

Distance Recognition and Internet of Things-Based Intelligent Security

Sonali Prasad¹, Pankaj Jain²

¹M. Tech. scholar, Department of Computer Science & Engineering/Information Technology, Global Institute of Technology Jaipur (Rajasthan), India-302022

²Assistant Professor, Department of Computer Science & Engineering/Information Technology, Global Institute of Technology Jaipur (Rajasthan), India-302022

DOI: <https://doi.org/10.51244/IJRSI.2025.12060007>

Received: 05 June 2025; Accepted: 09 June 2025; Published: 26 June 2025

ABSTRACT

Security systems are more ubiquitous in our daily lives. It is well acknowledged that safeguarding assets in a secure place, such as a bank vault or similar facility, is essential. Businesses are increasingly utilizing surveillance cameras to create safe environments. IoT Security pertains to the examination and formulation of strategies to safeguard data and infrastructure employed by IoT devices. IoT security pertains to the safeguarding of networked or Internet-connected devices against unauthorized access. The notion of the "Internet of Things" has gained prominence with technological advancements. Devices ranging from thermostats to video game consoles to wristwatches now possess the capability to connect to the internet and other electronics. The techniques and instruments employed to ensure the security of interconnected devices are collectively termed "Internet of Things" (IoT) security. The interlinked nature of IoT devices renders them more vulnerable to assaults. The extent of IoT security is boundless, akin to the IoT itself. Consequently, several methods exist to impact IoT security. IT administrators may safeguard their businesses from the escalating risks of cybercrime and cyberterrorism by employing strategies such as public key infrastructure (PKI) authentication, network security, and API security. It is scarcely discernible. Currently, individuals are more focused about safeguarding their information, residences, lifestyles, and belongings. As security technology advances, an increasing number of dangers arise. The implementation of security measures using the Internet of Things (IoT) and biometric identification has escalated as a consequence of this. Cyber-attacks persist as a significant concern because to the inconsistent design of IoT devices and the substantial volume of data they handle. Numerous prominent cases using the exploitation of fundamental IoT devices to breach and assault larger networks have underscored the necessity of IoT security. Distance recognition is regarded as a straightforward and easily collected biometric technique. Biometric distance recognition employs the same differentiating traits that individuals utilize for personal identification. This work utilized distance recognition to develop a reliable door locking mechanism. The system performs real-time distance detection and identification when an object or someone enters a specified range of the door detector. A functioning prototype was developed and assessed.

Key Points

1. Internet Of Things
2. Security
3. Use Of Sensors
4. Multiple SMS Notifications
5. Arduino and Boards Connectivity
6. Program Code

7. Firebase

Highlights

1. Based on Internet of Things technology and utilizing a distance detector, we suggested an intelligent home monitoring system.
2. This recommended technique protects against theft and unsolicited threats.
3. This system was constructed with an Arduino, cables, an HC-SR04 ultrasonic sensor, and an API.
4. We are sending message using SMS and API to multiple system.

INTRODUCTION

Any Internet-connected gadget is vulnerable to being hacked and abused. In the Internet of Things era of today, anybody might theoretically exploit one of the billions of linked devices to steal sensitive data, transmit malware, or even inflict actual injury [15]. Everyone involved in manufacturing and utilizing IoT devices will prioritize network security as more organizations and people utilize and depend on them. Certain IoT security measures are the responsibility of manufacturers and end users [16]. It's a team effort.

Cybersecurity is always changing. Whether you're making a smart car, a baby monitor, a pacemaker, a monitoring device for agricultural equipment, or something altogether new, you can be confident in the safety of your IoT application with the right technology, design, and maintenance [16]. We need a mechanism to notify you if your home or business property goes missing. A network of computer devices that can connect, analyze, and interact is the Internet of Things. "IoT" definitions usually exclude internet-connected gadgets. IoT devices, especially Bluetooth ones, need good security. IoT data breaches may be rising due to these issues.

Protecting gadgets that are online or linked to a network is what's meant by "IoT security." The concept of an "Internet of Things" has become more widespread as technology has advanced [17]. Almost everything manufactured today, from wristwatches to thermostats to gaming consoles, is capable of communicating with the web and other gadgets.

IoT Security refers to a set of measures used to ensure the safety of connected devices. Surprisingly, the increased vulnerability of IoT devices to attack comes from their very connectivity.

A sampling of the industries and fields that make use of IoT devices is as follows:

- Consumer Applications - Consumer IoT devices include smartphones, wearables, and "smart homes," which enable you control everything from the temperature to who enters and exits your home with a single app.
- Business Applications – Smart security cameras, vehicles, ships, product trackers, and sensors that gather data on industrial equipment are just a few examples of the Internet of Things devices currently in use by organisations.
- Government Applications – Examples of government IoT applications include animal tracking devices, traffic congestion monitors, and catastrophe alert systems. IoT devices are currently used on a global scale in the billions. Their growing importance in contemporary life has prompted closer examination of any weaknesses. We promise to look about this.

How to manage Internet of Things devices

To work, IoT devices need internal control (software updates) and external control (interoperability). Each IoT device is connected to a Command and Control (C & C) Center for management. Hub software installs and

validates software maintenance and configuration, firmware upgrades to fix bugs and security flaws, and device registration. API allows electrical devices to communicate. When an API is public, other devices and applications can use it to gather and share data. The gadget may also be managed directly via different APIs.

IOT Vulnerabilities and Security Issues

- The C&C Centre and API do a great job of handling the routine tasks involved with the IoT [18]. However, there are many exploitable weaknesses due to its centralized nature, including.
- **Unpatched vulnerabilities**—Network difficulties or requiring end users to manually download updates from the C & C Center both contribute to devices running outdated software and being open for extended periods of time. I have. Concerning recently discovered issues [19].
- **Weak Authentication**—IoT devices (like home routers) are often sent from the factory with weak passwords that may be readily broken by service providers and end users. The widespread application of automated scripts to exploit these devices is made possible when they are accessible remotely [1].
- **Vulnerable APIs** – APIs are a common target for attacks like man-in-the-middle (MITM), code injection (like SQLI), and DDoS attacks. Find out more about the effects of attacks on APIs. Devices that may be exploited provide two sorts of dangers: those to the user and those to the community at large.

User Threats

Users' sensitive information, such as financial and medical records, might be jeopardized when IoT devices are hacked. Theft of this information occurs when devices are not properly secured. Insecure endpoints may also act as entry points to the rest of the network, allowing hackers to get access to otherwise inaccessible resources [20].

Physical Harm

Pacemakers, cardiac monitors, defibrillators, and other IoT devices are examples of IoT gadgets that are becoming prominent in the medical market. These devices have their uses (doctors may adjust a patient's pacemaker from a distance), but they are also open to cyber threats [21].

The medical treatment of the patient may be jeopardized if the device is put in improperly. Although it happens seldom, it must be considered when creating regulations to protect Internet of Things devices.

Remote Exposure

IoT devices provide a far bigger attack surface than traditional technologies since they are connected to the internet. Despite the apparent advantages, this makes the device vulnerable to remote hacking. This explains why hacking techniques like phishing are so effective. IoT security must cover several access points, much as cloud security, to protect important information and resources.

Chapter 1 summarizes the device-development project. For our experiment, the HC-SR04 ultrasonic sensor detected our distance from an item. We achieved our goal with Arduino. After distance detection at many places, we suggested notifying multiple authorities via SMS.

Its low cost and easy installation assist society, businesses, and farmers. Its decreased pricing save farmers and businesses time and money. Its theft prevention purpose will create any severe security failure. This paper discusses IoT security, its relevance, and its primary dangers and attack pathways. Internet of Things device, data, and network security is another challenge. This research will assist developers create cost-effective and efficient IoT project safety solutions.

Chapter 2 examines a literature study on Internet of Things (IoT) security and the attendant problems involved.

Chapter 3 integrates the ESP8266 Node MCU hardware, the Arduino IDE (Integrated Development Environment), Firebase, and the API (Application Programming Interface) of an SMS gateway.

Chapter 4 provides detailed information on the suggested strategy or techniques for this job.

Chapter 5 gives information regarding current technologies and scope related to this work.

Chapters 6 and 7 are Discussion and Conclusions of Proposal Results.

LITERATURE SURVEY

1. **Lan Zhang et al [2005]** RFID has been viewed as a way to save time and money in a number of areas, including manufacturing, supply chain management, and retail inventories. The RFID community, however, is increasingly interested in learning more about the challenges associated with RFID system privacy and information security in order to find a workable solution to these issues. This article describes the state-of-the-art techniques for protecting personal information with RFID application systems. After that, a more secure method is introduced using symmetric encryption and random number generators. A security and privacy analysis follows in the study. The authors draw the conclusion that the technique prevents tracking, spoofing, and attacks from potential threats [2].
2. **Prathamesh Timse et al [2014]** This work offers a survey allowing one to recognize individuals by their unique face traits. To ascertain the unique identification of a particular face, the face recognition system matches it to the ones kept in the database. Finding the face in the database that closest like the given one is the aim [1].
3. **Dragos Mocrii et al [2018]** This paper provides a survey of the most prominent smart home technologies that rely on the Internet of Things. The tone of the piece is established with an explanation of what a smart house is. The article explains the smart home's complex, IoT-based architecture and details its expanded user and system functionality. Additionally, it situates smart houses in relation to the smart grid. The main areas of this research include related communication technologies, privacy and security concerns, and software elements for controlling smart homes. We have a look at the things holding back widespread use of smart home devices, as well as the ways in which these problems are now being addressed and the hopeful prospects that arise in the future [5].
4. **Vasyl Kushnir et al [2019]** This research provides a framework for developing a tool that may aid people with visual impairments in gaining more awareness of their environment. It's shown as a tracking device that can also be used as a phone. The information from the tracker is gathered by the mobile device and evaluated by a convolutional neural network that has been trained for object identification. We may use this neural network to create item labels and annotate our data. It could encourage kids to pay closer attention to their surroundings [6].
5. **Andreas et al [2019]** suggests the door is an important part of home security. The door will be kept closed all the time so that the inhabitants may feel safe in their home. But sometimes people are in such a rush to get out of the home that they don't take the time to double check that they locked the door. When it comes to monitoring the door's state, managing the door, and boosting home security, we advise using an Android app called Door Security System that makes use of IoT technology. The MQTT cloud protocol facilitates interaction between the smartphone and the door lock system. A passive infrared (PIR) sensor is embedded into the lock, and a touch sensor is attached to the handle. The alarm will sound if the door is pushed open, warning the homeowners of the intruder's presence. The evaluation determined that the door lock and smartphone interactions were safely transmitted because of appropriate encryption, and that the motion detector was accurate to within 1.6 metres [9].
6. **Teddy Surya Gunawan et al [2020]** tells us that signatures are an integral part of our identities and, as such, have great significance in our daily lives. Internet of Things (IoT)-based smart home solutions are becoming more and more common. Signature verification and recognition may also have applications in the banking, insurance, and home security sectors. The primary goal of this project is to design and

build a signature recognition system that can function on a single board computer, specifically a Raspberry Pi 3 with an LCD touchscreen. The collected signature image was first chopped and reduced in size. Next, the ANN was trained using the collected binary image attributes. To verify the authenticity of the input signature, the trained ANN was applied to the data. A recognition percentage of 99.7 percent was identified [4].

7. **Zhu Zhiguo et al [2020]** tells In this era of computers and the internet, authentication is of paramount importance. The state of autonomous embedded systems has advanced greatly in the present day. Applications like surveillance and private security have shown the worth of an automated embedded system. The vulnerability of today's smart door locks to damage and failures compromises safety. Almost all externally advanced door locks are insecure because they need a password or face recognition. In order to offer an Efficient attitude tracking method (EATA), this research will first introduce the reader to the open-source programme OpenCV. Additionally, this article intends to confirm that both an old-fashioned key lock system and a contemporary keyless entry system provide a certain degree of security and reliability [12].
8. **Chanthaphone Sisavath et al [2021]** shows how, as the number of burglaries increases, people are growing more concerned about keeping their homes safe. The "Internet of Things close to life and easy to use" concept is used to create an IoT-based smart home system. This idea includes the node module, the escape board section, and the APP module. The following are the main features that are implemented: The temperature and humidity readings from the node board's sensors can be seen on the PC's browser, and the control board's LED light may be turned on and off using the same interface. In addition, the node board's collected temperature and humidity data may be shown on a connected app. The master MCU for the gateway board, which serves as a communication bridge, is an NPC LPC1769. Using the DP83848 chip and the LPC1769 integrated Ethernet module, the system creates an Ethernet service controller. I created a gateway system using the W25Q128 memory chip, NB-IOT module, and WiFi module. The programme, which is based on, is integrated into the web server function [8].
9. **Navya Saxena et al [2021]** Combine two separate and developing technologies—facial authentication and voice recognition—to create an all-encompassing Smart Home Security system that improves privacy and security. The user may check in on his house from his mobile device, laptop, or desktop computer with the help of the suggested app. Using this technique, a live video feed of the visitor at the door is analysed to determine if a known face is present. If so, the owner's identity is verified by matching their face against a database of registered users. In order to validate the outcomes of the face authentication procedure, speech recognition was applied. Throughout the process, neural networks are used. The suggested model has an overall accuracy of 82.71 percent, with a face authentication accuracy of 87.5 percent and a speaker authentication accuracy of 84.62 percent. In the present COVID-19 scenario, as well as in theft and burglary cases, the ability to recognise faces through masks will aid in precisely validating a person's identification, which is beneficial. This intelligent security system may thus be employed in places other than residences, such businesses, banks, and shopping malls [10].
10. **Jinu Mohan et al [2021]** The author asserts that in light of growing societal dangers, installing a home security system is more crucial than ever. Automatic user identification based on observable behavioural or physiological traits is the focus of biometrics research and development. Secret sharing with visual cryptography entails encrypting a secret picture into shares that can't be decoded back into the original secret image. The biometric template is vulnerable to manipulation since it is kept in a centralised database. An approved user's access to a resource may be revoked if the biometric template for that user is altered.. Visual cryptography methods that secure facial recognition can be used to solve this problem. The research's major goal is to create an algorithm that blends CVC and Siamese networks. This research blends face image visual cryptography into a biometric application. By comparing learning characteristics to verification tasks, the Siamese network is required to accomplish one-shot learning. Through the identification of a face in an input picture, we apply face authentication in this study to assure robustness. This research explores the possibility of utilising visual cryptography to protect the privacy of biometric information. It is decided that the suggested method is preferable to other systems due to its higher accuracy (93%) when compared to those in use at the present time [11].

11. **Trinanjana Bagchi et al [2022]** reminds us that protecting information, property, and people is paramount. Biometric authentication and the Internet of Things (IoT) have gained popularity as a means of enhancing security. One widely accepted kind of biometric identification is facial recognition since it can be used everywhere data is accessible. Biometric face recognition uses the same features that allow people to tell one person apart from another. This research looked at face recognition as a potential method for creating a foolproof door lock. A face-ID security system was developed with the help of the ESP32-CAM. The technology does in-camera face detection and recognition in real-time. To analyse the operation, efficiency, and efficacy, a prototype was constructed and tested [3].
12. **Ismail Keshta et al [2022]** indicate that the Internet of Things (IoT) has recently brought the idea of a smarter world into focus with a variety of services and a substantial amount of data "Smart health care" is becoming increasingly popular in the healthcare community, industry, government, and academia as smart Multiple Sensorial Media (MulSeMedia) systems, the cloud, and things technologies advance. The study set out to determine the security and privacy threats posed by AI-driven IoT (AIIoT) and provide solutions to those problems. A qualitative methodology was used, and information was gleaned through interviews and documents found in libraries and archives. The study's results suggest that the proliferation of AI-driven IoT (AIIoT) has led to a rise in the number of connected sensors and devices, which in turn has opened up a variety of security flaws [7].
13. **Sonali Prasad et al [2022]** stated that the HC-SR04 ultrasonic sensor can be programmed using Arduino IDE to determine the distance to an item as part of an intelligent security system based on distance recognition and the Internet of Things. After a certain distance has been detected, a message may be sent using an SMS gateway.

Introduction About Software

About Arduino IDE

Components

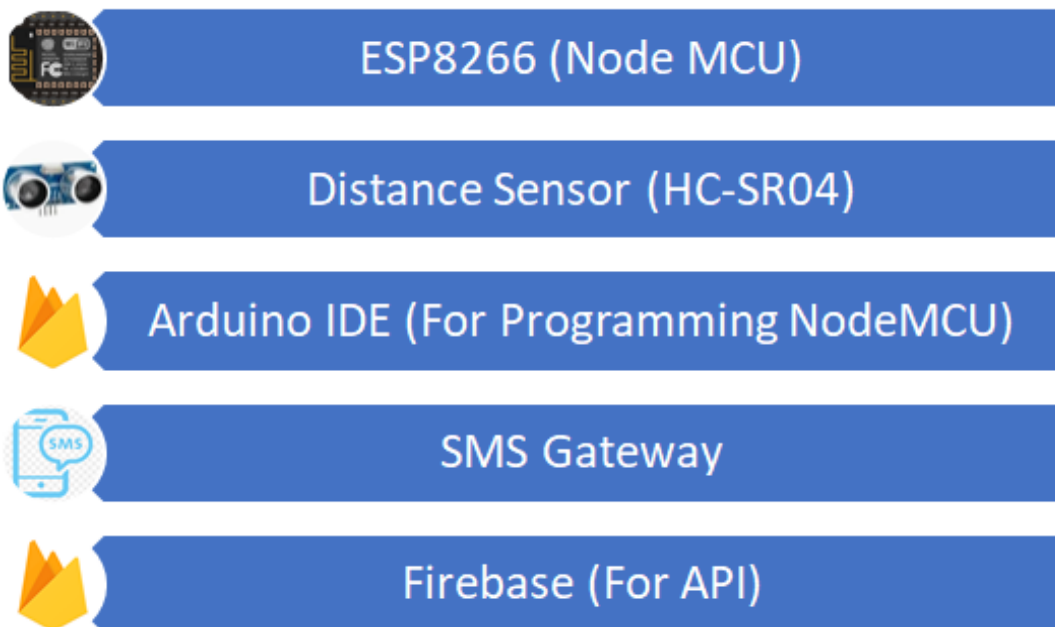


Fig. 1 Workflow Component

Node MCU: NodeMCU is a popular development board based on the ESP8266 Wi-Fi module, which is widely used for Internet of Things (IoT) projects. It integrates a microcontroller with Wi-Fi capabilities, making it easy to connect devices to the internet. The board is often programmed using the Lua script or the Arduino IDE, thanks to its user-friendly interface and extensive community support. NodeMCU is known for its affordability, versatility, and ease of use, making it a favorite among hobbyists and professionals for creating smart gadgets and automation systems [14].

Distance Sensor (HC-SR04): The HC-SR04 is a widely used ultrasonic distance sensor. It measures the distance to an object by emitting an ultrasonic pulse and then calculating the time it takes for the echo to return. This sensor is commonly used in robotics and automation projects to measure distances accurately. It's known for its simplicity, affordability, and ease of integration with microcontrollers like Arduino and Raspberry Pi. With a range typically between 2 cm and 4 meters, it's useful for obstacle detection, level measurement, and other distance-related applications [22].

Arduino IDE: The Arduino IDE (Integrated Development Environment) is a software application used for writing, compiling, and uploading code to Arduino boards. It provides a user-friendly interface for programming these microcontrollers [23]. Key features of the Arduino IDE include:

1. **Code Editor:** It has a simple text editor with syntax highlighting for Arduino's programming language, which is based on C/C++.
2. **Libraries:** It includes a variety of built-in libraries and allows users to include additional libraries to simplify coding for different sensors and modules.
3. **Compiler:** The IDE compiles your code into machine-readable format, which the Arduino board can execute.
4. **Uploader:** It communicates with the Arduino board and NodeMCU to upload the compiled code via USB.
5. **Serial Monitor:** This feature allows real-time communication between the Arduino board and the computer, useful for debugging and monitoring data.
6. **Cross-Platform:** It is available for Windows, macOS, and Linux, making it accessible to a wide range of users.

The Arduino IDE is praised for its simplicity and accessibility, making it a popular choice for beginners and experienced developers alike in the world of electronics and microcontroller programming.

Firebase: Firebase is a google based container which is widely used for hosting applications (android, iOS, web based among many others) with robust framework of database (Firestore) and Restful APIs. In our project we will use Firestore to store the data and will create Restful API which will be called from the program flashed in ESP8266 when object distance is less than a limit (i.e. 10 cms for example) [24].

SMS Gateway: SMS gateway is an external party APIs which are used to send SMS. In our case we will use an API provided by BhashSMS company [29].

About Node MCU

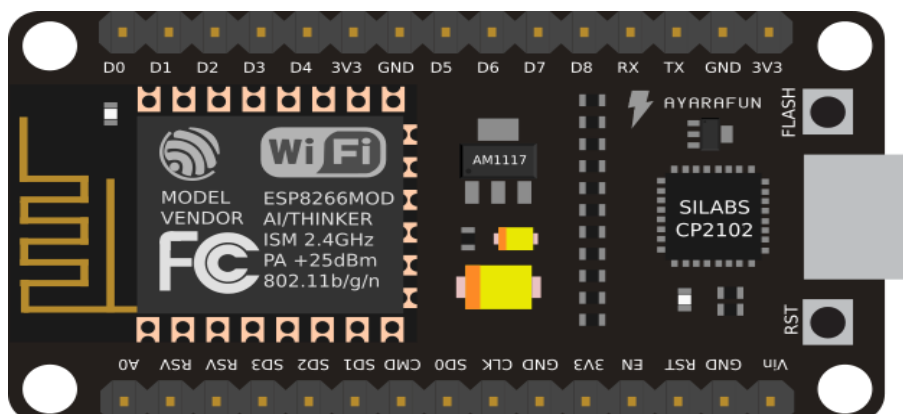


Fig. 2 Node MCU

Pins Description [25]

The major pins used are listed as under –

Vin: This is the 5V power input pin.

GND: Ground pins, usually two or more, used to complete the circuit.

D0 - D8: These are digital input/output pins. Each pin has specific functions and can be used for general-purpose I/O.

A0: This is the analog input pin, which can read analog signals from 0 to 3.3V.

RST: Reset pin, used to reset the microcontroller.

EN: Enable pin, used to enable or disable the chip.

TXD: Transmit pin for serial communication.

RXD: Receive pin for serial communication.

About Node MCU

NodeMCU is a popular open-source development platform that integrates a microcontroller with Wi-Fi capabilities, making it a favored choice for Internet of Things (IoT) projects. Its name is a blend of "Node" (referring to its roots in Node.js) and "MCU" (Microcontroller Unit).

At its core, NodeMCU features the ESP8266 microcontroller, a low-cost Wi-Fi system on a chip (SoC) developed by Espressif Systems. The ESP8266 is known for its ability to handle complex tasks while maintaining a low power consumption profile, which is ideal for battery-operated IoT devices.

Hardware Overview:

The NodeMCU board combines the ESP8266 chip with a USB-to-serial adapter, which simplifies programming and communication with a computer. The board is equipped with several GPIO (General Purpose Input/Output) pins, which can be used for various digital and analog functions. It also includes other peripherals such as PWM (Pulse Width Modulation), I2C (Inter-Integrated Circuit), and SPI (Serial Peripheral Interface) interfaces, allowing for flexibility in connecting sensors, actuators, and other modules.

Software Development:

One of NodeMCU's strengths is its support for multiple programming environments. The most common programming language for NodeMCU is Lua, thanks to the NodeMCU firmware. Lua is a lightweight scripting language known for its simplicity and efficiency. However, many developers prefer using the Arduino IDE, which supports NodeMCU via additional libraries and tools. The Arduino IDE allows programmers to use C/C++ for coding, which might be more familiar to those with a background in traditional microcontroller programming.

The NodeMCU board also supports MicroPython, a version of the Python programming language optimized for microcontrollers. MicroPython offers a more straightforward and high-level approach to programming, which can be appealing for rapid development and prototyping.

Networking and Connectivity:

NodeMCU's built-in Wi-Fi capability is a key feature that enables it to connect to wireless networks and communicate with other devices over the internet. This connectivity is crucial for IoT applications where devices need to send or receive data from cloud services, other devices, or user interfaces. The ESP8266's Wi-

Fi capabilities support both client and server modes, allowing NodeMCU to function as a Wi-Fi access point or to connect to existing Wi-Fi networks.

Applications:

NodeMCU is versatile and used in a wide range of applications. Common projects include home automation systems, where it can control lights, temperature, and other household devices remotely. It is also used in environmental monitoring, such as tracking temperature and humidity levels and sending data to cloud-based services. Other applications include creating smart gadgets, prototyping electronic projects, and educational purposes, where its low cost and ease of use make it a valuable tool for learning about microcontrollers and IoT.

Community and Resources:

The NodeMCU community is vibrant and active, with numerous online forums, tutorials, and example projects available. This support ecosystem is beneficial for both beginners and experienced developers, providing resources for troubleshooting, learning, and inspiration.

In summary, NodeMCU is a powerful and accessible development platform that combines the ESP8266 microcontroller's capabilities with easy programming options. Its integration of Wi-Fi and versatility in programming environments make it an excellent choice for a wide array of IoT and electronics projects.

Distance Sensor (HC-SR04) :



Fig. 3 Ultrasonic Sensor

The HC-SR04 ultrasonic sensor calculates distance using the time-of-flight principle [26]. Here's a step-by-step explanation of how it works:

1. **Ultrasonic Pulse Emission:** The sensor has a trigger pin (Trig) that you need to send a short pulse to. When you send a HIGH signal to the trigger pin, it generates an ultrasonic pulse (sound waves) from the transmitter (Tx) at a frequency of 40 kHz.
2. **Echo Reception:** The pulse travels through the air and reflects off any nearby object. The reflected sound waves are then received by the sensor's receiver (Rx).
3. **Pulse Duration Measurement:** When the echo returns to the sensor, it causes the Echo pin to go HIGH. The sensor measures the duration for which the Echo pin stays HIGH. This duration represents the time it took for the ultrasonic pulse to travel to the object and back.
4. **Distance Calculation:** The time measured is used to calculate the distance using the speed of sound. The formula for calculating distance is:

$$\text{Distance} = \frac{\text{Time} \times \text{Speed of Sound}}{2}$$

The division by 2 is because the time measured is for the round trip (to the object and back).

In most cases, the speed of sound is approximately 343 meters per second (or 0.0343 cm/μs), but it can vary based on temperature and humidity. For practical purposes in many applications, this constant is used to convert the time to distance in centimetres or inches.

5. **Output:** The calculated distance is then used for various applications, such as obstacle avoidance in robots, distance measurement in automation systems, or even level sensing.

In programming terms, you typically send a trigger pulse, measure the time it takes for the echo pulse to return, and then compute the distance based on this time.

METHODOLOGY

Project Description

This work aims to detect any suspicious activities happening at the entry door of user. For example, if in absence of user, someone tries to break open the door, this unit (which is attached to the wall and is less than 10 cm away), the distance sensor monitors the opening of door which becomes less than 10cms when door is opened, NodeMCU unit calls the Firebase API which calls the SMS API which in turn sends the SMS to the users as defined in database.

Number of users who are supposed to receive the SMS are saved in firebase database who all receive SMS when door is opened and they get alerted of possible intrusion.

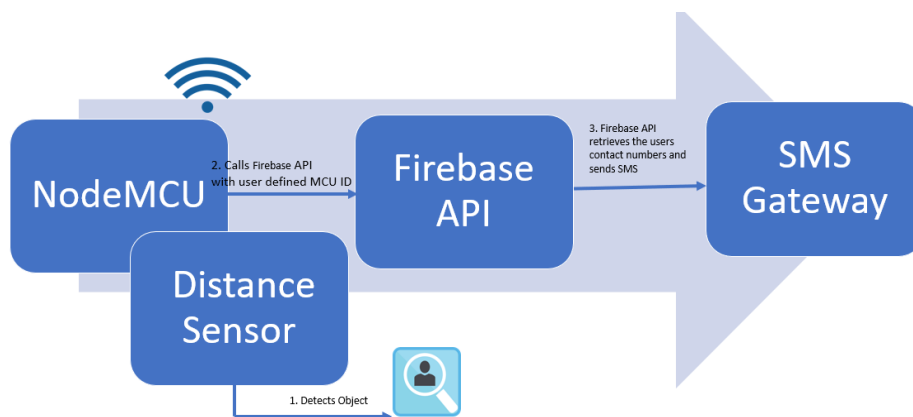


Fig. 4 Process Flow of Work Using Firebase

Workflow of the project

1. Distance Sensor (HC-SR04) is attached with NodeMCU
2. NodeMCU is configured with the WIFI available at home.
3. When someone opens the door, It passes through the distance sensor attached at the door wall. At some point while opening the door distance sensor senses the distance of object was less than 10cms away.
4. NodeMCU calls the Firebase API which in turn retrieves the users who are supposed to receive SMS.
5. It calls SMS gateway API to send SMS and all the related users receive the SMS.

6. A flag is configured in the database, and when disabled, the SMS are not sent to users. Which is useful in cases when user may not want to receive SMS in some situations for example, when they are at home.

Circuit Diagram and Explanation

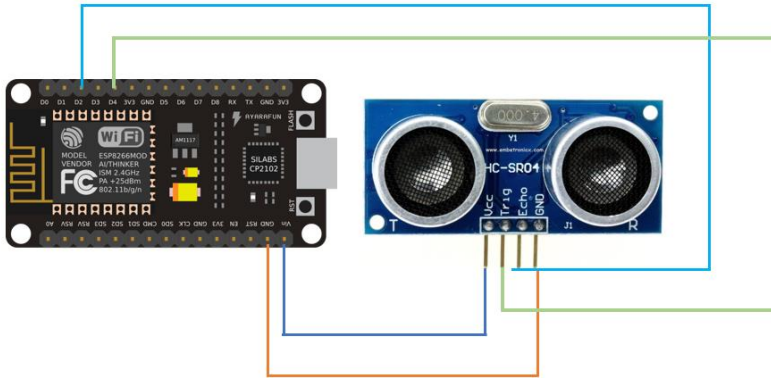


Fig. 5 Circuit Dig. Of HC-SR04 ultrasonic sensor and the ESP8266

1. Vcc pin of distance sensor is connected to Vin of Node MCU
2. GND pin of distance sensor is connected to GND pin of Node MCU
3. Echo pin of distance sensor is connected to D2 pin of Node MCU
4. Trigger pin of distance sensor is connected to D4 pin of Node MCU

Arduino Program for distance calculation in Arduino UNO IDE

```
#include <ESP8266WiFi.h>

#define echoPin D2
#define triggerPin D4
#define ledPin D0

#define WIFI_SSID "ManXXX4g"
#define WIFI_PASSWORD "9971XXX7"

long duration, distance;
void setup() {
  Serial.begin(9600);
  pinMode(triggerPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(ledPin, OUTPUT); // Initialize the LED_BUILTIN pin as an output
  Serial.begin(9600);
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);

  Serial.print("connecting");
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
  }

  Serial.println();
  Serial.print("connected: ");
  Serial.println(WiFi.localIP());
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(triggerPin, LOW); // Turn the LED on (Note that LOW is the voltage level
  delayMicroseconds(2);
  digitalWrite(triggerPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(triggerPin, LOW);

  duration = pulseIn(echoPin, HIGH);
  distance = duration / 58.2;
  String disp = String(distance);

  Serial.print("Distance :");
  Serial.print(disp);
  Serial.print(" cm\n");

  if(distance < 10){
    digitalWrite(ledPin, HIGH);
  }
  else{
    digitalWrite(ledPin, LOW);
  }
  delay(1000); // Wait for a second
}
```

Program Code 1

The program described above activates the LED when an object is detected within 10 centimetres of the distance sensor. This prototype ensures the distance sensor functions correctly with the NodeMCU, and future developments will involve using the API to send SMS messages. Below is an explanation of the program:

1. The Echo pin of the distance sensor is connected to the D2 pin on the NodeMCU, and is therefore designated as the Echo Pin.
2. The Trigger pin of the distance sensor is connected to the D4 pin on the NodeMCU, and is thus designated as the Trigger Pin.
3. An LED is connected to the D0 port on the NodeMCU. The LED will be turned on when an object is within 10 centimetres.
4. The Wi-Fi network name and password are stored in variables named WIFI_SSID and WIFI_PASSWORD, respectively.
5. Global variables for duration and distance are defined.
6. The setup function initializes the Arduino settings. In this function:
 - The NodeMCU's baud rate is set to 9600 bps (bytes per second), which determines the data transfer rate.
 - The Trigger pin is configured as an output.
 - The Echo pin is set as an input.
 - The status method of the Wi-Fi class is called to connect to the Wi-Fi network.
7. The loop function continuously executes in a cycle.
8. In the loop function:
 - The Trigger pin is first set to LOW.
 - After a 2-millisecond delay, the Trigger pin is set to HIGH, generating an ultrasonic pulse.
 - After 10 milliseconds, the Trigger pin is set back to LOW.
 - The duration for the ultrasonic pulse to return to the Echo pin is measured.
 - This duration is then converted into a distance measurement in centimetres.
 - If the calculated distance is less than 10 centimetres, the LED is turned on.

Firestore

Firestore is a comprehensive platform developed by Google that provides a suite of tools and services designed to help developers build and manage high-quality applications. Launched in 2011 and acquired by Google in 2014, Firestore has become a popular choice for developers looking to create web and mobile applications with robust backend functionality [27]. Its offerings span various aspects of app development, including real-time databases, authentication, cloud storage, and analytics, making it a versatile solution for both small projects and large-scale applications.

One of the core features of Firestore is its Real-time Database. This NoSQL cloud database allows developers to store and sync data between users in real-time. The real-time capabilities are particularly useful for applications that require instant updates, such as chat applications or collaborative tools. Data is stored as JSON and synchronized across all clients in real-time, ensuring that all users see the same information simultaneously. Firestore's Realtime Database also supports offline capabilities, allowing apps to continue functioning even when the device loses connectivity.

Another key feature is Firebase Firestore, a more advanced and flexible database offering compared to the Real-time Database. Firestore supports more complex data structures and querying capabilities. It also provides automatic scaling and strong consistency guarantees [28]. Like the Real-time Database, Firestore allows for real-time synchronization, but it also offers enhanced querying and indexing features, making it suitable for applications with more sophisticated data requirements.

Firebase Authentication simplifies the process of managing user identities. It supports various authentication methods, including email and password, social media logins (such as Google, Facebook, and Twitter), and even anonymous authentication. This integration reduces the need for developers to create their own authentication systems and ensures a secure and seamless user experience.

For file storage needs, Firebase offers Cloud Storage. This service allows developers to upload, download, and manage user-generated content such as images, videos, and other files. Cloud Storage is built on Google Cloud Storage, providing high scalability and security. It integrates seamlessly with other Firebase services, enabling developers to manage file uploads and downloads efficiently.

Firebase Hosting is another valuable feature, providing fast and secure hosting for web applications. It offers a global content delivery network (CDN) to ensure that your application loads quickly from anywhere in the world [30]. Firebase Hosting supports single-page applications (SPAs), dynamic content, and custom domains, making it an ideal choice for deploying web apps with minimal configuration.

Analytics and performance monitoring are critical for understanding and optimizing app usage. Firebase Analytics offers detailed insights into user behaviour, allowing developers to track user interactions, app usage patterns, and other key metrics. This data helps developers make informed decisions about app improvements and marketing strategies. Firebase Performance Monitoring provides real-time insights into app performance, identifying bottlenecks and issues that can affect user experience.

Firebase also integrates with various Google Cloud services, enabling developers to leverage additional tools and capabilities. For example, Firebase can be used in conjunction with Google Cloud Functions to run server-side code in response to events triggered by Firebase features. This integration allows for complex workflows and automation without needing to manage servers manually.

In summary, Firebase is a powerful and comprehensive platform for app development that addresses many common backend needs. Its real-time databases, authentication, cloud storage, hosting, and analytics tools offer a wide range of functionalities, helping developers build scalable, high-performance applications with ease. Whether you're developing a simple app or a complex enterprise solution, Firebase provides the tools and infrastructure needed to create a robust and reliable product.

About Firestore

Firestore, formally known as Firebase Firestore, is a cloud-hosted NoSQL database service offered by Google as part of the Firebase suite. It provides a flexible, scalable solution for storing and synchronizing data in real-time, making it particularly well-suited for modern web and mobile applications [31].

Data Model:

Firestore organizes data into documents and collections. A document is a JSON-like object that contains fields (key-value pairs) and can hold various data types such as strings, numbers, booleans, arrays, and nested objects. Documents are grouped into collections, which are essentially containers for documents. This hierarchical structure allows for a flexible and scalable data organization, accommodating a wide range of application needs [32].

Real-Time Synchronization:

One of Firestore's standout features is its real-time synchronization capabilities. It allows clients to listen to changes in the database and automatically update the user interface when data changes. This is particularly

useful for applications requiring real-time collaboration, such as chat apps or collaborative editing tools, where users need to see changes from other users instantly.

Scalability and Performance:

Firestore is designed to handle large-scale applications with high data throughput. It supports automatic scaling, meaning it can grow with your application's needs without requiring manual intervention. Firestore's underlying infrastructure ensures high performance, even with large datasets and complex queries. It achieves this through its distributed architecture, which allows for horizontal scaling and efficient data handling [32].

Offline Support:

Firestore provides offline support for both web and mobile applications. This means that users can continue interacting with the application even when they are not connected to the internet. Changes made offline are automatically synchronized with the server when connectivity is restored. This feature enhances user experience by ensuring that applications remain functional and responsive regardless of network status.

Security:

Firestore includes robust security features through Firebase Security Rules, which control access to data based on user authentication and authorization. Rules can be configured to enforce complex access controls, ensuring that only authorized users can read or write specific data. This helps protect sensitive information and maintain data integrity.

Integration and APIs:

Firestore integrates seamlessly with other Firebase services, such as Firebase Authentication, Firebase Cloud Functions, and Firebase Analytics. This integration provides a comprehensive ecosystem for developing, deploying, and monitoring applications. Firestore offers client libraries for various platforms, including Android, iOS, and web, as well as a REST API for server-side interactions. These libraries and APIs facilitate easy integration into diverse application architectures.

Querying and Indexing:

Firestore supports powerful querying capabilities, allowing developers to filter and sort data based on various criteria. Queries can be performed on individual documents or across collections, and Firestore supports compound queries with multiple filter conditions. To optimize query performance, Firestore automatically indexes fields used in queries, though developers can also define custom indexes for more complex query scenarios.

Pricing:

Firestore follows a pay-as-you-go pricing model, where costs are based on the amount of data stored, the number of read and write operations, and network bandwidth usage. This model can be cost-effective for smaller applications but may require careful management as usage scales.

Community and Ecosystem:

Firestore benefits from a strong community and a rich ecosystem of tools and resources. Its extensive documentation, tutorials, and community support help developers quickly get started and resolve issues. The service is continually updated with new features and improvements, driven by user feedback and technological advancements.

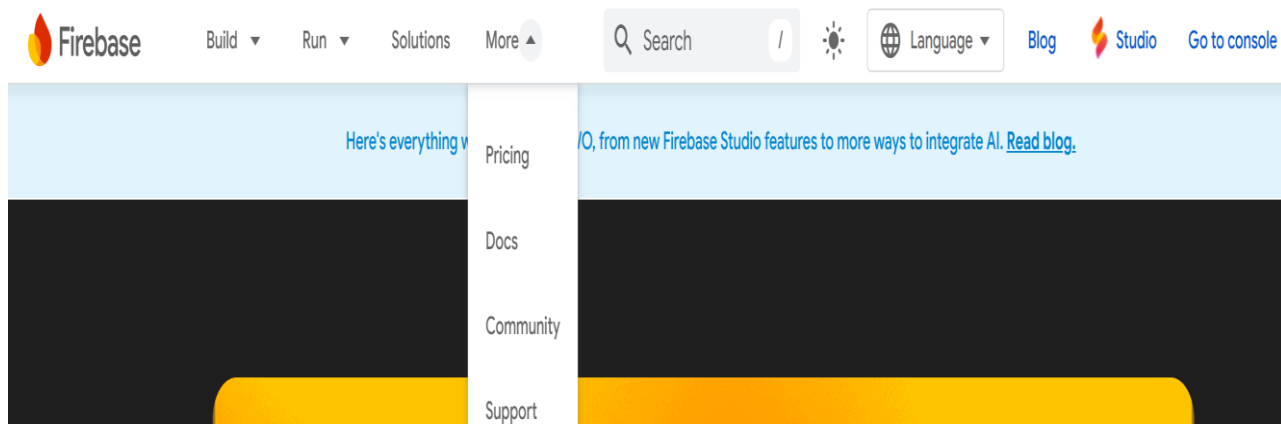
In summary, Firestore is a powerful and versatile NoSQL database service that offers real-time synchronization, offline support, and scalability. Its integration with Firebase and its robust security and

querying features make it a popular choice for building modern, data-driven applications across various platforms.

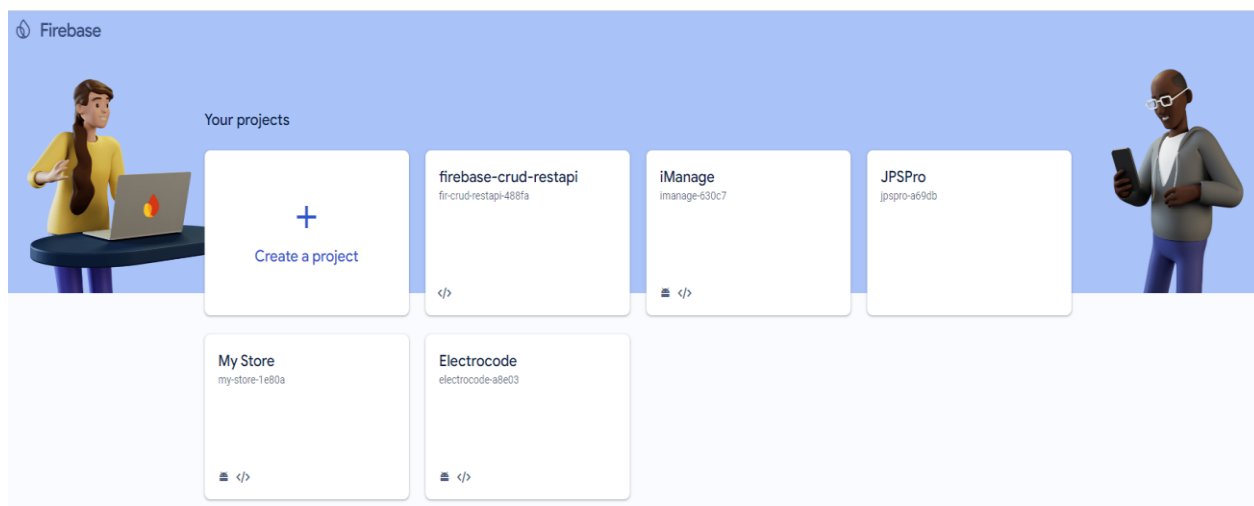
Creating a Dataset at Firestore

Firestore is a NoSQL database provided by Firebase. We first need to create a project at Firebase, and then we can use Firestore to create our dataset. First, we will see in the below steps on how to create a project in firebase [33].

1. Open URL – www.firebase.google.com
2. Login with your google account
3. Press “Go to Console” at top right corner
- 4.



5. You will be directed to below page



6. You need to press Create a Project and enter the project details. Thereafter, you need to follow the steps and you will have your project ready.
7. For our purpose, I have already created the project firebase-crud-restapi as can be seen in screenshot.

Firestore database

We created our firebase project in above steps. Now Firestore database is ready for us to use. In our project we are manually creating the records but as an enhancement, this can be done by a web page or android app [34].

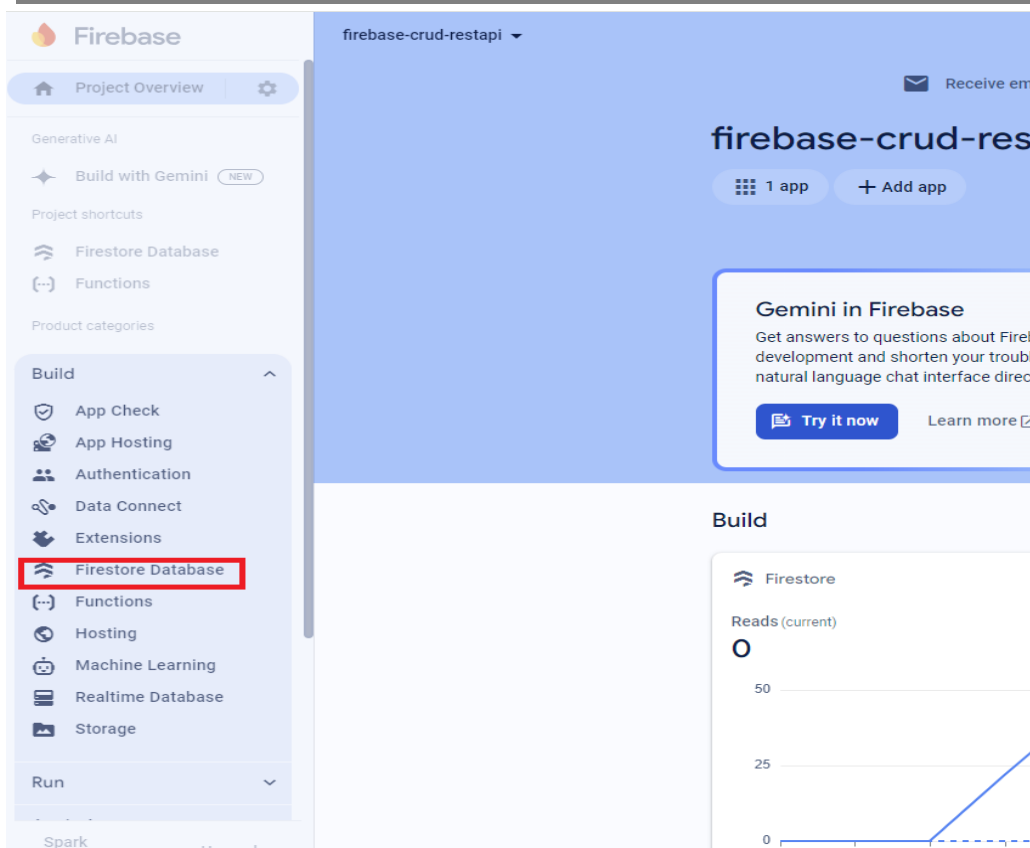
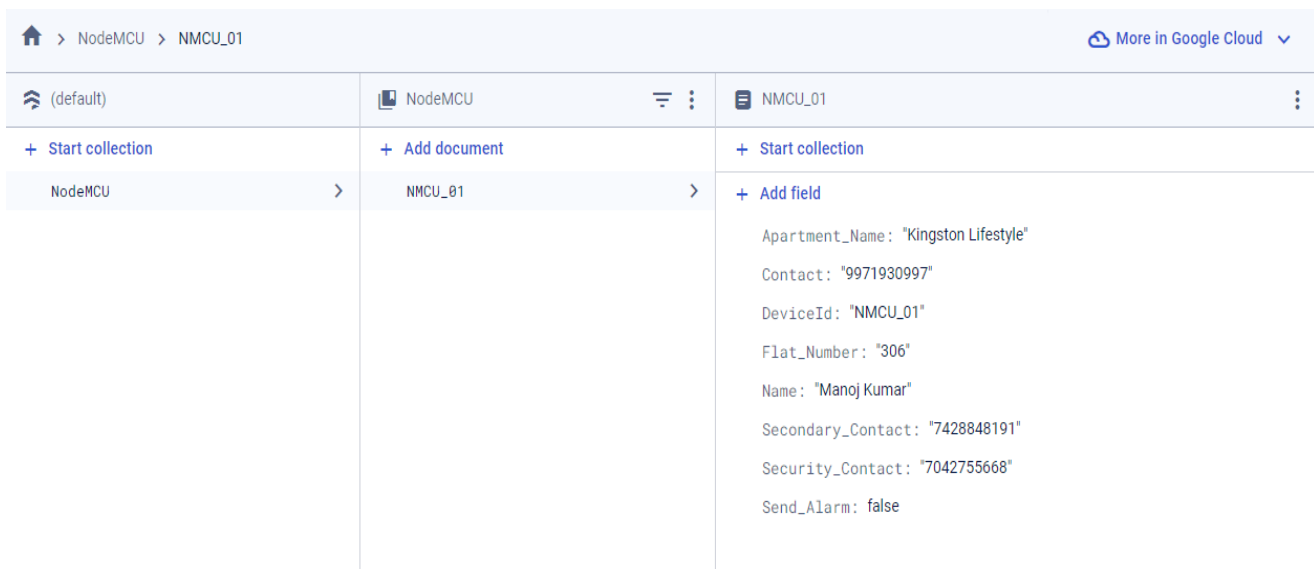


Fig. 6 Firestore Database

Upon clicking on the project we are directed to above screen. Firestore Database is highlighted in above screen shot. Once we click Firestore Database we get to see below screen-



Above screenshot is the data we are going to use for our project. Firebase is a NoSQL database which works on Collection -> Document -> Collection model and is an established framework.

In our example NodeMCU is the main collection which can have the collection of controllers (In our case NMCU_01)

Each controller is a separate document which can have it's own collection. (Apartment name, Contact (to whom send SMS), Secondary Contact (To send SMS) Flat Number, Name of owner of Flat, Security Contact number (Who will receive the SMS mentioning flat number) and a flag whether to send SMS at all or not. If true, SMS will be sent or else any activity on the device will be ignored.

In our example, we have just one document in NodeMCU collection, but if we have more devices deployed in different flats those documents (with their device id) will be added to the collection.

Creating the REST API to send SMS

Firebase is a full fledged container which offers set of services to create easy to go APIs. We can make use of Firebase functions to create RESTful API.

Firebase Functions

Firebase Functions, also known as Cloud Functions for Firebase, is a serverless computing service offered by Google as part of the Firebase platform. It allows developers to run backend code in response to various events without having to manage server infrastructure. This serverless model simplifies application development by handling scaling, provisioning, and maintenance automatically.

Core Concepts:

1. **Event-Driven Execution:** Firebase Functions are designed to execute code in response to specific events. These events can come from various Firebase and Google Cloud services. Common triggers include changes in Firestore or Realtime Database, authentication events, Cloud Storage operations, and HTTP requests. This event-driven nature makes it easy to build reactive applications that respond dynamically to changes.
2. **Function Types:**
 - **HTTP Functions:** These are triggered by HTTP requests and can be used to create RESTful APIs, handle webhooks, or serve dynamic content. They operate as standard web servers and can respond with JSON, HTML, or any other data format.
 - **Background Functions:** These functions respond to events from Firebase services like Firestore, Realtime Database, Cloud Storage, or Firebase Authentication. They are useful for tasks like processing data, sending notifications, or performing asynchronous operations in response to changes in your Firebase database or storage.
3. **Development and Deployment:**

Firebase Functions are written in JavaScript (Node.js), TypeScript, or Python. Developers write their code using the Firebase CLI or integrate it into their development workflow with tools like Visual Studio Code. Once the code is ready, it is deployed to Google Cloud, where it runs in a managed environment. The deployment process involves using the Firebase CLI to push your functions to the cloud, where they are automatically scaled and managed.

4. Scaling and Performance:

Firebase Functions automatically handle scaling based on demand. Whether your function receives a few requests per day or thousands per minute, Firebase Functions will scale up and down as needed, ensuring that performance remains consistent without manual intervention. This scalability is particularly useful for handling variable workloads and ensuring that applications remain responsive.

5. Security and Access Control:

Security is a key consideration for Firebase Functions. Functions can be secured using Firebase Authentication and Firebase Security Rules. For HTTP functions, access can be controlled through authentication tokens and API keys. For background functions, permissions are managed through Google Cloud IAM (Identity and Access Management), ensuring that only authorized entities can invoke the functions.

6. Billing:

Firebase Functions follow a pay-as-you-go pricing model. Costs are based on the number of function invocations, execution time, and memory usage. There is also a free tier that allows for a certain number of invocations and compute time per month, making it cost-effective for small-scale applications or testing purposes.

7. Integration with Other Firebase Services:

Firebase Functions integrate seamlessly with other Firebase products like Firestore, Realtime Database, Cloud Storage, and Firebase Authentication. This tight integration allows for the creation of powerful workflows and automations, such as processing data changes, managing user authentication, or handling file uploads.

8. **Use Cases:** Common use cases for Firebase Functions include:

- **Data Validation:** Validating or transforming data before it is saved to Firestore or Realtime Database.
- **Notifications:** Sending push notifications or emails in response to user actions or database changes.
- **Webhook Handling:** Responding to events from external services or APIs.
- **Scheduled Tasks:** Performing periodic tasks using Cloud Scheduler, such as data clean-up or report generation.

Community and Ecosystem:

Firebase Functions benefit from a broad community and a rich ecosystem of tools and resources. The Firebase documentation, tutorials, and community forums provide extensive support for developers at all skill levels.

In summary, Firebase Functions offer a flexible and scalable serverless computing solution that simplifies backend development by automating infrastructure management. Its event-driven model, seamless integration with Firebase services, and automatic scaling make it a powerful tool for building dynamic and responsive applications.

Creating the API

1. We have used Visual Studio Code as an IDE
2. We will use Javascript to create firebase function

Below is the code snippet-

1. From Line No. 1 till Line 16 are setup for firebase.
2. At line number 18 we have created a GET route for the rest API, which will take mcuid as a parameter.
3. From the firebase we get the document related to this mcuid.
4. Once we have the document we are fetching the details like Primary Contact, Secondary Contact, Security Contact, Flat Number, and a flag (send_alarm) which tells whether to send SMS or not.
5. We have defined a function called SendSMS which takes the contact number and message as parameters and we send SMS to each party involved in 34 till 37


```

1 const functions = require('firebase-functions');
2 const admin = require('firebase-admin');
3 var serviceAccount = require("../permissions.json");
4
5 admin.initializeApp({
6   credential: admin.credential.cert(serviceAccount)
7 });
8 var db = admin.firestore();
9
10 const express = require('express');
11 const app = express();
12 const cors = require('cors');
13
14 app.use(cors ({origin : true}));
15
16 //Route
17
18 app.get('/api/getmcuid/:mcuid', (req, res)=> {
19
20   (async() => {
21
22     try{
23       const document = db.collection('NodeMCU').doc(req.params.mcuid);
24       const nodemcu = await document.get();
25       let response = nodemcu.data();
26       const userContact = response.Contact;
27       const secondaryContact = response.Secondary_Contact;
28       const securityContact = response.Security_Contact;
29       const flatNumber = response.Flat_Number;
30       const normalMsg = "Someone opened the door. Please check";
31       const securityGurardMsg = "Unauthorized entry detected at flat " + flatNumber + " Please check";
32       const send_alarm = response.Send_Alarm;
33       if(send_alarm){
34         sendSMS(userContact, normalMsg);
35         sendSMS(secondaryContact, normalMsg);
36         sendSMS(securityContact, securityGurardMsg);
37       }
38       return res.status(200).send(response);
39     }
40     catch(error){
41       console.log(error);
42       return res.status(500).send(error);
43     }
44   })();
45 }
46
47 function sendSMS(contact, message){
48
49   const url = "http://bhashsms.com/api/sendmsg.php?user=9971930997&pass=95c48ab&sender=SYSINT&phone="
50   + contact + "&text=" + message;
51
52   const Http = new XMLHttpRequest();
53   Http.open("GET", url);
54   Http.send();
55   return true;
56 }
57
58
59
60 exports.app = functions.https.onRequest(app);

```

Program Code 2

Hosting the API on local server

At the terminal of visual studio code run command “npm run serve” it will allocate a web URL where the service will be hosted.

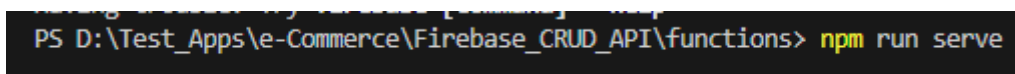


Fig. 7 Run Command

After executing above command, we get the following URL where service is hosted-

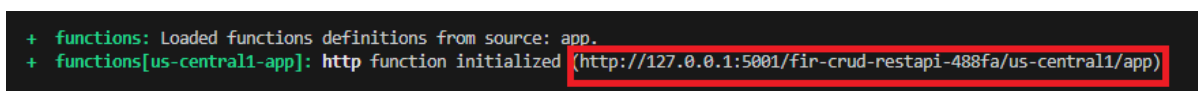


Fig. 8 Service Hosted URL

Testing our API on POSTMAN

About POSTMAN

Postman is a popular API development and testing tool designed to simplify the process of interacting with and managing APIs. It provides a user-friendly interface for sending HTTP requests, examining responses, and automating testing workflows [35].

Key Features:

- **Request Building:** Users can construct and send various types of HTTP requests (GET, POST, PUT, DELETE, etc.) with customizable headers, parameters, and body content.

- **Response Viewing:** Postman displays responses in a readable format, including status codes, response bodies, and headers, which helps in debugging and validating API functionality.
- **Collections:** Users can organize requests into collections for easy management and sharing. Collections can be exported and imported, facilitating collaboration and version control.
- **Environment Management:** Postman supports environment variables, allowing users to manage different configurations and switch between environments (e.g., development, staging, production) seamlessly.
- **Automation:** The tool supports automated testing with the ability to write scripts in JavaScript to test responses, validate data, and handle pre-request logic.
- **Documentation:** Postman can generate API documentation from collections, making it easier to share API details and usage instructions with teams or clients.

Overall, Postman is a versatile tool that enhances API development, testing, and collaboration through its comprehensive features and intuitive interface.

Testing our API

As in above screenshot we saw our service is hosted at URL - `http://127.0.0.1:5001/fir-crud-restapi-488fa/us-central1/app`

Our Get ROUTE is defined as `/api/getmcuid`, with `mcuid` as a parameter henceforth our URL becomes-
`http://127.0.0.1:5001/fir-crud-restapi-488fa/us-central1/app/api/getmcuid/NMCU_01`

In our next step we will make a get request using POSTMAN –

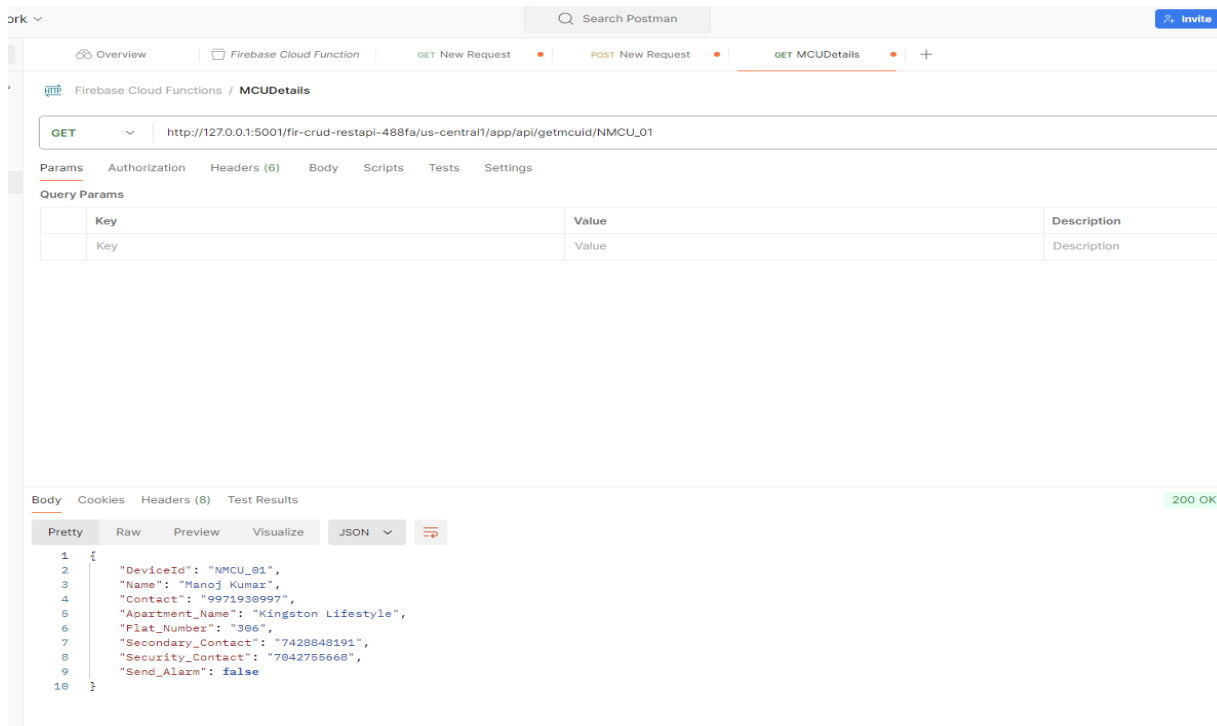


Fig. 9 Get Request Postman Output

As we can see we are getting all the required information in the response, henceforth confirming our API works fine.

In our next step we will call this API from Arduino code which will complete the application.

Deploying the API on cloud

Till now we have created the API on local. Firebase is an excellent container and can host our API on it's own cloud [36].

We need to issue “Firebase Deploy” command for terminal like below –

```
PS D:\Test_Apps\e-Commerce\Firebase_CRUD_API\functions> firebase deploy
```

Fig. 10 Deploy Command

This command will deploy our API at cloud and will provide as a web URL which we can use for our purpose. Once done we will get the message like below –

```
+ functions: functions folder uploaded successfully
+ firestore: released rules firestore.rules to cloud.firestore
i functions: creating Node.js 18 (1st Gen) function app(us-central1)...
+ functions[app(us-central1)]: Successful create operation
Function URL (app(us-central1)): https://us-central1-fir-crud-restapi-488fa.cloudfunctions.net/app
! functions: Cleaning up build files...
! functions: Unhandled error cleaning up build images. This could result in a small monthly bill if not
hem manually at https://console.cloud.google.com/gcr/images/fir-crud-restapi-488fa/us/gcf
+ Deploy complete!
Project Console: https://console.firebase.google.com/project/fir-crud-restapi-488fa/overview
```

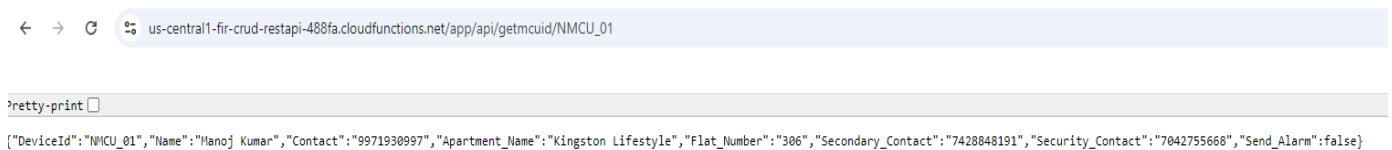
Fig. 11 Output Message After Deployment

So as we can see in above screenshot the URL for our API is - <https://us-central1-fir-crud-restapi-488fa.cloudfunctions.net/app>

Since our route for GET is `api/getmcuid/{mcu_id}` our final URL for get request becomes –

https://us-central1-fir-crud-restapi-488fa.cloudfunctions.net/app/api/getmcuid/NMCU_01

We can hit this URL on web browser to check if we can verify our results –



```
["DeviceId":"NMCU_01","Name":"Manoj Kumar","Contact":"9971930997","Apartment_Name":"Kingston Lifestyle","Flat_Number":"306","Secondary_Contact":"7428848191","Security_Contact":"7042755668","Send_Alarm":false}
```

Fig. 12 Output Data About Controller Id NMCU_01

We can see hitting the URL gives us the data about controller id NMCU_01.

Putting Things Together :

1. Now we have our API ready which is capable to send SMS to the users as defined in firestore database.
2. We have our hardware ready with proper connections.
3. We have our Arduino program ready which is capable of glowing LED when distance of the object from the distance sensor is less than 10 cms.

Now we need to modify our existing Arduino program and make it capable to call the Firebase API

Below is the updated code:

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include <WiFiClient.h>
#define echoPin D2
#define triggerPin D4
#define ledPin D0

#define WIFI_SSID "ManojKumar4g"
#define WIFI_PASSWORD "9971930997"
#define MCU_ID = "NMCU_01"

WiFiClient wifiClient;
HTTPClient httpClient;
long duration, distance;
void setup() {
  Serial.begin(9600);
  pinMode(triggerPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(ledPin, OUTPUT); // Initialize the LED_BUILTIN pin as an output
  Serial.begin(9600);
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);

  Serial.print("connecting");
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
  }

  Serial.println();
  Serial.print("connected: ");
  Serial.println(WiFi.localIP());
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(triggerPin, LOW); // Turn the LED on (Note that LOW is the voltage level
  delayMicroseconds(2);
  digitalWrite(triggerPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(triggerPin, LOW);

  duration = pulseIn(echoPin, HIGH);
  distance = duration / 58.2;
  String disp = String(distance);

  Serial.print("Distance :");
  Serial.print(disp);
  Serial.print(" cm\n");

  if(distance < 10){
    //Glow the LED
    digitalWrite(ledPin, HIGH);

    String serverPath = "https://us-centrall-fir-crud-restapi-488fa.cloudfunctions.net/app/api/getmcuid/" + MCU_ID;
    httpClient.begin(wifiClient, serverPath.c_str());

    //Send HTTP Get Request
    int httpResponseCode = http.GET();
    if(httpResponseCode > 0){
      Serial.print("HTTP Response Code: ");
      Serial.println(httpResponseCode);
      String payload = http.getString();
      Serial.println(payload);
    }
    else{
      Serial.print("Error Code: ");
      Serial.println(httpResponseCode);
    }
  }
  else{
    digitalWrite(ledPin, LOW);
  }
  delay(1000); // Wait for a second
}
```

Program Code 3

Explanation of code-

1. The Echo pin of the distance sensor is connected to the D2 pin on the NodeMCU, and is therefore designated as the Echo Pin.
2. The Trigger pin of the distance sensor is connected to the D4 pin on the NodeMCU, and is thus designated as the Trigger Pin.
3. An LED is connected to the D0 port on the NodeMCU. The LED will be turned on when an object is within 10 centimetres.
4. The Wi-Fi network name and password are stored in variables named WIFI_SSID and WIFI_PASSWORD, respectively.
5. Global variables for duration, distance, WifiClient and HTTP Client

6. The setup function initializes the Arduino settings. In this function:
 - The NodeMCU's baud rate is set to 9600 bps (bytes per second), which determines the data transfer rate.
 - The Trigger pin is configured as an output.
 - The Echo pin is set as an input.
 - The status method of the Wi-Fi class is called to connect to the Wi-Fi network.
7. The loop function continuously executes in a cycle.
8. In the loop function:
 - The Trigger pin is first set to LOW.
 - After a 2-millisecond delay, the Trigger pin is set to HIGH, generating an ultrasonic pulse.
 - After 10 milliseconds, the Trigger pin is set back to LOW.
 - The duration for the ultrasonic pulse to return to the Echo pin is measured.
 - This duration is then converted into a distance measurement in centimetres.
 - If the calculated distance is less than 10 centimetres, the LED is turned on.
9. If distance is less than 10 cms
 - Invoke the API with board ID. In our case NMCU_01
 - Turn ON the LED

RESULTS AND DISCUSSIONS

The suggested technology may be used in real time by Internet of Things-based home monitoring devices. Smart homes are built with an integrated architecture of sensors, cameras, and other cutting-edge technology; this research will examine unlawful entry into such homes. The system operates on two levels: hardware and software.

This technique was created to avoid flat theft while the owners are gone. After receiving the SMS, the owner could notify the security guard to conduct a security check at the flat. Because a security camera is a relatively unknown thing, robbers may be unaware of the system and will continue to ransack, increasing their chances of being caught.

In this work the SMS is received generally within 5 seconds of object detection. To avoid false positives, we can configure in Firebase when to send SMS. For example, configure it to send only when no one is there or the house is locked.

Future Scope

1. As of now, we have manually created the entries in the database for this project, but this can be done by using a web interface.
2. A dedicated App can be created for users where they can see the logs of all the activities.
3. App can have the feature to turn off the flag which in turn will not send SMS. As of now we are manually updating it.

CONCLUSION

This system will be relatively inexpensive, with component costs ranging from 400 to 500 rupees, and it has the potential to be industrialized with simple tweaks, providing huge benefits to the corporation and peace of mind to users. Such systems may find use outside of their initial setting as well.

1. Heat control alarm systems (used when a chamber's temperature shouldn't rise over a specified point). The program should be able to force a shutdown if this happens.
2. Water waste management systems: Managers in high-volume water-use communities may monitor the overflow from the comfort of a single location utilizing a centralized system comprised of a water sensor, node MCU modules, and a set of corresponding applications.
3. Automatic Irrigation: Farmers may also use this idea when they have to schedule their irrigation activities perfectly. A Node MCU, motors, relay switches, and an app are all components that may be used to create a functional analogue. The farmer may begin or stop irrigation with the touch of a button on his smartphone.
4. Distance detection helps in vehicle accidents by determining the separation between the driving vehicle and the vehicle in front of it. These methods rely on three main tools: deep learning theory, laser sensors, and conventional computer vision theory. In the conventional computer vision-based, vanishing point identification, vehicle segmentation, particle filtering, and road detection are used to identify the presence of the preceding vehicle and determine the distance.

REFERENCES

1. P. Timse, P. Aggarwal, P. Sinha, N. Vora, Face recognition based door lock system using opencv and C# with remote access and security features, *Int. J. Eng. Res. Appl.* 4 (2014) 52–57.
2. Zhang, L., Zhou, H., Kong, R., & Yang, F. (2005, September). An improved approach to security and privacy of RFID application system. In *Proceedings. 2005 International Conference on Wireless Communications, Networking and Mobile Computing, 2005.* (Vol. 2, pp. 1195-1198). IEEE.
3. Bagchi, T., Mahapatra, A., Yadav, D., Mishra, D., Pandey, A., Chandrasekhar, P., & Kumar, A. (2022). Intelligent security system based on face recognition and IoT. *Materials Today: Proceedings*.
4. T. Gunawan, M.H.H. Gani, F. Rahman, M. Kartiwi, Development of face recognition on raspberry pi for security enhancement of smart home system, *Indon. J. Electr. Eng. Inform.* (2017).
5. Mocrii, D., Chen, Y., & Musilek, P. (2018). IoT-based smart homes: A review of system architecture, software, communications, privacy, and security. *Internet of Things*, 1, 81-98.
6. V. Kushnir, B. Koman, V. Yuzevych, IoT image recognition system implementation for blind peoples using esp32, mobile phone and convolutional neural network, in: *2019 XIth International Scientific and Practical Conference on Electronics and Information Technologies (ELIT)*, 2019, pp. 183–187.
7. Keshta, I. (2022). AI-driven IoT for smart health care: Security and privacy issues. *Informatics in Medicine Unlocked*, 30, 100903.
8. Sisavath, C., & Yu, L. (2021). Design and implementation of security system for smart home based on IOT technology. *Procedia Computer Science*, 183, 4-13.
9. Aldawira, C. R., Putra, H. W., Hanafiah, N., Surjarwo, S., & Wibisurya, A. (2019). Door security system for home monitoring based on ESP32. *Procedia Computer Science*, 157, 673-682.
10. Saxena, N., & Varshney, D. (2021). Smart Home Security Solutions using Facial Authentication and Speaker Recognition through Artificial Neural Networks. *International Journal of Cognitive Computing in Engineering*, 2, 154-164.
11. Mohan, J., & Rajesh, R. (2021). Enhancing home security through visual cryptography. *Microprocessors and Microsystems*, 80, 103355.
12. Zhu, Z., & Cheng, Y. (2020). Application of attitude tracking algorithm for face recognition based on OpenCV in the intelligent door lock. *Computer Communications*, 154, 390-397.

13. Prasad, S., & Rajaan, R. (2022). Intelligent security system based on distance recognition and Internet of Things. *International Journal of Emerging Technologies and Innovative Research*, 9(8), f621–f639. <http://www.jetir.org/papers/JETIR2208566.pdf>
14. Santos, R. (2015, January 4). Getting started with ESP8266 WiFi transceiver (Review). Random Nerd Tutorials. <https://randomnerdtutorials.com/getting-started-with-esp8266-wifi-transceiver-review/>
15. TechTarget. (n.d.). IoT security (Internet of Things security). IoT Agenda. <https://www.techtarget.com/iotagenda/definition/IoT-security-Internet-of-Things-security>
16. Kaspersky. (n.d.). What is cyber security? Kaspersky Resource Center. <https://www.kaspersky.com/resource-center/definitions/what-is-cyber-security>
17. Internet Society. (n.d.). Internet of Things (IoT). https://www.internetsociety.org/iot/?gad_source=1&gclid=CjwKCAjwodC2BhAHEiwAE67hJDSBr-5t4TitS1RhpS4M9jupHkzU_atOnqnRUAJTzBMEDnbljt2JxBoCnQwQAvD_BwE
18. Wattlecorp Cybersecurity Labs LLP. (n.d.). IoT security: How to protect your connected devices. <https://www.wattlecorp.com/iot-security/>
19. ProWriters Insurance. (n.d.). Why are unpatched vulnerabilities a serious business risk? ProWriters Insurance Cyber Insurance Blog. Retrieved June 12, 2025, from <https://prowritersins.com/cyber-insurance-blog/unpatched-vulnerability-risks/#:~:text=Unpatched%20vulnerabilities%20refer%20to%20security,unauthorized%20access%20to%20a%20network.>
20. Sasi, T., Lashkari, A. H., Lu, R., Xiong, P., & Iqbal, S. (2024). A comprehensive survey on IoT attacks: Taxonomy, detection mechanisms and challenges. *Computers & Security*. <https://doi.org/10.1016/j.cose.2023.103627>
21. Author(s). (2020). Title of paper. In *Journal Title* (Vol. 1712, Issue 1, Article 012009). IOP Publishing. <https://doi.org/10.1088/1742-6596/1712/1/012009>
22. APMonitor. (2024, November 1). Ultrasonic distance measurement HC-SR04. Retrieved June 12, 2025, from <https://apmonitor.com/dde/index.php/Main/UltrasonicDistanceSensor>
23. Arduino. (2024, March 27). Arduino IDE v1 basics. Retrieved June 12, 2025, from <https://docs.arduino.cc/software/ide-v1/tutorials/arduino-ide-v1-basics/>
24. Google Cloud. (2025, March 24). Firestore documentation. Retrieved June 12, 2025, from <https://firebase.google.com/docs/firestore>
25. Make-It.ca. (n.d.). NodeMCU ESP8266 specifications, overview and setting up. Retrieved June 12, 2025, from <https://www.make-it.ca/nodemcu-details-specifications/>
26. HowToMechatronics. (n.d.). Ultrasonic Sensor HC-SR04 and Arduino – Complete Guide. Retrieved June 12, 2025, from <https://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/>
27. Terrell Hanna, K., & Rosencrance, L. (2023, May). What is Google Firebase? SearchMobileComputing. Retrieved June 12, 2025, from <https://www.techtarget.com/searchmobilecomputing/definition/Google-Firebase>
28. IonicThemes. (n.d.). Mastering Firebase databases: Real-time and Cloud Firestore. Retrieved June 12, 2025, from <https://ionicthemes.com/tutorials/mastering-firebase-databases-real-time-and-cloud-firestore>
29. BhashSMS. (n.d.). BhashSMS – Bulk SMS & WhatsApp Business API provider in India. Retrieved June 12, 2025, from <https://bhashsms.com/>
30. nitya10h3vn. (n.d.). Introduction to Firebase Hosting. GeeksforGeeks. Retrieved June 12, 2025, from <https://www.geeksforgeeks.org/introduction-to-firebase-hosting/>
31. EITCA Academy. (2023, August 3). What are the main features and use cases of Cloud Firestore and Cloud Bigtable? EITCA Academy. Retrieved June 12, 2025, from <https://en.eitca.org/cloud-computing/eitc-cl-gcp-google-cloud-platform/gcp-overview/gcp-data-and-storage-overview/examination-review-gcp-data-and-storage-overview/what-are-the-main-features-and-use-cases-of-cloud-firestore-and-cloud-bigtable/>
32. Firebase. (2025, March 4). Choose a Database: Cloud Firestore or Realtime Database. Retrieved June 12, 2025, from <https://firebase.google.com/docs/database/rtdb-vs-firestore>

33. Google Cloud. (2025, March 26). Quickstart: Create a Firestore database by using a server client library. Retrieved June 12, 2025, from <https://cloud.google.com/firestore/docs/create-database-server-client-library>
34. Google Cloud. (n.d.). Cloud Firestore overview. Retrieved June 12, 2025, from <https://cloud.google.com/firestore?hl=en>
35. Postman. (2024, May 1). Test APIs and write scripts in Postman. Retrieved June 12, 2025, from <https://learning.postman.com/docs/tests-and-scripts/tests-and-scripts/>
36. Google Cloud. (n.d.). Deploy to your site using the Firebase Hosting REST API. Retrieved June 12, 2025, from <https://firebase.google.com/docs/hosting/api-deploy>