Software Requirements Specification for SFWRENG 4G06: subtitle describing software

Team 28, Cowvolution Minds
Aryan Patel
Harshpreet Chinjer
Krish Patel
Martin Ivanov
Shazim Rahman

October 10, 2024

Contents

Revision History

Date	Version	Notes
October 6, 2024	1.0	Initial SRS document

1 Purpose of the Project

1.1 User Business

Insert your content here.

1.2 Goals of the Project

Insert your content here.

2 Stakeholders

2.1 Client

Insert your content here.

2.2 Customer

Insert your content here.

2.3 Other Stakeholders

Insert your content here.

2.4 Hands-On Users of the Project

Insert your content here.

2.5 Personas

Insert your content here.

2.6 Priorities Assigned to Users

Insert your content here.

2.7 User Participation

Insert your content here.

2.8 Maintenance Users and Service Technicians

Insert your content here.

3 Mandated Constraints

3.1 Solution Constraints

Insert your content here.

3.2 Implementation Environment of the Current System

Insert your content here.

3.3 Partner or Collaborative Applications

Insert your content here.

3.4 Off-the-Shelf Software

Insert your content here.

3.5 Anticipated Workplace Environment

Insert your content here.

3.6 Schedule Constraints

Insert your content here.

3.7 Budget Constraints

Insert your content here.

3.8 Enterprise Constraints

Insert your content here.

4 Naming Conventions and Terminology

4.1 Glossary of All Terms, Including Acronyms, Used by Stakeholders involved in the Project

Insert your content here.

5 Relevant Facts And Assumptions

5.1 Relevant Facts

Insert your content here.

5.2 Business Rules

Insert your content here.

5.3 Assumptions

Insert your content here.

6 The Scope of the Work

6.1 The Current Situation

The current state of dairy farming presents challenges in efficiently predicting the health, productivity, and breeding outcomes of cattle. Farmers typically rely on historical records, but the analysis is done manually, often leading to reactive management. The existing systems do not utilize advanced technologies such as machine learning for predictive analytics. As a result, there is limited proactive management regarding milk production, breeding success, and herd longevity, which directly impacts farm profitability and sustainability.

The current solution environment lacks integration of large datasets from multiple sources, such as individual cow health records, breeding history, and environmental conditions, into a single system that can offer actionable predictions.

6.2 The Context of the Work

This project aims to develop a machine learning model that will leverage historical herd data to predict important traits such as milk yield, breeding success rates, and the likelihood of a cow leaving the herd. This model will be integrated into a farm management system, providing farmers with actionable insights. The goal is to move from reactive to proactive herd management.

The model will use data such as the health, breed, and genetic history of both the mother and father to predict traits in calves. The software will be developed as part of a partnership with CATTLEytics Inc., ensuring seamless integration into their existing platform used by dairy farmers.

6.3 Work Partitioning

The project will be divided into several key components:

1. **Data Collection and Preprocessing:**
    - Collection of historical data from existing systems, including cow health records, breeding data, and productivity metrics.
    - Cleaning and standardizing the data for input into the machine learning model.
2. **Model Development:**
    - Designing and implementing the machine learning model for trait prediction (e.g., milk production, herd retention).
    - Training and validating the model on historical datasets.
    - Iterative testing and refinement.
3. **Integration:**

- Integrating the prediction model into the CATTLEytics Inc. farm management system.
- Ensuring that outputs are presented in a user-friendly format for farmers to make decisions.

4. **Testing and Validation:**
    - Testing the software in real-world farm environments to validate predictions and refine the user interface.
5. **Documentation and Training:**
    - Providing clear documentation for users and training for farmers to effectively use the system.

6.4 Specifying a Business Use Case (BUC)

Title: Predicting Cow Traits for Optimized Herd Management
Primary Actor: Dairy farmer using the CATTLEytics Inc. system.
Precondition: The farmer has access to a herd management system integrated with the prediction model. Historical data on breeding, milk production, and herd turnover are available.
Trigger: The farmer initiates the model to predict the outcomes of a planned breeding or evaluates the likelihood of an existing cow leaving the herd.
Main Success Scenario:

1. The farmer selects cows for breeding and inputs the necessary data (e.g., parent traits).
2. The system processes the input and returns predictions for milk yield and herd retention likelihood.
3. Based on the model's predictions, the farmer makes informed decisions on breeding strategies or management actions to prevent herd loss.

Postconditions:

- The farmer has actionable insights to improve herd productivity and manage herd turnover proactively.

7 Business Data Model and Data Dictionary

7.1 Business Data Model

This section will be completed once the relevant data model details are available.

7.2 Data Dictionary

This section will be completed once the relevant data model details are available.

8 The Scope of the Product

8.1 Product Boundary

The product's primary function is to predict cow traits such as milk production, breeding success, and herd retention based on parental data. The machine learning model will integrate into a platform like CATTLEytics Inc., and its boundaries will include:

- **Included**: The product will use historical data to generate predictive insights on milk production, breeding success rates, and herd retention likelihood. Farmers will be able to input relevant data to receive predictions.
- **Not Included**: Real-time data collection and analysis, live health monitoring, or any complex integrations with external devices (e.g., IoT sensors).

This product will focus solely on predictive analytics based on historical data and will not handle aspects like external raw data collection or advanced herd management automation beyond providing insights.

8.2 Product Use Case Table

This table outlines the core use cases currently identified for the product.

ID	Title	Actor	Description
PUC1	Predict Breeding Success	Dairy Farmer	Farmers input breeding data to get predictions on the likelihood of successful breeding.
PUC2	Forecast Milk Production	Dairy Farmer	Farmers input cow data to get a prediction of future milk production.
PUC3	Predict Herd Retention Likelihood	Dairy Farmer	Farmers receive predictions on whether cows are likely to stay in or leave the herd.

8.3 Individual Product Use Cases (PUC's)

8.3.1 PUC1: Predict Breeding Success

- **Primary Actor**: Dairy Farmer
- **Preconditions**: Farmer has historical data about cow and parental traits available for input.
- **Trigger**: The farmer initiates a request to predict breeding success.
- **Main Success Scenario**:

1. The farmer enters relevant breeding data.
2. The system processes the input using historical records.
3. A prediction is generated on the likelihood of breeding success.

- **Postcondition**: The farmer gets an actionable prediction to decide whether to proceed with the breeding.

8.3.2 PUC2: Forecast Milk Production

- **Primary Actor**: Dairy Farmer
- **Preconditions**: Historical data for milk production and parental traits is available for input.
- **Trigger**: The farmer requests a prediction for future milk production.
- **Main Success Scenario**:

1. The farmer inputs the cow's data.
2. The system processes the input data.
3. A prediction on future milk production is generated.

- **Postcondition**: The farmer receives a prediction that helps in planning milk yield expectations.

8.3.3 PUC3: Predict Herd Retention Likelihood

- **Primary Actor**: Dairy Farmer
- **Preconditions**: Health and productivity data is available for the cows in question.
- **Trigger**: The farmer requests predictions on herd retention likelihood.
- **Main Success Scenario**:

1. The farmer selects a cow or group of cows for analysis.
2. The system processes the available data.
3. A prediction is generated on whether the cows are likely to stay in or leave the herd.

- **Postcondition**: The farmer receives predictions to assist in managing herd turnover.

8.3.4 PUC1: Predict Breeding Success

- **Primary Actor**: Dairy Farmer
- **Preconditions**: Farmer has historical data about cow and parental traits available for input.
- **Trigger**: The farmer initiates a request to predict breeding success.
- **Main Success Scenario**:
    1. The farmer enters relevant breeding data.
    2. The system processes the input using historical records.
    3. A prediction is generated on the likelihood of breeding success.
- **Postcondition**: The farmer gets an actionable prediction to decide whether to proceed with the breeding.

8.3.5 PUC2: Forecast Milk Production

- **Primary Actor**: Dairy Farmer
- **Preconditions**: Historical data for milk production and parental traits is available for input.
- **Trigger**: The farmer requests a prediction for future milk production.
- **Main Success Scenario**:
    1. The farmer inputs the cow's data.
    2. The system processes the input data.
    3. A prediction on future milk production is generated.
- **Postcondition**: The farmer receives a prediction that helps in planning milk yield expectations.

8.3.6 PUC3: Predict Herd Retention Likelihood

- **Primary Actor**: Dairy Farmer
- **Preconditions**: Health and productivity data is available for the cows in question.
- **Trigger**: The farmer requests predictions on herd retention likelihood.
- **Main Success Scenario**:
  1. The farmer selects a cow or group of cows for analysis.
  2. The system processes the available data.
  3. A prediction is generated on whether the cows are likely to stay in or leave the herd.
- **Postcondition**: The farmer receives predictions to assist in managing herd turnover.

9 Functional Requirements

9.1 Functional Requirements

FR1: Predict Breeding Success

- **Description**: The system shall predict the likelihood of a successful breeding event between two cows based on input data regarding parental traits and historical breeding records.
- **Rationale**: This feature will help farmers make more informed breeding decisions, improving breeding efficiency and reducing failures.
- **Fit Criterion**: The system will output a probability of breeding success based on parental data, and this probability must be verified by comparing predicted outcomes with actual breeding success over time.

FR2: Forecast Milk Production

- **Description**: The system shall forecast the milk production of a cow based on historical milk yield and parental traits.

- **Rationale**: Accurate predictions of future milk yield will enable farmers to better plan for production and make decisions on herd management.
- **Fit Criterion**: The forecasted milk production must be within 10% (to be determined) accuracy when compared to actual milk yield over a specified period.

FR3: Predict Herd Retention Likelihood

- **Description**: The system shall predict the likelihood of cows leaving the herd based on health records, productivity, and other historical data.
- **Rationale**: This feature will enable farmers to proactively manage their herd, reducing unexpected departures and improving herd stability.
- **Fit Criterion**: The system will provide a prediction score (e.g., high, medium, low) for herd retention, which can be evaluated by tracking actual herd retention over a six-month period.

FR4: Data Input for Predictions

- **Description**: The system shall allow the farmer to input relevant data, such as breeding records, milk production history, and health records, into the prediction model.
- **Rationale**: To generate accurate predictions, the system requires access to a range of historical data that can be inputted by the user.
- **Fit Criterion**: The input form must successfully accept and validate required fields for at least 95% of user inputs, with clear error handling for missing or incorrect data.

FR5: Report Generation

- **Description**: The system shall generate a report summarizing predictions for breeding success, milk production, and herd retention for selected cows.

- **Rationale**: Farmers need a consolidated report that provides actionable insights based on the predictions generated by the system.
- **Fit Criterion**: The system will generate reports that can be exported to a PDF format and include all requested predictions in a structured layout.

FR6: User Access Control

- **Description**: The system shall provide secure login and role-based access control, ensuring that only authorized users can access or modify the prediction data.
- **Rationale**: Farm management data is sensitive and should only be accessible by authorized personnel.
- **Fit Criterion**: The system must enforce unique login credentials for each user and restrict access based on roles (e.g., farmer, supervisor), with at least 99% reliability in access control enforcement.

FR7: Integration with Farm Management System

- **Description**: The system shall be designed to integrate with existing farm management platforms, such as CATTLEytics Inc, allowing seamless data exchange.
- **Rationale**: Integration with existing platforms will enable the system to leverage historical data and provide predictions without requiring manual data entry.
- **Fit Criterion**: The system should successfully exchange data with the farm management platform 90% of the time during testing, without errors in data transmission.

9.2 Formal Specification

Specification 1: Breeding Success Prediction

- **Description**: The system must be able to predict the likelihood of breeding success between two cows based on historical data, such as parental traits and previous breeding records.

- **Formal Specification**:
  Let X represent a breeding event.
  Let Y represent the set of all possible breeding events.
  Let P represent the predicted probability of success.

$$\forall X \in Y : \text{Prediction}(X) \rightarrow P \in [0, 1]$$

  The system shall compute the probability P for each breeding event X.

Specification 2: Milk Production Forecast

- **Description**: The system shall forecast future milk production for a given cow based on historical data of both the cow and its parents.

- **Formal Specification**:
  Let C represent a cow in the herd.
  Let Y represent historical milk production data.

$$\forall C : \text{Forecast}(C, Y) \rightarrow \text{PredictedMilkProduction}(C)$$

  The system shall provide a forecast of future milk production for each cow C based on input data Y.

Specification 3: Herd Retention Likelihood

- **Description**: The system must predict the likelihood of a cow staying within or leaving the herd, based on its health, productivity, and historical data.

- **Formal Specification**:
  Let H represent a cow's health record.
  Let P represent the predicted probability of retention.

$$\forall H : \text{RetentionPrediction}(H) \rightarrow P \in [0, 1]$$

  The system shall compute the retention probability P for each cow based on its health records and other historical data.

Specification 4: Data Input Validation

- **Description**: The system must validate the input data for cows and breeding events to ensure it is accurate and complete before generating predictions.
- **Formal Specification**: Let D represent the input data for a cow or breeding event.

  $$\forall D : \text{InputValid}(D) = \begin{cases} \text{True} & \text{if data passes validation checks} \\ \text{False} & \text{otherwise} \end{cases}$$

  The system must ensure that all data D is valid before processing it for predictions.

Specification 5: Report Generation (TBD)

- **Description**: The system must generate a report summarizing predictions for breeding success, milk production, and herd retention likelihood.
- **Formal Specification**:
  Let R represent the report generated.

  $$\forall P, C : \text{GenerateReport}(P, C) \rightarrow R$$

  The system shall generate a report R based on the predictions P and input data C.

10 Look and Feel Requirements

10.1 Appearance Requirements

LFR1: Dashboard Display of Predictions

- **Description**: The system's dashboard shall present the predicted cow traits (e.g., milk production, breeding success) in a structured and organized manner, clearly showing individual predictions for each cow.
- **Rationale**: Farmers need to quickly and easily interpret the predictions without searching through large amounts of data. An organized display ensures that all predictions can be understood at a glance.

- **Fit Criterion**: The system shall display predictions for multiple cows in a table format, with clear labels for each trait, such as milk production and herd retention likelihood.

LFR2: Text Contrast for Readability

- **Description**: All text displayed on the system interface shall use a high-contrast color scheme to ensure readability.
- **Rationale**: Farmers and users may access the system in various lighting conditions. High contrast, such as black text on a white background, will ensure clarity.
- **Fit Criterion**: The system shall use a high-contrast color scheme for all text, ensuring that it meets standard readability guidelines under different lighting conditions.

10.2 Style Requirements

LFS1: Consistent Formatting for Input Fields

- **Description**: All data input fields, such as for entering cow or parental traits, should follow a consistent format with clear labels and input validation.
- **Rationale**: A consistent layout for input fields will minimize errors and ensure ease of use when farmers input or update data.
- **Fit Criterion**: Input fields shall maintain a uniform format, with clear labels and consistent spacing throughout the interface.

LFS2: Minimalist Design for the Dashboard

- **Description**: The dashboard interface shall maintain a clean and minimalist design, avoiding unnecessary clutter or decorative elements.
- **Rationale**: A simplified interface will allow farmers to focus on the essential data (predictions) without distractions, ensuring ease of use.
- **Fit Criterion**: Over 80% of users in a usability test shall report that the dashboard is free from unnecessary elements and easy to navigate.

11 Usability and Humanity Requirements

11.1 Ease of Use Requirements

UHR1: Task Completion Efficiency

- **Description**: The user interface must allow users to complete common tasks, such as inputting data and retrieving predictions, within no more than 10 minutes.
- **Rationale**: Minimizing the number of steps needed to complete tasks will improve user efficiency and reduce frustration, ensuring a smoother workflow.
- **Fit Criterion**: During usability testing, all common tasks must be completed within 10 minutes for at least 90% of users.

UHR2: Quick Onboarding

- **Description**: New users without prior machine learning experience should be able to generate predictions using the system within 10 minutes, with assistance from help documentation or tooltips.
- **Rationale**: Reducing the learning curve ensures that new users can quickly adopt the system, improving overall usability and reducing training time.
- **Fit Criterion**: During user trials, at least 80% of new users must be able to generate a prediction within 10 minutes.

11.2 Personalization and Internationalization Requirements

UHR3: Unit Customization

- **Description**: The system should support switching between metric and imperial units for both input and output data related to cow metrics.

- **Rationale**: Users may use different measurement systems, so supporting unit customization ensures the system is applicable to a broader audience and will prevent users from using incorrect inputs due to unit mismatches.
- **Fit Criterion**: Usability testing must show that at least 90% of users can switch between units and see corresponding changes in inputs/outputs.

UHR4: Language Support

- **Description**: The system must offer support for English and French, ensuring all text elements and documentation are fully translated.
- **Rationale**: Offering English and French options makes the system more accessible to all Canadians in Ontario, expanding its usability.
- **Fit Criterion**: Usability testing must confirm that users can switch between languages in under 5 seconds, and all UI elements and help documentation must be fully translated with no missing text.

11.3 Learning Requirements

UHR5: User Proficiency

- **Description**: Users should be able to gain proficiency in the system's primary functions, such as data input and prediction generation, within 2 hours of active use.
- **Rationale**: Ensuring a reasonable learning curve enhances adoption rates and minimizes user frustration.
- **Fit Criterion**: At least 80% of users must pass a proficiency test after 2 hours of use, demonstrating the ability to use key system functions.

UHR6: Quick Access to Help

- **Description**: he system must provide contextual help (e.g., tooltips or tutorials) that can be accessed within 15 seconds for key features.
- **Rationale**: Easy access to help reduces confusion and minimizes time wasted on seeking external support, improving the overall user experience.

- **Fit Criterion**: Usability testing must demonstrate that users can activate contextual help for any key feature in under 5 seconds.

11.4 Understandability and Politeness Requirements

UHR7: Clear Output Explanation

- **Description**: The system must present model predictions in an understandable format, including explanations of key factors influencing the results.
- **Rationale**: Non-technical users should be able to understand how the model arrived at its conclusions, increasing trust and usability of the system.
- **Fit Criterion**: At least 80% of non-technical users must indicate that they understand the output and influencing factors after using the system.

UHR8: Informative Error Handling

- **Description**: Error messages should be clear, non-technical, and provide actionable steps for resolution, while being dismissible within 5 seconds.
- **Rationale**: Polite and informative error handling reduces user frustration and helps users quickly recover from mistakes or system issues.
- **Fit Criterion**: Usability testing must show that at least 90% of users can dismiss error messages within 5 seconds and understand how to resolve the issue.

11.5 Accessibility Requirements

UHR9: Keyboard Navigation Support

- **Description**: The system must allow users to navigate and interact with all functional elements using only the keyboard.
- **Rationale**: Ensuring that the system is accessible to users with motor disabilities or those who prefer keyboard navigation enhances inclusivity.

- **Fit Criterion**: Accessibility testing must confirm that all interactive elements can be accessed via keyboard shortcuts, with no more than 5% error rate.

UHR10: WCAG 2.1 AA Compliance

- **Description**: The system must meet WCAG 2.1 AA standards, ensuring that elements like color contrast, text size, and screen reader compatibility are accessible to users with disabilities.
- **Rationale**: Compliance with established accessibility standards ensures that the system is usable by individuals with a wide range of abilities, enhancing inclusivity.
- **Fit Criterion**: Automated accessibility testing and user trials must confirm that the system meets WCAG 2.1 AA standards, with no major violations.

12 Performance Requirements

12.1 Speed and Latency Requirements

PFR1: Prediction Response Time

- **Description**: The system must process input data and return predictions within 10 seconds under normal operating conditions.
- **Rationale**: Speed is critical for maintaining smooth user experiences and ensuring timely decision-making.
- **Fit Criterion**: 95% of predictions must be returned within 10 seconds during testing under normal conditions (100 concurrent users).

PFR2: User Interface Response Time

- **Description**: The system's user interface (UI) must respond to user interactions, such as button clicks and input field changes, within 0.5 seconds.
- **Rationale**: Fast UI response times ensure a smooth user experience and reduce frustration, especially for data input and navigation tasks.
- **Fit Criterion**: During usability testing, the system must respond to 90% of UI interactions within 0.5 seconds.

12.2 Safety-Critical Requirements

No safety-critical requirements needed.

12.3 Precision or Accuracy Requirements

PFR3: Prediction Accuracy Threshold

- **Description**: The machine learning model must achieve a minimum prediction accuracy of 85% when tested against a historical dataset of cow offspring features.
- **Rationale**: High prediction accuracy is essential for the system to be considered reliable in its intended agricultural context.
- **Fit Criterion**: Testing with a dataset of at least 1,000 samples must demonstrate a minimum accuracy of 85%, with results compared to known historical outcomes.

PFR4: Consistency of Prediction Output

- **Description**: The model must consistently produce the same prediction for the same set of input data, with no more than 1% variation in output across multiple trials.
- **Rationale**: Consistency in model outputs ensures reliability, preventing users from seeing inconsistent results when using identical inputs.
- **Fit Criterion**: Repeated predictions using the same input data must produce the same result 99% of the time in testing across 100 trials.

12.4 Robustness or Fault-Tolerance Requirements

PFR5: System Recovery From Failures

- **Description**: In the event of a system failure, the system must be able to recover within 30 seconds and restore all data from the last stable state.
- **Rationale**: Fault tolerance ensures that users do not lose data or progress due to system crashes, increasing system reliability.

- **Fit Criterion**: Testing must demonstrate system recovery within 30 seconds after simulated failures, with 100% of data restored to its last known state.

PFR6: Graceful Handling of Invalid Inputs

- **Description**: The system must handle invalid inputs without crashing, providing clear feedback to the user and offering solutions to correct the input.
- **Rationale**: Robust handling of invalid inputs ensures system stability and improves the user experience by guiding them through input corrections.
- **Fit Criterion**: Fault tolerance testing must show that 100% of invalid inputs are flagged, and appropriate error messages are displayed without causing system crashes.

PFR7: Load Balancing for Heavy Traffic

- **Description**: The system must automatically distribute traffic across servers in case of high load, ensuring stable performance for all users.
- **Rationale**: Robust load balancing prevents system downtime or performance degradation during periods of heavy usage.
- **Fit Criterion**: Load tests must demonstrate that the system can handle spikes in traffic (up to 150% of normal load) without response times exceeding 10 seconds.

12.5 Capacity Requirements

PFR8: User Load Capacity

- **Description**: The system must be able to handle at least 500 concurrent users without performance degradation.
- **Rationale**: Ensuring high capacity allows for scalability and supports widespread adoption of the system by large user bases.
- **Fit Criterion**: Load testing must confirm that the system can support at least 500 concurrent users while maintaining prediction response times of under 15 seconds.

12.6 Scalability or Extensibility Requirements

PFR9: Modular Design

- **Description**: The system's architecture must be modular, allowing new features such as additional machine learning models, data types, or metrics to be integrated without major redesign.
- **Rationale**: A modular design enables easy expansion as new business requirements emerge, increasing the system's longevity and adaptability.
- **Fit Criterion**: Documentation and testing must confirm that new features can be added or updated without affecting the core functionality of existing modules.

PFR10: Vertical Scaling for Increased Demand

- **Description**: The system must be able to scale vertically to handle increased demand by adding more computational resources without redesigning the architecture.
- **Rationale**: Vertical scalability ensures that as demand grows, the system can continue to function efficiently by utilizing additional hardware resources.
- **Fit Criterion**: Stress tests must demonstrate that adding computational resources (e.g., CPUs, memory) results in proportional performance gains under increased loads.

12.7 Longevity Requirements

PFR11: Code Maintainability

- **Description**: The system's codebase must be well-documented, following the set coding standards to ensure that it can be maintained and extended over the next 5 years without requiring significant rework.
- **Rationale**: Ensuring code maintainability reduces future technical debt and allows for easier updates and feature expansions.

- **Fit Criterion**: Code reviews and external audits must confirm that the code adheres to industry-standard practices and includes sufficient documentation for future maintainers.

PFR12: Support for Future Technologies

- **Description**: The system must be designed to support the integration of future technologies, including newer machine learning models or external data sources, without requiring a full system overhaul.
- **Rationale**: Designing for longevity ensures that the system can evolve with advances in technology, preserving its relevance and utility.
- **Fit Criterion**: System architecture and documentation must demonstrate that the system can accommodate integration of future technologies with minimal redesign.

13 Operational and Environmental Requirements

13.1 Expected Physical Environment

OER1: Network Stability

- **Description**: The system must operate reliably over both stable and fluctuating internet connections.
- **Rationale**: Given that the CATTLEytics system may be deployed in rural farm settings with less stable internet, the system needs to handle both stable and intermittent connectivity.
- **Fit Criterion**: The system must function with no critical failures in environments with intermittent connectivity and maintain prediction response times within 25% of normal operation during network fluctuations.

13.2 Wider Environment Requirements

OER2: Sustainability in Hosting

- **Description**: The system's cloud-based infrastructure must prioritize sustainability, with at least 50% of its hosting powered by renewable energy sources.
- **Rationale**: Environmental sustainability is a growing priority, and using renewable energy sources can reduce the system's carbon footprint.
- **Fit Criterion**: Hosting provider documentation must show that at least 50% of the energy used in hosting services comes from renewable sources.

13.3 Requirements for Interfacing with Adjacent Systems

OER3: API Compatability with CATTLEytics

- **Description**: The system must provide a well-documented API to interface with CATTLEytics' existing application and third-party tools, supporting data exchange in standard formats.
- **Rationale**: Ensuring smooth data exchange between the prediction model and adjacent systems, such as farm management platforms, is essential for integration and workflow efficiency.
- **Fit Criterion**: API testing must confirm successful data exchange with the CATTLEytics platform and third-party tools, with 100% of test cases passing.

OER4: Seamless Data Transfer

- **Description**: The system must ensure that data transfers between the prediction model and adjacent systems are fault-tolerant, with automatic retries for failed transmissions.
- **Rationale**: Seamless data integration is essential to prevent loss of data during transmission, ensuring that all related systems remain synchronized and accurate.
- **Fit Criterion**: In testing, the system must recover from 100% of simulated transmission failures by automatically retrying data transfers without user intervention.

13.4 Productization Requirements

OER5: Automated Updates

- **Description**: The system must support automated updates, ensuring that all future software patches and improvements can be deployed without user intervention.
- **Rationale**: Automated updates ensure that the system remains up-to-date with the latest features and security patches without requiring manual intervention, reducing downtime and maintenance costs.
- **Fit Criterion**: Testing must confirm that updates are automatically deployed in 100% of test cases, with no need for user interaction during the update process.

13.5 Release Requirements

OER6: Testing and Revision Cycles

- **Description**: The system must undergo a formal testing phase, involving key stakeholders (e.g., farm operators, internal CATTLEytics team members), with feedback collected to identify potential issues before final release.
- **Rationale**: Testing with actual users ensures that real-world use cases and unforeseen issues are identified and resolved, improving overall product quality.
- **Fit Criterion**: Feedback from testers must be collected and analyzed, with all critical issues resolved before the final product release. At least 10 users must be involved in this testing phase.

OER7: Documentation for End Users and Developers

- **Description**: The system must include comprehensive documentation for both end users (features, usage) and developers (API, integration, system architecture) prior to release.
- **Rationale**: Clear and thorough documentation ensures that both users and developers can effectively utilize and maintain the system, reducing support costs and improving product adoption.

- **Fit Criterion**: Documentation must pass an internal review, with at least 90% of users and developers rating the documentation as "sufficient" or better in post-release surveys.

OER8: Release Versioning and Rollback Support

- **Description**: Each system release must be versioned, with the ability to rollback to a previous stable version if critical issues arise post-release.
- **Rationale**: Versioning and rollback capabilities are essential for ensuring stability and addressing issues that might emerge in new releases, minimizing system downtime and user impact.
- **Fit Criterion**: All releases must be properly versioned, and rollback testing must confirm that previous versions can be restored without data loss or functionality degradation.

14 Maintainability and Support Requirements

14.1 Maintenance Requirements

MSR1: Scheduled Maintenance Windows

- **Description**: The system must support scheduled maintenance windows, during which the application can be updated or maintained without impacting user activity.
- **Rationale**: Planned maintenance reduces the risk of unexpected downtime and allows system updates to be performed in a controlled manner.
- **Fit Criterion**: Testing must demonstrate that the system can be taken offline for maintenance and restored within a scheduled maintenance window of no more than 4 hours, with no data loss.

MSR2: Automated Testing for Releases

- **Description**: The system must incorporate automated testing to ensure that all new maintenance releases pass unit, integration, and regression tests before deployment.

- **Rationale**: Automated testing ensures that updates or bug fixes do not introduce new errors, reducing the time and effort required for manual testing.
- **Fit Criterion**: Automated tests must cover at least 80% of the codebase, with no critical failures in tests run before each maintenance release.

14.2 Supportability Requirements

MSR3: Multi-tier Support Structure

- **Description**: The system must support a multi-tier support structure, offering different levels of assistance, from basic user help (FAQs) to advanced technical support (dedicated IT assistance).
- **Rationale**: A tiered support system ensures that users receive the appropriate level of assistance based on the complexity of the issue, improving resolution efficiency.
- **Fit Criterion**: Support logs must show that 90% of issues are resolved within the expected timeframe for each support tier, with at least 75% of basic issues resolved through self-service (e.g., FAQs).

MSR4: Comprehensive Error Logging

- **Description**: The system must implement comprehensive error logging, capturing detailed error messages for all exceptions and performance issues.
- **Rationale**: Detailed error logs help support teams quickly diagnose and resolve problems, ensuring that users experience minimal downtime.
- **Fit Criterion**: Error logs must capture 100% of critical errors during testing, with logs providing detailed information sufficient for issue resolution in 90% of cases.

14.3 Adaptability Requirements

MSR5: Plug-in Architecture for Future Features

- **Description**: The system must be built using a plug-in architecture, allowing new features or modules (e.g., additional prediction algorithms) to be added without impacting core functionality.
- **Rationale**: A plug-in architecture allows the system to be extended easily in the future, enabling it to adapt to evolving business or technical requirements.
- **Fit Criterion**: At least one new feature or module must be added as a plug-in during testing, with no impact on the core system or existing features.

15 Security Requirements

15.1 Access Requirements

SCR1: Role-Based Access Control (RBAC)

- **Description**: The system must implement role-based access control to restrict access based on user roles (e.g., administrator, data analyst, external user).
- **Rationale**: RBAC ensures that only authorized users can access sensitive parts of the system, reducing the risk of unauthorized access or manipulation.
- **Fit Criterion**: Access tests must verify that users with different roles can only access the functionalities or data assigned to their role, with 100% compliance in role-based access rules during testing.

SCR2: Session Timeout

- **Description**: The system must automatically log out users after 15 minutes of inactivity to prevent unauthorized access via unattended sessions.

- **Rationale**: Automatic session timeout reduces the likelihood of unauthorized access due to forgotten or unattended active sessions.
- **Fit Criterion**: Testing must show that all user sessions are automatically terminated after 15 minutes of inactivity, with users being required to log back in to regain access.

15.2 Integrity Requirements

SCR3: Data Validation on Input

- **Description**: The system must implement robust input validation to prevent invalid or malicious data from being entered and processed by the application.
- **Rationale**: Proper data validation protects against injection attacks (e.g., SQL injection) and ensures that only valid data is processed.
- **Fit Criterion**: Penetration testing must confirm that no invalid or malicious data bypasses input validation mechanisms in 100% of test cases.

SCR4: Transaction Atomicity

- **Description**: The system must ensure that all data transactions are atomic, meaning they are either fully completed or fully rolled back in case of an error.
- **Rationale**: Atomic transactions ensure data integrity by preventing partial updates that could corrupt data or leave the system in an inconsistent state.
- **Fit Criterion**: During testing, all simulated transaction failures must result in a full rollback with no partial updates to the database.

SCR5: Data Consistency Across Systems

- **Description**: The system must maintain data consistency across all integrated systems, ensuring that updates or changes are reflected accurately across the board.

- **Rationale**: Consistent data across integrated systems prevents discrepancies that could lead to erroneous predictions or reports.
- **Fit Criterion**: Consistency checks during testing must verify that updates in one system are correctly reflected across all other systems in 100% of test cases.

15.3 Privacy Requirements

SCR6: Data Encryption

- **Description**: All sensitive data, including personal information and prediction results, must be encrypted both at rest and in transit using industry-standard encryption protocols.
- **Rationale**: Encrypting sensitive data ensures that even if the data is intercepted or accessed without authorization, it remains unreadable and secure.
- **Fit Criterion**: Testing must confirm that all sensitive data is encrypted with no plaintext exposure during storage or transmission, potentially using protocols such as AES-256 for encryption at rest and TLS 1.2+ for encryption in transit.

15.4 Audit Requirements

SCR7: Data Access Monitoring

- **Description**: The system must monitor and log all access to sensitive data, with alerts triggered for any unauthorized access attempts.
- **Rationale**: Continuous monitoring of data access ensures that any unauthorized or suspicious activity is detected early, preventing data breaches.
- **Fit Criterion**: Testing must confirm that unauthorized access attempts trigger an alert in real time, and all data access is logged and available for review in audit logs.

SCR8: Audit Logging of Sensitive Operations

- **Description**: The system must log all sensitive operations (e.g., data access, user account changes, system configuration changes) for auditing and compliance purposes.
- **Rationale**: Audit logs allow tracking of who did what and when, providing transparency and accountability for system changes and data access.
- **Fit Criterion**: Testing must verify that all sensitive operations are logged, with each log entry containing a timestamp, user ID, and details of the action performed.

SCR9: Audit Log Retention Policy

- **Description**: The system must retain audit logs for a minimum of 1 year, ensuring that historical data can be reviewed for compliance and forensic investigations.
- **Rationale**: Retaining audit logs for an adequate period is essential for compliance with regulatory requirements and for investigating any incidents that occur.
- **Fit Criterion**: Testing must confirm that audit logs are stored securely and accessible for at least 1 year, with no loss of log data during the retention period.

15.5 Immunity Requirements

SCR10: Resilience Against Denial-of-Service (DoS) Attacks

- **Description**: The system must be resilient to denial-of-service (DoS) attacks, with automatic measures in place to mitigate the impact of such attacks.
- **Rationale**: DoS attacks can disrupt system availability and cause significant downtime, making resilience essential for maintaining uninterrupted service.
- **Fit Criterion**: Simulated DoS attacks must show that the system remains available for 95% of legitimate users, with automated scaling or mitigation strategies successfully limiting the attack's impact.

SCR11: Recovery From Cyberattacks

- **Description**: The system must have a defined recovery protocol that ensures operations can be restored within 4 hours in the event of a successful cyberattack.
- **Rationale**: In case of a security breach, a fast recovery process minimizes downtime and limits the damage caused by the attack.
- **Fit Criterion**: Disaster recovery tests must confirm that the system can be fully restored within 4 hours after a simulated breach, with no loss of critical data.

16 Cultural Requirements

16.1 Cultural Requirements

CR1: Language and Units

- **Description**: The system must primarily use English for all user interfaces and documentation. All data and measurements should follow Canadian standards, including liters for milk production, kilograms for weight, hectares for land area, Celsius for temperature, and metric tons for large quantities.
- **Rationale**: Using standardized language and units ensures the product is accessible and relevant to Canadian dairy farmers and adheres to local agricultural practices.
- **Fit Criterion**: User interface testing must confirm that all text appears in English, and data entries for milk, weight, temperature, and land area are consistently displayed in the appropriate Canadian units.

17 Compliance Requirements

17.1 Legal Requirements

LR1: Code of Practice for Dairy Cattle

- **Description**: The project must comply with the Code of Practice for the Care and Handling of Dairy Cattle, a regulation by the Canadian government for ethical treatment and welfare of dairy cattle.
- **Rationale**: This code outlines the mandatory guidelines for ensuring the health and welfare of dairy cattle in Canada. Any recommendations or outputs from the machine learning model must align with these regulations.
- **Fit Criterion**: Testing must confirm that all suggested management actions and predictions follow the guidelines outlined in the Code of Practice, ensuring ethical treatment of the cattle.

LR2: PIPEDA Compliance

- **Description**: The project must comply with PIPEDA (Personal Information Protection and Electronic Documents Act) when handling dairy farmer information. This includes personal details like contact and financial information.
- **Rationale**: PIPEDA governs how personal information is collected, stored, and shared, ensuring the privacy of individuals associated with the project.
- **Fit Criterion**: Testing must confirm that any dairy farmer data is stored securely, access is restricted, and no personal data is shared without consent, in full compliance with PIPEDA regulations.

17.2 Standards Compliance Requirements

- There are no specific standards for collecting dairy farming data in this project. All relevant aspects of data collection and handling are already covered under Legal Requirements, specifically in compliance with PIPEDA for managing sensitive information about dairy farmers, and the Code of Practice for the Care and Handling of Dairy Cattle for ensuring the welfare of the animals.
- For coding standards, the project will adhere to PEP8 to ensure consistent and readable Python code. More information on PEP8 can be found here.

18 Open Issues

- **Data Availability and Quality:** The accuracy of predictions will heavily depend on the quality and completeness of the data obtained from CATTLEytics and Lactanet. Inconsistent or missing data might affect the performance of the model.
- **Model Accuracy:** The machine learning model may need to be fine-tuned multiple times to achieve high accuracy in predicting cow traits. This requires testing with diverse datasets to ensure the model generalizes well.
- **User Interface Usability:** The graphical representation of the family tree and predicted traits needs to be intuitive and user-friendly for farmers with varying levels of technical skill. Determining the best design and ensuring it meets users' needs could take time.
- **Integration with CATTLEytics:** Seamlessly integrating the tool into the existing CATTLEytics system without causing disruptions or requiring major system changes could be technically challenging.
- **Regulatory Compliance:** Ensuring that the predictions and recommendations made by the model comply with Canadian regulations for dairy farming (Code of Practice for the Care and Handling of Dairy Cattle) will require thorough review and potential adjustments during development.
- **Model Interpretability:** Farmers may need clear explanations for how predictions are made to trust and use the tool effectively. Ensuring the model's predictions are explainable is an open issue.
- **Performance Considerations:** The tool needs to be efficient and scalable, handling large amounts of data without significant lag or performance issues, especially as it gets adopted by multiple farms.

19 Off-the-Shelf Solutions

19.1 Ready-Made Products

- There are no fully ready-made products that address the predictive capabilities being developed in this project. While tools like Lactanet provide dairy farm data, they do not offer predictive models based on genetic and health data. Lactanet data will be used primarily for training the custom machine learning model.

19.2 Reusable Components

- Machine learning libraries, such as PyTorch or TensorFlow, will be utilized to develop the custom AI model for cow trait prediction. Additionally, front-end libraries such as D3.js or React Tree Visualization libraries could be considered for visualizing the family-tree diagrams.

19.3 Products That Can Be Copied

- There are no existing products to be copied for this project. However, open-source family-tree visualization tools might serve as inspiration for the graphical aspects of the project.

20 New Problems

20.1 Effects on the Current Environment

- Introducing this system could change how farmers currently select sires or evaluate herd performance. Some may resist adopting new technology due to unfamiliarity.

20.2 Effects on the Installed Systems

- The project will be integrated into the existing Cattleytics software, which is already used to manage dairy farms. The machine learning tool will act as an additional module within Cattleytics, allowing farmers to visualize the family tree of cows and predict future traits based on

genetic data. Seamless integration with the current system will be prioritized to ensure smooth adoption and ease of use.

20.3 Potential User Problems

- Users may face difficulties interpreting complex AI model outputs, so ensuring the tool's recommendations are easy to understand is key.

20.4 Limitations in the Anticipated Implementation Environment That May Inhibit the New Product

- The tool will need to function effectively on standard farm computing systems, which may have limited processing power or internet connectivity.

20.5 Follow-Up Problems

- Continuous updates may be needed to improve the model based on feedback from farmers. Future updates may also need to address changes in farming practices

21 Tasks

21.1 Project Planning

The project will be scheduled to follow the deliverables deadlines as outlined in the SFWRENG 4G06 course outline.

21.2 Planning of the Development Phases

The development of the project will be conducted in four primary phases:

1. Data Acquisition and Initial Integration: This initial phase focuses on acquiring, cleaning, and integrating historical dairy farm data into the system. Activities include:

- Collecting and preprocessing datasets from CATTLEytics and Lactanet to ensure data quality and consistency.

Table 1: Project Documentation Tasks

Phase	**Task**	**Due Date**
Phase 1	Hazard Analysis	October 23, 2024
	Verification and Validation Plan	November 1, 2024
	Proof of Concept Demonstration	November 11–22, 2024
	Design Documentat	January 15, 2025
	Revision 0 Demonstration	February 3–14, 2025
Phase 2	Verification and Validation Report	March 7, 2025
	Final Demonstration	March 24–30, 2025
	Final Documentation	April 2, 2025

- Developing scripts to handle data normalization, unit conversion (e.g., to liters and kilograms), and encryption for compliance with PIPEDA.
- Implementing an initial data pipeline to allow seamless integration between the existing CATTLEytics system and the new module.

This phase will conclude with a Proof of Concept (POC) demonstration, where the pipeline and initial data handling capabilities will be validated.

2. Development of Predictive Model and Core Functionality: The second phase is dedicated to developing and integrating the core machine learning model responsible for predicting key outcomes such as breeding success rates. This phase involves:

- Experimenting with various algorithms (e.g., logistic regression, decision trees, neural networks) using Python libraries such as Scikit-learn, TensorFlow, or PyTorch to determine the most suitable approach for our data.
- Developing the model's training, validation, and testing processes, ensuring the system is optimized for accuracy and performance.
- Implementing backend APIs for data retrieval and model output generation to support the frontend development in the next phase.

This phase will culminate with an evaluation of the model's initial accuracy and robustness, and any necessary refinements will be documented for the next sprint.

3. System Integration and Frontend Development: In the third phase, the focus shifts to building the user-facing components and integrating the machine learning model into the existing CATTLEytics system. Activities include:

- Developing a React-based frontend interface that allows farmers to input data, visualize predictions, and interpret results in an intuitive manner.
- Implementing interactive visualizations using libraries like D3.js to display herd health trends, genetic family trees, and predictive insights.
- Integrating the frontend with the backend APIs, ensuring seamless data flow and responsiveness across the system.
- Initial system testing to verify that the user interface works well with the machine learning outputs and data pipeline.

This phase will end with a functional demo of the frontend and backend integration, allowing for early user feedback and adjustments.

4. Testing, Deployment, and Refinement: The final phase focuses on refining the system, conducting extensive testing, and preparing for deployment. Key activities include:

- Performing unit, integration, and system-level testing to validate the accuracy, performance, and reliability of the predictive model and frontend interface.
- Setting up a Continuous Integration/Continuous Deployment (CI/CD) pipeline using GitHub Actions to automate testing and deployment to a staging environment.
- Conducting user testing sessions with industry stakeholders to gather feedback, refine features, and enhance the usability of the system.
- Finalizing user documentation, training materials, and preparing the system for deployment to the production environment.

This phase will conclude with a complete system demonstration and the final handoff to stakeholders, ensuring the solution meets all requirements and is ready for the EXPO presentation.

All phases will run concurrently with documentation development to meet capstone deliverables and ensure that non-functional requirements, such as performance and security, are integrated throughout the development process. Each phase will have some overlap to allow flexibility and responsiveness to evolving requirements.

22 Migration to the New Product

22.1 Requirements for Migration to the New Product

The new predictive module that is being developed for this project will need to be integrated into the existing CATTLEytics software. Key requirements include:

- Ensuring compatibility of the new module with existing CATTLEytics data formats and APIs.
- Ensuring that the frontend interface seamlessly integrates with the existing user interface of CATTLEytics.
- Developing a rollback plan to ensure that, if issues occur during migration, the system can revert to the previous stable state.
- Testing the migration process on a staging environment to validate functionality before deployment.

22.2 Data That Has to be Modified or Translated for the New System

This section will be completed once the relevant data model details are available.

23 Costs

The goal for this project is to minimize costs by leveraging open-source tools and resources wherever possible. The primary development tools, including

Python, TensorFlow, PyTorch, React, and GitHub for version control, are open-source and free to use. Continuous Integration will be managed through GitHub Actions, which offers a free tier suitable for our needs.

Any additional datasets needed from third-party providers (e.g., Lactanet) may incur small fees, but these will be minimized through the use of open or freely available datasets when possible. The team at CATTLEytics will provide access to a dataset that they have scraped from Lactanet.

At this time, no other significant direct monetary costs are anticipated. The project plan remains flexible to adapt if unforeseen expenses arise during development, but the team will prioritize cost-effective and open-source solutions whenever feasible.

24 User Documentation and Training

24.1 User Documentation Requirements

The user documentation for the system will be concise and user-friendly, aimed at ensuring farmers and stakeholders can easily understand and utilize the tool. The documentation will include:

- Step-by-step instructions on accessing and utilizing the platform.
- Instructions on interpreting reports generated by the system.
- A troubleshooting section to address common issues.

24.2 Training Requirements

No formal training will be required as the tool is designed to be intuitive and straightforward. The provided documentation will be sufficient for users to operate the system without additional assistance.

25 Waiting Room

This section will be updated to highlight requirements that need to be put on hold due to time constraints or other factors.

26 Ideas for Solution

Insert your content here.

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Lifelong Learning. Please answer the following questions:

1. What knowledge and skills will the team collectively need to acquire to successfully complete this capstone project? Examples of possible knowledge to acquire include domain specific knowledge from the domain of your application, or software engineering knowledge, mechatronics knowledge or computer science knowledge. Skills may be related to technology, or writing, or presentation, or team management, etc. You should look to identify at least one item for each team member.

2. For each of the knowledge areas and skills identified in the previous question, what are at least two approaches to acquiring the knowledge or mastering the skill? Of the identified approaches, which will each team member pursue, and why did they make this choice?