

SafetyEye – AI-Powered Workplace Occupancy & Safety Monitor

Project Statement:

Develop an AI-based system that uses video surveillance feeds to monitor workplace occupancy levels and detect safety compliance. The platform will help office and industrial space managers improve space utilization and ensure employees follow safety protocols (e.g., wearing helmets, vest and etc).

Outcomes:

Identify safety compliance violations such as missing helmets etc

Generate visualizations and alerts to improve safety

Present results via a real-time dashboard for administrators

Dataset Sources:

[Construction Site Safety Image Dataset Roboflow] -

(<https://www.kaggle.com/datasets/snehilsanyal/construction-site-safety-image-dataset-roboflow>)

Modules:

1. **Data Prep** - Load & process YOLO-formatted images
2. **Model Train** - Train YOLOv8 for PPE detection
3. **Detection** - Real-time PPE violation spotting
4. **Alerts** - Send notifications for missing gear
5. **Dashboard** - Live view + compliance stats

Milestone 1 (Week 1–2): Data Preparation & Environment Setup

Goals:

- Define the complete project architecture.
- Prepare dataset for training.
- Set up development and model training environments.

Tasks:

- Finalize requirements, safety rules (e.g., missing helmet = violation).
- Download and explore [Construction Site Safety Image Dataset](#).
- Preprocess and format data into YOLOv8-compatible format.
- Split dataset into training, validation, and test sets.
- Set up YOLOv8 training environment (e.g., with Ultralytics).
- Install necessary libraries (PyTorch, OpenCV, etc.).

Deliverables:

- Cleaned and organized dataset.
- Working development environment.
- Documented data preprocessing pipeline.

Milestone 2 (Week 3–4): Model Training & PPE Detection

Goals:

- Train and fine-tune a YOLOv8 model to detect safety equipment (PPE).
- Evaluate model accuracy and refine detection results.

Tasks:

- Train initial YOLOv8 model using the prepared dataset.
- Validate and test the model.
- Perform hyperparameter tuning for improved accuracy.
- Use data augmentation techniques to enhance robustness.
- Evaluate model using metrics like mAP, precision, recall.
- Visualize predictions to confirm correct detection of helmets, vests, etc.

Deliverables:

- Trained and validated PPE detection model.
- Evaluation metrics report.
- Sample detections and visual results.

Milestone 3 (Week 5–6): Real-Time Detection & Alert System

Goals:

- Implement real-time video processing pipeline.
- Add logic for detecting safety violations and sending alerts.

Tasks:

- Set up video stream input (webcam or surveillance footage).
- Implement real-time inference using trained YOLOv8 model.
- Add detection overlay (bounding boxes + labels).
- Build logic to detect violations (e.g., no helmet).
- Integrate an alert system (email, notification, console log, or GUI warning).

Deliverables:

- Real-time detection system.
- Working violation rule engine.
- Functional alerting mechanism.

Milestone 4 (Week 7–8): Dashboard Development & Final Integration

Goals:

- Build and deploy a real-time dashboard for monitoring and analytics.
- Finalize project with integration, optimization, and documentation.

Tasks:

- Design and build dashboard (e.g., using Streamlit, Dash, or Flask + React).
- Display live video feed with overlays.
- Show compliance statistics (charts, logs).
- Store and retrieve violation logs.
- Conduct full system tests (accuracy, latency).
- Create final presentation/report/demo video.

Deliverables:

- Live dashboard with analytics and alerts.
- Final project report and source code.
- Working demo or walkthrough video.

Architecture Diagram:

