Software Design Specification

Prepared by: Madeline Cohen, Josh Sindelar, and Spencer Van Rossum

November 8, 2025

System Description

The Mountain Lion Detection System (MLDS) is a specialized wildlife monitoring system developed for the San Diego County Parks and Recreation Department. Its primary purpose is to detect, record and report potential mountain lion activity within park boundaries to ensure visitor safety and help with wildlife management.

This system builds on the Animals-R-Here animal detection platform, which uses noise detection sensors to identify various animal sounds across an area of up to five square miles. The MLDS customizes this platform to specifically identify mountain lion noises and provide detailed alerts.

Each sensor will detect and transmit data such as type of animal noise, sound strength and location. The controlling computer, at the ranger station, receives all mountain lion alerts, maintains detailed logs for the past 30 days, and stores summarized alert data for up to one year.

When a new mountain lion alert is received, the system will sound an alarm at the ranger station. This alarm will continue until a ranger manually turns it off. Rangers can classify alerts as definite, suspected, or false, depending on the likelihood that a real mountain lion was detected.

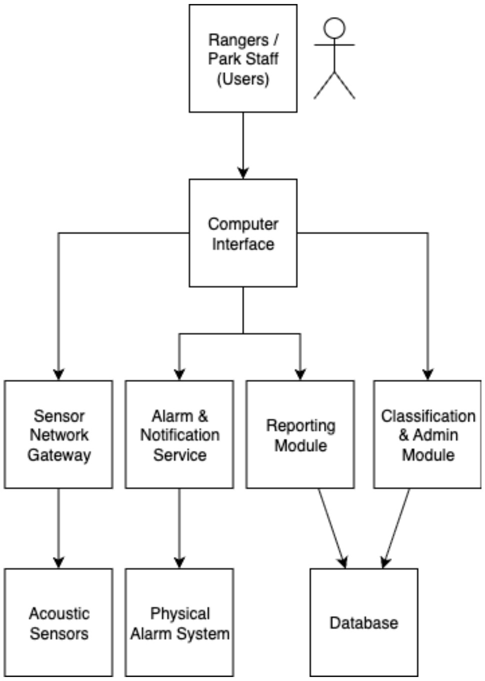
Additionally, the system allows rangers to generate several reports:

- A list of detections categorized by classification.
- A summary of detection for a specific sensor location.
- A map showing detection points within the park and up to two miles out of the park.

The MLDS is designed with scalability in mind so that it can be easily adapted for other parks in the California State Parks system

Software Architecture Overview

- Architectural diagram of all major components



Web/Computer Interface

- The main user interface, accessible from computers at ranger stations.
- It allows rangers to view incoming detection alerts, acknowledge or silence alarms, classify detections (definite/suspected/false), and generate reports.

Sensor Network Gateway

- The hub for all incoming sensor data. It manages connections from acoustic sensors around the park. It filters audio data and forwards relevant detection to physical alarms.

Acoustic Sensors

- A network of field-deployed microphones is positioned throughout the park.
- Each sensor continuously listens for animal sounds and transmits detected noise data to the Sensor Network Gateway.

Alarm & Notification Service

- Responsible for alerting rangers when a possible mountain lion detection occurs.
- It receives alert signals from the ranger station computer and activates the Physical Alarm System

Physical Alarm System

- An on-site alarm mechanism triggered by the Alarm & Notification Service. It provides audio or visual alerts at the ranger station.

Reporting Module

- Generates detailed and summary reports of all detections and classifications.
- Reports assist rangers and administrators in tracking wildlife trends over time.

Classification & Admin Module

- Provides tools for rangers to classify detections and manage system settings.
- Rangers can mark alerts as definite, suspected, or false.

Database

- Stores all persistent data for the MLDS, including detailed detection logs, ranger classifications, and system configurations.

Rangers/Park Staff → Computer Interface

- User actions, control, and monitoring

Computer Interface → Sensor Network Gateway

- Receives detection data

Computer Interface → Alarm & Notification Service

- Triggers alerts

Computer Interface → Reporting Module

- Generates reports and maps

Computer Interface → Classification & Admin Module

- For ranger review and classification

Sensor Network Gateway → Acoustic Sensors

- Gathers raw sound data

Alarm & Notification Service → Physical Alarm System

- Activates local alarm

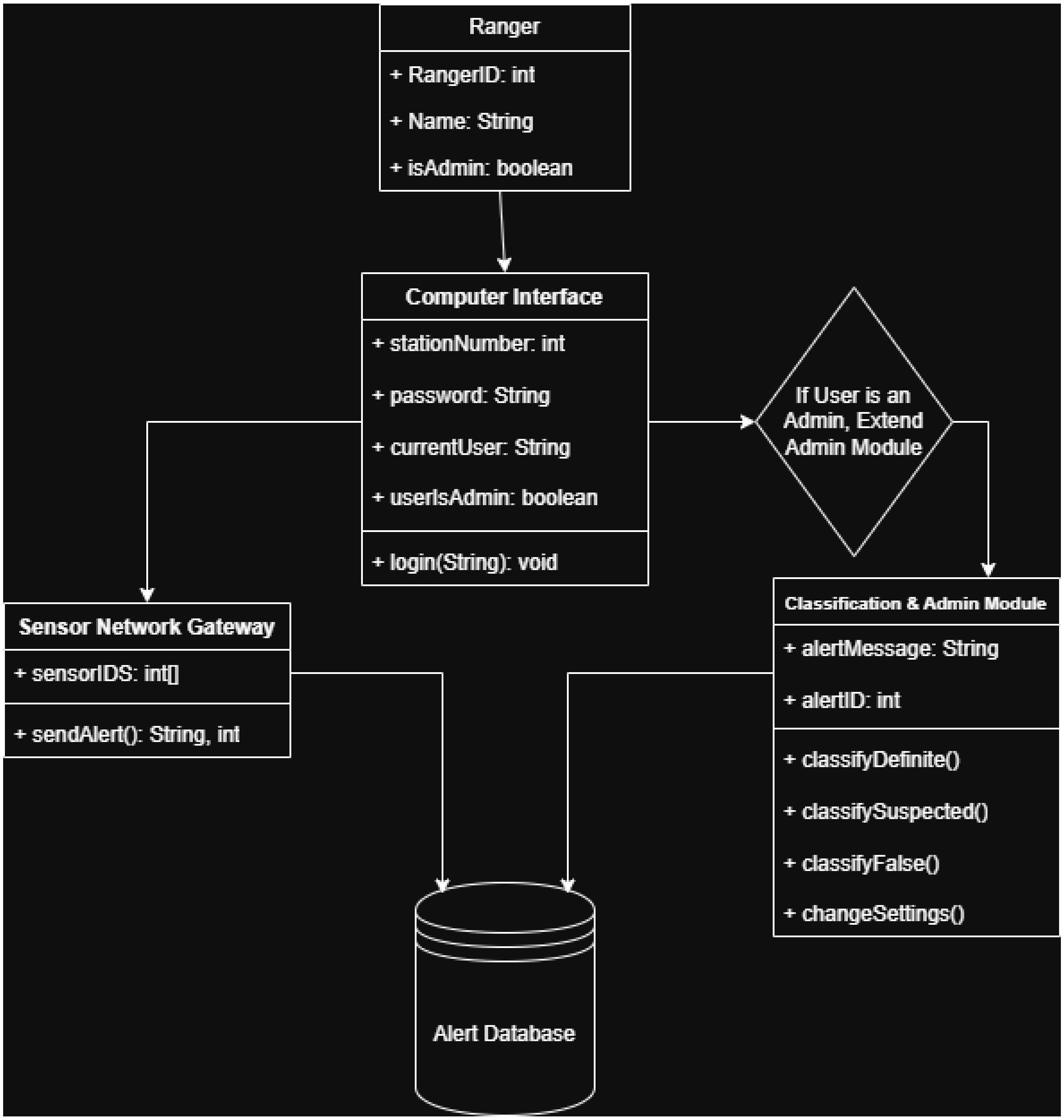
Reporting Module → Database

- Reads from stored data

Classification & Admin Module → Database

- Stores classifications

- UML Class Diagram:



- Description of classes - The main class is the Ranger class, which extends the Admin Module depending on if the specific ranger is an Admin. The Ranger class is meant as a foundation to access the other classes that define the detection system. For example, the different types are the Computer Interface, which is meant to be implemented in each computer in each park station. The Sensor Network Gateway tracks every sensor and their IDs throughout the park and sends an alert to the ranger and database with the specific sensor that caused the alert. Lastly, the Classification

& Admin Module allows admin rangers to change settings and classify alerts based on if they’re definite, suspected, or false.

- Description of attributes - The attributes listed in the Ranger class includes an int RangerID, a String Name, and a boolean isAdmin. The Computer Interface includes an int stationNumber, a String password, a String currentUser, and a boolean userIsAdmin. These attributes mainly describe the specific ranger accessing the computer interface, what station the computer is in, and if the ranger has admin authority. The Sensor Network Gateway has an array of integers that stores the sensorIDS, and the Admin Module contains alert messages and IDs when they are being set off.
- Description of operations - The Computer Interface Class has one operation, which is login(). The parameters for this operation is the RangerID and a preset password. Once a ranger is logged in, they have access to view the Alert Database which shows all of the alerts that went off in the past 30 days. If the ranger is an Admin, then they are able to classify alerts with the methods: classifyDefinite(), classifySuspected(), and classifyFalse(). They also have the ability to change settings with the changeSettings() methods, which would allow for events such as adding new sensors to the park’s system or removing one. The last operation that is present is in the Sensor Network Gateway which is the sendAlert() operation. This will automatically send alerts to the database and to rangers for classification when a sensor gets set off, when this happens it shows a detailed message of the alert and the sensorID that got set off.

Development Plan and Timeline

Partitioning of Tasks

- Requirements Review and Design Finalization (Week 1-2, from project start)
  - Review system requirements with the client.
  - Finalize architecture and communication between sensors and ranger station.
  - Plan the long term data storage and retention
- Backend Development (Week 3-5)

    - Build the sensor collection and alert processor software.
    - Add alarm and classification features.
    - Set up APIs for alert data and report generation.
- Frontend Development (Week 6-8)
    - Design the ranger system dashboard for lice alerts and alarm control.
    - Add classification tools and map-based visual reports.
    - Confirm all pieces smoothly connect with the backend.
- Testing and Integration (Weeks 9-10)
    - Test the system with both simulated and real data
    - Check alarms, data logging, and reports.
    - Fix any bugs or timing issues found.
- Deployment and Documentation (Weeks 11-12)
    - Install the finished system at ranger station.
    - Ensure employees are family with software and how to use it.
    - Write and finalize user and maintenance documentation.

Team Member Responsibilities

- Madeline Cohen- Project coordination, requirements review, documentation, task scheduling, and timeline management
- Josh Sindelar- Software architecture design, backend API development, and database integration.
- Spencer Van Rossum- UML class diagram creation, class and method documentation, and frontend implementation.

Data Management Strategy

Data Base Choice

- Our system uses a SQL database to store and organize all park data.
- SQL is reliable, easy to manage, and works well for structured information like alerts, detections, and user accounts.

- Each park will have its own database to keep information organized and secure.

Data Stored

- The database will keep track of detections, alerts, summaries, sensors, stations, and user accounts
- Detailed data is saved for 30 days, and summarized data is stored for one year to track long term trends.
- Audio recordings will be stored separately, with the database only storing a link to access them.

Security and Backup

- All data is encrypted during storage and transfer.
- Rangers and admin have different access levels for safety and privacy.
- All user actions are logged.
- Regular backups are performed to prevent data loss and ensure recovery.