SW Engineering CSC648-848-05 Fall2023

OrderOwl

Team 06

Team Lead: Rita Belú Velazco, rvelazco@mail.sfsu.edu

Front End Lead: Jimmy Pan

Back End Lead: Tin Nguyen

GitHub Master: Luis Ramirez

Document Editor: Mankit Yeung

M2 Editor: Komaldeep Kaur

Database Master: David Lien

Milestone 2

October 10, 2023

History Table:

Revision	Date
M2V2	-
M2V1	10/12/23
M1V2	10/12/23
M1V1	09/21/23

Table of Contents

I. Data Definitions:

a. **Registration (passwordID, UserID, emailID, firstNameId, lastNameId)**
User shall be able to register to our site to be able to use our services with OrderOwl. We shall request the user a password, username, email. We need the username to be unique in order for no confusion to happen with other accounts that have been created. For the user's password the user shall create a password that must have at least one capital letter, special character, and a minimum of 10 characters. We shall also request an email as well in order send an email with a confirmation link to finalize the creation of their account. Finally, the user's email shall be used to send updates about their packages.

b. **User login (password, username, email) (UserId, userpswd, userEmail)**
We shall store our users information in our database in order for their information to be stored with us. The user shall use their information to log back in to OrderOwl and to access their packages. All of the users information will be stored in our database encrypted for the safety of our users. Our team shall use "Let's Encrypt" in order to encrypt the users information in case of any hacking attempts to our software.

c. **Tracking numbers and links (trackNum, trackLink)**
A lot of the websites we're working with either have a tracking number or a link where the user can keep track of their orders. This will also help our team to make sure everything is being tracked and updated accordingly. Users shall also be able to see their tracking numbers on the app in case they want to share it with anyone else.

d. **Amount of orders/Order History**
Our users might make more than one order and would want to keep track of all of them. We will store the number of items they've ordered to keep count and to be able to show them more than one order. This also helps the user to have a list of all their orders all in one website without having to switch. The user will have no limit on the amount of orders on the list. The user shall keep as many as they want and as long as they want, even after the item was delivered. The reason this might be important is because a user might want to check their past orders in case they need to look at some information.

e. **Accounts from different websites**
To make it easier for users we will be storing some of the user's accounts in our database so we can get the information for their orders on their account. This way we can access their tracking numbers or links to keep them updated on their orders. This also can help us in situations where the site might not have some type of tracking number. At this point we're forced to have access to the account to give the user the information they need to see.

f. **Privacy**
We shall include private information in our database that also needs to be encrypted like the passwords of our users. We shall also take precaution when someone tries logging to an account that is not theirs. If by any chance this happens, the user shall receive a notification containing that there was an attempted login, with the time and location of around the area where the attempted login happened. This shall let our user know that they might want to consider changing their password. On top of that, there will also be a notification sent out whenever someone has successfully logged in to the account. This shall also let our user know that someone logged in to their account.

g. **Location**
Users shall be able to see where their package is at and where the package is going to. With the location access, we will be able to notify the users when the package is near the house. An example of this would be when Amazon is delivering to you and you can see where the truck is.

h. **Order Category**
The orders shall have a category depending on the item they ordered. For example whether it was clothing, electronics, or food. This will allow the user to have a better experience navigating through all their orders. There will be no need to scroll through so many orders in order to find a specific one. In case the user doesn't see a category that fits their order, they shall be able to create their own category. This shall work sort of like a music playlist, but this time we are putting in orders to be able to track.

i. **Transaction Amount**
On the orders, the user should be able to see how much they've spent on the item they've Ordered. There could be a case where the user needs to look back at how much they've spent on orders. Being able to see those transactions on one app could help calculate and manage how much you spend on orders.

j. **Order Information**
Users shall see the details about the item they've ordered from the website. This shall provide the user with a description of the item they ordered in case they forget what it was.

k. **Deleted Orders**
Users shall have the option of deleting their past orders in case they don't need it anymore. The software shall store the deleted orders in case for about 7 days and delete

them permanently after. We shall keep an amount of days displayed next to the order before the order information gets permanently deleted from the users account.

l. **Supplier/Website**
We shall be using information from our suppliers and other websites to provide the most information to our users on their orders. While we can just use the tracking number that the user can provide us, that could be a hassle for some people. The easiest way is for users to link their account from the website they ordered. Our users would only have to do this once, no need to be logging in and out all the time trying to remember your password.

m. **Updates**
We shall inform our users of our rolled out updates to keep them updated on what is new. Sometimes when a user runs into a new update it's hard to understand and see what is new. However, if the user is informed with a list of what has been implemented, they can easily access the new features without missing anything.

n. **Notifications**
Users shall receive notifications about their order status and when it arrives so they won’t miss anything. Users might forget that they are receiving an order soon, causing some problems. Their package could get stolen or they could miss the package because they weren’t there to sign it off. This could potentially lead to less packages getting stolen and more people being ready to receive their package.

o. **Customer Support Tickets**
In case a customer has any questions or has any problems, they can start a support ticket with us and we will respond back to them in a timely manner. This shall also benefit us as well because we have a system where we can help our users when they have an error or for some reason they couldn’t get the help they needed through a message.

p. **Customer Reviews**
When a user orders an item we shall gather information from the website they ordered form including the reviews. This shall allow our users to keep an eye on the product if it is good or not. The user shall see a constantly updated review for the item they ordered in case something is wrong with the product. Our users shall also be able to leave a review through our app of the item they’ve purchased so our other users are able to see other peoples experiences.

q. **Data from our customers**
We shall collect data from our customers in order to ensure the best experience for our

Users. We will never give away the information we receive from our users. We will only use it for the benefit of a better experience on OrderOwl.

r. **Administration**
We shall be able to login to our administrative accounts in order to do some testing with features that haven't come out yet. In order to separate basic user features, we shall enable our own accounts as admin accounts. This way whenever we log in to these accounts, we can test these features on our software and as well as communicate with our users if they need any help.

s. **Recommendations**
From the data we collect from our users, we shall give them recommendations of what they might order next. If the user by any chance finds something they would like to purchase on the recommendations, they could click on the item. Our user shall be sent straight to the site where they can add the item to their cart to finalize their purchase. Once they make this purchase, their order should be able to be tracked on OrderOwl once they receive the tracking information.

t. **Order Status**
Our users will have an updated status of their package. This will tell them the time and location of the item. On the list of the user’s packages, they shall be able to see a piece of text letting them know the current status of their package. After more updates on the package start, the user shall see a list of all the updates from newest at the top to oldest at the bottom.

u. **Return/Refund**
If by any chance our user needs to return or refund an item they’ve ordered. There will be an option for them to choose this, it will send them to the suppliers website where they can start a return process. We shall also change the information on how much money they got back from their refund. This shall allow the user to keep track if they ever received the money back from the shipper.

v. **Refund Status**
Once the user starts a refund, they shall be able to track the item if they need to ship it back. Instead of the order showing the item has been delivered, the item will now show the shipping status of the item going back to the shipper. This shall help our users to keep track of their item to make sure they receive their money.

w. **Messages**
Our users shall send developers messages in order to help them out with a problem they

might have. Once the user has sent the message, the user shall receive help regarding the problem their encountering. Administration shall login to their admin accounts in order to help the user resolve their problem.

x. **FAQ (Frequently Asked Questions)**
Some questions that our users will be asked frequently in messages. We will store these in a separate section of our OrderOwl website in case they have any doubts of how to use something. Users shall be able to find their answer quicker than having to wait for a message back from an admin. We shall update this list whenever we notice a pattern of the same question going on.

y. **Payment Status**
Some websites that our users use have the option of setting up a payment plan for their order. They would be able to choose if they want to pay weekly, every other week, or monthly. Whichever plan they choose, we shall keep them updated whether their next payment is coming up or if the payment went through already. This could also notify our user if the payment was successful after they've bought something. If by any chance the payment didn't go through, they will also receive a notice on their payment status.

z. **Links to Suppliers**
We shall include helpful links that send our users to the suppliers website in case they need help from their end or they need to order again. The user shall be able to go on their order to look at further information regarding the link they need. An example of this could be the suppliers page when an item is going back on stock.

aa. **Hidden Packages**
Users shall have the option of hiding a package that they've ordered. If a user needs some sort of privacy for something they ordered, they could hide the package in a hidden tab. An example of this would be if a user orders a gift for someone in the same household but they use their phone. In this case, they could hide the package and limit the notifications that they receive.

bb. **Sharing**
User shall be able to share their tracking information on a certain order that they've purchased. This could be a gift they got for someone and they want to share the information to them so they can be ready to receive the package. The user shall send a link through text message to the other individual that needs the tracking in order for them to see that information.

cc. **Reset Password**

Our users shall have the option of resetting their password and creating a new one. This shall prevent other individuals from logging in to the users account that they might not want on there. User shall also use this tool whenever they receive a notification from us about someone trying to login into their account.

II. Prioritized Functional Requirements:

1, 2, 3. (1- must have; 2 - desired; 3 - opportunistic as defined in the class)

a) Priority 1 (must have):

User:

1.1 User shall be able to track all their packages with tracking information.

1.2 User shall be able to add/delete tracking information.

1.3 User shall be able to login.

1.4 User shall be able to create/delete their account.

1.5 User shall be able to update their profile information.

1.9 User shall be able to sort their tracking information.

1.12 User shall be able to submit a ticket for help.

1.13 User shall be able to send/receive messages.

1.15 User shall be able to delete or edit product reviews they've written.

1.17 User shall be able to rate a product review.

Admin:

3.1 Admin shall be able to access all data.

3.2 Admin shall be able to modify any data.

3.4 Admin shall be able to create any type of account.

3.5 Admin shall be able to send messages.

3.7 Admin shall be able to create a log.

3.11 Admin shall be able to receive and review user feedback and reports regarding system performance and functionality.

System:

4.1 System shall be able to collect traffic data.

4.2 System shall be able to store traffic data to improve user experience.

Notification:

5.1 Notification shall be able to be sent.

5.2 Notification shall be able to be received.

5.3 Notification shall be able to be stored.

5.4 Notification shall be able to be viewed.

5.5 Notification shall be able to be replied to.

5.6 Notification shall be able to be deleted.

5.8 Notification shall be time stamped to indicate the date and time of sending or receiving.

Tracking Information:

6.1 Tracking information shall be able to be stored.

6.2 Tracking information shall be able retrieved in real time according to shipping carrier updates

6.4 Tracking information shall maintain an up to date database.

b) Priority 2 (desired):

User:

1.6 User shall be able to recover password when needed.

1.7 User shall be able to see/hide their history of completed tracked orders.

1.11 User shall be able to write product reviews.

1.14 User shall be able to view/delete messages.

1.16 User shall be able to view a product review other users post.

Admin:

3.6 Admin shall be able to view another account.

3.10 Admin shall have the ability to conduct system maintenance and updates without disrupting user access.

System:

4.5 System shall be able to recommend businesses/products.

4.6 System shall have the ability to predict traffic patterns and provide suggested routes for optimization.

Notification:

5.7 Notification shall support multimedia attachments (e.g., images, documents).

5.9 Messages shall support different message formats (e.g., text, voice, video).

Tracking Information:

6.3 Tracking information shall maintain an up to date dispatch alerts.

c) Priority 3 (opportunistic):

User:

1.8 User shall be able to save their tracking history into a file.

1.10 User shall be able to receive email/text notification of tracking updates.

1.18 User shall be able to create a family account.

1.19 User shall be able to become part of a family account.

Admin:

3.3 Admin shall be able to suspend any account if needed.

3.9 Admin shall be able to assign specific access levels and permissions to different user roles (e.g., standard user, moderator).

3.12 Admin shall have the authority to implement security protocols, including password policies and two-factor authentication, for user accounts.

System:

4.4 System shall be able to analyze traffic data.

4.7 System shall allow users to set custom alerts or notifications based on specific traffic conditions or events.

Tracking Information:

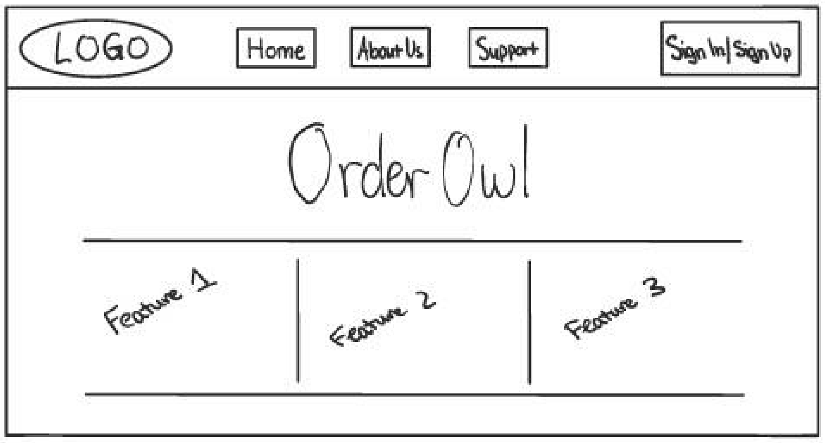
6.5 Tracking information shall be accessible through a mobile application for on-the-go access.

6.6 The system shall integrate with external data sources to enhance tracking accuracy (e.g., weather conditions affecting delivery).

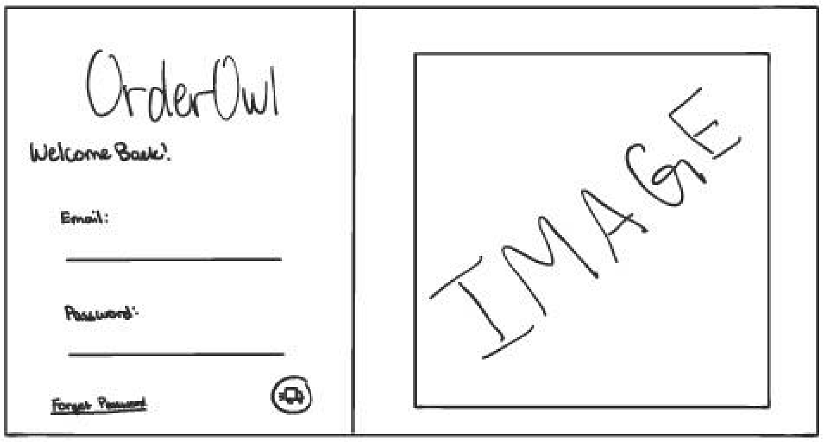
III. UI Mockups and Storyboards (high level only):

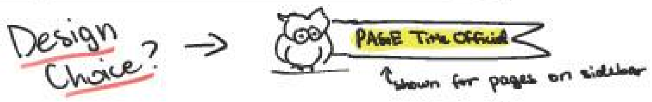
General UI/UX Doodle :)

Home Page

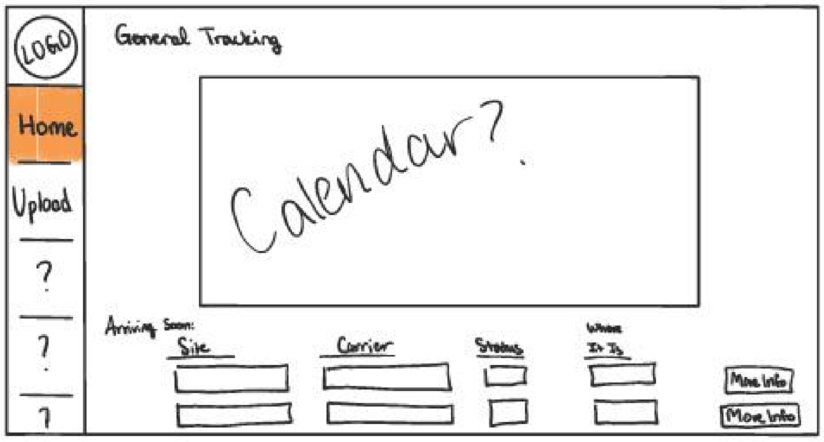


Login



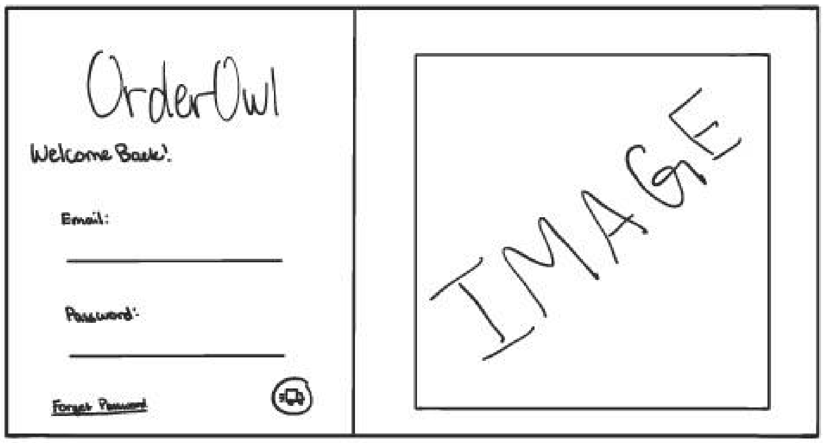


Dashboard

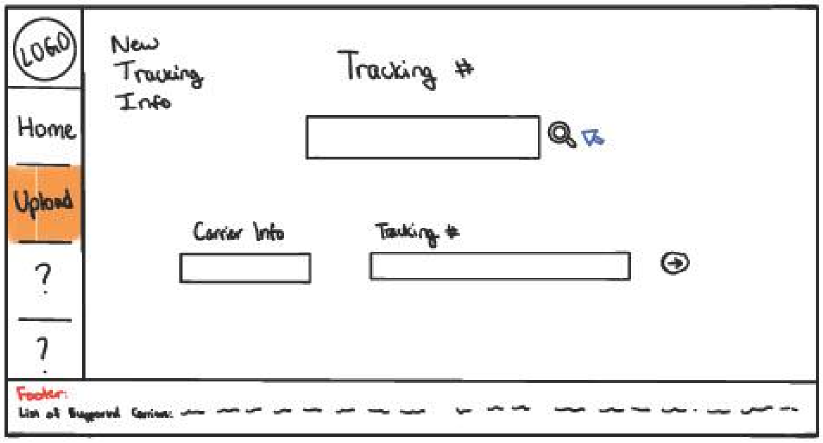


USE CASE 1:

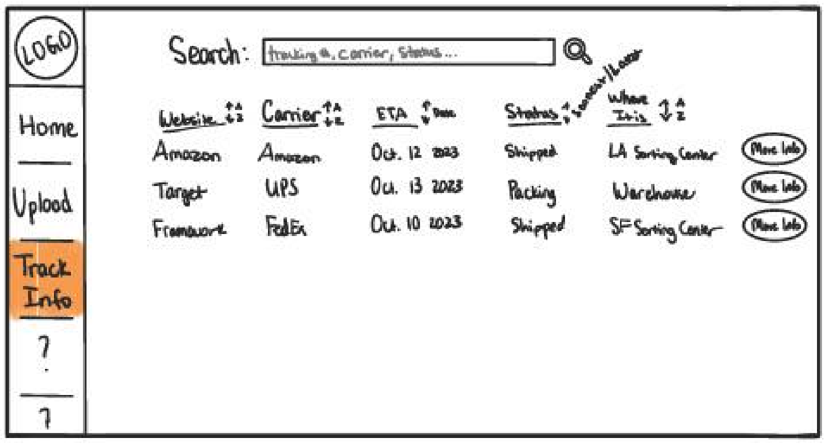
Login



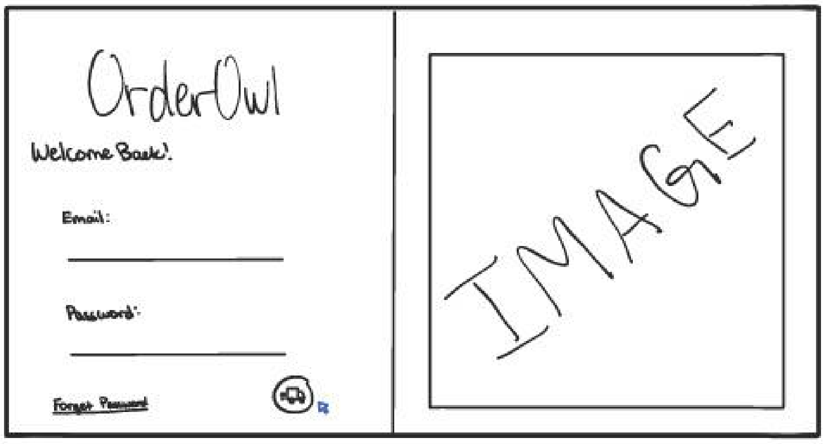
Upload

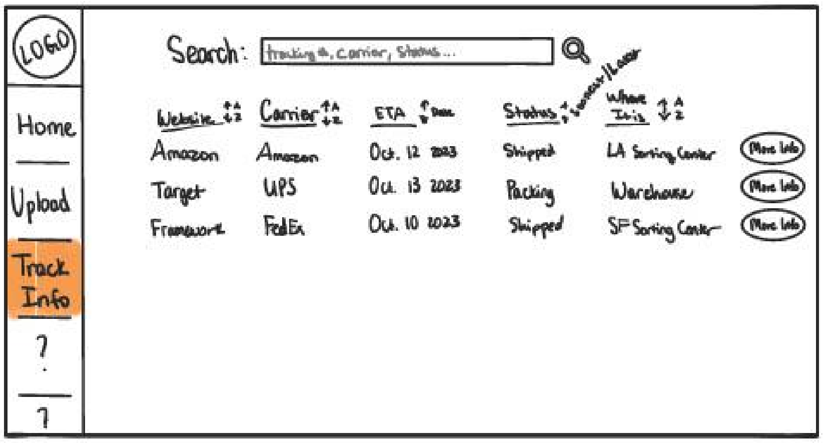


Client View



USE CASE 2:

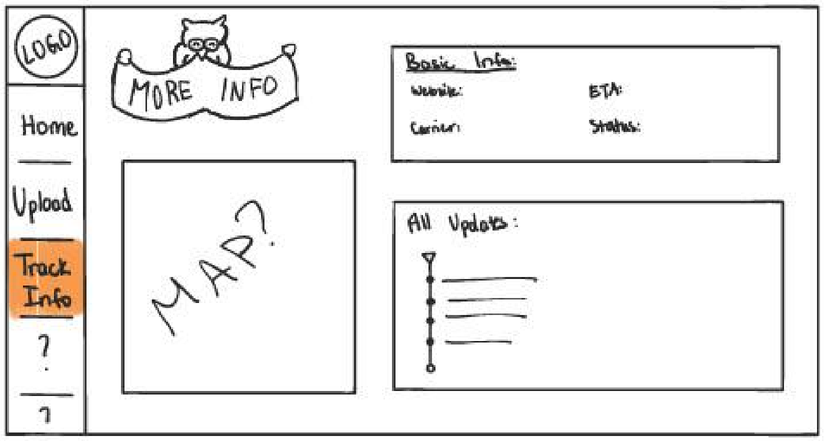




Website	Carrier	ETA	Status	Where It is	
Amazon	Amazon	Oct. 12 2023	Shipped	LA Sorting Center	More Info
Target	UPS	Oct. 13 2023	Packing	Warehouse	More Info
Framework	FedEx	Oct. 10 2023	Shipped	SF Sorting Center	More Info

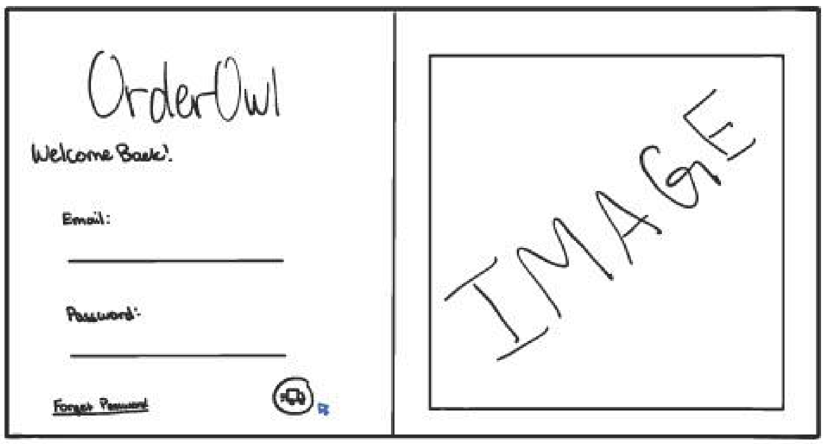
Design Choice?

(Extra page - not shown on sidebar

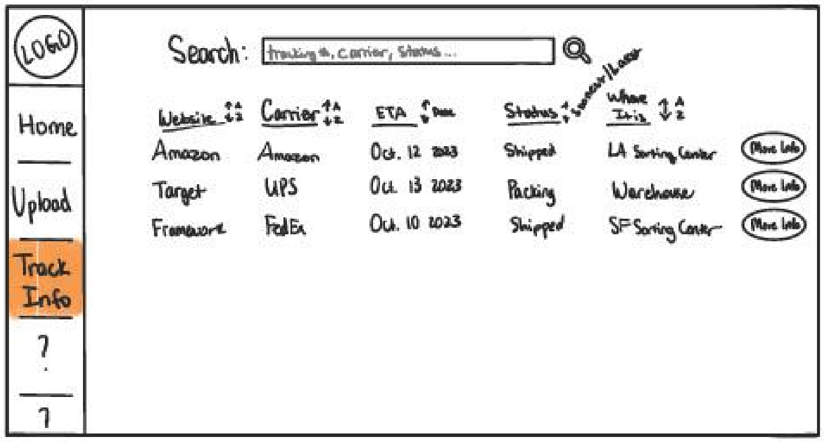


USE CASE 3:

Login



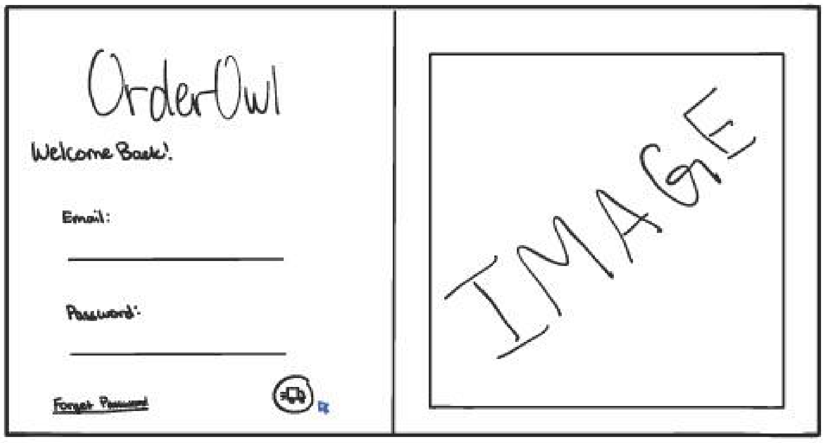
Client View:
All Packaged



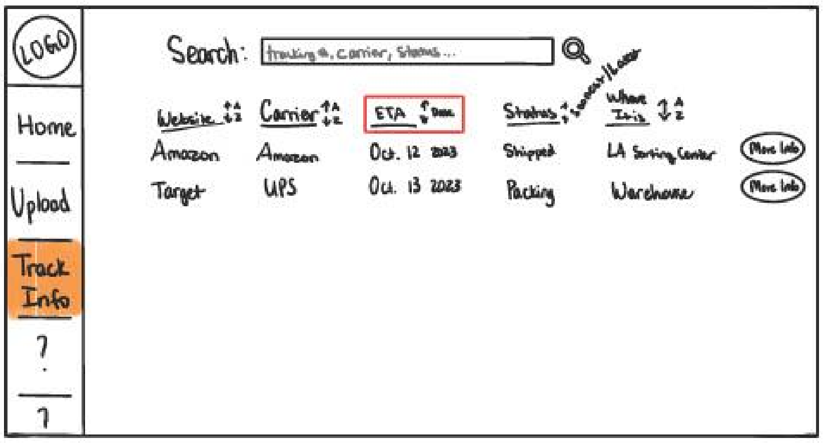
Website	Carrier	ETA	Status	Where It is	
Amazon	Amazon	Oct. 12 2023	Shipped	LA Sorting Center	More Info
Target	UPS	Oct. 13 2023	Packing	Warehouse	More Info
Framework	FedEx	Oct. 10 2023	Shipped	SF Sorting Center	More Info

USE CASE 4:

Login



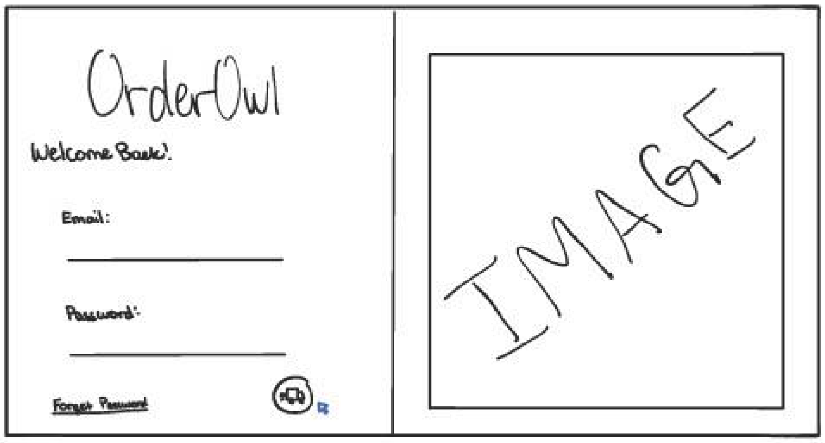
Client View: Specific Sorting



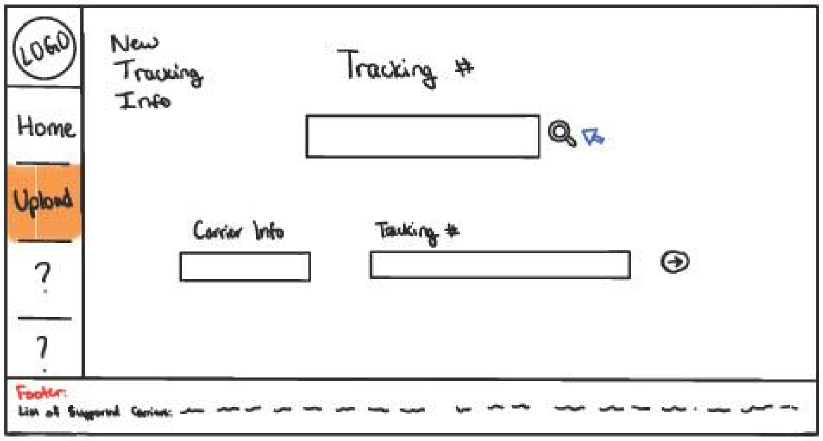
Website	Carrier	ETA	Status	Where Is it	
Amazon	Amazon	Oct. 12 2023	Shipped	LA Sorting Center	More Info
Target	UPS	Oct. 13 2023	Packing	Warehouse	More Info

USE CASE 5:

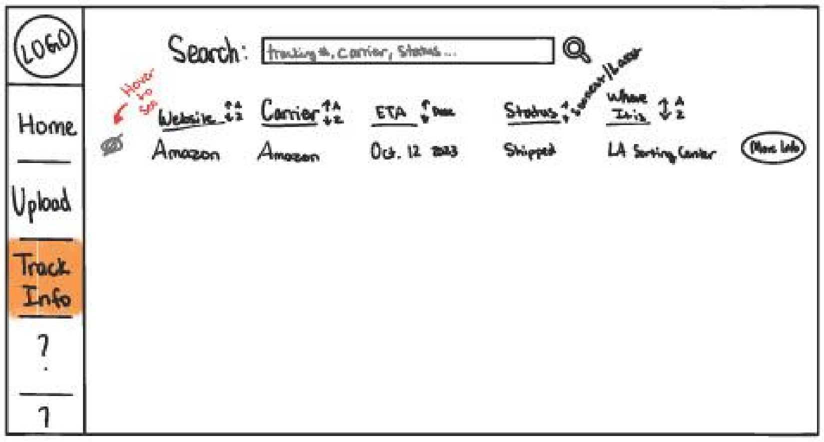
Login



Upload

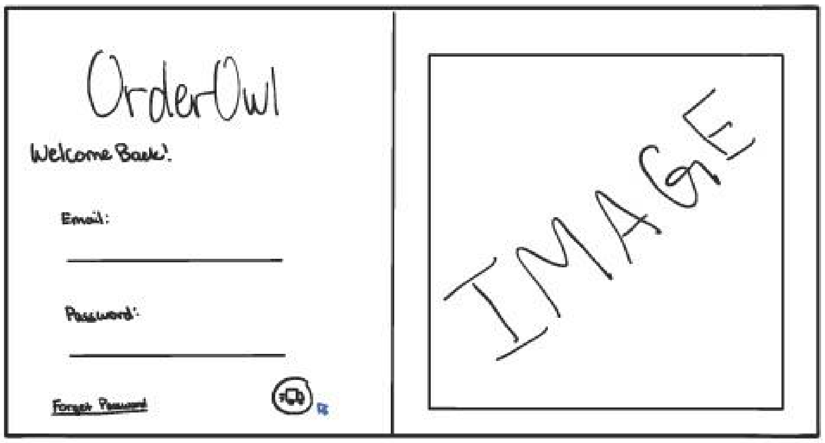


? Client View

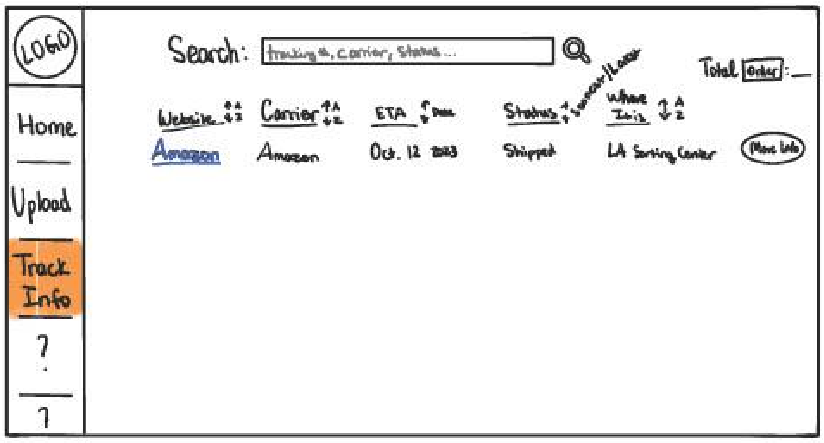


USE CASE 6:

Login

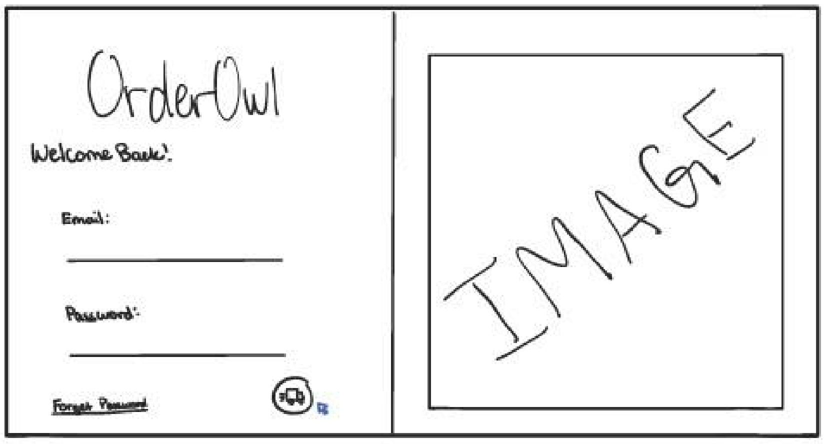


Client View

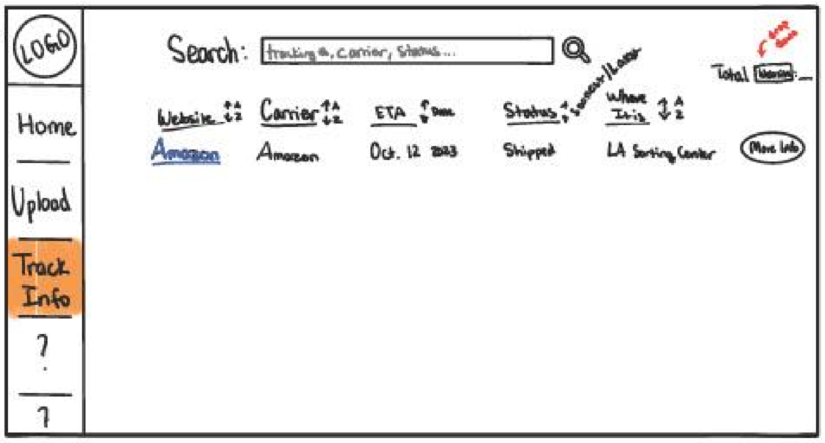


USE CASE 7:

Login

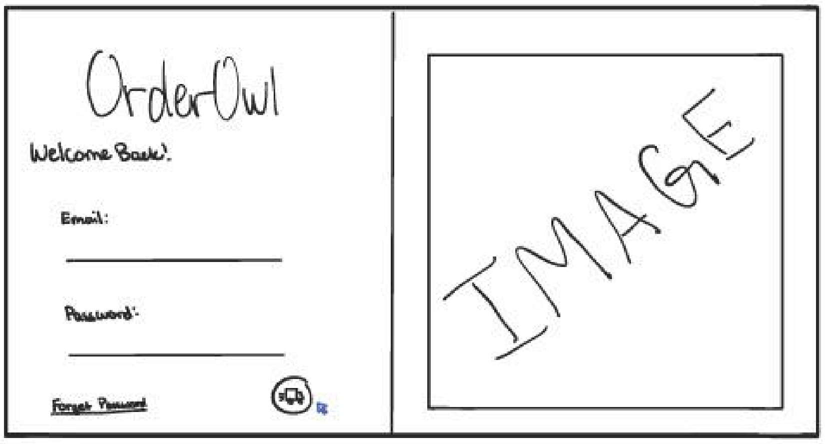


Client View

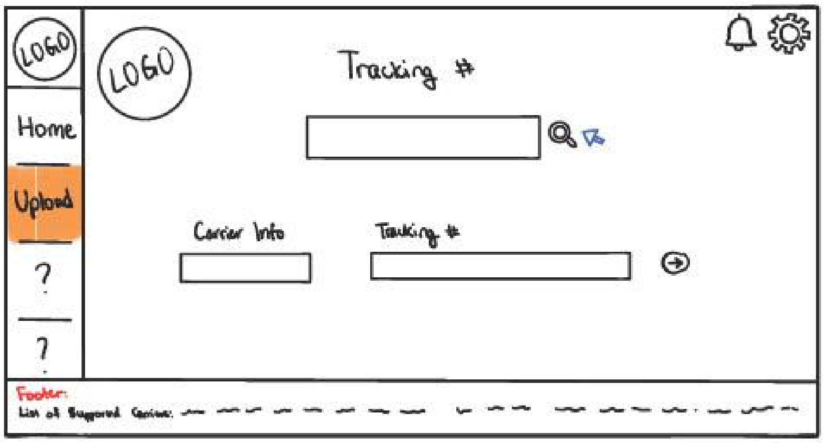


USE CASE 8:

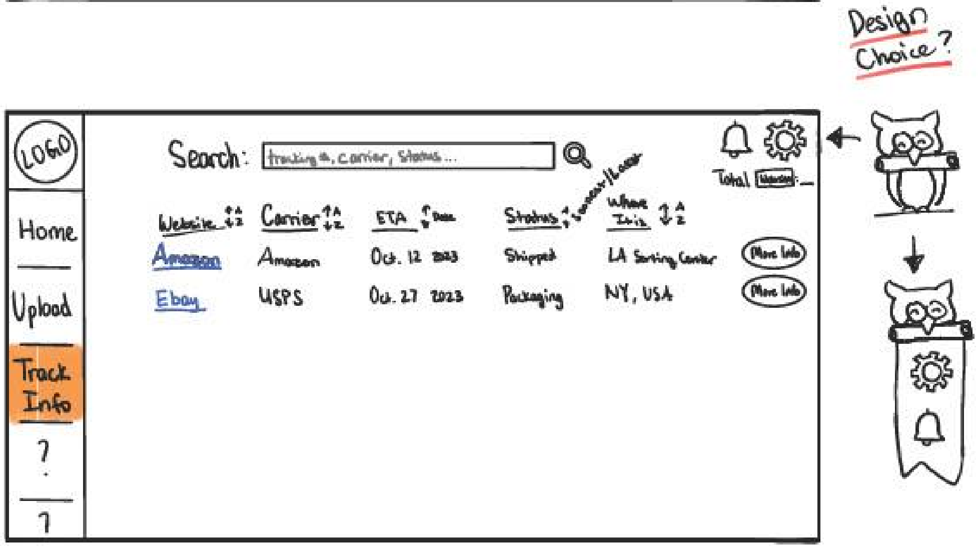
Login



Upload

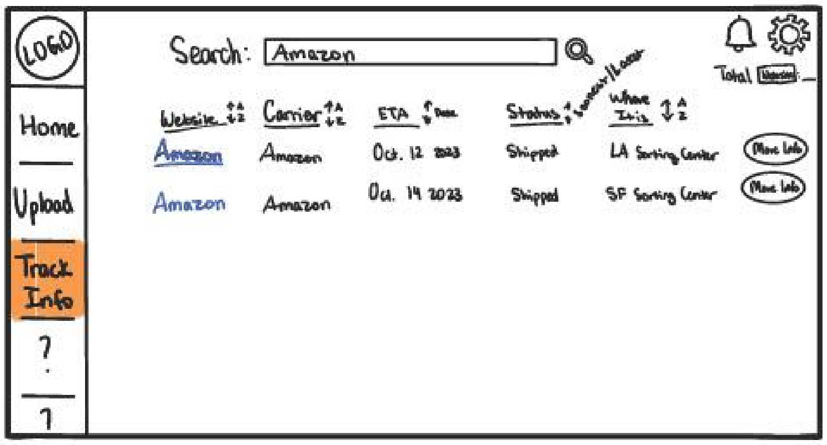


General Client View



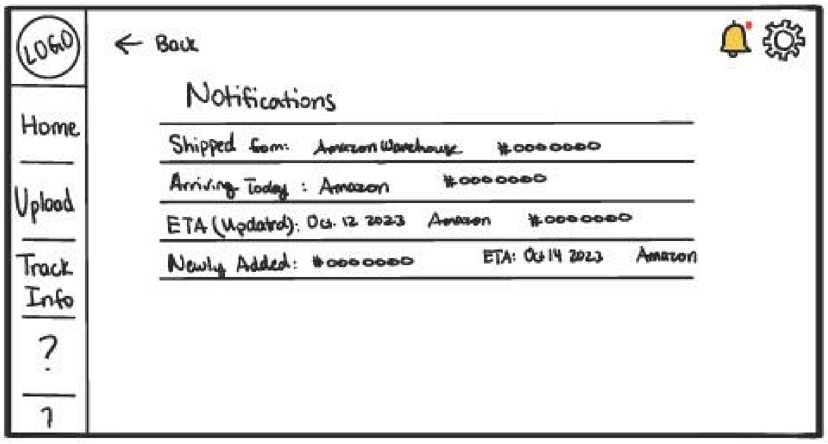
Website	Carrier	ETA	Status	Where Is it	
Amazon	Amazon	Oct. 12 2023	Shipped	LA Sorting Center	More Info
Ebay	USPS	Oct. 27 2023	Packaging	NY, USA	More Info

Ordered Sublist



Website	Carrier	ETA	Status	Where It is
Amazon	Amazon	Oct. 12 2023	Shipped	LA Sorting Center
Amazon	Amazon	Oct. 14 2023	Shipped	SF Sorting Center

Notifications



Notifications

Shipped from: Amazon Warehouse #0000000

Arriving Today : Amazon #0000000

ETA (Updated): Oct. 12 2023 Amazon #0000000

Newly Added: #0000000 ETA: Oct 14 2023 Amazon

IV. High Level Database Architecture and Organization:

1. DB organization:
    a. Main entities:

        Users

        1. user_id (key, numeric)
        2. email (composite, multivalue, alphanumeric)
        3. name (alphabetic)
        4. phone number (multivalue, numeric)
        5. address (composite, multivalue, alphanumeric)

        Item

        6. item_id (key, numeric)
        7. tracking_number (alphanumeric)
        8. status (alphabetic)
        9. name (alphanumeric)
        10. order_id (foreign key, numeric)

        Orders(collection of packages)

        11. order_id (key, numeric)
        12. order_date (composite, multivalue, alphanumeric)
        13. user_id (foreign key, numeric)
        14. hidden (boolean)
        15. retailer (alphanumeric)

        Notifications

        16. notification_id (key, numeric)
        17. receiver_id (numeric)
        18. sender_id (numeric)
        19. content (multivalue, alphanumeric)
        20. timestamp (numeric)

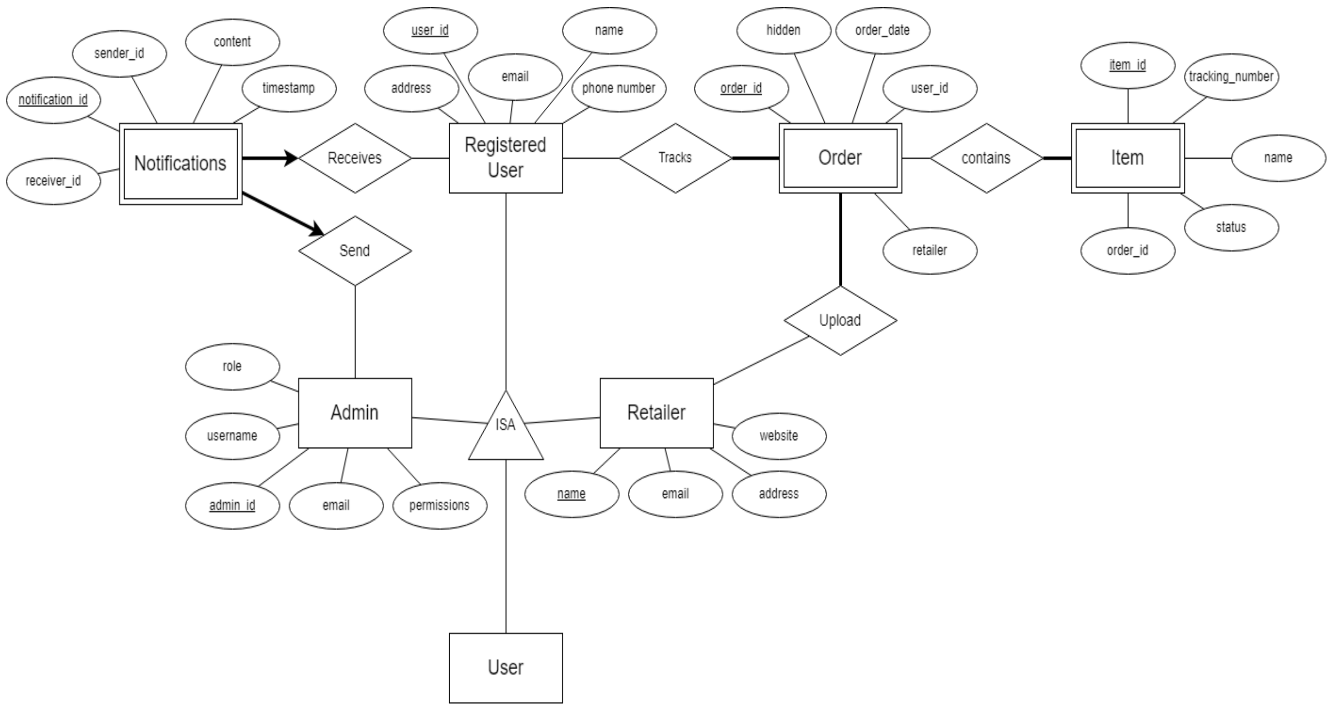
        Retailers/Website

        21. name (key, alphanumeric)
        22. email (composite, multivalue, alphanumeric)
        23. address (composite, multivalue, alphanumeric)
        24. website (composite, alphanumeric)

        System/Admin

        25. admin_id (key, numeric)
        26. username (alphanumeric)
        27. email (composite, multivalue, alphanumeric)
        28. role (alphanumeric)
        29. permissions (alphanumeric)
        30. password (alphanumeric)

b. Diagram



2. Media Storage
    a. We will keep images, video, and audio in the file system.
    b. We will need GPS data
    c. Will use relative link: application/backend/src/main/MediaStorage
3. Search/Filter architecture and implementation
    a. We are planning to use a combination of Spring Boot and query commands through our Java application.
    b. We will organize a search based on the best match based on information given in the input.
    c. DB Terms to be searched: tracking number, address, and Retailer name.
    d. By using Spring Boot using the application.properties file to connect to OrderOwl's database, then sending SQL queries, and returning the results of those queries. OrderOwl's search algorithm shall be able to traverse through the database by meeting certain criterias. SQL queries shall be met with these standards:
    "SELECT t FROM TrackingEntity t WHERE t.trackingNumber LIKE %:searchText% OR t.retailer LIKE %:searchText% OR t.address LIKE %:searchText%"
    e. As of this milestone, the data is organized by id

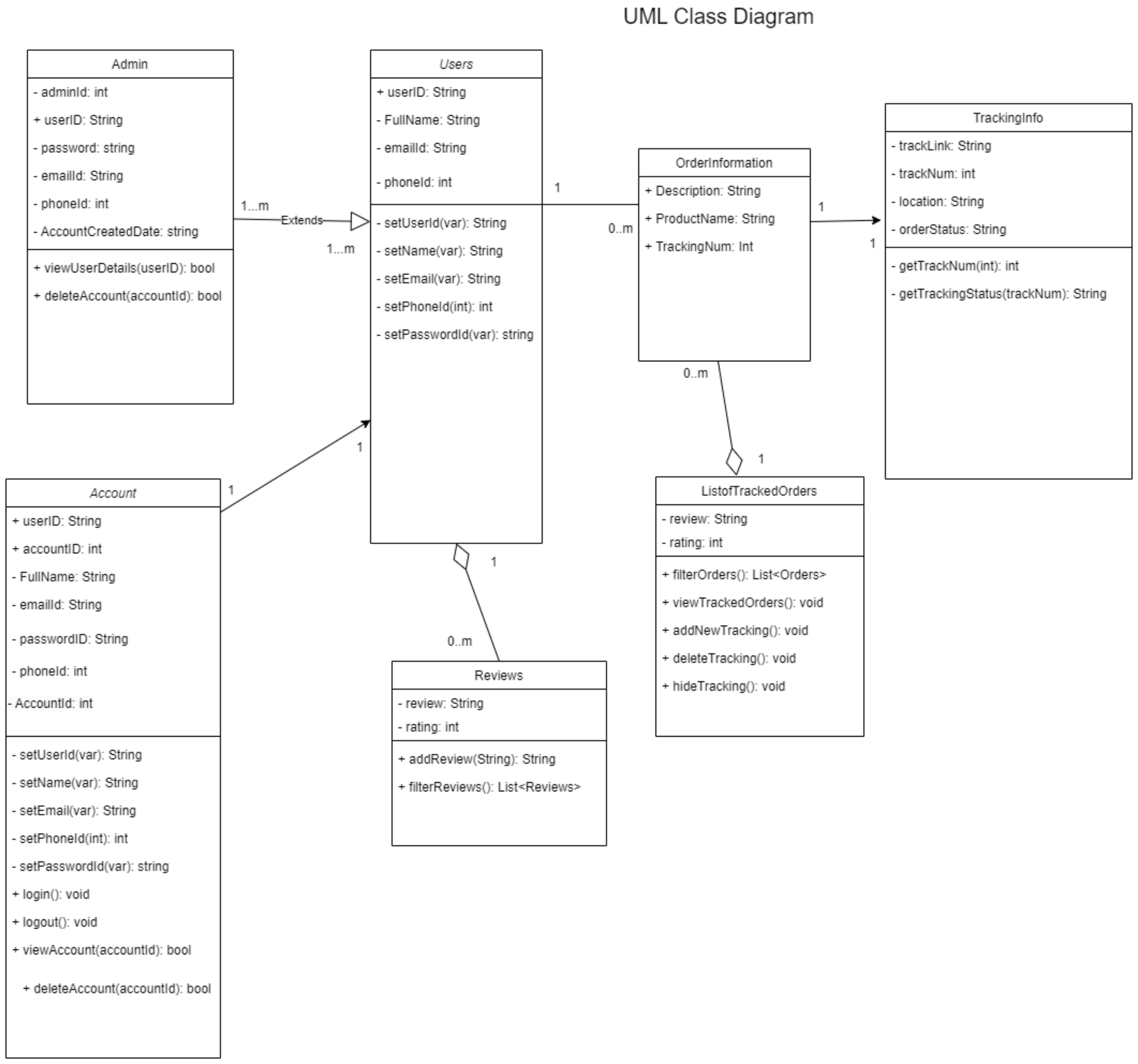
V. High Level APIs and Main Algorithms:

1. High Level APIs
   a. Shipping Carrier
      i. This API will allow OrderOwl to retrieve real-time tracking information and delivery status from shipping carriers similar to FedEx, UPS and USPS. It will enable OrderOwl to integrate with major shipping companies for accurate shipment tracking. Through the Shipping Carrier API, OrderOwl can ensure that users can monitor the progress of their orders, receive updates on shipment status, and access accurate estimated delivery times. This integration will enhance the overall user experience.
   b. GPS/Mapping
      i. This API will enable OrderOwl to offer mapping and location services, ensuring a positive user experience by providing precise location data and mapping capabilities. It allows OrderOwl to display real-time tracking information to users, ensuring that they can easily track and visualize the locations of their orders.
   c. Notification and email
      i. The Notification and Email API enables OrderOwl to keep users informed about their orders in real-time through various communication channels. This component is important to OrderOwl's functionality. By enabling notifications via email or SMS, OrderOwl can send order status updates, tracking information, delivery confirmations, and important alerts. This ensures that users remain well-informed even when they are not actively using the website, offering a convenient and complete way of staying up to date with their online orders.
   d. Thumbnail Retrieval
      i. This API will enable OrderOwl to retrieve a picture for each order it is tracking. These thumbnails will be displayed alongside the shipping status when viewing the shipment. The API will establish connections with various online retailers to gather and show thumbnail images of products included in users' orders. By using the Thumbnail Retrieval API, OrderOwl enhances the user experience by offering visual representations of ordered items, enabling users to quickly identify and confirm the contents of their packages.
   e. Database
      i. This API serves as a bridge between OrderOwl and its database management system, allowing the application to securely interact with the database by encapsulating SQL commands within a data object. Some of the key functionalities of this API include data retrieval, storage,

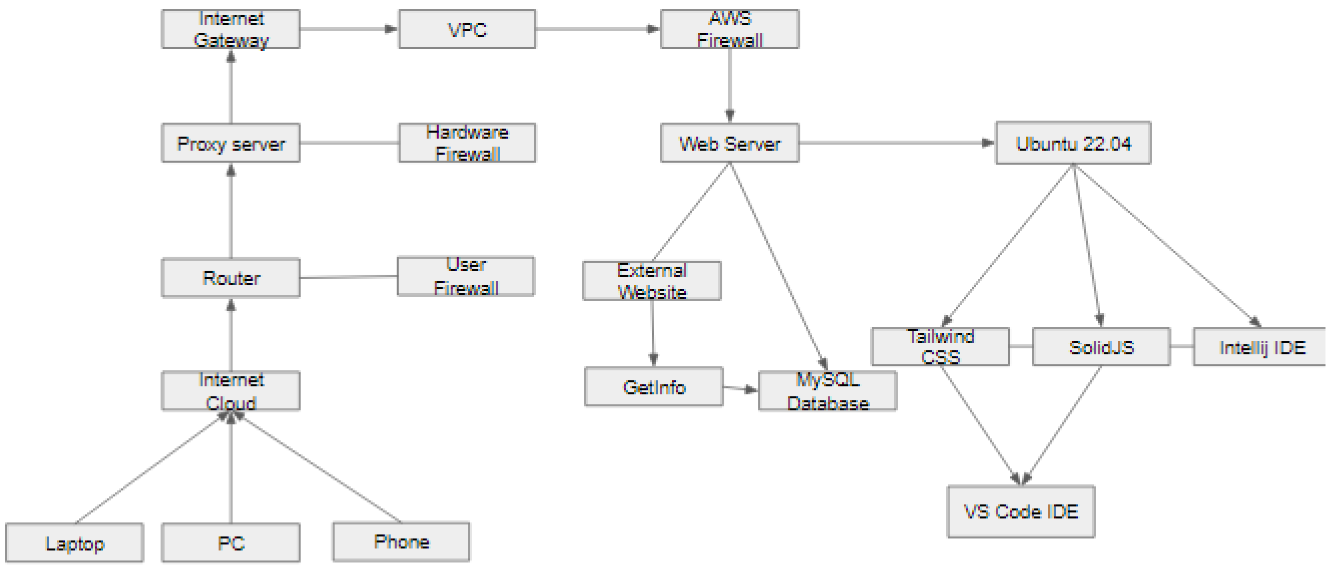
modification, and search operations. By utilizing this API, users will never need to learn SQL to interact with the database.

2. Non-trivial Algorithms
    a. Delivery Date
        i. This algorithm helps users determine which packages are of high priority by utilizing estimated delivery dates and user preferences. The algorithm takes the delivery date into account and calculates a priority score, which increases as the item's estimated delivery date approaches the current date. There is a direct relationship between the score and the proximity of the delivery date. When a user has a preference, such as marking a particular package as high-priority, the score is adjusted accordingly. Once the algorithm completes the priority score calculation, it sorts the orders in descending order based on the score, placing those with the highest priority score at the top of the user's tracking list. With this algorithm, users can easily focus on packages that require immediate action or monitoring.
    b. Recommendations
        i. This algorithm will help users with recommended items that might be on their list and if there is no tracking list available then it will redirect to an add tracking number page.
3. Tech Stack Change
    a. We have added Spring Boot to our techstack to handle the HTTP responses and to work with the database.

VI. High Level UML Diagrams:



VII. High Level Application Network and Deployment Diagrams:



VIII. Identify Actual Risks for your Project at this Time:

- Skill risks:
    1. The skill risk we are having in this project is that we are using SolidJS, which we have never used before, so learning a new tool would definitely cause quite an amount of time to do; thus a risk.
       -We can resolve this problem by giving ourselves enough time to study. We could also have people helping each other out so it could speed up the process.

    2. We are required to have database and computer network knowledge and none of us have taken computer networks and only two members are taking databases right now.
       - The professor will go over these two topics in class so all of us can learn from it and improve. We can also schedule office hour appointments to ask the processor to clarify any necessary questions.

    3. In this project we will need to use tools like AWS, and Cloudflare that we have not used before and the unfamiliarity of them would slow down our progress.
       - For this risk, the way to solve it is to use it and learn about it. The more we interact with it the more we will know.

- Schedule risks:
    1. When we split our team into two smaller teams, to work on our prototype, it became difficult to schedule meetings and assign deadlines within these sub-teams.
       - The entire team would discuss a time that everyone agreed on. We have everyone in the team to fill out an availability sheet to help us on the schedule issue. We could also have two different groups make up their own schedule just between the team members.

    2. When members cannot meet deadlines.
       - Sometimes due to unexpected issues, members will have to push back deadlines. When it happens, we can all communicate to make this work. For example when members can’t meet the deadline they would let the team know what happened and some members in the team that finish early on their part could help out.

- Technical risks:
    1. Developing a way to connect all those tracking websites together and taking all the tracking information into our website will be difficult.
       - We understand this risk could be the biggest problem for our project. How we are going to solve this problem is to do the necessary research to learn what is the way we can put this into practice. We would also ask as many questions as we can so we can succeed.

2. Finding a way to connect data from the website into the MYSQL database.
   - We would ask as much as we can, attend the office and do research online to find out how it works. Also we could use the class notes from CSC 317 and CSC 413 to help us on the process.

3. Since our project will have to collect many types of data, we will need to build a database that fits our requirements and that will be difficult.
   - We have members taking a database class right now and we could use them to help with our project. We would also do research and ask questions when facing obstacles.

4. Create an API that fits for our use.
   - In order to tackle this risk we will need to do research and again ask the professor for help.

- Teamwork risks:
  1. We were having issues when working on the prototype. In this milestone when split the team up into two groups, front end and back end but when it comes to split up the code work it is a little tricky.
     - How we are going to solve this risk is we would discuss it in the team meeting and then the front end team would have their front end team meeting and so is the back end team so we can keep our progress going.

  2. When we have multiple people work on the same part.
     - We could find out a way to have each member have a separate code file to be in charge of and work on it by themselves.

- Legal/content risks:
  1. In our prototype we will track products from sites and those sites like amazon already have their version of cargo tracking and it could create a problem that we are taking the user from their websites into ours because it could lower their user daily activity and it is a very important point to the internet company.
     - Instead of replacing the tracking system of sites like Amazon, We can integrate their tracking interface within our platform. By clicking on the link the user will still link to the original tracking system.

  2. In our product we would need to get access to the user account for those websites so we could know if there is an order. There might be a legal problem in it.
     - We could ask the user to fill out an agreement that allows us to do that. We could also learn what the competitors do and we could do that as well.

IX. Project Management:

For this milestone, we split up the overall work like Milestone One in the sense that the first half, regarding the documentation, is checkpoint #1, and the second half, regarding the actual application/vertical-prototype, is checkpoint #2. Within our checkpoint #1, we distributed the work amongst members so each member is either taking charge and/or working with a partner on sections for the documentation. We decided to focus on the documentation first as a lot of the documentation is focused on the design and planning for the layout of our vertical prototype; thus our first deadline was to complete these sections to the best of our abilities. Once we completed the first drafts of these sections, we assigned everyone another section to peer review, so every section has had at least a second set of eyes to check if it followed requirements. During this peer review period, we also started working on the actual vertical prototype itself; checkpoint #2. For this checkpoint, we split the team into two sub units: the front end team and the back end team. Each team had their own set of tasks relating to their team's focus and both teams regularly discussed whenever the front-end and back-end met within the code. The back-end team had a bit of a hard time developing this section of the vertical prototype because this side of application development is not something that the back-end team members had much experience with. The back-end team had to work closely together by sharing resources and progress status and creating zoom meeting work spaces; similarly as the front-end team who worked together to bring our mockups/storyboard designs to reality. Some of the tools that we used for our project management included the use of Trello, which is where we would organize our incomplete and completed tasks for both checkpoints, and Discord, which is where we created threads for the sub units (back-end team and front-end team) and other topics that needed a separate space for discussion aside from the general chat. Overall, our game plan was to break up this milestone into much smaller parts, attack them in priority order, and constantly keep each other updated on the progress of the entire Milestone 2 progress.

X. Detailed List of Contributions:

Team Lead:

Team Members	Contribution	Score
Belu Velazco	● In charge of sections 6, 9, and 10 ● Edited Milestone 1 V 2 and M2V2 ● Created Trello board and Milestone 2 document (set up Milestone 2 document) ● Frequently gave updates on progress on assigned tasks as well as checked in with members on their progress ● Assigned deadlines to tasks and sections within the milestone ● Checked on progress of vertical prototype and gave feedback when needed ● Worked with back-end team to help with and work on registration ● Set up the folders and files for back end ● Provided resources to help with prototype development ● Gave feedback to documentation throughout the milestone ● Reminded team about requirements that need to be met within both “checkpoints” of the milestone ● Led team meetings	7

	and wrote up team meeting reports after every meeting ● Always kept updated with the discussion channel and made sure to respond to messages within 1-2 hours ● Hosted workshop meetings towards the end of the milestone for the development of the prototype ● Scheduled 2 back-to-back office hour appointments ● Actively participated and communicated	
David Lien	● In charge of sections 5 and part of 4 ● Worked with back-end team regarding search algorithm and finished registration ● Provided resources to help with the development of the vertical prototype ● Actively participated and communicated ● Attended all in-class and outside-of-class meetings ● Attended and hosted workshop meetings toward the end of the milestone ● Actively kept up to date with back end thread and general channel ● Helped other teammates	8

Komaldeep Kaur	● In charge of section 2 ● Worked on part of section 3 regarding the mockup for use case 1 ● Worked with front-end team and worked on upload page ● Participated and communicated ● Attended all in-class and outside-of-class meetings ● Joined in on a workshop meeting towards the end of the milestone ● Formatted the first half of the documentation	Team Lead's score: 5 Komal's score: 7
Jimmy Pan	● In charge of section 3 specifically on creating the mockups for use cases 2-8 ● Worked with front-end team regarding the homepage, dashboard, upload page. And helped modify and edit others ● Participated and communicated ● Attended all in-class and outside-of-class meetings ● Helped other teammates ● Provided resources to help with prototype development ● Created logo for our application ● Attended office hour	8

	appointments ● Provided resources to help with the development of the vertical prototype ● Constantly kept updated with the front end part of our prototype ● Actively worked with teammates	
Mankit Yeung	● In charge of section 8 and part of section 7 ● Helped come up its hook and added it to the homepage ● Worked with front-end team regarding the homepage and tracking information pages ● Actively participated and communicated ● Attended all in-class and outside-of-class meetings ● Attended office hour appointments ● Actively worked with teammates ● Helped other teammates ● Attended workshop meetings towards end of the milestone ● Actively updated on progress ● Formatted last half of the documentation and did peer reviews ● Edited Milestone 1 V 2	8
Luis Ramirez	● In charge of section 1	8

	● Worked with back-end team regarding login ● Provided resources to help with the development of the vertical prototype ● Actively communicated and participated ● Helped other teammates ● Attended all in-class and outside-of-class meetings ● Constant updates on parts ● Joined in on workshop zoom meetings with team towards end of the milestone ● Actively worked with teammates	
Tin Nguyen	● In charge of part of section 4 and part of section 7 ● Worked with back-end team regarding search algorithm ● Participated in team meetings ● Attended all in-class meetings except for one but informed the team ahead of time ● Attended all outside-of-class meetings ● Joined in one workshop meeting over zoom towards the end of the milestone. ● Edited Milestone 1 V 2	6