

**SRS ZOO MANAGEMENT SYSTEM**

Authors:

Baiat Frumos Frumoasa Doamna

Version 1.1

Date : 11-05-2020

Table of content

[Context description 3](#_bookmark0)

[Actors 4](#_bookmark1)

[Requirements 5](#_bookmark2)

[Functional requirements 5](#_bookmark3)

[Non Functional requirements 5](#_bookmark4)

[System architecture 6](#_bookmark5)

[Domain Model 7](#_bookmark7)

[Use Case 8](#_bookmark9)

[Sequence Diagram time sheet approval 9](#_bookmark11)

[Look and Feel 10](#_bookmark13)

[Acceptance Testing 11](#_bookmark15)

# Context description

The Drienerloo zoo is looking for a new zoo management system to replace their current paper based system.

The zoo is a relative simple privately owned zoo by Anthony van den Berg. The zoo has different kinds of animals who are housed in cages. Cages can only contain the same kind of animals.

The zoo has three type of employees. Keepers who take care of the animals. Supervisors who manage the keepers. And administrators who handle all the administrative tasks including the finances.

Supervisors and Administrators earn a fixed wage. Keepers have a zero hour contract. They need to keep a timesheet. On basis of this timesheet they get their wages. The keeper should be able to manage his own timesheet via the management system. It would be nice if keepers can also keep track of there vacations to make the planning task of the administrators easier.

The supervisor needs to checks this timesheet. Before the administrator can pay the keepers. The norm for the check is that keepers should not spend more then 30 minutes per animal in the cage without a clear explanation.

Each keepers has his own specialty. So not all keepers can take care of all type of Animals, just the animals he is certified for.

The Drienerloo zoo expects the application to be delivered as a web app. It should be functional on all devices. The interface needs to be clear and simple, in that no training should be necessary for Keepers to use the system to keep track of their time.

The system should be safe to use and secure so that no unauthorized access to the system is possible. Keepers should not be able to see each other’s data. The data should be adequately backed up.

The system should also be fast. People don’t like waiting so loading should not take more then 3 seconds and is should be available 24/7. Although the zoo is small right now. The owner of the zoo wants us to make the system scalable. He does not want the system to restrict his growth.

The system will be developed by the software engineers of the Aplha scrum team of the company No Nonsense Software.

# Actors

The following actors have been identified.

1. The owner

The initiator of the project. Very important person!

1. Administrators

The system will change their way of working. A very important source of information

1. Supervisors

The system will change their way of working. An important source of information

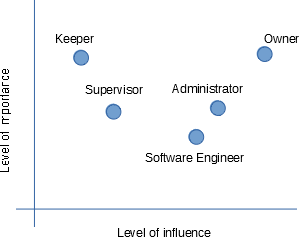
1. Keepers

Will have to use the system. But they do not have much influence in how the system will be developed

1. Software developers

Will develop the system. There choices will influence the lives of the other actors. But will not be using the system.

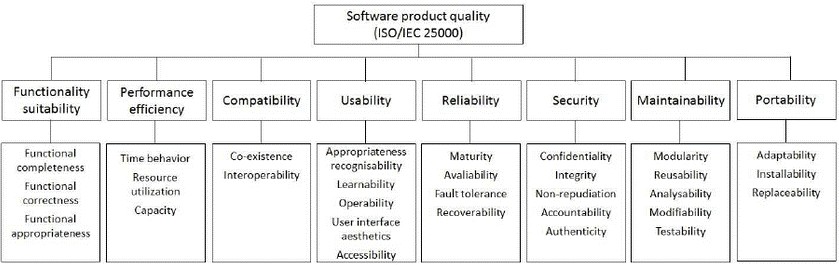
We have categorized these actors in the following Power Influence Matrix:



*Figure 1 Power influence Matrix*

# Requirements

We will categorize the requirements in functional an nonfunctional requirements. They will be prioritized according to MOSCOW categorized according to the ISO 25010 quality of software model.



*Figure 2 The iso 25010 model*

## Functional requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nr** | **Description** | **Source** | **ISO** | **Moscow** |
| F1 | Keepers should be able to manage their timesheet | 1 | FS | M |
| F2 | Only Supervisors can add employees to the system | 1 | FS | M |
| F3 | Supervisors need to check the timesheet before payment | 1 | FS | M |
| F4 | Cages can only hold one type of animal | 1 | FS | M |
| F5 | Each animal had to have a unique name | 1 | FS | M |
| F6 | Each keeper has a supervisor | 1 | FS | M |
| F7 | Keepers should be able to keep track of their vacations | 2 | FS | W |
| F8 | Keepers are certified on certain animals | 3 | FS | S |
| F9 | Each employee has a unique three digit number | 2 | FS | M |

## Non Functional requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nr** | **Description** | **Source** | **ISO** | **Moscow** |
| NF1 | The application needs to be a app that works on all devices | 1 | Portability / Install. | M |
| NF2 | The data should be backed up | 2 | Reliability / Recover. | M |
| NF3 | Only authorized uses can use the system | 2 | Security / Confident. | M |
| NF4 | Keepers should not see each other timesheet | 2 | Security / Confident. | M |
| NF5 | Pages should load within 3 seconds | 2 | Perf.Effic. / time beh. | S |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| NF6 | The interface needs to be clear and simple for the keepers | 2 | Usability / Learnability | S |
| NF7 | The system should be scalable | 1 | Perf. Effic. / Capacity | S |

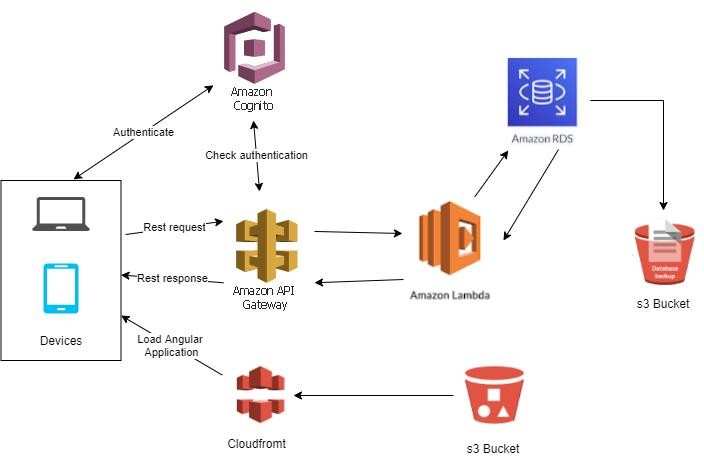
# System architecture

To make the system scalable and 24/7 accessible we will make use of the amazon cloud services of AWS as shown in [Figure 3.](#_bookmark6)

Cognito will be used to authenticate the users.

The devices of the users will get the Angular application from CloudFront via a secure SSL connection. This application uses the REST protocol to communicate with a the amazon API gateway via a secure connection. The API gateway passes the request on to the lambda functional layer. Witch retrieves or changes the requested data om an Amazon RDS MySQL database.

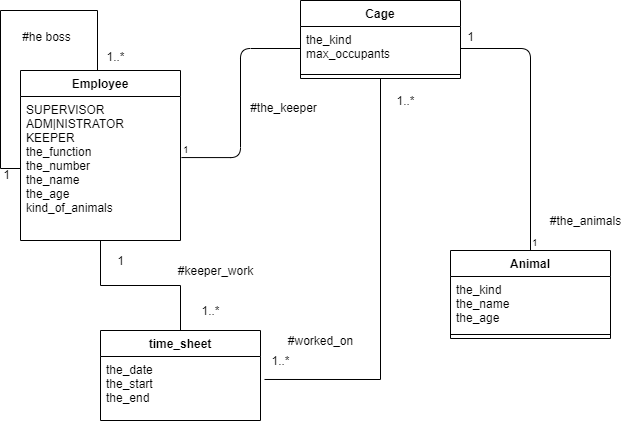
This database is backed up daily to an S3 bucket from which it can also be restored.



*Figure 3 ZMS system architecture*

# Domain Model

Our domain consists of cages animals employees and their corresponding timesheet as can be seen in [Figure 4.](#_bookmark8)



*Figure 4 the zoo domain model*

Employees can be of three kinds, Supervisor, Administrator or Keepers. Each keeper has a supervisor and can work on certain kind of animals. The keeper keeps a timesheet where he tracks on what date he has worked on a certain cage.

A supervisor has 1 or more keepers which he or she supervises.

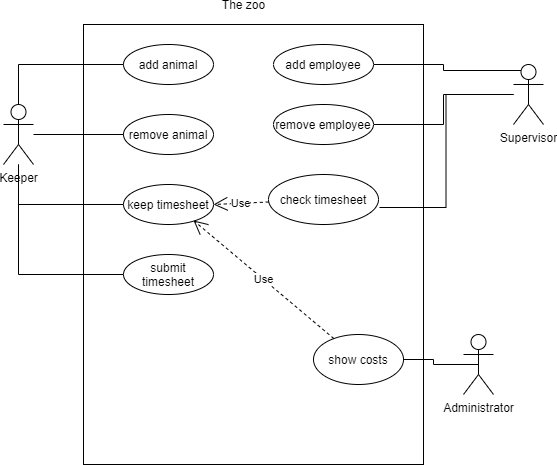
Cages can only contain one kind of animal and there is predetermined maximum number of animals to a cage. Each cage has a keeper who Is more or less the main keeper of the cage although other keepers can also work on the cage as long as they have the necessary skills.

# Use Case

The zoo contains three actors as can be seen in [Figure 5.](#_bookmark10)

Keepers can add and remove animals in cages, but only is the keeper is the cages primary care giver. Each keeper can keep his or her own timesheet. Once a week he or she needs to submit it to her supervisor who needs to approve his hours. Once submitted the timesheet for that week cannot be changed. Sometime the supervisor might ask the keeper for further details or explanation of the hours submitted. The keeper can then change the submitted timesheet and resubmit.

The supervisor can add and remove employees to the zoo and approve submitted timesheets. The Administrator can supply the total employee costs for a specific week.



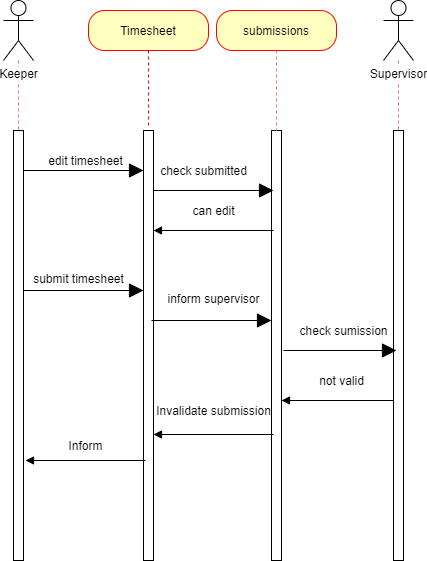
*Figure 5 Zoo USE CASE*

# Sequence Diagram time sheet approval

One of the key functionalities of the application is the keeping and approval of a timesheet by the keepers. This process can be viewed in [Figure 6.](#_bookmark12)

Keepers are allowed to edit there timesheets as long as they have not submitted them for approval by their supervisor.

After the keeper submits the timesheet the supervisor will be informed that he or she can approve the timesheet. If the supervisor needs more information the submitted timesheet can be invalidated and the keeper can resubmit it after applying corrections.



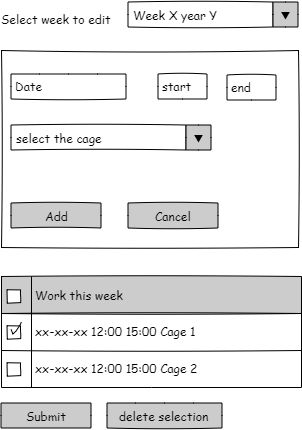
*Figure 6 Timesheet approval*

**Look and Feel Timesheet editing**

[Figure 7](#_bookmark14) shows a wireframe for the editing of the timesheet by the Keeper.

The keeper manages a timesheet per week. Only weeks which have not been submitted or have been invalidated will be shown in de combo box for selection.

The current date will be defaulted in the date field. The Keeper can only see the cages he is certified to keep. If he is the “owner” of a cage, this cage will default in the combo box to select a cage.



*Figure 7 Editing the timesheet*

# Acceptance Testing

When testing the system for acceptance we want to show that the system works according to the requirements.

We will provide a series op test suites. The traceability matrixes at the end of this chapter ( ) will show that we can demonstrate the correct presence of all the requirements.

## TS1 Test suite employees

### T1.1 Test Employee number

|  |  |  |
| --- | --- | --- |
| **Test-case** | **Add Employee** | **TC1.1.1** |
| Purpose | Verify employee number range  Good weather lower value of a boundary value test | |
| Actor | Manager | |
| Precondition | Employee 100 does not exist in the system | |
| Input | Employee number | 100 |
| Input | Other values are not relevant |  |
| Postcondition | Employee with number 100 has been accepted | |

|  |  |  |
| --- | --- | --- |
| **Test-case** | **Add Employee** | **TC1.1.2** |
| Purpose | Verify employee numbers outside of the scope are not accepted Bad weather lower value of a boundary value test | |
| Actor | Manager | |
| Precondition | None | |
| Input | Employee number | 99 |
| Input | Other values are not relevant |  |
| Postcondition | Employee with number 99 will generate an error and will not be added to the system | |

|  |  |  |
| --- | --- | --- |
| **Test-case** | **Add Employee** | **TC1.1.3** |
| Purpose | Verify employee numbers outside of the scope are not accepted Good weather upper value of a boundary value test | |
| Actor | Manager | |
| Precondition | None | |
| Input | Employee number | 999 |
| Input | Other values are not relevant |  |
| Postcondition | Employee with number 999 has been accpeted | |

|  |  |  |
| --- | --- | --- |
| **Test-case** | **Add Employee** | **TC1.1.4** |
| Purpose | Verify employee numbers outside of the scope are not accepted Bad weather upper value of a boundary value test | |
| Actor | Manager | |
| Precondition | None | |
| Input | Employee number | 1000 |
| Input | Other values are not relevant |  |
| Postcondition | Employee with number 101 will generate an error and will not be added to the system | |

|  |  |  |
| --- | --- | --- |
| **Test-case** | **Add Employee** | **TC1.1.5** |
| Purpose | Verify employee duplicate employee numbers not accepted | |
| Actor | Manager | |
| Precondition | Employee 100 does exist in the system | |
| Input | Employee number | 100 |
| Input | Other values are not relevant |  |
| Postcondition | Employee with number 100 will generate an error and will not be added to the system | |

### T1.2 Test only Mangers can add employees

|  |  |  |
| --- | --- | --- |
| **Test-case** | **Add Employee** | **TC1.2.1** |
| Purpose | Verify employee number range  Good weather lower value of a boundary value test | |
| Actor | Not a manager | |
| Precondition | none | |
| Input | Other values are not relevant |  |
| Postcondition | A non manager should not be able to add an employee | |

**TS2 Test application on multiple devices**

### T2.1 Test on multiple devices

|  |  |  |
| --- | --- | --- |
| **Test-case** | **Test application on desktop** | **TC2.1.1** |
| Purpose | Verify that the application works on a desktop | |
| Actor | Keeper | |
| Precondition | Windows OS with the browser chrome Select the timesheet functionality | |
| Input | Any valid date time and cage combination |  |
| Action | Click on add |  |
| Postcondition | The entered combination should be appended to the table The input section should be empty. | |
| Repeat | with the browser edge with the browser Firefox | |

|  |  |  |
| --- | --- | --- |
| **Test-case** | **Test application on iPhone** | **TC2.1.2** |
| Purpose | Verify that the application works on a iPhone | |
| Actor | Keeper | |
| Precondition | iPhone 6 or later with the browser safari Select the timesheet functionality | |
| Input | Any valid date time and cage combination |  |
| Action | Click on add |  |
| Postcondition | The entered combination should be appended to the table The input section should be empty. | |

|  |  |  |
| --- | --- | --- |
| **Test-case** | **Test application on iPad** | **TC2.1.2** |
| Purpose | Verify that the application works on a iPad | |
| Actor | Keeper | |
| Precondition | iPad third generation or later with the browser safari Select the timesheet functionality | |
| Input | Any valid date time and cage combination |  |
| Action | Click on add |  |
| Postcondition | The entered combination should be appended to the table The input section should be empty. | |

|  |  |  |
| --- | --- | --- |
| **Test-case** | **Test application on android** | **TC2.1.2** |
| Purpose | Verify that the application works on android | |
| Actor | Keeper | |
| Precondition | Any android phone with Android 4.4 or later with the browser chrome Select the timesheet functionality | |
| Input | Any valid date time and cage combination |  |
| Action | Click on add |  |
| Postcondition | The entered combination should be appended to the table The input section should be empty. | |

**Traceability matrix**

Below are two traceability matrixes. The first one describes what requirement is tested by which tests.

### Functional requirements

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Requirement Test | F1 | F2 | F3 | F4 | F5 | F6 | F7 | F8 | F9 |
| T1.1 |  |  |  |  |  |  |  |  | X |
| T1.2 |  | X |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

*Table 1 Traceability Matrix functional requirements*

### Non functional requirements

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Requirement Test | NF1 | NF2 | NF3 | NF4 | NF5 | NF6 | NF7 |
| T2.1 | X |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

*Table 2 Traceability matrix for the non functional requirements*