A blue and black logo

Description automatically generated

Project Report File

Parking Management System

BSCS-1G

By

Abdul Ahad, Syed Aayan Mahmood & Syed Hur Ali

# Parking Management System

# Introduction (Aim or Motivation)

Efficient management of parking facilities is a critical aspect of urban infrastructure, directly impacting traffic flow, accessibility, and user satisfaction. The **Parking Management System** aims to streamline the process of parking and unparking vehicles, monitor occupancy, manage waiting lists, and maintain historical records of vehicle movements. By automating these tasks, the system reduces manual intervention, minimizes errors, and enhances the overall parking experience for both administrators and users.

# Background (Research & Project Selection)

**Research & Project Selection**

Urbanization has led to an exponential increase in the number of vehicles, thereby escalating the demand for organized parking solutions. Traditional parking management methods, often reliant on manual record-keeping and physical monitoring, are prone to inefficiencies and inaccuracies. Modern advancements in software development provide opportunities to develop automated systems that address these challenges. The decision to develop a Parking Management System stems from the need to leverage technology to optimize parking operations, enhance user convenience, and ensure effective utilization of parking spaces.

**Project Specification**

The Parking Management System is designed to cater to both administrators and users with distinct functionalities:

* **Admin Features:**
  + Create and configure parking areas.
  + View and manage parked vehicles.
  + Access detailed vehicle histories.
  + Monitor and modify parking occupancy.
  + Generate Reports.
* **User Features:**
  + Park and unpark vehicles.
  + View real-time parking availability.
  + Manage waiting lists when parking areas are full.
  + Receive notifications and receipts for transactions.

The system is developed using the C programming language, utilizing libraries such as <conio.h>, <windows.h>, and standard I/O libraries for handling user interactions and system operations.

# Problem Analysis

Users Managing parking facilities manually presents several challenges:

1. **Inefficient Space Utilization:** Without a systematic approach, it's difficult to maximize the usage of available parking spaces.
2. **Time-Consuming Operations:** Manual logging of vehicle entries and exits can lead to delays and increased waiting times.
3. **Error-Prone Record-Keeping:** Human errors in recording vehicle details can result in data inconsistencies and difficulties in tracking.
4. **Lack of Real-Time Information:** Administrators and users lack immediate access to current parking availability, leading to frustration and inefficiency.
5. **Difficulty in Managing Waiting Lists:** Manual handling of overflow situations when parking areas are full is cumbersome and unreliable.

The Parking Management System addresses these issues by automating key processes, ensuring accurate record-keeping, and providing real-time information to both administrators and users.

.

# Solution Design (Project Detail, Functionality, and Features)

* + The Parking Management System is structured around two primary user roles: Admin and User. Below are the detailed functionalities and features designed to meet the project specifications:

**Admin Module:**

* + **Login Authentication:**
  + Secure login mechanism with username and PIN verification to prevent unauthorized access.
  + **Create Parking Area:**
  + Allows admins to define the number of rows and columns in the parking area, effectively configuring the total number of parking spots.
  + **View Parking Area:**
  + Displays a visual representation of the parking area, indicating occupied and available spots.
  + **Empty Parking Area:**
  + Enables admins to clear all parked vehicle data, resetting the parking area to its initial state.
  + **View Parked Vehicles List:**
  + Provides a comprehensive list of all currently parked vehicles, including details such as type, plate number, parking address, charges, and time of parking.
  + **Search Vehicle:**
  + Facilitates searching for a vehicle by its plate number, displaying relevant details if found.
  + **View History:**
  + Maintains a historical log of all unparked vehicles, capturing data like parking and unparking times, charges, and vehicle details.
  + **Generate Report:**
  + Generate reports including profit calculation, parking area details, and Vehicles History

**User Module:**

* + **Park Vehicle:**
  + Enables users to park their vehicles by entering vehicle type, plate number, and desired parking duration.
  + Offers manual or automatic parking options, assigning parking addresses accordingly.
  + Offers extra services according to user.
  + Manages waiting lists when parking areas are full, notifying users of their waiting status.
  + **Unpark Vehicle:**
  + Allows users to unpark their vehicles by entering the assigned parking address.
  + Calculates parking charges based on duration and vehicle type.
  + Generates receipts detailing parking and unparking times, duration, and total charges.
  + **View Parking Area:**
  + Provides users with a real-time view of parking availability, highlighting occupied and vacant spots.
  + Notify about Waiting List for Vehicles

### System Architecture

The system employs a matrix-based approach to represent the parking area, with rows and columns corresponding to physical parking spots. Each spot is uniquely identified by an address derived from its row and column indices. The system maintains arrays to track parked vehicles, waiting lists, and historical records. File I/O operations are utilized to persist data across sessions, ensuring that parked vehicles are saved and loaded appropriately.

### User Interface

The console-based interface provides intuitive navigation through menus and options. Color-coded outputs enhance readability, distinguishing between different statuses such as occupied spots (red) and available spots (green). Animations and sound notifications (beeps) are incorporated to improve user engagement and feedback.

# Implementation & Testing

**Implementation**

The system is implemented in C, leveraging various standard libraries for functionality:

* **Input/Output Handling:** <stdio.h> for standard input and output operations.
* **Console Manipulation:** <conio.h> for capturing keystrokes without echoing them to the console.
* **Windows-Specific Functions:** <windows.h> for setting console colors and sound notifications.
* **Time Management:** <time.h> for recording parking and unparking timestamps.
* **File Operations:** <stdlib.h> and <string.h> for dynamic memory management and string manipulations.

Key functions include:

* **loadVehicles() and saveVehicles():** Handle data persistence by reading from and writing to text files.
* **parkVehicle() and unparkVehicle():** Manage the core functionalities of parking and unparking vehicles, including charge calculations and waiting list management.
* **adminLogin() and adminMenu():** Facilitate secure access and administrative operations.
* **userMenu():** Provides user-specific functionalities for managing parking transactions.
* **displayParkingArea():** Renders the parking layout visually on the console.
* **notify() and animate():** Enhance user experience with notifications and animations.

**Testing**

Comprehensive testing was conducted to ensure the reliability and accuracy of the system:

* **Unit Testing:**
  + Each function was individually tested with various inputs to validate correct behavior and error handling.
* **Integration Testing:**
  + Combined modules (e.g., admin and user functionalities) were tested to ensure seamless interaction and data consistency.
* **User Acceptance Testing:**
  + Simulated user interactions were performed to verify the intuitiveness and responsiveness of the interface.
* **Edge Case Testing:**
  + Scenarios such as full parking areas, maximum waiting list capacity, and invalid inputs were tested to assess the system's robustness.

**Project Breakdown Structure (Workload Distribution with Timeline)**

|  |  |  |  |
| --- | --- | --- | --- |
| Phase | Tasks | Duration | Responsible |
| Planning | Requirement Analysis, Project Planning | 1 Week | Abdul Ahad |
| Development | Coding of Modules (Login, Tools, UI) | 3 Weeks | Abdul Ahad+Aayan + Hur Ali |
| Testing | Unit Testing, Integration Testing , Final Review | 1 Weeks | Abdul Ahad |
| Documentation | Writing Project Report, User Manual | 2 days | Syed Aayan Mahmood |
| Design | System Architecture, UI Design | 5 days | Hur Ali |

**Task Assignments**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **(Project Distribution with Timeline)**   |  |  |  | | --- | --- | --- | | Task | Duration | Responsible | | Admin Function | 7 days | Syed Aayan Mahmood+ Syed Hur Ali | | Customer | 7 days | Abdul Ahad | | Designing | 7 days | Syed Hur Ali | | Filing | 4 days | Syed Aayan Mahmood + Abdul Ahad | | Testing | 1 week | Abdul Ahad + Syed Aayan Mahmood + Syed Hur ALi | |  |

# Results (Output Screenshots)

**Welcome Screen**

**Login**

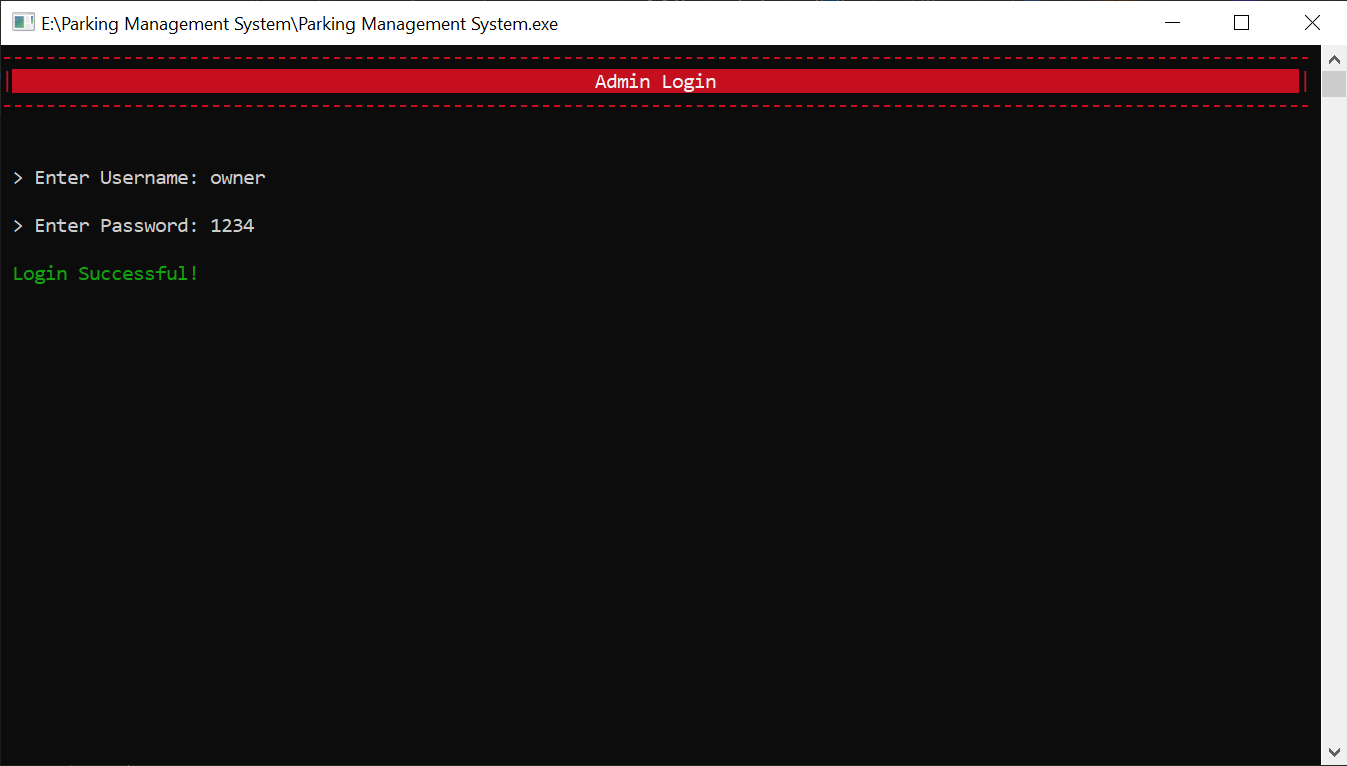
Displayed a menu option to login as owner or a user.

A screenshot of a computer

Description automatically generated

**ADMIN PANEL**

Display a login menu for Admin. If the password is wrong or username is wrong 3 times will return to login page.



A screenshot of a computer

Description automatically generated

**Display Admin Menu**

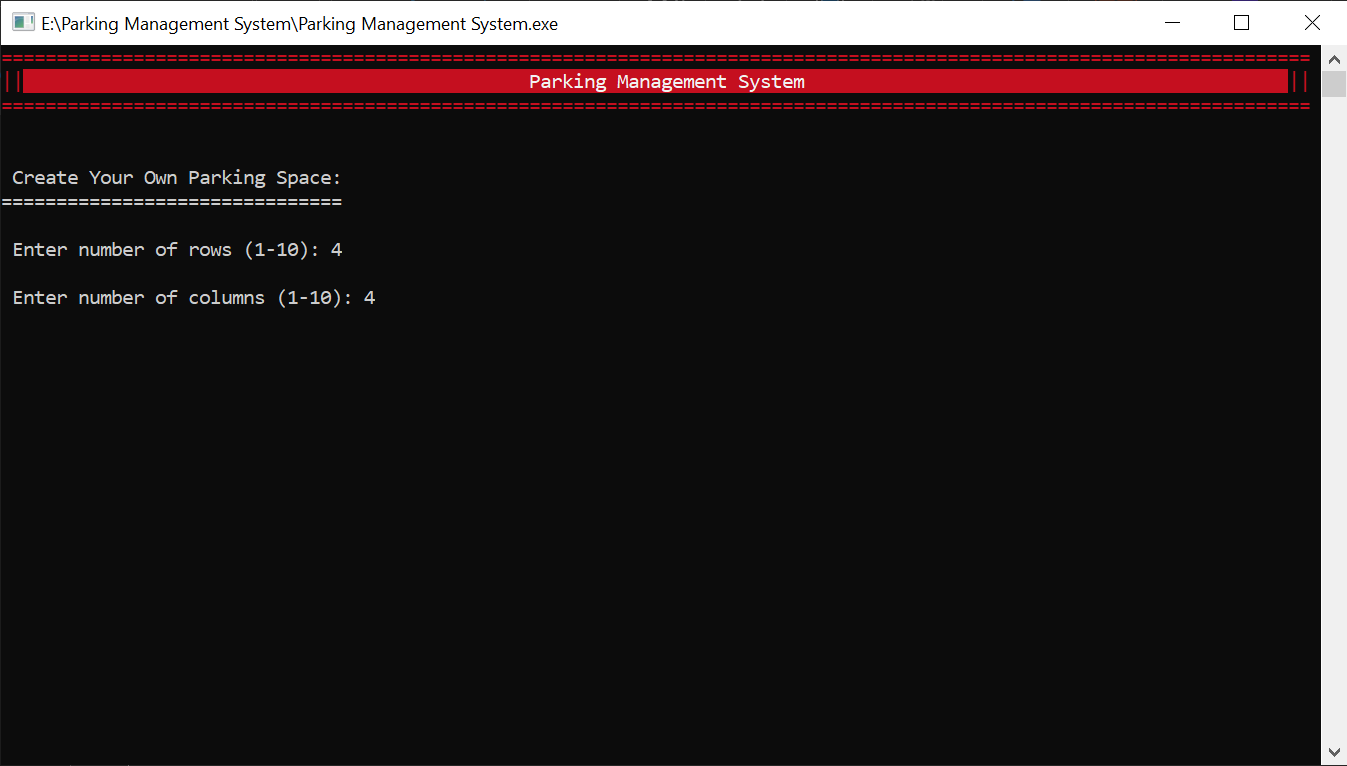
Presented a co menu listing all available options which Allows admin to select following options by entering corresponding numbers.

A screenshot of a computer

Description automatically generated

**Custom Parking Area**

Display a 2d Parking Area with options available for owner to make his own parking area .



A screenshot of a computer

Description automatically generated

**Delete a Parking Area**

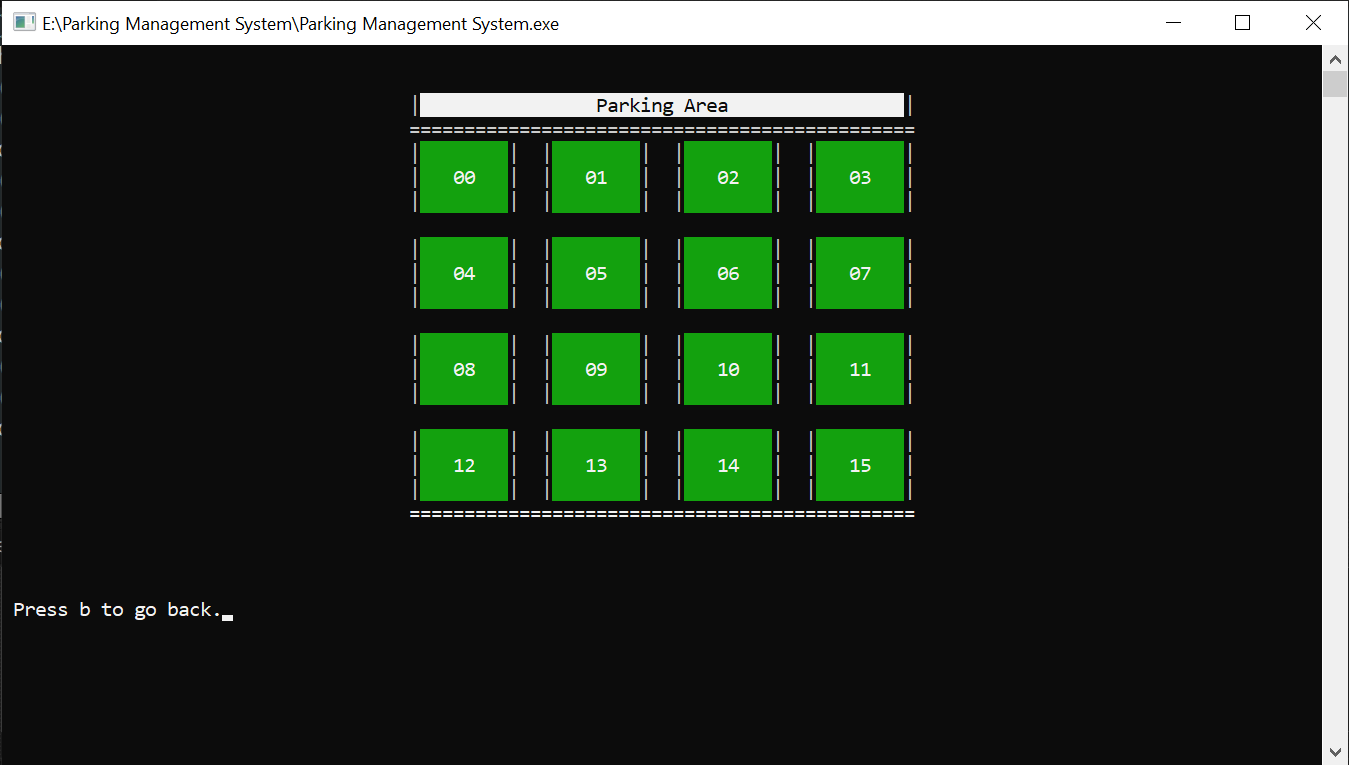
Delete a parking area made by owner if the user wants to.

A screenshot of a computer

Description automatically generated

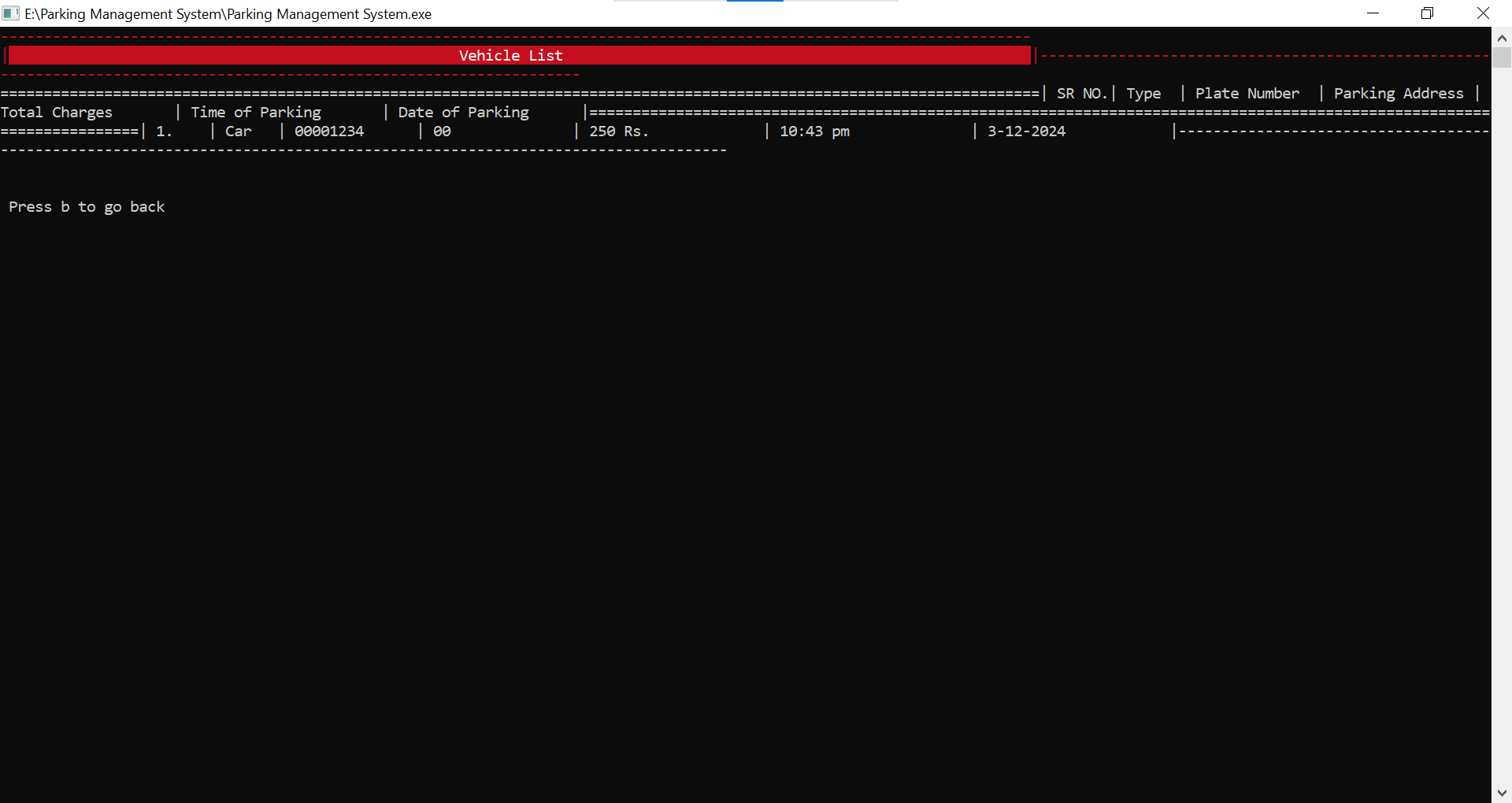
**View Parking Area:**

Display A Parking Area with parked cars being colored red if a car is park else green color will be shown

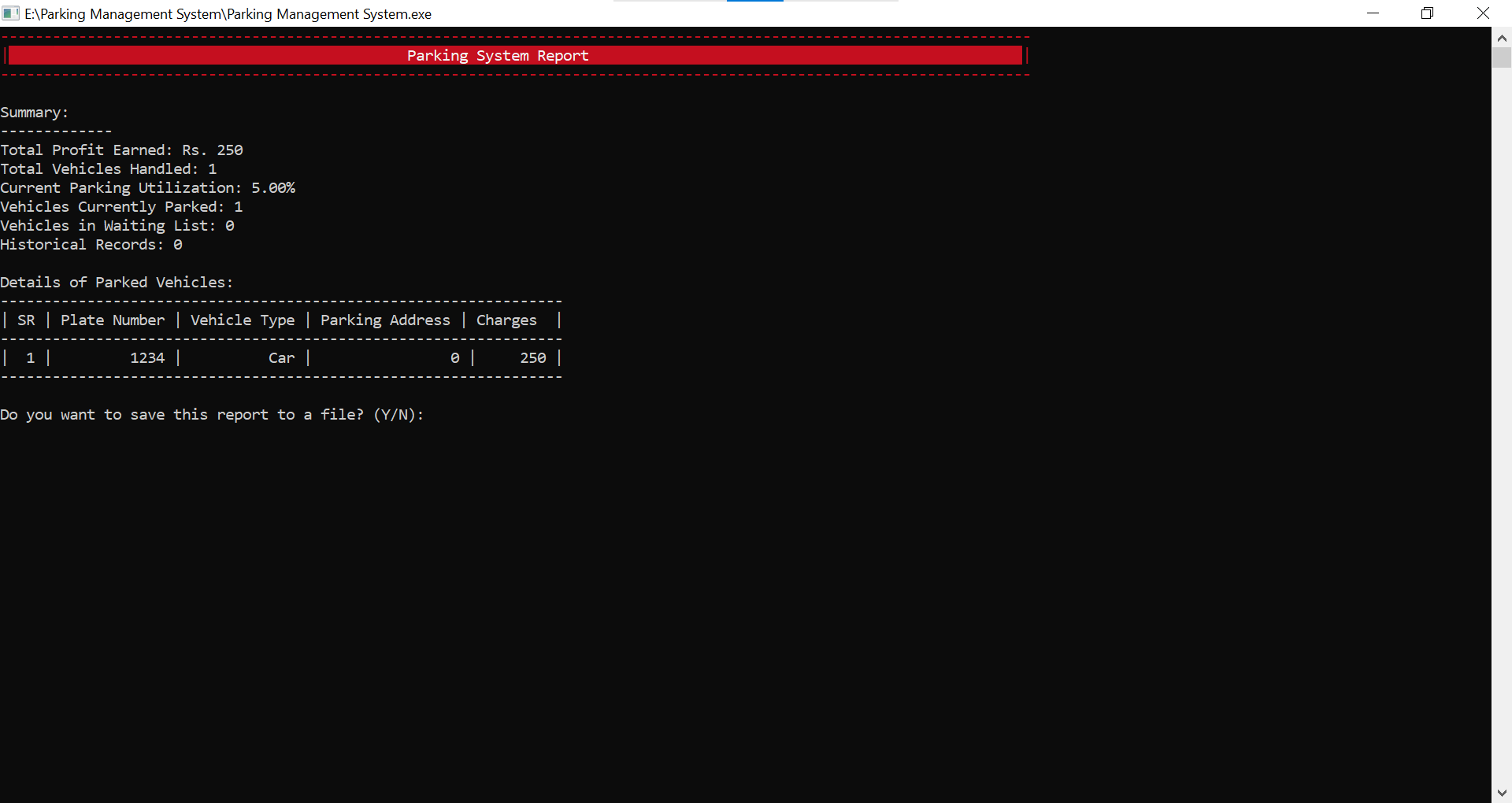


**Parked Vehicle History**

Display parked Vehicle history if a car is parked by the user

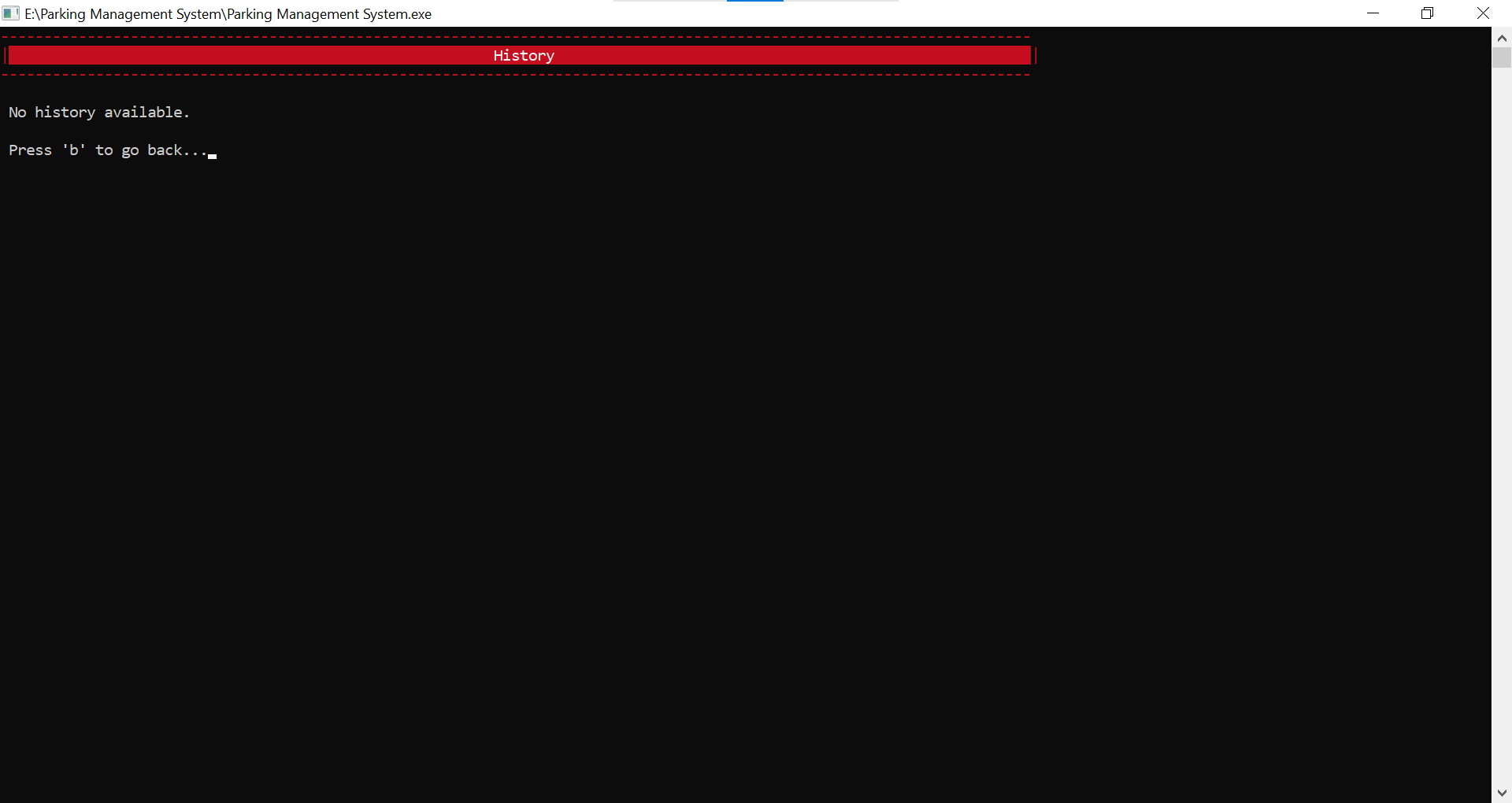


**Generate Parking Report**.



**History**

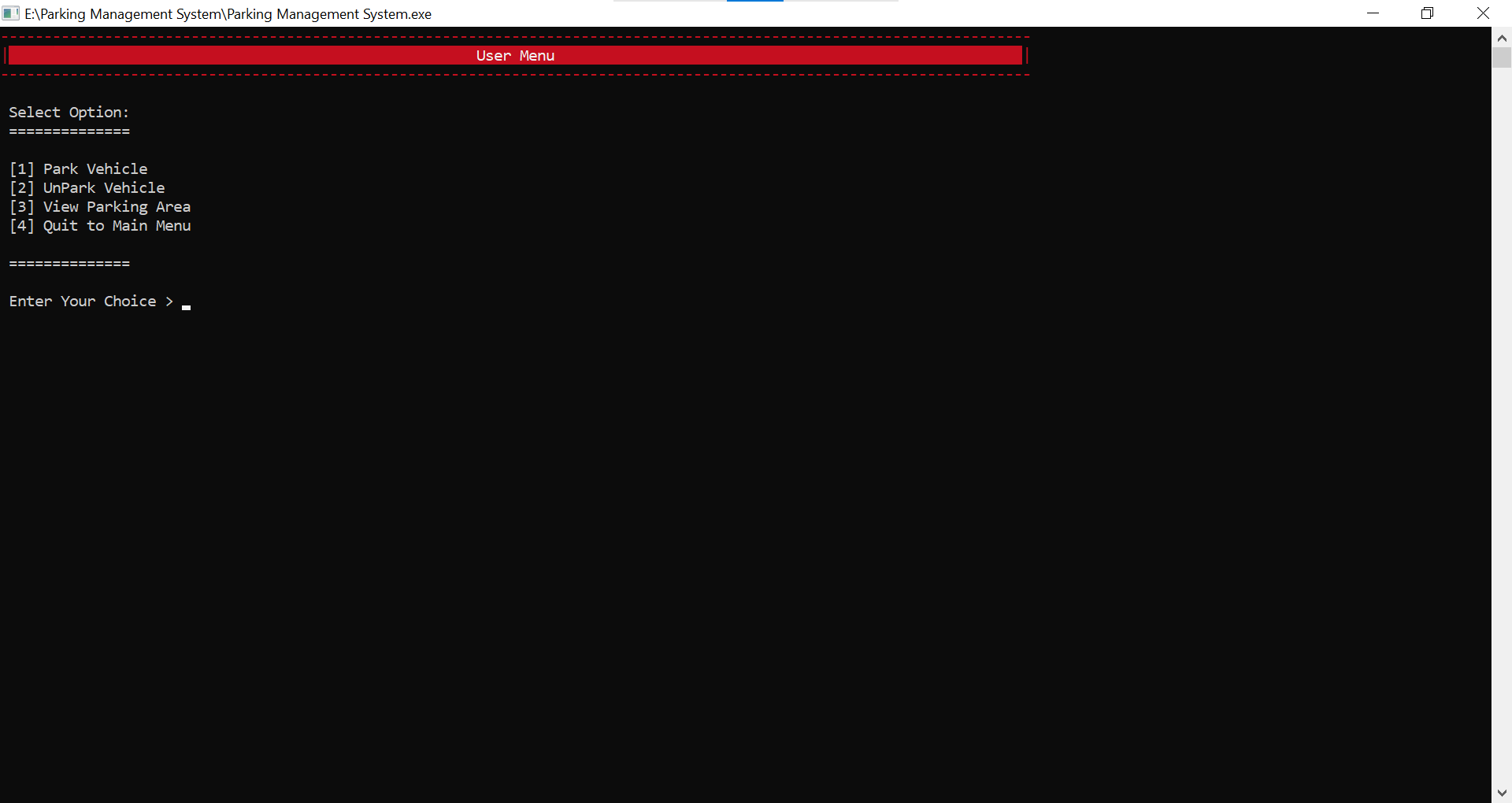
History Option to maintain the data for the unparked vehicle



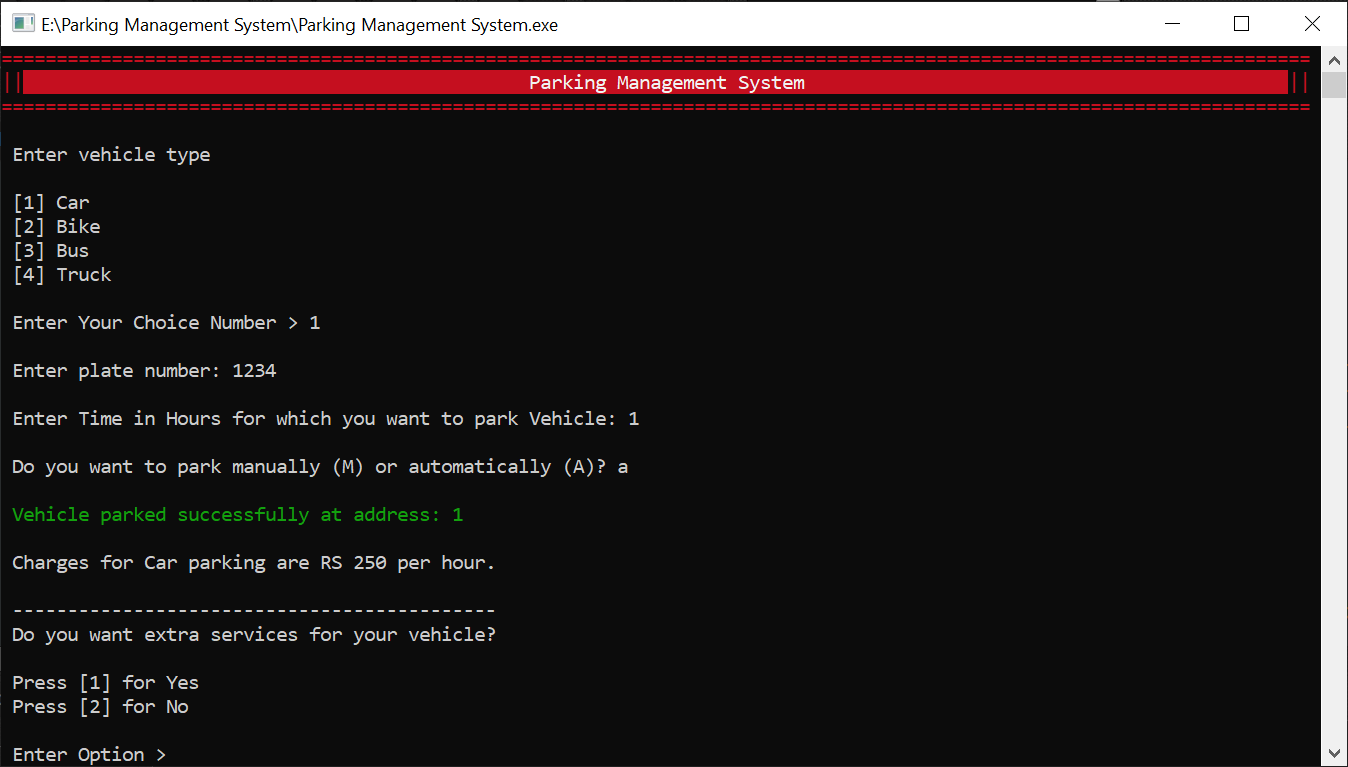
***USER***

**User Menu**

Options Available for User such as parked vehicles, unparked vehicles, view parking space, and exit option



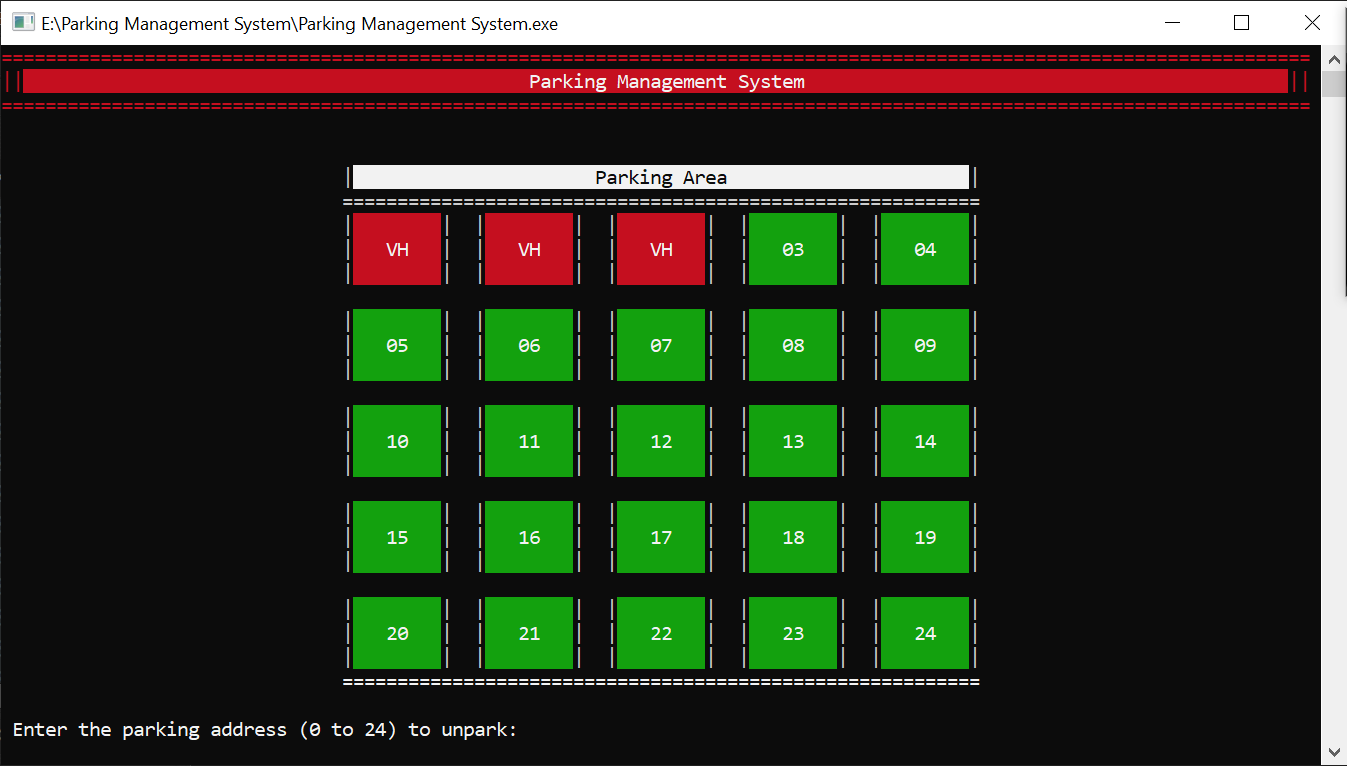
**Parked Vehicle Option**

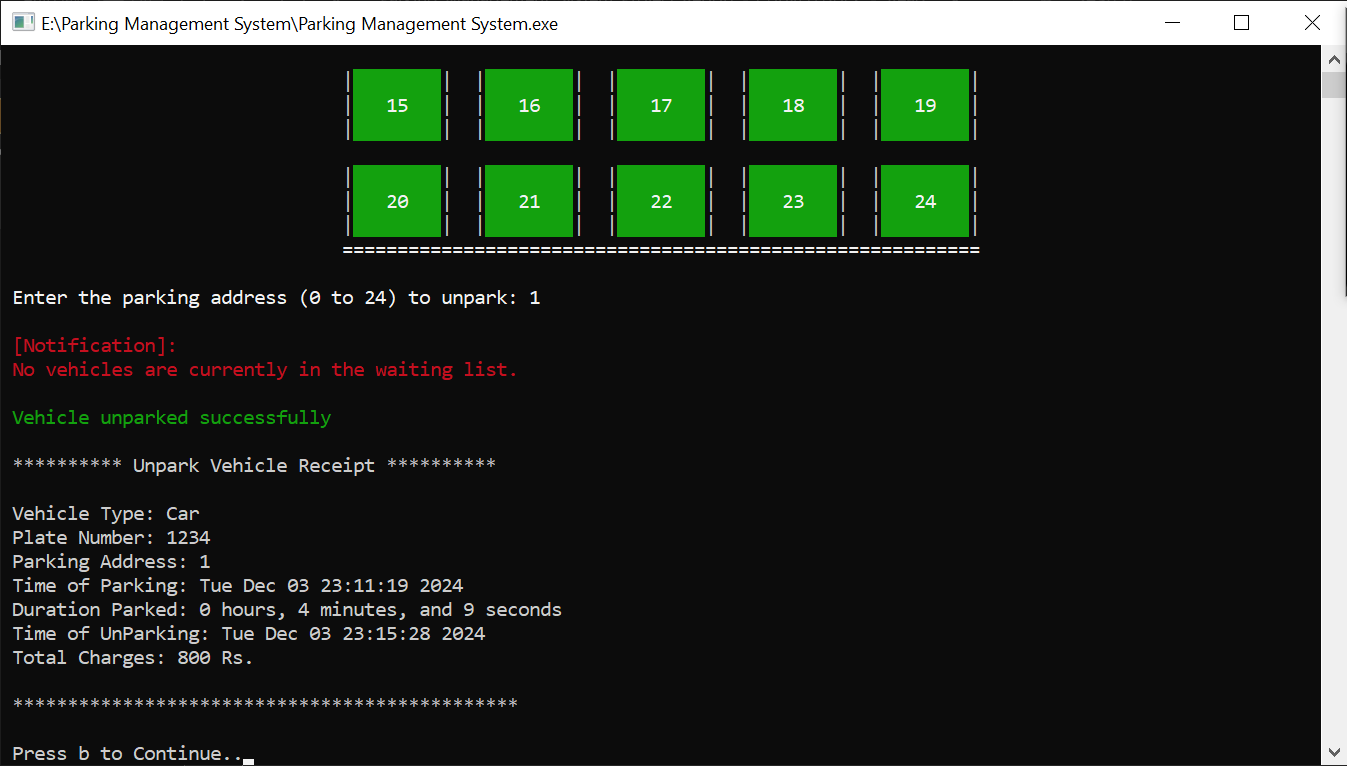
****

**A screenshot of a computer

Description automatically generated**

**Un park Option**

****

****

**View Parking Area**

****

# Conclusion (Summary & Discussion)

**Summary**

The **Parking Management System** project successfully delivers an automated solution for managing parking facilities, addressing the inefficiencies and challenges inherent in manual parking management. Developed using the C programming language, the system provides comprehensive functionalities tailored for both administrators and users, ensuring streamlined operations and enhanced user experience.

**Key Achievements:**

1. **Automated Parking Operations:** The system automates the processes of parking and unparking vehicles, significantly reducing the time and effort required compared to manual methods.
2. **Efficient Space Utilization:** By accurately tracking occupied and available parking spots through a matrix-based representation, the system optimizes the utilization of parking spaces, minimizing wastage and maximizing capacity.
3. **Real-Time Monitoring:** Administrators and users can access real-time information about parking availability, facilitating informed decision-making and reducing instances of overbooking.
4. **Robust Record-Keeping:** The system maintains detailed records of parked and unparked vehicles, including timestamps, charges, and vehicle details, ensuring accurate and reliable data management.
5. **User-Friendly Interface:** The console-based interface, enhanced with color-coded outputs and intuitive navigation, provides a seamless and engaging user experience for both administrators and users.
6. **Waiting List Management:** The implementation of a waiting list feature ensures that overflow situations are handled efficiently, preventing frustration and ensuring fairness in parking allocations.
7. **Data Persistence:** Utilizing file I/O operations, the system ensures that vehicle data is preserved across sessions, maintaining continuity and reliability in operations.

**Discussion**

The development and implementation of the Parking Management System have demonstrated the feasibility and effectiveness of automated solutions in addressing real-world challenges in parking management. Several aspects of the project warrant further discussion:

1. **Technical Challenges:**
   * **Concurrency Handling:** While the current implementation handles single-user interactions effectively, scaling the system to support multiple concurrent users would require enhancements such as mutexes or semaphores to manage data consistency.
   * **Error Handling:** Although basic error handling mechanisms are in place, incorporating more robust validation and exception handling would enhance the system's resilience against unexpected inputs and scenarios.
2. **User Experience Enhancements:**
   * **Graphical User Interface (GUI):** Transitioning from a console-based interface to a GUI would significantly improve user interaction, making the system more accessible and visually appealing.
   * **Mobile Integration:** Developing a mobile application interface would provide users with greater flexibility and accessibility, allowing them to manage parking operations on-the-go.
3. **Scalability and Performance:**
   * **Large-Scale Deployments:** For larger parking facilities, optimizing the system to handle increased data volumes and ensuring rapid response times would be essential. Implementing more efficient data structures and algorithms could address these scalability concerns.
   * **Integration with Hardware:** Integrating the system with parking sensors and automated gate controls would enable real-time occupancy tracking and automated access management, further enhancing operational efficiency.
4. **Security Considerations:**
   * **Data Protection:** Ensuring the security of stored vehicle and user data is paramount. Implementing encryption and secure authentication mechanisms would protect against unauthorized access and data breaches.
   * **Access Control:** Refining the role-based access control system to include more granular permissions would enhance the security framework, allowing for differentiated access levels based on user roles.
5. **Future Enhancements:**
   * **Payment Integration:** Incorporating online payment gateways would facilitate seamless financial transactions, enabling users to pay for parking services electronically.
   * **Analytics and Reporting:** Developing analytical tools and reporting features would provide administrators with valuable insights into parking patterns, revenue generation, and operational efficiency, aiding in strategic decision-making.
   * **Notification Systems:** Implementing automated notification systems via SMS or email would keep users informed about parking statuses, waiting list positions, and payment confirmations, enhancing user engagement and satisfaction.

**Conclusion**

The Parking Management System project has successfully demonstrated the potential of software solutions in revolutionizing traditional parking management practices. By automating key operations, ensuring efficient space utilization, and maintaining robust records, the system offers a reliable and user-friendly platform that addresses the core challenges of parking management.

While the current implementation lays a solid foundation, there are numerous opportunities for further enhancements and scalability to meet the evolving demands of modern parking facilities. Future developments focusing on user experience, security, scalability, and integration with emerging technologies will propel the system towards becoming a comprehensive solution for intelligent parking management.

In summary, the project not only fulfills its initial objectives but also paves the way for innovative advancements in the domain of parking management, contributing to smarter and more efficient urban infrastructure.

# Acknowledgements

We extend our gratitude to our instructor, (YOUR INSTRUCTOR NAME), for providing invaluable guidance and support throughout the development of this project. Additionally, we acknowledge the contributions of our group members:

**Abdul Ahad (Project Leader):** For his unwavering leadership, guidance, and continuous support throughout the course of this project. His expertise in project management and technical aspects was invaluable in steering this project to completion.

**Syed Aayan Mahmood & Syed Hur Ali (Team Members):** For their hard work, dedication, and collaborative efforts in the development of the Parking Management System. Their contributions to coding, system testing, and problem-solving were crucial to the project's success. for their dedication and collaborative efforts.

# Contact Information

**Abdul Ahad**  
Email: K24-0733@nu.edu.pk

**Syed Aayan Mahmood**

Email: k240642 [@nu.edu.pk](mailto:0615@nu.edu.pk)

**Syed Hur Ali**

Email: k24 0615 @nu.edu.pk