**Introduction:**

Our Theft Detection System, powered by a Raspberry Pi, is a smart security solution combining sensors, a camera, and Wi-Fi connectivity. It operates in two modes: active and standby, using piezoelectric sensors to trigger an alarm in case of theft. The unique feature involves a camera capturing images for more detailed information.

**Project Description:**

**Smart Security Setup:**

The project uses a Raspberry Pi as the main brain and piezoelectric sensors strategically placed to detect theft. A camera takes pictures when sensors are triggered, and a buzzer sounds an alert.

Images are sent over Wi-Fi for further analysis.

**Easy Code Structure:**

We've organized the code using Object-Oriented Programming (OOP) principles, making it easy to understand, modify, and expand. Inheritance and Polymorphism are used for efficient coding. **Balancing Security and Energy:**

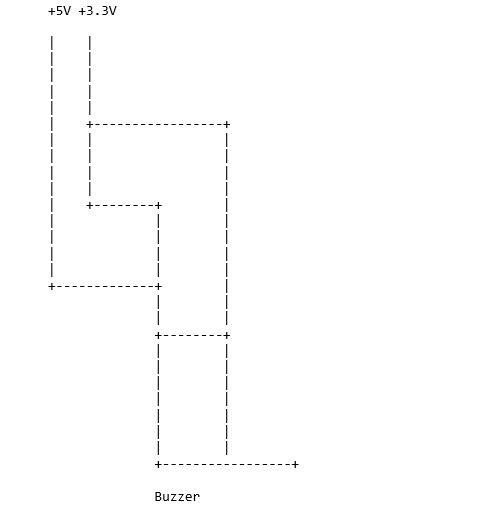
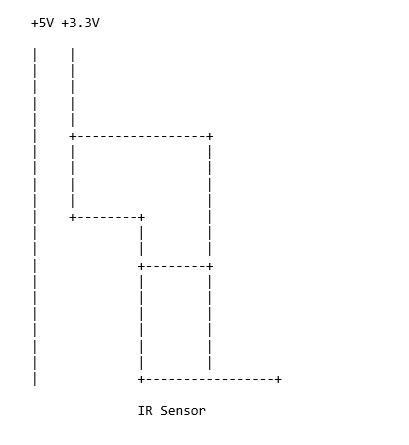
The system is designed to save energy when not in use (standby mode) and actively monitor during potential threats (active mode). The added camera improves theft detection accuracy and provides valuable image data.

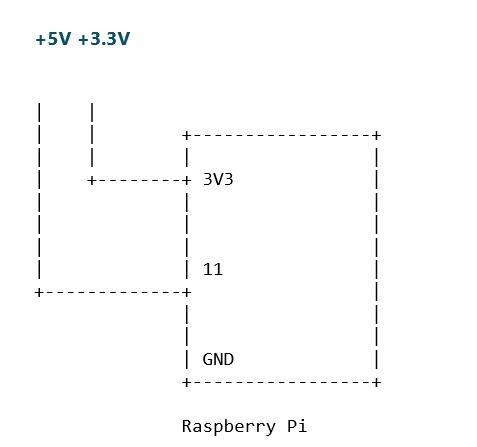
# Report

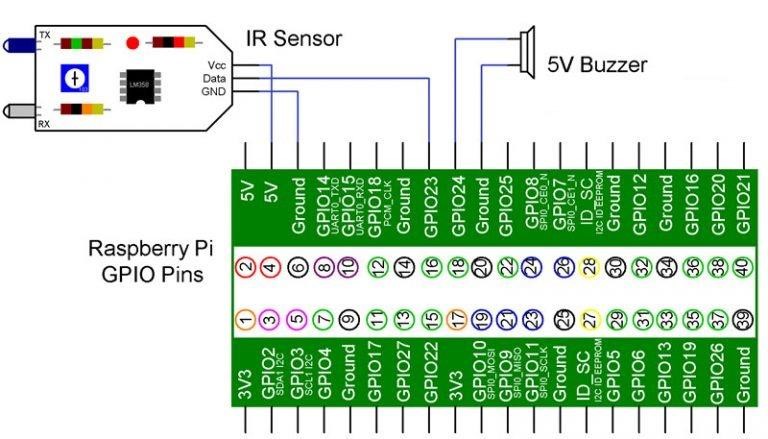
**1.1 Data Sheet / Pins Configuration:**

* **RPi.GPIO Library:** Used for interfacing with the Raspberry Pi's GPIO pins.
* **Pin Configuration:**
* **GPIO Pin 3:** Input pin connected to the motion sensor.
* **GPIO Pin 5:** Output pin connected to an LED to indicate motion detection.

* 1. **Circuit Diagram:**







* 1. **Methods:**
* **\_init\_():** Initializes the GPIO pins, setting GPIO Pin 3 as input and GPIO Pin 5 as output.
* **detect\_motion():** Continuously checks the input from the motion sensor. If motion is detected, it turns on the LED, prints a message, captures an image, and waits for 3 seconds before checking again.
* **capture\_image():** Captures an image using the Raspberry Pi camera module and saves it with a timestamped filename.

# Limitation

* **Real-Time Processing:** This project captures images when motion is detected, but it doesn't stream or transmit the images in real-time.
* **Single Motion Sensor:** It supports only one motion sensor connected to GPIO Pin 3. Multiple sensors would require additional code for handling.
* **No Motion Sensitivity Adjustment:** There's no provision for adjusting the sensitivity of the motion sensor.

# Future Extensions / Advancements

* **Real-Time Streaming:** Implementing real-time streaming of captured images to a server or cloud storage.
* **Multiple Sensors:** Enhancing the project to support multiple motion sensors for broader area coverage.
* **Motion Sensitivity Adjustment:** Adding functionality to adjust the sensitivity of the motion sensor based on environmental conditions.

# Python Source Code

|  |
| --- |
| **Source Code** |
| import RPi.GPIO as GPIO  import time import os  import datetime    class MotionDetector: def \_\_init\_\_(self):  GPIO.setmode(GPIO.BOARD)  GPIO.setup(3, GPIO.IN) |
| GPIO.setup(5, GPIO.OUT)    def detect\_motion(self): while True:  val = GPIO.input(3) if val == 1:  GPIO.output(5, GPIO.LOW) print("Motion Not Detected") else:  GPIO.output(5, GPIO.HIGH) print("Motion Detected") self.capture\_image()  time.sleep(3)    def capture\_image(self):  timestamp = datetime.datetime.now().strftime("%Y%m%d%H%M%S") image\_name = f"image\_{timestamp}.jpg"  os.system(f"raspistill -o {image\_name}") # Capture image using the camera module    if \_\_name\_\_ == "\_\_main\_\_":  motion\_detector = MotionDetector()  motion\_detector.detect\_motion() |

|  |
| --- |
| **Output** |
|  |

# Implementation of OOPs Concept

* **Class:** The entire functionality is encapsulated within a class MotionDetector.
* **Encapsulation:** Methods like detect\_motion() and capture\_image() encapsulate related functionality.
* **Abstraction:** The user doesn't need to know the internal workings of GPIO pins or camera commands; they interact with high-level methods of the class.
* **Constructor:** The \_\_init\_\_() method initializes the GPIO pins, providing a clean way to set up the environment.
* **Instance Method:** Both detect\_motion() and capture\_image() are instance methods, operating on the specific instance of MotionDetector.
* **Conditional Execution:** The if \_\_name\_\_ == "\_\_main\_\_": block ensures that the program runs only when the script is executed directly, allowing the class to be imported without running the main code.