ICT Engineering

SEP 1Y A16

Project Report

Group 1

Andreea Buturca 253691

Karolina Beliharova 253810

Marek Lowy 253652

Martin Janosik 253934

Supervisors

Steffen Vissing Andersen

Birgitte von Fyren Balsløv

Date: 16. 12. 2016

**Table of contents**

[1 Abstract 5](#_Toc469331392)

[2 Introduction 6](#_Toc469331393)

[3 Analysis 7](#_Toc469331394)

[3.1 Requirements 7](#_Toc469331395)

[3.2 Use Case Diagram 9](#_Toc469331396)

[3.3 Use Case Description 11](#_Toc469331397)

[3.4 Activity Diagram 12](#_Toc469331398)

[4 Design 13](#_Toc469331399)

[4.1 Graphical User Interface (GUI) 13](#_Toc469331400)

[4.2 Database Design 14](#_Toc469331401)

[4.3 Model class diagram 15](#_Toc469331402)

[4.4 Class diagram 16](#_Toc469331403)

[4.5 Sequence diagram 17](#_Toc469331404)

[5 Implementation 18](#_Toc469331405)

[6 Testing 19](#_Toc469331406)

[6.1 Testing methods according to Use Case 19](#_Toc469331407)

[6.2 Test of GUI 19](#_Toc469331408)

[7 Results 20](#_Toc469331409)

[8 Conclusion 21](#_Toc469331410)

[9 References 22](#_Toc469331411)

[9.1 Books 22](#_Toc469331412)

[9.2 Additional resources 22](#_Toc469331413)

[10 Appendices 23](#_Toc469331414)

[10.1 Appendix 1 – Activity diagram 23](#_Toc469331415)

[10.2 Appendix 2 – Use Case Description 24](#_Toc469331416)

[10.3 Appendix 3 – Class diagram 30](#_Toc469331417)

[10.4 Appendix 4 – User Guide 30](#_Toc469331418)

**List of figures and tables**

[Figure 1: Priority Use Case Diagram 9](file:///C:\Users\Karolina\Documents\GitHub\SEP1\Documentation\ProjectReport.docx#_Toc469331976)

[Figure 2: Class Diagram 16](file:///C:\Users\Karolina\Documents\GitHub\SEP1\Documentation\ProjectReport.docx#_Toc469331977)

[Figure 3: Class Diagram 30](file:///C:\Users\Karolina\Documents\GitHub\SEP1\Documentation\ProjectReport.docx#_Toc469331978)

[Table 1: Manage buses 24](file:///C:\Users\Karolina\Documents\GitHub\SEP1\Documentation\ProjectReport.docx#_Toc469332009)

[Table 2: Register a new chauffeur 25](file:///C:\Users\Karolina\Documents\GitHub\SEP1\Documentation\ProjectReport.docx#_Toc469332010)

[Table 3: Create tour 26](file:///C:\Users\Karolina\Documents\GitHub\SEP1\Documentation\ProjectReport.docx#_Toc469332011)

[Table 4: Make reservation 27](file:///C:\Users\Karolina\Documents\GitHub\SEP1\Documentation\ProjectReport.docx#_Toc469332012)

[Table 5: Find Reservation/Tour 28](file:///C:\Users\Karolina\Documents\GitHub\SEP1\Documentation\ProjectReport.docx#_Toc469332013)

[Table 6: Edit Reservation/Tour 29](file:///C:\Users\Karolina\Documents\GitHub\SEP1\Documentation\ProjectReport.docx#_Toc469332014)

# Abstract

*VIA Bus is a company located in Horsens, Denmark with Trip Driver as the manager. At VIA Bus you can either rent a bus with a chauffeur – driving to a destination of your choice, or travel by bus to one of the predefined locations, i.e. a few European countries and some Danish sites and events.*

*VIA Bus is need of some kind of system to keep track of the tours, chauffeurs, bus routes and customers, and therefore the manager Trip Driver decided to contact us to create the system for him. System based on Java must meet the requirements of the customer.*

# Introduction

The project presented to us is based on an interview with VIA Bus Company having wishes for their booking system. VIA Bus is a company that offers different types of tours for customers. They organize predefined tours or make a new trip by renting a bus with a chauffeur.

The program must meet their requirements, run on Java, which means it has to be used on various platforms. Simple and usability tested GUI have to be created to make sure the employees will have no problems using it. It must be single user system, running on one single front desk.

The VIA Bus company wants a user system that will be easy and clear for using. We need to create simple, but powerful tool that will handle every aspect which the employees of the company might encounter. The employee wants to have an easy way to reserve tour. The employee also wants to store information like customer and chauffeur data in a database. The employees should be able to find available chauffeurs and busses and assign them new trips according to their wishes. The system will automatically calculate prices for each reservation according to travel type, extra services etc. All data must be saved in the computer using database which will be easy to back up in case of losing data.

# Analysis

## Requirements

* The employee should be able to search through the list of destinations for the two kinds of tours: predefined trips and travels, and redesigned trips by renting a bus with a chauffeur.
* The employee should be able to see all available seats in bus for specific tour.
* The employee should be able to pick available seat and reserve it for customer including destination, date and time.
* The employee should be able to see details about reservation as price, extra services, discount, and amount of passengers.
* For private customers, the employee should be able to register the name, address, the passenger’s birthday, and when it is permitted the email address for the newsletter
* For companies, the employee should be able to register name, address, phone number, email from the person who is making the reservation and the company’s name.
* The employee should be able to find data about customer in system as name, address, date of birth, email, phone number.
* The employee should be able to search through the list of customers to find if the customer is frequent or not.
* The employee should be able to reward and give discounts to the frequent customers and company.
* The employee should be able to add any additional notes to reservation.
* The employee should be able to make a new reservation, which was not predefined, rent bus and chauffeur according destination, date and number of passengers.
* Register employees by: name, address, employee ID, calendar trips, email, phone, preferences for trip.
* The employee should be able to store info about the type of contract: full-time or vicar, for new employee.
* The employee should be able to search through the list of busses one specific type for the service “bus and-chauffeur”. (party bus, classic bus, luxury bus, mini bus)
* The employee should be able to find an available chauffeur after the suitable bus was found.
* The employee should be able to search through the list of chauffeurs: full time or vicars.
* The employee should be able to hire a bus-and-chauffeur for a day, or couple of hours by searching through the destination list.
* The employee should be able to cancel vicar contract any time.
* For a bus and a chauffeur service, the employee should be able to reserve additional services such as breakfast, lunch in the bus, reserve a restaurant dinner, and guide party, suitable for the chosen destination.
* The employee should be able to register the type of a customers by their profile: company or private person, including at least name, address and phone number.
* The employee should be able to specify wishes for full-time employees for trips.
* When the employee creates new tour, should be able to choose chauffeur according to his wishes for the trip
* For the “bus and chauffeur” services the chauffeurs must be able to register time of departure, arrival destinations, and arrival back to the terminal.
* The employee should be able to find a reservation in the system search by customer or company, date and destination.
* The employee should be able to delete reservation.
* The employee should be able to modify reservations.
* The employee should be able to save reservation if there are some changes.
* The employee should be able to set the prices for each reservation according to the travel type, extra services.
* The employee stores all the data in database.
* The system has to be implemented in Java.

## C:\Users\Karolina\Desktop\usecase.PNGUse Case Diagram

Priority Use Case Diagram 3.1

Figure 1: Priority Use Case Diagram

**Create tour:** Employee is asked to enter information about new tour: date, time, departure, destination, distance, he has to choose bus from the bus list and chauffeur from the chauffeur list, fill in extra services, price, bus stop and time for break. If it is private tour, employee has to fill data about customer, otherwise tour is added to the list.

**Register a new chauffeur:** Employee is asked to enter information about new chauffeur: name, address, phone number, email address, employee ID (5-digit number), and type of contract: full-time or vicar. If it is full-time, employee specify wishes for trip and the chauffeur is added to the list.

**Manage buses:** Employee is asked to enter information about new bus: type, registration plate and number of seats and the bus is added to the list. Employee is able to remove bus from the list.

**Make reservation:** Employee can make a reservation.

**Find Reservation/Tour:** Employee can search reservation by customer data: name / company name, address, email and phone number. Employee is able to remove reservation. Employee can find tour search by tour information: destination, departure, date. Employee is able to remove tour from the list.

**Edit reservation/tour:** When employee fined reservation/tour, he can edit some information.

## Use Case Description

## Activity Diagram

# Design

## Graphical User Interface (GUI)

## Database Design

## Model class diagram

This basic model shows us the structure of the system. “Bus” class is used to create new vehicle in the system and information about it and through this class we get information about bus type from classes “LuxuryBus, PartyBus, ClassicBus, MiniBus”. “Bus list” class stores all busses and information about them. “Person” class is extended by classes “Passenger, Customer, and Chauffeur” and is used to store data about a person. “ChauffeurList” is used to hold a list of chauffeurs and it can split them according to criteria. “CustomerList” is used to hold a list of customers and split them according to criteria and the same is for “PassengerList” with passengers. Every object of class “Reservation” stores information about each reservation. “ReservationList” is used to hold a list of reservation and it can filter them by various criteria. Information in class “Destination” is about place and stops and it is stored to “DestinationList”. In class “Trip” is created new tour and all data is saved to “TripList”. “Main” class contains main functions of our system. “DataHandler” is used for connecting database with the “Main” class and help us to have fixed connection with database when we do some changes like add bus/reservation/chauffeur.

## Class diagram

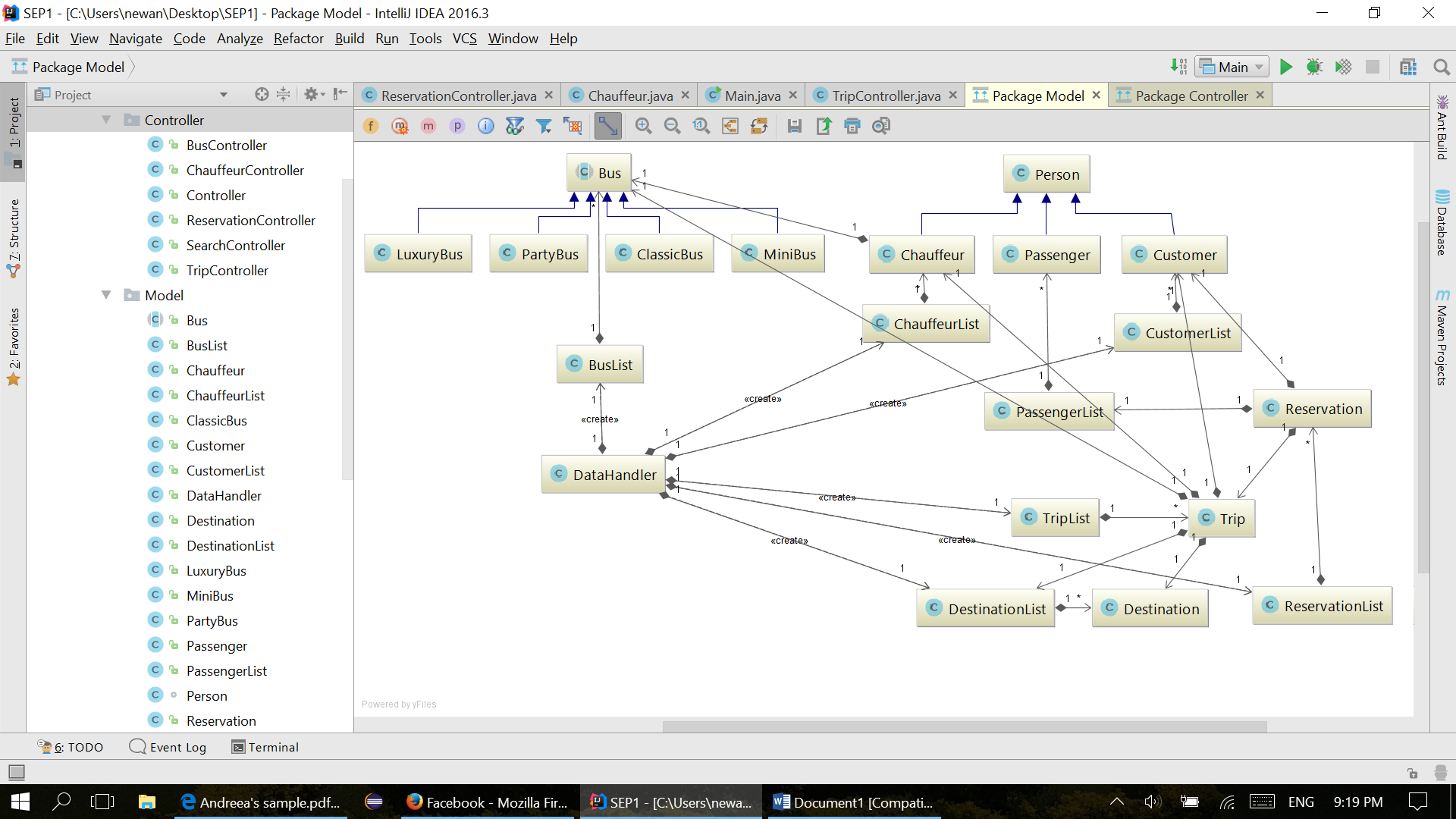


Figure 2: Class Diagram

## Sequence diagram

# Implementation

# Testing

## Testing methods according to Use Case

## Test of GUI

# Results

# Conclusion

# References

## Books

## Additional resources

Presentations from classes

JAVA 1 and UML (SDJ1)

# Appendices

## Appendix 1 – Activity diagram

## Appendix 2 – Use Case Description

**Manage buses:**

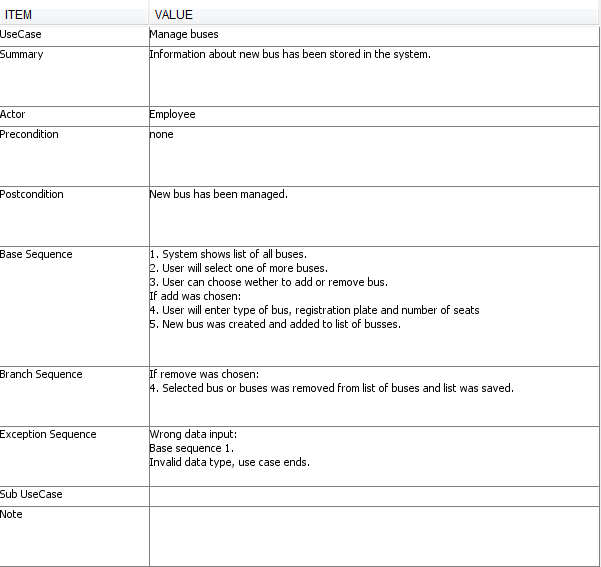


Table 1: Manage buses

**Register a new chauffeur:**

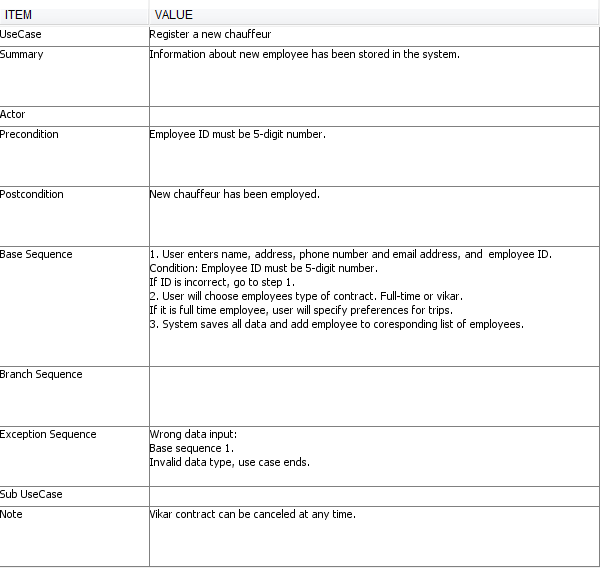
****

Table 2: Register a new chauffeur

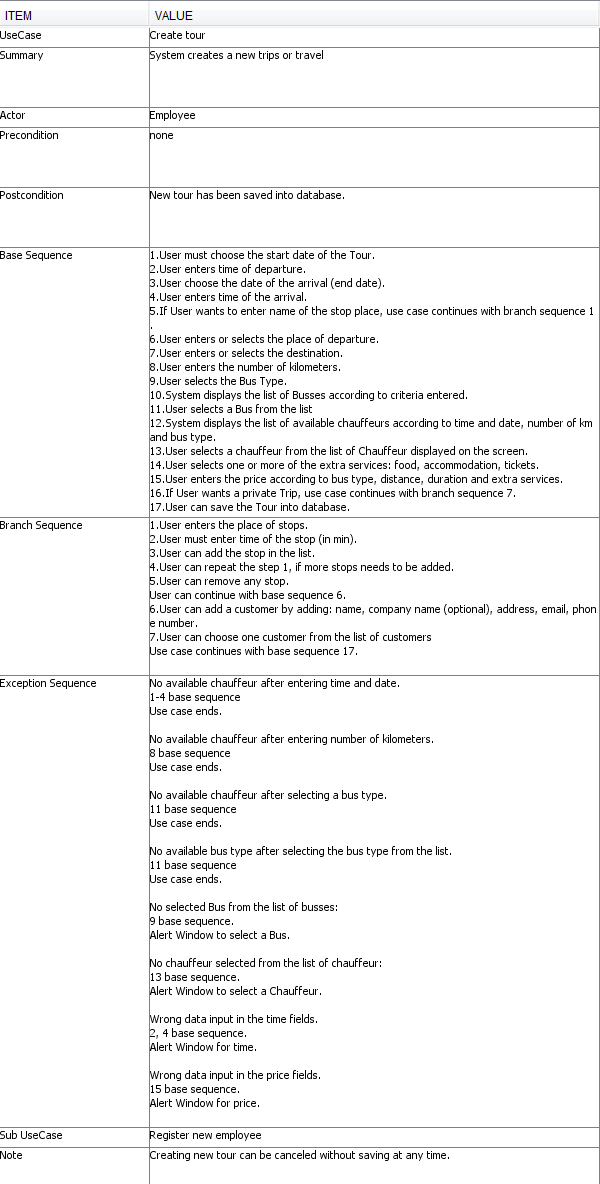
**Create a tour:**

Table 3: Create tour

**Make reservation:**

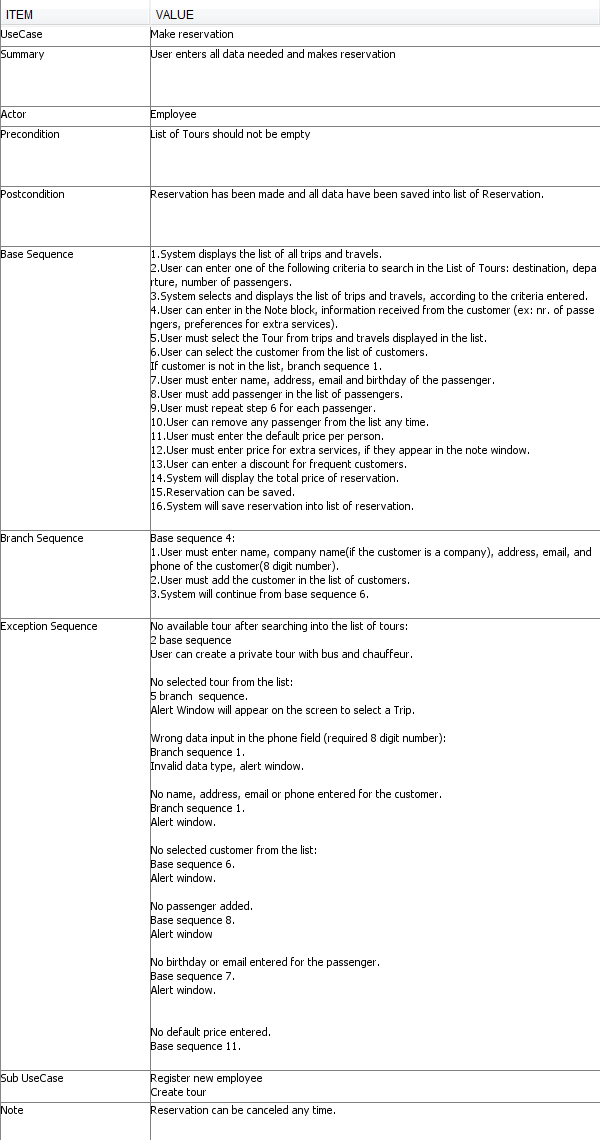
****

Table 4: Make reservation

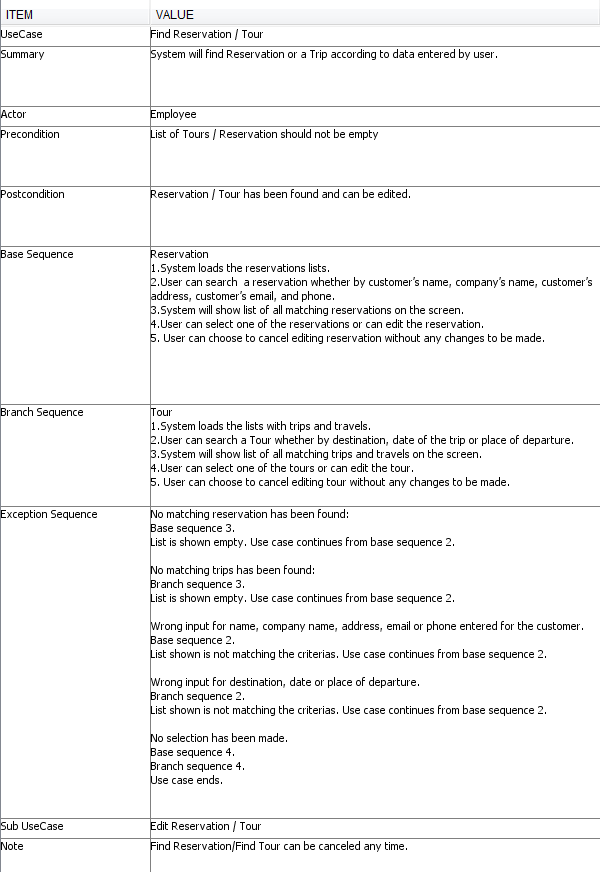
**Find Reservation/Tour:**

Table 5: Find Reservation/Tour

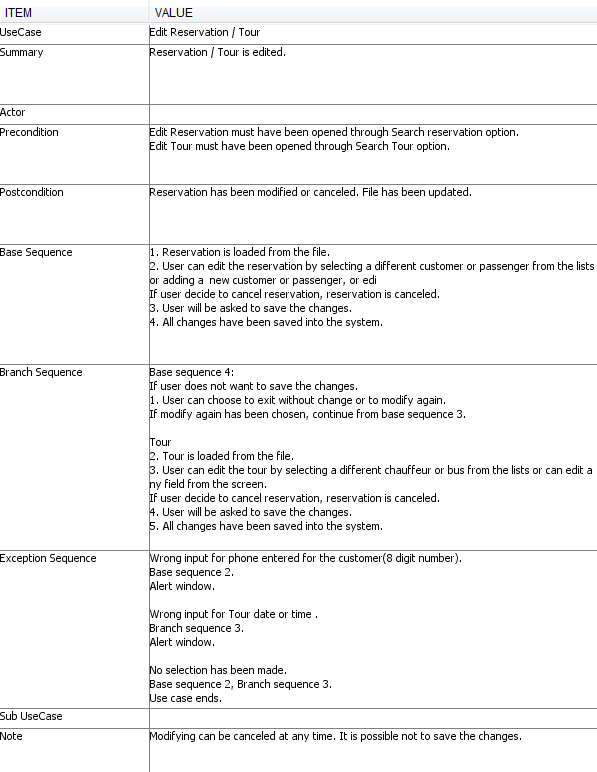
**Edit Reservation/Tour:**

Table 6: Edit Reservation/Tour

## Appendix 3 – Class Diagram

Figure 3: Class Diagram

## Appendix 4 – User Guide