Software Requirements Specification
Version 1.0
December 2024

Logistics Management System

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to present a detailed description of a transport logistics system for a company called Swift Logistics. It will explain the purpose, featu... of the system and what the system will do.

1.2 Scope of the project

The system is generally a management system as it is

designed to allow for addition, display and storage of vehicles and customers in files. The system also allows for the retrieval of the data from saved files.

This system aims to enhance and assist the Swift Logistics company's operations through providing an organized and efficient manner to manage their fleet and customer data.

1.3 Definitions

- Vehicle:
A representation of the transport vehicles.
- Car:
A type of vehicle with seating capacity.
- Truck:
A type of vehicle with a load capacity.
- Customer:
A representation of a client of the company.

1.4 References

- Object-Oriented Programming Concepts
- C++ Standard Library Documentation for file I/O and exception handling.

1.5 Overview of Document

The following sections of this document will give an overview of the software product. It will describe the functionality and requirements of the system whether functional or non-functional. It will also give descriptions of the features and design.

2. System Description

2.1 Functional Requirements

1. Vehicle Management

- Add Vehicle

The system should allow for addition of new vehicle whether car or truck.

- Display Vehicle

The system should show a list of all vehicles stored in memory.

- Save Vehicle

The system should allow for saving of vehicle details to a file.

- Load Vehicle

The system should allow for loading vehicle details from a file.

2. Customer Management

- Add Customer

The system should allow addition of a customer with name and contact information.

- Display Customer

The system should show a list of all customers.

- Save Customer

The system should allow for saving of customers details to a file.

- Load Customer

The system should allow for loading a customer details from a file.

3 Error Handling

It can handle file reading and writing errors. It can also handle invalid inputs from users to interact with the software.

2.2 Non-Functional Requirements

1. Performance:
File I/O and data processing should complete in real-ti
for typical use cases.
2. Usability:
The system should be easy to use and provide error
messages.
3. Portability:
The system should be compatible with any platform that
supports a C++ compiler.
4. Modularity
The system should be organized into modules for eas
updates and extensions.
5. Scability
The system should ~~many~~ large amounts of vehicle and
customer records efficiently.

3. System Features

3.1 Add Vehicles

- Users can add either a car or a truck where they
input model of the vehicle, registration number and
seating / payload capacity.

3.2 Display Vehicles

- Lists all the vehicles present in the system with all
their details.

3.3 Save Vehicles to file

- It saves the list of vehicles to a specified file.

3.4 Load Vehicles from file

- It loads vehicle data from the user-specified
file.

3.5 Add Customer

- Users can be able to add new customers with the

name and contact number

3.6 Display Customers

- Lists all the customers present in the system with their names and contacts.

3.7 Save Customers to file

- It saves the list of customers to a user-specified file.

3.8 Load Customers from file

- It loads the customer data from a user-specified file.

3.9 Error Handling

- It displays error messages for invalid user inputs or file I/O errors.

3.10 Exit the system

- Allows users to exit the system making sure operations are completed.

4. System Design Description

4.1 Classes

Vehicle Base Class

The attributes are type, model, registration number.

Derived Classes

1. Car

It inherits from vehicle and adds the attribute of seating Capacity.

2. Truck

It inherits from vehicle and adds the attribute of pay-load Capacity.

Customer Class

It has the attributes of name of customer and contact details.

4.2 Operations

Vehicle

- The operations are save and load vehicles where the vehicles can be saved or loaded from a file using the save to file and load from file methods.
  They are serialized as CSV lines in the format:
  - Car, Model, RegistrationNumber, SeatingCapacity
  - Truck, Model, RegistrationNumber, PayloadCapacity
- There is also displaying of vehicle details using the display() method

Customer

- The operations are save and load customers from a file in CSV format.
- There is also displaying of customer details using the displayCustomers.

4.3 File Operations

The system saves using the following file format:

- Stored in vehicles.txt
- Stored in customers.txt.

The system checks for file accessibility and handles errors when loading invalid data.

5. Design Constraints

Some of the constraints of this system are:

- This system can only run on platforms which support C++.
- File formats are restricted to simple comma-separated values (CSV)
- The system has restricted classes and attributes

- It has numeric limits meaning invalid numbers cannot be handled such as negatives.

6. Appendices

6.1 Sample Data Format

- Vehicle

(vehicles.txt)

Car, Model X, CBC 483, 6

Truck, Node B, ADE694, 10

- Customers

(customers.txt)

Havana Smith, 0710968861

Accra Melvin, 0198668844