Addis Ababa Institute of Technology

School of Electrical and Computer Engineering

Innovation Center

Smart Home Automation System Project Documentation

Team Members

1,Ayalkbet Teketel

2,Bamlak Tadesse

3,Eyosiyas Tiruneh

4,Kenna Baye

5,Lemi Negaso

Submitted to -Innovation Center

Date of Completion or Last Update-1-20-2025

Table of Contents

You can access the complete documentation, source code, simulation diagram, and other related resources on our GitHub repository.

https://github.com/mr-Ayalk/SmartHomeAutomationProject

Smart Energy Management (Interreg Europe Policy)

"Smart energy management involves the efficient monitoring, controlling, and optimization of energy systems and infrastructures through information and communication technologies and data-driven strategies. By integrating sensors, analytics, and automation, smart energy management systems collect and analyze real-time data to make informed decisions, optimize energy consumption, reduce waste, and lower costs. Smart energy technologies are essential for creating the smart grid and improving system efficiency, sustainability, and cost-effectiveness."

Abstract

The **Smart Home Energy Automation** project is an innovative solution designed to revolutionize energy management in residential, educational, healthcare, and industrial settings. By integrating advanced technologies such as **Arduino Uno**, **HC-05 Bluetooth module**, **relays**, and a **Java-based desktop application**, the system enables users to remotely control household appliances and optimize energy consumption. The project comprises two core functionalities: **remote control of appliances** and **smart energy management**.

The system allows users to monitor and control appliances via a user-friendly desktop GUI, which is powered by a backend server and a **MySQL database**. It provides real-time analytics, including energy consumption graphs, bill calculations, and personalized recommendations to optimize energy usage. Additionally, the system issues warnings and advice based on device functionality and power consumption, ensuring efficient energy utilization.

This project aligns with the **Smart City Initiative**, aiming to create autonomous homes and contribute to sustainable urban development. It addresses critical issues such as inefficient energy usage, unfair billing disputes, and the need for disaster forecasting. By offering a scalable and adaptable solution, the project not only enhances convenience and energy efficiency but also creates job opportunities in hardware integration, software maintenance, and system installation.

The ultimate goal of this project is to establish a small enterprise focused on developing engineering technology solutions to address societal challenges, promoting efficient and optimized resource consumption through innovative electronic and tech-related products.

Introduction

Background

The rapid advancement of technology has paved the way for smarter and more efficient ways to manage energy consumption in homes, educational institutions, healthcare facilities, and industrial settings. Traditional energy management systems often lack the ability to provide real-time monitoring, remote control, and intelligent optimization, leading to inefficiencies and higher energy costs. The **Smart Home Energy Automation** project addresses these challenges by integrating cutting-edge technologies such as **Arduino Uno**, **HC-05 Bluetooth module**, **relays**, and a **Java-based desktop application** to create a comprehensive energy management solution.

This project is inspired by the global **Smart City Initiative**, which aims to create sustainable and autonomous urban environments. By starting with smart homes, we aim to contribute to this larger vision while addressing immediate societal needs such as energy efficiency, fair billing, and disaster forecasting. The system not only enhances convenience and control for users but also promotes sustainable energy practices through real-time analytics and AI-driven recommendations.

Objectives

The primary objectives of the Smart Home Energy Automation project are:

1. **Remote Control of Appliances**: Enable users to control household appliances remotely using smartphones, laptops, or other devices, enhancing convenience and energy savings.
2. **Smart Energy Management**: Utilize AI to forecast energy usage, provide optimized recommendations, and issue warnings for high energy-consuming devices.
3. **Real-Time Monitoring and Analytics**: Display real-time energy consumption data, calculate bills, and generate insights to help users make informed decisions.
4. **Energy Efficiency**: Promote efficient energy usage by allowing users to schedule and automate appliance operations.
5. **Fair Billing**: Resolve billing disputes by tracking energy consumption for individual devices.

6. **Scalability and Adaptability**: Design a system that can be easily scaled and adapted for use in homes, educational institutions, healthcare facilities, and factories.

Scope

The scope of the Smart Home Energy Automation project includes:

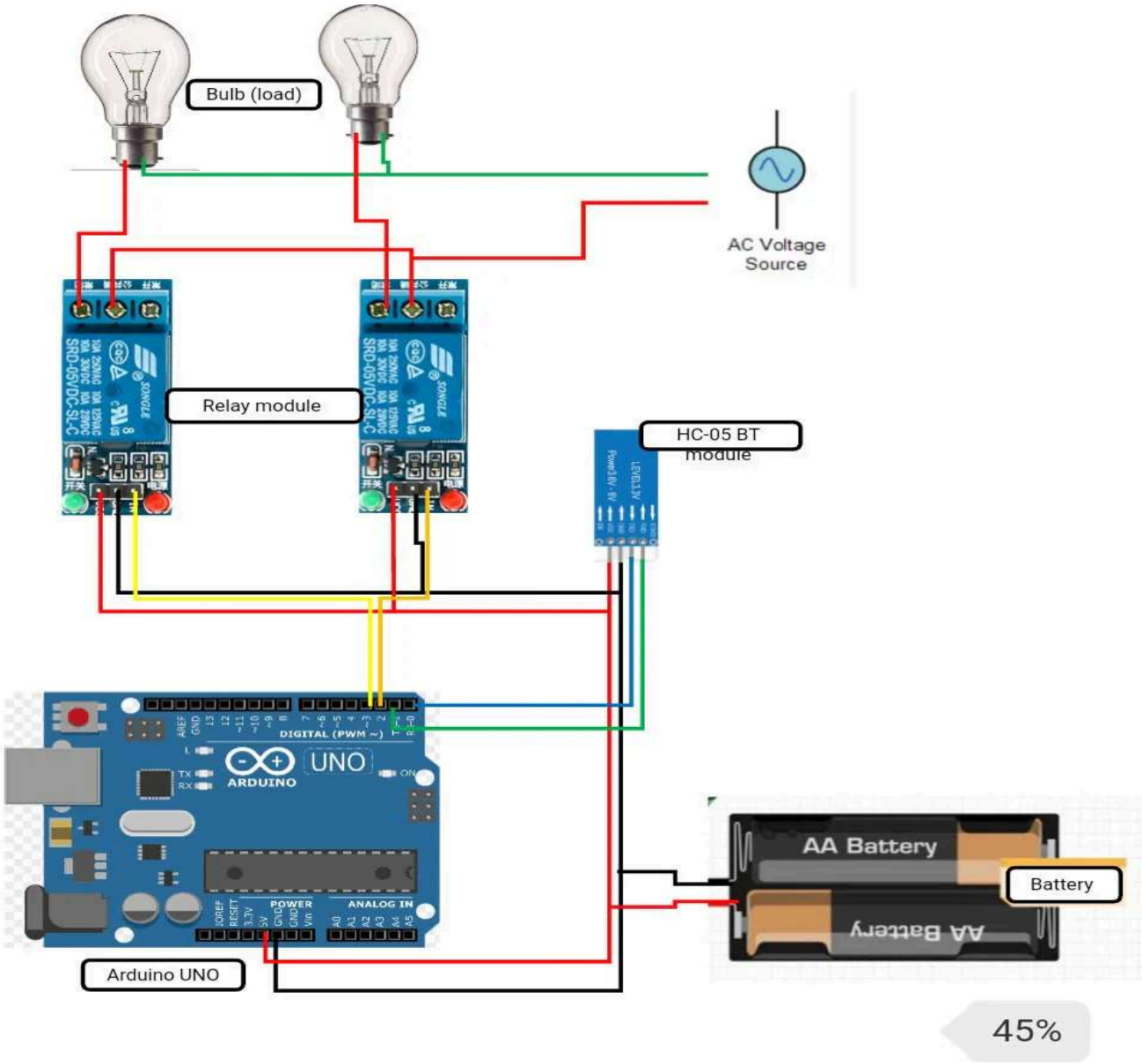
- **Hardware Integration**: Using Arduino Uno, HC-05 Bluetooth module, relays, and sensors to create a robust hardware setup for controlling appliances and monitoring energy consumption.
- **Software Development**: Developing a Java-based desktop application with a user-friendly GUI, backend server, and MySQL database for real-time data processing and storage.
- **Remote Control**: Enabling users to control appliances remotely via the desktop application.
- **Energy Analytics**: Providing real-time energy consumption data, bill calculations, and AI-driven recommendations for optimized energy usage.
- **Target Applications**: The system is designed for use in homes, educational institutions, healthcare facilities, and factories, making it a versatile solution for various settings.
- **Future Expansion**: The project is designed to be scalable, with potential future integrations such as disaster forecasting, advanced AI algorithms, and IoT-enabled devices.

System Overview

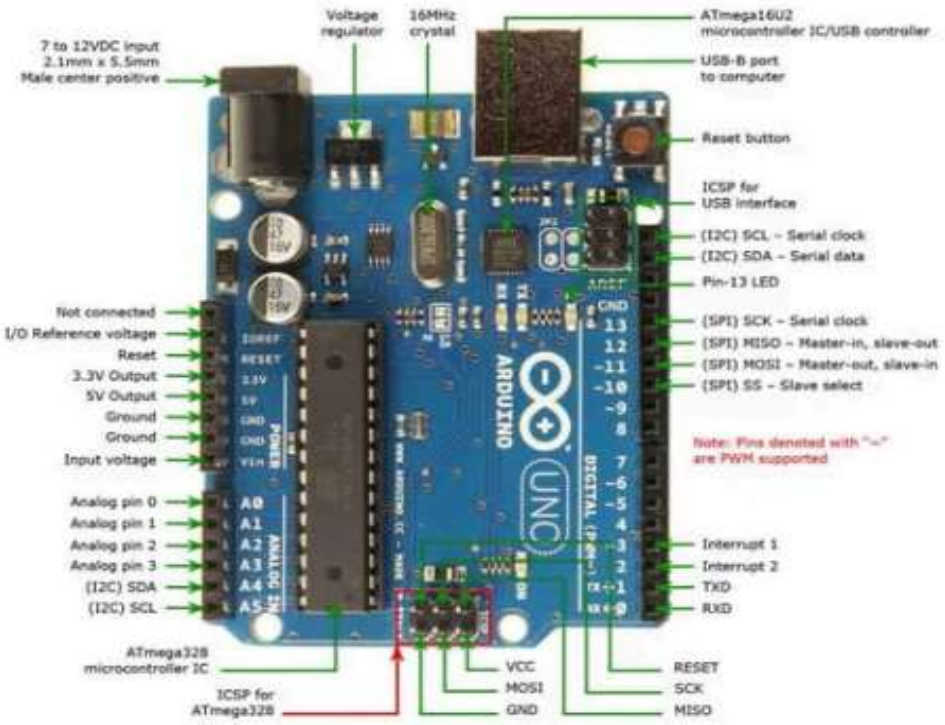
The Smart Home Energy Automation system is designed to provide users with remote control of household appliances and intelligent energy management. The system consists of three main components:

1. **Hardware Setup**: Built around the **Arduino Uno** microcontroller, the hardware includes sensors, relays, and the **HC-05 Bluetooth module** for communication.
2. **Desktop Application**: A **Java-based GUI** application that serves as the user interface for controlling appliances, monitoring energy consumption, and receiving recommendations.
3. **Backend and Database**: A **MySQL database** and backend server that store energy consumption data and enable real-time analytics.

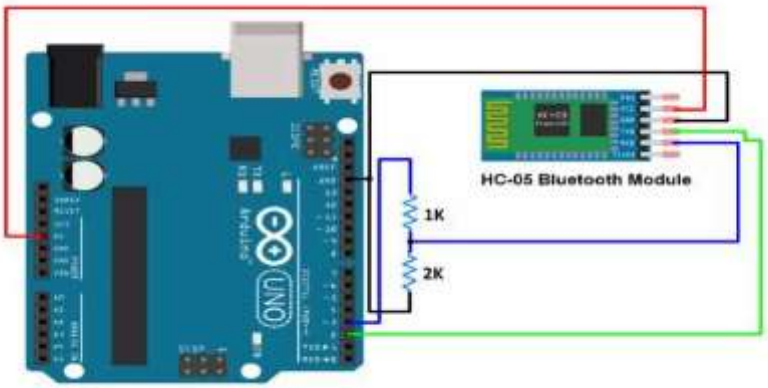
Block Diagram



- **Arduino Uno**: The central microcontroller that controls appliances and reads sensor data.



- **HC-05 Bluetooth Module**: Enables wireless communication between the Arduino and the desktop application.
- **Relays**: Act as switches to control high-power appliances (e.g., lights, fans, AC).
- **Sensors**: Measure energy consumption and environmental data (e.g., temperature, light intensity).
- **Java Desktop Application**: Provides a user-friendly interface for remote control and energy monitoring.
- **MySQL Database**: Stores energy consumption data and user preferences.
- **Backend Server**: Processes data and generates recommendations.



-

Components List

The system uses the following components:

- **Arduino Uno**: The main microcontroller for controlling appliances and processing sensor data.
- **HC-05 Bluetooth Module**: Facilitates wireless communication between the hardware and the desktop application.
- **Relays**: Used to switch high-power appliances on and off.
- **Sensors**: Measure energy consumption and environmental parameters.
- **Jumper Wires and Breadboard**: For connecting components.
- **Power Supply**: Provides power to the Arduino and connected components.
- **Desktop Computer**: Runs the Java application and hosts the MySQL database.

Software Tools

The following software tools were used in the project:

- **Arduino IDE**: For programming the Arduino Uno.
- **Java Development Kit (JDK)**: For developing the desktop application.
- **Eclipse and Intellij IDE**: For writing and debugging the Java code.
- **MySQL**: For database management and storage of energy consumption data.
- **Libraries**: Specific libraries for Bluetooth communication, sensor interfacing, and GUI development.

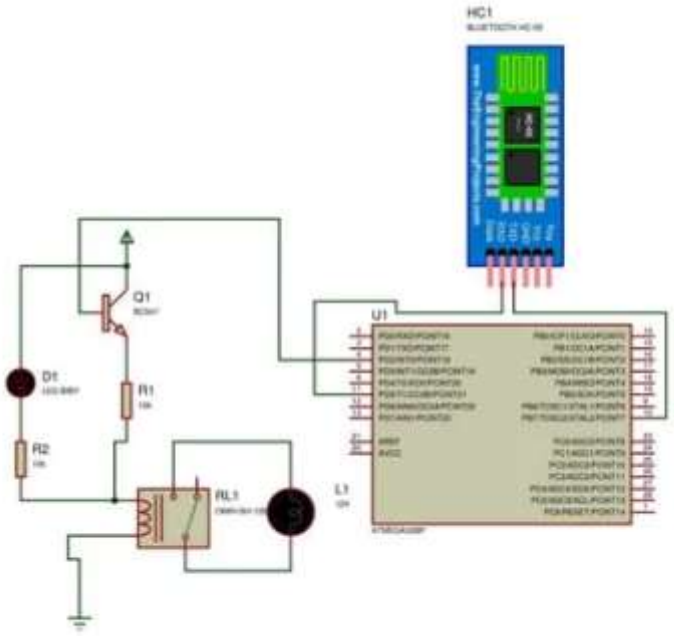
System Workflow

1. **User Interaction**: The user interacts with the Java desktop application to control appliances and view energy consumption data.
2. **Communication**: The application sends control commands to the Arduino via the HC-05 Bluetooth module.

3. **Hardware Execution**: The Arduino processes the commands and controls the relays to switch appliances on or off.
4. **Data Collection**: Sensors measure energy consumption and environmental data, which is sent back to the application via Bluetooth.
5. **Data Storage and Analysis**: The backend server stores the data in the MySQL database and generates real-time analytics, recommendations, and warnings.
6. **Feedback to User**: The application displays energy consumption graphs, bill calculations, and AI-driven recommendations to the user.

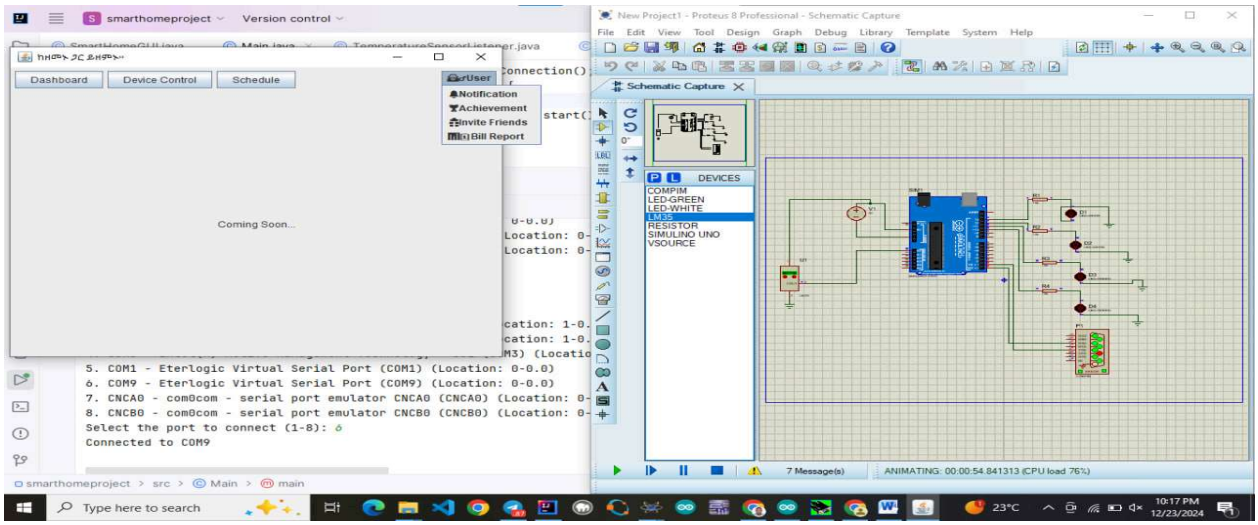
Key Points to Highlight in the System Overview

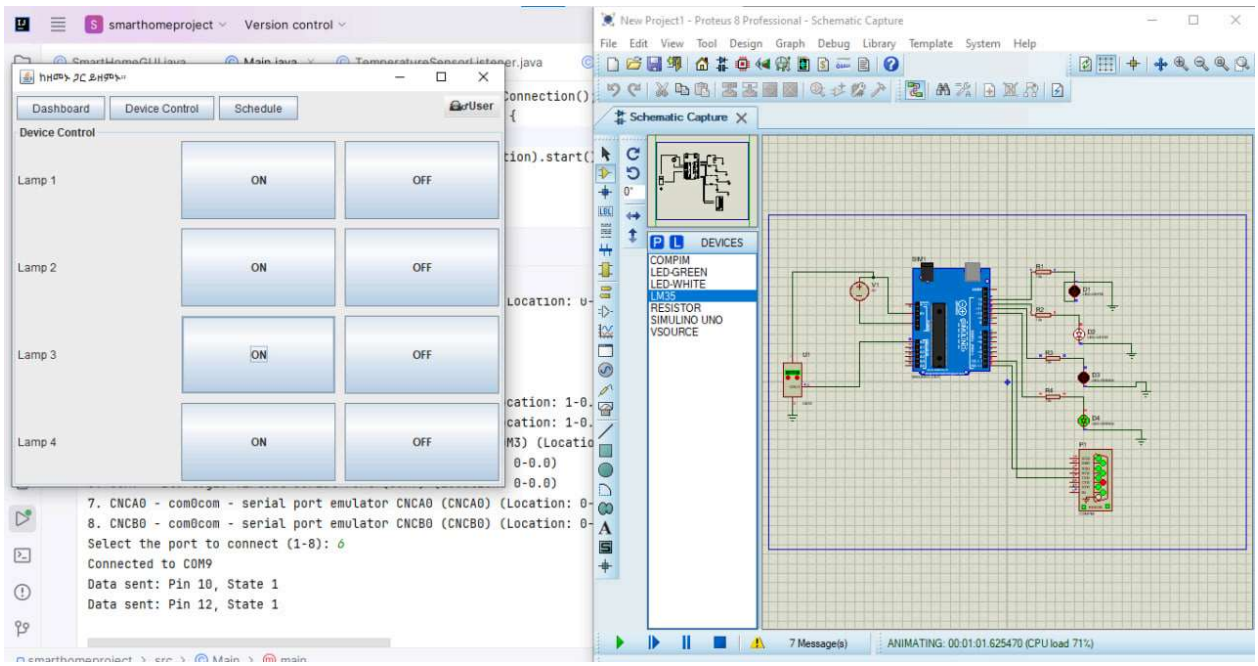
- **Interactions**: Explain how the hardware, software, and database interact with each other.
- **Flow of Data**: Describe how data flows through the system (e.g., from sensors to the Arduino, to the application, and to the database).
- **User Experience**: Highlight how the user interacts with the system and what benefits they gain (e.g., remote control, energy savings, real-time analytics).

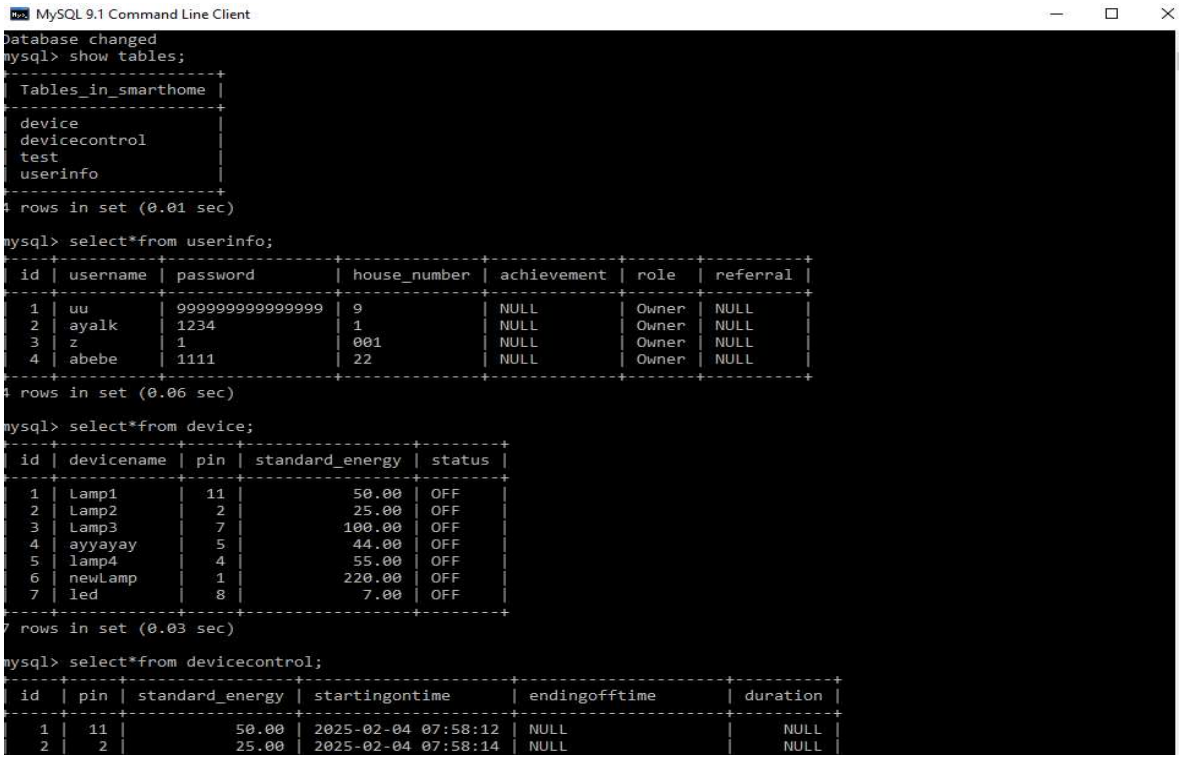


-
- **Arduino Uno**: Acts as the brain of the system, controlling appliances and reading sensor data.

- **HC-05 Bluetooth Module**: Enables wireless communication between the Arduino and the Java application.
- **Relays**: Serve as switches to control high-power appliances.
-
- **Java Application**: Provides a user-friendly interface for remote control and energy monitoring.
- **MySQL Database**: Stores energy consumption data and user preferences.
- **Backend Server**: Processes data and generates recommendations.

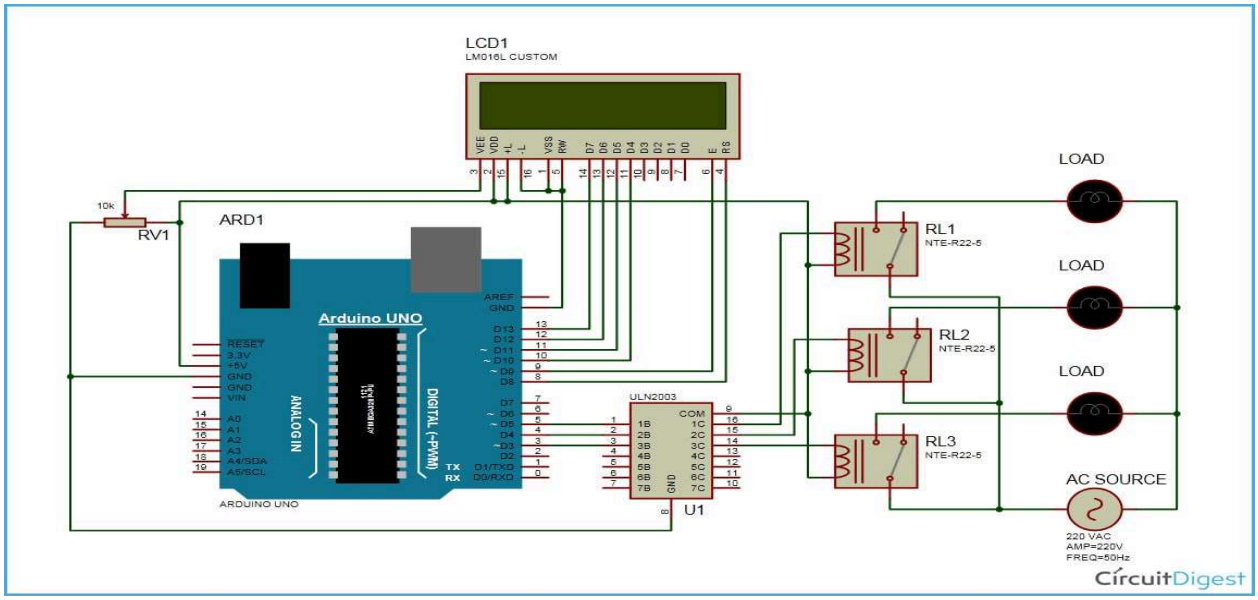






Hardware Design

Circuit Diagram



- A detailed circuit diagram is provided to illustrate the connections between the **Arduino Uno**, **HC-05 Bluetooth module**, **relays**, **sensors**, and other components.
- The diagram shows how the relays are connected to control high-power appliances and how the Bluetooth module facilitates communication between the Arduino and the Java application.

Component Descriptions

- **Arduino Uno**: The microcontroller that processes sensor data and controls relays.
- **HC-05 Bluetooth Module**: Enables wireless communication between the Arduino and the Java application.
- **Relays**: Electromagnetic switches used to control high-power appliances (e.g., lights, fans, AC).
- **Sensors**: Measure energy consumption and environmental data (e.g., temperature, light intensity).
- **Jumper Wires and Breadboard**: Used for connecting components in the circuit.
- **Power Supply**: Provides 5V power to the Arduino and connected components.

Power Supply

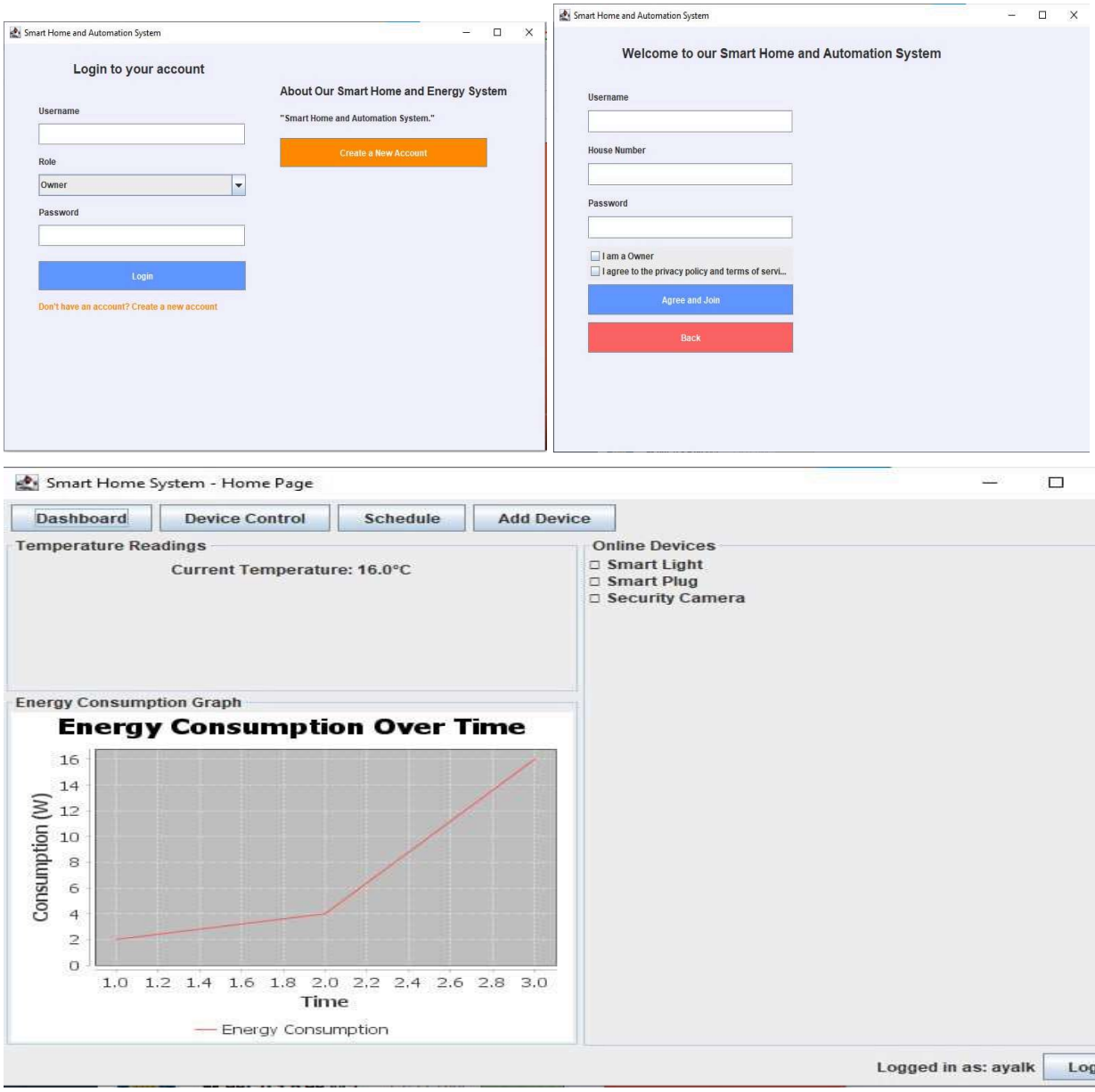
- The system is powered by a **5V power supply** connected to the Arduino Uno.
- Relays and high-power appliances are powered separately to ensure safe and stable operation.
- Proper voltage regulation is implemented to prevent damage to components.

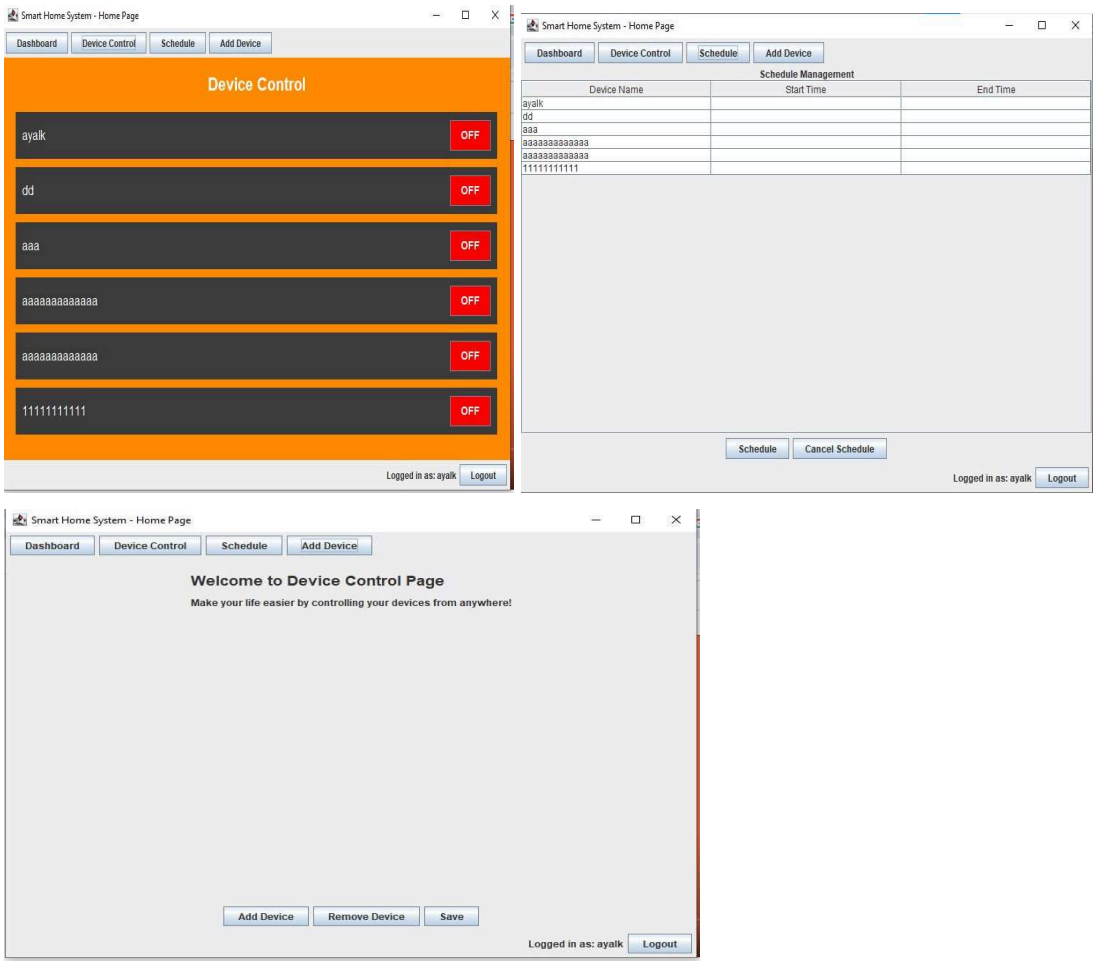
Safety Considerations

- **Insulation**: All high-power connections are properly insulated to prevent short circuits.
- **Fuse Protection**: Fuses are used to protect the circuit from overcurrent.
- **Heat Management**: Relays and other components are placed in well-ventilated areas to prevent overheating.
- **Component Ratings**: All components are selected based on their voltage and current ratings to ensure safe operation.
-

Software Design

Application Overview





The **Smart Home and Energy Management System** is a multifunctional application developed using **Java Swing GUI** for the front-end and **Arduino** for hardware interaction. It allows users to monitor and control various smart devices in their home, optimize energy usage, and receive insights into power consumption.

Key Functionalities

1. **User Authentication and Roles:**

- Users can log in with a username, password, and role selection (e.g., Owner).
- A new user registration system is available.

2. **Device Control and Automation:**
   - Users can turn smart devices (e.g., lights, plugs, security cameras) **on/off** from the app.
   - The system integrates with an Arduino to send commands for controlling devices.
3. **Temperature Monitoring:**
   - The system reads temperature data from a **DHT11** sensor connected to an **Arduino**.
   - The Java application receives the temperature via **serial communication** and displays it in real-time.
4. **Energy Consumption Analysis:**
   - The app features a graphical representation of **energy consumption over time**.
   - Users can monitor the power usage of devices and receive warnings about excessive consumption.
5. **Billing and Cost Calculation:**
   - The system can estimate the energy cost based on power usage and electricity rates.
6. **Warnings and Recommendations:**
   - The app provides **advice** on energy-saving practices.

- If power consumption exceeds a threshold, users receive alerts.

Technical Overview

- **Arduino Code:**
  - Reads temperature using the **DHT11 sensor** and sends it to the Java app via **serial communication**.
  - Can receive commands from the Java application to control smart devices.
- **Java Swing Application:**
  - Handles **user authentication**, **device control**, and **data visualization**.
  - Uses **JFreeChart** (or similar) to display energy consumption trends.
  - Communicates with Arduino using **jSerialComm** for real-time monitoring.

Serial communication code

```
//import com.fazecast.jSerialComm.SerialPort;

import com.fazecast.jSerialComm.SerialPort;

import java.io.IOException;
import java.util.Scanner;

public class SerialConnection {
   private SerialPort serialPort;

   // Connect to the first available port
   public boolean connectToAvailablePort() {
       SerialPort[] ports = SerialPort.getCommPorts();

       if (ports.length == 0) {
```

```
            System.err.println("No available serial ports found.");
            return false;
        }

        System.out.println("Available ports:");
        for (int i = 0; i < ports.length; i++) {
            System.out.println((i + 1) + ". " +
ports[i].getSystemPortName());
        }

        Scanner scanner = new Scanner(System.in);
        System.out.print("Select the port to connect (1-" + ports.length
+ "): ");
        int choice = scanner.nextInt();

        if (choice < 1 || choice > ports.length) {
            System.err.println("Invalid selection.");
            return false;
        }

        serialPort = ports[choice - 1];
        serialPort.setComPortParameters(9600, 8, 1, 0);
        serialPort.setComPortTimeouts(SerialPort.TIMEOUT_WRITE_BLOCKING,
1000, 1000);

        if (serialPort.openPort()) {
            System.out.println("Connected to " +
serialPort.getSystemPortName());
            return true;
        } else {
            System.err.println("Failed to open the port: " +
serialPort.getSystemPortName());
            return false;
        }
    }

    // Send data to the connected port
    public void sendData(byte pin, byte state) {
        try {
            if (serialPort != null && serialPort.isOpen()) {
```

```
                serialPort.getOutputStream().write(new byte[]{pin,
state});
                serialPort.getOutputStream().flush();
            }
        } catch (IOException e) {
            System.err.println("Error sending data: " + e.getMessage());
        }
    }

    // Close the port
    public void close() {
        if (serialPort != null) {
            serialPort.closePort();
            System.out.println("Port closed.");
        }
    }
}
```

Database config code

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

class DatabaseConfig {
    private static final String URL =
"jdbc:mysql://localhost:3306/smarthome";
    private static final String USER = "root";
    private static final String PASSWORD = "smarthome";


    public static Connection getConnection() throws SQLException {
        try {

            Class.forName("com.mysql.cj.jdbc.Driver");
        } catch (ClassNotFoundException e) {
            System.err.println("MySQL JDBC Driver not found. Include it
in your library path.");
            e.printStackTrace();
        }
        return DriverManager.getConnection(URL, USER, PASSWORD);
```

```
    }

}
```

Arduino code

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>

#define DHTPIN 2        // Pin where the DHT sensor is connected
#define DHTTYPE DHT11   // Change to DHT22 if using a DHT22 sensor

DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  dht.begin();
}

void loop() {
  float temperature = dht.readTemperature(); // Read temperature in Celsius

  if (isnan(temperature)) {
    Serial.println("Error reading temperature!");
  } else {
    Serial.println(temperature); // Send the temperature value via Serial
  }

  delay(1000); // Wait for 1 second before reading again
}
```

Results

Functionality and Performance

The Smart Home Automation System was successfully implemented and tested to evaluate its performance in real-world scenarios. The system demonstrated effective remote control of household appliances, real-time monitoring of energy consumption, and data-driven decision-making for optimizing energy usage. Below are the key findings from the system's functionality and performance tests:

1. **Remote Appliance Control:**
    - The Java-based desktop application effectively communicated with the Arduino Uno via the HC-05 Bluetooth module.
    - Users were able to switch appliances on and off with minimal latency.
2. **Energy Monitoring and Analysis:**
    - The system accurately recorded real-time energy consumption data and stored it in the MySQL database.
    - The graphical representation of energy usage trends provided insightful analytics for optimizing power consumption.
3. **Smart Energy Management:**
    - The system successfully generated warnings for high power-consuming devices.
    - AI-driven recommendations provided useful suggestions to reduce energy waste and lower electricity bills.
4. **Performance Efficiency:**
    - The system showed stable and reliable operation, with smooth communication between hardware and software components.

- Energy monitoring updates were received in near real-time, allowing users to make informed decisions.

5. **Scalability and Adaptability:**

    - The modular design allows for easy expansion, supporting additional sensors and appliances.

    - The system is adaptable for use in various settings such as homes, schools, healthcare facilities, and industrial environments.

Conclusion

The Smart Home Automation System successfully achieved its goal of providing an efficient and user-friendly solution for remote appliance control and smart energy management. By integrating Arduino Uno, the HC-05 Bluetooth module, relays, and a Java-based desktop application, the system enables users to monitor and optimize their energy consumption effectively.

This project contributes to sustainable energy practices by offering real-time analytics, AI-driven recommendations, and automation capabilities. The system enhances convenience, reduces energy waste, and promotes cost-effective energy consumption. Furthermore, it aligns with the broader Smart City Initiative by introducing intelligent home automation solutions.

While the project has demonstrated strong performance, future work may focus on integrating additional IoT capabilities, cloud-based data storage, and advanced AI algorithms for even more efficient energy management. The implementation of a mobile application for greater accessibility and an internet-based control mechanism could further improve the system's usability and functionality.

Acknowledgment

First and foremost, we would like to express our deepest gratitude to God for granting us the strength, patience, and perseverance to complete this project successfully.

We extend our heartfelt appreciation to **Dr. Libsework** for his invaluable support, from providing us with a dedicated workspace and necessary materials to assigning a capable team of supervisors and advisors to guide us throughout the project. His contribution played a crucial role in the successful completion of this endeavor.

We are also immensely grateful to our supervisor and advisors, **Kidus, Hermon, and Mailaf**, for their continuous technical and psychological support. Their expertise, guidance, and encouragement helped us overcome numerous challenges and improve the quality of our work.

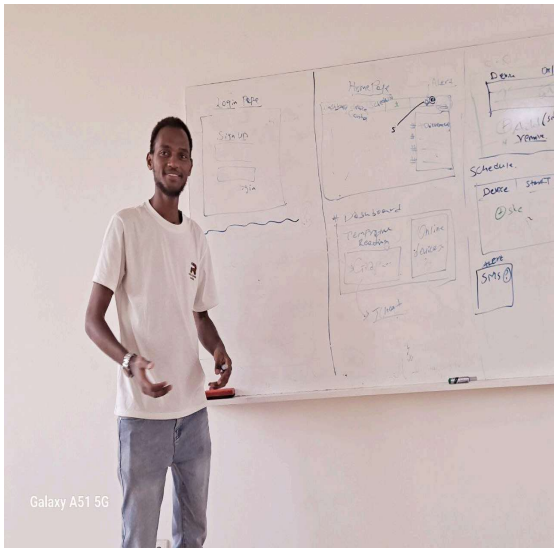
We would also like to thank the **Addis Ababa Institute of Technology Innovation Center** for offering us an excellent environment to develop and implement our ideas. The resources and assistance provided by the center greatly contributed to the progress of our project.

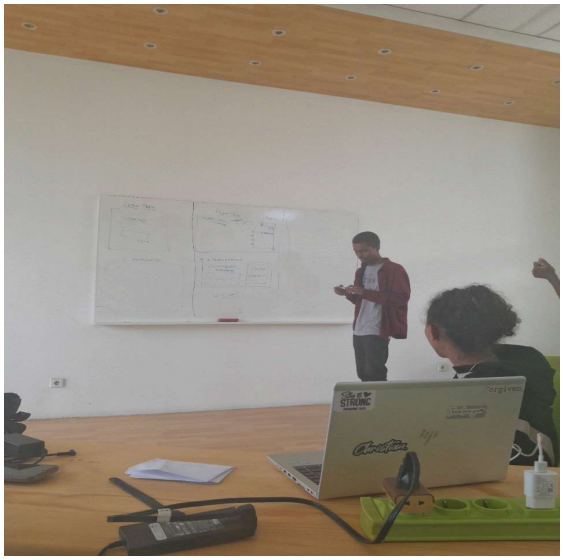
Additionally, we extend our sincere appreciation to our **Java instructor, Hailemelekot**, for his dedication and effort in teaching us essential programming skills that played a critical role in developing the software component of our system.

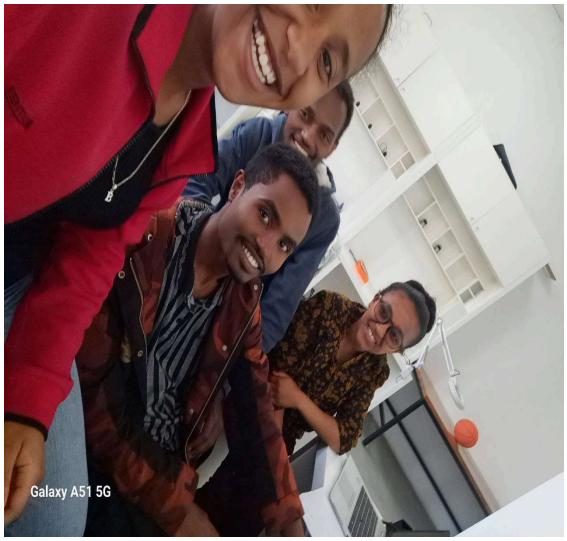
Finally, we would like to thank all individuals who supported us directly or indirectly throughout this journey. Their encouragement, feedback, and assistance were instrumental in bringing this project to fruition.

Working and Meeting Time

-






Galaxy A51 5G