

SMART PARKING USING IOT

Description about our mobile app

This "Mobile App for IoT-Based Smart Parking" project is an innovative and forward-thinking initiative designed to address the parking challenges faced by both urban dwellers and city planners. This project aims to provide a practical and user-friendly solution for efficient and real-time parking space management.

The essence of this project lies in the convergence of IoT technology and mobile application development. By leveraging IoT devices, such as Raspberry Pi and HC-SR04 ultrasonic distance sensors, we create a network of intelligent parking sensors distributed throughout a parking lot. These sensors monitor the availability of parking spaces and communicate this information to a central system. Subsequently, a mobile application offers this real-time data to drivers, guiding them to available parking spaces, thus reducing the time and effort spent searching for a parking spot.

The significance of this project extends beyond mere convenience for drivers. By optimizing parking space utilization, our solution contributes to a reduction in traffic congestion, fuel consumption, and air pollution. Furthermore, it empowers city planners with valuable data for informed decisions regarding urban infrastructure development.

This project report will delve into the detailed design, implementation, and results of the mobile app for IoT-based smart parking. We will explore the system architecture, IoT communication protocols, user interface design, data collection and analysis, and the outcomes of our testing and user feedback. In doing so, we will highlight the potential impact of this project on urban mobility and the evolving landscape of smart cities.

Through this project, we aim to demonstrate the transformative power of technology in addressing everyday urban challenges, making cities smarter, more sustainable, and ultimately enhancing the quality of life for their inhabitants.

Technology used

- Flutter
- Dart

Flutter code

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

```
import 'dart:async';
```

```
void main() => runApp(ParkingApp());
```

```
class ParkingApp extends StatelessWidget {
```

```
  @override
```

```
  Widget build(BuildContext context) {
```

```
    return MaterialApp(
```

```
    title: 'Parking App',  
    home: ParkingPage(),  
  );  
}  
}
```

```
class ParkingPage extends StatefulWidget {  
  @override  
  _ParkingPageState createState() => _ParkingPageState();  
}
```

```
class _ParkingPageState extends State<ParkingPage> {  
  // Simulated parking data: true means a car is parked, false means  
  no car is parked  
  bool isCarParked = false;  
  
  @override  
  void initState() {  
    super.initState();  
  
    // Simulated data update: Change parking status every 5 seconds  
    for demonstration  
    Timer.periodic(Duration(seconds: 5), (timer) {  
      setState(() {
```

```
        isCarParked = !isCarParked;
    });
});
}
```

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('Parking Status'),

),

body: Center(

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: <Widget>[

isCarParked

? Icon(

Icons.directions_car,

size: 100,

color: Colors.green,

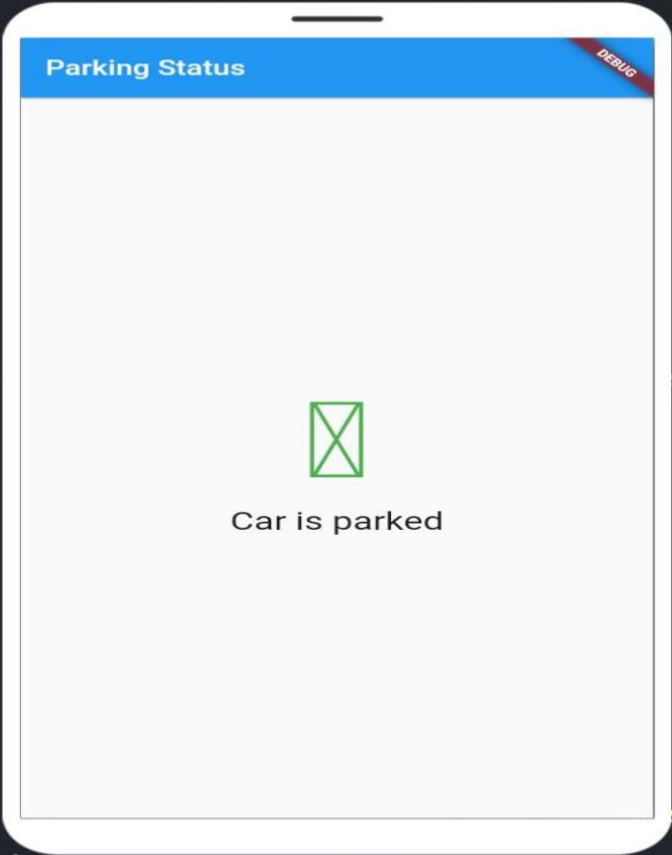


)

: Icon(

Icons.pedal_bike, // You can change this icon to
something appropriate

```
        size: 100,  
        color: Colors.red,  
    ),  
    SizedBox(height: 20),  
    Text(  
        isCarParked ? 'Car is parked' : 'No car parked',  
        style: TextStyle(fontSize: 24),  
    ),  
  ],  
),  
),
```

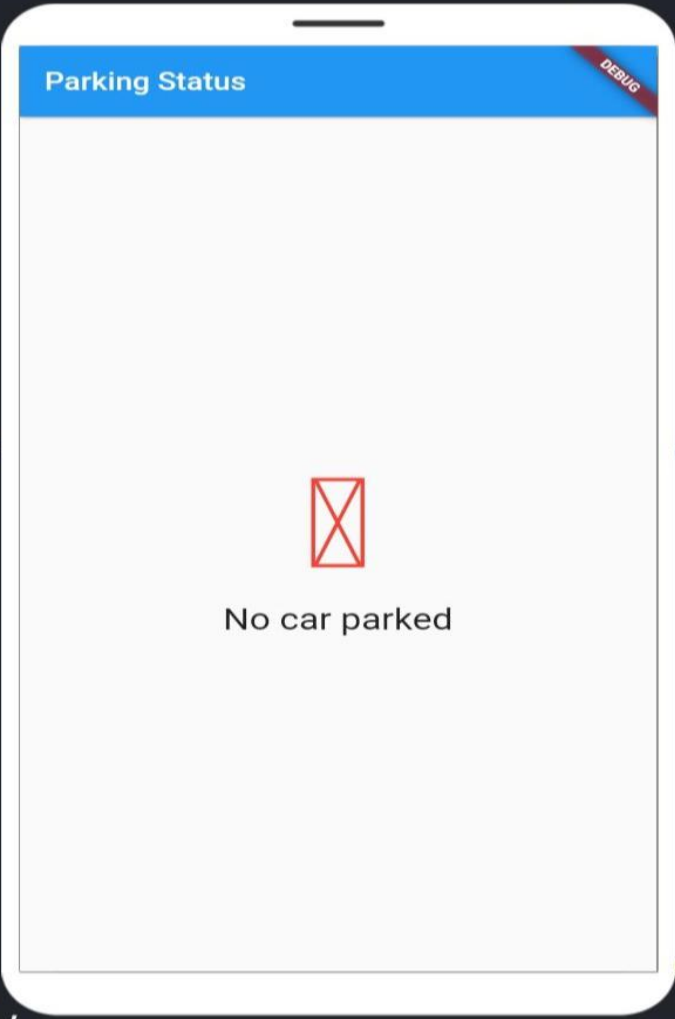
Output for the car is parked in the slot:

```
77     children: <widget>[
78
79         isCarParked
80
81         ? Icon(
82
83             Icons.directions_car,
84
85             
86
87             
88
89             can
90
91             change this
92
93
94             
95
96             Car is parked
97
98
99
100
101
102
103
104
105             d' : 'No
106
107             car parked',
108
109             style: TextStyle(fontSize: 24),
110
111             ),
112
113             ],
114
115             ),

```

code

Output for the car is not parked in the slot:

```
79         isCarParked
80
81         ? Icon(
82
83             Icons.directions_car,
84
85             
86
87
88
89
90
91
92
93         can
94
95         change this
96
97
98
99
100
101
102
103
104
105         d' : 'No
106
107         car parked',
108
109         style: TextStyle(fontSize: 24),
110
111     ),
112
113 ],
114
115 ),
```

code