

### Question 3

```
import 'dart:convert';

import 'package:flutter/material.dart';
import 'package:http/http.dart' as http;
import 'package:shared_preferences/shared_preferences.dart';
import 'package:url_launcher/url_launcher.dart';

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

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

  static const String homeRoute = '/home';
  static const String weatherRoute = '/weather';
  static const String tasksRoute = '/tasks';
  static const String contactRoute = '/contact';
  static const String aboutRoute = '/about';

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Student Connect',
      theme: ThemeData(
        colorSchemeSeed: Colors.lightBlue,
        useMaterial3: true,
      ),
      initialRoute: homeRoute,
      routes: {
        homeRoute: (context) => const HomeScreen(),
        weatherRoute: (context) => const WeatherScreen(),
        tasksRoute: (context) => const TasksScreen(),
        contactRoute: (context) => const ContactAdminScreen(),
        aboutRoute: (context) => const AboutScreen(),
      },
    );
  }
}
```

```

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

  @override
  Widget build(BuildContext context) {
    return Drawer(
      child: ListView(
        children: [
          const DrawerHeader(
            decoration: BoxDecoration(
              color: Colors.lightBlue,
            ),
            child: Text(
              'Student Connect',
              style: TextStyle(
                fontSize: 24,
                color: Colors.white,
              ),
            ),
          ),
          ListTile(
            leading: const Icon(Icons.home),
            title: const Text('Home'),
            onTap: () {
              Navigator.pushNamedAndRemoveUntil(
                context, StudentConnectApp.homeRoute, (route) =>
false);
            },
          ),
          ListTile(
            leading: const Icon(Icons.cloud),
            title: const Text('Weather'),
            onTap: () {
              Navigator.pushNamedAndRemoveUntil(
                context, StudentConnectApp.weatherRoute, (route) =>
false);
            },
          ),
          ListTile(
            leading: const Icon(Icons.check_box),
            title: const Text('Tasks'),
            onTap: () {
              Navigator.pushNamedAndRemoveUntil(

```

```

                context, StudentConnectApp.tasksRoute, (route) =>
false);
        },
    ),
    ListTile(
        leading: const Icon(Icons.contact_mail),
        title: const Text('Contact Admin'),
        onTap: () {
            Navigator.pushNamedAndRemoveUntil(
                context, StudentConnectApp.contactRoute, (route) =>
false);
        },
    ),
    ListTile(
        leading: const Icon(Icons.info),
        title: const Text('About'),
        onTap: () {
            Navigator.pushNamedAndRemoveUntil(
                context, StudentConnectApp.aboutRoute, (route) =>
false);
        },
    ),
],
),
);
}
}

// Home Screen with banner image and background color
class HomeScreen extends StatelessWidget {
  const HomeScreen({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      drawer: const AppDrawer(),
      appBar: AppBar(title: const Text('Home')),
      body: Container(
        color: Colors.lightBlue.shade50,
        child: ListView(
          padding: const EdgeInsets.all(16),
          children: [
            ClipRRect(

```

```

        borderRadius: BorderRadius.circular(12),
        child: Image.network(
'https://images.unsplash.com/photo-1504384308090-c894fdcc538d?auto=format&fit=crop&w=800&q=80',
            height: 180,
            fit: BoxFit.cover,
        ),
    ),
    const SizedBox(height: 16),
    const Text(
        'Welcome to Student Connect!',
        style: TextStyle(
            fontSize: 24,
            fontWeight: FontWeight.bold,
            color: Colors.lightBlue,
        ),
        textAlign: TextAlign.center,
    ),
    const SizedBox(height: 12),
    const Text(
        'This app helps you keep track of weather updates, your
personal tasks, and allows you to contact the admin easily.',
        style: TextStyle(fontSize: 16),
        textAlign: TextAlign.center,
    ),
    const SizedBox(height: 24),
    Card(
        elevation: 4,
        shape: RoundedRectangleBorder(
            borderRadius: BorderRadius.circular(10)),
        child: ListTile(
            leading: const Icon(Icons.cloud, color:
Colors.lightBlue),
            title: const Text('Check Weather'),
            trailing: const Icon(Icons.arrow_forward_ios),
            onTap: () =>
                Navigator.pushNamed(context,
StudentConnectApp.weatherRoute),
        ),
    ),
    const SizedBox(height: 12),
    Card(

```

```

        elevation: 4,
        shape: RoundedRectangleBorder(
          borderRadius: BorderRadius.circular(10)),
        child: ListTile(
          leading: const Icon(Icons.check_box, color:
Colors.lightBlue),
          title: const Text('Manage Tasks'),
          trailing: const Icon(Icons.arrow_forward_ios),
          onTap: () =>
            Navigator.pushNamed(context,
StudentConnectApp.tasksRoute),
        ),
      ),
      const SizedBox(height: 12),
      Card(
        elevation: 4,
        shape: RoundedRectangleBorder(
          borderRadius: BorderRadius.circular(10)),
        child: ListTile(
          leading: const Icon(Icons.contact_mail, color:
Colors.lightBlue),
          title: const Text('Contact Admin'),
          trailing: const Icon(Icons.arrow_forward_ios),
          onTap: () =>
            Navigator.pushNamed(context,
StudentConnectApp.contactRoute),
        ),
      ),
    ],
  ),
);
}
}

// Weather Screen showing previous days' temperature history
class WeatherScreen extends StatefulWidget {
  const WeatherScreen({super.key});

  @override
  State<WeatherScreen> createState() => _WeatherScreenState();
}

```

```

class _WeatherScreenState extends State<WeatherScreen> {
  List<String> dates = [];
  List<double> temperatures = [];
  bool isLoading = true;
  String? error;

  Future<void> fetchWeatherHistory() async {
    setState(() {
      isLoading = true;
      error = null;
    });

    // Fetch last 7 days including today
    // Open-Meteo API expects dates in yyyy-MM-dd format
    final now = DateTime.now();
    final startDate = now.subtract(const Duration(days: 6));
    final startDateStr = "${startDate.year.toString().padLeft(4, '0')}-${
      "${startDate.month.toString().padLeft(2, '0')}-${
      "${startDate.day.toString().padLeft(2, '0')}}";
    final endDateStr = "${now.year.toString().padLeft(4, '0')}-${
      "${now.month.toString().padLeft(2, '0')}-${
      "${now.day.toString().padLeft(2, '0')}}";

    final url = Uri.parse(
      'https://api.open-meteo.com/v1/forecast?latitude=12.9716&longitude=77.5
      946&start_date=$startDateStr&end_date=$endDateStr&daily=temperature_2m_
      max,temperature_2m_min&timezone=Asia/Kolkata');

    try {
      final response = await http.get(url);
      if (response.statusCode == 200) {
        final data = jsonDecode(response.body);
        final daily = data['daily'];
        final List<String> fetchedDates =
          List<String>.from(daily['time'] ?? []);
        final List<dynamic> tempMaxList = daily['temperature_2m_max'] ??
[];
        final List<dynamic> tempMinList = daily['temperature_2m_min'] ??
[];

        if (fetchedDates.isEmpty ||
            tempMaxList.isEmpty ||

```

```

        tempMinList.isEmpty ||
        fetchedDates.length != tempMaxList.length ||
        fetchedDates.length != tempMinList.length) {
      setState(() {
        error = 'Incomplete data received.';
        isLoading = false;
      });
      return;
    }

    setState(() {
      dates = fetchedDates;
      // Store average temperature for each day
      temperatures = List.generate(
        tempMaxList.length,
        (index) =>
          ((tempMaxList[index] + tempMinList[index]) /
2).toDouble());
      isLoading = false;
    });
  } else {
    setState(() {
      error = 'Failed to load weather data.';
      isLoading = false;
    });
  }
} catch (e) {
  setState(() {
    error = 'Error fetching weather data.';
    isLoading = false;
  });
}
}

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

@override
Widget build(BuildContext context) {
  return Scaffold(

```

```

drawer: const AppDrawer(),
appBar: AppBar(title: const Text('Weather History')),
body: isLoading
    ? const Center(child: CircularProgressIndicator())
    : error != null
      ? Center(child: Text(error!, style: const TextStyle(color:
Colors.red)))
      : Padding(
padding: const EdgeInsets.all(16),
child: Column(
children: [
const Icon(Icons.cloud_queue,
size: 80, color: Colors.lightBlue),
const SizedBox(height: 16),
Text(
'Bengaluru - Last 7 Days Avg Temperature',
style:
Theme.of(context).textTheme.headlineSmall,
textAlign: TextAlign.center,
),
const SizedBox(height: 12),
Expanded(
child: ListView.builder(
itemCount: dates.length,
itemBuilder: (context, index) {
final date = dates[index];
final temp = temperatures[index];
final dateFormatted = DateTime.parse(date);
final displayDate =
"$${dateFormatted.day}-${dateFormatted.month}-${dateFormatted.year}";

return Card(
margin: const
EdgeInsets.symmetric(vertical: 6),
child: ListTile(
leading: const Icon(Icons.thermostat,
color: Colors.orange),
title: Text(displayDate),
trailing: Text(
'${temp.toStringAsFixed(1)} °C',
style: const TextStyle(
fontWeight: FontWeight.bold,

```



```

        fontSize: 18),
      ),
    ),
  );
},
),
),
ElevatedButton.icon(
  onPressed: fetchWeatherHistory,
  icon: const Icon(Icons.refresh),
  label: const Text('Refresh'),
),
],
),
),
);
}
}

// Tasks Screen with CRUD & SharedPreferences storage
class TasksScreen extends StatefulWidget {
  const TasksScreen({super.key});

  @override
  State<TasksScreen> createState() => _TasksScreenState();
}

class _TasksScreenState extends State<TasksScreen> {
  List<String> tasks = [];
  final _taskController = TextEditingController();
  int? editingIndex;

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

  Future<void> _loadTasks() async {
    final prefs = await SharedPreferences.getInstance();
    final savedTasks = prefs.getStringList('tasks') ?? [];
    setState(() {
      tasks = savedTasks;
    });
  }
}

```

```

    });
}

Future<void> _saveTasks() async {
  final prefs = await SharedPreferences.getInstance();
  await prefs.setStringList('tasks', tasks);
}

void _showTaskDialog({String? initialText, int? index}) {
  if (initialText != null) {
    _taskController.text = initialText;
    editingIndex = index;
  } else {
    _taskController.clear();
    editingIndex = null;
  }
  showDialog(
    context: context,
    builder: (context) {
      return AlertDialog(
        title: Text(editingIndex == null ? 'Add Task' : 'Edit Task'),
        content: TextField(
          controller: _taskController,
          decoration: const InputDecoration(
            hintText: 'Enter task',
          ),
          autofocus: true,
        ),
        actions: [
          TextButton(
            onPressed: () {
              Navigator.pop(context);
              _taskController.clear();
            },
            child: const Text('Cancel')),
          ElevatedButton(
            onPressed: () {
              final text = _taskController.text.trim();
              if (text.isNotEmpty) {
                setState(() {
                  if (editingIndex == null) {
                    tasks.add(text);
                  } else {

```

```

        tasks[editingIndex!] = text;
    }
    });
    _saveTasks();
    Navigator.pop(context);
    _taskController.clear();
  }
},
child: const Text('Save')),
],
);
},
);
}

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

@override
Widget build(BuildContext context) {
  return Scaffold(
    drawer: const AppDrawer(),
    appBar: AppBar(title: const Text('Tasks')),
    body: tasks.isEmpty
      ? const Center(
        child: Text(
          'No tasks added yet.\nTap + to add your first task.',
          textAlign: TextAlign.center,
          style: TextStyle(fontSize: 18),
        ),
      )
      : ListView.builder(
        itemCount: tasks.length,
        itemBuilder: (context, index) {
          return Card(
            margin:
              const EdgeInsets.symmetric(horizontal: 12,
vertical: 6),
            child: ListTile(

```

```

        leading: const Icon(Icons.task_alt, color:
Colors.lightBlue),
        title: Text(tasks[index]),
        trailing: Row(
          mainAxisAlignment: MainAxisAlignment.min,
          children: [
            IconButton(
              icon: const Icon(Icons.edit, color:
Colors.orange),
              onPressed: () => _showTaskDialog(
                initialText: tasks[index], index: index),
            ),
            IconButton(
              icon: const Icon(Icons.delete, color:
Colors.red),
              onPressed: () => _deleteTask(index),
            ),
          ],
        ),
      ),
    ),
  );
},
),
floatingActionButton: FloatingActionButton(
  onPressed: () => _showTaskDialog(),
  tooltip: 'Add Task',
  child: const Icon(Icons.add),
),
);
}
}

// Contact Admin screen with a form, input & mail launcher
class ContactAdminScreen extends StatefulWidget {
  const ContactAdminScreen({super.key});

  @override
  State<ContactAdminScreen> createState() => _ContactAdminScreenState();
}

class _ContactAdminScreenState extends State<ContactAdminScreen> {
  final _formKey = GlobalKey<FormState>();
  final nameController = TextEditingController();

```

```

final _emailController = TextEditingController();
final _messageController = TextEditingController();

Future<void> _sendEmail() async {
  final name = Uri.encodeComponent(_nameController.text.trim());
  final email = Uri.encodeComponent(_emailController.text.trim());
  final message = Uri.encodeComponent(_messageController.text.trim());

  final mailtoUrl =

'mailto:admin@college.edu?subject=Student%20Contact&body=Name:%20$name%
0AEmail:%20$email%0A%0AMessage:%20$message';

  if (await canLaunchUrl(Uri.parse(mailtoUrl))) {
    await launchUrl(Uri.parse(mailtoUrl));
  } else {
    ScaffoldMessenger.of(context).showSnackBar(
      const SnackBar(content: Text('Could not open mail app')),
    );
  }
}

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

@override
Widget build(BuildContext context) {
  return Scaffold(
    drawer: const AppDrawer(),
    appBar: AppBar(title: const Text('Contact Admin')),
    body: Padding(
      padding: const EdgeInsets.all(16),
      child: Form(
        key: _formKey,
        child: ListView(
          children: [
            const Icon(Icons.contact_mail,
              size: 80, color: Colors.lightBlue),

```

```

const SizedBox(height: 16),
TextFormField(
  controller: _nameController,
  decoration: const InputDecoration(
    labelText: 'Name',
    border: OutlineInputBorder(),
    prefixIcon: Icon(Icons.person),
  ),
  validator: (value) =>
    value == null || value.isEmpty ? 'Enter your name' :
null,

),
const SizedBox(height: 12),
TextFormField(
  controller: _emailController,
  decoration: const InputDecoration(
    labelText: 'Email',
    border: OutlineInputBorder(),
    prefixIcon: Icon(Icons.email),
  ),
  keyboardType: TextInputType.emailAddress,
  validator: (value) {
    if (value == null || value.isEmpty) {
      return 'Enter your email';
    }
    final emailRegex = RegExp(r'^^[^@]+@^[^@]+\.[^@]+' );
    if (!emailRegex.hasMatch(value)) {
      return 'Enter valid email';
    }
    return null;
  },
),
const SizedBox(height: 12),
TextFormField(
  controller: _messageController,
  decoration: const InputDecoration(
    labelText: 'Message',
    border: OutlineInputBorder(),
    prefixIcon: Icon(Icons.message),
  ),
  maxLines: 5,
  validator: (value) =>

```

```

        value == null || value.isEmpty ? 'Enter your
message' : null,
      ),
      const SizedBox(height: 20),
      ElevatedButton.icon(
        icon: const Icon(Icons.send),
        label: const Text('Send'),
        onPressed: () {
          if (_formKey.currentState!.validate()) {
            _sendEmail();
          }
        },
      ),
    ],
  ),
),
),
);
}
}

// About Screen with app info
class AboutScreen extends StatelessWidget {
  const AboutScreen({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      drawer: const AppDrawer(),
      appBar: AppBar(title: const Text('About')),
      body: Padding(
        padding: const EdgeInsets.all(16),
        child: ListView(
          children: const [
            Icon(Icons.info_outline, size: 80, color: Colors.lightBlue),
            SizedBox(height: 16),
            Text(
              'Student Connect App\n\n'
              'Version 1.0.0\n\n'
              'This app is designed to help students manage their daily
tasks, '
              'check weather forecasts, and easily contact the
administration.',
            ),
          ],
        ),
      ),
    );
  }
}

```


```
        style: TextStyle(fontSize: 16),
        textAlign: TextAlign.center,
      ),
    ],
  ),
);
}
```







## Welcome to Student Connect!






This app helps you keep track of weather updates, your personal tasks, and allows you to contact the admin easily.

 Check Weather >

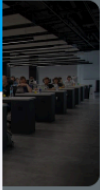
 Manage Tasks >

 Contact Admin >

Student Connect

-  Home
-  Weather
-  Tasks
-  Contact Admin
-  About

**DEMO**



**Connect!**

For updates,  
contact the

>

>

>



## Weather History

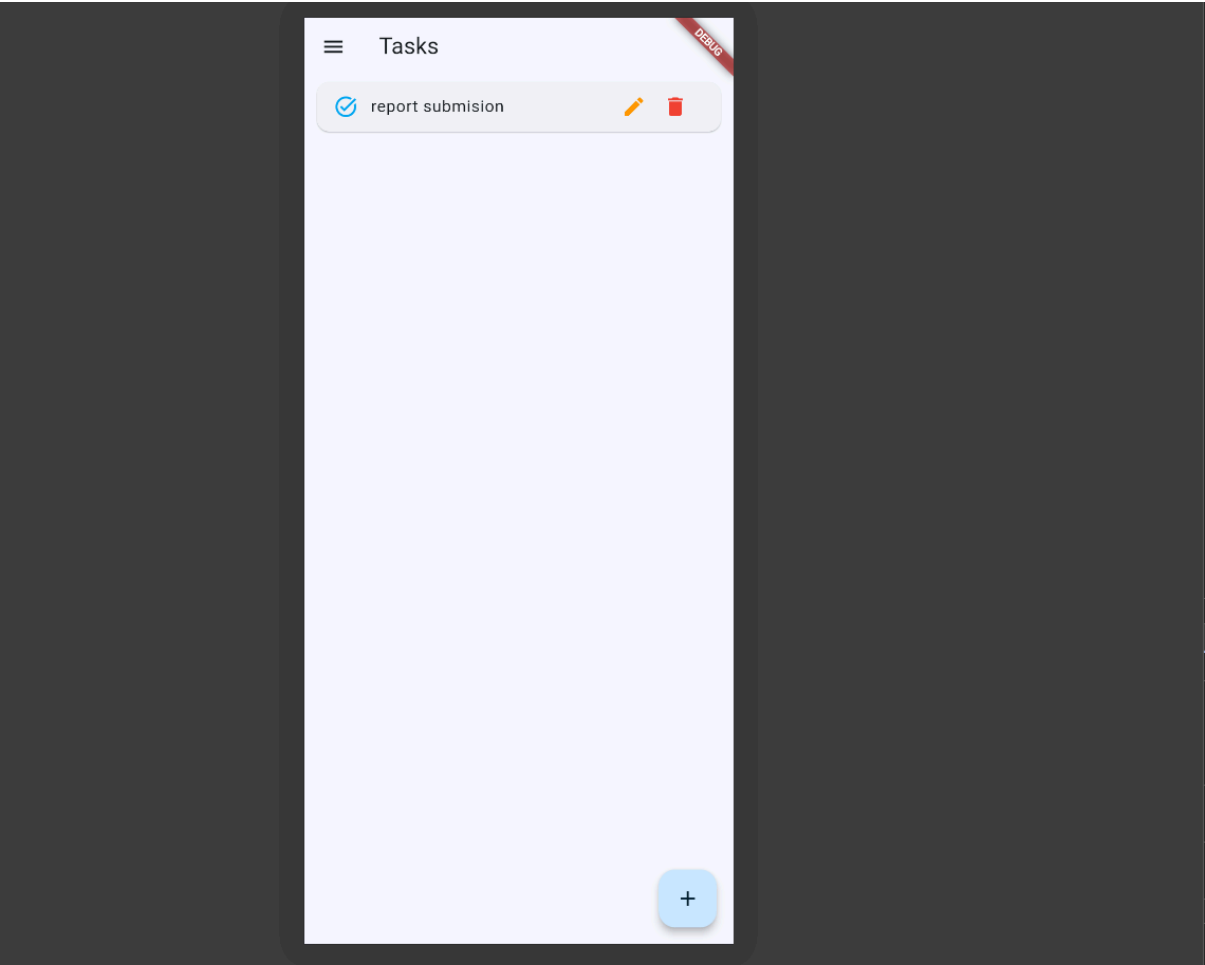
DESIGN



### Bengaluru - Last 7 Days Avg Temperature

 5-8-2025	24.6 °C
 6-8-2025	24.8 °C
 7-8-2025	24.5 °C
 8-8-2025	24.4 °C
 9-8-2025	23.6 °C
 10-8-2025	23.3 °C
 11-8-2025	22.6 °C

 Refresh





## Contact Admin

Design



Name



Email



Message



Send

Choose an application to open the mailto link.

System Handler



Gmail

<https://mail.google.com>

Choose other Application

Choose...

☐ Always use this application to open mailto links

Cancel

Open Link



## Contact Admin

Debug



Name

 xyz

Email

 xyz@gmail.com

Message

hello



 Send



## About

Debug



Student Connect App

Version 1.0.0

This app is designed to help students manage their daily tasks, check weather forecasts, and easily contact the administration.