**SMART PARKING**

**PHASE 3**

**DEVELOPMENT(Part1)**

**INTRODUCTION**

Introducing the Smart Parking System revolutionizing the parking experience. Real-time availability, reservation, and payment options ensure convenience for drivers. Efficient space management and navigation features reduce traffic congestion. Enhanced security and safety measures provide peace of mind. Data-driven insights empower parking lot operators to optimize operations and maximize revenue

**PROGRAMJ J ava Script:**

// script.js

// Define the number of parking spots

const totalSpots = 25; // Adjust the total number of parking spots as per your requirement

// Generate the parking lot grid dynamically

const parkingGrid = document.querySelector('.parking-grid');

for (let i = 1; i <= totalSpots; i++) {

const spot = document.createElement('div');

spot.classList.add('parking-spot');

spot.textContent = i;

parkingGrid.appendChild(spot);

}

# Adafruit IO Setup

ADAFRUIT\_IO\_USERNAME =

'YOUR\_ADAFRUIT\_IO\_USERNAME'

ADAFRUIT\_IO\_KEY = 'YOUR\_ADAFRUIT\_IO\_KEY'

aio = Client(ADAFRUIT\_IO\_USERNAME, ADAFRUIT\_IO\_KEY)

# Noise Sensor Setup

spi = board.SPI()

cs = digitalio.DigitalInOut(board.D5)

mcp = MCP.MCP3008(spi, cs)

channel = AnalogIn(mcp, MCP.P0) # Use the appropriate channel

def read\_noise\_level():

return channel.voltage # Modify this as needed based on your sensor's output

while True:

try:

noise\_level = read\_noise\_level()

print(f'Noise Level: {noise\_level} V')

# Send data to Adafruit IO

aio.send('noise-level', noise\_level)

time.sleep(60) # Send data every 60 seconds (adjust as needed)

except Exception as e:

print(f"Error: {e}")

time.sleep(10) # Wait for a while before retrying

// Function to change the status of a parking spot

function toggleSpotStatus(spot) {

spot.classList.toggle('occupied');

}

// Function to handle spot click event

function handleSpotClick(event) {

const spot = event.target;

toggleSpotStatus(spot);

}

// Add event listener to each parking spot

const spots = document.querySelectorAll('.parking-spot');

spots.forEach(spot => {

spot.addEventListener('click', handleSpotClick);

});

time.sleep(60) # Send data every 60 seconds (adjust as needed)

except Exception as e:

print(f"Error: {e}")

time.sleep(10) # Wait for a while before retrying

**SUMMARY:**

In this code, we first define the number of parking spots using the totalSpots variable. You can adjust this value based on your requirements.

Next, we select the parking grid element using document.querySelector('.parking-grid') and loop through the number of spots using a for loop. Inside the loop, we create a div element for each spot, assign it the class parking-spot, and set its text content to the spot number. We then append each spot element to the parking grid.

We define a toggle Spot Status function that toggles the occupied class of a spot element when called. This function is responsible for changing the status (occupied or available) of a parking spot.

The handle Spot Click function is assigned as the event listener for each parking spot. It calls the toggle Spot Status function with the clicked spot element when a spot is clicked. This function is responsible for handling the spot click event and toggling the spot's status.

**NOTE:**

Make sure to save this code in a separate JavaScript file (e.g., script.js) and link it to your HTML file using the <script src="script.js"></script> tag.

**CONCLUSION:**

Finally, we select all the parking spots using document.querySelectorAll('.parking-spot') and add the handle Spot Click event listener to each spot using the for Each method.