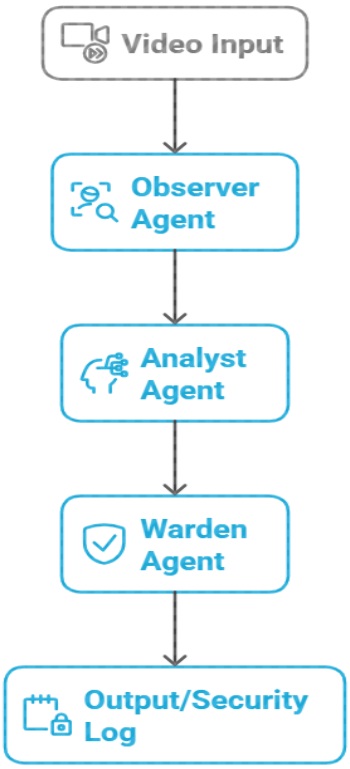
An Agentic AI Framework for Real-Time Crowd Management

A. Executive Summary

This project is all about a high-performance surveillance system for automated occupancy tracking is called Sentinel-AI. The system offers autonomous vision, cognitive reasoning, and emergency response by utilizing a Multi-Agent AI Architecture driven by the YOLOv8 model. Multi-agent modeling is the most reliable method for controlling crowd dynamics in crowded settings, according to research. **[2]**.

B. System Architecture & Agent Design

The "Assembly Line" architecture, sometimes referred to as the Sequential Pipeline Pattern, is the foundation upon which the system is constructed **[6]**.We can now see this architecture divides the procedure into distinct sections, in contrast to conventional monolithic scripts.



a The Observer (Perception Agent)

Here in the sensory input is provided by the ObserverAgent. The YOLOv8 (You Only Look Once) architecture is used **[1]**.

Function: It ensures high frame rates by segmenting video frames into grid cells and simultaneously predicting object bounding boxes **[5]**.

Specialisation: It removes background noise to concentrate only on person density by filtering to Class 0 (Person).

b The Analyst (Cognitive Agent)

Now the AnalystAgent is the brain of the system. It uses a safety heuristic depending on your configured capacity after receiving the raw integer count from the Observer.

Warning Buffer: This method, which has been proven to be effective for proactive event safety, determines an 80% threshold to deliver a "Yellow Alert" phase **[6]**.

c The Warden (Action Agent)

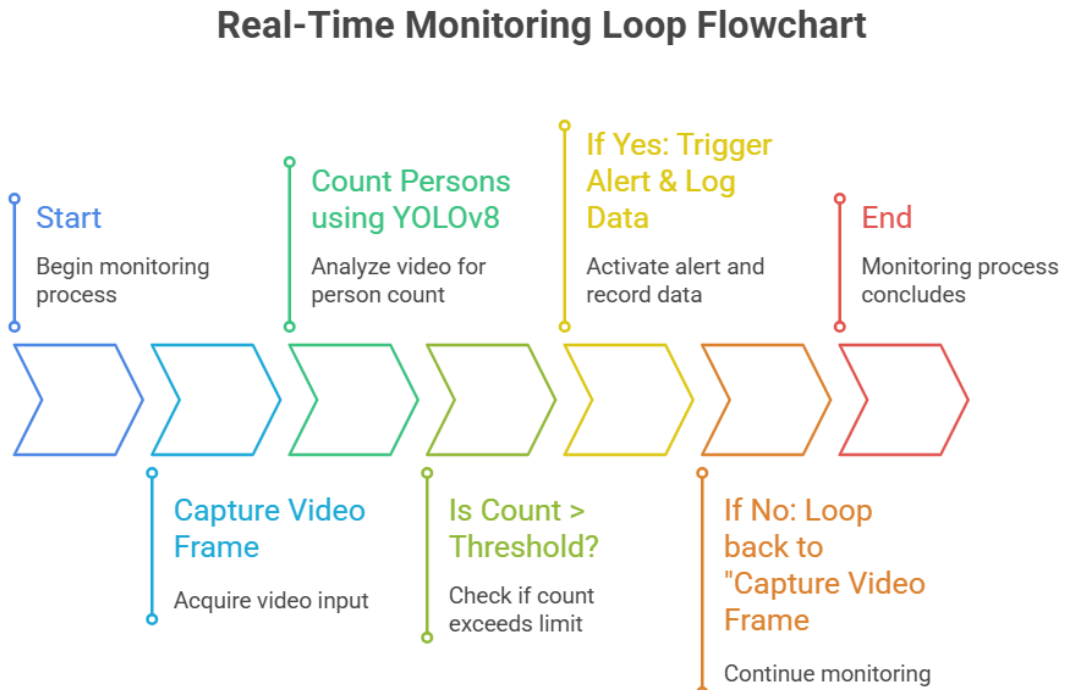
For which the "Response" phase is overseen by the WardenAgent.

Execution: If the status is "CRITICAL," a timestamped entry is recorded in temple_security_logs.txt and a hardware-level beep is triggered. This guarantees an ongoing security intelligence audit trail **[7]**.

C. Functional Logic Flow

The data follows a deterministic, linear path that makes debugging and verification simple. **[6]**:

1. **Capture:** The camera's frame is taken.
2. **Conclusion:** YOLOv8 returns coordinates after detecting people.
3. **Reasoning:** The threshold T is compared to the count.
4. **Action:** The Warden logs data and sends out alarms if Count T.



D. Code Explanation & User Guide

a. How the Code Works

- First in the model the script makes use of the lightest version of YOLOv8, yolov8n.pt (Nano). It is selected because of its "speed-first" methodology, which enables real-time processing on laptops without a dedicated GPU **[4].**
- **Visuals:** Then the outcome.The detection boxes are automatically drawn using the plot() method, and the "Danger" warnings are directly superimposed on the live video via cv2.putText.
- The ctypes.windll.user32.Message is safe.A low-level system call is beep. Even if the software-based music players on your computer are muted, it still functions.

b. How to Use the System

- **Environment Setup:** Ensure you have Python installed. Install dependencies using: `pip install ultralytics opencv-python`
- **Calibration:** Open the script and change `threshold=10` in the `main()` function to your venue's capacity (e.g., `threshold=500`).
- **Execution:** Run the script with `python filename.py`.
- **Monitoring:**

    - **Green Text:** Normal status.
    - **Red Text + Sound:** Critical Overcrowding.
- **Logs:** Access the `temple_security_logs.txt` file in your project folder to view a history of all overcrowding incidents.

E. Mathematical Logic

The system state S is determined by:

Occupancy Count (c)	**Security Status (S)**	**System Action**
c < 0.8T	**NORMAL**	Display Green UI
0.8T =< c < T	**WARNING**	Display Yellow UI
c >= T	**CRITICAL**	**Trigger Alarm & Log Alert**

Where c is the current count and T is the user-defined threshold.

6. Technical Challenges & Mitigations

Computer vision systems encounter environmental challenges in real-world deployments. Your research gains credibility when you acknowledge these.

- Obscurity & Density: People overlap in very thick crowds, making it challenging for the observer to distinguish between different bounding boxes.

  Mitigation: When individual detection fails, a "heat-check" can be provided by using a Crowd Density Map (pixel-wise estimation) in conjunction with YOLOv8.

- Variable Lighting: Low light or shadows, which are typical during evening temple ceremonies, can lower detection confidence.

Mitigation: Using infrared-capable cameras or Auto-Exposure algorithms for round-the-clock surveillance.

- Perspective Distortion: People in the back appear smaller than those in the front when cameras are positioned at low angles.

  Mitigation: Establishing a Region of Interest (ROI) and using a "perspective scale factor" to adjust counts according to distance.

F. Ethical Considerations & Privacy

Privacy is a major worry because this device keeps an eye on people in a sensitive setting (a temple) **[3]**.

- **Data Anonymization:** Without saving identifiable facial information, the system should be built to process frames in real-time.
- **Consent & Transparency:** Visitors should be made aware by signage that AI-based occupancy tracking is used for their protection rather than for personal monitoring.
- **Non-Invasive Monitoring:** YOLOv8 person detection recognizes "bodies" and "heads," protecting pilgrims' privacy in contrast to facial recognition.

G. Proposed Enhancements (Future Scope)

The following characteristics are advised in order to convert Sentinel-AI from a stand-alone script into a "Smart Infrastructure" component:

1. **Multi-Camera Fusion:** Creating a comprehensive "Live Occupancy" dashboard by combining feeds from multiple entry and exit points.
2. **Using past log data** to forecast "Peak Hours" (such as Friday prayers or festival days) and notify workers before the crowd comes is known as predictive surge analysis.
3. **Cloud-Integrated Warden:** Sending automated WhatsApp/SMS alerts to temple administrators by connecting the Warden agent to a cloud provider (such as AWS or Twilio) **[7]**.

H. Conclusion

So to conclude an Agentic AI Framework may successfully close the gap between intricate deep learning and useful safety applications, as Sentinel-AI shows. So therefore the system provides a high level of dependability and real-time responsiveness by assigning jobs to specialized agents. So hereby guaranteeing that technology fulfills the basic human need for safety, this strategy establishes a standard for the digital transition of conventional high-traffic location.

I. References

[1] Jocher, G., Chaurasia, A., & Qiu, J. (2023). *YOLO by Ultralytics (Version 8.0.0)*. [Software]. Available at: https://github.com/ultralytics/ultralytics .

[2] Gong, X., Herty, M., Piccoli, B., & Visconti, G. (2023). Crowd dynamics: Modeling and control of multiagent systems. *Annual Review of Control, Robotics, and Autonomous Systems*, *6*(1), 261-282.

[3] Information Commissioner's Office (2024). *Video surveillance and data protection guidance: Maintaining privacy in public occupancy tracking*. [Online] Available at:

[4] Ultralytics Documentation (2024). *YOLOv8 Modes and Model Comparisons: Speed vs. Accuracy for Real-Time Edge Computing*. [Online] Available at:

[5] Terven, J., & Cordova-Esparza, D. (2023). *A Comprehensive Review of YOLO: From YOLOv1 to YOLOv8 and Beyond*. arXiv preprint arXiv:2304.00501.

[6] Microsoft Azure Architecture Center (2025). *Software Design Patterns: Implementing Sequential Pipelines in AI Agentic Workflows*.

[7] Amazon Web Services (2025). *Designing Agentic Workflows: Implementing Observability and Automated Alerting via Warden Nodes*.