UNIVERSITÀ
DEGLI STUDI
DELL'AQUILA

Model Driven Engineering
2024-2025

Assignment 4

EMF-Based Airport Management System

Name	Matriculation number	Email
Prachi	301124	prachi.prachi@student.univaq.it
Eric Rode	301145	eric.rode@student.univaq.it
David Aron Vigh	301150	davidaron.vigh@student.univaq.it

Introduction

Airport management systems are complex infrastructures that require meticulous design to manage resources such as terminals, airside areas, gates, and flights. Using model-driven engineering (MDE) principles, this project leverages EMF to define a metamodel that encapsulates the airport's operational domain. The metamodel specifies entities, relationships, and constraints, enabling efficient validation and behavior testing.

The system is developed to include derived fields and validation constraints for real-world relevance. For example, the isInternationalattribute of a flight is derived from its associated city, and operations such as findGateByName encapsulate domain-specific logic.

Metamodel Design

The metamodel is specified in Ecore, representing the structural hierarchy of an airport. Key components include:

1. **Terminal**: The central entity managing various functional and spatial components, such as airside areas, landside areas, and transportation services.
2. **Airside and Landside**: Subdivisions of the terminal, responsible for operations such as gate management and passenger transitions.
3. **GateArea**: A crucial component for managing boarding gates and linking flights.
4. **Flight**: Represents individual flights with attributes like flight number, type (arrival or departure), and derived attributes like isInternational.
5. **Validation Rules**: Integrated into the metamodel using Object Constraint Language (OCL). For instance, ValidTime ensures that flight times conform to the HH:mm format.

To validate its applicability, the metamodel is instantiated in two implementations:

1. **Debrecen International Airport**: This scenario emphasizes features such as multiple floors, Schengen and non-Schengen gate areas, and comprehensive landside and airside facilities.
2. **Pescara Airport Terminal 1**: This implementation highlights a simpler structure with a focus on a single-floor terminal, basic gate area configurations, and essential transportation services.

Implementation

Model Instantiation and Object Validation

The system uses the EMF-generated factory classes to instantiate model elements. For instance, the AirportFactory is utilized to create Terminal, GateArea, Flight, and other entities. Object validation is performed using the EMF Diagnostician to ensure compliance with defined constraints.

The following code illustrates the creation and validation of model objects:

```
Terminal terminal = factory.createTerminal();
terminal.setName("Main Terminal");
validateObject(terminal);
```

Validation results are presented using diagnostic messages. This ensures that all objects conform to their constraints before deployment.

Derived Attributes and Operations

The metamodel includes derived attributes such as isInternational in the Flight class, which evaluates whether a flight is international based on its associated city. Additionally, the GateArea class features operations like calculateTotalGates and findGateByName, implemented in Java for runtime execution.

For example, the findGateByName operation searches for a gate with a specific name within a GateArea:

```
Gate foundGate = gateArea.findGateByName("Gate A");
```

Testing Derived Fields

A test suite validates the correctness of derived attributes and operations. The DerivedFieldsTest class demonstrates how changes in the model impact derived fields and ensures they behave as expected.

```
flight.setCity("LocalCity");
System.out.println("Is International: " + flight.isIsInternational()); // Expected: false
```

Behavior Testing

Functional behavior is tested using unit tests, such as validating the total number of gates in a GateArea and searching for specific gates. The GateAreaTest class ensures that methods like calculateTotalGates yield accurate results.

Results and Validation

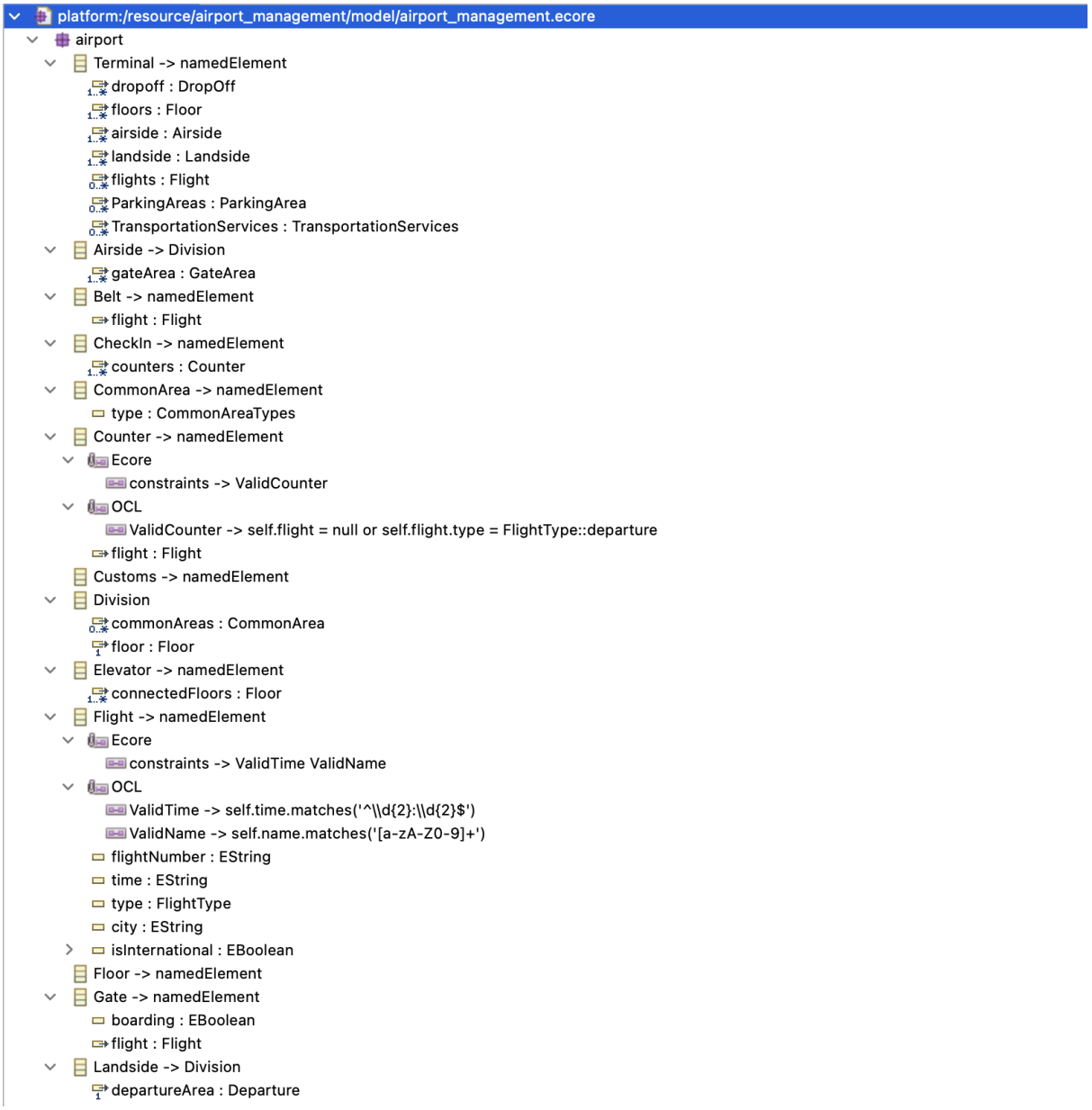
The system effectively demonstrates the utility of EMF for modeling complex domains. Validation tests confirm that constraints such as flight time format and valid counter associations are enforced. Derived attributes provide real-time insights into the system's state, ensuring accurate and consistent behavior.

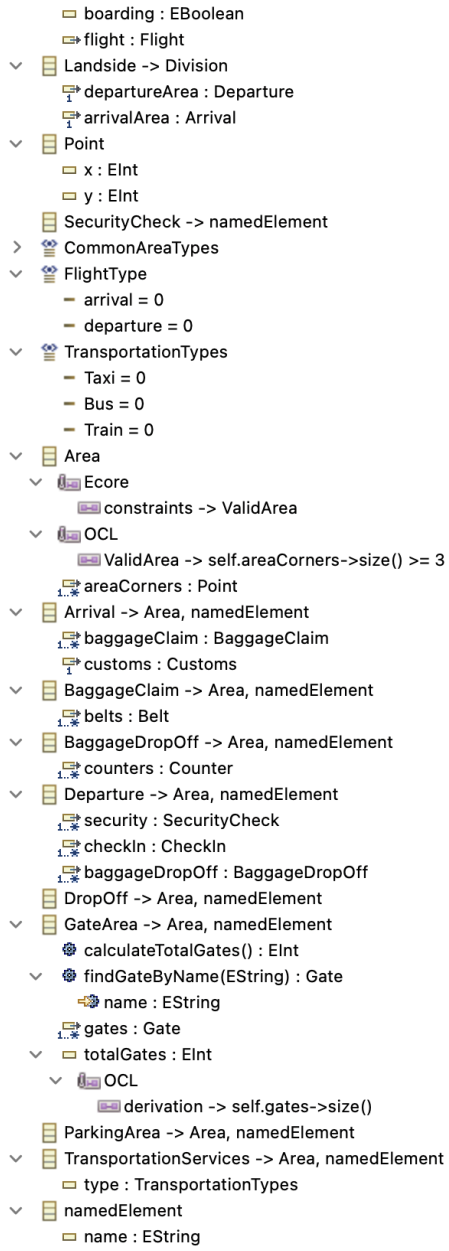
Testing the GateArea operations resulted in the following outcomes:

- calculateTotalGates correctly counted gates.
- findGateByName successfully located gates by their names.

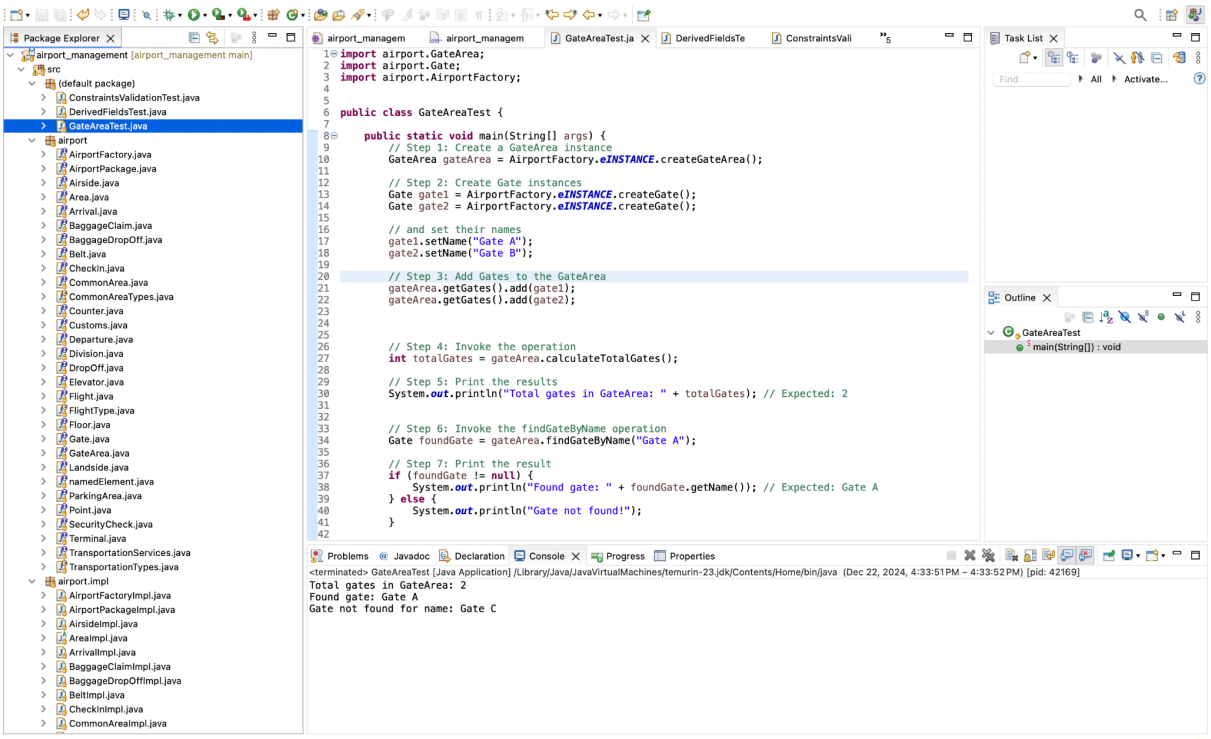
Appendix

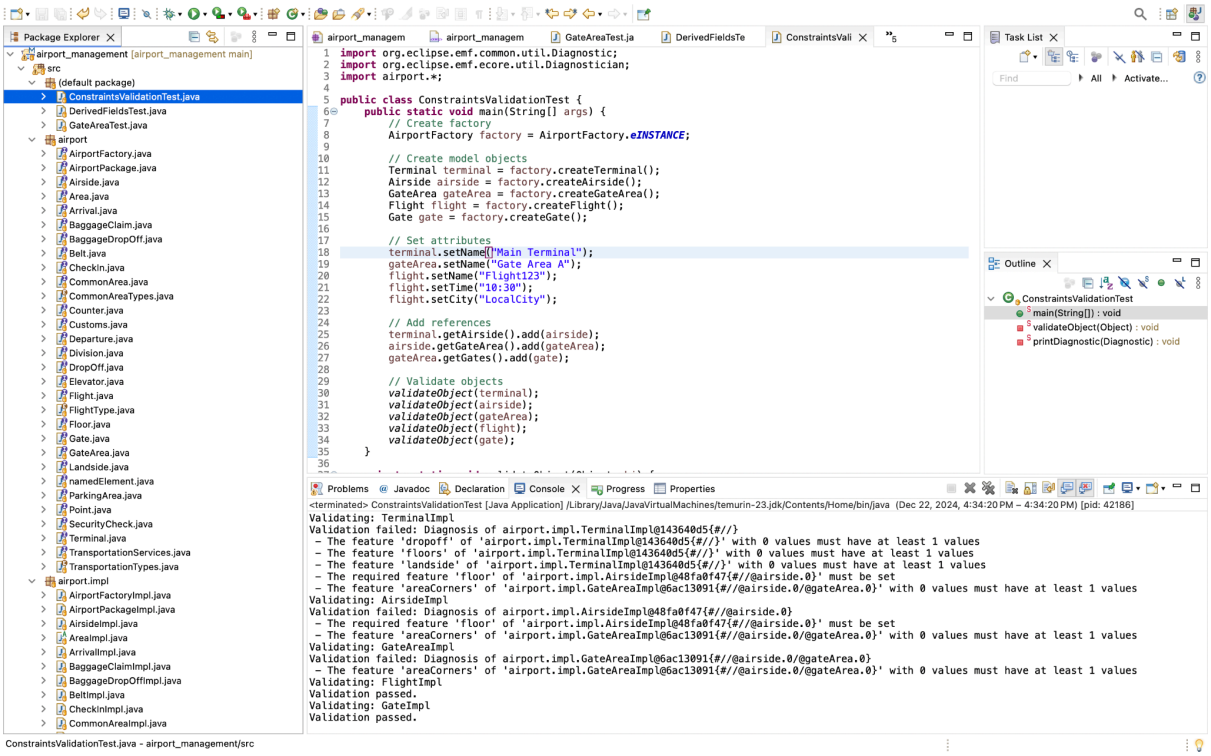
Metamodel

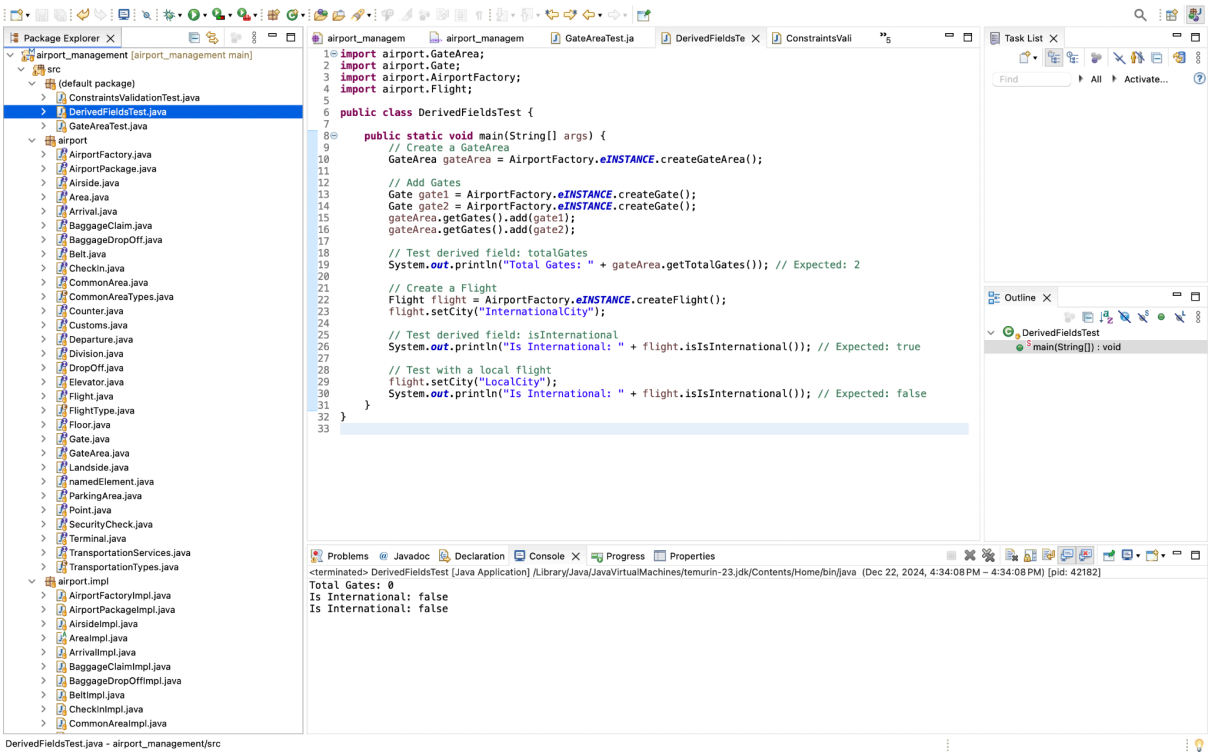

platform:/resource/airport_management/model/airport_management.ecore
airport
Terminal -> namedElement
dropoff : DropOff
floors : Floor
airside : Airside
landside : Landside
flights : Flight
ParkingAreas : ParkingArea
TransportationServices : TransportationServices
Airside -> Division
gateArea : GateArea
Belt -> namedElement
flight : Flight
CheckIn -> namedElement
counters : Counter
CommonArea -> namedElement
type : CommonAreaTypes
Counter -> namedElement
Ecore
constraints -> ValidCounter
OCL
ValidCounter -> self.flight = null or self.flight.type = FlightType::departure
flight : Flight
Customs -> namedElement
Division
commonAreas : CommonArea
floor : Floor
Elevator -> namedElement
connectedFloors : Floor
Flight -> namedElement
Ecore
constraints -> ValidTime ValidName
OCL
ValidTime -> self.time.matches('^\\d{2}:\\d{2}$')
ValidName -> self.name.matches('[a-zA-Z0-9]+')
flightNumber : EString
time : EString
type : FlightType
city : EString
isInternational : EBoolean
Floor -> namedElement
Gate -> namedElement
boarding : EBoolean
flight : Flight
Landside -> Division
departureArea : Departure


boarding : EBoolean
flight : Flight
Landside -> Division
departureArea : Departure
arrivalArea : Arrival
Point
x : EInt
y : EInt
SecurityCheck -> namedElement
CommonAreaTypes
FlightType
arrival = 0
departure = 0
TransportationTypes
Taxi = 0
Bus = 0
Train = 0
Area
Ecore
constraints -> ValidArea
OCL
ValidArea -> self.areaCorners->size() >= 3
areaCorners : Point
Arrival -> Area, namedElement
baggageClaim : BaggageClaim
customs : Customs
BaggageClaim -> Area, namedElement
belts : Belt
BaggageDropOff -> Area, namedElement
counters : Counter
Departure -> Area, namedElement
security : SecurityCheck
checkIn : CheckIn
baggageDropOff : BaggageDropOff
DropOff -> Area, namedElement
GateArea -> Area, namedElement
calculateTotalGates() : EInt
findGateByName(EString) : Gate
name : EString
gates : Gate
totalGates : EInt
OCL
derivation -> self.gates->size()
ParkingArea -> Area, namedElement
TransportationServices -> Area, namedElement
type : TransportationTypes
namedElement
name : EString

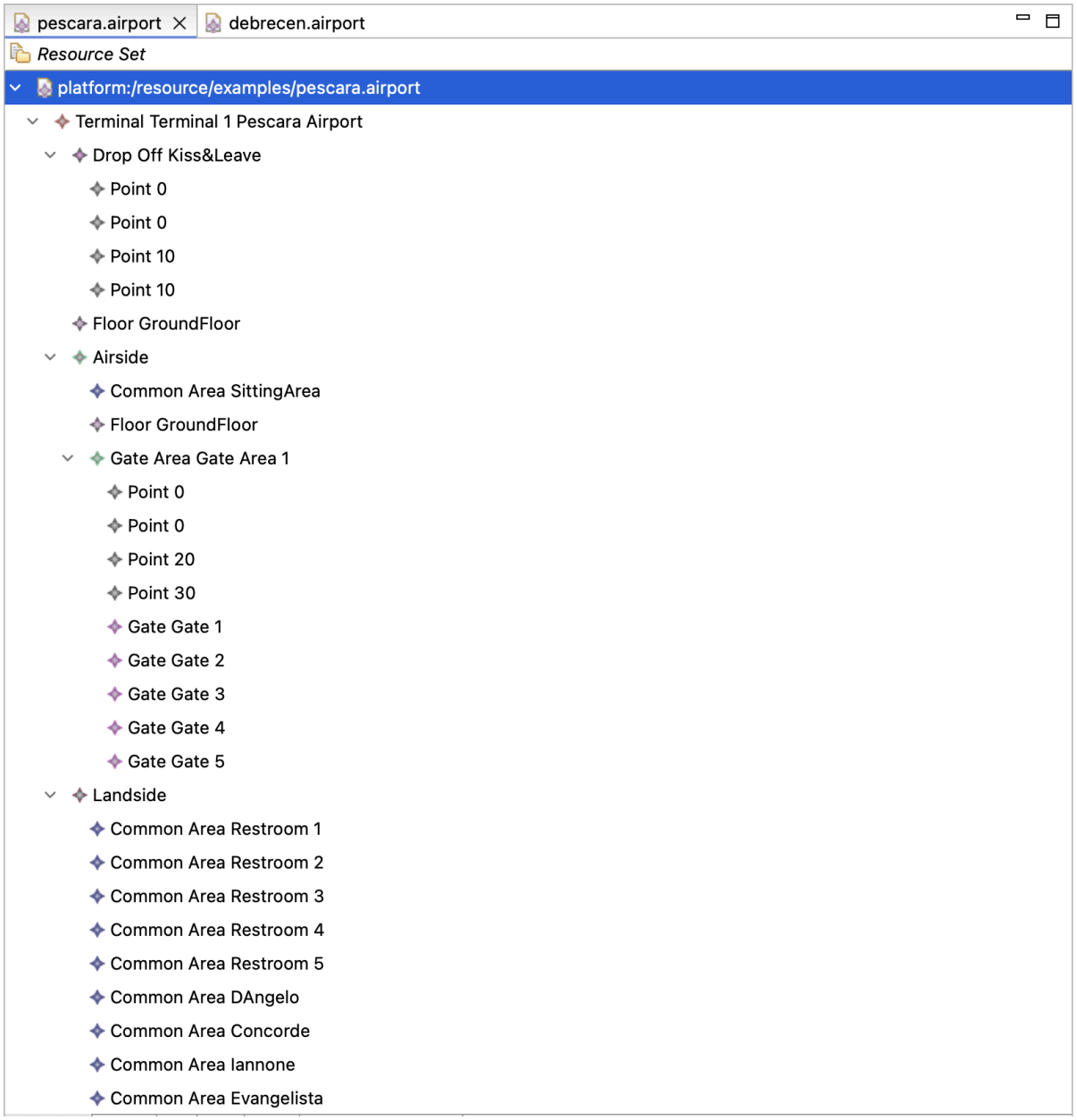
Implemented Tests

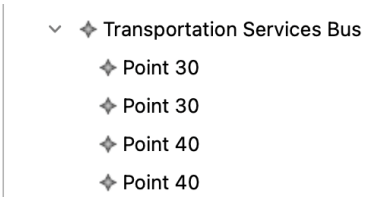

Package Explorer
airport_management [airport_management main]
src
(default package)
ConstraintsValidationTest.java
DerivedFieldsTest.java
GateAreaTest.java
airport
AirportFactory.java
AirportPackage.java
Airside.java
Area.java
Arrival.java
BaggageClaim.java
BaggageDropOff.java
Belt.java
CheckIn.java
CommonArea.java
CommonAreaTypes.java
Counter.java
Customs.java
Departure.java
Division.java
DropOff.java
Elevator.java
Flight.java
FlightType.java
Floor.java
Gate.java
GateArea.java
Landside.java
namedElement.java
ParkingArea.java
Point.java
SecurityCheck.java
Terminal.java
TransportationServices.java
TransportationTypes.java
airport.impl
AirportFactoryImpl.java
AirportPackageImpl.java
AirsideImpl.java
AreaImpl.java
ArrivalImpl.java
BaggageClaimImpl.java
BaggageDropOffImpl.java
BeltImpl.java
CheckInImpl.java
CommonAreaImpl.java
airport_managem
airport_managem
GateAreaTest.ja
DerivedFieldsTe
ConstraintsVali
5
1 import airport.GateArea;
2 import airport.Gate;
3 import airport.AirportFactory;
4
5
6 public class GateAreaTest {
7
8 public static void main(String[] args) {
9 // Step 1: Create a GateArea instance
10 GateArea gateArea = AirportFactory.eINSTANCE.createGateArea();
11
12 // Step 2: Create Gate instances
13 Gate gate1 = AirportFactory.eINSTANCE.createGate();
14 Gate gate2 = AirportFactory.eINSTANCE.createGate();
15
16 // and set their names
17 gate1.setName("Gate A");
18 gate2.setName("Gate B");
19
20 // Step 3: Add Gates to the GateArea
21 gateArea.getGates().add(gate1);
22 gateArea.getGates().add(gate2);
23
24
25
26 // Step 4: Invoke the operation
27 int totalGates = gateArea.calculateTotalGates();
28
29 // Step 5: Print the results
30 System.out.println("Total gates in GateArea: " + totalGates); // Expected: 2
31
32
33 // Step 6: Invoke the findGateByName operation
34 Gate foundGate = gateArea.findGateByName("Gate A");
35
36 // Step 7: Print the result
37 if (foundGate != null) {
38 System.out.println("Found gate: " + foundGate.getName()); // Expected: Gate A
39 } else {
40 System.out.println("Gate not found!");
41 }
42
Task List
Find
All
Activate...
Outline
GateAreaTest
main(String[]) : void
Problems
Javadoc
Declaration
Console
Progress
Properties
<terminated> GateAreaTest [Java Application] /Library/Java/JavaVirtualMachines/temurin-23.jdk/Contents/Home/bin/java (Dec 22, 2024, 4:33:51 PM – 4:33:52 PM) [pid: 42169]
Total gates in GateArea: 2
Found gate: Gate A
Gate not found for name: Gate C
GateAreaTest.java - airport_management/src


Package Explorer
airport_management [airport_management main]
src
(default package)
ConstraintsValidationTest.java
DerivedFieldsTest.java
GateAreaTest.java
airport
AirportFactory.java
AirportPackage.java
Airside.java
Area.java
Arrival.java
BaggageClaim.java
BaggageDropOff.java
Belt.java
CheckIn.java
CommonArea.java
CommonAreaTypes.java
Counter.java
Customs.java
Departure.java
Division.java
DropOff.java
Elevator.java
Flight.java
FlightType.java
Floor.java
Gate.java
GateArea.java
Landside.java
namedElement.java
ParkingArea.java
Point.java
SecurityCheck.java
Terminal.java
TransportationServices.java
TransportationTypes.java
airport.impl
AirportFactoryImpl.java
AirportPackageImpl.java
AirsideImpl.java
AreaImpl.java
ArrivalImpl.java
BaggageClaimImpl.java
BaggageDropOffImpl.java
BeltImpl.java
CheckInImpl.java
CommonAreaImpl.java
airport_managem
airport_managem
GateAreaTest.ja
DerivedFieldsTe
ConstraintsVali
5
1 import org.eclipse.emf.common.util.Diagnostic;
2 import org.eclipse.emf.ecore.util.Diagnostician;
3 import airport.*;
4
5 public class ConstraintsValidationTest {
6 public static void main(String[] args) {
7 // Create factory
8 AirportFactory factory = AirportFactory.eINSTANCE;
9
10 // Create model objects
11 Terminal terminal = factory.createTerminal();
12 Airside airside = factory.createAirside();
13 GateArea gateArea = factory.createGateArea();
14 Flight flight = factory.createFlight();
15 Gate gate = factory.createGate();
16
17 // Set attributes
18 terminal.setName("Main Terminal");
19 gateArea.setName("Gate Area A");
20 flight.setName("Flight123");
21 flight.setTime("10:30");
22 flight.setCity("LocalCity");
23
24 // Add references
25 terminal.getAirside().add(airside);
26 airside.getGateArea().add(gateArea);
27 gateArea.getGates().add(gate);
28
29 // Validate objects
30 validateObject(terminal);
31 validateObject(airside);
32 validateObject(gateArea);
33 validateObject(flight);
34 validateObject(gate);
35 }
36
Task List
Find
All
Activate...
Outline
ConstraintsValidationTest
main(String[]) : void
validateObject(Object) : void
printDiagnostic(Diagnostic) : void
Problems
Javadoc
Declaration
Console
Progress
Properties
<terminated> ConstraintsValidationTest [Java Application] /Library/Java/JavaVirtualMachines/temurin-23.jdk/Contents/Home/bin/java (Dec 22, 2024, 4:34:20 PM – 4:34:20 PM) [pid: 42186]
Validating: TerminalImpl
Validation failed: Diagnosis of airport.impl.TerminalImpl@143640d5{#//}
- The feature 'dropoff' of 'airport.impl.TerminalImpl@143640d5{#//}' with 0 values must have at least 1 values
- The feature 'floors' of 'airport.impl.TerminalImpl@143640d5{#//}' with 0 values must have at least 1 values
- The feature 'landside' of 'airport.impl.TerminalImpl@143640d5{#//}' with 0 values must have at least 1 values
- The required feature 'floor' of 'airport.impl.AirsideImpl@48fa0f47{#//@airside.0}' must be set
- The feature 'areaCorners' of 'airport.impl.GateAreaImpl@6ac13091{#//@airside.0/@gateArea.0}' with 0 values must have at least 1 values
Validating: AirsideImpl
Validation failed: Diagnosis of airport.impl.AirsideImpl@48fa0f47{#//@airside.0}
- The required feature 'floor' of 'airport.impl.AirsideImpl@48fa0f47{#//@airside.0}' must be set
- The feature 'areaCorners' of 'airport.impl.GateAreaImpl@6ac13091{#//@airside.0/@gateArea.0}' with 0 values must have at least 1 values
Validating: GateAreaImpl
Validation failed: Diagnosis of airport.impl.GateAreaImpl@6ac13091{#//@airside.0/@gateArea.0}
- The feature 'areaCorners' of 'airport.impl.GateAreaImpl@6ac13091{#//@airside.0/@gateArea.0}' with 0 values must have at least 1 values
Validating: FlightImpl
Validation passed.
Validating: GateImpl
Validation passed.
ConstraintsValidationTest.java - airport_management/src


Package Explorer
airport_management [airport_management main]
src
(default package)
ConstraintsValidationTest.java
DerivedFieldsTest.java
GateAreaTest.java
airport
AirportFactory.java
AirportPackage.java
Airside.java
Area.java
Arrival.java
BaggageClaim.java
BaggageDropOff.java
Belt.java
CheckIn.java
CommonArea.java
CommonAreaTypes.java
Counter.java
Customs.java
Departure.java
Division.java
DropOff.java
Elevator.java
Flight.java
FlightType.java
Floor.java
Gate.java
GateArea.java
Landside.java
namedElement.java
ParkingArea.java
Point.java
SecurityCheck.java
Terminal.java
TransportationServices.java
TransportationTypes.java
airport.impl
AirportFactoryImpl.java
AirportPackageImpl.java
AirsideImpl.java
AreaImpl.java
ArrivalImpl.java
BaggageClaimImpl.java
BaggageDropOffImpl.java
BeltImpl.java
CheckInImpl.java
CommonAreaImpl.java
airport_managem
airport_managem
GateAreaTest.ja
DerivedFieldsTe
ConstraintsVali
5
1 import airport.GateArea;
2 import airport.Gate;
3 import airport.AirportFactory;
4 import airport.Flight;
5
6 public class DerivedFieldsTest {
7
8 public static void main(String[] args) {
9 // Create a GateArea
10 GateArea gateArea = AirportFactory.eINSTANCE.createGateArea();
11
12 // Add Gates
13 Gate gate1 = AirportFactory.eINSTANCE.createGate();
14 Gate gate2 = AirportFactory.eINSTANCE.createGate();
15 gateArea.getGates().add(gate1);
16 gateArea.getGates().add(gate2);
17
18 // Test derived field: totalGates
19 System.out.println("Total Gates: " + gateArea.getTotalGates()); // Expected: 2
20
21 // Create a Flight
22 Flight flight = AirportFactory.eINSTANCE.createFlight();
23 flight.setCity("InternationalCity");
24
25 // Test derived field: isInternational
26 System.out.println("Is International: " + flight.isIsInternational()); // Expected: true
27
28 // Test with a local flight
29 flight.setCity("LocalCity");
30 System.out.println("Is International: " + flight.isIsInternational()); // Expected: false
31 }
32 }
33
Task List
Find
All
Activate...
Outline
DerivedFieldsTest
main(String[]) : void
Problems
Javadoc
Declaration
Console
Progress
Properties
<terminated> DerivedFieldsTest [Java Application] /Library/Java/JavaVirtualMachines/temurin-23.jdk/Contents/Home/bin/java (Dec 22, 2024, 4:34:08 PM – 4:34:08 PM) [pid: 42182]
Total Gates: 0
Is International: false
Is International: false
DerivedFieldsTest.java - airport_management/src

Implemented Models

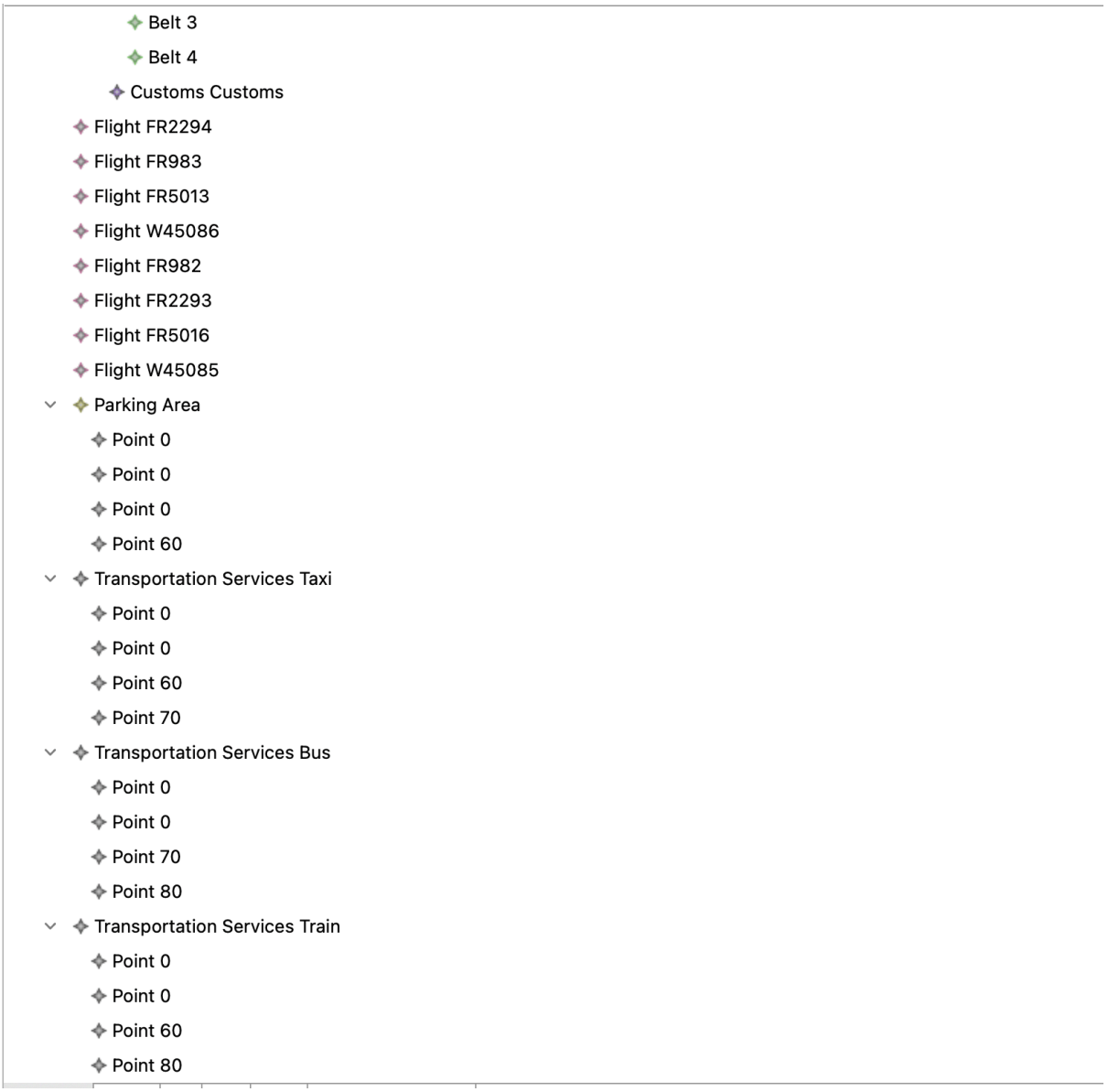




- Baggage Claim Schengen arrival
  - Point 80
  - Point 80
  - Point 100
  - Point 100
  - Belt 1
- Baggage Claim Non-Schengen arrival
  - Point 100
  - Point 100
  - Point 110
  - Point 110
  - Belt 2
- Customs Customs
- Flight LH1687
- Flight IZ702
- Flight W92701
- Flight D4703
- Flight LH1686
- Flight IZ701
- Flight W92702
- Parking Area Main Parking
  - Point 0
  - Point 20
  - Point 20
  - Point 0
- Transportation Services Taxi
  - Point 20
  - Point 20
  - Point 30
  - Point 30

Landside
Departure Main Departure
Point 40
Point 40
Point 60
Point 60
Security Check Passesnger Security Check
Check In Check-in
Counter 1
Counter 2
Counter 3
Counter 4
Counter 5
Counter 6
Baggage Drop Off DropOff
Point 40
Point 40
Point 60
Point 60
Counter 1
Counter 2
Counter 3
Counter 4
Counter 5
Counter 6
Arrival Main arrival
Point 60
Point 60
Point 80
Point 80

platform:/resource/examples/debrecen.airport
Terminal Debrecen International
Drop Off Repuloter Parkolo
Point 0
Point 0
Point 10
Point 10
Floor Ground Floor
Floor First Floor
Airside
Common Area Waiting Area
Common Area Toilet 3
Common Area Toilet 4
Common Area Toilet 5
Common Area Shop
Common Area Snack Bar
Floor Ground Floor
Gate Area Schengen Countries
Point 30
Point 30
Point 50
Point 50
Gate 1
Gate 2
Gate Area Non-Schengen Countries
Point 50
Point 50
Point 70
Point 70
Gate 3
Gate 4


Belt 3
Belt 4
Customs Customs
Flight FR2294
Flight FR983
Flight FR5013
Flight W45086
Flight FR982
Flight FR2293
Flight FR5016
Flight W45085
Parking Area
Point 0
Point 0
Point 0
Point 60
Transportation Services Taxi
Point 0
Point 0
Point 60
Point 70
Transportation Services Bus
Point 0
Point 0
Point 70
Point 80
Transportation Services Train
Point 0
Point 0
Point 60
Point 80

Common Area SittingArea
Floor GroundFloor
Departure Departure 1
Point 0
Point 0
Point 20
Point 20
Security Check Security
Check In CheckIn
Counter 1
Counter 2
Counter 3
Baggage Drop Off
Point 0
Point 0
Point 20
Point 20
Counter 1
Counter 2
Counter 3
Counter 4
Arrival Arrival 1
Point 30
Point 50
Point 30
Point 50
Baggage Claim BC 1
Point 30
Point 50
Point 30
Point 50