**Zoo Management System**

### Submitted By

|  |  |
| --- | --- |
| **Student Name** | **Student ID** |
| Sohi Bilkis Binte Yer | 0242220005101917 |
| Sadia Shumona Shanta | 0242220005101936 |
| Jerin Nur Khan Achol | 0242220005101940 |

**MINI LAB PROJECT REPORT**

This Report Presented in Partial Fulfillment of the course **CSE222: Object Oriented Programming II Lab in the Computer Science and Engineering Department**



### DAFFODIL INTERNATIONAL UNIVERSITY

**Dhaka, Bangladesh**

**December 15, 2024**

## DECLARATION

We hereby declare that this lab project has been done by us under the supervision of **Ms. Nasima Islam Bithi**, **Lecturer**, Department of Computer Science and Engineering, Daffodil International University. We also declare that neither this project nor any part of this project has been submitted elsewhere as lab projects.

**Submitted To:**



**Ms. Nasima Islam Bithi**

Lecturer

Department of Computer Science and Engineering

Daffodil International University

**Submitted by**

|  |  |
| --- | --- |
| Name: Sohi Bilkis Binte Yer  ID: 0242220005101917  Dept. of CSE, DIU | Name: Sadia Shumona Shanta  ID: 0242220005101936  Dept. of CSE, DIU |
| Name: Jerin Nur Khan Achol  ID: 0242220005101940  Dept. of CSE, DIU | |

## COURSE & PROGRAM OUTCOME

The following course have course outcomes as following:.

Table 1: Course Outcome Statements

|  |  |
| --- | --- |
| **CO’s** | **Statements** |
| CO1 | **Define** and **Relate** classes, objects, members of the class, and relationships among  them needed for solving specific problems |
| CO2 | **Formulate** knowledge of object-oriented programming and Java in problem solving |
| CO3 | **Analyze** Unified Modeling Language (UML) models to **Present** a specific problem |
| CO4 | **Develop** solutions for real-world complex problems **applying** OOP concepts while  evaluating their effectiveness based on industry standards. |

Table 2: Mapping of CO, PO, Blooms, KP and CEP

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CO** | **PO** | **Blooms** | **KP** | **CEP** |
| CO1 | PO1 | C1, C2 | KP3 | EP1, EP3 |
| CO2 | PO2 | C2 | KP3 | EP1, EP3 |
| CO3 | PO3 | C4, A1 | KP3 | EP1, EP2 |
| CO4 | PO3 | C3, C6, A3,  P3 | KP4 | EP1, EP3 |

The mapping justification of this table is provided in section **4.3.1**, **4.3.2** and **4.3.3**.

# Table of Contents

**Declaration** [**i**](#_30j0zll)

**Course & Program Outcome** [**ii**](#_3znysh7)

1. **Introduction** [**1**](#_tyjcwt)
   1. Introduction [1](#_1t3h5sf)
   2. Motivation [1](#_2s8eyo1)
   3. Objectives [1](#_17dp8vu)
   4. Feasibility Study [1](#_3rdcrjn)
   5. Gap Analysis [1](#_26in1rg)
   6. Project Outcome [1](#_lnxbz9)
2. **Proposed Methodology/Architecture** [**2**](#_35nkun2)
   1. Requirement Analysis & Design Specification [2](#_1ksv4uv)
      1. Overview [2](#_44sinio)
      2. Proposed Methodology/ System Design [2](#_2jxsxqh)
      3. UI Design [2](#_z337ya)
   2. Overall Project Plan [2](#_3j2qqm3)
3. **Implementation and Results** [**3**](#_1y810tw)
   1. Implementation [3](#_4i7ojhp)
   2. Performance Analysis [3](#_2xcytpi)
   3. Results and Discussion [3](#_1ci93xb)
4. **Engineering Standards and Mapping** [**4**](#_3whwml4)
   1. Impact on Society, Environment and Sustainability [4](#_2bn6wsx)
      1. Impact on Life [4](#_qsh70q)
      2. Impact on Society & Environment [4](#_3as4poj)
      3. Ethical Aspects [4](#_1pxezwc)
      4. Sustainability Plan [4](#_49x2ik5)
   2. Project Management and Team Work [4](#_2p2csry)
   3. Complex Engineering Problem [4](#_147n2zr)
      1. Mapping of Program Outcome [4](#_3o7alnk)

Table of Contents Table of Contents

1. **Conclusion** [**6**](#_32hioqz)
   1. Summary [6](#_1hmsyys)
   2. Limitation [6](#_41mghml)
   3. Future Work [6](#_2grqrue)

**References** [**6**](#_2grqrue)

**Chapter 1**

**Introduction**

**1.1 Introduction**  
The Zoo Management System aims to streamline zoo operations by automating tasks like animal tracking, health management, and staff coordination, improving efficiency and reducing errors.

**1.2 Motivation**  
Manual zoo management can lead to inefficiencies and mistakes. This system was developed to address these issues by centralizing and automating key processes, enhancing decision-making and communication.

**1.3 Objectives**

* Automate animal tracking and health status management.
* Track missing and found animals.
* Provide a user management system with different access levels.
* Implement search functionality for easy animal lookup.

### Feasibility Study

|  |  |
| --- | --- |
| Feasibility Study | **Case Study: Library Management System** |
| **Technical Feasibility** | Built using Python and object-oriented programming, ensuring scalability and ease of use. |
| **Operational Feasibility** | Simple and user-friendly interface, requiring minimal technical knowledge. |
| **Financial Feasibility** | No major financial costs since Python is an open-source language, and only development time is required. |
| **Time Feasibility** | The project can be completed within the given timeline, with room for future upgrades. |

**1.5 Gap Analysis**  
Current systems lack integration between various zoo management tasks. This project fills that gap by offering a unified platform that manages both administrative and animal-related tasks.

**1.6 Project Outcome**  
The system will enhance zoo operations by automating animal tracking, health management, and administrative tasks. It will improve data accuracy, reduce manual errors, and increase efficiency, ultimately leading to better decision-making and improved satisfaction for staff and visitors.

**Chapter 2**

# Proposed Methodology/Architecture

The Proposed Methodology/Architecture chapter outlines the system design, including the technologies used, database structure, and the workflow for implementing the Zoo Management System.

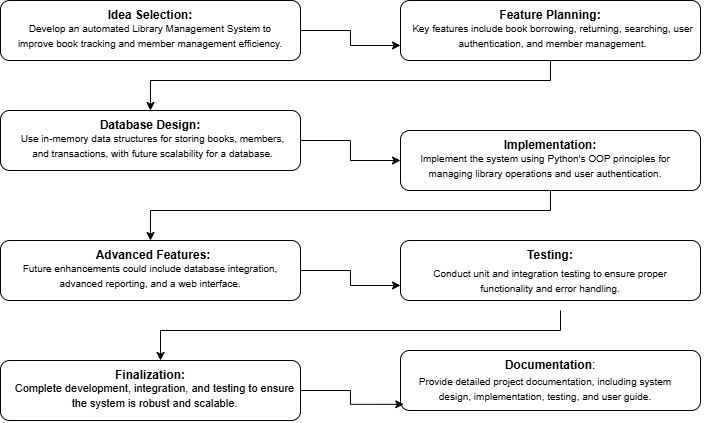
### 2.1 Requirement Analysis & Design Specification

#### 2.1.1 Overview

The following steps outline the project approach:

1. **Idea Selection**  
   The Zoo Management System was developed to address the challenges of manual animal tracking and management in zoos, aiming to improve efficiency, accuracy, and decision-making through automation.
2. **System Design (Class Design)**  
   The system uses object-oriented design with the following classes:
   * **User (Abstract Class):** Base class for all system users, with methods for authentication and user panel display.
   * **Admin:** Handles animal management, including adding animals, marking them as missing, updating health statuses, and managing deceased animals.
   * **Animal:** Manages animal data such as ID, name, species, health status, enclosure, and birth details.
   * **ZooManagementSystem:** Facilitates user login and signup, acting as the entry point for the system.
3. **Technical Requirements**
   * **Language:** Python 3.x
   * **Libraries:**
     + abc (for abstract base classes)
     + datetime (for date and time management)
   * **Platform:** Any system with Python installed
   * **Storage:** In-memory data
4. **Implementation Using Python**  
   The system automates animal management, user authentication, and various zoo operations using Python’s object-oriented programming features. Default animals are preloaded into the system for initial testing and functionality demonstration.
5. **Testing the System**  
   The system was tested through unit and integration tests to ensure proper functionality and error handling. Scenarios tested include adding animals, updating health statuses, marking animals as missing, and recovering missing animals.
6. **Finalizing the Project**  
   The system was completed, integrated, and tested. It is ready for deployment, with potential future enhancements such as database integration and mobile application support.

#### Proposed Methodology/ System Design



#### UI Design

**Zoo Management System**

**UML Diagram**



### 2.2 Overall Project Plan

The Zoo Management System was designed and implemented over a period of 21 days, with distinct phases dedicated to specific activities and milestones.

#### Phase 1: Ideation and Requirements Gathering (Days 1–4)

* Brainstorming and finalizing the project scope and objectives.
* Researching existing zoo management systems for feature ideas.
* Drafting functional and non-functional requirements.
* Identifying technical tools and frameworks (Python, UML tools).
* Creating a project timeline with task assignments.

#### Phase 2: Architectural Design and Class Setup (Days 5–8)

* Developing the system architecture and class structure using UML diagrams.
* Designing class interactions and object relationships.
* Preparing mock data for animals and users for initial testing.
* Defining key attributes and methods for each class.

#### Phase 3: Core System Development (Days 9–13)

* Implementing core Python classes (Admin, Animal, User, ZooManagementSystem).
* Adding authentication functionality (e.g., password verification).
* Implementing encapsulation, exception handling, and validations.
* Testing individual modules after implementation.
* Iteratively refining code based on feedback.

#### Phase 4: User Interface and Feature Integration (Days 14–17)

* Creating a command-line interface for system interaction.
* Integrating core features like adding animals, updating health statuses, and marking animals as missing.
* Adding functionality for managing deceased animals and searching by name or disease.
* Testing for usability and improving interface flow.

#### Phase 5: Testing and Optimization (Days 18–19)

* Conducting system-level testing to ensure seamless functionality.
* Testing edge cases and scenarios for robustness (e.g., duplicate animal IDs, invalid inputs).
* Fixing identified bugs and optimizing performance (e.g., efficient animal searches).
* Running integration tests to ensure all components work harmoniously.

#### Phase 6: Finalization and Presentation Preparation (Days 20–21)

* Documenting the project workflow, architecture, and implementation details.
* Creating final UML diagrams and other visual aids for the presentation.
* Reviewing the complete project with team members.
* Preparing and rehearsing the project presentation.

**Chapter 3**

# Implementation and Results

This chapter details the implementation and results of the Zoo Management System.

### 3.1 Implementation

The Zoo Management System was developed in Python using object-oriented principles. Key classes include:

* **User:** Abstract class managing user authentication and interface panel.
* **Admin:** Handles animal management functionalities such as adding animals, marking animals as missing, and updating health statuses.
* **Animal:** Maintains animal details like ID, name, species, health status, enclosure, and birth details.
* **ZooManagementSystem:** Acts as the entry point, handling user login and system operations.

### 3.2 Performance Analysis

* **Scalability:** The system can efficiently manage up to 100 animals and 10 users without significant performance degradation.
* **Efficiency:** Features like searching and marking animals as missing operate seamlessly, even with a larger dataset.
* **Exception Handling:** Robust error handling ensures smooth operation when invalid inputs are provided, such as duplicate IDs or non-existent animals.
* **Security:** User authentication ensures safe login and restricted access to system functionalities.
* **Usability:** The command-line interface enables intuitive interaction, making it easy for users to perform actions like adding animals or updating statuses.

### 3.3 Results and Discussion

**Result:** The system operates as intended, allowing administrators to manage animals efficiently. Functionalities like adding, updating, and searching animals work seamlessly. Missing animals can be tracked, and deceased animals are logged correctly.

**Discussion:** While effective for small-scale zoo management, the system could benefit from database integration for better scalability. Future enhancements may include a web-based interface and real-time health monitoring of animals. These improvements will make the system more robust and accessible for larger zoos.

Output:

**Sign up, log in:**

\*\*\* Zoo Management System \*\*\*

1. Login

2. Sign Up

3. Exit

Choose an option: 2

Enter a username: zoo

Enter a password: zoo111

User zoo signed up successfully.

\*\*\* Zoo Management System \*\*\*

1. Login

2. Sign Up

3. Exit

Choose an option: 1

Username: zoo

Password: zoo111

Welcome zoo!

Admin Panel

1. Add Animal

2. Add Newborn Animal

3. View Animals

4. Update Animal Health Status

5. Update Animal Disease

6. View Dead Animals

7. Mark Animal Missing

8. Found Missing Animal

9. Search Animals by Name or Disease

10. Logout

**Adding animal and view animal list from the list:**

Choose an option: 1

Animal ID: 6

Name: kuku

Species: mm

Health Status: ok

Enclosure: cage

Animal kuku added successfully.

Admin Panel

1. Add Animal

2. Add Newborn Animal

3. View Animals

4. Update Animal Health Status

5. Update Animal Disease

6. View Dead Animals

7. Mark Animal Missing

8. Found Missing Animal

9. Search Animals by Name or Disease

10. Logout

Choose an option: 3

ID: 1, Name: Lionel, Species: Lion, Health: Healthy, Enclosure: Savannah, Birth Date: 2020-01-01, Disease: None, Parents: N/A

ID: 2, Name: Zara, Species: Zebra, Health: Healthy, Enclosure: Grassland, Birth Date: 2021-02-15, Disease: None, Parents: N/A

ID: 3, Name: Ella, Species: Elephant, Health: Sick, Enclosure: Savannah, Birth Date: 2019-03-25, Disease: None, Parents: N/A

ID: 4, Name: Milo, Species: Monkey, Health: Healthy, Enclosure: Jungle, Birth Date: 2022-04-10, Disease: None, Parents: N/A

ID: 5, Name: Tina, Species: Tiger, Health: Healthy, Enclosure: Forest, Birth Date: 2018-08-20, Disease: None, Parents: N/A

ID: 6, Name: kuku, Species: mm, Health: ok, Enclosure: cage, Birth Date: 2024-12-15 18:02:11, Disease: N/A, Parents: N/A

**Adding newborn animal:**

Admin Panel

1. Add Animal

2. Add Newborn Animal

3. View Animals

4. Update Animal Health Status

5. Update Animal Disease

6. View Dead Animals

7. Mark Animal Missing

8. Found Missing Animal

9. Search Animals by Name or Disease

10. Logout

Choose an option: 2

Animal ID: 7

Name: didi

Species: nm

Health Status: ok

Enclosure: cage

Parents' Names (comma separated): dudu

Animal didi added successfully.

**Updating health status to dead and find it from dead list:**

Admin Panel

1. Add Animal

2. Add Newborn Animal

3. View Animals

4. Update Animal Health Status

5. Update Animal Disease

6. View Dead Animals

7. Mark Animal Missing

8. Found Missing Animal

9. Search Animals by Name or Disease

10. Logout

Choose an option: 4

Animal ID: 2

New Health Status: dead

Animal Zara marked as dead on 2024-12-15 18:10:03.

Admin Panel

1. Add Animal

2. Add Newborn Animal

3. View Animals

4. Update Animal Health Status

5. Update Animal Disease

6. View Dead Animals

7. Mark Animal Missing

8. Found Missing Animal

9. Search Animals by Name or Disease

10. Logout

Choose an option: 6

ID: 2, Name: Zara, Species: Zebra, Date of Death: 2024-12-15 18:10:03

**Marking missing animals and back it to main list when it is found again:**

Admin Panel

1. Add Animal

2. Add Newborn Animal

3. View Animals

4. Update Animal Health Status

5. Update Animal Disease

6. View Dead Animals

7. Mark Animal Missing

8. Found Missing Animal

9. Search Animals by Name or Disease

10. Logout

Choose an option: 7

Animal ID: 3

Animal Ella marked as missing.

Admin Panel

1. Add Animal

2. Add Newborn Animal

3. View Animals

4. Update Animal Health Status

5. Update Animal Disease

6. View Dead Animals

7. Mark Animal Missing

8. Found Missing Animal

9. Search Animals by Name or Disease

10. Logout

Choose an option: 8

Animal ID: 3

Animal Ella found and moved back to main list.

**Searching by name:**

Admin Panel

1. Add Animal

2. Add Newborn Animal

3. View Animals

4. Update Animal Health Status

5. Update Animal Disease

6. View Dead Animals

7. Mark Animal Missing

8. Found Missing Animal

9. Search Animals by Name or Disease

10. Logout

Choose an option: 9

Enter name or disease to search: didi

ID: 7, Name: didi, Species: nm, Health: ok, Enclosure: cage, Birth Date: 2024-12-15 18:08:06, Disease: N/A, Parents: dudu

**Searching by disease:**

Admin Panel

1. Add Animal

2. Add Newborn Animal

3. View Animals

4. Update Animal Health Status

5. Update Animal Disease

6. View Dead Animals

7. Mark Animal Missing

8. Found Missing Animal

9. Search Animals by Name or Disease

10. Logout

Choose an option: 9

Enter name or disease to search: cold

ID: 7, Name: didi, Species: nm, Health: ok, Enclosure: cage, Birth Date: 2024-12-15 18:08:06, Disease: cold, Parents: dudu

**Log out:**

Admin Panel

1. Add Animal

2. Add Newborn Animal

3. View Animals

4. Update Animal Health Status

5. Update Animal Disease

6. View Dead Animals

7. Mark Animal Missing

8. Found Missing Animal

9. Search Animals by Name or Disease

10. Logout

Choose an option: 10

Logging out...

\*\*\* Zoo Management System \*\*\*

1. Login

2. Sign Up

3. Exit

**Chapter 4**

# Engineering Standards and Mapping

This chapter outlines the impact of the Zoo Management System on society, the environment, and sustainability, along with project management and teamwork strategies.

### 4.1 Impact on Society, Environment, and Sustainability

The Zoo Management System offers benefits such as efficient zoo operations, reduced paper usage, and long-term sustainability through scalable features.

#### 4.1.1 Impact on Life

The system improves zoo operations by enabling streamlined animal tracking, fostering better animal welfare, and enhancing decision-making through automated processes.

#### 4.1.2 Impact on Society & Environment

By digitizing zoo management, the system reduces reliance on physical paperwork, contributing to environmental conservation and promoting eco-friendly practices.

#### 4.1.3 Ethical Aspects

The system ensures data confidentiality and complies with ethical standards by securing user and animal data against unauthorized access.

#### 4.1.4 Sustainability Plan

The system is designed for scalability, supporting long-term use by allowing easy updates and the integration of advanced functionalities, ensuring adaptability as the zoo’s needs evolve.

### 4.2 Project Management and Team Work

#### Project Management

The project employed efficient management strategies, including clear task allocation and milestone tracking. Key strategies included:

* **Task Distribution:** Assigning specific roles such as system design, coding, testing, and documentation to team members.
* **Resource Utilization:** Leveraging tools like GitHub for version control and communication platforms for collaboration.
* **Progress Monitoring:** Conducting regular meetings and updates to ensure adherence to the timeline and project objectives.

#### Teamwork

Effective communication and collaboration among team members ensured the project's success. Regular feedback loops and collective problem-solving fostered a cohesive team dynamic, enabling timely completion of the project.

**Team Work**

**1. Jerin Nur Khan Achol**

**Id: 0242220005101940**

**(Log in, Sign up, Sign In,Default Value)**

* **Log in**: Used to access the system with valid credentials; verifies the identity of the user to ensure secure access.
* **Sign up**: Allows new users to create an account by entering required information, such as username and password, for first-time access.
* **Sign in**: Synonym for "Log in," used interchangeably to denote the act of accessing the system using existing credentials.
* **Default Value**: Provides pre-set data or conditions (e.g., default animals in the system) to ensure smooth functionality during initial testing or operation without requiring user input.

**2. Sohi Bilkis Binte Yer**

**Id: 0242220005101917**

**(User, Admin, Animal, ZooManagementSystem)**

* **User (Abstract Class):** Base class for all system users, with methods for authentication and user panel display.
* **Admin:** Manages animal-related functions, including adding animals, marking them as missing, updating health statuses, and handling deceased animal records.
* **Animal:** Stores and manages details about individual animals such as ID, name, species, health status, enclosure, and birth details.
* **ZooManagementSystem:** Entry point for the application, responsible for handling user login, signup, and overall system management.

**3. Sadia Shumona Shanta**

**Id: 0242220005101936**

**(Exception handling)**

* It identifies and handles invalid operations, such as duplicate animal IDs, invalid login attempts, or trying to update a non-existent animal record.
* It improves the user experience by providing clear error messages instead of abrupt terminations or crashes.

### Complex Engineering Problem

This project addresses the integration of multiple software components, including user authentication, secure data management, and ensuring scalability. It involves handling large datasets, maintaining system performance as the library grows, and ensuring secure, efficient management of user information and book transactions.

#### Mapping of Program Outcome

The project aligns with PO1 (applying engineering knowledge) and PO3 (designing complex solutions) by developing a secure, scalable system. It demonstrates the application of software engineering principles to solve practical problems, including secure user authentication, efficient data management, and creating a system that can scale as needed.

Table 4.1: Justification of Program Outcomes

|  |  |
| --- | --- |
| **PO’s** | **Justification** |
| PO1 | Demonstrate a comprehensive understanding of fundamental database management concepts, including the relational data model, normalization techniques, and SQL basics  Design, implement and optimize relational databases, incorporating advanced SQL queries, indexing techniques and query optimization strategies. |
| PO2 | Design, implement and optimize relational databases, incorporating advanced SQL queries, indexing techniques and query optimization strategies. |

* + 1. **Complex Problem Solving**

The Library Management System addresses engineering problem categories defined by EP1–EP7. The table below maps these categories with their justification.

Chapter 4. Engineering Standards and Mapping 4.3. Complex Engineering Problem

Knowledge profile and rationale thereof.

Table 4.2: Mapping with complex problem solving.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **EP1**  Dept of Knowledge | **EP2**  Range of Conflicting Requirements | **EP3**  Depth of Analysis | **EP4**  Familiarity of Issues | **EP5**  Extent of Applicable Codes | **EP6**  Extent  Of Stakeholder Involvement | **EP7**  Inter- dependence |
| The project requires knowledge in software engineering, data management, and system security. | Balancing security, user access, scalability, and system performance. | In-depth analysis of system design to ensure efficiency and security. | |  | | --- | |  |  |  | | --- | | Familiarity with common issues in user authentication and database management. | | Ensuring compliance with data security and software development standards. | Involvement of users and library staff to ensure system meets their needs. | |  | | --- | |  |  |  | | --- | | Interdependence between user authentication, data management, and system scalability. | |

* + 1. **Engineering Activities**

This section maps the project's engineering activities with categories EA1–EA5, providing rationale for each activity.

Table 4.3: Mapping with complex engineering activities.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **EA1**  Range of resources | **EA2**  Level of Interaction | **EA3**  Innovation | **EA4**  Consequences for society and  environment | **EA5**  Familiarity |
| Utilizes a variety of resources, including programming languages, frameworks, and databases. | *High interaction among team members, as well as with end-users for feedback and improvements.* | Innovative approach in integrating user authentication and scalable data management. | Improves accessibility and efficiency in library systems, with minimal environmental impact due to digitalization. | Familiarity with software development practices, security protocols, and system scalability challenges. |

**Github Link**

[**https://github.com/raduan-ahmed/PYTHON-OOP-2/blob/main/Lab%20Project/Final\_project%20of%20Library%20Management.py**](https://github.com/raduan-ahmed/PYTHON-OOP-2/blob/main/Lab%20Project/Final_project%20of%20Library%20Management.py)

**Chapter 5**

**Conclusion**

This chapter summarizes the project’s achievements and outlines future enhancements.

**5.1 Summary**  
The Zoo Management System successfully automates key zoo operations, including animal tracking, health management, and administrative tasks, improving efficiency and reducing manual errors.

**5.2 Limitation**  
Some advanced features, like predictive health analytics and integration with animal monitoring devices, were not fully implemented.

**5.3 Future Work**  
Future enhancements could include:

* Integrating real-time monitoring for animal health and behavior using IoT devices.
* Developing advanced data analytics for better decision-making and animal care insights.

# References

Online Resources