Containerized microservice framework on edge device for health care IOT system

By

Hafiz Talha Bin Hussain

MS-IT 20

Supervisor

Dr. Farzana Janbeen

Department of Computing

A thesis submitted in partial fulfillment of the requirements for the degree of Masters of Science in Information Technology (MS IT)

In

School of Electrical Engineering & Computer Science (SEECS) ,

National University of Sciences and Technology (NUST),

Islamabad, Pakistan.

(July 2023)

THESIS ACCEPTANCE CERTIFICATE

Certified that final copy of MS/MPhil thesis entitled "Containerized microservice framework on edge device for healthcare IoT systems" written by HAFIZ TALHA BIN HUSSAIN, (Registration No 00000318409), of SEECS has been vetted by the undersigned, found complete in all respects as per NUST Statutes/Regulations, is free of plagiarism, errors and mistakes and is accepted as partial fulfillment for award of MS/M Phil degree. It is further certified that necessary amendments as pointed out by GEC members of the scholar have also been incorporated in the said thesis.

Signature: _____Farzana_____

Name of Advisor: _____Dr. Farzana Jabeen__

Date: _____

HoD/Associate Dean:_____

Date: _____

Signature (Dean/Principal): _____

Date: _____

Approval

It is certified that the contents and form of the thesis entitled "Containerized microservice framework on edge device for healthcare IoT systems" submitted by HAFIZ TALHA BIN HUSSAIN have been found satisfactory for the requirement of the degree

Advisor: Dr. Farzana Jabeen

Signature: Farzana

Date: **07-Jul-2023**

Committee Member 1:Dr. Asad Waqar Malik

Signature: Asad

07-Jul-2023

Committee Member 2:Dr. Syed Muhammad Bilal Ali

Signature: S. M. Bilal

Date: **08-Jul-2023**

Signature: _____

Date: _____

Dedication

This thesis is dedicated to all deserving children who do not have access to quality education, especially young girls.

Certificate of Originality

I hereby declare that this submission titled "Containerized microservice framework on edge device for healthcare IoT systems" is my own work. To the best of my knowledge it contains no materials previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any degree or diploma at NUST SEECS or at any other educational institute, except where due acknowledgement has been made in the thesis. Any contribution made to the research by others, with whom I have worked at NUST SEECS or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except for the assistance from others in the project's design and conception or in style, presentation and linguistics, which has been acknowledged. I also verified the originality of contents through plagiarism software.

Student Name:HAFIZ TALHA BIN HUSSAIN

Student Signature: 

Acknowledgments

Glory be to Allah (S.W.A), the Creator, the Sustainer of the Universe. Who only has the power to honor whom He pleases, and to abase whom He pleases. Verily no one can do anything without His will. From the day, I came to NUST till the day of my departure, He is the only one Who blessed me and opened ways for me, and showed me the path of success. There is nothing that can pay back for His bounties throughout my research period to complete it successfully. I would like to express my gratitude to my supervisor Dr. Asad Waqar Malik for his support and guidance through out my thesis work and my committee members for their valuable suggestions. I would like to thank Dr. Jeremy Murray for his continuous involvement with my work his help and support has played an important role in completion of my thesis.

Hafiz Talha Bin HUssain

Contents

List of Figures

Abstract

The aging population in Western and developed countries, combined with advancements in medical sciences, has resulted in a growing demand for smart home healthcare monitoring systems. These systems cater to the needs of elderly individuals who prefer to age in place rather than relocate healthcare facilities. To address this demand, researchers have explored the integration of Internet of Things (IoT) techniques and lightweight virtualization technology in edge computing. This approach offers improved resource management, service isolation, and seamless deployment of diverse hardware components. In this study, a containerized architectural framework is proposed to enable the concurrent deployment of multiple IoT applications in senior citizen homes. The framework included sensor networks and containerized microservices, which are centrally managed and orchestrated following the principles of DevOps.

The primary objective of the proposed smart healthcare monitoring system is to monitor the activities and well-being of occupants and provide crucial information to healthcare providers. By tracking various parameters and identifying signs of discomfort, the system facilitates timely and effective assistance from healthcare professionals. The utilization of containerization technology ensures efficient management and scalability of the deployed applications, enabling customization and adaptation to individual needs. This research aims to bridge the gap between traditional healthcare settings and the comfort of home, offering elderly individuals a comprehensive and personalized healthcare monitoring solution that promotes independence and enhances their overall quality of life.

Chapter 1

Introduction

1.1 Background Information

1.1.1 Internet of Things

The Internet of Things (IoT) has emerged as a groundbreaking technology, connecting various objects and devices to the Internet to enable data collection, exchange, and analysis. Within the healthcare sector, the IoT holds tremendous potential for revolutionizing patient care, enhancing healthcare delivery, and improving overall system efficiency. By integrating medical devices, sensors, wearables, and other interconnected objects, healthcare providers can access real-time patient data, remotely monitor vital signs, and leverage advanced analytics to develop personalized treatment plans and proactive interventions[24]

. The Internet of Things (IoT) has brought about significant transformations in various sectors and holds the potential to reshape our lives in numerous ways. It refers to a network of interconnected physical objects, devices, sensors, and software applications that autonomously exchange data over the Internet, enabling automation, real-time monitoring, and intelligent decision-making.

- **Future Prospects of IoT**

  The future prospects of IoT are extensive and promising, with expected growth across domains such as healthcare, transportation, agriculture, manufacturing, and smart cities. Potential applications range from smart homes to autonomous vehicles and industrial automation, offering opportunities to enhance efficiency,

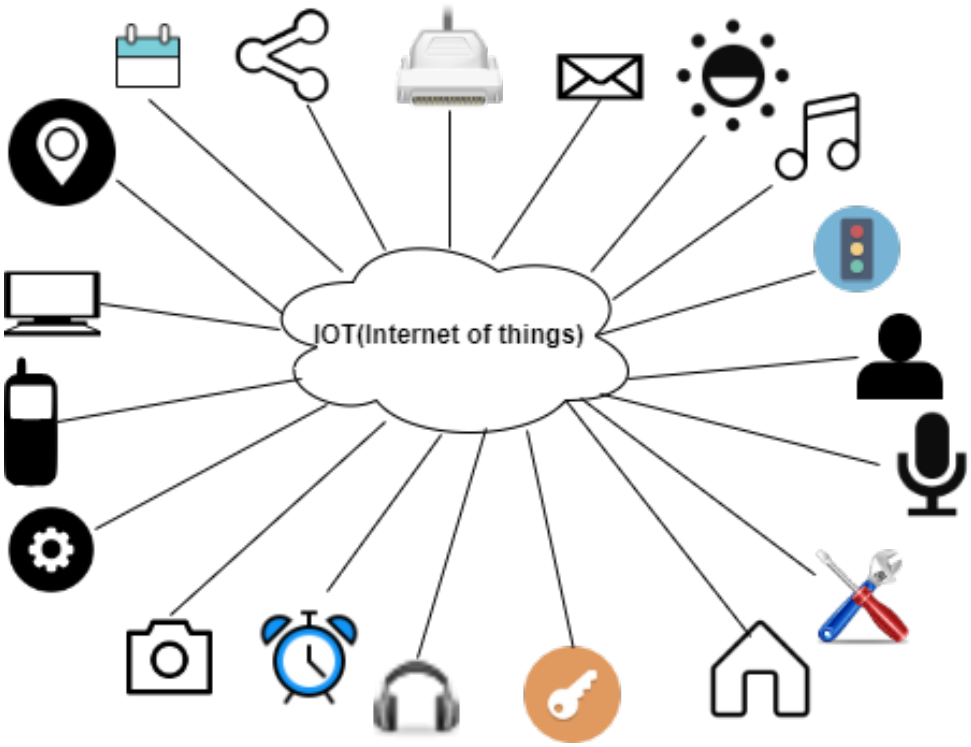

Figure 1.1: Internet of Things

productivity, and overall quality of life.

- **Demands of IoT**

  The demand for IoT stems from the increasing need for connectivity, automation, and data-driven insights. Gathering and analyzing real-time data from devices empower informed decision-making, driving businesses to optimize operations, improve customer experiences, and foster innovation. Consumers expect seamless connectivity and personalized experiences through IoT-enabled devices and applications.

- **Challenges of IoT**

  While IoT presents significant potential, challenges must be addressed for widespread adoption and successful implementation. Interoperability and standardization across devices, communication protocols, and data formats are crucial for seamless integration. Privacy and security concerns arise from the collection of vast amounts of data, requiring robust measures to protect personal information and mitigate cybersecurity risks. Scalability, data management, and power consumption are critical considerations, necessitating efficient infrastructure for storage, process-

ing, and analysis[24]. Power efficiency is vital for devices with limited battery life or relying on energy harvesting techniques.

The Internet of Things holds tremendous promise in transforming industries and improving our daily lives. Addressing challenges such as interoperability, privacy, security, scalability, and power efficiency is essential for successful adoption. By overcoming these challenges, IoT can revolutionize our world, creating interconnected ecosystems that drive productivity, innovation, and an enhanced quality of life.

- **Gateway Devices**

  n the realm of the Internet of Things (IoT), a gateway device plays a crucial role as a bridge between various smart devices and cloud applications. It serves as a central component within the network layer, connecting the smart IoT devices at the bottom layer (perception layer) to diverse applications in the upper application layer. The gateway device takes on essential tasks, including protocol translation between sensors and the internet, as well as providing local data storage.

  The importance of a gateway device in IoT stems from its role in enabling seamless communication and data exchange between IoT devices and the cloud. It acts as a mediator, facilitating the flow of information between the devices and the applications. By translating protocols, the gateway device ensures that different devices with distinct communication protocols can communicate effectively with each other and with the cloud[3].

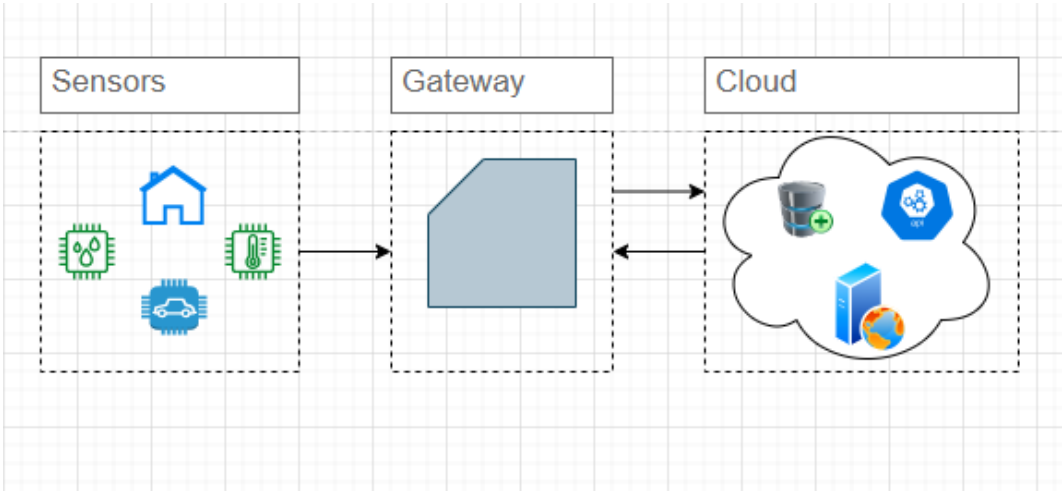

Figure 1.2: Gateway Setup

One of the key functions of a gateway device is to aggregate and filter data collected by connected IoT devices. It collects data from multiple sensors and devices in its vicinity and processes it before transmitting it to the cloud. This data aggregation and preprocessing at the edge help in reducing latency and optimizing network bandwidth utilization. It also enables real-time or near-real-time decision-making by filtering and forwarding only the relevant data to the cloud for further analysis and action[2].

The gateway device also plays a vital role in enabling local data storage and edge computing capabilities. It allows for data to be stored and processed locally, reducing the dependency on cloud resources and improving the overall system efficiency. This is particularly important in scenarios where low latency, offline operations, or intermittent connectivity are critical requirements.

Furthermore, the gateway device provides an additional layer of security by acting as a firewall and implementing security protocols. It helps in protecting the IoT network and devices from potential cybersecurity threats by applying encryption, access control, and authentication mechanisms. The gateway can also facilitate secure remote access and management of IoT devices[10].

Overall, the gateway device holds significant importance in IoT systems as it acts as a central point of connectivity, data aggregation, preprocessing, storage, and security. It enables efficient and reliable communication between IoT devices and the cloud, optimizes network bandwidth, reduces latency, and enhances overall system performance. By serving as a gateway between the edge and the cloud, it empowers IoT systems to gather, process, and utilize data effectively, enabling various applications and services to leverage the power of IoT technology.

1.1.2 Microservice Arcitecture

Microservice architecture has gained significant attention and popularity in the context of the Internet of Things (IoT) due to its ability to address the challenges posed by the distributed and heterogeneous nature of IoT systems. In a microservices architecture, an application is built as a collection of small, loosely coupled services that can be independently developed, deployed, and scaled. Each service focuses on a specific functionality or feature, and they communicate with each

other through well-defined APIs[23][13].

In the context of IoT, microservices offer several benefits. Firstly, they enable modularity and flexibility in IoT system design. Since IoT systems consist of a diverse range of devices, protocols, and data formats, breaking down the system into microservices allows for granular control and management of individual components. Developers can focus on building and maintaining small, specialized services that are better suited to handle the specific requirements of IoT devices and applications.

[3]Secondly, microservices promote scalability and agility. As the number of IoT devices and the volume of data they generate continue to grow rapidly, scalability becomes a crucial aspect. With microservices, scaling can be done selectively based on the demand for specific services, rather than scaling the entire monolithic application. This approach ensures efficient resource utilization and allows for better responsiveness to changing IoT workloads.

Also microservices enable easier integration and interoperability in IoT systems. With well-defined APIs, different microservices can communicate seamlessly and exchange data, regardless of the underlying technologies or protocols used. This interoperability is vital in IoT, where devices and systems from different vendors need to work together to provide cohesive solutions.

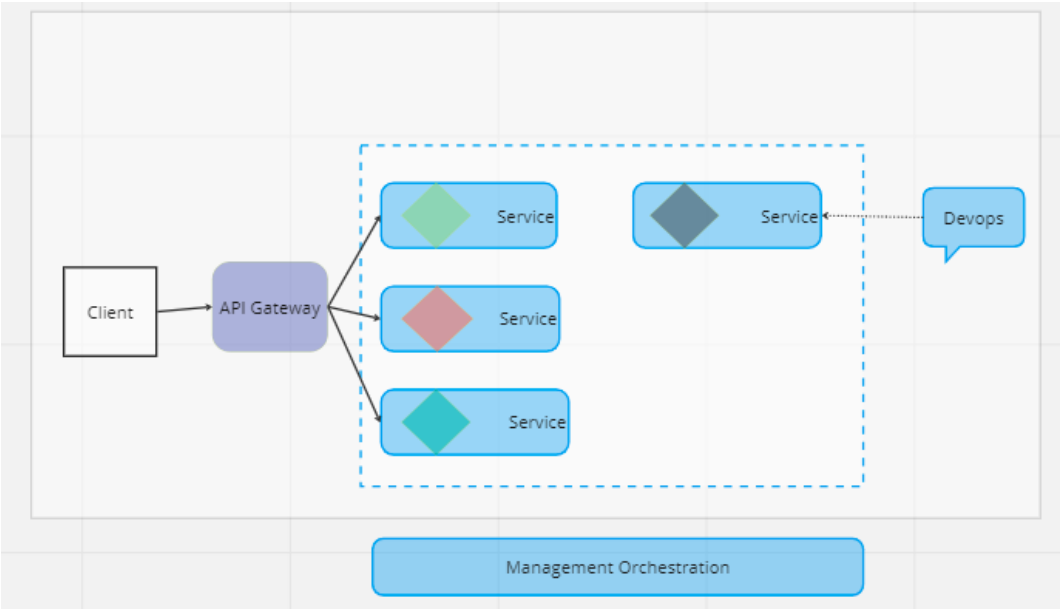

Figure 1.3: Microservice Architecture

Microservices also contribute to fault tolerance and resilience in IoT systems. By isolating services and encapsulating their functionalities, failures or issues in one service do not necessarily impact the entire system. This enables easier fault detection, isolation, and recovery, improving the overall reliability and availability of IoT applications.

However, implementing microservices in IoT comes with its own challenges. It requires careful consideration of issues such as service discovery, data consistency, event-driven communication, security, and deployment orchestration. These challenges need to be addressed to ensure the seamless operation and management of microservices in IoT environments.

Hence microservices architecture offers a promising approach to tackling the complexities and scale of IoT systems. By breaking down applications into smaller, independent services, microservices enable modularity, scalability, flexibility, integration, and fault tolerance. It allows for efficient management of IoT devices and data, facilitating the development of innovative and scalable IoT applications and services.

- **Microservice Architecture for health care IOT system**

  The applications of IoT in healthcare are vast, spanning remote patient monitoring, telemedicine, smart hospitals, medication management, and health behavior tracking. Wearable devices, for instance, can continuously track an individual's heart rate, physical activity, sleep patterns, and other health-related data, enabling personalized recommendations, preventive care strategies, and early interventions that lead to improved patient outcomes and reduced healthcare costs[12].

  However, the adoption of IoT in healthcare also presents notable challenges that must be addressed. These challenges include ensuring the privacy and security of data, managing the substantial volume of data generated by IoT devices, establishing interoperability and standardization across devices and data formats, and addressing ethical and legal considerations associated with IoT-driven healthcare systems.

  To overcome these challenges and unlock the full potential of IoT in healthcare, innovative frameworks and architectures are essential. This thesis focuses on exploring the feasibility and benefits of a containerized microservice framework de-

ployed on edge devices for healthcare IoT systems. By harnessing the capabilities of edge computing, this framework aims to overcome the limitations of traditional centralized architectures, enabling efficient data processing, reduced latency, improved scalability, and robust data privacy and security measures.

The research conducted in this thesis aims to contribute to the development of more efficient, flexible, and secure architectures for healthcare IoT applications. Specifically, it investigates the application of containerization and microservices in healthcare IoT systems, particularly when deployed on edge devices. These advancements have the potential to drive widespread adoption of IoT technologies in healthcare, resulting in enhanced patient care, increased operational efficiency, and the realization of the full transformative potential of IoT in the healthcare landscape.

The proliferation of IoT devices in healthcare has created a tremendous amount of data, emphasizing the critical requirement for efficient and scalable systems to effectively manage this data. Managing the large volume, velocity, and variety of healthcare data generated by IoT devices in a timely and scalable manner has become a significant challenge.

Healthcare IoT devices, including wearable sensors, medical monitoring devices, and smart healthcare equipment, continuously generate diverse data such as vital signs, patient records, diagnostics, and treatment information. Traditional healthcare systems often struggle to handle the dynamic and data-intensive nature of this information, highlighting the need for innovative approaches.

Efficient and scalable systems are essential for managing healthcare data derived from IoT devices. These systems must be capable of processing high data rates and meeting real-time processing demands presented by continuous data streams. Additionally, they should offer robust storage capabilities to accommodate the growing volume of healthcare data.

Scalability is crucial to adapt to the increasing number of IoT devices and the exponential growth of data generation. By adopting scalable systems, healthcare organizations can seamlessly expand their infrastructure to meet the rising demand for IoT devices and effectively manage the corresponding increase in data volume[26].

Efficiency plays a pivotal role in optimizing resource utilization, minimizing latency, and ensuring timely access to critical healthcare data. By implementing efficient data processing algorithms and technologies, healthcare organizations can derive valuable insights from IoT-generated data, enabling quicker decision-making and improved patient outcomes.

In response to the need for efficient and scalable systems in healthcare IoT, novel

approaches like containerized microservice frameworks deployed on edge devices have gained prominence. These frameworks leverage the computing capabilities of edge devices to enable data processing and analysis closer to the data source. By distributing computing tasks to edge devices, latency can be reduced, and bandwidth utilization can be optimized[3].

To summarize, the escalating volume of healthcare data generated by IoT devices necessitates the development of efficient and scalable systems. These systems must be capable of handling high data volumes, ensuring real-time processing, and accommodating the increasing number of IoT devices. Innovative approaches, such as containerized microservice frameworks on edge devices, offer potential solutions to overcome the challenges associated with healthcare IoT data management. By adopting these approaches, healthcare organizations can enhance healthcare delivery, improve patient care, and create more efficient healthcare systems[16].

In recent years, the Internet of Things (IoT) has revolutionized numerous sectors and has a significant impact on our daily lives. The rapid growth in IoT devices has been exponential, with billions already deployed and a projected 50 billion devices expected by 2020. Researchers have explored various applications and future prospects of IoT devices. For example, in the authors emphasized the increasing interest in real-time locating systems and the utilization of radio frequency identification (RFID) for object identification and tracking in response to the surge in IoT devices.

With the continuous expansion of IoT applications, the characteristics of connected devices are evolving. The ubiquitous connectivity and information-sharing capabilities of IoT systems are reshaping industries. This connectivity leads to the generation of massive amounts of data from embedded sensors, requiring efficient data processing and computation power. The authors highlighted the significance of deep computation, data analytics, and machine learning in achieving state-of-the-art performance for feature learning on big data in industrial IoT applications.

As the IoT ecosystem continues to evolve, it is essential to address the challenges associated with processing and analyzing the vast volumes of data generated by IoT devices. Efficient data management, computation, and analytics play a crucial role in harnessing the potential of IoT technologies and leveraging the insights derived from IoT-generated data.

The exponential growth of IoT devices has brought about transformative changes across various sectors. The ability of IoT systems to connect and share information has become pervasive, leading to the generation of substantial data that require efficient processing and computation. Researchers are actively exploring advanced techniques such as deep computation, data analytics, and machine learning to tackle the challenges and unlock the full potential of IoT in industrial and everyday applications. In the healthcare IoT domain, the implementation of microservices architecture offers the opportunity to break down complex healthcare applications into smaller, self-contained services. These microservices are designed to address specific functionalities or domains within the healthcare system, such as patient monitoring, electronic health records (EHR) management, data analytics, and telehealth services. This modular approach facilitates independent development, deployment, and maintenance of each microservice, enabling agile practices and quicker time-to-market for healthcare solutions[24].

Scalability is a crucial requirement in healthcare IoT, given the increasing number of connected medical devices and the growing volume of patient data. Microservices architecture provides a granular scalability capability, allowing individual services to scale independently based on demand. This flexibility ensures efficient resource utilization and the ability to handle diverse workloads across various healthcare applications.

Interoperability is another notable advantage offered by microservices in the healthcare IoT landscape. Through well-defined APIs, microservices can seamlessly communicate and exchange data, regardless of the underlying technologies, devices, or protocols involved. This interoperability facilitates the integration of disparate healthcare systems, devices, and applications, fostering smooth data sharing and collaboration among healthcare providers, patients, and other stakeholders.

Moreover, microservices facilitate the delivery of personalized and patient-centric care in healthcare IoT. By decomposing applications into smaller services, healthcare providers can develop tailored solutions that address specific patient needs. For instance, a microservice focused on remote patient monitoring can collect real-time data from wearable devices and securely transmit it to other microservices responsible for data analysis, anomaly detection, or generating alerts. This personalized approach enables timely interventions and enhances patient outcomes.

Nevertheless, the adoption of microservices in healthcare IoT is not without challenges.

Data security and privacy emerge as critical concerns when dealing with sensitive patient information. It is essential to implement robust encryption, authentication, and access control mechanisms to safeguard patient data across microservices and ensure compliance with privacy regulations. Additionally, effective coordination and communication among microservices must be established through service discovery, event-driven communication, and orchestration mechanisms to guarantee the overall seamless functioning of the healthcare system.

As microservices architecture provides numerous benefits for healthcare IoT by enabling the decomposition of complex applications into smaller, independent services. The modular design facilitates scalability, interoperability, and personalized care delivery. However, addressing challenges related to data security, privacy, and effective coordination is essential for successful implementation in healthcare IoT environments.

1.1.3 DevOps and GitOps

DevOps and GitOps are two methodologies widely employed in software development and deployment to improve collaboration, streamline processes, and enhance the reliability and scalability of software systems.

DevOps, short for Development and Operations, is a cultural and collaborative approach that emphasizes breaking down silos between development and operations teams. It focuses on automating processes, fostering continuous integration and delivery (CI/CD) pipelines, and promoting a shared responsibility for software development and deployment. DevOps practices, including infrastructure as code, continuous monitoring, and iterative feedback loops, enable organizations to deliver software more rapidly and reliably[15].

By embracing DevOps, organizations can reap various benefits. They can reduce time-to-market by implementing automated and frequent releases through CI/CD practices, resulting in shorter feedback cycles and the ability to adapt swiftly to changing customer needs and market demands. DevOps also enhances the stability and dependability of software systems by automating testing, deployment, and monitoring processes. Collaboration and shared ownership among development and operations teams lead to increased transparency and efficiency throughout the software development lifecycle.

GitOps, an extension of the DevOps philosophy, focuses on utilizing Git as a central

repository for managing infrastructure and application configurations. With GitOps, organizations can version control and orchestrate their entire infrastructure and deployment process using Git repositories. Storing infrastructure and application configurations as code allows for versioning, easy rollbacks, and seamless replication across different environments.

A key principle of GitOps is declarative infrastructure and application management. The desired state of the system is defined within Git repositories, and specialized tools such as Kubernetes operators or deployment agents ensure that the actual state aligns with the desired state. This approach simplifies infrastructure management, improves traceability, and enables auditable and reproducible deployments.

Implementing GitOps practices provides several advantages. It promotes consistency, transparency, and reproducibility in software deployments. Changes to infrastructure and application configurations can be tracked, allowing for straightforward rollbacks and facilitating collaboration between development, operations, and security teams. GitOps also enables the automation of deployment processes, reducing manual intervention and minimizing the potential for human errors.

In summary, DevOps and GitOps are complementary methodologies that foster collaboration, automation, and reliability in software development and deployment. DevOps emphasizes breaking down barriers and optimizing processes, while GitOps extends these principles by utilizing Git as a centralized repository for managing infrastructure and application configurations. Embracing these practices empowers organizations to enhance the speed, quality, and stability of their software systems.

1.2 Problem Statement

By the year 2030, the number of IoT devices and sensors is projected to reach around 100 billion[25]. With the increasing adoption of IoT technology, various applications demand high availability, scalability, low latency, and data privacy. The healthcare sector, in particular, faces unique challenges due to the limited processing power of edge gateway devices and concerns about data privacy. Therefore, there is a pressing need for a versatile framework for edge devices (gateway devices) that can effectively address these requirements and be applicable across different IoT applications.

The management and processing of healthcare IoT data pose significant challenges and limitations for traditional centralized architectures. These limitations can have a substantial impact on the efficiency and effectiveness of healthcare IoT systems. The following key challenges and limitations are observed[29]:

- Latency: Traditional centralized architectures introduce latency as data is sent to remote servers for processing and analysis. This delay in accessing and processing healthcare IoT data can hinder real-time decision-making and patient care, which is critical in the healthcare domain.
- Bandwidth Constraints: The continuous data generation by healthcare IoT devices strains network bandwidth during data transmission to centralized servers. Limited bandwidth can lead to data congestion and slower transmission, negatively affecting the overall system performance and responsiveness.
- Scalability: Traditional centralized architectures may face challenges in effectively scaling healthcare IoT systems to accommodate the increasing number of devices and the resulting growth in data volume. Significant infrastructure upgrades and resource allocation may be required to meet the expanding demands of healthcare IoT systems.
- Data Privacy and Security: The sensitive and private nature of healthcare IoT data raises concerns about data privacy and security when transmitting it to centralized servers. Ensuring robust data encryption, access control, and compliance with privacy regulations can be challenging for traditional architectures.
- Reliability and Fault Tolerance: Dependence on a single centralized server introduces a single point of failure. Any issues or downtime experienced by the server can disrupt the entire healthcare IoT system, impacting data collection, analysis, and decision-making processes.
- Edge Computing Advantages: The adoption of a containerized microservice framework on edge devices offers advantages over traditional centralized architectures. Edge devices enable local data processing, reducing the need for constant data transmission to remote servers. This approach minimizes latency, optimizes bandwidth usage, and facilitates real-time data analysis at the edge, enhancing system responsiveness and efficiency[13].

By implementing a containerized microservice framework on edge devices, healthcare IoT systems can overcome these challenges and limitations. This framework harnesses the computing capabilities of edge devices, allowing for distributed data processing, improved scalability, reduced latency, and enhanced data privacy. Containerization facilitates the deployment of lightweight and modular microservices, ensuring efficient resource utilization and seamless scalability to meet the specific requirements of healthcare IoT systems[8].

1.3 Research Objective

The following research objectives aim to explore the feasibility, benefits, and challenges of utilizing containerized microservice frameworks on edge devices in healthcare IoT systems. The research seeks to contribute to the understanding of how such frameworks can address the limitations of traditional centralized architectures and provide efficient, scalable, secure, and privacy-conscious solutions for managing and processing healthcare IoT data[22] [24].

- Assess the viability and effectiveness of implementing a containerized microservice framework on edge devices for healthcare IoT systems.
- Investigate the impact of edge device utilization on healthcare IoT data processing, including latency reduction and enhanced system responsiveness.
- Evaluate the scalability of containerized microservice frameworks on edge devices to accommodate the increasing number of IoT devices and data volume in healthcare environments. Examine the security and privacy implications associated with deploying containerized microservice frameworks on edge devices for managing healthcare IoT data.
- Compare the performance and resource utilization of containerized microservice frameworks with traditional centralized architectures in healthcare IoT systems.
- Analyze the advantages and limitations of containerized microservice frameworks on edge devices for healthcare IoT applications, such as improved data privacy, reliability, and fault tolerance.

- Develop guidelines and best practices for the implementation of containerized microservice frameworks on edge devices in healthcare IoT systems, considering deployment strategies, orchestration mechanisms, and system monitoring.

1.4 Limitations

The deployment of containerized microservice frameworks on edge devices for healthcare IoT systems faces several limitations. These include the resource constraints of edge devices, connectivity challenges, device heterogeneity, security risks, management complexity, limited remote control and maintenance, and integration with existing systems. Addressing these limitations is crucial to ensure the successful implementation and maximize the benefits of this approach in healthcare IoT applications[19].

- Edge Device Resource Constraints: Edge devices typically have limited computing resources such as processing power, memory, and storage capacity. This may pose limitations on the complexity and scale of the containerized microservice framework that can be deployed on these devices. Resource-intensive applications or scenarios with a high number of concurrent microservices may exceed the capabilities of edge devices.

- Connectivity Challenges: Edge devices may face connectivity challenges, especially in remote or rural areas with limited network coverage. In such scenarios, maintaining consistent and reliable connectivity for real-time data processing and communication with the central system can be problematic. These connectivity limitations can affect the overall performance and responsiveness of the healthcare IoT system.

- Edge Device Heterogeneity: The landscape of edge devices used in healthcare IoT systems can be highly heterogeneous. Different devices may have varying capabilities, operating systems, or compatibility with containerization technologies. This heterogeneity can introduce complexities in deploying and managing a standardized containerized microservice framework across diverse edge devices.

- Security Risks: Edge devices deployed in healthcare IoT systems may be susceptible to security risks and vulnerabilities. These devices are often deployed in open

and uncontrolled environments, making them potential targets for attacks. Ensuring the security of the containerized microservice framework on edge devices becomes crucial to protect sensitive healthcare data and maintain the integrity of the system.

- Management and Orchestration Complexity: Managing and orchestrating containerized microservices on edge devices can be complex, particularly when dealing with a large number of devices and microservices. Efficiently deploying, updating, and monitoring the microservices across distributed edge devices require robust management and orchestration mechanisms. The complexity increases as the system scales and evolves.

- Limited Remote Control and Maintenance: Edge devices are often located in remote or physically inaccessible locations, making remote control and maintenance challenging. Troubleshooting, updating, and managing the containerized microservice framework on edge devices may require physical access or involve additional logistical considerations, which can impact the efficiency of system maintenance and updates.

- Integration with Existing Systems: Integrating a containerized microservice framework on edge devices into existing healthcare IT infrastructure can present challenges. Interoperability and compatibility with legacy systems, data formats, and protocols need to be addressed to ensure seamless integration and data exchange between the edge devices and the centralized healthcare system[14].

Chapter 2

Literature Review

Researchers have actively explored the development of containerized microservice frameworks on edge devices for IoT systems. This comprehensive literature review provides an in-depth analysis of the latest research advancements, methodologies, and challenges in this field. The literature review delves into the various aspects of containerized microservice frameworks for healthcare IoT systems on edge devices. It explores the fundamental principles of architectural design, deployment models, and communication protocols relevant to this domain. Furthermore, it examines the advantages and limitations of these frameworks, considering factors such as resource constraints, security considerations, interoperability, and real-time data processing capabilities.

To assess the effectiveness of containerized microservice frameworks, it investigates the evaluation methodologies and metrics employed by researchers. It critically evaluates the performance, scalability, reliability, and usability of different frameworks, highlighting their respective strengths and weaknesses.

In [9], the author presents an innovative system for home-based healthcare using IoT, Fog computing, and cloud computing technologies. The proposed system architecture incorporates both local and remote monitoring functionalities by utilizing sensing units and a mobile application acting as the Fog server. This design allows for real-time analysis of environmental aspects within the patient's room, while vital signs are captured by the sensors. The integration of IoT devices and Fog computing enables efficient data processing and analysis, leading to timely interventions and improved healthcare decision-making.

A notable contribution of the system is its utilization of Fog computing, which brings

computation, storage, and networking capabilities closer to the edge devices. By offloading data processing tasks to Fog nodes, the system reduces latency and enhances responsiveness, making it suitable for time-sensitive healthcare applications[9]. To address the challenges posed by data heterogeneity, the system employs a NoSQL database for data persistence. This choice enables efficient storage and retrieval of diverse healthcare data, accommodating the complex nature of such information and ensuring scalability and flexibility. Additionally, the system provides access to the collected data through a web server REST API, allowing nurses and medical staff to retrieve patient information and make informed decisions. This feature facilitates effective communication and collaboration among healthcare professionals, contributing to improved patient care.

The technique highlights the significance of the proposed home hospitalization system in enhancing healthcare services through the integration of IoT, Fog computing, and cloud computing technologies. The real-time monitoring, data analysis, and timely interventions enabled by this system hold great potential for improving the quality of home-based healthcare. This research contributes to the existing knowledge in the field of IoT-enabled healthcare systems. The integration of Fog computing and cloud computing with IoT devices offers promising opportunities for the development of efficient and reliable home hospitalization solutions. Continued research and development in this area can further advance the capabilities of such systems, leading to better patient outcomes and a more sustainable healthcare ecosystem. It has several limitations. These include the lack of cost-effectiveness, as the financial implications are not discussed. The system also fails to address the incorporation of existing sensors and legacy systems, hindering interoperability. Additionally, assumptions about stable WiFi connections and elderly users' ability to handle smart devices may not reflect real-world scenarios. Lastly, the absence of a microservice architecture limits the system's modularity and adaptability. Addressing these limitations would improve the practicality and effectiveness of the proposed system. In [18], The author proposes a platform for remote patient monitoring in intensive care units (ICUs) using IoT technology. The methodology employed in this study involves the integration of sensors to capture physiological and environmental data, analysis of the collected data to detect anomalies, and the use of a smartphone application and a web interface for data visualization.

One advantage of the methodology is its ability to provide efficient and centralized

monitoring of ICU patients, enabling healthcare providers to remotely monitor multiple patients simultaneously. The use of IoT sensors and data analysis techniques allows for early detection of anomalies and timely alarms, enhancing patient safety and facilitating proactive medical interventions. Additionally, the integration of the standard multi-parameter monitor using MQTT gateway technology enables seamless data transmission to the cloud, ensuring data availability and accessibility.

However, there are limitations to consider. The paper does not extensively discuss the implementation challenges or technical issues encountered during the deployment of the proposed platform. The scalability and interoperability of the system with existing healthcare infrastructure and devices may also require further investigation. Additionally, the paper does not provide detailed insights into the data privacy and security measures implemented to protect patient information.

Further research and evaluation are necessary to assess the platform's effectiveness, scalability, and reliability in real-world ICU environments. Future work should also consider addressing the challenges related to data integration, system performance, and ensuring compliance with privacy and security regulations in healthcare settings.

In [21], the author adopts a methodology that involves designing and implementing a three-layer system architecture for healthcare IoT applications. The technique includes the development of an edge gateway responsible for device management, cloud connection, data storage, data preprocessing, and security. The wireless router is utilized to extend the range of wireless sensor nodes and enable efficient edge task execution. The paper also presents three healthcare case studies to demonstrate the reliability and effectiveness of the proposed platform in different scenarios.

This lie in its comprehensive approach to addressing the specific requirements of healthcare IoT applications. By incorporating edge computing capabilities, the system enables real-time data processing, reducing latency and enhancing responsiveness. The use of a three-layer architecture allows for distributed data management, ensuring efficient utilization of resources and improved scalability. Additionally, the inclusion of wireless routers extends the communication range, enabling seamless connectivity and data collection from remote sensor nodes.

It has some limitations too as the paper does not fully explore the potential of microservices, which could hinder scalability and modularity. Additionally, the absence

of a detailed discussion on the implementation of a firewall at the gateway level raises concerns about data security. Furthermore, the ease of deploying containerized microservices, which could enhance system flexibility and management, is not addressed. In future, it should focus on leveraging microservices to maximize system scalability and adaptability. Implementing robust security measures, such as firewalls, at the edge gateway level is essential to protect sensitive healthcare data. Additionally, considering the deployment of containerized microservices can improve the system's agility and ease of management. Addressing these limitations will contribute to the advancement and practical implementation of edge-based hybrid systems for long-range safety and healthcare IoT applications.

In [11], the author presents a framework to enhance the security and efficiency of edge computing in healthcare systems using Software-Defined Networking (SDN) technology.

The paper focuses on addressing the challenges of securing IoT-enabled healthcare systems by leveraging edge computing capabilities. It proposes an architecture that combines SDN principles with edge computing to establish a secure and scalable infrastructure. The framework incorporates features such as network slicing, encryption, authentication, and access control to protect sensitive healthcare data.

The study highlights the advantages of using SDN-based edge computing in healthcare, including improved data processing efficiency, reduced latency, enhanced security, and better resource utilization. It also discusses the practical implementation and evaluation of the proposed framework, demonstrating its effectiveness in meeting the security and performance requirements of healthcare IoT applications.

Overall, it contributes to the field of IoT-enabled healthcare systems by providing a comprehensive framework that addresses the security challenges associated with edge computing. It offers valuable insights and recommendations for designing secure and efficient healthcare infrastructures, paving the way for the adoption of IoT technologies in healthcare with enhanced security measures.[11] The research paper proposes a secured framework for SDN-based edge computing in IoT-enabled healthcare systems. The framework consists of three layers: the infrastructure layer, edge computing layer, and core computing layer. The infrastructure layer includes low-poared embedded sensors and IoT devices, while the edge computing layer comprises different edge servers for data exchange, storage, processing, and job migration. The core computing layer

involves core networks and cloud services responsible for hosting applications and managing the IoT architecture. The paper presents a lightweight authentication approach, SDN-based collaborative edge computing, and job migration on edge servers as part of the proposed framework. However, limitations of the research include the lack of consideration for containerization and microservices and potential security vulnerabilities such as denial-of-service attacks[11].

In [20], The proposed methodology presents an edge-IoT framework for smart healthcare applications based on blockchain technology. The framework involves generating an intrusion detection dataset in the edge network where IoMT data is processed. Preprocessing techniques are applied to handle missing and noisy data, and Principal Component Analysis (PCA) is used for feature selection. The Deep Neural Network (DNN) is trained using the generated dataset, and a Crow Search algorithm is employed for hyperparameter tuning. The system architecture is designed to enhance the reliability of IoMT devices and improve intrusion detection capabilities. The methodology also discusses user configuration validation and the mathematical representation of intrusion detection in the IoMT environment. The article highlights the importance of designing and developing secure IoMT ecosystems and addresses various algorithmic approaches for streamlining IoT and IoMT devices.

In another research article [28], explores the integration of fog computing into healthcare Internet of Things (IoT) applications. It emphasizes the significance of addressing performance, security, and offloading challenges to enhance the efficiency and effectiveness of healthcare systems.

The discussed proposals aim to improve performance by reducing service response time, optimizing energy consumption, and enhancing decision-making accuracy. Several studies demonstrate considerable enhancements in these areas compared to existing solutions. Privacy and security are also crucial considerations in healthcare IoT applications, with researchers proposing frameworks and architectures to safeguard patient information.

Efficient offloading strategies for fog servers are another focal point, given the exponential growth of IoT devices and data. These proposals strive to minimize overload on fog computing servers by optimizing task assignments. The survey results emphasize the role of fog computing in real-time healthcare applications such as monitoring hypertension attacks, diagnosing diseases, and supporting COVID-19 treatment.

While significant advancements have been made, the paper acknowledges the remaining challenges in fog-IoT applications. These challenges include refining offloading schemes, developing efficient scheduling algorithms, and ensuring privacy and security in fog servers. As the integration of fog computing and IoT applications is still in its early stages, further research is necessary to overcome these obstacles.

This concludes with an overview of the current state of fog computing in healthcare IoT applications, highlighting the proposed performance, security, and offloading approaches by various researchers. The survey results demonstrate the potential of fog computing in improving healthcare systems while underscoring the need for further advancements to tackle the existing challenges. In [5], Author proposed the development of a secure edge computing framework for smart healthcare surveillance. The framework aims to enhance the security and privacy of healthcare data while enabling real-time monitoring and analysis of patient information. It leverages edge computing technology to process data locally at the edge devices, reducing latency and network congestion. The proposed framework incorporates various security mechanisms, including encryption, access control, and anomaly detection, to safeguard sensitive healthcare data. The authors conducted experiments to evaluate the performance and effectiveness of the framework, demonstrating its ability to provide secure and efficient healthcare surveillance. the research paper has a few limitations that should be considered. Firstly, the framework's evaluation is based on experiments conducted in a controlled environment, which may not fully reflect real-world conditions and challenges. Further studies and field trials are necessary to assess the framework's scalability and robustness in diverse healthcare settings. Secondly, while the paper discusses security mechanisms, it does not delve deeply into the potential vulnerabilities and attack vectors that the framework may face. A more comprehensive analysis of security threats and countermeasures would enhance the paper's insights and provide a clearer understanding of the framework's resilience.

The methodology focuses primarily on the technical aspects of the surveillance framework, neglecting potential ethical and legal considerations associated with smart healthcare surveillance. It would be beneficial to address the ethical implications and regulatory requirements, such as data privacy and consent, to ensure the framework aligns with relevant standards and guidelines.

In [27], A Reinforcement Learning-Based Approach" explores the utilization of con-

tainerization and reinforcement learning techniques to optimize the deployment of micro-services in fog computing environments. The main objective is to address the resource allocation and task scheduling challenges in fog devices, aiming to enhance the overall performance and scalability of fog computing systems. The authors begin by emphasizing the increasing demand for fog computing, which extends cloud computing capabilities to the edge of the network, close to the data source. This approach offers advantages such as reduced latency, improved data privacy, and enhanced real-time decision-making. However, deploying micro-services in fog devices presents challenges due to limited resources, diverse hardware configurations, and dynamic workloads.

To tackle these challenges, the authors propose an approach that leverages containerization and technologies like Docker to encapsulate micro-services and their dependencies in lightweight and portable containers. By utilizing containers, fog devices can effectively manage resource allocation, isolation, and scalability. Moreover, the paper introduces a reinforcement learning-based approach to optimize micro-service deployment in fog devices. Through reinforcement learning algorithms such as Q-learning or Deep Q-networks, the system learns from interactions with the fog environment, enabling intelligent decision-making regarding resource allocation and task scheduling. The authors provide detailed insights into the design and implementation of the reinforcement learning model, highlighting its adaptability to dynamic fog computing environments and its potential to enhance micro-service performance.

The proposed approach is evaluated through simulations and experiments, demonstrating its effectiveness in optimizing resource allocation and task scheduling. The results showcase improvements in service latency, energy consumption reduction, and scalability compared to traditional approaches. This research paper makes a significant contribution to the field of fog computing by addressing the challenges of micro-service deployment in fog devices. The combination of containerization and reinforcement learning in the proposed approach offers a promising solution for optimizing resource allocation and improving the performance of fog computing systems.

However, it is important to acknowledge certain limitations in this study. The paper primarily focuses on the technical aspects of the proposed approach and does not extensively discuss potential trade-offs or challenges associated with implementing containerization and reinforcement learning in fog devices. A more comprehensive analysis

of the limitations and constraints of these technologies within fog computing scenarios would enhance understanding of their practical implications.

Additionally, the evaluation of the proposed approach relies primarily on simulations and experiments, which may not fully capture the complexities and dynamics of real-world fog environments. Future work should consider conducting field trials or real-world deployments to validate the effectiveness and scalability of the approach across diverse fog computing scenarios.

Despite these limitations, this research paper presents a valuable contribution by introducing a containerization-based approach combined with reinforcement learning to optimize micro-service deployment in fog devices. The findings provide insights into the potential benefits of utilizing these technologies to enhance the performance and scalability of fog computing systems. Further research is necessary to address the identified limitations and explore additional optimization techniques for micro-service deployments in fog environments.

Chapter 3

Design and Methodology

The Internet of Things (IoT) has revolutionized the way we interact with our surroundings, enabling seamless connectivity and communication between devices. At the heart of the IoT ecosystem lies the gateway device, which acts as a bridge between the local network and the wider internet. Gateway devices play a crucial role in collecting, processing, and transmitting data from connected devices, making them an essential component in IoT deployments. In this chapter, we will explore different frameworks for gateway devices in the IoT domain and discuss their effectiveness in various types of applications.

- Overview of Gateway Devices

  Gateway devices serve as the intermediary between the local IoT network and external networks, such as the cloud or the internet. They provide connectivity, security, and data management functionalities, enabling efficient data flow between devices and backend systems. Gateway devices are responsible for protocol translation, data filtering, data aggregation, and often include local processing capabilities to perform edge computing tasks.

- Importance of Frameworks in IoT Gateway Devices

  Frameworks for gateway devices provide a structured approach to develop, deploy, and manage these crucial components in IoT systems. A framework offers a set of tools, libraries, and predefined modules that simplify gateway development, accelerate time-to-market, and enhance overall system reliability. Different frameworks cater to diverse application requirements and development preferences, each

offering unique features and capabilities.

In the previous chapter, we examined various proposed architectures for IoT gateways, and it is evident that many of them are based on the monolithic gateway architecture. However, the monolithic architecture inherits several shortcomings that can impact the scalability, flexibility, reliability, modularity, performance, and security of IoT gateways.

- Scalability: Monolithic architectures are not easily scalable. As the number of devices and the volume of data increase, the monolithic gateway may struggle to handle the growing load. The lack of scalability can lead to performance degradation, increased response times, and an inability to efficiently process and manage the influx of data. Additionally, adding or removing functionalities without affecting the entire system becomes challenging in a monolithic architecture.
- Flexibility: Monolithic gateways tend to be rigid and inflexible. Introducing new features or integrating with different protocols and technologies can be complicated and time-consuming. Modifying one component often requires modifications to the entire system, making it difficult to adapt to changing requirements or emerging technologies. The lack of flexibility can hinder the agility and adaptability of IoT gateways.
- Single Point of Failure: In a monolithic architecture, if one component fails, it can bring down the entire gateway. This single point of failure increases the risk of downtime and disruption of services. Troubleshooting and resolving issues become challenging since isolating a specific component without affecting the rest of the system is difficult. The dependency on a single monolithic gateway poses a significant risk to the overall reliability and availability of the IoT system.
- Limited Modularity: Monolithic architectures lack modularity, making it difficult to reuse or replace individual components. Upgrading or replacing a specific functionality within the gateway becomes complex, as it often requires modifications to the entire system. The absence of modularity inhibits the ability to leverage advancements in specific areas or incorporate specialized functionalities without extensive reconfiguration or redevelopment efforts.
- Performance Bottlenecks: In monolithic gateways, performance bottlenecks can occur due to the centralized nature of the architecture. The processing and han-

dling of data from multiple devices may become a bottleneck, leading to delays and decreased overall system performance. The lack of distributed processing capabilities can limit the gateway's ability to scale with increasing data volumes and computational demands.

- Security Risks: Monolithic architectures pose security risks since a vulnerability in one component can potentially expose the entire system. The centralized nature of the architecture makes it challenging to implement granular security measures and isolate potential threats. A breach in one component can compromise the entire gateway, putting the IoT ecosystem at risk. The limited ability to implement specialized security mechanisms for different components or devices further increases the vulnerability of the system.

To overcome these limitations, alternative architectural approaches, such as microservices-based architectures and distributed gateway frameworks, have gained prominence in the IoT domain. These alternative approaches offer improved scalability, flexibility, modularity, fault tolerance, and security by decoupling functionalities into smaller, independently deployable units. By adopting these architectures, IoT gateways can address the shortcomings of the monolithic approach and enhance the overall performance, reliability, and security of the system.

To overcome the limitations of monolithic gateways and address the issue of vendor lock-in, a containerized service architecture can be adopted. This architecture leverages containerization technology, such as Docker, to encapsulate individual services or functionalities into independent containers. Each container runs as a lightweight, isolated unit that can be easily deployed, scaled, and managed.

In a containerized service architecture for IoT gateways, different functionalities or services, such as data ingestion, protocol adaptation, data processing, and communication with cloud platforms, are decoupled and packaged as individual containers. These containers can be developed, tested, and deployed independently, allowing for greater flexibility and modularity in the gateway system.

The use of containerization provides several benefits for IoT gateways. Firstly, it enables vendor-agnosticism, as containers abstract the underlying hardware and operating systems. This means that components from different vendors can be encapsulated as containers, allowing for easy replacement or upgrade without impacting the entire sys-

tem. Organizations can freely choose hardware, software, and services based on their requirements, reducing dependency on specific vendors and promoting a competitive market environment.

Secondly, containerization enhances scalability and agility. With container orchestration platforms like Kubernetes, the deployment and management of containers can be automated, allowing for dynamic scaling and efficient resource utilization. This elasticity ensures that the gateway can handle increasing data volumes and traffic demands without compromising performance or stability.

Additionally, containerization facilitates the development of microservices-based architectures, where each container represents a specific microservice responsible for a particular functionality. This modular approach simplifies development, testing, and maintenance, as individual services can be updated or replaced without impacting the entire system. It also promotes reusability, as containers can be easily shared and deployed across different gateways, enabling organizations to leverage existing solutions and accelerate development.

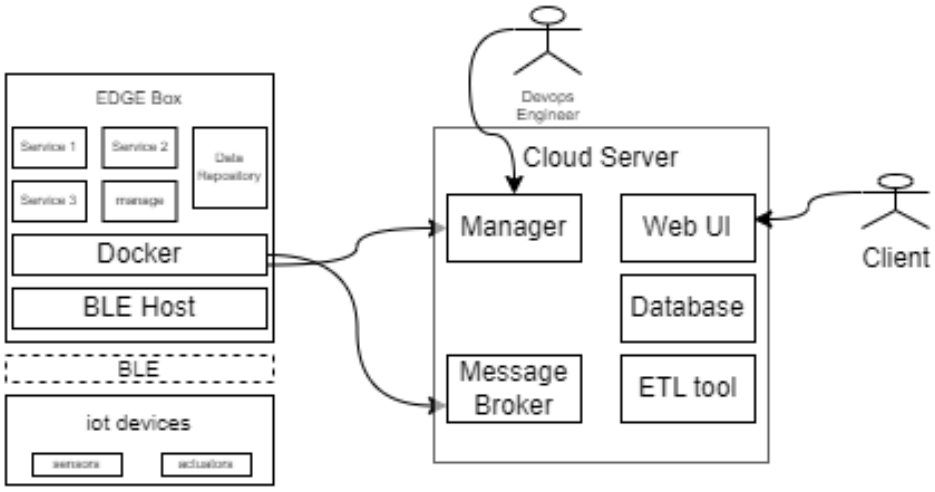

Figure 3.1: Flowchart

Furthermore, containerization improves security by isolating services within their own containers. This isolation prevents vulnerabilities in one service from affecting the entire system, reducing the attack surface and enhancing overall system resilience.

By adopting a containerized service architecture for IoT gateways, organizations can overcome the limitations of monolithic architectures, mitigate vendor lock-in risks, and gain flexibility, scalability, modularity, and enhanced security. This architectural approach enables the seamless integration of diverse components and technologies, promotes innovation, and fosters a more competitive ecosystem in the IoT domain.

Next, we will explore different frameworks and approaches that leverage these alternative architectural paradigms to overcome the limitations of monolithic gateway architectures in IoT deployments.

3.0.1 Underlying Hardware Components

Hardware components are integral elements within the architecture of Internet of Things (IoT) systems, specifically when considering IoT gateways. These components serve as the foundation by providing the necessary computational power, connectivity options, and physical interfaces essential for facilitating seamless communication among IoT devices, edge networks, and the cloud. With a range of components, including microcontrollers, microprocessors, wireless modules, sensors, and actuators, each plays a crucial role in collecting, processing, and transmitting data, effectively bridging the gap between the digital and physical domains. This chapter aims to explore the key hardware components commonly utilized in IoT gateways, outlining their functionalities and assessing their impact on overall system performance and capabilities. By comprehending the roles and characteristics of these hardware components, stakeholders can make informed decisions during the design, deployment, and optimization phases of IoT gateways, leading to successful IoT implementations.

- **Sensor Nodes**

  Sensor nodes are fundamental components in the field of IoT, playing a crucial role in collecting and transmitting data from the physical environment to the digital realm. These nodes are equipped with various sensors that enable them to detect and measure different environmental parameters such as temperature, humidity, light intensity, motion, and more. Sensor nodes are typically compact and power-efficient, making them suitable for deployment in large numbers across diverse environments.

  The primary function of a sensor node is to capture data from the surrounding

environment. The sensors integrated into the node allow it to gather information about specific physical phenomena. This data can be in the form of analog signals, which are then converted into digital signals for processing and transmission. Sensor nodes are designed to operate with minimal power consumption to ensure long battery life and extended periods of operation, making them well-suited for applications where frequent maintenance or power supply is not feasible.

Sensor nodes also include additional components such as microcontrollers, wireless communication modules, and memory storage, enabling them to process, transmit, and store the collected data. The microcontroller acts as the brain of the node, executing tasks and coordinating sensor operations. The wireless communication module enables the transmission of data to other devices or central servers, facilitating real-time monitoring and analysis. Memory storage allows for the temporary or permanent storage of data, depending on the application requirements.Sensor nodes serve as vital components in the IoT ecosystem, enabling the collection and transmission of data from the physical environment. They consist of sensors, microcontrollers, wireless communication modules, and memory storage, all working together to capture, process, and transmit data efficiently. With their compact size, low power consumption, and diverse sensing capabilities, sensor nodes are essential building blocks for various applications, ranging from environmental monitoring and smart cities to healthcare and industrial automation.

- **Edge Box**

  The Xiao BLE Sense is a highly adaptable sensor node that leverages Bluetooth Low Energy (BLE) technology for wireless data collection and transmission. Its compact size and versatile features make it an excellent choice for IoT and wearable applications. Equipped with a microcontroller, BLE module, and a variety of sensors, this device offers a comprehensive solution for real-time data sensing. Despite its small form factor, the Xiao BLE Sense delivers powerful performance and is easily integrated into space-constrained environments or portable devices.

  One of the notable advantages of the Xiao BLE Sense is its low power consumption, which is a result of utilizing BLE technology. This ensures prolonged battery life, making it suitable for applications requiring long-term operation without frequent recharging or battery replacements. Additionally, the device boasts a range of cus-

tomizable features that can be tailored to specific application requirements. With the ability to program the microcontroller with custom firmware, developers can optimize the device's performance and execute specific algorithms for personalized data processing and analysis.

This flexibility enables the Xiao BLE Sense to adapt to a wide array of use cases and deliver tailored solutions to meet diverse application needs.The Xiao BLE Sense stands out as a compact yet powerful sensor node that utilizes BLE technology for wireless data collection. Its small size, low power consumption, and customizable features make it well-suited for IoT and wearable applications, providing seamless integration and reliable data transmission. With its diverse sensor capabilities and adaptability to different use cases, the Xiao BLE Sense offers a versatile solution for real-time data sensing in IoT ecosystems.

- **Embedded Board**

  [17] The Raspberry Pi 4 (Rpi 4) is an affordable and high-performance embedded board widely used as a gateway in the IoT domain. It serves as a crucial component in connecting IoT devices to the internet, facilitating seamless data transmission and efficient management within IoT ecosystems. With its robust processing capabilities, multiple connectivity options, and GPIO pins for sensor integration, the Rpi 4 offers a comprehensive solution for building reliable and cost-effective IoT gateways.

  The Rpi 4 stands out for its powerful processing capabilities, enabling it to handle complex tasks and perform real-time data processing. This allows for efficient communication and interaction between IoT devices and the cloud. Moreover, the Rpi 4 provides various connectivity options, including Ethernet, Wi-Fi, and Bluetooth, ensuring secure and reliable connections to the internet. This versatility enables the gateway to be deployed in diverse networking environments, catering to different connectivity requirements.

  In addition to its processing and connectivity features, the Rpi 4 offers GPIO pins that enable integration with a wide range of sensors and actuators. This allows the gateway to collect data from the physical world and seamlessly incorporate it into IoT applications. The flexibility provided by the GPIO pins enhances the overall functionality and adaptability of the system, making the Rpi 4 a valuable

choice for IoT projects.

Overall, the Raspberry Pi 4 serves as a powerful and cost-effective solution for IoT gateways. Its processing capabilities, connectivity options, and support for sensor integration make it a versatile platform for enabling secure and efficient data transmission within IoT ecosystems[17]. By utilizing the Rpi 4 as an embedded board, IoT developers can build scalable and reliable IoT systems tailored to the specific requirements of various applications

3.0.2 Underlying Software Component

Software components play a vital role in the architecture of Internet of Things (IoT) systems, enabling the management, analysis, and utilization of data collected from IoT devices. These components encompass a wide range of functionalities, including data processing, communication protocols, security mechanisms, and application development frameworks. By leveraging software components effectively, IoT systems can unlock the full potential of the collected data, enabling actionable insights, automation, and seamless integration with various applications and services.

- **Docker**

  Docker is a container engine that revolutionizes the way applications are deployed and managed. It simplifies the process by allowing developers to package their applications and their dependencies into lightweight, isolated containers. These containers encapsulate everything needed for the application to run, ensuring consistency and eliminating compatibility issues across different systems. This consistency and portability make Docker ideal for deploying applications in various environments, from development to production[4].

  One of the key advantages of Docker is its efficient resource utilization and scalability. By using containers, Docker minimizes resource overhead and allows applications to run efficiently, even in resource-constrained environments. Additionally, Docker's ability to scale applications horizontally by running multiple containers of the same application provides flexibility and enables efficient utilization of resources. This scalability is particularly beneficial for microservices architectures, where applications are divided into smaller, independent services that can be individually scaled up or down as needed.

With Docker, developers can streamline the application deployment process. They can package applications and their dependencies into Docker containers, which can be easily distributed and deployed across different systems. This simplifies the deployment process and improves productivity, as developers can focus on writing code rather than dealing with complex deployment configurations. Overall, Docker empowers developers to build, distribute, and deploy applications faster, enhancing productivity and efficiency in the development and deployment lifecycle[4].

3.0.3 Data Management and Data Format

Data management and data format play crucial roles in the realm of information systems, including the Internet of Things (IoT). Data management involves the processes and technologies used to efficiently acquire, store, organize, secure, and retrieve data. It encompasses activities such as data collection, integration, storage, processing, and analysis. Effective data management ensures that data is accessible, reliable, and accurate, enabling organizations to make informed decisions and derive valuable insights from their IoT systems.

Data format refers to the structure and organization of data, determining how it is stored, transmitted, and interpreted. It encompasses the representation, encoding, and decoding of data elements and their relationships. In the context of IoT, data format plays a critical role in ensuring interoperability and seamless communication between different devices, systems, and applications. Standardized data formats such as JSON (JavaScript Object Notation) or XML (eXtensible Markup Language) enable data to be exchanged and interpreted consistently across different platforms and technologies.

By effectively managing and organizing data, organizations can derive actionable insights, identify patterns, and make informed decisions based on the data collected from IoT devices. Furthermore, utilizing standardized data formats promotes interoperability and enables the integration of diverse IoT systems, allowing data to be shared and utilized seamlessly across different domains. Effective data management and the use of appropriate data formats are essential for unlocking the full potential of IoT systems and harnessing the value of the vast amount of data generated by IoT devices.

- **Couchbase**

  Couchbase is a highly versatile distributed NoSQL database platform designed to

meet the demands of modern applications. It combines the flexibility of JSON (JavaScript Object Notation) with the scalability and performance required in today's data-intensive environments. At its core, Couchbase utilizes a document-oriented data model, allowing developers to store and retrieve data flexibly and schema-less. This approach enables seamless adaptation to evolving application requirements, making it ideal for agile development processes. One of Couchbase's key strengths is its ability to support flexible querying. It provides a powerful query language called N1QL (pronounced as "nickel"), allowing developers to perform rich, ad-hoc queries on JSON documents. This query language provides a familiar SQL-like syntax, making it accessible to a wide range of developers. Additionally, Couchbase incorporates various performance optimization techniques such as indexing and caching to ensure efficient and low-latency data access[30].

In terms of scalability and availability, Couchbase offers robust features like replication, auto-sharding, and built-in caching. These features allow the platform to seamlessly distribute data across multiple nodes, ensuring high availability and scalability as data volumes and user loads increase. Furthermore, Couchbase's built-in caching mechanism accelerates data access by storing frequently accessed data in memory, reducing the need for disk I/O and improving overall performance.

Overall, Couchbase provides developers with a powerful and flexible NoSQL database platform that combines the advantages of JSON flexibility with the scalability and performance required by modern applications. With its document-oriented data model, flexible querying capabilities, and built-in scalability features, Couchbase enables developers to build and deploy robust, high-performing applications that can handle the challenges of today's data-driven world.

- **Nifi**

  NiFi, [6] also known as Apache NiFi, is a powerful open-source data integration platform designed to streamline the process of transferring and transforming data between various systems. Its user-friendly visual interface allows users to easily design and manage complex data pipelines. With NiFi, data engineers and administrators can efficiently route, filter, enrich, and transform data in real time, ensuring seamless data flow across different systems and components. One of the key strengths of NiFi is its ability to handle data provenance, which enables

tracking the complete history and lineage of data. This feature is crucial for data governance and compliance purposes, as it allows organizations to understand the origin, transformations, and destinations of their data. Data provenance in NiFi provides granular visibility into each step of the data flow, aiding in troubleshooting, auditing, and ensuring data integrity and quality.

NiFi's extensible and modular architecture makes it highly adaptable to diverse use cases and data integration scenarios. It supports a wide range of connectors, processors, and controllers that can be easily configured and orchestrated to meet specific requirements. Additionally, NiFi offers advanced features such as flow control, load balancing, and data prioritization, enabling efficient data management and resource utilization.In summary, NiFi is an open-source data integration platform that simplifies the process of transferring, transforming, and managing data across various systems. Its visual interface, data provenance capabilities, and extensibility make it a valuable tool for organizations seeking to optimize their data integration workflows, ensure data governance, and achieve seamless data flow and transformation[6].

- **SQLite**

  SQLite is a lightweight, serverless relational database management system (RDBMS) widely used in research and industry. It offers a compact and self-contained database engine that can be embedded within applications, eliminating the need for separate server processes. SQLite's simplicity, efficiency, and cross-platform compatibility make it an attractive choice for managing structured data in various domains, including IoT, mobile applications, and small-scale deployments.

- **InfluxDB**

  InfluxDB is a popular time-series database designed specifically for handling high volumes of time-stamped data. It provides efficient storage, retrieval, and analysis of time-series data, making it suitable for a wide range of applications such as IoT monitoring, real-time analytics, and operational metrics tracking. InfluxDB's scalable architecture, flexible querying language (InfluxQL), and support for continuous queries and downsampling make it a valuable tool for researchers and practitioners seeking to analyze and visualize time-series data in their thesis writeups.

3.0.4 Monitoring Tools

Monitoring tools are essential components in any system or network infrastructure as they provide real-time visibility into the performance, availability, and health of various components. These tools collect and analyze data from different sources, enabling administrators to identify and address potential issues proactively. With features such as automated alerts, customizable dashboards, and historical data analysis, monitoring tools play a crucial role in ensuring the stability, reliability, and optimal functioning of systems and networks.

- **Grafana**

  Grafana is an open-source platform that provides powerful data visualization and monitoring capabilities. It is widely used to create interactive and visually appealing dashboards that allow users to analyze and visualize data from diverse sources. Grafana supports a wide range of data storage and monitoring systems, enabling users to gather real-time data from multiple sources and display it in a unified and intuitive manner.One of the key strengths of Grafana is its user-friendly interface, which makes it accessible to users with varying levels of technical expertise. The platform allows users to customize their dashboards, choosing from a wide range of visualization options, including charts, graphs, and tables. Additionally, Grafana offers advanced features such as alerting capabilities, allowing users to set up notifications based on predefined conditions, and ensuring proactive monitoring of critical metrics.

  Grafana's extensive plugin support is another notable feature, providing users with the ability to extend its functionality and integrate with various data sources and services. This flexibility enables users to connect to different data storage systems, such as relational databases, time-series databases, and cloud-based platforms, allowing them to gather and visualize data from diverse sources in a unified dashboard. Overall, Grafana is a highly versatile platform that empowers users to analyze and visualize data in a user-friendly and customizable manner. Its support for multiple data sources, extensive visualization options, alerting capabilities, and plugin ecosystem make it a valuable tool for researchers, analysts, and professionals in various industries who seek to gain insights from their data and monitor key metrics effectively.

- **Zabbix**

  Zabbix is a widely used open-source monitoring solution that provides comprehensive monitoring and alerting capabilities for networks, servers, applications, and other IT resources. With its flexible and scalable architecture, Zabbix enables administrators to monitor and collect data from a variety of sources, including performance metrics, log files, and SNMP devices. It offers a user-friendly web interface that allows for easy configuration, visualization, and analysis of monitored data. Zabbix provides a wide range of monitoring features, including real-time monitoring, trend analysis, event correlation, and anomaly detection. It supports proactive monitoring through customizable triggers and alerts, allowing administrators to receive notifications when predefined thresholds are breached or specific events occur. Zabbix also offers extensive reporting capabilities, enabling users to generate detailed reports and graphs to gain insights into system performance and trends over time.

  Furthermore, Zabbix supports a distributed monitoring architecture, allowing the deployment of multiple Zabbix servers and proxies to efficiently monitor large-scale environments. It also provides integration with other tools and systems, enabling seamless integration with existing infrastructure and workflows. Overall, Zabbix is a powerful and flexible monitoring solution that empowers organizations to effectively monitor and manage their IT infrastructure, ensuring optimal performance, availability, and reliability.

3.0.5 Network Connectivity and Protocols

Network connectivity and protocols are fundamental aspects of IoT systems, allowing devices to communicate and exchange data. Various protocols, such as MQTT and CoAP, provide efficient and secure communication, while network technologies like Wi-Fi and cellular connectivity enable seamless connectivity between devices and the Internet.

- **MQTT**

  MQTT, also known as Message Queuing Telemetry Transport, is a lightweight messaging protocol specifically designed for efficient communication within Internet of Things (IoT) systems. Its publish-subscribe model enables devices to publish data

to specific topics, while other devices or servers can subscribe to those topics to receive the published data. MQTT is highly regarded for its minimal bandwidth and resource usage, making it ideal for IoT applications involving devices and networks with limited resources.One of MQTT's key advantages is its lightweight nature, enabling it to operate effectively in low-bandwidth and unreliable network conditions. It requires minimal processing power and has a small network footprint, allowing for implementation on various devices, including small microcontrollers and low-power sensors. Additionally, MQTT incorporates a Quality of Service (QoS) mechanism that supports different levels of message reliability, ensuring data delivery based on the desired level of assurance. The publish-subscribe model employed by MQTT offers scalability and flexibility in data transmission within IoT systems. Devices can publish data to multiple topics, and interested parties can subscribe to those topics to receive real-time data. This decoupling of senders and receivers simplifies data distribution and reduces the complexity associated with point-to-point communication. MQTT also supports diverse communication patterns, such as one-to-one, one-to-many, and many-to-many, accommodating a wide range of IoT use cases. MQTT is a lightweight and efficient messaging protocol that has gained significant popularity in IoT deployments. Its minimal bandwidth and resource requirements, combined with the publish-subscribe model, make it well-suited for IoT applications that prioritize resource conservation and scalable communication[7].

- **VPN**

  To ensure secure access to the gateway, we have chosen to implement WireGuard VPN. WireGuard VPN is a contemporary and reliable virtual private network protocol that offers enhanced security features. It utilizes efficient encryption techniques to establish secure connections over untrusted networks, such as the Internet. WireGuard stands out for its simplicity, impressive performance, and strong security measures, making it a preferred solution for safeguarding communication in diverse environments, including IoT implementations.

- **4G LTE**

  To ensure consistent and secure internet connectivity for the gateway, a 4G LTE modem is employed. This allows the gateway to have uninterrupted access to the internet, ensuring its availability at all times. 4G dongles are widely utilized in IoT applications where wired connections are not feasible or accessible, enabling seamless remote connectivity and efficient transmission of data. These dongles leverage the high-speed capabilities of 4G networks, offering reliable and robust connectivity for IoT deployments. The use of a 4G LTE modem ensures that the gateway remains connected even in scenarios where traditional wired connections are impractical, providing a reliable and versatile solution for IoT applications.

- **BLE**

  BLE (Bluetooth Low Energy) is a wireless communication technology specifically developed for establishing short-range connections between devices. It facilitates the exchange of data with minimal power consumption, making it highly suitable for IoT devices and applications that prioritize energy efficiency. BLE enables devices to communicate with each other in close proximity, typically within a range of a few meters. It offers a reliable and efficient means of transmitting data, making it an essential technology for various IoT use cases, such as smart home devices, wearable technology, and asset tracking systems. With its low power requirements and optimized data transfer mechanisms, BLE has emerged as a key enabler for IoT solutions that demand reliable, low-energy communication between devices[1].

3.1 Framework

The proposed architecture focuses on distributing services across different computing sites to enhance modularity and flexibility. The underlying services in the framework are designed as separate deployable components or containers, allowing for easy management and testing while abstracting their implementation details. This approach enables small and loosely coupled characteristics, improving the testability and manageability of the application. By leveraging containerization, the necessary environmental capabilities are provisioned on each end device node, ensuring the proper functioning of the application.

Figure 3.1 depicts the layers of the proposed architecture, each accompanied by its corresponding computational resources. The device layer, also known as the edge layer, comprises the essential capabilities for sensing and communication with the surrounding environment. Sensor nodes are organized into groups, simplifying connection and disconnection management. The gateways work collaboratively to efficiently cache and process transmitted data, resulting in reduced system response times. Serving as a single-entry point to local systems, the middle layer orchestrates communication among the sensing nodes. In the cloud layer, the manager containers closely coordinate with device groups to ensure the updates of system resources. By facilitating data and resource status sharing across multiple local domains, the manager containers enable the concurrent execution of resource-intensive applications.

The primary goal of the proposed scheme is to effectively manage multiple deployed sensor nodes and their resources within dynamic distributed systems. The manager, represented by Rancher in Figure 3.2, plays a crucial role in optimizing resource allocation by monitoring service deployment. It efficiently handles sensor node groups, facilitates system scale-up, and ensures the timely update of containers across machine clusters. This approach enables the efficient management of resources in highly dynamic distributed systems, promoting effective coordination and scalability.

- Sense Layer/Edge Layer: The sensing layer of the proposed architecture consisted of multiple sensor nodes strategically deployed throughout the testing environment. These sensor nodes are equipped with various functionalities such as sound detection, door monitoring, CO2 measurement, movement sensing, and PIR (Passive Infrared) sensing, among others. Each sensor node operated independently and is poared by its onboard batteries, providing backup availability for up to two years. The nodes utilized Bluetooth Low Energy (BLE) modules for wireless communication, enabling them to transmit and receive messages to and from the gateway seamlessly.

- Gateway Layer: The gateway layer played a crucial role in facilitating communication between the edge layer (sensor nodes) and the cloud layer. Containerized services are deployed within the gateway, encapsulating specific functionalities and utilizing various network protocols for data transmission. Additionally, an intermediate database is managed at this layer to ensure data integrity and prevent

loss in case of network unavailability. When network connectivity is disrupted, the data is cached within the database, and once the connection is restored, it is transmitted through the appropriate channels. This approach guaranteed reliable data transmission and mitigated the risk of data loss during temporary network disruptions.

- Cloud Layer: The cloud layer encompassed cloud services responsible for the management and accessibility of the implemented microservices resources. Central message hub services are deployed and fully managed in the cloud, providing controlled data transmission through efficient queuing mechanisms. Database servers are utilized to store persistent data for future use and enable efficient data retrieval. Web services are employed to facilitate data access and user permissions, ensuring secure and controlled interactions across various user services within the cloud layer. Additionally, data analytics operations are conducted in the cloud, taking advantage of the scalability and abundant resources available in the cloud infrastructure, which are typically limited at the edge nodes.

  To facilitate the deployment of these services, Argo CD, a GitOps continuous delivery tool, is utilized. Deploying services with Argo CD offered several advantages. It followed a declarative approach to manage deployments, ensuring consistency and reproducibility across different environments. The integration with Git enabled version control and change tracking of the deployed services, simplifying the management of configurations and updates. Argo CD also provided automated deployment and rollback capabilities, streamlining the process of deploying and managing complex services. With its user-friendly interface and robust access control mechanisms, Argo CD facilitated efficient collaboration among teams, enhancing the reliability, scalability, and maintainability of the entire system.

Chapter 4

Implementation and Results

4.1 Implementation

4.1.1 Practical use case

In developed countries, the aging population is on the rise due to increasing life expectancy and declining birth rates. Assisted living has become a prominent concept to cater to the needs of the elderly. While old homes or healthcare centers are available options, many senior citizens prefer to stay in their own homes. However, ensuring their health and well-being becomes a challenge as regular visits from healthcare staff can be expensive and time-consuming.

To address this issue, there is a growing need for smart healthcare systems specifically designed for the elderly. These systems would enable healthcare providers to remotely monitor the health conditions of patients and provide affordable healthcare services. Previous research has proposed IoT frameworks for healthcare, but they often lack reusability, modular design, and security.

This research aims to fill these gaps by proposing IoT frameworks tailored for smart healthcare in elderly care. These frameworks prioritize key capabilities such as reusability, scalability, accessibility, interoperability, and update ability. The goal is to develop flexible and adaptable solutions that can be easily deployed, expanded, integrated with other systems, and updated to meet the evolving needs of elderly care.

By implementing these IoT frameworks, healthcare providers can remotely assess the health conditions of elderly patients, ensuring their well-being while allowing them to

remain in the comfort of their own homes. Additionally, the frameworks enable cost-effective and accessible healthcare solutions, benefiting both the patients and healthcare providers.

4.1.2 Floor Plan

In the deployment of sensor nodes, a strategic arrangement has been made to ensure comprehensive monitoring of the living environment. Five sound sensors have been deployed in key locations including the hallway, living room, kitchen, bathroom, and bedroom. These sound sensors are responsible for capturing audio data and providing valuable insights into the activities and events occurring in each area.(Figure 4.2)

To validate the accuracy of the information captured by the sound sensors, infrared (IR) sensors have been placed in the same locations. These IR sensors serve as a complementary technology to confirm the presence and movement detected by the audio sensors, enhancing the reliability and accuracy of the overall monitoring system. In addition, light sensors have been deployed in the bedroom and living room to validate occupancy. These sensors can detect variations in light levels, helping to determine whether the rooms are currently occupied or vacant.

To monitor the outdoor movements of the patients, a door sensor has been installed. This sensor can detect and report any instances of the patient leaving the house or attempting to exit through the monitored door.

To ensure effective communication and data transmission between all the deployed sensors and the central monitoring system, a gateway device has been strategically positioned in the hallway, which serves as the middle of the house. This location allows the gateway to be within the optimal broadcast range of all the deployed sensors, facilitating seamless data collection and transmission.

Overall, this well-planned deployment of sensor nodes ensures comprehensive coverage of the living space, enabling efficient monitoring and providing valuable insights into the activities, occupancy, and movements within the house.

4.1.3 Pre-Deployment Steps

Before deploying the sensor nodes, several crucial steps are necessary to ensure the smooth operation and functionality of the system. Firstly, the gateways need to be bootstrapped by manually installing the Ubuntu 20.04 operating system. Following that, all the required packages are installed using Ansible, including the BLE stack, Zabbix agent, Docker, K3s, Argo CD, and the integration with Rancher. These installations are essential for establishing the foundation of the gateway and enabling its connectivity and communication capabilities.

- Bootstrap the gateways: Manually install the Ubuntu 20.04 operating system on the gateways.
- Install required packages: Use Ansible to install necessary packages such as the BLE stack, Zabbix agent, Docker, K3s, Argo CD, and establish the link with Rancher.
- Configure sensors: Set up the sensors to transmit data via Bluetooth Low Energy (BLE) for communication with the gateway.
- Calibrate sound sensors: Fine-tune the sound sensors to ensure accurate data collection and monitoring of the acoustic environment.
- Configure pipelines in the cloud: Set up pipelines to extract data from the messaging broker and load it into the database.
- Configure post-processing pipelines: Create pipelines to process the stored data for visualization on the dashboard.
- Establish plug-and-play capability: Ensure that the system is ready for seamless integration and deployment.

4.1.4 Deployment Steps

It is important to configure all the sensors to transmit their information via Bluetooth Low Energy (BLE). This configuration ensures that the sensors can effectively broadcast the collected data to the gateway, establishing the communication link between the sensor nodes and the central monitoring system.

- Position sensor nodes: Deploy the sensors according to the desired floor plan, placing them in locations such as the hallway, living room, kitchen, bathroom, and bedroom.
- Turn on the gateway: Activate the gateway device to establish communication with the deployed sensor nodes.
- Activate microservices: After a minimal interval, the microservices within the gateway and cloud infrastructure come alive, ready to receive and process the sensor data.
- Monitor data transmission: Monitor the data transmission from the sensors to the gateway, ensuring that the information is being captured accurately.
- Visualize data on the dashboard: Use the post-processing pipelines to process the data and display it on the dashboard for easy visualization and analysis.

By following these steps, the system is equipped with a plug-and-play capability, allowing for seamless integration of the deployed sensors and ensuring that the gateway and microservices are operational. Once the sensors are deployed according to the desired floor plan, the gateway device is turned on, and after a minimal interval, the microservices come alive and start listening to the messages broadcasted by the sensors. This synchronization enables real-time data collection and monitoring within the smart healthcare system.

The implementation pipeline depicted in the figure 4.1 demonstrates an efficient data broadcasting process using Bluetooth Low Energy (BLE) to the gateway. The microservice deployed on the gateway utilizes the Kubernetes proxy API communication capabilities to receive these broadcasted messages via the host network. By leveraging the host network effectively and providing the necessary credentials to the respective pods, the messages are securely transmitted to the gateway.

Once received, these messages are forwarded to the MQTT broker deployed in the cloud using the TCP network protocol. This ensures that the messages are promptly transmitted to the cloud with minimal latency. To account for potential network disruptions or temporary outages, the gateway also persists the data locally using SQLite, ensuring data availability and reliability.

In the cloud infrastructure, the MQTT broker receives the messages from the gateway, enabling seamless data transfer. To extract data from the MQTT broker and load it into the database, an open-source tool called Apache NiFi is utilized. This tool not only facilitates the data transfer process but also offers additional benefits such as data lineage, tracking, validation, historical analysis, and performance optimization. These features are especially valuable in the IoT domain, where multiple data generation points need to be efficiently managed.

To visualize the data on the dashboard, a FastAPI-based microservice is deployed in the cloud stack. This microservice performs the necessary post-processing on the data, preparing it for visualization on Grafana. As the primary database does not have a built-in plugin for our requirements, we opted for InfluxDB as the data source for the dashboard. InfluxDB provides the necessary capabilities to store and retrieve data efficiently, enabling seamless integration with the Grafana dashboard.

Overall, this implementation pipeline ensures a robust and streamlined flow of data from the sensor nodes to the cloud, incorporating data persistence, extraction, processing, and visualization steps. By leveraging advanced technologies and tools, the system can efficiently handle the challenges associated with data management and visualization in IoT applications.

4.2 Results

After the implementation let's discuss the main components and how they performed in our proposed framework. The main focus here to see if all the functionalities present the and framework is able to deliver all the required tools and services microservice.

After the deployment, the gateway cluster is visible in the UI which can be seen in Fig 4.15. So as soon as the gateway comes live we have complete control of the cluster and can deploy workload and manage the cluster remotely. Rancher's Projects enable dynamic grouping of related namespaces, allowing for strict access control and resource limits. which are very important if multiple teams are working inside the same cluster. The microservice running on the gateways is able to scan the broadcasted messages from the sensor using host ble stack Fig 4.14 shows this. In Fig 4.6 and Fig 4.7, the Nifi running on the cloud all relevant processors are configured to extract data sent

to the MQTT broker and load it to Couchbase our primary Database. The data is then post-processed and sent to the relevant dashboard to be shown to the healthcare provider to access and examine the situation of the patients. Fig 4.8 illustrates the activity happening in the bedroom and different sensors present in the bedroom and their respective charts and 4.9 show the sound sensor performance and its ability to detect different activities performed in the bathroom. All of these charts are shown to the healthcare provider to better access the patient's health and also observe their pattern and plan the visits accordingly.

The main components of the cloud stack are provisioned using Argo CD which is visible in Fig 4.2 in deployment of this kind when adding a new functionality, function, feature, or maintenance there is no need to manually deploy changes no member of the team needs access to the cluster itself all configuration and deployment files resides inside a git repository and only way to commit changes inside the cluster is to commit changes in git repository so in this way our framework encompasses the single source of truth principle. Argo CD also provides automated deployment and rollback capabilities, simplifying the management of complex deployments. Its intuitive user interface and robust access control mechanisms facilitate efficient collaboration among teams. Overall, deploying services with Argo CD enhances the reliability, scalability, and maintainability of the system.

The monitoring of the gateways deployed is done via a Zabbix agent installed on the gateways and Zabbix server deployed on the cloud. The Zabbix agent is a lightweight software component that can be installed on the gateway devices, enabling them to report valuable metrics and data to the Zabbix monitoring system. This allows for real-time monitoring of key performance indicators, such as CPU usage, memory utilization, and network connectivity. Exported metrics are seen in Fig 4.3, 4.10,4.11, and 4.13.

In addition to the metrics displayed By monitoring IoT gateways with the Zabbix agent, administrators can gain insights into the health and performance of the gateways, identify potential issues or bottlenecks, and proactively address them. This ensures the smooth operation and optimal functionality of the IoT gateway infrastructure.

We will judge our gateway on the limitations that we previously discussed in the methodology section and some essential functionalities that we think our gateway must have:

- Monitoring: provided by Zabbix open source tool

- scalability: provided by deploying services on a Kubernetes cluster
- High Availability: and Resilience also provided by kubernets
- Analytics and Insights: Grafana is to provide valuable analytics which is also open source
- Interoperability: microservice architecture enables us to have an interoperable system
- Remote Management: Rancher and Argo give us the ability of remote management
- Flexibility.
- Edge Computing Capabilities: the workload deployed on the k3s cluster running on the gateway provides us with necessary
- Device Management: With Rancher Device management
- Continuous delivery: Argo CD

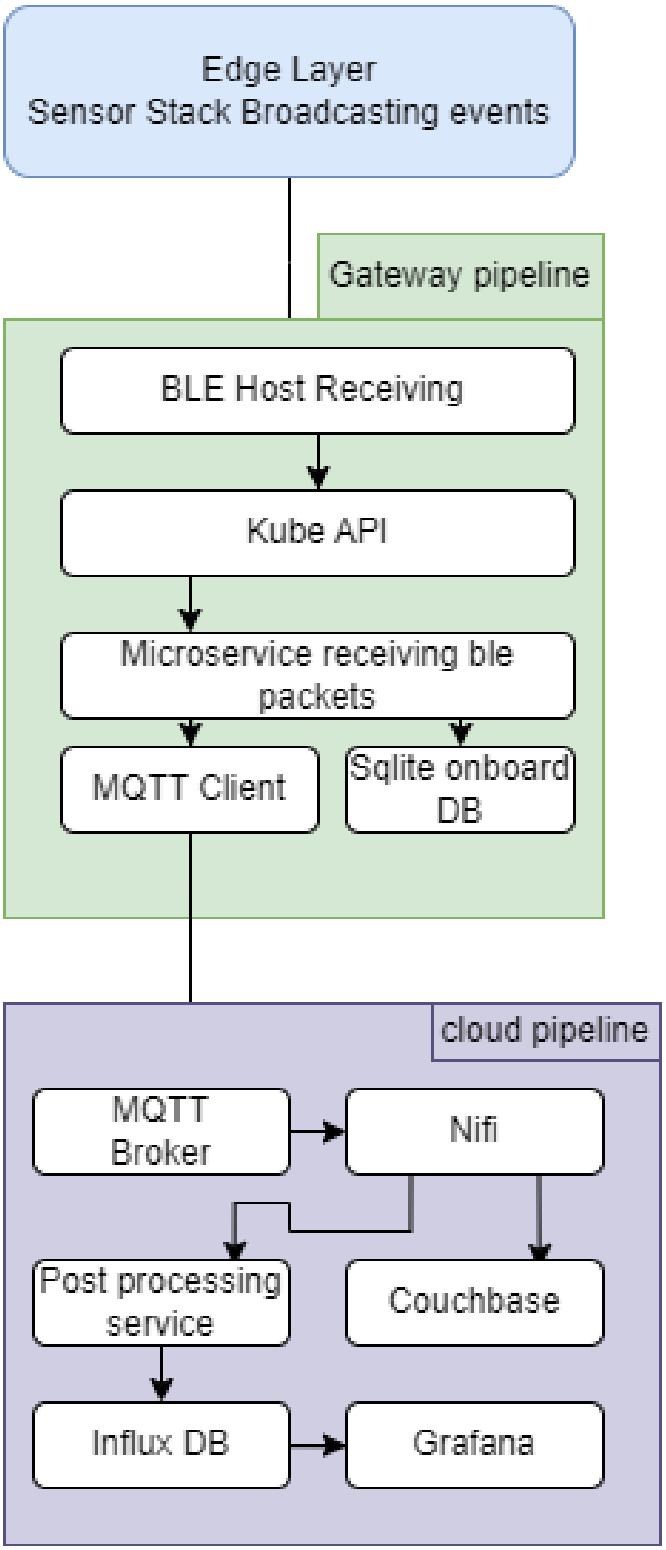

Figure 4.1: Implementation Flow Diagram

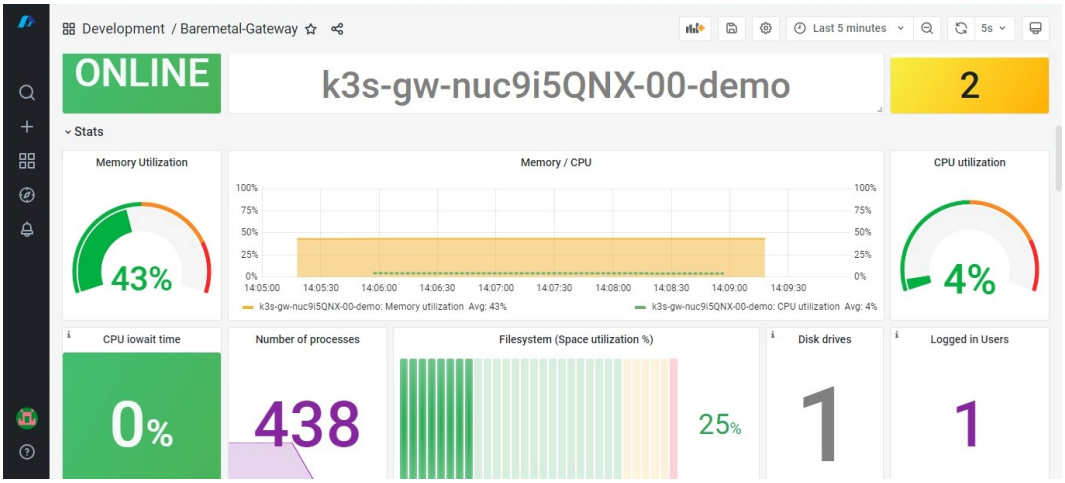

Figure 4.2: Grafana Dashboard for device health monitoring

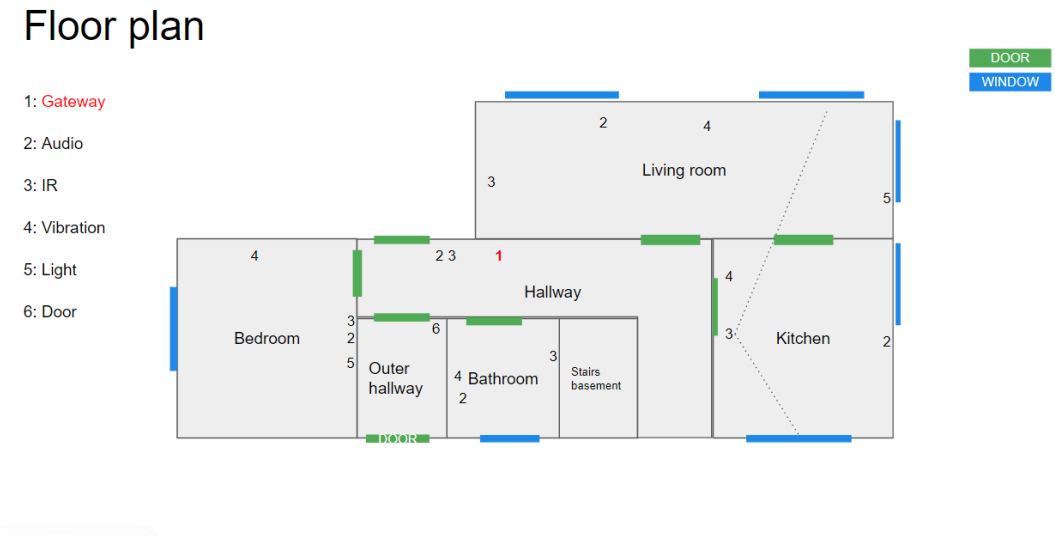

Figure 4.3: Floor Plan

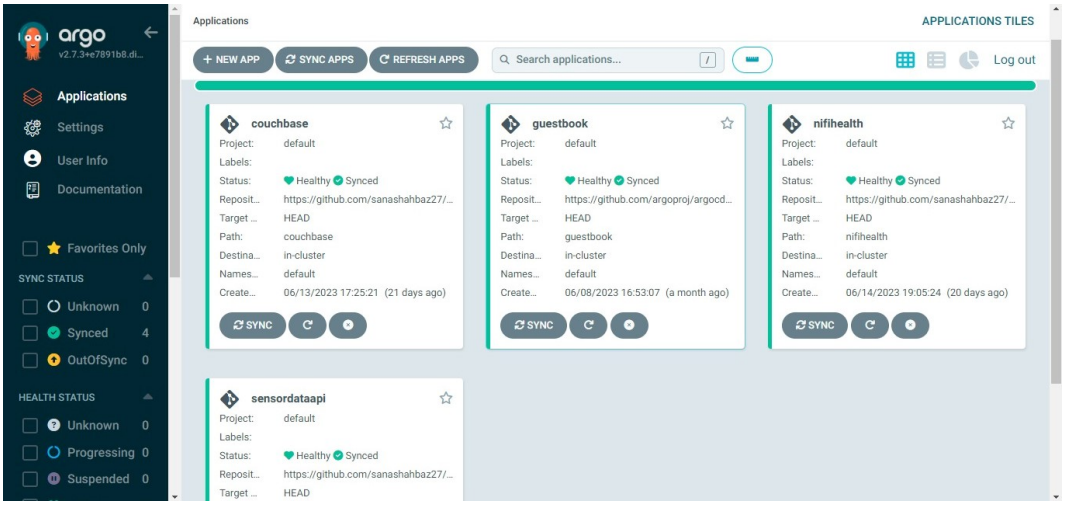

Figure 4.4: Argo CD UI for Cloud stack Deployment

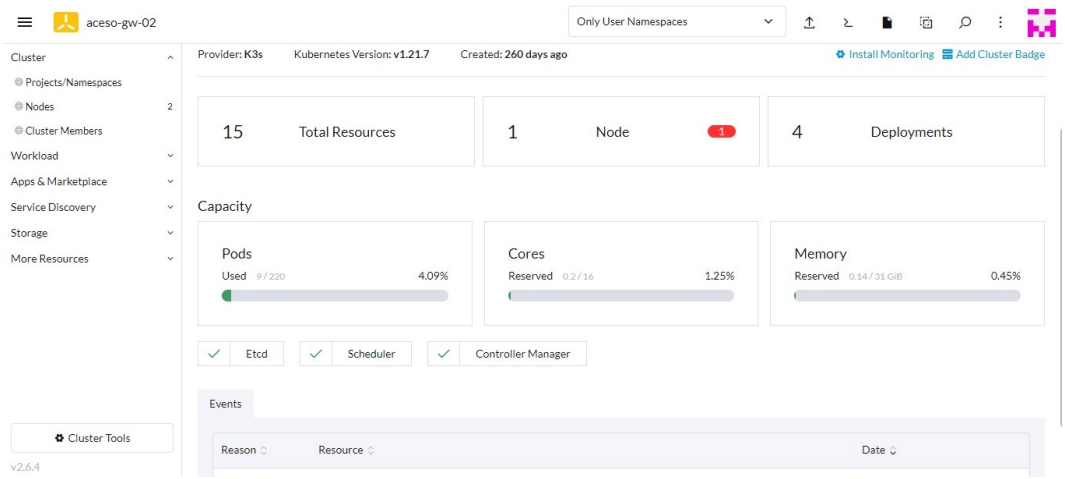

Figure 4.5: Gateway cluster visible from Rancher UI

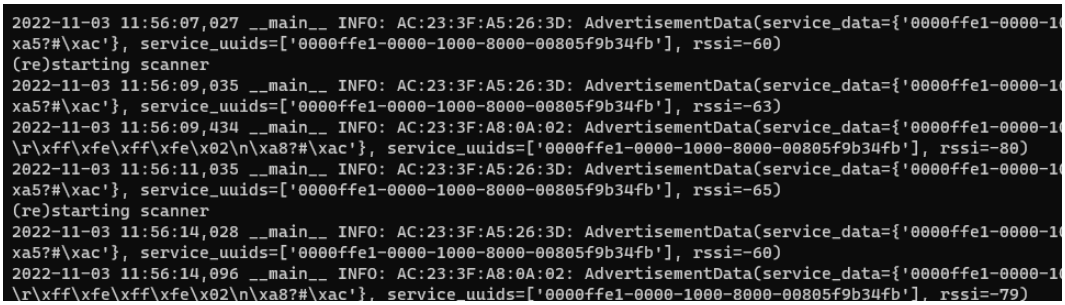

Figure 4.6: View of logs running in microservices

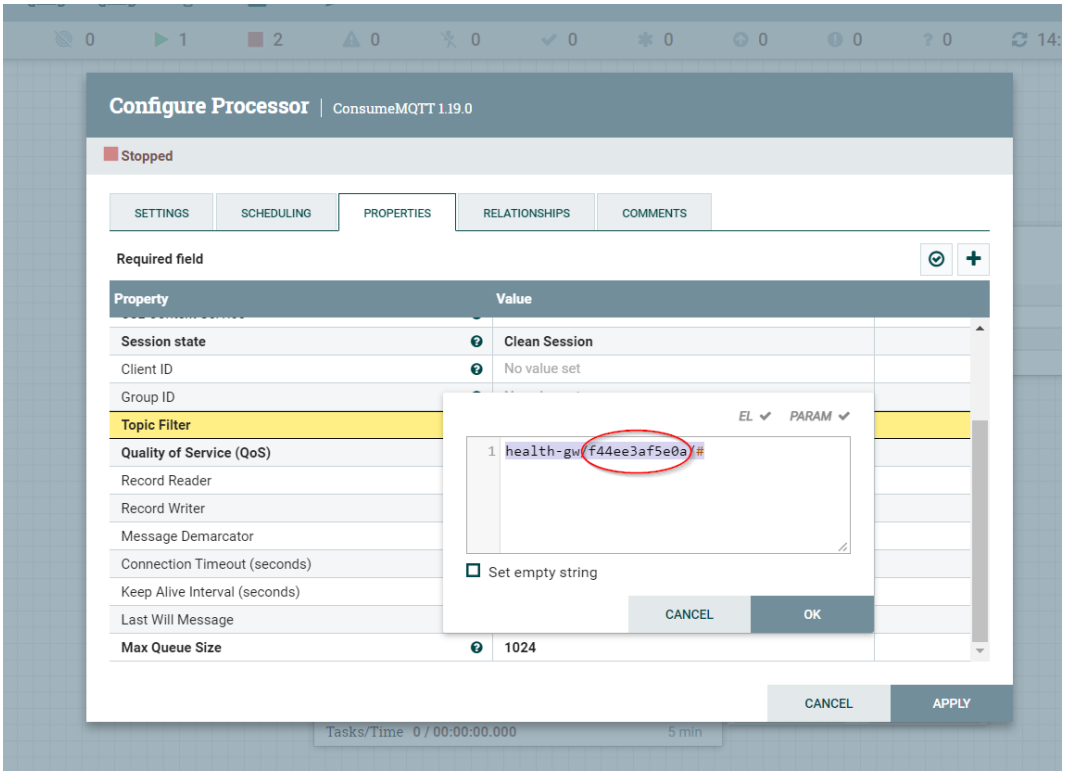

Figure 4.7: Configuration of Couchebase

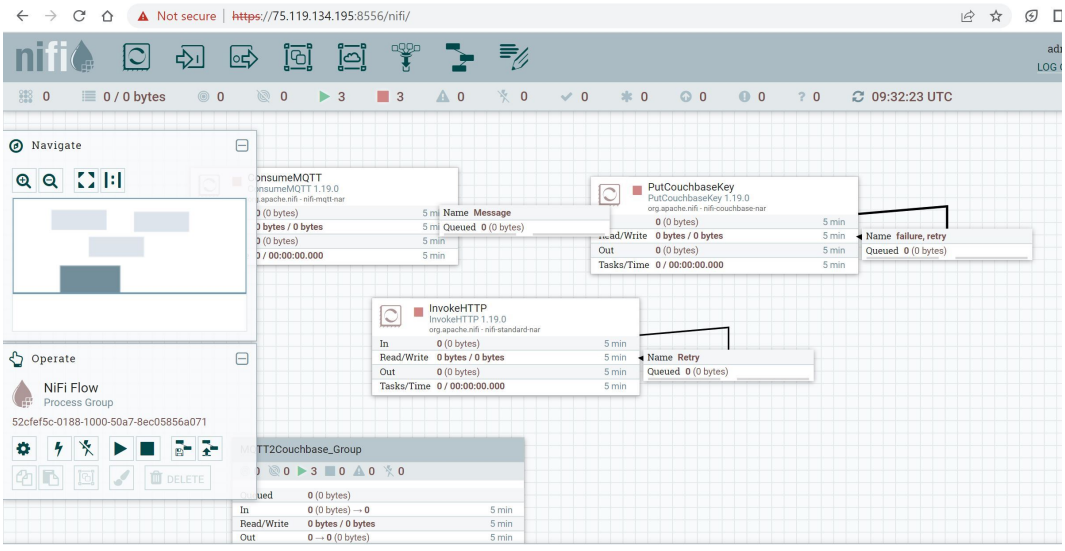

Figure 4.8: Complete Pipeline and relevant Processors

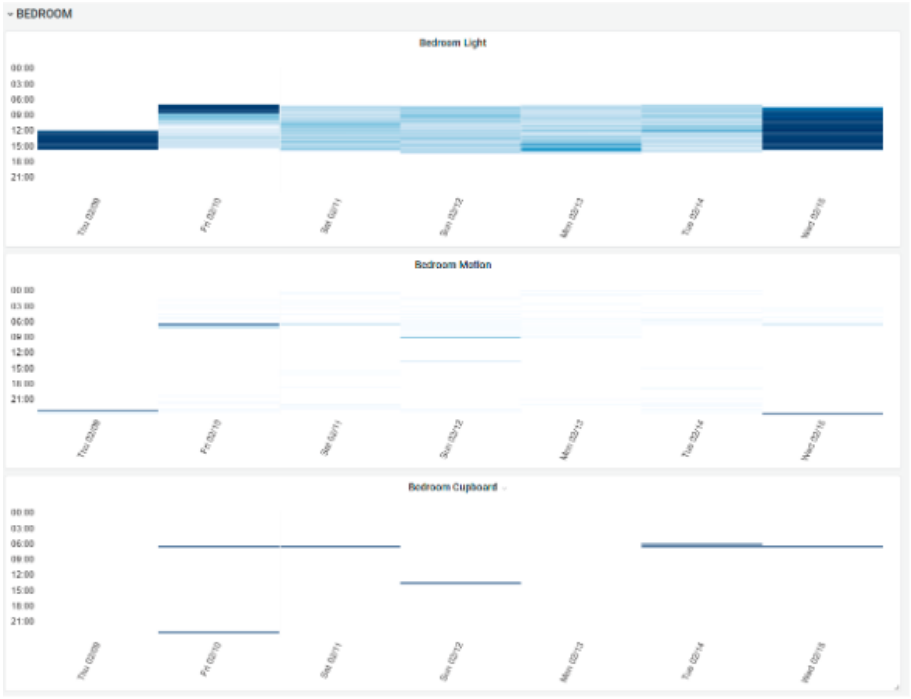

Figure 4.9: A Plot for activity in the bedroom area after post-processing

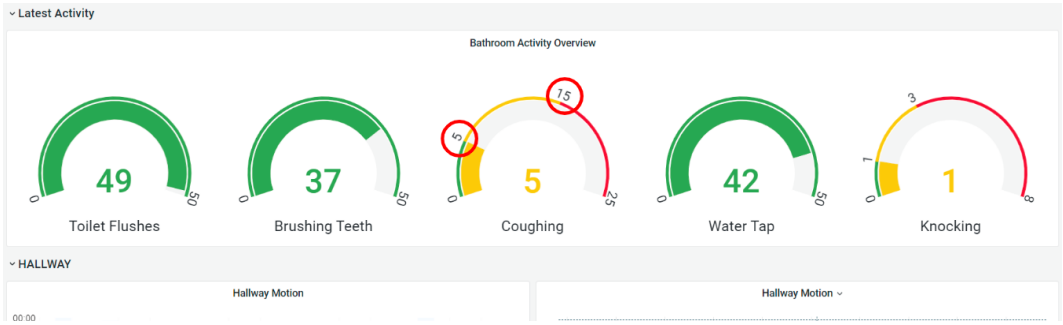

Figure 4.10: Overall activity tracker of the patient in-house

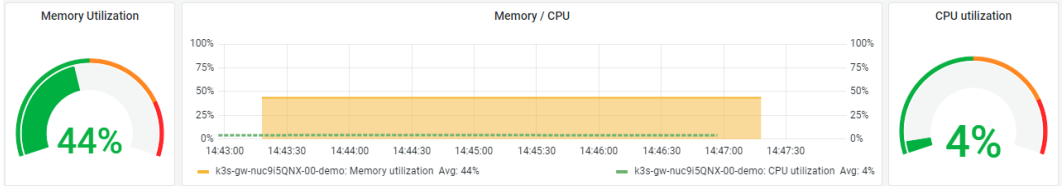

Figure 4.11: Monitoring Functionality using Zabbix Agent and Grafana

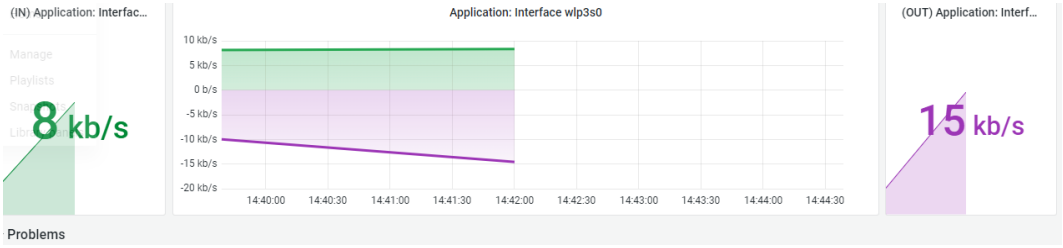

Figure 4.12: Network monitoring through Zabbix agent 1

State	Name	Version	Provider	Machines	Age	
Active	aceso-gw-01	v1.21.7+k3s1	K3s	1	16 days	Explore
Active	aceso-gw-02	v1.21.7+k3s1	K3s	1	6 mins	Explore
Active	local	v1.22.7+k3s1	Local K3s	2	109 days	Explore

Figure 4.13: All gateway clusters available at rancher UI

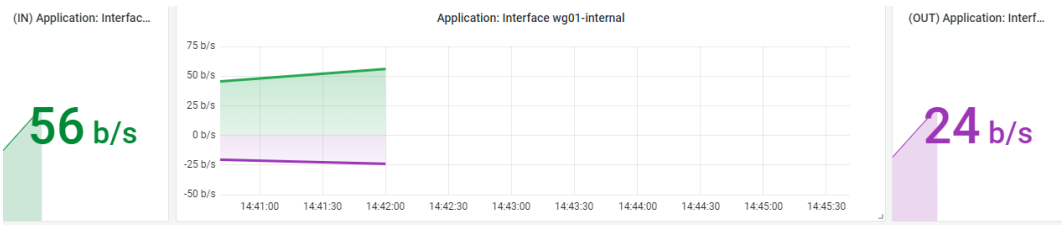

Figure 4.14: Network monitoring through Zabbix agent 2

Chapter 5

Conclusion and Future Work

5.1 Conclusion

This research demonstrates the potential of integrating IoT techniques and lightweight virtualization technology through a containerized architectural framework for smart home healthcare monitoring systems. By combining sensor networks and containerized microservices, the deployment of multiple IoT applications simultaneously in senior citizen homes becomes feasible. The centralized management and orchestration of these applications, following the principles of DevOps, contribute to efficient resource utilization and scalability.

The proposed smart healthcare monitoring system offers significant advantages for elderly individuals who prefer to age in place. It enables continuous monitoring of their activities and well-being,[12] allowing healthcare providers to offer timely assistance when necessary. Through the use of containerization, the system provides flexibility and adaptability to cater to the unique needs of each individual, promoting independence and enhancing their overall quality of life.

This research emphasizes the potential of the containerized framework to revolutionize home healthcare for the elderly. The seamless integration of IoT technologies and virtualization opens up new opportunities for remote healthcare monitoring, improved resource management, and personalized care. Future research can explore the integration of legacy network protocols to enhance communication capabilities, ensuring seamless connectivity in diverse home environments.

In conclusion, the implementation and evaluation of the proposed smart healthcare

monitoring system lay the foundation for advancing home-based healthcare solutions to meet the increasing needs of an aging population. By embracing emerging technologies and innovative approaches, we can envision a future where elderly individuals can age comfortably and securely in their own homes while receiving the necessary care and support[10].

5.2 Future work

While the implemented system showcases a robust implementation pipeline for data broadcasting and processing, there are several areas that can be explored for future enhancements and advancements.

It is important to address the reliance on the host network for the BLE service pod to receive messages. Currently, the system requires an API proxy to effectively communicate with the host network. In the future, further research and development can focus on improving the communication capabilities between the service pod and the host network. This could involve exploring alternative communication mechanisms or developing more efficient proxy solutions to enhance the overall system performance.

Additionally, an interesting avenue for future work is the communication ability with legacy network protocols. The current implementation predominantly focuses on utilizing modern network protocols, such as BLE and TCP. However, many existing systems and devices still rely on legacy network protocols. Extending the system's communication abilities to seamlessly integrate with legacy protocols would greatly enhance its interoperability and compatibility with a wider range of devices and networks.

So the system's scalability and adaptability can be further explored. As the number of sensor nodes and data sources increases, ensuring efficient data handling and processing becomes crucial. Future work can focus on optimizing the system's architecture, leveraging technologies like edge computing and distributed processing, to handle large-scale deployments and high-volume data streams effectively.

Lastly, ongoing advancements in data analytics and machine learning can be leveraged to enhance the system's capabilities. This includes exploring predictive analytics, anomaly detection, and real-time data processing techniques to extract valuable insights from the collected data. Integrating advanced analytics capabilities can enable proactive mon-

itoring, early detection of critical events, and improved decision-making in healthcare scenarios.

Overall, the implementation process presented in this thesis provides a solid foundation for further exploration and development. By addressing the communication challenges, embracing legacy network protocols, ensuring scalability, and incorporating advanced analytics, the system can continue to evolve and provide even more efficient, reliable, and comprehensive solutions for smart healthcare and IoT applications.

Bibliography

[1] Elke Mackensen, Matthias Lai, and Thomas M Wendt. "Bluetooth Low Energy (BLE) based wireless sensors". In: *SENSORS, 2012 IEEE*. IEEE. 2012, pp. 1–4.

[2] Alexandr Krylovskiy. "Internet of things gateways meet linux containers: Performance evaluation and discussion". In: *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*. IEEE. 2015, pp. 222–227.

[3] Corentin Dupont, Raffaele Giaffreda, and Luca Capra. "Edge computing in IoT context: Horizontal and vertical Linux container migration". In: *2017 Global Internet of Things Summit (GIoTS)*. IEEE. 2017, pp. 1–4.

[4] Babak Bashari Rad, Harrison John Bhatti, and Mohammad Ahmadi. "An introduction to docker and analysis of its performance". In: *International Journal of Computer Science and Network Security (IJCSNS)* 17.3 (2017), p. 228.

[5] Abdulatif Alabdulatif et al. "Secure edge of things for smart healthcare surveillance framework". In: *IEEE Access* 7 (2019), pp. 31010–31021.

[6] Saroja Chatti. "Using spark, kafka and NIFI for future generation of ETL in IT industry". In: *J Innov Inf Technol* 3.2 (2019), pp. 11–14.

[7] Dan Dinculeană and Xiaochun Cheng. "Vulnerabilities and limitations of MQTT protocol used between IoT devices". In: *Applied Sciences* 9.5 (2019), p. 848.

[8] Tom Goethals, Filip De Turck, and Bruno Volckaert. "Extending kubernetes clusters to low-resource edge devices using virtual kubelets". In: *IEEE Transactions on Cloud Computing* 10.4 (2020), pp. 2623–2636.

[9] Hafedh Ben Hassen, Nadia Ayari, and Belgacem Hamdi. "A home hospitalization system based on the Internet of things, Fog computing and cloud computing". In: *Informatics in Medicine Unlocked* 20 (2020), p. 100368.

[10] Jaeyeop Jeong et al. “Empirical analysis of containers on resource-constrained IOT gateway”. In: *2020 IEEE International Conference on Consumer Electronics (ICCE)*. IEEE. 2020, pp. 1–6.

[11] Junxia Li et al. “A secured framework for sdn-based edge computing in IOT-enabled healthcare system”. In: *IEEE Access* 8 (2020), pp. 135479–135490.

[12] Bryan Pearson and Daniel Plante. “Secure Deployment of Containerized IoT Systems”. In: *2020 SoutheastCon*. Vol. 2. IEEE. 2020, pp. 1–8.

[13] Marco Anisetti et al. “Towards an assurance framework for edge and IoT systems”. In: *2021 IEEE International Conference on Edge Computing (EDGE)*. IEEE. 2021, pp. 41–43.

[14] Daniil Ermolenko et al. “Internet of Things services orchestration framework based on Kubernetes and edge computing”. In: *2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus)*. IEEE. 2021, pp. 12–17.

[15] Janis Judvaitis, Rihards Balass, and Modris Greitans. “Mobile IoT-Edge-Cloud Continuum Based and DevOps Enabled Software Framework”. In: *Journal of Sensor and Actuator Networks* 10.4 (2021), p. 62.

[16] Yu Liu et al. “Performance evaluation of containerization in edge-cloud computing stacks for industrial applications: A client perspective”. In: *IEEE Open Journal of the Industrial Electronics Society* 2 (2021), pp. 153–168.

[17] T Maragatham, P Balasubramanie, and M Vivekanandhan. “IoT Based Home Automation System using Raspberry Pi 4”. In: *IOP Conference Series: Materials Science and Engineering*. Vol. 1055. 1. IOP Publishing. 2021, p. 012081.

[18] Itamir de Morais Barroca Filho et al. “An IoT-based healthcare platform for patients in ICU beds during the COVID-19 outbreak”. In: *Ieee Access* 9 (2021), pp. 27262–27277.

[19] Emmanuel Raj et al. “Edge mlops: An automation framework for aiot applications”. In: *2021 IEEE International Conference on Cloud Engineering (IC2E)*. IEEE. 2021, pp. 191–200.

[20] Naif Khalaf Al-Shammari, TH Syed, and Muzamil Basha Syed. "An Edge–IoT framework and prototype based on blockchain for smart healthcare applications". In: *Engineering, Technology & Applied Science Research* 11.4 (2021), pp. 7326–7331.

[21] Fan Wu et al. "Edge-based hybrid system implementation for long-range safety and healthcare IoT applications". In: *IEEE Internet of Things Journal* 8.12 (2021), pp. 9970–9980.

[22] Khaled Alanezi and Shivakant Mishra. "Utilizing Microservices Architecture for Enhanced Service Sharing in IoT Edge Environments". In: *IEEE Access* 10 (2022), pp. 90034–90044.

[23] Luiz CBC Ferreira et al. "Edge Computing and Microservices Middleware for Home Energy Management Systems". In: *IEEE Access* 10 (2022), pp. 109663–109676.

[24] Mohammad Goudarzi, Shashikant Ilager, and Rajkumar Buyya. "Cloud Computing and Internet of Things: Recent Trends and Directions". In: *New Frontiers in Cloud Computing and Internet of Things* (2022), pp. 3–29.

[25] Mohammad Goudarzi, Shashikant Ilager, and Rajkumar Buyya. "Cloud Computing and Internet of Things: Recent Trends and Directions". In: *New Frontiers in Cloud Computing and Internet of Things* (2022), pp. 3–29.

[26] Eric Hitimana et al. "Containerized Architecture Performance Analysis for IoT Framework Based on Enhanced Fire Prevention Case Study: Rwanda". In: *Sensors* 22.17 (2022), p. 6462.

[27] Shubha Brata Nath et al. "Containerized deployment of micro-services in fog devices: a reinforcement learning-based approach". In: *The Journal of Supercomputing* (2022), pp. 1–29.

[28] Vu Khanh Quy et al. "Smart healthcare IoT applications based on fog computing: architecture, applications and challenges". In: *Complex & Intelligent Systems* 8.5 (2022), pp. 3805–3815.

[29] Tiago Veiga et al. "Towards containerized, reuse-oriented AI deployment platforms for cognitive IoT applications". In: *Future Generation Computer Systems* 142 (2023), pp. 4–13.

[30] Carlos Martınez Lorezno and Pablo Molina Mata. “Couchbase”. In: ().