VisionNext: An AI-Powered Video Monitoring Platform for Real-Time Detection and Automated Event Management

Multi-Source Video Stream Analysis using VLMs, YOLO and SAM 3

'Krishna Sahu'

Department of Computer Science, Patkar-Varde College, University of Mumbai

---

Abstract—Advanced video monitoring systems may include many cameras, sources of videos, AI models, and the notification systems. Working with them separately may complicate and prolong the process of monitoring. VisionNext is the AI platform for video monitoring that integrates all these processes in one system. It allows its users to connect multiple live video sources via RTSP, RTMP, and mobile/OBS streaming and analyzes them with the help of Vision Language Models (VLMs). There are multiple standard cases of detection in VisionNext, which include fire detection, PPE detection, intrusion detection, and safety monitoring. Users can also configure their own detection requests via custom prompts. YOLO and SAM 3 models can be used for solving some specific computer vision problems such as object detection, segmentation, and tracking. Once any configured event occurs, the platform records such information as camera source, type of the event, time stamp, and image frame, and then sends email and phone call notifications to the necessary users. Also, there is role-based access control to the cameras, events, and other platform features. VisionNext platform is developed with FastAPI, MongoDB, Redis, and Docker for backend services, database, processing, and deployment.

Keywords—*AI-driven video surveillance; video stream analysis; Vision-Language Models; custom prompt-based detection; event detection; live surveillance; notification systems; role-based access control; YOLO; SAM 3; RTSP; RTMP..*

1. INTRODUCTION

The concept of video surveillance in today's world does not necessarily mean one video camera. A typical surveillance system comprises multiple cameras, live feeds, AI models to detect objects, and some notification mechanism to notify the user of an event. However, in reality, these elements exist in isolation, which makes it hard not just to receive the video but to connect all the following dots: processing the feed, detecting the events, managing the objects to detect, viewing control, and quick reaction.

It may also be possible for AI to integrate video processing and analysis into one single operation process. Vision Language Models (VLMs) analyze the video in accordance with a monitoring requirement expressed in written form, whereas other models such as YOLO and SAM 3 analyze objects, segment and track them. Therefore, an effective platform should allow for multiple sources of videos, have predefined and customizable detection requirements, and detections should become events.

Today, however, camera streaming, AI-based analysis, event logging, and alerting functions are commonly performed in separate systems. The camera streams using RTSP or RTMP protocol, whereas detection and alerting happens somewhere else. It becomes difficult to maintain situational awareness in a multi-camera scenario.

This problem is solved by VisionNext through one workflow that includes connecting video sources, processing live streams, analyzing them using AI models, recording detected events, and notifying users about them. It has the capability to process RTSP and RTMP streams as well as mobile video from OBS. It has use cases pre-built for fire detection, PPE detection, intrusion detection, persons detection, and safety monitoring, as well as providing the ability to create custom detection prompts. RBAC provides user role-based restrictions on accessing different resources.

The contribution of this paper is VisionNext architecture, which is a single platform for integration of multiple video sources, AI-assisted detection, prompt-based monitoring, events management, RBAC, and automated notifications that leverages FastAPI, MongoDB, Redis, and Docker technologie.

2. RELATED WORK

With respect to AI-powered video surveillance research, efforts have been increasingly shifted from camera-based surveillance systems to automatic understanding and action recognition systems based on video. While previous surveillance systems were reliant on pre-defined rules or model-based detections, current systems rely more heavily on the application of computer vision and AI in order to detect persons, objects, and events in videos..

Computer vision models are a key part of these kinds of systems. Object detection models like YOLO can detect objects and people in video frames and can provide their location and scores [1]. This makes YOLO suitable for applications like person detection, PPE detection, and general object detection. Likewise, the segmentation model provides more details regarding the occupied areas by these objects. SAM 3 takes this further and offers support for detection, segmentation, and tracking of objects in images and videos through text, visual, and exemplar prompts [2]. This makes these kinds of models suitable for applications that require more details on the objects in the scene.

Another notable advancement is the use of visual language models (VLMs) in the context of visual analysis. Visual language models have the ability to take image and textual inputs as well as provide textual responses related to visual inputs [3]. This means that a system can make visual analyses of images using natural language instructions instead of relying solely on pre-defined objects. A specific monitoring

task can thus be described using an instruction for a certain situation that must be detected by the system..

Video streaming is another vital component in an AI-driven surveillance system. RTSP is a protocol widely used to receive live feeds from network cameras [4], but RTMP may also be used for other live streaming video sources. The support for several streaming protocols ensures that the monitoring system is capable of connecting various types of cameras and video sources at the same time. It is becoming increasingly difficult to handle the continuous video streams along with the AI computation and event recording when many cameras are connected.

Currently available technologies typically work on separate stages of the monitoring workflow. One technology deals with video stream input, another one does the detection with its AI model, and the third one controls events or notifications. Thus, the biggest problem is not only detecting but also integrating all these technologies into one consistent workflow. This is what VisionNext aims at: it integrates multiple video inputs, AI-based detection, custom prompts-based analytics, event management, RBAC, and notifications into a single platform. As for the current state of the art, it is clear that computer vision models, prompt-based AI and video stream technologies are able to offer useful functionality for automated monitoring workflow.

3. PROBLEM DEFINITION AND OBJECTIVES

The issue that has been identified for this paper is the disjointed nature of video monitoring processes in different systems. During monitoring processes, there are various activities that occur independently. These include camera monitoring, video processing, AI-driven detection, event management, user management, alerts, and monitoring processes. Such disjointed processes make it hard to manage several cameras, link detection data with events, manage users, and act fast when significant events arise.

VisionNext aims at delivering one workflow, where all the monitoring tasks will be linked to each camera source, video feed, detection need, event, notifications, and the access level of the user. VisionNext is designed on five core objectives: (1) connecting and managing multiple video feeds; (2) offering AI-based detection with the help of VLMs and computer vision models; (3) enabling detection needs through pre-defined and prompt-based needs; (4) creating events and notifying the users via email and phone calls automatically; and (5) maintaining all events and monitoring information centrally for their management and review. Table 1 provides a mapping of all the objectives and their corresponding mechanisms along with the evidence to be delivered by them.

TABLE I. VisionNext objectives and corresponding system evidence

Objective	System mechanism	Expected evidence
Multiple video sources	RTSP, RTMP, and OBS streaming videos	Connection to cameras and the status of video stream
AI-based detection	VLM, YOLO, and SAM 3	Detection outcome and details
Flexible monitoring	Predefined use cases and prompt for custom ones	Configuration and analysis of detection
Automated response	Email and phone call notifications	Status of notification
Centralized monitoring	FastAPI, MongoDB, Redis, and dashboard	Event history and monitoring record

4. PROPOSED METHODOLOGY

4.1 Design Principles

The architecture of VisionNext has been defined based on splitting the monitoring task into several modules, with each module being responsible for certain functionality. FastAPI acts as the primary backend system which takes care of API management, users, camera parameters, detection requirements, and events. Processing of videos and running of AI models have been kept separate in order to avoid interference with the regular functioning of the application by the constant analysis of videos. MongoDB is used to store data related to users, cameras, detection requirements, and events, whereas Redis assists in processing, caching, and communication.

The system can operate with various video inputs. It can handle RTSP and RTMP video feeds, while mobile video input can also be used by connecting it to OBS via RTMP. After video is connected, the system will process it and send it for AI analysis. The use of VLMs for analyzing the video on a basis of specific use cases or prompts is possible. There are also options to use models like YOLO and SAM 3, in case the application requires object detection or segmentation.

The whole design has been based on the following monitoring process: the user adds the video feed, the system processes and analyses it, and whenever there is a particular condition set up, an event is triggered. The event carries all the necessary details that can be viewed and managed using the platform. According to the particular requirements set up, the system can even send email and call notifications about particular events. RBAC is applied in order to determine which cameras, events, and features each user can access.

4.2 System Architecture

The architecture of VisionNext system at high level is shown in Fig. 1. Interaction with the system occurs through the user interface, in which the users can manage the cameras, detection needs, and events. This request is then handled by the FastAPI backend and passed to the corresponding services

of the system. Inputs to the video processing part of the system include RTSP, RTMP, and mobile feeds using OBS.

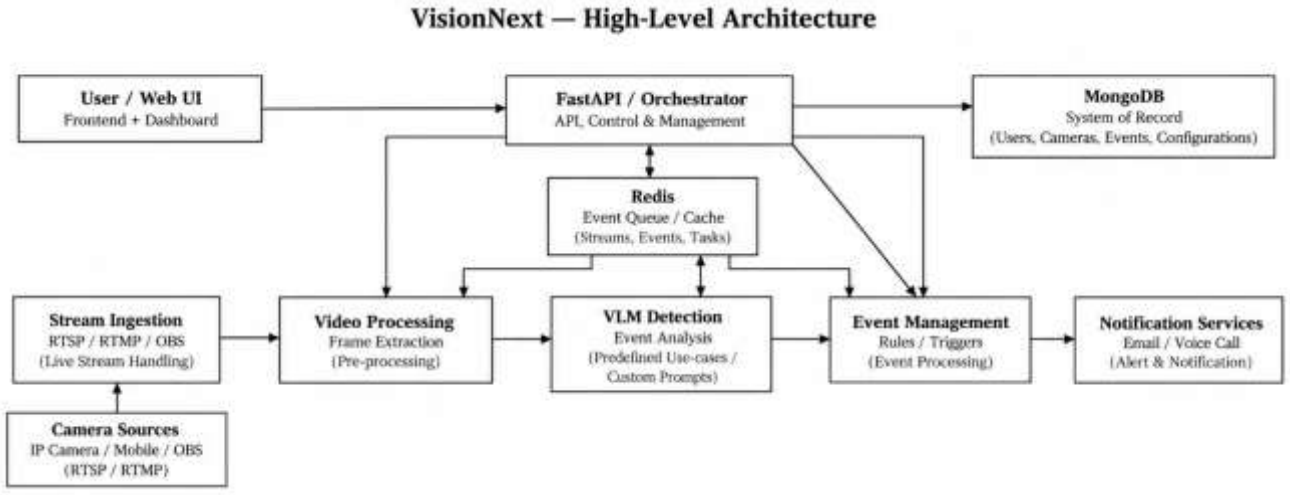


Fig. 1. High-level architecture of the proposed VisionNext platform.

4.3 End-to-End Workflow

The process starts when a user signs in into VisionNext and connects a camera or a video source to the application. The application supports video capture through RTSP, RTMP, and mobile streaming via OBS. RTSP offers a standardized protocol to control the delivery of live media including audio and video [4], and OBS Studio offers features for capturing the sources and streaming them to the configured streaming service or server [5]. After connecting the video stream, the backend processes the stream and sends the video to be analyzed by AI-based techniques. The detection layer leverages VLMs for the analysis of the video based on certain use cases or prompts [3], and YOLO and SAM 3 for the purposes of object detection and segmentation whenever needed [1], [2]. In case of occurrence of a certain condition, an event is created that contains information like camera source, event type, time stamp, and the captured frame.

Event details are stored in MongoDB which stores application data in the form of documents [6], while Redis allows for processing and caching across various components and offers data structures that can be used as queues and for event processing [7]. On the basis of the configurations made for notifications, the system will be able to send emails as well as make phone calls to alert the required users. The events detected and their statuses can be managed from the monitoring interface and RBAC will allow users to access only those cameras, events, and functionalities allocated to them. Docker can be used to containerize various components of the platform [8].

4.4 Event Normalization and Fault Handling

VisionNext employs the same event representation to ensure consistency of data obtained from different methods of detection. The representation may contain the following information: camera, detection method, detected object/condition, confidence/analysis result, timestamp, capture frame, status of the event, and other necessary information. The use of the same event representation enables events produced by different AI models or detection algorithms to be processed similarly in the backend and dashboard. At the same time, this approach does not require other elements of the system to rely on a specific format of the output produced by a particular AI model, making it easy to introduce any detection algorithms in the future, including VLM, YOLO, and SAM 3. Table 2 shows the role of each technology.

TABLE II. Technologies / models and their roles in VisionNext

Technology / Model	Main function within VisionNext	Generated / normalized information
VLM	Video analysis through prompts and custom detections	Detection results, prompts, responses to analyses, event details
YOLO	Real-time object detection and tracking	Object class, confidence, bounding box, detection results
SAM 3	Real-time object segmentation and tracking	Object regions, segmentation results, tracking details
RTSP	Live video streaming from cameras	Camera source, stream information, video frames
RTMP / OBS	Live video streaming from phones and other devices	Stream source, stream status, video frames

The video-processing and AI detection processes are independent of the primary API processing flow. The processing task gets the video stream, does its analysis, saves the detection/processing result, and transforms the resulting information into the format of a standard event in VisionNext. Then this information is saved and processing state changed so that the dashboard could demonstrate the process of analysis completion. If one of the detection techniques/AI

services is unavailable, the system registers a processing error in relation to this technique/service, while others remain operational. In this way, the problem with one AI element cannot be considered as a good detection result.

4.5 Event Generation and Notification Policy

VisionNext distinguishes between the process of detecting an event and the steps taken post-detection of an event. In the case of a detection of any event according to the set-up requirement of monitoring, an event is formed with data related to the camera source, event type, time, detection outcome, and a captured image. VisionNext supports various requirements for monitoring, which include fire detection, PPE detection, intrusion detection, person detection, and safety monitoring. Events are saved to enable their viewing and management by the user through the dashboard. When notification is enabled for an event, email and phone call notifications can be sent to the required users. Thus, it becomes easier for the user to be aware of any critical event without having to monitor each connected camera.

4.6 AI-Assisted Analysis

The AI element in VisionNext is primarily employed to comprehend the video content and recognize any conditions based on the surveillance requirements. The system may employ VLMs to perform analysis of the video content based on predefined use cases and prompts. For instance, the user may enter a prompt specifying the scenario that needs to be recognized, and the AI model will be able to analyze the video content based on such requirements. In case more advanced computer vision requirements arise, YOLO and SAM 3 may be applied for object detection and segmentation.

When the analysis by the AI system is done, the backend process will convert it to the common event format, as used by VisionNext. The event may now be stored and displayed at the dashboard level, as well as activate the notification process, if need be. In cases where the AI service is not available, or the results cannot be used, the error will be recorded instead of considering it as a detection process. In general, the AI layer operates with the video processing and the backend layers and not independently as the whole process of monitoring.

4.7 Processing Lifecycle and Event History

After completing the detection and configuration phases, the VisionNext processing modules execute the needed video processing and AI analysis services for the chosen camera streams. As illustrated in Fig. 2, there is a defined sequence of processing phases which include: STREAM_CONNECTED, PROCESSING, AI_ANALYSIS, EVENT_CHECK, EVENT_CREATED, NOTIFICATION and COMPLETION, FAILURE or STOPPING phases. Maintaining such state will help in tracking the progress of the monitoring process and the relationship between video processing, AI analysis, event creation and notification.

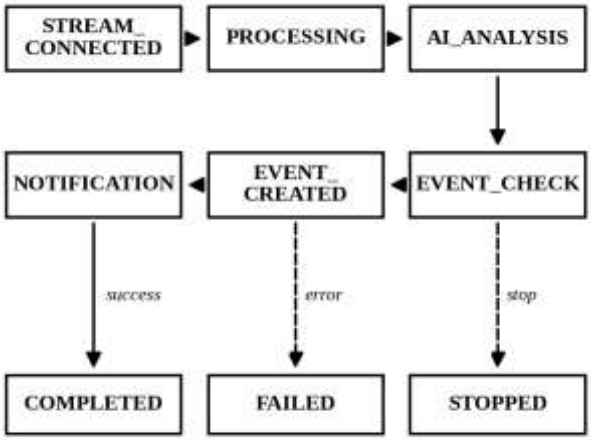


Fig. 2. Processing lifecycle of a monitoring task in VisionNext.

In addition, VisionNext has the capability to store event and monitoring history to give the user an overview of the events detected before and the corresponding camera information. Such information includes the source of the camera, the type of the event, the timestamp, the detection outcome, the frame taken, and whether there is any notification done regarding the event.

4.8 Data Model and Asynchronous Processing

VisionNext uses MongoDB as the database for storing information that is needed by VisionNext. The core information includes user details, role details, camera sources, detection setups, events, notifications, monitoring logs, and system setup. Information about the event can include camera source, type of detection, date and time, image captured, result of detection, and whether a notification was sent out. In addition, the use of documents makes it possible to store different types of detection results as well as custom prompt settings.

Redis is employed to facilitate asynchronous processing and communication among various components of the system. The processing flow may include video processing, AI analysis, event handling, and notification functions. Every function may store data regarding its ID, status, retry flag, timeout, and process logs. Processes involving video and AI processing will be executed independently of the regular API calls to ensure that the constant monitoring process does not affect the backend operations. In case of failure during the processing of a particular function, the data regarding the error can be saved for further investigations.

4.9 Authentication, Authorization and Audit

VisionNext offers authentication and role-based access control for the purpose of controlling access to the monitoring platform. Users will log directly into the platform, and access to cameras, detections, events, and other platform resources can be restricted based on roles assigned to users and corresponding permissions. Depending on system configuration, VisionNext may allow roles such as Super Admin, Admin, and other allowed user roles to be used within the platform. Security-related activities, such as camera management, changes to detection requirements, event

viewing, permission modification, system configuration changes, and other actions, can be logged as part of system logs. Such logs can contain the following data: time, user, action taken, object involved, outcome, and other metadata.

5. IMPLEMENTATION STATUS

VisionNext, which is currently being worked on, is partially deployed. Hence, this paper describes the architecture and workflow for VisionNext in general as the technical contribution rather than stating that all the features that have been planned in VisionNext are ready for production use. In the current implementation of VisionNext, we are mostly focusing on the components required for video surveillance, which include the FastAPI backend, MongoDB as the database, processing support through Redis, integration with video stream via RTSP and RTMP, artificial intelligence for analysis through VLMs, and computer vision through YOLO and SAM 3.

These aspects can be broken down into phases for VisionNext, which include backend/database initialization, video streaming integration, AI detection, event handling, notification handling, access control, and containerization in Docker. This approach is an effective roadmap to follow when implementing the system. It is, therefore, evident from the manuscript that there are three parts involved: the implemented parts, the system architecture, and the extensible parts.

6. EVALUATION FRAMEWORK

In the current state of partial implementation, the evaluation task is concentrated on establishing criteria for assessing the system rather than presenting any measurements of its qualities which could not be performed so far. There are several aspects which make the basis of evaluating VisionNext, namely video connection, AI accuracy, events handling and notifications delivery. Fig. 3 and Table 3 show the evaluation criteria and measurements methods for VisionNext.

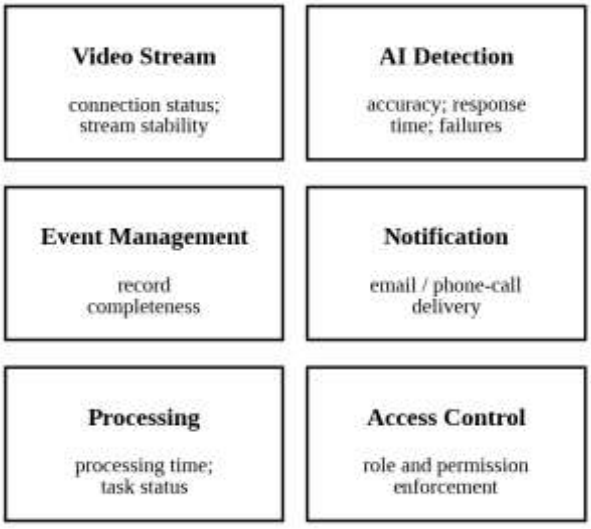


Fig. 3. Evaluation dimensions defined for the partially implemented VisionNext prototype.

TABLE III. Evaluation plan for the VisionNext prototype

Dimension	Metric / observation	Measurement method
Video Stream	Stream connection and availability	Connect controlled RTSP/RTMP streams and track the connection status and any stream interruptions
AI Detection	Detection accuracy and speed	Test predefined and custom detection cases and track detection results
Event Management	Event creation and completeness	Check if the created events include information about the camera, type of the event, time and frame
Notification	Notification delivery via email and calls	Evoking the controlled events and check if the configured notifications are delivered
Processing	Processing time and processing status	Repeat monitoring tasks and track processing time and processing status
Access Control	Permissions management	Test different user roles and permissions to control cameras and events

6.1 Experimental Design

The future experiment on VisionNext would require the use of controlled video samples which would have predetermined events like fire, PPE violations, intrusion, and person detections. This is because each video sample should have some predetermined expected event so that the results can be compared to the expected events. The controlled video samples will be processed repeatedly to see whether the process is consistent. The experiment will also involve comparing the expected events for different types of detections to see whether the expected events are similar to those produced by the VLM, YOLO, and SAM 3 and see whether the key information on the events is recorded. The controlled experiments would also involve event and notification tests to see whether the events are created, the frames captured, the processing times, and the email and phone calls notifications configured.

In terms of AI, the research should consider factors such as detection accuracy, processing time, processing failures, and the relevance of the detection process to the chosen monitoring requirement. The AI findings should not be considered as the absolute truth but should be compared to the expected events that are already known. Video processing testings should be conducted on a stream using either RTSP, RTMP, or OBS.

6.2 Threats to Validity

The first limitation is the implementation maturity as the VisionNext software is implemented only partly and thus there are no grounds for discussing its ability to work reliably in the

production environment. The second limitation is the detection coverage as no one detection technology can guarantee detection of all possible events. The third limitation is related to AI variability – the result of detection can depend on the choice of VLM or Computer Vision model, on the model version, prompt, video quality, and even on availability of services. The fourth limitation is related to external validity as results achieved in the controlled project environment can hardly be transferred to the large scale surveillance systems which use different cameras and other elements of the process. The fifth limitation is the use-case dependency.

7. DISCUSSION

The key feature of VisionNext is the unification of various video monitoring tasks within a single system. Instead of performing video processing, AI detection, event handling, and alerts separately, VisionNext combines all these elements within one process chain. Such an approach allows handling multiple camera feeds and using a consistent method of processing detection outcomes and creation of events. FastAPI can be also considered a convenient solution for the backend implementation of APIs and authentication [9].

Various techniques of AI also give flexibility to monitor the process. While VLMs can perform analysis of the video according to the set requirements and custom prompts [3], the YOLO and SAM 3 algorithms can help in detecting, segmenting and tracking specific objects [1], [2]. Thus, it gives the possibility to cover the needs of monitoring regardless of the detection method. The information is then structured in a single event format for storage purposes.

Separation of AI analysis and event management serves another benefit. The AI model performs analysis of the video feed and detecting a potential condition, while the backend deals with event creation, persistence, and notifications. Redis may help with real-time events processing and communication within the system components by means of Streams [7]. This will make the workflow more convenient and enable recording an event independently of a particular AI model output format.

The system design also facilitates the investigation and monitoring of identified incidents. Details like camera data, incident detection, time stamp, captured image, incident state, and notifications can be tied to each incident. As such, using the document data format of MongoDB, monitoring information as well as other results of detection can be kept in one document [6]. In cases where an incident or processing procedure fails, the system is able to save the details rather than just failing.

In terms of functionality, VisionNext integrates some of the features that are required in practice: multiple source videos support, AI-based detection, customizable prompt-based monitoring, events handling, RBAC, and notifications via emails and phone calls. But still, VisionNext needs to be treated as an AI-powered video monitoring tool rather than a comprehensive enterprise surveillance platform. Some more advanced features like distributed computing on a larger scale, richer analytics capabilities, and improved model management could be implemented in the future.

8. LIMITATIONS AND FUTURE WORK

The primary limitation in relation to VisionNext is the fact that the implementation is not yet fully done. Thus, the claims regarding detection accuracy, speed, efficiency of the notification system, and even performance of the AI cannot be quantitatively justified until all necessary controlled experiments are done. The second limitation is associated with the dependence of system performance on the video stream quality, which may come from RTSP, RTMP, or mobile streams. Another possible limitation may be related to the use of external AI models and services.

Future research would involve completion of the implementation that is discussed in Section 6. Possible enhancements could be implemented in terms of providing a larger number of camera streams, advanced event analytics, intelligent AI model management, logging and monitoring in a central place, as well as adding notifications. Other video source types and different detection use cases may be added according to project needs. Such enhancements will need to be tested in order to see their influence on the general performance of VisionNext.

9. CONCLUSION

VisionNext was introduced in this work, and it represents a system that integrates video streaming, detection based on artificial intelligence, event management, and notifications. The proposed platform is integrated with RTSP, RTMP, mobile, and VLM-based video streams and computer vision models such as YOLO and SAM 3.

Since VisionNext has not yet been completely realized, we have no test results at this point. Therefore, we do not make any statements about its detection precision and performance. Instead, we provide the system architecture, implementation guidelines and evaluation scheme to assess the effectiveness of the system on controlled video data sets. We consider it a solid foundation for an AI-based video surveillance system and further evaluations.

10. DECLARATIONS

Declaration of Conflict of Interest: Authors state that there are no financial or other personal interests that may have affected the results of the study described in this paper.

Funding: This research was not supported by any external funding.

Availability of Data and Source Code: The source code and project realization are not currently publicly accessible as of the preparation of the manuscript. The data and realization can become available in the future should an appropriate repository be created.

Ethics: This work does not involve human participants, personal medical data, animal experiments, or human-subject experimentation.

REFERENCES

[1] Ultralytics, "Object Detection with Ultralytics YOLO," Ultralytics Documentation. [Online]. Available: https://docs.ultralytics.com/tasks/detect/

[2] N. Carion, L. Gustafson, Y.-T. Hu, S. Debnath, R. Hu, D. Suris Coll-Vinent, et al., "SAM 3: Segment Anything with Concepts," Meta AI Research, Nov. 2025. [Online]. Available: https://ai.meta.com/research/publications/sam-3-segment-anything-with-concepts/

[3] Hugging Face, "Image-text-to-text: Vision Language Models," Transformers Documentation. [Online]. Available: https://huggingface.co/docs/transformers/main/tasks/image_text_to_text

[4] H. Schulzrinne, A. Rao, R. Lanphier, M. Westerlund, and H. Stiemerling, "Real-Time Streaming Protocol Version 2.0," RFC 7826, IETF, Dec. 2016. [Online]. Available: https://www.rfc-editor.org/rfc/rfc7826.html

[5] OBS Project, "OBS Studio Overview Guide," OBS Studio Documentation. [Online]. Available: https://obsproject.com/kb/obs-studio-overview

[6] MongoDB, Inc., "Documents," MongoDB Documentation. [Online]. Available: https://www.mongodb.com/docs/manual/core/document/

[7] Redis, "Redis Data Types," Redis Documentation. [Online]. Available: https://redis.io/docs/latest/develop/data-types/

[8] Docker, Inc., "What is Docker?," Docker Documentation. [Online]. Available: https://docs.docker.com/get-started/docker-overview/

[9] FastAPI, "Security – OAuth2 with Password and Bearer with JWT Tokens," FastAPI Documentation. [Online]. Available: https://fastapi.tiangolo.com/tutorial/security/oauth2-jwt/