A TERM PAPER REPORT

ON

**INDUSTRIAL AUTOMATION SYSTEM**

Submitted in partial fulfilment of requirements for the completion of 3rd Year 2nd Semester in

BACHELOR OF TECHNOLOGY

**(BATCH 2021-2025)**

**IN**

**DEPT. OF CSE (IOT)**

SUBMITTED BY

|  |  |
| --- | --- |
| **Name** | **Reg.No** |
| Y. Vinnutna | Y21CO059 |
| Sk. Fakruddin | Y21CO046 |
| N. Prabhash | Y21CO039 |
| Sk. Saidavali | Y21CO047 |
| D. Vijay | Y21CO016 |

UNDER THE GUIDANCE OF

**Dr.N.Naga Malleswara Rao**, Prof & HOD, Dept of IoT



**DEPARTMENT OF CSE (INTERNET OF THINGS)**

**RVR & JC COLLEGE OF ENGINEERING (Autonomous)**

NAAC A+ grade | Approved by AICTE | Affiliated to Acharya Nagarjuna University Accredited by NBA |

Sponsored by Nagarjuna Education Society | Chandramoulipuram, Chowdavaram, GUNTUR-522019

**RVR & JC COLLEGE OF ENGINEERING (Autonomous)**

DEPARTMENT OF CSE (INTERNET OF THINGS)



**CERTIFICATE**

This is to certify that the Term Paper report entitled **“INDUSTRIAL AUTOMATION SYSTEM”** has been submitted to Department of CSE (IoT), RVR & JC College of Engineering(A) by **Y.VINNUTNA (Y21CO059), SK.FAKRUDDIN (Y21CO046), N.PRABHASH (Y21CO039), SK.SAIDAVALI (Y21CO047), D.VIJAY (Y21C0016)** for partial fulfilment of requirements for the completion of 3rd Year 2nd Sem in Bachelor of Technology (batch 2021-2025) in Dept. of CSE (IOT) is a bonafide work carried out by them. This work is not submitted to any university for the award of any degree.

**Head of the Department**

Dr. N. Naga Malleswara Rao

Professor & HOD

Department of CSE (IoT), RVRJCCE.

**Project Guide**

Dr. N. Naga Malleswara Rao

Professor & HOD

Department of CSE (IoT), RVRJCCE.

**DECLARATION**

This project work entitled **“INDUSTRIAL AUTOMATION SYSTEM”** is carried out in partial fulfilment of requirements for the award of degree BACHELOR OF TECHNOLOGY (B. Tech) in COMPUTER SCIENCE & ENGINEERING (IoT) to Acharya Nagarjuna University, Guntur. We hereby declare that this Project Report has not been submitted to any other University/Institution for the award of any Degree.

Date:

Place: Chowdavaram

Y.Vinnutna

(Regd.No:Y21CO059)

Sk.Fakruddin

(Regd.No:Y21CO046)

N.Prabhash

(Regd.No:Y21CO039)

Sk.Saidavali

(Regd.No:Y21CO047)

D.Vijay

(Regd.No:Y21CO016)

**ACKNOWLEDGEMENT**

The successful completion of any task would be incomplete without a proper suggestion, guidance and environment. Combination of these three factors acts like backbone to our Term Paper **“INDUSTRIAL AUTOMATION SYSTEM”.**

We would like to express our gratitude to the Management of **R.V.R & J.C COLLEGE OF ENGINEERING** for providing us with a pleasant environment and excellent lab facility.

We regard our sincere thanks to our Principal, **Dr.Kolla.Srinivas** for providing support and stimulating the environment.

We are greatly indebted to **Dr.N.Naga MalleswaraRao**, Professor and Head of the Department Internet of Things, for his valuable suggestion during the course period.

We would like to express our special thanks of gratitude to our guide **Dr.N.Naga Malleswara Rao** who helped us in doing the Term Paper successfully.

We would like to thank our Term Paper in-charge who gave us the opportunity to do this work.

Y.Vinnutna

(Regd.No:Y21CO059)

Sk.Fakruddin

(Regd.No:Y21CO046)

N.Prabhash

(Regd.No:Y21CO039)

Sk.Saidavali

(Regd.No:Y21CO047)

D.Vijay

(Regd.No:Y21CO016)

**ABSTRACT**

This paper presents an innovative IoT-based embedded system designed to revolutionize automatic industries by offering real-time monitoring, control, measurement, product counting, and data analysis. The system is built around a Raspberry Pi that operates the entire system, analyses machinery performance, and provides self-analysed data results to enhance productivity and efficiency. The system leverages cloud server connectivity via Cayenne software, ensuring seamless data storage and retrieval, while its foundation of Linux and Python programming languages guarantees stability, security, and easy integration with various sensors and devices. HTML-based user interfaces provide intuitive data visualization and access, making the system both efficient and user-friendly.

In summary this integrated IoT-based embedded system offers ground-breaking capabilities for automatic industries. By utilizing the power of Raspberry Pi, Linux, Python, HTML, and Cayenne software, industries can now effortlessly monitor their operations, control industrial processes, and optimize their performance for improved productivity and efficiency. The system's comprehensive features make it an indispensable tool for the modern automatic industry, facilitating streamlined operations and fostering overall growth and success. As an innovative solution to real-world industry challenges, this IoT-based embedded system promises to revolutionize the automatic industry landscape by providing powerful tools to collect, process, and analyze data in real-time. The seamless integration of various technologies and components ensures that this system is adaptable to a wide range of applications, making it an invaluable asset for any automatic industry seeking to maximize efficiency and productivity.

The system's effectiveness is rooted in its ability to harness the potential of cutting-edge technologies, such as Raspberry Pi, Linux, Python, HTML, and Cayenne software. By leveraging these powerful tools, industries can now easily collect data from various sensors and devices, analyse it for insights and trends, and use this information to make informed decisions that drive their operations forward.

Moreover, the user-friendly nature of this integrated IoT-based embedded system ensures that even individuals with limited technical expertise can benefit from its advanced features. The intuitive data visualization provided by HTML-based interfaces makes it easy to understand complex data, while the centralized control system allows for seamless management of industrial processes.

Ultimately, the implementation of this innovative IoT -based embedded system has the potential to transform automatic industries by providing comprehensive monitoring and control capabilities. By optimizing industrial processes and ensuring maximum efficiency, these systems can significantly contribute to increased profitability and overall competitiveness in today's fast-paced global market.

In addition, the ability to store and analyse data in real-time using Cayenne software empowers industries with valuable insights that can be used to identify areas for improvement and make

proactive changes to enhance their performance. This data-driven approach enables automatic industries to adapt quickly to changing market demands and stay ahead of their competitors.

As the world continues to evolve and become more connected, it is crucial for automatic industries to embrace innovative technologies like this integrated IoT-based embedded system. By doing so, they can ensure their long-term success and maintain a competitive edge in the ever-changing global landscape. The advancements offered by this system provide a comprehensive solution to optimize industrial processes and drive growth in the automatic industry sector, ultimately paving the way for a more connected, efficient, and prosperous future.

**CONTENTS**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **ABSTRACT** | |  |
|  | **LIST OF FIGURES** | |  |
| **1** | **INTRODUCTION** | | 1 |
|  | 1.1 Application | | 3 |
|  | 1.2 Technology | | 3 |
| **2** | **LITERATURE SURVEY** | | 4 |
|  | 2.1 Inferences from Literature Survey | | 5 |
|  | 2.2 Open problems in Existing System | | 7 |
| **3** | **REQUIREMENTS ANALYSIS** | | 9 |
|  | 3.1 | Feasibility Studies/Risk Analysis of the Project | 10 |
|  | 3.2 | Software Requirements Specification Document | 11 |
| **4** | **DESCRIPTION OF PROPOSED SYSTEM** | | 14 |
|  | 4.1 | Selected Methodology or process model | 16 |
|  | 4.2 | Description of Software for Implementation and Testing plan of the Proposed Model/System | 26 |
|  | 4.3 | Project Management Plan | 27 |
|  | 4.4 | Architecture / Overall Design of Proposed System | 28 |
| **5** | **SUMMARY** | | 42 |
|  | 5.1 | Code Execution | 44 |
|  | 5.2 | Interface Outputs | 57 |
|  | 5.3 | Conclusion | 60 |
| **6** | **REFERENCES** | | 62 |
|  | **APPENDIX A- BASE RESEARCH PAPER** | | 65 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| 4.1.1 | Soda Factory (Example) | 17 |
| 4.1.2 | Counting from the Top Diagram | 17 |
| 4.1.3 | Counting in the motion Diagram | 17 |
| 4.1.4 | Counting in a Bunch | 18 |
| 4.1.5 | Multiple Parts Counting | 18 |
| 4.1.6 | Labelled data | 19 |
| 4.1.7 | Generate Features | 20 |
| 4.1.8 | Training Output | 20 |
| 4.1.9 | Model testing output | 21 |
| 4.1.10 | Recent Count Display | 21 |
| 4.1.11 | Current Count Display | 22 |
| 4.1.12 | SCADA for small applications | 23 |
| 4.3.1 | Circuit Diagram | 24 |
| 4.2.1 | System Architecture | 28 |
| 4.2.2 | Use case Diagram | 31 |
| 4.2.3 | Activity Diagram | 33 |
| 4.2.4 | Sequence Diagram | 35 |
| 4.2.5 | Class Diagram | 37 |
| 4.2.6 | Data flow Diagram | 38 |
| 4.2.7 | ER Diagram\_1 | 40 |
| 5.1.1 | Current, Voltage Measurement on LCD | 57 |
| 5.1.2 | Voltage Vs Time Chart | 57 |
| 5.1.3 | Current Vs Time Chart | 58 |

|  |  |  |
| --- | --- | --- |
| 5.1.4 | Temperature value on LCD | 58 |
| 5.1.5 | Product vs Time Chart | 59 |
| 5.1.6 | Dc gear motor controlling and cooler fan controlling from Cayenne app online | 59 |
| 5.1.7 | Overall system monitoring using PI camera | 59 |
| 5.2.1 | Live Monitoring Interface | 56 |
| 5.2.2 | Interface Output | 60 |
|  |  |  |

.

**CHAPTER -1**

**INTRODUCTION**

**INTRODUCTION**

Nowadays, technology is advancing promptly and with the development of this technology the use of the internet is enormous. And this internet is further extended to connecting things or devices to interact with each-other intellectually so that it can provide useful and meaningful information. This phenomenon is called the Internet of Things (IoT). Undoubtedly it is a must-have technology for the modern world. We can’t imagine a single day without the internet. With the development in science and technology, many devices have been developed for the betterment of human life. India is a developing country. The country has become digitized in recent years. The number of internet users in our country has increased day by day. Technology is updating day by day as well. Various technologies were used in previous days. All systems may have some drawbacks. Some systems use conventional technology. Some systems use modern technologies, but they also have some limitations in monitoring which relates to the automation system. We will use raspberry pi for industrial automation and voltage, current, temperature. Product counter, DC gear motor control will be used in its controlling arena. For this Automation purpose, we will adopt the IoT technology. Thus, we can monitor or can control our entire system of the industry from anywhere in the world. For the industrial automation system, we used Raspberry Pi 3 SBC as the heart of the system. This device is an open source IoT platform. We used a voltage converter, Wi-Fi Module, breadboard power adapter – YW robot power MBV2, motor man life. Industrial YW robot power MBV2, motor man life.

The Internet of Things (IoT) is the system of physical gadgets, vehicles, home appliances, and different things installed with hardware, programming, sensors, actuators and availability. This system engages these things to interface and trade information, making open doors for progressively clear coordination of the physical world into PC based frameworks, achieving profitability changes and decreased human efforts. The applications of IoT devices are enormous i.e. Smart home, Health care, smart city and connected cars etc. IoT devices have been used to collect, monitor, evaluate and notify the Current time information.

The IoT technology has many benefits, such as:

(i)Greater control and independence in setting the home environment,

(ii)Time-saving and easy maintenance

(iii)Improved security

(iv)Reducing utility bills

**(v)Auto-alert in emergency situations.**

Automation is the creation of modern technology which helps to perform or complete a process or procedure with minimum human assistance. It also plays an important role in operating different equipment such as machinery, boilers, switching on telephone networks, processes in factories and heat-treating ovens.

Industrial Automation is the control of machinery and processes used in various industries by autonomous systems using technologies like robotics and computer networks. Internet based automation technologies also utilize Machine-learning techniques to discover the daily activities of the user and generate automation rules and actions that mimic these actions.

The basic objectives of this project:

* To implement an IoT based industrial automation system.
* To control the states of the industrial appliances.
* To control the electrical motors.
* To count & observe product’s quantity.
* To monitor overall system & upload to web server.

**Application:**

The application aspect underscores the swift pace of technological progress and the pervasive influence of the internet in contemporary society. It emphasizes the critical role of the Internet of Things (IoT) in enabling intelligent interactions between interconnected devices to deliver valuable and meaningful information. By highlighting IoT as a crucial technology for the modern world, the application section showcases its versatility in enhancing industrial automation, monitoring systems, and home automation solutions. Moreover, it accentuates the transformative potential of IoT in enabling remote monitoring and control of systems from any location globally, illustrating its practical utility in optimizing operational efficiency and user convenience.

**Technology:**

The technology segment delves deeper into the core concept of the Internet of Things (IoT) as a sophisticated system that integrates physical devices with advanced hardware, software, sensors, and connectivity capabilities. It elaborates on how IoT facilitates seamless data exchange and intelligent interactions among interconnected devices, paving the way for enhanced functionality and operational efficiency. The discussion further explores the diverse applications of IoT devices across various domains, including smart homes, healthcare, smart cities, and connected vehicles, showcasing the breadth of its impact on modern living. Additionally, the technology section articulates the key advantages of IoT technology, such as empowering users with greater control over their environments, streamlining maintenance tasks, bolstering security measures, reducing operational costs, and providing automated emergency notifications. Furthermore, it sheds light on the pivotal role of automation in modern technology, emphasizing its significance in optimizing processes, enhancing productivity, and minimizing human intervention in industrial operations.

**CHAPTER-2**

**LITERATURE SURVEY**

**LITERATURE SURVEY**

**2.1 Inference from Literature Survey**

Literature survey is the most important step in the software development process. Before developing the tool it is necessary to determine the time factor, economy and company strength. Once these things are satisfied, then the next step is to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need a lot of external support. This support can be obtained from senior programmers, from books or from websites. Before building the system, the above considerations are taken into account for developing the proposed system.

**Increased Efficiency and Productivity:**

Automation likely leads to higher production output and faster completion times within factories. Studies may point to specific technologies, like Programmable Logic Controllers (PLC), that contribute to efficiency gains

**Improved Safety and Quality:**

The literature might discuss reduced human error through automation, leading to a safer work environment. Increased consistency and precision in manufacturing processes due to automation could be a finding to expand more.

**Evolving Technologies and Communication Protocols:**

The survey might highlight advancements in areas like Industrial IoT (IIoT) and their role in industrial automation. Communication protocols like CAN bus may be explored for their efficiency and reliability in automation systems.

**Challenges and Considerations:**

The literature might discuss factors to consider when evaluating industrial automation projects, such as cost and return on investment. Cybersecurity risks and the need for secure communication within automated systems could be addressed.

**Future Directions:**

The survey may point to research areas in industrial automation, like further integration of AI and machine learning.

The importance of interoperability and data exchange between different automation systems might be a future direction Remember, these are just some possible inferences. The specific insights you glean from the literature survey will depend on the exact sources you reviewed and their focus areas.

**EXISTING METHODS**

**1. RFID, ZIGBEE, Bluetooth, GSM, and Wi-Fi Technologies:**

These are various wireless communication technologies used in home and industrial automation systems.

Each technology has its unique specifications and applications. For instance, RFID is commonly used for access control, ZIGBEE for short-range communication among devices, Bluetooth for connecting devices over short distances, GSM for remote communication via cellular networks, and Wi-Fi for high-speed internet connectivity.

**2. Cell Phone-Based Home Automation System (Arduino BT Board):**

Designed by R. Piyare, M. Tazil, et al., this system utilizes a standalone Arduino BT board.

Home appliances are connected to the input/output ports of the Arduino BT board relays.

The system provides flexibility and security while being cost-effective.

**3. Bluetooth-Based Home Automation System:**

Developed by Vignesh Govindraj, et al., this system is focused on converting traditional homes into smart homes using IoT and mobile applications.

However, it has limitations such as a limited Bluetooth range (10-20 meters), disconnection

Issues, and complexities in adding new users.

**4. GSM-Based Home Automation System:**

Two systems were designed using GSM technology:

"Design and Implementation of a GSM Based Remote Home Security and Appliance Control System" by G.M. Sultan Mahmud Rana, et al.

"Smart GSM Based Automation System" by Rozita Teymourzadeh, et al.

These systems enable remote control and monitoring of home appliances via mobile phones.

However, they have drawbacks such as network dependence, costliness, and susceptibility to errors in message formatting.

**5. Security and Surveillance System using Raspberry Pi:**

Developed by Syed Ali Imran Quadri, et al., this system utilizes ARM-11 architecture and Linux OS-based Raspberry Pi-3 board along with a USB camera and DC motor.

The system aims to provide security and surveillance to homes via the internet.

**6. Low-Cost Smart Home System:**

Developed by G. Mahalakshmi, et al., this system consists of hardware interface and software communication modules.

It offers features such as environmental monitoring and switching functionalities at a low cost.

**7. Advancements in the Presented System:**

The described system in the paper introduces advancements over previous research.

It utilizes GSM technology for communication between mobile phones and Arduino (or Raspberry Pi).

Unlike traditional systems using Arduino or PLC, Raspberry Pi is employed due to its advanced programming and controlling capabilities, larger storage capacity, and ability to handle multiple devices simultaneously.

Additionally, monitoring features have been added to enhance safety, making it beneficial for industry owners and supervisors.

**2.2 OPEN PROBLEMS IN EXISTING SYSTEM:**

**Limited Range and Connectivity Issues:**

The reliance on short-range communication technologies like Bluetooth limits the coverage area of automation systems. In larger homes or industrial facilities, where devices may be spread out over significant distances, this limitation can hinder the effectiveness of automation.

Connectivity issues, such as interference or signal attenuation, can disrupt communication between devices, leading to unreliable operation and potential system failures.

Moreover, the need for proximity between devices and their controllers (e.g., smartphones or central hubs) restricts the mobility of users and may inconvenience them.

**Dependency on Network Infrastructure:**

GSM-based systems depend on cellular network infrastructure for communication. While this allows for remote operation, it also introduces vulnerabilities associated with network outages, coverage gaps, or disruptions.

In areas with poor network coverage or during emergencies when network congestion is high, the reliability of GSM-based automation systems may be compromised.

Additionally, reliance on external networks exposes the system to potential security threats, such as interception of communication or unauthorized access to devices.

**Complexity in Adding New Users:**

Systems utilizing Bluetooth communication may require complex pairing procedures to add new users or devices. This process often involves manual configuration steps and may require technical expertise, leading to user frustration and errors.

In scenarios where multiple users need access to the automation system, managing and maintaining device pairings can become cumbersome and time-consuming.

**Costliness:**

The costliness of some automation systems, particularly those based on GSM technology, presents a barrier to adoption for users with limited financial resources.

High initial investment costs for hardware components, subscription fees for cellular services, and ongoing maintenance expenses can deter potential users from implementing automation solutions.

Cost considerations may also extend to factors such as power consumption, data usage, and software licensing fees, further impacting the overall affordability of the system.

**Susceptibility to Errors:**

Automation systems relying on SMS communication between mobile phones and micro controllers are vulnerable to errors in message formatting.

Users may inadvertently input incorrect commands or omit essential parameters, resulting in misinterpretation by the system and improper execution of tasks.

Error-prone communication channels can undermine the reliability and effectiveness of automation, leading to inefficiencies and potentially hazardous situations.

**Security Concerns:**

The open nature of wireless communication channels in automation systems raises concerns about cybersecurity threats.

Unauthorized access, interception of data transmissions, and manipulation of control commands are potential risks associated with insecure communication protocols.

Protecting sensitive information, ensuring data integrity, and implementing robust authentication and encryption mechanisms are critical for mitigating security risks in automation systems.

**Scalability:**

As automation systems expand to accommodate additional devices or functionalities, scalability becomes a crucial consideration.

Limitations in hardware capabilities, communication protocols, or system architectures may hinder the seamless integration of new components or the management of growing infrastructure.

Ensuring that automation solutions can scale effectively to meet evolving user needs and accommodate future technological advancements is essential for long-term viability and adoption.

**CHAPTER -3**

**REQUIREMENT ANALYSIS**

**REQUIREMENT ANALYSIS**

**3.1 FEASIBILITY STUDIES/RISK ANALYSIS OF THE PROJECT:**

**Feasibility Studies:**

**Technical Feasibility:**

The project's technical feasibility hinges on the successful integration of IoT technology with industrial automation systems. This involves ensuring that the hardware components (such as Raspberry Pi, sensors, actuators) are compatible and reliable for seamless operation.

Conducting thorough testing and prototyping will be essential to validate the technical feasibility of the system and identify any potential challenges early in the development process.

**Operational Feasibility:**

Assessing the operational feasibility involves evaluating how easily the system can be operated and adopted by users within the industrial setting. User training and documentation will play a crucial role in ensuring smooth adoption and utilization of the system.

Understanding the impact of the system on existing operational processes and workflows is vital to gauge the overall operational feasibility of the project.

**Economic Feasibility:**

Economic feasibility analysis will involve determining the cost-effectiveness of implementing the IoT-based industrial automation system. This includes assessing the return on investment (ROI) in terms of increased productivity, reduced manual efforts, and operational cost savings.

Conducting a cost-benefit analysis and comparing the projected benefits with the initial and ongoing costs will help in determining the economic viability of the project.

**Risk Analysis:**

**Technical Risks:**

Technical risks may include hardware failures, software bugs, compatibility issues between components, and scalability challenges as the system grows. Mitigation strategies such as regular maintenance, software updates, and scalability planning should be in place to address these risks.

**Operational Risks:**

Operational risks could involve system downtime, data security breaches, user errors, and resistance to change from employees. Implementing robust data backup procedures, security protocols, user training programs, and change management strategies can help mitigate these risks.

**Financial Risks:**

Financial risks may stem from budget overruns, unexpected expenses in hardware procurement or software licensing, and inadequate cost estimation. Regular financial monitoring, contingency planning, and transparent budget management are essential to mitigate financial risks.

**Regulatory Risks:**

Regulatory risks include non-compliance with industry standards, data privacy regulations, and cybersecurity requirements. Conducting regular compliance audits, staying updated on regulatory changes, and implementing necessary security measures can help mitigate regulatory risks.

**3.2 Software Requirements Specification Document:**

**Introduction:**

The software requirements specification document serves as a blueprint for the development of the IoT-based industrial automation system, outlining the functional and non-functional requirements to be met.

**Functional Requirements:**

The functional requirements detail the specific capabilities and features that the system must possess to meet the project objectives, including controlling appliances, monitoring data, and uploading information to a web server.

**Non-Functional Requirements:**

Non-functional requirements focus on aspects such as performance, reliability, security, scalability, and usability. These requirements ensure that the system operates effectively, securely, and efficiently while meeting user expectations.

**Software Requirements:**

**Operating System:**

The software should be compatible with the Raspberry Pi's operating system, such as Raspbian, to ensure smooth operation and integration with the hardware components.

**IoT Platform:**

The software needs to interface with the selected IoT platform for data management, communication, and remote monitoring capabilities in the industrial automation system.

**Web Server Software:**

A web server software stack comprising PHP, JavaScript, CSS, and HTML is required to host the server-side application for viewing and controlling industrial appliances.

**Programming Languages:**

The software development should utilize programming languages like PHP, JavaScript, and Python for server-side and device-side programming to enable data processing and control functionalities.

**API Integration:**

Integration with APIs, such as the Cayenne API for controlling devices and the Canny library API for cooler fan control, is necessary to facilitate communication and interaction with external systems.

**Hardware Requirements:**

**Raspberry Pi 3 B+:**

The Raspberry Pi 3 B+ serves as the central controlling unit for the industrial automation system. It provides processing power, GPIO interfaces for sensor and actuator connections, and networking capabilities for communication with external devices and the IoT platform.

**Sensors:**

Various sensors are essential for monitoring critical parameters in the industrial environment. This may include voltage sensors, current sensors, temperature sensors (e.g., DS18B20 digital thermometer), and any other sensors required for data collection and analysis.

**Actuators:**

Actuators such as DC gear motors and cooler fans are necessary for controlling industrial appliances based on the data received from sensors. These actuators enable the system to take automated actions in response to environmental conditions or user inputs.

**Networking Components:**

Hardware components like Wi-Fi modules are crucial for establishing wireless connectivity between the Raspberry Pi and other devices in the network. This enables data transmission, remote monitoring, and control of industrial processes from anywhere with network access.

**Power Supply Setup:**

A reliable power supply setup is vital to ensure continuous operation of the system. This includes powering the Raspberry Pi, sensors, actuators, and any other connected devices. Additionally, provisions for alternative power sources like batteries can provide backup in case of power outages.

**Display and Input Devices:**

Components like LCD monitors can be used to display real-time data, system status, and alerts to users. Input devices such as keyboards, touchscreens, or buttons may also be incorporated for user interaction, enabling manual control or configuration of the industrial automation system.

**Circuit Components:**

Various circuit components like resistors, capacitors, transistors, and connectors may be required for interfacing sensors, actuators, and other electronic devices with the Raspberry Pi. Proper circuit design and assembly are essential for reliable operation and data accuracy.

**Interfaces:**

User interface and communication interfaces are defined in the document, detailing how users will interact with the system and how data will be exchanged between devices. A user-friendly web interface and reliable communication protocols are essential for system usability.

**Constraints:**

Budget constraints and compatibility limitations are identified as constraints that may impact the project's development and implementation. Managing these constraints effectively is key to ensuring the project's success within the defined scope and resources.

**CHAPTER -4**

**DESCRIPTION OF THE PROPOSED SYSTEM**

**DESCRIPTION OF THE PROPOSED SYSTEM**

The proposed system is an IoT-based industrial automation system designed using Raspberry Pi as the central controlling unit and integrated with various sensors, actuators, and networking components. The system aims to monitor, control, measure, and provide real-time data updates for automatic industries, enhancing productivity and efficiency. Key components and features of the proposed system include:

**Hardware Components:**

* Raspberry Pi 3 B+ as the controlling unit.
* Sensors for monitoring parameters like voltage, current, and temperature.
* Actuators such as DC gear motors and cooler fans for controlling industrial appliances.
* Networking components like Wi-Fi modules for wireless connectivity.
* Power supply setup with provisions for alternative power sources.

**Software Integration:**

* Utilization of IoT platforms for data management and communication.
* Web server software stack for remote monitoring and control.
* Programming languages like Python for system development.
* Integration with APIs like Cayenne for device control.

**System Functionality:**

* Real-time monitoring of industrial processes through web servers.
* Product counting and observation of quantities.
* Power measurement and temperature monitoring.
* Device control through the Cayenne system.
* Remote access and control from anywhere in the world.

**ADVANTAGES OF THE PROPOSED SYSTEM:**

**Enhanced Monitoring and Control:**

The system allows for real-time monitoring and control of industrial processes, enabling operators to make informed decisions and adjustments as needed.

**Improved Productivity and Efficiency:**

By providing data insights and automation capabilities, the system can enhance productivity and efficiency in industrial operations, leading to optimized performance.

**Remote Accessibility:**

The ability to access and control the system from anywhere in the world offers convenience and flexibility for monitoring and managing industrial processes.

**Data Analysis and Reporting:**

The system facilitates data analysis and reporting, allowing for the generation of monthly reports and performance evaluations to drive continuous improvement.

Scalability and Customization:

The modular design of the system allows for scalability and customization based on specific industrial requirements, ensuring adaptability to different settings and needs.

**4.1 SELECTED METHODOLOGY OR PROCESS MODEL**

The system is proposed to have the following modules:

* Product Counting
* Power Detection Module
* System Cooling
* System Controlling

**PRODUCT COUNTING :**

Automatic counting machines are very essential for correct packing in manufacturing industries. Currently, industries count either mechanically or through weight. Mechanical counting is restricted by size and shape of the product and it is often time-consuming.

Weight based counting assumes that each part has the exact same weight and uses a weight average to count. Even the most sophisticated manufacturing systems produce parts with slight variations in size and shape. These are even more pronounced for materials like wood and rubber where density changes by up to 50%. In addition to correct packing, these vision-based counters will be used to estimate the defective parts in a certain batch of production.

Consider if there are a higher number of defective parts, we can assume that something might be wrong with the production units. This data can also be used to improve the quality of production and thus industry can make more products in less time. So our adaptable counters are evolving as a solution to the world's accurate and flexible counting needs.

The Adaptable Counter is a device consisting of a Raspberry Pi 4 and camera module, and the counting process is fully powered by FOMO. So it can count faster and more accurately than any other method. Adaptable counters are integrated with a cool looking website.

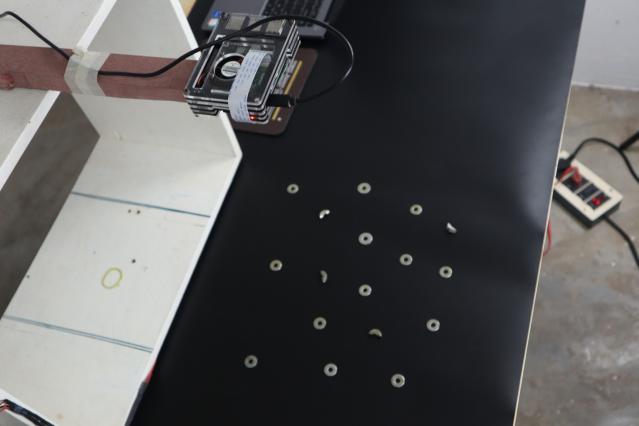


**Fig 4.1.1 Soda Factory (Example)**

**USE-CASES :**

These sample use-cases can be applied to any industry.

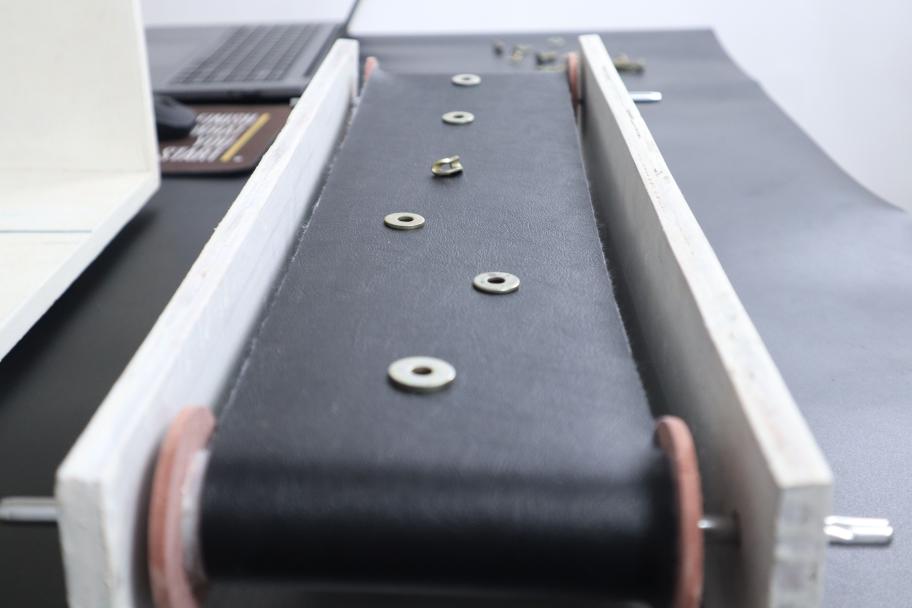
1. **Counting from the Top**



**Fig 4.1.2 Counting from the Top Diagram**

In this case, we are counting defective and non-defective washers.

1. **Counting in Motion**



**Fig 4.1.3 Counting in Motion Diagram**

In this case, we are counting bolts and washers and faulty washers passing through the conveyor belt.

**3. Counting in a Bunch**

In this case, we are counting the bunch of lollipops.



**Fig 4.1.4 Counting in a Bunch**

1. **Multiple Parts Counting**



**Fig 4.1.5 Multiple Parts Counting**

In this case, we are counting multiple parts such as Washers and Bolts.

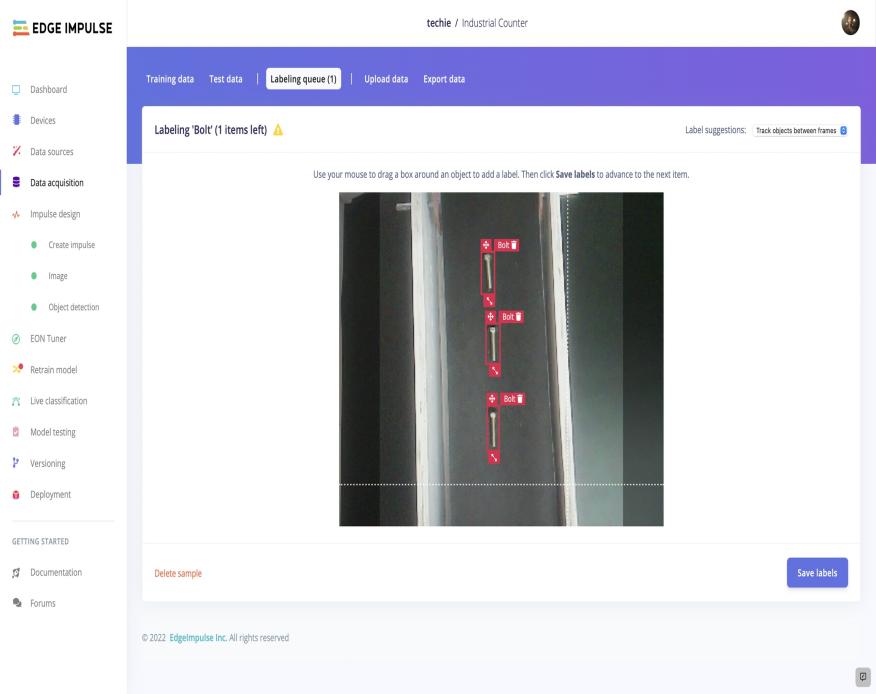
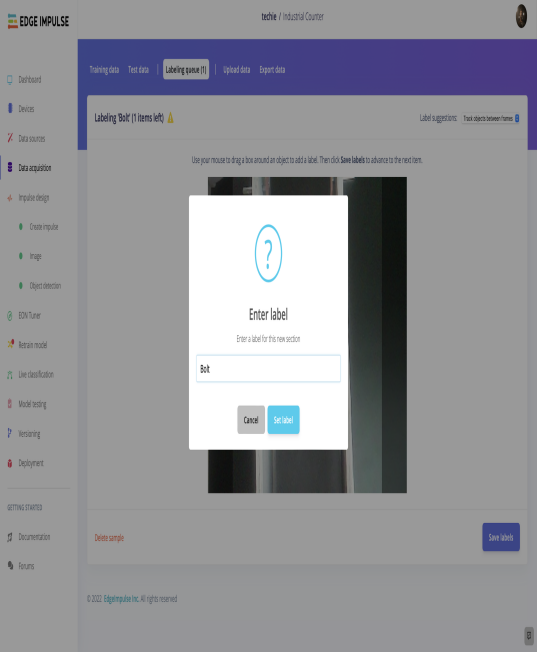
**Data Acquisition**

Every machine learning project starts with data collection. A good collection of data is one of the major factors that influences the performance of the model. Make sure you have a wide range of perspectives and zoom levels of the items that are being collected. You may take data from any device or development board, or upload your own datasets, for data acquisition. As we have our own dataset, we are uploading them using the Data Acquisition tab.Simply navigate to the Data acquisition tab and select a file to upload. After that, give it a label and upload it to the training area. Edge Impulse will only accept JPG or PNG image files. Convert it to JPG or PNG format using a converter if you have any other formats.

In our case we have four labels - Washer, Faulty Washer, Lollipop, Bolt. We have uploaded all the collected data for these four different classes. Therefore, the computer will only recognize these items while counting. You must upload the dataset of other objects if you wish to recognize any other objects. The more data that neural networks have access to, the better their ability to recognize the object. This is our counting setup (Just attached the Adaptable counter on the top of a small wooden plank).

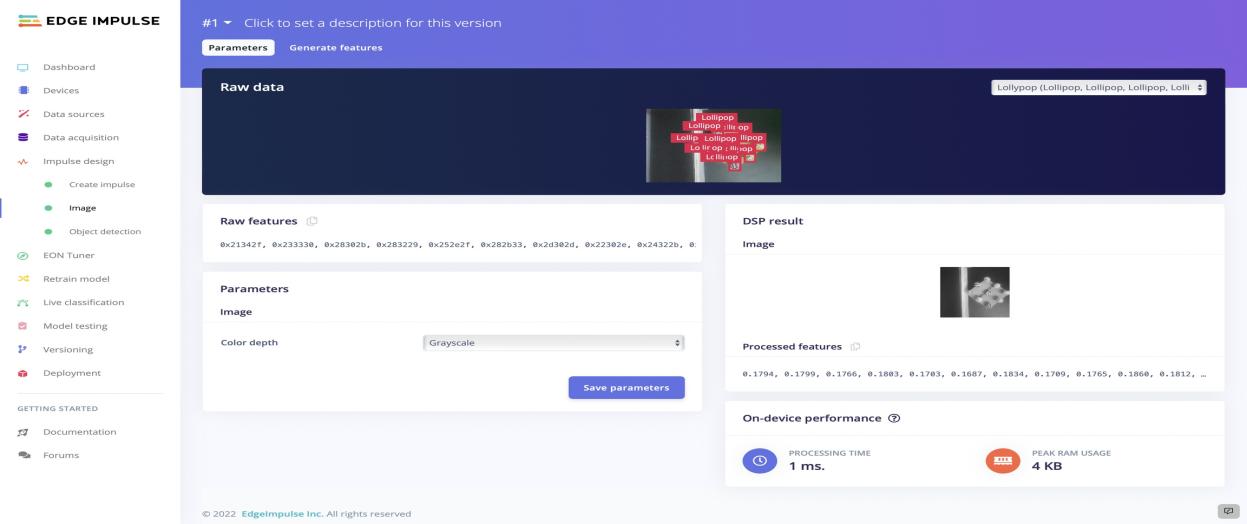
**Labeling Data**

You may view all of your dataset's unlabeled data in the labeling queue. Adding a label to an object is as simple as dragging a box around it. Edge Impulse attempts to automate this procedure by running an object tracking algorithm in the background in order to make life a little easier. If you have the same object in multiple photos the box moves for you and you just need to confirm the new box. Drag the boxes, then click Save labels. Continue doing this until your entire dataset has been labeled.



**Fig 4.1.6 Labelled data**

After generating the features for our data, we can see the individual measurable properties of the data represented in a 3-dimensional space. The below figure shows the features generated from our dataset. The generated features are well distinguishable.

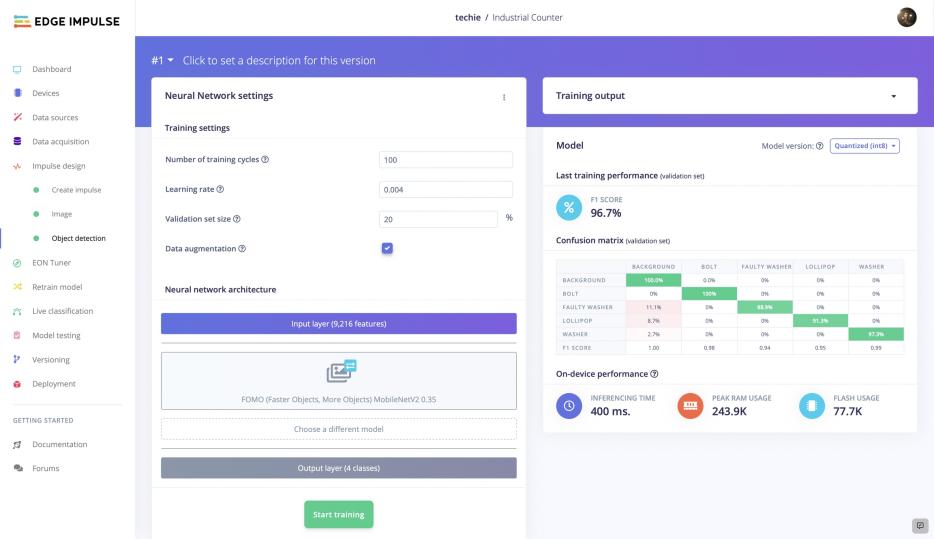


**Fig 4.1.7 Generate Features**

Now it's time to start training the machine learning model. Generating a machine learning model from scratch requires great time and effort. Instead, we will use a technique called "transfer learning" which uses a pre-trained model on our data. That way we can create an accurate machine learning model, with fewer data inputs. Head over to the Object detection tab for the model generation.

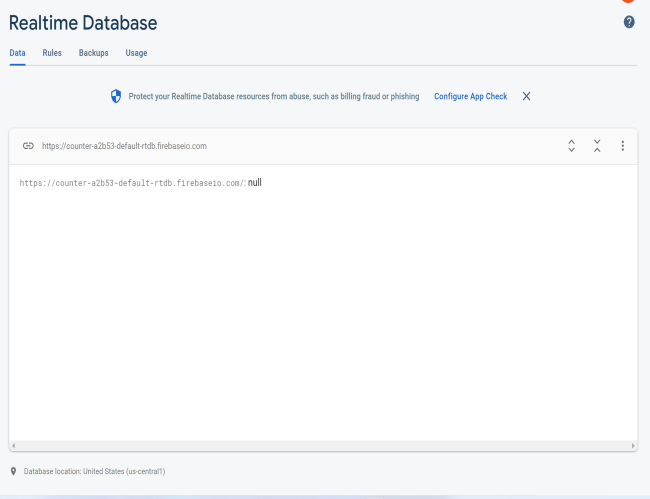
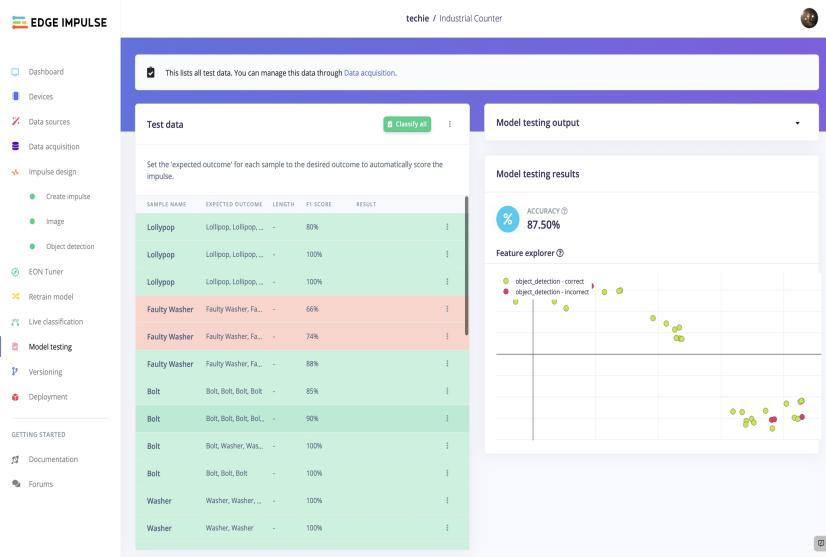
In this case we are using the FOMO algorithm to train the model. So change the object detection model to **FOMO (Faster Objects, More Objects) MobileNetV2 0.35** and change the neural network settings as shown in the image. FOMO is a novel machine learning algorithm created by Edge Impulse, specifically designed for highly constrained devices. It works very well with the Raspberry Pi 4.

Now start training. After the model is done you'll see accuracy numbers below the training output. We have now trained our model with a training accuracy of 96.7%, pretty good.



**Fig 4.1.8 Training Output**

To validate your model, go to **Model testing** and select **Classify all**. Here we hit 87.5% accuracy, which is great for a model with so little data.

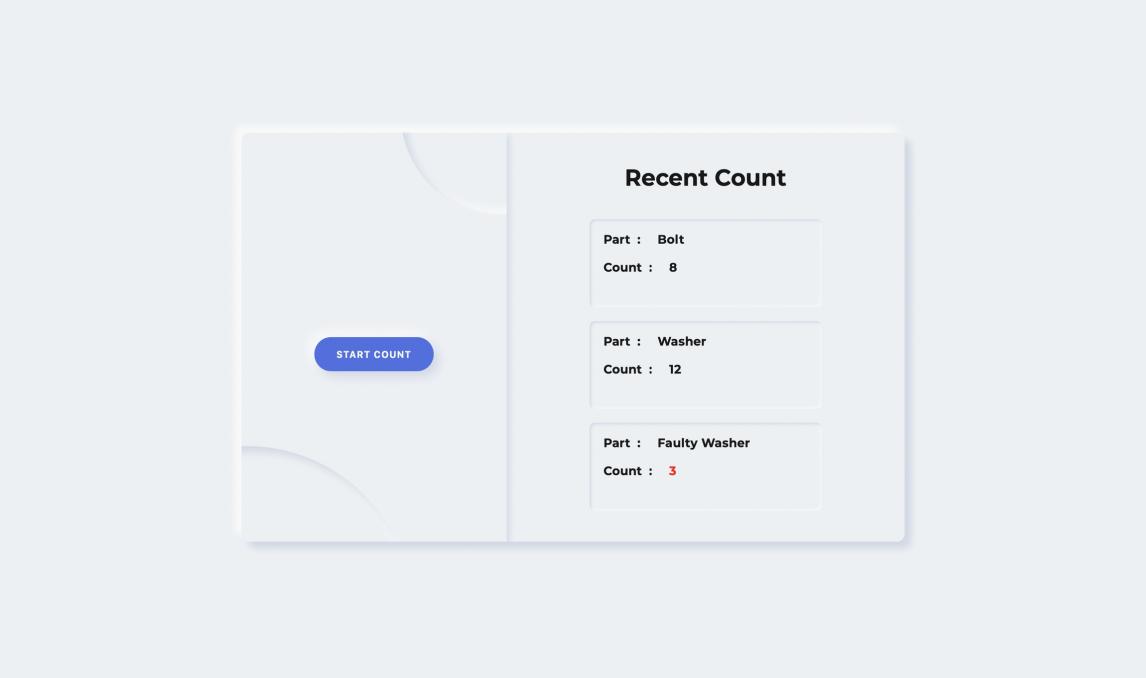


**Fig 4.1.9 Model testing output**

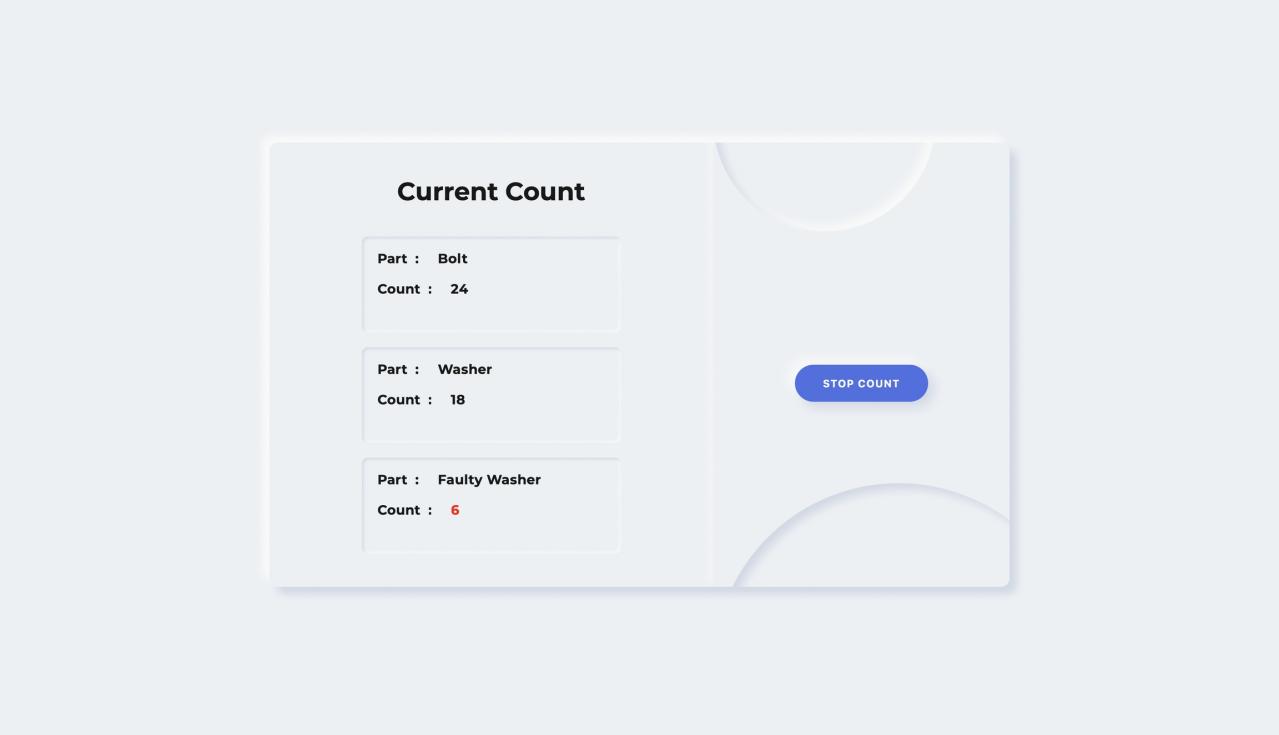
For use with only user-based authentication we can create the following configuration, that should be added in our Python code

**Website**

web page is created using HTML, CSS and JS to display the count in real time. The data updated in Firebase is reflected in the web page in real time. The web page displays **Recent Count** when the counting process is halted, and displays **Current Count** whenever the counting process is going on.



**Fig 4.1.10 Recent Count Display**



**Fig 4.1.11 Current Count Display**

For ease of testing, we'll use small conveyor belt to prototype the system so that objects pass into and out of the camera's field of view. This way we can test both scenarios as mentioned: the number of objects in view at a distinct moment, and the total count of objects that have moved past the camera.

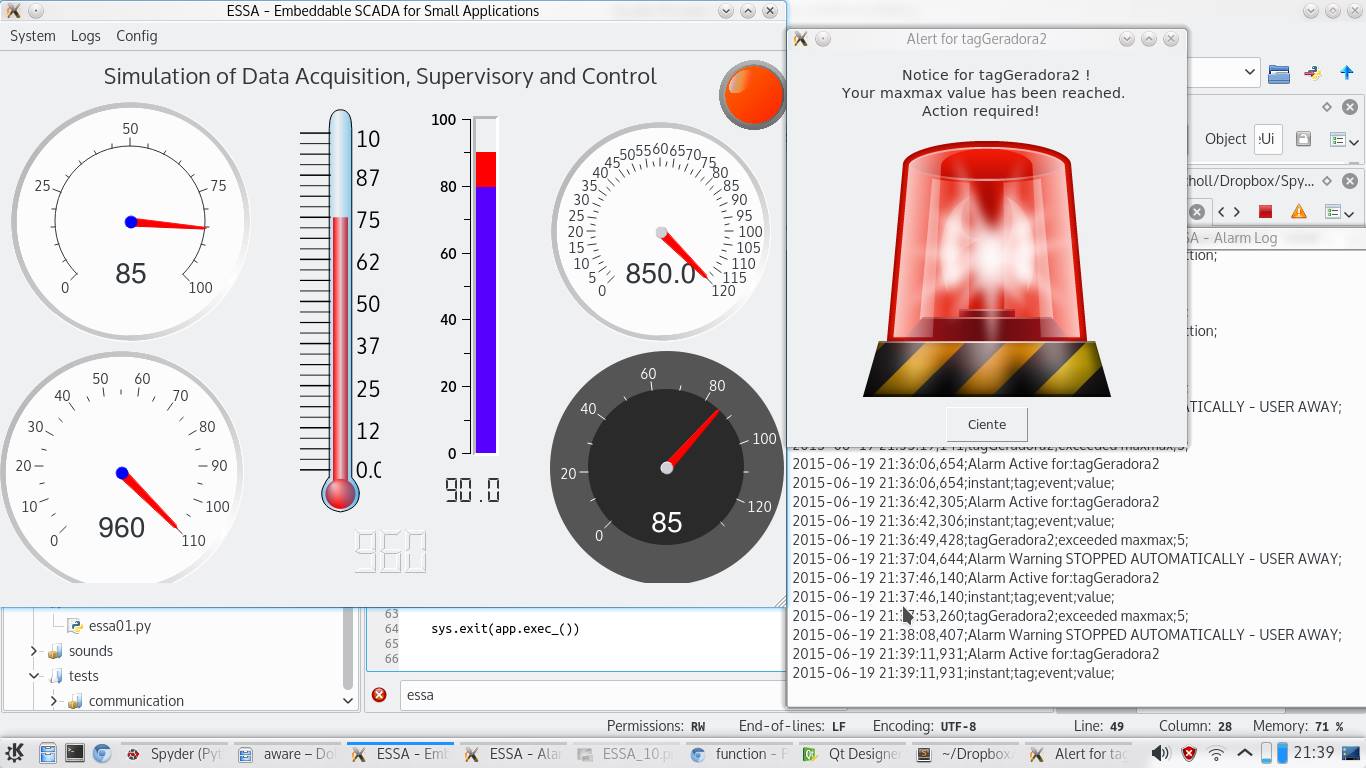
**POWER DETECTION MODULE:**

We used the INA219 power module to get the power unit measurement. The INA219 module is used to detect shunt voltage and shunt current across the motor. So we connect it to the motor driver module so that we can measure machine load power and load voltage current. And using python urllib2 HTTP Library to send the data to our EB panel using web API endpoints. On the other hand we can connect the power module to the product counter so that we can measure the overall performance to that endpoint.

**SYSTEM COOLING:**

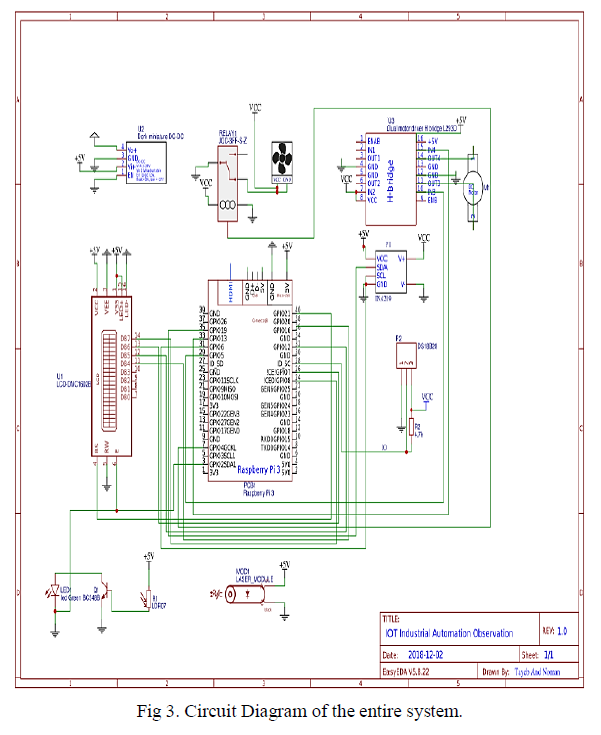
We have an asynchronous task in our system and that is cooler controlling, we have an infinite while loop in our python code that checks motor state continuously. Whenever we enable the motor switch from Cayenne, the module starts the GPIO output so that our resultant value would become high. On the other hand, the GPIO output remains low and our cooler remains low.

The IoT Based Industrial Automation System, the flow of functions is designed to provide a seamless and efficient process for monitoring, controlling, and optimizing industrial operations. Here is a more detailed elaboration of the flow of functions within the system.



**Fig 4.1.12 SCADA for small applications**

**4.3 CIRCUIT DIAGRAM**



**Data Acquisition and Sensor Integration:**

The system begins by collecting data from various sensors deployed in the industrial environment. Sensors such as the INA219 power measuring unit, DS18B20 digital thermometer, and laser module for product counting continuously gather data on key parameters like voltage, current, temperature, and production quantities.

**Data Processing and Analysis:**

The Raspberry Pi 3 B+ serves as the central processing unit, receiving raw data from the sensors. It processes the incoming data, performs calculations, and applies algorithms to analyze the information. This analysis helps in identifying patterns, trends, and anomalies in the industrial processes.

**Decision Making and Control Actions:**

Based on the analyzed data, the Raspberry Pi makes decisions regarding control actions. It triggers commands to the actuators, such as adjusting motor speeds, activating or deactivating devices, and regulating operational parameters. These control actions are aimed at optimizing the performance and efficiency of the industrial setup.

**Communication with Cloud Server and Data Storage:**

The processed data is transmitted from the Raspberry Pi to the cloud server using the Wi-Fi module. The cloud server acts as a centralized platform for storing and managing the data. It ensures data integrity, security, and accessibility for authorized users.

**Web Interface for Monitoring and Control:**

Users can access a web interface hosted on a web server to monitor the real-time status of the industrial system. The web interface displays graphical representations of data, including voltage vs. time, current vs. time, and temperature values. Users can also interact with the system, view statistics, and control devices remotely through the web interface.

**Integration with Cayenne Application for Device Control:**

The system integrates the Cayenne application for seamless device control. Users can utilize the Cayenne app to remotely control the cooler fan and DC gear motor through the Wi-Fi network. This integration enhances user experience and provides a user-friendly interface for managing devices.

**Continuous Monitoring, Analysis, and Optimization:**

The system enables continuous monitoring of industrial processes, allowing users to track performance metrics and make informed decisions. By analyzing historical data and trends, users can optimize operations, improve efficiency, and identify areas for enhancement in the industrial setup.

**4.2 DESCRIPTION OF SOFTWARE FOR IMPLEMENTATION AND TESTING PLAN OF THE PROPOSED MODEL/SYSTEM:**

Operating System (Raspbian): Raspbian OS is utilized on the Raspberry Pi for device control and data processing. It provides a stable platform for running Python scripts and interacting with hardware components.

**CAYENCE Software:**

The cloud server created by CAYENCE software acts as a bridge between the industry and control devices. It stores and manages data transmitted by the Raspberry Pi, facilitating remote monitoring and control functionalities.

**Python Programming:**

Python is employed for coding purposes to interact with sensors, process data, and communicate with the web server. Python scripts enable real-time data processing, control logic implementation, and communication with external systems.

**Web Server Integration:**

The system integrates with a web server to display real-time data on voltage, current, temperature, and product count. This integration enables remote monitoring, data visualization, and analysis for informed decision-making.

**Testing Plan:**

The testing plan involves validating the functionality of each component, ensuring accurate data transmission, reliable control operations, and responsive monitoring capabilities. Testing will verify sensor accuracy, communication reliability, and system performance under various scenarios.

**Functionality:**

The software components include the Raspbian OS, CAYENCE software, Python programming, and web server integration for data processing, storage, and remote access.

**How It Works**:

Raspbian OS runs on Raspberry Pi to execute Python scripts for interacting with sensors and controlling devices. CAYENCE software facilitates cloud storage and remote access to data. Python scripts process sensor data and communicate with the web server to display real-time information for monitoring and analysis.

**4.3 PROJECT MANAGEMENT PLAN:**

**Resource Allocation:**

Allocate resources effectively for hardware procurement, software development, and testing phases to ensure smooth project execution and timely completion.

**Timeline:**

Establish a project timeline with clear milestones for hardware setup, software implementation, testing, and deployment. This timeline ensures progress tracking and adherence to project deadlines.

**Risk Management:**

Identify potential risks such as hardware failures, software bugs, or communication issues. Develop mitigation strategies to address these risks and minimize project disruptions.

**Collaboration:**

Foster collaboration among team members for effective communication, task delegation, and progress tracking. Regular team meetings and updates ensure alignment and coordination throughout the project.

**Documentation:**

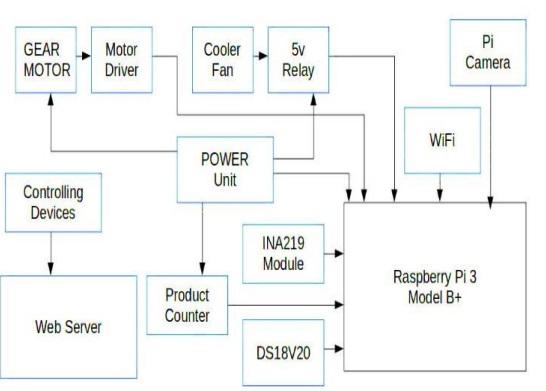
Maintain detailed documentation of hardware configurations, software codes, testing results, and project updates. Comprehensive documentation serves as a reference for troubleshooting, future enhancements, and knowledge transfer within the team.

**Functionality:**

The project management plan focuses on resource allocation, timeline establishment, risk management, collaboration, and documentation for successful project execution.

**How It Works:**

Resources are allocated efficiently for hardware procurement and software development. A timeline with milestones ensures project progress tracking. Risk management strategies mitigate potential issues. Collaboration among team members fosters effective communication and task delegation. Detailed documentation serves as a reference for troubleshooting and future enhancements.

**4.2.1 ARCHITECTURE/OVERALL DESIGN OF PROPOSED SYSTEM**

**Fig. 4.2.1 ARCHITECTURE/OVERALL DESIGN OF PROPOSED SYSTEM**

The overall block diagram of the system is shown in “Fig. 4.4.1”. Every systems and subsystems are depicted from where its operating procedure can be described. Various sensor modules are connected with system. LCD which uses as a monitor in this project. Voltage, current, temperature will be shown on it and internet using Wi-Fi module and sends the values to a remote IoT server where the owner or connected person can view the state from anywhere of the world.

**Raspberry Pi 3 Model B+ [Raspberry Pi]:** The Raspberry Pi is a single-board computer that acts as the brain of the system. It runs the operating system and controls the other devices through the connected controllers.

**Web Server:** The web server software running on the Raspberry Pi allows users to remotely monitor and control the system through a web interface over WiFi.

**Power Unit:** The power unit supplies power to the Raspberry Pi and other devices in the system.

**Motor Driver:** The motor driver controls the speed and direction of the motor based on signals from the Raspberry Pi.

**Motor:** The motor provides mechanical power to the system. The specific type of motor used would depend on the application.

**Gear:** The gear may be connected to the motor and alters the speed or torque delivered by the motor.

**Camera:** The camera captures images or video of the system or its surrounding environment.

**Cooler:** The cooler cools a specific part of the system, possibly the motor or another heat-generating component.

**Relay:** Relays are electronic switches that can be used to turn devices on and off based on signals from the Raspberry Pi.

**Product Counter:** The product counter keeps track of the number of items produced by the system.

**THE SYSTEM INCLUDE TWO SENSORS:**

**INA219 Module:** The INA219 module is a current sensor that can be used to measure the current flowing through a circuit. This could be used to monitor the power consumption of the system or a specific device.

**DS18V20:** The DS18V20 is a temperature sensor that can be used to measure the temperature of a specific location in the system.

**UMLS(UNIFIED MODELLING LANGUAGE)**

The Unified Modelling Language allows the software engineer to express an analysis model using the modelling notation that is governed by a set of syntactic semantic and pragmatic rules**.**

A UML system is represented using five different views that describe the system from a distinctly different perspective. Each view is defined by a set of diagrams, which is as follows.

**User Model View**

This view represents the system from the user's perspective. The analysis representation describes a usage scenario from the end-users perspective.

**Structural model view**

In this model the data and functionality are arrived from inside the system. This model view models the static structures.

**Behavioural Model View**

It represents the dynamic of behavioural as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.

**Implementation Model View**

In this the structural and behavioural as parts of the system are represented as they are to be built.

**Environmental Model View**

In this the structural and behavioural aspects of the environment in which the system is to be implemented are represented.

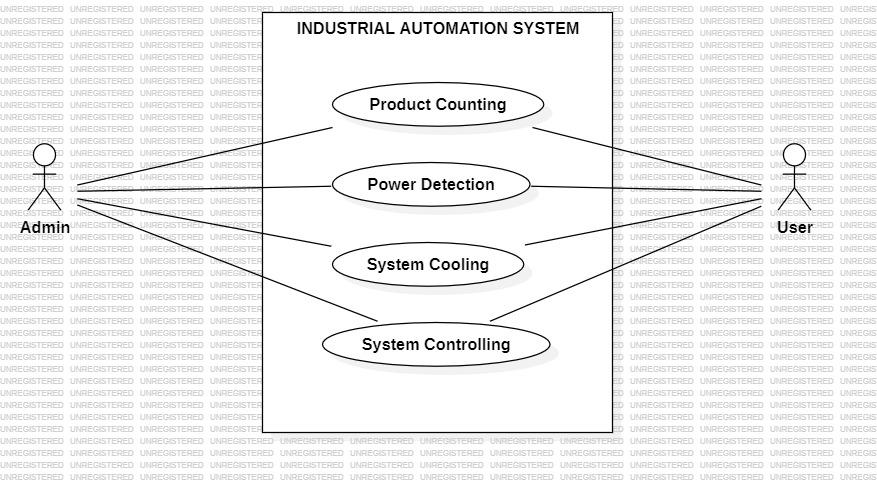
UML is specifically constructed through two different domains they are:

* UML Analysis modelling, this focuses on the user model and structural model views of the system.
* UML design modelling, which focuses on the behavioural modelling, implementation modelling and environmental model views.

Use case Diagrams represent the functionality of the system from a user’s point of view. Use cases are used during requirements elicitation and analysis to represent the functionality of the system. Use cases focus on the behaviour of the system from an external point of view.

Actors are external entities that interact with the system. Examples of actors include users like administrators, bank customers …etc., or another system like a central database.

**USE CASE DIAGRAM:**

This use case diagram contains two actors Admin and User. The User performs operations like interaction with the system from the web server. Performing usage scenarios described in the analysis representation. Providing input for requirements elicitation and analysis. Interacting with the system's functionality from an external point of view, as represented in the use case diagrams. Engaging in dynamic interactions with the system, as depicted in the behavioural model view.

**Fig 4.2.2 Use Case Diagram for the Industrial Automation System**

1. **Actors:**

User: Interacts with the system to monitor, control, and analyze data.

Admin: Devices within the industrial automation system that collect sensor data and this data has been monitoring by the admin and the admin is responsible for holding and maintaining the user details like username, password

1. **Use Cases:**

Monitor System: Allows the User to monitor the industrial system in real-time.

Control Devices: Enables the User to send control signals to IoT Devices for operation.

Analyse Data: Permits the User to analyse sensor data and monitoring information.

Receive Alerts: Notifies the User about any detected motion or gas conditions.

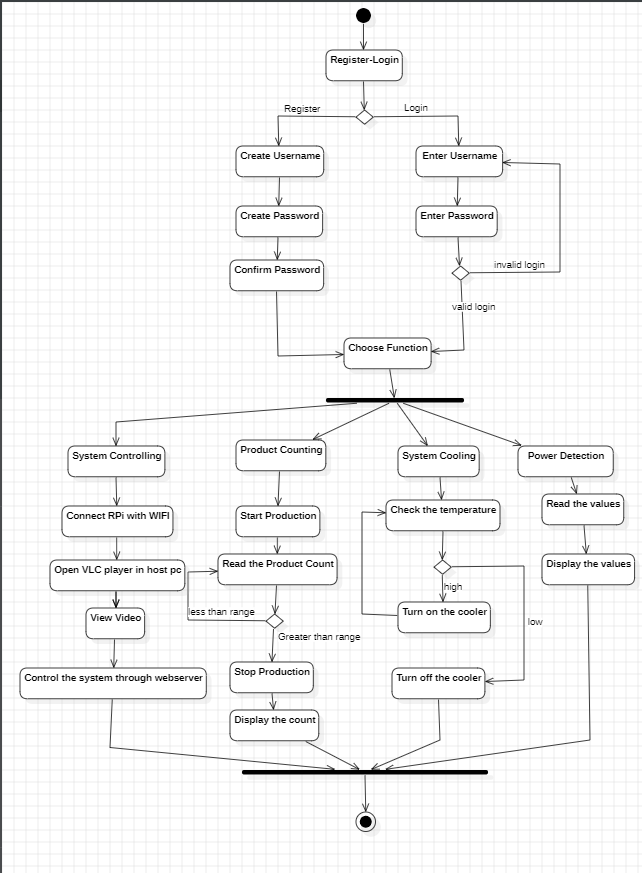
Update Information: Provides updated information about the industrial system to the User.

Connect to Web Server: Establishes a connection between the system and the Web Server for data storage and retrieval.

1. **Use Case Relationships:**
   * User interacts with all the mentioned use cases.
   * IoT Devices are involved in the "Control Devices" use case.
   * Web Server is connected to the system for data exchange and storage.
2. **Description:**
   * The User can monitor the system, control devices, analyse data, receive alerts, and update information through the system interface.
   * IoT Devices receive control signals from the User and provide sensor data for monitoring.
   * The Web Server acts as a bridge between the system and external users, facilitating data storage and retrieval.

**ACTIVITY DIAGRAM**

The activity diagram visually presents a series of actions followed by the user from the time he/she logged into the system. This gives the reader a quick understanding about the system without much knowledge about the system. We can depict both sequential processing and concurrent processing of activities using an activity diagram i.e., an activity diagram focuses on the condition of flow and the sequence in which it happens. “Fig 4.4.4” gives the activity diagram of our system.



**Fig 4.2.3 Activity Diagram**

The user first registers into the system by creating username and password for the first and then after the user can login into the system by entering username and password. If the user enters an invalid username or password he/she must enter the username or password. If the username and password are then the user can choose functions like system controlling, product counting, and system cooling and Power Detection.

When the user chooses the System Control he/she ensures to connect RPi with the Wi-Fi or internet then he need to open the VLC player in the host pc/mobile and then click on the start/pause button to view the video and then he/she can control the system through web server/RealVNC server.

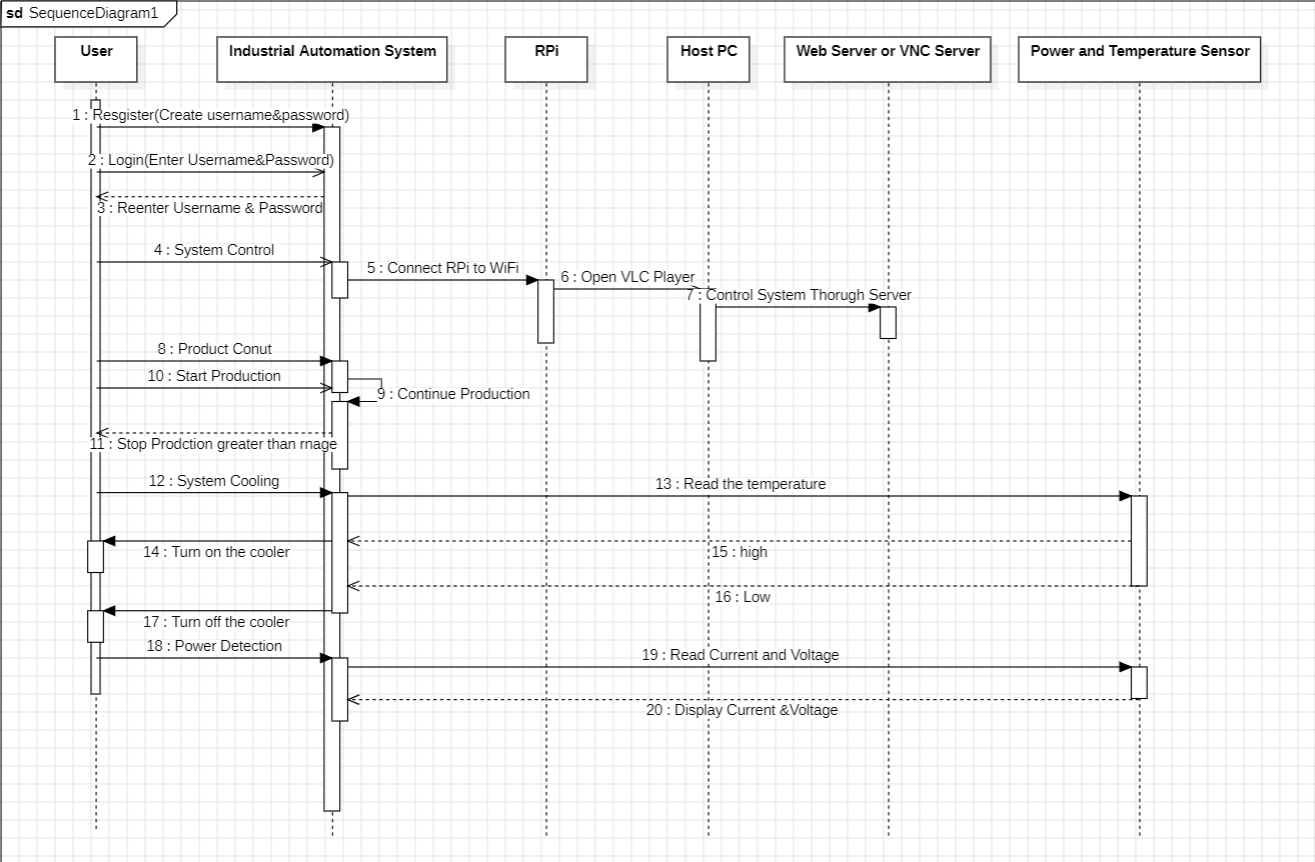
When the user chooses the Product Count Function they must start the production and read the product count whenever the product crosses the threshold value he/she must stop production and the count is displayed on the screen otherwise the product count continues to count the production.

If the user chooses the System Cooling function he/she first checks the temperature of the system then if the temperature of the system is higher than expected then the cooler fan must be turned on when the temperature is low then the cooler fan must be turned off.

The Power Detection Function is used to display the current and voltage values of the system.

**SEQUENCE DIAGRAM**

A sequence diagram is a visual representation of how objects interact in a specific scenario. It depicts the messages exchanged between objects chronologically, showcasing the order of events**.** Sequence diagrams are particularly useful for understanding complex interactions between objects, making them valuable tools in software design and system analysis. They help identify potential bottlenecks, clarify message flow, and ensure a clear understanding of how objects collaborate to achieve a specific goal. “Fig4.4.5” gives the sequence diagram of the Industrial Automation System.



**Fig 4.2.4 Sequence Diagram for Industrial Automation System**

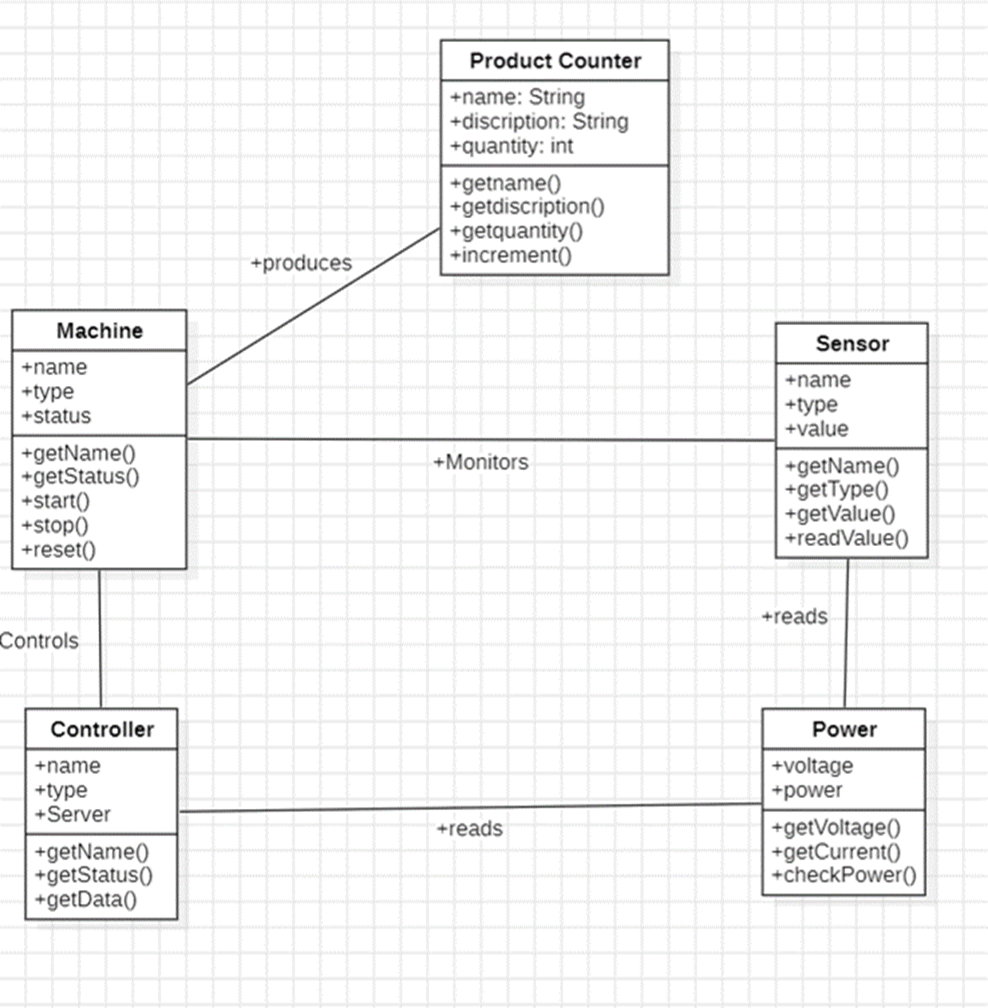
The Sequence Diagram has five objects: user, RPi, Industrial Automation System, Host PC, and Web Server/RealVNC, Power and Temperature Sensor. The login into the system by creating username and password by Create Username and Password Function. Them the user login into the system using Login function and then system validates the details entered by the user and if the details are invalid it sends “Renter Username and Password” message to the user and after successful login the user can choose different function like system control for the Industrial Automation System sends connect RPi to wifi and then the user open the VLC player in their system and can view various appliances in the industry and can control them through the web server or RealVNC. When the user wants to know the product count they send a product count message to the system and the system reads the product count and displays the count on the screen, if the count of the product is greater than the limit the machine stops the production and displays product count else it continues to count the product.

When the user want to cool the system he/she want send message to the system to cool the system then the system sends a message to the power or temperature sensor to read the temperature message then power or temperature sensor sends a display temperature message to the system then the user decides to turn on/off the cooler fan based on the displayed temperature. When the user wants to know the reading of the current or voltage value of the machine he/she sends a detect power message to the system then the system sends a read current or voltage message to the power or temperature sensor then the power or temperature sensor sends a display current or voltage message to the user.

Then the user wants to continue to be in the system or leave the system.

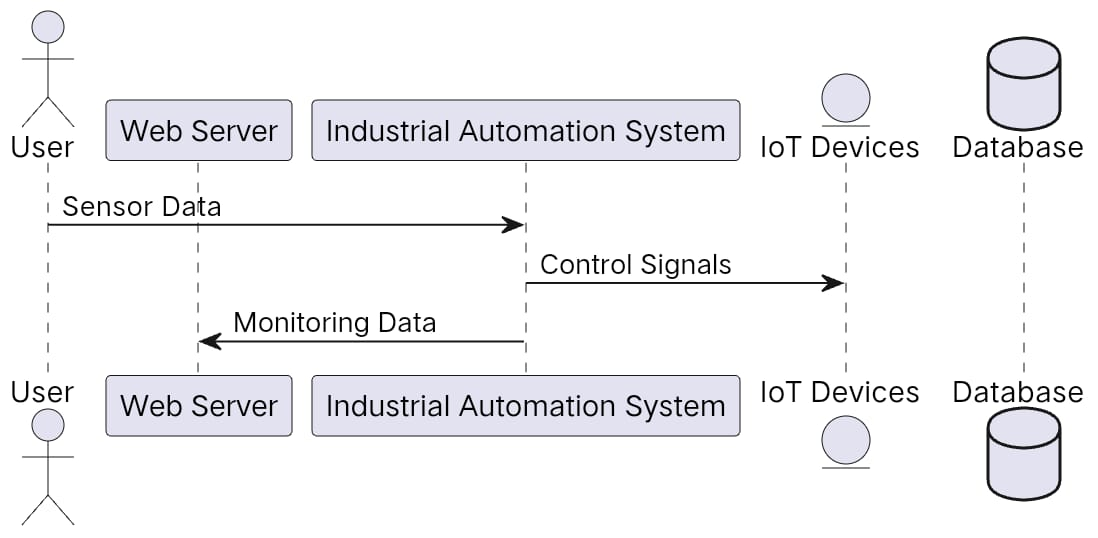
**CLASS DIAGRAM FOR INDUSTRIAL AUTOMATION SYSTEM**

This class diagram contains five classes that are Product counter, Machine, Controller, Sensor, and Power. Product Counter will perform the operations like describing the product count, quantity. Machine class performs operations like start, stop and rest the machine. The sensor class performs operations like reading the value from the machine. The controller class gets the value from the machine and gets the current status of the machine. The power class performs operations like reading power and voltage generated by the system and machines and this diagram shows the relationship between these classes like a machine produces products.



**Fig 4.2.5 Class Diagram for Industrial Automation System**

**DATA FLOW DIAGRAM**



**Fig 4.2.6 Data Flow Diagram for Industrial Automation System**

**Processes:**

Monitor System: This process involves monitoring the industrial system in real-time. It includes receiving sensor data from IoT Devices, displaying monitoring information to the User, and updating the system status.

Control Devices: This process allows the User to send control signals to IoT Devices for operation. It involves receiving control commands from the User, transmitting signals to the IoT Devices, and executing the desired actions.

Analyze Data: This process involves analyzing sensor data and monitoring information. It includes processing data for trends, generating reports, and providing insights to the User.

Receive Alerts: This process notifies the User about any detected motion or gas conditions. It involves monitoring sensor data for specific events, triggering alerts, and notifying the User.

Update Information: This process provides updated information about the industrial system to the User. It includes fetching real-time data, updating system status, and displaying the latest information.

Connect to Web Server: This process establishes a connection between the system and the Web Server for data storage and retrieval. It involves transmitting data to the Web Server, receiving commands from external users, and managing data exchange.

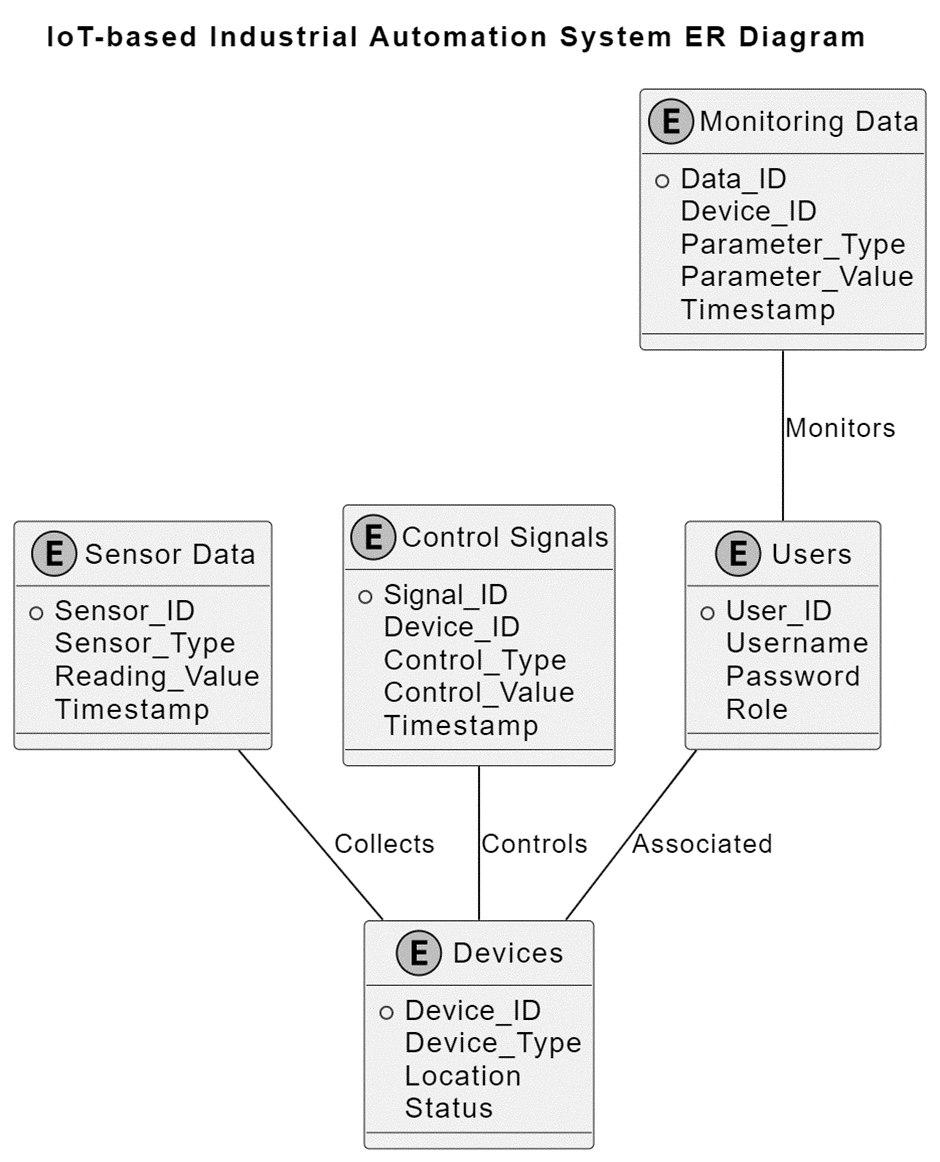
**Data Stores**:

* Sensor Data Store: Stores sensor data collected from IoT Devices.
* Control Data Store: Stores control signals sent to IoT Devices.
* Monitoring Data Store: Stores monitoring information and analysis results.
* Alerts Data Store: Stores information related to detected alerts.
* System Information Store: Stores system status and updated information.
* Web Server Data Store: Stores data exchanged with the Web Server.

**Data Flows:**

* Data flows between processes, data stores, and external entities to facilitate the flow of information within the system.
* Inputs from Users trigger processes, while outputs from processes update data stores and provide feedback to Users.

**ER DIAGRAM**

****

**Fig 4.2.7 ENTITY Diagram for Industrial Automation System**

**Entities:**

Sensor Data: Represents the data collected from various sensors in the industrial system. Attributes may include Sensor\_ID (Primary Key), Sensor\_Type, Reading\_Value, and Timestamp.

Control Signals: Represents the signals sent to control devices in the system. Attributes may include Signal\_ID (Primary Key), Device\_ID, Control\_Type, Control\_Value, and Timestamp.

Monitoring Data: Represents the monitoring information and analysis results. Attributes may include Data\_ID (Primary Key), Device\_ID, Parameter\_Type, Parameter\_Value, and Timestamp.

Users: Represents the users interacting with the system. Attributes may include User\_ID (Primary Key), Username, Password, and Role.

Devices: Represents the IoT devices within the system. Attributes may include Device\_ID (Primary Key), Device\_Type, Location, and Status.

**Relationships:**

* Sensor Data is associated with Devices as it collects data from sensors installed on IoT devices.
* Control Signals are associated with Devices as they control the operation of IoT devices.
* Monitoring Data is associated with Users as they monitor and analyze the data.
* Users are associated with Devices as they interact with the IoT devices in the system.

**Cardinalities:**

* Sensor Data and Devices: Many-to-One (Many sensor data can be collected from one device).
* Control Signals and Devices: Many-to-One (Many control signals can be sent to one device).
* Monitoring Data and Users: Many-to-One (Many monitoring data can be analyzed by one user).
* Users and Devices: Many-to-Many (Many users can interact with many devices).

**Attributes:**

Ensure to include primary keys (PK) for each entity and relevant attributes that describe the characteristics of the entities.

**CHAPTER-5**

**SUMMARY**

**SUMMARY**

The IoT Based Industrial Automation System project aimed to enhance industrial monitoring, control, and efficiency through IoT technology and Raspberry Pi integration. Here is a detailed summary of the project outcomes:

**System Development and Implementation:**

The project successfully developed a comprehensive industrial automation system using Raspberry Pi 3 B+ as the central controlling unit. Various sensors, including the INA219 power measuring unit for voltage and current, DS18B20 digital thermometer for temperature measurement, and a laser module for product counting, were integrated into the system.

The Raspberry Pi camera was utilized for live video monitoring on a web server, providing visual insights into the industrial processes. The Cayenne application was employed for controlling the cooler fan and DC gear motor, enabling remote operation and management of devices.

**Broadband Connection and Performance:**

The industrial automation system leveraged a broadband connection for data transmission and communication between the Raspberry Pi and the cloud server. This connectivity ensured reliable and efficient performance of the system, meeting the project's expectations for seamless operation.

**Sensor Placement and Data Acquisition:**

Proper placement of appliances and sensors within the industrial setup was crucial for accurate data acquisition and monitoring. The project utilized the sensors effectively, with the INA219 measuring unit capturing voltage and current data, the digital thermometer monitoring temperature in Celsius, and the laser module facilitating product counting.

**Achievement of Expected Results:**

The project successfully met its objectives by implementing a functional and efficient industrial automation system. By integrating IoT technology, Raspberry Pi, and sensor modules, the system demonstrated the capability to monitor, control, and optimize industrial processes effectively.

**Remote Monitoring and Control:**

The system's integration with the Cayenne application enabled remote monitoring and control of devices such as the cooler fan and DC gear motor. Users could access the system from anywhere, allowing for real-time adjustments and management of industrial equipment.

**Enhanced Efficiency and Productivity:**

Through continuous monitoring, analysis, and optimization, the industrial automation system contributed to enhanced efficiency and productivity in the industrial environment. Users could make data-driven decisions, identify operational improvements, and streamline processes for better performance.

**EXECUTED CODE:**

**TEMP**

#include <ESP8266WiFi.h>

#include <BlynkSimpleEsp8266.h>

#include <OneWire.h>

#include <DallasTemperature.h>

char auth[] = "YourAuthCode";

char ssid[] = "YourWiFiSSID";

char pass[] = "YourWiFiPassword";

#define ONE\_WIRE\_BUS D4 // Pin for the temperature sensor

OneWire oneWire(ONE\_WIRE\_BUS);

DallasTemperature sensors(&oneWire);

BlynkTimer timer;

void setup() {

  Serial.begin(9600);

  Blynk.begin(auth, ssid, pass);

  sensors.begin();

  timer.setInterval(1000L, sendTemperature); // Send temperature every second

}

void loop() {

  Blynk.run();

  timer.run();

}

void sendTemperature() {

  sensors.requestTemperatures();

  float temperature = sensors.getTempCByIndex(0);

  Blynk.virtualWrite(V1, temperature); /

**LASER SENSOR**

#!/usr/bin/env python

#import device\_patches       # Device specific patches for Jetson Nano (needs to be before importing cv2)

import cv2

import os

import sys, getopt

import signal

import time

from edge\_impulse\_linux.image import ImageImpulseRunner

import pyrebase

import json

import time

Washer\_count = 0

Faulty\_washer\_count = 0

Bolt\_count = 0

Lollipop\_count = 0

start = 0

flag = 0

start\_value = {"Value" : start}

count\_data = {"Bolt" : Bolt\_count,"Washer":Washer\_count,"Faulty\_Washer":Faulty\_washer\_count,"Lollipop":Lollipop\_count,}

runner = None

# if you don't want to see a camera preview, set this to False

show\_camera = True

if (sys.platform == 'linux' and not os.environ.get('DISPLAY')):

    show\_camera = False

def now():

    b = round(time.time() \* 1000)

    print("NOW", b)

    return b

def get\_webcams():

    port\_ids = []

    for port in range(5):

        print("Looking for a camera in port %s:" %port) else

        camera = cv2.VideoCapture(port)

        if camera.isOpened():

            ret = camera.read()[0]

            if ret:

                backendName =camera.getBackendName()

                w = camera.get(3)

                h = camera.get(4)

                print("Camera %s (%s x %s) found in port %s " %(backendName,h,w, port))

                port\_ids.append(port)

            camera.release()

    return port\_ids

def sigint\_handler(sig, frame):

    print('Interrupted')

    if (runner):

        runner.stop()

    sys.exit(0)

signal.signal(signal.SIGINT, sigint\_handler)

def help():

    print('python classify.py <path\_to\_model.eim> <Camera port ID, only required when more than 1 camera is present>')

def main(argv):

    global Washer\_count,Faulty\_washer\_count,Bolt\_count,Lollipop\_count,start,flag

    if flag == 0:

        config = {

        # You can get all these info from the firebase website. It's associated with your account.

        "apiKey": "va6SGLXk01DPOICa8afZkiQh8kWhx3z1usS1s9Qb",

        "authDomain": "counter-a2b53.firebaseapp.com",

        "databaseURL": "https://counter-a2b53-default-rtdb.firebaseio.com/",

        "storageBucket": "projectId.appspot.com"

                }

        firebase = pyrebase.initialize\_app(config)

        db = firebase.database()

        db.child("Count").set(count\_data)

        db.child("Count\_start").set(start\_value)

        flag = 1

    try:

        opts, args = getopt.getopt(argv, "h", ["--help"])

    except getopt.GetoptError:

        help()

        sys.exit(2)

    for opt, arg in opts:

        if opt in ('-h', '--help'):

            help()

            sys.exit()

    if len(args) == 0:

        help()

        sys.exit(2)

    model = args[0]

    dir\_path = os.path.dirname(os.path.realpath(\_\_file\_\_))

    modelfile = os.path.join(dir\_path, model)

    print('MODEL: ' + modelfile)

    with ImageImpulseRunner(modelfile) as runner:

        try:

            model\_info = runner.init()

            print('Loaded runner for "' + model\_info['project']['owner'] + ' / ' + model\_info['project']['name'] + '"')

            labels = model\_info['model\_parameters']['labels']

            if len(args)>= 2:

                videoCaptureDeviceId = int(args[1])

           :

                port\_ids = get\_webcams()

                if len(port\_ids) == 0:

                    raise Exception('Cannot find any webcams')

                if len(args)<= 1 and len(port\_ids)> 1:

                    raise Exception("Multiple cameras found. Add the camera port ID as a second argument to use to this script")

                videoCaptureDeviceId = int(port\_ids[0])

            camera = cv2.VideoCapture(videoCaptureDeviceId)

            ret = camera.read()[0]

            if ret:

                backendName = camera.getBackendName()

                w = camera.get(3)

                h = camera.get(4)

                print("Camera %s (%s x %s) in port %s selected." %(backendName,h,w, videoCaptureDeviceId))

                camera.release()

            else:

                raise Exception("Couldn't initialize selected camera.")

            next\_frame = 0 # limit to ~10 fps here

            print("Setting next\_frame zero")

            for res, img in runner.classifier(videoCaptureDeviceId):

                if (next\_frame > now()):

                    print("NEXT FRAME : ",next\_frame)

                    a = (next\_frame - now()) / 100

                    print("SLEEPING TIME : ",a)

                    time.sleep(a)

                # print('classification runner response', res)

                if "bounding\_boxes" in res["result"].keys():

                    print('Found %d bounding boxes (%d ms.)' % (len(res["result"]["bounding\_boxes"]), res['timing']['dsp'] + res['timing']['classification']))

                    Count = len(res["result"]["bounding\_boxes"])

                    print(Count)

                    Count\_start = db.child("Count\_start").get()

                    print(Count\_start.val())

                    if Count\_start.val()['Value'] == 1:

                      for bb in res["result"]["bounding\_boxes"]:

                        #print('\t%s (%.2f): x=%d y=%d w=%d h=%d' % (bb['label'], bb['value'], bb['x'], bb['y'], bb['width'], bb['height']))

                        img = cv2.rectangle(img, (bb['x'], bb['y']), (bb['x'] + bb['width'], bb['y'] + bb['height']), (255, 0, 0), 1)

                        Label  = bb['label']

                        score  = bb['value']

                        if score > 0.85 :

                          if Label == "Washer":

                             Washer\_count+=1

                          elif Label == "Faulty Washer":

                             Faulty\_washer\_count+=1

                          elif Label == "Bolt":

                             Bolt\_count+=1

                          elif Label == "Lollipop":

                             Lollipop\_count+=1

                    db.child("Count").update({"Bolt" : Bolt\_count,"Washer":Washer\_count,"Faulty\_Washer":Faulty\_washer\_count,"Lollipop":Lollipop\_count,})

                    print("Washer no",Washer\_count)

                    print("Faulty Washer",Faulty\_washer\_count)

                    Washer\_count = 0

                    Faulty\_washer\_count = 0

                    Bolt\_count = 0

                    Lollipop\_count = 0

                if (show\_camera):

                    cv2.imshow('edgeimpulse', cv2.cvtColor(img, cv2.COLOR\_RGB2BGR))

                    if cv2.waitKey(1) == ord('q'):

                        break

                next\_frame = now() + 100

                #print("UPDATED NEXT FRAME",next\_frame)

        finally:

            if (runner):

                runner.stop()

if \_\_name\_\_ == "\_\_main\_\_":

   main(sys.argv[1:])

**COUNTING MOTION.PY**

#!/usr/bin/env python

#import device\_patches       # Device specific patches for Jetson Nano (needs to be before importing cv2)

import cv2

import os

import sys, getopt

import signal

import time

from edge\_impulse\_linux.image import ImageImpulseRunner

import pyrebase

import json

import time

Washer\_count = 0

Faulty\_washer\_count = 0

Bolt\_count = 0

Lollipop\_count = 0

start = 0

flag = 0

start\_value = {"Value" : start}

count\_data = {"Bolt" : Bolt\_count,"Washer":Washer\_count,"Faulty\_Washer":Faulty\_washer\_count,"Lollipop":Lollipop\_count,}

runner = None

# if you don't want to see a camera preview, set this to False

show\_camera = True

if (sys.platform == 'linux' and not os.environ.get('DISPLAY')):

    show\_camera = False

def now():

    b = round(time.time() \* 1000)

    print("NOW", b)

    return b

def get\_webcams():

    port\_ids = []

    for port in range(5):

        print("Looking for a camera in port %s:" %port)

        camera = cv2.VideoCapture(port)

        if camera.isOpened():

            ret = camera.read()[0]

            if ret:

                backendName =camera.getBackendName()

                w = camera.get(3)

                h = camera.get(4)

                print("Camera %s (%s x %s) found in port %s " %(backendName,h,w, port))

                port\_ids.append(port)

            camera.release()

    return port\_ids

def sigint\_handler(sig, frame):

    print('Interrupted')

    if (runner):

        runner.stop()

    sys.exit(0)

signal.signal(signal.SIGINT, sigint\_handler)

def help():

    print('python classify.py <path\_to\_model.eim> <Camera port ID, only required when more than 1 camera is present>')

def main(argv):

    global Washer\_count,Faulty\_washer\_count,Bolt\_count,Lollipop\_count,start,flag

    if flag == 0:

        config = {

        # You can get all these info from the firebase website. It's associated with your account.

        "apiKey": "va6SGLXk01DPOICa8afZkiQh8kWhx3z1usS1s9Qb",

        "authDomain": "counter-a2b53.firebaseapp.com",

        "databaseURL": "https://counter-a2b53-default-rtdb.firebaseio.com/",

        "storageBucket": "projectId.appspot.com"

                }

        firebase = pyrebase.initialize\_app(config)

        db = firebase.database()

        db.child("Count").set(count\_data)

        db.child("Count\_start").set(start\_value)

        flag = 1

    try:

        opts, args = getopt.getopt(argv, "h", ["--help"])

    except getopt.GetoptError:

        help()

        sys.exit(2)

    for opt, arg in opts:

        if opt in ('-h', '--help'):

            help()

            sys.exit()

    if len(args) == 0:

        help()

        sys.exit(2)

    model = args[0]

    dir\_path = os.path.dirname(os.path.realpath(\_\_file\_\_))

    modelfile = os.path.join(dir\_path, model)

    print('MODEL: ' + modelfile)

    with ImageImpulseRunner(modelfile) as runner:

        try:

            model\_info = runner.init()

            print('Loaded runner for "' + model\_info['project']['owner'] + ' / ' + model\_info['project']['name'] + '"')

            labels = model\_info['model\_parameters']['labels']

            if len(args)>= 2:

                videoCaptureDeviceId = int(args[1])

            else:

                port\_ids = get\_webcams()

                if len(port\_ids) == 0:

                    raise Exception('Cannot find any webcams')

                if len(args)<= 1 and len(port\_ids)> 1:

                    raise Exception("Multiple cameras found. Add the camera port ID as a second argument to use to this script")

                videoCaptureDeviceId = int(port\_ids[0])

            camera = cv2.VideoCapture(videoCaptureDeviceId)

            ret = camera.read()[0]

            if ret:

                backendName = camera.getBackendName()

                w = camera.get(3)

                h = camera.get(4)

                print("Camera %s (%s x %s) in port %s selected." %(backendName,h,w, videoCaptureDeviceId))

                camera.release()

            else:

                raise Exception("Couldn't initialize selected camera.")

            next\_frame = 0 # limit to ~10 fps here

            print("Setting next\_frame zero")

            for res, img in runner.classifier(videoCaptureDeviceId):

                if (next\_frame > now()):

                    print("NEXT FRAME : ",next\_frame)

                    a = (next\_frame - now()) / 100

                    print("SLEEPING TIME : ",a)

                    time.sleep(a)

                if "bounding\_boxes" in res["result"].keys():

                    print('Found %d bounding boxes (%d ms.)' % (len(res["result"]["bounding\_boxes"]), res['timing']['dsp'] + res['timing']['classification']))

                    Count = len(res["result"]["bounding\_boxes"])

                    print(Count)

                    Count\_start = db.child("Count\_start").get()

                    print(Count\_start.val())

                    fps = 1

                    if Count\_start.val()['Value'] == 1:

                      for bb in res["result"]["bounding\_boxes"]:

                        #print('\t%s (%.2f): x=%d y=%d w=%d h=%d' % (bb['label'], bb['value'], bb['x'], bb['y'], bb['width'], bb['height']))

                        img = cv2.rectangle(img, (bb['x'], bb['y']), (bb['x'] + bb['width'], bb['y'] + bb['height']), (255, 0, 0), 1)

                        Label  = bb['label']

                        score  = bb['value']

                        if score > 0.85 :

                          if Label == "Washer":

                             Washer\_count+=(1/fps)

                          elif Label == "Faulty Washer":

                             Faulty\_washer\_count+=(1/fps)

                          elif Label == "Bolt":

                             Bolt\_count+=(1/fps)

                          elif Label == "Lollipop":

                             Lollipop\_count+=(1/fps)

                    db.child("Count").update({"Bolt" : Bolt\_count,"Washer":Washer\_count,"Faulty\_Washer":Faulty\_washer\_count,"Lollipop":Lollipop\_count,})

                    print("Washer no",Washer\_count)

                    print("Faulty Washer",Faulty\_washer\_count)

                    '''Washer\_count = 0

                    Faulty\_washer\_count = 0

                    Bolt\_count = 0

                    Lollipop\_count = 0'''

                if (show\_camera):

                    cv2.imshow('edgeimpulse', cv2.cvtColor(img, cv2.COLOR\_RGB2BGR))

                    if cv2.waitKey(1) == ord('q'):

                        break

                next\_frame = now() + 100

                #print("UPDATED NEXT FRAME",next\_frame)

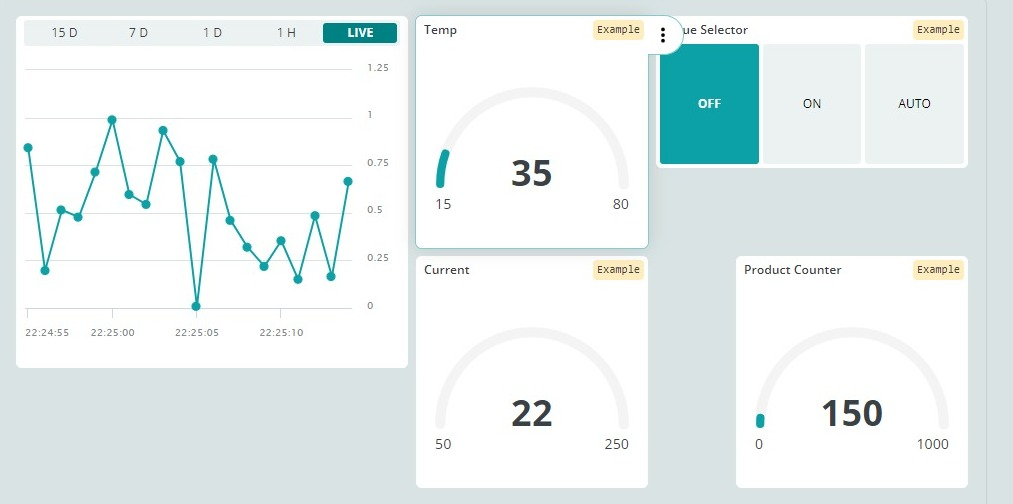
        finally:

            if (runner):

                runner.stop()

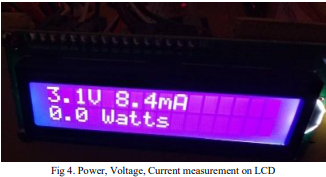
if \_\_name\_\_ == "\_\_main\_\_":

   main(sys.argv[1:])



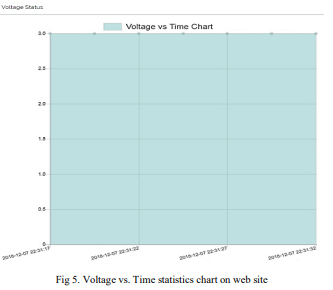
**Fig 5.2.1 Live Monitoring Interface**

**RESULTS**

The entire project outcomes. For getting the desired output at first we have to develop the system shown in the “Fig.4.4.1” in the previous chapter. Industrial automation observation system employs broadband connection, resulting in acceptable performance. Eventually, expected results were met. For establishing a proper system the appliances and the sensors placement is very much necessary. We used INA219 power measuring unit for getting voltage and current, Used Digital thermometer for measuring temperature in Celsius, we used laser module for product counting, we have used raspberry pi camera for monitoring the live video on web server. We used a cayenne app for controlling the cooler fan and also used a dc gear motor on the cayenne app for running them forward and reverse biased.

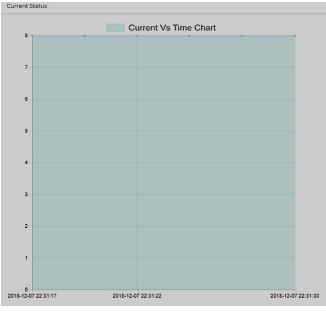
**Fig 5.1.1 Current,Voltage Measurement on LCD**

**Voltage vs. Time**: We used a voltage vs. time graph that has been sent from the pi and updated on the web. On the other hand, times are being updated from the current time zone. Generally we count the GMT+6 time to update our data through the web.



**Fig 5.1.2 Voltage Vs Time Chart**

**Current vs. Time**: We used the current vs. time graph that has been sent from the pi and updated on the web. On the other hand, the values of time are being updated from the current time zone. Generally we count the GMT+6 time to update our data through the web.



**Fig 5.1.3 Current Vs Time Chart**

In “Fig. 4” the value of voltage, current and power is measured on LCD. The next figures are “Fig. 5” and “Fig. 6” which actually deal with the measured value on the LCD showing in the web server. The practical value of the measurement is almost accurate to the value which is obtained from the website.

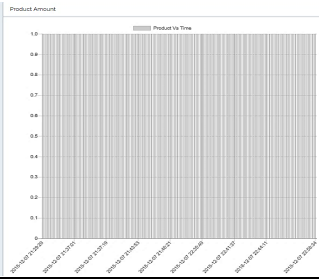


**Fig 5.1.4 Temperature value on LCD**

Product amount counting based on product counter value shown on “Fig. 5.5”. “Fig. 5.6” shows the DC gear motor controlling and cooling fan were controlled through Cayenne app online. Both the observation and controlling are easily done by this software.

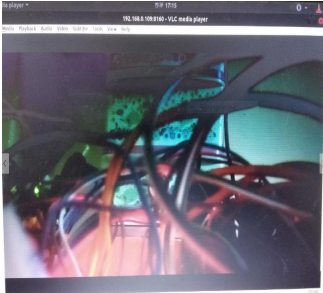


**Fig 5.1.5 Product vs Time Chart**



**Fig 5.1.6 Dc gear motor controlling and cooler fan controlling from Cayenne app online**

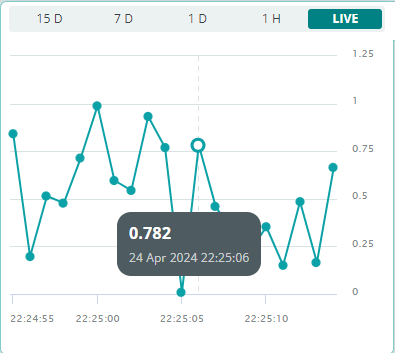
In “Fig. 5.7” the overall system was monitored by a PI camera. The condition of internal wires and their connection can be observed from a distance.



**Fig 5.1.7 Overall system monitoring using PI camera**

**Interface Output**

Fig 5.. represents the current with respective time which helps the user to read the live readings as well as can get the readings for the past 15 days.



**Fig 5.2.2 Interface Output**

Demo Video link : <https://vimeo.com/737906235>

**CONCLUSION**

The IoT Based Industrial Automation System project successfully implemented a comprehensive solution for monitoring, controlling, and optimizing industrial processes using IoT technology and Raspberry Pi integration. The project's conclusion can be summarized as follows:

The utilization of IoT technology in industrial automation proved to be cost-effective and efficient. The system provided a means for remote monitoring, control, and data analysis, contributing to improved operational efficiency and productivity in industrial settings. The system offered convenience and accessibility to users by enabling remote monitoring and control of industrial devices through a web interface. Users could access real-time data, make informed decisions, and manage equipment from anywhere in the world, enhancing operational flexibility.By incorporating features such as live video monitoring, temperature measurement, and product counting, the system enhanced safety measures and provided comprehensive monitoring capabilities. Users could receive alerts for anomalies, track performance metrics, and ensure the smooth operation of industrial processes. The integration of the Cayenne application for device control added a user-friendly interface to the system, simplifying the management of devices such as the cooler fan and DC gear motor. This interface facilitated seamless interaction with the industrial automation system, making it accessible to a wide range of users.

The project demonstrated scalability and adaptability by showcasing how IoT technology can be applied to various industries for remote monitoring and control. The system's design with medium-cost instruments and locally available components made it easily deploy able and adaptable to different industrial environments. The industrial automation system laid the foundation for future innovation and advancements in industrial processes. The integration of IoT technology, cloud connectivity, and Raspberry Pi capabilities opens up possibilities for further enhancements, such as artificial intelligence integration and predictive maintenance.

In conclusion, the IoT Based Industrial Automation System presented a robust and efficient solution for modernizing industrial operations, improving efficiency, and enabling remote monitoring and control. The project's outcomes highlighted the benefits of IoT technology in industrial settings and emphasized the potential for continued innovation and development in the field of industrial automation.

CHAPTER -6

**REFERENCES**

**REFERENCES**

[1] Karim AB, Hassan AZ, Akanda MM, “Monitoring food storage humidity and temperature data using IoT,” *MOJ Food Process* *Technology*, Vol. 6, pp. 400-404, August 2018.

[2] “Applications of IOT,” [Online]. Available at: <https://www.businessinsider.com/internet-of-thingsdevicesapplications-> examples-2016-8. [Accessed: January 27, 2020].

[3] Noel Rohith Mervin, Aravind R, Bestley Joe S., "PLC based automatic grinding system for door frames," *International Journal of* *Engineering & Technology*, Vol. 7, pp. 170-174, May 2018.

[4] Ogawa, M., Tamura, T., Yoda, M., & Togawa, T. (n.d.), “Fully automated biosignal acquisition system for home health monitoring,” Proceedings of the *19th Annual International Conference of the IEEE* *Engineering in Medicine and Biology Society,* Vol. 6, pp. 2403-2405, November 1997.

[5] Nagendra Reddy, P. S., Kumar Reddy, K. T., Kumar Reddy, P. A., Kodanda Ramaiah, G. N., & Kishor, S. N., “An IoT based home automation using android application,” *2016 International Conference* *on Signal Processing, Communication, Power and Embedded System* *(SCOPES),* pp. 285-290, October 2016.

[6] Piyare, Rajeev & Tazil, M., “Bluetooth based home automation system using cell phone,” *IEEE Transactions on Consumer Electronics*, Vol. 15, pp. 192-195, June 2011.

[7] Govindraj, Vignesh & Sathiyanarayanan, Mithileysh & Abubakar, Babangida, “Customary homes to smart homes using Internet of Things (IoT) and mobile application,” *2017 International Conference On* *Smart Technologies For Smart Nation,* pp. 1059-1063, August 2017.

[8] Mahmud Rana, G. M. S., Mamun Khan, A. A., Hoque, M. N., & Mitul,A. F., “Design and implementation of a GSM based remote home security and appliance control system,” *2013 2nd International* *Conference on Advances in Electrical Engineering (ICAEE),* pp. 291-

295, December 2013.

[9] Teymourzadeh, R., Salah Addin Ahmed, Kok Wai Chan, & Mok Vee Hoong, “Smart GSM based Home Automation System,” *2013 IEEE* *Conference on Systems, Process & Control (ICSPC),* pp. 306-309, December 2013.

[10] S. A. I. Quadri and P. Sathish, “IoT based home automation and surveillance system," *2017 International Conference on Intelligent* *Computing and Control Systems (ICICCS),* pp.861-866, June 2017.

[11] Gunasekaran, Mahalalakshmi, “IOT Based Home Automation Using Arduino,” *International Journal of Engineering and Advanced* *Research Technology,* Vol. 3, ISSN: 2454-9290, August 2017.

[12] “Raspberry Pi As a 3g (Huawei E303) Wireless (Edimax EW-7811Un) Router,” [Online]. Available at: https://www.instructables.com/id/Raspberry-Pi-as-a-3g-Huawei-E303-wireless-Edima/` [Accssed: February 11, 2020].

**APPENDIX A  
BASE RESEARCH PAPER**