Bachelor's Thesis

Bachelor of Engineering, Information and Communication Technology

2023

Thapa, Gaurav Bikram

Arduino-based Medicine Dispenser

TURKU AMK
TURKU UNIVERSITY OF APPLIED SCIENCES

Bachelor’s Thesis | Abstract

Turku University of Applied Sciences

Bachelor of Engineering,Information and Communication Technology

2023 | 30

Thapa, Gaurav Bikram

Arduino-based Medicine Dispenser

The introduction of technology to healthcare sectors has brought plenty of benefits to many people worldwide. From the development of X-ray machines to the improvement of surgical techniques, technology has enriched many people's lives and made them healthier.

This thesis aimed to build a prototype of an Arduino-based medicine dispenser that is affordable and straightforward to make. This project follows the waterfall methodology and has five phases: requirement and planning, design phase, development phase, and testing and maintenance.

Keywords:

Arduino, medicine dispenser, health technology, embedded system, waterfall methodology

Content

Figures

List of abbreviations (or) symbols

Abbreviation	Explanation of abbreviation (Source)
DC	Direct Current
GSM	Global System for Mobile Communication
IDE	Integrated Development Environment
LED	Light-Emitting Diode
NCD	Non-Communicable Diseases
PWM	Pulse-Width Modulation
RC	Remote Control
RGB	Red, Green, and Blue
RTC	Real-Time Clock
WHO	World Health Organisation

1 Introduction

Consumption of unhealthy foods and an unhealthy lifestyle cause several health problems early in life. Over the years, they tend to be very problematic, and people must rely on medications to alleviate pain or symptoms. Therefore, different drugs become essential to a person's day-to-day life. According to WHO, noncommunicable diseases (NCDs) kill 41 million people yearly, equivalent to 74% of all deaths globally. [1] With so many people dying of such diseases and many more getting diagnosed with numerous NCDs daily, there is greater demand for healthcare. As a result, the health sector has worked with technology to meet that demand and improve global health.

Technology adoption in the health sector has been dramatically transforming the health industry. As a result, we now have access to innovative technologies that help us provide better healthcare services. Because of technology, the lines between consumer technologies and professional health technologies are now blurry. Now people can access several assistive devices that help them monitor and improve their health condition. Assistive devices are valuable achievements of technology in the health sector. [2] Medicine dispensers are a splendid example of assistive devices. A *medicine dispenser* is an assistive device that supplies medications at the proper times established per the medical professional's suggestion. This helps the patients avoid forgetting their medication.

The project Arduino-based medicine dispenser presents a simple, low-cost, and intuitive implementation of technology in the health sector. The medicine dispenser uses an Arduino as a controller, for timekeeping uses a real-time clock (RTC) module, uses a servo motor to take medicine from the medicine container and dispense it, uses a buzzer as an alarm when taking medication, uses an ultrasonic sensor to detect when the pill has been taken off the box or not and sends an alert message first when the medicine is dispensed, later when the medicine is not taken for more than 5 minutes using the GSM module.

1.1 Background of the project

Around 41 million (about twice the population of New York) people die annually from noncommunicable diseases, accounting for 71 % of all deaths globally. It is expected to increase by 17 % by 2025. [3][4] NCDs, otherwise called chronic diseases, are in general, long-term and are the result of unhealthy lifestyles and other environmental factors. The primary non-communicable diseases are cardiovascular diseases, cancers, chronic respiratory diseases, and diabetes. They require rigorous medication adherence for an extended period to prevent or minimize their symptoms. One of the frequent problems for medication adherence is remembering the right time to ingest the right medicine. Moreover, forgetfulness or confusion in consuming medication can harm a person's health and bring complications.

Patients tend to forget to take their medications at the indicated hours because most live alone or their guardians forget about them. According to the poll conducted by Epilepsy Research UK, almost fifty percent of people forget to take their medication at least once a month. [5] Since it has become a widespread problem, researchers are actively working with technology and health services to understand the non-adherence of medication regimens better and find its solutions. Furthermore, there are alternatives to this problem, such as Medicine dispensers. Medicine dispensers have become a beneficial tool for patients. It allows them to remember when to take their medications, avoiding the inefficiency of the treatment for their health. Among the few medicine dispensers available in the market are mentioned bellow:

- Gogooda Weekly PIL Organizer
- Sagely Smart XL Weekly PILL Organizer
- Ezy Dose Push Button Pill Planner
- Hero Automatic Medication Dispenser
- E-Pill Voice
- MedaCube
- Lizimandu Weekly Travel Pill Case

1.2 Scope of the Project

One of the most used methods to improve medical treatment efficiency is having an assistant or a caregiver who will care for and provide medication at the proper times. However, hiring a caretaker 24/7 demands much more money and human labour. Other widely used pillboxes are organizers with several sections where the medicines are deposited. This way, it is possible to have the medicine organized for its administration. However, as a disadvantage, most of them cannot generate some alarm, which would improve the treatment efficiency.

Another innovative idea consists of implementing an automatic medicine dispenser, whose primary function is to remind patients that they must take medicines and simultaneously generate an alternative in administering medications. However, several automatic dispensers are either expensive or not readily available.

1.3 Objective of the project

1.3.1 General Objective

The project's main objective is to design, implement and test the prototype of an Arduino-based medicine dispenser that is simple, low-cost, and easy to build.

1.3.2 Specific Objective

- Design a prototype of a medicine dispenser machine utilizing the Tinkercad simulation software.
- Design a mechanism to dispense the medicine.
- Implementation of an alarm system using the real-time clock.
- Alert using RGB LEDs and a buzzer.
- Sending alert messages
- Testing

.

2 Theoretical background

The medicine dispenser made for the project is an example of an embedded system. An embedded system of the project is a combination of hardware and software components to dispense medicine at a specific time.

Embedded systems perform specific tasks in a more extensive system. They operate in real-time, i.e., the clock simulated in the system operates the same as a clock in real life to ensure the scheduled system behaviour. [6] Embedded systems are generally written in C/C++ and use various specialized libraries, frameworks, or operating systems to provide fixed functionality to the system. Examples of embedded systems include consumer electronics, home automation systems, automobiles, and Internet of Things. Due to the ongoing advancement of technology and rising demand for intelligent and connected devices, embedded systems have experienced significant growth in today's world. This embedded system has the following components:

2.1 Servo motor

A servo motor is an electrical component widely used in Arduino-based projects. It provides controlled and precise movements by rotating within in specific range of angles, generally from 0 to 180 degrees. [7] It consists of a DC motor, a gear, and a feedback mechanism. The Servo motor has three wires, a power supply, a ground, and a control signal, each serving a specific purpose. The power supply wire supplies power to the servo motor, the ground wire establishes a ground connection, and the control signal wire receives Pulse Width Modulation (PWM) signal from Arduino, which rotates the motor at a specific angle depending on the duty cycle of the PWM signal. [7] Servo motors have a feedback mechanism that allows them to maintain precise movements, even under load. The best example of servo motor application can be observed in Remote Controlled (RC) vehicles, robotics, drones, etc.

Figure 1. Servo motor

Source:https://www.electronicwings.com/arduino/servo-motor-interfacing-with-arduino-uno

2.1.1 Pulse Width Modulation

Pulse Width Modulation (PWM) is a technique used in electronics to control the average power delivered to a device by varying the width of pulses in a periodic signal. It involves rapidly turning a signal ON and OFF with varying ON time durations, known as the duty cycle. Changing the duty cycle can control the average power delivered to a device. [8] PWM allows precise control of devices with differing power requirements, making it a vital tool for many Arduino-based projects and other embedded systems to control motors, LEDs, and other devices.

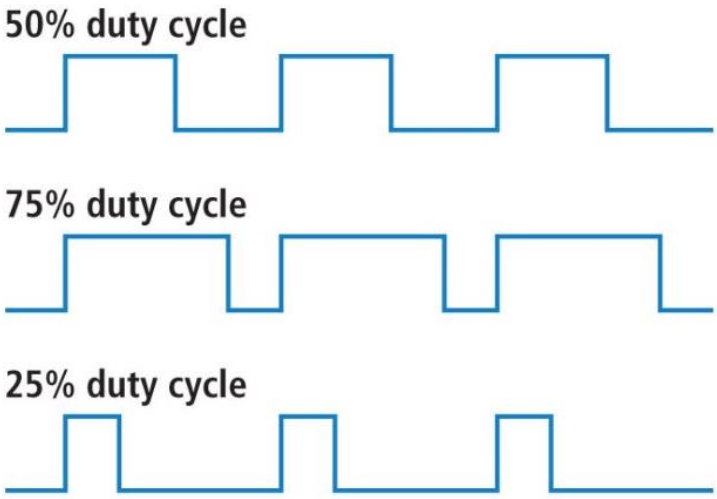


Figure 2. An example of duty cycles.

Source: https://learn.sparkfun.com/tutorials/pulse-width-modulation/all

2.2 RTC Module

The Real Time Clock (RTC) module is a system used for monitoring accurate date and time, even when an external power supply is absent. It uses a clock crystal, a backup power supply, and a microcontroller. A stable clock signal from the clock crystal is used to maintain accurate timekeeping. The microcontroller manages power management features and maintains timekeeping by reading the clock crystal. Since RTC modules are designed to operate at low power consumption, they are commonly used in Arduino projects.

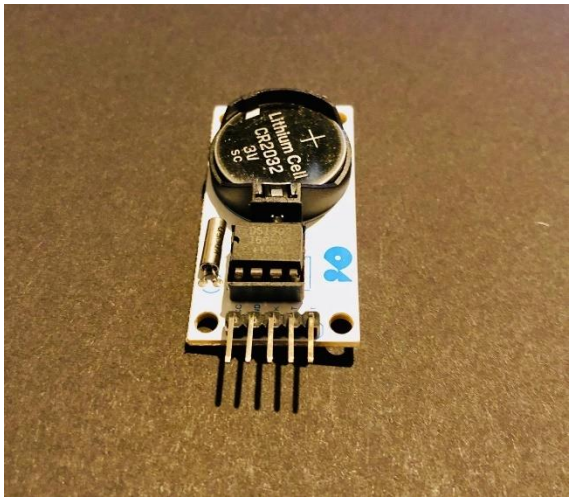


Figure 3. RTC Module.

2.3 GSM Module

Global System for Mobile Communication (GSM) module is a specialized electrical component that accepts SIM cards, and like a cell phone, it operates over a subscription to a mobile operator. It is often used in projects for wireless communication over cellular networks by enabling the functions of sending and receiving messages, making phone calls, and establishing data connections. It is interfaced with an Arduino through digital or serial communication interfaces, which allows sending and receiving commands. [9] They are often used in remote monitoring systems, security systems, IoT devices, asset tracking, etc.

Figure 4. GSM Module

https://lastminuteengineers.com/sim800l-gsm-module-arduino-tutorial/?utm_content=cmp-true

2.4 Ultrasonic Sensor

An ultrasonic sensor is a compact electronic device that helps detect and measure the distance to objects or obstacles. First, it emits high-frequency sound waves and measures the duration taken for them to bounce back after hitting an object, allowing it to measure distance. [10] They are commonly used in industrial applications, robots, and automobiles.

Figure 5. Ultra Sonic Sensor

Source: https://www.sparkfun.com/products/15569

2.5 RGB LED

RGB stands for Red, Green, and Blue. These are the primary colours used in electronic displays and lighting systems to create various colours. [11] It is a colour model that represents colours as the combination of these three primary colours at different intensities.

Figure 6. RGB WS2812B

Source: https://www.amazon.com/BTF-LIGHTING-Flexible-Individually-Addressable-Non-waterproof/dp/B01CDTEJBG

2.6 Piezo Buzzer

Piezo Buzzer is an electronic output component that generates audible signals by vibrating a piezoelectric element in response to an electric current. [12] It is typically used in alarm systems, timers, sensors, and notifications systems. It is commonly used in Arduino-based projects to generate audible alerts or simple melodies, as it is small, cost-effective, and consumes low power.

Figure: Piezo Buzzer

Source: https://fi.rsdelivers.com/product/rs-pro/rs-pro-88db-panel-mount-continuous-piezo-buzzer-x/dc-min-120v-ac/dc-max/1812673

2.7 Tinkercad

Tinkercad is an online platform that allows users to design, simulate and prototype electronic circuits in a virtual environment. It includes various digital electronic components such as resistors, motors, LEDs, sensors, logic gates, counters, etc. Creating and simulating complex digital circuits becomes much easier with Tinkercad and its drag-and-drop interface without any physical components. It also has built-in integration with Arduino, which enables any Arduino-based project to prototype and test its electronic circuits. [13]

Figure 7. Tinkercad Logo

Source: https://www.tinkercad.com/things/9CWFBMYFhxz-tinker-cad-logo

2.8 Arduino

Arduino was first developed in 2005 in Italy. [14] Arduino is a very popular open-source electronic prototyping platform that comprises software and hardware components. In recent times, Arduino has been popular among many people, and it is because it has made controlling microcontrollers much easier. Also, connecting other electrical components to the microcontroller is easy with an Arduino board.

2.8.1 Arduino Hardware:

Arduino hardware consists of physical components, i.e., Arduino boards. There are several types of Arduino boards. Among all, Arduino UNO is the most popular one. [15] All the Arduino boards consist of a microcontroller board, input and output pins, and other components. The microcontroller executes program instructions that control the device's behaviour.

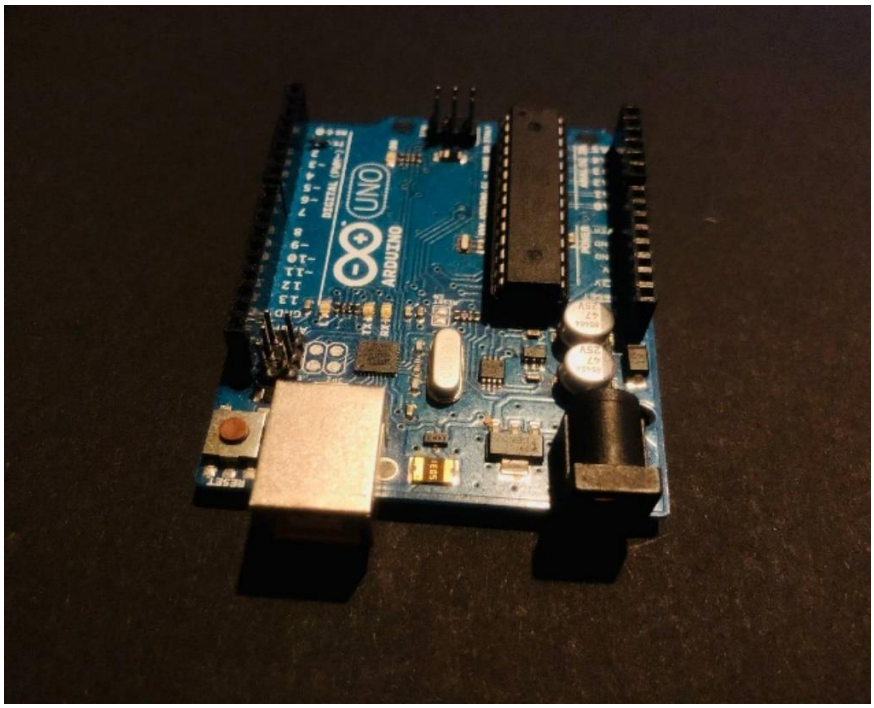


Figure 8. Arduino Uno

2.8.2 Arduino Software/Arduino IDE:

Arduino Integrated Development Environment (IDE) is a cross-platform software application to program Arduino boards. [16] The codes are written and then loaded onto the Arduino boards in the Arduino IDE.

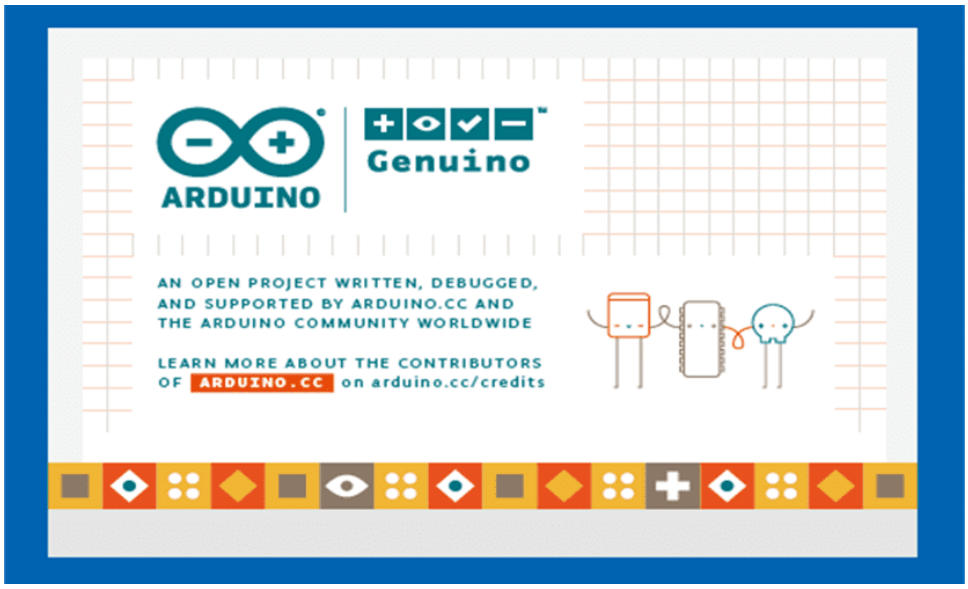


Figure 9. Arduino IDE.

Source: https://ubunlog.com/en/arduino-ide-on-ubuntu/

2.8.3 Sketch:

The unit of code written on the Arduino IDE, which is later downloaded to the Arduino board, is called Sketch. It contains setup() and loop() functions is all the arduino sketch. The setup() function is called at the beginning to do the tasks like setting pin modes and initializing libraries. The loop() function is executed continuously. [17] It controls the behaviour of the program by continuously reading inputs. These two are mandatory function for Arduino programming.

3 Methodology

3.1 Method

The project aims to build an Arduino-based medicine dispenser that dispenses medicine at a specific time, generates an alarm, and sends alert messages. To keep track of the project and achieve the project's objectives, it is imperative to use project management methodologies. Project management methodologies guide principles and processes to plan and manage any projects. [18] The waterfall methodology is remarkably effective for the project Arduino-based medicine dispenser.

With the waterfall, the project is broken into five discrete phases: requirements gathering, design, implementation, verification, and maintenance, with each new stage starting only when the previous step is completed. [19]

Figure 10. Waterfall Methodology.

Source: https://managementhelp.org/waterfall-methodology

3.2 Procedure

3.2.1 Requirement and Planning

In waterfall methodology, it is a crucial stage. This phase involves gathering information and project requirements before proceeding to the next phase of designing the medicine dispenser. Since the project's objectives had already been defined, all the hardware and software components were identified based on their applications.

Hardware Requirements

- Arduino Uno
- GSM Module
- RTC Module
- Ultrasonic Sensor
- Servo motor
- RGB LED
- Power supply
- Cables

Software Requirements

- Arduino IDE
- Tinkercad

3.2.2 Design

This phase used the requirements gathered to develop a comprehensive and detailed design for the medicine dispenser. Using Tinkercad, two designs were generated, and only one suitable design concept (Figure 11) was selected. After that, the electronic circuit of the medicine dispenser was simulated and tested. The mechanism for the dispense was also designed.

Figure 11. First design for the medicine dispenser

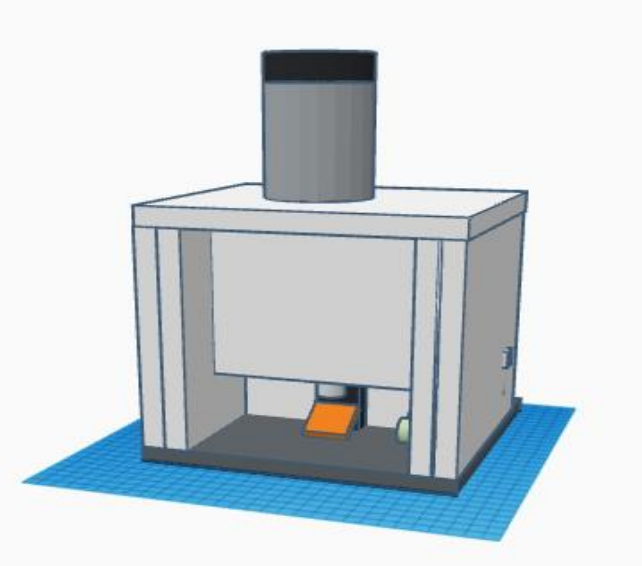


Figure 12. Second design for the medicine dispenser

3.2.3 Implementation

Once the design phase was completed, the implementation phase involved building the medicine dispenser with hardware components and developing the software component. Arduino was connected to the computer and programmed using Arduino IDE.

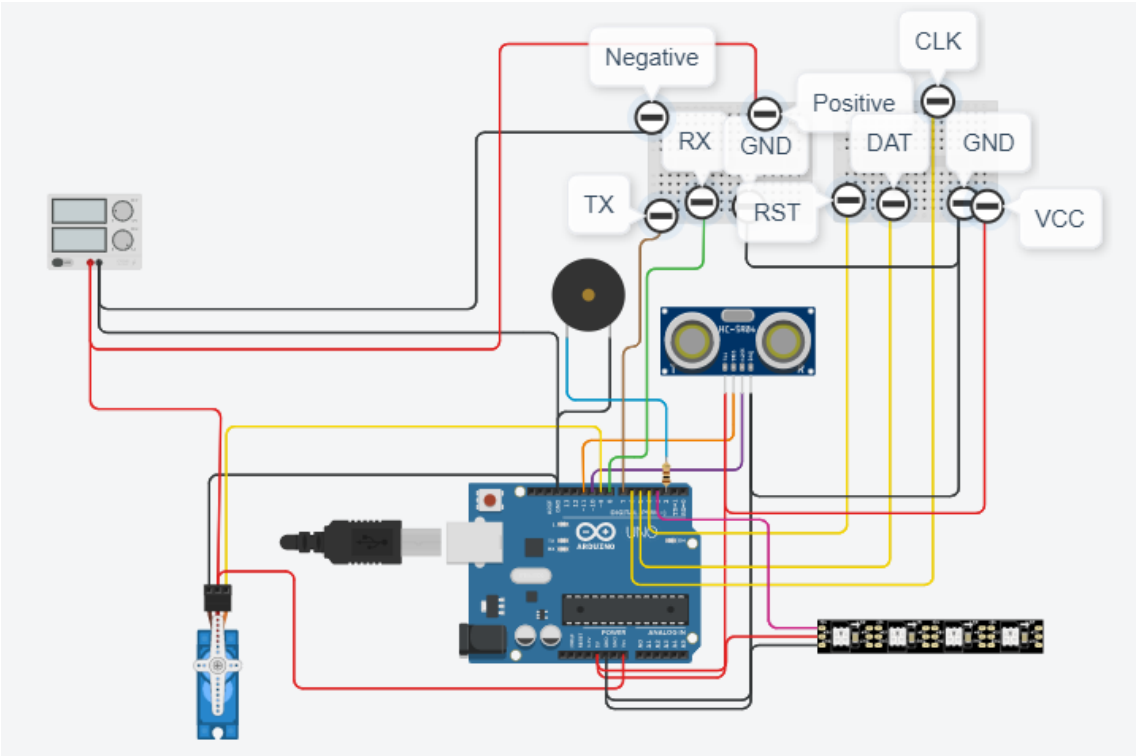


Figure 13. Virtual Circuit connection for medicine dispenser

As can be seen from figures 13 and 14, servo's control pin was connected to the PWM enabled pin 9 of the Arduino. Similarly, RGB LED model RGB WS2812B was connected to pin 3 of Arduino. Likewise, piezo buzzer was connected to pin 2 and pins 4, 5 and 6 were used for RTC module. Similarly, TX & RX of the GSM SIM900A module were connected to pin 7 and 8 of Arduino respectively. Finally, pins 10 and 11 of the Arduino were connected to ultrasonic sensor.

Arduino UNO	GSM Module	Servo	Buzzer	HC-SR04	RGB Led	RTC	Power
GND	GND	GND	Negative	GND	GND	GND	Negative
5V				VCC	5V	VCC	
VIN							Positive
2			Buzzer positive				
3					In		
4						RST	
5						DAT	
6						CLK	
7	TX						
8	RX						
9		Signal					
10				ECHO			
11				TRIG			
	Negative						Negative
	Positive	Positive					Positive

Figure 14. Pin Schematics

```
String f1001 = "+358xxxxxxxxx";  // cell number to which you want to send the security alert message
#include <Servo.h>                // Include servo library to run the Servo motor
#include <virtuabotixRTC.h>       // Real time clock library in order to store clock time
#include <SoftwareSerial.h>       // External Serial communication library
#include <FastLED.h>              // Library for controlling RGB WS2812B leds
#define LED_PIN     3             // RGB signal pin
#define NUM_LEDS    4             // Number of RGB leds used
SoftwareSerial SIM900(7, 8);  //TX & RX of the GSM SIM900A module
virtuabotixRTC myRTC(6, 5, 4);  //CLK,DAT,RST
String textForSMS;                // Declares the string for storing the text message to send
int minutes, hours, seconds;      // declare variables to store the current time from RTC
int fixminutes = 40;              // The fixed minute on which the medicine will be dispensed
int fixhours = 14;                // The fixed hours on which the medicine will be dispensed
int fixseconds = 1;               // The fixed seconds on which the medicine will be dispensed (1 is default)
int passedminute = 0;             // Store variable to put the delay time of taking medicine from the dispenser
int buzzer = 2;                   // Buzzer is connected on arduino pin no. 2
int rgbdif = 0, rgbmillis = 0;    // Declare variables in order to control the Color of RGB WS2812B light
int i = 0, d = 0;                 // Declare variables to run the for loops


const int trig_pin = 11;          //Ultrasonic sensor HC-SR04 TRIG Pin to Arduino pin 11
const int echo_pin = 10;          //Ultrasonic sensor HC-SR04 ECHO pin to Arduino pin 10
long distance, duration;          //Store variables to control the alarm sytems in the loop
int previous = 0, current, di#;  //Store variables to control the alarm sytems in the loop
boolean status1 = true, buzzerstatus = true, rgbstatus = false;   //  Store booleans (true/false) to control the loop
Servo myservo;                    // Declare the servo name from Servo.h library to control specific servo in the loop

CRGB leds[NUM_LEDS];              // Define how many RGB LEDs are used ( 4 )

```

Figure 15. Code snippet I

As can be seen from figure 15, all the necessary libraries were included such as servo.h to run servo motor, virtuabotixRTC.h to store clock time, SoftwareSerial.h for

external serial communication and FastLED.h for controlling RGB WS2812B. All the necessary pins were defined for controlling RGB WS2812B, GSM Module, RTC Module, Ultrasonic Sensor, and Buzzer. Time to dispense medicine was hardcoded to 14 hours and 40 minutes. A string was declared for storing the text message to send to the user. Different Boolean values are set, such as status1, buzzer status and RGB status which denotes the value true or false for status of medicine dispensed, status of buzzer and status of RGB respectively.

```
void setup() {                                  // The loop to run once
  myRTC.setDS1302Time(18, 10, 10, 7, 18, 9, 2022);   // Set time & date(seconds, minutes, hours, day of the week, day of the month, month, year)
  pinMode(trig_pin, OUTPUT);                    // Declare ultrasonic sensor HC-SR04 as an OUTPUT to give 5V power supply to the pin
  pinMode(echo_pin, INPUT);                     // Declare ultrasonic sensor HC-SR04 as a INPUT to sense the Reflected wave
  myservo.attach(9);                            // Servo motor is attatched to pin no. 9
  myservo.write(i);                             // Set servo position to 0

  Serial.begin(9600);                           // Start Serial monitor with 9600 baud rate (optional)
  FastLED.addLeds<WS2812, LED_PIN, GRB>(leds, NUM_LEDS); // Trigger the drivers on the WS2812B RGB led strip
  pinMode(buzzer, OUTPUT);                        // Declare the buzzer pin as an OUTPUT to give 5V through the pin
  SIM900.begin(9600);                             // Start the GSM module with Fixed baud rate ( To perform the send message action )
  Serial.println(" logging time completed!");     // When GSM module started

  for (d = 0; d < NUM_LEDS; d++) {              // Light up all RGB leds in nocolor
    leds[d] = CRGB(0,0,0);
    FastLED.show();
  }
  delay(5000);                                    // Wait 5 seconds to make the GSM module prepared
  textForSMS =  "\n The Dispenser Bot has started";   // Notify that the bot has got power

  sendsms(textForSMS, f1001);                     //sendSMS(textForSMS)
  Serial.println(textForSMS);                     // Print the message on Serial monitor (optional)
  Serial.println("message sent.");                // Print the success message on Serial monitor (optional)
}
```

Figure 16. Code snippet II

From figure 16, current time is set for the RTC clock module. The ultrasonic pins were declared as output, trig_pin and input, echo_pin which were later used to detect if the medicine has been taken or not, by measuring the distance. A pin is defined for the servo motor, and it is set to the initial position 0.

FastLED.addLeds<WS2812, LED_PIN, GRB>(leds, NUM_LEDS)

This line of code initializes RGB driver and lets the program know how many LEDs have been used. The RGB is later used to denote if the medicine is taken or not. The buzzer pin is declared as an output pin which would later be used for alarm system in the machine.

SIM900.begin(9600);

This part of the code initializes GSM module with fixed baud rate of 9600 to communicate with the Arduino. The GSM module is used to send text messages from the dispenser machine to the phone.

All the lights are set to no colour. After the delay of 5 seconds, the textForString stores the new value which is sent via SMS with next part of code.

```
void loop() {

 myRTC.updateTime();                                  // Keep the time updated via RTC module

 minutes = myRTC.minutes;                             // Get current minutes from the RTC module
 hours = myRTC.hours;                                 // Get current hours from the RTC module
 seconds = myRTC.seconds;                             // Get the current seconds from the RTC module
 if (minutes == fixminutes && hours == fixhours && seconds == fixseconds  && status1 == true) {  // If the time = fixed time
  previous = millis();                                // Store the current milliseconds (1 second = 1000 milliseconds)


  status1 = false;                                    // Then the status1 will be 1 in order to run the loop once
  for (i = 0; i < 90; i++) {                          // Position the servo from 0 degree to 90 degrees slowly, so that the medicine stays fine

   myservo.write(i);
   delay(10);
  }
```

```
    for (d = 0; d < NUM_LEDS; d++) {                  // Light up all RGB leds in light blue color
      leds[d] = CRGB(52, 127, 247);
      FastLED.show();
    }
    textForSMS =  "\nMedicine has been dispensed";    // Send message that medecine has been dispensed
    //sendSMS(textForSMS);
    sendsms(textForSMS, f1001);
    Serial.println(textForSMS);
    Serial.println("message sent.");

  }
```

Figure 17. Code snippet III

The time is updated using RTC module, and is stored in integers minutes, hours, and seconds. In the next line, if the hardcoded 14 hours and 40 minutes have passed, the current time is saved, and the servo motor slowly turns from its initial position 0 to 90 degree to dispense the medicine. Immediately, all LEDs change colour into blue and the boolean value is set to false which prevents the if block of code to run every loop. And a new alert is sent to the phone in the form of text message notifying that the medicine has been dispensed.

```
if (status1 == false) {                               // When medicine is not take but it is dispensed already
  current = millis();                                 // Set the current milliseconds at a variable
  dif = current - previous;                           // Calculate the difference of time in milliseconds from the dispensing time till now

  if (dif > 300000 && dif < 300100) {                 // If 5 minutes (1 minute = 60000 milliseconds) has passed still the medecine is not taken yet
    for (d = 0; d < NUM_LEDS; d++) {                  // All RGB leds turn Light Red color
      leds[d] = CRGB(255, 0, 34);
      FastLED.show();
    }

    textForSMS =  "\nAlert! Medicine has not been taken yet";   // Send the Alert SMS
    //sendSMS(textForSMS);
    sendsms(textForSMS, f1001);
    Serial.println(textForSMS);
    Serial.println("message sent.");
    buzzerstatus = true;                                        // Turn the buzzer on
    delay(100);
  }
```

Figure 18. Code snippet IV

Once the medicine is dispensed the int current stores the current value of milliseconds after this the difference between current value and previous value is compared, if the value exceeds more than 5 mins all the LEDs will turn red, alert message is sent and buzzer starts to go off.

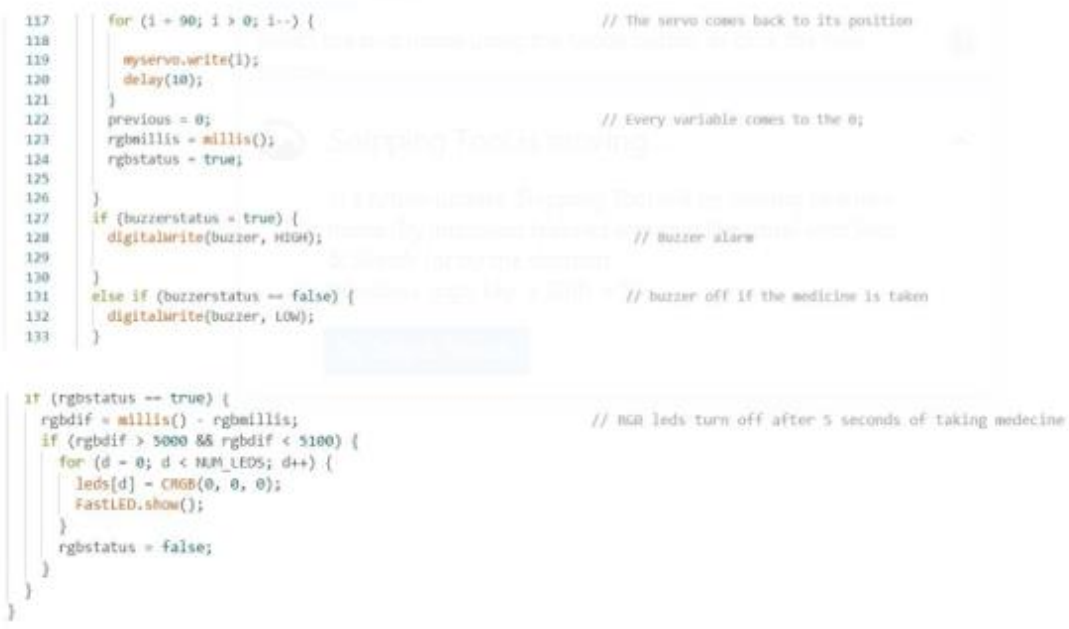
```
103     digitalWrite(trig_pin, HIGH);                    // Measure the distance of sensor to check if the medicine is being taken or not
104     delayMicroseconds(20);
105     digitalWrite(trig_pin, LOW);
106     delayMicroseconds(20);
107     duration = pulseIn(echo_pin, HIGH);
108     distance = duration * 0.034 / 2;
109   }
110   if (distance < 15 && status1 == false) {            // If the medicine is being taken
111     buzzerstatus = false;                             // Turn the buzzer off if it is on
112     status1 = true;                                   // Make the loop start from scratch (reset)
113     for (d = 0; d < NUM_LEDS; d++) {                  // Turn all RGB leds into light green color
114       leds[d] = CRGB(9, 255, 0);
115       FastLED.show();
116     }
```

Figure 19. Code snippet V

This code of block takes care of the condition where the medicine is dispensed and taken. This is where trig pin and echo pin of ultrasonic sensor sends pulse of sound and receive it. First, a pulse is sent using trig_pin and is received by echo_pin. The measured pulse duration is multiplied by 0.034 and divided by 2 to get the distance. The number 0.034 is the speed of sound in centimetres per microsecond. The division by 2 is used to get one-way value from the round-trip measurement.

By calculating the distance, the dispenser detects if the medicine has been taken off the machine or not.

Since the distance between ultrasonic sensor and medicine bowl is 15 mm, if the distance is greater than 15mm, the machine refers the medicine has been taken, all the led turn into green light.



```
    for (i = 90; i > 0; i--) {                          // The servo comes back to its position
      myservo.write(i);
      delay(10);
    }
    previous = 0;                                       // Every variable comes to the 0;
    rgbmillis = millis();
    rgbstatus = true;

  }
  if (buzzerstatus = true) {
    digitalWrite(buzzer, HIGH);                         // Buzzer alarm

  }
  else if (buzzerstatus == false) {                     // buzzer off if the medicine is taken
    digitalWrite(buzzer, LOW);
  }

if (rgbstatus == true) {
  rgbdif = millis() - rgbmillis;                        // RGB leds turn off after 5 seconds of taking medecine
  if (rgbdif > 5000 && rgbdif < 5100) {
    for (d = 0; d < NUM_LEDS; d++) {
      leds[d] = CRGB(0, 0, 0);
      FastLED.show();
    }
    rgbstatus = false;
  }
}
}
```

Figure 20. Code snippet VI

After that, the servo comes to zero position, all values are reset, and new measurement of elapsed time is stored. Finally, the LED and the buzzer functionalities are handled.

```
void sendsms(String message, String number)                    // Fix the sms sending system via GSM module
{
  String mnumber = "AT + CMGS = \"" + number + "\"";
  SIM900.print("AT+CMGF=1\r");
  delay(100);
  SIM900.println(mnumber);  // recipient's mobile number, in international format

  delay(100);
  SIM900.println(message);                        // message to send
  delay(100);
  SIM900.println((char)26);                       // End AT command with a ^Z, ASCII code 26
  delay(100);
  SIM900.println();
  delay(100);                                    // give module time to send SMS
  // SIM900power();
}
```

Figure 21. Code snippet VII

Figure 21 defines the SMS functionality of the dispenser machine. This function is called inside loop() function whenever the GSM module is used. The function receives recipient’s phone number and text message as arguments. First, the recipient’s phone number is formatted in international format. Then, the text message is sent using println function from the included library.

3.2.4 Verification

After the implementation phase is finished, the next phase to follow is verification phase. The functionality and durability of the medicine dispenser are evaluated. This phase involves ensuring the product can meet objectives.

The virtual circuit was designed and tested in the Tinkercad platform. The RTC module was replaced by a button to trigger the time to dispense the medicine.

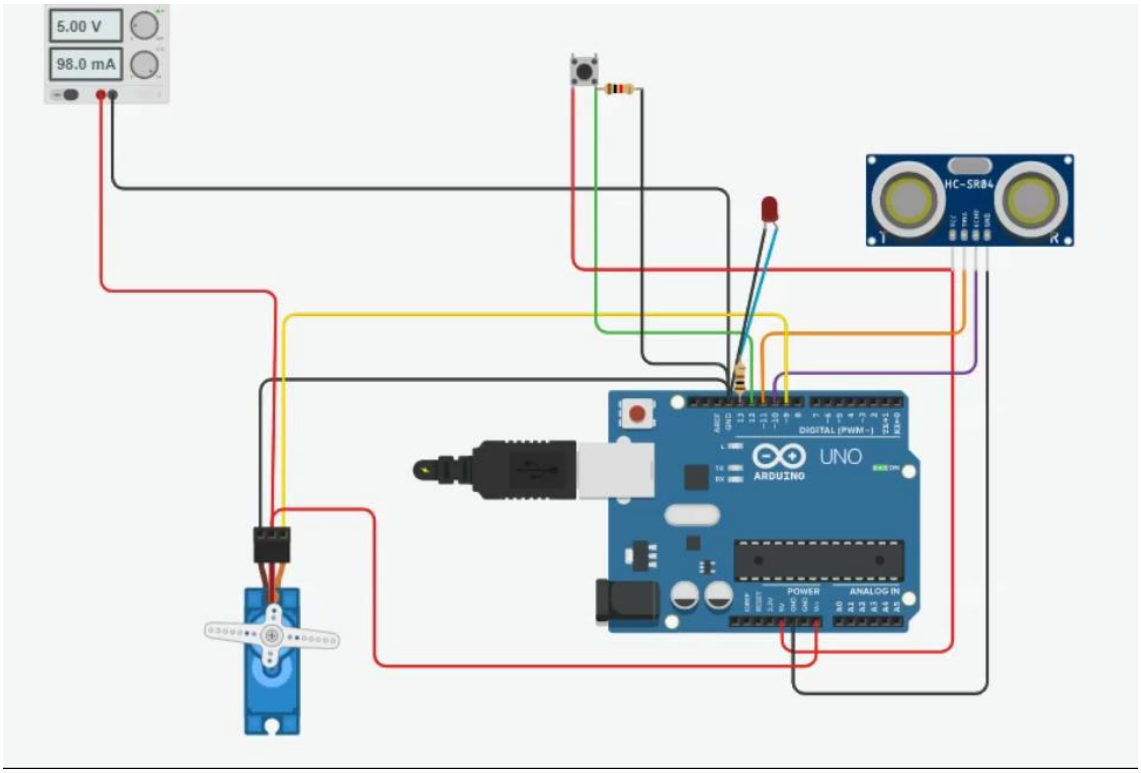


Figure 22. Virtual Circuit

3.2.5 Maintenance

This phase involved fixing any issues that came up after the medicine dispenser was designed and built. The GSM module was faulty at times.

4 Conclusion

In conclusion, the development of an Arduino based medicine dispenser was a success. The waterfall methodology helped to keep track of the project and make it successful. The main objective was to design, implement and test the prototype the dispenser that is simple, low-cost, and easy to build.

The dispenser uses an Arduino as a controller, uses a real-time clock (RTC) module for timekeeping, uses a servo motor to take medicine from the medicine container and dispense it, uses a piezo buzzer and RGB as alarm system, uses an ultrasonic sensor to detect when the pill has been taken off the box or not and sends an alert message first when the medicine is dispensed, later when the medicine is not taken for more than 5 minutes using the GSM module(Felipe & Rojas, n.d.).

For the future development, logs can be kept by connecting the dispenser machine to the cloud. Also, the dispenser machine can be made to dispense multiple medicines.

References

1. World Health Organization. (2021, February 1). Noncommunicable diseases. Retrieved from https://www.who.int/news-room/fact-sheets/detail/noncommunicable-diseases#:~:text=Key%20facts,%2D%20and%20middle%2Dincome%20countries.
2. World Health Organization. (2021, March 3). Assistive technology. Retrieved from https://www.who.int/news-room/fact-sheets/detail/assistive-technology
3. World Health Organization. (2021, February 1). Noncommunicable diseases. Retrieved from https://www.who.int/news-room/fact-sheets/detail/noncommunicable-diseases
4. NCD Alliance. (n.d.). NCDs: The basics. Retrieved from https://ncdalliance.org/why-ncds/NCDs
5. Epilepsy Research UK. (2018, January 17). Poll shows that almost 50% of people forget to take their medication at least once a month. Retrieved from https://epilepsyresearch.org.uk/poll-shows-that-almost-50-of-people-forget-to-take-their-medication-at-least-once-a-month/
6. Margaret Rouse. (2015, January). Embedded System. Retrieved from https://www.techtarget.com/iotagenda/definition/embedded-system
7. Circuit Digest. (n.d.). Servo motor: Working and basics. Retrieved from https://circuitdigest.com/article/servo-motor-working-and-basics
8. Analog IC Tips. (2016, May 10). Pulse width modulation (PWM). Retrieved from https://www.analogictips.com/pulse-width-modulation-pwm/
9. Arduino. (n.d.). Arduino GSM Shield. Retrieved from https://docs.arduino.cc/retired/shields/arduino-gsm-shield

10. Arduino Get Started. (n.d.). Arduino Ultrasonic Sensor. Retrieved from https://arduinogetstarted.com/tutorials/arduino-ultrasonic-sensor
11. Arduino Get Started. (n.d.). Arduino RGB LED. Retrieved from https://arduinogetstarted.com/tutorials/arduino-rgb-led
12. Instructables. (n.d.). How to use a Piezo Buzzer. Retrieved from https://www.instructables.com/How-to-Use-a-Piezo-Buzzer/
13. Tinkercad. (n.d.). Basics of Arduino | TINKERCAD. Retrieved from https://www.tinkercad.com/projects/Basics-of-Arduino-TINKERCAD
14. CircuitsToday. (2015, March 22). Story and history of development of Arduino. Retrieved from https://www.circuitstoday.com/story-and-history-of-development-of-arduino
15. EDUCBA. (n.d.). What is Arduino UNO? Retrieved from https://www.educba.com/what-is-arduino-uno/
16. Arduino. (n.d.). Arduino IDE v1 Basics. Retrieved from https://docs.arduino.cc/software/ide-v1/tutorials/arduino-ide-v1-basics
17. Arduino. (n.d.). Sketches. Retrieved from https://docs.arduino.cc/learn/programming/sketches
18. Adobe. (2021, January 27). Project management methodologies: A beginner's guide. Retrieved from https://business.adobe.com/blog/basics/methodologies#:~:text=Project%20management%20methodologies%20are%20a,work%20is%20prioritized%20and%20completed.
19. Profit.co. (2021, April 1). The 5 phases of Waterfall project management. Retrieved from https://www.profit.co/blog/task-management/the-5-phases-of-waterfall-project-management/