

Mini Project Report

“HOME AUTOMATION WITH AR & IoT”



Submitted by---

Rajesh Haldar

Bijayeeta Biswas

Aritra Roy

Rajdeep Bhattacharjee

Amit Mandal

Suvendu Mondal

Department of Electronics & Communication Engineering

3rd Year 6th semester

Session: 2022-2023

Under supervision of:

Prof. Nilanjan Mukherjee



GLOBAL INSTITUTE OF MANAGEMENT AND TECHNOLOGY

(Affiliated to West Bengal University of Technology)

Krishnanagar – 741102, Nadia, WB

:- CONTENTS :-

1. Introduction.....	1
1.1. Overview.....	1
1.2. Background And Motivation.....	1
1.3. Objective.....	2
1.4. Methodology.....	2
2. Tool Description.....	3
2.1. Hardware Description.....	4
2.1.1. ESP32-WROOM-32.....	4
2.1.2. BC547 Transistor.....	9
2.1.3. PC817 Optocoupler.....	10
2.1.4. JQC-3FC (T73) DC06V Relay.....	12
2.1.5. 1N4007 Diode.....	13
2.1.6. 5MM LED.....	14
2.1.7. 5Volt 2Amp Power Supply.....	15
2.1.8. Momentary switch or tactile switch.....	16
2.1.9. Resistor.....	17
2.1.9.1. 330 Ω resistor.....	17
2.1.9.2. 1k Ω Resistor.....	17
2.2. Software Description.....	19
2.3. Blynk Server & Blynk IoT App.....	20
2.3.1. Blynk Server.....	20
2.3.2. Blynk IoT App.....	21
2.4. Unity Hub.....	22
2.5. Vuforia engine.....	23
2.6. EasyEDA.....	24
3. Circuit and Operation.....	25
3.1. Switched-Mode Power Supply (SMPS).....	25
3.2. IoT Hardware Kit.....	26
3.3. Integration of Hardware with IoT Server.....	28
3.4. Integration of IoT and AR.....	29
3.5. Circuit Schematic.....	31
3.6. Model Picture.....	32
3.7. Program Code.....	33
4. Project Costing.....	38
5. Result.....	39
6. Conclusion.....	40
7. Future work & References.....	41

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to *Prof. Nilanjan Mukherjee (HOD)*, *Prof. Dipankar Saha* and *Prof. Saswati De* for their guidance and support throughout the project. Their valuable suggestions and feedback helped me in completing the project successfully. I would also like to thank my friends for their constant encouragement and support.

The project "**Home Automation with Augmented Reality (AR) and Internet of Things (IoT)**" would not have been possible without their support.

May 2022

Rajesh Haldar
Bijayeeta Biswas
Aritra Roy
Rajdeep Bhattacharjee
Amit Mandal
Suvendu Mondal

Overview

This report presents the results of developing a "**Home Automation with Augmented Reality (AR) and Internet of Things (IoT)**" system. This system is based on the ESP32-WROOM-32 microcontroller, which is an open-source board with a 32-bit chip is based on a Tensilica Xtensa LX6 dual core microprocessor from Espressif Systems. The ESP32-WROOM-32 is the core of the project. The system can be controlled wirelessly through the "**Blynk IoT**" application and the internet. Additionally, this project can be controlled by a custom-made Augmented Reality application called "**AIoT Home**" that was created using the "Unity Hub" software. This system can be installed indoors as a regular switch board, but it has the added features of AR and IoT.

Background and Motivation

Home automation is the process of using technology to control various aspects of a home, such as lighting, temperature, security, entertainment, and appliances. Home automation can provide convenience, comfort, energy efficiency, and safety for the residents. However, most home automation systems rely on conventional interfaces, such as switches, buttons, remotes, or mobile apps, which may not be intuitive or user-friendly. Augmented Reality (AR) is a technology that overlays digital information onto the real world, creating an immersive and interactive experience for the user. AR can enhance the user's perception and understanding of the environment and provide natural and intuitive ways of interacting with it. Internet of Things (IoT) is a network of physical devices that can communicate and exchange data over the internet. IoT can enable remote monitoring and control of various devices and sensors in a home automation system. Combining AR and IoT can create a novel and innovative way of controlling a home automation system, where the user can see and manipulate the digital representations of the physical devices in their surroundings. This is what we develop "**Home Automation with Augmented Reality (AR) and Internet of Things (IoT)**" system, which is based on the ESP32-WROOM-32 microcontroller board. The system can be controlled wirelessly through the "Blynk IoT" application and the internet, as well as through a custom-made AR application that was created using the "Unity Hub" software.

Objective

The objective of this project is to design and implement a home automation system that can be controlled by both conventional and novel interfaces. The system will use the ESP32-WROOM-32 microcontroller board as the main component, which will communicate with various devices and sensors in the home. The system will also use the “Blynk IoT” application and the internet to enable wireless control of the devices. Moreover, the system will use a custom-made AR application that will overlay digital information and controls onto the real world, allowing the user to interact with the devices in a natural and intuitive way. The system will be evaluated based on its functionality, usability, and user satisfaction.

Methodology

To implement the above goals the following methodology needs to be followed:

1. Specifying the Application and various components of the project.
2. Specifying the bindings between the tasks and the resources either manually or by the design tools.
3. Specifying the port interconnections between the components.
4. Setup Blynk IoT server.
5. Build APK using Unity Hub software.

A physical project is a combination of both hardware and software. Some problems can be simplified by software by which we can operate the hardware portion efficiently. This project is also not an exception, rather we can say it is mostly a software-based project. Let's be familiar with them—

Hardware

1. ESP32-WROOM-32
2. Transistor— BC547
3. Optocoupler— PC817
4. Relay— JQC-3FC(T73)DC06V
5. Diode— 1N4007
6. LED— 5MM
7. Power Supply— 5Volt 2Amp
8. Momentary switch or tactile switch
9. Resistor— 330 Ω , 1K Ω
10. Others hardware
 - i. AC lamp holder
 - ii. AC Socket
 - iii. PVC Box
 - iv. AC Switch
 - v. Connecting Wires

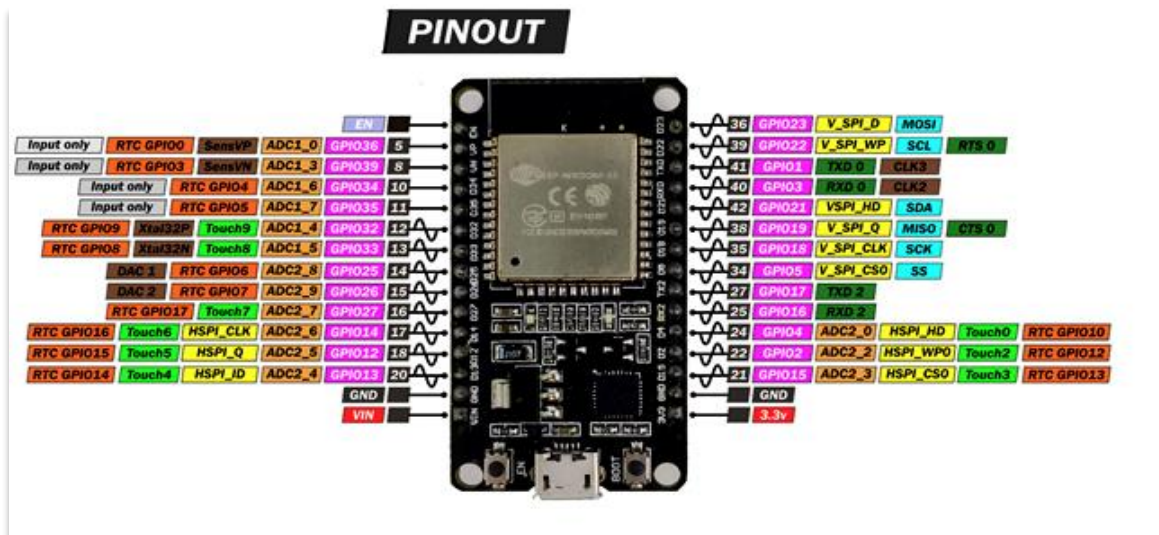
2. Software

1. Arduino IDE
2. Blynk IoT App
3. Blynk Server
4. Unity Hub
5. Vuforia engine
6. EasyEDA

1. Hardware Description

1.1. ESP32-WROOM-32

The ESP32 is a low-cost, low-power SoC (System on a Chip) and module series developed by Espressif Systems. It is the successor of the popular ESP8266, which had built-in Wi-Fi. The ESP32 also has Wi-Fi, as well as Bluetooth and Bluetooth Low Energy. It can be described as an enhanced version of the ESP8266. The chip also has very low power consumption, thanks to its power saving features such as clock synchronization and multiple operation modes. The quiescent current of the ESP32 chip is less than 5 μ A, which makes it suitable for battery-powered projects or IoT applications.



- **Specifications**

- ❖ **Processor:**

The ESP32-WROOM-32 module has an ESP32-D0WDQ6 chip at its core, which is a dual-core microprocessor with an operating frequency of up to 240 MHz.

The chip has a Tensilica Xtensa LX6 microprocessor architecture, which is a 32-bit RISC processor with a high-performance instruction set.

- ❖ **Wireless Connectivity:**

Compatible with 802.11 b / g / n in the 2.4GHz band, reaching speeds of up to 150 Mbits/s. It also includes Bluetooth communication compatible with Bluetooth v4.2 and Bluetooth Low Energy (BLE).

❖ **Memory:**

- i. Program memory: to store the sketch.
- ii. SRAM (520 KiB) memory: To store the variables that are used in the code.
- iii. EEPROM memory: to store variables that do not lose their value even when the device is turned off.
- iv. ROM memory (448 KiB).
- v. RTC SRAM (16 KiB): this memory is used by the co-processor when the device operates in deep sleep mode.
- vi. Efuse (1 Kilobit): 256 bits of this memory are used by the system itself and the remaining 768 bits are reserved for other applications.
- vii. Flash embedded (Embedded flash): This memory is where our application code is stored.

❖ **Encryption hardware accelerators:**

One of the most important factors in any system is security. That is why the ESP32 has algorithm accelerators aimed at encryption:

- **AES** (Advanced Encryption Standard) — FIPS PUB 197
- **SHA** (Secure Hash Algorithm)—FIPS PUB 180-4
- **RSA** (Rivest-Shamir-Adleman)

❖ **Peripheral Features:**

The ESP32 has a total of 34 digital pins. Most of these pins support the use of internal pull-up, pull-down, and high impedance status as well. This makes them ideal for connecting buttons and matrix keyboards, as well as for applying LED control techniques such as the well-known Charlieplexing.

ESP32 WROOM module has 25 GPIO pins out of which there are only input pins, pins with input pull up and pins without internal pullup.

Maximum current drawn per a single GPIO is 40mA according to the “Recommended Operating Conditions” section in the ESP32 datasheet.

❖ **ADC (Analog to digital converters):**

Some of the pins listed in the pinout diagram can also be used to interact with analog sensors, same as analog pins of an Arduino board.

For this, the ESP32 has a 12-bit(0-4096 resolution which means when voltage observed is 0 the value is 0 and when max voltage like 3.3v is observed the value goes to 4096),

18-channel analog to digital converter, which means you can take readings from up to 18 analog sensors.

This allows to develop very compact connected applications, even when using multiple analog sensors.

Analog input pins:

ADC1_CH0 (GPIO 36)	ADC1_CH6 (GPIO 34)	ADC2_CH4 (GPIO 13)
ADC1_CH1 (GPIO 37)	ADC1_CH7 (GPIO 35)	ADC2_CH5 (GPIO 12)
ADC1_CH2 (GPIO 38)	ADC2_CH0 (GPIO 4)	ADC2_CH6 (GPIO 14)
ADC1_CH3 (GPIO 39)	ADC2_CH1 (GPIO 0)	ADC2_CH7 (GPIO 27)
ADC1_CH4 (GPIO 32)	ADC2_CH2 (GPIO 2)	ADC2_CH8 (GPIO 25)
ADC1_CH5 (GPIO 33)	ADC2_CH3 (GPIO 15)	ADC2_CH9 (GPIO 26)

❖ **DAC (Digital to Analog Converters):**

PWM signals are used on most Arduino boards to generate analog voltages. The ESP32 has two 8 bit digital to analog converters. This allows two pure analog voltage signals to be generated.

DAC Pins:

DAC1 (GPIO25)

DAC2 (GPIO26)

❖ **Capacitive Touch GPIOs:**

In case want to develop applications with no mechanical buttons, we can use the touch sensitive pins on ESP32s to achieve it. Capacitive Touch pins:

T0 (GPIO 4)	T5 (GPIO 12)
T1 (GPIO 0)	T6 (GPIO 14)
T2 (GPIO 2)	T7 (GPIO 27)
T3 (GPIO 15)	T8 (GPIO 33)
T4 (GPIO 13)	T9 (GPIO 32)

❖ **RTC:**

RTC GPIO support in the core section. The GPIOs which are routed to the RTC low-

power management subsystem can be used when the ESP32 is in deep sleep. These RTC GPIOs can be used to wake up the ESP32 from deep sleep when the Ultra-Low Power (ULP) co-processor is running.

The following GPIOs can be used as an external wake up source.

RTC_GPIO0 (GPIO36)	RTC_GPIO6 (GPIO25)	RTC_GPIO10 (GPIO4)	RTC_GPIO14 (GPIO13)
RTC_GPIO3 (GPIO39)	RTC_GPIO7 (GPIO26)	RTC_GPIO11 (GPIO0)	RTC_GPIO15 (GPIO12)
RTC_GPIO4 (GPIO34)	RTC_GPIO8 (GPIO33)	RTC_GPIO12 (GPIO2)	RTC_GPIO16 (GPIO14)
RTC_GPIO5 (GPIO35)	RTC_GPIO9 (GPIO32)	RTC_GPIO13 (GPIO15)	RTC_GPIO17 (GPIO27)

❖ **SD / SDIO / MMC driver:**

This peripheral allows the ESP32 to interact with SD and MMC cards directly. In fact, by combining this controller with the analog digital converter it is possible to improve our little audio player.

❖ **UART:**

Many microcontrollers have UART modules, which on Arduino are known as Serial ports. These allow asynchronous communications between two devices using only two pins.

The ESP32 has three UART ports:

- UART0
- UART1
- UART2

All of these are compatible with RS-232, RS-485 and IrDA protocols.

❖ **I2C:**

The ESP32 have two interfaces I2C or TWI that support the operating modes master and slave. Its features include:

- Standard mode (100 Kbit/s)
- Fast mode (400 Kbit/s)
- 7 and 10 bit addressing

I2C Pins— GPIO 21 (SDA), GPIO 22 (SCL)

❖ SPI

The ESP32 also has SPI communication. It has three fully functional buses:

- i. Four transfer modes: This means that it is compatible with all or almost all SPI and QSPI devices available on the market.
- ii. All SPI ports are capable of high speeds (theoretically up to 80 MHz).
- iii. 64-byte buffer for transmission and reception.

❖ Infrared remote controller:

The ESP32 also allows the transmission and reception of signals using various infrared protocols (the same as those used by the television remote).

Therefore, you can also use your ESP32 to create your own remote control that allows you to interact with your TV or your stereo.

❖ PWM

Like the ESP8266, the ESP32 also supports the use of analog outputs using PWM. The big difference is in ESP32 it is possible to use up to 16 pins as PWM outputs where ESP8266 only supports 8 and Arduino UNO board that only supports 6.

1.2. BC547 Transistor

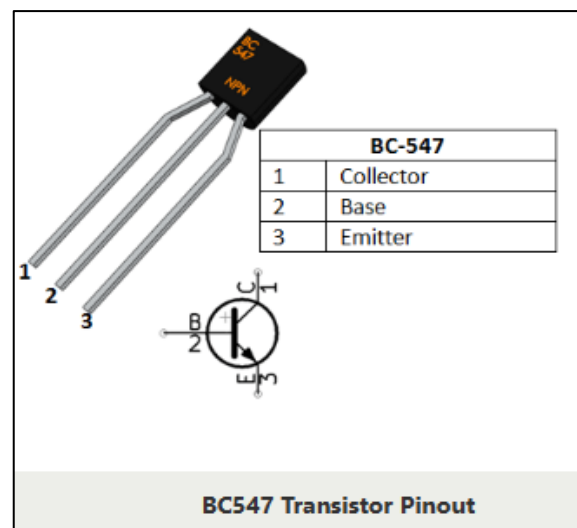
BC547 is a general purpose NPN transistor that can be used for current amplification and switching. It has three terminals: Emitter, Base, and Collector. The current flowing from base to emitter controls the current flowing through the collector. BC547 has a gain value of 110 to 800. The maximum collector current is 100mA when the base current is limited to 5mA. The typical voltage across the collector-emitter or base-emitter is 200 and 900 mV respectively.

• Specifications

- Transistor type: Bi-Polar NPN Transistor
- DC Current Gain (hFE): 110-800
- Collector current (IC): 100mA
- Collector Base Voltage (VCB): 50 V
- Collector-Emitter voltage (VCE) : 45V
- Emitter Base Voltage (VEB): 6V
- Maximum Power dissipation: 500mW
- Junction temperature: 150°C
- Current gain-bandwidth product: 300MHz
- Noise figure: 10 dB
- Available in TO-92 Package

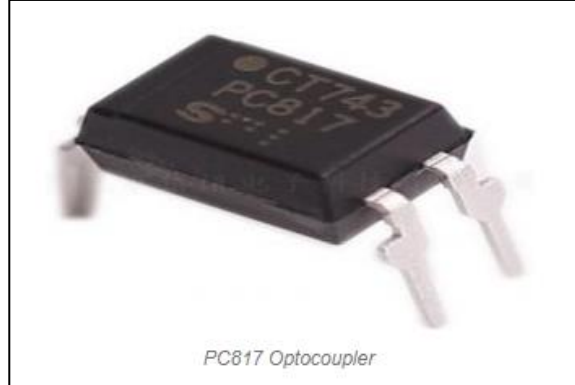
• Applications

- Current amplification
- Switch
- Pulse width modulation
- Darlington pair
- Driver module



1.3. PC817 Optocoupler

An optocoupler is also called an optoisolator, photo-coupler & optical isolator is one kind of semiconductor device that allows the electrical signal to transmit between two isolated circuits through light. This component includes two parts like an LED and a photosensitive device.



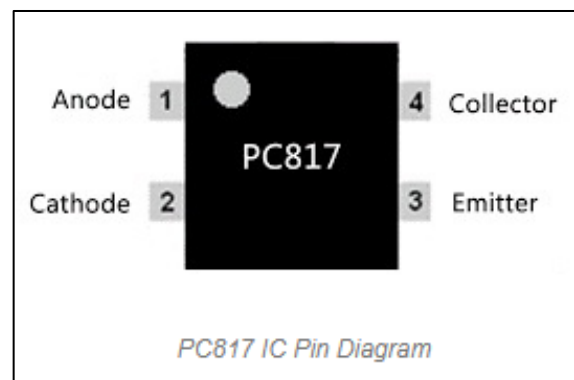
PC817 is a general-purpose photocoupler that contains an infrared diode and a phototransistor. It is used for electrical isolation between two circuits. It is packaged in a 4pin DIP, available in wide-lead spacing option and SMT gullwing lead-form option. Input-output isolation voltage (RMS) is 5.0kV. The forward voltage of the input diode is 1.25V. The maximum collector current is 50mA. The maximum collector-emitter voltage is 80V.

• Pin Configuration:

Pin1 (Anode): In the optocoupler IC, this is an Anode pin of infrared LED (Tx). This pin provides a logical input signal toward the internal IR.

Pin2 (Cathode): In this IC, this is the cathode pin of the infrared LED (Tx). It will provide the infrared to create the common GND through the circuit & power supply

Pin3 (Collector): This is an o/p pin of the IR Tx in the optocoupler and it provides the logical o/p through getting the infrared signal.



Pin4 (Emitter): This is a GND pin for IR Rx in the IC & it is used to build the common GND through the circuit & power supply.

• Specifications

The specifications of the PC817 Optocoupler include the following.

- Forward Voltage of Input Diode is 1.25V
- At the collector terminal, the maximum current ratio is 50mA

-
- The maximum Voltage of the Collector-Emitter is 80V (max)
 - At the Collector & Emitter Terminals, the ratio of maximum voltage is 80V
 - Maximum Collector Current is 50mA
 - Rise Time is 18us
 - Fall Time is 18us
 - The cut-off frequency is 80 kHz
 - The max operating temperature ranges from -30 to 100 degrees.
 - Power Dissipation is 200mW.
 - The internal resistance is 100 ohms.
 - The internal storage temperature of this IC ranges from -55 -125 degrees.
 - While doing soldering, the optocoupler's temperature range is 260 degrees. Once the temperature increases then the IC will be damaged.

• Features

The features of the PC817 IC Optocoupler include the following.

- This IC includes 4-pins and is available in two packages like SMT and DIP
- This IC includes internal protection for both input & output from electrical isolation and it protects up to 5KV.
- This IC is used with an additional resistor through high voltage devices to function through fewer voltage devices.
- This IC can function through any type of device including internal interfaces such as Microcontrollers, TTL devices & also with HIGH DC voltage through some inside resistors.
- PC817 Optocoupler comes with inside safety from reverse current, because of the flow of current in one way; this IC defends the infrared from any current.

• Applications

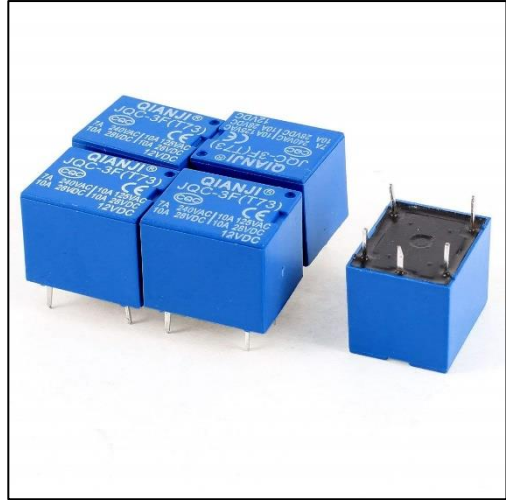
The applications of PC817 Optocoupler include the following.

- In Isolation Circuits
- Switching Circuits with Microcontroller I/O
- Isolation of Signal
- Circuits of Noise Coupling
- AC/DC Power control
- It is Reliable to utilize because of its functionality.
- For Transmission of Signal
- Noise Coupling Circuit.

1.4. JQC-3FC (T73) DC06V Relay

JQC-3FC (T73) DC06V Relay is a power relay that can switch up to 10A of current at 250V AC or 30V DC. It has a SPDT (single pole double throw) configuration, which means it has one common terminal, one normally open terminal and one normally closed terminal²³⁴. It is a sugar cube relay, which means it has a small size and a square shape. It is designed for PCB mounting and has 5 pins. The coil voltage is 6V DC and the coil resistance is 70 ohms.

This product is known as 10A Relay, 10A SPDT Relay, 6V 10A Relay, 6V 10A SPDT PCB Mount Relay, 6V 10A SPDT Relay, 6V Relay, 6V SPDT Relay, Component, PCB Mount Relay, Relay, SPDT, SPDT PCB Mount Relay.



- **Specifications**

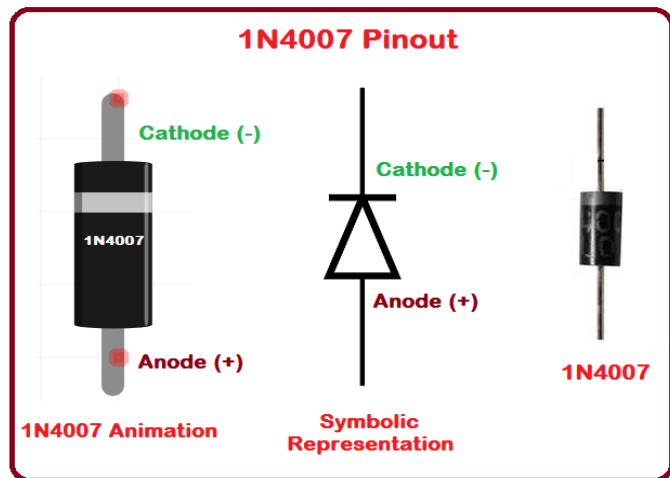
- Manufacturer: HL
- Part No: JQC-3FC(T73)DC06V
- Relay Type: SPDT
- Coil Voltage: 3~24VDC
- Coil Power: 0.36W,0.45W
- Current: 10A
- Contact Resistance: $\leq 100\text{M}\Omega(1\text{A } 6\text{VDC})$
- Rated Load: 10A 250VAC/30VDC
- Pin counts: 5
- Operate time: 10ms max
- Release time: 5ms max
- Ambient temperature: -40°C to +85°C

- **Application**

- Domestic control and switching
- Car control switching box
- HVAC
- Home Automation and Security system
- Motor Control System etc.

1.5. 1N4007 Diode

The 1N4007 diode is a widely used rectifier diode with specific characteristics and specifications. The 1N4007 is a standard rectifier diode designed for general-purpose rectification in electronic circuits. It belongs to the 1N400x series of diodes, which are commonly available and cost-effective. The diode is typically encapsulated in a DO-41 package, which has a cylindrical shape and two axial leads. It follows the standard color coding convention



with a black band marking the cathode terminal. The 1N4007 diode has a high voltage rating, making it suitable for rectifying higher voltage AC signals into DC in various applications, such as power supplies, battery chargers, and motor control circuits. Its low forward voltage drop and fast recovery time make it efficient for these purposes.

- **Specifications**

- Maximum Recurrent Peak Reverse Voltage 1000V
- Maximum RMS Voltage 700V
- Maximum DC Blocking Voltage 1000V
- Average Forward Current: 1.0A
- Peak Forward Surge Current: 30A
- Maximum Instantaneous Forward Voltage: 1.0V
- Maximum DC Reverse Current At Rated DC Blocking Voltage: 5.0 μ A @ 25°C
- Typical Junction Capacitance: 15pF
- Typical Reverse Recovery Time: 2.0 μ s
- Mounting Type: Through Hole
- Operating Temperature: -55°C ~ 150°C

- **Application**

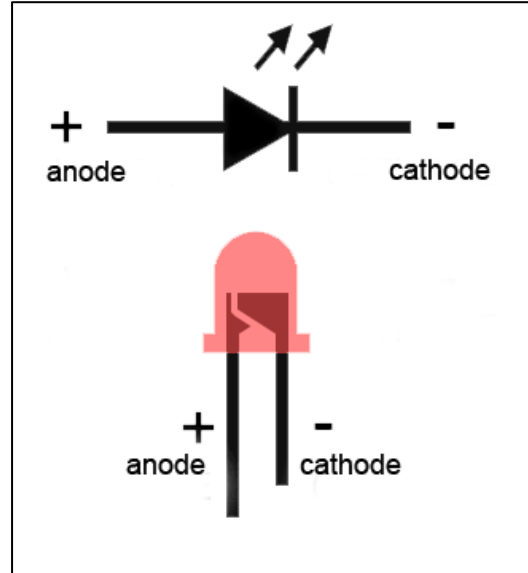
- Power Supplies
- Rectification
- Battery Chargers
- Motor Control
- Inverters

1.6. 5MM LED

A 5mm LED (Light-Emitting Diode) is a popular type of LED that emits light when current flows through it in the forward direction. A 5mm LED is a small electronic component consisting of a semiconductor material that emits light when activated. It typically has a cylindrical shape with a diameter of 5mm and is available in various colors, including red, green, blue, yellow, and white.

- **Specifications**

- Superior weather resistance
- 5mm Round Standard Directivity
- UV Resistant Epoxy
- Forward Current (IF): 30mA
- *Forward Voltage (VF): 1.8V to 2.4V
- Reverse Voltage: 5V
- Operating Temperature: -30°C to +85°C
- Storage Temperature: -40°C to +100°C
- Luminous Intensity: 20mcd



*Forward voltage of the LED according to its color—

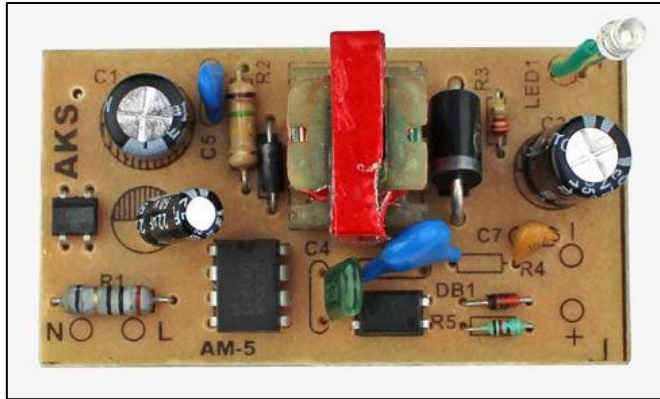
LED Color	Forward Voltage
Red	1.63 ~ 2.03V
Yellow	2.10 ~ 2.18V
Orange	2.03 ~ 2.10V
Blue	2.48 ~ 3.7V
Green	1.9 ~ 4.0V
Violet	2.76 ~ 4.0V
UV	3.1 ~ 4.4V
White	3.2 ~ 3.6V

- **Application**

- Indicator lights
- Illumination
- DIY projects
- Circuit prototyping

1.7. 5Volt 2Amp Power Supply

A 5V 2A AC to DC Switch Mode Power Supply Module (SMPS) is a compact and efficient power supply module that converts AC (alternating current) input voltage from a wall outlet to a regulated and stable DC (direct current) output voltage of 5 volts. It is designed for general-purpose applications where a reliable and efficient power source is required. It is designed to convert AC (alternating current) voltage from a wall outlet into a stable and regulated DC voltage suitable for powering electronic devices. It typically consists of a power transformer, rectifier, voltage regulator, and other components housed in a compact enclosure.



- **Specifications**

- Input voltage: AC 100-240V, 50/60Hz (compatible with worldwide power standards)
- Output voltage: 5V DC (direct current)
- Output current: 2A (maximum)
- Output power: 10 watts (5V x 2A)
- Efficiency: Typically above 80%, indicating that it converts at least 80% of the input power into usable output power.
- Regulation: The module is designed to provide a regulated and stable output voltage of 5 volts, maintaining a relatively constant voltage under varying load conditions.
- Protection features: It may include built-in protection mechanisms such as overcurrent protection, overvoltage protection, and short circuit protection to safeguard against potential faults or damage.
- Connector type: The module typically has solder pads or connectors for easy integration into the target system or application.

- **Application**

- Charging smartphones, tablets, and other USB-powered devices
- Powering microcontrollers, single-board computers, and development boards
- Driving LED strips, lighting modules, and other low-power lighting applications
- Powering audio/video equipment such as speakers, amplifiers, and media players

1.8. Momentary switch or tactile switch

A DC small push button, also known as a momentary switch or tactile switch, is a type of switch that is activated by pressing it and returns to its original position when released. It is a compact switch with a button-shaped actuator that is designed to be easily pressed with a finger or thumb. It is commonly used in electronic circuits and devices where momentary control or signal input is required.



- **Specifications**

- Contact configuration: Usually single-pole, single-throw (SPST), meaning it has one set of contacts that are open (off) when the button is not pressed and closed (on) when the button is pressed.
- Contact rating: It typically ranges from a few volts to around 12V DC and a few mill amperes to a few amperes, depending on the specific switch.
- Switching mechanism: Tactile switches often use a mechanical mechanism with metal or conductive contacts that make or break electrical connections when the button is pressed.
- Operating force: The force required to press the button and activate the switch. It is typically measured in grams or Newton.
- Lifespan: The number of cycles the switch is rated for, indicating the number of times it can be pressed and released reliably. Typical ratings range from a few thousand to several hundred thousand cycles.

- **Application**

- Momentary control functions in electronic devices, such as power buttons, reset buttons, or mode selection buttons.
- Input switches in keyboards, remote controls, or game controllers.
- Signal inputs in circuits or systems, such as triggering an event or initiating a specific action.
- Circuit testing and prototyping, where the button is used to manually simulate or trigger specific conditions.

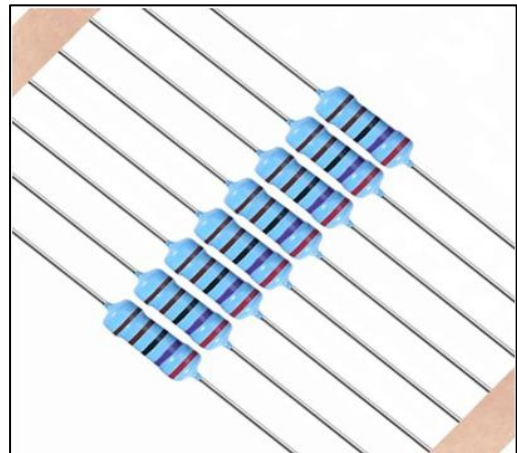
1.9. Resistor

1.9.1. 330 Ω resistor

A 330 Ω resistor is a passive electronic component with a resistance value of 330 ohms. It is typically cylindrical in shape and made of a resistive material with leads or terminals for easy connection to a circuit.

- **Specifications**

- Resistance value: 330 ohms (Ω)
- Tolerance: $\pm 5\%$ and $\pm 1\%$.
- Power rating: Power ratings range from 0.25 watts (1/4W) to 0.5 watts (1/2W) or higher.
- Temperature coefficient: It is typically given in parts per million per degree Celsius (ppm/ $^{\circ}\text{C}$).
- Construction: The resistor can be composed of various resistive materials, such as carbon composition, metal film, or thin film.



- **Application**

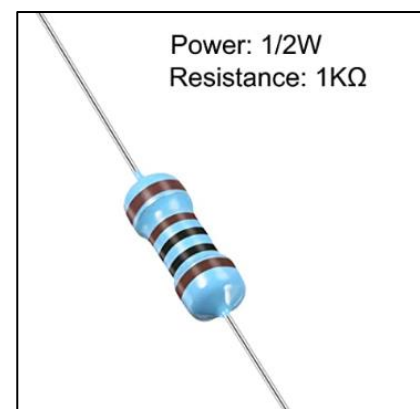
- Current limiting
- Voltage division
- Pull-up or pull-down resistors
- Biasing

1.9.2. 1k Ω Resistor

A 1k Ω resistor is a passive electronic component with a resistance value of 1 kilohm (1000 ohms). It is typically cylindrical in shape with leads or terminals for easy connection to a circuit.

- **Specifications**

- Resistance value: 1 kilohm (1k Ω),
- Tolerance: $\pm 5\%$ and $\pm 1\%$.



-
- Power rating: Range from 0.25 watts (1/4W) to 0.5 watts (1/2W) or higher.
 - Temperature coefficient: The temperature coefficient of resistance (TCR) indicates how the resistance value changes with temperature. It is typically given in parts per million per degree Celsius (ppm/°C).
 - Construction: The resistor can be composed of various resistive materials, such as carbon composition, metal film, or thin film.

- **Application**

- Current limiting
- Voltage division
- Pull-up or pull-down resistors
- Biasing

2. Software Description

2.1. Arduino IDE

Arduino IDE is a software platform for programming and developing projects with Arduino microcontrollers. It has various tools and features that help users of all skill levels to write, edit, and upload code to Arduino boards and others boards.

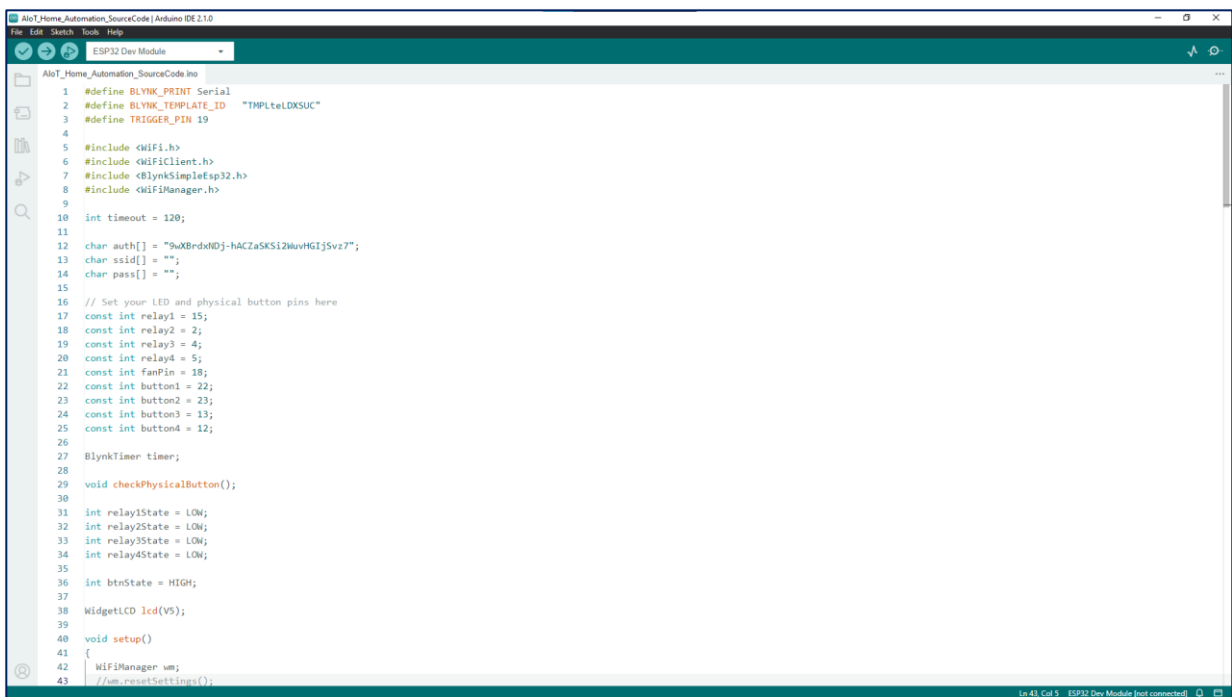
Arduino IDE uses a simplified programming language based on C/C++, called the Arduino programming language.

The IDE has a built-in code editor with syntax highlighting, auto-indentation, and code suggestions. It also has a compilation process that converts the code into binary files that can run on the microcontroller. It checks the code for errors and shows them to the user.

Arduino IDE lets users manage libraries easily. It also has a serial monitor tool for communicating with Arduino boards and viewing serial data.

The Arduino IDE is popular for its simplicity, versatility, and community support. It is used for many applications, such as prototyping, robotics, home automation, and IoT projects.

Here are some of the snaps of the same:



```
1 #define BLYNK_PRINT Serial
2 #define BLYNK_TEMPLATE_ID "TMPLTtL0KSUC"
3 #define TRIGGER_PIN 19
4
5 #include <WiFi.h>
6 #include <WiFiClient.h>
7 #include <BlynkSimpleEsp32.h>
8 #include <WiFiManager.h>
9
10 int timeout = 120;
11
12 char auth[] = "9uXBrdxIDj-hACZaSKSi2MuvHG1JSvz7";
13 char ssid[] = "";
14 char pass[] = "";
15
16 // Set your LED and physical button pins here
17 const int relay1 = 15;
18 const int relay2 = 2;
19 const int relay3 = 4;
20 const int relay4 = 5;
21 const int fanPin = 18;
22 const int button1 = 22;
23 const int button2 = 23;
24 const int button3 = 13;
25 const int button4 = 12;
26
27 BlynkTimer timer;
28
29 void checkPhysicalButton();
30
31 int relay1State = LOW;
32 int relay2State = LOW;
33 int relay3State = LOW;
34 int relay4State = LOW;
35
36 int btnState = HIGH;
37
38 WidgetLCD lcd(V5);
39
40 void setup()
41 {
42   WiFiManager wm;
43   //wm.resetSettings();
```

2.2. Blynk Server & Blynk IoT App

Blynk is an IoT (Internet of Things) platform that consists of the Blynk IoT App and the Blynk Server. Together, they provide a comprehensive solution for building and controlling IoT projects, allowing users to connect and control hardware devices remotely using a mobile application.

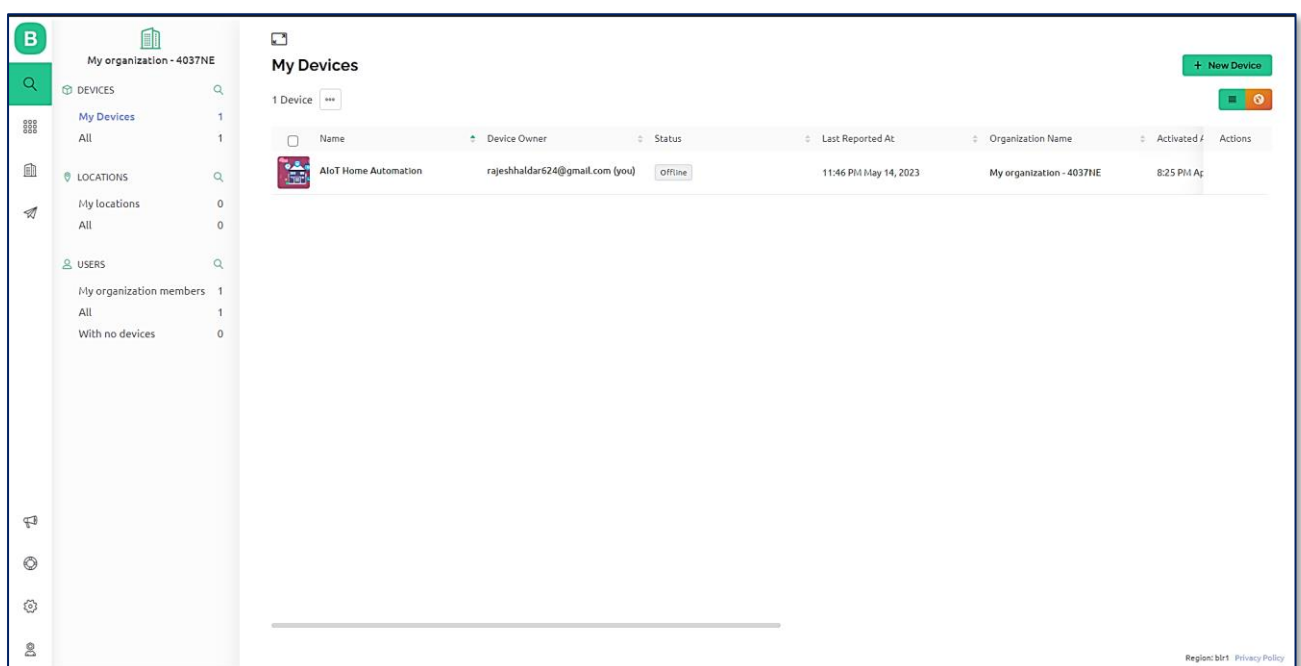
2.2.1. Blynk Server

Blynk Server is the backend of the Blynk platform. It connects the Blynk IoT App and the hardware devices. It handles the communication between the app and the hardware, enabling real-time data exchange and device control.

Blynk Server has an API that lets developers use Blynk in their IoT projects or connect with other services. It uses secure protocols, such as HTTP, WebSockets, and MQTT, to protect data privacy and integrity.

Blynk Server also supports cloud connectivity, letting users access and control their IoT devices remotely from anywhere with internet. It has features like user management, device sharing, and project collaboration, making it good for personal and commercial IoT applications.

Here are some of the snaps of the same:

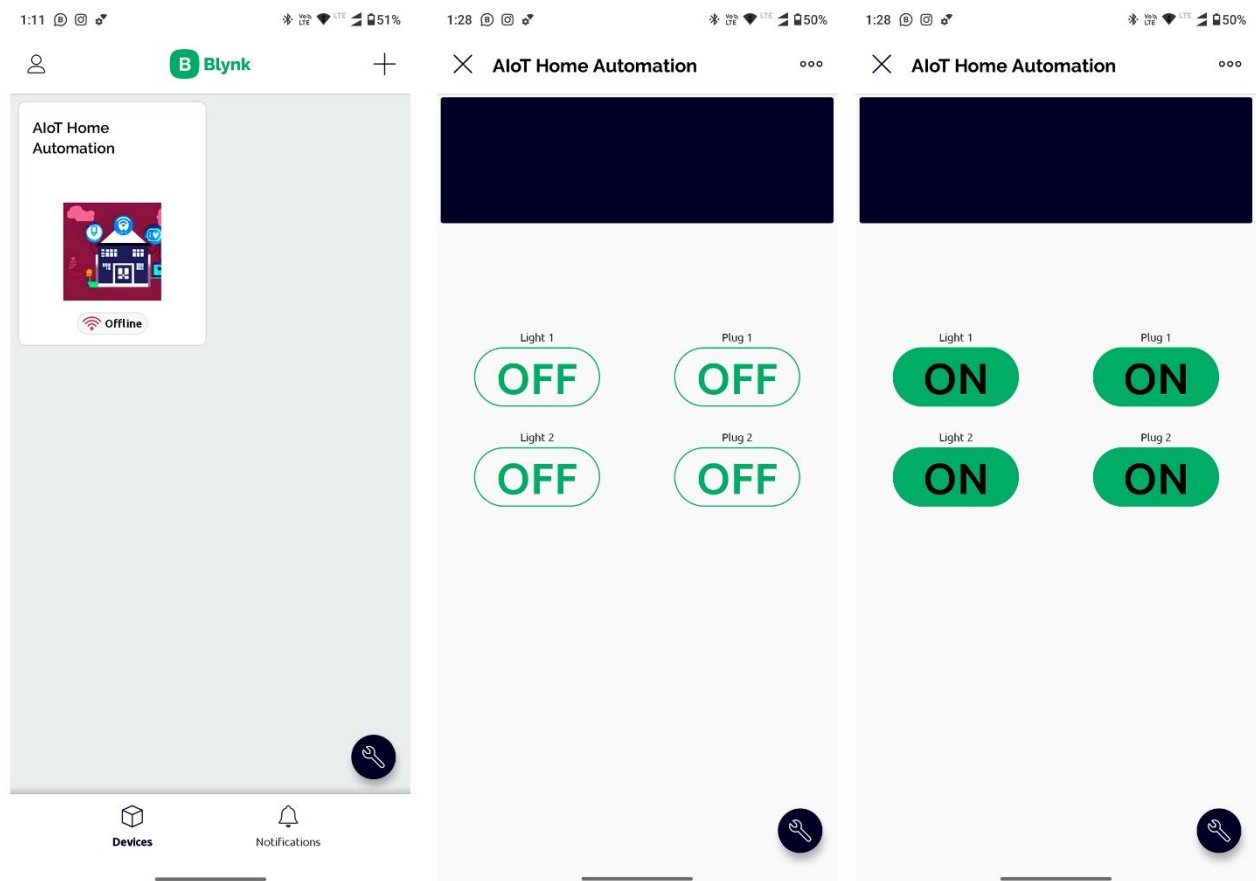


2.2.2. Blynk IoT App

Blynk IoT App is a mobile app for iOS and Android devices. It lets users monitor and control their IoT projects. Users can make custom GUIs by dragging and dropping widgets, such as buttons, sliders, graphs, and gauges. These widgets are controls and indicators for the hardware devices.

Blynk IoT App connects to IoT hardware through Wi-Fi, Ethernet, Bluetooth, or cellular networks. Users can remotely monitor sensor data, control actuators, and get real-time notifications from their IoT devices. The app also logs data and shows historical data over time.

Here are some of the snaps of the same:



Overall, Blynk IoT App and Blynk Server make a powerful and user-friendly platform for creating, controlling, and monitoring IoT projects. It makes the development process easy by offering a drag-and-drop interface, a large widget library, and smooth connectivity options. Blynk helps hobbyists, makers, or professionals create advanced IoT applications without much programming knowledge or infrastructure setup.

2.3. Unity Hub

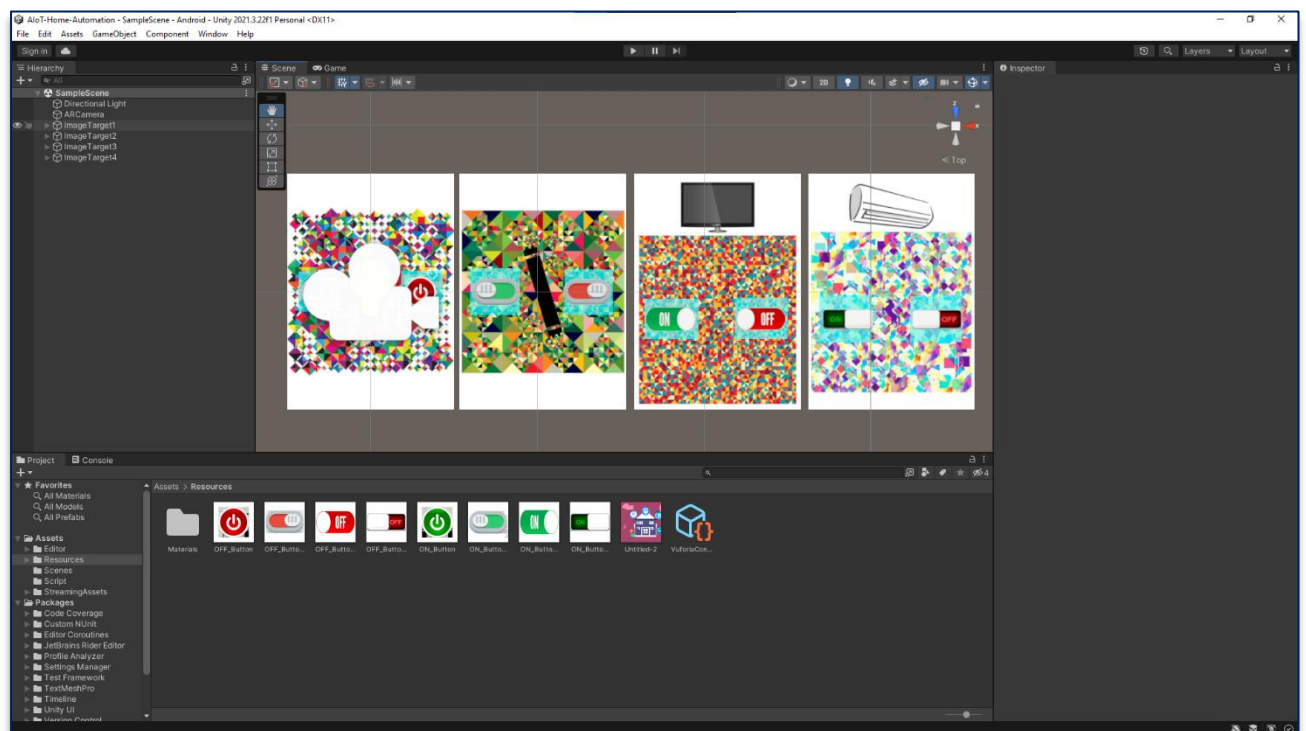
Unity Hub is a hub or control center for Unity development. It lets developers organize and access their Unity projects, install and manage different Unity Editor Versions, and handle licenses and add-ons.

Unity Hub supports AR app development. Unity is a game development engine that also has tools and features for building AR apps. With Unity Hub, developers can easily create and manage their AR projects. Developers can use Unity Hub to create and manage AR projects, install the Unity Editor with AR features, and use Unity's AR development tools and plugins.

Unity's AR Foundation, an official Unity package, gives a framework for building AR experiences that work on different AR platforms, such as ARKit (iOS) and ARCore (Android). Developers can use Unity Hub to install and manage the AR Foundation package and other AR packages.

In short, Unity Hub is a powerful tool that streamlines AR app development using Unity. It provides a platform for managing Unity projects, installations, and licenses, making it easier for developers to create, collaborate on, and deploy AR apps.

Here are some of the snaps of the same:



2.4. Vuforia engine

Vuforia Engine is a comprehensive AR development platform that provides developers with the tools and capabilities to create interactive and visually compelling AR applications. It offers a range of features, including computer vision technology, object recognition, tracking, and content management. Vuforia Engine allows developers to seamlessly integrate virtual content into the real world, enabling users to interact with digital elements in a natural and immersive manner. Vuforia Engine allows developers to create AR experiences that recognize and track specific objects or images. By training the engine to recognize target objects or images, developers can trigger AR content based on real-world objects, such as packaging, logos, or products. Vuforia Engine supports the recognition and tracking of 2D images as targets for AR content. This feature allows developers to overlay digital content onto printed images, paintings, advertisements, or other visual media. Vuforia Engine can also recognize and track 3D objects as targets for AR experiences. By using physical objects or CAD models as targets, developers can create AR applications that interact with and augment real-world objects.

The top screenshot shows the Vuforia Engine developer portal interface. The navigation bar includes links for Home, Pricing, Downloads, Library, Develop, and Support. The user is logged in as 'Hello rajeshaldar'. The 'License Manager' tab is active, showing details for 'AIoT-Home-Automation'. The license key is displayed, and the plan type is 'Basic'. The status is 'Active', created on 'Apr 12, 2023 23:12', and the license UUID is 'b6345e4e1d4d425480fa08656277e1f8'. The history shows 'License Created - Apr 12, 2023 23:12'.

The bottom screenshot shows the 'Target Manager' tab for 'AIoT-Home-Automation-Target-Image-Database'. It displays a table of targets with columns for Target Name, Type, Rating, Status, and Date Modified. There are four targets listed, all of type 'Image' and status 'Active', created on 'Apr 12, 2023 23:14'.

Target Name	Type	Rating	Status	Date Modified
Image4	Image	★★★★★	Active	Apr 12, 2023 23:15
Image3	Image	★★★★★	Active	Apr 12, 2023 23:14
Image2	Image	★★★★★	Active	Apr 12, 2023 23:14
Image1	Image	★★★★★	Active	Apr 12, 2023 23:14

2.5. EasyEDA

EasyEDA is an online software for electronic design. Users can design, simulate, and work together on circuit designs. It has many tools and features for schematics, PCB layouts, and simulations. It also has a large component library and a user-friendly interface.

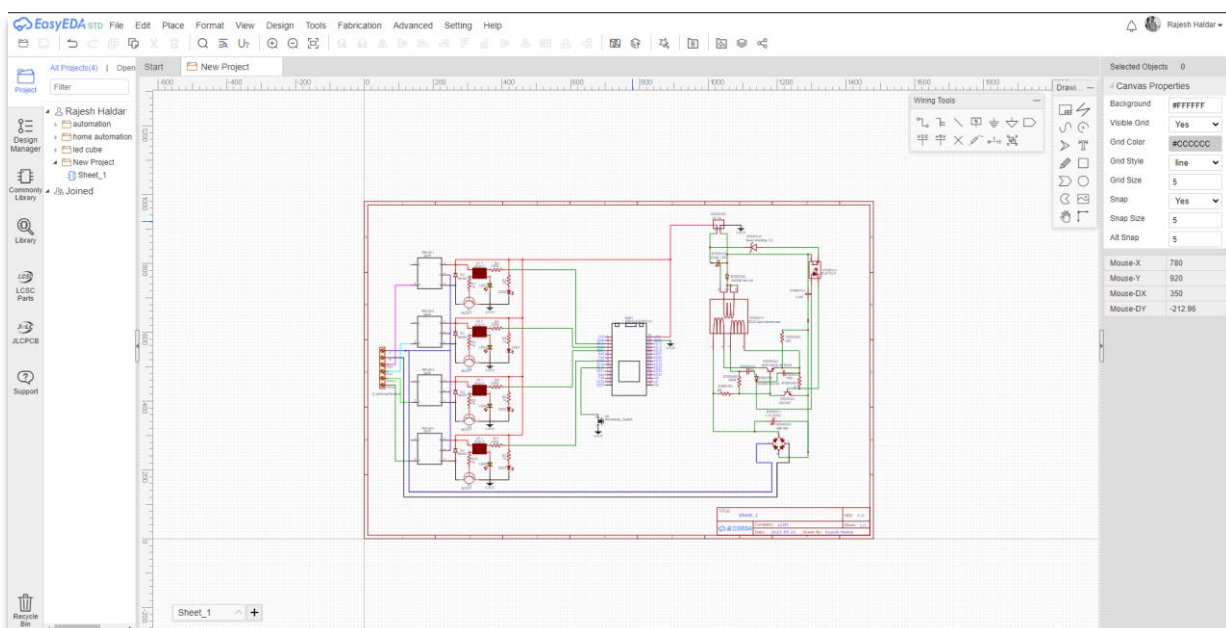
EasyEDA lets users design PCB layouts from schematics. It has a powerful PCB editor with auto-routing, design rule checking, and 3D visualization. Users can optimize the layout for manufacturing.

It can simulate circuits with a SPICE simulator. Users can do different types of analysis to check the performance and behavior of their designs.

It supports collaboration and cloud storage. Users can share their designs, work in real-time, and give feedback. They can access their projects from anywhere and track design changes.

It also supports many file formats and integrates with other tools and services. Users can export Gerber files for PCB fabrication, BOM for component sourcing, and other schematic and layout formats. They can also use JLCPCB and LCSC for PCB making and component buying.

EasyEDA is a versatile and user-friendly online software for electronic design. It simplifies the process of designing circuits. It is popular among electronics hobbyists, students, and engineers.



Circuit and Operation

In our project, there are a total of two parts - Hardware and Software. In this chapter, we will discuss the hardware part in detail and increase your understanding of it.

3.1. Switched-Mode Power Supply (SMPS)

The SMPS takes in an AC input voltage, typically from the mains power supply, and converts it to a higher voltage level using a transformer.

The higher voltage AC output from the transformer is then rectified using diodes to convert it into a pulsating DC waveform.

The rectified waveform is smoothed using filter capacitors, which reduce the ripple and provide a more stable DC voltage.

The SMPS utilizes a switching circuit, typically composed of a power MOSFET, to chop the DC voltage into high-frequency pulses.

The width of the pulses generated by the switching circuit is controlled by a PWM controller. The controller monitors the output voltage and adjusts the pulse width to maintain the desired output voltage level.

To provide feedback and control the switching circuit, an optocoupler is used. The optocoupler consists of an LED on the primary side and a phototransistor on the secondary side. The PWM controller drives the LED portion of the optocoupler.

The output voltage is compared to a reference voltage, and any difference is amplified and fed back to the optocoupler's phototransistor. The phototransistor's conduction varies based on the feedback signal, which in turn adjusts the PWM controller's duty cycle.

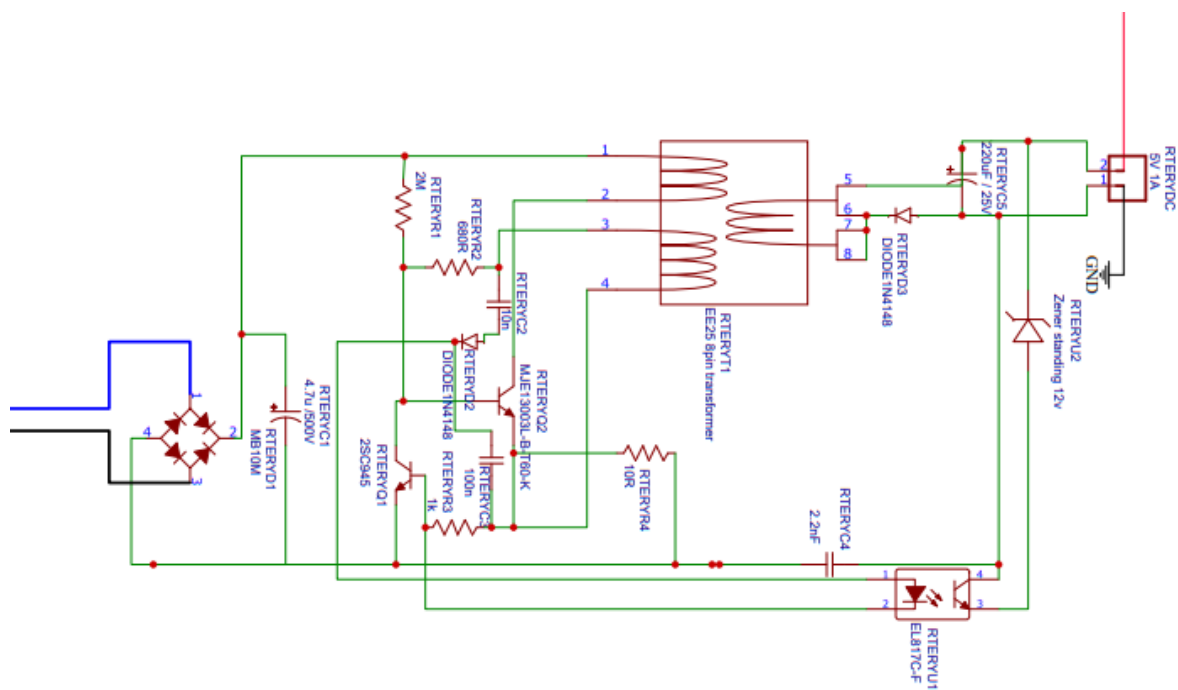
The optocoupler ensures electrical isolation between the high-voltage primary side and the low-voltage secondary side of the SMPS, protecting the connected devices and ensuring safety.

The PWM controller adjusts the duty cycle of the switching circuit to maintain a stable output voltage level despite variations in the input voltage or load conditions.

Finally, the regulated and isolated output voltage, typically 5V in this case, is obtained, and it is usually further filtered to reduce any residual noise or ripple.

By utilizing these principles, the SMPS with an optocoupler efficiently converts the input AC voltage to a regulated DC output voltage of 5V with a maximum current of 2A.

The diagram of our SMPS circuit is shown below.



3.2. IoT Hardware Kit

From the output of SMPS

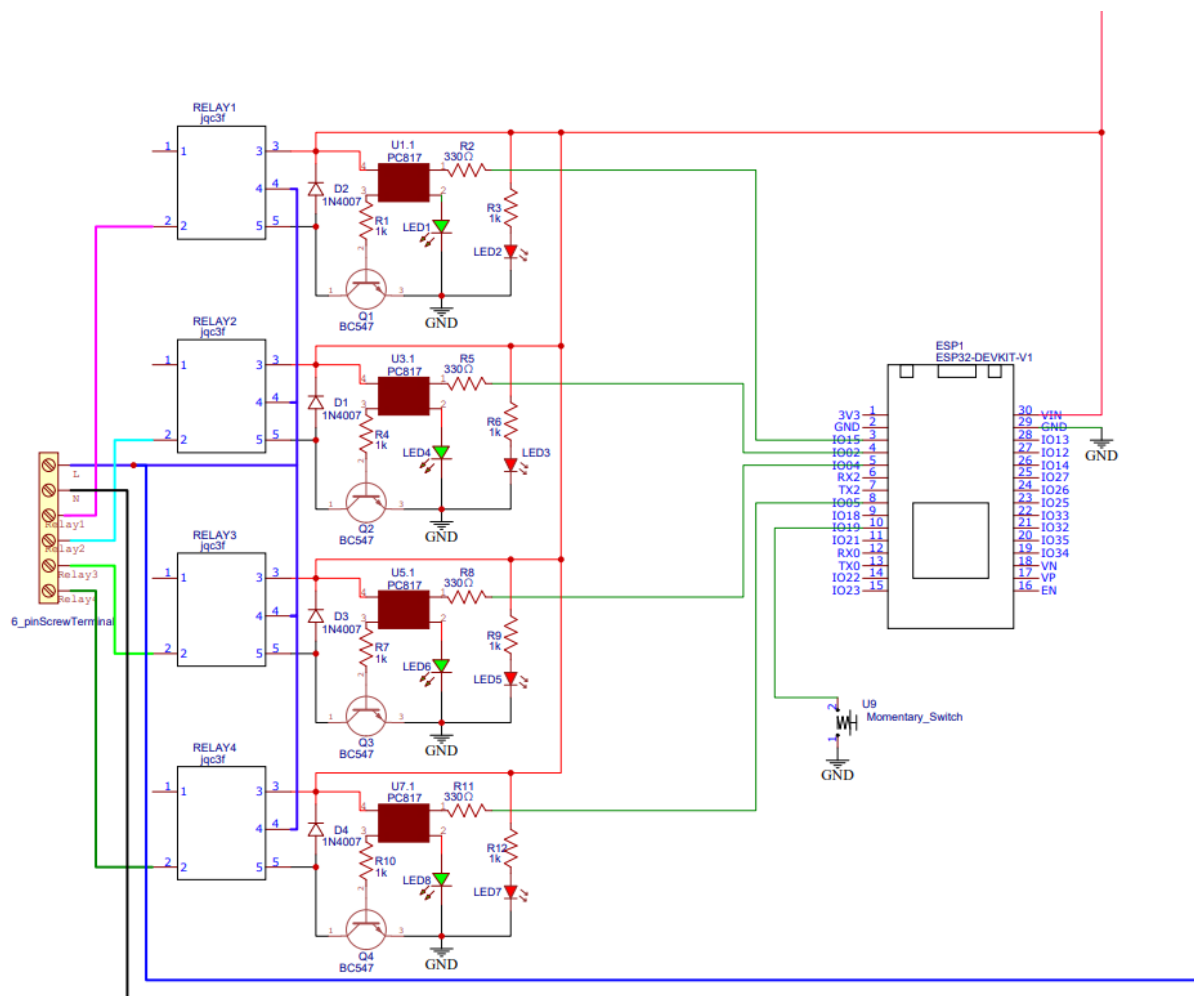
Setting up ESP32: The ESP32 is programmed with code that includes the necessary libraries for Wi-Fi connectivity and the Blynk platform. It establishes a Wi-Fi connection to your local network and connects to the Blynk server.

Configuring the Blynk App: Using the Blynk app, you create a project and design a user interface with buttons, sliders, or other widgets to control the relay module connected to the ESP32.

Communication between ESP32 and Blynk: The ESP32 communicates with the Blynk server over the Wi-Fi network. It establishes a secure connection and sends/receives data to/from the Blynk app based on the user's interactions with the interface.

Controlling the Relays: When you interact with the buttons or other widgets in the Blynk app, it sends commands to the Blynk server. The server then relays these commands to the ESP32, which interprets them and controls the corresponding relay channels (pins 15, 2, 4, and 5). This, in turn, switches the connected devices on or off.

Wi-Fi Reset: If needed, pressing the Wi-Fi reset button attached to pin 19 triggers a reset function in the ESP32 code, which resets the Wi-Fi connection and re-establishes the connection with the Blynk server.



3.3 Integration of Hardware with IoT Server

Explanation of how the hardware components and the Blynk virtual buttons work together

1. **ESP32:** The ESP32 microcontroller is connected to the hardware components, including the relay module and the Blynk platform. It serves as the interface between the physical devices and the virtual buttons.
2. **4-Channel Relay Module:** The relay module is connected to the ESP32 and controls the state of the relays based on the signals received from the microcontroller. Each relay is connected to a specific GPIO pin on the ESP32.

GPIO pin 15: This pin is connected to relay 1 of the module.

GPIO pin 2: This pin is connected to relay 2 of the module.

GPIO pin 3: This pin is connected to relay 3 of the module.

GPIO pin 5: This pin is connected to relay 4 of the module.

By toggling these GPIO pins HIGH or LOW, the ESP32 can control the corresponding relays, turning them ON or OFF.

3. **Blynk Virtual Buttons:** In the Blynk IoT setup, four virtual buttons (V1, V2, V3, V4) are defined. These virtual buttons are associated with the specific GPIO pins on the ESP32, allowing you to control the state of the relays remotely using the Blynk app or web dashboard.

V1: This virtual button is associated with GPIO pin 15, which controls relay 1.

V2: This virtual button is associated with GPIO pin 2, which controls relay 2.

V3: This virtual button is associated with GPIO pin 3, which controls relay 3.

V4: This virtual button is associated with GPIO pin 5, which controls relay 4.

When you press a virtual button in the Blynk app or web dashboard, it sends a command to the Blynk server. The server then forwards the command to the ESP32, specifically to the corresponding GPIO pin associated with the virtual button. The ESP32 receives the command and sets the GPIO pin HIGH or LOW based on the command's value (ON or OFF). This, in turn, controls the state of the relay connected to that GPIO pin, switching it ON or OFF accordingly.

3.4 Integration of IoT and AR

This is the main features and main attraction of our project.

To incorporate augmented reality (AR) into the system using the Blynk API, Vuforia engine, and Unity software, the following steps can be followed:

I. Blynk API Integration:

- a) Set up the Blynk API in your project by creating a Blynk account and obtaining an API token.
- b) Install the Blynk library in your development environment or platform (such as Arduino IDE or ESP-IDF for the ESP32 microcontroller).
- c) Use the Blynk library to establish a connection between your ESP32 and the Blynk server, allowing bidirectional communication.

II. Virtual Buttons in Blynk:

- a) In the Blynk app, create a project and add four virtual buttons (V1, V2, V3, V4) to the user interface.
- b) Configure each virtual button to send a specific command or value when pressed.

III. Database Setup for Image Targets in Vuforia:

- a) Create an account on the Vuforia Developer Portal (<https://developer.vuforia.com/>).
- b) Set up a new Vuforia project and access the Target Manager.
- c) Upload and define the image targets you want to use for AR experiences. Each target should have a unique identifier and associated metadata.

IV. Unity Integration with Vuforia:

- a) Install Unity software and the Vuforia Engine package.
- b) Create a new Unity project and import the Vuforia Engine package into your project.
- c) Set up Vuforia in Unity by configuring the license key and selecting the desired image targets from the Vuforia Target Manager.
- d) Develop the AR experience in Unity by adding 3D models, animations, or interactive elements to be displayed when the image targets are recognized.

V. Blynk and Vuforia Integration:

- a) Modify your ESP32 code to include the Blynk API integration and handle the commands received from the virtual buttons (V1, V2, V3, V4).
- b) Depending on the received commands, trigger specific actions in Unity, such as

showing/hiding objects or initiating animations, using the Unity API.

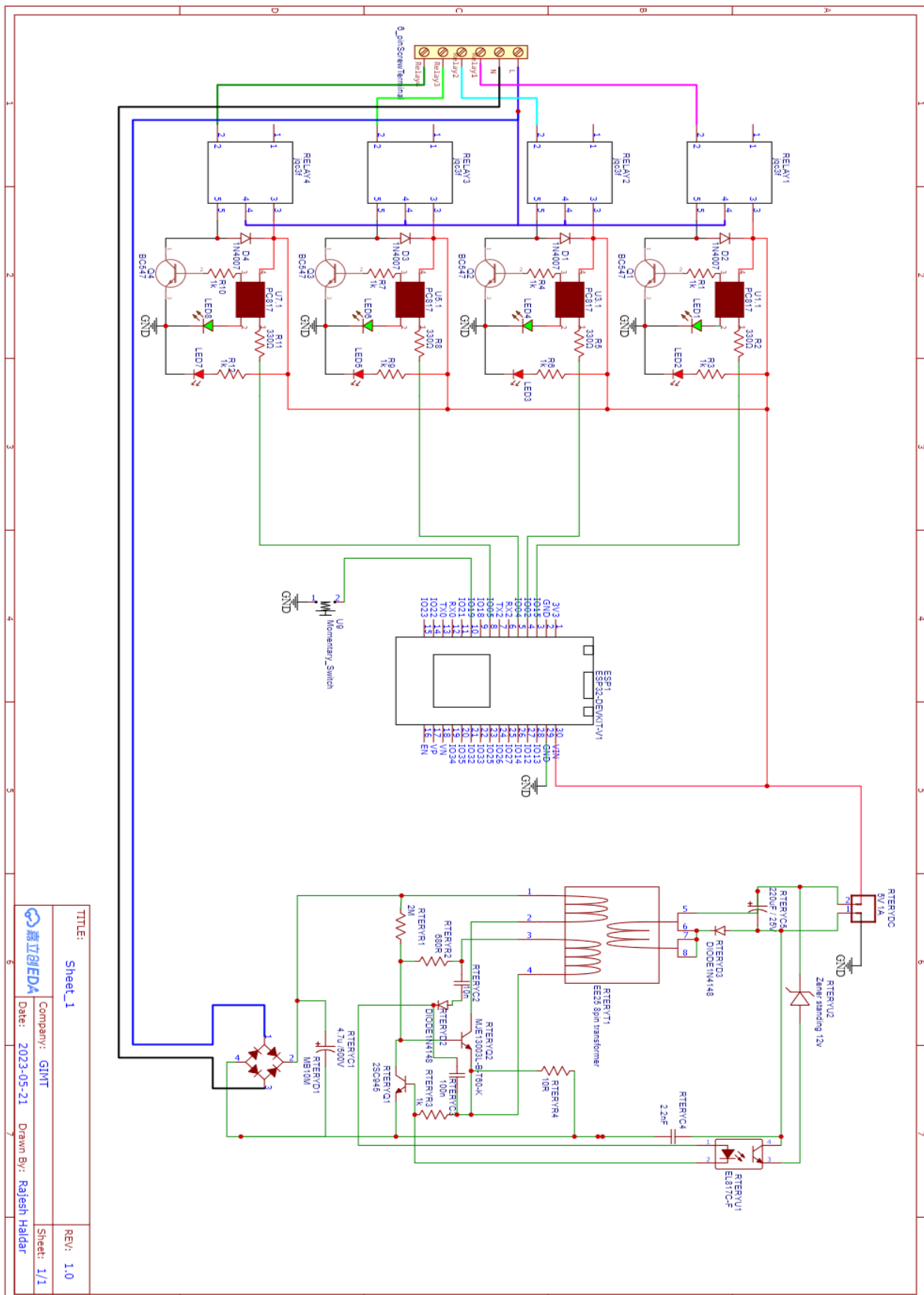
- c) Ensure that the ESP32 and the computer running the Unity project are connected to the same network for communication between Blynk and Unity.

VI. Testing and Deployment:

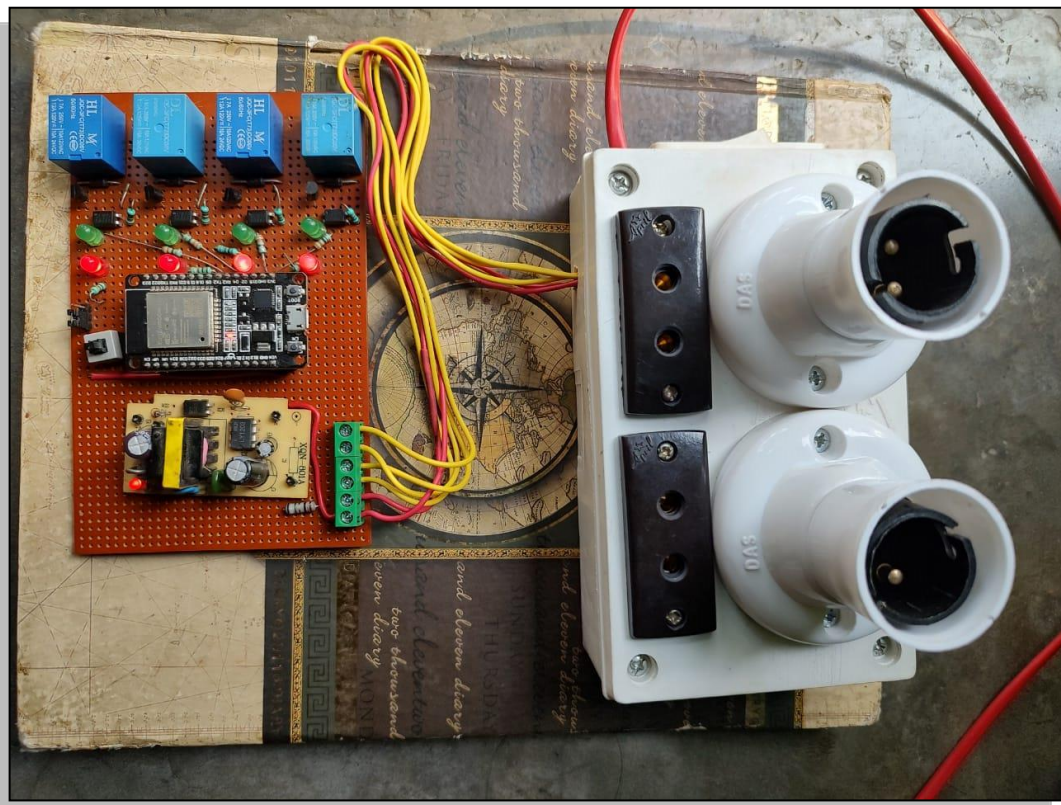
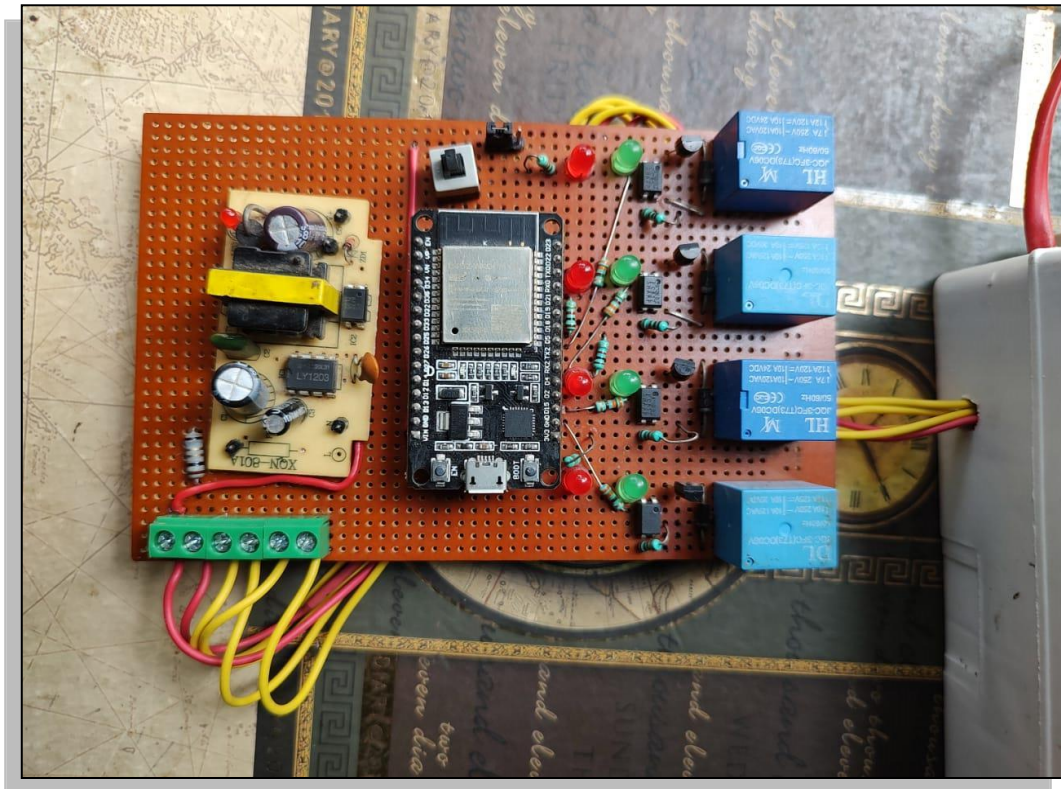
- a) Compile and upload the modified ESP32 code to your microcontroller.
- b) Build and run the Unity project on the target platform, such as a computer, mobile device, or AR headset.
- c) Launch the Blynk app on your mobile device and connect to the Blynk project.
- d) Interact with the virtual buttons in the Blynk app, which will send commands to the ESP32.
- e) When the ESP32 receives a command, it triggers actions in Unity based on the command received, resulting in AR experiences tied to the recognized image targets.

By integrating the Blynk API, Vuforia engine, and Unity software, we can control and trigger AR experiences by interacting with virtual buttons in the Blynk app. The ESP32 acts as a bridge between the Blynk app and the Unity project, allowing for dynamic control and interaction in the AR environment based on user input.

3.4. Circuit Schematic



3.5. Model Picture



3.6. Program Code

```
#define BLYNK_PRINT Serial
#define BLYNK_TEMPLATE_ID   "TMPLteLDXSUC"
#define TRIGGER_PIN 19
#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
#include <WiFiManager.h>

int timeout = 120;
char auth[] = "9wXBrdxNDj-hACZaSKSi2WuvHGIjSvz7";
char ssid[] = "";
char pass[] = "";

// Set your LED and physical button pins here
const int relay1 = 15;
const int relay2 = 2;
const int relay3 = 4;
const int relay4 = 5;

/*const int fanPin = 18;
const int button1 = 22;
const int button2 = 23;
const int button3 = 13;
const int button4 = 12;*/
BlynkTimer timer;

void checkPhysicalButton();

int relay1State = LOW;
int relay2State = LOW;
int relay3State = LOW;
int relay4State = LOW;

int btnState = HIGH;

WidgetLCD lcd(V5);

void setup()
{
```

```
WiFiManager wm;
//wm.resetSettings();
WiFi.mode(WIFI_STA);
Serial.begin(9600);
Serial.println("\n Starting");
pinMode(TRIGGER_PIN, INPUT_PULLUP);

bool res;
res = wm.autoConnect("AioT_Home","password");
if(!res) {
    Serial.println("Failed to connect");
    // ESP.restart();
}
else {
    //if you get here you have connected to the WiFi
    Serial.println("connected... :)");
}
Serial.begin(9600);

Blynk.begin(auth, ssid, pass);

// Setup physical button pins
pinMode(button1, INPUT_PULLUP);
pinMode(button2, INPUT_PULLUP);
pinMode(button3, INPUT_PULLUP);
pinMode(button4, INPUT_PULLUP);

// Setup relay pins
pinMode(relay1, OUTPUT);
pinMode(relay2, OUTPUT);
pinMode(relay3, OUTPUT);
pinMode(relay4, OUTPUT);

// Connect to Wi-Fi
WiFi.begin(ssid, pass);
while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi...");
}
Serial.println("Connected to WiFi");

// Connect to Blynk
```

```
Blynk.begin(auth, WiFi.SSID().c_str(), WiFi.psk().c_str());
Serial.println("Connected to Blynk");

// Setup a function to be called every 100 ms
timer.setInterval(100L, checkPhysicalButton);
}

// Every time we connect to the cloud...
BLYNK_CONNECTED() {
    // Request the latest state from the server
    Blynk.syncVirtual(V1);
    Blynk.syncVirtual(V2);
    Blynk.syncVirtual(V3);
    Blynk.syncVirtual(V4);
    Blynk.syncVirtual(V5);
}

// When App button is pushed - switch the state
BLYNK_WRITE(V1) {
    relay1State = param.asInt();
    digitalWrite(relay1, relay1State);
    lcd.print(0,0,"Relay1: ");
    lcd.print(8,0,relay1State ? "ON " : "OFF");
}

BLYNK_WRITE(V2) {
    relay2State = param.asInt();
    digitalWrite(relay2, relay2State);
    lcd.print(0,1,"Relay2: ");
    lcd.print(8,1,relay2State ? "ON " : "OFF");
}

BLYNK_WRITE(V3) {
    relay3State = param.asInt();
    digitalWrite(relay3, relay3State);
    lcd.print(0,0,"Relay3: ");
    lcd.print(8,0,relay3State ? "ON " : "OFF");
}

BLYNK_WRITE(V4) {
    relay4State = param.asInt();
    digitalWrite(relay4, relay4State);
```

```

    lcd.print(0,1,"Relay4: ");
    lcd.print(8,1,relay4State ? "ON " : "OFF");
}

/*BLYNK_WRITE(V0) {
    int fanSpeed = param.asInt();
    ledcWrite(0,fanSpeed); // channel , duty cycle
}*/

void checkPhysicalButton()
{
    if (digitalRead(button1) == LOW) {
        // btnState is used to avoid sequential toggles
        if (btnState != LOW) {

            // Toggle LED state
            relay1State = !relay1State;
            digitalWrite(relay1, relay1State);

            // Update Button Widget
            Blynk.virtualWrite(V1, relay1State);
        }
        btnState = LOW;
    } else {
        btnState = HIGH;
    }

    if (digitalRead(button2) == LOW) {
        if (btnState != LOW) {
            relay2State = !relay2State;
            digitalWrite(relay2, relay2State);
            Blynk.virtualWrite(V2, relay2State);
        }
        btnState = LOW;
    } else {
        btnState = HIGH;
    }

    if (digitalRead(button3) == LOW) {
        if (btnState != LOW) {
            relay3State = !relay3State;
            digitalWrite(relay3, relay3State);
        }
    }
}
```

```

        Blynk.virtualWrite(V3, relay3State);
    }
    btnState = LOW;
} else {
    btnState = HIGH;
}

if (digitalRead(button4) == LOW) {
    if (btnState != LOW) {
        relay4State = !relay4State;
        digitalWrite(relay4, relay4State);
        Blynk.virtualWrite(V4, relay4State);
    }
    btnState = LOW;
} else {
    btnState = HIGH;
}
}

void loop()
{
    if ( digitalRead(TRIGGER_PIN) == LOW) {
        WiFiManager wm;

        wm.resetSettings();
        wm.setConfigPortalTimeout(timeout);

        if (!wm.startConfigPortal("OnDemandAP")) {
            Serial.println("failed to connect and hit timeout");
            delay(3000);
            ESP.restart();
            delay(5000);
        }
        Serial.println("Connected... :)");
    }
    Blynk.run();
    timer.run();
}

```


Project Costing

SL No.	Name of component's	Quantity	Price (₹)
1	ESP32-WROOM-32	1	550
2	Transistor— BC547	4	25
3	Optocoupler— PC817	4	36
4	Relay— JQC-3FC(T73)DC06V	4	84
5	Diode— 1N4007	4	20
6	LED— 5MM	8	20
7	Power Supply— 5Volt 2Amp	1	85
8	Momentary switch or tactile switch	1	5
9	Resistor 330 Ω , 1K Ω	4 8	18
10	Pitch screw terminal	3	12
11	AC lamp holder	2	24
12	AC Socket	2	24
13	AC Switch	1	12
14	Connecting Wires		43
15	PVC Box	1	45
Total			1003



Result

The home automation system developed using Blynk, ESP32, and a 4-channel relay module offers convenient control over various household devices. With the integration of Blynk's IoT platform and the ESP32 microcontroller, users can remotely control the state of four relays using virtual buttons in the Blynk app or web dashboard.

The system allows users to control high-power devices, such as lights, motors, or appliances, by simply pressing virtual buttons on their mobile devices or computers. The virtual buttons in the Blynk app are associated with specific GPIO pins on the ESP32, which are connected to the relays through the relay module. When a virtual button is pressed, the corresponding GPIO pin receives a command from the Blynk server, and the ESP32 toggles the state of the relay connected to that GPIO pin accordingly.

Additionally, the system includes a push button that triggers the Wi-Fi configuration process. When this button is pressed, the ESP32 initiates a Wi-Fi configuration mode, allowing users to enter new Wi-Fi credentials through the Wi-Fi Manager.

The Blynk platform enables not only remote control but also provides real-time feedback through an LCD widget that displays the status of the relays. The LCD widget shows the current state (ON or OFF) of each relay, providing visual confirmation of the device's status.

Overall, this home automation system offers a user-friendly interface for controlling and monitoring household devices from anywhere, providing convenience, flexibility, and enhanced control over your home environment.



Conclusion

This project has demonstrated the potential of IoT technology in home automation, offering users the ability to control high-power devices with ease. The integration of the relay module with optocouplers ensures safe and reliable operation, protecting the microcontroller from potential electrical disturbances.

The inclusion of a push button for Wi-Fi configuration enhances the system's flexibility by allowing users to easily reset and update the Wi-Fi settings. The utilization of the Wi-Fi Manager library simplifies the configuration process and provides a user-friendly experience for connecting the system to a Wi-Fi network.

The real-time feedback provided through the LCD widget in the Blynk app offers users a visual representation of the relay states, ensuring accurate monitoring and control of the connected devices.

Overall, this project showcases the potential of combining IoT platforms, microcontrollers, and relay modules to create sophisticated home automation systems. The developed system provides users with the convenience of remote control, flexibility in managing devices, and real-time feedback, offering an enhanced and streamlined home automation experience.

The motivation of this project came from the desire to make a Home Automation system by the Augmented Reality (AR) and Internet of Things (IoT). To control home appliances both conventional and novel interfaces that will be operated electrically.

In this project we have implemented the Augmented Reality (AR) technology, and we have future planning of upgrading this project with Artificial intelligence (AI) that can help control home appliances more interesting and futuristic way.

This is a raw idea that we have thought of. We have to do a vast work to implement this idea.

Reference

Here are some sites from where we took help in the process of implementing the project—

- www.easyeda.com
- developer.vuforia.com
- www.arduino.cc
- blynk.io
- www.circuitschools.com
- www.mischianti.org
- Components101.com
- images.google.com
- potentiallabs.com
- www.alldatasheet.com

Except this sites we also took help from some blogs etc.

This project seem to be an ordinary one but implementing this was not an easy task