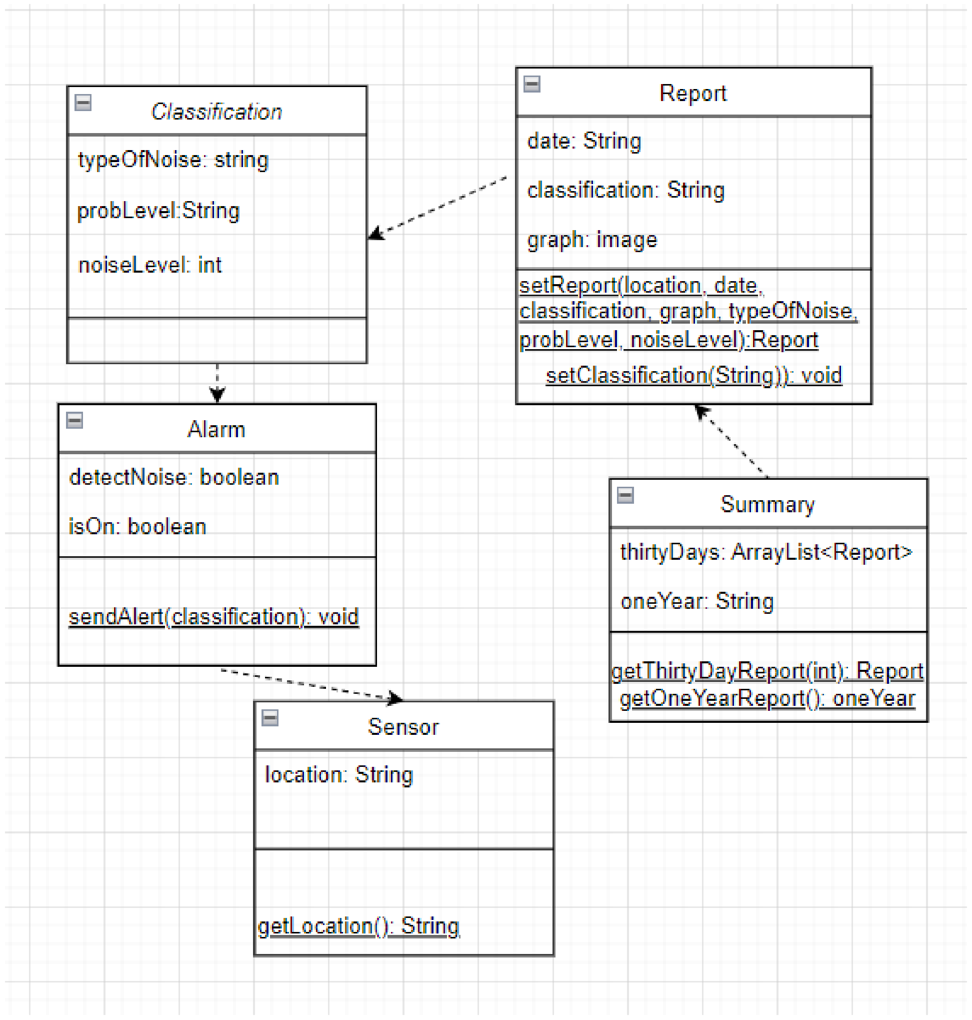
Mountain Lion Detection System

Aaron S.
Diego L.
Antonio T.

1. **System Description**
   This document introduces a mountain lion detection system designed for park rangers and staff. The system will generate reports based on animal noises detected by the sensor, allowing them to monitor and take appropriate actions. The system is designed for safety and emergency response purposes, facilitating quick responses in case of incidents involving mountain lions. It offers real-time monitoring, early warning alerts, data collection and analysis, mapping and visualization, alarm monitoring, and integration with park operations. The implementation of such systems can have both positive and potentially challenging impacts depending on the context and design. Potential benefits include improved tools and data for park staff to respond to risks and emergencies involving mountain lions, enhancing visitor safety.

2. **Software Architectural Overview**
   a. UML class Diagram classes


Classification
typeOfNoise: string
probLevel:String
noiseLevel: int
Report
date: String
classification: String
graph: image
setReport(location, date, classification, graph, typeOfNoise, probLevel, noiseLevel):Report
setClassification(String)): void
Alarm
detectNoise: boolean
isOn: boolean
sendAlert(classification): void
Summary
thirtyDays: ArrayList<Report>
oneYear: String
getThirtyDayReport(int): Report
getOneYearReport(): oneYear
Sensor
location: String
getLocation(): String

Key:

Class
Attributes
Operations

i. Sensor
   1. *Location: String* - Holds the location of the specified alarm sensor.
   2. getLocation():String - Operation gets the location of the specified alarm sensor.

ii. Alarm - uses Sensor
   1. *detectNoise: boolean* - Alarms will detect noise; if noise is detected detectNoise becomes true, otherwise it will return false
   2. *isOn: boolean* - If the alarm is on this will return true; otherwise will be false until a worker turns it back on after an alarm has been triggered
   3. sendAlert(classification) - When a noise is detected the alarm will use the Classification class to classify and send an alert to the control program. This will allow the creation of a report using information taken by the alarm.

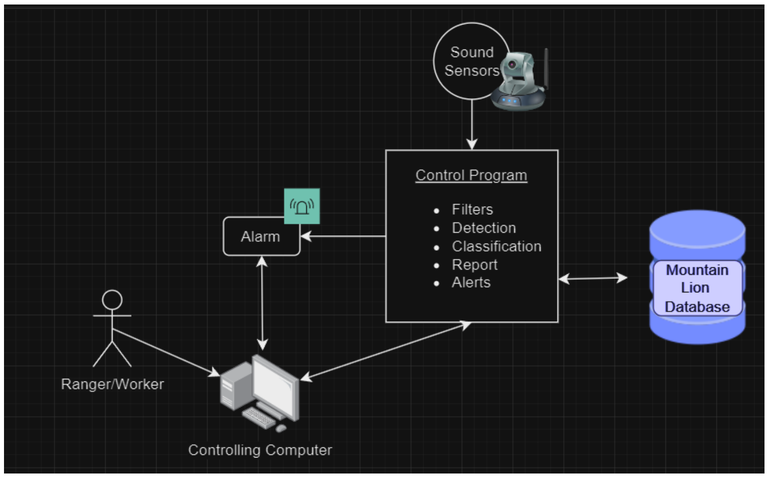
iii. Classification - uses Alarm class
   1. *typeOfNoise: String* - holds the type of noise taken from the alarm
   2. *probLevel: String* - holds the probability level that the noise from the alarm is a mountain lion.
   3. *noiseLevel: String* - holds the noise level that was taken from the alarm

iv. Report
   1. *date: String* - holds current date
   2. *classification: String* - holds classification of the noise
   3. *graph: image* - holds graph of general location where noise was heard from
   4. setClassification(classification): void - Operation sets the classification of the noise (being of mountain lion origin or not)
   5. createReport(location, date, classification, graph, typeOfNoise, probLevel, noiseLevel): Report - creates a Report based on location, date, classification, graph, typeOfNoise, probLevel, and noiseLevel.

v. Summary
   1. *thirtyDays: ArrayList<Report>* - array list holds Reports from the last 30 days

2. *oneYear: String* - holds all reports in a one year report
3. getThirtyDayReport(int): Report - Operation gets specified report from 30 day report array list
4. getOneYearReport(): oneYear - Operation gets the one year report

b. SWA Diagram



i. Description

The Mountain Lion System is composed of five primary components: the Controlling Computer, the Alarm System, the Control Program, Sound Sensors, and the Mountain Lion Database.

Controlling Computer: This serves as the central hub for system management and is accessible by authorized personnel such as Rangers or Workers. Its primary functions include:

- Querying the Control Program for reports and data summaries.
- Classifying sound threat levels that will be sent to the Control Program.
- Setting thresholds for alarm activation that the Control Program will save.

Alarm System: This component is responsible for issuing alerts based on predefined criteria. It is closely integrated with the Controlling Computer, which has the capability to deactivate the alarm once triggered.

Control Program: Often considered the "brain" of the Mountain Lion System, the Control Program has the following capabilities:

- Interacts with all other system components except the Actor.

- Sends and retrieves information from the Mountain Lion Database.
- Initiates the Alarm System when specific threat criteria are met.
- Send and receive information from the Controlling Computer

<u>Sound Sensors</u>: These sensors are designed to detect sound and send the corresponding data to the Control Program for analysis.

<u>Mountain Lion Database</u>: This is the system's data repository, storing information that can be queried or updated by the Control Program.

3. Development Plan and Timeline

Project Initiation (1 Month) -
To start, we will need stakeholder meetings and requirement gathering to define the objectives and constraints of the project. A project manager will be needed to look at overall project management and coordination. This person will need great communication skills, as they will be in communication with the stakeholders. In addition, this project manager will need to make sure that everything is coming along in a timely manner.

System Requirements and Design (2-3 Months) -
We will need to create a detailed software design specification document. In addition, we will need to develop system requirements based on stakeholders input. Lastly, we will need to create a high level system architecture and user interface design. At this time, we will need to hire a system architect to help with the system architecture and technology identification. In addition, we will need to have about 2-3 software developers who focus on the user interface design.

Development (4-8 Months) -
This is the time that the mountain lion detection software system will be built. We will implement real time monitoring and data collection features. This software will develop early warning alerts, as well as integrate mapping and visualization tools. The software developers will help develop the system modules.

Testing and Quality Assurance (9-10 Months) -
This is when we will conduct unit testing. This will help us verify the entire system's functionality. In addition, we will be able to address any bugs or issues relating to the system. For this, we will need to hire a QA team to help with testing and bug identification.

Monitoring and Maintenance (Ongoing) -
We will need to hire a maintenance and support team to focus on the ongoing monitoring and issue resolution.

Overall, this partitioning of tasks and team member responsibilities should ensure efficient project execution and successful development of the mountain lion detection system. Regular team meetings and progress checks will be essential for keeping the project on track.