PHASE 1: PROBLEM DEFINITION AND DESIGN THINKING

PROJECT TITLE	**9238 - IOT SMART PARKING**
NAME	**VIGNESH M J**
TEAM ID	**5251**
TEAM NAME	**PROJ_204187_TEAM_2**
COLLEGE CODE - NAME	**9238 - MANGAYARKARASI COLLEGE OF ENGINEEING PARAVAI, MADURAI.**
GROUP	**5**
GITHUB REPOSITORY LINK	**https://github.com/vignesh2214/IOT-NAAN-MUDHALVAN-IOT.git**

IOT SMART PARKING

1. Project Definition

1.1. Project Overview

The project aims to create a Smart Parking system by integrating IoT sensors into parking spaces. These sensors will monitor parking space occupancy and availability, and this real-time data will be made accessible to the public through a mobile app. The primary goal is to enhance the efficiency and quality of public parking services.

1.2. Project Objectives

Real-time Parking Space Monitoring: Implement IoT sensors to continuously monitor parking space occupancy in real-time.

Mobile App Integration: We are planned to develop a user-friendly mobile application that provides real-time parking availability information to users.
Efficient Parking Guidance: This Project Provide users with accurate and timely information to guide them to available parking spaces, reducing time spent searching for parking.

2. Design Thinking

2.1. IoT Sensor Design

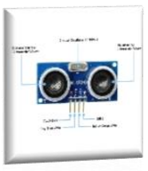
2.1.1. Sensor Selection

IoT sensors for monitoring parking space occupancy. Consider factors such as accuracy, cost, power consumption, and communication capabilities. This are the sensors are required for this Project:

i. **Ultrasonic Sensor**
ii. **Infrared Sensor**
iii. **Micro controller**
iv. **Camera**
v. **Connectivity medium like wireless module.**
vi. **Cloud based data management.**

- **Ultrasonic sensors** are used in this project to detect the presence of vehicles by emitting high-frequency sound waves and measuring their reflections, enabling accurate space occupancy monitoring.

- **Infrared sensors** play a crucial role in smart parking solutions by detecting the presence of vehicles through the emission and reception of infrared radiation, helping manage available parking spaces effectively.



- **Microcontrollers** serve as the brain of smart parking systems, processing sensor data and controlling various components to optimize parking space utilization and facilitate efficient vehicle management.

- **Cameras** are integrated into smart parking setups to capture real-time images or video footage of parking areas, enhancing security and assisting in license plate recognition for automated entry and exit. Cameras are also used for image processing.

- **Wireless modules** provide seamless connectivity in smart parking systems, enabling real-time communication between sensors, controllers, and a central management system for efficient data transmission and remote monitoring.

- **Cloud-based data management** platforms collect and store parking data from various sensors, enabling real-time analytics, mobile apps, and remote management, improving overall parking system efficiency and user experience.

2.1.2. Sensor Deployment

1. **Sensor Placement**: Strategically place sensors in each parking space to maximize coverage and accuracy. We need to ensure sensors are weather-resistant and durable.
2. **Communication**: Set up a wireless communication network LoRa for sensors to transmit data to a central hub.
3. **Power Supply**: Ensure a reliable power source for sensors, through batteries, solar panels.
4. **Data Transmission**: A data transmission protocol for sensors to send occupancy status updates to the central system.
5. **Image Processing:** We can do Image Processing to detect the vehicle using CCTV cameras.

2.2. Real-Time Transit Information Platform

2.2.1. Mobile App Interface

1. **User Interface (UI)**: We are planned to Design an intuitive and user-friendly interface for the mobile app. It Include features such as a map view with parking space markers, search functionality, and user profiles.
2. **Real-time Updates**: We planned to implement a mechanism to continuously update the app with real-time parking availability information.
3. **Navigation**: Integrate navigation functionality to guide users to available parking spaces.
4. **Reservation System**: Consider implementing a reservation system for users to reserve parking spaces in advance.

2.3. Integration Approach Using Raspberry Pi and Cloud

2.3.1. Data Collection with Raspberry Pi

1. **Sensor Selection:** Choose suitable sensors compatible with Raspberry Pi for monitoring parking space occupancy and availability.
2. **Raspberry Pi Selection:** Select a Raspberry Pi model that meets Our project's requirements.
3. **Sensor Wiring:** Connect the selected sensors to the Raspberry Pi's GPIO pins according to the manufacturer's specifications.
4. **Data Acquisition:** Write Python code to read data from the sensors. Raspberry Pi's GPIO pins can be used to interface with various sensors.
5. **Data Processing:** Implement code on the Raspberry Pi to process sensor data. Calculate occupancy status based on sensor readings and defined thresholds.

2.3.2. IoT Cloud Integration

1. **Cloud Platform Selection:** Choose an IoT cloud platform that fits Our needs. Popular options include AWS IoT, Google Cloud IoT, Microsoft Azure IoT, or IBM Watson IoT.

2. **Cloud Account Setup**: Create an account and set up Our IoT project on the chosen cloud platform.
3. **Device Registration:** Register Our Raspberry Pi-based device as an IoT thing on the cloud platform. You will receive credentials (such as certificates and keys) for secure communication.
4. **Data Transmission:** Configure the Raspberry Pi to use the MQTT (Message Queuing Telemetry Transport) or HTTPS protocol to send sensor data to the cloud platform securely.

2.3.3. Data Processing in the Cloud

1. **Cloud Functionality:** Set up cloud functions or serverless computing services on Our chosen cloud platform to receive and process incoming sensor data.
2. **Data Storage:** Store processed data in a cloud-based database or storage service (e.g., Amazon DynamoDB, Google Cloud Firestore, or Azure Cosmos DB).
3. **Real-time Processing:** Implement real-time data processing to determine parking space availability and occupancy trends.

2.3.4. Mobile App Integration

1. **Mobile App Development**: Develop a mobile app (iOS/Android) that communicates with the cloud-based IoT platform to retrieve real-time parking information. Use appropriate development frameworks and libraries.
2. **API Communication:** Integrate APIs or SDKs provided by the cloud platform to enable secure communication between the mobile app and the cloud.
3. **Real-time Updates:** Implement real-time updates within the mobile app, ensuring users have access to the latest parking availability information.

2.3.5. User Interface

1. **App Interface:** Design an intuitive and user-friendly interface for the mobile app. incorporate maps, markers, search functionality, and navigation features to guide users to available parking spaces.

2.3.6. Testing and Deployment

1. **Testing:** Conduct thorough testing of the integrated system, including Raspberry Pi sensor data collection, cloud-based data processing, and mobile app functionality. Address any issues or bugs encountered during testing.
2. **Deployment:** Once testing is successful, deploy the integrated system in the parking area. Ensure all sensors are correctly placed and communicate effectively with the cloud platform.
3. **User Training:** If necessary, provide user training on how to use the mobile app and navigate to available parking spaces.
4. **Monitoring:** Implement monitoring and maintenance procedures to ensure the system continues to operate smoothly over time.