Smart Parking

Phase 1:: Project Definition and Design Thinking

Project Definition:

The project involves integrating IoT sensors into public transportation vehicles to monitor ridership, track locations, and predict arrival times. The goal is to provide real-time transit information to the public through a public platform, enhancing the efficiency and quality of public transportation services. This project includesdefining objectives, designing the IoT sensor system,developing the real-time transit informationplatform, and integrating them using IoT technology and Python.

Design Thinking:

1.Project Objectives:

1)a.Real-time parking space monitoring:

Real-time parking monitoring can help measure the optimal performance of parking assets by understanding spatial factors such as walking distances, price impacts, and demand, along with land use that may affect volumes. There are two ways to get real-time parking availability data: vehicle movement detection and vehicle video tracking, which imply the use of sensors and cameras

1)b. Mobile app integration:

*Mobile app integration is the process of connecting different apps and devices to achieve seamless data exchange and optimized workflows.

* It involves the transition of functionalities or resources from one mobile application to another.

* APIs define how one app can make requests from and send data to another app, allowing apps to share data and functionality through application programming interfaces. Mobile app integration provides innovative ways to work effectively and gives you the option of utilizing your complete infrastructure behind your applications

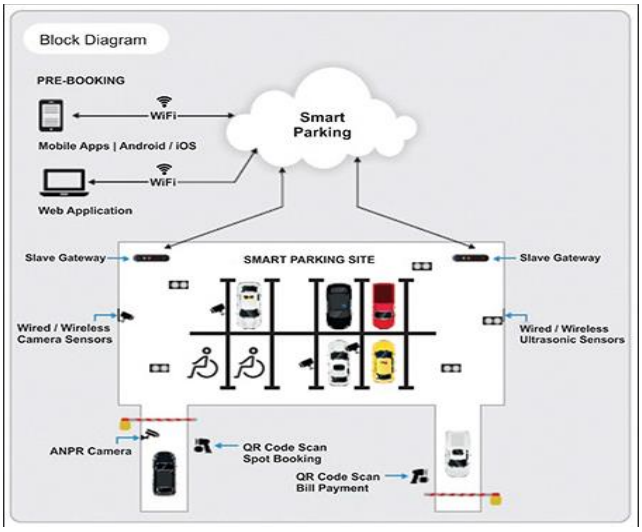
1)c.Efficient parking guidance:

Efficient parking guidance refers to a system thatprovides drivers with real-time information regarding the occupation and availability of parking spots in a controlled area. These systems are implemented in many different settings, such as airports, hospitals, malls, and campuses. Parking Guidance Systems (PGS) is an advanced car counting or vehicle detecting system that offers dynamic data to improve car park circulation and space optimization. It acts as an advanced car

counting or vehicle detection technology that facilitates vehicle circulation within a car park

2.IoT Sensor Design:

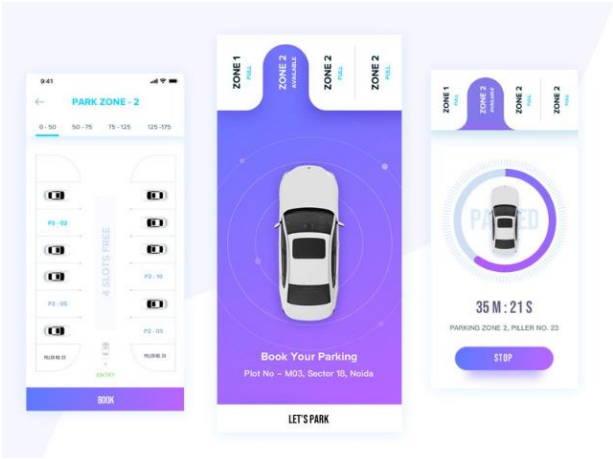
2)To design and deployment of IoT sensors in parking spaces to detect occupancy and availability:



A smart parking system uses IoT devices and sensors to collect real-time data on parking lot occupancy and transmits this information to the cloud or local network. It also involves building IoT apps for end-users, like parking administrators and drivers

3.Real-Time Transit Information Platform:

3)To Design a mobile app interface that displays real-time parking availability to users:



It presents a mobile application that can provide parking availability information as well as suggestions based on real-time sensor data. It provides a methodology for developing a user- friendly interface on Android handsets that can guide users in a fast and efficient way to perform online parking reservation for a specific duration.

4.Integration Approach:

4) Raspberry Pi will collect data from sensors and update the mobile app:

To collect sensor data with a Raspberry Pi, you need to:

*Ensure you are using a sensor that has an I2C, SPI, or UART interface.

*Connect the sensor to the appropriate pins on the Raspberry Pi.

*Write code to import the library corresponding to the interface, and then use that interface to read the sensor data.

Alternatively, you can collect data from the sensor using Python scripts available on GitHub.There are many ways to store the data, such as sending UDP messages from the Raspi to the PC, sending a TCP stream from the Raspi to the PC, throwing the readings into Redis and allowing anyone on your network to collect them, or publishing the readings from your Raspi with an MQTT client and subscribing to that topic on your PC as server.

To collect data from sensors and update the mobile app using Raspberry Pi, you could do the following:

*Connect sensors on breadboard and collect data using python and gpio.

*Install a webserver on your pi (apache2 or flask or others) to serve web pages containing that data.

*Look at that pi web page from your phone using chrome/firefox/other browser.

*Use MQTT to publish sensor data and subscribe with an app from the app store.

*Add MQTT to your Android app