FINAL REPORT

Foody – A Food Delivery Solution

Harsh Agheda
202001461

Guided By :

Prof. Jayprakash Lalchandani

2. Table of Contents

3. Abstract:

The "Foody" Food Delivery App is a comprehensive mobile application designed to streamline the process of ordering food from a variety of restaurants. This app offers a user-friendly interface that allows users to easily register, browse restaurant menus, place orders, and manage their delivery addresses. The objective of this project is to provide a convenient and efficient way for users to satisfy their food cravings while enjoying a seamless ordering experience.

4. List of Abbreviations and Acronyms:

- UI: User Interface
- API: Application Programming Interface
- SDK: Software Development Kit
- GPS: Global Positioning System
- UX: User Experience

5. Introduction

In today's digital age, the food delivery industry has witnessed a significant transformation, driven by advancements in technology and changing consumer preferences. This project presents a comprehensive solution for streamlining and enhancing the food ordering and delivery experience through a multi-app ecosystem comprising the User App, Rider App, and Restaurant App.

6. Motivation

The motivation behind the development of this multi-app system stems from the increasing demand for convenient and efficient food delivery services which also provide advanced order with options to order food for a certain duration of dates. With the rapid growth of online food ordering platforms, there exists a need for a robust and interconnected ecosystem that can seamlessly connect users, riders, and restaurants. The primary motivations for this project are:

1. *Enhancing User Experience*: To provide users with a user-friendly and efficient platform for ordering their favorite meals from a variety of restaurants.

2. *Optimizing Delivery Operations:* To empower riders with a dedicated app that helps them efficiently manage and fulfill food delivery orders.

3. *Empowering Restaurants*: To offer restaurants a tailored platform for receiving and managing orders, improving their reach to potential customers.

The overarching objectives and goals of this project are to create a cohesive multi-app ecosystem that simplifies the food ordering and delivery process. Specific objectives include:

1. User App:

- Enable users to easily browse restaurant menus.
- Facilitate secure and convenient online payment options.
- Provide real-time order tracking and status updates.
- Offer a user-friendly and intuitive interface.

2. Rider App:

- Empower riders to efficiently manage and deliver food orders.
- Optimize delivery routes for time and cost efficiency.
- Facilitate real-time communication between riders and users.
- Ensure the safety and security of riders during deliveries.

3. Restaurant App:

- Equip restaurants with a digital platform for receiving and managing orders.
- Enable menu customization and item availability updates.
- Streamline order preparation and processing.
- Enhance restaurant visibility to potential customers.

By achieving these objectives, this project aims to create a seamless and mutually beneficial ecosystem for users, riders, and restaurants, thereby revolutionizing the food delivery industry and catering to the evolving needs of today's consumers.

7. Problem Statement

The food ordering and delivery industry faces an evolving challenge: catering to the diverse lifestyles and preferences of modern consumers. Traditional on-demand food delivery services offer immediate gratification, but they fall short in accommodating users who plan their meals in advance or wish to schedule orders for specific dates and times. The absence of a platform tailored to this need often leaves a significant gap in the market, resulting in user frustration and missed opportunities for both restaurants and delivery personnel.

8. Purpose of the App

The primary objective of this multi-app system is to bridge this gap by introducing a groundbreaking feature that allows users to order food for specific durations and dates with unmatched convenience. The purpose of this feature-rich system is to:

1. *Empower Users*: Provide users with the ability to plan their meals according to their schedules, dietary preferences, and occasions. Whether it's a week-long family vacation, a special anniversary dinner, or just the convenience of setting up recurring daily orders, this app ensures that users have full control over when and what they eat.

2. *Revolutionize Restaurant Services*: Offer restaurants a platform to showcase their menus and enable advanced ordering options. This empowers restaurants to cater to a broader audience, better manage their resources, and optimize food preparation to meet scheduled orders efficiently.

3. *Enhance Delivery Efficiency*: Enable delivery personnel to manage their schedules more effectively. With pre-scheduled deliveries and route optimization, riders can make the most of their time, enuring timely and accurate order deliveries.

4. *Cater to Occasions*: Recognize the importance of special occasions in users' lives. The system allows users to place orders for birthdays, holidays, and other memorable events in advance, ensuring a hassle-free and delightful celebration.

9. Target Audience

This multi-app system caters to a diverse target audience:

- Users: Individuals and families seeking a flexible and convenient way to plan and schedule their food orders for various durations, whether it's a one-time meal or a week-long subscription.

- Restaurants: A wide range of eateries, from local cafes to high-end restaurants, eager to embrace a platform that helps them reach users who plan their dining experiences in advance.

- Riders: Independent or employed delivery personnel who can benefit from more predictable and organized delivery schedules.

By addressing these needs and empowering users to plan their food orders in advance for specific dates and durations, this innovative app system aims to redefine the food delivery industry, making it more user-centric and accommodating of varied lifestyles and preferences.

10. Development Process and Methodology

The development of our multi-app system, consisting of user, rider, and restaurant apps, followed an agile software development methodology. This methodology allowed us to adapt to changing requirements, maintain open communication with stakeholders, and ensure a rapid and iterative development process. Here's an overview of the development process:

1. *Project Inception* : We initiated the project with a comprehensive analysis of the problem statement and user requirements. This phase involved brainstorming sessions to identify key features and prioritize development tasks.

2. *Design Phase* : We created paper prototypes for each app to visualize the user interface and user experience design. This design-centric approach ensured that the apps are intuitive and user-friendly.

3. *Development* : The development phase involved writing code for each app. We used modular code architecture to ensure flexibility and easy maintenance. Each app

was developed concurrently but independently to streamline the development process.

11. Technologies and Tools

The development of our multi-app system leveraged a stack of modern technologies and tools to deliver robust, scalable, and user-friendly applications. Here's an overview of the key technologies and tools used:

- Programming Languages:

- *Dart* : Used for developing the mobile apps, as it's the primary language for Flutter.

- Frameworks and Libraries:

- *Flutter*: Utilized for building cross-platform user, rider, and restaurant apps. Flutter's single-codebase approach ensured consistency across platforms (iOS and Android).
- *Firebase*: Leveraged for backend-as-a-service (BaaS) capabilities, including authentication, real-time database, cloud functions, and storage.

- Database:

- *Firestore*: Chosen as the database to store user data, order information, and restaurant details. Firestore's real-time capabilities ensured seamless synchronization of data.

- Version Control:

- *Git*: Employed for version control to facilitate collaborative development and code management.

- Development Environment:

- *Android Studio*: Used as the integrated development environment (IDE) for Flutter app development.

- Third-party Services:

- *Google Maps API*: Integrated for location-based services.
- *Geolocator API*: Integrated for location-based services.

By adopting a modern tech stack and development methodology, we ensured that our multi-app system met the highest standards in terms of performance, security, and user experience. The combination of Flutter's cross-platform capabilities and Firebase's backend services allowed us to deliver a seamless and feature-rich solution to our users, riders, and restaurant partners.

12. Design and Architecture

Our multi-app system, which includes user, rider, and restaurant apps, follows a well-structured and scalable architecture that ensures flexibility and maintainability. Here, we'll present the architecture and design principles of the system, along with some illustrative diagrams.

System Architecture

The architecture of our multi-app system is based on a client-server model. Each app serves as a client, interacting with a centralized server that manages core functionalities, including user authentication, order processing, and data storage. Here's an overview of the system architecture:

1. Client Apps:

   - *User App*: Designed for end-users who want to browse restaurants, place orders, and track deliveries.

   - *Rider App*: Tailored for delivery personnel responsible for accepting delivery requests and ensuring timely delivery.

   - *Restaurant App*: Created for restaurant owners or staff to manage incoming orders and update menu items.

2. Server: The server component is responsible for handling requests from client apps and managing the core logic of the system. It interacts with the database and external services as needed.

3. Database:

   - *Firestore*: Used as the primary NoSQL database for storing user profiles, restaurant information, menu items, order history, and real-time order tracking.

Key Design Principles:

1. *Modularization*: Each app is designed with a modular approach, ensuring that different components (UI, data access, business logic) are separated and can be developed and maintained independently.

2. *Cross-Platform Compatibility*: We've employed Flutter as the framework for app development, allowing us to create a single codebase that can be extended to run on both iOS and ndroid devices. This approach saves development time and ensures consistent user experiences across platforms.

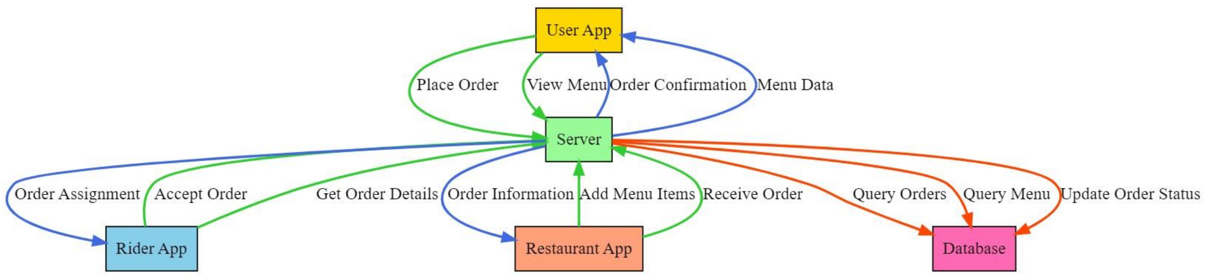
3. *Responsive UI*: The user interfaces of all apps are designed to be responsive, adapting to various screen sizes and orientations, providing an optimal user experience on smartphones and tablets.

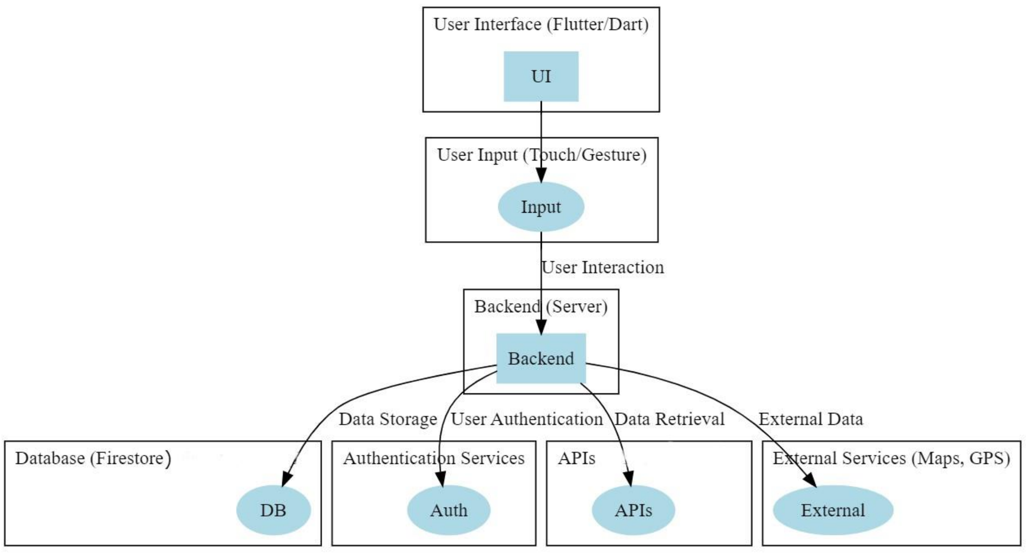
4. *Scalability*: The architecture is built with scalability in mind. As the user base grows, the system can easily handle increased traffic and data without significant performance degradation.

13. Illustrative Diagrams:

Below are some illustrative diagrams that provide a visual representation of the system's design and architecture:

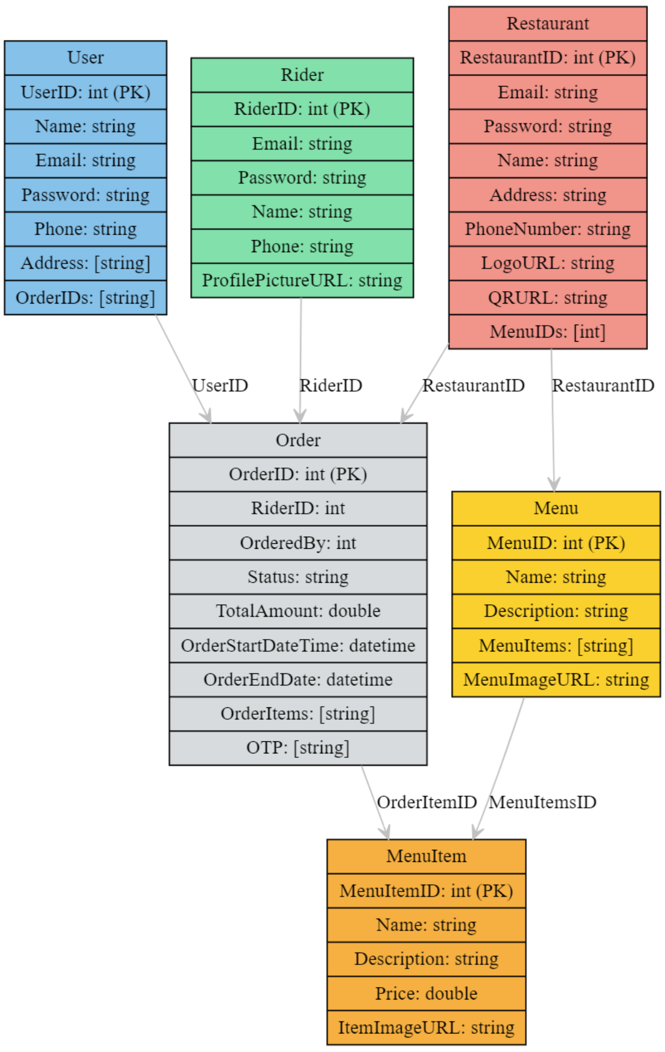
- System Overview:





These diagrams offer a high-level view of how our multi-app system is structured and how data flows between different components. They serve as a reference for both development and understanding the system's functionality.

-Database Design



By adhering to these design principles and utilizing Flutter's capabilities, our multi-app system is well-prepared to provide a seamless and efficient experience for users, riders, and restaurant partners while remaining adaptable to future enhancements and scaling requirements.

14. User App Implementation

1. Technical Aspects of App Development

- *Development Environment*: The User App was developed using the Flutter framework, which allows for cross-platform app development.

- *Programming Language*: Dart was used as the primary programming language for developing the app.

- *Authentication*: Firebase Authentication was integrated for user registration and login.

- *Database*: Firebase Realtime Database was used to store and retrieve user data, product information, and order history.

- *External Services*: External services, such as location data for maps, were integrated using appropriate APIs.

- *UI/UX Design*: The app's user interface was designed with a focus on user experience and modern design principles.

2. Features and Functionality

1. *Authentication :* Users can register and log in securely using email and password. Firebase Authentication handles user authentication.

2. *Home Screen* : The Home Screen displays a list of featured retaurants and categories. Users can browse and search for products.

3. *Item Details*: Users can view detailed information about any food item, including images, descriptions, and prices.

4. *Cart*: Users can add food items to their shopping cart. The cart displays product quantities and total amounts. Users can remove items from the cart.

5. *Order Placement*: Users can place orders by selecting items from their cart and providing delivery information. Order details are stored in the database.

6. *Order History*: Users can view their order history, including order statuses and delivery dates.

3. Code Snippets

Example : *User Authentication*

```
 Future<void> registerUser(String email, String password) async {
  try {
    UserCredential userCredential = await FirebaseAuth.instance
        .createUserWithEmailAndPassword(email: email, password:
password);
    // User registration successful
  } catch (e) {
    // Handle registration errors
  }

}

// Login
Future<void> loginUser(String email, String password) async {
  try {
    UserCredential userCredential = await FirebaseAuth.instance
        .signInWithEmailAndPassword(email: email, password:
password);
```

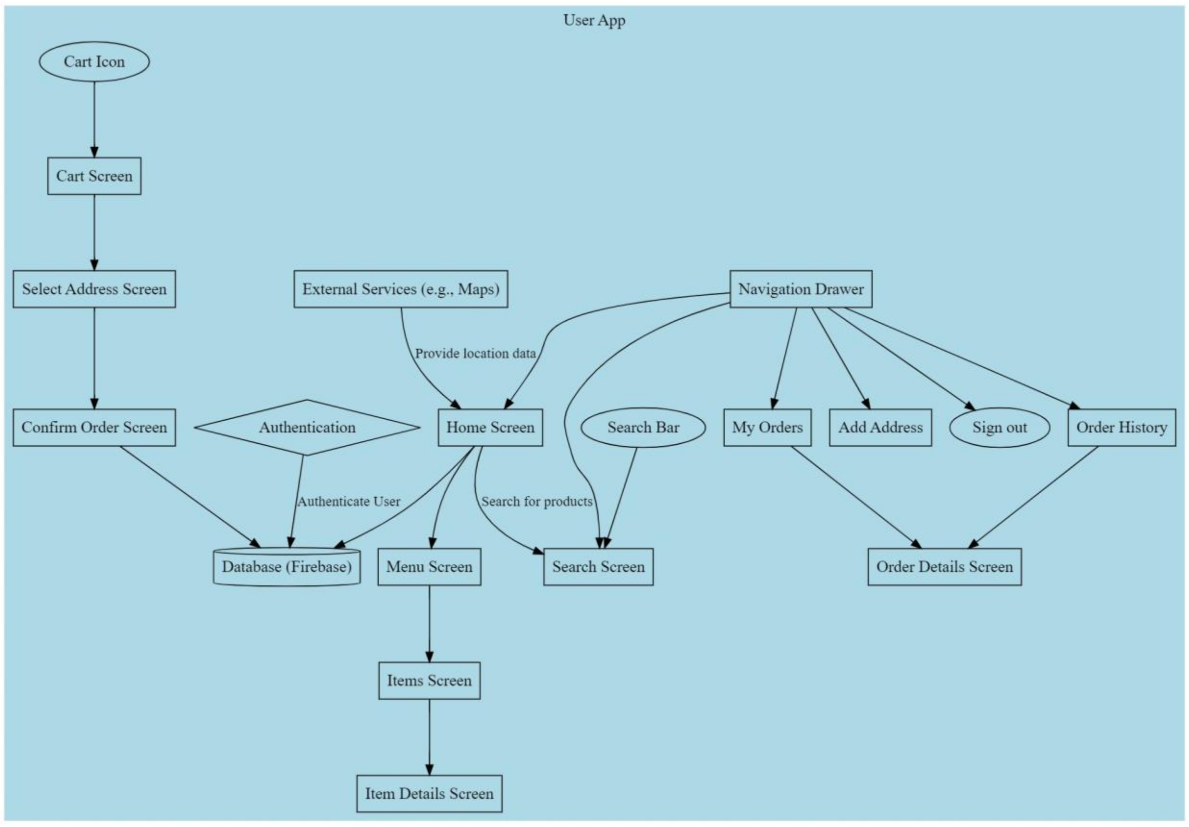
```
    // User login successful
  } catch (e) {
    // Handle login errors
  }
}
```

Example: *Adding Items to Cart*

```
class CartItem {
  String productId;
  double price;
  int quantity;

  CartItem(this.productId, this.price, this.quantity);
}

List<CartItem> cartItems = [];

void addToCart(String productId, double price) {
  // Check if the product is already in the cart
  for (var item in cartItems) {
    if (item.productId == productId) {
      item.quantity++;
      return;
    }
  }
  // If not, add a new item to the cart
  cartItems.add(CartItem(productId, price, 1));
}
```

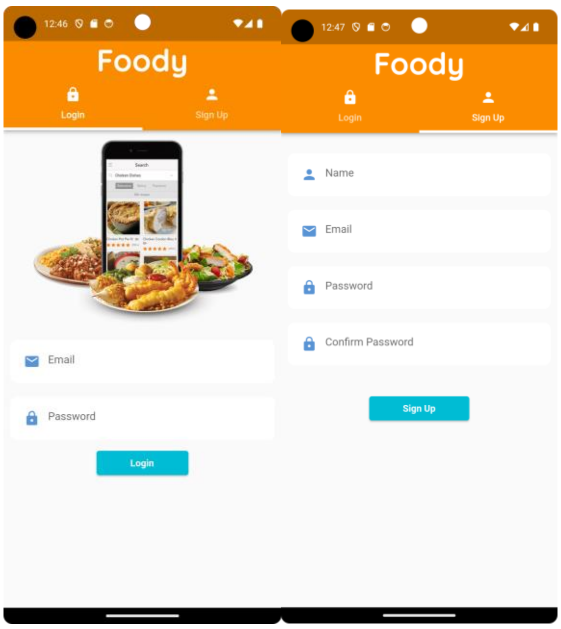
This provides a detailed overview of the User App's technical aspects, features, and code snippets for key components. Next, we can move on to the Rider App's Implementation.

4. User App Architecture



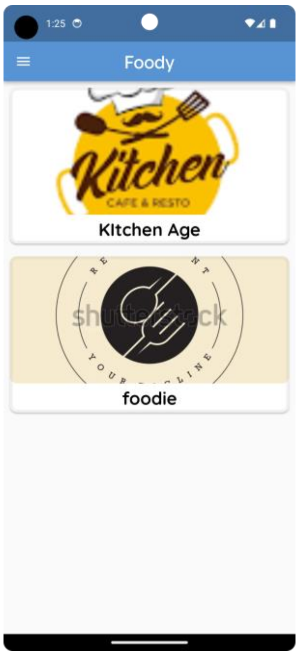
5. User App interfaces

1. Sign in / Register: Users can sign in to their existing accounts or register for new ones.



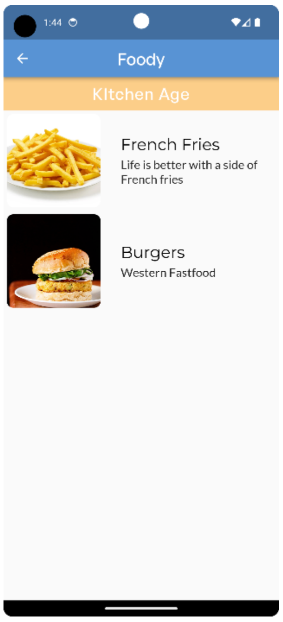
- The sign-in/register feature allows users to create accounts, which helps in personalizing their experience, saving delivery addresses, and tracking order history.

2. Home Screen: The home screen displays a curated list of nearby restaurants.



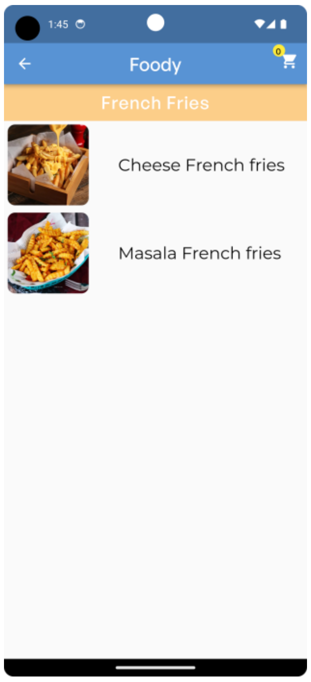
- The home screen provides users with quick access to restaurant options, promoting an easy browsing experience.

3. Restaurant Menu Screen: Users can view restaurant menus, and their descriptions.

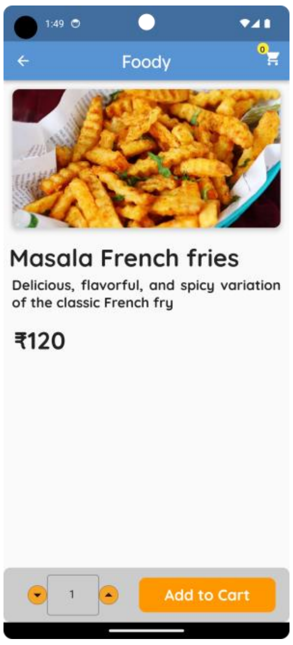


- The menu screen allows users to explore a restaurant's offerings, helping them make informed choices.

4. Menu Item Screen:
- Users can view all the menu items in a menu here.



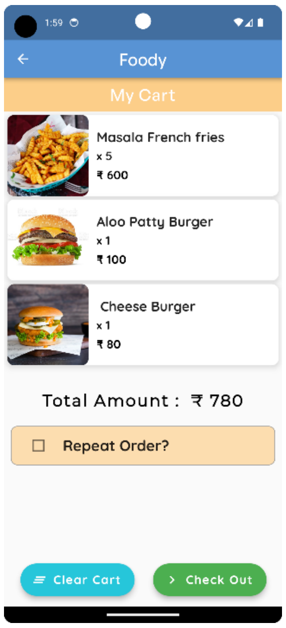
5. Item Details Screen: The "Item Details Screen" provides users with comprehensive information about a menu item, including its name, description, price and image.


1:49
Foody
0
Masala French fries
Delicious, flavorful, and spicy variation of the classic French fry
₹120
1
Add to Cart

- The "Item Details Screen" enhances the user experience by offering detailed information about menu items and displaying enticing images to encourage orders.

6. Cart Screen:

- Users can view items in their cart, including removing items.



Masala French fries
x 5
₹ 600

Aloo Patty Burger
x 1
₹ 100

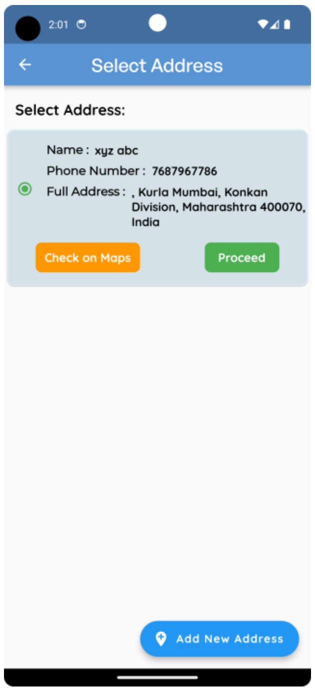
Cheese Burger
x 1
₹ 80

Total Amount : ₹ 780

- The cart screen serves as a virtual shopping cart where users can review their selected items and proceed to place an order.

7. Address Screen

- Users can select their delivery addresses here from all saved addresses and add new ones.

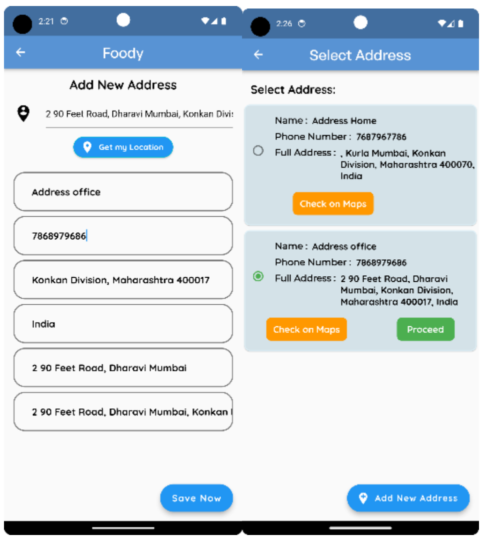


Select Address:

Name : xyz abc
Phone Number : 7687967786
Full Address : , Kurla Mumbai, Konkan Division, Maharashtra 400070, India

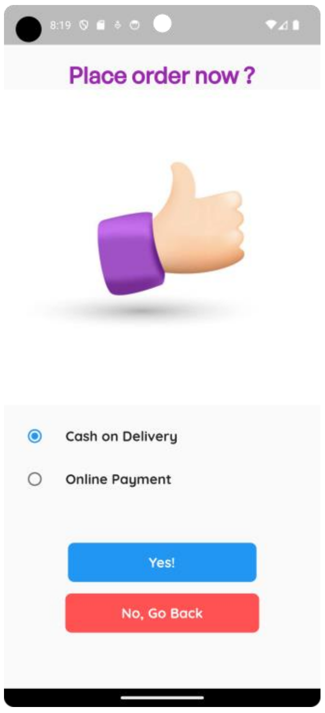
- The address screen provides users with the flexibility to save multiple delivery locations for convenience.

8. Add Address Screen: Users can add and save new delivery addresses.

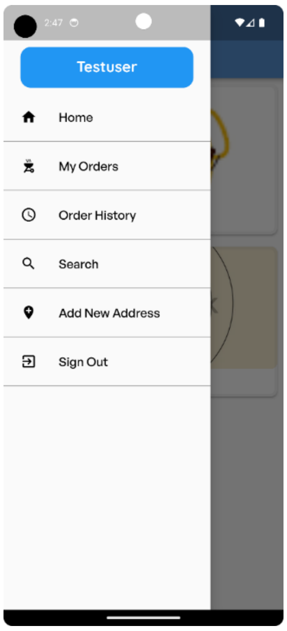

2:21
Foody
Add New Address
2 90 Feet Road, Dharavi Mumbai, Konkan Divi:
Get my Location
Address office
7868979686
Konkan Division, Maharashtra 400017
India
2 90 Feet Road, Dharavi Mumbai
2 90 Feet Road, Dharavi Mumbai, Konkan
Save Now
2:26
Select Address
Select Address:
Name : Address Home
Phone Number : 7687967786
Full Address : , Kurla Mumbai, Konkan Division, Maharashtra 400070, India
Check on Maps
Name : Address office
Phone Number : 7868979686
Full Address : 2 90 Feet Road, Dharavi Mumbai, Konkan Division, Maharashtra 400017, India
Check on Maps
Proceed
Add New Address

- The add address screen allows users to enter precise delivery information to streamline the ordering process.

9. Confirm Order Screen: The confirm order screen is the final step before placing an order, ensuring that users have the chance to verify their choices.



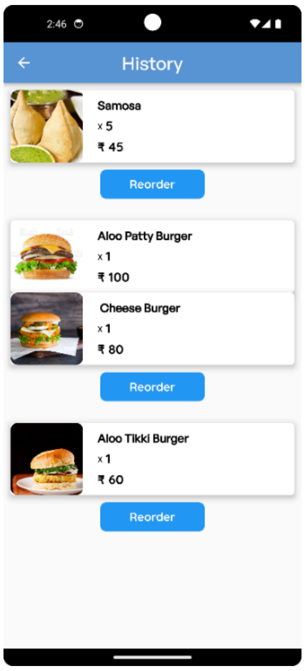
10. Side Bar: Users can access additional features such as order history, current orders, adding new address and searching restaurants.



- The side bar enhances user navigation by providing quick access to important sections of the app.

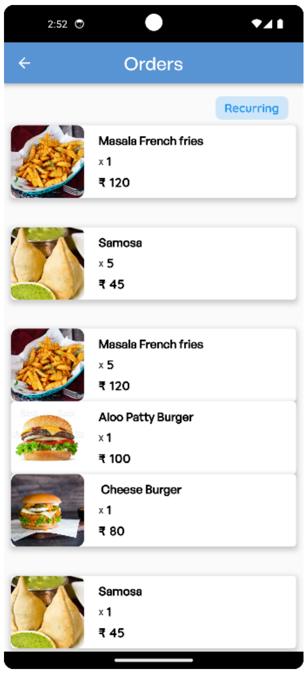
11. History Screen:

- Users can view their order history, including details of past orders.



- The history screen allows users to keep track of their previous orders, facilitating reordering or reviewing past transactions.

12. Current Orders Screen: Users can check the status of their current orders, including estimated delivery times



Masala French fries
x 1
₹ 120

Samosa
x 5
₹ 45

Masala French fries
x 5
₹ 120

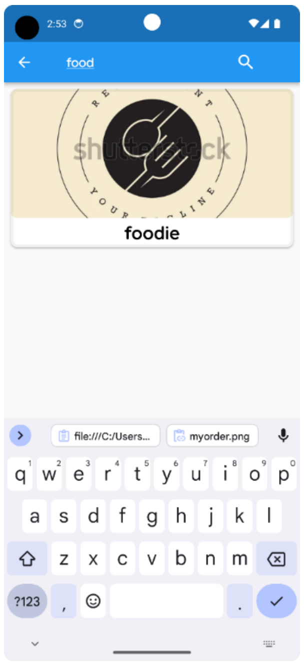
Aloo Patty Burger
x 1
₹ 100

Cheese Burger
x 1
₹ 80

Samosa
x 1
₹ 45

- The current orders screen provides real-time updates on orders in progress, enhancing transparency for users.

13. Search Restaurants: Users can search for specific restaurants, browse menus, and view details.



- The search feature enables users to quickly find their preferred restaurants or cuisines.

15. Restaurant App Implementation

1. Technical Aspects of App Development

- *Development Environment*: The Restaurant App was developed using the Flutter framework to ensure cross-platform compatibility.

- *Programming Language*: Dart was the primary programming language for developing the app.

- *Authentication*: Firebase Authentication was implemented for restaurant owners to securely manage their restaurants.

- *Database*: Firebase Realtime Database or Firestore (or a combination) was used to store and retrieve restaurant data, menu items, orders, and user reviews.

- *External Services*: External services like payment gateways for processing online payments were integrated into the app.

- *UI/UX Design*: The app's user interface was designed with a focus on usability, restaurant branding, and user engagement.

2. Features and Functionality

1. *Restaurant Profile*: Restaurant owners can create and manage their profiles, including restaurant name, contact information, location, and opening hours.

2. *Menu Management*: Restaurants can add, and delete menu items, including images, names, descriptions, and prices.

3. *Order Management*: Restaurants receive real-time notifications of incoming orders. They can accept, prepare, and mark orders as complete.

4.*Online Ordering*: Users can browse the restaurant's menu, add items to their cart, and place orders for delivery or pickup.

6. *Order Tracking*: Users can track the status of their orders, from preparation to delivery, and receive estimated delivery times.

3. Code Snippets

Example: *Managing Menu Items*

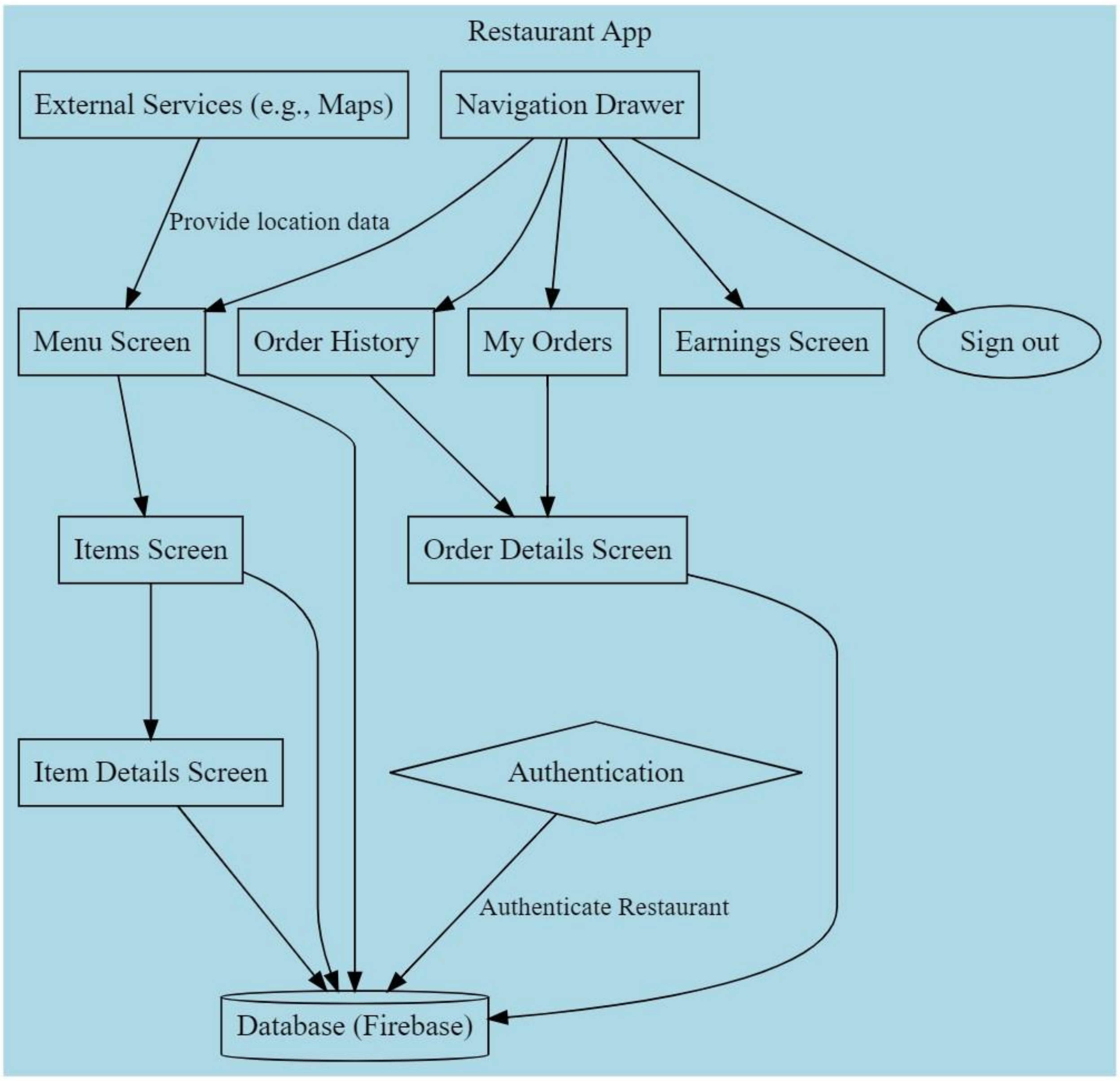
```
class MenuItem {
  String id;
  String name;
  String description;
  double price;
  String imageUrl;

  MenuItem(this.id, this.name, this.description, this.price,
this.imageUrl);
}

List<MenuItem> menuItems = [];

void addMenuItem(String name, String description, double price,
String imageUrl) {
  final newItem = MenuItem(
    DateTime.now().toString(),
    name,
    description,
    price,
    imageUrl,
  );
  menuItems.add(newItem);
  // Save the new menu item to the database
}

void deleteMenuItem(String id) {
  menuItems.removeWhere((item) => item.id == id);
  // Delete the menu item from the database
}
```

Example: *Handling Orders*

```
class Order {
  String orderId;
  String restaurantId;
  String userId;
  List<MenuItem> items;
  double totalAmount;
  DateTime orderTime;
  String status;

  Order(this.orderId, this.restaurantId, this.userId, this.items,
this.totalAmount, this.orderTime, this.status);
}

List<Order> orders = [];

void acceptOrder(String orderId) {
  final index = orders.indexWhere((order) => order.orderId ==
orderId);
  if (index >= 0) {
    orders[index].status = 'Accepted';
    // Update the order status in the database
  }
}
```

These code snippets illustrate how the Restaurant App handles menu management and order processing.

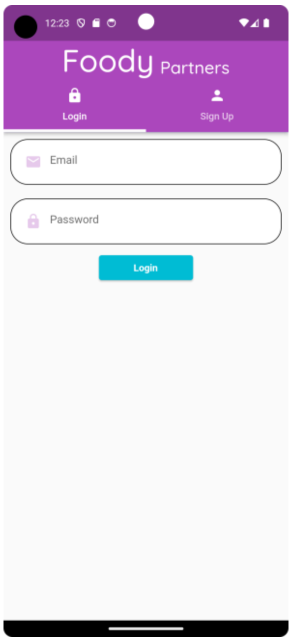
4. Restaurant App Architecture

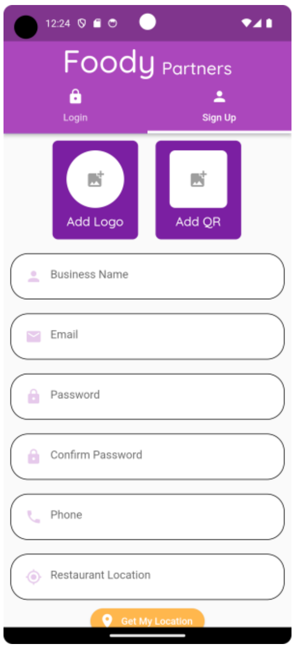


5. Restaurant App Interfaces

1. Sign In / Register:

Restaurants can create accounts, sign in, and manage their profiles.

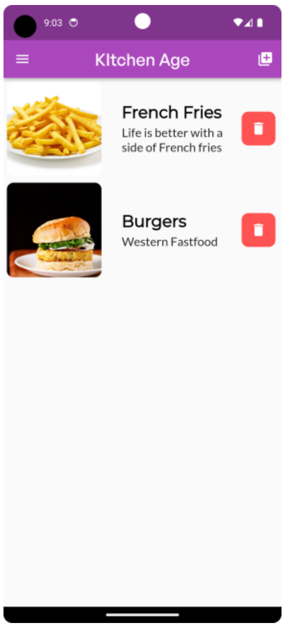




- The sign-in/register feature allows restaurants to access the platform, manage their information, and interact with orders.

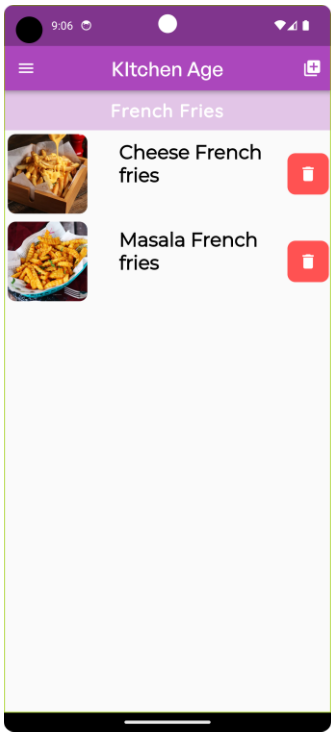
2. Home Screen:

Restaurants can see all of their menus to view and manage menu and its items.



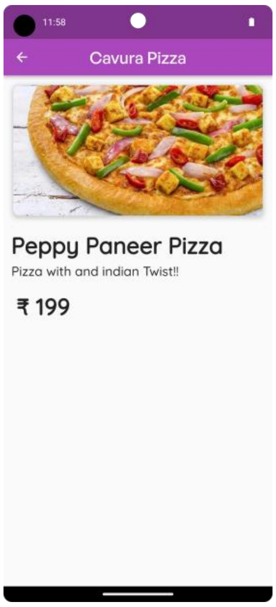
- The home screen serves as a medium for editing menus and its items for restaurants.

3. Menu Items Screen: Restaurants can create, delete, and update their menus, including menu names, descriptions and prices.



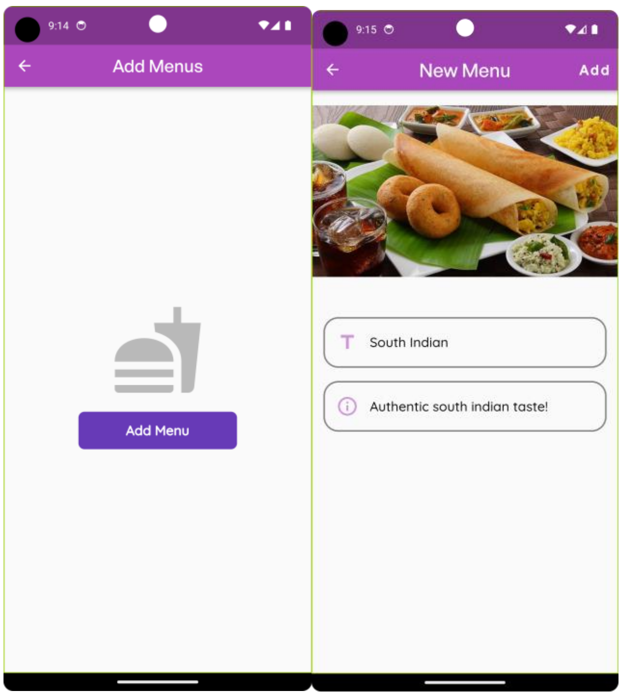
- Menu management empowers restaurants to showcase their offerings and make real-time updates as needed.

4. Item Details Screen: The "Item Details Screen" provides restaurants with comprehensive information about a menu item, including its name, description, price and images.



- The "Item Details Screen" enhances the user experience by offering detailed insights into menu items.

5. Add Menu Screen: The "Add Menu Screen" provides users option to add a menu, including its name, description and images.

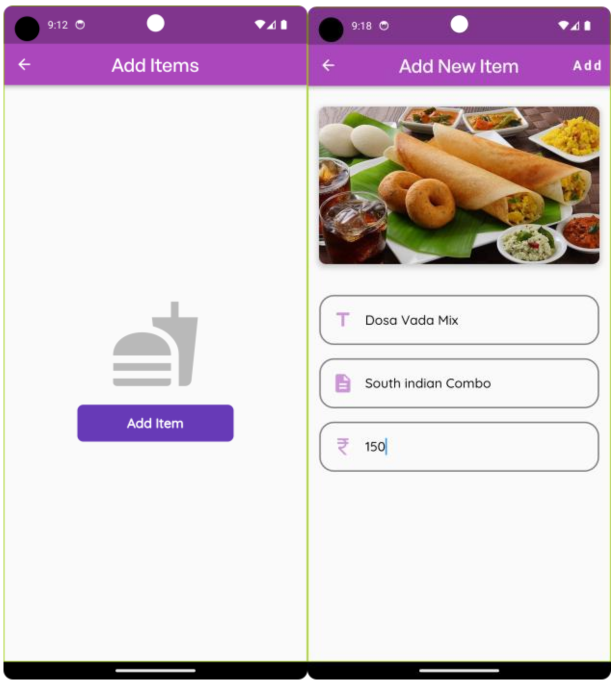


South Indian

Authentic south indian taste!

- The "Add Menu Screen" enhances the user experience by offering simple interface to add menus.

6. Add Item Screen: The "Add Item Screen" provides users option to add a menu item, including its name, description, price and images.



Dosa Vada Mix

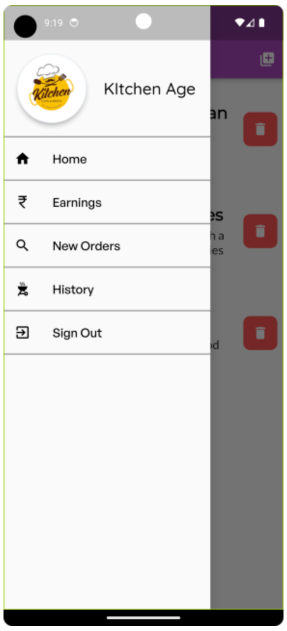
South indian Combo

₹ 150

- The "Add Menu Item Screen" enhances the user experience by offering simple interface to add menu items.

7. Side Bar:

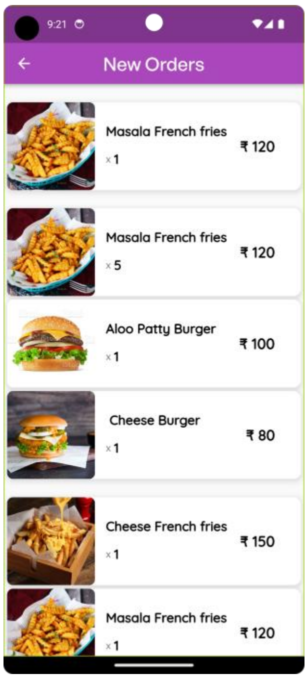
- Users can access additional features such as order history, current orders and earnings.



- The side bar enhances user navigation by providing quick access to important sections of the app.

8.New Orders Screen:

- “New Orders Screen” provides easy access to all the newly received orders.

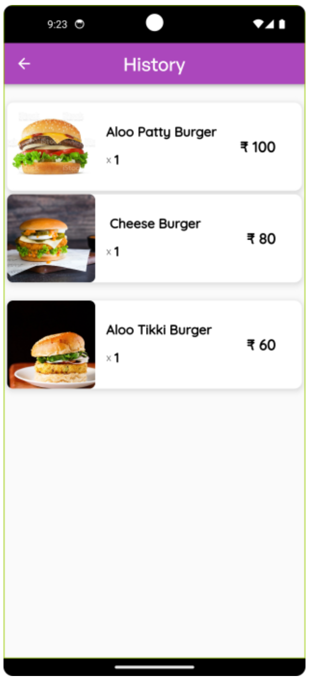


Masala French fries	x 1	₹ 120
Masala French fries	x 5	₹ 120
Aloo Patty Burger	x 1	₹ 100
Cheese Burger	x 1	₹ 80
Cheese French fries	x 1	₹ 150
Masala French fries	x 1	₹ 120

- The “New Orders Screen” gives restaurants an easy access to all the new orders received.

9. Order History:

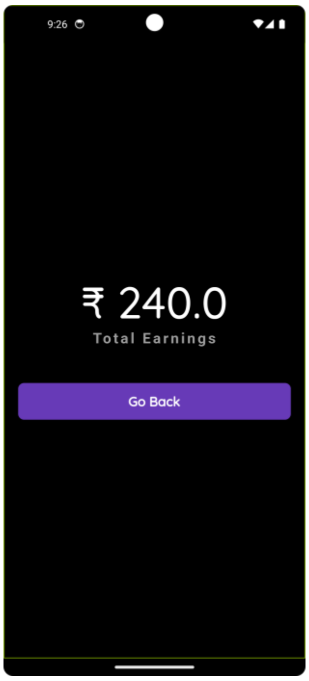
- Restaurants can access detailed order histories, including past orders, customer details, and transaction summaries.



Aloo Patty Burger	x 1	₹ 100
Cheese Burger	x 1	₹ 80
Aloo Tikki Burger	x 1	₹ 60

- Order history provides a comprehensive record of past transactions for reference and analysis.

10. Earnings Screen : The Earnings Screen gives restaurant easy access to the revenue they made.



16. Rider App Implementation

1. Technical Aspects of App Development

- *Development Environment*: The Rider App was developed using the Flutter framework, ensuring compatibility with both iOS and Android.

- *Programming Language*: Dart was used as the primary programming language for developing the app.

- *Authentication*: Firebase Authentication was implemented for riders to securely log in and manage their profiles.

- *Database*: Firebase Realtime Database or Firestore (or a combination) was used to store and retrieve order information, rider details, and route planning.

- *External Services*: Integration with location services and maps for real-time tracking of orders and route optimization.

- *UI/UX Design*: The app's user interface was designed with a focus on simplicity, efficient order handling, and navigation.

2. Features and Functionality

1. *Rider Profile*: Riders can create and manage their profiles, including name, contact information, and availability status.

2. *Order Assignment*: Orders are assigned to riders based on their availability and proximity to the restaurant and customer's location.

3. *Real-time Tracking*: Riders have access to real-time order tracking and navigation assistance to reach the restaurant and customer's location.

4. *Order Verification*: Riders can verify orders upon pickup to ensure correctness and completeness.

5. *Delivery Confirmation*: After delivering orders, riders can confirm successful deliveries, collect payments (if applicable), and update order statuses.

6. *Navigation and Route Optimization*: The app uses mapping and GPS services to provide optimized routes for riders.

7. *Notifications*: Riders receive notifications for new orders, updates on order status changes, and route guidance.

8. *Earnings and Delivery History*: Riders can access their earnings, including delivery histories.

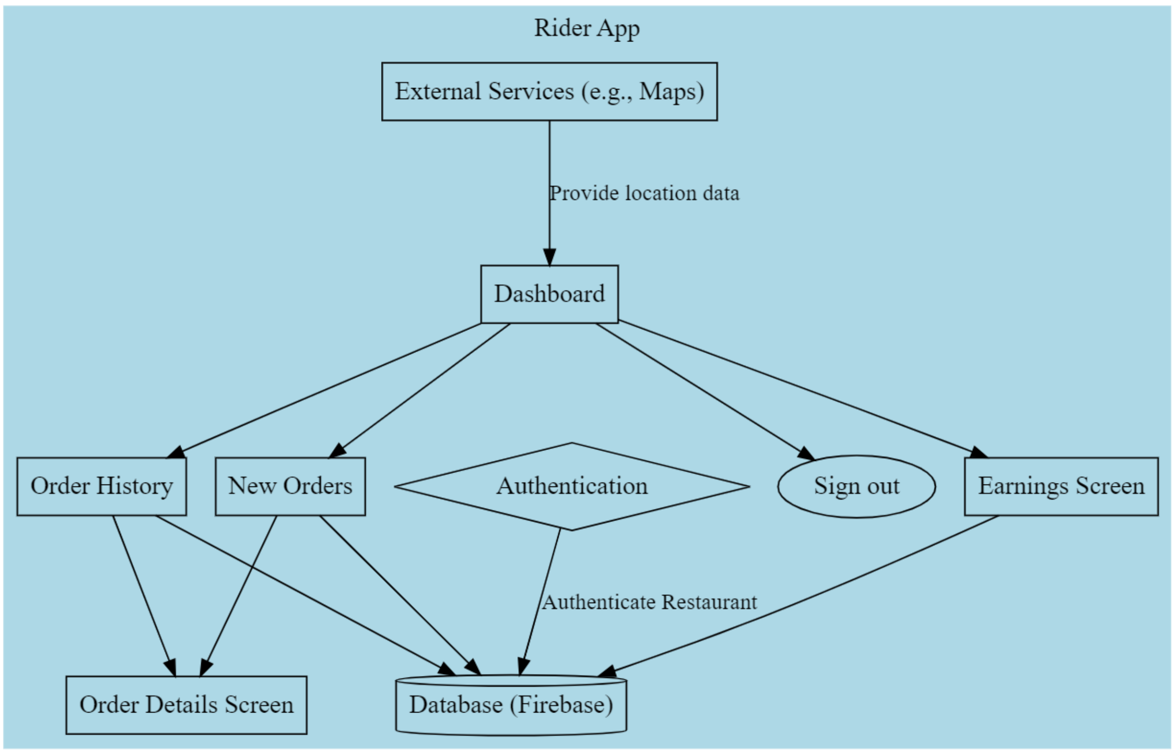
3. Code Snippets

Example: *Handling Order Assignments*

```
class OrderAssignment {
  String orderId;
  String riderId;
  bool accepted;
  DateTime assignedTime;

  OrderAssignment(this.orderId, this.riderId, this.accepted,
this.assignedTime);
}

List<OrderAssignment> orderAssignments = [];

void assignOrderToRider(String orderId, String riderId) {
  final newAssignment = OrderAssignment(orderId, riderId, false,
DateTime.now());
  orderAssignments.add(newAssignment);
  // Update the assignment in the database
}

void acceptOrderAssignment(String orderId) {
  final index = orderAssignments.indexWhere((assignment) =>
assignment.orderId == orderId);
  if (index >= 0) {
    orderAssignments[index].accepted = true;
    // Update the assignment status in the database
  }
}
```

These code snippets illustrate how the Rider App handles order assignments and real-time location tracking.

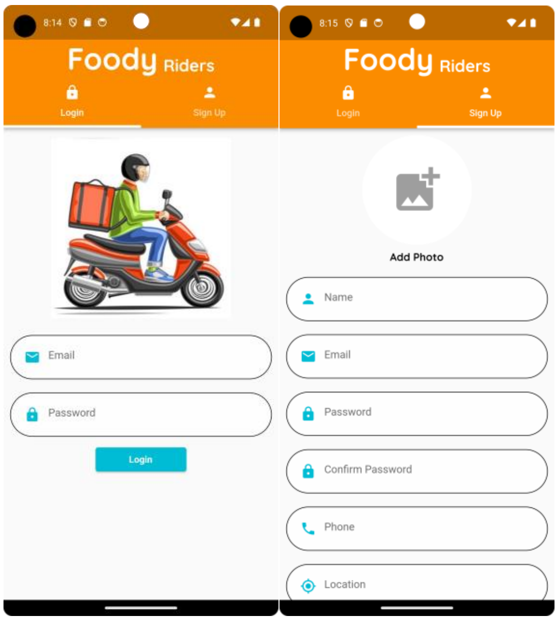
4. Rider App Architecture



5. Riders App Interfaces

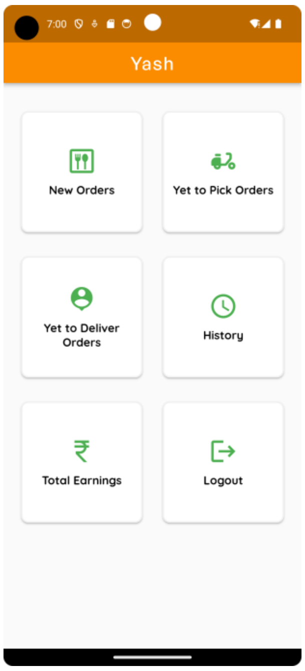
1. Sign In / Register:

- Riders can create accounts, sign in, and manage their profiles.



- The sign-in/register feature allows riders to access the platform, manage their information, and view and accept delivery requests.

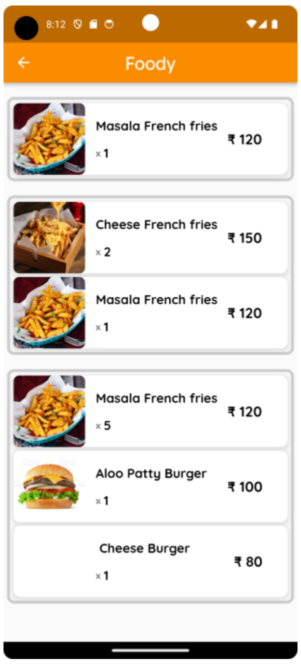
2. Dashboard: Riders have a centralized dashboard to view and manage incoming delivery requests.



- The dashboard serves as a control center for riders, displaying new orders ,yet to pick-up orders and yet to deliver orders,order history, earnings and option to sign out.

3.New Orders Screen :

- Riders can see all new delivery requests, view order details, and navigate to pick-up and drop-off locations.

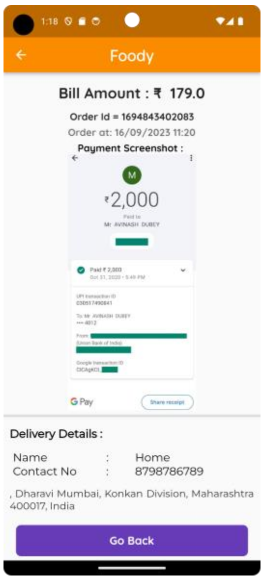


Masala French fries	×1	₹ 120
Cheese French fries	×2	₹ 150
Masala French fries	×1	₹ 120
Masala French fries	×5	₹ 120
Aloo Patty Burger	×1	₹ 100
Cheese Burger	×1	₹ 80

- Delivery management tools enable riders to efficiently fulfill delivery orders, ensuring timely and accurate service.

4. Order Details Screen:

Rider can see all the necessary details related to the order here.



Bill Amount : ₹ 179.0

Order Id = 1694843402083

Order at: 16/09/2023 11:20

Payment Screenshot :

₹2,000

Paid to

Mr AVINASH DUBEY

[redacted]

Paid ₹ 2,000

UPI transaction ID

To: Mr AVINASH DUBEY

From: [redacted]

(Union Bank of India)

[redacted]

Google transaction ID

CICAgKDI [redacted]

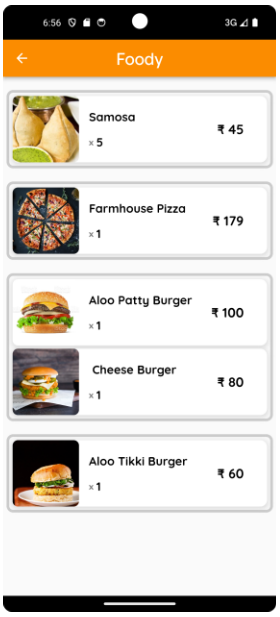
Delivery Details :

Name	:	Home
Contact No	:	8798786789

, Dharavi Mumbai, Konkan Division, Maharashtra 400017, India

5. Yet to Pick up Orders Screen:

Rider can see all the order that he /she has just accepted.



Samosa ×5 ₹ 45

Farmhouse Pizza ×1 ₹ 179

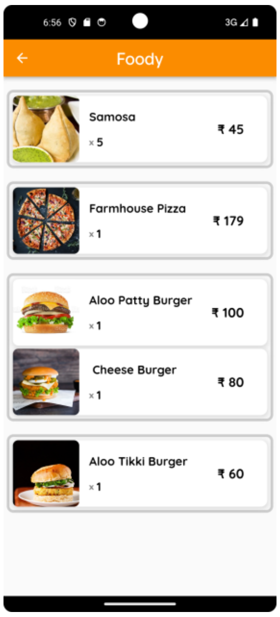
Aloo Patty Burger ×1 ₹ 100

Cheese Burger ×1 ₹ 80

Aloo Tikki Burger ×1 ₹ 60

6. Yet to Deliver Orders Screen :

Rider can see all the orders that he/she has just picked up from restaurant.



Samosa ×5 ₹ 45

Farmhouse Pizza ×1 ₹ 179

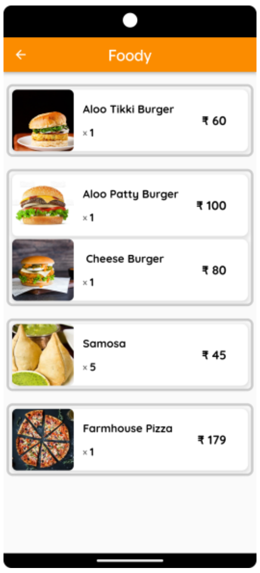
Aloo Patty Burger ×1 ₹ 100

Cheese Burger ×1 ₹ 80

Aloo Tikki Burger ×1 ₹ 60

7. Delivery History:

- Riders can access detailed delivery histories, including past deliveries, order details.



Aloo Tikki Burger ×1 ₹ 60

Aloo Patty Burger ×1 ₹ 100

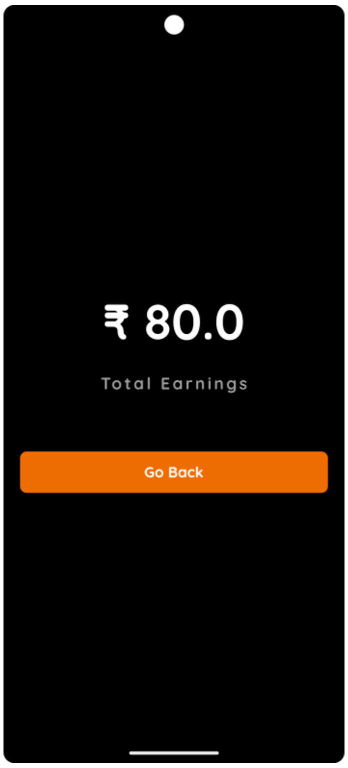
Cheese Burger ×1 ₹ 80

Samosa ×5 ₹ 45

Farmhouse Pizza ×1 ₹ 179

- Delivery history provides a comprehensive record of past deliveries for reference.

8. Earnings Screen : The Earnings Screen gives riders easy access to the amount of money they made.



17. Conclusion

The project has achieved significant milestones and delivered a comprehensive solution for users, restaurants, and riders. Below, we summarize the outcomes, share insights gained during the development process, and identify potential areas for future improvement.

1. Outcomes and Achievements

- *Empowering Local Businesses*: The User and Restaurant Apps have successfully bridged the gap between local restaurants and customers, allowing small businesses to expand their reach.

- *Enhanced User Experience*: Users can conveniently order food for specific dates and durations, providing flexibility and control over their dining experiences.

- *Efficient Order Management*: Restaurants can manage orders efficiently, reducing errors and ensuring deliveries.

- *Streamlined Delivery*: Riders benefit from real-time tracking and route optimization done by Google maps, resulting in faster and more accurate deliveries.

2. Learning Insights

Throughout the development process, several valuable insights were gained:

- *Cross-Platform Development*: Developing using Flutter significantly reduced development time and maintenance efforts.
- *Real-time Features*: Implementing real-time tracking and updation posed technical challenges but enhanced the overall user experience.
- *Firebase Integration*: Utilizing Firebase for authentication, real-time databases, and cloud storage streamlined backend development.

3. Limitations and Future Improvements

Despite the project's success, there are areas for further enhancement:

- *Payment Integration*: Integrating payment gateways to enable online transactions would make the platform more versatile.

- *Feedback Mechanisms*: Implementing feedback mechanisms for users to rate restaurants and riders can enhance service quality.

- *User Engagement*: Enhancing user engagement through personalized recommendations and promotions can increase user retention.

In conclusion, this project has provided a valuable solution for the food delivery industry by connecting users, restaurants, and riders. It has not only demonstrated technical proficiency but also highlighted the importance of user-centric design and efficient order management. With a commitment to ongoing improvement and innovation, this platform has the potential to revolutionize the food delivery experience.