***Zoo Bazaar***

URS

**Logo, company name

Description automatically generated**

**Date:** 02/03/2023

**Made by**:

**Version:** Version 1.0

**Status:** Open

# Agreements with clients

Our team has been given an exciting challenge: to create a Windows Form application for a zoo that will manage both employee information and animal care. We have been tasked with using C# as the primary programming language, and ensuring that the application can be used both on a website and Windows Form application. Additionally, we will be working with a SQL database to store and manage the data for both employees and animals.

Over the next six weeks, we will work collaboratively to design and develop the application, ensuring that it meets all of the requirements set out for us. We will need to create user-friendly interfaces for both the employee and animal management components, as well as ensuring that the application is secure and reliable. Ultimately, our goal is to create a high-quality, functional application that will enable the zoo to effectively manage its staff and provide the best possible care for its animals.

We also need to keep in mind that the app should be easy to extend and recreate, so we need to follow good programming practices, document our choices, and use appropriate design patterns. By doing this, we can ensure that the app can handle any changes that may arise in the future.

Our application has two core functionalities: employee management and animal care. To support employee management, we require an administration page that enables the administrator to manage the employee database. This page should provide the administrator with the ability to modify existing employee information, create new employee profiles, and remove outdated employee data.

For animal care, we need to develop a comprehensive system that allows us to track important information about each animal, including its name, location, health status, category, dietary needs, social relationships, and history. This system should also be accessible to the administrator, who can use it to manage the animals.

# 2. Functional Requirements

## 2.1 Core Requirements

* FR-01: Animal management
* FR-01-A: Administrator should be able to Create new animal.
* FR-01-B: Administrator should be able to Update an existing animal.
* FR-01-C: Administrator should be able to Remove an animal.
* FR-01-D: Administrator should be able to Search for an animal.
* FR-01-E: Administrator should be able to view details of animal.
* FR-02: Employee management
* FR-02-A: Administrator should be able to add an employee to the system.
* FR-02-B: Administrator should be able to update information of existing employee.
* FR-02-C: Administrator should be able to search employee.
* FR-02-D: Administrator should be able to remove existing employee.
* FR-03: User Management
* FR-03-A: User should be able to login to the system.
* FR-03-B: User should be able to logout to the system.
* FR-04: Security
* FR-04-A: Storing data in SQL provides increased privacy and security, as not everyone has unfettered access to the data.

## 2.2 Major Requirements

* FR-05: Animal management system
  + FR-05-A: System should be easy to navigate.
  + FR-05-B: System should provide a list of all the species.
  + FR-05-C: System should have a detailed page for each specie.
* FR-06: Employee management system
  + FR-06-A: System should be easy to navigate.
  + FR-06-B: System should be only accessible for Administrators.
  + FR-06-C: System should provide a list of all the employees.
  + FR-06-D: System should have a detailed page of each Employee.
* FR-07: Employee Schedule
  + FR-07-A: System should have an Schedule for employees.
  + FR-07-B: Schedule page should be editable by the HR.
  + FR-07-C: Schedule should make it possible to see multiple details.
* FR-08: animals schedule
  + FR-08-A: System should have an schedule for animals.
  + FR-08-B: Schedule should provide information of the animals of that day.

## 2.3 Minor Requirements

* FR-09: Notes
  + FR-09-A: System should have a option to leave notes.
* FR-10: Ticketing and admission management
  + FR-10-A: The system should be able to manage ticket sales and admissions for the zoo, including online and in-person purchases.
  + FR-10-B: The system should be able to track visitor numbers and generate reports on attendance and revenue.
* FR-11: Education and outreach Management
  + FR-11-A: The system should be able to manage and schedule educational programs and outreach events for the zoo, including school visits, public talks, and community outreach activities.
  + FR-11-B: The system should be able to track attendance and generate reports on program effectiveness and community engagement.
  + FR-12-C: The system should be able to manage and track the distribution of educational materials, such as brochures and flyers, to visitors and community partners.

|  |  |  |  |
| --- | --- | --- | --- |
| Must | Should | Could | Won’t |
| FR-01,  FR-02,  FR-03,  FR-04,  FR-05,  FR-06, | FR-07,  FR-08,  FR-09, | FR-10 | FR-11 |

# 3. Use Cases

**Use Case – FR-01-A:** Animal management

**Actor:** Administrator

**Description:** Add an animal to the zoo

**Main Success Scenario:**

1. Actor navigates to the animal management section of the system.
2. Actor selects “Add new animal” button.
3. System displays the animal creation form.
4. Actor fills in the required information about the animal.
5. Actor selects “Save ” button.
6. System validates the form data.
7. System updates the zoo’s inventory.
8. System notifies the actor that the animal has been added successfully.

**Extensions:**

**5a.** Actor has not filled out all required fields

1. System displays an error message indicating the required fields that have not been filled.
2. Actor fills in the missing information.
3. Return to MSS step 4.

**5b**. Actor enters invalid data in one or more fields

1. System displays an error message indicating the invalid data entered.
2. Actor corrects the invalid data.
3. Return to MSS step 4.

**Use Case – FR-01-B:** Animal management

**Actor:** Administrator

**Description:** Update an existing animal

**Main Success Scenario:**

1. Actor selects an animal from the list of animals
2. Actor selects modify button
3. System redirects actor to animal information page
4. Actor provides new data and selects save button
5. System notifies the actor that the animal has been updated

**Extensions:**

**4a.** Actor has not filled out all fields

1. System notifies the actor to fill in all fields
2. Return to MSS step 4

**Use Case – FR-01-C:** Animal management

**Actor:** Administrator

**Description:** Remove an animal

**Main Success Scenario:**

1. Actor selects an animal from the list of animals
2. Actor selects remove button
3. Systems prompt actor with message "Confirm removal?"
4. Actor selects confirm button
5. System notifies that the item has been removed

**Extensions:**

**5a.** System is not able to remove the selected animal

1. System notifies the user that the selected animal could not be removed
2. Return to MSS step 1

**Use Case – FR-01-D:** Animal management

**Actor:** Administrator

**Description:** Search for an animal

**Main Success Scenario:**

1. Actor selects the search box
2. Actor inserts search data and selects search button
3. System displays the search results

**Extensions:**

**2a.** search data is not found.

1. Actor fills in search query.
2. No items match the searched query.
3. System notifies the actor that there are no results.
4. Return to MSS step 1.

**Use Case – FR-01-E:** Animal management

**Actor:** Administrator  
**Description:** View animal details

**Main Success Scenario:**

1. Actor selects an animal from the list of animals
2. Actor selects view details.
3. System redirects the actor to animal information page

**Use Case – FR-02-A:** Employee management

**Actor:** Administration

**Description:** Add an employee to the system

**Main Success Scenario:**

1. Actor selects add new employee button
2. System redirects actor to the employee creation page
3. Actor provides data of the employee and selects save button
4. System notifies the actor that a new employee has been created

**Extensions:**

**3a.** Actor has not filled out all fields

1. System notifies the actor to fill in all fields
2. Return to MSS step 3

**Use Case – FR-02-B:** Employee management

**Actor:** Administration

**Description:** Update existing employee

**Main Success Scenario:**

1. Actor selects an employee
2. Actor selects modify button
3. System redirects actor to employee information page
4. Actor provides new data and selects save button
5. System notifies the actor that the employee has been updated

**Extensions:**

**4a.** Actor provides invalid information

1. System notifies the acter to fill in valid information.
2. Return to MSS step 4.

**Use Case – FR-02-C:** Employee management

**Actor:** Administration

**Description:** Search for an employee

**Main Success Scenario:**

1. Actor selects the search box
2. Actor inserts search data and selects search button
3. System displays the search results

**Extensions:**

**2a.** search data is not found.

1. Actor fills in search query.
2. No items match the searched query.
3. System notifies the actor that there are no results.
4. Return to MSS step 1.

**Use Case – FR-02-D:** Assign task to an employee

**Actor:** Resource Planner

**Main Success Scenario:**

1. Actor selects an employee
2. Actor selects assign task button
3. Actor selects task type and animal related to the task
4. Actor selects save button
5. System notifies the actor that a new task has been created

**Use Case – FR-03-A:** User Management

**Actor:** User

**Description:** Login to the system

**Main Success Scenario:**

1. Actor inserts account credentials
2. Actor selects Login button
3. System notifies the actor that the login is successful

**Extensions:**

**1a**. Actor inputs the wrong credentials

1. System notifies the customer that the credentials are incorrect
2. Return to MSS step 1

**Use Case – FR-03-B:** User Management

**Actor:** User

**Description:** Logout of the system

**Main Success Scenario:**

1. Actor selects logout button
2. System notifies the actor that they are logged out successfully

**Use Case – FR-05:** Animal management system

**Actor:** Administrator

**Description:** Administrator want to view detail page of specific animal

**Main Success Scenario:**

1. Actor navigates to animal page.
2. Actor selects specific animal.
3. System opens detail page of the specific animal.

**Use Case – FR-06:** Employee management system

**Actor:** Administrator

**Description:** Administrator want to view detail page of specific employee

**Main Success Scenario:**

1. Actor navigates to Employee page.
2. Actor selects specific employee and clicks on more details.
3. System opens detail page of the specific Employee.