Traffic Management System with C++

The Traffic Management System with C++ is a software solution aimed at automating and regulating traffic control procedures. It incorporates a multitude of functionalities such as challan recording, vehicle, and challan searches, traffic flow monitoring across various control booths, and provision of emergency contact and healthcare information. We will be utilizing C++ for file handling and data management. Our traffic system will efficiently automate tasks, demonstrating the application of programming in real-world scenarios like traffic management.

Introduction

The Traffic Management System is concerned with the process of managing traffic. We will learn how to provide effective storage of vehicle registration numbers and make data retrieval and access easy and fast. The user can do a variety of operations, such as accessing data records, searching for a vehicle and owner, etc.

Objectives

The objectives of this traffic management system with C++ are as follows:

1. Traffic Violation Management: The code provides an interface for recording and managing traffic violations (challans) linked with vehicle registration numbers and owner names.

2. Traffic Challan Search: The program offers functionality to search for traffic challans using either the vehicle's registration number or the vehicle owner's name.

3. Vehicle Search: The system enables search operations for vehicle records based on their registration numbers, aiding in tracking and monitoring.

4. Traffic Control Booth Monitoring: The code simulates the operation of traffic control booths in different locations, tracking the number of vehicles entering and leaving the city.

5. Provision of Emergency Information: The system provides important emergency contact information and details about nearby healthcare centers, serving as a useful resource during emergencies.

Requirements

1. Integrated Development Environment (IDE): A C++ compatible IDE like **Dev C++**, **Code Blocks**, or **VS code** will be needed for writing and debugging the code.

2. C++ Standard Library: The code makes use of standard C++ libraries like <iostream>, <fstream>, and <string>. Ensure that your compiler fully supports the C++11 or later standard.

3. File System Access: The code uses **file handling** operations to store and retrieve information about traffic violations. The development and deployment environment should allow for file creation, reading, and writing.

4. Operating System Compatibility: The code is platform-independent, but your IDE and C++ compiler must be compatible with your operating system, be it Windows, MacOS, or Linux.

Explanation of the Code

There are certain functions to record the details of vehicles and store the owner's name, search for a vehicle with its registration number, etc. Here's a breakdown of the functions and what each one does: All the functionalities are performed properly without any inconvenience.

1. class TrafficManagementSystem: This class encapsulates all the functionality of the Traffic Management System.

2. welcome(): This is the main menu of the application. It displays a welcome message along with the date and time, then presents the user with a menu of options: to record new vehicles, get records of challan (traffic violation fines), search for a vehicle's record, search for traffic control booths, control traffic, and access a help menu.

3. delay_0(), int delay_1(), int delay_2(): These three methods are used to introduce delays of varying lengths in the program for visual effect. They use the this_thread and chrono namespaces to achieve this.

4. recordOfVehicle(): This function displays a submenu for recording and searching for vehicles. It offers options to add a new vehicle, search for a vehicle using its registration number, and search for a vehicle by the owner's name.

5. recordOfVehicle_1(): This function lets the user add a new vehicle's registration number and owner's name to a text file.

6. recordOfVehicle_2(): This function allows the user to search for a vehicle by its registration number in the text file. If the registration number is found, it gives the user the option to either return to the home screen or search again.

7. recordOfVehicle_3(): Similar to recordOfVehicle_2(), but this function searches for a vehicle by the owner's name. If the name is found, it gives the user the option to either return to the home screen or search again.

8. recordOfChallan() function is the starting point of the program. It creates a simple console-based user interface for a traffic management system. This function offers four options to the user: add a new challan, search for a challan using the registration number, search for a challan using the owner's name, and to go back to the home screen. Depending on the choice entered by the user, it calls one of three other functions or returns to the home screen.

9. recordOfChallan_1() allows the user to add a new challan by inputting a vehicle's registration number and the owner's name. This information is appended to a text file. If the user wants to exit this function, they can enter './'.

10. recordOfChallan_2() searches for a challan using the registration number of the vehicle. It opens a text file and checks each word to see if it matches the registration number entered by the user. If a match is found, the function displays a success message. If no match is found, it displays a failure message.

11. recordOfChallan_3() functions similarly to recordOfChallan_2(), but it searches for a challan using the owner's name instead of the vehicle's registration number.

12. vehSearch() allows the user to search for a vehicle's record using its registration number.

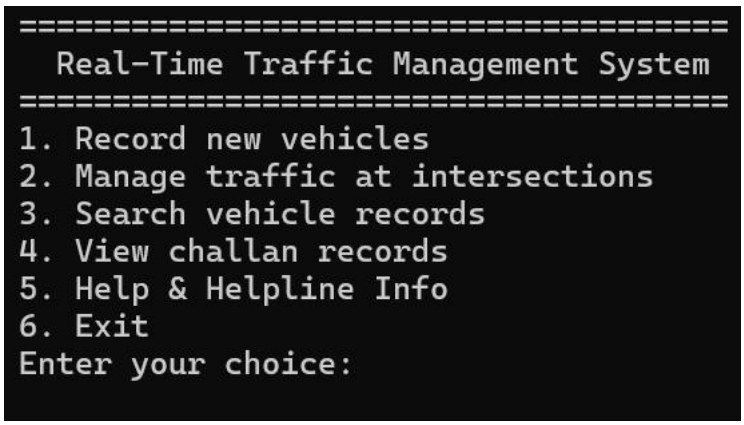
13. trafContBooth() and related **trafficContBooth_x()** functions simulate the output of three different traffic control booths. Each booth has a different way of calculating the number of vehicles coming in and going out.

14. helpInfo() provides helpline information and details about nearby healthcare centers. It allows users to view the helpline number or get info about hospitals in Cuttack or Puri. This information is read from text files.

15. TrafficManagementSystem is a class that contains all the above functions.

16. Finally, the **main()** function creates an instance of the TrafficManagementSystem class and calls the **welcome()** function to start the program.

Output



Conclusion

This Traffic Management System with C++ is a great example of how C++ can be used for file handling and data management to build a practical, real-world application. This is a very simple, automated, and convenient C++ system that helps the user to access numerous vehicle records as per need. It shows how programming can help in automating tasks in various fields, including traffic management.