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

November 2, 2023

History Table:

Revision	Date
M2V2	11/02/23
M2V1	10/13/23
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 and use our services with OrderOwl. We shall request the user a password, username, email, first name and last name. 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 receive 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 saved within our application for future use. The user shall use their information to log back into OrderOwl and to view their tracked 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 offer 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. We will store this information in the database so our users won’t lose their tracking information.

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, we will store all of the orders in our database and as well as their history of completed tracked orders.

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. After we get access to this information, we will make sure to encrypt it so no one will be able to look at this information.

f. Order Category

The orders shall have a category depending on the item they ordered, such as clothing, electronics, food, or other categories. 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.

g. 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. In addition, the order information should contain the amount they paid for the order, when they bought it, and the quantity of items under an order if applicable.

h. Deleted Orders

All our user's orders will be stored in our database after they are uploaded. After the user decides to delete the order from their account, the tracking information for that order will be gone from our database and account.

i. Updates

Before we roll out updates we will store the information in our database in order to be ready to push it out for the public. We shall inform our users of our rolled out updates to keep them updated on what is new. Sometimes when a user uses an updated application, some stuff might be moved around or there might be new features.

j. Notifications

We will be sending out notification to users whenever they have an upcoming delivery. We will store the notifications in our database to get a basic template of what we are sending out to our users. This could potentially lead to less packages getting stolen and more people being ready to receive their package.

k. 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 or sell the information we receive from our users. We will only use it for the benefit of a better experience for users on OrderOwl.

l. Administration

Admin shall be able to login to their administrative accounts in order to do some testing with features that haven't come out yet. We will store login information for admin usage separate from regular users. This will help us to test out features that are not out yet before sending it out.

m. Return/Refund

We will be storing the refund and return information in our database for the user to see the tracking information as well. 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.

n. 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.

o. Sharing

Users will be be able to share tracked packages amongst friends, family, or others to keep more than one person updated on the tracking of an order. Users will only be able to share the packages that are stored in your account because we will be able to share from our database where it is saved.

p. Hidden Packages

When our users use our software they will have the option of hiding their package. Even though it is hidden, it will still be in our database the entire time it is hidden, it will just look like it was deleted. There will be a tab of hidden orders in order to be able to check on them still.

q. Privacy

We will be taking care of privacy by encrypting in our database the users information and tracking information they’ve saved on our software.

II. Prioritized Functional Requirements:

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.

1.6 User shall be able to recover password when needed.

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

1.9 User shall be able to sort their tracking information.

1.11 User shall be able to write product reviews.

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

1.13 User shall be able to send/receive messages.

1.14 User shall be able to view/delete messages.

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

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

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.

3.6 Admin shall be able to view another account.

3.7 Admin shall be able to create a log.

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

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.

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:

5.1 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.

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

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

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

Tracking Information:

6.1 Tracking information shall be able to be stored.

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

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:

1.8 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.

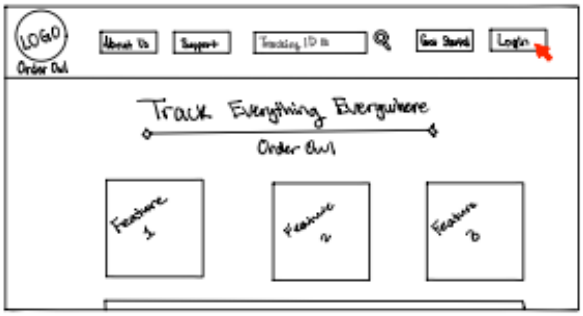
Tracking Information:

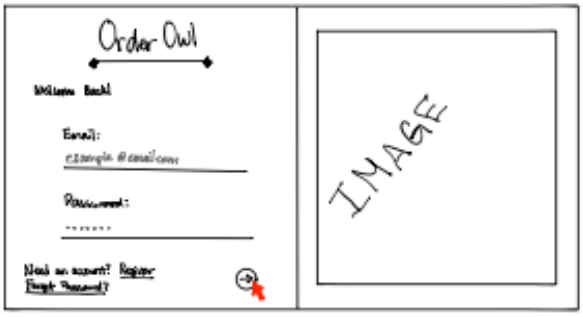
6.5 Tracking information shall be accessible through a mobile application for on-the-go access.

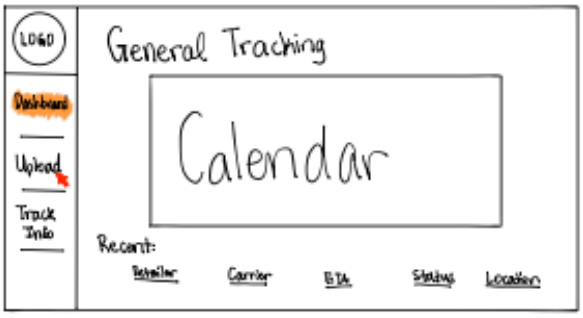
6.6 The system shall integrate with external data sources to enhance tracking accuracy (e.g., weather conditions affecting delivery).

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

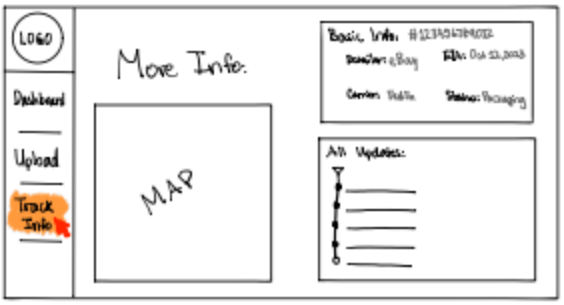
Alex is in college and has gotten into the business of being a purchasing agent. He buys stuff online ahead of time and needs to be able to keep track of all his ordering information.



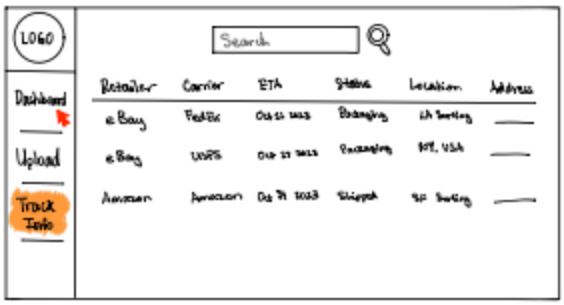




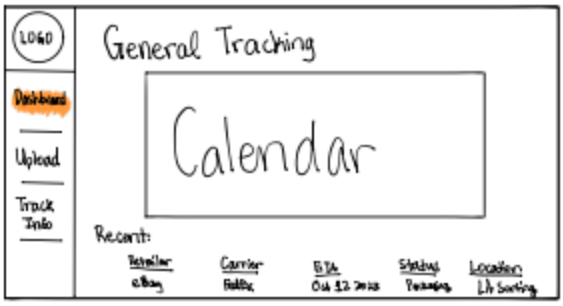
Here Alex can upload all new tracking information.



All of his tracking inofrmation can be found in the tracking info part.

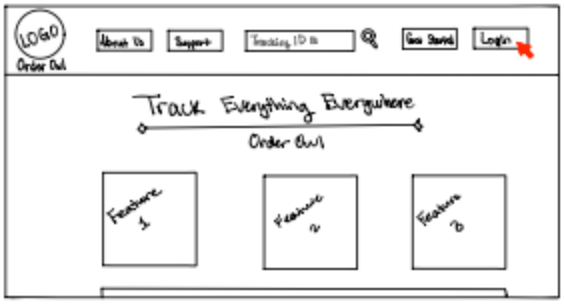


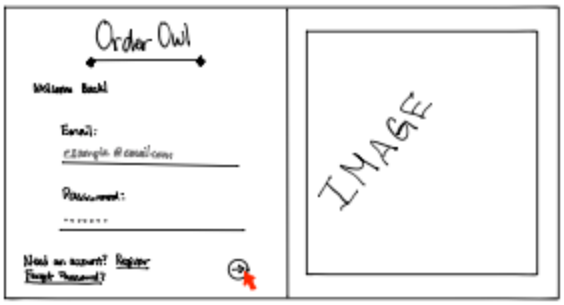
Here on the dashbaord page, he can also see his most recent additions.

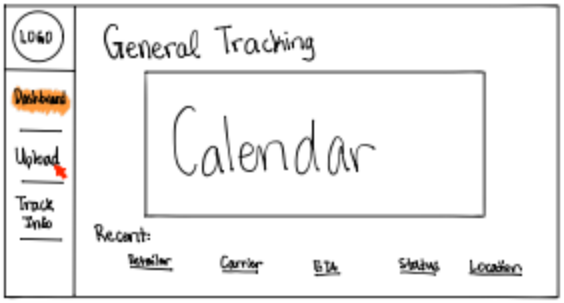


USE CASE 2:

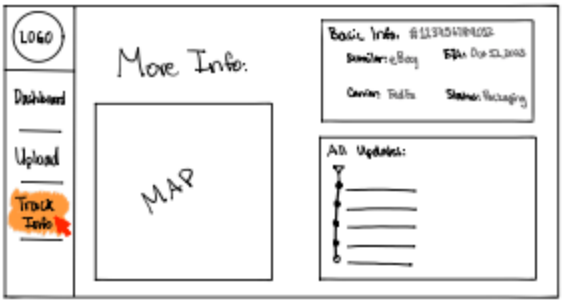
As a store manager, James needs to keep track of all the orders coming in for his store.



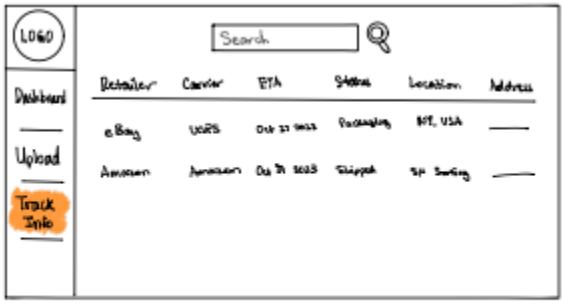




Here, he can upload all new purchases of new products.

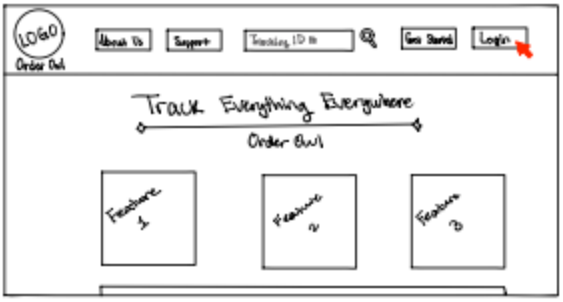


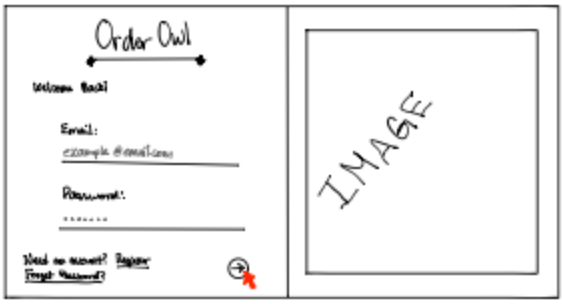
He can also go into each order and look at more information as well as delete ones that he doesn't want to see.



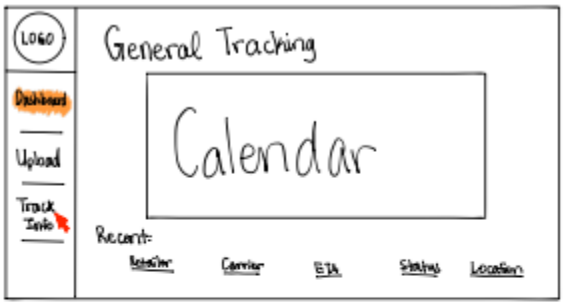
USE CASE 3:

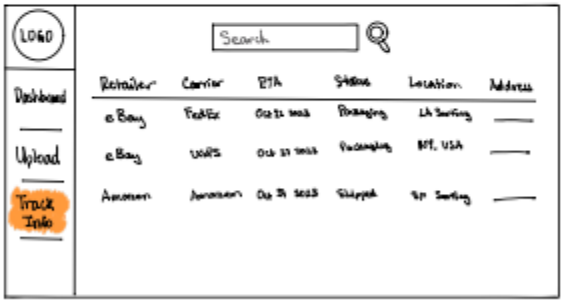
Bruce is a 70 year old man with lots of family far away from him. They like to buy him gifts, but they are often from different sites, making it difficult for him to navigate and know when they are coming.





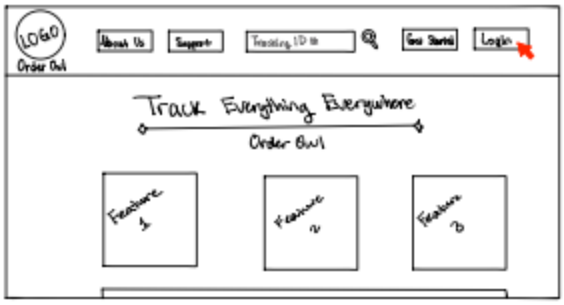
Once he uploads th tracking information, he is able to see all basic tracking info and when it is coming on the track info page.

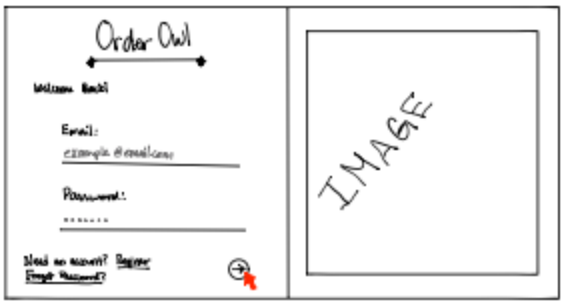




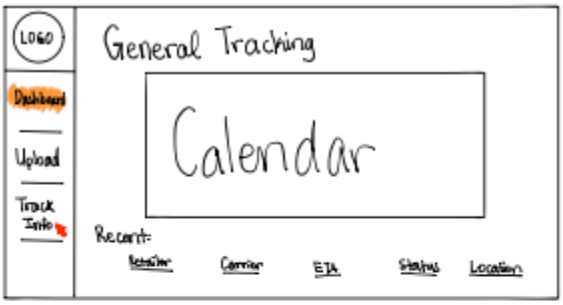
USE CASE 4:

Anna is a business woman who does not have a lot of time. She orders a lot of things but has found it to be too time consuming to keep track of all her order info from differnt websites.





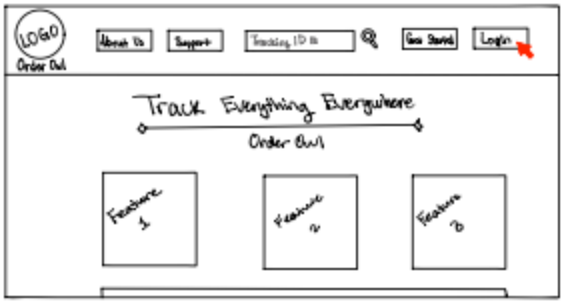
She can organize it by ETA (estimated time of arrival).



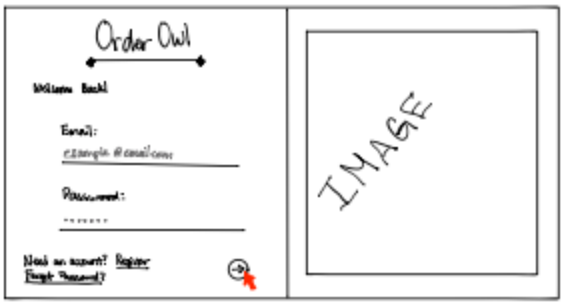
By clicking on it multiple times, it can organize it from earliest to latest arrival times and latest to earliest arrival times.

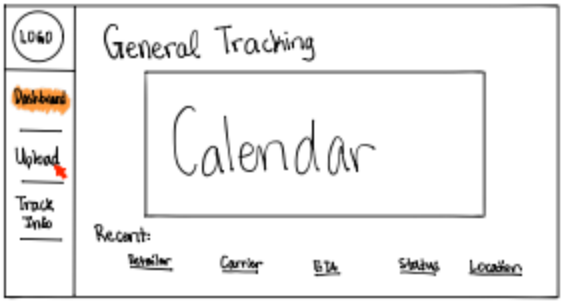
USE CASE 5:

This is Kyle. He works at an international shipping company as a transshipment warehouse manager. He finds the company tracking website to be insufficient. He comes across OrderOwl and finds that our website managers the tracked packages better.

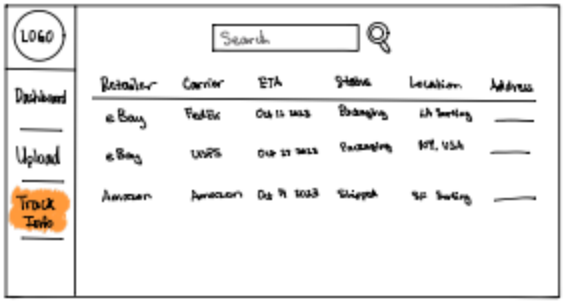


He logs in to his account to upload a new number to be tracked.



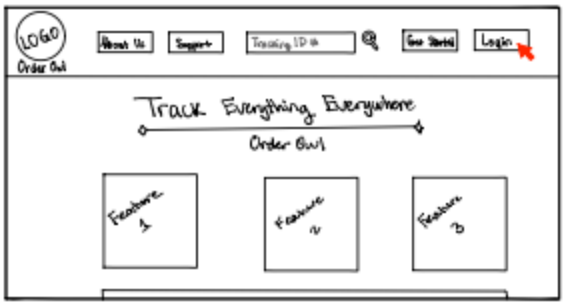


After adding the new tracking number, he can see the new entry among his other entries that he added.

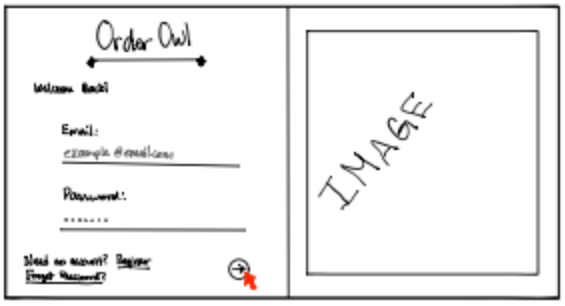


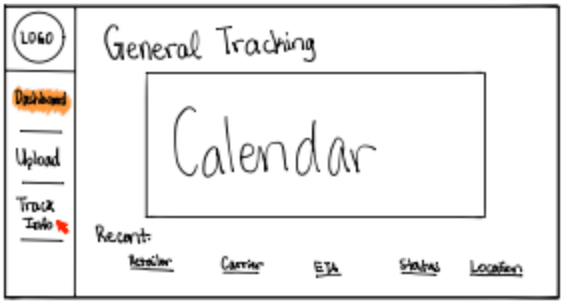
USE CASE 6:

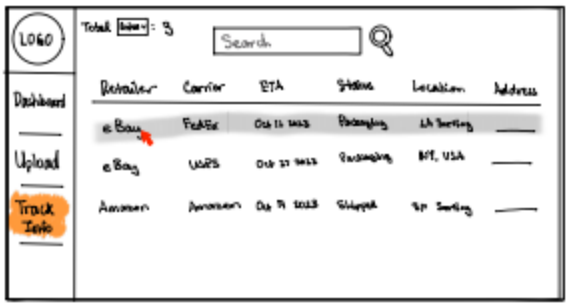
This is Kaly. She is a 17 year old girl who loves to shop. She used to go out shopping with her friend all the time, but once the pandemic started, she resorted to do all her shopping online. With all the different packages she order, she needs a place to keep track of all of them for her. This is where OrderOwl comes in. She is able to keep track of all her orders here, and knows when each of them are going to arrive.



After she logs in, she can view all her tracked packages and click into them for more details about each package.



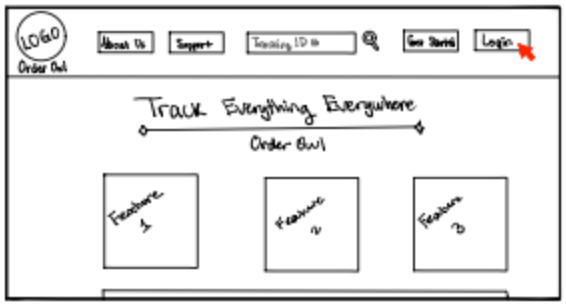



LOGO
Dashboard
Upload
Track Info
Search
Retailer
Carrier
ETA
Status
Location
Address

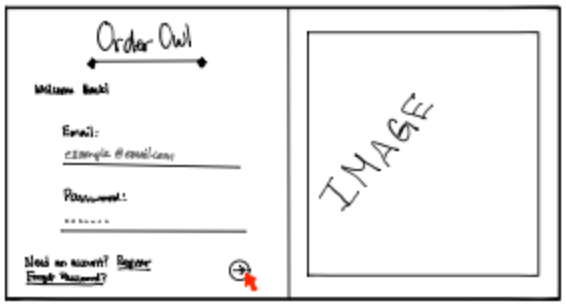

LOGO
Dashboard
Upload
Track Info
Move Info:
MAP
Basic Info.
All Updates:

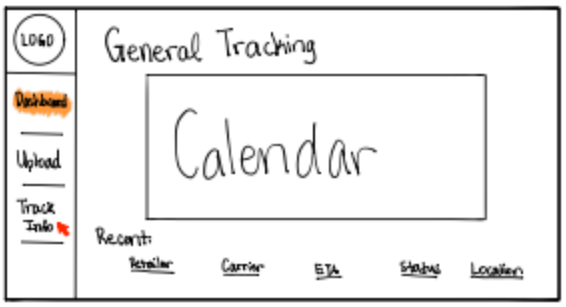
USE CASE 7:

Molly is a 62 year old stay at home wife. She loves to cook and orders her ingredients online. She would source her ingredients from different websites. Being non-tech savvy, she has a hard time tracking her ingredients across multiple websites, even with the help of her children. OrderOwl keeps track of all her orders which makes it much easier for her to keep track of her purchases by herself.

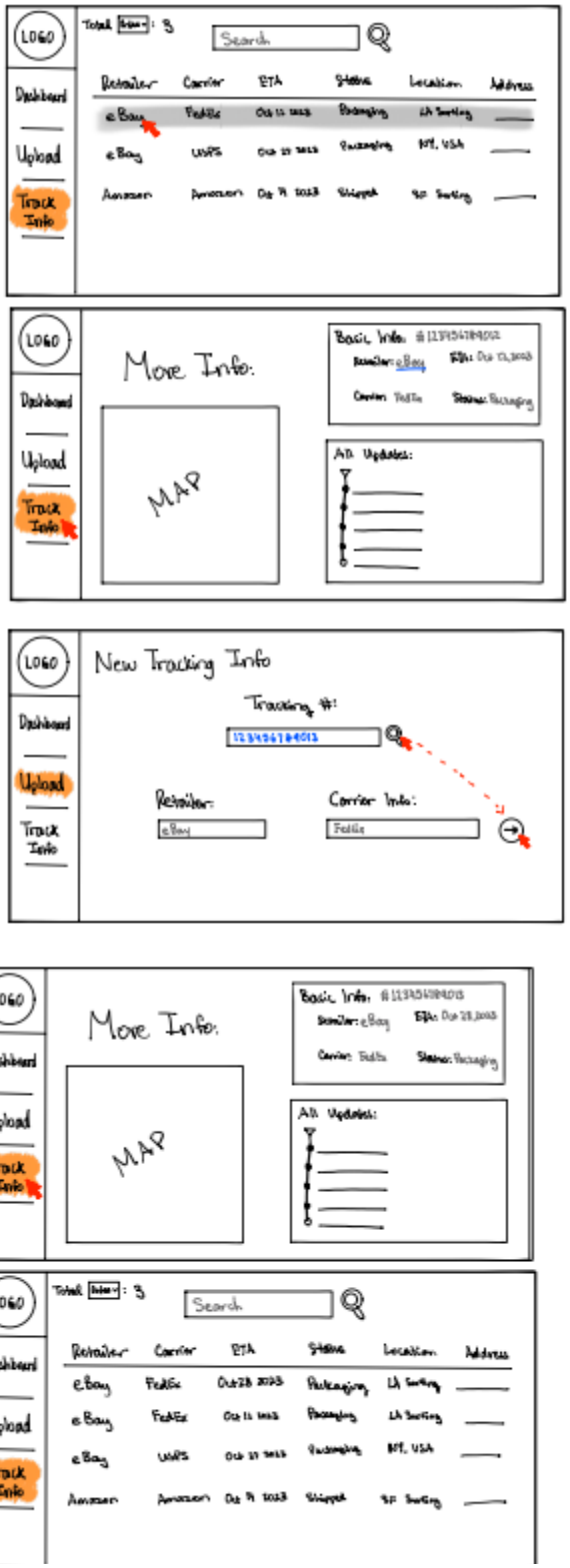


She starts by login into her account,
then checks on the status of her



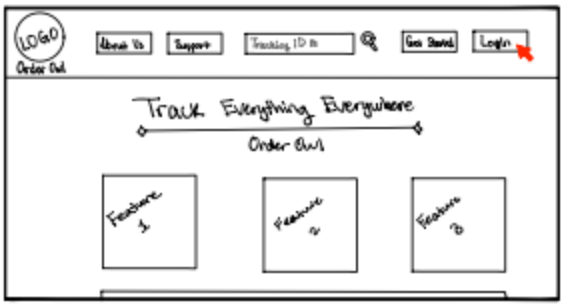


She then uploads the most recent tracking number she got from the order she just made. And now it is being tracked by our page.

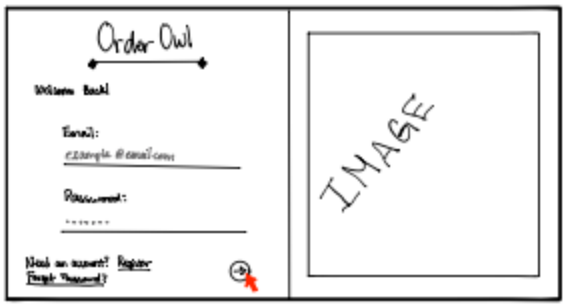


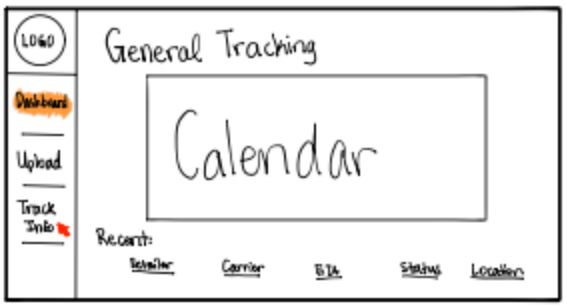
USE CASE 8:

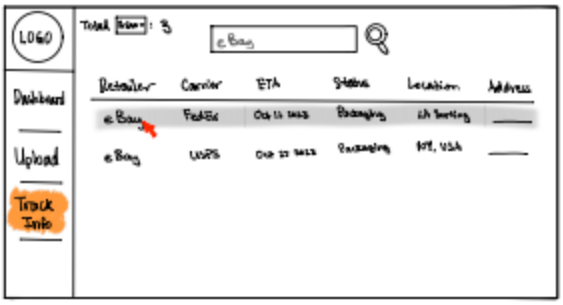
This is John. He and his roommates are indoor people, so he purchases most of the supplies they need from the internet. Because each person has their own needs, he has to go to different websites to order everything. Even though he is good at using the internet, he still finds it cumbersome to go to each website to track their order. This is where OrderOwl helps keep track of all his packages right from our website.



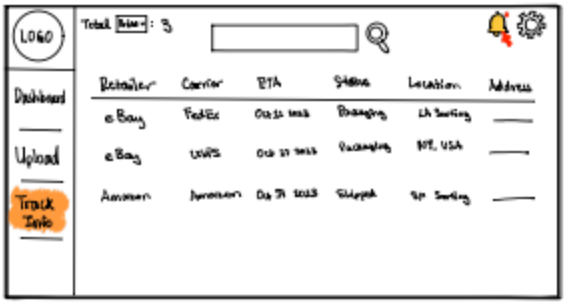
He can do things like look up orders from specific retailers among other things.

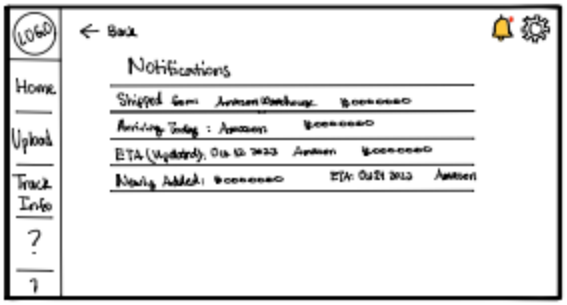






He also gets notifications on updates to each of his packages.




LOGO
Home
Upload
Track Info
?
Back
Notifications

IV. High Level Database Architecture and Organization:

1. DB organization:
    a. Main entities:

Users (Strong)

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

Tracking Information (Weak)

1. tracking_id (key, numeric)
2. date (composite, multivalue, alphanumeric)
3. tracking_number (alphanumeric)
4. status (alphabetic)
5. location (alphanumeric)
6. user_id (foreign key, numeric)
7. hidden (boolean)
8. retailer (alphanumeric)

Notifications (weak)

1. notification_id (key, numeric)
2. user_id (foreign key, numeric)
3. tracking_id (foreign key, numeric)
4. content (multivalue, alphanumeric)
5. timestamp (numeric)

Retailers/Supplier (Strong)

1. name (key, alphanumeric)
2. email (composite, multivalue, alphanumeric)
3. address (composite, multivalue, alphanumeric)
4. website (composite, alphanumeric)
5. role (alphabetic)

Admin (Strong)

1. admin_id (key, numeric)
2. username (alphanumeric)
3. email (composite, multivalue, alphanumeric)
4. role (alphanumeric)
5. password (alphanumeric)

Tracking History (Strong)

1. history_id (key, numeric)

2. user_id (fk, numeric)
3. tracking_id (fk, numeric)
4. date (numeric)

ii. Calendar (Weak)
1. user_id(fk, numeric)
2. history_id (fk, numeric)
3. calender_id

iii. Q and A questions (Weak)
1. question_id (key, numeric)
2. user_id (fk, numeric)
3. admin_id (fk, numeric)
4. date (composite, alphanumeric)
5. content (composite, multivalue, alphanumeric)

iv. Q and A answers (Weak)
1. answer_id (key, numeric)
2. admin_id (fk, numeric)
3. user_id (fk, numeric)
4. date (composite, alphanumeric)
5. content (composite, multivalue, alphanumeric)

v. Admin (Strong)
1. admin_id (key, numeric)
2. email (composite, multivalue, alphanumeric)
3. username (alphanumeric)
4. role (alphabetic)

b. Relationships

i. Admin
1. Admin manages one or more Registered User
2. Admin manages one or more Retailer
3. Admin can answer zero or more questions

ii. Registered User
1. Registered User is managed by zero or more Admin
2. Each Registered User can track zero or more Tracking Information
3. Each Registered User can upload zero or more tracking information to track
4. Each Registered User can track zero or more tracking information
5. Each Registered user may have zero or more items in Tracking History
6. Each Registered User can receive zero or more Notifications
7. Each Registered User can ask zero or more questions

8. Each Registered User can view one and only one calendar

iii. Retailer
   1. Retailer is managed by zero or more Admin
   2. Each Retailer can upload zero or more Tracking Information

iv. Tracking Info
   1. A tracking information can be tracked by one or more Registered User
   2. A tracking information can be tracked by one or more Unregistered User
   3. Each tracking information must be uploaded by at least one Retailer
   4. Each tracking information must be uploaded by at least one Registered User
   5. Each tracking information must be uploaded by at least one Unregistered User
   6. Each tracking information can be tracked by at least one Registered User
   7. Each tracking information can be tracked by at least one Unregistered User
   8. Each tracking information can be tracked by at least one Retailer
   9. Each tracking information must be part of one and only one Tracking History
   10. Each tracking information can alert Notifications zero or many times
   11. Each tracking information contains one or more item

v. Unregistered User
   1. Unregistered User may track zero or up to three tracking information per application visit
   2. Each Unregistered User can upload zero or three tracking information to track
   3. Each Unregistered User can track zero or more tracking information

vi. Tracking History
   1. Each Tracking History may be connected to zero or more Registered User
   2. Each Tracking History can be connected to zero or more Calendars
   3. Each Tracking History can contain zero or more Tracking Information

vii. Notification
   1. Each Notification must be sent to one or more Registered User

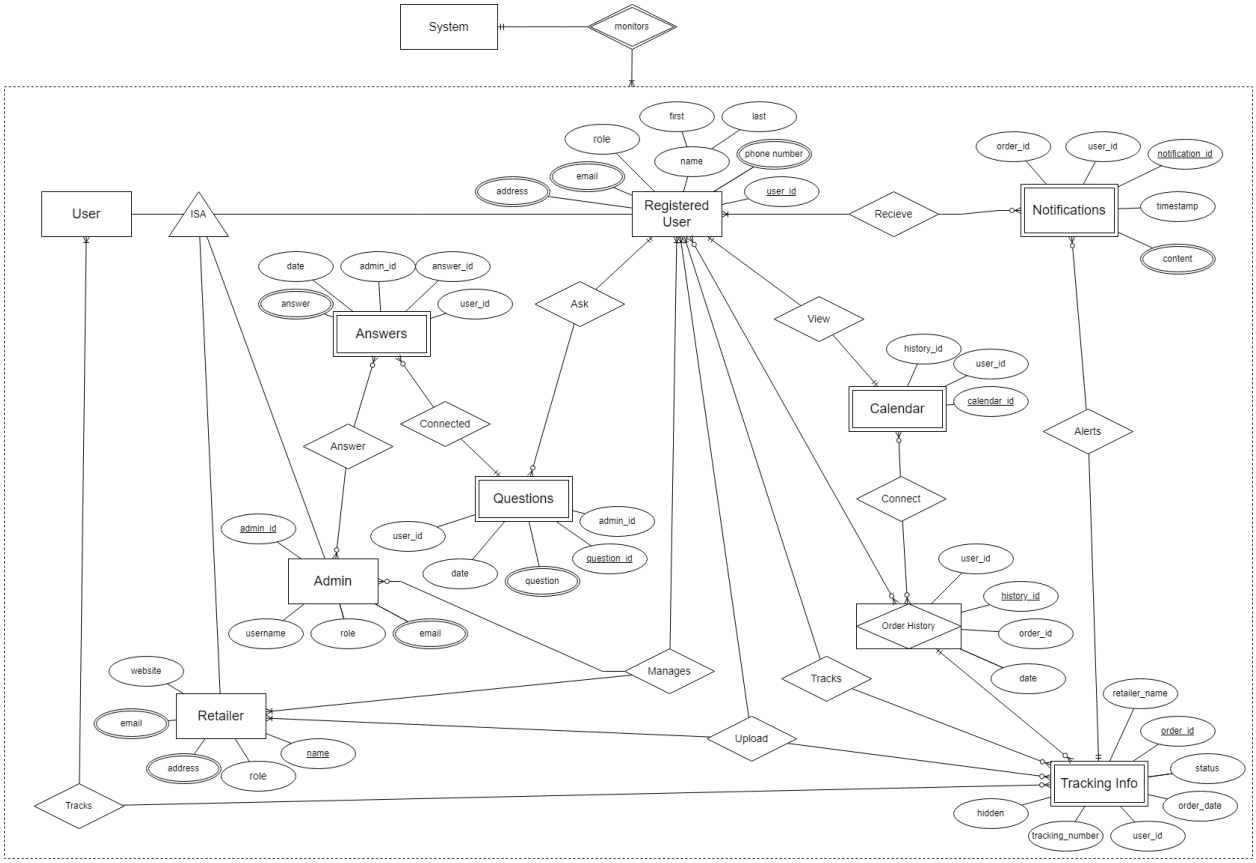
2. Each Notification must be alerted by Tracking Information only once

viii. Questions
1. Each question can be asked by one and only one Registered User
2. Each question is connected to zero or more Answers

ix. Answers
1. Each answer can be answered by one and only one Admin
2. Each answer is connected to one and only one Question

x. Calendar
1. Each Calendar shall be connected to at least one user
2. Each Calendar shall receive their information from zero or more tracking history

c. 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 shipment tracking.

    b. GPS/Mapping

        i. This API will allow OrderOwl to provide mapping and location services. This will allow user experience by offering precise location data and mapping ability. It allows OrderOwl to display real-time tracking information to users which will ensure they can easily track and visualize the location of their orders.

    c. Notification and email

        i. Notification and email API allows OrderOwl to keep users informed about their orders in real-time through these channels of communication. This is an important component of OrderOwl. By enabling notification via email or SMS, this will allow OrderOwl to send order status updates, tracking information, delivery confirmations, and important alerts. This will keep users well informed even when not on the website providing a convenient and complete way of staying up to date with their online orders.

    d. Thumbnail Retrieval

        i. This API will allow OrderOwl to retrieve a picture for each order it is tracking. This thumbnail will be displayed next to the shipping status when we view the shipment. The API will connect with various online retailers to collect and display thumbnail images of products included in users' orders.

e. Database

i. This API will be a bridge between OrderOwl and its database management system. It enables the application to interact with the database in a secure manner by encapsulating the SQL commands in a data object. Some functionality of this api are data retrieval, storage, modification, and searching. By using an API, the user will never need to learn how to use SQL to interact with the database.

2. Non-trivial Algorithms

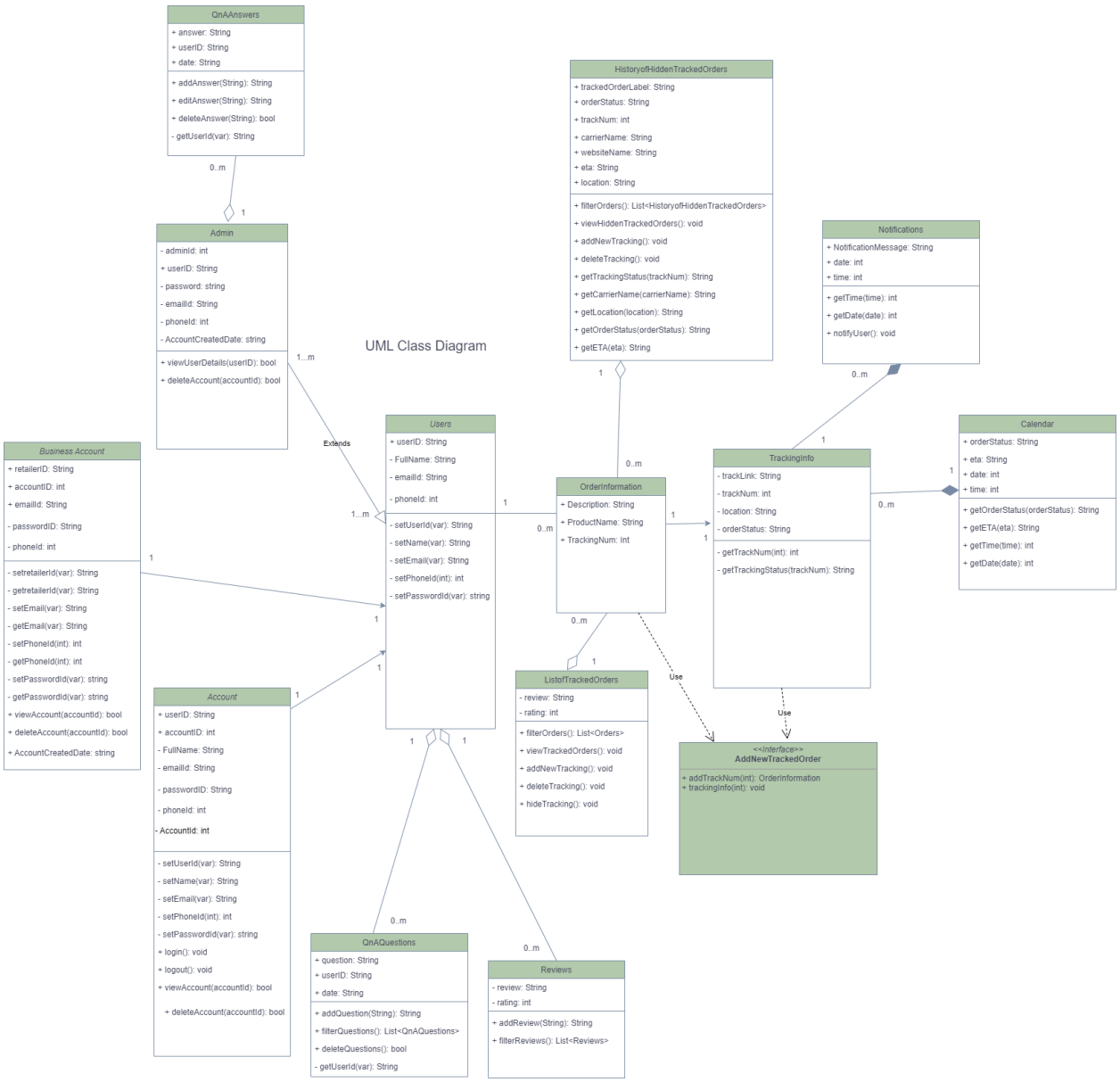
a. Search Recommendations

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. We plan to use this as a way to show users tracking information they might have meant to search. For example if a user entered a misspelled word the search algorithm should show users close matches to that word.

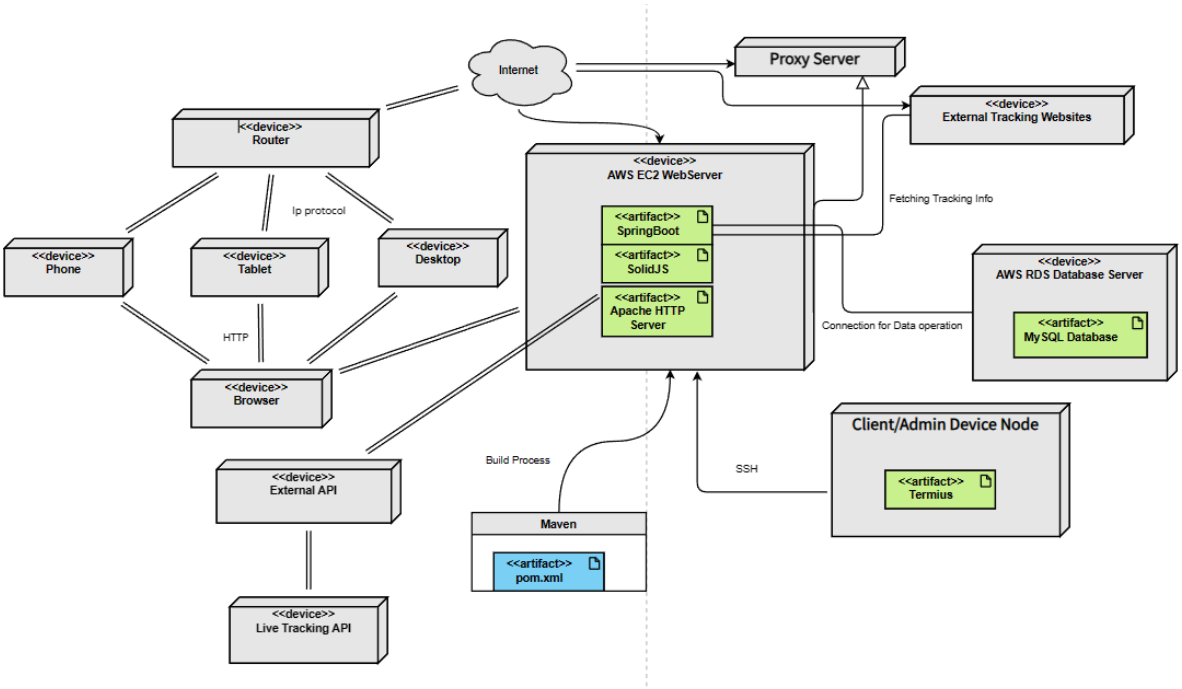
3. Tech Stack Change

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 and wrote up team meeting reports after every meeting	7

	• 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	Team Lead's score: 5 Komal's score: 7

	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	
Jimmy Pan	● In charge of section 3 specifically on creating the mockups for use cases 2-8 ● Worked with the 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 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	8
Mankit Yeung	● In charge of section 8	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 for sections ● Edited Milestone 1 V 2	
Luis Ramirez	● In charge of section 1 ● 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	8

	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