

Team members

Omar talal megahd 2305007

Ahmed mohamed saied 2305126

Ahmed wessam salem 2305034

Mobile E-Commerce Flutter Application

2. Project Overview

This mobile application is developed using **Flutter** and integrated with **Firebase** for backend services.

The app implements:

1. Firebase Authentication (Login/Signup)
2. Product browsing
3. Shopping cart
4. Favorites list
5. Firestore database integration
6. State management with Providers
7. Firebase options configured via `firebase_options.dart` [GitHub](#)

The project supports **Android** and **iOS** mobile platforms.

3. Tech Stack & Tools

Layer	Technologies
Frontend	Flutter
Backend	Firebase
Database	Firestore
State Management	Provider
Version Control	GitHub

4. Repository Structure

```
Moblie-project/  
├─ android/  
├─ ios/  
├─ lib/  
│   ├─ cart.dart  
│   ├─ cart_provider.dart  
│   ├─ favorites_provider.dart  
│   ├─ home_screen.dart  
│   ├─ login_screen.dart  
│   ├─ signup_screen.dart  
│   ├─ splash_screen.dart  
│   └─ main.dart  
├─ firebase.json  
├─ firebase_options.dart  
├─ pubspec.yaml  
├─ screenshots/  
└─ README.md
```

- ◆ This structure ensures MVC-like separation between UI, services, and providers. [GitHub](#)

5. How to Install Dependencies

Make sure Flutter is installed:

1. `flutter doctor`

Get all packages:

2. `flutter pub get`

6. Firebase Setup (Short Guide)

1. Go to **Firebase Console** → Create new project
2. Add Android & iOS apps

3. Download

- `google-services.json` → `android/app/`
- `GoogleService-Info.plist` → `ios/Runner/`

4. Enable **Firestore & Authentication** (Email/Password)

5. Replace the config values in your Flutter app

(Detailed steps can be added from your existing `firebase_setup.md` file.)

7. Firestore Rules (Example)

```
rules_version = '2';

service cloud.firestore {
  match /databases/{database}/documents {

    match /users/{userId} {
      allow read, write: if request.auth != null
        && request.auth.uid == userId;
    }

    match /carts/{cartId} {
      allow read, write: if request.auth != null;
    }

    match /favorites/{favoriteId} {
      allow read, write: if request.auth != null;
    }
  }
}
```

This ensures authenticated users can only access their own data.

8. How to Run the App

Android

```
flutter run
```


iOS


```
cd ios  
pod install  
cd ..  
flutter run
```


9. Screenshots (Visuals)


1:27


Create Account


 Full Name

 Email

 Password

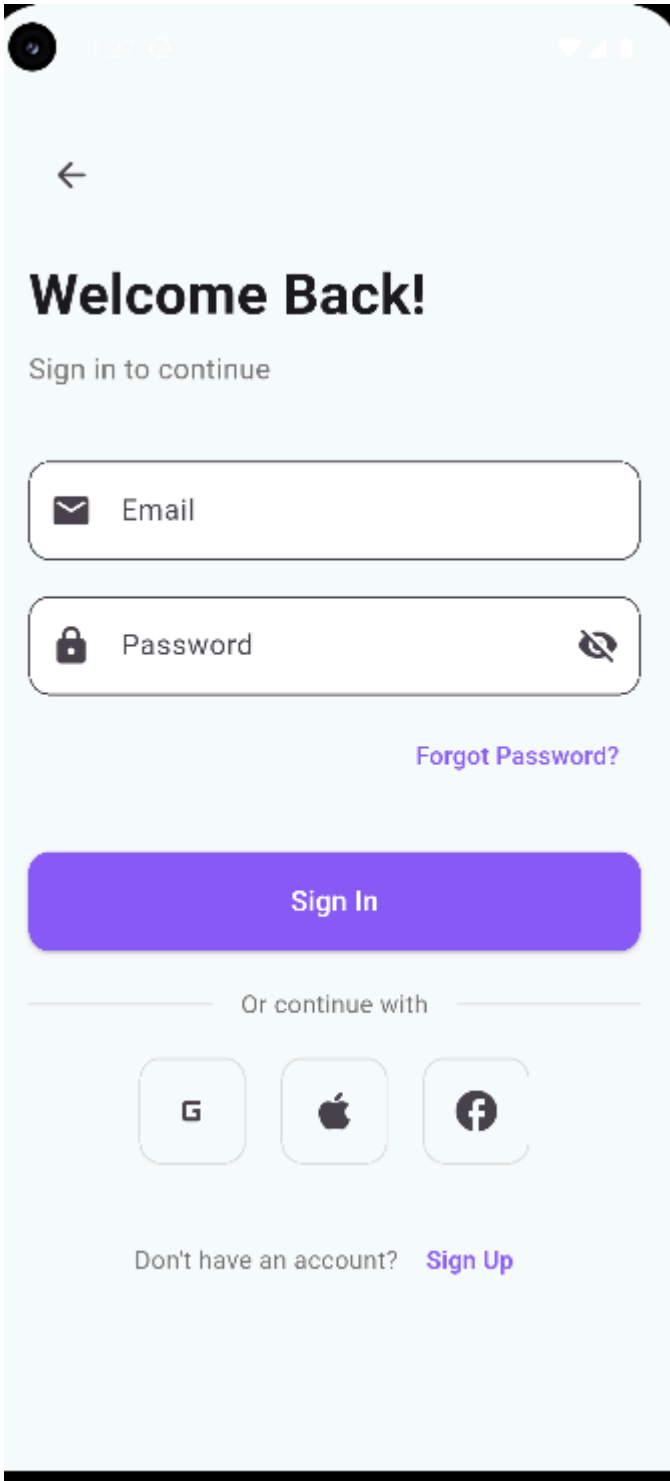


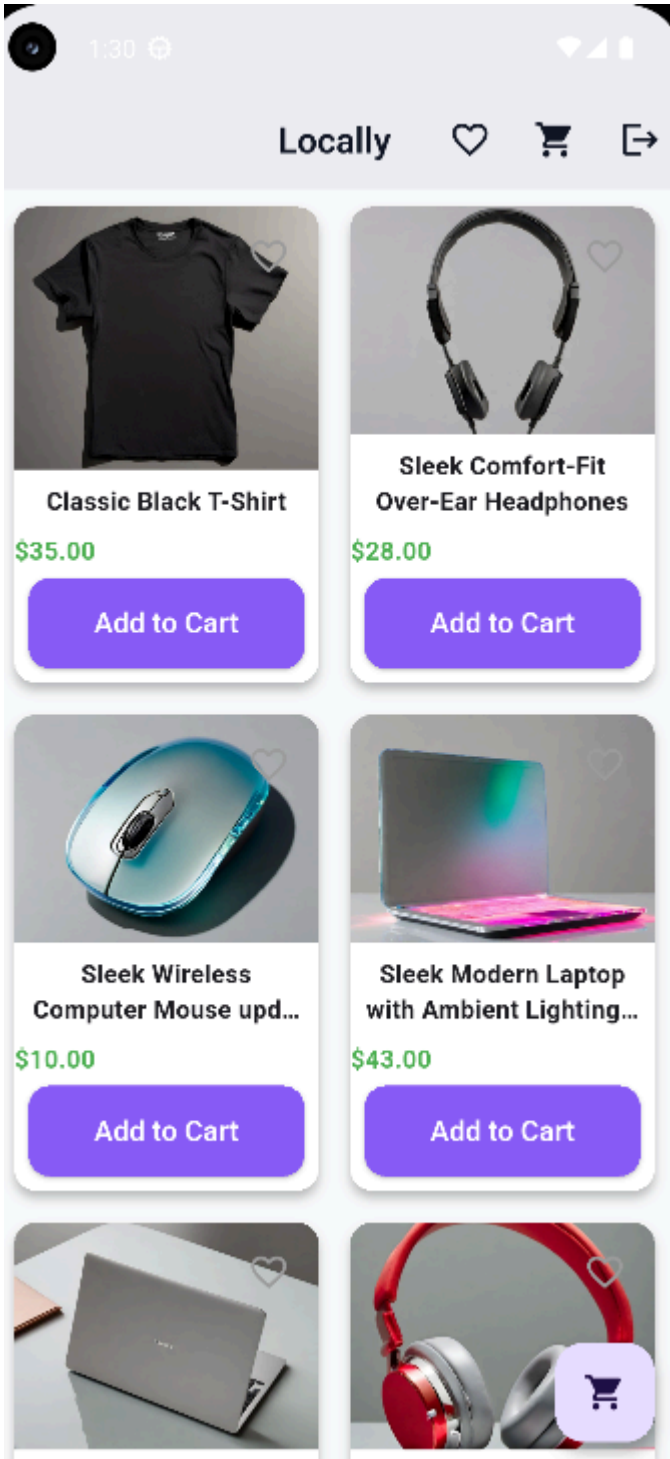
 Confirm Password

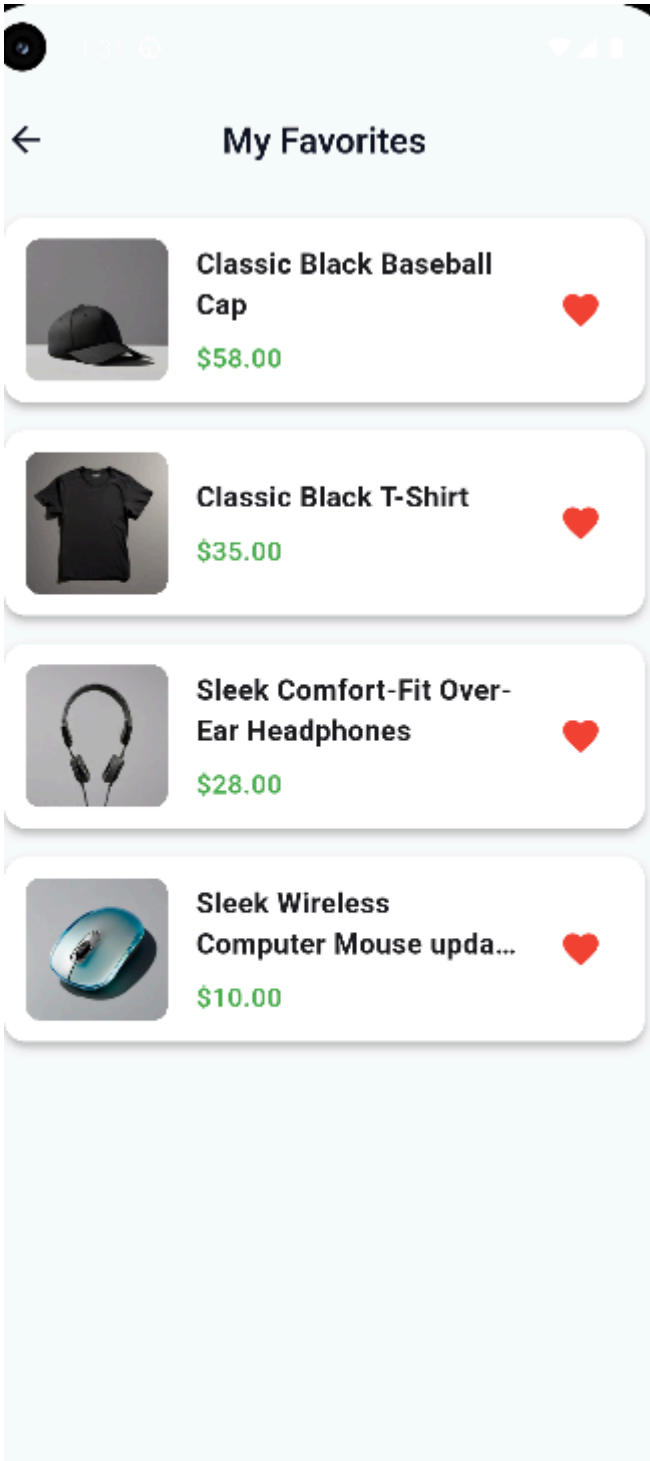


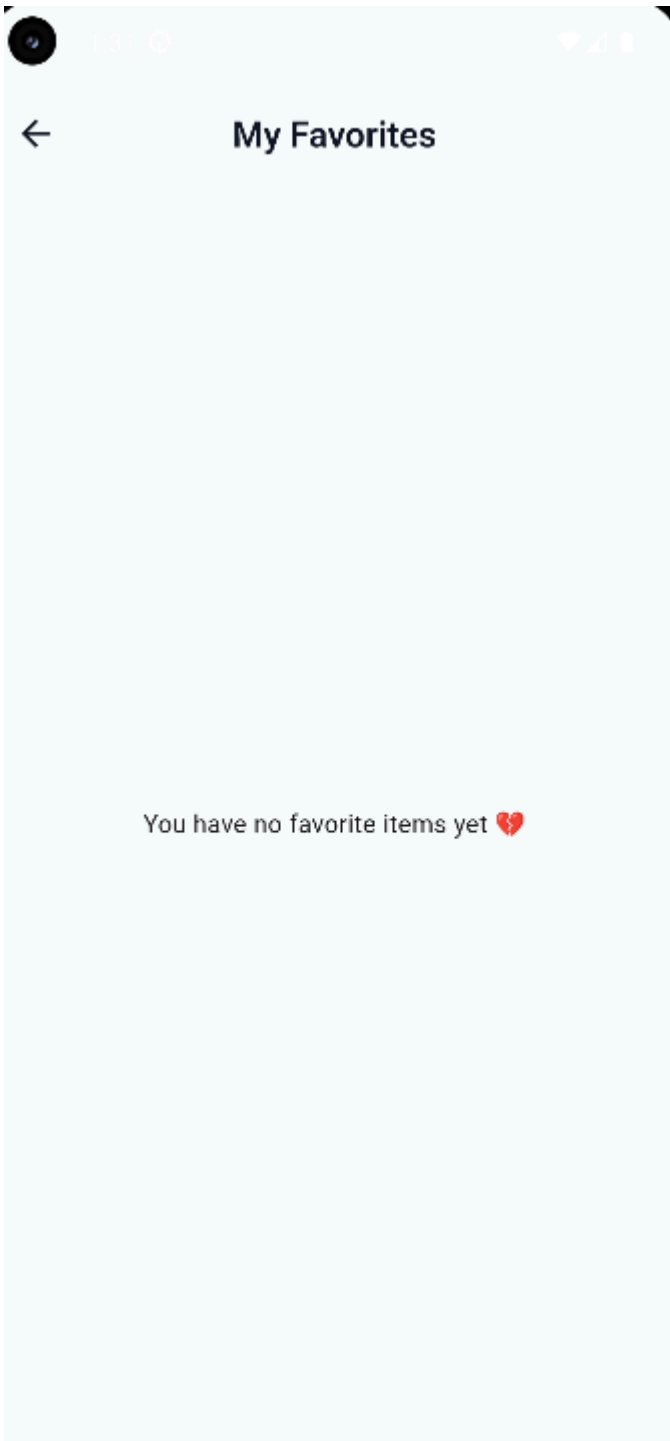
Sign Up

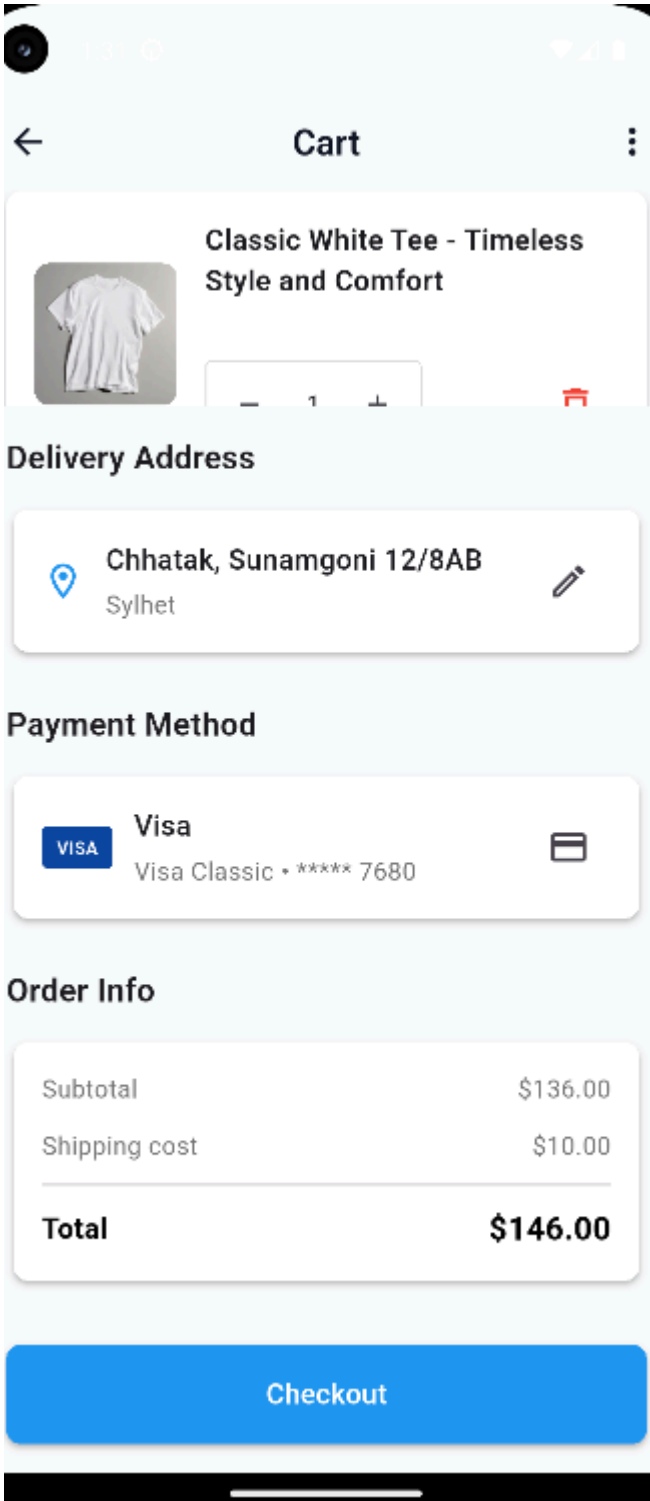
Already have an account? [Login](#)











1:12

←

Order Summary

Order Total

\$146.00

Shipping Address

Street Address

Smouha

City

Alexandria

ZIP Code

124512

Place Order

1

2

3

—

4

5

6

⌊

7

8

9

⌫

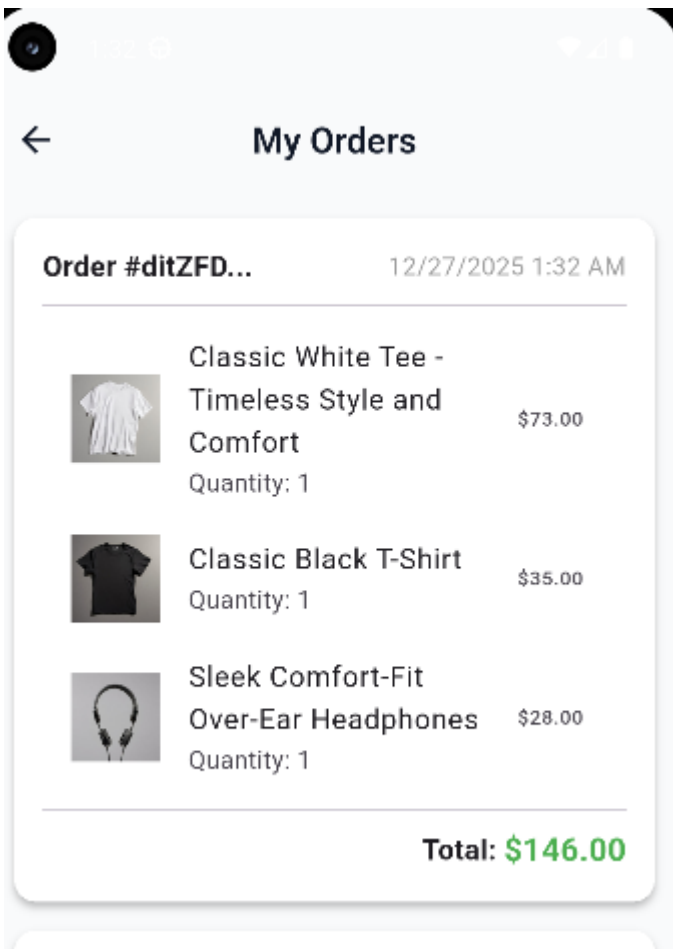
,

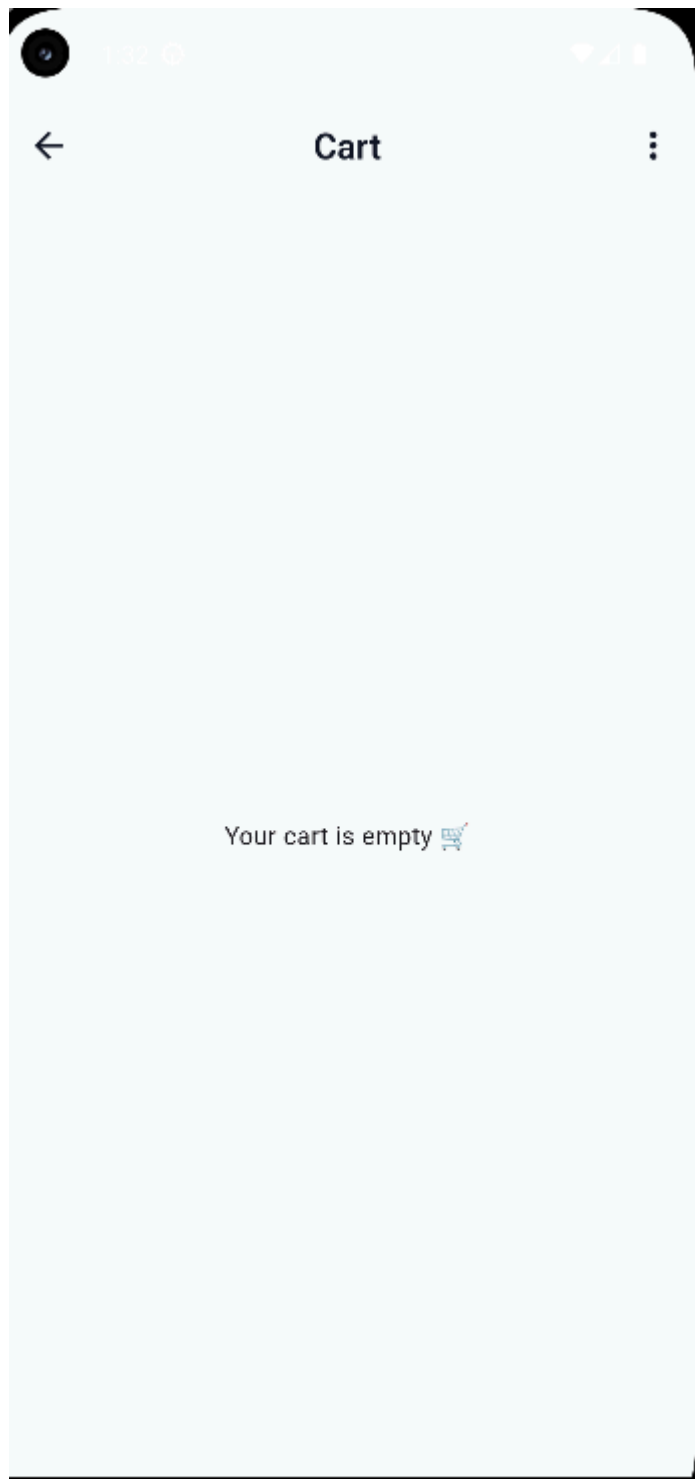
0

.

✓

▼





10. Features & Implementation

Authentication

Email/Password login & signup

User session persistence

Cart

Add / Remove items
Real time display with Provider

Favorites

Add to favorites
List favorites per user

Firestore

Collections: users, carts, favorites

11. Design Decisions

Use **Provider** for state management
Separation of UI & Logic (services & providers)
Firebase config auto generated using FlutterFire
Focus on simple scalable architecture

12. Conclusion

This project demonstrates a complete integration of a Flutter frontend with Firebase backend services, including realtime database features and state management, making it suitable as a **portfolio project**, **academic submission**, or **starter template for future apps**.

13. References

Flutter — UI toolkit for building natively compiled applications.
<https://github.com/omar14112004/Moblie-project.git>

FlutterFire — Firebase plugins for Flutter.
<https://github.com/omar14112004/Moblie-project.git>