Contract CO1: loadConnectionData

Operation: loadConnectionData (filePath)

Cross References: Use Case: Load Records from CSV

Preconditions:

- The system is operational.
- CSV Data Source is present and available to use in the correct structure.
- Admin has access to the data source and permissions to load data.

Postconditions:

- A new Connection is created for every valid CSV record. (instance creation)
- New City objects are created given that the City object has not already been created.
- Train objects are created if a train type was not previously seen. (instance creation)
- Attributes for Connection are set: routeId, departureTime, arrivalTime, daysOfOperation,ticket prices, and train Type. (attribute modification)
- Each Connection is associated with its departure City and arrival City. (assoc. formed)
- Each Connection is associated to a Train according to its trainType. (assoc. formed)

—---

Contract CO2: searchConnections

Operation: searchConnections (departureCity, arrivalCity, departTime, arriveTime, trainType, daysOfOperation)

Cross References: Use Case: Search for Connection

Preconditions:

- The system is operational.
- CSV Data has been loaded into the system memory.

Postconditions:

- For each result that matches with the input parameters, a Trip instance **t** was created. (instance creation)
- For each **t** that requires transfers, 1-3 Segment, **s,** instances were created. (instance creation)
- Each **s** was associated with one matching Connection **c** (train used by the segment). (assoc. formed)
- **t** was composed of its Segment instance(s). (assoc. formed)
- **s**.*connectionDuration* was computed from instance **c**'s arrival and departure times. (attribute modification).
- **t**.*totalTime*, **t**.*totalTransferTime*, and **t**.*price* were computed from the included Segment(s) and selected class prices. (attribute modification).

- Note: If there are no matches, the operator returns an empty list. No instances are created

Contract CO3: sortResults

Operation: sortResults (trip or price)

Cross References: Use Case: Sort Results

Preconditions:

- A list of Trips (non-empty) from the most recent search is available.

Postconditions:

- The list of trip instances was re-ordered according to duration or price.
- No instances were created or deleted.
- No associations were formed or broken.

Contract CO4: selectTrip

Operation: selectedTrip (selectedTrip)

Cross References: Use Case: Book Trip

Preconditions:

- System is operational
- Non-empty list of connections is displayed
- selectedTrip is a valid Trip instance from the list of search results

Postconditions:

- The selected Trip instance is stored/marked as an active booking in the database. (attribute modification)
- The system is ready to accept passenger information

Contract CO4: enterPassengerInfo

Operation: enterPassengerInfo(firstName, lastName, age, id)

Cross References: Use Case: Book Trip

Preconditions:

- The system is operational.
- A Trip has been selected.
- The passenger has provided valid credentials (firstName, lastName, age, id).
- No existing Reservation exists for a Client with the given id on the selected Connection.

Postconditions:

- A new Client instance c was created with the provided attributes. (instance creation)
- A new BookedTrip instance Bt was created. (instance creation)
- Bt was assigned a unique tripId. (attribute modification)
- Bt.travelDate was set to the provided travelDate. (attribute modification)
- Bt.isFirstClass was set based on class selection. (attribute modification)
- Bt was associated with the selected Trip. (association formed)
- A new Reservation instance r was created. (instance creation)
- r was assigned a unique reservationId. (attribute modification)
- r.isFirstClass was set based on class selection. (attribute modification)
- r was associated with the Connection from the selected Trip's first Segment. (association formed)
- r was associated with Client c. (association formed)
- Bt was composed of Reservation r (added to Bt.reservations list). (association formed)
- r was associated back to BookedTrip Bt through setTrip(). (association formed)
- A new Ticket instance t was created for Reservation r. (instance creation)
- t was assigned a unique ticketId. (attribute modification)
- t.firstClassPrice and t.secondClassPrice were computed from the Trip. (attribute modification)

- Bt was added to the TripCollection repository via saveTrip(). (association formed)
- Client c was persisted to the database.
- BookedTrip Bt was persisted to the database via DatabaseHelper.saveTrip() with all attributes (tripId, clientId, departure/arrival cities and times, dates, class, price, duration, stops).

Contract CO5: Validate Connection

Operation: layoverMinutesAcrossDays(firstConn, secondConn, arrivalDay)

Cross References: Use Case: Book a Trip, Use Case: Search for Connection

Preconditions:

- System is operational
- Two consecutive connections exist (firstConn and secondConn)
- Arrival day of week is known
- Days of operation for secondConn are defined

Main Success Scenario:

1. System receives first connection, second connection, and arrival day
2. System calculates arrival time in minutes from firstConn
3. System parses days of operation for secondConn
4. System calculates departure time in minutes from secondConn
5. System loops through next 7 days starting from arrival day:
    a. Check if candidate day is in secondConn's days of operation
    b. If same day (add=0) and departure >= arrival: return depMin - arrMin
    c. If different day and operations match: return (days * 1440) + (depMin - arrMin)

6. System returns calculated layover time in minutes

Postconditions:

- Layover time calculated between connections
- Validates feasibility of transfer based on days of operation
- Returns time in minutes for the transfer/layover period

Contract CO6: viewTrips

Operation: viewTrips(lastName, id)

Cross References: Use Case: View Trips

Preconditions:

- The system is operational.
- Client credentials (lastName and id) are inputted.
- The database contains at least one trip record for the given client.

Postconditions:

- A list of trips matching the client credentials was retrieved from the database.
- The retrieved trips were split into current/future trip lists based on trip dates.
- The trips are displayed to the GUI.