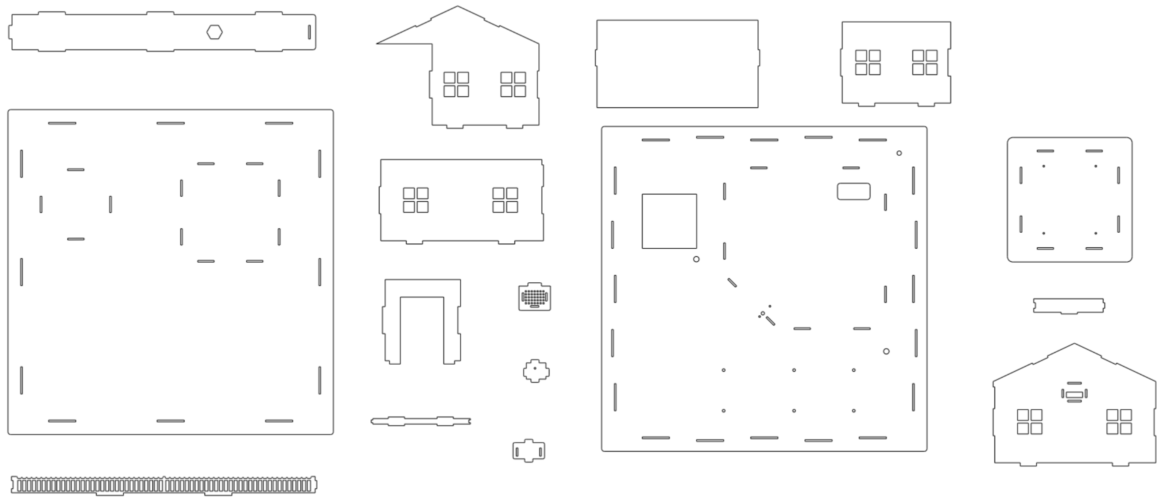
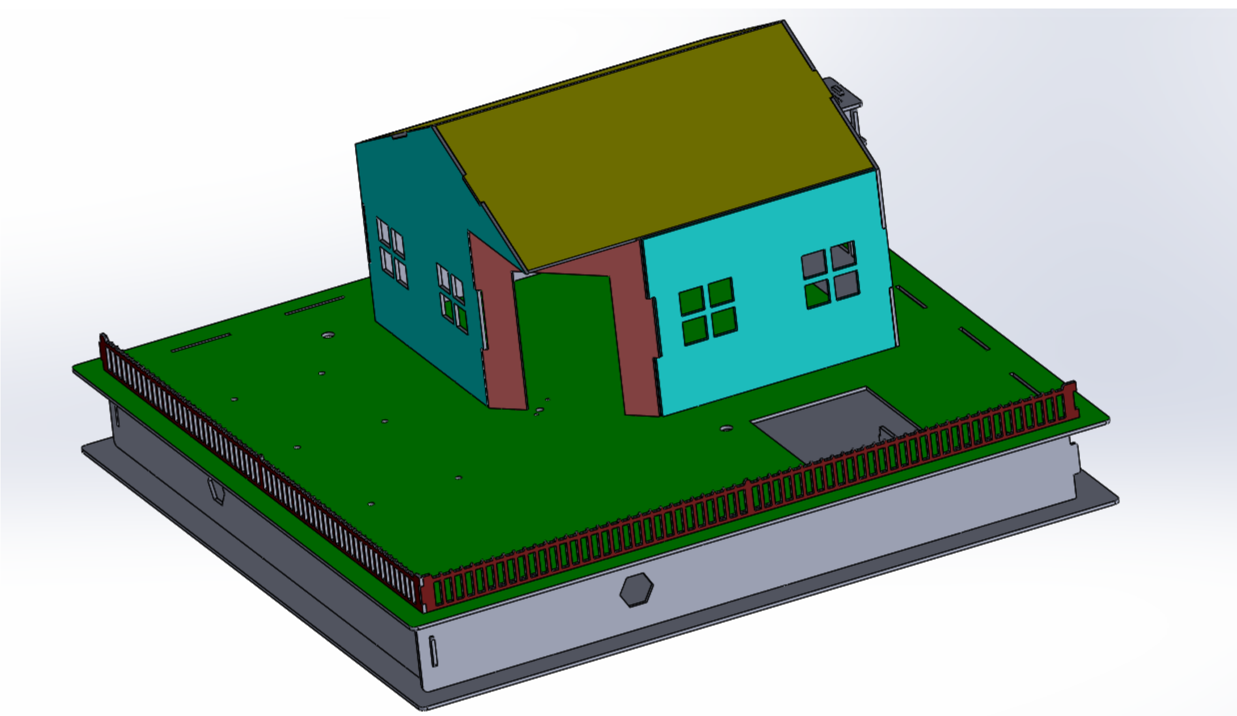
# A Small-Scale Smart Home Automation and ,IoT Integration

## Model

Wooden MDF boards with a diameter of 3 millimeters have been used for laser cutting the model. The model is divided into small parts, and each piece is designed using SolidWorks software. The parts do not require gluing and fit tightly together.



Finally, all parts have been configured in 3D software, ensuring the accuracy of their dimensions.



After laser engraving the designs onto the wood, I assembled the parts and finalized the configuration.

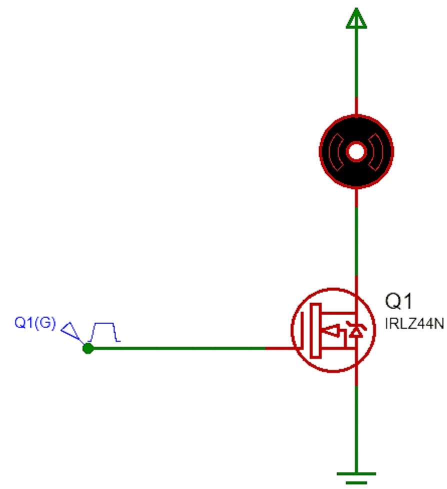
## Sensors and Modules

An LM35 sensor was used for temperature control. This sensor has analog output, making it easy to work with and readily available in the market. The SMT172 sensor was also tested, which proved to be more challenging to work with. After receiving data from the sensor and considering the user's desired temperature, the controller adjusts the fan (cooler) speed accordingly.

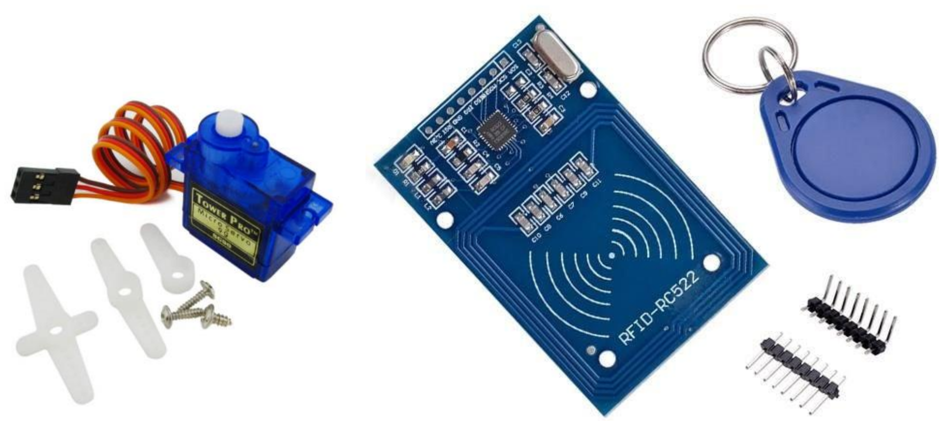

For the fire section, an MHMQ-2 sensor has been used. Due to its high accuracy and reasonable price, this sensor is one of the most popular gas detection sensors. It can detect the type of gas released and the level of gas accumulation in the room. The data reading from the sensor is done in analog form.


The irrigation and water level adjustment section used a pump and height sensor. No information on the pump was available, so a 5-volt water pump was obtained. After testing, satisfactory results were obtained, and the pump's controllability with a driver was also tested. Therefore, two water pumps are controlled using two MOSFETs via PWM. Using a water level detection sensor designed for the

Arduino package and considering its widespread use in this series of boards, I chose a controller for water level adjustment. It works by taking the desired height input from the user and adjusting the water level in the pool based on that.

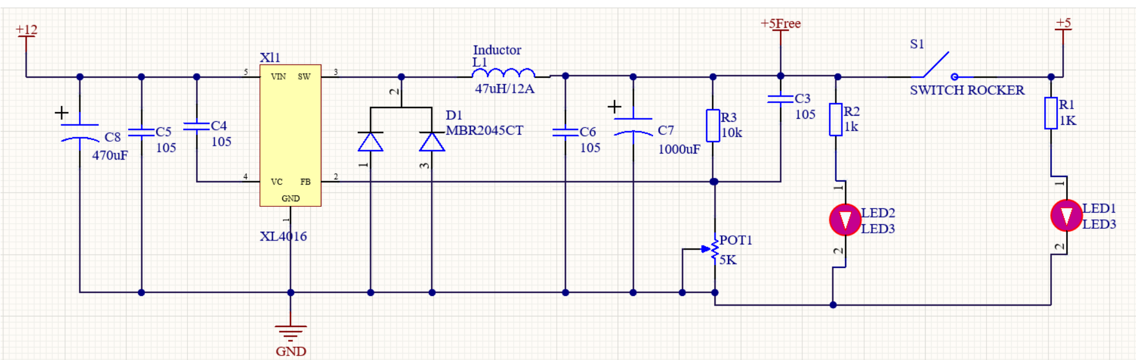



A servo motor and an RFID module control the entry door for automatic entry. The RFID module is the only available module on the market for this purpose, offering a suitable price and performance. The SG90 servo motor is also a low-cost servo motor capable of opening and closing this door.

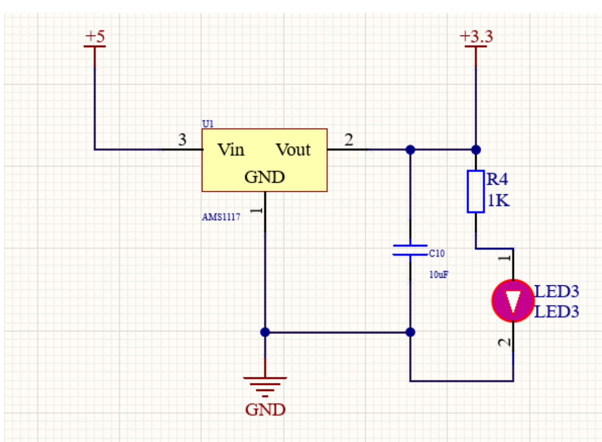


## Electronic Circuit:

Power Supply: The circuit and home power source is a 12-volt adapter capable of supplying two amps of current. The reason for using a 12-volt adapter is to illuminate the house. Typically, LED strip lights are powered by a 12-volt voltage. Voltage regulation is required to power the sensors and microcontroller. For this purpose, a buck converter to 5 volts was used. To supply 3.3 volts, a resistive voltage regulator was used. To supply 5 volts, an XL4016 regulator was used. The circuit schematic is visible below. The feedback pin can be utilized to adjust the output voltage of this buck converter. As seen in the circuit, using a voltage divider and a potentiometer, I could control the feedback pin voltage of the regulator and adjust the output voltage to around 5 volts. The datasheet for this regulator is also available in the attached files.

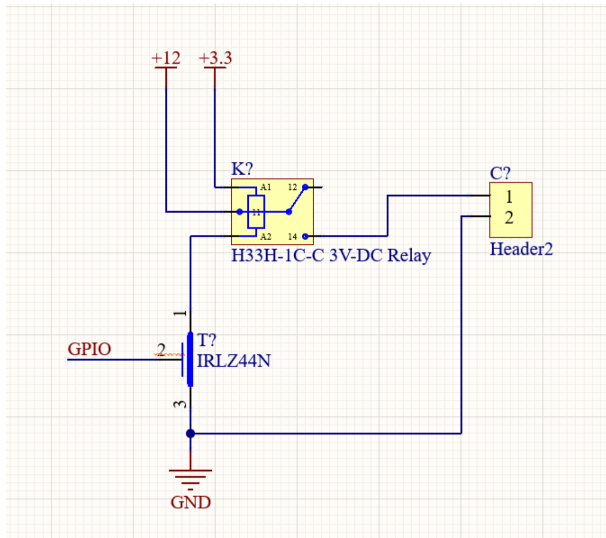



An AMS1117 regulator, one of the most commonly used resistive regulators and used in most microcontrollers, was also used to supply 3.3 volts. The datasheet for this regulator is available in the attached files. The circuit for this regulator is visible below.



## Lighting

The home lighting control circuit is also designed to control the actual lights in homes. The schematic of the lighting control circuit is drawn below.

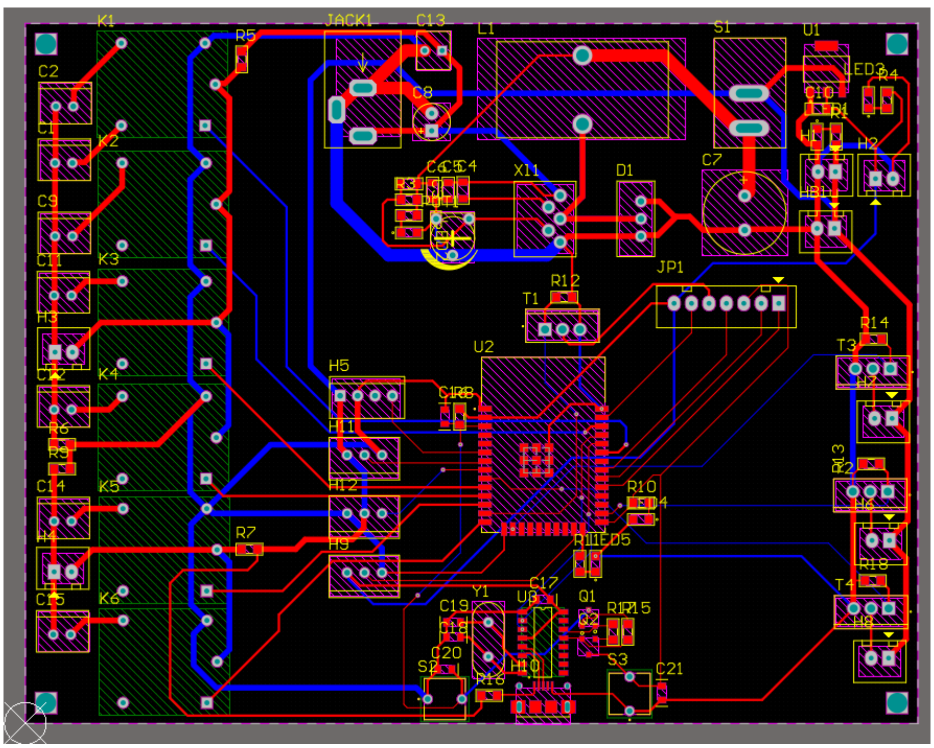


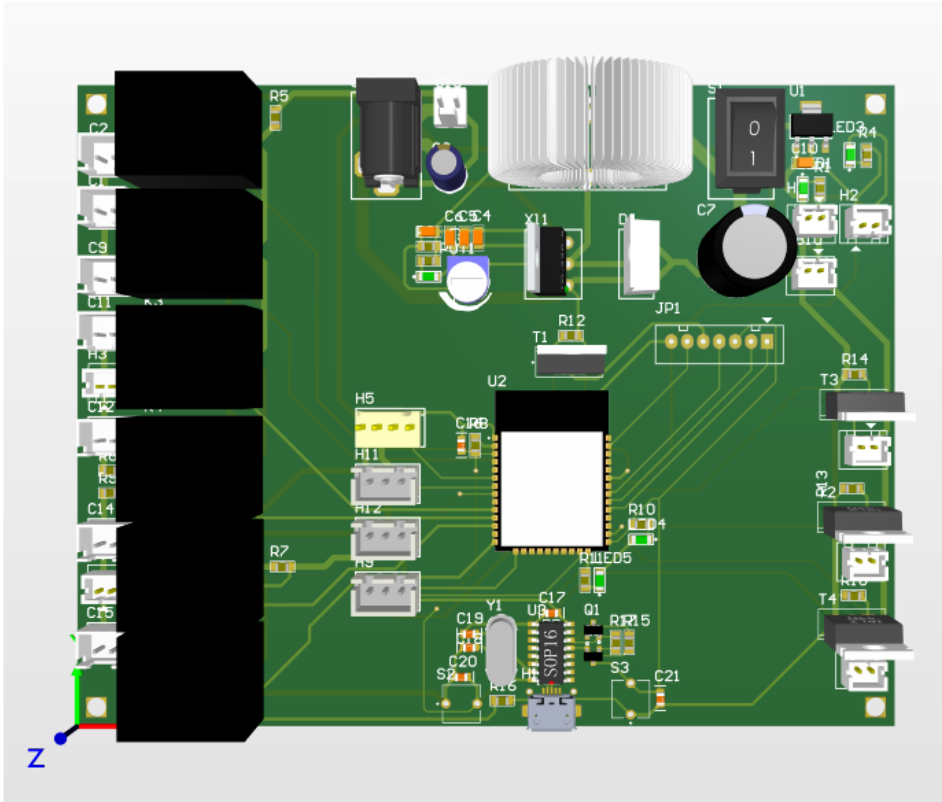

As you can see, to control the relay, an ESP32 microcontroller and a MOSFET are used to supply the required current for switching the relay on and off. When the microcontroller applies the necessary voltage (3.3 volts) to the gate pin of the MOSFET, the voltage across the relay's trigger terminals will be 3.3 volts, and a current of 150 milliamps will flow through it. Consequently, the relay connection will change, and the lamp (LED) will turn on. The datasheet for the IRLZ44N MOSFET is available in the attached files.

During the circuit's initial testing on the breadboard, attention was not paid to the current required to trigger the relay. The microcontroller pins were used to supply voltage and trigger current to activate the relay. After assembling the board and testing it, I realized the problem. To resolve this issue, I placed a transistor between the relay and the microcontroller pin.

## PCB Design

After testing the components separately and achieving satisfactory results, the circuits of various sections were correctly obtained. Initially, the circuit schematics were drawn using Altium software, and then the PCB layout was designed. The design files were then handed over to Koosar Circuits for printing, which produced relatively high-quality outputs. An important point to note is that I did not use an ESP32 development board; instead, I mounted the ESP32 module along with the IC and the USB to UART conversion circuit directly onto the board.



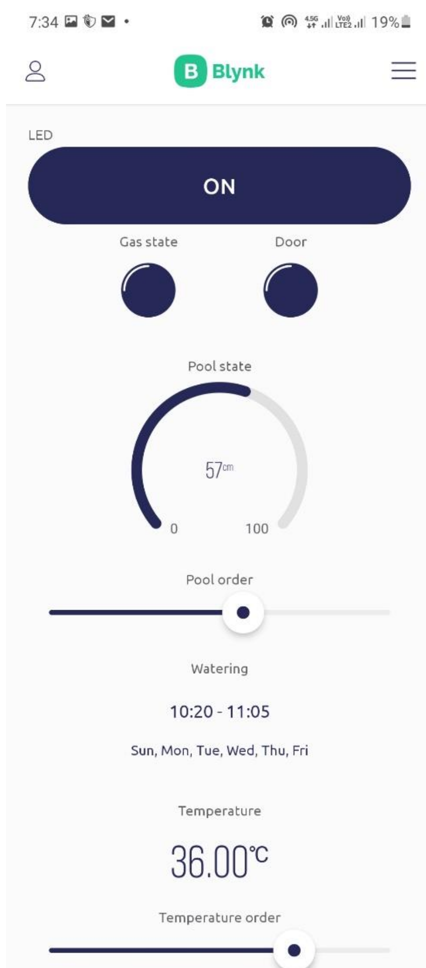



## **Software**:

- **Blynk**

Blynk is an online service that provides wireless communication services for various microcontrollers, including ESP8266 and ESP32. It also gives you the tools to design a control panel, including buttons, sliders, images, charts, and data display tools.

With the help of this tool and the Blynk application, I was able to design a configured page for the smart home. This page, shown in the figure below, includes lighting switches, a display of the pool water level and house temperature, a pool water level control slider, a house temperature control slider, the current status of the house door (open or closed), and a gas alert.




Using the Blynk library, microcontroller programming was done to communicate with the Blynk server and application for sending and receiving data. With the help of virtual variables called Virtual Pins, I could send and receive numerical values (integer and float) and character strings (string). Therefore, for each section of the house, such as lighting, pool, door status, gas status, and temperature, I defined a Virtual Pin and sent and received the necessary information by changing the values of these virtual variables on the Blynk server.

- **OTA**

One of the unique features of the ESP32 microcontroller is its wireless programming capability. With this feature, physical cables are no longer needed to program the microcontroller. This means that if the ESP32 is connected to the internet via WIFI, it can receive the program file through WIFI communication and the program itself. This capability is beneficial for updating the smart home microcontroller board's program without needing physical cable connections, and even remote programming becomes possible.

- **Smart Connection**

Another feature of the ESP32 microcontroller is its ability to connect to WIFI networks. Additionally, with the help of the WIFI. I provided smart functionality to change the microcontroller's WIFI access point in the library. When the microcontroller is powered on for the first time, it will wait for an SSID and password to connect to a WIFI network. Users can use the ESPTouch: SmartConfig application to send the necessary information for connecting the ESP32 to their home WIFI network .