

MPL Experiment 3

Name: Prathamesh Palve

Class: D15A

Roll no:31

Aim: To include icons, images, Fonts in Flutter app.

Theory:

Incorporating icons, images, and custom fonts in a Flutter application enhances the visual appeal and improves the user experience. Flutter provides different ways to add these elements:

1. Icons:

Icons can be added using the built-in `Icons` class with the `Icon` widget. Custom icons can also be used via the `flutter_launcher_icons` package.

2. Images:

Images can be displayed in two ways:

- **From assets:** Images stored locally in the app's assets folder can be loaded using the `Image.asset()` method.
- **From the internet:** Images can be fetched dynamically from a URL using the `Image.network()` method.

Examples:

- **Loading an image from assets:**

```
Image.asset(  
  
  'assets/images/sample.png'  
  
  , fit: BoxFit.cover,  
  
  width: double.infinity,  
  
)
```

- **Loading an image from a URL:**

```
Image.network(  
  
  'https://example.com/sample.jpg',  
  
)
```

3. Fonts:

)

Custom fonts improve typography and branding. To add custom fonts, font files must be placed in the `assets/fonts/` directory and declared in `pubspec.yaml` under the `flutter` section. Using `TextStyle` with the `fontFamily` property applies the custom font to text elements.

Steps:

Step 1: Create an `assets` folder inside the project directory. Inside the `assets` folder, create an `images` folder and add the required images.

Step 2: Open `pubspec.yaml` and add the following under the `flutter` section:

`flutter:`

```
name: chatgpt_clone_08
description: "A new Flutter project."

publish_to: 'none'

version: 1.0.0+1

environment:
  sdk: ">=3.0.0 <4.0.0"

dependencies:
  flutter:
    sdk: flutter
  cupertino_icons: ^1.0.8
  http: ^1.3.0
  flutter_dotenv: ^5.1.0

  # Firebase Core and Services
  firebase_core: ^2.27.0
  firebase_auth: ^4.17.8
  cloud_firestore: ^4.15.7
  firebase_storage: ^11.6.8 # For storing user profile pictures

  # State Management & Utilities
  shared_preferences: ^2.2.3 # For caching some user data locally

  # UI and Animations
  google_fonts: ^6.2.1 # Custom fonts for better UI
  fluttertoast: ^8.2.4 # Show small notifications
  intl: ^0.19.0 # Date formatting
  lottie: ^3.1.0 # For animations (optional)

dev_dependencies:
  flutter_test:
    sdk: flutter
  flutter_lints: ^3.0.0
```

Step 3: Run the following command in the terminal to apply the changes:

`flutter pub get`

Code:

home.dart

```
import 'package:flutter/material.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:cloud_firestore/cloud_firestore.dart';
import '../services/api_service.dart';
import '../services/firestore_service.dart';
import '../widgets/chat_bubble.dart';
import '../widgets/empty_state.dart';
import '../widgets/bottom_bar.dart';
import 'setting.dart';
import 'left_slidder.dart';

class HomeScreen extends StatefulWidget {
  final String chatId; // Accepts chatId dynamically

  HomeScreen({required this.chatId});

  @override
  _HomeScreenState createState() => _HomeScreenState();
}

class _HomeScreenState extends State<HomeScreen> {
  final TextEditingController _messageController = TextEditingController();
  final FirebaseAuth _auth = FirebaseAuth.instance;
  final FirestoreService _firestoreService = FirestoreService();
  final ApiService _apiService = ApiService();
  final ScrollController _scrollController = ScrollController();

  void _sendMessage() async {
    final user = _auth.currentUser;
    if (user == null || _messageController.text.trim().isEmpty) return;

    String userId = user.uid;
    String chatId = widget.chatId;
    String userMessage = _messageController.text.trim();

    try {
      // Store user message in Firestore
      await _firestoreService.addMessage(userId, chatId, userMessage, true);

      setState(() {
        _messageController.clear();
      });
    };
```

```

// Scroll to bottom after sending message
Future.delayed(Duration(milliseconds: 300), () {
  _scrollToBottom();
});

// Get response from Hugging Face API
String aiResponse = await _apiService.getHuggingFaceResponse(userMessage);

// Store AI response in Firestore
await _firestoreService.addMessage(userId, chatId, aiResponse, false);

print("✅ AI Response stored successfully!");
} catch (e) {
  print("❌ Error sending message: $e");
}
}

void _scrollToBottom() {
  if (_scrollController.hasClients) {
    _scrollController.animateTo(
      0, // Since we reversed ListView, scroll to position 0
      duration: Duration(milliseconds: 300),
      curve: Curves.easeOut,
    );
  }
}

@override
Widget build(BuildContext context) {
  final user = _auth.currentUser;
  if (user == null) {
    return Scaffold(
      body: Center(child: Text("User not logged in")),
    );
  }

  return Scaffold(
    backgroundColor: Colors.black,
    appBar: AppBar(
      backgroundColor: Colors.black,
      elevation: 0,
      leading: Builder(
        builder: (context) => IconButton(
          icon: Icon(Icons.menu, color: Colors.white),
          onPressed: () {

```

```

        Scaffold.of(context).openDrawer();
      },
    ),
  ),
  actions: [
    TextButton(
      onPressed: () {},
      child: Text("Get Plus ✨", style: TextStyle(color: Colors.white)),
    ),
    IconButton(
      icon: Icon(Icons.more_vert, color: Colors.white),
      onPressed: () {},
    ),
  ],
),
drawer: LeftSlider(currentChatId: widget.chatId),
body: Column(
  children: [
    Expanded(
      child: StreamBuilder<QuerySnapshot>(
        stream: _firestoreService.getMessages(user.uid, widget.chatId),
        builder: (context, snapshot) {
          if (snapshot.connectionState == ConnectionState.waiting) {
            return Center(child: CircularProgressIndicator());
          }

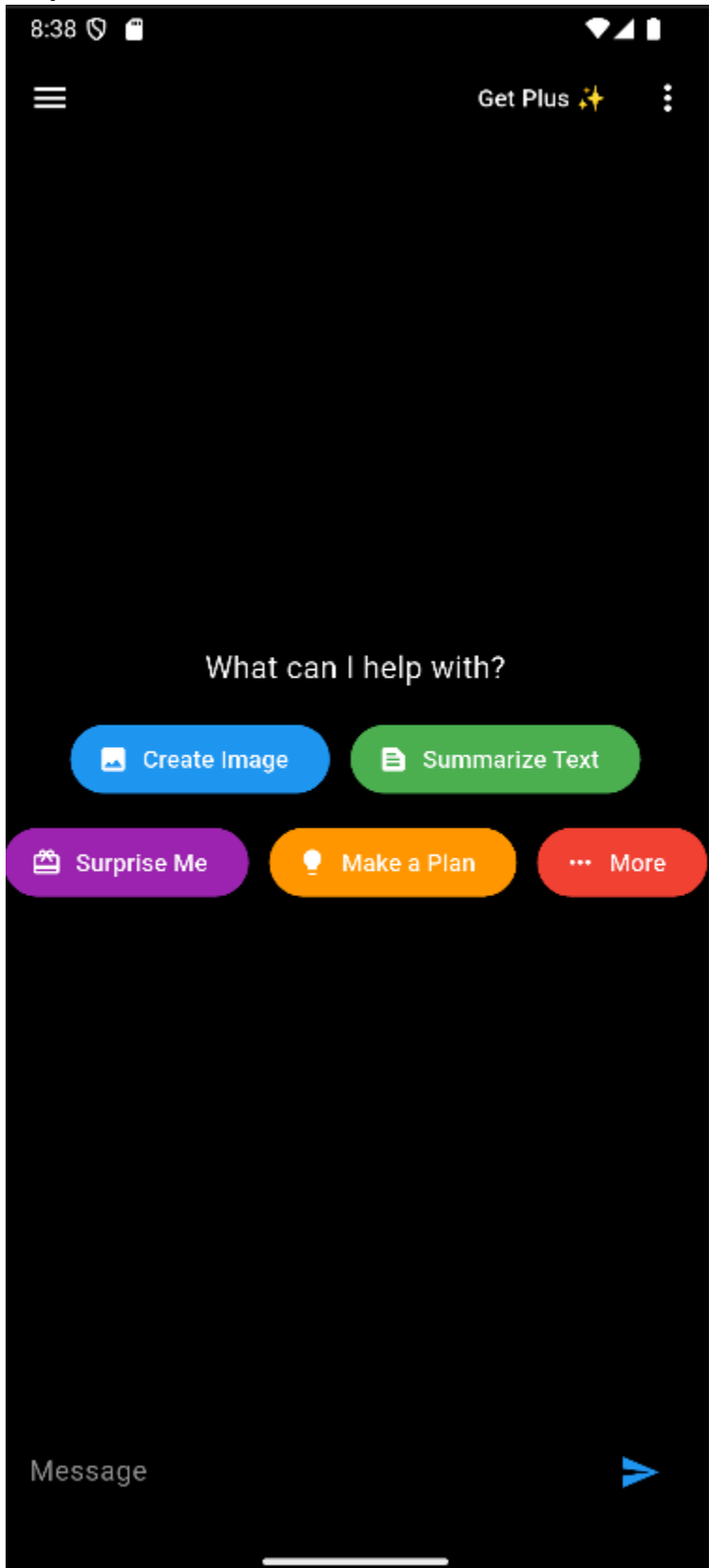
          if (!snapshot.hasData || snapshot.data!.docs.isEmpty) {
            return EmptyState(); // Show empty state if no messages
          }

          return ListView(
            controller: _scrollController,
            padding: EdgeInsets.all(16),
            reverse: true, // 🔥 Messages appear from bottom to top
            children: snapshot.data!.docs.map((doc) {
              Map<String, dynamic> data = doc.data() as Map<String, dynamic>;
              bool isUser = data['isUser'] ?? false;
              return ChatBubble(message: data['message'] ?? '', isUser:
isUser);
            }).toList(),
          );
        },
      ),
    ),
    BottomBar(
      messageController: _messageController,


```

```
        onSendPressed: _sendMessage,  
      ),  
    ],  
  ),  
);  
}  
}
```

Output:



8:38

 Search

Plus  

+ New Chat

Chat Ayf9Xr6i

Chat Apt7dmSK

Chat J4puFKGb

Chat eBsLA3CF

Chat yMVLc8xl

Chat SKzmykxF

Chat qbd3TxjL

Chat qYkdjLZq

Chat ubHnANZF

Chat twe4o0wo

ze Text

... More



prathameshpalve08@
gmail.com



8:38



Settings



prathmzz

prathameshpalve08@gmail.com



Email

prathameshpalve08@gmail.com



Phone number

Not linked



Upgrade to Plus



Customize



Data Controls



Voice



About



Sign out