Mini Industrial Project Report

SMART HOME AUTOMATION SYSTEM USING IOT

Submitted in partial fulfilment of the employment enhancement program of IT department

By

Debayan Dutta (14200221050)
Pijush Das (14200221027)
Ayaan Dey (14200221010)
Archita Kundu (14200221059)

Third year

Meghnad Saha Institute Of Technology

Under the supervision of
Mr. SK Suman

Academy Of Skill Development

Department of Information Technology

We hereby submit the project report prepared by us under the supervision of Mr. SK Suman entitled, **“SMART HOME AUTOMATION SYSTEM USING IOT”** for the partial fulfilment of the employment enhancement program of IT department of Meghnad Saha Institute of Technology, affiliated to Maulana Abul Kalam Azad University of Technology (MAKAUT).

DENT COMPUTECH PVT. L
KOLKATA

[illegible]

Mr. SK Suman
IOT and Android Developer
Academy of Skill Development.

[illegible]

Prof. Subir Hazra
HOD
IT, MSIT

ACKNOWLEDGEMENT

The achievement that is associated with the successful completion of any task would be incomplete without mentioning the names of those people whose endless cooperation made it possible. Their constant guidance and encouragement made all our efforts successful. We take this opportunity to express our deep gratitude towards our project mentor, **Mr. SK Suman** for giving such valuable suggestions, guidance and encouragement during the development of this project work. Last but not the least we are grateful to all the faculty members of Academy Of Skill Development for their support.

ABSTRACT

Smart Home Automation is a cutting-edge ESP32-based home automation and security system that seamlessly integrates occupancy-based lighting and fan control, water management, door entry security, temperature regulation, weather information, and LPG gas leakage detection. Using a combination of sensors and smart devices, the system enhances energy efficiency, user comfort, and safety. Occupancy sensors trigger automatic lighting and fan adjustments, while a water management module provides timely pump control notifications. The door entry system ensures security with forced entry detection and remote access through a mobile app. Temperature control optimizes indoor climate, and real-time weather information offers personalized suggestions. The system's proactive LPG gas leakage detection employs alarms and app notifications, emphasizing safety. Smart Home Automation offers a comprehensive solution for modernizing homes with intelligence, convenience, and security.

CONTENTS:

INTRODUCTION

In the ever-evolving landscape of home technology, the concept of a "Smart Home" has transcended mere convenience to encompass a harmonious integration of intelligence and security. Enter "Smart Home Automation," a pioneering project centered around the ESP32 microcontroller platform, weaving together an array of sophisticated features that redefine modern living. At its core, this system is a testament to the power of the Internet of Things (IoT), offering a dynamic and responsive environment within the confines of one's home.

Smart Home Automation seamlessly orchestrates a symphony of technologies, introducing groundbreaking elements such as occupancy-based lighting and fan control, water management, door entry security, temperature regulation, real-time weather updates, and LPG gas leakage detection. The ESP32 microcontroller serves as the brain, connecting and harmonizing various sensors and smart devices to create a cohesive and intelligent home ecosystem.

Occupancy sensors, driven by infrared precision, intelligently adjust lighting and fan settings, promoting energy efficiency while prioritizing occupant comfort. The water management module introduces a proactive dimension to resource utilization, offering real-time notifications for pump control based on water tank levels. The door entry security system, equipped with advanced sensors and cameras, not only detects forced entry attempts but also facilitates remote monitoring and control through an intuitive mobile app interface.

In matters of climate control, Smart Home Automation dynamically manages indoor temperatures, aligning with user preferences. Real-time weather updates sourced through APIs inform users about current conditions and provide personalized suggestions for weather-appropriate preparedness. Moreover, the system employs gas sensors to detect LPG gas leaks, triggering audible alarms and instantaneous mobile app notifications, thus championing a proactive approach to safety.

Smart Home Automation emerges not just as a technological marvel but as a holistic and user-centric solution, redefining the contours of contemporary living through its fusion of intelligence, sustainability, and security.

KEY FEATURES:

- **Occupancy-Based Lighting and Fan Control:**
  Utilizing infrared (IR) sensors, the system automatically switches lights and fans on upon detecting occupants in a room and turns them off during periods of inactivity, ensuring energy efficiency and convenience.
- **Water Management Module:**
  The system monitors water levels in a tank, sending app notifications to users for informed control of water pumps. This feature promotes efficient water usage and helps prevent overflows.
- **Door Entry Security:**
  Equipped with sensors and cameras, the system detects forced entry attempts and sends alerts to the homeowner. Additionally, the doorbell system captures images of visitors, allowing remote monitoring and control of door access through a mobile app.
- **Temperature Control:**
  Integrating temperature sensors, the system automatically adjusts the indoor climate by activating fans or heating systems based on user preferences. This ensures optimal comfort and energy conservation.
- **Weather Information and Suggestions:**
  The system fetches real-time weather data using APIs, displaying current conditions and providing personalized suggestions, such as carrying an umbrella or dressing appropriately, through the mobile app.
- **LPG Gas Leakage Detection:**
  Gas sensors are implemented to detect LPG gas leaks, triggering immediate alerts via a loud buzzer and mobile app notifications. Safety measures include the option for automated gas supply shut-off and integration with emergency contacts.

Smart Home Automation represents a cohesive integration of IoT technologies, particularly leveraging the capabilities of the ESP32 platform. The system prioritizes user convenience, energy efficiency, and safety, creating a versatile and responsive smart home environment. As technology continues to advance, Smart Home Automation serves as a scalable and adaptable solution for modern home automation and security needs.

COMPONENTS USED

- ESP 32
- Bread Board
- Jumper Wires
- LED
- Servo Motor
- Touch Switch
- RFID
- DHT11 Sensor
- Buzzer
- Ultrasonic Sensor
- Door Sensor
- IR Sensor
- 16 x 2 LCD
- I2C Driver
- MQ2 Gas Sensor

❖ ESP – 32:

ESP32 is a series of low-cost, low-power system on a chip microcontroller with integrated Wi-Fi and dual-mode Bluetooth. The ESP32 series employs either a Tensilica Xtensa LX6 microprocessor in both dual-core and single-corevariations, Xtensa LX7 dual-core microprocessor or a single-core RISC-V microprocessor and includes built-in antenna switches, RF balun, power amplifier, low-noise receive amplifier, filters, and power-management modules.
ESP32 is created and developed by Espressif Systems, a Shanghai-based Chinese company, and is manufactured byTSMC using their 40 nm process.[2] It is a successor to the ESP8266 microcontroller.

Since the release of the original ESP32, a number of variants have been introduced and announced. They form theESP32 family of microcontrollers. These chips have different CPUs and capabilities, but all share the same SDK andare largely code-compatible.

Features of the ESP-32:

- Single or Dual-Core 32-bit LX6 Microprocessor with clock frequency up to 240 MHz.
- 520 KB of SRAM, 448 KB of ROM and 16 KB of RTC SRAM.
- Supports 802.11 b/g/n Wi-Fi connectivity with speeds up to 150 Mbps.
- Support for both Classic Bluetooth v4.2 and BLE specifications.
- 34 Programmable GPIOs.
- Up to 18 channels of 12-bit SAR ADC and 2 channels of 8-bit DAC
- Serial Connectivity include 4 x SPI, 2 x I2C, 2 x I2S, 3 x UART.
- Ethernet MAC for physical LAN Communication (requires external PHY).
- 1 Host controller for SD/SDIO/MMC and 1 Slave controller for SDIO/SPI.
- Motor PWM and up to 16-channels of LED PWM.
- Secure Boot and Flash Encryption.
- Cryptographic Hardware Acceleration for AES, Hash (SHA-2), RSA, ECC and RNG.

ESP-32 HARDWARE PART:

Espressif Systems released several modules based on ESP32 and one of the popular options is the ESP-WROOM-32 Module. It consists of ESP32 SoC, a 40 MHz crystal oscillator, 4 MB Flash IC and some passive components.
The good thing about ESP-WROOM-32 Module is the PCB has edge castellation. So, what third-part manufacturers do is take the ESP-WROOM-32 Module and design a break-out board for this module. One such board is the ESP32 DevKit Board. It contains the ESP-WROOM-32 as the main module and also some additional hardware to easily program ESP32 and make connections with the GPIO Pins. There’s also 520 KB RAM and 4MB of Flash memory (for program and data storage) just enough to cope with the large strings that make up web pages, JSON/XML data, and everything we throw at IoT devices nowadays. ESP32 implements TCP/IP, full 802.11 b/g/n/e/i WLAN MAC protocol, and Wi-Fi Direct specification. This means ESP 32 can speak to most of the WiFi Routers out there when used in station(client) mode. Also, it is able to create an Access point with full 802.11 b/g/n/e/i.

POWER REQUIREMENT

The operating voltage of ESP32 ranges from 2.3 V to 3.6 V. When using a single-power supply, the recommended voltage of the power supply is 3.3 V, and its recommended output current is 500 mA or more.

- PSRAM and flash both are powered by VDD_SDIO. If the chip has an embedded flash, the voltage of
- VDD_SDIO is determined by the operating voltage of the embedded flash. If the chip also connects to an
- external PSRAM, the operating voltage of external PSRAM must match that of the embedded flash. This
- also applies if the chip has an embedded PSRAM but also connects to an external flash.
- When VDD_SDIO 1.8 V is used as the power supply for external flash/PSRAM, a 2 kΩ grounding resistor
- should be added to VDD_SDIO. For the circuit design, please refer to Figure ESP32•WROVER
- Schematics, in ESP32-WROVER Datasheet.
- When the three digital power supplies are used to drive peripherals, e.g., 3.3 V flash, they should complywith the peripherals' specifications.

With the use of advanced power-management technologies, ESP32 can switch between different power modes.Power modes:

- **Active mode:** The chip radio is powered on. The chip can receive, transmit, or listen.
- **Modem•sleep mode:** The CPU is operational and the clock is configurable. The Wi-Fi/Bluetooth baseband and radio are disabled.
- **Light•sleep mode**: The CPU is paused. The RTC memory and RTC peripherals, as well as the ULP coprocessors are running. Any wake-up events (MAC, host, RTC timer, or external interrupts)will wake up the chip.
- **Deep•sleep mode**: Only the RTC memory and RTC peripherals are powered on. Wi-Fi and Bluetooth connection data are stored in the RTC memory. The ULP coprocessor is functional.
- **Hibernation mode**: The internal 8 MHz oscillator and ULP coprocessor are disabled. The RTC recovery memory is powered down. Only one RTC timer on the slow clock and certain RTC GPIOs are active. TheRTC timer or the RTC GPIOs can wake up the chip from the Hibernation mode.

PERIPHERALS AND I/O

General Purpose Input/Output Interface (GPIO):

ESP32 has 34 GPIO pins which can be assigned various functions by programming the appropriate registers. There are several kinds of GPIOs: digital-only, analog-enabled, capacitive-touch-enabled, etc. Analog-enabledGPIOs and Capacitive-touch-enabled GPIOs can be configured as digital GPIOs.

Analog• to• Digital Converter (ADC):

ESP32 integrates two 12-bit SAR ADCs and supports measurements on 18 channels (analog-enabled pins). TheULP coprocessor in ESP32 is also designed to measure voltage, while operating in the sleep mode, which enables low-power consumption. The CPU can be woken up by a threshold setting and/or via other
triggers.
With appropriate settings, the ADCs can be configured to measure voltage on 18 pins maximum.

Digital• to• Analog Converter (DAC):

Two 8-bit DAC channels can be used to convert two digital signals into two analog voltage signal outputs. The design structure is composed of integrated resistor strings and a buffer. This dual DAC supports power supply asinput voltage reference. The two DAC channels can also support independent conversions.

Touch Sensor:

ESP32 has 10 capacitive-sensing GPIOs, which detect variations induced by touching or approaching the GPIOswith a finger or other objects. The low-noise nature of the design and the high sensitivity of the circuit allow relatively small pads to be used. Arrays of pads can also be used, so that a larger area or more points can be detected.

Ultra•Low•Power (ULP) Coprocessor:

The ULP coprocessor and RTC memory remain powered on during the Deep-sleep mode. Hence, the developercan store a program for the ULP coprocessor in the RTC slow memory to access the peripheral devices, internaltimers and internal sensors during the Deep-sleep mode.

Ethernet MAC Interface:

An IEEE-802.3-2008-compliant Media Access Controller (MAC) is provided for Ethernet LAN communications. ESP32 requires an external physical interface device (PHY) to connect to the physical LAN bus (twisted-pair, fiber,etc.). The PHY is connected to ESP32 through 17 signals of MII or nine signals of RMII.

I2C Interface

ESP32 has two I2C bus interfaces which can serve as I2C master or slave, depending on the user's configuration. The I2C interfaces support:

- Standard mode (100 Kbit/s)
- Fast mode (400 Kbit/s)
- Up to 5 MHz, yet constrained by SDA pull-up strength• 7-bit/10-bit addressing mode
- Dual addressing mode

Users can program command registers to control I2C interfaces, so that they have more flexibility

I2S Interface

Two standard I2S interfaces are available in ESP32. They can be operated in master or slave mode, in full duplexand half-duplex communication modes, and can be configured to operate with an 8-/16-/32-/48-/64-bit
resolution as input or output channels. BCK clock frequency, from 10 kHz up to 40 MHz, is supported. Whenone or both of the I2S interfaces are configured in the master mode, the master clock can be output to the external DAC/CODEC.
Both of the I2S interfaces have dedicated DMA controllers. PDM and BT PCM interfaces are supported.

Pulse Width Modulation (PWM)

The Pulse Width Modulation (PWM) controller can be used for driving digital motors and smart lights. The controller consists of PWM timers, the PWM operator and a dedicated capture sub-module. Each timer providestiming in synchronous or independent form, and each PWM operator generates a waveform for one PWM channel. The dedicated capture sub-module can accurately capture events with external timing.

Serial Peripheral Interface (SPI)

ESP32 features three SPIs (SPI, HSPI and VSPI) in slave and master modes in 1-line full-duplex and 1/2/4-linehalf-duplex communication modes. These SPIs also support the following general-purpose SPI features:

- Four modes of SPI transfer format, which depend on the polarity (CPOL) and the phase (CPHA) of the SPIclock
- Up to 80 MHz (The actual speed it can reach depends on the selected pads, PCB tracing, peripheral characteristics, etc.)
- Up to 64-byte FIFO

All SPIs can also be connected to the external flash/SRAM and LCD. Each SPI can be served by DMA controllers.

Accelerator

ESP32 is equipped with hardware accelerators of general algorithms, such as AES (FIPS PUB 197), SHA (FIPS PUB 180-4), RSA, and ECC, which support independent arithmetic, such as Big Integer Multiplication and Big Integer Modular Multiplication. The maximum operation length for RSA, ECC, Big Integer Multiply and Big IntegerModular Multiplication is 4096 bits.
The hardware accelerators greatly improve operation speed and reduce software complexity. They also supportcode encryption and dynamic decryption, which ensures that code in the flash will not be hacked.

On-board Switches & LED Indicator

EN

Reset button: pressing this button resets the system.

Boot

Download button: holding down the Boot button and pressing the EN button initiates the firmware download mode.Then user can download firmware through the serial port.

Switches & Indicators

The board also has a LED indicator which shows that the board has power.

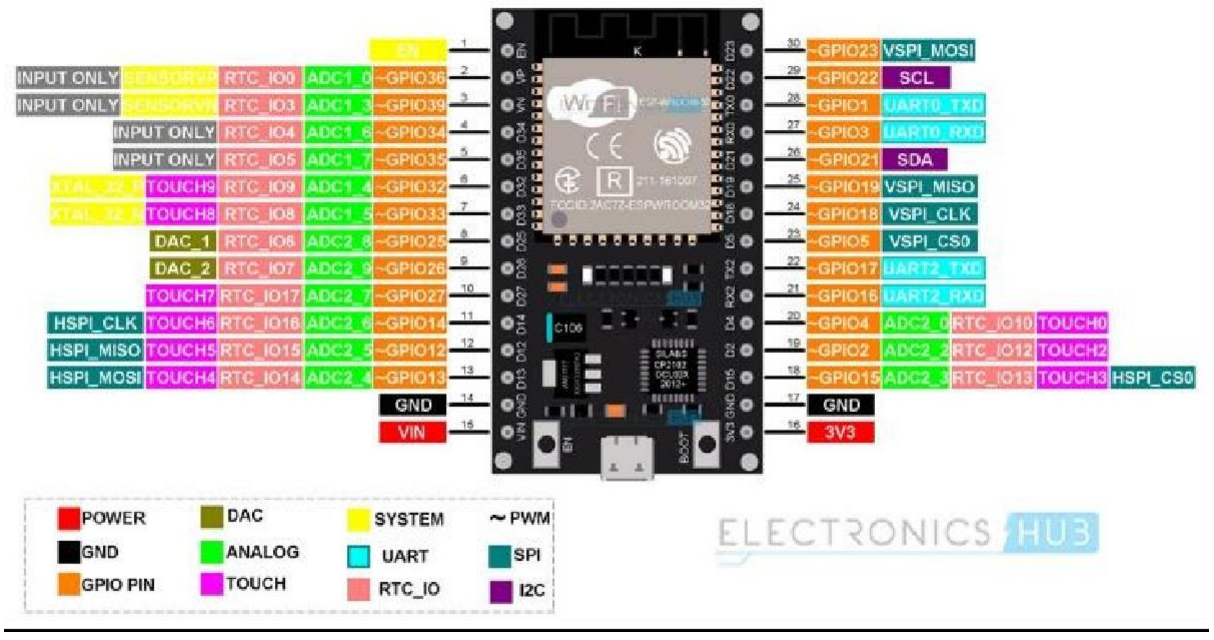
Serial Communication

The board includes CP2102 USB-to-UART Bridge Controller from Silicon Labs, which converts USB signal to serialand allows your computer to program and communicate with the ESP8266 chip.

- CP2102 USB-to-UART converter
- 4.5 Mbps communication speed

If you have an older version of CP2102 driver installed on your PC, we recommend upgrading now.

ESP – 32 Pinout


EN
INPUT ONLY SENSOR_VP RTC_IO0 ADC1_0 GPIO36
INPUT ONLY SENSOR_VN RTC_IO3 ADC1_3 GPIO39
INPUT ONLY RTC_IO4 ADC1_6 GPIO34
INPUT ONLY RTC_IO5 ADC1_7 GPIO35
XTAL_32_P TOUCH9 RTC_IO9 ADC1_4 GPIO32
XTAL_32_N TOUCH8 RTC_IO8 ADC1_5 GPIO33
DAC_1 RTC_IO6 ADC2_8 GPIO25
DAC_2 RTC_IO7 ADC2_9 GPIO26
TOUCH7 RTC_IO17 ADC2_7 GPIO27
HSPI_CLK TOUCH6 RTC_IO16 ADC2_6 GPIO14
HSPI_MISO TOUCH5 RTC_IO15 ADC2_5 GPIO12
HSPI_MOSI TOUCH4 RTC_IO14 ADC2_4 GPIO13
GND
VIN
GPIO23 VSPI_MOSI
GPIO22 SCL
GPIO1 UART0_TXD
GPIO3 UART0_RXD
GPIO21 SDA
GPIO19 VSPI_MISO
GPIO18 VSPI_CLK
GPIO5 VSPI_CS0
GPIO17 UART2_TXD
GPIO16 UART2_RXD
GPIO4 ADC2_0 RTC_IO10 TOUCH0
GPIO2 ADC2_2 RTC_IO12 TOUCH2
GPIO15 ADC2_3 RTC_IO13 TOUCH3 HSPI_CS0
GND
3V3
POWER
GND
GPIO PIN
DAC
ANALOG
TOUCH
SYSTEM
UART
RTC_IO
~ PWM
SPI
I2C
ELECTRONICS HUB

GPIO:

The most commonly used peripheral is the GPIO. ESP32 has 34 GPIO pins with each pin carrying out more than onefunction (only one will be active). You can configure a pin as either a GPIO or an ADC or an UART in the program.

ADC and DAC pins are predefined and you have to use the manufacturer specified pins. But other functions likePWM, SPI, UART, I2C etc. can be assigned to any GPIO pin through program.

RTC GPIO:

ESP32 has 16 RTC GPIOs, which are part of the RTC Low-Power subsystem. These pins can be used to wake ESP32from deep sleep as external wake-up source.

ADC:

ESP32 has two 12-bit SAR Analog to Digital Converter Modules with 8-channels and 10-channels each. So, ADC1 andADC2 blocks combined together have 18 channels of 12-bit ADC.

With 12-bit resolution, the output Digital values will be in the range of 0 – 4093.

DAC:

ESP32 Microcontroller has two independent 8-bit Digital to Analog Converter channels to convert digital values toanalog voltage signals. The DAC has internal resistor network and uses power supply as input reference voltage.

The following two GPIO Pins are associated with DAC functionalities.

DAC1 — GPIO25DAC2 — GPIO26
Capacitive Touch GPIOs

The ESP32 SoC has 10 capacitive-sensing GPIOs, which can detect variations in capacitance on a pin due to touchingor approaching the GPIO Pin with a finger or stylus. These Touch GPIOs can be used in implementing capacitive touch pads, without any additional hardware.

SPI:

The ESP32 Wi-Fi chip features three SPI blocks (SPI, HSPI and VSPI) in both master and slave modes. SPI is used tointerface with Flash Memory. So, you have two SPI interfaces.

I2C:

There are two I2C interfaces in ESP32 with complete flexibility on assigning pins i.e., SCL and SDA pins for both I2Cinterfaces can be assigned in the program by the user.

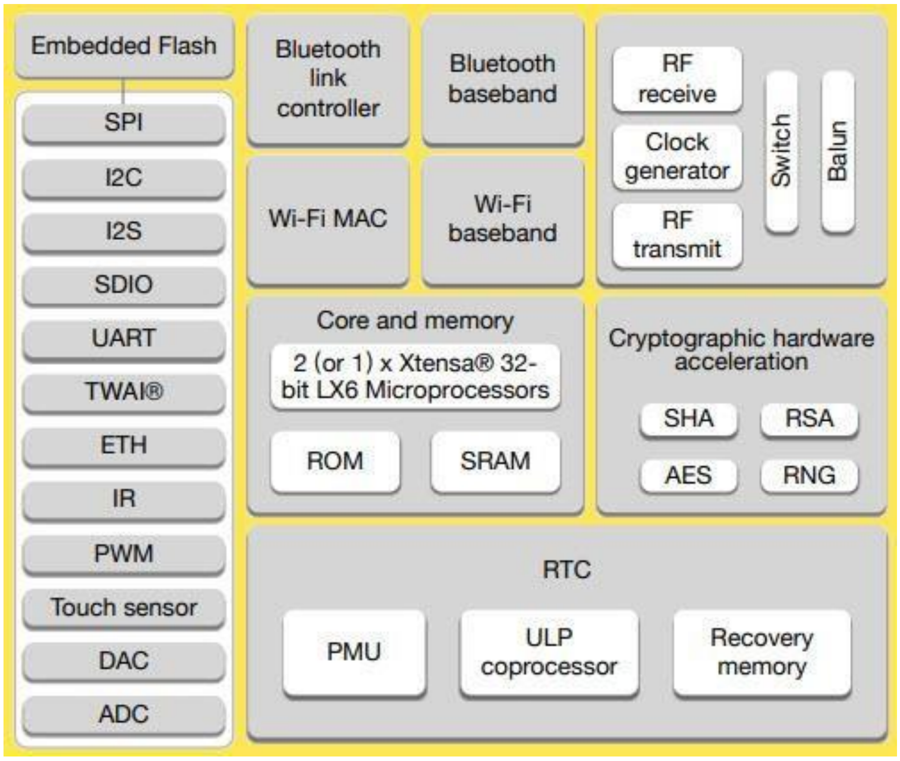
If you are using Arduino IDE, then the default I2C pins are:

SDA – GPIO21SCL – GPIO22

PWM:

The PWM Controller in ESP32 have 16 independent PWM waveform channels with configurable frequency and dutycycle. The PWM waveform can be used to drive motors and LEDs. You can configure the PWM signal frequency, channel, GPIO pin and also the duty cycle.

ESP-32 ARCHITECTURE:



❖ Bread Board:

A breadboard, often referred to as a "bread board" or "protoboard," is a fundamental tool in electronics for creating and testing electronic circuits. Here's a short note on a breadboard:

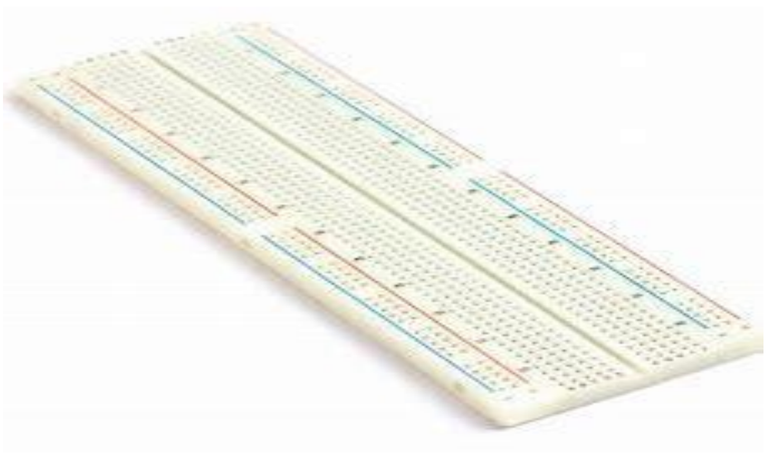
A breadboard is a rectangular board with a grid of holes that are used for prototyping and constructing electronic circuits without the need for soldering. The board typically consists of two main sections:

- Terminal Strips: Rows of metal clips or terminals run along the sides of the board. These are used for connecting power sources, such as batteries or external power supplies, and for providing voltage and ground connections.
- Tie Points: The main body of the board consists of a grid of holes, with each hole connected internally in a specific pattern. Components, such as resistors, capacitors, and integrated circuits, can be inserted into these holes, and their leads can be connected by jumper wires.

Features of the Bread Board:

- Reusable and Versatile: Components can be easily inserted and removed, making the breadboard reusable for various projects.
- No Soldering Required: Components are connected by inserting their leads into the holes, eliminating the need for soldering during the prototyping phase.
- Rapid Prototyping: Engineers and hobbyists use breadboards to quickly prototype and test circuit designs before committing to a more permanent arrangement.
- Ease of Troubleshooting: Breadboards facilitate easy circuit modification and troubleshooting since components can be adjusted or replaced without damaging the board.
- Jumper Wires: Jumper wires are used to create electrical connections between components on the breadboard, allowing for the creation of complex circuits.
- Common Patterns: Breadboards typically follow a standard layout for the connection of power and ground, making it easy to follow a consistent wiring convention.

While breadboards are invaluable for experimentation and initial circuit design, they are not suitable for permanent installations. Once a circuit design is finalized and tested, it is usually transferred to a printed circuit board (PCB) for a more durable and compact solution.



❖ Jumper Wires :

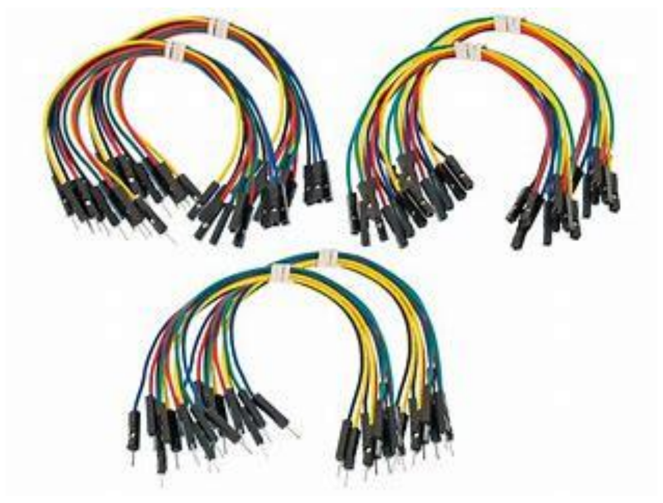
Jumper wires are essential components in electronics that are used to create electrical connections between different points on a breadboard or within a circuit. These wires are flexible, usually made of stranded copper with insulation, and come in various lengths and colors.

Features of the Jumper Wires:

- Connection on Breadboards: Jumper wires are commonly used on breadboards to connect various components, such as resistors, capacitors, and integrated circuits, by plugging them into the holes on the breadboard.
- Colors and Lengths: Jumper wires are available in different colours, helping to distinguish between different connections in a circuit. Common colours include red, black, blue, green, and yellow. The color-coding aids in organization and troubleshooting.
- They are also available in different lengths, ranging from a few inches to several feet, providing flexibility for different circuit layouts.
- Flexibility and Stranded Copper: Jumper wires are designed to be flexible, allowing for easy manipulation and placement on a breadboard or within a circuit.
- The wires are often made of stranded copper, which enhances flexibility and durability. The copper core provides a good conductor for electricity.
- Prototyping and Circuit Design: Jumper wires are extensively used during the prototyping and testing phases of circuit design. They allow engineers and hobbyists to quickly and easily make connections and modifications on a breadboard without the need for soldering.
- Breadboarding and Educational Use: Jumper wires are particularly popular in educational settings and electronics workshops, where they facilitate hands-on learning of circuitry and electronics without the complexities of soldering.
- Cable Management: Proper cable management using jumper wires helps keep a circuit organized and makes it easier to identify and troubleshoot connections.

Types of the Jumper Wires:

- **Male-to-Male:** Both ends have exposed male pins, suitable for connecting two points with male headers, such as on a breadboard or an Arduino board.
- **Male-to-Female:** One end has a male pin, and the other has a female connector, allowing for connections between male headers and components with female receptacles.
- **Female-to-Female:** Both ends have female connectors, useful for extending connections between components with male pins.



❖ Light Emitting Diode (LED):

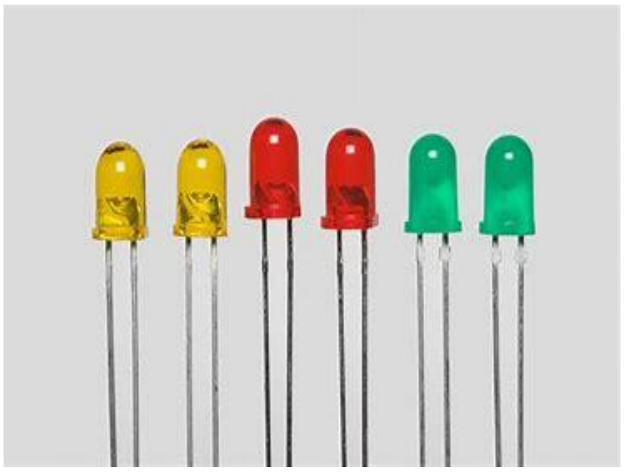
A Light Emitting Diode, commonly known as LED, is a semiconductor device that emits light when an electric current passes through it. LEDs operate based on the principle of electroluminescence. When an electric current is applied to the semiconductor material within the LED, it releases energy in the form of photons, producing light.LEDs are a type of solid-state lighting that has gained widespread popularity due to their energy efficiency, long lifespan, and compact size.They are used in a variety of applications, including indicator lights, displays, lighting fixtures, and electronic components.

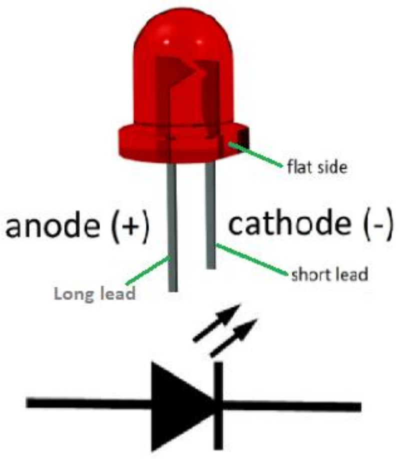
Features of the Jumper Wires:

- Efficiency: LEDs are highly energy-efficient, converting a significant portion of electrical energy into visible light. They consume less power compared to traditional light sources.
- Long Lifespan: LEDs have a much longer operational life than incandescent bulbs or fluorescent lamps. Their lifespan is typically measured in tens of thousands of hours.
- Size and Form Factor: LEDs come in various sizes and form factors, ranging from tiny surface-mount devices (SMDs) to larger, high-power LEDs. This flexibility allows for diverse applications.
- Instantaneous Illumination: LEDs illuminate instantly when powered, without the warm-up time required by some traditional light sources.
- Colour Variety: LEDs are available in a wide range of colors, including red, green, blue, and white.

Colour-changing LEDs use multiple diodes to produce a spectrum of colors.

- Low Heat Emission: LEDs emit very little heat in comparison to traditional bulbs, making them suitable for applications where heat generation is a concern.





❖ Servo Motor:

A servo motor is a rotary or linear actuator that allows for precise control of angular or linear position, velocity, and acceleration. It is a crucial component in various applications, providing accurate and controlled motion. A servo motor is a type of closed-loop system that incorporates a feedback mechanism to maintain the desired position or speed. It operates based on the principle of error correction, continuously adjusting its position based on the feedback from a sensor.

<u>Key Components:</u>

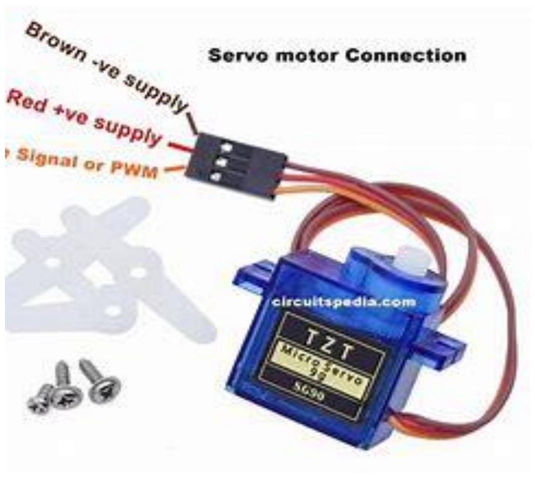
- Motor: The core component responsible for generating mechanical motion. Servo motors can be either DC or AC motors.
- Feedback Device: Typically, a potentiometer or an encoder provides feedback on the motor's current position. This information is crucial for the control system.
- Control Circuit: The control circuit interprets the feedback information and sends signals to adjust the motor's position or speed accordingly.
- Gears: Gears are often used in servo motors to optimize torque, precision, and speed.

Working Principle:

- Reference Signal: The user sets a reference signal, indicating the desired position or speed of the servo motor.
- Feedback System: The feedback device continuously monitors the actual position or speed of the motor.
- Error Calculation: The control circuit calculates the difference between the reference signal and the actual position or speed, known as the error signal.
- Correction Signal: Based on the error signal, the control circuit generates a correction signal that adjusts the motor's output to minimize the error.
- Closed-Loop Operation: This continuous feedback loop ensures that the servo motor maintains precise control over its position or speed, even in the presence of external disturbances.

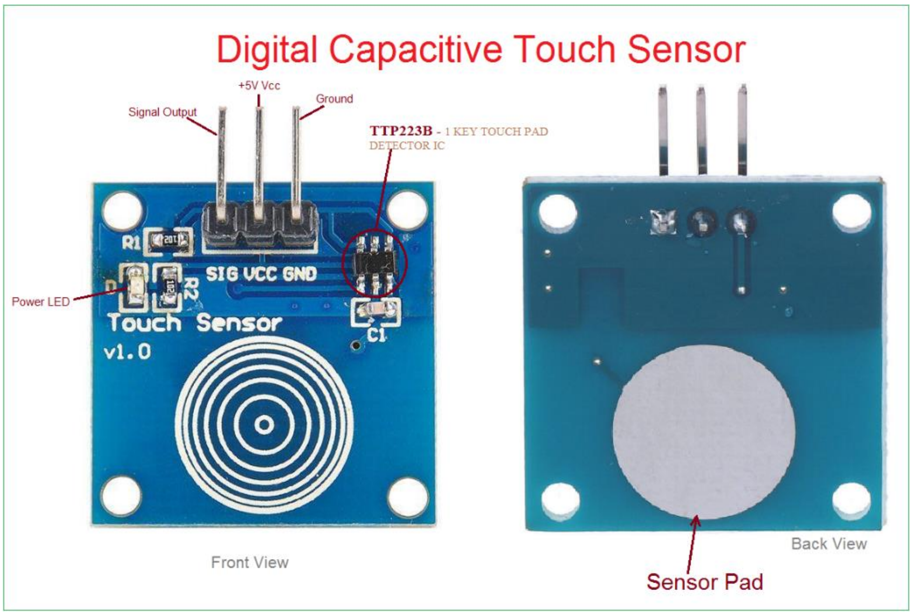
Types of Servo Motors:

- DC Servo Motors: Use a DC motor as the driving force. Commonly found in small to medium-sized applications.
- AC Servo Motors: Use an AC motor and are often more powerful than DC servo motors. Suitable for applications requiring higher torque.
- Linear Servo Motors: Provide linear motion instead of rotational motion. Used in applications where linear precision is essential.



❖ Touch Switch:

A touch sensor, also known as a touch switch or tactile sensor, is a device that detects physical touch or contact. It is widely used in various applications, ranging from consumer electronics to industrial control systems. A touch sensor is designed to detect the presence or touch of a user's finger or any conductive object, initiating a response or action in a connected system.



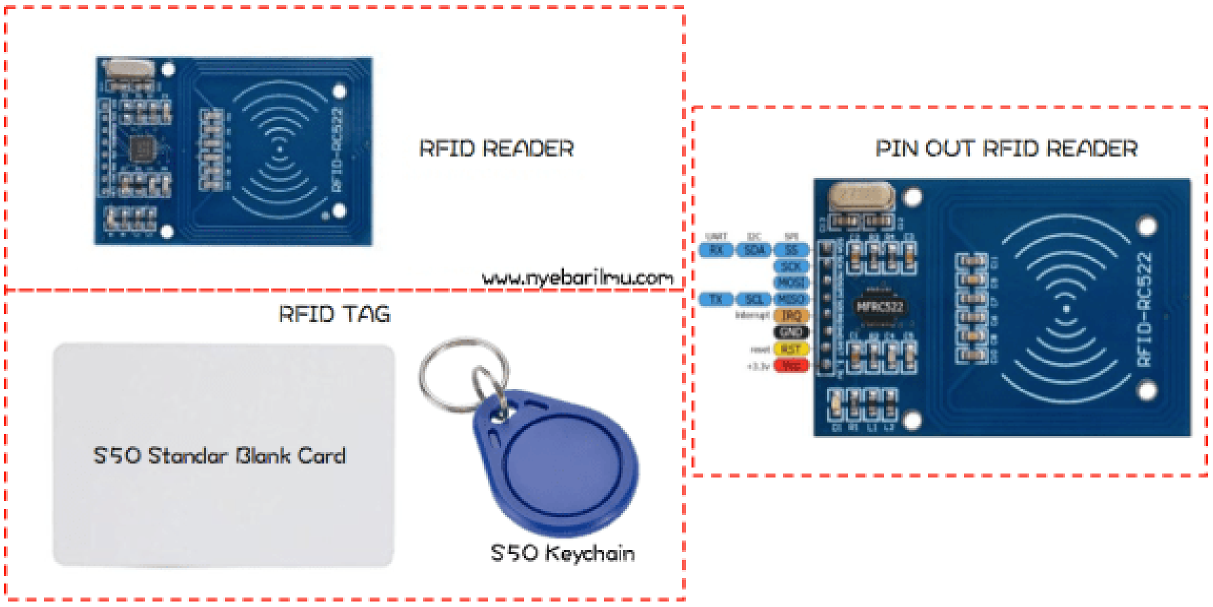
❖ RFID:

RFID, which stands for Radio-Frequency Identification, is a technology that uses radio waves to identify and track objects, people, or animals. It consists of RFID tags, RFID readers, and a system for processing and managing the collected data.

<u>Working Principle:</u>

- Tag Identification: When an RFID tag comes into the range of an RFID reader, the reader sends out a radio-frequency signal.
- Powering the Tag: Passive tags draw power from the reader's signal, activating the tag's microchip. Active tags have their own power source, enabling longer-range communication.
- Data Transmission: The RFID tag responds by transmitting its unique identifier and, if applicable, additional data back to the reader.
- Data Processing: The reader captures the data, and the middleware processes and interprets it.

The information is then sent to the back-end system for further analysis and integration with other business processes.



❖ DHT 11:

The DHT11 is a low-cost digital temperature and humidity sensor widely used in various applications. It is commonly utilized in hobbyist projects, weather stations, and home automation systems due to its simplicity and ease of use. The DHT11 sensor is a basic digital sensor that measures temperature and humidity levels. It comes in a compact form with a built-in resistive humidity element and a thermistor for temperature measurement. DHT11 provides digital output, making it easy to interface with microcontrollers without the need for additional analog-to-digital conversion.

Key Specifications:

- Temperature Range:DHT11 typically measures temperatures in the range of 0 to 50 degrees Celsius (32 to 122 degrees Fahrenheit).
- Humidity Range: It measures relative humidity in the range of 20% to 90%.
- Accuracy: The accuracy of the DHT11 is within a few degrees Celsius for temperature and around 5% for humidity.
- Operating Voltage: Typically operates at 3.3V or 5V, making it compatible with a wide range of microcontrollers.

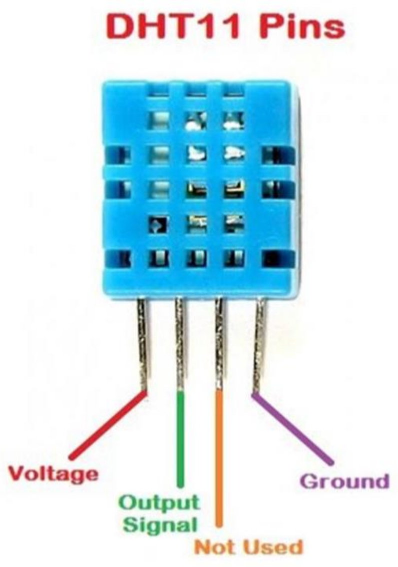
Working Principle:

- The DHT11 uses a capacitive humidity sensor and a thermistor to measure temperature.
- The humidity sensor changes capacitance based on the moisture in the air, and the thermistor changes resistance with temperature.
- The sensor converts these changes into digital signals, which can be read by a microcontroller.

Pin Configuration:

The DHT11 sensor usually has three pins:

- VCC (Power): Connect to the power supply (3.3V or 5V).
- Data: Connect to a digital pin on the microcontroller for data transmission.
- GND (Ground): Connect to the ground of the power supply.



❖ Buzzer:

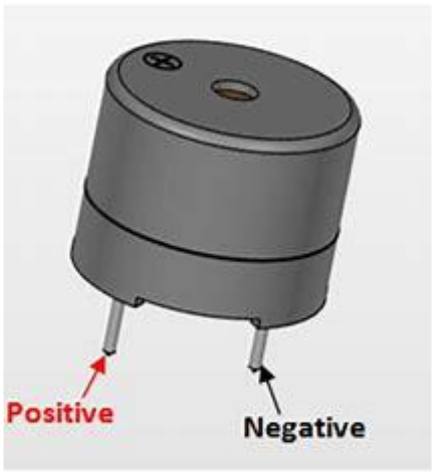
A buzzer is a simple electronic device that produces a buzzing or beeping sound when an electrical current passes through it. It is commonly used for various purposes, including providing audible alerts, notifications, or alarms in electronic systems.

Key Features:

- Operating Voltage: Buzzers are available in various voltage ratings, typically ranging from 3V to 24V. The voltage requirement depends on the specific buzzer model.
- Sound Output: Measured in decibels (dB), indicating the loudness of the sound produced by the buzzer.
- The sound output can vary between different buzzer models.
- Frequency: Buzzers produce sound at a specific frequency or a range of frequencies. The frequency is measured in hertz (Hz).
- Types of Sounds: Buzzers can emit continuous tones, intermittent beeps, or even different patterns of sounds based on their design and application.

Circuit Connection:

Buzzers typically have two terminals: positive (+) and negative (-). Connect the positive terminal to the power source and the negative terminal to the ground. Applying the appropriate voltage triggers the buzzer to produce sound.

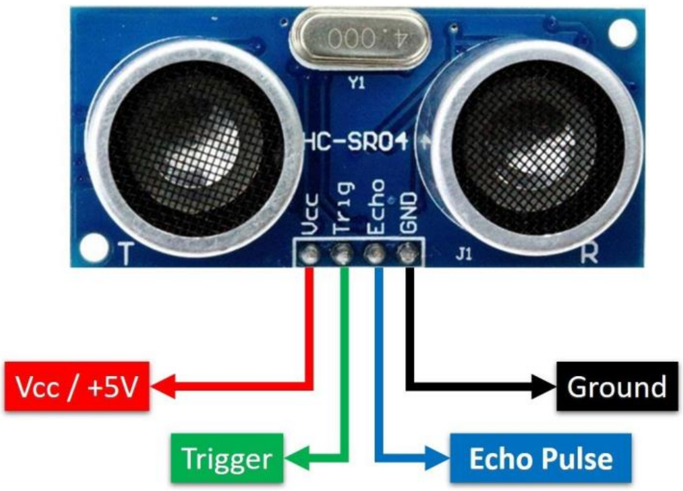


❖ Ultrasonic Sensor:

An ultraviolet (UV) sensor is a device designed to detect ultraviolet light, a part of the electromagnetic spectrum with wavelengths shorter than visible light. UV sensors find applications in various fields, including environmental monitoring, industrial processes, and consumer electronics. UV sensors are electronic devices that measure the intensity of ultraviolet radiation in a given environment. They are crucial for monitoring UV exposure, which can be harmful to living organisms and materials.

Working Principle:

- UV sensors typically use semiconductor materials that generate a current when exposed to UV light.
- The amount of current produced is proportional to the intensity of UV radiation.
- The sensor converts the generated current into a measurable electrical signal, which can be processed by microcontroller or other electronics.



❖ Door Sensor:

A door sensor, also known as a door/window contact or magnetic contact sensor, is a device used to detect the status of a door or window—whether it is open or closed. These sensors are commonly employed in security systems, home automation, and various applications where monitoring access points is important. Door sensors are composed of two main components: a sensor and a magnet. These components are typically installed on a door frame and the door itself, or on a window frame and the window itself. The sensor and magnet are positioned close to each other, and their relative proximity or separation is used to determine the door or window status.

Working Principle:

- When the door or window is closed, the magnet and sensor are in close proximity, allowing the sensor to detect the magnetic field.
- This proximity completes an electrical circuit, indicating a closed door or window.
- When the door or window is opened, the magnet moves away from the sensor, breaking the circuit and signaling that the door or window is open.

❖ IR sensor:

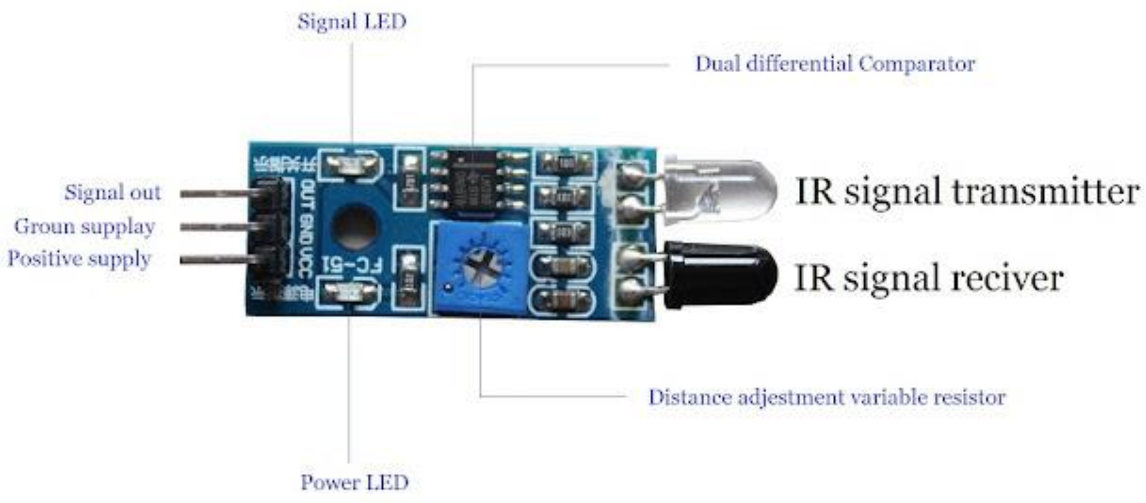
An Infrared (IR) sensor, often referred to as an infrared receiver or IR detector, is a device that receives and detects infrared signals. These sensors are commonly used in various applications, including remote controls, proximity sensors, and motion detectors. IR sensors are designed to detect infrared radiation, which is not visible to the human eye but can be emitted by objects based on their temperature.

Key Features:

- Sensitivity: IR sensors can be designed to be sensitive to specific wavelengths of infrared radiation.
- Modulation Support: IR receivers often support modulation, allowing them to receive signals that are encoded or modulated for specific purposes (e.g., remote control signals).
- Range: The range of an IR sensor depends on factors such as the power of the IR source and the sensitivity of the sensor.

Working Principle:

- IR sensors work by detecting changes in the intensity of infrared radiation reaching the sensor.
- In the case of IR receivers, they are designed to respond to modulated IR signals commonly used in remote controls.
- The sensor converts the detected IR radiation into an electrical signal, which can be processed by electronic circuits.



❖ 16 x 2 LCD:

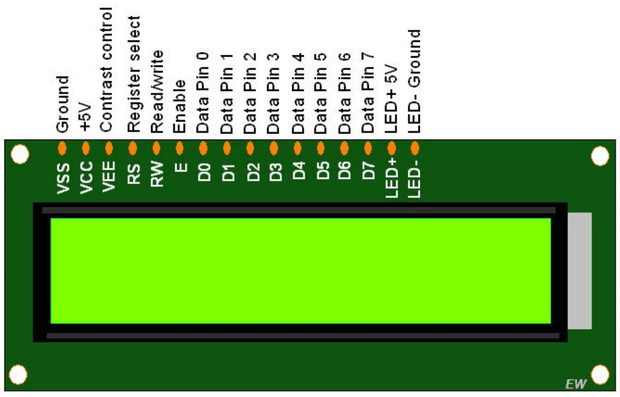
A 16x2 LCD (Liquid Crystal Display) is a commonly used alphanumeric display module that can display 16 characters per line and has 2 lines. It is widely used in embedded systems, microcontroller-based projects, and other applications where a simple text display is needed. The 16x2 LCD module consists of a liquid crystal display panel with 16 columns and 2 rows of characters, allowing the display of 16 characters on each line. Each character position is typically a 5x8 dot matrix, allowing the display of alphanumeric characters, symbols, and some custom characters.

Pin Configuration:
- VSS (Ground)
- VDD (Power Supply)
- V0 (Contrast Adjustment)
- RS (Register Select)
- R/W (Read/Write)
- E (Enable)
- D0 (Data Bit 0, if using 8-bit mode)
- D1 (Data Bit 1, if using 8-bit mode)

- D2 (Data Bit 2, if using 8-bit mode)
- D3 (Data Bit 3, if using 8-bit mode)
- D4 (Data Bit 4)
- D5 (Data Bit 5)
- D6 (Data Bit 6)
- D7 (Data Bit 7)
- A (Anode of Backlight, optional)
- K (Cathode of Backlight, optional)

Working Principle:

- The 16x2 LCD is controlled by sending commands and data through its pins.
- The RS (Register Select) pin is used to select whether the data sent is an instruction or actual character data.
- The R/W (Read/Write) pin determines the direction of data flow (read or write).
- The E (Enable) pin is used to enable the LCD for data/command reception.



❖ I2C Driver:

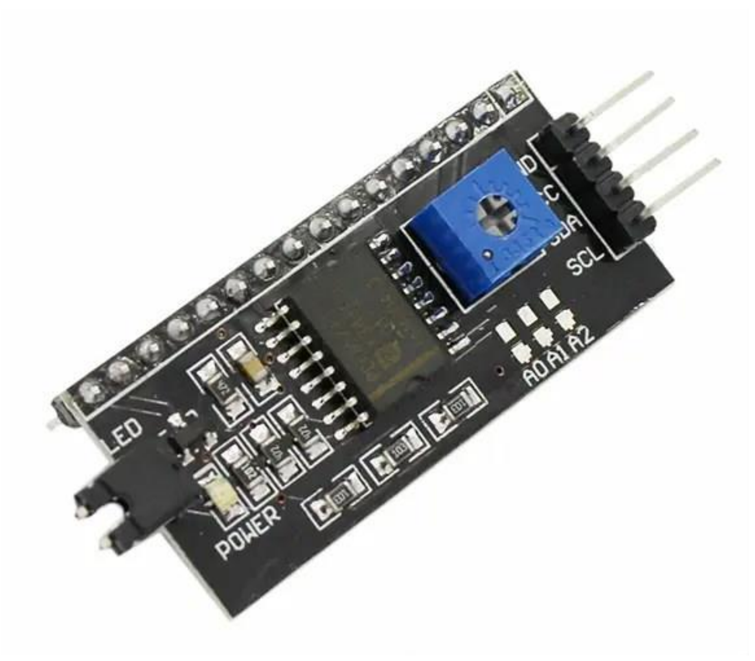
I2C, or Inter-Integrated Circuit, is a serial communication protocol that allows multiple devices to communicate with each other using a shared bus. In the context of microcontrollers and embedded systems, an I2C driver is a software module or library that facilitates communication with I2C devices.

I2C Driver Functions:

- Initialization: The I2C driver initializes the hardware peripherals (such as I2C controllers) on the microcontroller.
- Configuration: Configuration settings, such as the clock speed (baud rate), addressing mode,

and other parameters, can be set by the driver.

- Data Transmission: The driver provides functions to transmit data over the I2C bus. This includes sending data to specific device addresses.
- Data Reception: Functions for receiving data from I2C devices are provided by the driver. This involves specifying the target device and the number of bytes to read.
- Error Handling: The driver may include error-handling mechanisms to deal with issues such as bus errors, device not responding, or other communication problems.
- Interrupt Handling: In some cases, the driver supports interrupt-driven communication, allowing the microcontroller to handle I2C events asynchronously.



❖ MQ2 Gas Sensor:

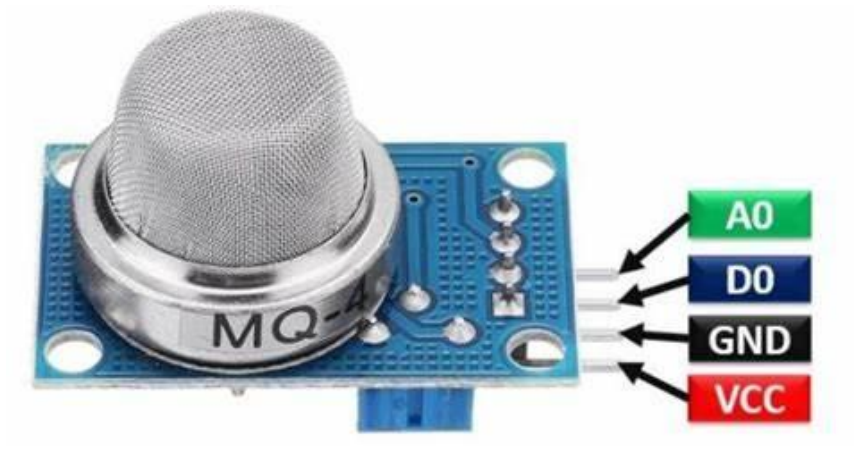
The MQ-2 sensor is a gas sensor module that detects various gases such as methane, propane, butane, alcohol, carbon monoxide, and smoke. It is commonly used in applications such as gas leakage detection, fire detection, and air quality monitoring. The MQ-2 sensor is a semiconductor-based gas sensor that operates on the principle of resistive changes in the presence of specific gases. It is sensitive to a range of combustible gases, as well as some other gases that affect its conductivity.

Key Features:

- Gas Detection: MQ-2 is designed to detect multiple gases, including methane, propane, butane, alcohol, carbon monoxide, and smoke.
- Analog Output: The sensor provides an analog voltage output that varies based on the

concentration of the detected gas.

- Heating Element: The sensor has a built-in heating element to raise its operating temperature, enhancing sensitivity.
- Resistance Changes: When exposed to gas, the sensor's resistance changes, leading to a change in the output voltage.
- Calibration: Calibration may be required for accurate gas concentration measurements, and the sensitivity can be adjusted through a potentiometer on the module.



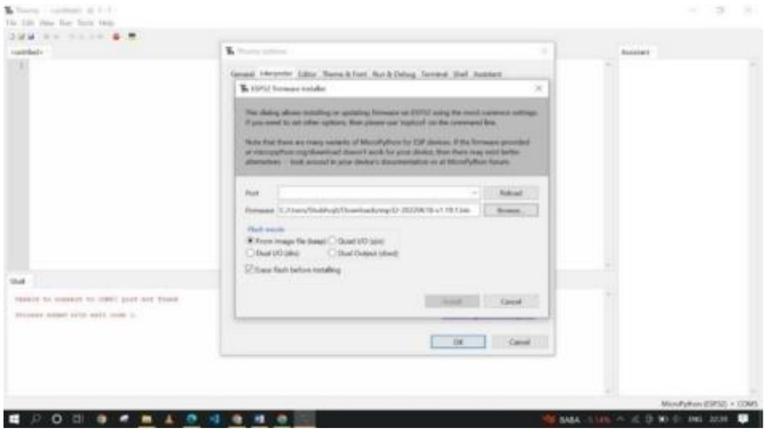
SOFTWARE USED:

❖ THONNY:

Thonny is an integrated development environment for Python that is designed for beginners. It was created by Aivar Annamaa, an Estonian programmer. It supports different ways of stepping through code, step-by-step expression evaluation, detailed visualization of the call stack and a mode for explaining the concepts of references and heap.

INSTALLATION OF MICROPYTHON USING THONNY IDE:

Step – 1:



ESP32 firmware installer

This dialog allows installing or updating firmware on ESP32 using the most common settings. If you need to set other options, then please use 'esptool' on the command line.

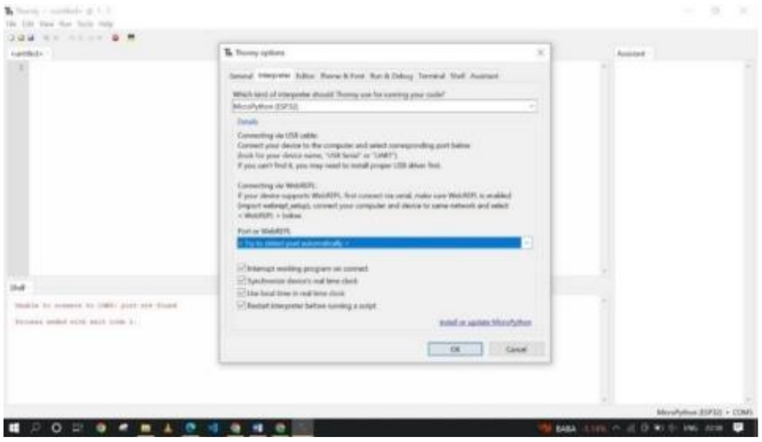
Note that there are many variants of MicroPython for ESP devices. If the firmware provided at micropython.org/download doesn't work for your device, then there may exist better alternatives — look around in your device's documentation or at MicroPython forum.

Port:
Firmware: [illegible]

Flash mode
[✓ From image file (keep)] Quad I/O (qio)
Dual I/O (dio) Dual Output (dout)

☑ Erase flash before installing

Step – 2:



Which kind of interpreter should Thonny use for running your code?
MicroPython (ESP32)

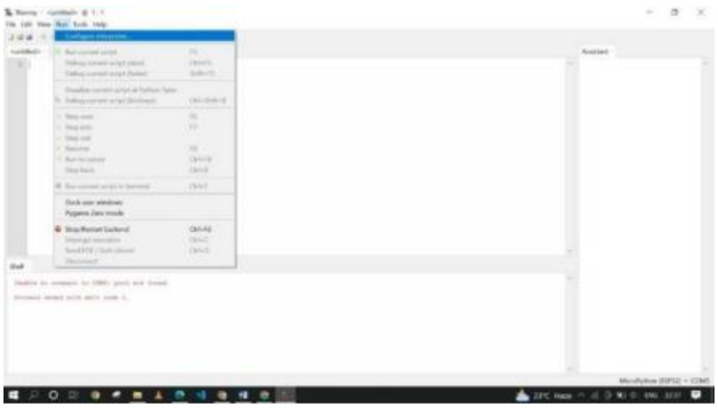
Details
Connecting via USB cable:
Connect your device to the computer and select corresponding port below
(look for your device name, "USB Serial" or "UART").
If you can't find it, you may need to install proper USB driver first.

Connecting via WebREPL:
If your device supports WebREPL, first connect via serial, make sure WebREPL is enabled
(import webrepl_setup), connect your computer and device to same network and select
< WebREPL > below

Port or WebREPL
< Try to detect port automatically >

☑ Interrupt working program on connect
☑ Synchronize device's real time clock
☑ Use local time in real time clock
☑ Restart interpreter before running a script

Step – 3:



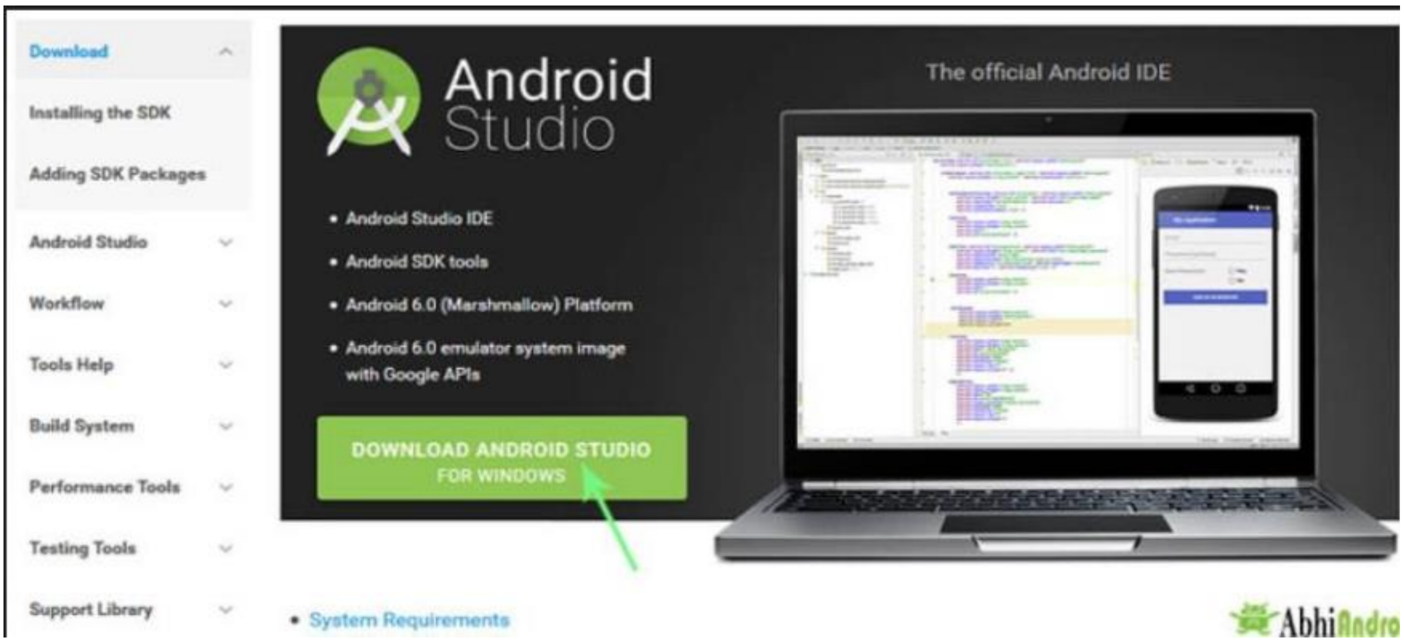
❖ ANDROID STUDIO:

Android Studio is the official integrated development environment (IDE) for Android app development, provided by Google. It is based on JetBrains' IntelliJ IDEA software and is tailored specifically for Android development. Here is an overview covering various aspects of Android Studio:

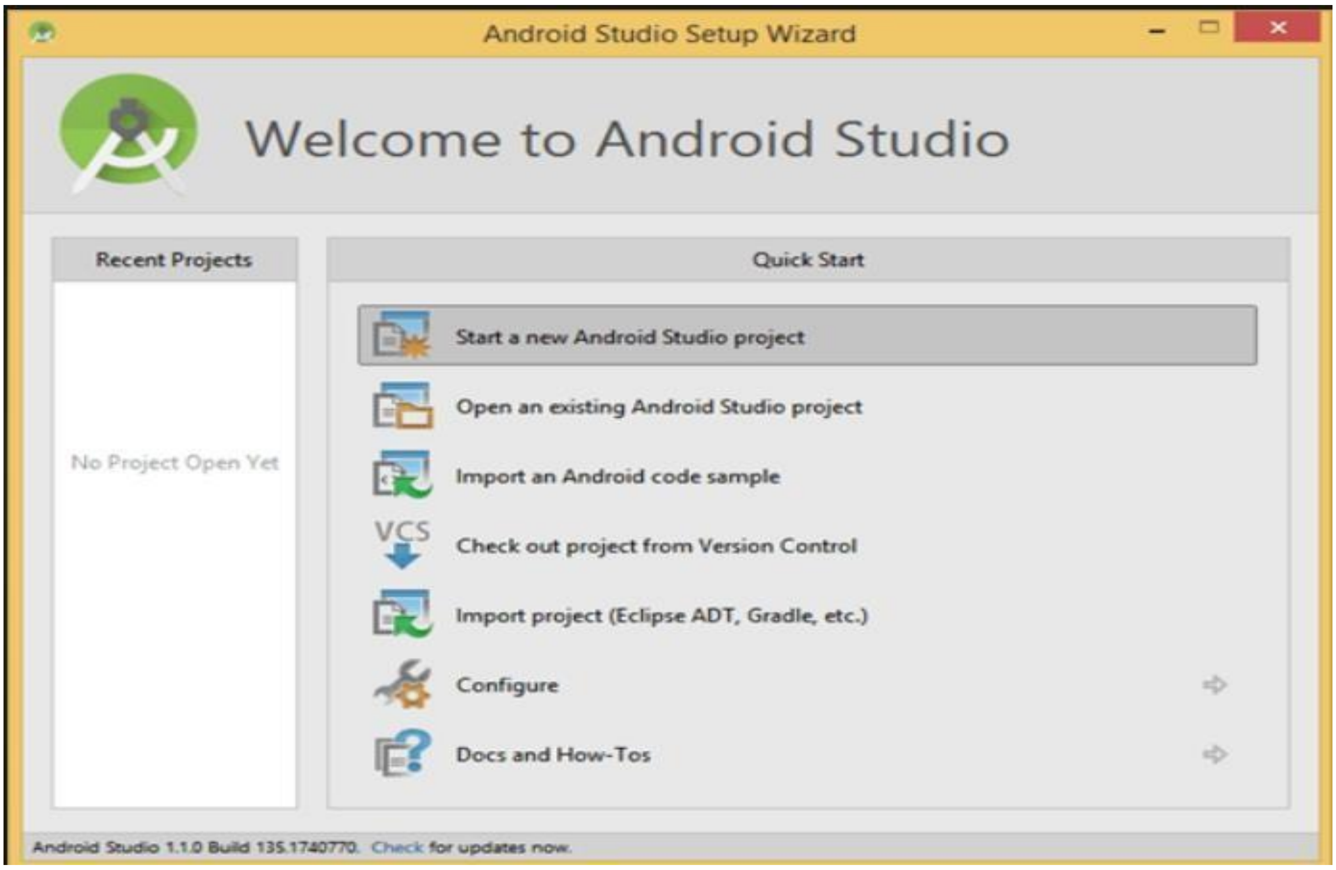
- Installation: Download and install Android Studio from the official website.
- User Interface: Android Studio has a user-friendly interface with various panels like the Project Explorer, Editor, and Gradle Console.
- Project Structure: Android projects are organized into modules, each containing source code, resources, and dependencies. The app module typically contains the main code and resources for your app.
- Gradle Build System: Android Studio uses Gradle as the build system for managing project dependencies and building the app.
- Code Editor: Offers a rich code editor with features like auto-completion, code analysis, and inline documentation. Supports Java, Kotlin, and C++ programming languages.
- Layout Editor: Provides a visual interface for designing app layouts using XML. Supports drag-and-drop UI design.
- Resource Manager: Manages app resources such as images, layouts, strings, and more. Allows for localization and customization.
- Emulator: Android Studio includes an emulator for testing apps on virtual devices with various Android versions and screen sizes.
- Device Debugging: Supports debugging apps on physical Android devices via USB. Integrated debugging tools for finding and fixing issues.

Installation process step by step:

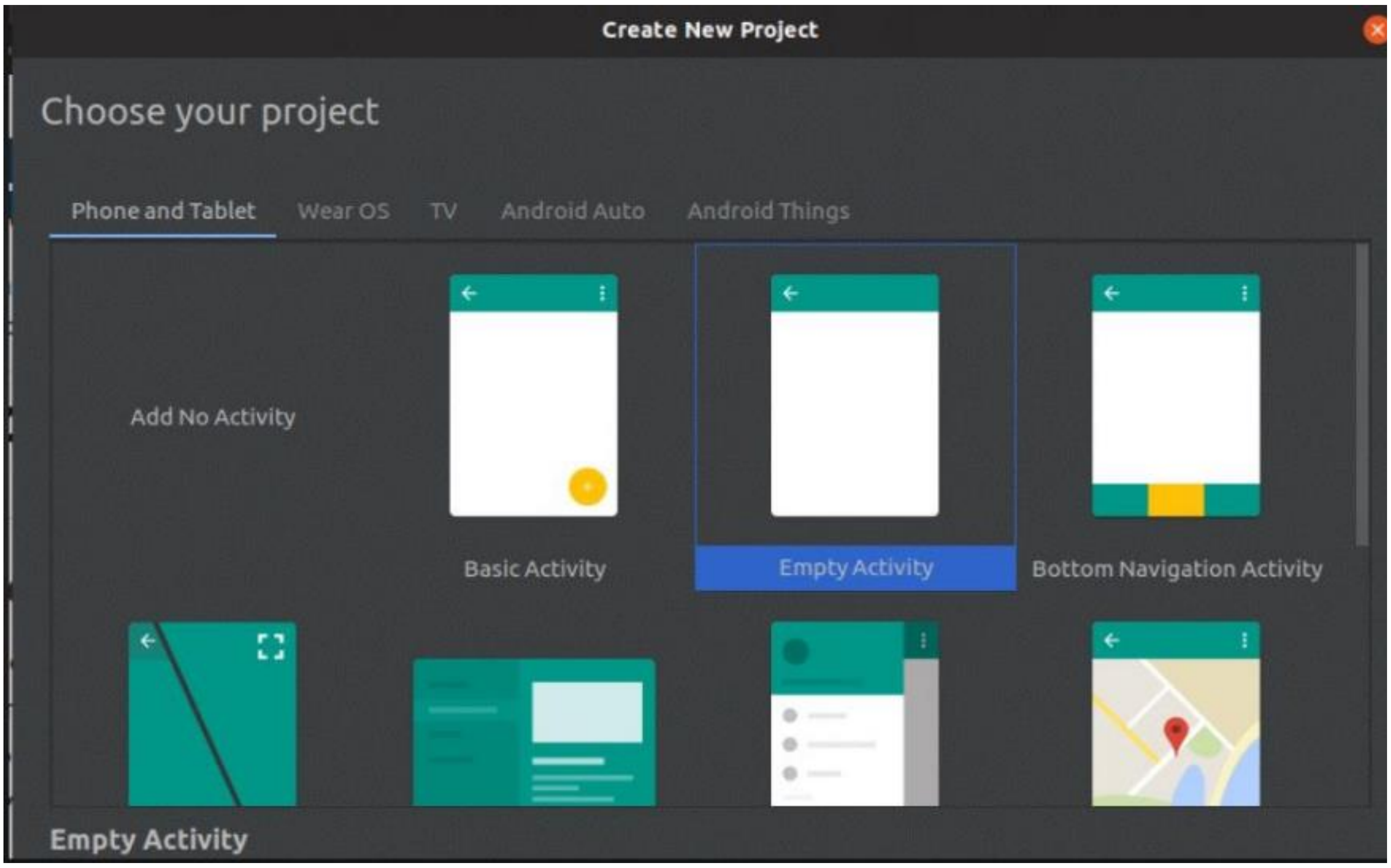
Step-1:



Step-2:


Android Studio Setup Wizard
Welcome to Android Studio
Recent Projects
No Project Open Yet
Quick Start
Start a new Android Studio project
Open an existing Android Studio project
Import an Android code sample
Check out project from Version Control
Import project (Eclipse ADT, Gradle, etc.)
Configure
Docs and How-Tos
Android Studio 1.1.0 Build 135.1740770. Check for updates now.

Step-3:



❖ FIREBASE:

Firebase is a comprehensive mobile and web application development platform developed by Firebase, Inc., which was acquired by Google in 2014. It offers a wide range of services and tools that facilitate various aspects of app development, including authentication, real-time database, cloud storage, hosting, and more. Here are some key features and services provided by Firebase:

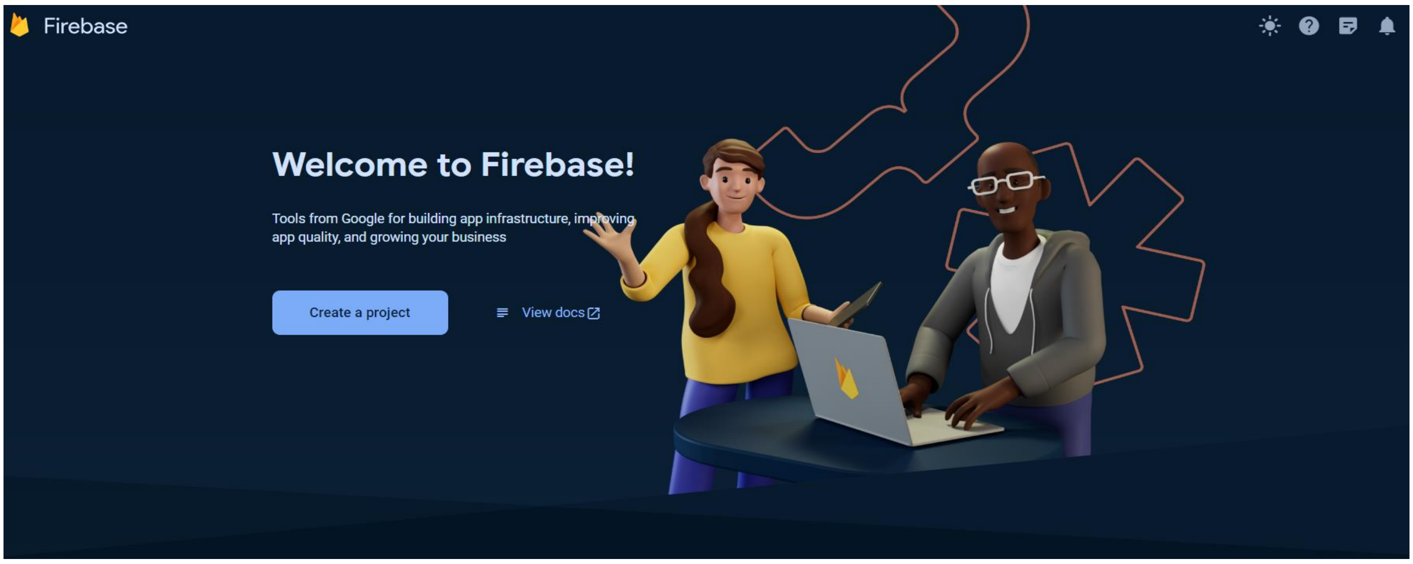
- Realtime Database: Firebase provides a NoSQL cloud database that allows developers to store and sync data in real-time. It uses a JSON-like data structure and is suitable for applications that require real-time updates.
- Authentication: Firebase Authentication simplifies the process of user authentication for your app. It supports authentication through email/password, social media logins (Google, Facebook, Twitter, etc.), and more. Firebase provides a set of tools and services to help secure user authentication, including user management, authentication state tracking, and various authentication providers.
- Cloud Firestore: Firestore is another NoSQL database offered by Firebase, providing a more scalable and flexible solution compared to the Realtime Database. It supports more complex queries and scales horizontally.
- Cloud Functions: Firebase Cloud Functions enable you to run backend code in response to events triggered by Firebase features and HTTPS requests. This allows you to execute server-side logic without managing your own servers.
- Hosting: Firebase Hosting allows you to deploy and host your web applications quickly and securely. It includes features like SSL, global CDN, and automatic scaling.
- Cloud Storage: Firebase Cloud Storage provides secure and scalable cloud storage for your app's user-generated content such as images, videos, and audio files.

- Machine Learning: Firebase ML Kit offers ready-to-use APIs for common machine learning tasks, such as text recognition, image labeling, face detection, and more, making it easier to incorporate machine learning features into your app.
- Analytics and Performance Monitoring: Firebase Analytics allows you to gain insights into user behavior and app performance. It also includes features like Crashlytics for crash reporting.

Firebase is widely used by developers for its ease of use, real-time capabilities, and integration with other Google Cloud services. It is particularly popular among startups and small to medium-sized development teams.

Installation process step by step:

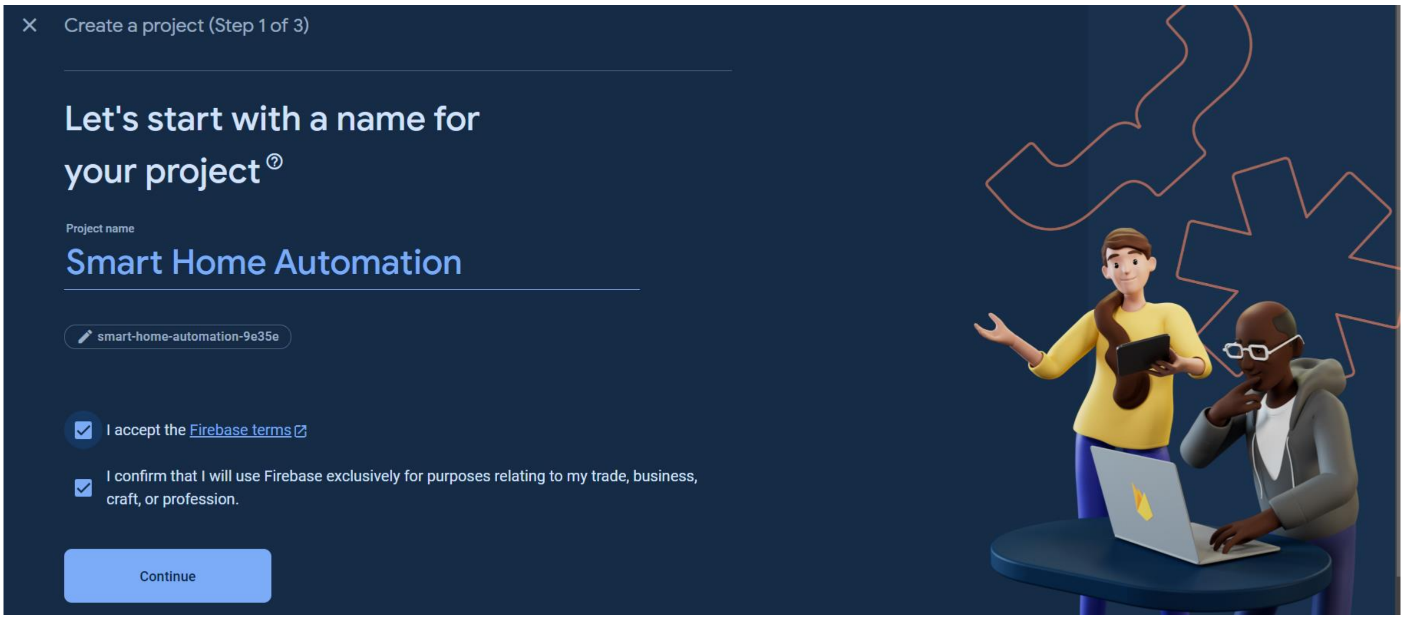
Step-1:



Welcome to Firebase!

Tools from Google for building app infrastructure, improving app quality, and growing your business

Step-2:



Create a project (Step 1 of 3)

Let's start with a name for your project

Project name

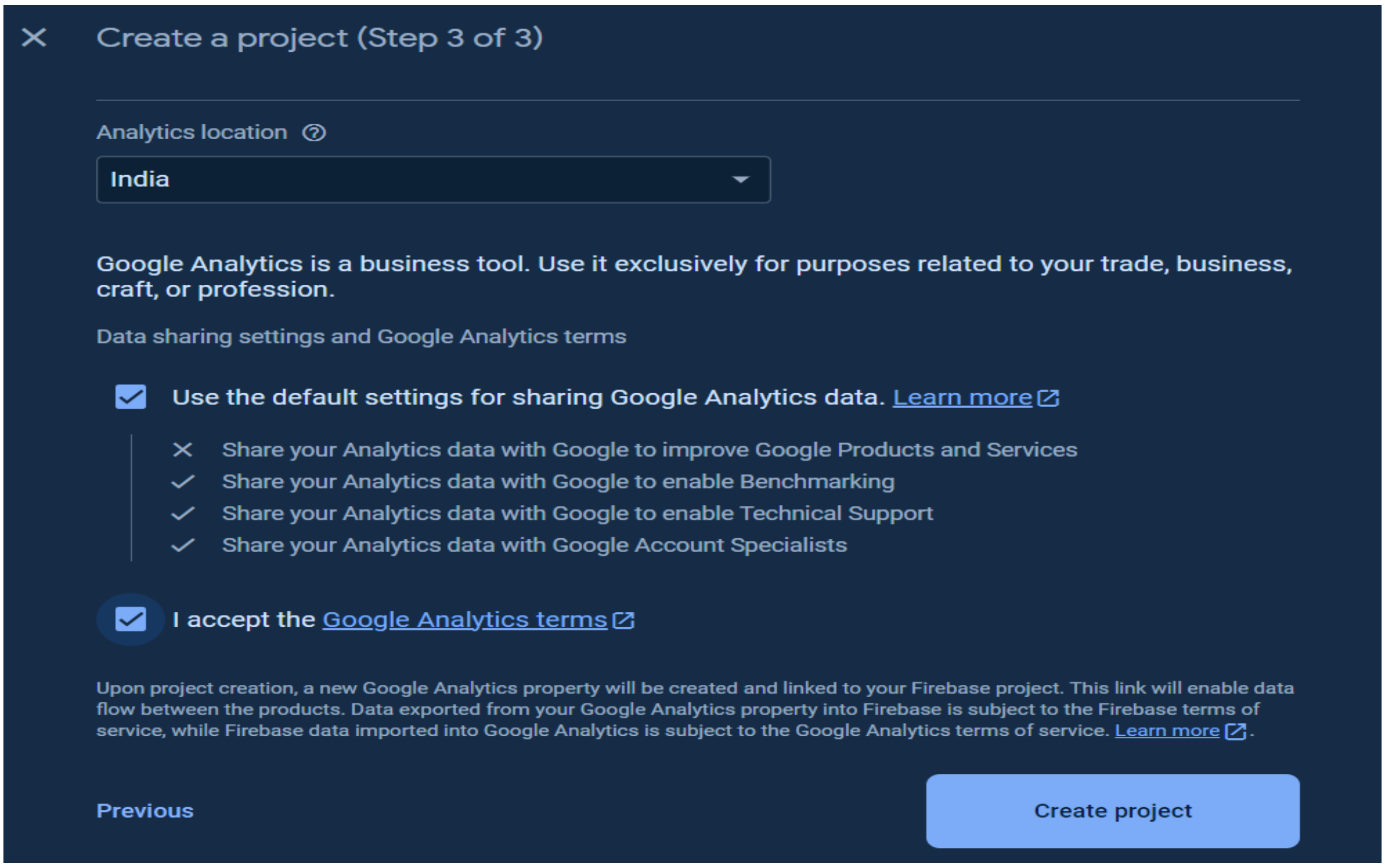
Smart Home Automation

smart-home-automation-9e35e

☑ I accept the Firebase terms

☑ I confirm that I will use Firebase exclusively for purposes relating to my trade, business, craft, or profession.

Step-3:



Create a project (Step 3 of 3)

Analytics location

India

Google Analytics is a business tool. Use it exclusively for purposes related to your trade, business, craft, or profession.

Data sharing settings and Google Analytics terms

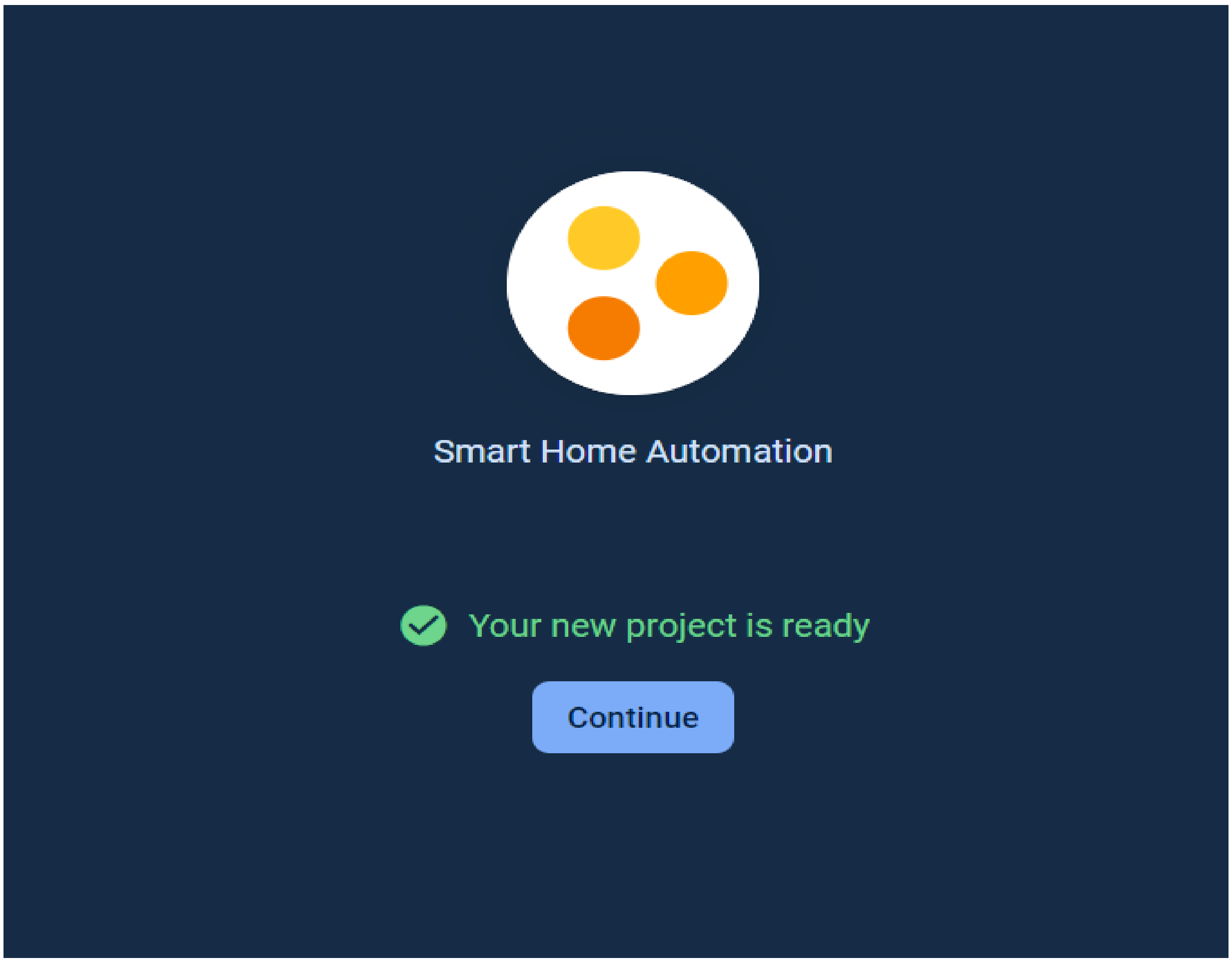
✓ Use the default settings for sharing Google Analytics data. Learn more

- ✗ Share your Analytics data with Google to improve Google Products and Services
- ✓ Share your Analytics data with Google to enable Benchmarking
- ✓ Share your Analytics data with Google to enable Technical Support
- ✓ Share your Analytics data with Google Account Specialists

✓ I accept the Google Analytics terms

Upon project creation, a new Google Analytics property will be created and linked to your Firebase project. This link will enable data flow between the products. Data exported from your Google Analytics property into Firebase is subject to the Firebase terms of service, while Firebase data imported into Google Analytics is subject to the Google Analytics terms of service. Learn more.

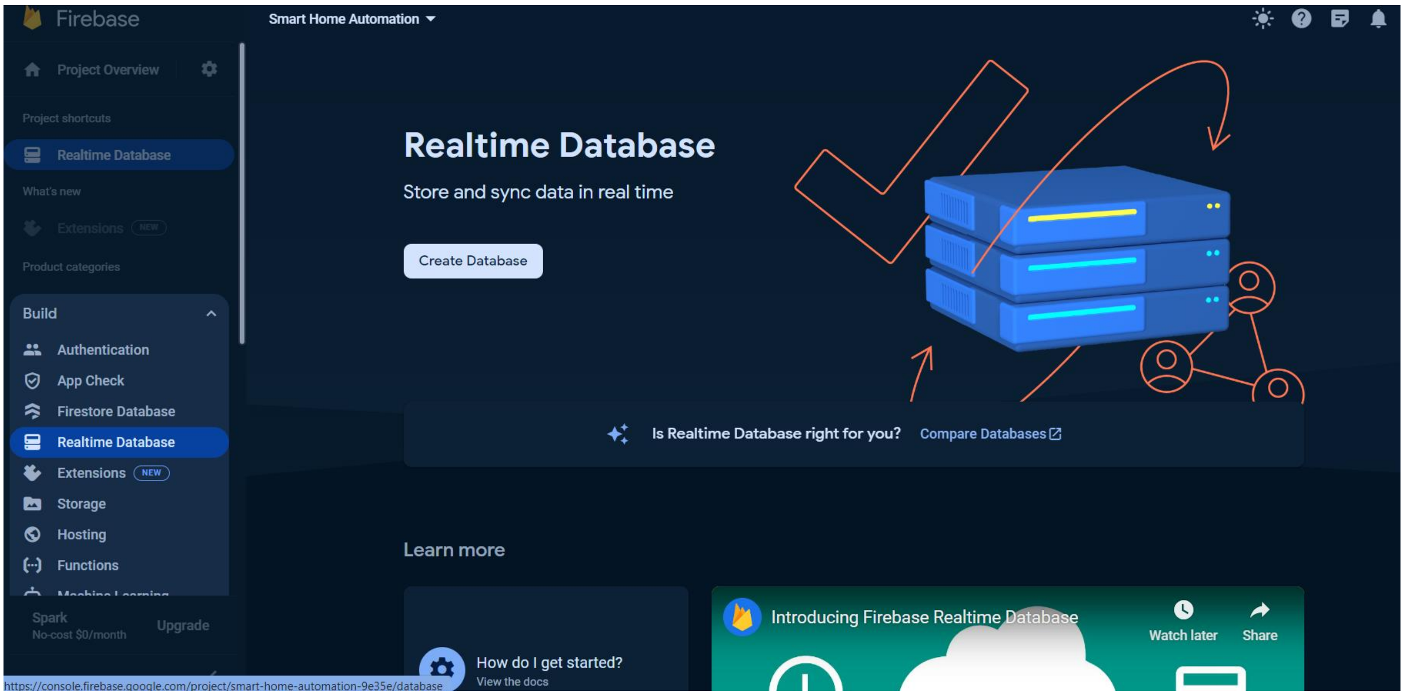
Step-4:



Smart Home Automation

✓ Your new project is ready

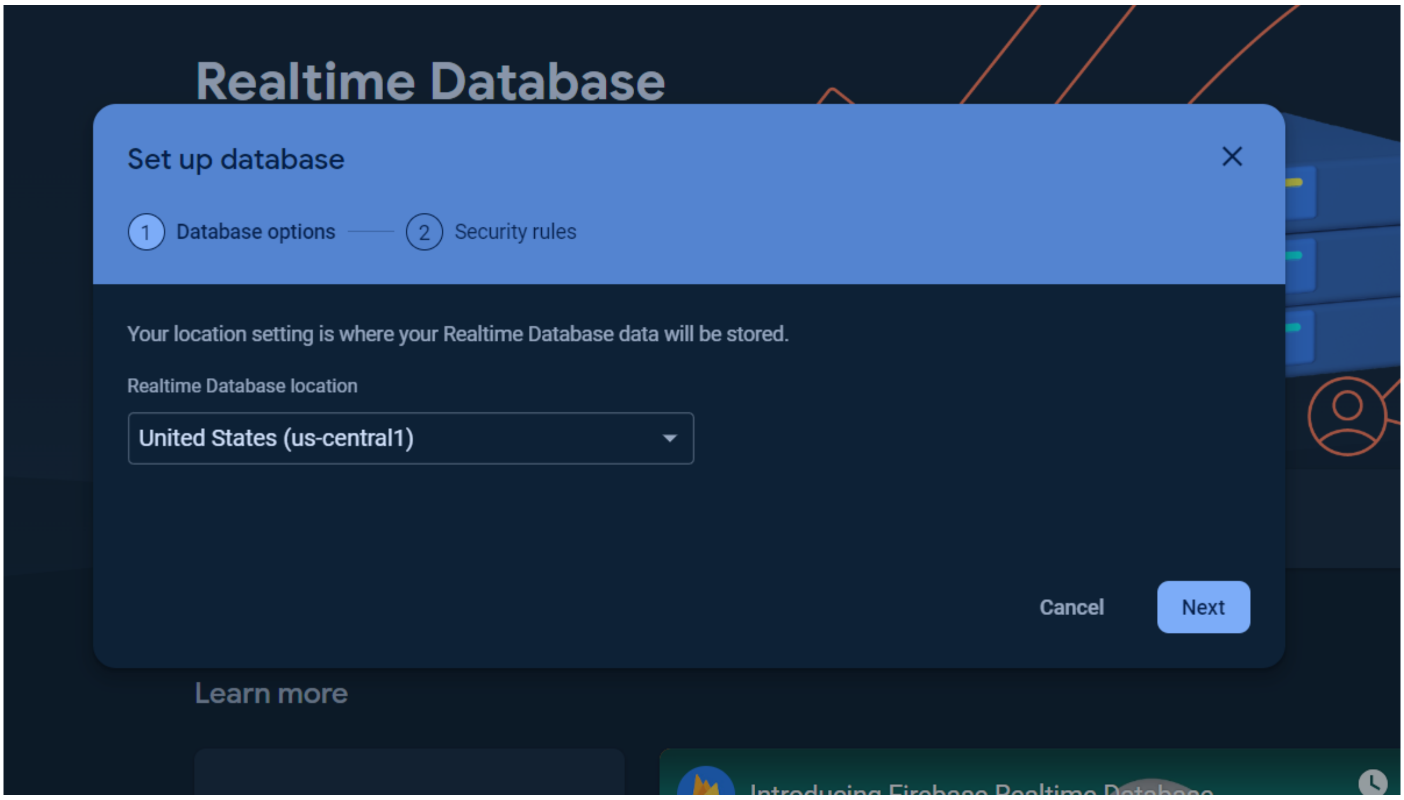
Step-5:



Realtime Database

Store and sync data in real time

Step-6:

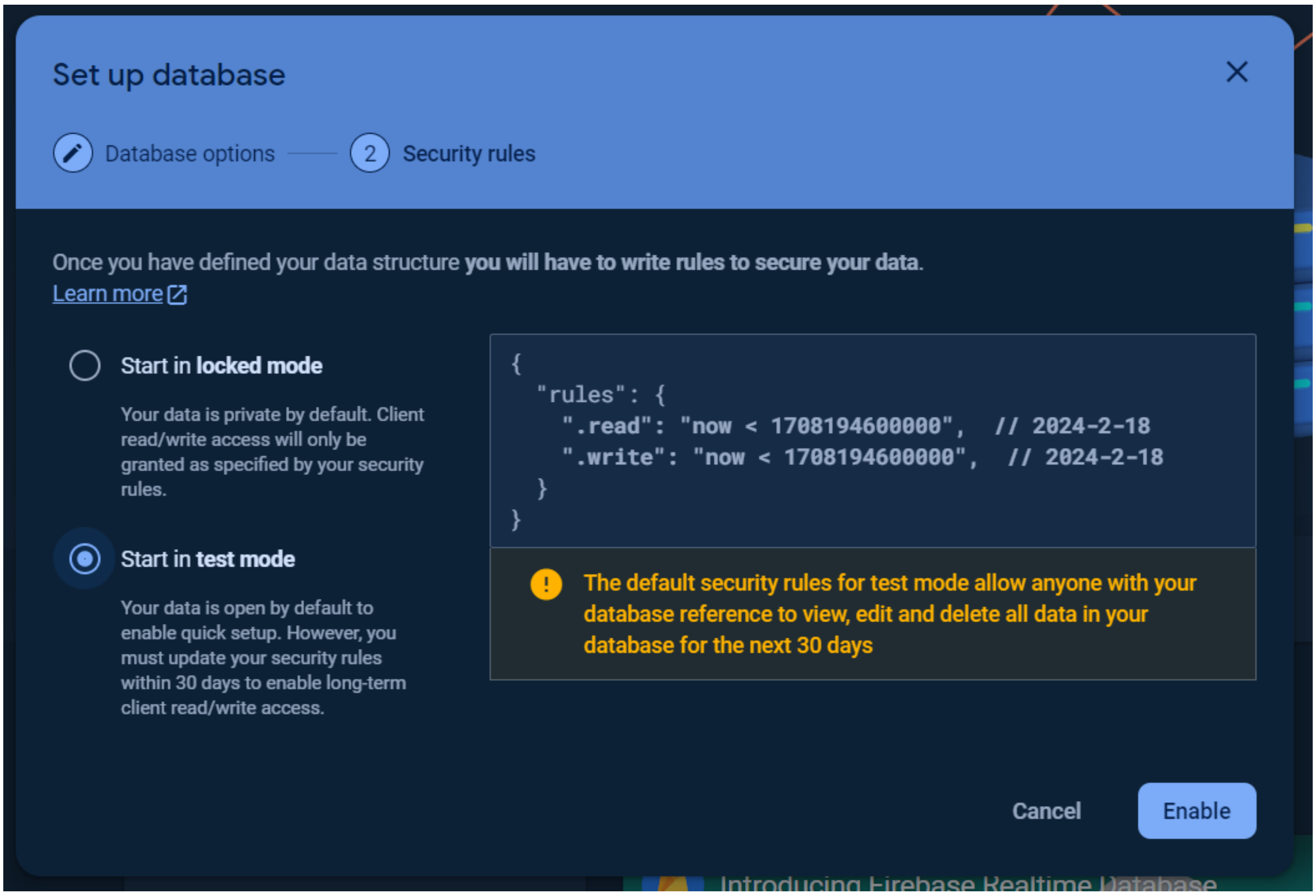


Set up database

Your location setting is where your Realtime Database data will be stored.

Realtime Database location: United States (us-central1)

Step-7:



Set up database

Database options — 2 Security rules

Once you have defined your data structure **you will have to write rules to secure your data.**
Learn more

Start in **locked mode**

Your data is private by default. Client read/write access will only be granted as specified by your security rules.

[✓ Start in **test mode**]

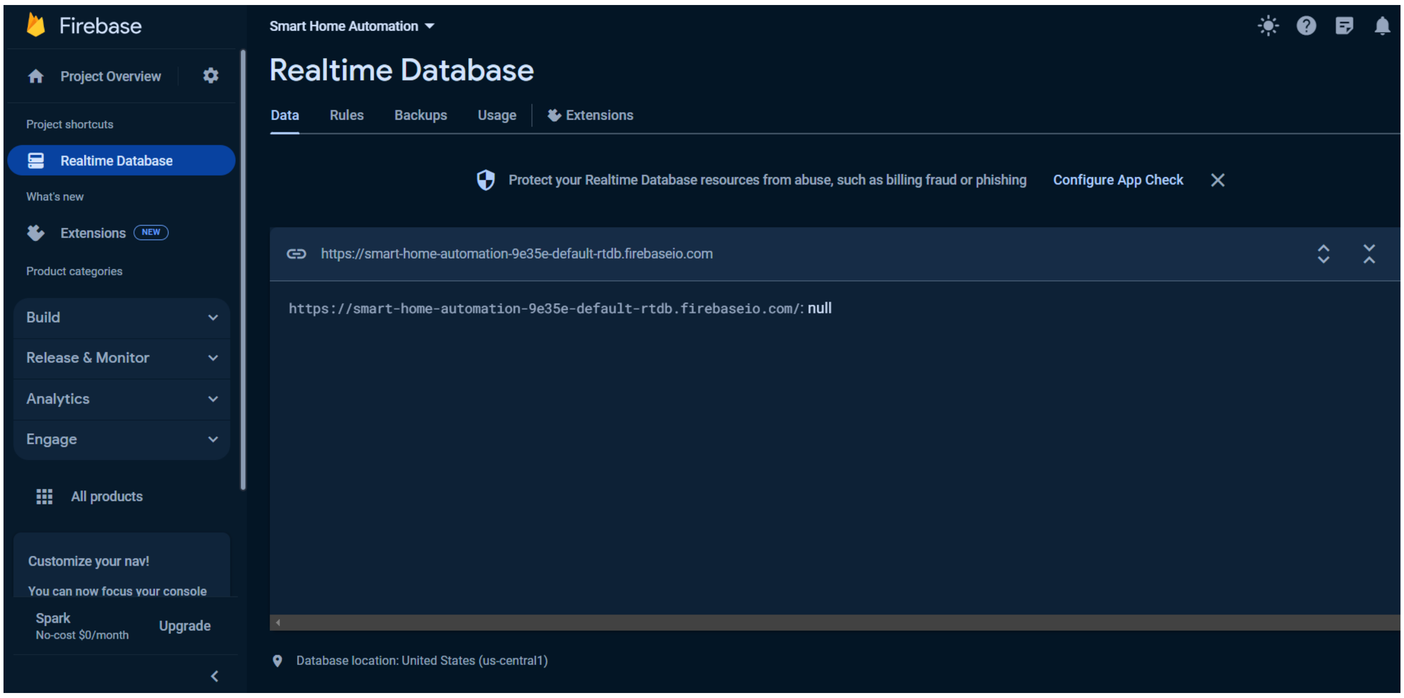
Your data is open by default to enable quick setup. However, you must update your security rules within 30 days to enable long-term client read/write access.

```
{
  "rules": {
    ".read": "now < 1708194600000",  // 2024-2-18
    ".write": "now < 1708194600000",  // 2024-2-18
  }
}
```

The default security rules for test mode allow anyone with your database reference to view, edit and delete all data in your database for the next 30 days

Cancel Enable

Step-8:



Realtime Database

Data Rules Backups Usage Extensions

Protect your Realtime Database resources from abuse, such as billing fraud or phishing Configure App Check

https://smart-home-automation-9e35e-default-rtdb.firebaseio.com

```
https://smart-home-automation-9e35e-default-rtdb.firebaseio.com/: null
```

Database location: United States (us-central1)

CODE:

❖ Main Code File:

```
import con

from machine import Pin, ADC, PWM, SPI, SoftI2C

from time import sleep

import dht

from hcsr04 import HCSR04

import ufirebase as firebase

from mfrc522 import MFRC522

from servo import Servo

import urequests

from lcd_api import LcdApi

from i2c_lcd import I2cLcd


I2C_ADDR = 0x27

totalRows = 2

totalColumns = 16


BASE_URL = 'https://api.openweathermap.org/data/2.5/weather?lat=22.571870190911607&lon=88.37140765745805&appid=af786b16ba952de2c0e7e3ef755ac358&units=metric'



URL='smart-home-automation-5e945-default-rtdb.firebaseio.com/'

url_lpg='smart-home-automation-5e945-default-rtdb.firebaseio.com/smart home automation/lpg'

url_db='smart-home-automation-5e945-default-rtdb.firebaseio.com/smart home automation/doorbell'

url_dl='smart-home-automation-5e945-default-rtdb.firebaseio.com/smart home automation/doorlock'

url_water='smart-home-automation-5e945-default-rtdb.firebaseio.com/smart home automation/water'

url_AC='smart-home-automation-5e945-default-rtdb.firebaseio.com/smart home automation/AC'

url_light='smart-home-automation-5e945-default-rtdb.firebaseio.com/smart home automation/light'

url_fan='smart-home-automation-5e945-default-rtdb.firebaseio.com/smart home automation/fan'
```

```
def get_weather_data():
    try:
        response = urequests.get(BASE_URL)
        data = response.json()
        return data
    except Exception as e:
        print("Error fetching weather data:", e)
        return None


def display_weather(weather_data):
    if weather_data:
        description = weather_data['weather'][0]['description']
        temp = weather_data['main']['temp']
        return description,temp
    else:
        return None


def display_suggestion(weather_description):
    suggestion = ""
    if "rain" in weather_description.lower():
        suggestion = "Take Umbrella!"


    elif "mist" in weather_description.lower():
        suggestion = "Low Visibility!"


    elif "haze" in weather_description.lower():
        suggestion = "Low Visibilty!"


    elif "snow" in weather_description.lower():
        suggestion = "Dress Warmly!"
    elif "cloud" in weather_description.lower():
        suggestion = "Cloudy Day!"

    else:
        suggestion = "Enjoy the Weather!"

    return suggestion

while True:

    #LPG Detection
    gas = ADC(34)
    buz = Pin(2,Pin.OUT)
    tv = 2500
    gas_value = gas.read()
    print(gas_value)
    if (gas_value>tv):
        firebase.put(url_lpg,"Alert")
        buz.value (1)
    else:
        firebase.put(url_lpg,"Normal")
        buz.value(0)
    #sleep(0.2)



    #Temperature Control
    dht_pin_number = 21
    dht_sensor = dht.DHT11(Pin(dht_pin_number))
    threshold_temp = 26
    led = Pin(26,Pin.OUT)
```

```
try:
    dht_sensor.measure()
    temperature = dht_sensor.temperature()
    humidity = dht_sensor.humidity()
    print(f"Temperature: {temperature}°C, Humidity: {humidity}%")
    if (temperature >threshold_temp):
        firebase.put(url_AC,"ON")
        led.value(1)
    else:
        firebase.put(url_AC,"OFF")
        led.value(0)
except Exception as e:
    pass
#sleep(0.2)



#Water Level in Tank
sensor= HCSR04(trigger_pin=5,echo_pin=25, echo_timeout_us=1000000)
tank_d=7
distance=sensor.distance_cm()
print(distance,'cm')
if distance>3 and distance<=tank_d:
    per=100-((100/tank_d)*distance)
    firebase.put(url_water,"Water Level: "+str(per)+"%")
    print("Water Level:",per,"%")
elif distance<=3:
    firebase.put(url_water,"Tank Full!")
    print("Tank Full!")
else:
    firebase.put(url_water,"Tank Empty!")
    print("Tank Empty!")
#sleep(0.2)

#Door Lock
lock = Pin(15,Pin.IN)
firebase.put(url_dl,lock.value())
print(lock.value())
#sleep(0.2)

#Door Bell
bell = Pin(23,Pin.IN)
firebase.put(url_db,bell.value())
print(bell.value())
#sleep(0.2)


#Light & Fan
Data_light ={}
#Data_fan={}
led_pwm = PWM(Pin(32), freq=1000)
#fan_pwm = PWM(Pin(26), freq=1000)
Data_light=firebase.get(url_light)
#Data_fan=firebase.get(url_fan)
#print(Data_fan)
led_pwm.duty(10*Data_light)
#fan_pwm.duty(10*Data_fan)


#RFID
spi = SPI(2, baudrate=2500000, polarity=0, phase=0)
spi.init()
```

```
rdr = MFRC522(spi=spi, gpioRst=4, gpioCs=5)
print("Place card")
sleep(2)
led = Pin(2,Pin.OUT)
(stat, tag_type) = rdr.request(rdr.REQIDL)
if stat == rdr.OK:
    (stat, raw_uid) = rdr.anticoll()
    if stat == rdr.OK:
        card_id = "uid: 0x%02x%02x%02x%02x" % (raw_uid[0], raw_uid[1], raw_uid[2], raw_uid[3])

        print(card_id)

        if card_id == "uid: 0x23c9a2fd":
            print('Door Open')
            led.value(1)
            sleep(0.5)
            led.value(0)
            motor=Servo(pin=13)
            for i in range(0,91,10):
                motor.move(i)
                sleep(0.5)
            sleep(3)
            for i in range(91,0,-10):
                motor.move(i)
                sleep(0.5)
            sleep(1)
            led.value(1)
            sleep(0.5)
            led.value(0)
        else:
            print('Door Locked')
        led.value(0)



#LCD Display
i2c = SoftI2C(scl=Pin(18), sda=Pin(21), freq=1000000)
lcd = I2cLcd(i2c, I2C_ADDR, totalRows, totalColumns)
lcd.clear()
try:
    weather_data = get_weather_data()
    w_d,t = display_weather(weather_data)
    print(t)

    if w_d:
        x = display_suggestion(w_d)

    degree = bytearray([0x06,0x09,0x09,0x06,0x00,0x00,0x00,0x00])
    lcd.custom_char(0, degree)
    lcd.putstr("Temp   : "+str(t)+chr(0)+"C")
    lcd.putstr("Climate: "+str(w_d)[0].upper()+str(w_d)[1::])
    sleep(3)
    lcd.clear()
    lcd.putstr(str(x))
except:
    pass

print("\n")
```

❖ **IR Sensor code:**

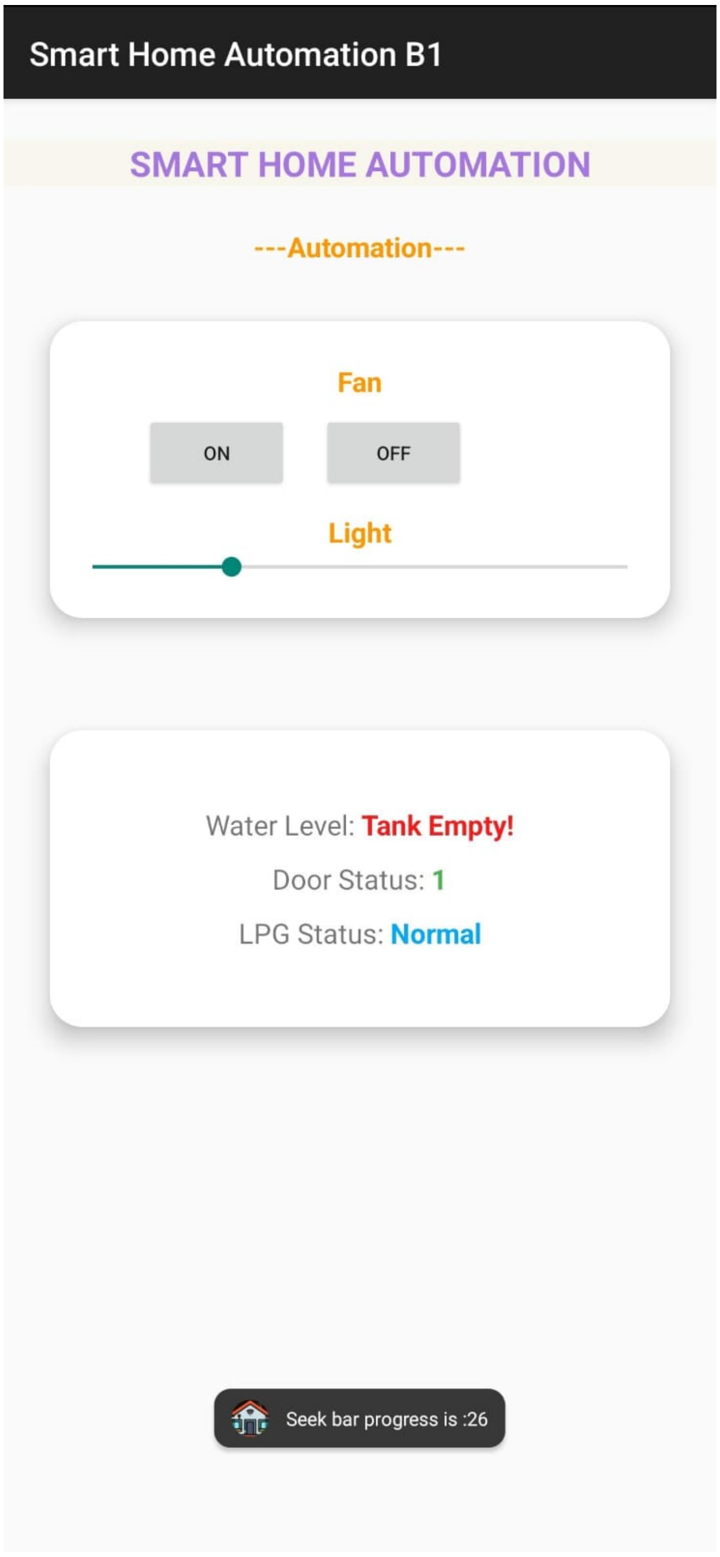
```
from machine import Pin
from time import sleep

ir1 = Pin(23, Pin.IN)
ir2 = Pin(22, Pin.IN)
```

```
led = Pin(15, Pin.OUT)

flag1 = False
flag2 = False
c=0
while True:
    d1 = ir1.value()
    d2 = ir2.value()
    print(d1,d2)
    if d1 == 0 and d2 == 1 and flag2 == False:
        flag1 = True
    elif flag1 == True:
        if d1 == 1 and d2 == 0:
            c+=1
            flag1 = False

    elif d1 == 1 and d2 == 0 and flag1 == False:
        flag2 = True
    elif flag2 == True:
        if d1 == 0 and d2 == 1:
            c-=1
            flag2 = False
    print(flag1, flag2)
    if c > 0:
        led.value(1)
    else:
        led.value(0)

    print("Number of Persons:",c)
    sleep(0.5)
```

- ❖ **Network Connection Code:**

```
import network



def connection():
    import network
    ssid = 'LIMITED2'
    password = '123.ILUV'

    #wlan.active(False)
    wlan = network.WLAN(network.STA_IF)

    wlan.active(True)

    if not wlan.isconnected():
        print('connecting to network...')
        wlan.connect(ssid, password)
        while not wlan.isconnected():
            print('Connecting...')
    print('network config:', wlan.ifconfig())

connection()
```

APP INTERFACE:



Smart Home Automation B1

SMART HOME AUTOMATION

---Automation---

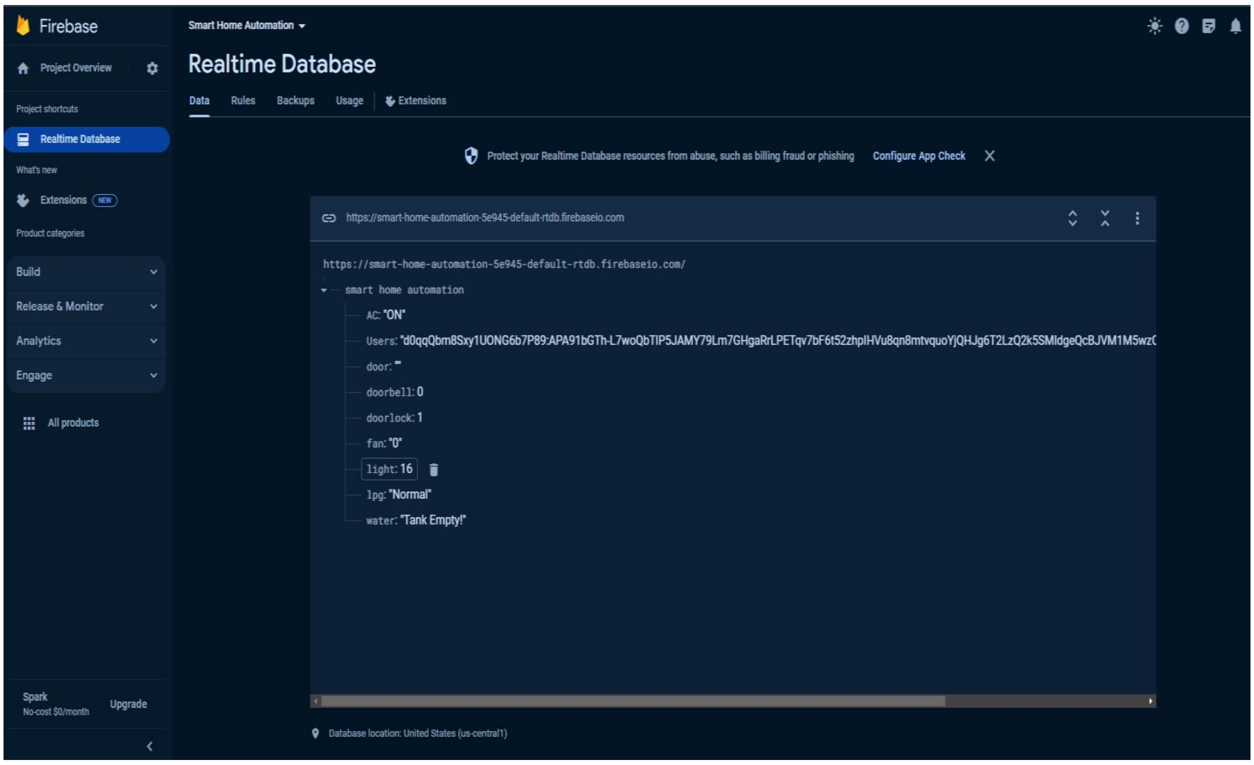
Water Level: Tank Empty!

Door Status: 1

LPG Status: Normal

Seek bar progress is :26

REALTIME DATABASE:



Realtime Database

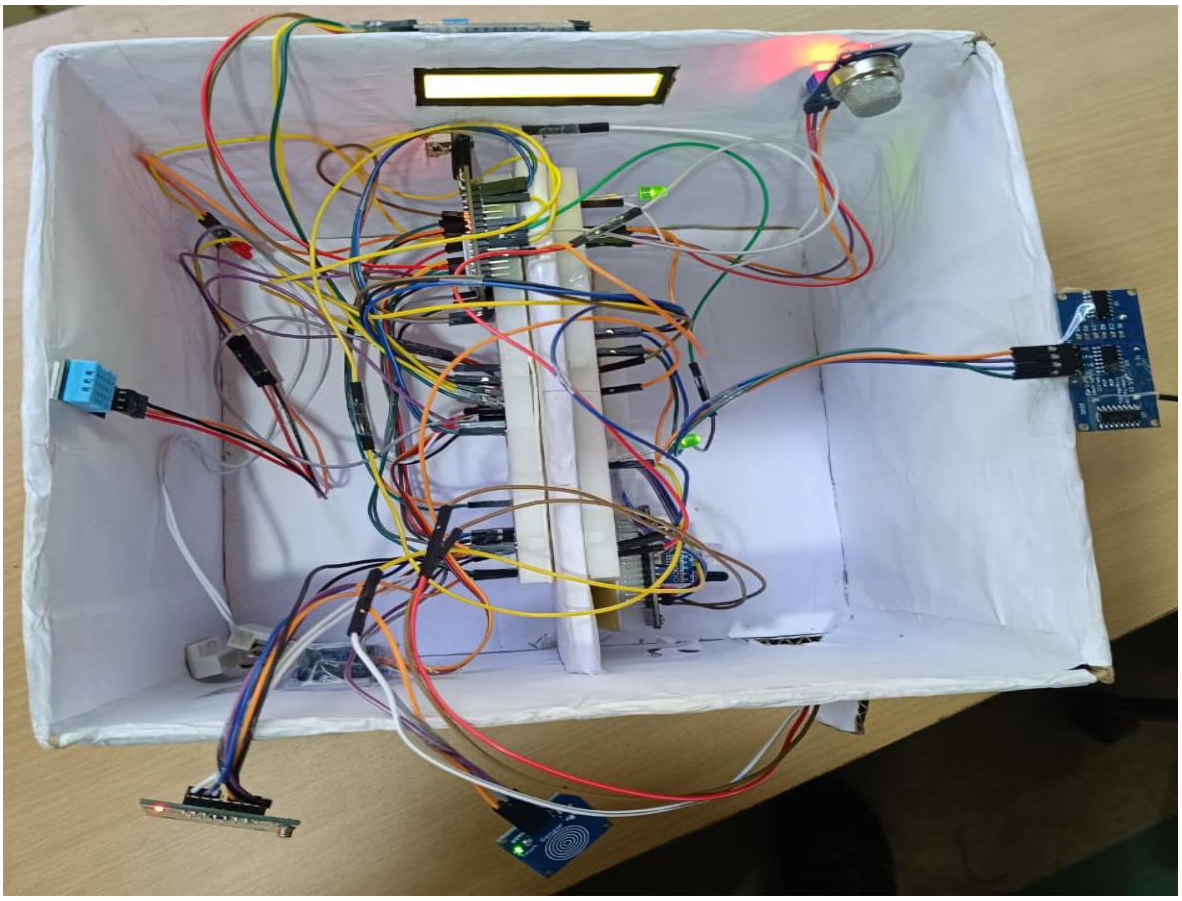
Protect your Realtime Database resources from abuse, such as billing fraud or phishing Configure App Check

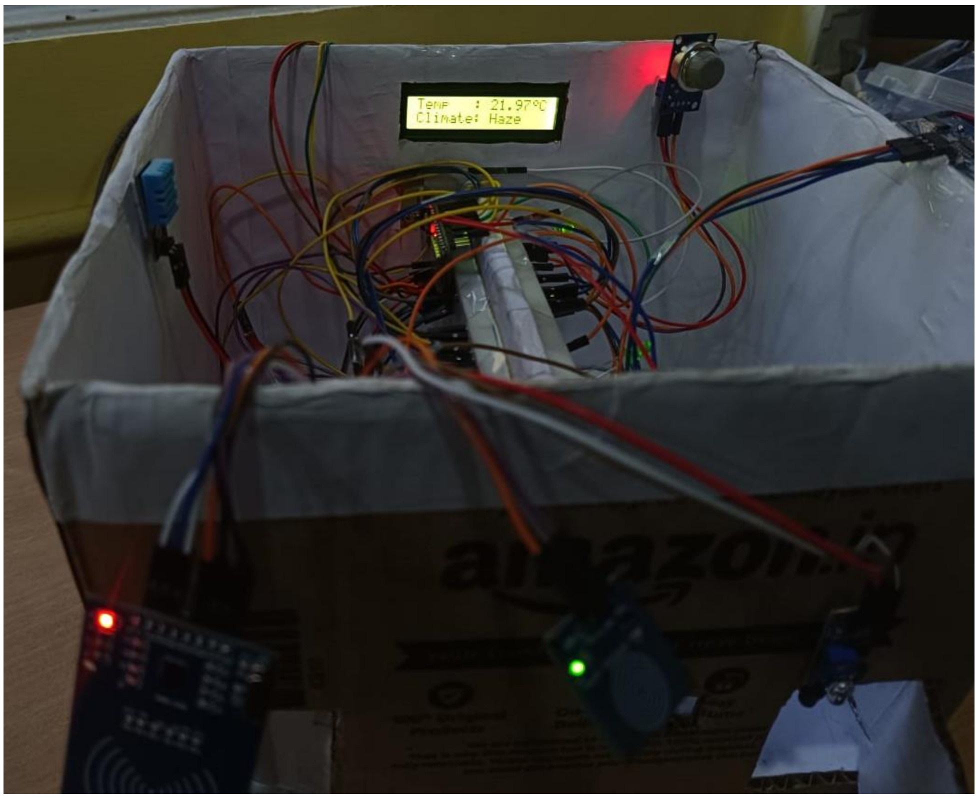
https://smart-home-automation-5e945-default-rtdb.firebaseio.com

```
https://smart-home-automation-5e945-default-rtdb.firebaseio.com/
▾ smart home automation
    AC: "ON"
    Users: "d0qqQbm8Sxy1UONG6b7P89:APA91bGTh-L7woQbTIP5JAMY79Lm7GHgaRrLPETqv7bF6t52zhplHVu8qn8mtvquoYjQHJg6T2LzQ2k5SMldgeQcBJVM1M5wz
    door: ""
    doorbell: 0
    doorlock: 1
    fan: "0"
    light: 16
    lpg: "Normal"
    water: "Tank Empty!"
```

Database location: United States (us-central1)

CIRCUIT DIAGRAM:




Temp : 21.97°C
Climate: Haze
amazon.in

CONCLUSION

In conclusion, Smart Home Automation epitomizes the next frontier of intelligent living, seamlessly integrating cutting-edge technologies to elevate home automation and security. Through the power of the ESP32 platform, this system not only optimizes energy usage, enhances comfort, and ensures safety but also embodies a user-centric ethos. By harmonizing occupancy-based controls, water management, door entry security, climate regulation, and real-time weather insights, Smart Home Automation pioneers a comprehensive and dynamic home environment. Its proactive LPG gas leakage detection and swift alerts further underscore its commitment to user safety. As technology continues to advance, Smart Home Automation stands as a beacon of innovation, offering a tangible and transformative solution that redefines the modern living experience.

REFERENCES

- Microcontroller and IoT Platform:
  - ESP32 Technical Reference Manual
  - Espressif Systems. (https://www.espressif.com/en/products/socs/esp32/resources)

- Occupancy Sensors and Infrared (IR) Technology:
  - Texas Instruments. "Understanding PIR Sensor Technology." (https://www.ti.com/lit/an/snoa926a/snoa926a.pdf)

- RFID Technology:
  - RFID Journal. (https://www.rfidjournal.com/)

- Water Level Monitoring:
  - Anbumani, B., and Manogaran, G. "A survey of big data architectures and machine learning algorithms in healthcare sector." Journal of King Saud University-Computer and Information Sciences.

- Door Entry Security and Cameras:
  - Axis Communications. "Introduction to Network Cameras." (https://www.axis.com/en-us/learning/web-articles/introduction-to-network-cameras)
  - Foscam. "IP Camera User Manual." (https://www.foscam.com/downloads/User%20Manual)

- Temperature Control and Sensors:
  - Texas Instruments. "Temperature Sensor Applications Guide." (https://www.ti.com/lit/ug/tidubr1/tidubr1.pdf)
  - Adafruit. "DHT11 Tutorial." (https://learn.adafruit.com/dht)

- Weather APIs:
  - OpenWeatherMap API. (https://openweathermap.org/api)

- Gas Leakage Detection:
  - MQ Gas Sensor Series Datasheets. (Example: MQ-7 Datasheet - https://www.sparkfun.com/datasheets/Sensors/Biometric/MQ-7.pdf)
  - Elster. "Gas Meter Technical Reference Manual." (https://www.elster.com/en)