Practical -1

Aim:- You are building a mobile application where users navigate through multiple screens like login, dashboard, and profile. Create a basic multi-screen Flutter app with navigation, passing data between pages.

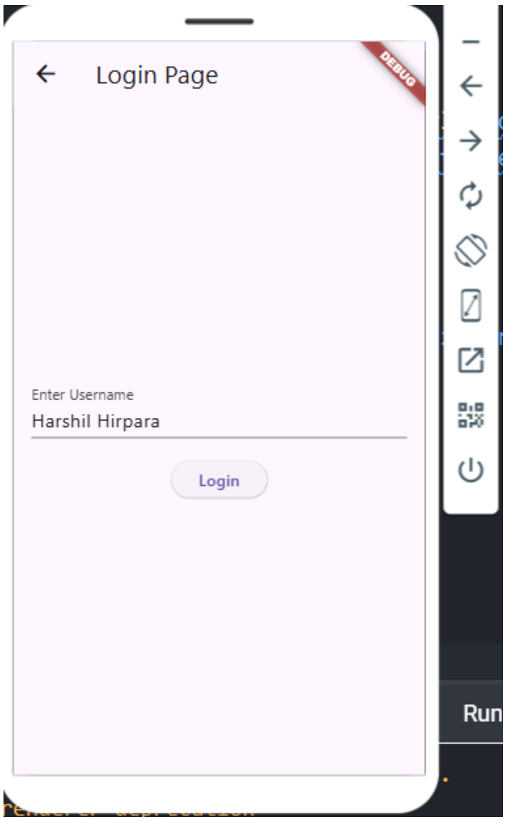
Code:-

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


void main() {
  runApp(MyApp());
}


class User {
  final String username;
  final String email;


  User({required this.username, required this.email});
}


class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Multi-Screen App',
      theme: ThemeData(primarySwatch: Colors.blue),
      home: LoginPage(),
    );
  }
}
```

```
class LoginPage extends StatelessWidget {
  final TextEditingController _usernameController = TextEditingController();

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("Login Page"),
        leading: IconButton(
          icon: Icon(Icons.arrow_back),
          onPressed: () => Navigator.of(context, rootNavigator: true).pop(),
        ),
      ),
      body: Padding(
        padding: EdgeInsets.all(16),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            TextField(
              controller: _usernameController,
              decoration: InputDecoration(labelText: "Enter Username"),
            ),
            SizedBox(height: 20),
            ElevatedButton(
              child: Text("Login"),
              onPressed: () {
                String username = _usernameController.text;
                Navigator.push(
              context,
              MaterialPageRoute(
                builder: (context) => DashboardPage(username: username),
              ),
            );
          },
        ),
      ],
    ),
  ),
 );
}
}

class DashboardPage extends StatelessWidget {
  final String username;

  DashboardPage({required this.username});

  @override
  Widget build(BuildContext context) {
    User user = User(username: username, email: "$username@example.com");

    return Scaffold(
      appBar: AppBar(
        title: Text("Dashboard"),
        leading: IconButton(
          icon: Icon(Icons.arrow_back),
          onPressed: () => Navigator.of(context, rootNavigator: true).pop(),
    ),
   ),
   body: Center(
    child: Column(
     mainAxisAlignment: MainAxisAlignment.center,
     children: [
      Text("Welcome, $username!", style: TextStyle(fontSize: 24)),
      SizedBox(height: 20),
      ElevatedButton(
       child: Text("Go to Profile"),
       onPressed: () {
        Navigator.push(
         context,
         MaterialPageRoute(
          builder: (context) => ProfilePage(user: user),
         ),
        );
       },
      ),
     ],
    ),
   ),
  );
 }
}
class ProfilePage extends StatelessWidget {
 final User user;

 ProfilePage({required this.user});
```

```
@override
Widget build(BuildContext context) {
 return Scaffold(
  appBar: AppBar(
   title: Text("Profile"),
   leading: IconButton(
    icon: Icon(Icons.arrow_back),
    onPressed: () => Navigator.pop(context),
   ),
  ),
  body: Center(
   child: Column(
    mainAxisAlignment: MainAxisAlignment.center,
    children: [
     Text("Username: ${user.username}", style: TextStyle(fontSize: 22)),
     Text("Email: ${user.email}", style: TextStyle(fontSize: 18)),
     SizedBox(height: 20),
     ElevatedButton(
      child: Text("Back to Login"),
      onPressed: () {
       Navigator.popUntil(context, (route) => route.isFirst);
      },
     ),
    ],
   ),
  ),
 );
} }
```

Output:-

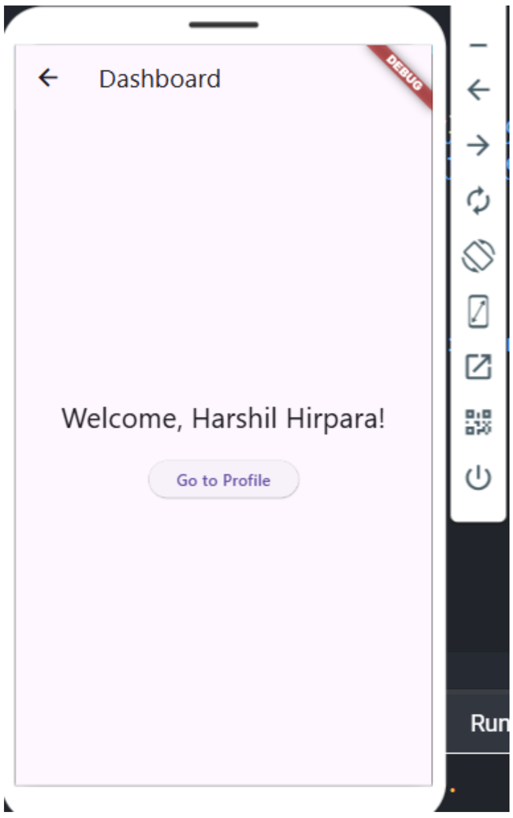


Login Page

Enter Username

Harshil Hirpara

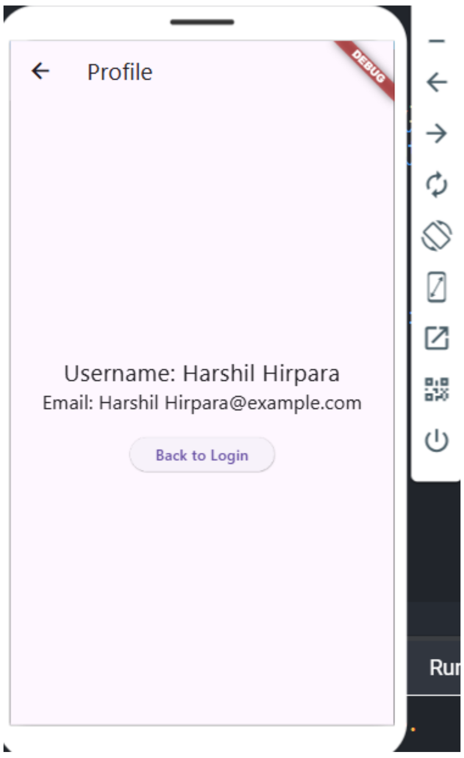
Login



Dashboard

Welcome, Harshil Hirpara!

Go to Profile



Profile

Username: Harshil Hirpara

Email: Harshil Hirpara@example.com

Back to Login

Practical – 2

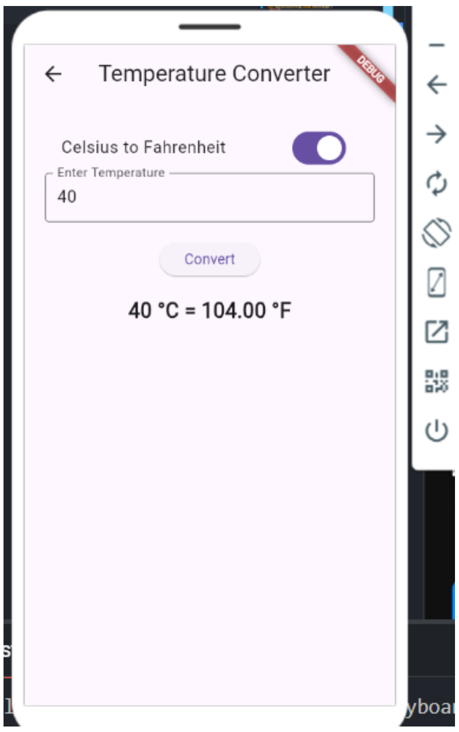
Aim:- You are building a mobile application where users navigate through multiple screens like login, dashboard, and profile. Develop a temperature converter app using Dart functions and input widgets.

Code:-

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


void main() {
  runApp(TemperatureConverterApp());
}


class TemperatureConverterApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Temperature Converter',
      theme: ThemeData(primarySwatch: Colors.teal),
      home: TemperatureConverter(),
    );
  }
}


class TemperatureConverter extends StatefulWidget {
  @override
  _TemperatureConverterState createState() => _TemperatureConverterState();
}


class _TemperatureConverterState extends State<TemperatureConverter> {
```

```
TextEditingController _controller = TextEditingController();
String _result = "";
bool _isCelsiusToFahrenheit = true;

String convertTemperature(String input) {
  double? temp = double.tryParse(input);
  if (temp == null) return "Invalid input!";
  double converted;

  if (_isCelsiusToFahrenheit) {
    converted = (temp * 9 / 5) + 32;
    return "$temp °C = ${converted.toStringAsFixed(2)} °F";
  } else {
    converted = (temp - 32) * 5 / 9;
    return "$temp °F = ${converted.toStringAsFixed(2)} °C";
  }
}

void handleConversion() {
  setState(() {
    _result = convertTemperature(_controller.text);
  });
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text('Temperature Converter'),
    leading: IconButton(
     icon: Icon(Icons.arrow_back),
     onPressed: () => Navigator.of(context, rootNavigator: true).pop(),
    ),
   ),
   body: Padding(
    padding: const EdgeInsets.all(20.0),
    child: Column(
     children: [
      SwitchListTile(
       title: Text(
        _isCelsiusToFahrenheit
            ? 'Celsius to Fahrenheit'
            : 'Fahrenheit to Celsius',
       ),
       value: _isCelsiusToFahrenheit,
       onChanged: (bool value) {
        setState(() {
         _isCelsiusToFahrenheit = value;
        });
       },
      ),
      TextField(
       controller: _controller,
       keyboardType: TextInputType.number,
       decoration: InputDecoration(
        labelText: 'Enter Temperature',
        border: OutlineInputBorder(),
       ),
          ),
          SizedBox(height: 20),
          ElevatedButton(onPressed: handleConversion, child: Text('Convert')),
          SizedBox(height: 20),
          Text(
            _result,
            style: TextStyle(fontSize: 20, fontWeight: FontWeight.bold),
          ),
         ],
        ),
      ),
    );
  }
}
```

Output:-


Temperature Converter

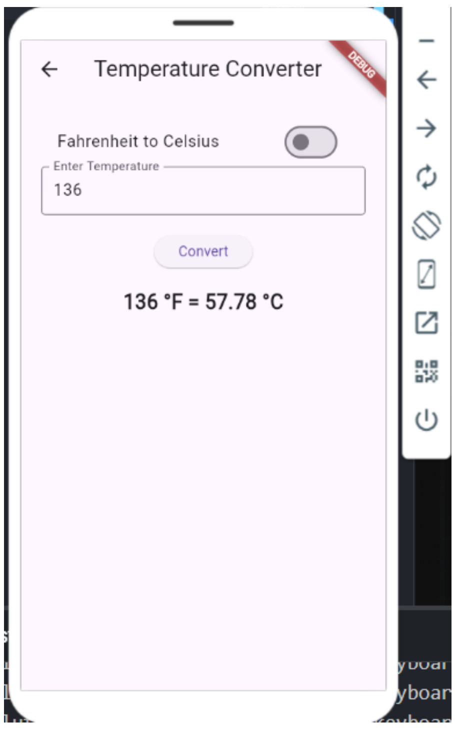
Celsius to Fahrenheit

Enter Temperature

40

Convert

40 °C = 104.00 °F


Temperature Converter

Fahrenheit to Celsius

Enter Temperature

136

Convert

136 °F = 57.78 °C

Practical – 3

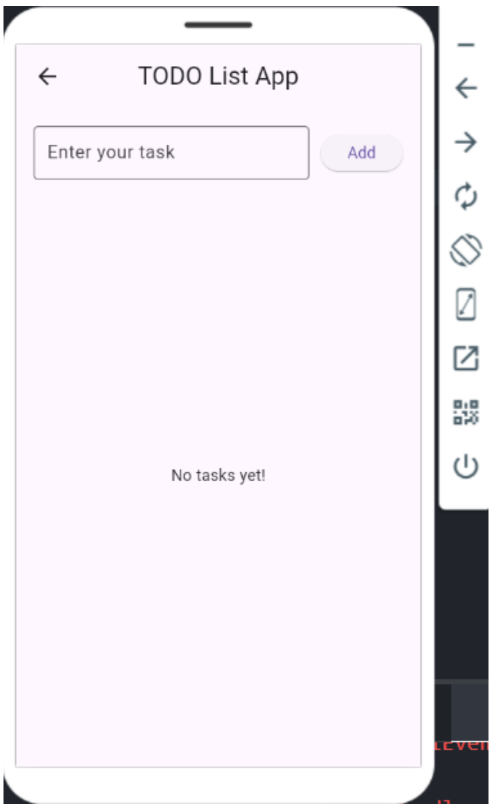
Aim:- You are building a mobile application where users navigate through multiple screens like login, dashboard, and profile. Create a dynamic TODO app using State Management (setState) and ListView.builder.

Code:-

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


void main() {
  runApp(TodoApp());
}


class TodoApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'TODO App',
      debugShowCheckedModeBanner: false,
      theme: ThemeData(primarySwatch: Colors.blueGrey),
      home: TodoHomePage(),
    );
  }
}


class TodoHomePage extends StatefulWidget {
  @override
  _TodoHomePageState createState() => _TodoHomePageState();
}
```

```
class _TodoHomePageState extends State<TodoHomePage> {
 List<String> tasks = [];

 TextEditingController taskController = TextEditingController();

 void addTask() {
  String task = taskController.text.trim();
  if (task.isNotEmpty) {
   setState(() {
    tasks.add(task);
    taskController.clear();
   });
  }
 }

 void deleteTask(int index) {
  setState(() {
   tasks.removeAt(index);
  });
 }

 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    title: Text("TODO List App"),
    centerTitle: true,
    leading: IconButton(
     icon: Icon(Icons.arrow_back),
          onPressed: () => Navigator.of(context, rootNavigator: true).pop(),
        ),
      ),
      body: Padding(
        padding: EdgeInsets.all(16.0),
        child: Column(
          children: [
            Row(
              children: [
                Expanded(
                  child: TextField(
                    controller: taskController,
                    decoration: InputDecoration(
                      labelText: "Enter your task",
                      border: OutlineInputBorder(),
                    ),
                  ),
                ),
                SizedBox(width: 10),
                ElevatedButton(onPressed: addTask, child: Text("Add")),
              ],
            ),
            SizedBox(height: 20),

            Expanded(
              child: tasks.isEmpty
                  ? Center(child: Text("No tasks yet!"))
                  : ListView.builder(
                      itemCount: tasks.length,
           itemBuilder: (context, index) {
            return Card(
              margin: EdgeInsets.symmetric(vertical: 5),
              child: ListTile(
                title: Text(tasks[index]),
                trailing: IconButton(
                  icon: Icon(Icons.delete, color: Colors.red),
                  onPressed: () => deleteTask(index),
                ),
              ),
            );
          },
         ),
     ),
    ],
   ),
  ),
 );
}
}
```

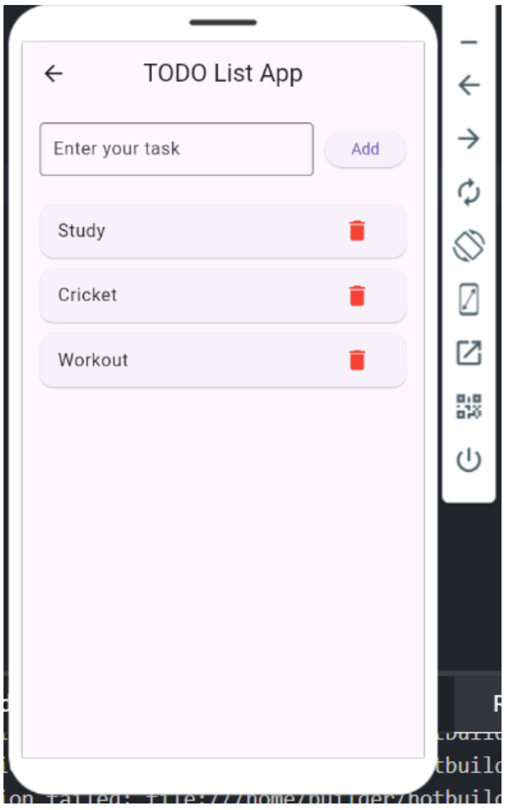
Output:-


TODO List App

Enter your task

Add

No tasks yet!


TODO List App

Enter your task

Add

Study

Cricket

Workout

Practical – 4

Aim:- You are building a mobile application where users navigate through multiple screens like login, dashboard, and profile. Design a Form-based Registration App with validation using TextFormField.

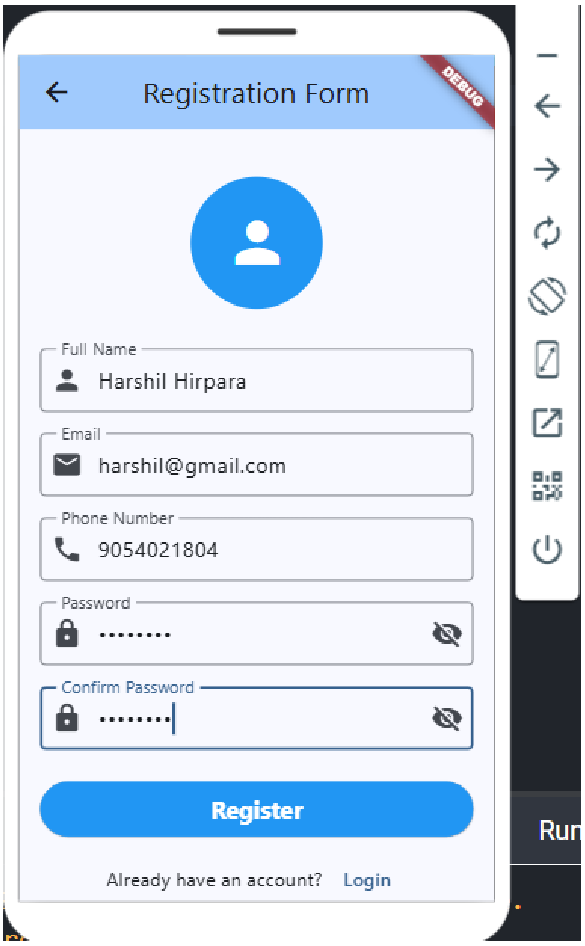
Code:-

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


void main() {
  runApp(const MyApp());
}


class MyApp extends StatelessWidget {
  const MyApp({super.key});


  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Registration App',
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.blue),
        useMaterial3: true,
      ),
      home: const RegistrationScreen(),
      routes: {
        '/dashboard': (context) => const DashboardScreen(),
        '/profile': (context) => const ProfileScreen(),
      },
    );
  }
}

class RegistrationScreen extends StatefulWidget {
  const RegistrationScreen({super.key});

  @override
  State<RegistrationScreen> createState() => _RegistrationScreenState();
}

class _RegistrationScreenState extends State<RegistrationScreen> {
  final _formKey = GlobalKey<FormState>();
  final _nameController = TextEditingController();
  final _emailController = TextEditingController();
  final _passwordController = TextEditingController();
  final _confirmPasswordController = TextEditingController();
  final _phoneController = TextEditingController();

  bool _isPasswordVisible = false;
  bool _isConfirmPasswordVisible = false;

  @override
  void dispose() {
    _nameController.dispose();
    _emailController.dispose();
    _passwordController.dispose();
    _confirmPasswordController.dispose();
    _phoneController.dispose();
    super.dispose();
```

```
}

void _submitForm() {
  if (_formKey.currentState!.validate()) {
    ScaffoldMessenger.of(context).showSnackBar(
      const SnackBar(
        content: Text('Registration Successful!'),
        backgroundColor: Colors.green,
      ),
    );

    Future.delayed(const Duration(seconds: 1), () {
      Navigator.pushReplacementNamed(context, '/dashboard');
    });
  }
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: const Text('Registration Form'),
      backgroundColor: Theme.of(context).colorScheme.inversePrimary,
      centerTitle: true,
      leading: IconButton(
        icon: const Icon(Icons.arrow_back),
        onPressed: () => Navigator.of(context, rootNavigator: true).pop(),
      ),
    ),
    body: SingleChildScrollView(
      padding: const EdgeInsets.all(16.0),
      child: Form(
        key: _formKey,
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.stretch,
          children: [
            const SizedBox(height: 20),

            const CircleAvatar(
              radius: 50,
              backgroundColor: Colors.blue,
              child: Icon(Icons.person, size: 50, color: Colors.white),
            ),

            const SizedBox(height: 30),

            TextFormField(
              controller: _nameController,
              decoration: const InputDecoration(
                labelText: 'Full Name',
                hintText: 'Enter your full name',
                prefixIcon: Icon(Icons.person),
                border: OutlineInputBorder(),
              ),
              validator: (value) {
                if (value == null || value.isEmpty) {
                  return 'Please enter your name';
                }
                if (value.length < 2) {
                  return 'Name must be at least 2 characters';
                }
                return null;
              },
            ),

            const SizedBox(height: 16),

            TextFormField(
              controller: _emailController,
              keyboardType: TextInputType.emailAddress,
              decoration: const InputDecoration(
                labelText: 'Email',
                hintText: 'Enter your email address',
                prefixIcon: Icon(Icons.email),
                border: OutlineInputBorder(),
              ),
              validator: (value) {
                if (value == null || value.isEmpty) {
                  return 'Please enter your email';
                }
                if (!RegExp(
                  r'^[\w-\.]+@([\w-]+\.)+[\w-]{2,4}$',
                ).hasMatch(value)) {
                  return 'Please enter a valid email address';
                }
                return null;
              },
            ),

            const SizedBox(height: 16),

            TextFormField(
             controller: _phoneController,
             keyboardType: TextInputType.phone,
             decoration: const InputDecoration(
              labelText: 'Phone Number',
              hintText: 'Enter your phone number',
              prefixIcon: Icon(Icons.phone),
              border: OutlineInputBorder(),
             ),
             validator: (value) {
              if (value == null || value.isEmpty) {
               return 'Please enter your phone number';
              }
              if (!RegExp(r'^\d{10}$').hasMatch(value)) {
               return 'Please enter a valid 10-digit phone number';
              }
              return null;
             },
            ),

            const SizedBox(height: 16),

            TextFormField(
             controller: _passwordController,
             obscureText: !_isPasswordVisible,
              decoration: InputDecoration(
                labelText: 'Password',
                hintText: 'Enter your password',
                prefixIcon: const Icon(Icons.lock),
                suffixIcon: IconButton(
                  icon: Icon(
                    _isPasswordVisible
                        ? Icons.visibility
                        : Icons.visibility_off,
                  ),
                  onPressed: () {
                    setState(() {
                      _isPasswordVisible = !_isPasswordVisible;
                    });
                  },
                ),
                border: const OutlineInputBorder(),
              ),
              validator: (value) {
                if (value == null || value.isEmpty) {
                  return 'Please enter your password';
                }
                if (value.length < 6) {
                  return 'Password must be at least 6 characters';
                }
                return null;
              },
            ),
            const SizedBox(height: 16),

            TextFormField(
              controller: _confirmPasswordController,
              obscureText: !_isConfirmPasswordVisible,
              decoration: InputDecoration(
                labelText: 'Confirm Password',
                hintText: 'Confirm your password',
                prefixIcon: const Icon(Icons.lock),
                suffixIcon: IconButton(
                  icon: Icon(
                    _isConfirmPasswordVisible
                        ? Icons.visibility
                        : Icons.visibility_off,
                  ),
                  onPressed: () {
                    setState(() {
                      _isConfirmPasswordVisible = !_isConfirmPasswordVisible;
                    });
                  },
                ),
                border: const OutlineInputBorder(),
              ),
              validator: (value) {
                if (value == null || value.isEmpty) {
                  return 'Please confirm your password';
                }
                if (value != _passwordController.text) {
                  return 'Passwords do not match';
                }
                return null;
              },
            ),

            const SizedBox(height: 24),

            ElevatedButton(
              onPressed: _submitForm,
              style: ElevatedButton.styleFrom(
                padding: const EdgeInsets.symmetric(vertical: 16),
                backgroundColor: Colors.blue,
                foregroundColor: Colors.white,
              ),
              child: const Text(
                'Register',
                style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),
              ),
            ),

            const SizedBox(height: 16),

            Row(
              mainAxisAlignment: MainAxisAlignment.center,
              children: [
                const Text('Already have an account? '),
                TextButton(
                  onPressed: () {
                    ScaffoldMessenger.of(context).showSnackBar(
                        const SnackBar(
                          content: Text('Login screen would open here'),
                          backgroundColor: Colors.orange,
                        ),
                      );
                    },
                    child: const Text('Login'),
                  ),
                ],
              ),
            ],
          ),
        ),
      ),
    );
  }
}

class DashboardScreen extends StatelessWidget {
  const DashboardScreen({super.key});

  Widget _buildDashboardCard(
    BuildContext context,
    String title,
    IconData icon,
    VoidCallback onTap,
  ) {
    return Card(
      elevation: 4,
    child: InkWell(
     onTap: onTap,
     borderRadius: BorderRadius.circular(12),
     child: Padding(
      padding: const EdgeInsets.all(20),
      child: Column(
       mainAxisAlignment: MainAxisAlignment.center,
       children: [
        Icon(icon, size: 48, color: Colors.blue),
        const SizedBox(height: 12),
        Text(
         title,
         style: const TextStyle(
          fontSize: 16,
          fontWeight: FontWeight.bold,
         ),
         textAlign: TextAlign.center,
        ),
       ],
      ),
     ),
    ),
   );
  }

  @override
  Widget build(BuildContext context) {
   return Scaffold(
    appBar: AppBar(
    title: const Text('Dashboard'),
    backgroundColor: Theme.of(context).colorScheme.inversePrimary,
    actions: [
      IconButton(
        icon: const Icon(Icons.person),
        onPressed: () => Navigator.pushNamed(context, '/profile'),
      ),
    ],
    leading: IconButton(
      icon: const Icon(Icons.arrow_back),
      onPressed: () => Navigator.of(context, rootNavigator: true).pop(),
    ),
  ),
  body: Padding(
    padding: const EdgeInsets.all(16.0),
    child: Column(
      crossAxisAlignment: CrossAxisAlignment.start,
      children: [
        const Text(
          'Welcome to Your Dashboard!',
          style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),
        ),
        const SizedBox(height: 8),
        Text(
          'Manage your account and explore features',
          style: TextStyle(fontSize: 16, color: Colors.grey[600]),
        ),
        const SizedBox(height: 32),
        Expanded(
      child: GridView.count(
        crossAxisCount: 2,
        crossAxisSpacing: 16,
        mainAxisSpacing: 16,
        children: [
          _buildDashboardCard(
            context,
            'Profile',
            Icons.person,
            () => Navigator.pushNamed(context, '/profile'),
          ),
          _buildDashboardCard(context, 'Settings', Icons.settings, () {
            ScaffoldMessenger.of(context).showSnackBar(
              const SnackBar(
                content: Text('Settings screen would open here'),
                backgroundColor: Colors.blue,
              ),
            );
          }),
          _buildDashboardCard(context, 'Messages', Icons.message, () {
            ScaffoldMessenger.of(context).showSnackBar(
              const SnackBar(
                content: Text('Messages screen would open here'),
                backgroundColor: Colors.green,
              ),
            );
          }),
          _buildDashboardCard(
            context,
            'Notifications',
            Icons.notifications,
            () {
              ScaffoldMessenger.of(context).showSnackBar(
                const SnackBar(
                  content: Text('Notifications screen would open here'),
                  backgroundColor: Colors.orange,
                ),
              );
            },
          ),
        ],
      ),
     ),
    ],
   ),
  ),
  );
  }
  }

  class ProfileScreen extends StatelessWidget {
    const ProfileScreen({super.key});

    @override
    Widget build(BuildContext context) {
      return Scaffold(
        appBar: AppBar(
          title: const Text('Profile'),
    backgroundColor: Theme.of(context).colorScheme.inversePrimary,
    leading: IconButton(
      icon: const Icon(Icons.arrow_back),
      onPressed: () => Navigator.pop(context),
    ),
  ),
  body: SingleChildScrollView(
    padding: const EdgeInsets.all(16.0),
    child: Column(
      children: [
        const CircleAvatar(
          radius: 60,
          backgroundColor: Colors.blue,
          child: Icon(Icons.person, size: 60, color: Colors.white),
        ),
        const SizedBox(height: 24),
        const Text(
          'User Profile',
          style: TextStyle(fontSize: 28, fontWeight: FontWeight.bold),
        ),
        const SizedBox(height: 32),
        Card(
          child: Padding(
            padding: const EdgeInsets.all(16.0),
            child: Column(
              children: [
                _buildProfileItem('Name', 'John Doe'),
                const Divider(),
                _buildProfileItem('Email', 'john.doe@example.com'),
                const Divider(),
                _buildProfileItem('Phone', '+1 234 567 8900'),
                const Divider(),
                _buildProfileItem('Location', 'New York, USA'),
              ],
            ),
          ),
        ),
        const SizedBox(height: 24),
        SizedBox(
          width: double.infinity,
          child: ElevatedButton(
            onPressed: () =>
                Navigator.pushReplacementNamed(context, '/dashboard'),
            style: ElevatedButton.styleFrom(
              padding: const EdgeInsets.symmetric(vertical: 16),
              backgroundColor: Colors.blue,
              foregroundColor: Colors.white,
            ),
            child: const Text(
              'Back to Dashboard',
              style: TextStyle(fontSize: 16, fontWeight: FontWeight.bold),
            ),
          ),
        ),
      ],
    ),
  ),
);
```

```
 }

 Widget _buildProfileItem(String label, String value) {
  return Padding(
   padding: const EdgeInsets.symmetric(vertical: 8.0),
   child: Row(
    mainAxisAlignment: MainAxisAlignment.spaceBetween,
    children: [
     Text(
      label,
      style: const TextStyle(
       fontSize: 16,
       fontWeight: FontWeight.bold,
       color: Colors.grey,
      ),
     ),
     Text(value, style: const TextStyle(fontSize: 16)),
    ],
   ),
  );
 }
}
```

<u>Output:-</u>

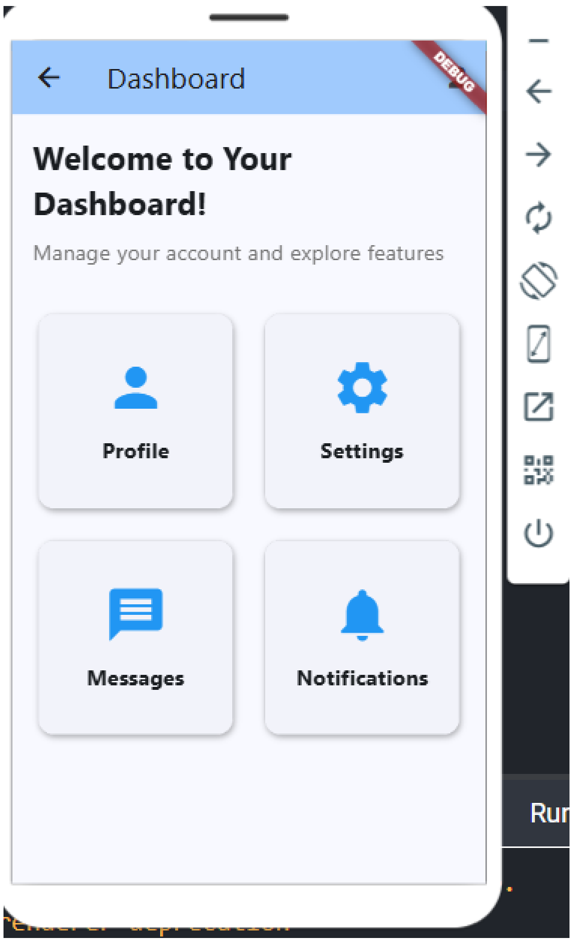


Registration Form

Full Name: Harshil Hirpara
Email: harshil@gmail.com
Phone Number: 9054021804
Password: ••••••••
Confirm Password: ••••••••

Register

Already have an account? Login



Dashboard

Welcome to Your Dashboard!

Manage your account and explore features

Practical – 5

Aim:- You are building a mobile application where users navigate through multiple screens like login, dashboard, and profile. Build a Student Records App with CRUD operations using SQLite.

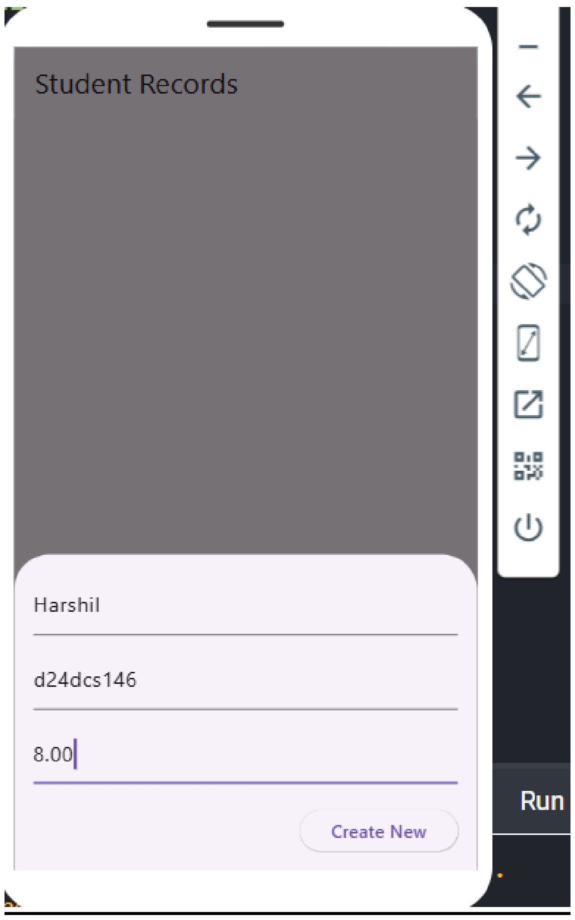
Code:-

```
import 'package:flutter/material.dart';
import 'package:localstorage/localstorage.dart';
void main() {
  runApp(const MaterialApp(
    debugShowCheckedModeBanner: false,
    home: Practical5(),
  ));
}
class Student {
  final String id; // using String as simple unique id
  final String name;
  final String rollNo;
  final String grade;

  Student({
    required this.id,
    required this.name,
    required this.rollNo,
    required this.grade,
  });

  Map<String, dynamic> toMap() {
    return {
      'id': id,
      'name': name,
      'rollNo': rollNo,
      'grade': grade,
    };
  }

  factory Student.fromMap(Map<String, dynamic> map) {
    return Student(
      id: map['id'],
      name: map['name'],
      rollNo: map['rollNo'],
      grade: map['grade'],
    );
  }
}

class Practical5 extends StatefulWidget {
  const Practical5({super.key});

  @override
  _Practical5State createState() => _Practical5State();
}

class _Practical5State extends State<Practical5> {
  final LocalStorage storage = LocalStorage('students_app');
  final _formKey = GlobalKey<FormState>();
  final _nameController = TextEditingController();
  final _rollNoController = TextEditingController();
  final _gradeController = TextEditingController();
  List<Student> students = [];

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

  Future<void> _loadStudents() async {
    await storage.ready;
    final data = storage.getItem('students');
    if (data != null) {
      setState(() {
        students = (data as List)
            .map((map) => Student.fromMap(Map<String, dynamic>.from(map)))
            .toList();
      });
    }
  }

  Future<void> _saveStudents() async {
    await storage.ready;
    storage.setItem('students', students.map((s) => s.toMap()).toList());
  }

  void _showForm(Student? student) {
    if (student != null) {
      _nameController.text = student.name;
      _rollNoController.text = student.rollNo;
      _gradeController.text = student.grade;
    } else {
      _nameController.clear();
      _rollNoController.clear();
      _gradeController.clear();
    }

    showModalBottomSheet(
      context: context,
      isScrollControlled: true,
      builder: (BuildContext context) => Padding(
        padding: EdgeInsets.only(
          top: 15,
          left: 15,
          right: 15,
          bottom: MediaQuery.of(context).viewInsets.bottom + 15,
        ),
        child: Form(
          key: _formKey,
          child: Column(
            mainAxisSize: MainAxisSize.min,
            crossAxisAlignment: CrossAxisAlignment.end,
            children: [
              TextFormField(
                controller: _nameController,
                decoration: const InputDecoration(hintText: 'Student Name'),
                validator: (value) =>
                    value == null || value.isEmpty ? 'Enter name' : null,
              ),
              const SizedBox(height: 10),
              TextFormField(
                controller: _rollNoController,
                decoration: const InputDecoration(hintText: 'Roll Number'),
                validator: (value) =>
                    value == null || value.isEmpty ? 'Enter roll no' : null,
              ),
              const SizedBox(height: 10),
              TextFormField(
                controller: _gradeController,
                decoration: const InputDecoration(hintText: 'Grade'),
                validator: (value) =>
                    value == null || value.isEmpty ? 'Enter grade' : null,
              ),
              const SizedBox(height: 20),
              ElevatedButton(
                onPressed: () {
                  if (_formKey.currentState!.validate()) {
                    if (student == null) {
                      // create new
                      setState(() {
                        students.add(Student(
                          id: DateTime.now().millisecondsSinceEpoch.toString(),
                          name: _nameController.text,
                          rollNo: _rollNoController.text,
                          grade: _gradeController.text,
                        ));
                      });
```

```
          } else {
           // update existing
           setState(() {
            final index = students.indexWhere((s) => s.id == student.id);
            students[index] = Student(
             id: student.id,
             name: _nameController.text,
             rollNo: _rollNoController.text,
             grade: _gradeController.text,
            );
           });
          }
          _saveStudents();
          Navigator.pop(context);
         }
        },
        child: Text(student == null ? 'Create New' : 'Update'),
       ),
      ],
     ),
    ),
   ),
  );
 }

 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(title: const Text('Student Records')),
body: ListView.builder(
  itemCount: students.length,
  itemBuilder: (context, index) {
    final student = students[index];
    return Card(
      margin: const EdgeInsets.all(12),
      child: ListTile(
        title: Text(student.name),
        subtitle: Text('Roll No: ${student.rollNo}\nGrade: ${student.grade}'),
        trailing: Row(
          mainAxisSize: MainAxisSize.min,
          children: [
            IconButton(
              icon: const Icon(Icons.edit),
              onPressed: () => _showForm(student),
            ),
            IconButton(
              icon: const Icon(Icons.delete, color: Colors.red),
              onPressed: () {
                setState(() {
                  students.removeAt(index);
                });
                _saveStudents();
              },
            ),
          ],
        ),
      ),
    );
```

```
      },
    ),
    floatingActionButton: FloatingActionButton(
      child: const Icon(Icons.add),
      onPressed: () => _showForm(null),
    ),
  );
 }
}
```

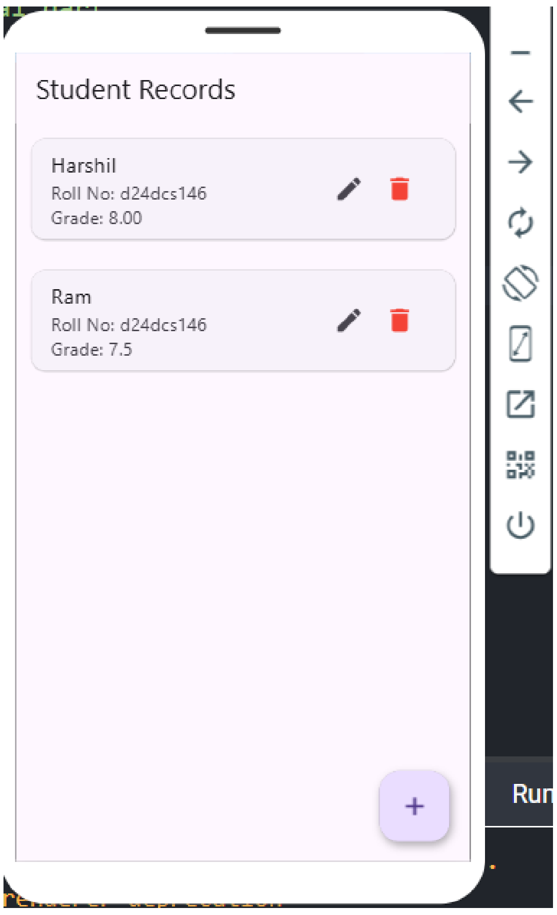
Output :


Student Records

Harshil

d24dcs146

8.00


Student Records

Harshil
Roll No: d24dcs146
Grade: 8.00

Ram
Roll No: d24dcs146
Grade: 7.5

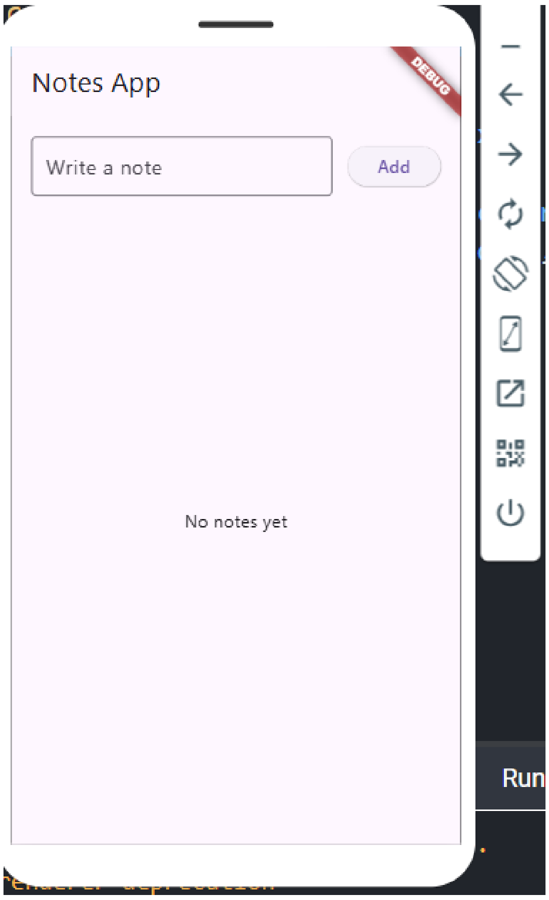
Practical – 6

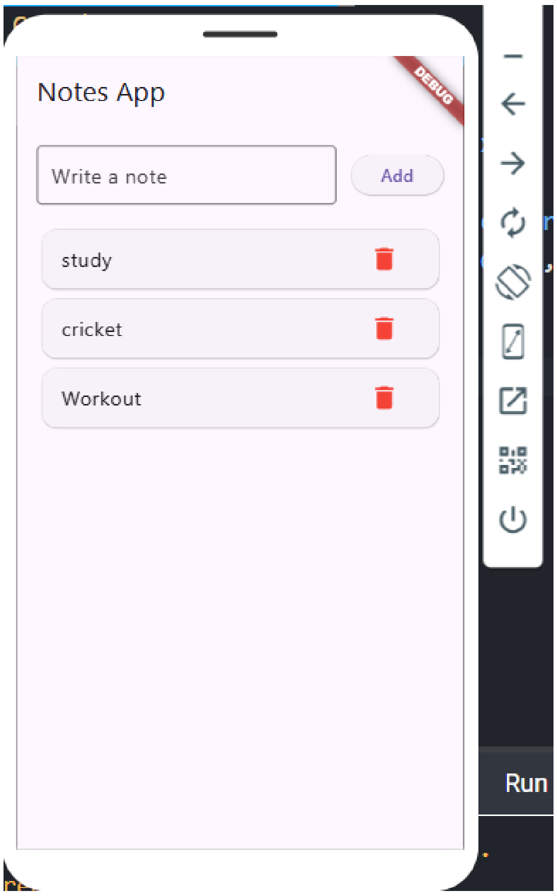
Aim:- You are building a mobile application where users navigate through multiple screens like login, dashboard, and profile. Create a Notes App with persistent storage using Shared Preferences.

Code:-

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

import 'package:localstorage/localstorage.dart';


class NotesApp extends StatefulWidget {

 @override

 _NotesAppState createState() => _NotesAppState();

}


class _NotesAppState extends State<NotesApp> {

 final LocalStorage storage = LocalStorage('notes_app');

 final TextEditingController _controller = TextEditingController();

 List<String> _notes = [];


 @override

 void initState() {

  super.initState();

  _loadNotes();

 }


 Future<void> _loadNotes() async {

  await storage.ready;

  final saved = storage.getItem('notes');

  if (saved != null) {
    setState(() {
      _notes = List<String>.from(saved);
    });
  }
}

Future<void> _saveNotes() async {
  await storage.ready;
  storage.setItem('notes', _notes);
}

void _addNote() {
  final text = _controller.text.trim();
  if (text.isEmpty) return;
  setState(() {
    _notes.add(text);
    _controller.clear();
  });
  _saveNotes();
}

void _deleteNote(int index) {
  setState(() {
    _notes.removeAt(index);
  });
  _saveNotes();
}

void _editNoteDialog(int index) {
  final editController = TextEditingController(text: _notes[index]);
  showDialog(
    context: context,
    builder: (_) => AlertDialog(
      title: const Text('Edit Note'),
      content: TextField(controller: editController),
      actions: [
        TextButton(onPressed: () => Navigator.pop(context), child: const Text('Cancel')),
        ElevatedButton(
          onPressed: () {
            final updated = editController.text.trim();
            if (updated.isNotEmpty) {
              setState(() {
                _notes[index] = updated;
              });
              _saveNotes();
            }
            Navigator.pop(context);
          },
          child: const Text('Save'),
        ),
      ],
    ),
  );
}

@override
Widget build(BuildContext context) {
  return MaterialApp(
    home: Scaffold(
      appBar: AppBar(title: const Text("Notes App")),
      body: Padding(
        padding: const EdgeInsets.all(16),
        child: Column(
          children: [
            Row(
              children: [
                Expanded(
                  child: TextField(
                    controller: _controller,
                    decoration: const InputDecoration(
                      labelText: "Write a note",
                      border: OutlineInputBorder(),
                    ),
                  ),
                ),
                const SizedBox(width: 8),
                ElevatedButton(onPressed: _addNote, child: const Text("Add")),
              ],
            ),
            const SizedBox(height: 16),
            Expanded(
              child: _notes.isEmpty
                  ? const Center(child: Text("No notes yet"))
                  : ListView.builder(
                      itemCount: _notes.length,
                      itemBuilder: (context, index) {
                        return Card(
              child: ListTile(
               title: Text(_notes[index]),
               onTap: () => _editNoteDialog(index),
               trailing: IconButton(
                icon: const Icon(Icons.delete, color: Colors.red),
                onPressed: () => _deleteNote(index),
               ),
              ),
             );
            },
           ),
         ),
        ],
       ),
      ),
     ),
    );
  }
}
```

<u>Output:-</u>


Notes App
DEBUG
Write a note
Add
No notes yet
Run


Notes App
DEBUG
Write a note
Add
study
cricket
Workout
Run

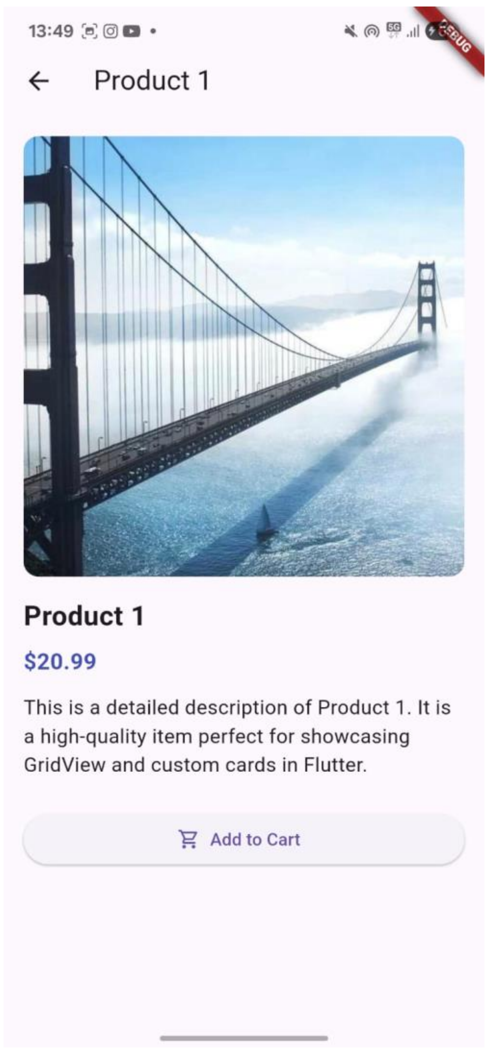
Practical – 7

Aim:- You are building a mobile application where users navigate through multiple screens like login, dashboard, and profile. Design a Product Catalog App using GridView and custom cards with images.

Code:-

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


class ProductCatalogApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Product Catalog',
      theme: ThemeData(primarySwatch: Colors.indigo),
      home: ProductGridPage(),
    );
  }
}


class Product {
  final String id;
  final String name;
  final String imageUrl;
  final double price;
  final String description;


  Product({
    required this.id,
    required this.name,
    required this.imageUrl,
    required this.price,
    required this.description,
  });
}

final List<Product> _sampleProducts = List.generate(12, (index) {
  final int num = index + 1;
  return Product(
    id: 'p$num',
    name: 'Product $num',
    imageUrl: 'https://picsum.photos/seed/product$num/600/600',
    price: 19.99 + num,
    description:
        'This is a detailed description of Product $num. It is a high-quality item perfect for
showcasing GridView and custom cards in Flutter.',
  );
});

class ProductGridPage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('Product Catalog'),
        leading: IconButton(
          icon: const Icon(Icons.arrow_back),
          onPressed: () => Navigator.of(context, rootNavigator: true).pop(),
        ),
      ),
      body: Padding(
        padding: const EdgeInsets.all(12.0),
        child: GridView.builder(
          gridDelegate: const SliverGridDelegateWithFixedCrossAxisCount(
            crossAxisCount: 2,
            mainAxisSpacing: 12,
            crossAxisSpacing: 12,
            childAspectRatio: 0.72,
          ),
          itemCount: _sampleProducts.length,
          itemBuilder: (context, index) {
            final product = _sampleProducts[index];
            return _ProductCard(product: product);
          },
        ),
      ),
    );
  }
}

class _ProductCard extends StatelessWidget {
  final Product product;

  const _ProductCard({required this.product});

  @override
  Widget build(BuildContext context) {
    return InkWell(
      onTap: () => Navigator.push(
     context,
     MaterialPageRoute(builder: (_) => ProductDetailPage(product: product)),
    ),
    child: Card(
     elevation: 4,
     shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(12)),
     clipBehavior: Clip.antiAlias,
     child: Column(
      crossAxisAlignment: CrossAxisAlignment.stretch,
      children: [
       Expanded(
        child: Hero(
         tag: product.id,
         child: Image.network(
          product.imageUrl,
          fit: BoxFit.cover,
          errorBuilder: (context, error, stackTrace) => Container(
           color: Colors.grey.shade200,
           alignment: Alignment.center,
           child: const Icon(Icons.image_not_supported, size: 40),
          ),
         ),
        ),
       ),
       Padding(
        padding: const EdgeInsets.all(10.0),
        child: Column(
         crossAxisAlignment: CrossAxisAlignment.start,
         children: [
          Text(
           product.name,
           maxLines: 1,
           overflow: TextOverflow.ellipsis,
           style: const TextStyle(
             fontSize: 16,
             fontWeight: FontWeight.w600,
           ),
          ),
          const SizedBox(height: 4),
          Text(
           '\$${product.price.toStringAsFixed(2)}',
           style: const TextStyle(
             fontSize: 14,
             color: Colors.indigo,
             fontWeight: FontWeight.bold,
           ),
          ),
          const SizedBox(height: 8),
          SizedBox(
           width: double.infinity,
           child: OutlinedButton.icon(
             icon: const Icon(Icons.info_outline),
             label: const Text('Details'),
             onPressed: () => Navigator.push(
               context,
               MaterialPageRoute(
                 builder: (_) => ProductDetailPage(product: product),
               ),
                    ),
                  ),
                ),
              ],
            ),
          ),
        ],
      ),
    ),
  );
 }
}
class ProductDetailPage extends StatelessWidget {
 final Product product;
 const ProductDetailPage({required this.product});
 @override
 Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(title: Text(product.name)),
    body: SingleChildScrollView(
      padding: const EdgeInsets.all(16.0),
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          Hero(
            tag: product.id,
            child: AspectRatio(
              aspectRatio: 1,
              child: ClipRRect(
                borderRadius: BorderRadius.circular(12),
                child: Image.network(product.imageUrl, fit: BoxFit.cover),
              ),
            ),
          ),
          const SizedBox(height: 16),
          Text(
            product.name,
            style: const TextStyle(fontSize: 22, fontWeight: FontWeight.bold),
          ),
          const SizedBox(height: 8),
          Text(
            '\$${product.price.toStringAsFixed(2)}',
            style: const TextStyle(
              fontSize: 18,
              color: Colors.indigo,
              fontWeight: FontWeight.bold,
            ),
          ),
          const SizedBox(height: 12),
          Text(product.description, style: const TextStyle(fontSize: 16)),
          const SizedBox(height: 24),
          SizedBox(
            width: double.infinity,
            child: ElevatedButton.icon(
              icon: const Icon(Icons.shopping_cart_outlined),
              label: const Text('Add to Cart'),
              onPressed: () {
                ScaffoldMessenger.of(context).showSnackBar(
                    const SnackBar(content: Text('Added to cart')),
                  );
                },
              ),
            ),
          ],
        ),
      ),
    );
  }
}
```

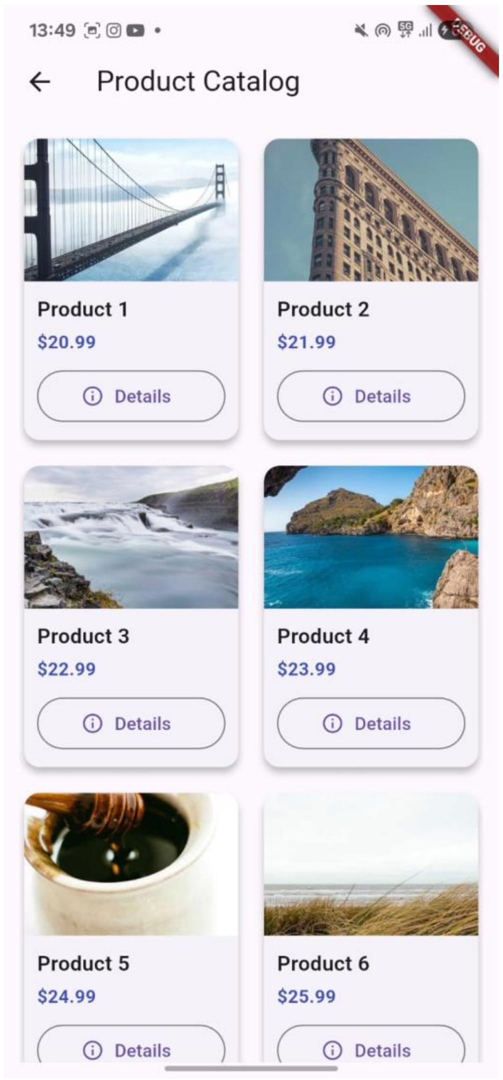
Output:-



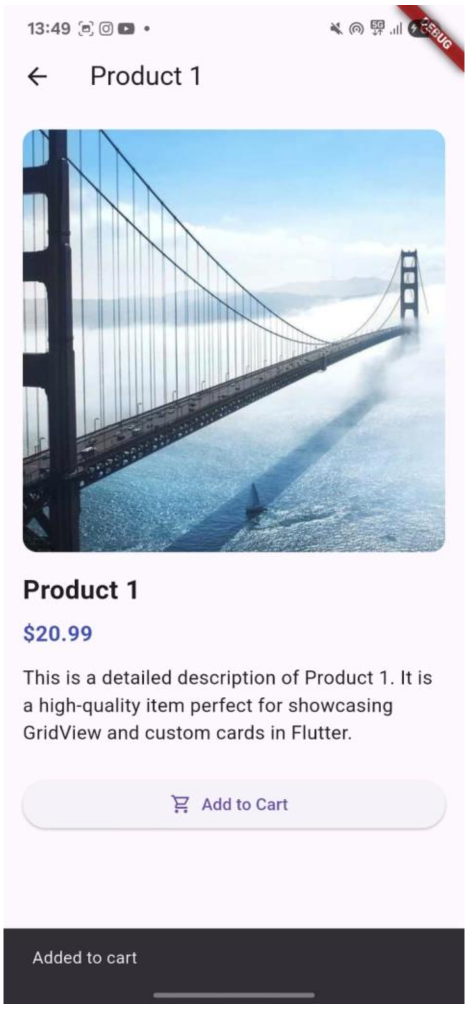
Product 1

$20.99

This is a detailed description of Product 1. It is a high-quality item perfect for showcasing GridView and custom cards in Flutter.



Product 1 $20.99
Product 2 $21.99
Product 3 $22.99
Product 4 $23.99
Product 5 $24.99
Product 6 $25.99



Product 1

$20.99

This is a detailed description of Product 1. It is a high-quality item perfect for showcasing GridView and custom cards in Flutter.

Added to cart

Practical – 8

Aim:- You are building a mobile application where users navigate through multiple screens like login, dashboard, and profile. Connect to a REST API (e.g., weather, user data) and display using FutureBuilder.

Code:-

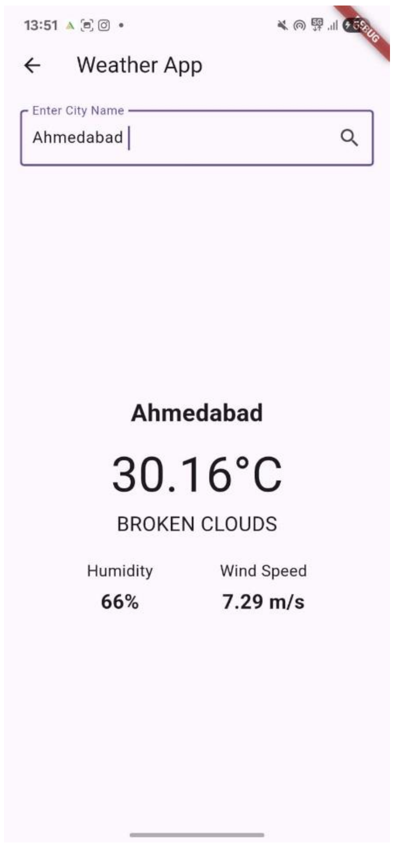
```
import 'package:flutter/material.dart';

import 'package:http/http.dart' as http;

import 'dart:convert';


class WeatherApp extends StatefulWidget {

  const WeatherApp({super.key});


  @override

  _WeatherAppState createState() => _WeatherAppState();

}


class _WeatherAppState extends State<WeatherApp> {

  final TextEditingController _cityController = TextEditingController();

  Future<Map<String, dynamic>>? _weatherData;


  Future<Map<String, dynamic>> fetchWeatherData(String city) async {

    if (city.isEmpty) {

      throw Exception('Please enter a city name');

    }


    const apiKey = '10ea2c0f059fa51f8efc518f80ddb9ce';

    try {

      final response = await http
```

```
     .get(
      Uri.parse(
       'https://api.openweathermap.org/data/2.5/weather?q=$city&appid=$apiKey&units
=metric',
      ),
     )
     .timeout(
      const Duration(seconds: 10),
      onTimeout: () {
       throw Exception(
        'Connection timed out. Please check your internet connection and try again.',
       );
      },
     );

   if (response.statusCode == 200) {
    final data = json.decode(response.body);
    return data;
   } else if (response.statusCode == 404) {
    throw Exception('City not found. Please check the city name.');
   } else if (response.statusCode == 401) {
    throw Exception(
     'API key is invalid or has expired. Please check your API key.',
    );
   } else if (response.statusCode == 429) {
    throw Exception(
     'Too many requests. Please try again in a few minutes.',
    );
   } else {
    final errorBody = json.decode(response.body);
```

```
        throw Exception(
          errorBody['message'] ??
              'Server error (${response.statusCode}). Please try again later.',
        );
      }
    } on FormatException {
      throw Exception('Invalid response from server. Please try again.');
    } catch (e) {
      if (e.toString().contains('SocketException')) {
        throw Exception(
          'No internet connection. Please check your network settings.',
        );
      }
      throw Exception(
        'Failed to connect to weather service. Please try again.',
      );
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: const Text('Weather App')),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          children: [
            TextField(
              controller: _cityController,
  decoration: InputDecoration(
   labelText: 'Enter City Name',
   border: OutlineInputBorder(),
   suffixIcon: IconButton(
    icon: const Icon(Icons.search),
    onPressed: () {
     if (_cityController.text.trim().isEmpty) {
      ScaffoldMessenger.of(context).showSnackBar(
       const SnackBar(
        content: Text('Please enter a city name'),
        backgroundColor: Colors.red,
       ),
      );
      return;
     }
     setState(() {
      _weatherData = fetchWeatherData(
       _cityController.text.trim(),
      );
     });
    },
   ),
  ),
 ),
 const SizedBox(height: 20),
 Expanded(
  child: FutureBuilder<Map<String, dynamic>>(
   future: _weatherData,
   builder: (context, snapshot) {
```

```
if (snapshot.connectionState == ConnectionState.waiting) {
  return const Center(
    child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        CircularProgressIndicator(),
        SizedBox(height: 16),
        Text('Fetching weather data...'),
      ],
    ),
  );
} else if (snapshot.hasError) {
  return Center(
    child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        Icon(
          Icons.error_outline,
          size: 48,
          color: Colors.red,
        ),
        SizedBox(height: 16),
        Text(
          'Error: ${snapshot.error}',
          style: TextStyle(color: Colors.red),
          textAlign: TextAlign.center,
        ),
        SizedBox(height: 8),
        Text(
          'Please check the city name and try again.',
          textAlign: TextAlign.center,
        ),
      ],
    ),
  );
} else if (!snapshot.hasData) {
  return const Center(
    child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        Icon(Icons.search, size: 48, color: Colors.blue),
        SizedBox(height: 16),
        Text(
          'Enter a city name to get weather information',
          textAlign: TextAlign.center,
        ),
      ],
    ),
  );
}

final weather = snapshot.data!;
return Center(
  child: Column(
    mainAxisAlignment: MainAxisAlignment.center,
    children: [
      Text(
        weather['name'],
 style: const TextStyle(
   fontSize: 24,
   fontWeight: FontWeight.bold,
 ),
),
const SizedBox(height: 10),
Text(
 '${weather['main']['temp']}°C',
 style: const TextStyle(fontSize: 48),
),
Text(
 weather['weather'][0]['description']
     .toString()
     .toUpperCase(),
 style: const TextStyle(fontSize: 20),
),
const SizedBox(height: 20),
Row(
 mainAxisAlignment: MainAxisAlignment.spaceEvenly,
 children: [
   WeatherInfo(
     title: 'Humidity',
     value: '${weather['main']['humidity']}%',
   ),
   WeatherInfo(
     title: 'Wind Speed',
     value: '${weather['wind']['speed']} m/s',
   ),
 ],
                    ),
                  ],
                ),
              );
            },
          ),
        ),
      ],
    ),
  ),
 );
 }
}

class WeatherInfo extends StatelessWidget {
 final String title;
 final String value;

 const WeatherInfo({super.key, required this.title, required this.value});

 @override
 Widget build(BuildContext context) {
  return Column(
    children: [
     Text(title, style: const TextStyle(fontSize: 16)),
     const SizedBox(height: 5),
     Text(
      value,
      style: const TextStyle(fontSize: 20, fontWeight: FontWeight.bold),
    ),
  ], }; } }
```

Output:-



13:51

Weather App

Enter City Name

Ahmedabad

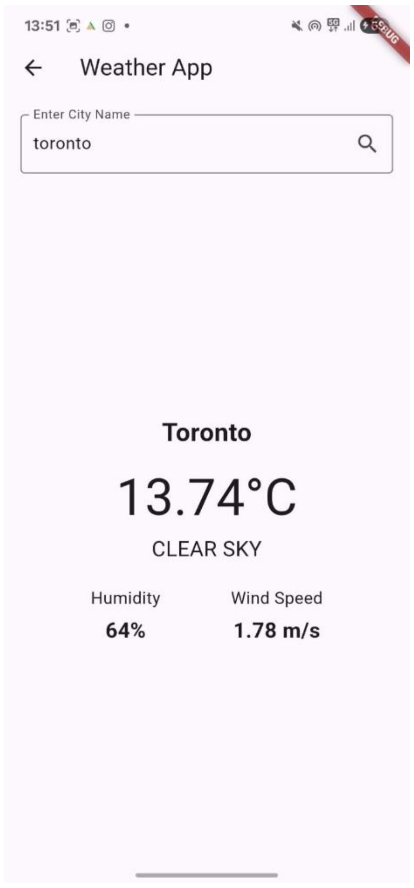
Ahmedabad

30.16°C

BROKEN CLOUDS

Humidity
66%

Wind Speed
7.29 m/s



13:51

Weather App

Enter City Name

toronto

Toronto

13.74°C

CLEAR SKY

Humidity
64%

Wind Speed
1.78 m/s

Practical – 9

Aim:- You are building a mobile application where users navigate through multiple screens like login, dashboard, and profile. Develop a Login Authentication App using API-based credential check and session handling.

Code:-

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

import 'package:shared_preferences/shared_preferences.dart';

import 'dart:convert';


class LoginApp extends StatelessWidget {

 const LoginApp({super.key});


 @override

 Widget build(BuildContext context) {

  return MaterialApp(

   title: 'Login Authentication Demo',

   theme: ThemeData(

    primarySwatch: Colors.blue,

    appBarTheme: const AppBarTheme(

     backgroundColor: Colors.blue,

     foregroundColor: Colors.white,

     elevation: 2,

    ),

   ),

   home: const AuthWrapper(),

   debugShowCheckedModeBanner: false,

  );

 }

}
```

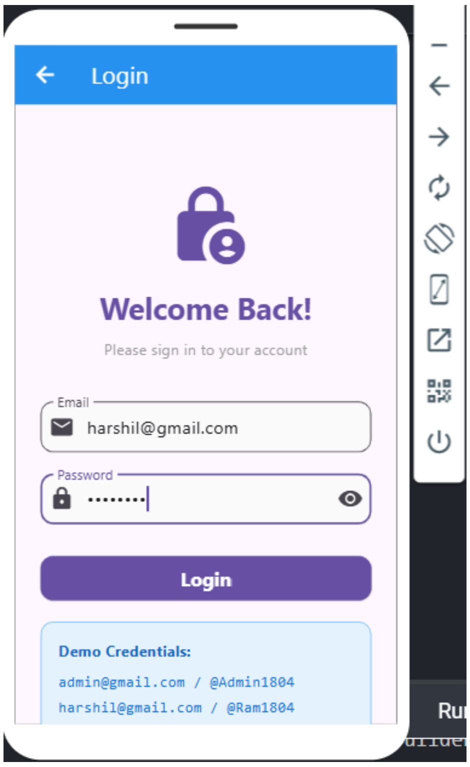
```
class AuthWrapper extends StatefulWidget {
  const AuthWrapper({super.key});

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

class _AuthWrapperState extends State<AuthWrapper> {
  bool isLoggedIn = false;
  bool isLoading = true;

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

  Future<void> _checkLoginStatus() async {
    final prefs = await SharedPreferences.getInstance();
    final sessionData = prefs.getString('user_session');

    if (sessionData != null) {
      final session = json.decode(sessionData);
      final loginTime = DateTime.parse(session['loginTime']);
      final now = DateTime.now();

      // Session expires after 1 hour
      if (now.difference(loginTime).inHours < 1) {
        setState(() {
          isLoggedIn = true;
          isLoading = false;
        });
        return;
      } else {
        // Clear expired session
        await prefs.remove('user_session');
      }
    }

    setState(() {
      isLoggedIn = false;
      isLoading = false;
    });
  }

  @override
  Widget build(BuildContext context) {
    if (isLoading) {
      return const Scaffold(body: Center(child: CircularProgressIndicator()));
    }

    return isLoggedIn ? const DashboardScreen() : const LoginScreen();
  }
}

class LoginScreen extends StatefulWidget {
  const LoginScreen({super.key});
  @override
  _LoginScreenState createState() => _LoginScreenState();
}

class _LoginScreenState extends State<LoginScreen> {
  final _formKey = GlobalKey<FormState>();
  final _emailController = TextEditingController();
  final _passwordController = TextEditingController();
  bool _isLoading = false;
  bool _obscurePassword = true;

  // Mock users database
  final List<Map<String, String>> _mockUsers = [
    {
      'email': 'admin@gmail.com',
      'password': '@Admin1804',
      'name': 'Admin User',
      'role': 'Administrator',
    },
    {
      'email': 'harshil@gmail.com',
      'password': '@Ram1804',
      'name': 'Regular User',
      'role': 'User',
    },
    {
      'email': 'demo@gmail.com',
      'password': '@Demo1804',
      'name': 'Demo User',
      'role': 'Demo',
    },
  ];

  Future<Map<String, dynamic>?> _authenticateUser(
    String email,
    String password,
  ) async {
    // Simulate API call delay
    await Future.delayed(const Duration(seconds: 2));

    // Check credentials against mock database
    for (var user in _mockUsers) {
      if (user['email'] == email && user['password'] == password) {
        return {
          'success': true,
          'user': {
            'email': user['email'],
            'name': user['name'],
            'role': user['role'],
            'id': DateTime.now().millisecondsSinceEpoch.toString(),
          },
        };
      }
    }

    return {'success': false, 'message': 'Invalid email or password'};
  }
```

```
Future<void> _login() async {
  if (_formKey.currentState!.validate()) {
    setState(() => _isLoading = true);

    try {
      final result = await _authenticateUser(
        _emailController.text.trim(),
        _passwordController.text,
      );

      if (result != null && result['success']) {
        // Save user session
        final prefs = await SharedPreferences.getInstance();
        final sessionData = {
          'user': result['user'],
          'loginTime': DateTime.now().toIso8601String(),
          'isAuthenticated': true,
        };

        await prefs.setString('user_session', json.encode(sessionData));

        if (mounted) {
          Navigator.of(context).pushReplacement(
            MaterialPageRoute(builder: (context) => const DashboardScreen()),
          );
        }
      } else {
        if (mounted) {
          ScaffoldMessenger.of(context).showSnackBar(
            SnackBar(
              content: Text(result?['message'] ?? 'Login failed'),
              backgroundColor: Colors.red,
            ),
          );
        }
      }
    } catch (e) {
      if (mounted) {
        ScaffoldMessenger.of(context).showSnackBar(
          SnackBar(
            content: Text('Network error: ${e.toString()}'),
            backgroundColor: Colors.red,
          ),
        );
      }
    }

    setState(() => _isLoading = false);
  }
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: const Text('Login'),
      leading: IconButton(
        icon: const Icon(Icons.arrow_back),
        onPressed: () => Navigator.of(context).pop(),
      ),
    ),
    body: SingleChildScrollView(
      padding: const EdgeInsets.all(24.0),
      child: Form(
        key: _formKey,
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          crossAxisAlignment: CrossAxisAlignment.stretch,
          children: [
            const SizedBox(height: 50),

            // Logo or Title
            Icon(
              Icons.lock_person,
              size: 80,
              color: Theme.of(context).primaryColor,
            ),
            const SizedBox(height: 20),

            Text(
              'Welcome Back!',
              textAlign: TextAlign.center,
              style: Theme.of(context).textTheme.headlineMedium?.copyWith(
                fontWeight: FontWeight.bold,
                color: Theme.of(context).primaryColor,
              ),
            ),
            const SizedBox(height: 10),

            const Text(
              'Please sign in to your account',
              textAlign: TextAlign.center,
              style: TextStyle(color: Colors.grey),
            ),
            const SizedBox(height: 40),

            // Email field
            TextFormField(
              controller: _emailController,
              keyboardType: TextInputType.emailAddress,
              decoration: InputDecoration(
                labelText: 'Email',
                prefixIcon: const Icon(Icons.email),
                border: OutlineInputBorder(
                  borderRadius: BorderRadius.circular(12),
                ),
                filled: true,
                fillColor: Colors.grey[50],
              ),
              validator: (value) {
                if (value == null || value.isEmpty) {
                  return 'Please enter your email';
                }
                if (!RegExp(
                  r'^[\w-\.]+@([\w-]+\.)+[\w-]{2,4}$',
                ).hasMatch(value)) {
                  return 'Please enter a valid email';
                }
                return null;
              },
            ),
            const SizedBox(height: 20),

            // Password field
            TextFormField(
              controller: _passwordController,
              obscureText: _obscurePassword,
              decoration: InputDecoration(
                labelText: 'Password',
                prefixIcon: const Icon(Icons.lock),
                suffixIcon: IconButton(
                  icon: Icon(
                    _obscurePassword
                        ? Icons.visibility
                        : Icons.visibility_off,
                  ),
                  onPressed: () {
                    setState(() {
                      _obscurePassword = !_obscurePassword;
                    });
                  },
                ),
                border: OutlineInputBorder(
                  borderRadius: BorderRadius.circular(12),
                ),
                filled: true,
                fillColor: Colors.grey[50],
              ),
              validator: (value) {
                if (value == null || value.isEmpty) {
                  return 'Please enter your password';
                }
                if (value.length < 6) {
                  return 'Password must be at least 6 characters';
                }
                return null;
              },
            ),
            const SizedBox(height: 30),

            // Login button
            ElevatedButton(
              onPressed: _isLoading ? null : _login,
              style: ElevatedButton.styleFrom(
                padding: const EdgeInsets.symmetric(vertical: 16),
                shape: RoundedRectangleBorder(
                  borderRadius: BorderRadius.circular(12),
                ),
                backgroundColor: Theme.of(context).primaryColor,
                foregroundColor: Colors.white,
              ),
              child: _isLoading
                  ? const SizedBox(
                    height: 20,
                    width: 20,
                    child: CircularProgressIndicator(
                      strokeWidth: 2,
                      valueColor: AlwaysStoppedAnimation<Color>(
                        Colors.white,
                      ),
                    ),
                  )
                : const Text(
                    'Login',
                    style: TextStyle(
                      fontSize: 18,
                      fontWeight: FontWeight.bold,
                    ),
                  ),
          ),
          const SizedBox(height: 20),

          // Demo credentials info
          Container(
            padding: const EdgeInsets.all(16),
            decoration: BoxDecoration(
              color: Colors.blue[50],
              borderRadius: BorderRadius.circular(12),
              border: Border.all(color: Colors.blue[200]!),
            ),
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.start,
                 children: [
                  Text(
                   'Demo Credentials:',
                   style: TextStyle(
                    fontWeight: FontWeight.bold,
                    color: Colors.blue[800],
                   ),
                  ),
                  const SizedBox(height: 8),
                  ..._mockUsers.map(
                   (user) => Padding(
                    padding: const EdgeInsets.symmetric(vertical: 2),
                    child: Text(
                     '${user['email']} / ${user['password']}',
                     style: TextStyle(
                      fontFamily: 'monospace',
                      color: Colors.blue[700],
                     ),
                    ),
                   ),
                  ),
                 ],
                ),
               ),
              ],
             ),
            ),
           ),
          );
          }

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

class DashboardScreen extends StatefulWidget {
  const DashboardScreen({super.key});

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

class _DashboardScreenState extends State<DashboardScreen> {
  Map<String, dynamic>? _userData;
  bool _isLoading = true;

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

  Future<void> _loadUserData() async {
    final prefs = await SharedPreferences.getInstance();
          final sessionData = prefs.getString('user_session');

          if (sessionData != null) {
            final session = json.decode(sessionData);
            setState(() {
              _userData = session['user'];
              _isLoading = false;
            });
          } else {
            // If no session data, redirect to login
            Navigator.of(context).pushReplacement(
              MaterialPageRoute(builder: (context) => const LoginScreen()),
            );
          }
}

Future<void> _logout() async {
  showDialog(
    context: context,
    builder: (context) => AlertDialog(
      title: const Text('Logout'),
      content: const Text('Are you sure you want to logout?'),
      actions: [
        TextButton(
          onPressed: () => Navigator.of(context).pop(),
          child: const Text('Cancel'),
        ),
        ElevatedButton(
          onPressed: () async {
```

```
          final prefs = await SharedPreferences.getInstance();
          await prefs.remove('user_session');

          if (mounted) {
            Navigator.of(context).pop(); // Close dialog
            Navigator.of(context).pushReplacement(
              MaterialPageRoute(builder: (context) => const LoginScreen()),
            );
          }
        },
        style: ElevatedButton.styleFrom(backgroundColor: Colors.red),
        child: const Text('Logout', style: TextStyle(color: Colors.white)),
      ),
    ],
  ),
 );
}

Future<void> _navigateToProfile() async {
 Navigator.of(context).push(
  MaterialPageRoute(
   builder: (context) => ProfileScreen(userData: _userData!),
  ),
 );
}

@override
Widget build(BuildContext context) {
 if (_isLoading) {
   return const Scaffold(body: Center(child: CircularProgressIndicator()));
  }

  return Scaffold(
   appBar: AppBar(
    title: const Text('Dashboard'),
    actions: [
     IconButton(
      icon: const Icon(Icons.person),
      onPressed: _navigateToProfile,
     ),
     IconButton(icon: const Icon(Icons.logout), onPressed: _logout),
    ],
    leading: IconButton(
     icon: const Icon(Icons.arrow_back),
     onPressed: () => Navigator.of(context).pop(),
    ),
   ),
   body: SingleChildScrollView(
    padding: const EdgeInsets.all(20.0),
    child: Column(
     crossAxisAlignment: CrossAxisAlignment.start,
     children: [
      // Welcome card
      Container(
       width: double.infinity,
       padding: const EdgeInsets.all(20),
       decoration: BoxDecoration(
        gradient: LinearGradient(
      colors: [Colors.blue[400]!, Colors.blue[600]!],
      begin: Alignment.topLeft,
      end: Alignment.bottomRight,
    ),
    borderRadius: BorderRadius.circular(16),
  ),
  child: Column(
    crossAxisAlignment: CrossAxisAlignment.start,
    children: [
      const Text(
        'Welcome back!',
        style: TextStyle(color: Colors.white, fontSize: 18),
      ),
      const SizedBox(height: 8),
      Text(
        _userData?['name'] ?? 'User',
        style: const TextStyle(
          color: Colors.white,
          fontSize: 28,
          fontWeight: FontWeight.bold,
        ),
      ),
      const SizedBox(height: 4),
      Text(
        _userData?['role'] ?? 'User',
        style: const TextStyle(color: Colors.white70, fontSize: 16),
      ),
    ],
  ),
),
const SizedBox(height: 30),

// Quick actions
const Text(
  'Quick Actions',
  style: TextStyle(fontSize: 20, fontWeight: FontWeight.bold),
),
const SizedBox(height: 16),

GridView.count(
  shrinkWrap: true,
  physics: const NeverScrollableScrollPhysics(),
  crossAxisCount: 2,
  crossAxisSpacing: 16,
  mainAxisSpacing: 16,
  children: [
    _buildActionCard(
      'Profile',
      Icons.person,
      Colors.blue,
      _navigateToProfile,
    ),
    _buildActionCard('Settings', Icons.settings, Colors.green, () {
      ScaffoldMessenger.of(context).showSnackBar(
        const SnackBar(
          content: Text('Settings feature coming soon!'),
        ),
      );
```

```
        }),
        _buildActionCard(
          'Notifications',
          Icons.notifications,
          Colors.orange,
          () {
            ScaffoldMessenger.of(context).showSnackBar(
              const SnackBar(content: Text('No new notifications')),
            );
          },
        ),
        _buildActionCard('Help', Icons.help, Colors.purple, () {
          showDialog(
            context: context,
            builder: (context) => AlertDialog(
              title: const Text('Help'),
              content: const Text(
                'This is a demo login app with session management.',
              ),
              actions: [
                TextButton(
                  onPressed: () => Navigator.pop(context),
                  child: const Text('OK'),
                ),
              ],
            ),
          );
        }),
      ],
        ),
      ],
    ),
  ),
 );
}

Widget _buildActionCard(
 String title,
 IconData icon,
 Color color,
 VoidCallback onTap,
) {
 return Card(
  elevation: 4,
  shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(12)),
  child: InkWell(
   onTap: onTap,
   borderRadius: BorderRadius.circular(12),
   child: Container(
    padding: const EdgeInsets.all(16),
    decoration: BoxDecoration(borderRadius: BorderRadius.circular(12)),
    child: Column(
     mainAxisAlignment: MainAxisAlignment.center,
     children: [
      Icon(icon, size: 48, color: color),
      const SizedBox(height: 12),
      Text(
       title,
                style: const TextStyle(
                  fontSize: 16,
                  fontWeight: FontWeight.w600,
                ),
              ),
            ],
          ),
        ),
      ),
    );
  }
}

class ProfileScreen extends StatelessWidget {
  final Map<String, dynamic> userData;

  const ProfileScreen({super.key, required this.userData});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: const Text('Profile')),
      body: SingleChildScrollView(
        padding: const EdgeInsets.all(20),
        child: Column(
          children: [
            // Profile picture
            const CircleAvatar(
              radius: 60,
              backgroundColor: Colors.blue,
              child: Icon(Icons.person, size: 80, color: Colors.white),
            ),
            const SizedBox(height: 20),

            // User info cards
            _buildInfoCard('Name', userData['name']),
            _buildInfoCard('Email', userData['email']),
            _buildInfoCard('Role', userData['role']),
            _buildInfoCard('User ID', userData['id']),

            const SizedBox(height: 30),

            // Session info
            Container(
              width: double.infinity,
              padding: const EdgeInsets.all(16),
              decoration: BoxDecoration(
                color: Colors.green[50],
                borderRadius: BorderRadius.circular(12),
                border: Border.all(color: Colors.green[200]!),
              ),
              child: Column(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                  Text(
                    'Session Status',
                    style: TextStyle(
                      fontWeight: FontWeight.bold,
                  color: Colors.green[800],
                 ),
                ),
                const SizedBox(height: 8),
                Text(
                 'Active Session',
                 style: TextStyle(color: Colors.green[700]),
                ),
                Text(
                 'Session expires in 1 hour from login time',
                 style: TextStyle(color: Colors.green[600], fontSize: 12),
                ),
               ],
              ),
             ),
            ],
           ),
          ),
         );
        }

        Widget _buildInfoCard(String label, String value) {
         return Container(
          width: double.infinity,
          margin: const EdgeInsets.only(bottom: 12),
          padding: const EdgeInsets.all(16),
          decoration: BoxDecoration(
           color: Colors.grey[50],
           borderRadius: BorderRadius.circular(12),
           border: Border.all(color: Colors.grey[300]!),
          ),
          child: Column(
           crossAxisAlignment: CrossAxisAlignment.start,
           children: [
            Text(
             label,
             style: const TextStyle(
              fontSize: 12,
              fontWeight: FontWeight.w600,
              color: Colors.grey,
             ),
            ),
            const SizedBox(height: 4),
            Text(
             value,
             style: const TextStyle(fontSize: 16, fontWeight: FontWeight.w500),
            ),
           ],
          ),
         );
        }
       }
```

Output:-



Login

Welcome Back!

Please sign in to your account

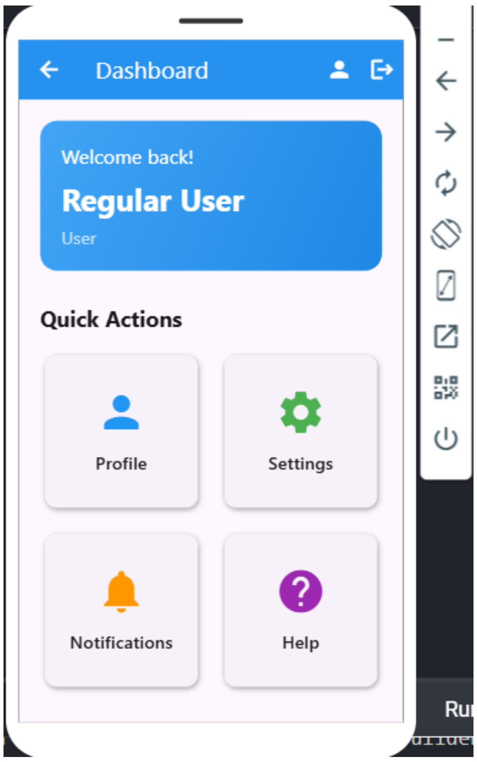
Email: harshil@gmail.com

Password: ••••••••

Login

Demo Credentials:

```
admin@gmail.com / @Admin1804
harshil@gmail.com / @Ram1804
```



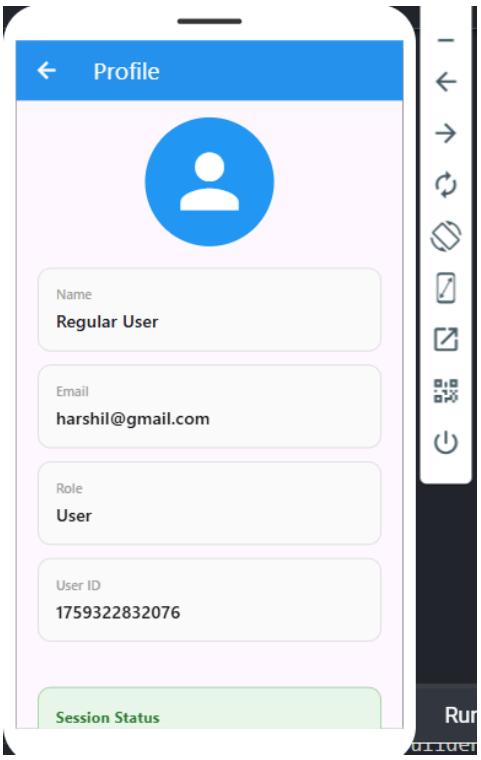
Dashboard

Welcome back!

Regular User

User

Quick Actions



Profile

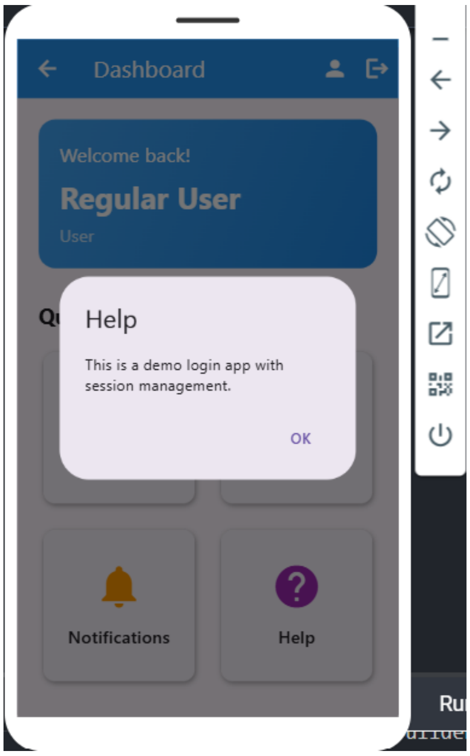
Name: Regular User

Email: harshil@gmail.com

Role: User

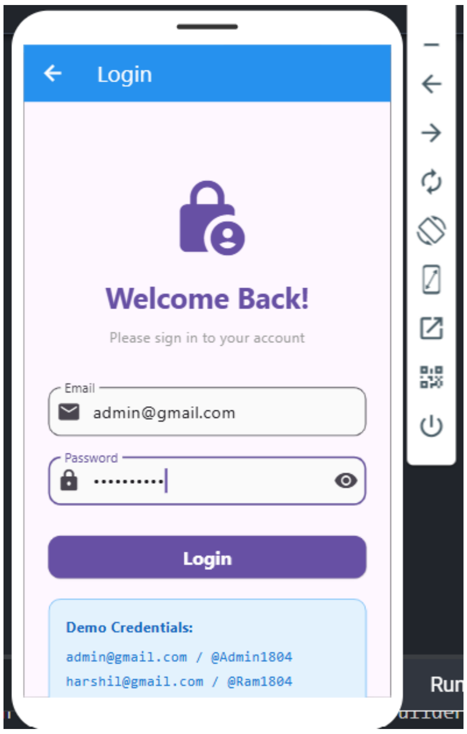
User ID: 1759322832076

Session Status

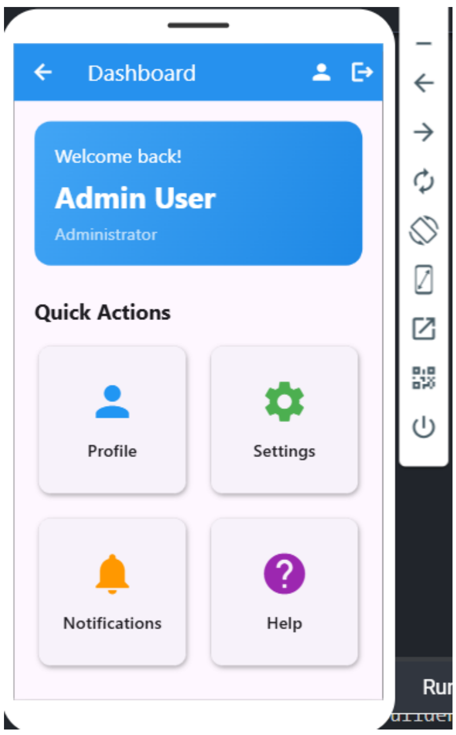


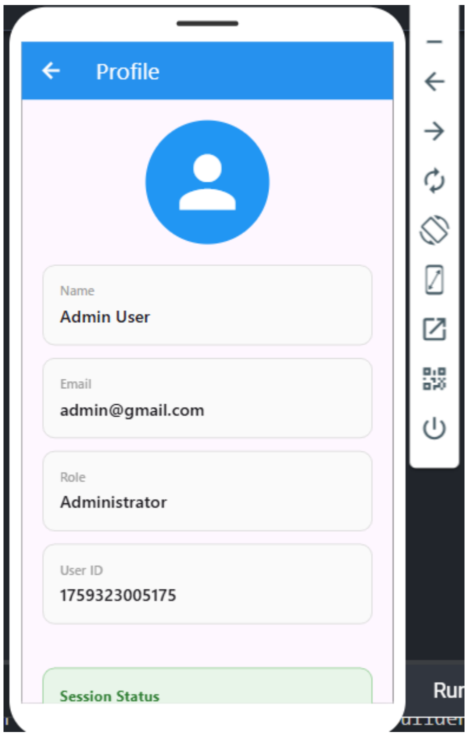
Help

This is a demo login app with session management.

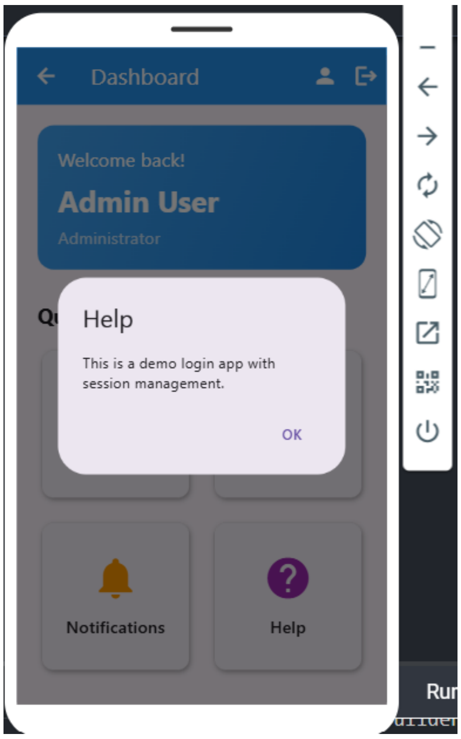


Demo Credentials:
admin@gmail.com / @Admin1804
harshil@gmail.com / @Ram1804

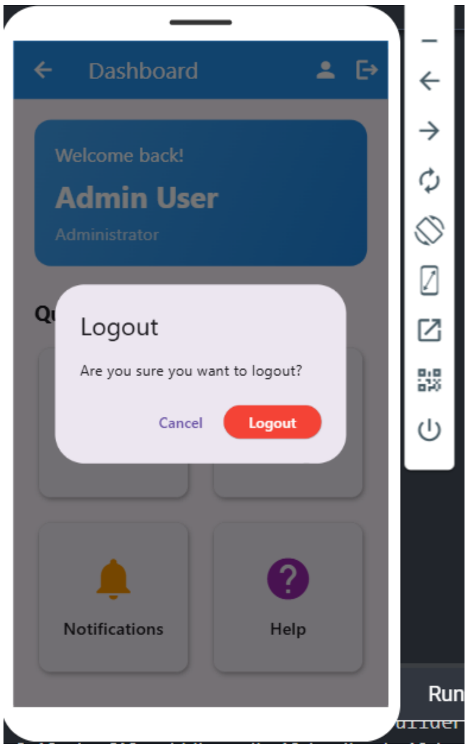




Name
Admin User
Email
admin@gmail.com
Role
Administrator
User ID
1759323005175



This is a demo login app with session management.


Dashboard
Welcome back!
Admin User
Administrator
Logout
Are you sure you want to logout?
Cancel
Logout
Notifications
Help
Run