AUTOMATIC PILL DISPENSER

DEPARTMENT OF ELECTRONICS ENGINEERING

MADRAS INSTITUTE OF TECHNOLOGY

ANNA UNIVERSITY: CHENNAI - 600 044

EC5512-SUMMER INTERNSHIP/SUMMER PROJECT

Submitted by

KEERTHANA 2022504024

YUVARAJ 2022504554

ABISHEK 2022504552

AUG 24-DEC 24

BONAFIDE CERTIFICATE

Certified that this project report "AUTOMATIC PILL DISPENSER" is the bonafide work of

KEERTHANA 2022504024

YUVARAJ 2022504554

ABISHEK 2022504552

Who carried out the project under my supervision.

SIGNATURE

Dr. K.MARIAMMAL

SUPERVISOR

Associate Professor

Department of Electronics Engineering,

Madras Institute of Technology,

Anna University, Chennai-44

ACKNOWLEDGEMENT

With profound gratitude and due regards, I sincerely acknowledge with thanks the opportunity provided to us by our respectful, Dean Dr.K.RAVICHANDRAN, Madras Institute of Technology, Anna University for providing a good environment and facilities.

I thank our respectful Head of the Department, Dr.D.MEGANATHAN, Professor and Head of Electronics Engineering, Madras Institute of Technology, Anna University for his encouragement during the Project work.

I sincerely express my gratitude to our faculty incharge, Dr. K.MARIAMMAL ,Associate Professor, Department of Electronics Engineering, for being an inspiration to us and for her valuable guidance throughout the course of this project.

Place: Chennai

Date: 15/11/2024

TABLE OF CONTENTS

ABSTRACT

The automatic pill dispenser project is designed to enhance medication management by ensuring timely, accurate pill dispensing and effective reminders. This system aims to support individuals who require regular medication, particularly those with complex regimens or memory challenges. At the core of the device is the Arduino R4 Wi-Fi microcontroller, which manages all operations and coordinates precise pill dispensing via an SG90 servo motor. The 1602 LCD screen serves as a user-friendly interface, displaying key information such as the next scheduled dosage and device status. An integrated buzzer provides audible alerts to remind users when it is time to take their medication, ensuring that doses are not missed due to distractions.

A standout feature of this project is its integration of email notifications. Leveraging the Wi-Fi capability of the Arduino R4, the system is programmed to send timely email reminders to users, ensuring that they are aware of their medication schedules even when away from the device. This multi-channel reminder system enhances adherence by combining physical and digital alerts for greater reliability.

The dispenser is designed with ease of use and flexibility in mind, allowing users or caregivers to set and adjust medication schedules according to individual needs. The SG90 servo motor ensures controlled and precise dispensing. By automating the process and providing notifications, this project aims to reduce the risks associated with medication non-adherence, such as missed or incorrect doses, which can compromise health.

Ultimately, this automatic pill dispenser enhances patient safety and adherence through the integration of mechanical precision and real-time communication, offering a reliable, scalable solution for modern medication management.

CHAPTER-1

1.1 INTRODUCTION

The automatic pill dispenser project is designed to simplify and improve medication adherence for individuals who need consistent dosing. By integrating an Arduino R4 Wi-Fi microcontroller, an SG90 servo motor, a 1602 LCD screen, and a buzzer, this device ensures that pills are dispensed accurately and on time. Additionally, it leverages Wi-Fi connectivity to send reminder emails to users, enhancing the reliability of medication schedules. This project combines precise mechanical control with digital communication, aiming to reduce the risk of missed or incorrect doses and promote better health outcomes.

1.2 PROJECT AIM

The aim of this project is to develop an automatic pill dispenser that ensures timely and accurate dispensing of medication while enhancing adherence through multiple reminder systems. By utilizing an Arduino R4 Wi-Fi microcontroller, an SG90 servo motor, a 1602 LCD screen, and an alerting buzzer, the device aims to automate the dispensing process and provide visual and audible reminders. Additionally, the integration of email notifications aims to keep users informed of their medication schedules even when they are away from the device. The overall goal is to improve user safety, reduce missed or incorrect doses, and provide a convenient, reliable solution for medication management.

1.3 PROJECT OBJECTIVE

The objective of this project is to develop an automatic pill dispenser that ensures accurate, timely medication dispensing while enhancing adherence through multiple reminder systems. By utilizing an SG90 servo motor controlled by an Arduino R4 Wi-Fi microcontroller, the device will dispense pills at scheduled times, with a 1602 LCD screen providing clear user information. It will also incorporate a buzzer for

audible alerts and send email notifications through the Wi-Fi capability of the Arduino R4 to remind users when it is time to take their medication. The system aims to improve user safety, reduce the risk of missed or incorrect doses, and offer a reliable, customizable solution for managing medication schedules.

1.4 PROJECT SCOPE

The scope of this project encompasses the design and development of an automatic pill dispenser aimed at improving medication adherence for individuals with complex regimens. It involves the integration of an SG90 servo motor, Arduino R4 Wi-Fi microcontroller, 1602 LCD screen, and a buzzer to automate the pill dispensing process, display relevant information, and provide auditory alerts. Additionally, the project includes the implementation of email notification functionality through the Wi-Fi capability of the Arduino, allowing users to receive timely reminders. The project will focus on creating a customizable and user-friendly system that ensures accurate dosing, reduces the risk of medication errors, and provides enhanced monitoring for users

1.5 DESCRIPTION OF THE PROJECT

The automatic pill dispenser is designed to help individuals adhere to their medication schedules by ensuring accurate and timely pill dispensing. This system is particularly beneficial for people with complex medication regimens, memory issues, or those who may forget to take their medication. At the core of the project is the Arduino R4 Wi-Fi microcontroller, which controls the entire operation, including the pill dispensing mechanism, timing, and communication functions. The SG90 servo motor is used to control the release of pills at precise intervals, while the 1602 LCD screen displays relevant information such as the next scheduled dose, remaining pills, and system status.

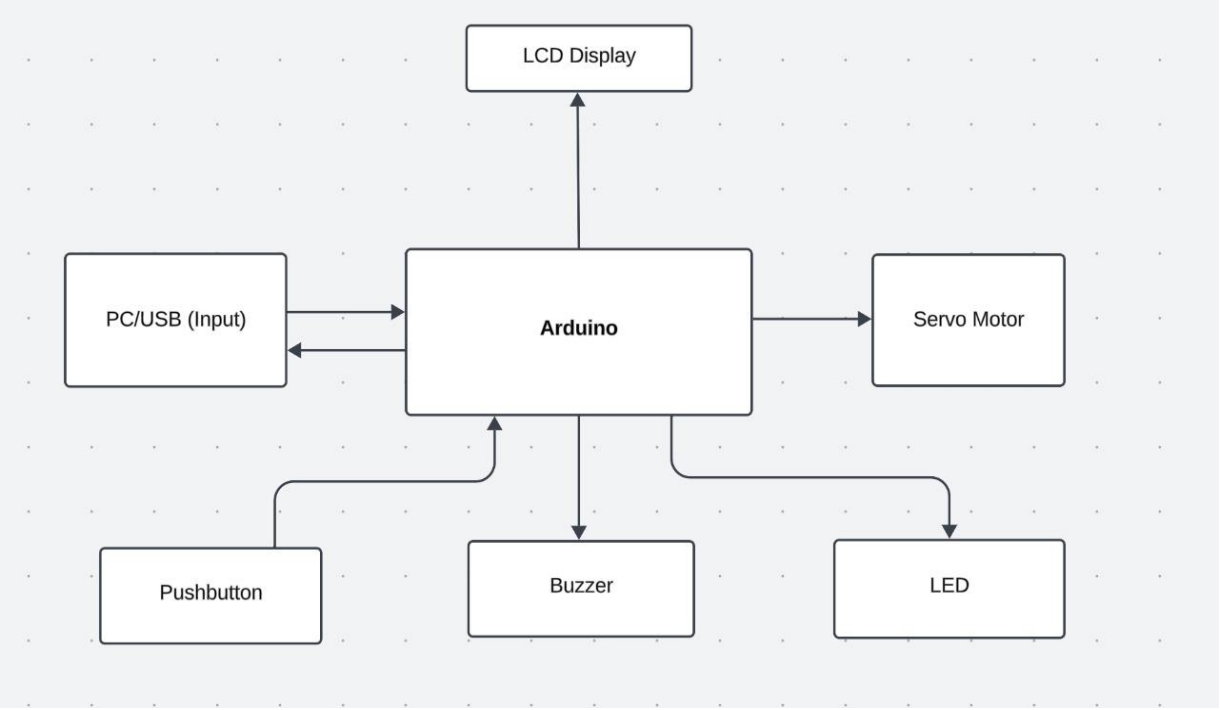

Figure 1.5(a). Block Diagram of Pill Dispenser

In addition to automates dispensing, the system features a buzzer that alerts the user when it is time to take their medication. The Wi-Fi connectivity of the Arduino R4 also enables the device to send email notifications to users or caregivers, providing timely reminders even if the user is away from the dispenser. This ensures that users never miss a dose, even in cases of distraction or forgetfulness. The project is designed to be customizable, allowing users to easily set their medication schedules. By automating the process and offering multi-channel reminders, the system aims to enhance medication adherence, improve safety, and provide a reliable solution for managing medication.

CHAPTER-2

2.1 HARDWARE REQUIREMENTS

1. **Arduino-compatible microcontroller board:** Serves as the main control unit for managing the entire system's operations, including timing, motor control, and notifications.

2. **USB cable:** Used to connect the microcontroller board to a computer for programming and power supply.

3. **Breadboard:** Provides a platform for building and testing the circuit without soldering, allowing easy adjustments.

4. **Jumper wires:** Facilitate connections between the microcontroller, breadboard, and various components for seamless circuit integration.

5. **Potentiometer:** Adjusts the contrast of the 1602 LCD screen to ensure clear readability of displayed information.

6. **LED:** Used as an indicator for system status or notifications in the circuit.

7. **Current-limiting resistors (470 Ω for UNO R4):** Protect the LED by limiting current flow, preventing damage to both the LED and microcontroller.

8. **Pushbutton:** Allows user interaction for functions like system reset or manual activation of the dispenser.

9. **10 kΩ resistor:** Pull-down resistor connected to the pushbutton to ensure stable input signals and prevent floating states.

10. **SG90 servo motor:** Controls the mechanical movement of the pill dispensing mechanism, providing precise rotation for accurate pill release.

11. **1602 LCD screen:** Displays real-time information such as dosage schedule, current status, and remaining pills.

12. **Active buzzer:** Emits sound alerts to notify the user when it is time to take their medication.

13. USB port: Provides a power source for the microcontroller board when connected to a computer or power adapter.

14. Additional tools and supplies:

i. Cardboard
ii. Cardstock
iii. Popsicle sticks
iv. Scissors
v. Hot glue gun
vi. Candy (to use for demonstration instead of real medication)

2.2 SOFTWARE REQUIREMENTS

1. Servo Library:

Provides essential functions for controlling the SG90 servo motor's movement, enabling precise operation of the pill dispensing mechanism. This library ensures smooth angular positioning and responsiveness. It is a core component for automating the mechanical aspect of the dispenser.

2. Liquid Crystal Library:

Facilitates communication between the Arduino and the 1602 LCD screen, allowing for text to be displayed effectively. It helps create a user-friendly interface by displaying information like the next scheduled dose and system status. The library supports customizable messages to enhance user interaction.

3. RTC Library:

Adds real-time clock functionality to the system, which is crucial for maintaining accurate timing for medication schedules. This library ensures the system operates independently of power interruptions by keeping track of time. It helps schedule precise pill dispensing at pre-set intervals.

4. Wi-FiS3 Library:

Manages Wi-Fi connectivity on the Arduino R4, allowing the system to connect to local networks. It is essential for enabling internet-based functions, such as sending email reminders. The library provides stable and secure network communication for real-time notifications.

5. ESP Mail Client Library:

Supports the sending of emails through SMTP protocols from the Arduino, making it a key component for sending notifications. This library simplifies coding efforts for handling email communication, enabling integration with various mail servers. It ensures users receive timely email reminders about their medication schedules.

6. Mail Account:

Acts as the sender address for dispatching reminder emails to the user or caregiver. The mail account needs to be properly configured and authenticated for secure communication. It plays a critical role in ensuring that notifications reach the intended recipient.

7. Arduino IDE:

A software platform used for coding, compiling, and uploading programs to the Arduino board. It provides the tools and libraries needed to develop and troubleshoot the project code. The IDE is user-friendly and supports multiple programming languages, including C/C++.

8. Google App Password:

A secure, one-time password generated from a Google account that allows the Arduino to access email services. It ensures that the Arduino can send emails through a secure connection without exposing the main account password. This step is crucial for maintaining the security of the email service used.

9. Wi-Fi:

Provides necessary internet connectivity for the system to communicate and send emails. It is essential for the real-time functionality of the project, ensuring that reminders are sent without delays. Reliable Wi-Fi access enables consistent and seamless operation of the notification system.

10. SMTP (Simple Mail Transfer Protocol):

SMTP (Simple Mail Transfer Protocol) is a protocol used for sending emails across networks, typically used by email clients (such as Outlook, Thunderbird, etc.) or web applications (like Gmail) to send email messages. When you want to send an email from an embedded system, SMTP server can be used to handle the email transmission.

2.3 DESCRIPTION OF HARDWARE USED

Arduino R4 Wi-Fi Microcontroller Board:

This is the central control unit for the entire system. It is responsible for executing the programmed code, managing the timing of pill dispensing, controlling the servo motor, and handling Wi-Fi connectivity for sending email notifications. Its Wi-Fi capability allows the device to communicate over the internet, making it possible to send real-time reminders and updates.

SG90 Servo Motor:

The SG90 is a small, lightweight servo motor that provides precise control over movement. In this project, it is used to actuate the pill dispensing mechanism, ensuring that pills are released accurately at the scheduled times. The servo motor's ability to rotate to specific angles makes it an ideal choice for controlled dispensing. Its

specifications are Operating Voltage is +5V typically, Torque: 2.5kg/cm, Operating speed is 0.1s/60°, Gear Type: Plastic, Rotation: 0°-180°, Weight of motor: 9gm, Package includes gear horns and screws.

1602 LCD Screen:

The 1602 LCD screen is used to display key information to the user, such as the current status of the device, the time for the next dose, and the number of pills remaining. It provides an easy-to-read interface that enhances user interaction with the system. The display is controlled using the Liquid Crystal library, which allows for simple integration with the Arduino.

Active Buzzer:

The active buzzer emits a sound alert when it is time for the user to take their medication. This audible signal ensures that users are reminded even if they are not looking at the device. The buzzer helps reduce the chances of missed doses due to distractions.

Potentiometer:

The potentiometer is used to adjust the contrast of the 1602 LCD screen. By fine-tuning the screen's contrast, the display can be made clearer and more legible, enhancing the overall user experience.

Pushbutton:

The pushbutton allows the user to interact with the system for manual operations, such as resetting the device or triggering a test dispensing. It provides a simple method for user control over the device. The pushbutton is connected to the Arduino and works in conjunction with a 10kΩ resistor to ensure stable input signals.

Wi-Fi Module:

The Wi-Fi module in the Arduino R4 is essential for connecting the system to the internet. This enables the pill dispenser to send email notifications via the SMTP server. The Wi-Fi connection is vital for the remote reminder functionality, ensuring that users or caregivers receive alerts no matter where they are.

2.4 PIN DESCRIPTION

1602 LCD display:

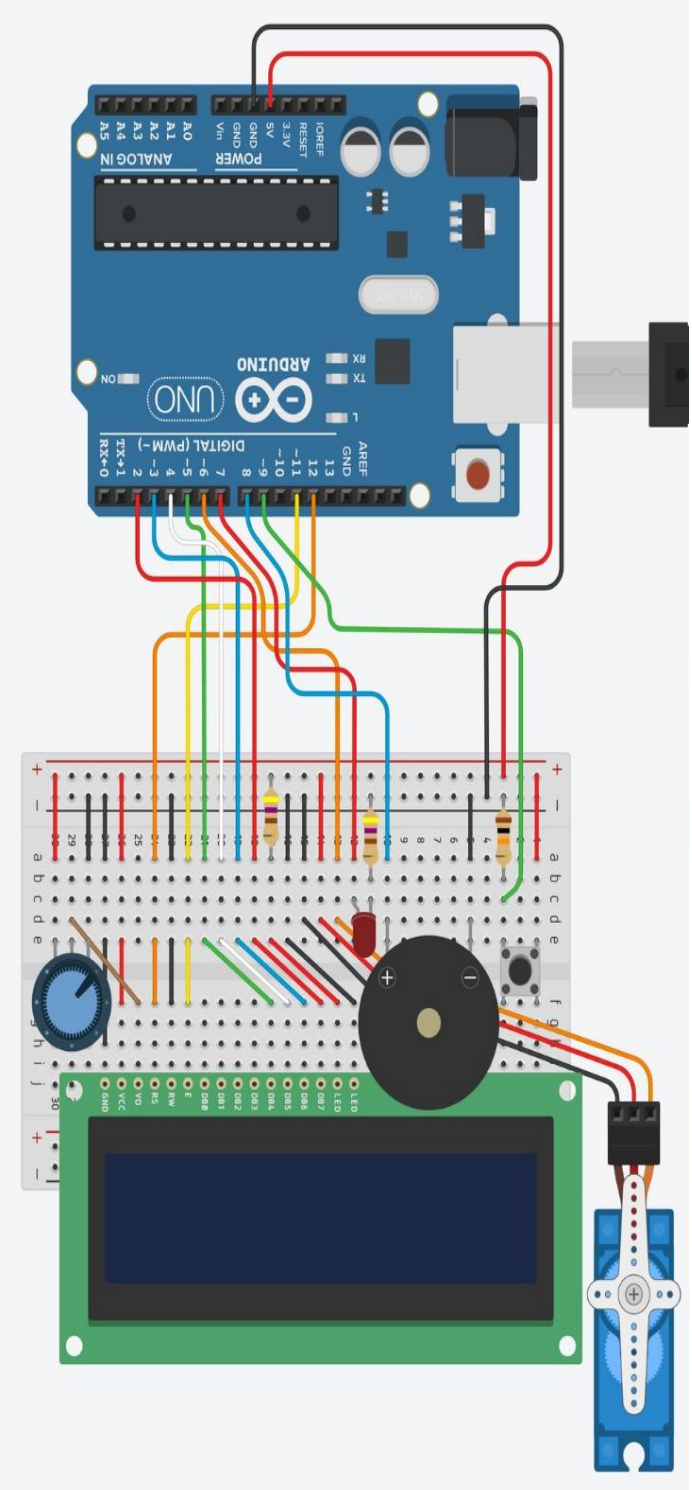

Figure 2.4(a). Image of the circuit

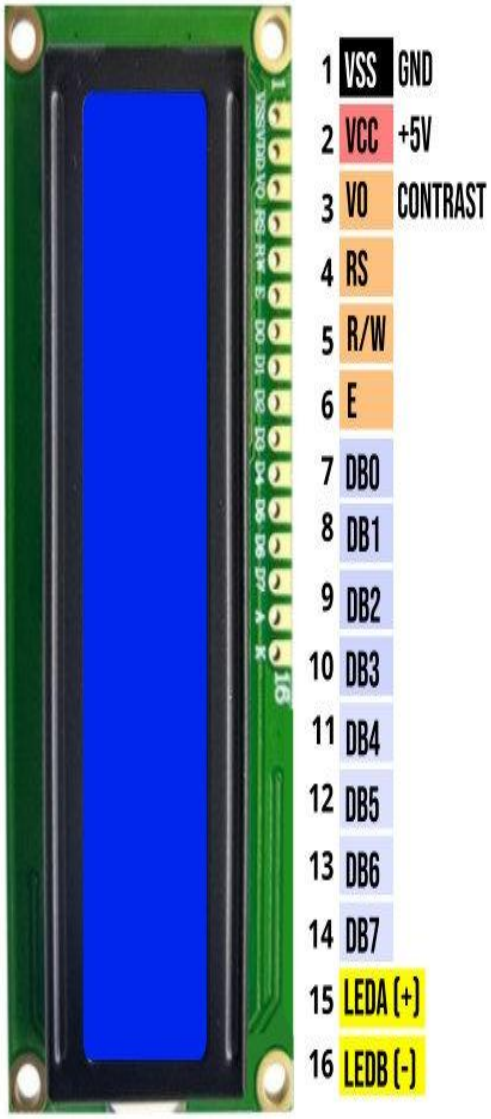

Figure 2.4(b).1602 Display

PIN Number	PIN Name	Description
1	Ground/Source Pin	This is a GND pin of display, used to connect the GND terminal of the microcontroller unit or power source.
2	VCC/Source Pin	This is the voltage supply pin of the display, used to connect the supply pin of the power source.
3	V0/VEE/Control Pin	This pin regulates the difference of the display, used to connect a changeable POT that can supply 0 to 5V.
4	Register Select/Control Pin	This pin toggles among command or data register, used to connect a microcontroller unit pin and obtains either 0 or 1(0 = data mode, and 1 = command mode).
5	Read/Write/Control Pin	This pin toggles the display among the read or writes operation, and it is connected to a microcontroller unit pin to get either 0 or 1 (0 = Write Operation, and 1 = Read Operation).
6	Enable/Control Pin	This pin should be held high to execute Read/Write process, and it is connected to the microcontroller unit & constantly held high.
7-14	Data Pins	These pins are used to send data to the display. These pins are connected in two-wire modes like 4-wire mode and 8-wire mode. In 4-wire mode, only four pins are connected to the microcontroller unit like 0 to 3, whereas in 8-wire mode, 8-pins are connected to microcontroller unit like 0 to 7.
15	+ve pin of the LED	This pin is connected to +5V
16	-ve pin of the LED	This pin is connected to GND.

SG90Servo motor:

Wire number	Wire colour	Description
1	Brown	Ground wire connected to the ground of system.
2	Red	Powers the motor typically +5V is used.
3	Orange	PWM signal is given in through this wire to drive the motor

CHAPTER-3

3.1 PRINCIPLE

The principle behind the automatic pill dispenser is to automate the medication management process, ensuring timely and accurate dispensing of pills to the user based on a pre-programmed schedule. The system operates by utilizing an Arduino R4 Wi-Fi microcontroller, which serves as the central control unit, responsible for managing the dispensing mechanism, scheduling, and communication features. The SG90 servo motor is used to control the movement of the dispensing mechanism, allowing the precise release of pills at scheduled times. The timing of these operations is set by the user and programmed into the system.

To alert the user when it's time to take their medication, the device incorporates an audible buzzer that sounds at the designated times. Additionally, the 1602 LCD screen provides real-time feedback and displays important information such as the current dose time, number of pills remaining, and device status, allowing for easy monitoring and

control. To further ensure adherence, the system leverages the Wi-Fi capability of the Arduino to send email notifications to the user or caregiver, reminding them to take their medication. This multi-channel approach—combining the mechanical dispensing process, visual feedback, auditory alerts, and digital notifications—ensures that the user receives timely reminders and can manage their medication schedule effectively, minimizing the risk of missed or incorrect doses.

3.2 DESIGN AND IMPLEMENTATION

The automatic pill dispenser is designed around the Arduino R4 Wi-Fi microcontroller, which acts as the central control unit. The microcontroller is connected to various hardware components: the SG90 servo motor for dispensing pills, the 1602 LCD screen for displaying medication information, and the active buzzer for alerting the user. The RTC library is used to maintain accurate time for scheduling pill dispensing. The Arduino is programmed to activate the servo motor at specific times based on the user's medication schedule.

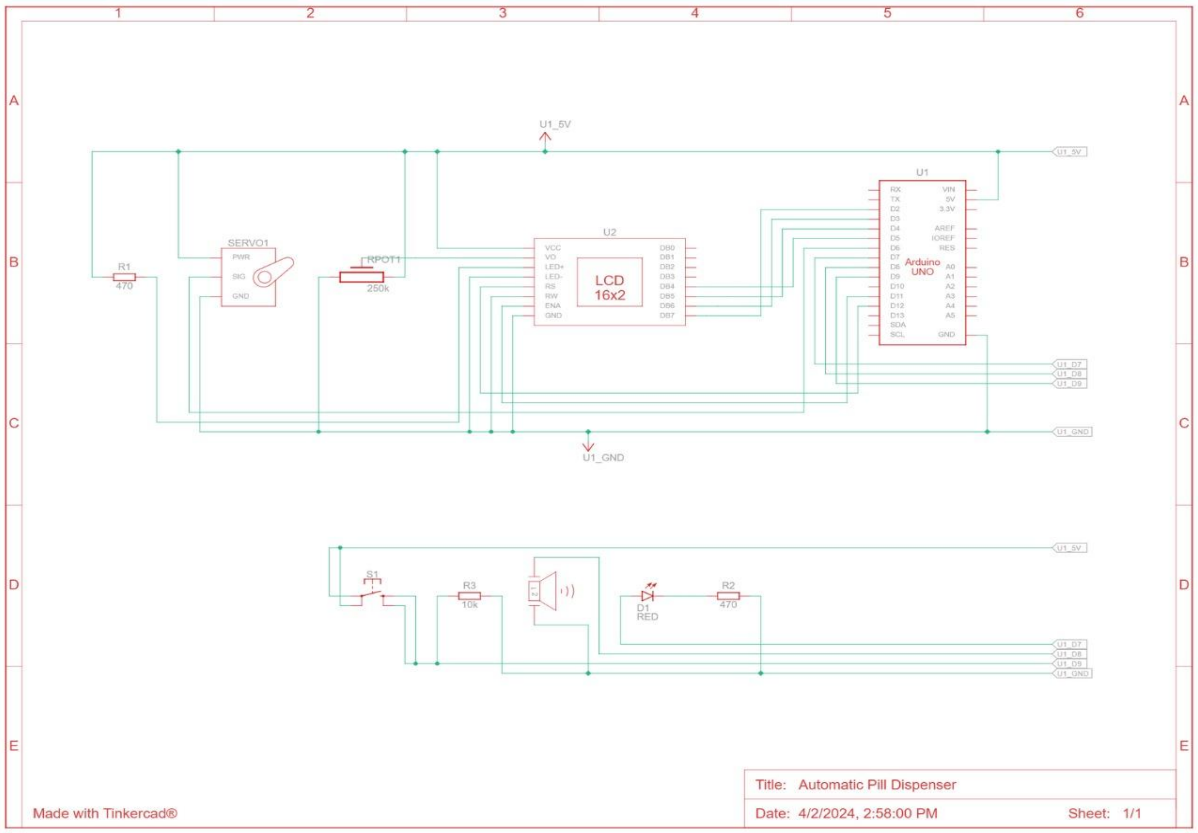

Figure 3.2(a). Circuit diagram

The LCD screen shows the upcoming medication time, the number of pills remaining, and system status, while the buzzer alerts the user at the scheduled dispensing times. Additionally, a pushbutton is incorporated for manual control, allowing the user to trigger pill dispensing or reset the system. The device is powered via a USB cable and assembled on a breadboard for testing and prototyping, with jumper wires connecting the various components.

For software implementation, the Arduino is programmed using the Arduino IDE. The system uses the Wi-FiS3 library to connect to a local network and enables the ESP Mail Client library to send email notifications via an SMTP server (such as Gmail). This feature provides timely reminders to the user or a caregiver about when to take medication. The system's software includes code for controlling the servo motor, managing the LCD display, and sending email notifications when it's time for medication. Below is a basic code snippet for the key functionalities,

```
// include libraries
#include <Servo.h>
#include <LiquidCrystal.h>
#include <RTC.h>
#include <WiFiS3.h>
#include <ESP_Mail_Client.h>

#define WIFI_SSID "POCO X6 5G" // CHANGE IT
#define WIFI_PASSWORD "keer2005" // CHANGE IT

// the sender email credentials
#define SENDER_EMAIL "umaakeeru@gmail.com" // CHANGE IT
#define SENDER_PASSWORD "mjxa mbis gvfd jpne"  // CHANGE IT to your Google
App password

#define RECIPIENT_EMAIL "umaakeeru@gmail.com" // CHANGE IT

#define SMTP_HOST "smtp.gmail.com"
#define SMTP_PORT 587

SMTPSession smtp;

Servo servo; // create servo object

// constant variables for pins
const int servoPin = 6;
const int buttonPin = 9;
const int ledPin = 7;
const int buzzerPin = 8;

// other variables
int angle = 0;
```

```
int angleIncrement = 45;  // default 45 degrees for 4 compartments, change
for different number of compartments
int newAngle;
int buttonState;
int movementDelay = 50;
int debounceDelay = 1000;

// time variables
int year;
int month;
int day;
int hour;
int minutes;
int seconds;

// initialize the library by associating any needed LCD interface pin
// with the arduino pin number it is connected to
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup(){ // setup code that only runs once
  pinMode(buttonPin, INPUT); // set button pin as input
  pinMode(ledPin,OUTPUT);    // set LED pin as output
  pinMode(buzzerPin,OUTPUT); // set buzzer pin as output
  digitalWrite(ledPin,LOW);  // make sure LED is off
  digitalWrite(buzzerPin,LOW);  // make sure buzzer is off
  servo.attach(servoPin);    // attach servo object to servoPin
  servo.write(angle);        // set servo to initial angle
  Serial.begin(9600);        // initialize serial for debugging

  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);

  // initialize the RTC and set the current date and time
  // you will need to manually adjust the time right before you upload the
code
  RTC.begin();
  RTCTime    startTime(14,    Month::NOVEMBER,    2024,    8,    59,    50,
DayOfWeek::THURSDAY, SaveLight::SAVING_TIME_ACTIVE);
  RTC.setTime(startTime);
}

void loop(){  // code that loops forever
  updateLCD();   // display the current date and time on the screen (see
function below)

  buttonState = digitalRead(buttonPin); // read button state (you can edit
the code to advance the servo when you press the button, useful for debugging)
  Serial.println(angle);                // print the servo angle

  // detect certain times and rotate the servo mechanism ahead by one
compartment.
  // the example code rotates the servo once per minute when the seconds
variable equals 0.
  // change to detect different times of day, for example (hours == 9 &&
minutes == 0 && seconds == 0) would detect 9:00:00 AM
  // use additional "else if" conditions to detect more than one time

  if(seconds == 0){                  // check for seconds = 0 (one minute
intervals)
    newAngle = angle + angleIncrement; // increase angle by increment
```

```
    if (newAngle<=180){           // if the new angle is less than or equal
to 180, increase angle
      while(angle < newAngle){    // increase angle until it reaches the new
angle
            angle = angle + 1;        // increase angle by 1
            servo.write(angle);       // move the servo
        Serial.println(angle);    // print the angle
            delay(movementDelay);     // delay to slow down movement
      }
      // flash LED and buzzer
      flashLED(4,150);        //  flashLED(number  of  flashes,  delay  in
milliseconds), see function below
    }
    else{ // if the new angle is greater than 180, reset angle to 0
      while(angle>0){         // decrease angle until it reaches 0
        angle = angle - 1;    // decrease angle by 1
        servo.write(angle);   // move the servo
        Serial.println(angle);// print the angle
        delay(movementDelay); // delay to slow down movement
      }
    }
  }
}
void updateLCD(){    // function to update LCD screen
  // get current time from the RTC
  RTCTime currentTime;
  RTC.getTime(currentTime);
  // store current time variables
  year = currentTime.getYear();
  month = Month2int(currentTime.getMonth());
  day = currentTime.getDayOfMonth();
  hour = currentTime.getHour();
  minutes = currentTime.getMinutes();
  seconds = currentTime.getSeconds();

  // set the cursor to column 0, line 1
  // (note: line 1 is the second row, since counting begins with 0):
  lcd.setCursor(0, 0);
  // print month/day/year (rearrange for different date formats)
  // add leading spaces if month or day are less than 10 to keep spacing
consistent
  // (always use 2 characters for month and day)
  if(month<10){   // print extra space if month is less than 10
    lcd.print(" ");
  }
  lcd.print(month);  // print the month
  lcd.print("/");
  if(day<10){        // print extra space if the day is less than 10
    lcd.print(" ");
  }
  lcd.print(day);    // print the day
  lcd.print("/");
  lcd.print(year);   // print the year

  // print time in hh:mm:ss format
  // add leading zeroes if minutes or seconds are less than ten to keep
spacing consistent
  lcd.setCursor(0, 1);  // move cursor to second row
  if(hour<10){          // print leading space if hour is less than 10
    lcd.print(" ");
  }
```

```
  lcd.print(hour);      // print the hour
  lcd.print(":");
  if(minutes<10){       // print leading zero if minute is less than 10
    lcd.print("0");
  }
  lcd.print(minutes);   // print the minute
  lcd.print(":");
  if(seconds<10){       // print leading zero if second is less than 10
    lcd.print("0");
  }
  lcd.print(seconds);   // print the second
}

void flashLED(int numFlashes, int flashDelay){  // alarm function to flash
LED and sound buzzer
  lcd.clear();              // clear the LCD screen
  lcd.setCursor(0, 0);      // set cursor to top left
  lcd.print("Take medicine!");  // display message
  for (int i = 0; i<numFlashes; i++){  // loop to flash LED/buzzer numFlashes
times
    digitalWrite(ledPin,HIGH);         // turn LED on
    digitalWrite(buzzerPin,HIGH);      // turn buzzer on
    delay(flashDelay);                 // delay
    digitalWrite(ledPin,LOW);          // turn LED off
    digitalWrite(buzzerPin,LOW);       // turn buzzer off
    delay(flashDelay);                 // delay
  }
  Serial.begin(9600);
  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();

  String subject = "Email Notification from Arduino";


  String textMsg = "This is an email sent from Arduino.\n";
  textMsg += "Hey there take your pill! ";
  textMsg += "Also don't foreget to off the alarm"; // OR replace this value
read from a sensor
  gmail_send(subject, textMsg);

  // wait for button press - the code will get stuck in this loop until you
press the button
  while(digitalRead(buttonPin) == LOW){
    digitalWrite(ledPin,HIGH);         // turn LED on
    digitalWrite(buzzerPin,HIGH);      // turn buzzer on
    delay(flashDelay);                 // delay
    digitalWrite(ledPin,LOW);          // turn LED off
    digitalWrite(buzzerPin,LOW);       // turn buzzer off
    delay(flashDelay);
  };
  delay(1000);    // delay before clearing screen
```

```
  lcd.clear();    // clear screen
}

void gmail_send(String subject, String textMsg) {
  // set the network reconnection option
  MailClient.networkReconnect(true);

  smtp.debug(1);

  smtp.callback(smtpCallback);
  Session_Config config;

  // set the session config
  config.server.host_name = SMTP_HOST;
  config.server.port = SMTP_PORT;
  config.login.email = SENDER_EMAIL;
  config.login.password = SENDER_PASSWORD;
  config.login.user_domain = F("127.0.0.1");
  config.time.ntp_server = F("pool.ntp.org,time.nist.gov");
  config.time.gmt_offset = 3;
  config.time.day_light_offset = 0;

  // declare the message class
  SMTP_Message message;

  // set the message headers
  message.sender.name = F("Arduino");
  message.sender.email = SENDER_EMAIL;
  message.subject = subject;
  message.addRecipient(F("To Whom It May Concern"), RECIPIENT_EMAIL);

  message.text.content = textMsg;
  message.text.transfer_encoding = "base64";
  message.text.charSet = F("utf-8");
  message.priority = esp_mail_smtp_priority::esp_mail_smtp_priority_low;

  // set the custom message header
  message.addHeader(F("Message-ID: <abcde.fghij@gmail.com>"));

  // connect to the server
  if (!smtp.connect(&config)) {
    Serial.println("Connection error: ");
    Serial.print("- Status Code: ");
    Serial.println(smtp.statusCode());
    Serial.print("- Error Code: ");
    Serial.println(smtp.errorCode());
    Serial.print("- Reason: ");
    Serial.println(smtp.errorReason().c_str());
    return;
  }

  if (!smtp.isLoggedIn()) {
    Serial.println("Not yet logged in.");
  } else {
    if (smtp.isAuthenticated())
      Serial.println("Successfully logged in.");
    else
      Serial.println("Connected with no Auth.");
  }

  // start sending Email and close the session
```

```
  if (!MailClient.sendMail(&smtp, &message)) {
    Serial.println("Connection error: ");
    Serial.print("- Status Code: ");
    Serial.println(smtp.statusCode());
    Serial.print("- Error Code: ");
    Serial.println(smtp.errorCode());
    Serial.print("- Reason: ");
    Serial.println(smtp.errorReason().c_str());
  }
}

// callback function to get the Email sending status
void smtpCallback(SMTP_Status status) {
  // print the current status
  Serial.println(status.info());

  // print the sending result
  if (status.success()) {
    for (size_t i = 0; i < smtp.sendingResult.size(); i++) {
      // get the result item
      SMTP_Result result = smtp.sendingResult.getItem(i);

      Serial.print("Status: ");
      if (result.completed)
        Serial.println("success");
      else
        Serial.println("failed");

      Serial.print("Recipient: ");
      Serial.println(result.recipients.c_str());
      Serial.print("Subject: ");
      Serial.println(result.subject.c_str());
    }
    Serial.println("----------------\n");

    // free the memory
    smtp.sendingResult.clear();
  }
}
```

This code initializes the system, connects to Wi-Fi, and sends an email reminder when the scheduled time is reached. The servo motor is controlled to release pills, and the LCD displays system status messages.

3.3 WORKING METHODOLOGY

The automatic pill dispenser works by utilizing an Arduino R4 Wi-Fi microcontroller to manage and control the entire system. Once the user or caregiver sets up the medication schedule and dosage requirements, the microcontroller uses an internal clock to monitor the designated times. At the scheduled intervals, the Arduino activates the SG90 servo motor to release the correct number of pills from the

dispenser. The motor is programmed to rotate to a specific position, ensuring that the pills are dispensed accurately.

The system features a 1602 LCD screen that provides real-time information to the user, such as the next scheduled dose, the number of pills remaining, and the current status of the device. This feedback helps the user track their medication and ensures that they are aware of upcoming doses. At the time of each scheduled dose, an audible buzzer sounds to alert the user that it is time to take their medication. This serves as a backup to the visual display, ensuring the reminder is not missed,even if the user is distracted or in another room.

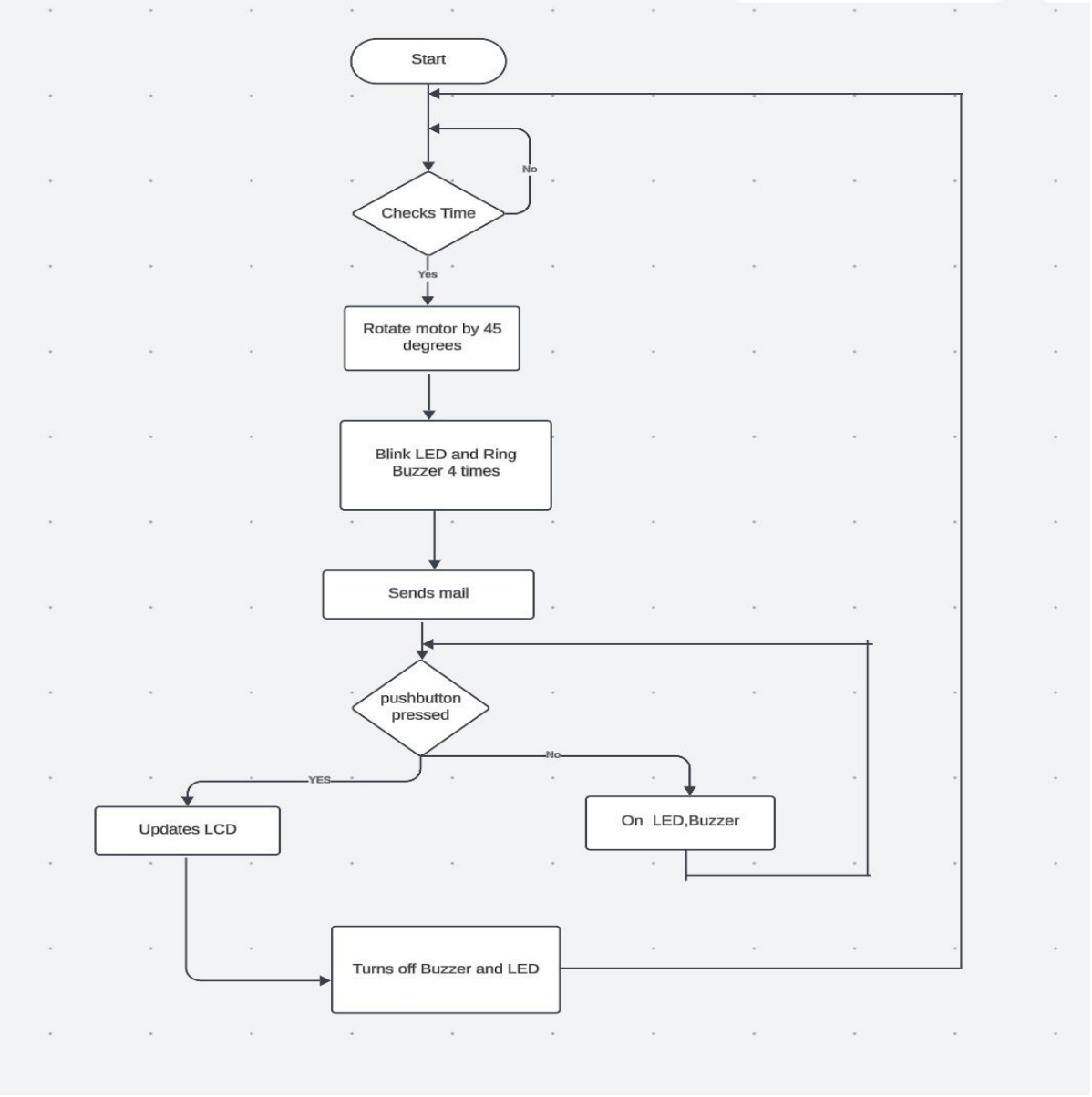

Figure 3.3(a). Flowchart of Pill Dispenser

In addition to the visual and audible reminders, the system utilizes the Wi-Fi capabilities of the Arduino R4 to send email notifications to the user or caregiver. These emails serve as an additional reminder for the scheduled medication time, ensuring that the user is notified even when away from the device. The email notification feature adds another layer of convenience and reliability to the system, promoting better adherence to the prescribed medication regimen.

The system is designed to be easily adjustable, allowing users to reprogram the device whenever necessary. This flexibility ensures that the pill dispenser can adapt to changing medication schedules and dosages, providing an ongoing solution to medication management. By automating the pill dispensing process and providing multiple forms of reminders, the system helps improve medication adherence, reduce the risk of missed doses, and enhance overall health management.

3.4 IMPLEMENTATION RESULTS

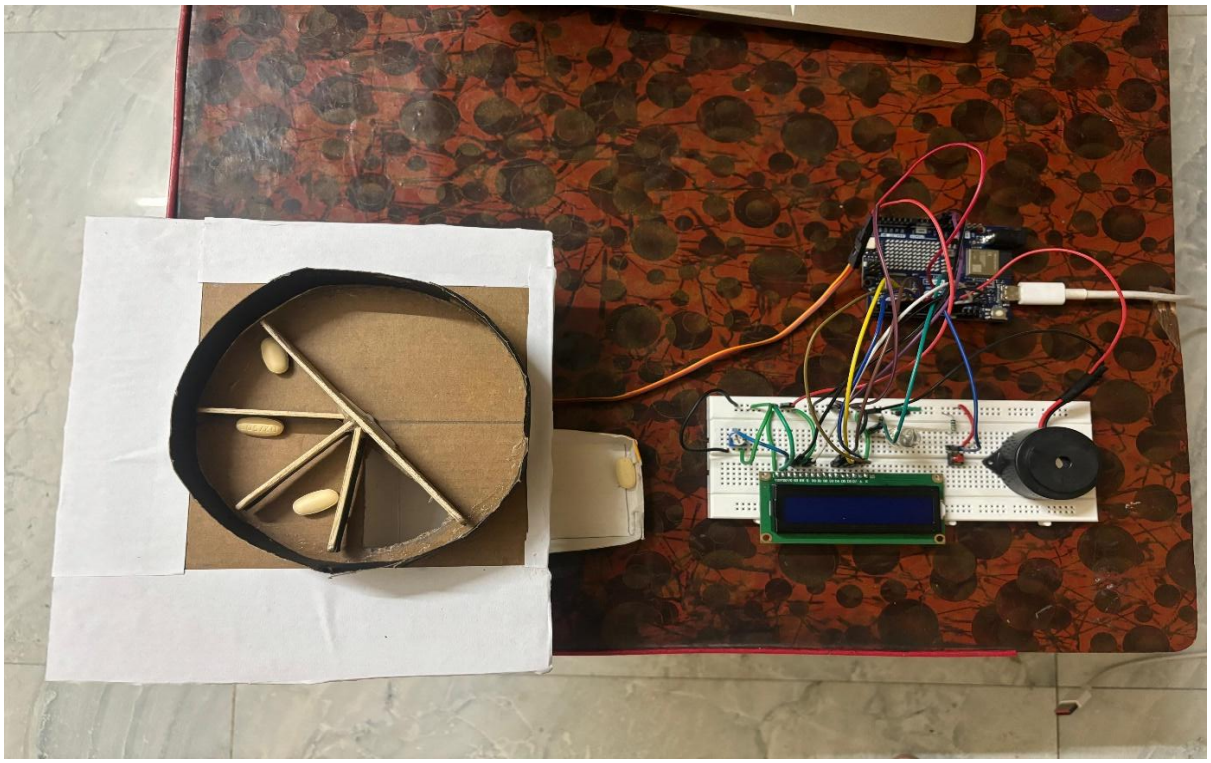

Figure 3.4(a). Top view of the circuit and its mechanism

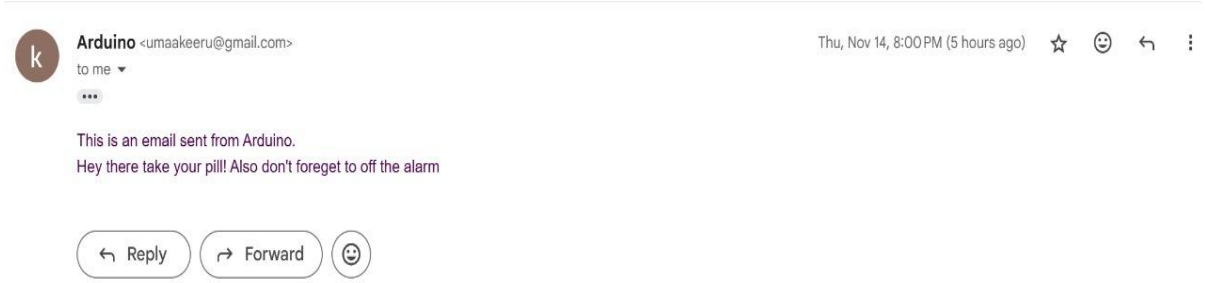

Figure 3.4(b). Received mail text from Arduino

Figure 3.4(c). Top view of showing pill dispensing demonstration with Arduino connection via USB cable

CHAPTER-4

4.1 RESULT

The automatic pill dispenser project is a fully functional system that successfully automates the pill dispensing process, ensuring timely medication adherence for users. The integration of the Arduino R4 Wi-Fi microcontroller, SG90 servo motor, 1602 LCD screen, and Wi-Fi connectivity enables the dispenser to accurately release pills at scheduled times, display relevant information, and send email reminders to users or caregivers. The system's buzzer provides auditory alerts, while the pushbutton allows for manual control when necessary. The project achieves its goal of improving medication adherence through automation and real-time notifications, making it a practical solution for managing medication schedules in various healthcare settings.

4.2 PROS AND CONS OF THE PROJECT

Pros:

1. Improved Medication Adherence: Automates pill dispensing and sends timely reminders via alerts and email, reducing missed doses.
2. User-Friendly: The LCD screen displays medication schedules and pill counts for easy tracking.
3. Remote Notifications: Sends email reminders to users or caregivers through Wi-Fi connectivity.
4. Automation: Reduces manual intervention with the servo motor for accurate pill dispensing.
5. Cost-Effective: Utilizes affordable components like Arduino R4 and SG90 servo motor.
6. Customizable: Can be adjusted for different medication schedules and user needs.

Cons:

1. Limited Storage: Pill storage capacity is restricted, requiring refills or maintenance.
2. Power Dependency: Relies on continuous power through USB, with no backup for outages.
3. Technical Setup: Requires basic knowledge for setup, including Wi-Fi and email configuration.
4. Wear and Tear: Mechanical parts like the servo motor may degrade over time.
5. Internet Dependency: Email notifications rely on a stable internet connection.
6. Security Concerns: Sending sensitive data over email may raise privacy issues without proper security measures.

4.3 APPLICATIONS OF THE PROJECT

Elderly Care: The automatic pill dispenser is highly beneficial for elderly individuals who may have difficulty remembering to take their medication on time. The system provides timely reminders via email and audible alerts, helping reduce the risk of missed doses and improving overall medication adherence, which is crucial for managing age-related health conditions.

Chronic Disease Management: For patients with chronic conditions such as diabetes, hypertension, or heart disease, maintaining a consistent medication schedule is critical. This system can automate the dispensing of multiple medications at specific times, ensuring that patients adhere to prescribed regimens and improve their long-term health outcomes.

Caregiver Support: Caregivers of individuals with cognitive impairments, such as those suffering from Alzheimer's or dementia, can benefit from the system. The automatic reminders and pill dispensing ensure that the patient receives the proper medication at

the right times, providing peace of mind to caregivers and reducing their workload.

Home Healthcare: The system can be integrated into broader home healthcare setups, allowing caregivers or healthcare professionals to remotely monitor medication adherence. With Wi-Fi connectivity and email alerts, caregivers can receive updates on whether the patient has taken their medication, improving patient care and reducing the likelihood of errors.

Mental Health and Paediatric Use: The system can help individuals with mental health conditions or children who require regular medication by providing consistent reminders and ensuring the correct dosage is taken. This is especially helpful in preventing missed doses, which can be critical for conditions like depression or ADHD.

Clinical Research: In clinical trials or pharmaceutical studies, accurate medication tracking is essential. The automatic pill dispenser can be used to monitor patient adherence to study protocols by providing real-time data on when and how medications are dispensed, helping researchers analyse the effects of treatment regimens.

4.4 CONCLUSION AND FUTUREWORKS

Conclusion

In conclusion, the automatic pill dispenser project successfully addresses the critical issue of medication adherence by combining automation, timely reminders, and remote notifications. By integrating components such as the Arduino R4 Wi-Fi, SG90 servo motor, 1602 LCD screen, and Wi-Fi connectivity, the system efficiently dispenses pills at scheduled times, ensures accurate dosing, and sends email alerts to users or caregivers. The project's practical results demonstrate its potential to assist elderly patients, individuals with chronic conditions, and caregivers in managing medication schedules, reducing the risk of missed doses, and improving overall health outcomes. This project not only showcases the power of automation in healthcare but also provides a cost-effective, customizable solution to enhance medication management and patient care.

Future works

Future work on the automatic pill dispenser project could focus on several enhancements to improve its functionality and user experience. Integrating voice assistance or text-to-speech features would provide better accessibility, especially for elderly or visually impaired users. A mobile app could be developed for real-time monitoring and management of medication schedules, allowing users to receive reminders and track doses remotely. The system could also be expanded to support multiple users with advanced scheduling features, while integration with smart home devices could offer seamless operation. Additionally, incorporating a pill refill tracking system, improving security with password protection, and adding a battery backup for uninterrupted service would enhance the device's practicality and reliability. These upgrades would further optimize the system for diverse healthcare needs and improve medication adherence.

4.5 REFERENCES

1) https://www.sciencebuddies.org/science-fair-projects/project-ideas/Elec_p105/electricity-electronics/automatic-pill-dispenser

2) https://newbiely.com/tutorials/arduino-uno-r4/arduino-uno-r4-email

3) https://www.theseus.fi/bitstream/handle/10024/793793/Thapa_Gaurav%20Bikram.pdf?sequence=2

4) https://github.com/quibueno/med_pill

5) https://projecthub.arduino.cc/mWiFiirandadrummond/the-pill-dispenser-10-abc990