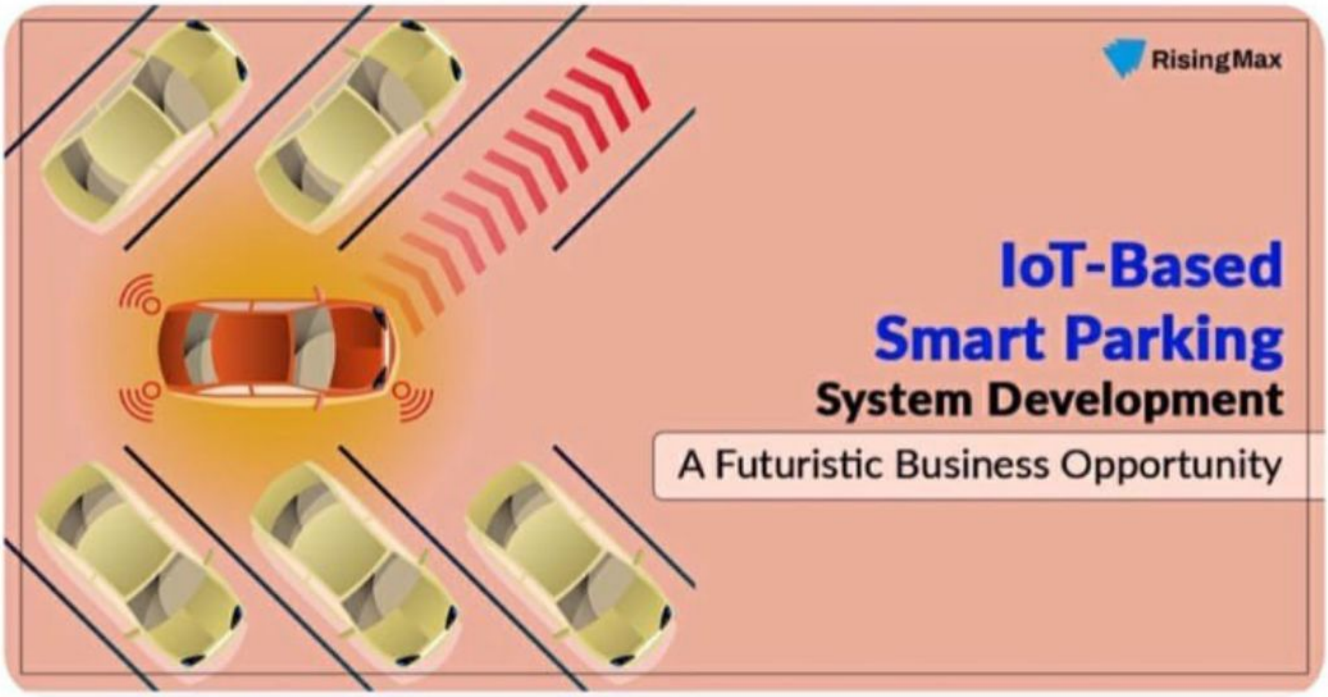
# □ PHASE-3 DEVELOPMENT PART

## □ <°°°*smart parking*°°°>

- In this technology project you will begin building your project by deploying IoT devices and then

developing a Python script on the IoT devices as per the project requirement. After performing the relevant activities create a document around it and share the same for assessment.



□ **Building a smart parking system using IoT devices involves several steps:**

1. **Define Objectives and Requirements:**

   - Clearly outline the goals and functionalities you want in the smart parking system.

2. **Select IoT Devices:**

   - Choose appropriate IoT devices like sensors, cameras, and communication modules that will be used to monitor parking spaces.

3. **Install Sensors and Devices:**

   - Install IoT sensors in parking spots to detect occupancy, and cameras for visual monitoring.

4. **Connect IoT Devices:**

   - Set up a communication network (e.g., Wi-Fi, LoRa, or cellular) to connect the IoT devices and allow them to transmit data.

5. **Develop IoT Software:**

   - Create software to process data from the sensors and cameras, enabling real-time monitoring and analysis of parking occupancy.

6. **Integrate Data Processing and Storage:**

   - Implement a platform to process and store data collected from the IoT devices securely.

7. **Implement Data Analysis and Insights:**

   - Utilize data analytics to derive insights, such as available parking spaces and patterns of usage.

8. **Develop User Interface:**

   - Create a user interface (e.g., a mobile app or a website) for users to check parking availability and make reservations.

9. **Incorporate Payment System (Optional):**

- Integrate a payment system into the user interface to facilitate payment for parking reservations.

10. **Test and Optimize:**

- Conduct thorough testing to ensure the system functions as intended, and make necessary optimizations for efficiency and accuracy.

11. **Deploy and Monitor:**

- Deploy the smart parking system in the intended locations and continually monitor its performance, addressing any issues that arise.

12. **Maintenance and Updates:**

- Regularly maintain the system, apply updates, and consider enhancements based on user feedback

- ``````python coding`````

1. **Install Flask:**

Ensure you have Flask installed by running:

```bash
pip install Flask
```

2. **Python Script (smart_parking.py):**

```python
from flask import Flask, jsonify, request
import random
import threading
import time

app = Flask(__name__)
parking_spots = [0] * 10  # Represents 10 parking spots, initially all vacant
```

```
  def simulate_iot_data():
    global parking_spots
    while True:
      # Simulate IoT data (occupancy: 0 for vacant,
1 for occupied)
      parking_spots = [random.randint(0, 1) for _ in
range(10)]
      time.sleep(5)  # Simulate data update every 5
seconds


  @app.route('/parking', methods=['GET'])
  def get_parking_status():
    global parking_spots
    return jsonify({'parking_status': parking_spots})


  if __name__ == '__main__':
    # Start IoT data simulation in a separate thread
    iot_thread =
threading.Thread(target=simulate_iot_data)
    iot_thread.daemon = True
    iot_thread.start()
```

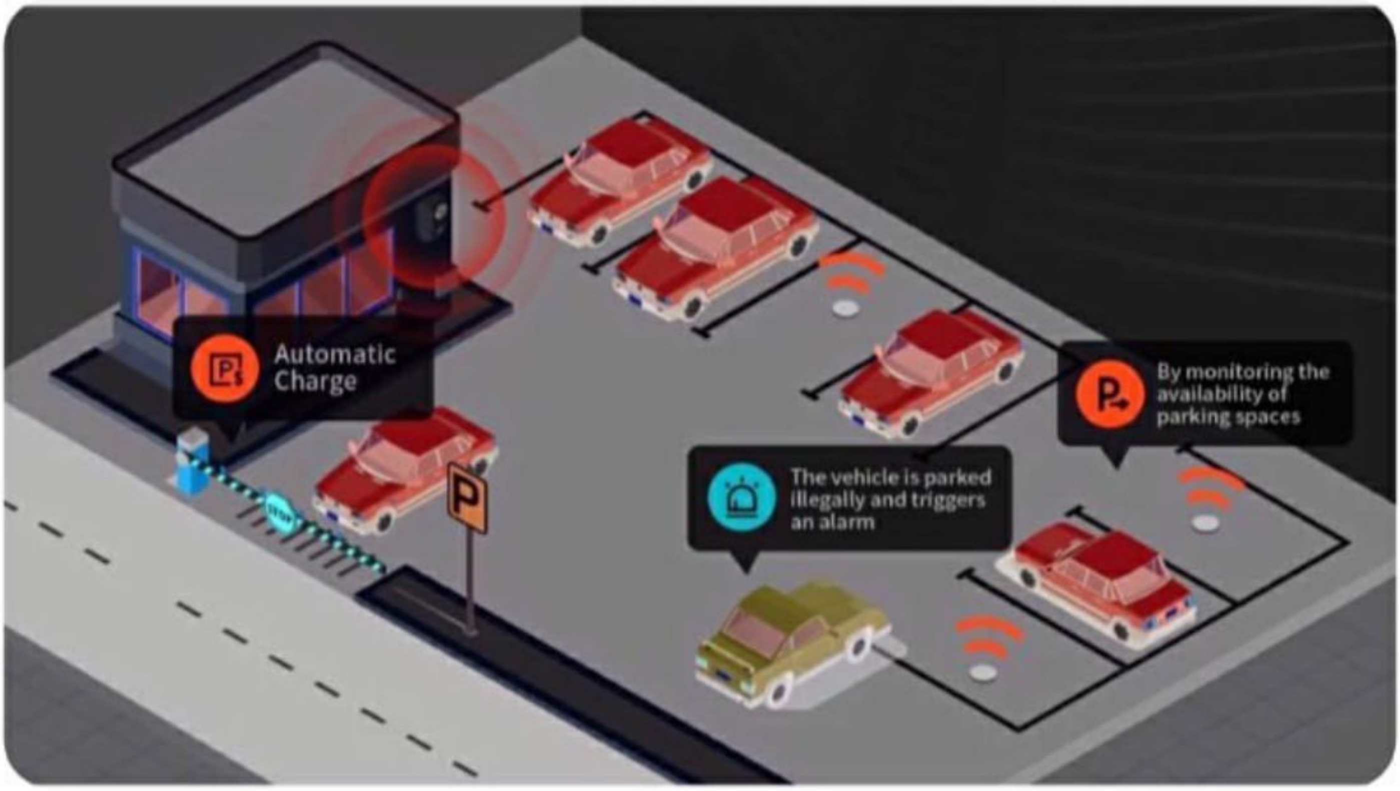
```
# Run the Flask app

app.run(debug=True)
```

3. **Run the Script:**

Run the script using:

```bash
python smart_parking.py
```



Automatic Charge
The vehicle is parked illegally and triggers an alarm
By monitoring the availability of parking spaces
P
STOP