Smart Door Lock System Using ESP32 And ESP32-CAM

Ridham Rangani, Neel Sheth

Electronics and Communication department, Nirma University Ahmedabad

21BEC103@nirmauni.ac.in

21BEC116@nirmauni.ac.in

***Abstract*— The Smart Door Lock System is an innovative security solution designed to authenticate users and grant access to premises efficiently and securely. Central to its functionality is a robust password authentication mechanism. This system allows users to input a password via a keypad. However, in the event of three consecutive incorrect password attempts, an automated email notification is triggered and sent to the registered user's email address. The email serves as a security alert, informing the user of potential unauthorized access attempts and prompting them to take appropriate action. Additionally, the email contains a link to access live streaming from the ESP32-CAM module, providing real-time visual verification of the door's surroundings. This feature enhances the system's security by allowing users to verify the situation remotely and take necessary actions.**

***Keywords*— ESP32, ESP32-CAM, Keypad, LCD Display,**

I. Introduction

At the core of the Smart Door Lock System lies the integration of cutting-edge hardware and software components. The utilization of ESP32 and ESP32-CAM microcontrollers represents a significant leap forward in access control technology. These microcontrollers not only offer powerful processing capabilities but also provide seamless connectivity options, enabling remote monitoring and control of the door lock system. The ESP32-CAM module, equipped with a high-resolution camera, adds an extra layer of security by facilitating visual verification of individuals seeking access. This integration of hardware components not only enhances the security posture of the system but also provides users with greater flexibility and convenience in managing access to their premises.

One of the standout features of the Smart Door Lock System is its robust password authentication mechanism. By requiring users to input a password via a keypad or mobile application, the system ensures that only authorized individuals are granted access. However, recognizing the importance of user feedback and proactive security measures, the system is designed to trigger an automated email notification in the event of three consecutive incorrect password attempts. This email serves as a security alert, notifying the registered user of potential unauthorized access attempts and prompting them to take appropriate action. Furthermore, the email contains a link to access live streaming from the ESP32-CAM module, allowing users to visually verify the situation remotely in real-time.

II. System Component Description

A smart door lock system includes a microcontroller, inputs, output and power supply.

A. Microcontroller and other Modules

a) ESP32

Figure 1 : ESP32 Board

The ESP32 is a powerful and versatile microcontroller and system-on-chip (SoC) developed by Espressif Systems. It has gained significant popularity in the embedded systems and IoT (Internet of Things) community due to its rich feature set, low cost, and wide range of applications.

Processor and Wi-Fi Module Version:

- The ESP32 typically features a dual-core Xtensa LX6 microprocessor clocked at up to 240 MHz, providing ample processing power for a wide range of applications.
- The Wi-Fi module integrated into the ESP32 supports 2.4 GHz Wi-Fi 802.11 b/g/n, allowing for wireless connectivity and communication with Wi-Fi networks and devices.

b) ESP32-CAM

Figure 2 : ESP32 CAM

The ESP32-CAM module is a compact development board based on the ESP32 microcontroller and designed specifically for camera applications. It integrates a camera module along with the ESP32 chip, making it a convenient solution for projects that require image capturing and processing capabilities.

Processor and Wi-Fi Module Version:

- The ESP32-CAM module features the same dual-core Xtensa LX6 microprocessor as the standard ESP32, providing processing power for image capture and processing tasks.
- The Wi-Fi module integrated into the ESP32-CAM supports 2.4 GHz Wi-Fi 802.11 b/g/n, enabling wireless connectivity and communication for camera-based projects.

Camera Module:

The ESP32-CAM features an OV2640 camera module, capable of capturing images with a resolution of up to 2 megapixels (1600x1200 pixels). This allows for the development of projects involving image capture, surveillance, object detection, and other computer vision applications.

B. Inputs

a) Keypad

A keypad is an input device commonly used in electronic systems to provide a user interface for entering data or commands. It consists of a matrix of keys, each representing a specific character, number, or function.

Figure 3: Keypad Module

Construction: Keypads typically consist of a grid of buttons arranged in rows and columns. Each button corresponds to a unique electrical contact, and pressing a button connects its corresponding row and column conductors, allowing the microcontroller or electronic system to detect the button press.

C. Outputs

a) LCD Display

A 16-pin LCD display typically refers to a liquid crystal display (LCD) module that utilizes a 16-pin interface for communication with a microcontroller or other electronic device. Here are some details about such LCD displays:

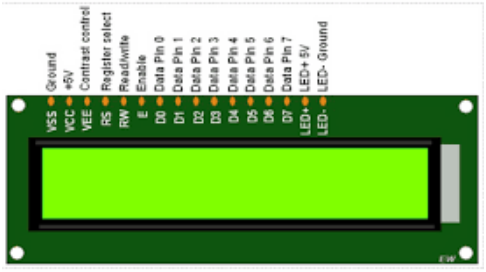

Figure 4 : LCD display

Construction: LCD displays consist of a liquid crystal layer sandwiched between two transparent electrodes and polarizing filters. The display also includes a backlight for illumination, which may be built-in or external.

16-Pin Interface: The 16-pin interface on the LCD module allows for communication with an external microcontroller or electronic system. These pins are typically used for power supply, data transmission, control signals, and backlight control.

D. Power Supply

Power supply considerations are crucial for ensuring the reliable operation of electronic components and systems. Here are some details about powering the last components of an electronic system, such as LCD displays, keypads, and microcontrollers:

Direct Power Supply

Most electronic components, including LCD displays, keypads, and microcontrollers, require a stable and regulated power supply voltage for proper operation.

A direct power supply connection can be established using various methods, including:

- **DC Power Adapter:** Connecting the components to a DC power adapter with the appropriate voltage and current rating.
- **Battery:** Powering the components using batteries, either rechargeable or non-rechargeable, depending on the application requirements.
- **Power Supply Module:** Using a dedicated power supply module or voltage regulator to provide the required voltage and current to the components.

III. SOFTWARE

A. *Arduino IDE*

The Arduino Integrated Development Environment (IDE) is a user-friendly software tool tailored for programming Arduino microcontroller boards. Offering a simplistic interface, it's accessible to beginners and professionals alike across Windows, macOS, and Linux platforms. Users can harness its simplified version of C and C++ languages, facilitated by features like syntax highlighting and auto-indentation. With an in-built code editor and library manager, developers can efficiently write, compile, and manage code for various Arduino board models. Debugging is made easier through its integrated serial monitor, aiding real-time data exchange between the Arduino and external devices.

Supported by a vast community, users have access to extensive documentation, forums, and tutorials. The IDE's extensibility allows for customization, enabling the addition of plugins and tools to meet specific project requirements. Overall, the Arduino IDE serves as a powerful yet accessible tool, empowering users to explore the world of electronics and bring their ideas to life.

IV. METHODOLOGY

A. Theory

To address the security concern, we opted for the ESP32 microcontroller due to its distinct advantages over other options. The ESP32 offers a plethora of I/O pins, along with integrated Bluetooth and Wi-Fi modules, providing seamless connectivity. In our hardware setup, we interfaced a keypad and an LCD display with the ESP32. Initially, the keypad facilitates the setup or reset of the password. Once configured, it serves as the means to input the password for door access. If an incorrect password is entered thrice consecutively, an automated email containing a link to live video streaming from the ESP32-CAM module is dispatched. This module, also Internet-enabled and powered, enhances security measures by offering real-time visual verification.

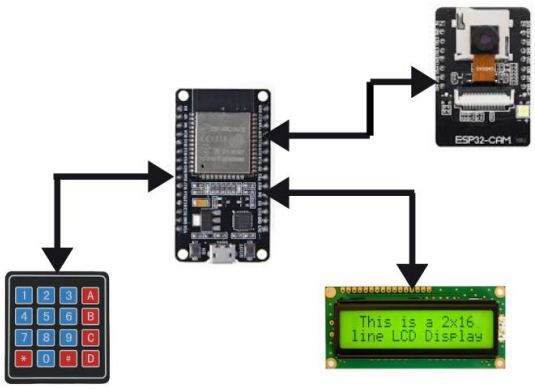

Figure 5 : Circuit Block diagram

This integrated system ensures that any unauthorized attempts to breach security trigger an immediate response. Homeowners are promptly alerted via email, empowering them to take swift action. Additionally, the live video feed from the ESP32-CAM provides concrete evidence of the security breach, enabling informed decision-making and potential intervention. In essence, our solution not only fortifies home security but also equips residents with the means to swiftly respond to any threats, thereby enhancing overall safety and peace of mind.

B. Flowchart

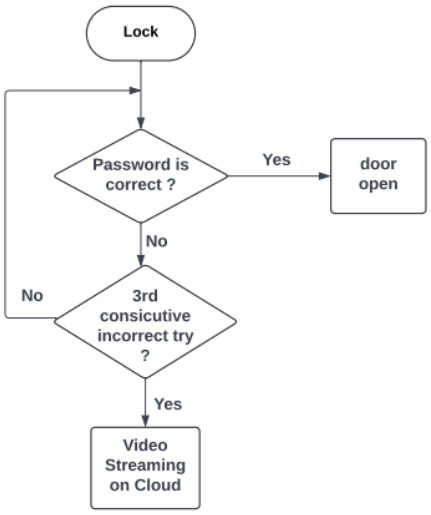

Figure 6 : Flowchart of the process

C. Results

The first time the system is turned on, users have access to set the password. After that, users only have the ability to enter the password to unlock the door. If a user wishes to change the password in the future, they must enter a specific permanent password for that purpose.

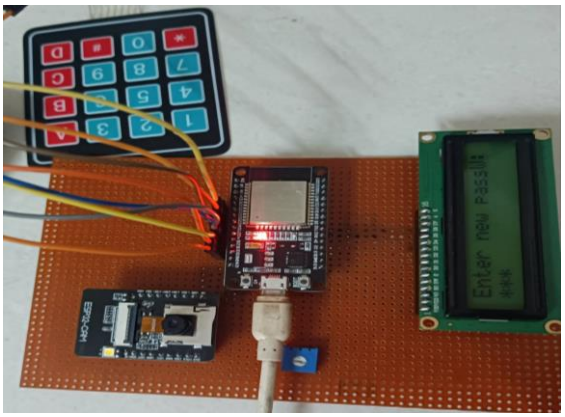

Figure 7 : When system is turned on for first time

After setting the password, the user will be asked to enter it, as shown in the image below.

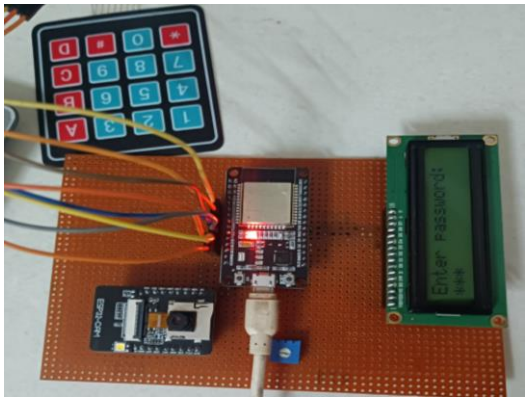

Figure 8 : When user is trying to unlock door

When the user enters the wrong password three consecutive times, they will receive a penalty of 5 seconds during which they will be unable to enter the password. Simultaneously, a security alert email containing a link to the camera streaming will be sent to the owner of the system.

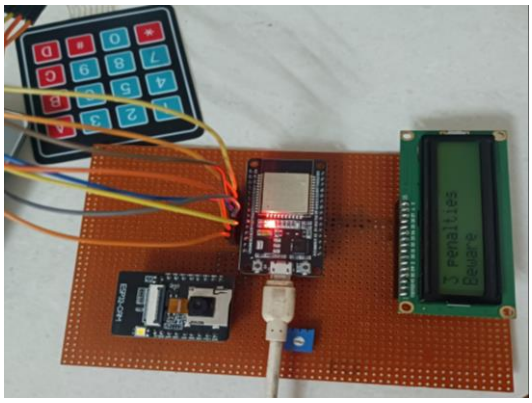

Figure 9 : When user have entered wrong password three times simultaneously

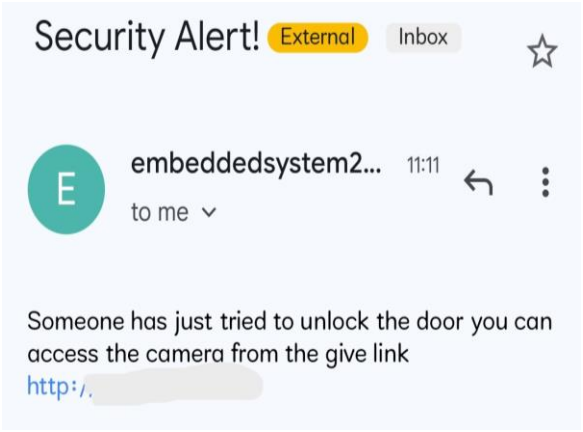

Figure 10 : mail which containing link of camera streaming

After clicking the link, the user will be directed to the video streaming interface of the ESP32-CAM module, where they can access live video from the same.

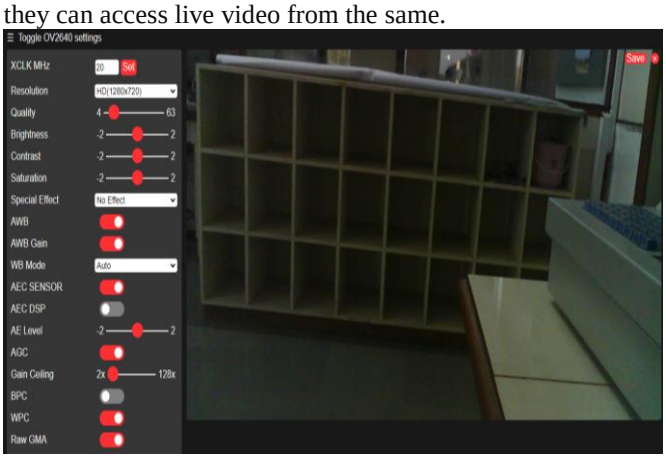

Figure 11 : Interface of the streaming link provided by ESP32 CAM module

V. Conclusions

In conclusion, the assignment presented a comprehensive solution for enhancing home security using the ESP32 microcontroller and associated peripherals. By leveraging the ESP32's advanced features such as abundant I/O pins, built-in Bluetooth and Wi-Fi modules, and integration capabilities, we devised a robust system to safeguard residential premises. The implementation involved interfacing a keypad and an LCD display with the ESP32, enabling password setup and access control. Furthermore, the system incorporated the ESP32-CAM module for live video streaming, providing visual confirmation in case of security breaches.

Through this assignment, we addressed the pressing need for proactive security measures in modern homes. By integrating intelligent technology and real-time monitoring, our solution empowers homeowners to promptly respond to unauthorized access attempts. The automated email alert system, triggered by consecutive incorrect password entries, ensures timely notification, enabling swift action to mitigate potential threats. Additionally, the inclusion of live video streaming adds an extra layer of security, providing tangible evidence and facilitating informed decision-making.

Overall, this assignment demonstrates the effectiveness of leveraging cutting-edge technology, such as the ESP32 microcontroller and associated modules, to enhance home security. By combining hardware interfacing, programming, and network connectivity, we have developed a comprehensive solution that not only fortifies security but also provides peace of mind to homeowners. As technology continues to evolve, solutions like these pave the way for safer and smarter homes in the digital age.

VI. APPENDIX

// ESP32 Code :

```
#include <LiquidCrystal.h>
#include <Keypad.h>
#include <EEPROM.h>
#include <EMailSender.h>
#include <WiFi.h>

const char* ssid = "wifi name";
const char* pass = "wifi password";

EMailSender emailSend("sender mail id", "App password");


// Door Lock variables
String password = "";   // the user defined password
String input = "";      // the user's input
bool unlocked = false;  // whether the door is unlocked or not
byte count = 0;         // represents no of elements entered for displaying each as '*' on LCD
byte false_counter = 0; // represents no of false attempts to alert by buzzer

// Define the address in EEPROM to store the password
int address = 0;

// Define the pins for the LCD and keypad
const byte ROWS = 4, COLS = 4; // 4*4 keypad
char keys[ROWS][COLS] = {
    {'1', '2', '3', 'A'},
    {'4', '5', '6', 'B'},
    {'7', '8', '9', 'C'},
    {'*', '0', '#', 'D'}};

byte rowPins[ROWS] = {13, 12, 14, 27}; // Digital pins
byte colPins[COLS] = {26, 25, 33, 32}; // Digital pins

// Initialize the LCD and keypad
LiquidCrystal lcd(15, 2, 4, 16, 17, 5);
Keypad keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

void setup()
{
  Serial.begin(115200);
  lcd.begin(16, 2);         // Set up the LCD
  keypad.setDebounceTime(50); // Set up the keypad
}

void loop()
{
  doorLock();
}

void sendmail()
{
  WiFi.begin(ssid, pass);
  WiFi.setSleep(false);

  while (WiFi.status() != WL_CONNECTED) {
```

```
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");

  EMailSender::EMailMessage message;
  message.subject = "Security Alert! ";
  message.message = "Someone has just tried to unlock the door you can access the camera from the give link\n (“ local ip of wifi which we got from cam module”)";

  EMailSender::Response resp = emailSend.send("receiver mail id", message);
}

void doorLock()
{
  char key = keypad.getKey();
  // If a key is pressed, then only add it to the input
  if (key)
  {
    if (password == "")
    {
      lcd.setCursor(0, 0);
      lcd.print("Enter new passW: ");
    }
    input += key;           // accumulate keys to input
    lcd.setCursor(count, 1); // move cursor to show each new char as secret ‘*’
    lcd.print("*");
    count++; // increment data array by 1 to store new char, also keep track of the number of chars entered
    if (key == '#')
    { // to clear input
      input = "";
      count = 0;
      if (password == "")
      {
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Enter new passW: ");
      }
      else
      {
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Enter password: ");
      }
      lcd.setCursor(0, 1);
      //key = keypad.getKey();
    }
  }
  // Check if the input is complete
  if (input.length() == 4)
  { // change this 4 to different values
    count = 0;
    // If the password has not been set yet, store the input as the new password
    if(input == "****")
    {
      input="";
```

```
    password="";
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Enter new passW: ");
    lcd.setCursor(0,1);
   }
   else if (password == "")
   {
    password = input;
    storePassword(password); // in EEPROM
    lcd.clear();
    lcd.print("Password set");
    delay(1000);
    lcd.clear();
    lcd.print("Enter password:");
    input = "";
   }
   // Otherwise, directly check if the input matches the password
   else
   {
    // If the input matches, unlock the door
    if (input == password)
    {
     unlocked = true;
     false_counter = 0;
     lcd.clear();
     lcd.print("Door unlocked");
     delay(2000);
     lcd.clear();
     lcd.print("Enter password:");
     input = ""; // make input null string again to retake input
    }
    // Otherwise, display an error message and reset the input
    else
    {
     false_counter++;
     lcd.clear();
     lcd.print("Incorrect");
     delay(1000);
     if (false_counter == 3)
     {
      lcd.clear();
      lcd.print("3 penalties");
      lcd.setCursor(0, 1);
      lcd.print("Beware");
      delay(5000);
      false_counter = 0; // reset false attempts
      sendmail();
     }
     lcd.clear();
     lcd.print("Enter password:");
     input = ""; // make input null string again to retake input
    }
    count = 0;
   }
  }
 // If the door is unlocked, display a message
```

```
  if (unlocked)
  {
    lcd.clear();
    lcd.print("Welcome!");
    delay(2000);
    lcd.clear();
    lcd.print("Enter password:");
    unlocked = false;
  }
}

// Function to store the password in EEPROM
void storePassword(String password)
{
  for (int i = 0; i < 4; i++)
  { // password length is 4
    EEPROM.write(address + i, password[i]);
  }
}
```

// ESP32 CAM Code :

```
#include "esp_camera.h"
#include <WiFi.h>

#define CAMERA_MODEL_AI_THINKER
#include "camera_pins.h"

const char* ssid = "wifi name";
const char* password = "wifi pass";

void startCameraServer();
void setupLedFlash(int pin);

void setup() {
  Serial.begin(115200);
  Serial.setDebugOutput(true);
  Serial.println();

  camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
  config.pin_d6 = Y8_GPIO_NUM;
  config.pin_d7 = Y9_GPIO_NUM;
```

```
  config.pin_xclk = XCLK_GPIO_NUM;
  config.pin_pclk = PCLK_GPIO_NUM;
  config.pin_vsync = VSYNC_GPIO_NUM;
  config.pin_href = HREF_GPIO_NUM;
  config.pin_sccb_sda = SIOD_GPIO_NUM;
  config.pin_sccb_scl = SIOC_GPIO_NUM;
  config.pin_pwdn = PWDN_GPIO_NUM;
  config.pin_reset = RESET_GPIO_NUM;
  config.xclk_freq_hz = 20000000;
  config.frame_size = FRAMESIZE_UXGA;
  config.pixel_format = PIXFORMAT_JPEG; // for streaming
   config.grab_mode = CAMERA_GRAB_WHEN_EMPTY;
  config.fb_location = CAMERA_FB_IN_PSRAM;
  config.jpeg_quality = 12;
  config.fb_count = 1;

  if(config.pixel_format == PIXFORMAT_JPEG){
    if(psramFound()){
      config.jpeg_quality = 10;
      config.fb_count = 2;
      config.grab_mode = CAMERA_GRAB_LATEST;
    } else {
      // Limit the frame size when PSRAM is not available
      config.frame_size = FRAMESIZE_SVGA;
      config.fb_location = CAMERA_FB_IN_DRAM;
    }
  } else {
    // Best option for face detection/recognition
    config.frame_size = FRAMESIZE_240X240;
#if CONFIG_IDF_TARGET_ESP32S3
    config.fb_count = 2;
#endif
  }

#if defined(CAMERA_MODEL_ESP_EYE)
  pinMode(13, INPUT_PULLUP);
  pinMode(14, INPUT_PULLUP);
#endif

  // camera init
  esp_err_t err = esp_camera_init(&config);
  if (err != ESP_OK) {
    Serial.printf("Camera init failed with error 0x%x", err);
    return;
  }
```

```
  sensor_t * s = esp_camera_sensor_get();
  // initial sensors are flipped vertically and colors are a bit saturated
  if (s->id.PID == OV3660_PID) {
    s->set_vflip(s, 1); // flip it back
    s->set_brightness(s, 1); // up the brightness just a bit
    s->set_saturation(s, -2); // lower the saturation
  }
  // drop down frame size for higher initial frame rate
  if(config.pixel_format == PIXFORMAT_JPEG){
    s->set_framesize(s, FRAMESIZE_QVGA);
  }

#if defined(CAMERA_MODEL_M5STACK_WIDE) || defined(CAMERA_MODEL_M5STACK_ESP32CAM)
  s->set_vflip(s, 1);
  s->set_hmirror(s, 1);
#endif

#if defined(CAMERA_MODEL_ESP32S3_EYE)
  s->set_vflip(s, 1);
#endif

// Setup LED FLash if LED pin is defined in camera_pins.h
#if defined(LED_GPIO_NUM)
  setupLedFlash(LED_GPIO_NUM);
#endif

  WiFi.begin(ssid, password);
  WiFi.setSleep(false);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  startCameraServer();
  Serial.print("Camera Ready! Use 'http://");
  Serial.print(WiFi.localIP());
  Serial.println("' to connect");
}

void loop() {
  // Do nothing. Everything is done in another task by the web server
  delay(10000);
}
```