

```
flutter pub add image_picker
```

```
flutter pub add firebase_storage
```

```
flutter build apk --release
```

main.dart

```
1 void main() async {
2     WidgetsFlutterBinding.ensureInitialized();
3
4     await Firebase.initializeApp(
5         options: DefaultFirebaseOptions.currentPlatform,
6     );
7
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 const MaterialApp(
17             debugShowCheckedModeBanner: false,
18             home: HomeScreen(),
19         );
20     }
21 }
```

home_screen.dart

image #1

```
1
2 class HomeScreen extends StatefulWidget {
3   const HomeScreen({super.key});
4
5   @override
6   State<HomeScreen> createState() => _HomeScreenState();
7 }
```

image #2

```
1 class _HomeScreenState extends State<HomeScreen> {
2   final TextEditingController controller = TextEditingController();
3
4   List<String> imageUrl = [];
5
6   @override
7   Widget build(BuildContext context) {
8     return SafeArea(
9       child: Scaffold(
10         body: SingleChildScrollView(
11           child: Padding(
12             padding: const EdgeInsets.all(20),
13             child: Column(
14               children: [

```

image #3

```
1 // enter user id
2 TextFormField(
3   controller: controller,
4   decoration: InputDecoration(
5     labelText: 'Enter user id',
6     border: OutlineInputBorder(
7       borderRadius: BorderRadius.circular(10),
8     ),
9   ),
10  keyboardType: TextInputType.number,
11 ),
12
13 const SizedBox(height: 40),
```

image #4

```
● ● ●  
1 // buttons  
2 Row(  
3   mainAxisAlignment: MainAxisAlignment.spaceEvenly,  
4   children: [  
5     // upload button  
6     ElevatedButton(  
7       onPressed: () async {  
8         await uploadImage(controller.text);  
9       },  
10      child: const Text('Upload image'),  
11    ),  
12  
13     // retrieve button  
14     ElevatedButton(  
15       onPressed: () {  
16         retrieveImages(controller.text);  
17       },  
18       child: const Text('Retrieve image'),  
19     ),  
20   ],  
21 ),  
22  
23 const SizedBox(height: 40),
```

image #5

```
1          // images list
2      if (imageUrls.isNotEmpty)
3          ListView.separated(
4              shrinkWrap: true,
5              physics: const NeverScrollableScrollPhysics(),
6              itemBuilder: (context, index) {
7                  return Container(
8                      height: 250,
9                      width: double.infinity,
10                     decoration: BoxDecoration(
11                         image: DecorationImage(
12                             image: NetworkImage(
13                                 imageUrls[index],
14                             ),
15                             fit: BoxFit.cover,
16                         ),
17                         borderRadius: BorderRadius.circular(10),
18                     ),
19                 );
20             },
21             separatorBuilder: (context, index) {
22                 return const SizedBox(height: 20);
23             },
24             itemCount: imageUrls.length,
25         ),
26         ],
27     ),
28     ),
29     ),
30     ),
31 );
32 }
```

image #6



A screenshot of a mobile application interface, likely an emulator or developer tool, displaying a Dart code snippet. The code is for an asynchronous function named `uploadImage` that takes a `String userId` parameter. It first checks if the `userId` is empty and returns if it is. It then creates an `ImagePicker` object and uses it to pick an image from the gallery. If a file is picked, it reads the file as bytes and creates a reference to the Firebase Storage instance. The reference is then used to upload the file to a specific path based on the `userId` and the file's path. Finally, it calls a `retrieveImages` method and prints any errors that occur during the process.

```
1 Future<void> uploadImage(String userId) async {
2     if (userId.isEmpty) {
3         return;
4     }
5     final picker = ImagePicker();
6     final pickedFile = await picker.pickImage(source: ImageSource.gallery);
7
8     if (pickedFile != null) {
9         final file = await pickedFile.readAsBytes();
10
11     final ref = FirebaseStorage.instance
12         .ref('${userId}/${pickedFile.path.split('/').last}');
13
14     ref.putData(file).then(
15         (_)
16             async {
17                 await retrieveImages(controller.text);
18             },
19         ).catchError(
20             (error) {
21                 print('Error occurred: $error');
22             },
23         );
24     }
25 }
```

image #7



A screenshot of a mobile application interface showing Dart code. The code is a function named `retrieveImages` that takes a `String userId` parameter. It first checks if the `userId` is empty and returns if it is. Then it clears the `imageUrls` list. It then creates a `FirebaseStorageReference` object and uses `listAll()` to get a `ListResult`. It iterates over the items in the `listResult.items` list, calling `getDownloadURL()` on each reference to get the URL, adding it to the `imageUrls` list, and then setting the state. If an error occurs, it prints the error message. The code is numbered from 1 to 22.

```
1 Future<void> retrieveImages(String userId) async {
2     try {
3         if (userId.isEmpty) {
4             return;
5         }
6
7         imageUrls.clear();
8
9         final firebaseStorageReference = FirebaseStorage.instance.ref(userId);
10        final ListResult listResult = await firebaseStorageReference.listAll();
11
12        for (var ref in listResult.items) {
13            String url = await ref.getDownloadURL();
14            setState(() {
15                imageUrls.add(url);
16            });
17        }
18    } catch (e) {
19        print('Error occurred: $e');
20    }
21 }
22 }
```

Whole code in one image 

لا تنسونا من صالح الداعع

```
•••
import 'package:firebase_storage/firebase_storage.dart';
import 'package:flutter/material.dart';
import 'package:image_picker/image_picker.dart';
4
5 class HomeScreen extends StatefulWidget {
6   const HomeScreen({super.key});
7
8   @override
9   State<HomeScreen> createState() => _HomeScreenState();
10 }
11
12 class _HomeScreenState extends State<HomeScreen> {
13   final TextEditingController controller = TextEditingController();
14
15   List<String> imageUrls = [];
16
17   @override
18   void build(BuildContext context) {
19     return SafeArea(
20       child: Scaffold(
21         body: SingleChildScrollView(
22           child: Padding(
23             padding: EdgeInsets.all(20),
24             child: Column(
25               children: [
26                 // enter user id
27                 TextFormField(
28                   controller: controller,
29                   decoration: InputDecoration(
30                     labelText: 'Enter user id',
31                     border: OutlineInputBorder(
32                       borderRadius: BorderRadius.circular(10),
33                     ),
34                   ),
35                   keyboardType: TextInputType.number,
36                 ),
37
38                 const SizedBox(height: 40),
39
40                 // buttons
41                 Row(
42                   mainAxisAlignment: MainAxisAlignment.spaceEvenly,
43                   children: [
44                     // upload button
45                     ElevatedButton(
46                       onPressed: () async {
47                         await uploadImage(controller.text);
48                         child: const Text('Upload image'),
49                     ),
50
51                     // retrieve button
52                     ElevatedButton(
53                       onPressed: () {
54                         retrieveImages(controller.text);
55                         child: const Text('Retrieve image'),
56                     ),
57
58                     ],
59                 ),
60
61                 const SizedBox(height: 40),
62
63                 // image
64                 if (imageUrls.isNotEmpty)
65                   ListView.separated(
66                     shrinkWrap: true,
67                     physics: const NeverScrollableScrollPhysics(),
68                     itemCount: (context, index) {
69                       return Container(
70                         height: 250,
71                         width: double.infinity,
72                         decoration: BoxDecoration(
73                           image: DecorationImage(
74                             image: NetworkImage(
75                               imageUrls[index],
76                             ),
77                           fit: BoxFit.cover,
78                           ),
79                           borderRadius: BorderRadius.circular(10),
80                         ),
81                       );
82                     },
83                     separatorBuilder: (context, index) {
84                       return const SizedBox(width: 20);
85                     },
86                     itemCount: imageUrls.length,
87                   ),
88
89                 ],
90               ),
91             ),
92           ),
93         ),
94       ),
95     );
96   }
97
98   Future<void> uploadImage(String userId) async {
99     if (userId.isEmpty) {
100       return;
101     }
102     final picker = ImagePicker();
103     final pickedFile = await picker.pickImage(source: ImageSource.gallery);
104
105     if (pickedFile != null) {
106       final file = await pickedFile.readAsBytes();
107
108       final ref = FirebaseStorage.instance
109         .ref('$userId${pickedFile.path.split('/').last}');
110
111       ref.putBytes(file).then(
112         () async {
113           await retrieveImages(controller.text);
114         },
115       ).catchError(
116         (error) {
117           print('Error occurred: $error');
118         },
119       );
120     }
121   }
122
123   Future<void> retrieveImages(String userId) async {
124     try {
125       if (userId.isEmpty) {
126         return;
127       }
128       imageUrls.clear();
129
130       final firebaseStorageReference = FirebaseStorage.instance.ref(userId);
131       final listResult listResult = await firebaseStorageReference.listAll();
132
133       for (var ref in listResult.items) {
134         String url = await ref.getDownloadURL();
135         setState(() {
136           imageUrls.add(url);
137         });
138       }
139     } catch (e) {
140       print('Error occurred: $e');
141     }
142   }
143 }
```