OBJECT ORIENTED PROGRAMMING (CS-212)

RailSync

A railway management project using OOPs

Team Members:
Sanjogita Bhagowati (2312087)
Agniv Kashyap (2312180)
Branch : Computer Science and Engineering (Sec A)

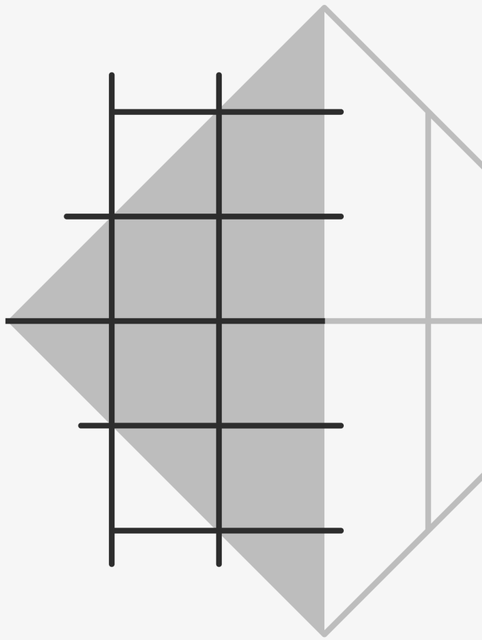
Overview

RailSync is a smart railway management system developed using C++ and Object-Oriented Programming (OOP) methodologies.

It provides a user-friendly, menu-driven interface that separates the functionality between:

- Admins: who manage trains, monitor bookings, and oversee food services.
- Users: who create accounts, book tickets, manage customer details, and place e-catering orders.

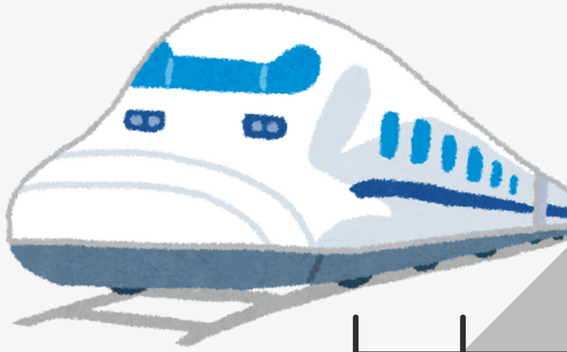
The system is file-based, lightweight, and modular, demonstrating practical applications of C++ features like encapsulation, inheritance, polymorphism, and abstraction.



The Challenge We Solved

How can we design a lightweight, modular, and extensible railway management system using Object-Oriented Programming (C++) that efficiently handles bookings, customer data, train schedules, and food services through structured file-based storage?

What RailSync Brings to the Table



RailSync offers a comprehensive and user-friendly system with two primary modes of operation:
Admin Mode and User Mode, along with an option to exit the application

WELCOME PAGE

Upon launching RailSync, users are presented with three options:

- Admin Mode
- User Mode
- Exit

What RailSync Brings to the Table

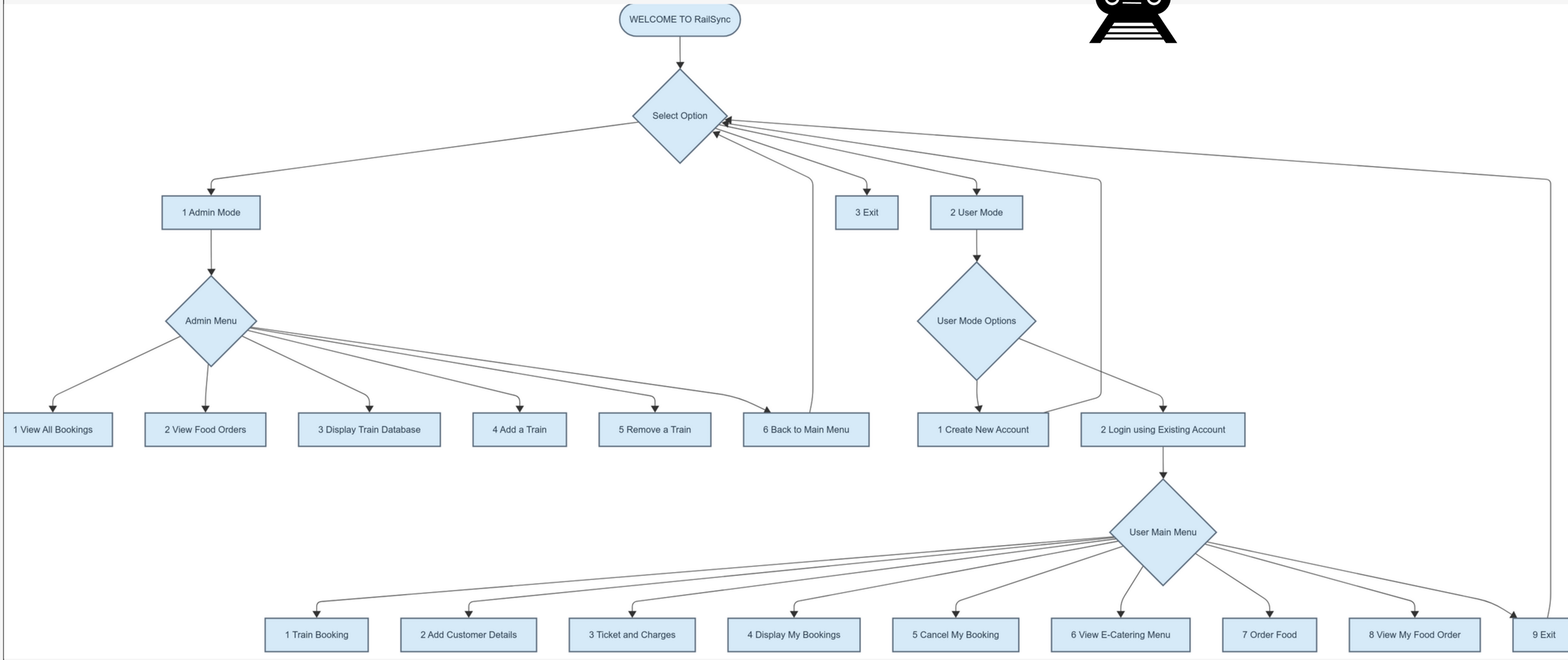
Admin Mode Workflow

- View All Bookings.
- View Food Orders
- Display Train Database
- Add a Train
- Remove a Train
- Back to Main Menu

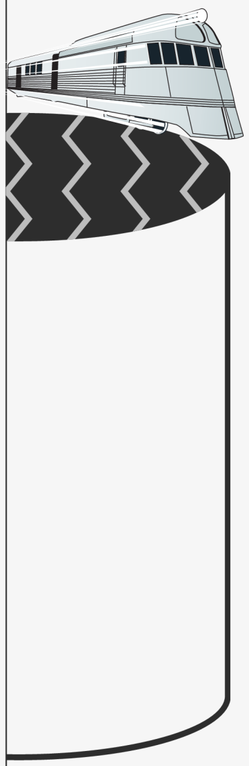
User Mode Workflow

- Create New Account
- Log in Using Existing Account
  - Train Booking
  - Add Customer Details
  - Ticket and Charges
  - Display My Bookings
  - Cancel My Booking
  - View E-Catering Menu
  - Order Food
  - View My Food Order
  - Back to Main Menu

Routing RailSync's Journey

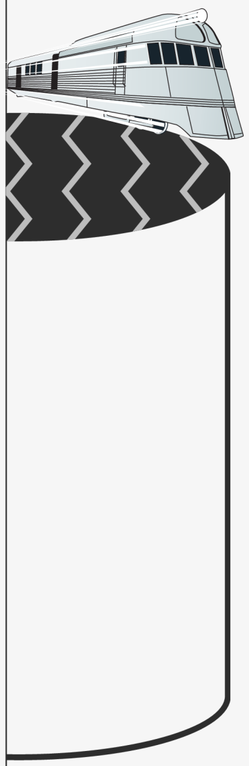


Powering RailSync: OOP in Action



File	OOP Concepts Applied	Key Benefits
main.cpp	Object Instantiation, Exception Handling	Controls flow via Management object; error handling prevents crashes during startup.
railway_system.h	Classes & Encapsulation, Constructors, Static Members, Inheritance	Structures railway entities; supports code reuse and modularity; efficient data management with static members.
station.cpp	Constructor, Encapsulation	Ensures valid initialization of Station objects; avoids uninitialized data issues.
train.cpp	Constructor, Exception Handling, Encapsulation	Validates train data during creation; prevents invalid schedule setups; robust input management.
ticket.cpp	Static Members, Inheritance, Exception Handling, Encapsulation	Shared booking data; reuse of customer fields; safe file operations; modular ticket management.

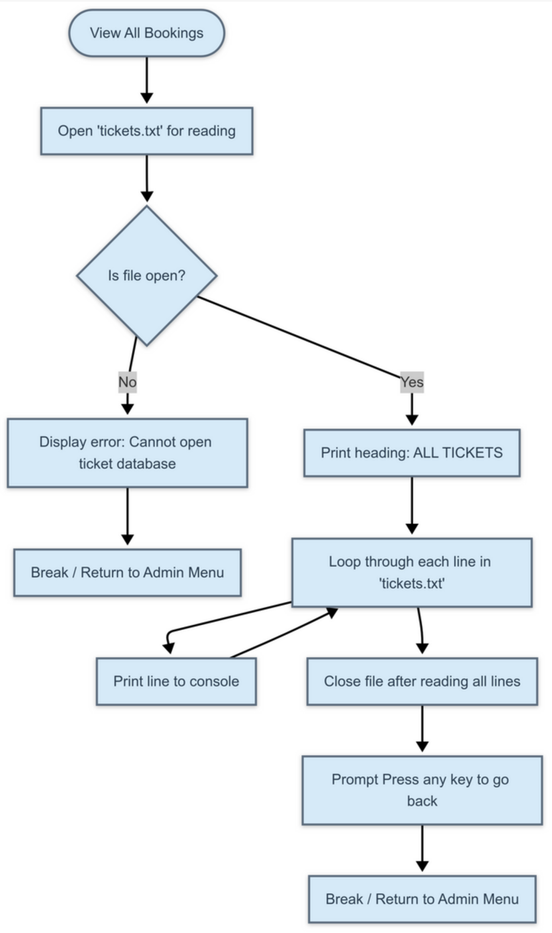
Powering RailSync: OOP in Action

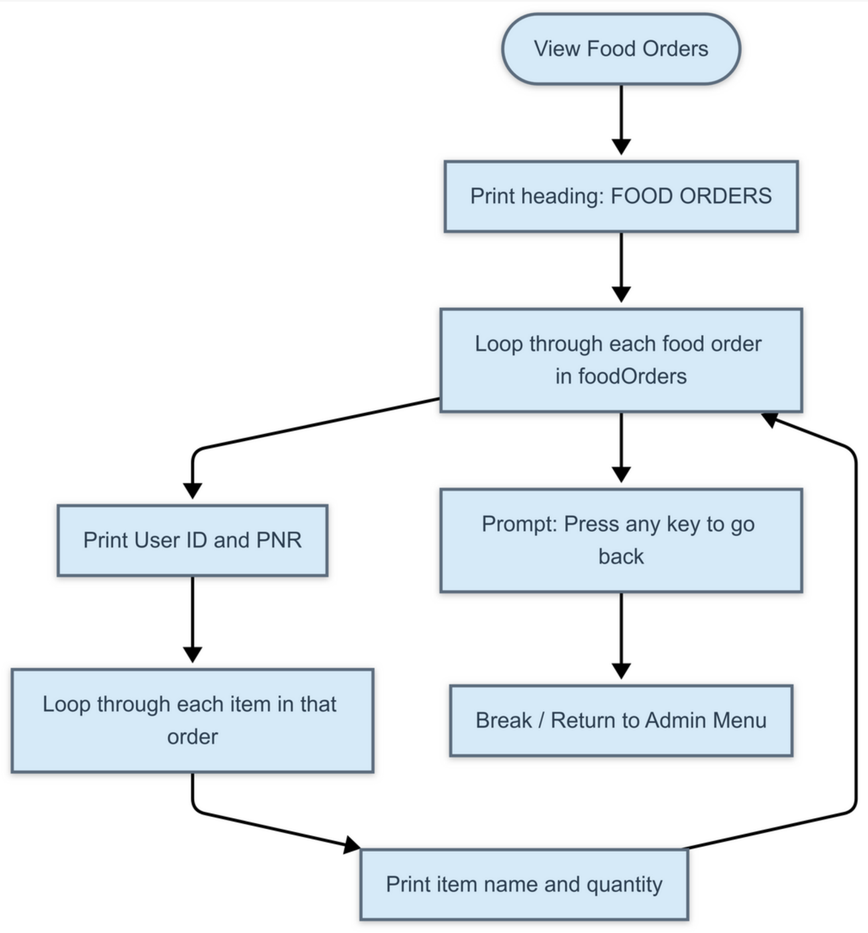


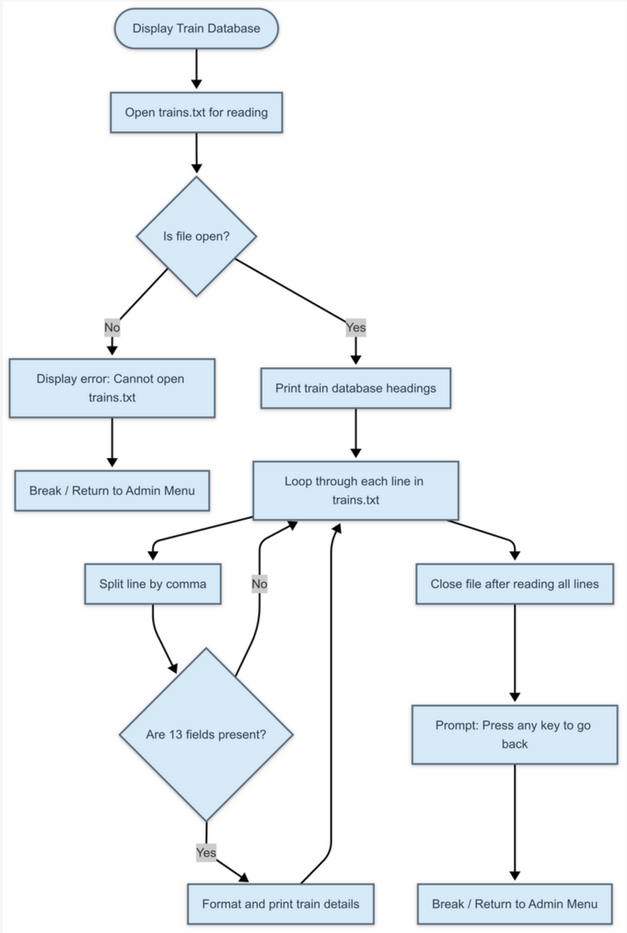
File	OOP Concepts Applied	Key Benefits
registration.cpp	Static Members, Inheritance, Polymorphism, Exception Handling, Encapsulation	Flexible train selection via interfaces; runtime polymorphism allows easy future expansion; error-proof booking process.
management.cpp	Composition (Aggregation), Static Members, Exception Handling	Manages overall system flow using instances of other classes; maintains correct user operations sequence; safe input handling.
database.cpp	Abstraction, Encapsulation	Hides file I/O details behind objects; clean, error-free station/train data loading; dynamic seat management.
order_food.cpp	Encapsulation, Constructor, Static & Global Data, Exception Handling	Manages food menu privately; organizes ordering logic; prevents invalid food orders by validating PNRs.

Core Logic Behind RailSync

Admin Features



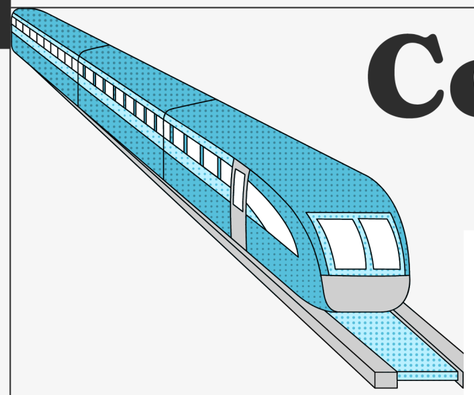




View All Bookings

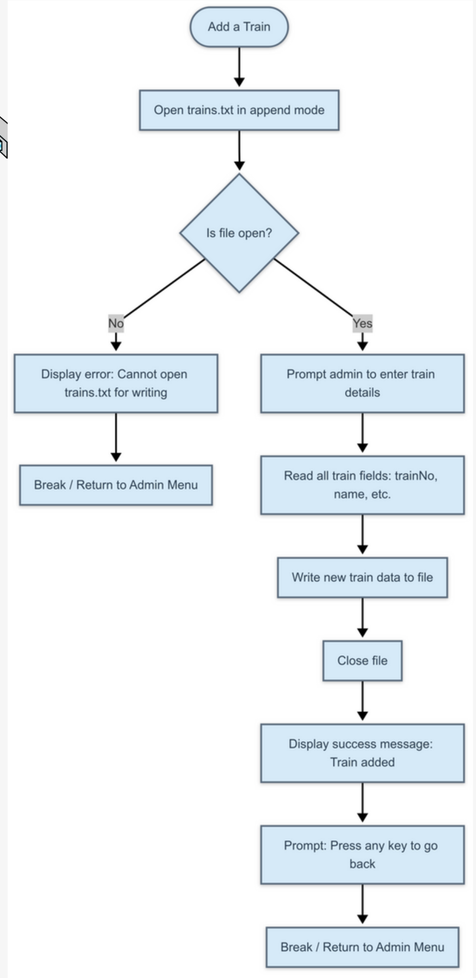
View Food Orders

Display Train Database

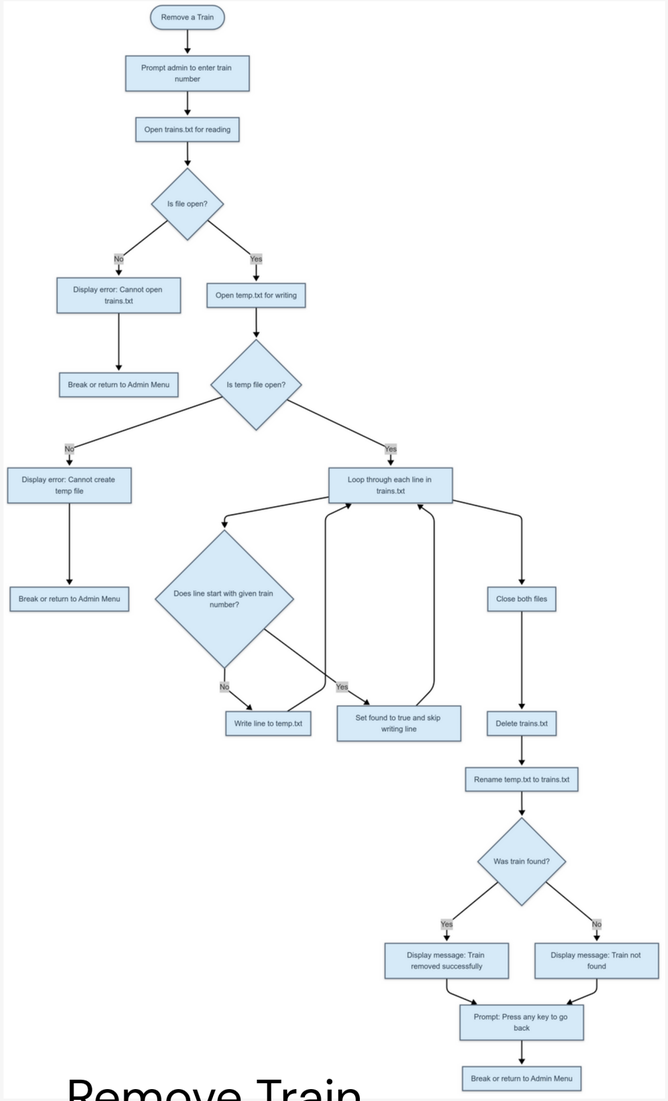


Core Logic Behind RailSync

Admin features



Add Train

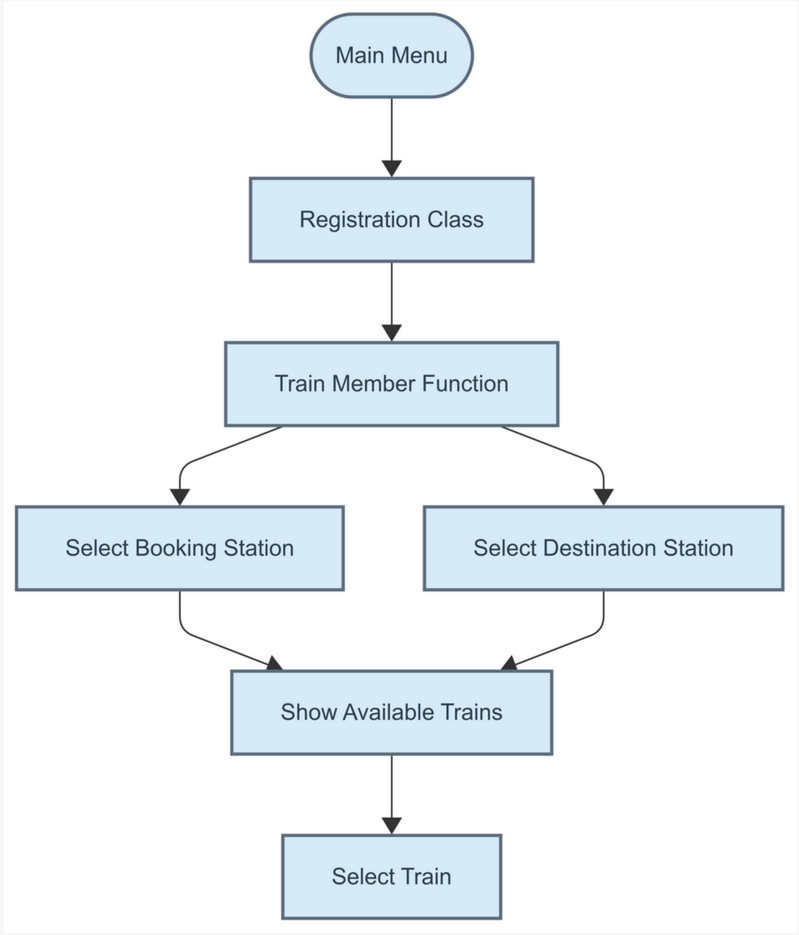


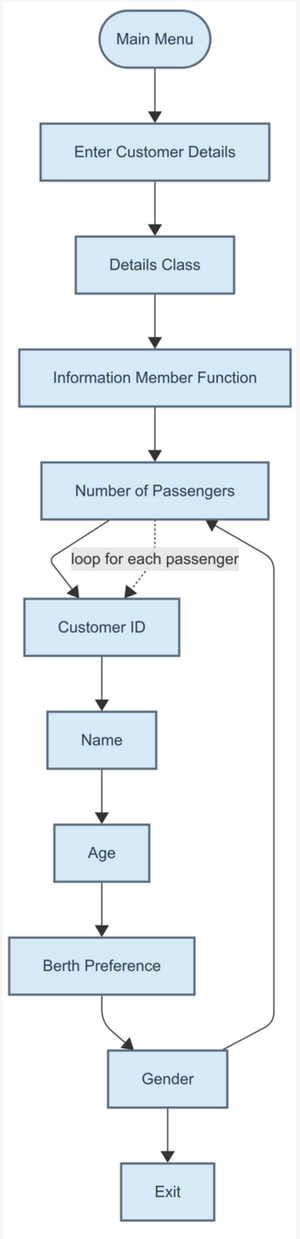
Remove Train

Core Logic Behind RailSync

User Features

Ticket Booking

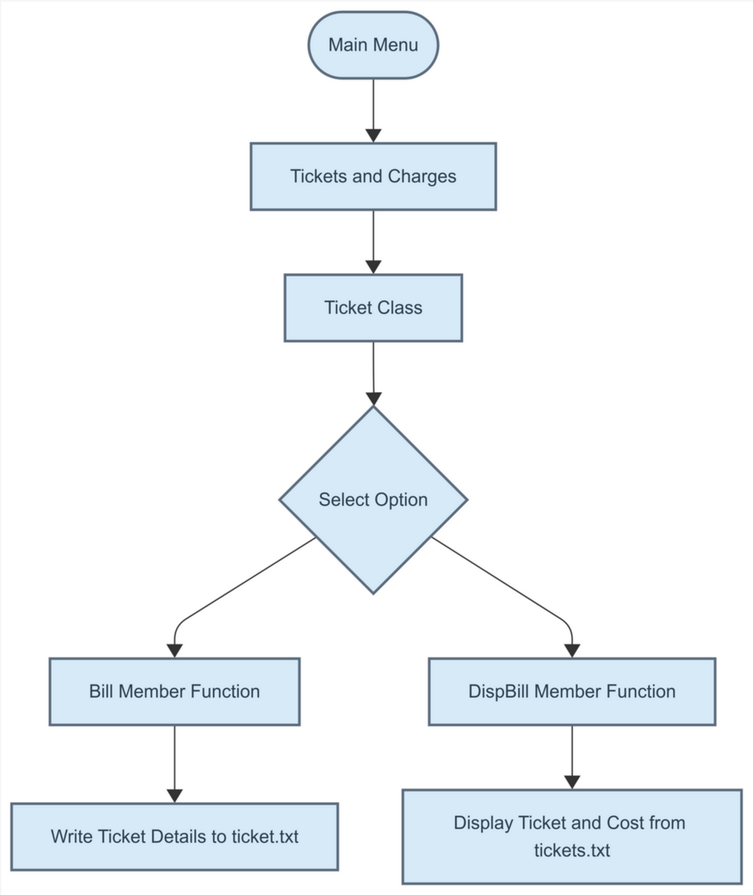




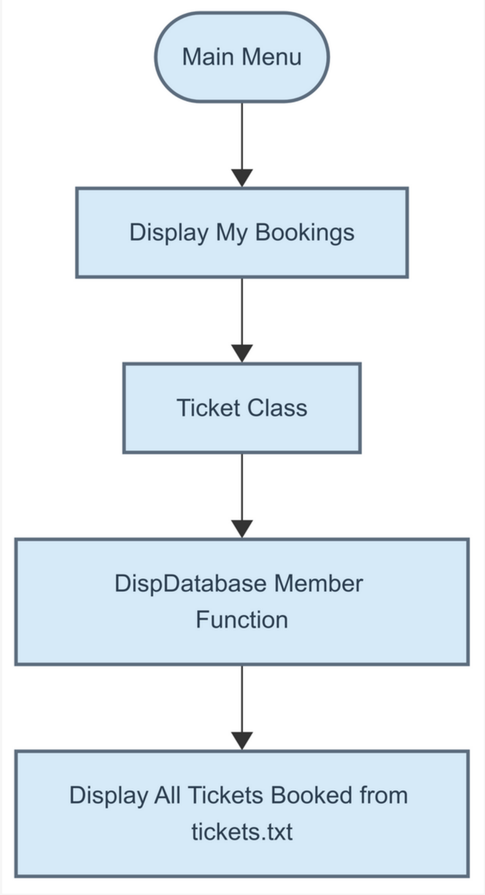
Add Customer Details

Core Logic Behind RailSync

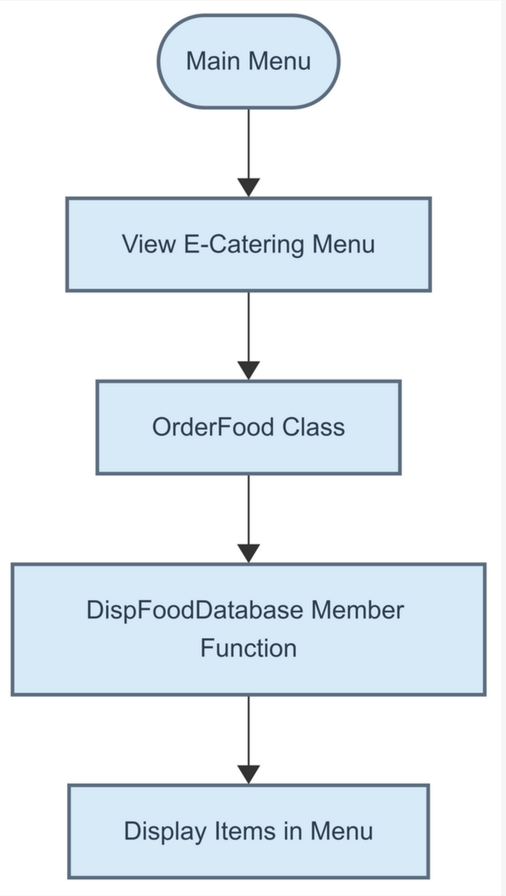
User Features



Ticket and Charges



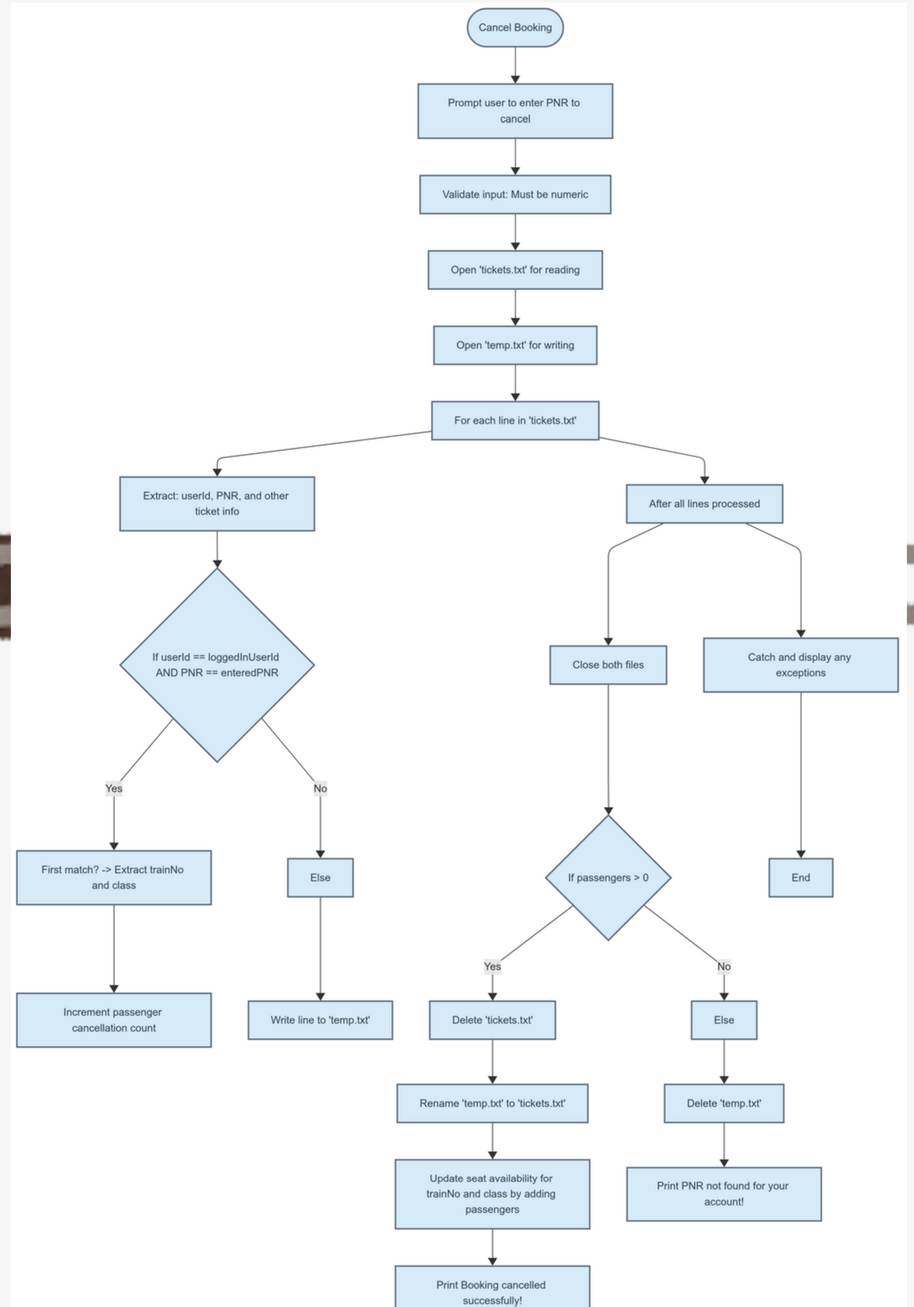
Display My Bookings



View E Catering Menu

Core Logic Behind RailSync

User Feature

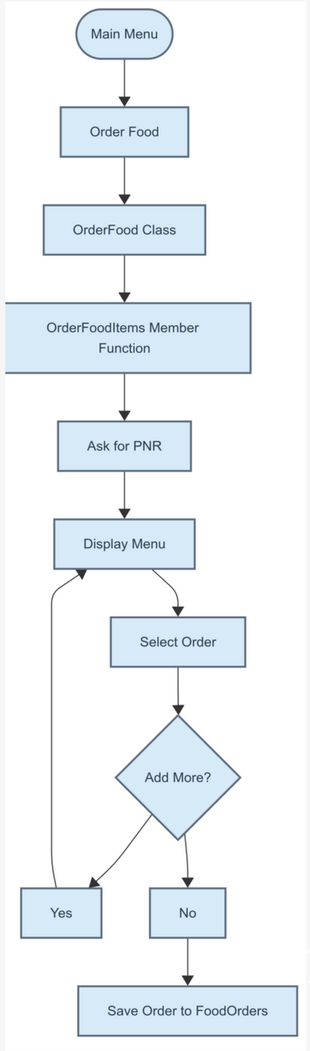


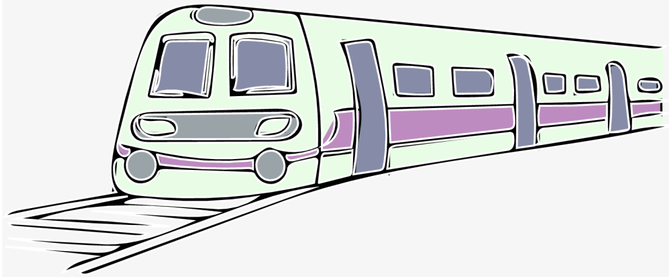
Cancel My Booking

Core Logic Behind RailSync

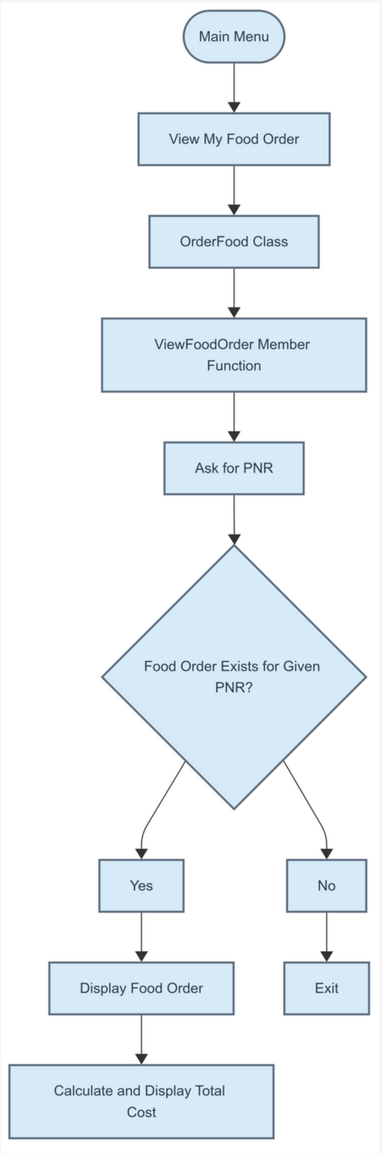
User Features

Order Food



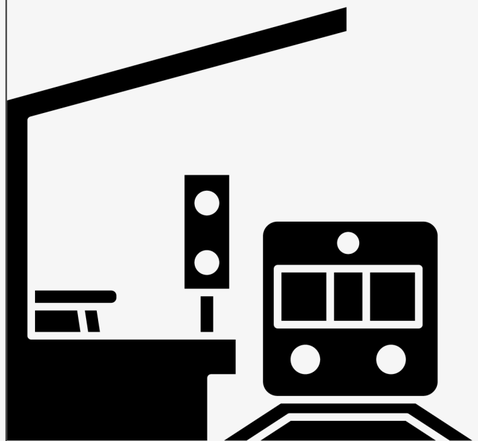


View My Food Order



On the Rails: What's Next?

- Transition to databases (SQLite, MySQL) for larger datasets.
- GUI (Qt, SFML) for better UX.
- Add payment integration for ticket bookings and food orders.



Thank you