WEAPON DETECTION SYSTEM

A MINI PROJECT REPORT

18CSC305J - ARTIFICIAL INTELLIGENCE

Submitted by

ABHISHEK SINGH [RA2011026010098]
VISHVESH BHARDWAJ [RA2011026010109]
RAGHAV KAPOOR [RA2011026010117]

Under the guidance of

Dr. M.S. Abirami

Associate Professor, Department of Computer Science and Engineering

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

of

FACULTY OF ENGINEERING AND TECHNOLOGY

SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
Deemed to be University u/s 3 of UGC Act, 1956

S.R.M. Nagar, Kattankulathur, Chengalpattu District

MAY 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that Mini project report titled **"WEAPON DETECTION SYSTEM"** is the bonafide work of **ABHISHEK SINGH (RA2011026010098), VISHVESH BHARDWAJ (RA2011026010109), RAGHAV KAPOOR (RA2011026010117)** who carried out the minor project under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SRM IST * Department of Computational Intelligence * Kattankulathur 603 203

10/5/23

SIGNATURE

Dr. M.S. Abirami
GUIDE
Associate Professor
Department of Computational Intelligence

SIGNATURE

Dr. R.Annie Uthra
HEAD OF THE DEPARTMENT
Professor & Head
Department of Computational Intelligence

ABSTRACT

A weapon detection system is an automated surveillance solution designed to detect and alert security personnel of the presence of firearms or other dangerous weapons in public places such as airports, schools, and shopping malls. The system uses various technologies such as artificial intelligence, computer vision, and machine learning algorithms to identify suspicious objects and alert authorities in real-time. The detection system is usually integrated with security cameras and other sensors that capture and analyze data to detect the presence of a weapon. With the growing concern of mass shootings and terror attacks, weapon detection systems have become a crucial tool for enhancing public safety and security. This abstract outline the key features and benefits of a weapon detection system and its importance in preventing violent incidents in public places.

TABLE OF CONTENTS

S.no.	Contents	Page No.
1	Abstract	3
2	List of Figures	5
3	Introduction	6
4	Literature Survey	7
5	System Architecture & Design	8
6	Methodology	10
7	Coding and Testing	11
8	Screenshots and Results	16
9	Conclusions and future Enhancements	17
10	References	18
11	GitHub Links	18

LIST OF FIGURES

S. No.	Name Of Figure	Page No.
1	System Architecture & Design	8
2	Yolov4 – Object Detection	8
3	Login Window Module	11
4	Detection Module	12
5	Detection Window Module	13
6	Setting Window Module	14
7	Main Module	15
8	Screenshots & Results	16

INTRODUCTION

The growing incidence of mass shootings and terrorist attacks in public places has highlighted the need for enhanced security measures to protect individuals and prevent such incidents from occurring. One of the critical tools in enhancing public safety and security is the weapon detection system. This automated surveillance solution is designed to detect the presence of firearms and other dangerous weapons in public places such as schools, airports, and shopping malls.

Weapon detection systems use advanced technologies such as artificial intelligence, computer vision, and machine learning algorithms to analyze data captured by security cameras and other sensors in real-time. The system alerts security personnel when it identifies a potential threat, allowing them to take prompt action to prevent violent incidents from occurring.

The benefits of weapon detection systems are numerous, including improved public safety, faster response times, and enhanced situational awareness. In this paper, we will explore the features and benefits of weapon detection systems, their effectiveness in preventing violent incidents, and the challenges associated with their implementation.

LITERATURE SURVEY

Various studies have been conducted to investigate the effectiveness and potential of weapon detection systems in enhancing public safety and preventing violent incidents.

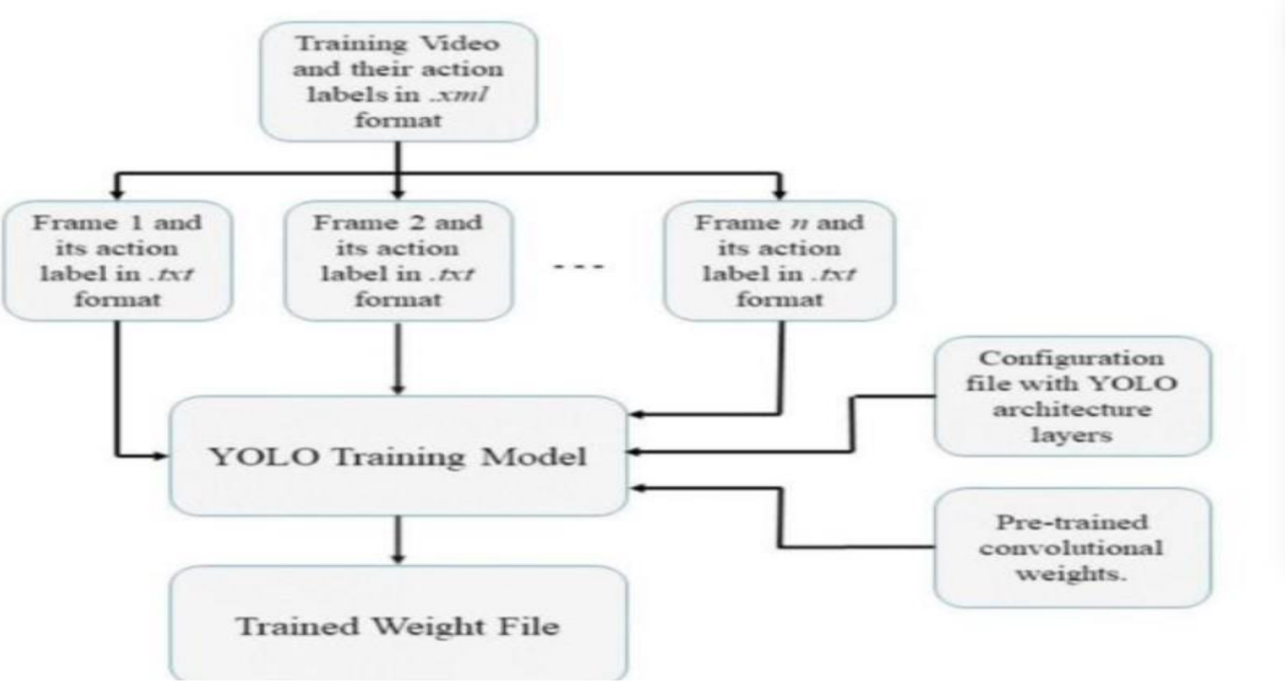
A study conducted by researchers at the University of California, Berkeley, evaluated the accuracy of a weapon detection system based on artificial intelligence and deep learning algorithms. The study found that the system was highly effective in detecting firearms and other weapons in real-time, with an accuracy rate of over 90%.

Another study published in the International Journal of Scientific Research in Computer Science and Engineering examined the challenges associated with the implementation of weapon detection systems in public places. The study highlighted the need for effective training and education of security personnel to ensure the accurate identification of potential threats and the prompt response to alerts generated by the system.

A review article published in the Journal of Electronic Imaging examined the various technologies used in weapon detection systems, including thermal imaging, x-ray imaging, and millimeter-wave imaging. The article concluded that the combination of multiple technologies, along with machine learning algorithms, can significantly enhance the accuracy and effectiveness of weapon detection systems.

Overall, the literature suggests that weapon detection systems have the potential to enhance public safety and prevent violent incidents in public places. However, their implementation requires careful consideration of the associated costs, challenges, and ethical implications, including privacy concerns and potential biases in the system's algorithm.

SYSTEM ARCHITECTURE AND DESIGN


Training Video and their action labels in .xml format
Frame 1 and its action label in .txt format
Frame 2 and its action label in .txt format
- - -
Frame n and its action label in .txt format
Configuration file with YOLO architecture layers
YOLO Training Model
Pre-trained convolutional weights.
Trained Weight File

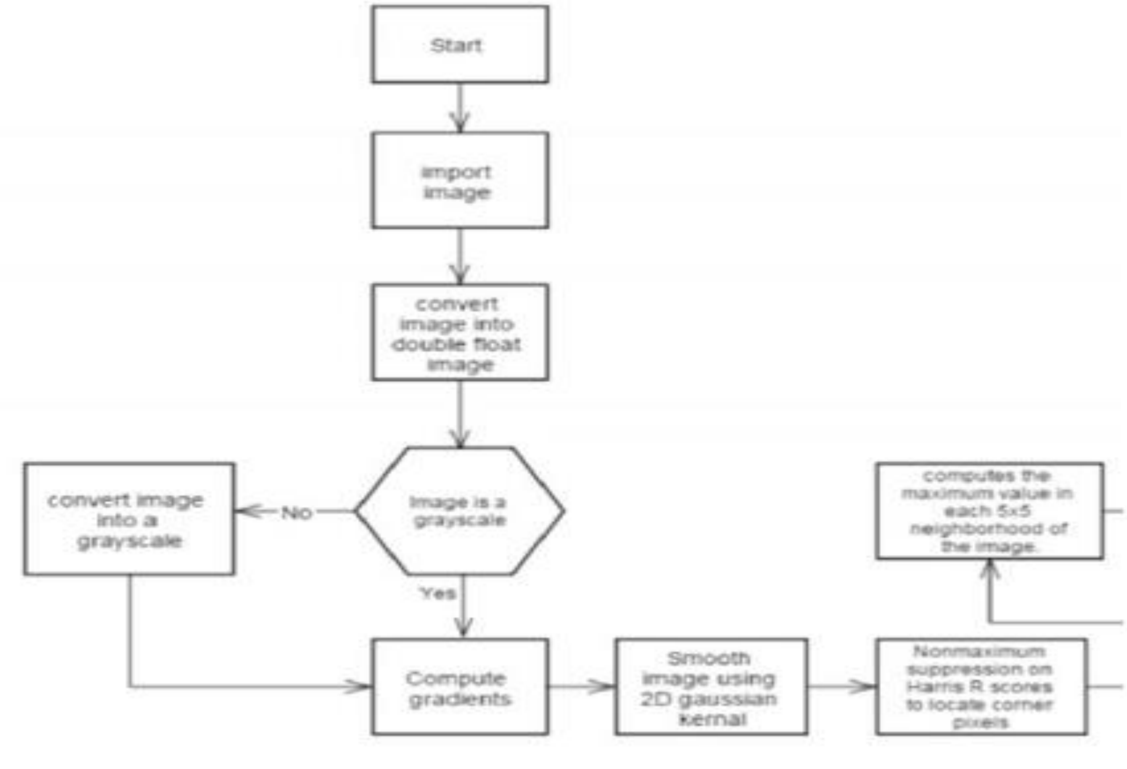

Start
import image
convert image into double float image
Image is a grayscale
No
convert image into a grayscale
Yes
Compute gradients
Smooth image using 2D gaussian kernal
Nonmaximum suppression on Harris R scores to locate corner pixels
computes the maximum value in each 5x5 neighborhood of the image.

Fig 1: Yolov4 Object Detection Algorithm

A weapon detection system is an application of object detection, which aims to identify firearms or other dangerous weapons in images or videos. The system architecture and design of a weapon detection system typically include the following components:

Dataset:

The first step in designing a weapon detection system is to gather a dataset of images or videos containing firearms and other dangerous weapons. This dataset is used to train the object detection model.

Object Detection Model:

The object detection model is the core component of the weapon detection system. There are several state-of-the-art object detection models available, such as YOLO, Faster R-CNN, and SSD. These models are typically trained on the weapon dataset to learn to detect firearms and other dangerous weapons in images or videos.

Preprocessing:

Before the input images or videos are fed into the object detection model, they are preprocessed to enhance the quality of the images and extract relevant features. This may include resizing the images, normalizing pixel values, and applying filters to remove noise.

Post-processing:

Once the object detection model has identified the presence of firearms or other dangerous weapons in the input images or videos, a post-processing step is typically applied to refine the results. This may include filtering out false positives, tracking the detected objects across frames, and estimating the location and size of the weapons in the image.

User Interface:

Finally, a user interface is designed to allow users to interact with the weapon detection system. This may include a graphical interface that displays the results of the object detection model, along with additional information about the detected weapons, such as the type of firearm or the brand of the weapon. The user interface may also include options for adjusting the sensitivity of the detection model, selecting different types of weapons to detect, and setting up notifications or alerts when a weapon is detected.

Overall, the system architecture and design of a weapon detection system involves a combination of machine learning techniques, image processing algorithms, and user interface design to create an effective and user-friendly application for detecting firearms and other dangerous weapon

METHODOLOGY

The methodology for a weapon detection system typically involves the following steps:

1. Data Acquisition: The first step is to acquire data from the sensors. This data can include images, video, or other forms of sensor data.

2. Pre-Processing: Once the data is acquired, it may need to be pre-processed to remove any noise or artifacts that could interfere with the detection process. This step can involve techniques such as filtering, normalization, or image enhancement.

3. Feature Extraction: The next step is to extract features from the pre-processed data that are relevant to weapon detection. This step can involve techniques such as edge detection, texture analysis, or object recognition.

4. Classification: The extracted features are then used to train a machine learning algorithm to recognize weapons. The classification algorithm can be a simple rule-based system or a more complex neural network.

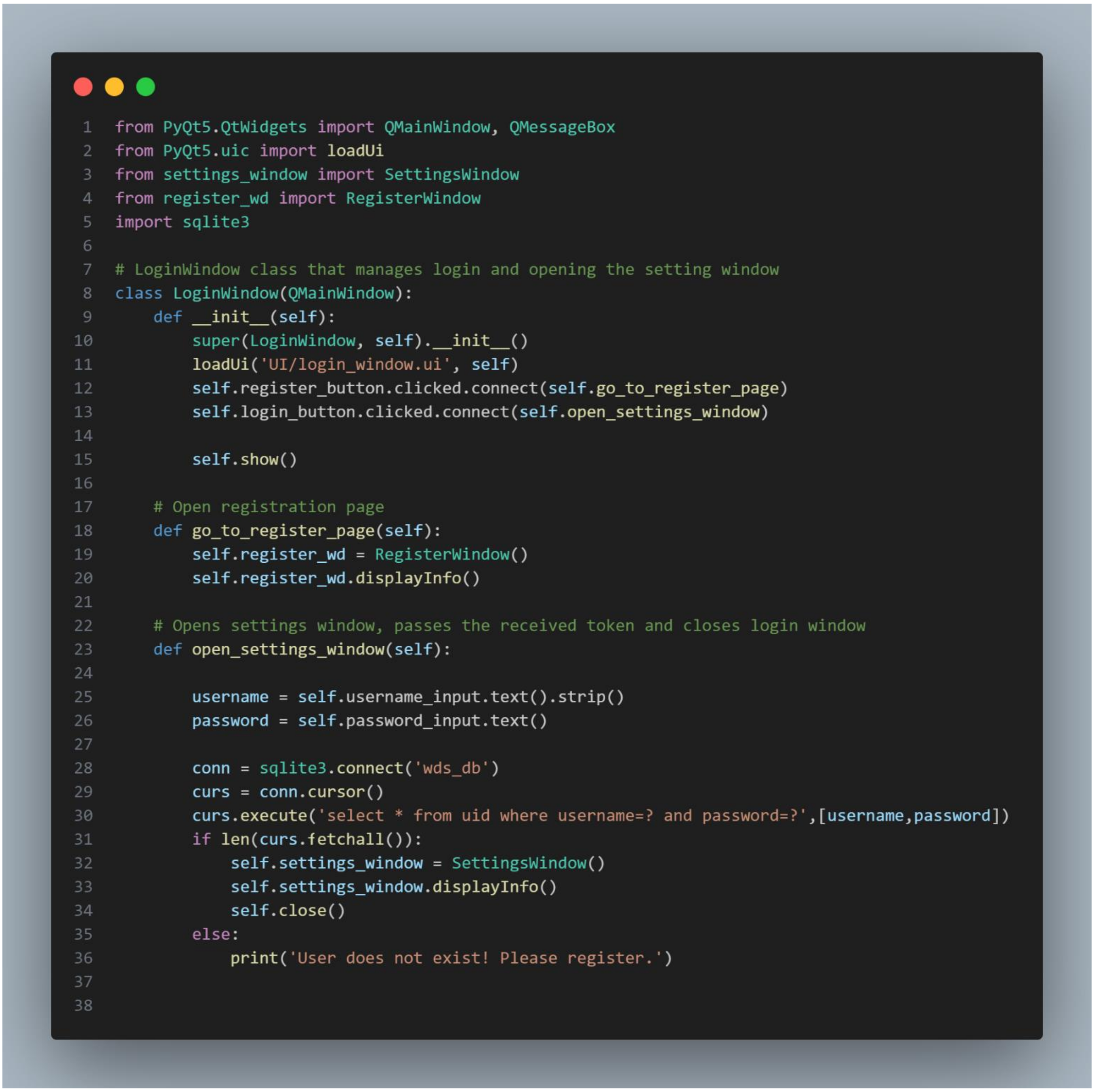
5. Testing and Evaluation: Once the classification algorithm is trained, it is tested on a set of validation data to evaluate its accuracy and performance. This step can involve techniques such as cross-validation or receiver operating characteristic (ROC) analysis.

6. Deployment: Finally, the weapon detection system is deployed in a real-world environment and monitored for accuracy and false positives. The system's performance can be evaluated and adjusted as needed.

Overall, the methodology for a weapon detection system involves a combination of data acquisition, pre-processing, feature extraction, classification, testing and evaluation, and deployment. The process is iterative and may require adjustments to improve the system's accuracy and performance in different environments and under varying conditions.

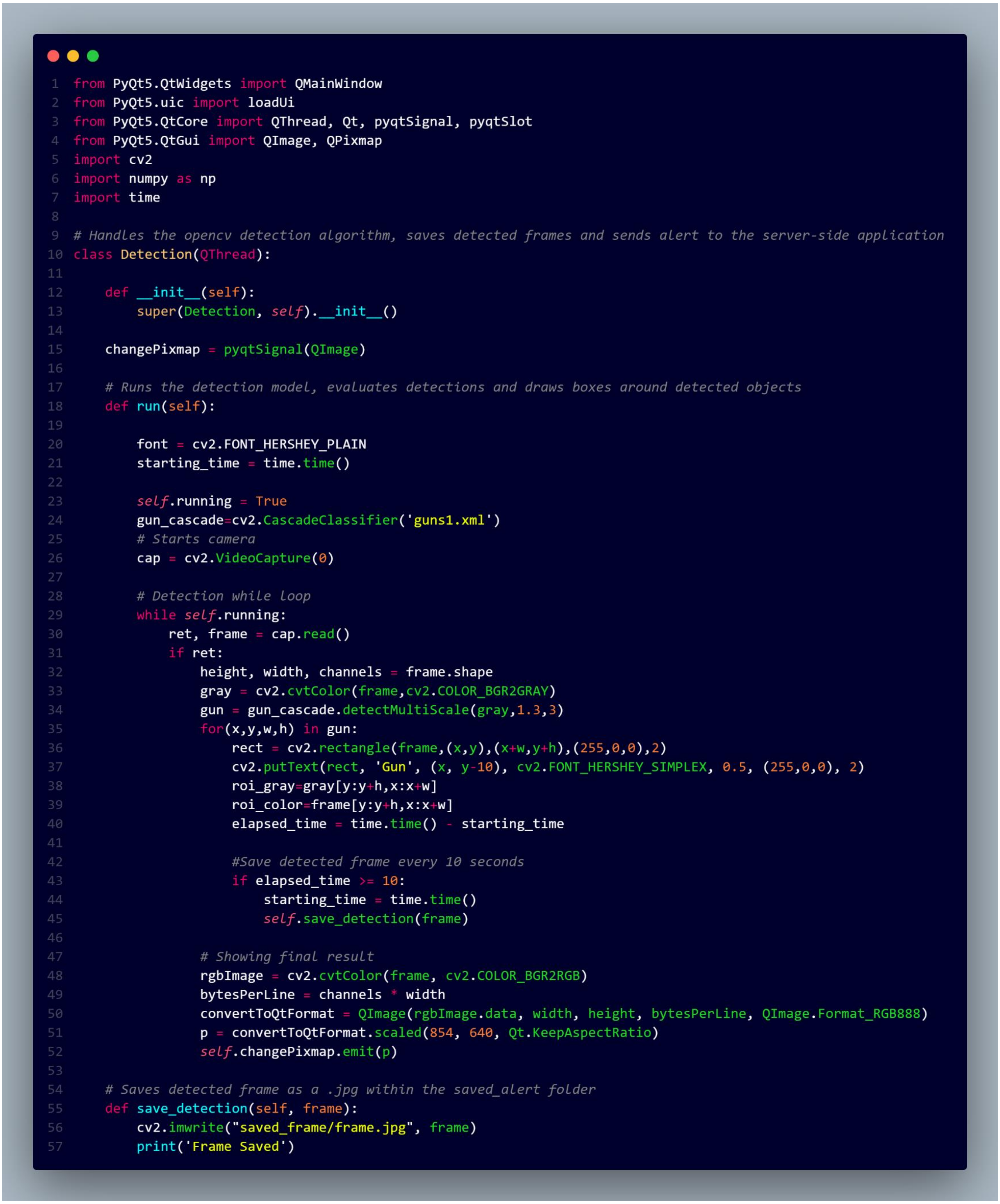
CODING AND TESTING

1.) LOGIN WINDOW MODULE



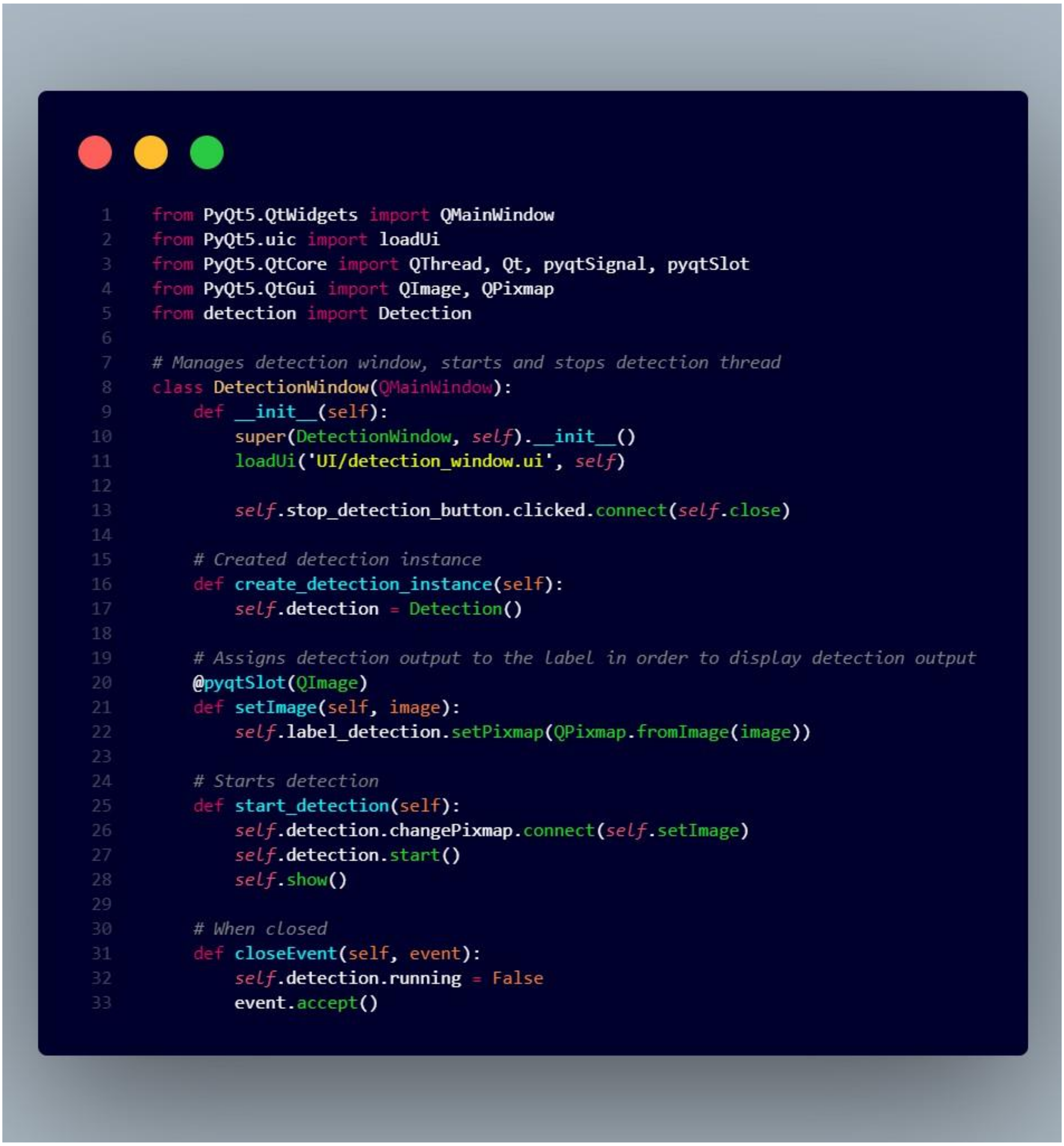
```python
from PyQt5.QtWidgets import QMainWindow, QMessageBox
from PyQt5.uic import loadUi
from settings_window import SettingsWindow
from register_wd import RegisterWindow
import sqlite3

# LoginWindow class that manages login and opening the setting window
class LoginWindow(QMainWindow):
    def __init__(self):
        super(LoginWindow, self).__init__()
        loadUi('UI/login_window.ui', self)
        self.register_button.clicked.connect(self.go_to_register_page)
        self.login_button.clicked.connect(self.open_settings_window)

        self.show()

    # Open registration page
    def go_to_register_page(self):
        self.register_wd = RegisterWindow()
        self.register_wd.displayInfo()

    # Opens settings window, passes the received token and closes login window
    def open_settings_window(self):

        username = self.username_input.text().strip()
        password = self.password_input.text()

        conn = sqlite3.connect('wds_db')
        curs = conn.cursor()
        curs.execute('select * from uid where username=? and password=?',[username,password])
        if len(curs.fetchall()):
            self.settings_window = SettingsWindow()
            self.settings_window.displayInfo()
            self.close()
        else:
            print('User does not exist! Please register.')
```

2.) DETECTION MODULE



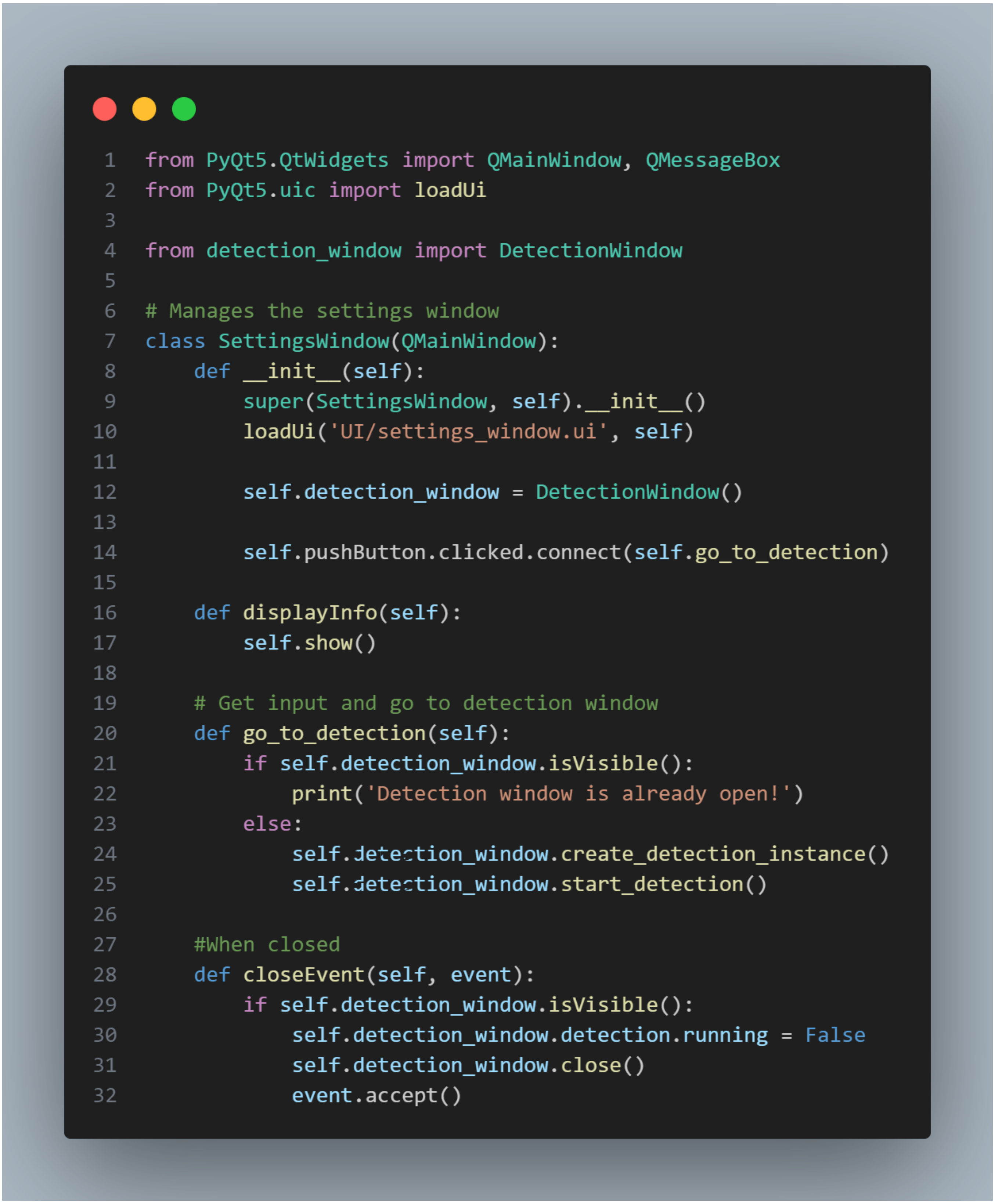
```python
from PyQt5.QtWidgets import QMainWindow
from PyQt5.uic import loadUi
from PyQt5.QtCore import QThread, Qt, pyqtSignal, pyqtSlot
from PyQt5.QtGui import QImage, QPixmap
import cv2
import numpy as np
import time

# Handles the opencv detection algorithm, saves detected frames and sends alert to the server-side application
class Detection(QThread):

    def __init__(self):
        super(Detection, self).__init__()

    changePixmap = pyqtSignal(QImage)

    # Runs the detection model, evaluates detections and draws boxes around detected objects
    def run(self):

        font = cv2.FONT_HERSHEY_PLAIN
        starting_time = time.time()

        self.running = True
        gun_cascade=cv2.CascadeClassifier('guns1.xml')
        # Starts camera
        cap = cv2.VideoCapture(0)

        # Detection while loop
        while self.running:
            ret, frame = cap.read()
            if ret:
                height, width, channels = frame.shape
                gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
                gun = gun_cascade.detectMultiScale(gray,1.3,3)
                for(x,y,w,h) in gun:
                    rect = cv2.rectangle(frame,(x,y),(x+w,y+h),(255,0,0),2)
                    cv2.putText(rect, 'Gun', (x, y-10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255,0,0), 2)
                    roi_gray=gray[y:y+h,x:x+w]
                    roi_color=frame[y:y+h,x:x+w]
                    elapsed_time = time.time() - starting_time

                    #Save detected frame every 10 seconds
                    if elapsed_time >= 10:
                        starting_time = time.time()
                        self.save_detection(frame)

                # Showing final result
                rgbImage = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
                bytesPerLine = channels * width
                convertToQtFormat = QImage(rgbImage.data, width, height, bytesPerLine, QImage.Format_RGB888)
                p = convertToQtFormat.scaled(854, 640, Qt.KeepAspectRatio)
                self.changePixmap.emit(p)

    # Saves detected frame as a .jpg within the saved_alert folder
    def save_detection(self, frame):
        cv2.imwrite("saved_frame/frame.jpg", frame)
        print('Frame Saved')
```

3.) DETECTION WINDOW MODULE



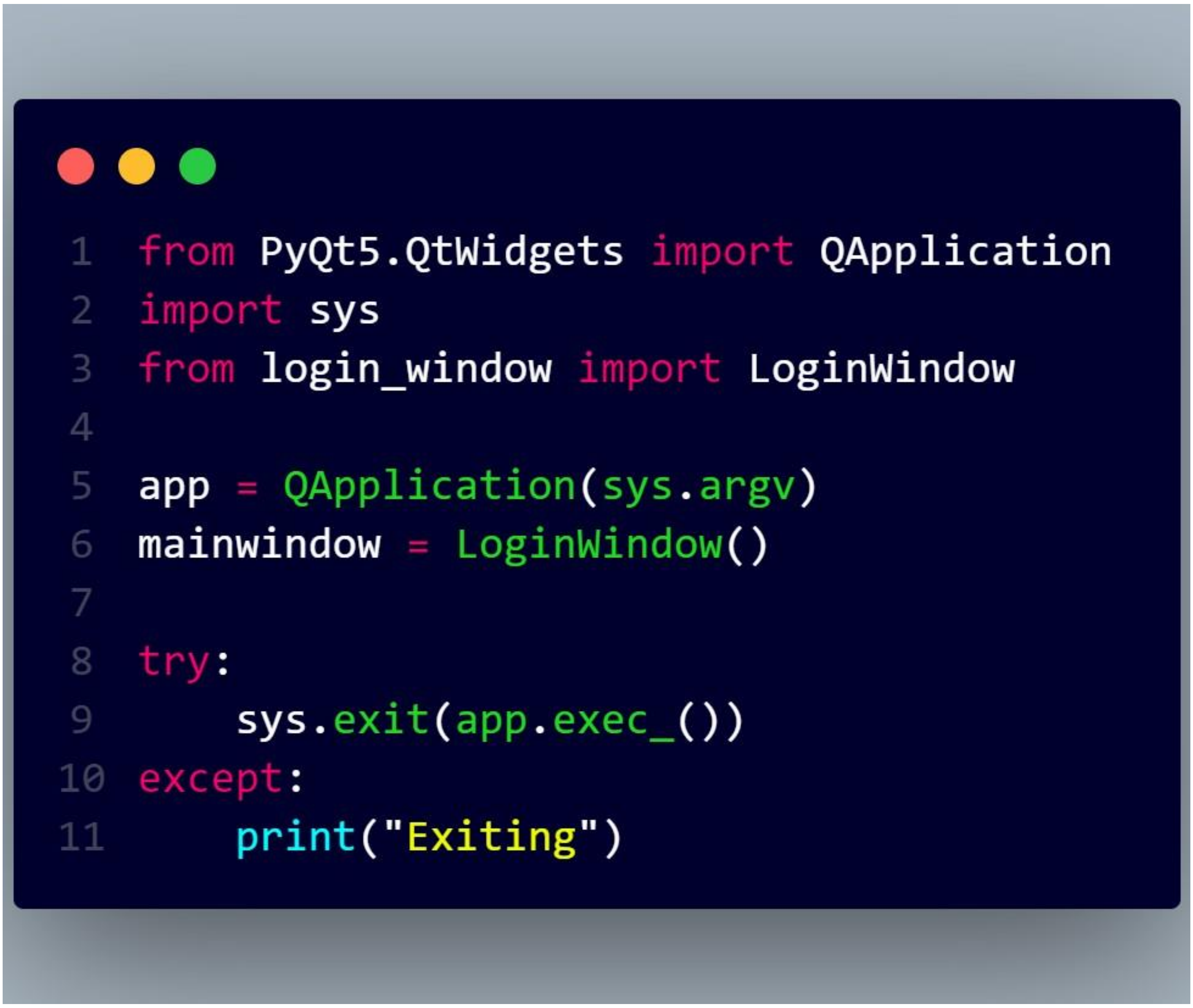
```python
from PyQt5.QtWidgets import QMainWindow
from PyQt5.uic import loadUi
from PyQt5.QtCore import QThread, Qt, pyqtSignal, pyqtSlot
from PyQt5.QtGui import QImage, QPixmap
from detection import Detection

# Manages detection window, starts and stops detection thread
class DetectionWindow(QMainWindow):
    def __init__(self):
        super(DetectionWindow, self).__init__()
        loadUi('UI/detection_window.ui', self)

        self.stop_detection_button.clicked.connect(self.close)

    # Created detection instance
    def create_detection_instance(self):
        self.detection = Detection()

    # Assigns detection output to the label in order to display detection output
    @pyqtSlot(QImage)
    def setImage(self, image):
        self.label_detection.setPixmap(QPixmap.fromImage(image))

    # Starts detection
    def start_detection(self):
        self.detection.changePixmap.connect(self.setImage)
        self.detection.start()
        self.show()

    # When closed
    def closeEvent(self, event):
        self.detection.running = False
        event.accept()
```

4.) SETTING WINDOW MODULE



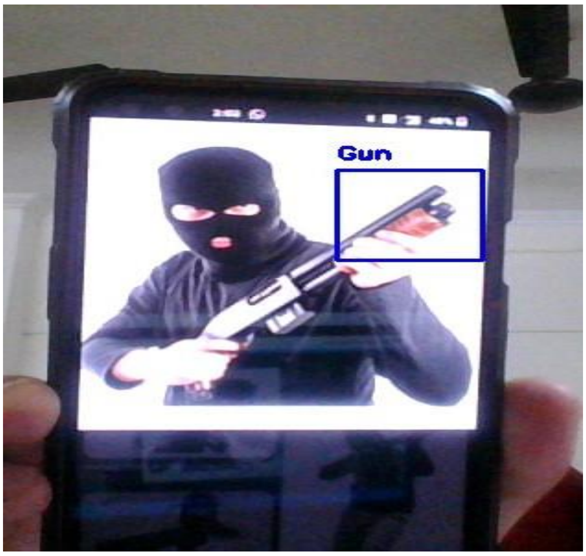
```python
from PyQt5.QtWidgets import QMainWindow, QMessageBox
from PyQt5.uic import loadUi

from detection_window import DetectionWindow

# Manages the settings window
class SettingsWindow(QMainWindow):
    def __init__(self):
        super(SettingsWindow, self).__init__()
        loadUi('UI/settings_window.ui', self)

        self.detection_window = DetectionWindow()

        self.pushButton.clicked.connect(self.go_to_detection)

    def displayInfo(self):
        self.show()

    # Get input and go to detection window
    def go_to_detection(self):
        if self.detection_window.isVisible():
            print('Detection window is already open!')
        else:
            self.detection_window.create_detection_instance()
            self.detection_window.start_detection()

    #When closed
    def closeEvent(self, event):
        if self.detection_window.isVisible():
            self.detection_window.detection.running = False
            self.detection_window.close()
            event.accept()
```

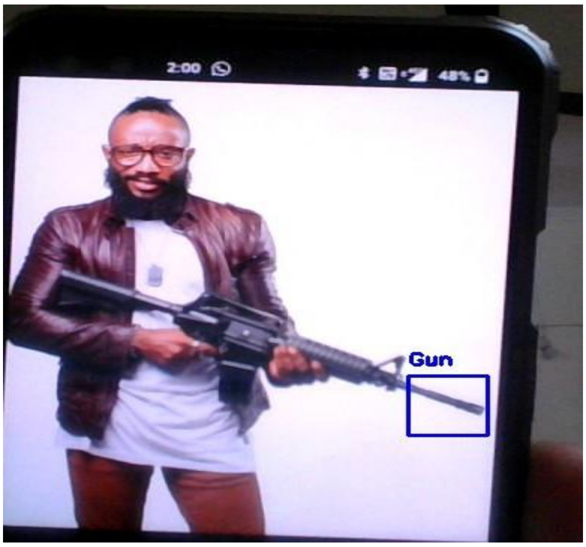
5.) MAIN MODULE

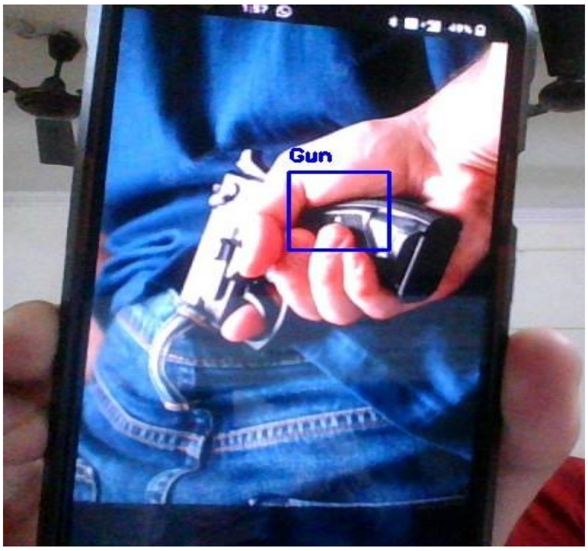


```python
from PyQt5.QtWidgets import QApplication
import sys
from login_window import LoginWindow

app = QApplication(sys.argv)
mainwindow = LoginWindow()

try:
    sys.exit(app.exec_())
except:
    print("Exiting")
```

SCREENSHOTS AND RESULTS







CONCLUSION AND FUTURE ENHANCEMENTS

In conclusion, weapon detection systems have the potential to enhance public safety and prevent violent incidents in public places. These systems use advanced sensor technologies and machine learning algorithms to detect potential threats and alert security personnel or law enforcement to respond promptly. However, the implementation of weapon detection systems requires careful consideration of the associated costs, challenges, and ethical implications, including privacy concerns and potential biases in the system's algorithms.

Future enhancements for weapon detection systems could include improvements in sensor technologies and machine learning algorithms to increase accuracy and reduce false positives. Additionally, the integration of other security systems such as facial recognition or object tracking could enhance the system's performance and provide a more comprehensive security solution. Another area of future research could be the use of blockchain technology to securely store and manage the data generated by the weapon detection system, ensuring privacy and accountability.

Overall, the development and implementation of weapon detection systems require a multi-disciplinary approach involving experts in sensor technologies, machine learning, security, and ethics. The continued research and development of these systems could provide valuable tools for enhancing public safety and preventing violent incidents in public places.

REFERENCES

- **www.opencv.org**
- **www.geeksforgeeks.com**
- **www.javatpoint.com**
- **www.researchgate.net**
- **www.kaggle.com**
- **www.tutorialspoint.com**
- **www.stackoverflow.com**
- **www.youtube.com**

GITHUB LINKS

ABHISHEK SINGH : https://github.com/AbhishekSingh1247/Weapon-Detection-System

VISHVESH BHARDWAJ : https://github.com/Vishvesh-Bhardwaj/Weapon-Detection-System

RAGHAV KAPOOR : https://github.com/Kraghav2002/Weapon-Detection-System