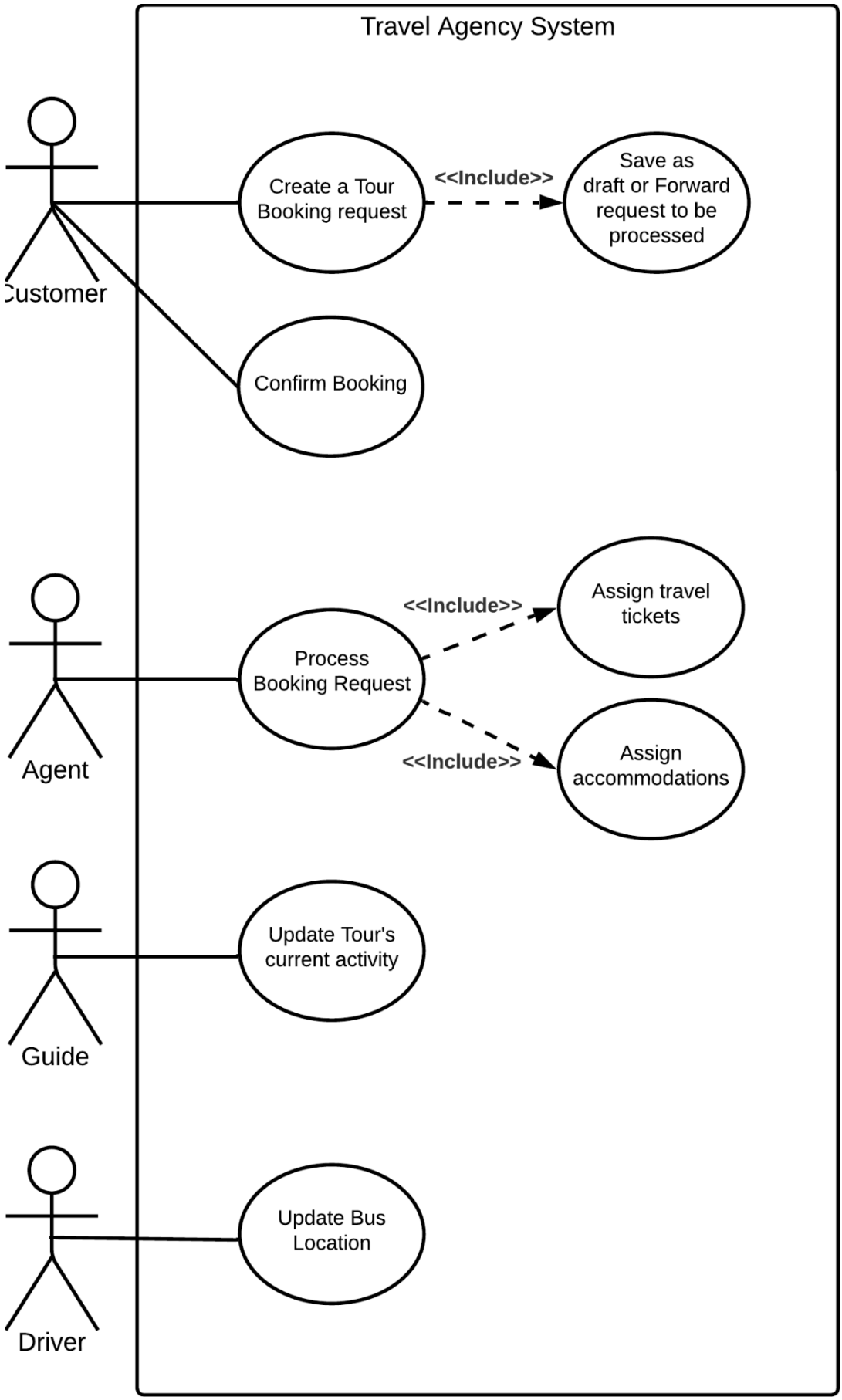
Travel Agency - Documentation

Youssef Toukabri - s24024

User Requirements

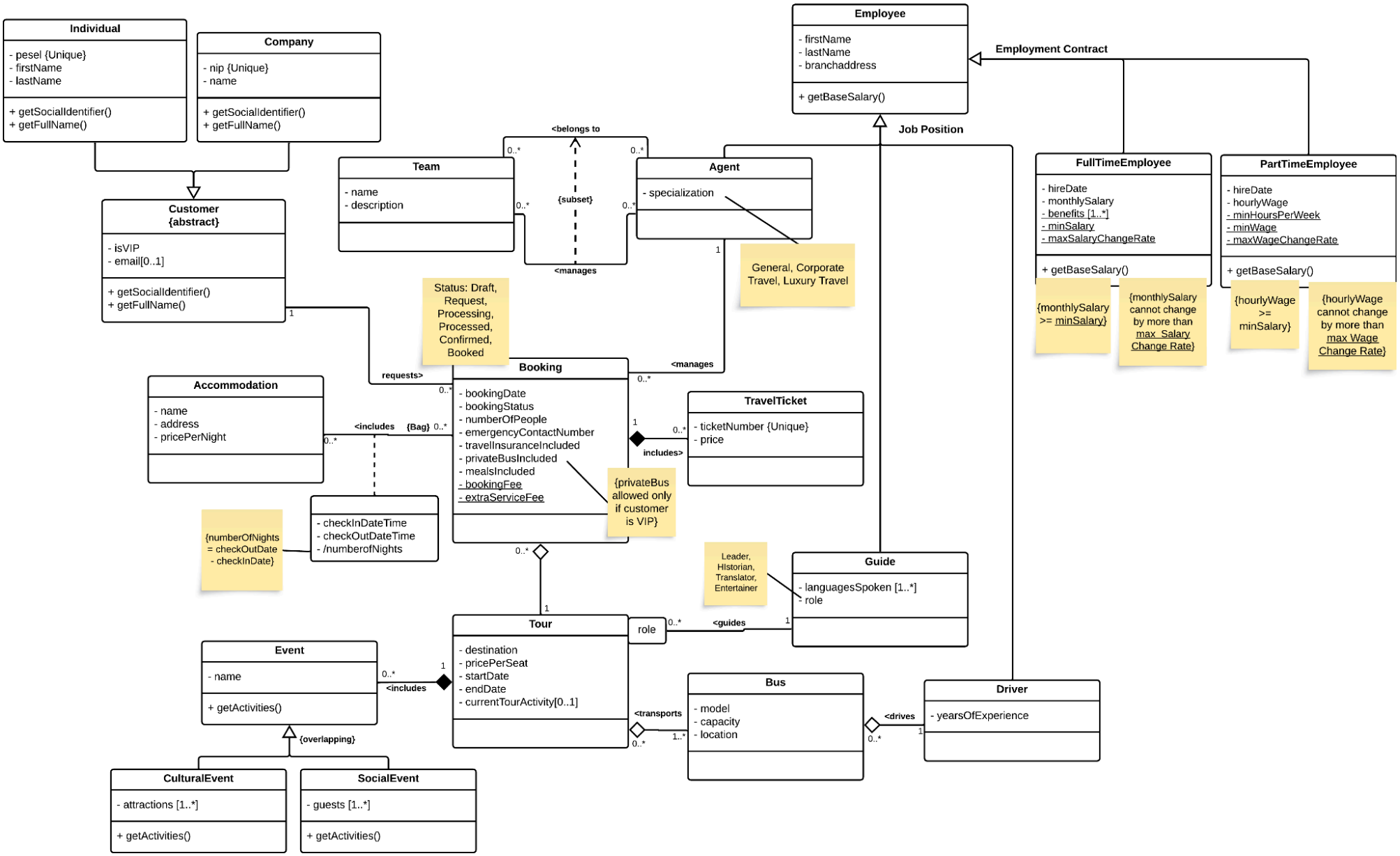
- Tour
  - System should provide a list of tour offers
  - System must provide the full details of each tour offer which include its different events, and guides.
  - System must allow many guides to guide the tour however it also must allow at most one guide from each role
- Customer
  - Customers must be allowed to browse and look through the tour offers.
  - Customers must be able to create a booking request for a desired tour offer.
  - Customers must be able to provide the number of people this request is for and choose to include extra services.
  - Only VIP Customers must be allowed to request private bus service.
  - The system must allow the customer to either save the request as a draft for later or forward it for processing by an agent.
  - The system must provide a way for the customer to access the saved request that are still relevant
  - The system must allow the customer to look through the details of their processed booking request and confirm them
- Employee
  - System must provide information about employees pay structure and employment type as well as their job position
- Agent
  - Agent must be able to see the forwarded booking requests and accept them
  - Agent must be able to update the booking by adding and assigning travel tickets and accommodations
- Guide
  - Guide must be able to update the system about the current status and activity of the tour they are leading
- Driver
  - Driver must be able to update the system about the current location of the Bus they are driving.
- Booking
  - System must allow the booking to go through different stages in order to be completed, which include the request stage from the customer, processing by the agent, and final confirmation and payment by the customer.

Use case diagram

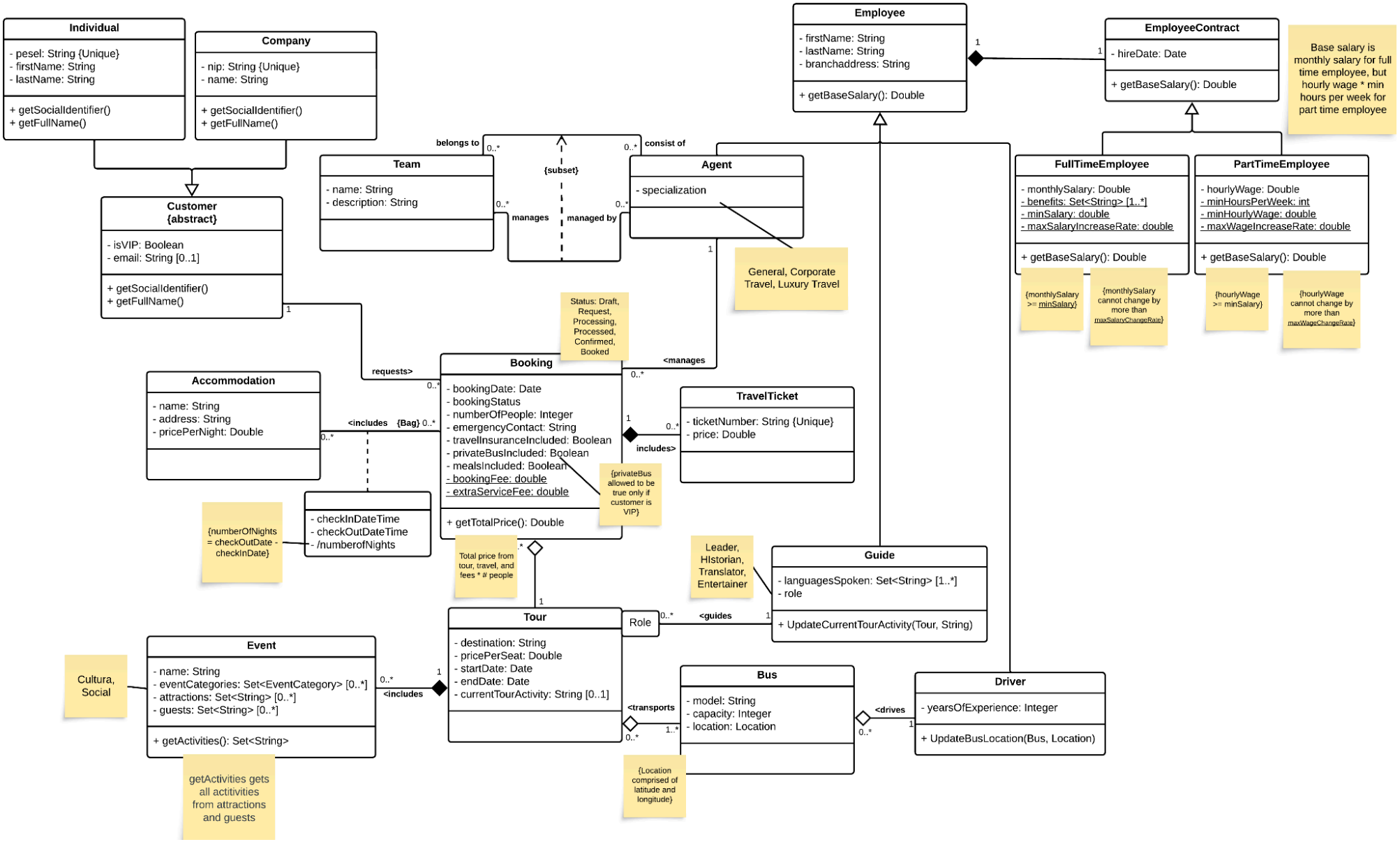


Class diagram

1. Analytical



2. Design


Individual
- pesel: String {Unique}
- firstName: String
- lastName: String
+ getSocialIdentifier()
+ getFullName()
Company
- nip: String {Unique}
- name: String
+ getSocialIdentifier()
+ getFullName()
Customer {abstract}
- isVIP: Boolean
- email: String [0..1]
+ getSocialIdentifier()
+ getFullName()
Employee
- firstName: String
- lastName: String
- branchaddress: String
+ getBaseSalary(): Double
EmployeeContract
- hireDate: Date
+ getBaseSalary(): Double
Base salary is monthly salary for full time employee, but hourly wage * min hours per week for part time employee
FullTimeEmployee
- monthlySalary: Double
- benefits: Set<String> [1..*]
- minSalary: double
- maxSalaryIncreaseRate: double
+ getBaseSalary(): Double
{monthlySalary >= minSalary}
{monthlySalary cannot change by more than maxSalaryChangeRate}
PartTimeEmployee
- hourlyWage: Double
- minHoursPerWeek: int
- minHourlyWage: double
- maxWageIncreaseRate: double
+ getBaseSalary(): Double
{hourlyWage >= minSalary}
{hourlyWage cannot change by more than maxWageChangeRate}
Team
- name: String
- description: String
belongs to
consist of
{subset}
manages
managed by
Agent
- specialization
General, Corporate Travel, Luxury Travel
requests>
Booking
- bookingDate: Date
- bookingStatus
- numberOfPeople: Integer
- emergencyContact: String
- travelInsuranceIncluded: Boolean
- privateBusIncluded: Boolean
- mealsIncluded: Boolean
- bookingFee: double
- extraServiceFee: double
+ getTotalPrice(): Double
Status: Draft, Request, Processing, Processed, Confirmed, Booked
<manages
{privateBus allowed to be true only if customer is VIP}
Accommodation
- name: String
- address: String
- pricePerNight: Double
<includes
{Bag}
- checkInDateTime
- checkOutDateTime
- /numberofNights
{numberOfNights = checkOutDate - checkInDate}
TravelTicket
- ticketNumber: String {Unique}
- price: Double
includes>
Total price from tour, travel, and fees * # people
Tour
- destination: String
- pricePerSeat: Double
- startDate: Date
- endDate: Date
- currentTourActivity: String [0..1]
Role
<guides
Leader, HIstorian, Translator, Entertainer
Guide
- languagesSpoken: Set<String> [1..*]
- role
+ UpdateCurrentTourActivity(Tour, String)
Event
- name: String
- eventCategories: Set<EventCategory> [0..*]
- attractions: Set<String> [0..*]
- guests: Set<String> [0..*]
+ getActivities(): Set<String>
<includes
Cultura, Social
getActivities gets all actitivities from attractions and guests
Bus
- model: String
- capacity: Integer
- location: Location
<transports
{Location comprised of latitude and longitude}
Driver
- yearsOfExperience: Integer
+ UpdateBusLocation(Bus, Location)
<drives

Selected use case - Create Tour Booking Request

1. Specification Scenario

Create a Tour Booking request - Specification Scenario

Actor: Customer

Purpose of context: Customer attempts to request a booking for tour offers.

Dependencies:
- None

Assumptions and Preconditions:
- None

Trigger: Customer selects "Request booking" button

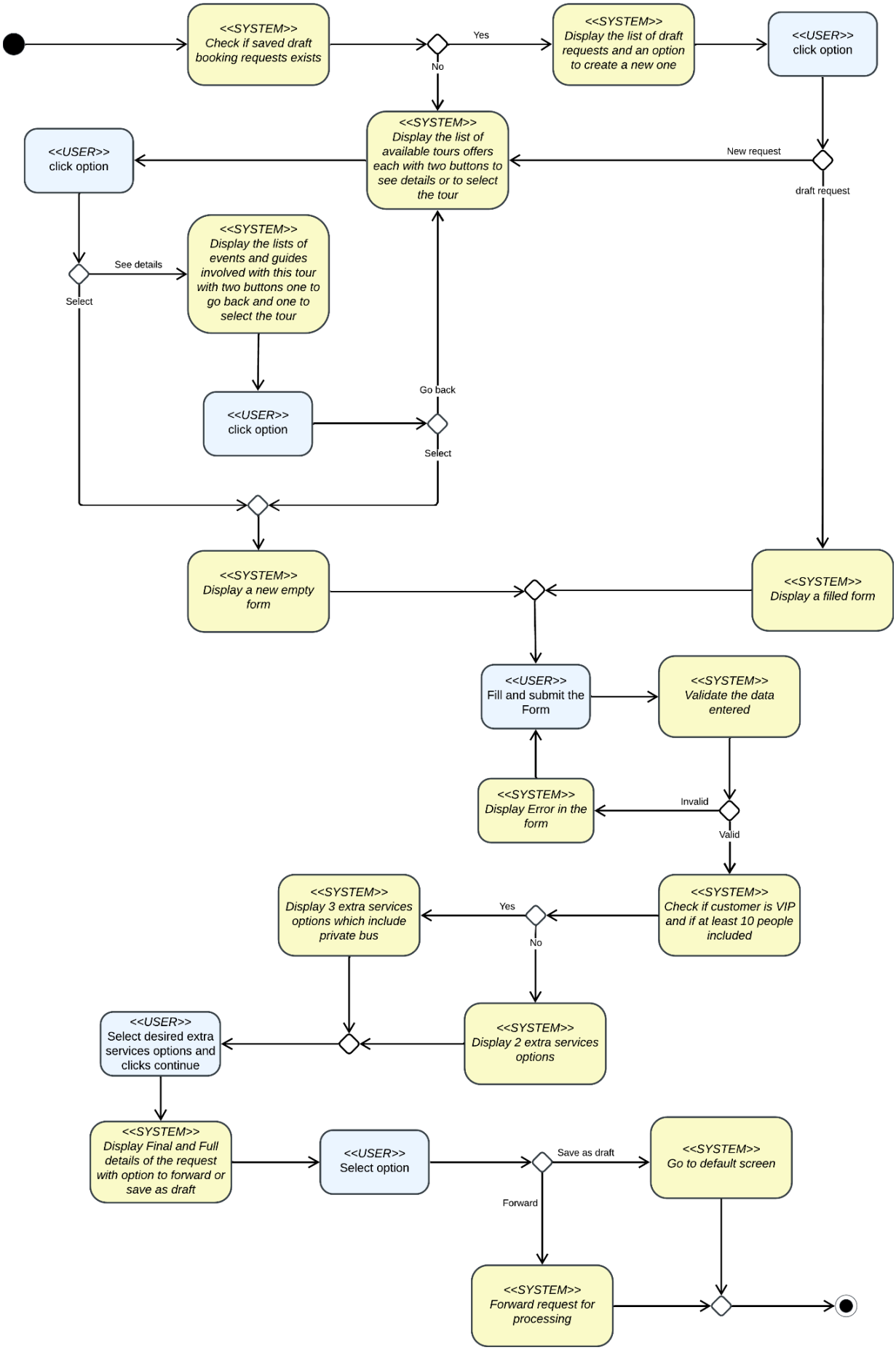
Basic flow of events:
1. System displays list of unexpired draft requests and an option to create a new one
2. Customer clicks on new request option
3. System displays the list of tour offers page, each with options to see tour details or selects it
4. Customer selects the desired tour offer
5. System displays empty booking details form
6. Customer fill form and clicks continue
7. System displays extra optional services page which include Travel Insurance, Meals, a private bus service option for VIP customers booking for at least 10 people.
8. Customer selects their desired extra services and clicks continue
9. System displays the full request details with options to forward for processing or save as a draft.
10. Customer selects their desired option to forward the request or save it as a draft

Alternative flow of events:
Use unexpired draft request:
2a1. Customer clicks on one of the saved draft requests
2a2. System displays filled booking details form along with the pre-selected tour
2a3. Go to point 6.

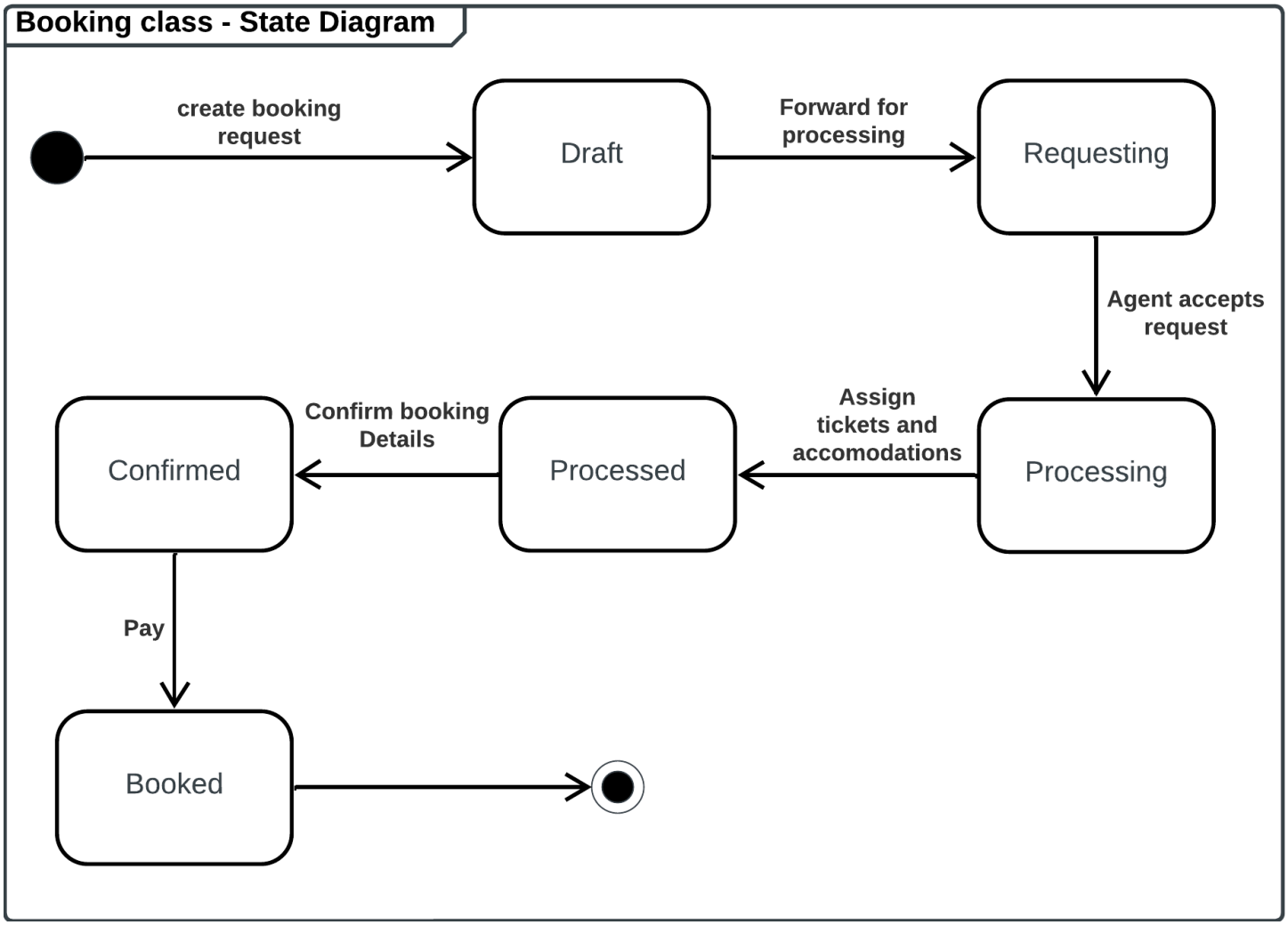
See tour details:
4a1. Customer clicks on see details of a tour
4a2. System displays tour details which onclude the list of events and guides involved, along with options to select the tour or to go back to tour offers list view
4a3. Customer selects the tour
4a4. Go to point 5.

Post-Conditions:
1. Booking request is created and saved in the system.

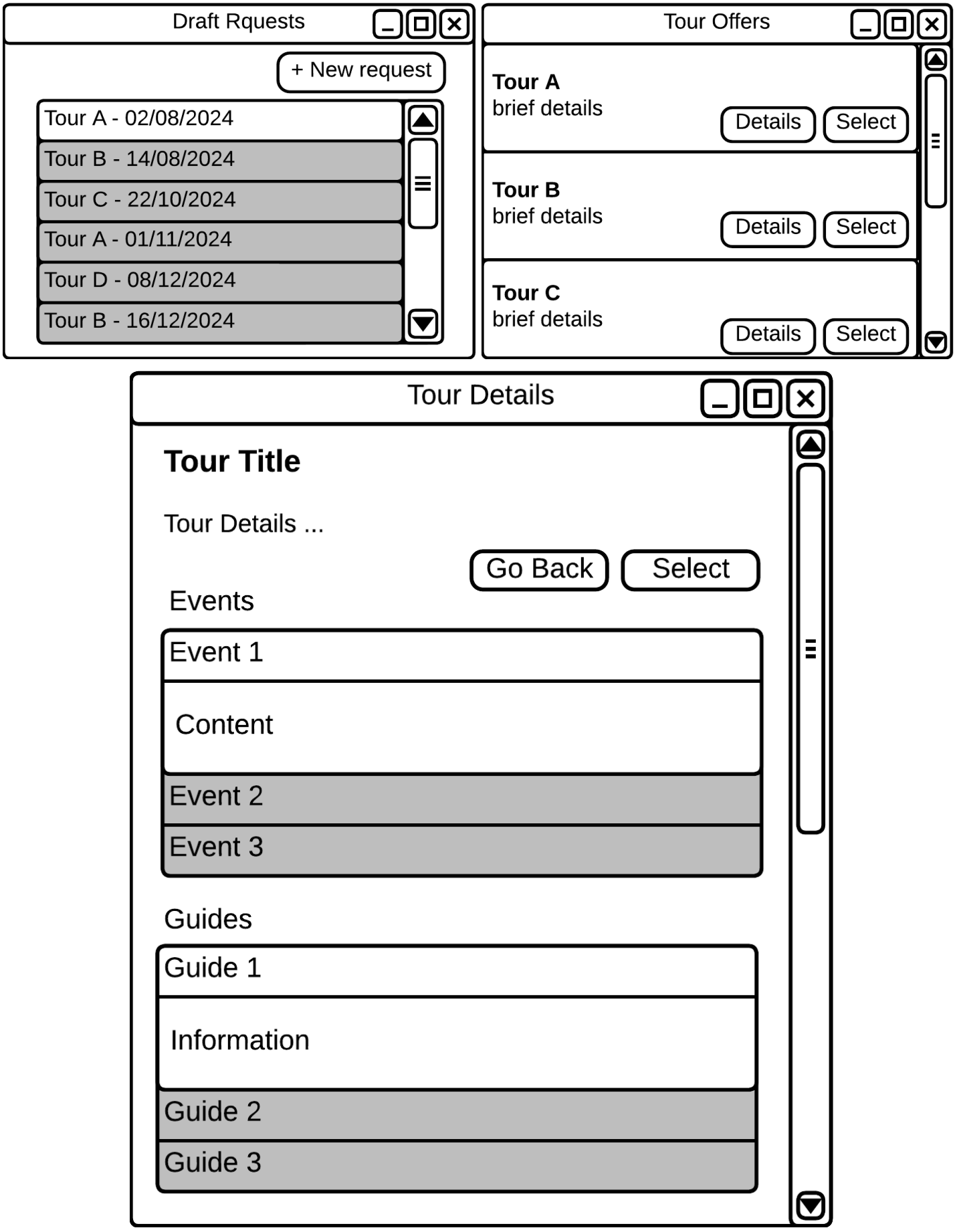
2. Activity diagram

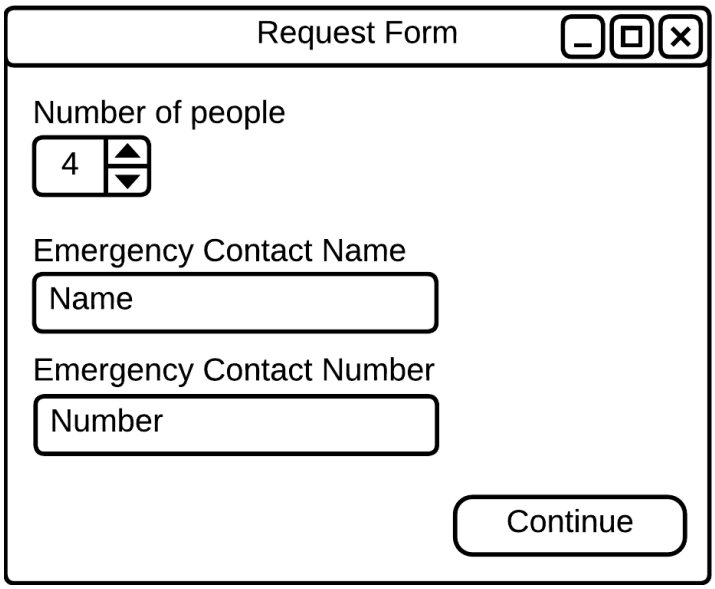


3. State diagram

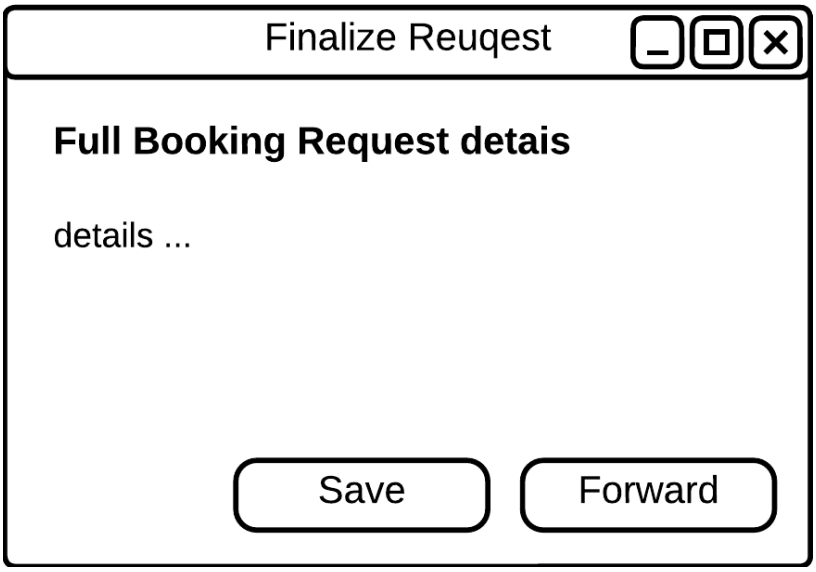


4. Design


Draft Rquests
+ New request
Tour A - 02/08/2024
Tour B - 14/08/2024
Tour C - 22/10/2024
Tour A - 01/11/2024
Tour D - 08/12/2024
Tour B - 16/12/2024
Tour Offers
Tour A
brief details
Details
Select
Tour B
brief details
Details
Select
Tour C
brief details
Details
Select
Tour Details
Tour Title
Tour Details ...
Go Back
Select
Events
Event 1
Content
Event 2
Event 3
Guides
Guide 1
Information
Guide 2
Guide 3


Request Form
Number of people
4
Emergency Contact Name
Name
Emergency Contact Number
Number
Continue


Request Services
Extra Services
Travel Insurance
Included Meals
Private Bus
Continue


Finalize Reuqest
Full Booking Request detais
details ...
Save
Forward

The discussion of design decisions and the effect of dynamic analysis

1. Inheritance and Polymorphism
    a. Customer: since customer can be either a company or an individual, an abstract inheritance is used with polymorphic methods to get enough general information about the customer through the system
    b. Event: an overlapping inheritance was used for the event since an event can be of multiple types at the same time. In the design phase this was combined into a single Event class.
    c. Employee: Multi aspect inheritance was used for the employee as the employee can have different pay structure and contract type aside from their particular position and job title. In the design however, a new EmployeeContract class was added with a 1 to 1 composition relationship with Employee for the implementation as multi aspect inheritance is not supported by default. The base salary of the employee can be accessed no matter their pay structure and contract type where for full time employee it is simply their current monthly salary, but for part time employees it is their hourly wage * the minimum hours per week * 4
2. Constraints
    a. Individual and Company customers use PESEL and NIP respectively as their unique social identifiers.
    b. Subset constraint was used between the Agent and their team, since agents not only can belong to teams but also can be managers of their teams.
    c. The Bag constraints was used for the association by attribute between the accommodation and booking since the same accommodation can be booked multiple times for different dates during the same tour as tourists can change locations and come back.
    d. Static and dynamic constraints were used for the employee to put restrict the minimum possible salary or hourly wage, as well as to put restrictions on the maximum rate the salary or hourly wage can change at a time
3. Attributes
    a. Bus location is a complex attribute to provide the longitude and latitude information of a bus
    b. Derived attribute was used to get the number of nights from the association by attribute between Accommodation and Booking, calculated as the days from the difference between end date and start date. This is used to calculate the total price in the booking
    c. Customer email and event’s current activity are optional attributes
    d. Full time employee benefits is represented as a multi-value class attribute

e. Guide's spoken languages are represented as multi-value attributes
f. Booking fee and Extra service fee are represented as class attributes used and shared by all booking instances, used to calculate the total price.

4. Methods

a. Customer class has two abstract methods that are overridden in Individual and Company classes to the full name and their social id
b. The full price of the booking is calculated through a method that combines the prices of travel tickets, accommodation bookings, Tour price, and booking fee and extra service fee, depending on the customer's choice, Times the number of people.
c. Methods were added to guide and driver classes to update the tour's current activity and bus location that they are managing respectively.