UBIDOT BASED ROBUST PRIMARY CARE MONITORING

A PROJECT REPORT

Submitted by

K. BRUNDHA

A. GOKULAKANNAN

A.K. VIGNESH

in partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

UNIVERSITY COLLEGE OF ENGINEERING, RAMANATHAPURAM

ANNA UNIVERSITY::CHENNAI 600 025

APRIL 2024

UBIDOT BASED ROBUST PRIMARY CARE MONITORING

A PROJECT REPORT

Submitted by

K. BRUNDHA	**(913020106001)**
A. GOKULAKANNAN	**(913020106002)**
A.K. VIGNESH	**(913020106004)**

in partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

UNIVERSITY COLLEGE OF ENGINEERING, RAMANATHAPURAM

ANNA UNIVERSITY::CHENNAI 600 025

APRIL 2024

ANNA UNIVERSITY : CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report "**UBIDOT BASED ROBUST PRIMARY CARE MONITORING"** is the bonafide work of "**K. BRUNDHA, A. GOKULAKANNAN, A. K. VIGNESH,"** who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr.M.SHANTHI M.E., PhD.,	**Dr.R.SORNA KEERTHI M.E., PhD.,**
Assistant Professor	Assistant Professor (Sr.Gr.)
HEAD OF THE DEPARTMENT	**SUPERVISOR**
Electronics & Communication Engg.,	Electronics & Communication Engg.,
University College of Engineering,	University College of Engineering
Ramanathapuram-623 513.	Ramanathapuram-623513.

Submitted for the Viva-voce examination held at University College of Engineering, Ramanathapuram on....................

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

At the very beginning we would like to express primarily our deep gratefulness to almighty to shower us strength and composure to complete the execution of this project with success.

We express our sincere gratitude to, **Dr. T. UDAYAKUMAR Ph.D.,** Dean, University College of Engineering, Ramanathapuram who kindly and heartily permitted us to go on with this project.

We articulate our thankfulness to **Dr. M. SHANTHI M.E.**, **Ph.D.,** Head of the Department and our Project coordinator , Department of Electronics and Communication Engineering, University College of Engineering, Ramanathapuram for her spending her valuable time in guiding us in the Perfect and right direction and for her constant inspiration and encouragement.

We acknowledge with recognition **Dr. R. SORNA KEERTHI M.E., Ph.D.**, Project guide , ECE, UCER, for the kind of patronage and attitude showed on us in enrouting the completion of the project work firmly by giving commendable suggestions.

Finally, we would like to express our thankfulness to our **parents and friends** who were involved directly and indirectly in successful completion of this project.

ABSTRACT

The project titled "Smart Health Care Monitoring using Ubidots" integrates various sensors and devices to create an advanced healthcare monitoring system. The system utilizes Arduino Uno as the main controller, incorporating sensors such as the Max30100 for pulse oximetry, an ECG sensor for real-time electrocardiogram data displayed on the serial plotter. The collected data is transmitted to the Ubidots Cloud platform using Node MCU, ensuring seamless remote monitoring. Additionally, the system provides immediate feedback through an LCD display and a buzzer for alert notifications, enhancing the overall effectiveness of patient care and healthcare management.

This comprehensive approach leverages the power of Internet of Things (IoT) technology to revolutionize healthcare monitoring and improve the quality of patient outcomes. The system incorporates Arduino Uno as the primary microcontroller, along with a suite of sensors including Max30100 for pulse oximetry and heart rate monitoring, an ECG sensor for electrocardiogram data acquisition. The Node MCU acts as the communication bridge, facilitating the transmission of sensor data to the Ubidots cloud platform. The proposed smart healthcare monitoring system offers a reliable, cost-effective solution for continuous patient monitoring, with potential applications in hospitals, clinics, and home healthcare settings. Its modular design allows for customization and expansion to accommodate diverse healthcare needs, contributing to improved patient outcomes and enhanced healthcare delivery.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATION

ABBREVIATION	EXPLANATION
IOT	Internet of Things
M2M	Machine to machine
ROM	Read only Memory
RTOS	Real time Operating System
ADC	Analog to Digital Convertor
DAC	Digital to Analog Convertor
TCP	Transfer Control Protocol
LAN	Local Area Network
WAN	Wide Area Network
IDE	Integrated Development Environment
URL	Uniform Resource Locator
LCD	Licuid Crystal Display
SPO2	Saturation of Peripherial Oxygen
MCU	Micro Controller Unit
ECG	Electro Cardio Gram

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

In recent years, the convergence of Internet of Things (IoT) technology, cloud computing, and healthcare has paved the way for innovative approaches to patient monitoring and healthcare delivery. Smart healthcare monitoring systems leverage these technologies to enable real-time monitoring of patients' vital signs, activities, and health status, facilitating proactive interventions and personalized care strategies. With the increasing prevalence of chronic diseases, aging populations, and the demand for remote healthcare services, there is a growing need for advanced monitoring solutions that can enhance patient outcomes, improve caregiver efficiency, and optimize resource utilization in healthcare facilities. Traditional healthcare monitoring methods often rely on periodic measurements and manual documentation, leading to delays in detecting critical health events and limiting the ability of healthcare providers to deliver timely interventions. Moreover, the fragmentation of monitoring solutions and data silos within healthcare systems hinder interoperability and data exchange, impeding collaborative decision-making and continuity of care.

In response to these challenges, smart healthcare monitoring systems offer a comprehensive and integrated approach to patient monitoring, leveraging IoT sensors, cloud-based platforms, and advanced analytics to provide continuous, real-time monitoring and actionable insights. These systems enable remote monitoring of patients' health status from any location with internet access, supporting telemedicine initiatives and improving accessibility to healthcare services for patients in remote or underserved areas. This paper presents a novel Smart Healthcare Monitoring System with Ubidots Integration, which combines Arduino-based sensor modules with the Ubidots cloud platform to enable real-time monitoring, data visualization, and remote accessibility of patient health data. The system integrates various sensors for measuring vital signs such as heart rate, temperature, and activity levels, along with cloud-based analytics and alerting mechanisms to facilitate timely interventions and proactive healthcare management.

1.2 INTERNET OF THINGS:

The Internet of Things (IoT) refers to the use of intelligently connected devices and systems to leverage data gathered by embedded sensors and actuators in machines and other physical objects. IoT is expected to spread rapidly over the coming years and this convergence will unleash a new dimension of services that improve the quality of life of consumers and productivity of enterprises, unlocking an opportunity that the GSMA refers to as the ‘Connected Life’.

For consumers, the IoT has the potential to deliver solutions that dramatically improve energy efficiency, security, health, education and many other aspects of daily life. For enterprises, IoT can underpin solutions that improve decision-making and productivity in manufacturing, retail, agriculture and other sectors.

Machine to Machine (M2M) solutions - a subset of the IoT – already use wireless networks to connect devices to each other and the Internet, with minimal direct human intervention, to deliver services that meet the needs of a wide range of industries. In 2013, M2M connections accounted for 2.8% of global mobile connections (195 million), indicating that the sector is still at a relatively early stage in its development. An evolution of M2M, the IoT represents the coordination of multiple vendors’ machines, devices and appliances connected to the Internet through multiple networks.

While the potential impact of the IoT is considerable, a concerted effort is required to move beyond this early stage. In order to optimise the development of the market, a common understanding of the distinct nature of the opportunity is required. To date, mobile operators have identified the following key distinctive features:

1. The Internet of Things can enable the next wave of life-enhancing services across several fundamental sectors of the economy.

2. Meeting the needs of customers may require global distribution models and consistent global services.

3. The Internet of Things presents an opportunity for new commercial models to support mass global deployments.

4. The majority of revenue will arise from the provision of value-added services and mobile operators are building new capabilities to enable these new service areas.

5. Device and application behaviour will place new and varying demands on mobile networks.

1.2.1 APPLICATIONS:

While the Internet of Things (IoT) will ultimately have an enormous impact on consumers, enterprises and society as a whole, it is still at an early stage in its development. As mobile operators and their partners pilot new services across multiple sectors, ranging from health to automotive, they have identified several distinctive features of the Internet of Things. A common understanding of the distinctive nature of this nascent opportunity should help hasten the development of this market. The five distinctive features are:

Industrial Sector: As the Internet of Things evolves, the proliferation of smart connected devices supported by mobile networks, providing pervasive and seamless connectivity, will unlock opportunities to provide life-enhancing services for consumers while boosting productivity for enterprises. As can be seen in Figure below, thirteen industry sectors are likely to show significant adoption of IoT services. For consumers, connectivity provided by the IoT could enhance their quality of life in multiple ways, such as, but not limited to, energy efficiency and security at home and in the city. In the home, the integration of connected smart devices and cloud-based services will help address the pressing issue of energy efficiency and security. Connected smart devices will enable a reduction in utility bills and outages, while also improving home security via remote monitoring.

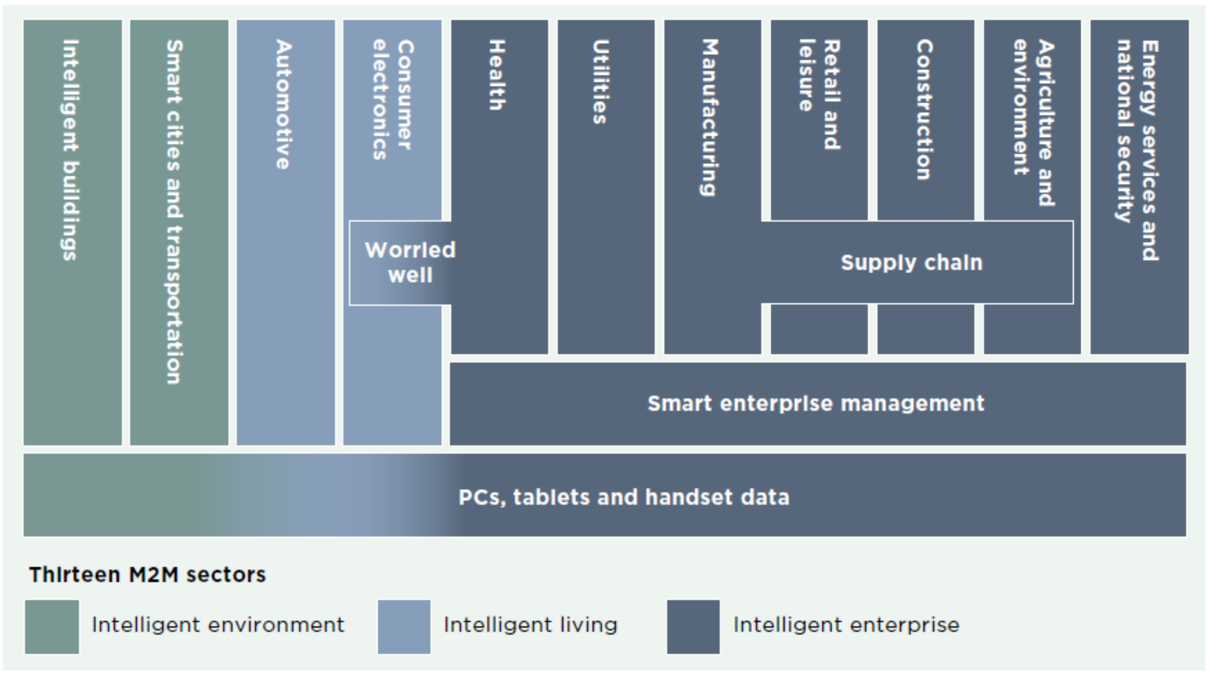


Fig 1.1 IOT Industry sector categories

Smart cities: In cities, the development of smart grids, data analytics and autonomous vehicles will provide an intelligent platform to deliver innovations in energy management, traffic management and security, sharing the benefits of this technology throughout society.

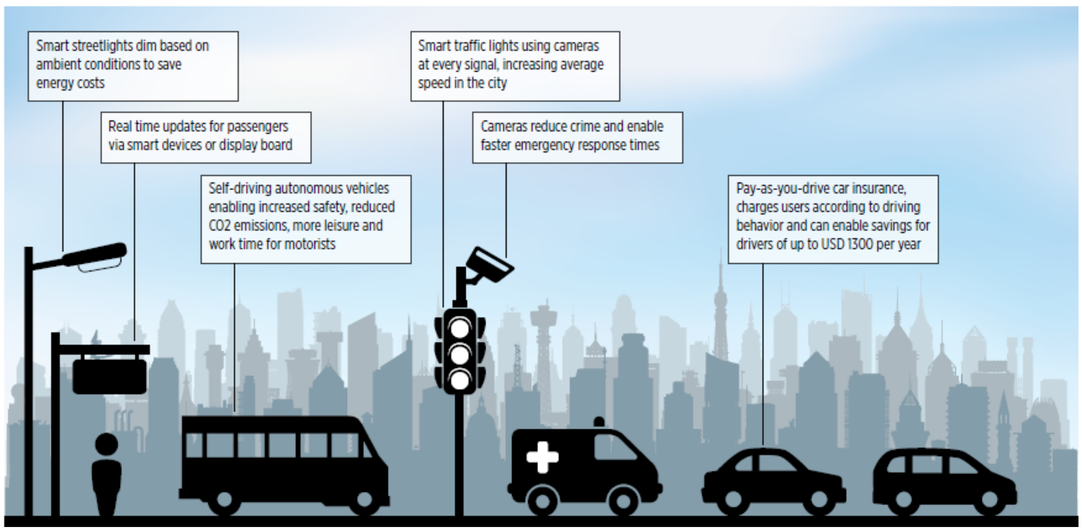

Smart streetlights dim based on ambient conditions to save energy costs
Real time updates for passengers via smart devices or display board
Self-driving autonomous vehicles enabling increased safety, reduced CO2 emissions, more leisure and work time for motorists
Smart traffic lights using cameras at every signal, increasing average speed in the city
Cameras reduce crime and enable faster emergency response times
Pay-as-you-drive car insurance, charges users according to driving behavior and can enable savings for drivers of up to USD 1300 per year

Fig 1.2 IOT Smart Cities Categories

Health Applications: The IoT will also help widen access and improve quality of education and health. As demand for healthcare doubles8, connected smart devices will help address this challenge by supporting a range of e-health services that improve access and enable monitoring of chronic diseases and age-related conditions in the home. In doing so, they will improve the quality of care and quality of life for patients, while reducing the strain on the wider healthcare system.

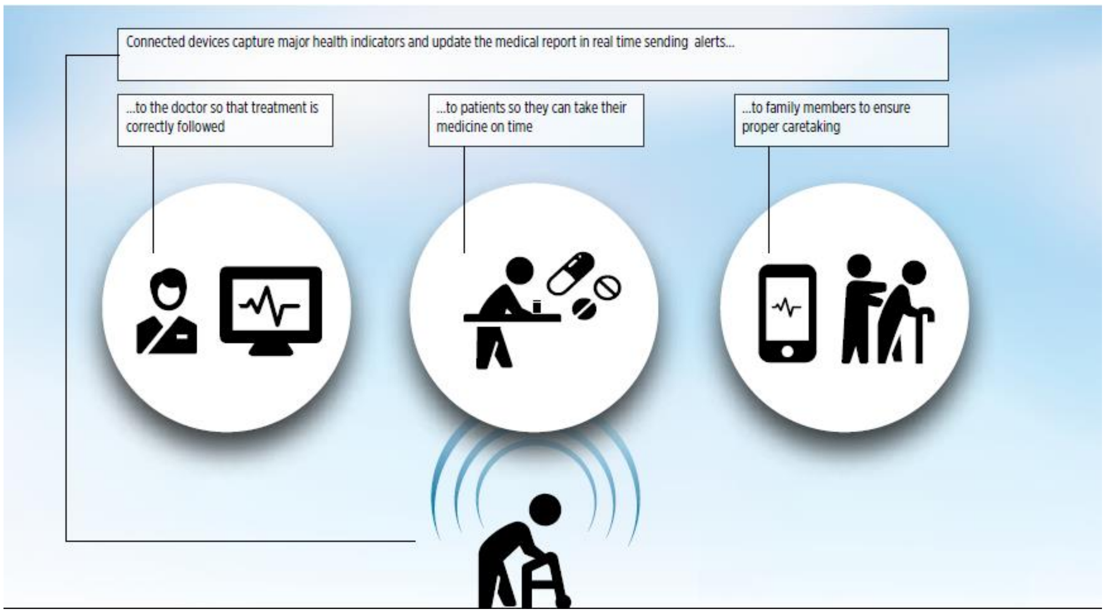

Connected devices capture major health indicators and update the medical report in real time sending alerts...
...to the doctor so that treatment is correctly followed
...to patients so they can take their medicine on time
...to family members to ensure proper caretaking

Fig 1.3 IOT Healthcare Categories

Education Applications: In education, mobile-enabled solutions will tailor the learning process to each student's needs, improving overall proficiency levels, while linking virtual and physical classrooms to make learning more convenient and accessible. Mobile education solutions have already been shown to improve learners' proficiency rates and reduce dropout rates, and have the potential to enable, by 2017, the education of up to 180 million additional students in developing countries who will be able to stay in school due to Education.

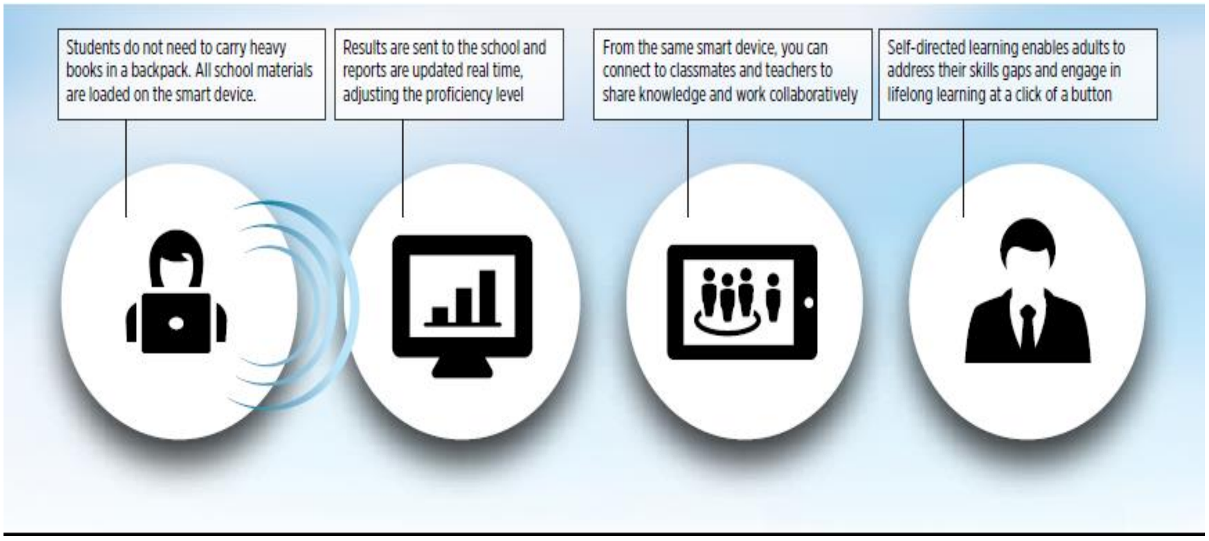

Students do not need to carry heavy books in a backpack. All school materials are loaded on the smart device.
Results are sent to the school and reports are updated real time, adjusting the proficiency level
From the same smart device, you can connect to classmates and teachers to share knowledge and work collaboratively
Self-directed learning enables adults to address their skills gaps and engage in lifelong learning at a click of a button

Fig 1.4 IOT Education Applications oriented

Productive applications: For enterprises, the ability of IoT to combine innovations in data analytics, 3D printing and sensors, will improve productivity by enabling a step change in the quality of decision making, efficiency of production, personalization of retail and productivity of food production.

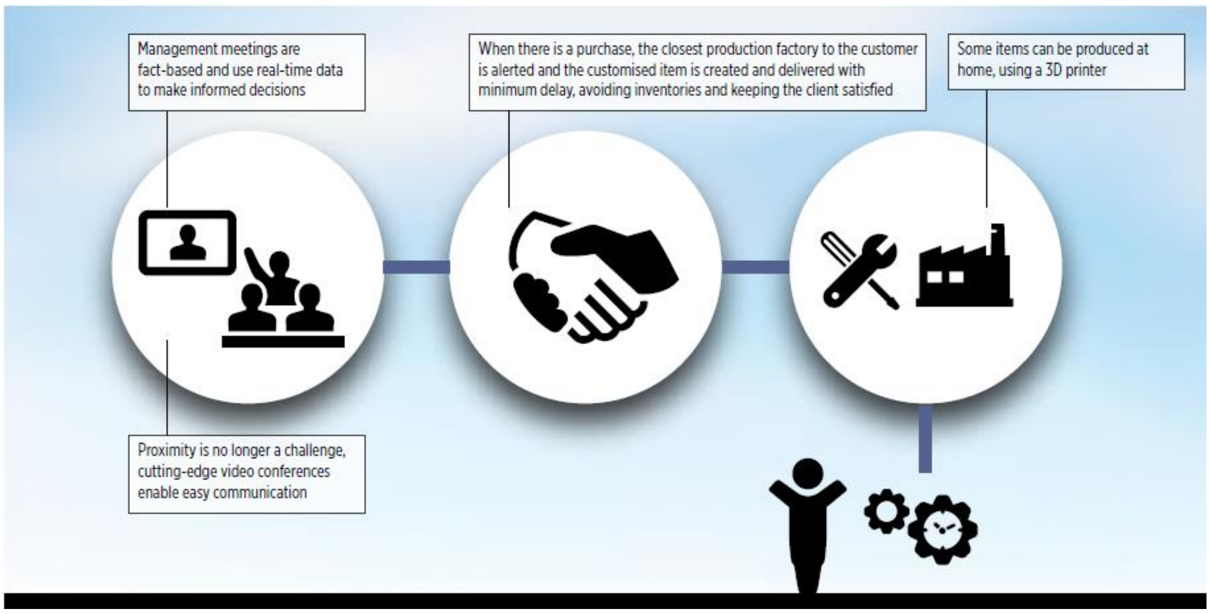

Management meetings are fact-based and use real-time data to make informed decisions
When there is a purchase, the closest production factory to the customer is alerted and the customised item is created and delivered with minimum delay, avoiding inventories and keeping the client satisfied
Some items can be produced at home, using a 3D printer
Proximity is no longer a challenge, cutting-edge video conferences enable easy communication

Fig 1.5 IOT Productive Applications

1. **3 EMBEDDED SYSTEMS**

A system is an arrangement in which all its unit assemble work together according to a set of rules. It can also be defined as a way of working, organizing or doing one or many tasks according to a fixed plan. For example, a watch is a time displaying system. Its components follow a set of rules to show time. If one of its parts fails, the watch will stop working. So we can say, in a system, all its subcomponents depend on each other An embedded system is a system that has software embedded into computer-hardware, which makes a system dedicated for an application or specific part of an application or product or part of a larger system. An embedded system is one that has a dedicated purpose software embedded in a computer hardware. It is a dedicated computer based system for an application or product. It may be an independent system or a part of large system. Its software usually embeds into a ROM (Read Only Memory) or flash.

As its name suggests, Embedded means something that is attached to another thing. An embedded system can be thought of as a computer hardware system having software embedded in it. An embedded system can be an independent system or it can be a part of a large system. An embedded system is a microcontroller or microprocessor based system which is designed to perform a specific task. For example, a fire alarm is an embedded system; it will sense only smoke. An embedded system has three components such as hardware, application software, Real Time Operating system (RTOS) that supervises the application software and provide mechanism to let the processor run a process as per scheduling by following a plan to control the latencies. RTOS defines the way the system works. It sets the rules during the execution of application program. A small scale embedded system may not have RTOS. So we can define an embedded system as a Microcontroller based, software driven, reliable, real-time control system.

In an embedded system the following components such as sensor, analog to digital converter(ADC), processor and ASICs, digital to analog converter, actuator. The sensor is used to measure the physical quantity and converts it to an electrical signal which can be read by an observer or by any electronic instrument like an A2D converter. A sensor stores the measured quantity to the memory. Analog-to-digital converter converts the analog signal sent by the sensor into a digital signal. Processors process the data to measure the output and store it to the memory. A digital-to-analog converter converts the digital data fed by the processor to analog data. The actuator compares the output given by the D-A Converter to the actual (expected) output stored in it and stores the approved output. The constraints of an Embedded System design are available system memory, available processor speed, limited power dissipation when running the system continuously in cycles of the system start, wait for event, wake-up and run, sleep and stop.

1.3.**1 CHARACTERISTICS OF EMBEDDED SYSTEM**

- Single-functioned – An embedded system usually performs a specialized operation and does the same repeatedly. For example: A pager always functions as a pager.
- Tightly constrained – All computing systems have constraints on design metrics, but those on an embedded system can be especially tight. Design metrics is a measure of an implementation's features such as its cost, size, power, and performance. It must be of a size to fit on a single chip, must perform fast enough to process data in real time and consume minimum power to extend battery life.
- Reactive and Real time – Many embedded systems must continually react to changes in the system's environment and must compute certain results in real time without any delay. Consider an example of a car cruise controller; it continually monitors and reacts to speed and brake sensors. It must compute acceleration or de-accelerations repeatedly within a limited time; a delayed computation can result in failure to control of the car.
- Microprocessors based – It must be microprocessor or microcontroller based.
- Memory – It must have a memory, as its software usually embeds in ROM. It does not need any secondary memories in the computer.
- Connected – It must have connected peripherals to connect input and output devices.

- HW-SW systems – Software is used for more features and flexibility. Hardware is used for performance and security.
- Frequently connected to physical environment through sensors and actuators,
- Hybrid systems (analog + digital parts).
- A real-time system must react to stimuli from the controlled object (or the operator) within the time interval dictated by the environment.
- For real-time systems, right answers arriving too late (or even too early) are wrong.
- All other time-constraints are called soft.
- A guaranteed system response has to be explained without statistical arguments
- Reliability, maintainability, safety and security.
- Dedicated functions.
- Dedicated complex algorithms.
- Dedicated and other user interfaces for the application

1.3.2 CLASSIFICATION OF EMBEDDED SYSTEMS

Embedded systems can be classified into different types based on performance, functional requirements and performance of the microcontroller.

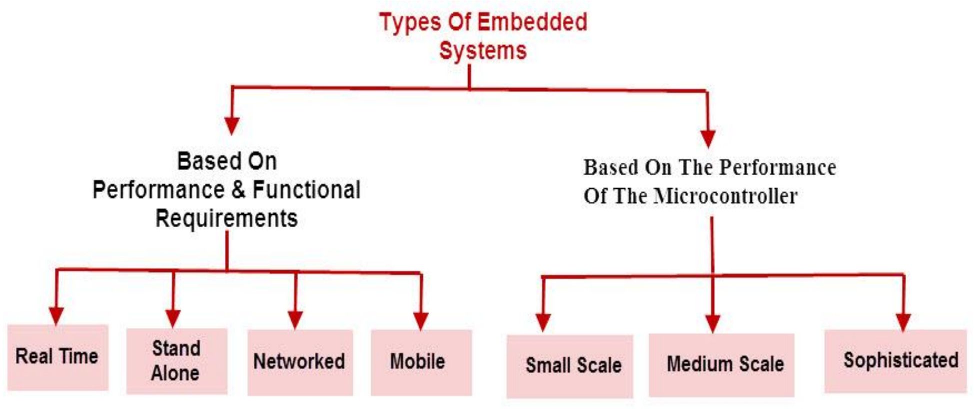

Types Of Embedded Systems
Based On Performance & Functional Requirements
Based On The Performance Of The Microcontroller
Real Time
Stand Alone
Networked
Mobile
Small Scale
Medium Scale
Sophisticated

Fig 1.6

Embedded systems are classified into four categories based on their performance and functional requirements:

- Stand alone embedded systems- Does not require a host system like a computer, it works by itself. It takes the input from the input ports either analog or digital and processes, calculates and converts the data and gives the resulting data through the connected device-Which either controls, drives and displays the connected devices. Examples for the stand alone embedded systems are mp3 players, digital cameras, video game consoles, microwave ovens and temperature measurement systems.
- Real time embedded systems- A real time embedded system is defined as, a system which gives a required o/p in a particular time. These types of embedded systems follow the time deadlines for completion of a task. Real time embedded systems are classified into two types such as soft and hard real time systems.
- Networked embedded systems- These types of embedded systems are related to a network to access the resources. The connected network can be LAN, WAN or the internet. The connection can be any wired or wireless. This type of embedded system is the fastest growing area in embedded system applications. The embedded web server is a type of system wherein all embedded devices are connected to a web server and accessed and controlled by a web browser.Example for the LAN networked embedded system is a home security system wherein all sensors are connected and run on the protocol TCP/IP.
- Mobile embedded systems- Mobile embedded systems are used in portable embedded devices like cell phones, mobiles, digital cameras, mp3 players and personal digital assistants, etc.The basic limitation of these devices is the other resources and limitation of memory.Embedded Systems are classified into three types based on the performance of the microcontroller such as
- Small scale embedded systems- These types of embedded systems are designed with a single 8 or 16-bit microcontroller, that may even be activated by

a battery. For developing embedded software for small scale embedded systems, the main programming tools are an editor, assembler, cross assembler and integrated development environment (IDE).

- **Medium scale embedded systems-** These types of embedded systems design with a single or 16 or 32 bit microcontroller, RISCs or DSPs. These types of embedded systems have both hardware and software complexities. For developing embedded software for medium scale embedded systems, the main programming tools are C, C++, JAVA, Visual C++, RTOS, debugger, source code engineering tool, simulator and IDE.
- **Sophisticated embedded systems-** These types of embedded systems have enormous hardware and software complexities, that may need ASIPs, IPs, PLAs, scalable or configurable processors. They are used for cutting-edge applications that need hardware and software Co-design and components which have to assemble in the final system.

1.4. AIM AND OBJECTIVE

The main goal was to create a reliable IoT-based patient monitoring system. Sensors, a data collecting unit, a microcontroller (i.e., Arduino UNO R3), and software were used to create a healthcare monitoring system that can offer real-time online information about a patient's physiological status (i.e., Arduino).

Real-time health monitoring systems using IoT can help doctors prioritize patients, and provide urgent care to those who are in the most danger, thereby saving lives. More competent patient management can help utilize the resources of the hospital more efficiently and save money.

1.5. SCOPE OF THE PROJECT

Remote healthcare is an emerging research field as the world moves towards remote monitoring, real-time and fast detection of illnesses. Remote healthcare has many categories, (e.g. telehealth, mobile health) all of which mean monitoring of patients outside hospital conditions by the means of technology. The advantages of remote monitoring of patients are: early and real-time detection of illnesses, ability to continuously monitor patients, prevention of worsening of illnesses and untimely deaths, cost reduction in hospitalizations, reduce the number of hospitalizations, obtain more accurate readings while permitting usual daily activities for patients, improve efficiency in healthcare services by utilizing communication technology, emergency medical care, service for patients with mobility issues, emergency care for traffic accidents and other injuries and usage of non-invasive medical interventions.

Remote monitoring of patients target several sub-groups of patients, such as patients diagnosed with chronic illnesses, patients with mobility issues, or other disability, post-surgery patients, neonates and elderly patients. All these types of patients have conditions that are better to be monitored continuously. The aim of good healthcare is the ability to support ordinary life as much comfortable as possible to all patients. Most research follow the policy of allowing the mobility and activity freedom at home or personal environments which are beneficial for the patients rather than being confined into a high-cost hospital room. Therefore, whole systems are being built to support this concept with the use of different technologies. With the new remote health monitoring applications, elderly patients can engage in daily activities without support from a caretaker. So, these applications support activities like sitting, standing, using the bathroom, watching television, reading and sleeping, with least inconvenience to the user. Even if there are wearable sensors, these pose minimum effect to the activities. One such example is smart wrist-watch based sensors.

For the patients of accidents and sudden injuries, the time they are monitored remotely could be only the time they are being transferred to the hospital in an ambulance. Nevertheless, efforts have been focused on safe journey to the hospital and remote monitoring helps for immediate medical interventions for highly critical cases. The doctors can monitor the deterioration or maintenance of the patient while also advising the paramedics who are physically with the patients as necessary.

Basic elements of a remote monitoring system are data acquisition system, data processing system, end-terminal at the hospital and the communication network. Data acquisition system is composed of different sensors or devices with embedded sensors with data transmission capability wirelessly. With the advancement of technology, sensors may not be medical sensors only; it could be cameras or smartphones. This is because, very recent research look into contactless methods where the devices do not touch the body of the patient

1.6. IMPORTANCE OF THE PROJECT

It will monitor the patients' heart rate, body temperature, blood oxygen saturation, blood pressure, glucose levels, skin perspiration, respiration rate, capnography, and other essential parameters. All this information is critical in preventing various illnesses, especially dangerous ones like heart attacks.

Common benefits included continuous monitoring of patients to provide prompt care, improvement of patient self-care, efficient communication.

CHAPTER 2

LITERATURE SURVEY

Here discussing the base papers and various literature papers methodogy and limitations.

2.1 VARIOUS LITERATURE PAPERS

1. Title: "IOT Based ICU Patient health monitoring System"

Journal Name: IEEE Access

Author: Lutfun Nabar, Syeda Samiha Zatar, Faria Binta Rafiq

Year: 2023

Methodology: The literature review paper provide an overview of iot based Smart healthcare application with implementation. It examines Various Care Such as ECG Sensor.

Limitations: While the Paper identifies key challenges and consideration for IOT implementation in healthcare it primarily focuses on ICU patients only.

2. Title: "A Versatile Data Fabric for advanced IOT based remote health monitoring"

Journal Name: IEEE Access

Author: Italo Bule je, Vince s.siu, kuan yu Hsieh, Nigel Hinds,Bing, Erhan Bilal.

Methodology: This paper presents a review of IOT and cloud computing technologies in the context of healthcare monitoring system. It discusses the collection of data.

Limitations: The paper primarily focuses on the conceptual framework and theoretical aspect of collection of data without providing a detailed implementation guidelines.

3. Title: "A Review of IoT and Cloud Computing in Healthcare System"

Journal Name: 2018 2nd International Conference on Inventive Systems and Control (ICISC)

Author: Banerjee, M., & Roy, S.

Year: 2018

Methodology: The paper conducts a comprehensive review of the integration of IoT and cloud computing technologies in healthcare systems. It discusses the benefits, challenges, and applications of these technologies in enhancing patient care, improving efficiency, and reducing healthcare costs.

Limitations: The paper mainly focuses on the theoretical aspects and potential benefits of IoT and cloud computing in healthcare, without providing detailed empirical evidence or case studies to validate the effectiveness of these technologies in real-world healthcare settings.

4. Title: "A review of IoT smart healthcare applications and challenges"

Journal Name: Wireless Networks

Author: Mekki, K., Al-Makhadmeh, Z., & Al-Jaroodi, J.

Year: 2018

Methodology: This literature review paper provides an overview of IoT-based smart healthcare applications and discusses the challenges associated with their implementation. It examines various use cases, such as remote patient monitoring, telemedicine, and wearable health devices, and discusses the technological, regulatory, and privacy challenges in deploying IoT solutions in healthcare.

Limitations: While the paper identifies key challenges and considerations for IoT implementation in healthcare, it primarily focuses on a high-level overview and does not delve into specific technical solutions or case studies to address these challenges.

5. Title: "IoT-based healthcare monitoring system: Design, development, and implementation"

Journal Name: IEEE Access

Author: Hossain, M. A., & Muhammad, G.

Year: 2019

Methodology: This paper presents the design, development, and implementation of an IoT-based healthcare monitoring system. It describes the architecture of the system, including sensor integration, data transmission, cloud storage, and user interface design. The paper also discusses the system's deployment in real-world healthcare settings and evaluates its performance.

Limitations: While the paper provides a detailed description of the system architecture and implementation, it lacks a comprehensive evaluation of the system's effectiveness, reliability, and scalability in real-world deployment scenarios.

6. Title: "Smart Health Monitoring System using IoT and Cloud Computing"

Journal Name: 2018 International Conference on Computing, Power and Communication Technologies (GUCON)

Author: Patil, V. P., & Naik, G. A.

Year: 2018

Methodology: This paper presents a review of IoT and cloud computing technologies in the context of smart healthcare monitoring systems. It discusses the architecture, components, and applications of such systems and evaluates their potential benefits in improving patient care and healthcare delivery.

Limitations: The paper primarily focuses on the conceptual framework and theoretical aspects of smart healthcare monitoring systems, without providing detailed implementation guidelines or empirical evidence to validate the effectiveness of these systems in real-world healthcare settings.

7. Title: "A Review of IoT and Cloud Computing Technologies for Smart Healthcare System"

Journal Name: Proceedings of the 4th International Conference on Deep Learning Technologies

Author: Chukwudi, E. M., & Chukwu, J.

Year: 2020

Methodology: This paper conducts a comprehensive review of IoT and cloud computing technologies for smart healthcare systems. It discusses various IoT-enabled healthcare applications, such as remote patient monitoring, telemedicine, and health data analytics, and evaluates the potential benefits and challenges of integrating IoT and cloud computing in healthcare.

Limitations: While the paper provides valuable insights into the potential applications and benefits of IoT and cloud computing in healthcare, it lacks detailed empirical evidence or case studies to support its findings and recommendations.

8. Title: "IoT and Cloud based Smart Health Monitoring System using ESP8266"

Journal Name: 2019 4th IEEE International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT)

Author: Sadashivappa, C., & Narayanaswamy, K.

Year: 2019

Methodology: This paper presents the design and implementation of an IoT and cloud-based smart health monitoring system using the ESP8266 microcontroller. It describes the system architecture, sensor integration, data transmission, and cloud storage components and evaluates the system's performance in real-world healthcare settings.

Limitations: While the paper provides a detailed description of the system design and implementation, it lacks a comprehensive evaluation of the system's scalability, reliability, and usability in diverse healthcare environments. Additionally, it does not discuss potential challenges or limitations associated with the implementation of such systems.

9. Title: "Development of Smart Healthcare Monitoring System using IoT Technology"

Journal Name: 2018 7th International Conference on Computer and Communication Engineering (ICCCE)

Author: Lim, C. P., Loo, C. K., Chow, W. L., Yap, V. V., & Hoo, W. Y.

Year: 2018

Methodology: This paper presents a literature review and development methodology for a smart healthcare monitoring system utilizing IoT technology. It discusses the integration of sensors, communication protocols, cloud platforms, and user interfaces to enable real-time monitoring and remote access to patient health data. The methodology includes system architecture design, sensor selection, data transmission protocols, and cloud platform integration.

Limitations: While the paper provides valuable insights into the design and development of smart healthcare monitoring systems, it primarily focuses on the technical aspects of system implementation and lacks a comprehensive evaluation of the system's performance, usability, and scalability in real-world healthcare settings. Additionally, it does not discuss potential challenges or limitations encountered during system deployment.

CHAPTER 3
SYSTEM ANALYSIS

Study of existing system, documenting of existing system and identifying

Current deficiency and establishing new goals.

3.1 **EXISTING SYSTEM**

Real-time Blood Oxygen Saturation Monitoring: Implement an SPO2 sensor to measure the blood oxygen saturation levels of the user in real-time. The acquired data will be processed and displayed on an LCD screen.

Vital Sign Display: Utilize an LCD display to show vital signs such as blood oxygen saturation levels, heart rate, and other relevant health parameters. The display should be user-friendly, providing clear and concise information.

Emergency Alert System: Integrate a buzzer into the system to generate an alert in case of abnormal health readings or critical situations. The alert system will be triggered based on predefined threshold values for vital signs

3.1.1Disadvantages

- Accuracy and Reliability-The accuracy and reliability of the health monitoring system heavily depend on the quality and calibration of the sensors used. Low-quality sensors may provide inaccurate readings, leading to false alarms or missed events.

- Power Consumption-Power consumption is a critical factor, especially for devices intended for continuous health monitoring. Arduino and GSM modules can consume significant power, which may require frequent recharging or the use of larger batteries.

- Size and Portability:Integrating multiple sensors and communication modules into a portable and user-friendly device can be challenging.

The size and weight of the monitoring device may affect user comfort and adherence.

- Cost-The cost of assembling a health monitoring system with various sensors and communication modules can add up quickly. This may limit accessibility for some users, particularly those in lower-income demographics.

3.2 PROPOSED SYSTEM

The proposed smart healthcare monitoring system integrates advanced technologies such as Arduino Uno, Max30100 temperature sensor, ECG sensor, wet sensor, Node MCU, LCD, and a buzzer to ensure comprehensive patient care. The Max30100 temperature sensor continuously monitor vital signs, including heart rate and body temperature. The ECG sensor provides real-time electrocardiogram data displayed on the serial plotter for quick analysis. The Node MCU facilitates seamless communication with the Ubidots Cloud, where all collected data is securely stored. The LCD screen provides a user-friendly interface for real-time monitoring, and the buzzer serves as an immediate alert system for critical events. This integrated solution ensures efficient and timely healthcare monitoring, allowing for early intervention and improved patient outcomes.

3.2.1 Advantages of proposed design:

- Real-time Monitoring-The system provides real-time monitoring of vital signs, such as heart rate, oxygen saturation, temperature, and patient position. Continuous monitoring allows for immediate detection of any abnormalities or critical changes in the patient's health.
- Cost-Effective-Arduino Uno is a cost-effective and versatile microcontroller platform, making the overall system affordable.

CHAPTER 4

PROJECT DESCRIPTION

Hardware Setup Begin by setting up the hardware components required for the monitoring system. This includes assembling the Arduino Uno, Max30100 pulse oximeter sensor, ECG sensor, Node MCU, LCD display, and buzzer.

4.1 PROPOSED METHODOLOGY

Sensor Integration Connect each sensor to the Arduino Uno microcontroller using appropriate wiring and interfaces. Configure the Arduino Uno to read data from each sensor at regular intervals, ensuring accurate and reliable data acquisition.

Data Transmission Use the Node MCU as a communication bridge to transmit sensor data to the Ubidots cloud platform. Implement communication protocols such as Wi-Fi or MQTT to establish a secure and reliable connection between the Node MCU and Ubidots.

Cloud Integration Set up an account on the Ubidots platform and create data variables corresponding to each sensor measurement (e.g., pulse rate, temperature, position, ECG data, wetness level). Configure the Node MCU to send sensor data to the respective variables in the Ubidots dashboard using the Ubidots API or MQTT client.

Data Visualization Design a user-friendly interface for visualizing sensor data in the Ubidots dashboard. Create custom widgets and dashboards to display real-time measurements, historical trends, and alerts for abnormal values. Configure thresholds and triggers to generate alerts for critical health events.

Local Feedback Develop code to display sensor data on the LCD display for local feedback. Implement logic to update the display with real-time measurements and status indicators, providing immediate feedback to caregivers and patients.

Alert System Integrate a buzzer alert system to notify caregivers of critical health events. Define alarm conditions based on sensor thresholds or predefined criteria, triggering the buzzer to alert caregivers of urgent situations.

Testing and Validation Conduct thorough testing of the monitoring system to ensure proper functionality, accuracy, and reliability. Validate sensor measurements against known standards and conduct simulated scenarios to evaluate system responsiveness and performance.

Optimization and Deployment Fine-tune the system parameters, optimize code efficiency, and address any identified issues or bugs. Once validated, deploy the monitoring system in real-world healthcare settings, providing training and support to healthcare professionals and caregivers as needed.

Continuous Monitoring and Maintenance Monitor the system continuously for data integrity, connectivity issues, and performance optimization. Implement regular maintenance procedures, including sensor calibration, software updates, and security audits, to ensure the long-term reliability and effectiveness of the monitoring system.

4.2.1. Admin Interface Module

The admin interface module allows the admin to login to the Phish Blocker website with their credentials. Once logged in, the admin can train the model with new phishing URLs and HTML pages, which will be used for real-time detection and blocking of phishing websites. The admin can view and manage the trained models, as well as view the attack history and analytics.

4.2 ARDUINO

The Arduino Uno is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Uno differs from all preceding boards in that it does not use the FTDIUSB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

4.2.2. SPECIFICATION

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically.External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.The board can operate on an external supply of 6 to 20 volts.

If supplied with less than 7V, however, the 5Vpin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts. Figure 5.2 depicts the Top view of Arduino Uno.

4.2.3 INTERFACING ARDUINO WITH EXTERNAL BOARD

- Using 12C Bus
- Serial Communication
- SPI
- USB Micro Port

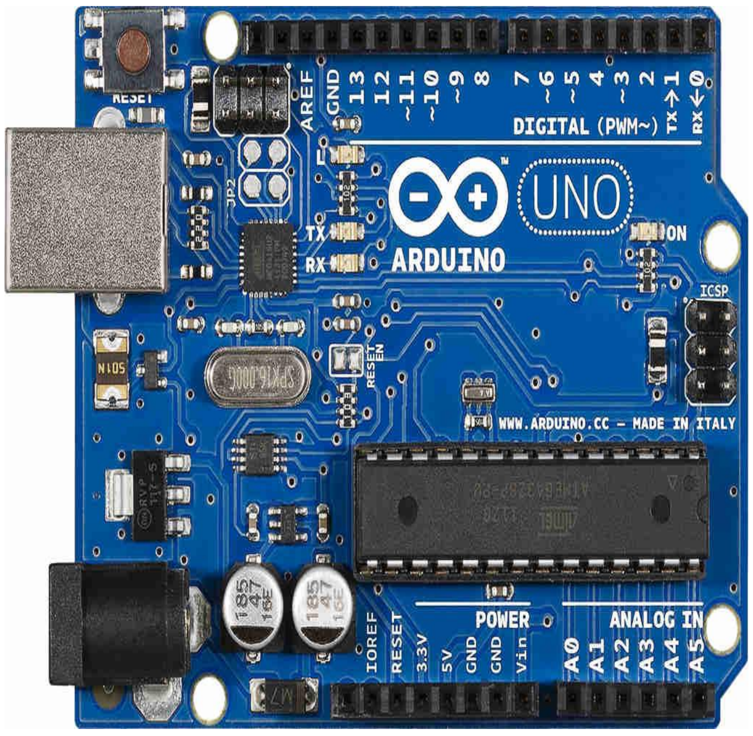


Fig 4.1 Arduino Uno

USB SERIAL COMMUNICATION BUS

USB communications device class (or USB CDC) is a composite Universal Serial Bus device class. The class may include more than one interface, such as a custom control interface, data interface, audio, or mass storage related interfaces.The communications device class is used for computer networking devices akin to a network card, providing an interface for transmitting Ethernet or ATM frames onto some physical media. It is also used for modems, ISDN, fax machines, and telephony applications for performing regular voice calls.Microsoft Windows versions prior to Windows Vista do not work with the networking parts of the USB CDC, instead using Microsoft's own derivative named Microsoft RNDIS, a serialized version of the Microsoft NDIS (Network Driver Interface Specification). With a vendor-supplied INF , Windows Vista works with USB CDC and USB WMCDC devices. Figure 5.3 shows the USB Serial Cable.

Fig 4.2 USB serial cable

This class can be used for industrial equipment such as CNC machinery to allow upgrading from older RS-232 serial controllers and robotics, since they can keep software compatibility. The device attaches to an RS-232 communications line and the operating system on the USB side makes the USB device appear as a traditional RS-232 port. Chip manufacturers such as Prolific Technology, FTDI, Microchip, and Atmel provide facilities for easily developing USB RS-232 devices.Devices of this class are also implemented in embedded systems such as mobile phones so that a phone may be used as a modem, fax or network port. The data interfaces are generally used to perform bulk data transfer.

DIGITAL:

An electronic signal transmitted as binary code that can be either the presence or absence of current, high and low voltages or short pulses at a particular frequency. Humans perceive the world in analog, but robots, computers and circuits use Digital. A digital signal is a signal that has only two states. These states can vary depending on the signal, but simply defined the states are ON or OFF, never in between. In the world of Arduino, Digital signals are used for everything with the exception of Analog Input. Depending on the voltage of the Arduino the ON or HIGH of the Digital signal will be equal to the system voltage, while the OFF or

LOW signal will always equal 0V. This is a fancy way of saying that on a 5V Arduino the HIGH signals will be a little under 5V and on a 3.3V Arduino the HIGH signals will be a little under 3.3V. To receive or send Digital signals the Arduino uses Digital pins # 0 - # 13. You may also setup your Analog In pins to act as Digital pins. To set up Analog In pins as Digital pins use the command: pinMode(pinNumber, value);where pinNumber is an Analog pin (A0 – A5) and value is either INPUT or OUTPUT. To setup Digital pins use the same command but reference a Digital pin for pinNumber instead of an Analog In pin.

Digital pins default as input, so really you only need to set them to OUTPUT in pin Mode. To read these pins use the command: digital Read (pin Number); where pin Number is the Digital pin to which the Digital component is connected. The digital Read command will return either a HIGH or a LOW signal. To send a Digital signal to a pin use the command: digital Write (pin Number, value); where pin Number is the number of the pin sending the signal and value is either HIGH or LOW. The Arduino also has the capability to output a Digital signal that acts as an Analog signal, this signal is called Pulse Width Modulation (PWM). Digital Pins # 3, # 5, # 6, # 9, # 10 and #11 have PWM capabilities. To output a PWM signal use the command: analog Write (pin Number, value); where, pin Number is a Digital Pin with PWM capabilities and value is a number between 0 (0%) and 255 (100%). For more information on PWM see the PWM worksheets or S.I.K. circuit 12. Examples of Digital: Values: On/Off, Men's room/Women's room, pregnancy, consciousness, the list goes on....ANN (Image Processing)/Interfaces: Buttons, Switches, Relays, CDs, etc....

Things to remember about Digital:

- Digital Input/Output uses the Digital pins, but Analog In pins can be used as Digital

- To receive a Digital signal use: digital Read(pin Number);
- To send a Digital signal use: digital Write(pin Number, value);
- Digital Input and Output are always either HIGH or LOW

ANALOG:

A continuous stream of information with values between and including 0% and 100%. Humans perceive the world in analog. Everything we see and hear is a continuous transmission of information to our senses. The temperatures we perceive are never 100% hot or 100% cold, they are constantly changing between our ranges of acceptable temperatures. This continuous stream is what defines analog data. Digital information, the complementary concept to Analog, estimates analog data using only ones and zeros. In the world of Arduino an Analog signal is simply a signal that can be HIGH (on), LOW (off) or anything in between these two states. This means an Analog signal has a voltage value that can be anything between 0V and 5V (unless you mess with the Analog Reference pin).

Analog allows you to send output or receive input about devices that run at percentages as well as on and off. The Arduino does this by sampling the voltage signal sent to these pins and comparing it to a voltage reference signal (5V). Depending on the voltage of the Analog signal when compared to the Analog Reference signal the Arduino then assigns a numerical value to the signal somewhere between 0 (0%) and 1023 (100%). The digital system of the Arduino can then use this number in calculations and sketches. To receive Analog Input the Arduino uses Analog pins # 0 - # 5.

These pins are designed for use with components that output Analog information and can be used for Analog Input. There is no setup necessary, and to read them use the command:

Analog Read(pin Number);

where pin Number is the Analog In pin to which the the Analog component is connected. The analog Read command will return a number including or between 0 and 1023.The Arduino also has the capability to output a digital signal that acts as an Analog signal, this signal is called Pulse Width Modulation (PWM). Digital Pins # 3, # 5, # 6, # 9, # 10 and #11 have PWM capabilities. To output a PWM signal use the command: analog Write (pin Number, value); where pin Number is a Digital Pin with PWM capabilities and value is a number between 0 (0%) and 255 (100%). On the Arduino UNO PWM pins are signified by a ~ sign. For more information on PWM see the PWM worksheets or S.I.K. circuit 12. Examples of Analog:Values: Temperature, volume level, speed, time, light, tide level, spiciness, the list goes on....ANN (Image Processing): Temperature sensor, Photoresistor, Microphone, Turntable, Speedometer, etc....

Things to remember about Analog:

- Analog Input uses the Analog In pins, Analog Output uses the PWM pins
- To receive an Analog signal use:

  Analog Read(pin Number);

- To send a PWM signal use:

  Analog Write(pin Number, value);

- Analog Input values range from 0 to 1023 (1024 values because it uses 10 bits, 210)
- PWM Output values range from 0 to 255 (256 values because it uses 8 bits, 28 Output Signals.

A signal exiting an electrical system, in this case a microcontroller. Output to the Arduino pins is always Digital, however there are two different types of

Digital Output; regular Digital Output and Pulse Width Modulation Output (PWM). Output is only possible with Digital pins # 0 - # 13. The Digital pins are preset as Output pins, so unless the pin was used as an Input in the same sketch, there is no reason to use the pin Mode command to set the pin as an Output. Should a situation arise where it is necessary to reset a Digital pin to Output from Input use the command:

```
pin Mode(pin Number, OUTPUT);
```

where pin Number is the Digital pin number set as Output. To send a Digital Output signal use the command:

```
digital Write(pin Number, value);
```

where pin Number is the Digital pin that is outputting the signal and value is the signal. When outputting a Digital signal value can be either HIGH (On) or LOW (Off).

Digital Pins # 3, # 5, # 6, # 9, # 10 and #11 have PWM capabilities. This means you can Output the Digital equivalent of an Analog signal using these pins. To Output a PWM signal use the command:

```
Analog Write(pin Number, value);
```

where pin Number is a Digital Pin with PWM capabilities and value is a number between 0 (0%) and 255 (100%). For more information on PWM see the PWM worksheets or S.I.K. circuit 12.Output can be sent to many different devices, but it is up to the user to figure out which kind of Output signal is needed, hook up the hardware and then type the correct code to properly use these signals.

Things to remember about Output :

- Output is always Digital
- There are two kinds of Output: regular Digital or PWM (Pulse Width Modulation)
- To send an Output signal use

  Analog Write(pin Number, value);

- (for analog) or digital Write(pin Number , value);(for digital)
- Output pin mode is set using the pin Mode command:

  pin Mode(pin Number, OUTPUT);

- Regular Digital Output is always either HIGH or LOW
- PWM Output varies from 0 to 255

INPUT SIGNALS

A signal entering an electrical system, in this case a microcontroller. Input to the Arduino pins can come in one of two forms; Analog Input or Digital Input.Analog Input enters your Arduino through the Analog In pins # 0 - # 5. These signals originate from analog ANN (Image Processing) and interface devices. These analog ANN (Image Processing) and devices use voltage levels to communicate their information instead of a simple yes (HIGH) or no (LOW). For this reason you cannot use a digital pin as an input pin for these devices. Analog Input pins are used only for receiving Analog signals. It is only possible to read the Analog Input pins so there is no command necessary in the setup( )function to prepare these pins for input. To read the Analog Input pins use the command:

Analog Read(pin Number);

where pin Number is the Analog Input pin number. This function will return an Analog Input reading between 0 and 1023. A reading of zero corresponds to 0

Volts and a reading of 1023 corresponds to 5 Volts. These voltage values are emitted by the analog ANN (Image Processing) and interfaces.

If you have an Analog Input that could exceed Vcc + .5V you may change the voltage that 1023 corresponds to by using the Aref pin. This pin sets the maximum voltage parameter your Analog Input pins can read. The Aref pin's preset value is 5V.Digital Input can enter your Arduino through any of the Digital Pins # 0 - # 13. Digital Input signals are either HIGH (On, 5V) or LOW (Off, 0V). Because the Digital pins can be used either as input or output you will need to prepare the Arduino uno to use these pins as inputs in your setup( ) function.

To do this type the command:

pin Mode(pin Number, INPUT);

inside the curly brackets of the setup( )function where pin Number is the Digital pin number you wish to declare as an input. You can change the pin Mode in the loop( ) function if you need to switch a pin back and forth between input and output, but it is usually set in the setup ( ) function and left untouched in the loop( ) function. To read the Digital pins set as inputs use the command:

digital Read(pin Number);

where pin Number is the Digital Input pin number.

Input can come from many different devices, but each device's signal will be either Analog or Digital, it is up to the user to figure out which kind of input is needed, hook up the hardware and then type the correct code to properly use these signals.

Things to remember about Input:

- Input is either Analog or Digital, make sure to use the correct pins depending on type.
- To take an Input reading use

  Analog Read(pin Number);(for analog)

- Or digital Read(pin Number);(for digital)
- Digital Input needs a pin Mode command such as

  pin Mode(pin Number, INPUT);

- Analog Input varies from 0 to 1023
- Digital Input is always either HIGH or LOW
-

4.3 Node MCU

- NodeMCU is an open-source firmware and development kit that helps you to prototype or build IoT products. It includes firmware that runs on the ESP8266 Wi-Fi SoC from Espress if Systems, and hardware which is based on the ESP-12 module. ... It is based on the eLua project and built on the Espressif Non-OS SDK for ESP8266.

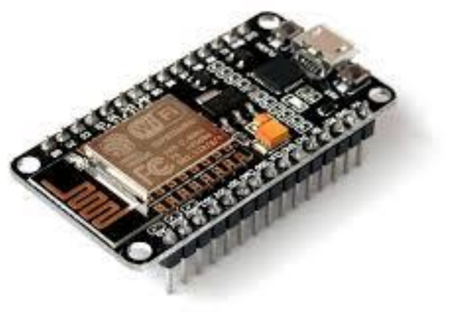


Fig 4.3 Node MCU

4.4 Max30100

The MAX30100 is an integrated pulse oximetry and heart- rate monitor sensor solution. It combines two LEDs, a photodetector, optimized optics, and low-noise analog signal processing to detect pulse oximetry and heart-rate signals. This library is designed to interface with the MAXIM MAX30100 Heart Rate and SpO2 sensor chip. The current status of the library allows reading of the IR values for Heart Rate determination and the manipulatuion of any register. The sensor has two light-emitting diodes and one photodiode. The LED's are used to emit the light and the photodiode is used to detect and measure the intensity of the received light. In MAX30100 one LED emits monochromatic light and the other LED emits infrared light. Advanced functionality improves measurement performance, high SNR provides robust motion artifact resilience integrated. ambient, light cancellation high sample rate capability fast data output capability. It is an integrated pulse oximetry and heart rate monitor sensor solution.

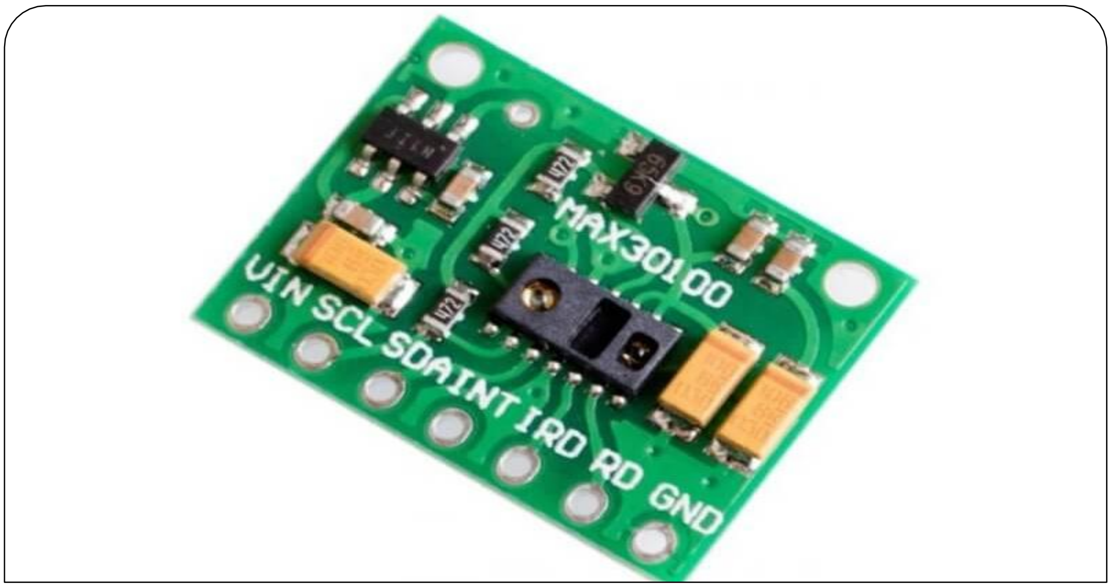


Fig 4.4 Max30100

4.7 ECG sensor

Electrocardiogram (ECG) sensor is a device commonly used by cardiologists to check for abnormal heart rhythm and signs of potential heart disease quickly and without intervention. This sensor can be used to extract heart rate data and other ECG features, enabling its application in research fields such as biomedical, biofeedback, psychophysiology, and sports, among many others. Compatibility: This sensor is only compatible with biosignalsplux acquisition system. The basic principle of the ECG is that stimulation of a muscle alters the electrical potential of the muscle fibres. Cardiac cells, unlike other cells, have a property known as automaticity, which is the capacity to spontaneously initiate impulses. The ECG is the most commonly used signal in the healthcare domain for analyzing heart and overall patient health. Acquisition of the ECG is fairly straightforward and non-invasive; surface electrodes are used on the limbs and/or the chest.

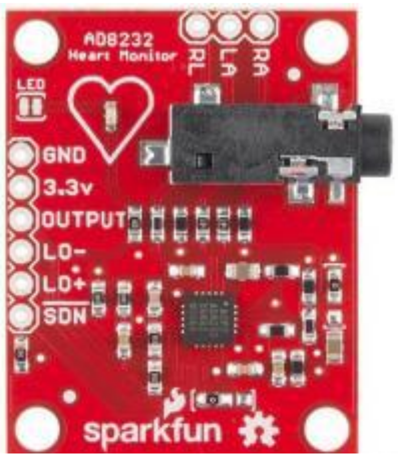


Fig 4.5 ECG Sensor

4.9 LCD Display

The LCD (Liquid Crystal Display) plays an essential role in the user interface of the intelligent power management system, providing real-time feedback and information display to users. In the context of the described Arduino project utilizing renewable energy sources like solar panels and a small windmill, the LCD serves several key functions:

The primary function of the LCD is to display critical system parameters and data to users, such as energy generation levels, battery status, voltage levels, current readings, and system alerts. This real-time information enables users to monitor the performance of the system and make informed decisions regarding energy usage and management.

The LCD provides visual feedback to users regarding the status and operation of the system, including notifications, warnings, and error messages. This feedback helps users identify potential issues or anomalies in the system and take appropriate actions to address them, enhancing system reliability and user experience.

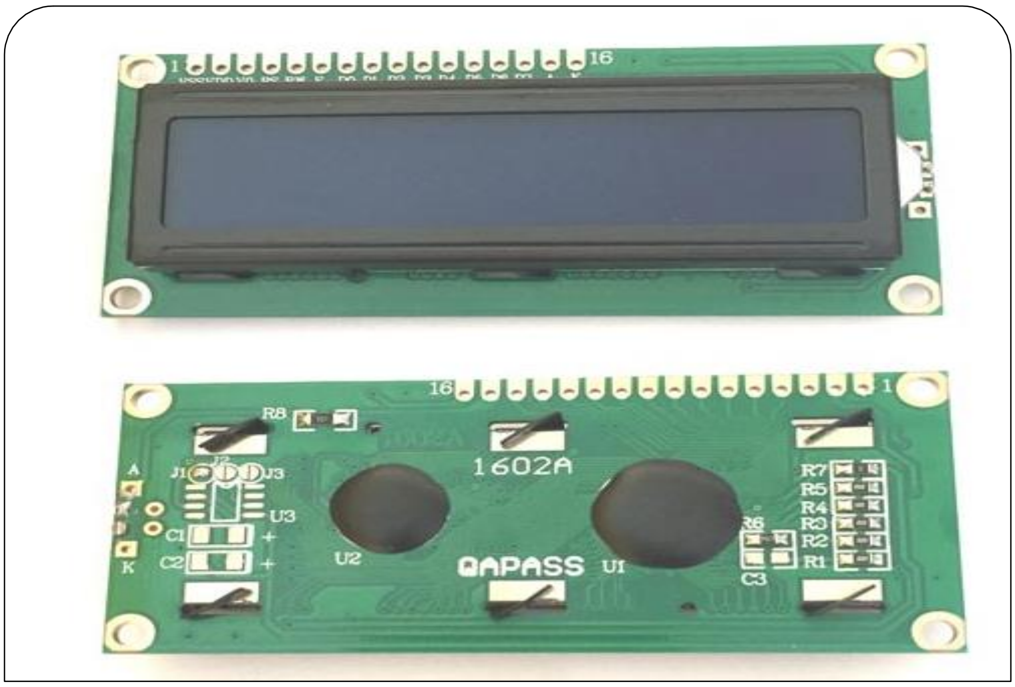


Fig 4.6 LCD Display

4.10 Buzzer

The buzzer serves as an auditory notification device within the intelligent power management system, providing audible alerts and feedback to users. In the context of the described Arduino project leveraging renewable energy sources such as solar panels and a small windmill, the buzzer performs several important functions:

Alerts and Alarms-The primary function of the buzzer is to emit audible alerts and alarms to notify users of critical events or system conditions. These alerts may include low battery warnings, overvoltage or undervoltage conditions, system faults, or emergency situations. The buzzer's sound serves as an effective means of drawing immediate attention to important system events, ensuring timely intervention and action.

Error Indication-The buzzer can be used to indicate error conditions or malfunctions within the system, such as sensor failures, communication errors, or hardware faults. By emitting distinct sounds or patterns, the buzzer communicates the nature of the error to users, facilitating troubleshooting and diagnosis of system issues.

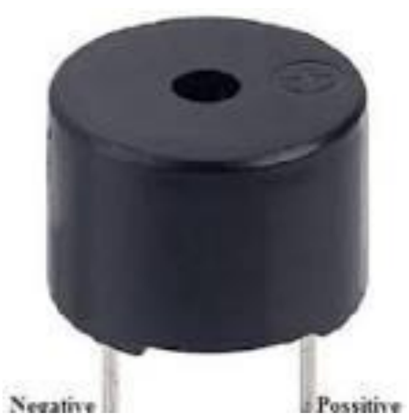


Fig 4.7 Buzzer

CHAPTER 5

IMPLEMENTATION

Functional Testing- Verify that each component of the monitoring system operates as intended. Test sensor readings, data transmission, cloud integration, and user interface functionality to ensure proper communication and data visualization.

Accuracy Testing- Validate the accuracy of sensor measurements by comparing them against known standards or reference devices. Conduct calibration procedures to adjust sensor readings if necessary and ensure consistent and reliable measurements.

Reliability Testing- Assess the system's reliability by monitoring its performance over an extended period. Conduct stress tests, such as continuous operation and data transmission under varying environmental conditions, to identify potential failure points or performance degradation.

Usability Testing- Evaluate the user experience of the monitoring system from both caregiver and patient perspectives. Assess the clarity of data visualization, ease of navigation in the Ubidots dashboard, and responsiveness of local feedback mechanisms (e.g., LCD display, buzzer alerts).

Data Security Testing- Verify the effectiveness of data security measures implemented in the system. Test encryption protocols, access controls, and data encryption mechanisms to ensure the confidentiality, integrity, and availability of patient health information.

Arduino Uno:

Role: The main microcontroller that manages and controls the entire system.

Functionality: Reads data from various sensors, processes the information, and sends it to Ubidots cloud through Node MCU.

Max30100 Pulse Oximeter (Heart Rate and Oxygen Saturation Sensor):

Role: Measures the heart rate and oxygen saturation levels in the patient's blood.

Functionality: Provides real-time data on the patient's vital signs, which can be sent to Ubidots for remote monitoring.

ECG Sensor:

Role: Monitors the patient's electrocardiogram (ECG) signals.

Functionality: Reads electrical signals generated by the heart and plots the ECG data on the serial plotter. This information helps in assessing the patient's cardiac health.

Node MCU:

Role: Connects the Arduino Uno to the Ubidots cloud platform.

Functionality: Sends the collected sensor data to the Ubidots cloud for remote monitoring and storage. The Node MCU acts as a bridge between the local Arduino system and the cloud.

LCD (Liquid Crystal Display):

Role: Displays real-time health data locally.

Functionality: Provides a local interface for displaying vital signs and other relevant information, ensuring that caregivers or medical professionals can quickly access critical data without needing to access the cloud platform.

5.2.1 PROJECT CODE:

Ecg.ino

```
const int ECG_PIN = A0; // ECG sensor connected to analog pin A0

const int LON_PIN = A1; // Lead-off negative connected to analog pin A1

const int LOP_PIN = A2; // Lead-off positive connected to analog pin A2


void setup() {

  Serial.begin(9600); // Initialize serial communication at 9600 baud

}


void loop() {

  // Read analog values from sensors

  int ecgValue = analogRead(ECG_PIN);

  int lonValue = analogRead(LON_PIN);

  int lopValue = analogRead(LOP_PIN);


  // Print the sensor values to Serial Monitor

  Serial.print("ECG: ");

  Serial.print(ecgValue);

  Serial.print(" | LON: ");

  Serial.print(lonValue);

  Serial.print(" | LOP: ");

  Serial.println(lopValue);
  // Delay for a short period of time before the next reading
  delay(100); // Adjust delay as needed
}
```

Patient_monitoring

```
#include <Arduino.h>
#include <avr/io.h>
#include <stdlib.h>
#include <string.h>
#include "gpio.h"
#include "utils.h"
#include "adc.h"
#include "lcd.h"
#include "uart.h"
#include "esp.h"
#include "tcp.h"
#include "i2c.h"
#include "max30100.h"
#ifndef iotsecureconfig
#define iotsecureconfig
#define username  "vickydanya"
#define mailid    "gyq69182@fosiq.com"
#define password  "PIC16f877a"
```

```
#define api       "BBUS-pPO8lntJNiJcX2STPCJw5bP8jfgoMJ"

#endif

#ifndef lcdpinconfig

#define lcdpinconfig

#define rs  13

#define en  12

#define d4  11

#define d5  10

#define d6  9

#define d7  8

#endif

#ifndef alarmpinconfig

#define alarmpinconfig

#define alarm 2

#endif


#ifndef ledpinconfig

#define ledpinconfig

#define greenled 3

#endif

#ifndef ecgpinconfig

#define ecgpinconfig

#define output  AN0
```

```
#define lon    AN1

#define lop    AN2

#endif

#ifndef spo2pinconfig

#define spo2pinconfig

#define spo2data   AN4

#define spo2clock  AN5

#define spo2speed  400

#define spo2address 0xAE

#endif

#ifndef esppinconfig

#define esppinconfig

#define esptransmitter 1

#define espreceiver   0

#define espbaudrate   115200

#endif

#ifndef maxthreshconfig

#define maxthreshconfig

#define tempminimum  35

#define tempmaximum  45

#define heartminimum 60

#define heartmaximum 120

#define spo2minimum  95
```

```
#endif

const char *heart = "\x0A\x1F\x1F\x1F\x1F\x0E\x04\x00";

max30100_result_variables max30100;

int ecgwave, beatperminute, beatindex;

float beats[250] = {0.0F};

float threshold = 415.0F;

bool belowthresh = false;

uint32_t beatold = 0;

uint32_t updateinterval = 0;

void setup()

{

  gpio_set_output(alarm);

  gpio_put_low(alarm);

  gpio_set_output(greenled);

  gpio_put_low(greenled);

  gpio_set_input(output);

  gpio_set_input(lon);

  gpio_set_input(lop);

  #ifdef _ESP8266_H

  esp_initialize(espbaudrate);

  #endif

  adc_initialize();

  lcd_initialize(rs, en, d4, d5, d6, d7);
  lcd_disp(0x80,"PATIENT MONITORI");
  lcd_disp(0xC0,"    SYSTEM      ");
  delay_ms(2500); lcd_function(clear_display);
  if(esp_found()) lcd_disp(0x80,"ESP8266 FOUND");
  else lcd_disp(0x80,"ESP NOT FOUND"); delay_ms(500);
  lcd_function(clear_display);
 if(esp_wifi_mode(STATION)) lcd_disp(0x80,"MODE CONFIG");
  else lcd_disp(0x80,"MODE ERROR"); delay_ms(500);
  lcd_function(clear_display);
 if(esp_wifi_connect()) lcd_disp(0x80,"WIFI CONNECTED");
  else lcd_disp(0x80,"WIFI ERROR"); delay_ms(500);
  lcd_function(clear_display);
if(tcp_start()) lcd_disp(0x80,"TCP CONNECTED");
  else lcd_disp(0x80,"TCP ERROR"); delay_ms(500);
  lcd_function(clear_display); serial_flush();
  if(max30100_initialize(spo2speed)) lcd_disp(0x80,"MAX30100 FOUND");
  else lcd_disp(0x80,"MAX NOT FOUND"); delay_ms(500);
  lcd_function(clear_display); serial_flush();


  max30100_set_mode(active_hr | active_spo2);
  max30100_set_ir_current(ir_17_4_v);
  updateinterval = millis() + 10000;
}
```

```
void loop()
{
 max30100_hold_update(&max30100);
 lcd_disp(0x80,"T:");
 lcd_decimal(0x82, max30100.temperature, 2, DEC);
 lcd_write(0x84, 'C');
 lcd_disp(0x86, "ECG:");
 lcd_decimal(0x8A, beatperminute, 3, DEC);
 lcd_disp(0x8D, "BPM");

 lcd_disp(0xC0, "SPO2:");
 lcd_decimal(0xC5, max30100.spo2, 3, DEC);
 lcd_write(0xC8, '%');

 lcd_disp(0xCA, "H:");
 lcd_decimal(0xCC, max30100.heartrate, 3, DEC);
 lcd_char(0xCF, 1, heart);

 if(max30100.temperature && max30100.heartrate && max30100.spo2)
 {
   if(((max30100.temperature < tempminimum)  || (max30100.temperature > tempmaximum))
||
   ((max30100.heartrate   < heartminimum) || (max30100.heartrate   > heartmaximum))  ||
   ((max30100.spo2 < spo2minimum)))
   {
     gpio_put_high(alarm);
     gpio_put_low(greenled);
   }
   else
   {
     gpio_put_high(greenled);
     gpio_put_low(alarm);
   }
 }
 if(!gpio_get(lop) && !gpio_get(lon))
 {
   ecgwave = read_adc(output);


   if(ecgwave > threshold && belowthresh)
   {
     belowthresh = false;
     uint32_t beatnew = millis();
     uint32_t diff = beatnew - beatold;
     float currentbpm = 60000 / diff;
     beats[beatindex] = currentbpm;
```

```
      float total = 0.0F;

      for(int i = 0; i < 250; i++)

      total = total + beats[i];

      beatperminute = int(total / 250);

      beatold = beatnew;

      beatindex = (beatindex + 1) % 250;

    }

    else if(ecgwave < threshold) belowthresh = true;

  }


  if(millis() > updateinterval)

  {

    tcp_send_raw("%s %d %s %d %s %d %s %d",

    variablelabel1, (unsigned)max30100.temperature,

    variablelabel2, (unsigned)max30100.heartrate,

    variablelabel3, (unsigned)max30100.spo2,

    variablelabel4, (unsigned)(beatperminute));

    delay_ms(100); serial_flush();

    updateinterval = millis() + 10000;

  }

}
```

5.3 PROPOSED BLOCK DIAGRAM:

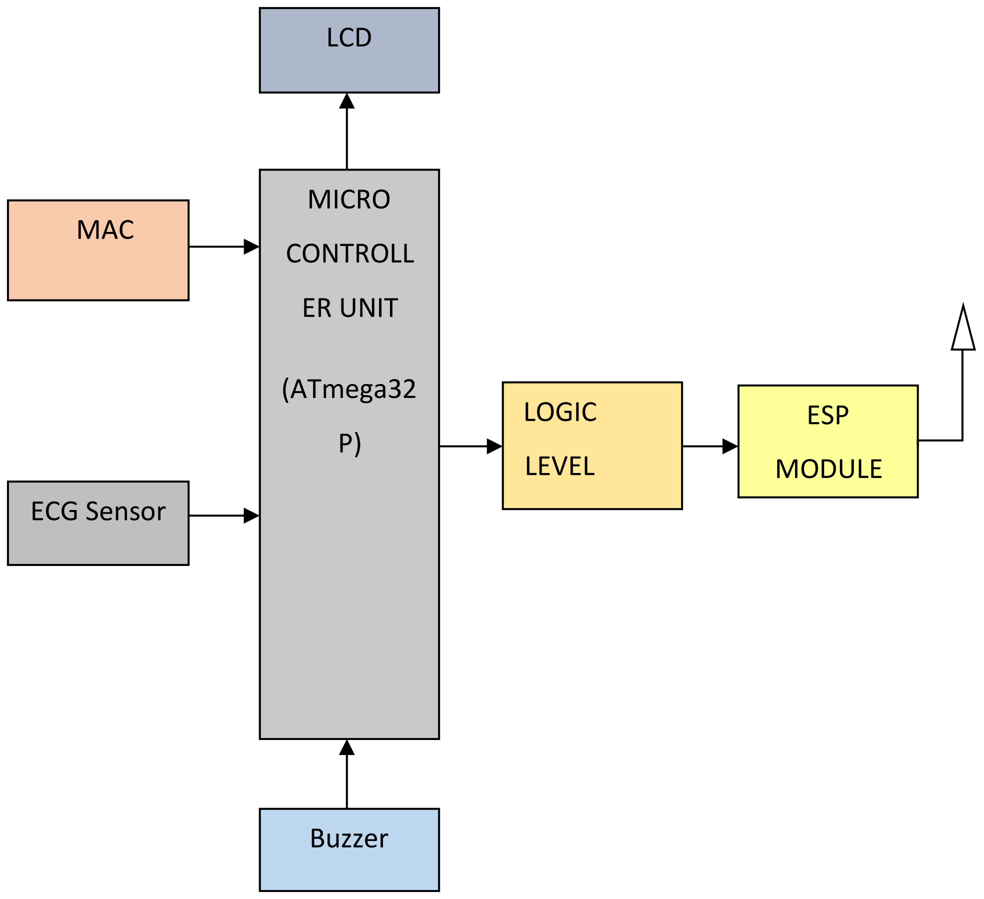


Fig 5.1 Proposed Block Diagram

Connecting the max30100 and ECG Sensor with micro controller unit Arduino UNO using Esp wifi module to store Data in Ubidots cloud and LCD is used for display the parameters to the patient view. The buzzer is used to alert the near by peoples the patients in abnormal condition.

CHAPTER 6

RESULTS & DISCUSSIONS

Overview of system testing outcomes accuracy testing reliability assessment data security scalability assessment integration testing and simulation testing and results are discussed here.

6.1 RESULTS:

Functional Testing Results- Provide an overview of the system's functionality testing outcomes, including the successful operation of sensors, data transmission, cloud integration, and user interface functionality. Discuss any observed issues or limitations encountered during testing and their resolutions.

Accuracy Testing Findings**-** Present the results of accuracy testing conducted to validate sensor measurements against known standards or reference devices. Discuss the degree of accuracy achieved by each sensor and any calibration procedures performed to enhance measurement precision.

Reliability Assessment- Describe the results of reliability testing, including stress tests conducted to evaluate the system's performance under varying environmental conditions and continuous operation. Discuss any instances of system failure or performance degradation observed during testing and their implications for real-world deployment.

Data Security Analysis: Present the results of data security testing, assessing the effectiveness of encryption protocols, access controls, and data encryption mechanisms in safeguarding patient health information. Discuss any vulnerabilities identified and recommendations for enhancing data security measures.

Scalability Assessment: Discuss the scalability testing results, including the system's performance under increased data loads and concurrent user access. Evaluate its ability to scale with growing healthcare demands and implications for scalability in real-world deployment scenarios.

Integration Testing Outcomes: Describe the findings from integration testing, assessing the interoperability of the monitoring system with existing healthcare infrastructure and information systems. Discuss the ease of data exchange and integration with electronic health records (EHR) systems and implications for seamless integration into healthcare workflows.

Simulation Testing Results: Present the outcomes of simulation testing, including the system's responsiveness and effectiveness in detecting critical health events under simulated healthcare scenarios and patient conditions. Discuss the system's ability to generate timely alerts and support clinical decision-making.

6.2 HARDWARE KIT:

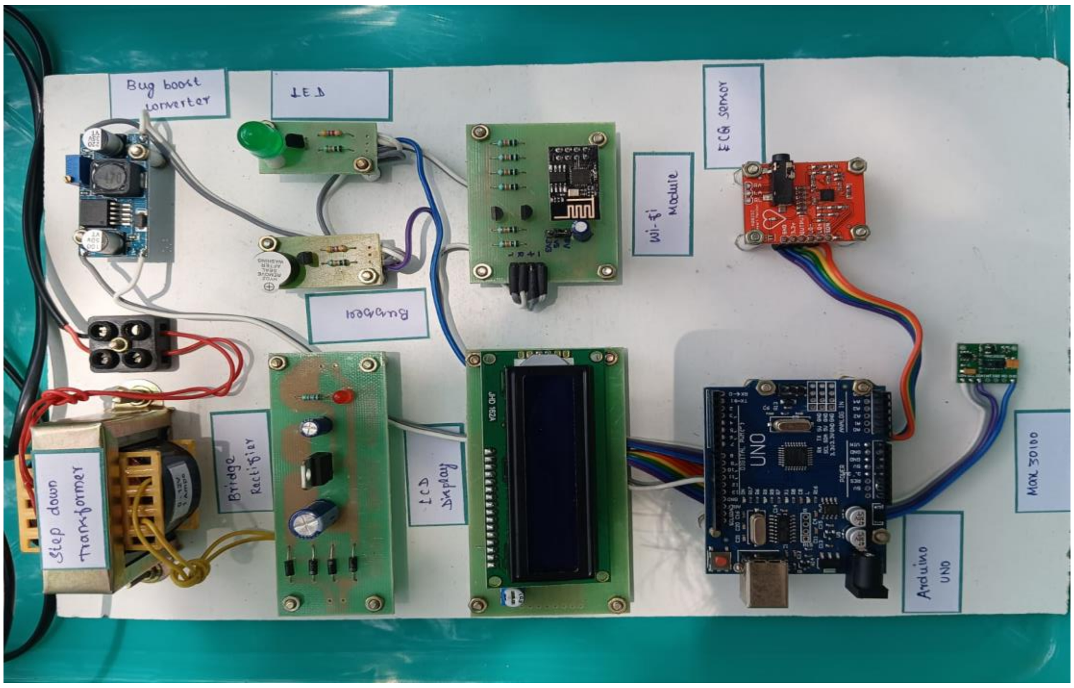


Fig 6.1 Hardware Output

6.3 Simulation output

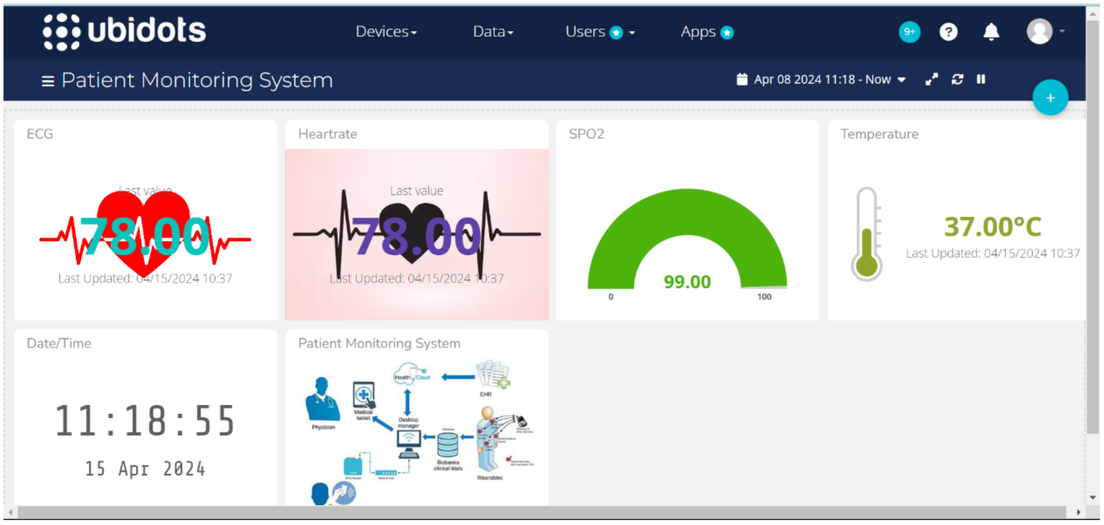
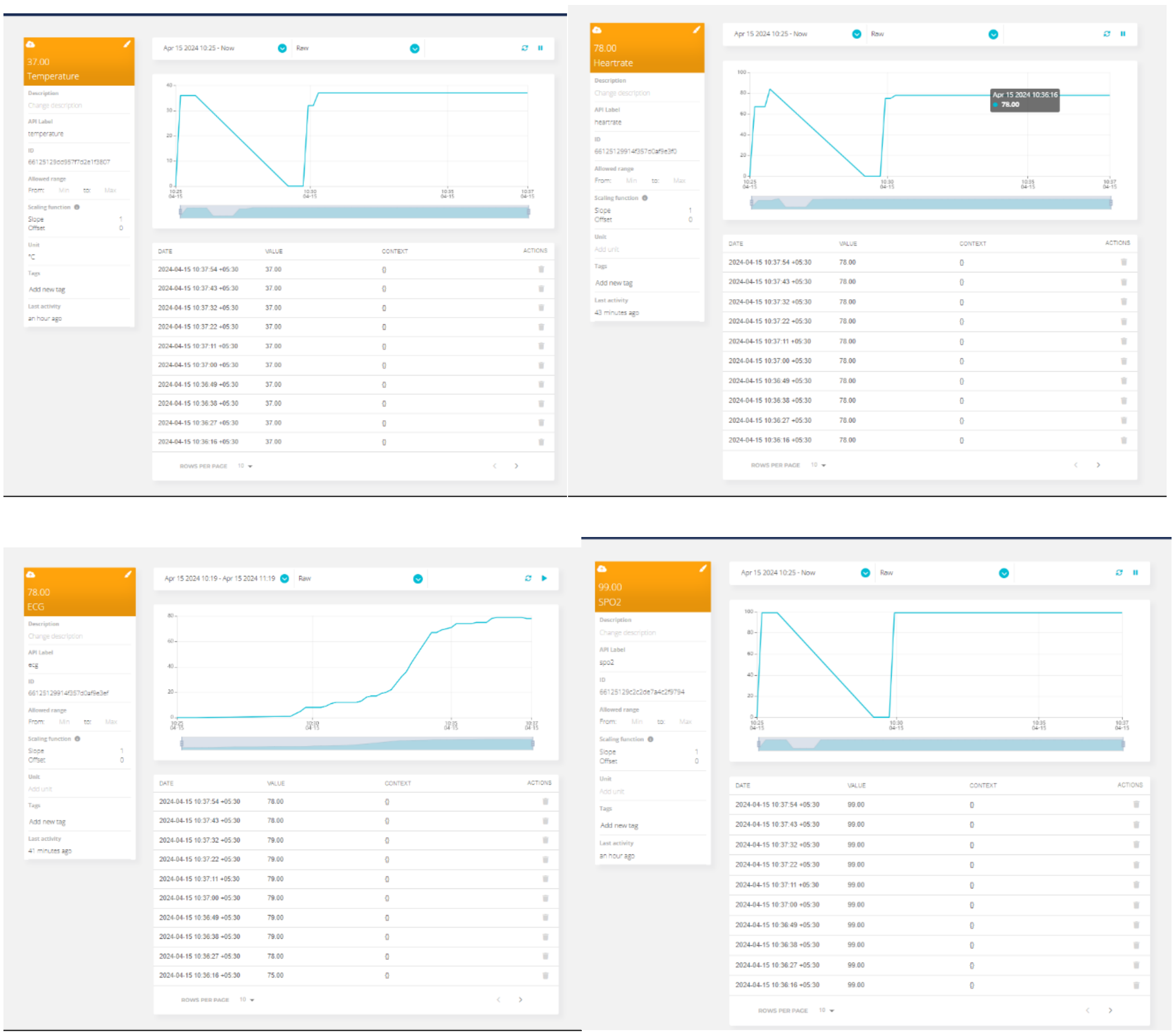


Fig 6.2 Ubidots cloud



37.00
Temperature
Description: Change description
API Label: temperature
ID: 66125129dd957f7d2e1f3807
Allowed range From: Min to: Max
Scaling function: Slope 1, Offset 0
Unit: °C
Tags: Add new tag
Last activity: an hour ago

DATE	VALUE	CONTEXT	ACTIONS
2024-04-15 10:37:54 +05:30	37.00	{}	
2024-04-15 10:37:43 +05:30	37.00	{}	
2024-04-15 10:37:32 +05:30	37.00	{}	
2024-04-15 10:37:22 +05:30	37.00	{}	
2024-04-15 10:37:11 +05:30	37.00	{}	
2024-04-15 10:37:00 +05:30	37.00	{}	
2024-04-15 10:36:49 +05:30	37.00	{}	
2024-04-15 10:36:38 +05:30	37.00	{}	
2024-04-15 10:36:27 +05:30	37.00	{}	
2024-04-15 10:36:16 +05:30	37.00	{}	

78.00
Heartrate
Description: Change description
API Label: heartrate
ID: 661251299914f357d0af9e3f0
Allowed range From: Min to: Max
Scaling function: Slope 1, Offset 0
Unit: Add unit
Tags: Add new tag
Last activity: 43 minutes ago

DATE	VALUE	CONTEXT	ACTIONS
2024-04-15 10:37:54 +05:30	78.00	{}	
2024-04-15 10:37:43 +05:30	78.00	{}	
2024-04-15 10:37:32 +05:30	78.00	{}	
2024-04-15 10:37:22 +05:30	78.00	{}	
2024-04-15 10:37:11 +05:30	78.00	{}	
2024-04-15 10:37:00 +05:30	78.00	{}	
2024-04-15 10:36:49 +05:30	78.00	{}	
2024-04-15 10:36:38 +05:30	78.00	{}	
2024-04-15 10:36:27 +05:30	78.00	{}	
2024-04-15 10:36:16 +05:30	78.00	{}	

78.00
ECG
Description: Change description
API Label: ecg
ID: 661251299914f357d0af9e3ef
Allowed range From: Min to: Max
Scaling function: Slope 1, Offset 0
Unit: Add unit
Tags: Add new tag
Last activity: 41 minutes ago

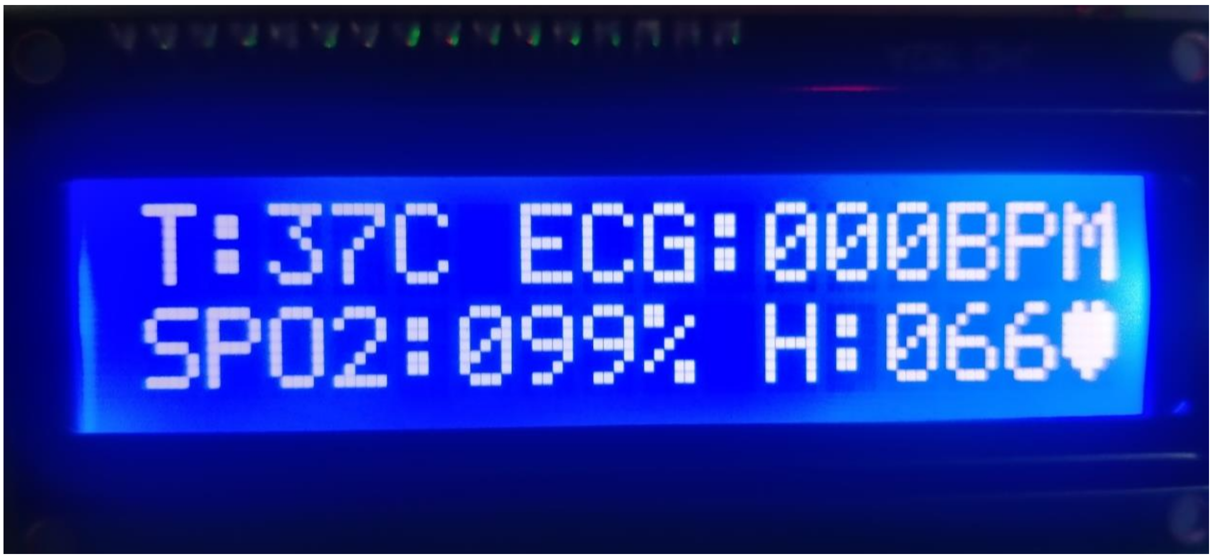
DATE	VALUE	CONTEXT	ACTIONS
2024-04-15 10:37:54 +05:30	78.00	{}	
2024-04-15 10:37:43 +05:30	78.00	{}	
2024-04-15 10:37:32 +05:30	79.00	{}	
2024-04-15 10:37:22 +05:30	79.00	{}	
2024-04-15 10:37:11 +05:30	79.00	{}	
2024-04-15 10:37:00 +05:30	79.00	{}	
2024-04-15 10:36:49 +05:30	79.00	{}	
2024-04-15 10:36:38 +05:30	79.00	{}	
2024-04-15 10:36:27 +05:30	78.00	{}	
2024-04-15 10:36:16 +05:30	75.00	{}	

99.00
SPO2
Description: Change description
API Label: spo2
ID: 66125129c2c2de7a4c2f9794
Allowed range From: Min to: Max
Scaling function: Slope 1, Offset 0
Unit: Add unit
Tags: Add new tag
Last activity: an hour ago

DATE	VALUE	CONTEXT	ACTIONS
2024-04-15 10:37:54 +05:30	99.00	{}	
2024-04-15 10:37:43 +05:30	99.00	{}	
2024-04-15 10:37:32 +05:30	99.00	{}	
2024-04-15 10:37:22 +05:30	99.00	{}	
2024-04-15 10:37:11 +05:30	99.00	{}	
2024-04-15 10:37:00 +05:30	99.00	{}	
2024-04-15 10:36:49 +05:30	99.00	{}	
2024-04-15 10:36:38 +05:30	99.00	{}	
2024-04-15 10:36:27 +05:30	99.00	{}	
2024-04-15 10:36:16 +05:30	99.00	{}	

Fig 6.3 Simulation Output

6.4 IMPLEMENTATION RESULT



```
T:37C ECG:000BPM
SPO2:099% H:066
```

Fig 6.4 Implementation Result

CHAPTER 7

CONCLUSION

Deploying the smart health care monitoring with functional finding, implication for healthcare, strength and limitations.

7.1 CONCLUSION:

Summary of Findings**:** Provide a brief overview of the main findings from the experimentation and analysis conducted throughout the project, highlighting key outcomes related to system functionality, accuracy, reliability, usability, data security, scalability, and integration.

Implications for Healthcare**:** Discuss the implications of the findings for healthcare practice and patient care, emphasizing how the Smart Healthcare Monitoring System addresses critical needs in continuous patient monitoring, remote accessibility, and data-driven decision-making.

Strengths and Limitations: Reflect on the strengths and limitations of the monitoring system identified through experimentation. Highlight its capabilities in providing real-time monitoring, comprehensive data integration, and remote accessibility, while acknowledging any areas for improvement or refinement.

Recommendations for Deployment: Provide recommendations for deploying the Smart Healthcare Monitoring System in real-world healthcare settings, considering factors such as system configuration, training needs, integration with existing infrastructure, and ongoing support requirements.

7.2 FUTURE SCOPE:

Advanced Data Analytics: Incorporate machine learning algorithms and data analytics techniques to analyze sensor data and detect patterns indicative of potential health issues. Implement predictive analytics to forecast health outcomes and personalize treatment plans based on individual patient data.

Enhanced Sensor Integration: Integrate additional sensors and wearables to capture a broader range of health parameters, such as blood pressure, respiratory rate, glucose levels, and activity levels. Explore emerging sensor technologies to enhance measurement accuracy and reliability.

Mobile Application Development: Develop a mobile application for caregivers and patients to access real-time health data, receive alerts, and communicate with healthcare providers. Incorporate features such as medication reminders, appointment scheduling, and teleconsultation capabilities to improve patient engagement and adherence to treatment plans.

Remote Monitoring Enhancements: Enhance remote monitoring capabilities by integrating video conferencing and telemedicine functionalities into the system. Enable virtual consultations between patients and healthcare providers, facilitating timely interventions and reducing the need for in-person visits.

Integration with Electronic Health Records (EHR): Strengthen integration with EHR systems to enable seamless data exchange and continuity of care. Develop standardized data interfaces and interoperability protocols to ensure compatibility with different healthcare IT systems and promote data exchange across healthcare networks.

REFERENCE

1. Chung, Y. F., Chen, Y. C., Hsieh, H. Y., & Wu, C. Y. (2019). A Real-Time Remote Monitoring System for Smart Healthcare. In 2019 IEEE International Conference on Consumer Electronics (ICCE) (pp. 1-2). IEEE.
2. Arora, A., & Verma, S. (2020). IoT-based Smart Healthcare Monitoring System. In 2020 International Conference on Communication, Computing and Electronics Systems (ICCCES) (pp. 562-567). IEEE.
3. Lim, C. P., Loo, C. K., Chow, W. L., Yap, V. V., & Hoo, W. Y. (2018). Development of Smart Healthcare Monitoring System using IoT Technology. In 2018 7th International Conference on Computer and Communication Engineering (ICCCE) (pp. 102-107). IEEE.
4. Dizdaroglu, B., & Coskun, V. (2020). Smart Healthcare Monitoring System Based on IoT and Cloud Computing. In 2020 International Conference on Artificial Intelligence and Data Processing (IDAP) (pp. 1-6). IEEE.
5. Banerjee, M., & Roy, S. (2018). A Review of IoT and Cloud Computing in Healthcare System. In 2018 2nd International Conference on Inventive Systems and Control (ICISC) (pp. 1318-1321). IEEE.
6. Sadashivappa, C., & Narayanaswamy, K. (2019). IoT and Cloud based Smart Health Monitoring System using ESP8266. In 2019 4th IEEE International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT) (pp. 1557-1562). IEEE.
7. Hossain, M. A., & Muhammad, G. (2019). IoT-based healthcare monitoring system: Design, development, and implementation. IEEE Access, 7, 129059-129075.
8. Mekki, K., Al-Makhadmeh, Z., & Al-Jaroodi, J. (2018). A review of IoT smart healthcare applications and challenges. Wireless Networks, 24(2), 393-407.

9. Patil, V. P., & Naik, G. A. (2018). Smart Health Monitoring System using IoT and Cloud Computing. In 2018 International Conference on Computing, Power and Communication Technologies (GUCON) (pp. 612-617). IEEE.
10. Chukwudi, E. M., & Chukwu, J. (2020). A Review of IoT and Cloud Computing Technologies for Smart Healthcare System. In Proceedings of the 4th International Conference on Deep Learning Technologies (pp. 149-157). Springer.