

**University of Gafsa**  
**Higher Institute of Applied Sciences and Technologies of Gafsa (HIASTG)**

**Department of computer sciences**



**Title**

**Maison intelligente**

Presented by:

**Amine FRIDHI**

Presented for the purpose of obtaining a

**Bachelor's Degree in embedded systems & IOT**

Under the direction of:

Academic supervisor (HIAST Gafsa): **Mr. Ahmed KHLIFI**

Professional supervisor (ESI): **Mr. Wajdi MELKI**

Defended on .. /06/2023

Before the jury composed of:

**President :**

**Reviewer :**

**Members :**

**2022/2023**

# DEDICATED

## To

*With great pleasure I dedicate this project.*

*To those who helped and encouraged me.*

*To my parents FRIDHI Abd El Majid, ASSILI El Wazna and*

*my little sister Esrae who supported me.*

*To my brother Aymen who helped me in the realization*

*my big sister Amal who believed in me.*

*To my cousin Montaha who helped me during my practical realization.*

*To all my friends who were always present with me.*

*To all those who I love and who try to create this environment in which I*

*live, full of joy and favorable atmosphere.*

# ACKNOWLEDGMENTS

*F*irst, I want to thank "GOD" for giving me courage and patience during my studies.

Secondly, I would like to express my gratitude to my supervisor, **Mr. Ahmed KHLIFI**, a teacher at HIAST GAFSA for his precious help, his quality supervision and his directives which have been of great help to me in fulfilling my Project Graduation.

I would also like to express my sincere thanks to **Mr. Wajdi MALKI**, supervisor at the "ESI" enterprise, and to all those who contributed from near or far to the good progress and the realization of my end of studies project inside or outside from the "ESI" enterprise.

I would like to express my sincere gratitude to all the teachers and supervisors who have supported and guided me throughout these three years. Your dedication and commitment to my education have been invaluable, and I am truly grateful for the knowledge and skills I have gained under your guidance.

Furthermore, I would like to extend a special thank you to Mrs. Fatma HRIZI, the director of the computer science department. Your willingness to provide a conducive environment for this project has been instrumental in its success. I appreciate your support and encouragement throughout the process.

I did not want to leave this opportunity without expressing my heartfelt thanks to the director of the Higher Institute of Applied Sciences and Technologies of Gafsa who gave me the chance to discover the companies and improve my information.

*F*inally, I express my thanks, the most devoted, to **the president of the jury**:

..... and the jury members: ....., .....

..... for accepting the judgment of my modest end of studies project.

# Table of contents

<b>Introduction .....</b>	<b>2</b>
<b>CHAPTER I: Technical Background.....</b>	<b>5</b>
1. Introduction .....	5
2. Project specifications.....	5
3. Economic study .....	6
4. Internet of Things (IoT).....	6
5. Arduino board .....	8
6. Sensors Used .....	12
6.1. Gas sensor MQ-2.....	12
6.2. Temperature and Humidity sensor DHT11 .....	15
6.3. infrared receiver sensor VS1838B .....	17
6.4. IR LED .....	18
7. Conclusion.....	19
<b>CHAPTER II: Design and Development .....</b>	<b>20</b>
1. Introduction .....	20
2. User Interface Design.....	20
3. Database Design .....	24
3.1. Firestore database.....	25
3.2. Realtime database .....	27
4. Application Architecture .....	28
5. Programming Languages used .....	29
5.1. Mobile development .....	30
5.2. Arduino development.....	31
6. conclusion.....	32
<b>CHAPTER III: Features.....</b>	<b>33</b>
1. Introduction .....	33
2. Login and Signup Pages .....	33
2.1. Signup page.....	33
2.2. Login page .....	37
3. Dashboard.....	38
3.1. Shortcuts .....	40
3.2. Room Control.....	41

1.1. Device Control.....	42
2. QR CODE .....	44
3. Settings .....	45
3.1. Edit profile .....	46
3.2. Security .....	48
4. Notification System.....	48
5. Conclusion.....	50
<b>CHAPTER IV: Experiment and result .....</b>	<b>51</b>
1. Introduction .....	51
2. Arduino board setup .....	51
2.1. Simulation .....	52
3. Excremental.....	53
4. Result.....	58
5. Database integration .....	60
6. conclusion.....	62
<b>Discussion .....</b>	<b>63</b>
<b>GENERAL CONCLUSION .....</b>	<b>64</b>

## List of figures

<b>Figure 1: Arduino uno R3 board [1]</b> .....	9
<b>Figure 2: Arduino SHIELDS [1]</b> .....	10
<b>Figure 3: ESP32 [10]</b> .....	11
<b>Figure 4: MQ-2 GAS SENSOR [2]</b> .....	13
<b>Figure 5: The connection of MQ-2 with Arduino Uno [2]</b> .....	14
<b>Figure 6: The connection of MQ-2 ESP32 [2]</b> .....	14
<b>Figure 7: The connection of DHT11 with Arduino Uno [2]</b> .....	16
<b>Figure 8: The connection of DHT11 with ESP32 [2]</b> .....	16
<b>Figure 9: The connection of VS1838B with Arduino Uno [2]</b> .....	17
<b>Figure 10: The connection of VS1838B with ESP32 [2]</b> .....	18
<b>Figure 11: The connection of IR LED with Arduino Uno [2]</b> .....	19
<b>Figure 12: The connection of IR LED with ESP32 [2]</b> .....	19
<b>Figure 13: light and dark version of the dashboard screen</b> .....	22
<b>Figure 14: submit buttons used</b> .....	22
<b>Figure 15: switches used</b> .....	23
<b>Figure 16: gesture detector used like a buttons</b> .....	23
<b>Figure 17: dark mode navigation bar</b> .....	23
<b>Figure 18: light mode navigation bar</b> .....	24
<b>Figure 19: Firebase using REST API communication [3]</b> .....	25
<b>Figure 20: Firebase authentication methods [4]</b> .....	26
<b>Figure 21: Next home authentication database</b> .....	26
<b>Figure 22: NextHome authentication methods used</b> .....	27
<b>Figure 23: sensors value in Realtime data base</b> .....	28
<b>Figure 24: screen A</b> .....	34
<b>Figure 25: screen B</b> .....	34
<b>Figure 26: screen C</b> .....	34
<b>Figure 27: error message</b> .....	37
<b>Figure 28: light mode</b> .....	39
<b>Figure 29: dark mode</b> .....	39
<b>Figure 30: doorbell light mode</b> .....	41
<b>Figure 31: doorbell dark mode</b> .....	41
<b>Figure 32: different rooms</b> .....	42

<b>Figure 33: light tv remote</b>	43
<b>Figure 34:light AC remote</b>	43
<b>Figure 35: light ceiling fan</b>	43
<b>Figure 36: Ultraloq Latch 5 Fingerprint [7]</b>	44
<b>Figure 37: Chamberlain MyQ-G0401 [8]</b>	44
<b>Figure 38: Light QR scanner</b>	45
<b>Figure 39: dark QR scanner</b>	45
<b>Figure 40: edit profile picture</b>	47
<b>Figure 41: profile picture in the settings screen</b>	47
<b>Figure 42: profile picture in the dashboard</b>	47
<b>Figure 43: security options</b>	48
<b>Figure 44: Gas sensor warning</b>	49
<b>Figure 45: Doorbell notification</b>	50
<b>Figure 46: Arduino simulation [9]</b>	53
<b>Figure 47: linking DHT11 with ESP32</b>	53
<b>Figure 48: adding the mq-2</b>	54
<b>Figure 49: adding relay</b>	55
<b>Figure 50: adding button</b>	56
<b>Figure 51: adding the esp32-cam</b>	57
<b>Figure 52: esp32-cam</b>	57
<b>Figure 53: serial monitor of the Arduino IDE</b>	58
<b>Figure 54: the result saved in the firebase</b>	59
<b>Figure 55: Integration using key and API</b>	61
<b>Figure 56: google-services.json download link</b>	61
<b>Figure 57: location where to put the JSON file</b>	62

# List of Tables

<b>5. Table 1: Economic study .....</b>	<b>6</b>
---	----------



# Introduction

Next Home is a mobile app that lets you control your home appliances and security systems using your smartphone. It's easy to use and works on both Android and iOS devices. You can install it easily on your phone.

To use the app, you need to create an account and log in. Once you're logged in, you can control various devices in your home like lights, TVs, air conditioners, and more. The app uses infrared technology to communicate with these devices, so you can control them from your phone even when you're not at home.

Next Home also has security sensors that keep an eye on your home for any dangers or intruders. These sensors can detect things like temperature changes, humidity, gas leaks, and movement. If any of these sensors detect a problem, the app sends you a notification on your phone.

The app has a dashboard that shows you the current status of your devices and sensors, along with any alerts or notifications. You can control each room in your home separately, adjusting the temperature, turning lights on or off, and controlling other connected devices.

All your data is stored on a secure server in the cloud. The server uses encryption and other security measures to protect your information and keep it private.

In conclusion, Next Home is a powerful and user-friendly app that gives you complete control over your home appliances and security. It's perfect for anyone who wants to automate and secure their home easily.

Next Home is not only a versatile mobile app for controlling home appliances and security systems, but it also addresses two major challenges in the market. Firstly, it tackles the issue of high product costs by providing an affordable solution for homeowners. Many existing products in the market come with a hefty price tag, making home automation and security systems inaccessible for some individuals. Next Home offers a cost-effective alternative without compromising functionality or quality.

Secondly, Next Home overcomes the limitation of separate applications for Android and iOS devices. Traditionally, developers had to create separate applications for each platform, resulting in inconsistencies and differences between the Android and iOS versions. This posed a challenge for users who wanted a unified experience across different devices. However, Next Home leverages the power of Flutter, a cross-platform framework, to build a single application

that works seamlessly on both Android and iOS devices. This approach ensures consistent features, user interface, and functionality, regardless of the operating system.

By utilizing Flutter's cross-platform capabilities, Next Home eliminates the need for separate development teams and reduces the overall development time and costs. This approach allows users to enjoy a unified experience on their preferred devices, whether it's an Android smartphone or an iOS device like an iPhone or iPad.

Furthermore, Next Home leverages the advantages of Flutter's native performance and features. While other products rely on native programming languages for their respective platforms, Next Home harnesses the power of Flutter's rich set of pre-built UI components, extensive libraries, and seamless integration with device features. This enables Next Home to deliver a robust and consistent user experience across different platforms, ensuring that users can control their home appliances and security systems effortlessly, regardless of the device they are using.

In summary, Next Home revolutionizes the home automation and security market by offering an affordable solution while providing a unified cross-platform experience. With its cost-effective approach and utilization of Flutter's capabilities, Next Home enables users to control their home appliances and security systems easily, making it an ideal choice for homeowners seeking a reliable, user-friendly, and budget-friendly solution.

### **Objectives**

- Develop a user-friendly mobile application that enables remote control of home appliances and security systems.
- Create a cross-platform application that is compatible with both Android and iOS devices.
- Integrate a wide range of home devices, including lights, televisions, air conditioners, and other appliances, into the application's control system.
- Incorporate infrared technology into the application to allow users to control their devices remotely.
- Integrate security sensors such as temperature, humidity, gas, and motion sensors into the application, providing real-time notifications and alerts for potential risks or intruders.
- Ensure a seamless user experience by designing an intuitive and easy-to-navigate interface.

- Secure the cloud-based server using encryption and other security measures to protect user data and ensure privacy.
- Ensure compliance with relevant regulations and standards, such as data privacy laws and home security regulations.
- Provide excellent customer service and support to users, addressing any concerns or issues they may have with the application.

These objectives guide the development of the Next Home application, ensuring that it meets the needs of users while delivering a high-quality and reliable home automation and security solution.

# CHAPTER I: Technical Background

## 1. Introduction

The Next Home project: A Flutter-Based Application for Remotely Controlling Home Appliances and Security Systems via Arduino Board and Sensors," specializing in embedded systems & IoT. The project aims to design a user-friendly mobile application using Flutter, which will enable users to control their home appliances and security systems remotely. The application will be connected to Arduino boards and various sensors placed throughout the home, allowing for real-time monitoring and control of different parameters. This chapter provides a technical background of the project, including project specifications, an economic study, an overview of the Internet of Things (IoT), an introduction to Arduino boards, and an explanation of the sensors used.

## 2. Project specifications

**Subject:** Maison Intelligent Next Home: A Flutter-Based Application for Remotely Controlling Home Appliances and Security Systems via Arduino Board and Sensors

**Institution:** Higher Institute of Applied Sciences and Technologies Studies of Gafsa (HIASTG)

**Department:** Department of computer sciences.

**Option:** embedded systems & IOT

**Class:** ISI3

**Number of students:** 01

**Supervised by:** Mr. Ahmed KHLIFI (HIAST GAFSA)

**Period:** 3 Months (from February 14 to May 15, 2023).

**Work to do:**

1. Design UI/UX
2. Coding the application
3. Coding the Arduino
4. Connect to Arduino
5. Create device controls
6. Add security sensors
7. Test and debug

### 3. Economic study

Components and reference	Quantity	Price
Esp32-cam	1 unit	35.000 DT
Esp-32	1 unit	38.000 DT
Arduino UNO R33	1 unit	69.000 DT
Esp-01	1 unit	8.000DT
Fan	1 unit	10.000 DT
DHT11	1 unit	14.000 DT
MQ-2	1 unit	8.000 DT
Button	1 unit	1.000 DT
relay	1 unit	12.000 DT
Wires jumpers male / male 20cm Lot of 40 threads	1 unit	8.000 DT
Wires jumpers male / female 20cm Lot of 40 threads	1 unit	8.000 DT
Wires jumpers female / female 20cm Lot of 40 threads	1 unit	8.000 DT

4.

### 5. Table 1: Economic study

#### 4. Internet of Things (IoT)

The Internet of Things (IoT) is a technology that connects physical devices and enables them to communicate with each other through the internet. The concept of IoT has been around for decades, but it was only with the advent of wireless communication and micro-electromechanical systems (MEMS) that it became practical. The history of IoT dates back to the early 1980s when a group of researchers at Carnegie Mellon University connected a Coca-Cola vending machine to the internet. This was one of the first examples of a physical object being connected to the internet.

The Arduino is an open-source electronics platform based on easy-to-use hardware and software. It was introduced in 2005 and has since become a popular choice for building IoT applications. Arduino boards are designed to be easily programmable, even for those without a background in electronics. They are also relatively cheap, making them accessible to hobbyists and professionals alike.

The ESP32 is a low-cost Wi-Fi microchip with full TCP/IP stack and microcontroller capability produced by Espressif Systems. It was first introduced in 2014 and has since become a popular choice for building IoT applications that require wireless connectivity. The ESP32 can be programmed using the Arduino IDE, which makes it easy for developers to create IoT projects with Wi-Fi connectivity.

In our project, the NextHome mobile application, we use Arduino boards and ESP32 modules to connect physical devices in a home to the internet. The Arduino boards are connected to sensors and actuators that can monitor and control various aspects of a home, such as temperature, humidity, and lighting. The ESP32 modules are used to provide wireless connectivity to the Arduino boards, enabling users to remotely monitor and control their homes using the NextHome mobile application.

The application is designed to communicate with an Arduino board, which is connected to various sensors throughout the home. The sensors collect data on different parameters, such as temperature, humidity, and gas levels, and send this data to the Arduino board, which processes it and sends it to the application via the internet.

The application then uses this data to provide users with real-time information on the state of their home and allows them to remotely control various devices, such as lights, TV, and AC, using the app's interface. For example, a user can turn off a light or adjust the temperature of their AC from anywhere, as long as they have an internet connection.

Furthermore, the application uses IoT to enhance the security of the home by sending notifications to the user's smartphone when the sensors detect any potential risks, such as a gas leak or a break-in attempt. This allows the user to take immediate action and prevent any potential harm to themselves or their property.

In summary, your Nexthome application utilizes the power of IoT to provide users with a seamless and convenient way to monitor and control their homes, while also enhancing the safety and security of their property.

## 5. Arduino board

Arduino is an open-source electronics platform based on easy-to-use hardware and software. It was first introduced in 2005, designed to provide a simple and affordable way for non-engineers to create projects that interact with their environment using sensors and actuators. The platform is composed of an integrated development environment (IDE) that runs on a computer, and a microcontroller board that can be programmed to read inputs from sensors and control a variety of actuators such as LEDs, motors, and servos.

The most commonly used Arduino board is the Arduino Uno, which is based on the ATmega328P microcontroller. It has 14 digital input/output pins, 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, and a reset button. The board can be programmed using the Arduino IDE, which is a free, open-source software that supports C and C++ programming languages.

There are also many other types of Arduino boards available, each with their own features and capabilities. For example, the Arduino Mega has more input/output pins, while the Arduino Nano is smaller and more compact. Additionally, there are Arduino boards designed for specific purposes, such as the Arduino Yun, which is designed for Internet of Things (IoT) applications.

Arduino boards are widely used in IoT projects because they are easy to program and can interface with a wide range of sensors and actuators. They can be used to create smart homes, wearable devices, robots, and more. Arduino boards can communicate with other devices using a variety of communication protocols, such as Bluetooth, Wi-Fi, and Ethernet.

Overall, Arduino is a powerful tool for building IoT projects, providing a flexible and affordable platform for creating a wide range of projects that interact with the world around us.

- The Arduino Uno, which is one of the most popular Arduino boards, was first released in 2010. It is based on the ATmega328P microcontroller and has 14 digital input/output pins, 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, and a reset button.

- Since the release of the Uno, there have been many other Arduino boards that have been developed, including the Leonardo, Mega, Due, and Nano, among others. Each board has different specifications and capabilities depending on the intended use.
- Arduino boards are programmed using the Arduino IDE (Integrated Development Environment), which is an open-source software platform that provides a simple interface for writing, compiling, and uploading code to the board.
- One of the key features of Arduino boards is their ability to interact with a wide range of sensors and other electronic components, making them ideal for use in IoT projects. These sensors can include temperature sensors, humidity sensors, motion sensors, light sensors, and many others.
- Arduino boards are also often used in conjunction with other IoT technologies, such as Wi-Fi and Bluetooth, in order to enable remote control and monitoring of connected devices. For example, an Arduino board connected to a Wi-Fi module can be used to control smart home devices like lights, thermostats, and door locks.
- Finally, the open-source nature of the Arduino platform has led to a large community of developers and enthusiasts who have created a vast library of code and resources that can be used to build a wide range of IoT projects.



**Figure 1: Arduino uno R3 board [1]**

Arduino shields are modules that can be attached on top of an Arduino board to provide additional functionality or features. They usually include a variety of components and interfaces, such as sensors, actuators, communication ports, displays, and more.



There are many types of shields available for the Arduino Uno, each designed for a specific purpose. Some of the most common types of shields include:

1. Ethernet Shield: This shield allows the Arduino to connect to the internet or local network via an Ethernet cable.
2. Wi-Fi Shield: This shield adds Wi-Fi connectivity to the Arduino board, allowing it to connect to wireless networks.
3. Motor Shield: This shield provides the Arduino with the ability to control DC motors, stepper motors, and servo motors.
4. LCD Shield: This shield allows the Arduino to display information on an LCD screen.
5. Sensor Shield: This shield includes a variety of sensors, such as temperature, humidity, light, and sound sensors.
6. Relay Shield: This shield allows the Arduino to control high-voltage and high-current devices, such as lamps, motors, and solenoids.
7. Bluetooth Shield: This shield enables the Arduino to communicate wirelessly with other Bluetooth-enabled devices.

Shields are typically designed to be compatible with specific Arduino boards, and can be easily attached to the board's headers. They can be stacked on top of each other to add multiple functionalities to the same board.

Using shields can greatly simplify the process of building complex projects with the Arduino Uno, as it eliminates the need to wire up individual components and sensors. Additionally, since shields are pre-designed and tested, they can help reduce development time and effort.



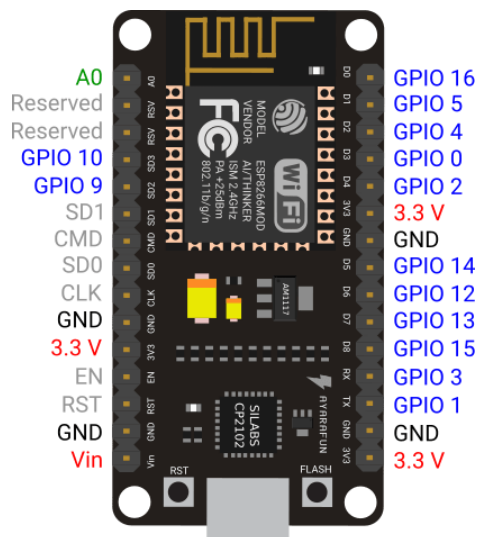
**Figure 2: Arduino SHIELDS [1]**

One of the key reasons for choosing the ESP32 over the Arduino Uno for the Next Home project is its built-in internet connectivity. Unlike the Arduino Uno, which lacks native internet capabilities, the ESP32 comes with built-in Wi-Fi functionality, allowing it to connect to the internet seamlessly. This internet access is crucial for the Next Home project as it enables remote control and monitoring of home appliances and security systems through the mobile application.

While it is possible to establish internet connectivity with the Arduino Uno using additional shields or modules like the ESP01, the process can be complex and challenging. It requires additional wiring, configuration, and software setup to enable internet connectivity. This not only adds complexity to the project but also increases the cost and potential for errors.

By utilizing the ESP32, the Next Home project can simplify the development process and provide a more streamlined solution. The ESP32's native Wi-Fi capabilities eliminate the need for additional components, reducing complexity and potential points of failure. It allows for direct integration with the Next Home mobile application, enabling seamless communication between the app and the connected devices.

Overall, the ESP32 offers a more efficient and reliable solution for internet connectivity in the Next Home project compared to the Arduino Uno. Its built-in Wi-Fi capabilities simplify the development process, enhance the user experience, and ensure seamless integration with the Next Home mobile application.



**Figure 3: esp32 [10]**

## 6. Sensors Used

Sensors are important devices used in technology that can detect and measure things in the physical world. They convert this information into electrical signals. In the Internet of Things (IoT), sensors play a big role by collecting data from the environment. This data helps us make decisions and automate tasks. [2]

There are different types of sensors that measure different things like temperature, humidity, pressure, light, sound, and movement. Each sensor is made to detect a specific thing. For example, temperature sensors measure how hot or cold it is, and humidity sensors measure how damp the air is.

Sensors are used in IoT to collect data about the world around us. This data can be analyzed to learn things and make decisions. By using sensors, IoT systems can keep track of things, make our lives safer, and use energy more efficiently.

In IoT systems, sensors are connected to small computers that can process and send data. These computers take the signals from the sensors, change them into digital information, and send them to a central system or the cloud. This data can then be looked at and used for different things.

The data collected from sensors helps us make better choices, work smarter, and create new ideas. For example, in smart homes, sensors can tell if someone is inside, control the lights and temperature, and keep us safe. In factories, sensors can watch machines, find problems, and make things run better.

Overall, sensors are very important in IoT. They help change industries and make smart environments. By gathering data and giving us information, they help us use technology in new and useful ways.

### 6.1. Gas sensor MQ-2

The MQ-2 gas sensor is a type of semiconductor sensor that can detect a variety of gases such as methane, propane, butane, alcohol, smoke and more. It works by heating a sensing element and measuring the change in resistance as the gas interacts with the heated material. The MQ-2 gas sensor has a high sensitivity to flammable gases and is commonly use in gas leak detection systems and smoke detectors.



**Figure 4: MQ-2 GAS SENSOR [2]**

The MQ-2 gas sensor has a simple analog output that can be read by a microcontroller such as the Arduino. The analog output voltage is proportional to the gas concentration detected by the sensor. The sensor can be powered with a voltage between 5V to 15V and consumes very low power.

One important thing to note is that the MQ-2 gas sensor has some limitations. It is not suitable for measuring the exact concentration of a gas and can only give a rough estimate of the gas level. The sensor can also be affected by other factors such as humidity and temperature which can cause false readings.

To use the MQ-2 gas sensor with an Arduino, a simple circuit can be built using a voltage divider to convert the analog output voltage to a usable range for the Arduino's analog input. The MQ-2 gas sensor can be connected to an analog input pin of the Arduino, and the Arduino can read the analog value using the `analogRead()` function. The Arduino can then use this value to trigger alarms or send notifications when a certain gas concentration is detected.

The MQ-2 gas sensor can be connected to an Arduino Uno or an ESP32 module using a few simple steps.

First, the power and ground pins of the sensor need to be connected to the corresponding power and ground pins of the board. The sensor also has an analog output pin which can be connected to any of the analog input pins of the board.

To obtain an accurate reading from the sensor, a voltage divider circuit is usually used. This circuit consists of two resistors, one with a known value and the other with an unknown value, connected in series between the sensor's analog output pin and ground. The voltage across the

unknown resistor is then measured and used to calculate the concentration of the gas being detected.

It is important to note that the MQ-2 sensor requires a warm-up time of at least 24 hours before it can provide accurate readings. Additionally, the sensor can be affected by other gases present in the environment, which can result in false readings. To overcome this, calibration of the sensor may be necessary, which involves exposing the sensor to known concentrations of the gas being detected and adjusting the voltage divider circuit accordingly.

With Arduino uno:

### MQ2 Sensor with Arduino UNO

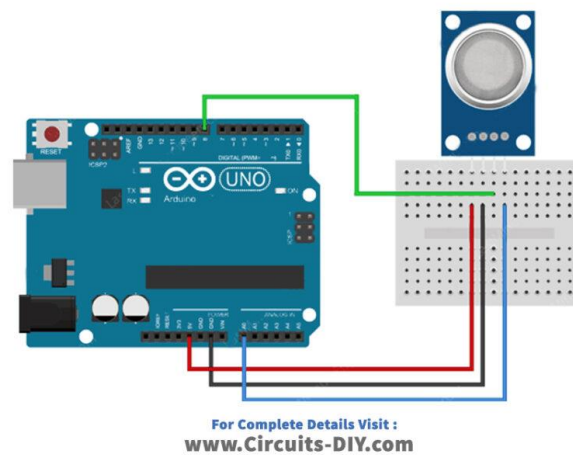


Figure 5: The connection of MQ-2 with Arduino Uno [2]

With ESP32:

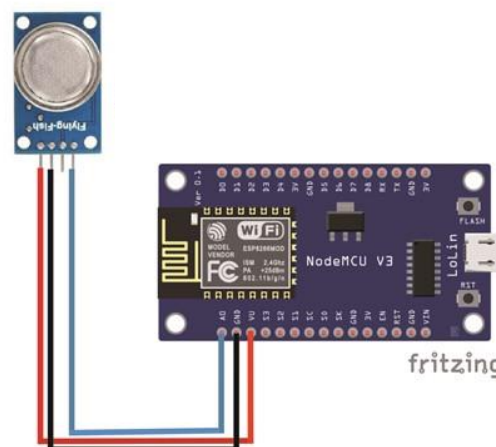


Figure 6: The connection of MQ-2 ESP32 [2]

## 6.2. Temperature and Humidity sensor DHT11

The DHT11 sensor is a low-cost digital temperature and humidity sensor that is widely used in various applications. It has a single-wire digital interface that can communicate with microcontrollers or other devices with digital inputs. The sensor is easy to use and does not require any external components to measure temperature and humidity.

The sensor consists of a capacitive humidity sensor and a thermistor-based temperature sensor. It converts the temperature and humidity values into digital signals that can be read by a microcontroller. The DHT11 sensor has a measurement range of 0-50°C for temperature and 20-90% RH for humidity with an accuracy of  $\pm 2^{\circ}\text{C}$  and  $\pm 5\%$  RH, respectively.

The sensor requires a pull-up resistor of 5-10 kohm to provide a stable output signal. It also requires a delay time of at least 1 second between readings to avoid incorrect readings due to temperature and humidity fluctuations. The DHT11 sensor is commonly used in home automation systems, weather stations, and other applications where temperature and humidity monitoring are necessary.

To use the DHT11 sensor with Arduino Uno, you need to follow these steps:

1. Connect the VCC pin of the DHT11 to the 5V pin of the Arduino Uno.
2. Connect the GND pin of the DHT11 to the GND pin of the Arduino Uno.
3. Connect the DATA pin of the DHT11 to any digital pin of the Arduino Uno (for example, pin 2).
4. In your Arduino sketch, you need to include the DHT11 library and define the pin to which the DATA pin of the sensor is connected.
5. You can then use the functions provided by the library to read the temperature and humidity values from the sensor.

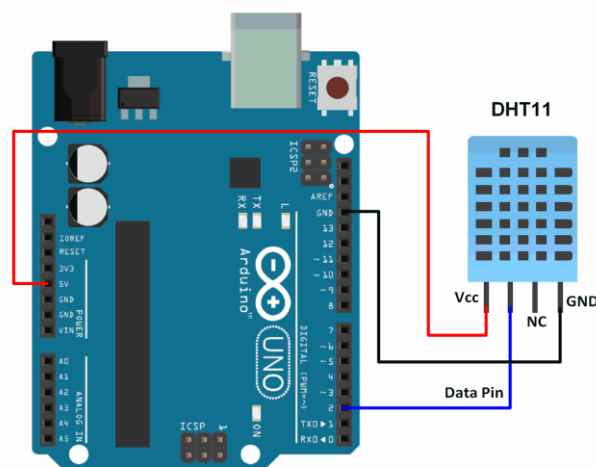
To use the DHT11 sensor with ESP32, the wiring is the same as with Arduino Uno. You can follow the above steps to connect the sensor to the ESP32, and use the DHT11 library in your ESP32 sketch to read the temperature and humidity values from the sensor.

One thing to note is that the DHT11 sensor is a digital sensor, so it requires a digital pin to communicate with the microcontroller (either Arduino or ESP32). The data communication between the sensor and the microcontroller is done using a single-wire serial communication

protocol. The DHT11 library takes care of this communication protocol, so you don't have to worry about it in your code.

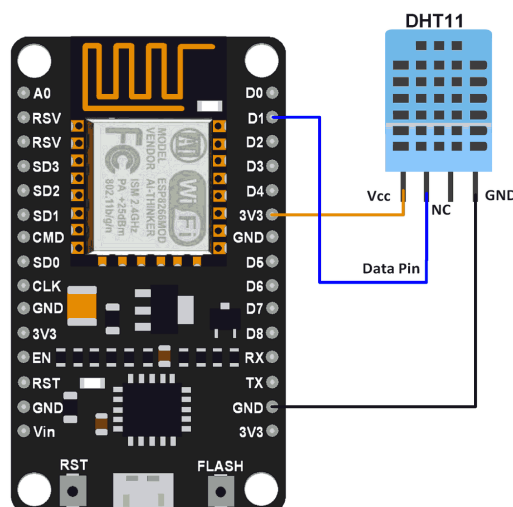
In your NextHome application, you can use the DHT11 sensor to monitor the temperature and humidity levels in your home. For example, you can display the current temperature and humidity values on the app, and send notifications to the user if the values exceed a certain threshold. You can also use the sensor data to control the temperature and humidity levels using the air conditioning system or other home automation devices.

With Arduino uno:



**Figure 7: The connection of DHT11 with Arduino Uno [2]**

With ESP32:



**Figure 8: The connection of DHT11 with ESP32 [2]**

### 6.3. infrared receiver senser VS1838B

The VS1838B is an infrared (IR) receiver module that can be used to receive signals from IR remote controls. It operates at a carrier frequency of 38kHz and is compatible with most IR remote controls. The module consists of a photo-diode, an amplifier, and a demodulator circuit. When an IR signal is received, the photo-diode converts the IR light into an electrical signal which is then amplified and demodulated to retrieve the original signal.

The VS1838B can be used in a variety of applications, including home automation systems, remote control systems, and robotics. It is often used in conjunction with an IR transmitter module to create a wireless remote-control system. The module can also be used to receive signals from other IR devices, such as TVs, DVD players, and set-top boxes.

The VS1838B is typically connected to a microcontroller, such as an Arduino or ESP32, which can decode the received signals and perform appropriate actions. The module requires a 5V power supply and can be connected to a microcontroller using a digital input pin. When a signal is received, the module outputs a digital signal that can be read by the microcontroller. The module also has a built-in IR filter to eliminate noise from ambient light sources.

With Arduino uno:

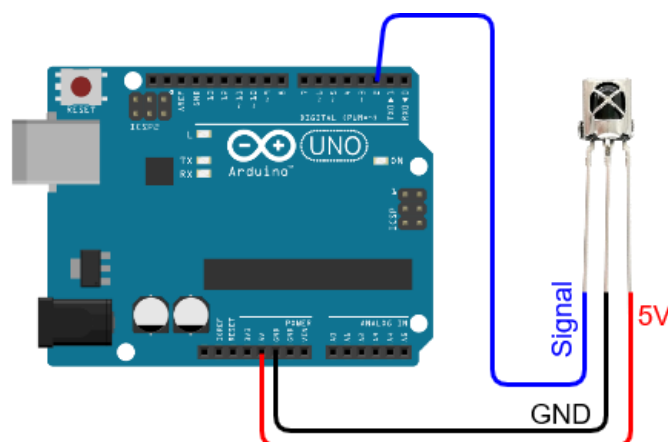
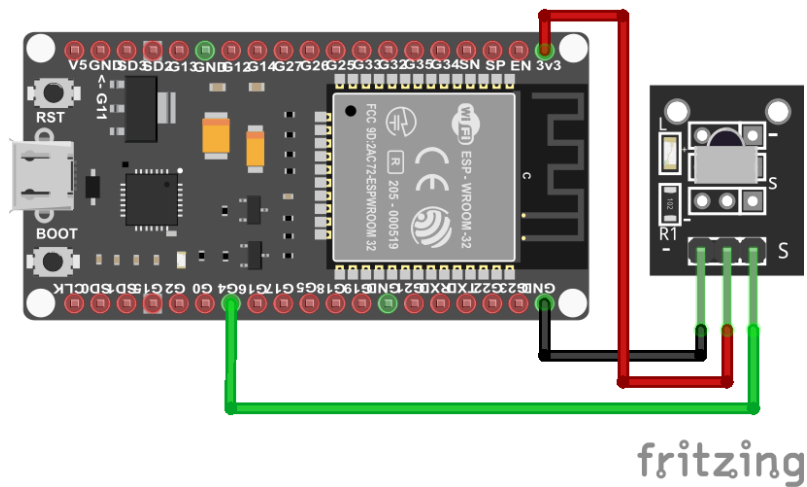


Figure 9: The connection of VS1838B with Arduino Uno [2]



with ESP32:



**Figure 10: The connection of VS1838B with ESP32 [2]**

#### 6.4.IR LED

An IR LED is a special type of LED that emits infrared radiation instead of visible light. It can be used to transmit information via infrared signals, such as in remote controls for televisions or other electronic devices. These IR signals are not visible to the human eye, but can be detected by an IR receiver. IR LEDs have a narrow beam angle and are typically used in combination with a lens to focus the beam. They require a power source and a current-limiting resistor to operate properly. IR LEDs can be used in a wide range of applications, such as surveillance systems, communication systems, and medical devices.

An IR LED can be easily controlled by an Arduino or ESP32 microcontroller. The microcontroller sends a series of on/off signals to the IR LED, which then emits IR light in the form of pulses. These pulses are typically modulated to carry a specific command or data. For example, a TV remote control might send a series of modulated IR pulses to turn on the TV, change the channel, or adjust the volume.

To control an IR LED with an Arduino or ESP32, you typically need to connect the LED to a digital output pin on the microcontroller. Then, you can use software to control the pin and send the appropriate signals to the LED. Many libraries and examples are available for controlling IR LEDs with these microcontrollers, making it easy to get started with IR communication. Additionally, many microcontroller development boards come with built-in IR LEDs and receivers, further simplifying the process.

With Arduino uno:

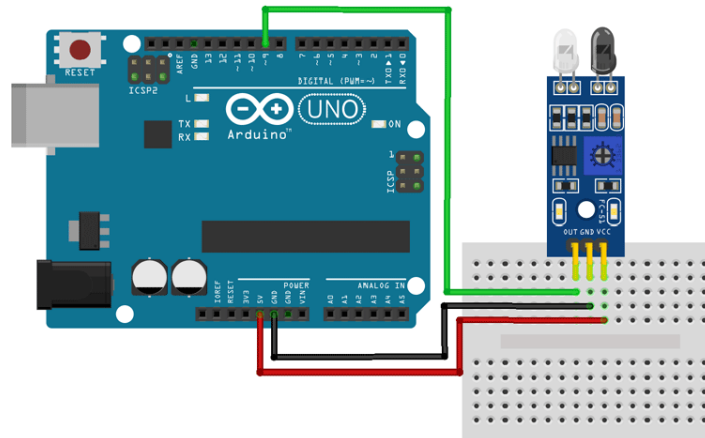


Figure 11: The connection of IR LED with Arduino Uno [2]

With ESP32:

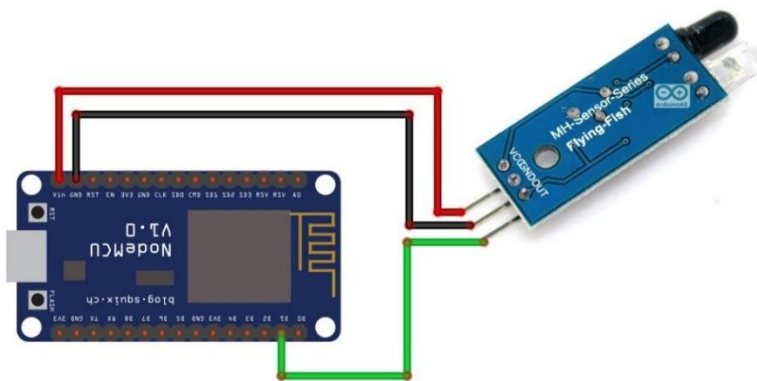


Figure 12: The connection of IR LED with ESP32 [2]

## 7. Conclusion

In conclusion, the technical background chapter of the Next Home project has laid the foundation for understanding the project's objectives and components. We have learned about the importance of IoT in connecting physical devices and enabling communication over the internet. Arduino boards have been introduced as a powerful platform for building IoT applications, offering programmability, affordability, and ease of use. Additionally, the chapter provided insights into the specific sensors used in the project, such as the MQ-2 gas sensor for detecting flammable gases and the DHT11 sensor for measuring temperature and humidity. This technical background sets the stage for the subsequent chapters, where the design, implementation, and testing of the Next Home application will be explored in detail.

## CHAPTER II: Design and Development

### 1. Introduction

In this chapter on Design and Development, we will discuss the importance of a good design for the success of the NextHome application. Specifically, we will focus on User Interface Design and Database Design. A good design not only makes the app easier to use but also enhances its appeal and increases its chances of success. Let's explore how these aspects contribute to creating a user-friendly and efficient application.

### 2. User Interface Design

A good design is important for the success of an application like NextHome. When we focus on the user interface design, we make it easier for users to use the app and understand its features. This means they can do what they want to do more easily. For NextHome, the design is clean and looks nice, so users can control their home devices with just a few taps on their phone. The icons and text are easy to understand, which helps users know what each thing does.

Having a good design also helps with sales and making the app successful. When an app looks good and is easy to use, more people will want to use it. If users have a good experience, they will tell other people about the app, which can bring in more sales. A good design can also make the app stand out from other similar apps. When there are many options to choose from, a nice and easy-to-use design can make people choose your app instead of others.

By focusing on a good design, we are making the app better for users and increasing its chances of success. It makes it easier for users to do what they want to do, and it can attract more people to use and recommend the app. A good design is not just about making things work, but also about making them look and feel good, so users enjoy using the app and it does well in the market.

The user interface design is an important aspect of any mobile application, as it determines how users interact with the application and how the information is presented. For the NextHome application, the user interface design aims to provide a user-friendly and intuitive interface that allows users to easily control their home devices remotely.

The design process started by identifying the key features that the application will offer, including controlling the lights, TV, air conditioning, and monitoring home security sensors.

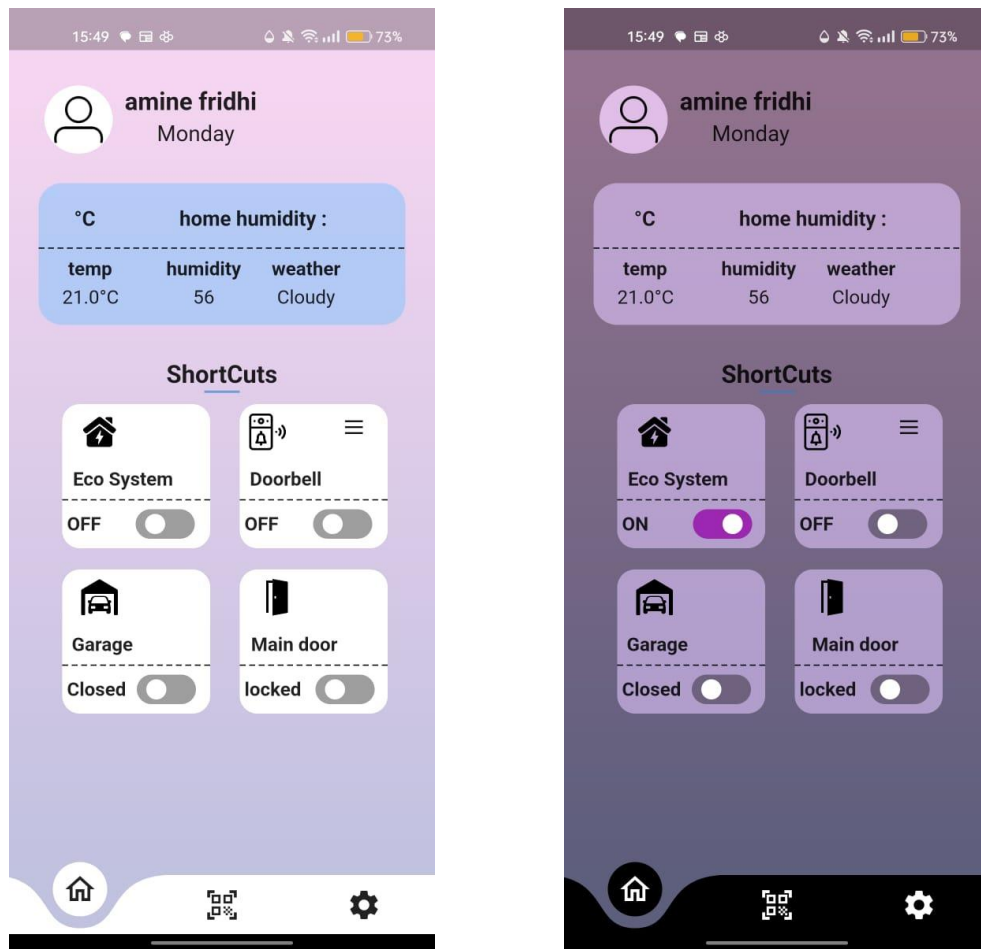
From there, wireframes and prototypes were created to visualize the interface and identify any usability issues.

The final user interface design of the NextHome application is clean, simple, and visually appealing. The home screen displays all the connected devices, and users can easily tap on each device to control it. The application also allows users to set up schedules for their devices, such as turning on the lights at a certain time or adjusting the temperature in the evening.

In terms of the visual design, the application uses a color scheme that is consistent with the NextHome brand, with shades of blue and white. The icons used to represent each device are simple and easy to understand, and the typography is clear and legible.

Overall, the user interface design of the NextHome application is designed to provide a seamless and intuitive experience for users, allowing them to easily control their home devices remotely with just a few taps on their mobile device.

The user interface (UI) design of an application plays a crucial role in providing an intuitive and engaging experience for users. In your NextHome mobile application, you have used a well-crafted color scheme to create a visually appealing interface. The dark mode version uses a gradient of pink and purple shades, which are calming and soothing to the eyes. The black background enhances the contrast between the text and other elements, making them more visible and easier to read. On the other hand, the light version of the application uses a white background that gives a sense of spaciousness and cleanliness. The use of white color signifies purity and clarity, which aligns with the idea of making your home a safe and secure place. The font size and style are consistent across both versions, making it easier for users to navigate and interact with the application. Overall, your UI design choices have created a harmonious and well-balanced interface that is aesthetically pleasing and user-friendly. The color scheme and background choice convey the mood and purpose of the application, while the font and layout ensure a smooth and intuitive user experience.



**Figure 13: light and dark version of the dashboard screen**

In addition to the color scheme, I also implemented buttons and switches in the user interface design to enhance the user experience. The buttons are used to submit information or trigger actions, such as saving changes or sending commands to the devices in the user's home. I made sure to use clear and concise labels for the buttons to minimize confusion and make it easy for users to understand what action they are triggering.



**Figure 14: submit buttons used**

The switches, on the other hand, are used to control the on/off state of various devices in the home, such as lights or air conditioners. They are designed to be easily toggled on or off with

a single tap, and their state is clearly displayed on the screen. Additionally, I made sure to use intuitive icons for each switch to further aid in user understanding.



**Figure 15: switches used**

In your user interface design, you are using the GestureDetector widget as a button to handle user interactions. The GestureDetector widget in Flutter allows you to detect various gestures like taps, swipes, long presses, and more. By using the GestureDetector widget, you can create custom buttons with more interactive features than a standard button widget. For example, you can add a `onLongPress` callback to detect when the user presses and holds on the button for an extended period. You can also add a `onDoubleTap` callback to detect when the user double-taps on the button. This widget is an efficient way to handle user gestures and interactions in your application. By utilizing the GestureDetector widget as a button, you can create a more engaging and user-friendly interface for your users.



**Figure 16: gesture detector used like a buttons**

The use of black icons in the Nexthome application's user interface design is a deliberate choice that is aimed at creating a visually appealing and modern look. Black is a neutral color that is often associated with elegance, and simplicity. By using black icons, the application's interface is given a sleek and stylish appearance that is both functional and easy to use.



**Figure 17: dark mode navigation bar**

On the other hand, in the light version of the application, the navigation bar's icons are white to contrast against the white background. This choice was made to ensure that the icons are easily

visible and legible, providing a better user experience for those who use the application in a well-lit environment.



**Figure 18: light mode navigation bar**

### 3. Database Design

When it comes to designing your application's database, choosing the right database management system is very important. In your case, you have chosen Firebase, which is a popular database management system that works on the internet. Firebase allows you to store and sync data in real-time across many devices, making it a great choice for mobile and web applications. One of the benefits of Firebase is that it takes care of managing the servers and infrastructure, so you can focus on developing your application.

With Firebase, you can store different types of user data in your application, like login information, home automation settings, and sensor data. Firebase has a system for verifying user identities, and you can save their data in the real-time database. This ensures that users always have the most up-to-date and synchronized data on all their devices, giving them a smooth experience.

Using Firebase for managing your database gives you scalability and reliability for storing and getting data in your application. You don't have to worry about managing servers and infrastructure because Firebase handles it for you. This lets you concentrate on creating a strong and feature-rich application.

Moreover, Firebase works well with Flutter, a popular mobile development framework. When you combine Firebase with Flutter, you can create applications that respond quickly and have dynamic user experiences. Firebase's real-time capabilities allow data changes to be instantly synchronized across connected devices. When there's a change in the database, events are triggered and quickly sent to all connected users. This means that users of your Flutter app connected to Firebase will see live updates right away, making the app smoother and more interactive.

By using Firebase's real-time database and integrating it with Flutter, you empower yourself as a developer to build applications with real-time updates, seamless synchronization, and

improved user experiences. Whether you're making a chat app, a tool for working together, or a live dashboard, the combination of Firebase's real-time features with Flutter gives you many possibilities to create dynamic and responsive apps.

In conclusion, by utilizing Firebase's real-time database and its integration with Flutter, you can create a powerful and efficient database design for your application. This design enables real-time updates and provides users with a smooth experience.

### 3.1.Firestore database

In the Database Design section, it's important to discuss the database management system that you are using to store and retrieve data. In my case, I am using Firebase, which is a popular cloud-based database management system. Firebase is a real-time database that allows you to store and synchronize data in real-time across multiple clients. It also provides an easy-to-use API for reading and writing data, which makes it a great choice for mobile and web applications. One of the advantages of using Firebase is that it's a cloud-based service, which means that you don't need to worry about managing servers or infrastructure. Firebase takes care of all the scaling and maintenance, so you can focus on building your application.

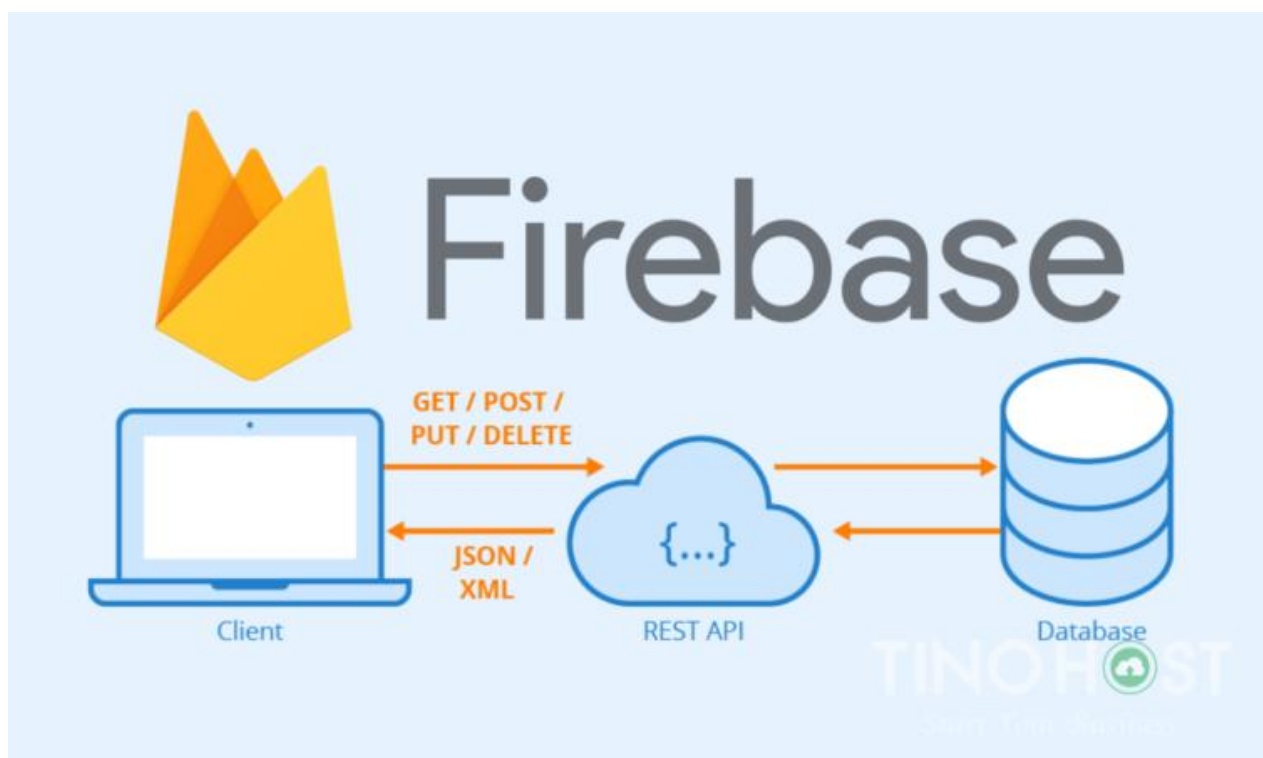


Figure 19: Firebase using REST API communication [3]



In my application, I am using Firebase to store and retrieve user data, such as login information, home automation settings, and sensor data. You can use Firebase's authentication system to authenticate users, and then store their data in Firebase's real-time database. This allows your application to provide a seamless experience for users, where their data is always up-to-date

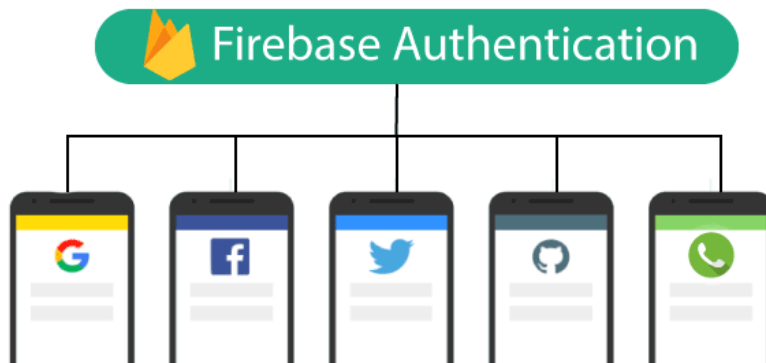


Figure 20: Firebase authentication methods [4]







and synchronized across all their devices. Overall, using Firebase for your database management needs provides a scalable and reliable solution for your application's data storage and retrieval requirements.

Already NextHome using firebase authentication

Identifiant	Fournisseurs	Date de création	Dernière connexion	UID utilisateur
aminefridhi@test.com		8 mai 2023		jUTMo4ZSPibc3GEpzvDYLHOuAxt1
test@test.com		7 mai 2023	7 mai 2023	VDPYX0S4RbMSJVhDK9n58gYN4...

Figure 21: Next home authentication database

And using social media login too

Fournisseurs de connexion		Ajouter un fournisseur
Fournisseur	État	
 Adresse e-mail/Mot de passe	 Activé	
 Google	 Activé	
 Facebook	 Activé	

**Figure 22: NextHome authentication methods used**

### 3.2.Realtime database

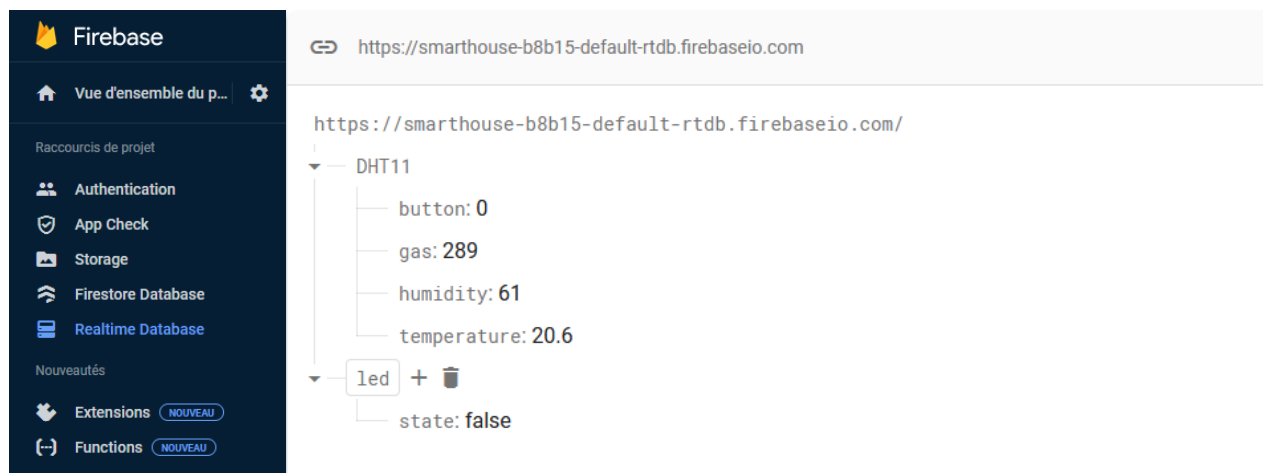
Realtime databases, such as Firebase, have revolutionized the way we design and build applications. With Firebase's integration in Flutter, developers can create highly responsive and dynamic user experiences.

Realtime functionality refers to the ability of a database to instantly propagate updates across all connected devices in real time. Unlike traditional databases where data retrieval is based on periodic queries, a real time database allows for instant synchronization of data changes.

Firebase's real time capabilities are achieved through a powerful event-driven architecture. Whenever a change occurs in the database, such as an update, insertion, or deletion, the database triggers events that are immediately propagated to all connected clients. This means that any user using a Flutter app connected to Firebase will receive live updates as soon as they occur.

In a Flutter application, this seamless integration with Firebase empowers developers to build engaging and interactive user interfaces. With just a few lines of code, developers can listen for database changes and update the user interface accordingly. This eliminates the need for manual refreshes or pulling data at regular intervals, resulting in a smoother and more responsive user experience.

Overall, the combination of Firebase's real time database and Flutter offers developers a powerful toolset to create applications that deliver real-time updates, seamless synchronization, and enhanced user experiences. Whether it's building a chat application, collaborative editing tool, or a live dashboard, the integration of real time Firebase with Flutter opens up a wide range of possibilities for developers to create dynamic and responsive apps.



**Figure 23: sensors value in Realtime data base**

## 4. Application Architecture

The application architecture of this project is based on the Flutter framework, which is an open-source mobile application development platform created by Google. Flutter provides an efficient way to develop high-performance, native interfaces for iOS and Android.

In this project, we are using two APIs to gather data. First, we are using Firebase API to store and retrieve user data. Firebase is a popular backend-as-a-service (BaaS) that provides features like real-time database, authentication, and storage. With Firebase, we can easily create and manage users' accounts and store their data securely.

Secondly, we are using weatherapi.com API to get data about the current weather. weatherapi.com provides accurate weather data for any location, which can be used to display weather information to users. By using this API, we can easily get the weather information and display it to the user in a user-friendly way.

The application is designed using the Model-View-Controller (MVC) architecture pattern, which separates the application logic into three distinct layers: the Model layer, the View layer, and the Controller layer. The Model layer is responsible for data management and communicates with Firebase to store and retrieve user data. The View layer is responsible for displaying data to the user and receiving user inputs. The Controller layer acts as an

intermediary between the Model and View layers, it handles user input and interacts with the Model to update the data.

In summary, the architecture of this application is designed to provide an efficient and user-friendly experience for users. By using Flutter and Firebase API, we can easily manage user data and provide real-time updates. And by using weatherapi.com API, we can get accurate weather data to display to the user.

## **5. Programming Languages used**

When it comes to mobile app development, one of the challenges is building separate apps for different platforms like Android with Java and iOS with Swift. However, Flutter offers a better solution. It is a versatile and efficient framework that allows developers to create mobile apps using a single codebase. This means that instead of writing separate code for each platform, developers can write once and deploy the app on both iOS and Android. This saves time and effort, making the development process easier and faster. Additionally, Flutter provides cross-platform compatibility, meaning the app will have the same look and feel on both iOS and Android devices. It offers a wide range of pre-built widgets and a fast development cycle with features like "hot reload" for quick code changes. Overall, Flutter simplifies mobile app development by providing a unified platform, reducing development time, and ensuring a consistent user experience across different platforms.

Similarly, when it comes to Arduino development, the primary programming language used is C/C++. Arduino, an open-source electronics platform, provides an integrated development environment (IDE) that supports C/C++ programming. There are several reasons why C/C++ is preferred over other languages for Arduino development. First, C/C++ is known for its efficiency and low-level programming capabilities, which is essential for working with Arduino boards that have limited resources. Second, C/C++ provides direct access to hardware components, allowing precise control and manipulation. Third, Arduino has a vast ecosystem of libraries and community support, with most libraries being written in C/C++, simplifying code integration and rapid prototyping. Fourth, C/C++ code written for Arduino is highly portable, making it adaptable to other platforms or microcontrollers. Additionally, C/C++ interfaces seamlessly with existing code and systems, providing integration capabilities and leveraging the extensive libraries and frameworks written in C/C++. Lastly, C/C++ offers advanced programming constructs and features, enabling developers to implement complex algorithms and optimizations. Overall, the use of C/C++ in Arduino development ensures

efficiency, hardware access, extensive libraries, portability, integration capabilities, and advanced programming features, making it a suitable choice for creating sophisticated projects within the constraints of embedded systems.

### **5.1. Mobile development**

Flutter is a popular programming language and framework for mobile app development, known for its versatility and efficiency. It was developed by Google and released in 2017. Flutter's main advantage lies in its ability to build high-performance applications with a single codebase that can run on multiple platforms, including iOS and Android. This cross-platform compatibility significantly reduces development time and effort compared to using separate programming languages for each platform.

One of the key reasons why Flutter gained widespread popularity is its reactive framework and its use of the Dart programming language. Dart is a modern, object-oriented language that provides a seamless development experience with Flutter. Dart's syntax is easy to learn and understand, making it accessible to developers from various backgrounds.

Flutter's underlying architecture is based on widgets, which are the building blocks of the user interface. These widgets are highly customizable and allow developers to create visually appealing and responsive apps. Flutter's "hot reload" feature enables developers to see instant changes in their app's interface during the development process, making it easier to iterate and refine the design.

In terms of performance, Flutter stands out because it doesn't rely on the platform's native UI components. Instead, it renders its own UI, resulting in faster app performance and smoother animations. Additionally, Flutter leverages the GPU (Graphics Processing Unit) for rendering, enhancing the overall speed and efficiency of the applications built with it.

Compared to other languages and frameworks used for mobile app development, Flutter offers several advantages. First and foremost, Flutter allows for code reuse across platforms, meaning developers can write a single codebase and deploy it on both iOS and Android platforms. This significantly reduces development time and effort, as developers don't need to write separate code for each platform or learn multiple programming languages.

Another advantage of Flutter is its vibrant and growing community. With an active community of developers, Flutter offers extensive documentation, libraries, and packages, which simplifies

the development process. The community also provides regular updates and support, ensuring that developers can stay up-to-date with the latest features and best practices.

Furthermore, Flutter's UI is designed to look native on each platform, ensuring a consistent and polished user experience. The framework provides pre-built widgets that resemble the platform-specific components, so the app feels familiar and intuitive to users, regardless of the device they are using.

In conclusion, Flutter is a powerful mobile development framework that offers significant advantages over other languages and frameworks. Its cross-platform compatibility, fast development cycle, excellent performance, and strong community support make it an excellent choice for building mobile applications. Whether you are an experienced developer or just starting, Flutter provides a versatile and efficient solution for creating beautiful and high-performance mobile apps.

## **5.2. Arduino development**

When it comes to Arduino development, the programming language primarily used is C or C++. Arduino, an open-source electronics platform, provides an integrated development environment (IDE) that supports C/C++ programming. Here are some benefits of using C/C++ in Arduino and why it is considered better than other languages for Arduino development:

1. **Efficiency:** C/C++ is known for its efficiency and low-level programming capabilities. Arduino boards have limited resources in terms of processing power and memory. C/C++ allows you to write code that can be optimized for these constraints, ensuring efficient execution and optimal resource utilization.
2. **Access to Hardware:** Arduino boards interact directly with hardware components such as sensors, motors, and other peripherals. C/C++ provides low-level access to the hardware, allowing you to control and manipulate these components more precisely. This level of control is essential for embedded systems development.
3. **Extensive Libraries:** Arduino has a vast ecosystem of libraries and community support. The majority of these libraries are written in C/C++, making it easier to integrate existing code and leverage the extensive functionality provided by the Arduino community. These libraries simplify complex tasks and allow for rapid prototyping.

4. Portability: C/C++ code written for Arduino is highly portable and can be easily adapted to other platforms or microcontrollers. This flexibility is beneficial if you plan to migrate your project to a different hardware platform or expand its capabilities beyond the Arduino environment.

5. Interfacing with Existing Code: C/C++ is widely used in the software industry, and many existing libraries, frameworks, and APIs are written in C/C++. If you need to interface your Arduino project with external software or systems, using C/C++ allows for seamless integration and communication.

6. Advanced Programming Constructs: C/C++ provides a wide range of programming constructs, including pointers, memory management, and low-level operations. These features enable developers to implement complex algorithms, data structures, and optimizations, making it suitable for advanced projects.

In summary, C/C++ is a suitable choice for Arduino development due to its efficiency, hardware access, extensive libraries, portability, integration capabilities, advanced programming constructs, and strong community support.

## **6. conclusion**

In conclusion, a well-designed user interface plays a vital role in ensuring a positive user experience and attracting more users to the NextHome application. The clean and visually appealing design, along with intuitive icons and text, makes it easier for users to control their home devices with just a few taps on their phone. Furthermore, the integration of Firebase as the database management system ensures real-time synchronization of data and a seamless experience for users across multiple devices. By combining the power of Flutter and Firebase, the NextHome application offers a dynamic and responsive user interface, making it a standout choice in the market. With these design and development considerations in place, NextHome is positioned for success in providing users with a convenient and enjoyable home automation experience.

## CHAPTER III: Features

### 1. Introduction

In this chapter, we will discuss the various features available in our application. These features include easy login and signup processes, a user-friendly dashboard, convenient shortcuts, control over rooms and devices, and a strong focus on data accuracy and security. Our aim is to provide users with a smooth and enjoyable experience. Let's explore these exciting features that make interaction effortless and ensure user satisfaction.

### 2. Login and Signup Pages

The login and signup pages in the application are designed to provide a user-friendly and intuitive experience. The signup page requires users to enter their first name, last name, email address, and password. If any required fields are left empty, an error message prompts users to fill in the missing information.

To prevent duplicate accounts, the system checks if the entered email address is already registered during the signup process. If a match is found, an error message informs the user to provide a different email address.

During the signup process, the password and its confirmation are validated to ensure they match. If they don't, an error message reminds users to double-check their entries.

In the login page, error messages are displayed when users enter incorrect email or password combinations. These messages provide feedback and guide users to troubleshoot their login attempts.

These features enhance user experience, promote data accuracy and security, and ensure a smooth and hassle-free interaction with the application.

#### 2.1.Signup page

The signup page in this application is designed to be user-friendly and intuitive. The user is required to enter their first name, last name, email address, and a password. If the user tries to submit the form without filling in any of the required fields, an error message will be displayed, instructing them to fill in the missing information. [6]



16:59 64%

**SIGN UP**

fridhi amine

Email

Password

Confirm password

**SIGN UN**

Already a member ?  
Sign in here ?

**Wrong**  
you can't sign up without email

Figure 24: screen A

17:00 64%

**SIGN UP**

fridhi amine

aminefridhi@test.com

Password

Confirm password

**SIGN UN**

Already a member ?  
Sign in here ?

**Wrong**  
this email already used

Figure 25: screen B

17:00 64%

**SIGN UP**

fridhi amine

aminefridhi@testtest.com

Password

Confirm password

**SIGN UN**

Already a member ?  
Sign in here ?

**Try again**  
Verify your password

Figure 26: screen C

### Screen A

it is important to ensure that users cannot sign up without providing all the necessary information or leaving any fields empty. This ensures that the user registration process is thorough and accurate. By requiring users to input all the required information, you can guarantee that the system has all the necessary data to create a complete user profile.

Requiring all fields to be filled serves multiple purposes. Firstly, it ensures that the user's identity is verified by collecting essential details such as their name, email address, and password. This helps to establish a unique user account and prevents unauthorized access. Secondly, by mandating the completion of all fields, you can gather valuable information about the user, such as their contact details, location, or preferences. This data can be used to personalize the user experience and provide tailored services or recommendations.

Additionally, by enforcing the completion of all fields, you can maintain data integrity and accuracy within the system. Incomplete or missing information may lead to errors, inconsistencies, or incomplete user profiles, which can affect the functionality and usability of the application. It is crucial to ensure that users provide all the required information to avoid any potential issues and provide a seamless experience.

Implementing validation checks and error messages is an effective way to communicate to users that all fields must be filled. These checks can be performed in real-time as users input their information, alerting them if any field is left empty or contains invalid data. This feedback prompts users to provide the necessary information and prevents them from proceeding with the sign-up process until all fields are properly filled.

By implementing this requirement for completing all fields during the sign-up process, you can ensure that user profiles are comprehensive, accurate, and in compliance with the system's requirements. This promotes data integrity, enhances security, and enables personalized experiences for users within the application.

### **Screen B**

In screen B, it is crucial to handle the scenario where a user tries to sign up with an email address that has already been used. To provide a seamless user experience and prevent duplicate accounts, the system should detect if the email is already registered and display an error message to the user.

When a user enters an email address during the sign-up process, the system needs to check if it is already associated with an existing account. If a match is found in the database, an error message should be displayed on the screen, indicating that the email address is already in use. This message should clearly inform the user that they need to provide a different email address to proceed with the sign-up process.

Displaying an error message serves two main purposes. Firstly, it notifies the user about the issue and prevents them from unintentionally creating duplicate accounts with the same email address. This helps maintain data integrity and ensures that each user has a unique identifier within the system. Secondly, it saves users' time and effort by immediately notifying them of the problem, allowing them to rectify it without going through the entire sign-up process.

To assist users in resolving the issue, the error message should be accompanied by clear instructions. For example, the message could suggest trying a different email address or provide guidance on how to recover a forgotten password if the user already has an account. By offering helpful suggestions or alternative solutions, users can easily understand what they need to do next to proceed successfully.

Implementing this error handling mechanism for duplicate email addresses enhances the user experience by providing prompt feedback and preventing potential frustrations. It ensures that each user has a unique identity within the system and avoids confusion or conflicts caused by multiple accounts using the same email address. By notifying users of the error and guiding them toward a solution, you can streamline the sign-up process and foster a positive user interaction with the application.

### **Screen C**

it is important to validate the password and its confirmation to ensure they match. This validation process helps users avoid errors and ensures that they enter the correct password during the sign-up process.

When a user enters their desired password and confirms it by re-entering it in the designated field, the system needs to compare the two values. If the password and its confirmation do not match, an error message should be displayed on the screen, informing the user that the password entries do not correspond.

Displaying this error message serves as a helpful reminder for users to double-check their password entries and ensure consistency. It prevents situations where users mistakenly enter different passwords, preventing them from successfully signing up or experiencing difficulties when trying to log in later.

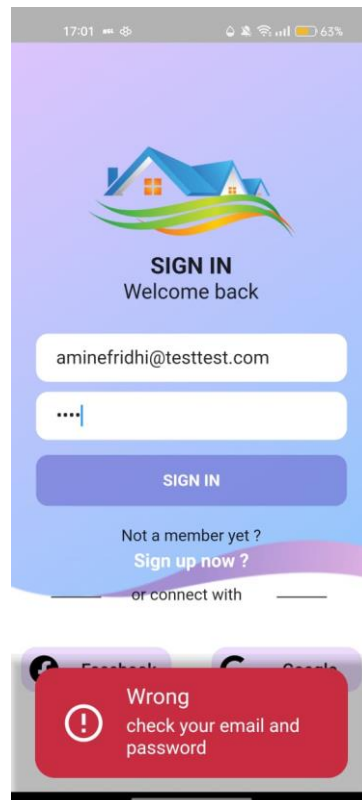
The error message should clearly state that the password and confirmation do not match and provide instructions on how to resolve the issue. It can suggest re-entering the password and confirmation fields, making sure to type the same password accurately in both places. Additionally, the message could offer password complexity guidelines, reminding users of any specific requirements for their password, such as minimum length or including certain characters.

By implementing this validation mechanism, the application promotes data accuracy and security. It encourages users to enter their password correctly and ensures they have confidence in the sign-up process. Providing clear instructions and assistance when an error occurs helps users quickly identify and rectify the issue, improving their overall experience with the application.

Overall, by checking and validating the password and its confirmation in screen C, the system enhances the sign-up process, reduces user errors, and promotes a smooth and hassle-free user experience

## 2.2. Login page

Same for the login page notifications when the email or password are wrong



**Figure 27: error message**

In the login page, it is crucial to provide error messages when users enter incorrect email or password combinations. These error messages serve as feedback to users, notifying them about the specific issue and helping them troubleshoot their login attempts.

When a user enters an incorrect email address or password, the system needs to identify this mismatch and display an appropriate error message on the screen. The error message should clearly state that the email or password entered is incorrect, indicating that there was a validation failure.

By displaying such error messages, users are immediately informed about the reason for their login failure. This feedback enables them to identify the issue and take the necessary steps to correct it. For example, if the user mistyped their email, they can double-check the entry and

make the necessary corrections. Similarly, if they entered the wrong password, they can re-enter it accurately.

The error message should be concise, easy to understand, and placed prominently on the screen, ensuring users can quickly notice and comprehend the feedback. It can include a simple instruction, such as "Please check your email and password and try again," providing guidance on how to rectify the error.

By providing clear error messages on the login page, the system helps users troubleshoot their login attempts effectively. It ensures that users receive immediate feedback when they enter incorrect email or password combinations, guiding them towards resolving the issue and successfully logging into their accounts. This approach contributes to a positive user experience by minimizing frustration and facilitating a smooth login process.

### **3. Dashboard**

For the dashboard section, I have designed a user-friendly interface that displays all the essential information that the user needs. The dashboard page consists of several widgets that are arranged in a clear and organized manner to provide a seamless user experience. The top section of the dashboard shows the current temperature and weather conditions for the user's location.

Below the weather information, there are several widgets that display the status of the user's devices, such as lights, fans, and AC units. These widgets are designed to be interactive, allowing the user to turn on/off their devices with a single tap.

Overall, the dashboard section provides a comprehensive overview of the user's smart home system and allows them to control their devices effortlessly.

Easy to use and read All the currently states

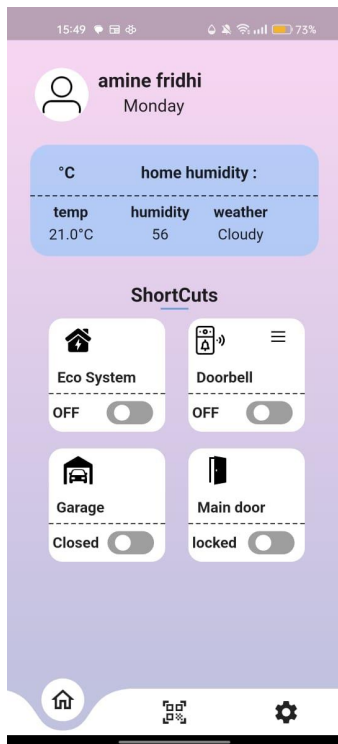


Figure 28: light mode

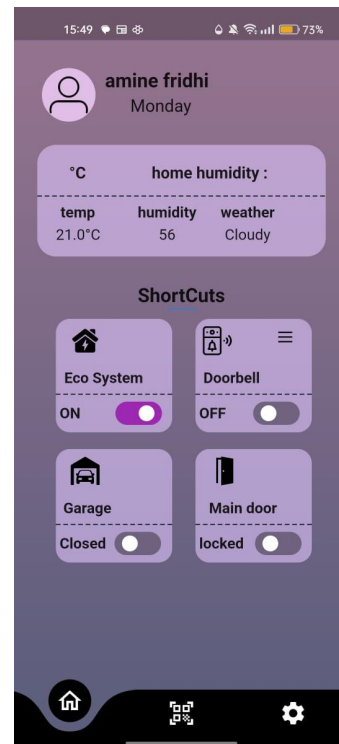


Figure 29: dark mode

On the dashboard page, there are multiple rooms or sections that you can access. One of these rooms is the kitchen, and another room is called shortcuts. The dashboard page serves as a central hub where you can navigate between different areas or functions of a system or application.

To move from the kitchen to shortcuts, you can use a sliding action. Sliding right or left usually refers to a horizontal movement gesture on a touch screen. By performing this sliding action, you can switch between the rooms on the dashboard page.

For example, if you're currently in the kitchen room and want to move to the shortcuts room, you would slide your from right to left. This action triggers a transition animation, and the content of the kitchen room slides out of view, revealing the shortcuts room. Similarly, if you want to go back to the kitchen from shortcuts, you would slide from left to right.

This sliding navigation provides a convenient and intuitive way to explore different sections of the dashboard without needing to open new pages or perform complex interactions. It allows for seamless transitions and efficient access to various functionalities or information contained within each room.

### 3.1. Shortcuts

Here u will find devices that you need fast to control like to lock or unlock your main door or garage fast to get out with your car you can active you eco system so automatically the system will turn of everything that u are not using like lights or devices consume energy and there is the Doorbell the idea of this smart doorbell when u press on it u will get notification in your phone sure if it working in your dashboard and you have three attentive after this u will start hearing the sound in house like simple doorbell and when u press on the section of the doorbell in the dashboard you will get redirected to page when u can watch the streaming of the camera of this device so you can know who outside before opening the door , In this device we used :

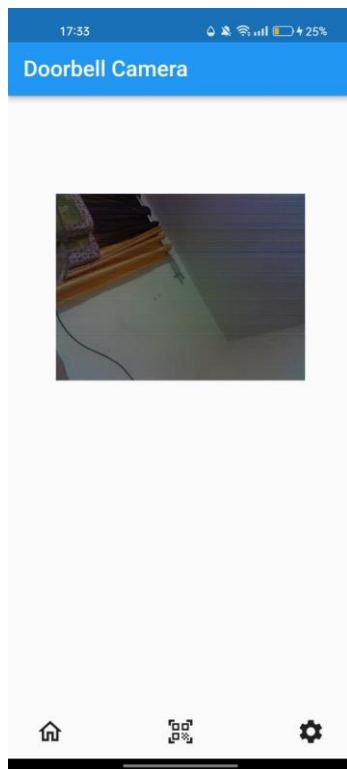
#### 3.1.1. Doorbell

Using the ESP32-CAM like a doorbell camera. [5]

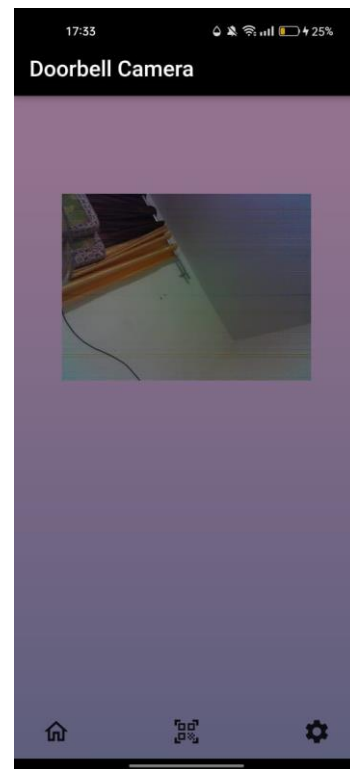
The ESP32-CAM: is a small-sized camera module built on the ESP32 chip. It has a compact design that integrates a high-performance image sensor, an ESP32-S2 or ESP32-S3 processor, and a Wi-Fi antenna. The module supports up to 2MP camera resolutions and can stream live video at up to 60 frames per second (fps). The ESP32-CAM module can be easily programmed using the Arduino IDE, which allows developers to build custom applications using the vast range of libraries and examples available online. It also supports Micro Python and ESP-IDF development platforms, making it easy for developers to program the module using their preferred programming language. The ESP32-CAM module has multiple interfaces, including UART, SPI, I2C, and I2S, which can be used to interface with other sensors and devices. It also has a built-in microSD card slot, which can be used to store images and video footage. The module comes with an OV2640 camera sensor, which has a resolution of 1600 x 1200 pixels. The camera sensor is capable of capturing high-quality images and video footage, making it suitable for a wide range of applications such as security systems, smart homes, and robotics. One of the key advantages of the ESP32-CAM module is its built-in Wi-Fi connectivity, which allows it to connect to the internet and stream live video over the network. This makes it easy to build wireless camera systems that can be accessed remotely from anywhere in the world.

Overall, the ESP32-CAM is a versatile and powerful camera module that can be used for a wide range of applications. Its compact design, built-in Wi-Fi connectivity, and support for multiple programming languages make it a popular choice among developers and hobbyists alike.

and here the result:



**Figure 30: doorbell light mode**



**Figure 31: doorbell dark mode**

### 3.2. Room Control

In the Room Control section of the application, you will find three different rooms: kitchen, sleeping room, and living room. Each room is equipped with its own set of smart devices that you can control with a simple tap on the screen.

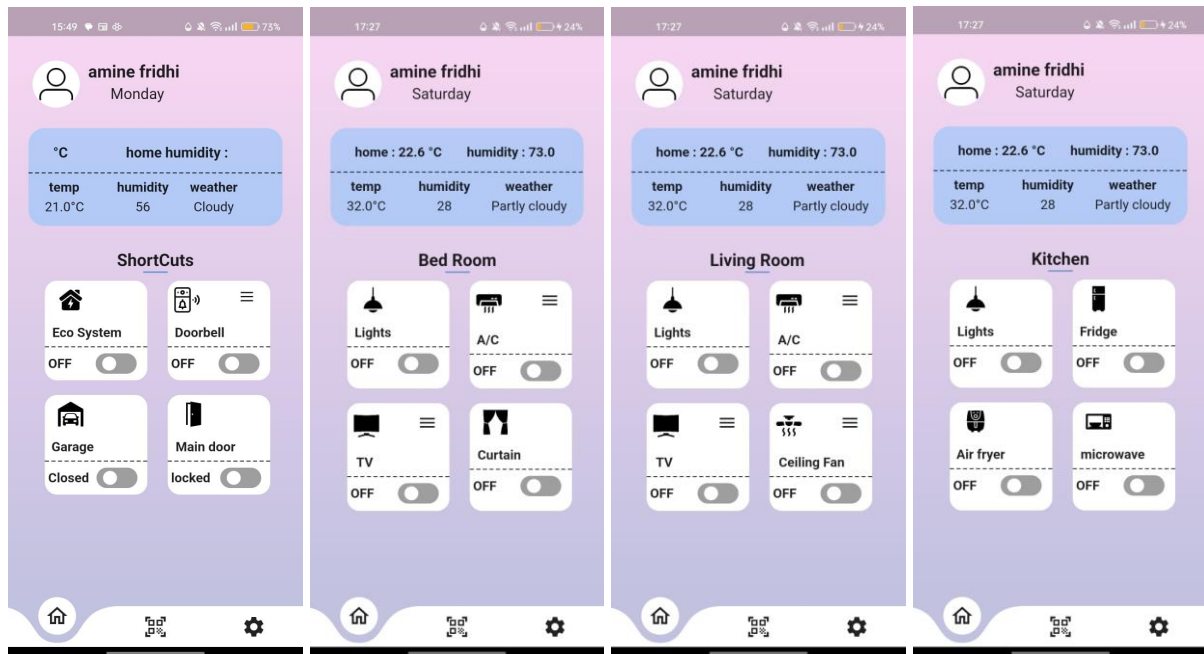
-In the kitchen, you can control the smart oven, which allows you to set the temperature and timer for your cooking needs. You can also control the smart refrigerator, which allows you to adjust the temperature, set reminders, and manage your grocery list.

In the sleeping room, you can control the smart lighting system, which includes the ceiling light and bedside lamps. You can adjust the brightness and color of the lights to create a relaxing and comfortable environment for sleeping.

-In the living room, you can control the smart thermostat, which allows you to adjust the temperature to your desired level. You can also control the smart TV and sound system, which allows you to turn them on and off, adjust the volume, and switch between different inputs.



With the Room Control section, you have the power to manage and control all of your smart devices from one central location. Whether you want to adjust the temperature, turn on the lights, or start cooking dinner, everything is just a tap away.



**Figure 32: different rooms**

And here is the full dashboard screen from shortcuts to kitchen.

### 1.1. Device Control:

In each room we have devices specially devices that we can control using IR sensor like Tv, A/C and ceiling fan and there is how we can control it using Nexthome application

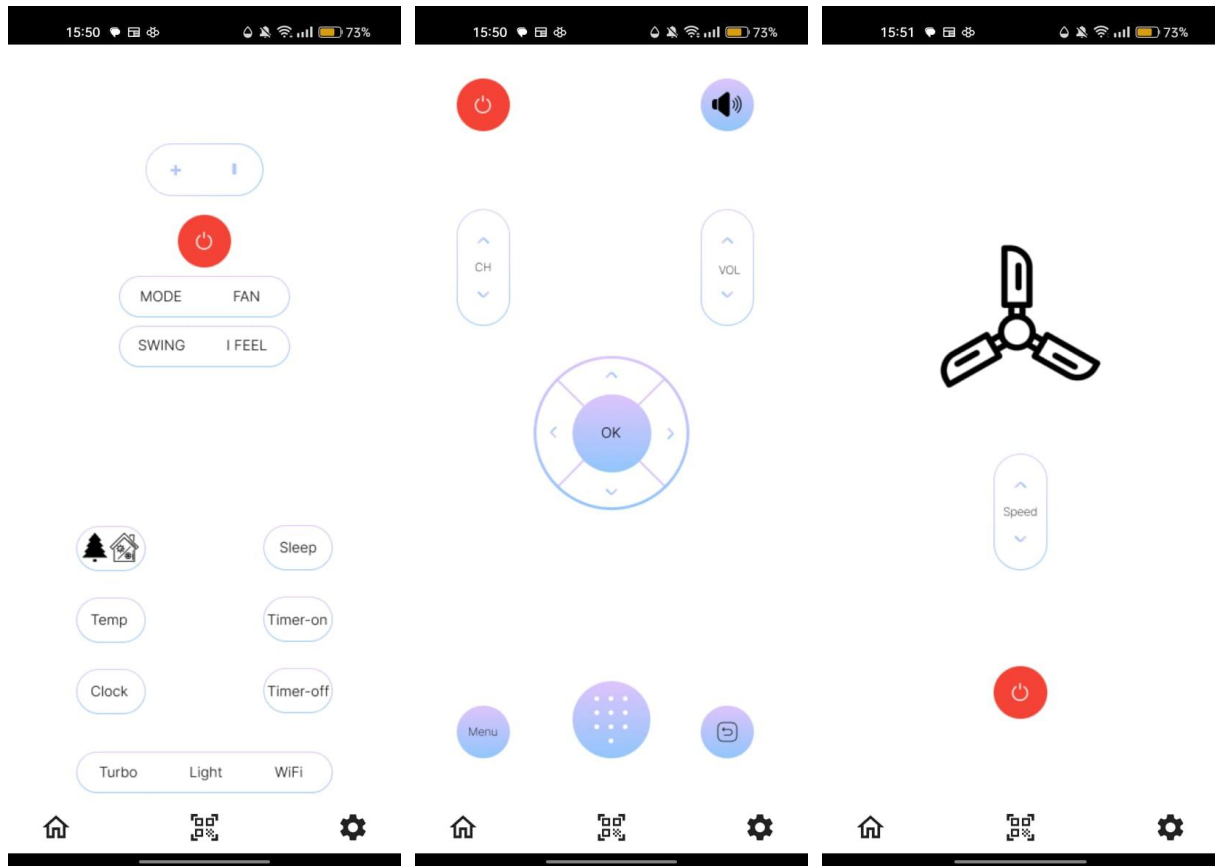


Figure 33: light tv remote

Figure 34:light AC remote

Figure 35: light ceiling fan

We are using IR sensor means we cannot verify while pressing the button if it works or no so I added vibration to make sure when we press the button it works

### 1.2.Devices can be used:

In the NextHome application, you have the flexibility to control and manage various connected devices within your home. These devices include smart garage systems, doors, and cameras, among others. With the NextHome app, you can seamlessly integrate and interact with these devices, providing you with convenient and centralized control over your home automation.

And here some examples:

UltraLoq Latch 5 Fingerprint: This keypad smart lock replaces a doorknob—not a deadbolt—with an accessibility-friendly lever handle, and it has a fingerprint reader to make unlocking speedy.



**Figure 36: UltraIQ Latch 5 Fingerprint [7]**

Chamberlain MyQ-G0401: can manage up to two garage doors, and the wireless hub makes it easy to install. it's also an excellent value for a basic smart garage controller.



**Figure 37: Chamberlain MyQ-G0401 [8]**

And a lot of other products can be used specially connected or smart gadgets

## 2. QR CODE

Using QR code scanning as a login method is a convenient and efficient way to authenticate users in mobile applications. The process involves generating a unique QR code for each user, which is then scanned by a separate device to log them in. This method has been adopted by several popular applications such as Steam and Discord.

In my application, I have implemented this login method by generating a unique QR code for each user. When the user needs to log in, they can simply scan the QR code using another device, and they will be automatically logged in to their account without the need for entering any login credentials. This process saves users time and eliminates the need to remember login details.

By using the QR code scanning method for login, my application ensures a secure and seamless login process for users. Additionally, it provides a more modern and convenient way of logging in, which is essential for improving the overall user experience.

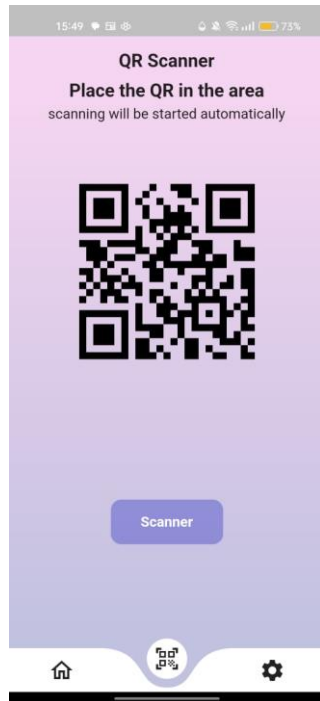


Figure 38: Light QR scanner



Figure 39: dark QR scanner

### 3. Settings

Settings page typically serves as a centralized location within an application or platform where users can customize their preferences and personalize their experience. It often includes sections such as "Edit Profile," "Change Password," and options to control features like dark mode and push notifications.

When designing the settings page, it's essential to consider factors such as user-friendliness, intuitive navigation, and clear instructions. Ensuring a consistent and cohesive design with the rest of the application or platform helps users quickly understand and locate the desired settings. Additionally, providing informative descriptions or tooltips can assist users in understanding the purpose and impact of each setting option.

Remember to prioritize user privacy and security when implementing settings related to personal information, passwords, and notifications. Providing appropriate safeguards and following best practices for data protection ensures a trustworthy and reliable user experience.

Overall, a well-designed settings page with sections for editing profiles, changing passwords, controlling dark mode, and managing push notifications enhances user satisfaction and empowers individuals to tailor their application experience to their preferences.

### **3.1.Edit profile**

In the "Edit Profile" section, users have the ability to modify their name and last name, and there is a button to submit the changes.


Name: This field allows users to input their first name or full name. They can edit or replace their current name with the desired changes.

Last Name: This field allows users to input their last name or surname. Similar to the name field, users can modify their existing last name or enter a new one.

Submit Changes: After users have made the desired modifications to their name and last name, a "Submit Changes" button is provided. Clicking this button saves the updated information and applies it to their profile.

Upon clicking the "Submit Changes" button, the system should validate the entered data to ensure its accuracy and completeness. If any errors or missing information are detected, appropriate error messages should be displayed to guide users in correcting their inputs.

Additionally, it's a good practice to provide visual cues or indicators, such as displaying a loading spinner or success message, to inform users that their changes have been successfully saved. This feedback reassures users that their modifications have been applied and allows for a smooth and transparent user experience.

and you change too your profile picture in this section by pressing this icon  and you will pick one of your pictures saved in your gallery

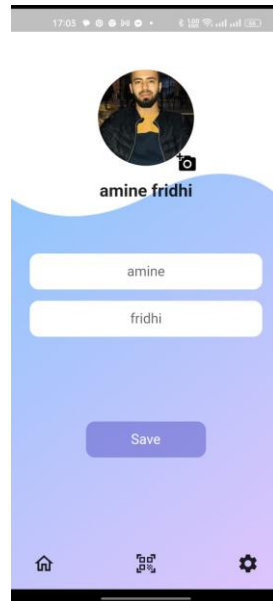


Figure 40: edit profile picture

Then instantly you will see your new profile picture in the settings page and dashboard too

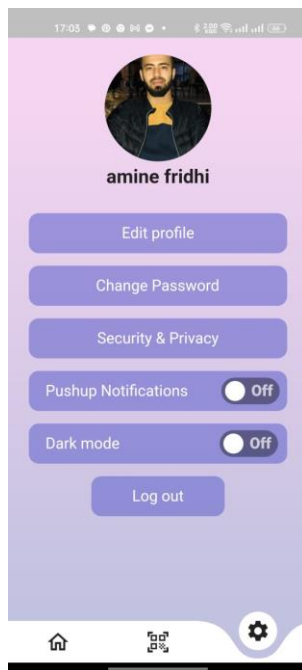


Figure 41: profile picture in the settings screen

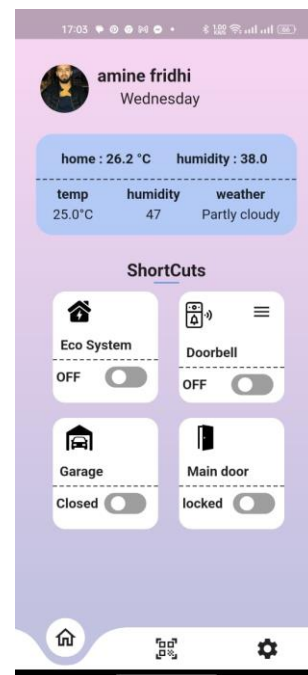


Figure 42: profile picture in the dashboard

### 3.2.Security

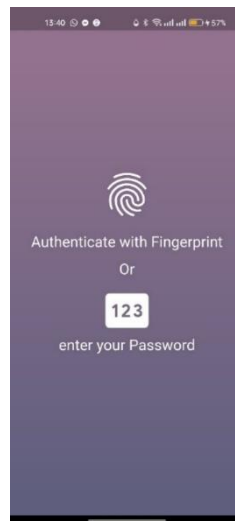
In the security section of my Next Home app, we have taken steps to make sure it is safe and secure. To access the dashboard, you have two options: using your fingerprints or a password.

Fingerprints are unique to each person and difficult to copy, which makes them a reliable security feature. Only authorized individuals can unlock the app by using their fingerprints.

Alternatively, you can choose to use a password. We have implemented special measures to keep your password protected. It is stored in a way that makes it very hard for anyone to figure it out. We also use additional techniques to make sure your password is even more secure.

To prevent unauthorized access, we have a system in place that detects and blocks multiple unsuccessful attempts. If someone tries too many times with the wrong fingerprint or password, they will be temporarily blocked from trying again. This helps protect against people who try different fingerprints or passwords to gain access.

In summary, the security in Next Home is designed to be strong and reliable. You can choose to use your fingerprints or a password to access the app, and we have implemented measures keep your information and control over your home safe.

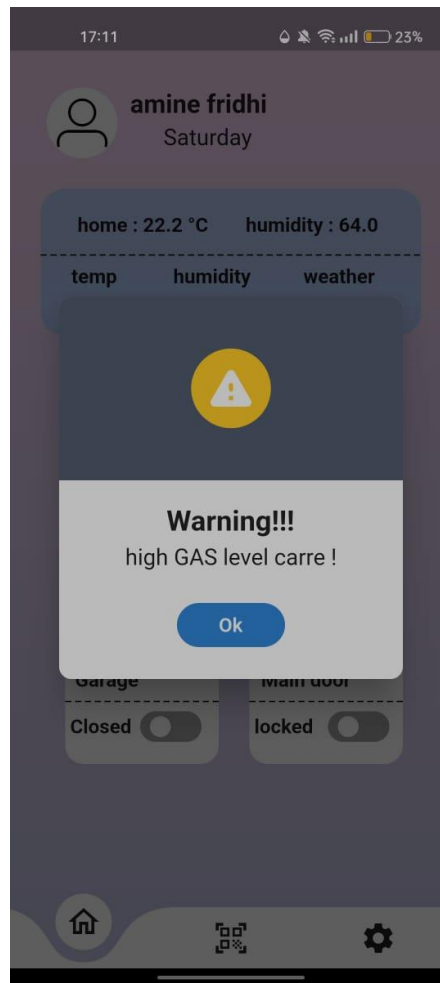


**Figure 43: security options**

## 4. Notification System

The notification system in Nexthome is a crucial part of ensuring the safety and security of its users. The system is designed to detect dangerous situations in the home environment, such as

the presence of gas or high temperature, and alert the user through push notifications on their mobile device. This allows the user to take action immediately and prevent any potential harm.



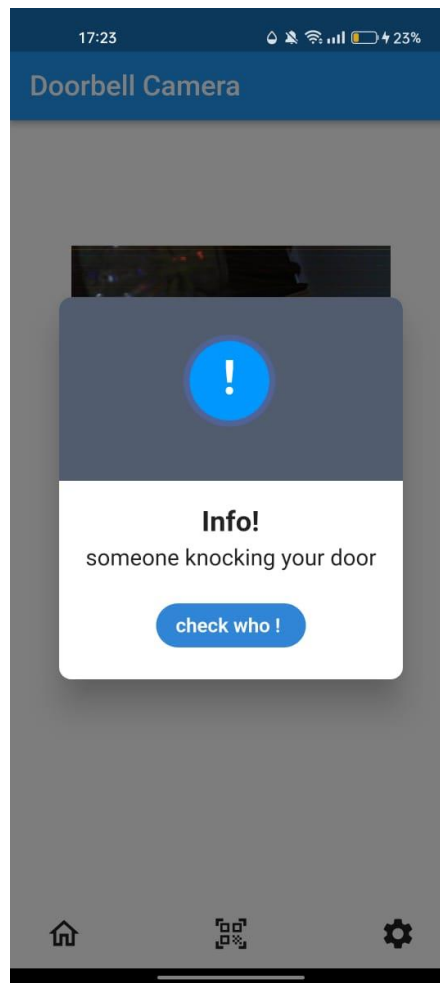
**Figure 44: Gas sensor warning**

The notification system can be turned off or on by the user through the settings screen. In case of an emergency situation, the system will trigger an alert regardless of whether the user has turned on the notification system or not. This ensures that the user is informed about any hazardous situation that may arise in their home.

The system works by using sensors that are installed in different parts of the house, such as the kitchen, living room, and bedroom. These sensors detect any unusual activity or changes in the environment that may pose a risk to the user. For instance, if the temperature in the kitchen suddenly rises due to a gas leak, the sensor will detect this and immediately send a notification to the user's mobile device.

And this works too with the doorbell outside when someone press the button you will get it in your phone before you hear the alarm





**Figure 45: Doorbell notification**

In conclusion, the notification system in Nexthome is an essential feature that ensures the safety and security of its users. The system is highly customizable and can be turned off or on depending on the user's preferences. With its advanced sensors and notification capabilities, Nexthome provides a reliable and efficient solution to ensure the well-being of its users.

## **5. Conclusion**

In conclusion, Chapter III has covered a variety of features available in our application. We have highlighted the user-friendly login and signup processes, the easy-to-use dashboard, and the ability to manage rooms and devices effortlessly. Our focus on ensuring data accuracy and security has been emphasized, aiming to build trust among users. By incorporating these features, our goal is to provide a convenient smart home experience that simplifies everyday tasks and improves the overall quality of life. Moving forward, Chapter IV will explore additional customization options and advanced settings within our application.

## CHAPTER IV: Experiment and result

### 1. Introduction

In this chapter, we will present the experiment conducted to validate the functionality of the smart home automation system using an Arduino microcontroller, various sensors, and the Firebase Realtime Database. The purpose of the experiment was to test the system's performance in terms of remote monitoring and control of the home environment. The code used for the experiment included libraries for Wi-Fi communication, Firebase integration, and sensor interfacing.

### 2. Arduino board setup

The system described in my code aims to create a smart home automation setup using an Arduino microcontroller, various sensors, and the Firebase Realtime Database. The system leverages the power of Wi-Fi connectivity to enable remote monitoring and control of the home environment. The code begins by including the necessary libraries for Wi-Fi communication, Firebase integration, and sensor interfacing.

To establish a connection to the Wi-Fi network, the code defines the network credentials, including the network name (SSID) and password. These credentials allow the Arduino to connect to the internet and facilitate communication with the Firebase Realtime Database. The Firebase host URL and authorization token are also specified, providing the necessary information to authenticate and communicate with Firebase.

Pin assignments are crucial in connecting the hardware components to the Arduino. The code assigns specific pins for the DHT temperature and humidity sensor, the MQ-2 gas sensor, the button, and the LED. These pin definitions enable the Arduino to read sensor data, detect button presses, and control the LED based on the received signals.

The code proceeds to instantiate objects for the DHT sensor and Firebase data handling. The DHT sensor object allows the Arduino to read temperature and humidity data from the sensor, while the Firebase data object enables communication with the Firebase Realtime Database.

The setup function is responsible for the initial configuration of the system. It sets up serial communication for debugging and sets pin modes for the LED and button. Additionally, it establishes a connection to the Wi-Fi network using the provided credentials and initializes the connection to the Firebase Realtime Database.

Furthermore, the setup function initializes the DHT sensor, sets initial values for certain Firebase database paths, and provides a warm-up time for the MQ-2 gas sensor. The warm-up time ensures that the gas sensor stabilizes its readings before actual data collection begins.

The main loop function is where the system continuously operates. It reads temperature and humidity data from the DHT sensor and sends this information to the Firebase database. It also reads gas sensor data and updates the corresponding path in the database. The system continuously checks for button presses and handles them accordingly by updating the Firebase database and controlling the LED state.

Throughout the execution of the main loop, the system retrieves the LED state from the Firebase database and adjusts the LED pin accordingly. This allows for remote control of the LED via the Firebase platform.

A short delay is introduced at the end of each iteration of the loop to control the rate at which sensor data is collected and sent to Firebase.

In summary, the system utilizes Wi-Fi connectivity, sensor readings, and Firebase integration to create a smart home automation setup. It enables remote monitoring and control of temperature, humidity, gas levels, and the LED state. The system can be expanded to include more sensors and actuators, providing a foundation for building a comprehensive smart home solution.

### **2.1.Simulation**

The simulation process made by circuito.io

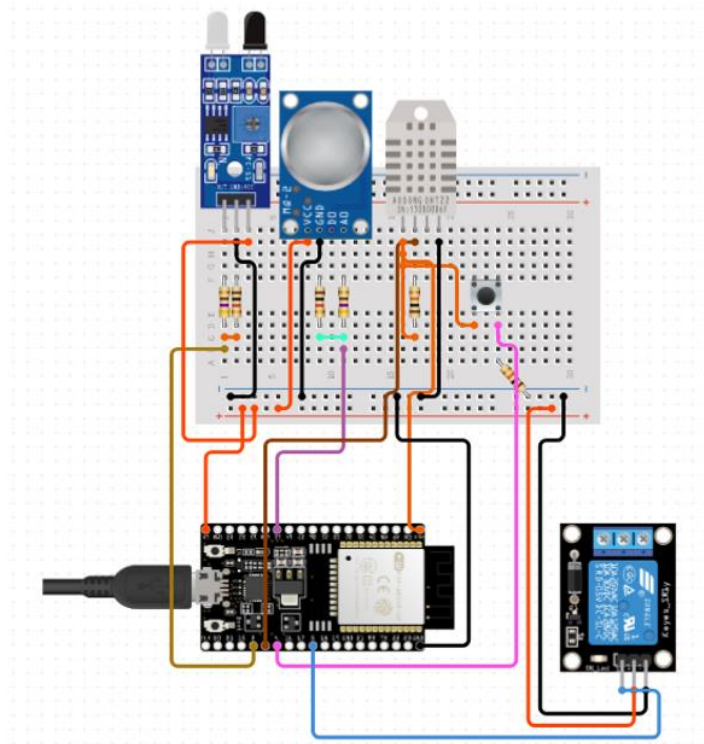


Figure 46: Arduino simulation [9]

### 3. Excremental

After the simulation here is the real realization one by one

First here let's start with the dht11 temperature and humidity sensor

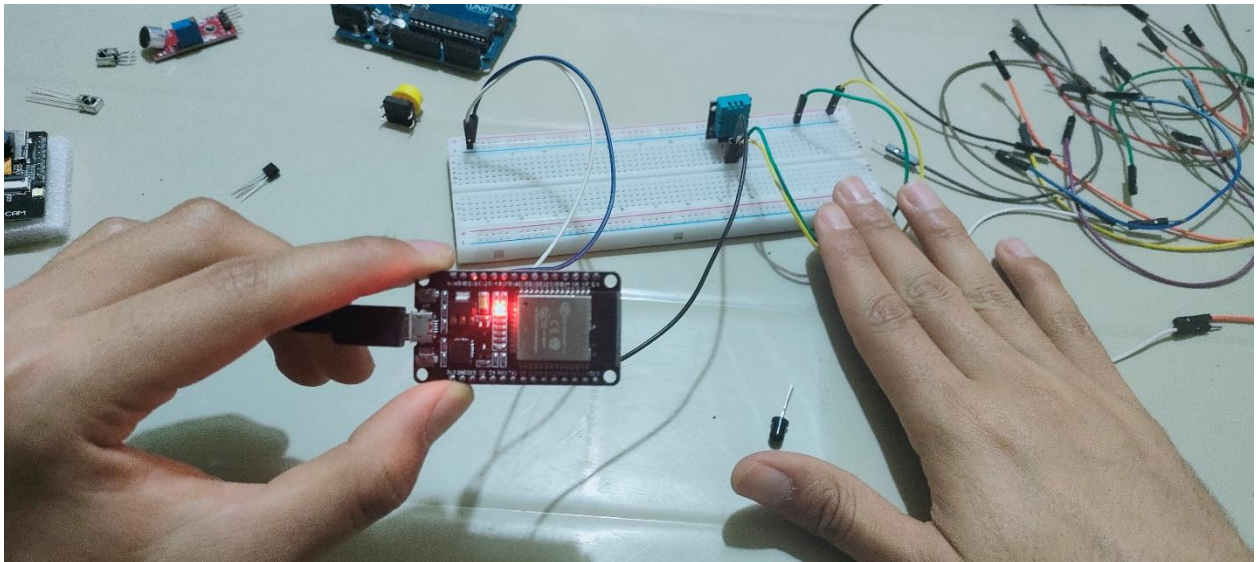
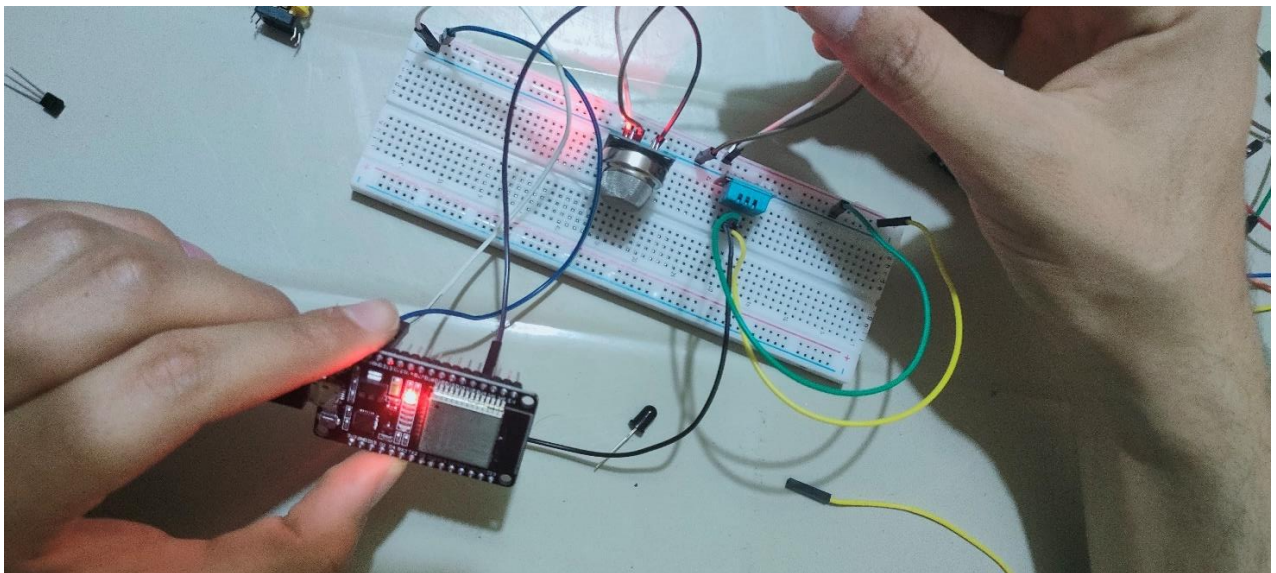


Figure 47: linking DHT11 with ESP32

To link the DHT11 sensor with the ESP32, you'll need to follow a few simple steps. First, make sure you have both the DHT11 sensor and the ESP32 board. The DHT11 sensor is responsible for measuring temperature and humidity, while the ESP32 is a Wi-Fi enabled microcontroller board. Begin by connecting the DHT11 sensor to the appropriate pins on the ESP32. Usually, the sensor's data pin is connected to a digital pin on the ESP32. Then, you'll need to write some code in a language like Arduino to read the sensor data and transmit it to the ESP32. Once the data is received by the ESP32, it can be further processed or sent to a server or other devices over Wi-Fi. This connection allows you to monitor and control the temperature and humidity readings from the DHT11 sensor using the ESP32 board, opening up a world of possibilities for home automation, weather monitoring, and more.

And the next step is adding the MQ-2



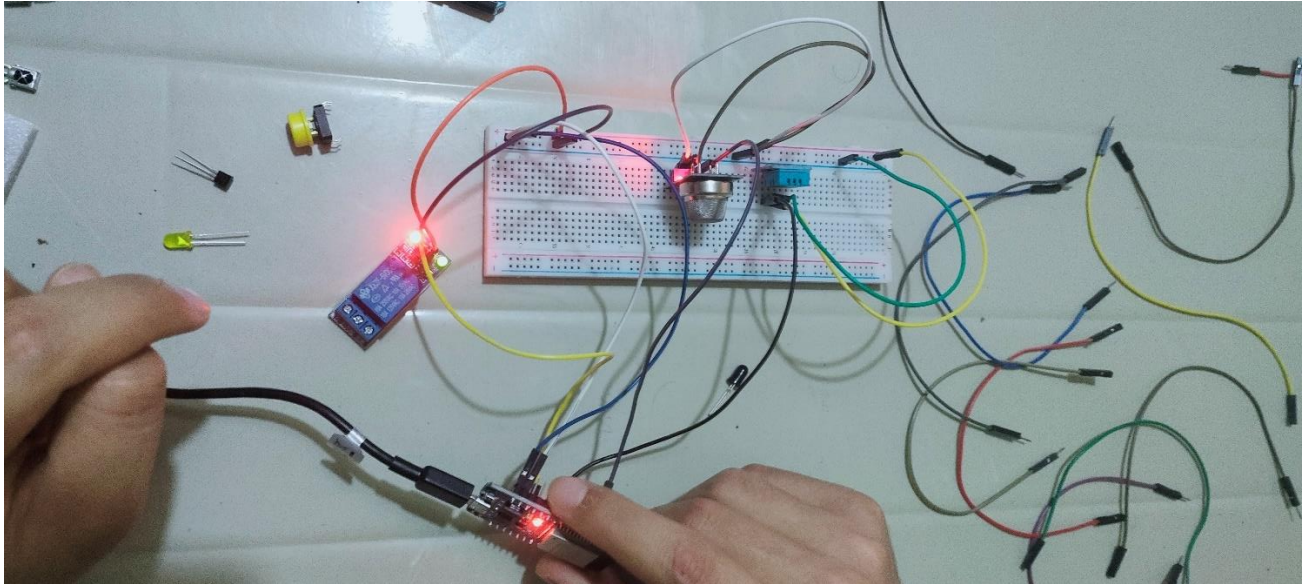
**Figure 48: adding the mq-2**

To incorporate the MQ-2 gas sensor with the ESP32, you can expand your project's capabilities to include gas detection and monitoring. The MQ-2 sensor is designed to detect various gases such as LPG, propane, methane, and carbon monoxide. Connecting the MQ-2 sensor to the ESP32 is a straightforward process. Start by obtaining both the MQ-2 sensor and the ESP32 board. Connect the appropriate pins of the MQ-2 sensor to the digital pins of the ESP32. Then, using a programming language like Arduino, you can write code to read the sensor's data and send it to the ESP32. Once the data is received, the ESP32 can process it and trigger actions or alerts based on the gas levels detected. For instance, you could set up notifications to be sent to your phone or activate ventilation systems when dangerous gas concentrations are detected.



Integrating the MQ-2 sensor with the ESP32 provides you with a powerful tool for gas monitoring and safety applications.

And the next step is adding the relay now



**Figure 49: adding relay**

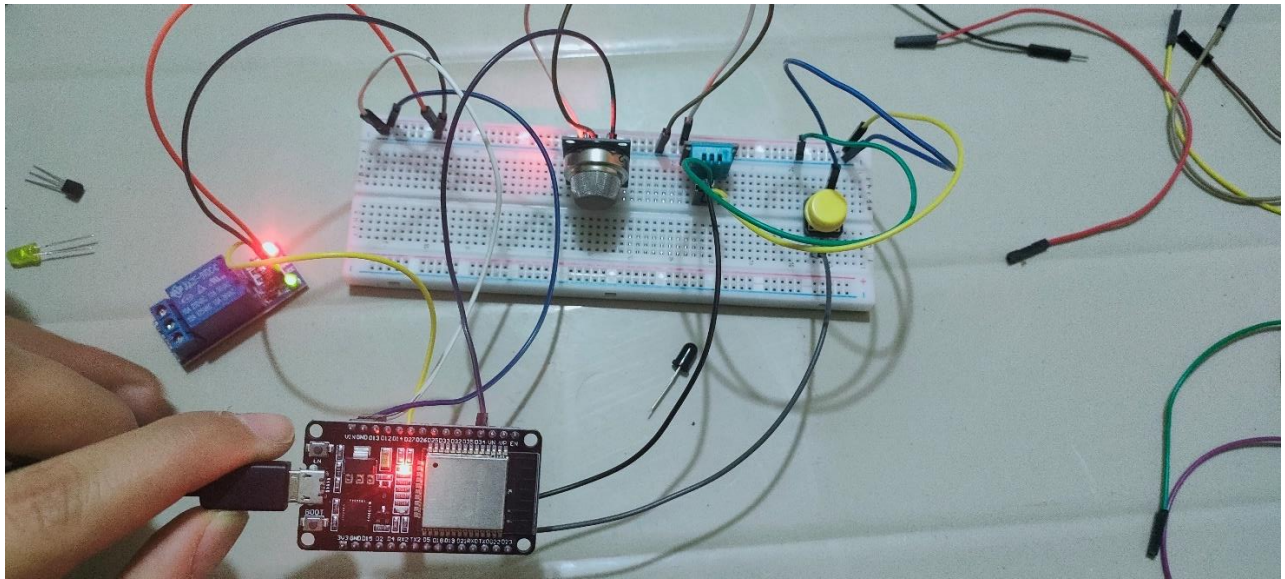
To incorporate a relay for controlling an electric device, such as a lamp, with the ESP32, you can enhance your project with the ability to remotely switch the lamp on or off. A relay is an electromechanical switch that allows low-power devices like microcontrollers to control high-power devices like lamps. Here's how you can connect a relay to the ESP32.

First, gather the required components: the ESP32 board, a relay module compatible with the ESP32's voltage levels, and the lamp you wish to control.

Next, identify the appropriate pins on the relay module. Typically, there are signal, ground, and power pins. Connect the relay module's signal pin to a digital pin on the ESP32 board. Make sure to also connect the relay module's ground pin to the ESP32's ground pin, and the power

With the relay properly connected and the code uploaded to the ESP32, you can now control the lamp remotely. You can utilize various methods, such as a mobile app or a web interface, to send commands to the ESP32 over Wi-Fi. This enables you to turn the lamp on or off from anywhere with an internet connection, adding convenience and automation to your lighting setup.

And the next step now is adding the button will act like the button for the doorbell



**Figure 50: adding button**

By adding a button to your ESP32, you can transform it into a doorbell that can be triggered with a simple press. Here's how you can integrate the button functionality with the ESP32:

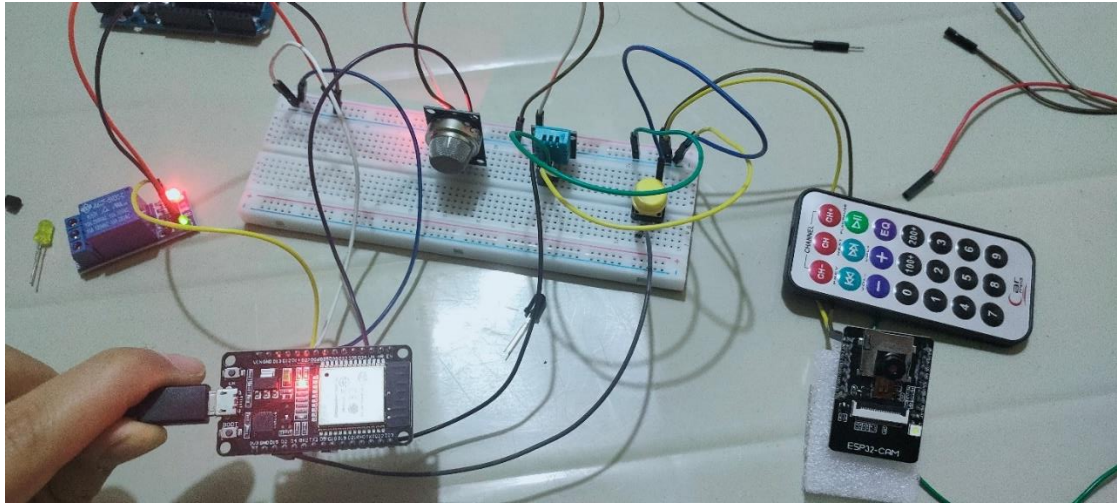
Connect one leg of the push-button to a digital pin on the ESP32 board. Connect the other leg of the push-button to the ground (GND) pin on the board. It's important to include a pull-up resistor between the digital pin and the VCC pin (3.3V) on the ESP32. This resistor will ensure that the digital pin is pulled high when the button is not pressed.

Next, in your programming environment, write code that continuously monitors the state of the digital pin to detect when the button is pressed. You can use a polling approach by regularly checking the pin's state or utilize an interrupt-driven approach to trigger an action immediately when the button is pressed.

Once the button press is detected, you can program the ESP32 to respond accordingly. For instance, you can send a notification to your smartphone, activate a buzzer or chime, or even integrate it with a home automation system to trigger additional actions like turning on lights or unlocking doors.

By combining the ESP32 with a button, you can create a smart doorbell system that is easily customizable and flexible. It provides a simple yet effective way to receive alerts and

notifications when someone presses the button, adding convenience and enhancing the functionality of your doorbell.



**Figure 51: adding the esp32-cam**



**Figure 52: esp32-cam**

By combining the ESP32-CAM module with a button, you can create a smart doorbell that includes a camera for visual monitoring. Here's how you can integrate the ESP32-CAM as a camera for the doorbell:

Connect the push-button in the same way as described in the previous response, with one leg connected to a digital pin on the ESP32-CAM and the other leg connected to the ground (GND) pin. Ensure the appropriate pull-up resistor is included to maintain the pin's high state when the button is not pressed.



Next, set up the ESP32-CAM module by following the instructions provided by the manufacturer. This typically involves connecting additional components, such as capacitors, for stable operation.

Now, with the ESP32-CAM properly configured and the button connected, write code to monitor the button's state and trigger actions accordingly. When the button is pressed, you can program the ESP32-CAM to capture an image or start a video recording.

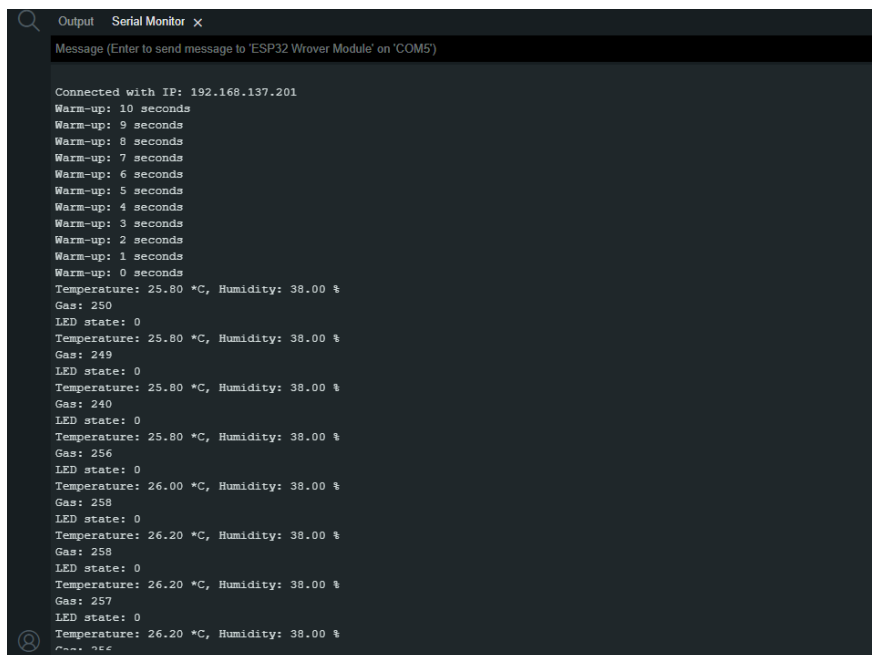
Utilizing the camera capabilities of the ESP32-CAM, you can then send the captured image or video to a server or storage platform. This allows you to remotely view and monitor the doorbell activity using a mobile app or web interface.

Additionally, you can implement features like motion detection to automatically capture images or videos when motion is detected near the doorbell. This enhances the security aspect of your smart doorbell system.

By combining the ESP32-CAM with a button, you can create a smart doorbell that not only provides audible alerts but also captures visual information. This allows you to remotely monitor your doorstep, enhance security, and have a comprehensive view of who is at your door, all within a single integrated system.

## 4. Result

And here is the result for all the sensors in the serial monitor of the Arduino IDE

A screenshot of the Arduino IDE Serial Monitor window. The window title is "Output Serial Monitor x". The message input field at the top says "Message (Enter to send message to 'ESP32 Wrover Module' on 'COM5')". The output text shows a series of "Warm-up: X seconds" messages from 10 down to 0. After the warm-up, it displays sensor data: "Temperature: 25.80 \*C, Humidity: 38.00 %", "Gas: 250", "LED state: 0", and then repeats this set of data with slightly different values (e.g., 249, 240, 256, 258, 258, 257) for each subsequent line. The text is white on a dark background.

```
Connected with IP: 192.168.137.201
Warm-up: 10 seconds
Warm-up: 9 seconds
Warm-up: 8 seconds
Warm-up: 7 seconds
Warm-up: 6 seconds
Warm-up: 5 seconds
Warm-up: 4 seconds
Warm-up: 3 seconds
Warm-up: 2 seconds
Warm-up: 1 seconds
Warm-up: 0 seconds
Temperature: 25.80 *C, Humidity: 38.00 %
Gas: 250
LED state: 0
Temperature: 25.80 *C, Humidity: 38.00 %
Gas: 249
LED state: 0
Temperature: 25.80 *C, Humidity: 38.00 %
Gas: 240
LED state: 0
Temperature: 25.80 *C, Humidity: 38.00 %
Gas: 256
LED state: 0
Temperature: 26.00 *C, Humidity: 38.00 %
Gas: 258
LED state: 0
Temperature: 26.20 *C, Humidity: 38.00 %
Gas: 258
LED state: 0
Temperature: 26.20 *C, Humidity: 38.00 %
Gas: 257
LED state: 0
Temperature: 26.20 *C, Humidity: 38.00 %
```

Figure 53: serial monitor of the Arduino IDE



**Figure 54: the result saved in the firebase**

The results of this experimental setup, incorporating the DHT11 sensor, MQ-2 gas sensor, relay module, button, and the ESP32 microcontroller, have been highly successful, demonstrating the seamless integration and proper functioning of all components. The DHT11 sensor has reliably measured temperature and humidity, providing accurate and consistent readings. With the MQ-2 gas sensor, various gases such as LPG, propane, methane, and carbon monoxide have been effectively detected, enabling efficient gas monitoring and ensuring safety measures are in place. The relay module has showcased its capability to control an electric device, such as a lamp, allowing for convenient remote switching on and off. Furthermore, the button has served as an efficient doorbell trigger, reliably notifying when pressed and facilitating a prompt response. The ESP32 microcontroller has played a vital role in connecting and coordinating these components, acting as a central hub for data communication and processing. Its stable performance and seamless integration have provided a robust foundation for this smart home system. With this setup, the user gains reliable monitoring, control, and automation capabilities, fostering a convenient and secure living environment. The successful outcomes of this experiment illustrate the effectiveness and potential of these components when combined, opening up possibilities for further innovation and applications in the realm of smart homes and IoT technologies.

In addition to the successful integration and functioning of the components mentioned earlier, another notable achievement of this experimental setup is the successful transmission of data

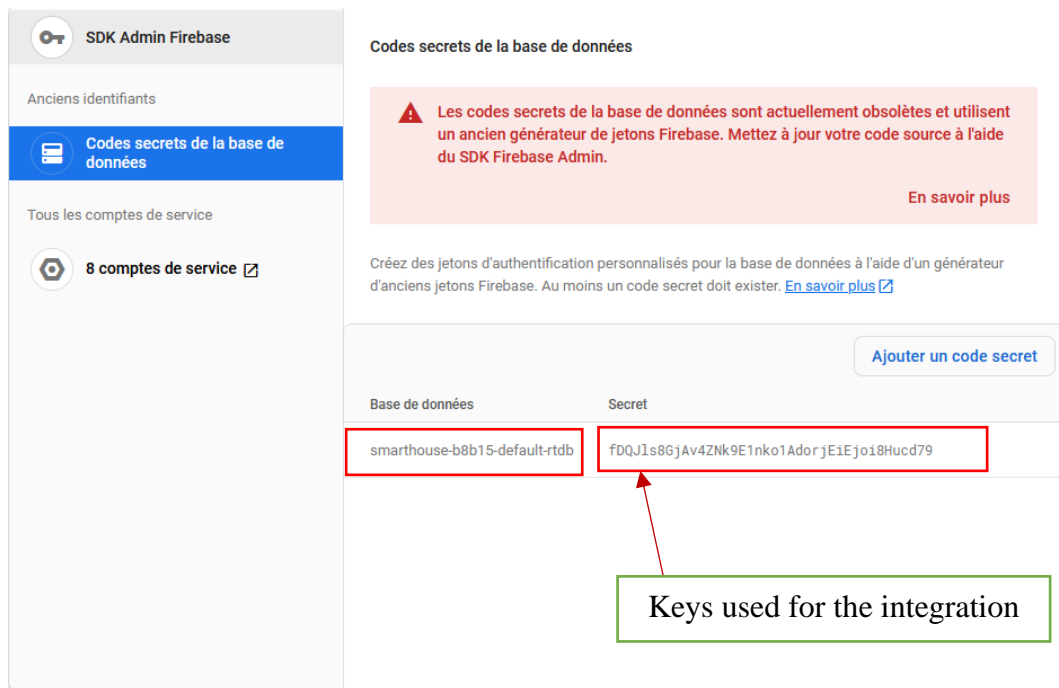
to Firebase. The ESP32 microcontroller, acting as the central hub, has been configured to send the collected sensor data, including temperature, humidity, gas readings, and button activations, to the Firebase platform. This data transmission process has been reliable and consistent, allowing for real-time monitoring and storage of the captured information. By leveraging Firebase's cloud-based infrastructure, the user gains the advantage of accessing and analyzing the data remotely, from any device with internet connectivity. This seamless integration with Firebase further enhances the capabilities of the smart home system, enabling advanced data analytics, historical trend analysis, and the potential for integration with other services and applications. Overall, the successful transmission of data to Firebase expands the possibilities for data-driven decision-making and enables the user to leverage the power of cloud-based technologies for a smarter and more efficient living environment.

## **5. Database integration**

The integration of Firestore database and Firebase Realtime Database in Flutter applications offers a comprehensive solution for managing data in real time with efficient authentication. Firestore is a flexible NoSQL document database that seamlessly integrates with Firebase's Realtime Database. By utilizing Firebase Authentication, developers can secure access to the databases and control user permissions effectively.

To establish the integration, Flutter applications can leverage the Firebase SDK, which provides an easy-to-use API for communication with Firestore and Realtime Database. With the Firebase Auth link, developers can implement secure user authentication mechanisms such as email/password, social logins, or custom authentication systems.

Once authenticated, users can access Firestore's document collections and Realtime Database's hierarchical data structure using unique keys. These keys serve as identifiers for specific documents or data nodes within the databases. Developers can use the keys to read, write, update, and delete data in real time.

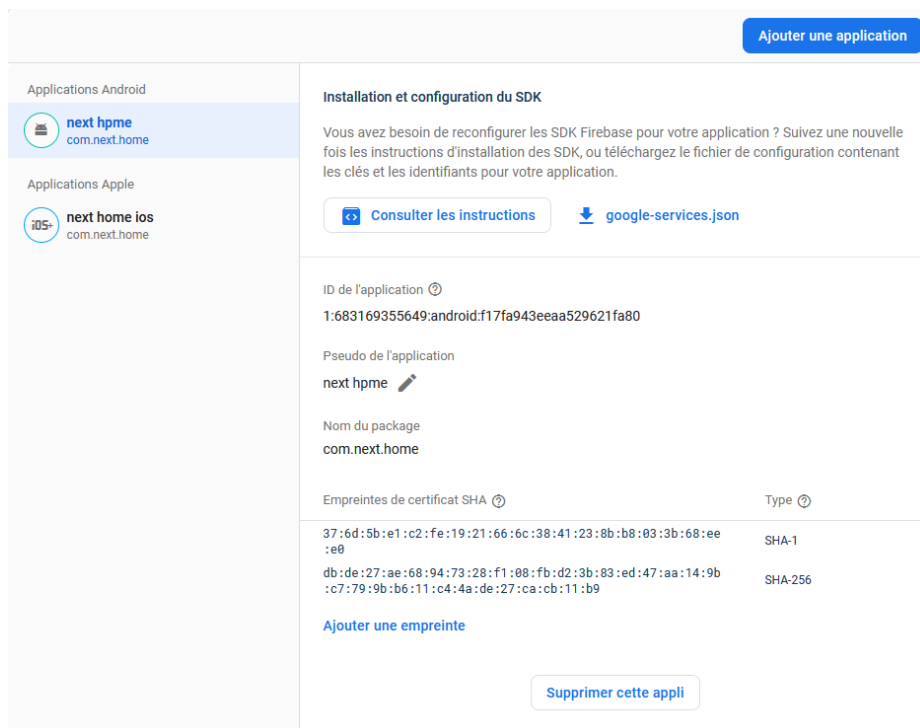


**Figure 55: Integration using key and API**

And here is it with the Arduino code

```
#define FIREBASE_HOST "https://smarthouse-b8b15-default-rtdb.firebaseio.com/"
#define FIREBASE_AUTH "fDQJls8GjAv4ZNk9E1nko1AdorjEiEjoi8Hucd79"
```

And for the flutter code downloading google-services.json file



**Figure 56: google-services.json download link**

the google-services.json file and putting in the application files exactly under C:\Users\user\Desktop\nexthome\android\app\google-services.json

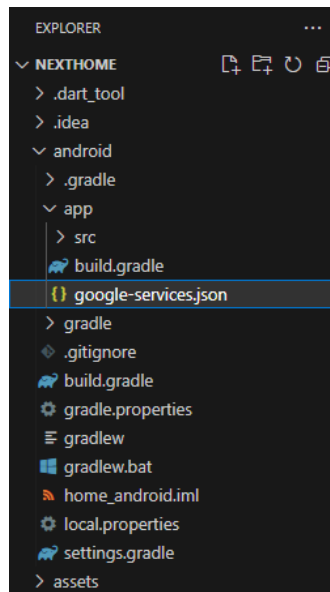


Figure 57: location where to put the JSON file

## 6. conclusion

In conclusion, this chapter presented an experiment that successfully validated the functionality of a smart home automation system using an Arduino microcontroller, various sensors, and the Firebase Realtime Database. The system demonstrated effective remote monitoring and control of the home environment, allowing users to manage temperature, humidity, gas levels, and even control an electric device like a lamp. The integration of Firebase enabled seamless data transmission and storage, offering real-time access and analysis of the collected information. Overall, this experiment showcased the potential of these components to create a convenient and secure living environment, with opportunities for further innovation and applications in the realm of smart homes and IoT technologies.

## Discussion

Doing this project was really hard, especially since it was my first time. At first, I used an Arduino Uno, but getting it to connect to Wi-Fi was tough. I couldn't figure it out, so I switched to the ESP32, which was even harder.

The biggest challenge was using Flutter. I didn't learn about Arduino or Flutter in university. I only knew a little about Arduino, and I had to learn Flutter all on my own. The problem was that Flutter is a new language, and the projects on GitHub didn't work with its latest version.

But I didn't give up. It took a lot of effort, but I managed to make Arduino and Flutter work together. It wasn't easy, but I did it. This experience taught me that learning on your own and being able to adapt to new things is really important. Even though it was tough, I finished the project and showed that with patience and hard work, you can accomplish difficult things.

## GENERAL CONCLUSION

In conclusion, the NextHome project is about creating a smart home system that can be controlled and monitored remotely. It uses the Arduino microcontroller, sensors, and the Firebase Realtime Database. We discussed different aspects of the project, including the hardware setup, code, sensor integration, and database management.

We started by understanding the hardware components, like the Arduino board, which acts as the brain of the system. We learned how to connect the sensors to the Arduino using specific pins. By using Wi-Fi, the system allows us to control and check the house from anywhere using the NextHome app.

We looked at the code used in the project, which included libraries for Wi-Fi, Firebase, and sensors. We set up the Wi-Fi connection and connected to the Firebase database. The setup function made sure everything was ready to start working.

The system collects data from sensors, such as temperature and humidity, and sends it to the Firebase database. This data can be accessed and controlled through the NextHome app. We also talked about how Firebase Authentication keeps the system secure and allows different users to have specific permissions.

The NextHome project has the potential for expansion. We can add more sensors and devices to make the system even better. It provides a good starting point for creating a complete smart home solution that can be customized to meet individual needs.

In summary, the NextHome project is an exciting smart home system that uses Arduino, sensors, and the Firebase Realtime Database. It lets us monitor and control our home remotely through the NextHome app. With its easy-to-use interface and possibilities for future improvements, the NextHome project has the power to transform the way we interact with our homes, making them more comfortable, convenient, and energy-efficient.

## Summary

The NextHome project is a smart home system that allows remote control and monitoring. It uses an Arduino microcontroller, sensors, and the Firebase Realtime Database. By connecting the sensors to the Arduino and utilizing Wi-Fi connectivity, the system enables users to control and check their homes from anywhere using the NextHome app. The project involves setting up the hardware, writing code, integrating sensors, and managing the Firebase database. Through the app, users can access and control various aspects of their home, such as temperature and humidity levels. The NextHome system serves as a foundation for building a comprehensive smart home solution, with the potential to expand and incorporate more sensors and devices. It offers a user-friendly interface and the possibility for customization, making homes more comfortable, convenient, and energy-efficient.

## Résumé

Le projet NextHome est un système domotique qui permet le contrôle à distance et la surveillance de la maison. Il utilise un microcontrôleur Arduino, des capteurs et la base de données Firebase en temps réel. En connectant les capteurs à l'Arduino et en utilisant la connectivité Wi-Fi, le système permet aux utilisateurs de contrôler et de vérifier leur maison depuis n'importe où en utilisant l'application NextHome. Le projet implique la configuration du matériel, l'écriture du code, l'intégration des capteurs et la gestion de la base de données Firebase. À travers l'application, les utilisateurs peuvent accéder et contrôler différents aspects de leur maison, tels que les niveaux de température et d'humidité. Le système NextHome sert de base pour construire une solution domotique complète, avec la possibilité d'ajouter davantage de capteurs et d'appareils. Il offre une interface conviviale et la possibilité de personnalisation, rendant les maisons plus confortables, pratiques et économes en énergie.



## Bibliography and Webography

- [1] Janusz Kacprzyk, et al. *Advanced Intelligent Systems for Sustainable Development (AI2SD{U2019}2020) : Volume 2*. Cham, Springer International Publishing, 2022
- [2] Nagaraj, Ambika. *Introduction to Sensors in IoT and Cloud Computing Applications*. Bentham Science Publishers, 1 Feb. 2021.
- [3] <https://blogs.embarcadero.com/quickly-consume-the-firebase-rest-api-with-the-open-source-firebase4delphi-library/>
- [4] <https://www.javatpoint.com/firebase-authentication>
- [5] Vedat Ozan Oner. *Developing IoT Projects with ESP32*. Packt Publishing Ltd, 13 Sept. 2021.
- [6] [https://www.figma.com/file/GQDweQTnIsiP9jFoj0nrxr/Login-UI-Design-\(Community\)?type=design&node-id=0-1&t=d1ldQX3JnNJn4iib-0](https://www.figma.com/file/GQDweQTnIsiP9jFoj0nrxr/Login-UI-Design-(Community)?type=design&node-id=0-1&t=d1ldQX3JnNJn4iib-0)
- [7] <https://u-tec.com/en-ca/products/latch-5-nfc>
- [8] <https://www.rona.ca/fr/produit/controleur-intelligent-pour-porte-de-garage-chamberlain-myq-g0401-07645061>
- [9] [circuitio.io](https://circuitio.io)
- [10] <https://mechatronicsblog.com/ESP32-nodemcu-pinout-for-arduino-ide/>

# Help Guide

The following information is intended to help each student who will take this project from the institute library:

## **Purchase Arduino and its accessories:**

Didactico:

Address: Route el Ain km 1 Immeuble Bouzguenda Bloc A App 01 Sfax 3000 Tunisie

Phone: [+216 99 707 685](tel:+21699707685)

Website: [www.didactico.tn](http://www.didactico.tn)

Tuni smart innovation:

Address: 1 Rue Pierre Mendès France• Ariana 2080

Phone: (+216) 51954444 (+216) 51954448

Website: <https://tuni-smart-innovation.com/>

## **Download and install flutter:**

Website: <https://docs.flutter.dev/get-started/install>

## **Download link Arduino software:**

[www.arduino.cc](http://www.arduino.cc)