🚧 Executive Summary:

SafeOps AI – Intelligent Safety Violation Monitoring System

SafeOps AI is a modular, end-to-end AI-driven platform engineered for monitoring and enforcing safety compliance in high-risk environments like mining and construction sites. It integrates **real-time computer vision using YOLOv8**, **image processing via OpenCV**, **reasoning through LLMs (Mistral 7B)**, and a **Streamlit-based dashboard** for safety intelligence, decision support, and incident management.

The system detects safety violations, reasons about their risk level contextually, generates action plans, escalates critical issues via email, and visualizes everything in a clean dashboard interface.

🧱 Architecture Overview

1. Computer Vision Detection (YOLOv8 + OpenCV):

- **YOLOv8 nano** (Ultralytics) model is used to perform **object detection** on each frame.
- Detects safety-critical classes: person, helmet, vest, excavator, etc.
- Uses **OpenCV** for:
  - Loading raw images
  - Annotating detections with bounding boxes
  - Saving annotated frames for review
- Output saved in data/annotated_frames/ for visualization.

2. Safety Rule-Based Notification System:

- Each frame is parsed for violations like:
  - no_helmet
  - no_safety_vest
  - trip_hazard
  - fatigue_posture
  - too_close_to_excavator

    - obstructed_exit
    - unsafe_manual_handling
- Violations are matched against a rulebook and logged.
- Critical violations trigger **real-time email alerts** via Gmail SMTP using yagmail.

3. LLM-Powered Reasoning & Planning (Mistral-7B):

- Uses a **locally hosted LLM (Mistral-7B via LM Studio)** to:
    - Evaluate risk and decide if escalation is needed (ask_llm_reasoning)
    - Generate a step-by-step action plan for supervisors (ask_llm_action_plan)
    - Recommend training or policy improvements to avoid repeat incidents (ask_llm_policy_recommendation)
- Responses are stored in:
    - llm_logs.txt
    - action_plan.txt
    - policy_recommendations.txt

4. Pipeline Orchestration (pipeline.py):

- Sequentially:
    - Reads detection output JSON
    - Handles per-violation alerting and logging
    - Calls all LLM modules
    - Saves results to logs
    - Can be extended to batch or streaming inputs

5. Interactive Dashboard (streamlit_app.py):

- Built with **Streamlit**, the dashboard displays:
    - All annotated frames from data/annotated_frames/
    - Detection summaries
    - LLM escalation reasoning
    - Action plans and policy suggestions
- Ideal for safety officers and compliance managers to audit results in real-time.

🛠️ Tech Stack

Component	Tool / Library
CV Detection	YOLOv8 (Ultralytics)
Image Processing	OpenCV
LLM Reasoning	Mistral-7B via LM Studio API
UI Dashboard	Streamlit
Notification Engine	Yagmail (Gmail SMTP)
Environment	Conda (safeops_ai_env)
Language	Python 3.10

🚨 Alert & Escalation Logic

- Rule-matched violations are evaluated using the LLM.
- If escalate: true, email alert is sent with a risk summary.
- Escalation logic considers:
  - Violation type
  - Location (e.g., near blast zone)
  - Time of shift (e.g., night shift)
- Sample LLM Output:

```
{
  "escalate": true,
  "notify_roles": ["Safety Officer", "Supervisor"],
  "shutdown_required": true,
  "summary": "Two workers are not wearing helmets near blast zone during day shift"
}
```

📁 Folder & File Summary

```
safeops_ai/
├── vision/          # Detection logic (dummy + YOLOv8)
├── agents/          # LLM + alerting logic
├── utils/          # Logging utilities
├── data/           # Input images & outputs
```

```
│   ├── dummy_frames/
│   ├── annotated_frames/
│   └── dummy_detections.json
├── streamlit_app.py     # Dashboard interface
├── pipeline.py          # Main orchestration script
├── requirements.txt
├── README.md
```

🧪 Deployment & Usage

```
# Setup environment
conda create -n safeops_ai_env python=3.10
conda activate safeops_ai_env
pip install -r requirements.txt

# Run detection (YOLOv8)
./safeops_ai_env/bin/python vision/yolo_detector.py

# Run pipeline
./safeops_ai_env/bin/python pipeline.py

# Launch dashboard
./safeops_ai_env/bin/streamlit run streamlit_app.py
```

Note: LM Studio must be running at http://192.168.0.14:1234.

🌱 Future Enhancements

- 🔴 Real-time live video stream integration via OpenCV or RTSP
- 🧠 Fine-tuned YOLOv8 models on PPE-specific datasets
- 📈 Historical dashboard with trends and compliance metrics
- 🔐 User login & multi-role access in dashboard