**Detailed Design Specification**

**Team Name:** Mean Green Solutions

**Team Members:**

* Albert Kileo
* Jack Follett
* Jesus Chavez

**Project Name:** Goat Observation and Assessment Technology (G.O.A.T)

**Sponsor Information:** Rebecca Wierschke ([rebeccawierschke@gmail.com](mailto:rebeccawierschke@gmail.com))

### **Table of Contents**

Table of Contents

[Table of Contents 2](#_Toc189225677)

[1. Project Summary 3](#_Toc189225678)

[1.1 Overview 3](#_Toc189225679)

[1.2 Goals 3](#_Toc189225680)

[1.3 Importance 3](#_Toc189225681)

[2. Points of Contact 4](#_Toc189225682)

[3. Introduction 5](#_Toc189225683)

[4. Overall Description 5](#_Toc189225684)

[4.1 Product Perspective 5](#_Toc189225685)

[4.2 Product Functions 5](#_Toc189225686)

[4.3 Operating Environment 5](#_Toc189225687)

[4.4 Design Constraints 5](#_Toc189225688)

[4.5 User Documentation 6](#_Toc189225689)

[5. Use Case Diagrams 6](#_Toc189225690)

[Key Elements 6](#_Toc189225691)

[6. Class Diagrams 8](#_Toc189225692)

[7. State Diagrams 9](#_Toc189225693)

[8. Interaction Diagram 10](#_Toc189225695)

9. Component Diagram………………………………………………………………………………………………………………………….11

10. Package Diagram………………………………………………………………………………………………………………………..……11

# 1. Project Summary

**1.1 Overview**

The **G.O.A.T. project** aims to automate the grading of live goats using AI-powered image recognition. The system captures multi-angle images, analyzes physical attributes, assigns grades, and stores the data in a structured SQL database. The automation will improve accuracy, operational efficiency, and standardization in the goat meat industry.

**1.2 Goals**

* Process over **100 goats per day**.
* Automate image processing and grading.
* Store all relevant goat data in a database.
* Provide an intuitive interface for workers.

**1.3 Importance**

This system introduces **machine vision-based grading** in an industry lacking automation. By standardizing grading and pricing, it can impact **traceability, food safety, and economic efficiency** in the goat market.

# 2. Points of Contact

**Project Manager:**  
Jesus Chavez (JesusChavez@my.unt.edu)

**Technical Lead:**  
Jack Follett (JackFollett@my.unt.edu)

**Developers:**  
Albert Kileo (AlbertKileo@my.unt.edu)  
Jack Follett (JackFollett@my.unt.edu)  
Jesus Chavez (JesusChavez@my.unt.edu)

**Sponsor Contact:**  
Rebecca Wierschke ([rebeccawierschke@gmail.com](mailto:rebeccawierschke@gmail.com))

# 3. Introduction

This document refines previous design specifications by incorporating **updated class structures, algorithms, and additional diagrams (Component and Package Diagrams).**.

# 4. Overall Description

### **4.1 Product Perspective**

The G.O.A.T. system will integrate with **multiple cameras** to capture goat images, analyze them, and store structured information in a database. It will operate in an **industrial setting**, prioritizing **durability, efficiency, and automation.**

### **4.2 Product Functions**

* **Camera Interface:** Captures images from multiple angles.
* **AI Grading System:** Uses image processing to classify goats.
* **Database Management:** Stores goat grades, weights, and breeder details.
* **User Interface:** Allows operators to manage the system with minimal input.

### **4.3 Operating Environment**

* **Industrial setup with exposure to dust and animal residues.**
* **High durability cameras required.**
* **System must function in real-time with minimal downtime.**

### **4.4 Design Constraints**

* Developed in **Python (preferred)**.
* Cameras must be **high resolution with wired connectivity**.
* Must comply with **traceability regulations**.

### **4.5 User Documentation**

A comprehensive **user guide and troubleshooting manual** will be provided.

# 5. Use Case Diagrams

* The Use Case Diagram has been reviewed and updated to match the system's latest design.
* The following corrections were made.
  + - Added "Includes" and "Extends" relationships between relevant use cases.
    - Clarified actor roles for database storage and grading.

### **Key Elements**

**Actors:**

* **User**: Operates the system.
* **System**: Handles image capture, grading, and data storage.
* **Database**: Stores and manages records.

**Primary Use Cases:**

1. **System Setup**: Configure cameras and connect to the database.
2. **Capture Images**: Take photos of the goat from multiple angles.
3. **Analyze and Grade**: Process images and assign a score.
4. **Assign Price**: Determine pricing based on grade.
5. **Store Data**: Save goat records to the database.
6. **Retrieve Data**: View previous grading results.

A diagram of a software system

Description automatically generated

# 6. Class Diagrams

This section includes Class Diagrams that outline the main classes within the system, their attributes, methods, and relationships with each other. Each class represents a key component in the system, such as the **UserInterface, CameraInterface, ImageProcessing, GradingSystem, and DatabaseManager.** These classes illustrate how the system's components interact to facilitate image capturing, analysis, grading, and data storage in a cohesive structure.

This is Updated to include visibility, data types, and method details:

* + Visibility: Public (+), Private (-), Protected (#).
  + Data Types and Default Values: Attributes include default values (e.g., price: float = 0.0).
  + Methods: Detailed return types and parameter types added.

A screenshot of a computer

Description automatically generated

# 7. State Diagrams

This section provides State Diagrams that depict the various states of the system or its components, along with transitions between those states. Key states include **Idle, Capture Image, Process**

**Image, Assign Grade and Price, Store Data, and Error Handling**. The State Diagram illustrates how the system moves from one state to another as it performs different actions in the goat grading process.

Illustrates transitions between key states:

* **Idle** → **Capture Image** → **Process Image** → **Assign Grade & Price** → **Store Data** → **Completed/Error Handling**

# A diagram of a software company Description automatically generated

# 8. Interaction Diagrams

This section contains Interaction Diagrams, such as Sequence or Collaboration Diagrams, that show how objects within the system interact with one another. The Sequence Diagram details the flow of interactions between objects like **UserInterface, CameraInterface, ImageProcessing,**

**GradingSystem, and DatabaseManager**. It highlights the steps in which these components communicate to capture images, process data, assign grades, and store information, ensuring the system functions smoothly.

Pseudocode:

*START*

*-User initiates image capture via CameraInterface.*

*-CameraInterface captures images and sends to ImageProcessor.*

*-ImageProcessor analyzes images and sends data to GradingSystem.*

*-GradingSystem assigns grade and price.*

*-GradingSystem updates DatabaseManager with results.*

*-User retrieves data from DatabaseManager as needed.*

*END*

A diagram of a process

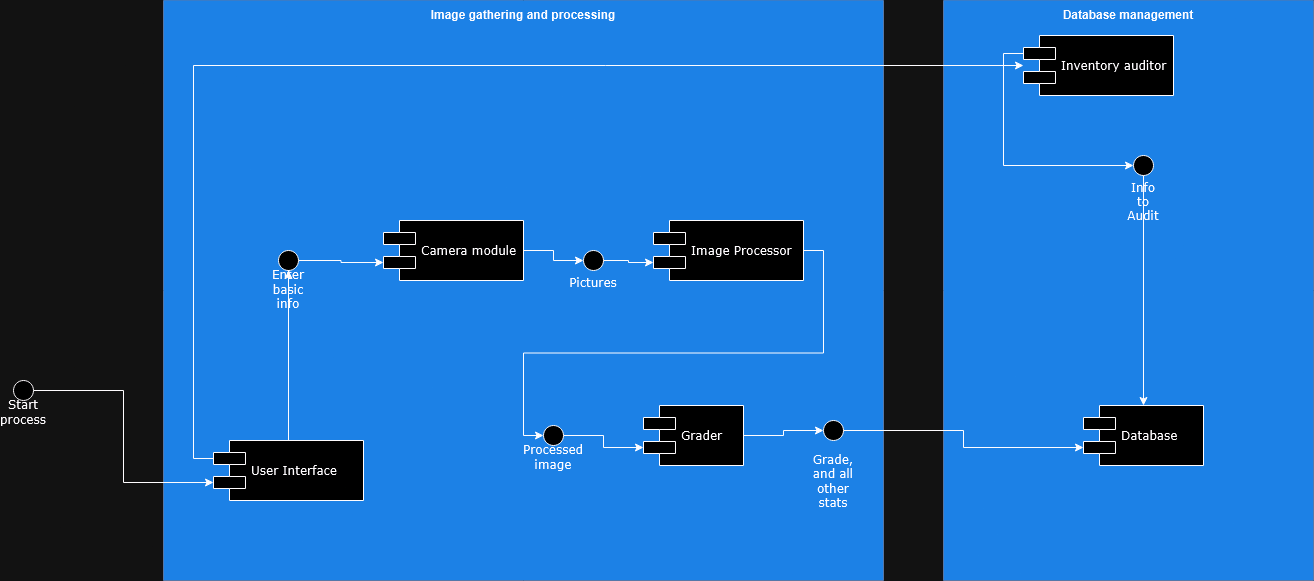
Description automatically generated

**9. Component Diagram**

The Component Diagram illustrates the architectural structure of the G.O.A.T system by showcasing its major components, their responsibilities, and interactions. It helps define how each component contributes to achieving the overall functionality of the system.

**Key Elements**:

* **User Interface**: Allows the worker to interact with the system for capturing images, retrieving data, and reviewing results.
* **Camera Module**: Handles the configuration and operation of cameras for image capture.
* **Image Processor**: Processes captured images to extract meaningful data for grading.
* **Grading Module**: Analyzes image data, assigns grades, and calculates prices.
* **Database Module**: Manages the storage and retrieval of goat grading and pricing data.



**10.Package Diagram**

The Package Diagram shows how classes and components are organized into logical groups or packages.

**Description**:

* The G.O.A.T System is divided into five main packages: UserInterfaceModule, CameraModule, ImageProcessingModule, GradingModule, and DatabaseModule.
* Dependencies between packages show how they interact and rely on each other to complete system operations.

**Key Elements**:

* **UserInterfaceModule**: Contains classes related to user interaction (e.g., UserInterface class).
* **CameraModule**: Groups classes that control and interface with cameras (e.g., CameraInterface class).
* **ImageProcessingModule**: Contains image analysis and processing logic (e.g., ImageProcessor class).
* **GradingModule**: Manages the grading logic and pricing calculation (e.g., GradingSystem class).
* **DatabaseModule**: Handles data storage and retrieval operations (e.g., DatabaseManager class).
* 