Technical Documentation

Arcadia Zoo - 2024

INFORMATIONS

Project Name	Arcadia Zoo
Document Type	Technical Documentation
Date	May 22, 2024
Author	Lorenzo FLORENTY

TABLE DES MATIÈRES

1 - Document Summary

This document is the official technical documentation for Arcadia's zoo management web application, Arcadia Zoo. It is divided into … parts :

- Initial Thoughts on the technologies used
- Bla
- Bla

2 - Reminder of the main information about the project

2.1 - Project Description

The project is focused on creating a user-friendly and eco-friendly web application for Zoo Arcadia in France.

The application will allow visitors to explore the different sections of the zoo, learn about the services available, and access schedules. The key features of the application include a home page that introduces the zoo, habitats, services, and reviews. The application menu will facilitate navigation and provide access to services, habitats, login, and a contact option.

The services view will display various services with customization options, while the habitats view will provide details about the habitats and associated animals. Visitors will also be able to leave comments, subject to validation, in the reviews section.

The administrator area will allow for the management of accounts, services, schedules, habitats, animals, and veterinary reports. The employee area will enable the validation of notices, modification of services, and registration of animal feeds. The veterinarian area will allow for the input of animal reports and comments on habitats.

The application will also have login functionality for administrators, veterinarians, and employees only.

Lastly, there will be a contact option for visitors to reach out to the zoo through a form, as well as a statistics feature to track habitat consultations and provide analytical data.

2.2 - Project Breakdown

The Arcadia Zoo project breaks down as follows:

FRONT-END :

I. **User Interface :**

Home page presenting the zoo, habitats, services and reviews.

Navigation menu to access the different sections of the application.

II. **Data view :**

Services view: Displays the zoo's various services with configuration options.

Habitats view: Presentation of habitats and associated animals with details.

III. **User Interaction :**

Leave feedback: Visitors can submit reviews, subject to validation.

IV. **Contact :**

Contact form for visitors to contact the zoo.

BACK-END :

I. **User management :**

Creation of administrator, veterinary and employee accounts.

Authentication and management of user sessions.

II. **Content management :**

Management of services, habitats and animals.

Validation of reviews submitted by visitors.

III. **Veterinary features :**

Enter reports on animals and habitats.

Consultation of animal feed data.

IV. **Employee functions :**

Validate reviews submitted by visitors.

Modification of zoo services.

Record daily animal feed.

V. **Statistics :**

Monitoring of animal consultations to provide analytical data.

DATA MANAGEMENT :

I. **Data Storage :**

Using a database to store information about users, reviews, services, habitats and animals.

II. **Data Processing :**

Handling of data for the various functionalities of the application.

III. **Security :**

Implementation of security measures to protect sensitive data and access to the application.

3 - Initial Thoughts on technologies used

The project involves creating a Web application for Arcadia Zoo, with a user-friendly administration interface and a dynamic Front-End. Object-oriented programming will be used to efficiently structure the various players (administrators, employees, veterinarians) and entities (animals, habitats, user opinions, etc). The use of a robust Back-End is necessary to feed the dynamic Front-End with data updated in real time.

In terms of databases, the idea is to combine a relational database for sensitive data and a non-relational database for simple data. This approach will minimise redundancies and ensure adequate data security.

The choice of frameworks was Symfony for the Back-End, because of its flexibility, robust architecture and compatibility with EasyAdmin to simplify the development of the administration interface. Although React JS was considered for the Front-End, a more traditional approach with HTML, Twig and CSS was preferred for a better separation of concerns and simpler management of the Front-End.

In terms of databases, PostgreSQL and MongoDB were chosen for their ease of implementation and good performance. These technological choices were made taking into account the specific needs of the project, with a constant concern for security, reliability and performance.