SMART PARKING 🚗

Name : Surya

Department : cse

Introduction

This section will provide an overview of the purpose of the document, which is to guide the development of a mobile app for displaying realtime parking availability. It will explain the importance of such an app and its potential benefits to both drivers and parking operators

Choosing a Mobile App Development Framework

- "The right choice of technology can make or break a mobile app development project"

In this section, we will discuss the different mobile app development frameworks available and explore why Flutter is a great choice for developing the parking availability app. We will also discuss other frameworks that you could consider based on your specific needs.

Using Python for Mobile App Development

Python offers many benefits for mobile app development, including its scalability, readability, and easy syntax. It also provides numerous libraries and frameworks that make app development faster and more efficient.

In this section, we will explore why Python is a great language for creating mobile apps and how it can help you create an app that meets your unique requirements.

Step 1: Set up Raspberry Pi

Get a Raspberry Pi board (e.g., Raspberry Pi 3 or 4) and ensure it's running a suitable operating system (Raspberry Pi OS or Raspbian).

Connect to your Raspberry Pi using SSH or VNC, or you can use its terminal directly if you have a keyboard and monitor attached.

Step 2: Install Python and Required Libraries

You'll need Python and some libraries to create a basic web server.

```
sudo apt update
```

```
sudo apt upgrade
```

```
sudo apt install python3 python3-pip
```

```
pip3 install flask
```

Step 3: Create a Python Script

Create a Python script for your Raspberry Pi that provides parking availability data. You can use Flask to create a simple web API. Here's an example script:

```
From flask import Flask, jsonify

App = Flask(__name__)


# Dummy parking data (replace with your actual data source)

Parking_data = {

  "parking_lot_1": {

    "total_spots": 50,

    "available_spots": 20

  },

  "parking_lot_2": {
```

```
        “total_spots”: 30,
        “available_spots”: 10
    }
}

@app.route(‘/parking_data’, methods=[‘GET’])
Def get_parking_data():
    Return jsonify(parking_data)

If __name__ == ‘__main__’:
    App.run(host=’0.0.0.0’, port=5000)
```

Save this script as parking_server.py on your Raspberry Pi.

Step 4: Run the Python Script

Run the script on your Raspberry Pi:

```
python3 parking_server.py
```

This will start a web server that provides parking availability data at http://raspberry_pi_ip:5000/parking_data

Step 5: Ensure Connectivity

Ensure your Raspberry Pi is on the same network as your mobile app, and the IP address is accessible. You may need to configure your router or firewall settings to allow incoming connections.

Step 6: Mobile App Integration

Now, you can integrate your Flutter app with the Raspberry Pi server by making HTTP requests to the Raspberry Pi's IP address and the /parking_data endpoint. Use the code example provided earlier in the Flutter app section for this purpose.

Please note that this is a basic example, and in a real-world scenario, you would replace the dummy parking data with data from sensors or other sources. Additionally, consider security measures, data validation, and error handling for a production system.

Realtime Parking Availability

In this section, we will discuss how to display parking availability data via the mobile app. We will cover the hardware and software requirements for receiving and displaying realtime data, and explain how to efficiently communicate with the Raspberry Pi to get the required data.

Displaying Parking Availability Data

Location	Available Spaces	Total Space
A.	8	10
B.	5	20
C.	12	15

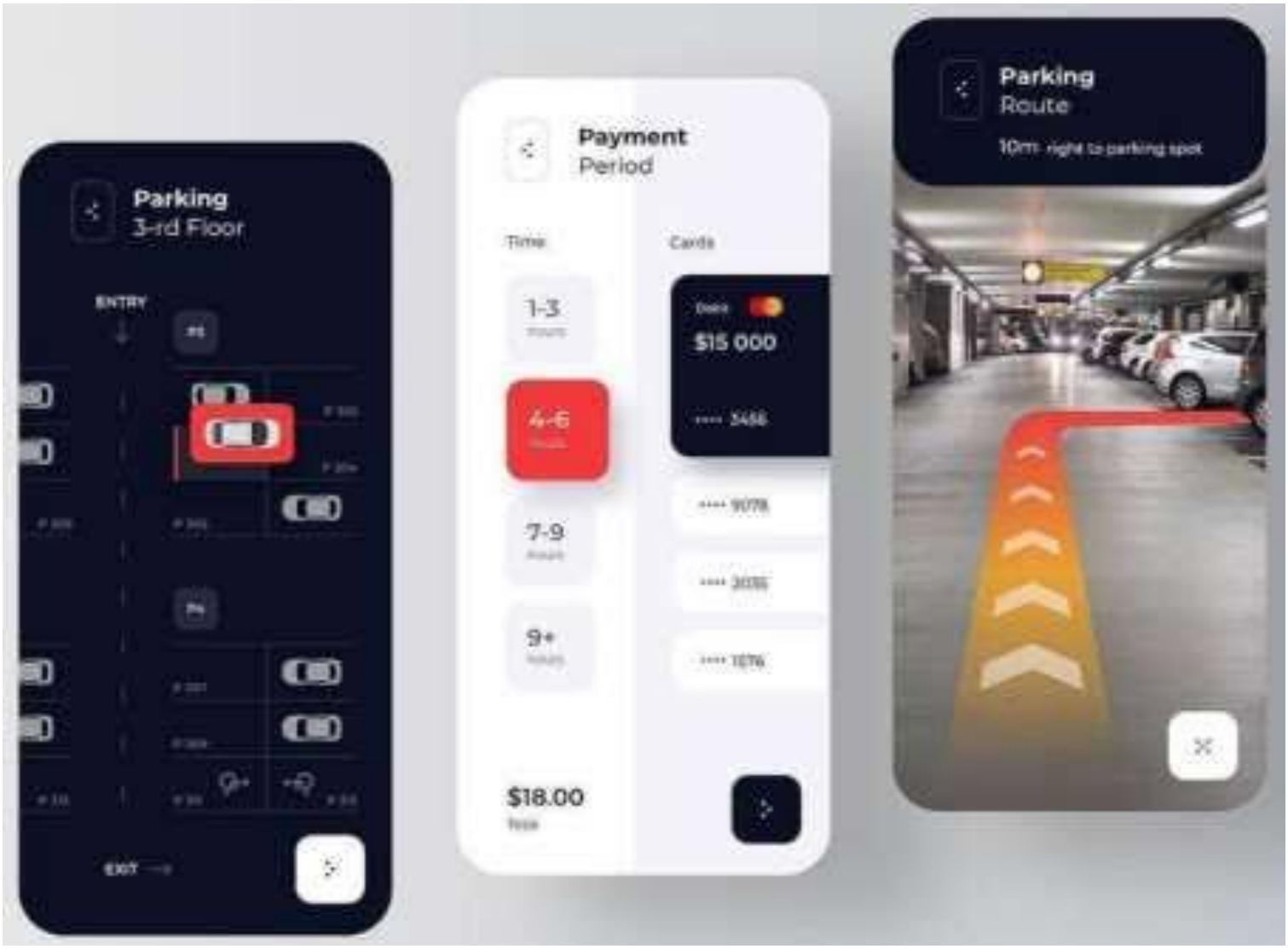
This section will describe how to display the parking availability data in a user-friendly way on the mobile app. We will explain what type of visualizations can prove to be most efficient and how to implement them on the app Interface.

Receiving Parking Availability Data from Raspberry Pi

- "Raspberry Pi is a small computer that can work wonders"

In this section, we will discuss the hardware and software requirements for receiving parking availability data on the mobile app. We will explain how to communicate with the Raspberry Pi and receive the required data efficiently, and how to handle any technical hitches that might arise.

App Functions and Design



Designing App Functions for Parking Availability

This section will describe how to design an app interface that is user-friendly, intuitive and efficient. We will go over the different functions required for the app and how to implement them in the app interface.

Displaying Realtime Parking Availability

In this section, we will describe how to show the available parking spaces in real-time on the mobile app. We will go over the different communication techniques needed to establish a connection between Raspberry Pi and mobile app.

Conclusion

This section will summarize the content of the document and reiterate the importance of creating a mobile app to display realtime parking availability. It will provide a brief overview of the key benefits and characteristics of the app, and provide some recommendations for further reading.