SOFTWARE REQUIREMENTS SPECIFICATION

for

Animal Transport Management Tool for AniTrans

Pre-Release

Version 0.2 approved

Prepared by Dave Meier, Olivier Ulrich, Luca Rolshoven, Julian Weyermann

Contents

Revision History

Name	Date	Reason For Changes	Version
Ananas	02.01.2017	Initial Release	Pre 0.1
Banana	04.01.2017	Adjusted due to more precise Project description	Pre 0.2

1 Introduction

1.1 Purpose

The purpose of this project is to create a piece of software which allows the AniTrans team to organize and schedule their jobs of transporting animals from point A to point B in Switzerland faster, more optimized and more flexible than the current solution with a lot of analog papers and calendars. Also the jobs can be modified by the tour-organisator at any given time while information on tours is available to all participants

1.2 Stakeholders

The stakeholders are the CEO of AniTrans, Mathias Fuchs and ESE-Team3 (Dave Meier, Olivier Ulrich, Luca Rolshoven, Julian Weyermann)

1.3 Definitions

- **Tour:** Starting from the base of AniTrans, the travel to the pick-up point of the freight, the travel to the deivery point, delivering the freight and travelling back to the base of AniTrans.
- **Unsucessful delivery:** A transport where the freight couldn't be delivered properly due to any circumstances. It will be sent back to the pick-up point and retried at a later point in time. The vehicle returns empty though and can be reused for another delivery.
- **Logistician:** Employees in the office who gathers jobs and the relevant data as tours and assigns them to a driver.
- **(Tour)driver:** Employees who do the tours.

1.4 System Overview

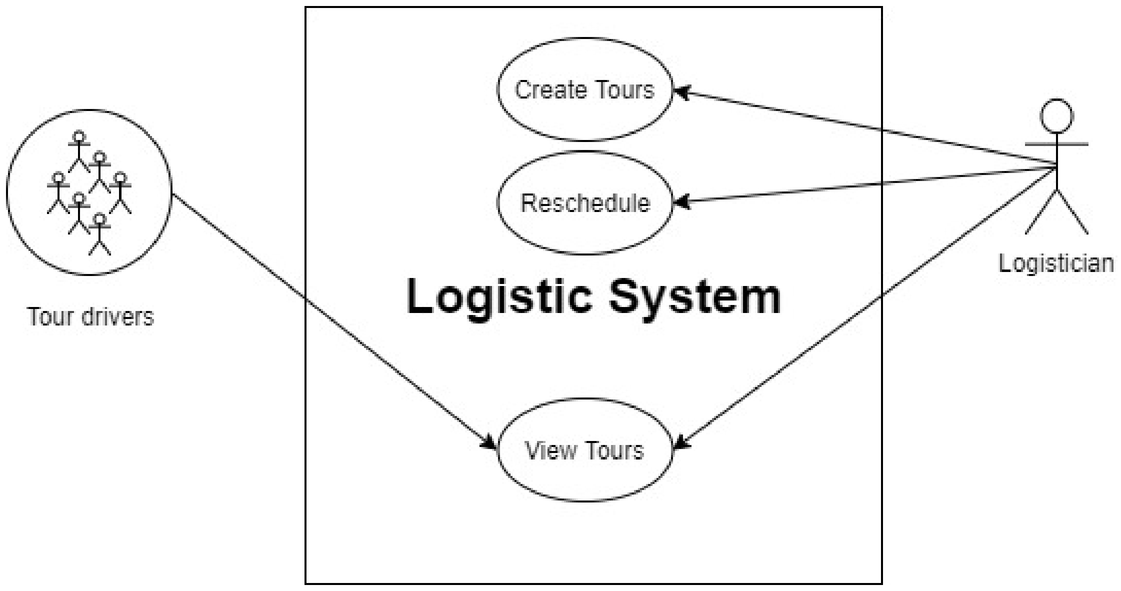
The users are the logistician and the tourdrivers. A tour as defined above is introduced by the logistician who can create it and assign it to a tourdriver additionally the logistician can define the order of the tours in which the driver has to deliver them. A tour consists of the animal species and the number of them, the pick-up point including the customer, the delivery point, the date of the creation of the tour (is this really neccessary?!?!?!?!), the starting time and date of the tour, the time window for delivery and also a manually

guessed tour duration. All those data can be entered into labeled fields on a webinterface on a browser.
Once a tour has started (according to the starting time/date), it can't be changed nor deleted.
The tourdriver can see the ordered tours (as assigned by the logistician) which are assigned to him. Once the tourdriver arrives at his destination, he can enter if the delivery was successful and he has the ability to write a short summary of how the tour went and what he has done in case the delivery was unsuccessful.

1.5 References

2 Overall Description

2.1 Use Cases



-

- **Drivers** log in through a webmask on their mobile device where they can see their current and upcoming tours. Once they delivered their freight, they can mark the tour as finished (= delivered) or failed (= no delivery possible) and also write down what issues they had to deal with. Once the tour is marked as finished or failed, it disappears, and the first one of the upcoming tours becomes the current one.

- The **Logistician** can gather the required data to create a tour on a different webinterface (from the one for the drivers) in a browser on a big screen. Furthermore he can modify existing tours and assign unassigned tours to a driver.

2.2 Actor Characteristics

While the logistician has many rights on the system and knows how to use it, the tourdriver can only see his tours in (by logistician) defined order and doesn't need to know anything about the system. It is (at least; in best case on both sides) on the driverside self-explanatory.

3 Specific requirements

3.1 Functional requirements

The system is able to contain the data of all tours (in the past, current and future) and process them in a useful way for the logistician. The orders must be sortable by all of the different data described in the System Overview section. Thus it should be possible to keep and view the data of past, current and upcoming tours as well as seing all failed tours at a glance.

3.2 Non-functional requirements

- Mobilefriendliness
- Real-time
- Possibly Multilanguage?!
- Java, Maven, Spring, MySql