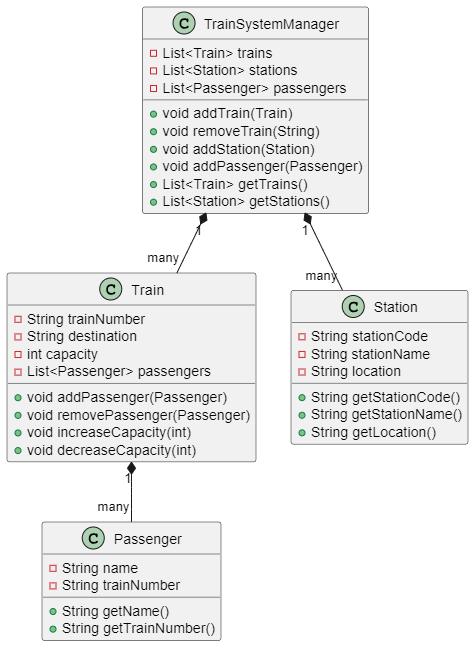
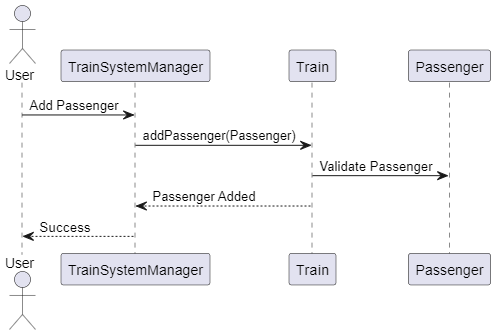
**UML Diagram**



**Sequence diagram**



**Skeloton code**

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

public class TrainManagementSystem {

private static List<Train> trains = new ArrayList<>();

private static List<Passenger> passengers = new ArrayList<>();

private static List<Station> stations = new ArrayList<>();

private static Scanner scanner = new Scanner(System.in);

public static void main(String[] args) {

boolean running = true;

while (running) {

System.out.println("\n===== Train Management System =====");

System.out.println("1. Add New Train");

System.out.println("2. Add Passenger");

System.out.println("3. Add Station");

System.out.println("4. List All Trains");

System.out.println("5. List All Passengers");

System.out.println("6. List All Stations");

System.out.println("7. Remove Train");

System.out.println("8. Exit");

System.out.print("Enter your choice: ");

int choice = scanner.nextInt();

scanner.nextLine(); // Consume newline

switch (choice) {

case 1 -> addTrain();

case 2 -> addPassenger();

case 3 -> addStation();

case 4 -> listTrains();

case 5 -> listPassengers();

case 6 -> listStations();

case 7 -> removeTrain();

case 8 -> running = false;

default -> System.out.println("Invalid choice! Please try again.");

}

}

System.out.println("Exiting system...");

}

private static void addTrain() {

// Implement logic to add a train

}

private static void addPassenger() {

// Implement logic to add a passenger

}

private static void addStation() {

// Implement logic to add a station

}

private static void listTrains() {

// Implement logic to list all trains

}

private static void listPassengers() {

// Implement logic to list all passengers

}

private static void listStations() {

// Implement logic to list all stations

}

private static void removeTrain() {

// Implement logic to remove a train

}

}

class Train {

private final String trainNumber;

private String destination;

private int capacity;

private List<Passenger> passengers;

public Train(String trainNumber, String destination, int capacity) {

// Initialize fields

}

public String getTrainNumber() {

// Return train number

}

public String getDestination() {

// Return destination

}

public int getCapacity() {

// Return capacity

}

public void setDestination(String destination) {

// Set destination

}

public void increaseCapacity(int amount) {

// Increase capacity

}

public void decreaseCapacity(int amount) {

// Decrease capacity

}

public void addPassenger(Passenger passenger) {

// Add passenger

}

public void removePassenger(Passenger passenger) {

// Remove passenger

}

@Override

public String toString() {

// Return string representation of the train

}

}

class Passenger {

private final String name;

private final String trainNumber;

public Passenger(String name, String trainNumber) {

// Initialize fields

}

public String getName() {

// Return name

}

public String getTrainNumber() {

// Return train number

}

@Override

public String toString() {

// Return string representation of the passenger

}

}

class Station {

private final String stationCode;

private final String stationName;

private final String location;

public Station(String stationCode, String stationName, String location) {

// Initialize fields

}

public String getStationCode() {

// Return station code

}

public String getStationName() {

// Return station name

}

public String getLocation() {

// Return location

}

@Override

public String toString() {

// Return string representation of the station

}

}

**Business logic code**

Adding Train

private static void addTrain() {

// Logic for adding a new train

System.out.print("Enter train number: ");

String trainNumber = scanner.nextLine();

System.out.print("Enter destination: ");

String destination = scanner.nextLine();

System.out.print("Enter capacity: ");

int capacity = scanner.nextInt();

scanner.nextLine(); // Consume newline

Train train = new Train(trainNumber, destination, capacity);

trains.add(train);

System.out.println("Train added successfully!");

}

Adding passenger

private static void addPassenger() {

// Logic for adding a new passenger

System.out.print("Enter passenger name: ");

String name = scanner.nextLine();

System.out.print("Enter train number: ");

String trainNumber = scanner.nextLine();

Train train = findTrain(trainNumber);

if (train == null) {

System.out.println("Train not found!");

return;

}

Passenger passenger = new Passenger(name, trainNumber);

train.addPassenger(passenger);

passengers.add(passenger);

System.out.println("Passenger added successfully!");

}

Adding stations

private static void addStation() {

// Logic for adding a new station

System.out.print("Enter station code: ");

String stationCode = scanner.nextLine();

System.out.print("Enter station name: ");

String stationName = scanner.nextLine();

System.out.print("Enter location: ");

String location = scanner.nextLine();

Station station = new Station(stationCode, stationName, location);

stations.add(station);

System.out.println("Station added successfully!");

}

Removing trains

private static void removeTrain() {

// Logic for removing a train

System.out.print("Enter train number to remove: ");

String trainNumber = scanner.nextLine();

boolean isRemoved = trains.removeIf(train -> train.getTrainNumber().equals(trainNumber));

if (isRemoved) {

passengers.removeIf(passenger -> passenger.getTrainNumber().equals(trainNumber));

System.out.println("Train removed successfully!");

} else {

System.out.println("Train not found!");

}

}