



ULAB

UNIVERSITY OF LIBERAL ARTS
BANGLADESH

Report

Submitted To :

Shakib Mahmud Dipto

Submitted By :

Name:

1. Robiur Rahman(193014049)
2. Hasin Imam Mehran(193014020)
3. Asib Shikder pranto(193014067)

Project Name: Vehicles Parking Space Finder Apps

Group No: 01

Course Title: Smartphone and Application

Course Code: CSE 438

Section: 01

Friends Circle Company

Date: 17/12/2023

Software Requirements Specification (SRS)

1. Introduction:

1.1 Purpose:

The purpose of the "**Vehicles Parking Space Finder**" app is to provide a convenient and efficient solution for users to locate available parking spaces for their vehicles. This app aims to address the following purposes

- Because of that there are no towing problems.
- And the vehicle has been parked as a secure condition.
- There is no risk for the vehicle owner for parking the car.

1.2 Product Scope:

In the modern age many people have vehicles. Vehicles are now a basic need. Every place is under

The process of urbanization. There are many corporate offices and shopping centers, hospitals,

schools, government organizations etc. These systems might be computerized or non-computerized

with the help of computerized system. We can deliver a good service to customers who want to park their vehicle into any place.

1.3 Document Conventions:

The main chapter headings are given with font. Times and font size of 18. The subheadings of each

chapter are given with font. Times and size of 14. Emphasize is given in Italics. Critical information

in specific scenarios is given in Bold Words.

2. Overall Description:

2.1 Product Perspective:

SPSFA ought to have the option to give an essential and simple exchange of data.

For example: it ought to have the option to eliminate the correspondence holes between a representative and the client.

2.2 Product Functions:

There are a lot of functions in **parking space finder apps**.these function can be placed into broader categories as follows:

1. Efficiency and functionality.
2. Security over performance.
3. Lightweight and elegant.

2.3 User Classes and Characteristics:

A user can just have his/her worker Id number or vehicle plate number as username so assuming he joins the parking then no one but he can login.

2.4 Operating Environment:

There are many OS environment where our products compatible version of OS as describes in hardware and software requirements, so it does not need to rely on any other operating system. Our

Software Requirements Specification for <Project> Page 3

product will provide the best features of both Android system.so it will have to provide complete support to old and new applications designed for both Android systems.

2.5 Design and Implementation Constraints:

Servers and the numbers of client can get to or can be online immediately. More is the quantity of client more will be the organization traffic and the server arrives in a down state .As an operating system the major constraint for parking space will be hardware and its performance there are several trivial and non-trivial constraint which are from both development perspective and context perspective. Some of them are listed as follows here;

1. Memory of the device.
2. Cope with boot up procedures of different manufacturers.
3. Prevent deadlock during application executions.

2.6 User Documentation:

First of all, the end users will be required to sign our terms and conditions while first running the Parking space finder product in their devices. These terms and conditions will contain and agreements relevant to the features implemented in the parking space finder product. Once the parking space has successfully run up for the first time, a detailed documentation page will be shown right on the device, which can be skipped as of now, upon clicking skip, the user will get instant no how to access the detailed documentation inside .

External Interface Requirements:

GUI;

User Interfaces:

This interface should be profoundly natural or intuitive on the ground that there won't be a help for the client who is working the Framework .at most of the spots assist work area with being accommodation client's comfort. The pin and secret word classification ought to be kept up this should be possible by utilizing reference marks at the secret phrase board. Legitimate security message ought to be shown all things consideration of the spots.

Software Requirements Specification for <Project> Page 4

Hardware Interfaces:

The parking space apps will be interfacing with hardware all the time and the real challenge during development is to make most efficient possible usage of hardware components.

Otherwise there will be no market of new parking space that does not even provide better performance .The parking space will be capable of running on all devices whether that be media and online servers and operating system.it will be interfacing with all the hardware components.

Software Interfaces:

- The last application should be bundled in a setup program with the goal that the items can be effectively introduced on machines.
- For the information base taking care of MYSQL should be introduced .These items are open sources items.

System Features:

3. Non-Functional Requirements

3.1 Performance :

- The system responds quickly.
- The tool for altering clothing must provide a responsive and enjoyable user experience.

3.2 Security :

- Passwords for users are entirely secure.
- The system encrypted sensitive data and user accounts.

3.3 Usability :

- The user interface has excellent usability.
- The tool for customizing clothing must provide thorough instructions.

3.4 Scalability :

- A lot of users should be able to use the system without any issues.
- The system should be easily expandable in order to allow for future expansion.

4. Constraints

- It is an Android application .
- The system should be developed by Dart

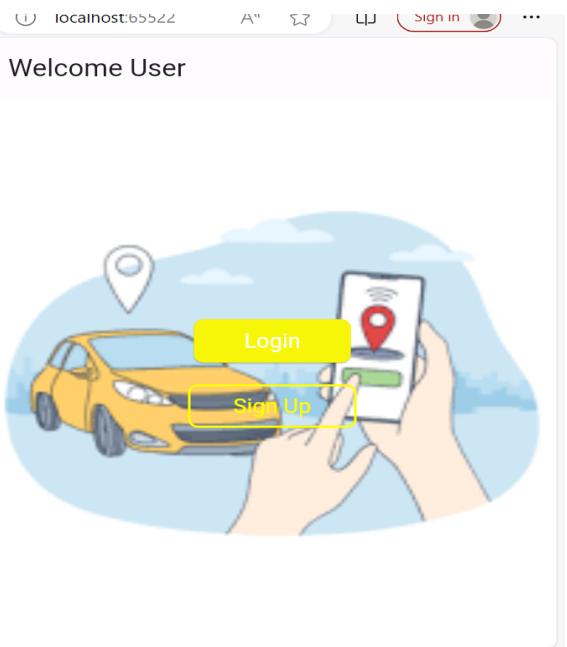
Outline :

- Authentication
- Log In (Admin & User& Space Owner)
- Sing Up
- View map

- User profile
- Parking details
- Confirm booking
- payment

Code Screenshot:

Authentication:



The screenshot shows the 'authentication_screen.dart' code on the left and the resulting mobile application interface on the right.

```

23   ElevatedButton(
24     onPressed: () {
25       Navigator.push(
26         context,
27         MaterialPageRoute(builder: (context) => LoginS
28       );
29     },
30     style: ElevatedButton.styleFrom(
31       primary: Colors.yellowAccent, // Background color
32       onPrimary: Colors.white, // Text color
33       padding: EdgeInsets.symmetric(horizontal: 40, ve
34       shape: RoundedRectangleBorder(
35         borderRadius: BorderRadius.circular(8.0),
36       ), // RoundedRectangleBorder
37     ),
38     child: Text(
39       'Login',
40       style: TextStyle(fontSize: 18),
41     ), // Text
42   ), // ElevatedButton
43   SizedBox(height: 20),
44   OutlinedButton(

```

The application interface displays a 'Welcome User' header. Below it is a circular illustration featuring a yellow car, a hand holding a smartphone with a login screen, and location pins. Two prominent yellow buttons labeled 'Login' and 'Sign Up' are overlaid on the phone screen.

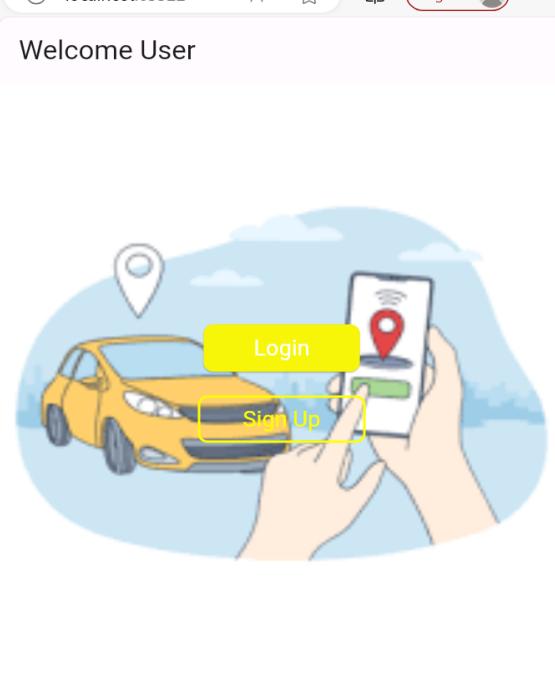
authentication_screen.dart pubspec.yaml login_screen.dart signup_screen.dart

Android SDK "Android API 29 Platform" is missing

```
1 import 'package:flutter/material.dart';
2 import 'login_screen.dart';
3 import 'signup_screen.dart';
4
5 class AuthenticationScreen extends StatelessWidget {
6   @override
7   Widget build(BuildContext context) {
8     return Scaffold(
9       appBar: AppBar(
10         title: Text('Welcome User'),
11       ), // AppBar
12       body: Container(
13         decoration: BoxDecoration(
14           image: DecorationImage(
15             image: AssetImage('assets/abc.png'), // Replace with your own image
16             fit: BoxFit.cover,
17           ), // DecorationImage
18         ), // BoxDecoration
19         child: Center(
20           child: Column(
21             mainAxisAlignment: MainAxisAlignment.center,
22             children: [

```

TODO Problems Terminal App Quality Insights App Inspection Logcat Services



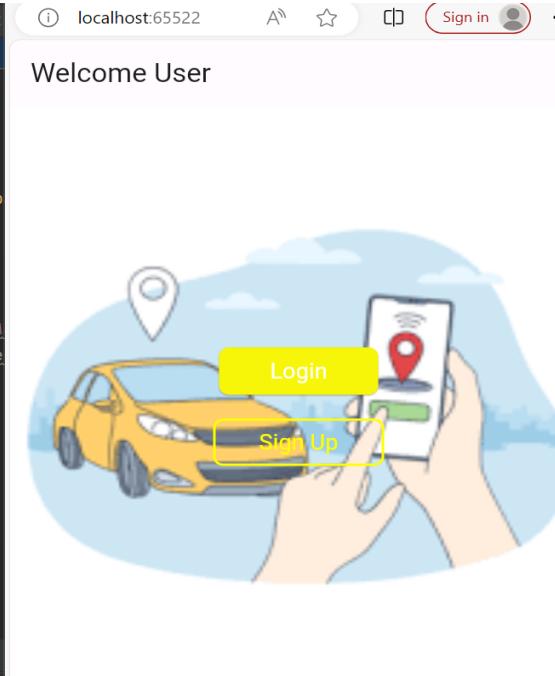
authentication_screen.dart pubspec.yaml login_screen.dart signup_screen.dart

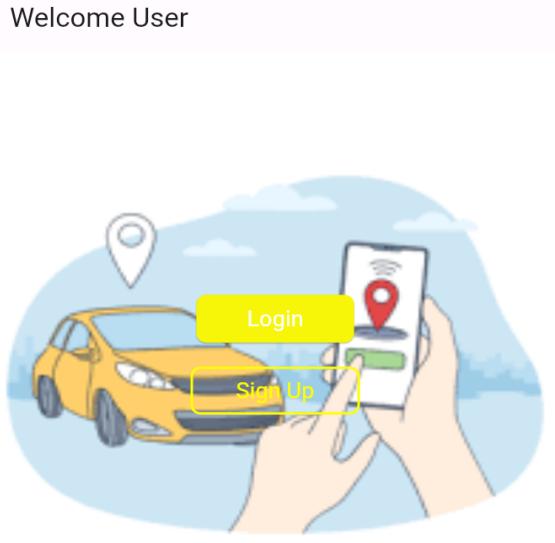
Android SDK "Android API 29 Platform" is missing

```
44   OutlinedButton(
45     onPressed: () {
46       Navigator.push(
47         context,
48         MaterialPageRoute(builder: (context) => SignUp()),
49     );
50   },
51   style: OutlinedButton.styleFrom(
52     primary: Colors.yellowAccent, // Border color
53     side: BorderSide(width: 2, color: Colors.yellowA),
54     padding: EdgeInsets.symmetric(horizontal: 36, ve
55     shape: RoundedRectangleBorder(
56       borderRadius: BorderRadius.circular(8.0),
57     ), // RoundedRectangleBorder
58   ),
59   child: Text(
60     'Sign Up',
61     style: TextStyle(fontSize: 18),
62   ), // Text
63   ), // OutlinedButton
64 ],

```

TODO Problems Terminal App Quality Insights App Inspection Logcat Services

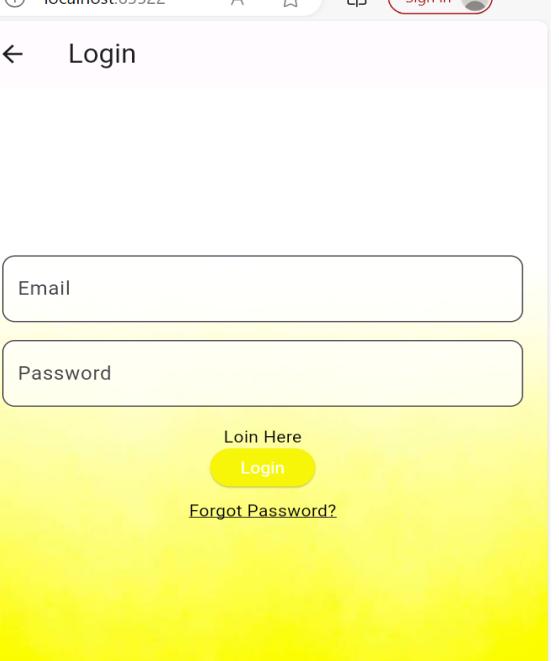




The screenshot shows a Flutter development environment with several files open in the top bar: `authentication_screen.dart`, `pubspec.yaml`, `login_screen.dart`, and `signup_screen.dart`. A message in the top right corner indicates that the Android SDK "Android API 29 Platform" is missing. The main view displays a "Welcome User" header and a central UI element. This element features a yellow car icon on the left, a yellow rounded rectangle containing a "Login" button in the center, and a hand holding a smartphone displaying a "Sign Up" button on the right. The entire UI is set against a light blue circular background.

```
padding: EdgeInsets.symmetric(horizontal: 36.0),
        shape: RoundedRectangleBorder(
            borderRadius: BorderRadius.circular(8.0),
        ), // RoundedRectangleBorder
        child: Text(
            'Sign Up',
            style: TextStyle(fontSize: 18),
        ), // Text
    ], // Column
), // Center
), // Container
); // Scaffold
}
}
```

Login:

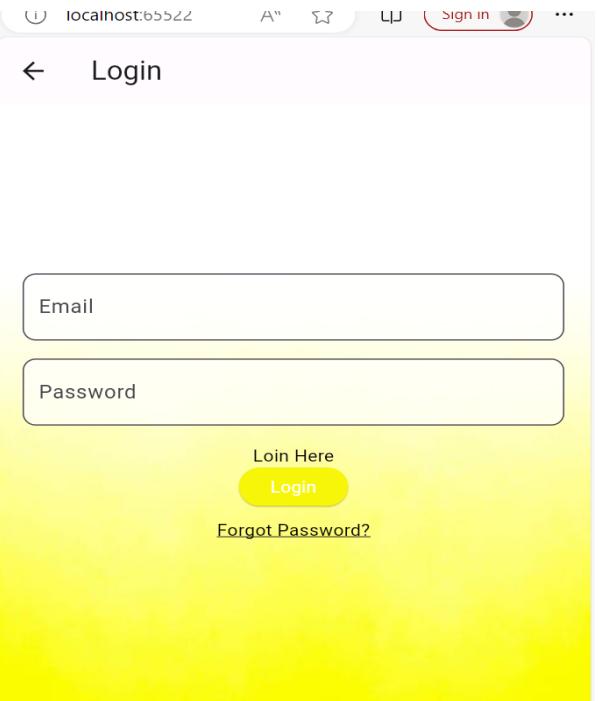


A screenshot of a Flutter application running in a browser at localhost:65522. The screen shows a 'Login' page with a yellow gradient background. It features two input fields: 'Email' and 'Password'. Below the inputs is a 'Login Here' button with rounded corners and a yellow gradient, which also contains the word 'Login'. To the right of the button is a link 'Forgot Password?'. The top of the screen shows the code editor with the file 'login_screen.dart' open, displaying the Dart code for the LoginScreen widget.

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

class LoginScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Login'),
      ),
      body: Container(
        decoration: BoxDecoration(
          image: DecorationImage(
            image: AssetImage('assets/vvv.jpg'), // Replace with your own image
            fit: BoxFit.cover,
          ), // DecorationImage
        ), // BoxDecoration
        child: Center(
          child: Padding(
            padding: const EdgeInsets.all(20.0),
            child: Column(
              mainAxisAlignment: MainAxisAlignment.center,

```



A screenshot of a Flutter application running in a browser at localhost:65522. The screen shows a 'Login' page with a white background. It features two input fields: 'Email' and 'Password'. Below the inputs is a 'Login Here' button with rounded corners and a yellow gradient, which also contains the word 'Login'. To the right of the button is a link 'Forgot Password?'. The top of the screen shows the code editor with the file 'login_screen.dart' open, displaying the Dart code for the LoginScreen widget, which uses the InputDecoration class for the text fields.

```
mainAxisAlignment: MainAxisAlignment.center,
      children: [
        // Email Text Field
        TextField(
          decoration: InputDecoration(
            labelText: 'Email',
            border: OutlineInputBorder(
              borderRadius: BorderRadius.circular(10),
            ), // OutlineInputBorder
            filled: true,
            fillColor: Colors.white70,
          ), // InputDecoration
        ), // TextField
        SizedBox(height: 15),
        // Password Text Field
        TextField(
          obscureText: true,
          decoration: InputDecoration(
            labelText: 'Password',
            border: OutlineInputBorder(
              borderRadius: BorderRadius.circular(10),

```

The screenshot shows a Flutter application interface. At the top, there is a navigation bar with a back arrow and the word "Login". Below this is a yellow gradient background. On the left, there are two input fields: one for "Email" and one for "Password", both with rounded corners. In the center, there is a yellow-outlined "ElevatedButton" labeled "Login". To the right of the button, there is a link labeled "Forgot Password?". At the bottom of the screen, there is a footer bar with various icons and links.

```

43     borderRadius: BorderRadius.circular(10),
44   ),
45   // OutlineInputBorder
46   filled: true,
47   fillColor: Colors.white70,
48 ), // InputDecoration
49 ), // TextField
50   SizedBox(height: 15),
51   Text(
52     'Login Here',
53     style: TextStyle(color: Colors.black),
54   ), // Text
55   ElevatedButton(
56     onPressed: () {
57       // Assume successful login, navigate to View
58       Navigator.pushReplacement(
59         context,
60         MaterialPageRoute(builder: (context) => View
61       );
62     },
63     style: ElevatedButton.styleFrom(

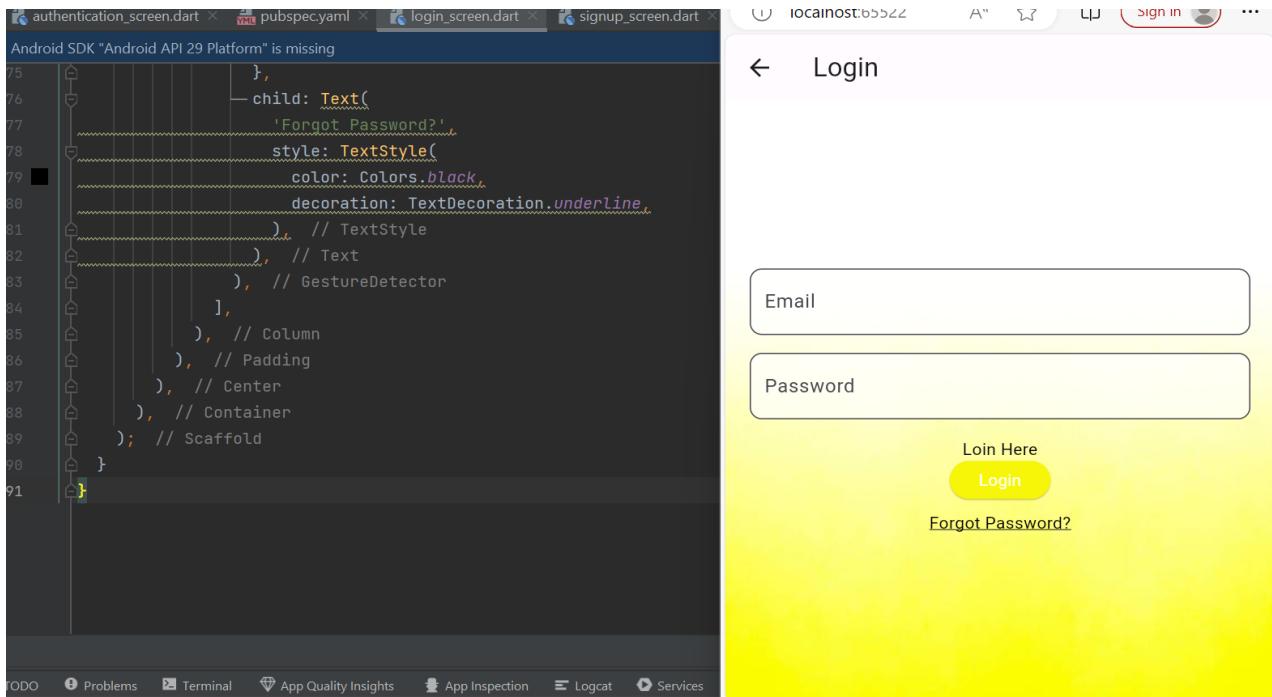
```

This screenshot shows the same Flutter application as the first one, but it is displayed in a web browser window. The browser's address bar shows "localhost:65522". The page content is identical to the first screenshot, featuring a yellow gradient background, input fields for "Email" and "Password", a central "Login" button, and a "Forgot Password?" link. The "Sign in" button from the browser's header is visible at the top right of the page.

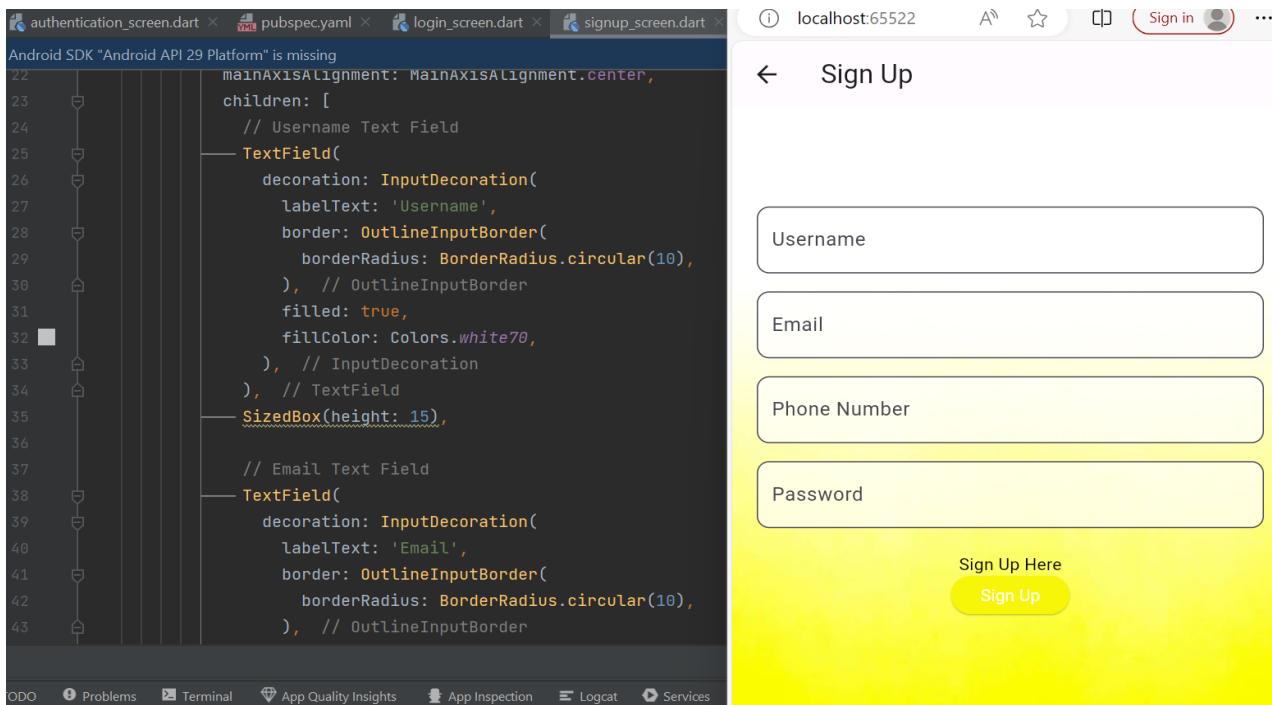
```

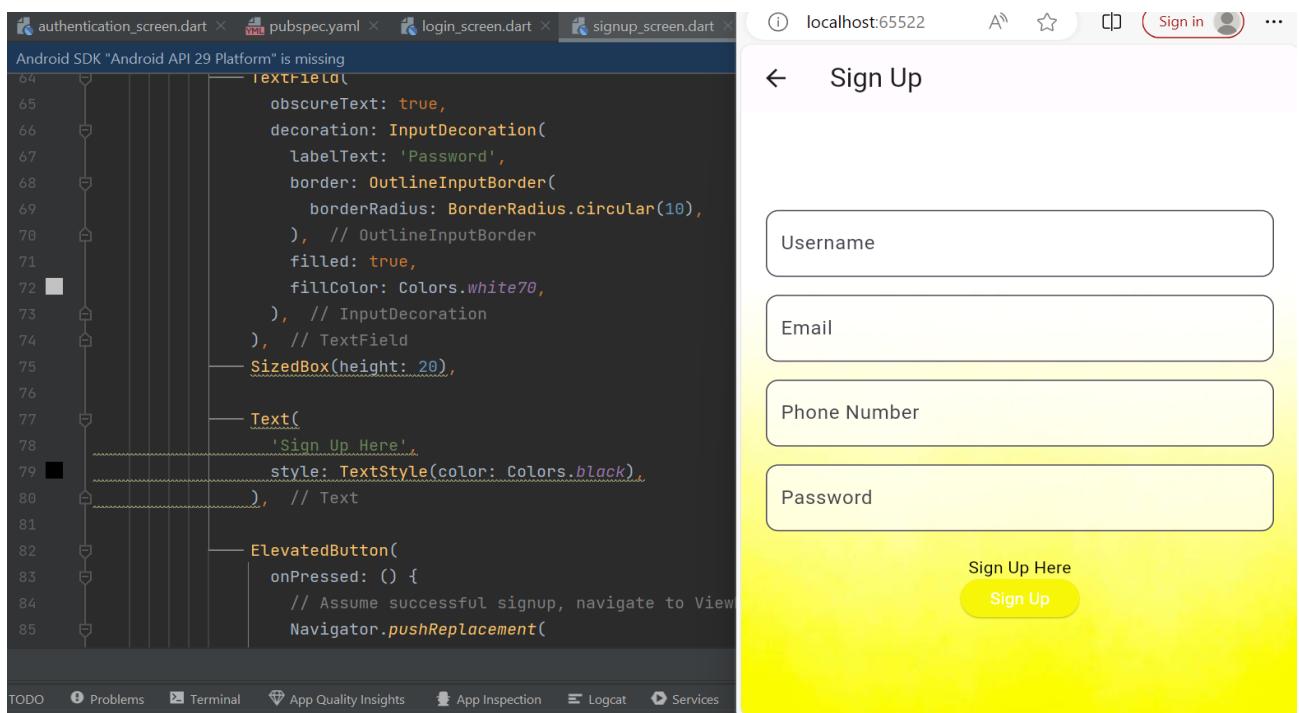
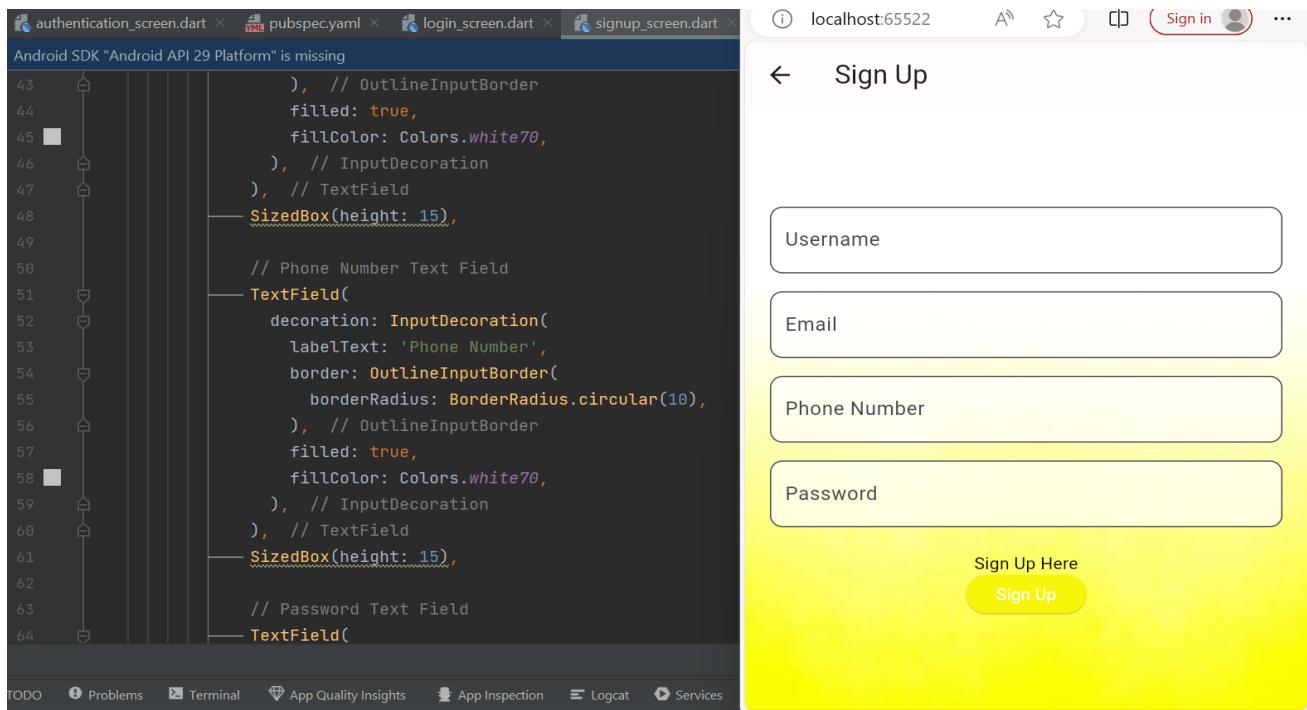
43     borderRadius: BorderRadius.circular(10),
44   ),
45   // OutlineInputBorder
46   filled: true,
47   fillColor: Colors.white70,
48 ), // InputDecoration
49 ), // TextField
50   SizedBox(height: 15),
51   Text(
52     'Login Here',
53     style: TextStyle(color: Colors.black),
54   ), // Text
55   ElevatedButton(
56     onPressed: () {
57       // Assume successful login, navigate to View
58       Navigator.pushReplacement(
59         context,
60         MaterialPageRoute(builder: (context) => View
61       );
62     },
63     style: ElevatedButton.styleFrom(

```



Signup:





localhost:65522

Sign Up

Username

Email

Phone Number

Password

Sign Up Here

Sign Up

```

85     Navigator.pushReplacement(
86         context,
87         MaterialPageRoute(builder: (context) => View
88     );
89 },
90     style: ElevatedButton.styleFrom(
91         primary: Colors.yellowAccent, // Background c
92         onPrimary: Colors.white, // Text color
93     ),
94     child: Text('Sign Up'),
95 ),
96 ],
97 ),
98 ),
99 ),
100 ),
101 );
102 }
103 }
```

user profile:

localhost:65522

Profile

Name

Location

Phone Number

Email

Settings

Log Out

```

1 import 'package:flutter/material.dart';
2
3 class UserProfileScreen extends StatelessWidget {
4   @override
5   Widget build(BuildContext context) {
6     return Scaffold(
7       appBar: AppBar(
8         title: Text('Profile'),
9       ),
10      body: SingleChildScrollView(
11        child: Padding(
12          padding: const EdgeInsets.all(20.0),
13          child: Column(
14            mainAxisAlignment: MainAxisAlignment.center,
15            children: [
16              // User Profile Picture
17              GestureDetector(
18                onTap: () {
19                  // Implement a method to choose or capture a profile picture
20                },
21                child: CircleAvatar(
22                  radius: 70,
```

authentication_screen.dart × pubspec.yaml × login_screen.dart × signup_screen.dart

Android SDK "Android API 29 Platform" is missing

```
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
```

```
// User Location
Padding(
  padding: const EdgeInsets.symmetric(horizontal: 3),
  child: TextField(
    decoration: InputDecoration(
      labelText: 'Location',
      border: OutlineInputBorder(
        borderRadius: BorderRadius.circular(10),
      ),
    ),
  ),
),
// Padding
SizedBox(height: 10),

// User Phone Number
Padding(
  padding: const EdgeInsets.symmetric(horizontal: 3),
  child: TextField(
    decoration: InputDecoration(
      labelText: 'Phone Number',
      border: OutlineInputBorder(
        borderRadius: BorderRadius.circular(10),
      ),
    ),
  ),
),
// Padding
```

TODO Problems Terminal App Quality Insights App Inspection Logcat Services

localhost:65522 Sign in ...

Profile

Name

Location

Phone Number

Email

Settings

Log Out

authentication_screen.dart × pubspec.yaml × login_screen.dart × signup_screen.dart

Android SDK "Android API 29 Platform" is missing

```
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
```

```
listTile(
  title: Text('Settings'),
  leading: Icon(Icons.settings),
  onTap: () {
    // Navigate to settings screen
  },
),
// ListTile
Divider(),
// Log Out
listTile(
  title: Text('Log Out'),
  leading: Icon(Icons.exit_to_app),
  onTap: () {
    // Implement log-out functionality
  },
),
// ListTile
],
),
// Column
),
// Padding
),
// SingleChildScrollView
);
// Scaffold
```

TODO Problems Terminal App Quality Insights App Inspection Logcat Services

localhost:65522 Sign in ...

Profile

Name

Location

Phone Number

Email

Settings

Log Out

view map:

The screenshot displays a Flutter application running on a web browser (localhost:65522). The application's title is "View Map". The UI consists of a map of Tongi, Bangladesh, with several location labels in English and Bengali. A yellow callout bubble with the text "Go to Parking Details" is overlaid on the map. At the top, there is a navigation bar with icons for back, search, and user profile. Below the bar is a search input field with the placeholder "Search Location". The bottom part of the screen shows the Dart code for the "ViewMapScreen" widget.

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

class ViewMapScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('View Map'),
        actions: [
          // User Profile Button
          IconButton(
            icon: Icon(Icons.person),
            onPressed: () {
              // Navigate to the user profile screen
              Navigator.push(
                context,
                MaterialPageRoute(builder: (context) => UserProfileScreen());
              );
            },
          ), // IconButton
        ],
      ),
      body: Container(
        decoration: BoxDecoration(
          image: DecorationImage(
            image: AssetImage('assets/map.jpg'), // Replace with your own image
            fit: BoxFit.cover,
          ),
        ),
        child: Padding(
          padding: const EdgeInsets.all(16.0),
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              Text(
                'Search Location',
                style: TextStyle(
                  fontSize: 20,
                  fontWeight: FontWeight.bold,
                  color: Colors.white,
                ),
              ), // TextStyle
            ],
          ),
        ),
      ),
    );
  }
}
```

```

55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84

```

```

46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67

```

parking details:

The image shows a Flutter application interface and its corresponding code editor. The application screen displays 'Parking Details' with the following information:

- Available Space: 40 spots
- Vehicle Type: Car
- Parking Time: 1 hours 0 minutes
- Amount: \$6

A yellow button at the bottom right says 'Confirm Booking & Payment'.

The code editor on the left shows the Dart code for the 'UserProfileScreen' widget. The first screenshot shows the initial state with a yellow box around the 'CircleAvatar' widget. The second screenshot shows the code after adding a 'SizedBox' with height 20 to the 'GestureDetector' container.

```
authentication_screen.dart pubspec.yaml login_screen.dart signup_screen.dart
```

Android SDK "Android API 29 Platform" is missing

```
1 import 'package:flutter/material.dart';
2
3 class UserProfileScreen extends StatelessWidget {
4   @override
5   Widget build(BuildContext context) {
6     return Scaffold(
7       appBar: AppBar(
8         title: Text('Profile'),
9       ), // AppBar
10      body: SingleChildScrollView(
11        child: Padding(
12          padding: const EdgeInsets.all(20.0),
13          child: Column(
14            crossAxisAlignment: CrossAxisAlignment.center,
15            children: [
16              // User Profile Picture
17              GestureDetector(
18                onTap: () {
19                  // Implement a method to choose or capture a profile picture
20                },
21                child: CircleAvatar(
22                  radius: 70,
23                ),
24              ),
25            ],
26          ),
27        ),
28      ),
29    );
30  }
31
32  // User Name
33  Padding(
34    padding: const EdgeInsets.symmetric(horizontal: 30),
35    child: TextField(
36      decoration: InputDecoration(
37        labelText: 'Name',
38        border: OutlineInputBorder(
39          borderRadius: BorderRadius.circular(10),
40        ),
41      ),
42    ),
43  ),
44}
```

Available Space: 40 spots

Vehicle Type: Car

Parking Time: 1 hours 0 minutes

Amount: \$6

Confirm Booking & Payment

```
authentication_screen.dart pubspec.yaml login_screen.dart signup_screen.dart
```

Android SDK "Android API 29 Platform" is missing

```
22
23   ),
24   ),
25   ),
26   ),
27   ),
28   ),
29   ),
30   ),
31   ),
32   ),
33   ),
34   ),
35   ),
36   ),
37   ),
38   ),
39   ),
40   ),
41   ),
42   ),
43   ),
```

Available Space: 40 spots

Vehicle Type: Car

Parking Time: 1 hours 0 minutes

Amount: \$6

Confirm Booking & Payment

The screenshot shows a Flutter application interface. At the top, there is a navigation bar with icons for back, forward, and search, followed by the text "Sign in". Below the navigation bar is a header titled "Parking Details" with a back arrow icon. The main content area displays the following information:

- Available Space: 40 spots
- Vehicle Type: Car (dropdown menu)
- Parking Time: 1 hours 0 minutes (dropdown menu)
- Amount: \$6

At the bottom right is a large yellow button labeled "Confirm Booking & Payment".

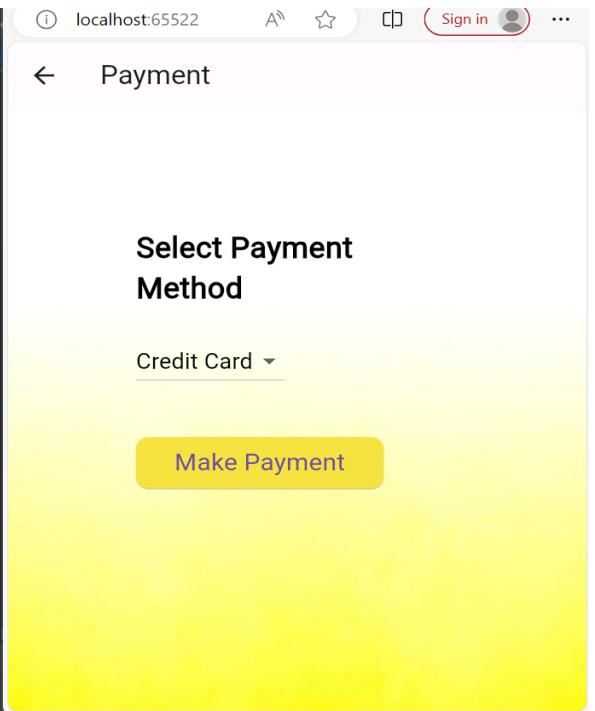
On the left side of the screen, there is a code editor window showing Dart code for a screen named "authentication_screen.dart". The code includes sections for "User Location" and "User Phone Number" fields, each with its own padding, text field, and decoration.

This screenshot is similar to the one above, showing the same "Parking Details" screen with available space, vehicle type, parking time, and amount.

On the left, the code editor window for "authentication_screen.dart" shows a different part of the code. It includes a list view with two items: "Settings" and "Log Out". Each item is represented by a ListTile with a leading icon and an onTap callback. A Divider is placed between the two items.

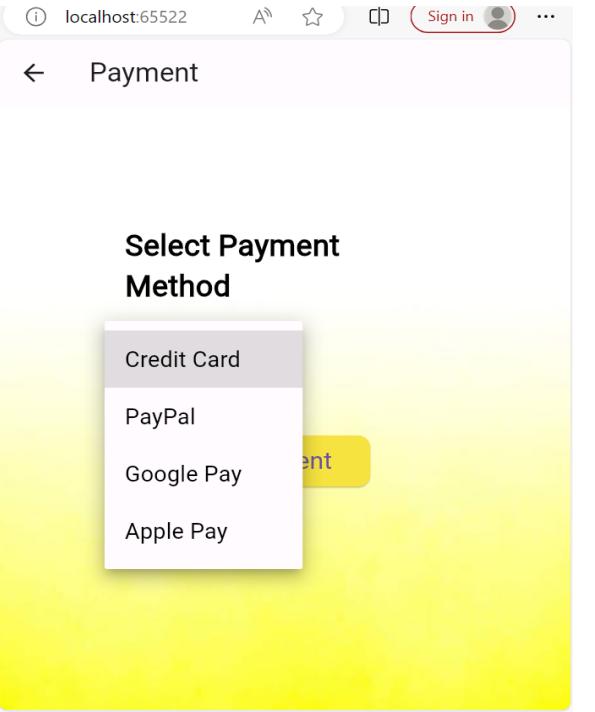
payment Method:

```
1 import 'package:flutter/material.dart';
2
3 class PaymentScreen extends StatefulWidget {
4   @override
5   _PaymentScreenState createState() => _PaymentScreenState();
6 }
7
8 class _PaymentScreenState extends State<PaymentScreen> {
9   String selectedPaymentMethod = 'Credit Card';
10
11   @override
12   Widget build(BuildContext context) {
13     return Scaffold(
14       appBar: AppBar(
15         title: Text('Payment'),
16       ), // AppBar
17       body: SingleChildScrollView(
18         child: Container(
19           height: MediaQuery.of(context).size.height,
20           decoration: BoxDecoration(
21             image: DecorationImage(
22               image: AssetImage('assets/yyy.jpg'), // Replace with
```



The screenshot shows the Android Studio interface with several tabs at the top: 'login_screen.dart', 'signup_screen.dart', 'view_map_screen.dart', and 'user_profile_screen'. The main area displays Dart code for a UI component. A vertical navigation bar on the left side of the code editor highlights specific lines of code, starting from line 22 and ending at line 43. The highlighted code is as follows:

```
22           image: AssetImage('assets/yyy.jpg'), // Replace with
23           fit: BoxFit.cover,
24       ),
25   ),
26   padding: const EdgeInsets.all(100.0),
27   child: Column(
28     crossAxisAlignment: CrossAxisAlignment.start,
29     children: [
30       Text(
31         'Select Payment Method',
32         style: TextStyle(fontSize: 24.0, fontWeight: Font
33       ),
34       const SizedBox(height: 20),
35       DropdownButton<String>(
36         value: selectedPaymentMethod,
37         onChanged: (String? newValue) {
38           setState(() {
39             selectedPaymentMethod = newValue!;
40           });
41         },
42         items: ['Credit Card', 'PayPal', 'Google Pay', 'A
43       
```



The screenshot shows a mobile application interface. At the top, there is a navigation bar with icons for back, forward, and search, along with a "Sign in" button. Below the navigation bar is a header with the text "Payment". A large, semi-transparent modal dialog is displayed in the center. The dialog has a title "Payment & Booking Successful" and a message "Your payment has been successfully processed and booking confirmed." At the bottom right of the dialog is a blue "OK" button. The background of the app shows a blurred "Select Payment Method" screen.

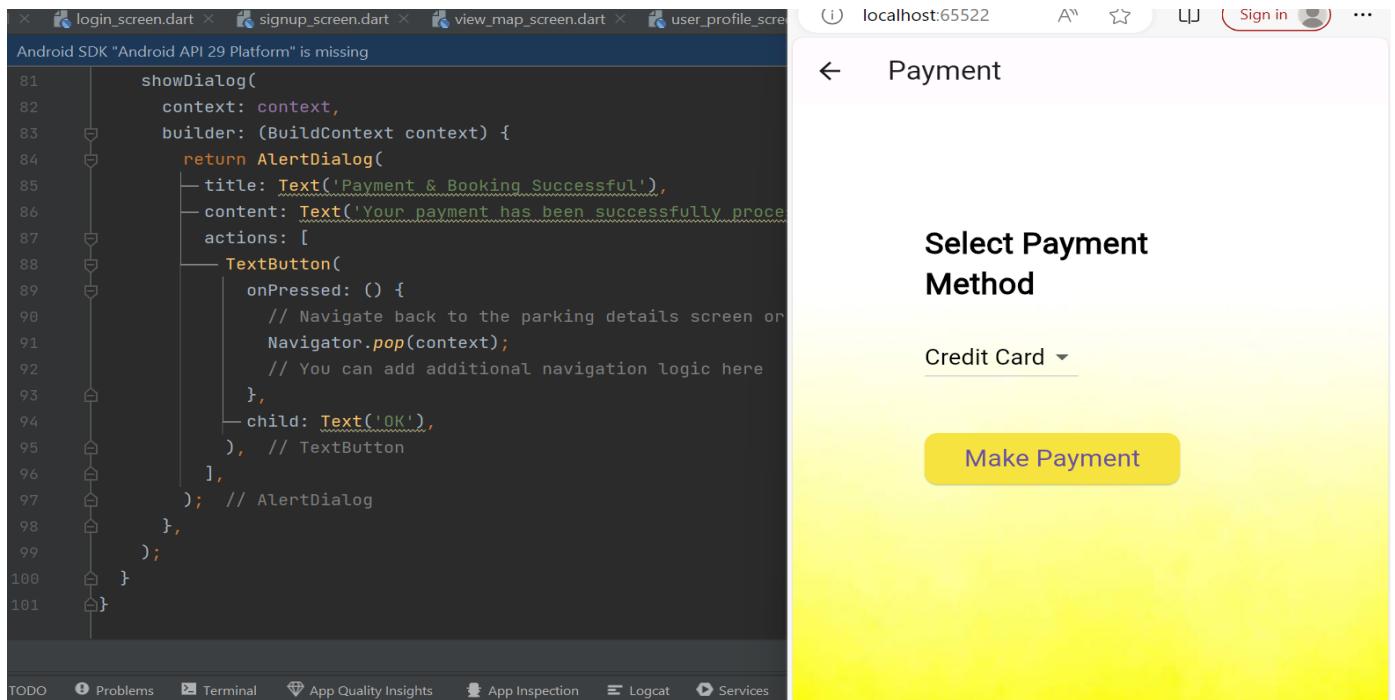
```
43     .map<DropdownMenuItem<String>>((String value) {
44       return DropdownMenuItem<String>(
45         value: value,
46         child: Text(
47           value,
48           style: TextStyle(fontSize: 18.0, color: Col
49         ), // Text
50       ); // DropdownMenuItem
51     }).toList(),
52   ), // DropdownButton
53   const SizedBox(height: 40),
54   ElevatedButton(
55     onPressed: () {
56       // Perform payment logic here
57       // You can integrate payment gateways or handle
58       // For demonstration purposes, let's assume pay
59       _showConfirmationDialog();
60     },
61     style: ElevatedButton.styleFrom(
62       padding: EdgeInsets.symmetric(vertical: 15, hor
63       shape: RoundedRectangleBorder(
64         borderRadius: BorderRadius.circular(10.0),

```

The screenshot shows a mobile application interface. At the top, there is a navigation bar with icons for back, forward, and search, along with a "Sign in" button. Below the navigation bar is a header with the text "Payment". A large, semi-transparent modal dialog is displayed in the center. The dialog has a title "Select Payment Method" and a dropdown menu labeled "Credit Card". Below the dropdown is a yellow "Make Payment" button. The background of the app shows a blurred "Payment & Booking Successful" screen.

```
65     ), // RoundedRectangleBorder
66     primary: Colors.yellow, // Button color
67   ),
68   child: Text(
69     'Make Payment',
70     style: TextStyle(fontSize: 20.0),
71   ), // Text
72   ],
73   ), // Column
74   ), // Container
75   ), // SingleChildScrollView
76   ); // Scaffold
77 }
78 }

80 void _showConfirmationDialog() {
81   showDialog(
82     context: context,
83     builder: (BuildContext context) {
84       return AlertDialog(
85         title: Text('Payment & Booking Successful'),
86         content: Text('Your payment has been successfully proce
```



Dependences for image setting:

```

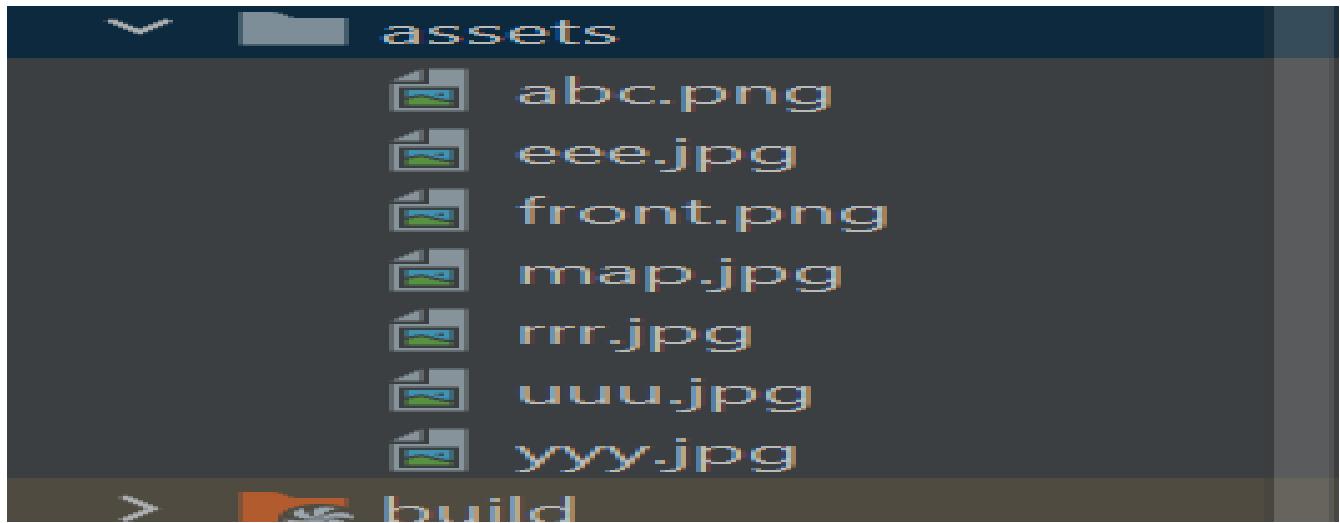
# the material icons class.
uses-material-design: true

# To add assets to your application, add an assets section, like this:
assets:
  - assets/
#   - images/a_dot_ham.jpeg

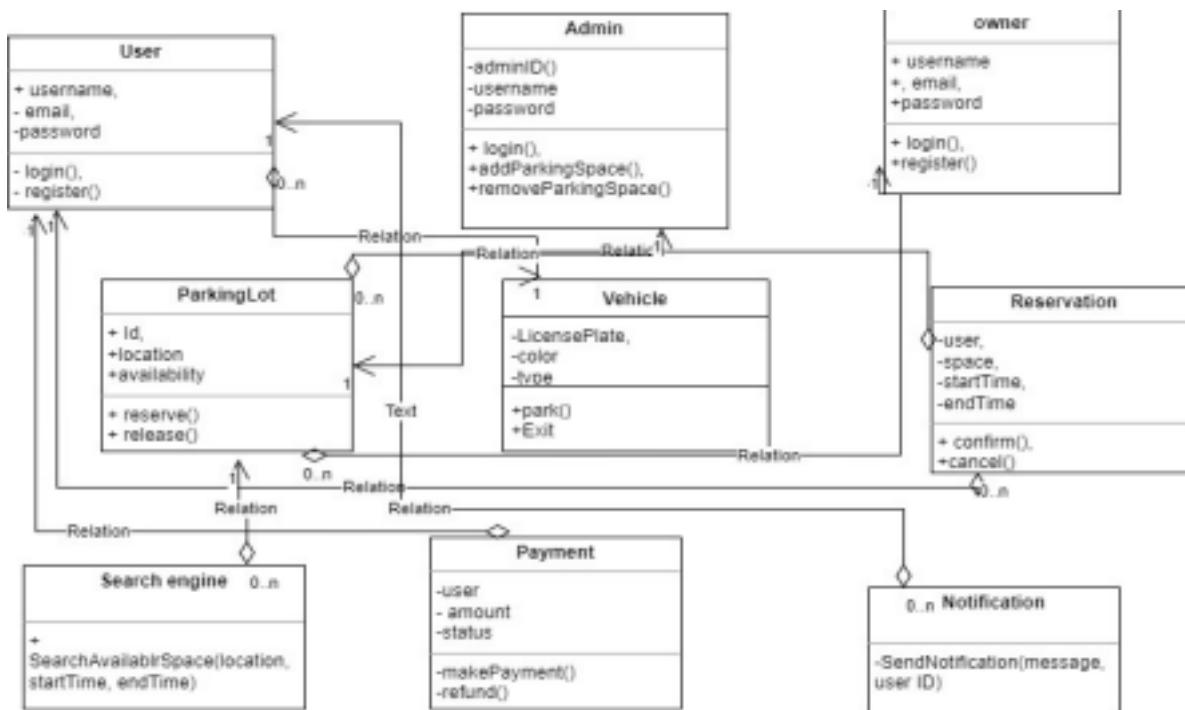
# An image asset can refer to one or more resolution-specific "variants", see
# https://flutter.dev/assets-and-images/#resolution-aware

```

Assets:



Class Diagram here:



Use Case Diagram:



App Demo

Start (LogIn) :

When the app launches, the login page first appears. Users must enter their email address and password to log in. After logging in, the user is directed to the home page.



Login

Email

Password

[Login Here](#)

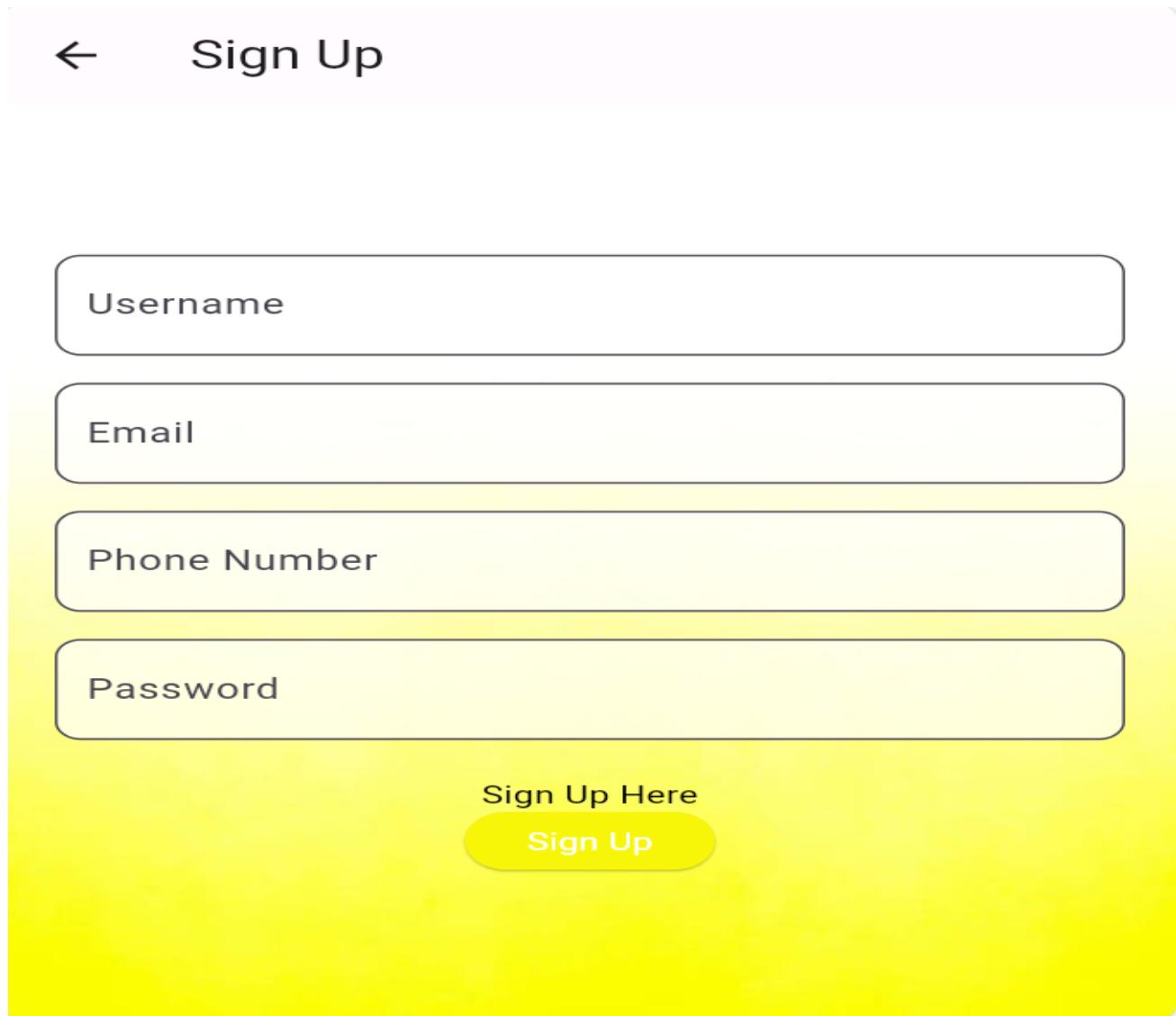
[Login](#)

[Forgot Password?](#)

Sign up :

page for new users to register. Users need to provide an email address, a password, and a second password confirmation in order to create an account. When users click the signup button, their account is created after they have filled out the "Email Address," "Password," and "Confirm

"Password" fields. The signup button sends the user back to the login page. When you've logged in, clicking the user's menu page



The image shows a mobile-style sign-up form. At the top left is a back arrow icon. To its right, the word "Sign Up" is displayed in a large, bold, black font. Below this are four input fields, each enclosed in a rounded rectangle with a thin gray border. The first field is labeled "Username". The second field is labeled "Email". The third field is labeled "Phone Number". The fourth field is labeled "Password". Below these fields is a yellow rectangular area containing two buttons. The top button is labeled "Sign Up Here" in black text. The bottom button is a yellow oval with the words "Sign Up" in white.

← Sign Up

Username

Email

Phone Number

Password

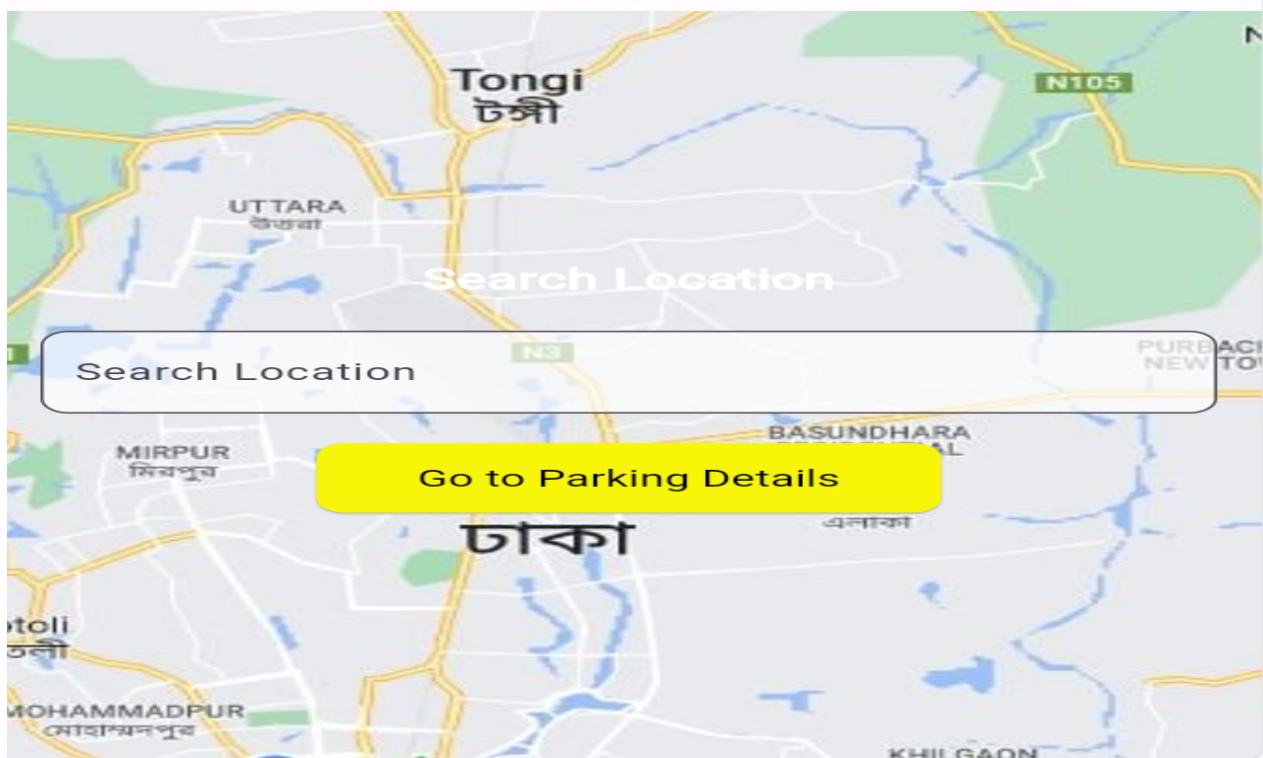
Sign Up Here

Sign Up

View Map:



View Map



View parking details:

Parking Details

Available Space: 40 spots

Vehicle Type:

Car

Parking Time:

1 hours ▾ 0 minutes ▾

Amount: \$6

Confirm Booking & Payment

Payment method:



Payment

Select Payment Method

Credit Card

PayPal

Google Pay

Apple Pay



Payment

Select Payment

Payment & Booking Successful

Your payment has been successfully processed
and booking confirmed.

OK