**Phase 3 project:**

**Project Title: Smart parking**

**Project ID:** proj\_223739\_Team\_4

**College:** Gnanamani College of Technology

**Branch:** B.Tech/Information Techology

**Year:** IIIrd

**Phase 3: Development**

**Sensor Technology:**

**Smart parking relies on sensors placed in parking spaces to detect the presence of vehicles. These sensors can be ultrasonic, infrared, magnetic, or camera-based**

**Data Collection:**

**Data from the sensors is collected and transmitted to a central system, often using IoT (Internet of Things) technology. This data includes information on available parking spaces.**

**Payment and Reservations:**

**Smart parking systems often allow users to make reservations and payments for parking spaces through mobile apps or online platforms, reducing the need for physical payment methods.**

**Integration with Smart Cities:**

**Smart parking is often integrated into broader smart city initiatives, connecting with traffic management systems, public transportation, and environmental monitoring for a more holistic urban experience.**

**Security and Privacy:**

**Ensuring the security of data and the privacy of users is a critical aspect of smart parking system development**

**CODING:**

**from collections import deque**

**class ParkingLot:**

**def \_\_init\_\_(self, rows, columns):**

**self.rows = rows**

**self.columns = columns**

**self.grid = [[0] \* columns for \_ in range(rows)] # 0 represents an available parking space**

**def mark\_occupied(self, row, column):**

**self.grid[row][column] = 1 # 1 represents an occupied parking space**

**def is\_valid(self, row, column):**

**return 0 <= row < self.rows and 0 <= column < self.columns**

**def find\_available\_spot(self, start\_row, start\_column):**

**visited = [[False] \* self.columns for \_ in range(self.rows)]**

**queue = deque([(start\_row, start\_column, 0)]) # (row, column, distance)**

**directions = [(1, 0), (-1, 0), (0, 1), (0, -1)] # Up, Down, Left, Right**

**while queue:**

**row, column, distance = queue.popleft()**

**if self.grid[row][column] == 0:**

**return row, column, distance**

**for dr, dc in directions:**

**new\_row, new\_column = row + dr, column + dc**

**if self.is\_valid(new\_row, new\_column) and not visited[new\_row][new\_column]:**

**visited[new\_row][new\_column] = True**

**queue.append((new\_row, new\_column, distance + 1))**

**return None**

**# Example usage**

**parking\_lot = ParkingLot(4, 4)**

**# Simulate some occupied spots**

**parking\_lot.mark\_occupied(1, 2)**

**parking\_lot.mark\_occupied(2, 1)**

**parking\_lot.mark\_occupied(2, 2)**

**start\_row, start\_column = 0, 0**

**result = parking\_lot.find\_available\_spot(start\_row, start\_column)**

**if result:**

**row, column, distance = result**

**print(f"Found available spot at ({row}, {column}) with a distance of {distance} units.")**

**else:**

**print("No available parking spots found.")**

**This code sets up a parking lot grid and uses BFS to find the nearest available parking spot from a given starting position. You can expand upon this basic example to create a more comprehensive smart parking system with additional features like reservations, payment processing, and real-time updates.**

**Team members:**

R.Abinaya (620821205001)

P.sneka(620821205054)

S.Pradeepa(620821205040)

M.Karthika(620821205026)

E.Sugashini(620821205057)