Travel system

Łukasz Strep s20351

1. User requirements.

Customer:

1. As a customer, I want to review all the trips so I know which one is the best for me and where I would like to go
2. As a customer, I want to see all specific details of a trip so I will know if it's suitable for me
3. As a customer, I want to know the price to know if I can afford it
4. As a customer, I want to know dates of the specific trip to know if they are good for me
5. As a customer, I want to make a booking so I will be able to go on a trip
6. As a customer, I want to add a transport option to the booking so I will be able to get to the given destination
7. As a customer, I want to know accommodation assigned to a trip so I will decide if it's appropriate for me
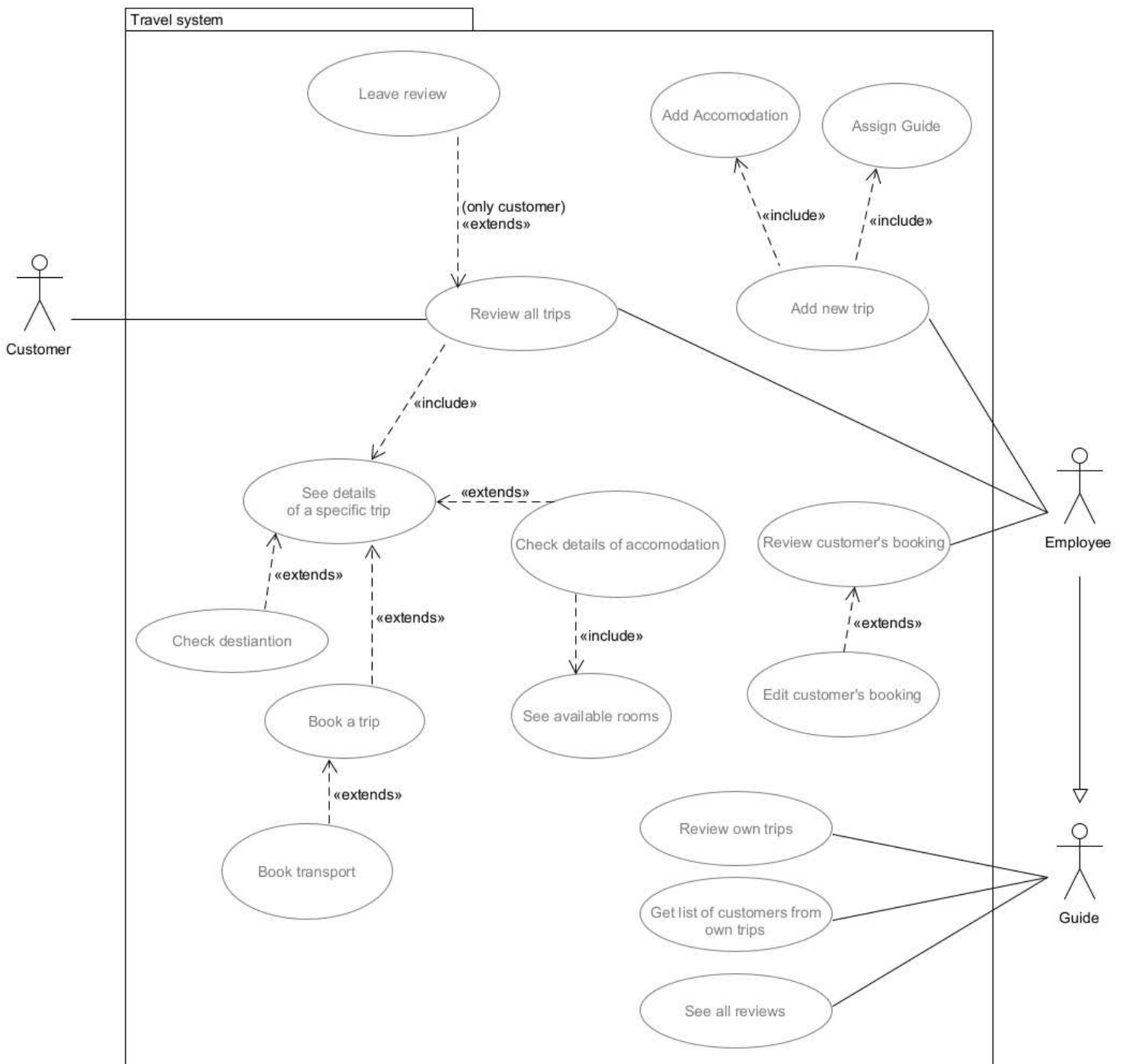8. As a customer, I want to leave a review of a trip so other people will know if it's worth going there

Employee:

1. As an employee, I want to edit customer's booking so I can change any mistakes made by him
2. As an employee, I want to review trips so I can give an advice the customer
3. As an employee, I want to add new trips to the database so clients will have bigger choice
4. As an employee, I want to assign guides to the trips so they will know when they work
5. As an employee, I want to review all the trips which are to the given destination so I can recommend some of them to the clients with preferences
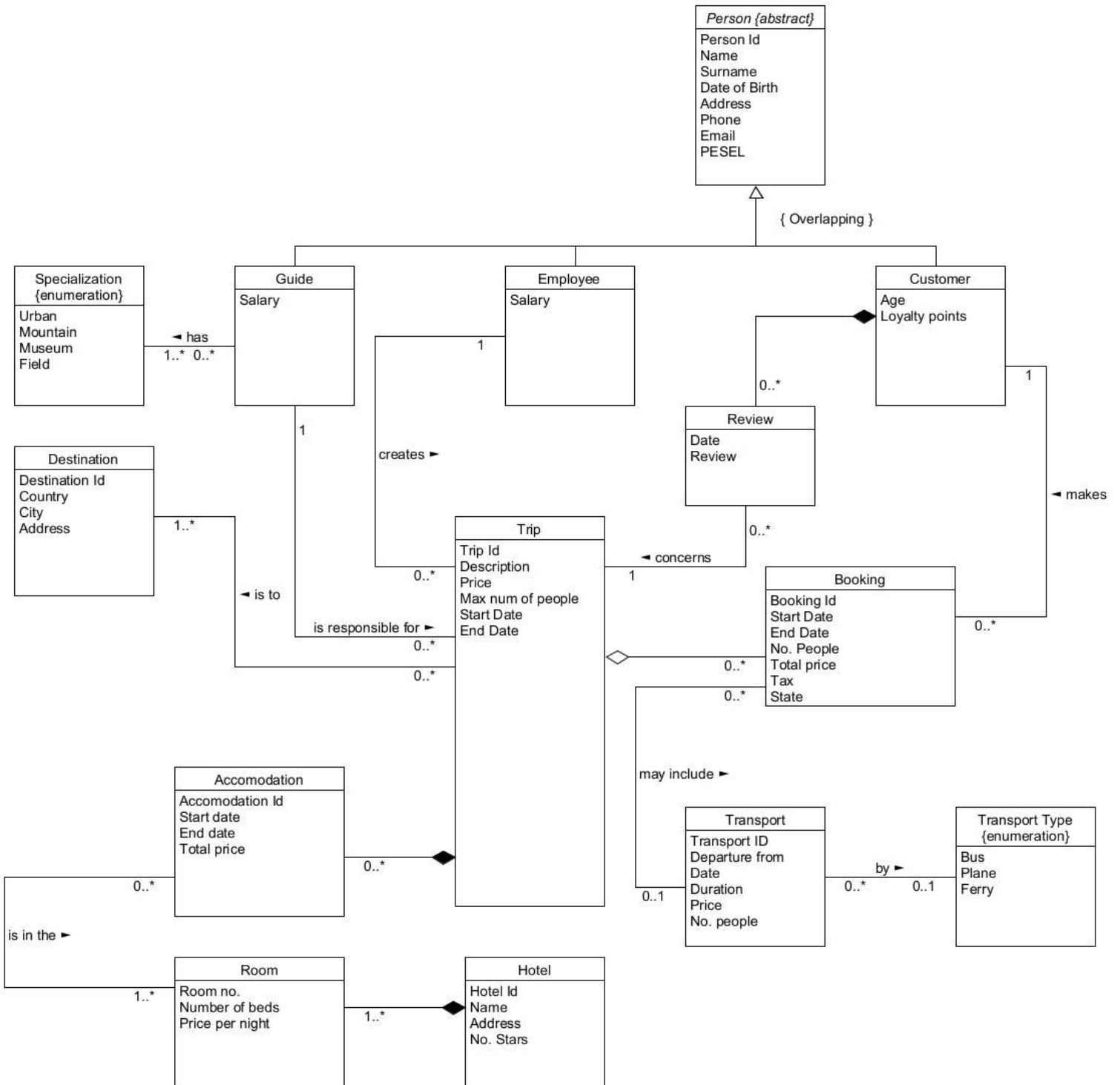
Guide:

1. As a guide, I want to review my trips so I will know when I work
2. As a guide, I want to get a list of all the customers going on a trip with me so I will be able to prepare materials
3. As I guide, I want to see all the reviews left by clients so I will know what to improve
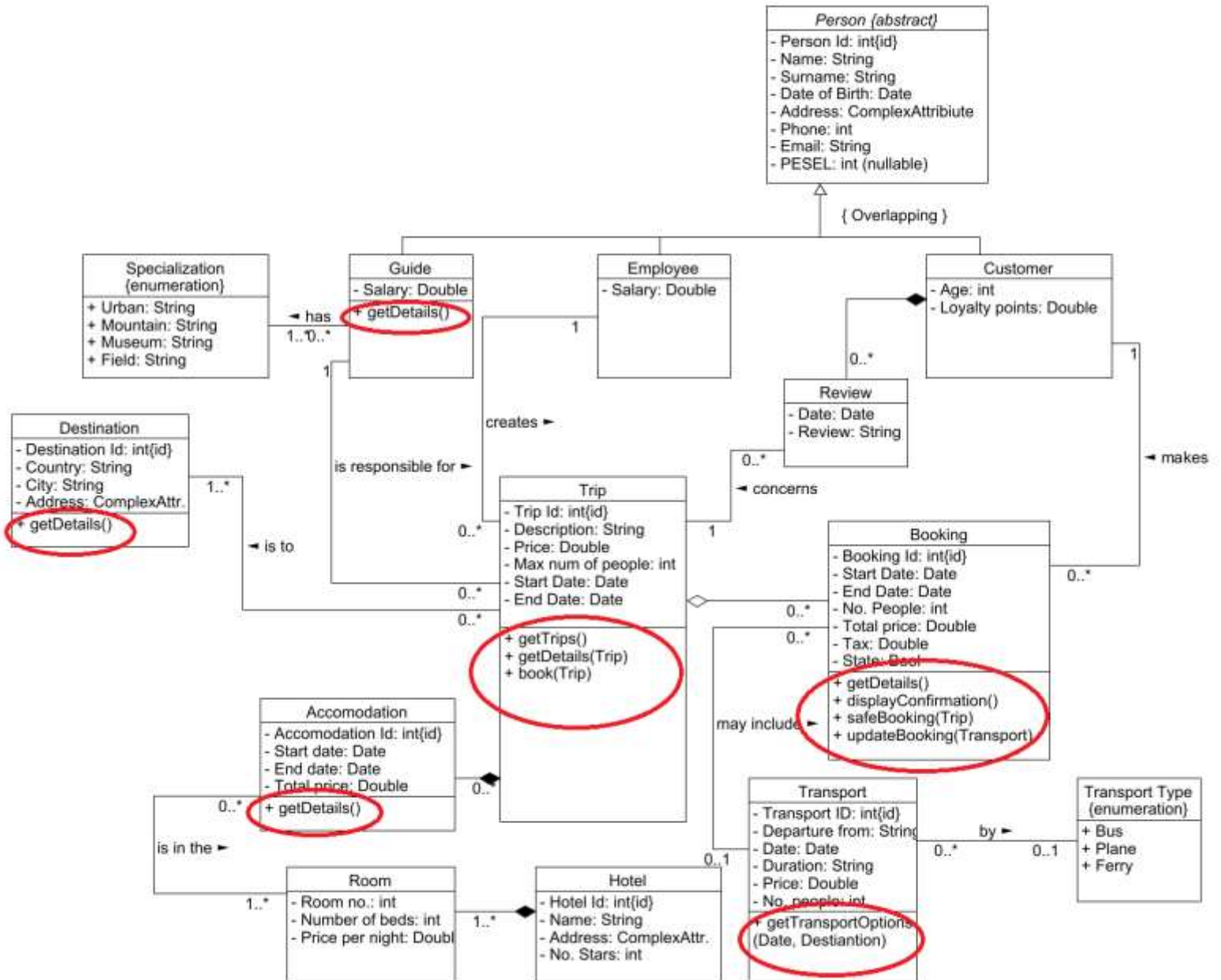
2. Use Case Diagram.

3. Analytical Class Diagram

Person {abstract}
- Person Id
- Name
- Surname
- Date of Birth
- Address
- Phone
- Email
- PESEL

{ Overlapping }

Specialization {enumeration}
- Urban
- Mountain
- Museum
- Field

Guide
- Salary

Employee
- Salary

Customer
- Age
- Loyalty points

◄ has
1..* 0..*

1

0..*

1

Review
- Date
- Review

creates ►

Destination
- Destination Id
- Country
- City
- Address

1..*

◄ is to

is responsible for ►
0..*

0..*

Trip
- Trip Id
- Description
- Price
- Max num of people
- Start Date
- End Date

◄ concerns
1

0..*

◄ makes

Booking
- Booking Id
- Start Date
- End Date
- No. People
- Total price
- Tax
- State

0..*

0..*

0..*

Accomodation
- Accomodation Id
- Start date
- End date
- Total price

0..*

0..*

may include ►

Transport
- Transport ID
- Departure from
- Date
- Duration
- Price
- No. people

0..1

by ►
0..*   0..1

Transport Type {enumeration}
- Bus
- Plane
- Ferry

is in the ►

0..*

Room
- Room no.
- Number of beds
- Price per night

1..*

1..*

Hotel
- Hotel Id
- Name
- Address
- No. Stars

4. Implementation/final class diagram.

Person (abstract)
- Person Id: int{id}
- Name: String
- Surname: String
- Date of Birth: Date
- Address: ComplexAttribiute
- Phone: int
- Email: String
- PESEL: int (nullable)

{ Overlapping }

Specialization (enumeration)
+ Urban: String
+ Mountain: String
+ Museum: String
+ Field: String

Guide
- Salary: Double
+ getDetails()

◄ has   1.. 0..*

Employee
- Salary: Double

1

Customer
- Age: int
- Loyalty points: Double

0..*   1

Review
- Date: Date
- Review: String

0..*   ◄ makes

Destination
- Destination Id: int{id}
- Country: String
- City: String
- Address: ComplexAttr.
+ getDetails()

1..*   ◄ is to

creates ►

is responsible for ►

0..*   ◄ concerns

Trip
- Trip Id: int{id}
- Description: String
- Price: Double
- Max num of people: int
- Start Date: Date
- End Date: Date
+ getTrips()
+ getDetails(Trip)
+ book(Trip)

1

0..*
0..*
0..*

Booking
- Booking Id: int{id}
- Start Date: Date
- End Date: Date
- No. People: int
- Total price: Double
- Tax: Double
- State: Bool
+ getDetails()
+ displayConfirmation()
+ safeBooking(Trip)
+ updateBooking(Transport)

0..*
0..*
0..*

may include ►

Accomodation
- Accomodation Id: int{id}
- Start date: Date
- End date: Date
- Total price: Double
+ getDetails()

0..*   0..

is in the ►

Room
- Room no.: int
- Number of beds: int
- Price per night: Doubl

1..*

Hotel
- Hotel Id: int{id}
- Name: String
- Address: ComplexAttr.
- No. Stars: int

1..*

0..1

Transport
- Transport ID: int{id}
- Departure from: String
- Date: Date
- Duration: String
- Price: Double
- No. people: int
+ getTransportOptions (Date, Destiantion)

by ►   0..*   0..1

Transport Type (enumeration)
+ Bus
+ Plane
+ Ferry

5. Use case scenario.

|	
Use Case ID	UC1
Name of use case scenario	Booking a trip
Actors	Customer
Initial conditions	Customers goes to a section with trips
Final conditions	Customer booked a trip which is available for viewing in the database
Result	Customer can go on a trip
Main scenario	1. Use case begins when customer clicks on a list with available trips. 2. Customers sees list showing all trips and picks one which is the most interesting for him 3. Customer sees all the details of a chosen trip, which include maximum number of people, price, destination, and a guide. 4. Customer is interested so he books a trip 5. Customer needs to be transported to the place of the trip, so he chooses way of transport out of available options provided by external partner 6. System displays confirmation of a booking
Alternative scenario	1. Use case begins when customer clicks on a list with available trips 2. Customers sees list showing all trips and picks one which is the most interesting for him 3. Customer checks details of provided accommodation 4. Customer sees which room he would receive if he would decide to go on a trip 5. Customer goes back to details about trip 6. Customer checks details about destination 7. Customer goes back to details about trip

	8. Customer books a trip and it ends this use case
Exception scenarios	No trips are in the database: 1. System displays information about the lack of trips There are no transports to the starting point of a trip: 1. System displays information about the lack of transport to the starting point of a trip There are no places to go on a trip 1. System displays information that there is no available places to go on a trip

6. Activity Diagram for Use Case.

Activity diagram for the main scenario from the use case scenario. Assumes that there are available trips in the database and there is a way of transport to the starting point of a trip.
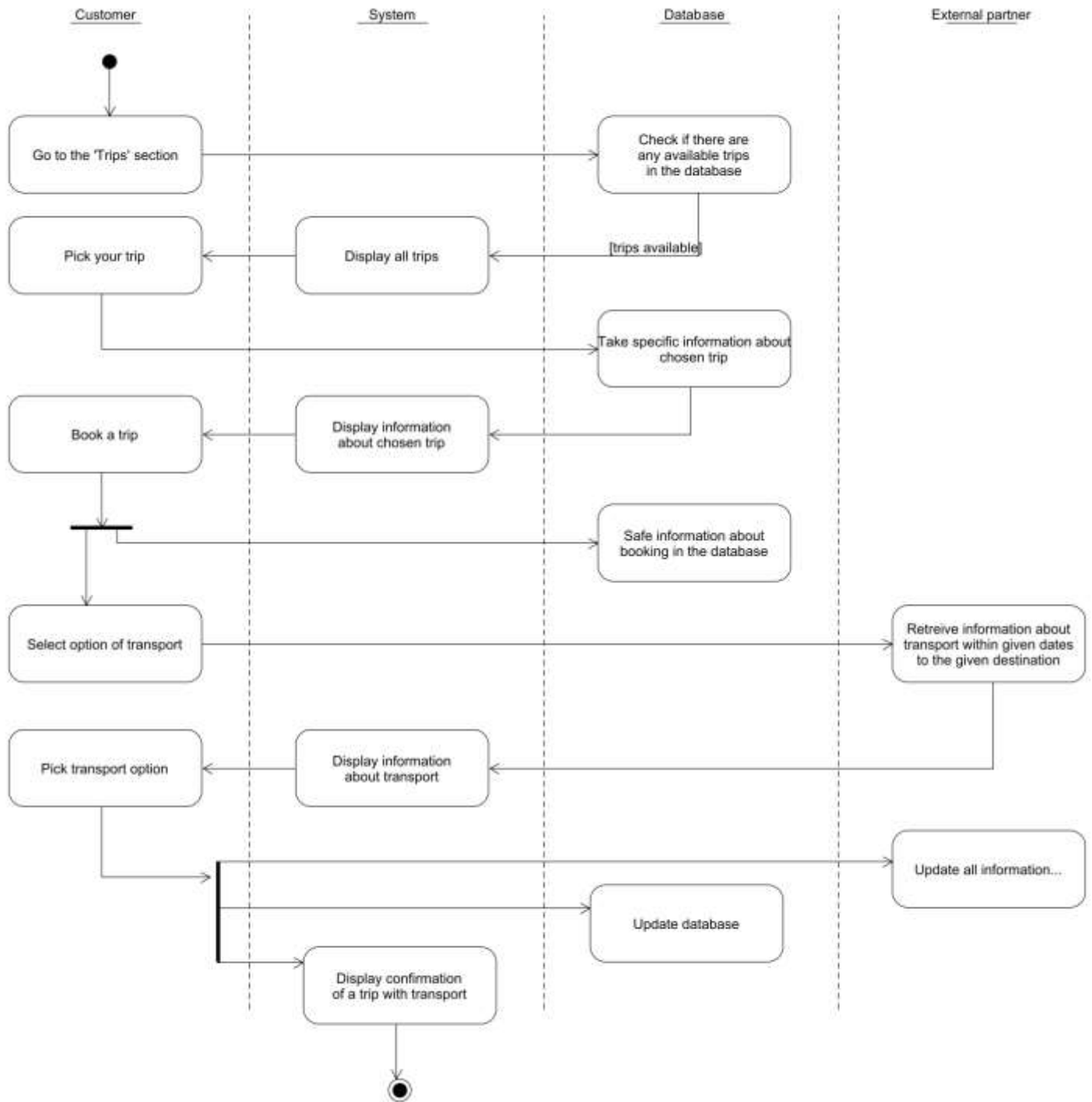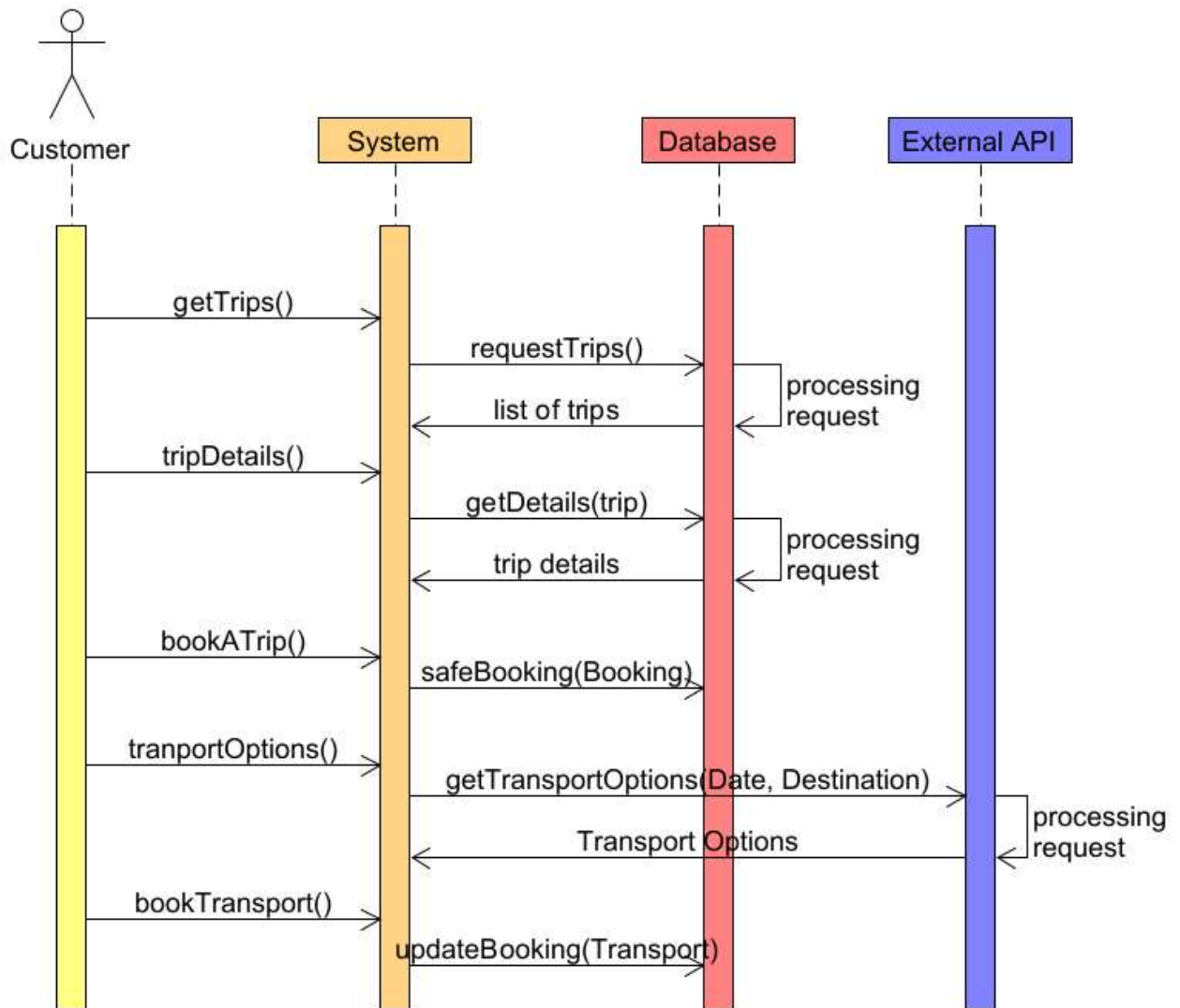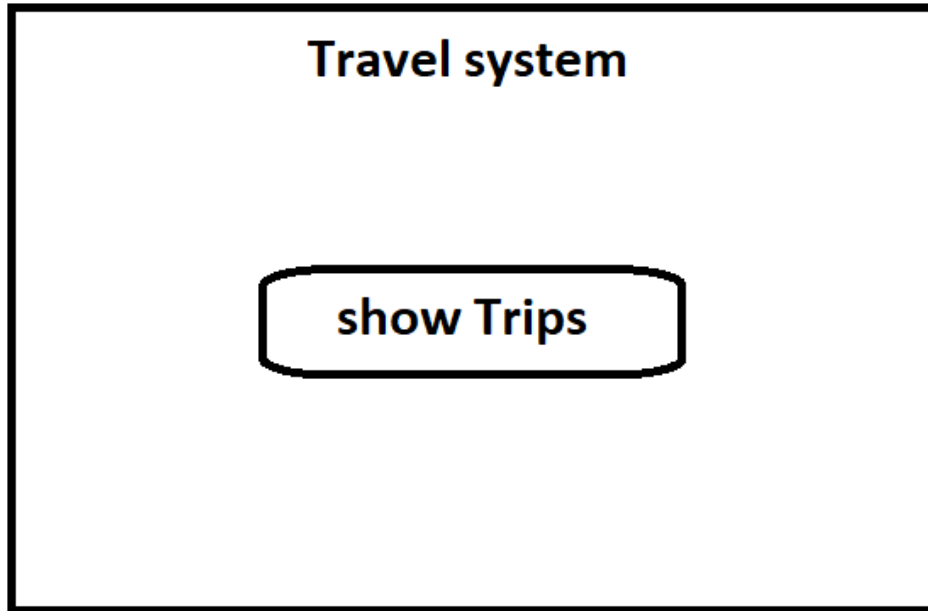
7. State diagram for a class.
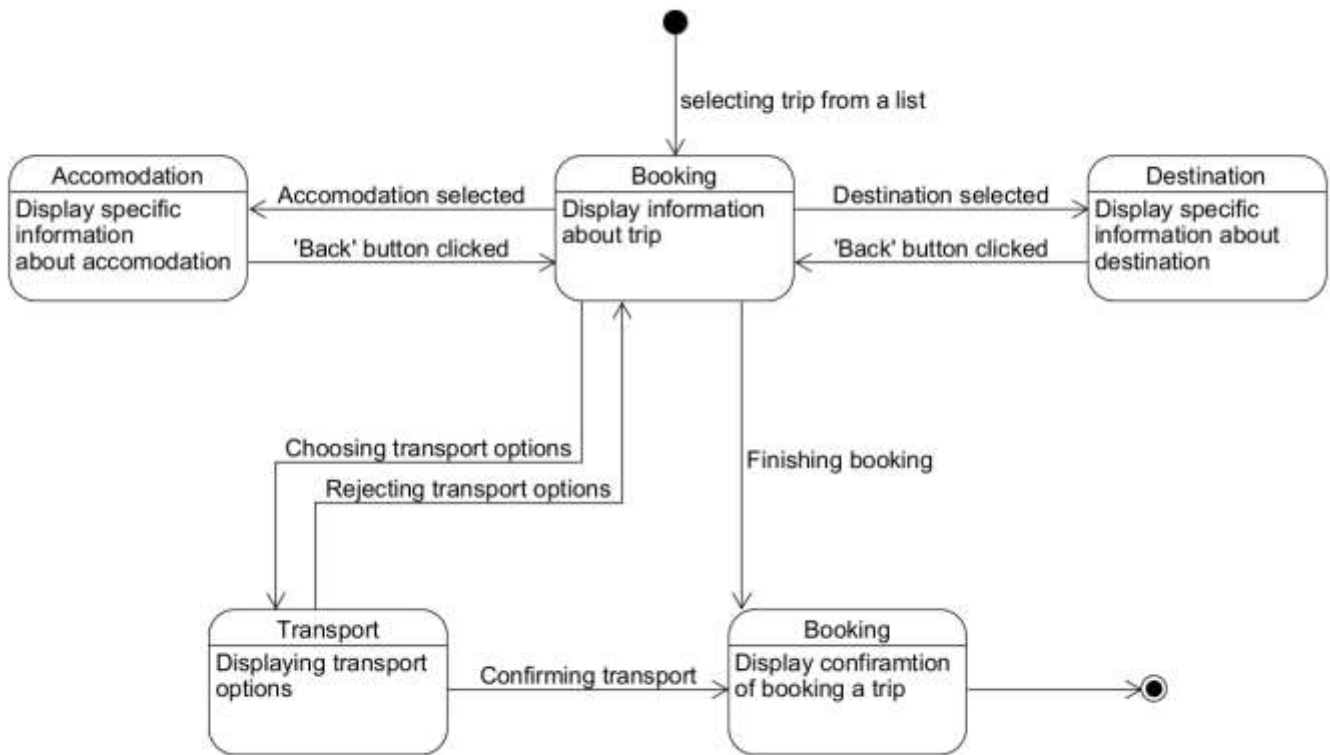
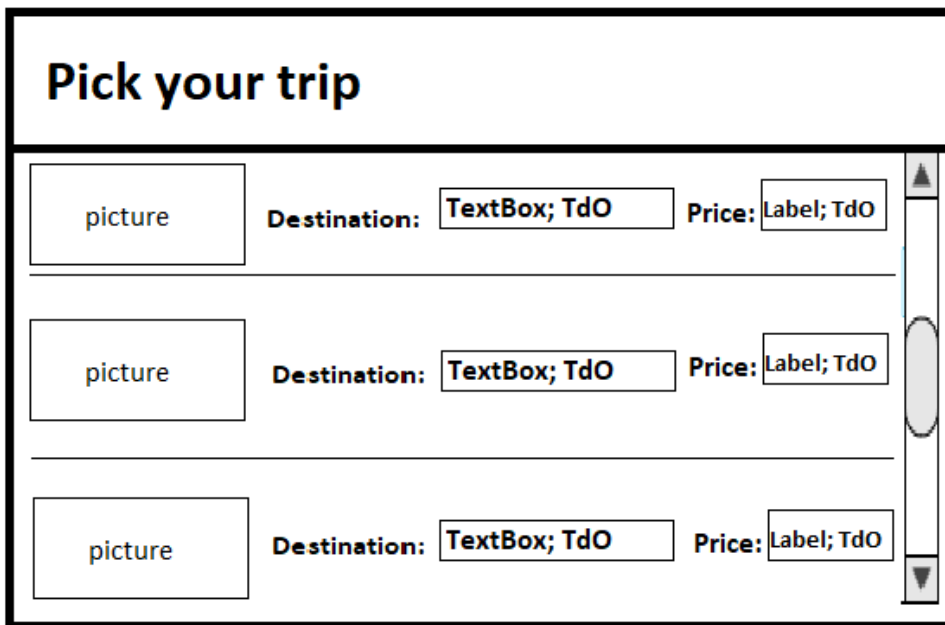**8. Interaction diagram (sequence) for use case.**

9. GUI project.

Customers clicks button showing all available trips

Travel system

show Trips

After that, list of all trips is shown. Customer picks trip which is most suitable for him.

Pick your trip

picture	Destination:	TextBox; TdO	Price:	Label; TdO
picture	Destination:	TextBox; TdO	Price:	Label; TdO
picture	Destination:	TextBox; TdO	Price:	Label; TdO

After choosing trip, details are being shown:

Details of your trip

Destination: TextBox; TdO

Description: TextBox

Dates: TextBox

Accomodation: TextBox

Price per person: TextBox

Guide: TextBox

PICTURE

Book with transport

Book

After clicking "Book with transport":

Transport to: TextBox; TdO

Departure at: TextBox; TdO **By:** TextBox; TdO **Price:** Label; TdO

Departure at: TextBox; TdO **By:** TextBox; TdO **Price:** Label; TdO

Departure at: TextBox; TdO **By:** TextBox; TdO **Price:** Label; TdO

Book

10. Discussion of the design decisions and the effects of dynamic analysis.

After dynamic analysis methods were added to the classes on diagrams:

Name of the class	Methods added	Justification
Booking	- getDetails() - displayConfirmation() - safeBooking(Trip) - updateBooking(Transport) - book(Trip)	Client needs to book a specific trip and update booking if he picks transport option. Confirmation of a booking will be shown after every booking. Additionally, If clients decides to book a trip, this information should be saved in the database
Trip	- getTrips() - getDetails(Trip)	Those methods will provide connection with a database and ensure retrieving necessary information about available trips or specific details of them.
Transport	- getTransportOptions(Date, Destination)	Method which will provide connection with the API of partner providing transport to the place of a trip
Guide	- getDetails()	Method used by class Trip
Accommodation	- getDetails()	Method used by class Trip
Destination	- getDetails()	Method used by class Trip

The extension of the classes will be realized by means of a connection to the database. This will ensure that the stored data is up to date. As the system will not process large amounts of data, it will not cause delays in its operation.