

**KGiSL INSTITUTE OF TECHNOLOGY**

(Approved By AICTE, New Delhi, Affiliate to Anna University Recognized by UGC, Accredited by NBA(IT)

265, KGISL Campus, Thudiyalur Road, Saravanampatti, Coimbatore-641035**)**

**DEPARTMENT OF**

**ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**

**NAAN MUDHALVAN - INTERNET OF THINGS**

**SMART PARKING**

**NAME :KISHORE .S**

**REG NO :** 711721243048

**NM ID :** au711721243048

**TEAM MENTOR :** Mr**.** Mohankumar M

**TEAM EVALUATOR:** Ms. Akilandeeshwari M

**INTRODUCTION:**

Smart parking systems leveraging the Internet of Things (IoT) technology have emerged as transformative solutions to address urban congestion, optimize parking space utilization, and enhance overall traffic management. Traditional parking systems often suffer from inefficiencies, contributing to increased traffic, environmental pollution, and driver frustration. The integration of IoT into parking infrastructure aims to mitigate these issues by providing real-time data and intelligent management of parking spaces.

**SOFTWARE REQUIREMENTS:**

* Python (3.x)
* Flask (for web server)
* SQLite or any other database for storing parking space status
* Web development tools (HTML, CSS) for the user interface
* Libraries for sensor communication (e.g., GPIO libraries for Raspberry Pi)

**HARDWARE REQUIREMENTS:**

* Raspberry Pi or Arduino (for sensor integration and data processing)
* Parking space occupancy sensors (e.g., ultrasonic sensors, infrared sensors)
* LED indicators for each parking space
* Internet connectivity (Wi-Fi or Ethernet)

**PROCEDURE:**

**Step 1: Initialization**

Initialize an array or data structure to represent the status of each parking space. For example, parking\_status = [0, 0, 1, 0, 1] where 0 denotes a vacant space and 1 denotes an occupied space.

**Step 2: Sensing and Updating**

Continuously monitor the status of each parking space using sensors.

Update the parking\_status array based on real-time occupancy information.

**Step 3: Allocation Logic**

When a new vehicle arrives:

* Check for the nearest vacant parking space based on user preferences.
* If a preferred space is available, allocate that space.
* Otherwise, allocate the nearest available space.

**Step 4: Deallocation Logic**

When a vehicle leaves:

* Mark the corresponding parking space as vacant.
* If desired, update the user's historical data for future predictions.

**Step 5: Optimization**

Implement optimization techniques to improve space allocation efficiency.

For example, consider historical data to predict peak usage times and allocate spaces accordingly. Prioritize spaces that are easier to access or strategically allocate spaces to reduce congestion.

**Step 6: Real-time Adjustments**

Continuously adjust the parking allocation based on real-time data.

Reevaluate and update the allocation if conditions change (e.g., unexpected high demand).

**Step 7: User Interface**

Update the display to reflect the current parking space status to the user.

**SOURCE CODE:**

**BACKEND USING PYTHON FLASK:**

pip install Flask

from flask import Flask, render\_template, request

app = Flask(\_\_name\_\_)

parking\_status = [0, 0, 1, 0, 1] # Sample parking status array

@app.route('/')

def home():

return render\_template('index.html', parking\_status=parking\_status)

@app.route('/allocate', methods=['POST'])

def allocate():

# Logic to allocate parking space

# For simplicity, this example assumes that the frontend sends a POST request

# with the parking space ID to be allocated.

space\_id = int(request.form.get('space\_id'))

parking\_status[space\_id] = 1 # Mark the space as occupied

return render\_template('index.html', parking\_status=parking\_status)

@app.route('/deallocate', methods=['POST'])

def deallocate():

# Logic to deallocate parking space

space\_id = int(request.form.get('space\_id'))

parking\_status[space\_id] = 0 # Mark the space as vacant

return render\_template('index.html', parking\_status=parking\_status)

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

**FRONTEND (HTML AND CSS):**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Smart Parking System</title>

<style>

.occupied {

background-color: red;

}

.vacant {

background-color: green;

}

</style>

</head>

<body>

<h1>Smart Parking System</h1>

<ul>

{% for idx, status in enumerate(parking\_status) %}

<li class="{% if status == 1 %}occupied{% else %}vacant{% endif %}">

Parking Space {{ idx }} - {% if status == 1 %}Occupied{% else %}Vacant{% endif %}

{% if status == 0 %}

<form method="post" action="/allocate">

<input type="hidden" name="space\_id" value="{{ idx }}">

<button type="submit">Allocate</button>

</form>

{% else %}

<form method="post" action="/deallocate">

<input type="hidden" name="space\_id" value="{{ idx }}">

<button type="submit">Deallocate</button>

</form>

{% endif %}

</li>

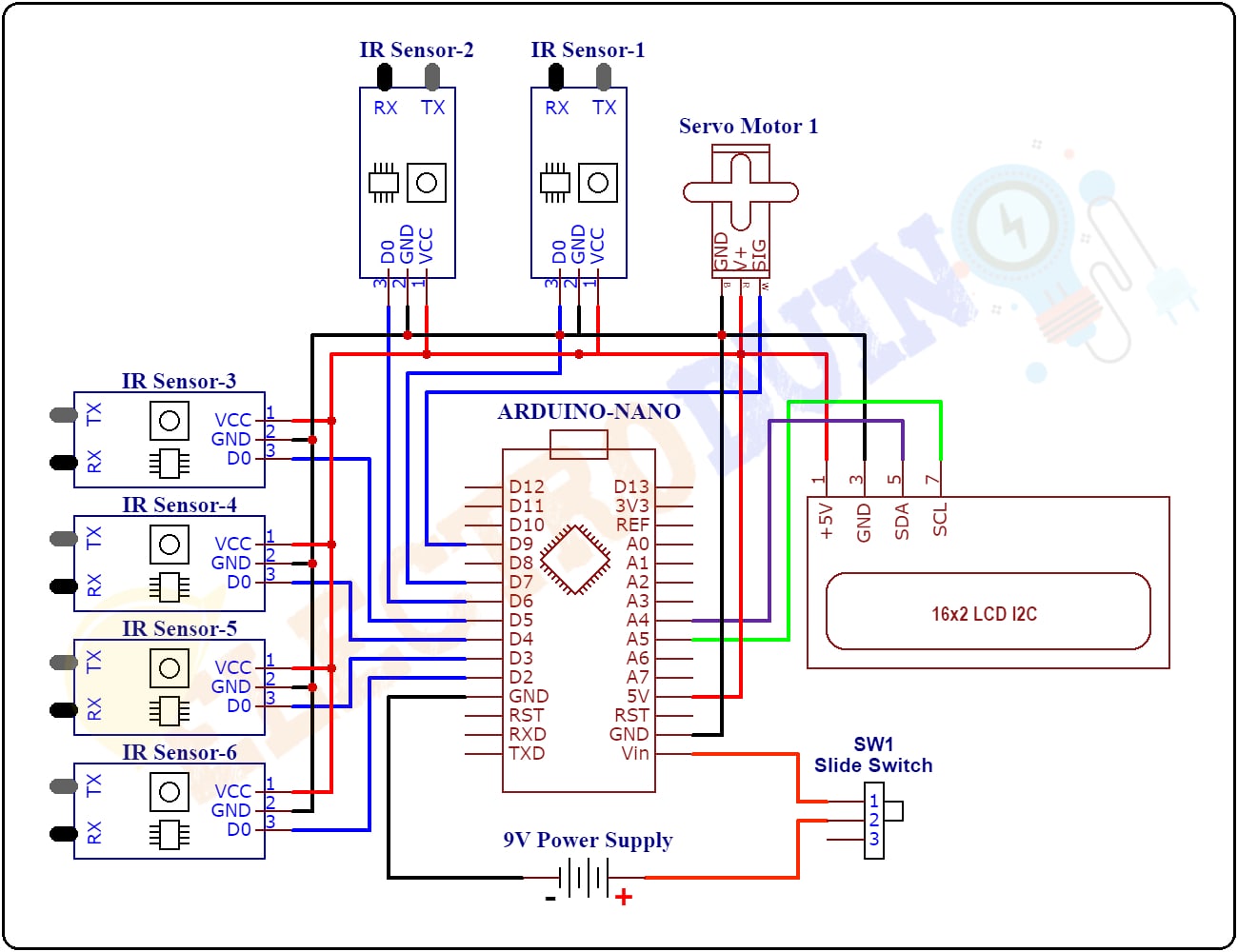
{% endfor %}

</ul>

</body>

</html>

**PCB DIAGRAM:**



**CONCLUSION:**

In conclusion,Smart parking systems powered by the Internet of Things (IoT) have emerged as powerful solutions to address the challenges associated with urban parking and traffic management. It represents a significant step toward creating more livable, sustainable, and technologically advanced cities The integration of IoT technologies into parking infrastructure has led to a paradigm shift in how we perceive and manage parking spaces. By leveraging real-time data, connectivity, and intelligent algorithms, these systems contribute to the transformation of urban mobility, making parking more efficient, convenient, and environmentally friendly. The continued development and deployment of smart parking solutions hold promise for creating smarter, more accessible, and more enjoyable urban spaces for everyone.