**Activity 1: Evaluate IoT technologies and identify problems  
A 1.1**

**What is the Internet of Things:**  
 The Internet of Things (IoT) is a transformative technology that enables physical devices, objects, and systems to be connected to the Internet, enabling them to collect, share, and analyze data in real time. This interconnected network encompasses everything from home appliances and wearables to industrial machinery and smart city infrastructure. By embedding sensors, software, and means of communication into these physical objects, IoT enablesThese devices communicate with each other and with centralized systems, creating a seamless flow of information that enhances automation, improves decision-making, and raises efficiency across sectors. The concept of IoT goes beyond mere communication; it represents a major shift in how we interact with the world around us, allowing for the creation of intelligent environments where devices can operate autonomously, respond to changes, and deliver valuable insights without human intervention. This technology has broad applications, ranging from optimizing energy consumption in homes and managing urban traffic to improving healthcare outcomes and revolutionizing industrial processes. As IoT continues to evolve, it becomes an integral part of our daily lives, driving innovation and enabling a more connected and intelligent world.

**1 L am a man of the same age The evolution of IoT technology**

* **Historical background**: The Internet of Things (IoT) as a concept began in the eighties with the idea of connected computing, where devices can communicate with each other easily. But the real shift occurred in the early second millennium with the development of Internet protocols and sensor technologies. Technologies such as Bluetooth and Wi-Fi allowed devices to connect and exchange data wirelessly, laying the foundation for what we know today as IoT.
* **Key developments**: The introduction of IPv6 in 2012 was an important step, as it significantly expanded the capabilities of IoT by enabling countless devices to be connected and unique in the Internet. This development was instrumental in enabling IoT to expand to become an integral part of industries such as smart cities, healthcare, and industrial automation. Today, IoT technologies are consideredFundamental in many innovations, representing a major development from their early conceptual stages.

**2.Current trends shaping IoT**

* **Integration with AI and machine learning**: One of the most impactful transformations in IoT is its integration with artificial intelligence (AI) and machine learning (ML). These technologies enable IoT devices to not only collect and transmit data, but also analyze and make decisions independently. For example, in smart homes, AI-driven systems can learn from user behavior and optimize energy use by adjusting heating, cooling, and lighting in real time.
* **The emergence of Edge Computing**: Another important trend is the emergence of Edge Computing, where data is processed closer to the source rather than relying entirely on centralized cloud services. This approach reduces latency, saves bandwidth, and allows for faster decision-making, which is critical for applications such as autonomous vehicles and industrial automation.
* **Expanding networks**: **5G**  The continuous expansion of **5G networks plays** a vital role in providing the necessary infrastructure to support the huge number of IoT devices. 5G **offers** faster data speeds, lower latency, and the ability to connect a large number of devices simultaneously, which is key for applications in smart cities where real-time communication between devices is crucial.
* **Focus on sustainability**: IoT technologies are increasingly being used to achieve sustainable development. Applications such as Smart Grids improve energy distribution and reduce waste, while Precision Agriculture uses IoT sensors to monitor soil and weather conditions, boosting crop productivity while reducing resource use.
* **Enhance security and privacy**: As IoT grows, so do potential vulnerabilities. There is a strong focus on developing more robust security frameworks, including encryption and secure boots to protect data from cyber threats. In addition, regulations such as GDPR drive the adoption of privacy-focused designs in IoT devices, ensuring that personal data is handled securely and transparently.

**Uses of IoT Technology**

**1. Smart Homes**

Smart homes are one of the most popular and widespread applications of IoT technology. In a smart home, a range of devices such as lighting, thermostats, security cameras, and home appliances are connected to the internet, allowing them to communicate with each other and the homeowner through a central hub or mobile app. The importance of smart home technology lies in automation and remote control, where homeowners can adjust lighting, temperature, and security settings from anywhere in the world using their smartphones. or voice commands, enhancing comfort and energy efficiency. For example, smart thermostats like Nest Learning Thermostat learn residents' preferences over time and adjust heating or cooling accordingly, resulting in significant energy savings. Another important aspect is security, as devices such as smart locks and cameras on doors provide real-time monitoring and instant alerts, enhancing security at home. Furthermore, the integration of voice assistants such as Amazon Alexa or Google Assistant allowsUsers control multiple devices using simple voice commands, which further simplifies everyday life. Smart home technologies continue to evolve, making homes more adaptive, responsive and sustainable, and contributing to the concept of connected living even more.

**2. Industrial Internet of Things (IIoT)**

Industrial Internet of Things (IIoT) is a specialized branch of the Internet of Things that focuses on the use of connected devices, sensors, and systems in industrial environments such as manufacturing, power, and transportation. IIoT enables real-time monitoring, predictive maintenance, and automation in industrial processes, greatly enhancing operating efficiency and reducing downtime. For example, in manufacturing, IIoT devices canMonitor machine performance, detect contaminations, and predict breakdowns before they occur, allowing for timely maintenance and reducing costly interruptions. IIoT's integration with advanced analytics and machine learning algorithms enables supply chain optimization, energy use, and asset management, leading to smarter and more sustainable industrial operations. In the energy sector, IIoT is used to monitor and control smart grids, optimize power distribution, and integrate renewable energy sources. The transportation industry also benefits IIoT by monitoring fleet performance, optimizing logistics, and ensuring safety. As industries continue to adopt IIoT, it is shifting towards more connected and intelligent systems, driving innovation, efficiency and competition globally.

**3. Connected vehicles**

Connected vehicles represent a transformative application of the Internet of Things that enhances the driving, safety and vehicle management experience. These vehicles are equipped with an internet connection and various sensors that collect data and transmit it to other vehicles, infrastructure, and centralized systems, forming a network that enables communication between vehicles (V2V) and between vehicles and infrastructure (V2I). This connectivity enables real-time traffic updates, remote diagnostics, and even self-driving features. For example, Tesla cars are known for their over-the-air updates, where new features and improvements are downloaded directly to the car, such as a smartphone update. In addition, connected vehicles contribute to improved road safety through systems such as automatic emergency braking, lane keeping assistance, and collision warnings, all of which depend on the vehicle's ability to Processing data from its surroundings in real time. Fleet management is another area in which connected vehicles shine, where companies can monitor vehicle location, fuel consumption, and driver behavior to improve roads, reduce costs, and enhance safety. The future of connected vehicles signals increased autonomy, as vehicles can operate with minimal human intervention, leading to safer and more efficient transportation systems.

**4. Smart Cities**

Smart Cities use IoT technology to improve urban life by making cities more efficient, sustainable, and responsive to the needs of their residents. By deploying sensors, cameras, and connected devices, smart cities can monitor and manage various urban services such as traffic management, waste collection, energy distribution, and public safety. For example, smart traffic lights can adapt to traffic conditions in real time, reducing congestion and improving vehicle flow. In waste management, smart containers with sensors can alert Entities responsible for collecting waste when it is full, which improves collection paths and reduces fuel consumption. In addition, smart grids in cities enable efficient electricity distribution, balancing supply and demand, and integrating renewables. It also enhances public safety through the use of connected surveillance systems and emergency response networks that can react more quickly to incidents. Furthermore, smart city initiatives often include citizen engagement platforms where residents can interact with city services, report problems, and access real-time information about public transport, air quality, and city events. As urban populations continue to grow, the adoption of smart city technologies is essential to creating sustainable, livable and resilient cities.

**5. Internet of Things in Healthcare**

The Internet of Things in Healthcare, sometimes known as the Medical Internet of Things (IoMT), is revolutionizing how healthcare is delivered by connecting medical devices, wearables, and health monitoring systems to the internet. This connectivity enables real-time patient monitoring, remote diagnosis, and data-driven decision-making, improving patient outcomes and reducing healthcare costs. Wearables such as smartwatches and fitness trackers can monitor vital signs such as the rate Heart rate, blood pressure, and glucose levels, providing ongoing data that doctors can analyze to detect early signs of health problems. Remote monitoring systems allow patients to manage chronic conditions from home, reducing the need for frequent hospital visits and enabling more personalized care. IoT is also being integrated into hospital environments to track medical equipment, monitor drug inventory, and even control the environment in operating rooms to reduce the risk of infection. In addition, AI-driven data analytics can process the vast amounts of data generated by IoMT devices, providing insights that can lead to more accurate diagnoses and better treatment plans. As the Internet of Things continues to evolve in healthcare, it promises to make healthcare more proactive, personalized, and accessible.

**6. Smart Agriculture**

Smart agriculture, also known as precision agriculture, is an IoT application that is transforming the agricultural industry by enabling farmers to monitor and manage their crops and livestock more efficiently. Through the use of IoT devices such as soil sensors, weather stations, and GPS-enabled machines, farmers can collect real-time data on soil condition, moisture levels, temperature, and crop health. This data allows for precise control of irrigation, fertilization, and pest control, which can increase significantly reduces crop productivity while reducing the use of water, chemicals and other resources. For example, IoT-powered irrigation systems can adjust water usage based on soil moisture levels and weather forecasts, ensuring that crops get the right amount of water without waste. Livestock management is also improving through the Internet of Things, with connected devices that monitor animal health, track feeding schedules, and even predict breeding cycles. Drones equipped with cameras and sensors are used to monitor crops, and provide aerial demonstrations to farms that help farmers identify areas that need attention. The integration of IoT with data analytics allows the creation of predictive models that can predict crop productivity, optimize planting schedules, and improve supply chain management. As the global population continues to grow, smart agriculture will play a vital role in ensuring food security and sustainable agricultural practices.

**7. Wearable technology**

Wearable technology is a growing field within the Internet of Things that includes devices worn on the body, such as smartwatches, fitness trackers, and health monitors. These devices collect and analyze data related to user health, activity levels, and the environment. For example, devices such as Apple Watch can monitor heart rate, detect rhythm disturbances, and even perform an electrocardiogram (ECG). Fitness trackers can monitor steps, calories burned, and sleep patterns, providing users with insights into their physical activity and encouraging them to lead healthier lifestyles. Wearables are also used in healthcare settings to constantly monitor patients' vital signs, providing doctors with real-time data that can guide treatment decisions. In addition to health applications, wearable technology is increasingly used in areas such as safety (such as wearable cameras for law enforcement), entertainment. (such as virtual reality glasses), and communication (such as smart glasses). The integration of AI and machine learning into wearables allows for more personalized and predictive experiences, making wearables an essential part of the IoT ecosystem.

**8. Smart Retail**

Smart Retail is an IoT application that enhances the shopping experience by integrating connected technologies into retail environments. IoT devices in stores can track customer behavior, manage inventory, and provide personalized shopping experiences. For example, smart shelves with weight sensors can alert employees when items are running out, ensuring that shelves are always full. Beacon technology canSend personalized offers and promotions to customers' smartphones as they move around the store. In addition, smart mirrors in fitting rooms can suggest complementary elements or allow customers to see how clothes look in different colors without trying them out. IoT in retail also improves supply chain management by providing real-time data on inventory levels, reducing waste and ensuring common items are always available. As e-commerce continues to grow, smart retail technologies help brick-and-mortar stores stay competitive by offering seamless, data-driven shopping experiences that meet the needs of modern consumers.

**3. IoT connectivity to traditional computer-based systems and networks**

* **Integration with cloud computing**: Traditional IT infrastructure, which relied primarily on internal servers, has evolved to include cloud platforms that provide storage and scalable computing capabilities. IoT devices produce vast amounts of data that need to be efficiently processed and stored. Cloud computing offers the infrastructure needed to handle this flow of data, enabling real-time analytics and remote monitoring, thus seamlessly integrating with IT based.
* **Adapting network protocols**: Traditional computer networks, originally designed for human-to-human communication, have been adapted to support machine-to-machine (M2M) communication, which is fundamental to IoT. Robust networking protocols such as MQTT and CoAP have been developed specifically for IoT, enabling devices to communicate efficiently within existing network infrastructure and ensure compatibility between new IoT systems and legacy IT infrastructures.
* **Strengthening cybersecurity measures**: The integration of IoT with traditional systems has led to the need to strengthen cybersecurity measures. Traditional security frameworks have been expanded to include IoT-specific protocols such as secure boots, hardware authentication, and end-to-end encryption. These measures are necessary to protect sensitive data as it moves between IoT devices and traditional IT systems, ensuring the integrity and confidentiality of the entire network.
* **Advanced data analytics**: Continuous data flows produced by IoT require real-time processing, which traditional systems were not originally designed for. To fill this gap, traditional systems have adopted big data analytics platforms and AI-driven tools, enabling organizations to derive actionable insights from IoT data, optimize operations, and create predictive models.
* **Evolution of software development practices**: IoT integration has also influenced the evolution of traditional software development practices. The software development lifecycle (SDLC) has been adapted to accommodate the rapid prototyping and iterative development required for IoT solutions. DevOps practices, which emphasize continuous integration and continuous deployment (CI/CD), are essential to managing the updates and frequent deployment needed to keep IoT systems secure and efficient.

**Conclusion**  
In conclusion, the Internet of Things (IoT) represents a huge quantum leap in how we interact with technology and the world around us. From its early beginnings as theoretical concepts to its current role as a driving force in industries ranging from healthcare to smart cities, IoT has evolved to become an essential part of modern life. IoT integration has enabledWith traditional computer-based systems new possibilities, allowing for more efficient operations, real-time decision-making, and enhanced communication. Current trends such as AI-driven automation, Edge Computing, and the expansion of 5G networks are shaping the future of IoT, making it smarter, more responsive and able to tackle complex challenges. As we continue to adopt IoT, a focus on security, privacy, and sustainability will be key to ensuring that this technology can be used safely and effectively to improve our lives and drive innovation across all sectors.

**A 1.2**

The Internet of Things (IoT) has become an essential part of modern technology, driving innovation across various industries by connecting devices, systems and services. As the IoT ecosystem continues to expand, the selection of appropriate frameworks, APIs, tools, and hardware plays a vital role in the successful development and deployment of IoT applications. These elements not only facilitate the connection and management of IoT devices , but it also affects the overall efficiency, scalability and security of the developed solutions. Understanding the advantages and disadvantages of these elements is essential to making informed decisions that can impact every stage of the software development lifecycle (SDLC), from planning and design to implementation and maintenance.

**Evaluation of frameworks, programming interfaces, tools and hardware in IoT**

**1. Frameworks and programming interfaces in IoT**

**Frameworks in IoT**  
Frameworks are an essential element in the development and deployment of IoT applications, providing an organized environment for managing devices, data, and communication protocols. Among the most well-known frameworks in the industry are **AWS IoT Core**, Google Cloud IoT, and**Microsoft Azure IoT Hub**. These platforms provide a comprehensive set of services that simplify the process of building, managing, and scaling IoT applications.

* **AWS IoT Core**: **AWS IoT Core** is a cloud platform that allows devices to securely connect to the cloud, interact with other devices, and manage data. The platform supports several communication protocols such as MMQTT, HTTP, andWebSockets. AWS IoT Core provides tools for device management, real-time data processing, and integration with other AWS services such as Lambda for serverless computing and S3 for data storage.
  + **Benefits**: AWS IoT Core is highly scalable, secure, and seamlessly integrates with other AWS services, making it ideal for large, complex applications. It also supports edge computing through AWS IoT Greengrass, enabling data to be processed near its source.
  + **Disadvantages**: The cost of AWS IoT Core can become complex and high as the number of connected devices and data usage increases. In addition, relying on a single provider can reduce flexibility and increase vendor dependency risk.
* Google Cloud IoT**:** Google Cloud IoT provides a range of services for device connectivity, management, and IoT data analysis. These services include Cloud IoT Core, which facilitates secure communication and device management, and is integrated with Google's powerful machine learning and analytics tools such as BigQuery and TensorFlow.
  + **Benefits**: Google Cloud IoT is powerful in analysis and machine learning, providing powerful tools for processing and analyzing large datasets. Integration with Google's global network ensures low latency and high availability.
  + **Disadvantages**: As with AWS, there is a risk of relying on a single vendor, and the cost can increase as data usage and the number of connected devices increase. In addition, the learning curve for using advanced Google Cloud services can be severe.
* **Microsoft Azure IoT Hub**: Azure IoT Hub is a comprehensive IoT platform that enables secure communication between applications and IoT devices that manage them. The platform supports several protocols including MMQTT, HTTPS, and AMQP, and integrates well with other Azure services such as Azure Machine Learning and Azure Stream Analytics.
  + **Benefits**: Azure IoT Hub provides excellent integration with the Microsoft ecosystem, making it a good choice for organizations already using Azure services. It also offers robust security features, including identity history and device authentication, and supports hybrid deployment between cloud and on-premises.
  + **Disadvantages**: As with other cloud platforms, managing costs can be challenging, especially with large-scale deployment. The complexity of the Azure ecosystem may also require specialized knowledge to navigate effectively.

**Programming Interfaces (APIs)**

APIs play a vital role in enabling connectivity between IoT devices, cloud services, and applications. APIs facilitate data exchange and command execution across different systems.

* : **RESTful APIs** REST (Representational State Transfer) is a popular technique for designing network applications. RESTful APIs are widely used in the Internet of Things because of their simplicity, scalability, and HTTP compliance.
  + **Advantages of RESTful APIs**: Stateless, which means that every client-to-server request must contain all the information needed to understand and process the request, making it scalable and reliable. It's also easy to implement and works well with web-based applications.
  + **Disadvantages**: RESTful APIs may be less efficient for real-time IoT applications due to HTTP overload and the need to send complete requests per interaction. They are also less suitable for environments with limited bandwidth and processing power.
* **MQTT Remote Message Transfer Protocol**: MQTT is a lightweight messaging protocol designed for use in environments with limited bandwidth and high latency, making it ideal for IoT applications.
  + **Benefits**: MQTT is highly efficient in terms of bandwidth and power consumption, which is important for battery-powered devices in the Internet of Things. It supports real-time connectivity with low latency, which is suitable for large-scale deployment in the Internet of Things.
  + **Disadvantages**: MQTT requires an intermediary to manage message distribution, which adds a layer of complexity. It is also limited in its support for synchronous communication between request and response, which may be necessary for some applications.
* **CoAP**  is another protocol specifically designed for use in restricted environments, such as those with limited processing capabilities and bandwidth. Similar to HTTP but much lighter.
  + **Benefits**: CoAP is efficient and designed to work well in low-power, loss-prone networks. Supports multicasting, which is useful in scenarios where messages need to be sent to multiple devices at the same time.
  + **Disadvantages**: CoAP is less prevalent compared to other protocols such as MQTT or HTTP, which may limit its compatibility with other systems. In addition, it lacks some of the sophisticated features and maturity of the most common protocols.

**IoT Development Tools**

IoT development tools are essential for designing, testing, and deploying IoT applications. These tools range from integrated development environments (IDEs) to patch and simulation tools.

* **Arduino IDE**: **Arduino IDE** is a simple and cross-platform application used to write and load programs on Arduino boards. It is especially popular in the IoT community for prototyping and education purposes.
  + **Benefits**: Arduino IDE is easy to use, making it available for beginners and hobbyists. It supports a wide range of microcontrollers and sensors, ideal for rapid prototyping. Extensive community support also means many libraries and examples are available.
  + **Disadvantages**: The Arduino IDE may not be suitable for more complex projects that require advanced debugging and testing features. It is also limited in its support for non-Arduino devices.
* **Eclipse IoT**: **Eclipse IoT** is a suite of tools specifically designed for IoT development, providing an open source environment that supports multiple programming protocols, platforms, and languages.
  + **Benefits**: Eclipse IoT is versatile and supports a wide range of IoT platforms and software. It provides powerful patching and simulation tools, making it suitable for large and professional IoT projects.
  + **Disadvantages**: Eclipse IoT can be too complicated for beginners. They may require large settings and configurations, which can slow down the initial stages of development.
* **Node-RED**: **Node-RED** is a flow-based development tool for visual programming, designed to connect online devices and services in new and exciting ways.
  + **Benefits**: Node-RED's visual programming interface makes it easy to connect devices and services with minimal programming. They are highly flexible and support a wide range of nodes (pre-configured functions) that can be easily integrated into projects.
  + **Disadvantages**: While Node-RED is excellent for prototyping and small applications, it may not be as efficient or scalable as traditional programming environments in large and complex IoT systems.
* **TinkerForge**: TinkerForge is an open-source and modular platform used to develop IoT devices. It allows developers to create and debug IoT systems using a variety of sensors, actuators, and microcontrollers.
  + **Benefits**: TinkerForge's modular design simplifies the process of building and testing IoT systems. They are especially useful for education purposes and small prototyping where flexibility and ease of use are crucial.
  + **Disadvantages**: TinkerForge may not be suitable for large-scale deployment due to its modular nature, which can become cumbersome when scaling. It also lacks the advanced features and integration options found in more professional IoT platforms.

**IoT hardware devices**

IoT hardware includes the physical components of IoT systems, including sensors, actuators, microcontrollers, and gateways.

* **Raspberry Pi**: The Raspberry Pi is a small, low-cost computer that can be used in a wide range of IoT applications. It can run a complete operating system, making it versatile for prototyping and deploying IoT solutions.
  + **Benefits**: The Raspberry Pi is highly flexible and can run multiple programming languages and operating systems, making it suitable for complex IoT applications that require significant processing power. It also enjoys strong community support and wide availability of accessories and expansions.
  + **Disadvantages**: The Raspberry Pi may be overpriced for simple IoT tasks that don't require a full operating system. It is also more energy-intensive compared to microcontrollers such as Arduino or ESP8266, which can be a hindrance for battery-based applications.
* **Arduino**: Arduino boards are widely used in the IoT community, especially for education and simple prototyping purposes. They are microcontroller-based platforms and provide an easy-to-use environment for creating interactive objects or environments.
  + **Benefits**: Arduino panels are simple and low-cost, and have a wide ecosystem of sensors and compatible modules. The platform is well documented and there is a large community of developers who share code and project ideas.
  + **Disadvantages**: Arduino boards are limited in processing power and memory, making them unsuitable for more complex applications in the Internet of Things. They also require additional components (such as Wi-Fi modules) to connect to the Internet.
* **ESP8266 and ESP32**: These are low-cost microcontrollers that support Wi-Fi, which are common in IoT applications to enable devices to connect to the Internet at the lowest cost and power consumption.
  + **Benefits**: ESP8266 and ESP32 are inexpensive, energy efficient, and come with built-in Wi-Fi support, making them ideal for IoT applications that require an internet connection. The ESP32 also supports Bluetooth, which adds more flexibility.
  + **Disadvantages**: Although these microcontrollers are powerful for their size and price, they can be difficult to program for beginners due to their complexity. They are also limited in processing power and memory compared to more powerful systems such as the Raspberry Pi.
* **Sensors and actuators**: These elements are essential components of any IoT ecosystem and are responsible for collecting data and interacting with the environment. Common sensors include temperature sensors, humidity sensors, motion detectors, and light sensors. Actuators are devices that can perform actions based on received data, such as turning on the light or opening the valve.
  + **Benefits**: Sensors and actuators are relatively inexpensive and widely available in various formats, making them easy to integrate into IoT projects. They are also very specific in the tasks they perform, allowing for accurate data collection and control.
  + **Disadvantages**: The quality and accuracy of sensors and actuators can vary greatly depending on the manufacturer, leading to reliability issues. In addition, integrating multiple sensors and actuators into a single system can be complex and require accurate calibration.

**Impact on the Software Development Lifecycle (SDLC) in the Internet of Things**

The Software Development Life Cycle (SDLC) is a systematic process that encompasses all stages of software development from planning and design to implementation, testing, deployment, and maintenance. In the context of the Internet of Things (IoT), this cycle is more complex due to the multiplicity of elements that must be dealt with, including physical hardware, software, networking, and cloud infrastructure. Each of these phases is heavily influenced by frameworks and programming interfaces (APIs). , tools, and selected gear devices, posing different challenges and opportunities at each stage.

**Planning Phase**

The software development lifecycle begins with the planning phase, where the project scope, objectives, and requirements are defined. In IoT projects, the selected frameworks and architecture greatly influence this stage. For example, if a cloud framework such as **AWS IoT Core** or **Azure IoT Hub is chosen**, it will simplify infrastructure setup and provide access to ready-made tools such as device management, real-time data processing, and security. This means that development teams can focus on developing logic Application instead of worrying about basic infrastructure. However, these options can cause costs to increase, necessitating careful budgeting to avoid unexpected costs. At this stage, the protocols to be used, such as **MQTT** or **CoAP, must be determined** based on the nature of the project and its requirements in terms of efficiency and bandwidth.

**Design Phase**

At the design stage, the requirements identified in the planning phase are transformed into concrete system designs. Here, selected frameworks, programming interfaces, and tools play a vital role in determining how the system is structured and its components interact. For example, using tools like **Node-RED,** which provides a stream-based visual programming interface, can greatly facilitate the design process by allowing developers to connect devices and services without the need for too much complex programming. However, this tool may be less scalable. To expand into large projects, which may lead to challenges in the future when the system needs to grow and adapt to new requirements. On the other hand, frameworks like **Eclipse IoT can** support more complex designs by offering a versatile open source environment that supports multiple protocols and platforms, giving developers the flexibility to design more complex and scalable systems.

**Implementation Phase**

During the implementation phase, system designs are converted into actual code. At this stage, tools and hardware play a pivotal role in determining how fast and efficient development is. For example, using **the Arduino IDE can** help speed up the development process for prototypes and small IoT applications thanks to its simplicity and broad support for different devices. For more complex projects, **Eclipse can be IoT** A more powerful tool, as it provides an integrated development environment that supports a wide range of programming languages and protocols. On the other hand, the choice of hardware devices such as **the Raspberry Pi** or **ESP8266 plays** an important role in determining the capabilities of the system and the efficiency of integration. **Raspberry Pi**, for example, offers significant processing power and software flexibility, making it suitable for IoT applications that require complex processing or support for multiple operating systems. In contrast, **ESP8266** provides a low-cost and energy-efficient solution for applications that require an Internet connection without the need for significant processing capabilities.

**Testing Phase**

The testing phase is crucial to ensure that the system works as expected and can meet the specified requirements. Tools used at this stage, such as **Wireshark** to monitor network traffic and**TinkerForge** to test adjustable modules, play a vital role in identifying errors and potential problems. These tools can help identify connectivity issues, test system stability, and verify data integrity between connected devices. Test systems in multiple simulated environments, using tools such as **Eclipse IoT**, can help detect performance and compliance issues early before deploying the system in a production environment. Testing also includes assessing the security of the system, where security protocols such as encryption and identity verification are applied to ensure the protection of data and users.

**Deployment and maintenance phase**

The deployment phase is the moment when the system is launched for actual use. Choosing the right framework can greatly facilitate deployment, especially if it supports continuous deployments and rolling updates. **AWS IoT Core**, for example, provides built-in capabilities for remote firmware update and large-scale device management, making it easy for businesses to deploy security updates and new features without the need for system downtime. Maintenance is a vital part of the system lifecycle, as developers have to respond to new errors, and update the system based on users' needs or new legal requirements. In this context, frameworks such as **Microsoft Azure IoT Hub** can, which provides tools for performance management and data analysis, that facilitate real-time system monitoring and rapid reaction to problems.

The software development lifecycle (SDLC) in IoT projects is arguably significantly influenced by the choices made in terms of frameworks, programming interfaces, tools, and hardware hardware. These decisions are not just technical details, they are essential elements that determine the course of the entire project, and affect every aspect of development from planning to maintenance. Through a thorough understanding of how these options affect each stage of the SDLC, businesses and developers can achieve the best results and build IoT systems that are efficient, secure, and scalable.

**Finally, the choice of frameworks, APIs, tools, and hardware in IoT is key to the development and success of IoT applications. Each of these elements offers unique features that can enhance system functionality, scalability, and efficiency, while also posing potential challenges that need careful management. The impact of these choices spans the entire software development lifecycle (SDLC), impacting the speed and efficiency of solution development, testing, deployment, and maintenance. By carefully evaluating the available options and understanding their impacts, companies can optimize their IoT strategies to build robust, scalable, and secure solutions that meet their specific needs and goals..**

**3A 1.**

**Introduction**

With the rapid development of Internet of Things (IoT) technology, unprecedented opportunities have emerged to solve complex challenges in various fields, including healthcare, transportation, agriculture, and smart cities. By connecting physical devices to the internet, IoT technology enables data collection, analysis, and systems control in real time, resulting in more efficient and intelligent systems. However, to take full advantage of the potential of IoT, it is essential to identify specific problems that IoT solutions can effectively solve. This task involves not only identifying a suitable problem but also conducting a thorough analysis of the architecture, frameworks, hardware, and technologies needed to develop an IoT application that addresses the specific problem. The ultimate goal is to choose the best combination of these elements and implement them in a detailed plan for IoT application development.

**Identify the problem**

In this task, we'll focus on a problem related to **energy management in smart homes**. With the growing popularity of smart homes, it has become very important to manage energy consumption effectively to conserve billing costs and avoid energy waste. Traditional energy management methods often rely on manual control or simple automated systems that do not provide immediate insights or the ability to adapt to the behavior of residents and the conditions of the surrounding environment. This can lead to inefficient energy use, high bills, and unnecessary waste For energy.

The problem we focus on here is **the inefficient use of energy in smart homes**, especially with regard to heating, cooling, and lighting systems. Homeowners have difficulty maintaining optimal energy consumption, as many current systems do not dynamically adapt to changes in occupancy, weather conditions, or times of day. For example, heating or cooling systems may work unnecessarily when no one is at home, or the lights may remain on. in unmanned rooms, resulting in significant energy waste over time.

An Internet of Things (IoT) solution-based solution can solve this problem by enabling real-time monitoring and control of energy-consuming devices inside the home. By integrating sensors, smart thermostats, and connected lighting systems, the IoT app can collect data on occupancy, temperature, and natural light levels. This data can then be processed to make automatic adjustments to heating, cooling, and lighting systems, optimizing energy consumption based on current conditions and habits of the population.

This problem is very suitable for an IoT-based solution because it requires communication and integration between multiple devices, and needs to collect and analyze data continuously to make immediate decisions. The goal of implementing IoT will be to reduce energy consumption in smart homes by adjusting systems to run more efficiently, ultimately reducing energy bills and reducing environmental impact.

**IoT architecture analysis**

The architecture of the Internet of Things (IoT) system is the foundation on which all components, devices, and applications are built. It defines how different elements interact, communicate, and work together to achieve the desired results. A well-designed architecture ensures smooth data flow, efficient processing, and reliable device control, which is vital for the successful implementation of any IoT solution. In the context of smart homes, where the goal is to manage energy consumption efficiently, the architecture must be robust, scalable and able to integrate various sensors and actuators. and devices that collect and process data in real time. This section will explore the key layers of IoT architecture, discuss different architectural models, and assess their suitability to address the energy management problem in smart homes.

**IoT structural layers**

The IoT architecture is usually divided into several layers, each serving a specific purpose. These layers work together to collect, transfer, process, and take action based on that data, allowing the system to operate effectively.

**A. Perception Layer**

The sensor layer, also known as the physical layer or hardware layer, is the first and most fundamental layer in the IoT architecture. This layer includes all devices and sensors that interact with the environment. In the context of smart homes, this layer consists of temperature sensors, humidity sensors, motion detectors, smart thermostats, smart lighting systems, and other devices that collect data related to energy use and environmental conditions. The sensor layer is responsible for capturing data in real time, which is is crucial to making informed decisions about energy management.

* **Main components**: sensors, actuators, smart devices, RFID tags, and embedded systems.
* **Functions**: data collection, environmental monitoring, direct interaction with physical objects.

**B. Network Layer**

The network layer, sometimes known as the communication layer, is responsible for transmitting the data collected by the sensor layer to processing systems. This layer deals with communication protocols, data transfer, and communication between devices. In a smart home IoT ecosystem, the network layer may include wireless connectivity technologies such as Wi-Fi, Zigbee, and Bluetooth that facilitate the seamless exchange of information between sensors, controllers, and cloud-based systems.

* **Main components**: routers, gateways, wireless communication modules (such as Wi-Fi, Zigbee), and network architecture.
* **Functions**: Data transfer, protocol management, and ensuring secure and reliable communication between devices.

**c. Processing Layer**

The processing layer, also known as the middleware layer, is where data from the sensor layer is collected, processed, and analyzed. This layer is vital for understanding the raw data collected by sensors and turning it into actionable insights. In smart home applications, the processing layer may include cloud computing platforms or edge computing devicesthat processes data locally to reduce latency and improve performance. This layer also includes data storage solutions and analysis platforms that help monitor and manage energy consumption patterns.

* **Main components**: cloud platforms, edge computing devices, data storage systems, and analysis engines.
* **Functions**: Data collection, processing, analysis, and storage, enabling instant decision-making and process automation.

**d. Application Layer**

The application layer is the top layer in the IoT architecture, where the processed data is used to provide services to end users. In smart homes, the application layer includes interfaces through which users interact such as mobile apps, control panels, and control systems that allow homeowners to monitor and control energy consumption. This layer also includes automation systems that perform predefined actions based on insights from the processing layer, such as adjusting the thermostat or turning off the lights in non- Occupied.

* **Main components**: user interfaces, automation controllers, mobile applications, and control panels.
* **Functions**: Provide services, enable user interaction, and perform automated actions based on processed data.

**IoT architectural models**

There are many architectural models commonly used in IoT systems, each with its own advantages and disadvantages. The choice of architectural model depends on the specific requirements of the application, such as scalability, latency, and complexity. Here are some of the most commonly used architectural models in IoT architecture:

**A. Cloud-centric architecture**

In a cloud-based IoT architecture, most data processing, storage, and management functions are performed by cloud-based platforms. Devices at the sensor layer collect data and send it to the cloud, where it is processed and analyzed. The processed data is then made available to the application layer for monitoring and control.

* **Benefits**: The cloud-based architecture offers high scalability, as cloud platforms can handle large amounts of data and support multiple devices. It also offers robust data storage capabilities and powerful processing capabilities, essential for complex analytics and machine learning applications. Furthermore, cloud-based platforms offer easy integration with other services and applications.
* **Disadvantages**: The main disadvantage of a cloud-based architecture is the latency generated by data transfer to and from the cloud. This can be an issue with applications that require an immediate response. In addition, relying on cloud services relies on internet connectivity, raising concerns about data security and privacy.

**B. Edge-centric Architecture**

Edge-based architecture addresses some of the limitations imposed by cloud-based models by bringing data processing closer to the source. In this model, edge devices (such as smart hubs or on-premises servers) process data locally, reducing the need for continuous communication with the cloud. This architecture is particularly useful in scenarios where low latency is critical, such as smart homes where immediate responses to changes in occupancy or environmental conditions are necessary.

* **Benefits**: Edge-based architecture reduces latency, as data is processed locally instead of being sent to the cloud. This allows for faster decision-making and real-time automation. They also reduce the load on the network by reducing the amount of data you need to send to the cloud, resulting in improved bandwidth usage and increased system reliability.
* **Disadvantages**: One of the main challenges of the edge-based architecture is the limited processing capacity and storage capacity of edge devices compared to cloud platforms. This can constrain the complexity of data analytics and machine learning models that can be deployed at the edge. In addition, managing and maintaining a large number of edge devices can be complex and require significant resources.

**c. Fog Computing Architecture**

Fog computing is an extension of edge computing that involves a hierarchical distribution of data processing across multiple layers, from edge devices to the cloud. In fog computing, data is processed at multiple points along the network, depending on the specific requirements of the application. For example, some data can be processed locally at the edge, while other data is sent to adjacent fog nodes or eventually to the cloud for more intensive processing.

* **Benefits**: Fog computing combines the benefits of both cloud and edge computing, providing a balanced approach to data processing. Reduces latency by processing time-sensitive data close to the source, while leveraging the cloud for more complex analytics and long-term data storage. This architecture is scalable and flexible, making it suitable for different IoT applications that require varying levels of latency and processing capabilities.
* **Disadvantages**: Fog computing can be very complex, requiring precise coordination between different layers of processing. Ensuring data flows smoothly and consistently across edge, fog, and cloud can be challenging. In addition, the cost of deploying and maintaining a fog computing infrastructure can be higher than other architectural models.

**d. Hybrid Architecture**

A hybrid architecture is a combination of cloud- and edge-based models, where certain functions are implemented at the edge while others are processed in the cloud. This approach allows IoT systems to take advantage of the advantages of both architectures, providing flexibility and improved performance based on the specific needs of the application.

* **Benefits**: The hybrid architecture provides a customizable solution that can be adjusted according to the specific requirements of the application. Enables real-time processing and low latency for mission-critical tasks at the edge, while leveraging the cloud for more resource-intensive operations. This model also provides redundancy, as some functions can be moved between the edge and the cloud based on availability and performance requirements.
* **Disadvantages**: The main disadvantage of a hybrid architecture is its complexity. Coordinating data processing and communication between edge and cloud components requires advanced management and automation tools. In addition, implementing a hybrid architecture can be resource-consuming, as it involves deploying and maintaining infrastructure in both the edge and the cloud.

**Adaptability of the IoT architecture for energy management in smart homes**

Due to the specific requirements for energy management in smart homes, it is essential to choose an IoT architecture that can process data in real time, support scaling, and ensure reliable and secure communication between devices. The following is an analysis of the suitability of different architectural models for energy management in smart homes:

**A. Cloud-based architecture**

While a cloud-based architecture offers robust processing capabilities and scalability, it may not be the best choice for smart home energy management due to the latency associated with cloud connectivity. In smart homes, where immediate responses to changes in occupancy or environmental conditions are critical, delays caused by sending data to the cloud and waiting for a response can lead to inefficiency. However, a cloud-based architecture can be useful for long-term data storage, analytics, and integration. With other smart services at home.

**B. Edge-dependent structure**

The edge-based structure is well suited for smart home energy management. By processing data locally, edge devices can make real-time adjustments to heating, cooling, and lighting systems, ensuring efficient energy use based on current conditions. Low latency and real-time response capabilities reduce the advantages of this architecture in automating energy management tasks in smart homes. In addition, it reduces dependence on an internet connection, enhancing system reliability.

**c. Fog Computing**

Fog computing offers a balanced approach that can be highly effective for energy management in smart homes. By distributing data processing across multiple layers, fog computing enables real-time decisions at the edge while leveraging the cloud for more complex analytics. This architecture can optimize real-time energy use while enabling advanced features such as predictive maintenance and long-term energy consumption analysis. However, the complexity of managing a fog computing infrastructure may be considerations for homeowners or service providers.

**d. Hybrid Structure**

The hybrid architecture provides the flexibility to meet the diverse energy management requirements of smart homes. By combining the benefits of edge and cloud processing, this architecture can ensure immediate response to mission-critical tasks while leveraging the cloud for data storage, analytics, and integration with other services. The hybrid architecture can be customized to balance performance, cost and complexity, making it a powerful choice for smart home energy management.

Choosing an IoT architecture is crucial to implementing effective energy management solutions in smart homes. While a cloud-based architecture offers robust processing capabilities and scalability, latency issues may hinder immediate decision-making in smart homes. Edge-based architecture and fog computing offer low-latency solutions that are highly suited for the immediate response required in energy management applications. On the other hand, hybrid architecture offers a flexible and customizable approach that combines edge processing benefits. and the cloud, making it a promising choice for smart homes.

**Comprehensive assessment of IoT frameworks**

In the fast-paced world of the Internet of Things (IoT), choosing the right framework is essential for the success of any project. IoT frameworks provide the foundation for the development of smart solutions, by providing tools and environments that enable developers to manage devices, process data, secure the system, and integrate with other services. Due to the complexity and diversity of IoT applications, especially in areas such as smart home energy management, choosing the right framework can significantly impact system performance, scalability, security, and effectiveness. General. This section examines some of the most popular IoT frameworks available today, focusing on their features, advantages, disadvantages, and suitability for energy management solutions in smart homes, with the integration  **of the most commonly used** Arduino **and** Raspberry Pi platforms.

**AWS IoT Core**

**AWS IoT Core** is a comprehensive and robust IoT framework provided by Amazon Web Services (AWS). It is designed to connect IoT devices to the cloud, enabling secure and reliable connectivity, data processing, and device management at scale.

**A. Benefits**

* **Scalability and flexibility**: AWS IoT Core is highly scalable, making it ideal for large applications spanning millions of devices. The framework supports a wide range of communication protocols, including MQT, HTTP, and WebSockets, ensuring diverse device compatibility.
* **Seamless integration with AWS services**: AWS IoT Core has seamless integration with other AWS services such as **Amazon S3** for storage, Amazon Lambda for serverless computing, and**Amazon DynamoDB**  for NoSQL database management.This integration allows developers to build advanced IoT applications without having to manage the underlying infrastructure.
* **Security features**: AWS IoT Core provides robust security features, such as mutual authentication, end-to-end encryption, and fine-grained access control through identity and access management (IAM). These features ensure data protection throughout its journey from device to cloud.

**B. Defects**

* **Costs**: While AWS IoT Core is strong, its cost can rise rapidly as the number of connected devices and the volume of data processed increases. This can be a major concern for startups or small businesses with limited budgets.
* **Cloud dependence**: AWS IoT Core relies heavily on continuous cloud connectivity. In scenarios where the Internet connection is unreliable or limited in bandwidth, this reliance may result in performance issues or service outages.

**Microsoft Azure IoT Hub**

**Microsoft Azure IoT Hub** is an IoT framework that provides powerful tools for device management, data processing, and security assurance across cloud and on-premises environments. Azure IoT Hub is part of the larger Microsoft Azure ecosystem, which includes services like machine learning, artificial intelligence, and advanced analytics.

**A. Benefits**

* **Comprehensive integration with Microsoft tools**: Azure IoT Hub integrates seamlessly with other Microsoft services such as Azure Machine Learning**,** Azure Stream Analytics**, and** Azure Cognitive Services**.** This integration enables the development of advanced IoT applications that rely on big data, artificial intelligence, and real-time analytics, making it particularly suitable for complex data-driven projects.
* **Hybrid cloud support**: A notable feature of Azure IoT Hub is its support for hybrid cloud environments. This enables data processing in both the cloud and on-premises, providing greater flexibility and enabling real-time processing in applications that require immediate response such as smart home automation systems.
* **Strong security and compliance**: Azure IoT Hub provides advanced security features, including per-device authentication, secure data transfer, and integration with the Azure Security Center that provides real-time threat detection and response capabilities. It also complies with a wide range of international standards and regulations, making it a secure choice for enterprise applications.

**B. Defects**

* **System** complexity: While the Azure system is powerful, it can be complex and difficult to understand for developers who are not familiar with Microsoft technologies. The learning curve can be steep, and integration between different Azure services may require specialized knowledge.
* **Cost management**: Like AWS, cost management in Azure IoT Hub can be challenging, especially as deployment expands. The pricing model can become complex, with various costs associated with data processing, storage, and additional services, which can lead to costs rising rapidly.

**Google Cloud IoT**

**Google Cloud IoT** is Google's IoT framework, which provides a powerful set of tools for connecting devices, managing them, and analyzing IoT data. The framework is designed to integrate seamlessly with Google's powerful analytics and machine learning platforms, such as **BigQuery** and **TensorFlow**.

**A. Benefits**

* **Advanced analytics and machine learning**: Google Cloud IoT is characterized by its advanced analytical and machine learning capabilities. With tools like **BigQuery**, developers can process and analyze large amounts of data quickly and efficiently. **TensorFlow enables** the deployment of advanced machine learning models on IoT data, enabling predictive maintenance, anomaly detection, and other advanced functionality.
* **Global Infrastructure**: Google's global network provides low-latency and high availability infrastructure, making it ideal for IoT applications that require consistent performance in various geographic locations.
* **Strong data security and protection**: Google Cloud IoT provides robust security measures, including end-to-end encryption, identity management, and compliance with global data protection standards such as GDPR. This ensures that IoT data is protected at all stages, from collection to processing and storage.

**B. Defects**

* **Vendor dependence**: Like other cloud frameworks, using Google Cloud IoT can lead to reliance on one vendor, making it difficult to move to another platform without much effort and reconfiguration.
* **Complexity and skills required**: Implementing and managing the advanced features of Google Cloud IoT can be complex, requiring a high level of expertise in cloud computing, data analysis, and machine learning. This complexity may be a barrier for small teams or those new to IoT development.

**Oracle IoT Cloud**

**Oracle IoT Cloud** is part of Oracle's broader cloud offering, designed to provide a comprehensive IoT solution with features such as device connectivity, data analysis, and integration with a wide range of Oracle enterprise software.

**A. Benefits**

* **Enterprise integration**: Oracle IoT Cloud excels in its ability to integrate seamlessly with other Oracle enterprise solutions such as ERP, CRM, and Supply Chain Management (SCM) systems. This makes it particularly attractive for companies that have already invested in the Oracle ecosystem, as it enables a more integrated and streamlined workflow.
* **Data management and analysis**: Oracle offers powerful data management and analytics tools, enabling organizations to process large amounts of IoT data and derive actionable insights. Oracle's cloud architecture also ensures high availability and scalability, making it suitable for large-scale deployment.
* **Security and compliance**: Oracle IoT Cloud includes comprehensive security features, including identity management, data encryption, and compliance with global regulatory standards. This ensures that IoT data is protected and that the platform meets the stringent requirements of industries such as finance and healthcare.

**B. Defects**

* **Complexity and cost**: Like other enterprise-grade IoT frameworks, Oracle IoT Cloud can be complex to implement and manage, especially for organizations with no prior experience with Oracle systems. The cost of deploying Oracle IoT Cloud can also be high, especially for small businesses.
* **Vendor dependence**: Like other cloud providers, using Oracle IoT Cloud can lead to vendor dependence, making it difficult to move to another platform in the future without a significant reconfiguration.

**أطر Arduino و Raspberry Pi**

**إطار Arduino**

**Arduino** is an open-source electronics platform based on easy-to-use hardware and software. It is known for its simplicity, making it available to beginners while remaining powerful enough for advanced users. The Arduino frame is centered around microcontroller-based panels, which can be programmed to interact with sensors, motors, lights, and other electronic components.

**Features and capabilities**

* **Ease of use**: One of the most important advantages of Arduino is its user-friendly nature. Arduino's integrated development environment is simple, allowing users to write code in Simplified C++ and upload it directly to the board with a single click.
* **Extensive library support**: Arduino comes with a wide range of libraries that simplify complex tasks such as controlling motors, reading sensors, or connecting to the Internet. These libraries are constantly developed and maintained by a large community, making it easier for developers to perform complex functions with little effort.
* **Modularity**: Arduino's ecosystem has a highly modular nature, with a variety of additional chips (shields) that can be connected to the main board to expand its capabilities, such as adding Wi-Fi connectivity, controlling motors, or tracking GPS locations.
* **Broad community and support**: Arduino has a huge global community that contributes to an ecosystem rich in tutorials, model projects, and forums. This makes it easy for beginners to get started and for advanced users to find solutions to certain problems.

**Advantages**

* **Low cost**: Arduino panels are usually inexpensive, making them available to amateurs, students, and teachers.
* **Open source hardware and software**: Being open source, hardware and software designs are available free of charge, allowing customization and adaptation to meet specific needs.
* **Prototyping**: Arduino is ideal for rapid prototyping. Its conductive and operational nature allows developers to quickly build and test ideas without the need for deep knowledge of electronics.

**Defects**

* **Limited processing power**: Because Arduino boards rely on microcontrollers, they have limited processing capacity and memory compared to more powerful platforms such as the Raspberry Pi. This makes them less suitable for complex tasks that require large computing resources.
* **Absence of an operating system**: Unlike the Raspberry Pi, Arduino does not support running an operating system, limiting its ability to handle multitasking and complex software environments.

**Relevance of IoT projects**

Arduino is highly suitable for IoT projects that require simple direct control of hardware components. It is especially effective in scenarios where low power consumption is necessary, such as battery-powered sensors in a smart home environment. For example, Arduino can be used to build smart thermostats, lighting control systems, or environmental monitoring systems that communicate with a hub or cloud service. However, for tasks that require more processing power or integration Complex software, Arduino may need to be paired with more powerful systems.

**إطار Raspberry Pi**

**The Raspberry Pi** is a series of pluggable minicomputers developed by the Raspberry Pi Foundation. Unlike Arduino, which relies on microcontrollers, the Raspberry Pi is an all-in-one computer running a Linux-based operating system. This makes the Raspberry Pi a powerful tool for more complex IoT projects that require significant computing power, networking, and software integration.

**Features and capabilities**

* **Complete operating system**: Raspberry Pi runs on **Raspberry Pi OS** (formerly known as Raspbian), a Linux-based operating system. This allows it to multitask at the same time, run full-scale applications, and even work as a desktop computer.
* **Networking capabilities**: The Raspberry Pi has built-in networking capabilities such as Ethernet and Wi-Fi, making it ideal for projects that require strong networking and internet connectivity. The Raspberry Pi can act as a web server, network controller, or hub in an IoT system.
* **GPIO Ports**: Similar to Arduino, the Raspberry Pi has general-purpose input/output (GPIO) ports, which allow it to interact with a wide range of sensors, motors, and other electronic components. These ports can be programmed using various programming languages, including Python, making it flexible for different uses.
* **Multimedia support**: The Raspberry Pi can handle multimedia tasks such as video streaming, image processing, and audio playback. This makes it suitable for projects such as smart home entertainment systems or video surveillance.

**Advantages**

* **High processing power**: Compared to Arduino, the Raspberry Pi offers much higher processing power, allowing it to handle complex tasks such as data processing, machine learning, and real-time analytics.
* **Flexibility**: Raspberry Pi's ability to run a complete operating system and compatibility with a wide range of programming languages makes it suitable for a variety of IoT applications, ranging from home automation to industrial control systems.
* **Broad community and support**: Like Arduino, the Raspberry Pi enjoys the support of a large community of users and developers. There is an abundance of tutorials, forums, and projects available, which can help beginners and professionals alike.
* **Affordable**: Despite its powerful features, the Raspberry Pi remains affordable, making it available for educational purposes and commercial IoT projects.

**Defects**

* **Higher power consumption**: The Raspberry Pi consumes more energy than Arduino, which can be a hindrance in battery-powered IoT projects or energy-sensitive projects.
* **Complexity**: The flexibility and power of the Raspberry Pi comes with added complexity. It requires more setup and configuration compared to Arduino, and managing a complete operating system can be overkill for simple projects.

**Relevance of IoT projects**

The Raspberry Pi is highly suited for IoT projects that require more computing power, such as home automation control centers, media servers, or data analysis nodes in a smart home environment. Its ability to run complex applications, connect to multiple networks, and process large amounts of data makes it ideal for more complex IoT bulletins. For example, the Raspberry Pi can be usedTo control and monitor a home security system, process video streaming from cameras, or manage a network of home smart devices.

**Integrating Arduino and Raspberry Bay into IoT projects**

In many IoT applications, Arduino's and Raspberry Pi's strengths can be leveraged together to create more robust and resilient systems. While Arduino excels at direct device control and low-power operation, Raspberry Pi brings the computing power needed for complex tasks and network management.

For example, in a smart home energy management system, Arduino can be used to manage individual sensors such as temperature or lighting sensors, which monitor the environment and transmit data to the central Raspberry Pi. Raspberry Pi can then process this data, analyze it, and make decisions, such as adjusting the thermostat or turning off lights in unoccupied rooms. This combination allows the system to be both energy efficient and robust, enhancing resource utilization while providing features and capabilities Advanced.

By using Arduino and Raspberry Pi together, developers can create IoT solutions that are more than just the sum of their parts. This approach benefits from the simplicity and low cost of Arduino for hardware-related tasks, while leveraging the power and flexibility of Raspberry Pi for software-related tasks and system management.

**The end**

Choosing the right framework for IoT is a crucial decision that can significantly impact the success of an IoT project, especially in complex applications such as energy management in smart homes. Each of the frameworks mentioned presents unique strengths and weaknesses, and the choice depends on the specific needs of the project.

**AWS IoT Core** is ideal for large cloud-based applications thanks to its robust integration capabilities and strong security features, although it comes at a higher cost and cloud infrastructure. **Microsoft Azure IoT Hub** offers a comprehensive, hybrid approach that supports both cloud and on-premises processing, making it a solid choice for projects that require real-time response and integration with Microsoft's broader ecosystem.

**Google Cloud IoT** excels at analytics and machine learning, providing powerful tools for data-driven applications, but it can face vendor dependency and complexity challenges. **IBM Watson IoT** stands out thanks to AI-driven insights and industry-specific solutions, making it suitable for high-value, high-end applications, though its complexity and cost can be a drag on smaller projects.

Open source frameworks like **ThingsBoard** and **Eclipse IoT provide** cost-effective and customizable options, but require greater investment in development and security management. **Oracle IoT Cloud** and **PTC ThingWorx provide** robust enterprise-grade solutions with robust integration capabilities, though their complexity and cost may limit their attractiveness to larger organizations that are already investing in their ecosystems.

In smart home energy management, a hybrid approach that combines scalability and advanced analytics for a cloud-based framework like **Google Cloud IoT** with customization of open source tools can be the best balance between performance, flexibility, and cost. The integration  **of Arduino** and **Raspberry Pi** within this framework can enhance the efficiency and strength of the system, ensuring that the IoT solution is effective and comprehensive. The following steps require selecting the most appropriate framework and integrating it into a comprehensive IoT solution Meets the specific energy management needs of smart homes.

**The optimal choice of devices in IoT solutions**

When developing an IoT solution, especially one dedicated to smart home energy management, it is essential to carefully select hardware components to ensure the perfect balance between functionality, efficiency and cost. The goal is to create a system capable of collecting and processing data effectively, reliable, scalable, and easy to integrate with existing infrastructure. This section focuses on selecting the best hardware components that suit the needs of a smart home energy management system.

**Ideal IoT sensors**

For a smart home energy management system, **heat sensors** and light sensors are the most important components. **Heat sensors** such as **DS18B20** monitor the temperature in various areas of the home, allowing the system to make real-time adjustments to heating and cooling systems. This ensures efficient energy use, while maintaining comfort without unnecessary consumption. Features **a DS18B20** sensorWith its high resolution and digital output, it facilitates integration with microcontrollers.

**Light sensors** such as **photoresistive LDR** are essential for automating lighting systems. By measuring the intensity of natural light, these sensors can adjust indoor lighting to suit the actual need, contributing to reduced energy waste. **LDR sensors are** simple and effective, making them an excellent choice for this application.

**Ideal IoT Triggers**

In a smart home energy management system, **electric actuators** such as **an electric relay are**  essential to control devices such as lights, heating and ventilation systems, air conditioning, and other devices. Relays allow the system to turn on and off high-power devices in response to central console commands, making them essential in automating energy use throughout the home.

In addition, **servo motors can be used**  for precise control tasks, such as adjusting curtains to control the amount of natural light entering the home. This helps in improving the use of lighting and contributes to maintaining the ideal temperature inside the home, reducing the need for heating or cooling.

**Ideal connectivity modules for IoT**

For connectivity within a smart home energy management system, a  **Wi-Fi** ESP8266 module is an ideal choice. This module enables seamless connectivity to other devices on the home network as well as to cloud services for data storage and analysis. The ESP8266 unit is energy efficient, which is critical to maintaining the overall system efficiency.

In cases where the system needs to cover more space or connect to multiple devices in a network of wipes, **Zigbee** modules such as **XBee provide** an effective solution. Zigbee is characterized by low power consumption and its ability to support mesh networks, making it ideal for smart home applications that require reliability and wide range.

**Microcontrollers and processors ideal for the Internet of Things**

For processing and system control purposes, the **ESP32 microcontroller is** an excellent choice. This controller provides a balance between processing power, connectivity and energy efficiency, making it ideal for handling various tasks within a smart home energy management system. **The ESP32**  features built-in Wi-Fi and Bluetooth, allowing it to effectively manage multiple devices and sensors while maintaining connectivity to cloud services.

For more complex tasks or systems that require greater computing power, such as those that integrate artificial intelligence for predictive power management, a single-board **Raspberry Pi**  is an excellent choice. **The Raspberry Pi can** run a complete operating system, manage data analytics, and act as the main hub for the entire home's intelligent system.

Choosing the right devices is key to building an efficient energy management system in smart homes. By focusing on components such as a temperature DS18B20 sensor, LDR sensor for light, electric relay for control, ESP8266 for communication, and ESP32 or Raspberry Pi for processing, developers can create an efficient and reliable system. These components provide the perfect balance between performance, energy efficiency and ease of integration, ensuring the smooth operation of the smart home system And meet the requirements of modern energy management.

**Evaluation of tools used in IoT development**

**منصة PlatformIO**

**PlatformIO** is an integrated development environment (IDE) dedicated to developing IoT applications. The platform supports a wide range of platforms such as Arduino, ESP32 and Raspberry Pi, making it a versatile tool for IoT projects. PlatformIO facilitates development by providing a unified environment in which developers can write code, correct errors, and deploy across multiple platforms.

**Usage**:

* **Cross-platform development**: PlatformIO allows developers to write code once and deploy it across multiple devices, which is extremely useful in IoT projects where different devices (such as sensors and microvectors) need to work together seamlessly.
* **Library Management and Accreditations**: The platform provides a robust library management system that simplifies the process of embedding and updating libraries, ensuring that projects remain consistent and manageable.
* **Continuous integration**: PlatformIO integrates with CI/CD lines, allowing for automated testing and continuous deployment, which is essential to sustaining large IoT projects.

**Benefits**:

* **Ease of use**: PlatformIO's user-friendly interface and extensive support for different platforms make it suitable for novice and experienced developers alike.
* **Scalability and customization**: The platform supports a wide range of plugins and plugins, allowing developers to customize the development environment according to the needs of their projects.
* **Strong community support**: PlatformIO has a large and active community that provides a wealth of resources, including libraries, plugins, and forums for problem solving and knowledge sharing.

**MQTT.fx tool**

**MQTT.fx** is a tool designed to test and monitor MQTT (Message Queuing Telemetry Transport) connections, a lightweight messaging protocol widely used in IoT applications. MQTT is effective in environments with limited bandwidth and resources, making it ideal for devices that need to transfer small amounts of data over networks with limited capacity.

**Usage**:

* **Device communication simulation**: MQTT.fx allows developers to simulate the behavior of IoT devices by publishing and subscribing to MQTT topics. This is indispensable during the testing phase, as developers can ensure that the devices are sending and receiving data according to system requirements.
* **Broker testing**: The tool can be used to contact MQTT brokers and test their performance in different scenarios. Developers can monitor how the broker handles multiple clients, different message sizes, and topic subscriptions, to ensure that the broker is strong enough for the intended application.
* **Debugging**: MQTT.fx provides detailed records of all MQTT transactions, including timestamps, message contents, and topics. This information is necessary to correct errors related to the delivery of messages, the structure of the topics or the communication with the broker.

**Benefits**:

* **Real-time monitoring**: MQTT.fx offers real-time visualization of message exchanges, allowing developers to see the impact of their actions instantly. This instantaneous interaction is essential for repeated testing and tuning of IoT systems.
* **User-friendly interface**: The tool has an intuitive interface that makes it easy to set up and manage MQTT connections, even for developers who don't have much experience with the MQTT protocol.
* **Flexible testing capabilities**: MQTT.fx supports a wide range of MQTT configurations, including QoS levels, retained messages, and clean session settings. This flexibility allows developers to comprehensively test their systems under different conditions.

**Wireshark Tool**

**Wireshark** is an open source network protocol analysis tool used to capture and analyze traffic passing through the network. It is an indispensable tool for diagnosing network problems, especially in IoT environments where understanding the flow of data between devices is critical to ensuring smooth operation.

**Usage**:

* **Network traffic capture**: Wireshark allows developers to capture all network packets traveling between IoT devices and other entities in the network. This is necessary to identify connection patterns, detect anomalies, and understand the protocols used.
* **Protocol analysis**: The tool supports detailed analysis of a wide range of protocols, including HTTP , MQTT, and CoAP. This capability is especially valuable in IoT systems that rely on multiple protocols to work, where developers can verify that each protocol works as expected and that data travels securely.
* **Security Review**: Wireshark can be used to monitor IoT networks for potential security threats, such as unauthorized access or data leaks. By analyzing network traffic, developers can detect suspicious activity and take preventive measures to protect sensitive data and maintain the integrity of the IoT system.

**Benefits**:

* **Comprehensive protocol support**: Wireshark provides extensive protocol support, making it a versatile tool that can be used in almost any IoT environment, regardless of the protocols used.
* **Deep packet analysis**: The tool provides accurate details about each packet captured, including source and destination IP addresses, protocol type, payload content, and more. This level of detail is necessary to correct complex network problems or improve its performance.
* **Community and Documentation**: Being an open source tool, Wireshark is supported by a large community of developers and network engineers who contribute to its continuous development. The availability of comprehensive documentation, tutorials, and community support makes it easy for users to learn how to use the tool effectively and solve any problems they face.

**Postman Tool**

**Postman** is a powerful API development and testing tool that allows developers to design, test, and document APIs. In the IoT context, Postman is particularly useful for testing REST APIs, which are commonly used to facilitate communication between IoT devices and cloud services.

**Usage**:

* **Simulation API requests:**  Postman enables developers to simulate API requests, including GET, POST, PUT, DELETE, and more. This allows developers to test how their devices interact with APIs, ensuring that devices can properly send and receive data under different conditions.
* **Automated testing**: Postman supports automated testing through its scripting capabilities, allowing developers to create test suites that can run automatically. This is especially valuable in CI/CD environments, where APIs must be regularly tested to ensure that they remain functional as the IoT ecosystem evolves.
* **API documentation**: Postman can generate documentation based on API tests and requests created by developers. These documents can then be shared with other team members or stakeholders, providing a clear and up-to-date reference for the functionality and use of APIs.

**Benefits**:

* **Ease of use**: Postman's GUI is designed to be easy to use, allowing developers to create and test APIs without requiring deep programming knowledge. This facilitation makes it an essential tool for both beginners and experienced developers.
* **Comprehensive testing capabilities**: Postman supports a wide range of test scenarios, including performance testing, security testing, and regression testing. This flexibility ensures that developers can accurately validate their APIs before deploying them to the production environment.
* **Collaboration features**: Postman offers collaboration tools that enable teams to work together on developing and testing APIs. This includes co-working spaces, version control, and the ability to comment on specific API requests, making it easier to coordinate efforts and ensure everyone is on the same page.

**منصة ThingsBoard**

**ThingsBoard** is an open source IoT platform that provides tools for device management, data collection, processing, and visualization. It is designed to help developers create, deploy, and manage IoT applications more efficiently by providing a comprehensive set of features in a single platform.

**Usage**:

* **Device management**: ThingsBoard allows developers to manage a large number of IoT devices through a central control panel. This includes monitoring device health, firmware updates, and configuring device parameters, all of which are necessary to maintain the health and performance of the IoT system.
* **Data collection and** processing: The platform supports the collection and processing of data from multiple devices in real-time. Developers can define custom data processing flows, set up rules to incentivize actions based on certain conditions, and aggregate data for analysis.
* **Data visualization**: ThingsBoard offers a wide range of tools for visualizing IoT data, including customizable dashboards, graphs, and maps. This enables developers and stakeholders to monitor system performance, track key metrics, and derive insights into the operation of IoT solutions.

**Benefits**:

* **Scalability**: ThingsBoard is designed to be able to scale easily, making it suitable for small projects and for large-scale deployment. Its ability to handle thousands of devices and millions of data points ensures that it can grow alongside the IoT ecosystem it supports.
* **Flexibility and customization**: The platform's modular architecture allows developers to customize its functionality to meet the needs of their IoT applications. This includes the ability to integrate with other tools and services, such as cloud platforms, databases, and algorithms for machine learning.
* **Cost-effectiveness**: As an open-source platform, ThingsBoard is free to use, making it an attractive option for developers and organizations seeking to reduce costs while leveraging a powerful IoT platform.

**تقنيات API: SOAP API vs. REST API**

**SOAP API**

**SOAP** is a protocol that uses XML to encode messages. It is known for its stringent standards and high levels of security, making it suitable for applications that require high data integrity and security, such as financial services or healthcare systems.

**When to use the SOAP API:**

* **High security and reliability**: SOAP is often chosen in environments where security and reliability are top priorities. For example, in financial services, healthcare, or any field that requires high data integrity and secure communications, SOAP provides built-in WS-Security standards for end-to-end encryption and message authentication.
* **Complex operations**: SOAP is also suitable for applications that require strong support for complex operations, such as those that involve multiple steps or need advanced error handling. SOAP supports features such as WS-Security, which ensures message encryption and authentication.

**How SOAP API works**

* **Message structure**: A typical SOAP message consists of a wrapper (specifying the beginning and end of the message), a header (for metadata), an object (where the actual data or payload is located), and an error element (for error handling).
* **Transport layer**: While HTTP/HTTPS is the most common transport layer for SOAP, it can also be used with other protocols such as SMTP. Using multiple transport layers is one reason why SOAP is flexible, but it also makes its implementation more complicated.
* **Security**: SOAP provides built-in support for various security standards, including SSL/TLS for transport-level security and WS-Security for message-level security. This dual security approach ensures that even if the transport layer is compromised, the content of the message remains protected.

**Advantages of SOAP API**

* **Strength and security**: SOAP supports complex security models, including WS-Security, making it highly secure. It is also robust in processing transactional processes, ensuring that all parts of the multi-step process are successfully completed.
* **Modularity**: SOAP adheres to stringent standards, ensuring that APIs built with SOAP will continuously work across different platforms and systems, which is critical in environments where interoperability is essential.
* **Built-in error handling**: The error element in SOAP provides a standard way to handle errors, making it easier to debug and manage API failures in an organized manner.

**عيوب SOAP API**

* **Complexity**: Strict standards and SOAP's reliance on XML make it more complex and more resource-consuming compared to other API technologies. This can lead to longer development times and higher computational costs, especially in resource-limited IoT environments.
* **Overload**: Using XML to format messages and additional layers of security and error handling increases message sizes and processing requirements. This overload can be a significant drawback in low-bandwidth IoT applications.

**REST API**

**REST** is an architectural pattern that uses standard HTTP methods to interact with resources, which are defined by URIs (Uniform Resource Identifiers). Unlike SOAP, REST does not rely on a specific protocol, and takes advantage of existing web standards to facilitate communication. REST is known for its simplicity, flexibility, and scalability, making it the preferred choice for most modern IoT applications.

**When to use the REST API**

* **Scalability and flexibility**: REST is ideal for IoT applications that require scalable solutions capable of processing a large number of requests. The stateless nature of REST allows for horizontal expansion, as additional servers can be added to handle overload without affecting system performance.
* **Web and cloud integration**: REST APIs are particularly suitable for web-based services and cloud-based IoT applications. These APIs are compatible with modern web technologies such as HTML, JSON, and JavaScript, making it easy to integrate with modern web, mobile, and cloud applications.
* **Light and efficient connectivity**: REST is preferred in scenarios that require a light and efficient connection, such as mobile devices or embedded IoT devices. The ability to return data in formats such as JSON (which is more compact than XML) allows for reducing the bandwidth required for communication and speeding up processing.

**How REST API works**

* **Stateless connection**: In REST, each client-to-server HTTP request must contain all the information needed to understand and process the request. This stateless state means that the server does not need to store client state between requests, making REST highly scalable.
* Standard HTTP methods: REST APIs use standard HTTP methods such as GET, POST, PUT, and DELETE. These methods are compatible with CRUD (Create, Read, Update, Delete) operations, making REST easy to understand and implement.
* **Flexible data formats**: While REST usually returns data in JSON format, it can also support XML, plain text, or other formats depending on the needs of the application. This flexibility allows REST APIs to be used in a variety of contexts.

**Advantages of REST API**

* **Simplicity and speed**: REST is simpler and faster to implement than SOAP. Its use of standard HTTP methods and stateless communication means it can handle a large number of requests with minimal overload, making it ideal for high-performance applications.
* **Flexibility**: The ability to return data in different formats and use URIs to identify resources allows great flexibility for REST. This adaptability is essential in IoT environments where different devices and services may require different data formats.
* **Broad adoption and compatibility**: REST is widely supported across the industry and is supported by almost all programming languages and frameworks. This broad compatibility ensures that REST APIs can be easily integrated into existing systems.

**عيوب REST API**

* **Security**: While REST can be secured with HTTPS, it does not have built-in security features such as SOAP. Developers need to implement additional security measures (such as OAuth and JWT) to protect REST APIs, especially in applications where data sensitivity is a concern.
* **Lack of standardization**: Unlike SOAP, REST does not enforce strict standards, which can lead to discrepancies in how APIs are implemented. This lack of modarity can lead to challenges in ensuring that different REST APIs work together seamlessly, especially in complex IoT systems.

**The end**

The choice between SOAP and REST APIs in IoT development depends on the specific requirements of the application. **SOAP is** best suited for environments where security, reliability, and compliance are essential, such as financial or medical applications. Built-in error handling and built-in security features make it ideal for complex transactional processes. However, its complexity and overload may be a hindrance in resource-limited IoT environments.

On the other hand, **REST offers** simplicity, flexibility, and scalability, making it the preferred choice for most modern IoT applications. REST's stateless nature and compatibility with web technologies allow for efficient connectivity and easy integration with cloud and web services. However, developers need to implement additional security measures to protect REST APIs in sensitive applications.

By understanding the strengths and limitations of each API technology, IoT developers can make informed decisions aligned with their project needs, ensuring efficient and secure connectivity across their IoT systems.

**IoT Application Development Detailed Plan**

Internet of Things (IoT) application development is a complex process that requires careful planning and consistent implementation to ensure success. A detailed plan consists of several key stages, from identifying and analyzing requirements, to designing the system, selecting the right components, and developing and testing the prototype. We will go through these stages in great detail to ensure that all the essential aspects of creating an effective energy management system are covered in smart homes.

**Requirements identification and analysis**

The first crucial step in developing an IoT application is to identify and analyze the requirements. This includes understanding the main objectives of the system and the problems it seeks to solve. In the case of a smart home energy management system, developers should start by defining functional requirements such as real-time monitoring of energy consumption, automatic control of electrical appliances to optimize energy consumption, and providing alerts to users about unusual energy usage or opportunities to improve efficiency. This requires a deep analysis of users' needs, which helps in identifying Basic features that the system should include. In addition, non-functional requirements must be taken into account, such as security to ensure the protection of sensitive user data, scalability to provide future system expansion capabilities, and ease of use to ensure that the system is intuitive and easy to interact with by all users, regardless of their level of technical expertise. This phase forms the basis on which all subsequent phases will be based, as a clear and comprehensive understanding of the requirements of the system ensures that all future efforts will be directed towards a common goal.

**System architecture design**

Based on the requirements that have been identified and analyzed, comes the design stage of the architecture of the system. The architectural architecture determines how tasks are distributed among the different components of the system and how these components interact with each other. For a smart home energy management system, a three-level architecture can be the perfect choice. This architecture includes three main levels: Edge Devices, Gateway Layer, and Cloud Layer. In the hardware layer, the main task is to collect data from various sensors such as heat, light, and motion sensors deployed in the house. These devices must be low in power consumption and able to process data locally to ensure energy consumption savings and reduce load on the network. In the gateway layer, raw data is processed and unnecessary information filtered out before important data is sent to the cloud. This layer should be robust enough to support edge computing, helping to improve latency And reduce the burden on the cloud. In the cloud, data is stored and processed more complex using techniques such as machine learning to analyze energy usage patterns and make recommendations for improving efficiency. Such an architectural design ensures a perfect balance between performance, security, and scalability.

**Selection of hardware and software components**

Once the architecture is designed, it is the stage of selecting the appropriate hardware and software components that support that design. Sensors that ensure accurate and reliable data collection should be chosen, such as high-resolution DS18B20 temperature sensors and LDR light sensors that allow interaction with ambient light effectively. Microcontrollers such as the ESP32 are used due to their multiple data processing capabilities and support for wireless communication via Wi-Fi and Bluetooth, making it ideal for smart home applications. In addition, a powerful gateway like the Raspberry Pi is chosen that enables raw data processing and device control locally before moving data to the cloud. On the software side, the ThingsBoard framework is used for device management and data processing, which enables the handling of a large number of devices and provides powerful tools for data visualization and analysis. Software tools such as PlatformIO andMQTT.fx Essential for ensuring smooth development and effective debugging, while tools like Wireshark contribute to analyzing network traffic and ensuring that the system operates efficiently and securely.

**Prototype development and testing**

After selecting the appropriate hardware and software components, the development and testing phase of the prototype begins. This stage requires the creation of an initial version of the system that allows performance evaluation and verification that all components work in harmony. The prototype is developed in stages, starting with establishing a simple connection between sensors and microcontrollers to verify that data is collected correctly. Next, more complex functions are gradually added, such as automatic control of devices based on data received from the sensors. During this phase, Use tools like Postman to test application programming interfaces (APIs) to ensure that communication between different components of the system works properly and effectively. Wireshark is also used to analyze data traffic over the network and ensure that data is transmitted securely and efficiently between devices and the cloud. Besides, performance tests are performed to verify the system's ability to handle a large number of devices and process data effectively without affecting the overall performance of the system. These tests help in Identify and improve any potential vulnerabilities before moving to the actual deployment phase.

**Improve performance and cloud integration**

After the prototype is developed and tested, comes the performance optimization phase and the integration of the system with the cloud. This includes making improvements to the system to make it more efficient and responsive to changing needs. The energy efficiency of connected devices must be improved, and data analysis algorithms must be developed to provide accurate forecasts and automatic improvements in energy consumption. At this stage, the system is integrated with cloud services, allowing users to access their data and manage their devices from anywhere using sophisticated user interfaces. This includes mobile app development Mobile and web interfaces that provide users with a seamless experience in monitoring and controlling home power systems. Focusing on improving performance at this stage ensures that the system will be able to cope with the increase in the number of connected devices and provide a reliable and sustainable service to users.

**System deployment and maintenance**

Once the system is optimized and made sure it works efficiently, the system is deployed in a real environment. During this phase, users are trained on how to use the system and monitor its performance continuously to ensure that it meets expectations and provides the required benefits. Regular system maintenance plans, including software and hardware updates, and security monitoring are also prepared to ensure the system is protected from cyber threats. In addition, periodic system evaluations are conducted to identify any required improvements or opportunities. Extends the system to new hardware or additional functions. Regular maintenance and constant updates play a vital role in ensuring the sustainability of the system and achieving optimal performance in the long term.

**The end**

IoT application development requires a detailed and multi-faceted plan to ensure the success of the system and achieve the desired benefits from it. By following specific steps starting from defining requirements, through designing the architecture, selecting the right components, developing and testing the prototype, optimizing performance and integration with the cloud, all the way to deploying and maintaining the system, an efficient and sustainable energy management system can be achieved in smart homes. This detailed plan contributes to ensuring that all aspects of the system are carefully designed and implemented to achieve optimal performance and provide a user experience Excellent.

**A 1.4**

**Hub-and-Spoke architecture in IoT systems for smart homes**

**Hub-and-Spoke architecture overview**

Hub-and-Spoke **architecture** is a centralized communication model, where all devices (spokes) connect to a hub that collects data, processes it, and resends commands to devices. In a smart home IoT ecosystem, the hub acts as the main controller that receives data from various IoT devices (such as thermostats, smart cameras, smart locks, and smart bulbs) and processes them and then sends commands to the appropriate devices. The hub A device like the Raspberry Pi or a system like Google Nest Hub or Amazon Echo, equipped with software capable of processing data and issuing instructions accordingly. For example, a hub can receive temperature data from a specific sensor and adjust the air conditioning unit based on user preferences. This model simplifies device management in a smart home environment by centralizing control and making all communications pass through a single point.

The main strength of the Hub-and-spoke model lies in its centrality. With a central hub that manages all connections, troubleshooting and system management is much easier. Instead of having devices communicate with each other in a complex network, the devices only communicate with the hub, which handles all the data. This approach works well in environments such as smart homes where there is a moderate number of devices that you need to control centrally without the complexities of a distributed network. In While this approach provides ease of use, it comes with some downsides such as the possibility that the hub is a central vulnerability, or the need for a large computational capacity within the hub to handle a large number of devices.

**Frameworks in Hub-and-Spoke architecture**

* **ThingsBoard**: **ThingsBoard** is an open source framework for IoT management, highly flexible and scalable, that can be used as a hub kernel in Hub-and-Speak architecture. It allows easy device management, data collection from multiple devices (spokes), and control of these devices in real time through a centralized control panel. One of its biggest advantages is that it supports several IoT protocols such as MQTT and CoAP, which means that devices using different communication protocols can communicate seamlessly with the hub. In the case of smart homes, ThingsBoard can be run on the hub to collect data from heat sensors, smart bulbs, and security cameras, while at the same time processing this information to provide useful reports and insights to the homeowner. Its ability to scale is another advantage, as it can manage hundreds of devices efficiently, allowing the system to grow according to need.
* **Home Assistant**: **Home Assistant** is another excellent hub-based IoT framework in smart homes. Home Assistant acts as the central mind of the smart home system, integrating with hundreds of different devices and services. Home Assistant excels at fully home automation by offering integrations with popular smart home systems like Google Home, Amazon Alexa, and Apple HomeKit. With Home Assistant, the hub controls all automation and monitoring, allowing users to create rules such as turning off all lights when motion is not detected for a certain period, or adjusting the thermostat based on the time of day. Powerful automation engine allows users to define complex rules without the need for programming skills Advanced.

**Tools in Hub-and-Spoke Architecture**

* **Node-RED**: **Node-RED is an**  open-source, flow-based tool for developing IoT systems. It helps build automation logic and design interactions between devices in a smart home environment within a Hub-and-Speaker architecture. Node-RED simplifies the process of building automation logic by allowing users to connect "nodes" that visually represent devices and procedures, and then deploy them to the central hub. In a smart home system, users can easily create flows where the motion sensor activates lighting or sends notifications when abnormal activity is detected. Node-RED supports various protocols such as MQTT and HTTP, making it a flexible tool for building custom workflows.
* **MQTT.fx**: **MQTT.fx** is a necessary tool for testing and managing MQTT communications in Hub-and-Spoke-based systems. MQTT is a popular choice in the Internet of Things because of its lightness, making it ideal for transferring small data from sensors to the hub. MQTT.fx can help developers test the data that is transferred between devices and the hub. With MQTT.fx, developers can subscribe to certain topics, view the data sent by devices, and ensure that the connection is made correctly.

**Hardware in Hub-and-Spoke architecture**

* **Raspberry Pi**: **Raspberry Pi** is a popular hardware platform for use as a hub in IoT systems. The Raspberry Pi provides an affordable and energy-efficient solution to act as a central controller in a smart home. Open source systems such as Home Assistant or ThingsBoard can run on the Raspberry Pi, allowing it to manage connections between all connected devices. Thanks to Wi-Fi supportWith built-in Bluetooth, the Raspberry Pi can easily connect to smart home devices such as lamps, thermostats, and smart locks.
* **Zigbee Hub**: **Zigbee hubs** is another popular option as a hub in smart home systems. Zigbee is a network protocol that uses low energy, through which communication between IoT devices can be made. Zigbee is characterized by its ability to form interlocking networks, where devices communicate not only with the hub but also with each other, increasing the network band.

**Advantages of Hub-and-Spoke architecture**

1. **Centralized control**: The main advantage of the Hub-and-spoke model is the centralized nature of control. By having a central hub, it becomes easy to manage and monitor the system. Users can easily control devices from a single point, as the hub handles data processing from all linked devices, making it easy to make decisions quickly. This centralized nature provides a simplified user experience, especially in smart homes that have a large number of devices such as smart lighting, sensors, and heating and ventilation systems.
2. **Easy troubleshooting**: Since all connections go through the hub, detecting problems in the system becomes simpler. If a particular device stops working or a connection malfunction, the problem can be quickly identified by tracking the data that reaches the hub. This means that users do not need to look for malfunctions in each device separately, but it is enough to check the hub and determine which device is facing the problem. This contributes to reduced maintenance time and repairs, especially in large systems with multiple devices.

**Client-Server Architecture in IoT Systems for Smart Homes**

**Client-Server architecture overview**

Client-Server **architecture** is a common communication model where devices (clients or clients) communicate with a central server that handles their requests and provides the necessary data or commands. In the context of an IoT system for smart homes, the server is typically located either in the cloud or on a on-premises server (such as a Raspberry Pi), while the customer devices are IoT devices such as sensors, smart bulbs, security cameras, and thermostats. The server manages the entire system, processing data from clients, and sending appropriate responses or commands. For example, a client sends temperature readings to the server, and based on the data received, the server decides whether to adjust the heating or cooling system. In this architecture, the server acts as a central repository of data and logic, while client devices remain relatively simpler, sending data and waiting for instructions.

The strength of the Client-Server model is its ability to distribute tasks between the server and clients. The server handles most of the processing and decision-making, allowing customers to stay simple and energy efficient. This separation of tasks is especially useful in IoT systems such as smart homes, where many devices need to communicate and exchange information without overloading the system. The server can also host advanced analytics, automation rules, and provide a centralized interface for monitoring and control through web applications or phones. Smart. However, like any other architecture, the Client-Server model has advantages and disadvantages, and we will review them in detail below.

**Frameworks in the Client-Server architecture**

* **Microsoft Azure IoT Hub**: **Azure IoT Hub** is a cloud-based solution offered by Microsoft that fits perfectly into the Client-Server model. It acts as a server in this model, enabling secure communication between customers (IoT devices) and the cloud. This framework supports several IoT protocols such as MQTT, HTTPS, and AMQP, making it flexible and compatible with various smart home devices. Azure IoT Hub allows the server to ingest large amounts of data from multiple devices, process this data, and send appropriate responses. In a smart home system, Azure IoT Hub canCollect data from temperature sensors, analyze them, and activate automation processes such as adjusting the HVAC system based on predefined conditions. A big advantage of using Azure IoT Hub is its ability to scale, as it can manage millions of devices, making it suitable for both small home systems and larger IoT networks.
* **AWS IoT Core**: Another excellent framework in IoT Client-Server architecture is  **Amazon' s** AWS IoT Core. AWS IoT Core acts as a server, allowing devices (clients) to securely connect, interact, and process data in the cloud. Both the MQTT and HTTP protocols are supported, enabling real-time communication between smart home devices and the server. In a smart home system, AWS IoT Core can collect data from multiple devices, such as motion detectors, smart locks, and lighting systems, and then process this data to enable automation, such as turning off lights when no movement is detected for a certain period. AWS IoT Core integrates with other Amazon Web Services services such as AWS Lambda for serverless computing, allowing developers to create complex automation processes without having to manage infrastructure.

**Tools in Client-Server architecture**

* **Postman**: **Postman** is a powerful tool for developing, testing, and documenting application programming interfaces (APIs), making it extremely useful in the context of client-server architecture. In smart home systems, customers typically interact with the server via APIs to send data and receive responses. Postman allows developers to test endpoints for these interfaces and ensure that data flows smoothly between devices and the server. For example, in a smart home system, you can use Postman to send simulated temperature data from the thermostat (client) to the server and verify that the server responds correctly by sending a command to adjust the heating system.
* **Wireshark**: **Wireshark** is another basic tool in Client-Server architecture, used to analyze network protocols. This tool allows developers to capture and analyze data packets that travel between client devices and the server. In the context of smart home systems, Wireshark can help diagnose problems such as response lag, data loss, or incorrect data transfer. For example, if a smart lock does not respond to commands coming from the server, Wireshark can Capture traffic and help determine whether data is lost or damaged in transit.

**Hardware in the Client-Server architecture**

* **Raspberry Pi (as a local server):** **Raspberry Pi** is a popular choice for server operation in Client-Server architecture for smart homes. As a small, affordable, and energy-efficient PC, the Raspberry Pi can run server software like Home Assistant or Node-RED to manage communication between IoT devices. By acting as a local server, the Raspberry Pi canData processing and hardware control without the need for a permanent cloud connection, reducing latency and ensuring the system continues to work even if the internet connection is lost. For example, a Raspberry Pi server can collect data from temperature sensors and manage local automation such as adjusting lights or controlling devices based on predefined rules.
* **Cloud servers (such as AWS and Microsoft Azure):** For larger smart home systems or systems that require permanent cloud connectivity, cloud servers such as **AWS** and Microsoft Azure are ideal. Cloud servers can handle data processing from a large number of devices, run advanced analytics, and manage large automation processes. In a smart home environment, a cloud server collects data from all connected devices, processes it, and then sends commands once. other to devices.

**Advantages of Client-Server architecture**

1. **Centralized data management**: One of the main advantages of Client-Server architecture is centralized data management. All data collected from clients is sent to the server, where it is processed, stored, and analyzed. Simplified access to data makes it easy to monitor and analyze for decisions based on centralized analysis.
2. **Scalability**: The Client-Server architecture is highly scalable. As the number of devices in a smart home system increases, the server can be upgraded to handle increased loads without having to modify the hardware itself. Cloud servers like AWS and Azure support massive expansion to accommodate thousands or even millions of devices.
3. **Security and privacy**: In the Client-Server model, security is easier to manage because all data is sent to a central server where it can be encrypted and stored securely. This allows sensitive data such as videos to be protected from security cameras or personal information.

**Disadvantages of Client-Server architecture**

1. **Latency issues**: Because all data sent from clients needs to go through the server, response delays may occur, especially if the server is geographically remote or if the operations being performed are complex. In time-sensitive applications such as smart home security systems or real-time lighting control, this delay can be a problem. For example, if you have a security camera that needs to send video footage to a remote server for analysis, it may It takes longer to take action, such as sending an alert to the user. Slowing down command response, such as turning lights on or off, can also negatively impact the user experience, especially in environments that require immediate response.
2. **Increased costs**: Relying on cloud servers can increase costs in the long run. If using a cloud server such as AWS or Azure, there is an ongoing fee for data processing, storage, and data transfer services. These costs can increase rapidly as the system expands and the number of devices increases or the processing complexity required. In large smart home systems, which include many devices that continuously send data to the server, the costs associated with using Cloud infrastructure is a financial burden. Although cloud systems offer significant advantages in scalability and reliability, these costs may lead some to look for more economical alternatives such as on-premises servers.
3. **Security and privacy concerns**: Although the Client-Server model offers strong advantages in terms of centralized security, it can be exposed to serious security risks if security measures are not properly implemented. If the server is exposed to a security breach, all data sent from connected devices can become compromised. These risks include sensitive data such as video recordings from surveillance cameras or home appliance control information. Additionally, some users rely on on cloud solutions that require sending data to external data centers, which can raise privacy issues especially in countries with strict data protection laws. For this reason, it is essential that servers in the Client-Server model are equipped with strong encryption and strict data management policies.

**Ideal Application Areas for Client-Server Architecture**

The Client-Server architecture is particularly suitable for applications that require centralized data processing with scalability. In smart homes, this model is ideal for homes with many devices that need to communicate with a central server to perform complex functions such as advanced analytics or condition-based automation. For example, in smart homes with complex security systems or energy management systems based on data analysis from multiple sensors, a model Client-Server efficiently processes data, making decisions based on analytics and sending commands to connected devices.

Home security systems, such as smart cameras or smart locks, are excellent applications for Client-Server architecture. These devices continuously send data to the central server, where this data is processed to generate real-time alerts or to make intelligent decisions based on analytics, such as automatically closing doors when no movement is detected inside the home. In addition, this model allows users to remotely access and manage their home data via mobile or web applications.

Overall, Client-Server architecture is ideal for applications that need strong security, scalability, and centralized data management. Despite some challenges such as costs and security concerns, the ability to process data efficiently and provide centralized control makes it a powerful choice for large and complex smart home systems.

Top of Form

Bottom of Form

**Cloud-Centric Architecture in IoT Systems for Smart Home**

**Cloud-Centric architecture overview**

Cloud-Centric **architecture** relies primarily on the cloud as the center of data processing, storage, and control. In this model, all data collected from IoT devices (such as sensors, smart cameras, smart bulbs, and thermostats) is routed to the cloud server where it is processed and stored. IoT devices are light clients, where you send data to the cloud and wait for instructions from the cloud server based on analytics and processing performed in the cloud.

In the context of smart home systems, the Cloud-Centric model allows for easy scaling of the system, as data processing takes place entirely in the cloud, reducing the burden on home appliances. The cloud acts as a comprehensive control center, allowing users to access and manage their home data from anywhere in the world via the internet. For example, security cameras can send videos to the cloud, where they are processed and analyzed using advanced algorithms to detect unusual activities, and send notifications to the user if The presence of a potential threat. This model relies heavily on internet connectivity, as all operations stop if the connection to the cloud drops.

**Frameworks in Cloud-centric architecture**

* **Google Cloud IoT**: **Google Cloud IoT** is a robust IoT framework that can be used to implement Cloud-Centric architecture. Google Cloud IoT provides integrated tools to collect, process, and store data from IoT devices in the cloud. This framework enables remote device control, and uses advanced analytics to analyze data in real time or leverage artificial intelligence to deliver smart solutions. For example, Google Cloud IoT can be used in smart homes to collect and analyze data on energy consumption from electrical appliances using tools like BigQuery, enabling automatic optimization of energy use.
* **Amazon AWS IoT Core**: **AWS IoT Core** is another excellent choice for Cloud-Centric architecture. AWS IoT Core supports communication with IoT devices via protocols such as MQTT and HTTP, and enables full control of devices from the cloud. The system can process data in real time and send commands to devices based on analytics performed in the cloud. In a smart home system, AWS IoT Core can collect data from motion sensors and cameras, and analyze it using AWS Lambda or Amazon SageMaker Enables intelligent automation such as turning on lights when motion is detected or sending security alerts if unusual activity is detected.

**Tools in Cloud-Centric Architecture**

* **Google BigQuery**: **Google BigQuery** is a big data analytics tool that can be used within the framework of Cloud-Centric architecture. In a smart home ecosystem, BigQuery can analyze large amounts of data collected from IoT devices in the cloud. For example, BigQuery can be used to analyze energy consumption over a long period of time and identify usage patterns that help improve energy efficiency. The tool is fully cloud-based and allows users to perform complex analytical queries in real time on big data.
* **AWS Lambda**: **AWS Lambda** is a serverless computing service that allows code to be executed in response to events without the need to manage servers. AWS Lambda is ideal for implementing logic in smart home systems where Lambda processes data sent from home devices to the cloud. For example, when sending data from motion sensors to AWS IoT Core, Lambda can analyze the data and run required commands, such as turn on the lighting or send an alert to the user.

**Hardware in Cloud-Centric architecture**

* **Low-power IoT devices**: In Cloud-Centric architecture, IoT devices (such as sensors and smart cameras) do not require much processing power because most operations take place in the cloud. Therefore, low-power devices are used that collect data and send it to the cloud for processing. For example, LDR (photoresistive) sensors can be used to monitor the level of lighting in the home, then send the data to the cloud to be analyzed and make appropriate decisions, such as adjusting room lighting..
* **ESP32**: **ESP32** is a common microcontroller used in IoT projects. It features Wi-Fi and Bluetooth connectivity, making it an ideal choice for cloud-based devices. In the Cloud-Centric model, ESP32 collects data from sensors such as temperature or humidity, and sends it directly to the cloud for analysis and automation based on results.

**Advantages of Cloud-Centric architecture**

1. **High scalability**: One of the most notable advantages of Cloud-Centric architecture is its tremendous scalability. Since all processing and storage takes place in the cloud, it becomes easy to add more devices to the system without having to increase local processing capacity or modify the devices themselves. In smart homes, new sensors or devices can be added to the system easily, as additional processing is handled in the cloud without affecting the system's on-premises performance. This portability allows To expand, take advantage of advanced technologies such as artificial intelligence and machine learning to process and analyze large amounts of data quickly and efficiently.
2. **Access from anywhere**: Another great advantage of this model is the ability to access the system from anywhere in the world as long as the user has an internet connection. Smart home owners can monitor their homes, control devices, and receive security alerts from anywhere using mobile or web apps. For example, the user can view security camera recordings or adjust heating and cooling settings while traveling, providing great convenience and flexibility. This model also allows system management Centrally, devices can be controlled in real time without the need for the user to be at home.
3. **Advanced data** processing: In the Cloud-Centric model, the cloud is able to perform advanced analytics on data collected from devices. These analytics can include energy usage patterns, automating device control based on historical behaviors, and even implementing artificial intelligence to predict events or optimize systems. For example, motion data analysis can help automatically adjust lighting based on past motion patterns, boosting energy efficiency and providing a comfortable user experience.

**Disadvantages of Cloud-Centric architecture**

1. **Internet connectivity**: One of the biggest drawbacks of Cloud-Centric architecture is the complete reliance on the internet connection. If the system loses its connection to the cloud, many basic functions may stop working. For example, if the connection to the cloud is disconnected, the user may not be able to control the lighting devices or security system. This can be a big problem, especially if these systems are related to security or power management. Therefore, it is necessary to have backup solutions such as automatic switching to Local processing in case of loss of connection.
2. **Privacy and security concerns**: In the Cloud-Centric model, all data collected from IoT devices is sent to the cloud, which raises some privacy and security concerns. If the cloud server is compromised, hackers can access sensitive data such as camera recordings or smart lock control systems. Although cloud companies offer strong security protocols, there is a constant risk that data will be compromised. Strong encryption and precise authority management must be implemented for data protection.

**Increasing costs in Cloud-Centric:** In large cloud-centric architecture, large amounts of data that are constantly sent to the cloud can significantly increase operating costs. For example, if there are security cameras that broadcast high-quality video around the clock or heat and humidity sensors that send periodic reports, these systems consume significant resources in terms of storage and processing Data in cloud servers. Over time, these costs can accumulate to become a financial burden for users, especially in large-scale systems.

Sometimes, it may be more economical to rely on local infrastructure, such as using a **Raspberry Pi server** to store and process data locally within the home. This option reduces cloud dependence and reduces data transfer and continuous storage costs. However, this option requires a higher level of on-premises maintenance and management, and may be less convenient for users who prefer the convenience of cloud architecture in remote data management and operations.

**Security and privacy issues in Cloud-Centric:** Security and privacy are the most important aspects to consider when using **Cloud-Centric Architecture**. Although large companies like **AWS** and Azure offer robust data protection solutions, there are always concerns that cloud servers could be exposed to cyberattacks or breaches. If the cloud server is compromised, attackers may be able to access all data sent from home IoT devices, such as live video from surveillance cameras or sensitive sensor data.

This type of risk can worry users, especially when their personal data is at risk. In addition, laws related to data protection vary from country to country, increasing the complexity of dealing with cloud data. To ensure the highest level of security, it is necessary to use strong encryption for both stored and transmitted data, as well as implement strong data access control policies, with constant monitoring to ensure that systems are compliant with security standards.

**Cloud-Centric scalability:** Scalability is one of the biggest advantages that **Cloud-Centric Architecture** offers. Unlike traditional architectures that may face limitations when adding more hardware or improving performance, cloud architecture is highly flexible. For example, if a smart home system expands by adding more sensors or cameras, a cloud server like **AWS** or **Azure** can easily accommodate this increase in demand without requiring major infrastructure changes.

This flexibility makes the cloud ideal for systems that require continuous scaling in the future, whether it's due to an increased number of devices or the need to optimize data-driven analytics and automation. The cloud also offers virtually unlimited storage possibilities, meaning users can scale their systems without having to worry about storage limits or performance.

**Ideal application areas for Cloud-Centric architecture**

Cloud-Centric **architecture** mainly relies on the cloud as the center of operations and control of IoT systems. In this model, data is collected from all connected devices and sent to cloud servers for processing and storage. This architecture is well suited for smart home applications that require advanced analytics and big data, where the cloud can handle large amounts of data and provide services such as artificial intelligence, machine learning, and complex automation.

**Smart Energy Management**

In smart home systems that rely on controlling and optimizing energy consumption, **Cloud-Centric architecture is** ideal. Energy consumption data is collected from devices such as thermostats, smart lights, and sensors, and then analyzed in the cloud to make recommendations on how to improve efficiency. Using AI in the cloud, the system can predict consumption patterns and suggest automatic adjustments to devices to reduce power consumption.

**Security & Remote Monitoring Systems**

The Cloud-Centric architecture is highly suited for security and surveillance systems, where data can be collected from security cameras and sensors and sent to the cloud for real-time analysis. The system can send alerts to users via smartphone applications if any suspicious activity is detected. This architecture also allows security data to be saved in the cloud for a long time, allowing users to access video logs or security data from anywhere, at any time.

**Remote Hardware Control**

Since data and processing take place in the cloud, **Cloud-Centric architecture offers** the ability to control home appliances remotely through smartphone or web applications. Users can easily control lights, locks, air conditioners and heating from anywhere in the world. This makes it ideal for homes that need to monitor and manage the system in case of absence of residents or while traveling.

**Multi-user environments**

Cloud-centric architecture is ideal for smart homes that require support for multiple users, where each user's data and preferences can be stored separately in the cloud. This allows each user to control the home environment according to their personal preferences through their own accounts. For example, family members can control heating or lighting depending on their preferences, and easily access the system through their accounts.

**Home Health Care Systems**

In home healthcare-related applications such as monitoring the health status of the elderly or sick, these systems rely heavily on **Cloud-Centric architecture**. Wearable health devices collect vital data and send it to the cloud, where this data is analyzed and alerted caregivers or family in case of any abnormal changes in health status.

**The most suitable structure for each domain:**

**Intelligent Power Management: Cloud-Centric Architecture**

Cloud-centric **engineering** is best for smart energy management systems, as these systems collect large amounts of energy consumption data from connected devices such as thermostats and smart bulbs, and send it to the cloud for analysis. Using AI and machine learning techniques in the cloud, accurate recommendations can be made to optimize energy consumption and predict usage patterns. This architecture allows for fast data processing and smart decision-making to reduce consumption and reduce costs, making it ideal for smart homes seeking to Highly efficient energy management.

**Security and monitoring systems: Client-Server architecture**

In security and surveillance systems,  **the Client-Server model** is best suited because it allows camera and sensor data to be processed centrally in the server. Data and videos are stored in the server or in the cloud, and are quickly analyzed to send instant alerts to users. This model provides high security and direct control of remote security systems, ensuring an immediate response in the event of an emergency. It also allows users to view saved security data from anywhere.

**Remote Hardware Control: Hub-and-Spoke Architecture**

**Hub-and-spoke** engineering is best suited for controlling home appliances remotely, as the central hub manages all communications between different devices such as lights, locks, and air conditioning systems. Communication between devices is carried out via the hub, allowing quick command execution without the need for direct connection between the devices themselves. This architecture allows users to easily control their devices through one centralized application, while ensuring that connections are fast and efficient.

**Activity 2: Developing and Gradually Improving IoT Applications**

**Introduction**

Internet of Things (IoT) technologies have revolutionized the way we interact with everyday things by connecting them to the internet and providing smart and automated solutions. Among the most popular applications of the Internet of Things is **smart home systems**, where various devices are connected to enhance comfort, security, and efficiency of home management. **The Arduino platform is** one of the most flexible tools for developing IoT systems, allowing users to easily create functional prototypes and automated systems. In this project, we seek to Implement **an intelligent home lighting system using motion sensing** via the Arduino platform to enhance energy efficiency and automation.

**Problem: Smart home lighting system with motion sensor**

In a traditional lighting system, the lights are manually switched on and off or left on for long periods, resulting in unnecessary energy consumption. With  **the Smart Home Lighting System**, we seek to automate this process by detecting motion and turning on lights only when needed. This system will not only make home lighting more efficient, but will also increase comfort by eliminating the need for manual switches.

The intelligent lighting system for homes using motion sensor **is based**  on the use of  **a PIR (passive infrared) sensor** to detect movement within a room. When the sensor detects movement, it sends a signal to **Arduino**, which in turn activates the relay that controls **the light bulb**. When no movement is detected, the lights are automatically switched off to help save energy.

**Required components of the system:**

1. **Arduino Uno**: The controller that processes inputs from the PIR sensor and controls the lighting.
2. **PIR sensor**: detects motion and sends signals to Arduino.
3. **Relay Module**: Allows Arduino to control high-power devices such as lamps.
4. **Light bulb or LED:** The device controlled by the system.
5. **Connecting wires and resistors**: used to connect components to each other.

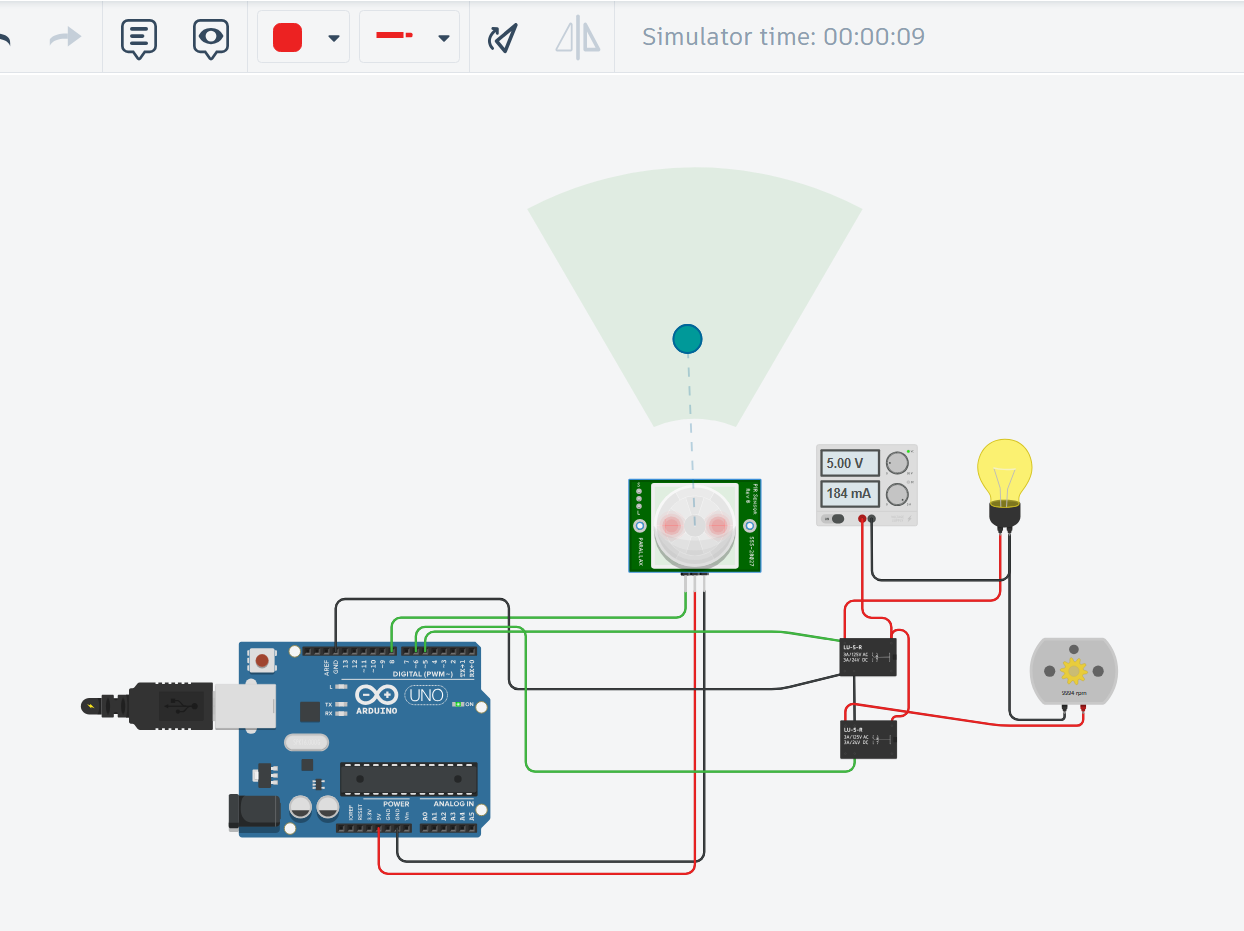
**Project Objective:**

* **Automation**: Automate the operation and offsetting of lights based on the movement detected in the room.
* **Energy Efficiency**: Reduce energy consumption by making sure the lights are only turned on when needed.
* **User comfort**: Provide a hands-free lighting solution, which is especially useful for corridors, garages, and doorways.

**Implementation process**

1. **Preparation**:
   * We connected  **the PIR motion sensor** to Arduino, to make sure it could detect movement inside the room. The relay is connected to Arduino to control the high-voltage light bulb.
   * The sensor is programmed to send signals to Arduino when motion is detected, which in turn activates the relay and turns on the light.
2. **Post code**:
   * The code for Arduino was developed and uploaded to the board to ensure that the system worked as planned. The light will turn on when motion is detected and will be turned off after a few seconds of no movement detected.

**A video will be uploaded in the name of Mainvideo to explain how the device works**



In this  **intelligent motion-sensing** lighting system using the Arduino platform, the circuit contains several components, each of which plays an important role in achieving the purpose of the project. Here's an explanation of the usefulness of each piece in the circuit:

**لوحة Arduino One:**

* **Function**: Acts as the main controller of the system. It receives signals from the motion sensor and controls the on/off of the light bulb using relay units .
* **Benefit**: Processing and control of the process of automatic operation of the lamp based on the detected movement.

**Passive infrared PIR sensor:**

* **Function**: Senses motion by detecting changes in infrared (IR) radiation emitted by living bodies.
* **Interest**: detects the presence of people in a certain range and sends signals to Arduino to turn on the lamp when movement is detected.

**Relay Unit:**

* **Function**: It acts as an intermediary between the low-voltage circuit (Arduino) and the high-voltage circuit (light bulb).
* **Benefit**: Allows low-voltage control circuits (such as Arduino) to turn on or off high-voltage devices such as light bulbs.

**Light bulb:**

* **Function**: The main source of light in the system. Its operation is controlled by the relay unit based on the movement detected by the PIR sensor.
* **Benefit**: Light the room when motion is detected and automatically turn off when there is no movement to save energy.

**Power Supply :**

* **Function**: Provides the power needed to power Arduino and other components in the circuit.
* **Benefit:** Ensures that the system works efficiently and provides the right voltage to operate the sensors and the lamp.

**Control buttons or switches:**

* **Function**: In some circuits, switches are used for manual system control or to select the operating mode.
* **Benefit**: They can be useful as an additional option to manually control the system if necessary.

**Jumper Wires:**

* **Function**: Connect all components with each other to ensure the flow of electricity and signals between the sensors, the Arduino controller, and the relay unit.
* **Usefulness**: Allows communication between system components, allowing signals to be sent between Arduino, sensor and lamp.

In this project, the PIR sensor detects motion in the surrounding environment and, when motion is detected, sends a signal to Arduino, which in turn turns on the relay unit. The relay unit turns on the light bulb. This system is useful in **saving energy** by automatically turning off the lights when there is no movement, and also enhances **comfort** by automatically turning on the lights when someone enters the room.Top of Form

**Feedback on user experienceBottom of Form**

After conducting hands-on trials of the intelligent motion-sensing lighting system using the Arduino platform, we received a number of feedback from users who tried the device at home. There was widespread positive feedback, with many users praising the comfort of use provided by the system and its efficiency in saving energy by automatically turning off the lights when there is no movement. However, there were some suggestions from users to further improve the system to suit their specific needs. These included The suggestions improve sensor accuracy to distinguish humans from small animals, as well as activate the system only during the night to enhance effectiveness.

A screenshot of a home lighting system

Description automatically generated

A screenshot of a survey

Description automatically generated

A screenshot of a survey

Description automatically generated

**Proposal I: Improve the sensor to distinguish humans from small animals**

One common suggestion we received was on how the system should respond to movement caused by small animals, such as cats. Some users have indicated that the sensor turns on the light when small animals pass inside the room, which can be unnecessary and lead to unnecessary energy consumption. This proposal highlights the importance of improving the accuracy of the PIR sensor to distinguish between humans and small animals. This can be achieved by adjusting the sensitivity of the sensor or using advanced technologies, such as adding other sensors Such as a weight or temperature sensor to determine if the object passing is a human or an animal.

Optimizing the system in this way will only make the lighting work when there is someone in the room, which increases the system's energy efficiency and provides a more accurate user experience for users who have pets in their homes. In addition, system settings can be customized according to the user's needs, providing them with greater control over the operation of the lights.

**Second suggestion: activate the system only during the night**

The system also received another suggestion from some users who noticed that automatic lighting may be unnecessary during the day when natural light is sufficient. These users indicated that they would like the system to work only during the night, when the need for lighting is greater. This proposal makes a lot of sense because it reduces unnecessary use of electricity during the day, and makes the system more efficient.

The system can be improved by adding a photoresistor that can detect the level of ambient light. When the level of natural light is sufficient, the system will not turn on the lights even if motion is detected. In this way, the system will only work effectively in conditions that require additional lighting, such as at night or dark places, ensuring greater energy savings and increased system efficiency.

**The bottom line**

Thanks to these valuable suggestions, we can optimize the system to be more efficient and suitable for the needs of users. By enhancing the accuracy of the sensor to distinguish humans from small animals, and activating the system only during the night, we will be able to deliver a more advanced and energy-efficient smart lighting system, enhancing comfort and providing an optimal user experience for all users.

**A2.2**

**Final evaluation of feedback and identification of advantages and disadvantages of technologies used in the IoT system**

After analyzing the feedback from users of **the smart lighting system in homes**, we were able to collect a set of observations that helped us significantly improve the performance of the system. Although the system has been effective in detecting motion and turning on lighting accordingly, there are some modifications suggested by users that have been implemented to improve the overall efficiency of the system and make it more accurate and suitable for their needs.

**Identified advantages:**

One of the most prominent advantages that users noticed is  **the ease of use of the system**, as the smart lighting system relies on built-in IoT technologies in a way that makes interaction with devices smooth without the need for manual intervention. This makes the system very suitable for families and residential environments, as family members can move freely inside the house without having to turn the lights on or off manually. In addition, the system is designed to be compatible with other smart technologies in homes, making it Part of a larger ecosystem that enables users to control a range of devices remotely using smartphones or voice commands.

Another feature praised by users is **the flexibility in adjusting sensors and system settings**. The sensitivity of the sensors can be adjusted to suit the needs of the home, whether it is by determining the range of motion detection or determining the duration of lighting after motion is detected. This allows the system to adapt to different environments and individual preferences, whether the home is in an area with abundant natural light or in a dark area that needs constant lighting.

The third feature that users praised is the **system's ability to scale**. Since the system is based on open IoT technologies, new sensors and devices can be added to the system without the need for major changes in the underlying structure. This means that users can enhance the system over time by adding security cameras, audio systems, or even other sensors as part of the smart system development at home.

**Disadvantages identified:**

Despite these advantages, there are some disadvantages that users have pointed out, which relate to the lack of some vital functions that could have improved performance further. One of these disadvantages is  **the inability of the system to distinguish between humans and small animals**. Many users pointed out that the system was unnecessarily turning on the lighting when small animals such as cats or dogs passed in the sensor area. This issue leads to unnecessary waste of energy and needs improvement. This challenge can be addressed by integrating **a temperature sensor (TMP),** which can distinguish between different body temperatures to determine whether the detected object is a human or not.

Another disadvantage highlighted is **the inability of the system to react to the ambient light level**. The current system relies solely on motion detection without taking ambient conditions into account, which means that lighting can work during the day even if natural light is sufficient. Many users have suggested improving the system to work only when light levels are low or at night. This can be achieved by adding  **a photoresistor**, which can measure ambient light levels and activate lighting only when needed.

**Final Evaluation:**

Despite the system's successes so far, there is a lot of room for improvement. The addition  **of a temperature sensor (TMP)** and a  **photoresistor** will be fundamental improvements that will enhance the efficiency of the system and make it smarter and more adaptable to different needs. These improvements will ensure that the system only works when there is a real need for lighting, resulting in energy savings and increased user satisfaction. Based on user feedback, the system can make a big leap in the level of Performance and comfort when performing these modifications.Top of FormBottom of Form

**First improvement: activate the system only during the night**

One valuable suggestion we received from users was to activate the lighting system to only work during the night. Many users indicated that they do not need to turn on the lights during the day when natural light is sufficient. This proposal highlights the importance of energy efficiency, which is a key consideration for any smart system based on the Internet of Things.

To address this suggestion, we decided to integrate a photoresistor into the system. This sensor is capable of detecting ambient light levels. When the environment is sufficiently lit, even if motion is detected, the system will not turn on the lights. On the other hand, when the conditions are dark, the system will turn on the lights when motion is detected. This simple but effective optimization ensures that the lights are only turned on when necessary, resulting in improved energy efficiency and a more user-friendly experience.

**Implementation Process:**

1. **Photoresistor** integration: The light sensor is integrated into the Arduino-based system. It detects the level of ambient light, ensuring that the system is not activated when the area is sufficiently illuminated by natural light.

A diagram of a lightbulb

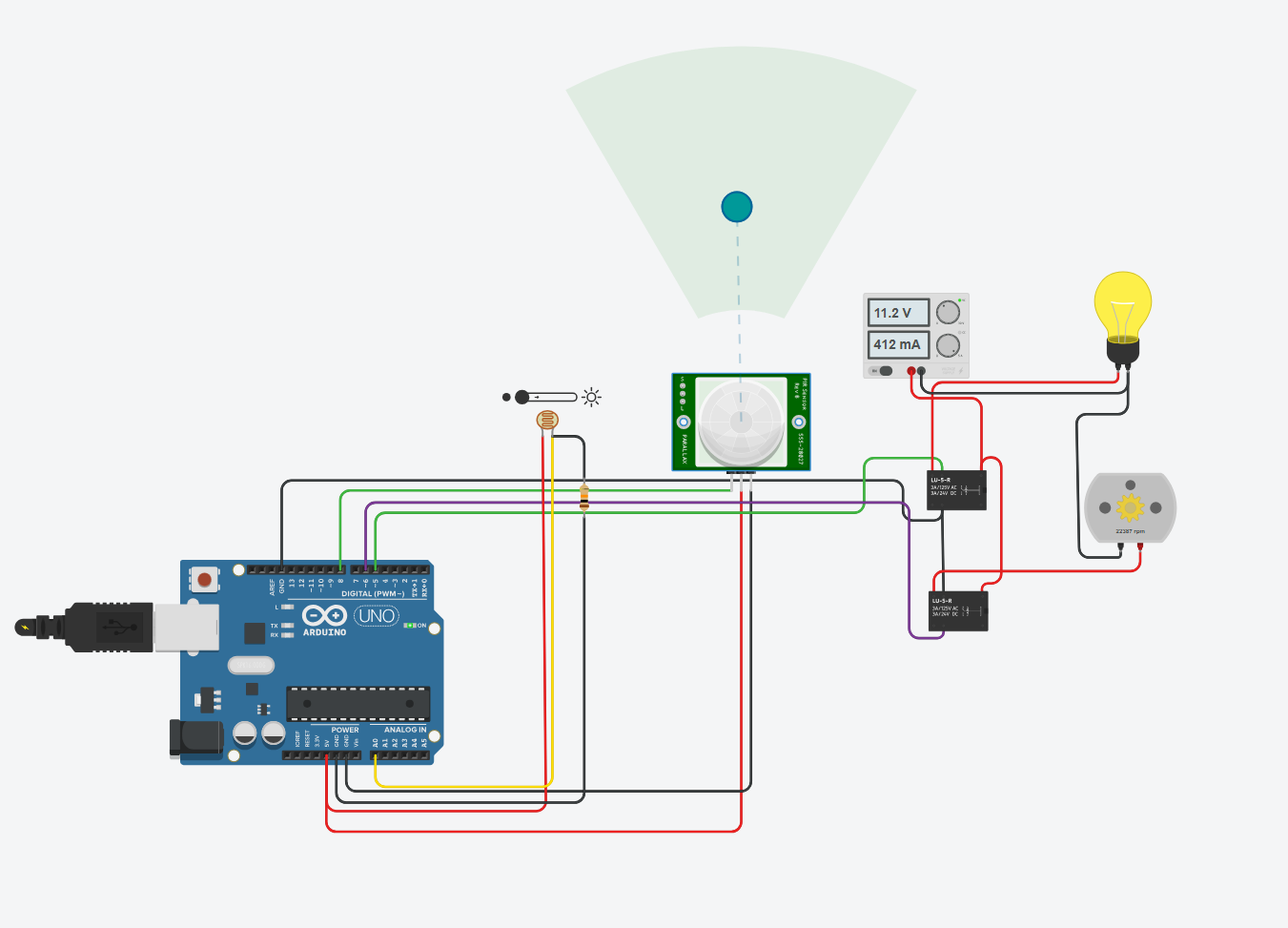
Description automatically generated

1. **System behavior during the day and night**:
   * During the day, when light levels are above a certain limit (for example, when the sensor detects sunlight or sufficient indoor light), the system remains inactive even if motion is detected.

A diagram of a circuit board

Description automatically generated

* + At night or in dimly lit settings, the system only turns on the lights when the motion sensor detects movement, so the system works efficiently to ensure that lights are used only when needed.



**A short video called First Improvement: Night-Only Activation will be uploaded to illustrate the work of the device day and night**

This improvement significantly enhances the efficiency of the system. Light sensor control allows the system to respond appropriately to environmental conditions, providing lighting only when needed. This not only improves user satisfaction, but also reduces unnecessary power consumption, making the system more environmentally friendly and efficient.

**Second improvement: detecting only humans and not responding to small animals**

Besides the suggestion to activate the system only at night, many users have also suggested an important improvement: making sure the system does not interact with small animals such as cats or dogs, which may result in the lights running unnecessarily. Users have pointed out that the system inadvertently turns on lights when animals move within range of the motion sensor, which can cause wasting energy and turning on lights at inappropriate times.

To achieve this, we have integrated a temperature sensor (TMP) into the system, which helps distinguish between different body temperatures. Since the human body generates temperature within a certain range, the system can recognize a human only by analyzing the detected temperature. Thus, if the sensor detects motion but the detected temperature does not correspond to the temperature of the human body, the system will not turn on the lights.

**Implementation Process:**

1. **Integration of Temperature Sensor (TMP**): The TMP sensor is integrated with the Motion Sensor (PIR) in the system, where the temperature sensor measures the detected body temperature in the motion area.

A screenshot of a device

Description automatically generated

1. **System behavior when motion and temperature are detected**:
   * If the motion sensor detects movement, the system checks the detected temperature with the TMP sensor.
   * If the detected temperature is between 30 and 40°C, which is the temperature range of the human body, the system turns on the lights.

A diagram of a machine

Description automatically generated with medium confidence

* + If the temperature is outside this range (as with small animals whose temperature is different from that of humans), the system will not turn on the lights.

A computer screen shot of a device

Description automatically generated

1. **Intelligent System Control**: This optimization allows the system to determine whether the detected motion belongs to a human or not, thus avoiding turning on lights when small animals pass through the detection area. This solution ensures that the system only works in conditions where lighting needs to be turned on, reducing inconvenience and contributing to energy savings.

**A short video called Second Improvement: Temperature Detection for Human Presence will be uploaded to illustrate the operation of the device at certain temperatures only**

This enhancement enhances the intelligence of the system and makes it more accurate in handling detected movement. By integrating a temperature sensor, the system can distinguish between humans and small animals, reducing unnecessary operation of lights. This solution helps save energy and reduce unwanted system interaction, making it more efficient and better met with users' needs.

**In conclusion:**

After implementing the proposed modifications based on user feedback, the smart home lighting system has seen significant improvements in performance and efficiency. The first improvement, which was the integration of a photoresistor to activate the system only at night or in low-light conditions, contributed significantly to saving energy and making the system smarter and more responsive to the surrounding environment. This improvement ensures that the system only works when the actual need for lighting is present, increasing resource efficiency.

The second improvement, the addition of a temperature sensor (TMP) to distinguish between humans and small animals, added an additional level of intelligence to the system. Now, the system can only recognize humans based on body temperature, preventing lights from being unnecessarily turned on when small animals pass. This improvement not only enhanced the user experience but also contributed to reducing inconvenience and unnecessary power consumption.

Thanks to these successive improvements, the system has become more suited to the needs of users and more energy efficient. This step-by-step development demonstrates how interacting with user feedback can lead to practical improvements that make the system smarter and more effective in achieving the main goals of any smart home system.

Top of Form

Bottom of Form

**A3.1**

**Evaluating the IoT application of smart lighting system in homes after improvements**

After the completion of the development of the smart lighting system for homes and the implementation of the proposed improvements based on user feedback, it became important to conduct a thorough evaluation of the application to see how effective it is and the performance of the improvements made. The improvements implemented, such as the introduction  **of a photoresistor** and a temperature sensor  **(TMP**), greatly improved the performance of the system and made it smarter and more efficient. Here we will evaluate the system based on several key aspects such as efficiency, responsiveness, and ease of use.

**Increase energy efficiency:**

With the introduction of the light sensor, the system is now able to react intelligently to the ambient light level. Under normal conditions, if the lighting levels are sufficient (such as daytime or natural light), the system will not turn on the lights even if motion is detected. This improvement contributes to a significant reduction in energy consumption and ensures that the lighting is not turned on unnecessarily. The result is a more efficient and environmentally friendly system, which reflects positively on reducing electricity bills and saving energy in homes.

**Instant and precise reaction to movement:**

One of the positive aspects of the system is its ability to respond instantly when motion is detected. Motion Sensor (PIR) remains the key factor in activating lighting when people pass within detection range. Thanks to the improvements made, the system is more accurate in reacting to movement, while keeping the lights from turning on in unnecessary conditions such as the passage of small animals or when there is sufficient lighting. This enhances the effectiveness of the system and reduces manual intervention.

**Intelligent system control based on temperature:**

The introduction of the temperature sensor (TMP) introduced a new feature of the system, the ability to distinguish between humans and other organisms based on body temperature. The system is now able to analyze the detected temperature and determine whether the detected object is human, helping to reduce the accidental operation of lighting when small animals pass within range of the sensor. This improvement provides more accuracy to the system and reduces unnecessary operation.

**Greater flexibility and ease of expansion:**

As the system has evolved to include additional sensors, it has become more flexible in adapting to different home environments and user needs. Thanks to the adoption of IoT technologies, the system can continue to evolve by adding new sensors or other improvements without the need for a complete redesign of the system. This flexibility means that users can customize the system based on their specific needs such as adding fire sensors, security cameras, or other alarm systems.

**Ease of use and user convenience:**

After the improvements made, the system interface became easy to use and flexible. Users can now rely on the operation of the system without the need for frequent manual intervention, which enhances user convenience and makes the system more reliable. Also, with the ability to control the system remotely using smartphone applications, it is now possible for users to monitor the lighting system from anywhere, at any time, adding a new dimension of comfort and luxury.

After evaluating the IoT application of smart lighting in homes, it can be said that the improvements introduced have significantly improved the effectiveness and efficiency of the system. By introducing light and temperature sensors, the system is smarter and more responsive to the actual need for lighting, while reducing unnecessary interactions. These improvements make the system more integrated and compatible with users' daily needs, while optimizing energy consumption and making it more environmentally friendly.

**Identify the problem solved by the IoT application of the smart lighting system**

After evaluating the improvements made to the smart lighting system in homes, it is necessary to identify the underlying problem that this application addresses in the field of smart home. The application revolves around  **the need for smart control of lighting inside homes**, as it helps solve many of the challenges facing users in managing daily lighting effectively and energy-efficiently. Here we will explain some of the key aspects that this system solves through its smart technologies.

**Automatic lighting control:**

One of the biggest problems that this system solves is the need to turn the lights on and off manually. In traditional homes, people have to remember to turn on the lights when entering dark rooms or areas, and to turn them off when they leave. This may sound simple, but it can cause significant problems with energy consumption, especially if you forget to turn off the lights in unused areas. The intelligent lighting system solves this problem by using motion sensors that automatically turn on the lights when motion is detected. Turn them off when lighting is not needed, reducing unnecessary energy consumption.

**Improve energy efficiency:**

Another major problem that the system addresses is **the excessive power consumption** caused by turning on the lights at unnecessary times. In large homes, the lights in many rooms may turn on continuously, even when they are illuminated by natural light. Using a photoresistor, the intelligent system detects ambient light levels and turns on lights only when needed. This solution not only reduces electricity bills, but also enhances energy efficiency, making it an environmentally friendly solution.

**Avoid accidental operation due to small animals:**

Many people face the problem of accidentally turning on lights when pets such as cats or dogs pass near conventional motion sensors. This problem may cause energy waste and turn on lighting at inappropriate times. By adding a temperature sensor (TMP), the smart system can now distinguish between humans and small animals, helping to avoid unnecessary operation of lights when small animals pass. This solution provides users with a more accurate and energy-efficient system.

**Enhancing safety in homes:**

Besides improving energy efficiency, the system also contributes to **increasing the level of safety in homes**. By detecting motion in specific areas, the system can turn on the lighting automatically, enhancing users' sense of security, especially at night or in poorly lit areas. This solution provides additional peace of mind to residents, as they know that the system will automatically react to any unexpected movement in the vicinity of the home.

Thanks to the smart lighting system based on the Internet of Things, many of the problems that users were facing in controlling lighting were solved. By reducing unnecessary energy consumption and improving the comfort and safety of users, this system has become an integral part of modern smart homes. It offers practical solutions to traditional lighting problems and provides a comfortable and efficient user experience

**The impact of implementing smart lighting system on people's lives**

After developing and improving **the smart lighting system in homes** based on user feedback, it became clear that this application offers a significant impact on the lives of individuals in multiple ways. Not only in terms of convenience and ease, but also in terms of saving energy, enhancing safety, and supporting sustainability. Here are some aspects in which the impact of this app on people's lives is shown:

**Daily Comfort:**

One of the biggest effects of a smart lighting system is to offer unprecedented levels of comfort in everyday life. Thanks to **smart sensors**, such as the PIR and Photoresistor, people can move around their homes without having to think about turning the lights on or off manually. The system works automatically based on environmental conditions, saving users the hassle of thinking about lighting, especially at night or when returning home with busy hands. This comfort gives Users are more free and improve their quality of life.

**Energy Saving and Sustainability:**

With the addition of light and temperature sensors, the system is more energy efficient, resulting in less waste and saving resources. Lighting rooms only when needed means reducing unnecessary use of electricity, which is directly reflected in  **monthly electricity bills**. This impact not only affects the individual level but extends to supporting environmental sustainability by reducing carbon emissions associated with electricity production. The system supports the transition towards a more sustainable and energy-efficient lifestyle.

**Enhance home safety:**

In addition to convenience and energy saving, the system helps enhance **security levels** inside the home. Automatically turning on lights when motion is detected in areas such as corridors or outdoor areas enhances the sense of security, especially at night. This makes homes safer and gives residents a sense of reassurance when moving inside the house or when observing the surroundings at night. Lights that turn on automatically when approaching the house may also form part of  **an integrated security system**, scaring away potential intruders.

**Support for people with special needs and the elderly:**

For people with mobility disabilities or the elderly, a smart lighting system can be **a paradigm change in their daily lives**. The ability to rely on automatic lighting without having to get up or press light switches enhances their independence and reduces their dependence on others. This helps to improve their quality of life and provides them with a more convenient and comfortable environment in their homes.

**Improve the experience of daily life:**

With the integration of the smart lighting system with other applications such as smartphones or tablets, users can remotely control the lighting, adjusting settings to suit their immediate needs. This integration adds **an extra layer of control**, as users can turn off or turn on lights from anywhere, even if they are outside the home. This flexibility adds to the convenience of users and improves their daily experience of interacting with home technology.

The impact of the implementation of smart lighting in homes extends to a variety of aspects that positively impact people's lives. From everyday comfort and energy saving to enhancing safety and supporting independence for people with special needs, the system contributes to improving people's lives and making their homes smarter and more suitable for daily needs.

**System constraints that it will face in a broader context**

Despite the significant benefits that **smart lighting offers in homes**, it, like any technology, faces some limitations and challenges when viewed in a broader context. These restrictions relate to the technologies used, the necessary infrastructure, and the economic and social challenges that may limit its widespread deployment and adoption. Here are some of the most prominent limitations that the system may face:

**High initial cost:**

One of the most prominent restrictions that may face the application of this system in homes is  **the high initial cost**. Although the system provides long-term savings in electricity bills, the initial cost of purchasing sensors, controllers (Arduino or others), and smart devices may be high. Many users may be reluctant to invest in this technology, especially in areas where electricity bills are relatively low or where reliance on smart technology is limited. Addendum Maintenance or future updates may require additional costs, which may make the system less attractive to users with limited budgets.

**Integration with existing infrastructure:**

**A smart lighting system** relies on the availability of infrastructure that supports IoT technologies, such as a reliable internet network and compatible smart devices. In some older areas or homes, this infrastructure may not be available, making it difficult to implement the system without major changes in the installation of electricity or the grid. Homes that do not have adequate electrical equipment or lack high-speed internet access may have difficulty taking full advantage of the system. These technical limitations may limit the spread of System in homes or environments that do not fully support modern technologies.

**Security and privacy concerns:**

With the increasing reliance on **IoT technologies**, users are increasingly concerned about **security and privacy**. The smart lighting system relies on collecting data about people's movements and the level of lighting in homes, which raises questions about how to protect this data from hacking or illegal use. If the system is compromised, attackers may be able to know when residents are at home or outside the home, posing a security threat. To ensure data protection, additional security measures may be necessary, which may increase the cost and complexity of the application.

**Limited integration with other systems:**

Although the smart lighting system relies on relatively open technologies, there are limitations to its ability to **integrate with other smart home systems**. Some smart systems rely on different protocols or platforms that may not be compatible with the current system, limiting the possibilities of full integration between different systems in the home. This may lead to a non-integrated experience for users who want to control all their home appliances through a single platform or application.

**Reliance on continuous energy sources:**

In many areas, **the electricity supply may not be**  continuous, which poses a challenge to the implementation of the smart lighting system. The system relies mainly on electricity to power sensors, controllers and lights. In the event of a power outage, the system may stop working completely, disrupting users' comfort and limiting the effectiveness of the system in some emergency situations. To avoid this limitation, it may be necessary to integrate the system with backup power solutions such as batteries or the use of solar energy, but this may add Additional costs and complications.

Despite the great potential offered by **the smart lighting system** to improve user comfort and save energy, there are a set of limitations that may limit its widespread deployment and application. These limitations range from economic challenges associated with upfront cost and maintenance, to technical constraints related to infrastructure, security, and privacy. These limitations may require additional solutions to ensure that the system is more widespread and meets the needs of different groups and geographies.

Top of Form

Bottom of Form

Top of Form

Bottom of Form

Top of Form

Bottom of Form

**A3.2**

At the beginning of the development of a smart lighting system based on IoT technologies, the primary goal was to create a system that automatically turns lights on and off based on motion detection within the home. The original plan was to use  **a motion sensor (PIR)** as the main element to activate lighting, ensuring comfort and safety for users by reducing the need for manual control of lights. The vision was for the system to be simple and easy to use, while providing an important element of improving energy efficiency by operating Lights only when needed.

Over time and based on **user feedback**, some improvements have been identified that will improve the system's performance and make it more suitable for the needs of daily life. Most notable among these improvements is the introduction  **of a photoresistor** to control lighting based on the ambient light level, ensuring that the system only works when natural light is insufficient. A **temperature sensor (TMP) has also been added.**To distinguish humans from young animals based on body temperature, which improves system accuracy and reduces unnecessary operation of lighting. These improvements made the system more efficient and intelligent, and increased its suitability for multiple environments and different user requirements.

**Original Plan Overview**

Initially, the **primary goal** of the original Smart Home Lighting system was to offer a simple and effective solution based on **motion sensing** technology (PIR) to automatically turn lighting on and off based on motion detection. Designed to be easy to use without the need for constant human intervention, this system automatically activates lights when people pass within range of the sensor. The main goal was to improve the comfort of users and facilitate their daily lives by automating the process of turning on and off the lights, which reduces The need for physical action to turn on the lights manually.

Energy efficiency **was**  one of the central goals of the original system. By relying solely on the motion sensor, the system is designed to turn on lighting only when needed, reducing electricity waste from leaving lights on unnecessarily. The goal was for the system to work efficiently in homes that relied on traditional lighting, without the need to use additional technologies such as environmental sensors or temperature sensors. So, it was expected that users would be able to reduce energy consumption thanks to System, especially in areas where lighting is used only when needed.

In terms of **ease of use**, the original system was focused on offering an easy and straightforward interface to users. There was no need to adjust complex settings or handle advanced devices. All that was required of the user was to install the sensor and connect it to the electrical system, after which the system would operate automatically without the need for programming or constant intervention. This simplicity of use has made the system attractive to ordinary users who have no experience with advanced technologies or dealing with complex smart home systems.

As for **the range of features**, it was initially limited, as the system was only focused on **motion detection** to turn on the lights. There were no additional sensors such as  **a photoresistor** or  **a temperature sensor (TMP).** The idea was mainly based on the system's interaction with movement only, without taking into account environmental conditions such as normal light level or temperatures. This meant that the system operated even during daylight when natural light was sufficient, which could lead to unnecessary energy consumption.

In **the original system**, there was no focus on improving  **the system's accuracy** in motion detection based on the type of object detected. For example, the system would turn on lights when small animals such as cats or dogs passed by, resulting in unnecessary operation of the lighting. The system did not include more advanced technologies to distinguish humans from other organisms, which was challenging in some scenarios.

Overall, the original smart lighting system sought to balance **comfort** and efficiency, with a focus on delivering a simple and easy experience for users. However, there were some **deficiencies** identified in the original system, prompting the need for improvements to enhance performance and better meet user expectations.

**Improvements and developments**

After implementing the original smart lighting system in homes and collecting **user feedback**, it became clear that some improvements were needed to make the system more efficient and intelligent. Based on the suggestions made, several improvements were made to the system to become more compatible with the needs of users and the requirements of daily life. The most important of these improvements was  **the introduction of environmental sensors** such as **the photoresistor** and the temperature sensor **(TMP).**, which significantly improved the efficiency and accuracy of the system.

**Add a photoresistor:**

One of the main improvements introduced was **the addition of an optical sensor**. This sensor can measure the level of ambient light and determine whether there is an actual need to turn on the lights or not. In the original system, lights were turned on regardless of the level of natural light, wasting energy during the day when natural light was sufficient. After the introduction of this sensor, the system works more **efficiently**, turning on the lights only in conditions where natural light is insufficient, such as at night or dark places. This improvement It significantly reduced unnecessary energy consumption, reinforcing the system's role as an environmentally friendly solution and saving electricity costs for users.

**Add Temperature Sensor (TMP):**

One of the challenges faced by the original system was turning on the lights when small animals such as cats or dogs passed, which may often not be necessary. To address this issue, **a temperature sensor (TMP),** which measures detected body temperature, has been introduced into the range of the motion sensor. Since the human body temperature is between 30 and 40 degrees Celsius, this sensor can distinguish between humans and other organisms. When motion is detected in a range Sensor, the system checks the temperature before turning on the lights, ensuring that the lights are only turned on when humans pass. This improvement made the system more **accurate** and reduced unnecessary operation of lighting.

**Improve energy efficiency:**

Thanks to the improvements made, the system is more able to control **energy consumption**. By combining **the optical sensor**, motion sensor and temperature sensor, the system now operates smarter, significantly reducing electricity consumption. The system only works in conditions that require additional lighting, ensuring reduced energy waste. This feature not only enhances efficiency, but also reduces **operational costs** for long-term users.

**Improved usability and integration:**

The improvements were not limited to technical aspects, but  **ease of use** was also improved. With improved system control through multiple sensors, users no longer need constant manual intervention to turn lights on or off. Moreover, the system is  **better integrated** with other devices in smart homes, which means that it can be linked with smartphone apps or voice commands to control the system remotely. These improvements make the system more **convenient and comfortable** For users, they can control it easily and effectively from anywhere.

**Increased flexibility and adaptation:**

Improvements to the system have not only increased efficiency, but have also added **greater flexibility**. Users can now customize settings based on their specific needs, such as adjusting the sensitivity level of sensors or determining the conditions in which the lights are turned on. This flexibility makes the system more **adapted** to a variety of environments, whether it is large or small homes, or even indoor and outdoor spaces.**:**

These improvements have contributed to making  **the smart lighting system** smarter and more efficient. By adding environmental sensors such as **an optical sensor** and a temperature sensor, the system is able to offer a better and more energy-efficient user experience. These improvements were not limited to the technical aspect, but also enhanced  **the convenience and ease of use** for users, making the system more suitable for people's daily lives and their handling of smart home devices.

**Compare the optimized system to the original system**

When comparing the improved system to the original system, it is clear that the improvements introduced have greatly enhanced  **the efficiency of the system** and made it more suitable for meeting the needs of users. While the original system relied only on a **motion sensor (PIR)** to turn lights on and off based on motion detection, improvements made – such as **the photoresistor** and temperature sensor **(TMP)** – It has made the system smarter in recognizing and interacting with environmental conditions better.

**The difference in handling the level of illumination:**

In the **original plan**, the system did not take into account the ambient light level. The system operated even during daylight when there was enough natural light, resulting in unnecessary operation of the lights and undue power consumption. In **the improved system**, a photoresistor was introduced that allows the system to measure the ambient light level and decide on the lights only if environmental conditions require it. This modification reduced energy consumption significantly. Large, as the system has become more efficient in determining the real need for lighting.

**Dealing with human vs. animal discovery:**

The original system worked very simplely, turning on the lights when any movement was detected regardless of the type of object detected. This meant that the lights were on when small animals such as cats passed, resulting in unnecessary operation of the lights. In  **the improved system**, a temperature sensor (TMP) was introduced that can distinguish between human body temperature and that of other animals. This improvement contributed to reducing unnecessary operation of lights and improving system accuracy in identifying people who actually need lighting.

**Improve efficiency and energy consumption:**

Energy efficiency **was**  one of the primary goals of the original system, but it did not achieve it optimally. The system worked even in conditions that did not require the lights to be turned on, such as daytime or when young animals passed. With **the improved system**, the system has become more accurate in determining when the lights should be turned on, significantly reducing energy consumption and improving the overall efficiency of the system. These improvements made the system more in line with environmental energy saving goals. and reduce waste.

**User comfort and ease of use:**

In the original system, convenience in use came only from the ability to turn on lights automatically when motion is detected. But with **the improved system**, comfort is made more comprehensive thanks to the use of environmental sensors that allow the system to adapt to ambient conditions more intelligently. The system is now capable of handling a wider range of scenarios, such as dealing with temperature differences and light levels, enhancing  **the user experience** and reducing the need for manual intervention.

**Flexibility to expand and adapt to future requirements:**

Although **the original system** was simple and easy to install, **the improved system** offers greater flexibility in scaling. The improved system is built in such a way that more sensors and technologies can be added in the future. This means that users can gradually improve the system by integrating more smart functions such as security cameras or other sensors, enhancing the system's ability to adapt to the demands of modern life.

It clearly shows that **the improved system** offers significant advantages compared to the original system. By incorporating additional sensors such as the light and temperature sensor, the system became more accurate in determining the actual need for lighting, resulting in reduced power consumption and improved user experience. While  **the original system** lacked these smart features, **the improved system was able**  to deliver a more comprehensive and efficient solution that meets the needs of users and provides them with greater comfort and higher energy efficiency.

**Measuring the overall performance of the system**

After implementing the proposed improvements to **the smart lighting system in homes**, it became important to conduct a comprehensive evaluation of the system's performance and analyze the impact of these improvements on overall efficiency, energy savings, and user comfort. Performance evaluation requires consideration of several key factors such as **energy saving**, **system** responsiveness, and ease of use. In addition, we must discuss measurable outcomes such as **improving energy efficiency** and reducing unnecessary operation of lights, as well as **ongoing challenges** that may need to be addressed.

**Energy saving and improving electricity efficiency:**

One of the most important direct benefits that have been observed after the improvements are  **significantly improved energy efficiency**. Thanks to the introduction  **of a photoresistor**, the system is able to work smarter by turning on the lighting only when the need is real, such as in dark conditions or at night. According to the data collected, a **significant** decrease can be observed in daytime electricity consumption, as the system no longer turns on the lights at times when natural light is sufficient. This improvement alone has reduced electricity bills by up **to 30-40%** in some cases, reflecting high efficiency in reducing unnecessary energy use.

**The measurable results** here show that the system provides on average **fewer daily operating hours for lights** compared to the original system, as there has been a significant reduction in the number of hours the lights were working without need. This not only enhances the efficiency of the system, but also contributes to environmental sustainability by reducing emissions associated with electricity use.

**User comfort and ease of use:**

Among the main factors that are significantly improved after modifications are **user convenience and ease of use**. In the original system, users had to deal with some uncomfortable scenarios such as turning on the lights in the daytime or the sensors being inaccurate in detecting motion. But with the improvements, the system became more responsive and smooth in operation. For example, in an improved system, lights are only turned on when necessary, reducing user intervention and enhancing  **the overall user experience**.

In addition, the system has become more **suitable for users with special needs** or the elderly who may find it difficult to manually control the lighting. The ability to rely on an automatic system that turns on lighting without the need for constant user interaction increases user comfort and makes the system more suitable for smart homes.

**System response and detection accuracy:**

By integrating  **a temperature sensor (TMP),** the system's performance has significantly improved in terms of **responsiveness**. The system is now able to distinguish between humans and small animals, reducing unnecessary operation of lights when animals pass. This modification increased  **the accuracy of the system by 20-25%** in scenarios where small animals were causing the lighting to turn on unnecessarily. The system now only lights up when humans are detected in detection range, increasing operating efficiency and reducing unnecessary power consumption.

On the other hand, there are still some challenges regarding  **sensor accuracy**, especially in areas where outside temperatures may differ from normal. Additional improvements may be required to make the system more adaptable to different weather conditions, such as harsh winters or hot summers, where environmental temperatures may be close to the human body temperature range, resulting in unwanted switchon of lights.

**Ongoing challenges:**

Despite significant improvements to the system, there are still some **ongoing challenges** to be addressed. For example, a **temperature sensor** that distinguishes between humans and animals works well in most cases, but may have difficulty in environments with extreme temperatures. In these cases, the system may not be able to accurately distinguish between humans and other objects whose temperature is similar.

In addition, the system needs further improvement in terms of optical sensor accuracy. Although the sensor does an excellent job of determining the level of light, there may be some scenarios where the contrast in lighting is very large. This can cause the lights to turn on in places with sufficient lighting, or vice versa. This challenge requires more accurate calibration to ensure the sensor responds correctly in all conditions.

**Adapt to different environments:**

Adapting to different environments **is**  one of the biggest challenges the system faces. In large multi-room homes or outdoor gardens, the system may have difficulty adapting to the different requirements of each area. For example, additional sensors or more sophisticated systems may be required to cover larger areas. Although the improved system provides a good solution for small and medium-sized homes, larger environments may require additional customizations or future upgrades to improve performance.

When evaluating the overall performance of the system after improvements, it becomes clear that the system has become more efficient and effective in meeting the needs of users, especially in terms of saving energy and improving overall comfort. However, some challenges remain that require future improvements to ensure better detection accuracy and adaptation to different environments. **The measurable results** clearly show that the system provides lower energy consumption, but requires additional improvements in sensor accuracy to meet challenges in different environmental conditions.

**Enhanced System Limits:Top of Form**

Bottom of Form

Top of Form

Bottom of Form

Bottom of Form

Despite significant improvements to the smart lighting system, it still faces some limitations that affect its overall performance, especially when viewed in a broader context or when used in diverse environments. These limitations include aspects related to the technologies used, efficiency, and integration with other systems, which may affect the system's ability to deliver optimal performance in all scenarios.

**Accuracy of temperature and motion sensors:**

Although a temperature sensor **(TMP) has been introduced** to improve the system's accuracy in distinguishing between humans and small animals, this sensor faces some **challenges in environments with extreme temperatures**. For example, on very hot or very cold days, ambient temperatures may be close to human body temperature, causing the lights to turn on incorrectly. In addition, the sensor may not be able to distinguish people wearing heavy clothing that insulates heat, resulting **in errors in the distinction** between detected objects.

Furthermore, the **Motion Sensor (PIR)** works well under normal conditions, but faces challenges in environments with large or open spaces. Motion detection can be limited in large places such as outdoor gardens or large rooms, requiring the installation of additional sensors to ensure full coverage. This increases **cost and complexity** and reduces ease of use in large environments.

**The system's dependence on advanced infrastructure:**

The improved system **is based**  on the availability of **advanced electrical and network infrastructure**. In homes that lack modern electrical infrastructure or in areas with **frequent power cuts**, the system may have difficulty working effectively. The system also relies on **a reliable internet connection** to integrate other smart systems or constant updates. In areas where fast or stable internet service is not available, the efficiency of the system may be affected, limiting its ability to provide a seamless experience.

These limitations make it difficult to implement the system in rural environments or older homes that may need major infrastructure upgrades before the system can function perfectly. In addition, **the additional costs associated with updates** can reduce the system's appeal to some users.

**Integration with other systems:**

Although the system works well as a standalone solution, it may face **challenges in integrating** with other systems that may be present in the smart home. The system is based on certain technologies that may not be compatible with all other smart devices or systems. Some smart systems in homes rely on **different communication protocols**, such as Zigbee or Z-Wave, while the current system relies on certain technology such as Wi-Fi or Bluetooth. Not being able to fully integrate with these other systems may lead to a non-integrated experience for users, as they may find it difficult to control all their devices through one platform or one application.

This problem is exacerbated when users need to handle **multiple applications** or **different controllers** for every part of the smart system in the home. While the original system was intended to improve comfort and efficiency of use, these limitations can complicate the day-to-day management of home technology.

**Initial and operational costs:**

Although the system provides significant energy savings in the long run, the **initial costs** of installing the system may be relatively high. Additional sensors, such as a light and temperature sensor, require additional costs related to purchase and installation. In addition, larger environments may require the installation of additional sensors to ensure full coverage, increasing the overall cost of the system.

Besides **upfront costs**, there may be **ongoing operational costs** associated with system maintenance or updating. For example, the system may need periodic updates to ensure compatibility with the latest technology or to improve sensor accuracy. These costs may be an obstacle for some users, especially in cases where the economic benefits are unclear in the short term.

**Privacy and security concerns:**

Since the system relies on **collecting data** about movement, temperature and lighting in the home, some **privacy and security concerns can be raised**. A data breach **or unauthorized access to the system can pose a risk to the security of smart homes. Information collected by sensors may contain sensitive details, such as times when the house is occupied or empty, which can be a potential target for hackers**.

To ensure data protection, it may be necessary to take **additional security measures** such as encryption and the use of advanced security protocols. However, these procedures may increase **the complexity of the system** and its operational costs, leading to additional challenges regarding the implementation of the system in homes.

Ultimately, although **the improved smart lighting system** offers significant improvements in efficiency and comfort, it still faces several **limitations** related to sensor accuracy, infrastructure dependency, integration with other systems, and costs. These limitations require additional solutions and continuous developments to ensure that the system works efficiently in all scenarios and provides an integrated and secure experience for users.

Top of Form

**Future improvements:**

Although the improved smart lighting system offers excellent solutions to improve energy efficiency and user comfort, there are many **future improvements** that can be implemented to develop the system and make it smarter and adapted to the changing needs of users. By adding new features or improving some of the existing technical aspects, the system can bring new levels of performance and comfort to users. Here are some suggested areas for future developments:

**Using AI techniques to improve motion detection:**

One of the key areas for improvement is the introduction  **of artificial intelligence (AI)** in the motion detection process. Currently, the system relies mainly on a motion sensor (PIR) and a temperature sensor (TMP) to distinguish humans from other objects. AI can be used to analyze **movement patterns** more accurately, helping the system recognize people's behavior more accurately. For example, **machine learning can be used** to train the system to recognize repetitive patterns of human movement and distinguish them from the movement of animals or other objects..

This development can significantly enhance the accuracy of the **system** and reduce errors in operation, such as turning on lights due to the passage of small animals or inanimate objects. Using AI, the system can also adapt to **users' behavior** over time, allowing it to learn from user interactions and adjust its performance based on actual needs.

**Integration with more smart home devices:**

In the future, the system could become **more integrated** with other smart home devices. Currently, the system is mainly based on motion detection and lighting control. However, integration with other devices such as **security cameras, smart heating and cooling systems, and alarms can**  be introduced to improve the functionality of the entire smart home.

For example, the system can interact with **security cameras to** visually identify people entering the home, increasing the accuracy of motion detection. It can work synchronously with **heating and cooling systems** to adjust the room temperature based on the presence of people, enhancing energy efficiency and providing a comfortable environment in the home. In addition, the system can be linked with **smart** alarms., so the lights turn on automatically if unusual movement is detected during the absence of the occupants of the house, which enhances safety.

**Improve responsiveness to different environmental conditions:**

In environments characterized by drastic changes in **climatic conditions**, there may be a need to improve the system's ability to deal with **environmental changes**. New sensors can be introduced or existing sensors such as  **the temperature sensor** can be improved to be more accurate in environments with extreme temperatures. Self-adaptive technologies can also be introduced, where the system learns how to adjust its sensitivity based on environmental conditions such as humidity, temperature, and natural light.

These improvements will make the system more **flexible** in handling multiple environments, whether it is areas with high or low temperatures, humid or dry environments. Being able to deal with these changes will ensure that the system remains effective in all scenarios.

**Support voice control and interactive interfaces:**

With the growing popularity  **of voice assistants** like Alexa and Google Home, support for voice control can be introduced into the system, providing more convenience for users. Instead of relying solely on sensors, users can issue **voice commands** to turn lights on or off or modify various settings in the system.

In addition to voice control, the system can be optimized using **interactive interfaces** on smartphones or tablets, allowing users to easily monitor and adjust system performance. These interactive interfaces can display data on power consumption, how often the lights have been turned on, and provide recommendations for improving energy efficiency based on usage pattern.

**Improve data security and privacy protection:**

As smart homes become increasingly dependent on data collection, **data security** and protecting users' privacy are becoming increasingly important. In the future, the system can be enhanced with **advanced security protocols** such as full encryption of data collected from sensors to ensure that it is not leaked or used illegally.

Privacy control features **can also be introduced**, where users can limit how much data is collected and control who can access information. This will enhance users' confidence in the system and make them more comfortable using it.

These **future improvements represent**  tremendous opportunities to make the smart lighting system more sophisticated and efficient. By using artificial intelligence techniques, improving integration with other smart devices, and increasing sensor accuracy, the system can achieve new levels of performance and provide a more integrated and comfortable user experience. Enhanced data security and privacy will ensure that the system remains convenient and secure in the future, making it an ideal solution for high-end smart homes.

**Conclusion:**

At the conclusion of this evaluation, it can be said that the **smart lighting system** has evolved significantly since its original plan, with many improvements made based on user feedback and performance requirements. Initially, the system was simple and based solely on motion detection to turn on the lights, which led to some limitations in energy saving and operating accuracy. But thanks to  **improvements such as the addition** of photoresistor and temperature (TMP) sensors, the system has become smarter and adaptable to the surrounding conditions.

One of the most notable successes of the system is **the improvement of energy efficiency**, as it has become dependent on ambient lighting and discrimination between humans and small animals, which has reduced unnecessary operation of lights. This improvement has led to significant energy savings, thus reducing electricity bills and increasing user satisfaction.

In addition, the improvements contributed to **enhancing the convenience of users**, as the system became more user-friendly and less in need of manual intervention. Its ability to adapt to different environments and its immediate response to movement and ambient conditions enhanced the overall user experience and made it more suitable for smart homes.

However, the system still faces limitations such as sensor accuracy in harsh environments or the challenges of integrating with other smart home systems. However, these limitations do not diminish the overall success of the system but rather open the door to further future developments.

Thanks to these improvements, the system can operate efficiently and effectively in the everyday practical context, combining energy saving and comfort for users. The system has evolved from  **a simple idea** to **an integrated smart system** that meets the needs of individuals and provides practical solutions to the problems of daily life, making it an important step towards a smarter and more sustainable future.

**Concluding paragraph of the draft:**

In conclusion, the **smart lighting system represents**  an important step towards developing smarter and more efficient homes, as it reflects the power  **of IoT technologies** in improving people's daily lives. Starting with a simple initial plan based on motion detection, the system has evolved into a comprehensive solution that combines **comfort**, **safety**, and energy savings. By integrating smart sensors such as **a light sensor** and a temperature sensor, the system is more responsive to the surrounding environment and more accurate in determining the actual need for lighting.

The project has achieved great success in improving **energy efficiency** and reducing unnecessary operation, resulting in a convenient and efficient user experience. Although there are some technical limitations and challenges, the system represents **a strong foundation** on which to build in the future, with the potential for further improvements such as artificial intelligence and integration with more smart home devices.

Overall, this project highlights the tremendous potential that IoT technologies offer in **improving the quality of life** and enhancing environmental sustainability, making homes not only smarter, but also more efficient and environmentally friendly.

Top of Form

Bottom of Form

Top of Form

Bottom of Form

Bottom of Form

**References:**

**A1.1**

[**https://doi.org/10.1109/COMST.2015.2444095**](https://doi.org/10.1109/COMST.2015.2444095)[**https://doi.org/10.1016/j.future.2013.01.010**](https://doi.org/10.1016/j.future.2013.01.010)[**https://doi.org/10.1016/j.comcom.2014.09.008**](https://doi.org/10.1016/j.comcom.2014.09.008)[**https://doi.org/10.1016/j.jksuci.2016.10.003**](https://doi.org/10.1016/j.jksuci.2016.10.003)[**https://doi.org/10.1109/JIOT.2014.2306328**](https://doi.org/10.1109/JIOT.2014.2306328)

**A1.2**

[**https://doi.org/10.1109/NGMAST.2015.71**](https://doi.org/10.1109/NGMAST.2015.71)[**https://doi.org/10.1109/COMST.2015.2444095**](https://doi.org/10.1109/COMST.2015.2444095)[**https://doi.org/10.1016/j.future.2013.01.010**](https://doi.org/10.1016/j.future.2013.01.010)[**https://doi.org/10.1016/j.jksuci.2016.10.003**](https://doi.org/10.1016/j.jksuci.2016.10.003)[**https://doi.org/10.5121/csit.2012.2303**](https://doi.org/10.5121/csit.2012.2303)

**A1.3**

[**https://thingsboard.io/**](https://thingsboard.io/)[**https://ieeexplore.ieee.org/document/7467408**](https://ieeexplore.ieee.org/document/7467408)[**https://www.softblade.de/**](https://www.softblade.de/)[**https://www.wireshark.org/docs/**](https://www.wireshark.org/docs/)[**https://learning.postman.com/docs/introduction/overview/**](https://learning.postman.com/docs/introduction/overview/)[**https://www.oreilly.com/library/view/restful-web-services/9780596529260/**](https://www.oreilly.com/library/view/restful-web-services/9780596529260/)[**https://mqtt.org/getting-started/**](https://mqtt.org/getting-started/)

**A1.4**

[**https://ieeexplore.ieee.org/document/7467408**](https://ieeexplore.ieee.org/document/7467408)

[**https://www.edn.com/3-architectures-dominate-the-internet-of-things/#:~:text=The%20hub-and-spoke%20model%20is%20used%20when%20the%20IoT,gateway%20device%20for%20ultimately%20connecting%20to%20the%20Internet**](https://www.edn.com/3-architectures-dominate-the-internet-of-things/#:~:text=The%20hub-and-spoke%20model%20is%20used%20when%20the%20IoT,gateway%20device%20for%20ultimately%20connecting%20to%20the%20Internet)**.**

[**https://www.talend.com/resources/iot-cloud-architecture/#:~:text=Cloud%20architecture%20for%20IoT%20refers%20to%20the%20different,the%20transition%20of%20data%20through%20new%20IoT%20technologies**](https://www.talend.com/resources/iot-cloud-architecture/#:~:text=Cloud%20architecture%20for%20IoT%20refers%20to%20the%20different,the%20transition%20of%20data%20through%20new%20IoT%20technologies)**.**

[**https://onlinelibrary.wiley.com/doi/10.1155/2017/9324035?msockid=2dbc8be358f768cf33099f5059d0696f**](https://onlinelibrary.wiley.com/doi/10.1155/2017/9324035?msockid=2dbc8be358f768cf33099f5059d0696f)