**CATTLE HERDING DRONE SYSTEM DESIGN**

1. **INTRODUCTION**

Advancements in computing, technology and telecommunications have led to the possibility of using drones for a range of activities which were previously carried out in a traditional way, one of which involves herding of livestocks by farmers (using e.g dogs). The traditional method involves a rigorous set of activities such as moving the herd, taking count, locating cattle that dispersed away from the herd, looking out for predators and accessing areas that are difficult to navigate.

With the proposed system, farmers will easily locate, monitor, count, and move herds to a designated location, thereby reducing the time involved in herding cattle, drastically reducing the number of accidents involving farmers on the farm, as well as the amount of pressure and stress on the animals.

The proposed drone design is equipped with a camera to locate the herd in the pastures, speakers for emitting barking sounds when moving the herd of cattle, solar panels for enhanced battery life, and back to base feature to enable the drone to fly back to base when the battery level reaches a set percentage.

Legislation was also put into consideration during the design process with regards to the privacy of user data to be collected by the software during usage. The amount of data collected, how this data is stored, user consent and explicit description of the collected information and how they will be used were not left out.

**1.1 OBJECTIVE**

The aim of this report is to detail the design process of a proposed drone herding system which can be used to herd cattle effectively without human interference by following software engineering methods. The designed system will be able to monitor, herd cattle, aggregate dispersed cattle and drive them to a designated designation.

**1.2 PROJECT STAKEHOLDERS**

Stakeholders can be described as those who have an impact in the success of this project. Stakeholders could be actors who will interact with the system and/or those involved in the development process.

The project stakeholders are the project managers, farmers, herdsmen, testers, and the software engineers. These sets of people engage in meetings, brain-storming sessions, requirements engineering process, design reviews, etc., until the final product is developed, tested, put into operation and maintained.

The project manager is responsible for coordinating the activities between the farmers, herdsmen, testers, and software engineers in order to be on the same page during the course of developing the system. The user requirements of the system are obtained from the farmers through meetings, interactions, etc., after which a proper analysis is carried out on the communicated requirements for possible refinement.

It is important for these stakeholders to be included in project design to ensure that the right requirements are built into the system, and ensure that everyone is on the same page in order for the final system to meet the farmers’ requirements.

1. **DESIGN OF THE DRONE CATTLE HERDING SYSTEM**

The proposed cattle herding system will be equipped with an aerial camera, barking sound system, GPS, and solar panel components. The aerial camera will be used for taking accurate and consistent imagery of the cattle and its environment, as well as for recording videos.

The barking sound system component will be responsible for emitting sounds which will aid in gathering and driving the cattle to a designated location. The barking feature will work as a replacement for the traditional role played by dogs, and helps in regrouping of dispersed cattle with the group.

To enhance the battery life of the proposed system, solar panels would be integrated to enable self charging feature for the battery while the drone is in use. As part of a way to minimise the impact of sudden shutdown as a result of critical battery level, this drone will include a return to base feature when the battery reaches a set level.

**2.1 IMPACT OF LEGISLATION ON THE DESIGN PROCESS**

The areas impacted by legislation are;

(i) data security,

(ii) data privacy,

(iii) users consent to collection and use of their personal data (data storage).

How user data is obtained, stored, processed and used is strongly impacted by legislation and therefore, serious attention must be paid to how users data is managed. GDPR frawns at companies storing data beyond what is needed and deemed necessary. It is required to gain explicit consent before collecting and storing users’ data, as well as documenting how such data will be used. The data needs to be obtained in such a way that there is no room for misinterpretation, it is required by GDPR that companies provide their privacy policy written in a plain language and accessible to users’ of the companies services.

In order to prevent legislative sanctions, the design process strictly adhered to data protection by design. This required the design process to securely store necessary users data, integrate regular deletion of securely stored data when they are no longer needed or after a period of time.Transparency about the use of users’ data was included, by providing information about what is being processed, why, how, and for how long they are being kept.

**2.2 THE SOFTWARE ENGINEERING PROCESS**

The successful development of any software requires following a set of software engineering methodologies. Software engineering encompasses the tools, methods, planning and processes involved in building a software. While methodology in software engineering can be described as a framework used to organise, plan, structure and control the process of developing a software system.

Automation of livestock herding is still very much in its early stage. The traditional method of using dogs for herding cattle is still the dominant in use around the world. Therefore, to deliver an optimal and effective automated herding system would require meticulous adherence to software engineering methods. The set of activities involved in designing the drone cattle herding system are discussed below:

**2.2.1 REQUIREMENTS ENGINEERING**

These are a set of activities carried in order to define the functionalities and operation constraints of the software to be developed. Requirements engineering comprises feasibility study, requirements specification, elicitation and validation.

**Requirements Gathering and Analysis**: The requirements for the potential system were gathered and analysed for possible modification or discarding of non promising requirements. These requirements were derived through meeting with the livestock farmers, herdsmen and other potential users, as well as through observation of the existing systems for cattle herding.

**Feasibility Study**: A study was carried out with respect to the gathered requirements, available resources, etc., to ascertain if the farmers' need for automated drones for herding cattle is cost effective when evaluated from a business point. The outcome of this study influenced and informed the decision of moving on to the next phase.

**Requirements Specification**: At this stage the set of requirements considered necessary after the analysis activity were translated into a document referred to as the requirements document. This document contains the system requirements which are the description of the requirements to be provided by the software and the user requirements which are the features desired by the potential users of the system. The requirements include camera for monitoring and visualisation, sound system with barking, solar panels for enhancing battery life, back to base feature when battery gets to a set level, amongst others.

**Requirements Validation**: Here, efforts were made in verifying the accepted conditions for literalism, predictability, and absoluteness. Requirements validation process ensures anomalies in the requirements document are inescapably discovered and corrected to resolve the noticed problems.

**2.2.2 SOFTWARE DESIGN AND IMPLEMENTATION**

This encompasses the set of concepts, principles and practices that leads to the development of a high quality system. Design is what almost every engineer wants to do. It is the place where stakeholders requirements, business needs and technical considerations all come together in the formulation of a system.

Implementation involves the transformation of the designed software into an actual system, using a programming language suitable for accomplishing the necessary requirements.

**LOGICAL VIEW OF THE PROPOSED SYSTEM**

The proposed system is made up of several classes which are responsible for different activities to be performed by the system. The detail and importance of each class will be explained alongside the methods it contains.

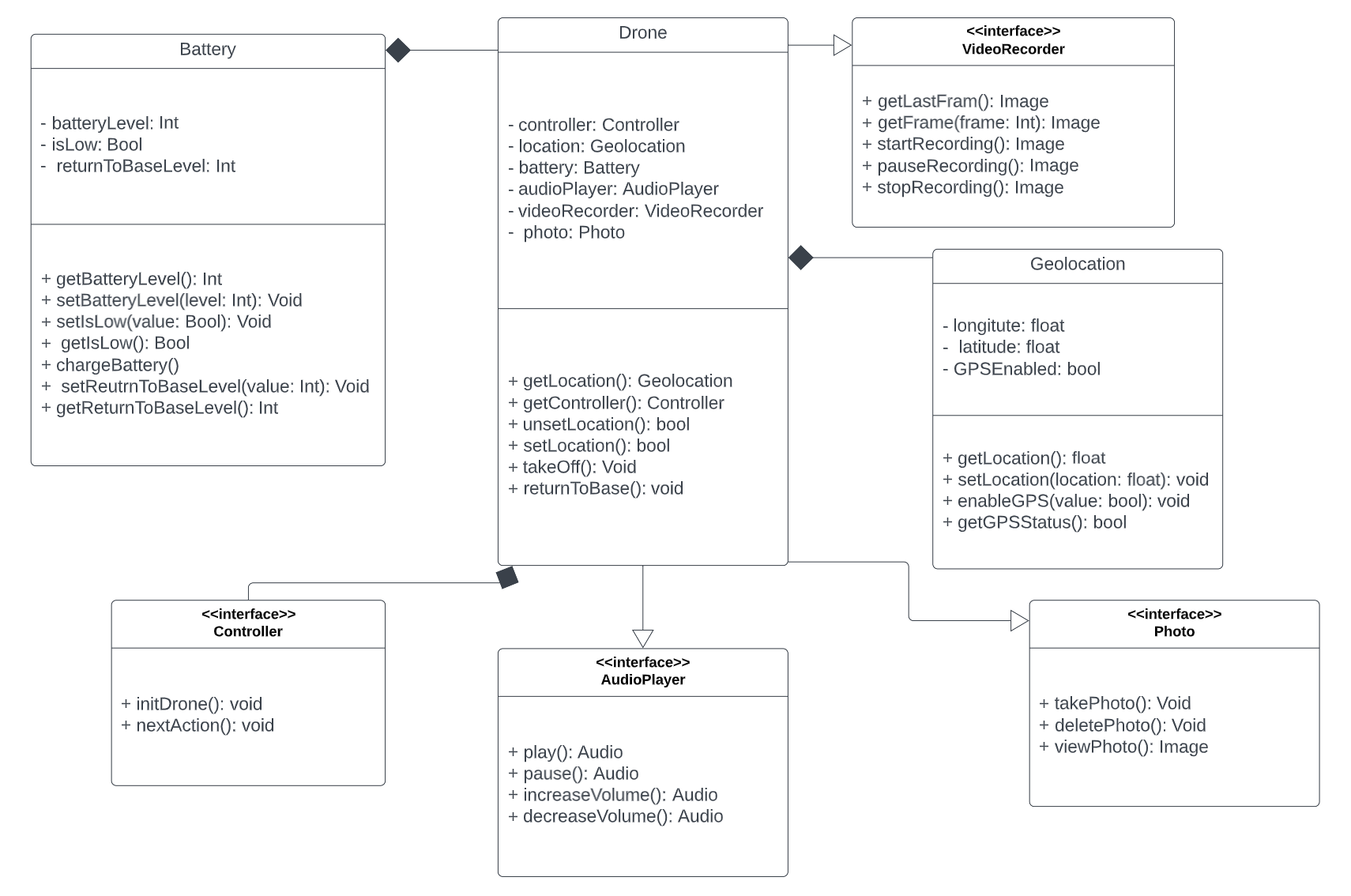


Fig 2.1 Class diagram of the proposed system

**Methods Encapsulation within Classes**:

Encapsulation can be described as the process of wrapping data and methods that perform actions on the data into a single unit referred to as a class. It is a technique for limiting unauthorised access to class data members and functions by using access modifiers to specify the level of access.

**Controller interface**: This is a general interface to be inherited by all classes that implements a controlling technique.This class has the abstract classes; initDrone to initialise the drone and nextAction for execution of the next function.

**Battery class**: is responsible for activities relating to the drone battery such as charging, setting and getting battery level, return base battery level, etc.

**Geolocation class**: is responsible for basic geolocational information. setLocation and getLocation are used for setting/retrieving current location by utilising GPS precision. getGPSStatus shows whether GPS is enabled or not.

**Drone class**: this is the class responsible for handling and conducting the activities of the drone in response to certain occurrences. Location, controller, battery, audio playing, video recording and photographing activities are taken care of within this class.

USE CASE DIAGRAM OF THE PROPOSED SYSTEM

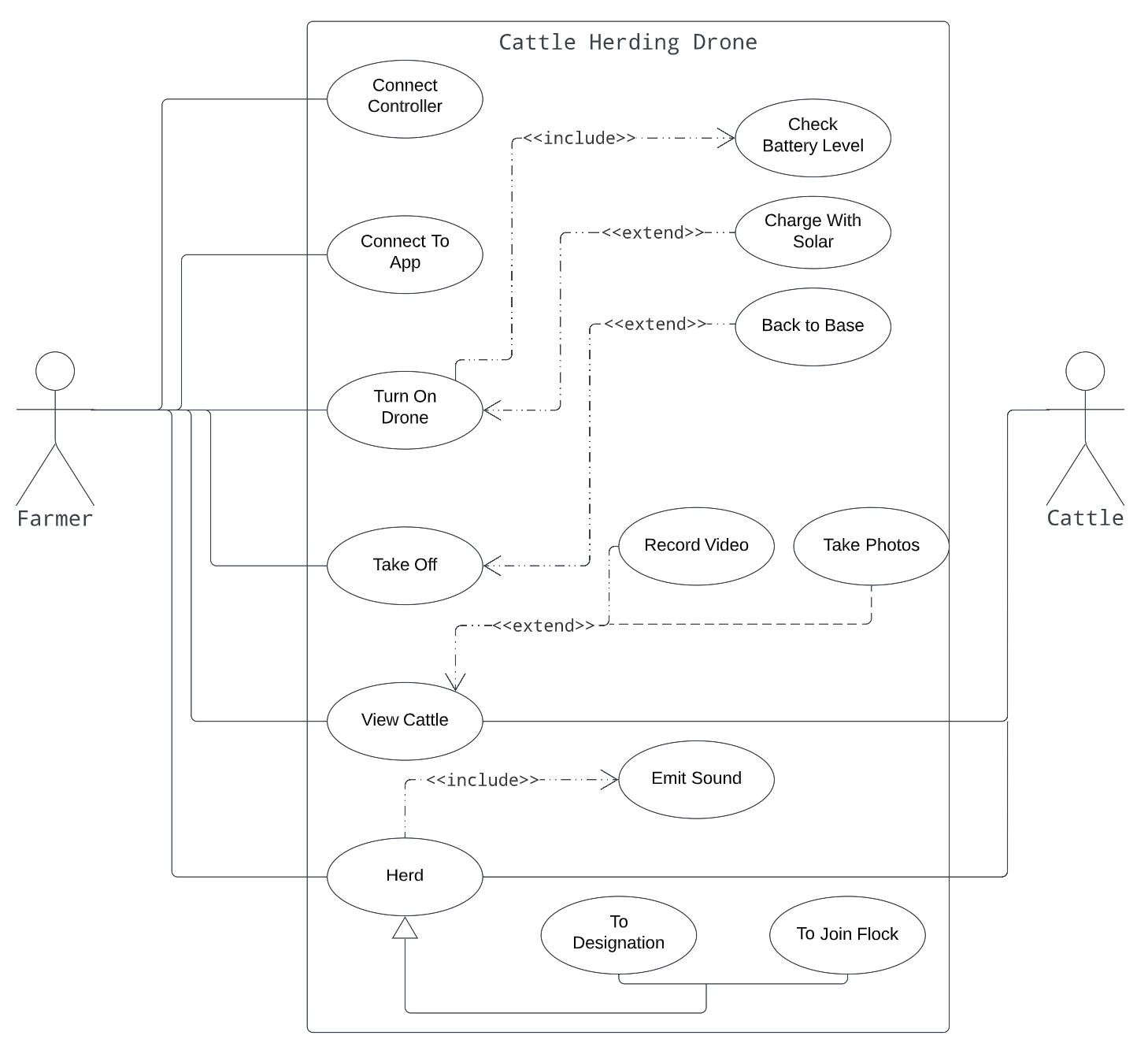


Fig 2.2 Use case diagram of the proposed system

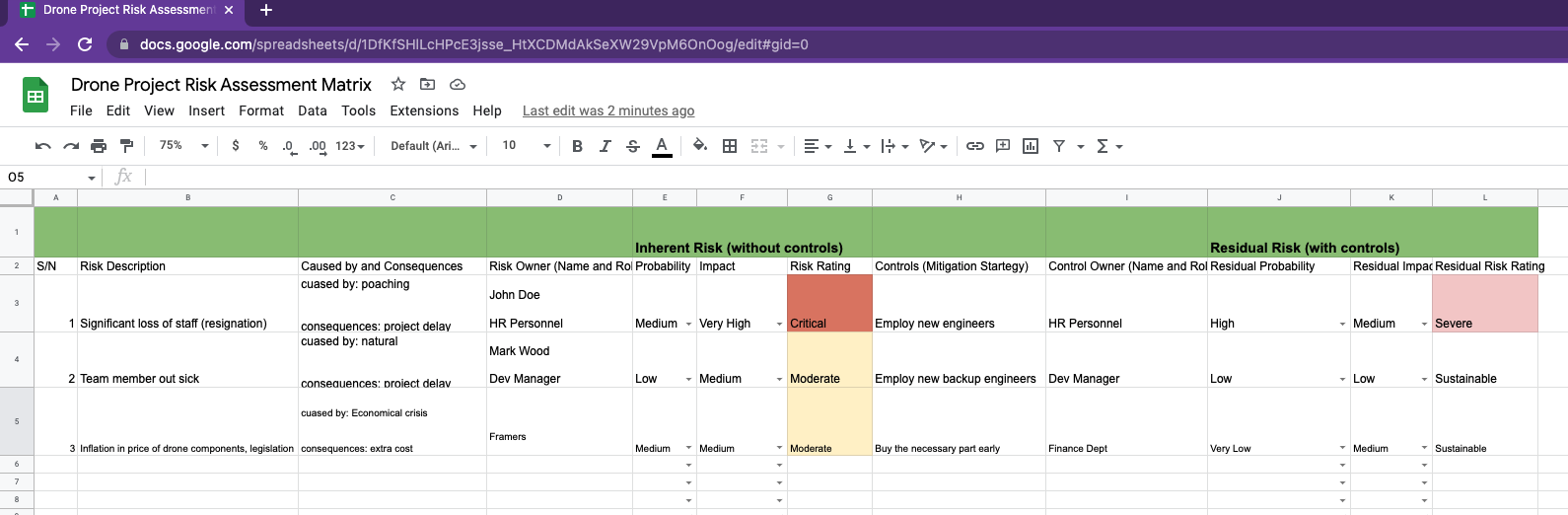
**3. SOFTWARE DESIGN PATTERNS**

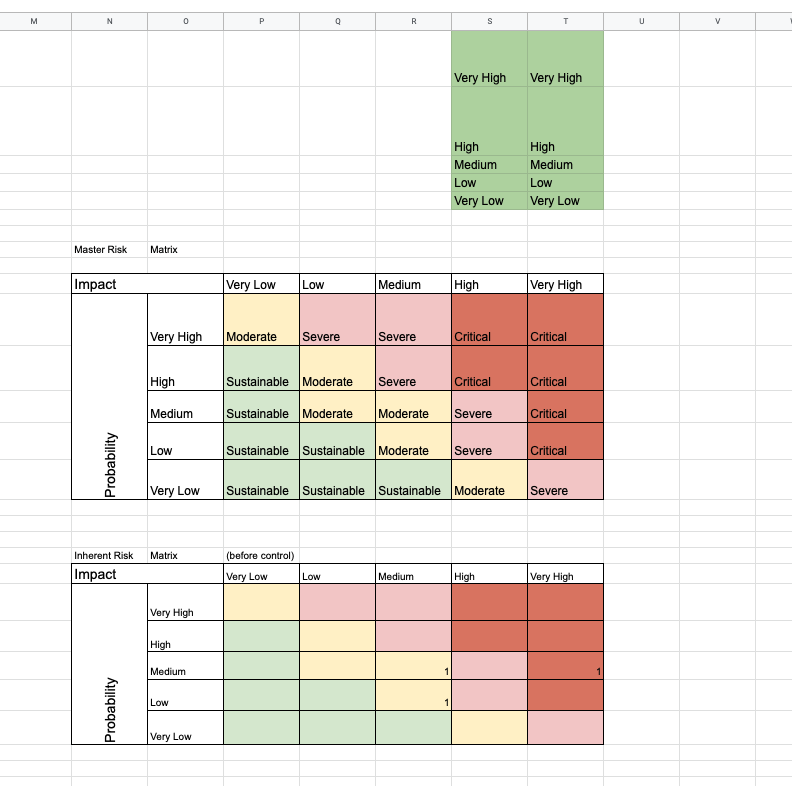
Describes a design structure that solves a particular design problem within a specific context and amid forces that may have impact on the manner in which the design is applied and used. The pattern is not a detailed specification. Rather, you can think of it as a description of accumulated wisdom and experience, a proven solution to a common problem. They are ways to describe best practices, good designs, and capture experience in a way that it is possible for others to reuse this experience.

The intent of each design pattern is to provide a description that enables a designer to determine whether the pattern is applicable to the current work, whether the pattern can be reused, and whether the pattern can serve as a guide for developing a similar pattern.

Other iterative activities to be carried out during and after the software implementation includes testing, requirements validation, software operation and evolution.

**RISK ASSESSMENT MATRIX FOR THE PROJECT**





**REFERENCES**

Lloyd, I. (2014). Information technology law. Oxford: Oxford University Press.

Pressman, R. (2015). Software engineering: A practitioner’s approach. New York: McGraw-Hill. Retrieved from,

<https://fliphtml5.com/pyiue/kkho/basic/301-350>

Sommerville, I. (2016). Software engineering (10th ed.). Essex, England: Pearson Education.

Xiaohui, L. *et al.,* (2022)**.** Robotic Herding of Farm Animals Using a Network of Barking Aerial Drones.

DOI; <https://doi.org/10.3390/drones6020029>

Alberto, R. *et al.* (2018).Detection of Cattle Using Drones and Convolutional Neural Networks

DOI; <https://doi.org/10.3390/s18072048>