

**S.K.D.A.V. GOVERNMENT POLYTECHNIC**

DEPARTMENT OF INFORMATION TECHNOLOGY,

ROURKELA, ODISHA, 769012

**SMART SWITCH BOARD**

A project report in partial fulfilment of the requirement for the award

Of

**DIPLOMA IN INFORMATION TECHNOLOGY**

BY

**Mr AKASH KUMAR SAHOO F21011024001 (IT-6TH SEM)**

**GUIDEDBY MISS STELLA KUJUR**

S.K.D.A.V. GOVERNMENT

POLYTECHNIC ROURKELA-12

**CERTIFICATE**

(Approved AICTE, Affiliated to SCTE&VT, Odisha)

This is to certify that this is entitled, “**SMART SWITCH BOARD**” submitted by Mr Akash Kumar Sahooin partial fulfilment of the requirement for the award of Diploma 6thsemester in**INFORMATION TECHNOLOGY** engineering during session **2023 - 2024**in the department of **INFORMATION TECHNOLOGY S.K.D.A.V. GOVERNMENT POLYTECHNIC, ROURKELA.**

It is an authentic work carried out by them under my supervision and guidance. To the best of my knowledge, matter embodied in the project has not been submitted to any other university/institute for the award of any degree or diploma.

**ACKNOWLEDGMENT**

First and for most we offer our sincere gratitude and respect to our project supervisor, **MISS STELLA KUJUR**Department of **Information Technology** Engineering, for her valuable guidance and suggestions to us during our study. We consider ourselves extremely lucky to have the opportunity of associating our self with her for one year. This project was made possible by her patience and persistence. We wish to express our deep gratitude to all those who extended their helping hands towards us in various ways during our short tenure at **S.K.D.A.V. Government Polytechnic, Rourkela.**

We express our sincere thanks to **Miss STELLA KUJUR, HOD** Department of **Information Technology** and also the other staff members of Department of **Information Technology, S.K.D.A.V. Government Polytechnic, Rourkela**-**12** for providing us the necessary facilities that is required to conduct the experiment and compute our project

**ABSTRACT**

* The smart switch board is a technological advancement in the field of home automation, designed to revolutionize traditional electrical systems. This innovative device integrates smart technology to provide users with enhanced control, automation, and monitoring capabilities over their connected appliances and devices.
* At its core, the smart switch board allows users to remotely control electrical components through various interfaces, such as mobile applications. This remote accessibility not only enhances convenience but also contributes to energy efficiency and overall user experience.
* As technology continues to evolve, the smart switch board represents a significant step towards creating intelligent, interconnected homes. While ensuring convenience and efficiency, it also emphasizes the importance of energy conservation and sustainable living. As these devices become more prevalent, they have the potential to redefine how individuals interact with and manage their domestic electrical systems, paving the way for a smarter and more connected future.

**CONTENT**

|  |  |  |
| --- | --- | --- |
| **SL NO.** | **CONTENT** | **PAGE NO.** |
| 01 | INTRODUCTION | 6 |
| 02 | OVERVIEW | 7 |
| 03 | FEASIBILITY STUDY | 8 |
| 04 | HARDWARE AND SOFTWARE REQUIREMENTS | 9 |
| 05 | ARDUINO | 11 |
| 06 | BREAD BOARD | 12 |
| 07 | JUMPER WIRE | 13 |
| 08 | ESP-32 | 14 |
| 09 | RELAY | 15 |
| 10 | WIRE | 171 |
| 11 | SWITCH BOARD | 18 |
| 12 | ARDUINO IDE | 20 |
| 13 | CODING | 28 |
| 14 | CIRCUIT DIAGRAM | 34 |
| 15 | TESTING | 35 |
| 16 | FURTHER ENHANCEMENT | 36 |
| 17 | CONCLUSION | 37 |
| 18 | REFERENCE | 38 |

**INTRODUCTION**

* A smart switchboard is an advanced and technologically enhanced version of the traditional electrical switchboard.
* It incorporates intelligent features and connectivity options to provide users with greater control and automation of their home or office electrical systems.
* The primary goal of a smart switchboard is to enhance convenience, energy efficiency, and overall management of electrical devices and systems.

**OVERVIEW**

* The main purpose of the smart switch board is “we control our electric component through mobile”.
* A smart switch board is a device that allows homeowners to control various aspects of their home.
* Smart switch boards can be controlled via a webpage on a smartphone or tablet. This allows homeowners to have complete control over their home while they’re away from it.
* For example, if you forget to turn the lights off before leaving for work, you can simply open any search engine enter Ip address of device and turn off them under 20 meters.
* Smart switch board, also known as smart home technology, allows homeowners to manage and monitor different aspects of their home remotely or through automated processes.

**FEASIBILITY STUDY**

In our project "SMART SWITCH BOARD" the main Aim of Feasibility study is to determine whether it could be financially and technically feasible to develop the product.

Thereare3 types of feasibility study:

* **Technical feasibility:** This project is technically feasible because we already know uses of required programs is Arduino UNO programming languages.
* **Operational feasibility:** This project is operationally feasible because the system and the user interface is user friendly and it does not require much expertise and training.
* **Financial feasibility:** - It involves study to establish the cost benefit analysis. Money spent on the system must be recorded in the form of benefit from the system.
* Our project is technically, operationally & financially feasible to the user.

**Hardware and Software Requirements**

**Hardware: -**

|  |  |
| --- | --- |
| Bread board |  |
| Jumper wire |  |
| Esp-32 |  |
| Relay |  |
| Wire |  |
| switch board |  |

**Software: -**

**ARDUINO IDE**



**ARDUINO**

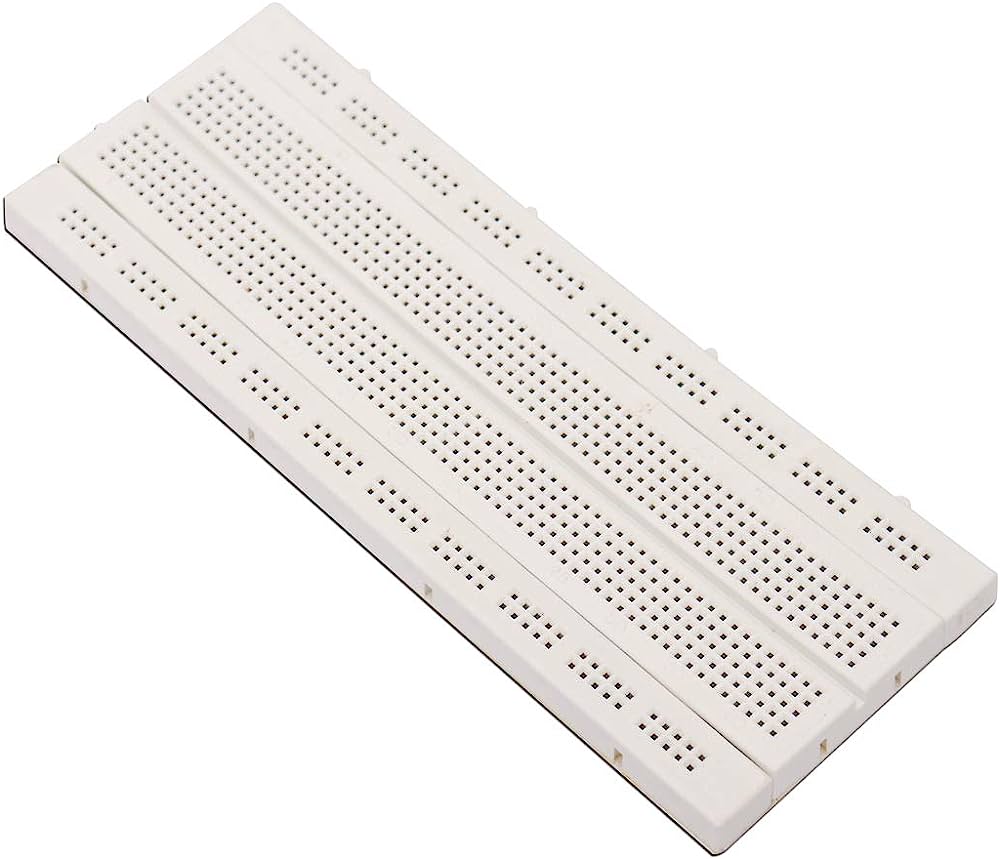
**Arduino** is an [open-source hardware](https://en.wikipedia.org/wiki/Open-source_hardware) and [software](https://en.wikipedia.org/wiki/Open-source_software) company, project, and user community that designs and manufactures [single-board microcontrollers](https://en.wikipedia.org/wiki/Single-board_microcontroller) and [microcontroller](https://en.wikipedia.org/wiki/Microcontroller) kits for building digital devices. It’s hardware products are licensed under a [CC BY-SA license](https://en.wikipedia.org/wiki/Creative_Commons_license), while software is licensed under the [GNU Lesser General Public License](https://en.wikipedia.org/wiki/GNU_Lesser_General_Public_License) (LGPL) or the [GNU General Public License](https://en.wikipedia.org/wiki/GNU_General_Public_License) (GPL), permitting the [manufacture](https://en.wikipedia.org/wiki/Manufacturing) of Arduino boards and software distribution by anyone. Arduino boards are available commercially from the official [website](https://en.wikipedia.org/wiki/Website) or through authorized distributors. Arduino board designs use a variety of [microprocessors](https://en.wikipedia.org/wiki/Microprocessor) and controllers. The boards are equipped with sets of digital and analog [input/output](https://en.wikipedia.org/wiki/Input/output) (I/O) pins that may be interfaced to various expansion boards ('shields') or [breadboards](https://en.wikipedia.org/wiki/Breadboards) (for prototyping) and other circuits. The boards feature serial communications interfaces, including [Universal Serial Bus](https://en.wikipedia.org/wiki/Universal_Serial_Bus) (USB) on some models, which are also used for loading programs. The microcontrollers can be programmed using the [C](https://en.wikipedia.org/wiki/C_(programming_language)) and [C++](https://en.wikipedia.org/wiki/C%2B%2B) [programming languages](https://en.wikipedia.org/wiki/Programming_language), using a standard API which is also known as the **Arduino language**, inspired by the [Processing language](https://en.wikipedia.org/wiki/Processing_(programming_language)) and used with a modified version of the Processing IDE. In addition to using traditional [compiler](https://en.wikipedia.org/wiki/Compiler) [toolchains](https://en.wikipedia.org/wiki/Toolchains), the Arduino project provides an [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) and a command line tool developed in [Go](https://en.wikipedia.org/wiki/Go_(programming_language)).

The Arduino project began in 2005 as a tool for students at the [Interaction Design Institute Ivrea](https://en.wikipedia.org/wiki/Interaction_Design_Institute_Ivrea), Italy, aiming to provide a low-cost and easy way for novices and professionals to create devices that interact with their environment using [sensors](https://en.wikipedia.org/wiki/Sensor) and [actuators](https://en.wikipedia.org/wiki/Actuator). Common examples of such devices intended for beginner hobbyists include simple [robots](https://en.wikipedia.org/wiki/Robot), [thermostats](https://en.wikipedia.org/wiki/Thermostat) and [motion detectors](https://en.wikipedia.org/wiki/Motion_detector).

The name *Arduino* comes from a bar in [Ivrea](https://en.wikipedia.org/wiki/Ivrea), Italy, where some of the founders of the project used to meet. The bar was named after [Arduino of Ivrea](https://en.wikipedia.org/wiki/Arduin_of_Ivrea), who was the [margrave](https://en.wikipedia.org/wiki/Margrave) of the [March of Ivrea](https://en.wikipedia.org/wiki/March_of_Ivrea) and [King of Italy](https://en.wikipedia.org/wiki/King_of_Italy) from 1002 to 1014.



**BREAD BOARD:-**



A breadboard is a fundamental tool in electronics prototyping, providing a platform for engineers, hobbyists, and students to build and test electronic circuits quickly and without the need for soldering. It consists of a plastic board with a grid of holes into which electronic components like resistors, capacitors, and integrated circuits can be inserted.

The board's surface is typically divided into columns and rows, with metal strips underneath connecting the holes in each row or column. This arrangement allows for the temporary connection of components and wires without the need for soldering. The term "breadboard" originates from the idea of a board on which you can prototype and experiment with circuit designs before committing them to a more permanent form.

Breadboards are versatile and reusable, making them ideal for experimenting with various circuit configurations. They are commonly used in educational settings, electronics labs, and by hobbyists for testing and refining circuit designs before moving to a more permanent circuit board.

**JUMPER WIRE:-**



Jumper wires are short wires with connectors at each end, commonly used in electronics to create temporary connections on a breadboard or between components. They play a crucial role in prototyping and experimenting with circuits, allowing for the easy and quick establishment of electrical connections without the need for soldering.

Typically, jumper wires have pins or connectors that can be easily inserted into the holes of a breadboard, connecting various points on the board to form a circuit. The other end of the wire may have a pin, a male header, or a connector that can be plugged into other electronic components, such as sensors, microcontrollers, or integrated circuits.

Jumper wires come in different lengths, colors, and styles, providing flexibility in organizing and identifying connections within a circuit. They are essential tools for anyone working with electronics, as they enable the creation and modification of circuits on a breadboard in a simple and reversible manner.

**ESP-32:-**

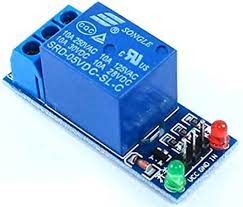


The ESP32 is a versatile and widely used microcontroller and system-on-a-chip (SoC) in the field of electronics and IoT (Internet of Things). It is developed by Espressif Systems. Here's a brief overview:

1. Microcontroller and SoC: The ESP32 integrates a dual-core processor, which makes it more powerful than its predecessor, the ESP8266. It also includes a variety of peripherals, such as GPIO (General Purpose Input/Output) pins, UART (Universal Asynchronous Receiver-Transmitter) interfaces, I2C (Inter-Integrated Circuit), SPI (Serial Peripheral Interface), and more.
2. Wireless Connectivity: One of the key features of the ESP32 is its built-in Wi-Fi and Bluetooth capabilities. This makes it suitable for a wide range of applications where wireless communication is essential, including IoT devices, home automation, and sensor networks.
3. Low Power Consumption: The ESP32 is designed to be power-efficient, supporting various low-power modes, which is crucial for battery-operated and energy-conscious applications.
4. Development Environment: It can be programmed using the Arduino IDE, MicroPython, or the EspressifIoT Development Framework (ESP-IDF), providing flexibility for developers with different programming preferences.
5. Rich Set of Libraries and Tools: The ESP32 ecosystem includes a rich set of libraries and tools that simplify the development process. This includes libraries for handling Wi-Fi and Bluetooth communication, as well as libraries for interfacing with various sensors and peripherals.
6. Dual-Core Architecture: The dual-core architecture of the ESP32 allows for more efficient multitasking and parallel processing in applications.
7. Affordability and Popularity: The ESP32 is relatively affordable, making it a popular choice for hobbyists, makers, and professionals alike. Its popularity is also attributed to its feature-rich nature and the extensive community support.

Overall, the ESP32 has become a go-to choice for many IoT projects due to its powerful features, wireless connectivity, and ease of use in the development process.

**RELAY:-**



A relay is an electrical switch that uses an electromagnet to control the opening or closing of a circuit. It is a crucial component in various electronic and electrical systems, providing a way to control high-power devices with a low-power signal. Here's a brief explanation of how relays work and their common applications:

1. Operating Principle: A relay consists of an electromagnet, a set of contacts (switches), and a spring. When an electric current flows through the coil of the electromagnet, it generates a magnetic field that attracts a movable armature or lever connected to the switch contacts. This movement either opens or closes the contacts, depending on the relay type.
2. Types of Relays:

* Normally Open (NO) Relay: In its resting state, the relay contacts are open. When the electromagnet is energized, the contacts close.
* Normally Closed (NC) Relay: In its resting state, the relay contacts are closed. When the electromagnet is energized, the contacts open.
* Changeover (CO) or Single-Pole Double-Throw (SPDT) Relay: This type has one common terminal, one normally open contact, and one normally closed contact. It can be used to switch between two different circuits.

1. Applications:

Switching High-Power Devices: Relays are commonly used to control high-power devices, such as motors, lights, or heaters, with a low-power signal from a microcontroller or a sensor.

1. Remote Control Systems: Relays are essential in remote control systems, where a low-power signal from a remote device can control a device located at a distance.
2. Automotive Applications: In automobiles, relays are used for various functions like controlling headlights, fans, and other electrical systems.
3. Industrial Automation: Relays play a vital role in industrial automation, where they are used to control large machinery and equipment.
4. Advantages:

* Relays provide electrical isolation between the control circuit and the load circuit, enhancing safety.
* They allow the use of low-power control signals to switch high-power loads.

1. Drawbacks:

* Mechanical relays have moving parts, and over time, they can wear out.
* Mechanical relays may produce audible clicks when switching, which may be undesirable in certain applications.

In summary, relays are versatile components that serve as electrically controlled switches, finding applications in a wide range of electronic, industrial, and automotive systems. They are crucial for interfacing low-power control circuits with high-power devices.

**WIRE:-**



In the context of electronics and electrical systems, "wire" generally refers to a conductor used to carry electrical current. Here's a brief explanation:

1. Conductive Material: Wires are typically made of conductive materials, most commonly copper or aluminum. These materials have low resistance, allowing for the efficient flow of electric current.
2. Insulation: To prevent short circuits and to ensure the safe transmission of electrical power, wires are often insulated. The insulation material can be made of various substances, such as rubber, plastic, or fiberglass. The insulation serves to contain the electrical current within the wire and protect it from external influences.
3. Stranding: Wires can be solid or stranded. Solid wires consist of a single, solid conductor, while stranded wires are made up of multiple smaller strands twisted or braided together. Stranded wires are more flexible and are often preferred in applications where flexibility is important.
4. Gauge or Thickness: Wires come in different thicknesses, usually specified by their gauge. The gauge of a wire determines its diameter, and thicker wires generally have lower resistance. The choice of wire gauge depends on the current-carrying capacity required for a particular application.
5. Color Coding: In many electrical systems, wires are color-coded for easy identification. Different colors may signify different purposes or voltages, helping electricians and technicians during installation, maintenance, and troubleshooting.
6. Applications: Wires are fundamental components in electrical and electronic circuits. They are used for power transmission, signal transmission, and interconnecting various components within a system. Wires are found in everything from household wiring to the internal connections of electronic devices.
7. Types of Wire:

* Hookup Wire: Used for general-purpose wiring and low-current applications.
* Power Cables: Thicker wires designed for transmitting higher currents in power distribution systems.
* Coaxial Cable: Consists of a central conductor surrounded by insulation, a metallic shield, and an outer insulating layer. Commonly used for transmitting signals in telecommunications and cable television.

In summary, wires are essential components in electrical and electronic systems, serving as conduits for the transmission of electric current. They come in various types and sizes, each suited to specific applications based on factors like current requirements, flexibility, and insulation properties.

**SWITCH BOARD:-**



It seems like there might be a small confusion in your question, as "switchboard" and "bbox" are typically distinct terms in the context of electrical installations.

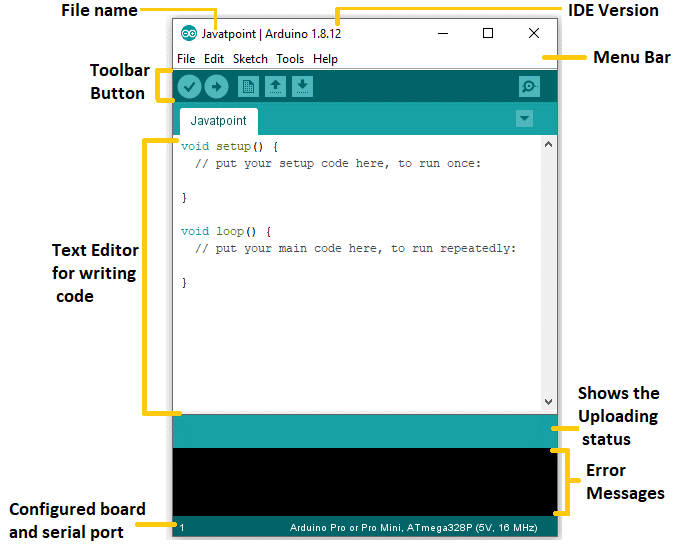
1. Switchboard (Switch Box): A switchboard, also known as a switch box or electrical panel, is an enclosure that houses electrical switches, circuit breakers, and other devices for controlling and protecting electrical circuits. It is a crucial component of the electrical distribution system in buildings. The switchboard is responsible for distributing electrical power from the main source to various circuits within the building.
2. Functions: It allows for the control and distribution of electricity, providing a centralized location for switches that control different circuits. Circuit breakers within the switchboard protect against overcurrent and short circuits.
3. Bbox (Junction Box): "Bbox" could potentially refer to a junction box. A junction box is an enclosure used to contain wiring connections, providing a safe and organized space for splicing wires and protecting the connections from environmental factors.
4. Functions: Junction boxes are used to facilitate the connection of electrical wires, such as those from different cables or conduits. They help protect the connections from damage, prevent exposure to live wires, and contain any potential fire hazards.

If you are referring to a specific product or term with "Bbox," please provide more context or clarification. The switchboard and junction box serve different purposes in an electrical system, with the switchboard managing the distribution of electrical power and the junction box facilitating safe and organized wire connections.

**ARDUINO IDE**

The Arduino IDE is an open-source software, which is used to write and upload code to the Arduino boards. The IDE application is suitable for different operating systems such as **Windows, Mac OS X, and Linux**. It supports the programming languages C and C++. Here, IDE stands for **Integrated Development Environment**.

The program or code written in the Arduino IDE is often called as sketching. We need to connect the Arduino board with the IDE to upload the sketch written in the Arduino IDE software. The sketch is saved with the extension '.ino.'

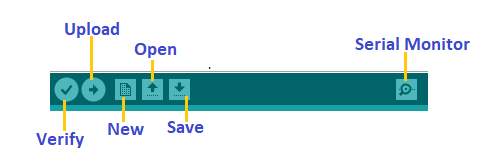
The Arduino IDE will appear as:

**Let's discuss each section of the Arduino IDE display in detail: -**

## **Toolbar Button**

The icons displayed on the toolbar are **New, Open, Save, Upload,** and **Verify**.

It is shown below:



### Upload

The Upload button compiles and runs our code written on the screen. It further uploads the code to the connected board. Before uploading the sketch, we need to make sure that the correct board and ports are selected.

We also need a USB connection to connect the board and the computer. Once all the above measures are done, click on the Upload button present on the toolbar.

The latest Arduino boards can be reset automatically before beginning with Upload. In the older boards, we need to press the Reset button present on it. As soon as the uploading is done successfully, we can notice the blink of the Tx and Rx LED.

If the uploading is failed, it will display the message in the error window.

We do not require any additional hardware to upload our sketch using the Arduino Bootloader. A **Bootloader** is defined as a small program, which is loaded in the microcontroller present on the board. The LED will blink on PIN 13.

### Open

### The Open button is used to open the already created file. The selected file will be opened in the current window.

### Save

The save button is used to save the current sketch or code.

### New

It is used to create a new sketch or opens a new window.

### Verify

The Verify button is used to check the compilation error of the sketch or the written code.

### Serial Monitor

The serial monitor button is present on the right corner of the toolbar. It opens the serial monitor.

It is shown below:

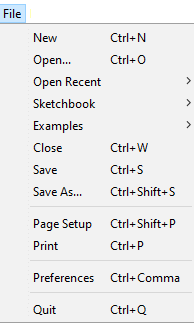
Description: Arduino IDE

When we connect the serial monitor, the board will reset on the operating system Windows, Linux, and Mac OS X. If we want to process the control characters in our sketch, we need to use an external terminal program. The terminal program should be connected to the COM port, which will be assigned when we connect the board to the computer.

## **Menu Bar**

* **File**

When we click on the File button on the Menu bar, a drop-down list will appear. It is shown below:



Let's discuss some option in detail.

**New**

The New button opens the new window. It does not remove the sketch which is already present.

**Open**

It allows opening the sketch, which can be browsed from the folders and computer drivers.

**Open Recent**

The Open Recent button contains the list of the recent sketches.

**Sketchbook**

It stores the current sketches created in the Arduino IDE software. It opens the selected sketch or code in a new editor at an instance.

**Examples**

It shows the different examples of small projects for a better understanding of the IDE and the board. The IDE provides examples of self-practice.

**Close**

The Close button closes the window from which the button is clicked.

**Save**

The save button is used to save the current sketch. It also saves the changes made to the current sketch. If we have not specified the name of the file, it will open the '**Save As...'** window.

**Save As...**

We can save the sketch with a different name using the '**Save As...'** button. We can also change the name accordingly.

**Undo**

The Undo button is used to reverse the last modification done to the sketch while editing.

**Redo**

The Redo button is used to repeat the last modification done to the sketch while editing.

**Cut**

It allows us to remove the selected text from the written code. The text is further placed to the clipboard. We can also paste that text anywhere in our sketch.

**Copy**

It creates a duplicate copy of the selected text. The text is further placed on the clipboard.

**Paste**

The Paste button is used to paste the selected text of the clipboard to the specified position of the cursor.

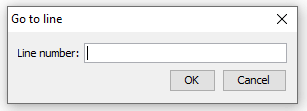
**Select All**

It selects all the text of the sketch.

**Go to line...**

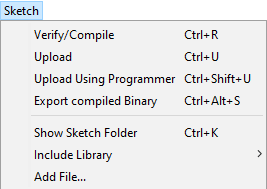
It moves the cursor to the specified line number.

The window will appear as:



* **Sketch**

When we click on the Sketch button on the Menu bar, a drop-down list appears. It is shown below:



Let's discuss some option in detail.

**Verify/Compile**

It will check for the errors in the code while compiling. The memory in the console area is also reported by the IDE.

**Upload**

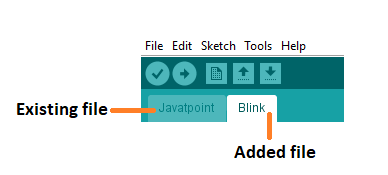
The Upload button is used to configure the code to the specified board through the port.

**Include Library**

Include Library includes various Arduino libraries. The libraries are inserted into our code at the beginning of the code starting with the #. We can also import the libraries from .zip file.

**Add File...**

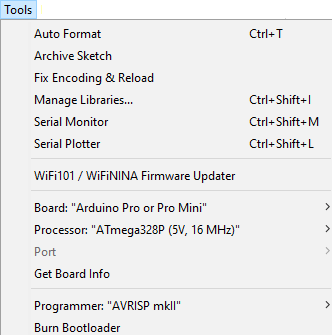
The Add File... button is used to add the created file in a new tab on the existing file.



We can also delete the corresponding file from the tab by clicking on the **small triangle** -> **Delete** option.

### Tools

When we click on the Tools button on the Menu bar, a drop-down list appears. It is shown below:



Let's discuss some option in detail.

**Manage Libraries...**

It shows the updated list of all the installed libraries. We can also use this option to install a new library into the Arduino IDE.

**Board**

We are required to select the board from the list of boards. The selected board must be similar to the board connected to the computer.

**Processor**

It displays the processor according to the selected board. It refreshes every time during the selection of the board.

**Port**

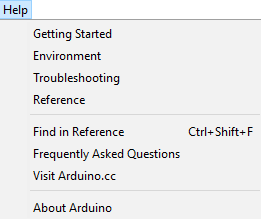
It consists of the virtual and real serial devices present on our machine.

**Get Board Info**

It gives the information about the selected board. We need to select the appropriate port before getting information about the board.

### Help

When we click on the Help button on the Menu bar, a drop-down list will appear. It is shown below:



The Help section includes several documents that are easy to access, which comes along with the Arduino IDE. It consists of the number of options such as Getting Started, Environment, Troubleshooting, Reference, etc. We can also consider the image shown above, which includes all the options under the Help section.

Some documents like Getting started, Reference, etc., can be accessed without the internet connection as well. It will directly link us to the official website of Arduino.

**CODING:-**

// LoadWi-Fi library

#include <WiFi.h>

// Replace with your network credentials

const char\* ssid = "Samsung";

const char\* password = "123456789";

// Set web server port number to 80

WiFiServerserver(80);

// Variable to store the HTTP request

String header;

// Auxiliar variables to store the current output state

String output26State = "off";

String output27State = "off";

// Assign output variables to GPIO pins

constint output26 = 26;

constint output27 = 27;

void setup() {

Serial.begin(115200);

// Initialize the output variables as outputs

pinMode(output26, OUTPUT);

pinMode(output27, OUTPUT);

// Set outputs to LOW

digitalWrite(output26, LOW);

digitalWrite(output27, LOW);

// Connect to Wi-Fi network with SSID and password

Serial.print("Connecting to ");

Serial.println(ssid);

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

// Print local IP address and start web server

Serial.println("");

Serial.println("WiFi connected.");

Serial.println("IP address: ");

Serial.println(WiFi.localIP());

server.begin();

}

void loop(){

WiFiClient client = server.available(); // Listen for incoming clients

if (client) { // If a new client connects,

Serial.println("New Client."); // print a message out in the serial port

String currentLine = ""; // make a String to hold incoming data from the client

while (client.connected()) { // loop while the client's connected

if (client.available()) { // if there's bytes to read from the client,

char c = client.read(); // read a byte, then

Serial.write(c); // print it out the serial monitor

header += c;

if (c == '\n') { // if the byte is a newline character

// if the current line is blank, you got two newline characters in a row.

// that's the end of the client HTTP request, so send a response:

if (currentLine.length() == 0) {

// HTTP headers always start with a response code (e.g. HTTP/1.1 200 OK)

// and a content-type so the client knows what's coming, then a blank line:

client.println("HTTP/1.1 200 OK");

client.println("Content-type:text/html");

client.println("Connection: close");

client.println();

// turns the GPIOs on and off

if (header.indexOf("GET /26/on") >= 0) {

Serial.println("GPIO 26 on");

output26State = "on";

digitalWrite(output26, HIGH);

} else if (header.indexOf("GET /26/off") >= 0) {

Serial.println("GPIO 26 off");

output26State = "off";

digitalWrite(output26, LOW);

} else if (header.indexOf("GET /27/on") >= 0) {

Serial.println("GPIO 27 on");

output27State = "on";

digitalWrite(output27, HIGH);

} else if (header.indexOf("GET /27/off") >= 0) {

Serial.println("GPIO 27 off");

output27State = "off";

digitalWrite(output27, LOW);

}

// Display the HTML web page

client.println("<!DOCTYPE html><html>");

client.println("<head><meta name=\"viewport\" content=\"width=device-width, initial-scale=1\">");

client.println("<link rel=\"icon\" href=\"data:,\">");

// CSS to style the on/off buttons

// Feel free to change the background-color and font-size attributes to fit your preferences

client.println("<style>html { font-family: Helvetica; display: inline-block; margin: 0px auto; text-align: center;}");

client.println(".button { background-color: #4CAF50; border: none; color: white; padding: 16px 40px;");

client.println("text-decoration: none; font-size: 30px; margin: 2px; cursor: pointer;}");

client.println(".button2 {background-color: #555555;}</style></head>");

// Web Page Heading

client.println("<body><h1>ESP32 Web Server</h1>");

// Display current state, and ON/OFF buttons for GPIO 26

client.println("<p>GPIO 26 - State " + output26State + "</p>");

// If the output26State is off, it displays the ON button

if (output26State=="off") {

client.println("<p><a href=\"/26/on\"><button class=\"button\">ON</button></a></p>");

} else {

client.println("<p><a href=\"/26/off\"><button class=\"button button2\">OFF</button></a></p>");

}

// Display current state, and ON/OFF buttons for GPIO 27

client.println("<p>GPIO 27 - State " + output27State + "</p>");

// If the output27State is off, it displays the ON button

if (output27State=="off") {

client.println("<p><a href=\"/27/on\"><button class=\"button\">ON</button></a></p>");

} else {

client.println("<p><a href=\"/27/off\"><button class=\"button button2\">OFF</button></a></p>");

}

client.println("</body></html>");

// The HTTP response ends with another blank line

client.println();

// Break out of the while loop

break;

} else { // if you got a newline, then clear currentLine

currentLine = "";

}

} else if (c != '\r') { // if you got anything else but a carriage return character,

currentLine += c; // add it to the end of the currentLine

}

}

}

// Clear the header variable

header = "";

// Close the connection

client.stop();

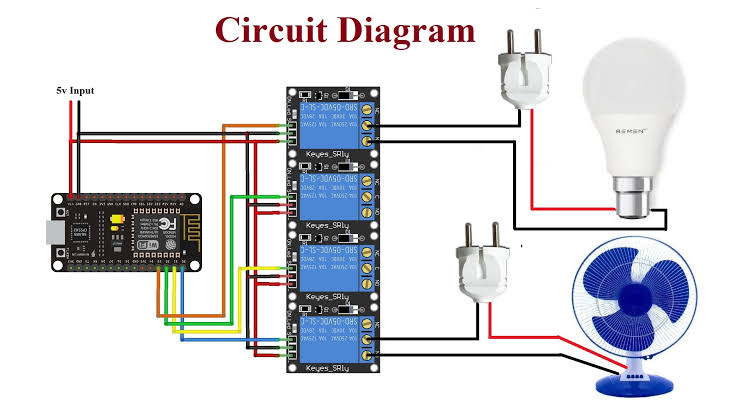
Serial.println("Client disconnected.");

Serial.println("");

}

}

**CIRCUIT DIAGRAM**



**TESTING**

In our project still we have not found any error, it works properly when a person whiff in the sensor.

**Alpha Testing**-In this phase, the testing was done by our group that is we have tested each and every part of the project.

**Beta Testing**-In this phase, the testing was being done by our group of faculty member of our institution.

**Acceptance Testing**-In this phase the testing was done by the principle of our institution and also invigilator who have visited for seminar of our project.

**FURTHER ENHANCEMENT**

**Fan Implementation:** To improve the system's performance and efficiency, we have implemented a fan mechanism. This addition ensures better air circulation and heat dissipation, leading to optimal functioning of the system components.

**Application Development:** To complement the hardware enhancements, we have developed a dedicated application that facilitates efficient system management and monitoring.

**Increased Control Area:**We expanded the control area of the system to provide better coverage and enhanced functionality.

**CONCLUSION**

* The implementation of the smart switch board in the head of department (HOD) room has proven to be a valuable enhancement for managing electrical components efficiently and effectively.

**REFERENCE**

1. [**https://randomnerdtutorials.com/esp32-relay-module-ac-web-server/**](https://randomnerdtutorials.com/esp32-relay-module-ac-web-server/)
2. [**https://www.google.com/amp/s/www.instructables.com/ESP32-WiFi-Bluetooth-Home-Automation-With-Manual-S/%3famp\_page=true**](https://www.google.com/amp/s/www.instructables.com/ESP32-WiFi-Bluetooth-Home-Automation-With-Manual-S/%3famp_page=true)

THANK YOU