A

Mini Project Report

On

Advancing Confidentiality In IOT Health Assistances Utilizing Fog Computing

Submitted in partial fulfillment of the Requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

S. HARSHITHA

(20Q61A0550)

Under the guidance of

MR. VIRUPAKSHI RAVINDRANATH

Department of CSE

Department of Computer Science and Engineering

AVANTHI INSTITUTE OF ENGINEERING & TECHNOLOGY

(Affiliated to JNTUH Approved by AICTE, Recognized by Govt of T.S)

Accredited by NBA, NAAC Gunthapally (V), Abdullapurmet (M),

R.R.District-50512,(2023-2024)

AVANTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY
(Affiliated to JNTUH Approved by AICTE, Recognized by Govt of T.S, Accredited by NBA,NAAC)
Gunthapally (V), HayathNagar (M), R.R.District-501512

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the project work entitled "**Advancing Confidentiality In IOT Health Assistances Utilizing Fog Computing**" is being submitted by **S. Harshitha (20Q61A0550),** in partial fulfillment of requirement for the award of the degree of **B.Tech** in the **Computer Science and Engineering**, Avanthi Institute of Engineering and Technology, Hyderabad is a record of bonafide work carried out by him under my guidance. The results presented in this project have been verified and are found to be satisfactory. The result embodied in this project has not been submitted to an other University for the award of any other degree.

Mr. VIRUPAKSHI RAVINDRANATH
M.Sc
Internal Guide
Department of CSE

Dr.N.RAMANA REDDY
M.C.A,M.Tech,MBA,Ph.D
Asst.professor
Department of CSE
HEAD OF THE DEPARTMENT

EXTERNAL EXAMINER

Dr. G.RAMACHANDRAREDDY
B.Tech,M.Tech, Ph.D
PRINCIPAL

ACKNOWLEDGEMENT

This is an acknowledge of the intensive drive and technical competence of many individuals who have contributed to the success of our project work.

We are grateful to chairman, **Avanthi Group of Institutions Sri. M. SRINIVASA RAO** for granting us the permission for undergoing the practical training through development of this thesis in college.

Our sincere thanks to the **Principal Dr.G.RAMACHANDRAREDDY**, Avanthi Institute of Engineering & Technology and to all the faculty members.

We would like to express our gratitude to head of the department **Dr.N.RAMANA REDDY , M.C.A, M.Tech,MBA,Ph.D**,**Assistant Professor** for his valuable suggestions during the course of our project work.

We are immensely thankful to our B.Tech Co-coordinator **S.RAJENDER, Assistant Professor** for **Department of Computer Science and Engineering** for this work, who helped in completing this project work successfully.

We are immensely thankful to our internal guide **Dr.Virupakshi Ravindranath ,Assistant Professor ,Department of CSE,** for his valuable guidance and suggestion in each and every stage of this work, which helped us in completing this project work successfully.

We are thankful to one and all, which are co-operated with us to complete our project successfully.

DECLARATION

We hereby declare that the results embodied in this dissertation entitled "**Advancing Confidentiality In IOT Health Assistances Utilizing Fog Computing**" is carried out by us during the year 2023-2024 in partial fulfillment of the award of **B.Tech**, **Computer Science and Engineering** from **Avanthi Institute of Engineering and Technology**. We have not submitted the same to any other university or organization for the award of other degree.

Student name with signature

S. Harshitha (20Q61A0550)

ABSTRACT

Internet of Things (IoT) is the interconnection of physical objects or devices that can transmit and receive data through the internet without human involvement. With the advancement in IoT devices particularly in healthcare sector, huge amount of data is collected from different sensors and all this data are transferred and stored in cloud. It becomes difficult to handle such huge amount of data in cloud specially the healthcare data where it requires real time data computation and storage. Security of the data is also major challenge in cloud. Fog computing is the answer to overcome the challenges. Fog nodes works at the edge side and enhances data security, accuracy, consistency and reduces the latency rate which is an important factor for application like medical data. Implementation work is also described in the paper where a digital human temperature sensor device is built using DS18B20 temperature sensor. The data collected from it is being encrypted in fog node using Advance Encryption Standard(AES) algorithm and it is send to cloud. Therefore, the security of the health care data is enhanced using Fog computing.

INDEX

TOPICS

- Certificates
- Acknowledgement
- Abstract
- Figures/Tables of Contents

1. INTRODUCTION

1. INTRODUCTION

A number of IoT services, such as computation resources, storage capabilities, heterogeneity, high processing, and others that brought a technological revolution, are provided by cloud computing. The cloud provides the virtualization of computing resources at various levels. Almost all the human life domains have adopted cloud computing. However, cloud computing has drawbacks in terms of high delays which have an adverse effect on the IoT tasks that require a real-time response. Furthermore, it does not match industrial control systems which require a low-delay response time. In 2012, Cisco announced an infrastructure paradigm called fog computing, which is a new computing concept, so as to tackle the limitations of cloud computing. They asserted that fog computing is applicable at three networking levels: the collection of data from the devices in the edge (sensors, vehicles, roadways, and ships); multiple devices connecting to a network and sending all the data; the collected data from the devices should be processed in less than a second along with decision making.

The term fog computing shifts capabilities of the cloud near to the end user, and provides storage, computation, and communication to edge devices, which facilitate and enhance mobility, privacy, security, low latency, and network bandwidth so that fog computing can perfectly match latency-sensitive or real-time applications. On the one hand, fog computing infrastructure consists of plenty of fog nodes, edge device networks, and even virtualized data Centres or IoT devices that are connected to these nodes. These are connected to the cloud for the purpose of implementing large storage and rich computing. The distribution of functions between the cloud and the fog nodes is considered a crucial factor. Millisecond to sub-second latency offered by fog, even faster than real-time interaction, supports multitenancy and performs better in low-latency applications. The concept of fog computing has been designed to satisfy the applications that require low latency with a real-time response such as healthcare IoT systems.

Similarly, the performance of emergency and health monitoring services can be affected in terms of low latency, and also the delay that may be experienced while transferring data to the cloud f receiving the instructions back to the application. Healthcare applications provide large volumes of data which require storage in the cloud rather than depending on the limited computing resource and storage devices. The outcome data of healthcare applications is fairly large. In healthcare diagnosis, a large amount of data is generated, which should be

stored and retrieved in a perfect manner. Streaming-based transmissions in E-Health applications should be managed considering the real-time requirements. For designing healthcare applications, fog computing is considered the best method to rely on because these applications are latency sensitive, show low response time, and produce a large amount of data. Fog computing significantly contributes to healthcare applications by serving elderly people through home nursing. Real-time monitoring (e.g., neurological diseases) is one of the important features in healthcare applications that require a low latency and high response time, therefore fog computing can be the best solution for such applications.

One fog node or many computation nodes that are connected jointly can be used to build fog computing infrastructure. The connected fog computing nodes can significantly improve scalability, redundancy, and elasticity, and when more computing is required, it is possible to add more fog nodes. The above-mentioned characteristics conform to the requirement of healthcare applications. It is clear that one can rely on fog computing as it properly supports many healthcare applications because of its enhanced service quality, minimum response time, low latency, location awareness, high mobility, etc. However, fog nodes (e.g., smart routers, gateways, servers, base stations, etc.) cannot meet these requirements unless the architecture of fog nodes is redesigned to be compatible with healthcare applications.

1.1-What is Fog Computing?

The Internet of Things (IoT) supports billions of physical gadgets for data gathering and transmission to different administrations, like environmental monitoring, infrastructure control, and home automation. On the other hand, IoT possesses unsupported elements (e.g., with low latency, locality awareness, and geographic dissemination) that are significant for some IoT administration, including smart traffic lights, home energy control, and augmented reality. Several substantial devices are linked at an unequalled speed from the existence of the IoT. The relaying of information is possible due to the lining together of the devices inclusive of sensors, smart meters, mobile phones, smart automobiles, radio-frequency identification tags, personal digital assistants, and different gadgets. The broadening of IoT results in the production of enormous information (Big Data) that consumes large computing assets, cache memory, and transmission capability. Cisco expects that 50 billion devices will be associated with the Internet by 2020.

The extension technology for IoT is Cloud Computing (CC). Many users and large organizations have used this technology. Cloud computing can be defined by creating a group of computers and servers interconnected in a network using the Internet. Using cloud computing, the cloud data centre is far from the end-user that causes high latency, and the Cloud enables us to perform tasks corresponding to customer demand. There is a large number of available resources. However, even these could not be used effectively, as even distributed computing could not use it as efficiently as it could be used. This situation is accepted by the corporate sector and web application domain. However, it is not appropriate in the mobility domain, which requires low latency and fast response time since data needs to be accessed more quickly. Hence, there is a pressing need for this kind of access. Despite cloud computing providing migration solutions to tackle the certainty and privacy difficulties, there is still persistence of certainty and privacy difficulties because of its distinct characteristics like localized facilities, movement capability, locality recognition, and little inertia. Therefore, Cisco comes up with a new technology concept, termed Fog Computing (FC). FC provides excellent services compared to cloud computing, thanks to the restricted information storage and information dissemination serving end-users with cloud Centers absence. Fog nodes can act as representatives for the end-services to discharge reliable functions, where gadgets have a scarcity of sufficient resources.

In the world of technology where we are living in, almost all the devices are connected to internet. The number of IoT devices are increasing in an exponential rate and all these devices are relying on cloud computing system for data computation and storage. It becomes a bottle neck problem when it comes to real time data operation which is the major drawback in the existing IoT healthcare system. In order to overcome the problem Fog Computing concept has been introduced. Fog computing in an archetype that extends the cloud computing platform. Fog acts as a middle layer between the cloud server and the end devices. It is not the complete replacement of cloud, rather it complements the functionality of cloud. Fog works closer to the edge devices and provides computing resources to these devices. Fog computing overcomes the scalability and reliability issues which is there in the traditional IoT-cloud architecture. Since Fog nodes works at the edge side and more geographically distributed as in, it enhances data security, accuracy, consistency and reduces the latency rate which is an important factor for application like medical data. As well as the overall bandwidth to cloud is saved, thus achieving better quality of service (QoS).

Fog Computing is the term coined by Cisco that refers to extending cloud computing to an edge of the enterprise's network. Thus, it is also known as Edge Computing or Fogging. It facilitates the operation of computing, storage, and networking services between end devices and computing data Centres.

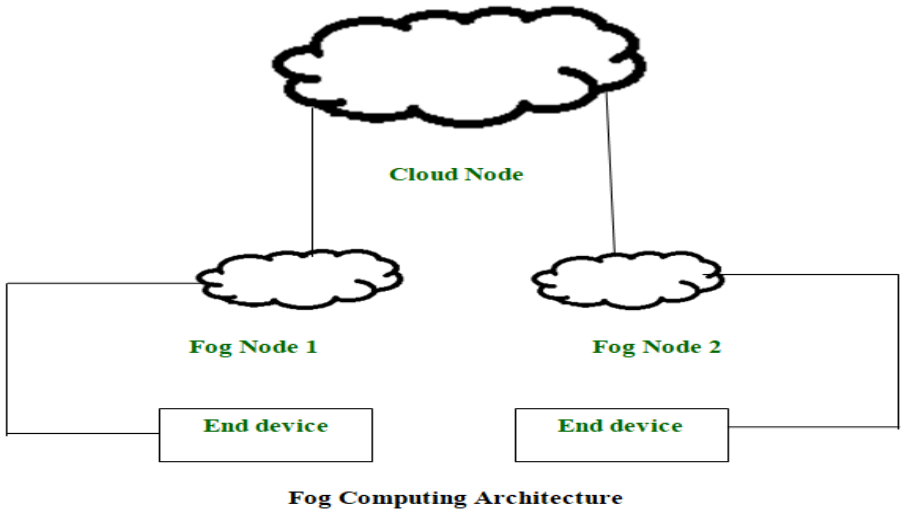


Fig.no:1.1 Fog computing Architecture

1. The devices comprising the fog infrastructure are known as fog nodes.
2. In fog computing, all the storage capabilities, computation capabilities, data along with the applications are placed between the cloud and the physical host.
3. All these functionalities are placed more towards the host. This makes processing faster as it is done almost at the place where data is created.
4. It improves the efficiency of the system and is also used to ensure increased security.

The rapid growth of wireless technology has given mobile device users tremendous computing power. No matter the industry vertical, today's enterprises see an outpouring of data from consumers. The internet of things (IOT) drives data-intensive customer experiences involving anything from smart electric grids to fitness trackers. Cloud computing and artificial intelligence allow for the dynamic processing and storage of these large amounts of data. This data enables organizations to make informed decisions and protect themselves from vulnerabilities at both, business and technological levels. This data explosion has, however, left organizations questioning the quality and quantity of data that they store in the cloud. Cloud

costs are notorious for escalating quickly, and sifting through petabytes of data makes real-time response difficult.

Let's consider the data sent by a temperature sensor in a factory line. The temperature recording can be pushed to the cloud every second with a service checking for fluctuations. But a more intelligent way of storing this information would be to check if there have been any temperature changes in the last few seconds.

When a temperature change is noticed, the data is pushed to the cloud for storage to verify the proper operation of the production line. The temperature may take up little space, but this kind of scenario is also common with devices such as CCTV cameras that produce large video and audio data. This small storage and computation of data before sending it over to the cloud is fog computing. Fog computing involves the usage of devices with lower processing capabilities to share some of the cloud's load. The goal of fog computing is to use the cloud only for long-term and resource-intensive analytics. These devices at the 'edge' of the cloud, i.e., where the organization's system interacts with the outside world, take care of short-term and time-critical analytics such as fault alerts, alarm status, etc.

Edge computing is a subset of fog computing that involves processing data right at the point of creation. Edge devices include routers, cameras, switches, embedded servers, sensors, and controllers. In edge computing, the data generated by these devices are stored and computed at the device itself, and the system doesn't look at sharing this data with the cloud.

Fog computing introduces a layer between edge devices and the cloud. This layer relies on a bunch of small computing servers that reside near the edge devices and not necessarily on the device itself. The servers are connected to each other and centralized cloud servers, enabling the intelligent flow of information. These small units work together to handle pre-processing of data, short-term storage, and rule-based real-time monitoring. The fog computing architecture reduces the amount of data transported through the system and improves overall efficiency.

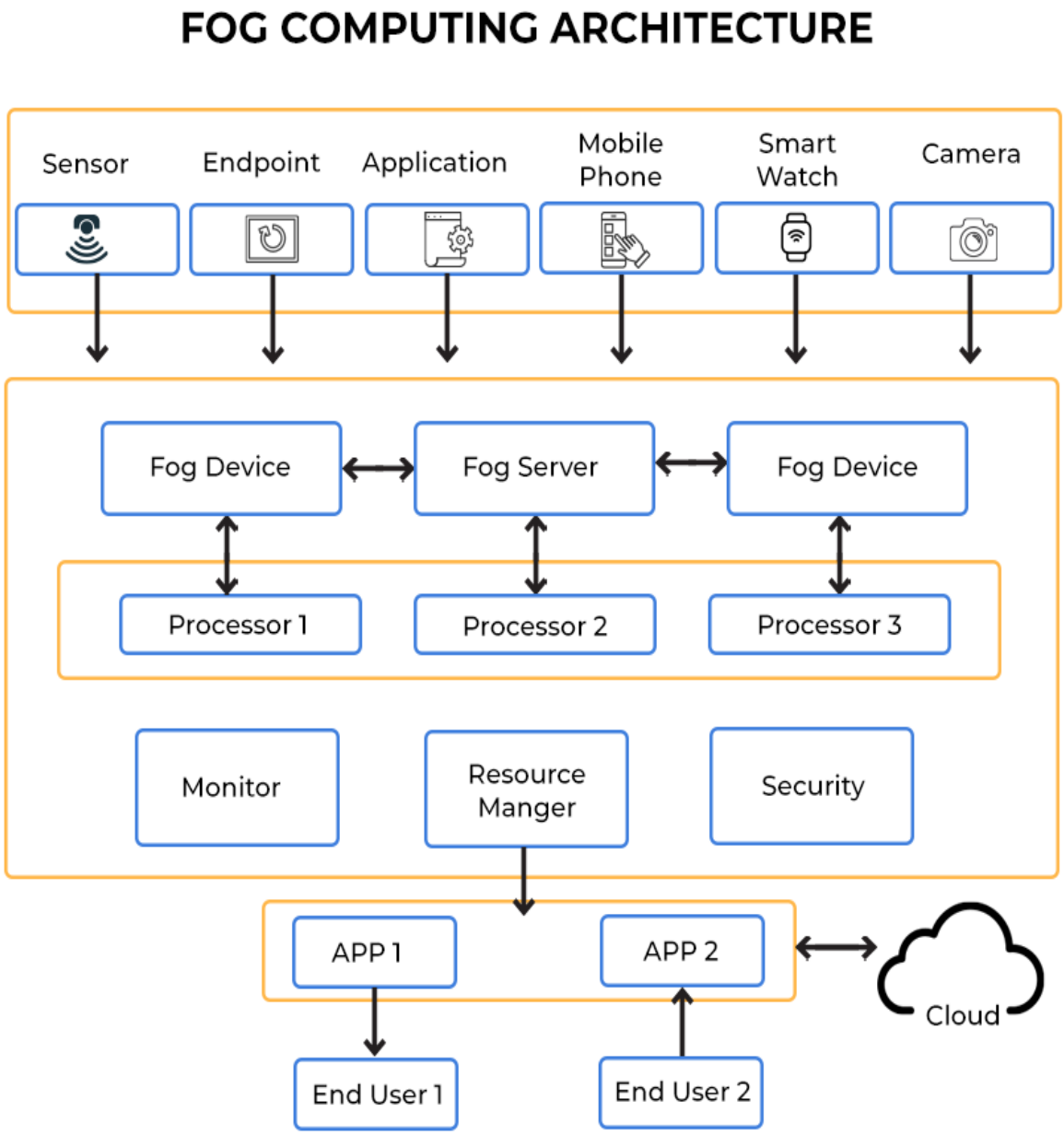


Fig.no:1.2 Fog commputing Architecture

1.2-History of Fog Computing:

The term fog computing was coined by Cisco in January 2014. This was because fog is referred to as clouds that are close to the ground in the same way fog computing was related to the nodes which are present near the nodes somewhere in between the host and the cloud. It was intended to bring the computational capabilities of the system close to the host

machine. After this gained a little popularity, IBM, in 2015, coined a similar term called "Edge Computing".

When to use Fog Computing?

1. It is used when only selected data is required to send to the cloud. This selected data is chosen for long-term storage and is less frequently accessed by the host.
2. It is used when the data should be analysed within a fraction of seconds i.e. Latency should be low.
3. It is used whenever a large number of services need to be provided over a large area at different geographical locations.
4. Devices that are subjected to rigorous computations and processing must use fog computing.
5. Real-world examples where fog computing is used are in IoT devices (e.g. Car-to-Car Consortium, Europe), Devices with Sensors, Cameras (IIOT-Industrial Internet of Things), etc.

1.3-Advantages of fog computing:

- This approach reduces the amount of data that needs to be sent to the cloud.
- Since the distance to be travelled by the data is reduced, it results in saving network bandwidth.
- Reduces the response time of the system.
- It improves the overall security of the system as the data resides close to the host.
- It provides better privacy as industries can perform analysis on their data locally.

1.4-Disadvantages of Fog Computing:

- Congestion may occur between the host and the fog node due to increased traffic (heavy data flow).
- Power consumption increases when another layer is placed between the host and the cloud.
- Scheduling tasks between host and fog nodes along with fog nodes and the cloud is difficult.

- Data management becomes tedious as along with the data stored and computed, the transmission of data involves encryption-decryption too which in turn release data.

1.5-Applications of fog computing

- It can be used to monitor and analyse the patients' condition. In case of emergency, doctors can be alerted.
- It can be used for real-time rail monitoring as for high-speed trains we want as little latency as possible.
- It can be used for gas and oils pipeline optimization. It generates a huge amount of data and it is inefficient to store all data into the cloud for analysis.

2. LITERATURE SURVEY

2. LITERATURE SURVEY

2.1- Fog computing in health: A systematic literature review

AUTHORS: Humberto Jorge de Moura Costa , Cristiano André da Costa,

Rodrigo da Rosa Righi & Rodolfo Stoffel Antunes

ABSTRACT: Study and finding out challenges and open questions of this area Currently, technology greatly benefits the area of healthcare. Modern computers can quickly process a large volume of patient health records. Due to recent advances in the area of Internet of Things and healthcare, patient data can be dispersed in multiple locations. As a result, scientists have been proposing solutions based on Cloud Computing to manage healthcare data.

However, such solutions present challenges regarding access latency, context-awareness, and large volumes of data. There is an increased probability of processing and transmission errors are more likely to occur as health data sets become larger and more complex. In this context, Fog Computing presents itself as an alternative to reduce health data management complexity, consequently increasing its reliability. To that end, it is important to comprehend the associated challenges before defining a Fog Computing-based architecture to manage healthcare data. This article presents a systematic literature review of fog computing being applied to healthcare area.

We propose a taxonomy to explore the open issues and most important challenges on these fields of study. We selected 1070 scientific articles published in the last 10 years, filtering the 44 most significant works for an in-depth analysis. We found that there are several challenges to be addressed such as interoperability, privacy, security, data processing, management of resources and Big Data issues. Also, our contribution includes developing a taxonomy for the Fog Computing and healthcare fields.

2.2 Fog Computing Service in the Healthcare Monitoring System for Managing the Real-Time Notification

AUTHORS: Ahmed Elhadad , Fulayjan Alanazi, Ahmed Taloba, and Amr Abozeid

ABSTRACT: A new computing paradigm that has been growing in computing systems is fog computing. In the healthcare industry, Internet of Things (IoT) driven fog computing is being developed to speed up the services for the general public and save billions of lives. This new computing platform, based on the fog computing paradigm, may reduce latency when transmitting and communicating signals with faraway servers, allowing medical services to be delivered more quickly in both spatial and temporal dimensions.

One of the necessary qualities of computing systems that can enable the completion of healthcare operations is latency reduction. Fog computing can provide reduced latency when compared to cloud computing due to the use of only low-end computers, mobile phones, and personal devices in fog computing. In this paper, a new framework for healthcare monitoring for managing real-time notification based on fog computing has been proposed.

The proposed system monitors the patient's body temperature, heart rate, and blood pressure values obtained from the sensors that are embedded into a wearable device and notifies the doctors or caregivers in real time if there occur any contradictions in the normal threshold value using the machine learning algorithms. The notification can also be set for the patients to alert them about the periodical medications or diet to be maintained by the patients. The cloud layer stores the big data into the cloud for future references for the hospitals and the researchers.

2.3-Evaluation and Quality Assurance of Fog Computing-Based IoT for Health Monitoring System

AUTHORS: FtikharAhmad, Xiaoqunliao

ABSTRACT: Computation and data sensitivity are the metrics of the current Internet of Things (IoT). In cloud data Centres, current analytics are often hosted and reported on suffering from high congestion, limited bandwidth, and security mechanisms. Various platforms are developed in the area of fog computing and thus implemented and assessed to run analytics on multiple devices, including IoT devices, in a distributed way.

Fog computing advances the paradigm of cloud computing on the network edge, introducing a number of options and facilities. Fog computing enhances the processing, verdicts, and interventions to occur through IoT devices and spreads only the necessary details. The ideas of fog computing based on IoT in healthcare frameworks are exploited by shaping the disseminated delegate layer of insight between sensor hubs and the cloud.

The cloud proposed a system adapted to overcome various challenges in omnipresent medical services frameworks, such as portability, energy efficiency, adaptability, and unwavering quality issues, by accepting the right to take care of certain weights of the sensor network and a distant medical service group. An overview of e-health monitoring system in the context of testing and quality assurance of fog computing is presented in this paper. Relevant papers were analyzed in a comprehensive way for the identification of relevant information. The study has compiled contributions of the existing methodologies, methods, and approaches in fog computing e-healthcare.

2.4- Fog computing: A taxonomy, systematic review, current trends and research challenges.

AUTHORS: Jagdeep Singh, Parminder Singh, Sukhpal Singh Gill.

ABSTRACT: There has been rapid development in the number of Internet of Things (IoT) connected nodes and devices in our daily life in recent times. With this increase in the number of devices, fog computing has become a well-established paradigm to optimize various key Quality of Service (QoS) requirements such as latency, bandwidth limitation, response time, scalability, privacy and security.

In this paper, we present a systematic literature review of fog computing. This review article aims to classify recently published studies and investigate the current status in the area of fog computing. In this work, we have discussed the important characteristics of fog computing frameworks and identified various issues related to its architectural design, QoS metrics, implementation details, applications and communication modes.

We have proposed taxonomy for fog computing frameworks based on the existing literature and compared the different research work based on taxonomy. Finally, various open research challenges and promising future directions are highlighted for further research in the area of fog computing.

2.5-A Systematic Survey on Fog and IoT Driven Healthcare: Open Challenges and Research Issues

AUTHORS: Aijaita Kashyap, Ashok Kumar, Ajay Kumar, Yumin Hu

ABSTRACT: Technological advancements have made it possible to monitor, diagnose, and treat patients remotely. The vital signs of patients can now be collected with the help of Internet of Things (IoT)-based wearable sensor devices and then uploaded on to a fog server for processing and access by physicians for recommending prescriptions and treating patients through the Internet of Medical Things (IoMT) devices.

This research presents the outcome of a survey conducted on healthcare integrated with fog computing and IoT to help researchers understand the techniques, technologies and performance parameters. A comparison of existing research focusing on technologies, procedures, and findings has been presented to investigate several aspects of fog computing in healthcare IoT-based systems, such as increased temporal complexity, storage capacity, scalability, bandwidth, and latency.

Additionally, strategies, tools, and sensors used in various diseases such as heart disease, chronic disease, chikungunya viral infection, blood pressure, body temperature, pulse rate, diabetes.

3. SYSTEM ANALYSIS

3. SYSTEM ANALYSIS

Fog computing in IoT health assessment involves analysing the system's architecture, data processing at the network edge, and efficient utilization of resources. It optimizes data transfer, enhances real-time processing for health monitoring, and minimizes latency in delivering critical insights. The analysis would consider factors like edge device capabilities, data security, and the seamless integration of fog computing into the broader health ecosystem.

In a system analysis of fog computing in IoT health assessment, key components include.

Edge Devices: Identify the types of IoT devices (e.g., wearable health trackers) involved in data collection. Assess their capabilities, communication protocols, and compatibility with fog computing.

Fog Nodes: Analyse the deployment of fog nodes at the network edge. Evaluate their processing power, storage capacity, and ability to perform real-time analytics for health data.

Communication Protocols: Examine the communication protocols between edge devices and fog nodes, ensuring efficient and secure data transfer. Consider protocols like MQTT or CoAP for lightweight, low-latency communication.

Data Processing: Evaluate how fog nodes process health data locally. Assess the algorithms used for real-time analytics, anomaly detection, and other health assessment tasks.

Resource Utilization: Analyze how fog computing optimizes resource utilization. This includes assessing the distribution of computing tasks between edge devices, fog nodes, and potential interaction with cloud resources.

Security Measures: Examine security protocols in place to protect sensitive health data. This includes encryption, access control mechanisms, and measures against potential cyber threats.

Latency Reduction: Assess how fog computing minimizes latency by processing critical health data closer to the source. This is crucial for real-time health monitoring and timely decision-making.

Scalability: Analyse the system's scalability to accommodate a growing number of IoT devices and increasing data volumes. Ensure that the fog computing infrastructure can adapt to the dynamic nature of IoT health applications.

Integration with Cloud: Evaluate how the fog computing system integrates with cloud services for storage, backup, and more extensive analytics.

Overall Performance: Conduct performance testing to ensure that the fog computing system meets health assessment requirements, considering factors such as respond.

3.1-EXISTING SYSTEM:

Currently fog uses the Decoy system as a security service from malicious attacker. Like in Cloud, Fog uses this method to trick the attacker by providing fake data when they try to extract the data. In the decoy system the user has to sign up then login, while logging in the system will ask security questions related to information given while signing up. So when an attacker tries to login her/she will be trapped with the question and the system will give back spurious file which is very such similar to the original file and when the attacker tries to download it will turn out to be a fake data. But there is chance that the attacker might guess the questions right. Therefore, this system is not a very good way of securing data.

The existing system in fog computing in IoT health involves the use of edge devices to collect and process data from various healthcare devices. However, it faces challenges such as limited processing power, high latency, and security vulnerabilities. One of the challenges in fog computing in IoT health is the scalability of the system, as it needs to handle a large amount of data from multiple devices. Another challenge is ensuring data privacy and security in a distributed environment.

3.1.1-DISADVANTAGES OF THE EXISTING SYSTEM

- ❖ In the decoy system the user has to sign up then login, while logging in the system will ask security questions related to information given while signing up
- ❖ It becomes a bottle neck problem when it comes to real time data operation which is the major drawback in the existing IoT healthcare system

3.2-PROPOSED SYSTEM: In the proposed system a three-type architecture model is considered. First layer will be the Edge devices which will collect the data and this data will be transferred to the middle layer. The middle layer will be the fog layer; encryption process of the collected data will be performed in this layer. The encrypted data from the middle layer will then be send to the third layer which is the cloud layer. In the cloud the final encrypted data will be permanently stored.

The proposed system in fog computing in IoT health aims to address the challenges of the existing system by introducing a hierarchical architecture that includes edge, fog, and cloud layers. This architecture enables efficient data processing, real-time decision-making, and secure communication. The proposed system in fog computing in IoT health brings significant improvements in scalability, security, and latency. Future research should focus on optimizing resource allocation, developing standardized protocols, and addressing privacy concerns to further enhance the system's capabilities.

3.2.1-ADVANTAGES OF PROPOSED SYSTEM

In order to overcome the problem Fog Computing concept has been introduced. Fog Computing in an archetype that extends the cloud computing platform. Fog acts as a middle layer between the cloud server and the end devices. It enhances data security, accuracy, consistency and reduces the latency rate which is an important factor for application like medical data. As well as the overall bandwidth to cloud is saved, thus achieving better quality of service (QoS).

Improved Performance: The proposed system improves overall system performance by reducing latency and enhancing scalability.

Enhanced Security: With the proposed improvements, data privacy and security in fog computing in IoT health are significantly enhanced.

Real-time Decision-making: The proposed system enables real-time decision-making by processing data closer to the edge devices.

4. SYSTEM REQUIREMENTS

4. SYSTEM REQUIREMENTS

This section elaborates on the functional requirements of the application. The SRS itself can be divided into module, each module having specifications.In order to carry out the project, the following hardware and software is required.

4.1-NON-FUNCTIONAL REQUIREMNTS

Data Processing and Analysis:

Description: Fog computing should be able to process and analyse data locally, near the edge devices, to reduce latency and improve real-time decision-making.

Example: Perform real-time analytics on sensor data to detect anomalies or patterns.

Resource Management:

Description: Efficient allocation and management of computing, storage, and networking resources at the edge devices.

Example: Dynamically allocate resources based on the workload and demand.

Interoperability:

Description: Support for interoperability between various devices, protocols, and communication technologies.

Example: Enable communication between devices using different communication protocols.

Security:

Description: Ensure the security of data, devices, and communication in the fog computing environment.

Example: Implement encryption, access controls, and authentication mechanisms.

Scalability:

Description: The ability of the fog computing system to scale up or down based on the number of connected devices and the volume of data.

Example: Easily add new edge devices to the network without compromising performance.

Fault Tolerance:

Description: The system should be resilient to failures at the edge devices and should be able to recover gracefully.

Example: Implement redundant communication paths to handle network failures.

Real-time Communication:

Description: Support for real-time communication and collaboration between edge devices and with the cloud.

Example: Enable low-latency communication for applications such as video streaming or telemedicine.

Application Deployment and Orchestration:

Description: Provide tools and mechanisms for deploying, managing, and orchestrating applications at the edge to streamline development and maintenance.

Location Awareness:

Description: Utilize location information to optimize resource allocation, data processing, and application behaviour based on the geographical context of edge devices.

Edge-to-Cloud Integration:

Description: Enable seamless integration between edge computing resources and cloud services for a unified and coordinated computing environment.

4.2- FUNCTIONAL REQUIREMNTS

4.2.1-HARDWARE REQUIREMENTS:

System : i5 core
Hard Disk : 40 GB.
Monitor : 15 VGA Colour.
Mouse : Logitech.
Ram : 8 GB.

4.2.2-SOFTWARE REQUIREMENTS:

Technology	**:**	Java 2 Standard Edition, JDBC
Web Server	**:**	Tomcat 7.0
Client-Side Technologies:		HTML, CSS, JavaScript
Server-Side Technologies:		Servlets, JSP
Data Base Server	**:**	MySQL
Editor	**:**	Netbeans8.

5. SYSTEM STUDY

5. SYSTEM STUDY

5.1-FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

5.2-FEASIBILITY ANALYSIS

Three key considerations involved in the feasibility analysis are

- **ECONOMICAL FEASIBILITY**
- **TECHNICAL FEASIBILITY**
- **SOCIAL FEASIBILITY**

5.2.1-ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

5.2.2-TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

5.2.3-SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of system.

6. SYSTEM DESIGN

6. SYSTEM DESIGN

6.1-DATA FLOW DIAGRAM

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.

2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.

3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.

4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

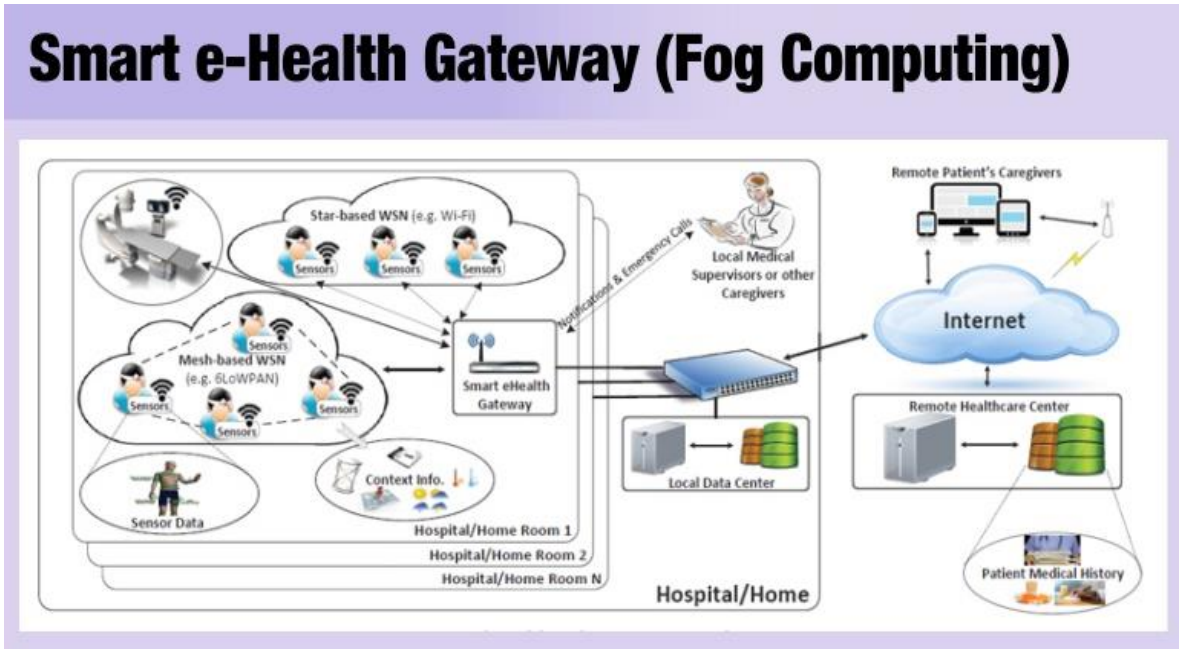


Fig.no:6.1 Smart e-Health Gateway

6.2-UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

6.3-USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

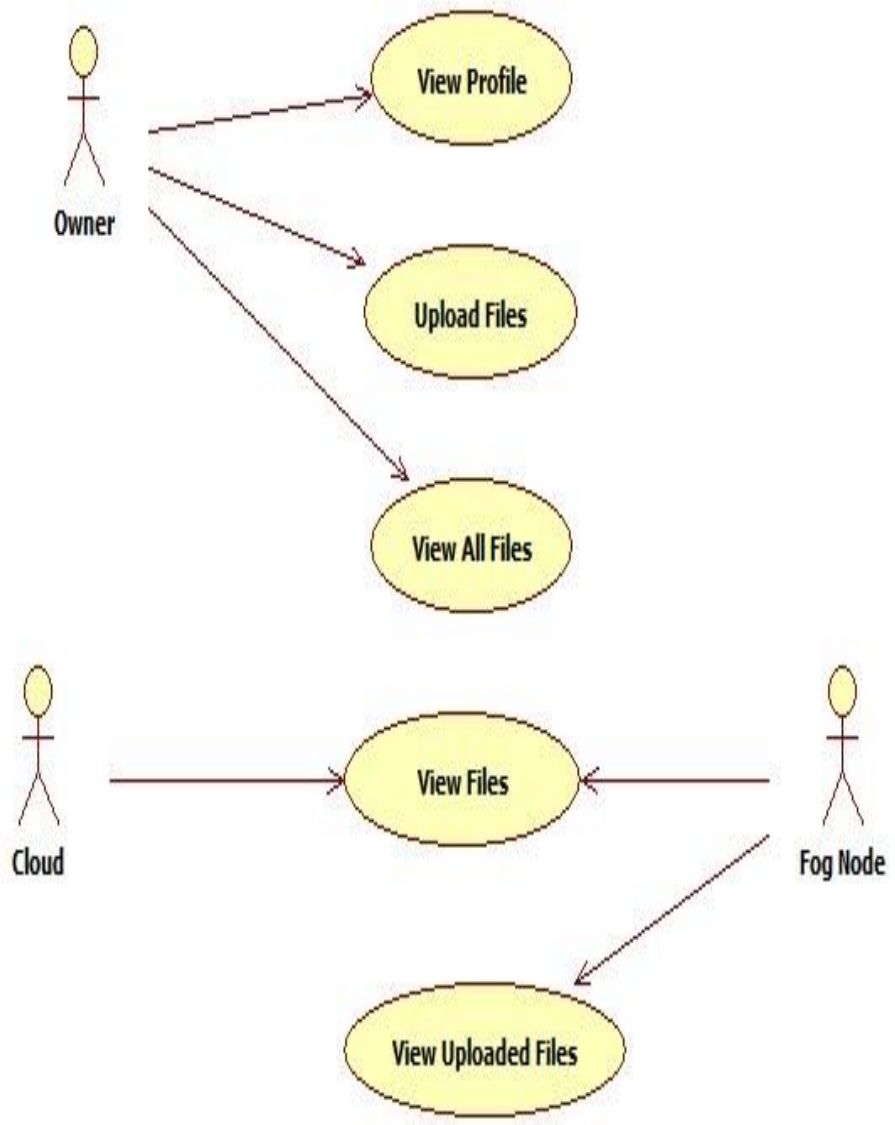


Fig 6.3 Use case Diagram

6.4-CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

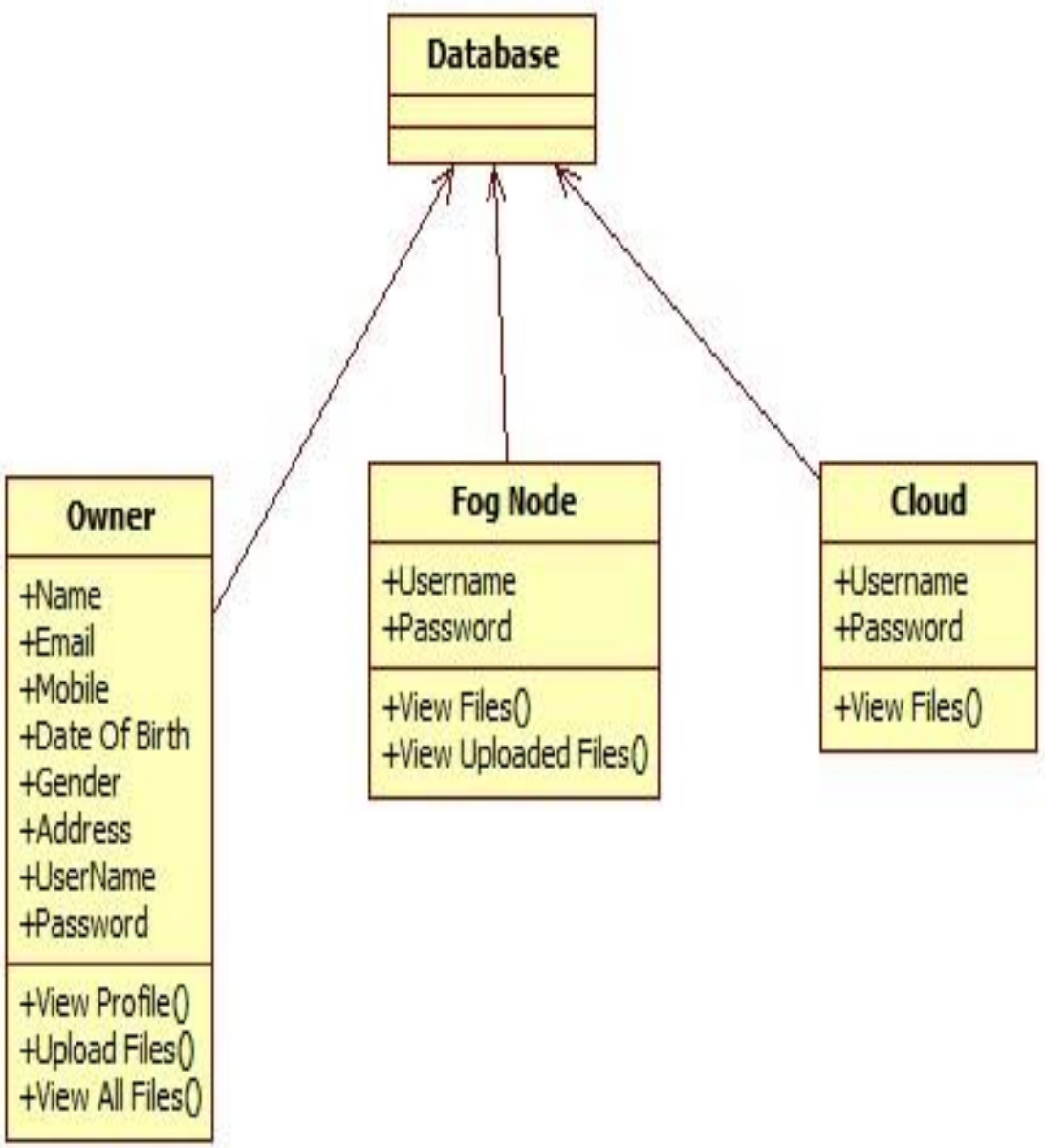


Fig 6.4 Class Diagram

6.5-SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

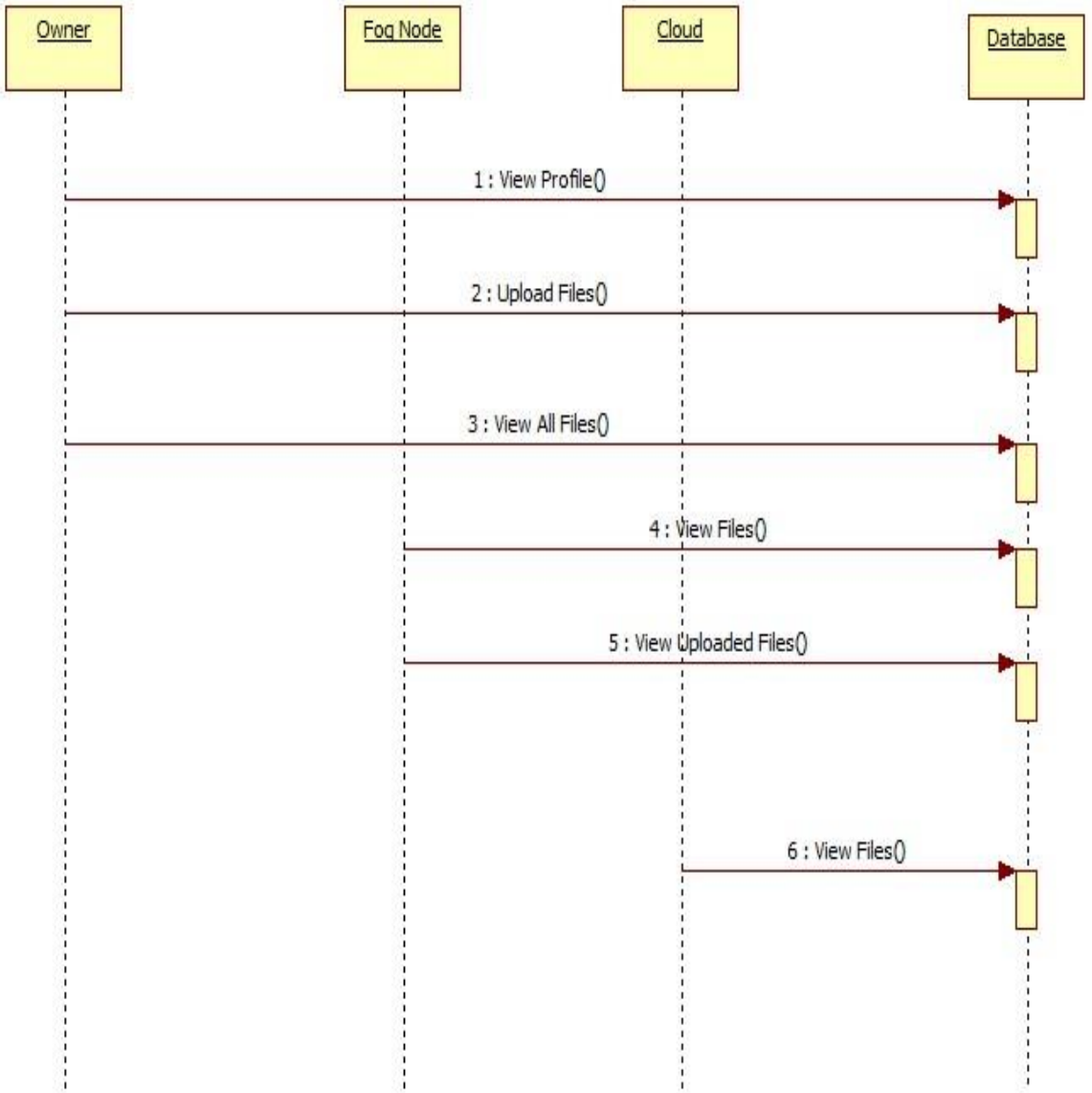


Fig 6.5 Sequence diagram

6.6-ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

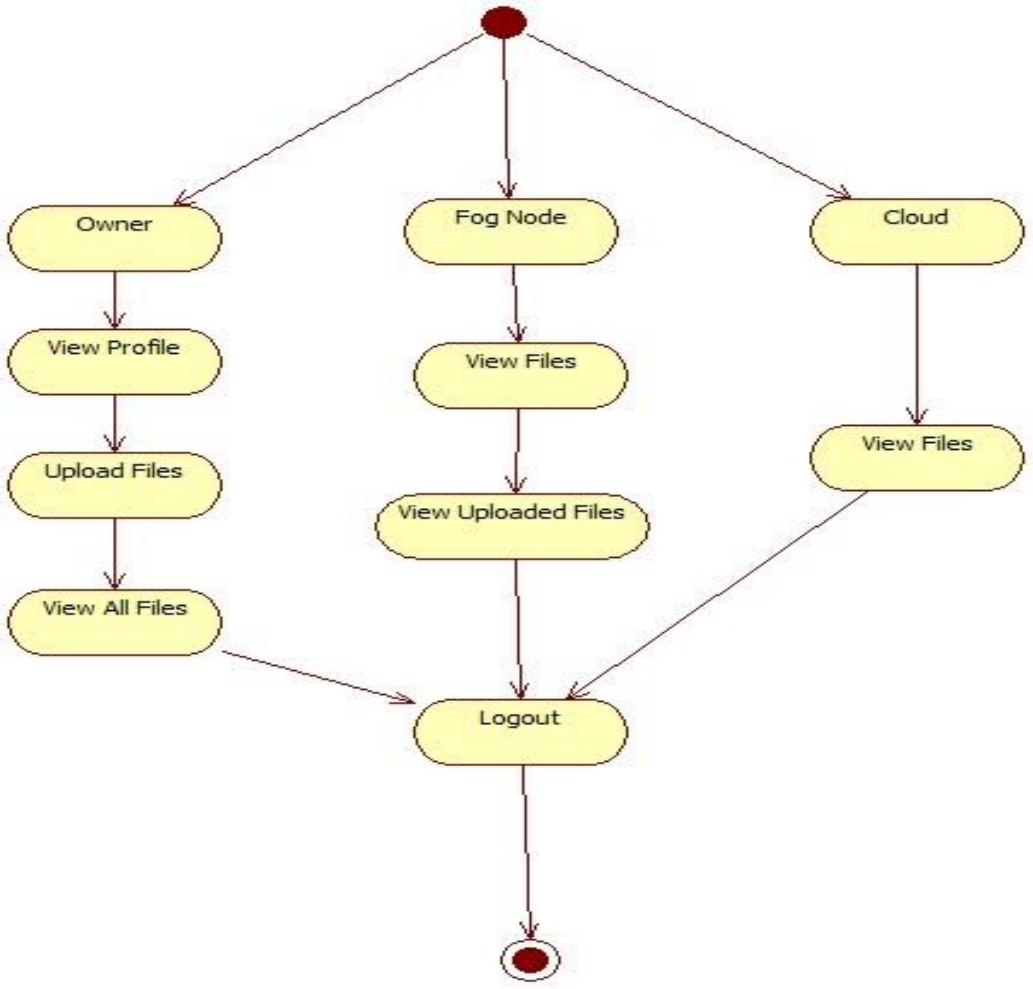


Fig 6.6 Activity Diagram

6.7-COMPONENT DIAGRAM:

A component diagram is used to break down a large object-oriented system into the smaller components, so as to make them more manageable. It models the physical view of a system such as executables, files, libraries, etc. that resides within the node.

It visualizes the relationships as well as the organization between the components present in the system. It helps in forming an executable system. A component is a single unit of the system, which is replaceable and executable. The implementation details of a component are hidden, and it necessitates an interface to execute a function. It is like a black box whose behavior is explained by the provided and required interfaces.

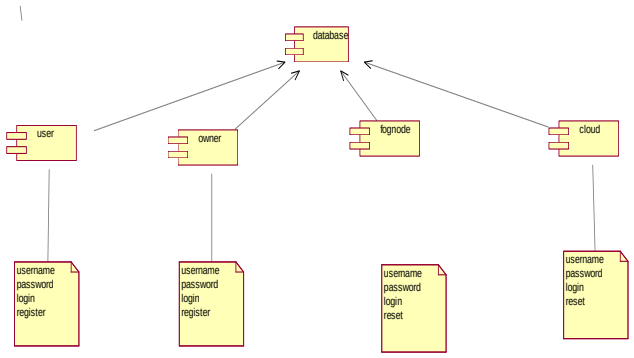


Fig 6.7 Component Diagram

6.8-DEPLOYMENT DIAGRAM:

The deployment diagram visualizes the physical hardware on which the software will be deployed. It portrays the static deployment view of a system. It involves the nodes and their relationships.

It ascertains how software is deployed on the hardware. It maps the software architecture created in design to the physical system architecture, where the software will be executed as a node. Since it involves many nodes, the relationship is shown by utilizing communication paths. The main purpose of the deployment diagram is to represent how software is installed on the hardware component. It depicts in what manner a software interacts with hardware to perform its execution.

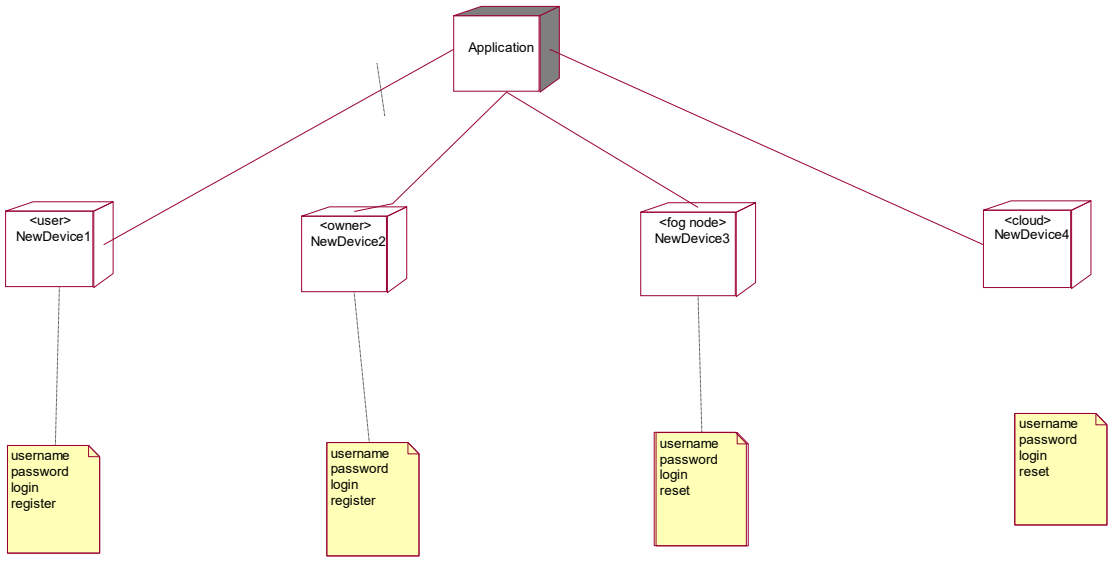


Fig 6.8 Deployment Diagram

6.9-DATA DICTIONARY DIAGRAM:

download	Field name	Data type	Description	Constraints
	Id	Int	Any document with name and information.	NOT NULL
	Email	varchar	It is a communication method that uses electronic devices to deliver messages across computer networks.	NULL
	Fname	varchar	It contains letters, digits, underscores, and dollar signs	NULL
	Data	varchar	It's structured information that describes content and makes it easier to find or use.	NULL
file	Id	Int	Any document with name and information.	NOT NULL
	fid	varchar	It specifies the name by which a user-defined function or function prototype is known and assigns selected attributes to that function or function prototype.	NULL

	Owner	varchar	A person who owns something.	NULL
	Fname	varchar	It contains letters, digits, underscores, and dollar signs	NULL
	Data	longtext	It's structured information that describes content and makes it easier to find or use.	NULL
	enc_data	longtext	Its converts data from a readable, plaintext format into an unreadable, encoded format: cipertext.	NULL
	Skey	varchar	A one-time password scheme based on a non-reversible cryptographic hash of a secret string.	NULL
	Status	varchar	The situation at a particular time during a process.	NULL
	Fdate	varchar	It allow programmes to exchange directly with a data base and modify object in-line	NULL
	cdate	varchar	The idea that source code written in a programming	NULL

			language can be manipulated as data.	
	Status2	varchar	The situation at a particular time during a process.	NULL
Owner	Id	Int	Any document with name and information.	NOT NULL
	Name	varchar	A word or words by which somebody/something is known.	NULL
	Email	varchar	It is a communication method that uses electronic devices to deliver messages across computer networks.	NULL
	Mobile	varchar	A protable telephone that can make and receive calls over a radio frequency link while the user is moving within a telephone service area.	NULL
	Dob	varchar	The date on which the person was born.	NULL
	gender	varchar	It refers to the characteristics of men, women, girls	NULL

			&boys that are socially constructed.	
	Address	varchar	The number of building and the name of the street and place where somebody lives or works.	NULL
	Username	varchar	An identification used by a person with access to a computer, network, online services.	NULL
	Password	varchar	It is a string of characters used to verify the identity of a user during the authentication process.	NULL
	Status	varchar	The situation at a particular time during a process.	NULL
user	Id	Int	Any document with name and information.	NOT NULL
	Name	varchar	A word or words by which somebody/something is known.	NULL
	Email	varchar	It is a communication method that uses	NULL

			electronic devices to deliver messages across computer networks.	
	Mobile	varchar	A protable telephone that can make and receive calls over a radio frequency link while the user is moving within a telephone service area.	NULL
	Dob	varchar	The date on which the person was born.	NULL
	Gender	varchar	It refers to the characteristics of men, women, girls &boys that are socially constructed.	NULL
	Address	varchar	The number of building and the name of the street and place where somebody lives or works.	NULL
	username	varchar	An identification used by a person with access to a computer, network, online services.	NULL
	Password	varchar	It is a string of characters used to	NULL

			verify the identity of a user during the authentication process.	
	Status	varchar	The situation at a particular time during a process.	NULL

Fig 6.9 Data dictionary Diagram

6.10-COLLABORATION:

The collaboration diagram is used to show the relationship between the objects in a system. Both the sequence and the collaboration diagrams represent the same information but differently. Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming. An object consists of several features. Multiple objects present in the system are connected to each other. The collaboration diagram, which is also known as a communication diagram, is used to portray the object's architecture in the system.

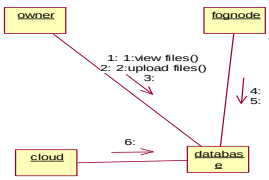


Fig 6.10 Collaboration diagram

7. INPUT AND OUTPUT DESIGN

7. INPUT AND OUTPUT DESIGN

7.1-INPUT DESIGN:

Considering the requirements, procedures to collect the necessary input data in most efficiently designed. The input design has been done keeping in view that, the interaction of the user with the system being the most effective and simplified way.

Also the measures are taken for the following

- Controlling the amount of input
- Avoid unauthorized access to the classroom.
- Eliminating extra steps
- Keeping the process simple
- At this stage the input forms and screens are designed.

One possible design of an IoT health assistance system using fog computing is as follows:

- The system consists of three layers: the IoT layer, the fog layer, and the cloud layer.
- The IoT layer includes various sensors and wearable devices that collect health data from the patients, such as body temperature, heart rate, blood pressure, blood glucose, etc. The IoT devices can communicate with each other and with the fog nodes using wireless protocols such as Bluetooth, Wi-Fi, ZigBee, etc.
- The fog layer consists of fog nodes that are distributed near the IoT devices, such as routers, gateways, access points, smartphones, etc. The fog nodes can perform data processing, analysis, aggregation, filtering, and encryption on the health data received from the IoT devices. The fog nodes can also provide feedback and notification to the patients and the caregivers based on the health status and the predefined rules. The fog nodes can communicate with the cloud servers using the Internet or other networks.

- The cloud layer provides storage, backup, and advanced analytics for the health data transmitted from the fog nodes. The cloud servers can also store the medical records, prescriptions, and recommendations for the patients. The cloud servers can provide access and visualization of the health data to the authorized users, such as doctors, nurses, researchers, etc.

7.2-OUTPUT DESIGN:

All the screens of the system are designed with a view to provide the user with easy operations in simpler and efficient way, minimum key strokes possible. Instructions and important information is emphasized on the screen. Almost every screen is provided with no error and important messages and option selection facilitates. Emphasis is given for speedy processing and speedy transaction between the screens. Each screen assigned to make it as much user friendly as possible by using interactive procedures. So to say user can operate the system without much help from the operating manual.

To design an output for IoT health assistances utilising fog computing, you need to consider the following aspects:

- The type and number of sensors that will be used to collect health data from the patients. The sensors should be wearable, non-invasive, and reliable. Some examples of sensors are temperature, heart rate, blood pressure, oxygen saturation, and electrocardiogram (ECG) sensors .
- The fog nodes that will be deployed at the edge of the network to process, aggregate, and store the health data locally. The fog nodes should be able to communicate with the sensors, the cloud, and other fog nodes. They should also have enough computing power, memory, and battery to perform data analysis, machine learning, and decision-making tasks .
- The cloud servers that will be used to store the health data in a centralized location, provide backup and recovery services, and perform advanced analytics and visualization. The cloud servers should be able to handle large volumes of data, provide security and privacy, and offer scalability and elasticity .
- The output format that will be used to display the health data to the users, such as patients, doctors, nurses, and caregivers. The output format should be user-friendly, intuitive, and informative.

8. MODULES

8. MODULES

8.1-MODULES:

The major modules of the project are

1.User

2.Owner

3.Fog Node

4.Cloud

8.2-MODULE DESIGN:

The major modules of the project are

1.**User**: In this module, there are n numbers of users are present. User should register before doing any operations. Once user registers, their details will be stored to the database. After registration successful, he has to login by using authorized user name and password. Once Login is successful user will do some operations like Home, User, Owner, Fog Node, Cloud.

2.**Owner:** In this module, there are n numbers of Owners are present. Owner should register before doing any operations. Once Owner registers, their details will be stored to the database. After registration successful, he has to login by using authorized user name and password. Once Login is successful Owner will do some operations like Home, User, Owner, Fog Node, Cloud.

3.**Fog Node:** In this module, the Server login by using valid user name and password. After login successful he can do some operations such as Home, View Files, View Uploads Files, Logout.

4.**Cloud:** In this module, the Server login by using valid user name and password. After login successful he can do some operations such as Home, View Files, View Uploads Files, Logout.

9. SOFTWARE ENVIRONMENT

9. SOFTWARE ENVIRONMENT

9.1-JAVA

9.1.1-HTML

Html is a language which is used to create web pages with html marking up a page to indicate its format, telling the web browser where you want a new line to begin or how you want text or images aligned and more are possible.

We used the following tags in our project.

Table:

Tables are so popular with web page authors is that they let you arrange the elements of a web page in such a way that the browser won't rearrange them web page authors frequently use tables to structure web pages.

TR:

TR is used to create a row in a table encloses <TH> and <TD> elements. <TR> contain many attributes. Some of them are,

- ALIGN: specifies the horizontal alignment of the text in the table row.
- BGCOLOR: Specifies the background color for the row.
- BORDERCOLOR: Sets the external border color for the row.
- VALIGN: Sets the vertical alignment of the data in this row.

TH:

TH is used to create table heading.

- ALIGN: Sets the horizontal alignment of the content in the table cell. Sets LEFT, RIGHT, CENTER.
- BACKGROUND: Species the back ground image for the table cell.
- BGCOLOR: Specifies the background color of the table cell
- VALIGN: Sets the vertical alignment of the data. Sets to TOP, MIDDLE, BOTTOM or BASELINE.

- WIDTH: Specifies the width of the cell. Set to a pixel width or a percentage of the display area.

TD:

TD is used to create table data that appears in the cells of a table.

- ALIGN: Species the horizontal alignment of content in the table cell. Sets to LEFT, CENTER, RIGHT.
- BGCOLOR: Specifies the background image for the table cell.
- BGCOLOR: sets the background color of the table cells.
- WIDTH: Species the width of the cell

9.1.2-Frames

Frames are used for either run off the page or display only small slices of what are supposed to be shown and to configure the frame we can use <FRAMESET>There are two important points to consider when working with <FRAMESET>.

- <FRAMESET> element actually takes the place of the <BODY> element in a document.
- Specifying actual pixel dimensions for frames .

<FRAME> Elements are used to create actual frames.

From the frameset point of view dividing the browser into tow vertical frames means creating two columns using the <FRAMESET> elements COLS attribute.

The syntax for vertical fragmentation is,

<FRAMESET COLS =”50%, 50%”>

</FRAMESET>

Similarly, if we replace COLS with ROWS then we get horizontal fragmentation.

The syntax for horizontal fragmentation is,

<FRAMESET ROWS=”50%, 50%”>

</FRAMESET>

Form:

The purpose of FORM is to create an HTML form; used to enclose HTML controls, like buttons and text fields.

9.1.3-Attribute

- ACTION: Gives the URL that will handle the form data.
- NAME: Gives the name to the form so you can reference it in code set to an alphanumeric string.
- METHOD: method or protocol is used to sending data to the target action URL. The GET method is the default, it is used to send all form name/value pair information in an URL. Using the POST method, the content of the form are encoded as with the GET method, but are sent in environment variables.

Controls in HTML:

<INPUT TYPE =BUTTON>:

Creates an html button in a form.

ATTRIBUTES:

- NAME: gives the element a name. Set to alphanumeric characters.
- SIZE: sets the size.
- VALUE: sets the caption of the element.

<INPUT TYPE = PASSWORD>:

Creates a password text field, which makes typed input.

ATTRIBUTES:

- NAME: gives the element a name, set to alphanumeric characters.
- VALUE: sets the default content of the element.

<INPUT TYPE=RADIO>:

Creates a radio button in a form.

ATTRIBUTE:

- NAME: Gives the element a name. Set to alphanumeric character.

- VALUE: Sets the default content of the element.

<INPUT TYPE=SUBMIT>:

Creates a submit button that the user can click to send data in the form back to the web server.

ATTRIBUTES:

NAME: Gives the element a name. Set to alphanumeric characters.

VALUE: Gives this button another label besides the default, Submit Query. Set to alphanumeric characters.

<INPUT TYPE=TEXT>:

Creates a text field that the user can enter or edit text in.

ATTRIBUTES:

NAME: Gives the element a name. Set to alphanumeric characters.

VALUE: Holds the initial text in the text field. Set to alphanumeric characters.

9.1.4-Java Script

Java script originally supported by Netscape navigator is the most popular web scripting language today. Java script lets you embedded programs right in your web pages and run these programs using the web browser. You place these programs in a <SCRIPT> element, usually within the <HEAD> element. If you want the script to write directly to the web page, place it in the <BODY> element.

Java script Methods:

Writeln:

Document writeln () is a method, which is used to write some text to the current web page.

On Click:

Occurs when an element is clicked.

On Load:

Occurs when the page loads.

On Mouse Down:

Occurs when a mouse button goes down.

On Mouse Move:

Occurs when the mouse moves.

On Unload:

Occurs when a page is unloaded.

9.1.5 MySQL

MySQL is an open-source relational database management system (RDBMS). This is the most popular database system used with PHP. MySQL is distributed and supported by Oracle Corporation.

MySQL runs on almost all platforms including Linux, Unix and Windows. Although it can be used in a wide range of applications, MySQL is often associated with web applications and online publishing.

MySQL is an essential constituent of an open-source enterprise stack called LAMP. LAMP is a web development platform that uses Linux as an operating system, in the form of Apache web server, MySQL relational database management system and PHP object-oriented scripting language.

Advantages of MySQL

Data Security: MySQL is globally renowned for being the most secure and reliable database management system used in popular web applications including WordPress, Drupal, Joomla, Facebook and Twitter.

High Performance: MySQL features a distinct storage-engine framework that facilitates system administrators to configure the MySQL database server for a flawless performance.

Round-the-Clock Up-time: MySQL comes with the assurance of 24×7 up-time and offers a wide range of high-availability solutions, including specialized cluster servers and master/slave replication configurations.

The Flexibility of Open Source: All the fears and worries that arise in an open-source solution can be brought to an end with MySQL's round-the-clock support and enterprise indemnification. The secure processing and trusted software of MySQL combine to provide effective transactions for large-volume projects. It makes maintenance, debugging and upgrades fast and easy while enhancing the end-user experience.

9.1.6-JDBC Drivers

The JDBC API only defines interfaces for objects used for performing various database-related tasks like opening and closing connections, executing SQL commands, and retrieving the results. We all write our programs to interfaces and not implementations. Either the resource manager vendor or a third party provides the implementation classes for the standard JDBC interfaces. These software implementations are called JDBC drivers. JDBC drivers transform the standard JDBC calls to the external resource manager-specific API calls. The diagram below depicts how a database client written in java accesses an external resource manager using the JDBC API.

DRIVER:

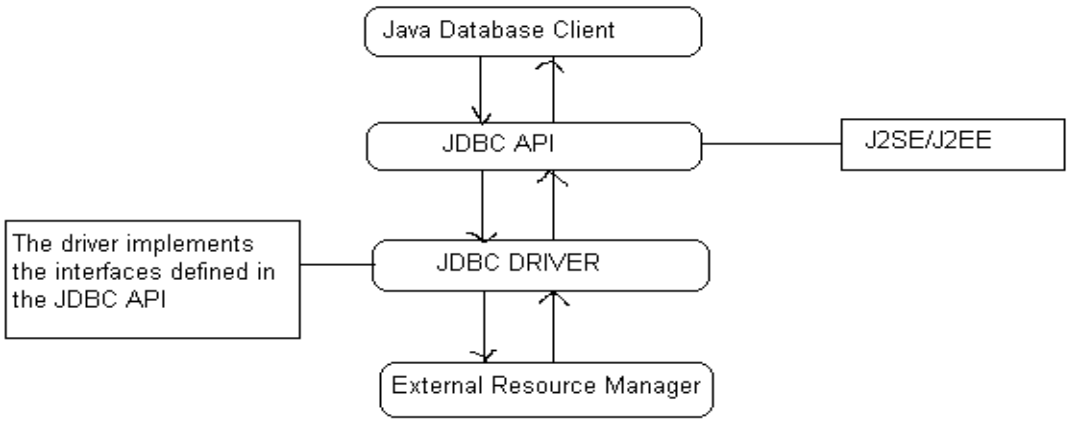


Fig.no 9.1.6 driver

Depending on the mechanism of implementation, JDBC drivers are broadly classified into four types.

TYPE1:

Type1 JDBC drivers implement the JDBC API on top of a lower-level API like ODBC. These drivers are not generally portable because of the independency on native libraries.

These drivers translate the JDBC calls to ODBC calls and ODBC sends the request to external data source using native library calls. The JDBC-ODBC driver that comes with the software distribution for J2SE is an example of a type1 driver.

TYPE2:

Type2 drivers are written in mixture of java and native code. Type2 drivers use vendors specific native APIs for accessing the data source. These drivers transform the JDBC calls to vendor specific calls using the vendor's native library.

These drivers are also not portable like type1 drivers because of the dependency on native code.

TYPE3:

Type3 drivers use an intermediate middleware server for accessing the external data sources. The calls to the middleware server are database independent. However, the middleware server makes vendor specific native calls for accessing the data source. In this case, the driver is purely written in java.

TYPE4:

Type4 drivers are written in pure java and implement the JDBC interfaces and translate the JDBC specific calls to vendor specific access calls. They implement the data transfer and network protocol for the target resource manager. Most of the leading database vendors provide type4 drivers for accessing their database servers.

9.1.7-DRIVER MANAGER AND DRIVER

The java. SQL package defines an interface called Java.sql.Driver that makes to be implemented by all the JDBC drivers and a class called java. SQL. Driver Manager that acts as the interface to the database clients for performing tasks like connecting to external resource

managers, and setting log streams. When a JDBC client requests the Driver Manager to make a connection to an external resource manager, it delegates the task to an approate driver class implemented by the JDBC driver provided either by the resource manager vendor or a third party.

JAVA.SQL.DRIVERMANAGER:

The primary task of the class driver manager is to manage the various JDBC drivers register. It also provides methods for:

- Getting connections to the databases.
- Managing JDBC logs.
- Setting login timeout.

Managing drivers:

JDBC clients specify the JDBC URL when they request a connection. The driver manager can find a driver that matches the request URL from the list of register drivers and delegate the connection request to that driver if it finds a match JDBC URLs normally take the following format:

<protocol>:<sub-protocol>:<resource>

The protocol is always jdbc and the sub-protocol and resource depend on the type of resource manager. The URL for postgreSQL is in the format:

Jdbc: postgres ://< host> :< port>/<database>

Here host is the host address on which post master is running and database is the name of the database to which the client wishes to connect.

Managing controls:

Driver Manager class is responsible for managing connections to the databases:

public static Connection get Connection (String URL, Properties info) throws SQL Exception

This method gets a connection to the database by the specified JDBC URL using the specified username and password. This method throws an instance of SQL Exception if a database access error occurs.

Connections:

The interface java. SQL. Connection defines the methods required for a persistent connection to the database. The JDBC driver vendor implements this interface. A database 'vendor-neutral' client never uses the implementation class and will always use only the interface. This interface defines methods for the following tasks:

- Statements, prepared statements, and callable statements are the different types of statements for issuing sql statements to the database by the JDBC clients.
- For getting and setting auto-commit mode.
- Getting meta information about the database.
- Committing and rolling back transactions.

Creating connections:

The interface java. SQL. Connection defines a set of methods for creating database statements. Database statements are used for sending SQL statements to the database:

Public Statement create Statement () throws SQL Exception

This method is used for creating instances of the interface java. SQL. Statement. This interface can be used for sending SQL statements to the database. The interface java. SQL. Statement is normally used for sending SQL statements that don't take any arguments. This method throws an instance of SQL Exception if a database access error occurs:

Public Statement create Statement (int retype, int reoccurrence) throws SQL Exception.

JDBC result set:

A JDBC result set represents a two-dimensional array of data produced as a result of executing SQL SELECT statements against databases using JDBC statements. JDBC result sets are represented by the interface java. SQL. Result Set. The JDBC vendor provider provides the implementation class for this interface.

Scrolling result set:

public Boolean next () throws SQL Exception

public Boolean previous () throws SQL Exception

public Boolean first () throws SQL Exception

public Boolean last () throws SQL Exception

Statement:

The interface java. SQL. Statement is normally used for sending SQL statements that do not have IN or OUT parameters. The JDBC driver vendor provides the implementation class for this interface. The common methods required by the different JDBC statements are defined in this interface. The methods defined by java. SQL. Statement can be broadly categorized as follows:

- Executing SQL statements
- Querying results and result sets
- Handling SQL batches
- Other miscellaneous methods

The interface java. SQL. Statements defines methods for executing different SQL statements like SELECT, UPDATE, INSERT, DELETE, and CREATE.

Public Result set execute Query (string SQL) throws SQL Exception.

The following figure shows how the Driver Manager, Driver, Connection, Statement, Result Set classes are connected.

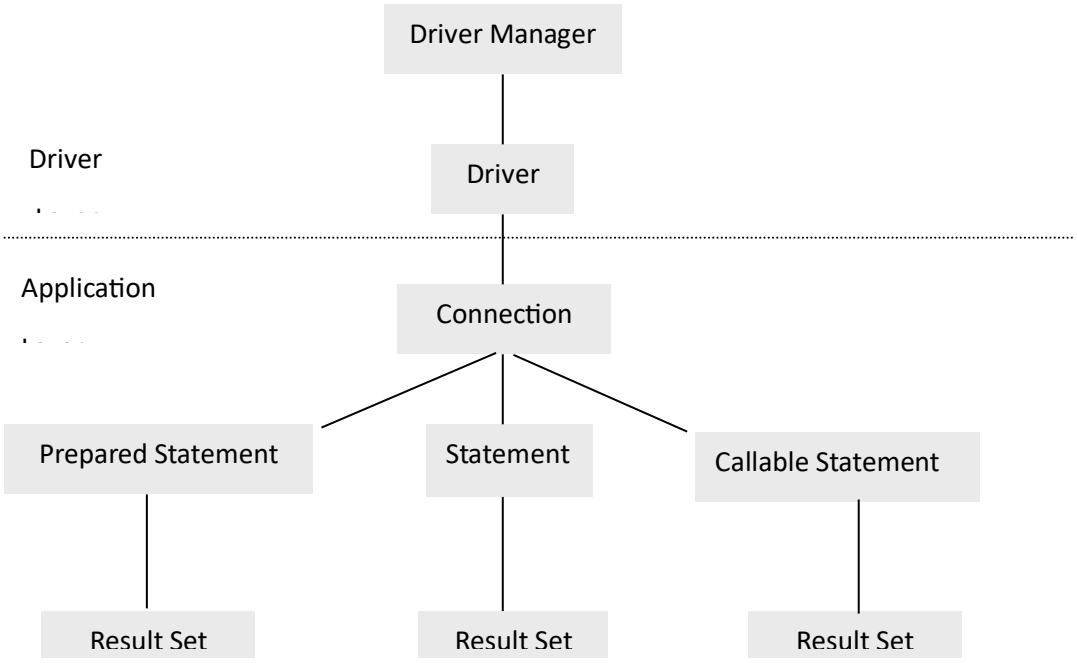


Fig.no 9.1.7 Driver manager

9.1.8-JAVA SERVER PAGES (JSP)

Introduction:

Java Server Pages (JSP) technology enables you to mix regular, static HTML with dynamically generated content. You simply write the regular HTML in the normal manner, using familiar Web-page-building tools. You then enclose the code for the dynamic parts in special tags, most of which start with <% and end with %>.

The need of JSP:

Servlets are indeed useful, and JSP by no means makes them obsolete. However,

- It is hard to write and maintain the HTML.
- You cannot use standard HTML tools.
- The HTML is inaccessible to non-Java developers.

Benefits of JSP:

JSP provides the following benefits over servlets alone:

- It is easier to write and maintain the HTML: In this no extra backslashes, no double quotes, and no lurking Java syntax.
- You can use standard Web-site development tools:

  We use Macromedia Dreamweaver for most of the JSP pages. Even HTML tools that know nothing about JSP can used because they simply ignore the JSP tags.

- You can divide up your development team:

  The Java programmers can work on the dynamic code. The Web developers can concatenate on the representation layer. On large projects, this division is very important. Depending on the size of your team and the complexity of your project, you can enforce a weaker or stronger separation between the static HTML and the dynamic content.

Creating template text:

A large percentage of our JSP document consists of static text known as template text. In almost all respects, this HTML looks just likes normal HTML follows all the same syntax rules, and simply "passed through" to that client by the servlet created to handle the page. Not only does the HTML look normal, it can be created by whatever tools you already are using for building Web pages.

There are two minor exceptions to the "template text passed through" rule. First, if you want to have <% 0r %> in the out port, you need to put <\% or %\> in the template text. Second, if you want a common to appear in the JSP page but not in the resultant document,

<%-- JSP Comment -- %>

HTML comments of the form:

<! —HTML Comment -->

are passed through to the client normally.

Types of JSP scrolling elements:

JSP scripting elements allow you to insert Java code into the servlet that will be generated from the JSP page. There are three forms:

1. **Expressions** of the form <%=Java Expression %>, which are evaluated and inserted into the servlet's output.
2. **Script lets** of the form <%Java code %>, which are inserted into the servlet's_jspService method (called by service).
3. **Declarations** of the form<%! Field/Method Declaration %>, which are inserted into the body of the servlet class, outside any existing methods.

Using JSP Expressions:

A JSP element is used to insert values directly into the output. It has the following form:

<%= Java Expression %>

The expression is evaluated, converted to a string, and inserted in the page. This evaluation is performed at runtime (when the page is requested) and thus has full access to the information about the request for example, the following shows the date/time that the page was requested.

Current time: <%=new java. util. Date () %>

Predefined variables:

To simplify expressions, we can use a number of predefined variables (or "implicit objects"). The specialty of these variables is that, the system simply tells what names it will use for the local variables in _jspService.The most important ones of these are:

- **request**, the HttpServletRequest.
- **response**, the HttpServletResponse.
- **session,** the HttpSession associated with the request
- **out,** the writer used to send output to clients.
- **application,** the ServletContext. This is a data structure shared by all servlets and JSP pages in the web application and is good for storing shared data.

Here is an example:

Your hostname: <%= **request.** getRemoteHost () %>

Comparing servlets to JSP pages:

JSP works best when the structure of the HTML page is fixed but the values at various places need to be computed dynamically. If the structure of the page is dynamic, JSP is less beneficial. Sometimes servlets are better in such a case. If the page consists of binary data or has little static content, servlets are clearly superior. Sometimes the answer is neither servlets nor JSP alone, but rather a combination of both.

Writing script lets:

If you want to do something more complex than output the value of a simple expression .JSP script let's let you insert arbitrary code into the servlet's _jspService method. Script lets have the following form:

<% Java code %>

Script lets have access to the same automatically defined variables as do expressions (request, response, session, out , etc) .So for example you want to explicitly send output of the resultant page , you could use the out variable , as in the following example:

```
<%
    String query Data = request. getQueryString ();
    out. println ("Attached GET data: "+ query Data);
%>
```

Script let Examples:

As an example of code that is too complex for a JSP expression alone, a JSP page that uses the bgColor request parameter to set the background colour of the page. Simply using

<BODY BGCOLOR=" <%= request. get Parameter ("bgcolor") %> ">

would violate the cardinal rule of reading form data.

Using declarations:

A JSP declaration lets you define methods or fields that get inserted into the main body of the servlet class. A declaration has the following form:

<%! Field or Method Definition %>

Since declarations do not generate output, they are normally used in conjunction with JSP expressions or script lets. In principle, JSP declarations can contain field (instance variable) definitions, method definitions, inner class definitions, or even static initializer blocks: anything that is legal to put inside a class definition but outside any existing methods. In practice declarations almost always contain field or method definitions.

We should not use JSP declarations to override the standard servlet life cycle methods. The servlet into which the JSP page gets translated already makes use of these methods. There is no need for declarations to gain access to service, do get, or do post, since calls to service are automatically dispatched to _jspService, which is where code resulting from expressions and script lets is put. However, for initialization and cleanup, we can use jspInit and jspDestroy-

the standard in it and destroy methods are guaranteed to call these methods in the servlets that come from JSP.

Jakarta Tomcat:

Tomcat is the Servlet/JSP container. Tomcat implements the Servlet 2.4 and Java Server Pages 2.0 specification. It also includes many additional features that make it a useful platform for developing and deploying web applications and web services.

Terminology:

Context – a Context is a web application.

$CATALINA_HOME – This represents the root of Tomcat installation.

Directions and files:

/Bin – Startup, shutdown, and other scripts. The *.sh files (for Unix systems) are functional duplicates of the *.bat files (for Windows systems). Since the Win32 command-line lacks certain functionality, there are some additional files in here.

/Conf – Configuration files and related DTDs. The most important file in here is server.xml. It is the main configuration file for the container.

/Logs – Log files are here by default.

/Webapps – This is where webapps go\

Installation:

Tomcat will operate under any Java Development Kit (JDK) environment that provides a JDK 1.2 (also known as Java2 Standard Edition, or J2SE) or later platform. JDK is needed so that servlets, other classes, and JSP pages can be compiled.

Deployment directions for default web applications:

HTML and JSP Files

- Main Location
  $CATALINA_HOME/webapps/ROOT

- Corresponding URLs.

  http://host/SomeFile.html

  http://host/SomeFile.jsp

- More Specific Location (Arbitrary Subdirectory).

  $CATALINA_HOME/webapps/ROOT/Some Directory

- Corresponding URLs

  http://host/SomeDirectory/SomeFile.html

  http://host/SomeDirectory/SomeFile.jsp

Individual Servlet and Utility Class Files

- Main Location (Classes without Packages).
- $CATALINA_HOME/webapps/ROOT/WEB-INF/classes

- Corresponding URL (Servlets).
- http://host/servlet/ServletName

- More Specific Location (Classes in Packages).
- $CATALINA_HOME/webapps/ROOT/WEB-INF/classes/package Name

- Corresponding URL (Servlets in Packages).
- http://host/servlet/packageName.ServletName

Servlet and Utility Class Files Bundled in JAR Files

- Location

  $CATALINA_HOME/webapps/ROOT/WEB-INF/lib

- Corresponding URLs (Servlets)

  http://host/servlet/ServletName

  http://host/servlet/packageName.ServletName

9.1.9-XAMPP

XAMPP:

XAMPP stands for Cross-Platform (X), Apache (A), MariaDB (M), PHP (P) and Perl (P).

It is simply a web server if we want to make a website or designing and make a working website then XAMPP is useful .it gives an environment of how server works.

1.It contains Apache, MySQL, FileZilla servers by which we can use them and helps us in login and logout sessions, cookies we give a good help in websites

2.Also, it has WordPress feature by which it contains many themes of websites which are popular and we can use them to make a website without using so much php coding, HTML, CSS etc.

3.How to use it: 1. if we are working on MySQL then we just on the server of MySQL and go to php admin page.

4.To work on php based web pages we just on the server and then, code on a notepad by using php pages.

10. IMPLEMENTATION

10. IMPLEMENTATION

SOURCE CODE:

10.2.1-C_View_Files.jsp:

```
<%--
    Document: OWNER
    Created on: NOV 10, 2023, 10:49:57 PM
    Author: Acer
--%>
<%@page import="java.sql. Result Set"%>
<%@page import="com. connection. Queries"%>
<%@page content Type="text/html" page Encoding="UTF-8"%>
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta name="description" content="">
    <meta name="author" content="">
 <title>Advancing Confidentiality</title>
   <! -- Bootstrap Core CSS -->
    <link href="vendor/bootstrap/css/bootstrap.min.css" rel="stylesheet">
    <! -- Theme CSS -->
```

```
<link href="css/clean-blog.min.css" rel="stylesheet">
<link href="table.css" rel="stylesheet">
<! -- Custom Fonts -->
<linkhref="vendor/font-awesome/css/font-awesome.min.css"rel="stylesheet" type="text/css">
<linkhref='https://fonts.googleapis.com/css?family=Lora:400,700,400italic,700italic' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Open+Sans:300italic,400italic,600italic,700italic,800italic,400,300,600,700,800' rel='stylesheet' type='text/css'>
<! -- HTML5 Shim and Respond.js IE8 support of HTML5 elements and media queries -->
<! -- WARNING: Respond.js doesn't work if you view the page via file:// -->
<! --[if lt IE 9]>
<script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>
<script src="https://oss.maxcdn.com/libs/respond.js/1.4.2/respond.min.js"></script>
<! [endif]-->
</head>
<body>
<! -- Navigation -->
<nav class="navbar navbar-default navbar-custom navbar-fixed-top">
<div class="container-fluid">
<! -- Brand and toggle get grouped for better mobile display -->
<div class="navbar-header page-scroll">
<button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#bs-example-navbar-collapse-1">
```

```
            <span class="sr-only">Toggle navigation</span>
            Menu <iclass="fa fa-bars"></i>
          </button>
          <a class="navbar-brand" href="index.html">Fog Computing</a>
        </div>
  <! -- Collect the nav links, forms, and other content for toggling -->
        <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
          <ul class="nav navbar-nav navbar-right">
            <li>
              <a href="Cloud_Home.jsp">Home</a>
            </li>
            <li>
              <a href="C_View_Files.jsp">View Files</a>
            </li>
           <li>
              <a href="C_View_Files_In_Chart.jsp">View Downloads In Chart</a>
            </li>
            <li>
              <a href="Cloud.jsp">Log out</a>
            </li>
          </ul>
        </div>
        <! -- /. navbar-collapse -->
      </div>
```

```
    <! -- /.container -->
  </nav>
  <! -- Page Header -->
  <! -- Set your background image for this header on the line below. -->
  <header class="intro-header" style="background-image: URL('img/home-bg.jpg')">
    <div class="container">
      <div class="row">
        <div class="col-lg-8 col-lg-offset-2 col-md-10 col-md-offset-1">
          <div class="site-heading">
            <h2>Advancing Confidentiality</h2>
            <hr class="small">
            <span class="subheading">In IoT Health Assistance Utilizing Fog Computing</span>
          </div>
        </div>
      </div>
    </div>
  </header>
 <! -- Main Content -->
  <div class="container">
    <div class="row">
      <div class="col-lg-8 col-lg-offset-2 col-md-10 col-md-offset-1">
        <div class="post-preview">
          <h2>All Available Files</h2>
```

```
</div>
<hr>
<div class="post-preview" style="border:1px solid black; margin-left:0px;">
<table border="1">
<tr>
<th>FILE ID</th>
<th>FILE NAME</th>
<th>DATA</th>
<th>CIPHER DATA</th>
<th>UPLOADED BY</th>
<th>UPLOAD DATE</th>
<th>STATUS</th>
</tr>
<%try {
String query="select *from file where status2='uploaded_to_cloud'";
Result Set r=Queries.getExecuteQuery(query);
while (r. next ()) {
String fid=r. gets String("fid");
String data=r. get String("data");
%>
<tr>
<td><%=fid%></td>
<td><%=r. gets String("fname") %></td>
<td><textareacols="20" rows="5"><%=data%></textarea></td>
```

```
                        <td><textareacols="20" rows="5"><%=
r. get String("enc_data") %></text area></td>
                        <td>Fog Node</td>

                        <td><%=r. get String("cdate") %></td>
                        <td><font color="red">Available</font></td>
                      </tr>
                  <%
                      }
                } catch (Exception e) {
                      out. println(e);
                      }%>
 </table>
                </div>
                  <div class="post-preview">
                </div>
              <div class="post-preview">
                </div>
              <hr>
            </div>
        </div>
    </div>
<hr>
<! -- jQuery -->
```

```
    <script src="vendor/jQuery/jquery.min.js"></script>
    <! -- Bootstrap Core JavaScript -->
    <script src="vendor/bootstrap/js/bootstrap.min.js"></script>
    <! -- Contact Form JavaScript -->
    <script src="js/jqBootstrapValidation.js"></script>
    <script src="js/contact_me.js"></script>
    <! -- Theme JavaScript -->
    <script src="js/clean-blog.min.js"></script>
</body>
</html>
```

10.2.2-C_View_Files_In_Chart.jsp

```
<%--
    Document: OWNER
    Created on: nov 11, 2023, 10:49:57 PM
    Author: Acer
--%>
<%@page content Type="text/html" page Encoding="UTF-8"%>
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta name="description" content="">
```

```
    <meta name="author" content="">
 <title>Advancing Confidentiality</title>
<! -- Bootstrap Core CSS -->
    <link href="vendor/bootstrap/css/bootstrap.min.css" rel="stylesheet">
<! -- Theme CSS -->
    <link href="css/clean-blog.min.css" rel="stylesheet">
<! -- Custom Fonts -->
    <link href="vendor/font-awesome/css/font-awesome.min.css" rel="stylesheet"
type="text/css">
    <linkhref='https://fonts.googleapis.com/css?family=Lora:400,700,400italic,700italic'
rel='stylesheet' type='text/css'>
    <link
href='https://fonts.googleapis.com/css?family=Open+Sans:300italic,400italic,600italic,700ita
lic,800italic,400,300,600,700,800' rel='stylesheet' type='text/css'>
   <! -- HTML5 Shim and Respond.js IE8 support of HTML5 elements and media queries -->
    <! -- WARNING: Respond.js doesn't work if you view the page via file:// -->
    <! --[if lt IE 9]>
        <script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>
        <script src="https://oss.maxcdn.com/libs/respond.js/1.4.2/respond.min.js"></script>
    <! [endif]-->
</head>
<body>
 <! -- Navigation -->
    <nav class="navbar navbar-default navbar-custom navbar-fixed-top">
        <div class="container-fluid">
```

```
<! -- Brand and toggle get grouped for better mobile display -->
<div class="navbar-header page-scroll">
  <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#bs-example-navbar-collapse-1">
    <span class="sr-only">Toggle navigation</span>
    Menu <i class="fa fa-bars"></i>
  </button>
  <a class="navbar-brand" href="index.html">Fog Computing</a>
</div>
<! -- Collect the nav links, forms, and other content for toggling -->
<div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
  <ul class="nav navbar-nav navbar-right">
    <li>
      <a href="Cloud_Home.jsp">Home</a>
    </li>
    <li>
      <a href="C_View_Files.jsp">View Files</a>
    </li>
     <li>
      <a href="C_View_Files_In_Chart.jsp">View Downloads In Chart</a>
    </li>
    <li>
      <a href="Cloud.jsp">Log out</a>
    </li>
```

```
            </ul>
          </div>
          <! -- /. navbar-collapse -->
        </div>
        <! -- /.container -->
    </nav>
<! -- Page Header -->
    <! -- Set your background image for this header on the line below. -->
    <header class="intro-header" style="background-image: url('img/home-bg.jpg')">
        <div class="container">
            <div class="row">
                <div class="col-lg-8 col-lg-offset-2 col-md-10 col-md-offset-1">
                    <div class="site-heading">
                        <h2>Advancing Confidentiality</h2>
                        <hr class="small">
                        <span class="subheading">In IoT Health Assistance Utilizing Fog Computing</span>
                    </div>
                </div>
            </div>
        </div>
    </head>
<! -- Main Content -->
    <div class="container">
```

```
        <div class="row">
          <div class="col-lg-8 col-lg-offset-2 col-md-10 col-md-offset-1">
            <div class="post-preview">
              <h2>Welcome to Cloud Page </h2>
            </div>
            <hr>
            <div class="post-preview">
              <h4>View Downloads in Chart</h4>
                <iframe src="ViewDownload" width="600" height="420"></iframe>
             </div>
          <div class="post-preview">
             </div>
        <div class="post-preview">
            </div>
            <hr>
          </div>
        </div>
  <hr>
<! -- jQuery -->
    <script src="vendor/jquery/jquery.min.js"></script>
<! -- Bootstrap Core JavaScript -->
    <script src="vendor/bootstrap/js/bootstrap.min.js"></script>
<! -- Contact Form JavaScript -->
    <script src="js/jqBootstrapValidation.js"></script>
```

```
    <script src="js/contact_me.js"></script>
 <! -- Theme JavaScript -->
    <script src="js/clean-blog.min.js"></script>
</body>
</html>
```

10.2.3-Cloud.jsp

```
<%@page content Type="text/html" page Encoding="UTF-8"%>
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta name="description" content="">
    <meta name="author" content="">
 <title>Advancing Confidentiality</title>
<! -- Bootstrap Core CSS -->
    <link href="vendor/bootstrap/css/bootstrap.min.css" rel="stylesheet">
<! -- Theme CSS -->
    <link href="css/clean-blog.min.css" rel="stylesheet">
<! -- Custom Fonts -->
```

```
<linkhref="vendor/font-awesome/css/font-awesome.min.css"rel="stylesheet"
type="text/css">

<linkhref='https://fonts.googleapis.com/css?family=Lora:400,700,400italic,700italic'
rel='stylesheet' type='text/css'>

<linkhref='https://fonts.googleapis.com/css?family=Open+Sans:300italic,400italic,600italic,7
00italic,800italic,400,300,600,700,800' rel='stylesheet' type='text/css'>

<! -- HTML5 Shim and Respond.js IE8 support of HTML5 elements and media queries -->

<! -- WARNING: Respond.js doesn't work if you view the page via file:// -->

<! --[if lt IE 9]>

<script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>

<script src="https://oss.maxcdn.com/libs/respond.js/1.4.2/respond.min.js"></script>

<! [endif]-->

</head>

<body>

<! -- Navigation -->

<nav class="navbar navbar-default navbar-custom navbar-fixed-top">

<div class="container-fluid">

<! -- Brand and toggle get grouped for better mobile display -->

<div class="navbar-header page-scroll">

<buttontype="button"class="navbar-toggle"data-toggle="collapse"data-
target="#bs-example-navbar-collapse-1">

<span class="sr-only">Toggle navigation</span>
```

```
            Menu <iclass="fa fa-bars"></i>
          </button>
          <a class="navbar-brand" href="index.html">Fog Computing</a>
        </div>
  <! -- Collect the nav links, forms, and other content for toggling -->
        <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
          <ul class="nav navbar-nav navbar-right">
            <li>
              <a href="index.html">Home</a>
            </li>
             <li>
              <a href="User.jsp">User</a>
            </li>
            <li>
              <a href="OWNER.jsp">OWNER</a>
            </li>
            <li>
              <a href="FogNode.jsp">Fog Node</a>
            </li>
            <li>
              <a href="Cloud.jsp">Cloud</a>
```

```
                </li>
              </ul>
            </div>
            <! -- /. navbar-collapse -->
        </div>
        <! -- /.container -->
    </nav>
<! -- Page Header -->
    <! -- Set your background image for this header on the line below. -->
    <header class="intro-header" style="background-image: url('img/home-bg.jpg')">
        <div class="container">
            <div class="row">
                <div class="col-lg-8 col-lg-offset-2 col-md-10 col-md-offset-1">
                    <div class="site-heading">
                        <h2>Advancing Confidentiality</h2>
                        <hr class="small">
                        <span class="subheading">In IoT Health Assistance Utilizing Fog Computing</span>
                    </div>
                </div>
            </div>
```

```
      </div>
   </header>
<! -- Main Content -->
   <div class="container">
      <div class="row">
         <div class="col-lg-8 col-lg-offset-2 col-md-10 col-md-offset-1">
            <div class="post-preview">
              Cloud Login
            </div>
            <hr>
            <div class="post-preview">
              <form action="CloudAction.jsp" method="post">
                                        <table>
                        <tr><th>UserName:</th><td><input type="text" name="uname"
required=""/></td></tr>
                      <tr><th>Password:</th><td><input type="password" name="pwd"
required=""/></td></tr>
                    <tr><th></th><td><input type="submit" value="Login">
                      <input type="reset" value="Reset"></td></tr>
              </table>
                 </form>
            </div>
```

```
            <hr>
            <div class="post-preview">
        </div>
        <hr>
         <div class="post-preview">
         </div>
            <hr>
           </div>
       </div>
    </div>
<hr>
<! -- jQuery -->
    <script src="vendor/jQuery/jquery.min.js"></script>
<! -- Bootstrap Core JavaScript -->
    <script src="vendor/bootstrap/js/bootstrap.min.js"></script>
<! -- Contact Form JavaScript -->
    <script src="js/jqBootstrapValidation.js"></script>
    <script src="js/contact_me.js"></script>
<! -- Theme JavaScript -->
    <script src="js/clean-blog.min.js"></script>
</body>
```

```
</html>
```

10.2.4-CloudAction.jsp:

```
<%@page import="java. SQL. Result Set"%>

<%@page import="com. connection. Queries"%>

<%@page content Type="text/html" page Encoding="UTF-8"%>

<%

String uname=request. get Parameter("uname");

String pwd=request. get Parameter("pwd"); try {

  if (uname. Equals("Cloud”) && pwd. equals("Cloud")) { %>

    <script type="text/java script">

       window. Alert ("Login Success...!!");

       window. Location="Cloud_Home.jsp";

    </script> <%

 } else {

  %>

    <script type="text/java script">

window. Alert ("Login Access Failed...!!");

       window. Location="Cloud.jsp";

    </script> <%   }} catch (Exception e) {

 out. Println (e);

}
```

%>

10.2.5-Cloud_Home.jsp:

```
<%@page content Type="text/html" page Encoding="UTF-8"%>
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1">
   <meta name="description" content="">
   <meta name="author" content="">
<title>Advancing Confidentiality</title>
<! -- Bootstrap Core CSS -->
   <link href="vendor/bootstrap/CSS/bootstrap.min.css" rel="stylesheet">
<! -- Theme CSS -->
   <link href="CSS/clean-blog.min.css" rel="stylesheet">
<! -- Custom Fonts -->
   <linkhref="vendor/font-awesome/CSS/font-awesome.min.css"rel="stylesheet"
type="text/CSS">
   <linkhref='https://fonts.googleapis.com/css?family=Lora:400,700,400italic,700italic'
rel='stylesheet' type='text/CSS'>
   <link
href='https://fonts.googleapis.com/css?family=Open+Sans:300italic,400italic,600italic,700ita
lic,800italic,400,300,600,700,800' rel='stylesheet' type='text/CSS'>
<! -- HTML5 Shim and Respond.js IE8 support of HTML5 elements and media queries -->
```

```
  <! -- WARNING: Respond.js doesn't work if you view the page via file:// -->
  <! --[if lt IE 9]>
    <script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>
    <script src="https://oss.maxcdn.com/libs/respond.js/1.4.2/respond.min.js"></script>
  <! [endif]-->
</head>
<body>
 <! -- Navigation -->
  <nav class="navbar navbar-default navbar-custom navbar-fixed-top">
    <div class="container-fluid">
      <! -- Brand and toggle get grouped for better mobile display -->
      <div class="navbar-header page-scroll">
        <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#bs-example-navbar-collapse-1">
          <span class="sr-only">Toggle navigation</span>
          Menu <i class="fa fa-bars"></i>
        </button>
        <a class="navbar-brand" href="index.html">Fog Computing</a>
      </div>
<! -- Collect the nav links, forms, and other content for toggling -->
      <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
        <ul class="nav navbar-nav navbar-right">
         <li>
            <a href="Cloud_Home.jsp">Home</a>
```

```
            </li>
            <li>
              <a href="C_View_Files.jsp">View Files</a>
            </li>
             <li>
              <a href="C_View_Files_In_Chart.jsp">View Downloads In Chart</a>
            </li>
             <li>
              <a href="Cloud.jsp">Log out</a>
            </li>
          </ul>
        </div>
        <! -- /. navbar-collapse -->
    </div>
    <! -- /.container -->
</nav>
<! -- Page Header -->
<! -- Set your background image for this header on the line below. -->
<header class="intro-header" style="background-image: URL('img/home-bg.jpg')">
    <div class="container">
   <div class="row">
          <div class="col-lg-8 col-lg-offset-2 col-md-10 col-md-offset-1">
            <div class="site-heading">
              <h2>Advancing Confidentiality</h2>
```

```
                <hr class="small">
                <span class="subheading">In IoT Health Assistance Utilizing Fog Computing</span>
              </div>
            </div>
          </div>
        </div>
      </header>
<! -- Main Content -->
    <div class="container">
      <div class="row">
        <div class="col-lg-8 col-lg-offset-2 col-md-10 col-md-offset-1">
          <div class="post-preview">
            <h2>Welcome to Cloud Home Page </h2>
          </div>
          <hr>
          <div class="post-preview">
            <p>Here the Cloud Can view The Data Which encrypted and Uploaded From The Fog Node.</p>
    </div>
      <div class="post-preview">
        </div>
     <div class="post-preview">
          </div>
```

```
            <hr>
           </div>
       </div>
    </div>
  <hr>
 <! -- jQuery -->
    <script src="vendor/jQuery/jquery.min.js"></script>
 <! -- Bootstrap Core JavaScript -->
    <script src="vendor/bootstrap/js/bootstrap.min.js"></script>
<! -- Contact Form JavaScript -->
    <script src="js/jqBootstrapValidation.js"></script>
    <script src="js/contact_me.js"></script>
<! -- Theme JavaScript -->
    <script src="js/clean-blog.min.js"></script>
</body>
</html>
```

10.2.6-Download.jsp:

```
<%@page content Type="text/html" page Encoding="UTF-8"%>
<!DOCTYPE html>
<%
String fname=request. get Parameter("fname");
session. set Attribute ("fname", fname);
response. send Redirect ("Download2?"+fname); %>
```

11. RESULTS/DISCUSSIONS

11. RESULTS/DISCUSSIONS

11.1-SYSTEM TEST:

What do you mean by software testing?

Testing involves operation of a system or application under controlled conditions and evaluating the results. The controlled conditions should include both normal and abnormal conditions. Testing should intentionally attempt to make things go wrong to determine if things happen when they shouldn't or things don't happen when they should. It is oriented to 'detection'. The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

11.2-TYPES OF TESTING:

11.2.1- Unit Testing:

Unit testing is a software development process in which the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation. Unit testing is often automated but it can also be done manually. This testing mode is a component of Extreme Programming (XP), a pragmatic method of software development that takes a meticulous approach to building a product by means of continual testing and revision. Unit tests are written from a programmer's perspective. They ensure that a particular method of a class successfully performs a set of specific tasks. Each test confirms that a method produces the expected output when given a known input.

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit

tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

11.2.2-Performance Testing:

Performance testing is the process of determining the speed or effectiveness of a computer, network, software program or device. This process can involve quantitative tests done in a lab, such as measuring the response time or the number of MIPS (millions of instructions per second) at which a system functions. Qualitative attributes such as Reliability, scalability and interoperability may also be evaluated. Performance testing is often done in conjunction with stress testing.

Performance testing can verify that a system meets the specifications claimed by its manufacturer or vendor. The process can compare two or more devices or programs in terms of parameters such as speed, data transfer rate, bandwidth, throughput, efficiency or reliability.

Performance testing can also be used as a diagnostic aid in locating communications bottlenecks. Often a system will work much better if a problem is resolved at a single point or in a single component. For example, even the fastest computer will function poorly on today's Web if the connection occurs at only 40 to 50 Kbps (kilobits per second).

11.2.3-Integration Testing:

Integration testing, also known as integration and testing (I&T), is a software development process which program units are combined and tested as groups in multiple ways. In this context, a unit is defined as the smallest testable part of an application. Integration testing can expose problems with the interfaces among program components before trouble occurs in real-world program execution. Integration testing is a component of Extreme Programming (XP), a pragmatic method of software development that takes a meticulous approach to building a product by means of continual testing and revision. Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration

tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

11.2.4-Functional testing:

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centred on the following items:

- Valid Input: identified classes of valid input must be accepted.
- Invalid Input: identified classes of invalid input must be rejected.
- Functions: identified functions must be exercised.
- Output: identified classes of application outputs must be exercised.
- Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

11.2.5-System Testing:

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

11.2.6-White Box Testing:

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

11.2.7 Black Box Testing:

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document.

11.2.8 Test cases:

Test case for Login form:

FUNCTION:	**LOGIN**
EXPECTED RESULTS:	Should Validate the user and check his existence in database
ACTUAL RESULTS:	Validate the user and checking the user against the database
LOW PRIORITY	**No**
HIGH PRIORITY	**Yes**

Test case2:

Test case for User Registration form:

FUNCTION:	**USER REGISTRATION**
EXPECTED RESULTS:	Should check if all the fields are filled by the user and saving the user to database.
ACTUAL RESULTS:	Checking whether all the fields are field by user or not through validations and saving user.
LOW PRIORITY	**No**
HIGH PRIORITY	**Yes**

Test case3:

Test case for Change Password:

When the old password does not match with the new password, then this results in displaying an error message as " OLD PASSWORD DOES NOT MATCH WITH THE NEW PASSWORD".

FUNCTION:	Change Password
EXPECTED RESULTS:	Should check if old password and new password fields are filled by the user and saving the user to database.
ACTUAL RESULTS:	Checking whether all the fields are field by user or not through validations and saving user.
LOW PRIORITY	No
HIGH PRIORITY	Yes

Test case 4:

Test case for Forget Password:

When a user forgets his password, he is asked to enter Login name, ZIP code, Mobile number. If these are matched with the already stored ones then user will get his original password.

Module	Functionality	Test Case	Expected Results	Actual Results	Result	Priority
User	Login Use case	1. Navigate To Www.Sample.Com 2. 2.Click on Submit Button Without Entering Username and Password	A Validation Should Be as Below "Please Enter Valid Username & Password"	A Validation Has Been Populated as Expected	Pass	High
		1. a Navigate to Www.Sample.Com 1. 2. Click on Submit Button with Out Filling Password and With Valid Username Test UsernameField	A Validation Should Be as Below "Please Enter Valid Password or Password Field Can Not Be Empty"	A Validation Is Shown as Expected	Pass	High

		1. Navigate To Www.Sample. Com 2. Enter Both Username and Password Wrong and Hit Enter	A Validation Shown as Below "The Username Entered Is Wrong"	A Validation Is Shown as Expected	Pass	High
		1. Navigate To Www.Sample. Com 2. Enter Validate Username and Password and Click on Submit	Validate Username and Password in Database and Once If They Correct Then Show the Main Page	Main Page/ Home Page Has Been Displayed	Pass	High

12. OUTPUT SCREENS

12. OUTPUT SCREENS

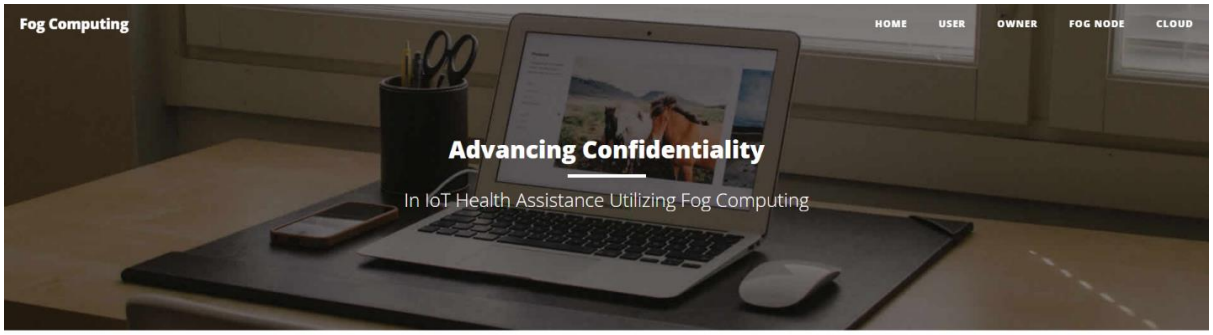


Fig 12.1.1: Home Page

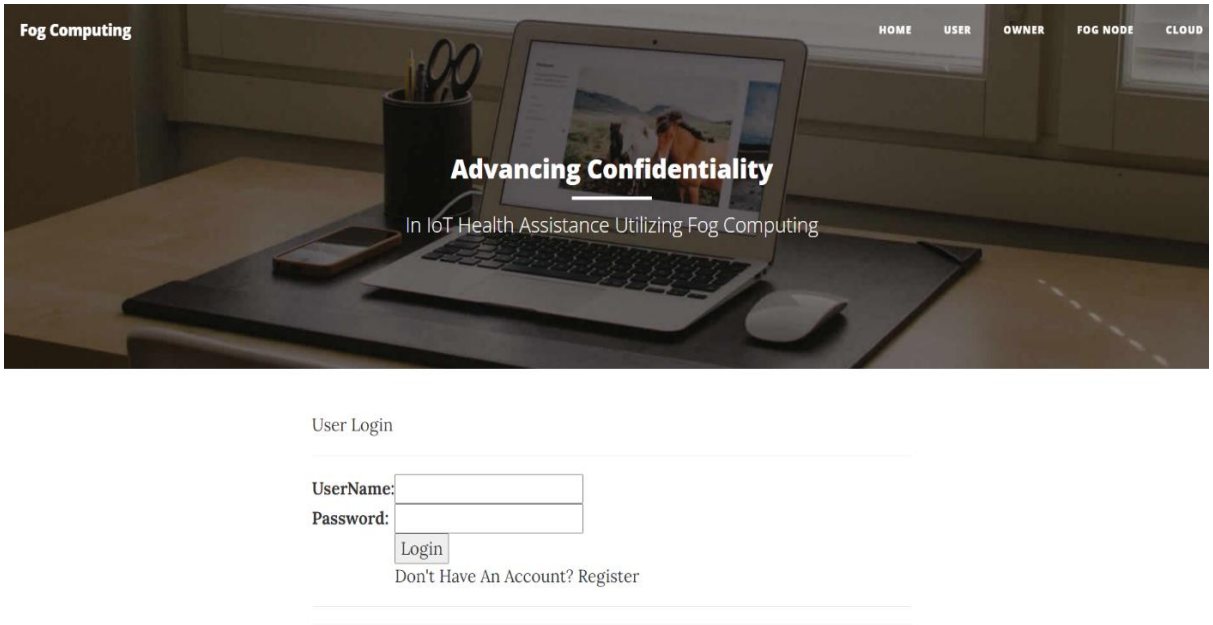


Fig 12.1.2: User login Page

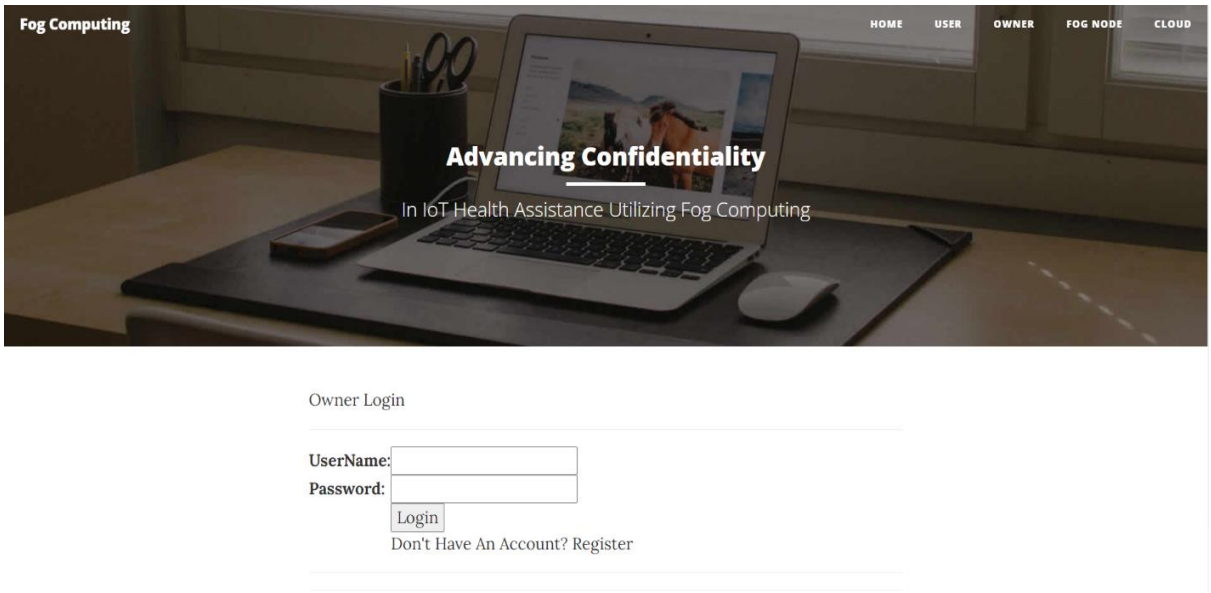


Fig 12.1.3: Owner login Page

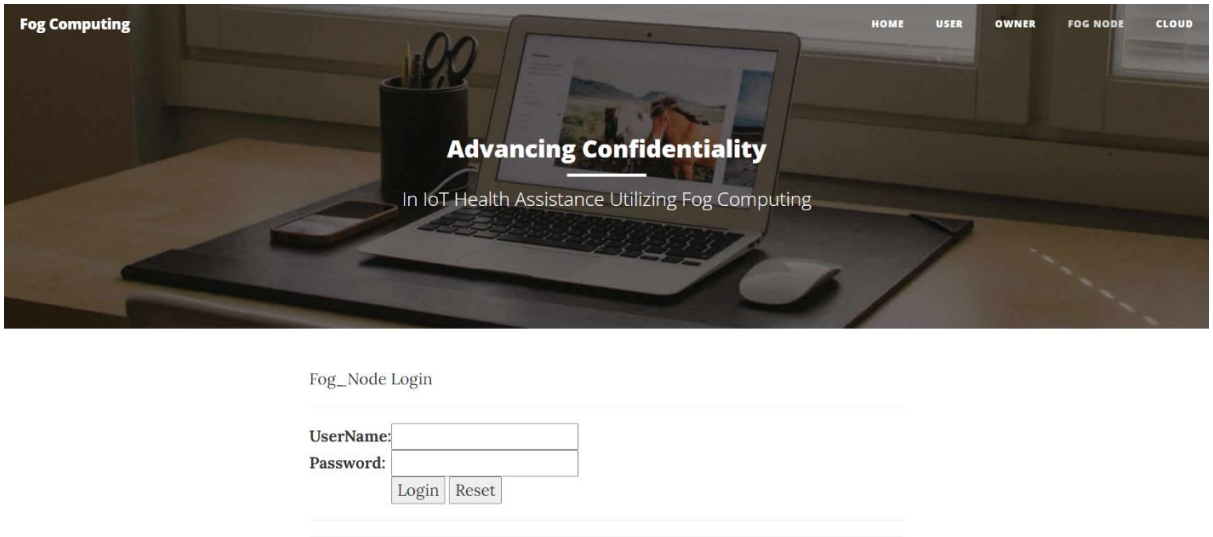


Fig 12.1.4: Fog Node login Page

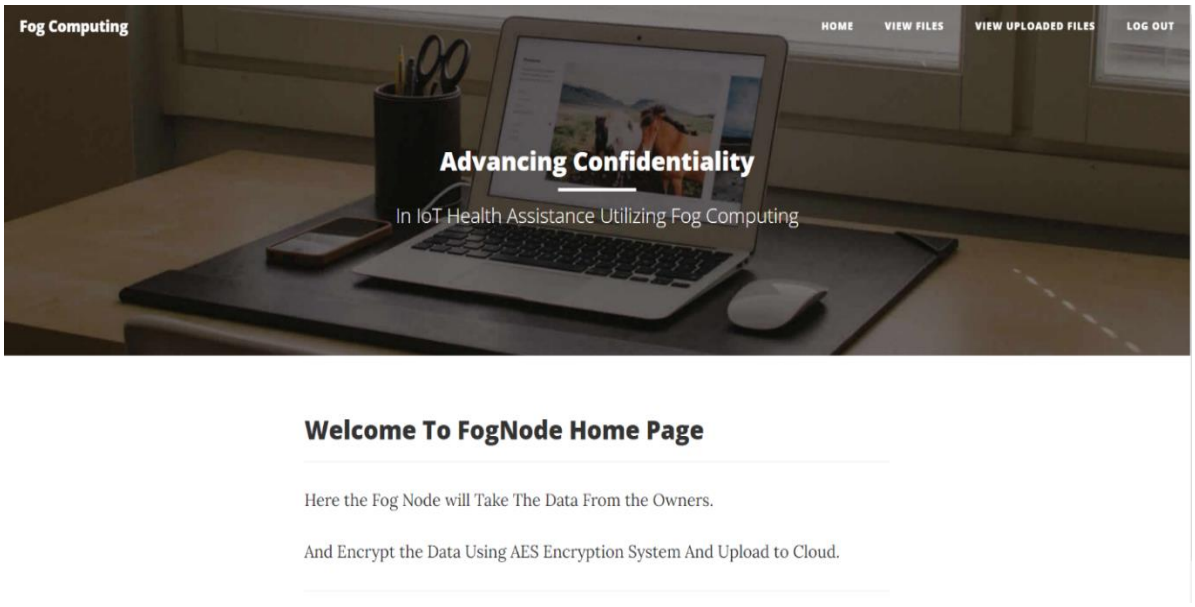


Fig 12.1.5: Fog Node Home Page

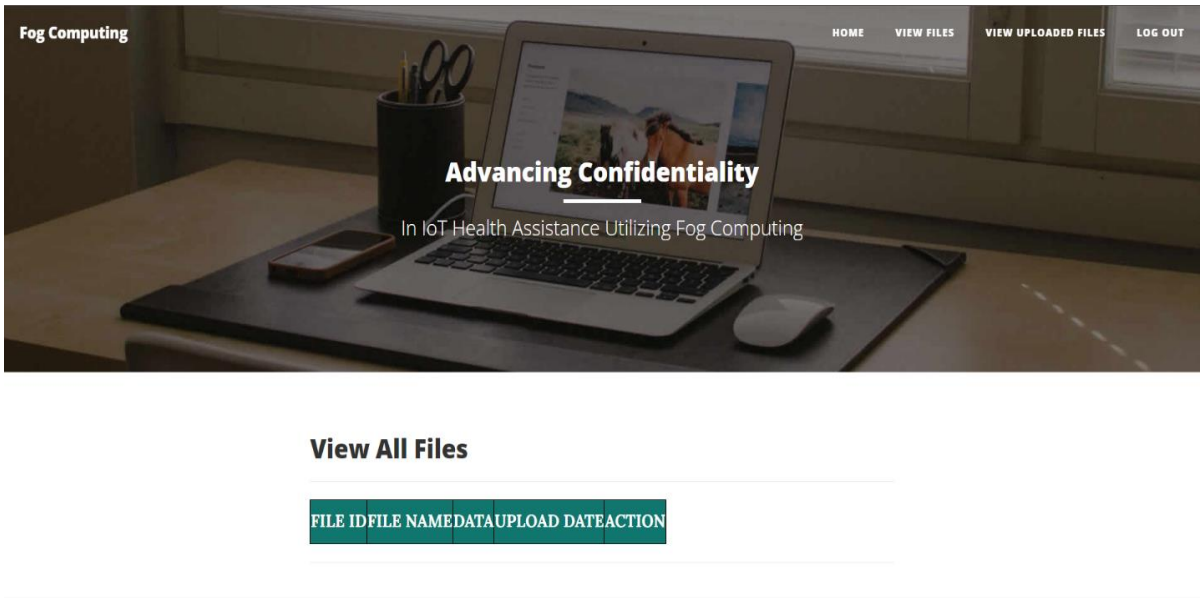
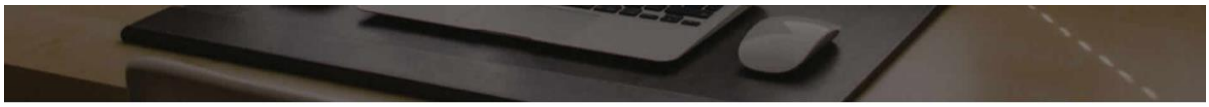


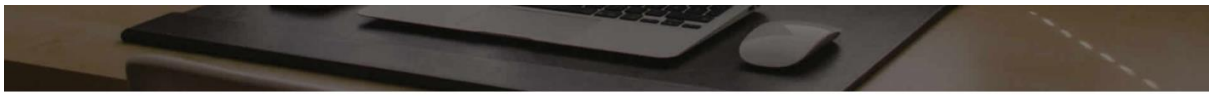
Fig 12.1.6: View All Files page



All Available Files

FILE ID	FILE NAME	DATA	CIPHER DATA	UPLOADED BY	UPLOAD DATE	STATUS
61153	sample.txt	this is the sample file	0cdiAEokp/eANOD5Ji4EF4gbdTRBGa9oSElLJ0W/YwI=	Fog Node	2020-03-08 06:52:50	Available
80141	a	hai this is file	7llreaU/lBTK7vTpQSPdyzPhSm5g/rPpmArL+YtzXfA=	Fog Node	2023-08-07 12:10:23	Available

12.1.7: View All Available Files Page



View All Uploaded Files into The Cloud

FILE ID	FILE NAME	DATA	UPLOAD DATE	STATUS
61153	sample.txt	this is the sample file	2020-03-08 06:52:50	uploaded_to_cloud
80141	a	hai this is file	2023-08-07 12:10:23	uploaded_to_cloud

Fig 12.1.8: View All Upload Files Page

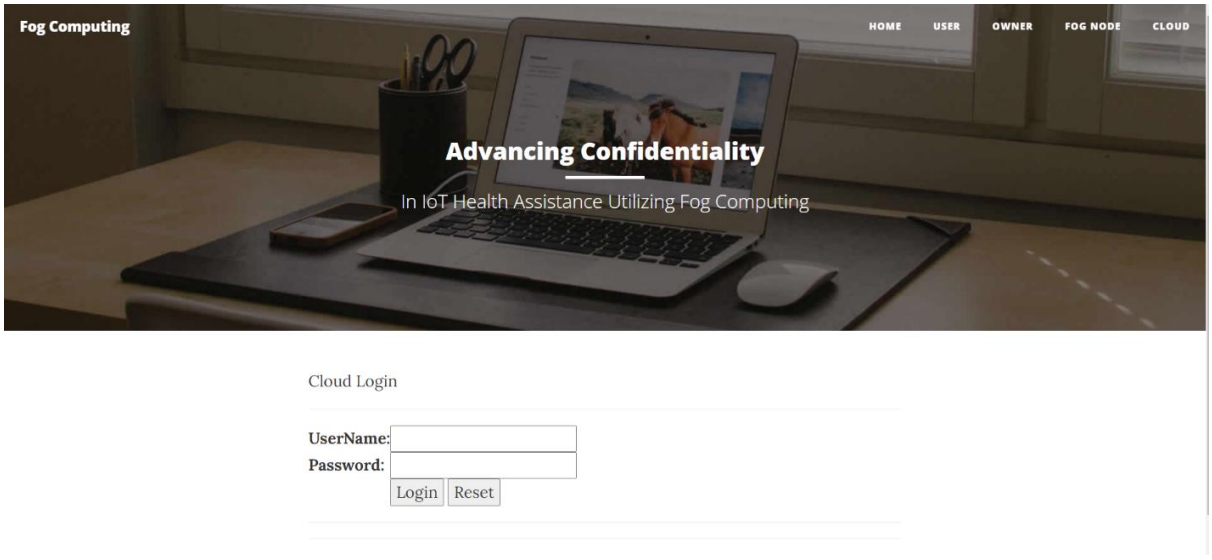


Fig 12.1.9: Cloud Login Page

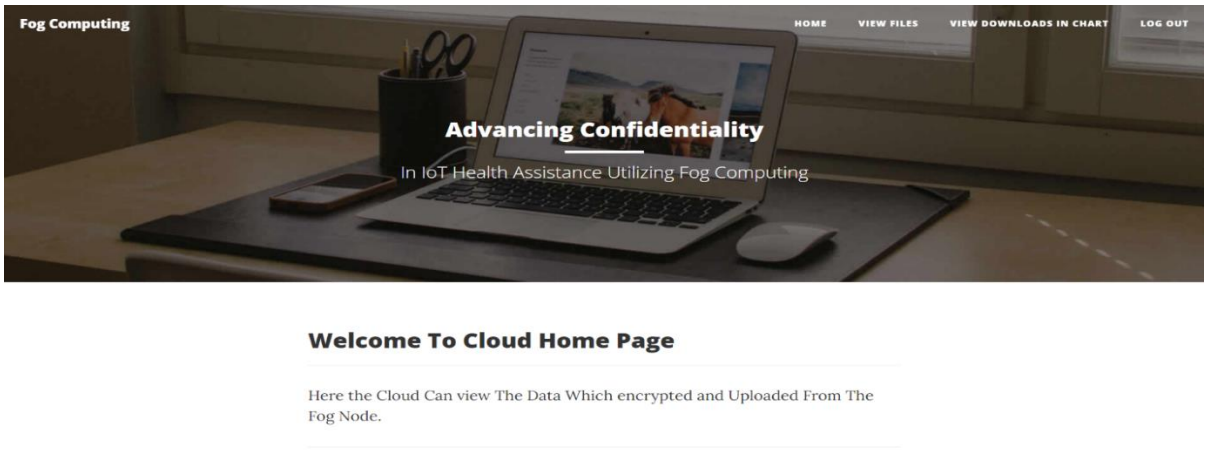


Fig 12.1.10: Cloud Home Page.

12.CONCLUSION

12. CONCLUSION

12.1 CONCLUSION:

Fog computing architecture is able to overcome the security challenges of the traditional IoT cloud architecture to some extent. By introducing fog as a middle layer and performing at the edge side it enhances data security, accuracy, consistency, reduces the latency rate and enhances the overall quality of service. In the near future IoT-Fog-cloud architecture will be widely used as more and more IoT devices are developed and the increasing demand for fast computation. The implementation can be enhanced in future by developing a reliable real time data monitoring system application with the mentioned architecture as a core. And to give a computational prove of how much fog can enhance the traditional IoT Cloud architecture.

Fog computing is considered as one of the important research directions for many purposes in healthcare IoT systems. Research endeavors in this direction are still in progress. However, pertinent portrayals and limits continue to be considered ambiguous. In this study, acquiring understanding and insights into this domain is considered to be significant. By reviewing and arranging applicable research exertions, this study intends to add to such understanding and knowledge.

Various challenges of Healthcare 4.0 environment using fog computing technology such as data management, security and privacy, scalability, human interfaces, and interoperability have been covered in this paper. Fog computing helps the doctors to take smart decisions during emergency for time-critical Healthcare applications. It also helps to protects sensitive data with reduced delay in comparison to the standalone cloud-based application.

The concept of fog computing and Smart e-Health Gateways in the context of Internet-of-Things based healthcare systems was presented. Smart gateways at the close proximity of sensor nodes in smart home or hospital premises can exploit their unique strategic position to tackle many challenges in IoT-based health systems such as mobility, energy efficiency, scalability, interoperability, and reliability issues.

12.2 FUTURE SCOPE:

Nowadays, IoT is applied in almost all domains of humanity and it will play an extremely vital role as well as the key driving force for the development of the future Internet. In the next decade, billions of IoT devices and smart applications will emerge in a series of areas such as smart manufacturing, smart agriculture, augmented reality, healthcare, etc. and

make basic changes for our world. To respond to these huge requirements, a robust, reliable, and flexible FC architecture to optimize network systems is topical and urgent. We discussed some challenges and possible research directions as presented in for future edge computing as follows:

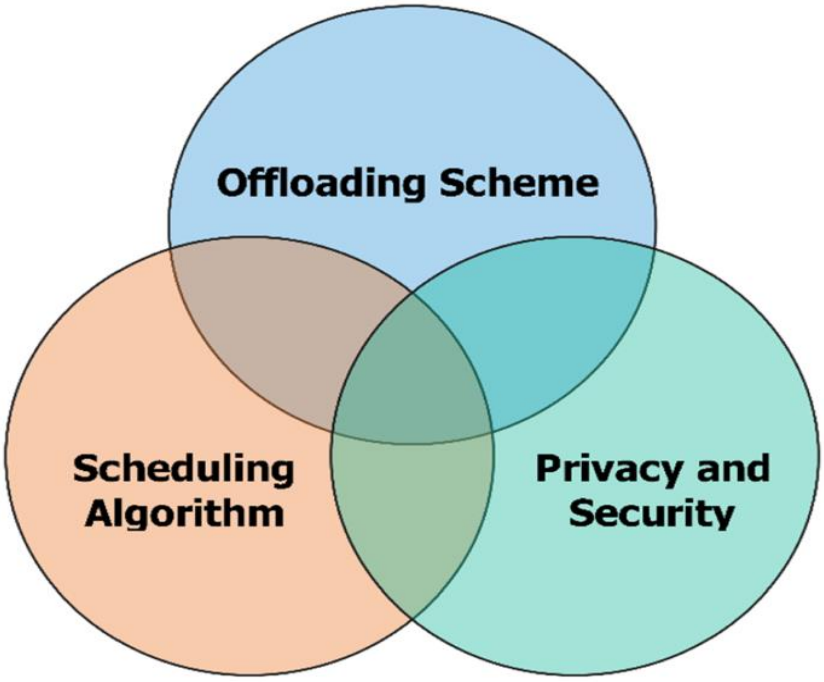


Fig.no 11.2 Future Scope

Some key challenges of IoT applications

1. An efficient computing offloading scheme to achieve optimized performance needs to be considered. This scheme must be able to allocate appropriate tasks to both fog computing and cloud computing systems. In, the authors introduced a hybrid offloading schema between cloud and fog to tasks associated with complex applications. In this study, based on the task requirements, the IoT node based on the Q-learning algorithm will choose to offload tasks to the fog servers or to the cloud to load balancing and executed more efficiently.
2. An efficient scheduling algorithm to achieve energy efficiency, reliably manage and control fog servers in heterogeneous network environments needs to be considered. In, the authors introduced an efficient scheduling algorithm to saving energy for fog-based IoT systems. This algorithm aims to derive the optimal scheduling decision based on multiple neighbour helper nodes sharing their computation resources. Besides, in the

authors also introduced an efficient scheduling algorithm for efficient resource utilization on fog servers. Specifically, they rank fog nodes based on latency and resource utilization criteria. Then, intelligent scheduling algorithms based on the game theory approach are used to optimise the allocation of tasks.

3. In healthcare, the need for security on fog servers have been highlighted in recent studies due to potential unsafe issues from public network environments as well as resource sharing of servers. Accordingly, to Turgut et demonstrated that the successful deployment of the IoT applications will only happen if the customer benefits (the potential benefits that IoT systems provide to exceed their physical value and security and privacy costs). In our opinion, if privacy and security is not guaranteed, these potential risks can obscure the benefits of Internet of health Things applications.

BLOCKCHAIN IN HEALTHCARE: Blockchain technology can be used to create an unalterable and secure patient data management system.

QUANTUM COMPUTING: Quantum computing is a powerful resource that can help improve the processing and security of patient data.

EDGE AI: Edge AI can help devices learn from data patterns and provide highly personalized care to patients.

13.REFERENCES/DISCUSSION

13. REFERENCES/BIBILOGRAPHY

1. Kraemer, Frank Alexander, Anders Eivind Braten, Natta chart Tamkittikhun, and David Palma. "Fog Computing in Healthcare–A Review and Discussion." IEEE Access 5 (2017): 9206-9222.

2. Khan, Saad, Simon Parkinson, and Yongrui Qin. "Fog computing security: a review of current applications and security solutions." Journal of Cloud Computing 6, no. 1 (2017):19.

3. Fog Computing Architecture, https://www.slideshare.net/saisharansai/fogcomputing-46604121, July 2018.

4. Al Hamid, Hadeal Abdulaziz, Sk Md Mizanur Rahman, M. Shamim Hossain, Ahmad Almgren, and Atif Alamri. "A Security Model for Preserving the Privacy of Medical Big Data in a Healthcare Cloud Using a Fog Computing Facility with Pairing-Based Cryptography." IEEE Access 5 (2017): 22313-22328.

5. Alrawais, Arwa, Abdulrahman Alhothaily, Chunqiang Hu, and Xiuzhen Cheng. "Fog computing for the internet of things: Security and privacy issues." IEEE Internet Computing 21, no. 2 (2017): 34-42.

6. Elminaam, Diaa Salama Abdul, Hatem Mohamed Abdul Kader, and Mohie Mohamed Hadhoud. "Performance evaluation of symmetric encryption algorithms." IJCSNS International Journal of Computer Science and Network Security 8, no. 12 (2008): 280-286.

7. Coppersmith, Don, Donald Byron Johnson, and Stephen M. Matyas. "A proposed mode for triple-DES encryption." IBM Journal of Research and Development 40, no. 2 (1996): 253-262.

8. Gia, Tuan Nguyen, Mingzhe Jiang, Amir Mohammad Rahmani, Tomi Westerlund, Pasi Liljeberg, and Hannu Tenhunen. "Fog computing in healthcare internet of things: A case study on ecg feature extraction." In Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference on, pp. 356-363. IEEE, 2015.

9. Mukherjee, Mithun, Rakesh Matam, Lei Shu, Leandros Maglaras, Mohamed Amine Farag, Nikuman Choudhury, and Vikas Kumar. "Security and privacy in fog computing: Challenges." IEEE Access 5 (2017): 19293- 19304.

10. Shrestha, N. M., Abeer Alsadoon, P. W. Prasad, L. Hourany, and A. Elchouemi. "Enhanced e-health framework for security and privacy in healthcare system." In Digital Information Processing and Communications (ICDIPC), 2016 Sixth International Conference on, pp. 75-79. IEEE, 2016.

11. Wang, Qixu, Dajing Chen, Ning Zhang, Zhe Ding, and Zhiguang Qin. "PCP: A Privacy Preserving Content-Based Publish–Subscribe Scheme with Differential Privacy in Fog Computing." IEEE Access 5 (2017): 17962- 17974.

12. Wanve, Balu, Rahul Kamble, Sachin Patil, and Jayshree Katti. "Framework for client-side AES encryption technique in cloud computing." In Advance Computing Conference (IACC), 2015 IEEE International, pp. 525-528. IEEE, 2015.

13. Vishwanath, Akhilesh, Ramya Peruri, and Jing (Selena) He. Security in fog computing through encryption. Digital Commons@ Kennesaw State University, 2016.

14. Sadikin, Mohamad Ali, and Rini Wisnu Wardhani. "Implementation of RSA 2048-bit and AES 256-bit with digital signature for secure electronic health record application." In Intelligent Technology and Its Applications (ISITIA), 2016 International Seminar on, pp. 387-392. IEEE, 2016.

15. Shao, Fei, Zinan Chang, and Yi Zhang. "AES encryption algorithm based on the high-performance computing of GPU." In Communication Software and Networks, 2010. ICCSN'10. Second International Conference on, pp. 588-590. IEEE, 2010.

16. Uma Maheswari E, Ajay DM, Umang Sindal, Scope of Internet of Things: A Survey, Asian Journal of Pharmaceutical and Clinical Research, April 2017.

APPENDIX-A

DIAGRAMS:

APPENDIX-B

DIAGRAMS:

TABLES OF CONTENTS

PROGRAM OUTCOMES:

Program Outcomes (POs) describe what students are expected to know and be able to do by the time of graduation to accomplish Program Educational Objectives (PEOs).The Program Outcomes for Computer Science and Engineering graduate are:

PO1: Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2: Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3: Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4: Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5: Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6: The engineer and society: Apply reasoning informed by the contextual knowledge to asses societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7: Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9: Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10: Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentation, and give and clear instructions.

PO11: Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12: Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technologies change.

PROGRAM SPECIFICATION OUTCOMES:

PSO1: Design, implement, test and evaluate a computer system, or algorithm to meet desired needs and to solve a computational problem.

PSO2: Ability to analyze, design and implement hardware and software components.

AVANTHI INSTITUTION OF ENGINEERING AND TECHNOLOGY

(Affiliated to JNTUH, Approved by AICTE, Recognized by Govt of T.S, Accredited by NBA,NAAC)

PROJECT TITLE: ADVANCING CONFIDENTIALITY IN IOT HEALTH ASSISTANCES UTILIZING FOG COMPUTING

INTERNAL GUIDE: Mr.VIRUPAKSHI RAVINDRANATH M.S.C, Assistant Professor

TEAM MEMBERS: S.HARSHITHA (20Q61A0550)

K.HARSHITHA (20Q61A0557)

E.SAIKIRAN (20Q61A0519)

K.YASWANTH (20Q61A0534)

PO ATTAINMENT

COURSE KNOWLEDGE	PAGE NO	PO ATTAINED	DESCRIPTION
SOFTWARE ENGINEERING	29	PO.5, PO.8	Development Phases
JAVA	53-65	PO.1, PO.3	Java is used to build apps on android
WT	50-53	PO.1, PO.5	HTML,CSS,HTML5,CSS3
OOAD	31-43	PSO1,PSO2,PO.5	UML Diagrams
STM	91-97	PSO1,PSO2,PO.5	Software Testing
OOAD LAB	31-43	PO.1,PO.3,PO.5	UML Diagrams
STM LAB	91-97	PSO1,PSO2,PO.5	Software Testing and phases of testing