# VigilantEye – Project Charter

# VigilantEye – Project Charter

Project Name: VigilantEye

Project Type: AI-Powered Multi-Agent Video Intelligence System

Technology Stack: Flask, Python 3.11, MySQL, Docker, Azure Container Apps

Deployment Status: Production-Ready (Azure Container Apps)

Project Purpose & Objective

VigilantEye is a production-ready AI-powered video intelligence platform that provides real-time video analysis, face recognition, anomaly detection, and intelligent alerting capabilities. The system processes CCTV feeds and video content to detect security threats, analyze behavior patterns, and generate actionable insights through a comprehensive web dashboard and Telegram bot integration.

Project Description

VigilantEye is a comprehensive AI-powered video intelligence system designed to transform traditional CCTV monitoring into an intelligent, automated security solution. The platform leverages advanced computer vision, machine learning, and real-time processing capabilities to analyze video feeds, detect anomalies, recognize faces, and generate actionable security alerts. Built on modern cloud-native architecture using Flask, Python, and Azure Container Apps, VigilantEye provides scalable, production-ready video intelligence capabilities for organizations requiring advanced security monitoring, threat detection, and behavioral analysis. The system integrates seamlessly with existing CCTV infrastructure while offering a user-friendly web dashboard and multi-channel notification system including Telegram bot integration for real-time alerting.

Problem or Opportunity Statement

Traditional CCTV monitoring systems face significant challenges including the need for constant human supervision, inability to process large volumes of video data in real-time, high false alarm rates, and lack of intelligent analysis capabilities. Security personnel are often overwhelmed by the volume of video feeds, leading to missed incidents and delayed response times. Additionally, manual video review for forensics is time-consuming and resource-intensive.

VigilantEye addresses these challenges by providing automated, AI-driven video analysis that can process multiple video streams simultaneously, detect anomalies and security threats in real-time, recognize individuals through face detection, and generate intelligent alerts. The system transforms passive video surveillance into an active, intelligent security solution that reduces human workload, improves response times, and enhances overall security effectiveness. This creates opportunities for organizations to improve security outcomes, reduce operational costs, and leverage video data for actionable insights and evidence collection.

Critical To Quality Metrics

Detection Accuracy Metrics:

- Face detection accuracy ≥ 85% with confidence threshold ≥ 0.35
- Object detection precision ≥ 90% for critical objects
- Anomaly detection recall ≥ 80% for security-relevant events
- False positive rate < 10% for all detection modules

Performance Metrics:

- Video frame processing latency < 2 seconds per frame
- API endpoint response time < 500ms (95th percentile)
- Alert delivery time < 5 seconds from detection to notification
- System uptime ≥ 99% availability
- Concurrent video stream processing: Support for multiple simultaneous streams

Quality Assurance Metrics:

- Test coverage ≥ 80% for all critical modules
- Zero critical bugs in production releases
- Database migration success rate: 100%
- Container deployment success rate ≥ 95%
- API endpoint availability ≥ 99.5%

User Experience Metrics:

- Dashboard page load time < 3 seconds
- Video upload success rate ≥ 95%
- User authentication success rate ≥ 99%
- Alert acknowledgment rate ≥ 80%

Security & Compliance Metrics:

- Data encryption at rest and in transit: 100%
- Authentication token validation: 100%
- SQL injection prevention: Zero vulnerabilities
- API rate limiting compliance: 100%

Assumptions

Technical Assumptions:

- Azure Container Apps platform will maintain availability and performance standards

- MySQL database will provide reliable connectivity and sufficient storage capacity
- Telegram Bot API will remain accessible and functional for notification delivery
- OpenCV and face_recognition libraries will continue to be maintained and compatible
- Python 3.11 runtime environment will be supported throughout the project lifecycle
- containerization will provide consistent deployment across environments
- Sufficient network bandwidth will be available for video upload and processing

Operational Assumptions:

- Users have access to modern web browsers supporting HTML5 and JavaScript
- CCTV cameras or video sources can provide video feeds in supported formats
- Users have Telegram accounts for receiving alerts (where Telegram integration is used)
- Database backups and disaster recovery procedures are in place
- Monitoring and logging infrastructure is available for production operations
- Team members have necessary technical skills or can acquire them during development

Business Assumptions:

- Stakeholders will provide timely feedback during development and testing phases
- Project scope will remain relatively stable during implementation
- Required computing resources and cloud infrastructure will be available when needed
- Security and privacy requirements will be clearly defined and remain consistent
- Integration with existing systems will be feasible within project constraints

Constraints/Dependencies

Technical Constraints:

- Azure Container Apps platform limitations (scaling, resource allocation, regional availability)
- MySQL database connection requirements and connection pool limitations
- Telegram Bot API rate limits (30 messages per second per bot)
- Docker image size constraints affecting deployment speed
- Video file size limitations (20GB maximum per upload)
- Memory and CPU constraints for real-time video processing
- Network bandwidth limitations for video streaming and uploads

Resource Constraints:

- Limited budget requiring use of open-source tools and free-tier cloud services where possible
- Fixed development timeline with defined milestones and deadlines
- Team size and skill set limitations
- Computing resource availability for AI/ML model training and inference
- Storage capacity for video archives and processed data

Dependencies:

- External Dependencies:
- Azure Cloud Platform availability and service reliability
- MySQL database service availability and connectivity
- Telegram Bot API availability and functionality
- OpenCV, face_recognition, and other Python library updates and compatibility
- Docker Hub or Azure Container Registry for image storage
- Internet connectivity for cloud-based deployments

Internal Dependencies:

- Database migrations must complete successfully before application startup
- Frontend dashboard depends on backend API availability
- Alert system depends on detection module outputs
- Video processing depends on sufficient storage and compute resources
- Authentication system must be operational before user access
- Telegram integration requires valid bot token and webhook configuration

Regulatory & Compliance Constraints:

- GDPR/HIPAA compliance requirements for data handling and storage
- Data retention policies and deletion requirements
- Privacy regulations regarding face recognition and biometric data
- Security standards for handling sensitive video content

In Scope

Core Functionality:

- Real-time video processing and analysis from CCTV feeds and uploaded videos
- Face detection, recognition, and demographics analysis using FaceAi module
- Object detection and classification using computer vision models
- Anomaly detection for unusual activities and security threats
- Behavior analysis and pattern recognition
- Crowd density analysis and monitoring
- Video management including upload, storage, segmentation, and clip generation
- Frame extraction and batch processing capabilities

User Management & Access Control:

- Multi-user authentication system (JWT for API, sessions for web)
- Role-based access control (Admin, Manager, Member, Viewer roles)
- Project-based organization with member management
- User profiles and site-based access control
- Secure password hashing and token management

Alert & Notification System:

- Telegram bot integration for real-time alerts
- Multi-channel notification support with configurable escalation
- Message acknowledgment and tracking
- Auto-escalation and closure mechanisms
- Webhook support for external integrations

Dashboard & User Interface:

- Web-based dashboard for monitoring and management
- FaceAi dashboard for face detection results
- CCTV Intelligence dashboard for video analysis
- AI Analytics dashboard for detection results
- Responsive design for desktop and mobile browsers

Infrastructure & Deployment:

- Azure Container Apps deployment with auto-scaling
- Docker containerization for consistent deployments
- MySQL database integration with SQLAlchemy ORM
- Database migrations using Alembic
- Health monitoring and status endpoints
- Production-ready configuration and security hardening

Testing & Quality Assurance:

- Comprehensive test suite covering authentication, user management, projects, videos, recordings, notifications, health checks, validation, integration, and edge cases
- Automated testing framework using pytest
- Test data management and cleanup procedures

Benefits

Operational Benefits:

- Reduced Manual Monitoring: Automated video analysis eliminates the need for constant human supervision, reducing operational costs and human error
- Faster Incident Response: Real-time detection and alerting enable immediate response to security threats, reducing response times from hours to seconds
- Improved Security Effectiveness: AI-powered detection identifies threats that might be missed by human operators, improving overall security posture
- Scalable Processing: Cloud-native architecture allows processing of multiple video streams simultaneously without proportional cost increases
- 24/7 Monitoring: Automated system provides continuous monitoring without fatigue or breaks

Business Benefits:

- Cost Reduction: Reduced need for security personnel and manual video review lowers operational expenses
- Enhanced Evidence Collection: Automated video analysis and incident reporting provide comprehensive forensic evidence
- Improved Decision Making: Data-driven insights from behavior analysis and anomaly detection support better security decisions
- Competitive Advantage: Advanced AI capabilities differentiate organizations in security-sensitive industries
- Risk Mitigation: Proactive threat detection reduces security incidents and associated costs

Technical Benefits:

- Modern Architecture: Cloud-native design ensures scalability, reliability, and maintainability
- Integration Ready: RESTful APIs and webhook support enable integration with existing security systems
- Flexible Deployment: Docker containerization allows deployment across different cloud providers or on-premises
- Comprehensive Testing: Robust test suite ensures code quality and reduces production issues
- Documentation: Well-documented codebase and APIs facilitate maintenance and future enhancements

User Benefits:

- Intuitive Interface: User-friendly web dashboard makes complex video intelligence accessible to non-technical users
- Real-time Alerts: Instant notifications via Telegram ensure critical incidents are never missed
- Comprehensive Analytics: Detailed reports and analytics provide actionable insights
- Multi-project Support: Project-based organization allows management of multiple sites or clients
- Role-based Access: Granular permissions ensure appropriate access levels for different user roles

Core Features & Capabilities

AI & Analytics Modules:

- Face Detection & Recognition – FaceAi module with demographics analysis, ambiguity detection, and face encoding
- Object Detection – Real-time object identification and tracking
- Anomaly Detection – Unusual activity detection with confidence scoring
- Behavior Analysis – Pattern recognition and behavioral analytics
- Crowd Density Analysis – Population density monitoring and alerts

Video Management System:

- Video upload, storage, and metadata management
- Frame extraction and batch processing
- Video segmentation and clip generation
- Recording sessions with live streaming support
- Multi-format video support with comprehensive metadata

Alert & Notification System:

- Telegram Bot Integration – Real-time alerts with acknowledgment buttons
- Multi-channel notification support (configurable escalation channels)
- Message processing with auto-escalation and closure
- Webhook support for external integrations

User & Project Management:

- JWT-based API authentication + session-based web authentication
- Role-based access control (Admin, Manager, Member, Viewer)
- Multi-project support with project members and permissions
- User profiles and site-based access management

Infrastructure & Deployment:

- Azure Container Apps – Auto-scaling (1-10 replicas), HTTPS, health checks
- Docker Containerization – Production-ready Dockerfile with Gunicorn
- MySQL Database – Production database with SQLAlchemy ORM
- Database Migrations – Alembic-based schema versioning
- CI/CD Ready – GitHub Actions compatible

Technical Architecture

Backend Stack:

- Framework: Flask 2.3.3 with Blueprint architecture
- Database: MySQL 8.0 with SQLAlchemy ORM
- Authentication: Flask-JWT-Extended, Flask sessions
- Task Scheduling: APScheduler for background jobs
- API: RESTful APIs (v1 and v2 endpoints)

AI/ML Libraries:

- OpenCV 4.8.1.78 (Computer Vision)
- face-recognition 1.3.0 (Face Detection)
- scikit-learn 1.3.2 (Machine Learning)
- NumPy, Matplotlib (Data Processing)

Frontend:

- HTML5 templates with responsive design
- JavaScript/jQuery for dynamic interactions
- CSS styling with custom themes
- Dashboard interfaces for FaceAi, CCTV, and AI Analytics

DevOps & Infrastructure:

- Container Platform: Azure Container Apps
- Container Registry: Azure Container Registry (ACR)
- Orchestration: Docker Compose for local development
- Web Server: Gunicorn (4 workers, 2 threads per worker)
- Health Monitoring: Built-in health check endpoints

Database Schema Overview

Core Models:

- Users, Projects, ProjectMembers
- Videos, Recordings, Segments, Clips, Frames
- Devices (CCTV cameras, sensors)
- Analytics and ViewEvents

AI Models:

- FaceDetections, FaceEncodings, DemographicsAnalysis, AmbiguityAnalysis
- ObjectDetections, AnomalyDetections, BehaviorAnalysis, CrowdDensityAnalysis
- SmartAlerts

Integration Models:

- OutboundMessages (Telegram notifications)
- FaceIdentityAgent (watchlist management)

Key Endpoints & APIs

Authentication:

⇒ \`/api/auth/login\`, \`/api/auth/register\`, \`/api/auth/logout\`
⇒ \`/login\`, \`/signup\` (web routes)

Video Management:

⇒ \`/api/v2/videos/*\` – Upload, list, get, delete videos
⇒ \`/api/v2/recordings/*\` – Recording session management
⇒ \`/api/v2/projects/*\` – Project CRUD operations

AI Analytics:

⇒ \`/api/faceai/*\` – Face detection and analysis
⇒ \`/api/cctv/*\` – CCTV intelligence endpoints
⇒ \`/api/ai/*\` – Object detection, anomaly detection, behavior analysis

Telegram Integration:

⇒ \`/api/telegram/ingest\` – Message ingestion
⇒ \`/webhook/telegram/<secret>\` – Webhook receiver

Health & Monitoring:

⇒ \`/api/v1/health\` – System health check
⇒ \`/health\` – Application status

Project Status & Milestones

Completed:

* Core Flask application architecture
* Database models and migrations
* Authentication system (JWT + sessions)
* Video management system
* FaceAi detection module
* AI Analytics modules
* Telegram bot integration
* Azure deployment configuration
* Docker containerization
* Comprehensive test suite