NOTRE DAME UNIVERSITY
BANGLADESH

Project Report Book

Course Title: Microprocessor & Assembly Language Programming Lab

Course Code: CSE 3106

Project Title: Invisible Security System

Batch: CSE (19+20)

Submitted by:

Istiak Alam (CSE 20)	**ID: 0692230005101005**
Sazzad Jelani	**ID: 0692220005101003**
Nafisa Tabassum	**ID: 0692220005101008**
Sadia Islam Mim	**ID: 0692220005101010**

Submitted to:

Dr. Shaheena Sultana
Professor & Chair, NDUB

Table of Contents

Table of Figures

Chapter 1: Introduction

The **Invisible Security System** project is designed to deliver a user-friendly solution for securing spaces by detecting unauthorized entry. By using a laser and an LDR, this system immediately detects any disruption to the beam, triggering an alert. With affordable components and straightforward assembly, this project is accessible to general users and provides a practical introduction to integrating IoT into security applications.

1.1 Motivation & Purpose of the Project

In recent years, there has been an increasing need for accessible, affordable, and effective security solutions in homes, businesses, and other facilities. Traditional security systems often come with high costs, complex installations, and ongoing service fees. This project was motivated by the potential to create a DIY security solution using easily accessible electronic components, including an Arduino, a laser module, an LDR, and a NodeMCU. By creating a system that is "invisible" yet reliable, this project offers a more discrete security measure that can be deployed in various settings.

The primary purpose of this project is to develop a security system that is both affordable and easily configurable for small-scale applications. It serves as a prototype for motion detection systems that can be deployed in homes, offices, or small establishments where an affordable, lightweight solution is desirable. The project also aims to demonstrate the integration of basic hardware with remote web-based control, thereby increasing its adaptability and convenience. This Invisible Security System provides users with a responsive, real-time security measure and offers practical learning experience in IoT, hardware-software interfacing, and automation technology.

1.2 Objective

The primary objective of the **Invisible Security System** project is to create a reliable, efficient, and low-cost security solution for securing spaces using a laser and Light Dependent Resistor (LDR). This project aims to detect unauthorized access by breaking a laser beam positioned across an entryway or other protected area. When the laser beam is interrupted, an alarm is triggered, providing immediate notification of a possible intrusion. The project also includes integrating a NodeMCU for remote control, allowing the laser to be toggled on or off from a web-based interface, enhancing usability and control.

1.3 Related Work

This project was highly inspired by the company Optex Pinnacle. They come up with innovative LiDAR based products which specializes in laser-based products and devices along with their installation system depending on the type of security. For more information visit Optex pinnacle.

1.4 Project Scope

Invisible Security System is ideal for -

- Banks, ATMs.
- Data centers.
- Museums.
- Outdoor restricted Areas. Such as Area51, Tunnels
- Security Vault.

1.5 Organization of the Report

This section includes how we have curated and organized our project report. It will contain a summary of each chapter of this report.

Chapter 1: Introduction

Introduces the Invisible Security System project, explaining its purpose, scope, objectives, and motivations. It discusses the project's background in security and its application. It also covers the work conceptual to our project.

Chapter 2: Hardware

Describes the hardware with details on each component's role (with citation). It includes the circuit diagram and setup to integrate Arduino, NodeMCU and the security sensors.

Chapter 3: Software

Covers the software components, including Arduino IDE and Tinkercad for circuit simulation. Details of programming languages, codes with their internal analysis including setup and loop functions, web commands, and LDR monitoring, the code structure with its flowchart.

Chapter 4: Project Output

Presents photos of the project from different angles and functioning scenario. This basically demonstrates the overall project with its agenda.

Chapter 5: Conclusion

Discusses issues faced, such as sensor sensitivity, connectivity, power stability, and environmental factors and how we recovered those. What more features can be added in the future and reference for the project.

Chapter 2 : Hardware

In the Hardware Section, we have added some general information about the components we used in this project and a detailed circuit diagram. This will give an overview of the building blocks of this project. The **Invisible Security System** is designed to secure spaces by creating an invisible laser barrier that activates an alert upon disruption. Key components like the laser module, LDR, and Arduino Uno combine to detect intrusions and trigger alerts. An optional feature with the NodeMCU ESP8266 allows users to remotely enable or disable the laser through a web interface, enhancing control over the system. This setup ensures robust security in a cost-effective and flexible solution.

2.1 Required Components

Here is the list of Components with their prices:

Sl. No.	Components	Amount (pieces)	Cost
1.	Arduino UNO R3	1	1050/-
2.	NodeMCU ESP8266	1	420/-
3.	Laser Module	2	200/-
4.	Passive Buzzer	1	15/-
5.	LCD Display (16x2)	1	230/-
6.	Breadboard 830point	1	150/-
7.	Breadboard 400point	1	85/-
8.	Mirror	6	50/-
9.	Light Depending Resistor (10-12 mm)	2	60/-
10.	Resistors (1kΩ, 10kΩ)	2	10/-
11.	Jumper wire (Female to Male)	10	100/-
12.	Connecting Wires	As required	150/-
13.	Power Bank	1	Not declared

Table 1 : Component List

2.2 Components Description

1) **Arduino Uno R3:** A versatile microcontroller that acts as the project's main controller. It reads signals from the LDR and activates the buzzer when the laser beam is disrupted. Coordinates system actions by continuously monitoring the LDR and responding to changes, ensuring quick and reliable alert triggering.

Figure 2.1: Arduino Uno R3

2) **Laser Module:** Emits a concentrated laser beam to create an invisible barrier. This will Forms the "invisible line" monitored by the LDR. When this beam is interrupted, it signals a breach.

Figure 2.2: Laser Module

3) **NodeMCU ESP8266:** The NodeMCU ESP8266 is a Wi-Fi-enabled microcontroller that enhances this security system with built-in Wi-Fi functionality, it acts as a web server, creating a user interface that allows for secure, remote control of the laser module. The NodeMCU connects with the Arduino via serial communication. Commands sent through the web interface can trigger the laser to turn on or off.

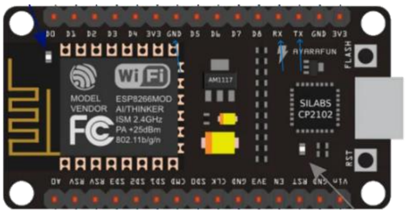

Figure 2.3: ESP8266

4) **Passive Buzzer:** A small device capable of producing a tone or sound when powered. Serves as the alarm for the security system. It sounds an alert when the laser beam is interrupted.

Figure 2.4: Passive Buzzer

5) **LCD Display (16x2):** A display module that shows messages and statuses. Provides real-time feedback, showing "Alarm Enabled" or "Alarm Disabled" based on system status, which helps users easily check if the alarm is active.

Figure 2.5: LCD Display

6) **LDR** (Light Dependent Resistor) **:** A sensor that changes resistance based on light exposure. This will detect light from the laser module. A drop in detected light (due to obstruction) triggers the Arduino to sound an alarm, marking an intrusion.

Figure 2.6: LDR

7) **Mirror:** The mirror will reflect the Leaser and Forms the "invisible line" from Leaser module to the LDR (Light Dependent Resistor)

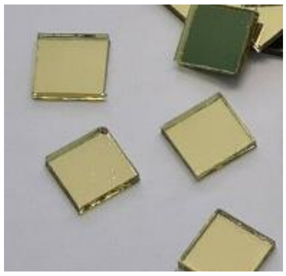

Figure 2.7: Mirror

8) **Resistor (1kΩ, 10kΩ) :** Passive electronic components that limit current. This will Protects components like the LDR by controlling current flow, ensuring stable and safe operation.

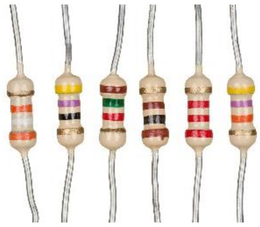

Figure 2.8: resistor

9) **Breadboard:** Holds the laser module, LDR, resistors, and jumper wires in place, enabling the entire circuit to be assembled neatly and tested quickly. This setup is ideal for adjusting the circuit design as required and troubleshooting connections in the security system.

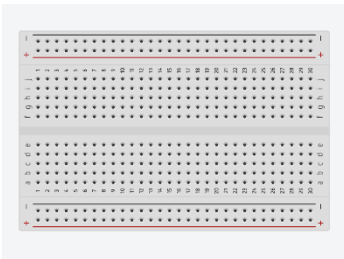

Figure 2.9: Breadboard

10) **Jumper Wire:** This will Connect the laser module, LDR, and other components to the Arduino, enabling signal transfer. They also link the NodeMCU to the Arduino, allowing for remote control of the laser module.

Figure 2.10: Jumper Wire

11) **Power Bank:** Powers the Arduino, NodeMCU, and other components of the security system.

Figure 2.11: Power Bank

2.3 Circuit Diagram

The circuit diagram illustrates the integration of the Arduino UNO, NodeMCU, laser module, LDR (light-dependent resistor), and supporting components. The Arduino controls the laser and processes input from the LDR to detect interruptions, while the NodeMCU adds remote control capability through a web interface. The power bank supplies portable energy, and jumper wires link components via the breadboard. This setup allows the Arduino to monitor and react to security breaches, with the NodeMCU facilitating remote activation and deactivation of the laser. The Circuit diagram is shown in figure 2.12.

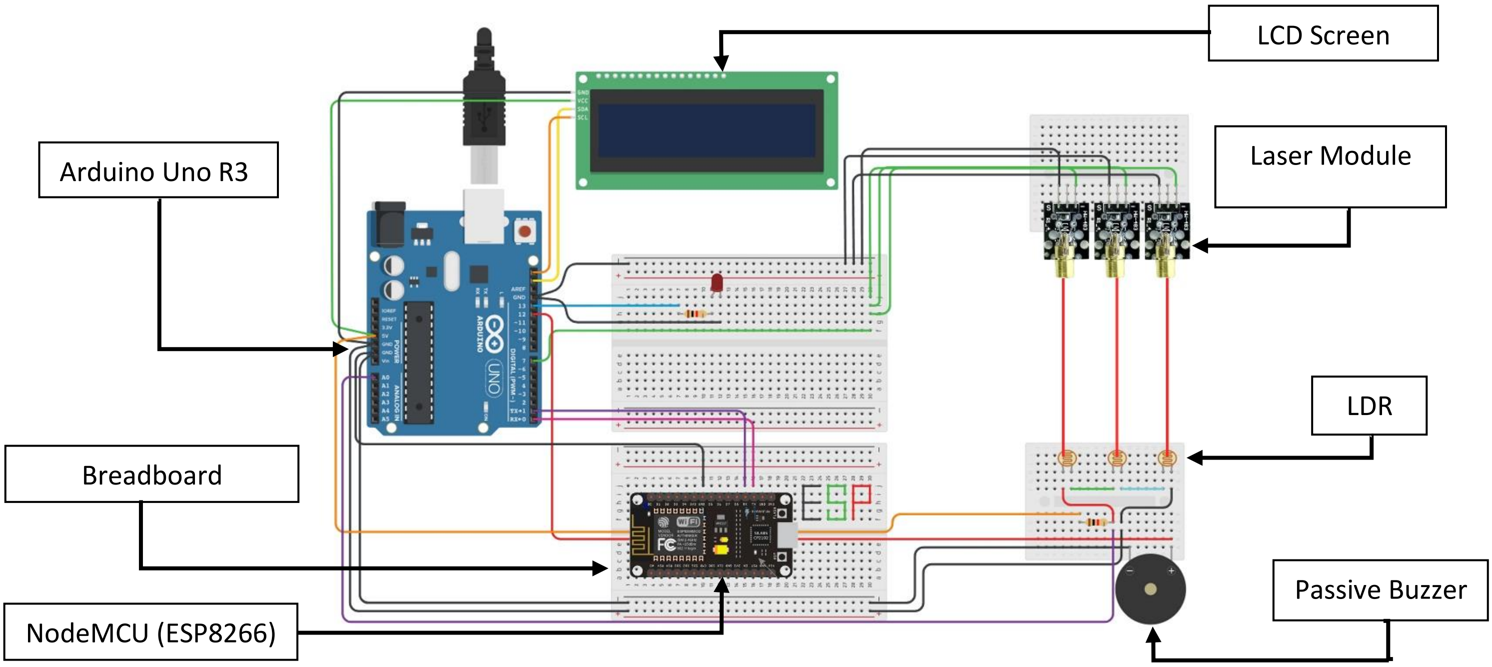

Figure 2.12 : Circuit Diagram

2.4 Circuit Overview and Connections

The project setup includes an Arduino Uno as the primary controller, an LDR to detect laser interruption, a laser module controlled by the Arduino, and a NodeMCU for web-based control. Each part is carefully configured to create a responsive security system.

- **Laser Module**: The laser module is connected to **digital pin 9** of the Arduino. This allows Arduino to control the laser's state.
- **LDR (Light Dependent Resistor)**: The LDR is connected to the **analog input A0** on the Arduino. The LDR detects light intensity changes when the laser beam is interrupted.
- **NodeMCU Communication**: The NodeMCU communicates with the Arduino via **SoftwareSerial** (digital pins 10 and 11 for RX and TX). This connection allows remote control through a web server hosted on the NodeMCU, which turns the laser ON/OFF.
- **Power Connections**: The Arduino is powered either through USB or an external power supply, and the laser module receives its power directly from the Arduino's 5V pin. The NodeMCU receives power from a power bank or a regulated source, ensuring the device remains active for remote access.

This detailed setup ensures the laser module only activates when required and deactivates as the NodeMCU commands through its web interface. The LDR captures any interruptions, triggering the security alert, effectively creating a reliable, remotely accessible laser security system.

Chapter 3 : Software

In the software section of this project, we cover the development tools, 3D models simulation, code or programs with their logical analysis, Testing and debugging of the programs which are used to control and monitor the Invisible Security System. The system relies on code written in Arduino IDE for both the Arduino and NodeMCU.

- **Arduino Code**: Manages the laser and LDR detection system. It processes light interruptions to detect breaches and sends signals to trigger the alarm.
- **NodeMCU Code**: Creates a web server interface for remote access, allowing users to activate or deactivate the laser through an internet-connected device.

The code integrates libraries for controlling the LCD display and setting up serial communication between the Arduino and NodeMCU. Additionally, the software includes routines to handle user commands from the web interface, making the system interactive and adaptable for security needs.

3.1 Required Software

Here's a breakdown of the software and IDE used for the Invisible Security System project:

1) **Arduino IDE:** The Arduino IDE is a powerful, user-friendly development environment used to program Arduino microcontrollers, including boards like the Arduino UNO and ESP8266-based NodeMCU. Here's an overview of its main components:

   - **Code Editor:** Allows users to write, edit, and format C/C++ code. Syntax highlighting and indentation make it easy to navigate code structures.

   - **Compiler and Uploader:** The "Verify" button compiles the code, checking for errors, while the "Upload" button transfers it to the connected microcontroller.

   - **Serial Monitor:** A built-in console that displays serial data from the board, useful for debugging and viewing real-time sensor data or feedback.

- **Library Manager:** Facilitates downloading and integrating libraries (like `LiquidCrystal_I2C` and `ESP8266WiFi`), allowing users to extend functionality without writing code from scratch.

- **Board Manager:** Allows users to add support for different boards, including the NodeMCU, by downloading specific configurations and core files.

2) **Tinkercad:** For this project, Tinkercad was used to design both the circuit and 3D model. Tinkercad is an online platform providing tools for electronics design, simulation, and 3D modeling. It allows users to create circuits with virtual components like microcontrollers, sensors, LEDs, and more, making it easy to test and refine designs before building physical prototypes. Additionally, Tinker cad's 3D modeling features let users design and visualize the physical enclosure or structure of the project, contributing to both the project's functionality and aesthetics.

3.2 Required Libraries

❖ **Programming Languages**:

C/C++ Used in the Arduino IDE for programming the Arduino and NodeMCU microcontrollers, providing the necessary control over hardware components.

❖ **Libraries**:

- **ESP8266WiFi Library**: Manages the Wi-Fi connectivity on NodeMCU, allowing the device to create and host a web server.
- **ESP8266WebServer Library**: Creates a web server interface for user interactions. Used to generate control buttons for turning the laser module on or off.
- **LiquidCrystal_I2C Library**: Controls the I2C LCD, displaying messages like "Alarm Enable" and "Alarm Disable" to communicate system status.
- **SoftwareSerial Library**: Enables serial communication between Arduino and NodeMCU, allowing them to send commands and control the laser from the web server.

3.3 3D Simulation

In this project, [Tinkercad] was used to create a 3D model that visualizes the physical setup of the laser security system (*figure:3.1*). The 3D model represents the structural layout, including the placement of components like the laser module, LDR, Arduino, and NodeMCU. By designing in Tinkercad, the project benefits from a precise and clear visualization of component arrangement, ensuring optimal space usage and alignment. This virtual modeling allowed for design adjustments before physical assembly, making the construction process more efficient and accurate.

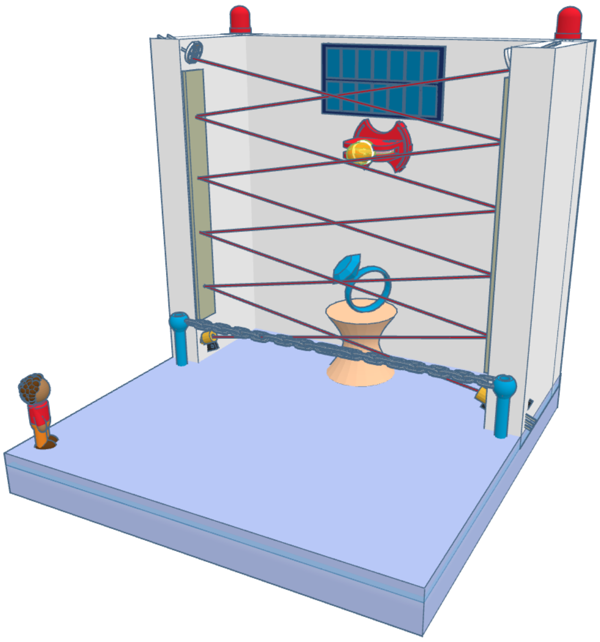

Figure 3.1 : 3D model of the System

3.4 Code Analysis

This section provides a comprehensive analysis of the project's core code, explaining how each part contributes to the functionality of the laser security system. It describes the purpose of essential libraries, pin configurations, and the setup and loop functions. The code integrates various components, such as the LCD for displaying the system's status and the LDR for detecting interruptions. It also covers communication with the NodeMCU for remote control, demonstrating how the code manages real-time monitoring, user commands, and feedback display for an efficient security system.

(A)The Arduino Controller Codes

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <SoftwareSerial.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);                  // Address 0x27, 16 columns, 2 row
SoftwareSerial arduinoSerial(0, 1);                  // RX, TX for communication with NodeMCU
int laserPin = 7;                                    // Pin connected to the laser module
int ledPin = 13;                                     //LED red light in the Arduino 13 pin
int ldrPin = A0;                                     // LDR connected to A0
int threshold = 800;                                 // LDR threshold for detection
int buzzerPin = 12;                                  // Pin for the buzzer
int ldrValue = 0;                                    // store LDR value
   void setup() {
           pinMode(laserPin, OUTPUT);
           pinMode(buzzerPin, OUTPUT);
           digitalWrite(laserPin, LOW);              // Start with laser OFF
           digitalWrite(buzzerPin, LOW);             // Start with buzzer OFF
           arduinoSerial.begin(9600);                // Start communication with NodeMCU
           Serial.begin(19200);                      // Start serial for debugging
           digitalWrite(laserPin, HIGH);             // Turn on the laser
           lcd.begin(1,1);                           // Initialize the LCD
           lcd.backlight();
           lcd.setCursor(0, 0);
           lcd.print("Authorized Only");             // Default display message
           lcd.setCursor(0, 1);
           lcd.print("Security Online");
          }
   void loop() {
           int ldrValue = analogRead(ldrPin);        // LDR monitoring
           Serial.print("LDR Value: ");              // Print LDR value for debugging
           Serial.println(ldrValue);
           if (ldrValue > threshold) {
                 lcd.clear();                        // Update LCD to show "Access Denied"
                 lcd.setCursor(0, 0);
                 lcd.print("Access Denied");
                 Serial.println("Laser interrupted!");
                 tone(buzzerPin, 800);               // Trigger alarm and update LCD
                 digitalWrite(ledPin, HIGH);
                 delay(500);
                 tone(buzzerPin, 900);
                 digitalWrite(ledPin, LOW);
                 delay(500);  }
           else {
                 noTone(buzzerPin);                  // Reset buzzer
                 digitalWrite(ledPin, LOW);
                 lcd.clear();                        // Update LCD to show "Authorized Only"
                 lcd.setCursor(0, 0);
                 lcd.print("Authorized Only");
                 lcd.setCursor(0, 1);
                 lcd.print("Security Online");  }
       if (arduinoSerial.available()) {              // Check for commands from the NodeMCU
           String command = arduinoSerial.readStringUntil('\n');
           Serial.println("Received: " + command);
           while (command == "ON") {
                digitalWrite(laserPin, HIGH);        // Turn ON laser
                Serial.println("Laser ON");
                lcd.clear();
                lcd.setCursor(0, 1);
                lcd.print("Security Online");    }
           while (command == "OFF") {
                digitalWrite(laserPin, LOW);         //Turn OFF laser
                noTone(buzzerPin);
                Serial.println("Laser OFF");
                lcd.clear();
                lcd.setCursor(0, 1);
                lcd.print("Access Granted");    }   }
       delay(100);       }                           //Delay for stability
```

(B)The NodeMCU Controller Codes

```
#include <ESP8266WiFi.h>
#include <SoftwareSerial.h>
const char* ssid = "NDUB-Student-Wifi";           // My network credentials
const char* password = "NDUB@STU@$#";
WiFiServer server(80);                             // Web server on port 80
SoftwareSerial arduinoSerial(D2, D3);              // SoftwareSerial to communicate with
                                                   //_Arduino
int ledPin = D5;

   void setup() {
           pinMode(LED_BUILTIN, OUTPUT);
           pinMode(ledPin, OUTPUT);
           Serial.begin(115200);                   // Start Serial for debug
           arduinoSerial.begin(9600);              // Communication with Arduino
           WiFi.begin(ssid, password);             // Connect to Wi-Fi
           while (WiFi.status() != WL_CONNECTED) {
                delay(500);
                Serial.print("Connecting...\n");
              }
           Serial.println("WiFi connected");
           server.begin();                         // Start the server
           Serial.println("Server started");
           digitalWrite(ledPin, 5);
           Serial.println(WiFi.localIP());         // Print the IP address
     }

   void loop() {
           Serial.println(WiFi.localIP());
           WiFiClient client = server.available();  // Check for clients
           if (!client) {
               return;
           }
           while (!client.available()) {           // Wait until the client sends some data
             delay(500);
             digitalWrite(LED_BUILTIN, HIGH);
             delay(500);
             digitalWrite(LED_BUILTIN, LOW);
             delay(500);
           }
           String request = client.readStringUntil('\r');   // Read the request
           Serial.println(request);
           client.flush();
           if (request.indexOf("/on") != -1) {              // Control the Buzzer
             arduinoSerial.println("ON");                   // Send ON command to Arduino
           } else if (request.indexOf("off") != -1) {
             arduinoSerial.println("OFF");                  // Send OFF command to Arduino
           }
  String html = "<!DOCTYPE html><html>";     // Prepare the HTML web page
  html += "<head><meta name=\"viewport\" content=\"width=device-width, initial-scale=1\">";
  html += "<style>html { font-family: Helvetica; display: inline-block; text-align:center;}";
  html += ".button { background-color: #4CAF50; border: none;color:white;padding:16px 40px;";
  html += "text-decoration: none; font-size: 30px; margin: 2px; cursor: pointer;}";
  html += ".button2 {background-color: #f44336;}</style></head>";
  html += "<body><h1>Laser Control</h1>";
  html += "<p><a href=\"/on\"><button class=\"button\">Laser ON</button></a></p>";
  html += "<p><a href=\"/off\"><button class=\"button button2\">Laser OFF</button></a></p>";
  html += "</body></html>";
         client.println("HTTP/1.1 200 OK");                 // Send the response to client
         client.println("Content-type:text/html");
         client.println("Connection: close");
         client.println();
         client.println(html);
         client.println();
         delay(10);
     }
```

3.4.1 Logical Breakdown of function( )

For the **Code Analysis** section of the project, here's a breakdown of each functional part of the code for your laser security system:

1. **Library Inclusions and Initialization:** The code starts with libraries, such as LiquidCrystal_I2C for the LCD display, and SoftwareSerial for serial communication with the NodeMCU. The LiquidCrystal_I2C library manages text display, while SoftwareSerial enables communication between the Arduino and NodeMCU.

2. **Pin Declarations:** Specific pins are assigned to components like the laser, LDR, and LCD. For example, laserPin is the pin connected to the laser module, while ldrPin is connected to the LDR for reading light levels. This setup ensures accurate control and monitoring.

3. **Setup Function:** Initializes communication with the NodeMCU and LCD and sets the laser pin as an output. The laser is initially set to the OFF state, ensuring security control is only activated when necessary.

4. **Main Loop:** The loop performs three tasks**:**

   a) **Web Commands:** Reads commands from the NodeMCU to control the laser. For example, "ON" and "OFF" commands are read and executed to control the laser.

   b) **LDR Monitoring:** Continuously monitors the LDR's analog value to detect if the laser beam is interrupted, signaling unauthorized access.

   c) **LCD Display:** Displays "Alarm Enable" or "Alarm Disable" based on the laser and LDR status, providing visual feedback.

5. **Serial Communication:** The communication between Arduino and NodeMCU is facilitated through SoftwareSerial. NodeMCU's web interface sends control commands to Arduino for activating or deactivating the laser, thereby integrating web-based control into the security system.

This analysis showcases how the code integrates real-time monitoring, user interaction, and feedback display, making the laser security system effective and easy to control remotely.

3.4.2 Code Breakdown of Arduino Uno R3

This Arduino code for the laser security system controls components like the laser, buzzer, LDR (Light Dependent Resistor), LCD, and LED. The NodeMCU sends commands to turn the laser on and off, while the LDR monitors light intensity. If the light drops (laser interrupted), it triggers an alarm and updates the LCD display to show "Access Denied." Here is the Algorithm of Arduino Uno :

1. **Initial Setup**:
   - ✓ The LCD, laser, and buzzer pins are initialized.
   - ✓ The LCD displays “Authorized Only” and “Security Online” by default.
2. **Loop Function**:
   - ✓ Continuously reads the LDR sensor's value. If it exceeds the threshold, indicating an intrusion, it:
     - Activates the buzzer with alternating frequencies.
     - Shows "Access Denied" on the LCD.
   - ✓ If no interruption is detected, the LCD displays the default message and resets the buzzer.
3. **NodeMCU Communication**:
   - ✓ If it receives the "ON" command from NodeMCU, it powers on the laser and resets the display.
   - ✓ If the command is "OFF," it turns off the laser, silences the buzzer, and updates the LCD to show “Access Granted.”

In conclusion, this Arduino-based laser security system effectively demonstrates how a combination of sensors, LEDs, and communication modules can form a basic yet powerful security mechanism. By using an LDR sensor to detect interruptions in the laser beam, an alarm can sound immediately upon unauthorized access. NodeMCU integration enables remote control of the laser system, showcasing how IoT technologies can enhance traditional security systems. This system is easily adaptable and could be extended for more complex and robust security applications in the future.

3.4.3 Flowchart of Arduino Uno R3

The flowchart (*figure:3.2*) for the Arduino Uno provides a visual representation of the program logic used in the project. It illustrates the sequential steps followed by the microcontroller to control the security system. Key operations include initializing components (LCD, laser, buzzer), connecting to the ESP8266 for Wi-Fi communication, reading the LDR sensor values to detect interruptions in the laser beam, and activating the alarm if unauthorized access is detected. This flowchart simplifies understanding of the workflow and logical decisions implemented in the system.

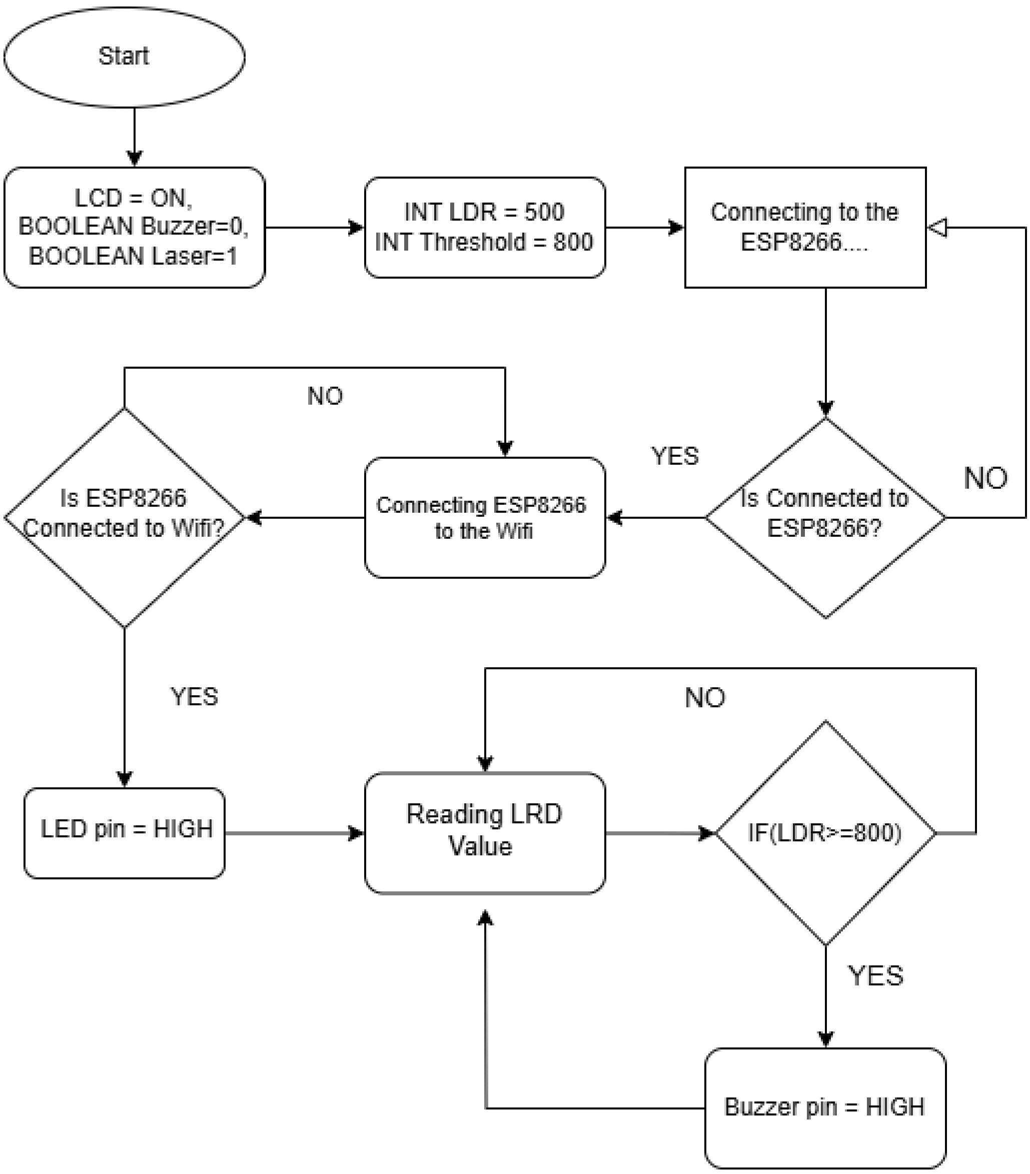

Figure 3.2 : Flowchart of Arduino

3.4.4 Code Breakdown of ESP8266

This NodeMCU code connects the ESP8266 to Wi-Fi, enabling a web server to control a laser module connected to an Arduino through software serial communication. Here is the Algorithm of ESP8266 :

1. **Wi-Fi Setup:** The `ssid` and `password` variables store Wi-Fi credentials, and `WiFi.begin` initiates the connection. The IP address is displayed in the serial monitor upon successful connection.
2. **Server Setup:** A web server is created on port 80 (`WiFiServer server(80)`), and once connected, the server begins listening for client requests.
3. **Web Interface and Communication:**
   - ✓ In loop, the server listens for clients and requests. If /on is requested, "ON" is sent to the Arduino over software serial, turning on the laser. For /off, "OFF" is sent to turn it off.
   - ✓ The HTML structure is created dynamically, (*fig:3.3*) featuring **ON** and **OFF** buttons to control the laser. The buttons trigger the respective Arduino commands through the NodeMCU web server.

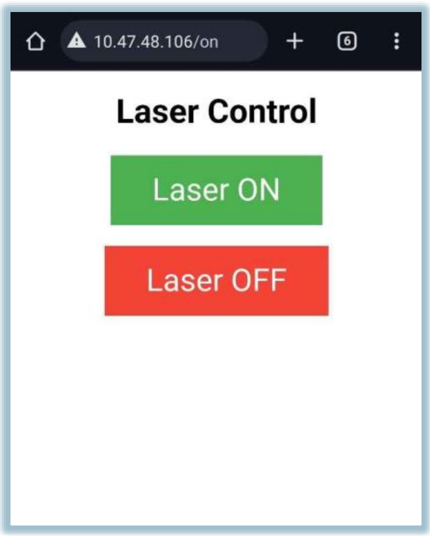

Figure 3.3 : Web Interface of ESP8266

4. **LED Indication:** LED on the NodeMCU blinks during the connection, providing a visual indicator that the web server is running and awaiting client requests.

This approach effectively controls the Arduino and laser system remotely via a web interface with basic HTTP commands.

3.4.5 Flowchart of ESP8266

The provided flowchart (*figure:3.4*) illustrates the operational flow of the ESP8266 module within the laser-based security system. It outlines the module's core tasks, beginning with establishing a Wi-Fi connection using predefined SSID credentials. Once connected, the ESP8266 initializes its server on port 80 to listen for incoming client requests. The flowchart highlights decision-making steps, such as verifying requests from the client to toggle the laser or buzzer states. This logical sequence ensures seamless communication and remote-control capabilities for the system.

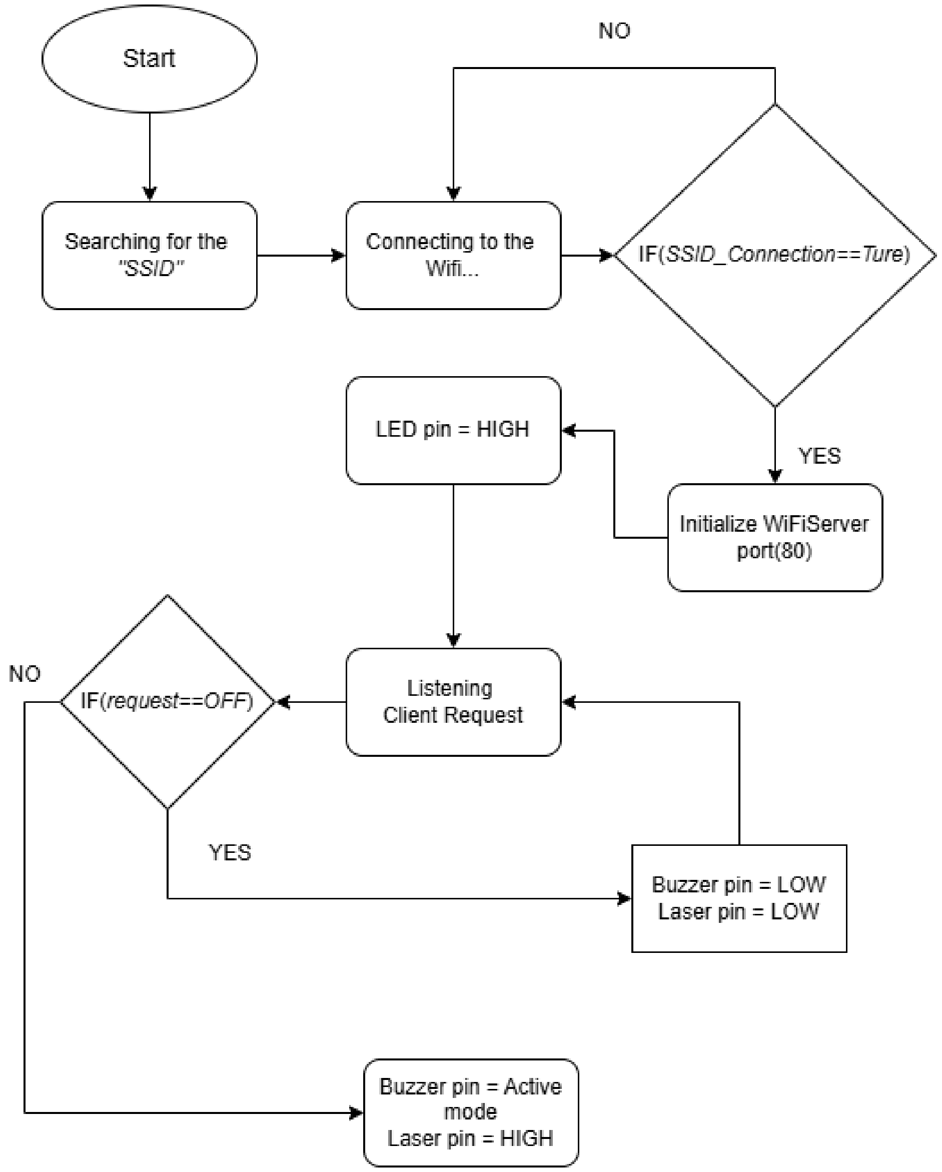

Figure 3.4 : Flowchart of ESP8266

Chapter 4 : Project Output

In this section, images of the completed laser security system project are presented to showcase the design, assembly, and overall layout. Here we have built a DIY museum, where a valuable item has been secured by the Invisible security system. Whenever any unauthorized entity tries to enter this secured area, The LDR will not get the laser beam. As a result, it will trigger the alarm. It will also show the message "**Access denied**" for unauthorized entry. These images or visuals serve to provide a clear understanding of the final build and demonstrate the successful implementation of the system. Each image is accompanied by brief descriptions to highlight essential features and scenario, detailing the setup and functionality.

4.1 Project Result

- ✓ **Front View:** This image captures the front perspective of the security system, displaying the primary components, including the laser module, LDR, and control interface. It showcases the overall appearance and design of the project.

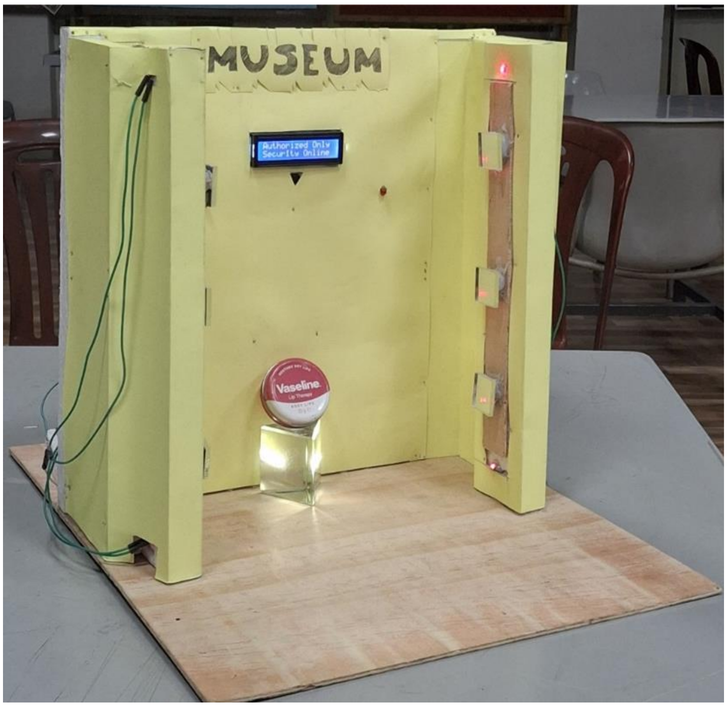

Figure 4.1 : Front View

- ✓ **Back View of Circuit Connection**: This photo provides a detailed look at the wiring and circuit connections at the back. It shows the setup of components like the Arduino, NodeMCU, and necessary wiring for laser control and detection.

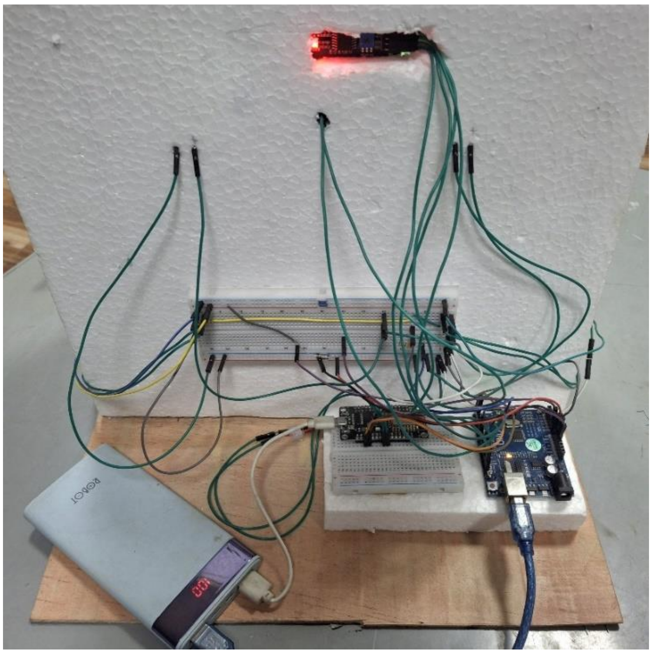

Figure 4.2 : Back View

- ✓ **Laser Beam Visible with Smoke:** Here, the laser beam becomes visible when smoke is introduced, highlighting the path of the beam. This image demonstrates how the laser provides a visible line for security detection.

Figure 4.3 : Visible Laser

4.2 Testing & Bug Report

- ✓ **Unauthorized Entry:** This image shows the LCD or display indicating "Access Denied" during unauthorized entry. It illustrates the alert system, signaling when the laser beam is interrupted.

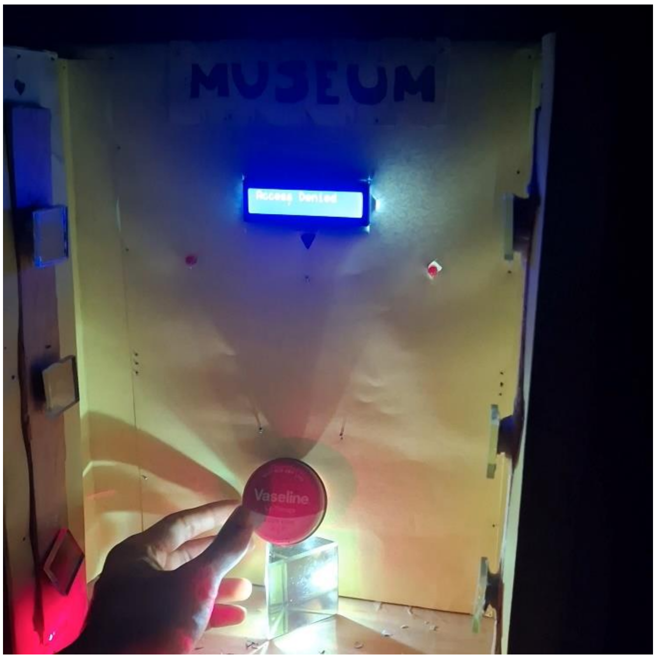

Figure 4.4 : LCD Display

These images collectively validate the system's effectiveness and align with the project's objectives of providing security and functionality. From the pictures we can come to the decision that the project worked successfully as per its agenda . Overall, the project combines hardware and software to create a responsive and interactive security solution.

4.3 Challenges & Debugging

Constructing a project from scratch is not an easy task, especially where hardware and software both are involved. During the development of the laser security system, several challenges could arise:

- **Sensor Sensitivity:** Calibrating and positioning the LDR to accurately detect laser interruptions without false triggers due to lighting changes required precise adjustment.
- **Connectivity Issues:** Ensuring stable communication between the NodeMCU and Arduino, particularly with Wi-Fi connections, can be challenging in areas with weak and multiple signals.
- **Power Supply Stability:** Providing consistent power to the system, especially when using multiple components, was essential to prevent malfunctions.
- **Hardware Integration:** Proper alignment of the laser and LDR, as well as securing components, was necessary for optimal performance.
- **Environmental Factors:** Adjusting for environmental elements like dust or smoke that could interfere with the laser beam's visibility and functionality.
- **Reflection Stability:** The mirrors should be adjusted in such a way that the laser beam reflects from bottom to top to the LDR in a crisscross pattern, But the angles of the mirror and the laser beam was not accurate at the point when we implement the circuit. Thus, it was not reflecting the laser to the LDR. And this issue was fixed by cutting the mirror pieces to the required size and placing them into the accurate position.

These challenges required careful troubleshooting and iterative testing to overcome, ultimately leading to a robust and reliable system.

Chapter 5: Conclusion

The Invisible Security System was our initial take on this project. In this section we have attached some of the future enhancements that can be integrated to this project for vast usage.

5.1 Potential Improvement

The Invisible Security System holds significant potential for future enhancements. Integrating artificial intelligence could enable advanced anomaly detection, reducing false alarms and ensuring more reliable security. Expanding sensor coverage with additional LDRs and lasers would allow the system to monitor larger areas. To address power outages and environmental adaptability, solar power backups and dynamic threshold adjustments could be implemented. IoT integration would enhance notifications through push alerts and emails, while compatibility with smart home ecosystems could streamline control. Moreover, advanced features like facial recognition or tamper detection would elevate its functionality and robustness.

The Invisible Security System can be improved with the following advancements:

1. **AI Integration for Anomaly Detection**: Incorporate AI models to analyze patterns and detect suspicious activities, reducing false alarms and improving security.
2. **Multiple Sensors**: Add more LDRs and laser modules to cover larger areas or multiple entry points.
3. **Environment Adaptability**: Implement adaptive thresholds to ensure the system works in varying light conditions.
4. **Advanced Notifications**: Integrate IoT platforms for real-time alerts via push notifications, email, or phone calls.
5. **Integration with Smart Home Systems**: Enable compatibility with platforms like Alexa or Google Home for seamless control and monitoring.
6. **Facial Recognition Access**: Incorporate facial recognition for advanced user authentication.
7. **Solar Power Backup**: Use solar panels for energy efficiency and uninterrupted operation during power outages.
8. **Tamper Detection**: Add tamper-proof mechanisms to alert when hardware is compromised.

5.2 Future Enhancement

Future enhancements for the Invisible Security System include integrating a live-streaming camera for remote monitoring, developing a mobile app for centralized control, and adding PIR motion sensors to reduce false alarms. A rechargeable battery backup ensures continuous operation during power outages, and implementing smart notifications via SMS or push alerts enhances user response. These updates improve reliability, accuracy, and user convenience, making the system more robust and adaptable to modern security needs.

- **Remote Monitoring with Live Streaming**: Integrating a camera allows real-time video feed access through a web app or mobile interface. Users can visually monitor secured areas, which is especially useful for remote property management, allowing for immediate visual assessment of security breaches.
- **Mobile App Integration for Comprehensive Control**: A dedicated mobile app could centralize all control functions, such as activating/deactivating the system, adjusting settings, and viewing camera feeds. This could also support two-factor authentication for added security.
- **Enhanced Motion Detection Using PIR Sensors**: Adding passive infrared (PIR) sensors would increase accuracy by detecting human presence rather than relying solely on light interruptions. This would help prevent false alarms caused by fluctuations in ambient lighting, ensuring alerts are more specific to actual motion.
- **Smart Notification System for Instant Alerts**: By enabling SMS and push notifications, users receive immediate alerts about security incidents. Integrating this feature with a cloud service can further customize notifications, including alerts based on location and specific user permissions. Notifications can include snapshots or live camera access, further aiding response time.
- **Integration with Smart Home Systems**: The system could integrate with popular smart home systems like Google Home for voice control. This would allow users to interact with the security system effortlessly, making activation and deactivation as simple as a voice command.
- **Data Logging and Reporting**: Adding a data logging feature would record all triggered events, providing a record of all security breaches, timestamped for reference. Users could access this data via the mobile app or web interface, enabling comprehensive reporting and review of security activity over time.

These improvements could make the system more robust, user-friendly, and adaptable for various security applications.

5.3 Conclusion

The Invisible Security System successfully demonstrates an innovative and efficient way to secure restricted areas using a laser beam, LDR, and IoT integration. The project achieves its objective of providing real-time intrusion detection through hardware and software synergy. The addition of a NodeMCU web server for remote control and visual indicators like the LCD enhances functionality and usability. With its adaptability and scope for future enhancements, such as smart notifications and mobile app integration, this system represents a scalable and reliable security solution for modern needs.

The project commenced on **26th August 2024** and was successfully completed on **9th November 2024**, spanning a total of **approximately 2.5 months**. During this timeline, the team focused on designing, assembling, coding, testing, and refining the system. Each phase, from circuit design and hardware integration to software development and troubleshooting, was carefully executed. This structured timeline allowed for efficient progress and ensured the delivery of a fully functional and reliable laser based Invisible Security System.

5.4 Reference

1. Optex Pinnacle *[Access on 09-09-24]*
2. How to make a laser security system with Arduino (Easy Tutorial, incl. Sketch Code) *[Access on 07-09-24]*
3. Learning Arduino *[Access on 26-08-24]*