Sri Lanka Institute of Information Technology

SMART ENERGY MANAGEMENT SYSTEM

Software Requirement Specification

Professional Engineering Practice and Industrial Management - IE2090

Project ID: PEP_33

Submitted by:

Registration No.	Student Name
IT22115584	W K T Duranga (Group leader)
IT22549518	P R M Chaminda
IT22555830	N K D Narangaspitiya
IT22052810	K LH D Jayalath

Submitted to:

……………………………..

Ms. Narmada

20/4/2024

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

Software Requirements Specification (SRS) for Smart Energy Management System. This SRS delineates the software requirements for the main system of the Smart Energy Management System and its four essential functions. It aims to facilitate a significant reduction in energy consumption by 10-20% across homes and offices, leveraging smart technologies to automate control and ensure efficient energy utilization.

1.2 Document Conventions

Chapter Title:

Font Size: 18

Font Style: Times New Roman

Sub Title:

Font Size: 14

Font Style: Times New Roman

Description Paragraphs:

Font Size: 12

Font Style: Times New Roman

Line space: 1.5

1.3 Intended Audience and Reading Suggestions

The SRS caters to a diverse audience, including project managers, business analysts, designers, developers, quality assurance (QA) professionals, and clients. Specifically tailored for those involved in IT development, such as designers, developers, and QA personnel, the document's second chapter onwards provides detailed insights into system functionalities, technical specifications, and testing requirements. Clients can benefit from the document's comprehensive overview, gaining a clear understanding of the system's intended features and functionalities. By following a structured sequence from overview sections to chapters pertinent to each reader type, stakeholders can efficiently grasp the project's scope, technical details, and testing strategies, ensuring effective collaboration and successful project execution.

1.4 Product Scope

The software being specified is an Energy Management System (EMS), aimed at addressing the challenges posed by rising electricity costs and environmental impact associated with excessive energy use. This contributes to cost savings, enhanced sustainability, and a more responsive energy management approach. The main expected outcomes of the project include real-time monitoring and insights, optimized energy consumption, remote monitoring and control, scalability, and data-driven decision-making. The objectives are to develop a smart energy management system applicable for both home and office environments, utilizing voice recognition, Arduino-based electronics, temperature sensors, and motion sensors to reduce energy waste and enable real-time usage measurement. The goals of the system encompass improving energy efficiency, reducing waste, and enhancing sustainability while meeting compliance and reporting requirements.

1.5 References

Hewagamage, N. [@NisalHewagamage]. (2018, August 22). *Arduino Voice Control Room Lighting | Make your home "Smart" [2018 Sinhala]*. Youtube. https://www.youtube.com/watch?v=EsfCOY9vDFo

Research Tamilan [@researchtamilan4106]. (2023, April 23). *iot smart energy meter using nodemcu esp32 with Blynk app | iot projects using esp32 #iotproject*. Youtube. https://www.youtube.com/watch?v=pp6OIvShxIo

RoboTechZone [@robotechzone]. (2023, July 15). *Temperature based fan speed controller using arduino UNO || ARDUINO PROJECTS*. Youtube. https://www.youtube.com/watch?v=Z1vYp8NE9LI

Shameer, T. T. [@TechTrendsShameer]. (2021, July 13). *Motion Sensor Control Electric Bulb | home automation*. Youtube.

https://www.youtube.com/watch?v=UUIAMvLilb0

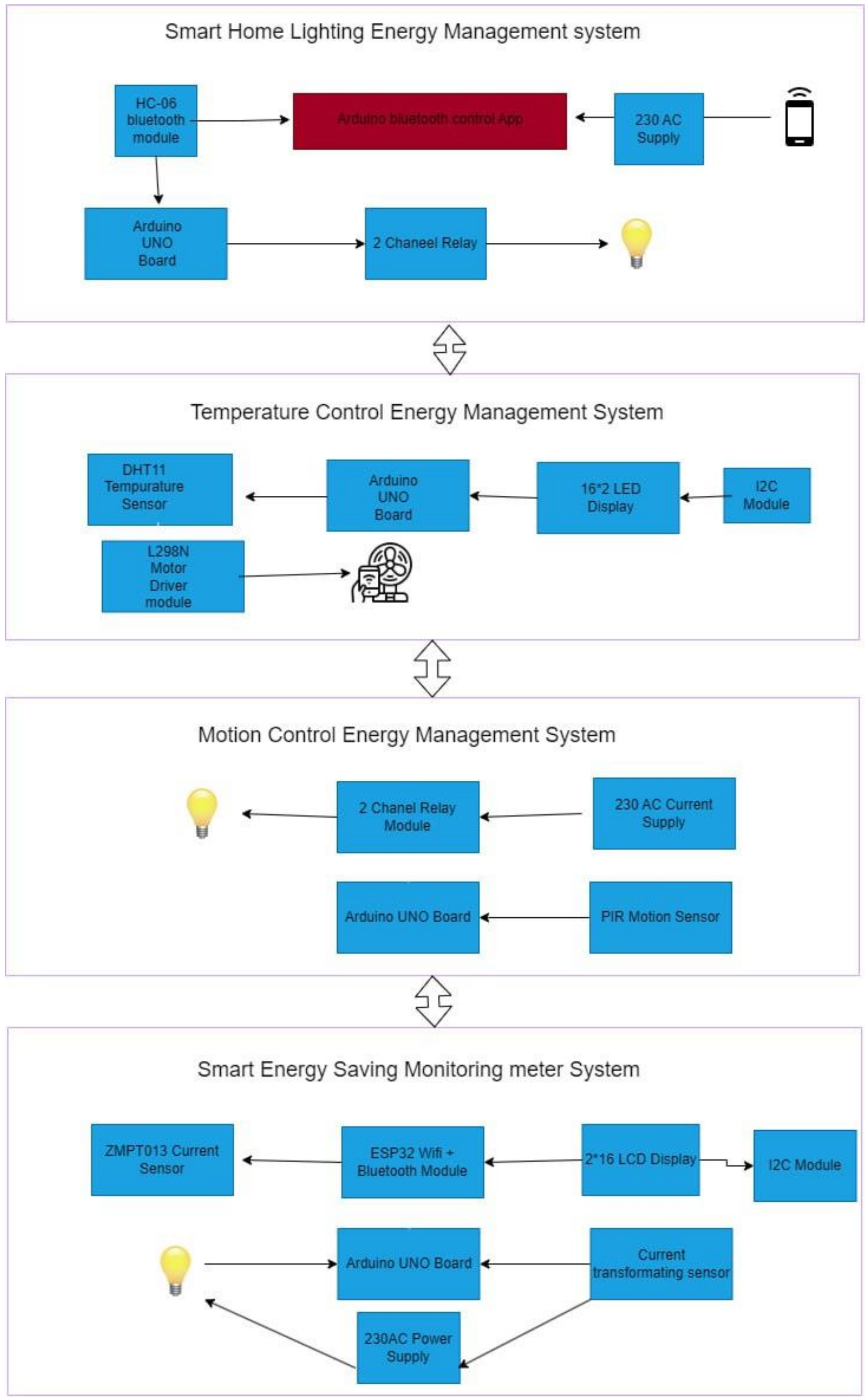
2. Overall Description

2.1 Product Perspective

The Smart Energy Management System project appears to be a replacement for certain existing systems. It aims to replace traditional lighting systems, electric meters, fans, and light fixtures with more advanced and energy-efficient alternatives. Therefore, it is not a new, self-contained product, nor is it explicitly described as a member of a product family. Instead, it focuses on upgrading and enhancing existing systems to improve energy management and efficiency.

2.2 Product Functions

These functions collectively form the Smart Energy Management System, aimed at improving energy efficiency, user convenience, and overall sustainability in residential or commercial settings.



2.2 - Figure 1 - High-level architecture diagram

2.3 User Classes and Characteristics

The Energy Management System (EMS) caters to various stakeholders with distinct roles and requirements.

For residential users, the EMS serves as an integral part of their daily routines, providing convenience and efficiency in managing energy consumption. Security features are paramount to ensuring users' trust and confidence in the product, enhancing their overall experience.

Testers play a crucial role in ensuring the EMS's functionality and reliability. Their quality assurance (QA) expertise is essential for thoroughly testing all components, including sensors, to guarantee optimal performance and adherence to standards.

Designers are tasked with creating a user-friendly interface and a seamless interaction experience. Their design skills and knowledge are instrumental in crafting intuitive interfaces that resonate with users, enhancing usability and satisfaction.

Fixers are responsible for accurately installing and maintaining EMS components. Their hands-on experience with the product ensures proper installation, troubleshooting, and upkeep, contributing to the system's overall reliability and longevity.

2.4 Operating Environment

The hardware platform for the Energy Management System (EMS) comprises Arduino boards and standard computing hardware. These components serve as the foundation for sensing, controlling, and managing energy usage within the system.

Regarding operating systems, the EMS supports a variety of platforms. Companion mobile apps are compatible with operating systems such as iOS and Android, providing users with convenient control and monitoring capabilities on their smartphones or tablets. Additionally, embedded operating systems like Atmel Studio facilitate the efficient operation of the EMS components.

Key software components and applications enhance the functionality of the EMS. Voice recognition software enables the Voice-Controlled Light System, enabling users to interact with the system using voice commands. This integration may extend to popular platforms like Amazon Alexa or

Google Assistant, enhancing user convenience. Furthermore, mobile applications are available for download on app stores, enabling users to remotely control and monitor the EMS from their iOS or Android devices, ensuring flexibility and accessibility.

2.5 Design and Implementation Constraints

Corporate security policies may dictate specific security measures, encryption standards, and access control mechanisms that must be implemented.

Compatibility with legacy hardware or systems may impose limitations on the selection of communication protocols and interface standards.

Third-party dependencies and licensing agreements may limit the options for integrating with external services or platforms.

Security requirements, such as encryption, authentication, and authorization mechanisms, may limit the options for implementing certain features or accessing sensitive data.

If the customer's organization will be responsible for maintaining the delivered software, developers may need to consider the organization's capabilities, preferences, and limitations in selecting technologies and designing the system architecture.

2.6 Project Documentation

The Arduino-based Smart Energy Management System is an innovative solution designed to optimize energy consumption and enhance efficiency in residential and commercial settings. Leveraging the power of Arduino microcontrollers, this system integrates various sensors and actuators to monitor and control energy usage intelligently. By collecting real-time data on electricity consumption, temperature, motion, and other relevant parameters, the system dynamically adjusts lighting, cooling systems, and other appliances to minimize waste and maximize savings. With user-friendly interfaces and seamless integration capabilities, this system empowers users to make informed decisions about energy usage, leading to reduced costs, environmental sustainability, and enhanced comfort and convenience.

2.7 User Documentation

The user manual for the Smart Energy Management System will serve as a comprehensive guide to help users effectively utilize all its features and functionalities. Here's what it will include:

1. **Introduction to the Smart Energy Management System:** This section provides an overview of the system, its purpose, and the benefits it offers in terms of energy efficiency and cost savings.
2. **Getting Started**: Users will find step-by-step instructions on how to set up the system, including hardware installation, software configuration, and initial setup procedures.
3. **User Interface Guide**: This guide will familiarize users with the system's user interface, explaining how to navigate menus, access different features, and interact with the system effectively.
4. **Functionality Guide**: Users will learn about the various functions and capabilities of the Smart Energy Management System, including how to control lights, monitor energy usage, adjust fan speed, and respond to motion detection.
5. **Troubleshooting and FAQs**: This section will provide solutions to common issues users may encounter while using the system, along with answers to frequently asked questions, to help troubleshoot problems quickly.
6. **Maintenance and Care**: Users will be informed about best practices for maintaining and caring for the system, including regular updates, hardware maintenance, and system checks to ensure optimal performance.

For each of the system components:

1. **Voice-Controlled Light System**: Users will learn how to control lights using voice commands and the Arduino Blue Control app, enhancing convenience and energy efficiency.

2. **Arduino-Based Electric Meter**: Users will understand how to monitor energy usage and costs effectively, leveraging real-time data displayed on the LED panel to optimize energy consumption.

3. **Smart Fan with Temperature Sensing**: Users will discover how the system automatically adjusts fan speed based on temperature readings, ensuring comfort and energy savings.

4. **Motion Sensor Light System**: Users will be informed about the system's ability to detect motion and control lighting accordingly, enhancing security and reducing energy waste.

2.8 Assumptions and Dependencies

The Smart Energy Management System project operates under several key assumptions and dependencies. It relies on the availability and compatibility of third-party components, the stability of the development environment, and consistent access to external data sources. Additionally, the success of the system hinges on user acceptance and adherence to regulatory compliance standards. Any deviations or changes in these factors could impact project timelines, deliverables, and overall effectiveness in optimizing energy management. Therefore, careful consideration and management of these assumptions and dependencies are crucial for project success.

The project relies on several critical dependencies to ensure its successful implementation and operation. These include the availability and functionality of hardware and software components essential for system functionality. Additionally, dependencies exist on the compatibility and accessibility of existing systems and external services, such as cloud platforms or data analytics services, which may provide additional functionality or processing capabilities. Furthermore, stakeholder involvement is vital throughout the development process for requirement gathering, feedback, and decision-making. Managing these dependencies effectively is essential for mitigating risks and ensuring the project's overall success.

3. External Interface Requirements

3.1 User Interfaces

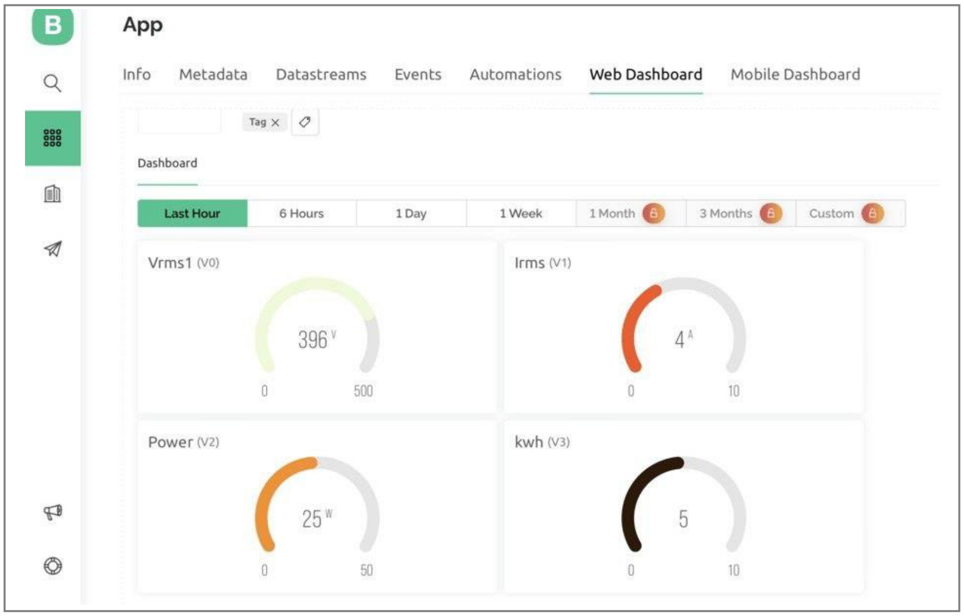
The Arduino Blue Control app:

The app facilitates a voice-controlled light system, enabling users to effortlessly control bulbs with voice commands. Integrated with Arduino hardware, users can issue verbal instructions via the Arduino Blue Control app, connecting to the system via Bluetooth. This seamless integration allows for easy management of lighting environments, whether it's turning lights on or off or adjusting illumination levels. With this innovative system, users experience hands-free operation and enhanced convenience in home automation.

3.1 – Figure 1 - The Arduino Blue Control app user interface

Blynk app:

The Arduino-Based Electric Meter's user interface uses the Blynk mobile application for intuitive monitoring of electricity usage. It features customizable widgets and real-time readings from sensors connected to the Arduino board. The interface adheres to Blynk's GUI standards, displays real-time readings of current and voltage from sensors connected to the Arduino board, offering standard buttons for navigation and interaction. Error messages are displayed concisely, guiding users to troubleshoot issues. This enhances user engagement and empowers informed decisions about energy usage and cost efficiency.



3.1 – Figure 2 - Blynk app user interface

3.2 Hardware Interfaces

For the Smart Fan with Temperature Sensing, the interface between the software and hardware involves communication with temperature sensors and fan control mechanisms. Data interactions entail reading temperature values from the sensor and sending control signals to the fan motor for speed adjustment. The logical characteristics involve interpreting temperature readings, determining appropriate fan speed levels (slow, medium, or high), and displaying them on an LCD display.

Regarding the Arduino-Based Electric Meter, the interface involves communication with current and voltage sensors to measure load parameters. Data interactions entail reading current and voltage values from the sensors and displaying them on an LCD display. The logical characteristics involve processing sensor readings to calculate current consumption and voltage levels.

3.3 Software Interfaces

The Energy Management System (EMS) is developed using the latest version of the Arduino IDE, incorporating libraries for sensor data processing and a lightweight database for storage and retrieval. It might also interact with IoT platforms via MQTT or RESTful APIs for efficient data exchange. Incoming data includes critical sensor readings such as temperature, voltage, and current, essential for energy monitoring and optimization. Outgoing data comprises control signals and energy reports sent to user interfaces or remote servers. Communications are secured and adhere to detailed protocols in API documentation. Data sharing across components, including sensor readings and user settings, is managed through a centralized mechanism in a multitasking environment, ensuring data integrity and thread safety.

3.4 Communications Interfaces

The system relies on robust communication functions to interact with IoT devices and Arduino hardware effectively. This includes protocols such as MQTT or HTTP for data transmission between sensors, actuators, and the central system. Message formatting may follow JSON or XML standards to ensure compatibility and readability across platforms. Communication security is paramount, requiring encryption mechanisms like SSL/TLS to protect sensitive data from unauthorized access or tampering. Data transfer rates should be optimized for real-time monitoring and control, with synchronization mechanisms ensuring data consistency and system reliability.

4. System Features

The functional requirements for the product are organized based on system features, illustrating the major services provided. This structure allows for a clear understanding of use cases, user classes, and functional hierarchy, ensuring logical organization and comprehensive coverage of system capabilities.

4.1 System Feature 1-Functional requirements of the project

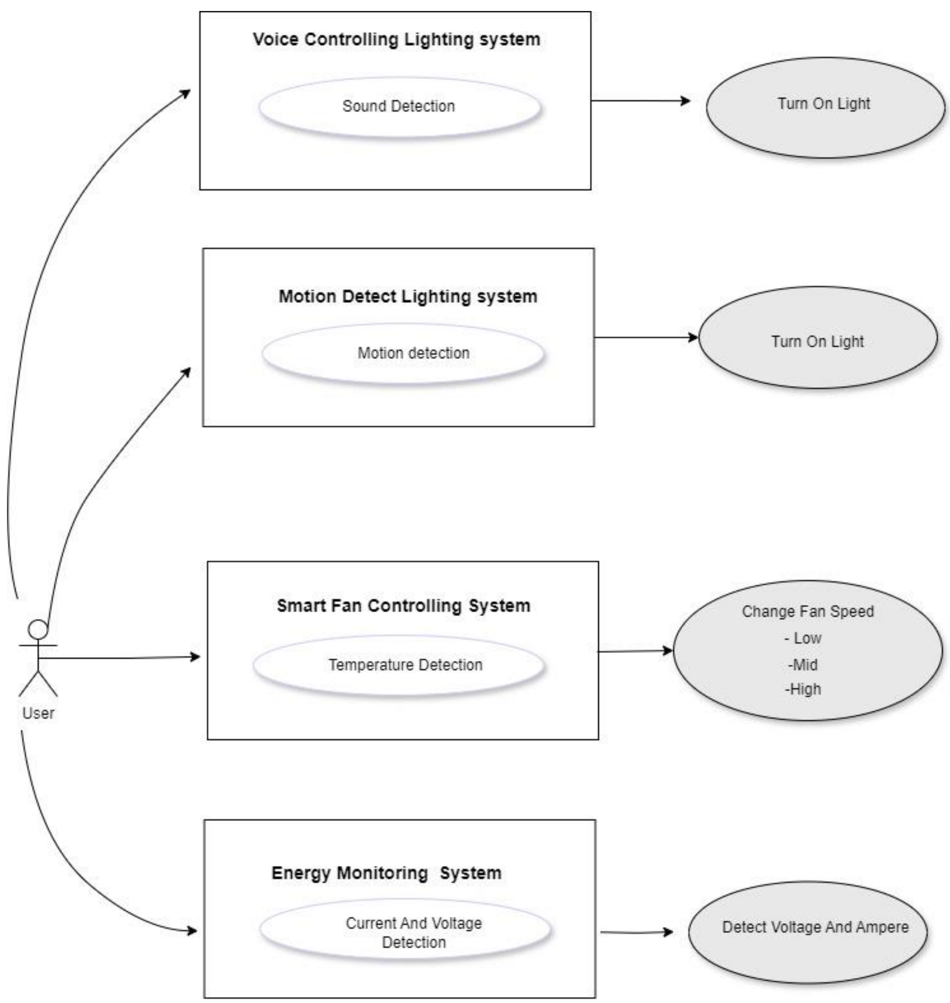
F1	**Voice-Controlled Light System**
Input	Voice commands from users to turn lights on/off , Bluetooth signals from the Arduino Blue Control app
Process	gathering requirements for related sensors and designing, coding, and hardware part
Output	when giving a voice command, the bulb is turned on or stays turned off. It is fully automating the system.
Definition	a voice-controlled light system to reduce electricity costs by controlling light bulbs as needed.

F2	**Arduino-Based Electric Meter**
Input	Current and voltage readings from current and voltage sensors
Process	Designing and connecting all sensors and equipment
Output	Per unit cost, efficient energy process, and other important details are displayed on the LED panel.
Definition	The unit cost related to the equipment used can be confirmed efficiently and effectively.

F3	**Smart Fan with Temperature Sensing**
Input	Temperature readings from temperature sensors
Process	Designing a circuit diagram using a temperature sensor, an Arduino board, and other necessary things.
Output	Fan speed automatically changes to slow, medium, or high levels according to the temperature.
Definition	Fan speed is controlled according to the current temperature format.

F4	**Motion Sensor Light System**
Input	Motion detection signals from motion sensors
Process	connecting to the motion sensor, Arduino Uno board, and other equipment
Output	If someone is exposed to the sensor, it is turned on; if it is not, the bulb is off.
Definition	The bulb only lights up according to motion.

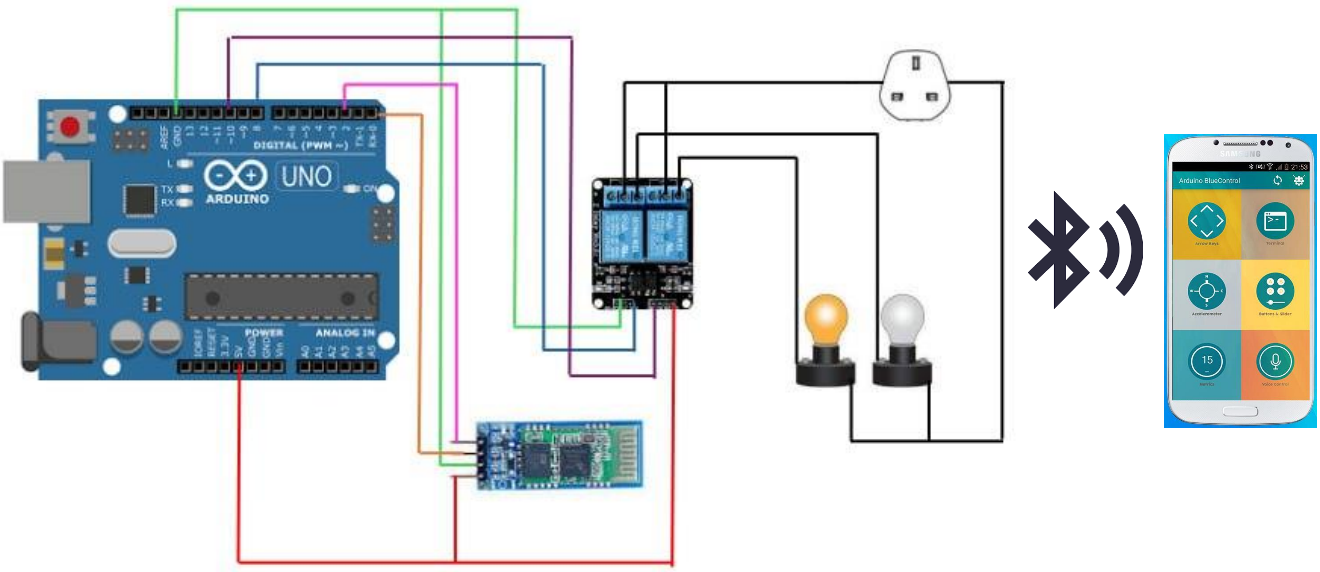
4.2 System Feature 2-Use case diagram



4.2 – Figure1 - Use case diagram for Smart Energy Management System

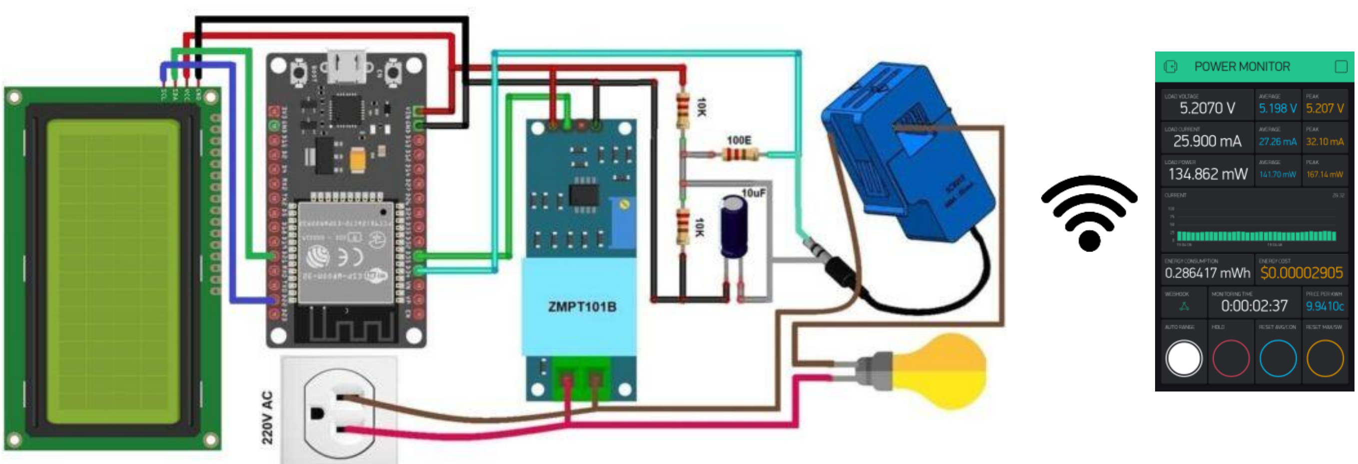
4.3 System Feature 3-Circuit diagrams

4.3.1 - Voice-Controlled Light System



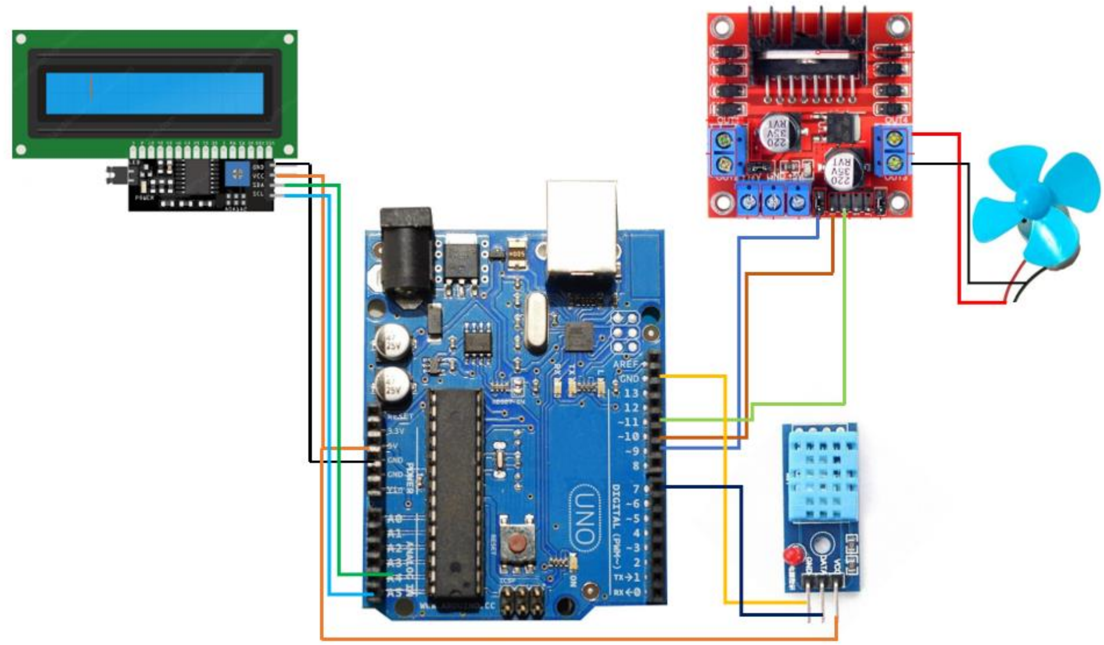
4.3.1 – Figure 1- Voice-Controlled Light System circuit

4.3.2 - Arduino-Based Electric Meter



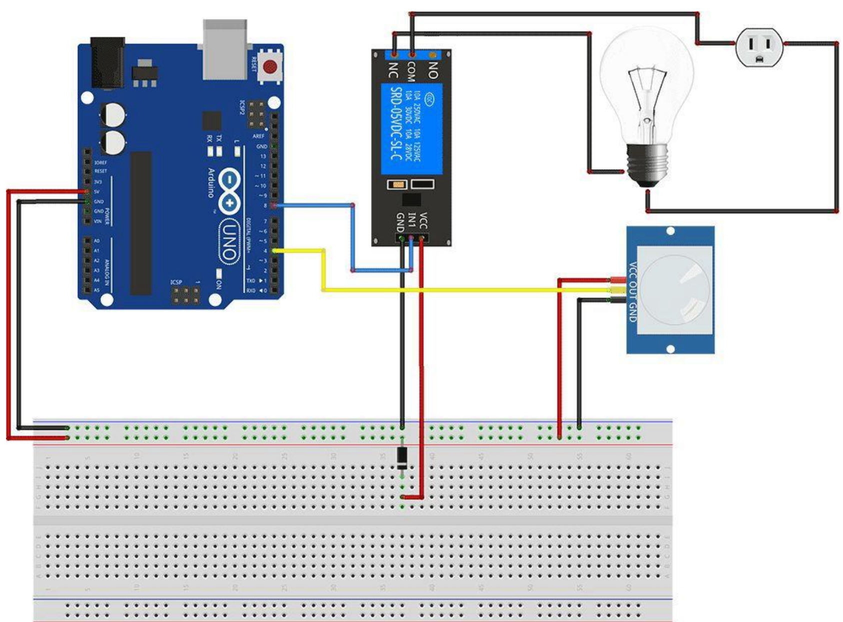
4.3.2 – Figure 1 - Arduino-based electric meter circuit

4.3.3 - Smart Fan with Temperature Sensing



4.3.3 – Figure 1 - Smart fan with temperature sensing circuit

4.3.4 - Motion Sensor Light System



4.3.4 – Figure 1 - Motion sensor light system circuit

5. Nonfunctional Requirements

5.1 Performance Requirements

The Voice-Controlled Light System is required to recognize and process voice commands with at least 95% accuracy in environments with ambient noise up to 60 dB, ensuring reliable operation. This high accuracy is crucial to prevent unintended light activations or deactivations, optimizing energy use and enhancing user satisfaction. Additionally, the system must respond to commands within 2 seconds, facilitating prompt and effective lighting control.

The Arduino-Based Electric Meter is designed to update its display with current usage details at least every 10 seconds, ensuring timely and accurate information for effective monitoring and management of electricity usage. This real-time data processing and display capability are crucial for enabling users to make informed decisions about their energy consumption without experiencing any delays.

The Smart Fan with Temperature Sensing must respond to temperature changes within 5 seconds and adjust its speed accurately based on set thresholds to maintain room comfort and enhance energy efficiency. Continuous monitoring and rapid adjustments are essential for optimal performance and significant energy savings.

The Motion Sensor Light System must detect motion within 0.5 seconds and activate lighting, which remains on for at least 30 seconds after movement ceases, to ensure safety and minimize energy waste. This quick responsiveness and controlled duration are critical for optimal energy efficiency and functionality.

5.2 Safety Requirements

Safety requirements are paramount to mitigating potential risks associated with the product's use. These include implementing safeguards to prevent electrical hazards, such as short circuits or overheating, and ensuring the system's components meet relevant safety standards and certifications. Additionally, the product must adhere to industry regulations and guidelines, such as those outlined by regulatory bodies like UL (Underwriters Laboratories) or CE (Conformité Européenne), to guarantee safe operation and minimize the risk of injury or property damage. Regular maintenance checks and user instructions for safe handling should also be provided to ensure continued safety compliance.

5.3 Security Requirements

Security requirements are paramount to safeguarding sensitive data and ensuring user privacy. This entails implementing encryption protocols for data transmission and storage, along with robust user authentication mechanisms to prevent unauthorized access. Compliance with regulations such as GDPR or HIPAA is essential, along with adherence to industry standards like ISO/IEC 27001. Obtaining certifications like SOC 2 validates the product's security posture and assures users of their data's protection. Regular security audits and updates are necessary to address emerging threats and maintain system integrity.

5.4 Software Quality Attributes

The software quality attributes for the product encompass adaptability, reliability, usability, and maintainability. Customers prioritize usability, requiring an intuitive interface for easy navigation and efficient task completion. Reliability is crucial, with a target of achieving a system uptime of at least 99% to ensure consistent performance. Developers focus on maintainability, aiming for clear code documentation and modular design to facilitate future updates and enhancements. Adaptability is also emphasized, enabling the system to seamlessly integrate with emerging technologies and evolving user needs.

5.5 Business Rules

Business rules dictate roles and permissions for performing specific functions within the product. For instance, only authorized administrators may access and modify system settings, while regular users have restricted privileges. Certain functions may require user authentication or managerial approval to proceed, ensuring accountability and data integrity. Compliance with regulatory standards, such as GDPR or industry-specific guidelines, must also be enforced in data handling and user interactions.

6. Other Requirements

Data Store Requirements: The EMS needs a robust data storage system to manage and analyze large amounts of energy-related data, including real-time sensor data, historical energy consumption records, user profiles, and system configurations. The system should support efficient retrieval, querying, backup, archiving, and retention for regulatory compliance.

Reuse Objectives: The EMS should focus on modularity, scalability, and reusability to optimize efficiency and reduce development efforts. It should be designed for easy integration into existing systems and future expansion, with standardized interfaces and protocols for interoperability with external systems. Documentation and knowledge sharing practices are also essential.

Performance Requirements: The EMS's performance is crucial for providing accurate energy consumption and operational efficiency insights. It should handle large data volumes in real-time, maintain responsiveness and scalability, and meet predefined benchmarks for user interactions. The system should adapt to varying workloads and environmental conditions, ensuring optimal performance under peak demand scenarios.

Integration and Interoperability: The EMS should integrate with existing infrastructure and third-party systems to maximize investments and comply with industry standards. Standardized communication protocols supported for interoperability with IoT devices, and enterprise systems like BMS or ERP software should facilitate cross-functional data exchange.

Appendix A: Glossary

IoT (Internet of Things)	The network of interconnected devices that can communicate and exchange data over the internet.
Embedded System	A computer system designed to perform specific functions within a larger system, often with real-time computing constraints.
Arduino	An open-source electronics platform based on easy-to-use hardware and software, widely used for prototyping and creating interactive projects.
Energy Management System (EMS)	A system designed to monitor, control, and optimize energy usage within a building or facility.
Sensor	A device that detects and responds to input from the physical environment, such as temperature, light, or motion.
Actuator	A component of a system that is responsible for controlling or moving a mechanism or system, such as turning a motor on/off.
Arduino IDE	Integrated Development integrated development environment used for programming Arduino boards.
LCD (Liquid Crystal Display)	A flat-panel display technology commonly used for digital clocks, calculators, and other devices for displaying information.
Microcontroller	A small computer on a single integrated circuit containing a processor core, memory, and programmable input/output peripherals.
Voltage Sensor	A device used to measure voltage levels in electrical circuits.
Current Sensor	A device used to measure the flow of electric current in a circuit.
Real-Time	Refers to systems that must respond to inputs or events within a specified time frame, typically in milliseconds or microseconds.

Appendix B: To Be Determined List

1. TBD: Define the specific requirements for user authentication and access control mechanisms.
2. TBD: Determine the encryption algorithms and key management strategies for securing sensitive data.
3. TBD: Identify backup and disaster recovery procedures to ensure data integrity and system availability.
4. TBD: Specify the monitoring and logging mechanisms for detecting and responding to security incidents.

Finalization of the Energy Management System

The finalization of the Energy Management System (EMS) marks the culmination of meticulous planning, development, and integration efforts. As the project draws to a close, the core functions of the system are seamlessly consolidated into a unified entity, representing a comprehensive solution tailored to meet the energy management needs of our stakeholders.

At this stage, the hardware components are meticulously orchestrated and integrated within a well-designed architecture diagram. This visual representation not only serves as a testament to the project's completion but also provides stakeholders with a clear understanding of the system's structure and functionality. Each hardware component, carefully selected and configured, plays a vital role in the efficient operation of the EMS, ensuring optimal performance and reliability.

In parallel, stringent security and data protection measures are devised and implemented to safeguard sensitive information and preserve system integrity. Robust authentication mechanisms, encryption protocols, and access controls are employed to mitigate potential threats and vulnerabilities. Additionally, comprehensive backup and disaster recovery procedures are established to mitigate data loss and ensure business continuity in the face of unforeseen events.

The finalization of the EMS signifies a significant milestone in our journey towards sustainable energy management. With a robust architecture in place, fortified by stringent security measures, we stand poised to empower our stakeholders with actionable insights and effective energy management capabilities, driving towards a greener and more sustainable future.