**Smart parking**

**Team Members:**

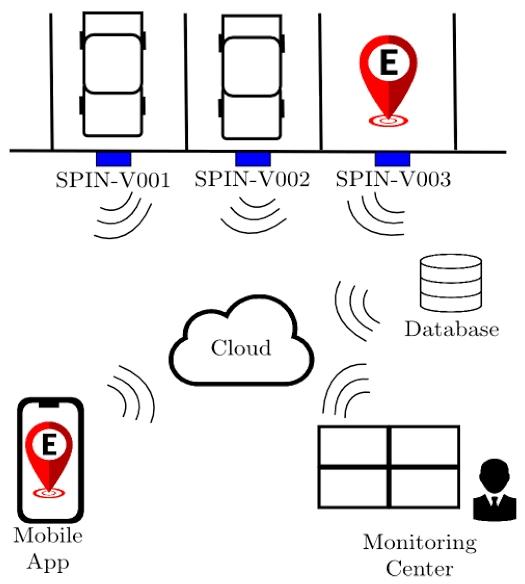
* **R. ARUNKUMAR**
* **C. PRASANNAKUMAR**
* **SHIVAS**
* **V. N. DEVA**
* **R.NARESH**
* **PHASE 2:-**
* **INNOVATION:**

**As of my last knowledge update in September 2021, the latest innovations in smart parking were focused on enhancing efficiency, user experience, and sustainability. While I don't have information on developments beyond that date, I can provide you with some of the potential trends and innovations that were emerging in the smart parking space around that time:**

1. **Autonomous Vehicle Integration:**

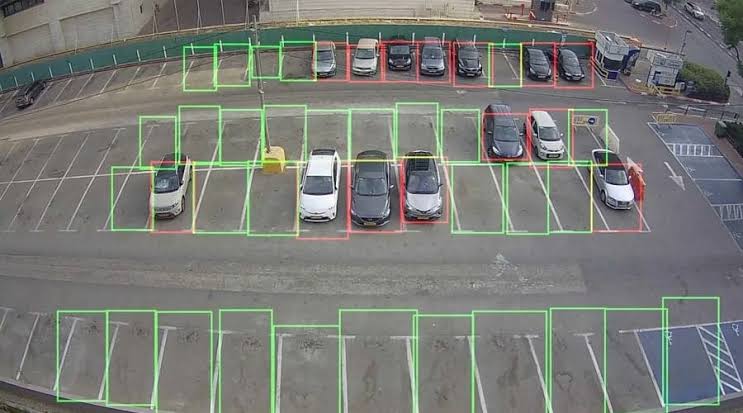
**The integration of autonomous vehicles with smart parking systems was gaining attention. Self-driving cars can park themselves efficiently, reducing the time and space required for parking.**

1. **IoT and Connectivity:**

****

**The Internet of Things (IoT) continued to play a crucial role in smart parking. More devices, such as smart meters and sensors, were being integrated to collect and transmit real-time data, enabling better management and optimization.**

1. **Artificial Intelligence and Machine Learning:**

****

**AI and machine learning were used for predictive analytics, helping drivers find available parking spaces and optimizing traffic flow within parking facilities.**

1. **Contactless Payment:**

**The adoption of contactless payment methods, including mobile wallets and apps, was becoming increasingly popular, making payments for parking more convenient and touch-free.**

1. **Environmental Initiatives:**

**Sustainability remained a key focus. Smart parking systems were incorporating features like EV charging stations, bike-sharing facilities, and eco-friendly construction materials for parking infrastructure.**

1. **Integration with Smart City Initiatives:**

**Smart parking systems were increasingly integrated into broader smart city initiatives, promoting urban sustainability and efficient transportation.**

1. **Data Sharing and Open Platforms:**

**Some cities and companies were moving toward open platforms that allow data sharing between different parking service providers, enhancing interoperability and accessibility for users.**

1. **User-Centric Apps:**

****

**parking apps were evolving to provide users with more features, including reservations, real-time navigation, and personalized recommendations based on user preferences.**

1. **Dynamic Pricing:**

**Some smart parking systems were implementing dynamic pricing models, where parking rates would vary based on demand, time of day, or special events.**

**Please note that the smart parking industry is dynamic, and there may have been further developments and innovations since my last update. I recommend checking the latest news and industry reports for the most up-to-date information on smart parking innovations.**

* **CV PROGRAM:**

**import cv2**

**import numpy as np**

**# Load the video feed or capture device (e.g., camera)**

**cap = cv2.VideoCapture('your\_video\_stream\_url\_or\_device\_index')**

**# Define the region of interest (ROI) where parking spots are located**

**roi = [(x1, y1, x2, y2), (x3, y3, x4, y4)] # Define the coordinates of the parking spots**

**while True:**

**ret, frame = cap.read()**

**if not ret:**

**break**

**# Convert the frame to grayscale**

**gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)**

**# Apply image processing techniques (e.g., edge detection, thresholding) to highlight parking spots**

**# Loop through parking spots in the ROI**

**for spot in roi:**

**x1, y1, x2, y2 = spot**

**roi\_frame = gray[y1:y2, x1:x2]**

**# Apply image processing to detect occupancy or vacancy of the parking spot**

**# If the spot is occupied, draw a red rectangle; otherwise, draw a green one**

**if spot\_occupied:**

**color = (0, 0, 255) # Red**

**else:**

**color = (0, 255, 0) # Green**

**# Draw a rectangle around the parking spot**

**cv2.rectangle(frame, (x1, y1), (x2, y2), color, 2)**

**# Display the processed frame with parking spot information**

**cv2.imshow('Smart Parking System', frame)**

**# Exit the program when the 'q' key is pressed**

**if cv2.waitKey(1) & 0xFF == ord('q'):**

**break**

**# Release the video capture and close the OpenCV window**

**cap.release()**

**cv2.destroyAllWindows()**

**Notes: haarcascade\_car Xml code is add in Github link**

**To use OpenCV (Open Source Computer Vision Library) in Python, you'll first need to install it if you haven't already. You can install OpenCV using pip:**

**pip install opencv-python**

**Once OpenCV is installed, you can use it in your Python code.**