Software requirements specification – Team 01

Inhaltsverzeichnis

Introduction 2

Purpose 2

Stakeholders 3

Definitions 3

System overview 3

References 3

Overall description 4

Use cases 4

Actor characteristics 4

Company/Logisticians: 4

Drivers: 4

Customers: 4

EPK 5

Objects 6

Specific requirements 6

User Interfaces 6

Hardware Interfaces 6

Software Interfaces 6

Communication Interfaces 6

Functional requirements 6

User Class 1 – The Logistician 6

User Class 2 – The Driver 6

User Class 3 – The customer 6

Non-functional requirements (external, performance, etc.) 6

Design constraints 6

Software System attributes 6

# Introduction

## Purpose

The goal of this software is to provide an application to simplify the process from ordering to delivery. In the current situation everything is done by hand from each employee in the process line, leading to an excessive amount of paper work. With an automated system the company wants to improve the efficiency of their logistics.

This Software Requirements Specification document serves as a guideline for the project and gives a detailed overview of the functionalities of the software. Its aim is to capture the needs of the company *RCP* and establish a common understanding of the functional and non-functional requirements of the project.



## Stakeholders

The stakeholders are Manuel Leuenberger of the company *RCP* and its employees including the logisticians and drivers.

## Definitions

**Logistician**: employee who handles incoming orders, prepares tours and assigns a driver for each tour.

**Driver**: carries out the tours, which are assigned to him, and delivers the products to the customer.

**Customer**: company to which the products are delivered to and which is able to accept or reject the delivery.

**Tour**: is a composition of different orders assembled by a logistician.

**Accepted delivery**: the order is successfully delivery and marked as accepted by the driver.

**Rejected delivery:** the order could not be successfully delivery due to a mistake in the delivery or the unavailability of the customer. It is then put back into the unassigned orders.

## System overview

The system should:

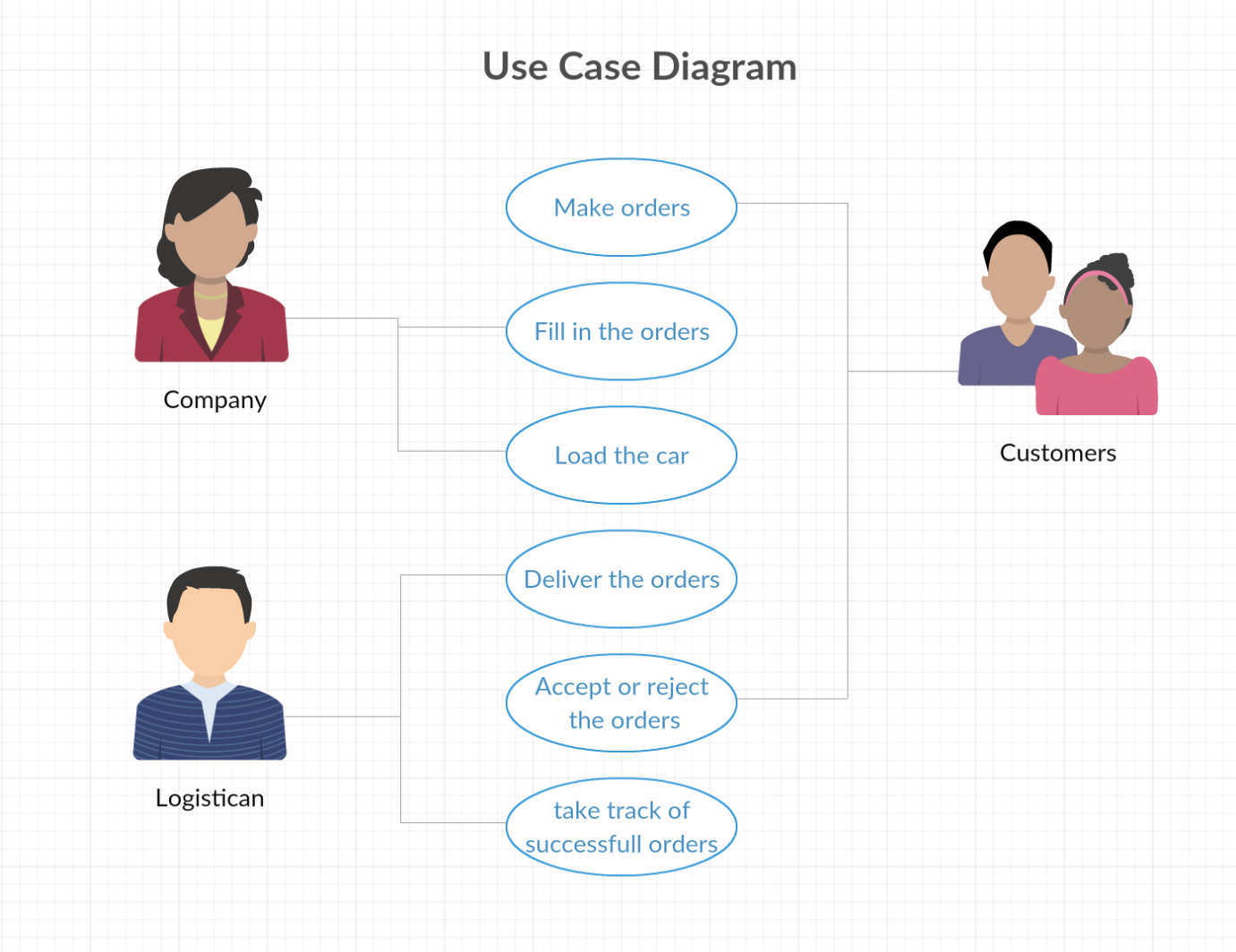
* Save the logisticians in the database, they have the access to add, change and look up things
* Save the drivers in the database, they can just look up things and add the state of the order
* Save the customers with their addresses in the database
* Save the orders from the client in a database
* Prepare tours with the different addresses of the clients
* Show the tours for the logisticians on a mobile device
* Let the drivers access the orders for changing the state of the order (delivered, rejected, not there)

## References

„here come what literature we use“

# Overall description

## Use cases



**Name:** Log into the system.

**Description:** A driver or logistician, which is already registered visits the website and logs in with his credentials.

**Preconditions:** User has access to the Internet, has an account on the website, knows his e-mail address and password

**Event Flow:** 1. User types in e-mail address

2. User types in password

3. User clicks login button.

4. System checks whether the credentials are correct

5. User is led to the driver or logistician homepage, depending on which he has access to. In case either the e-mail address or the password is incorrect, the user is requested to verify.

**Alternative Scenario:** 1. User types in e-mail address

2. User types in password

3. User clicks login button.

4. System detects invalid input of either the e-mail or password

5. User is requested to verify and type in e-mail and password again

**Name:** Log out of the system.

**Description:** A driver or logistician, who is logged in, logs out of the system.

**Precondition:** User has access to the Internet, has an account on the website, knows his e-mail address and password, is logged in

**Event Flow:** 1. User sees Log out button on the top right of the website

2. User clicks Log out button and is taken back to the Login page

**Name:** Overview orders

**Description:** A logistician wants to have an overview of all the unassigned and accepted orders.

**Precondition:** Logistician has access to the Internet, is logged in. At least one order exists.

**Event Flow:** 1. Logistician sees ‘Aufträge’ button on the homepage

2. Logistician clicks ‘Aufträge’ button and is taken to the list of orders

**Name:** Make a new order with a new customer

**Description:** A logistician wants to create a new order with a new customer in the system

**Precondition:** Logistician has access to the Internet, is logged in.

**Event Flow:** 1. Logistician sees ‘Neuer Auftrag erstellen’ button on the homepage

2. Logistician clicks ‘Neuer Auftrag erstellen’ button

3. Logistician clicks ‘Neuer Kunde erstellen’ button

4. Logistician fills in the form with the details of the new customer

5. Logistician clicks ‘Kunde speichern’ button and sees list of the details of the created customer

6. Logistician clicks ‘Diesem Kunden einen Auftrag zuordnen’ button

7. Logistician clicks on drop down list and chooses product and types in the amount of products

8. Logistician clicks ‘Bestellung Speichern’ button

**Postcondition:** The order is added to the database.

**Name:** Make a new order with an existing customer

**Description:** A logistician wants to create a new order with an existing customer in the system

**Precondition:** Logistician has access to the Internet, is logged in. At least one customer exists.

**Event Flow:** 1. Logistician sees ‘Neuer Auftrag erstellen’ button on the homepage

2. Logistician clicks ‘Neuer Auftrag erstellen’ button

3. Logistician clicks ‘Suche nach Kunden im System’ button

4. Logistician sees list of customer and selects the customer by clicking the ‘Weiter’ button next to the customer

5. Logistician clicks on drop down list and chooses product and types in the amount of products

6. Logistician clicks ‘Bestellung Speichern’ button

**Postcondition:** The order is added to the database.

**Name:** Delete an order

**Description:** A logistician wants to delete an existing order.

**Precondition:** At least one order exists.

**Event Flow:** 1. Logistician sees ‘Auftrag löschen’ button next to the order on the page showJobs

2. Logistician clicks ‘Auftrag löschen’ button

3. Logistician clicks ‘Auftrag definitiv löschen!’ button

4. Logistician sees confirmation that order is deleted

**Postcondition:** The order is deleted from the database.

**Name:** Create a new tour

**Description:** A logistician wants to create a new tour.

**Precondition:** User has access to the Internet, is logged in. There is at least one existing order and available vehicle and trailer.

**Event Flow:** 1. Logistician clicks ‘Touren’ button on homepage

2. Logistician clicks ‘Neue Tour erstellen’ on the bottom of the page

3. Logistician sees the new tour ID and clicks ‘Los!’

4. Logistician selects the desired driver for the tour by clicking ‘Weiter’

5. Logistician chooses the vehicle by clicking ‘Weiter’ next to it

6. Logistician chooses the trailer by clicking ‘Weiter’ next to it

7. Logistician adds orders to the tour by clicking the ‘+’ button

**Alternative Scenario:** 1. Logistician wants to add an order to the tour

2. Logistician sees notification that there isn’t enough space for the desired order

**Postcondition:** Tour is saved in the database.

**Name:** Edit tour

**Description:** A logistician wants to edit an existing tour.

**Precondition:** User has access to the Internet, is logged in. There is at least one existing tour.

**Event Flow:** 1. Logistician clicks on ‘Touren’ on the homepage.

2. Logistician clicks ‘Bearbeiten’ button next to the tour

3. Logistician can add orders to the tour by clicking ‘+’ button, or remove orders from the tour by clicking ‘-‘ button

**Alternative Scenario:** 1. Logistician wants to add an order to the tour

2. Logistician sees notification that there isn’t enough space for the desired order

**Postcondition:** Tour is adjusted in the database.

**Name:** Delete tour

**Description:** A logistician wants to delete an existing tour.

**Precondition:** User has access to the Internet, is logged in. There is at least one existing tour.

**Event Flow:** 1. Logistician clicks on ‘Touren’ on the homepage.

2. Logistician clicks ‘Bearbeiten’ button next to the tour

3. Logistician clicks ‘Tour löschen’ next to the tour

4. Logistician is asked to confirm and clicks ‘Tour definitiv löschen’

5. Logistician sees confirmation that tour is deleted

**Postcondition:** Tour is deleted from the database

**Name:** Show all users

**Description:** Logistician wants to have an overview over all logisticians and drivers.

**Precondition:** User has access to the Internet, is logged in. There is at least one existing user.

**Event Flow:** 1. Logistician clicks ‘Alle User’ on homepage

**Name:** Delete user

**Description:** Logistician wants to delete either another logistician or a driver

**Precondition:** User has access to the Internet, is logged in. There is at least one existing user.

**Event Flow:** 1. Logistician clicks ‘Alle User’ on homepage

2. Logistician clicks ‘User löschen’ next to the user the wants to delete

3. Logistician is asked to confirm and clicks ‘User definitiv löschen!’

4. Logistician sees confirmation that tour is deleted

**Postcondition:** User is deleted from the database.

**Name:** Create new user

**Description:** Logistician wants to create a new logistician or driver

**Precondition:** User has access to the Internet, is logged in.

**Event Flow:** 1. Logistician clicks on ‘Alle User’ button on homepage

2. Logistician clicks on ‘Neuer User erstellen’ at the bottom of the page

3. Logistician selects the role and fills in the form with the details of the new user

4. Logistician clicks ‘User speichern’

**Postcondition:** New user is saved in the database

**Name:** Show tours

**Description:** Driver wants to have an overview over his upcoming tours.

**Precondition:** User has access to the Internet, is logged in. There exists at least one tour that has been assigned to this driver.

**Event Flow:** 1. Driver clicks on ‘Touren anzeigen’ on the homepage

2. Driver selects which tour he wants to see by clicking ‘Weiter’ next to it

3. Driver sees the customer and the way of the tour on the map

**Name:** Complete tour

**Description:** Driver finished a tour and wants to mark it as completed

**Precondition:** User has access to the Internet, is logged in. There exists at least one tour that has been assigned to this driver.

**Event Flow:** 1. Driver clicks on ‘Touren anzeigen’ on the homepage

2. Driver selects which tour he wants to see by clicking ‘Weiter’ next to it

3. Driver clicks on ‘mehr’ next to the customer

4. Driver clicks on ‘Akzeptiert’ or ‘Abgelehnt’ depending on whether he could successfully deliver the products.

5. Driver clicks on ‘Tour abschliessen’

6. Driver is asked to confirm and clicks ‘Tour definitiv abschliessen!’

**Alternative Scenario:** 1. Driver clicks on ‘Tour abschliessen’

2. Driver is notified that not all deliveries are marked as accepted or rejected and is asked to mark each delivery before completing the tour

**Postcondition:** Rejected deliveries are shown again in the unassigned orders, so the logistician can assign them to another tour.

## Actor characteristics

### Logisticians:

The company that orders the program is taking the orders of the customers. They are making the loading lists and filling the trucks with the ordered goods.

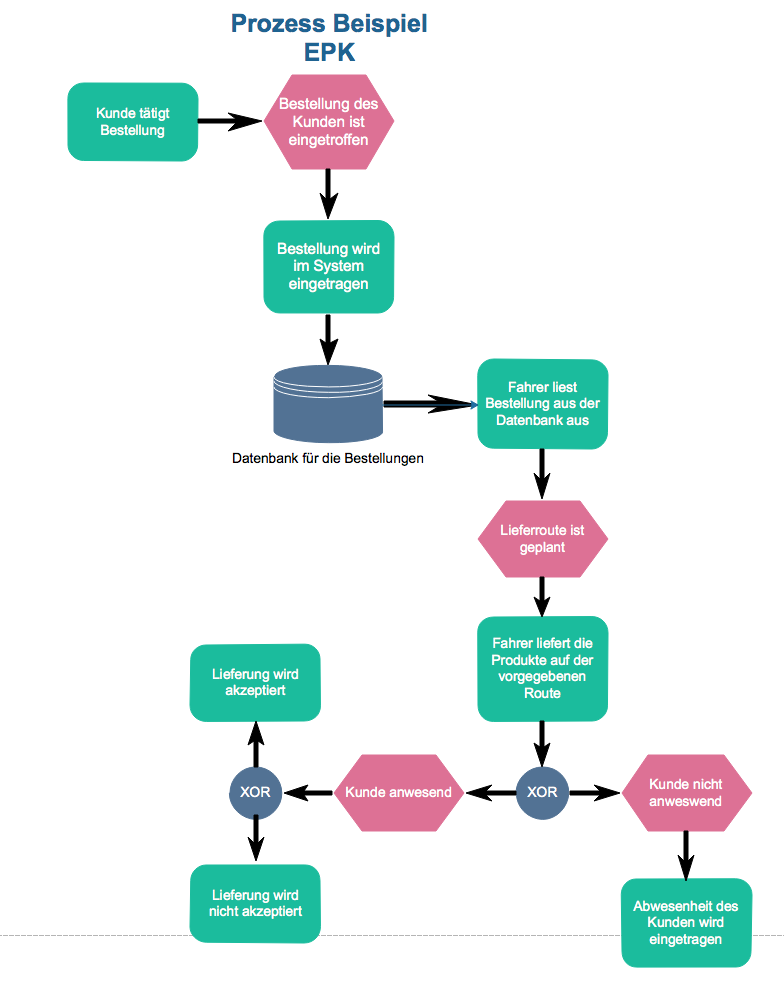
### Drivers:

They are taking the delivery lists and drive the ordered goods to the customers place. They have to take track of the accepted and rejected deliveries.

### Customers:

They order the goods from the company. They get the goods from the logistician and can accept or reject them when they are there.

## EPK



# Objects

# Specific requirements

### User Interfaces

### Hardware Interfaces

### Software Interfaces

### Communication Interfaces

## Functional requirements

### User Class 1 – The Logistician

### User Class 2 – The Driver

### User Class 3 – The Customer

## Non-functional requirements (external, performance, etc.)

## Design constraints

## Software System attributes