Bluetooth Control Home Automation Using Arduino-Uno

Govekar Navnath Sopan[1], Dhongade Omkar[2], Kirte Shubham[3], Vaishnavi Gandhale[4], Poonam Kamble[5]
[1]*Assist. Professor, Department of Electrical Engineering, Navsahyadri Group of institutes, Pune, Maharashtra, India*
[2,3,4,5]*Students, Department of Electrical Engineering, Navsahyadri Group of institutes, Pune, Maharashtra, India*

***Abstract*—In today's fast-paced world, the idea of turning your regular home into a smart home is no longer a luxury, it's a practical way to save time, energy, and effort. This project focuses on building a simple and low-cost home automation system using an Arduino UNO and Bluetooth technology, allowing users to control appliances like lights and fans directly from their smartphones.**

The system works with the Arduino UNO acting as the brain, connected to an HC-05 Bluetooth module that receives commands from a mobile app. When a user sends a command from their phone, Arduino processes it and activates the desired appliance through relay modules. Two 4-channel relay boards are used—one handles turning on/off appliances like two lights and a fan, while the other connects to a standard fan regulator. This clever setup lets users control fan speed wirelessly, by switching different resistance levels just like a manual dial. To make the system more interactive and smarter, a DHT11 sensor is added to measure room temperature. The readings are shown on a 16x2 LCD screen, so users can always check the current temperature. For safety, an MQ2 smoke sensor is included; if smoke is detected, a buzzer sounds immediately to alert everyone. Another smart feature is the use of a PIR motion sensor. If no movement is detected in the room, the system automatically switches off all appliances to avoid wasting energy—a simple step that helps save electricity. This project shows how everyday living can be improved with basic electronics. With easy-to-find components and simple coding, it's a great step toward building an affordable, efficient, and safer smart home. The popular concept of a home automation is of a machine that looks and works like a human being. The industry is moving from current state of automation to Robotization, to increase productivity and to deliver uniform quality. The industrial robots of today may not look the least bit like a human being although all the research is directed to provide more and more anthropomorphic and humanlike features and super-human capabilities in these. One type of robot commonly used in industry is a robotic manipulator or simply a robotic arm. It is an open or closed kinematic chain of rigid links interconnected by movable joints. In some configurations, links can be considered to correspond to human anatomy as waist, upper arm and forearm with joint at shoulder and elbow. At end of arm a wrist joint connects an end effect which may be a tool and its fixture or a gripper or any other device to work.

***Index Terms*—Smart Home Automation.**

I. INTRODUCTION

As technology becomes a bigger part of our everyday lives, more people want their homes to be smarter and easier to manage. Home automation systems help make life more comfortable, safe, and energy-efficient by letting you control things like lights and fans all from one place. This project is about creating a simple and affordable home automation system using an Arduino UNO microcontroller and Bluetooth, so anyone can control their home appliances right from their smartphone. Most homes still use manual switches to turn things on and off, which can be a hassle sometimes. Also, safety devices like smoke detectors usually work on their own and don't connect with other home systems. This project brings everything together into one easy-to-use system that lets you control appliances wirelessly while also keeping an eye on safety and the environment. The system uses relay modules to switch appliances like lights and fans on or off. Instead of buying expensive fan speed controllers, it uses a clever setup with a regular fan regulator and relays to adjust fan speed, making it simple and budget-friendly. A temperature sensor shows the room's current temperature on a small display, so you always know how warm it is.

For safety, there's a smoke sensor that sounds an alarm if it detects smoke. And with Bluetooth, you can easily control everything from your phone no need to get up to flip switches. To save energy, a motion sensor turns

off appliances when no one's in the room. Plus, a motorized window opener lets you open and close windows remotely, adding even more convenience. In short, this project is all about making your home smarter, safer, and more energy-efficient using technology that's easy to build and use

II. LITERATURE SURVEY

Home automation is becoming a big part of modern living, helping people control their homes more easily, save energy, and stay safer. Many researchers have worked on different ways to make homes smarter and more connected, and their ideas help shape projects like this one. For example, Lee and Kim (2018) created a Bluetooth-based system to control home devices wirelessly while using very little power. This is similar to our project, which also uses Bluetooth to let users control appliances easily from their phones. Ali and Hussain (2021) worked on smart fan speed control using fuzzy logic and environmental sensors. Instead of complex electronics, our project uses relays and a regular fan regulator to adjust fan speed, which is a simple and affordable way to achieve the same goal of comfort and energy savings.

Safety is important too. Brown and Davis (2019) designed a low-cost smoke detection and alert system, much like our system's smoke sensor that sounds an alarm automatically if there's danger. Energy saving is another key focus. Garcia and Rodriguez (2022) used motion sensors to detect when a room is empty and turn off devices to save power. We use a similar motion sensor setup to switch off lights and fans when no one's around. Other researchers, like Wang et al. (2017), have built affordable home automation systems using the Internet of Things. as Implementation using Arduino and Bluetooth can have better impact for short wireless control.

III. COMPONENTS OF HOME AUTOMATION

3.1 Structure

The structure of the home automation mainly consists the basic microcontroller for controlling the relay connected to the output devices that mainly consists of fan, motor-controlled window and all the devices that can be controlled using microcontrollers and relays. Bluetooth Module Is implemented for short range wireless control.

At present mostly (lead-acid) batteries are used, but potential power sources could be:

- Microcontroller Arduino-Uno R3
- Bluetooth module
- Relay module
- Software Code
- PIR motion sensor
- L298N motor driver module
- MQ2 Smoke Detector
- Gear Motor and Buzzer

3.3 Control

"Relays are like the 'hands' of a smart home, the parts which convert signals into action. By far the most popular relays are those that switch appliances on or off, and motorized systems that control home devices. But there are some recent advances in alternative types of relays, powered by Bluetooth, sensors, or wireless modules."

3.4 Touch

Imagine walking into your home, and the lights automatically adjust to your preferred brightness, the fan switches on to just the right speed, and your security system acknowledges your presence—all without you lifting a finger. Thanks to Bluetooth-enabled automation with Arduino Uno, this level of smart control is now a reality.

At the heart of the system is the Arduino Uno, acting as the brain that connects wirelessly with various sensors and devices. These smart modules—ranging from temperature and motion sensors to automated relays—work together to create a living space that responds to your needs. When you send a command from your smartphone, Bluetooth transmits the signal to the Arduino, which immediately processes it and activates the corresponding appliance or system.

Beyond simple control, real-time feedback keeps the home environment adaptive. Lights can dim as natural sunlight increases, fans adjust speed based on room occupancy, and doors lock automatically when no movement is detected. With every interaction, the system learns and refines its responses, making daily life more effortless and efficient.

This version feels more engaging and relatable while keeping the technical essence intact.

3.5 Vision

The ultimate vision is to make the reliable, efficient and safe, secure and controlled environment of home.

1. Seamless Control & Accessibility
- Enable users to control home appliances wirelessly via a smartphone.
- Provide instant response without requiring an internet connection.
- Ensure easy installation with minimal hardware complexity.

2. Energy Efficiency & Automation
- Optimize power consumption using motion sensors and light-dependent resistors (LDRs).
- Automate appliances based on environmental conditions, reducing waste.
- Implement smart scheduling for devices to enhance efficiency.

3. Security & Safety Enhancements
- Introduce Bluetooth-based smart locks for secure access control.
- Integrate smoke and gas sensors for real-time hazard detection.
- Provide instant alerts for unauthorized access or emergencies.

4. Scalability & Future Expansion
- Expand the system with Wi-Fi or IoT integration for remote control.
- Introduce voice assistants for hands-free operation.
- Implement machine learning to optimize appliance usage based on user habits.

The ultimate goal is to create a cost-effective, user-friendly, and intelligent home

IV. CONSTRUCTION OF HOME AUTOMATION SYSTEM

1. Arduino Uno R3:
- Microcontroller: ATmega328P
- Operating Voltage: 5V
- Input Voltage (recommended): 7-12V
- Digital I/O Pins: 14 (of which 6 provide PWM output)
- Analog Input Pins: 6
- Flash Memory: 32 KB (0.5 KB used by bootloader)
- SRAM: 2 KB
- EEPROM: 1 KB
- Clock Speed: 16 MHz
- USB Interface: Used for programming and power

2. 4-Channel Relay Module:
- Number of Channels: 4
- Relay Type: Electromagnetic
- Coil Voltage: Typically, 5V
- Switching Voltage (AC): Up to 250V AC
- Switching Current (AC): Up to 10A
- Switching Voltage (DC): Up to 30V DC
- Switching Current (DC): Up to 10A
- Interface: Control pins, common ground, screw terminals for load.

3. 16x2 I2C LCD Display:
- Display Format: 16 characters x 2 rows
- Interface: I2C
- Operating Voltage: 5V
- Controller IC: HD44780 compatible

4. DHT11 Temperature and Humidity Sensor:
- Temperature Range: 0°C to 50°C (±2°C accuracy)
- Humidity Range: 20% to 90% RH (±5% RH accuracy)
- Operating Voltage: 3.3V to 5V
- Output: Digital signal

5. MQ2 Smoke Detector Sensor:
- Sensing Gas: Smoke, LPG, Butane, Propane, Methane, Alcohol, Hydrogen
- Operating Voltage: 5V
- Output: Analog voltage (and often digital)

6. HC-05 Bluetooth Module:
- Bluetooth Version: 2.0 + EDR
- Operating Voltage: 3.3V (logic level shifting needed for 5V Arduino TX)
- Communication Interface: Serial (UART)

7. PIR Motion Sensor (HC-SR501):
- Operating Voltage: 4.5V to 20V (typically 5V)
- Output: Digital HIGH/LOW

8. L298N Motor Driver Module:
- Motor Supply Voltage: 5V to 35V DC
- Output Current (per channel): Up to 2A (peak)
- Logic Supply Voltage: 5V

9. 12V DC Gear Motor:
- Operating Voltage: 12V DC
- Speed (RPM): 2700 rpm
- Torque: (kg-cm)5

10. Buzzer:
- Operating Voltage: 5V
- Type: Electromagnetic or Piezoelectric

V. WORKING

1. Appliance Control (Lights and Fan On/Off):

- Hardware: Two four-channel relay modules and the Arduino Uno.
- Working Principle: Relays are electromechanical switches that allow a low-power circuit (controlled by the Arduino) to switch a high-power circuit (the AC mains powering the lights and fan). Each channel on the relay module corresponds to one appliance.
- Detailed Operation:
- The Arduino Uno sends digital signals to the control pins of the relay module.
- When a specific pin on the relay module receives a HIGH signal from the Arduino, the corresponding relay coil is energized.
- This energizing action causes the relay's internal switch to either close (if it was normally open) or open (if it was normally closed), completing or breaking the AC circuit to the respective light or the main power line feeding the fan regulator.
- Sending a LOW signal to the relay control pin de-energizes the coil, and the switch returns to its default state, turning the appliance off.
- The Arduino code will contain logic to map specific commands received (via Bluetooth) to the activation of the appropriate relay channels. For example, receiving "LIGHT1 ON" might trigger the Arduino to send a HIGH signal to the control pin of the relay connected to the first light.

2. Fan Speed Control:

- Hardware: Another four-channel relay module and a standard AC fan regulator.
- Working Principle: Standard AC fan regulators typically use a series of internal resistances or a triac-based circuit to reduce the voltage supplied to the fan motor, thereby controlling its speed. This project cleverly utilizes relays to switch between different fixed resistance levels within the regulator.
- Detailed Operation:
- The AC fan regulator is wired such that different speed settings correspond to different combinations of internal resistances being active.
- The input of the fan regulator is connected to the AC mains. The output of the regulator goes to the fan.
- The four channels of the relay module are connected in parallel to different points within the regulator's resistance network (or to bypass certain resistive elements).
- The Arduino sends specific digital signal patterns to the control pins of this relay module.
- By activating different combinations of relays, the Arduino effectively selects different effective resistances in the circuit, mimicking the action of manually turning the regulator knob to different speed settings.
- For example, one relay might bypass all extra resistance (highest speed), while other combinations introduce increasing levels of resistance (lower speeds). The Arduino code will need to be carefully calibrated to determine which relay combinations correspond to the desired fan speeds.

3. Temperature Monitoring:

- Hardware: DHT11 temperature and humidity sensor and a 16x2 I2C LCD display.
- Working Principle: The DHT11 sensor is a digital sensor that outputs temperature and humidity readings as digital signals. The I2C LCD communicates with the Arduino using the I2C serial communication protocol, requiring only two data lines (SDA and SCL) plus power and ground.
- Detailed Operation:
- The DHT11 sensor is connected to specific digital input/output pins on the Arduino.
- The Arduino code includes a library to communicate with the DHT11 sensor. This code periodically reads the temperature data from the sensor.
- The I2C LCD is connected to the Arduino's I2C pins (typically A4 for SDA and A5 for SCL on the Uno).
- Another library is used to control the LCD via the I2C interface.
- The Arduino code formats the temperature reading and sends it to the LCD module, which then displays it on its 16x2 character display.

4. Smoke Detection:

- Hardware: MQ2 smoke detector and a buzzer.
- Working Principle: The MQ2 sensor contains a heating element and a sensing layer that reacts to the presence of combustible gases, including smoke. Its resistance changes proportionally to the

gas concentration. This change in resistance is typically converted to a voltage output that can be read by the Arduino. The buzzer is an electromechanical transducer that produces an audible sound when a voltage is applied across its terminals.

- Detailed Operation:
- The MQ2 sensor is connected to an analog input pin on the Arduino (for reading the voltage output) and also requires power and ground.
- The Arduino code continuously reads the analog voltage from the MQ2 sensor.
- A predefined threshold value is set in the code. If the sensor reading exceeds this threshold (indicating a significant concentration of smoke), the Arduino triggers the buzzer.
- The buzzer is connected to a digital output pin on the Arduino. When the Arduino sets this pin to HIGH, current flows through the buzzer, producing sound. Setting the pin to LOW silences the buzzer.

5. Bluetooth Control:

- Hardware: HC-05 Bluetooth module.
- Working Principle: The HC-05 module allows serial communication over Bluetooth. It can be configured to act as either a master or a slave device. In this project, it will likely be configured as a slave, waiting for connections from a master device (like a smartphone running a Bluetooth terminal app).
- Detailed Operation:
- The HC-05 module is connected to the Arduino's serial communication pins (RX and TX). It's crucial to use a voltage divider on the Arduino's TX pin going to the HC-05's RX pin as the Arduino operates at 5V and the HC-05 at 3.3V.
- The Arduino code initializes the serial communication with the HC-05 at a specific baud rate.
- When a user sends commands from the Bluetooth terminal app on their phone, these commands are transmitted wirelessly to the HC-05.
- The HC-05 receives these commands and forwards them to the Arduino via the serial connection.
- The Arduino code parses these received commands. Based on the command (e.g., "LIGHT1 ON", "FAN SPEED 2"), the Arduino executes the corresponding actions, such as controlling the appropriate relays.
- The Arduino can also send data back to the Bluetooth terminal app (e.g., confirmation messages or sensor readings) by writing data to the serial port connected to the HC-05.

6. Motion-Based Automatic Appliance Off:

- Hardware: PIR (Passive Infrared) motion sensor.
- Working Principle: PIR sensors detect changes in infrared radiation emitted by moving objects (like humans). When motion is detected, the sensor outputs a HIGH signal for a given specific duration.
- Detailed Operation:
- The PIR sensor is connected to a digital input pin on the Arduino. It also requires power and ground.
- The Arduino code continuously monitors the digital input pin connected to the PIR sensor.
- When motion is detected, the pin goes HIGH. The Arduino code typically starts a timer when motion is last detected.
- If no motion is detected for a predefined period (e.g., 5 minutes), the Arduino code will automatically send signals to the relay modules to turn off the lights and the fan.
- If motion is detected again before the timer expires, the timer is reset, and the appliances remain in their current state or can be programmed to turn back on if they were previously off due to no motion.

7. Automated Window Control:

- Hardware: 12V DC gear motor and L298N motor driver.
- Working Principle: The L298N is a dual H-bridge motor driver IC that allows the Arduino to control the speed and direction of two DC motors. By controlling the voltage and polarity applied to the motor terminals, the motor can be made to rotate clockwise or counterclockwise at varying speeds. A gear motor reduces the speed of the motor while increasing its torque, making it suitable for moving the window.
- Detailed Operation:
- The 12V DC gear motor is connected to the output terminals of the L298N motor driver. The motor driver requires a separate 12V power supply for the motor.

- The control pins of the L298N (typically enable pins and input pins for each motor channel) are connected to digital output pins on the Arduino.
- To open the window, the Arduino code sends specific digital signals to the L298N's control pins, causing the motor to rotate in one direction for a 6 sec as calculated the length and gear ratio of motor or until a limit switch (if implemented, though not mentioned) is triggered.
- To close the window, the Arduino sends a different set of signals to the L298N, reversing the motor's direction for a same duration of previous rotation or until another limit switch is triggered.
- The Arduino code will likely be triggered by commands received via Bluetooth to initiate the window opening or closing sequence.

VI. METHODOLOGY

Phase 1: Planning and Design

1. Requirements Specification:

- Detailed Breakdown of Features: Clearly define all the functionalities of the system, including individual light control (on/off), fan speed control (multiple levels), temperature monitoring (display on LCD), smoke detection (with buzzer activation), Bluetooth-based remote control (all appliances), occupancy-based automatic appliance shutdown, and automated window control (open/close).
- User Interface Design: Plan the commands to be used via the Bluetooth terminal app for controlling each appliance and the window. Determine the information to be displayed on the LCD.
- Hardware Selection Finalization: Confirm the specific models of Arduino Uno, relay modules (number of channels and voltage ratings), DHT11, MQ2, HC-05, PIR sensor, 16x2 I2C LCD, L298N motor driver, 12V DC gear motor, buzzer, and any necessary supporting components (resistors, wires, breadboards, power supplies).
- Circuit Design and Schematic Diagram: Develop a detailed circuit diagram illustrating the connections between the Arduino, relay modules, sensors, LCD, Bluetooth module, motor driver, and other components. Pay close attention to power requirements and signal levels to ensure compatibility and safety.

2. Component Testing and Setup:

- Individual Module Testing: Before integration, test each individual component (relay, sensor, LCD, Bluetooth, motor driver, motor) with the Arduino to ensure they are functioning correctly. Write basic Arduino sketches to verify the input/output of each module. For example, test the relay by toggling it with a simple digital output, read values from the DHT11 and MQ2 and display them on the serial monitor, establish Bluetooth communication, and control the motor direction using the L298N.
- Library Installation: Identify and install the necessary Arduino libraries for the I2C LCD, DHT11 sensor, and potentially for simplified Bluetooth communication if needed.

Phase 2: Hardware Implementation and Wiring

1. Base Platform Assembly: Mount the Arduino Uno on a suitable prototyping platform (e.g., breadboard)
2. Power Distribution: Establish a stable and appropriate power supply for the Arduino and other components. Ensure proper voltage levels and current ratings for each module. Pay particular attention to isolating the low-voltage control circuitry from the high-voltage AC mains used for appliance control.

Sensor Integration: Connect the DHT11 temperature sensor and MQ2 smoke detector to the appropriate digital and analog pins of the Arduino, respectively, following the circuit diagram.

3. LCD Integration: Connect the 16x2 I2C LCD to the Arduino's I2C pins (SDA and SCL) and provide power and ground.
4. Relay Module Wiring:

- Appliance Control Relays: Wire the two four-channel relay modules to the AC power lines of the two lights and the input of the fan regulator. Ensure that the relays are switching the live (hot) wire and that proper safety precautions are taken when working with mains voltage. Use appropriate wiring and insulation.
- Fan Speed Control Relays: Wire the second four-channel relay module to the internal circuitry (resistances) of the AC fan regulator according to the planned speed control mechanism. This step might require some experimentation or reverse-

engineering of the specific fan regulator being used.

5. Bluetooth Module Integration: Connect the HC-05 Bluetooth module to the Arduino's serial communication pins (RX and TX), remembering to use a voltage divider on the Arduino's TX pin. Provide power and ground to the Bluetooth module.
6. PIR Sensor Integration: Connect the PIR motion sensor to a digital input pin on the Arduino and provide power and ground.
7. Motor Driver and Motor Wiring: Connect the 12V DC gear motor to the output terminals of the L298N motor driver. Connect the control pins (enable and input pins) of the L298N to digital output pins on the Arduino. Provide a separate 12V power supply to the motor driver.
8. Buzzer Integration: Connect the buzzer to a digital output pin on the Arduino and provide the necessary voltage and current.

Phase 3: Software Development (Arduino Programming)

1. Initialization: Write the Arduino code to initialize all the peripherals:

- Set the pin modes for digital input/output pins connected to sensors, relays, buzzer, and motor driver control.
- Initialize the serial communication for Bluetooth.
- Initialize the I2C communication for the LCD.
- Initialize the DHT11 library.

2. Temperature Monitoring Code: Write code to read temperature data from the DHT11 sensor at regular intervals and format it for display on the LCD.
3. Smoke Detection Code: Write code to read the analog value from the MQ2 sensor. Implement a threshold-based detection mechanism to trigger the buzzer when the smoke level exceeds the defined limit.
4. Bluetooth Communication Handling:

- Implement a serial communication protocol to receive commands from the Bluetooth terminal app. Define specific commands for turning lights on/off, setting fan speed levels, and controlling the window.
- Parse the received commands and execute the corresponding actions (e.g., toggling relay states, setting relay combinations for fan speed, controlling the motor driver).

5. Appliance Control Logic: Write the code to control the relay modules based on the commands received via Bluetooth and the logic for fan speed control using the second relay module and the AC regulator.
6. Motion Detection and Automatic Shutdown Logic: Write code to read the state of the PIR sensor. Implement a timer that starts when no motion is detected. If the timer exceeds a set duration, the code should send commands to turn off the lights and the fan. Reset the timer upon detecting motion.
7. Window Control Logic: Write code to control the L298N motor driver to open and close the window based on commands received via Bluetooth. Implement appropriate delays or potentially limit switch monitoring (if added) to control the duration of motor operation.
8. LCD Display Logic: Write code to display the temperature readings and potentially the status of other components (e.g., fan speed level, smoke detected) on the LCD.

Phase 4: Integration and Testing

1. System Integration: Combine all the hardware components and upload the complete Arduino code.
2. Functional Testing: Thoroughly test each functionality of the system:
   - Verify the on/off control of both lights and the fan via the Bluetooth app.
   - Test all the implemented fan speed levels using the Bluetooth app.
   - Check if the temperature is being displayed correctly on the LCD.
   - Test the smoke detection by exposing the MQ2 sensor to smoke and verify if the buzzer activates.
   - Test the automatic appliance shutdown by ensuring that the lights and fan turn off after a period of no motion.
   - Verify the opening and closing of the window using the Bluetooth app.
3. Debugging and Troubleshooting: Identify and resolve any hardware or software issues encountered during testing. Use the Arduino serial monitors for debugging the code and check all wiring connections.
4. Performance Evaluation: Evaluate the response time of the system to Bluetooth commands, the accuracy of the temperature readings, the

sensitivity of the smoke detector, and the effectiveness of the motion-based shutdown.

Phase 5: Documentation and Refinement

1. Detailed Documentation: Document the entire project, including:
- Project overview, aims, and objectives.
- Detailed circuit diagrams and wiring instructions.
- Explanation of the Arduino code and its logic.
- Bill of materials (BOM) with component specifications.
- Step-by-step instructions for setting up and using the system.
- Troubleshooting guide.
- Potential future enhancements.
2. Code Optimization: Refine the Arduino code for efficiency and readability.
3. User Interface Improvement: If possible, explore ways to improve the user experience with the Bluetooth terminal app (e.g., using more descriptive commands).
4. Safety Review: Conduct a final safety review of the entire system, especially the high-voltage AC wiring. Ensure proper insulation and protection against electrical hazards.

Component specification

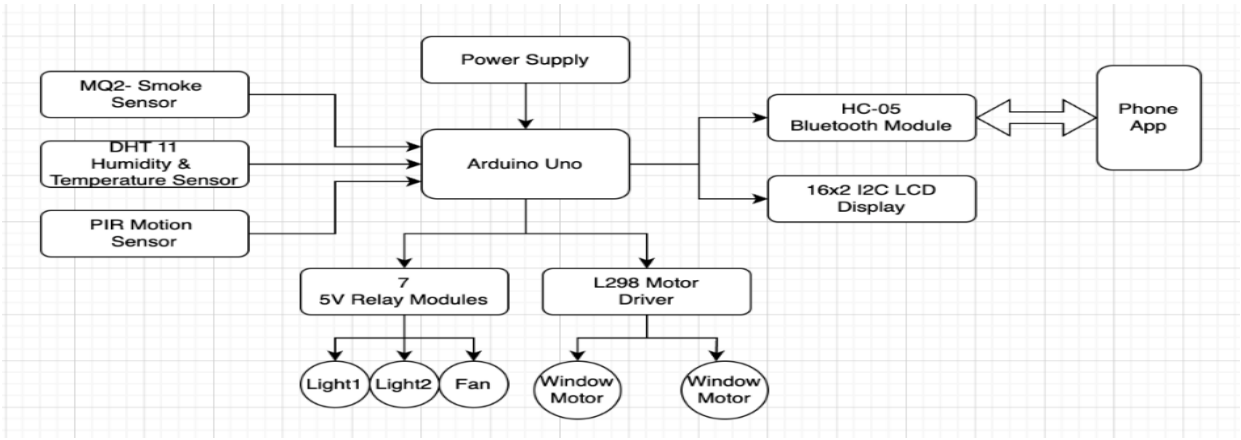

FIG.BLOCK DIAGRAM OF HOME AUTOMATION USING ARDUINO BASED SYSTEM

VIII. CONCLUSION

This project explores the exciting possibilities of home automation, blending affordability, functionality, and smart control into a single, user-friendly system. At its heart is the versatile Arduino Uno, orchestrating intelligent lighting and fan speed adjustments, real-time temperature monitoring, smoke detection for enhanced safety, and seamless wireless control via Bluetooth. With features like occupancy-based energy saving and automated window operation, the system creates a more responsive and efficient living environment.

By harnessing widely available components, this project showcases how home automation doesn't have to be complicated or expensive. The intuitive Bluetooth control ensures accessibility, while the PIR motion sensor optimizes energy use. Smoke detection adds an essential layer of safety, reinforcing the value of smart technologies in everyday life. The innovation in fan speed control using standard components further emphasizes the adaptability of Arduino-based solutions.

REFERENCES

[1] Lee, S., & Kim, K. (2018). Bluetooth-Enabled Wireless Control for Low-Power Home Automation Systems. Proceedings of the 2018 International Conference on Smart Technologies and Applications (ICSTA), 112-117.

[2] Ali, M., & Hussain, F. (2021). Intelligent Fan Speed Regulation in Smart Homes using Fuzzy Logic and Environmental Feedback. Journal of Ambient Intelligence and Humanized Computing, 12(5), 4897-4910.

[3] Brown, J., & Davis, L. (2019). A Cost-Effective Smoke Detection and Alert System for Enhanced Home Safety. Fire Safety Journal, 103, 25-33.

[4] Garcia, E., & Rodriguez, P. (2022). Occupancy-Based Energy Management in Smart Buildings using PIR Sensors. Energy and Buildings, 258, 111845.
[5] Wang, Q., Zhang, Y., & Li, Z. (2017). Design and Implementation of a Low-Cost Home Automation System Based on IoT. IEEE Access, 5, 16744-16751.
[6] Chen, H., & Wu, T. (2023). A Review of Wireless Communication Technologies for Smart Home Applications. Journal of Network and Computer Applications, 215, 103649.
[7] Patel, N., & Gupta, S. (2016). Development of a Smart Home Security System Using Arduino and GSM. International Journal of Computer Applications, 139(11), 1-5. (While focused on security, it highlights Arduino's versatility).
[8] Kim, J., Park, C., & Choi, H. (2024). Integration of Multiple Sensors for Enhanced Environmental Monitoring in Smart Homes. Sensors, 24(3), 789.
[9] Silva, R., Oliveira, B., & Sousa, A. (2020). A Survey on Home Automation Systems: Architectures, Technologies, and Applications. Energies, 13(22), 6069. (Provides a broader context for your specific system).