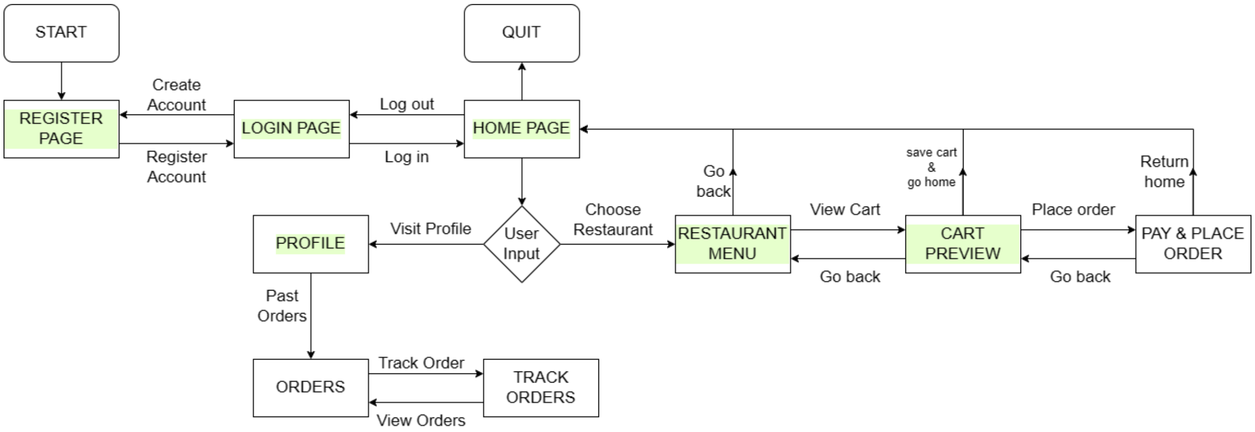
CISC 327 - Prof. Anwar Hossain - 2023-10-20
Group 5: Michael Osei-Ababio, Ryan Pleava, Keaton Tom

Online food delivery platform - Front-end Rapid Prototype

Link to Front-end demo: main--leafy-buttercream-e6d2fc.netlify.app/registration
*Note: Please try to click everything and play around with all of the features (filtering, adding to cart, removing from cart, etc)

App-Control Flow:

Below is the app control-flow diagram. It is a diagram depicting the necessary screen states a user must pass through to reach other areas of the program. It also defines actions within the frontend system that lead to a change in system state. The screens containing green are the ones that have already been prototyped.



Screens and underlying Screen Logic:

Profile:

<table>
<tr>
<td>User</td>
<td>
<table>
<tr><td>name</td><td>A string denoting the name of the signed in user</td></tr>
<tr><td>telephone</td><td>A string denoting the telephone number of the signed in user</td></tr>
<tr><td>favouriteRes</td><td>An array of strings corresponding to the favourite</td></tr>
</table>
</td>
</tr>
</table>

<table>
<tr><td></td><td>
<table>
<tr><td>taurants</td><td>restaurants of the signed in user</td></tr>
<tr><td>pastOrders</td><td>An array of properties of past orders, containing restaurant names, order dates, and total amounts for each respective order
<table>
<tr><td>restaurant</td><td>String denoting restaurant for a context dependant previous order</td></tr>
<tr><td>orderDate</td><td>String denoting the date of the order</td></tr>
<tr><td>totalAmount</td><td>String which corresponds to the total amount paid for a past order</td></tr>
</table>
</td></tr>
</table>
</td></tr>
</table>

The profile page exists to allow users to see their own account information. Along with a photo, this page allows signed in users to view their name, telephone number, a list of their favourite restaurants, and a list of their past orders. This is done with an effective *User* class.

Track Orders:

<table>
<tr><td>Order</td><td>
<table>
<tr><td>restaurant</td><td>String denoting restaurant ordered from</td></tr>
<tr><td>items</td><td>Array of strings per each item ordered from restaurant</td></tr>
<tr><td>modifications</td><td>String denoting additional modifications made in an order</td></tr>
<tr><td>timeOfOrder</td><td>Time order was made</td></tr>
<tr><td>expectedDelivery</td><td>Expected delivery time of order</td></tr>
</table>
</td></tr>
</table>

Driver	name	Name of delivery driver
	carModel	Car model of delivery driver
	licensePlate	License plate of delivery driver

The Track Orders page exists to allow users who have just placed an order to track when their order will be delivered along with information about the driver who will be delivering the order to them. This is executed with the use of classes for both and order and for a driver.

Registration:

User	email	Variable that allows users to input a string which can then be stored in database
	username	Variable that allows users to input a string which can then be stored in database
	password	Variable that allows users to input a string which can then be stored in database
	register	A process that takes users to login page and saves info to database
	sign-in	A process that takes users to login page

The registration page is meant for users to be able to create accounts if they do not already have one. Users can input an email, username, and password for their account creation and will be saved to the database and they will be brought to the login page. If users already have an existing account they can click the sign-in button and it will take them to the login screen where they can enter their credentials.

Login:

User	username	Variable that allows users to input a string which can then be stored in database
	password	Variable that allows users to input a string which can then be stored in database
	Log in	A process that takes users to home page

The login page exists so users can enter their valid credentials and be brought to the home page with their preferences and information linked to their account. Users can input a username and a password and if they have entered valid information then upon a click of the login button they will be brought to the home screen.

Pay & Place Order

User	pay	Variable that allows users to input strings which will be stored as the users payment method
	address	Variable that allows users to input a string that will be the address the order will be delivered to
Order	items	String that represents the items that are apart of the order
	track	Process that allows users to track orders after being placed

<table>
<tr><td></td><td><table><tr><td></td><td>by redirecting users to track orders page (not yet implemented)</td></tr></table></td></tr>
</table>

The pay and place orders page exists so users can enter payment information and where they want food to be delivered to. The items of the order will also be displayed to the user. After payment is confirmed users can press track order to see the progress of the order.

Home

The home screen served as the center for the application. Displayed on it is a navigation bar for traversing the site, and an organized menu of restaurant options to choose from grouped by their respective food type called categories. Click on a restaurant to go to its menu.

Category	Categories are lists of restaurants that serve similar foods. In the case of the frontend, restaurants will appear grouped by their respective categories, categories shown can be filtered. In the source-code, this is denoted by the <RestaurantFeed> component, which holds the list of restauarants.
Restaurant	Restaurants are the individual restaurants that belong to/in a <RestaurantFeed> component. Restaurants can be filtered by their price, category and name, although name search is not yet implemented and will come along with the backend. In the source code, this is denoted by the <RestaurantFeedItem>, which contains information about the restaurant such as food category, name, menu, etc. Clicking on a restaurant card will open up its menu, where you can select different items.

Restaurant Menu

The restaurant menu screen is where the actual menu items are displayed. Every item has an image, name, price, and calorie measure. In this menu user can click on the items they would like to add to their cart.

Restaurant Menu Items	Restaurant menu items are the available items that a restaurant has available to order from. A menu item contains the name, price, calories, and ingredients (not yet implemented) of the meal. They will be pulled from the backend and are linked to a particular Restaurant. In the source code, these are denoted as the <MenuItem> component.
Cart	The cart is an object that contains a list of all the chosen menu items the user has selected to add so far. The cart holds the information for each menu item inside of it, including price, name, calories, and also has a subtotal that actively recalculates itself.

Cart Preview

The cart preview screen is a sub-screen of the restaurant menu screen and only pops out of the side of the screen like a widget. It displays the items currently added to the user's cart, and allows users to remove items from the cart as well. Although not implemented yet, the cart preview screen will contain the control for progressing to the next stage of ordering, as that is really the object that contains the information needed moving forward (ie. ordered items, prices, etc).

Cart	The cart is an object that contains a list of all the chosen menu items the user has selected to add so far. The cart holds the information for each menu item inside of it, including price, name, calories, and also has a subtotal that actively recalculates itself.
Cart Items	A cart item is an individual restaurant menu item that has been added to a cart. Unlike a generic <MenuItem>, an item

	added to a cart also has an individual id to separate it from other <MenuItems> of the same type. Cart items can be removed from the cart as well.