Assignment 2: Individual Assignment (Practical)

**Link to Video:**

**Summary:**

For my assignment 1 I have written about smart home systems, therefore for this assignment 2 I had the idea of creating a system that could be used in smart home systems. The chosen system for this assignment is a Gas and Fire detection and alert system. Using the MQ-2 sensor to detect gasses near its atmosphere and the IR Flame sensor which is used to sense whether there is flame nearby. Both the MQ-2 and IR Flame sensor are very cheap and available everywhere. Furthermore, this system can be broadened and is able to have more complex features in terms of detection and alerting. Both the MQ-2 sensor and the IR Flame sensor are very flexible when it comes to its usage as it can communicate to the Arduino using both analog and digital signals. I would say the consistency and accuracy of the sensors way exceeds their price as it is sensitive to gas or flame near it and being able to adjust their settings for further optimization. Both sensors will be used to create the smart home system I envisioned.

The 2 sensors are connected to the Arduino board, both will be used to read and print data. If there is gas or flame detected, it will print out “Gas Detected”, “Gas Value: ” and “Flame Detected” where the data would then be sent to the Raspberrypi Desktop while also activating actuators such as the Red LED Light and the Buzzer. If there is gas detected, the Raspbian Desktop will read the data from the serial port and using the python script it would then take the gas values and upload them to the database and launch the alert php files to send warning emails to the emails registered. The gas values would also be displayed on a web UI hosted on Apache Local Host, where users can filter out times to check time specific gas leaks.

**Conceptual Design:**

A diagram of a computer network

Description automatically generated

Activity Diagram:

A diagram of a flowchart

Description automatically generated

A diagram of a software program

Description automatically generated

The Arduino board detects gas or flame from the MQ-2 or IR Flame Sensor and will trigger the red LED light to blink and the buzzer to beep. The system then sends the detection information and the gas values to the Raspberry Pi via Serial Communication. The python script on the Raspbian Desktop then inserts the received gas value data into the MySQL database of the specific time and value the gas is detected and runs the alert.php file on the webserver. The php file on the webserver would then send warning emails to all emails registered in the database. To display the data to a localhost webpage, PHP is also used to query the data and is put in a table on the website using CSS. The gas value table would be updated as long as the website is refreshed, and the webserver is on.

The MQ-2 Sensor will be using the analog output so that it can detect the gas values. This is so that it can set certain thresholds for the sensor to detect gas while being able to upload the gas level data onto the database. The IR flame sensor would be using the digital output as it is only needed to detect whether there is the presence of fire nearby, and no values are needed for it. On the gas value website, the gas value data is extracted from the MySQL database and inserted into a table in the HTML website. Users can filter out specific dates or time to check on the gas value.

**Implementation:**

***Sensors:***

* MQ-2 (Analog): To detect gas in the atmosphere above certain gas level threshold and upload the gas value data to the database.
* IR Flame Sensor (Digital): To detect whether there is there is the presence of flame nearby.

***Actuators:***

* 2 LEDs (Green and Red): Will light up green constantly when there are no issues but will blink red when gas or flame is detected.
* Buzzer: Will keep buzzing when there is flame or gas detected.

***Software:***

* Arduino IDE: To write the code into the Arduino device and connect to the Raspbian Desktop
* Oracle Virtualbox: Host the Raspbian Desktop virtual machine.
* Fritzing: Arduino, sensors, and actuators simulation.
* Xampp: Used to host the Apache webserver on local host on the virtual machine.
* Visual Studio Code: To write all frontend and backend webserver code.

***Libraries:***

* Python: To connect to the MySQL database and input data into it. Runs the php code.
* CSS: Displays the table for the gas values table on the website.
* PHP: Enable interaction with the database such as creating tables, querying, and sending automated emails.
* HTML: Display UI
* MySQL: Database to store necessary data.

**Resources:**

<https://srituhobby.com/iot-based-gas-leakage-monitoring-system/>

<https://srituhobby.com/flame-sensor-with-led-indicator-and-buzzer/>

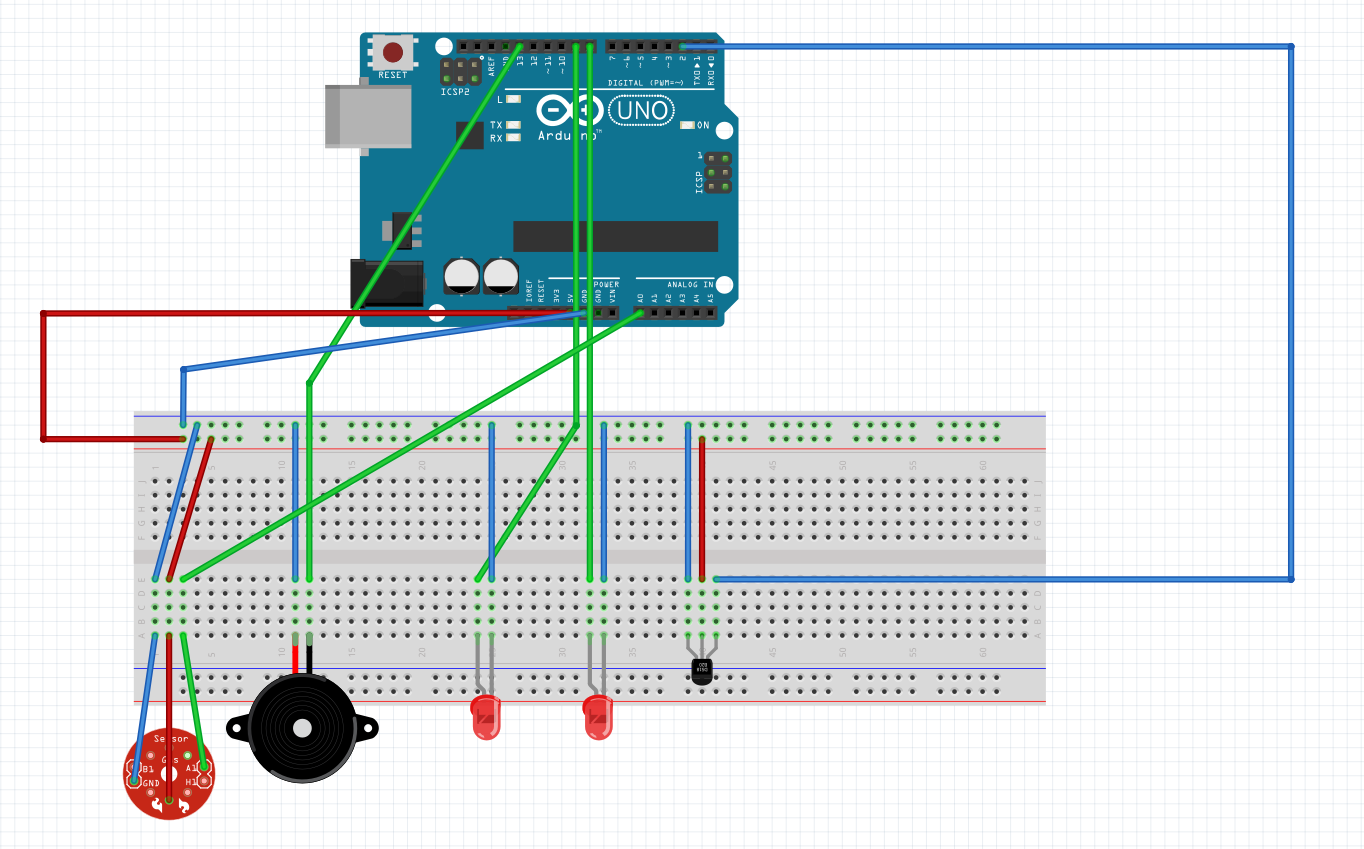
<https://www.cloudways.com/blog/send-emails-in-php-using-phpmailer/>

<https://orcacore.com/install-configure-xampp-debian-11/>

<https://support.google.com/a/answer/176600?hl=en>

**Appendix:**

Sketch of the node system simulation using fritzing



(The sensors and actuators from the diagram above may be an older version of the gadgets that are used. And one of the LED is green instead of red)

File system (Raspbian Desktop)

A diagram of a file network

Description automatically generated

Python code on console when flame is detected.

A screenshot of a computer

Description automatically generated

Python code on console when gas is detected.

A screenshot of a computer

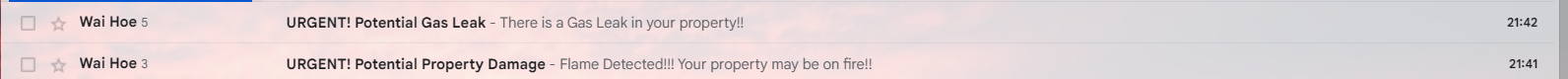
Description automatically generated

UI of the gas values inserted into a table.

A screenshot of a computer

Description automatically generated

Automated email which is sent when flame or gas is detected.



***Code:***

Assignment2IoT.ino:

int sensorPin = A0; // choose the input pin (for GAS sensor)

int flamePin = 2;

int buzzer = 13;    // choose the pin for the Buzzer

int G\_led = 8;      // choose the pin for the Green LED

int R\_led = 9;      // choose the pin for the Red LED

int read\_value;

int flameValue;

int set = 20; // Set threshold value for gas detection

const int flameThreshold = HIGH;

void setup() {

  Serial.begin(9600);

  pinMode(sensorPin, INPUT);

  pinMode(buzzer, OUTPUT);

  pinMode(R\_led, OUTPUT);

  pinMode(G\_led, OUTPUT);

}

void loop() {

  read\_value = analogRead(sensorPin);

  flameValue = digitalRead(flamePin);

  digitalWrite(G\_led, HIGH);  // Turn on green LED

  if (read\_value >= set) {

        digitalWrite(G\_led, LOW);  // Turn off green LED

    digitalWrite(buzzer, HIGH); // Turn on buzzer

    digitalWrite(R\_led, HIGH);  // Turn on red LED

    delay(500);                  // Delay for 500 milliseconds

    digitalWrite(R\_led, LOW);   // Turn off red LED

    delay(600);                  // Delay for 500 milliseconds

    digitalWrite(buzzer, LOW);  // Turn off buzzer

    Serial.println("Gas Detected");

    Serial.print("Gas Value: "); // Send gas value

    Serial.println(read\_value);  // Send gas value over serial port

    delay(1000);  // Wait before sending another message

  }

  else if(flameValue == flameThreshold){

    digitalWrite(G\_led, LOW);

    digitalWrite(buzzer, HIGH); // Turn on buzzer

    digitalWrite(R\_led, HIGH); // Turn on red LED

    delay(500);

    digitalWrite(R\_led, LOW);   // Turn off red LED

    delay(600);                  // Delay for 500 milliseconds

    digitalWrite(buzzer, LOW);  // Turn off buzzer

    Serial.println("Flame Detected");

    delay(1000);

    delay(500);                    // Delay for 500 milliseconds

    digitalWrite(R\_led, LOW);  // Turn off red LED

    digitalWrite(buzzer, LOW);  // Turn off buzzer

    delay(600);                    // Delay for 500 milliseconds

  }

  else {

    digitalWrite(R\_led, LOW);   // Turn off red LED

  }

  delay(100); // Add a short delay before next reading

}

alert.php:

<?php

use PHPMailer\PHPMailer\PHPMailer;

use PHPMailer\PHPMailer\Exception;

require 'Exception.php';

require 'PHPMailer.php';

require 'SMTP.php';

include 'settings.php'; // Include your database connection settings

// Fetch all emails from the database

$result = mysqli\_query($conn, "SELECT email FROM `user\_form`") or die('query failed');

// Iterate through each row and send email

while ($row = mysqli\_fetch\_assoc($result)) {

    $email = $row['email'];

    $mail = new PHPMailer(true);

    try {

        //Server settings

        $mail->isSMTP(); // Set mailer to use SMTP

        $mail->Host = 'smtp.gmail.com'; // Specify main and backup SMTP servers

        $mail->SMTPAuth = true; // Enable SMTP authentication

        $mail->Username = 'waihoetesting@gmail.com'; // SMTP username

        $mail->Password = 'uoro gwsv vcaf pxnr'; // SMTP password

        $mail->SMTPSecure = 'tls'; // Enable TLS encryption, `ssl` also accepted

        $mail->Port = 587; // TCP port to connect to

        //Recipients

        $mail->setFrom('waihoetesting@gmail.com', 'Wai Hoe');

        $mail->addAddress($email); // Add a recipient

        //Content

        $mail->isHTML(false); // Set email format to HTML

        $mail->Subject = 'URGENT! Potential Gas Leak';

        $mail->Body    = "There is a Gas Leak in your property!!";

        $mail->send();

        $message = 'Emails have been sent with instructions to reset your password.';

    } catch (Exception $e) {

        $message = "Message could not be sent. Mailer Error: {$mail->ErrorInfo}";

    }

}

?>

alert2.php:

<?php

use PHPMailer\PHPMailer\PHPMailer;

use PHPMailer\PHPMailer\Exception;

require 'Exception.php';

require 'PHPMailer.php';

require 'SMTP.php';

include 'settings.php'; // Include your database connection settings

// Fetch all emails from the database

$result = mysqli\_query($conn, "SELECT email FROM `user\_form`") or die('query failed');

// Iterate through each row and send email

while ($row = mysqli\_fetch\_assoc($result)) {

    $email = $row['email'];

    $mail = new PHPMailer(true);

    try {

        //Server settings

        $mail->isSMTP(); // Set mailer to use SMTP

        $mail->Host = 'smtp.gmail.com'; // Specify main and backup SMTP servers

        $mail->SMTPAuth = true; // Enable SMTP authentication

        $mail->Username = 'waihoetesting@gmail.com'; // SMTP username

        $mail->Password = 'uoro gwsv vcaf pxnr'; // SMTP password

        $mail->SMTPSecure = 'tls'; // Enable TLS encryption, `ssl` also accepted

        $mail->Port = 587; // TCP port to connect to

        //Recipients

        $mail->setFrom('waihoetesting@gmail.com', 'Wai Hoe');

        $mail->addAddress($email); // Add a recipient

        //Content

        $mail->isHTML(false); // Set email format to HTML

        $mail->Subject = 'URGENT! Potential Property Damage';

        $mail->Body    = "Flame Detected!!! Your property may be on fire!!";

        $mail->send();

        $message = 'Emails have been sent with instructions to reset your password.';

    } catch (Exception $e) {

        $message = "Message could not be sent. Mailer Error: {$mail->ErrorInfo}";

    }

}

?>

settings.php:

<?php

    $host = "localhost";

    $user = "root";

    $pwd = "";

    $sql\_db = "authentication\_test";

    $conn = mysqli\_connect($host, $user, $pwd, $sql\_db);

    if (!$conn) {

        die("Connection failed: " . mysqli\_connect\_error());

    }

creategastable.php:

<?php

include 'settings.php'; // Include your database connection settings

// Create the user\_form table if it doesn't exist

$query = "

    CREATE TABLE gas\_values (

    id INT AUTO\_INCREMENT PRIMARY KEY,

    value INT NOT NULL,

    created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

    )

";

mysqli\_query($conn, $query);

if(mysqli\_error($conn)) {

    echo "Error creating table: " . mysqli\_error($conn);

} else {

    echo "Table 'user\_form' created successfully!";

}

gasvalues.php:

<?php

include 'settings.php'; // Include your database connection settings

// Check if the form is submitted

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

    // Retrieve the selected date, start time, and end time from the form

    $selected\_date = $\_POST['selected\_date'];

    $start\_time = isset($\_POST['start\_time']) ? $\_POST['start\_time'] : '';

    $end\_time = isset($\_POST['end\_time']) ? $\_POST['end\_time'] : '';

    // Construct the datetime format for MySQL query

    $start\_datetime = $selected\_date . ' ' . $start\_time . ':00';

    $end\_datetime = $selected\_date . ' ' . $end\_time . ':00';

    // Query to select gas values for the selected date and time range

    if (!empty($start\_time) && !empty($end\_time)) {

        // If both start time and end time are provided, filter by the time range

        $query = "SELECT \* FROM gas\_values WHERE created\_at BETWEEN '$start\_datetime' AND '$end\_datetime'";

    } else {

        // If start time or end time is empty, filter by date only

        $query = "SELECT \* FROM gas\_values WHERE DATE(created\_at) = '$selected\_date'";

    }

    $result = mysqli\_query($conn, $query);

    if (!$result) {

        die('Query failed: ' . mysqli\_error($conn));

    }

}

?>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Gas Values</title>

    <style>

        table {

            border-collapse: collapse;

            width: 50%;

            margin: auto;

        }

        th, td {

            border: 1px solid #dddddd;

            text-align: left;

            padding: 8px;

        }

        th {

            background-color: #f2f2f2;

        }

    </style>

</head>

<body>

    <h2>Gas Values</h2>

    <form method="post">

        <label for="selected\_date">Select Date:</label>

        <input type="date" id="selected\_date" name="selected\_date">

        <br><br>

        <label for="start\_time">Start Time:</label>

        <input type="time" id="start\_time" name="start\_time">

        <label for="end\_time">End Time:</label>

        <input type="time" id="end\_time" name="end\_time">

        <br><br>

        <button type="submit">Filter</button>

    </form>

    <?php if (isset($result)) { ?>

        <table>

            <thead>

                <tr>

                    <th>ID</th>

                    <th>Gas Value</th>

                    <th>Timestamp</th>

                </tr>

            </thead>

            <tbody>

                <?php while ($row = mysqli\_fetch\_assoc($result)) { ?>

                    <tr>

                        <td><?php echo $row['id']; ?></td>

                        <td><?php echo $row['value']; ?></td>

                        <td><?php echo $row['created\_at']; ?></td>

                    </tr>

                <?php } ?>

            </tbody>

        </table>

        <?php mysqli\_free\_result($result); ?>

    <?php } ?>

</body>

</html>

alertscript2.py:

import subprocess

import time

import serial

import mysql.connector

import webbrowser

ser = serial.Serial('/dev/ttyS0', 9600)

try:

# Connect to MySQL database

db\_connection = mysql.connector.connect(

host="localhost",

user="root",

password="",

database="authentication\_test"

)

db\_cursor = db\_connection.cursor()

print("Connected to database")

except mysql.connector.Error as e:

print("Error connecting to database:", e)

exit(1)

# Read data from serial port

try:

while True:

# Read line from serial

data = ser.readline().decode().strip()

if data.startswith("Flame Detected"):

print("Flame Detected!")

webbrowser.get('google-chrome').open\_new("http://localhost/Iot/alert2.php")

# Check if gas value is received

if data.startswith("Gas Value:"):

gas\_value = int(data.split(":")[1]) # Extract gas value

# Save gas value to database

query = "INSERT INTO gas\_values (value) VALUES (%s)"

db\_cursor.execute(query, (gas\_value,))

db\_connection.commit()

print("Gas value inserted into database:", gas\_value)

# Open the URL in Google Chrome using subprocess

try:

subprocess.Popen(["google-chrome", "http://localhost/Iot/alert.php"])

print("Alert webpage opened")

except Exception as e:

print("Error opening web browser:", e)

# Add a small delay before reading the serial port again

time.sleep(1)

except KeyboardInterrupt:

print("Exiting...")

# Close serial port

ser.close()

# Close database connection

db\_cursor.close()

db\_connection.close()

**GitHub Link To All Files:**

https://github.com/Macaroneyy/SWE30011