An IoT-Enabled Smart Medical Bed for Monitoring Patient's Health

Shahad Al Yahyaei [1], Shaik Hussain [2]

sh.k.alyahyaee@gmail.com[1], notemaz@gmail.com[2]

Department of Computing and Electronics Engineering, Middle East College, Muscat, Oman [1, 2]

Abstract. The growing demand for healthcare personnel, combined with the increasing elderly population, highlights the need for more efficient and automated patient-monitoring solutions. Conventional monitoring requires frequent manual checks, making the process intrusive and labour-intensive for medical staff. This paper presents an IoT-enabled smart medical bed designed to monitor key physiological indicators using integrated sensing modules. The system focuses on elderly patients, who represent a large proportion of hospital admissions and require close monitoring of vital parameters. The proposed design captures heart rate, oxygen saturation, body temperature, and pressure through embedded sensors, transmitting the collected data to the Blynk platform and displaying it locally on an LCD panel. By enabling continuous, real-time observation of multiple patients, the system reduces the need for constant room visits by nursing teams and allows timely intervention when abnormal readings are detected.

Keywords: LCD panel, blynk software, IoT sensors, medical bed.

1 Introduction

The increasing demand for healthcare professionals, together with the rapid growth of the elderly population, has strengthened the need for automation within the medical sector. Conventional patient monitoring remains highly intrusive and requires continuous manual effort from medical staff [1]. This paper presents an IoT-enabled smart medical bed capable of tracking several vital parameters respiration, heart rate, oxygen saturation, pressure, and temperature using optical and environmental sensors with integrated spatial distribution [2]. These physiological indicators are central to evaluating patient well-being and to identifying potential complications in individuals residing in nursing homes or long-term care facilities.

To maintain consistent monitoring, sensors are typically positioned on various parts of the body to capture vital signs. Such methods rely on secure skin-to-sensor contact, often using Velcro straps or adhesive-based mechanisms [3]. However, these attachments may cause discomfort or become unreliable if displaced. Blood pressure readings provide physicians with critical insights into cardiovascular conditions, enabling early prediction of risks such as heart attacks and strokes. For older adults, a pulse pressure exceeding 60 is associated with heightened cardiac

MECSRC 2025, May 01, Muscat, Oman

DOI 10.4108/eai.1-5-2025.2361529

risk. Normal SpO_2 readings among elderly individuals generally fall between 95% and 100% [4], whereas reduced oxygen saturation can indicate respiratory challenges or insufficient blood oxygenation. Resting heart rates for this age group typically range from 80 to 136 beats per minute; values outside this range may suggest underlying medical concerns. Delay in responding to abnormal readings may result in serious complications, including heart failure or sudden cardiac arrest [5][6].

In this project, the GY-BMP280 sensor is used to monitor blood pressure, the MAX30100 module measures both heart rate and oxygen saturation, and the MLX90614 sensor captures body temperature to support timely detection of health deterioration [7][8]. The collected measurements are presented on an LCD screen for immediate observation and are simultaneously transmitted to the doctor's monitoring station through the Blynk platform, allowing real-time oversight of multiple elderly patients [9][10].

2 Literature Review

Recent advancements in temperature monitoring technologies have led to the replacement of conventional systems with intelligent sensing solutions capable of real-time measurement. Such approaches are frequently used in CNC machining and other industrial processes, where accurate and continuous temperature tracking is essential. In the referenced study, the initial step involved validating the temperature data recorded by the sensors, ensuring reliability before moving to the machining tests. The researchers began by comparing readings from the MLX90614 sensor with those obtained from a Fluke Ti400 thermal imaging device. Subsequent sections of the study discussed the recorded milling temperatures and highlighted broader concerns regarding measurement accuracy, data consistency, and the dependability of wearable monitoring devices. Usability and user experience also play a critical role, as long-term adoption of wearable health systems depends on comfort, intuitive interfaces, and reliable data output.

The integration of IoT technologies with e-health platforms has improved the efficiency and quality of telemedicine applications by enabling cost-effective, continuous monitoring for patients and caregivers. The Arduino Uno, based on the ATmega328 microcontroller, is widely utilized in IoT health projects due to its simplicity, flexibility, and ease of programming. It features 14 digital I/O pins six of which support PWM alongside six analogue inputs, a 16 MHz quartz oscillator, USB and ICSP ports, and a voltage regulator. The term "Uno," meaning "one" in Italian, reflects its significance as a foundational and highly accessible board within the Arduino family. Compared to similar controllers, the Uno provides an optimal balance of size, functionality, and affordability, making it suitable for rapid prototyping in health-monitoring applications.

In the project, the system development is based on a methodical approach of engineering the same to facilitate reliability and systematic implementation [11]. Systematic approaches are required in IoT-based healthcare systems to control the integration of hardware, software, and validation [12], [13]. There is a clear development structure that allows the design errors to be minimized and the testing activities to be synchronized with the requirements of the system. Thus, the sequential development model was followed to implement and validate the proposed smart medical bed.

Within hospital environments, nurses and medical staff manage demanding workloads while conducting frequent vital-sign checks on admitted patients. High patient volumes often limit the time available for personalised monitoring, requiring nurses to visit rooms individually for status updates. This challenge was underscored by the Director of State Health Services for Kuching, Datuk Dr. Zulkifli Jantan, who highlighted concerns regarding the nurse-to-patient ratio and the resulting strain on staff.

3 System Analysis and Design

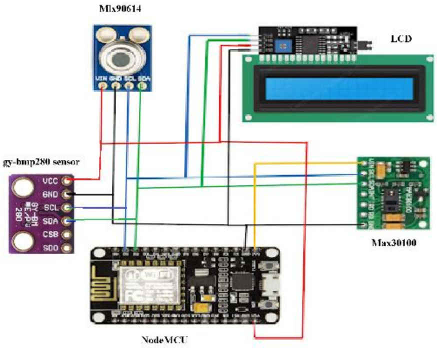


Fig. 1. System Design.

The system relies on connecting multiple sensors to the NodeMCU microcontroller to capture patient vitals. For the MAX30100 module, which measures heart rate and oxygen saturation, the VCC pin is linked to the 3.3V output of the NodeMCU. The SCL and SDA pins are connected to D1 and D2 respectively, while the GND terminal is tied to the board's ground. The MLX90614 temperature sensor follows the same wiring pattern: VCC to 3.3V, SCL to D1, SDA to D2, and GND to the common ground.

Similarly, the BMP280 pressure sensor uses identical connections. Its VCC pin interfaces with the 3.3V supply, the SCL and SDA pins are assigned to D1 and D2, and its GND pin is fixed to the NodeMCU ground. The LCD panel is also integrated using the same communication lines VCC to 3.3V, SCL to D1, SDA to D2, and GND to the shared ground terminal allowing patient data to be displayed in real time.

To ensure structured development of the system, the V-model methodology was selected. This approach supports early detection of system issues and enhances project management efficiency by pairing each development phase with a corresponding testing phase.

Usually known as the verification and validation model, the V-model is depicted in Figure 2. It is an organized Software development life cycle (SDLC) methodology where every stage of development is directly linked to a stage of testing. This makes mistakes easily noticed and enhances the reliability of the system since the requirements are checked at each phase of the development. Each stage begins only after the previous one is completed, ensuring a systematic flow throughout the project.

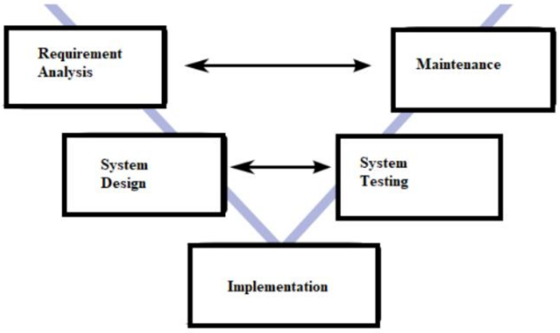


Fig. 2. V-Methodology

The phases applied in this work are summarised below:

Requirement Analysis:

The project requires several key components: a medical bed, an Arduino board, sensors including MLX90614, MAX30100, and GY-BMP280, the Blynk application, wiring accessories, a buzzer, and a breadboard.

System Design:

Once the requirements are understood, system design activities follow. Different diagrams, such as flowcharts and block diagrams, are used to better visualise and clarify how the system functions. This phase also involves analysing how the proposed implementation will meet the stated requirements.

Implementation / Coding:

This stage involves writing and deploying the program using the Arduino IDE. The Blynk application is used to receive and visualise sensor outputs. The software code manages the integration between hardware components and the IoT platform.

System Testing:

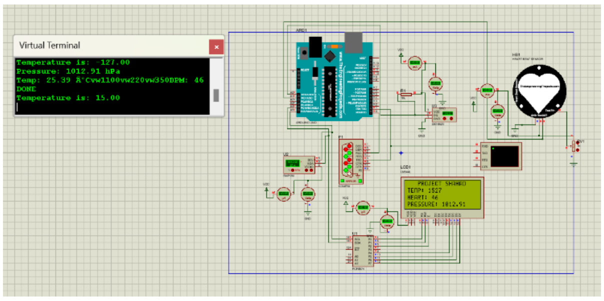
Test plans created earlier in the design phase are executed here. System testing assesses whether the application operates as expected, examining connectivity, functional performance, and interactions among system components. All sensors used in the project undergo verification to confirm their reliability.

Maintenance:

After deployment, system components must be monitored and maintained. Any malfunctioning parts are repaired or replaced to ensure continuous operation of the smart medical bed.

4 Results and Discussion

This section presents the outcomes obtained from both the Proteus simulation environment and the hardware validation carried out using the Arduino IDE and ESP8266 microcontroller. Figure 3 displays the circuit layout developed in Proteus, which was used to verify system functionality prior to physical implementation. The researchers relied on a combination of simulation tools and breadboard-based testing to confirm accurate sensor readings.



```
Temperature is: -127.00
Pressure: 1012.91 hPa
Temp: 25.39 Â°Cvw1100vw220vw350DPM: 46
DONE
Temperature is: 15.00
```

```
PROJECT SHAHAD
TEMP: 1527
HEART: 46
PRESSURE: 1012.91
```

Fig. 3. Proteus Design

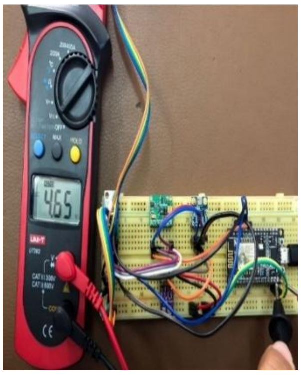


Fig. 4. Measured Voltage Value

To verify voltage levels, the positive test lead was connected to the 5V pin of the ESP8266, while the negative lead was attached to ground. Under this configuration, the measured supply voltage was approximately 4.65 V, as shown in Figure 4. A similar procedure was used for the MLX90614 temperature sensor, where the measured voltage registered at 4.52 V shown in Figure 5.

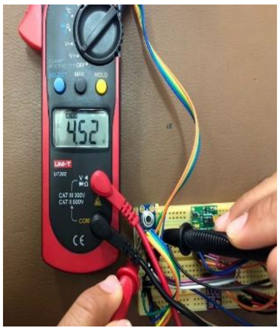


Fig. 5. Measured Voltage Value

Because the Proteus software does not include a native MLX90614 component, the DS18B20 was used as a substitute for temperature-related simulations. This configuration produced a stable reading of 5 V as depicted in Figure 6. The same approach was applied to the BMP280 pressure sensor: connecting the sensor's positive terminal to its 5V input and grounding the negative terminal yielded a measured value of 4.61 V shown in Figure 7. Again, due to limited component availability in Proteus, the BMP180 sensor was used in place of the BMP280 during simulation, resulting in a detected supply voltage of 5 V shown in Figure 8.

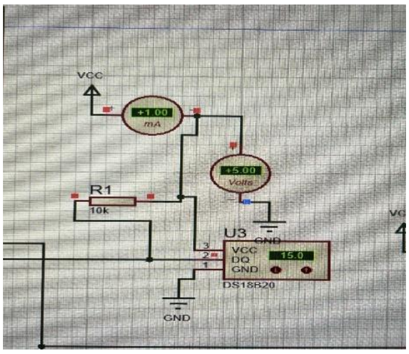


Fig. 6. Measured Voltage Value

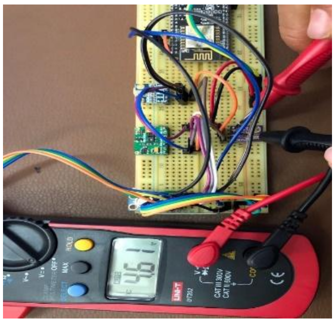


Fig. 7. BMP280 sensor voltage measurement

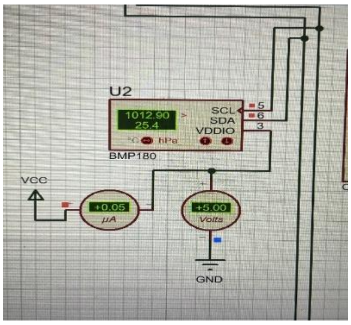


Fig. 8. BMP180 sensor

For the MAX30100 sensor, used to measure heart rate and oxygen saturation, voltage verification showed a reading of 3.07 V as in Figure 9. As with previous components, the absence of MAX30100 support in Proteus required the use of the HB1 sensor for simulation purposes, which also produced a reading of 5 V shown in Figure 10.

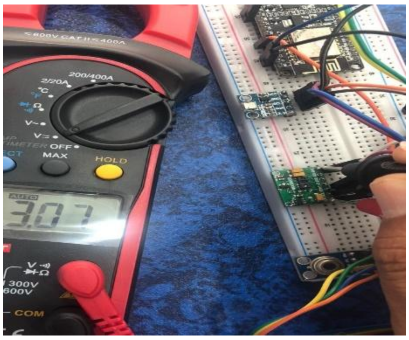


Fig. 9. MAX30100

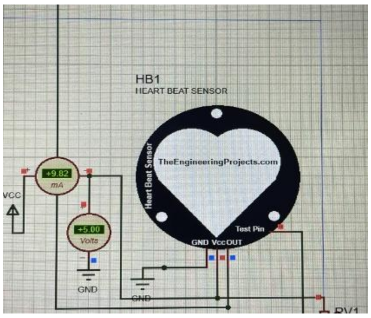


Fig. 10. HB1 sensor

The LCD module was assessed in a similar manner by connecting the positive probe to the 5V terminal and grounding the negative probe. This test recorded a voltage of 3.99 V illustrated in Figure 11. Proteus simulations using the I2C 20×4 LCD panel yielded a system-detected voltage of 5 V in Figure 12.

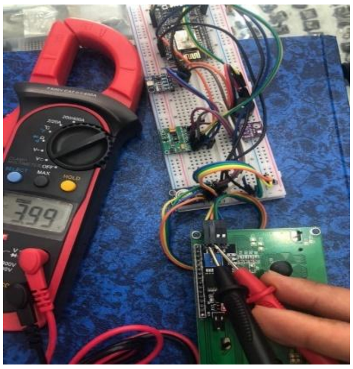
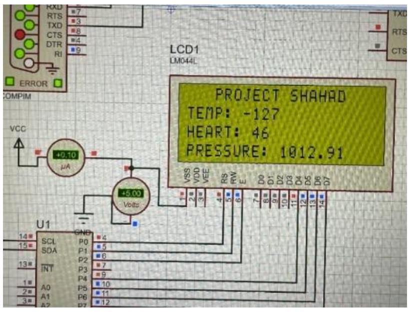


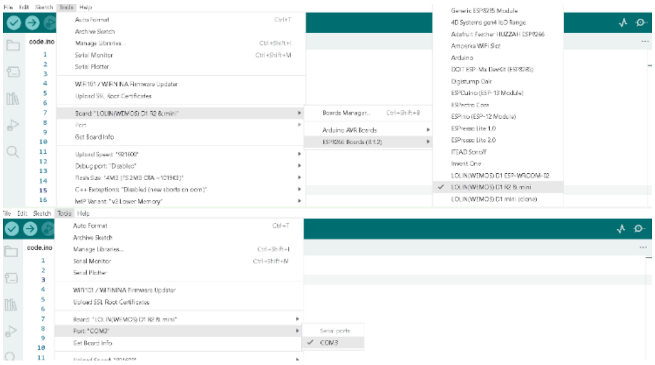
Fig. 11. LCD I2C

Prior to uploading the program, the appropriate libraries for the ESP8266, MAX30100, MLX90614, and BMP280 sensors were installed within the Arduino IDE. The board type "LOLIN (WEMOS) D1 R2 & mini" and communication port "COM3" were selected to ensure proper compilation and uploading of the firmware. Figure 13 summarises the required steps and library configurations.



```
PROJECT SHAHAD
TEMP: -127
HEART: 46
PRESSURE: 1012.91
```

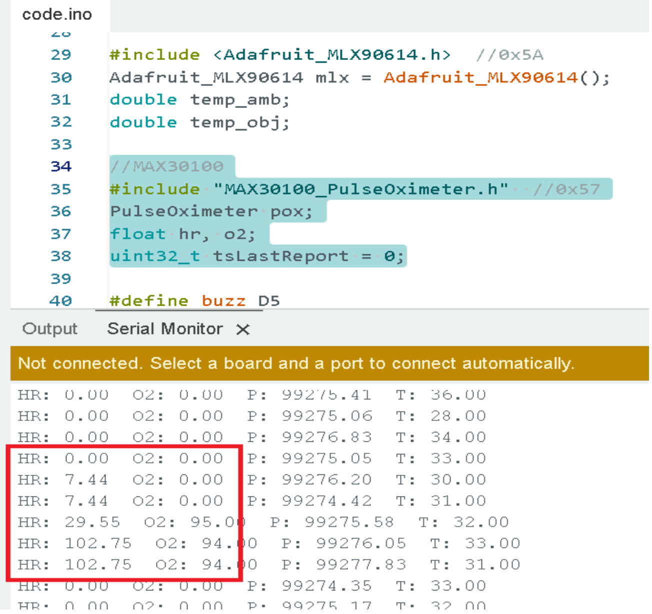
Fig. 12. I2C 20*4 LCD panel



Board: "LOLIN(WEMOS) D1 R2 & mini"
Upload Speed: "921600"
Debug port: "Disabled"
Flash Size: "4MB (FS:2MB OTA:~1019KB)"
C++ Exceptions: "Disabled (new aborts on oom)"
lwIP Variant: "v2 Lower Memory"
ESP8266 Boards (3.1.2)
✓ LOLIN(WEMOS) D1 R2 & mini
Port: "COM3"
✓ COM3

Fig. 13. Libraries

Once deployed, the sensor outputs were observed in both the Arduino Serial Monitor and the LCD panel. The MAX30100 data, including heart rate and oxygen-level readings, appeared correctly in the Serial Monitor in Figure 14 and on the LCD screen shown in Figure 15. The MLX90614 sensor also produced consistent temperature readings, visible through the Serial Monitor as in Figure 16 and subsequently displayed on the LCD depicted in Figure 17. Equivalent results were obtained for the BMP280 pressure sensor, with readings shown in Figures 18 and 19.



```
#include <Adafruit_MLX90614.h>  //0x5A
Adafruit_MLX90614 mlx = Adafruit_MLX90614();
double temp_amb;
double temp_obj;

//MAX30100
#include "MAX30100_PulseOximeter.h"  //0x57
PulseOximeter pox;
float hr, o2;
uint32_t tsLastReport = 0;

#define buzz D5
```

Not connected. Select a board and a port to connect automatically.

```
HR: 0.00  O2: 0.00  P: 99275.41  T: 36.00
HR: 0.00  O2: 0.00  P: 99275.06  T: 28.00
HR: 0.00  O2: 0.00  P: 99276.83  T: 34.00
HR: 0.00  O2: 0.00  P: 99275.05  T: 33.00
HR: 7.44  O2: 0.00  P: 99276.20  T: 30.00
HR: 7.44  O2: 0.00  P: 99274.42  T: 31.00
HR: 29.55  O2: 95.00  P: 99275.58  T: 32.00
HR: 102.75  O2: 94.00  P: 99276.05  T: 33.00
HR: 102.75  O2: 94.00  P: 99277.83  T: 31.00
HR: 0.00  O2: 0.00  P: 99274.35  T: 33.00
HR: 0.00  O2: 0.00  P: 99275.17  T: 32.00
```

Fig. 14. MAX30100

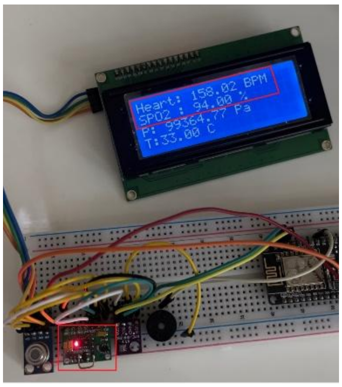
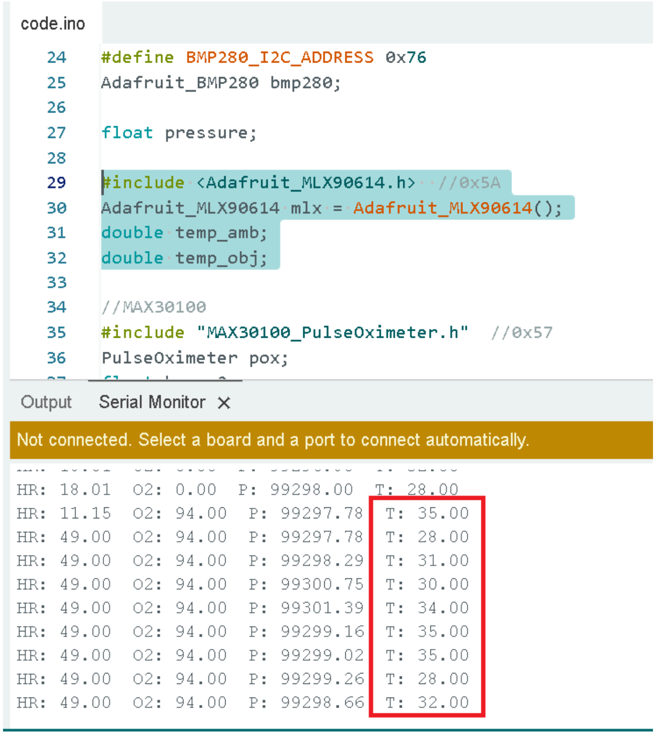


Fig. 15. MAX30100 sensor



```
#define BMP280_I2C_ADDRESS 0x76
Adafruit_BMP280 bmp280;

float pressure;

#include <Adafruit_MLX90614.h>  //0x5A
Adafruit_MLX90614 mlx = Adafruit_MLX90614();
double temp_amb;
double temp_obj;

//MAX30100
#include "MAX30100_PulseOximeter.h"  //0x57
PulseOximeter pox;
```

Not connected. Select a board and a port to connect automatically.

```
HR: 18.01  O2: 0.00  P: 99298.00  T: 28.00
HR: 11.15  O2: 94.00  P: 99297.78  T: 35.00
HR: 49.00  O2: 94.00  P: 99297.78  T: 28.00
HR: 49.00  O2: 94.00  P: 99298.29  T: 31.00
HR: 49.00  O2: 94.00  P: 99300.75  T: 30.00
HR: 49.00  O2: 94.00  P: 99301.39  T: 34.00
HR: 49.00  O2: 94.00  P: 99299.16  T: 35.00
HR: 49.00  O2: 94.00  P: 99299.02  T: 35.00
HR: 49.00  O2: 94.00  P: 99299.26  T: 28.00
HR: 49.00  O2: 94.00  P: 99298.66  T: 32.00
```

Fig. 16. MLX90614

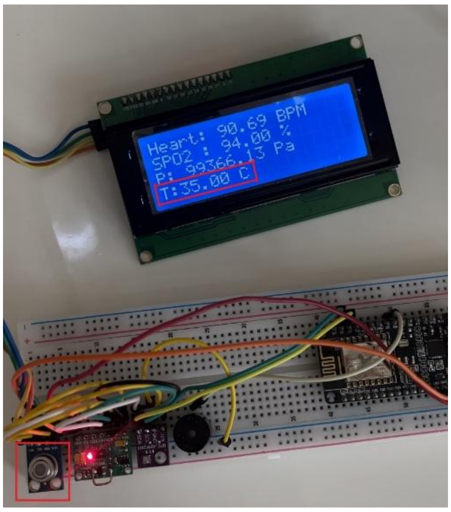
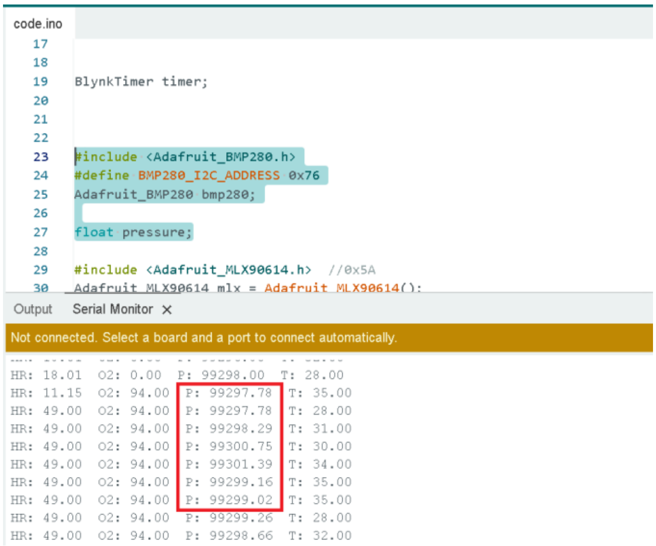


Fig. 17. MLX90614 sensor



```


BlynkTimer timer;



#include <Adafruit_BMP280.h>
#define BMP280_I2C_ADDRESS 0x76
Adafruit_BMP280 bmp280;

float pressure;

#include <Adafruit_MLX90614.h>  //0x5A
Adafruit_MLX90614 mlx = Adafruit_MLX90614();
```

Not connected. Select a board and a port to connect automatically.

```
HR: 18.01  O2: 0.00  P: 99298.00  T: 28.00
HR: 11.15  O2: 94.00  P: 99297.78  T: 35.00
HR: 49.00  O2: 94.00  P: 99297.78  T: 28.00
HR: 49.00  O2: 94.00  P: 99298.29  T: 31.00
HR: 49.00  O2: 94.00  P: 99300.75  T: 30.00
HR: 49.00  O2: 94.00  P: 99301.39  T: 34.00
HR: 49.00  O2: 94.00  P: 99299.16  T: 35.00
HR: 49.00  O2: 94.00  P: 99299.02  T: 35.00
HR: 49.00  O2: 94.00  P: 99299.26  T: 28.00
HR: 49.00  O2: 94.00  P: 99298.66  T: 32.00
```

Fig. 18. BMP280

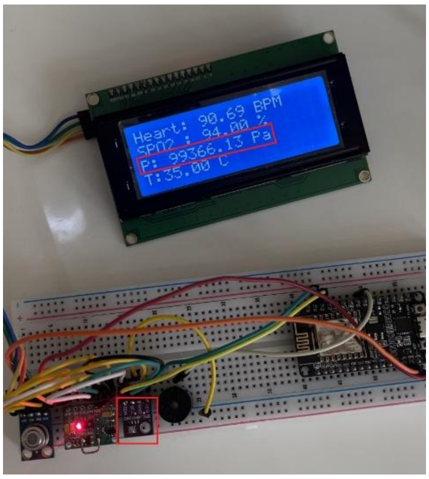


Fig. 19. BMP280 sensor

After establishing Wi-Fi connectivity, all sensor data including temperature, pressure, SpO₂, and heart-rate values were transmitted to the Blynk application for real-time monitoring. Figure 20 illustrates the values displayed in the app, where the system reported a temperature of 33 °C, a pressure of 99.3, an oxygen saturation of 94%, and a heart rate of approximately 105.87 bpm. These results confirm that the integrated sensors and IoT communication workflow function reliably, enabling remote observation of patients’ vital signs through the Blynk dashboard.

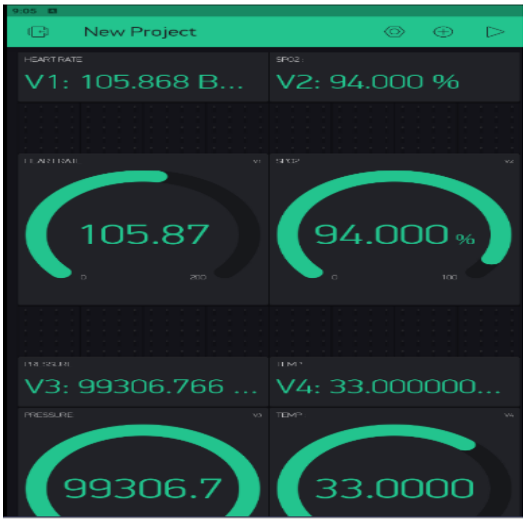


Fig. 20. Blynk application results

5 Conclusion

The persistent shortage of healthcare professionals, coupled with the rise in the elderly population, highlights the importance of integrating automation into patient-monitoring processes. Traditional monitoring approaches require frequent manual checks, making them both intrusive for patients and demanding for medical staff. This work presented an IoT-based smart medical bed designed to continuously monitor essential physiological parameters including respiration, heart rate, oxygen saturation, blood pressure, and temperature using a combination of embedded sensors.

These vital indicators are essential for early detection of health deterioration, especially for individuals in long-term care facilities. The system displays real-time measurements on an LCD panel for immediate observation and simultaneously sends all sensor data to the supervising physician through the Blynk platform. This capability enables healthcare providers to monitor multiple elderly patients at once, facilitates timely intervention, and reduces the need for repeated physical room visits by nurses.

6 Acknowledgment

The authors would like to deeply acknowledge Middle East College, Muscat, Oman for providing all types of resources and facilities in completing this bachelor project.

References

[1] Beginners guide to Arduino IDE and Arduino programming. (n.d.). Circuitdigest.com. Retrieved December 29, 2022, from https://circuitdigest.com/article/getting-started-with-arduino-programming-using-arduino-ide

[2] BibGuru. (n.d.). Bibguru.com. Retrieved December 29, 2022, from https://app.bibguru.com/p/b56e0233-bfb9-4814-9378-77d214a05a4c

[3] Bunker, J. (2015, July 6). Control an arduino with your smartphone via blynk. Make: DIY Projects and Ideas for Makers; Make: Community. https://makezine.com/article/technology/arduino/control-arduino-your-smartphone-via-blynk/

[4] Gong, L. (2020, September 30). Introduction to MLX90614. Latest Open Tech From Seeed. https://www.seeedstudio.com/blog/2020/09/30/introduction-to-mlx90614/

[5] Goyal, A. (2021, December 10). Difference between Max30100 and Max30102 sensor? Hatchnhack Cart. https://www.hnhcart.com/blogs/sensors-modules/a

[6] Hemmings, M. (n.d.). What is a jumper wire? Sparkfuneducation.com. Retrieved December 29, 2022, from https://blog.sparkfuneducation.com/what-is-jumper-wire

[7] Huang, C.-H., & Guo, J.-W. (2021). Design of reflectance pulse oximeter and BPM using the Max30100 sensor in early detection of hypoxemia in patients with cardiovascular disorders. International Journal of Advanced Health Science and Technology, 1(1), 1–6. https://doi.org/10.35882/ijahst.v1i1.1

[8] Lab, M. (2021, February 21). Interface MLX90614 Non-Contact IR temperature sensor with Arduino. Microcontrollers Lab. https://microcontrollerslab.com/mlx90614-contactless-ir-temperature-sensor-pinout-interfacing-arduino/

[9] MLX90614 non-contact IR temperature sensor. (n.d.). Components101. Retrieved December 29, 2022, from https://components101.com/sensors/melexis-mlx90614-contact-less-ir-temperature-sensor

[10] Mohamad Hadis, N. S., Amirnazarullah, M. N., Jafri, M. M., & Abdullah, S. (2020). IoT based patient monitoring system using sensors to detect, Analyse and monitor two primary vital signs. Journal of Physics. Conference Series, 1535(1), 012004. https://doi.org/10.1088/1742-6596/1535/1/012004

[11] S. Smith, L. Lai, and R. Khedri, “Requirements analysis for engineering computation: A systematic approach for improving reliability,” *Reliable Comput.*, vol. 13, pp. 83–107, 2007, doi: 10.1007/s11155-006-9020-7.

[12] E. Mezghani, E. Exposito, and K. Drira, “A model-driven methodology for the design of autonomic and cognitive IoT-based systems: Application to healthcare,” *IEEE Trans. Emerging Topics Comput. Intell.*, vol. 1, no. 3, pp. 224–234, Jun. 2017, doi: 10.1109/TETCI.2017.2699218.

[13] D. Dziak, B. Jachimczyk, and W. J. Kulesza, “IoT-based information system for healthcare application: Design methodology approach,” *Appl. Sci.*, vol. 7, no. 6, Art. no. 596, Jun. 2017, doi: 10.3390/app7060596.