Project Report

Name:-Lalit Singh

Roll no:-21095068

Branch:- Ece

Semester:- V

Course:- Ec373

Abstract:-

The purpose of this project is to create an electrical system that can measure and show a human body's instantaneous temperature correctly. The technology attempts to give real-time temperature data for medical and personal usage in a world where health monitoring is critical. This end-to-end solution consists of measuring the body temperature using the suitable sensor (in this case, the LM35), processing the analog signal received using the Arduino Nano board and the relevant code, and displaying the result—the measured temperature—on the seven-segment display. Proteus software is used to conduct all of the simulations.

Discussions:-

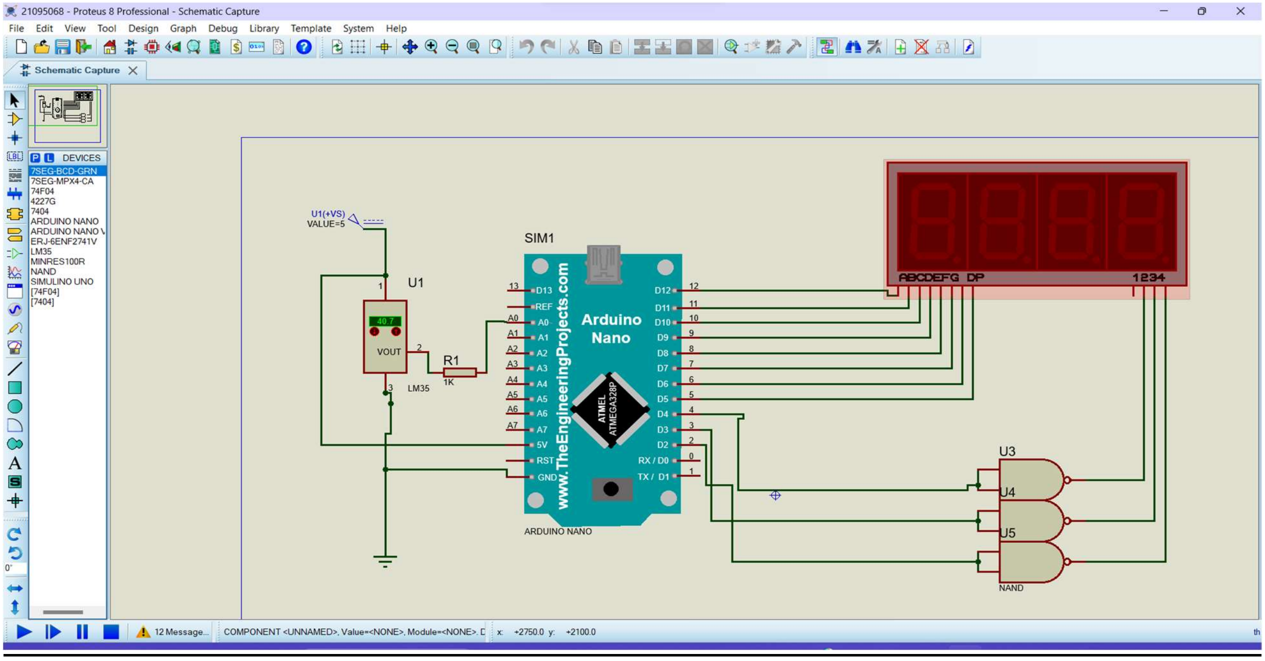
The temperature sensor I used in the system, the LM35, has a theoretical linear connection; for every degree Celsius that the temperature rises, its output increases by 10 milivolts. But when I used the aforementioned connection to take different measures, I discovered that the results weren't quite linear. After experimenting with several factor values, I discovered that a factor of 9.3 provided an ideal linear connection between output voltage and temperature change.The next step is to combine the seven segment display with the Arduino board so that the measured temperature may be displayed when the LM35 has been set. There are 7 port pins (A, B, C, D, E, F, and G) on each 7-segment display. We needed to display three digits, so I chose a four-digit seven-segment display that can multiplex four digits using seven post pins as it was not possible to connect each digit independently to the Arduino board. Regular interrupts from the Arduino interrupt service routine module are used for the multiplexing. A 4-digit, 7-segment display receives information from a single interrupt session and displays it. The subsequent digit is shown in the subsequent session, and so on. By doing this, we can use the same 7-8 ports on the Arduino board to transmit data for all three digits.

The remaining part of the code handles the display of various digits using the 7-segment display for each digit we set the values of various ports high or low. We use the below table for our reference:

Decimal	ABCDEFG	A	B	C	D	E	F	G
0	0×7E	1	1	1	1	1	1	0
1	0x30	0	1	1	0	0	0	0
2	0x6D	1	1	0	1	1	0	1
3	0x79	1	1	1	1	0	0	1
4	0x33	0	1	1	0	0	1	1
5	0x5B	1	0	1	1	0	1	1
6	0x5F	1	0	1	1	1	1	1
7	0x70	1	1	1	0	0	0	0
8	0x7F	1	1	1	1	1	1	1
9	0x7B	1	1	1	1	0	1	1

Along with the seven values we also have the decimal point after the second digit which is controlled by another pin. The seven segment display which we used was common anode so we inverted the values wherever necessary.

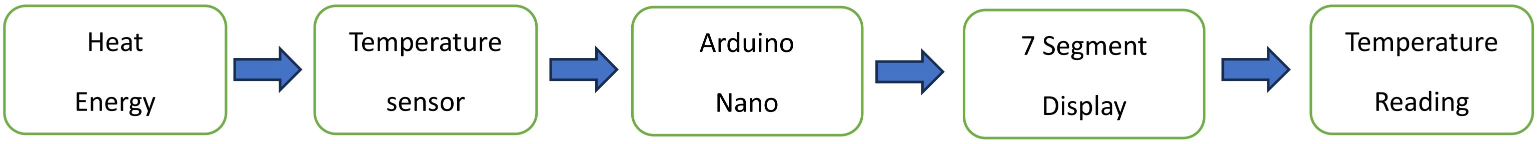
Circuit level schematic:-



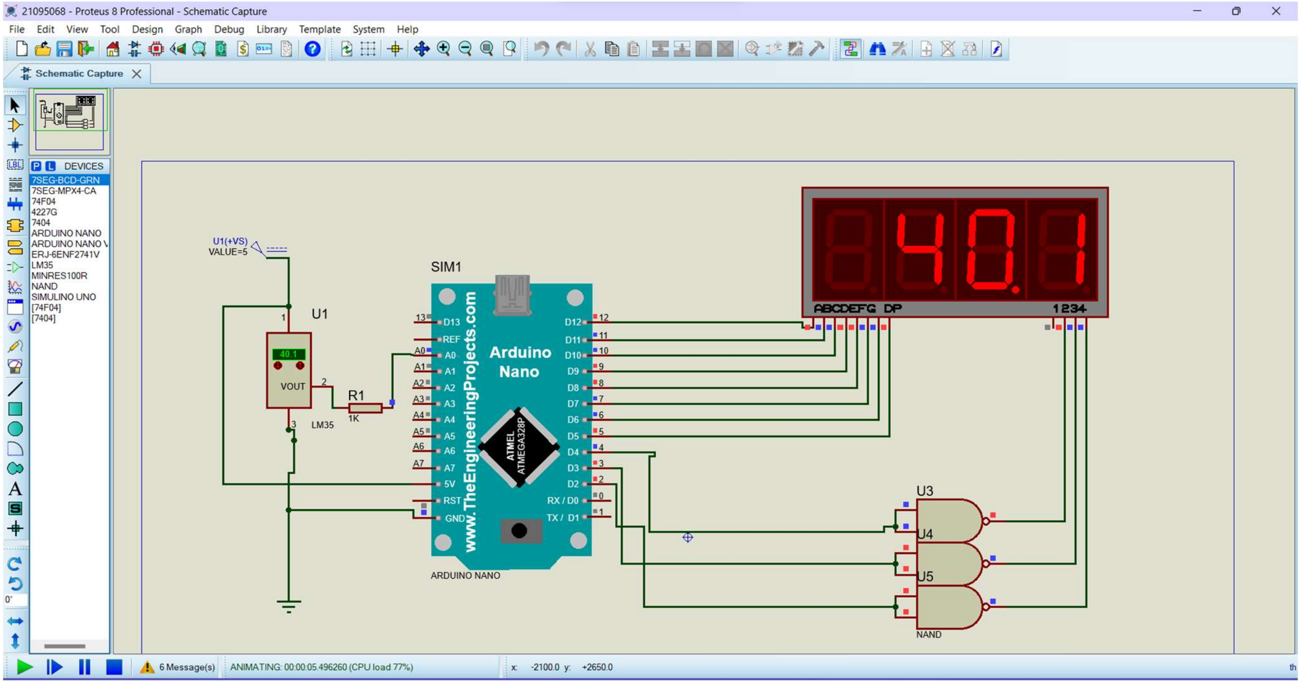
21095068 - Proteus 8 Professional - Schematic Capture

COMPONENT <UNNAMED>, Value=<NONE>, Module=<NONE>. D x: +2750.0 y: +2100.0

Block diagram:-



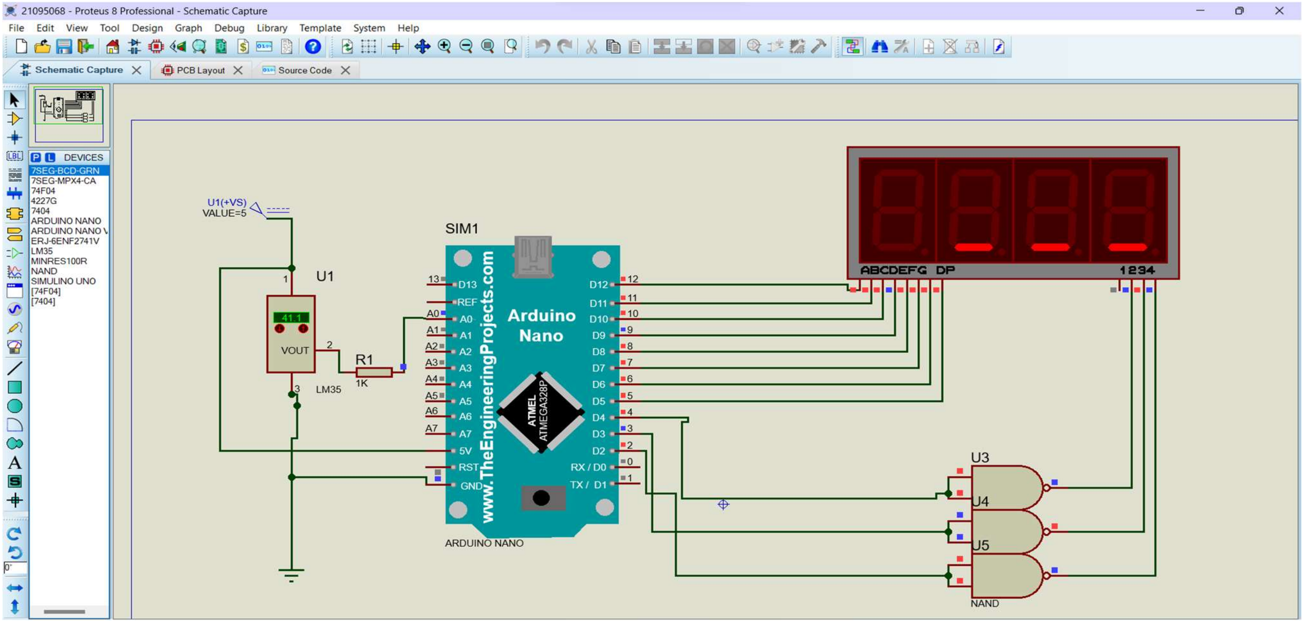
Outputs:-

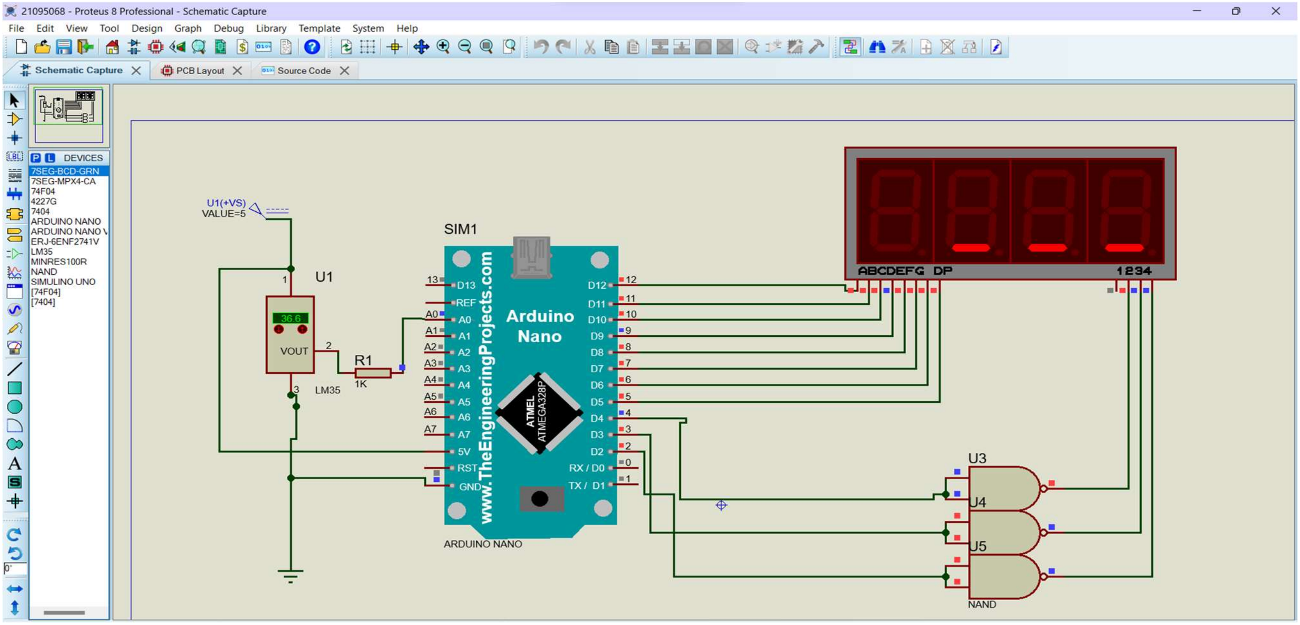


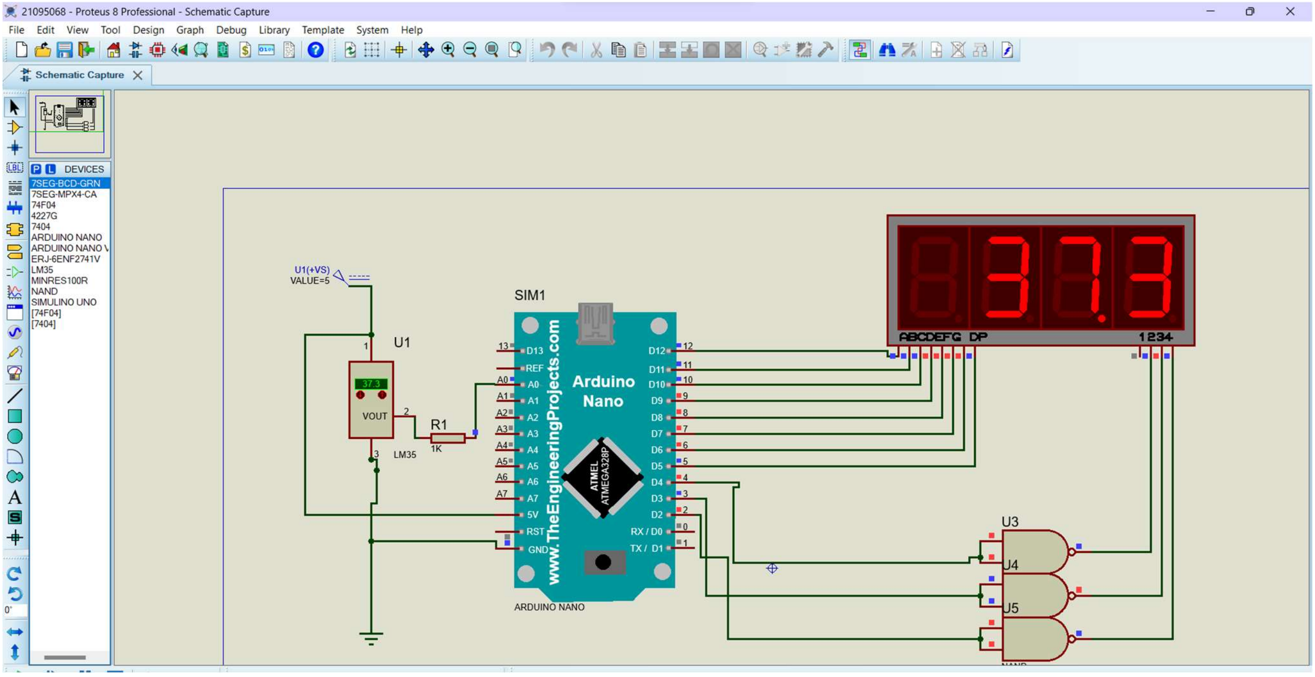
6 Message(s)

ANIMATING: 00:00:05.496260 (CPU load 77%)

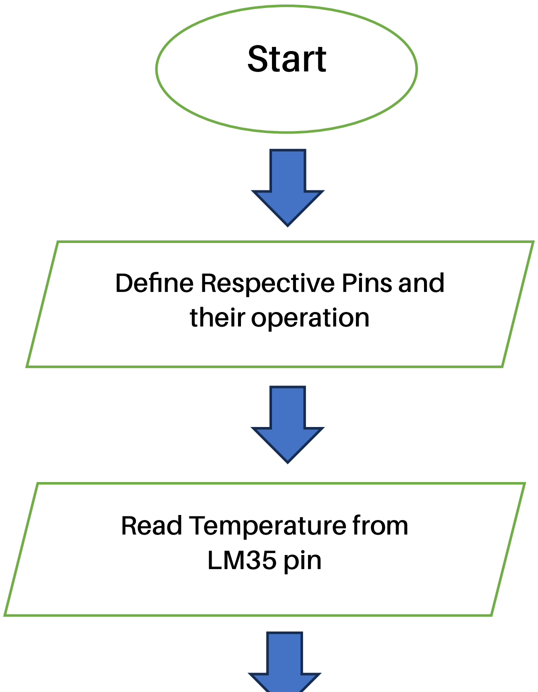
x: -2100.0 y: +2650.0





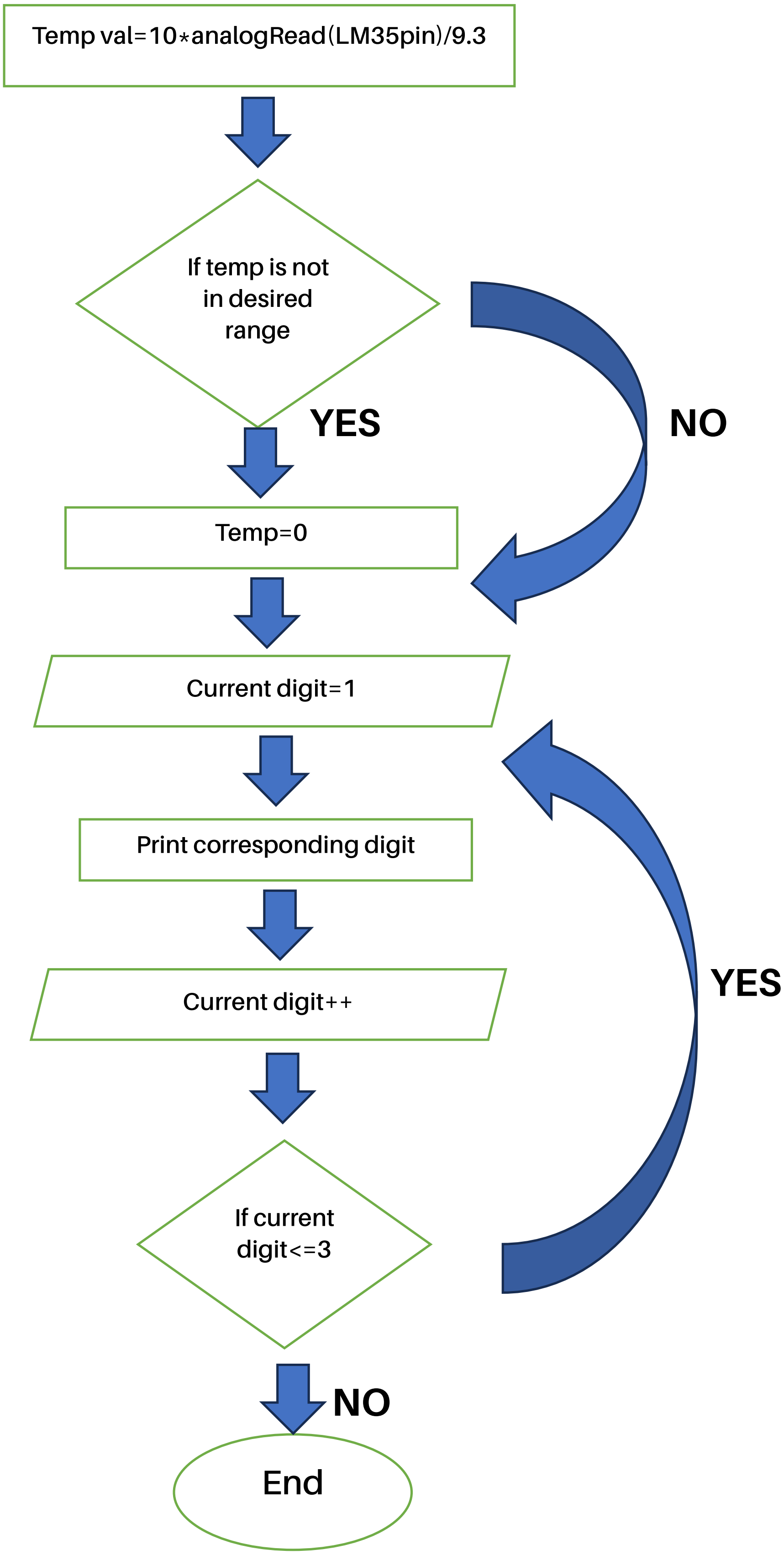


Flow Chart:-


Start

Define Respective Pins and their operation

Read Temperature from LM35 pin


Temp val=10*analogRead(LM35pin)/9.3
If temp is not in desired range
YES
NO
Temp=0
Current digit=1
Print corresponding digit
Current digit++
YES
If current digit<=3
NO
End

Program code for Arduino Nano:-

```
#include<stdio.h>

#define  LM35_pin  A0
```

```
#define SegA   12
#define SegB   11
#define SegC   10
#define SegD    9
#define SegE    8
#define SegF    7
#define SegG    6
#define SegDP   5
#define Dig1    4
#define Dig2    3
#define Dig3    2
byte current_digit;
int  temp;
void setup()
{
 pinMode(SegA, OUTPUT);
 pinMode(SegB, OUTPUT);
 pinMode(SegC, OUTPUT);
 pinMode(SegD, OUTPUT);
 pinMode(SegE, OUTPUT);
 pinMode(SegF, OUTPUT);
 pinMode(SegG, OUTPUT);
 pinMode(SegDP, OUTPUT);
 pinMode(Dig1, OUTPUT);
 pinMode(Dig2, OUTPUT);
 pinMode(Dig3, OUTPUT);
 disp_off();  // turn off the display

 // Timer1 module overflow interrupt configuration
 TCCR1A = 0;
 TCCR1B = 1;  // enable Timer1 with prescaler = 1 ( 16 ticks each 1 µs)
 TCNT1  = 0;  // set Timer1 preload value to 0 (reset)
 TIMSK1 = 1;  // enable Timer1 overflow interrupt
 analogReference(INTERNAL);  // set positive reference voltage to 1.1V
}
ISR(TIMER1_OVF_vect)   // Timer1 interrupt service routine (ISR)
{
 disp_off();  // turn off the display
 switch (current_digit)
 {
  case 1:
    if(temp!=0)
    {
     disp((temp / 100) % 10);
    }
    else
    {
     digitalWrite(SegA, HIGH);
     digitalWrite(SegB, HIGH);
     digitalWrite(SegC, HIGH);
     digitalWrite(SegD, LOW);
     digitalWrite(SegE, HIGH);
     digitalWrite(SegF, HIGH);
     digitalWrite(SegG, HIGH);
     digitalWrite(SegDP, HIGH);
    }

    digitalWrite(Dig1, LOW);  // turn on digit 1
   break;
  case 2:
   if(temp!=0)
   {
    disp( (temp / 10) % 10);   // prepare to display digit 2
    digitalWrite(SegDP, LOW);  // print decimal point ( . )
   }
   else
   {
     digitalWrite(SegA, HIGH);
     digitalWrite(SegB, HIGH);
     digitalWrite(SegC, HIGH);
     digitalWrite(SegD, LOW);
     digitalWrite(SegE, HIGH);
     digitalWrite(SegF, HIGH);
     digitalWrite(SegG, HIGH);
     digitalWrite(SegDP, HIGH);
   }
   digitalWrite(Dig2, LOW);   // turn on digit 2
   break;
  case 3:
   if(temp!=0)
   {
    disp(temp % 10);   // prepare to display digit 3
   }
   else
   {
    digitalWrite(SegA, HIGH);
     digitalWrite(SegB, HIGH);
     digitalWrite(SegC, HIGH);
     digitalWrite(SegD, LOW);
     digitalWrite(SegE, HIGH);
     digitalWrite(SegF, HIGH);
     digitalWrite(SegG, HIGH);
     digitalWrite(SegDP, HIGH);
   }
   digitalWrite(Dig3, LOW);    // turn on digit 3
 }
 current_digit = (current_digit % 3) + 1;
}
```

```
// main loop
void loop()
{
 temp = 10 * analogRead(LM35_pin) / 9.3;
 temp=temp-2;
 if(temp<370 || temp>410){
  temp=0;
 }
 delay(1000);
}
void disp(byte number)
{
 switch (number)
 {
  case 0:  // print 0
   digitalWrite(SegA, LOW);
   digitalWrite(SegB, LOW);
   digitalWrite(SegC, LOW);
   digitalWrite(SegD, LOW);
   digitalWrite(SegE, LOW);
   digitalWrite(SegF, LOW);
   digitalWrite(SegG, HIGH);
   digitalWrite(SegDP, HIGH);
   break;
  case 1:  // print 1
   digitalWrite(SegA, HIGH);
   digitalWrite(SegB, LOW);
   digitalWrite(SegC, LOW);
   digitalWrite(SegD, HIGH);
   digitalWrite(SegE, HIGH);
   digitalWrite(SegF, HIGH);
   digitalWrite(SegG, HIGH);
   digitalWrite(SegDP, HIGH);
  break
case 2:  // print 2
  digitalWrite(SegA, LOW);
  digitalWrite(SegB, LOW);
  digitalWrite(SegC, HIGH);
  digitalWrite(SegD, LOW);
  digitalWrite(SegE, LOW);
  digitalWrite(SegF, HIGH);
  digitalWrite(SegG, LOW);
  digitalWrite(SegDP, HIGH);
  break;
case 3:  // print 3
  digitalWrite(SegA, LOW);
  digitalWrite(SegB, LOW);
  digitalWrite(SegC, LOW);
 digitalWrite(SegD, LOW);
  digitalWrite(SegE, HIGH);
  digitalWrite(SegF, HIGH);
  digitalWrite(SegG, LOW);
  digitalWrite(SegDP, HIGH);
  break;
case 4:  // print 4
  digitalWrite(SegA, HIGH);
  digitalWrite(SegB, LOW);
  digitalWrite(SegC, LOW);
  digitalWrite(SegD, HIGH);
  digitalWrite(SegE, HIGH);
  digitalWrite(SegF, LOW);
  digitalWrite(SegG, LOW);
  digitalWrite(SegDP, HIGH);
  break;
case 5:  // print 5
  digitalWrite(SegA, LOW);
  digitalWrite(SegB, HIGH);
  digitalWrite(SegC, LOW);
  digitalWrite(SegD, LOW);
  digitalWrite(SegE, HIGH);
  digitalWrite(SegF, LOW);
  digitalWrite(SegG, LOW);
  digitalWrite(SegDP, HIGH);
  break;
case 6:  // print 6
  digitalWrite(SegA, LOW);
  digitalWrite(SegB, HIGH);
  digitalWrite(SegC, LOW);
  digitalWrite(SegD, LOW);
  digitalWrite(SegE, LOW);
  digitalWrite(SegF, LOW);
  digitalWrite(SegG, LOW);
  digitalWrite(SegDP, HIGH);
  break;
case 7:  // print 7
  digitalWrite(SegA, LOW);
  digitalWrite(SegB, LOW);
  digitalWrite(SegC, LOW);
  digitalWrite(SegD, HIGH);
  digitalWrite(SegE, HIGH);
  digitalWrite(SegF, HIGH);
  digitalWrite(SegG, HIGH);
  digitalWrite(SegDP, HIGH);
  break
case 8:  // print 8
  digitalWrite(SegA, LOW);
  digitalWrite(SegB, LOW);
  digitalWrite(SegC, LOW);
  digitalWrite(SegD, LOW);
```

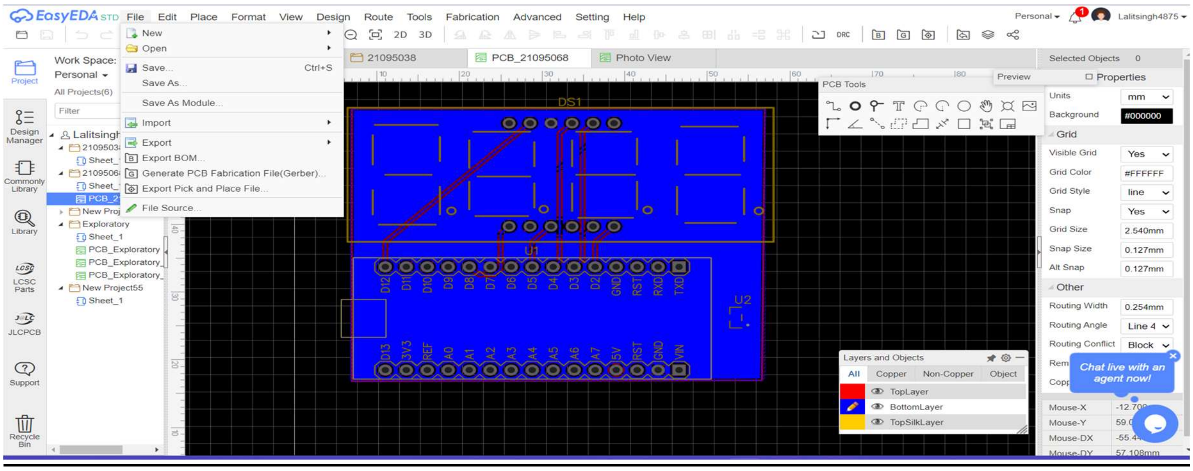
```
    digitalWrite(SegE, LOW);
    digitalWrite(SegF, LOW);
    digitalWrite(SegG, LOW);
    digitalWrite(SegDP, HIGH);
    break;
  case 9:  // print 9
    digitalWrite(SegA, LOW);
    digitalWrite(SegB, LOW);
    digitalWrite(SegC, LOW);
   digitalWrite(SegD, LOW);
    digitalWrite(SegE, HIGH);
    digitalWrite(SegF, LOW);
    digitalWrite(SegG, LOW);
    digitalWrite(SegDP, HIGH); }
}
void disp_off()
{ digitalWrite(Dig1, HIGH);
  digitalWrite(Dig2, HIGH);
  digitalWrite(Dig3, HIGH);}
```

PCB Layout:-

Dimensions:- Width:-50mm,Height:-50mm

Area:- 0.25 cm^2

Bottom Layer:-



Selected Objects 0

Properties

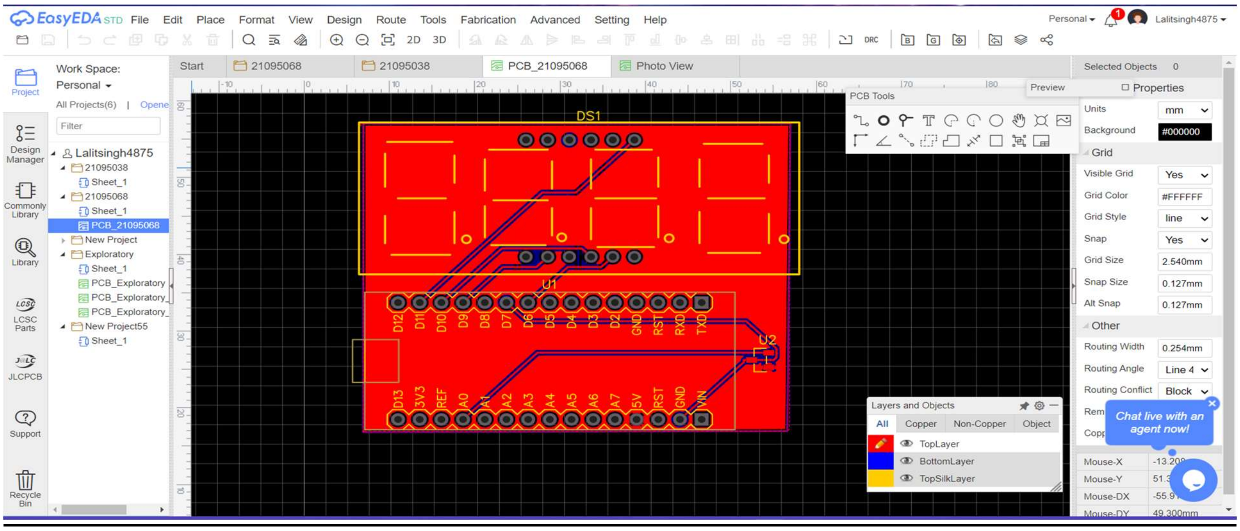
Units: mm
Background: #000000

Grid
Visible Grid: Yes
Grid Color: #FFFFFF
Grid Style: line
Snap: Yes
Grid Size: 2.540mm
Snap Size: 0.127mm
Alt Snap: 0.127mm

Other
Routing Width: 0.254mm
Routing Angle: Line 4
Routing Conflict: Block

Mouse-X: -12.700
Mouse-Y: 59.0
Mouse-DX: -55.4
Mouse-DY: 57.108mm

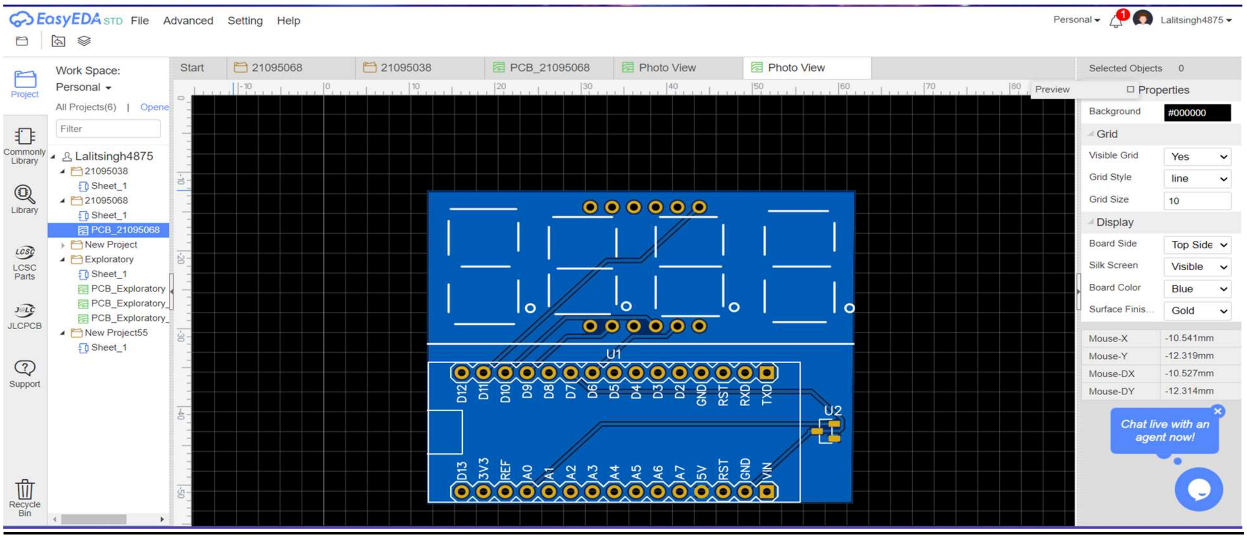
Top Layer:-



Selected Objects 0

Units	mm
Background	#000000
Grid	
Visible Grid	Yes
Grid Color	#FFFFFF
Grid Style	line
Snap	Yes
Grid Size	2.540mm
Snap Size	0.127mm
Alt Snap	0.127mm
Other	
Routing Width	0.254mm
Routing Angle	Line 4
Routing Conflict	Block
Mouse-X	-13.20
Mouse-Y	51.3
Mouse-DX	-55.9
Mouse-DY	49.300mm

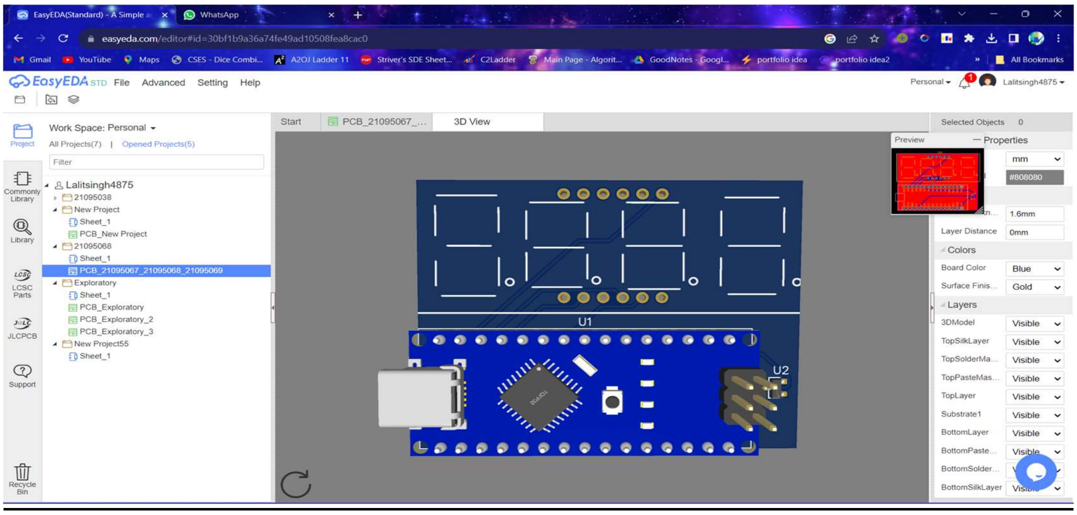
2D View:-



Selected Objects 0

Background	#000000
Grid	
Visible Grid	Yes
Grid Style	line
Grid Size	10
Display	
Board Side	Top Side
Silk Screen	Visible
Board Color	Blue
Surface Finis...	Gold
Mouse-X	-10.541mm
Mouse-Y	-12.319mm
Mouse-DX	-10.527mm
Mouse-DY	-12.314mm

3D View:-



Selected Objects 0

	mm
	#808080
	1.6mm
Layer Distance	0mm
Colors	
Board Color	Blue
Surface Finis...	Gold
Layers	
3DModel	Visible
TopSilkLayer	Visible
TopSolderMa...	Visible
TopPasteMas...	Visible
TopLayer	Visible
Substrate1	Visible
BottomLayer	Visible
BottomPaste...	Visible
BottomSolder...	
BottomSilkLayer	Visible