# ESP32-CAM Image Acquisition and Object Detection System

## INTRODUCTION

This project integrates an **ESP32-CAM** microcontroller with a **Python-based pipeline** to capture and process images for real-time object detection. The ESP32 acts as a **web server**, delivering live camera frames, while Python scripts handle camera configuration, automated image collection, and object detection using a **YOLOv11** model.

The primary objective is to **generate image datasets** and **detect specific objects** (e.g., insects, tools, or other custom classes) using **minimal hardware** and **open-source software**.

## SYSTEM SETUP

Hardware

- ESP32-CAM with OV2640 sensor.

- Connected with USB and Wi-Fi

Software Stack

- ESP32 firmware: WebServer Camera sketch (Arduino)

- Python Libraries:requests, serial, re, datetime, cv2, ultralytics

- Object detection model: YOLOv11, custom-trained (best.pt)

## Image Acquisition Pipeline

Upon startup, the ESP32 connects to a Wi-Fi network and launches a web server. A Python script communicates with the ESP32 via HTTP requests to:

- Configure the camera through /control

- Capture images through /capture

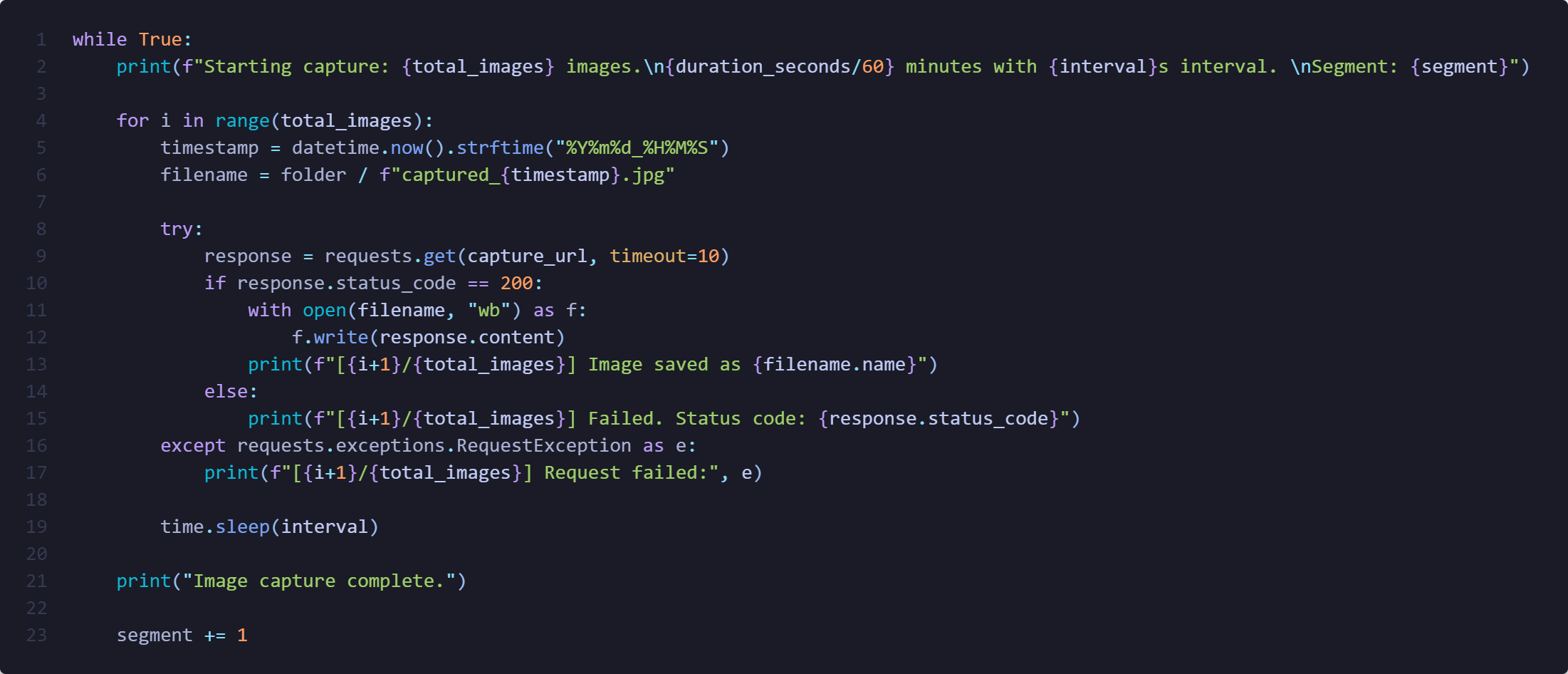
**Camera Settings:**

The ESP32 camera was configured for image quality and YOLOv11 compatibility (640x640) using the following settings:



|  |  |  |
| --- | --- | --- |
| **Setting** | **Value** | **Description** |
| framesize | 6 | VGA(640x480) |
| quality | 10 | Lower value = higher JPEG quality |
| brightness | 1 | Better visibility |
| contrast | 2 | Emphasize edges |
| saturation | 1 | Slightly richer colors |
| awb,aec | 1 | Automatic white and exposure balance |
| gainceiling | 2 | Avoid overexposure |
| dcw | 1 | Downsize |

For demonstration, the system was configured to capture 150 images over 2.5 minutes, at 1-second intervals.



## Object Detection Pipeline

**Model:**

A custom YOLOv11 model trained on relevant object classes is used via the ultralytics package.

**For each captured image:**

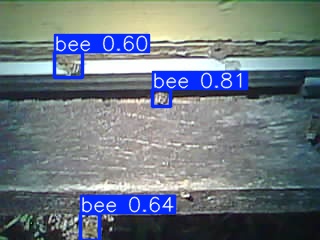
- YOLOv11 performs inference

- The resulting image is annotated with bounding boxes and class labels

- Annotated images are saved to the results/ folder



Example results:



## Summary

This system demonstrates how to:

- Build a lightweight Real-time camera system pipeline

- Automate data collection with the ESP32-CAM

- Apply YOLOv11 models for object detection

- Process and store annotated results for further analysis

The approach is scalable for applications such as wildlife monitoring, agriculture, home automation, and research data collection.