EX.NO: 1 **GUI COMPONENTS, FONTS AND COLORS**

AIM:

To implement an application that uses GUI components, Font, Colors.

PROCEDURE:

<u>GUI components:</u>

- Scaffold()
  - Creates a visual scaffold for Material Design widgets
  - appBar() id used to specify the title and background of the top bar.
  - body() is used to contain the primary content of the scaffold.
- MaterialApp()
  - contains widgets that are used for the material design of an application.
  - theme property is used to set the theme of the application to dark or light.
  - Home property defines the starting point of the application. It usually contains Scaffold.
- Text():
  - import 'package:*flutter*/material.dart';
  - specify the string to be displayed, withing quotes inside Text().
  - Style property can be used to add TextStyle like fontSize, color.
  - textAlign property can be used for alignment of specified text
- GridView.count()
  - creates a layout with a fixed number of tiles in the cross axis
  - children property is used to specify the widgets to be included in the layout. (Eg: containers)
  - To set spacing between items along main axis or cross axis, set the required double values for mainAxisSpacing property and crossAxisSpacing property respectively
- Container()
  - Helps to create a rectangular visual element.
  - The margin property uses EdgeInsets to set the margin for the four directions (LTRB).
  - Image or icon or text can be included placed inside the container using child parameter:
  - Decoration (BoxDecoration) can be used to give shape, backgroundColor etc. to a container.

<u>Font:</u>

- Style property can be used to add TextStyle like fontSize, color.
- To use google fonts,
  - Install using ‘flutter pub add google_fonts’
  - import 'package:google_fonts/google_fonts.dart';
  - Specify the font name in the style property of Text().
  - textStyle attribute can be used to format the text.
  - style: GoogleFonts.*rockSalt*(textStyle: const TextStyle(color: Colors.*black*,fontSize: 20)

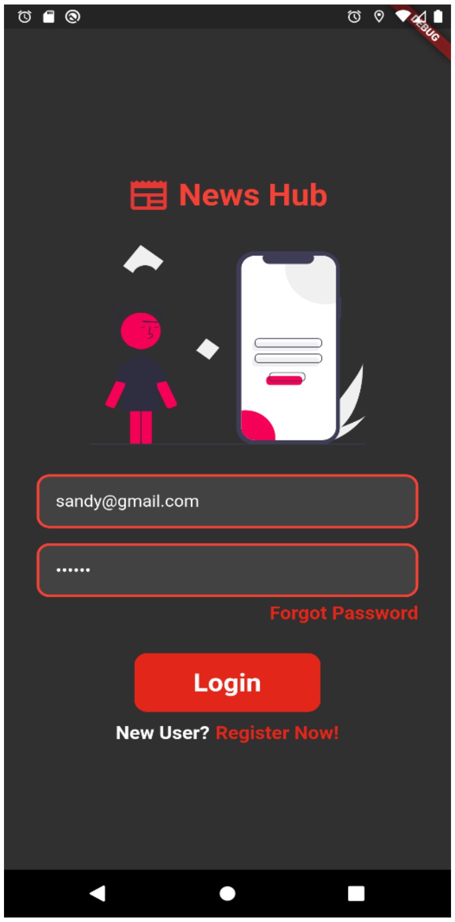
Colors:

- Color property can be used to specify the color using the Colors class.
- It can also be represented in the format of #RRGGBB where RR represents Red color, GG represents the Green color and BB represents the Blue color.

CODE:

```
import 'package:flutter/material.dart';
import 'package:flutter_svg/flutter_svg.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:news_hub/screens/forgot_password.dart';
import 'dart:async';

class LoginPage extends StatefulWidget {
  final VoidCallback showRegisterPage;
  const LoginPage({Key? key, required this.showRegisterPage}) : super(key: key);

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

class _LoginPageState extends State<LoginPage> {
  final _emailController = TextEditingController();
  final _passwordController = TextEditingController();

  Future signIn() async {
    await FirebaseAuth.instance.signInWithEmailAndPassword(
        email: _emailController.text.trim(),
        password: _passwordController.text.trim());
    var snackBar = const SnackBar(
        content: Text(
      "Login Successfull !!",
      style: TextStyle(fontWeight: FontWeight.bold),
    ));
    ScaffoldMessenger.of(context).showSnackBar(snackBar);
  }

  @override
  void dispose() {
    _emailController.dispose();
    _passwordController.dispose();
    super.dispose();
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
backgroundColor: Colors.grey[850],
body: SafeArea(
    child: Center(
  child: SingleChildScrollView(
    child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        const SizedBox(
          height: 20.0,
        ),
        Row(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Icon(Icons.newspaper_rounded,
                color: Colors.red[600], size: 40.0),
            const Text(
              " News Hub",
              style: TextStyle(
                  color: Colors.red,
                  fontWeight: FontWeight.bold,
                  fontSize: 30.0),
            ),
          ],
        ),
        const SizedBox(
          height: 30.0,
        ),
        // PhotoView(
        //   imageProvider: const AssetImage("assets/large-image.jpg"),
        //   customSize: const Size(200, 200),
        // ),
        SvgPicture.asset(
          "assets/images/login_img.svg",
          height: 200.0,
          width: 200.0,
        ),
        const SizedBox(
          height: 30.0,
        ),
        Padding(
          padding: const EdgeInsets.symmetric(horizontal: 30.0),
          child: Container(
            decoration: BoxDecoration(
                color: Colors.grey[800],
                border: Border.all(color: Colors.red, width: 2.5),
                borderRadius: BorderRadius.circular(12)),
            child: TextField(
      controller: _emailController,
      decoration: const InputDecoration(
          border: InputBorder.none,
          hintText: "Email",
          hintStyle: TextStyle(
              color: Color.fromARGB(255, 197, 182, 182)),
          contentPadding: EdgeInsets.all(15.0)),
      style: const TextStyle(color: Colors.white),
    ),
  ),
),
const SizedBox(
  height: 15.0,
),
Padding(
  padding: const EdgeInsets.symmetric(horizontal: 30.0),
  child: Container(
    decoration: BoxDecoration(
        color: Colors.grey[800],
        border: Border.all(color: Colors.red, width: 2.5),
        borderRadius: BorderRadius.circular(12)),
    child: TextField(
      obscureText: true,
      controller: _passwordController,
      decoration: const InputDecoration(
          border: InputBorder.none,
          hintText: "Password",
          hintStyle: TextStyle(
              color: Color.fromARGB(255, 197, 182, 182)),
          contentPadding: EdgeInsets.all(15.0)),
      style: const TextStyle(color: Colors.white),
    ),
  ),
),
const SizedBox(height: 5.0),
Padding(
  padding: const EdgeInsets.symmetric(horizontal: 30.0),
  child: Row(
    mainAxisAlignment: MainAxisAlignment.end,
    children: [
      GestureDetector(
        onTap: () {
          Navigator.push(
            context,
            MaterialPageRoute(
              builder: (context) {
                return const ForgotPasswordPage();
            },
          ),
        );
      },
      child: const Text("Forgot Password",
          style: TextStyle(
              color: Color.fromARGB(255, 226, 39, 26),
              fontSize: 18.0,
              fontWeight: FontWeight.bold)),
    ),
  ],
 ),
),
const SizedBox(
  height: 30.0,
),
Padding(
    padding: const EdgeInsets.symmetric(horizontal: 120.0),
    child: GestureDetector(
      onTap: signIn,
      child: Container(
        padding: const EdgeInsets.symmetric(
            horizontal: 12.0, vertical: 15.0),
        decoration: BoxDecoration(
            color: const Color.fromARGB(255, 226, 39, 26),
            borderRadius: BorderRadius.circular(12),
            shape: BoxShape.rectangle),
        child: const Center(
            child: Text(
          "Login",
          style: TextStyle(
              color: Colors.white,
              fontSize: 25.0,
              fontWeight: FontWeight.bold),
        )),
      ),
    )),
const SizedBox(
  height: 10.0,
),
Row(
  mainAxisAlignment: MainAxisAlignment.center,
  children: [
    const Text("New User? ",
        style: TextStyle(
            color: Colors.white,
            fontSize: 18.0,
                        fontWeight: FontWeight.bold)),
                    GestureDetector(
                      onTap: widget.showRegisterPage,
                      child: const Text("Register Now!",
                          style: TextStyle(
                              color: Color.fromARGB(255, 226, 39, 26),
                              fontSize: 18.0,
                              fontWeight: FontWeight.bold)),
                    ),
                  ],
                ),
              ],
            ),
          ),
        )));
  }
}
```

OUTPUT:



RESULT:

Thus, GUI components, Font and Colors have been implemented using Flutter.

EX.NO:2 **LAYOUT MANAGERS AND EVENT LISTENERS**

AIM:

To implement an application that uses layout managers and event listeners.

PROCEDURE:

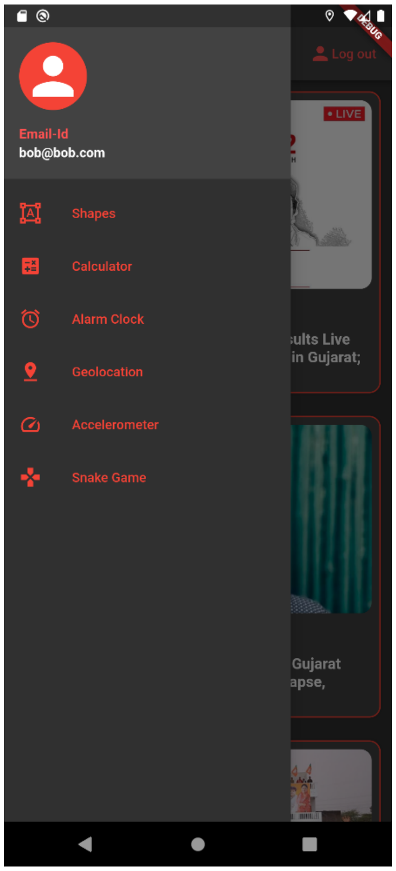
- Layout managers:
  - Column() class is used to display its children in a vertical way.
  - Children property is used to specify its descendants.
  - ListTile is a fixed-height row that typically contains some text as well as leading or trailing icon.
  - The icons (or other widgets) for the tile are defined with the leading and trailing parameters.
- Event listeners:
  - onPressed() property is used to assign a callback function to the button or icon.
  - The application executes this function whenever the user presses taps the chip.
  - If onPressed() is null, then it denotes disabled.

CODE:

```
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:news_hub/components/custom_list_tile.dart';
import 'package:news_hub/models/article_model.dart';
import 'package:news_hub/screens/accelerometer.dart';
import 'package:news_hub/screens/alarm.dart';
import 'package:news_hub/screens/calculator.dart';
import 'package:news_hub/screens/geolocator.dart';
import 'package:news_hub/screens/shapes.dart';
import 'package:news_hub/screens/snake_game.dart';
import 'package:news_hub/services/api_service.dart';

class HomePage extends StatefulWidget {
  const HomePage({
    Key? key,
  }) : super(key: key);

  @override
  State<HomePage> createState() => _HomePageState();
}

class _HomePageState extends State<HomePage> {
  ApiService client = ApiService();

  @override
  Widget build(BuildContext context) {
```

```
return Scaffold(
  backgroundColor: Colors.grey[850],
  appBar: AppBar(
    foregroundColor: Colors.redAccent,
    backgroundColor: Colors.grey[800],
    title: Row(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        Icon(Icons.newspaper_rounded, color: Colors.red[600], size: 30.0),
        const Text(
          " News Hub",
          style: TextStyle(
              color: Colors.red,
              fontWeight: FontWeight.bold,
              fontSize: 20.0),
        ),
      ],
    ),
    centerTitle: true,
    actions: [
      MaterialButton(
        onPressed: () {
          FirebaseAuth.instance.signOut();
        },
        child: Row(
          children: const [
            Icon(
              Icons.person,
              color: Colors.redAccent,
            ),
            Text(
              "Log out",
              style: TextStyle(
                color: Colors.redAccent,
              ),
            ),
          ],
        ),
      )
    ],
  ),
  drawer: Drawer(
    backgroundColor: Colors.grey[850],
    child: ListView(
      // Important: Remove any padding from the ListView.
      padding: EdgeInsets.zero,
      children: [
        UserAccountsDrawerHeader(
          decoration: BoxDecoration(color: Colors.grey[800]),
  accountName: const Text(
    "Email-Id",
    style: TextStyle(
      fontWeight: FontWeight.bold,
      color: Colors.redAccent,
    ),
  ),
  accountEmail: Text(
    userEmail.toString(),
    style: const TextStyle(
      fontWeight: FontWeight.bold,
    ),
  ),
  currentAccountPicture: const CircleAvatar(
    backgroundColor: Colors.red,
    child: Icon(Icons.person, size: 65.0, color: Colors.white),
  ),
),
ListTile(
  leading: const Icon(
    Icons.format_shapes,
    color: Colors.red,
  ),
  title: const Text(
    'Shapes',
    style: TextStyle(
      color: Colors.red,
    ),
  ),
  onTap: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (context) {
          return ShapesPage();
        },
      ),
    );
  },
),
ListTile(
  leading: const Icon(
    Icons.calculate,
    color: Colors.red,
  ),
  title: const Text(
    'Calculator',
    style: TextStyle(
      color: Colors.red,
    ),
  ),
  onTap: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (context) {
          return const CalculatorPage();
        },
      ),
    );
  },
),
ListTile(
  leading: const Icon(
    Icons.alarm,
    color: Colors.red,
  ),
  title: const Text(
    "Alarm Clock",
    style: TextStyle(
      color: Colors.red,
    ),
  ),
  onTap: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (context) {
          return AlarmPage();
        },
      ),
    );
  },
),
ListTile(
  leading: const Icon(
    Icons.pin_drop,
    color: Colors.red,
  ),
  title: const Text(
    'Geolocation',
    style: TextStyle(
      color: Colors.red,
    ),
  ),
  onTap: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (context) {
          return const GeolocatorPage();
        },
      ),
    );
  },
),
ListTile(
  leading: const Icon(
    Icons.speed,
    color: Colors.red,
  ),
  title: const Text(
    'Accelerometer',
    style: TextStyle(
      color: Colors.red,
    ),
  ),
  onTap: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (context) {
          return const AccelerometerPage(title: 'Accelerometer');
        },
      ),
    );
  },
),
ListTile(
  leading: const Icon(
    Icons.gamepad,
    color: Colors.red,
  ),
  title: const Text(
    'Snake Game',
    style: TextStyle(
      color: Colors.red,
    ),
  ),
  onTap: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (context) {
          return const SnakeGamePage(title: 'Snake Game');
        },
      ),
                );
              },
            ),
          ],
        ),
      ),
      body: FutureBuilder(
        future: client.getArticle(),
        builder: (BuildContext context, AsyncSnapshot snapshot) {
          if (snapshot.hasData) {
            List<Article>? articles = snapshot.data;

            return ListView.builder(
              itemCount: articles?.length,
              itemBuilder: (context, index) =>
                  customListTile(articles![index], context),
            );
          }
          return const Center(
              child: CircularProgressIndicator(
            color: Colors.redAccent,
          ));
        },
      ),
    );
  }
}
```

OUTPUT:



RESULT:

Thus, an application that uses layout managers and event listeners has been implemented using Flutter.

EX.NO: 3 **SIMPLE CALCULATOR**

AIM:

To develop a naive calculator application.

PROCEDURE:

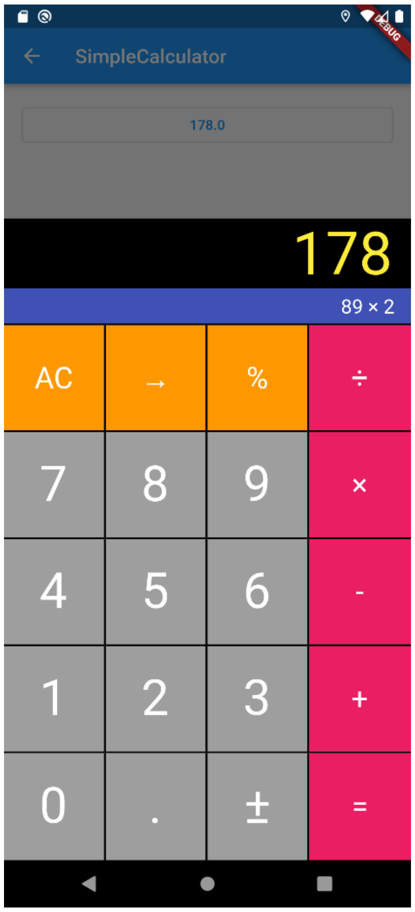
- Initialize num1, num2 and res (result) as 0
- Declare a function for each of the basic arithmetic operations (+ , - , * , /) which takes two operands as parameters and returns the result.
- Use the TextField, to get num1 and num2 as input.
- TextEditingController is used to retrieve the values of the TextField(s).
- Use another non-editable TextField to display the result.
- Use MaterialButton to perform the labelled arithmetic operation.

CODE:

```
import 'package:flutter/foundation.dart';
import 'package:flutter/material.dart';
import 'package:flutter_simple_calculator/flutter_simple_calculator.dart';

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

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          leading: IconButton(
            icon: Icon(Icons.arrow_back, color: Colors.white),
            onPressed: () => Navigator.of(context).pop(),
          ),
          title: const Text('SimpleCalculator'),
        ),
        body: const Padding(
          padding: EdgeInsets.all(18.0),
          child: SizedBox(
            width: double.infinity,
            child: CalcButton(),
          ),
        ),
      ),
    );
  }
}
```

```
class CalcButton extends StatefulWidget {
  const CalcButton({Key? key}) : super(key: key);

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

class _CalcButtonState extends State<CalcButton> {
  double? _currentValue = 0;
  @override
  Widget build(BuildContext context) {
    var calc = SimpleCalculator(
      value: _currentValue!,
      hideExpression: false,
      hideSurroundingBorder: true,
      autofocus: true,
      onChanged: (key, value, expression) {
        setState(() {
          _currentValue = value ?? 0;
        });
        if (kDebugMode) {
          print('$key\t$value\t$expression');
        }
      },
      onTappedDisplay: (value, details) {
        if (kDebugMode) {
          print('$value\t${details.globalPosition}');
        }
      },
      theme: const CalculatorThemeData(
        borderColor: Colors.black,
        borderWidth: 2,
        displayColor: Colors.black,
        displayStyle: TextStyle(fontSize: 80, color: Colors.yellow),
        expressionColor: Colors.indigo,
        expressionStyle: TextStyle(fontSize: 20, color: Colors.white),
        operatorColor: Colors.pink,
        operatorStyle: TextStyle(fontSize: 30, color: Colors.white),
        commandColor: Colors.orange,
        commandStyle: TextStyle(fontSize: 30, color: Colors.white),
        numColor: Colors.grey,
        numStyle: TextStyle(fontSize: 50, color: Colors.white),
      ),
    );
    return OutlinedButton(
      child: Text(_currentValue.toString()),
      onPressed: () {
        showModalBottomSheet(
          isScrollControlled: true,
          context: context,
          builder: (BuildContext context) {
            return SizedBox(
                height: MediaQuery.of(context).size.height * 0.75,
                child: calc);
          });
    },
  );
}
```

OUTPUT:



RESULT:

Thus, a simple naive calculator application is developed using Flutter.

EX.NO:4 **BASIC GRAPHICAL PRIMITIVES**

AIM:

To write an application that draws basic graphical primitives on the screen.

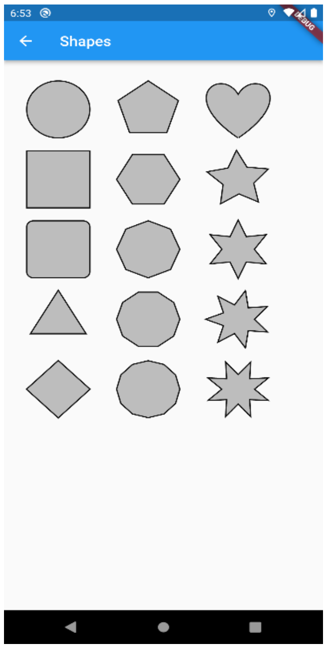
PROCEDURE:

- Declare a class for each graphical primitive.
- The CustomPainter class is used.
- The paint method takes canvas and size as parameters.
- Create an instance of Paint() class.
- canvas.drawRect() is used to draw a rectangle.
- Similarly, for line drawLine() is used.
- For circle and arc, drawCircle() and drawArc() are used respectively.
- Inside the scaffold, the required class is called by specifying it as the painter of CustomPaint class.

CODE:

```
import 'package:flutter/material.dart';
import 'package:flutter_shapes/flutter_shapes.dart';

class ShapesPage extends StatefulWidget {
  @override
  _ShapesPageState createState() => _ShapesPageState();
}

class _ShapesPageState extends State<ShapesPage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        leading: IconButton(
          icon: const Icon(Icons.arrow_back, color: Colors.white),
          onPressed: () => Navigator.of(context).pop(),
        ),
        title: const Text('Shapes'),
      ),
      body: Padding(
        padding: const EdgeInsets.all(50.0),
        child: SizedBox(
          width: double.infinity,
          child: CustomPaint(painter: _MyPainter()),
        ),
      ),
    );
  }
}

class _MyPainter extends CustomPainter {
  @override
  bool shouldRepaint(_MyPainter oldDelegate) {
    return false;
  }

  @override
  void paint(Canvas canvas, Size size) {
    final Paint stroke = Paint()
      ..color = Colors.black
      ..style = PaintingStyle.stroke
      ..strokeWidth = 3;
    final Paint fill = Paint()
      ..color = Colors.grey[400]!
      ..style = PaintingStyle.fill;
    const double radius = 40;
    final Shapes shapes = Shapes(canvas: canvas);
    for (String type in Shapes.types) {
      final int index = Shapes.types.indexOf(type);
      final double x =
          radius * 0.5 + radius * 2.9 * (index / 5).floor().toDouble();
      final double y = radius * 0.5 + radius * 2.5 * (index % 5).toDouble();
      for (Paint paint in <Paint>[stroke, fill]) {
        (shapes
              ..paint = paint
              ..radius = radius
              ..center = Offset(x, y))
            .draw(type);
      }
    }
  }
}
```

OUTPUT:



RESULT:

Hence, an application that draws basic graphical primitives on the screen has been implemented using Flutter.

EX.NO:5 **DATABASE CONNECTION**

AIM:

To develop an application that makes use of database.

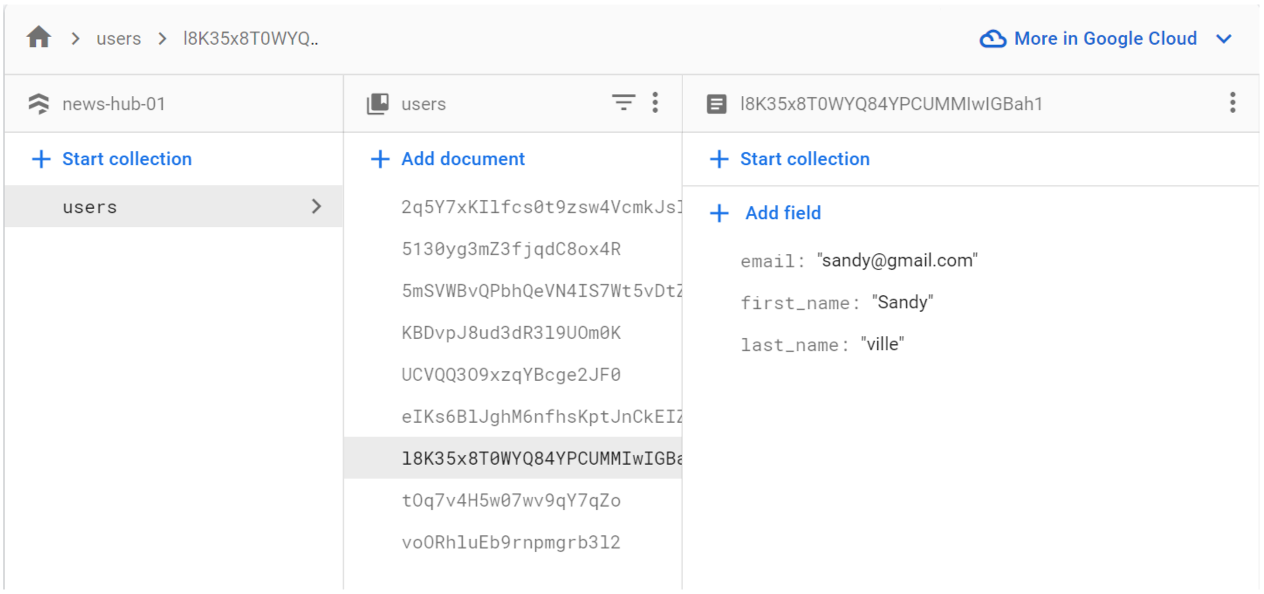
PROCEDURE:

- Install the following packages:
  - npm install firebase-tools
  - flutter pub add firebase_core
  - flutter pub add firebase_auth
- Use 'firebase login' command to login to google account
- Use 'flutterfire configure' to add a firebase project to the application.
- Import the generated 'firebase options' file to main.dart file.
- FirebaseAuth.*instance*.currentUser is used to get the current user object
- Use FilePicker to select files from the device.
- storage.ref().child() is used to store the chosen file to Firebase storage.

CODE:

```
import 'package:flutter/material.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:cloud_firestore/cloud_firestore.dart';

void main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp();
  runApp(const MyApp());
}

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

  @override
  Widget build(BuildContext context) {
    return const MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Firebase Firestore',
      home: CRUDPage(),
    );
  }
}

class CRUDPage extends StatefulWidget {
  const CRUDPage({Key? key}) : super(key: key);
  @override
  _CRUDPageState createState() => _CRUDPageState();
}

class _CRUDPageState extends State<CRUDPage> {
// text fields' controllers
  final TextEditingController _nameController = TextEditingController();
  final TextEditingController _priceController = TextEditingController();

  final CollectionReference _products =
      FirebaseFirestore.instance.collection('products');

  Future<void> _create([DocumentSnapshot? documentSnapshot]) async {
    await showModalBottomSheet(
        isScrollControlled: true,
        context: context,
        builder: (BuildContext ctx) {
          return Padding(
            padding: EdgeInsets.only(
                top: 20,
                left: 20,
                right: 20,
                bottom: MediaQuery.of(ctx).viewInsets.bottom + 20),
            child: Column(
              mainAxisSize: MainAxisSize.min,
              crossAxisAlignment: CrossAxisAlignment.start,
              children: [
                TextField(
                  controller: _nameController,
                  decoration: const InputDecoration(labelText: 'Name'),
                ),
                TextField(
                  keyboardType:
                      const TextInputType.numberWithOptions(decimal: true),
                  controller: _priceController,
                  decoration: const InputDecoration(
                    labelText: 'Price',
                  ),
                ),
                const SizedBox(
                  height: 20,
                ),
                ElevatedButton(
                  child: const Text('Create'),
                  onPressed: () async {
                    final String name = _nameController.text;
                    final double? price =
                      double.tryParse(_priceController.text);
                    if (price != null) {
                      await _products.add({"name": name, "price": price});

                      _nameController.text = '';
                      _priceController.text = '';
                      Navigator.of(context).pop();
                    }
                  },
                )
              ],
            ),
          );
        });
  }

  Future<void> _update([DocumentSnapshot? documentSnapshot]) async {
    if (documentSnapshot != null) {
      _nameController.text = documentSnapshot['name'];
      _priceController.text = documentSnapshot['price'].toString();
    }

    await showModalBottomSheet(
        isScrollControlled: true,
        context: context,
        builder: (BuildContext ctx) {
          return Padding(
            padding: EdgeInsets.only(
                top: 20,
                left: 20,
                right: 20,
                bottom: MediaQuery.of(ctx).viewInsets.bottom + 20),
            child: Column(
              mainAxisSize: MainAxisSize.min,
              crossAxisAlignment: CrossAxisAlignment.start,
              children: [
                TextField(
                  controller: _nameController,
                  decoration: const InputDecoration(labelText: 'Name'),
                ),
                TextField(
                  keyboardType:
                      const TextInputType.numberWithOptions(decimal: true),
                  controller: _priceController,
                  decoration: const InputDecoration(
                    labelText: 'Price',
                  ),
                ),
                const SizedBox(
                  height: 20,
                ),
                ElevatedButton(
                  child: const Text('Update'),
                  onPressed: () async {
                    final String name = _nameController.text;
                    final double? price =
                        double.tryParse(_priceController.text);
                    if (price != null) {
                      await _products
                          .doc(documentSnapshot!.id)
                          .update({"name": name, "price": price});
                      _nameController.text = '';
                      _priceController.text = '';
                      Navigator.of(context).pop();
                    }
                  },
                )
              ],
            ),
          );
        });
  }

  Future<void> _delete(String productId) async {
    await _products.doc(productId).delete();

    ScaffoldMessenger.of(context).showSnackBar(const SnackBar(
        content: Text('You have successfully deleted a product')));
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
        appBar: AppBar(
          title: const Center(child: Text('Firebase Firestore')),
        ),
        body: StreamBuilder(
          stream: _products.snapshots(),
          builder: (context, AsyncSnapshot<QuerySnapshot> streamSnapshot) {
            if (streamSnapshot.hasData) {
              return ListView.builder(
                itemCount: streamSnapshot.data!.docs.length,
                itemBuilder: (context, index) {
                  final DocumentSnapshot documentSnapshot =
                      streamSnapshot.data!.docs[index];
                  return Card(
                    margin: const EdgeInsets.all(10),
                    child: ListTile(
                      title: Text(documentSnapshot['name']),
                      subtitle: Text(documentSnapshot['price'].toString()),
                      trailing: SizedBox(
                        width: 100,
                        child: Row(
                          children: [
                            IconButton(
                                icon: const Icon(Icons.edit),
                                onPressed: () => _update(documentSnapshot)),
                            IconButton(
                                icon: const Icon(Icons.delete),
                                onPressed: () => _delete(documentSnapshot.id)),
                          ],
                        ),
                      ),
                    ),
                  );
                },
              );
            }

            return const Center(
              child: CircularProgressIndicator(),
            );
          },
        ),
// Add new product
        floatingActionButton: FloatingActionButton(
          onPressed: () => _create(),
          child: const Icon(Icons.add),
        ),
        floatingActionButtonLocation: FloatingActionButtonLocation.centerFloat);
  }
}
```

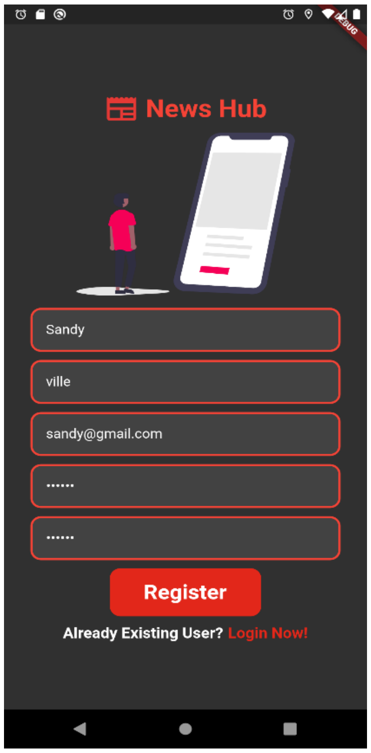
OUTPUT:



users > l8K35x8TOWYQ..

More in Google Cloud

news-hub-01	users	l8K35x8TOWYQ84YPCUMMIwIGBah1
Start collection	Add document	Start collection
users	2q5Y7xKIlfcs0t9zsw4VcmkJsl	Add field
	5130yg3mZ3fjqdC8ox4R	email: "sandy@gmail.com"
	5mSVWBvQPbhQeVN4IS7Wt5vDtZ	first_name: "Sandy"
	KBDvpJ8ud3dR319UOm0K	last_name: "ville"
	UCVQQ309xzqYBcge2JF0	
	eIKs6BlJghM6nfhsKptJnCkEIZ	
	l8K35x8T0WYQ84YPCUMMIwIGBa	
	tOq7v4H5w07wv9qY7qZo	
	voORhluEb9rnpmgrb312	



RESULT:

Thus, an application that makes use of Firebase Storage (Database) for storing, retrieving and deleting files of each user has been implemented using Flutter.

EX.NO:6 **RSS FEED**

AIM:

To develop an application that makes use of RSS Feed.

PROCEDURE:

- *Import packages.*
  import 'package:webfeed/webfeed.dart';
  import 'package:http/http.dart' as http;
  import 'package:url_launcher/url_launcher.dart';
- *Define RSS Feed URL (FEED_URL)*
- *Create a variable to hold our RSS feed data. (_feed)*
- *Create a place holder for our title (_title)*
- *Create a method to navigate to the selected RSS item* (openFeed)
- Use RssFeed.parse(response.body)to *grab the RSS data from the provided URL.*
- *Create the UI for the ListView and plug in the retrieved RSS data*

CODE:

```
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';

// Custom colors for my terminal theme.
import 'package:flutter_rss_reader/colors.dart';

// TODO 2: Import packages we added to our pubspec.yaml file.
import 'package:webfeed/webfeed.dart';
import 'package:http/http.dart' as http;
import 'package:url_launcher/url_launcher.dart';

class RSSReader extends StatefulWidget {
  RSSReader() : super();

  // Setting title for the action bar.
  final String title = '<Hacker News\\> | Jobs Feed';

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

class RSSReaderState extends State<RSSReader> {
  // Feed URL being used for the app. In this case is the Hacker News job feed.
  // TODO 3: Define RSS Feed URL
  static const String FEED_URL = 'https://hnrss.org/jobs';

  // TODO 4: Create a variable to hold our RSS feed data.
  RssFeed _feed; // RSS Feed Object
  // TODO 5: Create a place holder for our title.
```

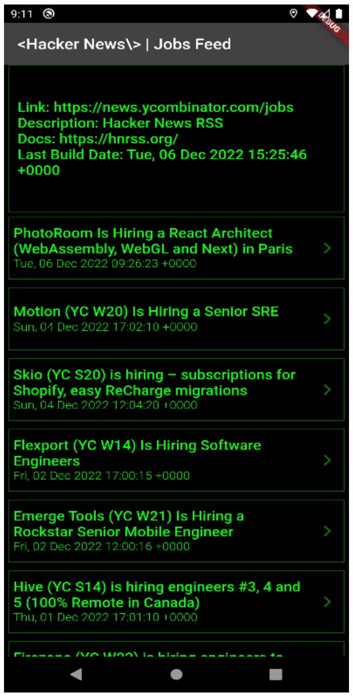
```
String _title; // Place holder for appbar title.

// TODO 6: Setup our notification messages.
// Notification Strings
static const String loadingMessage = 'Loading Feed...';
static const String feedLoadErrorMessage = 'Error Loading Feed.';
static const String feedOpenErrorMessage = 'Error Opening Feed.';

// TODO 7: Create a GlobalKey object to hold our key for the refresh feature.
// Key for the RefreshIndicator
// See the documentation linked below for info on the RefreshIndicatorState
// class and the GloablKey class.
// https://api.flutter.dev/flutter/widgets/GlobalKey-class.html
// https://api.flutter.dev/flutter/material/RefreshIndicatorState-class.html
GlobalKey<RefreshIndicatorState> _refreshKey;

// TODO 8: Create a method to update the user about data changes.
// Method to change the title as a way to inform the user what is going on
// while retrieving the RSS data.
updateTitle(title) {
  setState(() {
    _title = title;
  });
}

// TODO 9: Create a method to reload the RSS feed data when the refresh feature is used.
// Method to help refresh the RSS data.
updateFeed(feed) {
  setState(() {
    _feed = feed;
  });
}

// TODO 10: Create a method to navigate to the selected RSS item.
// Method to navigate to the URL of a RSS feed item.
Future<void> openFeed(String url) async {
  if (await canLaunch(url)) {
    await launch(
      url,
      forceSafariVC: true,
      forceWebView: false,
    );
    return;
  }
  updateTitle(feedOpenErrorMessage);
}

// TODO 11: Create a method to load the RSS data.
```

```
  // Method to load the RSS data.
  load() async {
    updateTitle(loadingMessage);
    loadFeed().then((result) {
      if (null == result || result.toString().isEmpty) {
        // Notify user of error.
        updateTitle(feedLoadErrorMessage);
        return;
      }
      // If there is no error, load the RSS data into the _feed object.
      updateFeed(result);
      // Reset the title.
      updateTitle("<Hacker News\\> | Jobs Feed");
    });
  }

  // TODO 12: Create a method to grab the RSS data from the provided URL.
  // Method to get the RSS data from the provided URL in the FEED_URL variable.
  Future<RssFeed> loadFeed() async {
    try {
      final client = http.Client();
      final response = await client.get(FEED_URL);
      print(RssFeed.parse(response.body).toString());
      return RssFeed.parse(response.body);
    } catch (e) {
      // handle any exceptions here
      // print(e);
    }
    return null;
  }

  // TODO 13: Override the initState() method and setup the _refreshKey variable, update the title, and call
the load() method.
  // When the app is initialized, we setup our GlobalKey, set our title, and
  // call the load() method which loads the RSS feed and UI.
  @override
  void initState() {
    super.initState();
    _refreshKey = GlobalKey<RefreshIndicatorState>();
    updateTitle(widget.title);
    load();
  }

  // TODO 14: Create a method to check if the RSS feed is empty.
  // Method to check if the RSS feed is empty.
  isFeedEmpty() {
    return null == _feed || null == _feed.items;
  }
```

```
// TODO 15: Create method to load the UI and RSS data.
// Method for the pull to refresh indicator and the actual ListView UI/Data.
body() {
  return isFeedEmpty()
      ? Center(
          child: CircularProgressIndicator(),
        )
      : RefreshIndicator(
          key: _refreshKey,
          child: list(),
          onRefresh: () => load(),
        );
}

@override
Widget build(BuildContext context) {
  return SafeArea(
    child: Scaffold(
      backgroundColor: colorHackerBackground,
      appBar: AppBar(
        title: Text(_title),
      ),
      body: body(),
    ),
  );
}

// TODO 16: Create the UI for the ListView and plug in the retrieved RSS data.
// ListView
// Consists of two main widgets. A Container Widget displaying info about the
// RSS feed and the ListView containing the RSS Data. Both contained in a
// Column Widget.
list() {
  return Column(
      crossAxisAlignment: CrossAxisAlignment.stretch,
      children: <Widget>[
        // Container displaying RSS feed info.
        Expanded(
          flex: 1,
          child: Container(
            padding: EdgeInsets.all(10.0),
            margin: EdgeInsets.only(left: 5.0, right: 5.0),
            decoration: customBoxDecoration(),
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.stretch,
              mainAxisAlignment: MainAxisAlignment.center,
              children: <Widget>[
        Text(
          "Link: " + _feed.link,
          style: TextStyle(
              fontSize: 18.0,
              fontWeight: FontWeight.w500,
              color: colorHackerHeading),
        ),
        Text(
          "Description: " + _feed.description,
          style: TextStyle(
              fontSize: 18.0,
              fontWeight: FontWeight.w500,
              color: colorHackerHeading),
        ),
        Text(
          "Docs: " + _feed.docs,
          style: TextStyle(
              fontSize: 18.0,
              fontWeight: FontWeight.w500,
              color: colorHackerHeading),
        ),
        Text(
          "Last Build Date: " + _feed.lastBuildDate,
          style: TextStyle(
              fontSize: 18.0,
              fontWeight: FontWeight.w500,
              color: colorHackerHeading),
        ),
      ],
    ),
  ),
),
// ListView that displays the RSS data.
Expanded(
  flex: 3,
  child: Container(
    child: ListView.builder(
      padding: EdgeInsets.all(5.0),
      itemCount: _feed.items.length,
      itemBuilder: (BuildContext context, int index) {
        final item = _feed.items[index];
        return Container(
          margin: EdgeInsets.only(
            bottom: 10.0,
          ),
          decoration: customBoxDecoration(),
          child: ListTile(
            title: title(item.title),
          subtitle: subtitle(item.pubDate),
          trailing: rightIcon(),
          contentPadding: EdgeInsets.all(5.0),
          onTap: () => openFeed(item.link),
         ),
        );
       },
      ),
     ),
    ),
   ]);
}

// Method that returns the Text Widget for the title of our RSS data.
title(title) {
  return Text(
    title,
    style: TextStyle(
        fontSize: 18.0,
        fontWeight: FontWeight.w500,
        color: colorHackerHeading),
    maxLines: 2,
    overflow: TextOverflow.ellipsis,
  );
}

// Method that returns the Text Widget for the subtitle of our RSS data.
subtitle(subTitle) {
  return Text(
    subTitle,
    style: TextStyle(
        fontSize: 15.0,
        fontWeight: FontWeight.w300,
        color: colorHackerHeading),
    maxLines: 1,
    overflow: TextOverflow.ellipsis,
  );
}

// Method that returns Icon Widget.
rightIcon() {
  return Icon(
    Icons.keyboard_arrow_right,
    color: colorHackerBorder,
    size: 30.0,
  );
}
```

```
 // Custom box decoration for the Container Widgets.
 BoxDecoration customBoxDecoration() {
   return BoxDecoration(
     border: Border.all(
       color: colorHackerBorder, // border color
       width: 1.0,
     ),
   );
 }
}
```

OUTPUT:



9:11

DEBUG

<Hacker News\> | Jobs Feed

Link: https://news.ycombinator.com/jobs
Description: Hacker News RSS
Docs: https://hnrss.org/
Last Build Date: Tue, 06 Dec 2022 15:25:46 +0000

PhotoRoom Is Hiring a React Architect (WebAssembly, WebGL and Next) in Paris
Tue, 06 Dec 2022 09:26:23 +0000

Motion (YC W20) Is Hiring a Senior SRE
Sun, 04 Dec 2022 17:02:10 +0000

Skio (YC S20) is hiring – subscriptions for Shopify, easy ReCharge migrations
Sun, 04 Dec 2022 12:04:20 +0000

Flexport (YC W14) Is Hiring Software Engineers
Fri, 02 Dec 2022 17:00:15 +0000

Emerge Tools (YC W21) Is Hiring a Rockstar Senior Mobile Engineer
Fri, 02 Dec 2022 12:00:16 +0000

Hive (YC S14) is hiring engineers #3, 4 and 5 (100% Remote in Canada)
Thu, 01 Dec 2022 17:01:10 +0000

RESULT:

Thus, an application that uses RSS feed has been developed using Flutter.

EX.NO:7 **MULTI-THREADING**

AIM:

To write an application that implements multi-threading.

PROCEDURE:

- Install the following packages:
  - npm install firebase-tools
  - flutter pub add firebase_core
  - flutter pub add firebase_auth
- Use ‘firebase login’ command to login to google account
- Use ‘flutterfire configure’ to add a firebase project to the application.
- Import the generated ‘firebase options’ file to main.dart file.
- FirebaseAuth.*instance*.currentUser is used to get the current user object
- Use FilePicker to select files from the device.
- storage.ref().child() is used to store the chosen file to Firebase storage.
- ‘async’ enables your program to start a potentially long-running task and still be able to be responsive to other events while that task runs, rather than having to wait until that task has finished.
- ‘await’ keyword is used before a call to a function that returns a promise. This makes the code wait at that point until the promise is settled, at which point the fulfilled value of the promise is treated as a return value, or the rejected value is thrown.

CODE:

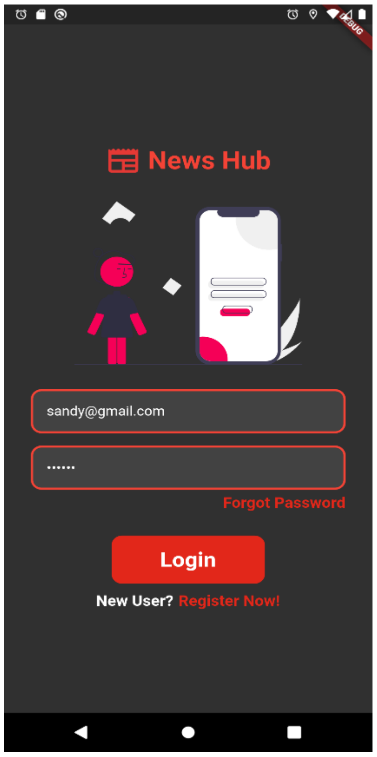
main.dart

```
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:news_hub/screens/auth_handle.dart';
import 'package:news_hub/screens/home.dart';
import 'package:firebase_core/firebase_core.dart';

Future<void> main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp();

  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);
  @override
  Widget build(BuildContext context) {
    return const MaterialApp(
      home: MainPage(), //wrapper to handle auth_pages(login and signup) / homePage
    );
  }
}

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: StreamBuilder<User?>(
          stream: FirebaseAuth.instance.authStateChanges(),
          builder: (context, snapshot) {
            if (snapshot.hasData) {
              return const HomePage();
            } else {
              return const AuthPage();
            }
          }),
    );
  }
}
```

Authentication.dart

```
import 'package:flutter/material.dart';
import 'package:flutter_svg/flutter_svg.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:news_hub/screens/forgot_password.dart';
import 'dart:async';

class LoginPage extends StatefulWidget {
  final VoidCallback showRegisterPage;
  const LoginPage({Key? key, required this.showRegisterPage}) : super(key: key);

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

class _LoginPageState extends State<LoginPage> {
  final _emailController = TextEditingController();
  final _passwordController = TextEditingController();

  Future signIn() async {
    await FirebaseAuth.instance.signInWithEmailAndPassword(
        email: _emailController.text.trim(),
        password: _passwordController.text.trim());
    var snackBar = const SnackBar(
        content: Text(
      "Login Successfull !!",
      style: TextStyle(fontWeight: FontWeight.bold),
    ));
    ScaffoldMessenger.of(context).showSnackBar(snackBar);
  }

  @override
  void dispose() {
    _emailController.dispose();
    _passwordController.dispose();
    super.dispose();
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
        backgroundColor: Colors.grey[850],
        body: SafeArea(
            child: Center(
          child: SingleChildScrollView(
            child: Column(
              mainAxisAlignment: MainAxisAlignment.center,
              children: [
                const SizedBox(
                  height: 20.0,
                ),
                Row(
                  mainAxisAlignment: MainAxisAlignment.center,
                  children: [
                    Icon(Icons.newspaper_rounded,
                        color: Colors.red[600], size: 40.0),
                    const Text(
                      " News Hub",
                      style: TextStyle(
                          color: Colors.red,
                          fontWeight: FontWeight.bold,
                          fontSize: 30.0),
                    ),
                  ],
                ),
                const SizedBox(
                  height: 30.0,
                ),
                // PhotoView(
                //   imageProvider: const AssetImage("assets/large-image.jpg"),
                //   customSize: const Size(200, 200),
                // ),
  SvgPicture.asset(
    "assets/images/login_img.svg",
    height: 200.0,
    width: 200.0,
  ),
  const SizedBox(
    height: 30.0,
  ),
  Padding(
    padding: const EdgeInsets.symmetric(horizontal: 30.0),
    child: Container(
      decoration: BoxDecoration(
          color: Colors.grey[800],
          border: Border.all(color: Colors.red, width: 2.5),
          borderRadius: BorderRadius.circular(12)),
      child: TextField(
        controller: _emailController,
        decoration: const InputDecoration(
            border: InputBorder.none,
            hintText: "Email",
            hintStyle: TextStyle(
                color: Color.fromARGB(255, 197, 182, 182)),
            contentPadding: EdgeInsets.all(15.0)),
        style: const TextStyle(color: Colors.white),
      ),
    ),
  ),
  const SizedBox(
    height: 15.0,
  ),
  Padding(
    padding: const EdgeInsets.symmetric(horizontal: 30.0),
    child: Container(
      decoration: BoxDecoration(
          color: Colors.grey[800],
          border: Border.all(color: Colors.red, width: 2.5),
          borderRadius: BorderRadius.circular(12)),
      child: TextField(
        obscureText: true,
        controller: _passwordController,
        decoration: const InputDecoration(
            border: InputBorder.none,
            hintText: "Password",
            hintStyle: TextStyle(
                color: Color.fromARGB(255, 197, 182, 182)),
            contentPadding: EdgeInsets.all(15.0)),
        style: const TextStyle(color: Colors.white),
      ),
    ),
  ),
  const SizedBox(height: 5.0),
  Padding(
    padding: const EdgeInsets.symmetric(horizontal: 30.0),
    child: Row(
      mainAxisAlignment: MainAxisAlignment.end,
      children: [
        GestureDetector(
          onTap: () {
            Navigator.push(
              context,
              MaterialPageRoute(
                builder: (context) {
                  return const ForgotPasswordPage();
                },
              ),
            );
          },
          child: const Text("Forgot Password",
              style: TextStyle(
                  color: Color.fromARGB(255, 226, 39, 26),
                  fontSize: 18.0,
                  fontWeight: FontWeight.bold)),
        ),
      ],
    ),
  ),
  const SizedBox(
    height: 30.0,
  ),
  Padding(
      padding: const EdgeInsets.symmetric(horizontal: 120.0),
      child: GestureDetector(
        onTap: signIn,
        child: Container(
          padding: const EdgeInsets.symmetric(
              horizontal: 12.0, vertical: 15.0),
          decoration: BoxDecoration(
              color: const Color.fromARGB(255, 226, 39, 26),
              borderRadius: BorderRadius.circular(12),
              shape: BoxShape.rectangle),
          child: const Center(
              child: Text(
            "Login",
            style: TextStyle(
                      color: Colors.white,
                      fontSize: 25.0,
                      fontWeight: FontWeight.bold),
                   )),
                  ),
                 )),
            const SizedBox(
              height: 10.0,
            ),
            Row(
              mainAxisAlignment: MainAxisAlignment.center,
              children: [
                const Text("New User? ",
                    style: TextStyle(
                        color: Colors.white,
                        fontSize: 18.0,
                        fontWeight: FontWeight.bold)),
                GestureDetector(
                  onTap: widget.showRegisterPage,
                  child: const Text("Register Now!",
                      style: TextStyle(
                          color: Color.fromARGB(255, 226, 39, 26),
                          fontSize: 18.0,
                          fontWeight: FontWeight.bold)),
                ),
              ],
            ),
          ],
        ),
      ),
    )));
   }
  }
```

OUTPUT:



RESULT:

Thus, an application that implements multithreading is implemented using Flutter and Firebase.

EX.NO:8 GPS LOCATION INFORMATION

AIM:

To develop a native application that uses GPS location information.

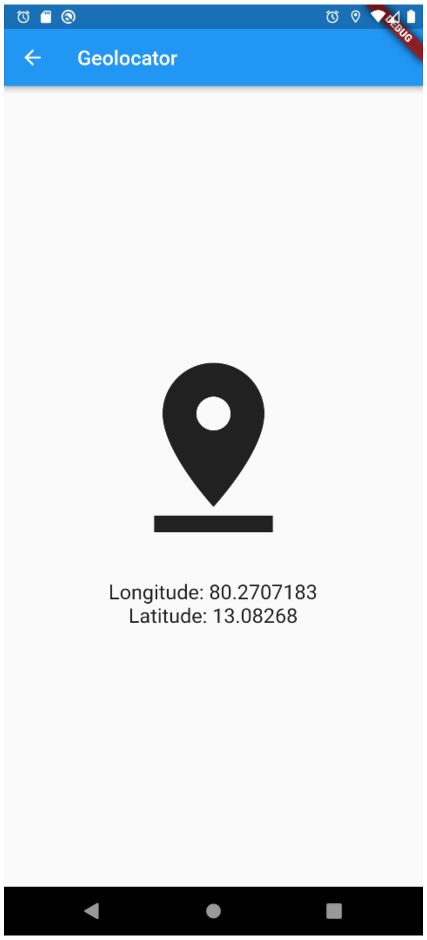
PROCEDURE:

- Install the following packages: geolocator & geocoding
- Import them using,
    - import 'package:geocoding/geocoding.dart';
    - import 'package:geolocator/geolocator.dart';
- Get current location of the device, by creating an instance of Geolocator and calling getCurrentPosition.
- Convert latitude and longitude values into address using placemarkFromCoordinates().

CODE:

```
import 'dart:async';

import 'package:flutter/material.dart';
import 'package:geolocator/geolocator.dart';

class GeolocatorPage extends StatefulWidget {
  const GeolocatorPage({Key? key}) : super(key: key);

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

class _GeolocatorPageState extends State<GeolocatorPage> {
  bool servicestatus = false;
  bool haspermission = false;
  late LocationPermission permission;
  late Position position;
  String long = "", lat = "";
  late StreamSubscription<Position> positionStream;

  @override
  void initState() {
    checkGps();
    super.initState();
  }

  checkGps() async {
    servicestatus = await Geolocator.isLocationServiceEnabled();
    if (servicestatus) {
      permission = await Geolocator.checkPermission();
      if (permission == LocationPermission.denied) {
        permission = await Geolocator.requestPermission();
        if (permission == LocationPermission.denied) {
          print('Location permissions are denied');
        } else if (permission == LocationPermission.deniedForever) {
          print("'Location permissions are permanently denied");
        } else {
          haspermission = true;
        }
      } else {
        haspermission = true;
      }

      if (haspermission) {
        // setState(() {
        //   //refresh the UI
        // });

        getLocation();
      }
    } else {
      print("GPS Service is not enabled, turn on GPS location");
    }

    // setState(() {
    //   //refresh the UI
    // });
  }

  getLocation() async {
    position = await Geolocator.getCurrentPosition(
        desiredAccuracy: LocationAccuracy.high);
    print(position.longitude); //Output: 80.24599079
    print(position.latitude); //Output: 29.6593457

    long = position.longitude.toString();
    lat = position.latitude.toString();

    setState(() {
      //refresh UI
    });

    LocationSettings locationSettings = LocationSettings(
      accuracy: LocationAccuracy.high, //accuracy of the location data
      distanceFilter: 100, //minimum distance (measured in meters) a
      //device must move horizontally before an update event is generated;
    );

    StreamSubscription<Position> positionStream =
      Geolocator.getPositionStream(locationSettings: locationSettings)
          .listen((Position position) {
    print(position.longitude); //Output: 80.24599079
    print(position.latitude); //Output: 29.6593457

    long = position.longitude.toString();
    lat = position.latitude.toString();

    setState(() {
      //refresh UI on update
    });
  });
}

@override
Widget build(BuildContext context) {
  return MaterialApp(
    home: Scaffold(
      appBar: AppBar(
        leading: IconButton(
          icon: const Icon(Icons.arrow_back, color: Colors.white),
          onPressed: () => Navigator.of(context).pop(),
        ),
        title: const Text('Geolocator'),
      ),
      body: Padding(
        padding: const EdgeInsets.all(18.0),
        child: SizedBox(
          width: double.infinity,
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              const Icon(
                Icons.pin_drop,
                size: 200.0,
              ),
              const SizedBox(
                height: 30.0,
              ),
              Text("Longitude: $long",
                  style: const TextStyle(
                    fontSize: 20,
                  )),
              Text(
                "Latitude: $lat",
                style: const TextStyle(
                  fontSize: 20,
                ),
              ),
              ],
            ),
          ),
        ),
      ),
    );
  }
}
```

OUTPUT:



Longitude: 80.2707183
Latitude: 13.08268

RESULT:

Thus, a native application that uses GPS location information has been developed.

EX.NO:9 **WRITING TO SD CARD**

AIM:

To implement an application that writes to SD card.

PROCEDURE:

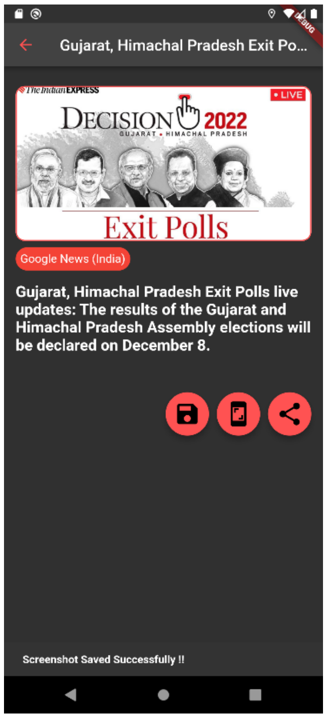
- Install path_provider package
- The path where is file is to be written is obtained using getExternalStorageDirectory() function.
- writeAsString(<String>) is used to write contents into a text file.
- readAsString() is used to read the contents of the file.

CODE:

```
import 'dart:io';
import 'dart:typed_data';

import 'package:flutter/material.dart';
import 'package:news_hub/models/article_model.dart';
import 'package:path_provider/path_provider.dart';
import 'package:screenshot/screenshot.dart';
import 'package:share_plus/share_plus.dart';

class ArticlePage extends StatefulWidget {
  final Article article;
  ArticlePage({required this.article});

  @override
  State<ArticlePage> createState() => _ArticlePageState();
}

class _ArticlePageState extends State<ArticlePage> {
  ScreenshotController screenshotController = ScreenshotController();

  snackbar_message(String text) {
    var snackBar = SnackBar(
        content: Text(
      text,
      style: TextStyle(fontWeight: FontWeight.bold),
    ));
    ScaffoldMessenger.of(context).showSnackBar(snackBar);
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
        appBar: AppBar(
  title: Text(
    widget.article.title,
    style: const TextStyle(color: Colors.white),
  ),
  backgroundColor: Colors.grey[800],
  foregroundColor: Colors.redAccent,
),
body: Screenshot(
  controller: screenshotController,
  child: Container(
    padding:
        const EdgeInsets.symmetric(horizontal: 15.0, vertical: 25.0),
    color: Colors.grey[850],
    child: Column(
      mainAxisAlignment: MainAxisAlignment.start,
      crossAxisAlignment: CrossAxisAlignment.start,
      children: [
        Container(
          height: 200.0,
          width: double.infinity,
          decoration: BoxDecoration(
              image: DecorationImage(
                  image: NetworkImage(widget.article.urlToImage),
                  fit: BoxFit.cover),
              borderRadius: BorderRadius.circular(12.0),
              border: Border.all(
                color: Colors.redAccent,
                width: 1.5,
              )),
        ),
        const SizedBox(
          height: 8.0,
        ),
        Container(
          padding: const EdgeInsets.all(6.0),
          decoration: BoxDecoration(
            color: Colors.red,
            borderRadius: BorderRadius.circular(30.0),
          ),
          child: Text(
            widget.article.source.name,
            style: const TextStyle(
              color: Colors.white,
              fontSize: 15.0,
            ),
          ),
        ),
const SizedBox(
  height: 15.0,
),
Text(
  widget.article.desc,
  style: const TextStyle(
    color: Colors.white,
    fontWeight: FontWeight.bold,
    fontSize: 20.0,
  ),
),
const SizedBox(
  height: 50,
),
Row(
  mainAxisAlignment: MainAxisAlignment.end,
  children: [
    FloatingActionButton(
      backgroundColor: Colors.redAccent,
      heroTag: "save_news",
      onPressed: () async {
        final directory = await getExternalStorageDirectory();
        String timestamp =
            DateTime.now().microsecondsSinceEpoch.toString();
        final filePath = await File(
                '${directory?.path}/${widget.article.title}_$timestamp.txt')
            .create();
        filePath.writeAsString(widget.article.desc);
        snackbar_message("News Saved Successfully !!");
      },
      child: const Icon(
        Icons.save,
        color: Colors.black,
        size: 35,
      ),
    ),
    const SizedBox(
      width: 10,
    ),
    FloatingActionButton(
      backgroundColor: Colors.redAccent,
      heroTag: "screenshot",
      onPressed: () async {
        await screenshotController
            .capture(delay: const Duration(milliseconds: 10))
            .then((Uint8List? image) async {
          if (image != null) {
```

```
                        final directory =
                            await getExternalStorageDirectory();
                        String timestamp = DateTime.now()
                            .microsecondsSinceEpoch
                            .toString();
                        final imagePath = await File(
                                '${directory?.path}/${widget.article.title}_$timestamp.png')
                            .create();
                        await imagePath.writeAsBytes(image);
                        snackbar_message(
                            "Screenshot Saved Successfully !!");
                      }
                    });
                  },
                  child: const Icon(
                    Icons.screenshot,
                    color: Colors.black,
                    size: 35,
                  ),
                ),
                const SizedBox(
                  width: 10,
                ),
                FloatingActionButton(
                  backgroundColor: Colors.redAccent,
                  heroTag: "share",
                  onPressed: () {
                    Share.share(
                      widget.article.desc,
                      subject: widget.article.title,
                    );
                  },
                  child: const Icon(
                    Icons.share,
                    color: Colors.black,
                    size: 35,
                  ),
                ),
              ],
            )
          ],
        ),
      ),
    ));
  }
}
```

OUTPUT:

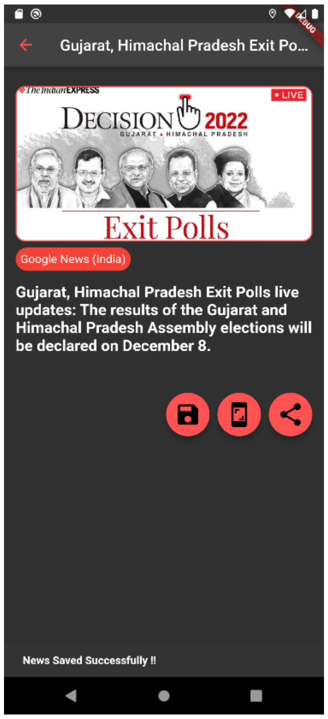


Gujarat, Himachal Pradesh Exit Po...

Google News (India)

Gujarat, Himachal Pradesh Exit Polls live updates: The results of the Gujarat and Himachal Pradesh Assembly elections will be declared on December 8.

Screenshot Saved Successfully !!



Gujarat, Himachal Pradesh Exit Po...

Google News (India)

Gujarat, Himachal Pradesh Exit Polls live updates: The results of the Gujarat and Himachal Pradesh Assembly elections will be declared on December 8.

News Saved Successfully !!

RESULT:

Hence, an application that writes to SD card has been implemented using Flutter.

EX.NO: 10 **ALERT BOX**

AIM:

To implement an application that creates an alert upon receiving a message.

PROCEDURE:

- On the To-do list page, create a TextButton labelled ‘ADD’ to add a new task.
- In the onPressed() property, use showDialog to specify the alert box contents.
- AlertDialog() is used to create the alert message box.
    - The content property is used to specify the message using Text(). In this case, the message displayed is “Task added”.
    - The action property is used to specify the buttons in the alert box using TextButton().

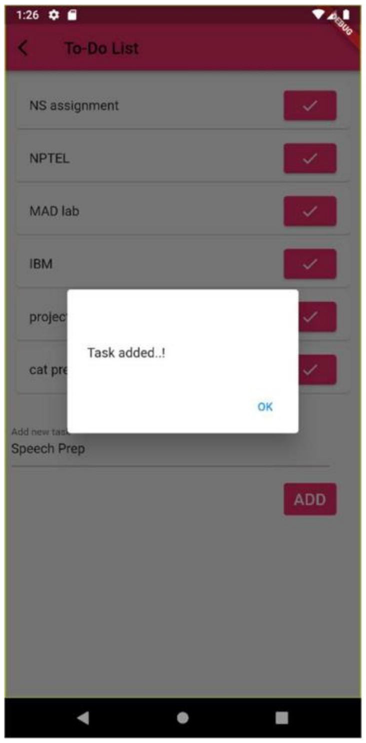
CODE:

```
import 'package:flutter/material.dart';

class SecondRoute extends StatefulWidget {
  @override
  _MyAppState createState() => _MyAppState();
}

class _MyAppState extends State<SecondRoute> {

  List<Task> tasks = [];

  final taskController = TextEditingController();

  @override
  void dispose() {
    // Clean up the controller when the widget is disposed.
    taskController.dispose();
    super.dispose();
  }

  @override
  void initState() {
    tasks.add(Task(tname: "NS assignment"));
    tasks.add(Task(tname: "NPTEL"));
    tasks.add(Task(tname: "MAD lab"));
    tasks.add(Task(tname: "IBM"));
    super.initState();
  }

  @override
  Widget build(BuildContext context) {
```

```
return Scaffold(
    appBar: AppBar(
      leading: GestureDetector(
        child: Icon(
          Icons.arrow_back_ios,
          color: Colors.black,
        ),
        onTap: () {
          Navigator.pop(context);
        },
      ),
      title: Text("To-Do List", style: TextStyle(color: Colors.black)),
      backgroundColor: Color(0xffef2e6c),
    ),
    body: Column(children: [
      SingleChildScrollView(
      child: Container(
          padding:  EdgeInsets.all(10),
          child: Column(
            children: tasks.map((taskone) {
              return Container(
                child:        Card(
                  child: ListTile(
                    title:  Text(taskone.tname),
                    trailing: ElevatedButton(
                      style:  ElevatedButton.styleFrom(
                          primary: Color(0xffef2e6c)),
                      child: Icon(Icons.check),
                      onPressed: () {
                      tasks.removeWhere((element) {
                          return element.tname == taskone.tname;
                        });
                        setState(() {
                          //refresh UI after deleting element from list
                        });
                      },
                    ),
                  ),
                ),
              );
            }).toList(),
          ),
        ),
      ),
      Center(
        child: Container(
            margin: const EdgeInsets.only(right: 30.0),
            child: Column(
              children: [
                Padding(
                  padding: const EdgeInsets.symmetric(
                      horizontal: 8, vertical: 16),
                  child: TextFormField(
                    controller: taskController,
                    decoration: const InputDecoration(
                    border: UnderlineInputBorder(),
                    labelText: 'Add new task',
                    ),
                  ),
                ),
                Align(
                    alignment: Alignment.bottomRight,
                    child: TextButton(
                      style: ButtonStyle(backgroundColor:
MaterialStateProperty.all(Color(0xffef2e6c))),
                      onPressed: () => showDialog<String>(
                        context: context,
                        builder: (BuildContext context) => AlertDialog(
                          title: const Text(''),
                          content: const Text('Task added..!'),
                          actions: <Widget>[
                            TextButton(
                              onPressed: () {
                                debugPrint(taskController.text);
                                tasks.add(Task(tname: taskController.text));
                                taskController.text = "";
                                setState(() {});
                                Navigator.pop(context, 'OK');
                              },
                              child: const Text('OK'),
                            ),
                          ],
                        ),
                      ),
                      child: const Text('ADD',style:TextStyle(color:Colors.white,fontSize: 20.0)),
                    )

                        ),
                  ],
                )),
      )
    ]));
  }
}
```

```
class Task {
  String tname = "";
  Task({required this.tname});
}
```

OUTPUT:



Task added..!

RESULT:

Thus, an application that creates an alert upon receiving a message is implemented using Flutter.

EX.NO:11 **ALARM CLOCK**

AIM:

To write a mobile application that creates an alarm clock.

PROCEDURE:

- Install the flutter_alarm_clock package using
    - flutter pub add flutter_alarm_clock
- Import it using
    - import 'package:flutter_alarm_clock/flutter_alarm_clock.dart';
- The FlutterAlarmClock.createAlarm() that takes hours and minutes as parameters.
- Hours and minutes are taken as input from user, using TextField().
- On clicking on "Create Alarm" button, a snackbar is displayed which appears when an alarm is set.
- The "Show Alarms" button, opens the clock application of the device which shows the created alarms.

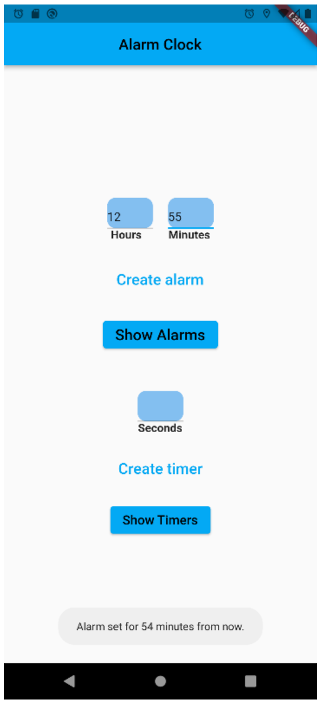
CODE:

```
import 'package:flutter/material.dart';
import 'package:flutter_alarm_clock/flutter_alarm_clock.dart';

class AlarmPage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Flutter Alarm Clock',
      theme: ThemeData(
        primarySwatch: Colors.lightBlue,
      ),
      home: MyHomePage(),
    );
  }
}

class MyHomePage extends StatefulWidget {
  @override
  State<MyHomePage> createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
  TextEditingController hourController = TextEditingController();
  TextEditingController minuteController = TextEditingController();
  TextEditingController secondController = TextEditingController();
  @override
  Widget build(BuildContext context) {
```

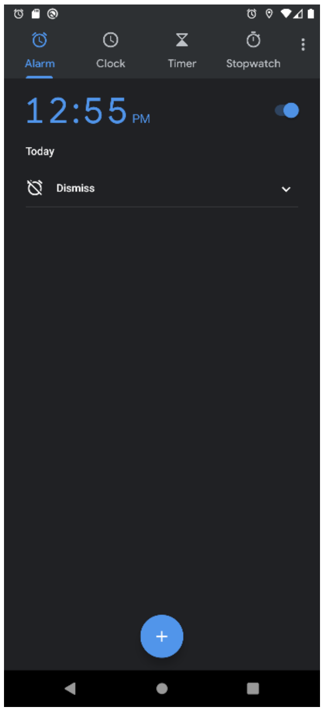
```
return Scaffold(
  appBar: AppBar(
    title: const Text('Alarm Clock'),
    centerTitle: true,
  ),
  body: Center(
      child: SingleChildScrollView(
    child: Column(children: <Widget>[
      const SizedBox(height: 10),
      Row(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          Container(
            height: 40,
            width: 60,
            decoration: BoxDecoration(
                shape: BoxShape.rectangle,
                color: Color.fromARGB(255, 131, 191, 240),
                borderRadius: BorderRadius.circular(11)),
            child: Center(
              child: TextField(
                controller: hourController,
                keyboardType: TextInputType.number,
              ),
            ),
          ),
          const SizedBox(width: 20),
          Container(
            height: 40,
            width: 60,
            decoration: BoxDecoration(
                shape: BoxShape.rectangle,
                color: Color.fromARGB(255, 131, 191, 240),
                borderRadius: BorderRadius.circular(11)),
            child: Center(
              child: TextField(
                controller: minuteController,
                keyboardType: TextInputType.number,
              ),
            ),
          ),
        ],
      ),
      Row(
        mainAxisAlignment: MainAxisAlignment.center,
        children: const [
          Text(
      "Hours",
      style: TextStyle(
        fontSize: 15,
        fontWeight: FontWeight.bold,
      ),
    ),
    SizedBox(width: 35),
    Text(
      "Minutes",
      style: TextStyle(
        fontSize: 15,
        fontWeight: FontWeight.bold,
      ),
    ),
  ],
),
Container(
  margin: const EdgeInsets.all(25),
  child: TextButton(
    child: const Text(
      'Create alarm',
      style: TextStyle(fontSize: 20.0),
    ),
    onPressed: () {
      int hour;
      int minutes;
      hour = int.parse(hourController.text);
      minutes = int.parse(minuteController.text);

      // creating alarm after converting hour
      // and minute into integer
      FlutterAlarmClock.createAlarm(hour, minutes);
    },
  ),
),
ElevatedButton(
  onPressed: () {
    // show alarm
    FlutterAlarmClock.showAlarms();
  },
  child: const Text(
    'Show Alarms',
    style: TextStyle(fontSize: 20.0),
  ),
),
const SizedBox(height: 50),
Container(
  height: 40,
  width: 60,
  decoration: BoxDecoration(
      shape: BoxShape.rectangle,
      color: Color.fromARGB(255, 131, 191, 240),
      borderRadius: BorderRadius.circular(11)),
  child: Center(
    child: TextField(
      controller: secondController,
      keyboardType: TextInputType.number,
    ),
  ),
),
const Text(
  "Seconds",
  style: TextStyle(
    fontSize: 15,
    fontWeight: FontWeight.bold,
  ),
),
Container(
  margin: const EdgeInsets.all(20),
  child: TextButton(
      child: const Text(
        'Create timer',
        style: TextStyle(fontSize: 20.0),
      ),
      onPressed: () {
        int seconds = int.parse(secondController.text.trim());

        // create timer
        FlutterAlarmClock.createTimer(seconds);
        showDialog(
            context: context,
            builder: (context) {
              return AlertDialog(
                contentPadding: EdgeInsets.zero,
                content: Center(
                  child: Text(
                    "Timer is set for \$seconds seconds",
                    style: const TextStyle(
                      fontSize: 20,
                      fontWeight: FontWeight.bold,
                    ),
                  ),
                ),
              );
```

```
                  });
               }),
            ),
            ElevatedButton(
              onPressed: () {
                // show timers
                FlutterAlarmClock.showTimers();
              },
              child: const Text(
                "Show Timers",
                style: TextStyle(fontSize: 17),
              ),
            )
          ]),
        )),
      );
    }
  }
```

OUTPUT:



Alarm set for 54 minutes from now.



RESULT:

Thus, a mobile application that creates an alarm clock is implemented using Flutter.

EX.NO:12 SIMPLE GAME WITH MULTIMEDIA SUPPORT

AIM:

To implement a simple gaming application with multimedia support.

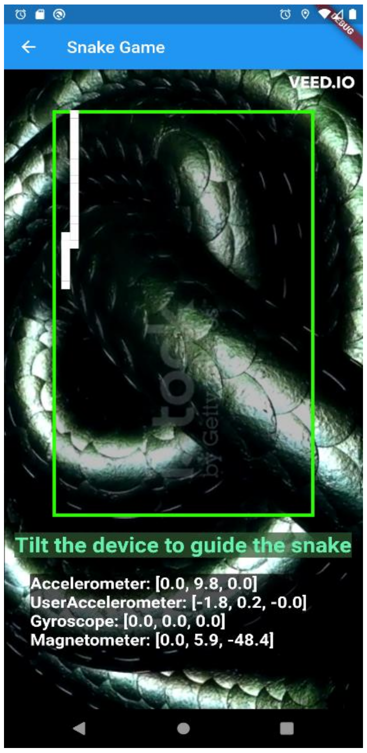
PROCEDURE:

- Create a class TileModel for each tile, which has the following as members
  - ImageAssetPath
  - IsSelected
- Create a list called 'pairs' which contains a pair of each tile of a specific image.
- Use GridView to display the tiles as a 4x4 grid.
- Initialize points as 0 using setState().
- For every matched tile, increment points by 100.
- Play until points == 800.
- Click on replay to restart the game.

CODE:

```
import 'dart:async';

import 'package:flutter/material.dart';
import 'package:news_hub/components/snake.dart';
import 'package:sensors_plus/sensors_plus.dart';
import 'package:video_player/video_player.dart';

class SnakeGamePage extends StatefulWidget {
  const SnakeGamePage({Key? key, this.title}) : super(key: key);

  final String? title;

  @override
  State<SnakeGamePage> createState() => _SnakeGamePageState();
}

class _SnakeGamePageState extends State<SnakeGamePage> {
  // Background video player
  VideoPlayerController? _videoPlayerController;

  //init state
  @override
  void initState() {
    super.initState();

    // bg-video set
    _videoPlayerController =
        VideoPlayerController?.asset("assets/videos/snake_vid.mp4")
        ..initialize().then((_) {
          _videoPlayerController?.play();
          _videoPlayerController?.setLooping(true);
          setState(() {});
        });

  _streamSubscriptions.add(
    accelerometerEvents.listen(
      (AccelerometerEvent event) {
        setState(() {
          _accelerometerValues = <double>[event.x, event.y, event.z];
        });
      },
    ),
  );
  _streamSubscriptions.add(
    gyroscopeEvents.listen(
      (GyroscopeEvent event) {
        setState(() {
          _gyroscopeValues = <double>[event.x, event.y, event.z];
        });
      },
    ),
  );
  _streamSubscriptions.add(
    userAccelerometerEvents.listen(
      (UserAccelerometerEvent event) {
        setState(() {
          _userAccelerometerValues = <double>[event.x, event.y, event.z];
        });
      },
    ),
  );
  _streamSubscriptions.add(
    magnetometerEvents.listen(
      (MagnetometerEvent event) {
        setState(() {
          _magnetometerValues = <double>[event.x, event.y, event.z];
        });
      },
    ),
  );
}

@override
void dispose() {
  _videoPlayerController?.dispose();
  // _audioPlayer.dispose();
  super.dispose();
  for (final subscription in _streamSubscriptions) {
    subscription.cancel();
  }
}

static const int _snakeRows = 50;
static const int _snakeColumns = 30;
static const double _snakeCellSize = 10.0;

List<double>? _accelerometerValues;
List<double>? _userAccelerometerValues;
List<double>? _gyroscopeValues;
List<double>? _magnetometerValues;
final _streamSubscriptions = <StreamSubscription<dynamic>>[];

@override
Widget build(BuildContext context) {
  final accelerometer =
      _accelerometerValues?.map((double v) => v.toStringAsFixed(1)).toList();
  final gyroscope =
      _gyroscopeValues?.map((double v) => v.toStringAsFixed(1)).toList();
  final userAccelerometer = _userAccelerometerValues
      ?.map((double v) => v.toStringAsFixed(1))
      .toList();
  final magnetometer =
      _magnetometerValues?.map((double v) => v.toStringAsFixed(1)).toList();

  return Scaffold(
    appBar: AppBar(
      title: const Text('Snake Game'),
    ),
    body: Stack(
      children: [
        SizedBox.expand(
          child: FittedBox(
            fit: BoxFit.fill,
            child: SizedBox(
              width: _videoPlayerController?.value.size.width,
              height: _videoPlayerController?.value.size.height,
              child: VideoPlayer(_videoPlayerController!),
            ),
          ),
        ),
        Column(
          mainAxisAlignment: MainAxisAlignment.start,
          children: <Widget>[
            const SizedBox(
              height: 50,
            ),
            Center(
              child: DecoratedBox(
                decoration: BoxDecoration(
                  color: Colors.black26,
                  border: Border.all(
                      width: 4.0,
                      color: const Color.fromARGB(255, 43, 255, 0)),
                ),
                child: SizedBox(
                  height: _snakeRows * _snakeCellSize,
                  width: _snakeColumns * _snakeCellSize,
                  child: Snake(
                    rows: _snakeRows,
                    columns: _snakeColumns,
                    cellSize: _snakeCellSize,
                  ),
                ),
              ),
            ),
            const SizedBox(
              height: 20,
            ),
            const Text(
              "Tilt the device to guide the snake",
              style: TextStyle(
                color: Colors.greenAccent,
                fontWeight: FontWeight.bold,
                fontSize: 26,
                backgroundColor: Color.fromARGB(186, 47, 87, 48),
              ),
            ),
            const SizedBox(
              height: 20,
            ),
            Container(
              margin: const EdgeInsets.symmetric(horizontal: 30),
              color: Color.fromARGB(174, 0, 0, 0),
              child: Column(
                mainAxisAlignment: MainAxisAlignment.center,
                crossAxisAlignment: CrossAxisAlignment.center,
        children: [
          Row(
            mainAxisAlignment: MainAxisAlignment.spaceBetween,
            children: <Widget>[
              Text(
                'Accelerometer: $accelerometer',
                style: const TextStyle(
                  color: Colors.white,
                  fontSize: 20,
                  fontWeight: FontWeight.bold,
                ),
              ),
            ],
          ),
          Row(
            mainAxisAlignment: MainAxisAlignment.spaceBetween,
            children: <Widget>[
              Text(
                'UserAccelerometer: $userAccelerometer',
                style: const TextStyle(
                  color: Colors.white,
                  fontSize: 20,
                  fontWeight: FontWeight.bold,
                ),
              ),
            ],
          ),
          Row(
            mainAxisAlignment: MainAxisAlignment.spaceBetween,
            children: <Widget>[
              Text(
                'Gyroscope: $gyroscope',
                style: const TextStyle(
                  color: Colors.white,
                  fontSize: 20,
                  fontWeight: FontWeight.bold,
                ),
              ),
            ],
          ),
          Row(
            mainAxisAlignment: MainAxisAlignment.spaceBetween,
            children: <Widget>[
              Text(
                'Magnetometer: $magnetometer',
                style: const TextStyle(
              color: Colors.white,
              fontSize: 20,
              fontWeight: FontWeight.bold,
             ),
            ),
           ],
          ),
         ],
        ),
       )
      ],
     ),
    ],
   ),
  );
 }
}
```

OUTPUT:



```
Tilt the device to guide the snake
Accelerometer: [0.0, 9.8, 0.0]
UserAccelerometer: [-1.8, 0.2, -0.0]
Gyroscope: [0.0, 0.0, 0.0]
Magnetometer: [0.0, 5.9, -48.4]
```

RESULT:

Thus, a simple gaming application that supports multimedia is implemented using Flutter.

EX.NO:13 **CONNECTIVITY VIA SOAP OR REST**

AIM:

To a mobile application for data handling and connectivity via SOAP or REST to backend services potentially hosted in a cloud environment.

PROCEDURE:

- Import,
    - http.dart
    - dart:convert
- Specify the URL of the API within “Uri.*parse*(<>)”
- http.get() is used to fetch url contents.

CODE:

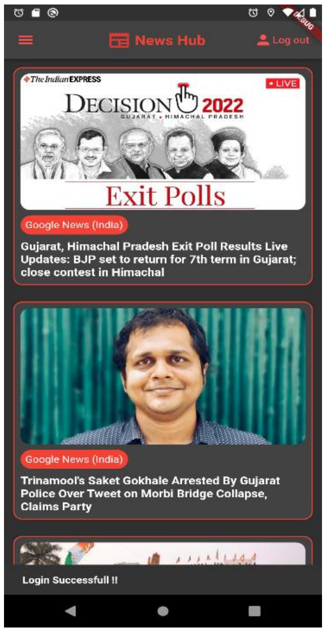
api_service.dart

```
import 'dart:convert';


import 'package:http/http.dart';
import 'package:news_hub/models/article_model.dart';


class ApiService {


  final endPointUrl = Uri.parse("https://newsapi.org/v2/top-
headlines?sources=google-news-in,the-hindu,the-times-of-
india&apiKey=14684e90b6344a4e8cbe486d022ae516");


  Future<List<Article>> getArticle() async {
    Response res = await get(endPointUrl);


    if (res.statusCode == 200) {
      Map<String, dynamic> json = jsonDecode(res.body);


      List<dynamic> body = json['articles'];


      List<Article> articles =
          body.map<Article>((dynamic item) =>
    Article.fromJSON(item)).toList();
          return articles;
        } else {
          throw ("Can't get the Articles");
        }
      }
    }
```

Home_page.dart

```
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:news_hub/components/custom_list_tile.dart';
import 'package:news_hub/models/article_model.dart';
import 'package:news_hub/screens/accelerometer.dart';
import 'package:news_hub/screens/alarm.dart';
import 'package:news_hub/screens/calculator.dart';
import 'package:news_hub/screens/geolocator.dart';
import 'package:news_hub/screens/shapes.dart';
import 'package:news_hub/screens/snake_game.dart';
import 'package:news_hub/services/api_service.dart';

class HomePage extends StatefulWidget {
  const HomePage({
    Key? key,
  }) : super(key: key);

  @override
  State<HomePage> createState() => _HomePageState();
}

class _HomePageState extends State<HomePage> {
  ApiService client = ApiService();

  final userEmail = FirebaseAuth.instance.currentUser?.email.toString();

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.grey[850],
      appBar: AppBar(
        foregroundColor: Colors.redAccent,
        backgroundColor: Colors.grey[800],
        title: Row(
          mainAxisAlignment: MainAxisAlignment.center,
    children: [
      Icon(Icons.newspaper_rounded, color: Colors.red[600], size: 30.0),
      const Text(
        " News Hub",
        style: TextStyle(
            color: Colors.red,
            fontWeight: FontWeight.bold,
            fontSize: 20.0),
      ),
    ],
  ),
  centerTitle: true,
  actions: [
    MaterialButton(
      onPressed: () {
        FirebaseAuth.instance.signOut();
      },
      child: Row(
        children: const [
          Icon(
            Icons.person,
            color: Colors.redAccent,
          ),
          Text(
            "Log out",
            style: TextStyle(
              color: Colors.redAccent,
            ),
          ),
        ],
      ),
    )
  ],
),
drawer: Drawer(
  backgroundColor: Colors.grey[850],
  child: ListView(
    // Important: Remove any padding from the ListView.
    padding: EdgeInsets.zero,
    children: [
      UserAccountsDrawerHeader(
        decoration: BoxDecoration(color: Colors.grey[800]),
        accountName: const Text(
          "Email-Id",
          style: TextStyle(
            fontWeight: FontWeight.bold,
            color: Colors.redAccent,
    ),
  ),
  accountEmail: Text(
    userEmail.toString(),
    style: const TextStyle(
      fontWeight: FontWeight.bold,
    ),
  ),
  currentAccountPicture: const CircleAvatar(
    backgroundColor: Colors.red,
    child: Icon(Icons.person, size: 65.0, color: Colors.white),
  ),
),
ListTile(
  leading: const Icon(
    Icons.format_shapes,
    color: Colors.red,
  ),
  title: const Text(
    'Shapes',
    style: TextStyle(
      color: Colors.red,
    ),
  ),
  onTap: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (context) {
          return ShapesPage();
        },
      ),
    );
  },
),
ListTile(
  leading: const Icon(
    Icons.calculate,
    color: Colors.red,
  ),
  title: const Text(
    'Calculator',
    style: TextStyle(
      color: Colors.red,
    ),
  ),
  onTap: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (context) {
          return const CalculatorPage();
        },
      ),
    );
  },
),
ListTile(
  leading: const Icon(
    Icons.alarm,
    color: Colors.red,
  ),
  title: const Text(
    "Alarm Clock",
    style: TextStyle(
      color: Colors.red,
    ),
  ),
  onTap: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (context) {
          return AlarmPage();
        },
      ),
    );
  },
),
ListTile(
  leading: const Icon(
    Icons.pin_drop,
    color: Colors.red,
  ),
  title: const Text(
    'Geolocation',
    style: TextStyle(
      color: Colors.red,
    ),
  ),
  onTap: () {
    Navigator.push(
      context,
      MaterialPageRoute(
            builder: (context) {
              return const GeolocatorPage();
            },
          ),
        );
      },
    ),
    ListTile(
      leading: const Icon(
        Icons.speed,
        color: Colors.red,
      ),
      title: const Text(
        'Accelerometer',
        style: TextStyle(
          color: Colors.red,
        ),
      ),
      onTap: () {
        Navigator.push(
          context,
          MaterialPageRoute(
            builder: (context) {
              return const AccelerometerPage(title: 'Accelerometer');
            },            },
    ),
    ListTile(
      leading: const Icon(
        Icons.gamepad,
        color: Colors.red,
      ),
      title: const Text(
        'Snake Game',
        style: TextStyle(
          color: Colors.red,
        ),
      ),
      onTap: () {
        Navigator.push(
          context,
          MaterialPageRoute(
            builder: (context) {
              return const SnakeGamePage(title: 'Snake Game');
            },
          ),
  ),
),
      body: FutureBuilder(
        future: client.getArticle(),
        builder: (BuildContext context, AsyncSnapshot snapshot) {
          if (snapshot.hasData) {
            List<Article>? articles = snapshot.data;

            return ListView.builder(
              itemCount: articles?.length,
              itemBuilder: (context, index) =>
                  customListTile(articles![index], context),
            );
          }
          return const Center(
              child: CircularProgressIndicator(
            color: Colors.redAccent,
          ));
    }
  }
```

OUTPUT:



Gujarat, Himachal Pradesh Exit Poll Results Live Updates: BJP set to return for 7th term in Gujarat; close contest in Himachal

Trinamool's Saket Gokhale Arrested By Gujarat Police Over Tweet on Morbi Bridge Collapse, Claims Party

Login Successfull !!

RESULT:

Hence, a mobile application for data handling and connectivity via SOAP or REST to backend services potentially hosted in a cloud environment.

EX.NO:14 GEO-POSITIONING, ACCELEROMETER AND RICH GESTURE BASED UI

AIM:

To write a mobile application that will take advantage of underlying phone functionality including GEO positioning, accelerometer, and rich gesture-based UI handling.

PROCEDURE:

Geo-positioning:

- Install the following packages: geolocator & geocoding
- Import them using,
    - import 'package:geocoding/geocoding.dart';
    - import 'package:geolocator/geolocator.dart';
- Get current location of the device, by creating an instance of Geolocator and calling getCurrentPosition.
- Convert latitude and longitude values into address using placemarkFromCoordinates().

Accelerometer:

- Install the sensors package.
- Import it using, 'import 'package:sensors/sensors.dart';'
- accelerometer readings tell if the device is moving in a particular direction.

Gesture-based UI:

- In the onTap() property of the GestureDetector(), pass the function to be performed.
- In this case, it reverses the boolean value isLightsOn.
- This is used to switch the theme of the screen as dark or light.
- The child property of GestureDetector() is used to specify icon, on clicking which the action is to be performed.

Geo-positioning:

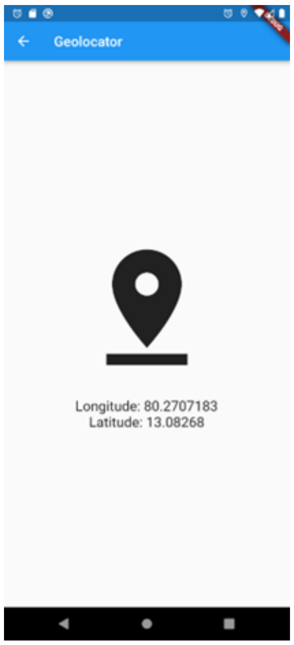
CODE:

```
import 'dart:async';

import 'package:flutter/material.dart';
import 'package:geolocator/geolocator.dart';

class GeolocatorPage extends StatefulWidget {
  const GeolocatorPage({Key? key}) : super(key: key);

  @override
  _GeolocatorPageState createState() => _GeolocatorPageState();
}
```

```
class _GeolocatorPageState extends State<GeolocatorPage> {
  bool servicestatus = false;
  bool haspermission = false;
  late LocationPermission permission;
  late Position position;
  String long = "", lat = "";
  late StreamSubscription<Position> positionStream;

  @override
  void initState() {
    checkGps();
    super.initState();
  }

  checkGps() async {
    servicestatus = await Geolocator.isLocationServiceEnabled();
    if (servicestatus) {
      permission = await Geolocator.checkPermission();

      if (permission == LocationPermission.denied) {
        permission = await Geolocator.requestPermission();
        if (permission == LocationPermission.denied) {
          print('Location permissions are denied');
        } else if (permission == LocationPermission.deniedForever) {
          print("'Location permissions are permanently denied");
        } else {
          haspermission = true;
        }
      } else {
        haspermission = true;
      }

      if (haspermission) {
        // setState(() {
        //   //refresh the UI
        // });

        getLocation();
      }
    } else {
      print("GPS Service is not enabled, turn on GPS location");
    }

    // setState(() {
    //   //refresh the UI
    // });
  }
```

```
getLocation() async {
  position = await Geolocator.getCurrentPosition(
      desiredAccuracy: LocationAccuracy.high);
  print(position.longitude); //Output: 80.24599079
  print(position.latitude); //Output: 29.6593457

  long = position.longitude.toString();
  lat = position.latitude.toString();

  setState(() {
    //refresh UI
  });

  LocationSettings locationSettings = LocationSettings(
    accuracy: LocationAccuracy.high, //accuracy of the location data
    distanceFilter: 100, //minimum distance (measured in meters) a
    //device must move horizontally before an update event is generated;
  );

  StreamSubscription<Position> positionStream =
      Geolocator.getPositionStream(locationSettings: locationSettings)
          .listen((Position position) {
    print(position.longitude); //Output: 80.24599079
    print(position.latitude); //Output: 29.6593457

    long = position.longitude.toString();
    lat = position.latitude.toString();

    setState(() {
      //refresh UI on update
    });
  });
}

@override
Widget build(BuildContext context) {
  return MaterialApp(
    home: Scaffold(
      appBar: AppBar(
        leading: IconButton(
          icon: const Icon(Icons.arrow_back, color: Colors.white),
          onPressed: () => Navigator.of(context).pop(),
        ),
        title: const Text('Geolocator'),
      ),
      body: Padding(
        padding: const EdgeInsets.all(18.0),
        child: SizedBox(
          width: double.infinity,
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          const Icon(
            Icons.pin_drop,
            size: 200.0,
          ),
          const SizedBox(
            height: 30.0,
          ),
          Text("Longitude: $long",
              style: const TextStyle(
                fontSize: 20,
              )),
          Text(
            "Latitude: $lat",
            style: const TextStyle(
              fontSize: 20,
            ),
          ),
        ],
      ),
    ),
   ),
  ),
 );
}
}
```

OUTPUT:



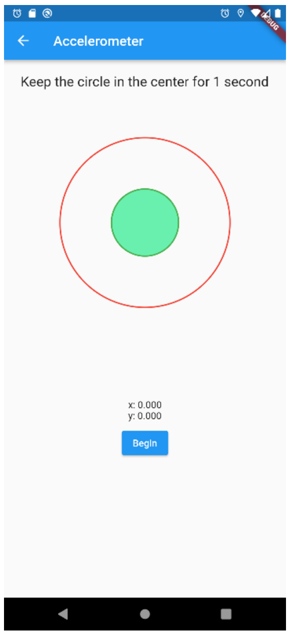
Longitude: 80.2707183
Latitude: 13.08268

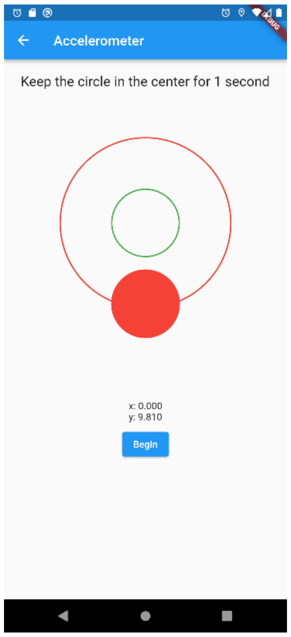
<u>Accelerometer:</u>

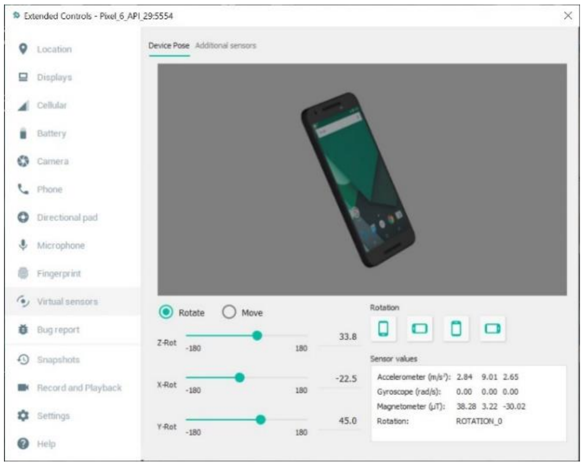
CODE:

```
import 'dart:async';

import 'package:flutter/material.dart';
import 'package:sensors/sensors.dart';


class FocusPage extends StatefulWidget {

  final String title='Focus!';

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

class FocusPageState extends State<FocusPage> {
  // color of the circle
  Color color = Colors.greenAccent;

  // event returned from accelerometer stream
  AccelerometerEvent? event;

  // hold a refernce to these, so that they can be disposed
  Timer? timer;
  StreamSubscription? accel;

  // positions and count
  double top = 125;
  double? left;
  int count = 0;

  // variables for screen size
  double? width;
  double? height;

  setColor(AccelerometerEvent event) {
    // Calculate Left
    double x = ((event.x * 12) + ((width! - 100) / 2));
    // Calculate Top
    double y = event.y * 12 + 125;

    // find the difference from the target position
    var xDiff = x.abs() - ((width! - 100) / 2);
    var yDiff = y.abs() - 125;

    // check if the circle is centered, currently allowing a buffer of 3 to make centering easier
    if (xDiff.abs() < 3 && yDiff.abs() < 3) {
      // set the color and increment count
      setState(() {
        color = Colors.greenAccent;
        count += 1;
      });
    } else {
      // set the color and restart count
      setState(() {
        color = Colors.red;
        count = 0;
      });
    }
  }

  setPosition(AccelerometerEvent event) {
    if (event == null) {
      return;
    }

    // When x = 0 it should be centered horizontally
    // The left positin should equal (width - 100) / 2
    // The greatest absolute value of x is 10, multipling it by 12 allows the left position to move
a total of 120 in either direction.
    setState(() {
      left = ((event.x * 12) + ((width! - 100) / 2));
    });

    // When y = 0 it should have a top position matching the target, which we set at 125
    setState(() {
      top = event.y * 12 + 125;
    });
  }


  startTimer() {
    // if the accelerometer subscription hasn't been created, go ahead and create it
    if (accel == null) {
      accel = accelerometerEvents.listen((AccelerometerEvent eve) {
        setState(() {
          event = eve;
        });
      });
    } else {
      // it has already ben created so just resume it
      accel?.resume();
    }
    // Accelerometer events come faster than we need them so a timer is used to only proccess
them every 200 milliseconds
    if (timer == null || !timer!.isActive) {
      timer = Timer.periodic(Duration(milliseconds: 200), (_) {
        // if count has increased greater than 3 call pause timer to handle success
        if (count > 3) {
          pauseTimer();
        } else {
          // proccess the current event
          setColor(event!);
          setPosition(event!);
        }
      });
    }
  }

  pauseTimer() {
    // stop the timer and pause the accelerometer stream
    timer?.cancel();
    accel?.pause();

    // set the success color and reset the count
    setState(() {
      count = 0;
      color = Colors.green;
    });
  }

  @override
  void dispose() {
    timer?.cancel();
    accel?.cancel();
    super.dispose();
  }

  @override
  Widget build(BuildContext context) {
    // get the width and height of the screen
    width = MediaQuery.of(context).size.width;
    height = MediaQuery.of(context).size.height;

    return Scaffold(
      appBar: AppBar(
        iconTheme: IconThemeData(
          color: Colors.black, //change your color here
        ),
        title: Text(widget.title,style:TextStyle(color:Colors.black)),
        backgroundColor : Color(0xffef2e6c),
      ),
      body: Column(
        children: [
        Padding(
            padding: const EdgeInsets.all(8.0),
            child: Text('Keep the circle in the center for 1 second',textAlign:
TextAlign.center,style: TextStyle(fontSize:25)),
          ),
          Stack(
            children: [
              // This empty container is given a width and height to set the size of the stack
              Container(
                height: height! / 2,
                width: width,
              ),

              // Create the outer target circle wrapped in a Position
              Positioned(
                // positioned 50 from the top of the stack
                // and centered horizontally, left = (ScreenWidth - Container width) / 2
                top: 50,
                left: (width! - 250) / 2,
                child: Container(
                height: 250,
                  width: 250,
                  decoration: BoxDecoration(
                    border: Border.all(color: Colors.red, width: 5.0),
                    borderRadius: BorderRadius.circular(125),
                  ),
                ),
              ),
              // This is the colored circle that will be moved by the accelerometer
              // the top and left are variables that will be set
              Positioned(
                top: top,
                left: left ?? (width! - 100) / 2,
                // the container has a color and is wrappeed in a ClipOval to make it round
                child: ClipOval(
                  child: Container(
                  width: 100,
                    height: 100,
                    color: color,
                  ),
                ),
              ),
              // inner target circle wrapped in a Position
              Positioned(
                top: 125,
                left: (width! - 100) / 2,
                child: Container(
                height: 100,
                  width: 100,
                  decoration: BoxDecoration(
                    border: Border.all(color: Colors.green, width: 2.0),
                    borderRadius: BorderRadius.circular(50),
                  ),
                ),
              ),
            ],
          ),
          Text('x: ${(event?.x ?? 0).toStringAsFixed(3)}',style:TextStyle(fontSize: 20)),
          Text('y: ${(event?.y ?? 0).toStringAsFixed(3)}',style:TextStyle(fontSize: 20)),
          Padding(
            padding: EdgeInsets.symmetric(horizontal: 16.0, vertical: 30.0),
            child: TextButton(
              style: ButtonStyle(backgroundColor: MaterialStateProperty.all(Color(0xffef2e6c))),
              onPressed: startTimer,
              child: Text('Begin.!!',style: TextStyle(fontSize: 30.0,color:Colors.white),),
              // color: Theme.of(context).primaryColor,
              // textColor: Colors.white,
            ),
          )
        ],
      ),
    );
   }
}
```

OUTPUT:







Z-Rot 33.8
X-Rot -22.5
Y-Rot 45.0

Sensor values

```
Accelerometer (m/s²):  2.84   9.01  2.65
Gyroscope (rad/s):     0.00   0.00  0.00
Magnetometer (µT):     38.28  3.22  -30.02
Rotation:              ROTATION_0
```

RESULT:

Thus, GEO positioning, accelerometer, and rich gesture-based UI handling have been implemented using Flutter.

EX.NO:15 **SOCIAL MEDIA INTEGRATION**

AIM:

To write an application for integrating mobile applications in the market, including social networking software integration with Google.

PROCEDURE:

- Download the following packages using flutter pub add.
  - firebase_auth
  - firebase_core
  - google_sign_in
- In the firebase console, enable Google as a provider under Authentication-> Sign In method.
- Get SHA key, by using the command gradlew signingReport at the android directory of the flutter application.
- Add SHA-1 fingerprint to the application.
- Now, get Google user credential using the await GoogleSignIn().signIn();
- Obtain the auth details from the request.
- Obtain the auth details from the request

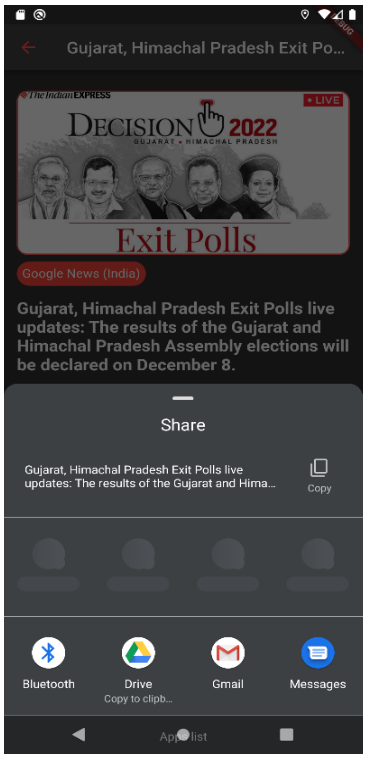
CODE:

<u>Share_article.dart</u>

```
import 'dart:io';

import 'dart:typed_data';


import 'package:flutter/material.dart';

import 'package:news_hub/models/article_model.dart';

import 'package:path_provider/path_provider.dart';

import 'package:screenshot/screenshot.dart';

import 'package:share_plus/share_plus.dart';


class ArticlePage extends StatefulWidget {

  final Article article;

  ArticlePage({required this.article});
  @override
  State<ArticlePage> createState() => _ArticlePageState();
}

class _ArticlePageState extends State<ArticlePage> {
  ScreenshotController screenshotController = ScreenshotController();

  snackbar_message(String text) {
    var snackBar = SnackBar(
        content: Text(
      text,
      style: TextStyle(fontWeight: FontWeight.bold),
    ));
    ScaffoldMessenger.of(context).showSnackBar(snackBar);
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
        appBar: AppBar(
          title: Text(
            widget.article.title,
            style: const TextStyle(color: Colors.white),
          ),
          backgroundColor: Colors.grey[800],
          foregroundColor: Colors.redAccent,
        ),
        body: Screenshot(
          controller: screenshotController,
child: Container(
  padding:
      const EdgeInsets.symmetric(horizontal: 15.0, vertical: 25.0),
  color: Colors.grey[850],
  child: Column(
    mainAxisAlignment: MainAxisAlignment.start,
    crossAxisAlignment: CrossAxisAlignment.start,
    children: [
      Container(
        height: 200.0,
        width: double.infinity,
        decoration: BoxDecoration(
            image: DecorationImage(
                image: NetworkImage(widget.article.urlToImage),
                fit: BoxFit.cover),
            borderRadius: BorderRadius.circular(12.0),
            border: Border.all(
              color: Colors.redAccent,
              width: 1.5,
            )),
      ),
      const SizedBox(
        height: 8.0,
      ),
      Container(
        padding: const EdgeInsets.all(6.0),
        decoration: BoxDecoration(
          color: Colors.red,
          borderRadius: BorderRadius.circular(30.0),
  ),
  child: Text(
    widget.article.source.name,
    style: const TextStyle(
      color: Colors.white,
      fontSize: 15.0,
    ),
  ),
),
const SizedBox(
  height: 15.0,
),
Text(
  widget.article.desc,
  style: const TextStyle(
    color: Colors.white,
    fontWeight: FontWeight.bold,
    fontSize: 20.0,
  ),
),
const SizedBox(
  height: 50,
),
Row(
  mainAxisAlignment: MainAxisAlignment.end,
  children: [
    FloatingActionButton(
      backgroundColor: Colors.redAccent,
      heroTag: "save_news",
  onPressed: () async {
    final directory = await getExternalStorageDirectory();
    String timestamp =
        DateTime.now().microsecondsSinceEpoch.toString();
    final filePath = await File(
            '${directory?.path}/${widget.article.title}_$timestamp.txt')
        .create();
    filePath.writeAsString(widget.article.desc);
    snackbar_message("News Saved Successfully !!");
  },
  child: const Icon(
    Icons.save,
    color: Colors.black,
    size: 35,
  ),
),
const SizedBox(
  width: 10,
),
FloatingActionButton(
  backgroundColor: Colors.redAccent,
  heroTag: "screenshot",
  onPressed: () async {
    await screenshotController
        .capture(delay: const Duration(milliseconds: 10))
        .then((Uint8List? image) async {
      if (image != null) {
        final directory =
            await getExternalStorageDirectory();
        String timestamp = DateTime.now()
            .microsecondsSinceEpoch
            .toString();
        final imagePath = await File(
                '${directory?.path}/${widget.article.title}_$timestamp.png')
            .create();
        await imagePath.writeAsBytes(image);
        snackbar_message(
            "Screenshot Saved Successfully !!");
      }
    });
  },
  child: const Icon(
    Icons.screenshot,
    color: Colors.black,
    size: 35,
  ),
),
const SizedBox(
  width: 10,
),
FloatingActionButton(
  backgroundColor: Colors.redAccent,
  heroTag: "share",
  onPressed: () {
    Share.share(
      widget.article.desc,
      subject: widget.article.title,
    );
```

```
                      },
                      child: const Icon(
                        Icons.share,
                        color: Colors.black,
                        size: 35,
                      ),
                    ),
        ));
  }
}
```

OUTPUT:



Gujarat, Himachal Pradesh Exit Po...

Google News (India)

Gujarat, Himachal Pradesh Exit Polls live updates: The results of the Gujarat and Himachal Pradesh Assembly elections will be declared on December 8.

Share

Gujarat, Himachal Pradesh Exit Polls live updates: The results of the Gujarat and Hima...

Copy

Bluetooth

Drive
Copy to clipb...

Gmail

Messages

RESULT:

Thus, an application that uses social networking software (Google) for authentication has been implemented.