

Professional Training Report

Submitted in partial fulfillment of the
requirements for the award of
Bachelor of Technology degree in Information Technology

by

ARCHITH SB(41120026)



DEPARTMENT OF INFORMATION TECHNOLOGY

SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Category-1 University by UGC

Accredited with Grade “A++” by NAAC

12B Status by UGC |Approved by AICTE

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

APRIL - 2024



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited "A++" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF INFORMATION TECHNOLOGY

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **ARCHITH SB(41120026)** who carried out the project entitled **"BEVERAGES ORDERING APPLICATION"** under my supervision from **January 2024 to May 2024**.

Internal Guide

Dr.P. Jeyanthi, M.E,Ph.D.,

Head of the Department

Dr.R. Subhashini, M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **Archith SB(41120026)** hereby declare that the Project Report entitled “**Beverages Ordering Application**” done by me under the guidance of **Dr.P.Jeyanthi, M.E., Ph.D.**, is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology degree in **Information Technology**.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

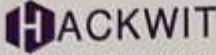
I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E.,Ph.D., Dean**, School of Computing **Dr.R.Subhashini M.E.,Ph.D.**, Head of the Department of Information Technology for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.P.Jeyanthi,M.E.,Ph.D.**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

CERTIFICATE OF COMPLETION

**HACKWIT**
HACKWIT TECHNOLOGIES PVT LTD
Redefine by Technologies

149, 1C, 2nd Floor, Mount Poonamallee Rd,
Ramapuram, Chennai - 600089.
admin@hackwitechnologies.com
www.hackwitechnologies.com

10.02.2024

CERTIFICATE FOR INTERNSHIP

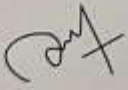
TO WHOMSOEVER IT MAY CONCERN

This is to Certify that **Archith SB (Reg No: 41120026)** student of Sathyabama Institute of Science and Technology, B. Tech Information Technology, has successfully completed the Internship at Hackwit Technologies from 22nd January to 30th January 2024.

During the internship, **Archith SB** has closely worked as a part of the **Flutter** by carrying out the project entitled "**Tea Shop App**".

He demonstrated good developing and learning skill with self-motivated attitude to learn new things. His contribution to implement the Flutter was good. We wish him all the best for his future endeavors.

Best Regards,



Ravikumar S
CEO & Managing Director
Hackwit Technologies Pvt Ltd



CERTIFICATE OF COMPLETION



ABSTRACT

Flutter Beverages is a mobile application designed using the Flutter framework to provide a seamless experience for both customers and coffee shop owners. This application aims to revolutionize the way customers interact with their favorite coffee shops by offering features such as menu browsing, ordering, payment processing, loyalty programs, and real-time order tracking. For customers, Flutter Bevarages offers a user-friendly interface where they can explore menus, customize orders, place orders for pickup or delivery, and conveniently pay through various payment methods. The application also incorporates a loyalty program to reward frequent customers and encourage retention. For coffee shop owners, FlutterBeverages provides a robust backend system to manage menus, track orders, analyze sales data, and communicate with customers. The application streamlines order processing, reduces wait times, and enhances overall efficiency for coffee shop operations. Utilizing the Flutter framework ensures cross-platform compatibility, allowing FlutterBeverage to run seamlessly on both iOS and Android devices. With its intuitive user interface, advanced features, and scalability, FlutterBeverage sets a new standard for coffee shop applications, enhancing the coffee shop experience for both customers and owners alike.

TABLE OF CONTENTS

Chapter No.	TITLE	Page No.
	ABSTRACT	vii
	LIST OF ABBREVIATIONS	x
	LIST OF FIGURES	xi
1	INTRODUCTION	1
	1.1 Flutter	2
	1.2 Dart	3
	1.3 Problem Statement	4
	1.4 Objectives	7
2	METHODOLOGY	9
	2.1 Initial Project Requirements	9
	2.2 Proposed System Architecture	12
	2.3 Flow Diagram	15
	2.4 Modules	17
	2.4.1 User Management	17
	2.4.2 Home Page	18
	2.4.3 Category Page	19
	2.4.4 Cart Page	20

	2.5	Development and SetUp	21
3		RESULTS AND DISCUSSION	23
4		CONCLUSION AND FUTURE WORK	29
	4.1	Conclusion	29
	4.2	Future Work	30
		REFERENCES	32
		APPENDICES	34
	A.	Sample Code	34
	B.	Screen Shots	37

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
UI	- User Interface
UX	- User Experience
iOS	- iPhone Operating System
IDE	- Integrated Development Environment

LIST OF FIGURES

Figure No.	Name of the Figure	Page No.
2.1	Proposed Application Architecture	12
2.2	Flow Diagram	15
3.1	User Management	17
3.2	Home Page	18
3.3	Category Page	19
3.4	Cart Page	20
3.5	Develop And Run App	21

CHAPTER 1

INTRODUCTION

In today's fast-paced world, the demand for convenience and efficiency in our daily routines is ever-growing. One area where this demand is particularly pronounced is in the realm of coffee consumption. Whether it's a quick caffeine fix on the way to work or a leisurely weekend coffee date, people expect seamless experiences when indulging in their favorite brews.

To meet this demand and elevate the coffee shop experience for both customers and owners, we introduce FlutterBeverages – a cutting-edge mobile application developed using the Flutter framework. FlutterBeverages reimagines the traditional coffee shop experience by harnessing the power of technology to streamline operations, enhance customer engagement, and drive business growth.

This introduction sets the stage for exploring how FlutterBeverages revolutionizes the coffee shop industry by seamlessly integrating innovative features and functionalities. From browsing menus to placing orders and managing loyalty programs, FlutterBBeverages offers a comprehensive solution that caters to the evolving needs of modern coffee enthusiasts and savvy business owners alike. Let's delve deeper into the world of FlutterBeverages and discover how it's reshaping the coffee shop landscape, one cup at a time.

1.1. FLUTTER

Flutter, the powerful cross-platform UI toolkit developed by Google, serves as the cornerstone of the Coffee Shop application, shaping its foundation and user interface with unparalleled efficiency and versatility. By harnessing the capabilities of Flutter, Coffee Shop transcends traditional app development approaches, offering a unified codebase that seamlessly adapts to the unique requirements of both iOS and Android platforms. This transformative technology enables Coffee shop to deliver a visually stunning and responsive user interface, ensuring an immersive shopping experience for users across all devices.

The utilization of Flutter revolutionizes the development process of Coffee Shop, streamlining workflows and eliminating the complexities associated with managing separate codebases for different platforms. Developers benefit from a cohesive development environment, characterized by accelerated iteration cycles and heightened productivity. Flutter's hot reload feature further enhances the development experience, allowing for real-time code changes and instant visual feedback, thus facilitating rapid iterations and optimizations.

One of the foremost advantages of Flutter lies in its ability to maintain consistency and uphold a high standard of user experience across a diverse range of devices. By leveraging Flutter's extensive collection of customizable widgets, Coffee Shop achieves a cohesive design language and fluid user interactions, regardless of the underlying operating system. This ensures that users can seamlessly transition between devices while enjoying a consistently superior shopping experience, marked by exceptional performance and aesthetics.

1.2. DART

i) Unified Codebase:

Beverages Ordering Dart, a versatile programming language, for implementing both frontend (Flutter) and backend logic. Dart's unified approach allows developers to maintain a single codebase for the entire application, facilitating consistency and seamless collaboration between different aspects of the app's development.

ii) Asynchronous Programming:

Dart's support for asynchronous programming plays a crucial role in Coffee shop backend operations. This feature enables the app to handle concurrent tasks efficiently, such as processing user requests, managing database operations, and interfacing with external APIs. Asynchronous programming ensures that Coffee shop remains responsive and performs optimally even when dealing with multiple simultaneous operations.

iii) Scalability:

The scalability of Dart is essential for accommodating Coffee shop potential growth and increasing user demand. As the user base expands and the volume of grocery orders rises, Dart's scalability capabilities enable the app to handle higher traffic levels without sacrificing performance.

iv) Strong Typing System:

Dart's strong typing system enhances code reliability and maintainability within Coffee shop. By enforcing strict type checking, Dart helps identify potential errors during the development phase, thereby reducing the likelihood of runtime issues.

1.3. PROBLEM STATEMENT

In today's fast-paced world, coffee lovers often find themselves facing long queues and limited options at their favorite coffee shops. Additionally, coffee shop owners struggle to efficiently manage orders, inventory, and customer preferences while providing a seamless experience. To address these challenges, we aim to develop a mobile application using Flutter that streamlines the ordering process for customers and enhances operational efficiency for coffee shop owners, ultimately revolutionizing the coffee shop experience. "Inefficient and time-consuming coffee shop experiences for both customers and owners due to long queues, limited options, and operational challenges."

The solution: Develop a Flutter-based mobile app that streamlines the ordering process, enhances customer experience, and improves operational efficiency for coffee shop owners. The coffee shop app aims to address the inefficiencies and challenges faced by both customers and owners in traditional coffee shop experiences. By leveraging Flutter, the app will streamline ordering processes, enhance customer satisfaction, and improve operational efficiency for coffee shop owners, ultimately revolutionizing the coffee shop experience.

The coffee shop app aims to address the inefficiencies and challenges faced by both customers and owners in traditional coffee shop experiences. By leveraging Flutter, the app will streamline ordering processes, enhance customer satisfaction, and improve operational efficiency for coffee shop owners, ultimately revolutionizing the coffee shop experience.

complete transactions. Incorporating location-based services, loyalty rewards, and social integration, the app will enhance user experience. The coffee shop app, built using Flutter, seeks to revolutionize the traditional coffee shop experience by addressing the inefficiencies faced by both customers and owners. Customers often encounter long queues and limited options, leading to frustration and time wastage. Meanwhile, owners struggle with managing orders, inventory, and customer preferences efficiently. Through the app, customers will enjoy a seamless ordering process, personalized recommendations, and the convenience of pre-ordering. Owners will benefit from improved operational efficiency, real-time inventory management, and insights into customer preferences, ultimately leading to a more satisfying and streamlined coffee shop experience for all stakeholders.

Our project aims to develop a feature-rich coffee shop app using Flutter, designed to provide users with a seamless and efficient ordering experience. Through intuitive navigation and customizable options, customers can effortlessly browse menus, personalize their orders, and securely engage and satisfaction. With a focus on accessibility and scalability, we aim to create a versatile platform that caters to diverse user needs and enables future expansions and enhancements.

In response to the growing demand for convenient coffee ordering, our project aims to develop a Flutter-based mobile application tailored specifically for coffee enthusiasts. The app will streamline the ordering process, offering users a user-friendly interface to browse through a comprehensive menu, customize their beverages, and place orders seamlessly.

Additionally, the app will integrate features like order tracking, enabling users to monitor the status of their orders in real-time. Loyalty rewards and personalized recommendations will further enhance user engagement and satisfaction. By harnessing the power of Flutter, we ensure a cross-platform solution that delivers a consistent experience across both Android and iOS devices, catering to a broader audience of coffee lovers.

Developing a beverage ordering app using Flutter presents an exciting challenge. The app aims to streamline the coffee ordering process, enhance customer experience, and boost sales for the business. The problem statement encompasses various aspects, including user interface design, functionality implementation, backend integration for order management, payment processing, and real-time communication between customers and the coffee shop staff.

Additionally, considerations for scalability, performance optimization, and cross-platform compatibility are vital to ensure a seamless experience across different devices. Through meticulous planning, efficient coding, and iterative testing, the goal is to deliver a robust, user-friendly coffee shop app that meets the needs of both customers and the business.

1.4. OBJECTIVES

The objectives for the coffee shop application using Flutter include enhancing customer experience through features like easy menu browsing and order placement, streamlining operations for coffee shop owners, increasing customer engagement through loyalty programs and promotions, ensuring cross-platform compatibility, scalability, and flexibility, prioritizing data security and privacy, ensuring performance and reliability, and representing the brand identity effectively. By achieving these objectives, the application aims to provide a seamless and engaging experience for customers while optimizing efficiency and driving growth for coffee shop owners.

Key Challenges:

- i) **User Interface Design:** Designing an intuitive and visually appealing user interface that facilitates easy navigation and enhances the overall user experience.
- ii) **Real-time Updates:** Implementing real-time updates for order status, menu changes, and promotions to keep customers informed and engaged.
- iii) **Backend Integration:** Integrating backend services for order management, inventory tracking, and payment processing to ensure seamless transactions and efficient operations.
- iv) **Cross-platform Compatibility:** Ensuring that the app functions consistently across various platforms and devices, including smartphones, tablets, and web browsers.

- v) **Performance Optimization:** Optimizing app performance to minimize load times, reduce battery consumption, and provide a smooth user experience, especially during peak usage period Maintenance and Updates.
- vi) **Security:** Implementing robust security measures to protect sensitive customer data, payment information, and transactions from unauthorized access or breaches.
- vii) **Localization:** Supporting multiple languages and currencies to cater to a diverse customer base and facilitate international transactions.
- viii) **Accessibility:** Ensuring that the app is accessible to users with disabilities by adhering to accessibility guidelines and providing alternative input methods and screen reader support.

CHAPTER 2

METHODOLOGY

2.1. INITIAL PROJECT REQUIREMENTS

The initial project requirements for Coffee Shop encompass a range of functionalities and technologies aimed at creating a seamless and user-friendly grocery shopping experience.

i)Project Scope:

- 1) Develop a mobile application named "Coffee shop" focused on enhancing the scope shopping experience for users.
- 2) Create a platform that offers a comprehensive solution for browsing, selecting, and purchasing coffee,as well as facilitating community engagement and collaboration.
- 3) Address the diverse needs and preferences of users, catering to various dietary restrictions, cultural considerations, and shopping habits.

ii)Technologies:

- 1) Flutter: Utilize Flutter for cross-platform mobile app development, ensuring compatibility and consistency across iOS and Android devices.
- 2) Dart: Employ Dart as the primary programming language for developing the Coffee Shop application, leveraging its capabilities for efficient app development.

iii) Functionalities:

- 1) User Authentication: Implement a secure user authentication system to allow users to create accounts, log in securely, and access personalized features.
- 2) Product Browsing and Selection: Provide users with a seamless experience for browsing, searching, and selecting coffee items from a diverse range of categories.
- 3) Coffee Lists: Enable users to create, manage, and share coffee lists, ensuring efficient planning and organization of coffee trips.
- 4) Personalized Recommendations: Develop algorithms to offer personalized product recommendations based on users' preferences, past purchases, and dietary restrictions.
- 5) Community Engagement: Implement features that foster community engagement, such as recipe sharing, discussion forums, and user-generated content.

iv) Packages Used:

i) get:

Version: 4.6.6

Purpose: State management and navigation framework.

ii) provider:

Version: 6.1.6

Purpose: State management package for Flutter applications, offering a simple and intuitive way to manage application state and perform dependency injection.

iii) http:

Version: 0.13.3

Purpose: HTTP client for Flutter applications, allowing users to make network requests to fetch data from APIs or interact with web servers.

iv) flutter_svg:

Version: 0.23.0

Purpose: Enabling users to display vector images .

v) shared_preferences:

Version: 2.0.8

Purpose: Package for storing key-value pair persistently.

vi) google_sign_in:

Version: 5.2.1

Purpose: Google Sign-In authentication.

vii) toast:

Version: 0.3.0

Purpose: Displaying simple toast notifications.

viii) fluttertoast:

Version: 8.0.7

Purpose: Displaying customizable toast messages.

ix) image_picker:

Version: 1.0.7

Purpose: Picking images from the gallery or taking photos.

2.2. PROPOSED APPLICATION ARCHITECTURE

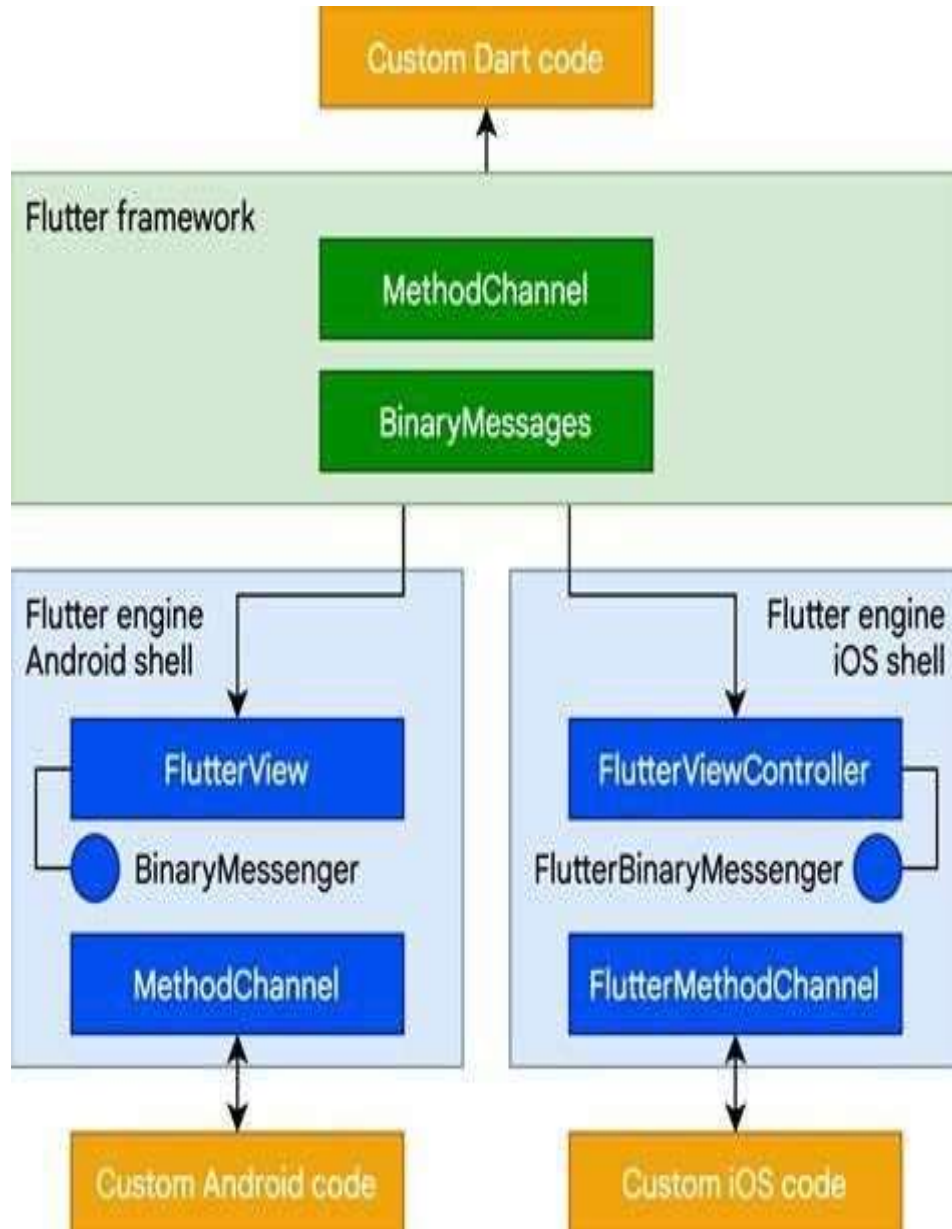


Fig:2.1: Proposed Application Architecture

1. Custom Dart Code:

This represents any custom code you write in Dart specifically for your Flutter app. It could be business logic, UI components, or any other functionality unique to your application. The custom Dart code interacts with the Flutter framework to achieve various tasks.

2. Flutter Framework:

The Flutter framework is the heart of your app. It provides a rich set of widgets, tools, and APIs for building cross-platform mobile applications.

3. Method Channel:

The Method Channel allows communication between Dart code and native platform code (Android or iOS). When we invoke a method on the Method Channel from Dart, it triggers a corresponding method call on the native side.

4. Binary Messages:

Binary Messages is another channel for communication between Dart and native code. It enables the exchange of binary data (such as images, files, or serialized objects) between the two sides.

5. Flutter Engine Android Shell:

On the Android side, the Flutter engine runs within an Android shell.

- i) Flutter View: This is the Android view that displays your Flutter UI.
- ii) Binary Messenger: It handles communication between the Flutter engine and the Android platform.

6. Flutter Engine iOS Shell:

On iOS, the Flutter engine runs within an iOS shell.

- i) Flutter View Controller: The iOS view controller responsible for displaying your Flutter UI.
- ii) Flutter Binary Messenger: Handles communication between the Flutter engine and the iOS platform.

7. Custom Android Code and Custom iOS Code:

These represent platform-specific code written in Java/Kotlin (for Android) and Objective-C/Swift (for iOS). You will write custom code here to interact with native features or APIs that aren't directly accessible from Flutter.

2.3. FLOW DIAGRAM

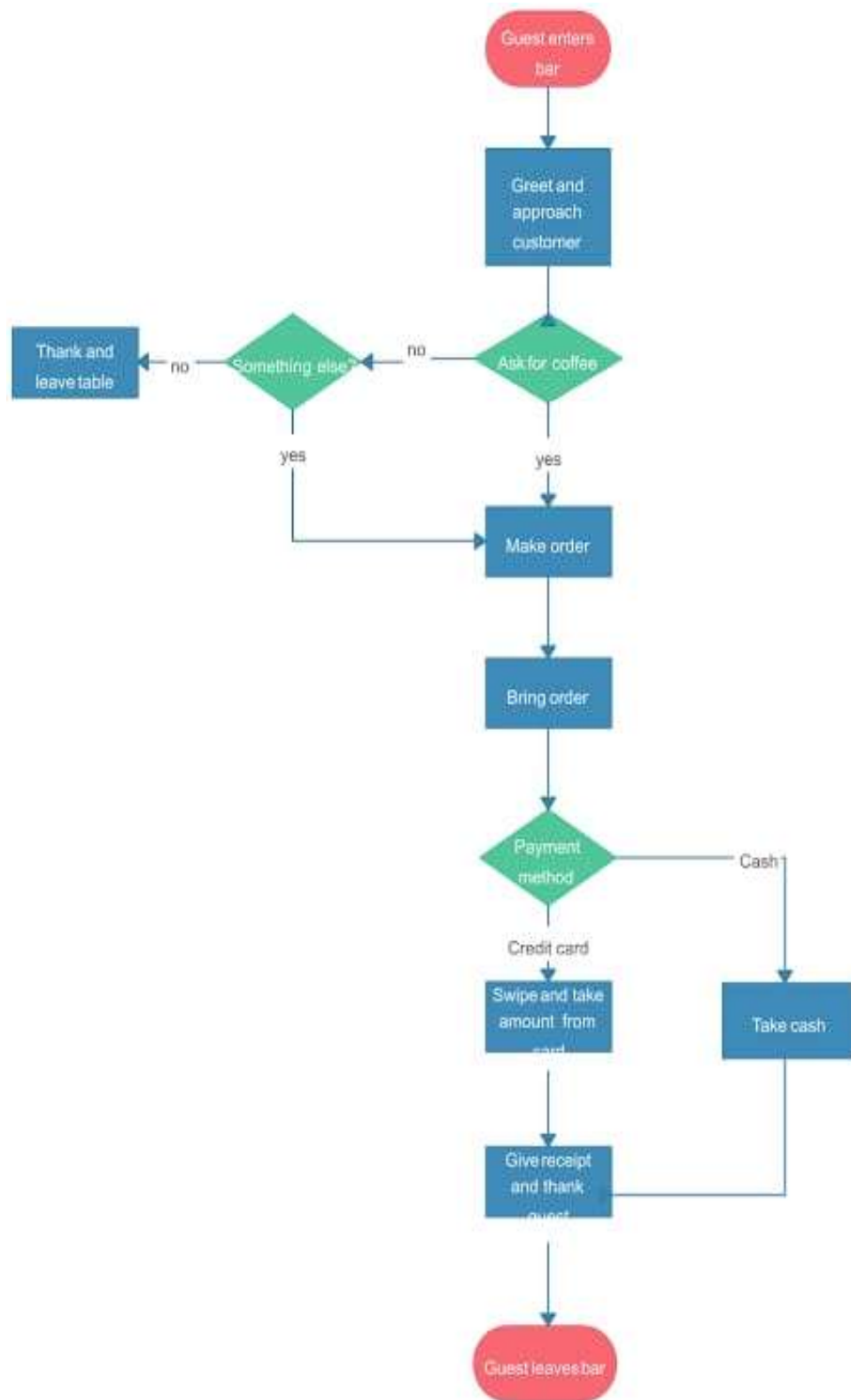


Fig:2.3: Flow Diagram

The following steps are used to describe the above flow diagram:

- i) The user opens the mobile app.
- ii) The app prompts the user to log in or sign up if they don't have an account yet.
- iii) If the user chooses to sign up, they will be taken to the sign-up page.
- iv) If the user is already registered, they can log in directly.
- v) Once the user is logged in, they will be taken to the home page.
- vi) From the home page, the user can access various features of the app such as their home page, cart page, category page, and user profile page.
- vii) The home page allows the user to search and select coffee products/items they want to purchase .
- viii) The category page contains a variety of coffees.
- ix) The cart page contains checkout process.

2.4. MODULES

2.4.1. USER MANAGEMENT:

Provides a comprehensive registration form(signup) with fields for personal information, contact details, and interests, enhancing user profile completeness.

Algorithm:

i) Display Registration Form: The registration process commences with the display of a user-friendly signup form within the coffee shop application. The form encompasses fields for essential user information, including personal details, contact information, and preferences, facilitating the creation of a comprehensive user profile.

ii) Collect User Information: Upon accessing the registration form, users are prompted to input necessary details such as their full name, email address, password, and preferences related to grocery shopping. These inputs are vital for establishing personalized profiles tailored to each user's preferences and needs.

iii) Validate Input Data: As users input their information, the system conducts validation checks to ensure the accuracy and integrity of the provided data.

iv) Create User Account: Following successful validation, the system proceeds to create a new user account utilizing the submitted information. Each user account is assigned a unique identifier or username within the coffee shop platform, enabling efficient user management and personalized experiences for users.

2.4.2. HOME PAGE:

Upon launching the coffee shop application, users are greeted with an intuitive and visually appealing home page designed to provide easy access to essential features and functionalities.

Algorithm:

- i) Display Home Page: The application's home page is prominently presented upon launch, featuring a clean and organized layout that prioritizes usability and user engagement. Users are greeted with a visually appealing interface that invites exploration and interaction.

- ii) Showcase Featured Items: The home page prominently displays featured grocery items, special promotions, or recommended products, enticing users to explore and discover new offerings. Featured items are showcased in visually appealing sections, accompanied by attractive images and relevant information to capture user interest.

- iii) Search Functionality: A prominent search bar is integrated into the home page interface, enabling users to quickly search for specific grocery items or browse through product categories. The search functionality is intuitive and responsive, providing real-time suggestions and facilitating efficient product discovery.

- iv) Display Personalized Recommendations: Leveraging user preferences and past purchase history, the home page presents personalized product recommendations tailored to each user's preferences and shopping habits. These recommendations are prominently displayed, guiding users towards relevant products and enhancing the shopping experience.

2.4.3. CATEGORY PAGE:

The Categories page within the Brew beverage ordering app application offers users a comprehensive overview of available product categories, enabling seamless navigation and exploration of dairy items.

Algorithm:

i) Display Categories Overview: Upon accessing the Categories page, users are presented with a comprehensive overview of available product categories, organized in a visually appealing and easily navigable layout. Categories are displayed in an intuitive grid or list format, providing users with a clear and structured view of available options.

ii) Categorize Products: Each product category is meticulously curated to encompass a diverse range of coffee items. Categories are logically grouped based on product type or usage, facilitating efficient navigation and exploration for users.

iii) Provide Visual Representation: Categories are accompanied by visually engaging icons or images that represent the respective product types, enhancing the overall user experience and aiding in quick recognition and selection. Visual cues provide users with a glimpse of the products available within each category, enticing them to explore further.

iv) Facilitate Wishlist Management: In addition to browsing and selecting products, users can add desired items to their wish list directly from the Categories page. A dedicated wish list icon or button is provided for each product, allowing users to easily add items to their wish list for future reference or purchase.

2.4.4. CART PAGE:

The Cart page within the coffee application provides users with a centralized view of their selected items, facilitating convenient management and checkout of items.

Algorithm:

i) Display Cart Contents: Upon accessing the Cart page, users are presented with a comprehensive overview of the items added to their shopping cart. The page displays product thumbnails, names, quantities, prices, and subtotal amounts, providing users with detailed information about their selected items.

ii) Enable Quantity Adjustment: Users can easily adjust the quantity of individual items within their shopping cart directly from the Cart page. Intuitive increment and decrement buttons or input fields allow users to increase or decrease the quantity of selected items as needed, ensuring accurate order customization.

iii) Facilitate Item Removal: In addition to quantity adjustment, users can remove unwanted items from their shopping cart with ease. Each item in the cart is accompanied by a dedicated remove button or option, enabling users to swiftly delete items they no longer wish to purchase.

iv) Display Total Cost: Users can easily view the total cost of their order and make informed decisions before proceeding to checkout.

v) Provide Checkout Options: Users are presented with convenient checkout options directly from the Cart page, allowing them to proceed to checkout with minimal effort. Prominent checkout buttons or links direct users to the checkout process, streamlining the transition from browsing to purchase.

2.5. DEVELOPMENT SETUP

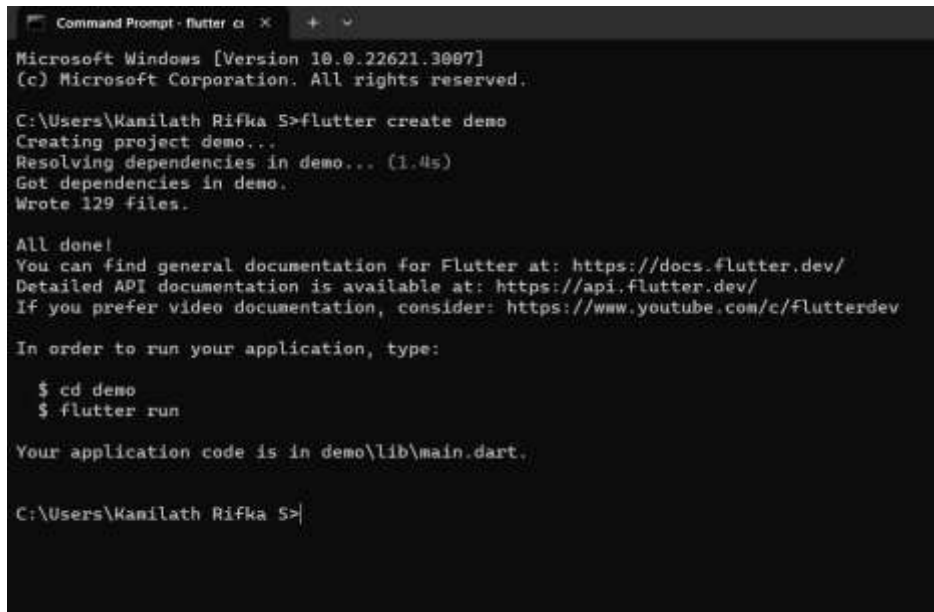
1. Development Environment:

Flutter and Dart need to be installed on the developer's machine.

IDEs like Visual Studio Code or Android Studio for coding.

2. Create Project:

>>flutter create project_name

A screenshot of a Windows Command Prompt window titled "Command Prompt - flutter cs". The window shows the output of the command "flutter create demo". The text in the prompt is as follows:

```
Microsoft Windows [Version 10.0.22621.3087]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Kamilath Rifka S>flutter create demo
Creating project demo...
Resolving dependencies in demo... (1.4s)
Got dependencies in demo.
Wrote 129 files.

All done!
You can find general documentation for Flutter at: https://docs.flutter.dev/
Detailed API documentation is available at: https://api.flutter.dev/
If you prefer video documentation, consider: https://www.youtube.com/c/flutterdev

In order to run your application, type:

$ cd demo
$ flutter run

Your application code is in demo\lib\main.dart.

C:\Users\Kamilath Rifka S>
```

Fig:2.4: Project Creation

3. Dependency Management:

Project dependencies specified in the pubspec.yaml file.

Key packages include:

- flutter_toast for displaying certain notifications.
- shared_preferences used for key-value storage mechanism.
- provider is a state management package.
- Other packages like get, toast, image_picker, etc., for additional functionalities.

4. Development Tools:

Flutter commands used for testing and building the application. Real-time testing on Android and iOS devices.

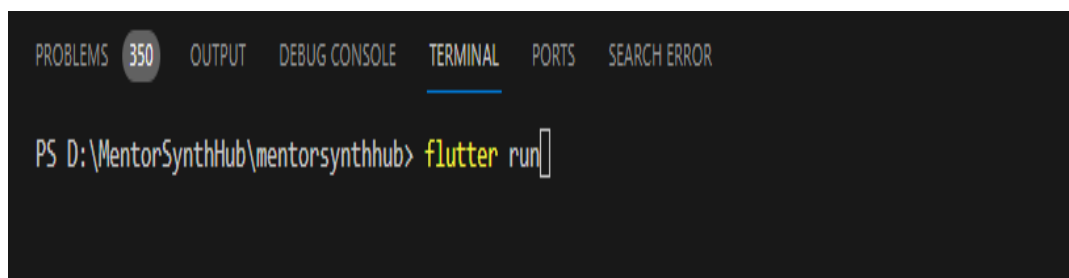


Fig:2.5: Flutter Run

5. Supported Platforms:

Cross-platform support for Android, iOS, and web.

6. Integrated Development Environment (IDE):

Developers can choose between Visual Studio Code or Android Studio for coding.

CHAPTER 3

RESULTS AND DISCUSSION

DART CODES:

i) `main.dart`: This file is typically the entry point of a Flutter application. It initializes the app, sets up necessary configurations, and runs the main application widget.

ii) `welcome.dart`: This file serves as the welcome screen of the Flutter application, often being the first point of interaction for users. It initializes the welcome screen, sets up necessary UI configurations, and renders the main welcome widget.

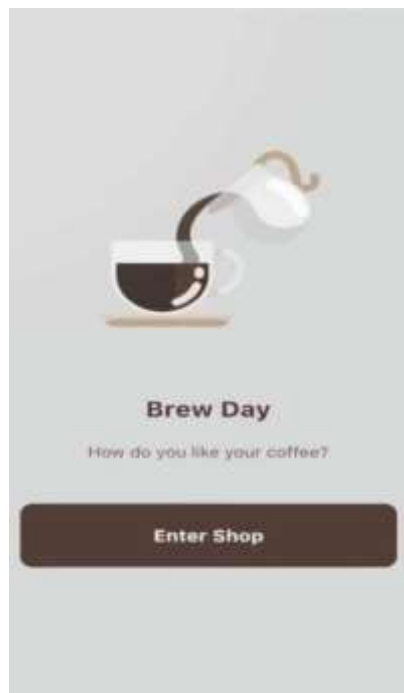


Fig:3.1 Welcome Page Interface

iii) `coffee_featured_item.dart`: This file defines the widget responsible for displaying featured grocery items within the BrewBuddy application. Sets up necessary configurations such as image loading and text styling, and renders the featured item widget.

iv) `Home_page.dart`: This file represents the main content of your app after a successful login or signup. It contains search dairy products ,category section and discount layout with various features.

v) `Home_banner_widget.dart`: This file defines the widget responsible for displaying banners or promotional content on the home page of the BrewBuddy application. It initializes the layout and appearance of the banner widget.

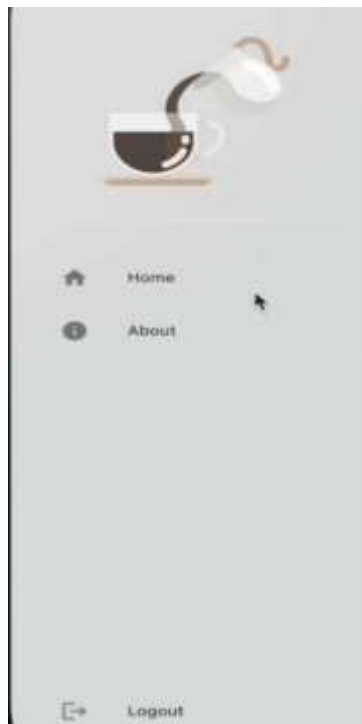


Fig:3.2 Home Screen Display

vi) category_item_screen.dart: This file defines the screen for displaying items within a specific category in the beverage ordering application. The category_item_screen.dart file may include features such as filtering options, sorting mechanisms, and interactive elements for users to navigate through items within the category.

vii) explore_screen.dart: This file defines the screen for exploring various categories and products within the beverage ordering app application. The explore_screen.dart file may include features such as search functionality, category navigation, and personalized recommendations based on user preferences.



Fig:3.3 Variety of coffee 1

viii) `account_screen.dart`: This file defines the screen for managing user accounts within the `coffeeshopapplication`. The `account_screen.dart` file may include features such as user authentication options, password reset functionality, and integration with external services for account management.

ix) `account_item.dart`: This file defines the widget responsible for displaying individual items within the user account on the `account_screen.dart`. The `account_item.dart` file may include features such as clickable elements for users to view detailed information about each item, edit options for updating account details, and integration with backend services to fetch and display dynamic account data.



Iced tea.png



Black.png

Fig:3.4 Variety of coffee 2

x) `variety_screen.dart`: This file defines the screen for applying filters and refining search results within the Brew Buddy application. It initializes the layout and appearance of the filter screen, sets up necessary configurations such as displaying filter options and applying selected filters, and renders interactive elements for users to customize their search criteria. The `filter_screen.dart` file may include features such as filter categories, price range selection, and dynamic filtering based on user input.

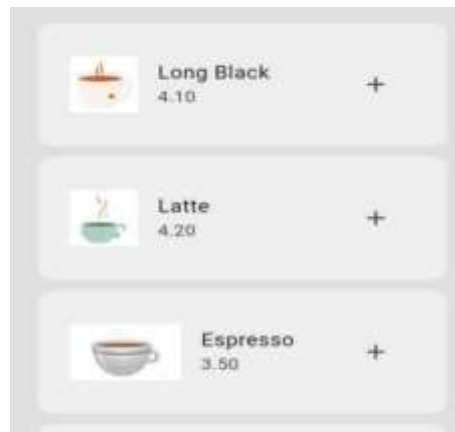


Fig:3.5 Filter Screen

xi) `cart_screen.dart`: This file defines the screen for managing the shopping cart within the Brew Buddy application. It initializes the layout and appearance of the cart screen, sets up necessary configurations such as displaying items added to the cart, calculating total costs, and rendering interactive elements for users to manage their cart contents. The `cart_screen.dart` file may include features such as quantity adjustment options, item removal functionality, and integration with backend services for updating cart data.



Fig:3.6 Cart Screen

CHAPTER 4

CONCLUSION AND FUTURE WORK

4.1. CONCLUSION

In conclusion, developing a coffee shop application using Flutter offers a seamless user experience across multiple platforms, thanks to its robust framework and UI toolkit. By leveraging Flutter's hot reload feature and native performance, we've created a responsive and visually appealing app that enhances the customer's interaction with our coffee shop. With features like easy ordering, loyalty rewards, and real-time updates, our Flutter-based coffee shop app provides convenience and value to both customers and the business. As we continue to iterate and enhance the app, we look forward to further improving the coffee shop experience for our users."

The coffee shop application developed using Flutter delivers a seamless user experience across various platforms. Its responsive design, coupled with features like easy ordering, loyalty rewards, and real-time updates, enhances both customer engagement and business efficiency. Leveraging Flutter's native performance and hot reload feature, the app ensures a visually appealing interface while offering convenience and value to users. Moving forward, continuous iteration promises to further refine and elevate the coffee shop experience."

4.2. FUTURE WORK

1.Enhanced User Profiles:

Introduce more detailed product profiles with additional information such as nutritional facts, ingredients, and user reviews, offering users a comprehensive understanding of each item.

2.Enhanced Recommendation Engine:

Develop a robust recommendation engine leveraging machine learning algorithms to provide personalized product suggestions based on user preferences, purchase history, and browsing behavior.

3.Integrated Recipe Suggestions:

Integrate recipe suggestions into the app based on the items in users' carts or wish list, offering inspiration and guidance for coffee planning and preparation.

4.Augmented Reality (AR) Features:

Explore the implementation of augmented reality features that allow users to visualize how products will look or fit in their home environment before making a purchase, enhancing the online shopping experience.

5.Community Recipe Sharing:

Enable users to share their favorite recipes within the app's community, fostering a collaborative environment where users can discover new meal ideas and culinary inspirations

6.Subscription Services Integration:

Partner with subscription services or meal kit providers to offer seamless integration within the app, allowing users to subscribe to

curated grocery bundles or meal plans directly from beverage ordering app.

7.Social Shopping Features:

Introduce social shopping features that enable users to create and share shopping lists with friends or family members, facilitating collaborative shopping experiences and group coordination.

8.Localized Deals and Promotions:

Implement location-based deals and promotions tailored to users' geographic locations, providing personalized savings opportunities and incentives for in-store or online purchases.

9.Advanced Inventory Management:

Enhance the app's inventory management capabilities for businesses and retailers, offering tools for real-time stock tracking, automated replenishment, and inventory optimization.

10.Continuous User Feedback Integration:

Establish a feedback loop with users to gather insights and suggestions for further improvements, ensuring that future updates and features align with user needs and preferences.

REFERENCES

1. Journals:

- P. Tyagi, “Building layouts,” Pragmatic Flutter, pp. 87–126, Jun.2021. doi:10.1201/9781003104636-7
- P. Tyagi, “Setting up environment,” Pragmatic Flutter, pp. 23–34, Jun. 2021. doi:10.1201/9781003104636-3
- R. Payne, “Using firebase with flutter,” Beginning App Development with Flutter, pp. 255–285, 2019. doi:10.1007/978-1-4842-5181-2_1

2. Books:

- A. Miola, Flutter Complete Reference 2.0: The Ultimate Reference for Dart and Flutter. Independently published, 2023.
- J. H. Iversen and M. A. Eierman, Learning Mobile App Development: A Hands-on Guide to Building Apps with IOS and Android. Upper Saddle River, New Jersey: Addison-Wesley, 2014
- A. K. S., Mastering Firebase for Android Development: BuildReal-Time, Scalable, and Cloud-Enabled Android Apps with Firebase. Birmingham, UK: Packt Publishing, Ltd., 2018

3. Internet Articles:

- D. Camp, "Best flutter blog articles from around the web,"Flutter articles,(accessed Apr. 25, 2024).
- Google,"Firebasedocumentation,"FlutterDocumentations" (accessed Apr. 25, 2024).
- W. Wilson, "The ultimate list of flutter resources , "FlutterResources,"<https://wilsonwilson.dev/articles/flutter-resources> (accessed Feb. 20, 2024).

APPENDICES

A.SAMPLE CODE

1. Main Code: Dart page

```
Flutter_application_2 > lib > % overridden %  
1 import 'package:flutter/material.dart';  
2 import 'package:flutter_application_2/models/coffee.dart';  
3 import 'package:flutter_application_2/models/coffee_shop.dart';  
4 import 'package:provider/provider.dart';  
5 import 'pages/home_page.dart';  
6  
7 void main() {  
8   runApp(const MyApp());  
9 }  
10  
11 class MyApp extends StatelessWidget {  
12   const MyApp({super.key});  
13  
14   @override  
15   Widget build(BuildContext context) {  
16     return ChangeNotifierProvider(  
17       create: (context) => CoffeeShop(),  
18       builder: (context, child) => const MaterialApp(  
19         debugShowCheckedStackBanner: false,  
20         home: HomePage(),  
21       ), // MaterialApp  
22     ), // ChangeNotifierProvider  
23   );  
24 }
```

2,Home_screen.dart:

```
Flutter_application_2 > lib > pages > % home_page.dart  
1 import 'package:flutter/material.dart';  
2 import 'package:flutter_application_2/components/bottom_bar.dart';  
3 import 'package:flutter_application_2/const.dart';  
4 import 'cart_page.dart';  
5 import 'shop_page.dart';  
6  
7 class HomePage extends StatefulWidget {  
8   const HomePage({super.key});  
9  
10  @override  
11  State<HomePage> createState() => _HomePageState();  
12 }  
13  
14 class _HomePageState extends State<HomePage> {  
15   // navigate bottom bar  
16   int _selectedIndex = 0;  
17   void navigateBottomBar(int index) {  
18     setState(() {  
19       _selectedIndex = index;  
20     });  
21   }  
22  
23   // pages  
24   final List<Widget> _pages = [;  
25     // shop  
26     ShopPage(),  
27     // cart  
28     CartPage(),  
29   ];  
30  
31  @override  
32  Widget build(BuildContext context) {  
33    return Scaffold(  
34      backgroundColor: backgroundColor,  
35      bottomNavigationBar: MyBottomNavBar(  
36        onPageChange: (index) => navigateBottomBar(index),  
37      ),  
38    );  
39  }  
40 }
```

3.shop_page.dart:

```
lib > models > coffee_shop.dart > CoffeeShop > _shop
1  import 'package:flutter/material.dart';
2  import 'coffee.dart';
3
4  class CoffeeShop extends ChangeNotifier{
5    // coffee for sale list
6    final List<Coffee> _shop = [
7      // black coffee
8      Coffee(
9        name: 'Long Black',
10       price: "4.10",
11       imagePath: "lib/images/black.png",
12     ),
13
14     // latte
15     Coffee(
16       name: 'latte',
17       price: "4.20",
18       imagePath: "lib/images/latte.png",
19     ),
20
21     // espresso
22     Coffee(
23       name: 'Espresso',
24       price: "3.50",
25       imagePath: "lib/images/espresso.png",
26     ),
27
28     // iced coffee
29     Coffee(
```

4.Car_Page.dart:

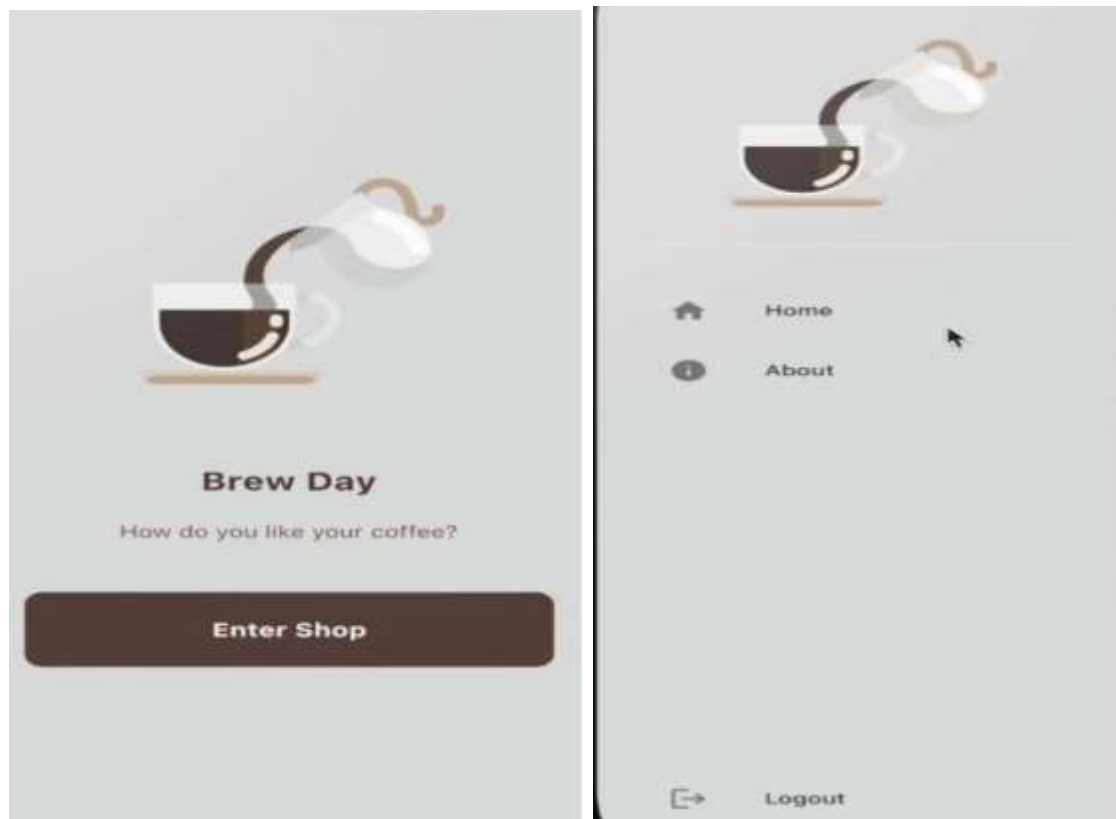
```
shop_page.dart  cart_page.dart  main.dart  bottom_nav_bar.dart  coffee_shop.dart
lib > pages > cart_page.dart > _CartPageState > build
3  import 'package:flutter_application_2/models/coffee.dart';
4  import 'package:flutter_application_2/models/coffee_shop.dart';
5  import 'package:provider/provider.dart';
6
7  class CartPage extends StatefulWidget {
8    const CartPage({super.key});
9
10   @override
11   State<CartPage> createState() => _CartPageState();
12 }
13
14 class _CartPageState extends State<CartPage> {
15
16   // remove item from cart
17   void removeFromCart(Coffee coffee) {
18     Provider.of<CoffeeShop>(context, listen: false).removeItemFromCart(coffee);
19   }
20
21   // pay button tapped
22   void payNow() {
23     /*
24     fill out your payment service here
25     */
26   }
27 }
28
29
```

5.Coffee_Page.dart:

```
lib > models > coffee.dart > Coffee
1  class Coffee {
2    final String name;
3    final String price;
4    final String imagePath;
5
6    Coffee({
7      required this.name,
8      required this.price,
9      required this.imagePath,
10   });
11 }
```

B. SCREEN SHOTS

i) Welcome Screen and Homepage:



ii) Category page and cart page

