Smart parking

Phase 4:

Set Up Flutter:

Install Flutter on your development environment.

Set up your favorite code editor or IDE with Flutter support.

Create a New Flutter Project:

Use the Flutter CLI to create a new project.

Design the User Interface:

Design the app interface with widgets to display real-time parking availability. Consider using maps or lists to show parking spots

Establish a Connection:

Use Python libraries, like socket or HTTP requests, in your Flutter app to establish a connection with the Raspberry Pi. Ensure the Raspberry Pi is running a server to provide real-time parking data.

Receive Data:

Write code in Flutter to receive parking availability data from the Raspberry Pi. You can use JSON or other data formats to transmit this information.

Display Data:

Update the UI based on the received data to show parking availability. You can use widgets like Text, List Views, or custom markers on a map, depending on your app's design.

Real-Time Updates:

Implement a mechanism for periodic data updates. You can use timers or push notifications to keep the data up-to-date

Error Handling:

Add error handling and provide feedback to users in case of connectivity issues or other errors.

Testing:

Thoroughly test the app to ensure it displays real-time parking availability accurately.

Deployment:

Once the app is ready, you can deploy it to the desired mobile platforms (iOS and Android).

User Documentation:

Create user documentation or onboarding screens to guide users on how to use the app effectively.

Optimize and Refine:

Continuously refine and optimize the app based on user feedback and any issues that arise.

Remember that the Raspberry Pi should have a way to provide real-time parking data, which might involve sensors, cameras, or other hardware components, depending on your project's specifics. Additionally, security considerations are essential when connecting your mobile app to the Raspberry Pi, especially if it's running as a server.

EXAMPLE Program for creating a mobile application for smart parking

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

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

import 'dart:convert';


void main() {

  runApp(MyApp());

}


class MyApp extends StatefulWidget {

  @override

  _MyAppState createState() => _MyAppState();

}


class _MyAppState extends State<MyApp> {

  String parkingAvailability = 'Loading...';


  Future<void> fetchParkingAvailability() async {

    final response = await http.get(Uri.parse('https://your-raspberrypi-url/parking-
data'));


    if (response.statusCode == 200) {

      final data = json.decode(response.body);

      setState(() {

        parkingAvailability = data['availability'];
      });
    } else {
      setState(() {
        parkingAvailability = 'Error fetching data';
      });
    }
  }

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

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: Text('Smart Parking App'),
        ),
        body: Center(
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: <Widget>[
              Text(
                'Parking Availability:',
                style: TextStyle(fontSize: 18),
              ),
              Text(
                parkingAvailability,
                style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),
              ),
            ],
          ),
        ),
      ),
    );
  }
}
```

Program 2:

```
import 'package:flutter/material.dart';
import 'package:http/http.dart' as http;
import 'dart:convert';

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

class ParkingApp extends StatefulWidget {
  @override
  _ParkingAppState createState() => _ParkingAppState();
}

class _ParkingAppState extends State<ParkingApp> {
  String parkingAvailability = 'Loading...';

  Future<void> fetchParkingAvailability() async {
    final response = await http.get(Uri.parse('https://your-raspberrypi-url/parking-
data'));

    if (response.statusCode == 200) {
      final data = json.decode(response.body);
      setState(() {
        parkingAvailability = data['availability'];
      });
    } else {
      setState(() {
        parkingAvailability = 'Error fetching data';
      });
    }
  }

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

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: Text('Smart Parking App'),
        ),
        body: Center(
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: <Widget>[
              Text(
                'Parking Availability:',
                style: TextStyle(fontSize: 18),
              ),
              Text(
                parkingAvailability,
                style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),
              ),
            ],
          ),
        ),
      ),
    );
  }
}
```