21110756_NguyenDinhMinhChien_OOP

Documentation Report

1. OOA Analysis

Objects

- **Station**: represents a bus station, responsible for managing vehicles and schedules.
- **Vehicle**: base class for all vehicles. Stores route, capacity, operational status, assigned station, and passenger list.
- **ExpressBus**: derived from Vehicle, adds stopsCount and overrides travel time calculation.
- **Passenger**: represents a customer who can book or cancel a seat on a vehicle.
- **Schedule**: represents a departure plan, containing departure time and the assigned vehicle.

Attributes and Methods

- **Station**:
  - Attributes: name, location, schedules, vehicles.
  - Methods: addSchedule, removeSchedule, addVehicle, removeVehicle, displayInfo.
- **Vehicle**:
  - Attributes: id, route, capacity, status, passengers, assignedStation.
  - Methods: addPassenger, removePassenger, calculateTravelTime, displayInfo.
- **ExpressBus**:
  - Attributes: stopsCount.
  - Methods: overrides calculateTravelTime (20 percent faster).
- **Passenger**:
  - Attributes: id, name, bookedVehicle.
  - Methods: bookVehicle, cancelBooking, displayInfo.
- **Schedule**:
  - Attributes: departureTime, vehicle.
  - Methods: displayInfo.

Inheritance

- **ExpressBus inherits Vehicle**.
- Method calculateTravelTime is overridden in ExpressBus, showing polymorphism.

2. Class Design

- **Encapsulation**: critical attributes are private or protected. Access is only possible via public methods.
- **Inheritance**: Vehicle is the base class, ExpressBus is the subclass.
- **Polymorphism**: calculateTravelTime is implemented differently for ExpressBus.
- **Associations**:
  - Station contains many schedules and vehicles.
  - Schedule connects a vehicle with a departure time.
  - Passenger books a seat on a vehicle.

This design ensures reusability, clarity, and realistic behavior in a transportation system.

3. Code Walkthrough

1. **Station**
   - Manages up to 10 schedules. If more are added, the station rejects them and prints a message.
2. **Vehicle**
   - Before adding a passenger, it checks the capacity. If full, the booking fails with an error message.
3. **ExpressBus**
   - Inherits Vehicle.
   - Overrides calculateTravelTime so that travel is 20 percent faster.
4. **Passenger**
   - Can book a seat using bookVehicle.
   - Can cancel booking, which removes the passenger from the vehicle's passenger list.
5. **Main function**
   - Creates one station, one normal vehicle, one express bus, and passengers.
   - Demonstrates booking, canceling, schedule limits, travel time calculation, and displaying details.

4. Test Results and Sample Output

OUTPUT:

```
=== Public Transportation Station Management System Demo ===
[Station created] Downtown Bus Hub (bus) at 12 Main St
[Station created] Central Train (train) at 1 Station Rd
[Vehicle created] BUS101 | route: A->B | capacity: 2
[Vehicle created] BUS202 | route: C->D | capacity: 3
[Vehicle created] EXP301 | route: X->Y Express | capacity: 4
[ExpressBus created] EXP301 | stops: 3
-- Scheduling tests (max 10 per station) --
[Schedule added] Departure | Vehicle: BUS101 | Time: 08:010 at station Downtown Bus Hub
[Schedule added] Departure | Vehicle: BUS101 | Time: 08:011 at station Downtown Bus Hub
[Schedule added] Departure | Vehicle: BUS101 | Time: 08:012 at station Downtown Bus Hub
[Schedule added] Departure | Vehicle: BUS101 | Time: 08:013 at station Downtown Bus Hub
[Schedule added] Departure | Vehicle: BUS101 | Time: 08:014 at station Downtown Bus Hub
[Schedule added] Departure | Vehicle: BUS101 | Time: 08:015 at station Downtown Bus Hub
[Schedule added] Departure | Vehicle: BUS101 | Time: 08:016 at station Downtown Bus Hub
[Schedule added] Departure | Vehicle: BUS101 | Time: 08:017 at station Downtown Bus Hub
[Schedule added] Departure | Vehicle: BUS101 | Time: 08:018 at station Downtown Bus Hub
[Schedule added] Departure | Vehicle: BUS101 | Time: 08:019 at station Downtown Bus Hub
[Schedule limit reached] Station Downtown Bus Hub cannot accept more schedules.
-- Display schedules at busStation --
Station: Downtown Bus Hub | Location: 12 Main St | Type: bus
  [1] Departure | Vehicle: BUS101 | Route: A->B | Time: 08:010
  [2] Departure | Vehicle: BUS101 | Route: A->B | Time: 08:011
  [3] Departure | Vehicle: BUS101 | Route: A->B | Time: 08:012
  [4] Departure | Vehicle: BUS101 | Route: A->B | Time: 08:013
  [5] Departure | Vehicle: BUS101 | Route: A->B | Time: 08:014
  [6] Departure | Vehicle: BUS101 | Route: A->B | Time: 08:015
  [7] Departure | Vehicle: BUS101 | Route: A->B | Time: 08:016
  [8] Departure | Vehicle: BUS101 | Route: A->B | Time: 08:017
  [9] Departure | Vehicle: BUS101 | Route: A->B | Time: 08:018
  [10] Departure | Vehicle: BUS101 | Route: A->B | Time: 08:019
-- Booking tests (capacity checks) --
[Passenger created] Alice (P100)
[Passenger created] Bob (P101)
[Passenger created] Carol (P102)
[Booked] Alice booked BUS101
```

```
[Booked] Bob booked BUS101
[Vehicle full] BUS101 cannot accept passenger Carol
[Booking failed] Carol could not book BUS101
-- Vehicle info after attempted bookings --
Vehicle ID: BUS101 | Route: A->B | Capacity: 2 | Booked: 2 | Speed: 45 km/h | Status: On-time
-- Cancel and retry booking --
[Cancelled] Bob cancelled BUS101
[Booked] Carol booked BUS101
Vehicle ID: BUS101 | Route: A->B | Capacity: 2 | Booked: 2 | Speed: 45 km/h | Status: On-time
-- Travel time comparison (distance 120.00 km) --
BUS202 time (hrs): 2.40
EXP301 time (hrs): 1.20 (20% faster)
-- Schedule express bus at trainStation --
[Schedule added] Arrival | Vehicle: EXP301 | Time: 09:45 at station Central Train
Station: Central Train | Location: 1 Station Rd | Type: train
  [1] Arrival | Vehicle: EXP301 | Route: X->Y Express | Time: 09:45
-- Remove schedule example --
[Schedule removed] Vehicle BUS101 removed from Downtown Bus Hub
Station: Downtown Bus Hub | Location: 12 Main St | Type: bus
  [1] Departure | Vehicle: BUS101 | Route: A->B | Time: 08:011
  [2] Departure | Vehicle: BUS101 | Route: A->B | Time: 08:012
  [3] Departure | Vehicle: BUS101 | Route: A->B | Time: 08:013
  [4] Departure | Vehicle: BUS101 | Route: A->B | Time: 08:014
  [5] Departure | Vehicle: BUS101 | Route: A->B | Time: 08:015
  [6] Departure | Vehicle: BUS101 | Route: A->B | Time: 08:016
  [7] Departure | Vehicle: BUS101 | Route: A->B | Time: 08:017
  [8] Departure | Vehicle: BUS101 | Route: A->B | Time: 08:018
  [9] Departure | Vehicle: BUS101 | Route: A->B | Time: 08:019

-- Passenger info --
Passenger: Alice (ID: P100) | Booked: BUS101
Passenger: Bob (ID: P101) | Booked: none
Passenger: Carol (ID: P102) | Booked: BUS101
=== Demo complete ===
[Vehicle destroyed] BUS202
[Station destroyed] Central Train
[ExpressBus destroyed] EXP301
[Vehicle destroyed] EXP301
[Station destroyed] Downtown Bus Hub
[Vehicle destroyed] BUS101
D:\HCM_UTE\YEAR 5_HCMUTE_25-26\OOP\Public Transportation Station Management System\x64\Debug\Public
Transportation Station Management System.exe (process 3620) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the
console when debugging stops.
Press any key to close this window . . .
```

Test Case 1: Adding schedules beyond the limit

- Station accepts up to 10 schedules.
- When adding the 11th and 12th schedules, the system prints:

Station Central: cannot add more schedules (max 10).

Confirms schedule limit is working.

Test Case 2: Booking passengers until vehicle capacity is full

- Vehicle capacity = 2.
- Booking 3 passengers: first 2 succeed, last one fails.

Passenger Alice booked Vehicle V1 successfully.

Passenger Bob booked Vehicle V1 successfully.

Vehicle V1 is full. Cannot add passenger Charlie.
Passenger Charlie failed to book Vehicle V1.
Confirms passenger capacity check.

Test Case 3: Canceling and rebooking

- Passenger Bob cancels his booking, freeing a seat.
- Passenger Charlie books again successfully.

Passenger Bob canceled booking on Vehicle V1.
Passenger Charlie booked Vehicle V1 successfully.
Confirms cancel and rebooking logic.

Test Case 4: Travel time comparison

- Normal Vehicle travel time for 120 km:

Vehicle V1 travel time for 120 km: 120 min

- ExpressBus travel time (20 percent faster):

Express Bus EB1 travel time for 120 km: 96 min
Confirms inheritance and overridden method.

Test Case 5: Displaying passenger info

Passenger[1] Alice booked Vehicle V1
Passenger[2] Bob has no booking
Passenger[3] Charlie booked Vehicle V1
Confirms passenger information display.

Test Case 6: Removing a schedule

Station Central removed schedule at 08:10.
Confirms schedule removal is working.

5. LLM Usage

During development, ChatGPT was used for:

- Brainstorming the object model and class design.
- Suggesting inheritance and encapsulation practices.
- Validating whether the implementation met the rubric.
- Drafting the documentation structure.

The core implementation was coded by the student. ChatGPT mainly supported explanation and error checking.

6. Conclusion

The system successfully simulates a station management environment.

- OOA was correctly applied with Station, Vehicle, ExpressBus, Passenger, and Schedule.
- The C++ implementation uses encapsulation, inheritance, polymorphism, and associations.
- Tests confirmed schedule limits, capacity limits, booking/cancel behavior, overridden methods, and display functions.
- Documentation and LLM usage are clearly presented.

This project fulfills all rubric requirements and demonstrates practical OOP design in C++.