Home Food Delivery Application Coding Scenario - Second Release
Group 2 - Ish Soni, Soham Patel, Krishna Bavana, Hari Seelam

Summary of First Release

As of right now, we have implemented account creation with validation of user information for both cooks and customers. Moreover, the cooks and customers can navigate through a taskbar to access different aspects of the application they need.

The customer can view all the different cooks and select menu items without any regards to purchasing. If the customer wants to switch accounts and access their cook page, then they can switch roles using the hamburger menu.

When the cooks sign up, they will be able to add to their menu, which will be uploaded into our database, and view their menu as well. If the cook wants to switch accounts and access their customer page, then they can switch roles using the hamburger menu.

Scenario "Home Food Delivery - Driver"

A feature of the Home Food Delivery App implemented in this scenario is the driver. The third and final user of our app will be a driver who delivers food from the cook to the customer. For this release, we want to integrate the user interface for the driver. First, a user must sign up to be a driver when they click on the driver from the hamburger menu. They will then be directed to a main driver page with a taskbar, designed similarly to how the customer and cook taskbar are. Once the driver toggles an "on" button, then they can choose any current order that they want to deliver. Once they choose that, they can view the status of that order until the order is delivered. In this scenario, we will not be focusing on GPS tracking regarding the driver location.

Scenario "Home Food Delivery - Ordering"

A feature we would like to add to the app is ordering. If the user is a customer, they will be able to select a cook to order from. They can select menu items and add them to their cart, specify the quantity that they want to order, when they would like the order to be ready and delivered (can specify a future date/time) and list any special requests they have with that menu item. For example, the customer can request that the cook make the item without sesame seeds. The customer will then finalize their order in the cart and place their order.

If the user is a cook, they will be able to see the orders that they have to fulfill. They can mark the progress of their order, and once it is finished for the date/time specified they can mark it as "ready to be picked up". Once the order has been picked up by the driver, they can mark it as "picked up".

Scenario "Home Food Delivery - API's"

There are three main API's that we want to integrate within our application. The first API we want to integrate is the Google Maps API. This API will enable the customers to see available cooks in nearby locations that they can order from. After placing an order, the customer will also be informed of the estimated arrival time based on the route of the driver. Moreover, the drivers will be able to see the location markers from the cook and the customer as well as the quickest route from the cook and the customer.

The next API we aim to add to our application is the Open Weather API. This enables the drivers to see the current location's weather conditions so that they can compute the travel challenges and update the estimated arrival time if necessary.

The final API that we want to implement is the EIA Open Data API. This API is used to track the current gas prices so that we can compute travel costs for the drivers.

<u>Figure 1 - UML Diagram for Order and Driver From HomeFoodDelivery.pdf</u>

Order
+ orderId: int + customer: Customer + cook: Cook + driver: Driver + food: array<FoodItem> + pickupAddress: Location + timeOrdered: String + status: string + orderCompleted: boolean + outForDelivery: boolean + delivered: boolean +timeDelivered: string
- uploadOrder(): void + sendRoute(): Route

Driver
+ driverID: int + name: String + address: String + emailAddress: String + phoneNumber: int + licenseId: int + VIN: string + rating: int + deliveries: int + feedback: array<DriverFeedback> + routes: array<Route>
- updateDelivery(): string - updateTime(): string - markDelivery(): boolean - calculateRating(): int + removeRoute(): boolean + addRoute() + assignOrder()