

# IntelliGaze: A Wearable AI Camera System

## Real-time Environmental Description via AI and TTS

Touhidul Alam Seyam (230240003)  
Eftakar Uddin (230240004)  
Tasmim Akther Mim (230240025)  
Shafiul Azam Mahin (230240022)  
Muntasir Rahman (230240002)

Microprocessor Lab  
Future Professor Radiathun Tasnia,  
Junior Lecturer,  
BGC Trust University Bangladesh

May 7, 2025

# Outline

- 1 Introduction
- 2 System Workflow
- 3 System Architecture
- 4 Core Logic and Flows
- 5 Technology and Code
- 6 Security Aspects
- 7 Team Contributions
- 8 Future Benefit
- 9 Conclusion

# What is IntelliGaze?

**Core Idea:** An innovative project integrating an ESP32-CAM on goggles to capture and document moments using AI.

- Provides real-time auditory descriptions of the surroundings.
- Allows users to stream live video (implicitly).
- Generates scene descriptions via AI (Google Gemini).
- Aims for hands-free operation via a mobile app.



## Applications:

- Assistive technology for the visually impaired.
- Contextual awareness tool.
- Personal memory archival (with descriptions).

# Challenges Addressed

- Challenges faced by visually impaired individuals in navigating and understanding their environment.
- Need for real-time, hands-free contextual information.
- Desire for an accessible way to document and recall visual experiences.
- Existing assistive technologies may lack real-time AI-powered descriptive capabilities or seamless integration.

# How IntelliGaze Helps

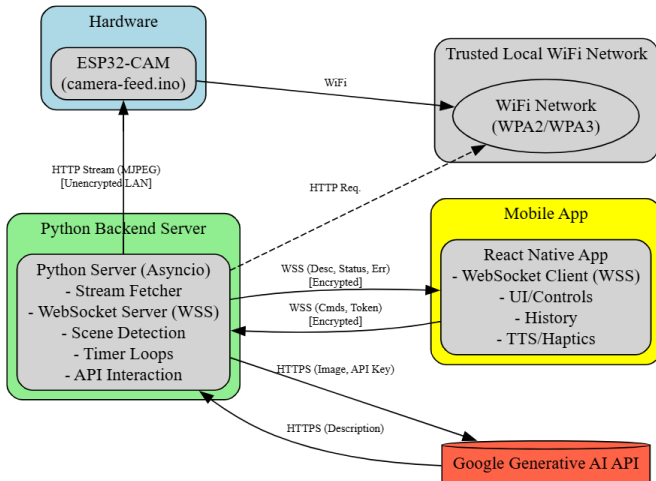
- Provides immediate auditory feedback about the surroundings through AI analysis and TTS.
- Offers a wearable, hands-free experience via goggles and a mobile app.
- Leverages the power of Google Gemini for intelligent scene description.
- Creates a potential tool for enhanced independence and richer environmental interaction for the visually impaired.
- Serves as a novel personal memory archival system with generated context.

# Why IntelliGaze is Needed

- Addresses the limitations of existing assistive tools by providing real-time, AI-driven environmental descriptions.
- Leverages accessible hardware (ESP32-CAM) and powerful AI (Google Gemini) for a cost-effective yet capable solution.
- Offers a hands-free, wearable approach for seamless integration into daily life.
- Provides significant value for visually impaired users (enhanced navigation, safety, context) and broader applications (memory assistance, context awareness).

- **ESP32-CAM:** Streams video continuously over local WiFi.
- **Python Backend Server:**
  - Fetches stream (with reconnection).
  - Runs WebSocket Server (WSS) for secure mobile app communication.
  - Manages client state (active/inactive, description interval).
  - **Automatic Mode:** Periodically checks for scene changes. If changed, sends frame to API, broadcasts description to active clients.
  - Handles commands from app ('start', 'stop', 'set\_interval', 'describe\_now').
- **React Native Mobile App:**
  - Connects securely to Python Server (WSS, with reconnection).
  - UI: Start/Stop, Interval config, Describe Now, History, Status indicators.
  - Sends commands to server.
  - Receives descriptions/errors.
  - Provides TTS output and Haptic feedback.

# Overall Component Interaction



Enhanced VisionVault System Architecture

Figure: High-level view of components and communication protocols.



# Modular Design

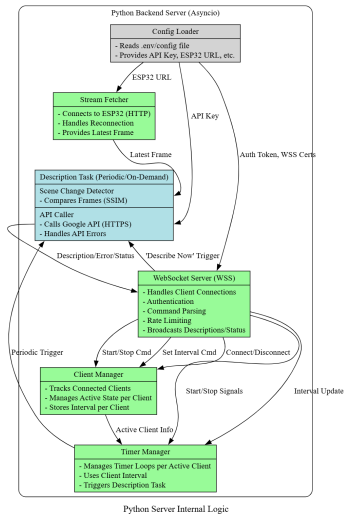


Figure: Key internal modules and data flow within the Python server.

# User Interaction States

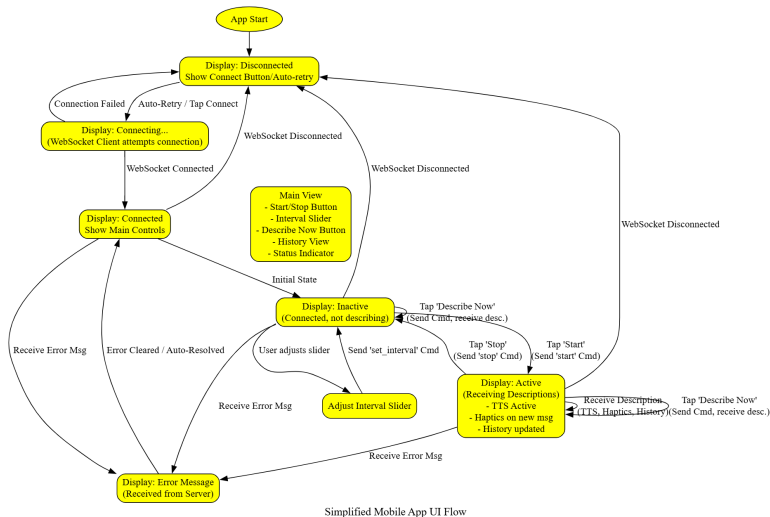


Figure: Simplified state transitions within the React Native application.

# Optimizing API Calls

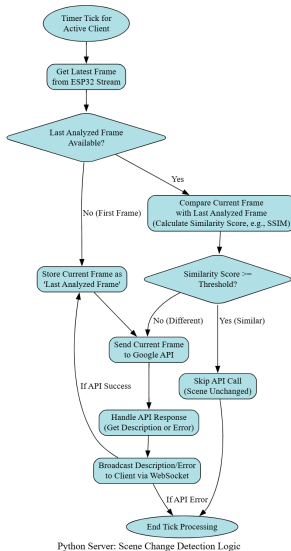
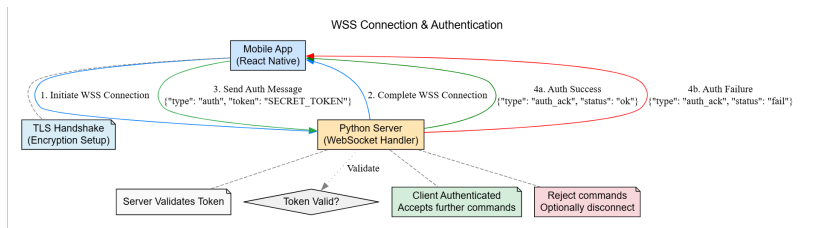


Figure: Flowchart for deciding whether to call the AI API based on frame

# Connecting Securely



**Figure:** Sequence for establishing a secure WebSocket connection with token authentication.

# Key Technologies Used

- **Hardware:** ESP32-CAM WiFi Module
- **Video Streaming:** MJPEG over HTTP
- **Backend:** Python 3 with 'asyncio', 'websockets', 'aiohttp'/'requests', 'opencv-python', 'scikit-image' (for SSIM), 'google-generativeai', 'python-dotenv'.
- **Frontend:** Expo React Native (TypeScript)
- **Communication:** Secure WebSockets (WSS)
- **AI:** Google Generative AI (Gemini API)
- **Output:** Text-to-Speech (TTS) Library (Expo TTS)
- **Feedback:** Haptics API (Expo Haptics)

# Risks and Mitigations (Trusted LAN Focus)

- **Network:** Assumes trusted WiFi (WPA2/WPA3). Public networks are risky.
- **ESP32 Stream (HTTP):** Unencrypted. Rely on WiFi security. HTTPS on ESP32 is challenging.
- **WebSocket (WSS):** *Mitigation:* Use Secure WebSockets ('wss://') with TLS certificates to encrypt app-server communication.
- **Authentication:** *Mitigation:* Implement token-based authentication for WebSocket clients to prevent unauthorized access/commands.
- **API Key:** *Mitigation:* Store securely (e.g., '.env' file, environment variables). Secure the host machine.
- **DoS:** *Mitigation:* Implement rate limiting on WebSocket commands/connections on the server.
- **Input Validation:** *Mitigation:* Sanitize all commands/data received from the mobile app on the server.
- **Resource Management:** Ensure server cleans up resources on client disconnect.

# Individual Contributions

- **Touhidul Alam Seyam:** Will lead project design and integration. Will develop the core Python backend server including asynchronous stream handling, WebSocket communication (WSS), scene change detection, and Google Gemini API integration. Will implement core React Native application logic and features. Will design system architecture and workflow. Will create diagrams and presentation structure.
- **Eftakar Uddin:** Will configure and flash the ESP32-CAM firmware. Will conduct initial video stream testing and debugging. Will assist with Python server frame fetching setup.
- **Tasmim Akther Mim:** Will design and implement the user interface (UI) and user experience (UX) for the Expo React Native mobile application. Will integrate Text-to-Speech (TTS) and Haptics feedback components.
- **Shafiul Azam Mahin:** Will perform integration testing between the Python server, mobile app, and ESP32-CAM stream. Will research WebSocket libraries and will contribute to debugging network communication issues.
- **Muntasir Rahman:** Will contribute to project documentation. Will research Google Generative AI API capabilities and usage patterns relevant to the project. Will assist with overall system testing.

# Potential Impact and Next Steps

- **Enhanced Accessibility:** Potential for significant improvement in independence and quality of life for visually impaired individuals.
- **Broader Applications:** Can be adapted for augmented reality, professional training simulations, or automated documentation.
- **Technological Advancement:** Platform for integrating more sophisticated AI models (object recognition, activity detection), sensor fusion (adding audio, GPS), or cloud integration.
- **Personalization:** Future versions could learn user preferences or adapt descriptions based on context or location.



# IntelliGaze System Overview

- A wearable system leveraging ESP32-CAM for continuous video input.
- Python backend processes the video, interacts with AI, and manages secure mobile app communication (WSS).
- Optimizes AI calls using scene change detection.
- React Native app provides user control (Start/Stop, Interval, On-Demand), displays history, and delivers auditory (TTS) / tactile (Haptic) feedback.
- Incorporates security measures for operation on trusted networks.
- Provides a robust framework for real-time, AI-driven environmental description.

**This design outlines the system; next steps involve implementation.**