SMART GLASS

Team Members:

1. **Soumya Kollipara(Super-Coordinator)**, EE18B054, Electrical Engineering, ee18b054@smail.iitm.ac.in
2. **Buddhavarapu Venkata Surya Sudheendra(Super-Coordinator),** CS18B006, Computer Science, cs18b006@smail.iitm.ac.in
3. Sowbarnika S**(Coordinator)**, CE19B024, Civil Engineering, ce19b024@smail.iitm.ac.in
4. Charulatha K**(Coordinator)**, CE19B032, Civil Engineering, ce19b032@smail.iitm.ac.in
5. M Jaswanth Kumar**(Coordinator)**, ED19B017, Engineering Design, ed19b017@smail.iitm.ac.in

INTRODUCTION :

Wearable technology represents the next level of advancement.The intention is to effortlessly enter into the day to day lives of individuals and become a useful part of them. The devices go along seamlessly and using them is easy. Wearable computing devices are advantageous because of their hands free nature.

Smart- Glasses are the wearable computing device used as an augmentation which can be fixed to the eyeglasses of the user, and can be paired with Smart Phones via Bluetooth. This eyewear has the ability to merge what you see in the real world with virtual information, usually superimposed on one of the glasses lenses.

BRIEF EXPLANATION :

In this project we created a Smart Glass also known as Wearable Computer Glasses which displays various features on a small OLED screen attached to the spectacles. The project works with an **Application** (Smart glass app-created using MIT App Inventor 2) installed on the smartphone which helps in pairing the Smart Glass and the Smart Phones via Bluetooth with the help of Bluetooth module inserted in the frame of Smart glass. **Arduino UNO**, microprocessor used here, controls the whole setup according to the code for smart glass and displays data on the OLED screen. The Arduino is powered using a 9V battery. The **Features** that the Smart glass displays via this mechanism is:

- **Date and Time**
- **Battery percentage**

- **Caller details (Phone no.)**

- **No. of Steps walked**

- **Temperature**
- **Humidity**

- **Social Distancing (gives warning when a person is within 1m radius)**

MECHANICAL MODULE :

This module was quite simple. It is made of plastic and is used as a frame to fit all the electrical modules. This module was attached to the side of any glass of interest. It also contains a square prism made of glass and mirror to reflect the letters displayed on the oled screen to the front of the glass.

ELECTRONIC MODULE :

Arduino Uno (1):

Arduino Uno is the microprocessor used to control the entire setup with the code uploaded in it.

OLED display module (1):

It is a 0.96 inch I2C Oled used to display the feature details of the Smart Glass.

PIR Sensor (1):

It is a Proximity sensor which detects a human body and sends a ‘HIGH’

signal when detected and ‘LOW’ otherwise.

Push button (1):

It is used to switch between different pages on Oled.

Battery (1):

We use a 9V battery to power the Arduino Uno which in turn powers all the remaining components.

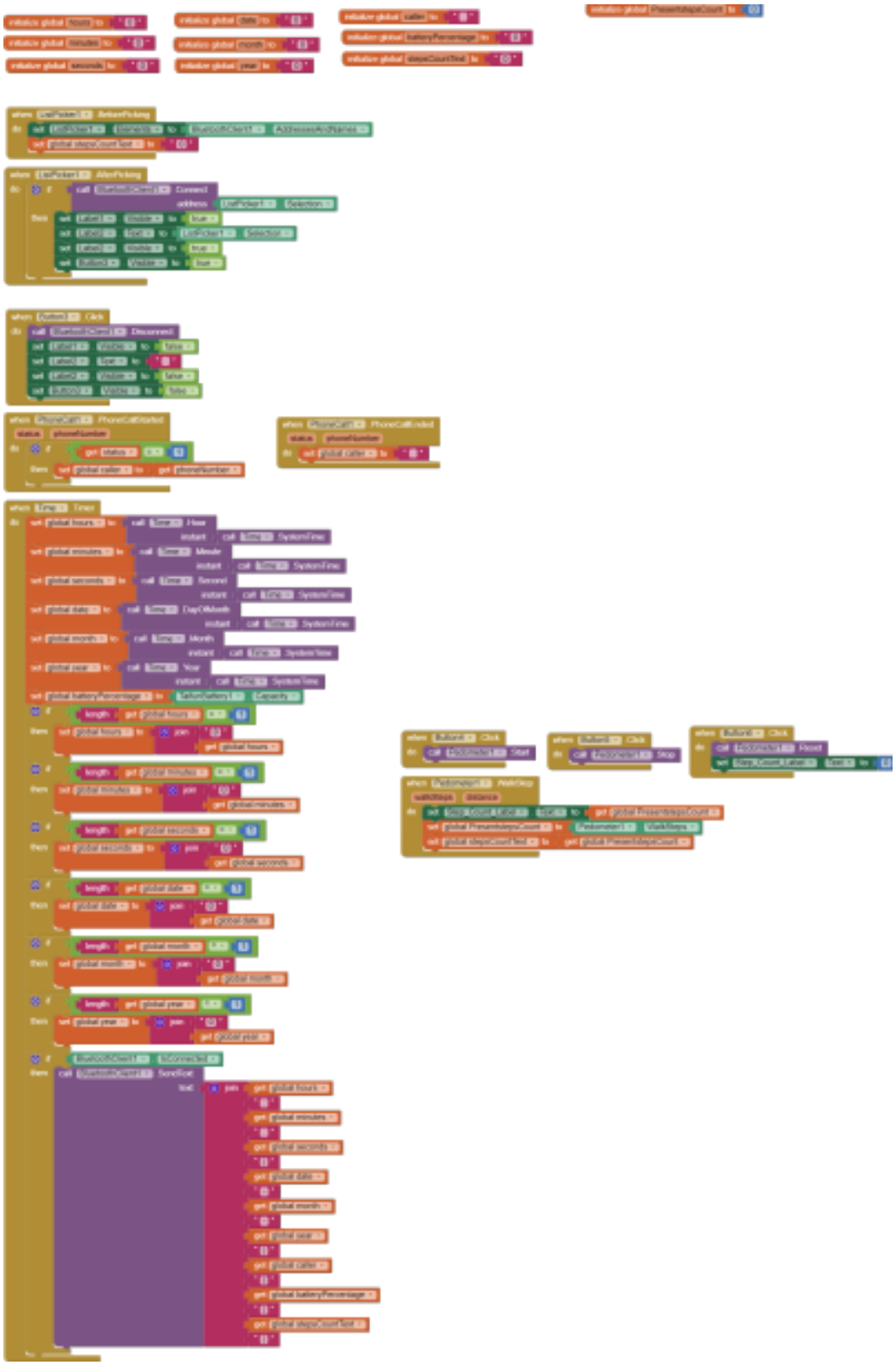
DHT11 (1):

This sensor is used to detect the temperature and humidity.

HC-05 (1):

It’s a Bluetooth module to connect the Smart Phone and the Smart Glass.

SMART GLASS APP :



CODE EXPLANATION :

This code was done using MIT App Inventor which is an Online tool for making apps. This app sends all the datas mentioned in the Brief Explanation in the order Hour, Minute, Second, Day of Month, Month, Year, Caller(if a person is calling else NULL), Battery Percentage, No. of Steps separated with '|' to separate each data. The sensor that must be present in the phone is a Pedometer.

ARDUINO CODE :

```
#include <Adafruit_Sensor.h>
#include<SoftwareSerial.h>

#include <dht.h>

#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64

#define OLED_RESET -1
#define SCREEN_ADDRESS 0x3C
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire,
OLED_RESET);

dht DHT;
#define DHT11_PIN 7

SoftwareSerial mySerial(0, 1); // RX, TX

int page = 1;
const int buttonPin = 12;
boolean buttonState = LOW;

const int PIRPin = 13;

float humidity;
```

```
float temperature;
void setup()
 {
  pinMode(buttonPin, INPUT);
  pinMode(PIRPin, INPUT);
  mySerial.begin(9600);
  if(!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS))
   {
    mySerial.println(F("SSD1306 allocation failed"));
    for(;;); // Don't proceed, loop forever
  }
}

void loop()
 {
  int chk = DHT.read11(DHT11_PIN);
  if(debounceButton(buttonState) == HIGH && buttonState == LOW)
   {
    if(page == 3)
      page = 1;
    else
      page++;
   buttonState = HIGH;
  }
  else if(debounceButton(buttonState) == LOW && buttonState == HIGH)
  {
      buttonState = LOW;
  }

  while (mySerial.available()>0)
  {
   display.clearDisplay();

   display.setTextSize(1.5);
   display.setTextColor(SSD1306_WHITE); // Draw white text
   display.setCursor(0, 0);
   display.cp437(true);

   String currentTime = mySerial.readStringUntil('|');
```

```
String currentDate = mySerial.readStringUntil('|');
String caller = mySerial.readStringUntil('|');
String batteryPercentageText = "";
int batteryPercentage = 0;
int decimalMultiplier;

while(1)
 {
  char a = mySerial.read();
  if(a == '|')
    break;
  batteryPercentageText = batteryPercentageText + a;
batteryPercentage = 10 * batteryPercentage + a - '0';
}

String stepsCount = mySerial.readStringUntil('|');

if(caller != "")
 {
  display.setTextSize(1);
  display.setCursor(0, 20);
  display.print(caller);
  display.print(" ");
  display.setCursor(0, 35);
  display.setTextSize(2);
  display.print("CALLING...");
  display.setTextSize(1);
}

else if(digitalRead(PIRPin) == HIGH)
 {
  display.setCursor(0, 35);
  display.setTextSize(1);
  display.println("WARNING!!");
  display.println("Move Away");
}

else
 {
```

```
      if(page == 1)
      {
       display.setCursor(0, 0);
       display.setTextSize(1);
       display.print(currentTime);
       display.print(" ");
       display.print(currentDate);
       drawBatteryPercentage(110, 30, batteryPercentage);
      }
      if(page == 2)
      {
       display.setCursor(30, 10);
       display.setTextSize(1);
       display.print("NO. OF STEPS");
       display.setCursor(60,30);
       display.print(stepsCount);
      }
      if(page == 3)
      {
       display.setCursor(20, 10);
       display.setTextSize(1);
       display.print("TEMP");
       display.setCursor(17,20);
       display.print(DHT.temperature);
       display.setCursor(60, 10);
       display.print("HUMIDITY");
       display.setCursor(67, 20);
       display.print(DHT.humidity);
       delay(100);
      }

    }
  }
  display.display();
  delay(100);
}


void drawBatteryPercentage(int x, int y, int batteryPercentage)
```

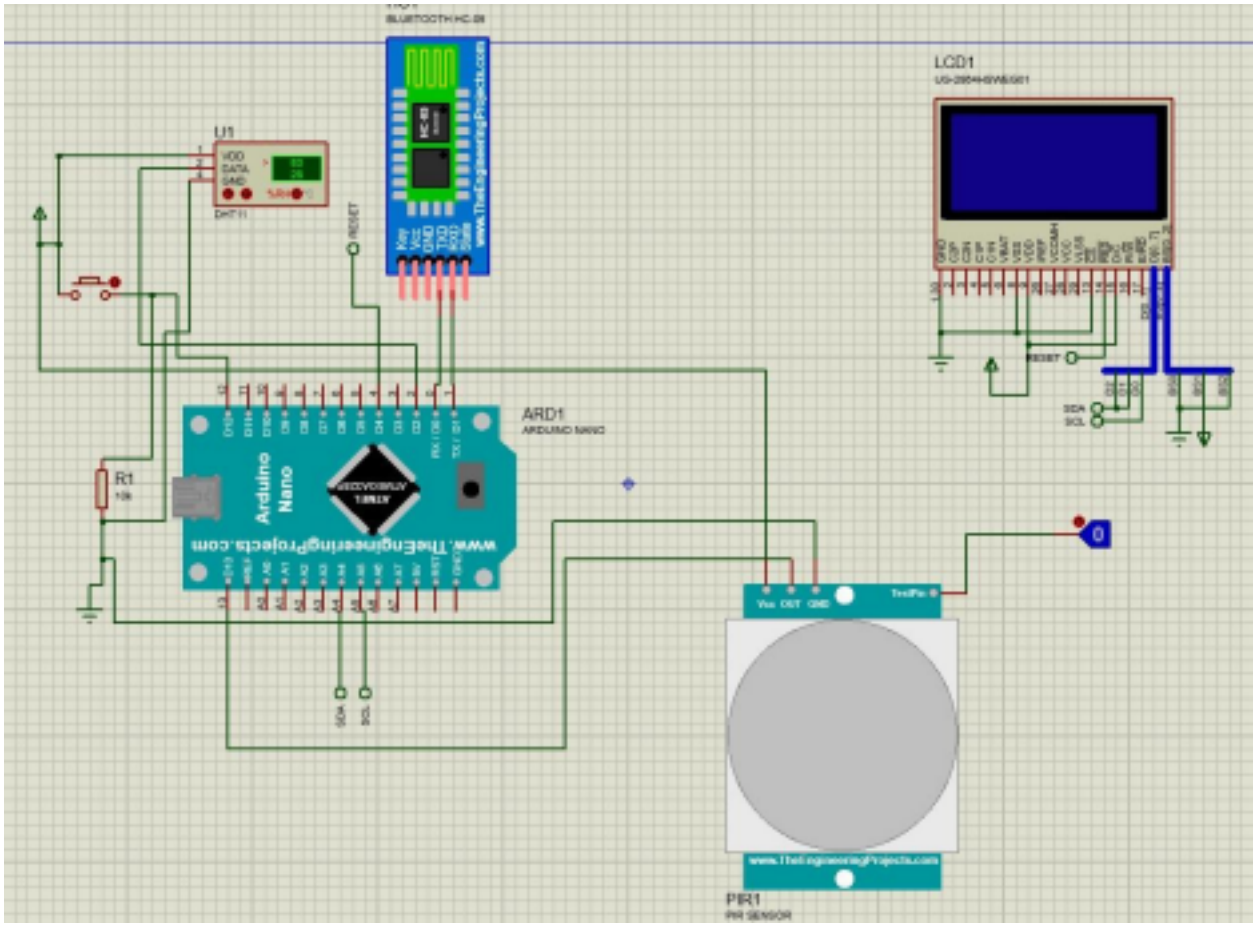
```
{
 int batteryChargeBars =0;
 batteryChargeBars = batteryPercentage/4;
 display.drawRect(0 + x, 2 + y, 14, 27, SSD1306_WHITE);
 display.drawRect(3 + x, 0 + y, 8, 3, SSD1306_WHITE);
 display.drawLine(4 + x, 2 + y, 9 + x, 2 + y, SSD1306_BLACK);

 for(int i=1; i <= batteryChargeBars; i++)
 {
  if(i==1 || i==25)
  {
   display.drawLine(2 + x, 27 - i + y, 11 + x, 27 - i + y, SSD1306_WHITE);
  }
  else
  {
   display.drawLine(2 + x, 27 - i + y, 11 + x, 27 - i + y, SSD1306_WHITE);
  }
 }
}

boolean debounceButton(boolean state)
{
 boolean stateNow = digitalRead(buttonPin);
 if(state!=stateNow)
 {
  delay(10);
  stateNow = digitalRead(buttonPin);
 }
 return stateNow;
}
```

CODE EXPLANATION :

The above code is uploaded in the Arduino UNO. The Arduino gets the serial data from the Bluetooth Module which receives the serial data from the Smartphone in the order Hour, Minute, Second, Day of Month, Month, Year, Caller(if a person is calling else NULL), Battery Percentage, No. of Steps separated with '|' to separate each data. The data received from the Bluetooth Module is read using mySerial.readStringUntil('|') and updated to their respective variables. The drawBatteryPercentage() is used to display the batteryPercentage visually rather than numerically by converting 100% as maximum

charge to 25 bars for maximum charge. Each time the button is clicked it gets incremented till it reaches the maximum page which then goes to page 1. The pop up messages like the Person Calling and Social Distancing Feature are default NULL and not NULL only when the number or HIGH respectively is sent via Bluetooth Module.

CIRCUIT :



The simulation was done in software. The actual circuit will have some variations with this circuit. The Logic Input to the PIR sensor is just for simulation purposes and there is no such connection in reality.

ISSUES FACED :

1. There were differences between Proteus Simulation and Reality.
2. We were not able to run OLED using Arduino Nano so we had to use a bigger version that is Arduino Uno which was tough during the assembly phase. 3. The resolution is only 128 x 64 pixels as we were using OLED for simplicity. 4. The battery remaining in the 9V battery is unknown and the Smart Glass can stop

working unexpectedly due to insufficient charge in the Battery.

ACKNOWLEDGEMENT :

We would like to thank our cores Jeeva and Ashutosh for their support throughout the tenure. We would like to extend our thanks to all the other super coordinators who were beside us the whole time. Lastly, we would like to thank all our fellow coordinators for appreciating and helping us whenever we required Them even though it was very difficult to communicate everything online. Thanks, everyone for making it a wonderful experience working for the first time ever Online Envisage.