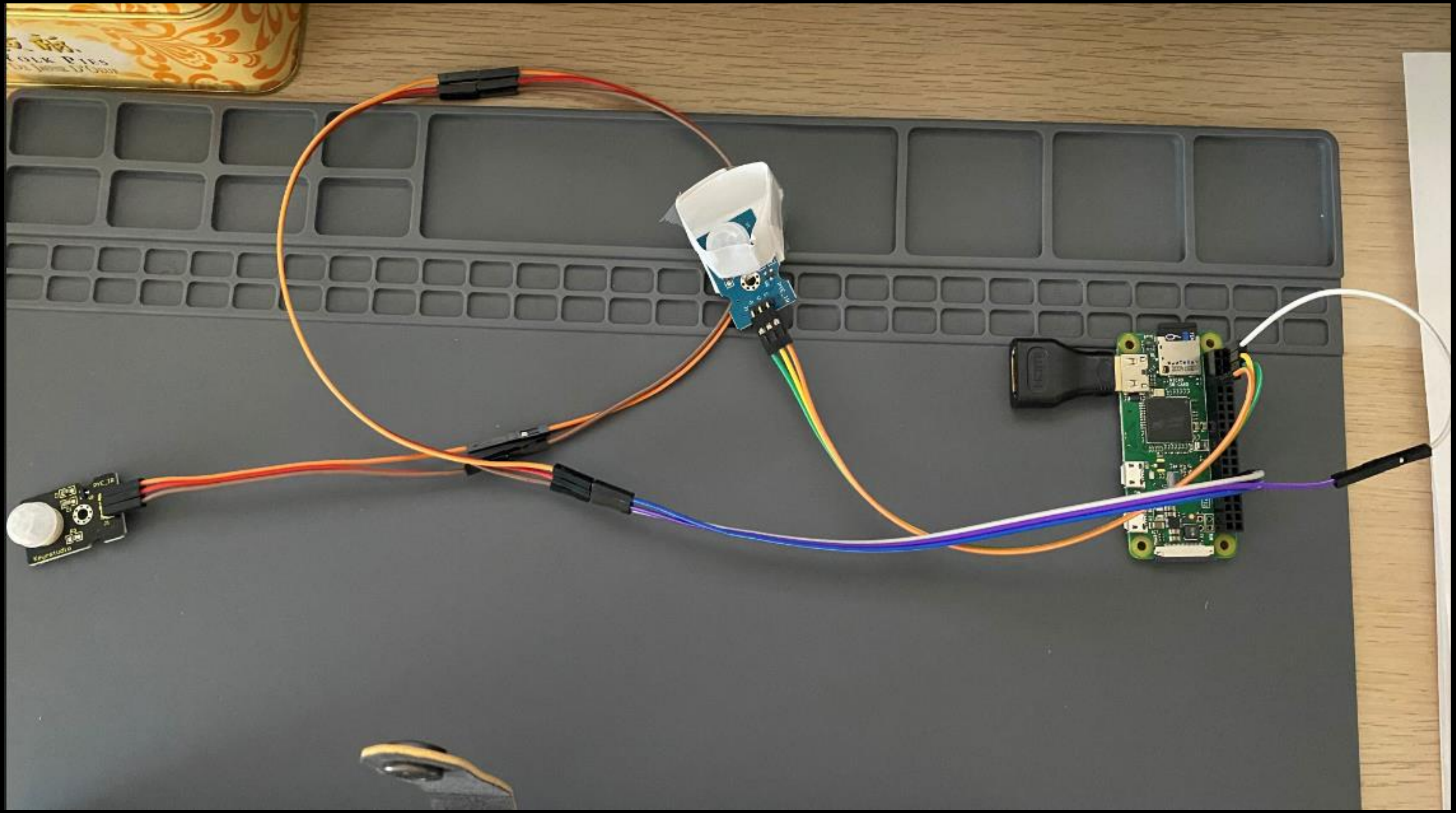
IOT 360 Project Update

Projected Deliverable Goal

- One man project
- It doesn’t have a physical cat gate
- A raspberry Pi project to prevent cat enter into areas where he is not allowed
- Cat will be detected when he approaches to the restricted area
- The motion sensor will send signal to the system
- Then the system will trigger play a music through the speak to scare away the cat

Current Setup

- 1 Rpi Zero
- Soldered female header
- Jump wires
- Two PIR sensors
  - Sensor A use pin 12 and sensor B use pin4
  - Sensor A will be placed about 1/2’ above the floor
  - Sensor B will be placed about 4‘ above the floor

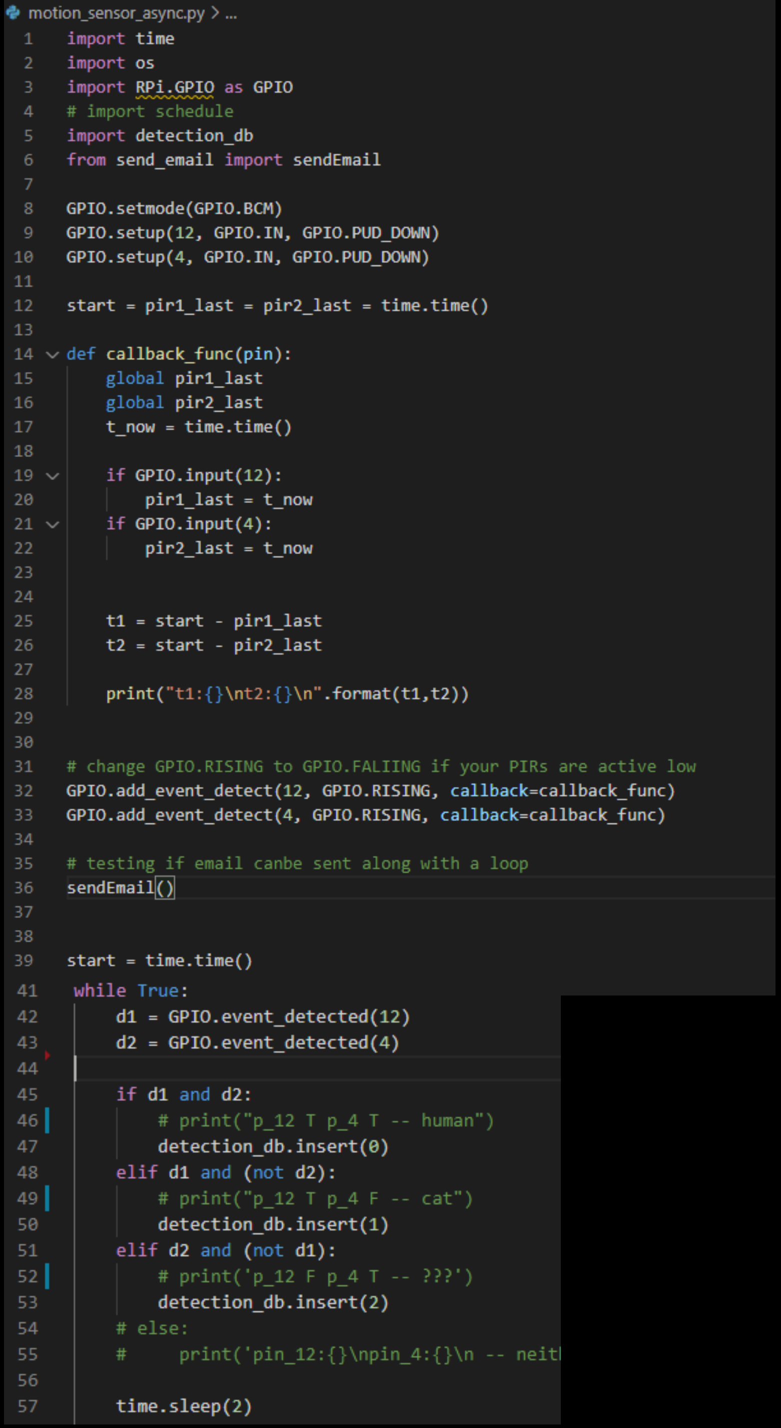


Changes & Challenges

- Use 2 PIR sensors instead of 1
  - Advantage: 1 PIR cannot differentiate cat from human passing through
  - Cat can only trigger the sensor placed near ground
  - Human will trigger both sensors
- Challenge:
  - 2 PIR sensor do not necessarily being triggered simultaneously.
- Proposed Solution:
  - Async functions / Callback functions
- Milestones:
  - Tested GPIO.add_event_detect, timeouts, etc..
  - Python wait/async functons
  - Continue testing

PIR Motion Detection

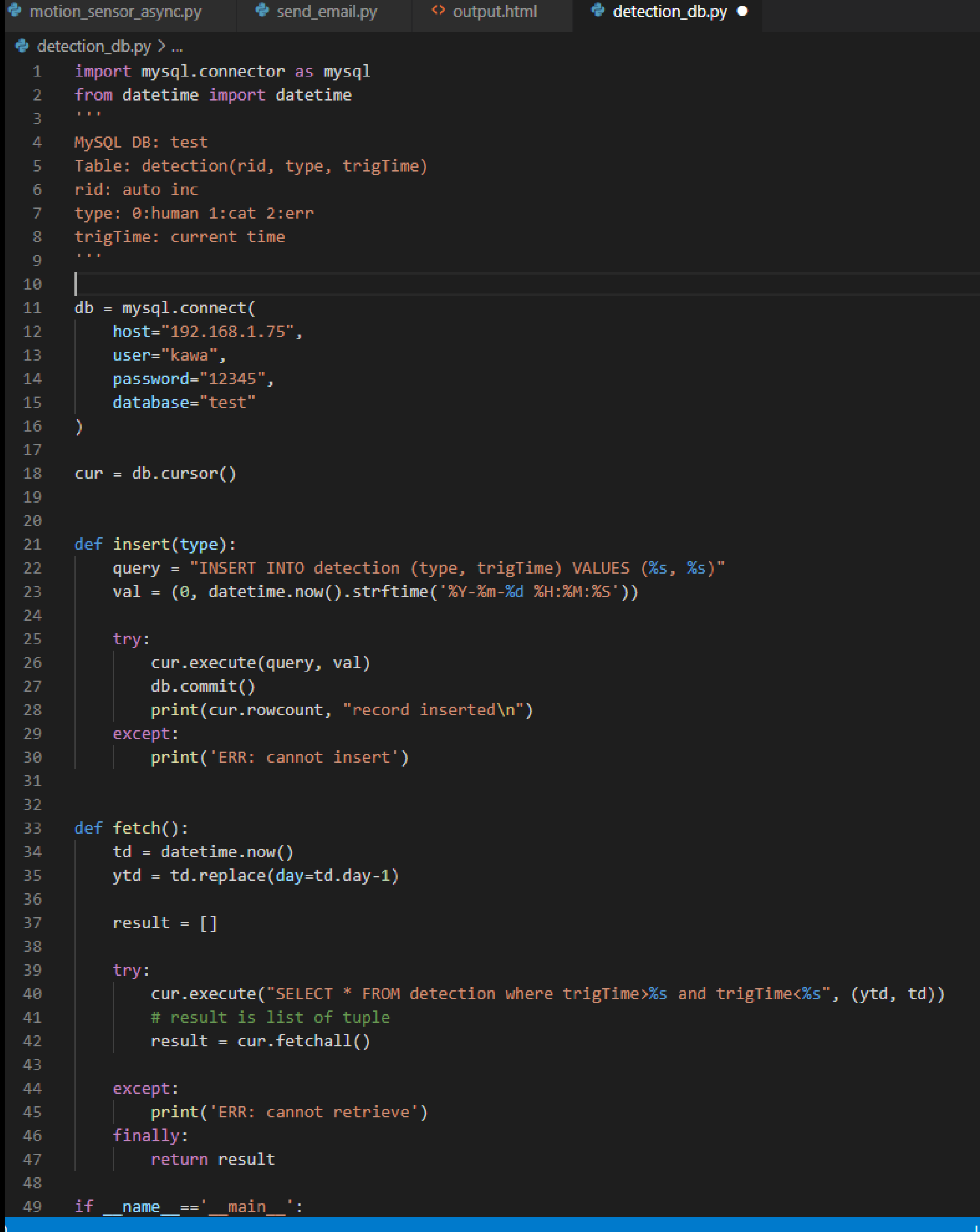
- Two PIR sensors
- Sensor A use pin 12 and sensor B use pin4
- Sensor A will be placed about 1/2’ above the floor
- Sensor B will be placed about 4 feet above the floor



```python
import time
import os
import RPi.GPIO as GPIO
# import schedule
import detection_db
from send_email import sendEmail

GPIO.setmode(GPIO.BCM)
GPIO.setup(12, GPIO.IN, GPIO.PUD_DOWN)
GPIO.setup(4, GPIO.IN, GPIO.PUD_DOWN)

start = pir1_last = pir2_last = time.time()

def callback_func(pin):
    global pir1_last
    global pir2_last
    t_now = time.time()

    if GPIO.input(12):
        pir1_last = t_now
    if GPIO.input(4):
        pir2_last = t_now


    t1 = start - pir1_last
    t2 = start - pir2_last

    print("t1:{}\nt2:{}\n".format(t1,t2))


# change GPIO.RISING to GPIO.FALIING if your PIRs are active low
GPIO.add_event_detect(12, GPIO.RISING, callback=callback_func)
GPIO.add_event_detect(4, GPIO.RISING, callback=callback_func)

# testing if email canbe sent along with a loop
sendEmail()


start = time.time()
while True:
    d1 = GPIO.event_detected(12)
    d2 = GPIO.event_detected(4)

    if d1 and d2:
        # print("p_12 T p_4 T -- human")
        detection_db.insert(0)
    elif d1 and (not d2):
        # print("p_12 T p_4 F -- cat")
        detection_db.insert(1)
    elif d2 and (not d1):
        # print('p_12 F p_4 T -- ???')
        detection_db.insert(2)
    # else:
    #     print('pin_12:{}\npin_4:{}\n -- neit

    time.sleep(2)
```

Connect to Database

- Setup MySQL database on local NAS
- DB's remote access only available within the local network
- Python script remotely connect to the database through mysql connector
- Functions to insert and fetch records

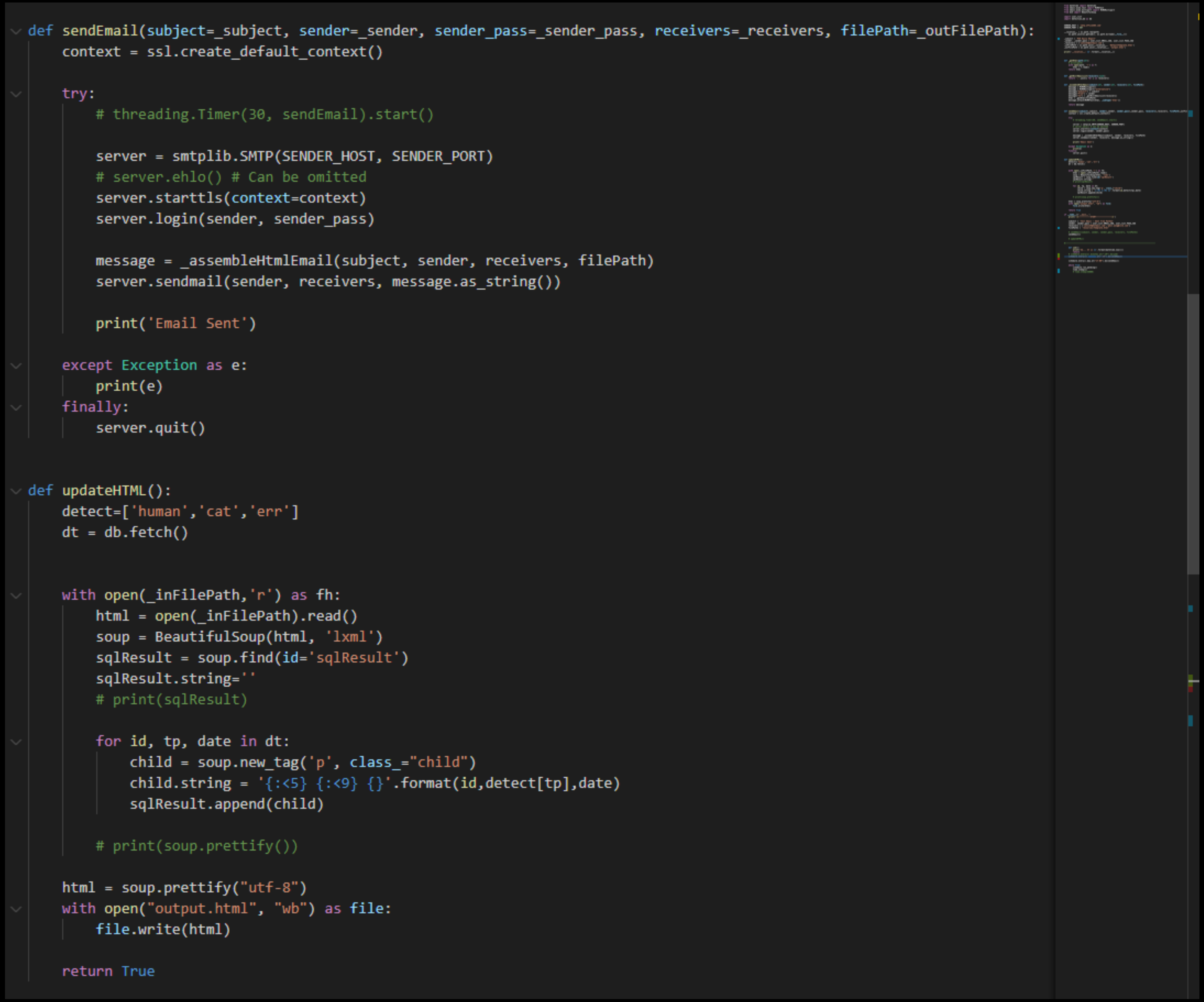

motion_sensor_async.py | send_email.py | output.html | detection_db.py

detection_db.py > ...

```python
import mysql.connector as mysql
from datetime import datetime
'''
MySQL DB: test
Table: detection(rid, type, trigTime)
rid: auto inc
type: 0:human 1:cat 2:err
trigTime: current time
'''

db = mysql.connect(
    host="192.168.1.75",
    user="kawa",
    password="12345",
    database="test"
)

cur = db.cursor()


def insert(type):
    query = "INSERT INTO detection (type, trigTime) VALUES (%s, %s)"
    val = (0, datetime.now().strftime('%Y-%m-%d %H:%M:%S'))

    try:
        cur.execute(query, val)
        db.commit()
        print(cur.rowcount, "record inserted\n")
    except:
        print('ERR: cannot insert')


def fetch():
    td = datetime.now()
    ytd = td.replace(day=td.day-1)

    result = []

    try:
        cur.execute("SELECT * FROM detection where trigTime>%s and trigTime<%s", (ytd, td))
        # result is list of tuple
        result = cur.fetchall()

    except:
        print('ERR: cannot retrieve')
    finally:
        return result

if __name__=='__main__':
```

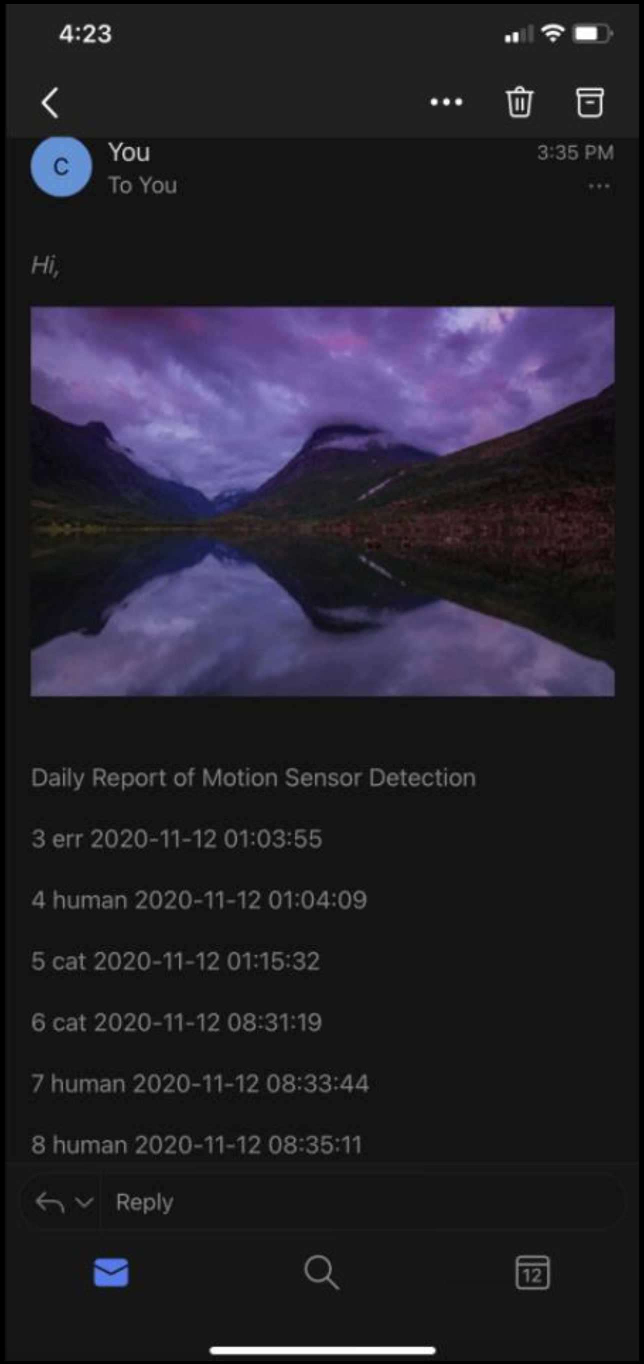
Email Daily Report

- A scheduled task to send daily email contains the past 24hrs passing through records
- Fetch records from database
- Insert the records into an HTML file
- Email the HTML file



```python
def sendEmail(subject=_subject, sender=_sender, sender_pass=_sender_pass, receivers=_receivers, filePath=_outFilePath):
    context = ssl.create_default_context()

    try:
        # threading.Timer(30, sendEmail).start()

        server = smtplib.SMTP(SENDER_HOST, SENDER_PORT)
        # server.ehlo() # Can be omitted
        server.starttls(context=context)
        server.login(sender, sender_pass)

        message = _assembleHtmlEmail(subject, sender, receivers, filePath)
        server.sendmail(sender, receivers, message.as_string())

        print('Email Sent')

    except Exception as e:
        print(e)
    finally:
        server.quit()


def updateHTML():
    detect=['human','cat','err']
    dt = db.fetch()


    with open(_inFilePath,'r') as fh:
        html = open(_inFilePath).read()
        soup = BeautifulSoup(html, 'lxml')
        sqlResult = soup.find(id='sqlResult')
        sqlResult.string=''
        # print(sqlResult)

        for id, tp, date in dt:
            child = soup.new_tag('p', class_="child")
            child.string = '{:<5} {:<9} {}'.format(id,detect[tp],date)
            sqlResult.append(child)

        # print(soup.prettify())

    html = soup.prettify("utf-8")
    with open("output.html", "wb") as file:
        file.write(html)

    return True
```

A Sample Email



Hi,

Daily Report of Motion Sensor Detection

```
3 err 2020-11-12 01:03:55

4 human 2020-11-12 01:04:09

5 cat 2020-11-12 01:15:32

6 cat 2020-11-12 08:31:19

7 human 2020-11-12 08:33:44

8 human 2020-11-12 08:35:11
```

What’s Next

- Start writing project report
- Run python script on each boot of the Rpi

- The Problem
- Proposed Solution(s)
- Platform of choice and equipment for creating the solution
- Milestones
- Timeline