Public Transportation Station Management System-Documentation

Name: Khong Dinh Tu

ID: 24110145

---

1. Object-Oriented Analysis (OOA)

Following the 4-step OOA model, the system has the following objects:

1.1 Objects

- **Vehicle**
- **Bus** (inherits from Vehicle)
- **Train** (inherits from Vehicle)
- **ExpressBus** (inherits from Vehicle)
- **Airplane** (inherits from Vehicle)
- **Ship** (inherits from Vehicle
- **Motorbike** (inherits from Vehicle)
- **Taxi** (inherits from Vehicle)
- **Driver**
- **Passenger**

1.2 Attributes for Each Object

- **Vehicle**: route, capacity, bookedSeats, status, speed, fuelType, year, driver
- **Bus**: (inherits from Vehicle)
- **Train**: (inherits from Vehicle)
- **ExpressBus**: (inherits from Vehicle)
- **Airplane**: (inherits from Vehicle)
- **Ship**: (inherits from Vehicle)

- **Motorbike**: (inherits from Vehicle)
- **Taxi**: (inherits from Vehicle)
- **Driver**: name, id, licenseType
- **Passenger**: name, id, tickets

1.3 Methods

- **Vehicle**: assignDriver(), calculateTravelTime(), bookSeat(), toggleStatus(), getRoute(), getStatus(), getCapacity(), getBookedSeats(), getFuelType(), getYear(), getDriver(), displayInfo(), getName()
- **Bus**: displayInfo(), getName()
- **Train**: displayInfo(), getName()
- **ExpressBus**: calculateTravelTime(), displayInfo(), getName()
- **Airplane**: displayInfo(), getName()
- **Ship**: displayInfo(), getName()
- **Motorbike**: displayInfo(), getName()
- **Taxi**: displayInfo(), getName()
- **Driver**: getName(), getId(), getLicenseType(), getDriver(), displayInfo()
- **Passenger**: bookRide(), displayInfo()

1.4 Inheritance Relationships

- Vehicle is the base class
- Bus, Train, ExpressBus, Airplane, Ship, Motorbike, Taxi inherit from Vehicle

2. Class Design & Example Data

2.1 Class Design Details

- **Encapsulation**:
  Most attributes are private; they are accessed and modified via getter/setter methods.
- **Inheritance**:
  Vehicle is the parent class. Bus, Train, ExpressBus, Airplane, Ship, Motorbike, Taxi inherit and reuse its attributes/methods.
- **Polymorphism**:
  - `displayInfo()` is virtual in Vehicle and overridden in each derived class.
  - `calculateTravelTime()` is overridden in ExpressBus for faster calculation.
- **Abstraction**:
  Vehicle defines a general structure; specific transport types implement their own details.

2.2 Example Data

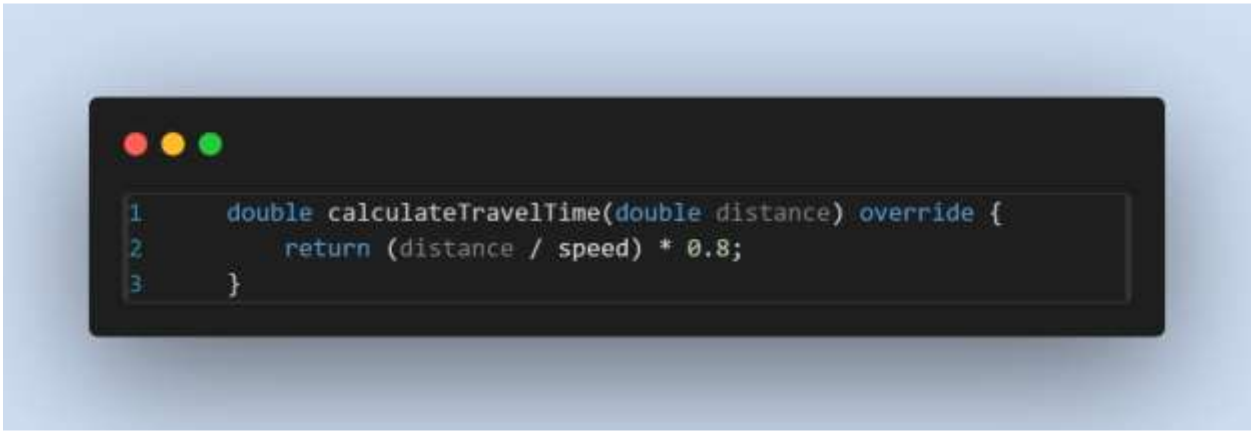
- **Vehicles**:
  Bus("A", 40, 50, "Diesel", 2015)
  Train("T1", 200, 80, "Electric", 2020)
  ExpressBus("E1", 30, 60, "Gasoline", 2018)
  Airplane("F1", 150, 600, "Jet fuel", 2021)
  Ship("S1", 500, 40, "Diesel", 2010)
  Motorbike("M1", 5, 100, "Gasoline", 2019)
  Taxi("T2", 4, 70, "Gasoline", 2017)

- **Passengers**:

Alice (ID: 101)
Bob (ID: 102)
Charlie (ID: 103)

3. Code Walkthrough

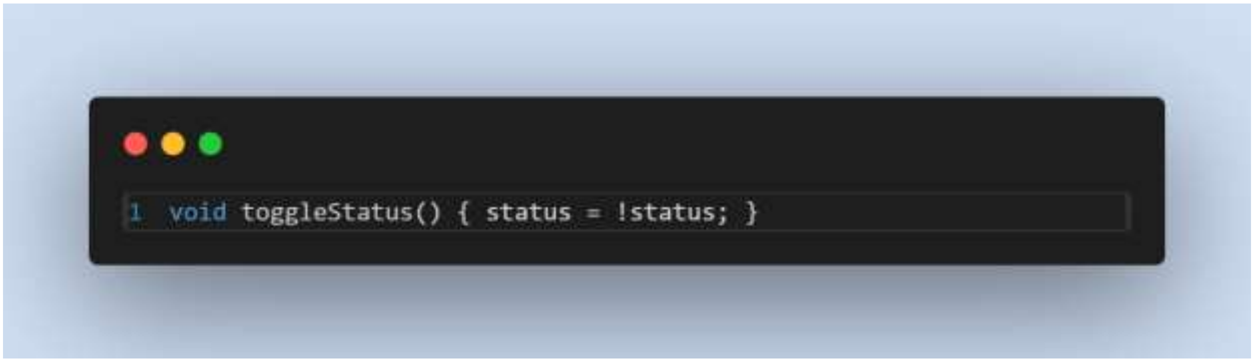
override in ExpressBus:



```
double calculateTravelTime(double distance) override {
    return (distance / speed) * 0.8;
}
```

->Regular Bus travel times are longer than ExpressBus 20%

toggleStatus() changes vehicle's activate/inactivate state



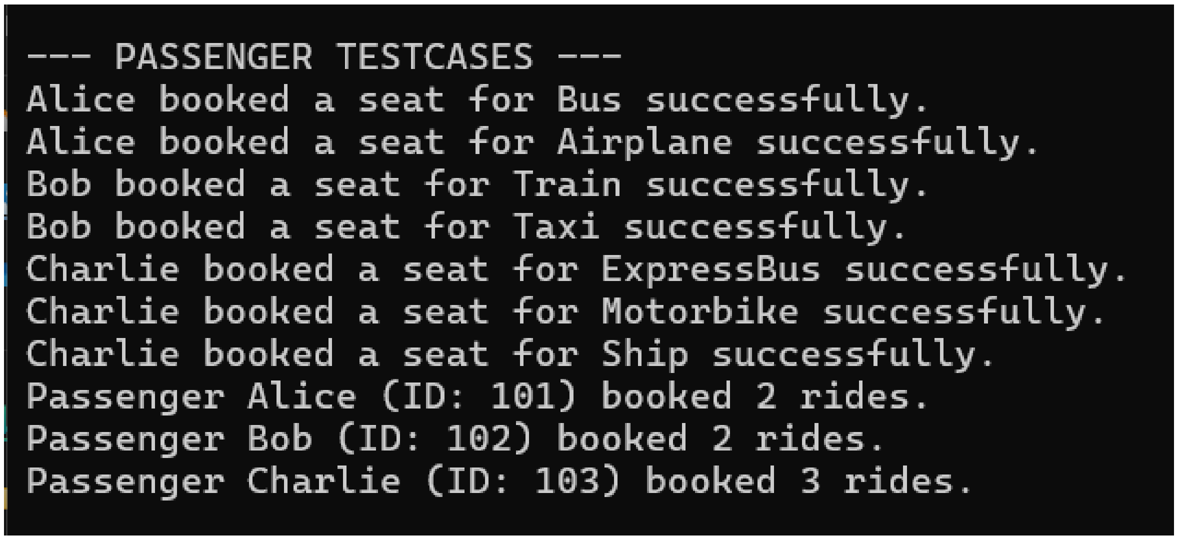
```
void toggleStatus() { status = !status; }
```

Menu System:

- Passenger Menu → booking, view routes, view status, view drivers.
- Admin Menu → view/add vehicles, change status, assign driver, calculate travel time, search route, view drivers.

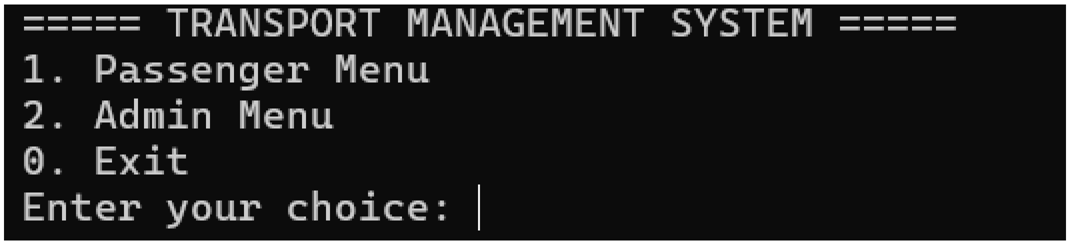
4. Sample Output

When running with the example data , console output includes:



```
--- PASSENGER TESTCASES ---
Alice booked a seat for Bus successfully.
Alice booked a seat for Airplane successfully.
Bob booked a seat for Train successfully.
Bob booked a seat for Taxi successfully.
Charlie booked a seat for ExpressBus successfully.
Charlie booked a seat for Motorbike successfully.
Charlie booked a seat for Ship successfully.
Passenger Alice (ID: 101) booked 2 rides.
Passenger Bob (ID: 102) booked 2 rides.
Passenger Charlie (ID: 103) booked 3 rides.
```

After the example, the system enters a loop presenting a menu:



```
===== TRANSPORT MANAGEMENT SYSTEM =====
1. Passenger Menu
2. Admin Menu
0. Exit
Enter your choice: 
```

This allows flexible operations after seeing sample data.

5. Use of LLM (ChatGPT)

I used ChatGPT for:

- Adding subclasses for Airplane, Ship, Taxi, Motorbike.
- Improving Passenger booking logic.
- Designing Admin and Passenger menu structure.
- ClarifyingOOP principles (inheritance, overriding, encapsulation).

Example Prompt:
"Add an ExpressBus class inheriting from Vehicle that overrides travel time."

Response:
ChatGPT suggested overriding calculateTravelTime() and adjusting displayInfo().

All code was understood and personally written.