IOT MANAGER - DEVICE MANAGEMENT

A PROJECT REPORT

Submitted by

AJIL K (TKM21MCA-2004)

to

The APJ Abdul Kalam Technological University

In partial fulfillment of the requirements for the award of the degree of

MASTER OF COMPUTER APPLICATION

Thangal Kunju Musaliar College of Engineering
Kerala

DEPARTMENT OF COMPUTER APPLICATIONS

MAY 2023

DECLARATION

I undersigned hereby declare that the project report on **IOT MANAGER - DEVICE MANAGEMENT**, submitted for partial fulfillment of the requirements for the award of degree of Master of Computer Application of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by me under supervision of **Prof. Vaheetha Salam**. This submission represents my ideas in my own words and where ideas or words of others have been included,I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in our submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not previously served as the basis for the award of any degree, diploma, or similar title by any other University.

Kollam

19-05-2023

AJIL K

DEPT. OF COMPUTER APPLICATIONS

TKM COLLEGE OF ENGINEERING

KOLLAM

2021 - 23

CERTIFICATE

This is to certify that the report entitled **IOT MANAGER - DEVICE MANAGEMENT** submitted by **AJIL K** (TKM21MCA-2004) to the APJ Abdul Kalam Technological University in partial fulfillment of the Masters degree in Computer Application is a bonafide record of the project work carried out by him under our guidance and supervision. This report, in any form, has not been submitted to any other University or Institute for any reason.

Internal Supervisor Head of the Department External Examiner

Knowledge Lens Pvt. Ltd
74/A, Keonics, Hosur road,
Electronic City Phase 1 - Bengaluru - 560100
CIN No: U72200KA2013PTC072670

Date:17-05-2023

TO WHOM IT MAY CONCERN

This is to certify that Mr. Ajil K from TKM College of Engineering, Kollam has completed the internship on **"IoT Manager"** project at **Knowledge Lens Pvt Ltd** from 16th January 2023 to 16th May 2023. During the period of his training with us he was found punctual, hardworking, and inquisitive.

We wish him all the success in future endeavors.

Yours Truly,

For **Knowledge Lens Pvt Ltd**.

Jayashree S
Sr. HR Manager
hr@knowledgelens.com

ACKNOWLEDGEMENT

First and foremost, I thank GOD almighty and my parents for the success of this project. I owe sincere gratitude and heart full thanks to everyone who shared their precious time and knowledge for the successful completion of my project.

I am extremely grateful to **Dr. Fousia M Shamsudeen**, Head of the Department, Department of Computer Applications, for providing me with best facilities.

I would like to thank my project guide **Prof. Vaheetha Salam**, Department of Computer Applications, who motivated me throughout the project.

With a profound sense of gratitude, I would like to express my heartfelt thanks to my advisor and coordinator **Prof. Natheera Beevi M**, Department of Computer Applications, for her expert guidance, co-operation, and immense encouragement.

I would like to thank my external coordinator **Mr. Athreya K S** and **Mr. Irfanuddin Shafi Ahmed**, Knowledge Lens, who guided me throughout my work.

I profusely thank all other faculty members in the department and all other members of TKM College of Engineering, for their guidance and inspirations throughout my course of study.

I owe my thanks to my friends and all others who have directly or indirectly helped me in the successful completion of this project.

AJIL K

ABSTRACT

IOT MANAGER - DEVICE MANAGEMENT, is a web application designed to provide a comprehensive solution for managing remote edge and IoT devices. It enables users to monitor, control, and configure devices from a centralized platform, making it easier to manage and maintain large networks of devices.

The application provides a user-friendly interface that allows users to quickly and easily access device data and perform various operations. Users can view device status, track device performance, and set alerts for key metrics. The application also allows users to remotely control devices, enabling them to configure settings, update firmware, and troubleshoot issues.

One of the key features of this application is its ability to handle large volumes of data from a variety of devices. It uses advanced analytics and machine learning algorithms to analyze data and generate insights that can be used to optimize device performance and improve operational efficiency.

Overall, this web application provides a comprehensive solution for managing remote edge and IoT devices, helping users to reduce downtime, improve device performance, and streamline operations.

Contents

List of Figures

Chapter 1

Introduction

IOT MANAGER - DEVICE MANAGEMENT is a portal that provides a complete overview of all devices that are deployed on-site under one roof. The portal contains features like remote monitoring of devices, real-time alerts, and management of devices like OTA, Logs, etc

The web application for managing remote edge and IoT devices are applicable to any industry that uses IoT devices to monitor and control their operations. It can be used by businesses of all sizes, from small startups to large enterprises, to manage and maintain their IoT devices. The application is particularly useful for industries where large networks of devices need to be monitored and maintained, such as manufacturing plants, logistics and transportation companies, and healthcare facilities.

For the application to operate the manufacturing plant successfully, it needs to contain a number of essential features. It should, first and foremost, offer a user management system that enables various user types, such as super admin, support admin, support user, site user, vendor, and customer, to log in with their specific roles and access levels. Second, it must enable users to install, remove, configure, and monitor edge and IoT devices inside the manufacturing facility. Thirdly, the application needs to include data analysis tools, such as charts, graphs, and dashboards, to assist users in examining the data produced by the devices. Fourthly, it must have a system for alerting users to any irregularities or problems with the products or the manufacturing process. Together, all of these features would make it possible to operate the production facility effectively, enabling prompt problem-solving and issue-identification.

1.1 Company Profile

Knowledge Lens's Lens product line automates and simplifies the finding of hidden insights in Big Data. The goal is to turn unstructured data into useful business insights. We are big data technology nerds with extensive experience in big data projects such as big data engineering, data science, and industry knowledge.

1.1.1 Products

- **iLens (Intelligent Lens)**

  iLens provides a centralised platform for the intelligent integration of multiple devices or sensors in major corporations, the manufacturing sector, the home, commercial properties, and so on. iLens provides a MQTT interface for the seamless integration of numerous field sensor devices in order to record time series data in real time. iLens offers the ability to generate alerts and alarms based on pre-configured rules.

- **MLens**

  MLens is a one-step solution for managing disaster recovery for big data and platforms. MLens features include:

  1. Migration of Large Data Backups
  2. Disaster Recovery that is Automated
  3. Data Encryption, Compression, and Archival
  4. High-Speed Batch Data Ingestion
  5. Schedules Monitoring
  6. Controls for Secure Access

- **AiLens**

  Next-generation AI platform featuring a collaborative workspace, experiment designer, engineering workbench for modelling features, and enterprise security and DevOps integrations with an AI/ML asset store. AiLens is a smart AI assistant with a unified graphical user interface for building Data Engineering and AI/ML pipelines. The unified AI orchestrator provided by AiLens allows customers to launch model executions on any runtime, including Tensorflow, PyTorch, SparkML, Theano, H2O, MxNet, and

Azure / AWS, from a single console. AiLens is incredibly versatile because every user will have the same experience. Because of the platform's meta model-driven design, independent of future technological improvements. Our software distinguishes itself from the competition by virtue of its user-friendly job submission and monitoring structure, safe integration with other external entities, built-in encryptions, and role-based access control have support. The following are the key features:

1. Any AI Algorithm, Any AI Stack, Anywhere
2. Unified Artificial Intelligence based Orchestrator
3. Simplified User Interface (UI) and User Experience (UX)
4. AI Intelligent Assistant
5. AI Modelling Environment with Integrated Data Preparation
6. Seamless Enterprise of Security Integration

- **GLens**

  GLens is a real-time acquisition, monitoring, and analysis system for industrial emissions, effluent discharges, and ambient air monitoring. The GLens DAS Software, GLens Server Platform, and GLens Environ Data Logger provide a one-stop shop for all industrial environmental requirements. Using a plug-and-play model, any analyzer, sensor, or device can be connected to the platform and real-time data collected.

  The key features of GLens are:

  1. Rest-based open protocol for deployment of multiple clients.
  2. Real-time notifications and alarms that integrate SMS and email.
  3. Analyzers can be remotely calibrated and configured.
  4. Complete protocol plug-and-play connection with any analyzer brand or model. - Data quality codes and integrated in accordance with ISO 7168.
  5. Predictive analytics and integrated analytics for efficient pollution control.
  6. Live industry dashboards that are consolidated.

1.1.2 Services

- **Big Data Engineering**

  We provides complete Big Data Protects architecture, design, development, testing, and deployment services.

- **Big Data Security**

  We are a specialised consulting organisation that provides Big Data Services.

- **Big Data Analytical Services**

  We reveal hidden insights from a variety of data sources using our pre-built analytical Lens.

- **Development of Big Data Competencies**

  Despite a lack of unique Big Data knowledge, we deliver one of the top Big Data Competency Development programmes for the company.

1.2 Existing System

This web applicatio n developed 6 years ago with older technology, aimed to provide a solution for managing remote edge and IoT devices. The application allowed users to monitor, control, and configure devices from a centralized platform, making it easier to manage large networks of devices. However, as technology has evolved over the years, the limitations of the older technology used to develop the application have become apparent. The application may not be scalable enough to handle the growing volume of data generated by IoT devices, and may not be able to support the latest security protocols. Furthermore, the user interface may not be as intuitive and user-friendly as modern applications, making it difficult for users to access and analyze data from their devices. The application may also lack some of the advanced analytics and machine learning capabilities that are now available in modern applications. In summary, while the older version of this web application provided a useful solution for managing remote edge and IoT devices, its limitations in terms of scalability, security, and user experience may hinder its effectiveness in today's technology landscape. Upgrading the application with modern technology and features may be necessary to keep up with the evolving needs of IoT device management.

1.3 Proposed System

All of the problems that the current system was having will be fixed by the proposed system. The problems that the current system is having are listed below.

The issue of managing and maintaining vast networks of devices in a manufacturing context is addressed by the existing system for monitoring, configuring, and controlling edge and IoT devices placed in industries for manufacturing analytics. Manufacturing plants typically use a wide range of edge and IoT devices to monitor and control various aspects of their operations, such as production lines, quality control systems, and environmental sensors. These devices generate vast amounts of data, which need to be processed and analyzed in real-time to gain insights into product performance and identify potential issues. The challenge is that these devices are often dispersed across a plant, making it difficult to monitor and control them effectively from a centralized location. Without a comprehensive solution to manage these devices, it can be challenging to ensure that they are operating efficiently and effectively, leading to increased downtime and reduced operational efficiency. Security is also a significant concern in manufacturing environments, as cyber threats can compromise the integrity of the production process, leading to lost revenue and reputational damage.

In the proposed system, the device's meta data and parameters data will be kept in the MongoDB (Meta DB).The RedisDB will keep track of the device's Heart-Beat last sync time. For forthcoming versions of the programme, audit logs will be kept in PostgreSQL and all timeseries data from the devices will be kept in TimeScaleDB.

Overall, The new web application will monitor, configure, and control edge and IoT devices installed in industries for manufacturing analytics and will handle every issue that the current system encountered. It will also address the need for a comprehensive solution that streamlines device management, improves security, and offers real-time analytics to optimise production performance and increase operational efficiency in a manufacturing environment.

1.4 Objectives

The goal is to accomplish the following:

- **Streamline device management**: The application aims to simplify the management of remote edge and IoT devices by providing a user-friendly interface that allows users to monitor device status, track performance, and configure settings.
- **Improve operational efficiency**: By providing real-time insights into device performance, the application aims to help users optimize their operations and reduce downtime.
- **Enhance security:** The application aims to enhance the security of IoT devices by providing advanced security features such as secure data transmission and access control.
- **Enable scalability:** The application aims to support the scalability needs of IoT device management by providing features that can handle large volumes of data and support the integration of new devices.
- **Provide advanced analytics:** The application aims to provide advanced analytics to generate insights that can be used to optimize device performance and improve operational efficiency.

Chapter 2

Literature Survey

A literature review is a A detailed assessment of the pertinent literature on a subject. Research questions are developed through a literature review, and then solutions are found by looking for and examining relevant material. An advantage of literature reviews is that they frequently result in new insights after re-analyzing the study's findings. A literature review summarises and explains the entire and most recent body of information on a topic as it is given in scholarly books and journal articles. You can be required to write one of two kinds of literary reviews in college: one is an independent assignment for a course. The second type of writing is an introduction or a prelude to a longer article, such a thesis or research report. The focus, angle, and development of a distinct hypothesis or thesis argument will depend on the type of review you are writing. You can learn about the differences between these two categories by reading published literature reviews or the introductory chapters of pertinent theses and dissertations. Consider about the way they tackle the topics and how their arguments are structured.

2.1 Purpose of the Literature Review

1. By highlighting relevant, noticeable, significant, and legitimate high-quality papers or studies, it makes studies on a certain subject understandable to readers.

2. By asking students to summarise, assess, and compare their original work in that subject, it provides young researchers in new areas with a great starting point.

3. It makes sure that researchers don't submit previously published material again.

4. It might offer indications about the course of future studies or propose topics to focus on.

5. It highlights the key findings.

6. It points out gaps, discrepancies, inconsistencies, and contradictions in the literature.

7. It offers a beneficial evaluation of the methods and strategies used by other studies.

2.2 Related Works

2.2.1 Internet of Things

Design and Implementation of a System for Remote IoT Device Management is a system for remote management of IoT devices. In this paper, the authors begin by introducing the Internet of Things and outlining the difficulties involved in remotely administering a significant number of IoT devices.The Remote IoT device management system's design and implementation are then covered in the article. The system is made up of an IoT device-based lightweight client application and a centralised management platform. The system's architecture and the technologies utilised to implement it are described by the authors. The system's capabilities to upgrade firmware and software, modify device settings, and remotely monitor and operate IoT devices are highlighted in the study. In order to ensure secure connection between the IoT devices and the management platform, the authors also go through the security measures put in place in the system. The findings of the system's performance evaluation are presented as the paper comes to a close. The system's throughput, scalability, and latency are all evaluated by the writers. The evaluation's findings demonstrate that the system can efficiently and quickly manage a large number of IoT devices.Overall, the paper makes a significant contribution to the subject of managing IoT devices. It shows a system that addresses the difficulties involved in remotely administering a significant number of IoT devices and offers a thorough overview of its implementation and its design. For researchers and professionals interested in creating or implementing IoT device management systems, the paper is a valuable resource. The study would have been more effective, though, if it had gone into greater depth on the precise use cases and real-world uses of the technology. [1]

The focus of the literature study is on investigating an IoT device management framework made especially for smart home situations. The goal of the article is to provide readers a thorough overview of the current state of knowledge and technological developments in IoT device management and how they apply to smart homes. The report begins by highlighting

the rise in smart home adoption and the crucial contribution of IoT devices to the development of a connected and automated living environment. It emphasises the demand for a reliable and secure infrastructure to control the substantial IoT deployment in smart homes. The literature analysis highlights a number of crucial facets of managing IoT devices in smart homes. First, it looks at methods for finding and setting IoT devices in a smart home network, known as device discovery and provisioning. The benefits and drawbacks of various strategies, including manual configuration, automated discovery, and zero-configuration protocols, are examined.In order to provide safe communication between devices in a smart home context, the evaluation also examines device authentication and access control systems. It explores several authentication strategies, highlighting the benefits and drawbacks of password-based authentication, biometrics, and public key infrastructure (PKI). The study also examines the value of device diagnostics and monitoring in preserving the functionality and health of IoT devices. It looks at methods for gathering sensor data, keeping track of device condition, and spotting errors or abnormalities. The evaluation also looks at failure detection and recovery techniques to ensure that IoT devices in smart homes run continuouslyThe difficulty of interoperability among various IoT devices and platforms is also covered in the literature study. It looks at current standards and protocols like Zigbee, Z-Wave, and MQTT that allow for easy integration and communication amongst gadgets from various manufacturers and technologies. The report also emphasises the need of data management and analytics in situations involving smart homes. It looks at methods for gathering, processing, and analysing the massive amounts of data produced by IoT devices in order to gain knowledge and make wise decisions that will improve user experiences. Finally, the paper addresses the current state of IoT device management for smart homes as well as potential future possibilities. It highlights opportunities for development, like improved security measures, energy efficiency optimisation, and intelligent automation, which may help smart home technologies evolve and be more widely used. The literature analysis offers a thorough overview of IoT device management in the context of smart homes, in conclusion. There are several topics covered, such as device detection, authentication, monitoring, interoperability, data management, and future directions. For academics, practitioners, and developers looking to create and put into practise efficient IoT device management frameworks for smart home environments, this article is an invaluable resource. [11]

2.2.2 MongoDB

Thorough analysis of MongoDB databases, a well-liked NoSQL Database Management System (NoSQL DBMS). Prior to explaining how MongoDB falls within the category of NoSQL databases, the author gives an overview of this category. The paper then discusses several MongoDB topics, such as data modelling, indexing, and querying. In contrast to conventional relational databases, MongoDB is a document-based database, allowing more flexible data modelling, as the paper mentions. The author outlines the benefits of this strategy, including its capacity to scale horizontally and its ability to hold complicated data structures. The performance benefits of indexing in MongoDB are also covered in the study. The author describes the various index types that MongoDB offers and offers advice on when you should use each kind. In the paper's conclusion, the significance of query optimisation in MongoDB is covered. The author outlines several methods for query optimisation, including utilising the aggregation framework in MongoDB and analysing query execution plans with the explain() method. Overall, the paper offers a useful overview to MongoDB and all of its different facets. It is a helpful tool for database managers and developers who are thinking about using MongoDB as a DBMS or who want to learn more about it. Anyway, the paper would have proven more effective if it had offered more specific illustrations and helpful advice on how to put the various strategies discussed into practise. [2]

The following is a survey of the literature for the article "A comparative study: MongoDB vs. MySQL" by C Gyrödi, R Gyrödi, and G Pecherle. The paper compares two well-known database management systems, MongoDB and MySQL. The authors begin by introducing both systems and their key characteristics. The comparative study's methodology is then presented in the publication. The performance features of MongoDB and MySQL were compared in a series of studies by the authors, including testing on data insertion, data retrieval, and query performance. The comparison study's findings are presented in the article, with an emphasis on the benefits and drawbacks of each method. The authors discovered that while MySQL was better at data retrieval, MongoDB was faster at data insertion and query performance. The scalability and fault tolerance of both systems were also examined by the authors. In its last section, the paper offers suggestions for programmers who are deciding between MongoDB and MySQL for their applications. Before making a choice, the authors advise developers to take into account their unique requirements as well as the advantages and disadvantages of each system. Overall, the article offers a useful comparison between MongoDB versus MySQL. It

offers information on the strengths and drawbacks of the systems' performance characteristics. For programmers deciding between MongoDB and MySQL for their projects, the paper is a valuable resource. The study, however, may have been more effective if it had included more useful advice on how to select between the two systems and enhance their effectiveness. [6]

2.2.3 Python

The popularity and expansion of the Python programming language in the last few years are discussed in the article. Python is described as the programming language with the quickest rate of growth by the author, who backs up his assertion with statistics and data. The popularity of Python and its attributes of simplicity, usability, and versatility are also discussed in the article. Python's development from its inception in the 1990s to its current position as one of the most commonly used programming languages are briefly covered by the author. Additionally, the paper discusses Python's many uses in fields like data analysis, web development, machine learning and artificial intelligence. The syntax, performance, and usability of Python are compared to those of other well-known programming languages, like Java and C++. The author also emphasises Python's benefits for beginners, including its accessible tools and easy learning curve.Overall, the paper offers a thorough summary of the development and acceptance of the Python programming language. The assertion that Python is the programming language with the quickest growth rate is supported by the author's analytical analysis and facts. The paper can have been made better, though, by going into the drawbacks and restrictions of using Python and giving additional instances of its applicability in different fields. [7]

The following is a review of the literature for the article "Python: The Programming Language of the Future" by Dr. Sunil Gupta, Deepak Sharma, Firoj Khan, and Akshansh Sharma: The article gives a brief introduction to Python, a well-liked programming language that is on the rise in the software development sector. The authors begin by describing the background and development of Python, as well as its design philosophy and salient characteristics. The readability, versatility, and simplicity of Python for software development are highlighted in the study. Additionally, the authors cover the numerous uses of Python in several disciplines, such as web development, machine learning, and data science. The paper continues by giving a thorough explanation of Python's syntax and foundational programming ideas, making it a valuable tool for those just learning the language. The writers also offer instructions on how to set up and install Python on various systems. Python's potential to

overtake other programming languages in the software development sector is covered in the paper's conclusion. The authors emphasise the increasing need for Python programmers and offer advice for programmers who want to become more proficient in Python. The paper offers a helpful introduction to Python and its main features overall. It is a helpful tool for programmers who are curious about Python's possible uses and benefits and are new to the language. The paper, however, could have been stronger if it had offered more specific illustrations of Python's uses in various industries as well as helpful advice on how to apply Python in particular situations. [3]

2.2.4 AngularJS

The study on the performance of AngularJS, a well-liked JavaScript framework for creating online apps, is presented in this paper. The writers begin by giving a brief overview of AngularJS's salient features. The survey study's methodology is then presented in the publication. Data from more than 1,000 developers with AngularJS experience was gathered by the authors. The poll asked questions about AngularJS's performance characteristics, developers' performance issues, and methods for improving the performance of AngularJS apps. The survey study's findings are highlighted in the article, along with the methods used to improve the AngularJS application's performance and the performance issues that developers most frequently face. The startup time, memory use, and runtime performance of AngularJS are all examined in-depth by the authors. The document offers advice for programmers who are creating AngularJS apps and are worried about performance in the conclusion. The authors advise optimising the AngularJS application source code and using tools like Chrome DevTools to identify performance issues. [5]

The main topic of the literature review study "Interpretation and Analysis of Angular Framework" is AngularJS, a well-liked JavaScript-based framework for creating online applications. The goal of the article is to offer a thorough knowledge of AngularJS by reviewing and analysing the literature that has already been written on the subject. The study starts with an overview of AngularJS, emphasising its main benefits and features. It talks about how Model-View-Controller (MVC) architecture and a declarative approach to developing dynamic and interactive web applications introduced by AngularJS revolutionised web development. The report then conducts a thorough analysis of scholarly works, academic journals, and technical documentation pertaining to AngularJS. The framework's fundamental ideas, architecture,

data binding techniques, dependency injection, and directives are only a few of the topics covered. The evaluation looks into the advantages and difficulties of using AngularJS for development, as well as performance issues and recommended practises. The literature study also emphasises case studies and real-world applications that show how AngularJS may be used effectively in various scenarios. It describes how AngularJS has been used to create mobile apps, enterprise-scale online solutions, and single-page applications (SPAs). The integration of AngularJS with other frameworks and technologies, such RESTful APIs and server-side technologies, is also covered in the study. The evolution of AngularJS and its transfer to subsequent versions, notably Angular (also known as Angular 2+), are highlighted throughout the examination. It sheds light on the architectural improvements, improved functionality, and migration concerns that distinguish AngularJS from Angular, among other distinctions. The literature study culminates with a summary of the important results from the reviewed literature and a discussion on AngularJS's future possibilities. It emphasises how, despite the development of other frameworks, AngularJS is still useful in some situations. Future study is also suggested by the report in areas including performance optimisation methods, security issues, and AngularJS development best practises. In conclusion, this literature study is a useful tool for practitioners, academics, and developers looking to gain a thorough grasp of AngularJS. It provides an assessment and analysis of the framework based on the body of material already in existence, illuminating its fundamental ideas, real-world applications, and prospective directions. [14]

2.2.5 PostgreSQL

The following is a summary of the research that went into the study "MongoDB Vs. PostgreSQL: A Comparative Study on Performance Aspects" written by Antonios Makris, Konstantinos Tserpes, Giannis Spiliopoulos, Dimitrios Zissis, and Dimosthenis Anagnostopoulos: In this study, two well-known database management systems—MongoDB and PostgreSQL—have their performance characteristics compared. The writers begin by introducing NoSQL and SQL databases and outlining how they differ from one another. They continue by contrasting data modelling, indexing, and querying between MongoDB and PostgreSQL. The study emphasises that PostgreSQL is a relational SQL database, whereas MongoDB is a document-based NoSQL database. The authors outline the benefits and drawbacks of each strategy and offer suggestions for when to utilise each database management system.

The report also offers an analysis of MongoDB and PostgreSQL's performance using various benchmarking tools. The read/write operations, parallelism, and scalability of the databases are all evaluated by the writers. The evaluation's findings reveal that MongoDB often outperforms PostgreSQL, especially in read-intensive applications. The ramifications of the study for database administrators and developers are covered in the paper's conclusion. The authors stress the significance of taking performance factors into account when selecting a database management system and offer guidance on when to use MongoDB or PostgreSQL based on the particular requirements of an application. Overall, the paper offers a useful comparison of PostgreSQL vs MongoDB's performance characteristics. It is a helpful tool for database managers and developers who are thinking about using MongoDB or PostgreSQL as a database management system. The study would have been more effective, though, if it had included more information on the precise situations in which MongoDB beats PostgreSQL as well as step-by-step instructions for putting the suggested database management system into practise.[8]

The PostgreSQL database management system (DBMS) is optimised for real-time Internet of Things (IoT) edge applications, according to a review of the literature titled "Optimisations of Database Management Systems for Real Time IoT Edge Applications". The article covers PostgreSQL's many performance-improving methods as they relate to the difficulties and expectations given by IoT edge situations. The article starts out by explaining the idea of IoT edge applications and emphasising the special properties and limitations of these systems, such as constrained computing resources, high data velocity, and low latency requirements. In order to support real-time decision-making and responsiveness, it emphasises the significance of effective data management and processing at the edge. The study then explores the precise PostgreSQL optimisations used to solve the aforementioned issues. It goes through numerous methods the DBMS uses to boost the efficiency of IoT edge applications, including indexing, caching, and query optimisation. The authors emphasise how these optimisations are specifically designed to manage the demands of IoT workloads, such as the requirement for low-latency queries and rapid data updates. The research also investigates the use of data representation methods and compression algorithms to lessen the storage and transmission overhead in IoT edge contexts. It explains how PostgreSQL makes use of these strategies to maximise storage efficiency and reduce network bandwidth usage. The literature study also examines PostgreSQL's integration with other technologies that are often used in IoT edge applications, such as distributed computing platforms and stream processing

frameworks. It looks at how these linkages improve the system's capacity for scalability, fault tolerance, and real-time processing. The study finishes by underlining the significance of database management system optimisations for effective IoT edge application performance and summarising the major findings. It highlights PostgreSQL's contribution to satisfying the particular requirements of real-time IoT edge settings and makes recommendations for further study in this field. In conclusion, the literature review study investigates the PostgreSQL optimisations used for real-time IoT edge applications. It covers a variety of methods, including indexing, caching, query optimisation, compression, and technology integration. The importance of these optimisations in raising the effectiveness and performance of IoT edge systems is emphasised in the article. [15]

2.2.6 FastAPI

The Introduction to Fast API Paper offers a thorough introduction to FastAPI, a cutting-edge web framework for creating Python APIs. The article begins by outlining the significance of APIs in the current software development environment before outlining the major characteristics of FastAPI that make it a desirable option for API development. The report emphasises that FastAPI uses common Python type hints and is built on Python 3.7+ to enable quick and effective code execution. Developers that wish to save time and effort on API documentation can greatly benefit from the framework's automated API documentation creation feature. The study also points out that FastAPI is a great option for developing high-traffic and real-time applications because it is intended to be quick and performant. The advantages of FastAPI's performance are emphasised when they are contrasted with those of other well-known Python web frameworks like Flask and Django. The report closes by stating that because of FastAPI's ease of use, speed, and automatic API documentation production, it has become popular among developers. It is regarded as the best option for creating cutting-edge, effective APIs that can satisfy the requirements of the current software development environment. Overall, the Introduction to Fast API Paper offers a helpful summary of FastAPI and its distinguishing characteristics. For developers who want to learn more about FastAPI and its advantages for API development, it is a useful resource. [4]

The subject of the research review of "AI-based botnet attack classification and detection in IoT devices" is the usage of the FastAPI framework for creating a classification and detection system to thwart botnet assaults in Internet of Things (IoT) devices. The article opens with a

discussion of the rising threat of botnet assaults aimed against Internet of Things (IoT) devices, which are susceptible due to their interconnection and lack of effective security measures and are growing in popularity. Attackers can utilise botnets, which are networks of hacked devices under their control, to carry out a variety of nefarious tasks including distributed denial of service (DDoS) assaults or data theft.The authors suggest using AI to categorise and identify botnet assaults on IoT devices. They decide to use FastAPI, a cutting-edge and effective Python web framework that enables speedy creation of reliable and scalable applications, to build their system. FastAPI is renowned for its high performance and asynchronous features, which make it appropriate for real-time analysis and reaction in the context of botnet detection. The essential ideas and methods employed in the suggested system are highlighted in the literature review. It examines several deep learning models such convolutional neural networks (CNN), support vector machines (SVM), random forests, and machine learning methods that are frequently used for attack categorization. For accurate assault detection, the authors stress the need of feature extraction and selection. The report also examines several data sources and attributes that are utilised to develop AI models. It includes system logs, network traffic analysis, and device behaviour monitoring as potential classification system inputs. The authors talk about the difficulties in managing massive amounts of data produced by IoT devices and the requirement for effective data processing methods. The literature study also explores the body of work on botnet identification in IoT devices and identifies the drawbacks and flaws of earlier methods. It goes over the benefits of utilising FastAPI for creating the suggested system, including how simple it is to use, how fast it is, and how well it integrates with other Python tools. The literature study paper concludes by providing an overview of the FastAPI-based AI-based botnet assault categorization and detection system. In addition to outlining the benefits of using FastAPI as a framework for creating such a system, it offers a thorough study of the methods, algorithms, and data sources employed in the system. The report sets the groundwork for future study and research in the area of botnet detection and IoT security. [13]

2.2.7 TimescaleDB

An analysis of the literature for ”Meteorological Sensor Data Storage Mechanism Based on TimescaleDB and Kafka” by Liqun Shen, Yuansheng Lou, Yong Chen, Ming Lu, and Feng Y is provided below. The study describes a TimescaleDB and Kafka-based data storage system for meteorological sensor data. The authors describe how the growing volume of data produced by meteorological sensors necessitates the development of a high-performance and scalable data storage system. Additionally, they go through the disadvantages of utilising standard databases for processing time-series data as well as the benefits of TimescaleDB. The three levels of the suggested architecture—the data gathering layer, the data processing layer, and the data storage layer—are thoroughly explained in the study. The authors describe how TimescaleDB is used to store and query the data, and how Kafka is utilised for real-time data streaming. They also talk about how to visualise data using Grafana. The performance and scalability of the suggested system are illustrated by experimental findings in the study. The authors demonstrate that TimescaleDB outperforms conventional databases when comparing the proposed system to them in terms of query performance and data compression. They also demonstrate how the suggested system can process a lot of data quickly and with little lag. Overall, the study describes a data storage system for meteorological sensor data that is well-designed and well-implemented. The usage of TimescaleDB and Kafka is justified, and the experimental findings show how well the suggested system works. The report may have been made better, though, by outlining the drawbacks and difficulties of the suggested system and giving additional information about the data processing layer.[10]

2.2.8 RedisDB

Here is a summary of the research for Andres Sacco’s chapter ”Redis: Key/Value Database”.An introduction to Redis, a well-known in-memory key/value store database, is given in this chapter. The benefits of utilising Redis, such as its fast speed, low latency, and support for a variety of data formats, are covered in the author’s first paragraph. The chapter then goes on to discuss Redis’ main attributes, including transaction support, pipelining, and Lua scripting. Additionally, the author describes the many data structures that Redis supports, such as texts, lists, sets, hashes, and sorted sets. Redis is used throughout the chapter’s practical examples for a variety of use cases, including caching, session management, and message

queueing. Redis may be combined with other technologies like Node.js, Python, and Java, as the author illustrates. Redis's many deployment configurations, including standalone, master-slave replication, and cluster mode, are also covered in this chapter. The author gives advice on how to pick the best deployment strategy for your application by outlining the benefits and drawbacks of each option. The chapter offers a thorough introduction to Redis as a key/value store database. It goes through the data structures, use cases, functionality, and deployment possibilities. Developers that are thinking about adopting Redis for their applications might benefit from the chapter. The chapter, however, may have been more effective if it had included more in-depth technical information on how to improve Redis speed and address frequent problems. [9]

The research paper "A Study on Time-Series DBMS Application for EdgeX-based Lightweight Edge Gateway Using RedisDB" examines how RedisDB, a well-known in-memory database system, may be used for time-series data management in the context of EdgeX-based lightweight edge gateways. The notion of edge computing is introduced at the outset of the article, along with its importance in enabling in-the-moment data processing and analysis at the network edge. It highlights how important it is to have effective data management and storage options in edge contexts, especially for time-series data produced by various IoT devices and sensors. The article then gives a brief description of EdgeX, an open-source platform that makes it easier to install and manage edge computing solutions. It emphasises how crucial it is to have a DBMS that is effective and scalable for managing time-series data in EdgeX installations. The literature study explores the properties of time-series data and the difficulties in successfully storing and processing it. It goes over the specifications for a good DBMS for managing time-series data, such as high write and read throughput, quick access, data compression, and adaptable query capabilities. The focus of the study shifts to RedisDB, an in-memory data structure store that offers time-series data storage that is quick and scalable. It examines RedisDB's attributes and design, demonstrating how it can effectively manage time-series data by employing data structures like sorted sets and strings. The literature study also addresses several ways that are currently in use and research projects that have used RedisDB to manage time-series data in edge computing settings. It summarises their conclusions and focuses on the benefits and drawbacks of adopting RedisDB for edge gateway applications, including enhanced performance, decreased storage needs, and streamlined data retrieval. The report concludes by summarising the most important conclusions from the

literature research and highlighting RedisDB's applicability as a time-series DBMS for EdgeX-based lightweight edge gateways. Additionally, it highlights topics for further study and development, such as investigating integration options with other edge computing frameworks and optimising RedisDB for resource-constrained edge devices. Overall, this literature review research offers insightful information about the use of RedisDB for managing time-series data in lightweight edge gateways based on EdgeX. It is a helpful tool for academics and professionals looking to employ RedisDB's features in edge computing settings. [12]

Chapter 3

Methodology

Users can get data and analysis about their product on the platform IOT MANAGER - DEVICE MANAGEMENT. Users of the system can view and manage all of their device through a portal. The heartbeat messages, which include critical information like the device's Mac address, IP address, and other device's telemetry information, are set to be transmitted by each device at regular intervals.

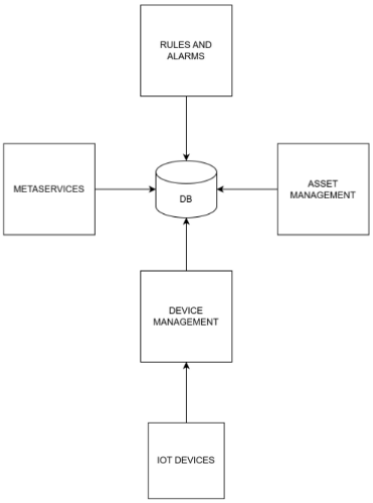

Figure 3.1: IoT Manager

The system contains a login process that offers choices for both standard and Google logins. For the standard sign-in, users can either establish a new username and password or log in using their Google account. The user will be sent to landing pages that display all the devices that specific user owns after a successful login. The users may access that website to view information about their devices, including the last sync time, mac address, memory and ram

use, and other telemetry data. The number of devices that went online and offline is also included in that landing page. According to the last sync time, if a device's last sync time exceeds a certain time limit, it is deemed to be offline; however, if it is within a predetermined time range of the present time, it is considered to be online.

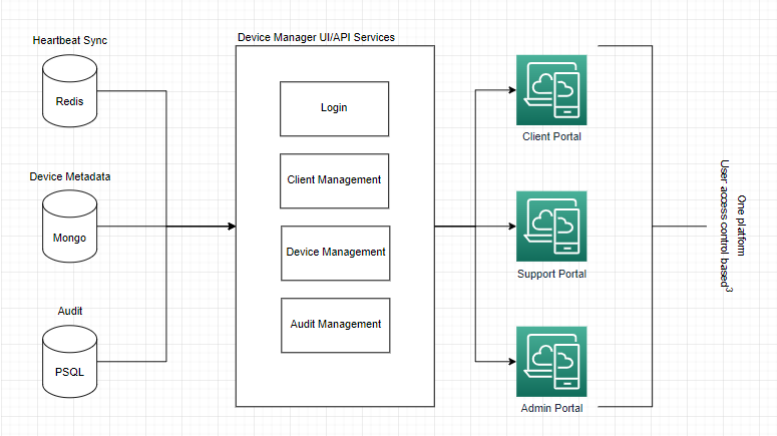

Figure 3.2: Device Manager UI/API Service

Device Manager Core: The Device Manager Core is the heart of the portal. It is responsible for orchestrating different workflows, receiving device events, and updating databases. The application should allow users to add, remove, configure, and monitor the IoT devices installed in the manufacturing plant. The application should provide data visualization tools such as charts, graphs, and dashboards to help users analyze the data generated by the devices. The application should have an alert and notification system that notifies users of any anomalies or issues with the devices or the manufacturing process. The application should provide configuration management tools that allow users to configure the devices and sensors according to the manufacturing plant's needs.

The following components are key API flows:

- Device Registration API: This API provisions the device. It also generates a device ID which the device team prints on the device.
  (Important Note: The device can also be auto-registered on the first heartbeat.)
- Heartbeat API: This API will be used by the device to send a heartbeat to the portal.
- Telemetry API: In this API, additional data like CPU usage, disk usage, and other device information is sent by the device.
  (Important Note: Telemetry can be sent along with heartbeat.)

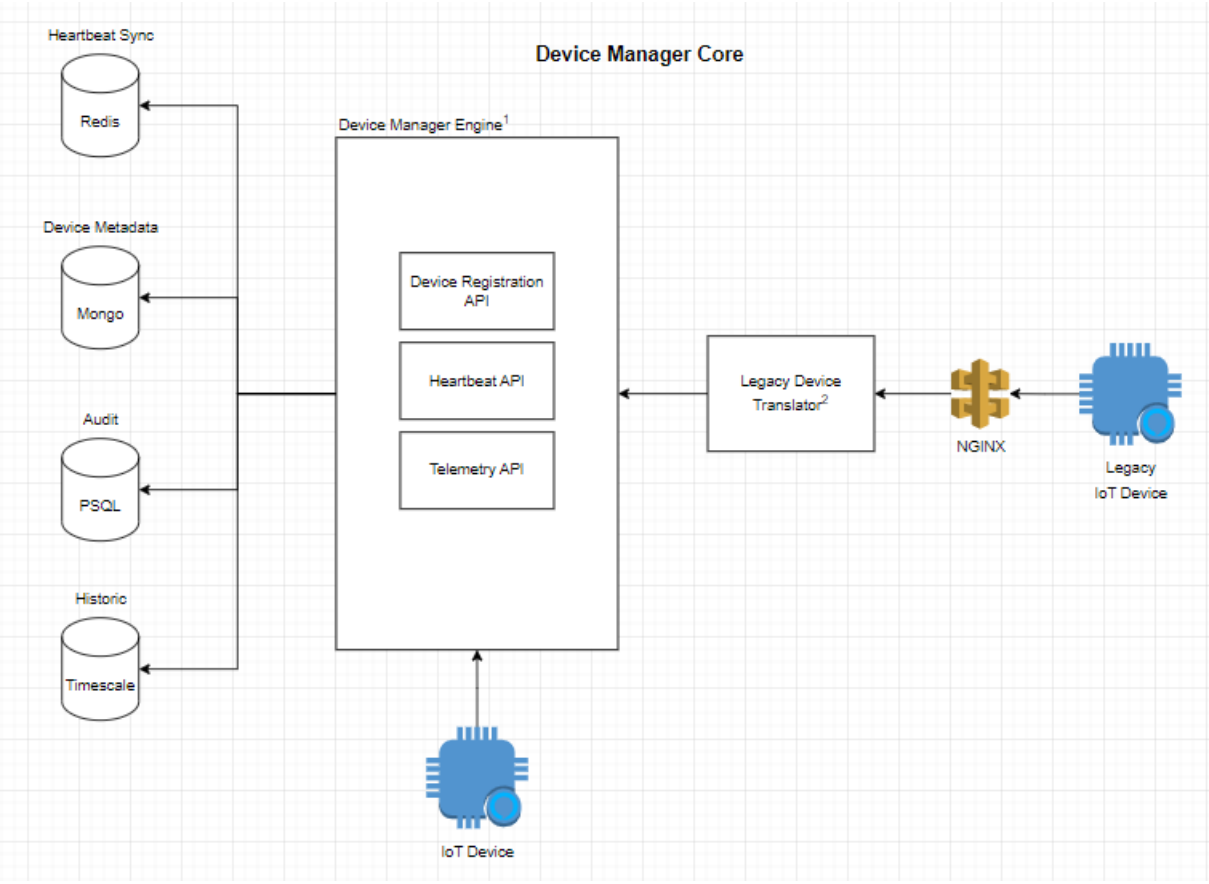

Figure 3.3: Device Manager Core

3.1 Key Features of IOT MANAGER - DEVICE MANAGEMENT

- Device Registration
- Telemetry data storage for
  - New Devices
  - Existing Devices in the portal
- Audit Logs
- Device Tagging

3.2 Module Description

3.2.1 Device Registration

The industrial devices and sensors are registered with the web application. The devices are identified by their unique IDs (Mac Address), and their metadata is collected, including location, type, and specifications. When the device is manufactured, a heartbeat will be sent from the test site to update its telemetry data. Then it will check in DB if device id mac is already present; if present, it is considered an existing device, and its data gets updated. If Mac is not present in DB, it is regarded as a new device, and data insertion happens in DB. If the device doesn't have a DeviceID, it will generate a unique ID based on its SoftwareName.

3.2.2 Telemetry Data

Telemetry data is an essential aspect of monitoring and managing IOT devices, and it plays a crucial role in improving their performance, efficiency, and reliability. Telemetry data is the information IoT devices collect and transmit about their status, behavior, and environment. This data is used by IOT developers, system administrators, and end-users to analyze the performance of the devices, identify issues, and optimize the operation of the devices. Telemetry data refers to the information collected and transmitted by IoT devices about their performance, environment, and activities. This data includes sensor readings, such as temperature, humidity, pressure, light intensity, motor status, battery level, and other relevant parameters from various equipment in manufacturing analysis. This data is visualized as charts and tables in our portal for easy usability and readability. In addition to providing valuable insights into the performance of IoT devices, telemetry data is also used for predictive maintenance. By analyzing the data, developers and system administrators can predict when a device will likely fail or require maintenance and take proactive measures to prevent downtime and minimize the risk of failure.

3.2.3 Device Tagging

The system's devices are all identified by tags. Filtering and access control both employ tagging. A device put in distillation, for example, may have the tag distillation_section. A site user has the ability to create tags, add tags to devices, and remove tags. Only after receiving

the Super Admin's permission can the tag be bound to a device. All newly added devices will automatically get the tag NEW-DEVICE.

The following are key metadata that a device will contain. The same will be used for filter and access control:

- Customer name
- Vendor Name
- Device Type
- Country, State and City

Default Tag will be:
customer_name$vendor_name$device_type$country_name$state_name$ city_name

3.2.4 Legacy Device Translator

We have around 16000 devices in the field. Many of them are in sensitive areas with limited connectivity. To avoid a firmware change, Nginx will be used to proxy pass to a translator layer.

3.2.5 Audit Logs

Audit logs are essential for tracking and documenting device management-related activities and occurrences in an IOT MANAGER - DEVICE MANAGEMENT. Accountability, security, and compliance are all facilitated by audit logs, which offer a thorough record of actions made by system administrators, users, and the system itself.
The audit logs should include the following important elements:

- **Event Logging:** Events including device registrations, firmware updates, configuration changes, and user actions like logins and access tries are all recorded in audit logs. Every event is timestamped and frequently contains pertinent information, such as the person or system account involved, the action's type, and any pertinent parameters or data.
- **Security Monitoring:** Audit logs can be employed to track security-related activities and find potential dangers or weaknesses. The audit log, for instance, can record an event and help with the investigation of security problems if a device is compromised or if a crucial configuration change is performed.

- **Compliance and Governance:** Audit logs assist organisations in adhering to internal standards and legal requirements. Organisations can prove compliance with standards, such as data protection laws (like the GDPR) or industry-specific compliance frameworks (like ISO 27001), by keeping a thorough record of their actions.

- **Troubleshooting and Forensics:** When issues arise with IoT devices or the management system, audit logs become valuable for troubleshooting and forensic analysis. Detailed event logs enable administrators to trace the sequence of actions leading up to the problem, identify the root cause, and take appropriate remedial measures.

- **User Accountability and Access Control:** By documenting user actions, audit logs promote accountability and transparency. To find instances of privilege abuse, unauthorised access attempts, or other policy infractions, administrators can examine the logs. If required, disciplinary measures may be taken using this information to enforce access control regulations.

- **Log Integrity and Protection:** The protection and integrity of audit logs must be guaranteed. For the purpose of preventing unauthorised additions or deletions, logs should be tamper-proof and kept in a safe location. To safeguard the sensitive data in the logs, access limits and encryption should be put in place.

- **Log Analysis and Reporting:** To get insights into system performance, user behaviour, and compliance patterns, audit logs may be analysed and summarised. Tools and techniques for log analysis can assist find trends, abnormalities, or suspicious activity, enabling proactive security and operational efficiency measures.

In this system, audit logs offer a comprehensive record of events, activities, and user actions. In order to ensure security, compliance, and effective troubleshooting, they are crucial to the administration of IoT devices.

3.3 System Specifications

On the basis of the requirements, the application's development architecture recognised for this project is described in this section.

3.3.1 Hardware Requirements

- Hard Drive: Minimum 4 MB (Recommended 6 GB or more)
- Memory (RAM): Minimum 1 GB (Recommended 4 GB or above)
- Processor: Minimum 1 GHz (Recommended 2GHz or more)

3.3.2 Software Specification

- Front End : Angular JS
- Back End : FastAPI-Python
- Web Server : Nginx
- Database : MongoDB, PostgreSQL, Redis, TimeScaleDB
- Operating System : Windows, Linux, Mac - (Any OS)
- Web Browser : Supported by most of the Browsers

3.3.3 Software Description

- **Python**

  A popular general-purpose, high-level, interactive, interpreted, object-oriented programming language is Python.Developers may quickly and simply convey notions using the Python programming language.Python has a sizable library and a sizable community of users.The majority of operating systems, including Windows, Linux, and macOS, support Python.

- **Angular**

  An open-source framework for creating web-based applications is called Angular. It is built on TypeScript. It is one of the most widely used UI frameworks right now. Angular

is a platform that makes it easy to construct online apps, and it also makes it easy to create web applications that are quicker. A quick and effective template syntax is offered by Angular for creating UI views. The primary characteristics of Angular include It is based on components, the use of TypeScript utilising ideas of services that might be used to many applications.

- **FastAPI**

  Python APIs may be created using the contemporary, high-performance web framework FastAPI. It has a number of benefits and is becoming more and more well-liked among developers. Due to the fact that it is based on the Starlette asynchronous web framework, one of its main features is that it performs well. As a result, FastAPI can process a lot of requests quickly and with little resource use. It offers an effective blend of speed, type checking, usability, and asynchronous functionality. These benefits make it a great option for creating scalable, dependable, and efficient Python APIs, enabling developers to build cutting-edge online services.

- **Nginx**

  Reverse proxy and lightweight web server Nginx is renowned for its effectiveness and scalability. It is well-liked for handling high traffic loads and enhancing website speed since it is built to manage a large number of concurrent connections and deliver static content well. Additionally, Nginx has the ability to function as a reverse proxy, effectively allocating inbound requests to several backend servers. Nginx is a recommended option for enhancing web server performance and enhancing the overall dependability and speed of online applications due to its flexibility, stability, and vast module ecosystem.

- **MongoDB**

  Popular NoSQL database MongoDB is renowned for its adaptability, scalability, and simplicity of usage. It is a document-oriented database that stores information in the adaptable, JSON-like BSON format. Due to its schema-less design, MongoDB is highly suited for managing unstructured or partially organised data since it allows for dynamic and developing data structures. MongoDB can grow horizontally to accommodate massive volumes of data and significant traffic loads because to its distributed design and intelligent sharding features. Additionally, it enables robust searching and indexing

features, facilitating effective data retrieval and analysis. In conclusion, MongoDB offers a reliable and scalable option for handling and modifying data in contemporary applications.

- **PostgreSQL**

  Powerful object relational databases include PostgreSQL. It has strong community support because a strong open source community has been supporting its development for the past 30 years. The majority of businesses today, like Uber, Netflix, Spotify, and Instagram, use Posgres in their infrastructure. PgAdmin or the command-line utility psql can be used to access Postgres.

- **RedisDB**

  A highly adaptable and effective solution for a variety of use scenarios, RedisDB is an open-source in-memory data structure store. It excels at task queuing, messaging systems, real-time analytics, and caching. The key-value data format used by RedisDB makes data retrieval and storing incredibly quick. It allows for versatile data processing by supporting a broad variety of data types, including strings, hashes, sets, lists, and more. Fault tolerance and high availability are guaranteed by RedisDB's integrated replication and clustering features. Data durability is guaranteed even in the case of system failures because to its capacity to store data to disc. RedisDB is a popular option for applications that need fast access to information and real-time processing due to its simplicity, speed, and adaptability.

- **TimeScaleDB**

  On top of PostgreSQL, TimeScaleDB is an open-source time-series database designed for processing massive amounts of time-series data. It combines PostgreSQL's scalability and dependability with specialised capabilities for effective time-stamped data storage, retrieval, and analysis. Data is arranged into hypertables by TimeScaleDB, which also automatically partitions and compresses data depending on time intervals. Faster data analysis is made possible by its enhanced time-series functions, which include time-based aggregations, downsampling, and continuous aggregates. TimeScaleDB is an effective tool for application dealing with high-frequency, time-series data, such as IoT monitoring, financial systems, and log analytics, thanks to its smooth interaction with well-known data visualisation and analytics tools.

Chapter 4

RESULT AND DISCUSSION

A dependable and effective method for administering and monitoring Internet of Things devices is offered by IOT MANAGER - DEVICE MANAGEMENT. Users can get the most recent device information, such as the device name, software name, MAC address, and online/offline status, from the heartbeat messages that are recorded in real-time. All device data is securely stored and managed by the MongoDB database and heartbeat engine. Users can simply manage and keep an eye on large-scale IoT deployments with the help of IoT Manager, increasing overall efficiency and lowering the possibility of device failure. Additionally, customers may proactively anticipate and address potential problems because to the platform's capacity to analyse heart-beat data, which reduces downtime and ensures peak performance. The portal's login and user management systems allowed users to readily access information about their devices.

It is impressive that the system can manage the details of up to 16000 devices at once from around the world. A service called "Heart-Beat Service" collects the data from the devices that are installed at various locations in industrial sites, and it then stores the device specifics in the meta database.

4.1 Testing Methods

By using criteria that are expected by the user or the organisation, testing assures that the system is error-free. A system's performance might range from high-end to low-end depending on the setting in which it functions.

- **Functional Testing:** Ensure that all the functionalities of the portal, such as user login, device overview, remote monitoring, device management (OTA, Logs), metrics dashboards, and alert system are tested thoroughly. Test each functionality with different user roles (Super Admin, Support Admin, Support User, Site User, Vendor, Customer) to validate their access rights and permissions.

- **User Interface (UI) Testing:** Validate the user interface of IOT MANAGER - DEVICE MANAGEMENT to ensure it is user-friendly, visually appealing, and responsive across different devices and browsers. Verify that all UI elements are displayed correctly, and users can easily interact with them.

- **Data Validation:** Validate the accuracy and integrity of data displayed in the portal, such as device information, metrics, logs, and alerts. Verify that data is retrieved and displayed correctly from the backend and any data inputs from users are validated properly.

- **Performance Testing:** Test the performance and scalability of IOT MANAGER - DEVICE MANAGEMENT by simulating different loads and stress on the portal. Verify that it can handle the expected number of concurrent users and devices, and response times are within acceptable limits.

- **Security Testing:** Validate the security measures implemented in IOT MANAGER - DEVICE MANAGEMENT, such as user authentication, authorization, and data encryption. Test for vulnerabilities such as SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF) to ensure the portal is secure against potential attacks.

- **Compatibility Testing:** Test IOT MANAGER - DEVICE MANAGEMENT on different browsers, operating systems, and devices to ensure it is compatible and functions correctly across different environments.

- **Error Handling:** Test the portal's error handling capabilities by intentionally triggering errors or invalid inputs and verifying that appropriate error messages are displayed, and the portal gracefully handles such scenarios.

- **Regression Testing:** Conduct regression testing to ensure that new changes or updates to the portal do not break existing functionalities and that previously fixed issues do not resurface.

- **Usability Testing:** Involve end-users in usability testing to gather feedback on the overall usability, ease of use, and satisfaction with IOT MANAGER - DEVICE MANAGEMENT. Incorporate user feedback to continuously improve the portal.
- **Test Environment:** Create a test environment that replicates the production environment as closely as possible to ensure accurate testing results.
- **Test Data:** Use realistic and representative test data to ensure comprehensive testing of IOT MANAGER - DEVICE MANAGEMENT, including various scenarios and edge cases.

4.2 Output Screens and Results

1. Login Page

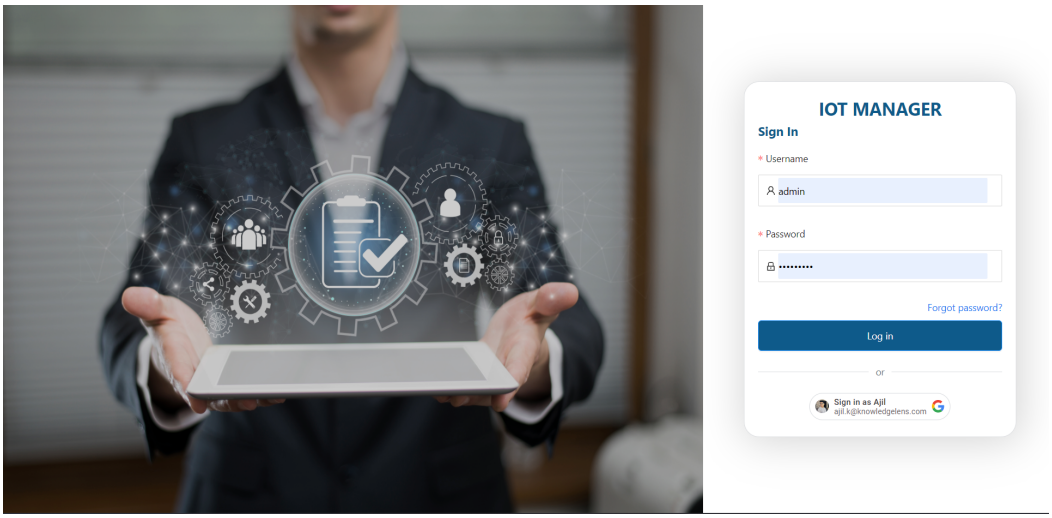

Figure 4.1: Login Page

User can log in by giving username and password or by their company Gmail ID (Google login) option.

2. Home Page

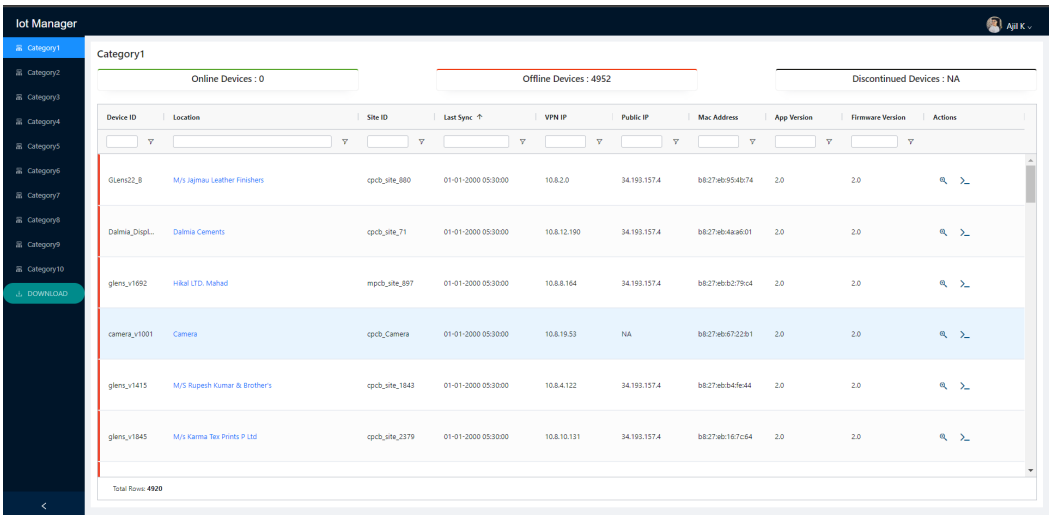

Figure 4.2: Home Page

It's the landing page after user login. Shows all details of different IoT devices and can be used to search IoT devices.

3. Device Analytics

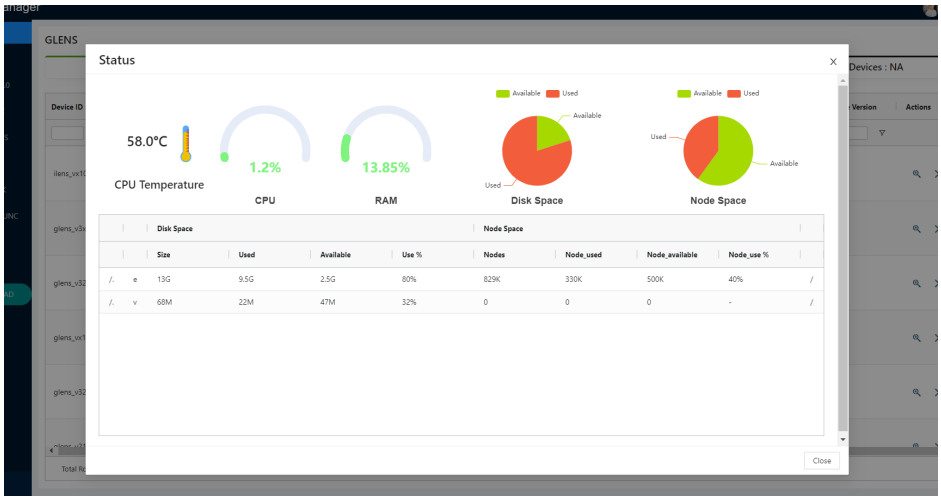

Figure 4.3: Device Analytics

Here we can get the analytical data of each devices - Telemetry Data. This view will open when we click on the respective device's action button.

4. Audit Logs

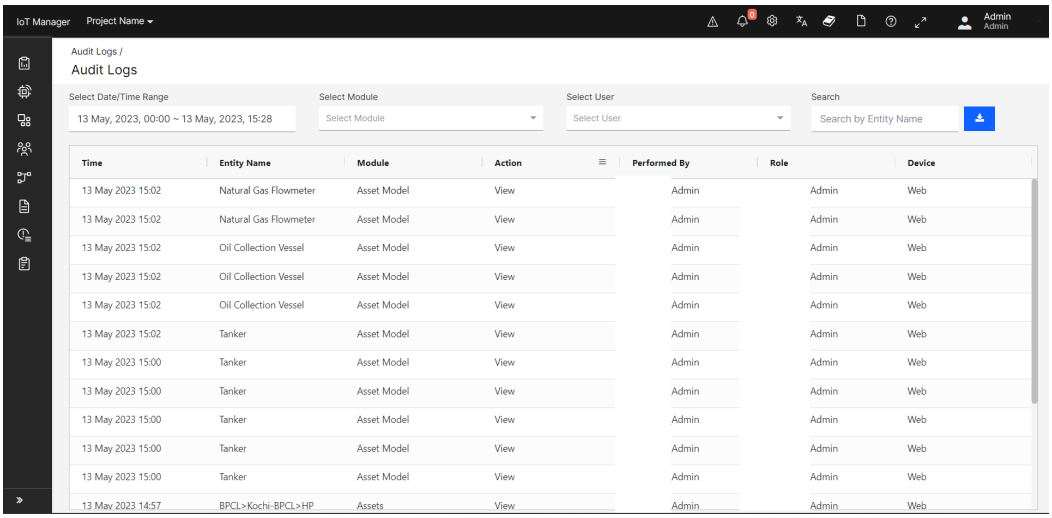

Figure 4.4: Audit Logs

Here we can view entities audit logs. In this page we can able to view the audit data of each device's record.

5. Download Page

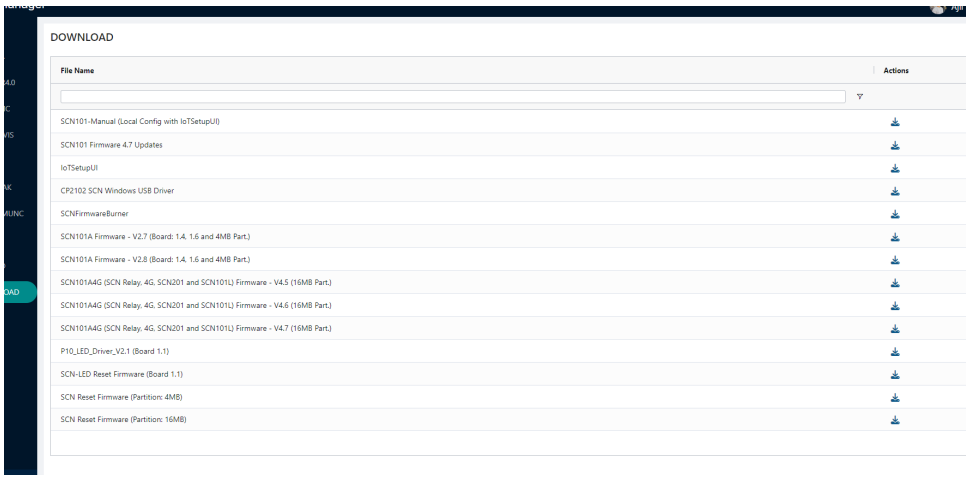

Figure 4.5: Download page

In this page the user can able to view and download device's update informations and release date. Also it provide links to download latest version of the softwares.

Chapter 5

CONCLUSION

IOT MANAGER - DEVICE MANAGEMENT portal's objective is to make it quick and simple for users to acquire device information and gain understanding of the data each device provides. This online application is made to manage a lot of users and devices. The application's architecture is scalable, allowing it to handle growing traffic and data volumes without suffering performance degradation. The system can be made faster and more dependable by utilising more modern and sophisticated technologies like FastAPI, redis, MongoDB, and Python. With IoT devices, security is a crucial concern, and the web application is built with strong security features to fend off potential cyber-attacks. This covers secure communication protocols, authentication, authorization, and encryption. The online application's user interface is simple to use, intuitive, and welcoming. It gives users the opportunity to rapidly and simply monitor, control, and configure devices. The portal offers real-time monitoring tools to find and address any anomalies or device problems. Models for alerting systems and real-time data analysis are included in this.

This website provides a centralised platform for managing and keeping track of their IoT and remote edge devices. When it comes to monitoring and maintaining massive networks of devices, the application is appropriate for a variety of industries. The main features of the application allow users to efficiently manage their devices, spot and fix problems quickly, and optimise their operations. These features include remote device monitoring, real-time alerts, user administration, data analysis tools, and alerting systems. This application is a useful solution for companies of all sizes to enhance device management skills and boost productivity and efficiency.

5.1 Future Enhancement

Enhanced Device Integration: Continuously expanding the compatibility of the portal with a wider range of devices, protocols, and communication technologies to accommodate a broader spectrum of IoT devices and networks.

- **Advanced Analytics and Insights:** Incorporating advanced analytics capabilities to gain deeper insights from device data, including predictive analytics, anomaly detection, and machine learning algorithms. This can enable proactive device management and predictive maintenance, optimizing device performance and minimizing downtime.
- **Expanded Device Management Features:** Adding more robust device management features, such as remote device configuration, firmware rollback, and device grouping for easier management of multiple devices with similar configurations.
- **Customizable Dashboards and Reports:** Providing users with the ability to customize dashboards and generate customized reports based on their specific requirements. This can enable users to visualize and analyze device data in a way that is most relevant to their business needs.
- **Enhanced Security Features:** Strengthening the security features of the portal, including multi-factor authentication, role-based access control, and continuous security monitoring, to ensure the protection of sensitive device data and prevent unauthorized access.
- **Integration with Enterprise Systems:** Integrating the portal with other enterprise systems, such as asset management systems, customer relationship management (CRM) systems, and enterprise resource planning (ERP) systems, to streamline workflows, automate processes, and optimize device management within the broader business ecosystem.
- **Mobile Application:** Developing a mobile application for the portal to provide users with on-the-go access to device information, alerts, and management features, enabling remote monitoring and management of devices from anywhere, anytime.
- **Scalability and Performance:** Continuously improving the scalability and performance of the portal to handle increasing volumes of devices, data, and users, ensuring smooth

and efficient operation even as the number of devices and users grows.

- **Industry-specific Features:** Developing industry-specific features and functionalities tailored to the unique requirements of different industries, such as healthcare, manufacturing, transportation, agriculture, and smart cities, to cater to the specific needs and use cases of those industries.

- **User Experience Enhancements:** Continuously enhancing the user experience of the portal through user feedback, usability studies, and iterative design improvements, ensuring that the portal remains user-friendly, intuitive, and efficient for users of all skill levels.

REFERENCES

[1] Nikolov, Neven, and Ognyan Nakov, *"Design and Implementation of a System for Remote IoT Device Management", 2020 28th National Conference with International Participation (TELECOM). IEEE, 2020.*

[2] Jose, Benymol, and Sajimon Abraham, *"Exploring the merits of nosql: A study based on mongodb.", 2017 International Conference on Networks Advances in Computational Technologies (NetACT). IEEE, 2017.*

[3] Sharma Akshansh, Firoj Khan, Deepak Sharma, and Dr. Sunil Gupta *"Python: the programming language of future." International Journal of Innovative Research in Technology 6.12 (2020): 115-118.*

[4] Lathkar, Malhar, *"Introduction to FastAPI." High-Performance Web Apps with FastAPI: The Asynchronous Web Framework Based on Modern Python. Berkeley, CA: Apress, 2023. 1-28.*

[5] Ramos, Miguel, Marco Tulio Valente, and Ricardo Terra. *"AngularJS performance: A survey study.", IEEE Software 35.2 (2017): 72-79.*

[6] Győrödi, Cornelia, *"A comparative study: MongoDB vs. MySQL." 2015 13th International Conference on Engineering of Modern Electric Systems (EMES). IEEE, 2015.*

[7] Srinath, K. R. *"Python–the fastest growing programming language.", International Research Journal of Engineering and Technology 4.12 (2017): 354-357.*

[8] Makris, Antonios, *"MongoDB Vs PostgreSQL: A comparative study on performance aspects." GeoInformatica 25 (2021): 243-268.*

[9] Sacco, Andres, *"Redis: Key/Value Database." Beginning Spring Data: Data Access and Persistence for Spring Framework 6 and Boot 3. Berkeley, CA: Apress, 2022. 189-212.*

[10] Shen, Liqun, *"Meteorological Sensor Data Storage Mechanism Based on TimescaleDB and Kafka." Data Science: 5th International Conference of Pioneering Computer Scientists, Engineers and Educators, ICPCSEE 2019, Guilin, China, September 20–23, 2019, Proceedings, Part I 5. Springer Singapore, 2019.*

[11] Perumal, Thinagaran, Soumya Kanti Datta, and Christian Bonnet, *"IoT device management framework for smart home scenarios." 2015 IEEE 4th global conference on consumer electronics (GCCE). IEEE, 2015.*

[12] Kim, Jaein, *"A study on Time-series DBMS Application for EdgeX-based lightweight edge gateway." 2020 International Conference on Information and Communication Technology Convergence (ICTC). IEEE, 2020.*

[13] Puri, Vikram, *"AI-based botnet attack classification and detection in IoT devices." 2022 IEEE International Conference on Machine Learning and Applied Network Technologies (ICMLANT). IEEE, 2022.*

[14] Geetha, G., *"Interpretation and Analysis of Angular Framework." 2022 International Conference on Power, Energy, Control and Transmission Systems (ICPECTS). IEEE, 2022.*

[15] Pupezescu, Valentin, Marilena-Cătălina Pupezescu, and Lucian-Andrei Perișoară., *"Optimizations of Database Management Systems for Real Time IoT Edge Applications." 2022 23rd International Carpathian Control Conference (ICCC). IEEE, 2022.*

APPENDIX

Screenshots

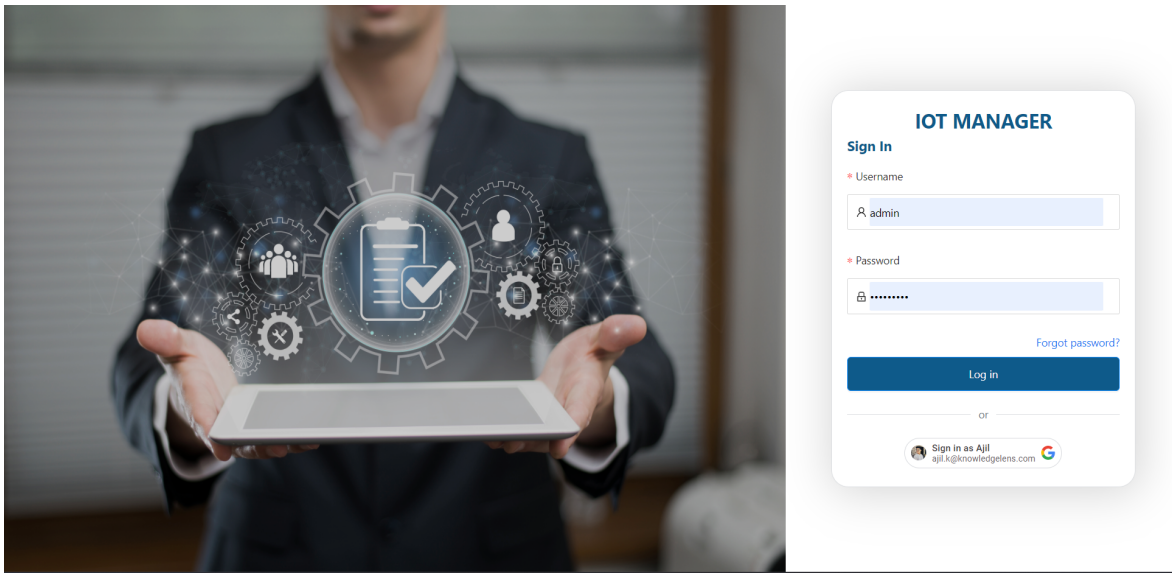

Figure A.1: Login Page

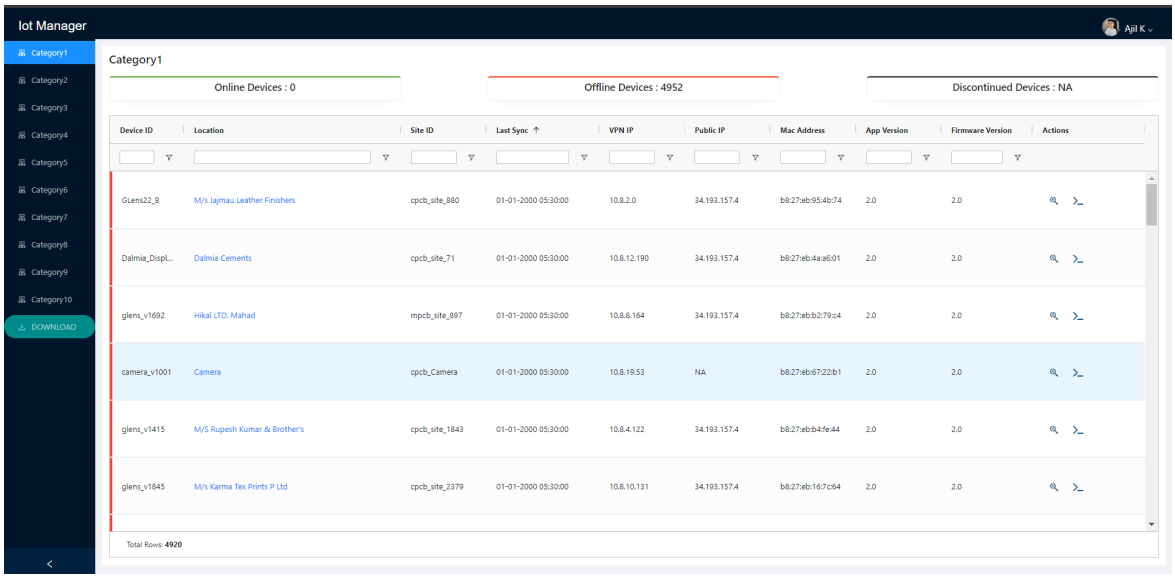

Figure A.2: Dashboard

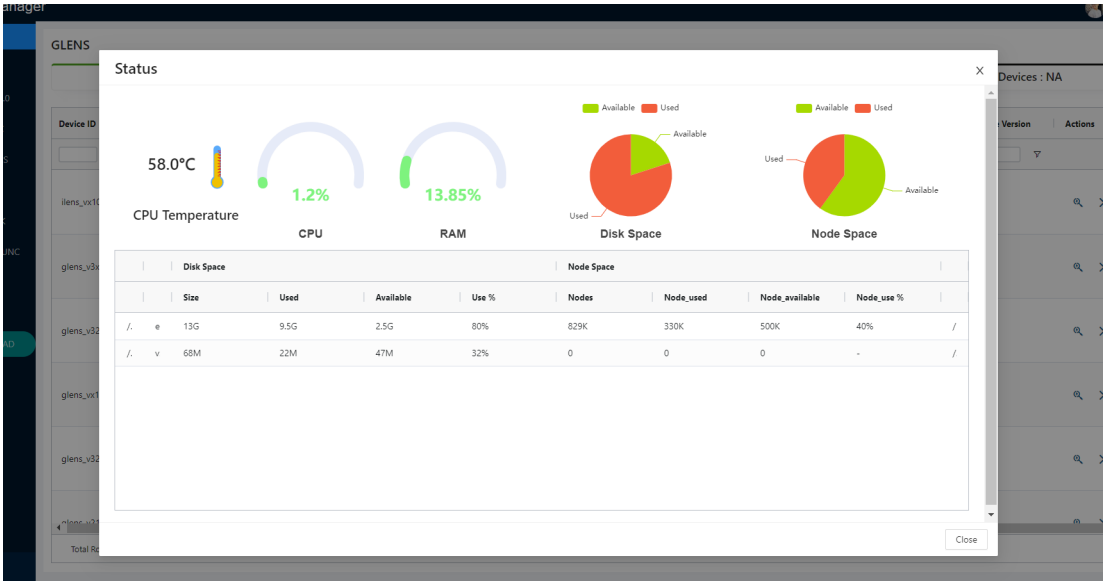

Figure A.3: Device Analytics

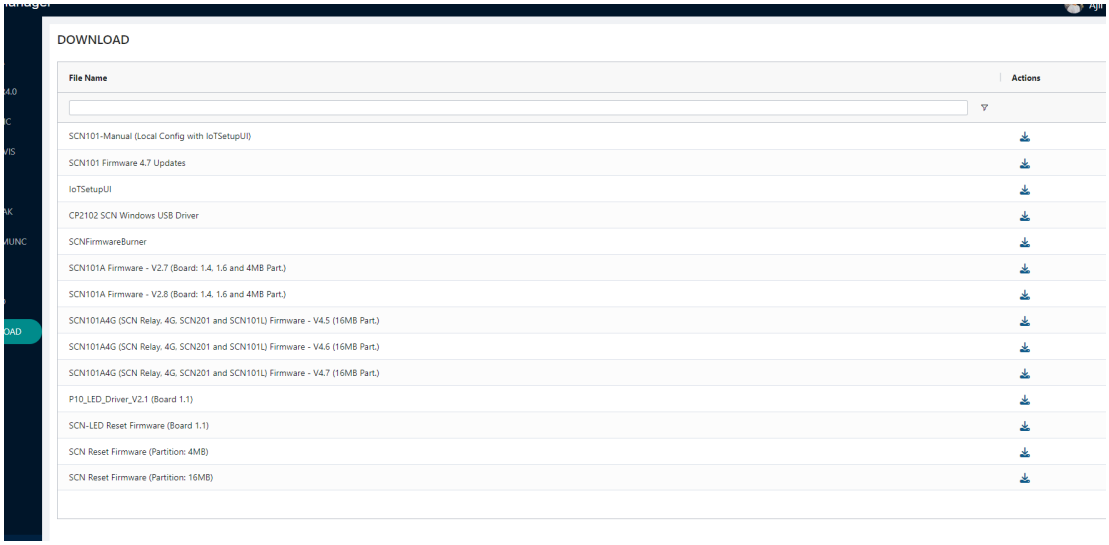

Figure A.4: Download Page

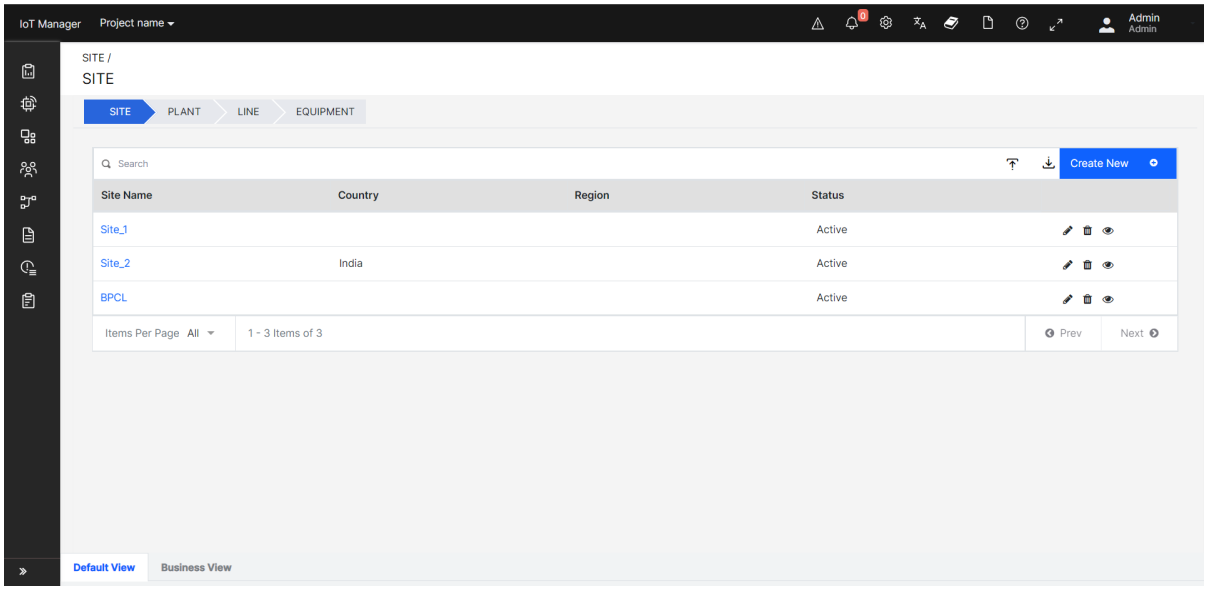

Figure A.5: Site View

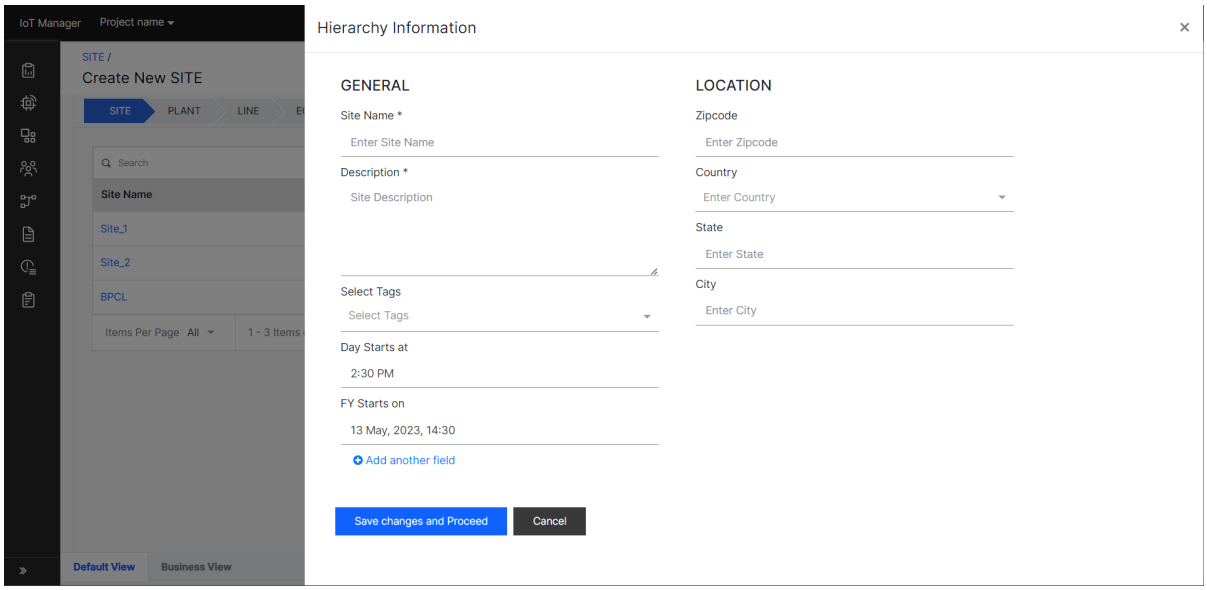

Figure A.6: Add new Site

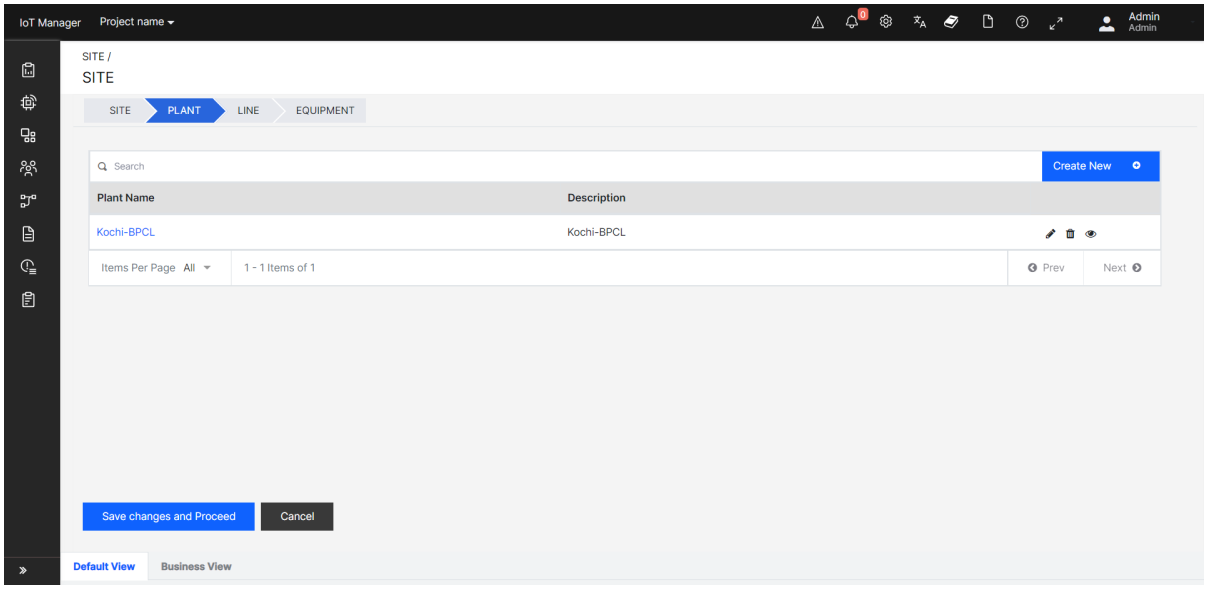

Figure A.7: Plant View

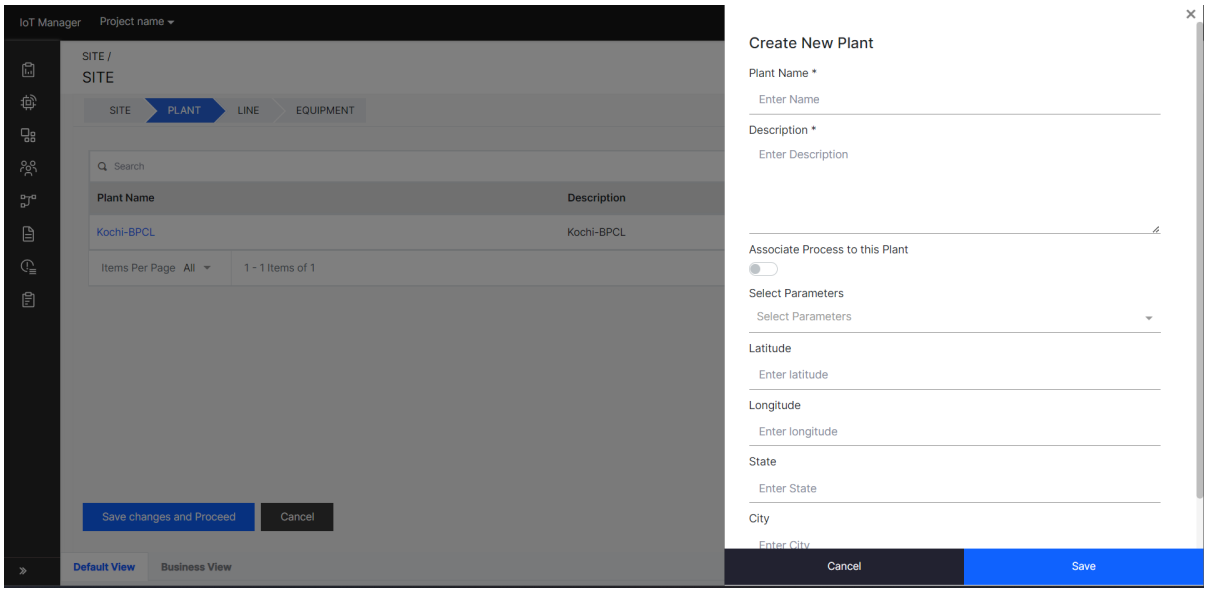

Figure A.8: Add new Plant

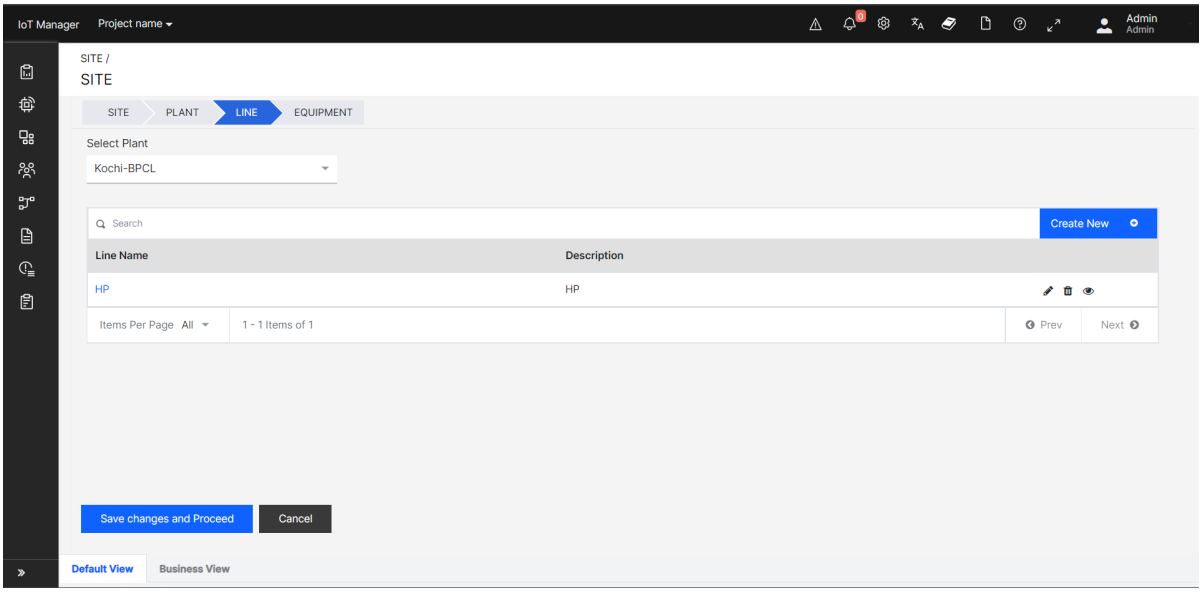

Figure A.9: Line View

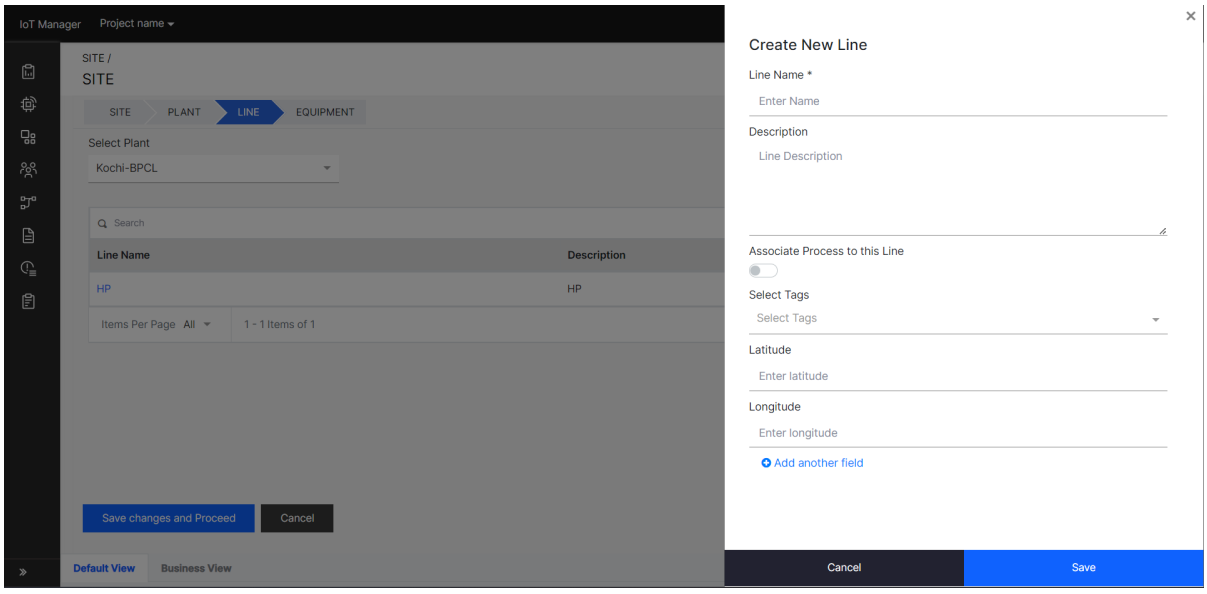

Figure A.10: Add new Line

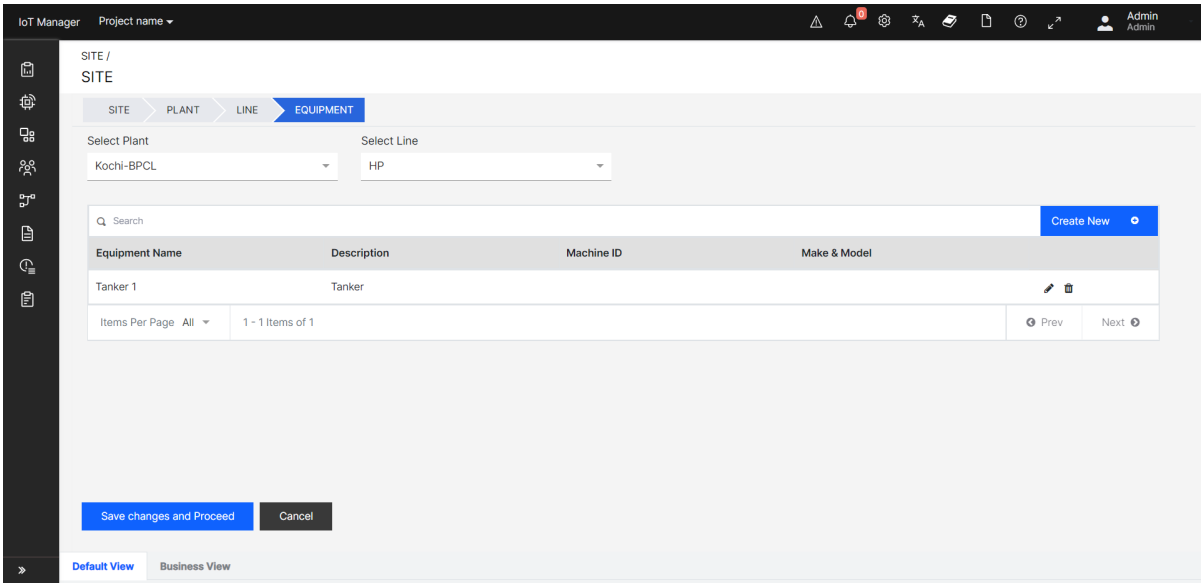

Figure A.11: Equipment View

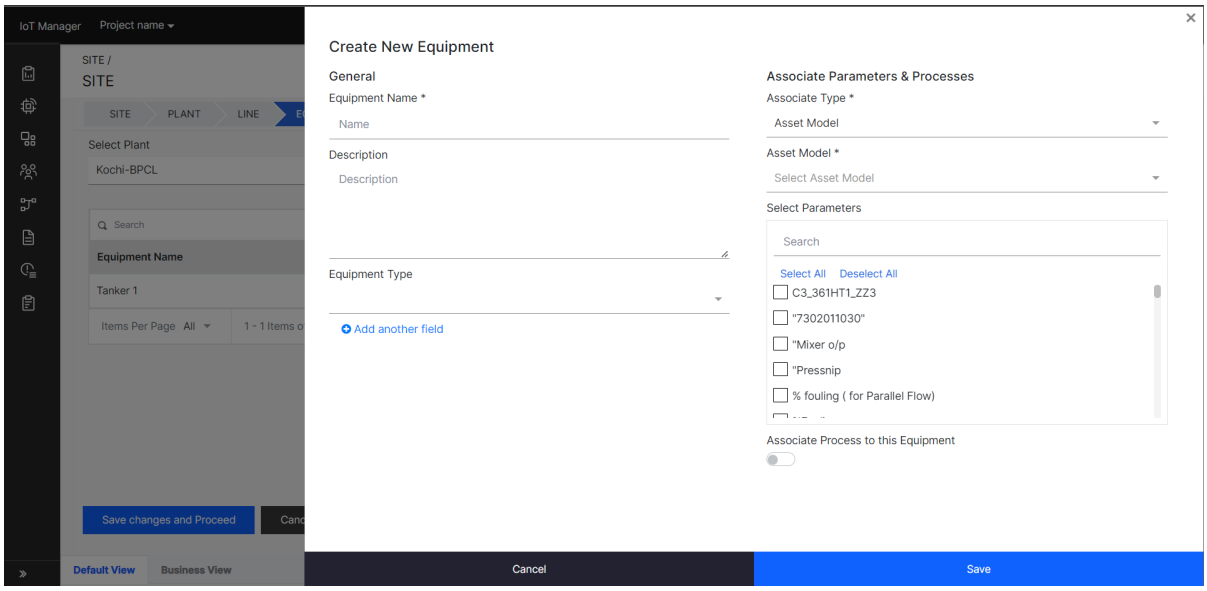

Figure A.12: Add new Equipment