**Project Manual: Car Theft Alert System**

**1. Project Overview**

The *Car Theft Alert System* is a smart security system designed to detect unauthorized activity near a vehicle and immediately notify the owner with visual evidence. It combines motion detection, distance measurement, image capture, and SMS notification to provide a robust solution for vehicle safety.

**2. System Components**

**Hardware Requirements**

1. **Raspberry Pi** (any version with GPIO and camera support)
2. **STM32 Microcontroller**
3. **LD2410 Radar Sensor**
4. **Raspberry Pi Camera Module**
5. **4x4 Matrix Keypad**
6. **Push Buttons**: Start, Stop, Arm
7. **Power Supply**: Raspberry Pi (5V, 3A) and STM32
8. **Connecting Wires and Breadboard/PCB**

**Software Requirements**

1. **Raspberry Pi OS** (Lite or Full Version)
2. **Python 3** (for Raspberry Pi scripts)
3. **C/C++ Compiler** (for STM32 firmware)
4. **Dropbox API**
5. **Twilio API**
6. **GPIO and Camera Libraries** (e.g., RPi.GPIO, Picamera)

**3. System Setup**

**Hardware Setup**

1. **Raspberry Pi Setup**:
   * Connect the **Camera Module** to the Raspberry Pi's camera interface.
   * Wire the **4x4 Keypad** to the GPIO pins for passcode input.
   * Attach **Start**, **Stop**, and **Arm** buttons to GPIO pins.
   * Connect a GPIO pin to receive input from the STM32.
2. **STM32 and Radar Sensor Setup**:
   * Connect the LD2410 radar sensor to the STM32 UART interface.
   * Configure one GPIO pin on STM32 to output a high signal when the distance is less than 30 cm.
   * Wire the STM32 GPIO pin to a Raspberry Pi input GPIO.
3. **Power Supply**:
   * Ensure all components are powered appropriately. Use separate power sources for Raspberry Pi and STM32 if required.

**Software Installation**

1. **Raspberry Pi OS Installation**:
   * Flash Raspberry Pi OS onto an SD card using tools like BalenaEtcher.
   * Boot the Raspberry Pi and complete the initial setup.
2. **Install Required Libraries**:
3. sudo apt update && sudo apt upgrade
4. sudo apt install python3-pip

pip3 install dropbox twilio RPi.GPIO picamera

1. **STM32 Firmware Development**:
   * Write a C program to process radar sensor data and send GPIO signals when motion is detected within 30 cm.
   * Use STM32CubeIDE or similar tools to compile and flash the firmware onto the STM32 microcontroller.

**4. System Operation**

**Step 1: Arming the System**

1. Press the **Start** button on the Raspberry Pi.
2. Enter the passcode using the 4x4 keypad.
3. If the passcode is correct, the system activates, and the Raspberry Pi signals the STM32 to begin monitoring.

**Step 2: Motion Detection**

1. The LD2410 radar sensor continuously measures distances.
2. If an object is detected within 30 cm, the STM32 sends a high GPIO signal to the Raspberry Pi.

**Step 3: Image Capture**

1. Upon receiving the signal, the Raspberry Pi activates the camera to capture an image.
2. The captured image is saved locally and uploaded to Dropbox using its API.

**Step 4: User Notification**

1. A sharable link to the uploaded image is generated using the Dropbox API.
2. The link is sent to the user via SMS using the Twilio API.

**Step 5: Stopping the System**

* Press the **Stop** button to deactivate the system.

**5. Implementation**

**Raspberry Pi Code**

Below is an example of the Python script running on the Raspberry Pi:

|  |
| --- |
| import RPi.GPIO as GPIO  from picamera import PiCamera  import time  import dropbox  from twilio.rest import Client  # GPIO Pin Configuration  START\_BUTTON = 17  STOP\_BUTTON = 27  ARM\_BUTTON = 22  INPUT\_SIGNAL = 18  # Setup GPIO  GPIO.setmode(GPIO.BCM)  GPIO.setup(START\_BUTTON, GPIO.IN, pull\_up\_down=GPIO.PUD\_UP)  GPIO.setup(STOP\_BUTTON, GPIO.IN, pull\_up\_down=GPIO.PUD\_UP)  GPIO.setup(ARM\_BUTTON, GPIO.IN, pull\_up\_down=GPIO.PUD\_UP)  GPIO.setup(INPUT\_SIGNAL, GPIO.IN)  # Camera Setup  camera = PiCamera()  # Dropbox and Twilio Configuration  dropbox\_access\_token = 'your\_dropbox\_access\_token'  twilio\_account\_sid = 'your\_twilio\_account\_sid'  twilio\_auth\_token = 'your\_twilio\_auth\_token'  receiver\_phone\_number = 'recipient\_phone\_number'  sender\_phone\_number = 'twilio\_phone\_number'  client = Client(twilio\_account\_sid, twilio\_auth\_token)  db\_client = dropbox.Dropbox(dropbox\_access\_token)  # Functions  def capture\_and\_notify():  timestamp = time.strftime("%Y%m%d-%H%M%S")  image\_path = f"/home/pi/car\_alert\_{timestamp}.jpg"    # Capture Image  camera.capture(image\_path)  # Upload to Dropbox  with open(image\_path, "rb") as file:  response = db\_client.files\_upload(file.read(), f"/car\_alert\_{timestamp}.jpg")  shared\_link = db\_client.sharing\_create\_shared\_link\_with\_settings(response.path\_display)    # Send SMS  message = client.messages.create(  body=f"Car alert! See the image: {shared\_link.url}",  from\_=sender\_phone\_number,  to=receiver\_phone\_number  )  print("Notification sent")  # Main Loop  try:  while True:  if GPIO.input(START\_BUTTON) == GPIO.LOW:  print("System armed")  while GPIO.input(STOP\_BUTTON) == GPIO.HIGH:  if GPIO.input(INPUT\_SIGNAL) == GPIO.HIGH:  capture\_and\_notify()  time.sleep(1)  except KeyboardInterrupt:  print("System stopped")  finally:  GPIO.cleanup() |

**6. System Testing and Troubleshooting**

**Testing**

1. Verify that the camera captures images correctly when activated.
2. Check the STM32’s GPIO output when an object is detected within 30 cm.
3. Ensure the Raspberry Pi receives the GPIO signal from the STM32.
4. Test Dropbox integration by uploading an image and generating a sharable link.
5. Confirm SMS notifications are sent with the correct Dropbox link.

**Troubleshooting**

* **Issue**: Camera does not capture images.
  + **Solution**: Check the camera connection and ensure the camera is enabled in the Raspberry Pi configuration.
* **Issue**: No GPIO signal received from STM32.
  + **Solution**: Verify the wiring and ensure the STM32 firmware is correctly flashed.
* **Issue**: SMS notifications are not sent.
  + **Solution**: Verify Twilio credentials and internet connectivity on the Raspberry Pi.

**7. Future Enhancements**

1. **Cloud Storage Options**: Add support for other cloud storage providers like Google Drive or OneDrive.
2. **Mobile App Integration**: Develop a mobile app for real-time notifications and system control.
3. **Improved Sensors**: Upgrade to higher-precision sensors for better motion detection.

**8. Safety Precautions**

* Ensure the power supply matches the voltage requirements of the Raspberry Pi and STM32 to avoid damage.
* Handle electronic components with care to prevent static discharge damage.
* Regularly test the system to ensure reliability in real-world scenarios.

**9. Conclusion**

The Car Theft Alert System provides an innovative and effective solution for vehicle safety. By integrating hardware and software components, it ensures quick detection and notification, offering peace of mind to vehicle owners.