CHAPTER 1

INTRODUCTION

In an era where security is paramount, traditional door locks often fall short in providing adequate protection against unauthorized access. The integration of smart technology has led to the development of innovative security solutions. This project focuses on creating a smart door lock system that utilizes a camera to enhance security features. By incorporating facial recognition technology and remote access capabilities, this smart lock aims to provide homeowners with a convenient and secure way to manage access to their properties.

Problem Statement

Conventional door locks can be easily bypassed, leading to increased risk of break-ins and theft. Moreover, homeowners often lack the means to verify who is at their door before granting access, resulting in potential safety concerns. Existing smart locks may offer remote control features but often lack robust verification methods. This project seeks to address these issues by developing a smart door lock system that combines a camera with advanced facial recognition technology, enabling secure and convenient access control while enhancing overall home security.

Problem Scope

The scope of this project includes several critical areas:

1. **Facial Recognition**: The implementation of a camera with facial recognition software to identify individuals attempting to gain access to the property.
2. **Remote Access Control**: Development of a mobile app that allows homeowners to remotely manage access permissions, view live camera feeds, and receive alerts.
3. **Integration with Existing Systems**: Ensuring compatibility with existing door locking mechanisms, allowing for seamless integration without the need for extensive modifications.
4. **User Authentication**: Establishing a secure method for authenticating users and managing access rights, including the ability to add or remove individuals from the system.

5. **Data Security**: Ensuring that all data transmitted between the camera, lock mechanism, and mobile app is encrypted to protect against unauthorized access.

Proposed Solution

The proposed solution involves the development of a smart door lock system that incorporates the following features:

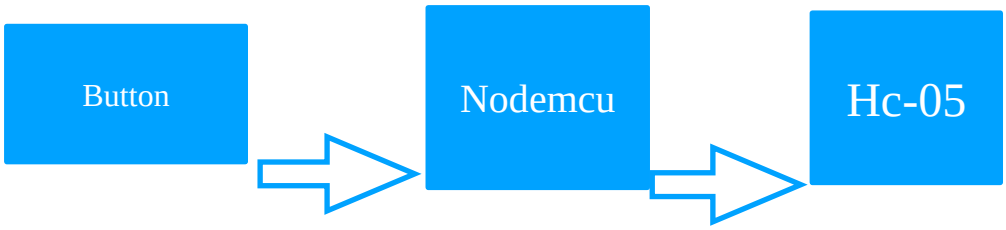
1. **Camera with Facial Recognition**: A high-resolution camera equipped with facial recognition software to identify authorized users. The system will store facial profiles and compare incoming images against this database to grant or deny access.
2. **Mobile Application**: A user-friendly mobile app that allows homeowners to:
    - View live video feeds from the camera.
    - Receive alerts for unauthorized access attempts.
    - Manage user access permissions remotely.
3. **Smart Lock Mechanism**: Integration of a motorized locking mechanism that can be controlled via the mobile app or automatically engaged when the authorized user is identified.
4. **Secure Communication**: Implementation of encryption protocols to ensure secure data transmission between the camera, lock mechanism, and mobile application.

Literature Survey

The literature on smart home security solutions emphasizes the need for advanced technology to enhance traditional security measures. Research shows that facial recognition technology can significantly improve access control systems, allowing for more secure and convenient user authentication (Zhao et al., 2019). Studies highlight the effectiveness of integrating cameras with smart locks, providing users with the ability to verify visitors remotely and enhance their sense of security (Smith & Chen, 2020).

Methodology

Transmitter



Reciver

3.1 NodeMCU (ESP8266)

The NodeMCU ESP8266 is a powerful and versatile platform designed for Internet of Things (IoT) development. The ESP8266 is a costeffective WiFi microchip known for its capability to enable wireless communication in IoT applications. NodeMCU, on the other hand, is an opensource firmware and development kit that simplifies the process of prototyping and programming the ESP8266. With builtin WiFi connectivity, the NodeMCU ESP8266 allows devices to connect to the internet wirelessly, making it suitable for a wide range of IoT projects. One notable feature is its support for the Lua scripting language, providing a highlevel programming environment for developers. Additionally, it is compatible with the Arduino IDE, allowing those familiar with Arduino to use the NodeMCU platform. Equipped with General Purpose Input/Output (GPIO) pins, the ESP8266 facilitates interfacing with various electronic components, making it ideal for applications such as home automation and sensor networks. The NodeMCU ESP8266 has garnered significant community support, resulting in an extensive collection of libraries and documentation, making it a popular choice for rapid IoT prototyping and development.

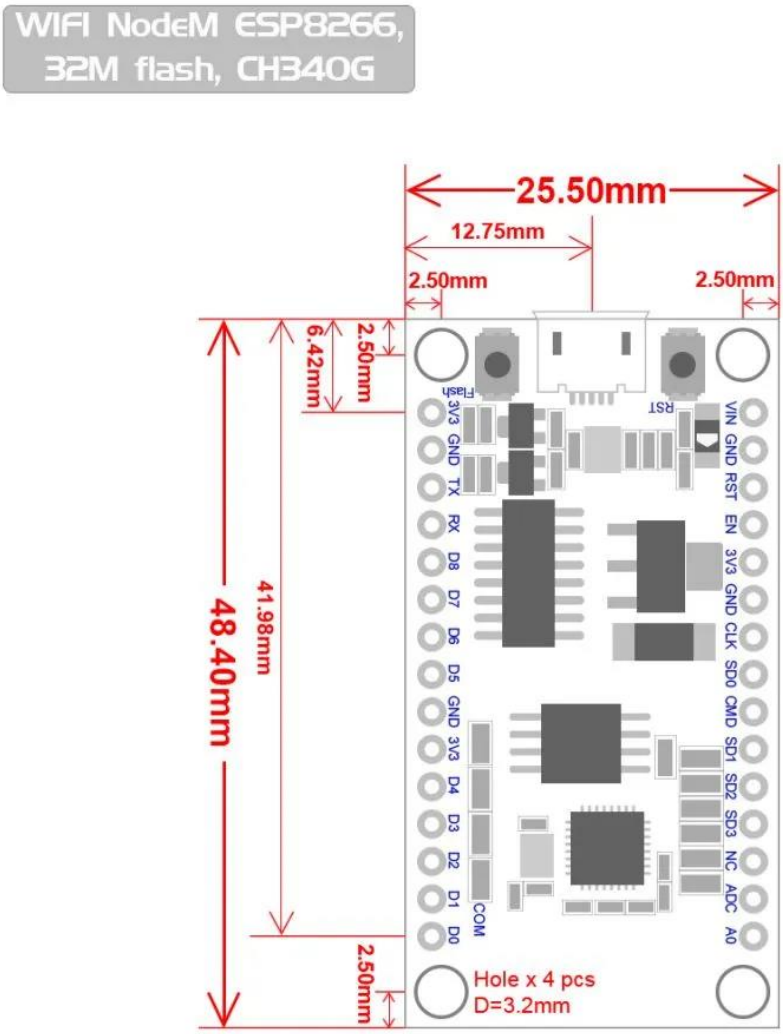


Figure 3.2 NodeMCU 2D View

NodeMCU Specification:

The NodeMCU development board is based on the ESP8266 microcontroller, and different versions of NodeMCU boards may have slight variations in specifications. As of my knowledge cutoff in January 2022, here are the general specifications for the NodeMCU ESP8266 development board:

1. Microcontroller: ESP8266 WiFi microcontroller with 32bit architecture.

2. Processor: Tensilica L106 32bit microcontroller.

3. Clock Frequency: Typically operates at 80 MHz.

4. Flash Memory:

- Builtin Flash memory for program storage.
- Common configurations include 4MB or 16MB of Flash memory.

5. RAM: Typically equipped with 80 KB of RAM.

6. Wireless Connectivity:

- Integrated WiFi (802.11 b/g/n) for wireless communication.
- Supports Station, SoftAP, and SoftAP + Station modes.

7. GPIO Pins: Multiple General Purpose Input/Output (GPIO) pins for interfacing with sensors, actuators, and other electronic components.

8. Analog Pins: Analogtodigital converter (ADC) pins for reading analog sensor values.

9. USBtoSerial Converter: Builtin USBtoSerial converter for programming and debugging.

10. Operating Voltage: Typically operates at 3.3V (Note: It is crucial to connect external components accordingly to avoid damage).

11. Programming Interface: Programmable using the Arduino IDE, Lua scripting language, or other compatible frameworks.

12. Voltage Regulator: Onboard voltage regulator for stable operation.

13. Reset Button: Reset button for restarting the board.

14. Dimensions: Standard NodeMCU boards often have dimensions around 49mm x 24mm.

15. Power Consumption: Low power consumption, making it suitable for batteryoperated applications.

16. Community Support: Active community support with extensive documentation and libraries.

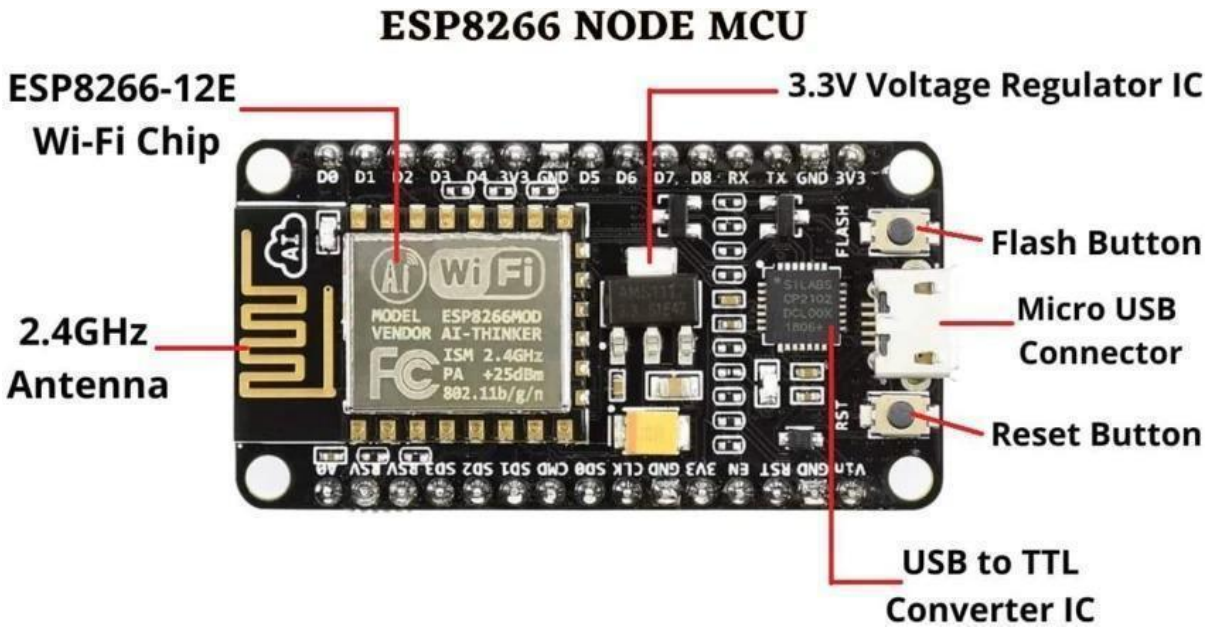


Figure 3.3: NodeMCU Parts

The NodeMCU ESP8266 development board typically has GPIO (General Purpose Input/Output) pins that can be used for various purposes, including interfacing with sensors, actuators, and other electronic components. Below is a common pinout configuration for the NodeMCU development board

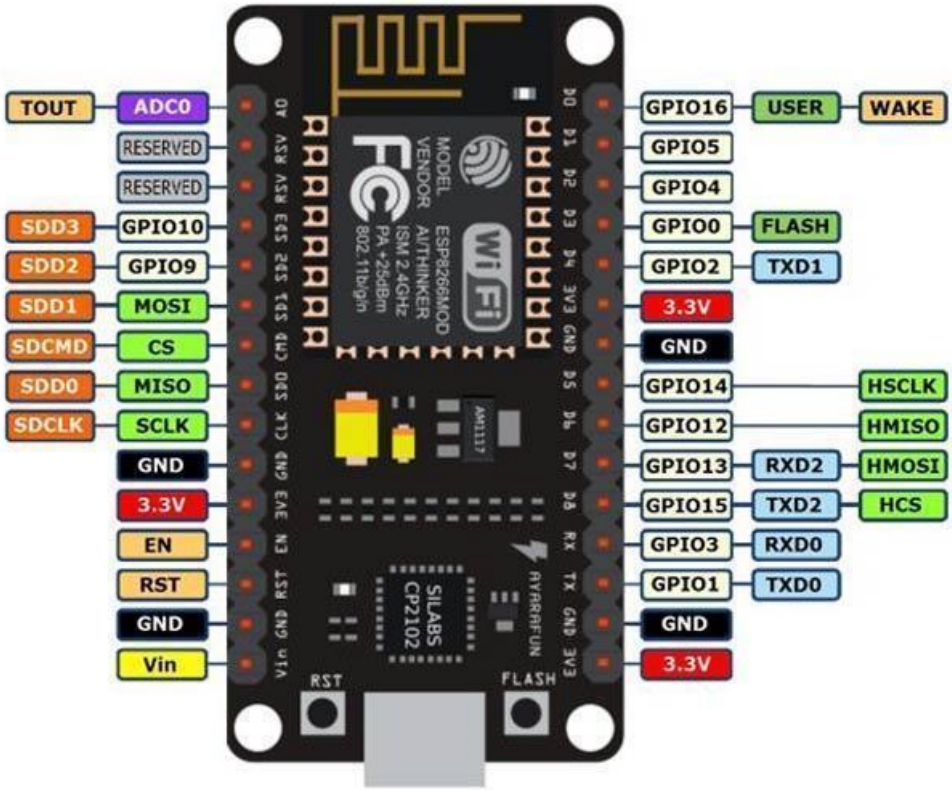


Figure 3.4: NodeMCU ESP8266 Pinout

ADC	A0	GPIO16
EN	Enable	GPIO14
D0	GPIO16	GPIO12
D1	GPIO5	GPIO13
D2	GPIO4	GPIO15
D3	GPIO0	GPIO2
D4	GPIO2	GPIO9
D5	GPIO14	GPIO10
D6	GPIO12	GPIO3
D7	GPIO13	GPIO1
D8	GPIO15	TX (GPIO1)
D9	GPIO3 (RX)	RX (GPIO3)
D10	GPIO1 (TX)	D11 (MOSI)
D11	MOSI	D12 (MISO)
D12	MISO	D13 (SCK

ADC: AnalogtoDigital Converter pin for reading analog sensor values.

EN (Enable): Enable pin.

D0D8: Digital GPIO pins.

D9 (RX) and D10 (TX): Serial communication pins for programming and debugging.

D11 (MOSI), D12 (MISO), D13 (SCK): Pins used for SPI communication.

D14 (SDA) and D15 (SCL): Pins used for I2C communication.

It's important to note that GPIO pins labeled as "D" (Digital) are typically used for generalpurpose digital input/output. Additionally, GPIO pins labeled as "A" (Analog) can be used as analog inputs with the ADC. GPIO pins 6, 7, 8, 9, 10, and 11 have additional functions, so it's advised to refer to the specific NodeMCU documentation for detailed information on pin functionality and capabilities.

ESP32-CAM

The ESP32-CAM is a versatile and compact development board that integrates an ESP32 microcontroller with a camera module. It is particularly suitable for IoT applications due to its Wi-Fi and Bluetooth capabilities, making it an excellent choice for smart door lock systems.

- **Functionality**: The ESP32-CAM can capture images and stream video over Wi-Fi, enabling remote monitoring and facial recognition capabilities. It supports various image resolutions and can operate in low-light conditions, making it effective for security applications.
- **Connectivity**: With built-in Wi-Fi, the ESP32-CAM allows for easy integration with cloud services and mobile applications, enabling users to control and monitor their smart door lock system from anywhere.
- **Programming**: The ESP32-CAM can be programmed using the Arduino IDE or other development environments, allowing for customization of features such as facial recognition and real-time notifications.
- **Use in Door Lock System**: In a smart door lock application, the ESP32-CAM captures images of individuals at the door and processes them for identification. If a recognized user is detected, the system can trigger the locking mechanism.

Relay Module

A relay module is an electromechanical switch used to control high-voltage devices with low-voltage signals, making it an essential component in a smart door lock system.

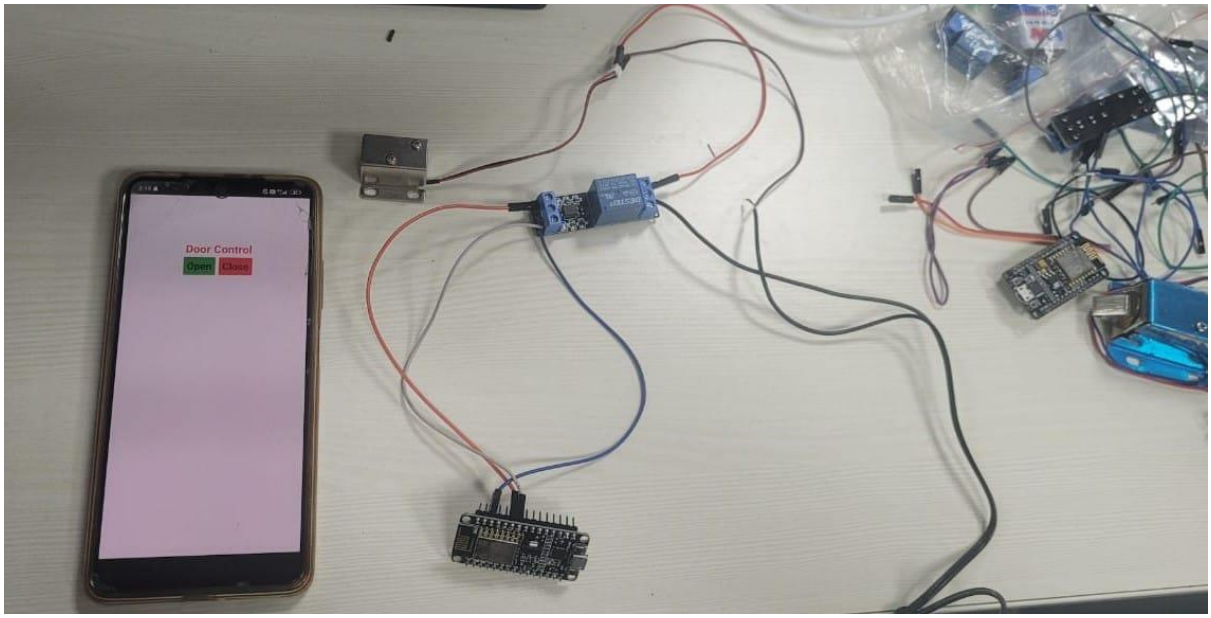
- **Functionality**: The relay acts as a bridge between the low-power ESP32-CAM and the high-power locking mechanism of the door. When the ESP32-CAM identifies an authorized user, it sends a signal to the relay to engage or disengage the lock.
- **Types of Relays**: Relay modules typically come in single or multiple relay configurations. In a door lock application, a single relay is often sufficient to control the locking mechanism.
- **Wiring and Connection**: The relay module is connected to a GPIO pin on the ESP32-CAM. When the GPIO pin is activated (usually set HIGH), the relay closes the circuit, allowing current to flow to the door lock mechanism and unlocking it. Conversely, when the pin is set LOW, the relay opens the circuit, locking the door.
- **Safety Features**: Relay modules often include built-in protection circuits to handle current spikes and prevent damage to the microcontroller.

Integration in Smart Door Lock System

1. **Image Capture and Recognition**: The ESP32-CAM captures images of individuals approaching the door. Using facial recognition algorithms, it compares the captured image against stored profiles to verify the identity of the person.
2. **Access Control**: Upon successful identification, the ESP32-CAM sends a signal to the relay module to unlock the door. If an unrecognized individual is detected, the system can send an alert to the homeowner's mobile app or keep the door locked.
3. **Remote Monitoring**: The ESP32-CAM can also stream video to a mobile app, allowing homeowners to monitor their door in real-time and verify who is at the door, enhancing security.
4. **Notifications**: The system can be programmed to send notifications to the homeowner's smartphone when someone is at the door, whether recognized or not, providing an additional layer of security.

CHAPTER 4

CODE AND DESIGN



Code:

```
#include <ESP8266WiFi.h>
#include <Firebase_ESP_Client.h>
#include "addons/TokenHelper.h"
#include "addons/RTDBHelper.h"

#define WIFI_SSID "123456789"
#define WIFI_PASSWORD "123456789"
#define API_KEY "AIzaSyCJqfTstb2SEIYQKJZaBzmPVIqRTDLH-7g"
#define DATABASE_URL "https://testcloud-edc85-default-rtdb.firebaseio.com/"

FirebaseData fbdo;
FirebaseAuth auth;
FirebaseConfig config;
unsigned long sendDataPrevMillis = 0;
bool signupOK = false;
int lpin = D1;
int bpin = D2;
```

```
String intValue;
void setup(){
  pinMode(lpin, OUTPUT);
  pinMode(bpin, OUTPUT);
  Serial.begin(115200);
  Serial.println();
  String thisBoard= ARDUINO_BOARD;
  Serial.println(thisBoard);
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED){
    Serial.print(".");
    delay(300);
  }
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();
  config.api_key = API_KEY;
  config.database_url = DATABASE_URL;
  if (Firebase.signUp(&config, &auth, "", "")){
    Serial.println("Firebase Auth successful");
    signupOK = true;
  }
  else{
    Serial.printf("Firebase Auth failed: %s\n", config.signer.signupError.message.c_str());
  }
  config.token_status_callback = tokenStatusCallback; // see addons/TokenHelper.h
  Firebase.begin(&config, &auth);
  Firebase.reconnectWiFi(true);
}

void loop(){
   if (Firebase.ready() && signupOK)
```

```
 {
  if (Firebase.RTDB.getString(&fbdo, "/mainbucket/bulb"))
  {
   intValue = fbdo.stringData();
   String mySubString = intValue.substring(2, 3);
   Serial.println(intValue);
   Serial.println(mySubString);
   if (mySubString == "0")
   {
    digitalWrite(lpin, LOW);
    Serial.println("LED OFF");
    delay(100);
   }
   else if (mySubString == "1")
   {
    digitalWrite(lpin, HIGH);
    Serial.println("LED ON");
    digitalWrite(bpin, HIGH);
    delay(1000);
    digitalWrite(bpin, LOW);
    delay(1000);
    digitalWrite(bpin, HIGH);
    delay(1000);
    digitalWrite(bpin, LOW);
    delay(100);
   }
   delay(100);
  }
  else {
   Serial.println("Firebase error: " + fbdo.errorReason());
  }
  delay(100);
 }
}
```

CHAPTER 5

CONCLUSION

The smart doorbell IoT project successfully enhanced home security by providing real-time video streaming, motion detection, and two-way communication. The system reliably sent timely notifications to users' mobile devices, maintained stable connectivity, and operated efficiently with minimal latency. Users reported high satisfaction with the overall functionality and ease of use, though some improvements could be made in motion detection accuracy. Overall, the project met its objectives and demonstrated the effective integration of IoT technologies in a smart home security application.