SnackPacks

Team Two Design Document

Clayton Detke, Meriem Bounab, Ryan Chen, Ryan Hennessee,
Stephen Davis, Jay Rixie

Table of Contents

Purpose

Students and residents of Burlington, Vermont have been using an unreliable, unorganized and unprofessional method (Snapchat) to order SnackPacks. Our project's focus is to provide a clean, simple to use website and an app that makes ordering, managing and distributing SnackPacks professional, reliable and organized for all parties involved.

Functional requirements

1. Cart
   As a user,
   - I want to see what is in my cart.
   - I want to be able to add items to my cart in a desired quantity.
   - I want to be able to change the quantity of items in my cart.
   - I want to be able to remove items from my cart.
   - I want to see the total price of the items in my cart.
   - I want to be able to reorder a previous order.
   - I want to be able to save an order.
   - I want to be able to customize my SnackPack.
   - I want to be able to view item allergen information.
2. Paying
   As a user,
   - I want to be able to select my payment method.
   - I want to be able to pay through the app.
   - I want the option to enter my address only once to save time when checking out.
   - I want the option to only enter my payment information once.
   - I want to be able to set up recurring orders which will automatically be paid.
   - I want to be able to opt into including the delivery tip into online payment.
   - I want to be able to opt out of including the delivery tip into online payment.
   - I want to receive a refund on cancelled, lost, or damaged SnackPacks.
3. Tracking
   As a user,
   - I would like to be able to see the status of my delivery.
   - I would like to be able to track the location of my driver.
   - I want an estimated delivery time.

   As an admin,
   - I want to be able to track working drivers locations and their statuses.
4. Reporting
   As a user,
   - I would like to be able to rate the service of the delivery driver on a scale of 1 to 5.

- I would like to be able to write about the quality of the delivery.
- I want to be able to report a damaged SnackPack.
- I want to be able to report a non-delivered SnackPack.
- I want to be able to cancel an order.

5. Reviewing

   As a user,
   - I would like to be able to leave a review for a SnackPack.
   - I would like to be able to upvote reviews.
   - I would like to be able to downvote reviews.
   - I would like to be able to sort reviews by rating and time.

   As an admin,
   - I want to be able to see the average ratings and written reviews of individual drivers.
   - I want a way to tell if a user is abusing the cancelled, lost, or damaged SnackPack refunds.

6. Drivers

   As an admin,
   - I want to be able to view a list of all drivers.
   - I want to be able to see the status of drivers.
   - I want to be able to add new drivers.
   - I want to be able to remove old drivers.

   As a driver,
   - I want to be able to view order information.
   - I want to be able to accept orders.
   - I want in-app navigation.

7. Searching / Sorting

   As a user,
   - I want to be able to browse SnackPacks and view additional information about them.
   - I want clear indications of allergens that a SnackPack contains.
   - I want to be able to sort/search SnackPacks by allergen information.
   - I want to be able to sort/search SnackPacks by user reviews.
   - I want to be able to sort/search SnackPacks by individual items within them.
   - I want to be able to sort/search SnackPacks by popularity.
   - I want to be able to sort/search SnackPacks by a combination of user reviews and popularity.
   - I want to be able to sort/search SnackPacks by price range.
   - I want to be able to sort/search SnackPacks by name.

   As an admin,
   - I want to be able to sort/search SnackPacks by least orders.
   - I want to be able to sort/search SnackPacks by most orders.
   - I want to be able to sort/search SnackPacks by trending / popular.

8. Rewards

As a user,

- I want to be able to claim rewards from the rewards program.
- I want to be able to view future rewards.
- I want to be able to view current rewards.

9. Contacting

   As a user,

   - I want a contact that I can reach out to so I can fix problems.

10. Troubleshooting

    As a user,

    - I want a help feature to troubleshoot problems.
    - I want the option to follow a tutorial upon first using the app.

11. Managing

    As an admin,

    - I want to be able to create new SnackPacks.
    - I want to be able to modify the content and prices of existing SnackPacks.
    - I want to be able to remove SnackPacks.
    - I want to be able to advertise SnackPacks.
    - I want to be able to track anonymized statistics regarding product performance.
    - I want to be able to ban users or addresses from ordering.

Non-functional requirements

1. Performance

   As a user,

   - I want the app to load in five seconds.
   - I'd like for the app to maintain to make transactions smoothly.
   - I would like for the app to run without crashing.

   As a developer

   - I'd like for the server to have minimal downtime.
   - I'd like for the server to be able to support real time GPS locations.
   - I'd like for the server and database to be able to support 50 concurrent users.

2. Security

   As a user,

   - I want to be able to control what sensors the app is able to access.
   - I want my information to be secured.

   As a developer,

   - As a developer, I'd like for the database to only be accessible by the administrator.
   - I would like the connection between the database and the server to have minimal downtime.

3. Usability

   As a user,

- I want to be able to toggle colorblind options.
- I would like the app to take up minimal space on my phone.
- I want the design to be user friendly.

Design Outline

High Level Overview

SnackPacks is a service that delivers snacks to hungry students. We are going to be using a client server model with two fat servers and two thin clients. Our first client is a web based interface for an admin to update prices, add, and edit each snack pack. Other features include the ability to view statistics and reviews on products and drivers as well as managing current drivers. The second client is the user and driver client. This will be in app form and will connect users and drivers.

Detailed Overview

Amazon Web Services

- Using Amazon's API endpoint to manage requests, Lambda to process calls to the payment server, and EC2 services to host the server.
- We will be running node on a micro EC2 instance to stay within the free tier.
- The API endpoint will be configured to relay data from the EC2 server and AWS RDS database to the user.
- The API endpoint will also be configured to relay calls to the payment server directly to implement a virtual network segregation for security practice, and create a direct route to the payment server, which will make he overall transaction faster.
- The API Endpoint will be configured for HTTPS and will return the data requested from the servers and database in JSON form, which is an easy way to send data through the API.

Payment Server

- We are going to be using Braintree as an API to handle secure payments.
- The two requests that will be forwarded to the payment server through the API Endpoint, and AWS Lambda from the user app are:
    - Requesting client tokens for the user session.
    - Payment Nonces to fulfil a transaction.

Web Client

- The administrative web client will be built using the React framework.
- It will use the AWS API endpoint to request data from the server in JSON format.
- The admin will be able to do the actions described in the user stories by having the data from the server processed to a user interface.
- The web client will be authenticated with only one user account for the owner of the company for security purposes.

User Client

- The user client will be built using the React-Native framework, which is almost identical to the React framework.
- Users and drivers will receive relevant data through the API endpoint in JSON format, which will then be processed and formatted into a simple user interface.
- This is the section of the project the users will be seeing and using, so by making a thin client it will improve user experience.
- The app will be sending requests for client tokens and payment nonces to the server.
- The app will receive client tokens and payment nonce acceptances through the server.

Design Issues

Functional Issues:

1. What information do we need for users to create an account?
    - Username and password
    - Username, password, and email address
    - Username, password, and phone number
    - Username, password, email address, and phone number

    Choice: Username, password, and phone number
    Justification: In order to create an account, a minimum of a username and password are necessary in order to identify a specific user. The user's phone number is needed for the drivers to contact them, in order to effectively deliver their SnackPacks to them. We decided against requiring an email address for account creation, although it would normally be an effective method of ensuring the security of the user's profile, because of access to the user's phone number, requiring their email address would be redundant.

2. Do users need an account to use our service?
    - No, the user only needs to input their phone number and the delivery address
    - Yes, an account is required

    Choice: No, the user only needs to input their phone number and the delivery address

Justification: While this would prevent the user from receiving the wonderful benefits of our rewards program, we decided that permitting the user to order from SnackPacks with the minimum required effort took priority. Instead of needing to input all of the required information for creating an account, the user would only need to input the bare minimum information to guarantee their SnackPacks are safely delivered to them.

Non-Functional Issues:

1. What framework/language should we use?
    - React
    - JavaScript/HTML
    - Vue
    - Angular

    Choice: React
    Justification: React requires the least time and effort to learn out of the frameworks/ languages we considered. Furthermore, React is setup in a way that appropriately complements object-oriented programming, making it very easy to use. React also offers APIs for payment systems, which greatly simplifies our goal of implementing a payment system.

2. Which payment system should we use?
    - PayPal
    - Braintree

    Choice: Braintree
    Justification: Braintree is one of the most reliable and secure payment systems accessible using React. Furthermore, Braintree offers an incredibly diverse quantity of payment options. Lastly, Braintree would be far more time-efficient to implement using React Native, as opposed to PayPal.

3. How should we host our backend system?
    - Local Server
    - Virtual Private Server
    - Amazon Web Services

    Choice: Amazon Web Services
    Justification: We chose Amazon Web Services because of how well their cost of operating scales based on the amount of requests. In comparison, using a local server seems to not be feasible, and using a Virtual Private Server would be far less cost efficient.

4. What database should we use?
    - Amazon Relational Database Services
    - SQLite
    - MongoDB Atlas

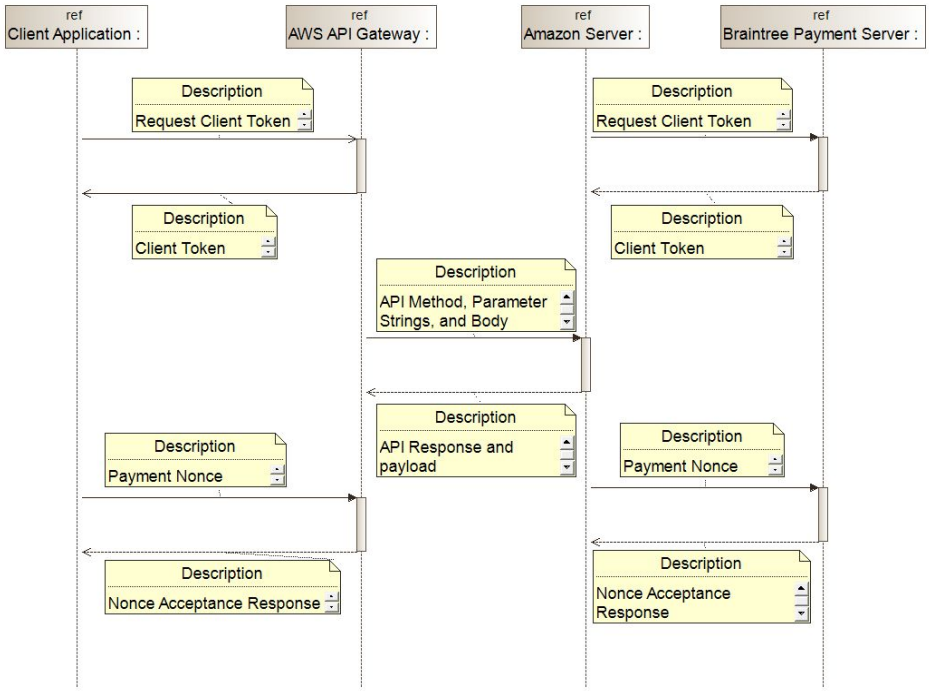
    Choice: Amazon Relational Database Services
    Justification: We chose Amazon Relational Database Services for its low cost of operation and its availability in the Amazon Web Suite. SQLite is convenient to implement, but doesn't offer very extensive backup options. MongoDB Atlas presents many usable features, but ultimately staying inside the Amazon AWS suite was deemed more convenient.

Design Details

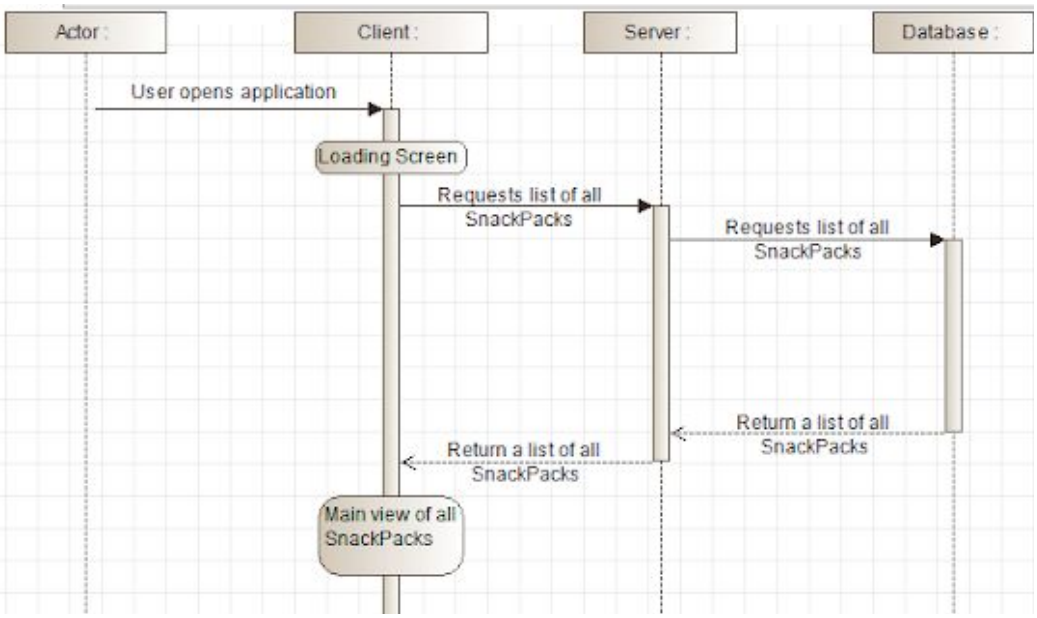
Sequence Diagrams

The sequence diagrams below depict how the client, server, and database interact for some of the most important events in our application. This includes payments, starting the application, admins sorting drivers by status, and user claiming rewards. When one of these events occur, the client makes a request to the server, the server then queries the database, the database responds to that query where the server can then perform any needed operations and respond to the client to display the result of that action.

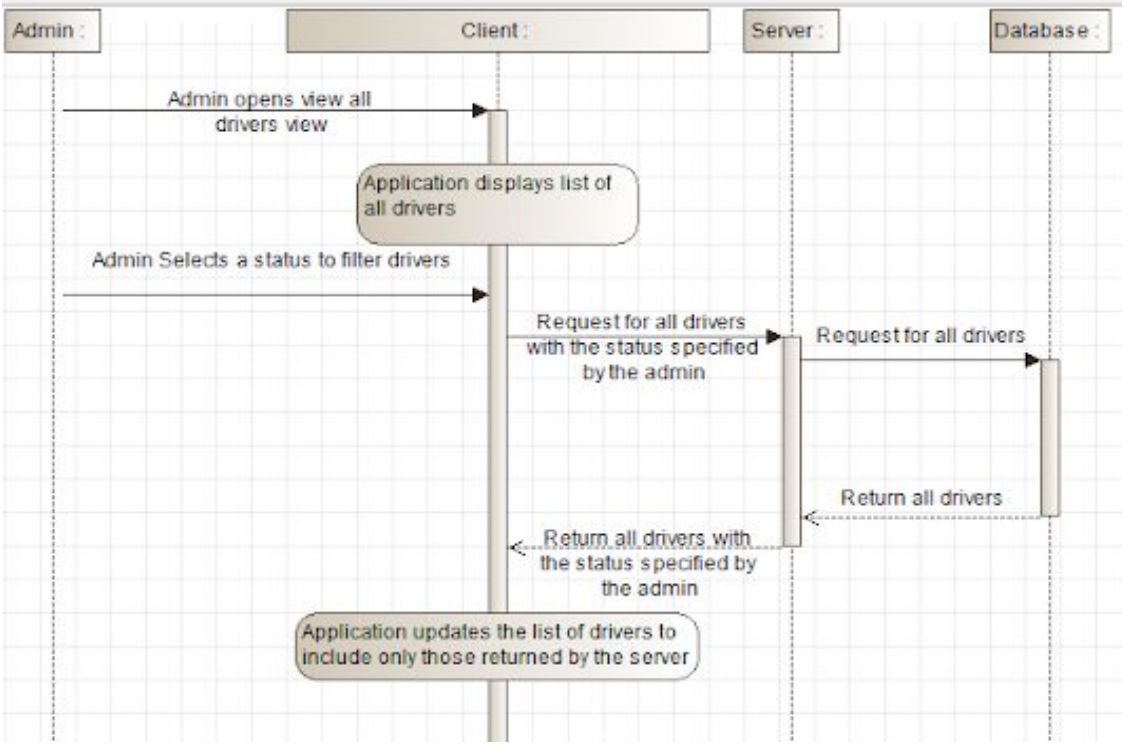
Sequence of events when user pays:



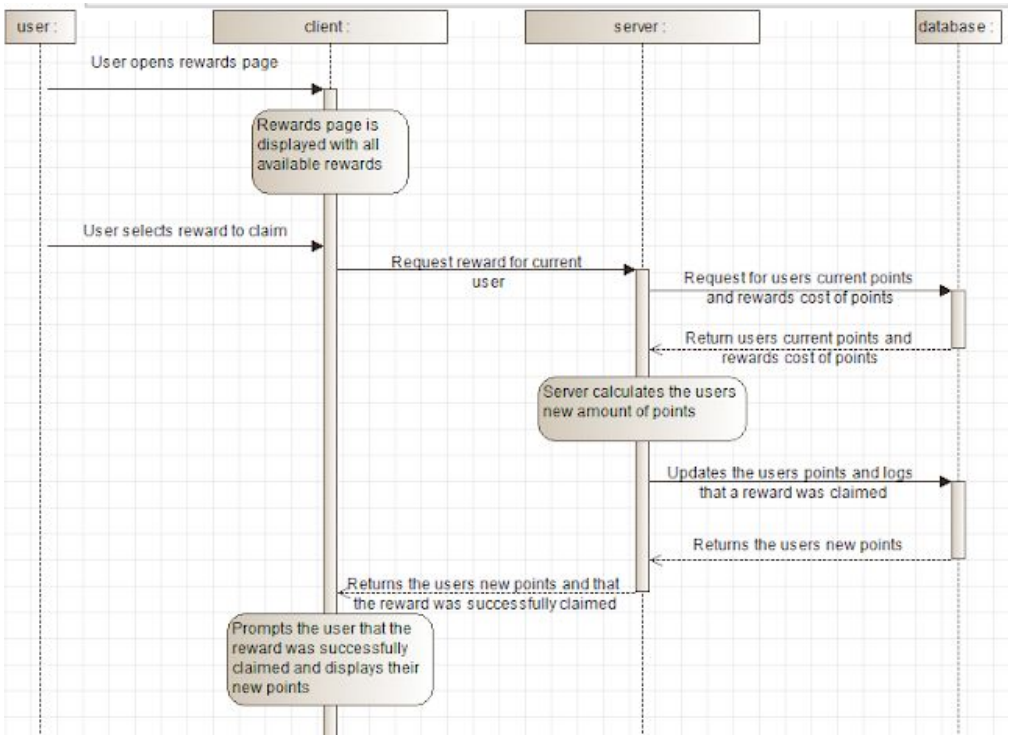
Sequence of events when user starts the application:



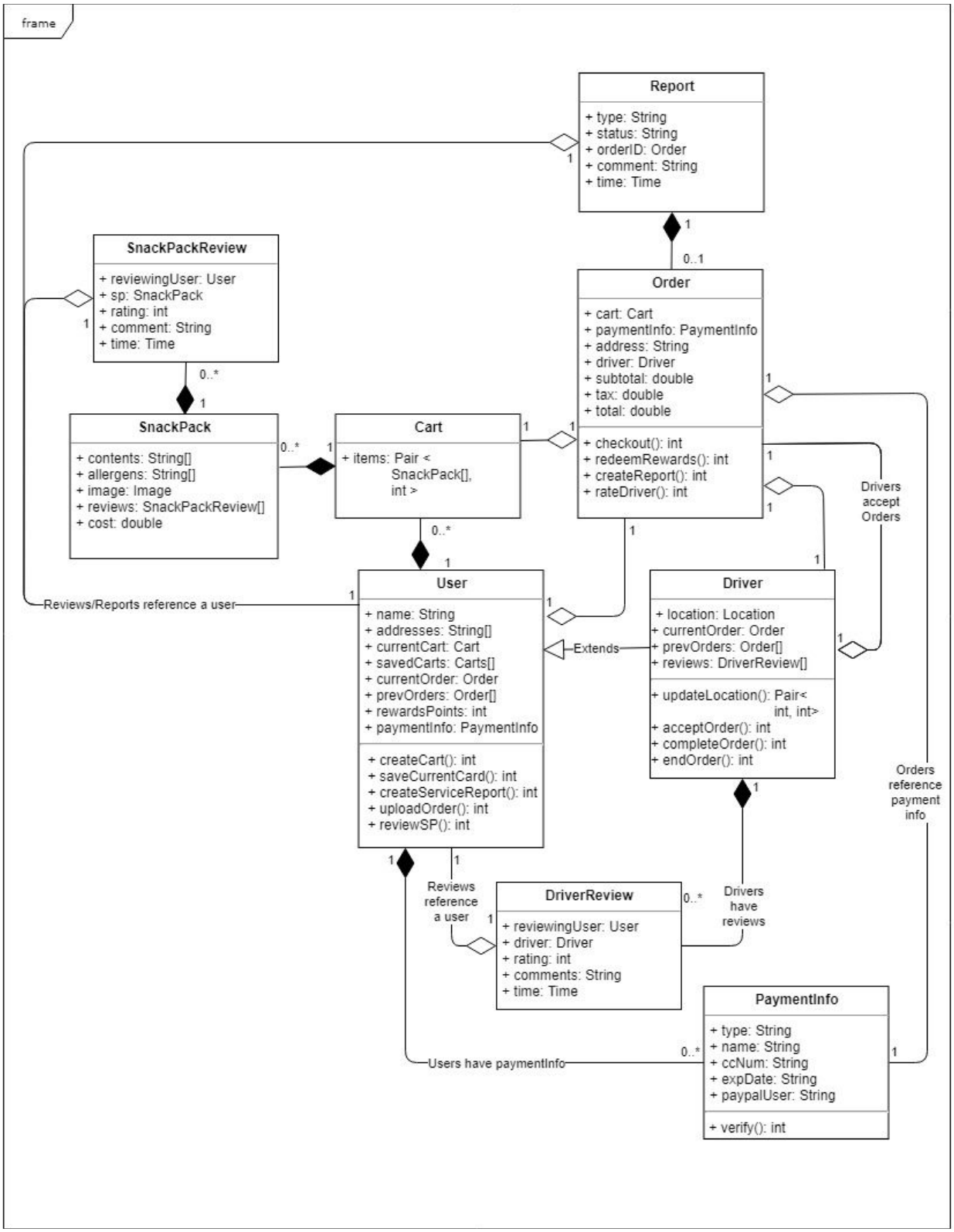
Sequence of events when admin sorts drivers by status:



Sequence of events when user accesses rewards system:



Class diagram



Descriptions of classes and interactions between classes

1) User
    - Users will have a list of saved address to make ordering easier.
    - Users will be able to manage their current cart, and save carts to make ordering easier.
    - Users will be able to track their current order, as well as reference previous orders they have made.
    - Users will have reward points redeemable at order checkout.
    - Users will be able to store methods of payment to make ordering easier.
    - Users will be able to create reports and reviews, as well as submit orders.

2) Driver
    - Driver will extend the user class.
    - Drivers will have their location tracked during delivery, to allow users to know where they are.
    - Drivers will have a current order assigned to them, to communicate status.
    - Drivers will have previous deliveries referenced, so their performance can be reviewed.
    - Drivers will have their location polled while working, to update the user on their location.
    - Drivers will be able to accept, complete, and end orders, to communicate status.

3) DriverReview
    - DriverReviews will reference a reviewing user.
    - DriverReviews will reference a reviewee driver.
    - DriverReviews will have a rating and comment to describe performance.
    - DriverReviews will have a timestamp to help time sorting.

4) PaymentInformation
    - Generate client token sends a request to the payment server to receive a client token, which gets passed back to the client application.
    - Send payment to the server, then to the payment server, which will return the acceptance response.

5) SnackPack
    - SnackPacks will have a list of their contents and allergens to display to a user.
    - SnackPacks will have an image to show them.
    - SnackPacks will reference SP Reviews so a user can reference them.
    - SnackPacks will have a cost to display to a user and the order.

6) SnackPack Review
    - SnackPackReviews will reference the reviewing user.
    - SnackPackReviews will reference the reviewee SnackPack.

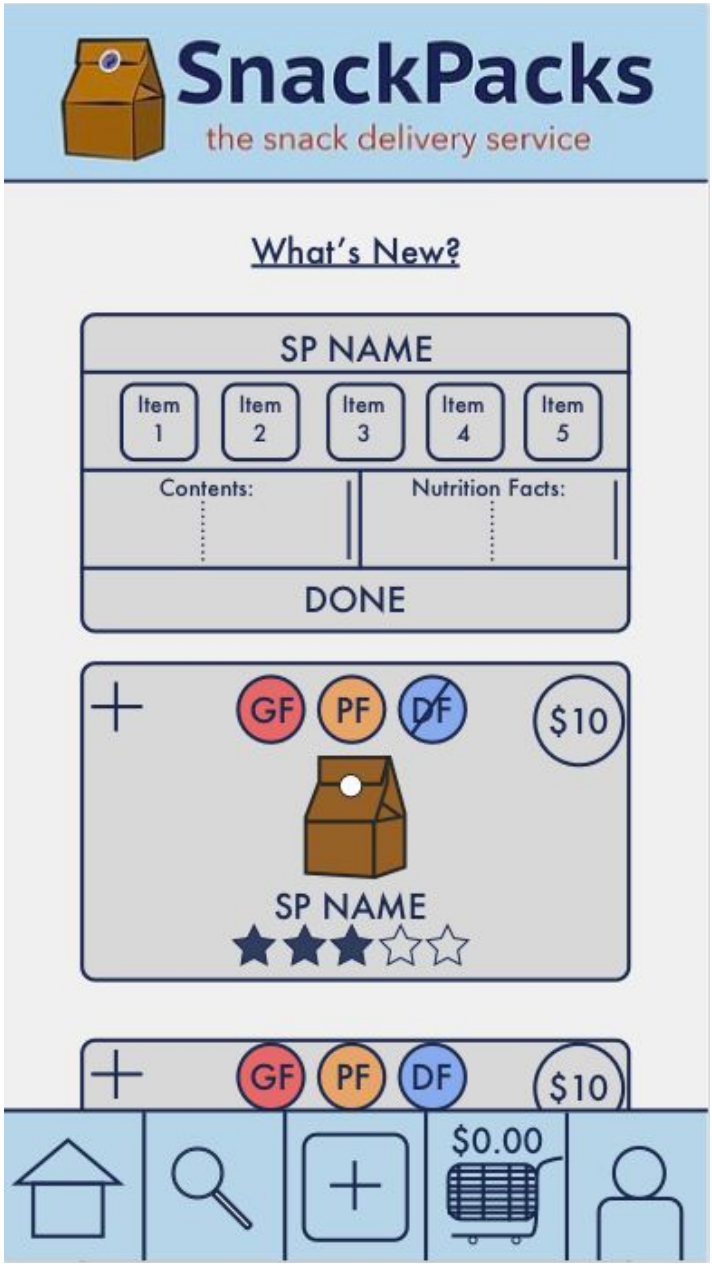
- SnackPackReviews will have a rating and comment to review SnackPack.
- SnackPackReviews will have a timestamp to help time sorting.

7) Cart
   - Carts will have a list of SnackPacks so they user knows what's in their cart, and can modify as they wish.
8) Order
   - Orders will reference a Cart to know the contents of the order.
   - Orders will reference a PaymentInfo to use during checkout.
   - Orders will reference a Driver so the user knows their active driver.
   - Orders will have an address so the driver knows the delivery point.
   - Orders will have a subtotal, tax, and total for use during checkout.
   - Orders will be able to checkout and complete transaction, and assign driver.
   - Orders will be able to redeem user rewards.
   - Orders will be able to create reports about them.
   - Orders will be able to rate the assigned driver.

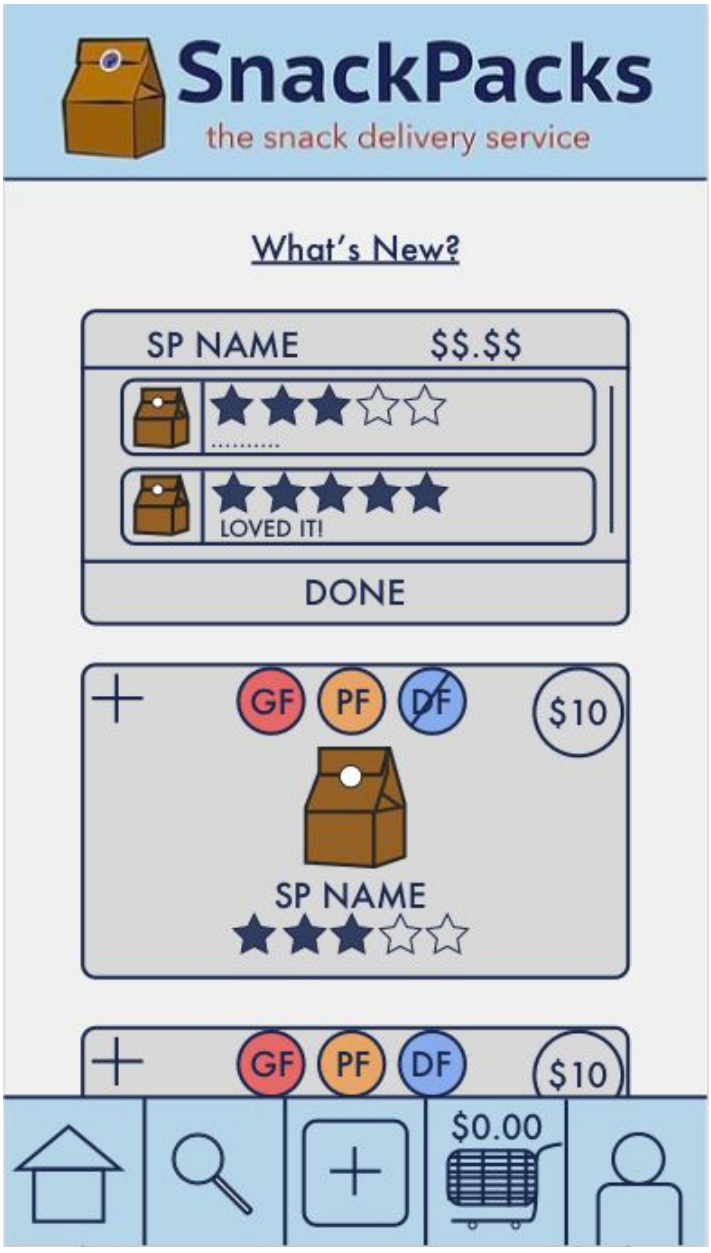
9) Report
   - Reports will have a type to indicate if their Order or Service reports.
   - Reports will have a status to indicate if they are active, under review, or complete.
   - Reports will reference an order if they are Order Reports.
   - Reports will have a comment describing the situation.

UI Mockup

Home screen: This is the initial and main view of the app. This has a feed that will contain SnackPacks that are featured, new or trending. Users can scroll through this main feed to see all SnackPacks on this page and can return to this page at any time by pressing the house icon.



SnackPack Detailed View: When a user presses on a SnackPack listing, they can see more details including details about the items that are contained in the SnackPack and Nutrition Facts on all items

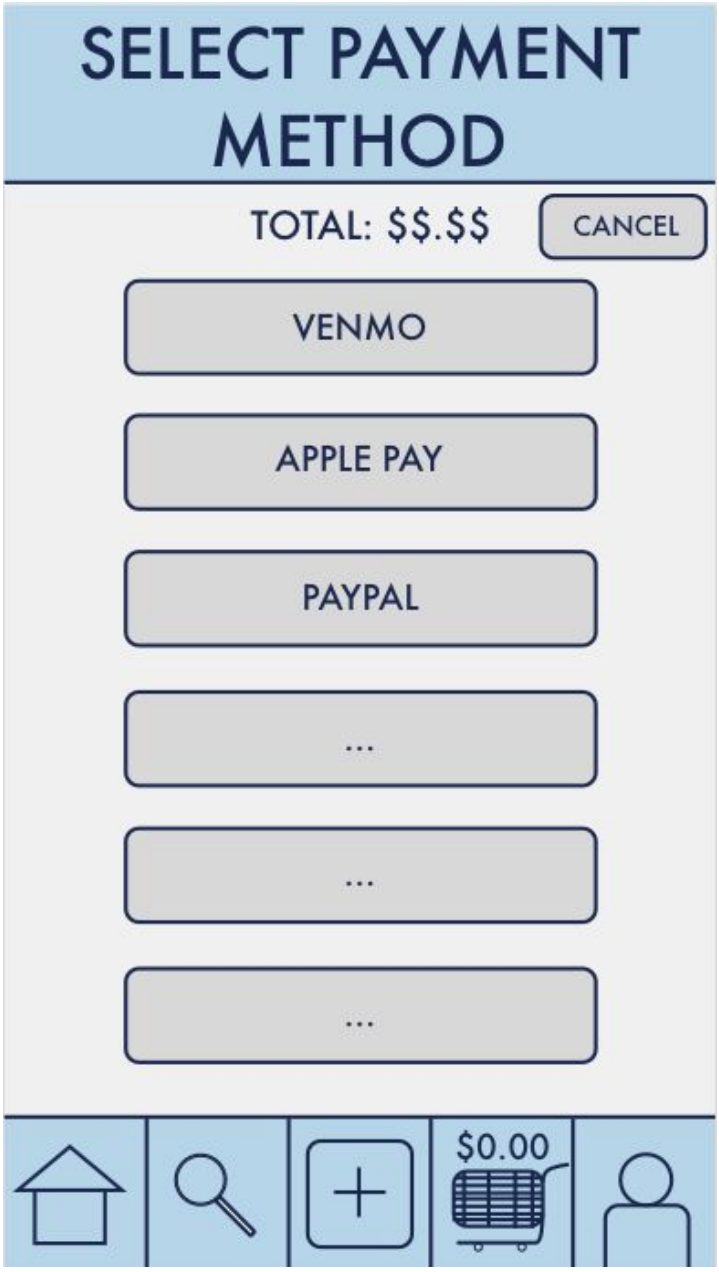


Review Detail View:
When scrolling through the main SnackPack Feed, a user can click on the number of stars that a SnackPack has in order to see the reviews that users have written about it. The user will only see a preview of the review and can select a review to expand it and read the complete review.

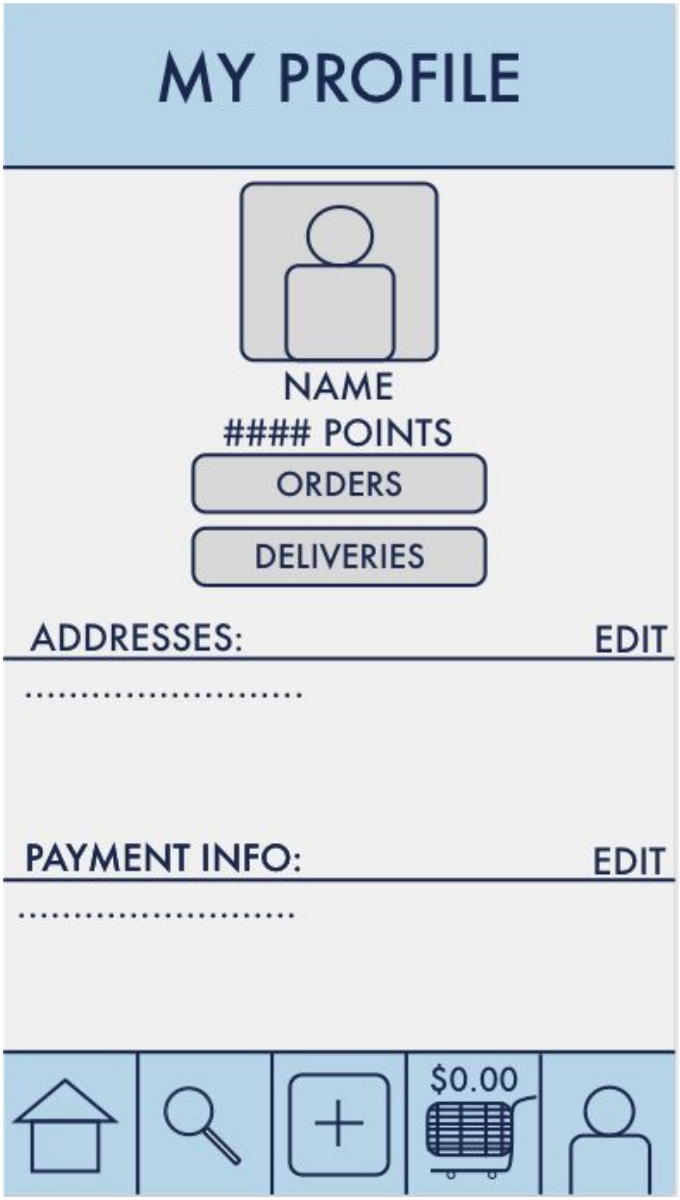
Search Screen:
A user can navigate to this page by clicking on the magnifying glass icon on the bottom menu. Here, a user can search for a SnackPack by allergen, name of SnackPack, or item contained within a SnackPack. A user can also sort the results by Price: low to high, Price: high to low, A-Z, Average Customer Reviews, Newest Arrivals

Custom SnackPack Screen:
A user can navigate to this page by clicking the plus icon. Here, a user can create their own custom SnackPack by adding items and subtracting items from the selection of individual items displayed. The user can see the total subtotal for the SnackPack that they are creating as well as adding the SnackPack to their cart once they're are finished.

My Cart Screen:
A user can navigate to this page by pressing on the cart icon at the bottom. Here the user can see the subtotal for all the items that are currently in their cart. The user can scroll through the items in their cart and see the individual quantity and subtotal for each SnackPack and edit the quantity or delete the SnackPack all together from their cart. They can also checkout their cart by pressing the order button.



Select Payment Screen:
A user gets to this page by pressing order when they are viewing their cart. Here, the user can choose their method of payment and will complete their payment or they can cancel to return to their cart.

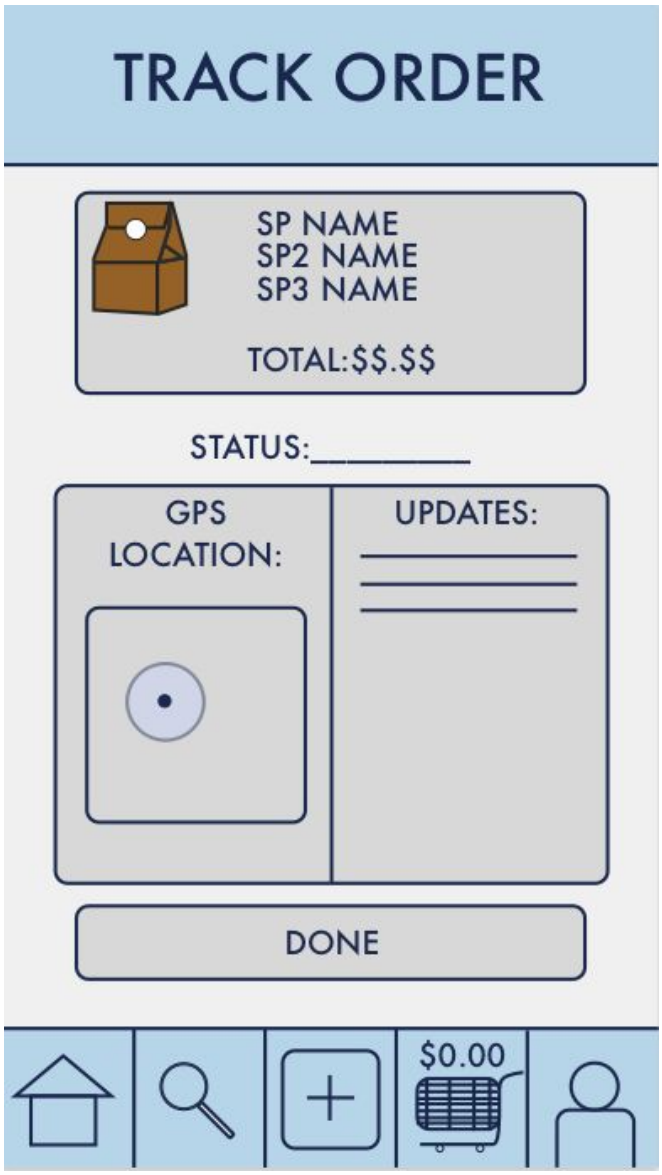


My Profile Screen:
A user can navigate to this screen by pressing the person icon in the bottom right hand corner. Here a user can see their profile picture, the number of points they have earned through purchasing SnackPacks, the address(es) the app has on file, as well the payment information that the app has on file as well as edit this information. They can also see their orders or deliveries only if their profile is registered as a driver.

My Orders Screen:
A user navigates to this page by pressing ORDERS from the MY PROFILE screen. Here a user can review an item, if it has been delivered, track an item, and cancel an order if it has not been shipped yet. A user can also check the status of their orders.

Leave a Review Screen:
A user navigates to this screen by pressing on REVIEW from the MY ORDERS screen. Here, a user can give a SnackPack a rating out of 5 stars as well as a written review of the SnackPack. The user can then cancel or submit their review.

Cancel Order Screen:
A user can navigate to this page by pressing on CANCEL from the MY ORDERS page. The user can only access this page if their order hasn't shipped yet. Here, the user can select a reason for cancelling their order and submit or cancel their request to cancel their order. If the user chooses to submit their request for cancelling their order, they may receive a refund



Track Order Screen:
A user navigates to this screen by pressing TRACK from the MY ORDERS screen. Here, a user can see the status of their order, shipping updates and the current GPS location of their package, if available

My Deliveries Screen:
A user can navigate to this screen only if their profile is registered as a driver and can press DELIVERIES from the MY PROFILE screen. Here the user can claim a delivery to make and can see the destination address and the amount owed, and whether or not the driver should collect cash from the customer or if the order was paid online.