Software Design Description Template

Author: Jordi van Velzen & Timo van den Hazel

Version: 1 (31/10/17)

1 Introduction 1

1.1 Overall Description 1

1.2 Purpose of this document 1

1.3 Definitions, acronyms, and abbreviations 1

2 Architectural Overview 1

3 Detailed Design Description 2

3.1 Deployment Diagram 2

3.1.1 Design Decisions related to deployment 2

3.2 Design Sub-System A 2

3.2.1 Design Class Diagram 2

3.2.2 Sequence Diagrams 2

3.2.3 Activity and State Diagrams 2

3.2.4 Design decisions made for the sub-system 2

3.3 Design Sub-System B (and so on) 3

3.4 Database Design 3

3.4.1 Design decisions related to the database 3

# Introduction

## Overall Description

*Rent-it* is a rent-a-car company and needs to have a system developed for that purpose.

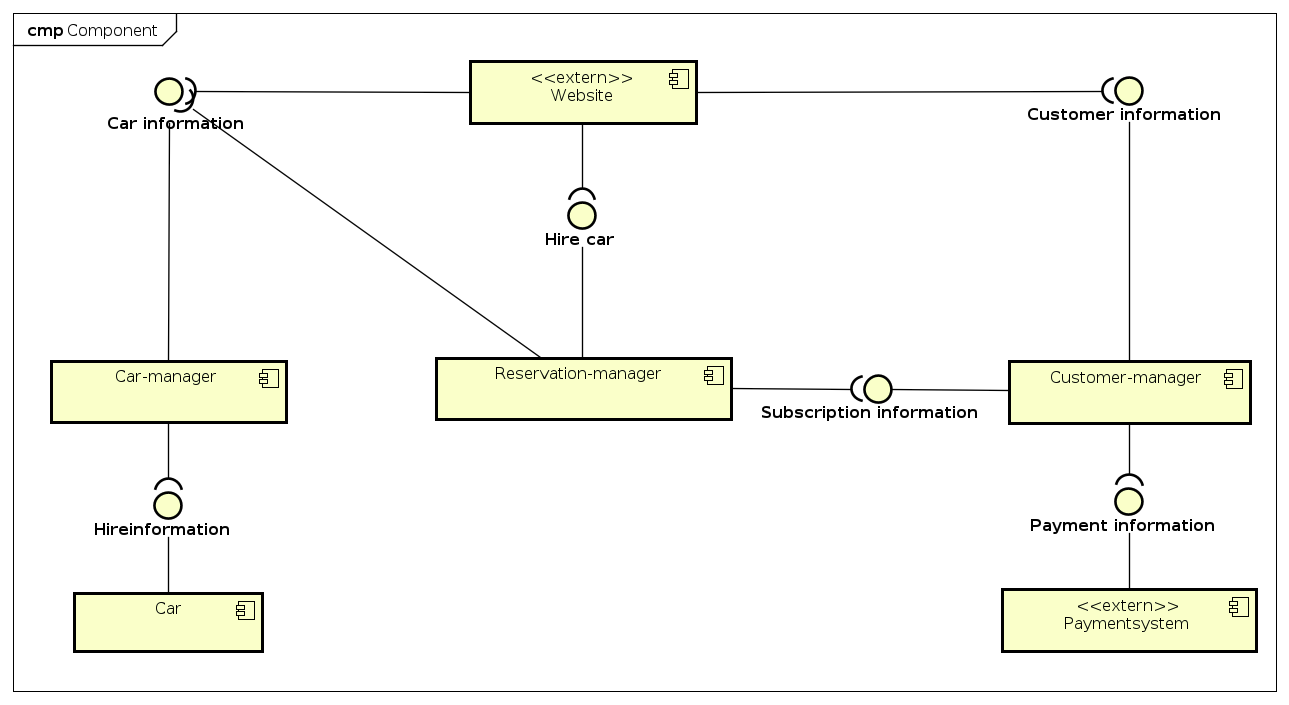
## Purpose of this document

This document describes the software architecture and implementations to show how the system is made. This document will also help developers implement the necessary functions to make the system function as a whole.

## Definitions, acronyms, and abbreviations

|  |  |
| --- | --- |
| Term | Description |
|  |  |

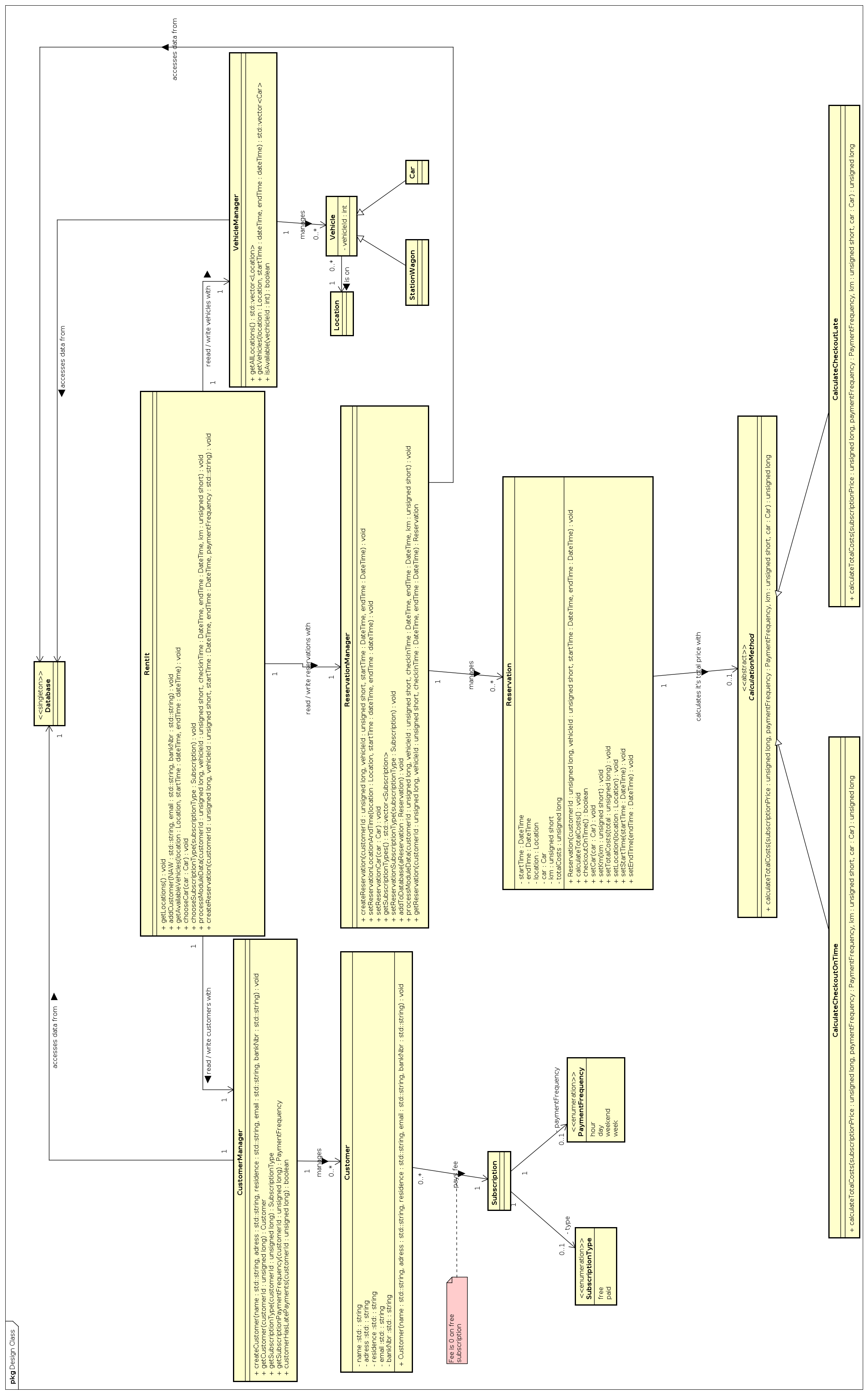
# Architectural Overview



Customers interact with the website (Rent-it), which will talk to the designated manager. The system is made of three different managers that interact with each other. In example: For a customer to make a reservation, Reservation-manager will need to interact with Customer-manager and Car-manager.

# Detailed Design Description Rent-it

## Design Class Diagram

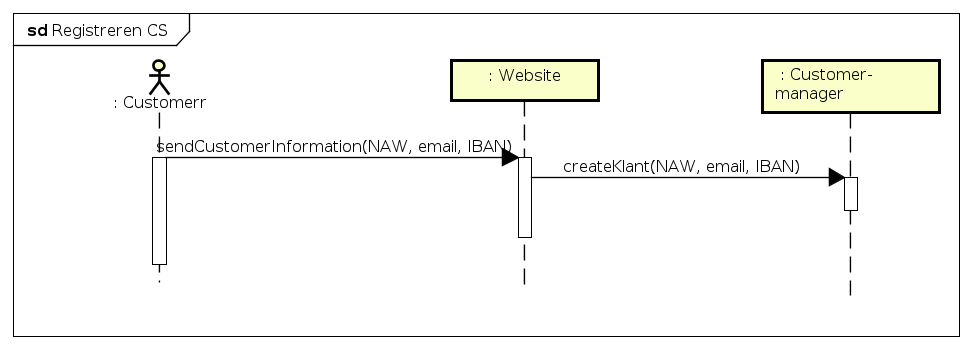


TODO: geen idee wat hierhover te zeggen

## 3.2 Sequence Diagrams

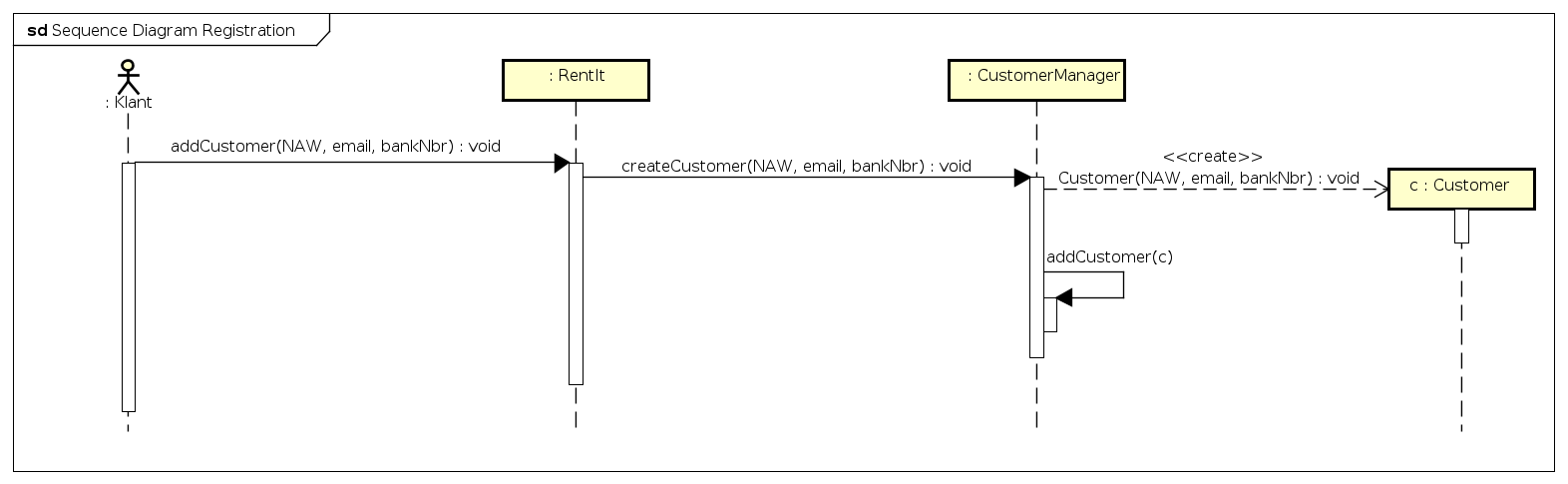
### 3.2.1 Registreren

Component Sequence:



Customer interacts with Rent-it through the website, which in turn will make Customer-manager process the data in order to register an account.

Sequence:



The more detailed sequence diagram shows a new Customer will be created and added to the database.

### 3.2.2 Berekenen kosten

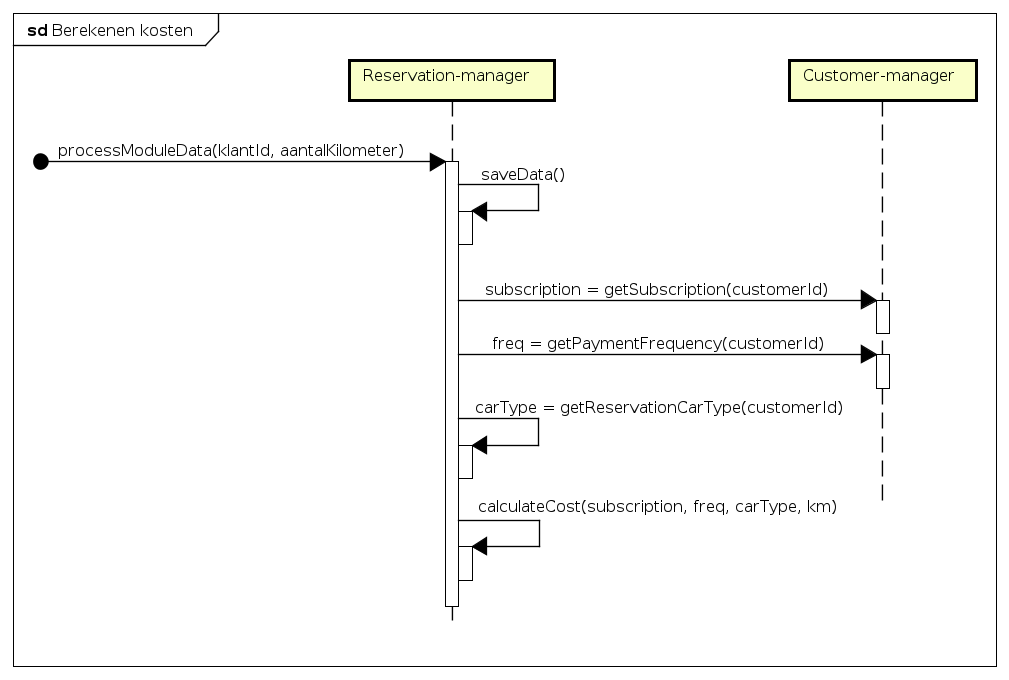
This use case can be started by 3 different scenarios:

- Customer used a car and checks out with customerpass (See 3.2.3)

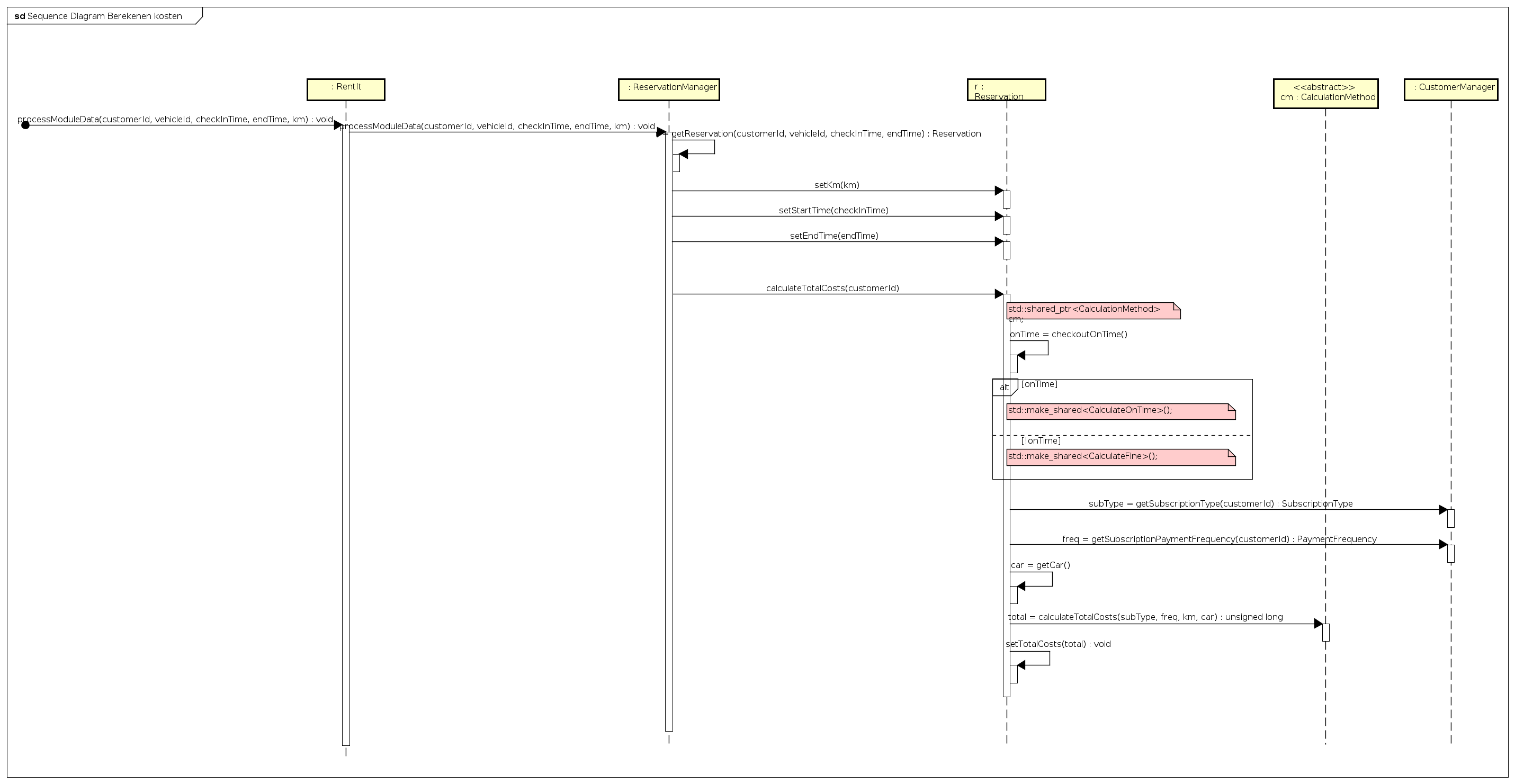
- Customer never checked in and the Reservation end-time has finished

- Customer checked in but has not checked out after the reservation end-time has finished plus a (to be determined) time

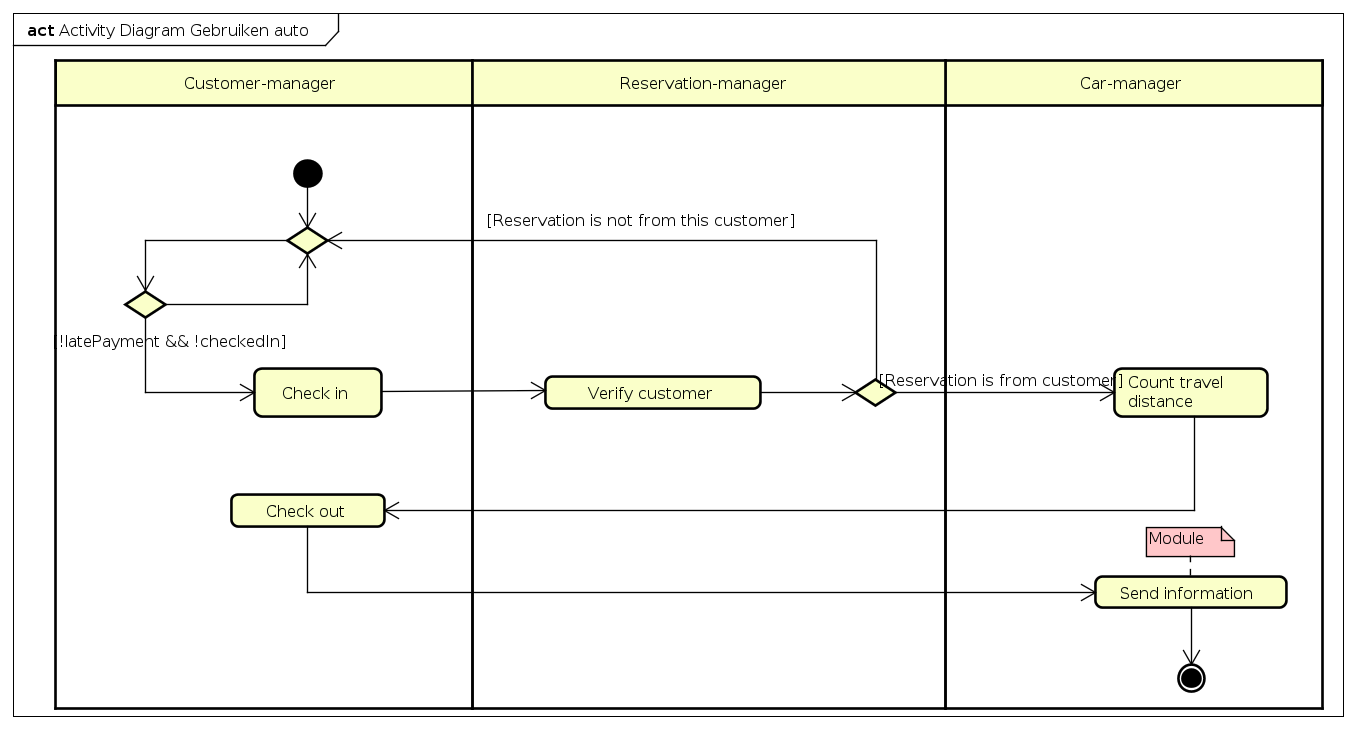
Component Sequence:

This diagram shows how Reservation-manager calculates the cost and uses Customer-manager to do so.

Sequence:

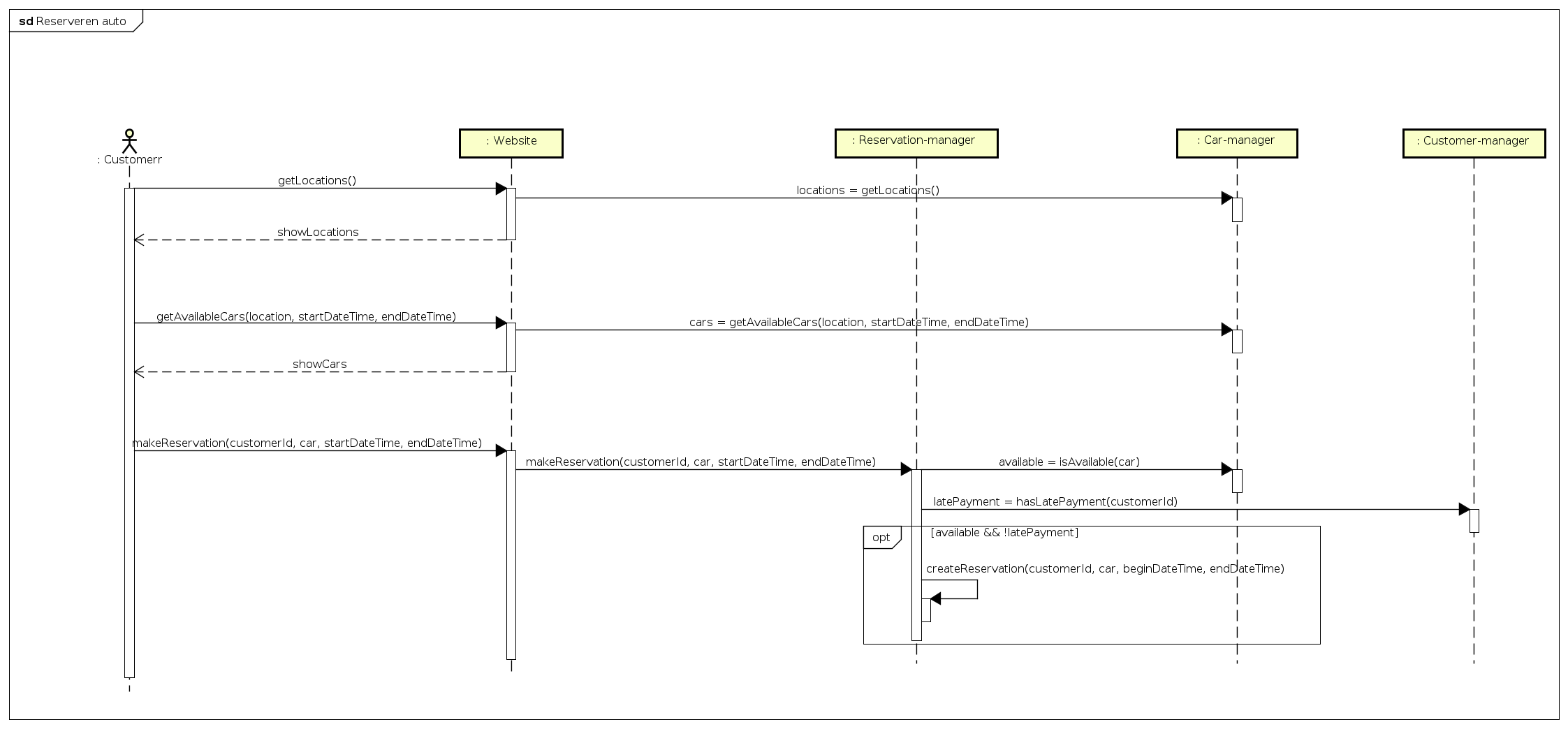
Based upon the scenario from which the use case has been started, a different calculation method will be used, as shown in this sequence diagram.

### 3.2.3 Use car



### 3.2.4 Car reservation

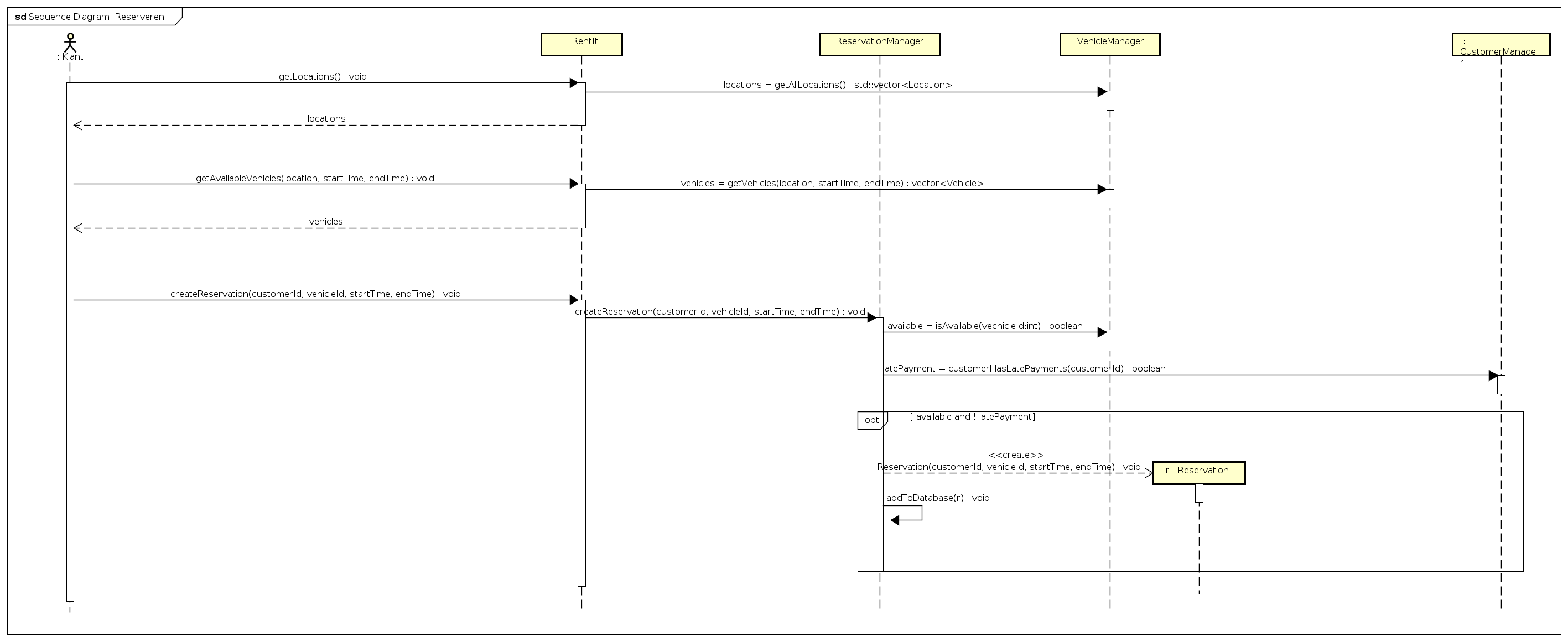
Component Sequence:



This component diagram shows the interaction between customer and the system, in which the customer will make a reservation.

First, the customer will choose a location, start-time and end-time for the reservation, after which the system will show the available cars. After a car is selected, if the customer has no late payments – the reservation will be made.

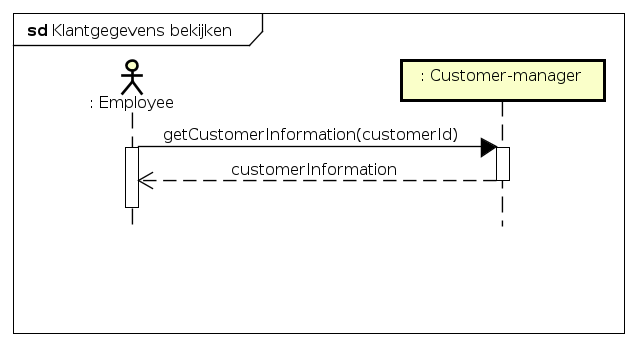
Sequence:



In addition to the component sequence diagram, this diagram shows a new Reservation being made and is added to the database.

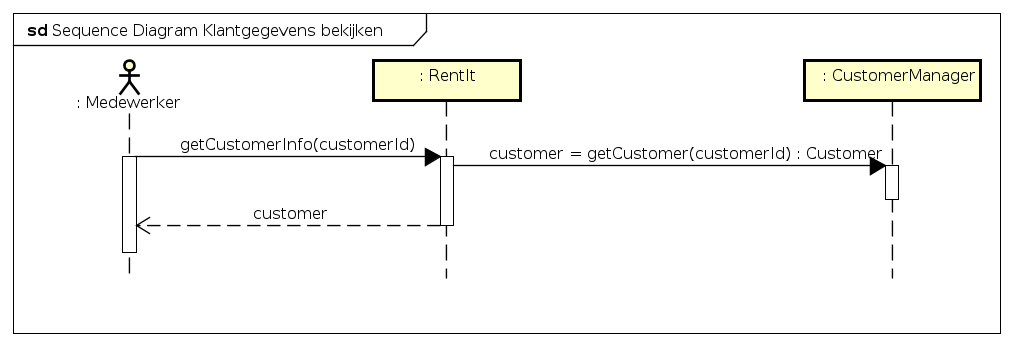
### 3.2.5 View customer information

Component sequence:



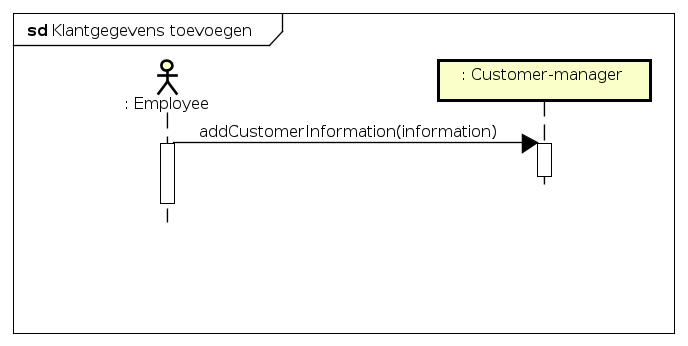
This diagram displays how an employee can view customerinformation.

Sequence:



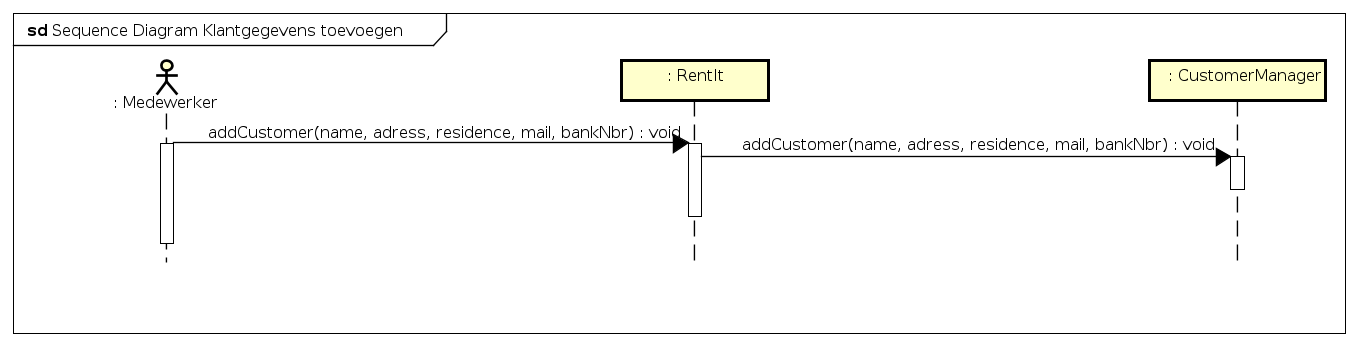
### 3.2.6 Add customer information

Component sequence:



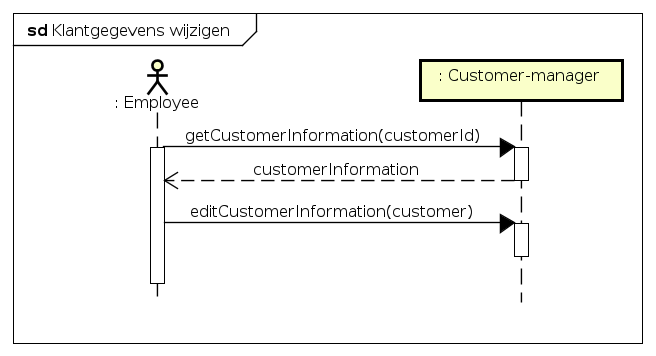
This diagram displays how an employee can add customer information

Sequence:



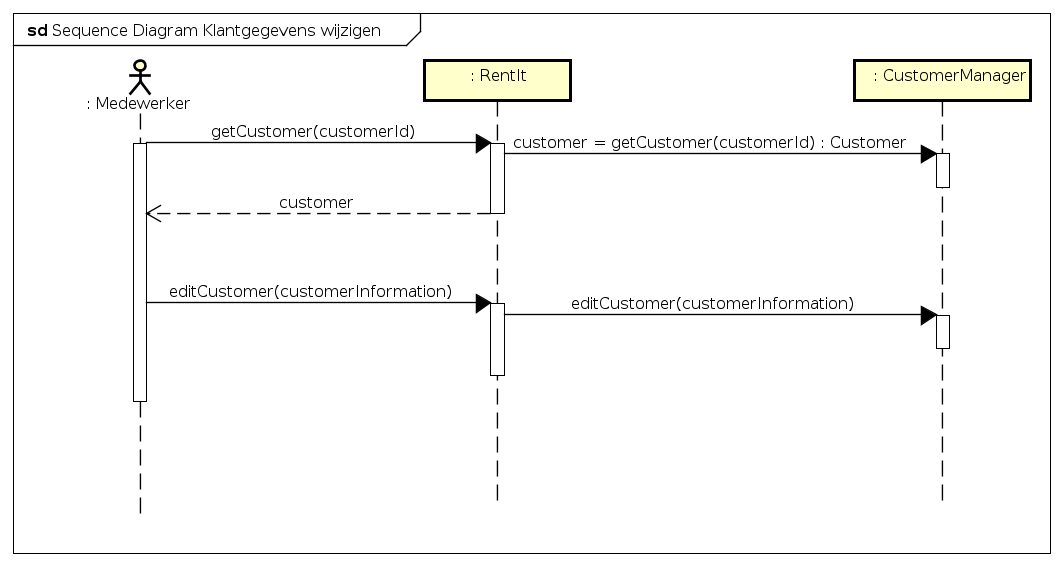
### 3.2.7 Edit customer information

Component sequence:



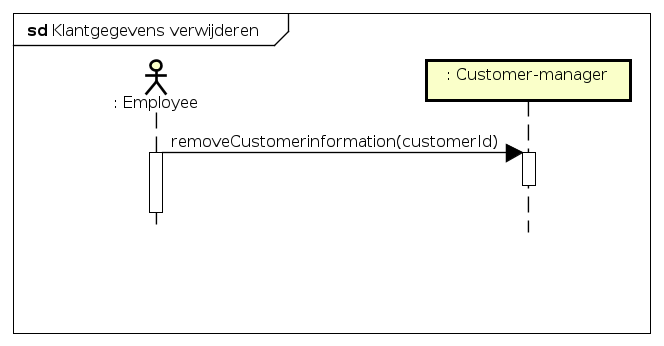
This diagram shows how an employee can edit customer information.

Sequence:



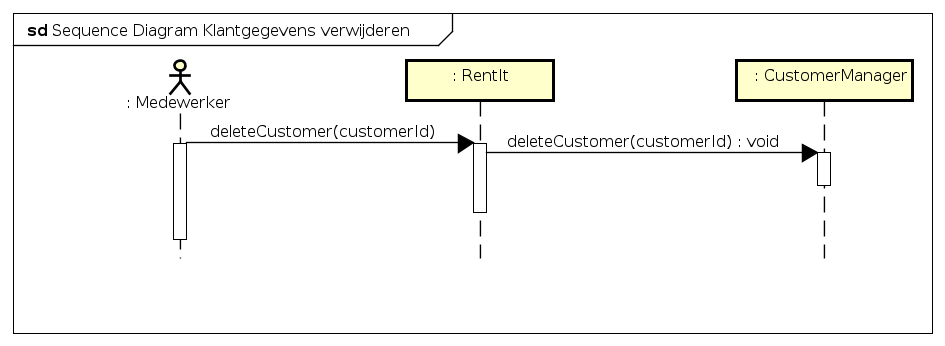
### 3.2.8 Delete customer

Component sequence:



This diagram shows how an employee can remove a customer.

Sequence:



## 3.3 Database Design

<. If your system uses relational databases, make sure you provide a physical datamodel here.>

### Design decisions related to the database

<Describe all design decisions made along the database. This could include the choice of the database management system, the use of certain triggers or stored procedures, special indexes and so on.>