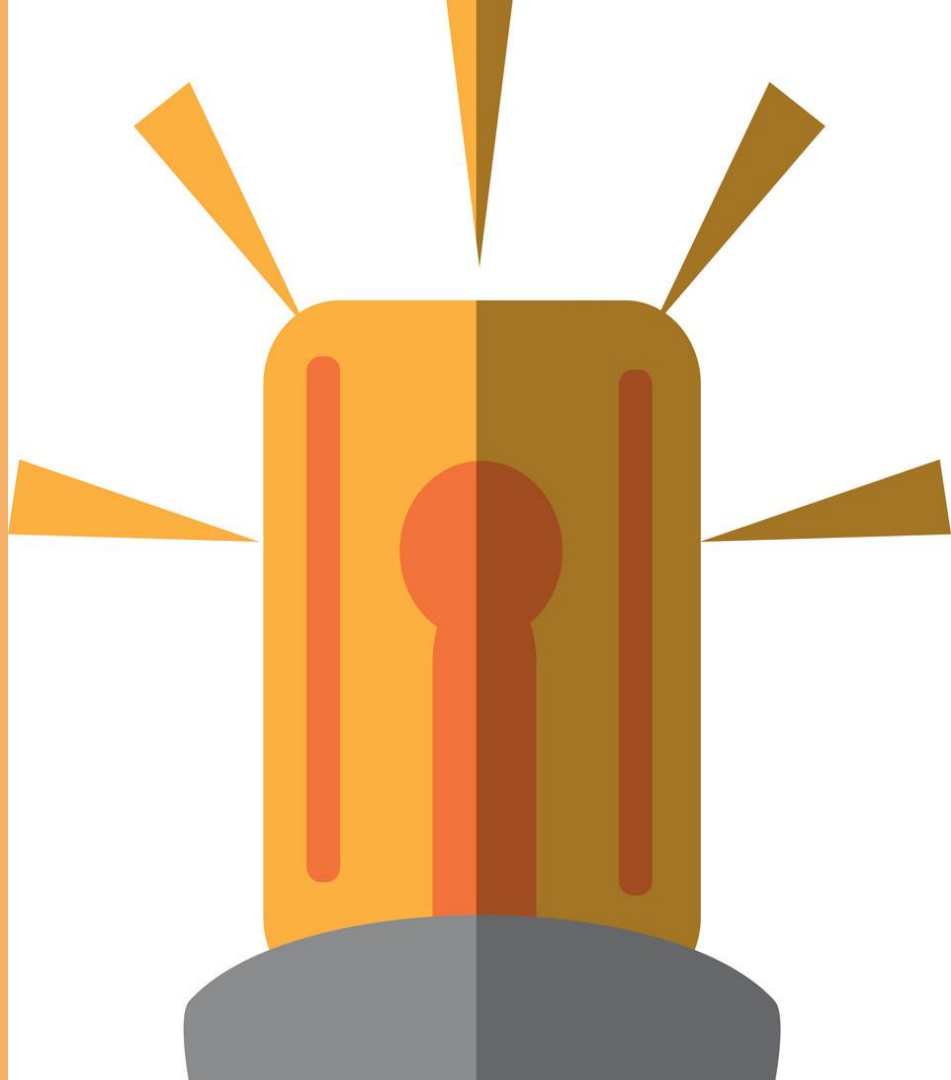
Home Security System

Amy Fan and Naomi Philips



Background

The Alarm System is created to provide an extra layer of security towards the user.

For this alarm system, it was decided to have key features such as an LED light, OLED Display, and Buzzer to notify the user of any potential threats that may occur. This was created to be a home-based system and the system is triggered by motion detection!

In order to scale the project, we would most likely need to add:

- Louder buzzer/sound and larger LED for easier alerting.
- Larger OLED Display for viewing.
- Tweaked code that has a more intricate system related to enable/disable options.

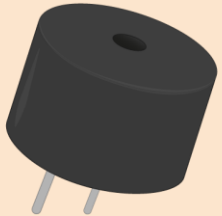
Project Objectives/Features

Objectives:

- Provide security measures with alarm system.
- Create an alarm system with sound, light, and display.
- Display a scenario where the user would be able to use the alarm system.
- Provide multiple forms of alerting the user of the alarm system being triggered.

Features of Alarm System:

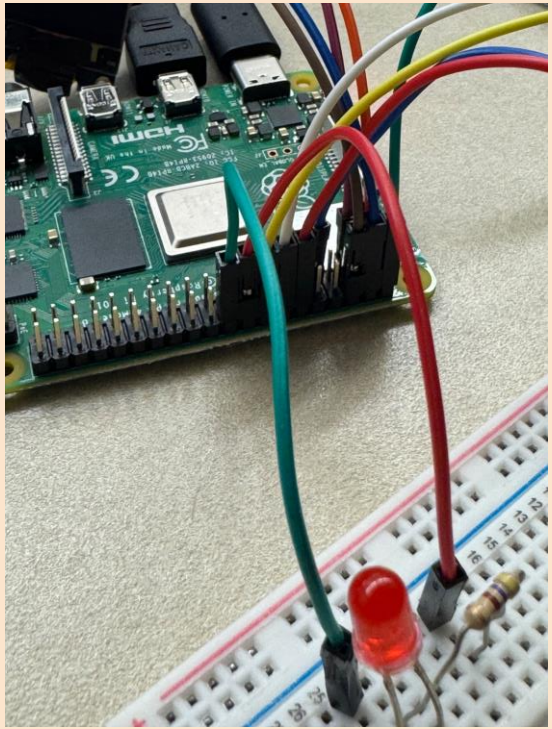
- Motion sensor to detect movement.
- OLED Graphical Display, LED light, and buzzer to alert the alarm owner.
- GUI for a user friendly interface.
- Asks for password input (1111).
- Confirms password input, only allows enabling or disabling of alarm after password is entered.
- 2 enable options, one with a 10 second delay and one without.

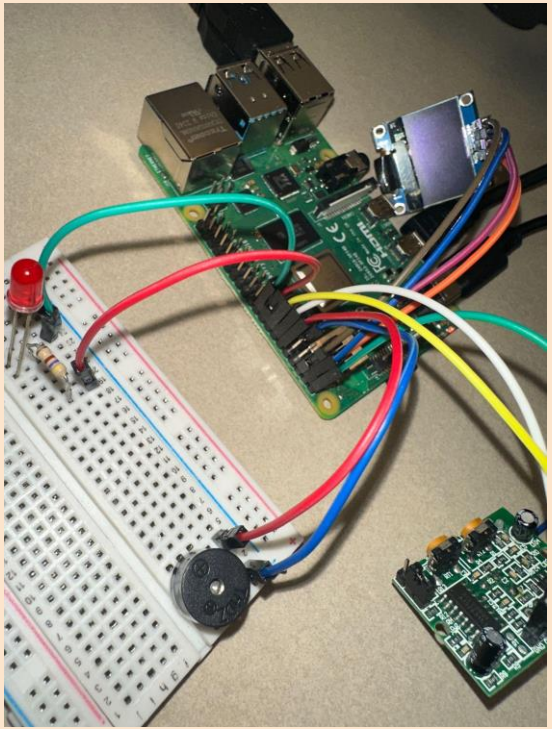


Hardware Components

The hardware used consisted of:

- Jumper wires.
- PIR motion detector module.
- Piezoelectric buzzer.
- I2C OLED display 128×64 pixels.
- Breadboard.
- Red LED.
- 470Ω resistor.





Other Components

Recipes Used:

13.9 Detecting Movement - Used to detect motion.

11.1 Connecting an LED.

15.4 Using an OLED Graphical Display - Used to notify the user of when movement has been detected and and display “Intruder” via the OLED display.

16.7 Making a Buzzing Sound - When movement is detected, it will let the user know someone is nearby by making a sound.

Software Imported:

```
import board
import digitalio
import time
from gpiozero import Buzzer, MotionSensor,
LED
from guizero import App, Text, TextBox,
PushButton, Window
from PIL import Image, ImageDraw, ImageFont
import adafruit_ssd1306
from time import sleep
from datetime import datetime, timedelta
```

Project Approach

We approached the project by creating a pretend scenario in which the alarm system would be used.

In this case, we used a pretend house with an owner and an intruder. The intruder and homeowner were at different levels in the house to represent how a typical break-in would occur.

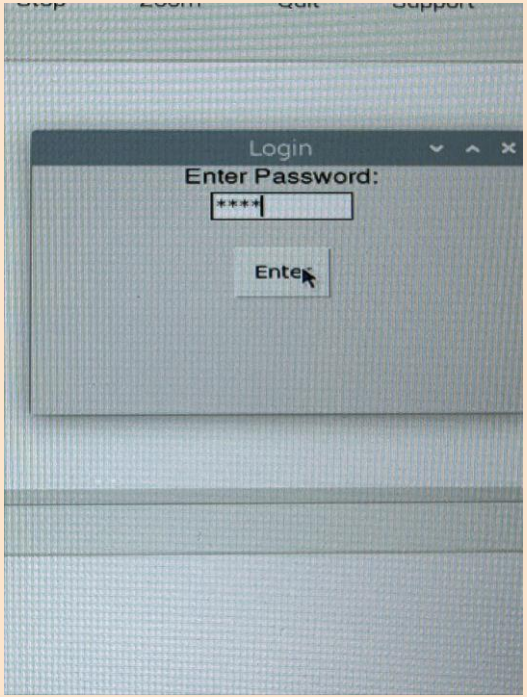
Scenario:

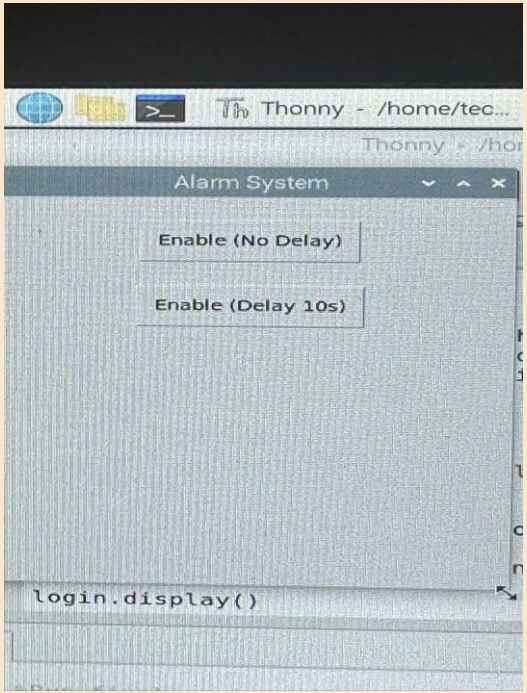
1) The owner of the alarm system turns on the alarm remotely from the GUI.

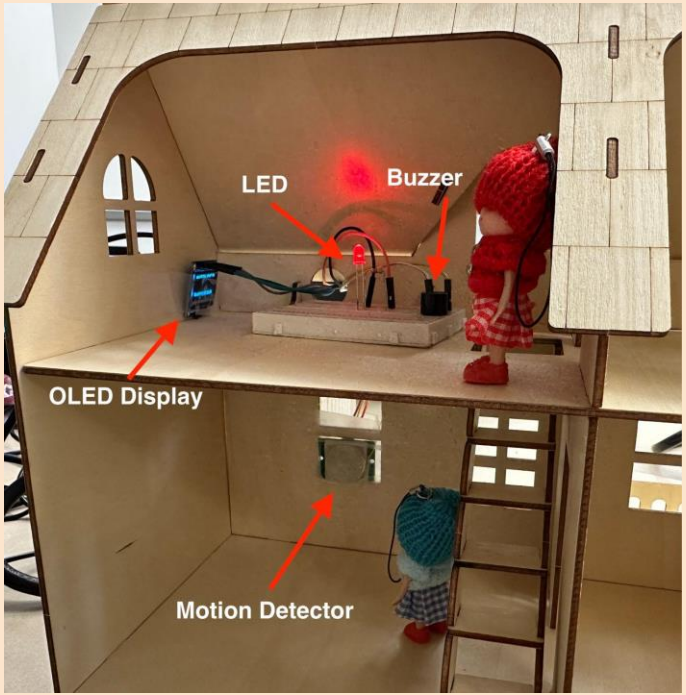
1) The motion detector gets triggered when the intruder is detected.

1) The LED, buzzer, and OLED display indicate that there is an intruder by making noise, turning on the red light, and displaying that there is an intruder on the OLED display, along with the date and time the motion was detected.

Results


Login
Enter Password:
****
Enter


Thonny - /home/tec...
Alarm System
Enable (No Delay)
Enable (Delay 10s)
login.display()


LED
Buzzer
OLED Display
Motion Detector

Conclusions: What Went Wrong, Difficulties, & Further Improvements

Issues with LED Light and Buzzer:

The LED from 11.1 was having issues “lighting up” and the buzzer wasn’t able to make any sound.

There were even issues running the initial lab/recipes and nothing was able to work.

To overcome these both, we first attempted to redo our circuit and then asked the professor for assistance.

It turned out to be an issue with gpiozero as it had to be reinstalled.

Broken SD Card and Lost Code:

In the last week of working on the final project, our SD card holding the RPi software stopped working.

To solve this issue, we received a new SD card and redownloaded the RPi software.

The code had to be rewritten and tested again to work with the circuit.

Further Improvements:

Unfortunately, with the restrictions on guizero, it proved to be quite difficult setting up the GUI for the program.

One major improvement we would like to add would be adding a disable/stop alarm button to the GUI itself, rather than having to disable the alarm from the terminal.

Our Code (Part 1)

To make sure the correct circuit was created along with the program, we mixed code from the recipes we choose and this was the outcome.

```
import board
import digitalio
import time
from gpiozero import Buzzer, MotionSensor, LED
from guizero import App,TExt, TextBox, PushButton, Window
from PIL import Image, ImageDraw, ImageFont
import adafruit_ssd1306
from time import sleep
from datetime import datetime, timedelta



# declare necessary variables
pir = MotionSensor(18)
buzzer = Buzzer (23)
led + Led(24)
12c = board.I2C ()
```

```
# set up display
disp = adafruit_ssd1306.SSD1306_IC2(128,64,i2c,addr=0x3C)
small_font = ImageFont.truetype ('Freesans.tt f' , 12)
large_font = ImageFont.truetype ('Freesans,ttf ' , 23)
disp.show()
disp.fill(0)
disp.show()

width = disp.width
height = disp.height
image = Image.new('1' , (width,height))
draw = ImageDraw.Draw(Image)
alert_message = 'Intruder'

def display_message(top_line, line_2):
     draw.rectangle((0,0,width,height),outline = 0, fill = 0)
     draw.text ((0,0), top_line , font = large_font, fill =255)
     draw.text((0,50), line_2, font=small-font, fill = 255)
     disp.image (image)
     disp. show()
```

Our Code (Part 2)

```
def start_alarm():
     while True:
          pir.wait_for_motion()
          buzzer.beep(on_time = 1/200, off_time =1/200)
          led.on()
          print (“Motion Detected”)
          sleep(0.1)
          now = datetime.now()
          date = ‘{:%B %d}’. format(now)
          time = ‘{:%H: %M: %S}’. format(now)
          date_time = date + “ | “ + time
          display_message(alter_message,date_time)

          sleep(0.1)

          stop_alarm = input(“\nEnter password to stop the
program :/n ”)
          if stop_alarm == “1111”;
               exit()
                 else:
                              print(“Wrong password. Try again.”)

def check_password():
     if password_box.value ==”1111”:
          login.hide()
          window.show()
   else:
        message.value = “Incorrect Password. Try Again”
```

```
def start():
    window.hide()
    start_alarm()

# initialize login screen, prompts user for password
login = App(title=”Login”, width-300, height=200)
Text(login, text = “Enter Password:”)
password_box = TextBox(login, hide_text = True)
message = Text(login, text=””)
PushButton(login, text=”Enter”, command= check_password)

# second window
window = Window(login, title = “Alarm System”, width =
200, height = 200)
window.hide()
status = Text(window,text=””)
enable_button = PushButton(window, text=”Enable”,
command=start)

login.display()
```

Questions?