Assignment 4 – Travel Ireland Bus Project

Object Oriented Programming

Maxwell Maia

21236277

Classes

Trip

Store and allow retrieval of information about a Trip.
Able to set a new value for availableSeats (does not allow availableSeats variable to be set to a negative number).

Fields:
(
int tripID
String startingLocation
String destination
String dateOfDeparture
String timeOfDeparture
String dateOfArrival
String timeOfArrival
double fare
int availableSeats
)

Booking

Store and allow retrieval of information about a booking.

Calculates the total cost of a booking.

Fields:
(
Trip selectedTrip
int seatsRequested
)

Vendor

Interface

It's role is to provide function definitions that each vendor has to follow. When a new vendor is added all the vendor need to do is provide concrete definitions for these methods. Using an interface allows the program to reference any vendor with as a Vendor object. This is because every vendor class "implements Vendor" so every vendor class "is-a" Vendor.

e.g. public class BusEireann implements Vendor

so BusEireann can be referenced as a Vendor object.

Vendor be = new BusEireann();

I can now create an array of all of the vendors if I wanted to but this is outside the scope of this assignment.

Any new vendor can be added to Travel Ireland without modifying existing code.

Open for extension, closed for modification.

This is a good programming practise because it makes code maintainable. I have chosen to use an interface as I have to implement this "open-close" principle.

Methods

public String displayTrips();

Creates and return a string of all of the details of all of the available trips.

public Trip getTrip(int id);

public boolean makeBooking(Booking booking);

A method that makes a booking.

- It checks that there are enough seats on trip to accommodate the booking.
- Update the number of available seats if there are enough seats for the booking.
- Returns a Boolean value that is true if the booking was a success.

<Any vendor> (BusEireann, CityLink, GoBus)

BusEireann, CityLink, GoBus are all vendors and are similar.

These classes “implements Vendor” so they must provide concrete definitions for the methods in the Vendor class.

An ArrayList was chosen to store all of the trips because ArrayLists are dynamic which makes them perfect for this application as a vendor may change the trips they offer as frequently as they like. In my assignment it makes it really easy to store and add Trip objects a data structure.

BusEireann

Is-a Vendor

* Provides concrete definitions of Vendor class.

* Provides concrete definitions of the following methods:

* displayTrips() display all information of all its trips.

* getTrip() return a trip by searching through its ArrayList for a trip with a specific ID.

* makeBooking() make a booking, check that there are enough seats available.

Travel_Ireland

This is a test class.

It initializes some vendors with trips stored in their ArrayLists.

It displays all of the information of all of the trips for each vendor.

It selects trips by searching them by their trip ID.

It displays a test which shows whether a booking should be successful or not based on available seats.

It attempts to make a booking for each vendor.

It displays the booking information (if the booking was actually successful) for every booking.

It displays all trip information for each vendor again to reflect that the number of available seats has decreased.

Source Code

Trip

```
/**
 * Trip
 *
 * Store and allow retrieval of information about a Trip
 * Able to set a new value for availableSeats (does not allow availableSeats variable to be set to a negative number)
 *
 * @author Maxwell Maia
 * @student id 21236277
 * @version 1.0
 */
public class Trip
{
   int tripID;
   String startingLocation;
   String destination;
   String dateOfDeparture;
   String timeOfDeparture;
   String dateOfArrival;
   String timeOfArrival;
   double fare;
   int availableSeats;


   public Trip(int tripID, String startingLocation, String destination, String dateOfDeparture, String timeOfDeparture, String dateOfArrival, String timeOfArrival, double fare, int availableSeats)
   {
      this.tripID = tripID;
      this.startingLocation = startingLocation;
```

```
    this.destination = destination;
    this.dateOfDeparture = dateOfDeparture;
    this.timeOfDeparture = timeOfDeparture;
    this.dateOfArrival = dateOfArrival;
    this.timeOfArrival = timeOfArrival;
    this.fare = fare;
    this.availableSeats = availableSeats;
}

//Getters for all fields.
public int getTripID()
{
    return tripID;
}

public String getStartingLocation()
{
    return startingLocation;
}

public String getDestination()
{
    return destination;
}

public String getDateOfDeparture()
{
    return dateOfDeparture;
}

public String getTimeOfDeparture()
```

```
{
   return timeOfDeparture;
}

public String getDateOfArrival()
{
   return dateOfArrival;
}

public String getTimeOfArrival()
{
   return timeOfArrival;
}

public double getFare()
{
   return fare;
}

public int getAvailableSeats()
{
   return availableSeats;
}

//Setter for available seats. Validity check for changing data.
public boolean setAvailableSeats(int newAvailableSeats)
{
   if(newAvailableSeats >= 0)
   {
      availableSeats = newAvailableSeats;
      return true;
```

```
        }

        //Return false and do not change seats if the new number of available seats is negative.
        return false;
    }
}
```

Booking

```
/**
 * Booking
 *
 * Store and allow retrieval of information about a booking
 * Calculates the total cost of a booking
 *
 * @author Maxwell Maia
 * @student id 21236277
 * @version 1.0
 */
public class Booking
{
    Trip trip;
    int requestedSeats;


    public Booking(Trip trip, int requestedSeats)
    {
        this.trip = trip;
        this.requestedSeats = requestedSeats;
    }
```

```
    //Getters for all fields.
    public Trip getTrip()
    {
        return trip;
    }

    public int getRequestedSeats()
    {
        return requestedSeats;
    }

    //Calculates and returns the total cost of the booking.
    public double getTotalCost()
    {
        return trip.getFare() * requestedSeats;
    }
}
```

Vendor

```
/**
 * Vendor
 *
 * Provides function definitions that each vendor has to use
 *
 * @author Maxwell Maia
 * @student id 21236277
 * @version 1.0
```

```
 */
public interface Vendor
{

   //Display all available trips and all the information regarding them
   // e.g. ID, Origin, Destination... , currently available seats.
   // (used to print all of the available fields of all Trip objects in an ArrayList).
   public String displayTrips();

   //Search for (in the ArrayList) and return the Trip object, given an id.
   //Returns null if no object is found with the id that is input.
   public Trip getTrip(int id);

   //Make a booking.
   //Will check that the available seats is more (or equal to) the number of passengers (requestedSeats).
   //Returns true if the booking was made.
   //Returns a false if the booking was not made. invalid details:
   // e.g. more seats booked than are available.
   public boolean makeBooking(Booking booking);
}
```

BusEireann

```
import java.util.ArrayList;
/**
 * BusEireann
 *
 * Is-a Vendor
 * Provides concrete definitions of Vendor class.
```

```
 * Stores an ArrayList of all of the trips this vendor offers.
 * Provides concrete definitions of the following methods:
 * displayTrips() display all information of all its trips.
 * getTrip() return a trip by searching through its ArrayList for a trip with a specific ID.
 * makeBooking() make a booking, check that there are enough seats available.
 *
 * @author Maxwell Maia, 21236277
 * @version 1.0
 */
public class BusEireann implements Vendor
{
   private ArrayList<Trip> trips = new ArrayList();


   /**
    * Constructor for objects of class BusEireann
    */
   public BusEireann()
   {
      //Add trips on initialization. (Assignment says that hardcoding this is allowed).
      Trip trip1 = new Trip(1010, "Galway", "Cork", "2022/11/25", "22:00", "2022/11/26", "01:10",
12.50, 60);
      Trip trip2 = new Trip(5124, "Galway", "Limerick", "2022/12/22", "16:15", "2022/12/22", "17:30",
6.50, 1);


      trips.add(trip1);
      trips.add(trip2);
   }


   public String displayTrips()
   {
      String ret = "";
      ret += "\n-----------------------------------";
```

```
    //Loop through the trip arraylist and compile a string to display all information on all trips.
    for(Trip trip : trips)
    {
        ret += "\nCompany: Bus-Eireann";
        ret += "\nTrip ID: " + trip.getTripID();
        ret += "\nOrigin: " + trip.getStartingLocation();
        ret += "\nDestination: " + trip.getDestination();
        ret += "\nDeparture Date: " + trip.getDateOfDeparture();
        ret += "\nDeparture Time: " + trip.getTimeOfDeparture();
        ret += "\nArrival Date: " + trip.getDateOfArrival();
        ret += "\nArrival Time: " + trip.getTimeOfArrival();
        ret += "\nFare: €" + trip.getFare() + " per passenger";
        ret += "\nCurrently Available seats: " + trip.getAvailableSeats();
        ret += "\n";
    }
    ret += "\n-------------------------------------";
    return ret;
}

public Trip getTrip(int id)
{
    //Search through the trip arraylist and return the trip that matches the trip id entered.
    for(Trip trip : trips)
    {
        if(id == trip.getTripID())
        {
            return trip;
        }
    }
    //If the trip with a matching ID has not been found, return null.
    return null;
```

```
    }

    /* Make a booking.
     * Check that there are enough seats on trip to accommodate the booking.
     * Update seats if there are enough seats for the booking.
     * Return a boolean that is true if the booking was a success.
     */
    public boolean makeBooking(Booking booking)
    {
        //Get available seats
        int availableSeats = booking.getTrip().getAvailableSeats();

        //Get requested seats (number of passengers in booking)
        int requestedSeats = booking.getRequestedSeats();

        //If there are enough seats to accomdate the customer's request, the booking will be successful.
        if(availableSeats >= requestedSeats)
        {
            //There are enough seats.
            //Change new amount of available seats and return "true" to indicated a successful booking.
            booking.getTrip().setAvailableSeats(availableSeats - requestedSeats);
            return true;
        }

        //Else there are not enough seats, so the booking was not a success.
        //Return "false" to indicate an unsuccessful booking.
        return false;
    }
}
```

CityLink

```
import java.util.ArrayList;
/**
 * CityLink
 *
 * Is-a Vendor
 * Provides concrete definitions of Vendor class.
 * Stores an ArrayList of all of the trips this vendor offers.
 * Provides concrete definitions of the following methods:
 * displayTrips() display all information of all its trips.
 * getTrip() return a trip by searching through its ArrayList for a trip with a specific ID.
 * makeBooking() make a booking, check that there are enough seats available.
 *
 * @author Maxwell Maia, 21236277
 * @version 1.0
 */
public class CityLink implements Vendor
{
   private ArrayList<Trip> trips = new ArrayList();

   /**
    * Constructor for objects of class BusEireann
    */
   public CityLink()
   {
      //Add trips on initialization. (Assignment says that hardcoding this is allowed).
      Trip trip1 = new Trip(2010, "Galway", "Cork", "2022/11/25", "22:00", "2022/11/26", "01:10",
12.50, 60);
      Trip trip2 = new Trip(6124, "Galway", "Limerick", "2022/12/22", "16:15", "2022/12/22", "17:30",
6.50, 1);
```

```
    trips.add(trip1);
    trips.add(trip2);
}

public String displayTrips()
{
    String ret = "";
    ret += "\n---------------------------------------";
    //Loop through the trip arraylist and compile a string to display all information on all trips.
    for(Trip trip : trips)
    {
        ret += "\nCompany: City-Link";
        ret += "\nTrip ID: " + trip.getTripID();
        ret += "\nOrigin: " + trip.getStartingLocation();
        ret += "\nDestination: " + trip.getDestination();
        ret += "\nDeparture Date: " + trip.getDateOfDeparture();
        ret += "\nDeparture Time: " + trip.getTimeOfDeparture();
        ret += "\nArrival Date: " + trip.getDateOfArrival();
        ret += "\nArrival Time: " + trip.getTimeOfArrival();
        ret += "\nFare: €" + trip.getFare() + " per passenger";
        ret += "\nCurrently Available seats: " + trip.getAvailableSeats();
        ret += "\n";
    }
    ret += "\n---------------------------------------";
    return ret;
}

public Trip getTrip(int id)
{
    //Search through the trip arraylist and return the trip that matches the trip id entered.
    for(Trip trip : trips)
```

```
    {
        if(id == trip.getTripID())
        {
            return trip;
        }
    }
    //If the trip with a matching ID has not been found, return null.
    return null;
}

/* Make a booking.
 * Check that there are enough seats on trip to accommodate the booking.
 * Update seats if there are enough seats for the booking.
 * Return a boolean that is true if the booking was a success.
 */
public boolean makeBooking(Booking booking)
{
    //Get available seats
    int availableSeats = booking.getTrip().getAvailableSeats();

    //Get requested seats (number of passengers in booking)
    int requestedSeats = booking.getRequestedSeats();

    //If there are enough seats to accomdate the customer's request, the booking will be successful.
    if(availableSeats >= requestedSeats)
    {
        //There are enough seats.
        //Change new amount of available seats and return "true" to indicated a successful booking.
        booking.getTrip().setAvailableSeats(availableSeats - requestedSeats);
        return true;
    }
```

```
        //Else there are not enough seats, so the booking was not a success.
        //Return "false" to indicate an unsuccessful booking.
        return false;
    }
}
```

GoBus

```
import java.util.ArrayList;
/**
 * GoBus
 *
 * Is-a Vendor
 * Provides concrete definitions of Vendor class.
 * Stores an ArrayList of all of the trips this vendor offers.
 * Provides concrete definitions of the following methods:
 * displayTrips() display all information of all its trips.
 * getTrip() return a trip by searching through its ArrayList for a trip with a specific ID.
 * makeBooking() make a booking, check that there are enough seats available.
 *
 * @author Maxwell Maia, 21236277
 * @version 1.0
 */
public class GoBus implements Vendor
{
    private ArrayList<Trip> trips = new ArrayList();

    /**
     * Constructor for objects of class BusEireann
```

```
     */
    public GoBus()
    {
        //Add trips on initialization. (Assignment says that hardcoding this is allowed).
        Trip trip1 = new Trip(3010, "Galway", "Cork", "2022/11/25", "22:00", "2022/11/26", "01:10",
12.50, 60);
        Trip trip2 = new Trip(7124, "Galway", "Limerick", "2022/12/22", "16:15", "2022/12/22", "17:30",
6.50, 1);

        trips.add(trip1);
        trips.add(trip2);
    }

    public String displayTrips()
    {
        String ret = "";
        ret += "\n-------------------------------------";
        //Loop through the trip arraylist and compile a string to display all information on all trips.
        for(Trip trip : trips)
        {
            ret += "\nCompany: GoBus";
            ret += "\nTrip ID: " + trip.getTripID();
            ret += "\nOrigin: " + trip.getStartingLocation();
            ret += "\nDestination: " + trip.getDestination();
            ret += "\nDeparture Date: " + trip.getDateOfDeparture();
            ret += "\nDeparture Time: " + trip.getTimeOfDeparture();
            ret += "\nArrival Date: " + trip.getDateOfArrival();
            ret += "\nArrival Time: " + trip.getTimeOfArrival();
            ret += "\nFare: €" + trip.getFare() + " per passenger";
            ret += "\nCurrently Available seats: " + trip.getAvailableSeats();
            ret += "\n";
        }
```

```
    ret += "\n-----------------------------------";
    return ret;
}

public Trip getTrip(int id)
{
    //Search through the trip arraylist and return the trip that matches the trip id entered.
    for(Trip trip : trips)
    {
        if(id == trip.getTripID())
        {
            return trip;
        }
    }
    //If the trip with a matching ID has not been found, return null.
    return null;
}

/* Make a booking.
 * Check that there are enough seats on trip to accommodate the booking.
 * Update seats if there are enough seats for the booking.
 * Return a boolean that is true if the booking was a success.
 */
public boolean makeBooking(Booking booking)
{
    //Get available seats
    int availableSeats = booking.getTrip().getAvailableSeats();

    //Get requested seats (number of passengers in booking)
    int requestedSeats = booking.getRequestedSeats();
```

```
        //If there are enough seats to accomdate the customer's request, the booking will be successful.
        if(availableSeats >= requestedSeats)
        {
            //There are enough seats.
            //Change new amount of available seats and return "true" to indicated a successful booking.
            booking.getTrip().setAvailableSeats(availableSeats - requestedSeats);
            return true;
        }

        //Else there are not enough seats, so the booking was not a success.
        //Return "false" to indicate an unsuccessful booking.
        return false;
    }
}
```

Travel_Ireland

```
/**
 * Write a description of class Travel_Ireland here.
 *
 * @author Maxwell Maia
 * @student id 21236277
 * @version 1.0
 */
public class Travel_Ireland
{
    public static void main(String[] args)
    {
        System.out.println("Welcome to Travel Ireland\n\nAvailable trips:");
```

```
//Vendor objects.
//Create a vendor object (BusEireann). It's array list is populated with default trips on initilization.
Vendor be = new BusEireann();
//Create a vendor object (CityLink). It's array list is populated with default trips on initilization.
Vendor cl = new CityLink();
//Create a vendor object (GoBus). It's array list is populated with default trips on initilization.
Vendor gb = new GoBus();

//Put vendors into a vendor array.
//Able to do this because each vendor is a Vendor.
Vendor[] vendorArray = {be, cl, gb};
int vendArrSize = 3;

//Display all available trips from all vendors.
for(int i = 0; i < vendArrSize; i++)
{
   System.out.println(vendorArray[i].displayTrips());
}



//BOOKINGS
//Initialize variables
Trip selectedTrip = null;
Booking booking = null;
Boolean success = false;

//BUS-EIREANN Booking
//Get the trip object that the customer wants. e.g. trip with id 1010
selectedTrip = be.getTrip(1010);
```

```
        //Test code
        //This is just code that helps verify if a booking should be successful or not.
        System.out.println("\n\nAttempting to book 10 passengers from on Bus-Eireann trip ID: 1010.");
        System.out.println("There are " + selectedTrip.getAvailableSeats() + " seats available on that
trip.");
        if(selectedTrip.getAvailableSeats() >= 10)
        {
           System.out.println("There are enough seats.\nThe booking should be successful.");
        }
        else
        {
           System.out.println("There are not enough seats.\nThe booking should not be successful.");
        }

        //Make a booking with the selected trip and the number of seats the customer wants to book.
        booking = new Booking(selectedTrip, 10);
        success = be.makeBooking(booking);

        //Display appropriate output based on whether the booking was successful or not.
        if(success)
        {
           System.out.println("\nBooking successful.");

System.out.println("============================================================");
           System.out.println("Number of passengers: " + booking.getRequestedSeats());
           System.out.println("Trip Details: ["+booking.getTrip().getStartingLocation()+"] to
["+booking.getTrip().getDestination()+"]");
           System.out.println("Trip ID: " + booking.getTrip().getTripID());
           System.out.println("Total cost: €" + booking.getTotalCost());

System.out.println("============================================================");
```

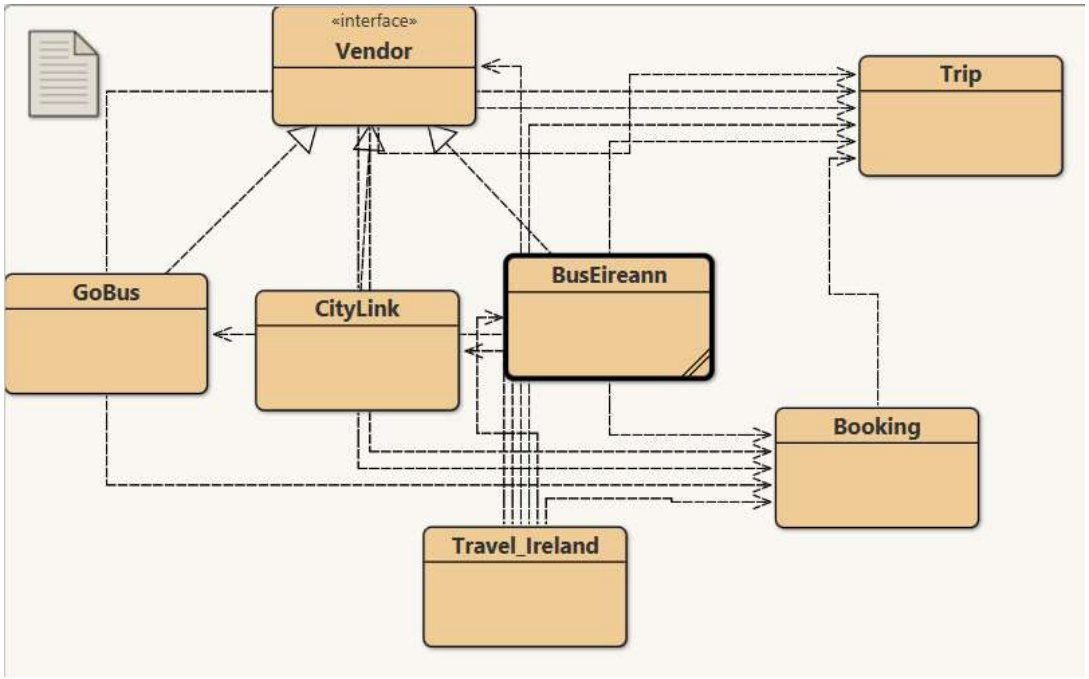
```
}
else
{
   System.out.println("\n\nBooking failed!");
}



//CITY-LINK Booking
//Get the trip object that the customer wants. e.g. trip with id 1010
selectedTrip = cl.getTrip(6124);

//Test code
//This is just code that helps verify if a booking should be successful or not.
System.out.println("\n\nAttempting to book 1 passenger from on City-Link trip ID: 6124.");
System.out.println("There are " + selectedTrip.getAvailableSeats() + " seats available on that trip.");
if(selectedTrip.getAvailableSeats() >= 1)
{
   System.out.println("There are enough seats.\nThe booking should be successful.");
}
else
{
   System.out.println("There are not enough seats.\nThe booking should not be successful.");
}

//Make a booking with the selected trip and the number of seats the customer wants to book.
booking = new Booking(selectedTrip, 1);
success = cl.makeBooking(booking);
```

```
        //Display appropriate output based on whether the booking was successful or not.
        if(success)
        {
            System.out.println("\nBooking successful.");

System.out.println("===============================================================");
            System.out.println("Number of passengers: " + booking.getRequestedSeats());
            System.out.println("Trip Details: ["+booking.getTrip().getStartingLocation()+"] to
["+booking.getTrip().getDestination()+"]");
            System.out.println("Trip ID: " + booking.getTrip().getTripID());
            System.out.println("Total cost: €" + booking.getTotalCost());

System.out.println("===============================================================");
        }
        else
        {
            System.out.println("\n\nBooking failed!");
        }




        //GOBUS Booking
        //Get the trip object that the customer wants. e.g. trip with id 1010
        selectedTrip = gb.getTrip(7124);


        //Test code
        //This is just code that helps verify if a booking should be successful or not.
        System.out.println("\n\nAttempting to book 2 passengers from on Bus-Eireann trip ID: 7124.");
        System.out.println("There are " + selectedTrip.getAvailableSeats() + " seats available on that
trip.");
        if(selectedTrip.getAvailableSeats() >= 2)
```

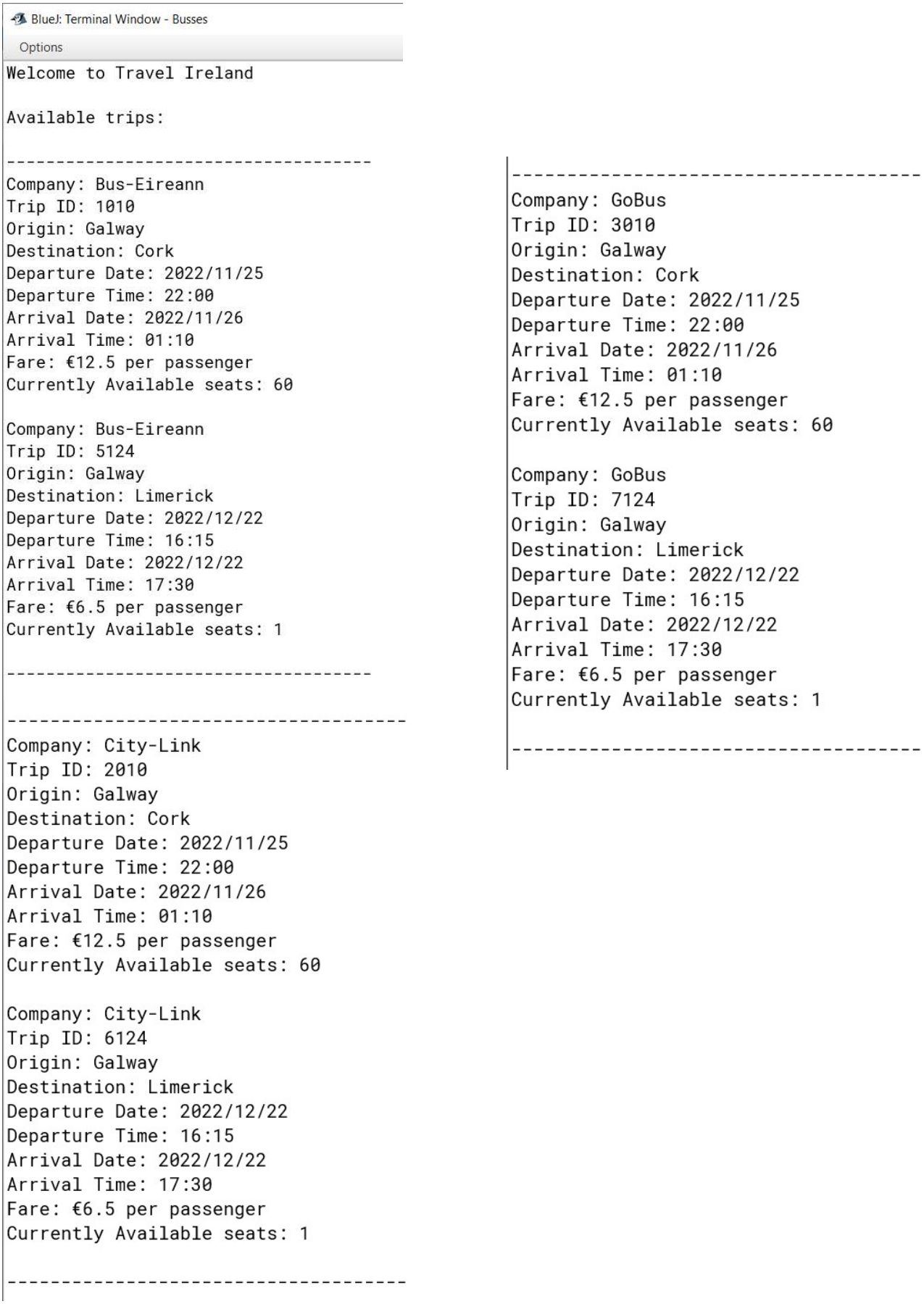
```
        {
            System.out.println("There are enough seats.\nThe booking should be successful.");
        }
        else
        {
            System.out.println("There are not enough seats.\nThe booking should not be successful.");
        }

        //Make a booking with the selected trip and the number of seats the customer wants to book.
        booking = new Booking(selectedTrip, 2);
        success = gb.makeBooking(booking);

        //Display appropriate output based on whether the booking was successful or not.
        if(success)
        {
            System.out.println("\nBooking successful.");

System.out.println("============================================================");
            System.out.println("Number of passengers: " + booking.getRequestedSeats());
            System.out.println("Trip Details: ["+booking.getTrip().getStartingLocation()+"] to
["+booking.getTrip().getDestination()+"]");
            System.out.println("Trip ID: " + booking.getTrip().getTripID());
            System.out.println("Total cost: €" + booking.getTotalCost());

System.out.println("============================================================");
        }
        else
        {
            System.out.println("\n\nBooking failed!");
        }
```

```
        //Display all available trips from all vendors.
        for(int i = 0; i < vendArrSize; i++)
        {
           System.out.println(vendorArray[i].displayTrips());
        }
    }
}
```

Screenshot of structure



Screenshots of output



```
Attempting to book 10 passengers from on Bus-Eireann trip ID: 1010.
There are 60 seats available on that trip.
There are enough seats.
The booking should be successful.

Booking successful.
================================================================
Number of passengers: 10
Trip Details: [Galway] to [Cork]
Trip ID: 1010
Total cost: €125.0
================================================================


Attempting to book 1 passenger from on City-Link trip ID: 6124.
There are 1 seats available on that trip.
There are enough seats.
The booking should be successful.

Booking successful.
================================================================
Number of passengers: 1
Trip Details: [Galway] to [Limerick]
Trip ID: 6124
Total cost: €6.5
================================================================
```

```
Attempting to book 2 passengers from on Bus-Eireann trip ID: 7124.
There are 1 seats available on that trip.
There are not enough seats.
The booking should not be successful.


Booking failed!
```

```
-------------------------------------
Company: Bus-Eireann
Trip ID: 1010
Origin: Galway
Destination: Cork
Departure Date: 2022/11/25
Departure Time: 22:00
Arrival Date: 2022/11/26
Arrival Time: 01:10
Fare: €12.5 per passenger
Currently Available seats: 50

Company: Bus-Eireann
Trip ID: 5124
Origin: Galway
Destination: Limerick
Departure Date: 2022/12/22
Departure Time: 16:15
Arrival Date: 2022/12/22
Arrival Time: 17:30
Fare: €6.5 per passenger
Currently Available seats: 1

-------------------------------------
```

```
-------------------------------------
Company: City-Link
Trip ID: 2010
Origin: Galway
Destination: Cork
Departure Date: 2022/11/25
Departure Time: 22:00
Arrival Date: 2022/11/26
Arrival Time: 01:10
Fare: €12.5 per passenger
Currently Available seats: 60

Company: City-Link
Trip ID: 6124
Origin: Galway
Destination: Limerick
Departure Date: 2022/12/22
Departure Time: 16:15
Arrival Date: 2022/12/22
Arrival Time: 17:30
Fare: €6.5 per passenger
Currently Available seats: 0

-------------------------------------
```

```
-------------------------------------
Company: GoBus
Trip ID: 3010
Origin: Galway
Destination: Cork
Departure Date: 2022/11/25
Departure Time: 22:00
Arrival Date: 2022/11/26
Arrival Time: 01:10
Fare: €12.5 per passenger
Currently Available seats: 60

Company: GoBus
Trip ID: 7124
Origin: Galway
Destination: Limerick
Departure Date: 2022/12/22
Departure Time: 16:15
Arrival Date: 2022/12/22
Arrival Time: 17:30
Fare: €6.5 per passenger
Currently Available seats: 1

-------------------------------------
```