**SMART PARKING**

**SYSTEM**

**A PROJECT REPORT**

**Submitted By :- SHUBHAM GUPTA**

**Date :- 20-08-2024**

**TABLE OF CONTENT**

|  |  |  |
| --- | --- | --- |
| **S.No** | **TOPIC** | **PAGE NO** |
| *1* | Introduction | 1 |
| *2* | Objective | 3 |
| *3* | Tools/Environment | 4 |
| *4* | Program Code | 7 |
| *5* | Input and Output Screens | 16 |
| *6* | Limitations of the Project | 19 |
| *7* | Future Applications of the Project | 20 |
| *8* | Bibliography | 21 |

**INTRODUCTION**

Smart parking systems are revolutionizing the way cities manage parking, addressing the persistent issue of finding available spaces in congested urban environments. Traditional parking methods often lead to inefficiencies, such as excessive traffic congestion and increased pollution, as drivers spend significant time searching for parking spots. In contrast, smart parking systems utilize a combination of advanced technologies, including Internet of Things (IoT) sensors, wireless communication networks, and data analytics, to provide real-time information on parking availability. Library Management System .

**SMART PARKING SYSTEM..**

These systems analyze the data to offer insights into parking patterns and predict available spaces, which can then be accessed by drivers through mobile apps or digital signage. This real-time information helps reduce the time spent searching for parking, thereby alleviating traffic congestion and lowering vehicle emissions. Additionally, smart parking systems can increase revenue for cities and private operators by optimizing space utilization and enhancing operational efficiency.



**Key Features of the Smart Parking System**

Smart parking systems offer a range of features designed to enhance the efficiency of parking management and improve the user experience. Here are some of the key features:

1. **Real-Time Occupancy Detection:**
   * Sensors: Utilizes various sensors (ultrasonic, infrared, magnetic) to detect whether parking spaces are occupied or available.
   * Updates: Provides real-time updates on space availability, allowing drivers to find open spots quickly.
2. **Mobile Applications:**
   * Navigation Assistance: Offers directions to available parking spaces via a mobile app.
   * Reservations: Allows users to reserve and pay for parking spots in advance.
3. **Dynamic Pricing:**
   * Pricing Models: Implements variable pricing based on demand, time of day, or location.
   * Revenue Optimization: Adjusts prices to maximize revenue and manage parking space utilization effectively.
4. **Digital Signage:**
   * Real-Time Information: Displays available parking spaces and directions on electronic signs located throughout the city.
   * Guidance: Helps drivers navigate to the nearest available parking spots.
5. **Data Analytics:**

Usage Patterns: Analyzes parking data to identify peak usage times, high-demand areas, and overall space utilization.

**OBJECTIVES**

The objectives of smart parking systems are designed to address common parking challenges and leverage technology to improve urban mobility. Here are the primary objectives:

1. **Reduce Parking Search Time:**
   * Efficiency: Minimize the time drivers spend searching for available parking spots, leading to a more efficient and stress-free parking experience.
2. **Alleviate Traffic Congestion:**
   * Flow Improvement: Decrease traffic congestion caused by drivers circling to find parking by providing real-time information on space availability.
3. **Optimize Space Utilization:**
   * Maximize Use: Improve the utilization of available parking spaces through better management and dynamic pricing strategies.
4. **Increase Revenue:**
   * Revenue Generation: Enhance the revenue potential from parking fees by implementing dynamic pricing models and efficient space management.
5. **Enhance User Experience:**
   * Convenience: Provide a user-friendly experience with features such as mobile apps for space reservations, real-time notifications, and navigation assistance.
6. **Support Sustainable Urban Development:**
   * Environmental Impact: Reduce vehicle emissions and pollution by minimizing the time vehicles spend idling while searching for parking.

**TOOLS AND ENVIORNMENT**

**HARDWARE REQUIREMENTS**

**Processor:** Minimum Pentium IV 2.4 GHZ

**RAM:** At Least 100 MB

**Disk Space:** At Least 500 MB

# **SOFTWARE REQUIREMENTS**

**Operating System:** Windows,IOS,LINUX,Etc.

**Code Compiler :** Visual Code Studio / Dev C++/ Turbo C++/Etc.

**ENTITY RELATIONSHIP DIAGRAM**

**OCCUPIES**

**PARKING SLOT**

**VEHICLE**

**PARKING\_SESSION**

**GENERATES**

**DATA FLOW DIAGRAM**

SMART PARKING SYSTEM

DSPLAY AVAILABILTIYENTER PARKING

**USER**

EXIT PARKING

**VEHICLE**

PAYMENT

DATA DATA

DATABASE

**PROGRAM CODE**

#include <iostream>

#include <string>

#include <iomanip>

#include <fstream>

using namespace std;

class ParkingSystem {

public:

string licensePlates[50];

string vehicleTypes[50];

double parkingCharges[50];

ParkingSystem() {

for (int i = 0; i < 50; ++i) {

licensePlates[i] = "";

vehicleTypes[i] = "";

parkingCharges[i] = 0.0;

}

}

void displayAvailableSlots() {

int columns = 10;

cout << "Available Slots:\n";

cout << "----------------------------\n";

for (int i = 0; i < 50; ++i) {

if (licensePlates[i].empty()) {

cout << setw(3) << i + 1 << " ";

} else {

cout << setw(3) << "X" << " ";

}

if ((i + 1) % columns == 0) {

cout << endl;

}

}

cout << "\n----------------------------\n";

}

void EnterVehicle() {

int slotNumber;

string plate, type;

double hours;

displayAvailableSlots();

cout << "Enter slot number to Enter: ";

cin >> slotNumber;

if (slotNumber < 1 || slotNumber > 50 || !licensePlates[slotNumber - 1].empty()) {

cout << "Invalid slot number!" << endl;

return;

}

cout << "Enter license plate: ";

cin >> plate;

cout << "Enter vehicle type: ";

cin >> type;

cout << "Enter number of hours to park: ";

cin >> hours;

licensePlates[slotNumber - 1] = plate;

vehicleTypes[slotNumber - 1] = type;

parkingCharges[slotNumber - 1] = calculateCharge(hours);

cout << "Vehicle parked in slot " << slotNumber << endl;

cout << "Charge for " << hours << " hours: Rs" << parkingCharges[slotNumber - 1] << endl;

SaveToFile();

}

void ExitVehicle() {

int slotNumber;

double payment;

cout << "Enter slot number to Exit: ";

cin >> slotNumber;

if (slotNumber < 1 || slotNumber > 50 || licensePlates[slotNumber - 1].empty()) {

cout << "Invalid slot number!" << endl;

return;

}

cout << "Your parking charge is: Rs" << parkingCharges[slotNumber - 1] << endl;

do {

cout << "Please enter payment amount: ";

cin >> payment;

if (payment < parkingCharges[slotNumber - 1]) {

cout << "Insufficient Amount.\n";

}

} while (payment < parkingCharges[slotNumber - 1]);

double change = payment - parkingCharges[slotNumber - 1];

cout << "Payment successful! Your change is Rs" << change << endl;

licensePlates[slotNumber - 1] = "";

vehicleTypes[slotNumber - 1] = "";

parkingCharges[slotNumber - 1] = 0.0;

cout << "Vehicle removed from slot " << slotNumber << endl;

SaveToFile();

}

void DisplayParkedVehicles() {

cout << "Parked Vehicles:\n";

int count = 0;

for (int i = 0; i < 50; ++i) {

if (!licensePlates[i].empty()) {

cout << "Slot " << i + 1 << ": " << licensePlates[i] << " (" << vehicleTypes[i] << "), Charge: Rs"

<< parkingCharges[i] << endl;

count++;

}

}

if (count == 0) {

cout << "No vehicles parked." << endl;

}

}

double calculateCharge(double hours) {

double ratePerHour = 10.00;

return hours \* ratePerHour;

}

void SaveToFile() {

ofstream outfile("parking\_data.txt");

if (outfile.is\_open()) {

for (int i = 0; i < 50; ++i) {

outfile << licensePlates[i] << endl;

outfile << vehicleTypes[i] << endl;

outfile << parkingCharges[i] << endl;

}

outfile.close();

} else {

cout << "Failed to open data file for writing.\n";

}

}

void FetchFromFile() {

ifstream infile("parking\_data.txt");

if (infile.is\_open()) {

for (int i = 0; i < 50; ++i) {

getline(infile, licensePlates[i]);

getline(infile, vehicleTypes[i]);

infile >> parkingCharges[i];

infile.ignore();

}

infile.close();

} else {

cout << "Failed to open data file. Starting with empty parking slots.\n";

}

}

};

int main() {

ParkingSystem sy;

sy.FetchFromFile();

int choice;

do {

cout << "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Smart Parking System\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout << "1. Enter Parking\n";

cout << "2. Exit Parking\n";

cout << "3. Display Parked Vehicles\n";

cout << "4. Exit\n";

cout << "Enter your choice: ";

cin >> choice;

cout << endl;

switch (choice) {

case 1:

sy.EnterVehicle();

break;

case 2:

sy.ExitVehicle();

break;

case 3:

sy.DisplayParkedVehicles();

break;

case 4:

cout << "Exiting...\n";

sy.SaveToFile();

break;

default:

cout << "Invalid choice! Try again.\n";

}

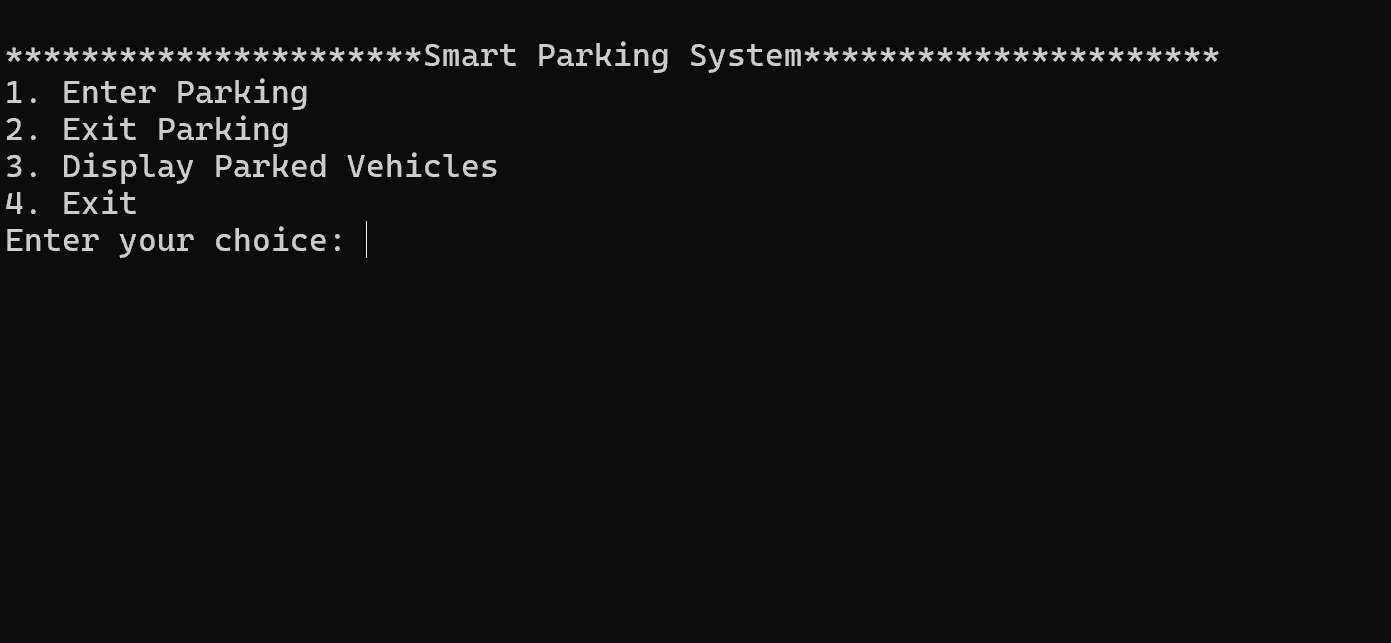
} while (choice != 4);

return 0;

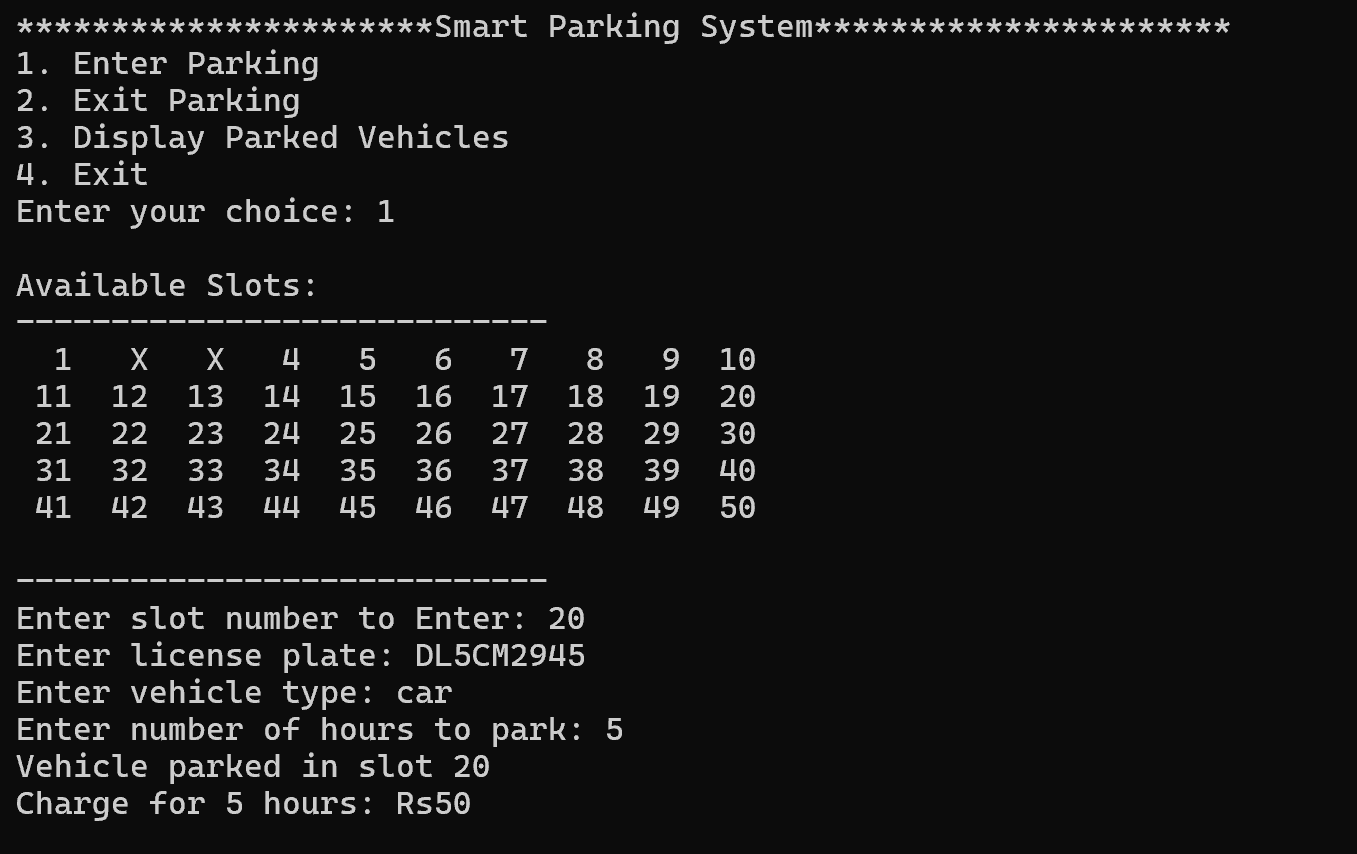
}

**INPUT / OUTPUT SCREENS**

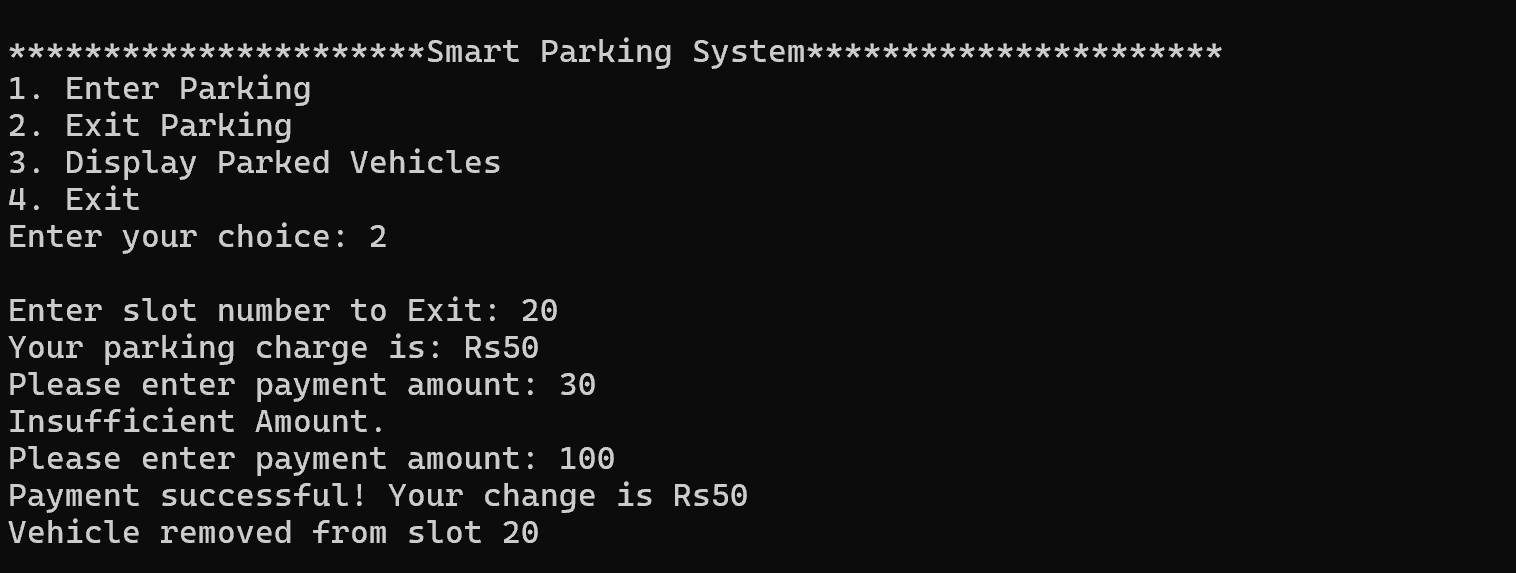
MAIN MENU



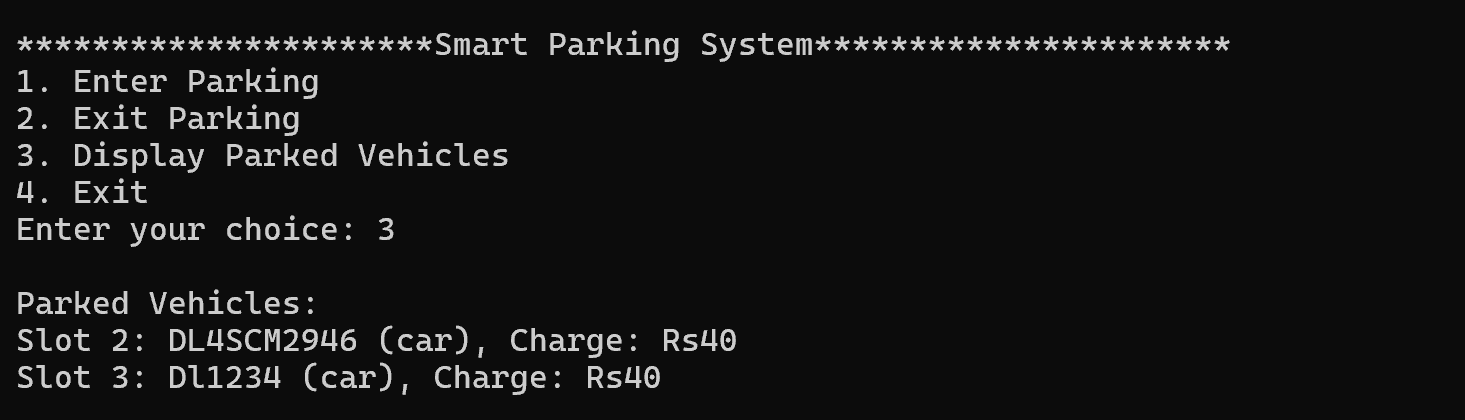
ENTER VEHICLE



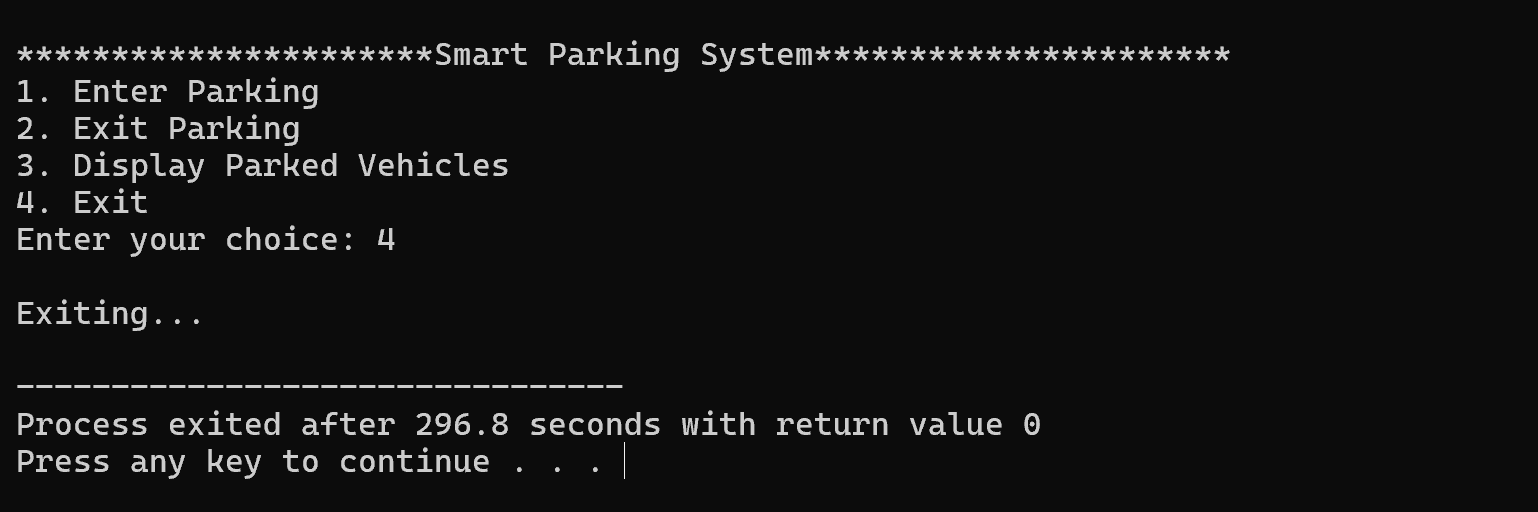
EXIT VEHICLE



DISPLAY PARKED VEHICLES



EXITING



**LIMITATIONS OF SMART PARKING SYSTEM**

Despite their numerous advantages, smart parking systems also come with certain limitations that can impact their effectiveness and adoption. Here are some of the key limitations:

1. **High Initial Costs:**
   * Installation Expenses: The deployment of sensors, communication networks, and management software requires significant upfront investment.
   * Maintenance Costs: Ongoing maintenance, including sensor calibration and system upgrades, adds to the overall expense.
2. **Complex Implementation:**
   * Infrastructure Challenges: Retrofitting existing parking infrastructure with smart technologies can be complex and time-consuming, particularly in older cities.
   * Integration Issues: Integrating smart parking systems with existing urban infrastructure, such as traffic management and public transportation systems, can be challenging.
3. **Data Privacy Concerns:**
   * Personal Data: The collection and storage of user data, including parking habits and location information, raise privacy concerns and require robust data protection measures.
4. **System Reliability:**
   * Sensor Malfunction: Sensors and other hardware components can fail or provide inaccurate data, leading to incorrect information about space availability.

**FUTURE APPLICATION OF THE PROJECT**

The future applications of smart parking systems are vast and will likely evolve alongside advancements in technology, urban planning, and the increasing complexity of smart cities. Here are some potential future applications:

1. **Integration with Autonomous Vehicles:**
   * **Self-Parking Vehicles:** Smart parking systems could work in tandem with autonomous vehicles, allowing them to identify available parking spaces and park themselves without human intervention.
   * **Vehicle Retrieval:** Autonomous vehicles could be summoned from parking spaces to pick up passengers, reducing the need for dedicated parking near high-demand areas.
2. **Smart Mobility Hubs:**
   * **Multimodal Integration:** Smart parking systems could be integrated into smart mobility hubs that combine parking with access to public transportation, bike-sharing, and ride-hailing services, promoting seamless urban mobility.
   * **EV Charging Stations:** Future parking systems could incorporate more advanced electric vehicle (EV) charging infrastructure, allowing for dynamic allocation of spaces based on charging needs.
3. **Dynamic Urban Space Utilization:**
   * **Flexible Parking Spaces:** As urban areas continue to densify, smart parking systems could enable dynamic reconfiguration of parking spaces, converting them into other uses (like pop-up retail or green spaces) during off-peak hours.
   * **Real-Time Space Allocation:** Systems could allocate parking spaces dynamically based on real-time demand, allowing for temporary repurposing of spaces for events or delivery zones.

**BIBLIOGRAPHY**

**Books**

* Let Us C by Yashavant Kanetkar.
* Let us C++ by Yashavant Kanetkar.
* C in Depth by S.K Srivastava.
* The C++ Programming Language By Bjarne Stroustrup.

**Websites**

* [www.google.com](http://www.google.com)
* [www.youtube.com](http://www.youtube.com)
* [www.w3schools.com](http://www.w3schools.com)
* [www.geeksforgeeks.com](http://www.geeksforgeeks.com)