Cairo University

Faculty of Computers and Artificial Intelligence

Department of Information Technology

An IoT Based Health Monitoring System with Medicine Box

Supervised by

Prof. Imane Aly Saroit Ismail

Implemented by

Name	ID
Mahmoud Ramadan Sayed	20180252
Mohamed Ahmed Abdelrahman	20180214
Ahmed Mohamed Ibrahim	20180021

Graduation Project

Final-Year Documentation

Academic Year 2022-2023

Abstract

Health is characterized as a full state of physical, mental, and social well-being and not merely a lack of illness. Health is a fundamental element of people's need for a better life. Unfortunately, the global health problem has created a dilemma because of certain factors, such as poor health services, the presence of large gaps between rural and urban areas, Physicians, and nurses' unavailability during the hardest time. According to the increased usage of mobile devices and smart devices like (Fitbit, and Mi-Fit) and the technology of IoT we can use it to design and make the system, Due to the shortage of nurses and the difficult health condition of the community, we decided to design this project. The project is defined as follows we monitor the patient's condition parameters by using sensors to prevent the patient from early death, so we will design a robot unit that will stay with the patient all the time and monitor its condition and update the case doctor all the time and if there any change in the parameters then the robot unit will go to the patient and gives him the medicine this will be held in the medicine box, and that will be under supervised of the doctor, so the doctor can update the medicine and check it in the time

Chapter 1: Introduction

1.1 Problem Definition

The problem that faced the community of health care today and in the future is how to provide the factor of the human being that is represented in our project at nursing staff and the doctors and how we can protect them from diseases by avoiding them from the direct communication with the patients, And how we can provide the needs of the patient from the medicines and the health care around the day and how we can minimize the risk at the patient life by follow up the patient's case around the day, and supply the doctor with all information needed to make the right decision for the patient's life or giving him the suitable medicine for its case all of these should be managed remotely without any physical communication with the patient to make sure that the nurse or the doctor in the safe zone, so we thought in this idea to overcome or reduce the percentage of the infection, An IoT health monitoring system with medicine box provides all these parts for the patient's life and the nurse or doctor safe life

1.2 Objectives

Prevent the spread of the virus and diseases:

Helping the nursing staff and institutions of health during the hardest time

Follow up with the patient around the day:

Helping the patients and their families by monitoring the patient's cases around the day

No risk:

Reducing healthcare costs and minimizing the risk

1.3 System usability

Hospitals:

The system can be used as a temporary nurse in the hospital.

In global epidemic:

Used in the hardest times like covid-19 (Coronavirus) to avoid the infection.

In Homes:

It can be used also in homes for patients that lie down in homes.

1.4 Using smart technology

The increased usage of mobile devices and smart devices in the health field or health care filed like (Fitbit, Mi-Fit, etc.) allowed to us have new ideas and projects. helping the institutions of health or helping doctors to connect to the patients. directly without any physical connection, the second part of this smart technology is. IoT Tee technology of IoT allowed us to connect to our parts and our chips of us. The project sent and received the data rapidly and allowed us to connect to the cloud. servers to store the data, process, and analysis.

1.5IoT used technology.

In the project, we use the technology of IoT to connect all parts of the project to the internet to achieve the connection remotely, collect the data from the sensors, and then send it to the cloud server that we used, Why IoT technology? Because it is very common and is very fast and gives us a high speed for interacting and connecting the parts of our project together and synchronously

1.6Follow up on the patient's condition.

The smart system is defined and will track the patient as follows: the sensor will sense the patient's vital parameters like temperature and oxygen and heart rate, and after that, the parameters will be displayed on the local screen (OLED screen) for patient's monitoring, the parameters will be sent to the cloud server to store and process it and display the analyzed data on the Blynk application if there is any change in the parameters, the doctor will be updated, and the robot unit will take action according to the update and will go to the patient to give him the medicine using the medicine box that medicines box will be on the robot unit and open based on the order that will come from the server and it will be matched for the order.

1.7The abstracted approach

We concluded that the system steps and the best model to build the system, robot unit consists of two main parts medico box and robot unit, will move on the colored track, there are two stations, first station is the patient destination station, the second station is the source station that the car will go.

1.Parameters sensing, the sensors sense the data from the patient

2.Display the data on the room screen

3.Sending the data to the cloud server

4.Analyzing the data on the cloud server and display it.

5.Sending the data to the doctor by the application

6.The doctor will take an action or not

7.Robot unit starting move

8.Reach the destination and giving the medicine

9. go back to the robot unit source

`

So, the system will monitor the data all the time until the change occurred, then the change will send to the doctor and the doctor will give the action to the robot unit to starting the move.

Chapter 2: System Definition

2.1 Project definition

The project is divided into three main parts, the first part is the medico box, this medico box consists of many components or devices, containing the sensors and the medicine boxes for the patient these boxes are opened and closed under the order of the doctor, and the OLED screen that we are displaying the parameters of the patient on it, The second main part is the robot unit, the robot unit consists of four dc motor with their wheels which are responsible for moving forward and moving at any direction that we want, And there is the Arduino UNO the main device on the robot unit so the car cannot take any action without this device and there are two motor drivers, and the color sensor, the third main part is the colored route, this colored route consists of four colors red, green, yellow, and black and by the definition, all these parts work together and synchronize to achieve the main concept (robot unit with medico boxes)

2.2 The robot unit structure

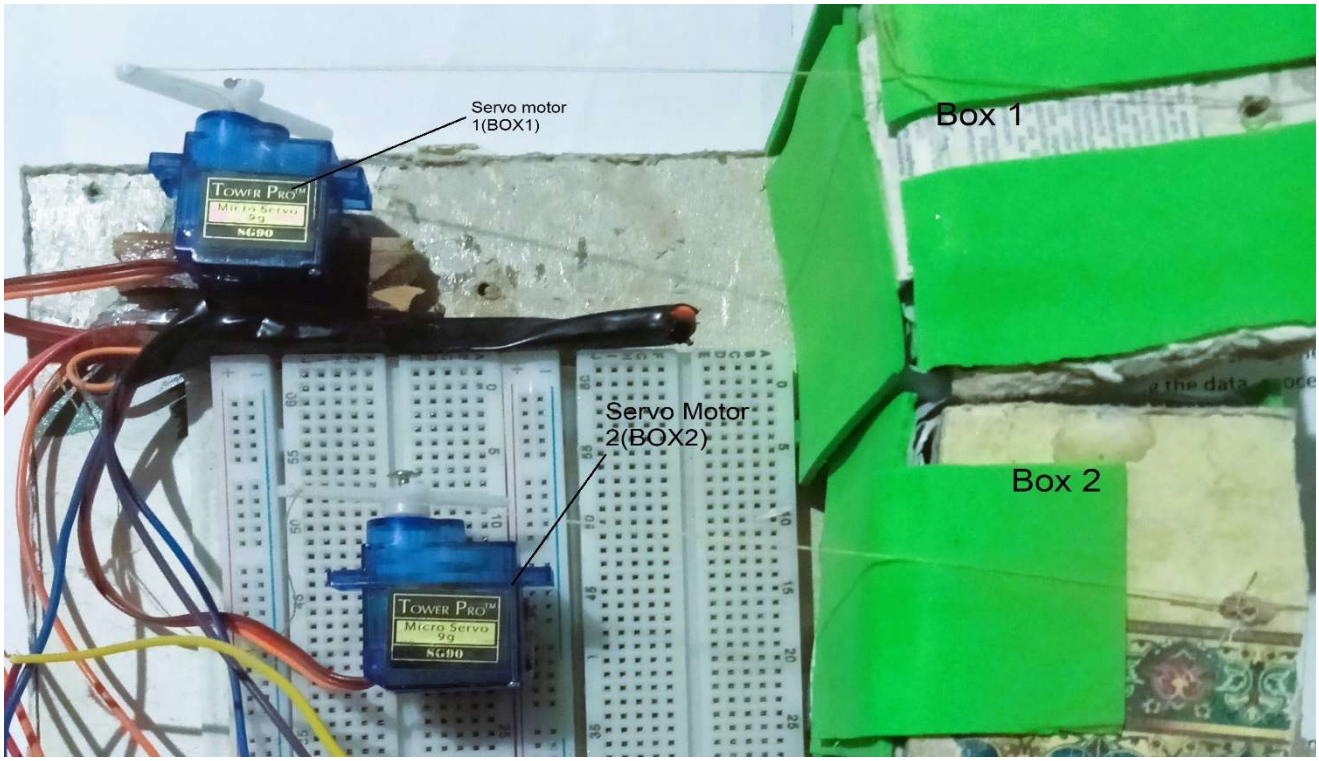
1.Health Monitoring kit with IoT

2.Medico Box

3.Arduino car

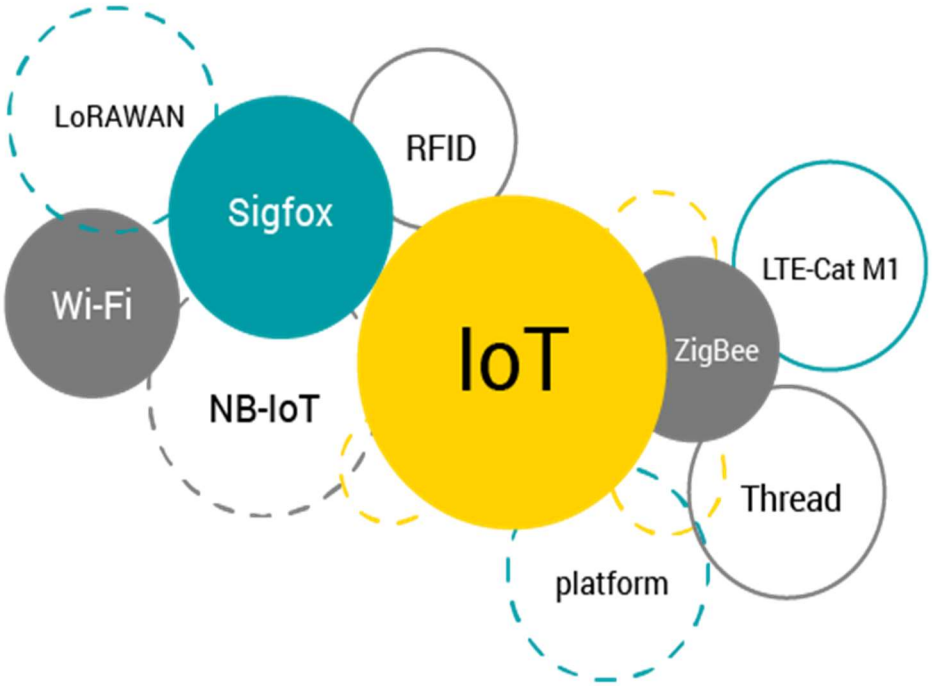
2.3 Health monitoring kit with IoT:

Patient Health Monitoring using IoT is a technology that enables the monitoring of patients without any physical contact with the patients. which may increase access to care and decrease healthcare delivery costs. This can significantly improve an individual's quality of life. It allows the patients to maintain the proper health the condition reduces the risk and minimizes personal costs. In addition, to this that patients and their family members feel comfort knowing that they are being monitored and will be supported if any problem arises. Medical personnel could be present in the place for guiding the People.



2.4 Medico Box

Medico Box is a technology that uses IoT in order to enable the patients to get a Real Time Updating about the medicine to be taken from Doctor or Physician. A Promising trend in healthcare is to move routine medical checks and other healthcare services from hospital. The use of IoT in Medicine Box has been made with the help of Mobile Application. If the Correct Medicines based on the Update received from Doctor or Physicians are taken at the right time, there are less chances that the condition of the patient getting worse.



2.5 The intelligent medicine box

In this project, the intelligent medicine box will help a patient to take his/her medication when it is time to take, and it depends on the health condition of the Patient. For example, if a patient has high temperature, then it will open the Box1 which contains Medicine related to High Human Body in addition to this it will get the approval from the doctor before proceeding.

The body temperature, SpO2, pulse values and medicines taken data will be stored in a server which can be accessed by both patient and doctor so that when it is time the doctor can review the medicine and can change if needed. Also, it will be helpful for doctors to keep updated about the patient's physical health condition. IoT is rapidly revolutionizing the healthcare industry. In this project, we have designed the IoT-Based Patient Health Monitoring System using ESP8266 & Arduino. The IoT platform used in this project is Blynk. Blynk is a new platform that allows you to quickly build interfaces for controlling and monitoring your hardware projects from your iOS and Android device. IoT device could read the SpO2, pulse rate and measure the surrounding temperature. It continuously monitors the SpO2, pulse rate and surrounding temperature and updates them to an IoT platform. The Arduino Sketch running over the device implements the various functionalities of the project like reading sensor data, converting them into strings, passing them to the IoT platform, and displaying measured SpO2, Pulse rate and temperature on OLED Display and Blynk platform

2.6 IoT technology.

The technology of IoT that allow to us communicate and interact with the patient so fast by the component that we installed it on the project, by connect it the component with the wi-fi and receive the parameters from the sensors and sending the data to the cloud server that the IoT cloud, for storing, processing, analyzing, so we will use the IoT platform for that (Blynk IoT platform)

2.7 Project stages

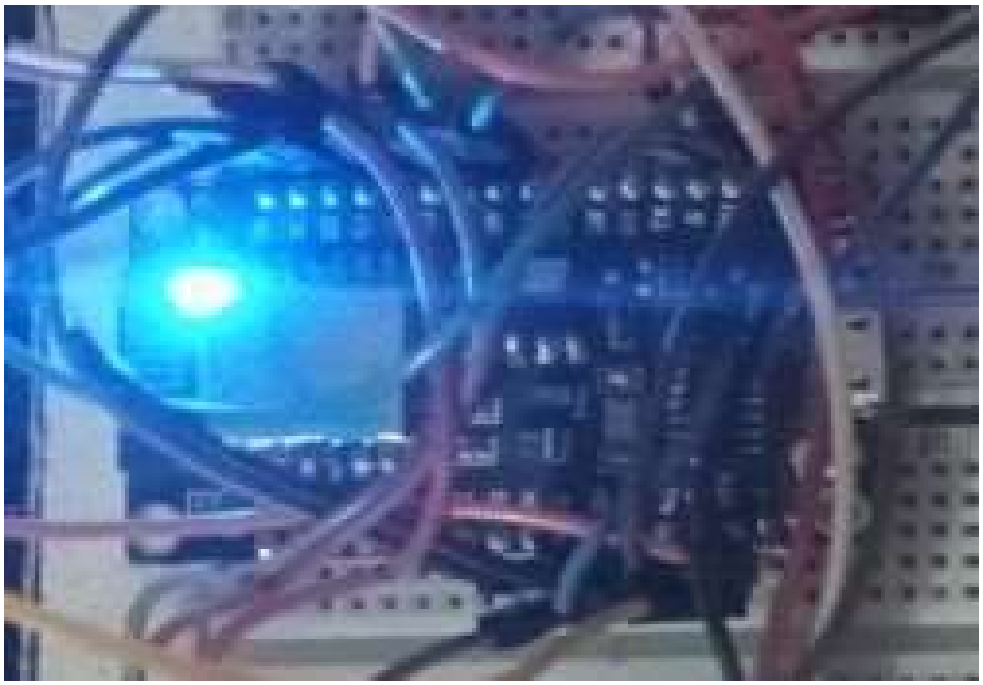
2.7.1 Stage I

The project divided into stages the first stage, the local monitoring that we will sense the parameters using the DHT11 sensor that for temperature and humidity and the MAX30100 oximeter sensor for sense the heart rate for the patient and the rate of oxygen, to display them on the display screen that will be on the robot unit in the patient's room for the patient's family or for the people that will be available with the patient and for the patient himself till this point this is the first stage we will design it using only two sensors and the display screen.


Westinghouse
RECHARGEABLE
200 mAh
Ready to Use
BLOCK 8.4H5
Nickel Metal Hydride
8.4V SIZE 9V
Ni-MH
BPM : 238.84
SpO2 : 94.00
Temp : 30.90

2.7.2 Stage II

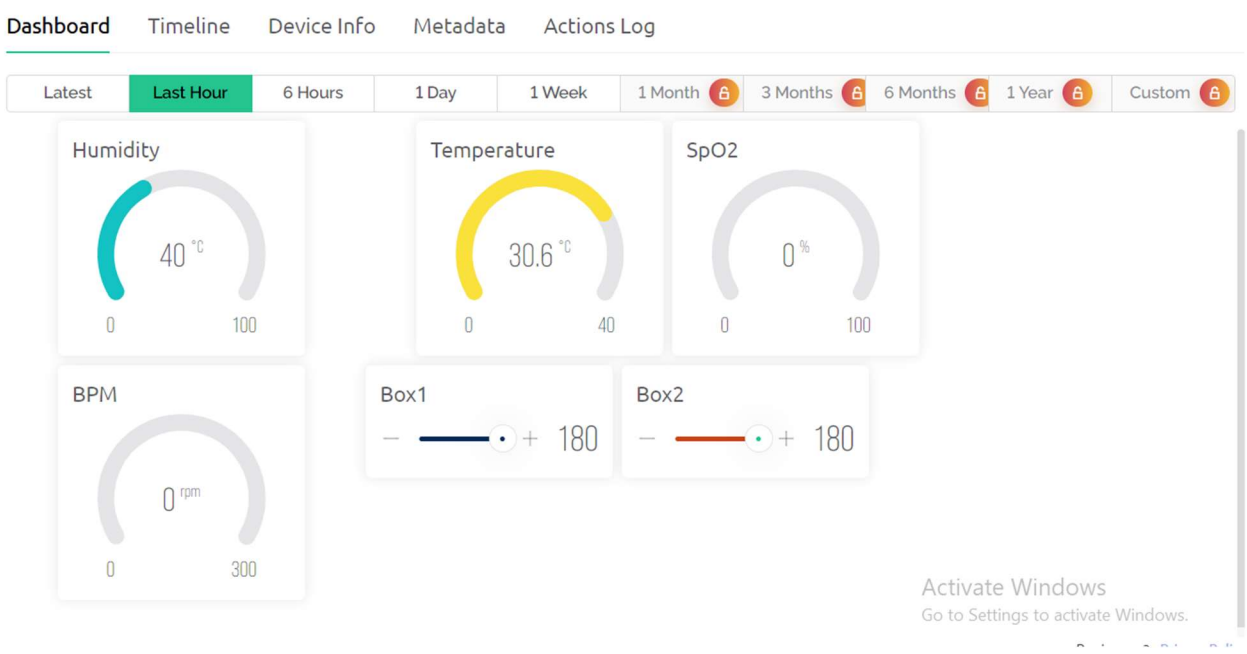
The second stage will be like that, at first we need to Wi-Fi module to connect to the cloud server and making the part of IoT so we will install the node MCU that contain the wi-fi module that will connect to the IoT cloud server, so the cloud server will receive and will be connected to robot unit throughout the node MCU, the sensors will sense the data and will display it on the OLED screen and send it to the cloud server using that node MCU, connect to the wi-fi and connect to the cloud server by using the account IoT cloud site.



2.7.3 Stage III

Receiving the data from the robot unit, store it and after that the cloud server will perform the processing on the data to check if there is any change in the parameters of the patient by checking the old parameters and the updated parameters, then it will display it on the cloud application that the doctor will monitor the patient's condition throughout this application.

The Cloud Site monitoring:



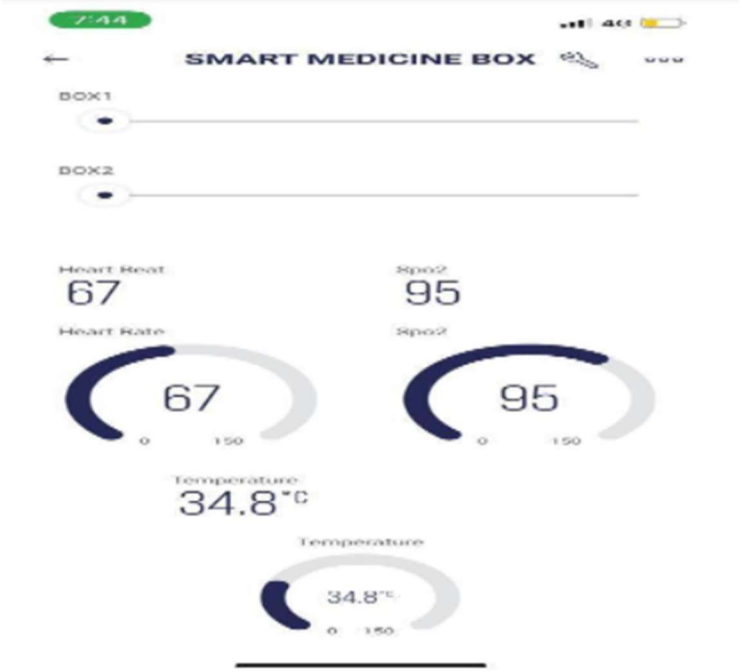
2.7.4 Stage IV

The Data will be sent to the doctor via the cloud server and by the application that we used to store and monitoring the data on the it, the doctor will be informed all the time with the changing or unchanging of the parameters of the patient so there are four pins for the temperature, humidity, heart rate, and oxygen rate, this pins show the digital readings for the previous four parameters and the notification will send all the time for the doctor using the cloud application.

Medico Boxes sliders:

The two medico boxes have two sliders on the cloud site and the application of the site so the doctor can open the boxes using these sliders, and the project contains two medico boxes the first box for the temperature parameter and the second box for the heart disease, so the doctor will choose the suitable medicine for the patient condition and will give the robot unit the order depending on this situation.

Cloud server application:



2.7.5 Robot unit methodology

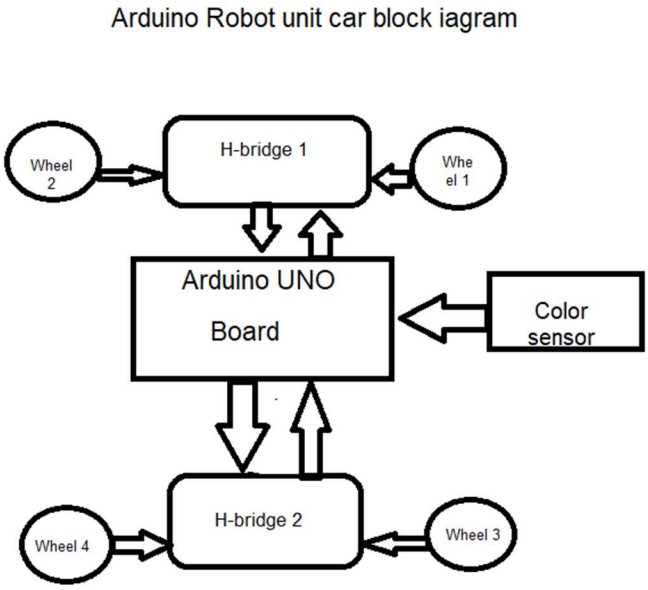
2.7.5.1 Robot unit block diagram

How the Arduino car will be connected to the medico boxes and how it can move in:

The block diagram of the Arduino car itself:

The Arduino car has dc motors and its wheels are connected, These dc motors are responsible for the moving, and the we should choose the suitable speed for this wheels but how, so the H-bridge dual module is connected to the Arduino UNO and the four dc motors to adapt the speed and the direction the car, project requires two H-bridge dual module to connect the four dc motors, so for each two dc motors we need one H-bridge dual module, the H-bridge dual module has 6 pins for input and four pins for output, the first two input for the direction for one wheel so we can change the direction, in clockwise and anti-clockwise, if you need to move in the direction of clockwise you connect the first pin to the HIGH pin and the second pin to the LOW pin in the Arduino UNO, so we can connect all the remaining three dc motors on this previous way, so we use the second two pins in the H-bridge module to connect the wheel 2 and the second H-bridge for the wheel 3 and 4, the question is how we can adjust the speed for the wheels the last two pins in the six pins of the H-bridge we use them to adjust the speed this pins are known PWM pins so for each wheel there is one PWM pin to adjust the speed, and this pins are available on the version of the Arduino UNO and are known by PWM pins and we use the pins 3, 9, 10, 11 to connect the H-bridge PWM pins.

Robot Unit block diagram:



And the important component to sense by it the color of the colored track is the color sensor that we use to detect the color of the colors.

Color sensor with robot unit:

color sensor that detects the basic three colors (RED, GREEN, BLUE), It reads ranges for the colors, so we decide the readable color according to the reading ranges for this color by the color sensor.

The color sensor is connected to the Arduino board and reads the color of the route and then the Arduino takes an action according to these readings.

Color sensor sensing mechanism:

1.Light source: The TCS3200 sensor requires a light source to illuminate the object or surface being analyzed. This can be achieved using built-in white LEDs or an external light source.

2.Color filtering: The TCS3200 sensor contains an array of photodiodes and color filters. The filters are typically red, green, blue, and clear (no filter). Each photodiode is covered by one filter, allowing it to respond primarily to a specific color of light.

3. detection: When the object or surface is illuminated, the filtered light is detected by the photodiodes beneath the respective color filters. The photodiodes generate electrical current proportional to the intensity of the detected light.

4.Light-to-frequency conversion: The TCS3200 sensor includes a light-to-frequency converter that converts the detected light intensity into a square wave frequency. The frequency of the output signal is directly proportional to the intensity of the detected light.

5.Output signals: The TCS3200 provides separate output pins for red, green, blue, and clear (no filter) frequencies. By reading the frequency output of each channel, you can determine the intensity of the respective color detected by the sensor.

6.Control pins: The TCS3200 also has control pins (S0 and S1) that allow you to set the frequency scaling of the output signal. By configuring these pins to specific logic levels (HIGH or LOW), you can adjust the sensitivity and frequency range of the color sensor.

7.Arduino interface: To interface the TCS3200 sensor with an Arduino, you connect the control pins (S0 and S1) to digital pins on the Arduino for frequency scaling. The output pins for each color (OUTR, OUTG, OUTB, and OUTC) are connected to digital input pins on the Arduino to read the corresponding frequencies.

8.By reading the frequency output of each channel and applying appropriate algorithms or calibration, you can determine the color and intensity of the object or surface being analyzed.

To connect a color sensor to an Arduino, you typically need to follow these steps:

1.Identify the type of color sensor you have: There are various color sensors available, such as the TCS3200, TCS34725, or the Adafruit Color Sensors. Make sure you have the datasheet or documentation for your specific sensor to understand its pinout and functionality.

2.Gather the necessary components: Besides your Arduino board and color sensor, you may require jumper wires, a breadboard (optional), and any additional components mentioned in the sensor's datasheet.

3.Understand the color sensor's pinout: Refer to the datasheet or documentation of your color sensor to identify the different pins and their purposes. Typical color sensors have pins for power supply (VCC and GND), data output (S0 and S1), data input (S2 and S3), and an interrupt pin (optional).

4.Connect power supply: Connect the VCC pin of the color sensor to the 5V pin on the Arduino board. Similarly, connect the GND pin of the color sensor to the GND pin on the Arduino board to complete the power supply circuit.

5.Establish communication: If your color sensor has frequency scaling pins (S0 and S1), connect them to digital pins on the Arduino. These pins control the frequency of the color sensing. For example, setting S0 to LOW and S1 to HIGH might select a certain range, while different combinations can be used to select other ranges.

6.Connect the data output: Connect the data output pin of the color sensor to a digital input pin on the Arduino. This pin will receive the color data from the sensor.

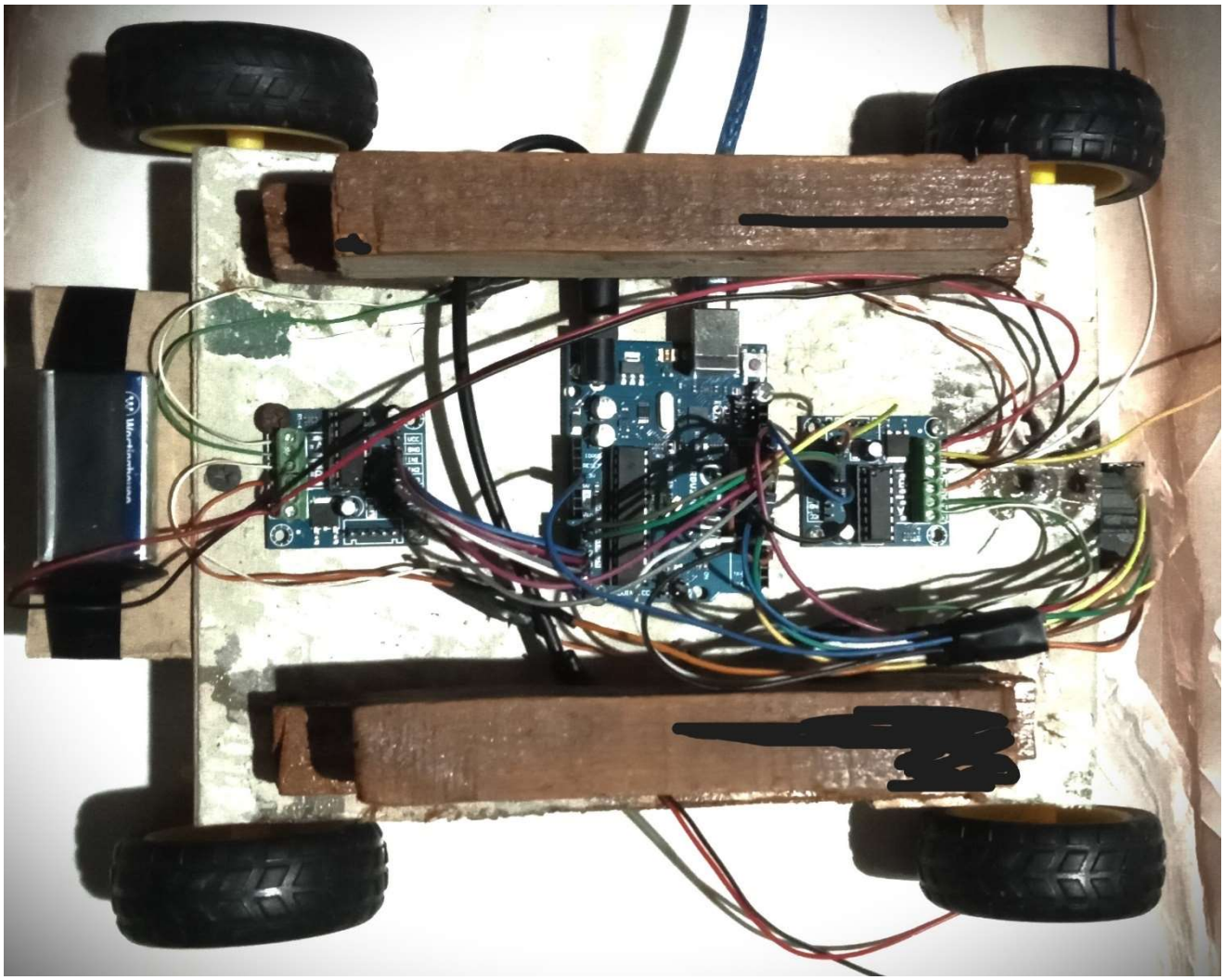
7.Optional: Connect the interrupt pin: If your color sensor has an interrupt pin, you can connect it to an available digital input pin on the Arduino. This pin can be used to trigger an interruption on the Arduino whenever a color event or specific condition occurs.

8.Check for additional components or connections: Some color sensors might require additional components such as capacitors or resistors as per the datasheet. Make sure to follow the recommended connections and values if applicable.

9.Upload and run the code: Once the connections are made, you need to upload the Arduino code that corresponds to your color sensor. The code will typically include libraries and functions to read the color sensor data and perform actions based on the detected colors.

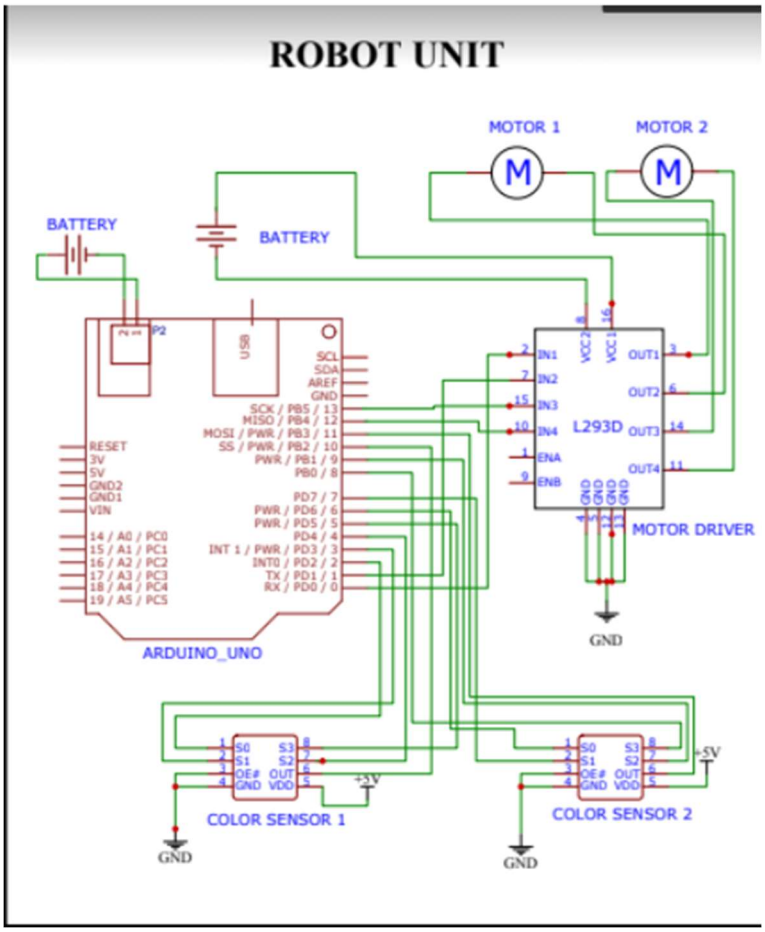
2.7.5.2 Arduino Car full body

The Arduino car part is ready to use, all parts are connected to give us the part of the Arduino car and that will be shown in the next figure.



2.7.5.3 Arduino car diagram

This diagram describes the relationship between the sensors and the Arduino UNO of the robot unit, the color sensor connection with the Arduino, The temperature sensor connection with the Arduino, The pulse rate sensor connection with the Arduino, And the H-bridge connection with its dc motors and the Arduino.



2.7.6 The color sensor circuit with Arduino

Steps of connection:

1.Connect the VCC pin of the TCS3200 sensor to the +5V pin on the Arduino Uno.
2.Connect the GND pin of the TCS3200 sensor to any GND pin on the Arduino Uno.
3.Connect the OUT pin of the TCS3200 sensor to digital pin 2 on the Arduino Uno.
4.Connect the S0 pin of the TCS3200 sensor to digital pin 3 on the Arduino Uno.

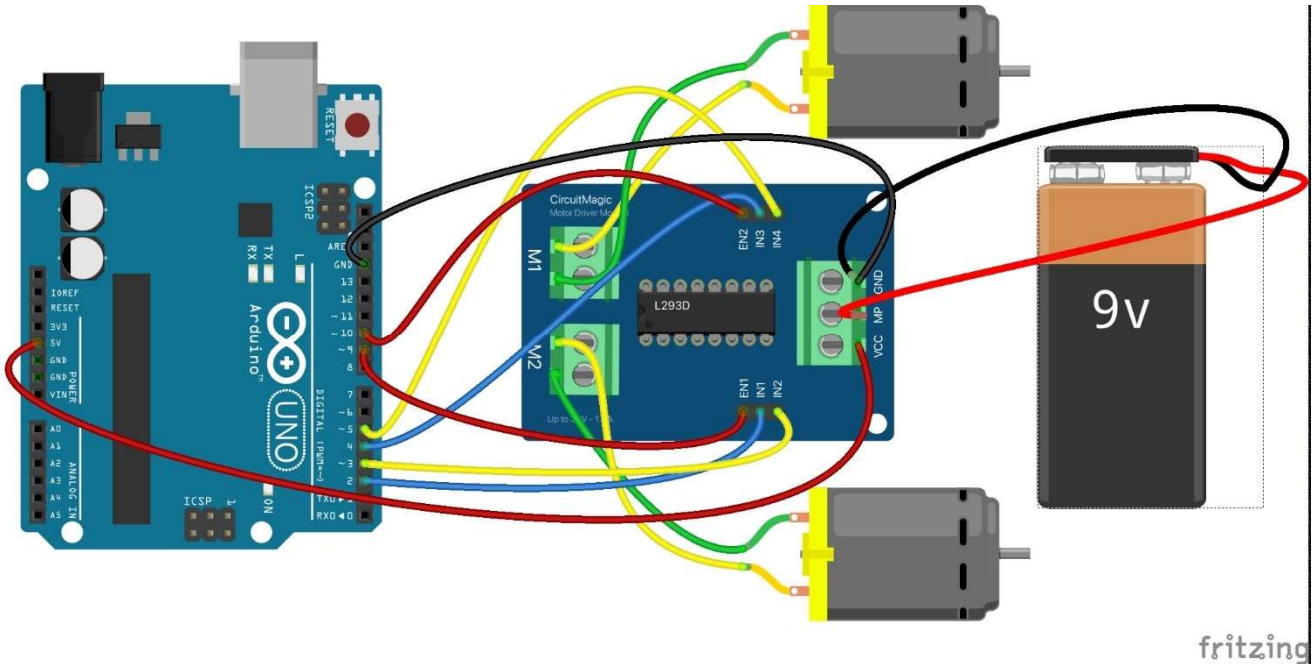
5.Connect the S1 pin of the TCS3200 sensor to digital pin 4 on the Arduino Uno.
6.Connect the S2 pin of the TCS3200 sensor to digital pin 5 on the Arduino Uno.
7.Connect the S3 pin of the TCS3200 sensor to digital pin 6 on the Arduino Uno.
8.Connect the OE (Output Enable) pin of the TCS3200 sensor to digital pin 7 on the Arduino Uno.
9.Connect the GND pins of the TCS3200 sensor to any GND pin on the Arduino Uno

2.7.7 The circuit between the H-bridge and the dc motor and the Arduino:

Steps of connection:

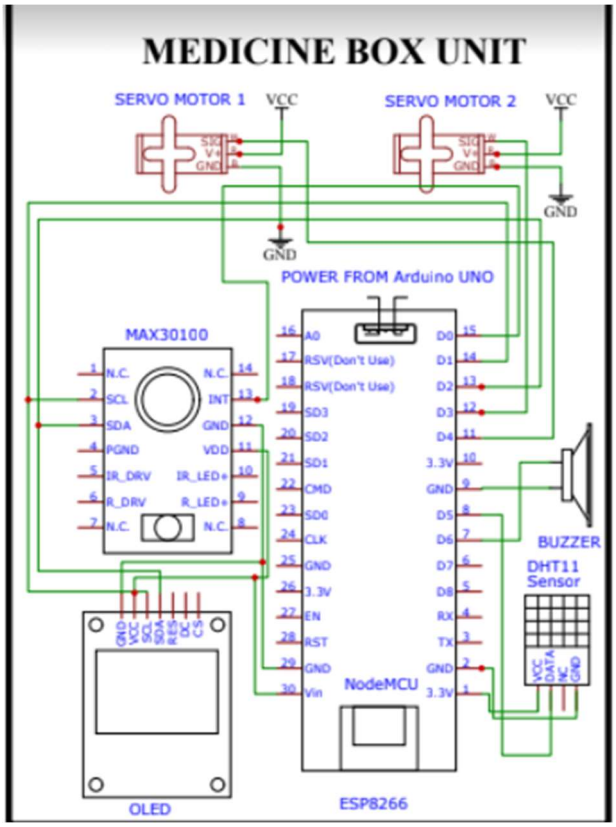
1.Connect the motor power supply's positive terminal to the +12V or the appropriate voltage pin on the H-bridge.
2.Connect the motor power supply's negative terminal to the GND pin on the H-bridge.

3.Connect the positive terminal of the DC motor to the M1A or OUT1 pin on the H-bridge.
4.Connect the negative terminal of the DC motor to the M1B or OUT2 pin on the H-bridge.
5.Connect the Enable1 (EN1) or Enable A pin on the H-bridge to any digital pin (e.g., pin 10) on the Arduino.
6.Connect the Input1 (IN1) pin on the H-bridge to a digital pin (e.g., pin 9) on the Arduino.
7.Connect the Input2 (IN2) pin on the H-bridge to another digital pin (e.g., pin 8) on the Arduino.
8.Connect the GND pin on the H-bridge to any GND pin on the Arduino.
9.Connect the VCC pin on the H-bridge to the +5V pin on the Arduino.
Once the connections are made, you can upload the appropriate Arduino code to control the DC motor using the H-bridge. The code will typically involve setting the direction and speed of the motor by toggling the digital pins connected to the H-bridge inputs and adjusting the enable pin.



2.8 Node-MCU diagram

This diagram describes the connection between the node-MCU (esp8266) and the parts of the medico box, The temperature sensor, the pulse rate sensor, the servo motors, the OLED display screen. The buzzer.

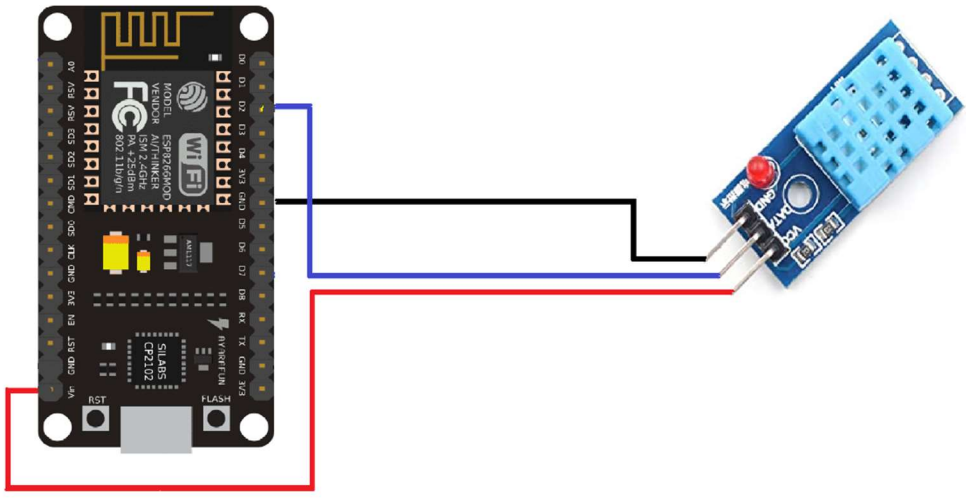


2.8.1 The circuit between the DHT11 temperature sensor and esp8266:

Steps of connection:

1.Connect the VCC pin of the DHT11 sensor to the +5V pin on the ESP8266 development board.
2.Connect the GND pin of the DHT11 sensor to any GND pin on the ESP8266 development board.
3.Connect the data pin (out) of the DHT11 sensor to digital pin D4 on the ESP8266 development board.

Once the connections are made, you can write the appropriate code to read the temperature and humidity data from the DHT11 sensor using the ESP8266. You'll need to install the necessary libraries for the DHT11 sensor and utilize the corresponding functions in your code to retrieve the sensor readings.

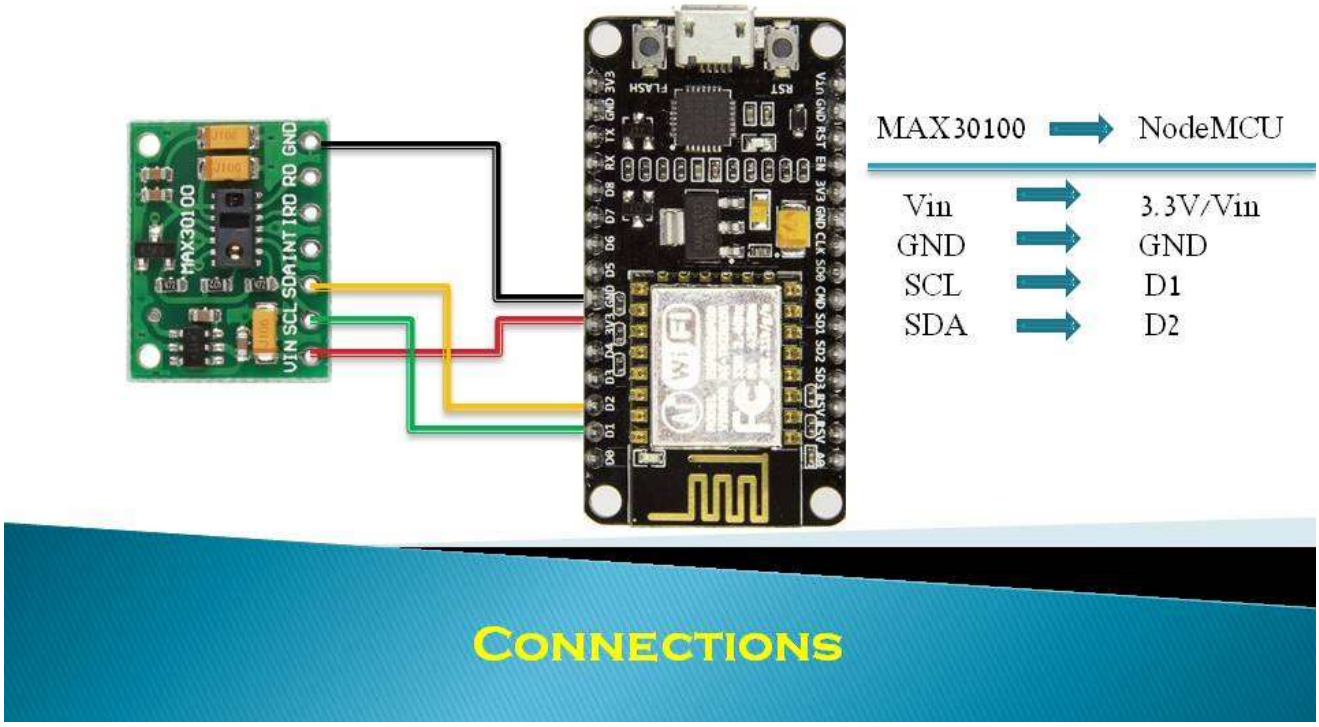


2.8.2 The circuit between the pulse rate sensor and the esp8266:

Steps of connection:

1.Connect the VCC pin of the pulse rate sensor module to the +5V pin on the ESP8266 development board.
2.Connect the GND pin of the pulse rate sensor module to any GND pin on the ESP8266 development board.
3.Connect the SDA pin of the pulse rate sensor module to the D1 (GPIO5) pin on the ESP8266 development board.

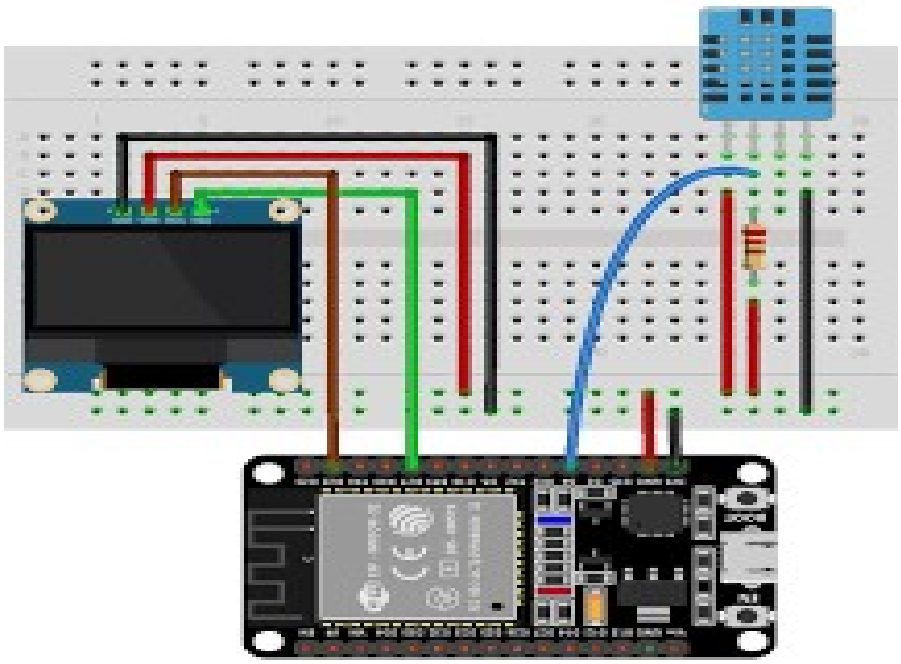
Once the connections are made, you can write the appropriate code to interface with the pulse rate sensor using the ESP8266. You'll need to utilize the I2C protocol to communicate with the sensor module and retrieve the pulse rate or heart rate data.



2.8.3 The circuit between the OLED, the DHT1, and MAX30100 sensors and the esp8266:

Steps of connection:

1.Connect the VCC pin of the DHT11 temperature sensor to the +5V pin on the ESP8266 development board.
2.Connect the GND pin of the DHT11 temperature sensor to any GND pin on the ESP8266 development board.
3.Connect the OUT pin of the DHT11 temperature sensor to digital pin D4 on the ESP8266 development board.
4.Connect the VCC pin of the pulse rate sensor module to the +5V pin on the ESP8266 development board.
5.Connect the GND pin of the pulse rate sensor module to any GND pin on the ESP8266 development board.
6.Connect the SDA pin of the OLED display module to the D1 (GPIO5) pin on the ESP8266 development board.
7.Connect the SCL pin of the OLED display module to the D2 (GPIO4) pin on the ESP8266 development board.



2.8.4 Circuit between servo motors and the esp8266:

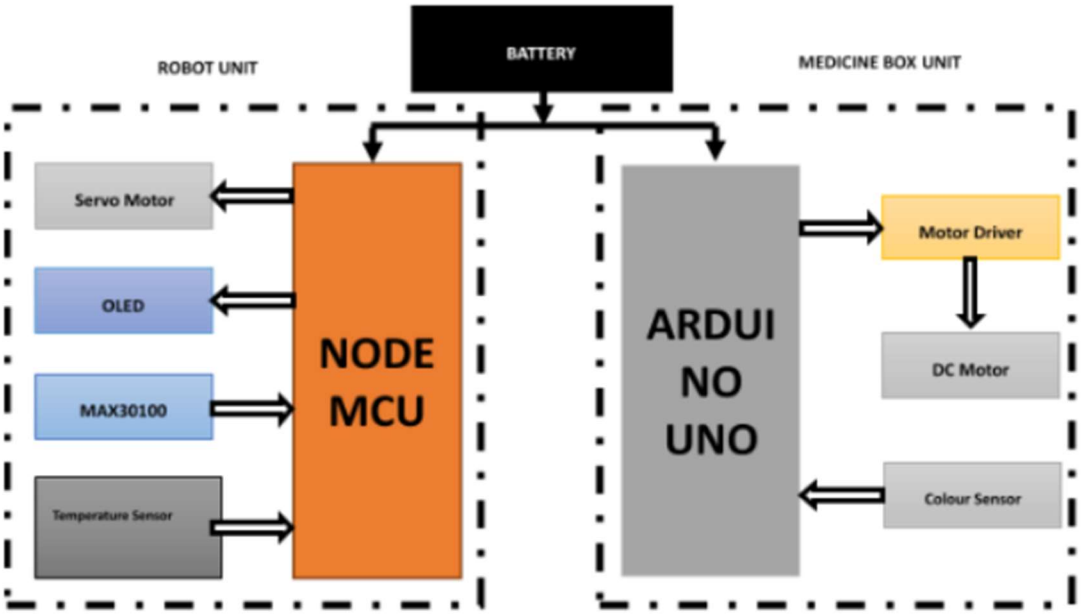
Steps of connection:

1.Connect the +5V pin of the first servo motor to the +5V pin on the ESP8266 development board.
2.Connect the GND pin of the first servo motor to any GND pin on the ESP8266 development board.
3.Connect the signal (SIG) pin of the first servo motor to digital pin D4 on the ESP8266 development board.
4.Connect the +5V pin of the second servo motor to the +5V pin on the ESP8266 development board.
5.Connect the GND pin of the second servo motor to any GND pin on the ESP8266 development board.
6.Connect the signal (SIG) pin of the second servo motor to another available digital pin on the ESP8266 development board.

Once the connections are made, you can write the appropriate code to control the servo motors using the ESP8266. You'll need to use the servo library and appropriate functions in your code to control the angle and movement of the servo motors.

2.9 Node-MCU and Arduino Car diagram

This is the diagram between the Node-MCU that is responsible for the IoT technology and connecting all the part of the medico box together and connecting them to the internet and the cloud server, and the robot unit this robot unit containing the medico box itself and its components and this diagram describes the inputs and outputs of each component and the circuit of it to the Arduino UNO or the Node-MCU



Chapter 3: Route Construction

At first, we need to construct the route for the Arduino car:

We have four colors, and the steps will be as follows:

1.Red color for forward moving

2.Green color for the turn left

3.Yellow color for the patient's destination

4.Black color for the Arduino car source

3.1 Color route algorithm:

The Arduino car will determine the color by the color sensor, so the robot unit at the start will be on the black color that mean the robot unit in its source, so the robot unit wait for the order from the application that with the patient's doctor, at this moment the car will start the move from the source to the patient's destination, the color sensor if detect red that mean the robot unit will move forward, and still move forward until detect different color for example if the color sensor detect the green color then the action will be turn left, and if the color sensor detect the yellow color then the robot unit will stop the detected box will open and the robot unit gives alert to notify the patient to take the medicine, after patient take the medicine the robot unit will close the box and starting the journey to go back to the source again, and waiting for the new order.

The robot unit colored route:

3.2 Abstract of the colored route:

1.If the color sensor detects red color the robot unit will take the action to move forward until reads a different color.

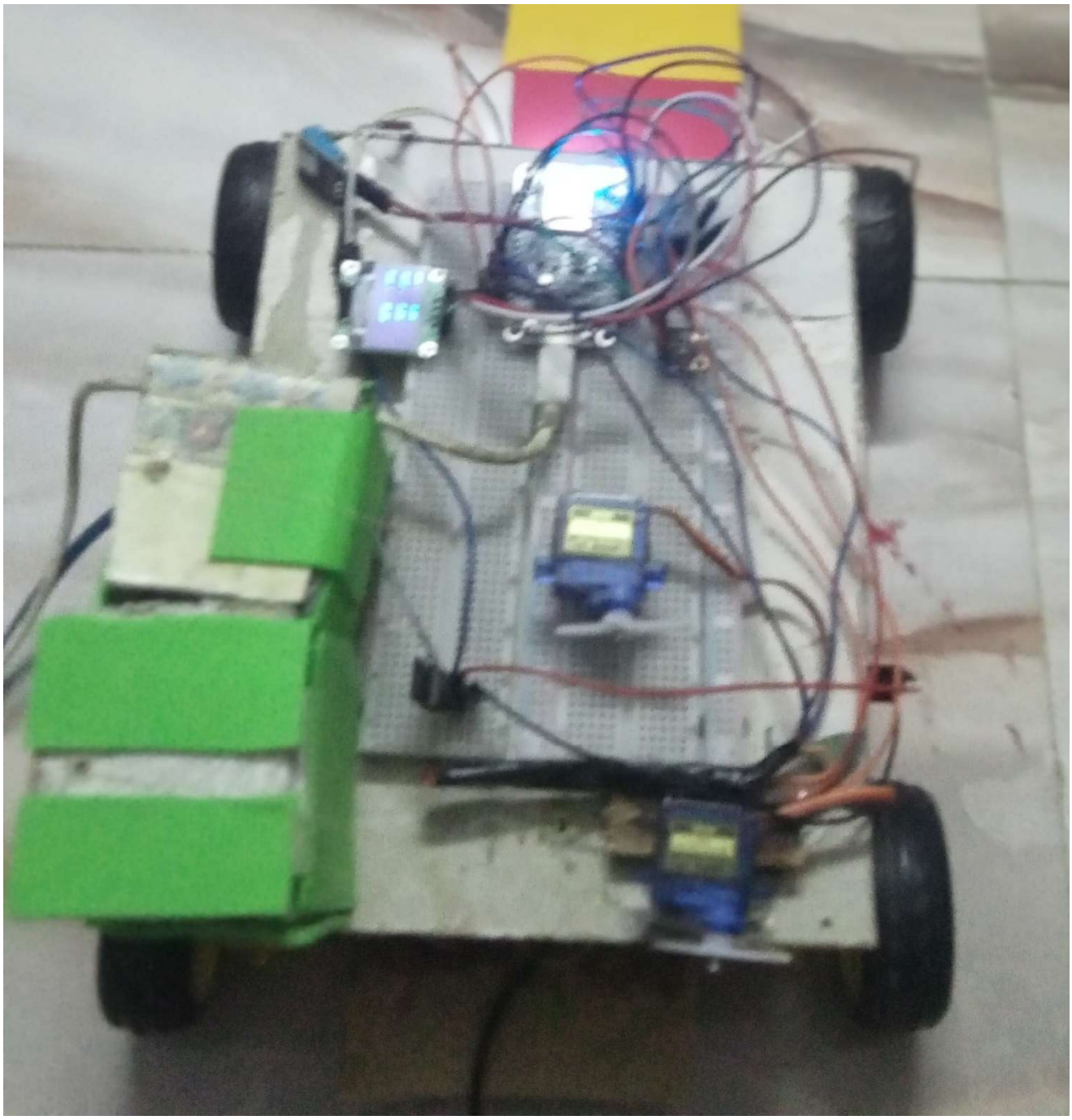
2.If the color sensor detects green color the robot unit will take the action to turn left and after it turned left the robot unit should read red color to move forward again.

3.If the color sensor detects yellow color the robot unit will stop and that mean it reached the patient's destination and gives the patient an alert to notify the patient to take its medicine.

4.If the color sensor detects black color, then the robot unit will stop and waiting for the new order from the doctor.

3.3 Full robot unit (product in complete case):

This is the robot unit in the completed case and the final shape:



Chapter 4: Project components

Arduino UNO:

Arduino's processor essentially makes use of the Harvard structure in which this system code and application statistics have separate reminiscence. The code is saved inside the flash application reminiscence, while the statistics is saved within the side of the statistics reminiscence. The Atmega328 has 32 KB of flash reminiscence for storing code, 2 KB of SRAM, and 1 KB of EEPROM and operates with a clock pace of 16MHz. The Arduino software program is well-matched with all styles of running structures like Windows, Linux, Macintosh, etc.

Battery (Power supply):

In electricity, a battery is a tool consisting of 1 or more electrochemical cells that convert saved chemical electricity into electric electricity. An unusual place electricity supply for lots of families and business applications. There are all sorts of batteries: number one batteries and secondary batteries.
A 6V battery is a lead-acid kind cell. It is likewise referred to as a lantern battery. It commonly makes use of 4 large. 6V batteries are utilized in canine schooling devices, clinical instruments, movie and virtual cameras, and plenty of different devices. A battery is a tool that converts chemical power immediately to electric power. It includes some voltaic cells; every voltaic molecule includes half-cells linked in collection through a conductive electrolyte containing anions and cations. One half10 molecular consists of an electrolyte and the electrode to which anions migrate, and the opposite half-molecular consists of an electrolyte and the electrode to which cations migrate. Some battery cautions are connecting the charger, initial, bulk charge mode, absorption charge mode, and float charge. Battery types are lead-acid batteries, disposable batteries, and solar-powered batteries.

Node-MCU:

Node MCU is an open supply IoT platform. It makes use of many open supply projects, which includes Launceston, and spiffs. It consists of firmware that runs at the ESP8266 Wi-Fi, and hardware that's primarily based totally on the ESP-12E module.ESP-12E is designed and evolved via way of means of Shenzhen Doctors of Intelligence & Technology (SZDOIT) primarily based totally on the Ultra-low energy intake UART-Wi Fi ESP8266, that's mainly for cellular gadgets and alertness of IoT. Now, ESP-12E is broadly carried out to the internet, verbal exchange in the 11-neighborhood area, clever home, business control, handed-gadgets, etc.ESP-12E Devitt has used the layout of the onboard antenna and encapsulated via way of means of 2. fifty-four direct insertions. It may be very handy to debug and deplume the device. IN ESP12E Devitt, Hardware API operation is encapsulated via way of means of Lau language, which can keep away from the hardware trouble for software program engineers, after which can velocity the expansion of products. This is simply the ESP-12 chip. If you're seeking out the breakout board with onboard regulator and preferred header compatibility.

Servo motors:

A servo motor is an electrical tool that can push or rotate an item with terrific precision. If you need to rotate the item at a few precise angles or distances, you then definitely use a servo motor. It is simply made of an easy motor which runs via servo mechanism. Basically, servo cars are labeled AC and DC servo cars relying upon the character of delivery used for its operation. Brushed everlasting magnet DC servo cars are used for easy programs attributable to their cost, performance, and simplicity. A servo includes a Motor (DC or AC), a potentiometer, tools meeting, and a controlling circuit. First, we use tools meeting to lessen RPM and to boom torque of motor such that there may be no electric sign generated on the output port of the potentiometer. Servomotor features are linear dating among the rate and electric powered manage signal., Steady country stability, Wide variety of pace management. The advantages of a servomotor are higher output than a 50Hz motor of the same size.

DC-Motors with wheels:

There is no explicit mention of a DC motor with wheels. However, I did mention "Robot car chassis with motor drivers" under the "Components Required" section. A robot car chassis typically includes DC motors with wheels attached to them. These motors are used to drive and control the movement of the robot car.

Motor Driver (L293D) dual module:

The L293D is designed to offer bidirectional force currents of as much as 600mA at voltages from four. five V to 36 V. gadgets are designed to force inductive masses inclusive of relays, solenoids, dc, and bipolar stepping motors, in addition to different high-current/high-voltage masses in positive-deliver applications. All inputs are TTL compatible. Each output is a whole totem-pole force circuit, with a Darlington transistor sink and a pseudo-Darlington source. Drivers are enabled in pairs, with drivers 1 and a couple of enabled through 1,2EN and drivers three and four enabled through three,4EN.

OLED screen:

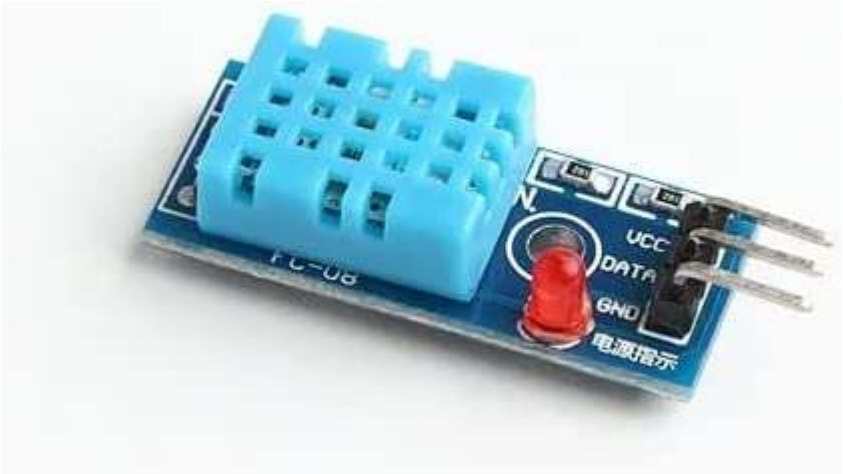
Used for monitoring local monitoring for the patient.

TCS Color sensor:

Used to detect the color of the track.

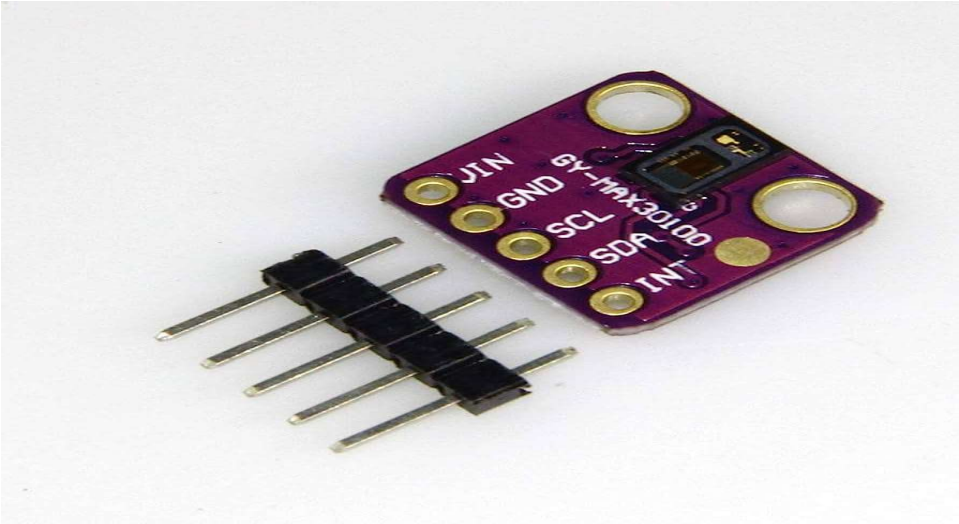
DHT11 temperature and humidity sensor:

This sensor is used for sense the temperature of the patient and the temperature of the room.



Pulse rate or heart rate and oxygen rate sensor:

The MAX30100 operates from 1.8V and 3.3V power supplies and can be powered down through software with negligible standby current, permitting the power supply to always remain connected.



IoT platform (Blynk platform):

Internet of Things (IoT):
The internet of things, or IoT, is a tool of interrelated computing devices, mechanical and digital machines, objects, animals, or people which may be furnished with identifiers (UIDs) and the potential to interchange data over a network without requiring human-to-human or human-to-laptop interplay. An element within the internet of things can be a person with a coronary heart display screen implant, a farm animal with a biochip transponder, a car that has included sensors to alert the reason pressure whilst tire pressure is low or each different natural or man-made object that can be assigned an Internet Protocol (IP) address and is able to transfer data over a network. Increasingly, businesses in some industries are using IoT to carry out greater efficiency, better understand customers to deliver stepped-forward client service, decorate decision-making and growth the charge of the enterprise. How does IoT artwork? An IoT environment consists of web-enabled smart devices that use embedded systems, which include processors, sensors and communique hardware, to accumulate, deliver and act on data they accumulate from their environments. IoT devices percent the sensor data they accumulate with the
resource of the usage of connecting to an IoT gateway or exceptional facet device in which data is each sent to the cloud to be analyzed or analyzed locally. , the ones devices talk with exceptional related devices and act on the information they get from one another. The devices do most of the artwork without human intervention, regardless of the reality that people will interplay with the devices — for instance, to set them up, deliver instructions or get proper access to the data. The sizeable set of packages for IoT gadgets is divided into numerous packages. These are the subsequent packages particularly customer application, clever domestic, clinical and tech care, transportation, constructing and domestic automation, commercial application, manufacturing, agriculture, infrastructure application, electricity control and environmental monitoring. While the idea of IoT has been in lifestyles for a prolonged time, a hard and fast of recent advances in several of an incredible era has made it practical. They are Access to low-cost, low energy sensor era, Connectivity, Cloud computing platforms, Machine learning and analytics, and Conversational artificial intelligence (AI). Industrial IoT refers to the application of the IoT era in enterprise settings, especially with understanding instrumentation and manipulation of sensors and devices that interact with cloud technologies. Recently, industries have used gadget-to-gadget communication (M2M) to gain wireless automation and manipulation.

BLYNK App:

Blynk is a brand-new platform that lets you fast construct interfaces for controlling and tracking your hardware initiatives out of your iOS and Android device. The primary recognition of the Blynk platform is to make it super smooth to increase the cell telecall smartphone application. As you may see on this course, growing a cell app that may speak in your Arduino is as smooth as dragging a widget and configuring a pin. With Blynk, you could manage an LED or a motor out of your cell telecall smartphone with 0 programs. This is truly the primary test that I will show on this course. But don’t permit this simple to make you believe you studied that Blynk is simplest beneficial for trivial applications. Blynk is a strong and scalable device this is utilized by hobbyists and the enterprise alike.,. You also can use it to manipulate clever fixtures that may study out of your routines. Blynk is loose to apply for nonpublic use and prototyping. Their commercial enterprise version generates earnings with the aid of promoting subscriptions to corporations that need to put up Blynk-powered apps for their hardware merchandise or services.

The BLYNK Smartphone App:

The Blynk app is absolutely an app editor. It permits you to create one or extra. projects. Each mission can comprise graphical widgets, like digital LEDs, buttons, fee presentations or even a textual content terminal, and might interact with one or extra devices. With the assistance of the Blynk library, it's far more viable to govern Arduino or ESP32 pins immediately out of your phone, while not having to write down any code at all.

The BLYNK Server:

The Blynk Cloud server is a fantastic desire for maximum projects, as it's far from usually there, prepared to use. We will use the Cloud server withinside the first few experiments on this route that will help you get began out with minimum effort. However, as you may see, the Cloud Blynk server has imposed barriers. Some barriers are because of the topology of the server: relying on your geographical location. Blynk is the usage of the idea of "electricity" to put into effect a pricing gadget for its widgets. In the Cloud server you can begin a brand-new mission with one thousand electricity gadgets. An LED widget might cost a little you two hundred gadgets, l easing 800 gadgets for the different widgets.

Node-MCU code:

```
#define BLYNK_TEMPLATE_ID "TMPLFw-ryhee"
#define BLYNK_TEMPLATE_NAME "DHT11 and pulse rate"
#define BLYNK_AUTH_TOKEN "g8Nw3ExM_q31mmx5PTheFnpszsGbYeWp"
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <DHT.h>
#include "MAX30100_PulseOximeter.h"
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <Servo.h>
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels
#define OLED_RESET    -1 // Reset pin # (or -1 if sharing Arduino reset pin)
#define SCREEN_ADDRESS 0x3C ///< See datasheet for Address; 0x3D for 128x64, 0x3C f
or 128x32
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
#define NUMFLAKES     10 // Number of snowflakes in the animation example
#define LOGO_HEIGHT   16
#define LOGO_WIDTH    16
char ssid[] = "OPPO A15";
char pass[] = "01234568";
char auth[] = "g8Nw3ExM_q31mmx5PTheFnpszsGbYeWp";
PulseOximeter pox;
uint32_t tsLastReport = 0;
#define REPORTING_PERIOD_MS     1000
float BPM, SpO2;
#define DHTPIN D3
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);
BlynkTimer timer;
Servo servo1;
Servo servo2;
BLYNK_WRITE(V4)
{
        servo1.write(param.asInt());
        int pos1 = param.asInt();
}
BLYNK_WRITE(V5)
{
        servo2.write(param.asInt());
        int pos2 = param.asInt();
}
```

```
void sendSensor() {
float h = dht.readHumidity();
float t = dht.readTemperature();
display.setTextSize(1);
display.setTextColor(1);
display.setCursor(0, 45);
display.println("Temp :");
display.setTextSize(1);
display.setTextColor(1);
display.setCursor(60, 45);
display.println(t);
display.display();
if (isnan(h) || isnan(t)) {
  Serial.println("Failed to read from DHT sensor!");
        return;
}
Blynk.virtualWrite(V0, h);
Blynk.virtualWrite(V1, t);
}
void onBeatDetected() {
Serial.println("Beat!");
}
void setup()
{
Serial.begin(115200);
        If (!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS)) {
        Serial.println(F("SSD1306 allocation failed"));
}
pinMode(sendDataPin, OUTPUT);
servo1.attach(D4);
servo2.attach(D5);
display.setTextSize(1.5);
display.setTextColor(1);
display.setCursor(0, 0);
display.println("Initializing pulse oximeter..");
display.display();
Blynk.begin(auth, ssid, pass);
if (!pox.begin()) {
Serial.println("FAILED");
display.clearDisplay();
display.setTextSize(1.5);
display.setTextColor(1);
display.setCursor(0, 0);
display.println("FAILED");
display.display();
}
else{
        display.clearDisplay();
```

```
 display.setTextSize(1.5);
 display.setTextColor(1);
 display.setCursor(0, 0);
 display.println("SUCCESS");
 display.display();
 Serial.println("SUCCESS");
 }
 pox.setOnBeatDetectedCallback(onBeatDetected);
 dht.begin();
 timer.setInterval(1000L, sendSensor);
}
void loop() {
        pox.update();
        if (millis() - tsLastReport > REPORTING_PERIOD_MS) {
        BPM = pox.getHeartRate();
        SpO2 = pox.getSpO2();
        Serial.print("BPM: ");
        Serial.println(BPM);
        Serial.print("SpO2: ");
        Serial.print(SpO2);
        Serial.println("%");
        Serial.println("**********************************");
        Serial.println();
        display.clearDisplay();
        display.setTextSize(1.5);
        display.setTextColor(1);
        display.setCursor(60, 0);
        display.println(BPM);
        display.setTextSize(1.5);
        display.setTextColor(1);
        display.setCursor(0, 0);
        display.println("BPM :");
        display.setTextSize(1.5);
        display.setTextColor(1);
        display.setCursor(0, 20);
        display.println("Spo2 :");
        display.setTextSize(1.5);
        display.setTextColor(1);
        display.setCursor(60, 20);
        display.println(SpO2);
        display.display();
        tsLastReport = millis();
        Blynk.virtualWrite(V2, BPM);
        Blynk.virtualWrite(V3, SpO2);
        }
         Blynk.run();
        timer.run();
}
```

Arduino robot unit code:

```
 #define M1 13
#define M2 12
#define M3 A0
#define M4 A1
#define M5 A2
#define M6 A3
#define M7 A4
#define M8 A5
#define PWM1 3
#define PWM2 9
#define PWM3 10
#define PWM4 11
#define S0 4
#define S1 5
#define S2 6
#define S3 7
#define sp 210
#define sensorOut 8
#define buz 2
int red = 0;
int green = 0;
int blue = 0;
void setup() {
Serial.begin(9600);
pinMode(buz, OUTPUT);
pinMode(S0, OUTPUT);
pinMode(S1, OUTPUT);
pinMode(S2, OUTPUT);
pinMode(S3, OUTPUT);
pinMode(PWM1, OUTPUT);
pinMode(PWM2, OUTPUT);
pinMode(PWM3, OUTPUT);
pinMode(PWM4, OUTPUT);
pinMode(M1, OUTPUT);
pinMode(M2, OUTPUT);
pinMode(M3, OUTPUT);
pinMode(M4, OUTPUT);
pinMode(M5, OUTPUT);
pinMode(M6, OUTPUT);
pinMode(M7, OUTPUT);
pinMode(M8, OUTPUT);
pinMode(sensorOut, INPUT);
//scaling to 20 %
digitalWrite(S0, HIGH);
```

```
digitalWrite(S1, LOW);
}
void loop() {
        color();
        if ( (( red >= 80) && ( red <= 100)) && (( green >= 260)&& (  green <= 280)) && ((
        blue >= 180) && ( blue <= 200 )))
        {
                Serial.println("RED");
                analogWrite(M1, 255);
                analogWrite(M2, 0);
                analogWrite(PWM1, 50);
                 analogWrite(M3, 255);
                analogWrite(M4, 0);
                analogWrite(PWM2, 50);
                digitalWrite(M5, 255);
                digitalWrite(M6, 0);
                analogWrite(PWM3, 50) ;
                digitalWrite(M7, 255);
                digitalWrite(M8, 0);
                analogWrite(PWM4, 50) ;
}
if ( (( red >= 240) && ( red <= 260)) && (( green >= 210) && (  green <= 230) && (( blue
>= 210) && ( blue <= 230)) ))
{
        Serial.println("GREEN");
        digitalWrite(M1, HIGH);
        digitalWrite(M2, LOW);
        analogWrite(PWM1, 180);//MOTOR 1
        digitalWrite(M3, LOW);
        digitalWrite(M4, HIGH);
        analogWrite(PWM2, 180); //MOTOR 2
        digitalWrite(M5, LOW);
        digitalWrite(M6, HIGH);
        analogWrite(PWM3, 180); //MOTOR 3
        digitalWrite(M7, HIGH);
        digitalWrite(M8, LOW);
        analogWrite(PWM4, 180) ; //MOTOR 4
}
if ( (( red >= 60) && ( red <= 80)) && (( green >= 110) && (  green <= 130) && (( blue >=
160) && ( blue <=180)) )
{
        Serial.println("YELLOW");
        digitalWrite(M1, LOW);
        digitalWrite(M2, LOW);
        digitalWrite(M3, LOW);
        digitalWrite(M4, LOW);
        digitalWrite(M5, LOW);
        digitalWrite(M6, LOW);
```

```
        digitalWrite(M7, LOW);
        digitalWrite(M8, LOW);
        tone(buz, HIGH);
        delay(500);
        noTone(buz, LOW);
        delay(500);
        noTone(buz, LOW);
        delay(10000);
        digitalWrite(M1, HIGH);
        digitalWrite(M2, LOW);
        analogWrite(PWM1, 180); //MOTOR 1
        digitalWrite(M3, HIGH);
        digitalWrite(M4, LOW);
        analogWrite(PWM2, 180); //MOTOR 2
        digitalWrite(M5, HIGH);
        digitalWrite(M6, LOW);
        analogWrite(PWM3, 180); //MOTOR 3
        digitalWrite(M7, HIGH);
        digitalWrite(M8, LOW);
        analogWrite(PWM4, 180) ; //MOTOR 4
        delay(2500);
}
if ( (( red >= 300) && ( red <= 330)) && (( green >= 360) && (  green <= 390)) && (( blue
>= 280) && ( blue <= 310)))
{
        Serial.println("BLACK");
        digitalWrite(M1, LOW);
        digitalWrite(M2, LOW);
        digitalWrite(M3, LOW);
        digitalWrite(M4, LOW);
        digitalWrite(M5, LOW);
        digitalWrite(M6, LOW);
        digitalWrite(M7, LOW);
        digitalWrite(M8, LOW);
}
}
void color() {
        // Setting RED (R) filtered photodiodes to be read
        digitalWrite(S2, LOW);   digitalWrite(S3, LOW);   // Reading the output frequency
        red = pulseIn(sensorOut, LOW); delay(100); // Setting GREEN (G) filtered
        photodiodes to be read   digitalWrite(S2, HIGH);   digitalWrite(S3, HIGH);
        // Reading the output frequency  green = pulseIn(sensorOut, LOW); delay(100);
        // Setting BLUE (B) filtered photodiodes to be read  digitalWrite(S2, LOW);
        digitalWrite(S3, HIGH); // Reading the output frequency
blue = pulseIn(sensorOut, LOW);
delay(100);
}
```

References:

https://www.circuito.io/app?components=512,10167,11021
https://circuitdigest.com/search/node?keys=MAX30100+
https://how2electronics.com/interfacing-max30100-pulse-oximeter-sensor arduino/
https://www.electronicsforu.com/electronics-projects/iot-based-health-monitoringsystem-medicine-box