Name - Tejas Shailendra Rokade
Div - D15A		Roll no - 50
Batch - C

Experiment - 6

Aim - To Connect Flutter UI with fireBase database

Theory -

Firebase is a popular mobile and web application development platform powered by Google, offering a suite of tools and services to streamline various aspects of app development. In Flutter, Google's UI toolkit for building natively compiled applications for mobile, web, and desktop from a single codebase, Firebase integration is seamless and efficient.

Firebase provides several services that cater to different aspects of app development:

1. Authentication: Firebase Authentication offers simple SDKs and ready-to-use UI libraries to authenticate users across various platforms. With Firebase Authentication, developers can implement authentication mechanisms such as email/password, phone number, social logins (Google, Facebook, Twitter, etc.), and more into their Flutter apps with ease.

2. Realtime Database: Firebase Realtime Database is a cloud-hosted NoSQL database that allows developers to store and synchronize data between users in real-time. It's particularly useful for applications that require real-time updates, such as chat apps, collaborative tools, and multiplayer games. Flutter developers can seamlessly integrate Firebase Realtime Database into their apps to store and sync structured data.

3. Cloud Firestore: Cloud Firestore is a flexible, scalable database for mobile, web, and server development from Firebase and Google Cloud Platform. It offers more powerful querying capabilities, real-time data synchronization, and offline support compared to the Realtime Database. Firestore is commonly used in Flutter apps for its scalability, real-time updates, and robust querying capabilities.

4. Cloud Storage: Firebase Cloud Storage provides secure and scalable object storage for storing user-generated content such as images, videos, and documents. It offers SDKs for easy integration into Flutter apps, allowing developers to upload, download, and manage files seamlessly.

5. Cloud Functions: Firebase Cloud Functions allows developers to run backend code in response to events triggered by Firebase features and HTTPS requests. Developers can write serverless functions in Node.js, Python, Java, Go, or TypeScript, and deploy them directly from

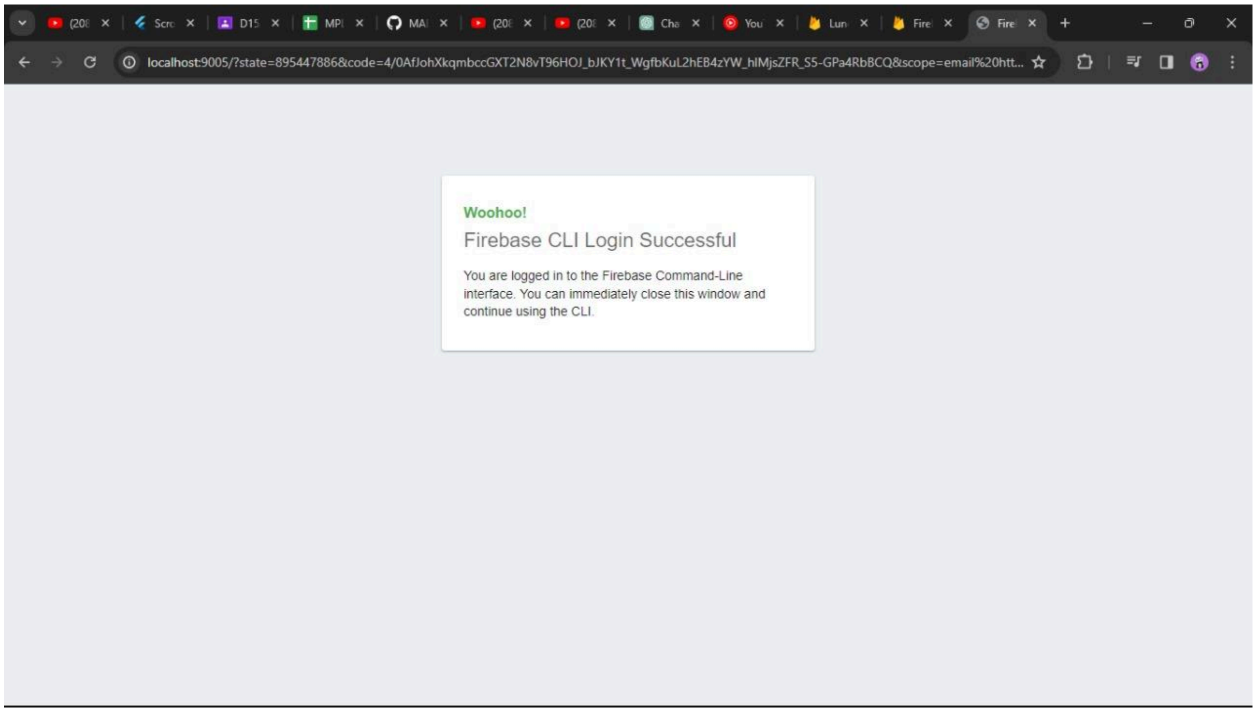
the Firebase CLI. Cloud Functions are often used to automate tasks, process data, and integrate with other Google Cloud services.

6. Cloud Messaging: Firebase Cloud Messaging (FCM) enables developers to send notifications and messages to users across platforms, including Android, iOS, and web. With FCM, Flutter developers can engage users with targeted messages, notifications, and updates, improving user retention and engagement.

7. Remote Config: Firebase Remote Config allows developers to customize the behavior and appearance of their Flutter apps without requiring an app update. It enables A/B testing, feature flag management, and dynamic content personalization, empowering developers to deliver tailored experiences to their users.

8. Analytics: Firebase Analytics provides comprehensive app usage and event tracking capabilities, allowing developers to gain insights into user behavior, app performance, and user engagement. Flutter developers can easily integrate Firebase Analytics into their apps to track user interactions, measure app performance, and make data-driven decisions.

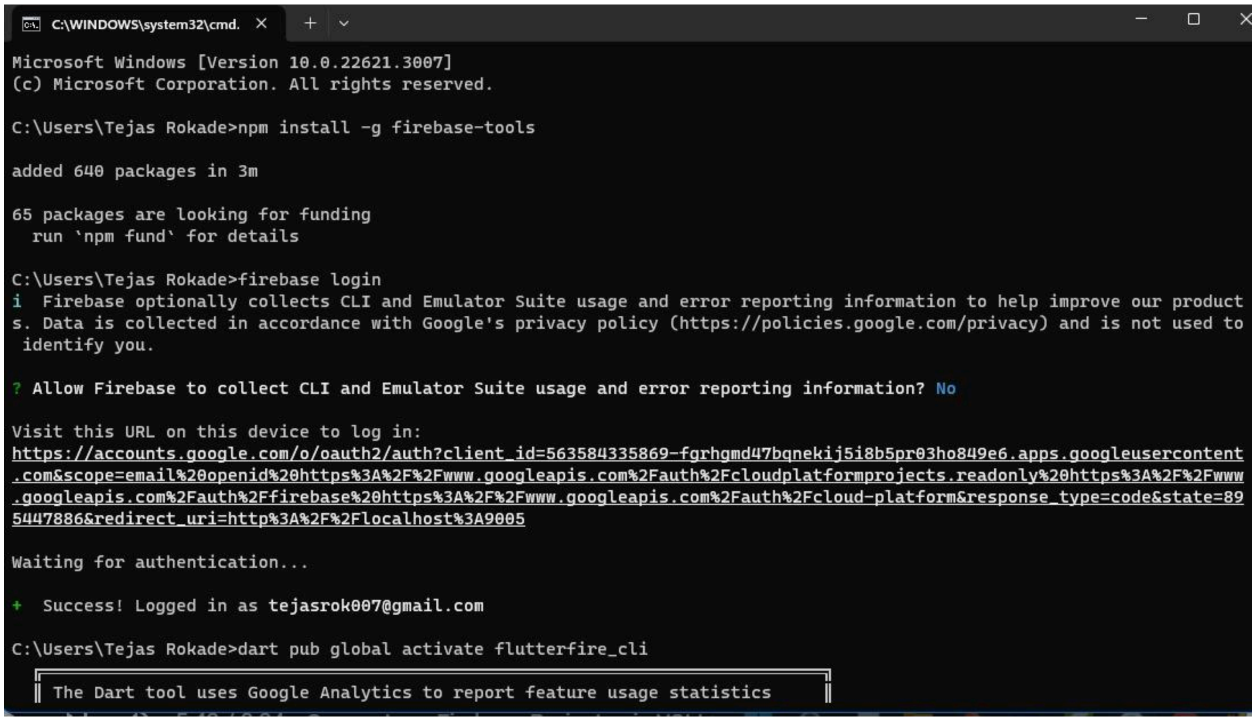
These are just a few of the many services offered by Firebase for Flutter app development. By leveraging Firebase's powerful tools and services, Flutter developers can build high-quality, feature-rich apps more efficiently, saving time and effort in the development process.


localhost:9005/?state=895447886&code=4/0AfJohXkqmbccGXT2N8vT96HOJ_bJKY1t_WgfbKuL2hEB4zYW_hlMjsZFR_S5-GPa4RbBCQ&scope=email%20htt...

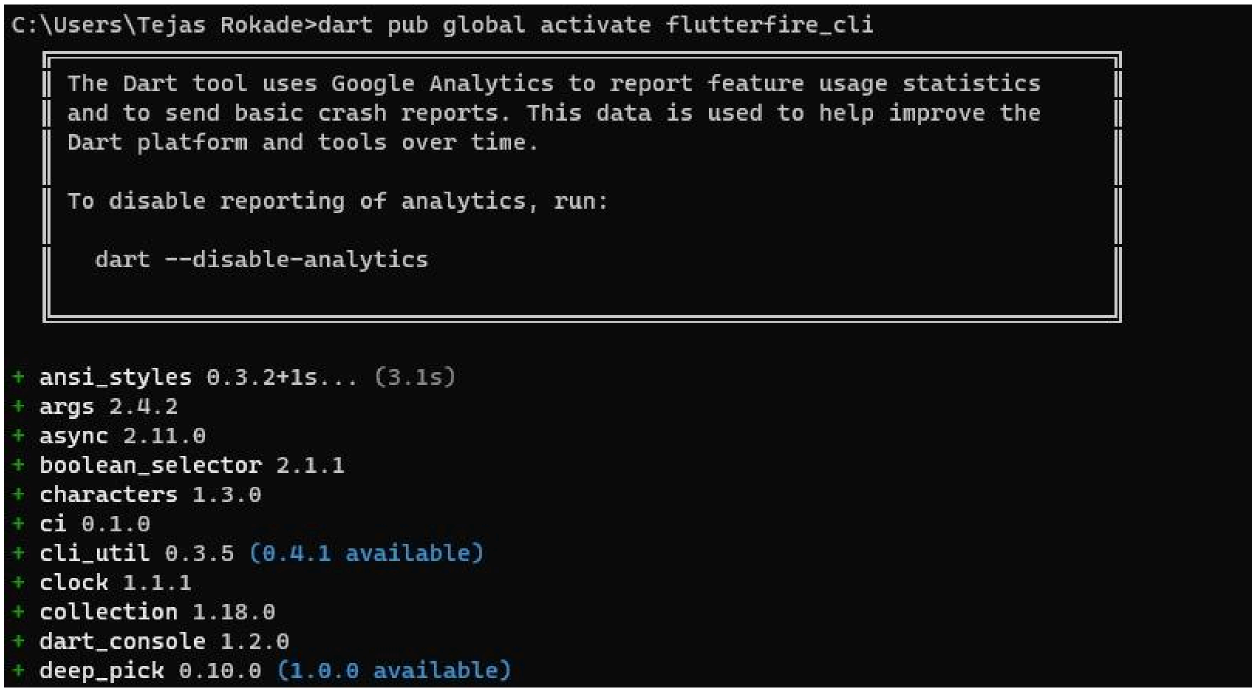
Woohoo!

Firebase CLI Login Successful

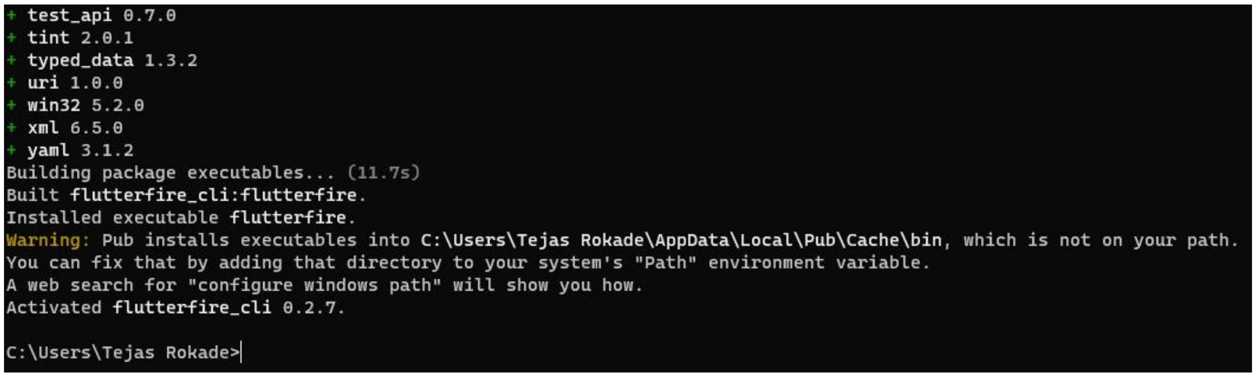
You are logged in to the Firebase Command-Line interface. You can immediately close this window and continue using the CLI.



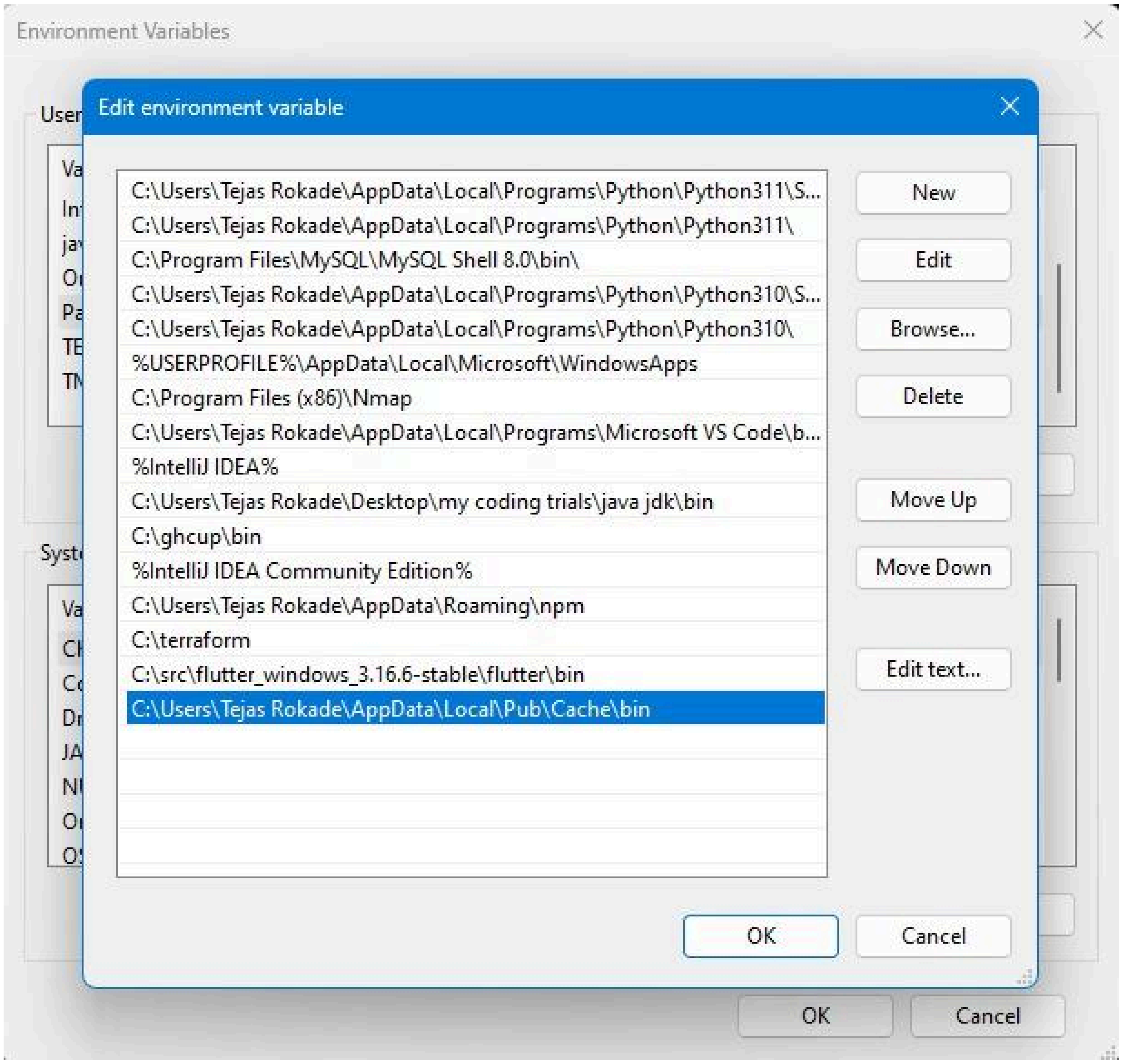
```
Microsoft Windows [Version 10.0.22621.3007]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Tejas Rokade>npm install -g firebase-tools

added 640 packages in 3m

65 packages are looking for funding
  run `npm fund` for details

C:\Users\Tejas Rokade>firebase login
i  Firebase optionally collects CLI and Emulator Suite usage and error reporting information to help improve our product
s. Data is collected in accordance with Google's privacy policy (https://policies.google.com/privacy) and is not used to
 identify you.

? Allow Firebase to collect CLI and Emulator Suite usage and error reporting information? No

Visit this URL on this device to log in:
https://accounts.google.com/o/oauth2/auth?client_id=563584335869-fgrhgmd47bqnekij5i8b5pr03ho849e6.apps.googleusercontent
.com&scope=email%20openid%20https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fcloudplatformprojects.readonly%20https%3A%2F%2Fwww
.googleapis.com%2Fauth%2Ffirebase%20https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fcloud-platform&response_type=code&state=89
5447886&redirect_uri=http%3A%2F%2Flocalhost%3A9005

Waiting for authentication...

+  Success! Logged in as tejasrok007@gmail.com

C:\Users\Tejas Rokade>dart pub global activate flutterfire_cli
  ╔════════════════════════════════════════════════════════════════════════╗
  ║ The Dart tool uses Google Analytics to report feature usage statistics ║
```



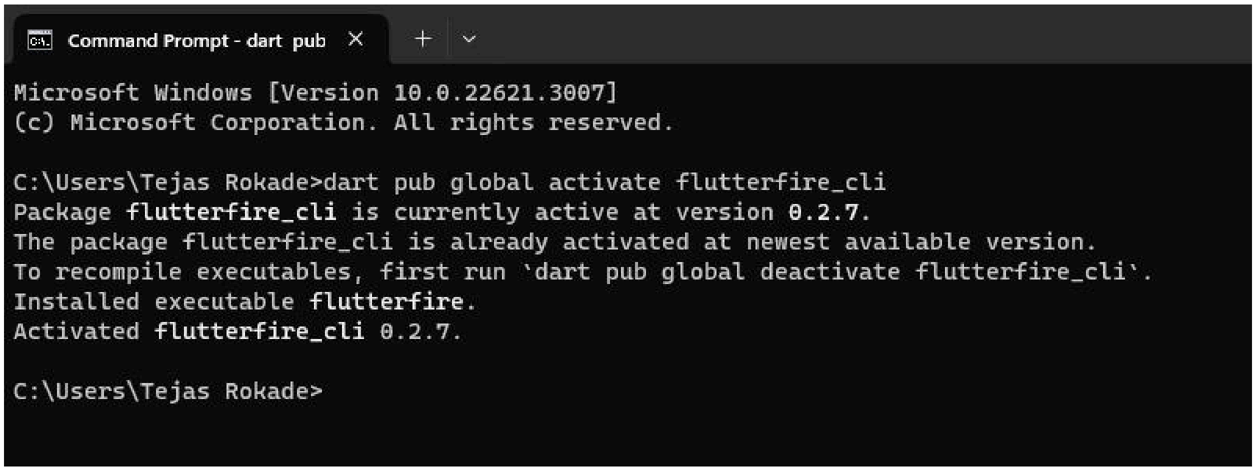
```
C:\Users\Tejas Rokade>dart pub global activate flutterfire_cli
  ╔════════════════════════════════════════════════════════════════════════════╗
  ║ The Dart tool uses Google Analytics to report feature usage statistics     ║
  ║ and to send basic crash reports. This data is used to help improve the     ║
  ║ Dart platform and tools over time.                                         ║
  ║                                                                            ║
  ║ To disable reporting of analytics, run:                                    ║
  ║                                                                            ║
  ║   dart --disable-analytics                                                 ║
  ║                                                                            ║
  ╚════════════════════════════════════════════════════════════════════════════╝

+ ansi_styles 0.3.2+1s... (3.1s)
+ args 2.4.2
+ async 2.11.0
+ boolean_selector 2.1.1
+ characters 1.3.0
+ ci 0.1.0
+ cli_util 0.3.5 (0.4.1 available)
+ clock 1.1.1
+ collection 1.18.0
+ dart_console 1.2.0
+ deep_pick 0.10.0 (1.0.0 available)
```



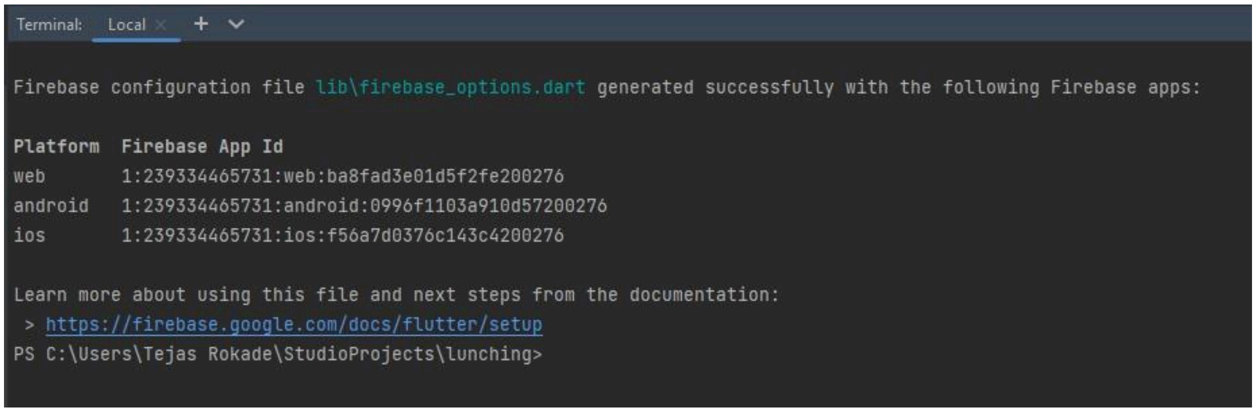
```
+ test_api 0.7.0
+ tint 2.0.1
+ typed_data 1.3.2
+ uri 1.0.0
+ win32 5.2.0
+ xml 6.5.0
+ yaml 3.1.2
Building package executables... (11.7s)
Built flutterfire_cli:flutterfire.
Installed executable flutterfire.
Warning: Pub installs executables into C:\Users\Tejas Rokade\AppData\Local\Pub\Cache\bin, which is not on your path.
You can fix that by adding that directory to your system's "Path" environment variable.
A web search for "configure windows path" will show you how.
Activated flutterfire_cli 0.2.7.

C:\Users\Tejas Rokade>
```



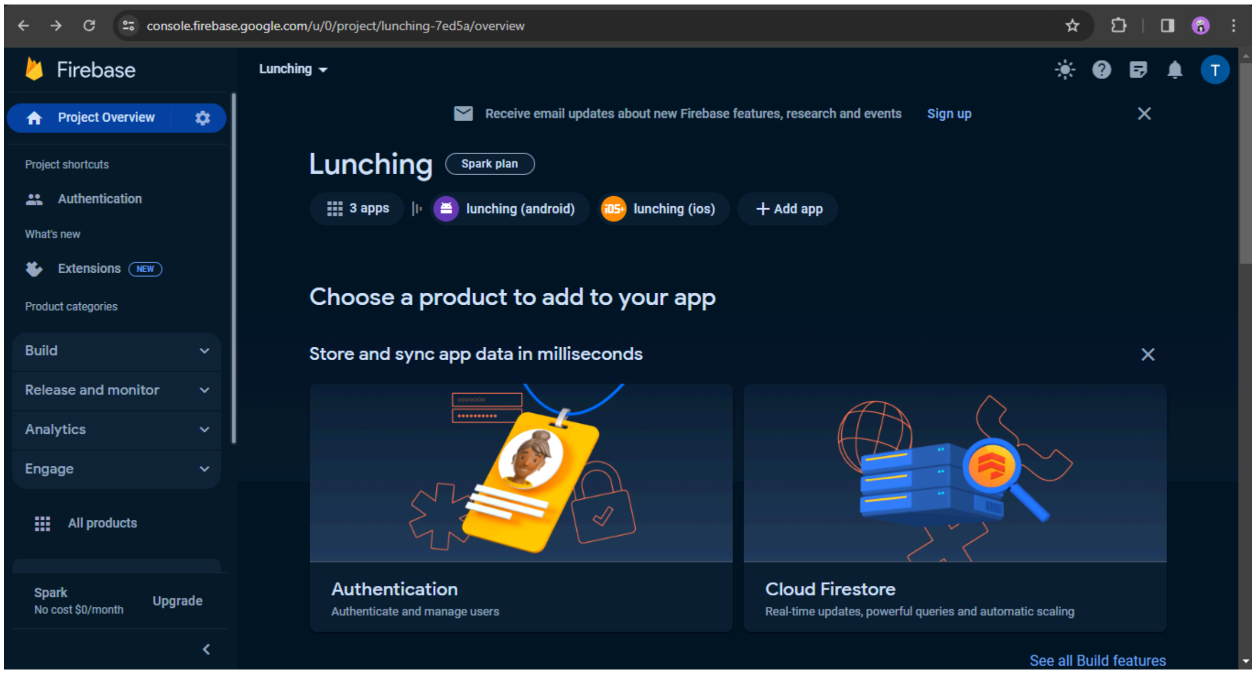
```
C:\Users\Tejas Rokade\AppData\Local\Programs\Python\Python311\S...
C:\Users\Tejas Rokade\AppData\Local\Programs\Python\Python311\
C:\Program Files\MySQL\MySQL Shell 8.0\bin\
C:\Users\Tejas Rokade\AppData\Local\Programs\Python\Python310\S...
C:\Users\Tejas Rokade\AppData\Local\Programs\Python\Python310\
%USERPROFILE%\AppData\Local\Microsoft\WindowsApps
C:\Program Files (x86)\Nmap
C:\Users\Tejas Rokade\AppData\Local\Programs\Microsoft VS Code\b...
%IntelliJ IDEA%
C:\Users\Tejas Rokade\Desktop\my coding trials\java jdk\bin
C:\ghcup\bin
%IntelliJ IDEA Community Edition%
C:\Users\Tejas Rokade\AppData\Roaming\npm
C:\terraform
C:\src\flutter_windows_3.16.6-stable\flutter\bin
C:\Users\Tejas Rokade\AppData\Local\Pub\Cache\bin
```

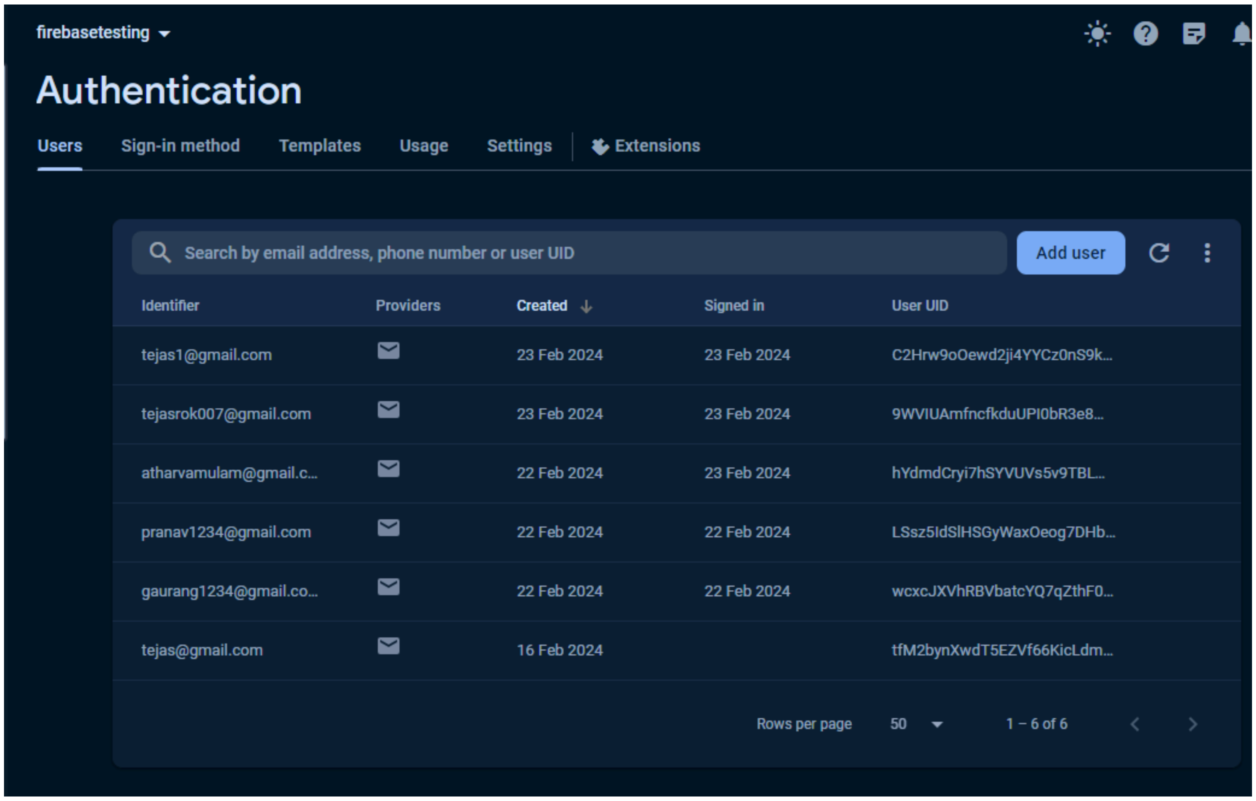


```
Microsoft Windows [Version 10.0.22621.3007]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Tejas Rokade>dart pub global activate flutterfire_cli
Package flutterfire_cli is currently active at version 0.2.7.
The package flutterfire_cli is already activated at newest available version.
To recompile executables, first run `dart pub global deactivate flutterfire_cli`.
Installed executable flutterfire.
Activated flutterfire_cli 0.2.7.

C:\Users\Tejas Rokade>
```



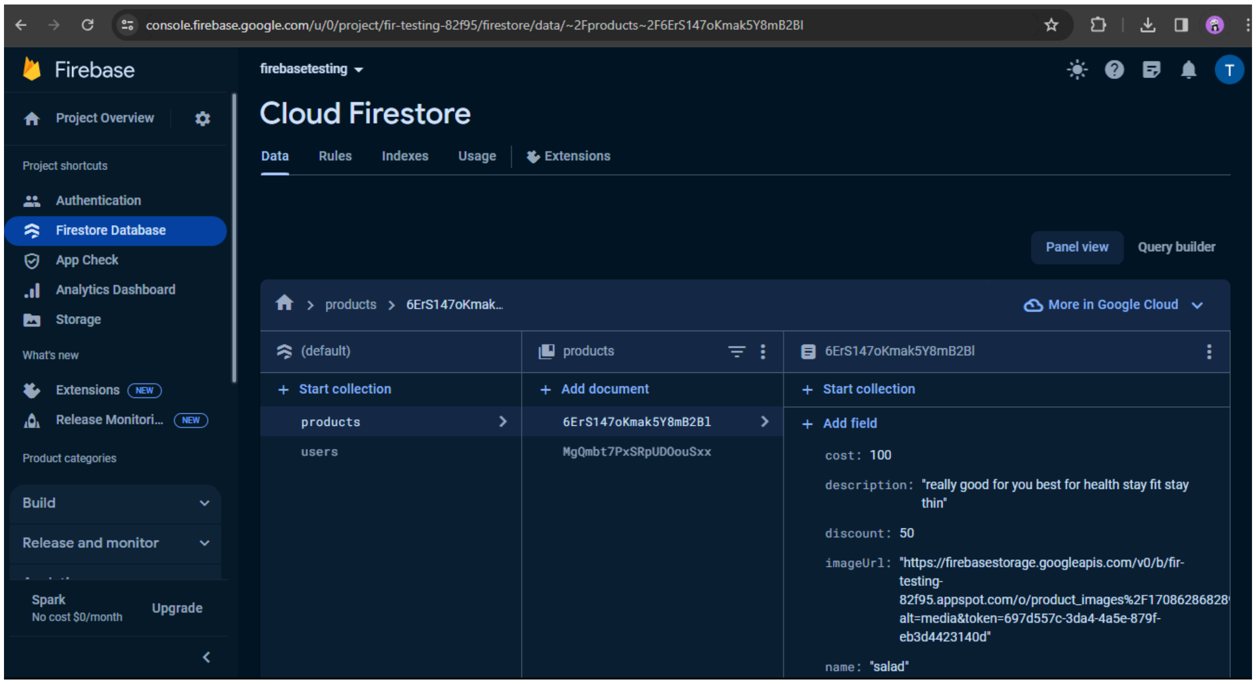
```
Firebase configuration file lib\firebase_options.dart generated successfully with the following Firebase apps:

Platform  Firebase App Id
web       1:239334465731:web:ba8fad3e01d5f2fe200276
android   1:239334465731:android:0996f1103a910d57200276
ios       1:239334465731:ios:f56a7d0376c143c4200276

Learn more about using this file and next steps from the documentation:
 > https://firebase.google.com/docs/flutter/setup
PS C:\Users\Tejas Rokade\StudioProjects\lunching>
```



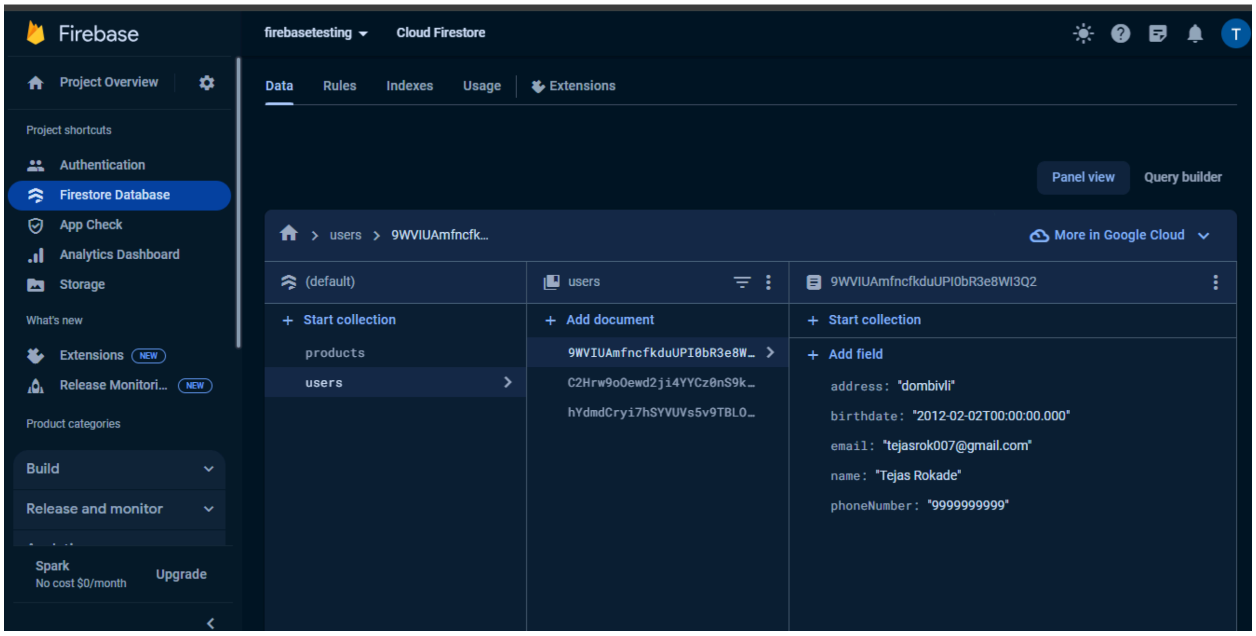


Identifier	Providers	Created	Signed in	User UID
tejas1@gmail.com	✉	23 Feb 2024	23 Feb 2024	C2Hrw9oOewd2ji4YYCz0nS9k...
tejasrok007@gmail.com	✉	23 Feb 2024	23 Feb 2024	9WVIUAmfncfkduUPI0bR3e8...
atharvamulam@gmail.c...	✉	22 Feb 2024	23 Feb 2024	hYdmdCryi7hSYVUVs5v9TBL...
pranav1234@gmail.com	✉	22 Feb 2024	22 Feb 2024	LSsz5IdSlHSGyWaxOeog7DHb...
gaurang1234@gmail.co...	✉	22 Feb 2024	22 Feb 2024	wcxcJXVhRBVbatcYQ7qZthF0...
tejas@gmail.com	✉	16 Feb 2024		tfM2bynXwdT5EZVf66KicLdm...

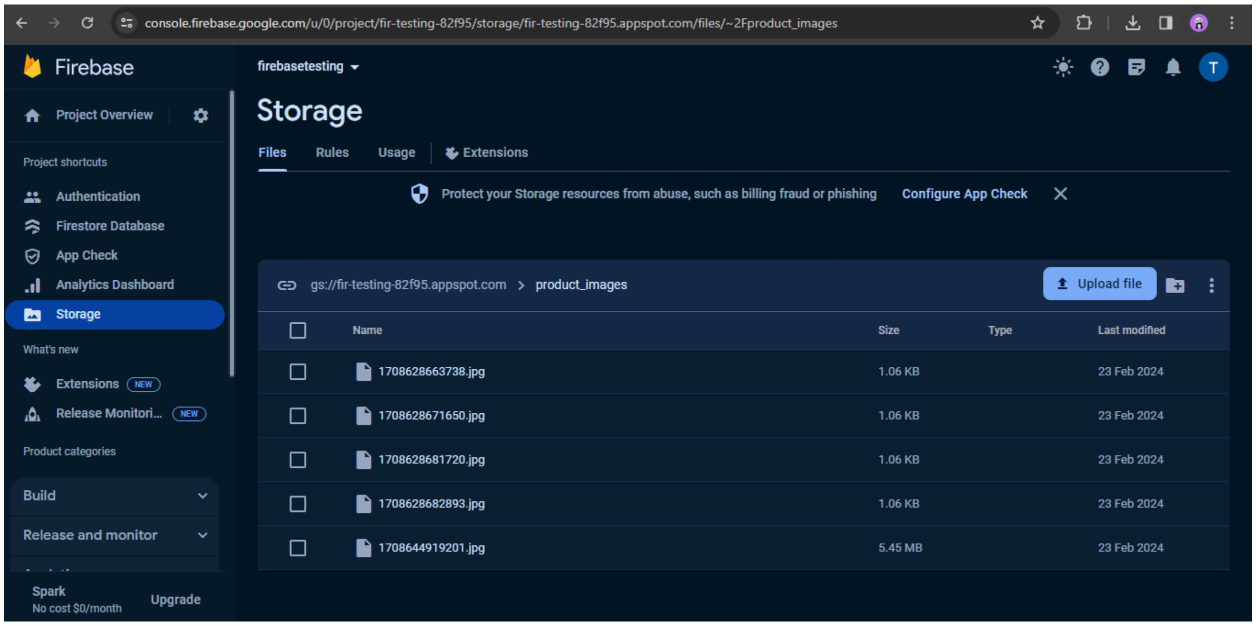
Rows per page 50 1 – 6 of 6



```
cost: 100
description: "really good for you best for health stay fit stay thin"
discount: 50
imageUrl: "https://firebasestorage.googleapis.com/v0/b/fir-testing-82f95.appspot.com/o/product_images%2F17086286828
alt=media&token=697d557c-3da4-4a5e-879f-eb3d4423140d"
name: "salad"
```



(default)	users	9WVIUAmfncfkduUPI0bR3e8WI3Q2
products	9WVIUAmfncfkduUPI0bR3e8W...	address: "dombivli"
users	C2Hrw9oOewd2j14YYCz0nS9k...	birthdate: "2012-02-02T00:00:00.000"
	hYdmdCryi7hSYVUVs5v9TBLO...	email: "tejasrok007@gmail.com"
		name: "Tejas Rokade"
		phoneNumber: "9999999999"

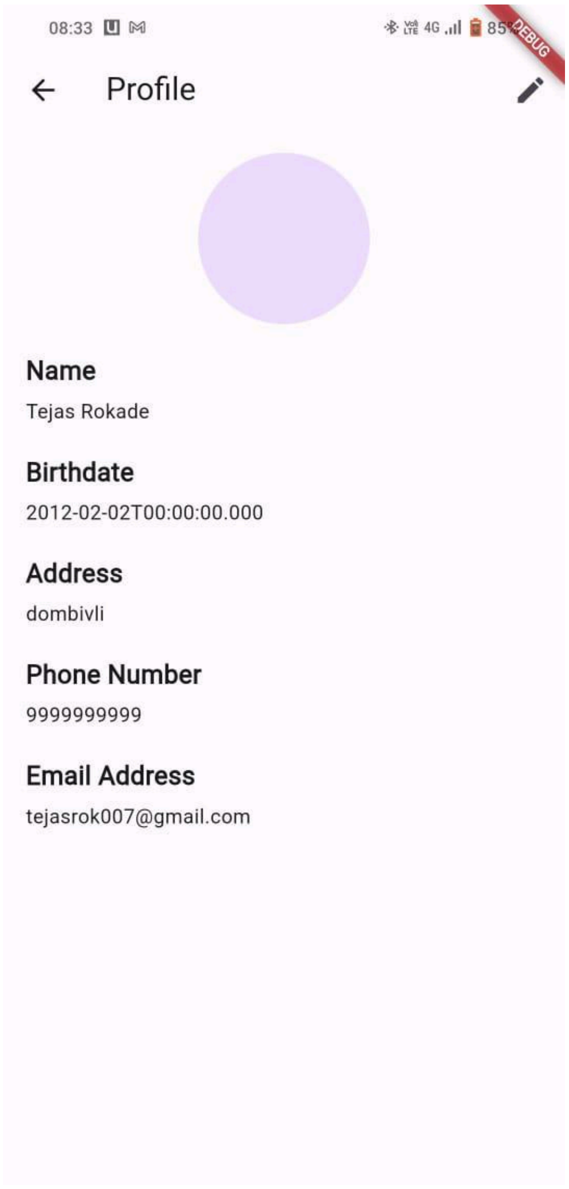


Name	Size	Type	Last modified
1708628663738.jpg	1.06 KB		23 Feb 2024
1708628671650.jpg	1.06 KB		23 Feb 2024
1708628681720.jpg	1.06 KB		23 Feb 2024
1708628682893.jpg	1.06 KB		23 Feb 2024
1708644919201.jpg	5.45 MB		23 Feb 2024

Code -

```
void main() async {
 WidgetsFlutterBinding.ensureInitialized();
 await Firebase.initializeApp(options: DefaultFirebaseOptions.currentPlatform,);
 runApp(MyApp());
}
class loginscreen extends StatefulWidget {
 const loginscreen({Key? key}) : super(key: key);
 @override
 State<loginscreen> createState() => _loginscreenState();
}
```

```
class _loginscreenState extends State<loginscreen> {
 final TextEditingController _emailController = TextEditingController();
 final TextEditingController _passwordController = TextEditingController();
 @override
 void dispose(){
   _emailController.dispose();
   _passwordController.dispose();
   super.dispose();
 }
 void loginUser() async{
   // Implementation of loginUser method
   String res = await AuthMethods().loginUser(email: _emailController.text, password: _passwordController.text);
   if(res == 'success'){
     Navigator.of(context as BuildContext).push(
       MaterialPageRoute(
         builder: (context) => HomePage(),
       ),
     );
   }
   else{
     // Show authentication failed message
     ScaffoldMessenger.of(context).showSnackBar(
       SnackBar(
         content: Text('Authentication failed'),
       ),
     );
   }
 }
 void navigateSignup(){
   // Implementation of navigateSignup method
   Navigator.of(context as BuildContext).push(
     MaterialPageRoute(builder: (context) => SignupPage()),
   );
 }
 @override
 Widget build(BuildContext context) {
   return Scaffold(
     body: SafeArea(
       child: Container(
         padding: const EdgeInsets.symmetric(horizontal: 32),
         child: Column(
           crossAxisAlignment: CrossAxisAlignment.center,
           children: [
             const Text(
               'Lunching',
               style: TextStyle(
                 fontSize: 24,
                 fontWeight: FontWeight.bold,
                 color: Colors.red,
               ),
             ),
             const SizedBox(height: 20),
             // Image.network(
             //   imageUrl,
             //   height: 150, // Adjust height as needed
             //   width: double.infinity,
             //   fit: BoxFit.cover,
             // ),
             Image.asset('assets/card_image_1.jpeg',height: 200,width:double.infinity ,fit: BoxFit.cover,),
             const SizedBox(height: 24),
             InputTextField(
               textEditingController: _emailController,
               hintText: 'Email',
               textInputType: TextInputType.emailAddress,
             ),
             const SizedBox(height: 24),
             InputTextField(
               textEditingController: _passwordController,
               hintText: 'Password',
               textInputType: TextInputType.text,
               isPass: true,
             ),
             const SizedBox(height: 24),
             ElevatedButton(
               onPressed: loginUser,
               style: ElevatedButton.styleFrom(
                 backgroundColor: Colors.green, // Set button color to green
               ),
               child: const Text('Log In'),
             ),
             const SizedBox(height: 20),
             Row(
               mainAxisAlignment: MainAxisAlignment.center,
               children: [
                 Text("Don't have an account?"),
                 GestureDetector(
                   onTap: navigateSignup,
                   child: Text(
                     " Sign up",
                     style: TextStyle(
                       fontWeight: FontWeight.bold,
                       color: Colors.blue, // Keep signup link color blue
                     ),
                   ),
                 )
               ],
             )
           ],
         ),
       ),
     ),
   );
 }
}
class InputTextField extends StatelessWidget {
 final TextEditingController textEditingController;
 final String hintText;
 final TextInputType textInputType;
 final bool isPass;

 const InputTextField({
   Key? key,
   required this.textEditingController,
   required this.hintText,
   required this.textInputType,
   this.isPass = false,
 }) : super(key: key);

 @override
 Widget build(BuildContext context) {
   return TextField(
     controller: textEditingController,
     keyboardType: textInputType,
     obscureText: isPass,
     decoration: InputDecoration(
       hintText: hintText,
       border: OutlineInputBorder(),
     ),
   );
 }
}
class DatabaseMethods {
 Future<void> addUserDetail(Map<String, dynamic> userInfoMap, String id) async {
   return await FirebaseFirestore.instance
       .collection('users')
       .doc(id)
       .set(userInfoMap);
 }
}
class SignupPage extends StatefulWidget {
 @override
 _SignupPageState createState() => _SignupPageState();
}
class _SignupPageState extends State<SignupPage> {
 final FirebaseAuth _auth = FirebaseAuth.instance;
 final DatabaseMethods _databaseMethods = DatabaseMethods();
 final TextEditingController _emailController = TextEditingController();
 final TextEditingController _passwordController = TextEditingController();
 final TextEditingController _nameController = TextEditingController();
 final TextEditingController _addressController = TextEditingController();
 final TextEditingController _phoneNumberController = TextEditingController();
 DateTime? _selectedDate;
 Future<void> _signUpWithEmailAndPassword() async {
   try {
     final UserCredential userCredential = await _auth.createUserWithEmailAndPassword(
       email: _emailController.text.trim(),
       password: _passwordController.text.trim(),
     );
     await _databaseMethods.addUserDetail({
       'email': _emailController.text.trim(),
       'name': _nameController.text.trim(),
       'address': _addressController.text.trim(),
       'phoneNumber': _phoneNumberController.text.trim(),
       'birthdate': _selectedDate != null ? _selectedDate!.toIso8601String() : null,
     }, userCredential.user!.uid);
   } catch (e) {
     print("Failed to register with email and password: $e");
     showDialog(
       context: context,
       builder: (BuildContext context) {
         return AlertDialog(
           title: Text("Registration Error"),
           content: Text("Failed to register with email and password."),
           actions: <Widget>[
             TextButton(
               child: Text("OK"),
               onPressed: () {
                 Navigator.of(context).pop();
               },
             ),
           ],
         );
       },
     );
   }
 }
 Future<void> _selectDate(BuildContext context) async {
   final DateTime? picked = await showDatePicker(
     context: context,
     initialDate: DateTime.now(),
     firstDate: DateTime(1900),
     lastDate: DateTime.now(),
   );
   if (picked != null && picked != _selectedDate)
     setState(() {
       _selectedDate = picked;
     });
 }
 @override
 Widget build(BuildContext context) {
   return Scaffold(
     appBar: AppBar(
       title: Text("Signup"),
     ),
     body: SingleChildScrollView(
       child: Padding(
         padding: EdgeInsets.all(16.0),
         child: Column(
           crossAxisAlignment: CrossAxisAlignment.center,
           mainAxisAlignment: MainAxisAlignment.center,
           children: <Widget>[
             TextField(
               controller: _emailController,
               decoration: InputDecoration(
                 labelText: 'Email',
                 border: OutlineInputBorder(
                   borderSide: BorderSide(color: Colors.lightBlue),
                 ),
               ),
             ),
             SizedBox(height: 16.0),
             TextField(
               controller: _passwordController,
               obscureText: true,
               decoration: InputDecoration(
                 labelText: 'Password',
                 border: OutlineInputBorder(
                   borderSide: BorderSide(color: Colors.lightBlue),
                 ),
               ),
             ),
             SizedBox(height: 16.0),
             TextField(
               controller: _nameController,
               decoration: InputDecoration(
                 labelText: 'Name',
                 border: OutlineInputBorder(
                   borderSide: BorderSide(color: Colors.lightBlue),
                 ),
               ),
             ),
             SizedBox(height: 16.0),
             TextField(
               controller: _addressController,
               decoration: InputDecoration(
                 labelText: 'Address',
                 border: OutlineInputBorder(
                   borderSide: BorderSide(color: Colors.lightBlue),
                 ),
               ),
             ),
             SizedBox(height: 16.0),
             TextField(
               controller: _phoneNumberController,
               decoration: InputDecoration(
                 labelText: 'Phone Number',
                 border: OutlineInputBorder(
                   borderSide: BorderSide(color: Colors.lightBlue),
                 ),
               ),
             ),
             SizedBox(height: 16.0),
             GestureDetector(
               onTap: () => _selectDate(context),
               child: AbsorbPointer(
                 child: TextFormField(
                   decoration: InputDecoration(
                     labelText: 'Birthdate',
                     border: OutlineInputBorder(
                       borderSide: BorderSide(color: Colors.lightBlue),
                                 ),
                                 suffixIcon: Icon(Icons.calendar_today),
                               ),
                               controller: TextEditingController(
                                 text: _selectedDate != null
                                     ? '${_selectedDate!.day}/${_selectedDate!.month}/${_selectedDate!.year}'
                                     : '',
                               ),
                             ),
                           ),
                         ),
                         SizedBox(height: 32.0),
                         ElevatedButton(
                           onPressed: _signUpWithEmailAndPassword,
                           style: ElevatedButton.styleFrom(
                             foregroundColor: Colors.white, backgroundColor: Colors.lightGreen,
                           ),
                           child: Text('Sign Up'),
                         ),
                       ],
                     ),
                   ),
                 ),
               );
              }
             }

             class NavBar extends StatelessWidget {
              const NavBar({Key? key}) : super(key: key);

              @override
              Widget build(BuildContext context) {
                return Drawer(
                  child: ListView(
                    padding: EdgeInsets.zero,
                    children: [
                      StreamBuilder(
                        stream: FirebaseAuth.instance.authStateChanges(),
                        builder: (BuildContext context, AsyncSnapshot<User?> snapshot) {
                          if (snapshot.connectionState == ConnectionState.waiting) {
                            return CircularProgressIndicator(); // Show loading indicator while fetching user data
                          }

                          if (!snapshot.hasData || snapshot.data == null) {
                            return Text('No user signed in'); // Handle the case where no user is signed in
                          }

                          String? userId = snapshot.data!.uid;

                          return StreamBuilder(
                            stream: FirebaseFirestore.instance
                                .collection('users')
                                .doc(userId)
                                .snapshots(),
                   builder: (BuildContext context, AsyncSnapshot<DocumentSnapshot> snapshot) {
                     if (snapshot.connectionState == ConnectionState.waiting) {
                       return CircularProgressIndicator(); // Show loading indicator while fetching data
                     }

                     if (snapshot.hasError) {
                       return Text('Error: ${snapshot.error}');
                     }

                     if (!snapshot.hasData || !snapshot.data!.exists) {
                       return Text('No data found'); // Handle the case where user data is not available
                     }

                     // Access user data from snapshot
                     Map<String, dynamic> userData = snapshot.data!.data() as Map<String, dynamic>;
                     String accountName = userData['name'] ?? ''; // Get account name from user data
                     String accountEmail = userData['email'] ?? ''; // Get account email from user data
                     return UserAccountsDrawerHeader(
                       accountName: Text(accountName),
                       accountEmail: Text(accountEmail),
                       currentAccountPicture: CircleAvatar(
                         backgroundImage: AssetImage('assets/Profile_image_1.jpg'),
                       ),
                       decoration: BoxDecoration(
                         color: Colors.blueAccent,
                         // Add background image if needed
                         image: DecorationImage(
                           image: AssetImage('assets/background_image_1.jpg'),
                           fit: BoxFit.cover,
                         ),
                       ),
                     );
                   },
                 );
               },
             ),
             ListTile(
               leading: Icon(Icons.person),
               title: Text('Add Food Product'),
               onTap: () {
                 Navigator.of(context).push(
                   MaterialPageRoute(builder: (context) => AddProductPage()),
                 );
               },
             ),
             ListTile(
               leading: Icon(Icons.person),
               title: Text('Profile'),
               onTap: () {
                 Navigator.of(context).push(
                   MaterialPageRoute(builder: (context) => ProfilePage()),
                 );
               },
             ),
                       ListTile(
                         leading: Icon(Icons.person),
                         title: Text('Cart'),
                         onTap: () {
                           Navigator.of(context).push(
                             MaterialPageRoute(builder: (context) => CartPage()),
                           );
                         },
                       ),
                       Divider(),
                       ListTile(
                         leading: Icon(Icons.person),
                         title: Text('Products Added'),
                         onTap: () {
                           Navigator.of(context).push(
                             MaterialPageRoute(builder: (context) => ProductList()),
                           );
                         },
                       ),
                       ListTile(
                         leading: Icon(Icons.logout),
                         title: Text('Logout'),
                         onTap: () {
                           // Logout functionality
                           // Example:
                           // Navigator.of(context).pop(); // Close drawer
                           // Navigator.of(context).pushReplacement(
                           //   MaterialPageRoute(builder: (context) => LoginScreen()),
                           // );
                         },
                       ),
                     ],
                   ),
                 );
               }
             }
```



Name

Tejas Rokade

Birthdate

2012-02-02T00:00:00.000

Address

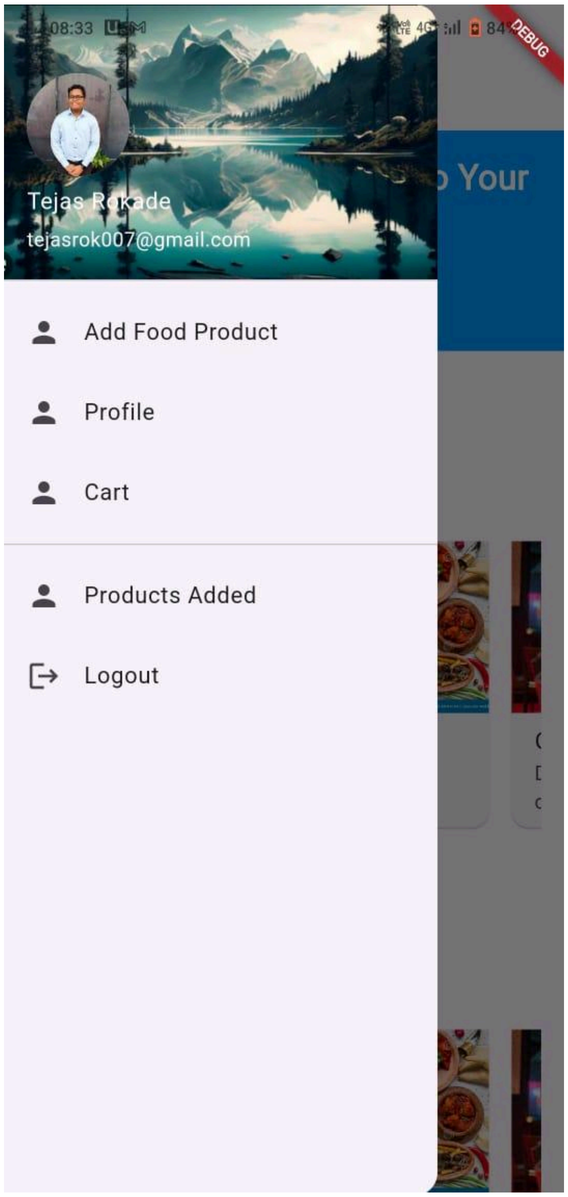
dombivli

Phone Number

9999999999

Email Address

tejasrok007@gmail.com



Tejas Rokade

tejasrok007@gmail.com

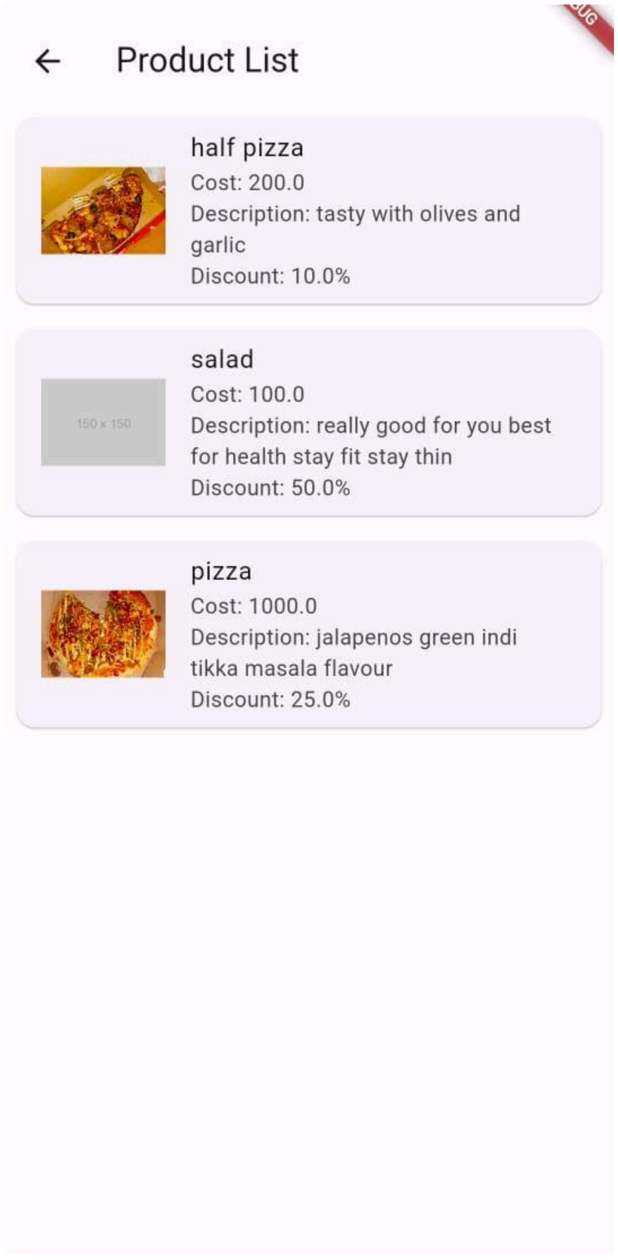
Add Food Product

Profile

Cart

Products Added

Logout



Product List

half pizza
Cost: 200.0
Description: tasty with olives and garlic
Discount: 10.0%

salad
Cost: 100.0
Description: really good for you best for health stay fit stay thin
Discount: 50.0%

pizza
Cost: 1000.0
Description: jalapenos green indi tikka masala flavour
Discount: 25.0%

Conclusion -

Through this experiment we understood implementation of navigator , haptics and gestures for implementing navigation from one tab to another tab as per need and gestures to recognise and for events to occur in the flutter application.