Journal of Advanced Research in Computing and Applications 42, Issue 1 (2026) 46-60

KARYA ILHAM PUBLISHING
CONNECTING MINDS, BRIDGING IDEAS, INSPIRING READERS

Journal of Advanced Research in Computing and Applications

Journal homepage:
https://karyailham.com.my/index.php/arca/index
ISSN: 2462-1927

Smart Medical Assistant Robot: A Contactless Solution For Monitoring Patient Health And Enforcing Pandemic Safety Measures

Haris Murshidi Mat Isa[1], Nor Diyana Md Sin[2,*], Muhammad Amir Syazwan Tukiman[3], Mohamad Zhafran Hussin[4], Norlina Md Zain[5], Muhammad Muzamil Mustam[6], Fazlinashatul Suhaidah Zahid[7] Norhalida Othman[8]

[1] Faculty of Electrical Engineering, Universiti Teknologi MARA, Cawangan Johor, Kampus Pasir Gudang, Johor, Malaysia

ARTICLE INFO	ABSTRACT

Article history:
Received 9 September 2025
Received in revised form 23 November 2025
Accepted 3 January 2026
Available online 3 February 2026

The COVID-19 pandemic has contributed to a sharp focus on innovation that will allow for reducing the number of human contacts and providing sufficient patient care and safety measures. This paper presents a Smart Medical Assistant Robot that can be utilized to solve the major issues in the healthcare facility: social distancing, the use of Personal Protective Equipment (PPE), and staffing. With an Arduino Uno R3 microcontroller, infrared, pulse, and temperature sensors, the robot can navigate hospital areas autonomously, detect the vital signs of patients (heart rate and body temperature), and wirelessly send the information to the devices of healthcare providers using a Wi-Fi module. Experimental analysis showed that the temperature sensor had 94% (within the range of ±0.5 °C to reference of ±0.3°C) accuracy, and the pulse rate measurement had 97% accuracy (within the range of ±3 BPM to reference of ±2 BPM). The robot was successful in 95 percent of path-following experiments (19 out of 20) and averagely worked 5 hours before having to recharge the battery. It is worth noting that through automating the regular checks of health and imposing the safety protocols against the pandemic, this contactless solution reduces the chances of the virus spreading and promotes environmental sustainability and cost-efficiency. In line with this, the suggested system can be shown to reduce close physical contact in healthcare facilities to help in safer working conditions during the outbreak of infectious diseases.

Keywords:
Minimum three keywords; Assistant Robot; Smart Medical; Contactless; Monitoring Patient; COVID-19

1. Introduction

The COVID-19 pandemic, or SARS-Cov-2 virus, is a challenge in the systems of global health, economics, and environmental sustainability that has never been seen in the past. The pandemic has prompted the world to take immediate actions to reduce its transmission, which has necessitated social distancing, wearing Personal Protective Equipment (PPE), and observing strict hygiene guidelines, which have been declared a worldwide crisis by the World Health Organization (WHO) in January 2020 [1][2]. Nonetheless, these actions have also brought to light such problems as the high

* *Corresponding author.*
E-mail address: diyana0366@uitm.edu.my

https://doi.org/10.37934/araset.42.1.4660

risk of close interaction between health professionals and patients, the environmental footprint of disposable PPE, and the overworking of medical workers. These obstacles highlight the necessity of new approaches to help prevent further virus spreading, protect healthcare workers and minimize environmental damage [3][4].

This study will offer to meet these challenges by proposing the creation of a Smart Medical Assistant Robot, which will reduce physical contact between medical professionals and patients, and thus have a possible chance of preventing the transmission. The robot is based on advanced technologies, such as Arduino Uno R3 microcontrollers, infrared sensors, temperature sensors and pulse sensors to independently navigate hospital settings and conduct contactless patient monitoring. The robot is capable of transporting drugs, testing vital indicators like the heart rate and body temperature, and sending information wirelessly to the healthcare professionals using a Wi-Fi chip by following a predetermined path on a tape. Additionally, the strategy should serve to assist patient care and alleviate the workload of overloaded medical personnel, especially in high-risk settings like COVID-19 wards [5][6].

The latest developments in medical robotics indicate the continued growth of artificial intelligence (AI) and automation in healthcare delivery and precision medicine. An example is the creation of smart robots like *Dr. HEMA*, which can help medical staff in the diagnosis of chronic diseases with high precision due to the use of AI and machine learning algorithms to interpret complicated medical data, thus increasing the rate of diagnosis, accuracy, and consistency in diagnosis [7]. On the same note, smart hospitals that have automated Intelligent Speech Technology (IST) have enhanced medical records, diagnosis of diseases, and communication between the medical staff and the medical devices, which have played a significant role in the early detection, rehabilitation, and treatment of diseases like stroke [8]. Simultaneously, the robotic surgery field is growing at an extremely rapid pace, with recent trends focusing on the incorporation of AI that would allow a surgical robot to become more precise in its actions and achieve the outcomes (and potentially be fully autonomous) (Level 5) [9][10]. In addition, even unique robotic systems like those in spine surgery have demonstrated significant enhancements in screw placement and precision in puncture methods, which further boosted clinical safety, accuracy and practicality [11]. All these new innovations highlight the disruptive role of intelligent robotic systems in health care today and the necessity of further study of cost-effective, adaptive medical robots like the Smart Medical Assistant Robot suggested in this paper, in addition to aiding clinical practice.

This project is influenced by some of the current technologies and ideas in robotics and healthcare. As an example, a smart Internet of Things (IoT) and deep learning-based robot was introduced by Mario Dias et al. to offer first aid and emergency support to people, especially the elderly, who live alone. The robot is able to sense distress by the audible screams and physical sight, locates the victim and gives aid or calls an emergency in case of necessity. It was tested in a home setup, and the parameters of evaluation were the minimization of the activation, search, and response time [12]. Likewise, Nanditha Krishna et al. also presented PILLBOT, a non-contact system of dispensing medicine to minimize the contact of health workers with COVID-19 patients. Being operated by voice commands and cloud services, PILLBOT dispenses pills and syrup and guarantees having minimal direct contact [13]. In the meantime, Md Abir Hossen et al. wrote about a mobile Automated Medical Assistant (AMA) that was deployed in a Bangladeshi hospital to assist medical staff with routine tasks. The robot has user-friendly navigation, collision avoidance, and energy efficiency and was effectively tried in a hospital, showing that it could help to optimize healthcare efficiency [14]. On the whole, these experiments can be a powerful guide to the design and implementation of a Smart Medical Assistant Robot, which will be an autonomous robot with built-in health monitoring capabilities. Although the earlier systems like PILLBOT [13] and AMA [14] had good prospects, they could not be applied in a small clinic due to the cost of implementation, reliance on complicated cloud architecture, and lack of adaptability. The Smart Medical Assistant Robot is proposed, and the proposed robot is characterized by low costs of development because of the Arduino-based design, flexible IoT integration, and scalable sensor structure, which makes it possible to implement the robot within resource-intensive healthcare facilities.

Moreover, the present project deals with the ecological issues that are related to the extensive use of disposable personal protective equipment. Sustainability is the main focus of the project as it aims to decrease the use of single-use protective equipment that leads to a major contribution to plastic waste and environmental pollution. Following this, automation of routine operations and requiring very minimal human contact, which is provided by the robot, makes its solution to the pandemic challenges both cost-effective and eco-friendly. Therefore, this paper will show how such robots can be viable and effective in helping to deliver healthcare, secure front-line employees, and ensure environmental sustainability during and after the COVID-19 pandemic.

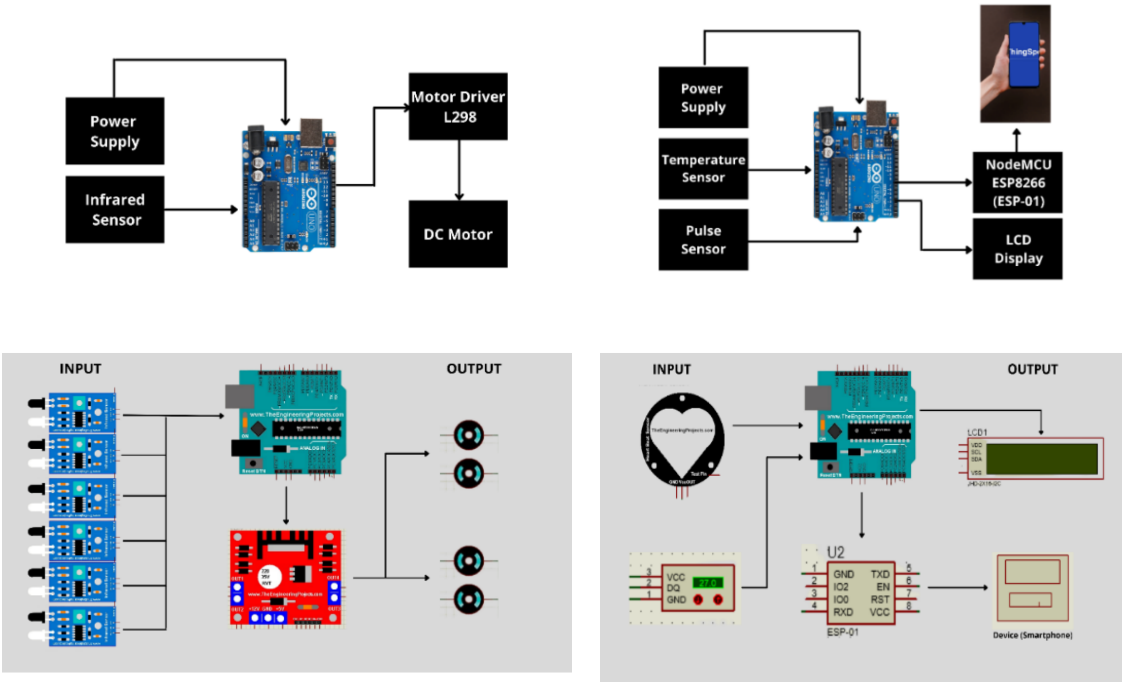
2. Methodology

2.1 System Diagram

The Smart Medical Assistant Robot is an engineering prototype that the design was based on functional design and that which was validated in Proteus simulation. It comprises two subsystems built in, one is a line following navigation unit with infrared sensors and motor drivers, and the other one is the health monitoring unit with pulse and temperature sensors.

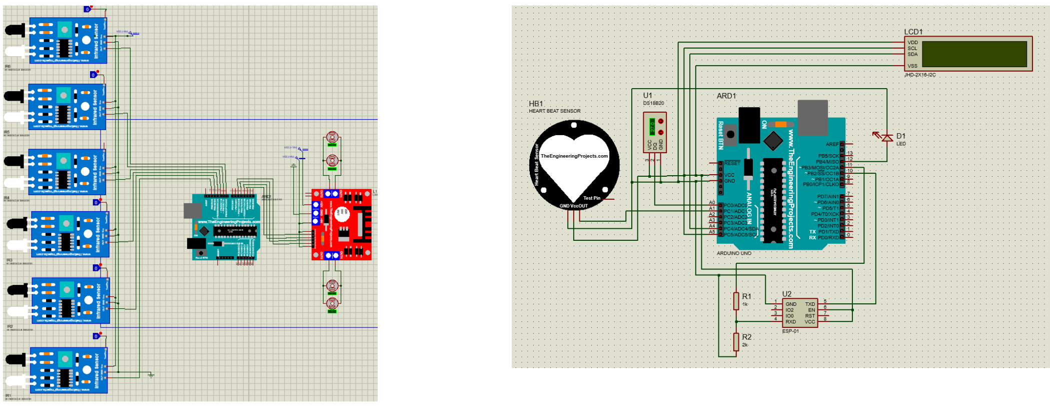
Figure 1(a) is the block diagram of the robot line follower, which is a description of the relationships between components, i.e., infrared sensors, motor driver (L298N), direct current (DC) motors, pulse sensor, temperature sensor (DS18B20), as well as Wi-Fi module (ESP8266). Specifically, the reflectance between the tape line and the rest of the surface is detected through the infrared sensors that are positioned in a linear array, allowing for precise navigation. Figure 1(b) represents the block diagram of the subsystem of heartbeat and temperature sensor monitoring, which includes pulse and temperature sensors which are connected to the Arduino to be processed and displayed. This capacity to be modular means that there will be smooth communication between the hardware and software control.

The schematic diagram of the robot line follower is presented in Figure 2 (a) and was done using ProTeus software. The line-following subsystem uses six infrared sensors connected to the digital pins of the Arduino, but the motor driver is linked to the DC motors to allow the subsystem to move. In the meantime, Figure 2 (b) is a scheme of the diagram of heartbeat and temperature sensors created in Proteus software. The health monitoring subsystem connects the pulse and temperature sensors to the Arduino and displays the outputs on the liquid-crystal display (LCD) and sends the data through the Wi-Fi module. Such a schematic design guarantees appropriate wiring and functionality, as well as there are reduced errors during their assembly.



(a) (b)

Fig. 1. (a) Robot Line Follower Block Diagram and (b) Heartbeat and temperature sensors Block Diagram



(a) (b)

Fig. 2. (a) Robot Line Follower schematic diagram using proteus software and (b) Heartbeat and temperature sensors schematic diagram using proteus software

Lastly, Figure 3 presents the flow chart that illustrates the logic of operational work of the system and begins with sensor inputs (infrared, pulse, and temperature) and ends with the output (motor movement and health data display). This step-by-step visualization will make sure the robot functions according to the program, including the navigation to the patient bed and providing vital signs measurements. Collectively, the aforementioned methodologies offer a sound methodology to develop and execute the project and make it achieve the objectives of the project, which are a possible reduction of physical contact, the need to monitor patients, and to assist the healthcare workers in the fight against COVID-19.

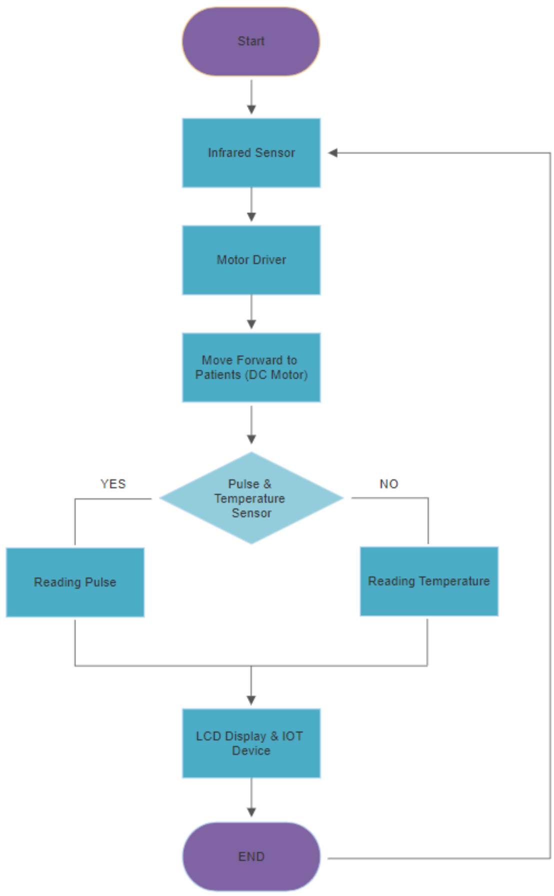


Fig. 3. Robot Line Follower schematic diagram using proteus software

2.2 System Development

The section of the programming code that was created to aid the operation of the line-following of the Smart Medical Assistant Robot is shown in Figure 4. The program starts with the incorporation of the L298N motor driver library and the declaration of both the input and output pins on the Arduino microcontroller of the DC motors and the infrared (IR) sensors. Under the setup section, every pin is initialized with the help of the *pinMode()* function that determines how it will be operated. The main logic section interprets digital signals of several IR sensors under the bottom of the robot, with each sensor reading and giving a logic value of “1” for a black line and “0” for a white surface. According to these readings, conditional statements are implemented to control the motor movements such that when the center sensors are triggered, the robot will move forward, when the side sensors are triggered, the robot will turn left or right, respectively. The pulse width modulation (PWM) control is used to control motor speed and direction so that the paths followed are smooth and properly tracked. This program allows the robot to travel independently in a given set route by constantly processing the sensor responses and adjusting the movements of the motors accordingly.

The following Figure 5 demonstrates a part of the code of the pulse and temperature sensing modules built into the Smart Medical Assistant Robot. The program is launched with the necessary libraries *OneWire, DallasTemperature, LiquidCrystal_I2C* and *ThingSpeak*, which allow interaction of the microcontroller with the DS18B20 temperature sensor, pulse sensor, LCD display, and the IoT platform. Every input pin is configured and declared with *pinMode()* to be sure that it acquires data correctly. The LCD start-up has a sequence of pre-programmed instructions that instruct the user to place their finger on the pulse sensor and their arm close to the temperature sensor. Data from the pulse sensor is taken in *analogRead()*, and timing variables are used, *sampleCounter* and *lastBeatTime,* to record the interval of the heartbeats and reduce the noise interruptions. The computed heart rate is shown in beats per minute (BPM) and also represented by the blinking of an LED with each beat detected. In temperature measurement, the *DallasTemperature* library asks the sensor to provide the data of the sensor at index 1, that is, the first sensor that is connected using the *ByIndex* function, which reads the value of the first sensor. The temperature values are computed and displayed in both Celsius and Fahrenheit units. Once all readings are acquired, the results are shown on the LCD and transmitted to the ThingSpeak IoT platform for cloud storage and real-time monitoring.

Fig. 4. Partial programming code for the line-following robot, showing the sensor-based logic used to control motor movements

Fig. 5. Partial programming code for the pulse and temperature sensing modules, showing library inclusion, sensor initialization, and data transmission to the ThingSpeak IoT platform

Journal of Advanced Research in Computing and Applications
Volume 42, Issue 1 (2026) 46-60

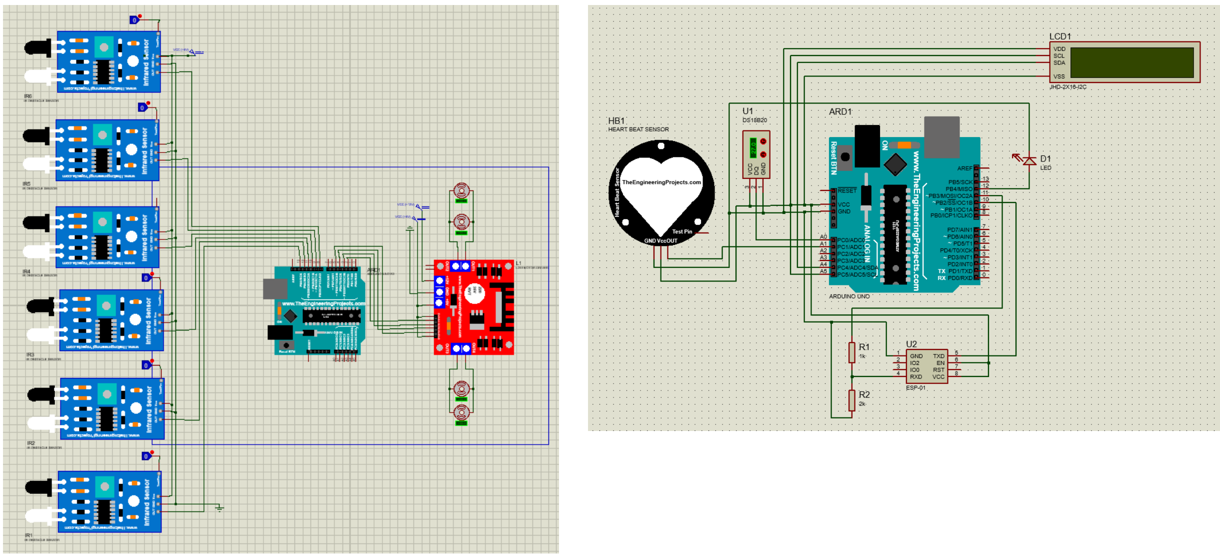
Figure 6. Partial programming code showing the pulse detection algorithm and the subsequent transmission of data to the ThingSpeak IoT platform. This IoT ThingSpeak is functions with the Wi-Fi Module ESP-01. For the IoT ThingSpeak, the program starts with declaration of ThingSpeak. file, where this file is a cloud based analytic IoT platform that able to gather, view, and analyze live data streams. This library can work with the avr, esp8266, sam, samd, esp32, samd_beta, and megaavr. Then, to functions the IoT ThingSpeak, the apiKey has been put in the starting of the IoT ThingSpeak. The apiKey obtained from the channel that has been made in the IoT ThingSpeak application. Next, the program of Wi-Fi Module ESP-01 begins with the declaration of TCP connection, where put the address code of IoT ThingSpeak in order to ESP-01 read the IoT ThingSpeak. This address code can be obtained on ThingSpeak website. Following that, on the ESP-01 program, there are included to put the network ID and password of home Wi-Fi or personal phone Hotspot in order to turns on the ESP-01. The data of temperature and pulse rate that has been transferred to the IoT ThingSpeak has some delay about 15 second order to get the right reading. The program concludes each cycle with an automatic reset and countdown display, preparing the system for the next measurement.

```
    case 11:
    Serial.println("--------------|----------------------");
    break;
  }
}

void sendDataToSerial(char symbol, int data )
{
  Serial.print(symbol);
  Serial.println(data);
}
  ISR(TIMER2_COMPA_vect) //triggered when Timer2 counts to 124
  {
    cli(); // disable interrupts while we do this
    Signal = analogRead(pulsePin); // read the Pulse Sensor
    sampleCounter += 2; // keep track of the time in mS with this variable
    int N = sampleCounter - lastBeatTime; // monitor the time since the last beat to avoid noise
    // find the peak and trough of the pulse wave
    if(Signal < thresh && N > (IBI/5)*3) // avoid dichrotic noise by waiting 3/5 of last IBI
      {
        if (Signal < T) // T is the trough
        {
        T = Signal; // keep track of lowest point in pulse wave
        }
      }
        if(Signal > thresh && Signal > P)
        { // thresh condition helps avoid noise
        P = Signal; // P is the peak
        } // keep track of highest point in pulse wave
        // NOW IT'S TIME TO LOOK FOR THE HEART BEAT
        // signal surges up in value every time there is a pulse
        if (N > 250)
          { // avoid high frequency noise
            if ( (Signal > thresh) && (Pulse == false) && (N > (IBI/5)*3) )
            {
              Pulse = true; // set the Pulse flag when we think there is a pulse
              digitalWrite(blinkPin,HIGH); // turn on pin 13 LED
              IBI = sampleCounter - lastBeatTime; // measure time between beats in mS
              lastBeatTime = sampleCounter; // keep track of time for next pulse
              if(secondBeat)

            if(firstBeat) // if it's the first time we found a beat, if firstBeat == TRUE
              {
              firstBeat = false; // clear firstBeat flag
              secondBeat = true; // set the second beat flag
              sei(); // enable interrupts again
              return; // IBI value is unreliable so discard it
              }
              // keep a running total of the last 10 IBI values
                word runningTotal = 0; // clear the runningTotal variable
                for(int i=0; i<=8; i++)
                { // shift data in the rate array
                rate[i] = rate[i+1]; // and drop the oldest IBI value
                runningTotal += rate[i]; // add up the 9 oldest IBI values
                }
              rate[9] = IBI; // add the latest IBI to the rate array
              runningTotal += rate[9]; // add the latest IBI to runningTotal
              runningTotal /= 10; // average the last 10 IBI values
              BPM = 23500/runningTotal; // how many beats can fit into a minute? that's BPM!
              QS = true; // set Quantified Self flag
              // QS FLAG IS NOT CLEARED INSIDE THIS ISR
              pulse = BPM;
            }
          }
           if (Signal < thresh && Pulse == true)
              { // when the values are going down, the beat is over
              digitalWrite(blinkPin,LOW); // turn off pin 13 LED
              Pulse = false; // reset the Pulse flag so we can do it again
              amp = P - T; // get amplitude of the pulse wave
              thresh = amp/2 + T; // set thresh at 50% of the amplitude
              P = thresh; // reset these for next time
              T = thresh;
              }
          if (N > 2500)
              { // if 2.5 seconds go by without a beat
              thresh = 560; // set thresh default
              P = 560; // set P default
              T = 560; // set T default
              lastBeatTime = sampleCounter; // bring the lastBeatTime up to date
              firstBeat = true; // set these to avoid noise
              secondBeat = false; // when we get the heartbeat back
              }
              sei(); // enable interrupts when youre done!
  }// end isr

void esp_8266()
{
  //TCP connection AT+CIPSTART=4,"TCP","184.106.153.149",80
  String cmd = "AT+CIPSTART=4,\"TCP\",\"";
  cmd += "184.106.153.149"; // api.thingspeak.com
  cmd += "\",80";
  ser.println(cmd);
  Serial.println(cmd);
  if(ser.find("Error"))
  {
    Serial.println("AT+CIPSTART error");
    return;
  }
  String getStr = "GET /update?api_key=";
  getStr += apiKey;
  getStr +="&field1=";
  getStr +=String(temp);
  getStr +="&field2=";
  getStr +=String(temp2);
  getStr +="&field3=";
  getStr +=String(pulse);
  getStr += "\r\n\r\n";
  // send data length
  cmd = "AT+CIPSEND=4,";
  cmd += String(getStr.length());
  ser.println(cmd);
  Serial.println(cmd);
  delay(1000);
  ser.print(getStr);
  Serial.println(getStr); //thingspeak needs 15 sec delay between updates
  delay(15000);
}

/*Main function, get and show the temperature*/
void read_temp()
{
  // call sensors.requestTemperatures() to issue a global temperature
  // request to all devices on the bus
  Serial.print("Requesting temperatures...");
  sensors.requestTemperatures(); // Send the command to get temperatures
  Serial.println("DONE");
  // After we got the temperatures, we can print them here.
  // We use the function ByIndex, and as an example get the temperature from the first sensor only.
  float tempC = sensors.getTempCByIndex(0);
  temp=tempC;
  // Check if reading was successful
  if (tempC != DEVICE_DISCONNECTED_C)
  {
    Serial.print("Temperature for the device 1 (index 0) is: ");
    Serial.println(tempC);
    lcd.setCursor(0, 0);
    lcd.print("Temperature:");
    lcd.setCursor(0, 1);
    lcd.print(tempC);
    lcd.print((char)223);
```

Fig. 6. Partial programming code showing the pulse detection algorithm and the subsequent transmission of data to the ThingSpeak IoT platform

3. Results

This project is divided into two parts: the robot line follower and two sensors placed on the robot, namely the heartbeat sensor and body temperature sensor. Both microcontrollers use Arduino Uno R3. For the line follower robot, the microcontroller is Arduino Uno r3, while the input is an L298N motor driver and infrared sensor. Accordingly, six infrared sensors are used to ensure the movement of the robot is accurate, and the robot is able to turn 90˚ degrees. At the same time, the motor driver, L298N, can control the speed and the movement of the robot. All responses will be routed to the DC motor to move the robot. In addition, sensors such as heartbeat sensors and body temperature sensors will be incorporated as extra features on the robot. The heartbeat sensor, body temperature, and Wi-Fi module esp8266 (esp-01) are linked to the microcontroller, which is an Arduino Uno r3, for the input section. The output for this part is an LCD that provides information regarding the patient's heartbeat and temperature. The output will also be sent to the device, such as a smartphone or laptop, through the IoT ThingSpeak application. The simulation can be observed in Figure 7 (a) simulation of robot line follower using proteus software and (b) simulation of heartbeat and body temperature sensor using proteus software. In general, this autonomous navigation system eliminates the need for human intervention, making it ideal for contactless operations in healthcare settings.



(a) (b)

Fig. 7. Simulation of robot line follower using proteus software and (b) Simulation of heartbeat and body temperature sensor using proteus software

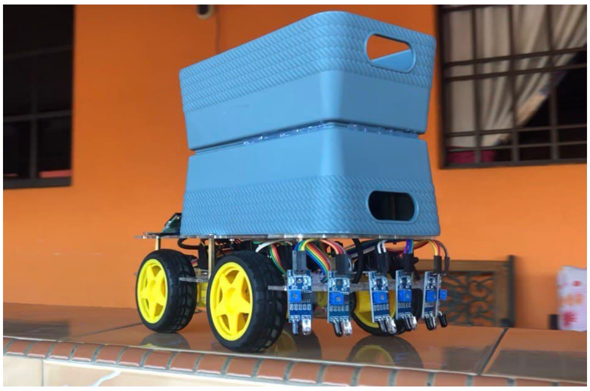
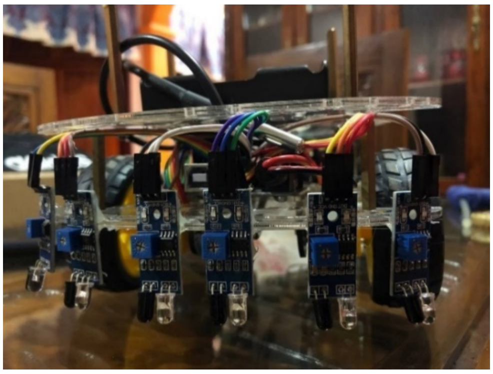


Fig. 8. Prototype for this project

The Figure 8 displays the prototype for this project, where the robots are used in health units such as hospitals, clinics, and quarantine center. The robot is equipped with four tires and one compartment to transport goods, medicine, and other things that patients need. Figure 9 (a) illustrates the configuration of the constructed robot equipped with six infrared sensors. The robot automatically moves forward on the tapeline using the input infrared sensor and output DC motor, where the infrared sensor detects the tape line and sends the information to Arduino. Then, the data in Arduino was sent to the motor driver to move the DC motor on the tapeline. Figure 9 (b) illustrates how the Smart Robot moves on the tapeline that is sensed by an infrared sensor.



(a)

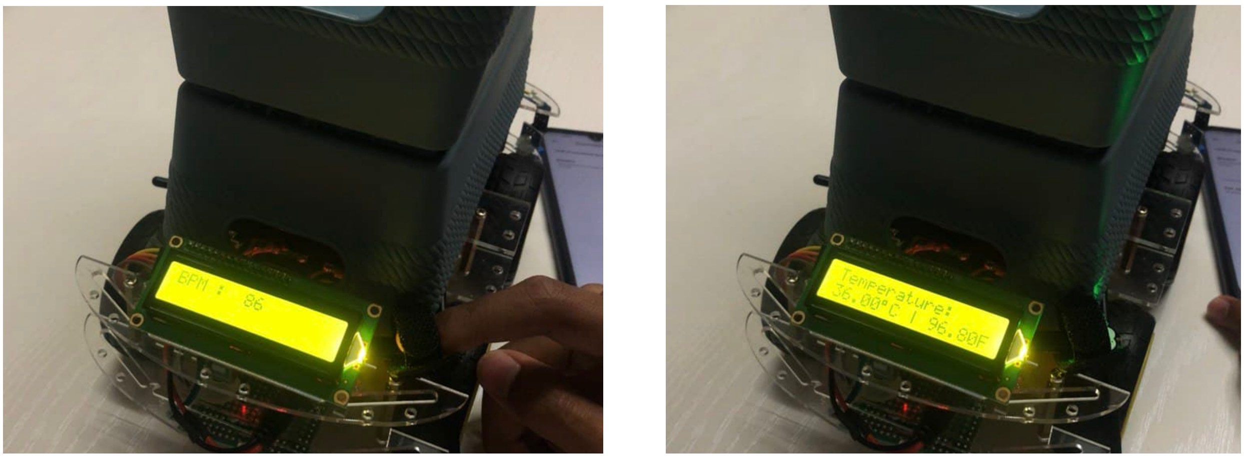
(b)

Fig. 9. (a) configuration of the constructed robot equipped with 6 infrared sensors and (b) Smart Medical Assistant Robot moving on the tapeline

Figure 10 portrays the connection between the pulse rate sensor and the body temperature sensor of the prototype. The pulse rate sensor measures a human's pulse by shining the green light on the finger and measuring the data reflected using the photosensor. Subsequently, the temperature sensor reads a human's temperature based on the human's need to hold or put on the armpit. Correspondingly, both sensors sent the information to Arduino, and data from Arduino was sent to the output of the LCD and IoT device (smartphone) through the Wi-Fi Module ESP-01. Figure 11 (a) displays the LCD for pulse rate in Beat Per Minute (BPM), and Figure 11 (b) illustrates the body temperature in Celsius and Fahrenheit. Next, Figure 12(a) illustrates results displayed on the LCD screen, presenting the patient's pulse rate (BPM), and Figure 12(b) presents the results, displaying the patient's body temperature in Celsius and Fahrenheit, on the ThingSpeak application. The normal resting heart rate for adults ranges from 60 to 100 BPM. Basically, a lower resting heart rate is the most efficient heart function and has better cardiovascular fitness. For instance, a well-trained athlete has a normal resting heart rate of about 40 BPM.

Both sensors are connected to the Arduino Uno R3 microcontroller, which processes and displays the data on an LCD screen. This non-contact monitoring system allows proper and effective check-ups regarding health and reduces the risk of infection. Therefore, the Wi-Fi unit ESP8266 (ESP-01) sends data collected on the Arduino Uno R3 to the ThingSpeak application, enabling health professionals to obtain real-time data on pulse rate and body temperature using smartphones or laptops. Remarkably, this attribute puts the robot at an advantage in monitoring patients without being physically present, and this is very useful, especially in cases of a pandemic.

Fig. 10. The pulse rate sensor and body temperature sensor



(a) (b)

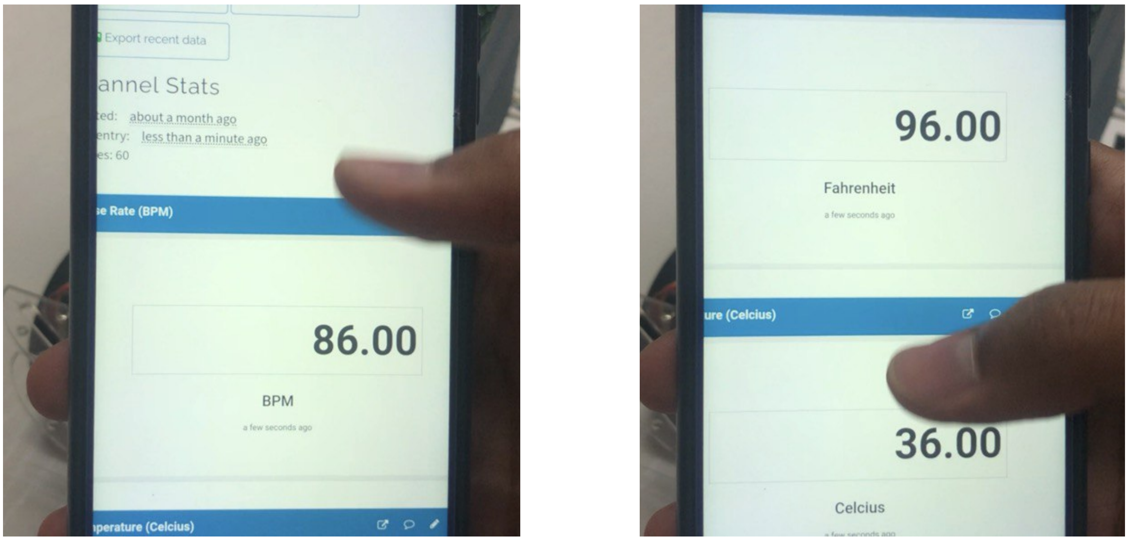
Fig. 11 (a) shows the LCD Display for pulse rate in Beat Per Minute (BPM), and (b) illustrates the body temperature in Celsius and Fahrenheit

This prototype was tested within a period of two weeks in order to determine its functionality, accuracy and endurance. Stable performance over the duration of observation was observed as the robot worked continuously, with the robot taking about five hours to be recharged. The power system is made of seven lithium-ion 18650 batteries (1500 mAh each), as illustrated by Figure 13: four are used to propel the robot and three to energize the temperature and pulse sensors, and they are connected in parallel. The charging needs four hours with a 4.2 V 1 A universal charger. The battery setup was adequate in terms of runtime, but the next generation could use a larger-capacity cell or replaceable battery packs to increase runtime.

To test the robot, it managed to keep its assigned path in 95% of the trials, with slight deviations being noted in the presence of bright ambient lighting. Heart and temperature were also similar to regular clinical devices, and the transmission of IoT data was not problematic; the latency changes slightly with the strength of the Wi-Fi signal. The final results of the performance assessment are given in Table 1.

Table 1
Performance evaluation results of the Smart Medical Assistant Robot prototype.

Parameter	Measured Value	Reference Value	Accuracy / Efficiency
Temperature sensor	±0.5°C	±0.3°C	94%
Pulse rate	±3 BPM	±2 BPM	97%
Path-following success	19/20 trials	-	95%
Average battery life	5 hours	-	-



(a) (b)

Fig. 12. (a) Results displayed on the LCD screen, showing the patient's Pulse Rate (BPM) and (b) Temperature (Celsius and Fahrenheit) on IoT ThingSpeak

.

Fig. 13. 18650 Lithium-ion battery for the robot

Some technical difficulties were faced in this project in the simulation of software, software programming and even hardware development. In the Proteus 8 Professional simulation, there were some problems that occurred because of the unavailability of the components (e.g., Arduino Uno R3, pulse sensor, infrared sensor, and Wi-Fi Module ESP-01) within the software library that were solved by downloading the necessary libraries on their own [15]. Also, the wrong wiring of the Wi-Fi Module ESP-01 was determined and changed through checking the connections and code [16][17]. During the programming phase of the Arduino IDE, the libraries that were not available (e.g., ThingSpeak, I2C LCD, Temperature Sensor DS18B20, and NodeMCU) were overcome by retrieving them online. Besides, the problem of a Serial Port recognition during the programming upload was solved by installing more software to help establish communication between the laptop and the Arduino board [18].

Hardware development was also a problem, including the sensitivity owing to natural light of the infrared sensor, which interfered with the movement of the robot. This was neutralized by adjusting the sensitivity of the sensor and changing defective units [19]. Moreover, the low quality of the 18650 Li-ion battery also impacted the performance of the robot and the motor loading capacity because of its short-lived nature [20][21]. This was overcome by buying a compatible charger to be able to recharge the batteries and increase the battery life. All these solutions were successful in ensuring that the project was completed successfully.

Although the demonstration proved successful, there were a number of limitations that were established. The infrared sensors were vulnerable to ambient lighting, which was very strong, and they could only work perfectly when there was controlled lighting. The Wi-Fi transmission, although workable, at times would have latency spikes of up to 10 seconds within a network-congested environment. The existing 1500mAh battery has a constant running time of roughly 5 hours, which might not be enough to work through the entire shift in a hospital. Additionally, the prototype is not equipped with sophisticated safety systems that this time would be important like an emergency stop button or dynamic obstacle avoidance, which would be essential to use in a dynamic clinical environment.

4. Conclusions

To summarize, the current paper provides an engineering prototype of a Smart Medical Assistant Robot designed with the potential to perform contactless health monitoring and ensure safer healthcare practices in case of a pandemic. The system, incorporating the Arduino Uno R3 microcontroller, infrared, pulse, and DS18B20 temperature sensors, showed its capability to measure the vital signs of patients and move along predefined routes without physical contact with healthcare providers and patients, therefore, minimizing the number of physical contacts between the healthcare provider and patients. The quantitative outcomes of the prototype showed that it had accurate sensing, a steady wireless data transfer and a steady line-following capability, proving that it can be applied in low-cost healthcare.

Although the present paper is concerned with the validation of the prototype, future studies will be conducted based on the systematic performance appraisals performed in the conditions of clinic-like settings, as well as the testing of safety compliance, interaction with the user, and the prolonged durability. Intended advancements like better battery capacity (e.g., 3000mAh) and waterproofing will also be tested experimentally to add functionality. Overall, this innovation contributes toward developing practical, sustainable, and scalable robotic systems capable of supporting contactless medical assistance in post-pandemic healthcare settings.

Acknowledgement
The author would like to thank to Universiti Teknologi MARA (UiTM) Cawangan Johor Kampus Pasir Gudang especially to the assistant engineer and all the staf. This project was not funded by any grant.

Conflict of Interest Statement
The authors declare that there is no conflict of interest regarding the publication of this paper. No financial support, grants, or other forms of compensation were received that could have influenced the outcomes of this work. The research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest

Author Contributions Statement
Haris Murshidi: Designed the study and conducted the experiments.
Nor Diyana: Contributed to study design, performed data analysis, and supervised the overall project prepared the manuscript draft.
Muhammad Amir Syazwan: Conducted the experiments and assisted with data collection.
Mohamad Zhafran: Assisted in manuscript drafting.
Norlina: Assisted in organizing the structure of the manuscript and contributed to editing.
Muhammad Muzamil: Contributed to manuscript refinement
Fazlinashatul Suhaidah: Assisted with proofreading, reference checking, and final manuscript polishing.
Norhalida Othman : Formatting

Data Availability Statement
This published article contains all the data that were generated or analyzed in this study.

Ethics Statement
The research was performed within the institutional/ national research committee ethical standards. Ethical approval was obtained where required. No human participants or animals were involved in this study.

References
[1] D. Pradhan, P. Biswasroy, P. Kumar Naik, G. Ghosh, and G. Rath, "A Review of Current Interventions for COVID-19 Prevention," *Archives of Medical Research*. 2020, doi: 10.1016/j.arcmed.2020.04.020.
[2] D. J. Janson, B. C. Clift, and V. Dhokia, "PPE fit of healthcare workers during the COVID-19 pandemic," *Appl. Ergon.*, 2022, doi: 10.1016/j.apergo.2021.103610.
[3] A. Ahmed, P. Boopathy, and S. Sudhagara Rajan, "Artificial intelligence for the novel corona virus (COVID-19) pandemic: Opportunities, challenges, and future directions," *Int. J. E-Health Med. Commun.*, 2022, doi: 10.4018/IJEHMC.20220701.oa5.
[4] M. Hasan *et al.*, "Personal protective equipment-derived pollution during Covid-19 era: A critical review of ecotoxicology impacts, intervention strategies, and future challenges," *Science of the Total Environment*. 2023, doi: 10.1016/j.scitotenv.2023.164164.
[5] B. Huegl, M. Kreuzer, Z. Doneva, S. Mirazchiyski, and B. Buchter, "Challenges of the COVID-19 pandemic in healthcare and the use of robotic technology - The Robotic Magnetic Navigation, to manage them in the field of electrophysiology | A single-center experience with case studies," *J. Atr. Fibrillation*, vol. 15, no. 6, pp. 65–71, 2022, doi: 10.4022/JAFIB.20200619.
[6] T. Haferlach, "Modern haematology between doctor-patient consultation, whole genome sequencing and artificial intelligence," *Padiatr. Prax.*, vol. 94, no. 3, pp. 485–495, 2020, [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85091709443&partnerID=40&md5=5031ec174614d8eab5d680bd67f9e0fd.
[7] Y. H. El-Gendy and M. K. Hassan, "The Efficiency of Medical Diagnostic Robots in the Light of Legal Liability," 2024,

doi: 10.1109/IC-FTAI62324.2024.10950026.

[8] J. Zhang *et al.*, "Intelligent speech technologies for transcription, disease diagnosis, and medical equipment interactive control in smart hospitals: A review," *Comput. Biol. Med.*, vol. 153, 2023, doi: 10.1016/j.compbiomed.2022.106517.

[9] A. Lee, T. S. Baker, J. B. Bederson, and B. I. Rapoport, "Levels of autonomy in FDA-cleared surgical robots: a systematic review," *npj Digit. Med.*, vol. 7, no. 1, 2024, doi: 10.1038/s41746-024-01102-y.

[10] M. Song, Q. Liu, H. Guo, Z. Wang, and H. Zhang, "Global trends and hotspots in robotic surgery over the past decade: a bibliometric and visualized analysis," *J. Robot. Surg.*, vol. 19, no. 1, 2025, doi: 10.1007/s11701-024-02203-2.

[11] B. Wang, J. Wang, and B. He, "Current research status and prospects for clinical application of robotics in spinal surgery," *Natl. Med. J. China*, vol. 104, no. 37, pp. 3471–3477, 2024, doi: 10.3760/cma.j.cn112137-20240403-00775.

[12] M. Dias, H. Aloj, N. Ninan, D. Koshti, and S. Kamoji, "First Aid and Emergency Assistance Robot for Individuals at Home using IoT and Deep Learning," in *2023 7th International Conference on Computing Methodologies and Communication (ICCMC)*, 2023, pp. 1261–1269, doi: 10.1109/ICCMC56507.2023.10083934.

[13] N. Krishna, R. N. Sree, R. Sajith, S. K. Ramji, and S. Ramesh, "PILLBOT: Non-Contact Medicine Dispensing System for Patients in Quarantine," in *2021 International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT)*, 2021, pp. 97–100, doi: 10.1109/RTEICT52294.2021.9573930.

[14] M. A. Hossen, E. Zahir, H. M. Ata-E-Rabbi, M. A. Azam, and M. H. Rahman, "Developing a Mobile Automated Medical Assistant for Hospitals in Bangladesh," in *2021 IEEE World AI IoT Congress (AIIoT)*, 2021, pp. 366–372, doi: 10.1109/AIIoT52608.2021.9454236.

[15] L. S. Ezema, E. C. Ifediora, A. A. Olukunle, and N. C. Onuekwusi, "Design and Implementation of an Esp32-Based Smart Embedded Industrial Poultry Farm," *Eur. J. Eng. Technol. Res.*, 2021, doi: 10.24018/ejers.2021.6.3.2397.

[16] G. Peralta, R. G. Cid-Fuentes, J. Bilbao, and P. M. Crespo, "Homomorphic encryption and network coding in IoT architectures: Advantages and future challenges," *Electron.*, 2019, doi: 10.3390/electronics8080827.

[17] L. Jocknoi and P. Kucharoen, "ESP32Exten: Designing and Developing an ESP32 Microcontroller Expansion for IoT Applications with Motor Propulsion and AI Image Processing," in *8th International Conference on Information Technology 2024, InCIT 2024*, 2024, pp. 278–283, doi: 10.1109/InCIT63192.2024.10810578.

[18] S. Yue, "Design of an Accessible Music Control Device for Individuals with Disabilities Based on Gesture Recognition," in *2024 5th International Conference on Computer Engineering and Application (ICCEA)*, 2024, pp. 1523–1526.

[19] M. Lacity, L. Willcocks, and D. Gozman, "Influencing information systems practice: The action principles approach applied to robotic process and cognitive automation," *J. Inf. Technol.*, 2021, doi: 10.1177/0268396221990778.

[20] L. S. Jayashree and G. Selvakumar, "Introduction to Enterprise IoT," in *Getting Started with Enterprise Internet of Things: Design Approaches and Software Architecture Models*, 2020.

[21] K. Maher and A. Boumaiza, "Performance and Safety of Lithium-Ion Batteries Under Hot Desert Climates," in *International Conference on Renewable Energy and Power Engineering, REPE*, 2024, no. 2024, pp. 185–189, doi: 10.1109/REPE62578.2024.10810068.