# **GuideSense: Installation Manual**

## **System Overview**

This system integrates real-time object detection with speech feedback using a three-tier architecture:

* **Frontend**: React-based web interface displaying video streams
* **Backend**: Node.js server handling text-to-speech conversion
* **FastAPI Server**: Python-based server for video processing and YOLO object detection

## **Prerequisites**

### **Hardware Requirements**

* CPU: 4+ cores recommended
* RAM: 8GB minimum, 16GB recommended
* GPU: NVIDIA GPU with CUDA support (optional but recommended)
* Storage: At least 10GB free space
* Internet connection (required for initial model download)

### **Software Requirements**

* Ubuntu 20.04 LTS or higher (recommended)
* Windows 10/11 or macOS also supported
* Python 3.8 or higher
* Node.js 14.x or higher
* npm 6.x or higher
* Git

## **Installation Guide**

### **1. Clone Repository**

git clone https://github.com/htmw/2025S-Power-Anger.git

cd 2025S-Power-Anger

### **2. Python Environment Setup**

#### **Create and activate a Conda environment (recommended)**

#### [Download Anaconda Distribution | Anaconda](https://www.anaconda.com/download)

conda create -n yolo-detection python=3.9

conda activate yolo-detection

#### **Install Python dependencies**

cd fastapi\_server

pip install -r requirements.txt

If the requirements.txt file is not available, create one with the following content:

fastapi>=0.103.1

uvicorn>=0.23.2

aiortc>=1.10.0

opencv-python>=4.8.0

numpy>=1.24.0

ultralytics>=8.0.0

av>=10.0.0

websockets>=11.0.3

python-multipart>=0.0.6

aiohttp>=3.8.4

pillow>=9.5.0

### **3. System Dependencies**

#### **Ubuntu/Debian**

sudo apt update

sudo apt install -y ffmpeg libavdevice-dev libavfilter-dev libavformat-dev libavcodec-dev libswresample-dev libswscale-dev libavutil-dev

# For audio playback

sudo apt install -y alsa-utils

#### **macOS (using Homebrew)**

brew install ffmpeg

#### **Windows**

Download and install FFmpeg from the official website: <https://ffmpeg.org/download.html>

Once unzipped, add the directory of the unzipped file to your system environment PATH variables.

### **4. Node.js Backend Setup**

cd backend

npm install

Install any missing packages, such as ‘play-sound’, with npm install play-sound

Create a .env file in the backend directory with the following variables:

GOOGLE\_SPEECH\_CRED=Your credentials

### **5. Google Cloud Setup for Text-to-Speech/Maps API**

1. Create a Google Cloud account if you don't have one (cloud.google.com)
2. Create a new project
3. Enable the Cloud Text-to-Speech API
4. Create a service account https://console.cloud.google.com/iam-admin/serviceaccounts/ (https://cloud.google.com/iam/docs/service-accounts-create) and download the JSON key file
5. Also enable Maps JavaScript API, Geocoding API, Directions API, Places API from API library for frontend. Maps Platform API?

### **6. Frontend Setup**

cd frontend

npm install

Create a .env file in the frontend directory with the following variables:

REACT\_APP\_CLERK\_PUBLISHABLE\_KEY=Your key

REACT\_APP\_GOOGLE\_MAPS\_API\_KEY=Your key

This can be found in the json key file retrieved from google cloud.

**7. Running the System**

Open three terminal windows and run each component:

#### **Terminal 1: FastAPI Server**

cd fastapi\_server

conda activate yolo-detection

uvicorn main:app --host 0.0.0.0 --port 8000 --reload

#### **Terminal 2: Node.js Backend**

cd backend

npm start

#### **Terminal 3: Frontend**

cd frontend

npm start

## **First Run Considerations**

1. On first run, the system will download the YOLO11 model, which may take some time depending on your internet connection
2. Initial connections may be slow while the model loads into memory
3. Subsequent connections should be faster

## **Troubleshooting**

### **No video stream appears**

* Check browser console for WebRTC errors
* Ensure your camera is not being used by another application
* Verify that all three services are running

### **No audio output from detections**

* Check that your Google Cloud credentials are correctly set up
* Verify audio playback is working on your system
* Check backend console for text-to-speech errors

### **Slow or laggy video processing**

* Consider reducing the processing frequency in the FastAPI server
* Use a system with high GPU support for better performance
* Adjust the YOLO model size (e.g., from yolo11n to yolo11s)

## **Advanced Configuration**

### **Changing the YOLO Model**

To use a different YOLO model, modify the model initialization in fastapi\_server/main.py:

# Use the smallest model for better performance

model = YOLO("yolo11n.pt") # Change to yolo11s.pt, yolo11m.pt, etc.

### **Adjusting Detection Frequency**

To reduce system load, adjust the frame skipping parameter in VideoTransformTrack class:

self.process\_every\_n\_frames = 5 # Process every 5th frame instead of every 3rd

### **Customizing Speech Generation**

Modify the speech generation parameters in speechService.js to change voice, language, or other text-to-speech settings.

## **Deployment Considerations**

For production deployment, consider:

1. Setting up a reverse proxy (Nginx/Apache) for the frontend and backend
2. Using process managers (PM2/systemd) for the Node.js backend
3. Using Gunicorn with Uvicorn workers for the FastAPI server
4. Implementing proper authentication and security measures
5. Setting up HTTPS for secure WebRTC connections

## **License Information**

Please ensure you comply with the licenses for all components:

* Ultralytics YOLO: AGPL-3.0 license (or Enterprise license for commercial use)
* Google Cloud Text-to-Speech: Subject to Google Cloud Platform Terms of Service