Requirements Definition

Group 2

Dan’s Cool Beans

1. **Introduction and Context**

Logan is full of thirsty college students, and it is up to us to satisfy their desires. With this new coffee shop we will have a large volume of customers, so it is up to us as developers to create an easy to use system that manages the various logistics that go into running a coffee shop.

Here are some basic requirements that need to go into our app:

* Keep track of inventory
* Keep track of users and employees
* Create a UI system where customers can order their customized drinks based on ingredient availability.
* Notify Customers when their drink is done
* Track of the money between the various users

1. **Users and their goals**

Situation 1: Customer creates online order

Entry Conditions: User is logged in with username

Exit Conditions: Coffee is made

Flow:

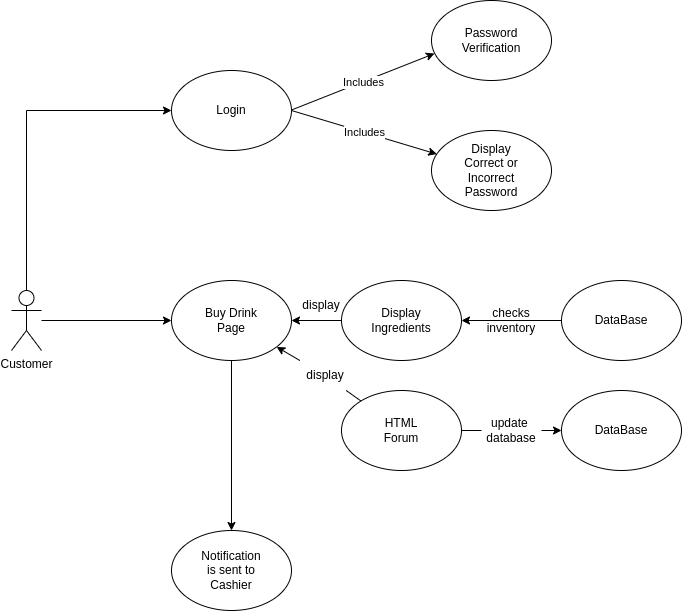
Customer logs into website

Customer picks custom ingredients to add to their drink

Database will update its inventory

App will notify Employee that an order is to be places

Employee will make order and submit done



Situation 2: Cashier creates order for customer

Entry Conditions: Employee gets a notification to make coffee

Exit Conditions: Order is confirmed

Flow:

Customer is logged in

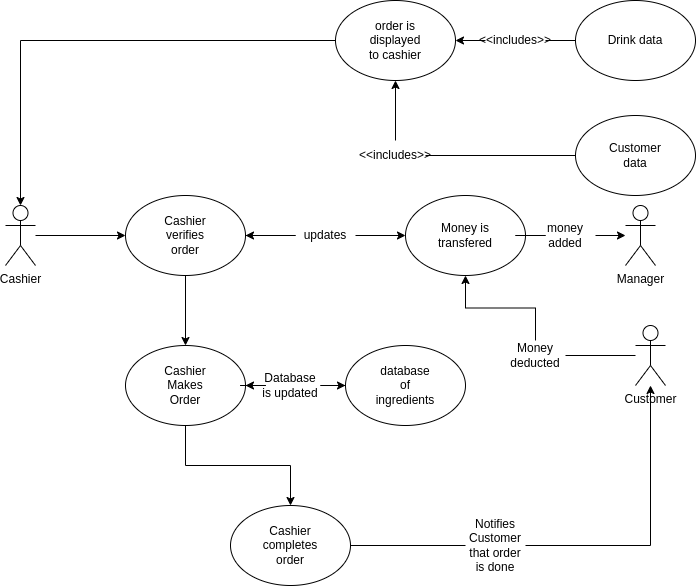
Customer creates custom coffee drink

Cashier is notified of order

Cashier queries ingredients

Cashier verifies order

Website notifies customer of order



Situation 3: Cashier submits order to barista

Entry Conditions: Cashier submits order to barista

Exit Conditions: Barista submits a notification to cashier

Flow:

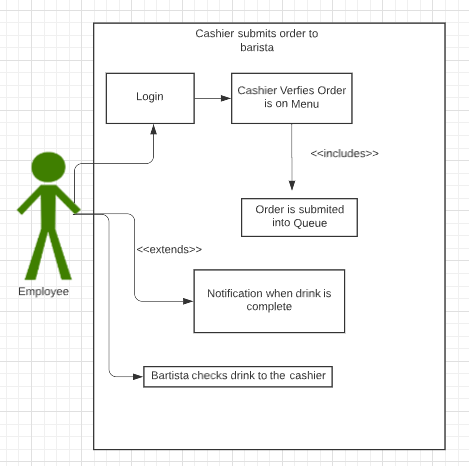
Cashier is notified of order

Cashier logins in

Cashier verifies order

Cashier puts the drink into the queue

Queue notifies the barista that a drink is needed to be made



Situation 4: Order Completion

Entry Conditions: Order notification for Barista

Exit Conditions: Barista submits drink to cashier

Flow:

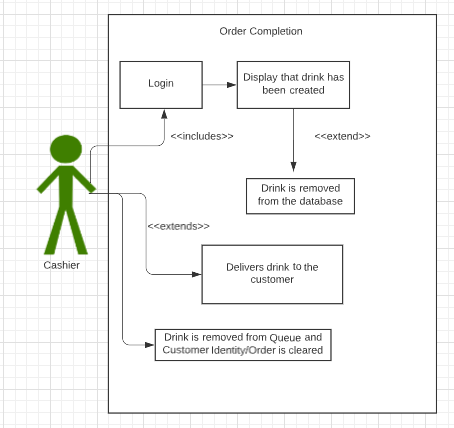
Cashier submits order into the queue

Barista gets a notification

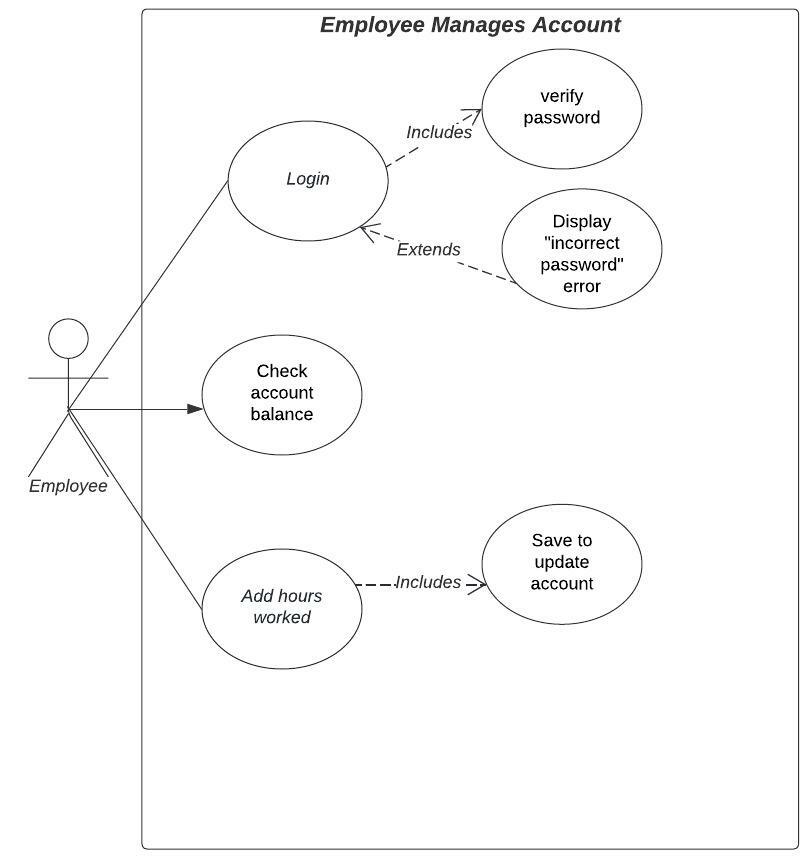
Barista makes drink

Barista notifies cashier that drink is finished

Cashier removes drink from the queue



Situation 5: Employee manages account



Participating actor: Employee

Entry Conditions:

* Employee wants to check balance or has worked and needs to report hours

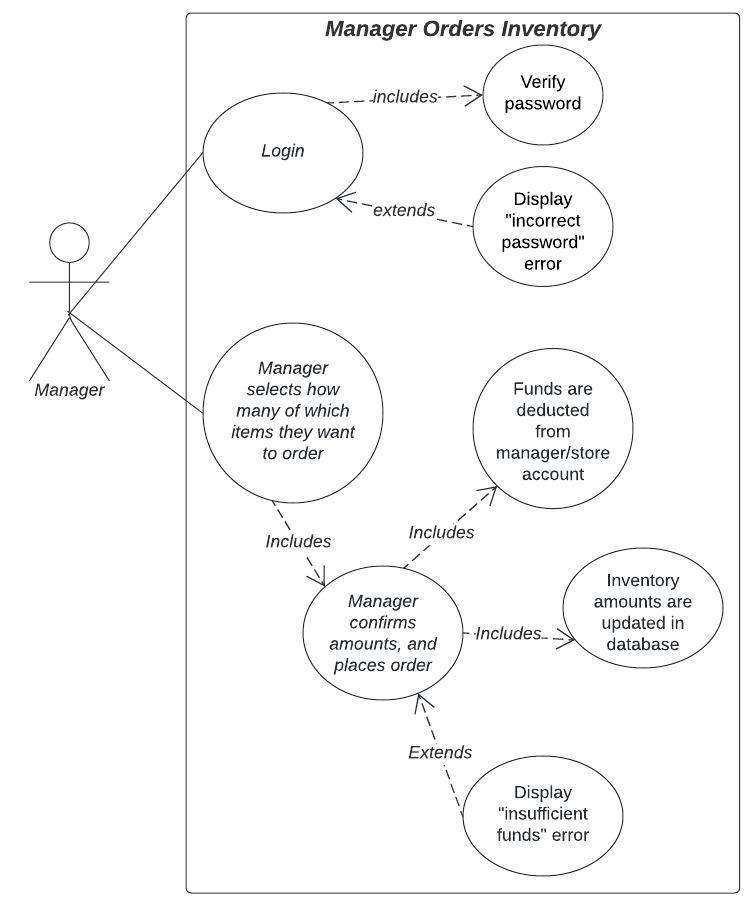
Exit Conditions:

* Employee leaves page or saves

Event Flow

1. Employee logs into their dashboard / account
2. Employee checks their account balance to see how much money they currently have
3. Employee enters hours worked into the dashboard
4. Employee saves to update hours worked during current week

Situation 6: Manager order inventory



Participating actor: Manager

Entry Conditions:

* Manager wants to order more store inventory

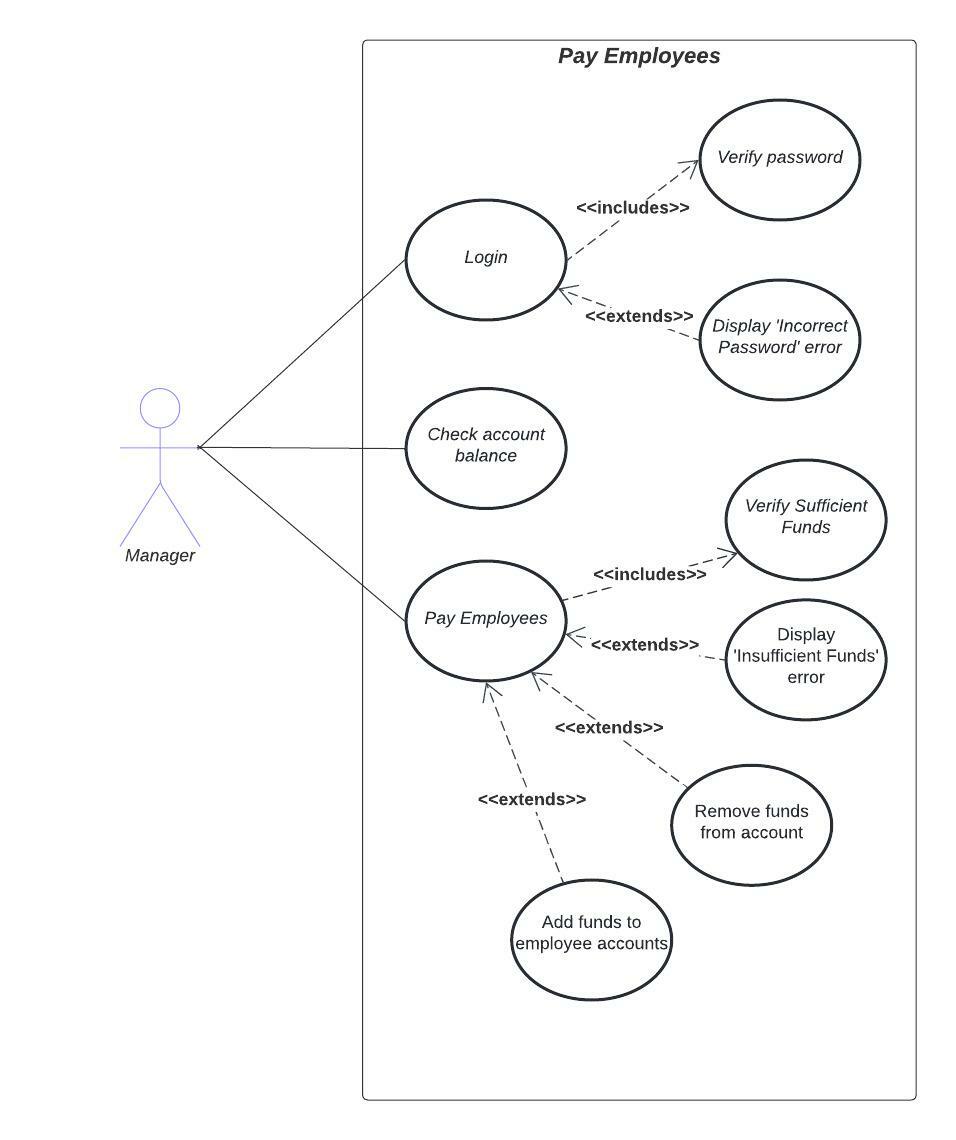
Exit Conditions:

* Database is updated with a successful order being placed

Event Flow

* Manager logs into their account
* Manager chooses which items and how many of them to order
* Manager confirms amounts and price
* Manager submits order
* Funds are deducted from account, if there aren’t enough, insufficient fund error displays and order is canceled
* When funds are removed successfully, the database is updated with new amounts of inventory items

Situation 7: Manager pays employees

**

Participating actor: Manager

Entry Conditions:

* Employees have entered billable time to their accounts

Exit Conditions:

* Employees are paid

Event Flow

Manager logs into their dashboard / account

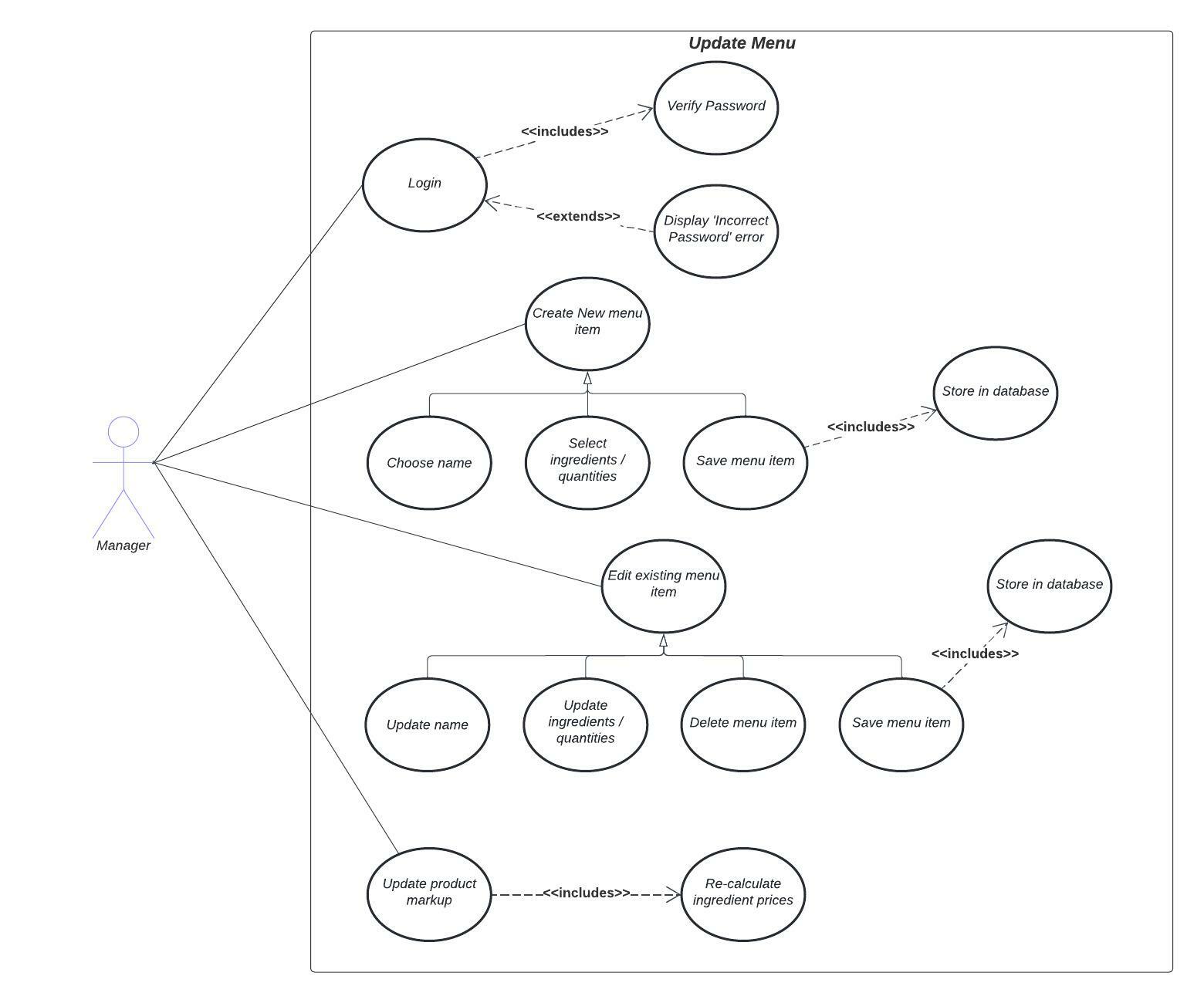
Manager checks their account balance to ensure they likely have enough money to pay their employees

Manager clicks a button to pay employees

System checks to see if there are sufficient funds in the manager account to make the payment

If so, the system removes the amount from the manager account and allocates funds to each employee account based on hours worked multiplied by wage

Situation 8: Manager updates Menu



Participating actor: Manager

Entry Conditions:

* Manager knows the ingredients of the drink they are creating/editing
* Manager determines adequate markup amount based on business-related variables (i.e. income vs expenses, supply vs demand)

Exit Conditions:

* Menu is updated

Event flow:

1. Manager logs in to their dashboard / account
2. Manager enters menu editor and chooses to create or edit a drink
   1. Manager (re)names the product and enters/updates the ingredients required to make the drink.
   2. Manager saves their changes
   3. System stores those changes
3. Manager updates product markup
   1. System calculates product markup with customer-facing ingredient prices

Situation 9: Employee creates Drinks

Diagram

Description automatically generated

Participating actor: Barista (Employee)

Entry Conditions:

* Barista gets the ticket/order from the cashier

1. **Functional Requirements**

Single web app that handles everything

* User abilities
  + Customer
    - Sign-up
      * Unique username
      * Encrypted password
    - Account management
      * Add money to their account
    - Online order
      * View menu
      * Select Drink
      * Customize ingredients/add-ons (optional)
        + Increment/decrement ingredients
      * Submit order
      * Money leaves account
        + *Goes to manager*
      * Go to Cashier
      * Give cashier username to look up the order
      * Wait for drink to be made
    - In-person order
      * Talk to Cashier
      * Give cashier username to look up the order
      * Tell cashier what they want
      * Wait for drink to be made
  + Cashier
    - Customer-facing
      * Gets customer’s username
      * If customer has not already submitted an order:
        + Cashier creates a new order for the customer
        + Cashier removes money from customer balance

*Money goes to manager*

* + - * Cashier submits it to Baristas
      * Cashier is notified when drink is complete
      * Cashier has queue of completed drinks
        + Click ‘Deliver’ button to deliver drink to customer
    - Account management
      * Enter hours worked
      * View payment/account balance
  + Barista
    - Customer-facing
      * Queue of submitted orders
      * Make the drink
      * Click order complete
    - Account management
      * Enter hours worked
      * View payment/account balance
  + Manager
    - Edit the menu
      * Allow them to upload a link to the image?
      * Set price of drinks
        + Price of ingredients + markup
        + Maybe manager just chooses the markup and it’s a global setting

Manager could override specific drinks?

* + - Manage inventory
      * View ingredient inventory
        + Include cups in inventory
      * Add to the inventory (by paying for inventory items)
        + *Money is removed from manager account and sent into the abyss*
    - Add money to their own account
    - Add employees
      * Choose employee role
      * Assign username/password
    - Pay employees
      * *Money is removed from manager account and added to employee account*
* Menu actions / functions
  + Set menu of drinks (that can be customized)
    - Each drink has a list of ingredients that are part of the inventory
  + One drink size
  + Drink list item
    - Display drink name, ingredients, ‘Buy Now’ button
  + Drink customization
    - Pre-populated option fields with drink ingredients
    - Increment or decrement drink ingredients
    - Updated drink price
    - ‘Order’ button
      * If a customer clicks order, it *should not* go straight to barista
        + “Add to Order” so users can buy multiple drinks
        + Confirm purchase on order page
      * If a cashier clicks order, it *should* go straight to the barista

Views needed:

* Customer sign-up
* Customer log-in
* Customer dashboard
* Employee dashboard (cashier mode / barista mode? -> tabs?)
* Manager dashboard
  + Store inventory management
  + Menu editor
  + Employee editor
* Menu (list of drinks)
* Drink customization (‘buy now’ page)
* View order page

NOTE: User hierarchy should allow higher tiered users to access the views in tires below it

1. **Non-functional Requirements**

Menu:

* Set menu of drinks that can be customized
  + Each drink has a list of ingredients that are part of the inventory

No front facing menu required (without login)

Users can make their own accounts

Customer can make an order online, go to the cashier and give them their username, and then the cashier says ok we’ll start making that drink

OR

Customer can go straight to cashier, give them their username, and the cashier will start a new order for them and submit it

1. **Future Features**

Add new limited edition ingredients that show up at the top

1. **Glossary**

This section includes a list of important terms and their definitions.

*Customer -* a user that interacts with the system to order and pay for frappuccinos

*Manager -* a user that pays employees, orders inventory and edits the menu

*Employee -* a user that can act as a cashier or barista, to take and complete orders

*User -* refers to any of the three types of users in the system