RAJALAKSHMI INSTITUTE OF TECHNOLOGY

(An Autonomous Institution, Affiliated to Anna University, Chennai)

DEPARTMENT OF CSE (ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING)

ACADEMIC YEAR 2025 - 2026

SEMESTER III

ARTIFICIAL INTELLIGENCE LABORATORY

MINI PROJECT REPORT

REGISTER NUMBER	**2117240030118**
NAME	**ROHITH H**
PROJECT TITLE	**Al-Based Smart Parking System**
DATE OF SUBMISSION	**31/10/2025**
FACULTY IN-CHARGE	**Mrs.M.Bhavani – Assistant professor**

Signature of Faculty In-charge

TITLE: AI - BASED SMART PARKING SYSTEM

INTRODUCTION

➢Artificial Intelligence (AI) enables machines to simulate human intelligence and perform tasks that normally require human cognition, such as learning, problem-solving, and decision-making.

➢Parking congestion is a common urban problem that causes delays and frustration. The Smart Parking System uses AI-based logic to manage and monitor parking spaces efficiently.

➢This project demonstrates how AI can optimize parking availability and reduce human effort.

PROBLEM STATEMENT

➢ Finding available parking slots manually is time-consuming and often leads to congestion.

➢There is a need for an intelligent system that can automatically track, update, and display available parking slots in real-time to improve efficiency and reduce waiting time.

GOAL

➢To design a smart parking system using AI principles that efficiently allocates available parking slots and provides a clear status of occupied and empty spaces.

➢The goal is to simulate real-time parking management using Python logic.

THEORETICAL BACKGROUND

➢This project applies AI logic to simulate real-time decision-making in parking management.

➢The algorithm keeps track of occupied and empty slots dynamically and updates them after every vehicle movement.

➢While real-world smart parking systems use image processing or IoT sensors, this mini-project focuses on logical simulation using Python, which demonstrates the AI decision-making process clearly.

➢A simple rule-based AI algorithm is used:

➢If slot available → Park vehicle

➢If no slot → Display “Parking Full”

➢If vehicle exits → Free up slot

ALGORITHM EXPLANATION WITH EXAMPLE

➢Algorithm Steps:

➢1. Initialize the parking lot with a fixed number of slots.

➢2. When a vehicle parks, increment the occupied slot count.

➢3. When a vehicle exits, decrement the occupied slot count.

➢4. Display the updated parking status each time.

➢5. Prevent over-parking or removing vehicles when none exist.

➢Example:

➢ total slots = 5 and 3 vehicles park,

➢Then Occupied = 3, Empty = 2.

➢If one vehicle leaves, Occupied = 2, Empty = 3.

IMPLEMENTATION AND CODE

➢ Python code

```
Class SmartParking:
   Def __init__(self, total_slots):
      Self.total_slots = total_slots
      Self.occupied_slots = 0
   Def park_vehicle(self):
      If self.occupied_slots < self.total_slots:
```

```
  Self.occupied_slots += 1
        Print(f"✅ Vehicle parked successfully. Slots filled: {self.occupied_slots}/{self.total_slots}")
      Else:


      Print("❌ Parking Full! No space available.")
   Def remove_vehicle(self):
      If self.occupied_slots > 0:
         Self.occupied_slots -= 1
         Print(f"🚗 Vehicle exited. Slots filled: {self.occupied_slots}/{self.total_slots}")
      Else:
         Print("⚠️ No vehicles to remove!")
   Def show_status(self):
      Empty_slots = self.total_slots – self.occupied_slots
      Print("\n--- Parking Lot Status ---")
      Print(f"Total Slots: {self.total_slots}")
      Print(f"Occupied Slots: {self.occupied_slots}")
      Print(f"Empty Slots: {empty_slots}")
      Print("--------------------------\n")
Parking = SmartParking(5)
Parking.show_status()
Parking.park_vehicle()
Parking.park_vehicle()
Parking.park_vehicle()
Parking.show_status()
Parking.remove_vehicle()
Parking.show_status()
```

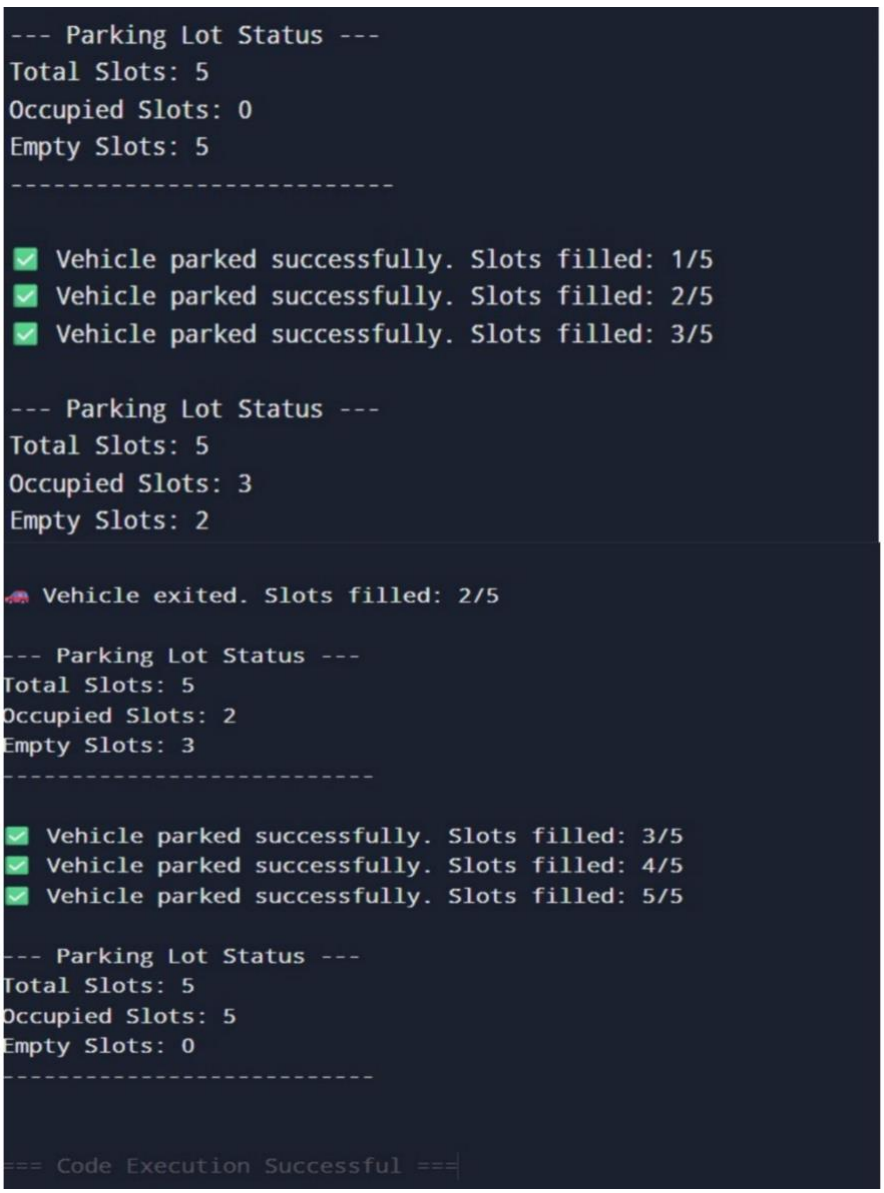
parking.park_vehicle()

parking.park_vehicle()

parking.park_vehicle()

parking.show_status()

OUTPUT



RESULTS AND FUTURE ENHANCEMENT

➢The Smart Parking System successfully simulates how AI logic can optimize parking management.
➢It efficiently tracks vehicle entries and exits and prevents overfilling.
➢In the future, this system can be enhanced using:
➢Camera sensors and image processing (OpenCV) for automatic vehicle detection.
➢IoT integration to monitor real-time data.
➢Mobile app interface for users to check slot availability.

REFERENCES

➢1. Python Official Documentation — https://docs.python.org

➢2. OpenCV Documentation — https://opencv.org

➢3. Towards Data Science — Smart Parking System Articles

➢4. ResearchGate — AI in Smart Cities Papers

➢5. IEEE Xplore — AI Applications in Urban Systems