**Motion Detection Automatic Switch using Python and IoT**

**Introduction:**

In this project, we aim to create a Motion Detection Automatic Switch system that leverages computer vision with Python and integrates with the Internet of Things (IoT) using Thing Speak. The system detects the presence of a person using a webcam and YOLOv3 (You Only Look Once) deep learning model. The status is then communicated to an IoT platform (Thing Speak) via an ESP8266-based IoT device. This allows for real-time monitoring and control of an external device, in this case, an LED.

**Abstract:**

The Motion Detection Automatic Switch combines computer vision techniques with IoT to provide an intelligent system for detecting human presence and controlling external devices. The project involves the use of YOLOv3, a popular object detection algorithm, for real-time person detection. The detected status is sent to ThingSpeak, an IoT platform, through an ESP8266-based device. The platform stores and allows retrieval of the detection status and related data. The IoT device controls an LED, serving as a representation of an external device. The system provides insights into power consumption and person availability time.

**Project Overview:**

The project consists of two main components: the Python-based computer vision module and the IoT module using an ESP8266-based device.

**Computer Vision Module (Python):**

* Utilizes OpenCV and YOLOv3 for real-time person detection using a webcam.
* Calculates power consumption based on the time a person is detected.
* Communicates with ThingSpeak API to update the detection status and power consumption.

**IoT Module (Arduino, ESP8266):**

* Connects to Wi-Fi using specified credentials.
* Reads data from ThingSpeak, including person detection status and power consumption.
* Controls an LED based on the person detection status. Collects and sends data to ThingSpeak for further analysis.

**Algorithm:**

**Person Detection Algorithm using YOLOv3:**

* Capture frames from the webcam. Preprocess frames and forward pass through the YOLOv3 network.
* Identify persons in the frame with a confidence threshold.
* Measure the time a person is detected.

**IoT Control Algorithm using ESP8266:**

* Connect to Wi-Fi using provided credentials. Read person detection status and power consumption from ThingSpeak.
* Control an LED based on the detection status.
* Transmit relevant data to ThingSpeak for monitoring.

**Circuit Connection and Diagram:**

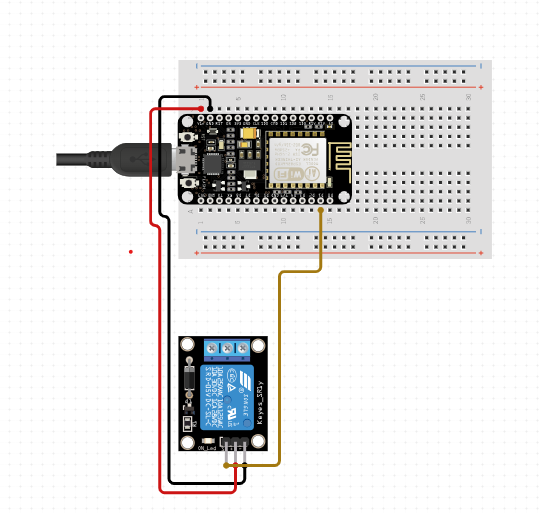
**Components:**

* Webcam
* ESP8266-based IoT device
* LED
* Power source

**Connections:**

* Connect the webcam to the computer.
* Connect the LED to the specified pin on the ESP8266.
* Connect the ESP8266 to the Wi-Fi network

**Circuit Diagram:**



**Coding:**

**Python Code (Transmitter):**

import cv2

import numpy as np

import requests

from datetime import datetime

# ThingSpeak API endpoint URL

api\_url = 'https://api.thingspeak.com/update'

api\_key = 'PV6CRIWEKLYVEBGL' # Replace with your Write API Key

# Load YOLO model and its configuration for YOLOv3

net = cv2.dnn.readNet('yolov3.weights', 'yolov3.cfg')

# Load COCO dataset names

with open('coco.names', 'r') as f:

classes = f.read().strip().split('\n')

# Initialize the webcam with the correct camera index

cap = cv2.VideoCapture(1) # Change the index to 0 if this doesn't work

person\_detected = False

start\_time = None

while True:

ret, frame = cap.read()

# Perform blob from the frame and forward pass through YOLO network

blob = cv2.dnn.blobFromImage(frame, 0.00392, (320, 320), (0, 0, 0), True, crop=False)

net.setInput(blob)

outs = net.forward(net.getUnconnectedOutLayersNames())

class\_ids = []

confidences = []

boxes = []

Width = frame.shape[1]

Height = frame.shape[0]

for out in outs:

for detection in out:

scores = detection[5:]

class\_id = int(np.argmax(scores))

confidence = scores[class\_id]

if confidence > 0.5 and class\_id == 0: # Class 0 is for 'person'

center\_x, center\_y, w, h = (detection[0:4] \* np.array([Width, Height, Width, Height])).astype(int)

x = center\_x - w // 2

y = center\_y - h // 2

class\_ids.append(class\_id)

confidences.append(float(confidence))

boxes.append([x, y, w, h])

indexes = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.4)

for i in range(len(boxes)):

if i in indexes:

person\_detected = True

start\_time = datetime.now()

break

if person\_detected:

elapsed\_time = (datetime.now() - start\_time).total\_seconds()

power\_consumption = elapsed\_time / 3600 # in watt-hours

print("Person detected")

print(f"Power Consumption: {power\_consumption:.2f} Wh")

payload = {'api\_key': api\_key, 'field1': 1, 'field2': power\_consumption}

else:

print("No person detected")

payload = {'api\_key': api\_key, 'field1': 0, 'field2': 0}

try:

# Make a POST request to the ThingSpeak API

response = requests.post(api\_url, params=payload)

# Check if the request was successful (status code 200)

if response.status\_code == 200:

print("ThingSpeak Update Successful")

else:

# Print an error message if the request was not successful

print(f"Error: {response.status\_code} - {response.text}")

except Exception as e:

# Handle any exceptions that may occur during the request

print("Error:", str(e))

# Break the loop if 'q' is pressed

if cv2.waitKey(1) == ord('q'):

break

# Release the capture object

cap.release()

cv2.destroyAllWindows()

**Arduino Code (Receiver):**

#include <ESP8266WiFi.h>

#include <ThingSpeak.h>

const char \*ssid = "Vimal";  // Add closing double-quote

const char \*password = "vimal2005";  // Add closing double-quote

const char \*api\_key = "AYCBNL23D9IMCFQJ";  // Replace with your Read API Key

const unsigned long channelID = 2421538;  // Replace with your ThingSpeak Channel ID

const int ledPin = D2;  // Define the pin connected to the LED

WiFiClient client;  // Create a WiFiClient object

bool personDetected = false;

unsigned long startTime;

void setup() {

  Serial.begin(115200);

  pinMode(ledPin, OUTPUT);

  // Connect to Wi-Fi

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL\_CONNECTED) {

    delay(1000);

    Serial.println("Connecting to WiFi...");

  }

  Serial.println("Connected to WiFi");

  // Initialize ThingSpeak

  ThingSpeak.begin(client);

}

void loop() {

  // Read ThingSpeak data

  int personDetectionStatus = ThingSpeak.readLongField(channelID, 1, api\_key);

  float powerConsumption = ThingSpeak.readFloatField(channelID, 2, api\_key);

  // Add similar lines to read data for Power Saving (Field 3) and Person Availability Time (Field 4)

  // Control the LED based on the person detection status

  if (personDetectionStatus == 1) {

    digitalWrite(ledPin, HIGH);  // Turn on the LED

    Serial.println("LED turned ON");

    personDetected = true;

    startTime = millis();  // Start measuring time

  } else {

    digitalWrite(ledPin, LOW);  // Turn off the LED

    Serial.println("LED turned OFF");

    if (personDetected) {

      unsigned long elapsedTime = millis() - startTime;

      // Perform actions with powerConsumption, Power Saving, and Person Availability Time

      // ...

      Serial.print("Power Consumption: ");

      Serial.print(powerConsumption);

      Serial.println(" Wh");

      Serial.print("Time Person Detected: ");

      Serial.print(elapsedTime);

      Serial.println(" ms");

      Serial.println("Data Received from ThingSpeak");

      personDetected = false;

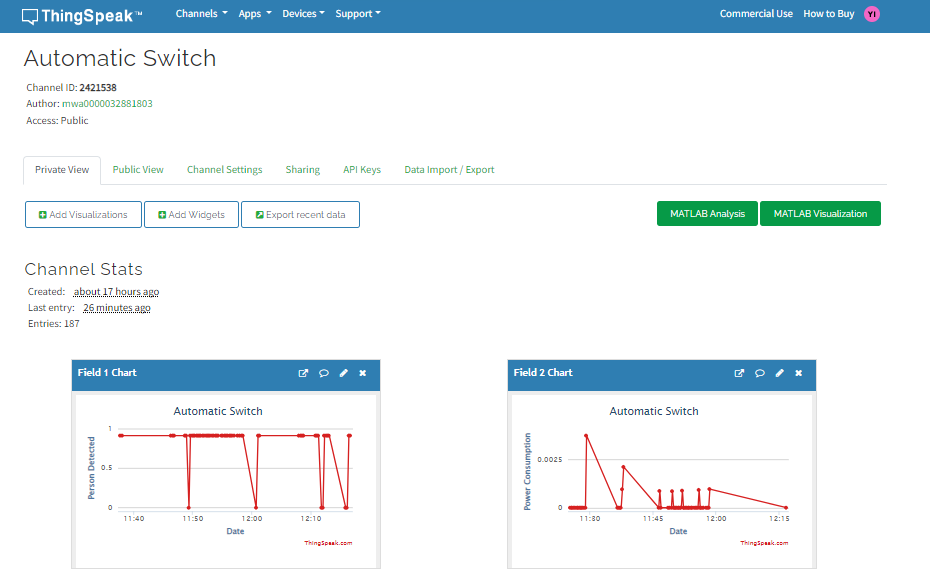
    }

  }

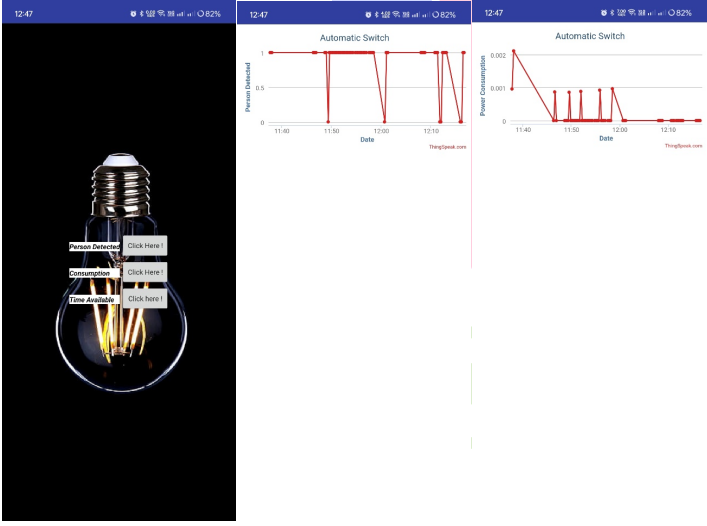
  delay(5000);  // Adjust the delay as needed

}

**ThingSpeak Database:**

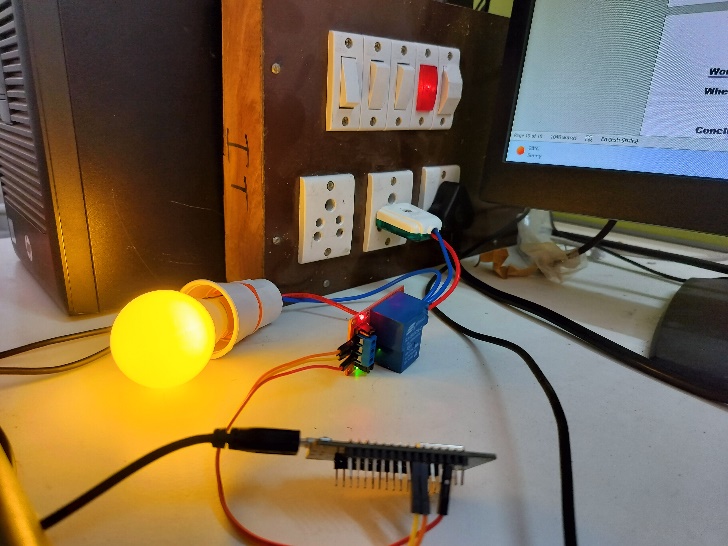


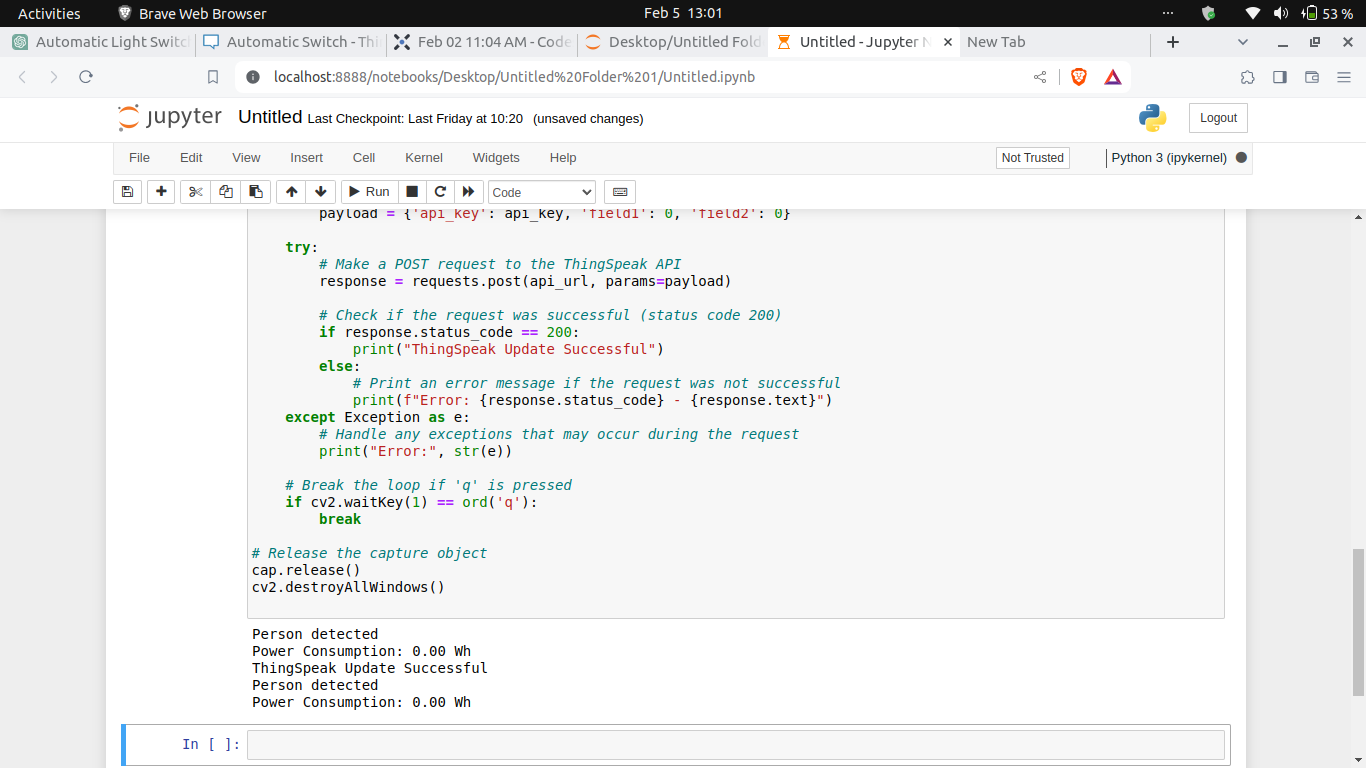
**App Connectivity :**



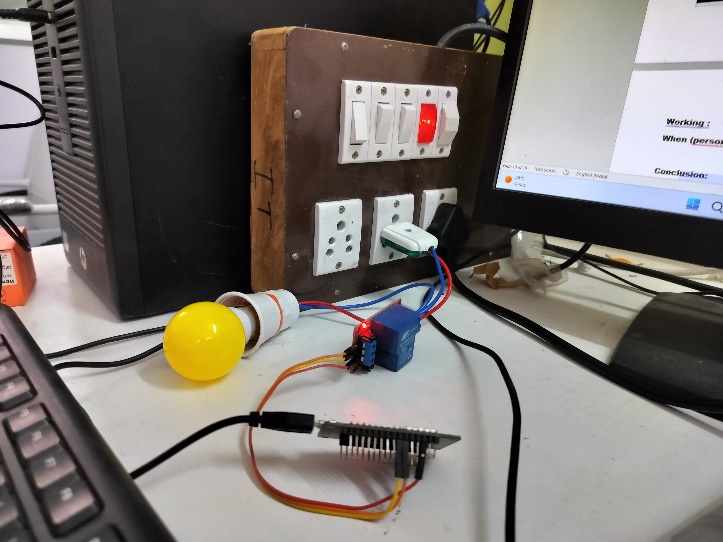
**Working :**

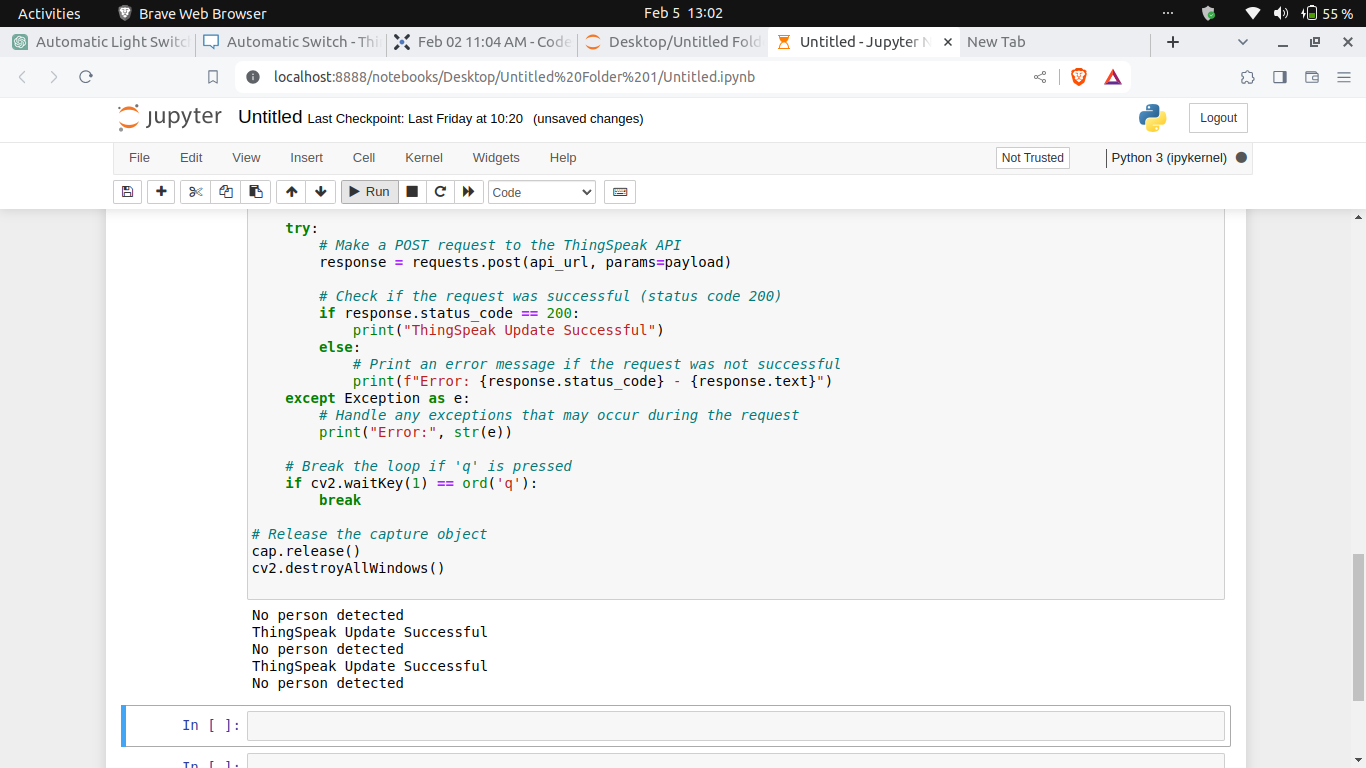
When (personDetectionStatus == 1):





When (personDetectionStatus == 0):





**Conclusion:**

The Motion Detection Automatic Switch project presents an innovative integration of computer vision and IoT technologies. This system provides an effective solution for automated environments, demonstrating the potential for applications in smart homes and energy-efficient solutions. The combination of real-time person detection and IoT communication enables a responsive and intelligent control system for external devices.