SOFTWARE REQUIREMENTS SPECIFICATION

for

Animal Transport Management Tool for AniTrans

Pre-Release

Version 0.1.1 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
Coconut	12.11.2017	Added/Integrated usecases	v0.1.1

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 orders 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 orders 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/Administrator:** Employees in the office who receives orders and the gathers the relevant data, assembles it to 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 starting time and date of the tour, the time window for delivery and also a manually guessed tour duration. Furthermore the logistician can assign the

appropriate vehicle to the tour which then isn't available anymore. Which vehicles are available currently can be seen in a dedicated tab in the application. 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

- Registration: A driver who has not yet accessed the logistic application wants to register a new account.
  - Actors: Driver
  - Preconditions: The driver loaded the initial page of the website, where the logistic system resides.
  - Flow:
    - The driver sees the login screen and clicks on the registration link
    - A new page with a registration form loads, the driver enters the needed information and proceeds
    - The driver has now access to his/her daily tours as well as a weekly and monthly overview of all coming tours, that are already planned. Just after the registration, there wont be any tour yet.
  - Postconditions: The driver has now access to all the tours that he/she will drive as well as a daily, weekly and monthly overview of the tours.
- Time tracking: The driver wants to track the time he needs for the current delivery.
  - Actors: Driver
  - Flow:
    - The start date in a certain tour is passed.
    - The time tracking mechanisms starts.
    - The driver reaches the delivery point and stops the time tracking by clicking on the Stop button.
  - Exceptions: Something unexpected occurred and the delivery could not be done. The driver clicks the Report button and writes a little comment about what went wrong.
  - Postconditions: The delivery time was saved and can now be seen by the driver and of course by the administrator.
- Modifying a tour: An admin wants to change the details of an existing tour.
  - Actors: Admin

  - Preconditions: The admin is logged into the application and there is at least one tour saved already.
  - Flow:
    * The admin chooses a tour to edit.
    * On the right hand side of the screen, the admin sees all the data corresponding to the selected tour.
    * The admin changes the values of the fields he wants to change.
    * The admin clicks on the save Button.
  - Alternate Flow:
    * There arent any tours saved yet and the admin clicks on the notification to create a new tour
  - Exceptions: The admin forgets a field or enters an invalid value. Which causes an error to show up, specifying which fields need to be revisited.
  - Postconditions: The tour has been saved correctly to the DB and the chosen driver will see it on his feed.

- Creating a new Tour: An admin wants to create a new tour.
  - Actors: Admin
  - Preconditions: There exists an Admin account, and the admin is signed in.
  - Flow:
    * The admin clicks on a button which returns a form for creating a new tour.
    * The admin enters all the data neccessary and stated in the "System Overview" section of this SRS, and assigns it to a driver.
    * The admin clicks on a "Save" button, which will save the Data just entered as a new tour.
  - Exceptions: The admin forgets a field or enters an invalid value. Which causes an error to show up, specifying which fields need to be revisited.
  - Postconditions: The tour is saved as a new tour in the Database

- Firing a driver: The admin wants to fire one of his drivers
  - Actors: Admin
  - Predonditions: At least one driver exists.
  - Flow:
    * The admin loads the diver overview
    * The admin chooses a driver to fire
    * The admin clicks on the Fire button next to the drivers name.

    - A popup opens to confirm his decision
  - Postconditions: The driver has been deleted from the Database

- Vehicle Pool overview: The admin wants to check which vehicles are currently available
  - Actors: Admin
  - Preconditions: The Admin exists and is logged in; there is at least one vehicle in the Database.
  - Flow:
    - The Admin switches the "Tour"-Tab inside the application
    - All the Vehicles are shown and also which ones are currently available/in use.

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