URS document

User Requirements Specification

**Group members:**

Radolina Petrova (4443527)

Stoil Yonchev (4435567)

Martin Todorov (4491114)

Danilo Gutesa (4334558)

Table of Contents

[Agreements and decision with the client 2](#_Toc96603329)

[Functional & Non-functional requirements 3](#_Toc96603330)

[Functional requirements: 3](#_Toc96603331)

[Non-functional requirements: 4](#_Toc96603332)

[Use cases 5](#_Toc96603333)

[UC-01: Manager should be able to create an account 5](#_Toc96603334)

[UC-02: Resource planners should be able to create schedules 6](#_Toc96603335)

[UC-03: User should be able to update their account details 7](#_Toc96603336)

[UC-04: Manager should be able to update animal information 8](#_Toc96603337)

[UC-05: User should be able to filter information about caretakers 9](#_Toc96603338)

[UC-06: Caretaker should be able to schedule a doctor appointment for an animal with health problems 10](#_Toc96603339)

[UC-07: User should be able to search for an animal by its species 11](#_Toc96603340)

# Agreements and decision with the client

1. Creation of different types of accounts – manager, caretaker, resource planner

* **Manager:** Creation of accounts, manage all the information about the animals and the employees
* **Resource planner:** Creation of schedules and assigning tasks to the caretakers
* **Caretaker:** Viewing the schedules

1. Contents of the feeding timetable

* The time the animal must be fed

1. Contents of breeding timetable

* Time of the year the animals are breedable. Keep track if breeding has been successful and if an offspring has occurred

1. Save account personal information regarding:

* First and last name, address, phone and email, emergency contact information, BSN, unique 5-digit ID
* Contract information: tasks that an employee performs, start date of employment, duration of the contract, FTE

1. Have a search functionality

* Look up employees by name and specialisation
* Get a list of all animals
* Look up feeding timetable of the animals

1. Store information about animals

* Name of the species and the animal itself
* ID of the animal
* Year of and reason for arrival
* Year of and reason for departure
* Location and cage number
* Diet
* Birthdate

|  |  |  |  |
| --- | --- | --- | --- |
| ***FR*** | ***Description*** | ***End user (Authorization)*** | ***Application*** |
| FR-01: | End user must be able to log in | Manager; Employee; Resource planner; | Desktop; |
| FR-02: | End user must be able to add an employee | Manager; | Desktop; |
| FR-03: | End user must be able to see all the employees | Manager; Resource planner; | Desktop; |
| FR-04: | End user must be able to see all the info about each of the employees | Manager; Resource planner; | Desktop; |
| FR-05: | End user must be able to remove an employee | Manager; | Desktop; |
| FR-06: | End user must be able to add an animal | Manager; | Desktop; |
| FR-07: | End user must be able to see all the animals | Managers; Resource planner; | Desktop; |
| FR-08: | End user must be able to see all the information about each of the animals | Manager; Resource planner; | Desktop; |
| FR-09: | End user must be able to remove an animal | Manager; | Desktop; |
| FR-10: | End user must be able to assign a task to an employee | Resource planner; | Desktop; |
| FR-11: | End user must be able to change the information about employees | Manager; | Desktop; |
| FR-12: | End user must be able to change the information about animals | Manager; | Desktop; |
| FR-13: | End user must be able to add a new species | Manager; | Desktop; |
| FR-14: | End user must be able to manage their account information | Manager; Resource planner; Employee | Desktop; |
| FR-15: | End user must be able to change location of a species | Manager; | Desktop; |
| FR-16: | End user must be able to change location of the animal (individually) | Manager; | Desktop; |
| FR-17: | End user must be able to store info about the employee in a database | Manager; | Desktop; |
| FR-18: | End user must be able to store info about animals in a database | Manager; | Desktop; |
| FR-19: | End user could be able to change a roll of an employee | Manager; | Desktop; |
| FR-20: | End user could be able to log into the web site | Managers; Employees; Customer; | Web; |
| FR-21: | End user could be able to buy a ticket | Customer; | Web; |

Functional & Non-functional requirements

Functional requirements:

## Non-functional requirements:

* Performance
* Scalability
* Authentication and authorization

Use cases

## UC-01: Manager should be able to add a new employee and create an account for them

**Pre-condition:** User needs to be logged in manager account on the desktop application

**Actor:** Manager

**Main success scenario:**

1. Actor clicks on the button to add a new employee
2. System requires information about the account that is being created
3. Actor fills up the information and confirms
4. System opens another form with the required information for the employee the account is being created for
5. Actor fills up the required information and confirms
6. System opens another form with the required information for the contract of the employee
7. Actor fills up the information and confirms
8. System sends a confirmation massage that the new account for the employee has been created

**Extensions:**

4a: Empty fields

.1: System displays an empty field message

.2: Actor fills up the missing information and confirms

.3: System sends a confirmation massage that new account is created

.4: End of use case

4b: Invalid information

.1: System displays invalid information massage

.2: Actor fills up the right information and confirms

.3: System sends a confirmation massage that new account is created

.4: End of use case

## UC-02: Manager should be able to update the information of an existing employee

**Pre-condition:** User needs to be logged in manager account on the desktop application

**Actor:** Manager

**Main success scenario:**

1. Actor clicks on the button to add a new employee
2. System requires information about the account that is being created
3. Actor fills up the information and confirms
4. System opens another form with the required information for the employee the account is being created for
5. Actor fills up the required information and confirms
6. System opens another form with the required information for the contract of the employee
7. Actor fills up the information and confirms
8. System sends a confirmation massage that the new account for the employee has been created

## UC-02: Resource planners should be able to assign caretakers to look after the animals in a cage

**Pre-condition:** User needs to be logged in

**Actor:** Resource planner

**Main success scenario:**

1. Actor chooses a date from the weekly schedule
2. System displays a daily schedule form
3. Actor chooses a time slot
4. System displays all the cages with animals that should be fed in that part of the day
5. Actor chooses a cage they want to assign a caretaker to
6. System displays the information about the animals in the cage and the specialized caretakers
7. Actor chooses a specialized caretaker and assigns them to a certain cage
8. System saves the information into the caretaker’s schedule

**Extensions:**

1a: Actor chooses a past date from the schedule

.2: System displays a daily schedule form

.3: Actor choose a time slot

.4: System displays all the cages with animals that were supposed to be fed in that part of the day

.5: Actor chooses a cage

.6: System displays the information about the animals in the cage and the caretaker, that was assigned to them

.2: End of use case

## UC-03: Resource planners should be able to edit the assigned caretaker

**Pre-condition:** User needs to be logged in

**Actor:** Resource planner

**Main success scenario:**

1. Actor chooses a date from the weekly schedule
2. System displays a daily schedule form
3. Actor chooses a time slot
4. System displays all the cages with animals that should be fed in that part of the day
5. Actor chooses a cage they want to edit the assigned caretaker
6. System displays the information about the animals in the cage and the assigned specialized caretakers
7. Actor chooses another caretaker and assigns them to the certain cage
8. System updates the information into the caretaker’s schedule

**Extensions:**

1a: Actor chooses a past date from the schedule

.2: System displays a daily schedule form

.3: Actor choose a time slot

.4: System displays all the cages with animals that were supposed to be fed in that part of the day

.5: Actor chooses a cage

.6: System displays the information about the animals in the cage and the caretaker, that was assigned to them

.2: End of use case

## UC-04: User should be able to update their account details

**Pre-condition:** User needs to be logged in their account

**Actor:** Manager, Resource-planner, Caretaker

**Main success scenario:**

1. Actor goes to the settings of the account
2. System displays the information of the currently logged in actor
3. Actor updates the information and confirms
4. System sends a confirmation massage that the information is updated

**Extensions:**

3a: Empty fields

.1: System displays an empty field message

.2: Actor fills up the missing information and confirms

.3: System sends a confirmation massage that the information is updated

.4: End of use case

3a: New information matches the old one

.1: System displays same data message

.2: User fills up new data and confirms

.3: System sends a confirmation massage that the information is updated

.4: End of use case

## UC-05: Manager should be able to update animal information

**Pre-condition:** User needs to be logged in a manager account

**Actor:** Manager

**Main success scenario:**

1. Actor searches for the animal, which information they want to change
2. System displays the information about the animal
3. Actor selects the information they want to update and confirms
4. System sends a confirmation massage

**Extensions:**

3a: Fields are empty

.1: System displays an empty field message

.2: Actor fills up the missing information and confirms

.3: System sends a confirmation massage

.4: End of use case

## UC-06: User should be able to filter information about caretakers

**Pre-condition:** User needs to be logged in

**Actor:** Manager, Resource planner

**Main success scenario:**

1. Actor goes to the tab with information about the employees
2. System displays all the current employees of the zoo franchise
3. Actor searches for employee by name or by specialisation
4. System displays the filtered information

**Extensions:**

4a: There are no results

.1: System displays no matching information message

.2: End of use case

## UC-07: Caretaker should be able to schedule a doctor appointment for an animal with health problems

**Pre-condition:** User must be logged in a caretaker account

**Actor:** Caretaker

**Main success scenario:**

1. Actor goes to the personal information of the animal with health problems
2. System displays all the information
3. Actor clicks on a button to schedule an appointment
4. System displays a form with all the needed information to fill for an appointment
5. Actor fills up the information and confirms

## UC-08: User should be able to search for an animal by its species

**Pre-condition:** User must be logged in

**Actor:** Manager, Resource-planner, Caretaker

**Main success scenario:**

1. Actor goes to the tab with information about the animals
2. System displays all the current animals in the zoo
3. Actor searches for a certain species
4. System displays the filtered information

**Extensions:**

4a: There are no results

.1: System displays no matching information message

.2: End of use case

## UC-09: Caretakers should be able to view their schedule in the web application