

TEAM ID	PNT2022TMID23984
PROJECT NAME	Personal Assistance for Seniors Who Are Self-Reliant

Sprint 1

The AIM of Sprint 1 is SIMULATION CREATION.

Duration : 6 days

In this Sprint 1 we have found out what the elderly people used to and suffer in Alzheimer disease .So the we had decided that we should find the solution of four different Ideas from our teammates

- 1) Our teammate Ramesh P, Medicine Remainder for Alzheimer disease person to keep an instant remainder for their medicine that is taken by them and keep them on track of their medicine taken.
- 2) Our teammate Prasath V,A Medicine Remainder for elderly person that are elderly person in home and Hospital and keep them check in Medicine that are forgotten by Caretaker and Person around them.
- 3) Our teammate Praveen Kumar A, A Stock Remainder of Medicine for Elderly Person and keep the medicine in check every time if the medicine prolonged for another week and have the refilled alarm for the medicine that need to taken.
- 4) Our teammate Maria Antony B ,A Scan of medicine to zoom and see the medicine those are taken by the elderly person and keep in check of medication of elderly person. If needed the medication can be upload the App.

By considering all the ideas that are given teammate .We conclude that are of decision made by teammate is make a combine idea of Medicine Remainder App that feature of scanning of medicine , medicine remainder at correct timing, Stock remainder.

What we should complete in sprint 1?The Issue collected from user

USN 1:As a user, I want to take Medicines on time and monitor my health

USN 2:As a user, I want to take my tablets on time by voice command

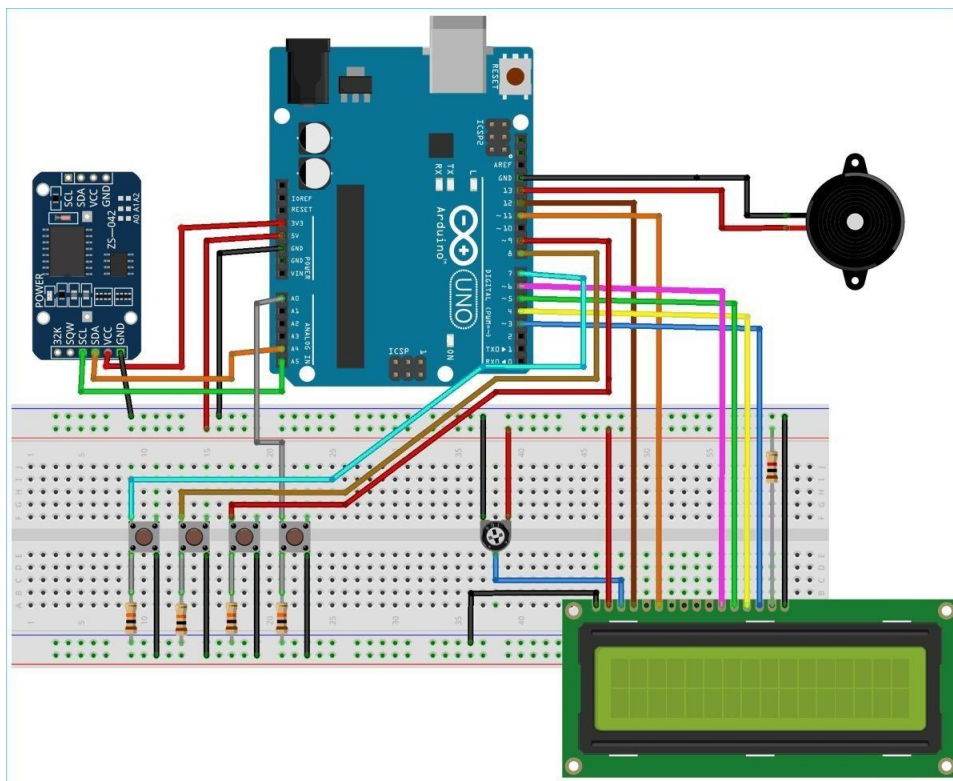
USN 3:As a user, I need to take my medicine and I am not able see the dosage of medicine properly

USN 4:As a user, Sometimes my medicine aren't in stock and I usually forget the Stock of my medication.

REQUIRED MATERIALS:

1. RTC DS3231 module
2. 16x2 LCD Display
3. Buzzer
4. Led(any color)
5. Breadboard
6. Push Buttons
7. 10K Potentiometer
8. 10K,1K Resistors
9. Jumper Wires
10. Arduino Uno

Simulation:



Code:

```
//Medicine Reminder using Arduino Uno  
  
// Reminds to take medicine at 8am, 2pm, 8pm
```

```

/* The circuit:

  LCD RS pin to digital pin 12
  LCD Enable pin to digital pin 11
  LCD D4 pin to digital pin 5
  LCD D5 pin to digital pin 4
  LCD D6 pin to digital pin 3
  LCD D7 pin to digital pin 2
  LCD R/W pin to ground
  LCD VSS pin to ground
  LCD VCC pin to 5V
  10K resistor:
    ends to +5V and ground
    wiper to LCD VO pin (pin 3)*/

#include <LiquidCrystal.h>
#include <Wire.h>
#include <RTClib.h>
#include <EEPROM.h>

int pushVal = 0;
int val;
int val2;
int addr = 0;

RTC_DS3231 rtc;

const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;           // lcd
pins

LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

#define getWellsoon 0
#define HELP_SCREEN 1
#define TIME_SCREEN 2

```

```

//bool pushPressed;                                //flag to keep track of push button
state

int pushpressed = 0;

const int ledPin =  LED_BUILTIN;                    // buzzer and led pin

int ledState = LOW;

int Signal = 0;

int buzz = 13;

int push1state, push2state, push3state, stopinState = 0;    //

int push1Flag, push2Flag, Push3Flag = false;                // push button flags

int push1pin = 9;

int push2pin = 8;

int push3pin = 7;

int stopPin = A0;

int screens = 0;                // screen to show

int maxScreen = 2;              // screen count

bool isScreenChanged = true;

long previousMillis = 0;

long interval = 500;             // buzzing interval

unsigned long currentMillis;

long previousMillisLCD = 0;      // for LCD screen update

long intervalLCD = 2000;         // Screen cycling interval

unsigned long currentMillisLCD;

//  Set Reminder Change Time

int buzz8amHH = 8;               //    HH - hours          ##Set these for reminder time in
24hr Format

int buzz8amMM = 00;              //    MM - Minute

int buzz8amSS = 00;              //    SS - Seconds

```

```

int buzz2pmHH = 14;           // HH - hours
int buzz2pmMM = 00;           // MM - Minute
int buzz2pmSS = 00;           // SS - Seconds
int buzz8pmHH = 20;           // HH - hours
int buzz8pmMM = 00;           // MM - Minute
int buzz8pmSS = 00;           // SS - Seconds

int nowHr, nowMin, nowSec;           // to show current mm,hh,ss

// All messages

void gwsMessege(){           // print get well soon messege
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Stay Healthy :)");    // Give some cheers
    lcd.setCursor(0, 1);
    lcd.print("Get Well Soon :)");    // wish
}

void helpScreen() {           // function to display 1st screen in LCD
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Press Buttons");
    lcd.setCursor(0, 1);
    lcd.print("for Reminder...!");
}

void timeScreen() {           // function to display Date and time in LCD screen
    DateTime now = rtc.now();           // take rtc time and print in display
    lcd.clear();

```

```

    lcd.setCursor(0, 0);
    lcd.print("Time:");
    lcd.setCursor(6, 0);
    lcd.print(nowHr = now.hour(), DEC);
    lcd.print(":");
    lcd.print(nowMin = now.minute(), DEC);
    lcd.print(":");
    lcd.print(nowSec = now.second(), DEC);
    lcd.setCursor(0, 1);
    lcd.print("Date: ");
    lcd.print(now.day(), DEC);
    lcd.print("/");
    lcd.print(now.month(), DEC);
    lcd.print("/");
    lcd.print(now.year(), DEC);
}

void setup() {
    Serial.begin(9600);                // start serial debugging
    if (! rtc.begin()) {                // check if rtc is connected
        Serial.println("Couldn't find RTC");
        while (1);
    }
    if (rtc.lostPower()) {
        Serial.println("RTC lost power, lets set the time!");
    }

    //    rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));           // uncomment this to
    set the current time and then comment in next upload when u set the time

```

```

rtc.adjust(DateTime(2019, 1, 10, 7, 59, 30));           // manual time set

lcd.begin(16, 2);

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Welcome To");                               // print a messege at
startup

lcd.setCursor(0, 1);

lcd.print("Circuit Digest");

delay(1000);

pinMode(push1pin, INPUT);                             // define push button
pins type

pinMode(push2pin, INPUT);

pinMode(push3pin, INPUT);

pinMode(stopPin, INPUT);

pinMode(ledPin, OUTPUT);

delay(200);

Serial.println(EEPROM.read(addr));

val2 = EEPROM.read(addr);                             // read previosuly saved value of
push button to start from where it was left previously

switch (val2) {

    case 1:

        Serial.println("Set for 1/day");

        push1state = 1;

        push2state = 0;

        push3state = 0;

        pushVal = 1;

        break;

    case 2:

```

```

    Serial.println("Set for 2/day");

    push1state = 0;
    push2state = 1;
    push3state = 0;
    pushVal = 2;
    break;
case 3:
    Serial.println("Set for 3/day");
    push1state = 0;
    push2state = 0;
    push3state = 1;
    pushVal = 3;
    break;
}

}

void loop() {
    push1(); //call to set once/day
    push2(); //call to set twice/day
    push3(); //call to set thrice/day

    if (pushVal == 1) { // if push button 1 pressed
    then remind at 8am

        at8am(); //function to start uzzing at
8am

    }

    else if (pushVal == 2) { // if push button 2 pressed
    then remind at 8am and 8pm

        at8am();

```



```

    at8pm(); //function to start uzzing at
8mm

}

else if (pushVal == 3) { // if push button 3 pressed
then remind at 8am and 8pm

    at8am();

    at2pm(); //function to start uzzing at
8mm

    at8pm();

}

currentMillisLCD = millis(); // start millis for LCD screen
switching at defined interval of time

push1state = digitalRead(push1pin); // start reading all push
button pins

push2state = digitalRead(push2pin);

push3state = digitalRead(push3pin);

stopinState = digitalRead(stopPin);

stopPins(); // call to stop buzzing

changeScreen(); // screen cycle function

}

// push buttons

void push1() { // function to set reminder once/day

    if (push1state == 1) {

        push1state = 0;

        push2state = 0;

        push3state = 0;

        // pushPressed = true;

```

```

EEPROM.write(addr, 1);

Serial.print("Push1 Written : "); Serial.println(EEPROM.read(addr)); // for
debugging

    pushVal = 1;                                //save the state of push
button-1

    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Reminder set ");
    lcd.setCursor(0, 1);
    lcd.print("for Once/day !");
    delay(1200);
    lcd.clear();
}
}

void push2() {                                //function to set reminder twice/day
    if (push2state == 1) {
        push2state = 0;
        push1state = 0;
        push3state = 0;
//    pushPressed = true;
        EEPROM.write(addr, 2);
        Serial.print("Push2 Written : "); Serial.println(EEPROM.read(addr));
        pushVal = 2;
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Reminder set ");
        lcd.setCursor(0, 1);
        lcd.print("for Twice/day !");
    }
}

```

```

    delay(1200);
    lcd.clear();
}
}

void push3() {                                //function to set reminder thrice/day
    if (push3state == 1) {
        push3state = 0;
        push1state = 0;
        push2state = 0;
//    pushPressed = true;
        EEPROM.write(addr, 3);
        Serial.print("Push3 Written : "); Serial.println(EEPROM.read(addr));
        pushVal = 3;
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Reminder set ");
        lcd.setCursor(0, 1);
        lcd.print("for Thrice/day !");
        delay(1200);
        lcd.clear();
    }
}

void stopPins() {                            //function to stop buzzing when user pushes stop
push button
    if (stopinState == 1) {
//    stopinState = 0;
//    pushPressed = true;

```

```

    pushpressed = 1;
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Take Medicine ");
    lcd.setCursor(0, 1);
    lcd.print("with Warm Water");
    delay(1200);
    lcd.clear();
}
}

void startBuzz() {                                // function to start buzzing when time reaches
to defined interval

// if (pushPressed == false) {
    if (pushpressed == 0) {
        Serial.println("pushpressed is false in blink");
        unsigned long currentMillis = millis();
        if (currentMillis - previousMillis >= interval) {
            previousMillis = currentMillis;          // save the last time you blinked the
LED
            Serial.println("Start Buzzing");
            if (ledState == LOW) {                    // if the LED is off turn it on and
vice-versa:
                ledState = HIGH;
            } else {
                ledState = LOW;
            }
            digitalWrite(ledPin, ledState);
        }
    }
}

```

```

}

else if (pushpressed == 1) {

    Serial.println("pushpressed is true");

    ledState = LOW;

    digitalWrite(ledPin, ledState);

}

}

void at8am() {                                // function to start buzzing at 8am

    DateTime now = rtc.now();

    if (int(now.hour()) >= buzz8amHH) {

        if (int(now.minute()) >= buzz8amMM) {

            if (int(now.second()) > buzz8amSS) {

                //////////////////////////////////////

                startBuzz();

                //////////////////////////////////////

            }

        }

    }

}

void at2pm() {                                // function to start buzzing at 2pm

    DateTime now = rtc.now();

    if (int(now.hour()) >= buzz2pmHH) {

        if (int(now.minute()) >= buzz2pmMM) {

            if (int(now.second()) > buzz2pmSS) {

                //////////////////////////////////////

                startBuzz();

            }

        }

    }

}

```

```

////////////////////////////////////////
}
}
}
}

void at8pm() {                                // function to start buzzing at 8pm

    DateTime now = rtc.now();

    if (int(now.hour()) >= buzz8pmHH) {

        if (int(now.minute()) >= buzz8pmMM) {

            if (int(now.second()) > buzz8pmSS) {

                //////////////////////////////////////////

                startBuzz();

                //////////////////////////////////////////

            }

        }

    }

}

//Screen Cycling

void changeScreen() {                        //function for Screen Cycling

    // Start switching screen every defined intervalLCD

    if (currentMillisLCD - previousMillisLCD > intervalLCD)                // save the
last time you changed the display

    {

        previousMillisLCD = currentMillisLCD;

        screens++;

        if (screens > maxScreen) {

```

```

        screens = 0; // all screens over -> start from 1st
    }
    isScreenChanged = true;
}
// Start displaying current screen
if (isScreenChanged) // only update the screen if the screen is changed.
{
    isScreenChanged = false; // reset for next iteration
    switch (screens)
    {
        case getWellsoon:
            gwsMessege(); // get well soon message
            break;
        case HELP_SCREEN:
            helpScreen(); // instruction screen
            break;
        case TIME_SCREEN:
            timeScreen(); // to print date and time
            break;
        default:
            //NOT SET.
            break;
    }
}
}
}

```

The **conclusion** of Sprint 1 we have analysed all the problems that are faced by the elderly people and created the dry code simulation that are given by ideas by our teammates.