Contents

[Analysis 2](#_Toc189434376)

[Investigating the project problem 2](#_Toc189434377)

[Methods for investigating the problem 2](#_Toc189434378)

[Describing the problem to be solved 2](#_Toc189434379)

[Modelling the problem 2](#_Toc189434380)

[Describing the specific objectives of the new system 3](#_Toc189434381)

[Conversation with Client 5](#_Toc189434382)

[Consideration of possible solutions to the problem 6](#_Toc189434383)

[Prototype 7](#_Toc189434384)

[Design 22](#_Toc189434385)

[Data Model 22](#_Toc189434386)

[Entity Relationship Diagrams 22](#_Toc189434387)

[Data Dictionary 23](#_Toc189434388)

[SQL Statements 24](#_Toc189434389)

[Overall System Design 25](#_Toc189434390)

[Data Structures 26](#_Toc189434391)

[Class Diagrams 26](#_Toc189434392)

[Interface Design 27](#_Toc189434393)

[Algorithms 31](#_Toc189434394)

[Hardware and Software selection 31](#_Toc189434395)

[Security and Integrity of Data 32](#_Toc189434396)

[Technical Solution 33](#_Toc189434397)

[Testing 33](#_Toc189434398)

[Evaluation 33](#_Toc189434399)

# Analysis

## 

## Investigating the project problem

The aim of the project is to create a computer simulation to model a complex environmental problem. Currently the deer population in Scotland has grown rapidly as the deer population has no natural predators. The aim of my project will be to create a computer simulation to explore whether the reintroduction of wolves as a predator would create an equilibrium within the ecosystem that would stabilize the deer population.

### Methods for investigating the problem

To investigate the problem, I have chosen three different methodologies. The first of these will be a structured interview with a stakeholder who is a student taking a degree in environmental science. The second form of research I will use will be a survey of computer science students in my school who will be able to give me feedback on my project idea. Thirdly, I will carry out some desk-based research to explore existing solutions on rewilding animals.

## Conversation with Client

The client is a relative, currently studying environmental science at university in Scotland. As part of their degree course, my client has chosen to do their thesis on rewilding wolves in the highlands.

The table underneath is a record of an interview that I carried out with the client.

|  |  |
| --- | --- |
| Client | Me |
| For the foxes and rabbits simulation, are you planning to include factors like food availability or environmental conditions, like seasons, in the model? And how are you approaching the interaction between the predators and prey—are you going with simple predation rates, or do you have more complex behaviors in mind? | Seasons and environmental conditions are rather out of the scope of this project, however food availability may be an interesting problem to tackle, though i will only be doing that if I end up with enough time to work on that, as I do not see that as completely necessary to the project. The interaction between predators and prey will be quite simple, it will be simulated in real time, where if a prey gets within a specific range of a predator, the predator will start chasing the prey |
| I see! Keeping the interaction simple like that makes sense, especially if you're aiming for a straightforward model. Do you plan to include any population dynamics, like reproduction rates for the foxes and rabbits, or will the population sizes remain static throughout the simulation? Also, will there be a limit on how many prey a predator can catch before it needs to "rest" or something similar, or will they just keep chasing as long as they’re in range? | I do plan on introducing reproduction for both prey and predators. All animals will have their own 'energy' which will decrease as they move around. Each animal will have different maximum energies. Predators can regain energy from consuming prey, and for now, prey will regain energy through eating grass (which will likely be modelled as simply randomly regaining energy), however if i implement food availability, the prey will gain energy through eating that food |
| That’s an interesting system! I like how energy management ties into the behavior of the animals. For the reproduction aspect, will animals reproduce once they reach a certain energy threshold or age? And will there be a limit to how many offspring they can have at once, or will it be more of a probabilistic process depending on their energy or other factors? | if possible, i am going to implement a system where if two animals of the same species interact/go onto the same grid space, and if they both have sufficient energy, offspring will be produced. Each species will have its own average offspring number, based on real life rates |
| Your approach to reproduction and energy management sounds really well thought out and realistic. It’ll be interesting to see how the balance between predators and prey plays out with those mechanics in place! Thanks for answering my questions! |  |

## Student Survey

The second part of my research was to carry out a survey with students in my computer science A-Level class. They will be able to provide feedback on my analysis of the problem and my proposed solution. I have built the questionnaire using Microsoft Forms so that I can distribute and analyse it more efficiently.

The table below shows the questions that I will have in my survey, in which I will ask people to give a rating out of 10 for their response.

|  |  |
| --- | --- |
| Questions | Responses |
| 1. Give a rating out of 10 based on your current understanding of the consequences of the growth of the deer population in Scotland |  |
| 2. Give a rating out of 10 for how informative it would be for people to run a simulation of population change as a result of rewilding |  |
| 3. Give a rating out of 10 for how useful it would be for the user to be able to input a starting size for each population |  |
| 4. Give a rating out of 10 for how useful it would be for the user to be able to input a number of steps to show population change over time |  |
| 5. Give a rating out of 10 for how useful users would find it to show a variety of different graphs of the final populations |  |
| 6. Please suggest any other features that you feel should be included in the simulation |  |

## Desk-Based Research

The final part of my initial research was to carry out some desk-based research by looking at websites that contain information about rewilding and rewilding simulations. The purpose of carrying out desk-based research is to gain insight into the topic which may help with my project.

## Research Conclusions

Having carried out my research, which included a client interview, a student survey, and desk-based research, I have come to the following conclusions about the main requirements of my own project:

|  |  |  |
| --- | --- | --- |
| Number | Requirement | Explanation |
| 1 | Individual Accounts |  |
| 2 | Simulation Visualization |  |
| 3 | User input parameters for environment setup |  |
| 4 | Multiple types of simulation |  |
| 5 | Collect data every time the simulation is run, in order to find a population equilibrium |  |

## Describing the problem to be solved

Background information about the client

Overview of the problem and explanation of why it needs a computational solution

Consider the limitations and constraints of the solution

## Problems faced by existing system

## Constraints and limitations

## Modelling the problem

To model the problem, I will use the following tools:

* IPSO Chart to give an overview of the problem
* Context Diagram to give an overview of the system

## IPSO Chart

The purpose of the IPSO Chart is to analyse the current problem in the highlands of Scotland, where the deer population has no natural predators.

|  |  |
| --- | --- |
| **Input**   * According to nature.scot, there are 400,000 deer living on open ground, and a further 100,000 living in woodland areas * There are no natural predators to deer currently in the highlands * Around 10,000 deer are killed every year through licensed hunting on estates. * Around 100,000 deer a year are culled | **Storage**   * Most deer population live in remote areas * As populations increase, the deer are encroaching on urban areas, becoming more of a risk on highways |
| **Processing**   * Deer are breeding more quickly due to lack of natural predators * Population has doubled since 1990 | **Output**   * There are around 9,000 deer/car collisions per year * Up to 150,000,000 young trees are vulnerable to damage from deer |

## Describing the specific objectives of the new system

|  |  |  |
| --- | --- | --- |
| Number | Objective | Performance Criteria |
| 1 | The user will be able to run a simulation of either the deer/wolf ecosystem or the rabbit/fox ecosystem | In the simulation screen, the user can do all of the following:   1. When user presses ‘Load description’ button, updates panel to show description 2. When user presses ‘Load Fox/Rabbit sim’ button, opens input screen for that simulation 3. When user presses ‘Load Wolf/Deer sim’ button, opens input screen for that simulation 4. When user presses statistics button, opens stats screen, passing the user id 5. When user presses quit button, application closes |
| 2 | The user will be able to input parameters for the fox/rabbit simulation | 1. User can input initial populations for rabbits as an integer 2. Input is **validated** to prevent erroneous data 3. User can input initial populations for foxes as an integer 4. Input is **validated** to prevent erroneous data 5. User can input number of steps to show how the populations change over time as an integer 6. Input is **validated** to prevent erroneous data 7. User can input the grid size to control the size of the landscape as an integer 8. Input is **validated** to prevent erroneous data |
| 3 | The user will be able to input parameters for the wolf/deer simulation | 1. User can input initial populations for deer as an integer 2. Input is **validated** to prevent erroneous data 3. User can input initial populations for wolves as an integer 4. Input is **validated** to prevent erroneous data 5. User can input number of steps to show how the populations change over time as an integer 6. Input is **validated** to prevent erroneous data 7. User can input the grid size to control the size of the landscape as an integer 8. Input is **validated** to prevent erroneous data |
| 4 | The user will be able to generate a visualization of the different fox and rabbit populations | When user launches simulation, the following things will occur:   1. Plot will appear on screen 2. Labelled grid will display, matching user input for grid size 3. Animation runs for each step that user has input 4. Blue and red data points on the graph representing rabbits and foxes respectively 5. Total population for each species displayed on graph 6. Loads population/time graph for both species 7. Loads graph of populations plotted against each other 8. Data from simulation gets inserted into ‘Game’ table, with simulation type |
| 5 | The user will be able to generate a visualization of the different wolf and deer populations | When user launches simulation, the following things will occur:   1. Plot will appear on screen 2. Labelled grid will display, matching user input for grid size 3. Animation runs for each step that user has input 4. Blue and red data points on the graph representing deer and wolves respectively 5. Total population for each species displayed on graph 6. Loads population/time graph for both species 7. Loads graph of populations plotted against each other 8. Data from simulation gets inserted into ‘Game’ table, with simulation type |
| 6 | To use the simulation, all users must create an account  All user data will be stored in a secure database, using hashed passwords | 1. User clicks on register button in main menu 2. User must enter first name with at least one character, or a presence check will force them to enter again 3. User must enter last name with at least one character, or a presence check will force them to enter again 4. User must enter an email address with domain “@sherborne.org” or a format check will prompt them to enter again 5. User must enter password that meets following criteria, or else will be forced to enter again:  * 8+ Characters * Uppercase and lowercase characters * At least 1 number  1. User must re-enter password that matches. If not matching, user will be prompted to input again. 2. Passwords are saved using sha256 hashing with a salt, and saved as a hexdigest 3. All user data will be stored in a user table in a remote SQL database 4. Once registration is complete, the registration screen closes, the main menu re-opens for the user to log in 5. To log in, the user must enter their email. If this is not found in the database, user is told 6. The user must enter the password corresponding to that email. If wrong password, user is told that they do not match |
| 7 | Users find the simulation easy to navigate and engaging and informative | Get a group of 10 users to test my system and rate these key features:   1. Navigability – At least 70% 2. Ease of use – At least 70% 3. Engagement with the main concept of running population simulations – At least 70% 4. Greater understanding of issues relating to rewilding – At least 70% 5. Users can suggest improvements, which I will then consider in my evaluation as areas for development |
| 8 | The system will be scalable so that extra features can be added easily | 1. All different components of the system will be coded as separate modules 2. All components will be coded as classes    1. Database    2. Startup Screen    3. Login Screen    4. Register Screen    5. Simulation Screen |

## Potential solutions

## Consideration of possible solutions to the problem

To develop my solution, I will be using Python 3.11 for a variety of reasons:

* It has many libraries which will be essential for different parts of my program, such as:
  + SQLite3 for the database
  + Tkinter for the GUI
  + Numpy and Matplotlib for visualizing the simulation n
  + re for creating a regular expression
* Python allows me to modularize my code, as I can create my own modules which can be imported into other programs
* Python is an Object-Oriented language, and I will be using classes

Also to develop my solution, I will be using pycharm as my IDE:

* It has development tools such as autocorrect
* It has Debugging tools such as syntax highlighting and error messages
* It has a package installer
* It has a directory view, to see all the files in my project

# Prototype

Here I have used classes to create a database. I used classes, as this will enable me to instantiate the database in different programs. In lines 11 to 17, I have used a DDL script to create a database table

A screenshot of a computer program

Description automatically generatedA close-up of a computer screen

Description automatically generated

Here is evidence of the registration program, the output, and the data that it collectsA screen shot of a computer program

Description automatically generatedA screen shot of a computer program

Description automatically generatedA screen shot of a computer program

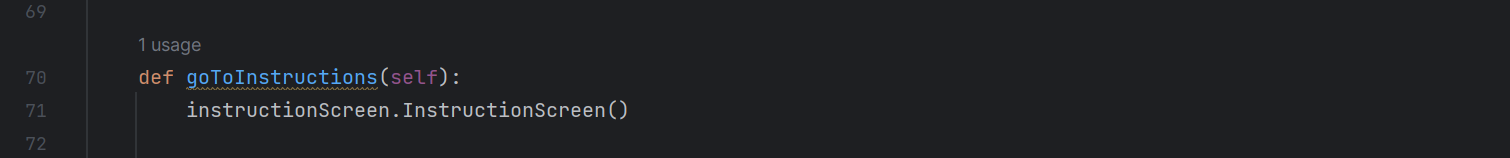
Description automatically generatedA screenshot of a computer screen

Description automatically generatedA screenshot of a computer

Description automatically generated

This is the login program, as well as its output A computer screen shot of a program

Description automatically generatedA screen shot of a computer program

Description automatically generatedA screenshot of a computer login form

Description automatically generated

This is the startup menu program, as well as its outputA screen shot of a computer program

Description automatically generatedA screenshot of a computer screen

Description automatically generated

A screen shot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

NEED TO ADD ANOTHER BUTTON TO MOVE TO SIMULATION SCREEN

This is the input screen program, where the parameters for the simulation are inputtedA screen shot of a computer program

Description automatically generatedA screen shot of a computer code

Description automatically generatedA screenshot of a computer

Description automatically generated

This is the simulation program (WIP):

TODO:

1. Add text to instruction page
2. Develop the simulation program
3. **IMPORTANT:** Develop the database so there are 2-3 tables, and I can run queries
   1. ~~Update DDL script so that it creates Game and Gameplay tables~~
   2. ~~When simulations run, need to have subroutine which stores final data values in gameplay table~~
4. More formatting of the landscape
5. **IMPORTANT:** Create button on simulation information window which loads statistics page with two options:
   1. ~~personal statistics~~
   2. ~~global statistics~~

# Design

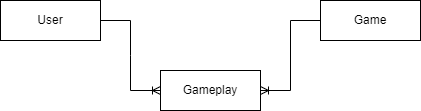
## Data Model

### Entity Relationship Diagrams

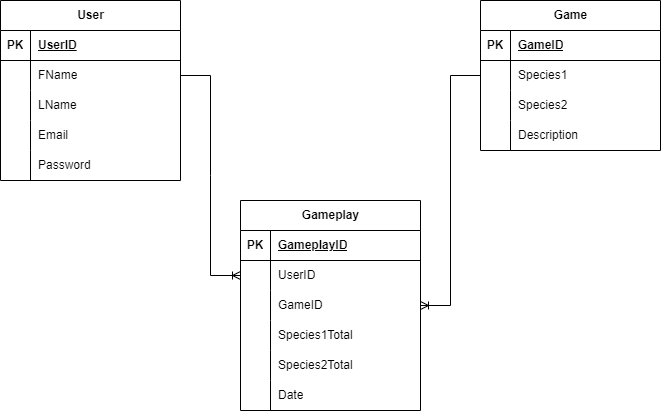
The ERD below is pre-normalised and shows a many-to-many relationship between user and game.



The ERD below shows a normalised version where there is a link table ‘Gameplay’ which has one-to-many relationships with User and Game.



The figure below shows a fully normalised entity-attribute model, with all the fields for each table.



### Data Dictionary

The table below shows all of the attributes in detail for the User entity.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | User | | |
| Primary Key | UserID | | |
| Foreign Keys |  | | |
| Field | Data Type | Validation | Sample Data |
| UserID | Auto-Increment | Not Null | 1 |
| Fname | String | Not Null | John |
| Lname | String | Not Null | Smith |
| Email | String | Not Null, Format Check, ‘@sherborne.org’ | xxxxxxxxx@sherborne.org |
| Password | String | Not Null, 8 Characters Min, 1+ Upper and Lower, 1+ Number | 64 character string |

The table below shows all of the attributes in detail for the Game entity.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Game | | |
| Primary Key | GameID | | |
| Foreign Keys |  | | |
| Field | Data Type | Validation | Sample Data |
| GameID | Auto-Increment | Not Null | 1 |
| Species1 | String | Not Null | Wolf |
| Species2 | String | Not Null | Deer |
| Description | String | Not Null | This is a simulation that uses algorithms to trace the evolution of two species. |

The table below shows all of the attributes in detail for the Gameplay entity.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Gameplay | | |
| Primary Key | GameplayID | | |
| Foreign Keys | UserID, GameID | | |
| Field | Data Type | Validation | Sample Data |
| GameplayID | Auto-Increment | Not Null | 1 |
| UserID | Integer | Not Null | 3 |
| GameID | Integer | Not Null | 7 |
| Species1Total | Integer | Not Null | 106 |
| Species2Total | Integer | Not Null | 238 |
| Date | Date | Not Null | 11/10/2024 |

### SQL Statements

|  |  |  |
| --- | --- | --- |
| Statement | Script | Description |
| DDL Script | CREATE TABLE IF NOT EXISTS User(  UserID INTEGER PRIMARY KEY,  FName TEXT NOT NULL,  LName TEXT NOT NULL,  Email Varchar(320) NOT NULL,  Password TEXT NOT NULL) |  |
| SQL Script | INSERT INTO User(FName, LName, Email, Password) VALUES(?, ?, ?, ?) |  |
| SQL Script | SELECT Email, Password FROM User WHERE email=? |  |
| SQL Script | INSERT INTO Gameplay(UserID, GameID, Species1Total, Species2Total, Date) VALUES(?, ?, ?, ?, ?) |  |
| SQL Script | SELECT Species1, AVG(Species1Total),  Species2, AVG(Species2Total)  FROM Gameplay, Game  WHERE GameID=?  AND UserID=?  AND Gamep  lay.GameID = Game.GameID | Returns Average Population Size of each species for all of the simulations that a specified user has run. |
|  | SELECT Species1, AVG(Species1Total),  Species2, AVG(Species2Total)  FROM Gameplay, Game  WHERE GameID=?  AND Gameplay.GameID = Game.GameID | Returns Average Population Size of each species for all simulations that have been run globally. This can be used to measure the relative accuracy of user simulations. |

## Overall System Design

Hierarchy diagram of system

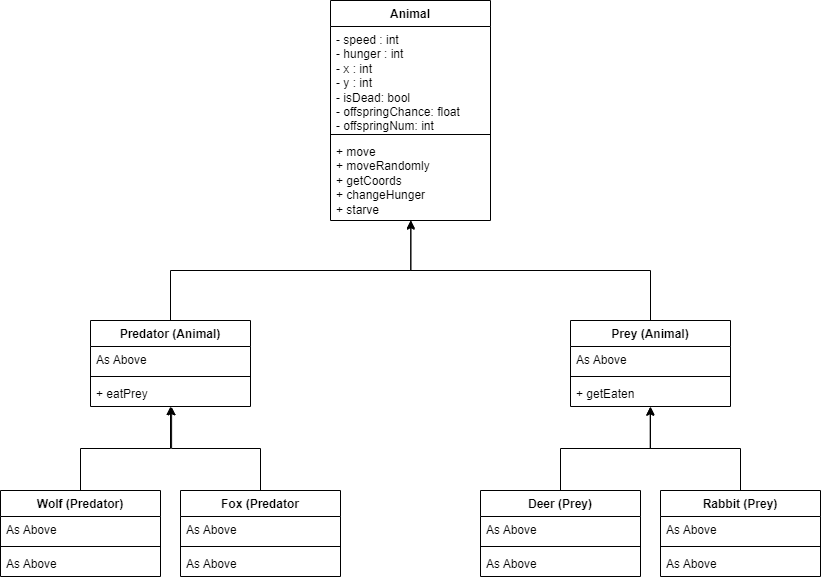
The green box represents the entire system, the blue boxes represent the subsystems that make up the overall system, and the red boxes represent the algorithms used in each subsystem.

In the technical solution, I will show how I have built each of the algorithms in each subsystem.HA screenshot of a computer screen

Description automatically generated

## Data Structures

## Class Diagrams



This is the class diagram for the Animal class and its subclasses. Animal is an abstract class, as are its subclasses, Predator and Prey. Wolf and Fox are subclasses of Predator, and Deer and Rabbit are subclasses of Prey.

## Interface Design

Interface for all of the menus

|  |  |
| --- | --- |
| Screen | Explanation |
| A screenshot of a computer screen  Description automatically generated | Main Menu Screen:   * Loads on program startup * Label to give user guidance on what to do * Log in button   + Opens Login screen   + Closes Main Menu screen * Register button   + Opens Registration screen   + Main menu screen stays open |
| A screenshot of a computer screen  Description automatically generated | Registration Screen:   * Screen opens from main menu screen * Label to give user guidance on what to do * Series of Labels and Entry boxes to instruct user what to input * Entry boxes have validation * Submit Button   + Performs validation checks on necessary fields   + Message box popup if validation rules broken, tells user which rule broken   + Inserts inputted data into user table   + Loads next screen   + Closes this screen |
| A screenshot of a computer login form  Description automatically generated | Login Screen:   * Screen opens from main menu screen * Label to give user guidance on what to do * Labels paired with Entry boxes to instruct users on what to input * Submit button   + Run presence check on inputted email and password in the user table   + Onscreen message displayed if presence check fails   + Loads next screen   + Closes this screen |
|  | Simulation Instruction screen:   * Opens from login screen * Label to inform user that they are in simulation menu * When Description or Simulation button pressed, update the large label on the left with corresponding text * Fox/Rabbit Sim button pressed   + Loads setup screen of Fox/Rabbit simulation   + Closes this screen * Wolf/Deer Sim button pressed   + Loads setup screen of Wolf/Deer simulation   + Closes this screen * Quit button closes program |
| A screenshot of a computer  Description automatically generated | Fox/Rabbit Simulation parameter screen:   * Opens from Simulation Instruction screen * Label to guide user on what to do * Labels paired with entry boxes to instruct users on what to input * All fields validated as integers * Submit button   + Runs validation check   + Message box popup if validation rules broken   + Passes inputted variables to simulation screen   + Runs simulation screen   + Closes this screen   **WOLF/DEER SCREEN WORKS THE SAME** |
|  | Simulation Screens:   * Parameters for demo simulation on the left:   + Starting Rabbits: 10   + Starting Foxes: 2   + Grid size: 10   + Steps: 50 * Ecosystem Screen   + Animated/Shows iterations   + User can watch simulation as it progresses   + Coloured points represent each species * Populations/Time graph   + Opens after Ecosystem screen   + Shows change in populations of both species over the course of the simulation * Fox/Rabbit Populations graph   + Opens after Pop/Time graph   + Shows change in Fox pop against change in Rabbit pop   + Helps show equilibrium points   **WOLF/DEER OUTPUT WORKS THE SAME** |

## 

## Algorithms

In the table below, I will show designs for all the algorithms needed in my system. I will present all of my algorithms as flow charts designed in UML

|  |  |
| --- | --- |
| Algorithm Design | Explanation/Justification |
|  |  |
|  |  |
|  |  |
|  |  |

## Hardware and Software selection

In the table below I will summarize the hardware and software needed to develop and run my system

|  |  |  |
| --- | --- | --- |
| Feature | Hardware/Software | Specification/Explaination |
| Sqlite Studio | Software | Version 3.4.11 / This is needed to view the database tables |
| Pycharm | Software | Community Version For Windows 24.3.1.1 / Needed for development and maintainance |
| Operating System | Software | Windows 10 or higher (64bit) |
| Processor | Hardware | 1GHz or faster |
| RAM | Hardware | 2Gb or higher |
| Secondary Storage | Hardware | 20Gb or higher |
| QWERTY Keyboard | Hardware | USB2 or higher / Input for registration, login, and simulation |
| Two-Button Mouse | Hardware | USB2 or higher / Need to be able to click on buttons to navigate between screens |
| Monitor | Hardware | 800x600 minimum / Need to be able to display outputs |

## 

## Security and Integrity of Data

There are several features of my solution that will enable me to ensure the security and integrity of data

|  |  |
| --- | --- |
| Feature | Explanation |
| Hashing user passwords | One way encryption using Sha256 so that even if database is breached, information is meaningless |
| Validation | Registation Screen  Format check and uniqueness check on email, Format and length check on password. Presence check on all fields.  Login Screen  Presence check  Input Screens  These are inputs for initial population numbers for each species, Grid size, and Number of steps.  Type checks for integers |
| Defensive programming | Throughout the system, the user uses buttons to navigate between screens. |
| External Database File | If program crashes, it doesn’t affect the data in the database |
| Use of encapsulation | Private attributes for animal classes so that they do not get overwritten |

# Technical Solution

# Testing

The table below shows all of the functional black box tests that I will use in my system. I will categorize the result of each test as met, partially met, or not met, and include a time stamp that corresponds to where these tests are evidenced in my video.

However, for the tests linked to objective 7, I will use a survey to gather user feedback.

|  |  |  |  |
| --- | --- | --- | --- |
| Number | Objective | Success Criteria | Met/Partially/Not Met |
| 1 | The user will be able to run a simulation of either the deer/wolf ecosystem or the rabbit/fox ecosystem | In the simulation screen, the user can do all of the following:   1. When user presses ‘Load description’ button, updates panel to show description 2. When user presses ‘Load Fox/Rabbit sim’ button, opens input screen for that simulation 3. When user presses ‘Load Wolf/Deer sim’ button, opens input screen for that simulation 4. When user presses statistics button, opens stats screen, passing the user id 5. When user presses quit button, application closes |  |
| 2 | The user will be able to input parameters for the fox/rabbit simulation | 1. User can input initial populations for rabbits as an integer   2. Input is **validated** to prevent erroneous data   1. User can input initial populations for foxes as an integer 2. Input is **validated** to prevent erroneous data 3. User can input number of steps to show how the populations change over time as an integer 4. Input is **validated** to prevent erroneous data 5. User can input the grid size to control the size of the landscape as an integer 6. Input is **validated** to prevent erroneous data |  |
| 3 | The user will be able to input parameters for the wolf/deer simulation | 1. User can input initial populations for deer as an integer 2. Input is **validated** to prevent erroneous data 3. User can input initial populations for wolves as an integer 4. Input is **validated** to prevent erroneous data 5. User can input number of steps to show how the populations change over time as an integer 6. Input is **validated** to prevent erroneous data 7. User can input the grid size to control the size of the landscape as an integer 8. Input is **validated** to prevent erroneous data |  |
| 4 | The user will be able to generate a visualization of the different fox and rabbit populations | When user launches simulation, the following things will occur:   1. Plot will appear on screen 2. Labelled grid will display, matching user input for grid size 3. Animation runs for each step that user has input 4. Blue and red data points on the graph representing rabbits and foxes respectively 5. Total population for each species displayed on graph 6. Loads population/time graph for both species 7. Loads graph of populations plotted against each other 8. Data from simulation gets inserted into ‘Game’ table, with simulation type |  |
| 5 | The user will be able to generate a visualization of the different wolf and deer populations | When user launches simulation, the following things will occur:   1. Plot will appear on screen 2. Labelled grid will display, matching user input for grid size 3. Animation runs for each step that user has input 4. Blue and red data points on the graph representing deer and wolves respectively 5. Total population for each species displayed on graph 6. Loads population/time graph for both species 7. Loads graph of populations plotted against each other 8. Data from simulation gets inserted into ‘Game’ table, with simulation type |  |
| 6 | To use the simulation, all users must create an account  All user data will be stored in a secure database, using hashed passwords | 1. User clicks on register button in main menu 2. User must enter first name with at least one character, or a presence check will force them to enter again 3. User must enter last name with at least one character, or a presence check will force them to enter again 4. User must enter an email address with domain “@sherborne.org” or a format check will prompt them to enter again 5. User must enter password that meets following criteria, or else will be forced to enter again:  * 8+ Characters * Uppercase and lowercase characters * At least 1 number  1. User must re-enter password that matches. If not matching, user will be prompted to input again. 2. Passwords are saved using sha256 hashing with a salt, and saved as a hexdigest 3. All user data will be stored in a user table in a remote SQL database 4. Once registration is complete, the registration screen closes, the main menu re-opens for the user to log in 5. To log in, the user must enter their email. If this is not found in the database, user is told 6. The user must enter the password corresponding to that email. If wrong password, user is told that they do not match |  |
| 7 | Users find the simulation easy to navigate and engaging and informative | Get a group of 10 users to test my system and rate these key features:   1. Navigability – At least 70% 2. Ease of use – At least 70% 3. Engagement with the main concept of running population simulations – At least 70% 4. Greater understanding of issues relating to rewilding – At least 70% 5. Users can suggest improvements, which I will then consider in my evaluation as areas for development | See user survey results below |
| 8 | The system will be scalable so that extra features can be added easily | 1. All different components of the system will be coded as separate modules 2. All components will be coded as classes    1. Database    2. Startup Screen    3. Login Screen    4. Register Screen    5. Simulation Screen |  |

## User Survey Results

# Evaluation