PHASE-3: Python Program

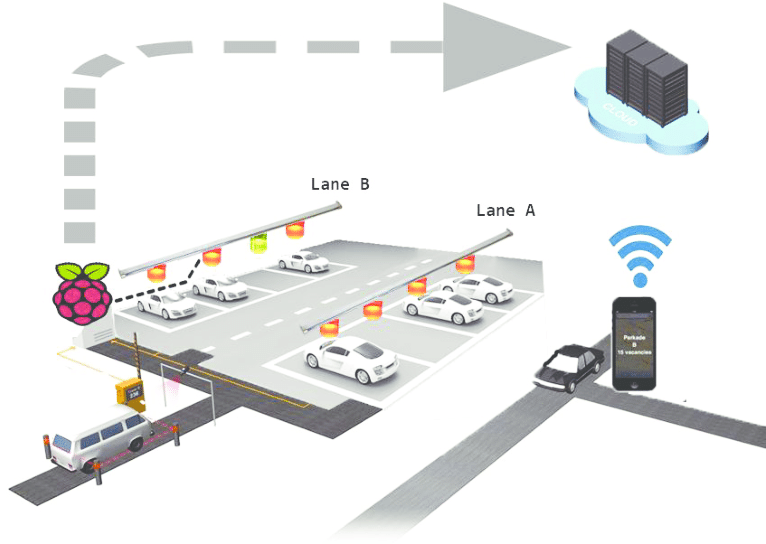
Project Title: Smart Parking System with IoT

Presented By : Sivaranjini M

Roll No : 312821243036

# Project Description:

Develop a Smart Parking System that leverages IoT technology, including IoT sensors and Raspberry Pi devices, to efficiently manage parking spaces, reduce congestion, and enhance the overall parking experience for users. This system will provide real-time parking space availability information and optimize parking space allocation.



To collect data from an IoT sensor and send it to a database using Python, these steps are followed:

1. Set up your IoT sensor: Ensure that your IoT sensor is properly connected to your Raspberry Pi, Arduino, or any other IoT device you are using.

2. Choose a database: Select a database to store your sensor data. For this example, we’re use MySQL as the database.

3. Install required libraries: Install the necessary Python libraries to interact with the IoT sensor and the database. Libraries like `pymysql`, `serial`, or others depending on your IoT device and database choice.

4. Create a database table: Set up a table in your database to store the sensor data. You can use a SQL command to create the table.

5. Collect and send data: Write a Python program to collect data from the IoT sensor and send it to the database. Here's an example program that assumes you're using a Raspberry Pi and a MySQL database:

# Program:

import pymysql

import serial

import time

# Database connection parameters

db\_host = "your\_database\_host"

db\_user = "your\_database\_user"

db\_password = "your\_database\_password"

db\_name = "your\_database\_name"

# IoT sensor configuration

serial\_port = "/dev/ttyUSB0"

# Change this to your sensor's serial port

baud\_rate = 9600

# Connect to the database

db = pymysql.connect(host=db\_host, user=db\_user, password=db\_password, database=db\_name)

cursor = db.cursor()

# Create a table to store sensor data

create\_table\_query = """

CREATE TABLE sensor\_data (

id INT AUTO\_INCREMENT PRIMARY KEY,

timestamp TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

sensor\_value FLOAT

)

"""

cursor.execute(create\_table\_query)

db.commit()

# Connect to the IoT sensor

ser = serial.Serial(serial\_port, baud\_rate)

try:

while True:

# Read data from the sensor

sensor\_data = float(ser.readline().decode().strip())

# Insert data into the database

insert\_query = "INSERT INTO sensor\_data (sensor\_value) VALUES (%s)"

cursor.execute(insert\_query, (sensor\_data,))

db.commit()

print(f"Data inserted: {sensor\_data}")

time.sleep(10) # Adjust the sleep time as needed

except KeyboardInterrupt:

ser.close()

db.close()

This code assumes your IoT sensor sends data as a single float value through a serial connection.

# Note:

Since the code is a prototype only placeholder values are given instead of actual references.