Midterm Project

CS426 - Mobile Device Application Development

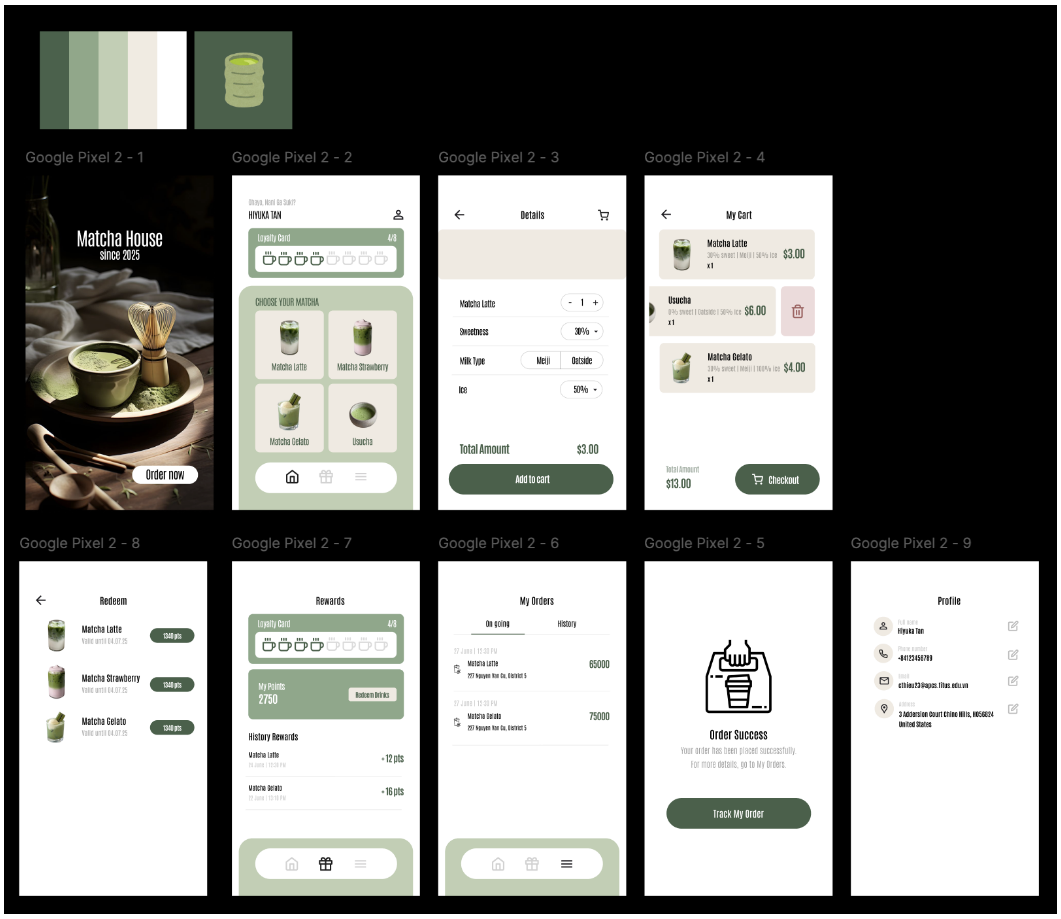
Name: Cao Thanh Hieu
ID Number: 23125034

02 Jul 2025

Abstract

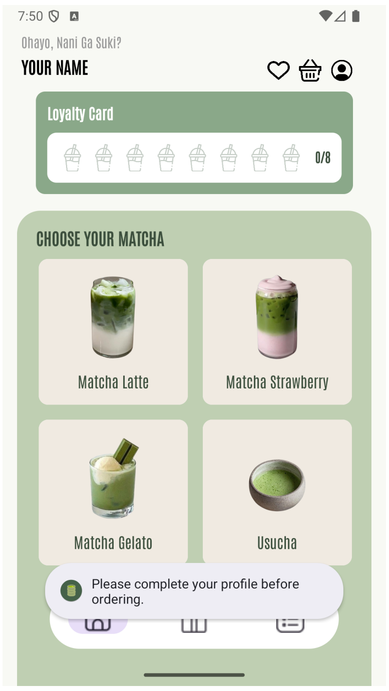
`MatchaHouse` is a multi-screen Android application developed in Kotlin, supported by Room for local data persistence. It models a digital matcha shop, offering a full user experience—from browsing products and customizing orders to tracking purchases and redeeming rewards. The app features a dynamic loyalty system, profile management, and a structured Room database with modular DAOs for clean and reactive data access. This project highlights key Android development practices including UI responsiveness, user state handling, and offline storage.

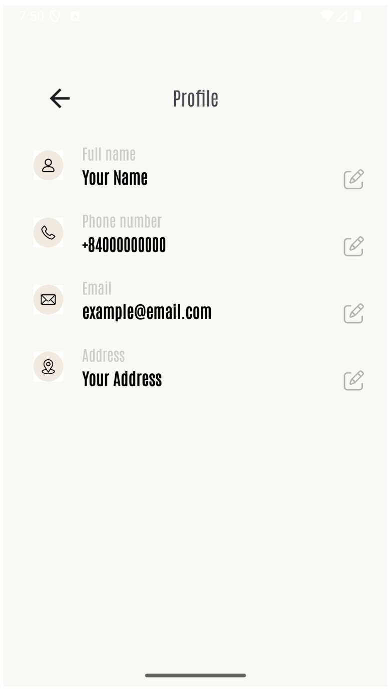
Novel features or requirements beyond the scope of assigment. Additional features include a database for back-end, first-launch profile redirection, dynamic favorite updates, cart item aggregation, reward feedback, and profile editing with progress retention.



Contents

1 Home Screen



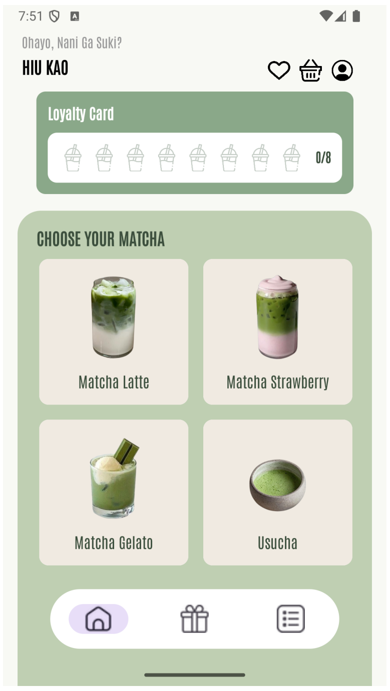


Full name
Your Name

Phone number
+84000000000

Email
example@email.com

Address
Your Address



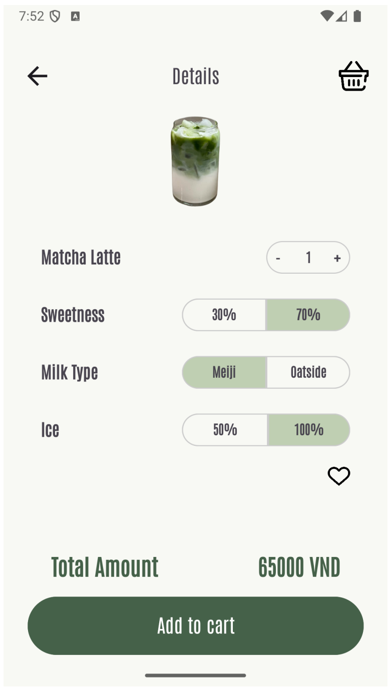
1.1 Assignment Requirements

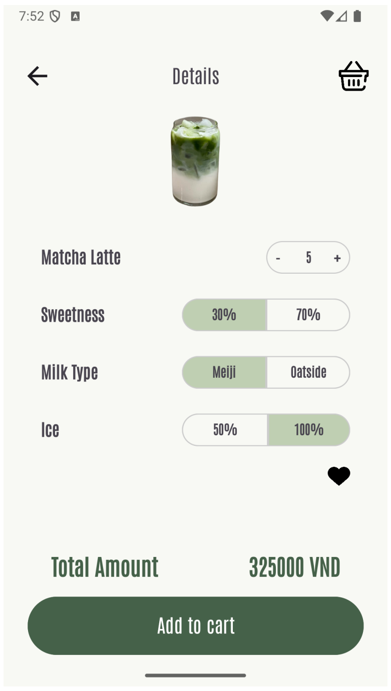
- Implement the fundamental UI layout for the home screen.
- Integrate the specified header element including greetings, user's name and entries for `Favourites`, `Cart` and `Profile`.
- Implement a functional bottom navigation controller.
- Display the user's loyalty card status.
- Populate and render a `Grid Layout` of available matcha products.
- Implement an `on-click` listener for list items that navigates the user to the corresponding product `Details` screen

1.2 Additional Features

- After installation, when the user launches the app for the first time, they are automatically directed to the `Profile` tab to ensure that shipping information is updated before proceeding.
- When clicking back, new information is handled by override `OnResume` function.

2 Details





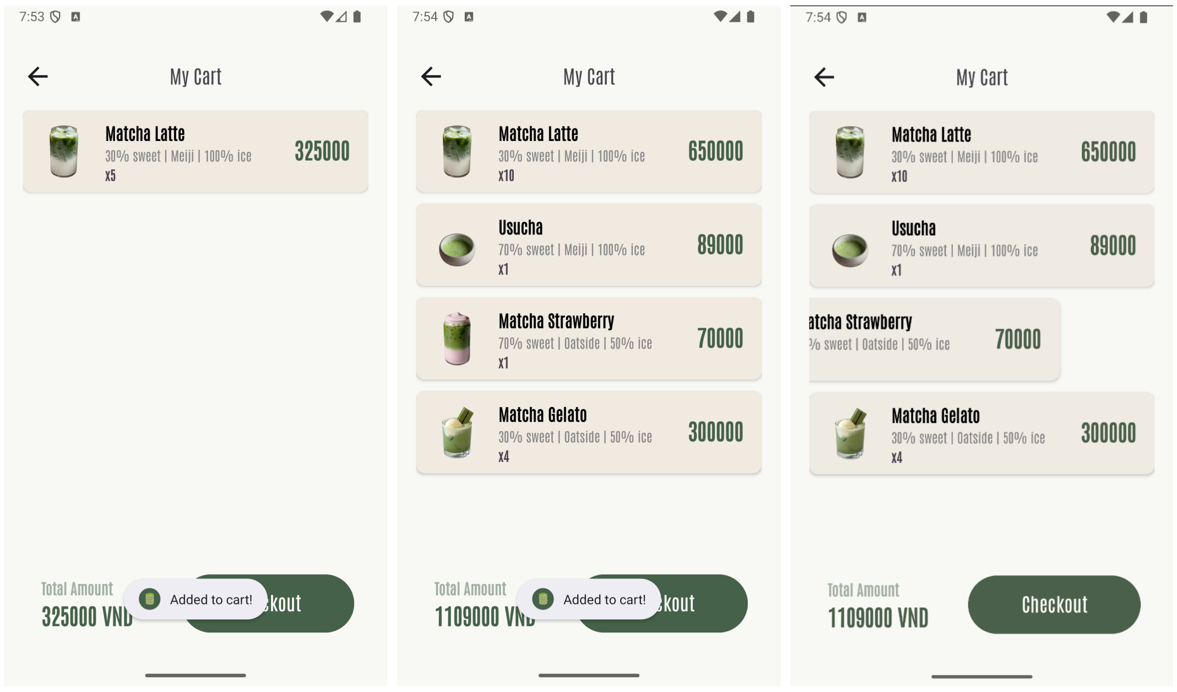
2.1 Assignment Requirements

- Implement UI controls to allow users to customize product options, including `Quantity, Sweetness, Milk Type, Ice`.
- On `Add to Cart` button press, persist the customized item to the cart's state and trigger a navigation event to the `My Cart` screen.
- Implement a `icon click` event to view the current items in the cart without navigating away from the details screen.
- Ensure the total amount dynamically updates in the UI in real-time as the user modifies item quantity or custom options.
- Implement `on-press back` event to return to the Home Screen.

2.2 Additional Features

- An `Add to Favourite` feature is implemented to allow users to save customized drink configurations.
- The favorite icon dynamically updates its state in response to any changes in the product options, ensuring accurate visual feedback based on the current selection.

3 My Cart



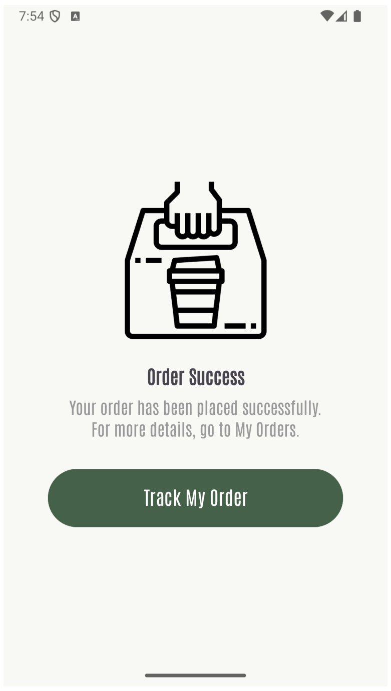
3.1 Assignment Requirements

- Implement the basic layout for the cart screen.
- Display a `RecyclerView` of all matcha items added to cart, reflecting their selected customizations.
- Compute and display the aggregate total price of all items in the cart.
- Implement the gesture detection `on-swipe-left` to remove items from the cart.
- On `Checkout` button press, navigate to the `Order Success` screen.

3.2 Additional Features

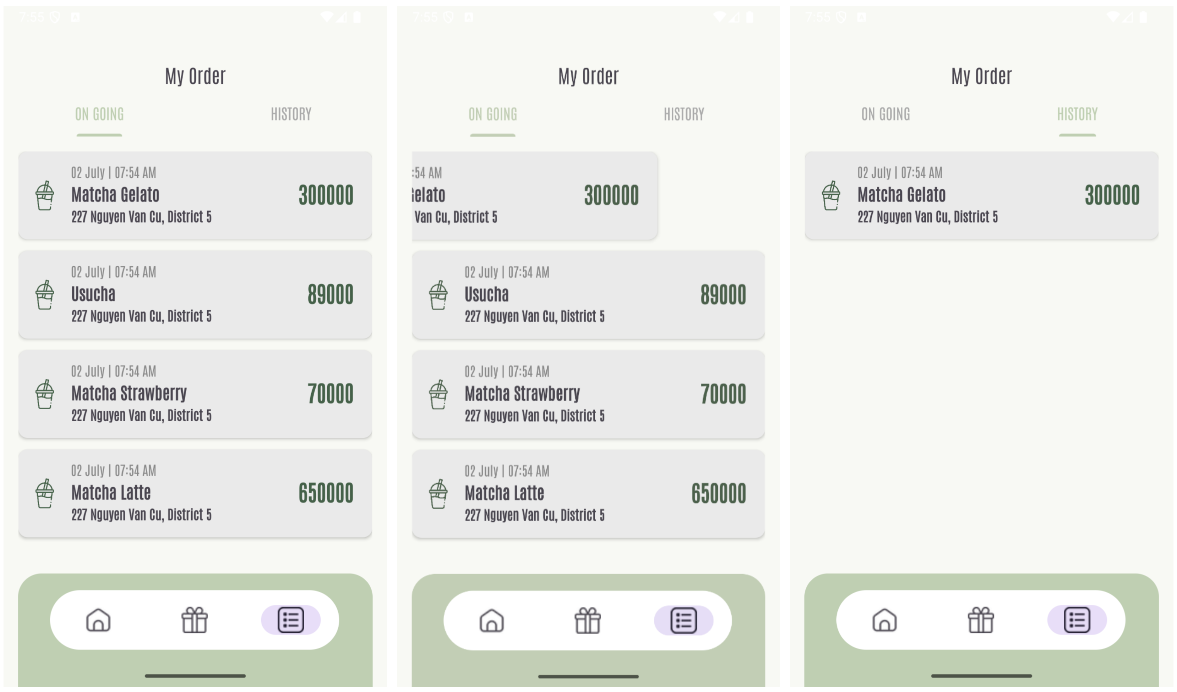
- When a user adds an item with the same product options as an existing item in cart, the application does not create a duplicate entry in the `RecyclerView`. Instead, it increments the quantity of the existing item, effectively aggregating identical selections to maintain a concise and organized cart view.
- A simple log message is displayed to inform the user that an item has been successfully added to the cart.

4 Order Success


7:54
Order Success
Your order has been placed successfully.
For more details, go to My Orders.
Track My Order

- Implement the layout for the order success confirmation screen.
- On `Track My Order` press, navigate to the `My Orders` screen.

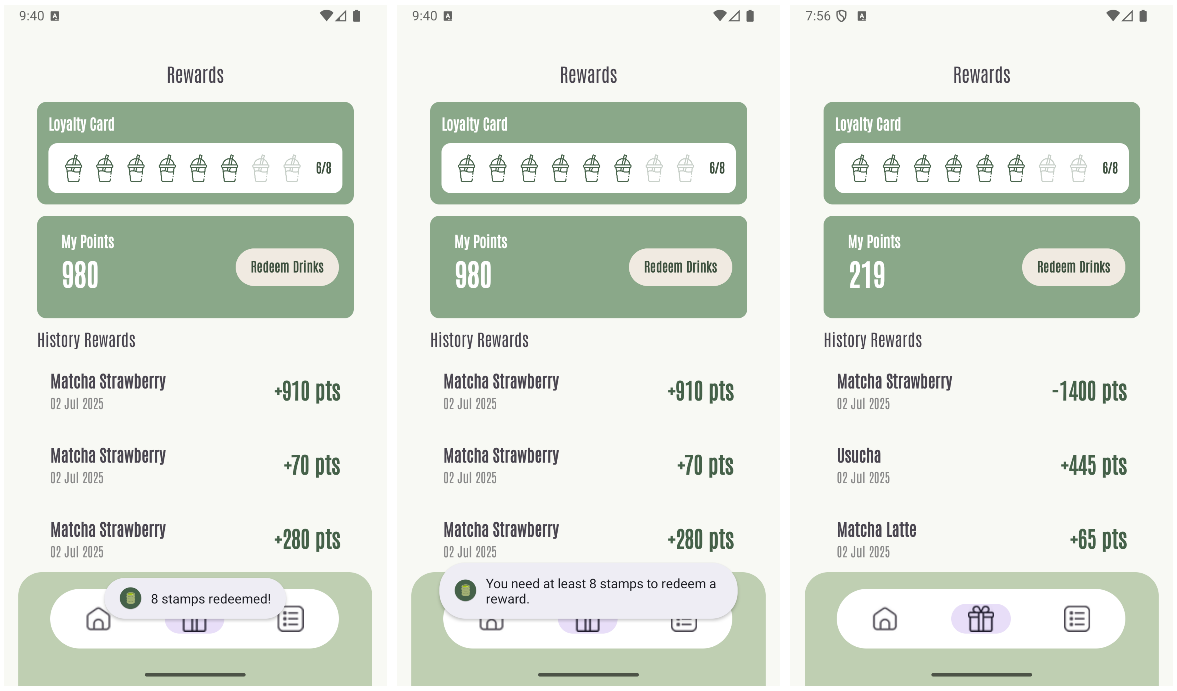
5 My Orders



- Implement the basic layout for the order history screen.

- Render a list segregating ongoing and completed/past orders.
- Implement an event handler `on-swipe` to transition an order's state from `On Going` to `History`.

6 Rewards



Matcha Strawberry 02 Jul 2025	+910 pts
Matcha Strawberry 02 Jul 2025	+70 pts
Matcha Strawberry 02 Jul 2025	+280 pts

8 stamps redeemed!

Matcha Strawberry 02 Jul 2025	+910 pts
Matcha Strawberry 02 Jul 2025	+70 pts
Matcha Strawberry 02 Jul 2025	+280 pts

You need at least 8 stamps to redeem a reward.

Matcha Strawberry 02 Jul 2025	-1400 pts
Usucha 02 Jul 2025	+445 pts
Matcha Latte 02 Jul 2025	+65 pts

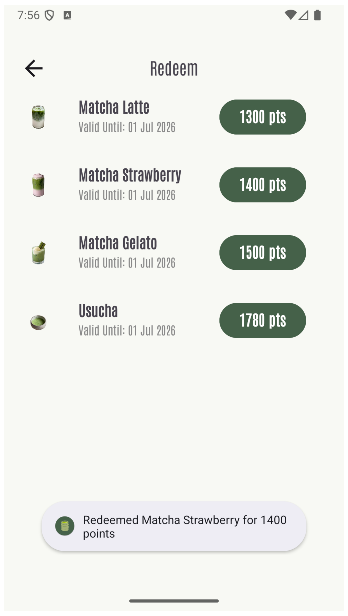
6.1 Assignment Requirements

- Implement the primary layout for the `Rewards` screen.
- For each completed order, increment the loyalty card stamp count by one, up to a maximum of eight.
- Implement an event `on-click` on the white field of `Loyalty Card` subtract stamp count by 8 for each drink redeemed.
- Implement logic to award reward points based on the total monetary value of each order and display them in a list.
- Display the sum of all accumulated reward points.

6.2 Additional Features

- Redeemed points are recorded and displayed in the `History Rewards` section for user reference.
- When the user attempts to exchange stamps for a drink, a notification message is displayed indicating whether the redemption was successful or if the user has insufficient stamps.

7 Redeem Rewards



Matcha Latte
Valid Until: 01 Jul 2026
1300 pts

Matcha Strawberry
Valid Until: 01 Jul 2026
1400 pts

Matcha Gelato
Valid Until: 01 Jul 2026
1500 pts

Usucha
Valid Until: 01 Jul 2026
1780 pts

Redeemed Matcha Strawberry for 1400 points

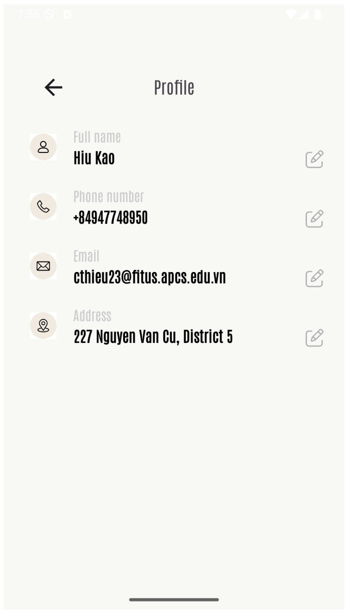
7.1 Assignment Requirements

On a user action, allow the exchange of accumulated points for a product, and correctly decrement the total points.

7.2 Additional Features

A simple log appears after each drink redeem.

8 Profile



Full name
Hiu Kao

Phone number
+84947748950

Email
cthieu23@fitus.apcs.edu.vn

Address
227 Nguyen Van Cu, District 5

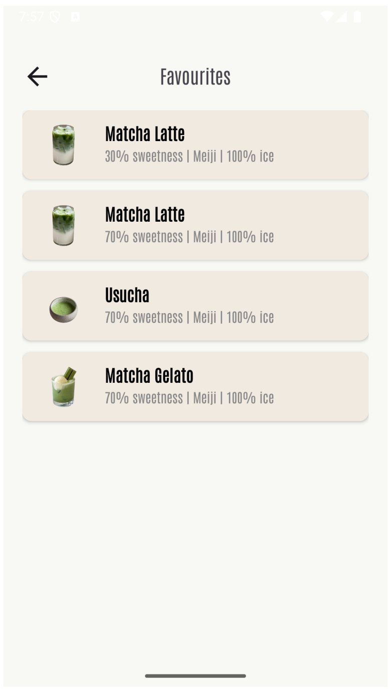
8.1 Assignment Requirements

- Implement the user profile screen layout.
- Enable an edit mode via an icon press, allowing for modification of user profile data

8.2 Additional Features

When editing the user profile, the existing values for `points` and `drinkCount` are preserved to ensure continuity of reward tracking.

9 Favourite (additional features)


Favourites

Matcha Latte
30% sweetness | Meiji | 100% ice

Matcha Latte
70% sweetness | Meiji | 100% ice

Usucha
70% sweetness | Meiji | 100% ice

Matcha Gelato
70% sweetness | Meiji | 100% ice

Implement the gesture detection `on-swipe-left` to remove items from the favourite.

10 Back-end Setting

Room database is designed for this matcha shop application and manages various entities related to user profiles, orders, cart items, favorites, and point tracking. It supports persistent storage, reactive data flows using Kotlin Flows, and prepopulates user profile data on first launch.
The database schema comprises six key entities, each representing a distinct data model within the application.

- `Drink`
  Although not stored in the database, this data class represents predefined static

drink menu items used for UI rendering and business logic.
Fields: `id, name, price, drawableName`.

- `CartItem (cart_items table)`
  Represents individual items added to the user's shopping cart.
  Fields: `id, drinkName, drinkImage, sweetness, milkType, iceLevel, quantity, totalAmount`.

- `OrderItem (orders table)`
  Captures completed and ongoing drink orders along with relevant metadata.
  Fields: `id, drinkName, drinkImage, sweetness, milkType, iceLevel, quantity, totalAmount, address, timestamp, status.`

- `PointsHistory (points_history table)`
  Maintains a log of loyalty points earned by the user across different purchases.
  `Fields: id, drinkName, pointsEarned, timestamp.`

- `FavoriteDrink (favorite_drinks table)`
  Stores user-defined drink customizations marked as favorites. This entity uses a composite primary key to ensure uniqueness based on customization parameters.
  Fields: `drinkName, drinkImage, sweetness, milkType, iceLevel.`

- `UserProfile (user_profile table)`
  Stores personal and account-related information for the user, including drink counts and accumulated points.
  Fields: `id, fullName, phoneNumber, email, address, drinkCount, points.`

For each activity that requires interaction with the database, a corresponding Data Access Object (DAO) is defined to encapsulate the necessary queries and operations.