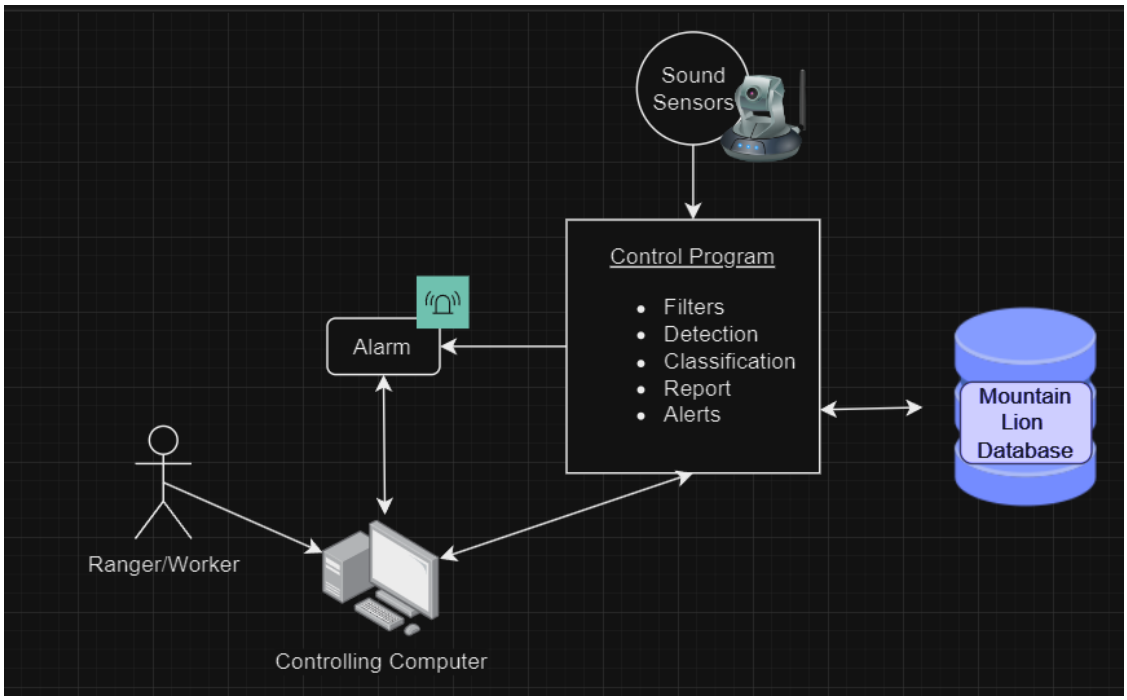
Mountain Lion Software Design 2.0

Aaron S.
Diego L.
Antonio T.

1. **Design Specification (SWA)**



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

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

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

2. **Data Management Strategy**

    a. Diagram

    Mountain Lion Database

    Sensors/Alarm Data

SensorID (int)	Location (String)	Status (Boolean)	NoiseDetected (Boolean)
001		True	False
002		True	False

    Classification Data (Connected to Sensor/Alarm)

ClassificationID (int)	TypeOfNoise (String)	ProbLevel (String)	NoiseLevel (int)

0001			
0002			

Reports Data (Connected to Classification data)

ReportID (int)	Date (String)	ClassificationID (int)	Graph (Image)
0001			

Summary data (Connected to Report Data)

SummaryID (int)	Report30Days (ReportID)	ReportOneYear (ReportID)
001		

b. Description

The Sensors/Alarm Data table, includes columns such as SensorID (an integer serving as a unique identifier), Location (a string specifying the sensor's location), Status (a boolean indicating operational status), and NoiseDetected (a boolean denoting whether the sensor has detected noise). The sample data within the table illustrates information for two sensors, where SensorID 001 is positioned in a location with detected noise, while SensorID 002 is in a location without noise detection but is also operational. This shows a system capable of tracking sensor-specific details, especially in scenarios where noise detection is a relevant parameter, such as security or environmental monitoring applications. The Classification Data (Connected to Sensor/Alarm) introduces a complementary table to the existing SQL tables, to incorporate classification details related to the sensor and alarm system. This Classification Data table

includes fields such as ClassificationID, TypeOfNoise, ProbLevel, and NoiseLevel. ClassificationID serves as a unique identifier for each entry, while TypeOfNoise is a string field for designating the category or type of noise detected by the sensors. The ProbLevel field represents the probability level associated with the detected noise type, and NoiseLevel is an integer field showing the intensity of the detected noise. This Reports Data table has columns such as ReportID, Date, ClassificationID, and Graph. ReportID functions as a unique identifier for each report, while the Date field records the date associated with the report. The ClassificationID column establishes a link with the previously introduced "Classification Data" table, showing that each report is connected to a specific noise classification. The Graph column is the inclusion of visual elements, potentially graphical representations or charts, within the reports. The Summary Data (Connected to Report Data), expands the database tables by incorporating a table specifically designed for summarizing information derived from reports. This table includes columns such as SummaryID, Report30Days (linked to ReportID from the last 30 days), and ReportOneYear (linked to ReportID from the last year). The SummaryID serves as a unique identifier for each summary entry. This addition implies a systematic approach to organizing key insights obtained from the generated reports.

3. TradeOff Discussion

We conducted a detailed analysis of database technology when developing our Mountain Lion Detection System, comparing and contrasting the benefits and downsides of SQL and non-SQL databases to find the best option for our needs.

Justification for SQL Database Selection:

We chose a relational database management system (RDBMS) employing SQL for several reasons:

Data Integrity and Relations: Our system requires robust data integrity with clearly defined relationships between various entities. SQL databases excel at enforcing these relationships through the use of foreign keys and unique constraints. For instance, our Reports Data table is directly tied to the Classification Data table, ensuring that reports are consistently linked to the correct classifications.
ACID Compliance: The Atomicity, Consistency, Isolation, Durability (ACID) properties guaranteed by SQL databases ensure reliable and safe transaction processing, a critical component for the emergency response features of our system.
Complex Querying: SQL databases provide powerful querying capabilities, allowing us to efficiently generate the complex reports and data summaries required by park rangers for proactive wildlife management.

Table Organization and Schema Design:

The schema has been carefully designed to normalize data and minimize redundancy. The tables have been structured to separate data logically, which simplifies maintenance and data retrieval processes. For example, sensor status updates can occur independently of classification data changes, reducing data redundancy and improving clarity.

Discussion of Alternatives and Tradeoffs:

While SQL databases offer numerous advantages for our application, we did consider non-SQL alternatives:

NoSQL Databases: These could provide schema flexibility and are well-suited to horizontal scaling. However, they typically lack the same level of support for ACID transactions and can make complex querying and reporting more cumbersome, which is a critical need for our system.

In summary, our necessity for dependable transactions, strict data integrity, and the need for advanced querying capabilities led us to choose SQL. We saw that NoSQL databases were flexible and scalable, but the compromises made them less suitable for our particular application. The SQL technique that was selected best fits our objectives of developing a stable, accurate, and efficient mountain lion detection system that supports park rangers' demands and enhances tourist safety.