Fig. 1 SETU logo

Software Development Year 4 Project

Functional Specifications

Project Title: SiteSafe AI

By: Tomas Smitas
Student No: C00276177
Date: 2024
Supervisor: Oisin Cawley

Fig. 2 Site Safe Logo

Introduction

This is a functional specification for a proposed object detection product demonstration that will use Machine Learning technology to detect people, high visibility jackets and helmets on safety critical construction sites. Once a person is detected to not have either a helmet or a high visibility jacket on a safety critical site an appropriate person will be automatically notified with an incident report through an email and/or phone message.

The key idea of this functional specification is to outline the path needed to follow to complete this project. It will include technologies used and why that technology was used, product overview that will describe in detail what the project is about, Mandatory, Discretionary and Exceptional features that need to be completed to reach each milestone.

Technologies

For the technology section extensive research has been carried out to try and predict the best technologies to use for this project. Most of these technologies that have been chosen to use have similar alternatives. I will outline these technologies and why they were chosen over their counterparts.

The main technologies that will be used are:

1. **Python**

Fig. 3 Pyton logo

Description: Python is a programming language that is being used because it has great support for machine learning and AI both of which are going to be implemented in the project. Has great integration with other technologies that will be mentioned. Easy language to understand and develop with. Has great real-time data handling which will be crucial in my project.

Alternative: R is a programming language that is also commonly used in machine learning projects. The main reason for not using this language over python was because it didn't have as much integration with other technologies like OpenCV and You Only Look Once.

2. YOLOv10

Description: You Only Look Once is a real-time object detection system that identifies and classifies multiple objects in images or video frames in a single pass, hence the name. Unlike traditional object detection methods that apply a classifier to various parts of an image, YOLO divides the image into a grid and predicting bounding boxes and class probabilities directly from the grid cells. This approach enables YOLO to achieve high speeds while maintaining good accuracy, making it particularly suitable for this project that requires real-time processing for the video surveillance.

Alternative: Real-Time DEtection TRansformer (RT-DETR) unlike YOLO this object detection tool is not designed for real-time object detection, there might have been issues implementing this with live video surveillance. YOLO provides detection in a single pass while RT-DETR uses the two-stage approach which makes the latency higher as seen in the figure below. The Figure below shows COCO AP (%) which is a standard object detection testing tool showing the load and on the y axis and shows the Latency (ms) of the ojects detection on the x axis. This figure gives a statistical reason why YOLOv10 was chosen.

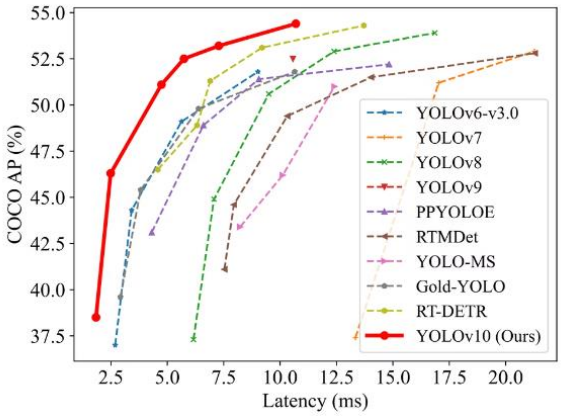


Fig. 4 YOLO model comparison

3. Ultralytics

Description: Ultralytics is a leading platform for real-time object detection tasks, optimized to work seamlessly with the YOLO (You Only Look Once) models. Known for its powerful, high-performance capabilities, Ultralytics enables efficient detection and classification of objects in both images and video streams, making it ideal for real-time applications like monitoring construction sites. The Ultralytics implementation of YOLO is designed for ease of use, providing a flexible, lightweight solution that supports CPU and GPU computations, making it a reliable choice for applications requiring quick deployment and rapid processing.

Alternative: Initially, I considered using Darknet for my project due to its strong integration with the YOLO family, optimised for real-time object detection. Darknet's lightweight structure and GPU support made it suitable for fast, accurate detection tasks, especially for real-time monitoring applications like those on construction sites. However, I wanted to use the latest version of YOLO implementation, which is YOLO11, the latest versions of YOLO implementation were not provided by Darknet. I later decided to switch to Ultralytics because it offered YOLO11 and a more flexible and user-friendly implementation of YOLO. Ultralytics is designed for ease of deployment and quick adaptation across platforms, making it an excellent choice for real-time, practical applications.

4. **OpenCV**

Fig. 5 OpenCV logo

Description: OpenCV (Open Computer Vision Library) is a powerful, open-source library designed for real-time computer vision tasks, such as image processing and video capture. It provides tools for working with images, videos, and machine learning which is perfect for this project. It can be easily integrated with YOLO and OpenCV. YOLO detects objects, OpenCV can be used to draw bounding boxes, labels, and other visual markers on the images or videos to highlight detected objects (like helmets and high-vis jackets). Together, they create a complete pipeline for real-time video processing and detection, ideal for the construction site monitoring project.

Alternative: This section had a clear winner. There were other alternative but none that had as many benefits that OpenCV provided. Pillow was an alternative looked at, but it lacked the community support that OpenCV has.

5. **Flask**

Fig. 6 Flask logo

Description: Flask is a lightweight and flexible web framework for Python that helps developers create web applications and APIs. It is simple to use and doesn't come with many built-in features, making it easy to scale by adding extensions as needed. This would be perfect for the backend. This backend may process the data to implement the object detection. It will be used to transfer the data to the frontend of the react website.

Alternative: FastAPI is also a backend framework that has high speed that would have been a good choice for my live monitoring project. The speed and efficiency of this backend did not however overtake all of the other advantages that Flask provides. FastAPI doesn’t have as big of a community support as flash. Flask is very minimalistic and straight forward making it easy to use which is a massive advantage for any technology.

6. **React**

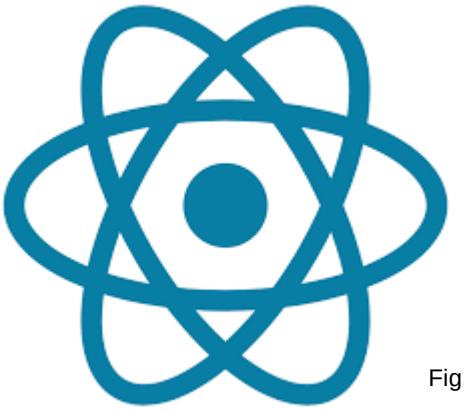


Fig. 7 react logo

Description: React is a popular JavaScript library for building user interfaces, particularly for single-page applications. It allows developers to create reusable UI components, manage application state effectively, and create dynamic, interactive web applications. This should be perfect for this project making the website interactive and responsive.

Alternative: Just using JavaScript without react was a consideration to make the frontend as simple as possible. But after considering all the benefits of using React. The decision was clear that react would make the website much better.

7. **Simple Mail Transfer Protocol (SMTP)**

Fig. 8 Email logo

Description: SMTP (Simple Mail Transfer Protocol) is a protocol used to send and receive emails over the Internet. It allows email clients to send messages to servers, which then deliver them to the recipient's server. SMTP is essential for sending emails and often works alongside other protocols for managing incoming messages.

Alternative: IMAP (Internet Message Access Protocol): Used for retrieving and managing incoming emails, allowing multiple devices to access the same mailbox was the alternative considered. Unlike the other technologies where I was a clear advantage of one over another. In this case they seemed to be similar achieving the same. One small advantage and why SMTP was chosen was that it is slightly more used and has bigger community support for it.

8. **MySQL**

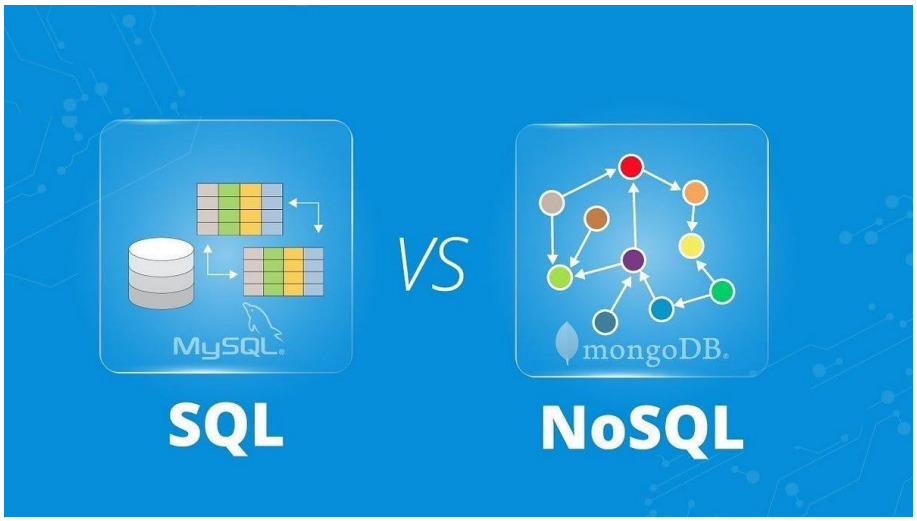


Fig. 9 Sql vs NoSql

Description: MySQL is an open-source relational database widely used for web applications. It is known for its reliability, scalability, and ease of use, supporting complex queries and large datasets. MySQL works well with Flask, making it a popular choice for dynamic websites and applications.

Alternative: MongoDB was considered as an alternative. As a NoSQL database, MongoDB offers flexibility with its unstructured data model, which could have been beneficial. However, after reviewing my requirements, I concluded that MySQL's structured approach would better support my data needs, making it easier to store and retrieve information.

Description

Problem Statement

In the construction industry, ensuring the safety of workers is of paramount importance. However, non-compliance with safety regulations, such as wearing helmets and high visibility jackets, poses significant risks to workers health and safety. Traditional methods of monitoring compliance are often labour-intensive, prone to human error, and can lead to delays in addressing safety issues.

To address these challenges, this project aims to develop an AI-driven monitoring system that utilises computer vision and machine learning to automatically detect whether construction workers are wearing the required safety gear in real-time. The system will analyse live video feeds from surveillance cameras installed on-site and provide immediate alerts to site managers when safety violations occur. Additionally, it will log incidents for future reference and compliance reporting.

By implementing this solution, the project seeks to improve safety compliance, reduce the likelihood of accidents, and enhance overall workplace safety in the construction industry.

Background

My motivation for developing the AI construction site monitoring project rises from my internship experience with Netwatch, a leading monitoring security company. During my time there, I witnessed firsthand the critical role that real-time surveillance plays in enhancing safety and security across various industries. This experience built my passion for applying advanced technologies to solve real-world problems, particularly in the live-monitoring sector.

In discussions with safety personnel and site managers, I learned about the persistent challenges they face in ensuring compliance with safety regulations. Many expressed that implementing an automated monitoring system to track worker safety gear would be a game-changer for their operations. Their enthusiasm and support for the idea reinforced my belief that this project could significantly improve workplace safety and efficiency.

I am driven to leverage my skills in AI and machine learning to create a solution that not only addresses these challenges but also fosters a culture of safety within construction environments. By integrating intelligent monitoring capabilities, I aim to contribute to safer work practices and ultimately protect the well-being of construction workers.

Core Features/Requirements

1. Real-time Object Detection and Classification:

- The system must detect construction workers in live video streams and accurately classify whether they are wearing required safety gear, such as helmets and high-visibility jackets, using YOLO for object detection.

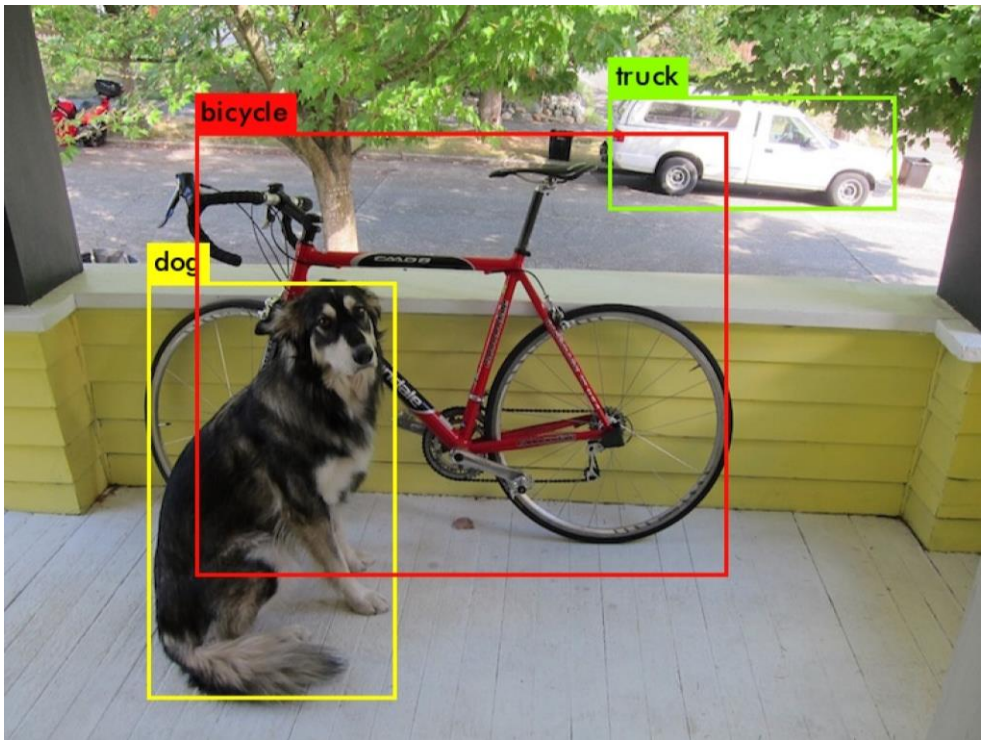


Fig. 10 YOLO detection

2. Automated Alert System:

- If a worker is detected without proper safety gear, the system must automatically trigger an alert via email (using SMTP) to the designated safety personnel.

Fig. 8 Email logo

Functional Requirements

1. **Live Video Feed Processing**
   - The system shall be capable of processing live video feeds from a camera connected to the user's device.
   - The system shall support various video formats and resolutions.

2. **Safety Gear Detection**
   - The system shall detect whether construction workers are wearing helmets and high-visibility jackets in real time.
   - The system shall utilize the machine learning model YOLO for object detection.

3. **Alert Generation**
   - The system will automatically generate alerts when a worker is not wearing the required safety gear for longer than 30 frames.
   - The alerts shall be sent to the site manager/safety officer via a notification system (email, SMS).
   - There shall be a delay of notifications for the same person of 1 hour.

4. **Incident Logging**
   - The system shall log all detected incidents of non-compliance, including timestamps, snapshot image and camera location.
   - The logs will be stored in a SQL database for future reference and reporting.

5. **User Interface**
   - The system shall provide a web-based user interface in React for demonstrating the product.
   - The interface shall allow users to view historical logs and reports of compliance incidents.
   - The user should have options to disable/enable email notification, SMS notification, screen alert.
   - An ability to change the receivers email/phone number.

6. **Reporting**
   - The incident logging shall be exportable in common format like CSV.

Non-Functional Requirements

1. **Performance**
   - The system shall process live video feeds and detect safety gear with a latency of no more than 2 seconds.
   - The system shall support different types of camera brands.

2. **Scalability**
   - The system shall be able to scale to support an increase in the number of cameras and users without requiring major architectural changes.
   - The system shall accommodate a growing database size with a minimum of 1,000 logged incidents without performance impact.

3. **Availability**
   - The system shall ensure an uptime of 99.9%, with minimal downtime for maintenance and updates.

4. **Security**
   - The system shall implement secure user authentication and authorization mechanisms to prevent unauthorized access.
   - All data transmissions between the cameras, servers, and user interface shall be encrypted using industry-standard protocols (HTTPS, SSL/TLS).

5. **Usability**
   - The user interface shall be intuitive and user-friendly, allowing site managers to easily navigate and monitor safety compliance.
   - The system shall provide clear and informative alerts that are easy to understand and act upon.

6. **Compatibility**
   - The system shall be compatible with standard web browsers (Chrome, Firefox, Safari).
   - The system shall support integration with existing construction management software through standard APIs.

7. **Reliability**
   - The system shall recover from failures automatically and continue to operate without loss of data or functionality.
   - The system shall have mechanisms to back up data regularly to prevent loss in case of system failure.

User (Construction Site Manager/ Site Safety Officer)

The construction site manager/ site safety officer oversees worker safety and compliance on-site. With the AI monitoring system, they will automatically receive real-time alerts via email or messages, notifying them if workers are not wearing helmets or high-visibility jackets. This allows managers to take immediate corrective actions, improving safety protocols efficiently.

Product Target (Construction Site Monitoring Companies)

The AI-based monitoring product is designed for companies that specialise in providing safety and surveillance services to construction sites. It enhances their existing offerings by incorporating live video analysis that automates the detection of safety gear compliance, reducing human oversight and improving overall site safety. A lot of these companies already have AI object detection for people, this product is designed to enhance the ability of that software to also detect helmets and high visibility jackets.

Risks

1. Accuracy of Object Detection

- **Risk**: The YOLO object detection model may not accurately detect helmets or high-visibility jackets in various lighting conditions, angles, or obstructions on the construction site.
- **Impact**: False positives or false negatives could lead to missed safety violations or unnecessary alerts, reducing the system's reliability.
- **Mitigation**: Continuously fine-tune and test the model with diverse datasets and real-world construction site footage to improve detection accuracy.

2. Latency in Real-Time Alerts

- **Risk**: The system may experience delays in detecting and sending alerts due to high video processing loads or slow network speeds.
- **Impact**: Delayed alerts reduce the effectiveness of real-time safety monitoring, as issues might not be addressed immediately.
- **Mitigation**: Optimize the video processing pipeline, ensure sufficient network bandwidth, and use cloud-based infrastructure for scalable performance.

3. Data Privacy and Security

- **Risk**: Video footage and personal information could be accessed or compromised if not properly secured.
- **Impact**: Breach of privacy regulations (such as GDPR) and potential legal repercussions for failing to protect sensitive data.
- **Mitigation**: Implement end-to-end encryption, secure storage solutions, and strict access controls. Ensure compliance with data privacy regulations.

4. System Downtime or Failure

- **Risk**: Unexpected system crashes or downtime due to hardware, software, or network failures could interrupt monitoring.
- **Impact**: Lack of real-time monitoring during downtime could result in missed safety violations, affecting site safety and trust in the product.
- **Mitigation**: Set up redundancy measures, backup systems, and regular maintenance schedules. Implement automated health checks and failover mechanisms.

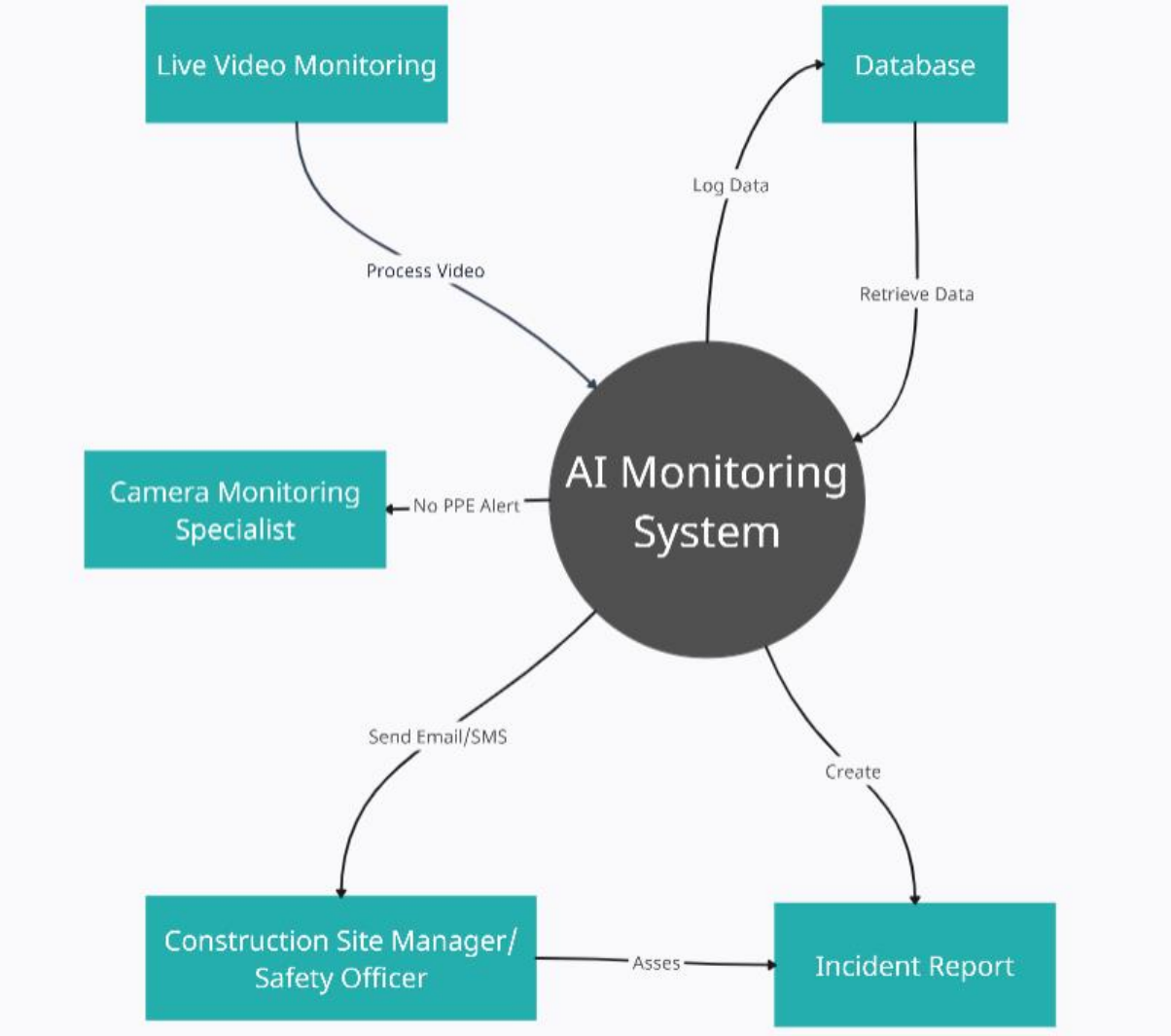
5. User Adoption and Training

- **Risk**: Site managers and monitoring companies may be reluctant to adopt the system if they find it difficult to use or integrate into their existing workflows.
- **Impact**: Low user adoption rates could result in the product not being fully utilized, diminishing its value to the company.
- **Mitigation**: Provide a user-friendly interface, detailed onboarding, training materials, and customer support to ease the transition to the new system.

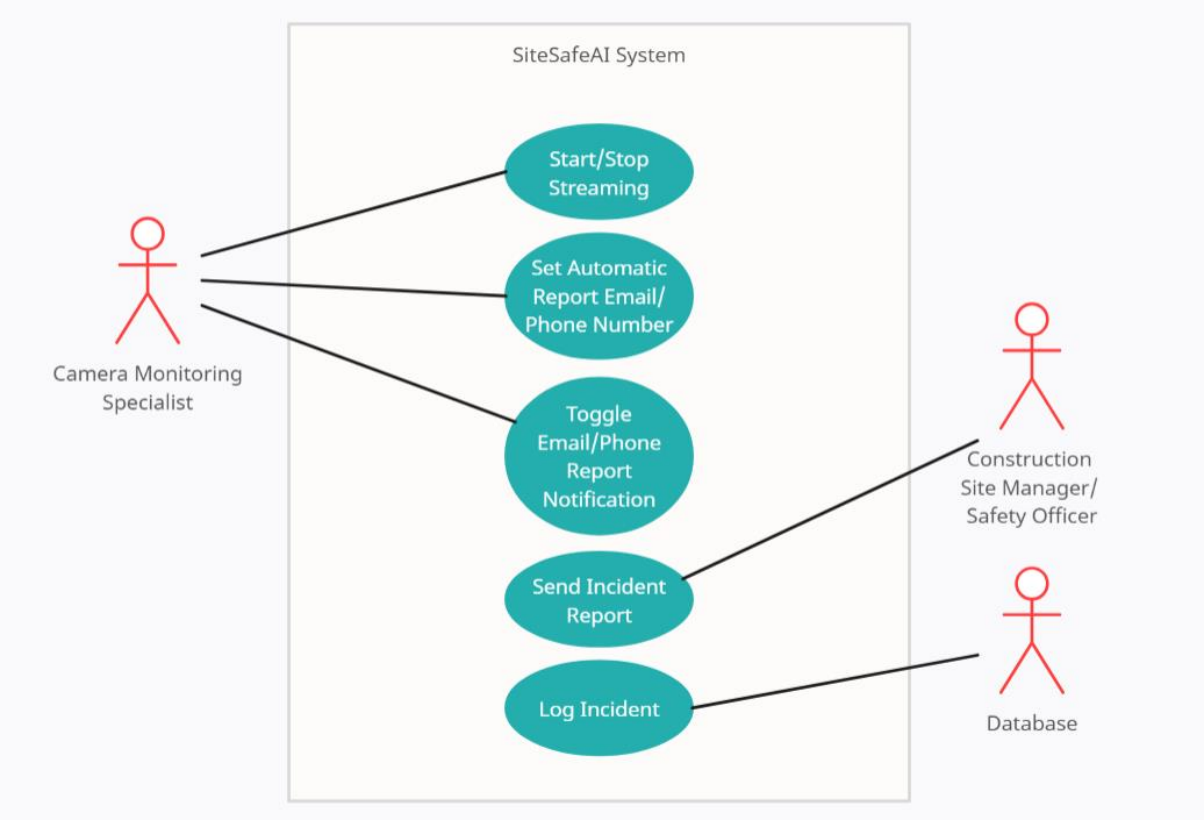
6. Integration Challenges

- **Risk**: Difficulty integrating the system with existing software or monitoring solutions used by construction companies.
- **Impact**: Integration issues may cause inefficiencies or prevent full utilization of the system.
- **Mitigation**: Ensure open APIs and modular design that allow easy integration with third-party systems. Work closely with clients during implementation to address compatibility concerns.

Context Diagram



Use Case Diagram



Use Case:	Turn on AI detection with Email and Phone number alerts enabled
Actor(s):	Camera Monitoring
Description:	Camera Monitoring personnel turns on live monitoring camera to start detecting for no helmets or high visibility jackets on a construction site
Steps:	1. **Navigate to Website**: The Camera Monitoring personnel accesses the AI project's monitoring website. 2. **Enter Contact Information**: The user inputs their email address and phone number into the required fields. 3. **Enable Alerts**: The user toggles on email and SMS alerts to receive notifications. 4. **Start Video Stream**: The user initiates the live video stream by pressing the "Start Stream" button.

References

Figures

Fig 1. SETU Logo: SETU Logo

Fig 2. SiteSafe Logo: SiteSafe Logo

Fig 3. Python Logo: Python Logo

Fig 4. YOLO Docs: YOLO Documentation

Fig 5. OpenCV Logo: OpenCV Logo

Fig 6. Flask Logo: Flask Logo

Fig 7. React Logo: React Logo

Fig 8. Email Logo: Email Logo

Fig 9. SQL vs NoSQL: MySQL vs NoSQL

Fig 10. YOLO object detection example: YOLO Object Detection Example

General Information

Construction site high visibility jacket standards: High Visibility Clothing Standards

R docs: R Documentation

RTDETR docs: RTDETR Documentation

Darknet docs: Darknet Documentation

PyTorch docs: PyTorch Documentation

SMTP wiki: SMTP Wiki

Context Diagram Creately: Creately Context Diagram

Ultralytics docs: Ultralytics