Real-Time ECG Monitoring System

Mrs. P. Tejaswini[1], P. Ajay[2], P. Nithin reddy[3], P. Raghuvardhan reddy[4], P. Sathish Yadav[5]
[1,2,3,4,5]*Tkr College of Engineering and Technology*

I. INTRODUCTION

The human heart is one of the most vital organs in the body, responsible for pumping blood and maintaining proper circulation. Monitoring heart activity is crucial for diagnosing and preventing cardiovascular diseases, which are among the leading causes of death worldwide. Electrocardiography (ECG) is a fundamental diagnostic tool used to measure the electrical activity of the heart over time. It provides essential information about heart rhythm, rate, and overall cardiac health.

An ECG signal is generated due to the electrical impulses that regulate the contraction and relaxation of the heart muscles. These signals are typically represented in the form of waveforms consisting of P, Q, R, S, and T waves. Any irregularity in these waveforms can indicate abnormalities such as arrhythmias, heart block, or other cardiac disorders. Therefore, continuous monitoring of ECG signals plays a significant role in early diagnosis and timely medical intervention.

Traditionally, ECG monitoring systems are confined to hospitals and diagnostic centers, where patients are connected to large and expensive machines. These systems require trained medical personnel for operation and interpretation of results. While effective, such setups are not suitable for continuous monitoring, especially for patients who require long-term observation or those living in remote areas. Frequent hospital visits can also be inconvenient, costly, and time-consuming.

With advancements in embedded systems and communication technologies, there has been a growing interest in developing portable and cost-effective ECG monitoring solutions. The integration of microcontrollers such as Arduino and ESP32, along with compact biomedical sensors, has made it possible to design systems capable of capturing and processing ECG signals in real time. These systems are not only affordable but also highly efficient, making them accessible to a wider population.

In recent years, the Internet of Things (IoT) has revolutionized the healthcare industry by enabling remote monitoring and real-time data sharing. IoT-based healthcare systems allow medical data to be transmitted over the internet to cloud platforms, where it can be stored, analysed, and accessed by healthcare professionals from anywhere in the world. This technology eliminates geographical barriers and ensures continuous monitoring without requiring the patient to be physically present in a hospital.

The Real-Time ECG Monitoring System proposed in this project leverages the capabilities of IoT to provide an efficient and reliable solution for cardiac monitoring. The system uses an ECG sensor, such as the AD8232 module, to detect electrical signals from the heart. These signals are then processed using a microcontroller like ESP32, which is equipped with built-in Wi-Fi capabilities. The processed data is transmitted to an IoT platform such as Blynk, where it is visualized in the form of real-time graphs and numerical values like heart rate (beats per minute).

One of the key features of this system is its ability to provide instant alerts in case of abnormal heart activity. By setting predefined thresholds for heart rate, the system can detect conditions such as tachycardia (high heart rate) and bradycardia (low heart rate) and notify users or healthcare providers immediately. This feature is particularly useful for critical patients who require constant monitoring and quick medical response.

Additionally, the system supports data logging, which allows historical ECG data to be stored and analysed

over time. This helps doctors identify patterns and trends in heart activity, leading to more accurate diagnosis and better treatment planning. The portability of the system ensures that patients can carry it easily and use it in their daily lives without discomfort.

Another important aspect of this project is its cost-effectiveness. Compared to traditional ECG machines, which are expensive and not easily accessible, this system uses low-cost components and open-source platforms. This makes it suitable for use in rural and underdeveloped areas where healthcare resources are limited. By providing an affordable and reliable monitoring solution, the system contributes to improving overall healthcare accessibility.

Furthermore, the system is designed with user-friendliness in mind. The interface provided by the IoT platform is simple and intuitive, allowing users to easily monitor ECG signals and heart rate. Even individuals without medical expertise can understand the basic information displayed by the system. This empowers patients to take a more active role in managing their health.

In conclusion, the Real-Time ECG Monitoring System represents a significant advancement in modern healthcare technology. By combining biomedical sensors, microcontrollers, and IoT, the system provides continuous, remote, and real-time monitoring of heart activity. It addresses the limitations of traditional ECG systems by offering portability, affordability, and ease of use. This project not only enhances patient care but also contributes to the early detection and prevention of cardiovascular diseases, ultimately improving the quality of life.

1.1 Real-Time ECG Monitoring System

The Real-Time ECG Monitoring System is an advanced healthcare solution designed to continuously monitor and display the electrical activity of the human heart. An Electrocardiogram (ECG) is a vital diagnostic tool used to detect heart conditions by recording the timing and strength of electrical signals as they pass through the heart.

In recent years, the integration of embedded systems and IoT technologies has made it possible to develop portable, low-cost, and efficient ECG monitoring systems. This project utilizes sensors and microcontrollers such as the AD8232 ECG Sensor Module and ESP32 Dev Board (or Arduino Uno R3) to capture ECG signals from the human body in real time.

The system works by placing electrodes on the body to detect electrical signals generated by heart activity. These signals are then amplified and filtered by the ECG sensor module before being processed by the microcontroller. The processed data is displayed on a computer or transmitted wirelessly to a mobile application like the Blynk IoT App for remote monitoring.

This project addresses the growing need for continuous health monitoring, especially for patients with cardiovascular diseases. It enables real-time tracking, early detection of abnormalities, and remote access to patient data, making healthcare more accessible and efficient.

Overall, the Real-Time ECG Monitoring System demonstrates how modern technology can be applied to biomedical engineering to create affordable and reliable health monitoring solutions suitable for both clinical and home environments.

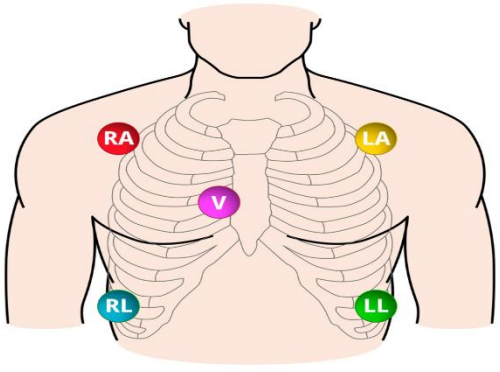


Figure 1: Real-Time ECG Monitoring System

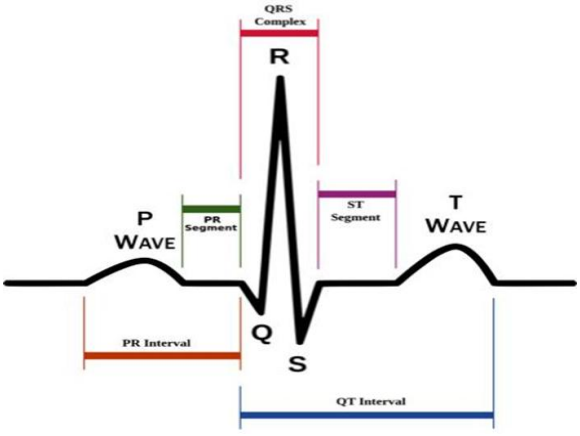


Figure 2: Systematic representation of normal ECG

1.2 Existing System

Traditional ECG monitoring systems are mainly used in hospitals and diagnostic centers. These systems are large, expensive, and require trained medical professionals to operate.

Features of Existing System

- Uses clinical-grade ECG machines
- Provides accurate and detailed reports
- Requires hospital setup and supervision

Limitations

- Not portable
- Expensive equipment
- No continuous remote monitoring
- Patients must visit hospitals frequently
- No real-time alerts for emergencies

1.3 Proposed System

The proposed system aims to develop a portable, low-cost, and real-time ECG monitoring system using embedded systems and Internet of Things (IoT) technology. This system overcomes the limitations of traditional ECG machines by enabling continuous monitoring and remote access to patient data.

The system uses an ECG sensor module (AD8232) to capture the electrical activity of the heart. Electrodes are placed on the patient's body to detect bio-signals, which are then amplified and filtered by the sensor. These analog signals are fed into a microcontroller such as ESP32 or Arduino for further processing.

The microcontroller converts the analog ECG signals into digital data using an Analog-to-Digital Converter (ADC). It then processes the signal to display the ECG waveform and calculate the heart rate in beats per minute (BPM). The processed data is transmitted to an IoT platform for real-time monitoring.

An IoT application (such as Blynk) is used to visualize the ECG waveform and heart rate on a smartphone or computer. This enables doctors and caregivers to monitor the patient remotely from anywhere. Additionally, the system can be programmed to generate alerts when abnormal heart rates are detected, improving emergency response.

1.4 Introduction to embedded system

An embedded system is a special-purpose system in which the computer is completely encapsulated by or dedicated to the device or system it controls. Unlike the general-purpose computer, such as personal computer, an embedded system performs one or a few predefined tasks, usually with very specific requirements. .Since the system is dedicated to specific tasks, design engineers can optimize it, reducing the size and cost of the product. Embedded systems are often mass-produced, benefiting from economies of scale. Personal digital assistants (PDAs) or handheld computers are generally considered embedded devices because of the nature of their hardware design, even though they are more expandable in software terms.

Examples of Embedded Systems

- Automatic teller machines (ATMs)
- Cellular telephones and telephone switches
- Engine controllers and antilock brake controllers for automobiles
- Home automation products, such as thermostats, air conditioners, sprinklers, and security monitoring systems
- Handheld calculators
- Handheld computers
- Household appliances, including microwave ovens, washing machines, television sets, DVD players and recorders
- Medical equipment
- Personal digital assistant
- Video game consoles
- Computer peripherals such as routers and printers
- Industrial controllers for remote machine operation.

1.5 Characteristics

Embedded systems are designed to do some specific task, rather than be a general-purpose computer for multiple tasks. Some also have real-time performance constraints that must be met, for reason such as safety and usability; others may have low or no performance requirements, allowing the system hardware to be simplified to reduce costs. An embedded system is not always a separate block - very often it is physically built-in to the device it is controlling.

The software written for embedded systems is often called firmware, and is stored in read-only memory or Flash memory chips rather than a disk drive. It often

runs with limited computer hardware resources: small or no keyboard, screen, and little memory.

1.6 Microcontroller

A microcontroller is a single-chip VLSI unit (also called microcomputer) which, although having limited computational capabilities, possesses enhanced input/output capability and a number of on-chip functional units.

II. LITERATURE SURVEY

The development of ECG monitoring systems has significantly progressed with advancements in embedded systems, signal processing, and Internet of Things (IoT) technologies. Researchers have focused on improving portability, real-time monitoring, and accuracy while reducing system cost and complexity.

A study by researchers in 2026 proposed an IoT-based ECG monitoring system using ESP32 and AD8232 sensors for real-time cardiac data acquisition and transmission. The system enabled continuous monitoring and cloud-based data storage, allowing healthcare professionals to access patient data remotely. The study also analysed heart rate variability (HRV) and demonstrated the effectiveness of the system in detecting early cardiac abnormalities.

Another research work (2022) developed a low-cost ECG monitoring system using the AD8232 sensor and ESP32 microcontroller. The system focused on signal acquisition and filtering techniques to reduce noise and improve waveform accuracy. Data was transmitted to a cloud platform for remote monitoring, making it suitable for rural and home-based healthcare applications.

In a separate study, researchers introduced a real-time ECG monitoring and arrhythmia detection system using advanced signal processing techniques such as the Pan-Tompkins algorithm. The system accurately detected QRS complexes and calculated heart rate, enabling identification of abnormal conditions like tachycardia and bradycardia. This work emphasized the importance of efficient algorithms in improving ECG signal analysis.

Another significant contribution involves the use of machine learning and deep learning techniques for ECG signal classification. A study implemented Convolutional Neural Networks (CNN) to automatically classify ECG signals and detect arrhythmias with high accuracy. This approach reduces the dependency on manual diagnosis and enhances the reliability of automated healthcare systems.

Additionally, research has explored wearable ECG monitoring devices integrated with IoT for continuous health tracking. These systems are designed to be compact, energy-efficient, and user-friendly, allowing patients to monitor their heart activity during daily routines.

III. THEORY

3.1 Block Diagram

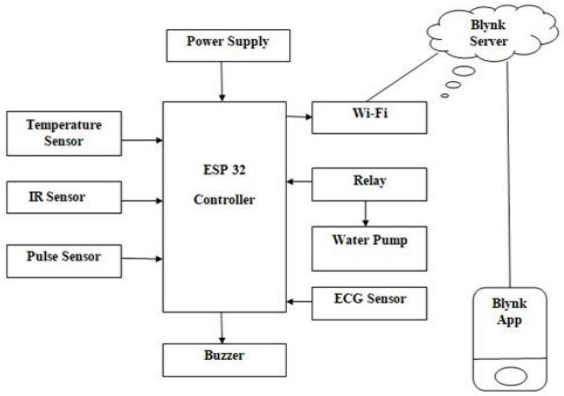


Figure 3: Block diagram of ECG

1. Power Supply

This block provides the required electrical power to all components in the system. It ensures stable voltage for the ESP32, sensors, relay, and other modules. Typically, a 5V or 3.3V regulated supply is used.

2. ESP32 Controller

This is the main processing unit of the system. It performs multiple tasks:

- Reads data from all sensors (ECG, pulse, temperature, IR)
- Processes signals (like calculating BPM)
- Controls output devices (buzzer, relay)
- Sends data to the cloud via Wi-Fi

3. Temperature Sensor

This sensor measures the body temperature of the patient. The ESP32 reads this data and sends it to the IoT platform for monitoring.

4. IR Sensor
The IR (Infrared) sensor is used to detect presence or object movement. In healthcare systems, it can be used for:

- Patient detection
- Monitoring hand movement or proximity

5. Pulse Sensor
This sensor measures the heart rate (BPM) by detecting blood flow changes. It works using light absorption and gives digital/analog signals to ESP32.

6. ECG Sensor
The ECG sensor (like AD8232) captures the electrical activity of the heart. It provides waveform signals that are processed by the ESP32 to display ECG graphs and analyse heart conditions.

7. Wi-Fi Module (Built-in ESP32)
ESP32 has inbuilt Wi-Fi, which is used to:

- Send sensor data to the cloud (Blynk Server)
- Enable remote monitoring through mobile apps

8. Blynk Server
This is the cloud platform where all data is stored and processed. It acts as a bridge between the ESP32 and the mobile application.

9. Blynk App
The Blynk mobile app is used to:

- Display ECG waveform
- Show heart rate, temperature, etc.
- Send alerts/notifications to users

10. Buzzer
The buzzer is used as an alert system. It turns ON when:

- Abnormal heart rate is detected
- Emergency condition occurs

S.No	Component	Pin Name	ESP32 Pin	Description
1	ECG Sensor (AD8232)	OUTPUT	GPIO34	Analog ECG signal input to ESP32
		LO+	GPIO32	Lead-off detection (positive)
		LO-	GPIO33	Lead-off detection (negative)
		VCC	3.3V	Power supply
		GND	GND	Ground
2	Pulse Sensor	Signal	GPIO35	Heart rate signal input
		VCC	3.3V	Power supply
		GND	GND	Ground
3	Temperature Sensor	Data	GPIO4	Temperature data input
	(e.g., DHT11)	VCC	3.3V	Power supply
		GND	GND	Ground
4	IR Sensor	OUT	GPIO5	Object detection signal
		VCC	3.3V	Power supply
		GND	GND	Ground
5	Relay Module	IN	GPIO18	Control signal from ESP32
		VCC	5V	Power supply
		GND	GND	Ground
6	Buzzer	Positive	GPIO19	Alert output
		Negative	GND	Ground

S.No	Component	Pin Name	ESP32 Pin	Description
7	Wi-Fi (ESP32)	Internal		Used for IoT communication
8	Power Supply	VIN / 3.3V	VIN	Main power input to ESP32
		GND	GND	Ground

3.2 CIRCUIT DIAGRAM

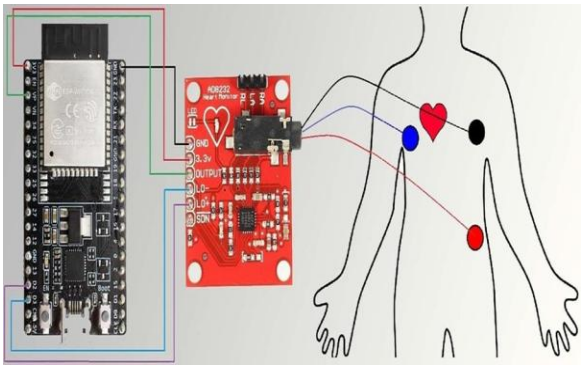


Figure 4: Circuit Diagram of ECG

The circuit diagram of the proposed ECG and health monitoring system illustrates the interconnection of various sensors and components with the ESP32 microcontroller. The ECG sensor (AD8232) is connected to the analog input pins of the ESP32 to capture the electrical activity of the heart, while the pulse sensor is used to measure heart rate and is also connected to an analog input pin. A temperature sensor such as DHT11 is interfaced with a digital GPIO pin to monitor body temperature, and an IR sensor is connected to detect the presence or movement of the patient. A buzzer is attached to a digital output pin to provide alerts during abnormal conditions, and a relay module is used to control external devices like a water pump. The ESP32, with its built-in Wi-Fi capability, transmits all the collected and processed data to the Blynk cloud server, which is then displayed on the Blynk mobile application for real-time monitoring. All components share a common ground, and appropriate power supply connections (3.3V and 5V) are provided to ensure stable operation. This circuit enables efficient data acquisition, processing, and remote monitoring of patient health parameters.

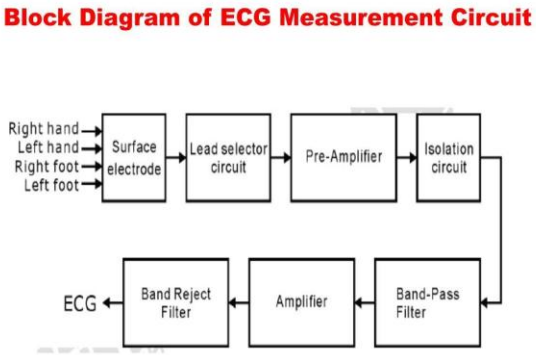


Figure 5: Block diagram of ECG measurement circuit

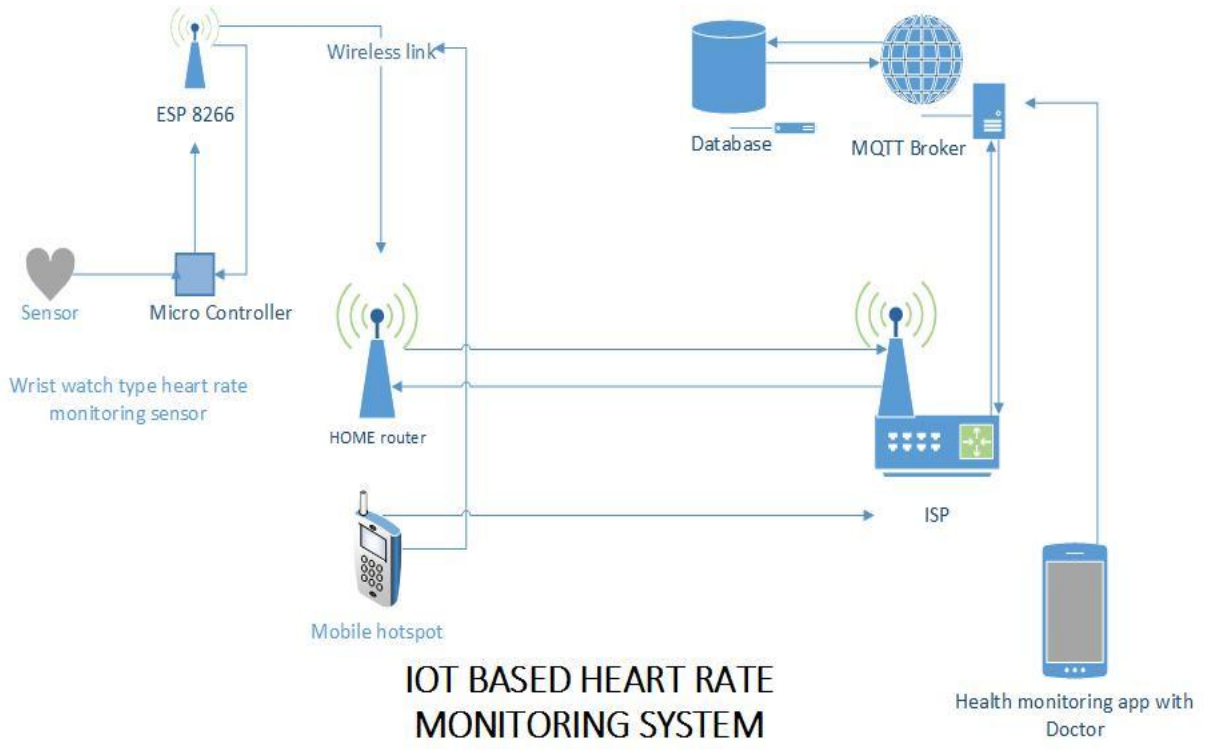


Figure 6: IOT Based Heart Rate Monitoring System

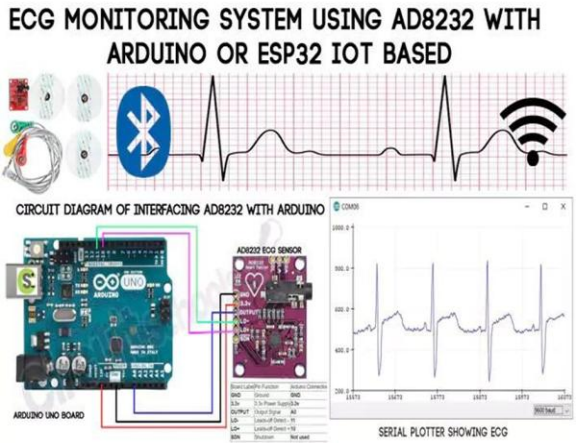


Figure 7: ECG Monitoring system using Arduino

3.3 Components Required

3.3.1 USB-CABLE

Introduction

This article provides information about the physical aspects of Universal Serial Bus, USB: connectors, cabling, and power. The initial versions of the USB standard specified connectors that were easy to use and that would have acceptable life spans; revisions of the standard added smaller connectors useful for compact portable devices. Higher-speed development of the USB standard gave rise to another family of connectors to permit additional data paths. All versions of USB specify cable properties; version 3.X cables include additional data paths. The USB standard included power supply to peripheral devices; modern versions of the standard extend the power delivery limits for battery charging and devices requiring up to 100 watts. USB has been selected as the standard charging format for many mobile phones, reducing the proliferation of proprietary chargers.

Figure 8: USB-CABLE

History

A group of seven companies began the development of USB in 1994: Compaq, DEC, IBM, Intel,

Microsoft, NEC, and Nortel. The goal was to make it fundamentally easier to connect external devices to PCs by replacing the multitude of connectors at the back of PCs, addressing the usability issues of existing interfaces, and simplifying software configuration of all devices connected to USB, as well as permitting greater data rates for external devices. Ajay Bhatt and his team worked on the standard at Intel; the first integrated circuits supporting USB were produced by Intel in 1995.Joseph Decuir, an American fellow of the Institute of Electrical and Electronics Engineers (IEEE) and one of the designers of the early Atari 8-bit game and computer systems (Atari VCS, Atari 400/800), as well as the Commodore Amiga, credits his work on Atari SIO, the Atari 8-bit computer's communication implementation as the basis of the USB standard, which he also helped design and on which he holds patents.

3.3.2 Jumper Wires

Introduction

A jump wire (also known as jumper, jumper wire, jumper cable, DuPont wire, or DuPont cable – named for one manufacturer of them) is an electrical wire, or group of them in a cable, with a connector or pin at each end (or sometimes without them – simply "tinned"), which is normally used to interconnect the components of a breadboard or other prototype or test circuit, internally or with other equipment or components, without soldering.

Figure 9: Jumper Wires

Different types of jumper wires

Some have the same type of electrical connector at both ends, while others have different connectors. Some common connectors are:

Solid tips – are used to connect on/with a breadboard or female header connector. The arrangement of the elements and ease of insertion on a breadboard allows increasing the mounting density of both components and jump wires without fear of short-circuits. The jump wires vary in size and color to distinguish the different working signals.

Crocodile-clips – are used, among other applications, to temporarily bridge sensors, buttons and other elements of prototypes with components or equipment that have arbitrary connectors, wires, screw terminals

Banana connectors – are commonly used on test equipment for DC and low-frequency AC signal

Registered jack (RJNN) – are commonly used in telephone (RJ11) and computer networking (RJ45).

RCA connectors – are often used for audio, low-resolution composite video signals, or other low-frequency applications requiring a shielded cable. 28

RF connectors – are used to carry radio frequency signals between circuits, test equipment, and antennas.

RF jumper cables - Jumper cables are smaller and more bendable corrugated cable which is used to connect antennas and other components to network cabling. Jumpers are also used in base stations to connect antennas to radio units. Usually, the most bendable jumper cable diameter is 1/2".

IMPLEMENTATION

The implementation of the proposed ECG and health monitoring system involves the

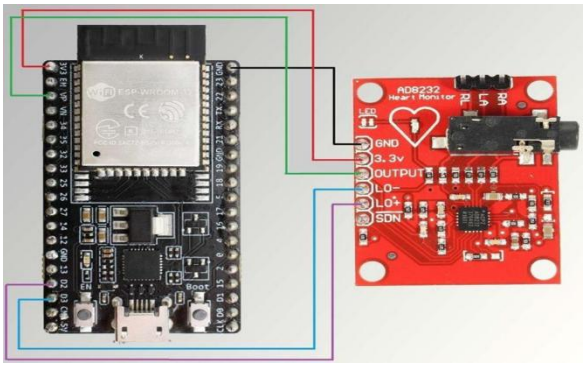


Figure 10: Schematic Pin Diagram

integration of hardware components and software programming to achieve real-time data acquisition, processing, and remote monitoring. The system is built using an ESP32 microcontroller as the central unit, interfaced with multiple sensors including the AD8232 ECG sensor, pulse sensor, temperature sensor, and IR sensor.

Initially, all hardware components are connected according to the circuit diagram. The ECG sensor is attached to the patient using electrodes to capture heart signals, while the pulse sensor and temperature sensor are used to measure heart rate and body temperature respectively. The IR sensor is used for detecting the presence of the patient. A buzzer and relay module are also connected to provide alert and control functionalities.

The ESP32 is programmed using the Arduino IDE. The code is designed to read analog and digital inputs from the sensors, process the ECG signal, and calculate the heart rate (BPM) by detecting peaks in the waveform. Signal filtering techniques may be applied to reduce noise and improve accuracy. The microcontroller also checks for abnormal conditions such as high or low heart rate.

For IoT functionality, the ESP32 uses its built-in Wi-Fi to connect to the Blynk cloud server. Sensor data including ECG waveform, BPM, and temperature is transmitted in real time to the Blynk mobile application. The app displays this data using widgets such as graphs and gauges, allowing users and doctors to monitor the patient remotely.

During abnormal conditions, such as irregular heart rate or sensor disconnection, the system activates the buzzer and can also send notifications through the Blynk app. The relay module can be triggered based on predefined conditions to control external devices if required.

The system is tested under different conditions to ensure accurate data acquisition and reliable performance. Proper calibration of sensors and stable power supply are maintained to minimize errors and noise.

IV. SOFTWARE DESCRIPTION

4.1 SOFTWARE ARDUNIO IDE:

Introduction:

The Arduino Integrated Development Environment - or Arduino Software (IDE) – contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them.

Writing sketches:

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension. ino. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom right-hand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

Verify

Checks your code for errors compiling it. Up load Compiles your code and uploads it to the configured board.

See uploading below for details.

Open

Presents a menu of all the sketches in your sketchbook. Clicking one will open within the current window overwriting its content.

Note: due to a bug in Java, this menu doesn't scroll; if you need to open a sketch late in the list, use the File | Sketchbook menu instead

Additional commands are found within the five menus: File, Edit, Sketch, Tools and Help. The menus are context sensitive, which means only those items relevant to the work currently being carried out are available.

File:

- New: Creates a new instance of the editor, with the bare minimum structure of a sketch already in place.

- Open: Allows loading a sketch file browsing through the computer drives and folders.
- Open: Recent provides a short list of the most recent sketches, ready to be opened.
- Sketchbook: Shows the current sketches within the sketchbook folder structure; clicking on any name opens the corresponding sketch in a new editor instance.
- Examples: Any example provided by the Arduino Software (IDE) or library shows up in this menu item. All the examples are structured in a tree that allows easy access by topic or library.
- Close: Closes the instance of the Arduino Software from which it is clicked.
- Save: Saves the sketch with the current name. If the file hasn't been named before, a name
- and other operating systems char maps.
- Serial Monitor: Opens the serial monitor window and initiates the exchange of data with any connected board on the currently selected Port. This usually resets the board, if the board supports Reset over serial port opening
- Board: Select the board that you're using. See below for descriptions of the various boards.
- Port: This menu contains all the serial devices (real or virtual) on your machine. It should automatically refresh every time you open the top-level tools menu.
- Programmer: For selecting a hardware programmer when programming a board or chip and not using the on-board USB-serial connection. Normally you won't need this, but if you're burning a boot loader to a new microcontroller, you will use this.
- Burn Boot loader: The items in this menu allow you to burn a boot loader onto the microcontroller on an Arduino board. This is not required for normal use of an Arduino or Genuine board but is useful if you purchase a new AT mega microcontroller (which normally comes without a boot loader). Ensure that you've selected the correct board from the Boards menu before burning the boot loader on the target board. This command also set the right fuses

6.2 HELP

Here you find easy access to a number of documents that come with the Arduino Software (IDE). You have access to Getting Started, Reference, this guide to the IDE and other documents locally, without an internet connection.

Preferences:

Some preferences can be set in the preferences dialog (found under the Arduino menu on the Mac, or File on Windows and Linux). The rest can be found in the preferences file, whose location is shown in the preference di

4.2 LANGUAGE SUPPORT

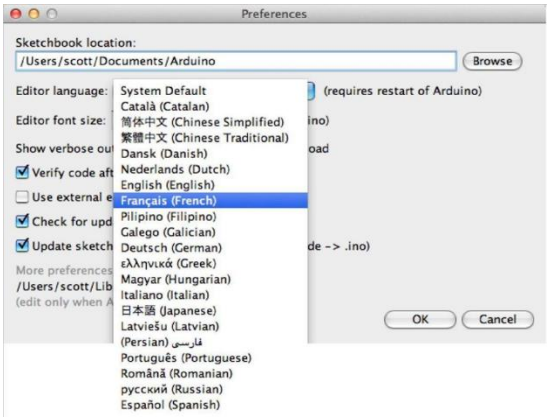


Figure 11: Language support

o Since version 1.0.1, the Arduino Software (IDE) has been translated into 30+ different languages. By default, the IDE loads in the language selected by your operating system. (Note: on Windows and possibly Linux, this is determined by the

locale setting which controls currency and date formats, not by the language the operating system is displayed in.)

o If you would like to change the language manually, start the Arduino Software (IDE)and open the Preferences window. Next to the Editor Language there is a dropdown menu of currently supported languages. Select your preferred language from the

o menu, and restart the software to use the selected language. If your operating system language is not supported, the Arduino Software (IDE) will default to English.

o You can return the software to its default setting of selecting its language based

o On your operating system by selecting System Default from the Editor Language

dropdown. This setting will take effect when you restart the Arduino Software (IDE).

o Similarly, after changing your operating system's settings, you must restart the Arduino Software (IDE) to update it to the new default language.

6.3 Boards:

o The board selection has two effects: it sets the parameters (e.g. CPU speed and baud rate) used when compiling and uploading sketches; and sets and the file and fuse settings used by the burn boot loader command. Some of the board definitions differ only in the latter, so even if you've been uploading successfully with a particular selection, you'll want to check it before burning the boot loader. You can find a comparison table between the various boards here.

o Arduino Software (IDE) includes the built in support for the boards in the following list, all based on the AVR Core. The Boards Manager included in the standard installation allows to add support for the growing number of new boards based on different cores like Arduino Due, Arduino Zero, Edison and Galileo and so on.

o Arduino/Genuino Uno

o An ATmega328P running at 16 MHz with auto-reset, 6 Analog In, 14 Digital I/O and 6PWM.

o Arduino Decimal or Demeaned w/ ATmega168An ATmega168 running at 16 MHz with auto-reset.

ARDUINO UNO INSTALLATION:

o In these we will get know of the process of installation of Arduino IDE and connecting Arduino Uno to Arduino IDE.

o Step 1 - First we must have our Arduino board (we can choose our favorite board) and a USB cable. In case we use Adriana UNO, Arduino Delanie, Nano, Arduino Mega 2660, or Decimals, we will need a standard USB cable (A plug to B plug), t In case we use Arduino Nano, we will need an A to Mini-B cable...

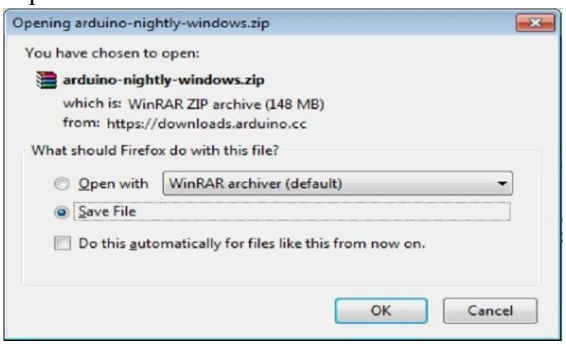
o Step 2 - Download Arduino IDE Software. We can get different versions of Arduino IDE from the Download page on the Arduino Official website. We must select were software, which is compatible with operating system (Windows, IOS, or Linux).

o After wear file download is complete, unzip the file. o Step 3 − Power up our board.

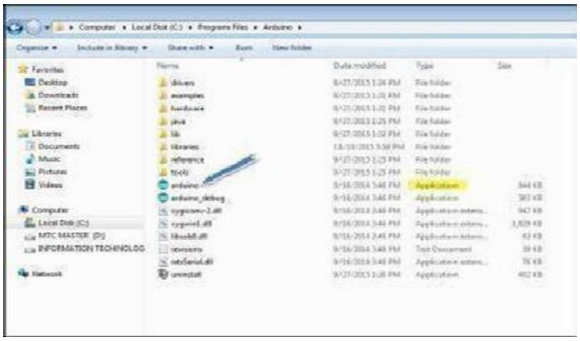
The Arduino Uno, Mega, Delanie and Arduino Nano automatically draw power from either, the USB connection to the computer or an external power supply. If we are using an Arduino Decimal, we have to make sure that the board is configured to draw power from the USB connection. The power source is selected with a jumper, a small piece of plastic that fits onto two of the three pins between the USB and power jacks.

Check that it is on the two pins closest to the USB port.Connect the Arduino board to were computer using the USB cable. The green power LED(labelled PWR) should glow.

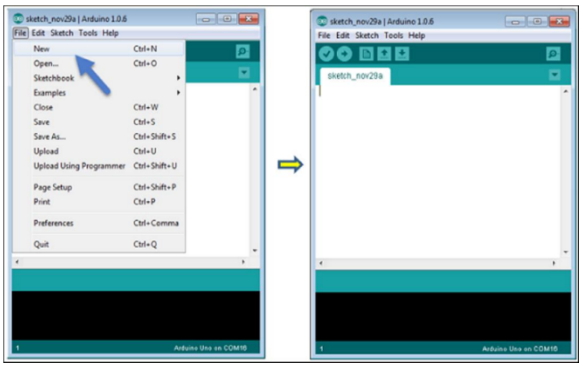
Step 4 − Launch Arduino IDE.



After our Arduino IDE software is downloaded, we need to unzip the folder. Inside the folder, we can find the application icon with an infinity label (application.exe).Double click the icon to start the IDE.



Step 5 − Open our first project.

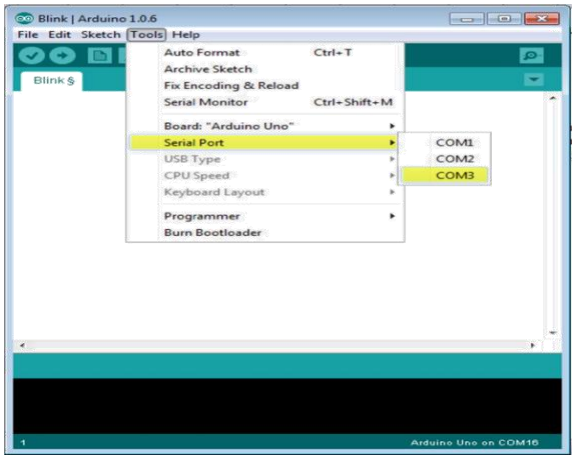


Once the software starts, we have two options

* Create a new project
* Open an existing project example.

To create a new project, select File → new.
To open an existing project example, select File → Example → Basics → Blink.
Here, we are selecting just one of the examples with the name Blink. It turns the LED onand off with some time delay. We can select any other example from the list.

Step 6 − Select our Arduino board.

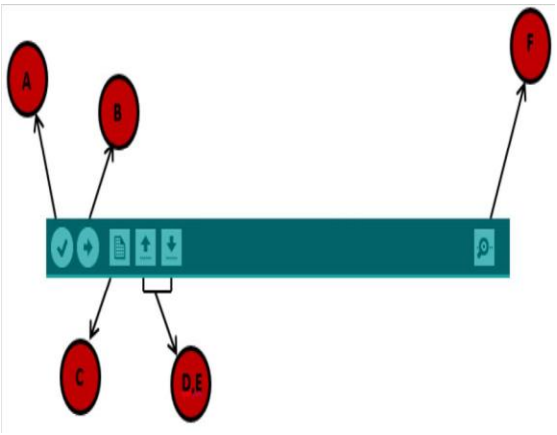


To avoid any error while uploading wear program to the board, we must select the correct Arduino board name, which matches with the board connected to were computer.

Go to Tools → Board and select wear board.
Here, we have selected Arduino Uno board according to our tutorial, but we must select the name matching the board that we are using.

o Step 7 − Select were serial port.
Select the serial device of the Arduino board. Go to Tools → Serial Port menu. Thisis likely to be COM3 or higher (COM1 and COM2 are usually reserved for hardware serial ports). To find out, we can disconnect were Arduino board and re-open the menu, the entry that disappears should be of the Arduino board. Reconnect the board and select that serial port.

o Step 8 − Upload the program to were board.
Before explaining how we can upload our program to the board, we must demonstratethe function of each symbol appearing in the Arduino IDE toolbar.



A − Used to check if there is any compilation error.
B − Used to upload a program to the Arduino board.
C − Shortcut used to create a new sketch.
D − Used to directly open one of the example sketches.
E − Used to save were sketch.

seconds; we will see the RX and TX LEDs on the board, flashing. If the upload is successful, the message
"Done uploading" will appear in the status bar.
Note − If we have an Arduino Mini, NG, or other board, we need to press the reset button physically on the board, immediately before clicking the upload button on the Arduino Software.

 STEPS TO WRITE A PROGRAM:

STEP1
Arduino microcontrollers come in a variety of types. The most common is the Arduino UNO, but there are specialized variations. Before you begin building, do a little research to figure out which version will be the most appropriate for your project.

STEP2
To begin, you'll need to install the Arduino Programmer, aka the integrated development environment (IDE).

STEP3
Connect your Arduino to the USB port of your computer. This may require a specific USB cable.
Every Arduino has a different virtual serial-port address, so you'll need to reconfigure the port if you're using different Arduinos.

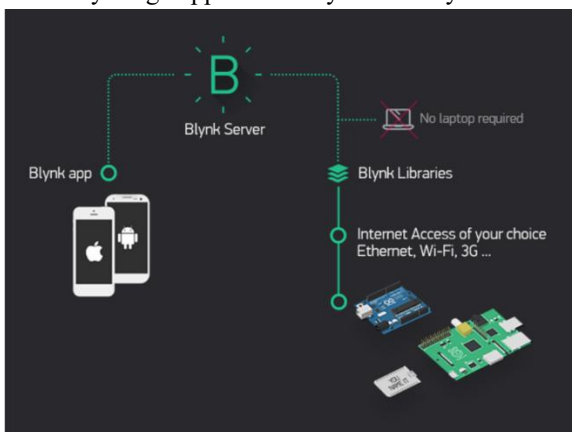
STEP4
Set the board type and the serial port in the Arduino Programmer.

STEP5
Test the microcontroller by using one of the preloaded programs, called sketches, in the Arduino Programmer. Open one of the example sketches, and press the upload button to load it. The Arduino should begin responding to the program: If you've set it to blink an LED light, for example, the light should start blinking.

BLYNK APP
Blynk was designed for the Internet of Things. It can control hardware remotely, it can display sensor data, it can store data, visualize it and do many other cool things.
There are three major components in the platform:

- Blynk App - allows to you create amazing interfaces for your projects using various widgets we provide.
- Blynk Server - responsible for all the communications between the smartphone and hardware. You can use our Blynk Cloud or run your private Blynk server locally. It's open-source, could easily handle thousands of devices and can even be launched on a Raspberry Pi.
- Blynk Libraries - for all the popular hardware platforms - enable communication with the server and process all the incoming and outcoming commands.

Now imagine: every time you press a Button in the Blynk app, the message travels to space the Blynk Cloud, where it magically finds its way to your hardware. It works the same in the opposite direction and everything happens in a blynk of an eye.



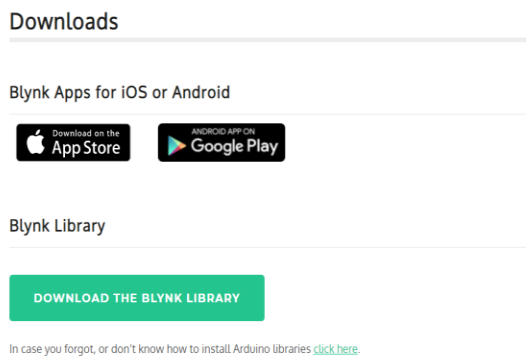
- Similar API & UI for all supported hardware & devices
- Connection to the cloud using:
- Wi-Fi
- Bluetooth and BLE
- Ethernet
- USB (Serial)
- GSM
- Set of easy-to-use Widgets
- Direct pin manipulation with no code writing
- Easy to integrate and add new functionality using virtual pins
- History data monitoring via Super Chart widget

Device-to-Device communication using Bridge Widget

- Sending emails, tweets, push notifications, etc.
- New features are constantly added!

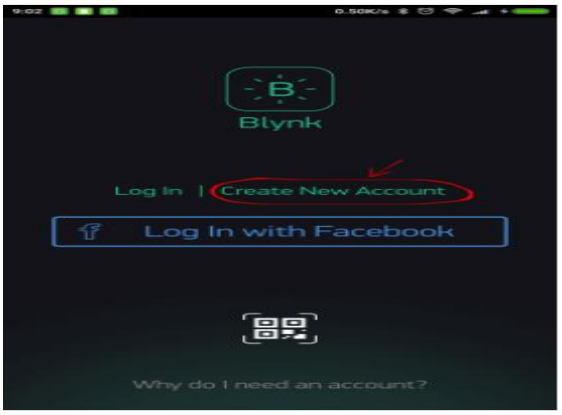
You can find example sketches covering basic Blynk Features. They are included in the library. All the sketches are designed to be easily combined with each other.

The Blynk App is a well-designed interface builder. It works on both iOS and Android, so no holy wars here,



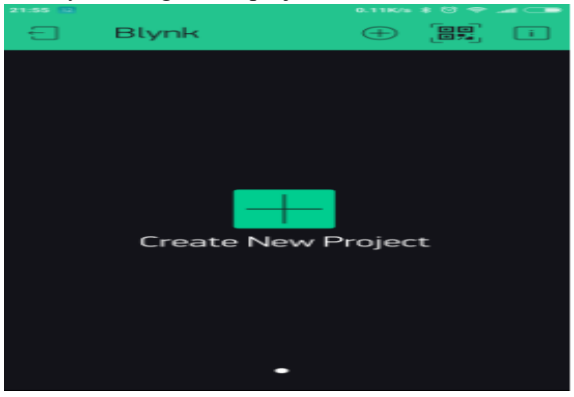
Create a Blynk Account

After you download the Blynk App, you'll need to create a New Blynk account. This account is separate from the accounts used for the Blynk Forums; in case you already have one.
We recommend using a real email address because it will simplify things later.



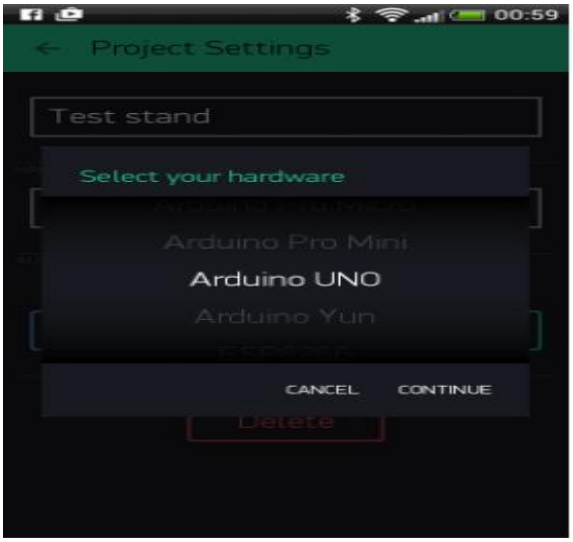
Create a New Project

After you've successfully logged into your account, start by creating a new project.



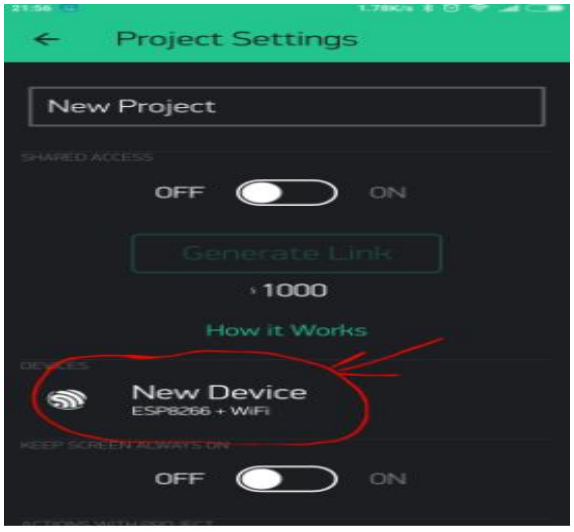
Choose Your Hardware

Select the hardware model you will use. Check out the list of supported hardware



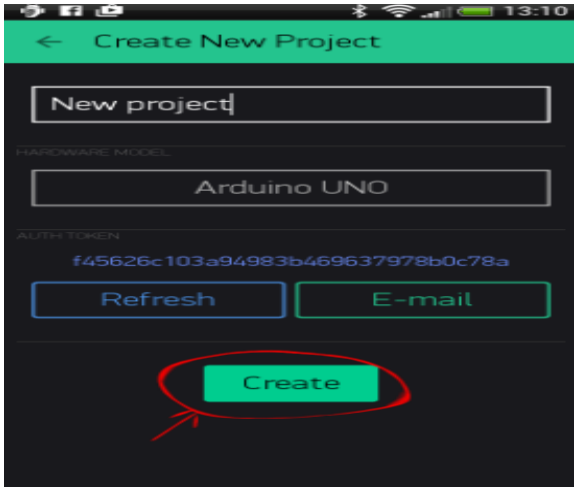
Auth Token

Auth Token is a unique identifier which is needed to connect your hardware to your smartphone. Every new project you create will have its own Auth Token. You'll get Auth Token automatically on your email after project creation. You can also copy it manually. Click on devices section and selected required device:



NOTE: Don't share your Auth Token with anyone, unless you want someone to have access to your hardware.
It's very convenient to send it over e-mail. Press the e-mail button and the token will be sent to the e-mail

address you used for registration. You can also tap on the Token line and it will be copied to the clipboard. Now press the “Create” button.

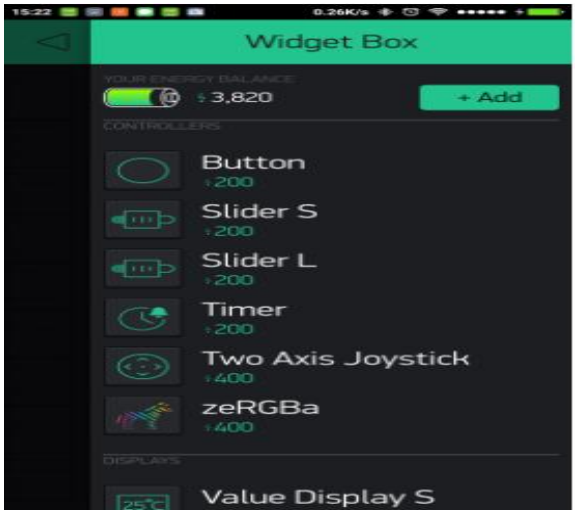


5. Add a Widget

Your project canvas is empty, let’s add a button to control our LED.

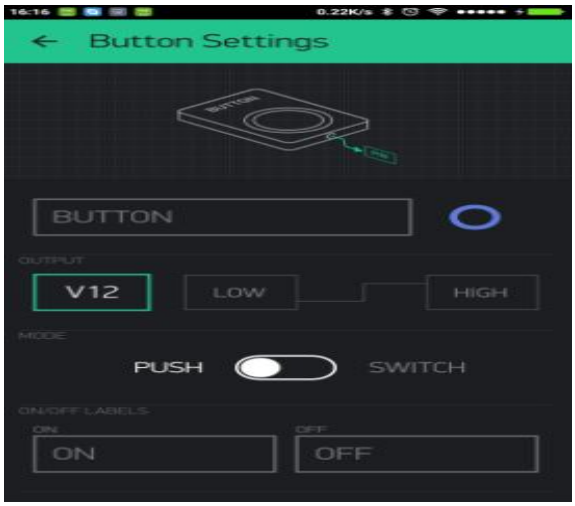
Tap anywhere on the canvas to open the widget box. All the available widgets are located here. Now pick a button.

Widget Box



Drag-n-Drop - Tap and hold the Widget to drag it to the new position.

Widget Settings - Each Widget has it’s own settings. Tap on the widget to get to them.



The most important parameter to set is PIN . The list of pins reflects physical pins defined by your hardware. If your LED is connected to Digital Pin 8 - then select D8 (D - stands for Digital).

Run The Project

When you are done with the Settings - press the PLAY button. This will switch you from EDIT mode to PLAY mode where you can interact with the hardware. While in PLAY mode, you won’t be able to drag or set up new widgets, press STOP and get back to EDIT mode.

You will get a message saying “Arduino UNO is offline”. We’ll deal with that in the next section

4.3 Data Visualization

The processed data, including the ECG waveform and BPM, is transmitted from the Arduino to a computer using a serial connection. On the computer side, a Processing sketch is employed to receive and visualize the data in real time. The Processing sketch performs the following tasks:

• Reads incoming data from the Arduino.

• Plots the ECG waveform dynamically using the line() function

• Displays the BPM periodically on the screen

1. ECG Signal Representation

The ECG signal is visualized as a waveform graph that shows voltage (y-axis) versus time (x-axis). The waveform consists of key components:

- P wave → Atrial contraction
- QRS complex → Ventricular contraction
- T wave → Ventricular relaxation

This graphical representation helps in identifying heart rhythm and abnormalities.

2. Real-Time Data Display

In a real-time system, ECG data is continuously collected from sensors and displayed instantly on a screen. The waveform moves from left to right like a live animation, showing the current heartbeat.

This allows:

- Instant monitoring
- Quick detection of irregularities
- Continuous patient observation

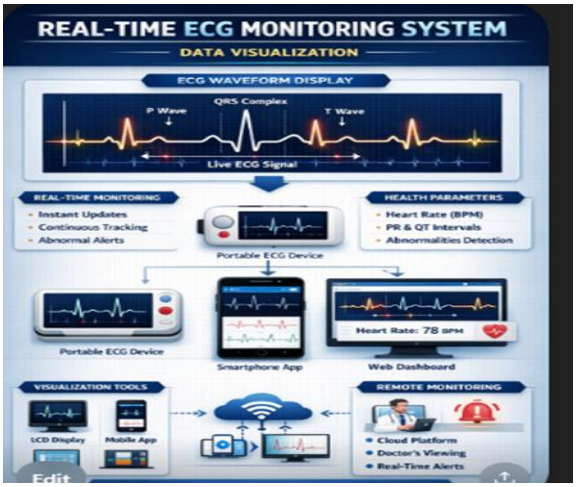


Figure 11: Data Visualization

3. Visualization Tools and Platforms

Data visualization can be implemented using different tools:

- LCD/LED Displays → For basic waveform display
- Computer Software (MATLAB, Python) → For advanced graphs
- Mobile Applications → For remote monitoring
- Web Dashboards (IoT-based) → For cloud-based visualization

4. Parameters Displayed

Apart from waveform, the system also displays important health parameters:

- Heart Rate (BPM – Beats Per Minute)
- Signal amplitude
- Time intervals (PR, QT intervals)
- Alerts for abnormal patterns

5. Graphical Features

Modern ECG visualization includes:

- Zoom in/out for detailed analysis
- Grid lines for accurate measurement
- Color coding (normal vs abnormal signals)
- Data recording and playback

6. Role in Diagnosis

Visualization helps doctors to easily identify:

- Arrhythmia (irregular heartbeat)
- Tachycardia (fast heart rate)
- Bradycardia (slow heart rate)
- Heart block or abnormalities

Without visualization, interpreting ECG data would be very difficult.

7. IoT-Based Visualization

In advanced systems, ECG data is transmitted using IoT to cloud platforms. Doctors can monitor patients remotely through:

- Mobile apps
- Web dashboards
- Real-time alerts

V. WORKING PRINCIPLE

5.1 Architecture performance

The plan created works under the reasoning of customer/server. It demonstrates the circulation of outline. Here are the highlights of the components of the server and customer is spoken to. ϖ Server: The server comprises of 3 essential parts:

- Detector setting: is the part responsible for getting setting information. The information is caught through the appropriate responses given by the net administrations that manufacture correspondence out there between the server, the data patients, and assortments of exercises, disorder and specialists.
- Reasoning motor: It is accountable for making deductions in light of the talk information gave by the finder setting. Ontology used to manufacture suggestions as practicing schedules to patients
- The server because of solicitations from net administration is expended normally by the portable customer. ϖ Customer: It is a framework created on android 4.4, that comprises of a primary

The architecture and performance of a Real-Time ECG Monitoring System are designed to ensure accurate, continuous, and efficient monitoring of heart activity. The system architecture typically consists of multiple layers, including data acquisition, signal processing, communication, and visualization. In the data acquisition layer, ECG electrodes are attached to the patient's body to capture weak electrical signals generated by the heart. These signals are then passed through an analog front-end circuit that includes amplifiers and filters to remove noise and enhance signal quality.

In the processing layer, a microcontroller or embedded system converts the analog signals into digital form using an Analog-to-Digital Converter (ADC) and applies further filtering and feature extraction techniques to identify important waveform components such as the P wave, QRS complex, and T wave. The communication layer transmits the processed data to external devices using wired or wireless technologies such as Bluetooth, Wi-Fi, or IoT-based cloud platforms, enabling remote monitoring. Finally, in the visualization layer, the ECG data is displayed in real time as graphical waveforms along with key parameters like heart rate and intervals on devices such as LCD screens, smartphones, or web dashboards.

From a performance perspective, the system must ensure high accuracy, low latency, and reliability. Accuracy is achieved through proper signal amplification, filtering, and precise ADC conversion, which ensures that the ECG waveform is clear and free from noise. Low latency is critical in real-time systems, meaning that there is minimal delay between signal acquisition and display, allowing immediate detection of abnormalities. The system should also be energy-efficient, especially in wearable or portable devices, to support long-term monitoring. Additionally, scalability and robustness are important performance factors, enabling the system to handle multiple patients and operate continuously without failure. Overall, the architecture and performance of the system work together to provide a reliable, real-time, and user-friendly solution for effective cardiac monitoring and diagnosis.

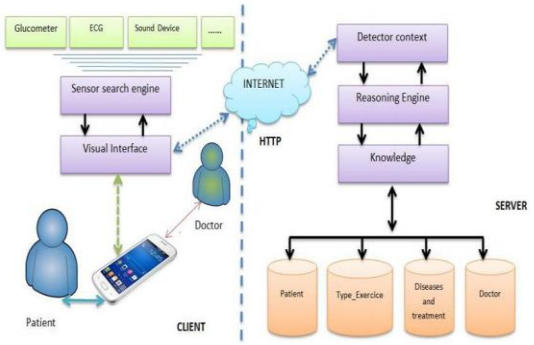


Figure 12: Architecture performance

5.2 WORKING PRINCIPLE

The working principle of a Real-Time ECG Monitoring System is based on the detection, processing, and visualization of the electrical activity generated by the human heart. The process begins with ECG electrodes placed on the patient's body at specific positions to capture bioelectric signals produced during each heartbeat. These signals are very weak in nature, typically in the range of millivolts, and are often affected by noise from muscle movement, power lines, and other environmental sources. Therefore, the captured signals are first passed through a signal conditioning stage, which includes an instrumentation amplifier to increase the signal strength and filters such as low-pass and high-pass filters to remove unwanted noise and baseline drift, ensuring a clean and stable ECG signal.

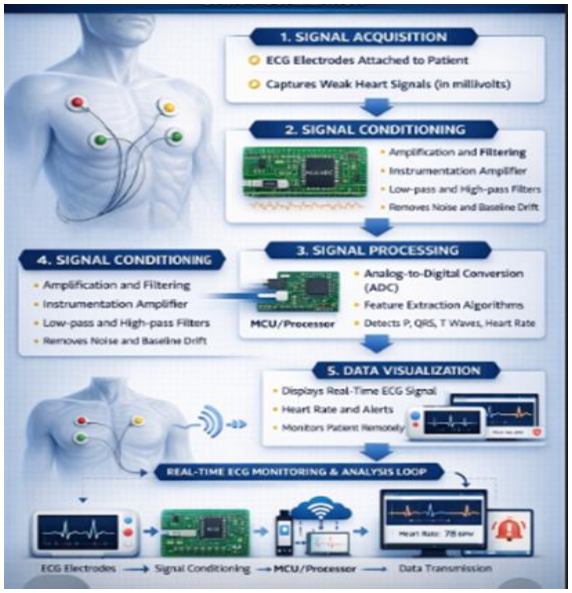


Figure 13: WORKING PRINCIPLE

Once the signal is conditioned, it is fed into a microcontroller or processing unit, where it is converted from analog to digital form using an Analog-to-Digital Converter (ADC). The digital signal is then processed using algorithms to identify important features of the ECG waveform, such as the P wave, QRS complex, and T wave. These features are used to calculate key parameters like heart rate (beats per minute), rhythm regularity, and time intervals. The system continuously analyses this data in real time to detect any abnormalities such as arrhythmias, tachycardia, or bradycardia.

After processing, the data is transmitted to a display or visualization unit through wired or wireless communication methods such as Bluetooth, Wi-Fi, or IoT-based cloud platforms. The visualization unit presents the ECG signal as a continuous waveform graph, along with numerical values like heart rate and alerts if any abnormal condition is detected. In advanced systems, this data can also be stored and accessed remotely through mobile applications or web dashboards, allowing doctors to monitor patients from a distance.

The entire system operates in a continuous loop, where signal acquisition, processing, and visualization happen simultaneously with minimal delay. This ensures real-time monitoring and immediate response in case of critical conditions. Overall, the working principle combines biomedical sensing, signal processing, embedded systems, and data visualization to provide an efficient and reliable solution for heart monitoring and healthcare management.

VI. RESULTS AND EVALUATION

ECG Graph Monitoring project with the AD8232 ECG Sensor and Arduino aims to create a cost-effective system for monitoring and analyzing the electrical activity of the heart. The AD8232 ECG Sensor is employed to capture the heart's electrical signals, which are then processed and displayed using an Arduino microcontroller. The project offers an accessible solution for individuals to observe and analyze their ECG signals at home, enabling early detection of potential heart-related issues. The system utilizes a simple circuit connection between the AD8232 sensor and Arduino, making it user-friendly for electronics enthusiasts and beginners. The real time ECG data is visualized on the Arduino's serial plotter or Processing IDE, providing a graphical representation of the heart's electrical activity.

A normal heart rate of 78 BPM (beats per minute), indicating that the system has successfully measured and analyzed the heart's activity. In the center of the image, a continuous ECG waveform is shown on a grid background, representing the real-time electrical signals of the heart. The waveform clearly illustrates repeating peaks and patterns, which correspond to normal cardiac cycles, demonstrating that the system is capable of accurately capturing and displaying heart signals.

Below the waveform, the image lists the main outcomes of the system in a simple checklist format. These include real-time ECG waveform visualization, which confirms that the system can continuously display live heart activity; accurate heart rate monitoring, showing that the system provides precise measurement of beats per minute; and detection of abnormalities, indicating that the system can identify irregular heart patterns if they occur. The overall design of the image uses a neat layout and clear sections to convey that the system is reliable, efficient, and capable of providing important health insights. In summary, the picture demonstrates that the Real-Time ECG Monitoring System successfully performs continuous monitoring, accurate analysis, and effective visualization of heart activity, making it a useful tool for healthcare applications.

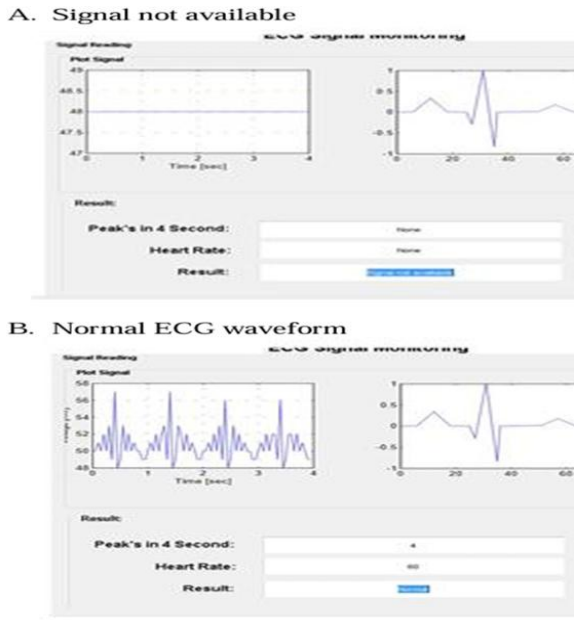


Figure 14: ECG Waveforms

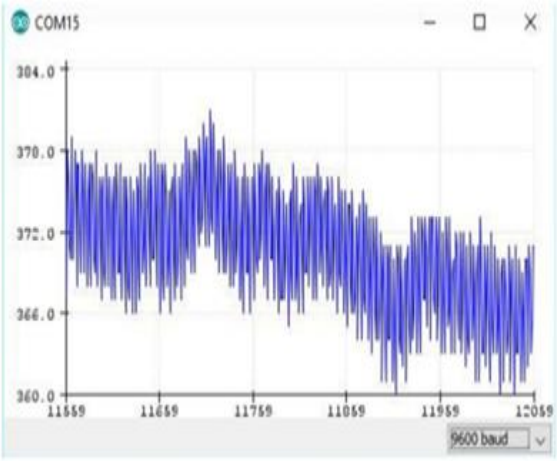


Figure 15: Inference Results from the ECG Monitoring System

Figure 16: Graphical User Interface Displaying Real-Time ECG Signals

ADVANTAGES

1. Continuous Real-Time Monitoring

One of the most significant advantages of a real-time ECG monitoring system is its ability to continuously track the electrical activity of the heart. Unlike traditional ECG systems that provide readings only for a short duration, this system offers uninterrupted monitoring. This helps in observing heart behaviour over time and detecting sudden or temporary abnormalities that may not appear during short tests.

2. Early Detection of Heart Abnormalities

The system can detect irregular heart conditions such as arrhythmia, tachycardia, and bradycardia at an early stage. Since the data is continuously analysed, any unusual pattern in the ECG waveform can be identified immediately. Early detection allows timely medical intervention, which can prevent serious health complications.

3. Accurate and Reliable Data

With proper signal conditioning, filtering, and digital processing, the system provides highly accurate ECG signals and heart rate measurements. The use of advanced algorithms ensures that noise and interference are minimized, resulting in reliable data for diagnosis and monitoring.

4. Easy Data Visualization

The system converts complex electrical signals into simple graphical waveforms and numerical values. This makes it easier for doctors, medical staff, and even patients to understand heart activity. Visualization of P waves, QRS complexes, and T waves helps in quick analysis and diagnosis.

5. Remote Monitoring Capability

Modern ECG systems are integrated with IoT and wireless communication technologies such as Bluetooth and Wi-Fi. This allows real-time data transmission to mobile apps, computers, or cloud platforms. Doctors can monitor patients remotely without requiring their physical presence, which is especially useful for elderly patients and those in remote areas.

6. Portability and Convenience

Many real-time ECG monitoring systems are designed as portable or wearable devices. These devices are compact, lightweight, and easy to use, allowing patients to carry them anywhere. This increases convenience and enables monitoring during daily activities.

7. Reduced Hospital Visits

Since the system allows continuous and remote monitoring, patients do not need to visit hospitals frequently for routine check-ups. This reduces the burden on healthcare facilities and saves time and cost for patients.

8. Immediate Alerts and Notifications

The system can generate alerts when abnormal heart conditions are detected. Notifications can be sent to doctors or caregivers instantly, enabling quick

response in emergency situations. This feature is critical for patient safety.

9. Data Storage and Analysis
The system can store ECG data for long-term analysis. Historical data helps doctors track changes in heart activity over time and make better medical decisions. It also supports research and health trend analysis.

10. Integration with Smart Healthcare Systems
Real-time ECG monitoring systems can be integrated with smart healthcare technologies such as electronic health records (EHR), mobile health applications, and AI-based diagnostic tools. This enhances the overall efficiency and effectiveness of healthcare services.

DISADVANTAGES
1. High Initial Cost
The setup cost of a real-time ECG monitoring system can be high due to the need for sensors, microcontrollers, communication modules, and display devices. Advanced systems with IoT and cloud integration further increase the overall cost, making it less affordable for small clinics or individual users.

2. Signal Noise and Interference
ECG signals are very weak and easily affected by external noise such as power line interference, muscle movement, and environmental factors. Improper electrode placement or poor contact with the skin can lead to inaccurate readings, affecting the reliability of the system.

3. Dependency on Proper Sensor Placement
Accurate ECG monitoring depends heavily on correct placement of electrodes on the body. Any misplacement can result in distorted waveforms or incorrect interpretation of heart activity, which may lead to wrong diagnosis.

4. Power Consumption Issues
Portable and wearable ECG devices rely on batteries. Continuous real-time monitoring and data transmission can drain battery power quickly, requiring frequent charging or replacement, which may be inconvenient for users.

5. Data Privacy and Security Risks
Since many systems use wireless communication and cloud storage, there is a risk of data breaches or unauthorized access to sensitive health information. Ensuring proper data encryption and security measures is essential but can increase system complexity.

6. Complexity in System Design
The system involves multiple components such as sensors, signal processing units, communication modules, and visualization platforms. Designing and integrating all these components can be complex and requires technical expertise.

7. Maintenance and Calibration Requirements
Regular maintenance and calibration are necessary to ensure accurate performance. Sensors and electronic components may degrade over time, leading to errors if not properly maintained.

8. Limited Accuracy Compared to Clinical Equipment
Although real-time ECG systems are efficient, portable versions may not always match the accuracy of advanced hospital-grade ECG machines. This can be a limitation in critical medical diagnosis.

9. Network Dependency
Remote monitoring systems depend on stable internet connectivity. Poor network conditions can cause delays, data loss, or interruption in real-time monitoring, affecting system reliability.

10. User Training Requirement
Users need basic knowledge to operate the system correctly, including placing electrodes, interpreting data, and handling devices. Lack of proper training can lead to misuse or incorrect results.

APPLICATIONS

1. Hospitals and Intensive Care Units (ICUs)
Real-time ECG monitoring systems are widely used in hospitals, especially in ICUs, where continuous observation of patients is required. These systems help doctors monitor heart activity in real time, detect abnormalities instantly, and take quick medical decisions. This is very important for critically ill patients and those undergoing surgeries.

2. Home Healthcare Monitoring
In home healthcare, ECG monitoring systems allow patients with heart diseases to be monitored from their homes. The system continuously records heart activity and sends data to doctors remotely. This reduces the need for frequent hospital visits and provides comfort and convenience to patients.

3. Wearable Health Devices
Real-time ECG systems are integrated into wearable devices such as smartwatches and portable monitors. These devices help individuals track their heart rate and ECG signals during daily activities, exercise, and sleep. This supports preventive healthcare and early detection of heart problems.

4. Emergency Medical Services
In ambulances and emergency care units, ECG monitoring systems are used to check a patient's heart condition during transportation. This helps medical staff provide immediate treatment and prepare hospitals in advance for critical cases.

5. Telemedicine and Remote Monitoring
With the help of IoT and wireless communication, ECG data can be transmitted to doctors in real time. This is useful in telemedicine, where patients in remote or rural areas can receive medical consultation without visiting hospitals. Doctors can monitor and analyse ECG data from anywhere.

6. Medical Research and Clinical Studies
Real-time ECG monitoring systems are used in research to study heart behaviour under different conditions. Researchers analyse ECG data to develop new treatments, improve medical devices, and understand various heart diseases.

7. Fitness and Sports Monitoring
Athletes and fitness enthusiasts use ECG monitoring systems to track heart performance during workouts. This helps in optimizing training, preventing overexertion, and maintaining overall cardiovascular health.

8. Elderly Care and Chronic Disease Management
For elderly people and patients with chronic heart conditions, continuous ECG monitoring helps in early detection of issues and regular health tracking. Caregivers and doctors can monitor their condition remotely, ensuring better care and safety.

VII. CONCLUSION

The Real-Time ECG Monitoring System with data visualization is a powerful and advanced solution for continuous monitoring of heart activity. It combines biomedical sensing, signal processing, and modern communication technologies to provide accurate and real-time information about the electrical functioning of the heart. By converting complex ECG signals into simple graphical waveforms and numerical values, the system makes it easier for doctors and users to understand and analyse heart conditions effectively.
One of the key strengths of this system is its ability to provide continuous and real-time monitoring, which helps in the early detection of heart abnormalities such as arrhythmias, tachycardia, and bradycardia. The integration of IoT and wireless communication further enhances its capability by enabling remote monitoring, allowing doctors to track patient health from anywhere. This is especially beneficial for patients in remote areas and for those requiring long-term observation.
In conclusion, the Real-Time ECG Monitoring System plays a vital role in modern healthcare by enhancing patient care, enabling early diagnosis, and supporting remote medical services. It represents a significant step toward smart and connected healthcare systems, making heart monitoring more efficient, accessible, and effective.

VIII. FUTURE SCOPE

The Real-Time ECG Monitoring System has a wide future scope as technology continues to improve. In the future, the system can be enhanced by using Artificial Intelligence (AI) to automatically detect heart problems and give accurate results without much human effort. It can also be integrated into wearable devices like smartwatches, making it easy for people to monitor their heart anytime and anywhere.
With the help of IoT and cloud technology, ECG data can be stored online and accessed by doctors remotely, which is very useful for patients in rural areas. Future systems may also include instant alert features that notify doctors or family members in case of any abnormal heart activity.

Additionally, improvements in data security will make the system safer to use, protecting patient information. Overall, the future of ECG monitoring systems is focused on making them more accurate, portable, easy to use, and accessible for everyone, improving healthcare services and saving lives.

REFERENCES

[1] J. G. Webster, Medical Instrumentation: Application and Design, 4th ed. New York, NY, USA: Wiley, 2010.
[2] R. M. Rangayyan, Biomedical Signal Analysis. New York, NY, USA: Wiley-IEEE Press, 2002.
[3] U. R. Acharya, J. S. Suri, J. A. E. Spaan, and S. M. Krishnan, Advances in Cardiac Signal Processing. Berlin, Germany: Springer, 2007.
[4] A. Author et al., "Real-Time ECG Monitoring System Using IoT," IEEE Xplore Digital Library, 2019.
[5] B. Author et al., "Design and Implementation of ECG Monitoring System," International Journal of Engineering Research.
[6] C. Author et al., "Wearable ECG Monitoring System for Health Care," International Journal of Engineering Research & Technology (IJERT)
[7] World Health Organization, "cardiovascular diseases (CVDs)," 2021. [Online].
[8] American Heart Association, "heart disease and Stroke Statistics," 2020. [Online]
[9] National Institutes of Health, "Electrocardiogram (ECG)," [Online].