67-262 Database Design and Development, Fall 2025
Project Phase-2 Report

Project Title: *Instacart*

Team Number: *T3-14*

Team Members: Jessica Cai (zhihanc), Grace Tian (gracetia)

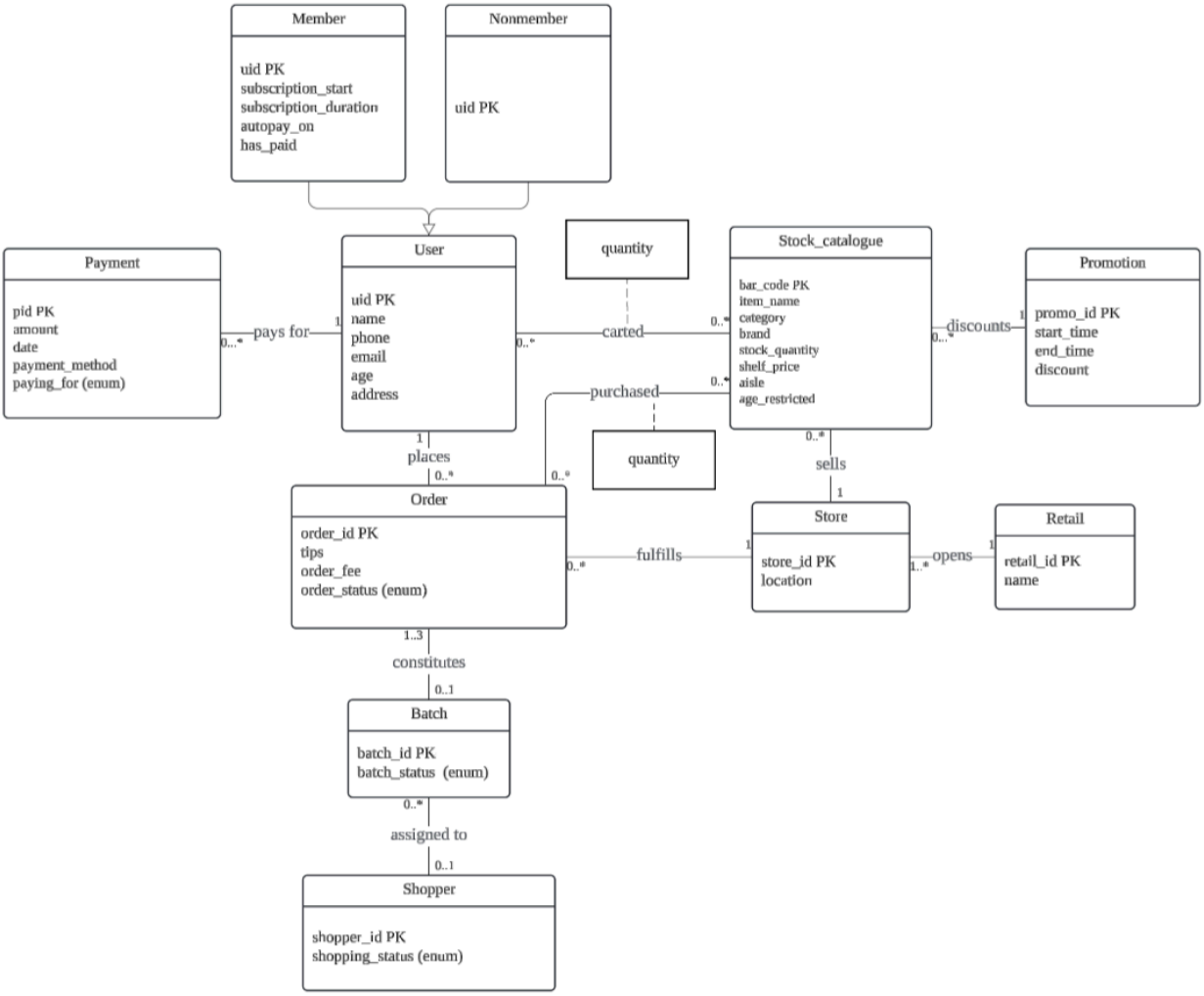
1. Updated List of User Stories

We had no feedback from phase 1. This is the exact same table of user stories as phase 1.

ID	Verb+Noun	As a <role>	I want <goal>	So that <reason>	Simple / Complex	Operational / Analytical
US1	Order results	User	I want to be able to order my search results by price from low to high across different grocery stores	So that I can choose the best store option and spend less	simple	analytical
US2	Add items	User	I want to be able to add items to my shopping cart and specify the quantity	So that I can purchase all of them together when I check out	simple	operational
US3	See orders	Shopper	I want to be able to see order details such as the grocery store, the distance the customer is from the store, and the delivery time	So that I can plan accordingly and bulk several deliveries for one shopping run	simple	analytical
US4	Compare stock	Grocery Store Staff	I want to be able to compare incoming orders with my current item stock	So that I can plan before shoppers arrive and restock items when needed	Complex	analytical
US5	Track	User	I want to track the	So that I can monitor the status of my	simple	operational

	progress		progress of my shopper	order		
US6	Add coupon	Grocery Store Staff	I want to be able to add a coupon to an item, which will reduce that item's price	so that they can attract customers with discounts	Complex	operational
US7	Update delivery status	Shopper	I want to be able to update the delivery status of my assigned orders (e.g., “picked up,” “in transit,” “delivered”)	So that customers can track real-time progress of their grocery deliver	Complex	operational
US8	See popular items	Instacart Manager	I want to be able to see the items that customers purchase for the most	So that I would know what items I should display on the front page	Simple	Analytical
US9	See average time	Instacart Manager	I want to see the average time that it took for shoppers to fulfill assigned batches of 1, 2 or 3 orders	So that I would know when it is better for Instacart to have one shopper shop for multiple orders, and when it is better to have multiple shoppers only shop for one order	Complex	Analytical
US10 **NEW**	Monitor Shoppers	User	I want to see the location of the shoppers who are actively delivering and making deliveries close to me	So that I know delivery coverage in my area right now and can plan accordingly	Complex	Analytical

2. Updated Conceptual Model



We had no feedback from phase 1. The only change we made was that we deleted the associative entity from batch to shopper.

Assumptions

- Batch → Shopper: each batch has exactly one shopper assigned to it, and all orders in that batch come from the same store (same store_id).
- Assignment(relation for now) → Order: one assignment corresponds to exactly one order (1:1).
- Users & membership: a user can be either a Member or a Nonmember at a time; “member” just adds subscription attributes.
- Store & retail: a retail brand operates many stores; Stock_catalogue is the stock information of a specific store location.
- Payment: the enum paying_for specifies whether a user’s payment is for an order or a subscription for a membership. The user can have multiple payments.
- Order:

    - Order status: order_status is an enum that moves through the fulfillment flow. We're thinking states like: unassigned, assigned_to_batch, assigned_to_shopper, shopper_in_progress, delivering, delivered, completed.
    - Order fee: a fee that is nonzero for nonmembers but zero for members.
- Carts vs purchases (just noting future realizations):
    - The "carted" relationship between User and stock_catalogue (store item) will be realized later as a CartItem relation/table with quantity, etc.
    - Similarly, the "purchased" relationship between Order and stock_catalogue will be realized later as an order-line relation/table (e.g., OrderItem) with quantity and price.
- Pricing & promos:
    - Promotions apply to store items during a time window; orders capture whatever discount applies at checkout.

3. Relational Model & 4. Functional Dependencies and Normalization

We chose to implement US7 in the list above, and we converted parts of the conceptual model that are relevant to US7 into the relational model. The user story and the relational model are as follows:

US7	Update delivery status	Shopper	I want to be able to update the delivery status of my assigned orders (e.g., "picked up," "in transit," "delivered")	So that customers can track real-time progress of their grocery deliver	Complex	operational

Order (order_id PK, tips, order_fee, order_status, store_id FK, batch_id FK)

- FD: order_id → tips, order_fee, order_status, store_id
    - Closure: {order_id}+ = {order_id, tips, order_fee, order_status, store_id}
- BCNF

Batch (batch_id PK, batch_status, shopper_id FK)

- FD: batch_id → batch_status, order_id
    - Closure: {batch_id}+ = {batch_id, batch_status, order_id}
- Assumption: One batch will have multiple orders associated with it
- BCNF

Shopper (shopper_id PK, shopping_status)

- FD: shopper_id → shopping_status, batch_id
    - Closure: {shopper_id}+ = {shopper_id, shopping_status, batch_id}
- BCNF

Store (store_id PK, location, retail_id FK)

- FD: store_id -> location, retail_id
    - Closure: {store_id}+ = {store_id, location, retail_id}
- BCNF

Retail (retail_id PK, name)

- FD: retail_id -> name
    - Closure: {retail_id} += {retail_id, name}
- BCNF

5. Physical Model

Below are parts of our physical model that are relevant to US7.