Midterm Project

CS426 - Mobile Device Application Development

Project Context

You are tasked with building a mobile application for a fictional coffee shop named "The Code Cup". This application allows users to conveniently place orders, collect reward points, and manage their accounts.

Learning Objectives

The learning objectives of this project are to evaluate your ability to:

- Build a complete multi-screen application.
- Implement effective application state management.
- Handle user input and events.
- Implement basic business logic and manage local data.

General Requirements

- This is an individual project.
- Students are permitted to use any framework of their choice, such as the native Android SDK, Flutter, React Native, or others.
- Regardless of the framework used, the application must target the Android platform.
- The project source code must be cleaned (i.e., remove all temporary build files) before submission.

Submission Requirements

A .zip archive named StudentID.zip must be submitted. The archive must adhere to the following directory structure:

StudentID/

├── source/ // Contains all application source code.

├── StudentID-demo.mp4 // A video demonstrating the application's functionality. An optional `demo.txt` file containing a YouTube link is an acceptable alternative.

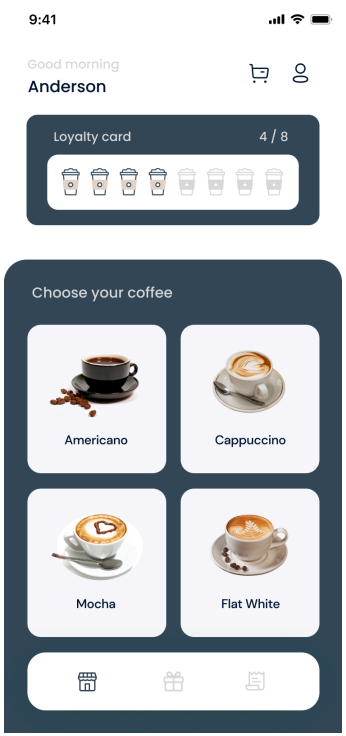
├── StudentID-app.apk // The compiled, installable Android Package Kit (APK) or AAB

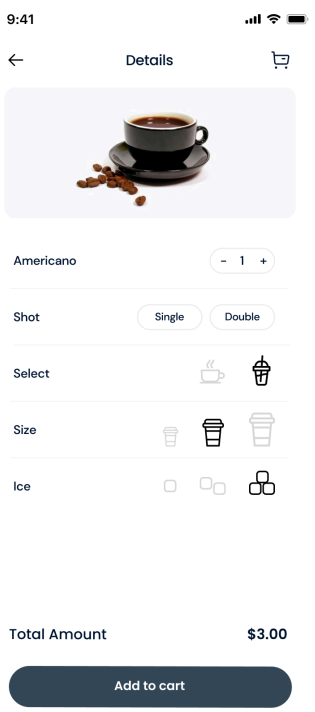
└── StudentID-report.pdf // A comprehensive project report.

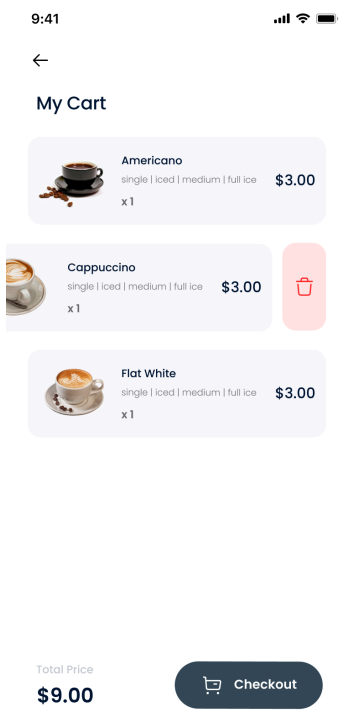
Deadlines: July 5th

Application Requirements

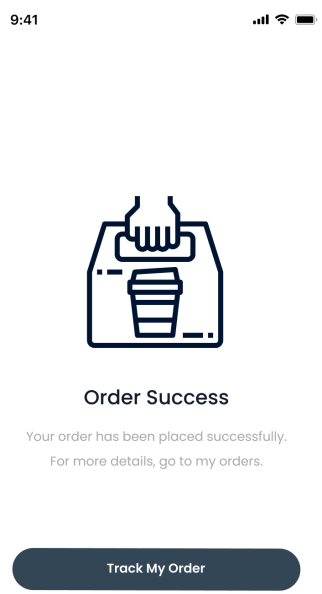
Create an app with the following features

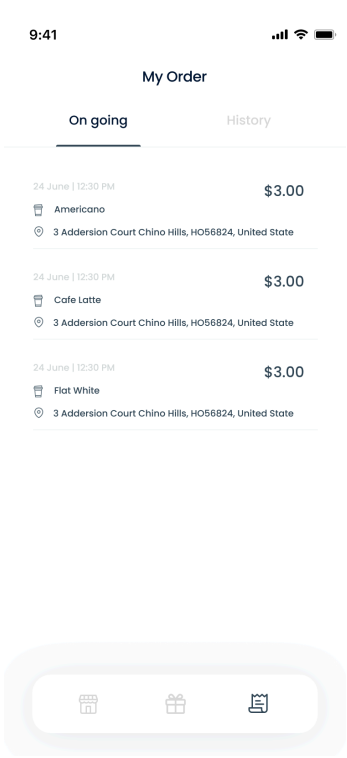
Home Screen		
	**UI Implementation**: Implement the fundamental UI layout for the home screen.	1
	Header Component: Integrate the specified header element.	2
	Bottom Navigation Bar: Implement a functional bottom navigation controller.	3
	Loyalty Card View: Display the user's loyalty card status.	3
	Coffee List View: Populate and render a `ListView` or `RecyclerView` of available coffee products.	3
	Navigation Intent : Implement an `on-click` listener for list items that navigates the user to the corresponding product "**Details**" screen	3

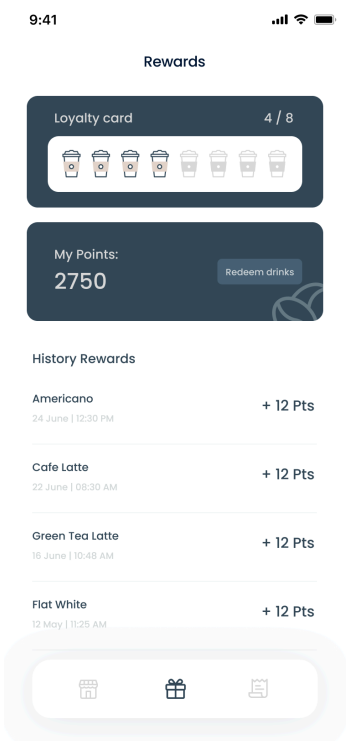
Details		
	**Product Customization Interface**: Implement UI controls to allow users to customize product options (e.g., shot, size, ice).	3
	Add to Cart Functionality: On "**Add to Cart**" button press, persist the customized item to the cart's state and trigger a navigation event to the "**My Cart**" screen.	3
	Cart Preview: Implement a method (e.g., icon click) to view the current items in the cart without navigating away from the details screen	3
	Dynamic Price Calculation: Ensure the total amount dynamically updates in the UI in real-time as the user modifies item quantity or custom options.	3
	Back Navigation: Implement a `Maps back` or `on-press back` event to return to the **Home Screen**.	1
My Cart		

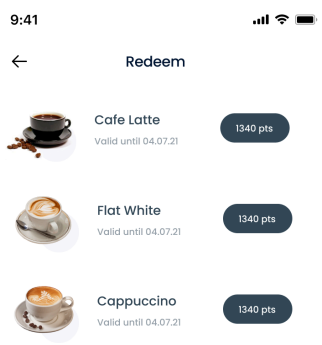


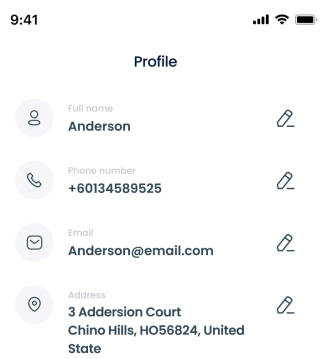
UI Implementation: Implement the basic layout for the cart screen	1
Cart Item Rendering: Display a `ListView` or `RecyclerView` of all coffee items added to the cart, reflecting their selected customizations.	7
Total Price Display Compute and display the aggregate total price of all items in the cart.	3
Gesture-Based Item Removal: Implement gesture detection (e.g., `on-swipe-left`) to remove items from the cart.	3
Checkout Navigation: On "**Checkout**" button press, navigate to the "**Order Success**" screen.	1

Order Success		
	**Order Success UI**: Implement the layout for the order success confirmation screen.	1
	Track Order Navigation: On "Track My Order" press, navigate to the "My Orders" screen.	1

My Orders		
	**My Orders UI**: Implement the basic layout for the order history screen.	1
	Order History Display: Render a list segregating ongoing and completed/past orders.	3
	Order Status Transition: Implement an event handler (e.g., on-click, on-swipe) to transition an order's state from "ongoing" to "history".	3

Rewards		
	**Rewards Screen UI:** Implement the primary layout for the Rewards screen.	1
	Loyalty Stamp Logic: For each completed order, increment the loyalty card stamp count by one, up to a maximum of eight.	3
	Loyalty Card Reset: Implement an event (e.g., on-click) to reset the stamp count to zero upon reaching eight stamps.	3
	Points Calculation & Display: Implement logic to award reward points based on the total monetary value of each order and display them in a list.	3
	Total Points Aggregation: Display the sum of all accumulated reward points.	2

Redeem Rewards		
	**Points Redemption**: On a user action, allow the exchange of accumulated points for a product, and correctly decrement the total points.	3

Profile		
	**UI Implementation**: Implement the user profile screen layout.	3
	Profile Editing Functionality: Enable an edit mode via an icon press, allowing for modification of user profile data.	3

General application requirements		
	State & Lifecycle Management: Implement robust state management to handle the application lifecycle (e.g., `onPause`, `onResume`, `onStop`) and preserve data across configuration changes.	12
	Data Persistence & Initialization: Implement a data persistence strategy (e.g., SharedPreferences, Room, SQLite) and handle the initial seeding of required application data.	6
	User-Defined Features: Implement novel features or requirements beyond the scope of this document.	**50**
Total	**Min(150, Total score of all requirements above)**	**150**
Additional Criteria	**Code Quality, Report & Demo (±10 Points):** The final score may be adjusted by up to 10 points based on the quality, clarity, and organization of the source code, the project report, and the video demonstration.	**[-15,+15]**
Your score: Min(10,(Total + Additional Criteria) / 15)		

Regulations and Support

Academic Integrity

- Plagiarism or direct copying of code from other students or online sources is strictly prohibited. All submitted code must be the student's own original work.
- The use of third-party libraries is permitted but must be properly attributed in the report.
- Any violation of these policies will result in a score of zero for the project and may be reported for further disciplinary action.

Use of AI Tools

- You are permitted to use AI-powered tools (e.g., GitHub Copilot, ChatGPT) to assist you.
- However, you are fully responsible for the code you submit. You must ensure that you thoroughly understand everything you add to your project.

- The instructor may ask you to explain how any part of your source code works during the final review. You must be able to explain the logic and functionality of the code you have submitted.

Support Channel

- All questions regarding the project requirements should be posted on the official class Facebook group for assistance.