IOT-Based Intruder Detection and Alert System

Pranita Shewale
Department of Electrical and Computer Engineering
Stevens Institute of Technology
Hoboken, NJ, USA
pshewale1@stevens.edu

Bathini Shravan Kumar
Department of Electrical and Computer Engineering
Stevens Institute of Technology
Hoboken, NJ, USA
skumar19@stevens.edu

Nirmohi Samirbhai Patel
Department of Electrical and Computer Engineering
Stevens Institute of Technology
Hoboken, NJ, USA
npatel24@stevens.edu

Abstract*—This project develops an intruder detection and alert system leveraging NodeMCU development board with integrated ESP8266 Wi-Fi chip to enable wireless connectivity for security implementations. It combines a passive infrared pyroelectric motion sensor and piezo acoustic alarm module interfaced via exposed input and output General Purpose Input/Output (GPIO) pins. Detected intrusion breaches by the sensor trigger loud audible alerts from buzzer exceeding 85 decibels. Adjustable detection thresholds balance responsiveness and false alarms. Integrated flash memory also hosts a web interface reflecting real-time system state for remote visualization. The implemented prototype serves as introductory demonstration for exploring more sophisticated future iterations augmenting complementary detectors, advanced connectivity to monitoring platforms, expanded deterrence mechanisms, and embedded decision intelligence.

***Keywords*— ESP8266, NodeMCU firmware, Lua script**

I. INTRODUCTION

Connected intruder detection platforms are transforming techniques for securing homes and assets by bringing Internet of Things advancements to physical security systems. Wireless capabilities enhance flexibility while reducing installation and infrastructure complexity associated with traditional wired setups. This paper discusses using the compact and extremely cost-effective NodeMCU embedded development modules with integrated ESP8266 Wi-Fi System-on-Chip (SoC) for rapidly developing a functional intrusion detection prototype taking inputs from a passive infrared motion sensor to trigger an audible piezo buzzer alarm.

II. RELATED WORK

Modern wireless intruder detection leverages technologies spanning embedded systems, sensor modules, and Internet connectivity advancements. Khutso and Maduemezia [1] developed a prototype using Arduino and ESP8266 to detect motion with a PIR sensor and trigger email alerts by interfacing with an SMTP server. Myla et al. [2] implemented home intrusion detection combining Arduino with Xbee modules for wireless mesh networking capabilities. Tareq et al. [3] realized a security system using Arduino web server hosting to visualize sensor dashboard. The NodeMCU hardware provides integrated WiFi connectivity and development framework appropriate for streamlined implementations.

III. SYSTEM OVERVIEW

Figure 1 shows the component block diagram. The system activates an 85dB piezo buzzer upon sensing motion via the HC-SR501 Passive IR sensor module. The core runtime logic is executed on NodeMCU processing module featuring ESP8266 wireless capabilities. It continually samples the analog output voltage from the sensor, comparing against predetermined thresholds to ascertain breaches, appropriately toggling the digital GPIO alarm pin. A web interface hosted on built-in memory reflects current detection state and hyperparameters.

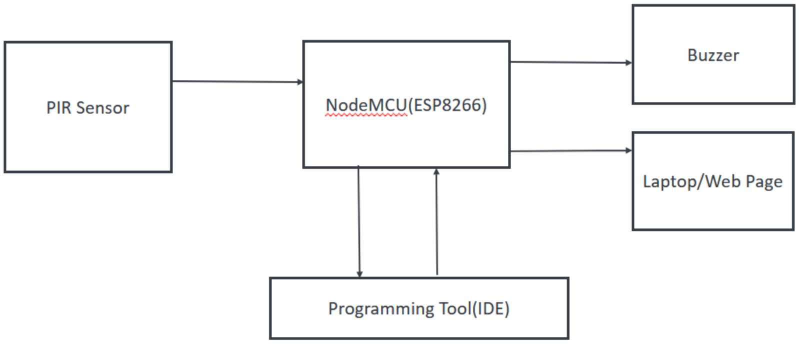


Figure 1 Block Diagram

IV. SYSTEM HARDWARE

NodeMCU Development Board: The NodeMCU open-source platform couples easy to use firmware with the ESP8266 WiFi system-on-chip, providing integrated microcontroller capability and wireless networking. It condenses antenna, memory, and GPIO interfaces into a compact low-cost module for IoT edge applications [4].

PIR Motion Sensor: The HC-SR501 utilizes a pyroelectric component that detects levels of infrared signatures emitted from objects. Changes indicate heat differentials associated with motion events. Analog voltage levels shift proportionally allowing interfacing to processors [5].

Piezo Passive Buzzer: The audio alarm unit contains a piezoelectric transducer that converts electrical pulses into audible tones. At 12V DC input, it produces 85+ decibel alerts suitable for alarming applications [6].

Electronic Connections: Figure 2 depicts system wiring. GPIO pins provide power and interfaces between devices. Pull up/down resistors improve reliability for voltage shifts indicating digital logic state transitions. The sensor analog output wire connects to an ADC input pin while the buzzer links to a designated digital GPIO pin for triggering alarms.

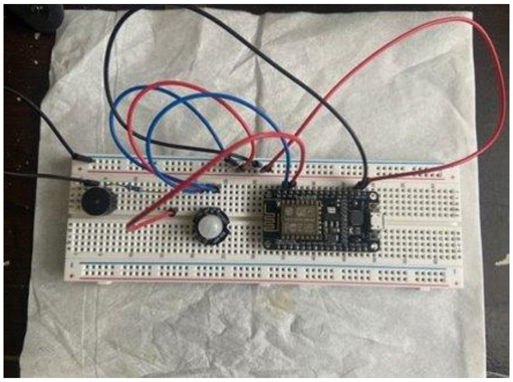


Figure 2 System Wiring

V. NODE MCU FEATURES

Node MCU is a development board which has ESP8266 Wi-Fi SOC (System on Chip), along with other peripherals. As shown in figure 3 which is Low cost, Simple, Smart, WI-FI enabled, open source, interactive, programmable, Arduino-like hardware and NodeJS style network API.

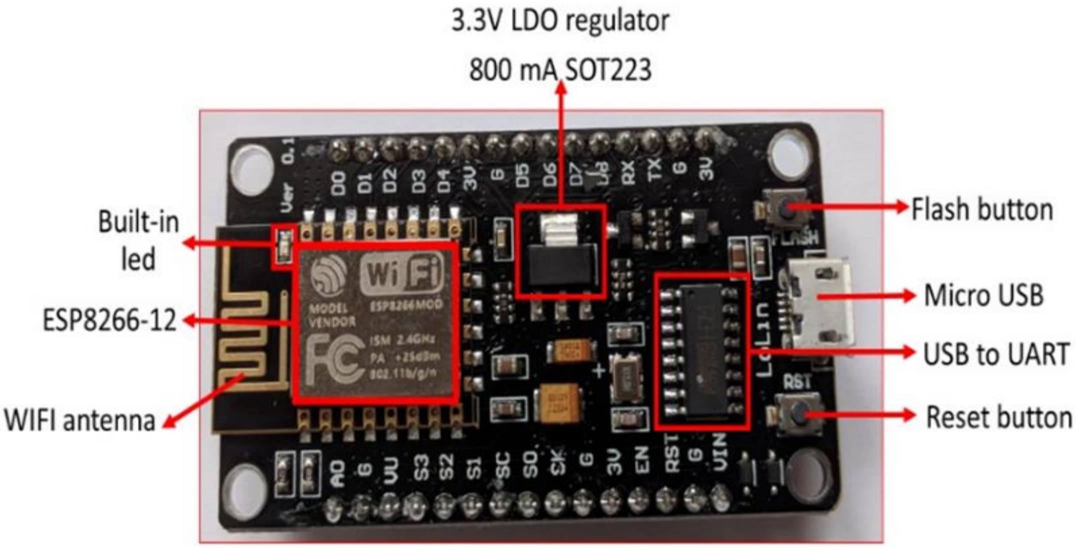


Figure 3 NodeMCU development board

The basic process to get started with NodeMCU consists of the following three steps:

- Build the firmware with the modules you need.
- Flash the firmware to the chip.
- Upload code to the device.

Preparation of development environment consists of following steps:
1. Setup of Toolchain
2. Getting ESP8266_RTOS_SDK from official site.
3. Installation and configuration of Eclipse /IDE
4. Writing code for applications, compiling, uploading, monitoring, or debugging application.

There are various approaches where we can develop application on NodeMCU one is using AT commands while other is NodeMCU Firmware.

A. ESP-AT

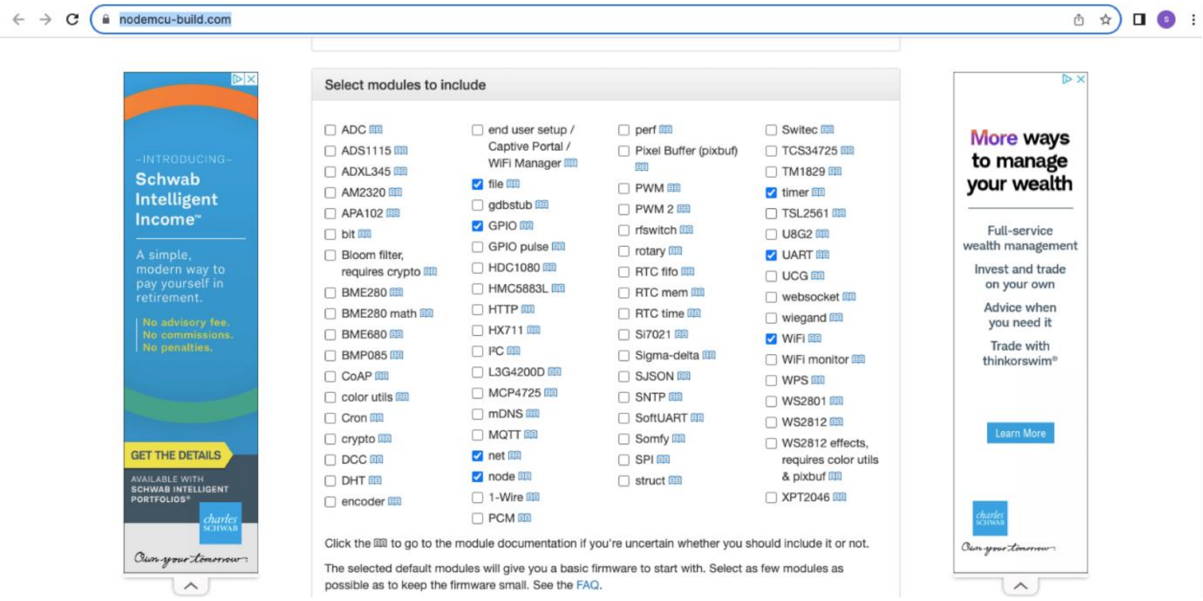
ESP-AT is a project based on ESP8266_RTOS_SDK, Node MCU acts as slave and Microcontroller acts as host Controller sends AT and ESP chip respond. ESP-AT commands Wi-Fi, TCP, Bluetooth, HTTP and many more. Download AT Firmware, Flash AT Firmware into Your Device (ESP Flash Download Tool and Check Whether AT Works on serial monitor.

B. NodeMCU Firmware

NodeMCU is an open-source Lua based firmware for the ESP8266. NodeMCU is implemented in C and is layered on the Espressif NON-OS SDK. Asynchronous event-driven programming model. It is Lua based firmware, which is layered in top of NodeMCU firmware which is implemented in C language on ESP8266 hardware.

VI. BUILDING LUA BASED FIRMWARE FOR NODEMCU

Before we can use the hardware board, we need to compile the firmware and flash it to the flash memory. A web service "NodeMCU custom builds" provides convenient cloud-based compilation and interactive configuration of modules of the firmware. Figure shows the use of an official website: https://nodemcu-build.com/, where we can include the modules required for our application to build our own firmware based on our work. It has various modules to include like the ADC, GPIO, PCM, WM, Timer, UART, WiFi etc. After selecting the modules, a custom generated firmware in binary file format is received for flashing on to the device.



Select modules to include

- □ ADC
- □ ADS1115
- □ ADXL345
- □ AM2320
- □ APA102
- □ bit
- □ Bloom filter, requires crypto
- □ BME280
- □ BME280 math
- □ BME680
- □ BMP085
- □ CoAP
- □ color utils
- □ Cron
- □ crypto
- □ DCC
- □ DHT
- □ encoder
- □ end user setup / Captive Portal / WiFi Manager
- ☑ file
- □ gdbstub
- ☑ GPIO
- □ GPIO pulse
- □ HDC1080
- □ HMC5883L
- □ HTTP
- □ HX711
- □ I²C
- □ L3G4200D
- □ MCP4725
- □ mDNS
- □ MQTT
- ☑ net
- ☑ node
- □ 1-Wire
- □ PCM
- □ perf
- □ Pixel Buffer (pixbuf)
- □ PWM
- □ PWM 2
- □ rfswitch
- □ rotary
- □ RTC fifo
- □ RTC mem
- □ RTC time
- □ Si7021
- □ Sigma-delta
- □ SJSON
- □ SNTP
- □ SoftUART
- □ Somfy
- □ SPI
- □ struct
- □ Switec
- □ TCS34725
- □ TM1829
- ☑ timer
- □ TSL2561
- □ U8G2
- ☑ UART
- □ UCG
- □ websocket
- □ wiegand
- ☑ WiFi
- □ WiFi monitor
- □ WPS
- □ WS2801
- □ WS2812
- □ WS2812 effects, requires color utils & pixbuf
- □ XPT2046

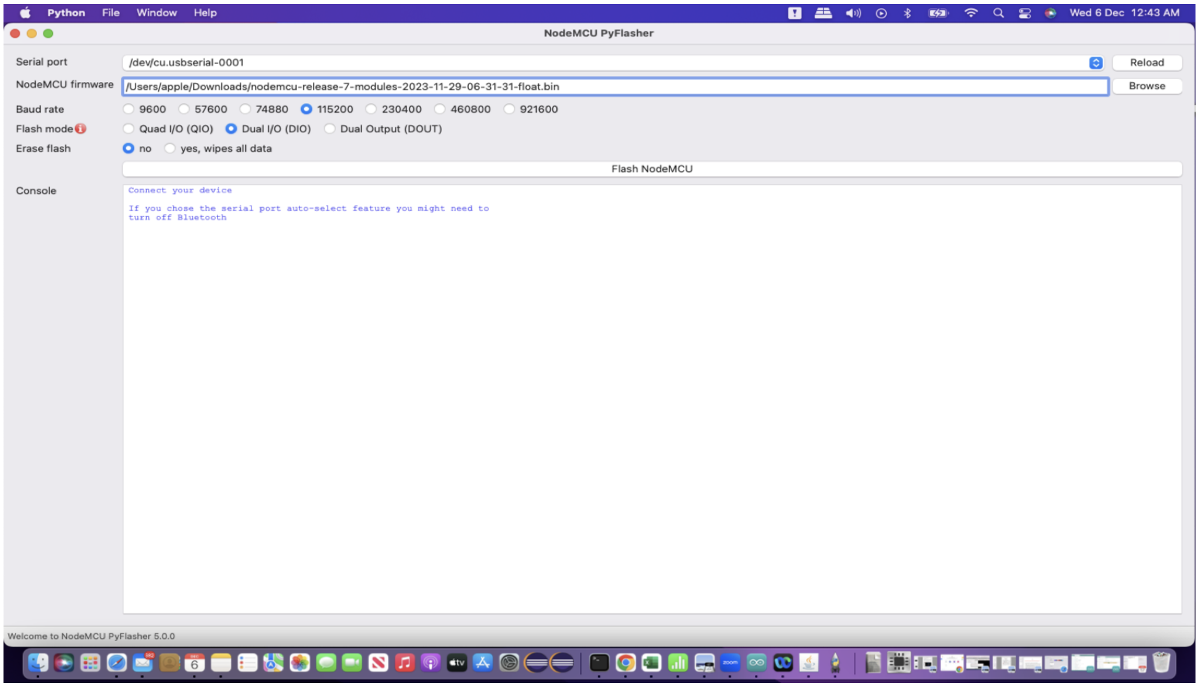
Click the 📖 to go to the module documentation if you're uncertain whether you should include it or not.

The selected default modules will give you a basic firmware to start with. Select as few modules as possible as to keep the firmware small. See the FAQ.

Figure 4 showing the use of Custom build tool for building the firmware.

VII. FLASHING THE FIRMWARE FOR NODEMCU

Once the binary file is received, then flashing is done using the NodeMCU PyFlasher software as shown in the below figure:



Serial port: /dev/cu.usbserial-0001

NodeMCU firmware: /Users/apple/Downloads/nodemcu-release-7-modules-2023-11-29-06-31-31-float.bin

Baud rate: ○ 9600 ○ 57600 ○ 74880 ● 115200 ○ 230400 ○ 460800 ○ 921600

Flash mode: ○ Quad I/O (QIO) ● Dual I/O (DIO) ○ Dual Output (DOUT)

Erase flash: ● no ○ yes, wipes all data

Console:

```
Connect your device

If you chose the serial port auto-select feature you might need to
turn off Bluetooth
```

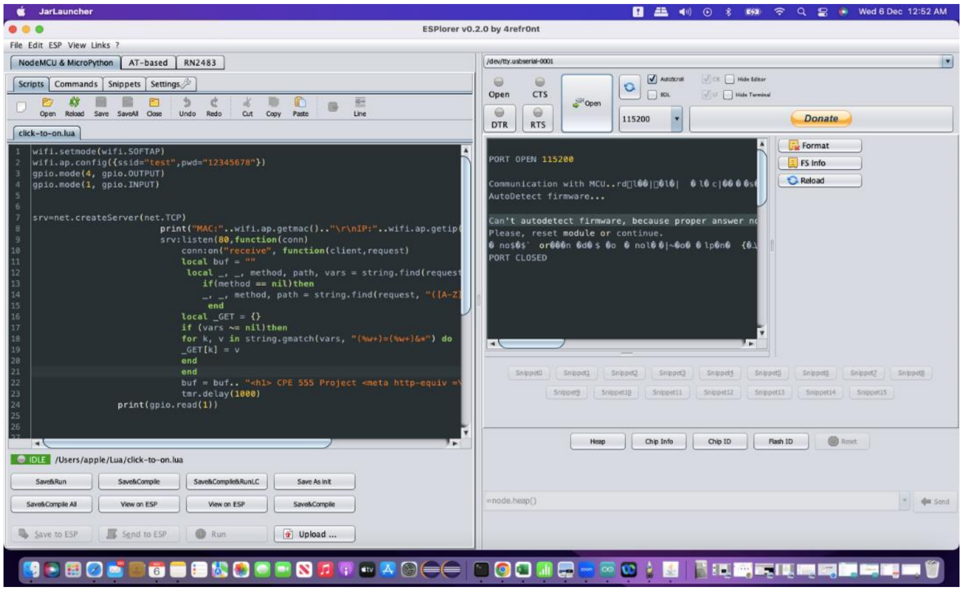
Welcome to NodeMCU PyFlasher 5.0.0

Figure 5 shows the use of PyFlasher tool used for flashing the firmware.

VIII. ESPLORER IDE FOR PROGRAMMING

ESPlorer is an integrated development environment (IDE) for the ESP8266 and ESP32 microcontroller platforms. It allows you to program and interact with these devices using the Lua programming language.

In this project, the board was programmed using this Esplorer IDE and code was written in Lua programming language. Esplorer is an IDE with an advantage of highlighting the LUA and python code, code editor, code auto completion and smart sending data to ESP8266. The interface of Esplorer looks like:



```
wifi.setmode(wifi.SOFTAP)
wifi.ap.config({ssid="test",pwd="12345678"})
gpio.mode(4, gpio.OUTPUT)
gpio.mode(1, gpio.INPUT)


srv=net.createServer(net.TCP)
                        print("MAC:"..wifi.ap.getmac().."\r\nIP:"..wifi.ap.getip(
                        srv:listen(80,function(conn)
                            conn:on("receive", function(client,request)
                            local buf = ""
                             local _, _, method, path, vars = string.find(request
                                if(method == nil)then
                                _, _, method, path = string.find(request, "([A-Z]
                                 end
                            local _GET = {}
                            if (vars ~= nil)then
                            for k, v in string.gmatch(vars, "(%w+)=(%w+)&*") do
                            _GET[k] = v
                            end
                            end
                            buf = buf.. "<h1> CPE 555 Project <meta http-equiv =\
                            tmr.delay(1000)
                print(gpio.read(1))
```

IDLE /Users/apple/Lua/click-to-on.lua

/dev/tty.usbserial-0001

☑ AutoScroll ☐ EOL ☑ CR ☑ LF ☐ Hide Editor ☐ Hide Terminal

115200

```
PORT OPEN 115200

Communication with MCU..rd□l��|□�l�|  � l� c|�� � �s�
AutoDetect firmware...

Can't autodetect firmware, because proper answer no
Please, reset module or continue.
� no$�$`  or���n �d� $ �o  � nol� �|~�o� � lp�n�  {�
PORT CLOSED
```

=node.heap()

Figure 6 shows the ESPlorer IDE used for programming.

A. LUA Programming Language:

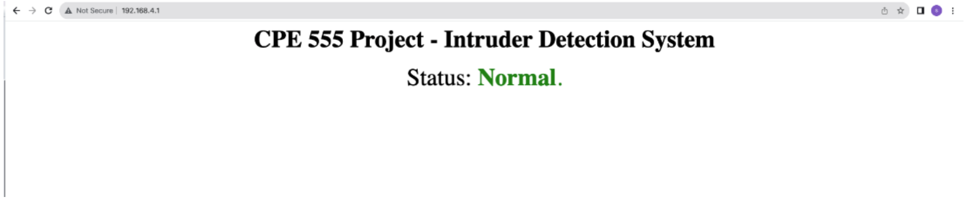
Lua is a powerful, efficient, and lightweight scripting language that is designed to be easily embedded into other programming languages, such as C and C++. It has dynamic typing, and simple syntax. It has been used in many industrial applications with focuses on embedded systems and games.

B. Webserver:

NodeMCU, with its built-in networking capabilities, is an ideal choice for creating web servers in IoT projects. Lua, as a scripting language, simplifies the development process, making it accessible to a broader audience. To set up network connectivity, the script configures the NodeMCU to connect to a Wi-Fi network. This step is crucial for enabling the device to be accessible over the network. The core functionality involves creating an HTTP server that listens on port 80. Upon receiving a request, the server responds with a simple message.

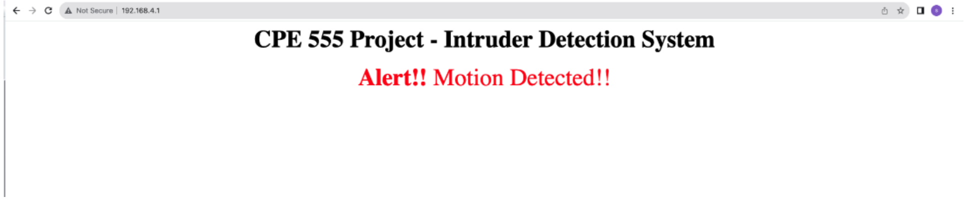
IX. RESULTS

The program is written in such a way that after connecting to the Wi-Fi network it displays the IP address of the Webserver. Whenever a motion is detected, the PIR sensor pulls the GPIO pin to High. The data on the Webpage is refreshed every 1 second. During this event, the controller checks for the status of the GPIO pin and displays the content on the Web page accordingly. The following pictures show the display page first one- when there is no motion detected, the second whenever there is a motion detected.



CPE 555 Project - Intruder Detection System

Status: **Normal**.



CPE 555 Project - Intruder Detection System

Alert!! Motion Detected!!

Figure 7 shows the result before and after intrusion

X. FUTURE SCOPE

Further improvements can be made to this project like expanding an intruder detection system to mobile application development to show the status and alerts, SMS and call alerts, as well as multi-sensor integration viz. this approach can be adopted in combination with other sensors like CCTVs to record the video whenever there is an alert. These additions can significantly enhance the capabilities of an intruder detection system.

X1. REFERENCES

[1] https://docs.espressif.com/

[2] https://nodemcu.readthedocs.io/en/release/

[3] Saleh Ahmad Khan, Ishtiaq Ahmad, Md. Sazzad Ahmed, Abu Shufian “A Smart Intruder Alert System Based on Microprocessor and Motion Detection,” 2019 International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST).