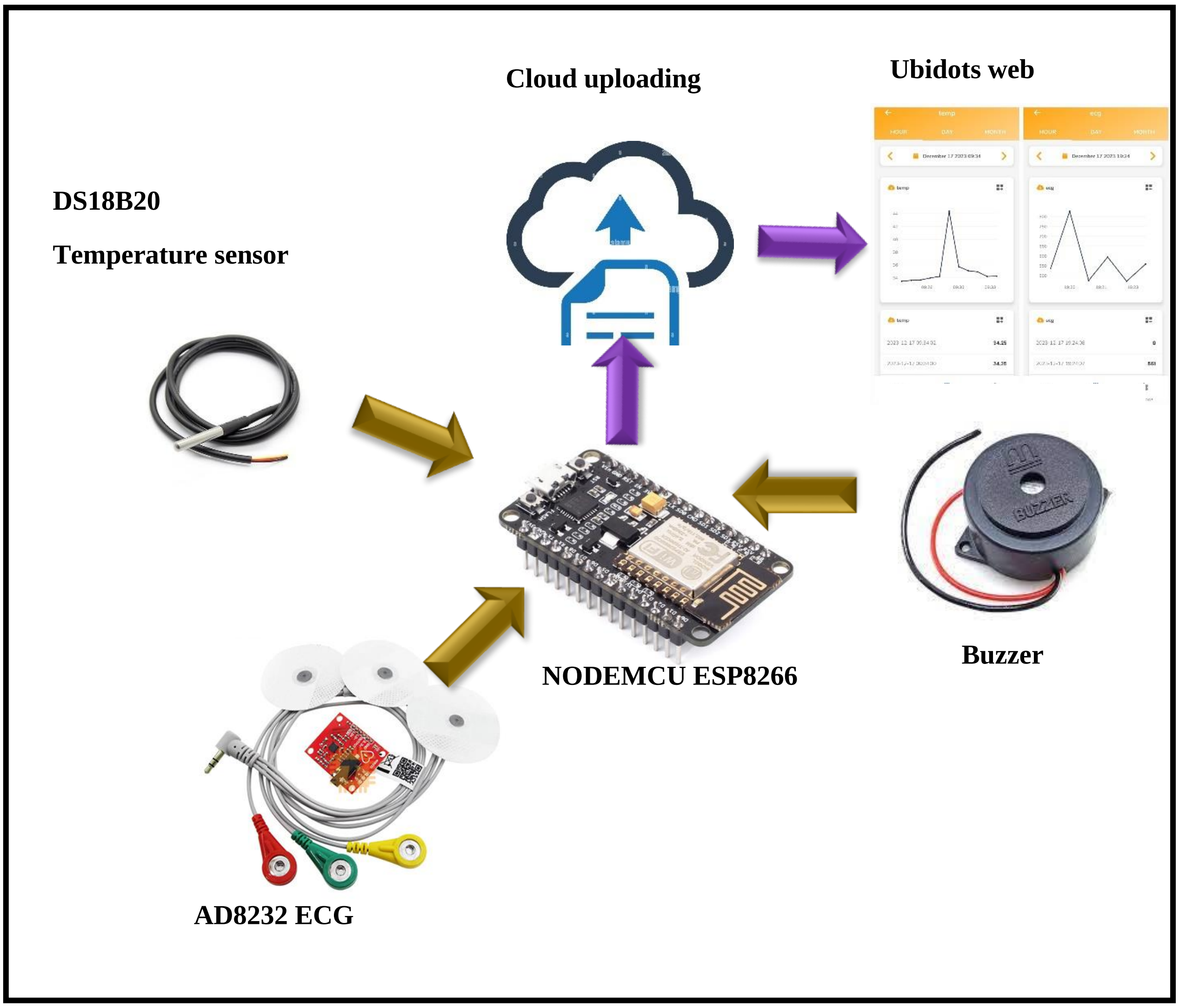
E-HEALTH MONITORING SYSTEM USING IoT

Keerthikan F.J

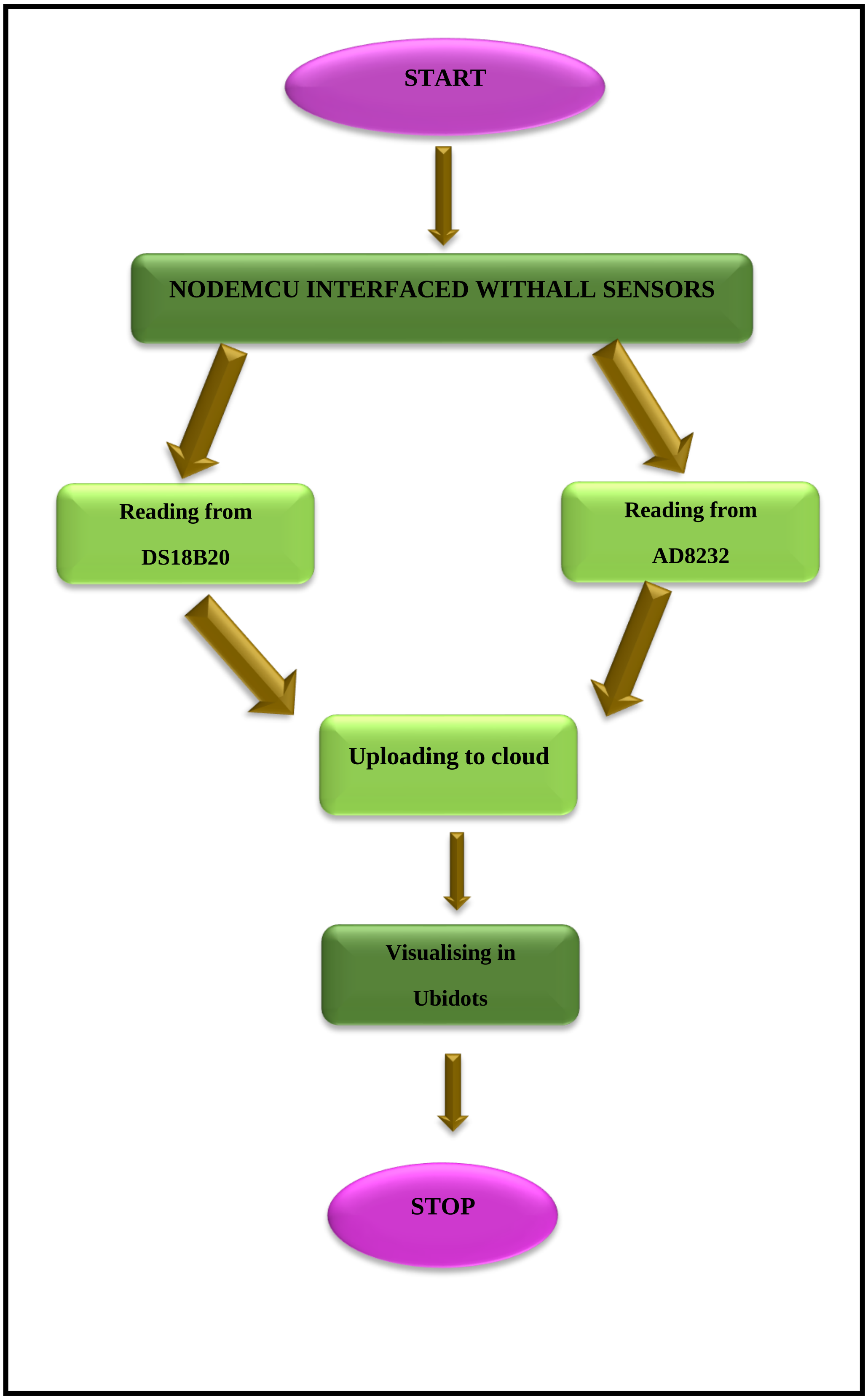
Department of computer Engineering

University of Jaffna

BLOCK DIAGRAM



1.5.2 FLOW CHART



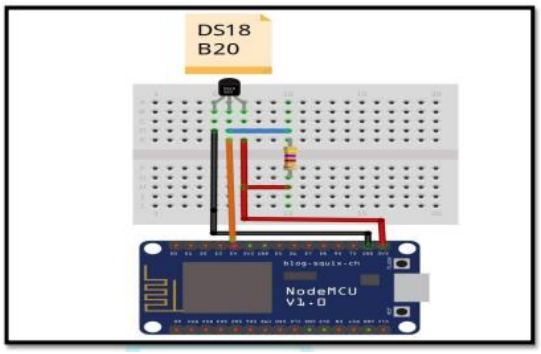
COMPONENTS

Hardware Used

- ✓ NODEMCU - 8266
- ✓ Temperature sensor - DS18B20
- ✓ ECG sensor - AD8232
- ✓ Bread board
- ✓ Jumper cables
- ✓ Buzzer
- ✓ Power source
- ✓ Three lead electrodes

Temperature sensor DS18B20

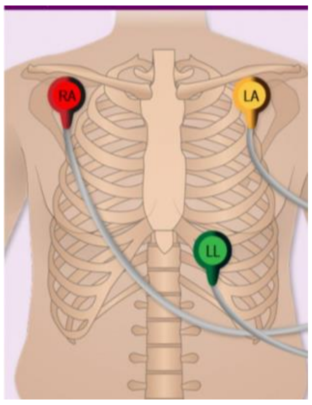
- ➢ This sensor basically works on the one wire communication protocol which places a resistor in between the VCC and the signal pin of DS18B20 to make it by default in the high state.
- ➢ This is a digital sensor with 12-bit of ADC embedded.
- ➢ The usable temperature range is from -55 to 125 degree Celsius as well as it has a temperature limit alarm system.

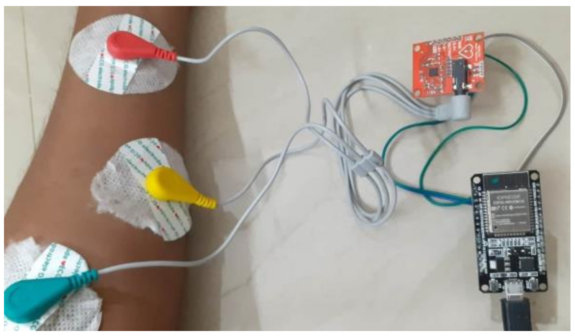


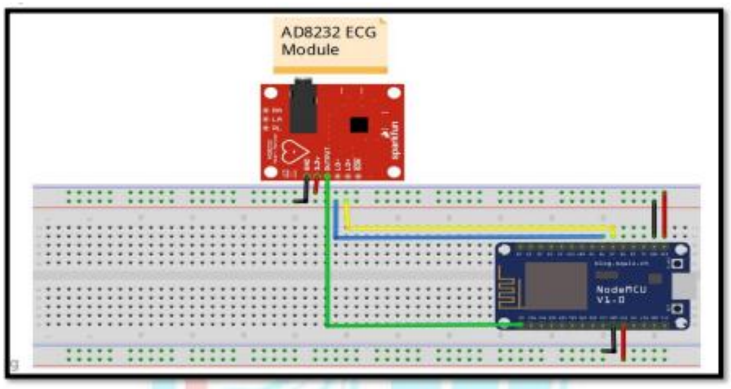
- NodeMCU 3.3V ------ -- 3.3V pin
- NodeMCU Digital Pin ------ D4
- NodeMCU Gnd ----------- Gnd

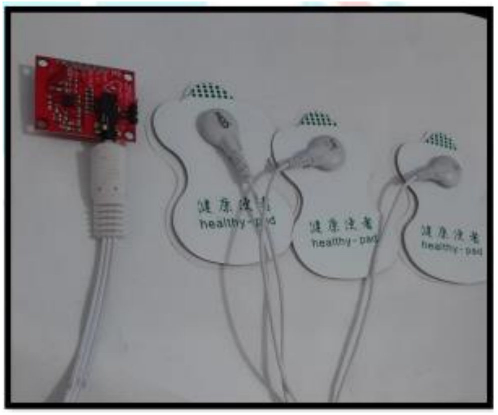
ECG SENSOR Module AD8232

- Whenever the body is connected with the electrodes of the ECG sensor, it conducts electrical signals from the skin surface with the help of gel present in the electrodes.
- With every oxygenated and deoxygenated blood pumps in and out, there happens to be an electrical energy spreading around the body but that is of very low power which in turn is not felt by us.
- In this case the conductive gel present inside the electrodes conducts that low power electrical signal and finally amplifies the signal in order to get the analog values of the ECG which is later on plotted on the ECG paper.





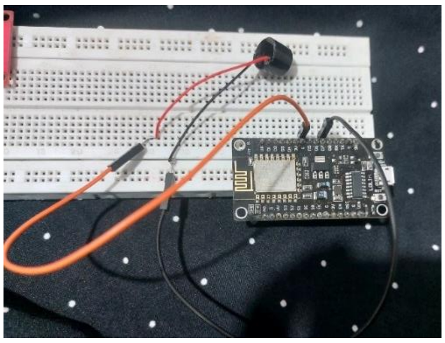




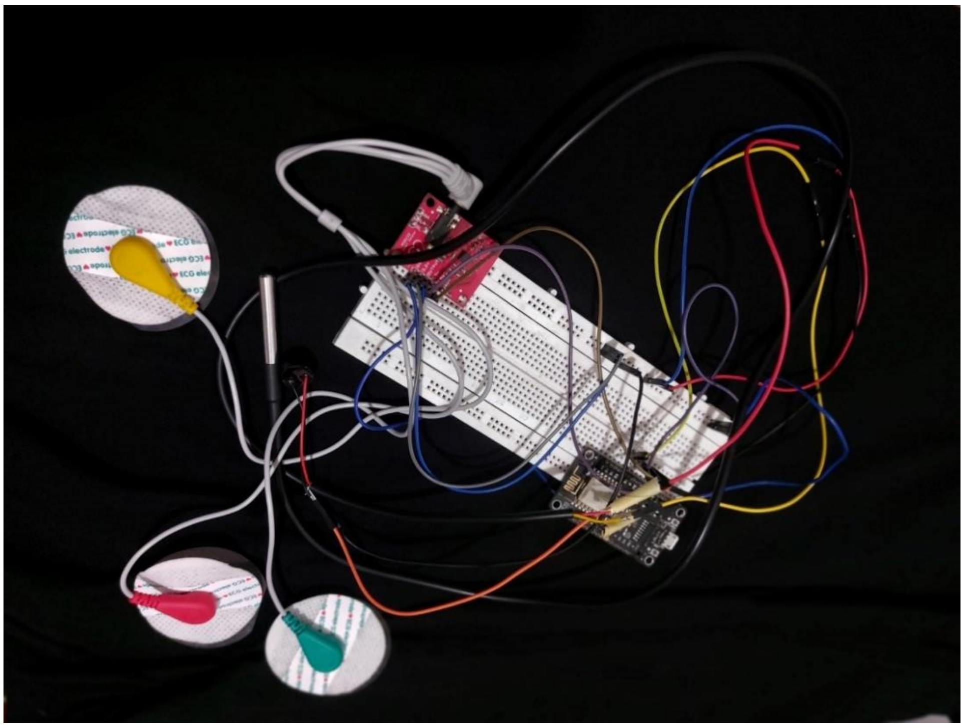
- NodeMCU 3.3V ------ -- 3.3V pin
- NodeMCU pin D6---------L0+
- NodeMCU Pin D5---------L0-
- NodeMCU Analog (A0)-----Output
- NodeMCU Gnd----------- Gnd

Buzzer

- It will make sound when temperature go above 37^{0}C.



Hardware Model



Software used

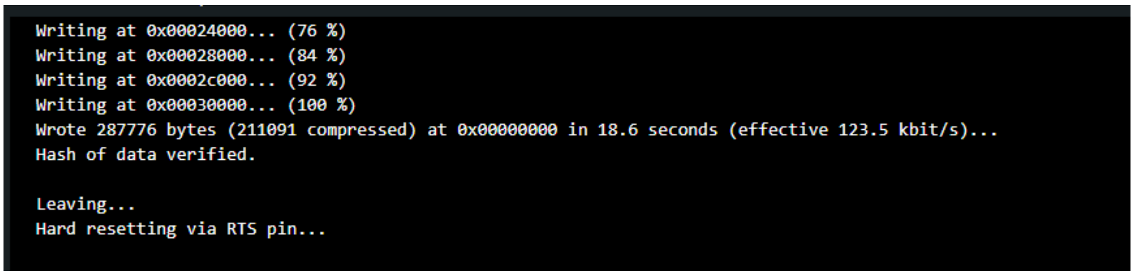
- ✓ Arduino IDE
- ✓ Ubidots

Code:

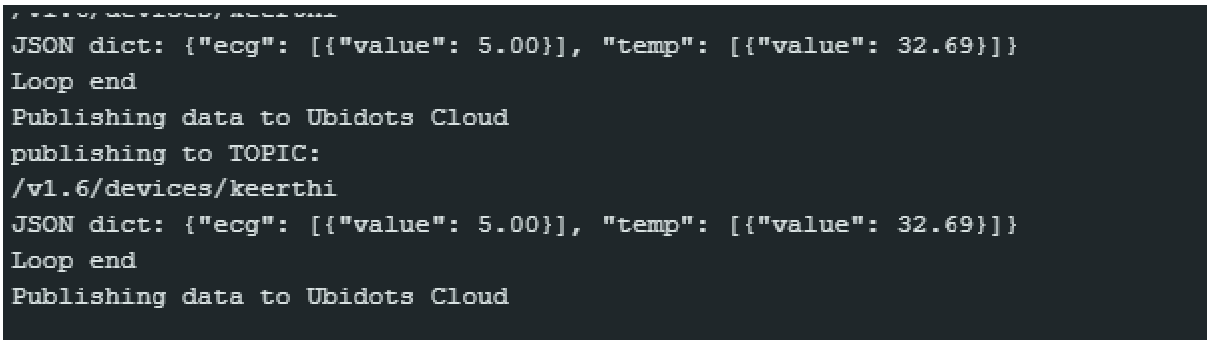
```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#include "UbidotsESPMQTT.h"

#define WIFISSID "Vithursi" // WiFi SSID
#define PASSWORD "12341234"        // WiFi password
#define TOKEN "BBUS-t3FPlKXprOMiaoTgH0RKeqao7hS7vp" // Ubidots TOKEN
#define MQTT_CLIENT_NAME "1234a5d6798" // MQTT client name
#define VARIABLE_LABEL "ecg"  // Variable label
#define DEVICE_LABEL "keerthi"// Device label
#define SENSOR A0 // Analog sensor connected to A0
#define LO_MINUS D5 // LO- connected to pin D5
#define LO_PLUS D6  // LO+ connected to pin D6
#define DATA_PIN D4  // Pin where the DS18B20 data line is connected
#define BUZZER_PIN D7 // Pin connected to the buzzer

char mqttBroker[]  = "industrial.api.ubidots.com";
char payload[100];
char topic[150];
char str_sensor[10];
float temp;
```

```
OneWire ourWire(DATA_PIN);
DallasTemperature sensors(&ourWire);
WiFiClient ubidots;
Ubidots ubidotsClient(TOKEN, MQTT_CLIENT_NAME);

void callback(char* topic, byte* payload, unsigned int length) {
  // Handle callback
}

void reconnect() {
  while (!ubidotsClient.connected()) {
    Serial.println("Attempting MQTT connection...");
    if (ubidotsClient.connect()) {
      Serial.println("Connected");
    } else {
      Serial.print("Failed to connect to Ubidots, try again in 2 seconds");
      delay(2000);
    }
  }
}

void setup() {
  Serial.begin(115200);
  WiFi.begin(WIFISSID, PASSWORD);
  pinMode(SENSOR, INPUT);
  pinMode(LO_MINUS, INPUT);
  pinMode(LO_PLUS, INPUT);
  pinMode(BUZZER_PIN, OUTPUT); // Set buzzer pin as output
  sensors.begin();
  Serial.println();
  Serial.print("Waiting for WiFi...");
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
  }

  Serial.println("");
  Serial.println("WiFi Connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());

  ubidotsClient.setDebug(true); // Enable debug prints to serial monitor

  ubidotsClient.wifiConnection(WIFISSID, PASSWORD);
  ubidotsClient.begin(callback);
}

void loop() {
  if (!ubidotsClient.connected()) {
    reconnect();
  }

  sprintf(topic, "%s%s", "/v1.6/devices/", DEVICE_LABEL);
  sprintf(payload, "%s", ""); // Clean the payload
  sprintf(payload, "{\"%s\":", VARIABLE_LABEL); // Add the variable label
}
  float sensor = analogRead(SENSOR);
  dtostrf(sensor, 4, 2, str_sensor);
  sprintf(payload, "%s {\"value\": %s}}", payload, str_sensor); // Add the value
  Serial.println("Publishing data to Ubidots Cloud");
  ubidotsClient.add(VARIABLE_LABEL, sensor); // Insert your variable label and the
value to be sent

  sensors.requestTemperatures();
  temp = sensors.getTempCByIndex(0);
  ubidotsClient.add("temp", temp); // Insert your temperature variable label and value

  if (temp > 37.0) {
    digitalWrite(BUZZER_PIN, HIGH); // Activate the buzzer
  } else {
    digitalWrite(BUZZER_PIN, LOW); // Deactivate the buzzerSs
  }

  ubidotsClient.ubidotsPublish(DEVICE_LABEL); // Publish the data to Ubidots under
device name 'keerthi'

  delay(1500);
  Serial.println("Loop end"); }
```

OUTPUT RESULTS



```
Writing at 0x00024000... (76 %)
Writing at 0x00028000... (84 %)
Writing at 0x0002c000... (92 %)
Writing at 0x00030000... (100 %)
Wrote 287776 bytes (211091 compressed) at 0x00000000 in 18.6 seconds (effective 123.5 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
```



```
JSON dict: {"ecg": [{"value": 5.00}], "temp": [{"value": 32.69}]}
Loop end
Publishing data to Ubidots Cloud
publishing to TOPIC:
/v1.6/devices/keerthi
JSON dict: {"ecg": [{"value": 5.00}], "temp": [{"value": 32.69}]}
Loop end
Publishing data to Ubidots Cloud
```

UPLOADING DATA TO CLOUD

- UbiDots application has very easy user interface and hence can prove to be useful for family members as well as doctors to monitor the data easily and give medication accordingly.
- It uses the TCP/IP protocol which securely communicates with the respective known device and transfers the data in real time.
- The interfacing of adding different widgets and creating its functionality is really very simple for non-tech people as well. Moreover, for long term application, one can also take the values in the csv files which can be used in the machine learning algorithm to predict and also analyse the data.

OBSERVATIONS AND RESULTS

- **Observation in Ubidot :-** The body parameters (body temperature, heart beat in BPM, oxygen rate in blood and ECG of the heart) of the patient are visible remotely on the Blynk app from any location.

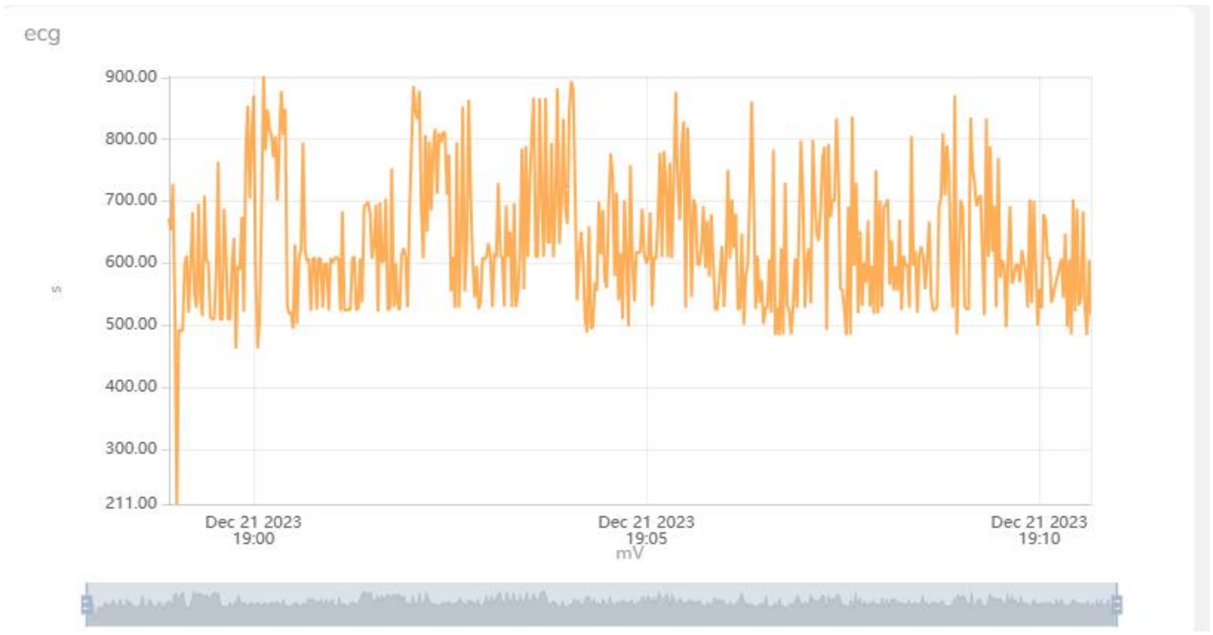


Fig 11 ECG reading in Ubidot (Lap)

ECG reading - : Body parameters of a healthy volunteer were obtained using our E-Health monitoring system. ECG plot parts were traced out and compared with normal range of ECG parameters

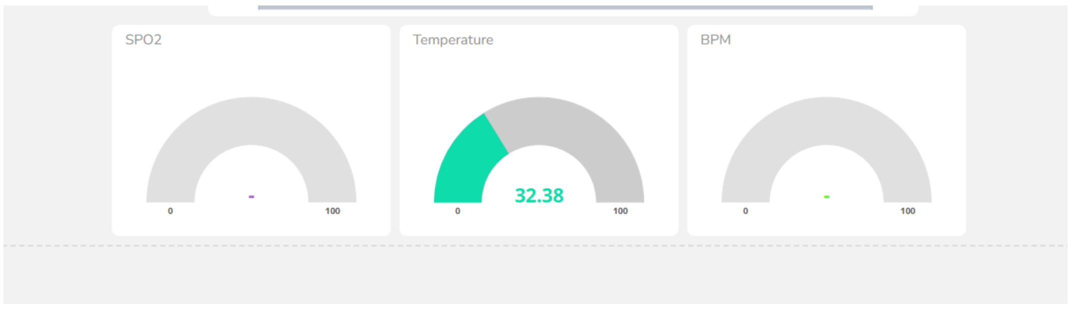
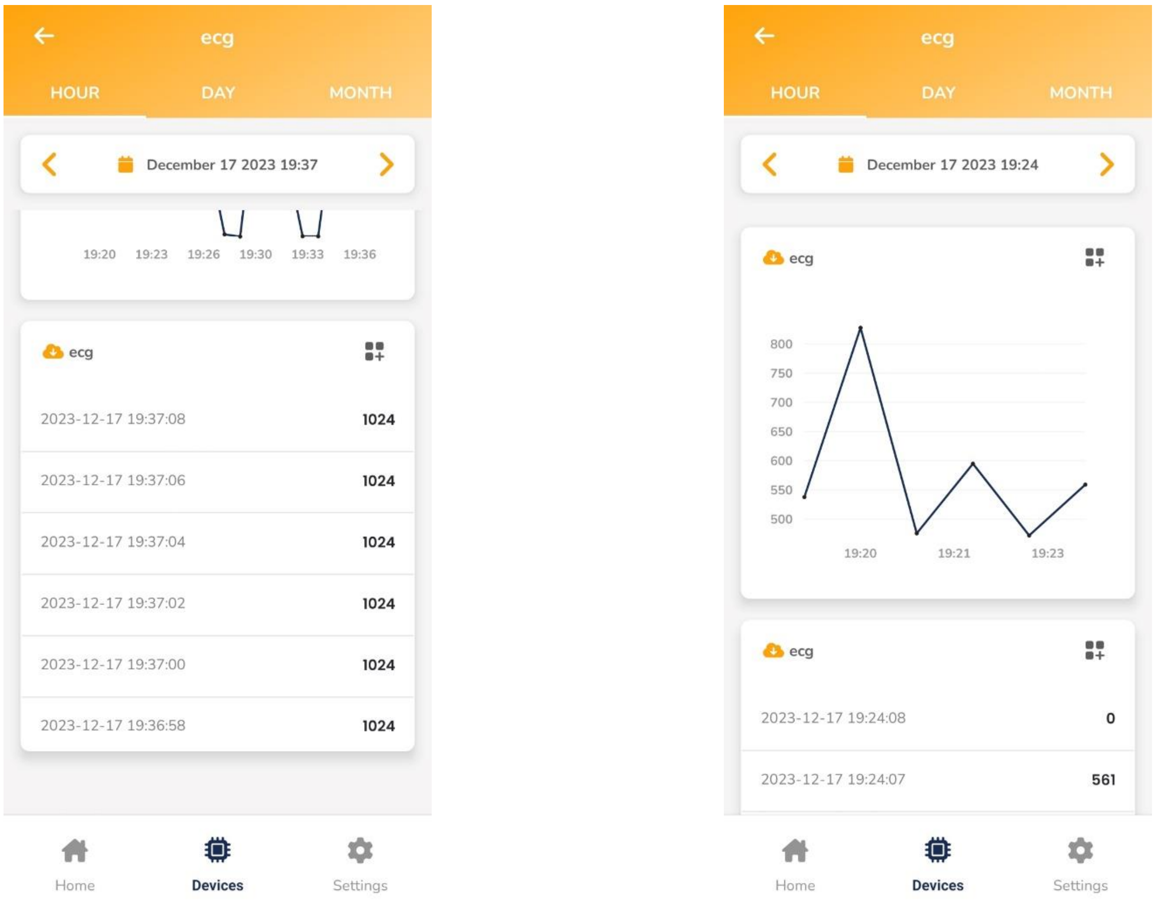


Fig 12 temperature reading in UbiDots (lap)

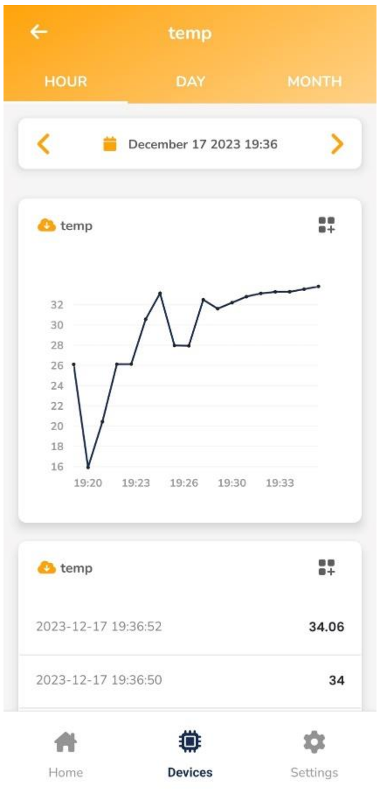
Temperature reading – Temperature sensor sense the body temperature. Temperature will increase up to 125^{0}C. But body temperature will not go to that value. It's not possible. When the temperature increase above 37^{0}C, alarm will make sound and alert it will alert us.



2023-12-17 19:37:08 1024
2023-12-17 19:37:06 1024
2023-12-17 19:37:04 1024
2023-12-17 19:37:02 1024
2023-12-17 19:37:00 1024
2023-12-17 19:36:58 1024

2023-12-17 19:24:08 0
2023-12-17 19:24:07 561

Fig 13 ECG reading in UbiDots (Mobile)



2023-12-17 19:36:52 34.06
2023-12-17 19:36:50 34

Fig 14 Temperature reading in Ubidots (Mobile)

EVALUATION

Name	Quantity	Unit Cost (Lkr)	Total cost(LKR)
NodeMCU	**1**	**950**	**950**
MAX30100	**2**	**750**	**1500**
DS18B20	**1**	**350**	**350**
AD8232	**1**	**1900**	**1900**
Breadboard	**1**	**500**	**500**
Buzzer	**1**	**150**	**150**
Other things			**600**
TOTAL			**5650**