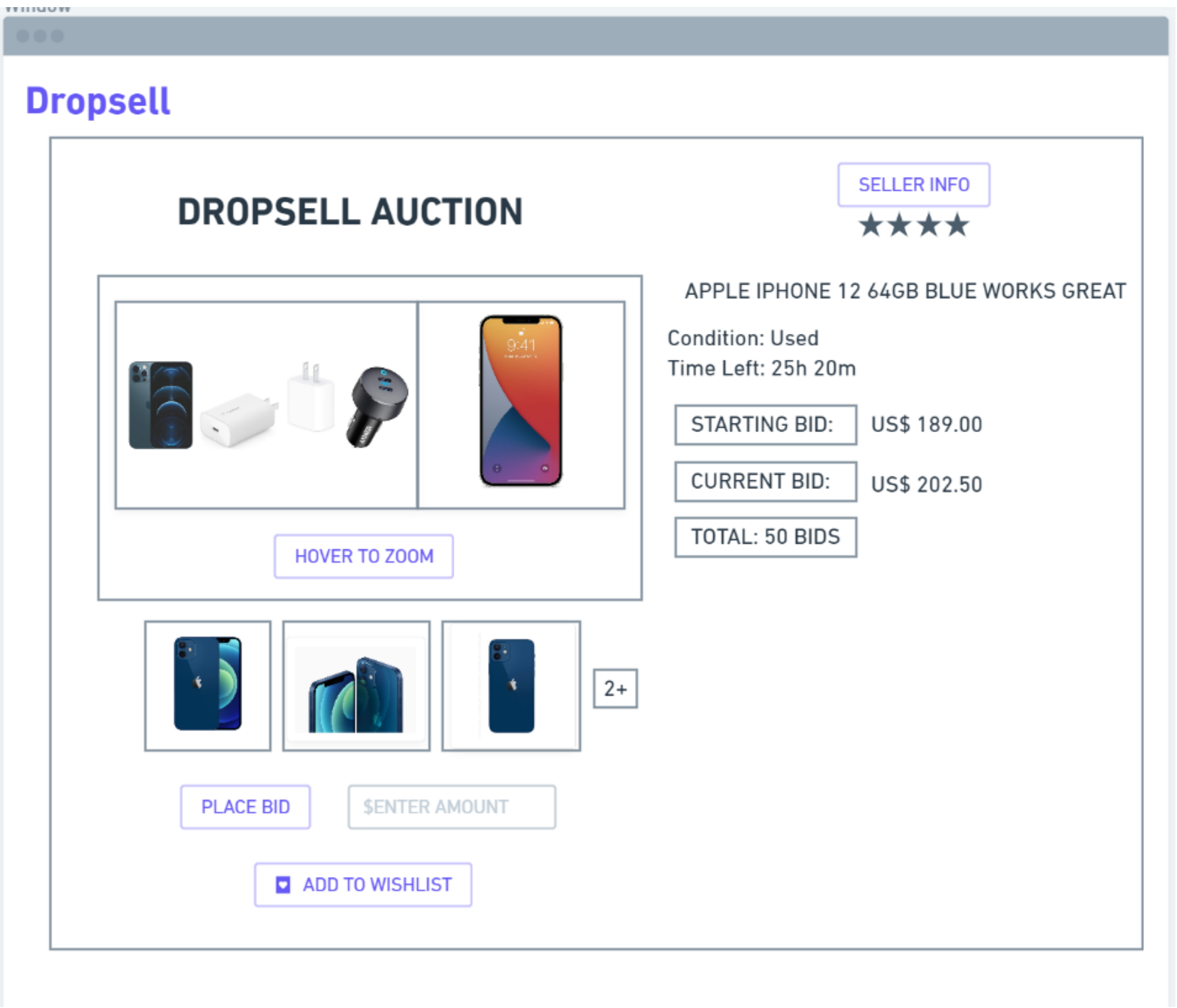
YOUR PURCHASE HAS BEEN CONFIRMED!

5:00PM, 06/26/2021

CONFIRMATION NUMBER: 0813-3842-1726
A confirmation has been sent by email.

QUANTITY	NAME	PRICE
1	IPHONE 12	$1000.00
3	MONITOR	$300.00
5	APPLES	$5.00
SUBTOTAL		$1305.00
FEES		$95.00
TAX		$50.00
TOTAL		$1450.00


Window
Dropsell
DROPSELL AUCTION
43 RESULTS
SORT BY
$149.00
8 WATCHING
$19.00
BEST OFFER
$255.00
FREE SHIPPING
$1.50
BEST OFFER
10 BIDS
DROPSELL AUCTION
(DETAILS ON
NEXT PAGE/WINDOW)



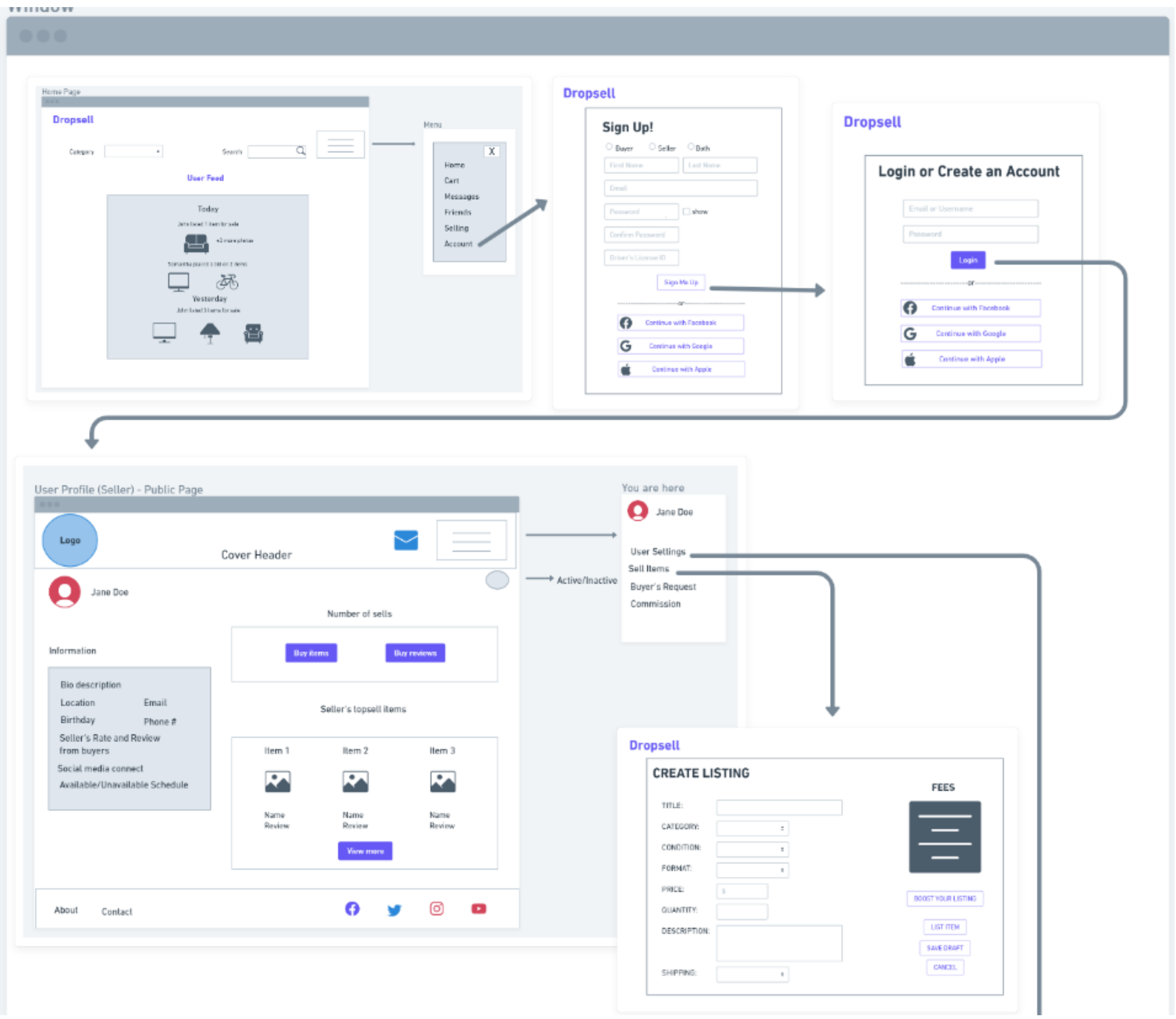
DROPSELL AUCTION

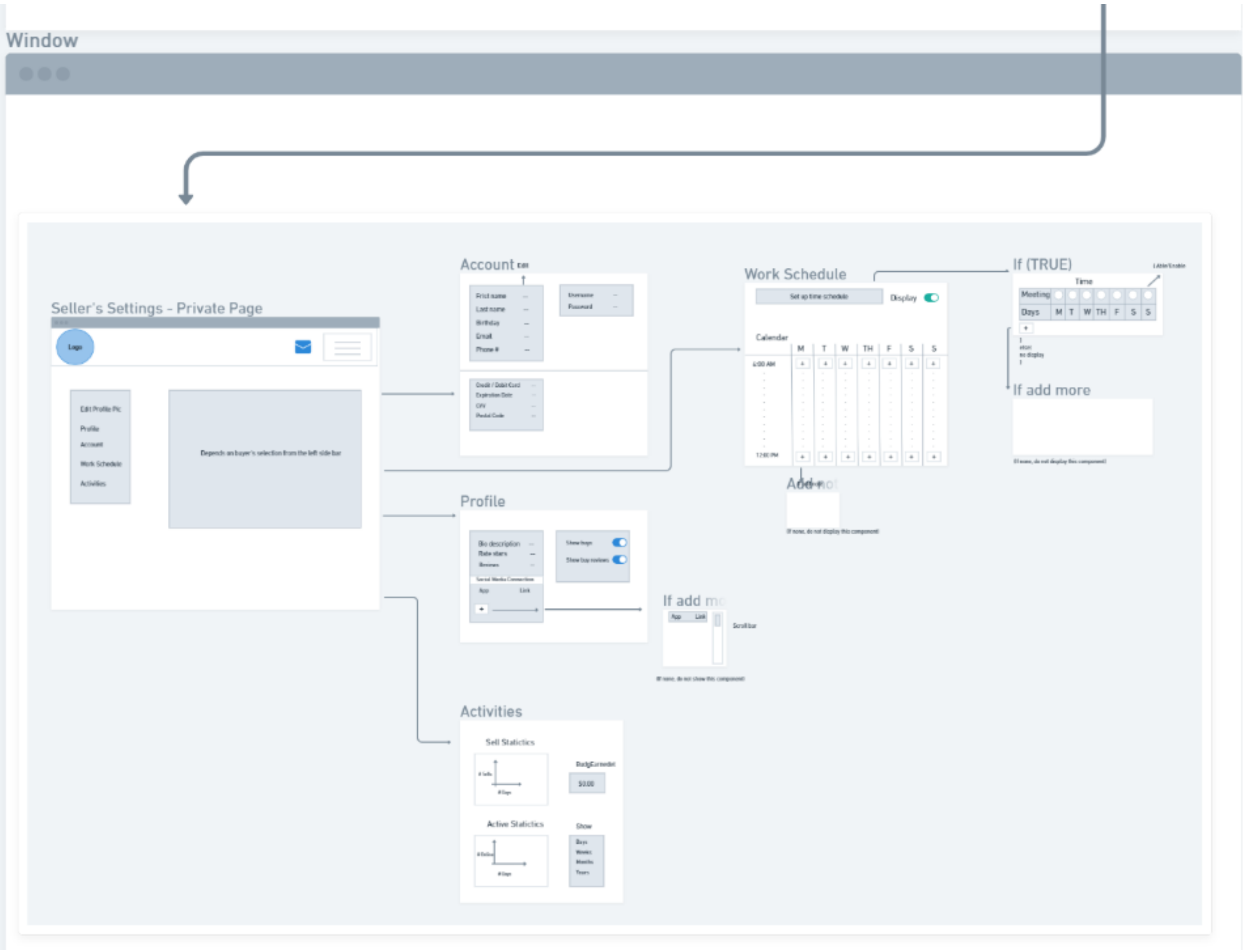
APPLE IPHONE 12 64GB BLUE WORKS GREAT

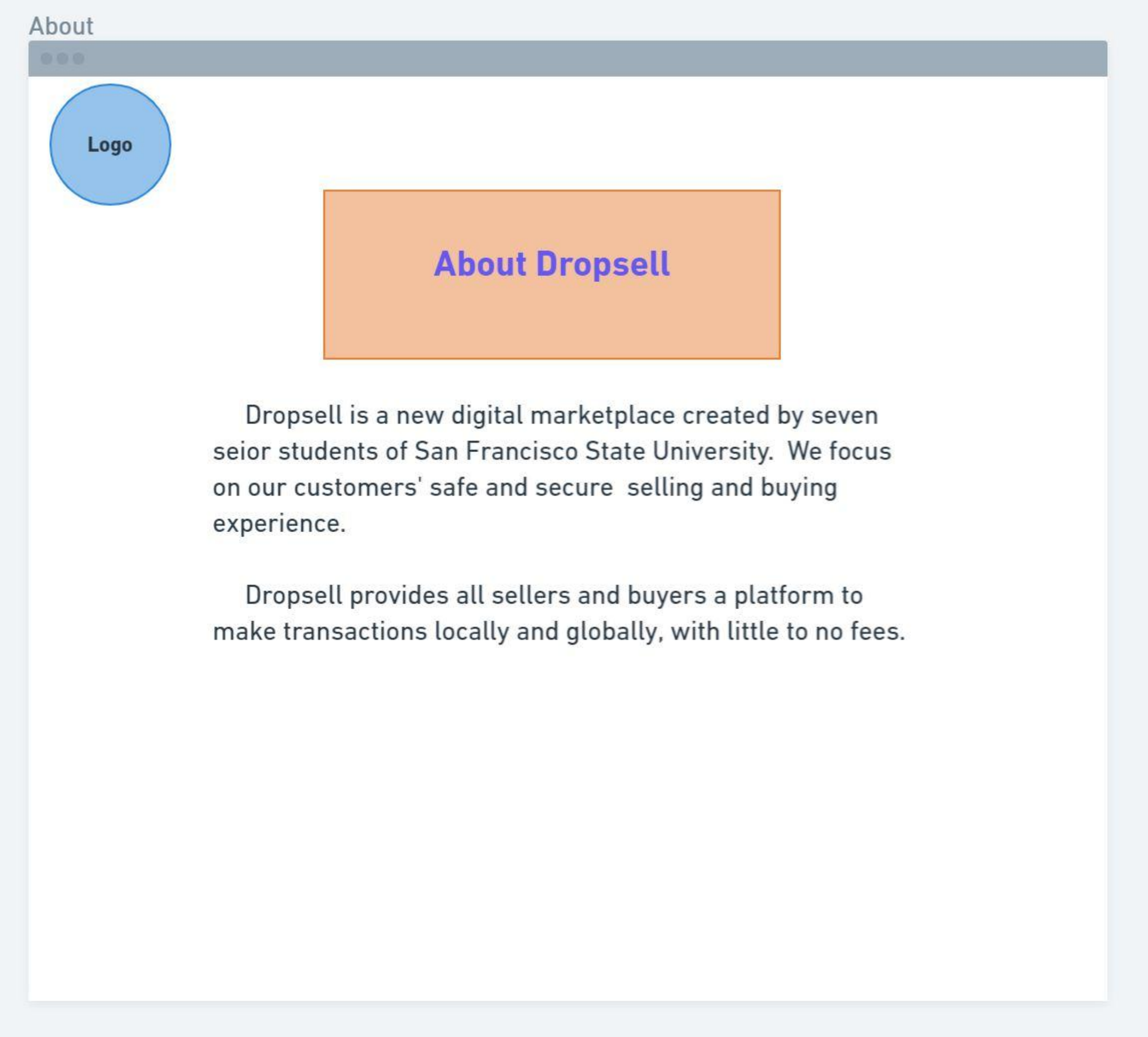
Condition: Used
Time Left: 25h 20m

STARTING BID: US$ 189.00
CURRENT BID: US$ 202.50
TOTAL: 50 BIDS

login -> register -> profile -> seller settings -> create product



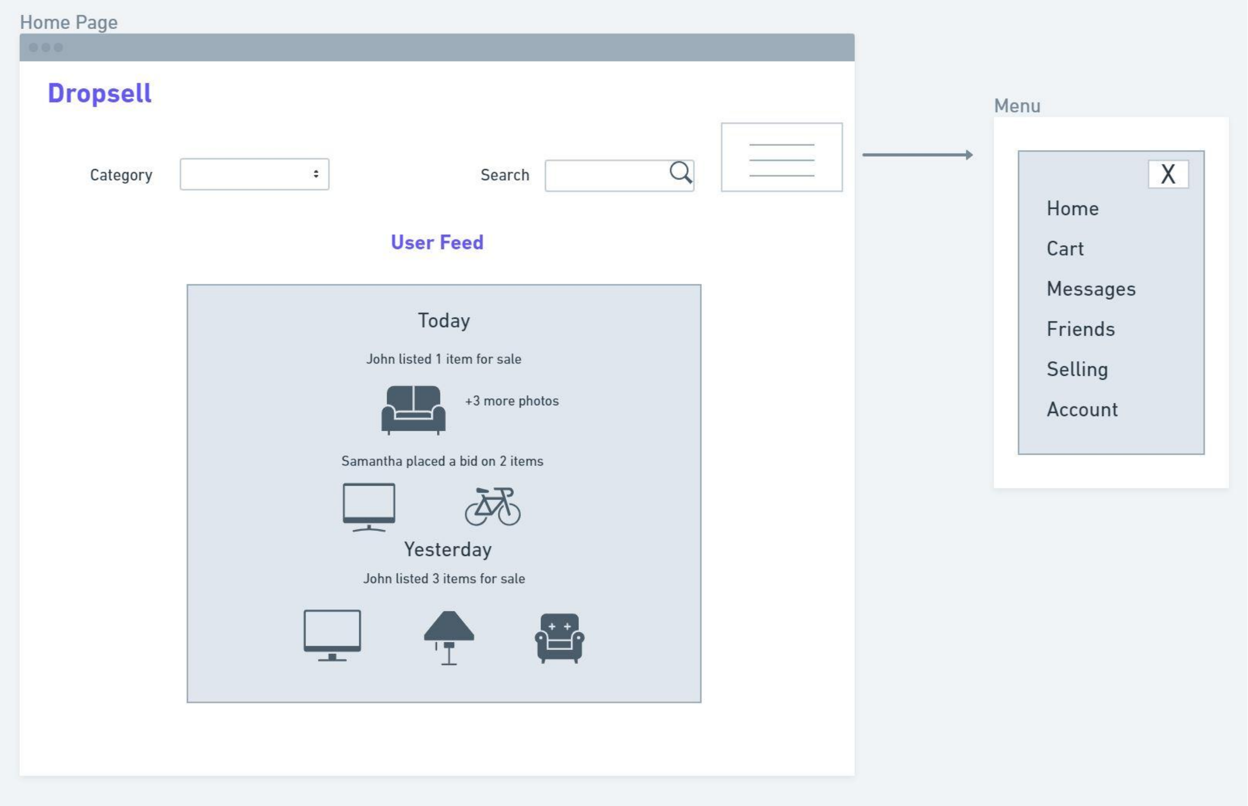

Window
Seller's Settings - Private Page
Logo
Edit Profile Pic
Profile
Account
Work Schedule
Activities
Depends on buyer's selection from the left side bar
Account Edit
First name —
Last name —
Birthday —
Email —
Phone # —
Username —
Password —
Credit / Debit Card —
Expiration Date —
CVV —
Postal Code —
Profile
Bio description —
Rate stars —
Reviews —
Social Media Connection
App
Link
+
Show logo
Show buy reviews
If add mo
App
Link
Scroll bar
If none, do not show this component
Activities
Sell Statictics
# Sells
# Days
BudgEarnedet
$0.00
Active Statictics
# Online
# Days
Show
Days
Weeks
Months
Years
Work Schedule
Set up time schedule
Display
Calendar
M
T
W
TH
F
S
S
6:00 AM
12:00 PM
+
Add mo
If none, do not display this component
If (TRUE)
Able/Enable
Time
Meeting
Days
M
T
W
TH
F
S
S
+
else
no display
If add more
(If none, do not display this component)



About Dropsell

Dropsell is a new digital marketplace created by seven seior students of San Francisco State University. We focus on our customers' safe and secure selling and buying experience.

Dropsell provides all sellers and buyers a platform to make transactions locally and globally, with little to no fees.



Today

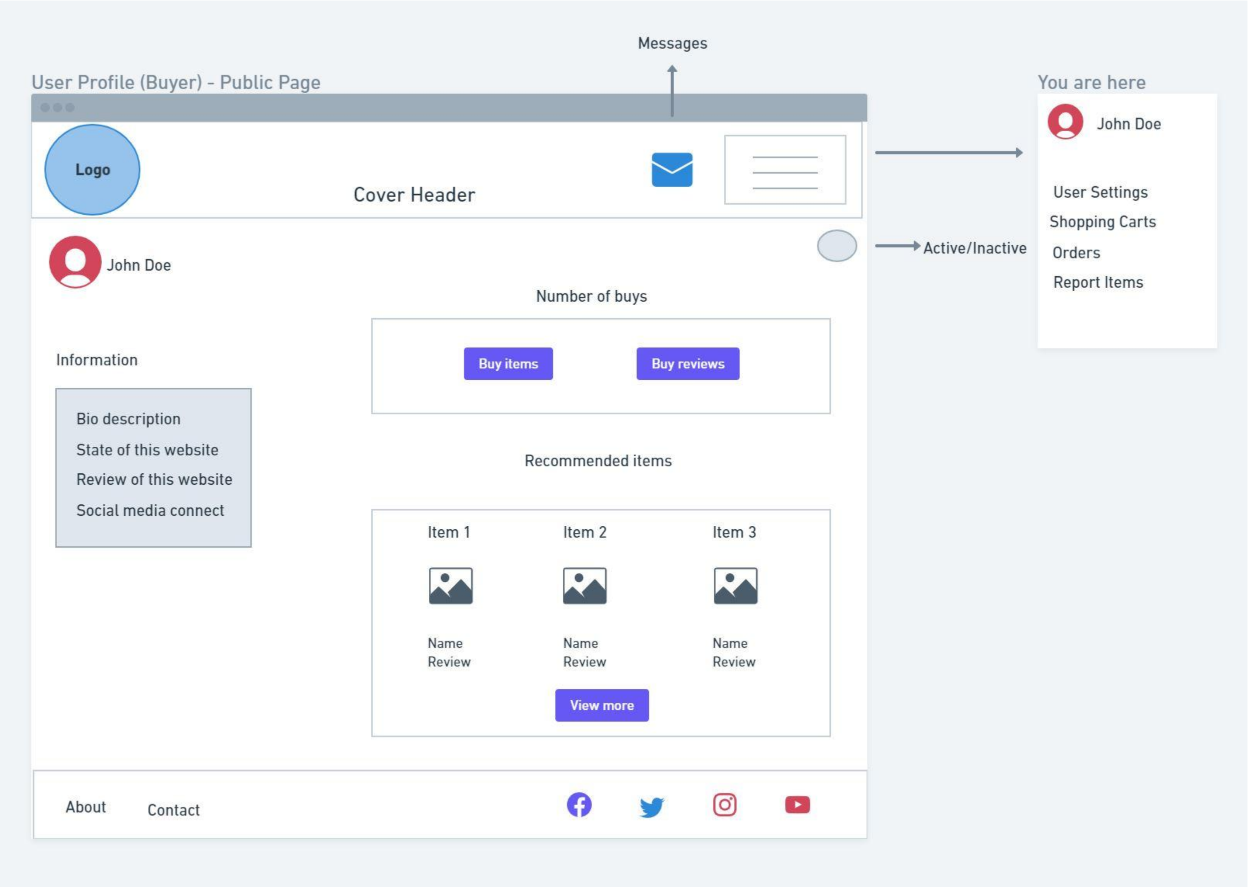
John listed 1 item for sale

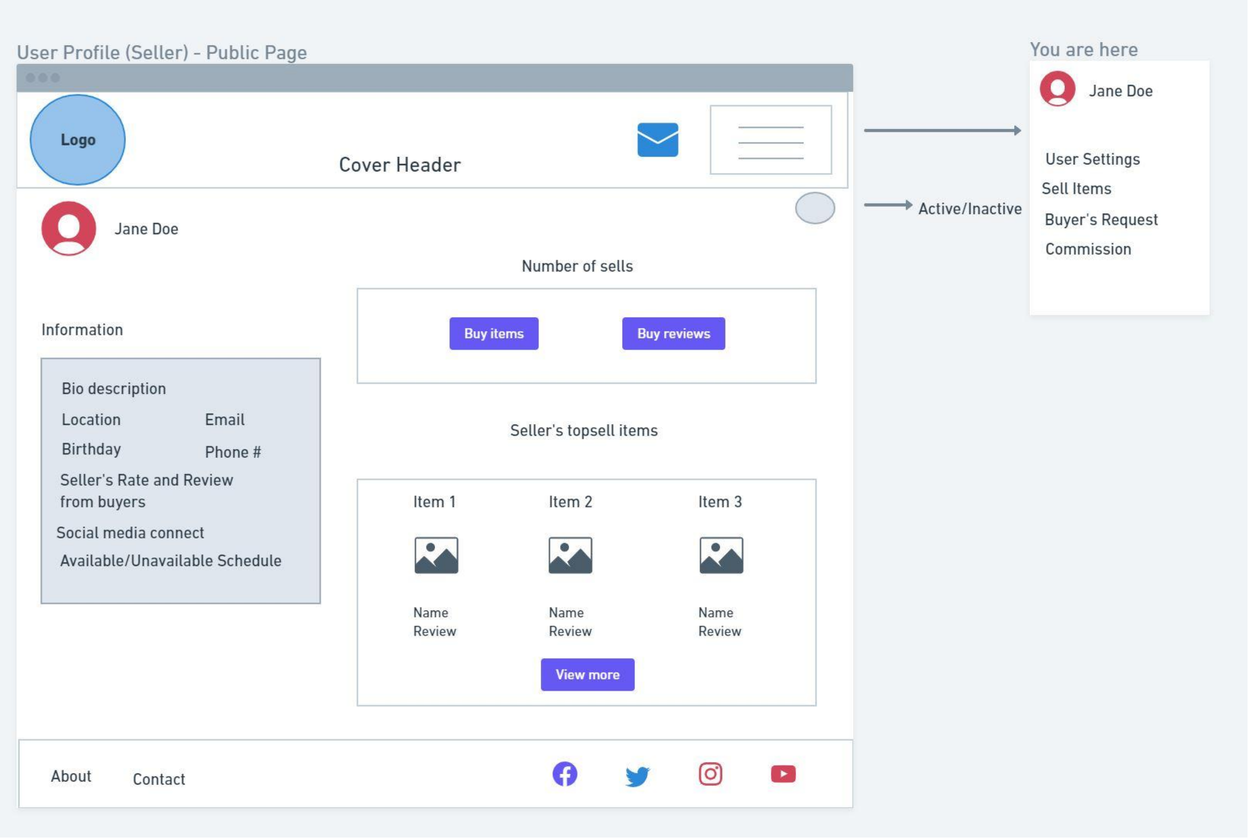
+3 more photos

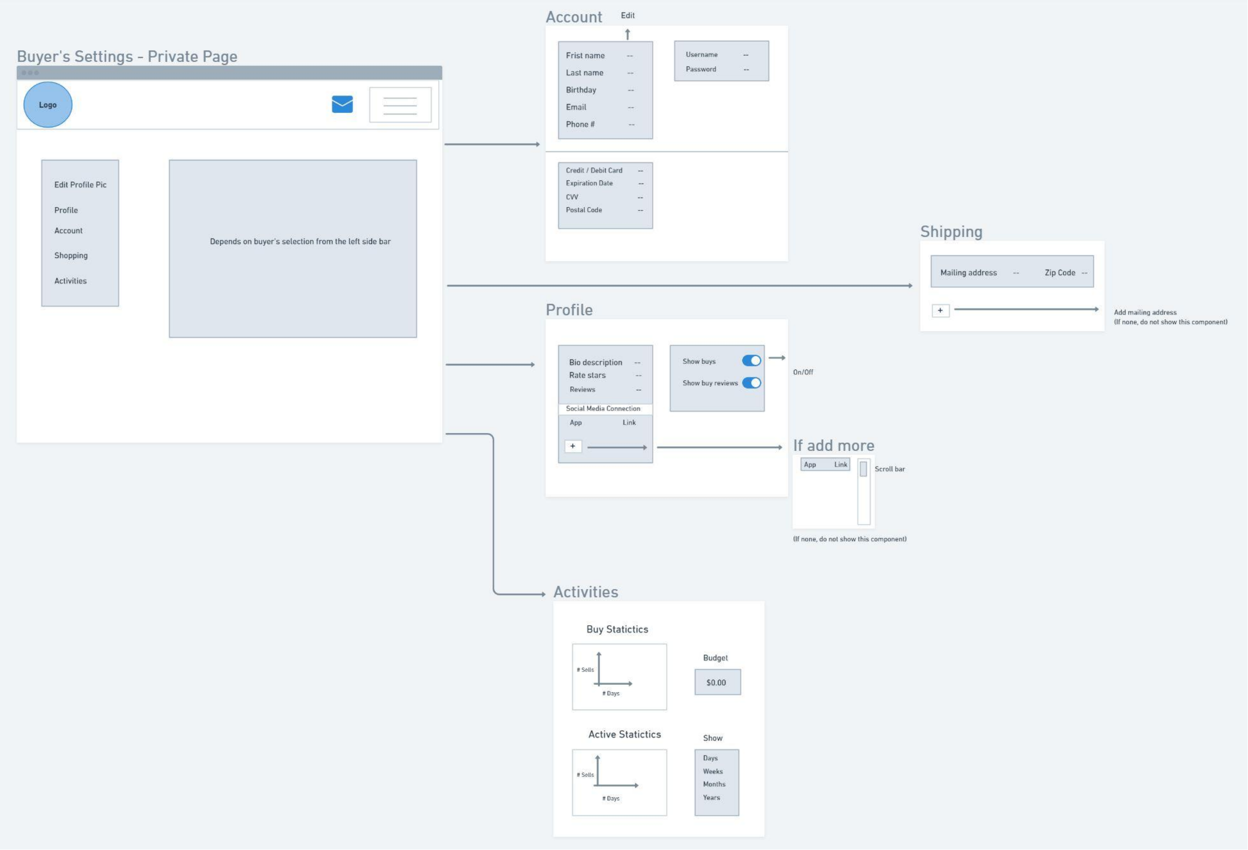
Samantha placed a bid on 2 items

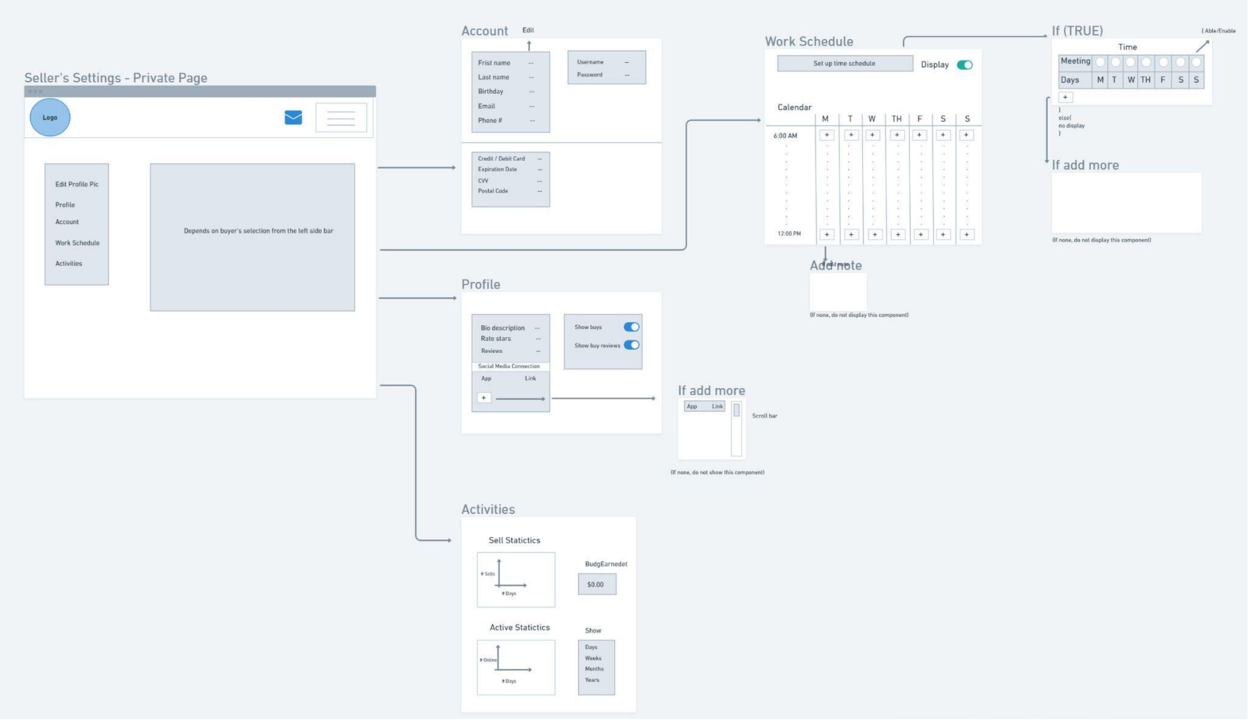
Yesterday

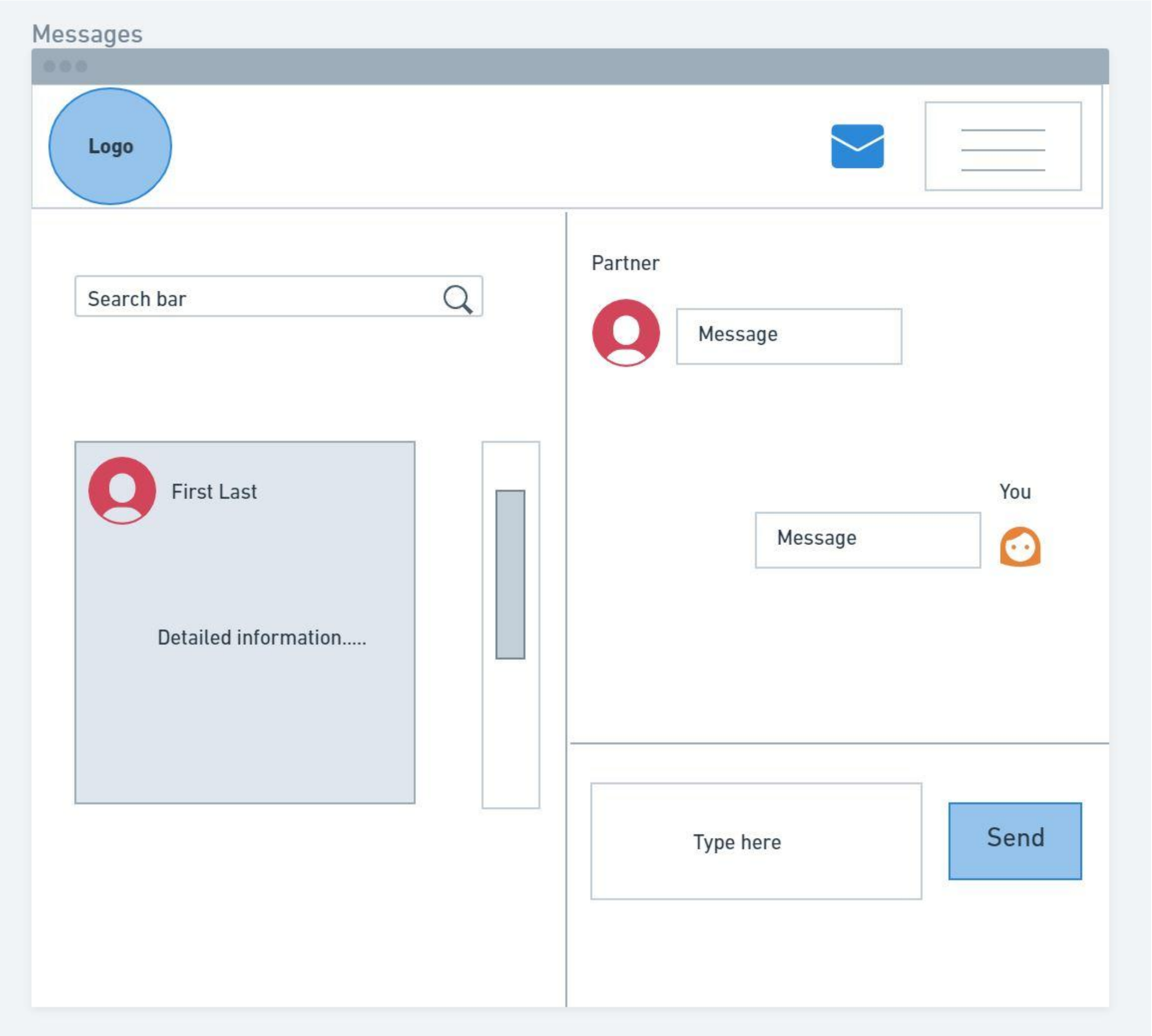
John listed 3 items for sale

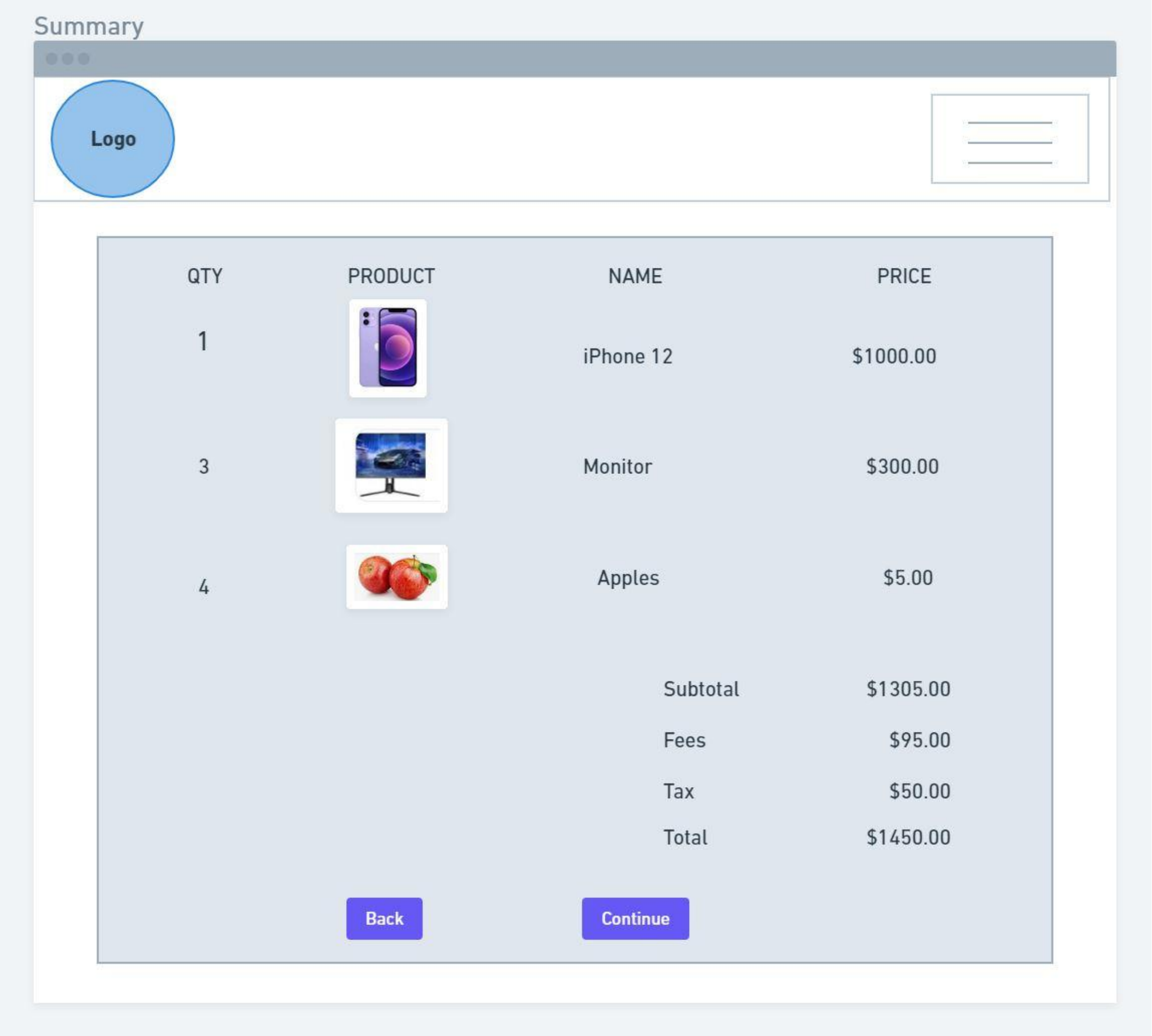

Messages
User Profile (Buyer) - Public Page
Logo
Cover Header
You are here
John Doe
User Settings
Shopping Carts
Orders
Report Items
Active/Inactive
John Doe
Number of buys
Buy items
Buy reviews
Information
Bio description
State of this website
Review of this website
Social media connect
Recommended items
Item 1
Item 2
Item 3
Name
Review
Name
Review
Name
Review
View more
About
Contact


User Profile (Seller) - Public Page
Logo
Cover Header
Jane Doe
Information
Bio description
Location
Email
Birthday
Phone #
Seller's Rate and Review from buyers
Social media connect
Available/Unavailable Schedule
Number of sells
Buy items
Buy reviews
Seller's topsell items
Item 1
Item 2
Item 3
Name
Review
Name
Review
Name
Review
View more
About
Contact
Active/Inactive
You are here
Jane Doe
User Settings
Sell Items
Buyer's Request
Commission


Buyer's Settings - Private Page
Logo
Edit Profile Pic
Profile
Account
Shopping
Activities
Depends on buyer's selection from the left side bar
Account
Edit
Frist name --
Last name --
Birthday --
Email --
Phone # --
Username --
Password --
Credit / Debit Card --
Expiration Date --
CVV --
Postal Code --
Shipping
Mailing address --
Zip Code --
+
Add mailing address
(If none, do not show this component)
Profile
Bio description --
Rate stars --
Reviews --
Social Media Connection
App
Link
+
Show buys
Show buy reviews
On/Off
If add more
App
Link
Scroll bar
(If none, do not show this component)
Activities
Buy Statictics
# Sells
# Days
Budget
$0.00
Active Statictics
# Sells
# Days
Show
Days
Weeks
Months
Years


Seller's Settings - Username Page
Logo
Edit Profile Pic
Profile
Account
Work Schedule
Activities
Depends on buyer's selection from the left side bar
Account
Edit
Frist name --
Last name --
Birthday --
Email --
Phone # --
Username --
Password --
Credit / Debit Card --
Expiration Date --
CVV --
Postal Code --
Work Schedule
Set up time schedule
Display
Calendar
M
T
W
TH
F
S
S
6:00 AM
12:00 PM
+
Add note
If add note
(If none, do not display this component)
If (TRUE)
{ Able/Enable
Time
Meeting
Days
M
T
W
TH
F
S
S
+
}
else{
no display
}
If add more
(If none, do not display this component)
Profile
Bio description --
Rate stars --
Reviews --
Social Media Connection
App
Link
+
Show buys
Show buy reviews
If add more
App
Link
Scroll bar
(If none, do not show this component)
Activities
Sell Statictics
# Sells
# Days
BudgEarnedet
$0.00
Active Statictics
# Online
# Days
Show
Days
Weeks
Months
Years

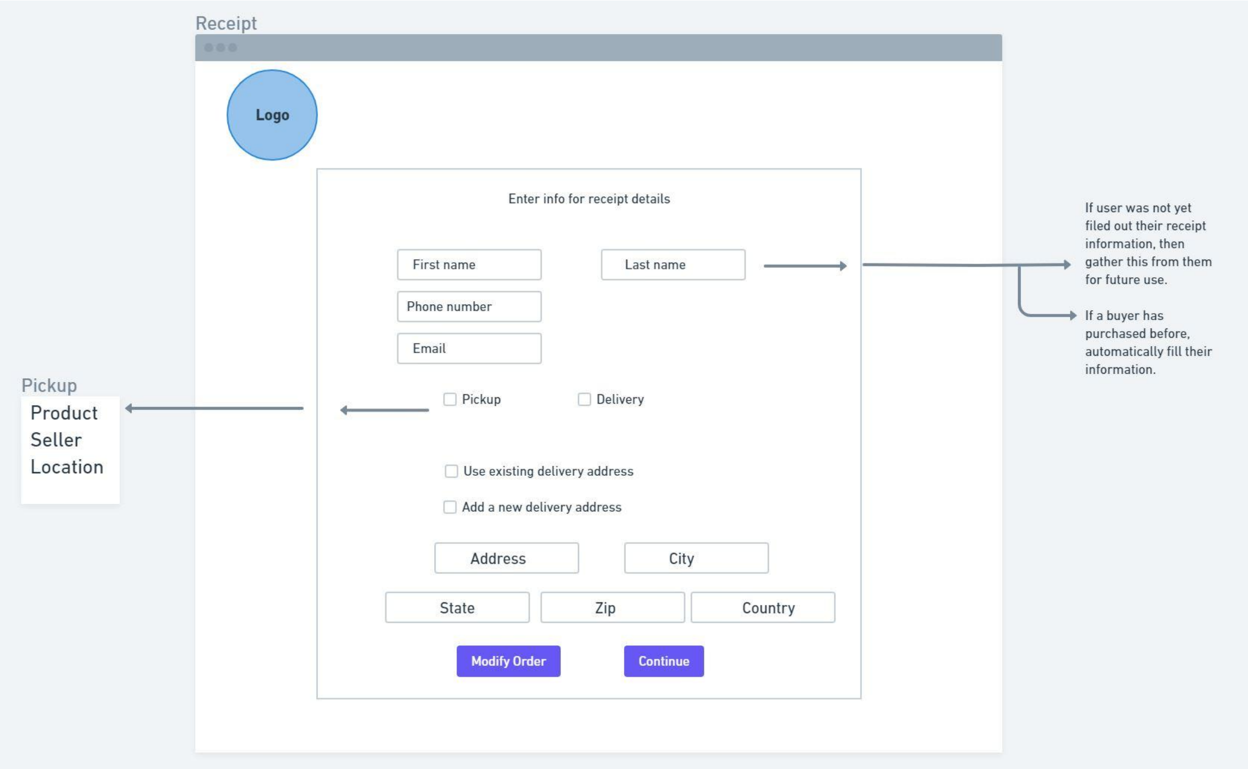

Messages
Logo
Search bar
First Last
Detailed information.....
Partner
Message
You
Message
Type here
Send



Summary

QTY	PRODUCT	NAME	PRICE
1		iPhone 12	$1000.00
3		Monitor	$300.00
4		Apples	$5.00

Subtotal	$1305.00
Fees	$95.00
Tax	$50.00
Total	$1450.00



If user was not yet filed out their receipt information, then gather this from them for future use.

If a buyer has purchased before, automatically fill their information.

4. High Level Database Architecture and Organization

Business Rules

1. One registered user can create zero, one, or many products. A registered user shall upload at least one image, many images are optional.
2. One registered user can have zero, one, or many seller ratings. Registered users can optionally keep track of the rating data to display in seller analytics.
3. One registered user can have one session assigned to them. User sessions can optionally store session data related to the user, in addition to mandatory session id and expiration values.
4. One registered user can have one shopping cart. It is optional for registered users to add products into their shopping cart, it is not necessary for records to exist in a user's shopping cart before a user is registered.
5. Many registered users can create zero, one, or many product comments. Registered users can optionally edit or delete their comment data.
6. Many registered users can create zero, one, or many conversations. It is optional for registered users to have conversations with other registered users, they should still have full functionality of the website.

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
        iii. conversations, messages
        iv. meetups
        v. seller_ratings
        vi. 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. user
7. Marketplace 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

v. top_purchased_products
vi. daily_deal_products
vii. shipping_products

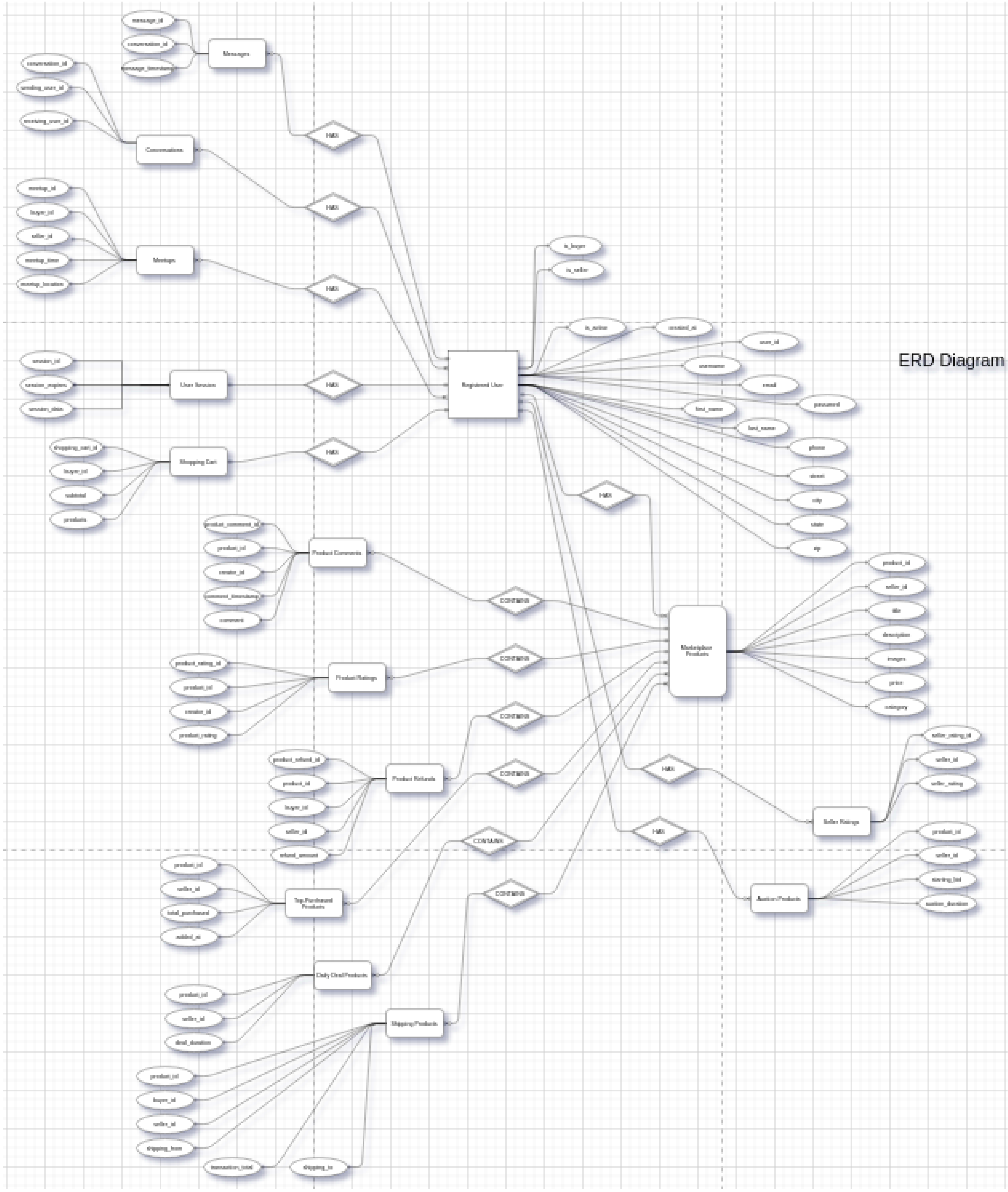
8. Product comments
   a. Attributes
      i. product_comment_id
      ii. creator_id
      iii. product_id
      iv. comment_timestamp
      v. comment
   b. Relationships:
      i. products
      ii. users
9. Product ratings
   a. Attributes
      i. product_rating_id
      ii. product_id
      iii. creator_id
      iv. product_rating
   b. Relationships
      i. products
      ii. users
10. 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
11. Auction products
   a. Attributes
      i. product_id
      ii. seller_id
      iii. starting_bid
      iv. auction_duration
   b. Relationships
      i. products
      ii. users
12. Top-purchased products
   a. Attributes
      i. product_id
      ii. seller_id
      iii. total_purchased
      iv. added_at
   b. Relationships
      i. products

        ii. users

13. Daily deal products
    a. Attributes
        i. product_id
        ii. seller_id
        iii. deal_duration
    b. Relationships
        i. products
        ii. users
14. 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

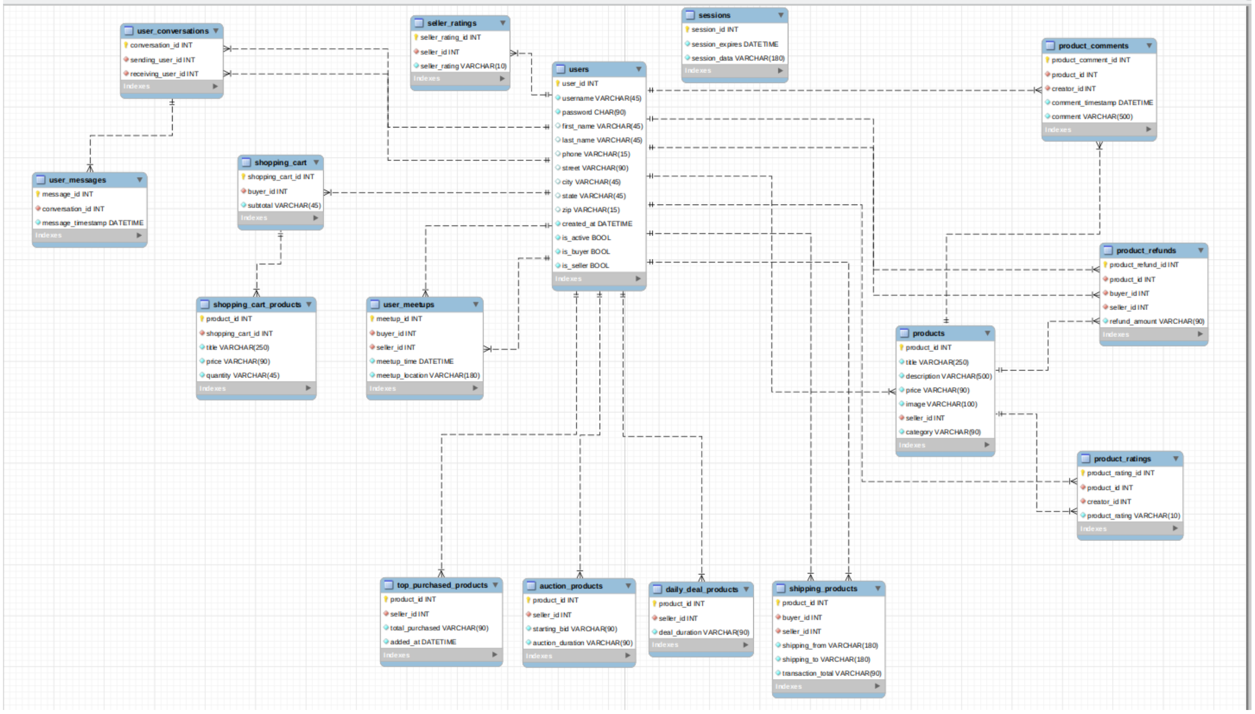
Entity Relationship Diagram (User)

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.

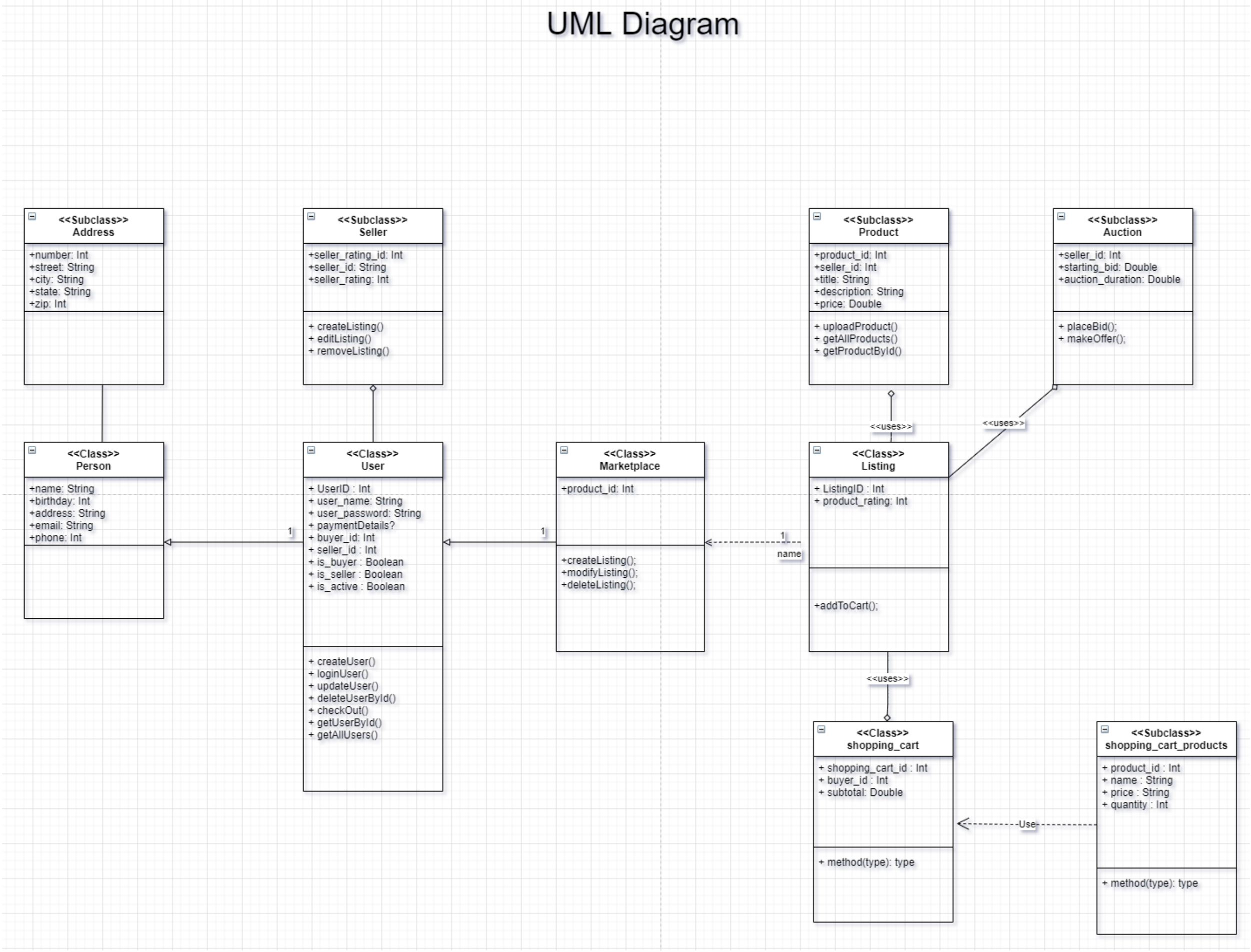
Search/filter architecture and implementation

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 Diagrams

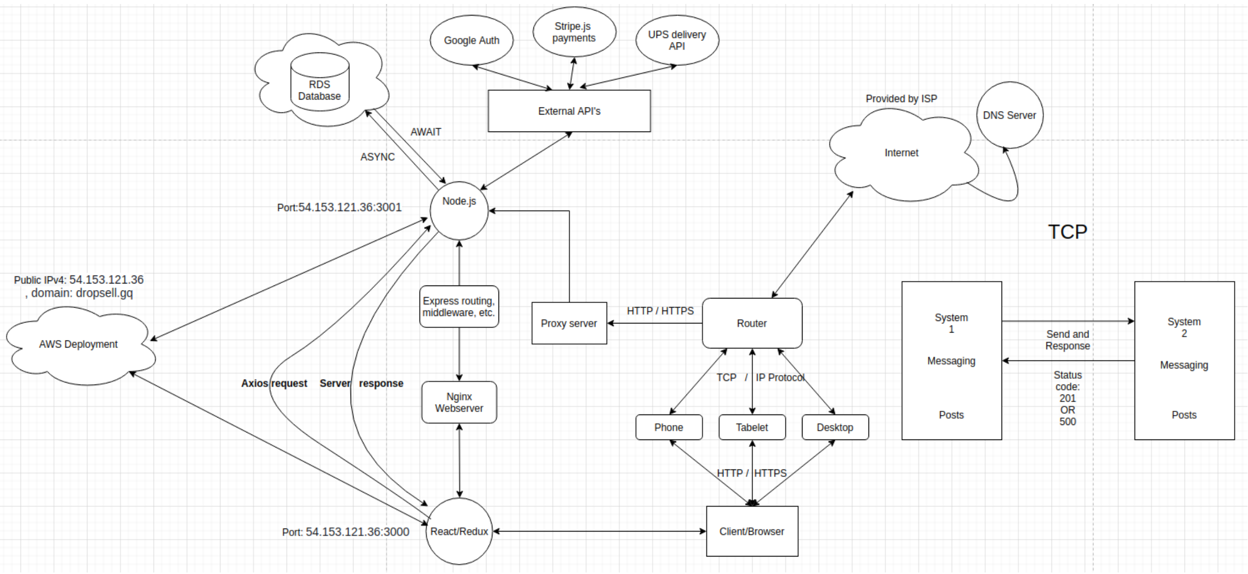
UML Diagram

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



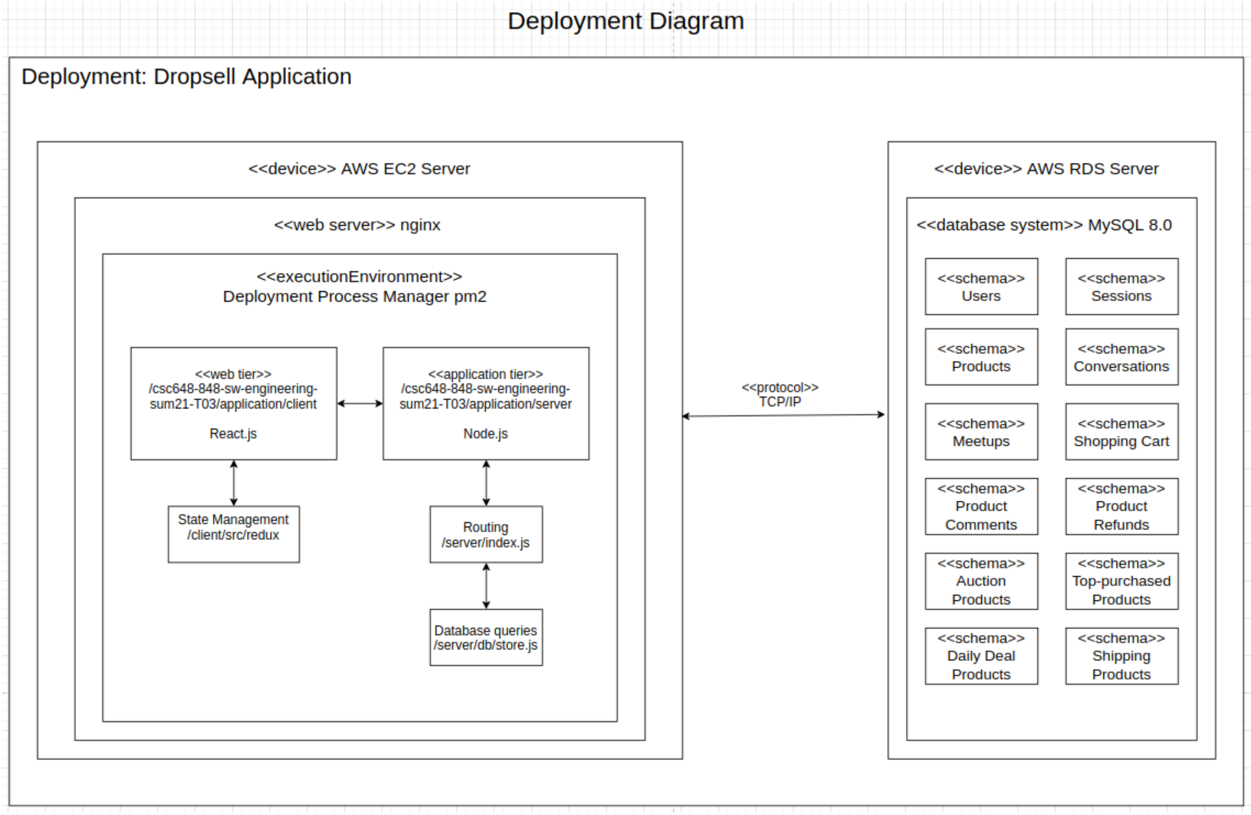
Application Network Diagram

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



Deployment diagram

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


Deployment Diagram
Deployment: Dropsell Application
<<device>> AWS EC2 Server
<<web server>> nginx
<<executionEnvironment>>
Deployment Process Manager pm2
<<web tier>>
/csc648-848-sw-engineering-sum21-T03/application/client
React.js
<<application tier>>
/csc648-848-sw-engineering-sum21-T03/application/server
Node.js
State Management
/client/src/redux
Routing
/server/index.js
Database queries
/server/db/store.js
<<protocol>>
TCP/IP
<<device>> AWS RDS Server
<<database system>> MySQL 8.0
<<schema>> Users
<<schema>> Sessions
<<schema>> Products
<<schema>> Conversations
<<schema>> Meetups
<<schema>> Shopping Cart
<<schema>> Product Comments
<<schema>> Product Refunds
<<schema>> Auction Products
<<schema>> Top-purchased Products
<<schema>> Daily Deal Products
<<schema>> Shipping Products

6. List of Contributions

Student Name	Contributions
Mitchel Baker	Mitchel started off Milestone 3 by setting up the redux actions/reducers for register, login, and products components. He also set up the backend routes for these components, which fetched specific data and updated these values in the database for future use. Mitchel also worked with Kenneth to create the buyer and seller settings pages. Mitchel also implemented Dropsell’s checkout experience pages: receipt info, summary, checkout, and final invoice, in addition to creating dynamic routes after clicking on products in the Home page. Lastly, he contributed to overall styling of Dropsell while also deploying our codebase to our AWS EC2 instance.
Charmaine Eusebio	Charmaine helped tremendously by researching the best wireframe software tools to use. She communicated what worked best for her, which was the website whimsical.com. As a result of her research efforts, she ended up creating clean wireframes which expressed exactly how we want our UI/UX to look. Charmaine also collaborated with Rowena to effectively split up the wireframes to do, which demonstrates initiative behind the tasks for the Milestone they were assigned to accomplish.
Kenneth N Chuson	Kenneth played a pivotal role in the creation of our buyer and seller settings pages. He took initiative by adding libraries for seller scheduling while also jumpstarting our efforts towards implementing user analytics with the graph library he added. There were a few initial bugs with the buyer and seller settings pages, but Kenneth and Mitchel worked together on this to

	solve the problem. Kenneth implemented many of our application's new components, ranging from Activities, Account, Profile, Shipping, and WorkSchedule. Kenneth also implemented the redux actions/reducers for the buyer and settings pages. Kenneth also added the seller and buyer settings information into our data definitions section.
Krina Bharatbhai Patel	Krina's efforts were key toward refining our Home page features. She started with a revamp of the styling we had, and then moved towards updating our NavBar component to include the sliding hamburger menu. Krina performed research on the best react UI libraries to use in regards to our NavBar component, while also implementing dynamic react components which open/close on click. After completing the NavBar, Krina also implemented our additional search filters, ranging from location, price, shipping, to condition of products.
Michael Schroeder	Michael started off Milestone 3 by updating our register UI, he also fixed the register actions for when users insert data into form inputs. As github master, Michael also helped to manage commits from branches by fixing merges and creating pull requests for features being created. Whenever there were merge conflicts, Michael communicated them to Mitchel where they were then able to solve the conflict efficiently with no time wasted. Michael has also taken initiative with our chat functionality in the application. He researched two different libraries for doing so which were cometchat and socket.io.
Rowena Elaine Echevarria	Rowena contributed to our register functionality by adding the Driver's license number input, in addition to adding the corresponding redux actions/reducer functions for this input.

	Rowena was also assigned to do the wireframing for our application, where she effectively split up tasks with Charmaine in order to get the job done. Rowena was at every team meeting, and always communicated the status of her progress through our Discord channels.
Jamie Dominic Walker	Jamie took initiative for Milestone 3 by updating our users schema in the database with additional information such as the user’s birthdate. Jamie has also been working on implementing the Stripe API for our user checkout experience. He has been conducting a ton of research on best practices in regards to securely charging users for products they purchase. Jamie has also shared all of the information he’s found in our Discord channels while also communicating to us his findings.