Team Members:

1. R.RESHMA MASHUTHA - 921021104040

2. C.BHARATHI - 921021104003

3. B.KIRUTHIKA - 921021104019

4. P.RAVINA - 921021104038

Smart parking using IoT

Smart parking:

● Smart parking is a system that uses the Internet of Things (IoT) to collect and analyse data about parking availability and occupancy. This data can then be used to improve the efficiency and convenience of parking for both drivers and parking lot operators.

● One of the key benefits of smart parking is that it can help drivers to find parking spaces more quickly and easily. This is done by providing drivers with real-time information about parking availability in nearby parking lots. Drivers can access this information through a mobile app or website, or through signs that are posted in the parking lot itself.

To build a mobile app for real-time parking availability using Python and Flutter, we can follow these steps:

1. Set up the development environment. Install Flutter and Python on our machine.

2. Create a new Flutter project. Use the following command to create a new Flutter project:

CODE:

```
flutter create smart_parking
```

3. Add the necessary dependencies. Add the following dependencies to the pubspec.yaml file:

CODE:

```
dependencies:
flutter:
sdk: flutter
mqtt_client: ^11.0.0
```

4.Create a new class to handle the MQTT connection. This class will be responsible for connecting to the Raspberry Pi and receiving parking availability data.

CODE:

```
class MqttClient {
  final client = mqtt.Client();

  Future<void> connect() async {
    await client.connect('localhost', 1883);
    await client.subscribe('parking/availability');
  }

  void onMessage(String topic, String message) {
    // Handle the received message here.
  }

  Future<void> disconnect() async {
    await client.disconnect();
  }
}
```

5.Create a new class to display the parking availability data. This class will be responsible for receiving parking availability data from the MQTT client and displaying it to the user.

CODE:

```
class ParkingAvailabilityDisplay {
  final MqttClient mqttClient;
  ParkingAvailabilityDisplay(this.mqttClient);

  void onMessage(String topic, String message) {
    // Update the UI with the new parking availability data.
  }
}
```

6.Update the main.dart file to connect to the MQTT client and start displaying the parking availability data.

CODE:

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

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

class MyApp extends StatefulWidget {
  @override
  State<MyApp> createState() => _MyAppState();
}

class _MyAppState extends State<MyApp> {
  final mqttClient = MqttClient();
  final parkingAvailabilityDisplay = ParkingAvailabilityDisplay(mqttClient);

  @override
  void initState() {
    super.initState();
    mqttClient.connect();
                parkingAvailabilityDisplay.onMessage((topic,
message) {
        // Update the UI with the new parking availability
data.
    });
  }

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: Text('Smart Parking'),
        ),
        body: Center(
          child: ParkingAvailabilityDisplay(),
        ),
      ),
    );
  }
}
```

7.Run the app. To run the app, use the following command

CODE:

```
flutter run
```

8.Sample output:

```
Smart Parking

Parking Availability:

Lot A: 10/10
Lot B: 8/10
Lot C: 5/10
```

Dependencies:

- Flutter SDK
- Python 3
- PySerial
- Requests

Testing and deploying the app:

- To test the app, we can run it on a mobile device or in a simulator. To deploy the app, we can use the Flutter SDK to build an APK or IPA file.

Additional features:

Once we have a basic working app, we can add additional features, such as:

- The ability to filter the parking availability data by location or type of parking space.

- The ability to receive notifications when a parking space becomes available.
- The ability to pay for parking directly from the app.

Conclusion:

- This project demonstrates how to develop a mobile app using Python and Flutter to display real-time parking availability. The app can be used to help drivers find available parking spaces more easily.