

**A Final Year Project Report on**  
**Automatic Medicine Reminder Device**

(A dissertation submitted in partial fulfilment of the requirements of Bachelor of Technology  
in Electronics & Communication Engineering of the West Bengal University of Technology,  
West Bengal)

**Academic Year: 2023-2024**

Submitted by

**Shubhasis Ghosh [MAKAUT Roll No: 25900321009]**

**Pritam Roy [MAKAUT Roll N.:25900321018]**

**Bijayeeta Biswas [MAKAUT RollNo:25900321015]**

**Sneha Das [MAKAUT Roll No: 25900321004]**

**Nitish Ghosh [MAKAUT Roll No: 25900321022]**

**Aritra Roy [MAKAUT Roll No: 25900321014]**

**Suman Singha [MAKAUT Roll No: 25900321008]**

**Deepraj Sarkar [MAKAUT Roll No: 25900321025]**

Under the guidance of

**Dr. Nilanjan Mukhopadhyay**

Head of The Department

Dept. of Electronics & Communication Engineering



**Global Institute of Management & Technology**

(Affiliated to West Bengal University of Technology)

Krishnanagar – 741102, Nadia, WB



## Global Institute of Management & Technology

(Affiliated to West Bengal University of Technology)

Krishnanagar – 741102, Nadia, WB

### Certificate of Approval

This is to certify that the project report on “**Automatic Medicine Reminder Device**” is a record of bonafide work, carried out by **our Group members** under my guidance and supervision.

In my opinion, the report in its present form is in conformity as specified by Global Institute of Management & Technology and as per regulations of the Maulana Abul Kalam Azad University. To the best of my knowledge the results presented here are original in nature and worthy of incorporation in project report for the B.Tech. program in Electronics & Communication in the academic year 2023-2024.

.....

Signature of  
External Invigilator

.....

Signature of  
Guide

.....

Signature of  
Head, Dept of ECE

## ACKNOWLEDGEMENTS

I would like to express our thanks to all the faculty members specially Dr. Nilanjan Mukherjee (head of the department) instructing me during the whole Semester for motivating constantly, providing this wonderful innovative project, preparation of this seminar & report, acquainting me with the required details, & providing us with this wonderful opportunity.

Name	Roll no	Signature
<b>Shubhasis Ghosh</b>	25900321009	
<b>Pritam Roy</b>	25900321018	
<b>Bijayeeta Biswas</b>	25900321015	
<b>Sneha Das</b>	25900321004	
<b>Nitish Ghosh</b>	25900321022	
<b>Aritra Roy</b>	25900321014	
<b>Suman Singha</b>	25900321008	
<b>Deepraj Sarkar</b>	25900321025	

# CONTENT

Topic	Page No
ABSTRACT	5
1. INTRODUCTION	6
2. THEORETICAL BACKGROUND	7-14
Hardware properties	
1. Arduino Uno Module	7
2. Bread board	8
3. DS3231 RTC module	9
4. 16*2 LCD Display	10
5. Potent meter	11
6. Push button	12
7. Resistor	12
8. Buzzer	13
9. USB Cable	13
10. Jumpier Wire	14
11. Single stand wire	14
○ 3.Circuit Diagram	15
○ 4. Hardware Image	16
○ 5.Module Image	17
○ 6.Arduino IDE Compiler	18
○ 7.Backend Codes	19-24
○ 8.Project Costs	25
● 9. Result	26
● 10. CONCLUSION	27
● 11. REFERENCE	28

# ABSTRACT

The Automatic Medicine Reminder Device (AMRD) project aims to address the challenge of medication adherence, particularly for individuals with chronic conditions or complex medication regimens. Medication non-adherence remains a significant issue in healthcare, leading to adverse health outcomes, increased healthcare costs, and decreased overall well-being. The AMRD project introduces an innovative solution to enhance medication adherence through the integration of smart technology into the daily lives of patients.

The key components of the AMRD include a portable electronic device, a mobile application, and a cloud-based platform. The device is designed to securely store and dispense medication at scheduled times, ensuring accurate dosage and timely administration. The mobile application serves as a user-friendly interface, allowing patients to set up personalized medication schedules, receive real-time reminders, and track their adherence progress. The cloud-based platform facilitates data storage, analysis, and communication between the device and the user's mobile application, creating a seamless and connected healthcare ecosystem.

## Key Features:

1. **Automated Medication Dispensing:** The AMRD device incorporates a secure medication compartment and a dispensing mechanism controlled by a microcontroller. It accurately dispenses the prescribed dosage at predetermined times, eliminating the risk of human error and promoting medication adherence.
2. **Customizable Reminders:** The mobile application provides users with the flexibility to customize medication schedules based on their unique prescription plans. Users can receive reminders through push notifications, ensuring they never miss a dose and promoting consistent adherence.
3. **User-Friendly Interface:** The mobile application features an intuitive interface, making it easy for users to input medication details, update their schedules, and track their adherence history. The application also allows users to receive educational content related to their medications and health conditions.

# INTRODUCTION

Medication adherence is a critical factor in the effective management of chronic conditions and overall healthcare outcomes. However, the complexities of modern life often lead to challenges in maintaining a consistent and accurate medication routine. Automatic Medicine Reminder Devices (AMRDs) have emerged as innovative solutions to address this issue by integrating smart technology into healthcare practices. The AMRD project presented in this document aims to contribute to the enhancement of medication adherence through the development of a sophisticated, user-friendly, and automated system.

In recent years, advancements in electronics, microcontrollers, and connectivity have paved the way for the integration of smart devices into various aspects of healthcare. The AMRD project capitalizes on these technological strides to create a comprehensive solution that goes beyond conventional pill organizers and manual reminders. By incorporating a portable electronic device, a mobile application, and a cloud-based platform, the AMRD seeks to provide a holistic approach to medication management.

The prevalence of chronic diseases and the aging population highlight the increasing need for effective medication adherence strategies. Poor adherence not only jeopardizes individual health but also contributes to the rising healthcare costs associated with preventable complications. The AMRD project recognizes the potential of technology to empower individuals in managing their health, offering a proactive and intelligent solution to bridge the gap between prescribed treatments and patient compliance.

This document outlines the key features of the AMRD, emphasizing its automated medication dispensing capabilities, customizable reminders, and user-friendly interfaces. The integration of cloud connectivity allows for real-time data exchange, enabling healthcare providers to monitor and intervene when necessary. The project envisions a future where smart healthcare solutions, such as the AMRD, play a pivotal role in improving medication adherence, enhancing patient outcomes, and ultimately contributing to a more efficient and sustainable healthcare system.

## THEORETICAL BACKGROUND

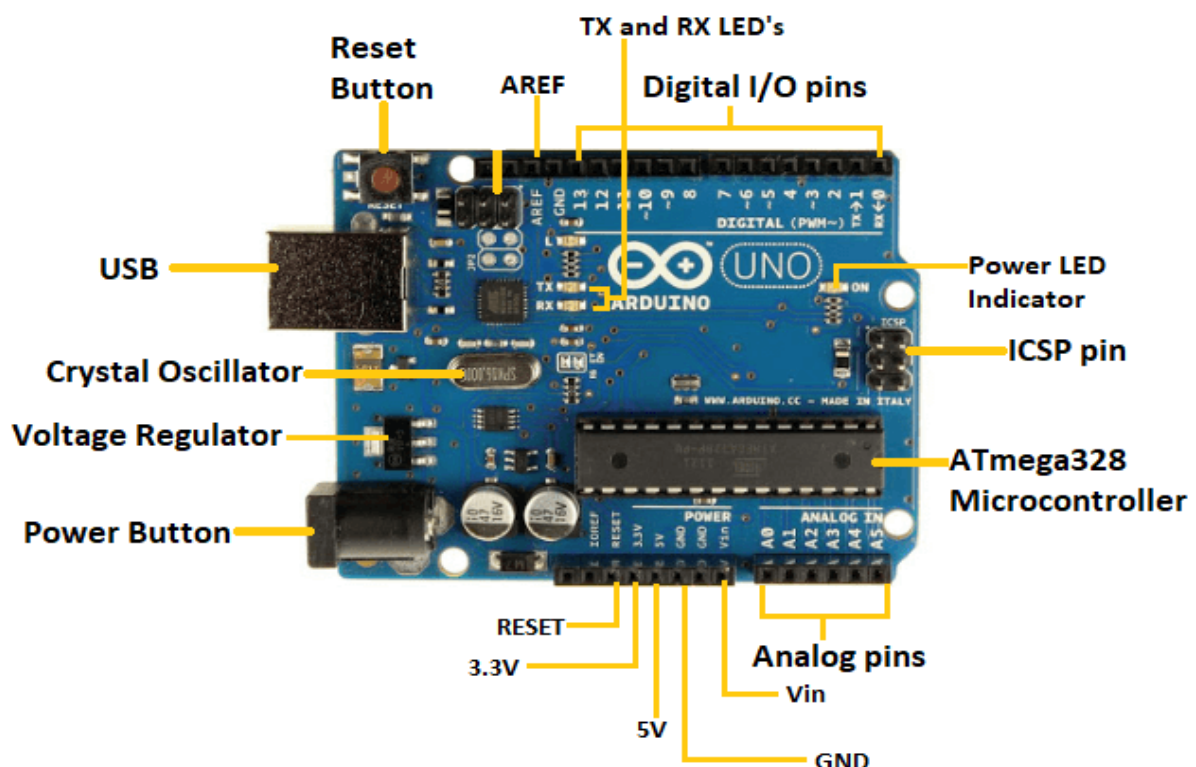
### Hardware properties

**1. Arduino Uno Module:** The Arduino UNO is a standard board of Arduino. Here UNO means 'one' in Italian. It was named as UNO to label the first release of Arduino Software. It was also the first USB board released by Arduino. It is considered as the powerful board used in various projects. Arduino.cc developed the Arduino UNO board.

Arduino UNO is based on an ATmega328P [microcontroller](#). It is easy to use compared to other boards, such as the Arduino Mega board, etc. The board consists of digital and analog Input/Output pins (I/O), shields, and other circuits.

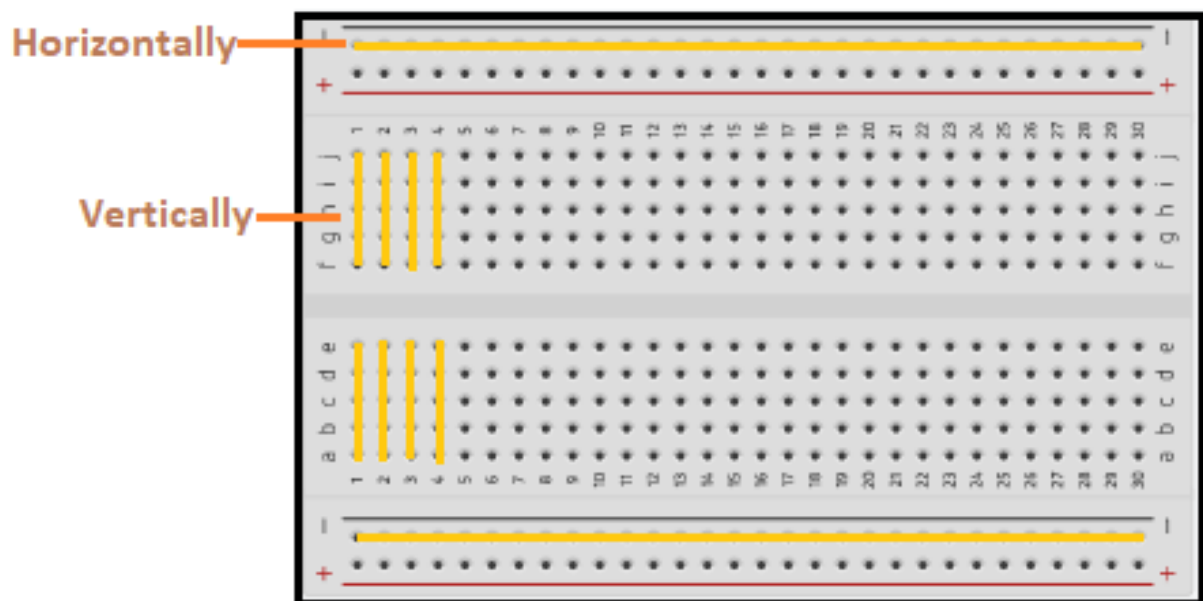
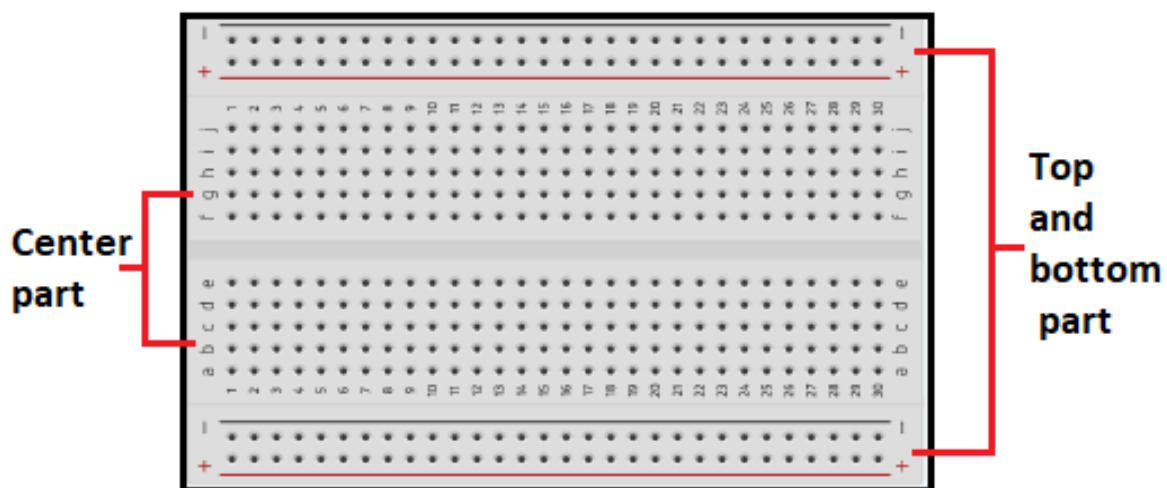
The Arduino UNO includes 6 analog pin inputs, 14 digital pins, a [USB](#) connector, a power jack, and an ICSP (In-Circuit Serial Programming) header. It is programmed based on IDE, which stands for Integrated Development Environment. It can run on both online and offline platforms.

The [IDE](#) is common to all available boards of Arduino.



**2. Bread board:** The breadboard is a white rectangular board with small embedded holes to insert electronic components. It is commonly used in electronics projects. We can also say that breadboard is a prototype that acts as a construction base of electronics.

A breadboard is also categorized as a **Solderless board**. It means that the component does not require any soldering to fit into the board. Thus, we can say that breadboard can be reused. We can easily fit the components by plugging their end terminal into the board. Hence, a breadboard is often called a **plugboard**.



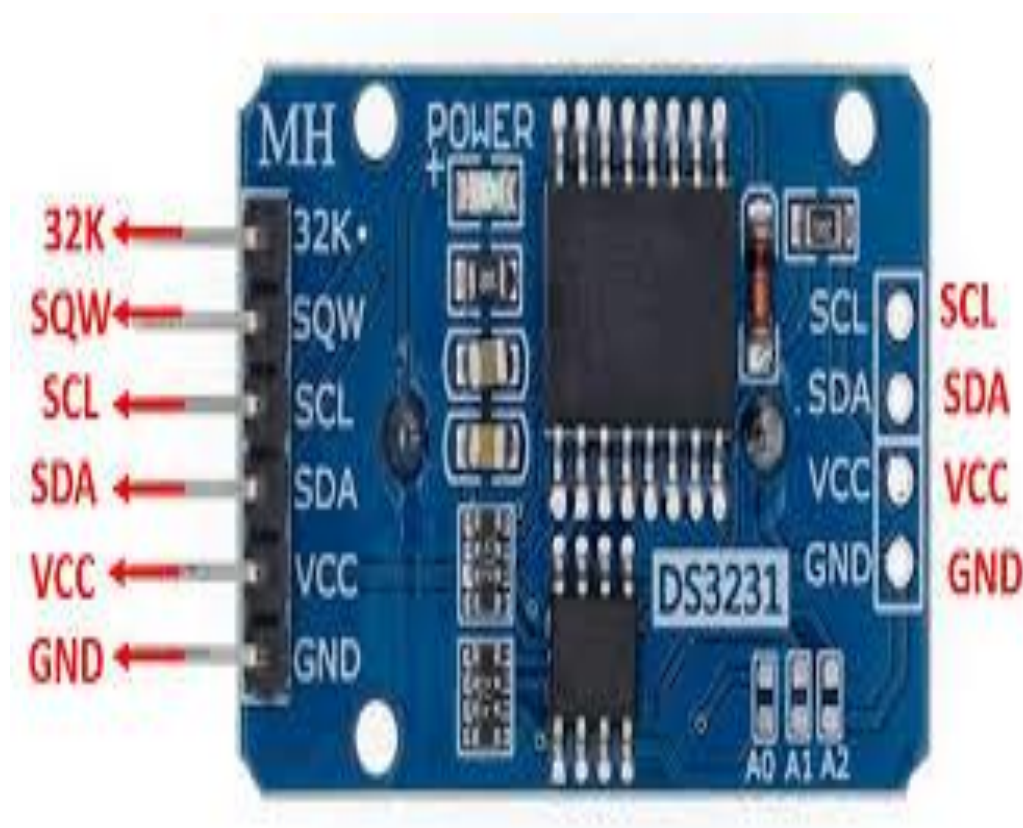


**3.DS3231RTC MODULE:** The DS3231 is a low-cost, extremely accurate I2C real-time clock (RTC) with an integrated temperature- compensated crystal oscillator (TCXO) and crystal. The device incorporates a battery input, and maintains accurate timekeeping when main power to the device is interrupted.

RTC module detects a voltage drop of the main power supply and automatically switches to a backup power supply for operation. while the MOS switch suppresses leakage current and voltage drops, enabling efficient power management of the entire system compared to diode OR circuits.

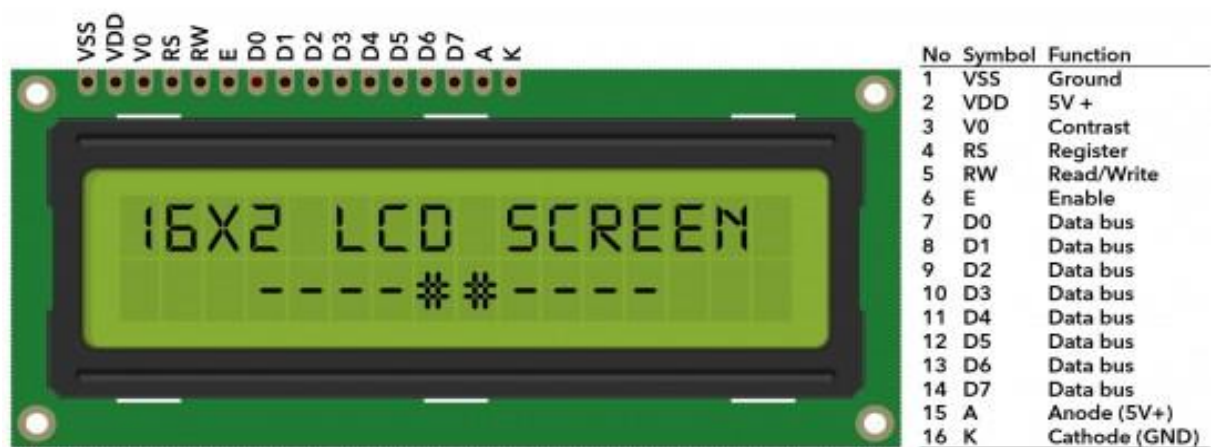
SDA is a serial data pin for the I2C interface. VCC provides power to the module. You can connect it to a 3.3 to 5 volt power supply.

When the module's power is interrupted, the device has a battery input and keeps a precise time. The device's long-term precision is improved by the inclusion of the crystal oscillator. The RTC keeps track of seconds, minutes, hours, days, dates, months, and years.



**4. 16\*2 LCD Display:** An LCD (Liquid Crystal Display) screen is an electronic display module and has a wide range of applications. A 16x2 LCD display is very basic module and is very commonly used in various devices and circuits. A 16x2 LCD means it can display 16 characters per line and there are 2 such lines.

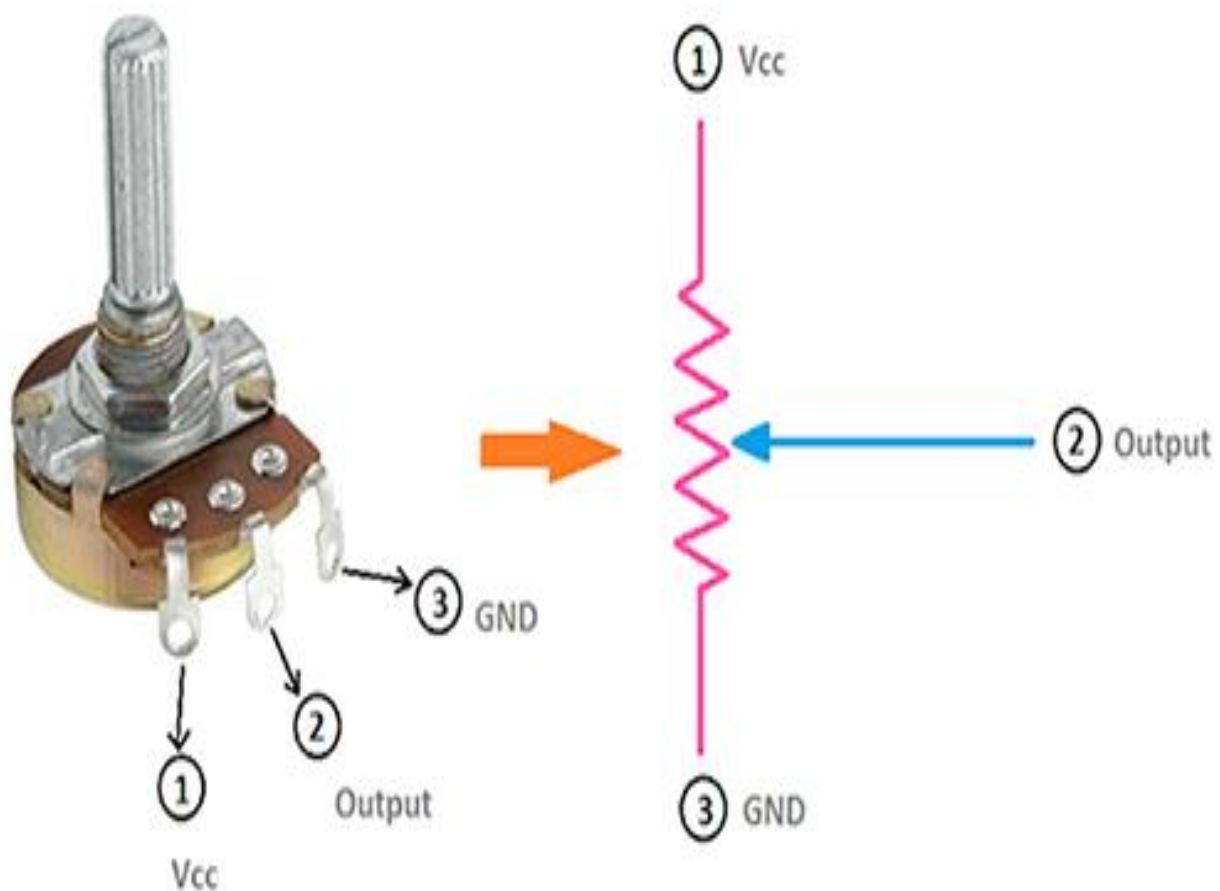
The second pin is the VCC which we connect the 5 volts pin on the Arduino Board. Next is the Vo pin on which we can attach a potentiometer for controlling the contrast of the display. Next, The RS pin or register select pin is used for selecting whether we will send commands or data to the LCD.



**4.POTENTIOMETER:** A potentiometer is an electronic device that measures the EMF (electromotive force) of a cell as well as the cell's internal resistance. It's also used to compare the EMFs of various cells. In most applications, it may also be used as a variable resistor.

A potentiometer comprises a long wire with a uniform space of the cross area. Normally the wire is comprised of manganin or constantan. Sometimes the wire might be cut into certain pieces and each piece is associated toward the end focuses through a thick metallic strip. Typically it will be copper strips. Each piece of wire has a length of one meter. For the most part, there will be six bits of wire and the all-out length of the wire is six meters. For the most part the length of the wire shifts from 4 m to 10 m. The more the length of the wire, the better the precision of the potentiometer.

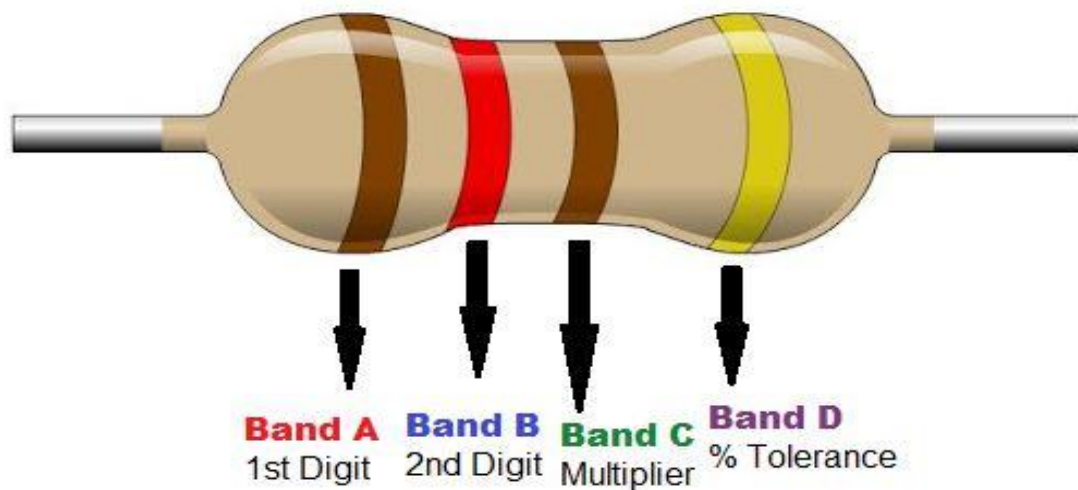
The potentiometer comprises a driving circuit that comprises a battery, key and rheostat. It additionally comprises a galvanometer and a rider. The end focuses or the terminals of the potentiometer are associated with the focuses where the potential contrast is to be estimated.



**6.PUSH BUTTON:**A push button switch is a mechanical device used to control an electrical circuit in which the operator manually presses a button to actuate an internal switching mechanism.



**7.RESISTOR:**A resistor is a passive two-terminal electrical component that implements electrical resistance as a circuit element. In electronic circuits, resistors are used to reduce current flow, adjust signal levels, to divide voltages, bias active elements, and terminate transmission lines, among other uses.



**8.BUZZER:**An audio signaling device like a beeper or buzzer may be electromechanical or piezoelectric or mechanical type. The main function of this is to convert the signal from audio to sound. Generally, it is powered through DC voltage and used in timers, alarm devices, printers, alarms, computers, etc. Based on the various designs, it can generate different sounds like alarm, music, bell & siren.



**9.USB CABLE:**The term USB stands for "Universal Serial Bus". USB cable assemblies are some of the most popular cable types available, used mostly to connect computers to peripheral devices such as cameras, camcorders, printers, scanners, and more.



**10.JUMPER WIRE:**A jump wire is an electrical wire, or group of them in a cable, with a connector or pin at each end, which is normally used to interconnect the components of a breadboard or other prototype or test circuit, internally or with other equipment or components, without soldering.



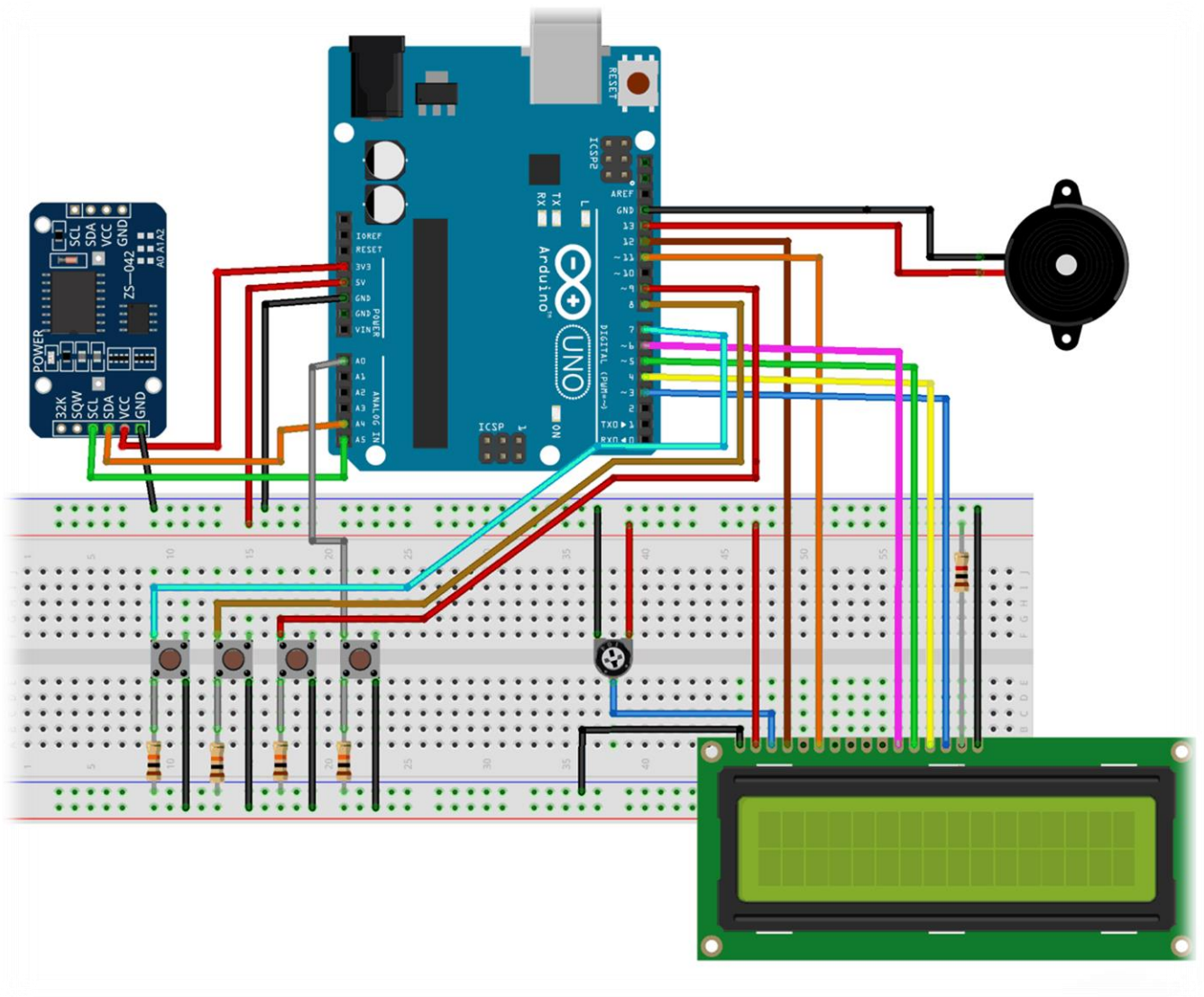
**11.SINGLE STAND WIRE:**Single-strand wire consists of a length of metal wire, usually wrapped in a protective sheath. A single core wire is a single core wire with an insulator. It is available in small diameters. Single stand wire is easy to make such a cable and can be used where flexibility is not important eg.



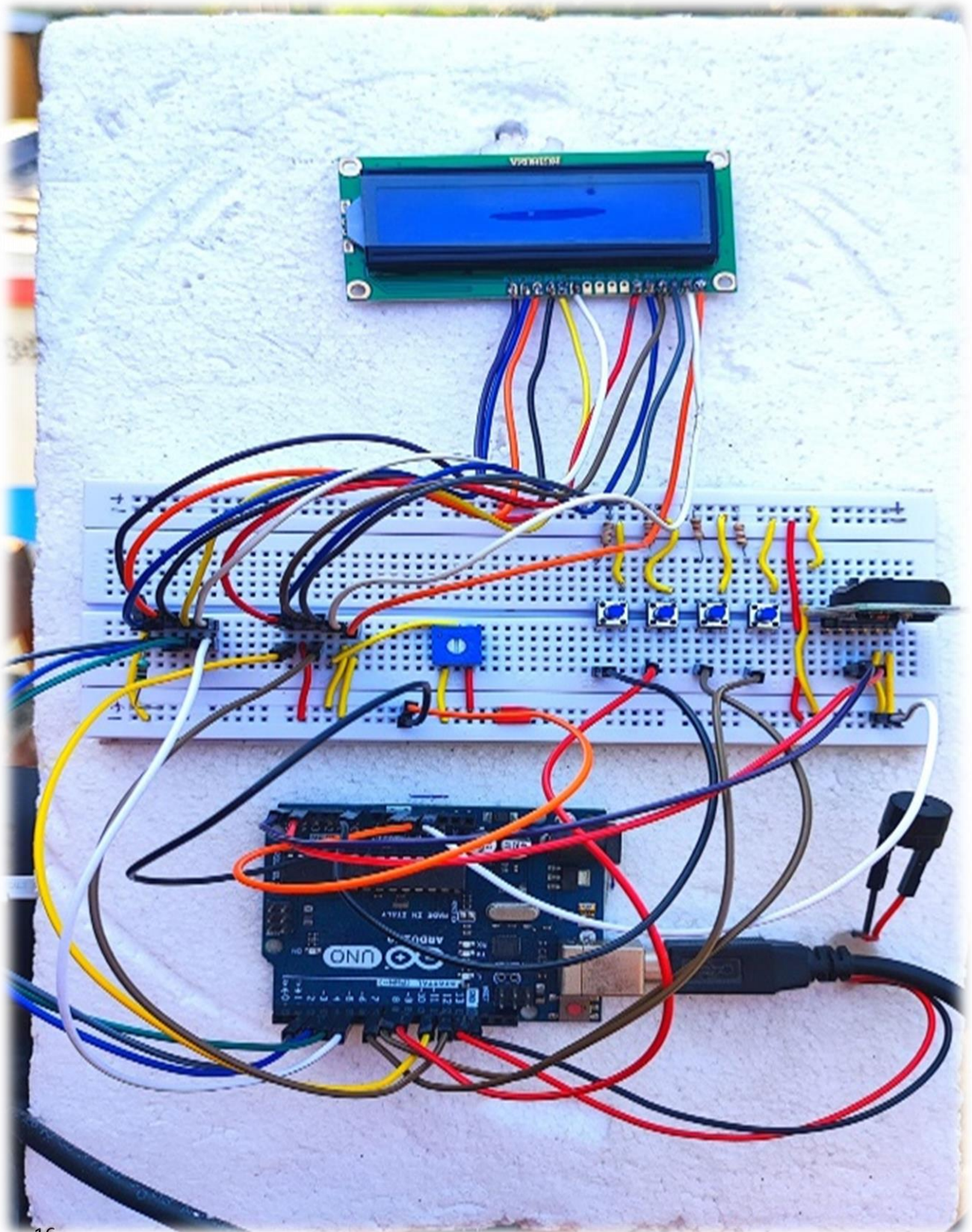


# CIRCUIT DIAGRAM

Circuit-Diagram-Arduino-Based-Medicine-Reminder-System

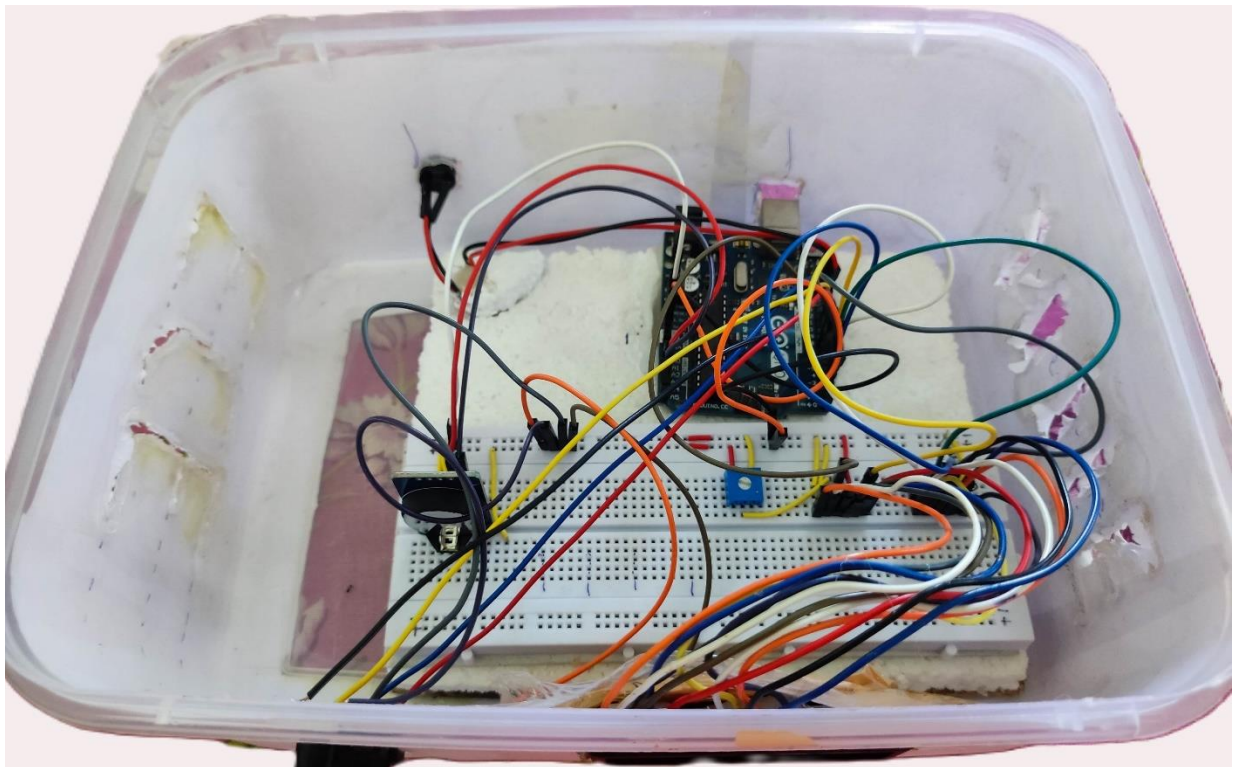


## HARDWARE IMAGE

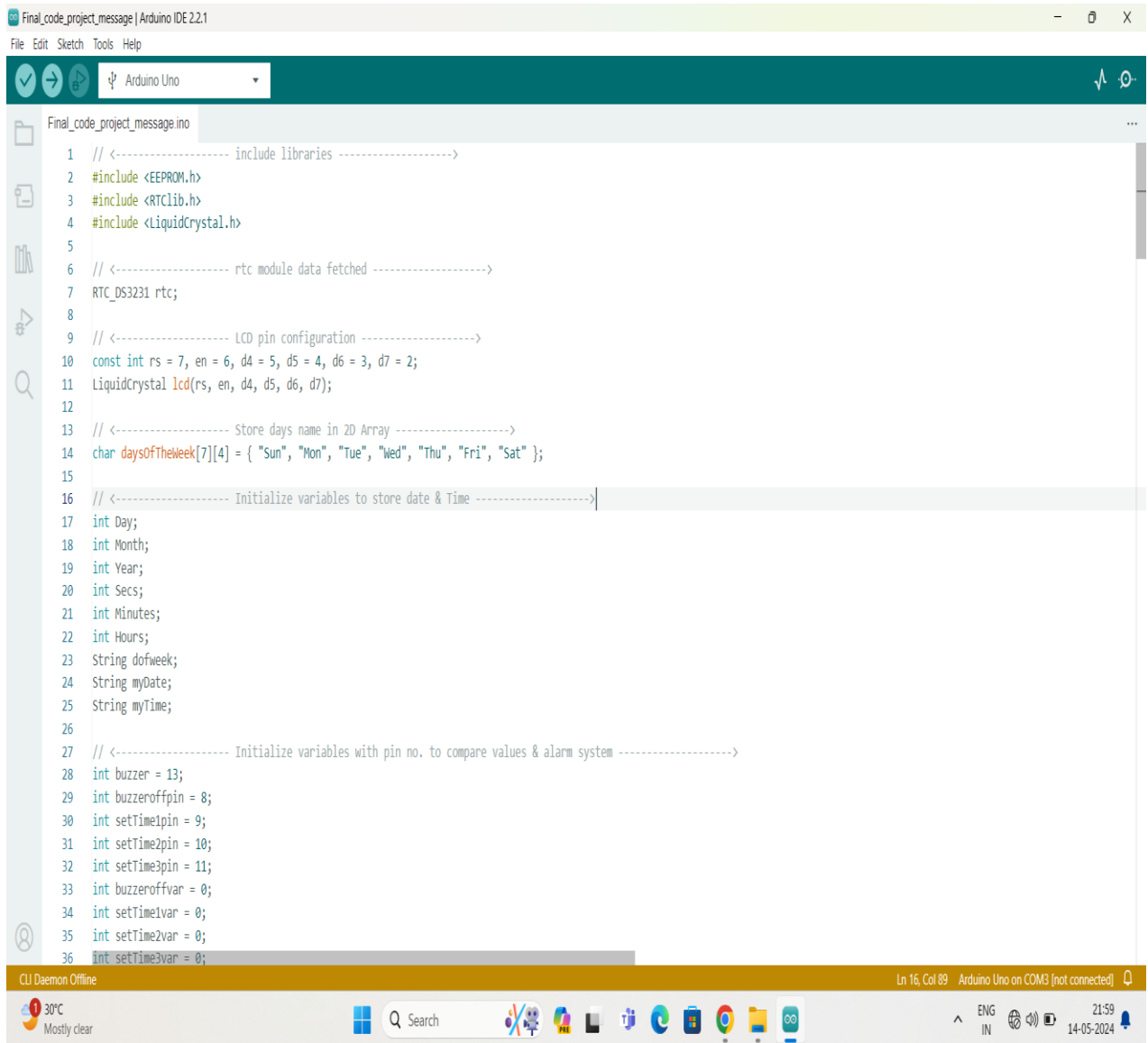




## Module Image



# Arduino IDE Compiler



The screenshot displays the Arduino IDE 2.2.1 environment. The main window shows a C++ sketch named 'Final\_code\_project\_message.ino'. The code includes libraries for EEPROM, RTClib, and LiquidCrystal. It defines pin configurations for an RTC module (DS3231) and an LCD display. The sketch also includes a 2D array for days of the week and initializes various variables for date, time, and buzzer control. The status bar at the bottom indicates the CLI Daemon is offline and the Arduino Uno is not connected to COM3.

```
1 // <----- include libraries ----->
2 #include <EEPROM.h>
3 #include <RTClib.h>
4 #include <LiquidCrystal.h>
5
6 // <----- rtc module data fetched ----->
7 RTC_DS3231 rtc;
8
9 // <----- LCD pin configuration ----->
10 const int rs = 7, en = 6, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
11 LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
12
13 // <----- Store days name in 2D Array ----->
14 char daysOfTheWeek[7][4] = { "Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat" };
15
16 // <----- Initialize variables to store date & Time ----->
17 int Day;
18 int Month;
19 int Year;
20 int Secs;
21 int Minutes;
22 int Hours;
23 String dofweek;
24 String myDate;
25 String myTime;
26
27 // <----- Initialize variables with pin no. to compare values & alarm system ----->
28 int buzzer = 13;
29 int buzzeroffpin = 8;
30 int setTime1pin = 9;
31 int setTime2pin = 10;
32 int setTime3pin = 11;
33 int buzzeroffvar = 0;
34 int setTime1var = 0;
35 int setTime2var = 0;
36 int setTime3var = 0;
```

## Backend Codes

```
// <----- include libraries ----->
#include <EEPROM.h>
#include <RTCLib.h>
#include <LiquidCrystal.h>

// <----- rtc module data fetched ----->
RTC_DS3231 rtc;

// <----- LCD pin configuration ----->
const int rs = 7, en = 6, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

// <----- Store days name in 2D Array ----->
char daysOfTheWeek[7][4] = { "Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"
};

// <----- Initialize variables to store date & Time ----->
int Day;
int Month;
int Year;
int Secs;
int Minutes;
int Hours;
String dofweek;
String myDate;
String myTime;

// <----- Initialize variables with pin no. to compare values &
alarm system ----->
int buzzer = 13;
int buzzeroffpin = 8;
int setTime1pin = 9;
int setTime2pin = 10;
int setTime3pin = 11;
int buzzeroffvar = 0;
int setTime1var = 0;
int setTime2var = 0;
int setTime3var = 0;

// <----- Initialize variables to store data in EEPROM memory -
----->
String setTime1 = "";
String setTime2 = "";
String setTime3 = "";
```

```

char buffer1[9]; // Buffer is to hold the setTime value as an array of
characters
char buffer2[9];
char buffer3[9];
String b1 = "";
String b2 = "";
String b3 = "";

// <----- This section will be execute only one time & also
setup all pinmodes ----->
void setup() {
    pinMode(buzzer, OUTPUT);
    pinMode(buzzeroFFpin, INPUT);
    pinMode(setTime1pin, INPUT);
    pinMode(setTime2pin, INPUT);
    pinMode(setTime3pin, INPUT);
    lcd.begin(16, 2);

    // <----- check if rtc module is available or connected OR
not ----->
    if (!rtc.begin()) {
        Serial.println("Couldn't find RTC");
        Serial.flush();
        // while (1)
        ;
    }
    // rtc.adjust(DateTime(F(__DATE__), F(__TIME__))); //please upload the code
2 times - first time uncomment this line & set the time & then please comment
this line again & upload code. It is necessary because everytime arduino power
on it will set time from beginning which is not correct if we do not upload
code 2nd time after commented this line
    // rtc.adjust(DateTime(2021, 1, 21, 3, 0, 0));
    Serial.begin(9600);
}

// <----- declaration of functions to setup alarm off & on ----
----->
void callAlarmOn() {
    digitalWrite(13, HIGH);
}

void callAlarmOff() {
    digitalWrite(13, LOW);
}

// <----- This section will be execute in loop -----
----->

```

```

void loop() {
  lcd.clear();
  // <----- storing date & time values in corresponding
variables----->
  DateTime now = rtc.now();
  Day = now.day();
  Month = now.month();
  Year = now.year();
  Secs = now.second();
  Hours = now.hour();
  Minutes = now.minute();
  dofweek = daysOfTheWeek[now.dayOfTheWeek()];

  // <----- storing date & time values as string -----
----->
  myDate = dofweek + " " + Day + "/" + Month + "/" + Year;
  myTime = String((Hours < 10 ? "0" : "") + String(Hours)) + ":" +
String((Minutes < 10 ? "0" : "") + String(Minutes)) + ":" + String((Secs < 10
? "0" : "") + String(Secs));

  // <----- display date & time values as string -----
----->
  Serial.println("module time - " + myDate);
  Serial.println("module time - " + myTime);
  lcd.setCursor(0, 0); //set coloumn, row value to display data in different
rows
  lcd.print(myDate);
  lcd.setCursor(0, 1); //set coloumn, row value to display data in different
rows
  lcd.print(myTime);
  delay(100);
  lcd.clear();

  lcd.setCursor(0, 0);
  lcd.print("Now Press one");
  lcd.setCursor(0, 1);
  lcd.print("switch to set time");

  // <----- read status of input pins & store data to variables
----->
  buzzeroffvar = digitalRead(buzzeroffpin);
  setTime1var = digitalRead(setTime1pin);
  setTime2var = digitalRead(setTime2pin);
  setTime3var = digitalRead(setTime3pin);

  // <----- set time if 1st or 2nd or 3rd switch is on -----
----->
  if (setTime1var == HIGH && setTime2var == LOW && setTime3var == LOW) {

```

```

        setTime1 = myTime;
        setTime1.toCharArray(buffer1, 9);           // Convert String 'a' to a
null-terminated char array
        for (int i = 0; i < setTime1.length(); i++) // Write each character of
the char array to EEPROM
        {
            EEPROM.update(i, buffer1[i]);
        }
        EEPROM.update(setTime1.length(), '\0'); // Add null terminator to the end
of the string in EEPROM
        Serial.println("EEPROM memory1 updated !!!!");
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Time 1 set");
        lcd.setCursor(0, 1);
        lcd.print("Successfully");
    }

    else if (setTime2var == HIGH && setTime1var == LOW && setTime3var == LOW) {
        setTime2 = myTime;
        setTime2.toCharArray(buffer2, 9);           // Convert String 'a' to a
null-terminated char array
        for (int i = 0; i < setTime2.length(); i++) // Write each character of
the char array to EEPROM
        {
            EEPROM.update(11 + i, buffer2[i]);
        }
        EEPROM.update(11 + setTime2.length(), '\0'); // Add null terminator to
the end of the string in EEPROM
        Serial.println("EEPROM memory2 updated !!!!");
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Time 2 set");
        lcd.setCursor(0, 1);
        lcd.print("Successfully");
    }

    else if (setTime3var == HIGH && setTime1var == LOW && setTime2var == LOW) {
        setTime3 = myTime;
        setTime3.toCharArray(buffer3, 9);           // Convert String 'a' to a
null-terminated char array
        for (int i = 0; i < setTime3.length(); i++) // Write each character of
the char array to EEPROM
        {
            EEPROM.update(22 + i, buffer3[i]);
        }
        EEPROM.update(22 + setTime3.length(), '\0'); // Add null terminator to
the end of the string in EEPROM

```

```

        Serial.println("EEPROM memory3 updated !!!!");
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Time 3 set");
        lcd.setCursor(0, 1);
        lcd.print("Successfully");
    }

    else if (setTime1var == HIGH && setTime2var == HIGH && setTime3var == HIGH
    || setTime1var == HIGH && setTime2var == HIGH || setTime2var == HIGH &&
    setTime3var == HIGH || setTime1var == HIGH && setTime3var == HIGH) {
        Serial.println("Do not press more than one timeSet switch while setting
time !!!");
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Press button");
        lcd.setCursor(0, 1);
        lcd.print("once a time");
    }
    Serial.println("setTime1 value - " + setTime1);
    Serial.println("setTime2 value - " + setTime2);
    Serial.println("setTime3 value - " + setTime3);

    // <----- Read characters from EEPROM and reconstruct the
String ----->
    for (int i = 0; i < EEPROM.length(); i++) {
        char character = EEPROM.read(i);
        if (character == '\0') {
            break; // break loop before it store null value into b string
        }
        b1 += character;
    }

    for (int i = 0; i < EEPROM.length(); i++) {
        char character = EEPROM.read(11 + i);
        if (character == '\0') {
            break; // break loop before it store null value into b string
        }
        b2 += character;
    }

    for (int i = 0; i < EEPROM.length(); i++) {
        char character = EEPROM.read(22 + i);
        if (character == '\0') {
            break; // break loop before it store null value into b string
        }
        b3 += character;
    }

```

```

// <----- compare EEPROM value in each loop with current time
to check if it's matching or not ----->
Serial.println("b1 status before check if condition- " + b1);
Serial.println("b2 status before check if condition- " + b2);
Serial.println("b3 status before check if condition- " + b3);

if (strcmp(myTime.c_str(), b1.c_str()) == 0 || strcmp(myTime.c_str(),
b2.c_str()) == 0 || strcmp(myTime.c_str(), b3.c_str()) == 0) {
    Serial.println("inside if block");
    if (buzzeroffvar == 0) {
        callAlarmOn();
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Take Your");
        lcd.setCursor(0, 1);
        lcd.print("Medicine");
    }
    else {
        callAlarmOff();
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Take Your");
        lcd.setCursor(0, 1);
        lcd.print("Medicine");
    }
} else {
    Serial.println("inside else block");
    if (buzzeroffvar == 1) {
        callAlarmOff();
    }
}

// <----- clearing all strings because every time at the end we
need fresh empty string to start next operation in next loop -----
--->
myDate = "";
myTime = "";
setTime1 = "";
setTime2 = "";
setTime3 = "";
b1 = "";
b2 = "";
b3 = "";
Serial.println("b1 status check in the last - " + b1);
Serial.println("b2 status check in the last - " + b2);
Serial.println("b3 status check in the last - " + b3);
Serial.println("");

delay(500);}

```

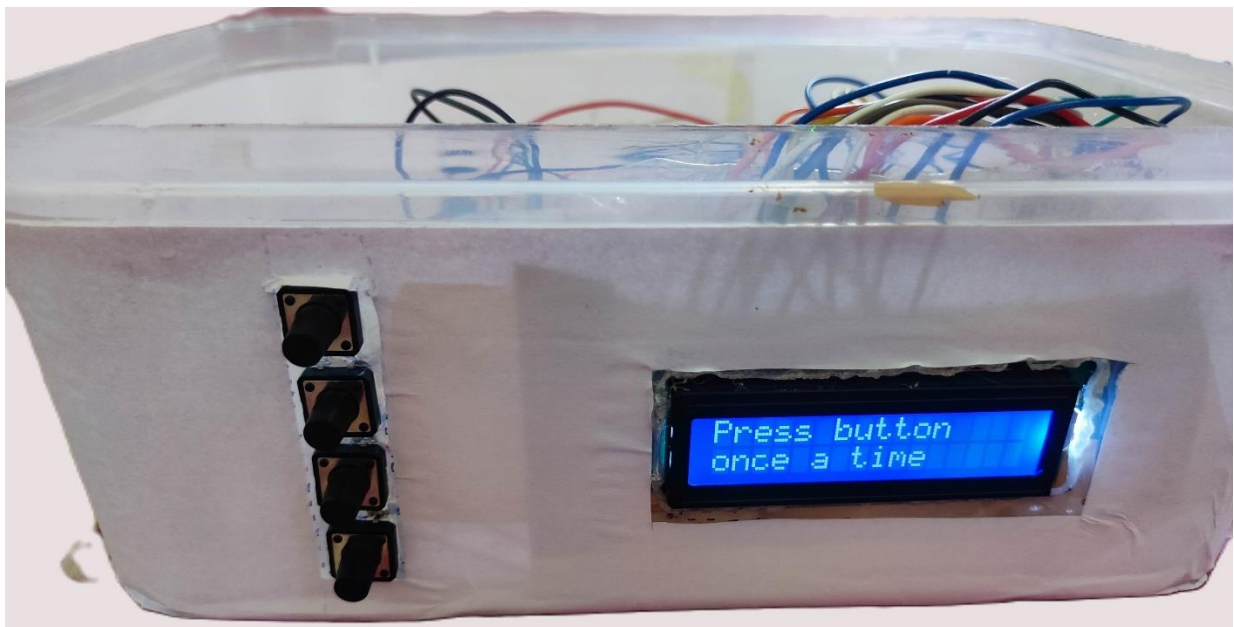
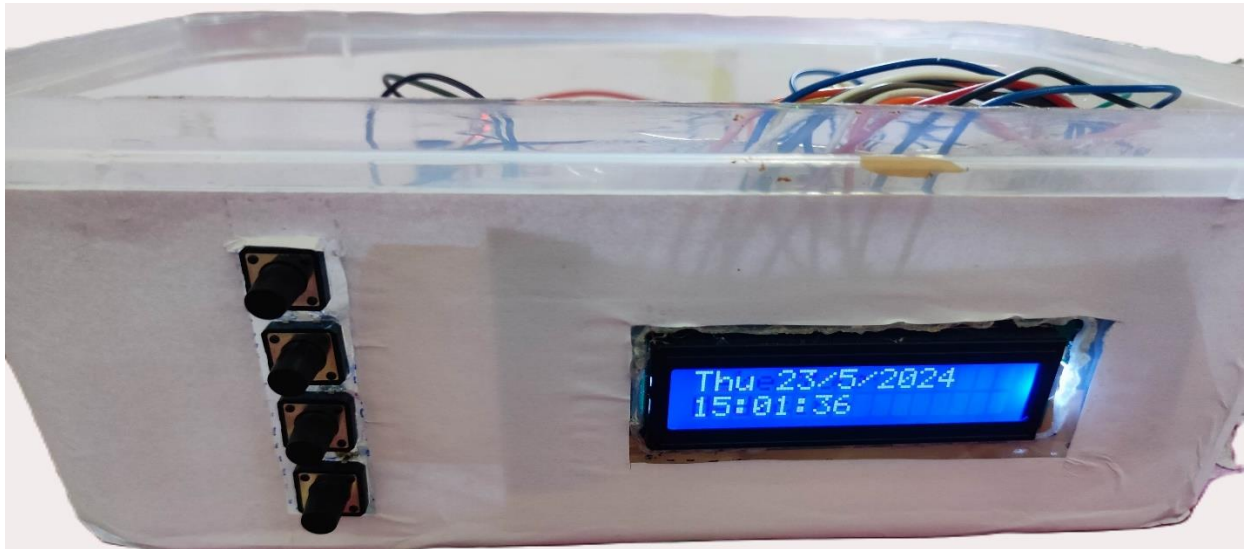


# Project cost

	PROJECT COST	
SL.NO	Product details	Buying cost
1	Arduino	850
2	Print & Misc	500
3	RTC SENSOR-1	240
4	LCD Display-1	230
5	Jumper wire	150
6	Bread Board-1	130
7	Cable	80
8	Box	80
9	Potentiometer-1	60
10	Battery DC	40
11	Push Button-	25
12	Buzzer	10
13	Resistor-4	5
14	TOTAL COST	2400

## RESULTS

Here is the final output result of our project module . It is running by the arduino based code and here first three switches are using for the set time and last below switch using as a buzzer off button .



## **CONCLUSION**

In this paper a low-cost, useful model for an automatic Medicine reminder device has been designed using simple electronics applications. For easy detection and alert, a buzzer and a LCD display has been attached so that the person in concern takes his Medicine in time in the right quantity without personalized supervision. The device also records the time and date of taking Medicine as a useful database for future medical consultation. Family members are alerted, if Medicine are not taken on time. This easy-to-use device can be a convenient option for households where family members have work-hour compulsions or are compelled to keep a mistress for the member with medical complications.

## **REFERENCE**

<https://www.google.com>[page 5-6]

<https://www.electrocnit.com>[page 15]

<https://www.E4U.com>[10-12]

<https://www.ElectronicsHub.com>[3-4]

<https://www.electronics-project-design.com>[7-14]