

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belgavi-590018



A PROJECT REPORT

On

“FOG DATA ANALYSIS FOR SMART HEALTHCARE SYSTEM”

A Dissertation Submitted in partial fulfillment of the requirement for the degree of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE & ENGINEERING

Submitted by

ABHINAND K P (1RG16CS003)

**Under The Guidance
of**

Mrs. SONIYA KOMAL

Asst. Professor, Dept. of CSE,

RGIT, Bengaluru-32

Department of Computer Science & Engineering

RAJIV GANDHI INSTITUTE OF TECHNOLOGY

Cholanagar, R.T.Nagar Post, Bengaluru-560032

2021-2022

RAJIV GANDHI INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University)

Cholanagar, R.T.Nagar Post, Bengaluru-560032

Department of Computer Science & Engineering



CERTIFICATE

Phase-II

This is to certify that the Project Report titled “**FOG DATA ANALYSIS FOR SMART HEALTHCARE SYSTEM**” is a bonafide work carried out by **Mr. ABHINAND K P (1RG16CS003)** in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the **Visvesvaraya Technological University, Belagavi**, during the year **2021-2022**. It is certified that all corrections/suggestions given for Internal Assessment have been incorporated in the report. This project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the said degree.

Signature of Guide

Mrs. Soniya Komal

Asst. Professor

Dept. of CSE,

RGIT, Bengaluru

Signature of HOD

Mrs. Arudra A

Head of Department

Dept. of CSE,

RGIT, Bengaluru

Signature of Principal

Dr. D G Anand

Principal

RGIT, Bengaluru

External Viva

Name of the Examiners

1.

2.

Signature with date



VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belgavi-590018

RAJIV GANDHI INSTITUTE OF TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING



DECLARATION

We hereby declare that the project work entitled **“FOG DATA ANALYSIS FOR SMART HEALTHCARE SYSTEM”** submitted to the **Visvesvaraya Technological University, Belagavi** during the academic year **2021- 2022**, is record of an original work done by us under the guidance of **Mrs. Soniya Komal**, Asst. Professor, Department of Computer Science and Engineering, RGIT, Bengaluru in the partial fulfillment of requirements for the award of the degree of **Bachelor of Engineering in Computer Science & Engineering**. The results embodied in this project have not been submitted to any other University or Institute for award of any degree or diploma.

ABHINAND K P (1RG16CS003)

ACKNOWLEDGEMENT

We take this opportunity to thank our college **Rajiv Gandhi Institute of Technology, Bengaluru** for providing us with an opportunity to carry out this project work.

We express our gratitude to **Dr. D G Anand**, Principal, RGIT, Bengaluru for providing the resources and support without which the completion of this project would have been a difficult task.

We extend our sincere thanks to **Mrs. Arudra A**, Associate Professor and Head, Department of Computer Science and Engineering, RGIT, Bengaluru, for being a pillar of support and encouraging us in the face of all adversities.

We would like to acknowledge the thorough guidance and support extended towards us by **Mrs. Geetha Pawar**, Assistant Professor, Dept. of CSE, RGIT, Bengaluru, **Dr. Anthony Raj**, Assistant Professor, Dept. of CSE, RGIT, Bengaluru. Their incessant encouragement and valuable technical support have been of immense help. Their guidance gave us the environment to enhance our knowledge and skills and to reach the pinnacle with sheer determination, dedication and hard work.

We also want to extend our thanks to the entire faculty and support staff of the Department of Computer Science and Engineering, RGIT, Bengaluru, who have encouraged us throughout the course of the Bachelor's Degree. We want to thank our family for always being there with full support and for providing us with a safe haven to conduct and complete our project. We are ever grateful to them for helping us in these stressful times.

Lastly, we want to acknowledge all the helpful insights given to us by all our friends during the course of this project.

ABHINAND K P (1RG16CS003)

ABSTRACT

Cloud computing is an on-demand service that provides storage, network, power to do intensive computing, it also provides intensive sharing of resources, even though the users use all these advantages, they have to also deal with the privacy-related issues to the data stored in the cloud, where the data may get leaked or may be accessed by the service provider. The current health care systems implemented on the cloud poses threat to the privacy of the data of the patients since data breaches cost millions to billions of dollars to the health care institutions every year this problem can be taken care with the fog based health care system, where the fog technology offers more additional advantages like low latency, resource management, less power consumption, etc.

Table of Contents

ACKNOWLEDGEMENT	I
ABSTRACT	II
TABLE OF CONTENTS.....	III
LIST OF FIGURES.....	VI
LIST OF TABLES.....	V

CHAPTER 1

INTRODUCTION.....	1
1.1 Problem Statement.....	2
1.2 Objectives of the project	2
1.3 Project deliverables	3
1.4 Current scope	3
1.5 Future scope	3

CHAPTER 2

LITERATURE SURVEY	4
--------------------------------	----------

CHAPTER 3

SYSTEM REQUIREMENT SPECIFICATION	9
3.1 Hardware Requirements	9
3.2 Software Requirements	9
3.3 Functional Requirements	10
3.4 Non Functional Requirements.....	10

CHAPTER 4

SYSTEM DESIGN AND ARCHITECTURE	11
4.1 System Architecture	11
4.2 Use case Diagram.....	13
4.3 Dataflow Diagram Level	15
4.4 E R diagram.....	16
4.5 Activity Diagram.....	17
4.6 Sequence Diagram.....	19
4.7 Class Diagram.....	21

CHAPTER 5

Table of Contents

IMPLEMENTATION.....	22
5.1 Tools instruction.....	22
5.2 Technology instruction.....	23
5.3 Overall view	24
5.4 Algorithm and implementation.....	25
5.5 Modules.....	27
5.6 Conclusion.....	28
 CHAPTER 6	
OUTPUT SNAPSHOTS	29
6.1 Results snapshots.....	29
6.2 Comparision table.....	35
6.3 Contribution to the existing system	41
 CHAPTER 7	
TESTING AND RESULTS	43
7.1 Testing.....	43
7.2 Test Cases.....	45
 CHAPTER 8	
CONCLUSION.....	47
 REFERENCES	49
BASE PAPER	55

FIGURES

Figure 2.1	Interaction of different software components withinFogBus [1]	5
Figure 2.2	The proposed architecture diagram presented by HealthFog [2]	5
Figure 2.3	Fog computing technical architecture [3]	6
Figure 2.4	Fog computing in healthcare[4]	6
Figure 2.5	System architecture of [5]	7
Figure 2.6	A network coding protocol for wireless sensor fog computing[6]	7
Figure 2.7	Architecture of Systems[7]	8
Figure 2.8	Distributed Data Processing in a Fog Environment [8]	8
Figure 4.1	Architecture design	12
Figure 4.2	Design of visual interface	13
Figure 4.3	Design of graphical user interface	14
Figure 4.4	Class diagram	15
Figure 4.5	E R diagram	16
Figure 4.6	Sequence diagram	17
Figure 4.7	Sequence diagram	18
Figure 4.8	Data flow diagram	19
Figure 4.9	Data flow diagram	20
Figure 5.1	General Flow of Algorithms on Dataset	26
Figure 5.2	Flow Chart of Naive Bayes Algorithm	27
Figure 6.1	Result snapshot	30
Figure 6.2	Comparision Table	36
Figure 6.3	Comaprision table	36
Figure 7.1	Testing process	46

INTRODUCTION

Today cloud computing is an area liable to investigate. Cloud computing is an on request administration given to the client. Uploading the information on the cloud is safe, as the client taking this advantage totally depends on the specialist co-op and is not concerned about how and where the information is being put away. Fog processing is currently turning into a spine to cloud computing. In fog computing, registering a few fringe gadgets are associated with the cloud. The nature of administration is expanded when we use fog to help the cloud administration. The few advantages incorporate transfer speed utilization, inactivity decrease etc. Maintaining the protection of the patient information in a medical care area has collected a ton of consideration and preparing that specific information should likewise be focused on since there is voluminous information accessible today giving it over the cloud turns into a dreary undertaking. An answer for that is fusing the fog layer in our organization. Cloud layer raises high inertness which is settled by the fog layer which offers low inactivity correspondence. The principle destinations of this framework are to improve effectiveness and diminish the high volume of information which must be moved to the cloud for handling, breaking down and capacity and accomplishing the security of information utilizing Fog climate as preparing information is nearby and kept hidden. The fog computing process is a functioning examination region with genuine execution. It is likewise basic to make upgrades to the current instruments. For building IoT-empowered frameworks to both fog and cloud foundation, this structure must be lightweight and must tackle both edge and distant assets which have IoT application arrangement, checking and the executives.

Nowadays in healthcare applications, the wireless sensor network plays a vital role. The data is fed to the network through sensors. Sensor data is huge and doesn't require any form of wired network. This huge amount of data may comprise of several redundant as well as bogus data. The data which is useful must alone be taken to carry out further procedures like analysis, aggregation, prediction etc. The data which is free from redundancy is used to train the model in the fog layer. Using just clouds may lead to the process not being efficient due to the factor of delay which might be added.

A solution to this is to provide fog architecture, as the data remains local, the computation is much faster. In this paper, our main aim is to improvise the existing work on the healthcare systems. We have discussed the various algorithms that can be used to get the result. The algorithm selected for this work is decided by comparing the various algorithms on factors like accuracy, precision, energy consumption etc. Then that particular algorithm is used to predict the patients' heart disease by studying factors like blood pressure, cholesterol etc. (discussed in section IV). The result is then forwarded further to patients' android devices. We have considered the master- slave architecture to build the fog system.

1.1 Problem Statement

Maintaining the privacy of the patient data in a health care sector has garnered a lot of attention and processing that particular data must also be prioritized since there is voluminous data available today handing it over the cloud becomes a tedious task. A solution to that is incorporating the fog layer in our network. Cloud layer brings up high latency which is resolved by the fog layer which offers low latency communication.

1.2 Objectives Of The Project

1. By using a fog framework, the prime goal is to improve efficiency over the cloud and reduce the large volume of the dataset which should be transported to the cloud for the purpose of processing, analyzing and storage.
2. To create a fog computing framework to offer low latency communication as most of the nodes are closer to the clients.
3. To achieve the security of data using a Fog environment, training data is local and kept private.

1.3 Project Deliverables

1. Health care data is collected from smart devices integrated with the synthetic data.
2. The data collected is further uploaded to the nodes present of the fog layer.
3. In fog layer data is preprocessed, cleaned and normalized.
4. After gaining data insights, data is further classified by applying machine learning algorithms.
5. Classified data is secured as the data is local data.

1.4 Current Scope

To collect data from several iot devices, save the data on the server. Then collect patient data, compare those parameters with the saved data and predict the output telling if the user is suffering from any heart disease or not by looking up several conditions. The data processing is done over the fog layer so that the burden on the cloud layer is reduced.

1.5 Future Scope

When saving the data, it incorporates a block chain so that the security is maintained and access to the data is done only to authorized persons

CHAPTER 2 LITERATURE SURVEY

2.1 Introduction

The fog infrastructure has a set of devices with no computational abilities. As we were going via the use of test cases, we saw that in fog computing despite the potential, there has been only a little implementation in the healthcare domain. Data management has a significant role in fog computing, this can be used by sensual statistics and is used to remove evocative data for operator response. It is also used for data filtration, data compression etc.

2.2 Related Works

For any latency-sensitive applications like smart healthcare and smart city, the high-latency interaction between IoT, sensor devices and the Cloud datacenters is very high and hence unacceptable and it degrades the quality of service. In addition, IoT devices can produce a vast amount of data in just a fraction of a second. When these huge number of IoT devices concurrently initiates transferring of the information to Cloud servers through the Internet, critical network congestion will occur. To erase these obstacles of the Cloud based IoT model, Fog computing architecture emerged as shown in Fig 2.1 [1].

A computing architecture involving Fog-based Smart Healthcare System for the prediction of disease which uses deep learning algorithms and input from IoT devices known as HealthFog was proposed to provide effective service to heart patients and others, who are in need of real-time output, with less delay in response, minimum energy and power consumption and better accuracy. HealthFog provides this service as a fog computing service and also effectively handles the information of patients and it gets generated from several IoT devices as shown in Fig. 2.2 [2].

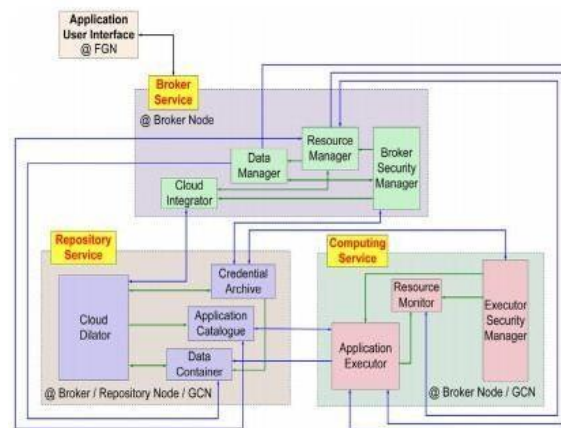


Fig. 2.1 Interaction of different software components within FogBus [1].

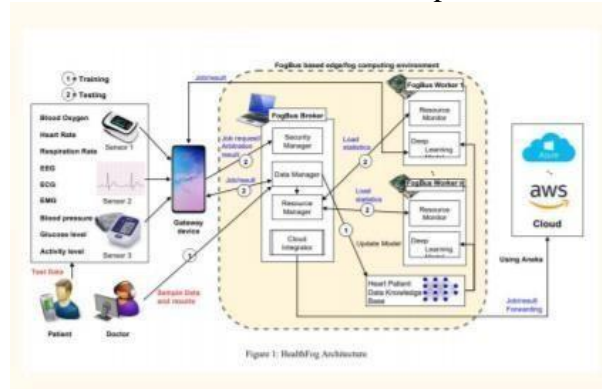


Fig. 2.2 The proposed architecture diagram presented by Health Fog [2].

The fog computing paradigm contains a bunch of devices having zero computational abilities such as sensors and collection of resources, which offer calculated power, storage capacity and more, which are fog architecture nodes and cloud architecture servers which are interconnected to form a system. We can find huge differences among the behaviours of all these devices. These might concern, among other characteristics, like the CPU generated power, hard disk capability, network bandwidth, computation latency, and RAM size as shown in Fig.2.3 [3].

The Internet of things that depends on skills gives a large number of facilities and inventions within the healthcare domain but the maximum of these facilities and inventions are still under development. The concept of IoT, fog computing and combination of both in the healthcare domain is presented in this research work as shown in Fig. 2.4 [4].

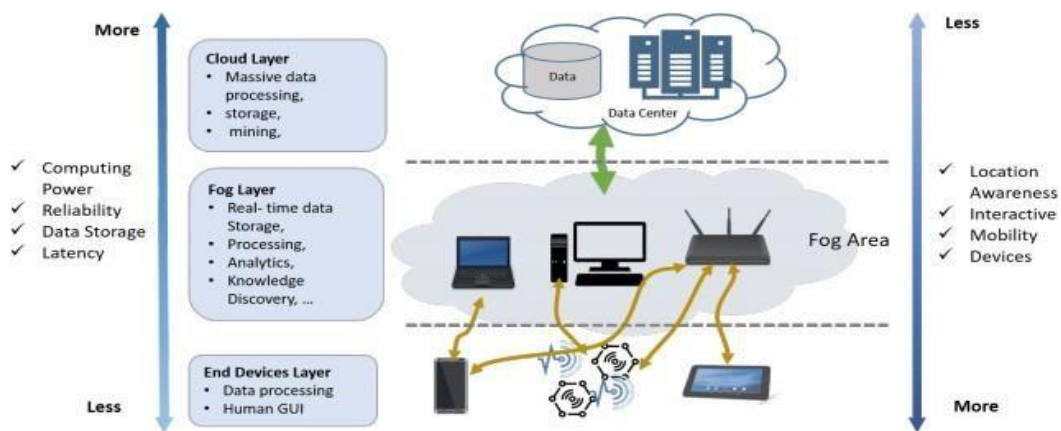


Fig. 2.3 Fog computing technical architecture [3]

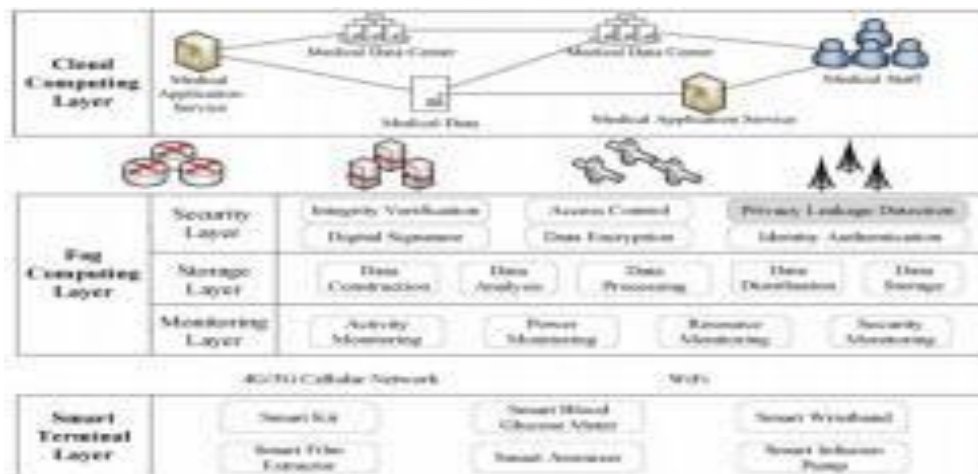


Fig. 2.4 Fog computing in health care[4].

Computing architecture involving the Fog paradigm is nothing but an elaboration of the existing classic computing architecture involving Cloud network. This is designed in order to support IoT applications characterized by limitations of security and latency. Even though computing, storage and networking are both Cloud computing and Fog computing features, the fog has specific features, low latency, geographic distribution, and a huge number of computing architecture nodes compared to the centralized Cloud computing, thus fog supports mobility and real-time interaction as shown in Fig. 2.5 [5].

In the work network coding convention for wireless fog computing, they have received a novel transport layer information dispersal protocol known as Dissemination of Small Values (DSV) of the architecture to apply the ideas of networks involving coding. DSV is another correspondence convention produced for the TinyOS working framework.

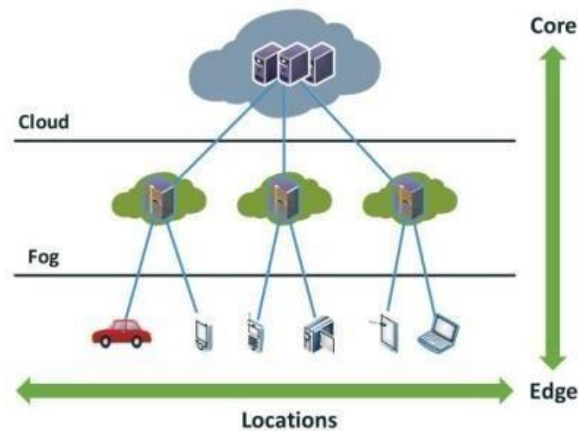


Fig. 2.5 System architecture of [5]

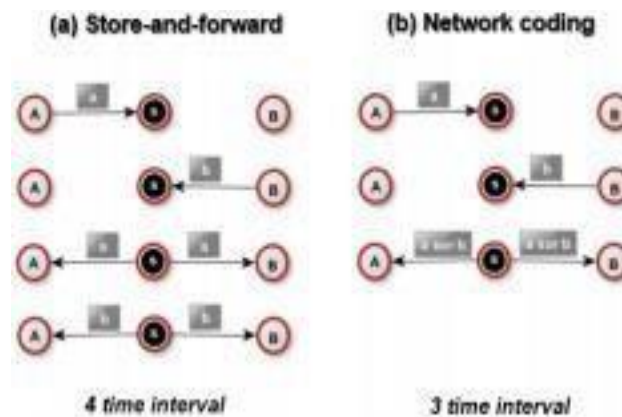


Fig. 2.6 A network coding protocol for wireless sensor fog computing[6].

For any Internet of things framework which comprises computing involving a fog architecture layer, the proposed engineering should be either application particular or application specific. This design can be comprehensively arranged into fundamental layers, things involving layers, fog architecture layers and a cloud architecture layer. The things layer is a point where IoT gadgets are available and an assortment of information occurs. The Fog layer comprises a few decentralized hubs present in every area. This layer plays out the undertaking of dealing with the organizations and the information obtained. The essential handling of information is done here. IoT based applications are improved by controlling the information transmission to the computing architecture involving the cloud layer, consequently decreasing the solicitation reaction delay taken for an IoT application as shown in Fig. 2.7 [7].

IfogSim, a simulation tool for fog computing, enables the modeling and simulation of Fog computing to evaluate resource management across edge resources under different scenarios. CloudSim is the layer responsible for taking care of operations between Fog architecture components in the iFogSim simulator. The implementation of iFogSim is made of simulated entities and services as shown in Fig. 2.8 [8].



Fig. 2.7 Architecture of Systems [7].

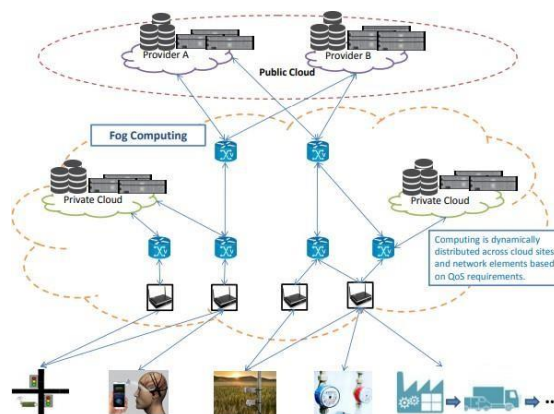


Fig. 2.8 Distributed Data Processing in a Fog Computing Environment[8].

CHAPTER 3

SYSTEM REQUIREMENT SPECIFICATION

It is a structured assortment of data that represents the necessities of a system. It includes in-depth descriptions of the software that will be developed. It is the collection of information on the requirements for a system.

3.1 External Interface Requirements

3.1.1 User Interfaces

- The first step is to gather vast amounts of data from smart devices.
- The data is received at the fog computing nodes, further preprocessed for training purposes and supplied to teach the model, and tested using the test data.
- The user can enter his current status of symptoms on the web application.
- Entered status of the patients is then computed with existing data and output is sent through the user interface.

3.1.2 Hardware Interfaces

Laptop / PC, Android device.

3.1.3 Software Interfaces

Software	Description
Operating System for Project	Windows 10 or any upgraded version of windows 10 can be used
Front-end	ASP.NET
Back-end	JAVA
API development	Flask REST

Table 5.1 Software Supported interfaces

ASP.NET

ASP.NET is a free web framework for building great websites and web applications using HTML, CSS, and JavaScript. You can also create Web APIs and use real-time technologies like Web Sockets.

ASP.NET offers three frameworks for creating web applications: Web Forms, ASP.NET MVC, and ASP.NET Web Pages. All three frameworks are stable and mature, and you can create great web applications with any of them. No matter what framework you choose, you will get all the benefits and features of ASP.NET everywhere.

Each framework targets a different development style. The one you choose depends on a combination of your programming assets (knowledge, skills, and development experience), the type of application you're creating and the development approach you're comfortable with.

JAVA

Java is an object-oriented programming language that produces software for multiple platforms. When a programmer writes a Java application, the compiled code (known as byte code) runs on most operating systems (OS), including Windows, Linux and Mac OS. Java derives much of its syntax from the C and C++ programming languages.

Java produces applets (browser-run programs), which facilitate graphical user interface (GUI) and object interaction by internet users. Prior to Java applets, web pages were typically static and non-interactive. Java applets have diminished in popularity with the release of competing products, such as Adobe Flash and Microsoft Silverlight.

Java applets run in a web browser with Java Virtual Machine (JVM), which translates Java byte code into native processor instructions and allows indirect OS or platform program execution. JVM provides the majority of components needed to run byte code, which is usually smaller than executable programs written through other programming languages. Byte code cannot run if a system lacks the required JVM.

FLASK REST

Flask-RESTful is an extension for Flask that adds support for quickly building REST APIs. It is a lightweight abstraction that works with your existing ORM/libraries. Flask-RESTful encourages best practices with minimal setup. If you are familiar with Flask, Flask-RESTful should be easy to pick up.

3.1.4 Communication Interfaces

Web application and Android application.

CHAPTER 4

SYSTEM DESIGN AND ARCHITECTURE

4.1 SYSTEM ARCHITECTURE

- Thinking about the abnormalities in the current framework, computerization of the entire action is being proposed after starting investigation.
- It may have happened so often that you or somebody of your needs specialists' assistance, and they couldn't be reached as a result of different barriers..
- The framework is accomplished as a web application which reviews the patient data and gives suggestions.
- The executed application exhibits instant results on the patient's coronary illness through a wise framework.
- The application is trained on multiple heart illnesses and can be updated with new diseases in the meantime this makes the system more feasible to make more changes.
- The executed framework enables the clients to know about their heart issues.
- The client has to enter his measures to get precise results
- Model is taught using data mining and machine learning techniques
- The output shows up with the disease, its symptoms and measures.
- The framework then sends a notification about the timestamp and predicted result and then the framework can be utilized in the event of a crisis.

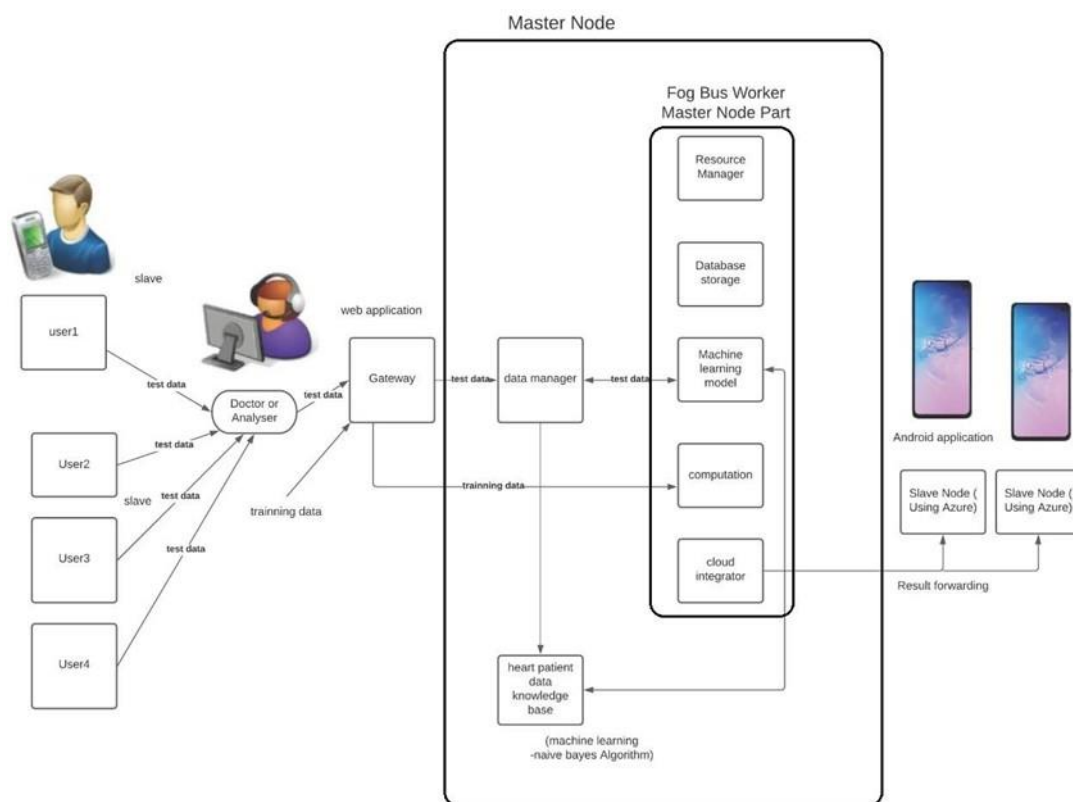


Fig. 4.1 Architecture Design

Like the cloud layer, the fog layer is also an environment having local systems as master and slave nodes, as shown in Fig. 4.1. Here we are creating the Fog computing architecture environment using a local system (our laptop where we are computing or achieving analysis of data) and an android app (our mobile where the user will get the data computing result). As our project architecture says we have users at one end i.e. patients and the doctor will collect the data for example. age, gender, blood sugar, blood pressure, ecg, cholesterol level etc. and at the doctor end he will feed the data through web application which acts a gateway and the local system which is master node (includes data manager, data storage, resource manager, machine learning model, cloud integrator) will take the data fed by doctor or analyzer at web application end and preprocess it and classify based on naive bayes algorithm and predicts the disease and forward the result from cloud integrator (azure) is delivered as notification of name, disease name and date of prediction to user's android application which is slave node. Here we are using local data which is kept private and it is only accessed by the admin (doctor). One important thing to note is that data is local, there is less delay in delivering the result so Fog is better than cloud.

4.2 Graphical User Interface

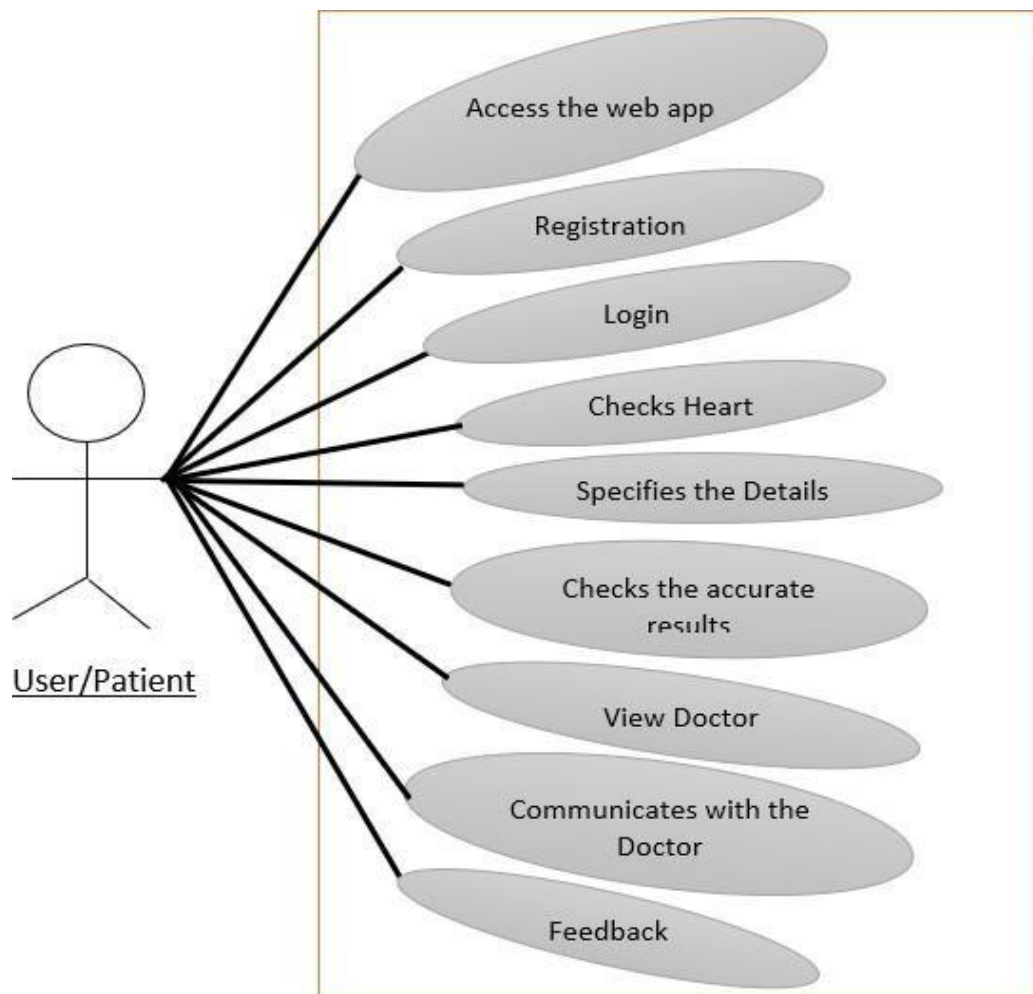


Fig. 4.2 Design of Visual interface.

A graphical user interface (GUI) is the way that users interface with the Windows and Macintosh operating systems. This is also referred to as a point-and-click interface. Users can use a mouse to click on an object and drag it into position. Our project graphical user interfacediagrams are shown in Fig. 4.2 and Fig. 4.3.

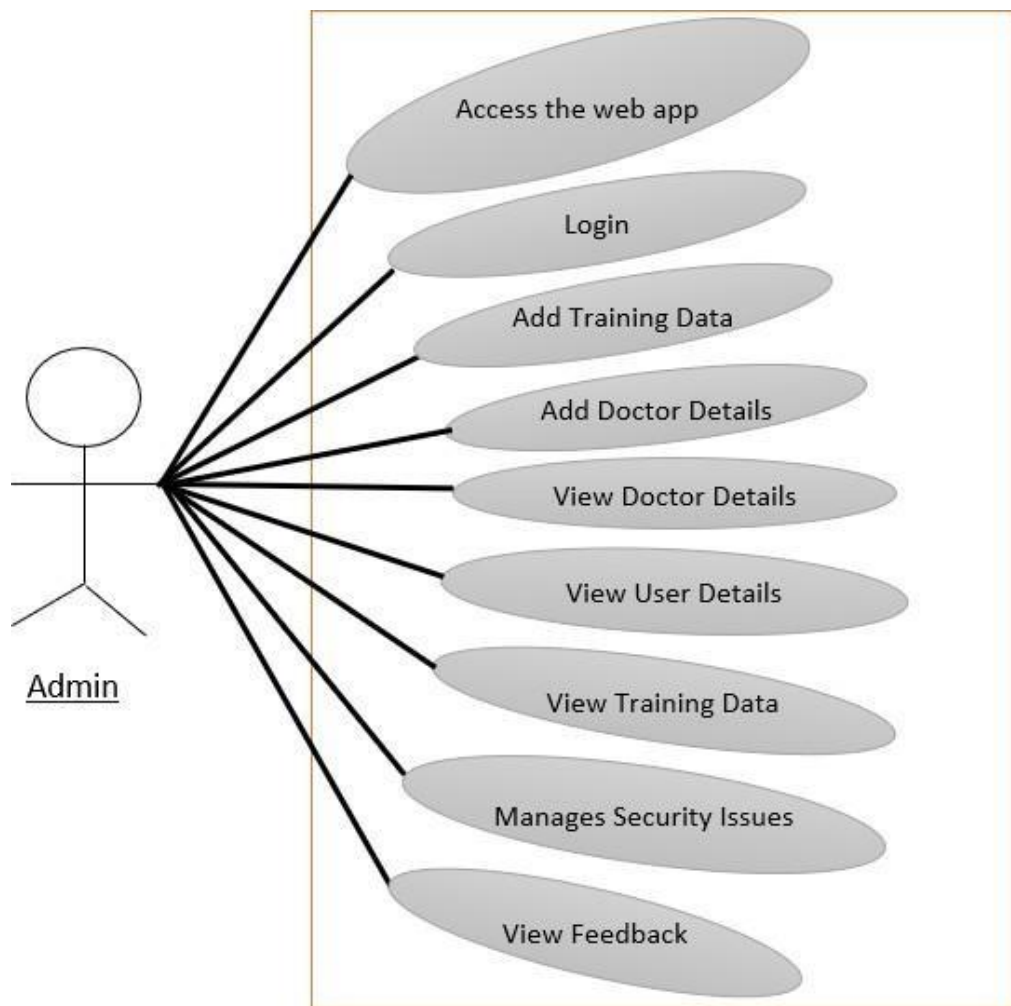


Fig. 4.3 Design of graphical User Interface

4.3 Class Diagram and Classes

A class diagram is used to visualize, describe, document various different aspects of the system, and also construct executable software code. It shows the attributes, classes, functions, and relationships to give an overview of the software system. It constitutes class names, attributes, and functions in a separate compartment that helps in software development. The diagram alongside represents the class diagram of the proposed system. Our project class diagram is shown in Fig. 4.4

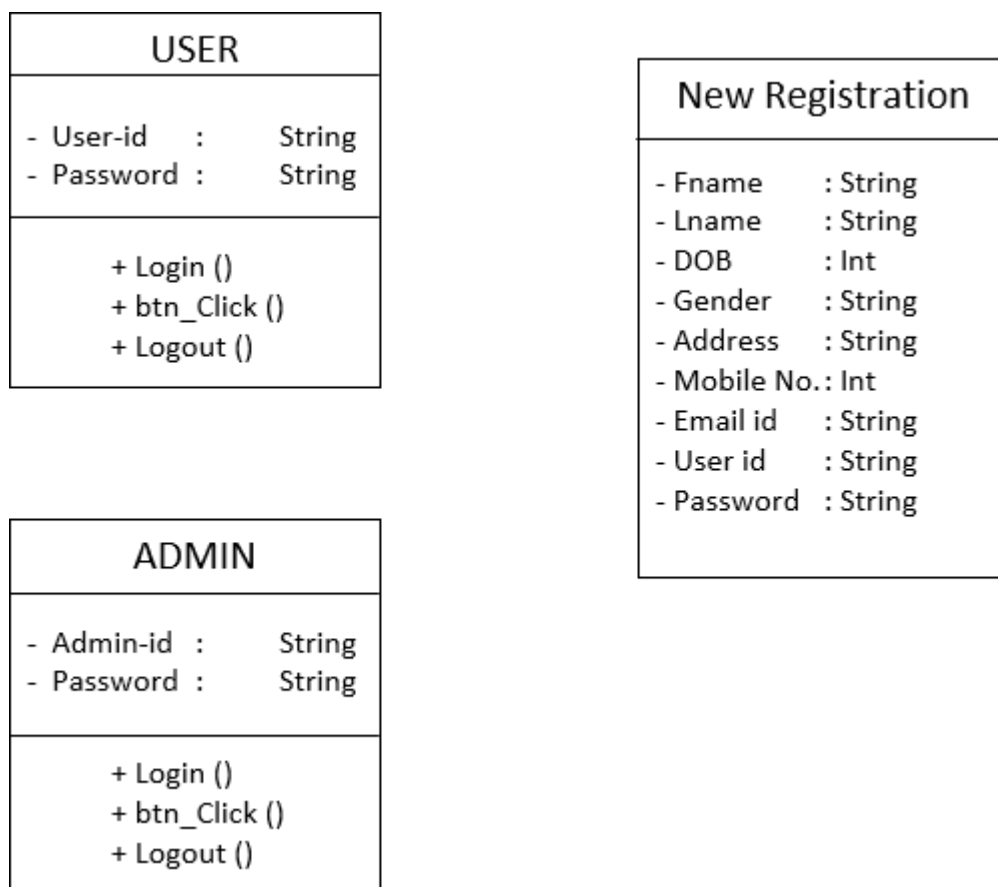


Fig. 4.4 Class Diagram

4.4 E-R Diagram

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how –entities‖ such as people, objects or concepts relate to each other within a system. ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education and research. Our project E-R diagram is shown in Fig. 4.5.

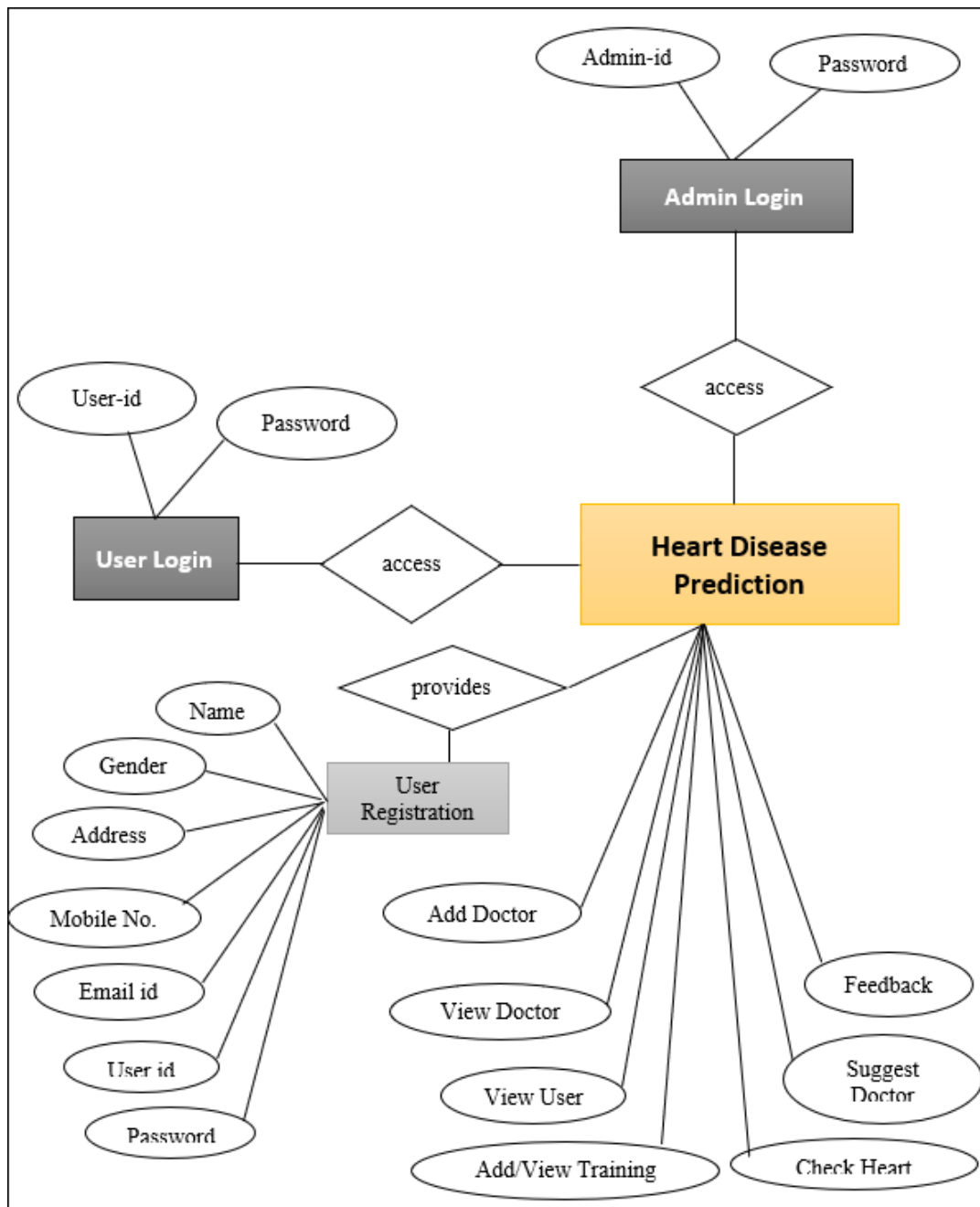


Fig. 4.5 ER-Diagram

4.5 Sequence Diagram

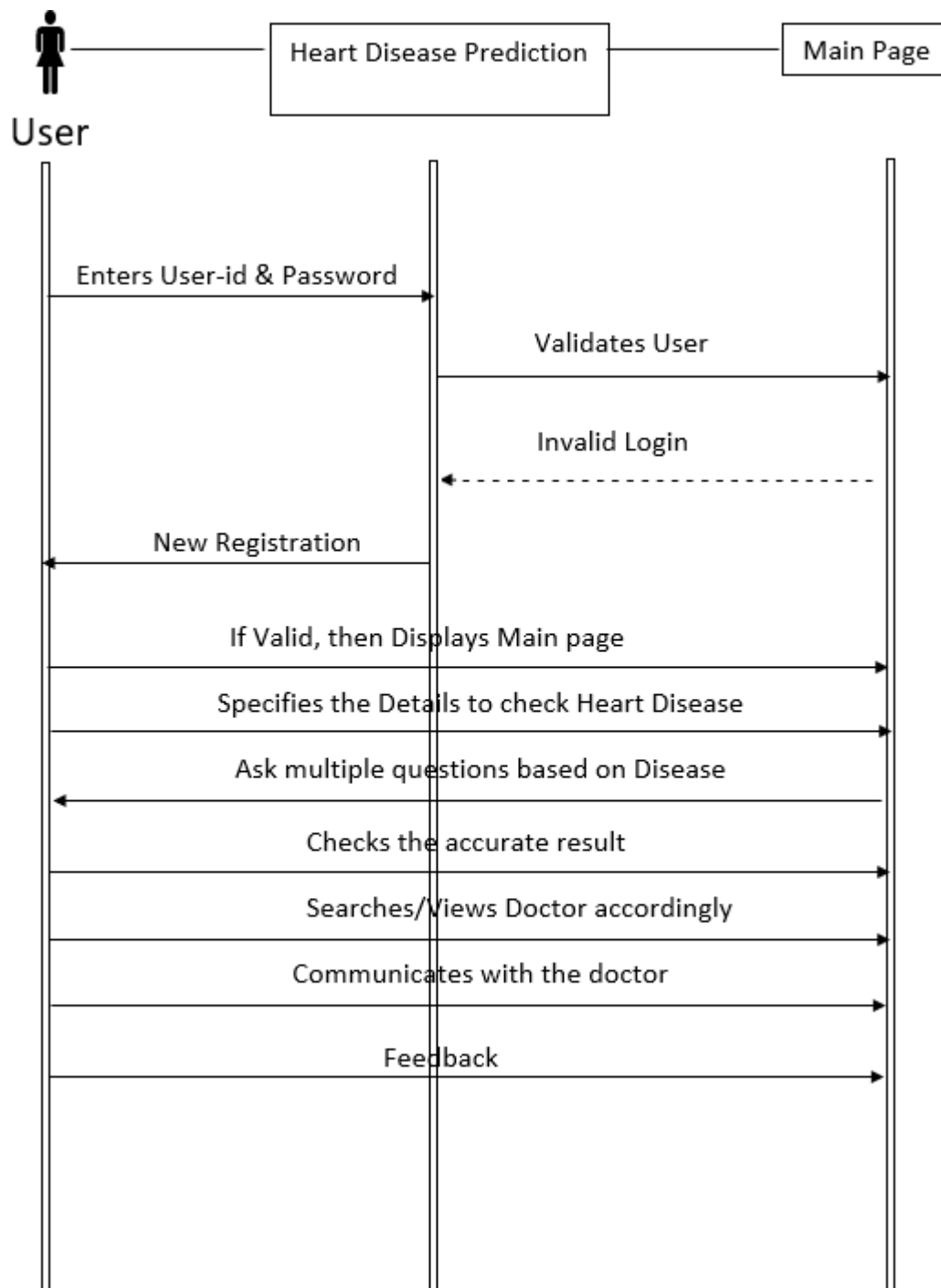


Fig. 4.6 Sequence Diagram

Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration. Sequence Diagrams are time focused and they show the order of the interaction visually by using the vertical axis

of the diagram to represent what messages are sent and when. Our project sequence diagrams are shown in Fig. 4.6 and Fig. 4.7.

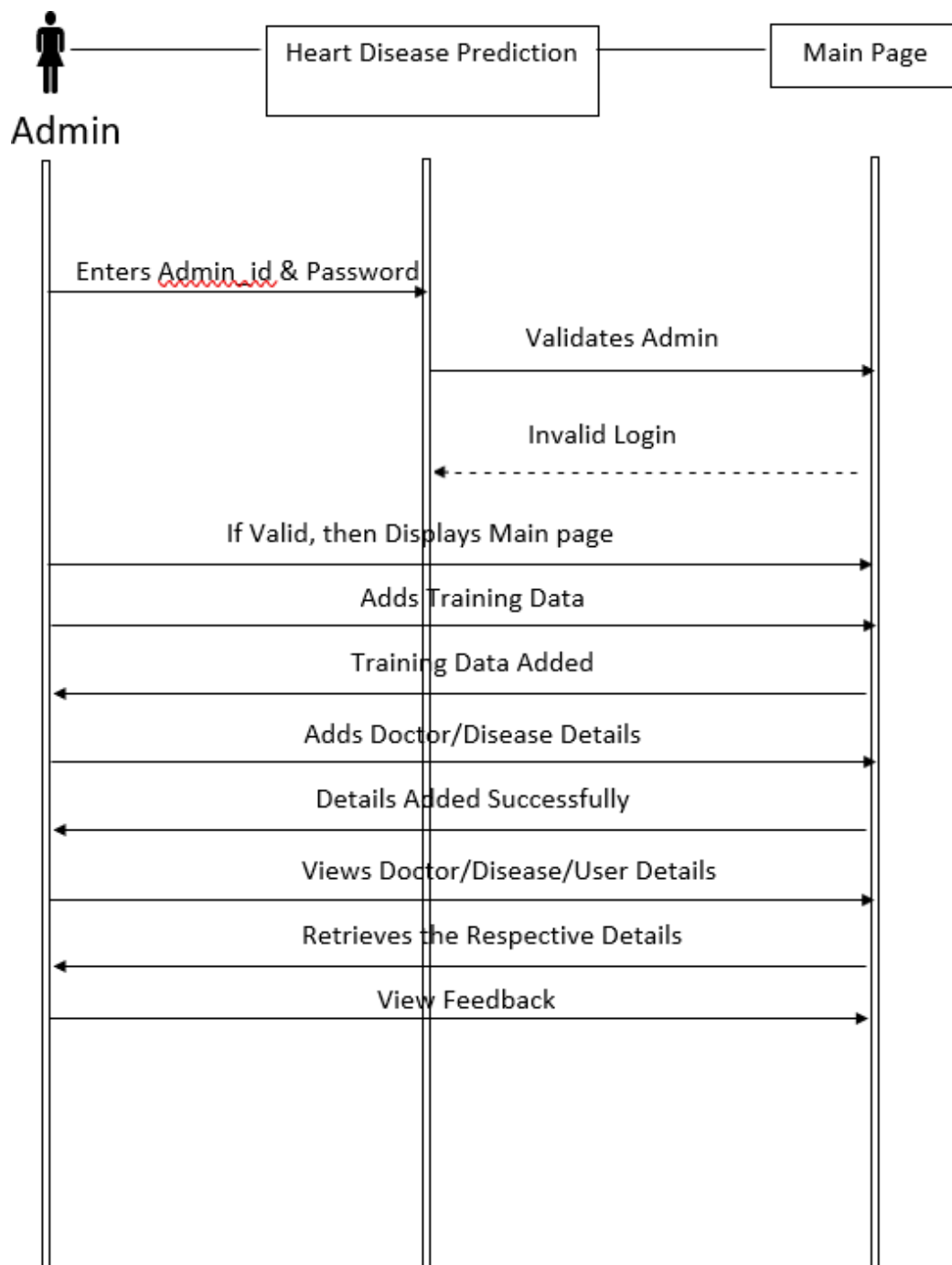


Fig. 4.7 Sequence Diagram

4.6 Data Flow Diagram

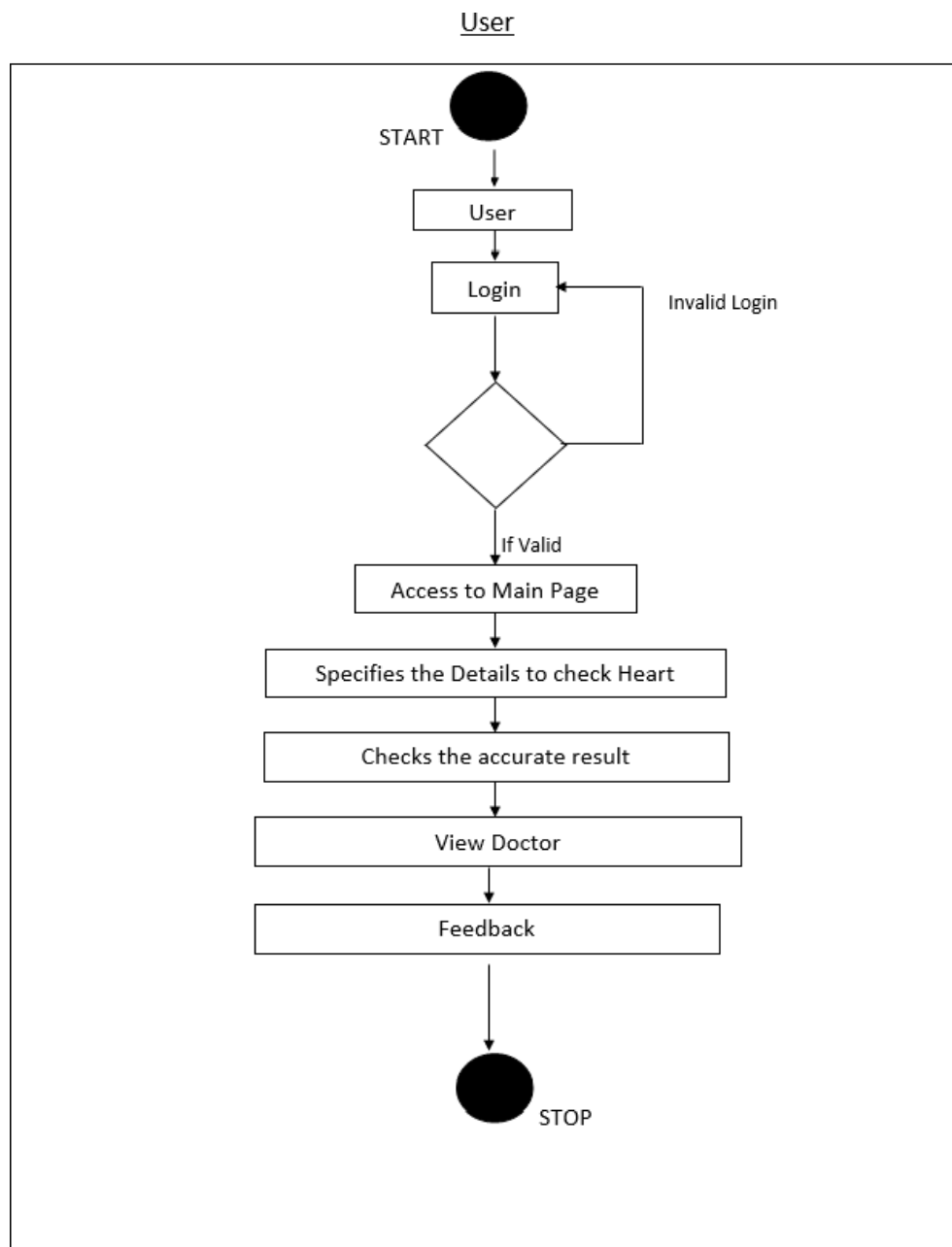


Fig. 4.8 Data Flow Diagram

A data flow diagram shows the way information flows through a process or system. It includes data inputs and outputs, data stores, and the various sub processes the data moves through. DFDs are built using standardized symbols and notation to describe various entities and their relationships. The objective of a DFD is to show the scope and

boundaries of a system as a whole. The diagram alongside represents the data flow diagram of the proposed system. Our project data flow diagrams at user and admin side are shown in Fig. 4.8 and Fig. 4.9.

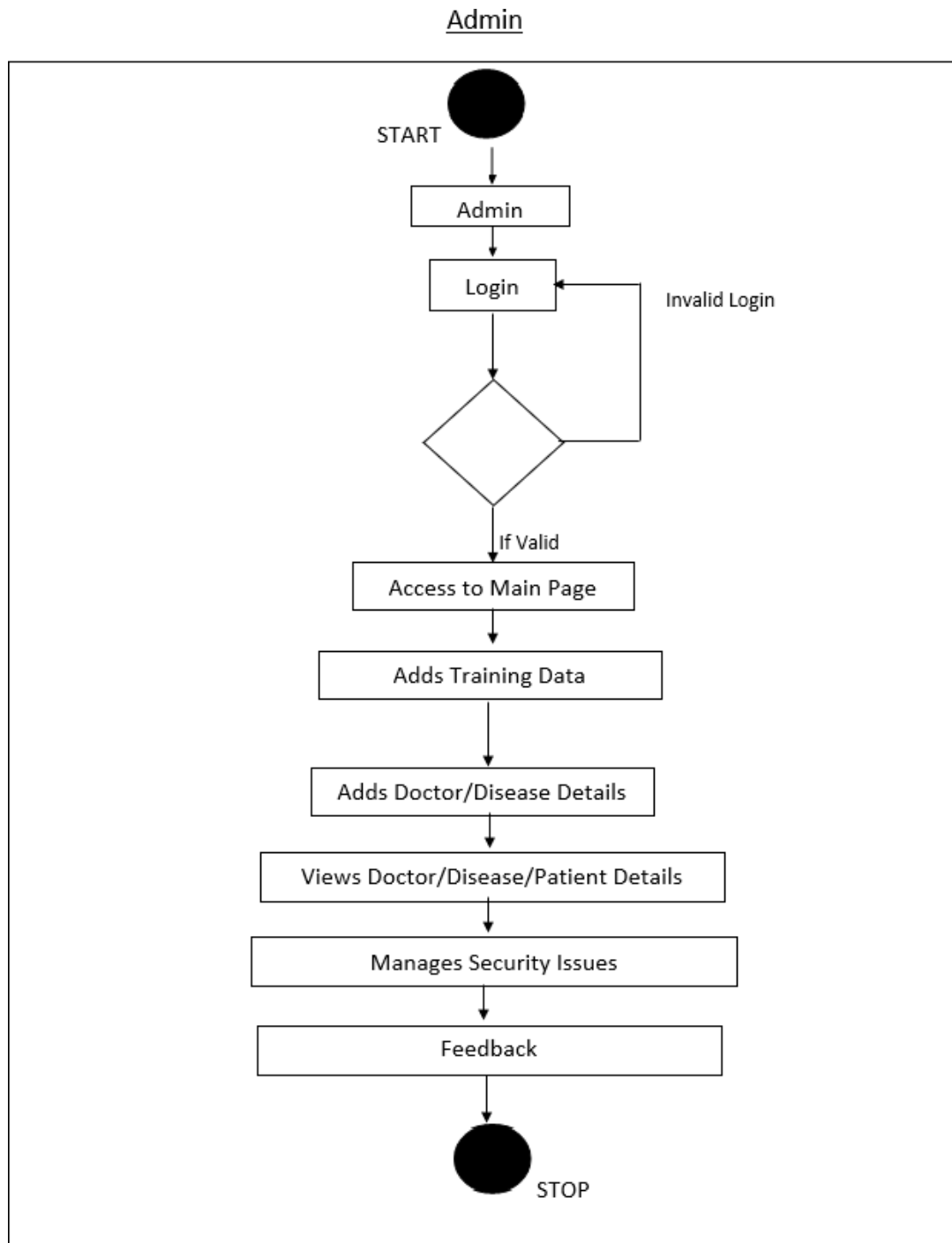


Fig. 4.9 Data Flow Diagram

4.7 Conclusion

In the conducted experiment, the main spotlight is on the domain of healthcare for patients suffering from heart problems by suggesting a novel Fog based Smart Healthcare System created with a fog environment with one master and many slave architectures for spontaneous detection of the heart using machine learning. Here we use decision trees and Naive Bayes techniques for prediction. Fog environment is very flexible and it can be used to provide healthcare as a service and coherently manage the input of the sufferer which is imminent from different IoT devices. Here we are reducing latency or delay in communication and improving efficiency. In the future we would like to develop a one to one communication between the master node and the slave node. And for further increase in security we would like to implement the system with blockchain.

IMPLEMENTATION

CHAPTER 5

5.1 Tools Introduction

- Visual code studio
- Microsoft Sql server management studio
- Android Studio

5.2 Technology Introduction

Visual Studio 2019 software is used for Designing and coding the framework described. SQL Server 2019 manages and optimises the schema of the database, with tables, extracting on demand data using query, for storing data or to keep records or to upgrade the database..

Hardware Requirement:

- Processor with an intel core 5 or any upgraded version.
- Memory: 1 GB RAM
- A disk space of : 50 GB
- Connectivity to the internet

Software Requirement:

- Windows 10
- Visual studio 2019.
- SQL Server 2019.

5.3 Overall view of the project in terms of implementation

FRONT END TECHNOLOGY

Microsoft .NET Framework

It was first featured in the late 1990s by Microsoft to execute primarily on windows os.

The library used to execute is FCL[Framework Class Library], which also supports using two or more programming languages on the same platform.

The environment created for execution is known as Common Language Runtime where a virtual concept is made use to manage all aspects like protection, data management, exception management.

Introduction to ASP.NET Server Controls : The primary controls used in ASP.NET are asp.net server pages. ASP.NET server controls are utilized by ASP.NET page designers to program Web pages.

Any customer entered values to the server are kept by Server controls. This control state will not be put away on the server. keep in mind likewise that customer-side content is not required.

These asp.net server controls can be used in structural jobs such as security, data access, data manipulation, creating master pages and validation. In addition, all the events, methods and properties possessed by the System.Web.UI.Control and WebControl class are inherited by asp.netserver control.

BACK END TECHNOLOGIES:

Microsoft SQL Server: The Microsoft SQL Server is basically a SQL based, client and worker information storage. Each of these words describes a foundation part of the working of a SQL Server.

Database: The database organises the data in a structured manner that can be accessed easily using language specific queries. The data in a database is usually supplied to a specific application that makes use of the data to produce useful business insights. A manufacturing data warehouse can be built to store huge amounts of data.

The database contains tables, attributes, views, and procedures to support the application. The DBMS answers for exhibits the information base construction:

- The structure of data relations can be seen in a simplified manner.
- The information displayed after each query is accurate.
- During the datacenter failures it also provides a system of backup for recovering the data

Examples of servers are:

A particular database can be allocated in the backend for a live worker. An interconnecting segment is used so applications can run for a single client and disclose the information base for an organization. The SQL Server correspondence segment permits execution between a framework executing on the worker node and SQL Server.

The design of the system application allows working with many customers simultaneously and SQL server also supports the same feature to support the application framework. It also provides isolation to the data such that other users cannot modify the data being used by others.

Like a worker in a client/server organization, SQL Server also planned to function; it is likewise supported for filling in a non dependent database straightforwardly on the clients. The versatility and ease of use highlights of Structured Query Language based Server permit this to execute proficiently upon a customer without burning-through an extra number of sources.

Structured Query Language (SQL) : To retrieve the information from a data set, there is a need to utilize a defined order and proclamations that are explained in the DBMS programming language syntax. There exist many programming languages, which are utilized along with social information bases.

1.1.1 Features of SQL Server : SQL Server of Microsoft provides a bunch of characteristics to the client which offers the below advantages:

- The deployment, installation and use of the application framework is effortless : SQL Server is embedded with a bunch of advanced device frameworks that enhance the capacity to implement, channel, supervise, and use SQL Server across a few locales.
- Scalability : SQL is supported by all versions of windows, linux, unix and other forms of system, it can also be scaled to other forms of databases available like RedShift etc.
- Data warehousing : The SQL query is structured for the ease of use for the clients it resembles more to the format of English.
- Integration of system with different server soft wares: Structured Query Language Server can integrate with almost all third party applications like email, the Internet, and Windowsos .

SQL Server manages various databases, with each informational index stores either data with relations or data disengaged in various informational collections. With a real world example, a laborer with one informational index that accommodates singular information and another one that saves dependent data. Contrarily, one information base can be loaded with live client request information, and another with a related data set can store chronicled client arrangements that are utilized for yearly revealing. An information base, comprehends the pieces of a data set and to plan the parts to assure that the data set works in a consistent format after it is carried out.

5.6 Explanation of Algorithm and How it is being Implemented

- Naive bayes classifier takes the assistance of the bayes theorem.
- This algorithm predicts the probability at a particular data point of that particular class for which the data belongs. The class having the highest probability contemplates being the most likely class to be shown in Fig. 7.1 and Fig. 7.2.
- To classify the classes, we compute X with respect to Y. This can be followed by carrying out the equation below:

$$P(Y = y|X = (x_1, x_2, \dots, x_k))$$

Find the most likely $y \rightarrow \text{prediction} = \text{argmax}_y P(Y = y|X = (x_1, x_2, \dots, x_k))$

The Naive Bayes classifier calculates the probability of an event in the following steps:

- Step 1: Calculate the prior probability for given class labels
- Step 2: Find Likelihood probability with each attribute for each class
- Step 3: Put these values in Bayes Formula and calculate posterior probability.
- Step 4: See which class has a higher probability, given the input belongs to the higher probability class.

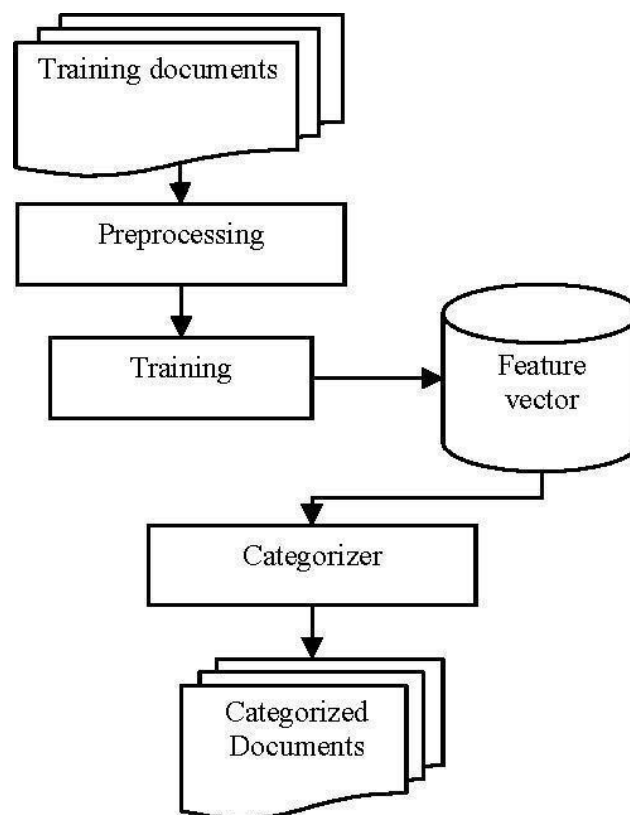


Fig. 5.1 General Flow of Algorithm on Dataset

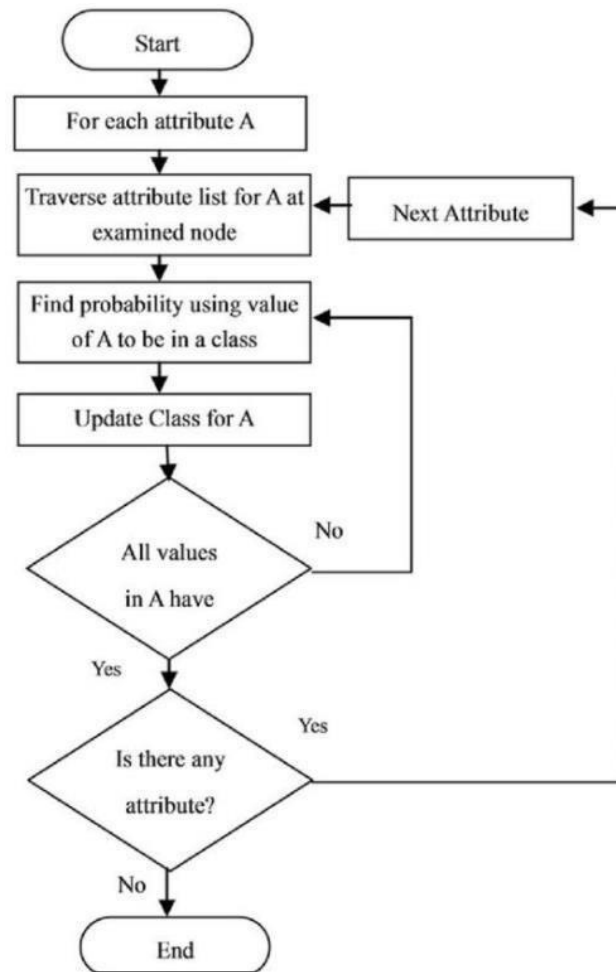


Fig. 5.2 Flow Chart of Naive Bayes Algorithm

5.4 Information about the implementation of Modules

The system comprises 2 major modules as follows:

Admin Module:

1. Add Training Data to the model
2. Provide doctor details
3. View User Details that is previously saved
4. Sight Feedback provided by the customers
5. Sight the Doctor Details

6. Sight the Training Data as well as add any new data if required.

User Module:

1. Register (With Details like Age, Sex, etc.)
2. Check Heart by providing Details like
 - Age in Years (29-62)
 - Gender (0- male, 1-female)
 - Chest Pain Type (1- typical angina, 2- atypical angina etc.,)
 - Fasting Blood Sugar (1- present, 0- absent)
 - Resting Electrographic Results (Restecg; 1- having, 0- normal)
 - Exercise Induced Angina (Exang; 1- present, 0-absent)
 - The slope of the peak exercise ST segment
 - Number of major vessels coloured by fluoroscopy (0-3 vessels count)
 - Thal- Thalassemia
 - Trest Blood Pressure (a numeric value)
 - Serum Cholesterol in mg/dl (a numeric value)
 - Maximum heart rate achieved (Thalach; 140,173)
 - ST depression induced by exercise (Old peak)
3. System will accordingly view the Doctor to consult.
4. Provide feedback
5. Sight doctor

5.5 Conclusion

Naive Bayes' calculations have been used extensively in spam filtering, determining the feelings of the crowd, giving suggestions based on the study of history and so forth. They are quick and elementary to execute, yet their greatest obstacle is that the prerequisite of index is to be autonomous. Complementing this with the decision tree will give us the most consequential result prediction.

Choice Support in the Heart Disease Prediction System is likewise evolved utilizing the Naive Bayesian Classification strategy. Treatment records of millions of patients can be put away and mechanized and information mining strategies may help in several significant and basic inquiries identified with medical care. Naive Bayes order can be utilized as a best choiceemotionally supportive network.

5.6 Code Implementation

MainLogin.aspx

```
<% @ Page Title="" Language="C#" MasterPageFile="~/MasterPage.master"
AutoEventWireup="true" CodeFile="MainLogin.aspx.cs" Inherits="MainLogin" %>
```

```
<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">
```

```
<style type="text/css">
```

```
.auto-style1 {
```

```
width: 100%;
```

```
}
```

```
.auto-style2 {
```

```
height: 125px;
```

```
width: 33%;
```

```
}
```

```
.auto-style3 {
```

```
height: 25px;
```

```
width: 33%;
```

```
}
```

```
</style>
```

```
</asp:Content>
```

```
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1"
```

```
Runat="Server">
```

```
<br />
```

```
<br />
```

```
<table style="border: 1px solid #C0C0C0" width="80%">
```

```
<tr>
<td height="250" align="center" class="auto-style2">
<asp:ImageButton ID="ImageButton1" runat="server"
ImageUrl="/images/Admin.jpg" PostBackUrl="/AdminLogin.aspx" Width="170px"
/>
</td>

<td width="33.35%" align="center">
<asp:ImageButton ID="ImageButton3" runat="server" ImageUrl="/images/User.png"
PostBackUrl="/UserLogin.aspx" Width="180px" />
</td>
</tr>
<tr>
<td align="center" class="auto-style3">
<asp:LinkButton ID="LinkButton1" runat="server" Font-Bold="True" Font-
Size="Large" Font-Underline="False" ForeColor="#333333"
PostBackUrl="~/AdminLogin.aspx" style="font-family: candara; font-size: xx-
large">Admin Login</asp:LinkButton>
</td>

<td width="33.35%" align="center" height="20">
<asp:LinkButton ID="LinkButton3" runat="server" Font-Bold="True" Font-
Size="Large" Font-Underline="False" ForeColor="#333333"
PostBackUrl="~/UserLogin.aspx" style="font-family: candara; font-size: xx-
large">User Login</asp:LinkButton>
</td>
</tr>
</table>
<br />
<br />
</asp:Content>
```

Web.config

```
<?xml version="1.0"?>
<configuration>
<connectionStrings>
<add      name="Heart"      connectionString="Data      Source=SG2NWPLS14SQL-
v09.shr.prod.sin2.secureserver.net;Initial
Catalog=Healthfog;User
ID=Healthfog;Password=Ws27$5xm" providerName="System.Data.SqlClient"/>
</connectionStrings>
```

The following attributes can be set on the <httpRuntime> tag.

```
<system.Web>
<httpRuntime targetFramework="4.5" />
</system.Web>
-->
<system.web>
<compilation debug="true" targetFramework="4.5"/>
<authentication mode="Forms">
<forms loginUrl="~/Account/Login.aspx" timeout="2880"/>
</authentication>
<membership>
<providers>
<clear/>
<add
name="AspNetSqlMembershipProvider"
type="System.Web.Security.SqlMembershipProvider"
connectionStringName="ApplicationServices" enablePasswordRetrieval="false"
enablePasswordReset="true"
requiresQuestionAndAnswer="false"
requiresUniqueEmail="false"
maxInvalidPasswordAttempts="5"
minRequiredPasswordLength="6"      minRequiredNonalphanumericCharacters="0"
passwordAttemptWindow="10" applicationName="/" />
</providers>
```

```
</membership>
<profile>
<providers>
<clear/>
<add
name="AspNetSqlProfileProvider" type="System.Web.Profile.SqlProfileProvider"
connectionStringName="ApplicationServices" applicationName="/" />
</providers>
</profile>
<roleManager enabled="false">
<providers>
<clear/>
<add
name="AspNetSqlRoleProvider" type="System.Web.Security.SqlRoleProvider"
connectionStringName="ApplicationServices" applicationName="/" />
<add
name="AspNetWindowsTokenRoleProvider"
type="System.Web.Security.WindowsTokenRoleProvider" applicationName="/" />
</providers>
</roleManager>
<pages controlRenderingCompatibilityVersion="4.0" />
</system.web>
<system.webServer>
<modules runAllManagedModulesForAllRequests="true" />
</system.webServer>
</configuration>
```

CheckHeart.aspx

```
<% @ Page Title="" Language="C#" MasterPageFile="~/MasterPage.master"
AutoEventWireup="true" CodeFile="CheckHeart.aspx.cs" Inherits="CheckHeart" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">
<style type="text/css">
```



```
.style1
{
font-size: xx-large;
}
.intLog
{
-webkit-border-radius: 10px;
-moz-border-radius: 10px;
}
.style9
{
height: 40px;
}
.style10
{
-webkit-border-radius: 10px;
-moz-border-radius: 10px;
font-family: Calibri;
}
.style11
{
font-family: Calibri;
font-size: large;
color: #333333;
font-weight: 700;
}
.style12
{
-webkit-border-radius: 10px;
-moz-border-radius: 10px;
font-family: Calibri;
font-size: large;
}
```

```
.style13
{
width: 20%;
}
.style14
{
width: 20%;
}
.auto-style1 {
height: 23px;
}
</style>
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1"
Runat="Server">
<asp:Panel ID="Panel1" runat="server">
<asp:Label ID="Label1" runat="server" Text="Results :" style="font-size: x-
large"></asp:Label>
<asp:Label ID="Label2" runat="server" Text="" style="font-size: x-
large"></asp:Label>
</asp:Panel>
<br />
<table style="border: 1px solid #C0C0C0; background-image: url('Images/bg.jpg'); "
width="60%" class="intLog">
<tr>
<td colspan="3"
style="background-image: url('Images/blurred.jpg')"
align="center" class="intLog" bgcolor="#CC3300">
<asp:Label ID="Label3" runat="server" style="font-family: calibri; font-size: xx-large;
color: #FFFFFF;" Text="-- Check your Heart --"></asp:Label>
</td>
</tr>
</tr>
```

```
<td colspan="3" align="right" style="color: #FF0000; font-size: small">
&nbsp;</td>
</tr>
<tr>
<td colspan="3" align="center" style="color: #FF0000; font-size: small">
* Kindly fill up all the details given below<br />
</td>
</tr>
<tr>
<td align="right" class="style13">
&nbsp;</td>
<td align="left">
&nbsp;</td>
<td width="33%" align="left" class="style14">
&nbsp;</td>
</tr>
<tr>
<td align="right" class="style13">
&nbsp;</td>
<td align="left">
<asp:TextBox ID="TextBox14" runat="server" CssClass="style12"
placeholder=" Chest Pain Type" required="Required"
Width="100%"></asp:TextBox>
</td>
<td width="33%" align="left" class="style14">
&nbsp;</td>
</tr>
<tr>
<td align="right" class="style13">
&nbsp;</td>
<td align="left">
&nbsp;</td>
<td width="33%" align="left" class="style14">
```

```
&nbsp;</td>
</tr>
<tr>
<td align="right" class="style13">
</td>
<td class="style9" align="left" >
<asp:TextBox ID="TextBox4" runat="server" CssClass="style12"
placeholder=" Fasting Blood Sugar" required="Required"
Width="100%"></asp:TextBox>
</td>
<td width="33%" align="left" class="style14">
&nbsp;</td>
</tr>

<tr>
<td align="right" class="style13">
&nbsp;</td>
<td>
&nbsp;</td>
</tr>

<tr>
<td align="right" class="style13">
&nbsp;</td>
<td>
<asp:TextBox ID="TextBox5" runat="server" CssClass="style12"
placeholder=" Resting Electrographic" required="Required"
Width="100%"></asp:TextBox>
</td>
</tr>
<tr>
<td align="right" class="style13">
&nbsp;</td>
```

```
<td>
&nbsp;</td>
<td width="33%" class="style14">
&nbsp;</td>
</tr>
<tr>
<td align="right" class="style13">
&nbsp;</td>
<td>
<asp:TextBox ID="TextBox6" runat="server" CssClass="style12"
placeholder=" Exercise Induced Angina" required="Required"
TextMode="SingleLine" Width="100%"></asp:TextBox>
</td>
<td width="33%" class="style14">
&nbsp;</td>
</tr>
<tr>
<td align="right" class="style13">
&nbsp;</td>
<td>
&nbsp;</td>
<td width="33%" class="style14">
&nbsp;</td>
</tr>
<tr>
<td align="right" class="style13">
</td>
<td>
<asp:TextBox ID="TextBox7" runat="server" CssClass="style12"
placeholder=" Slope" required="Required" TextMode="SingleLine"
Width="100%"></asp:TextBox>
</td>
<td width="33%" class="style14">
```

```
&nbsp;</td>
</tr>
<tr>
<td align="right" class="style13">
&nbsp;</td>
<td>
&nbsp;</td>
<td width="33%" class="style14">
&nbsp;</td>
</tr>
<tr>
<td align="right" class="style13">
&nbsp;</td>
<td>
<asp:TextBox ID="TextBox8" runat="server" CssClass="style12"
placeholder=" No. of Major Vessels(CA) " required="Required"
TextMode="SingleLine" Width="100%"></asp:TextBox>
</td>
<td width="33%" class="style14">
&nbsp;</td>
</tr>
<tr>
<td align="right" class="style13">
&nbsp;</td>
<td>
&nbsp;</td>
<td width="33%" class="style14">
&nbsp;</td>
</tr>
<tr>
<td align="right" class="style13">
&nbsp;</td>
<td>
```

```
<asp:TextBox ID="TextBox9" runat="server" CssClass="style12"
placeholder=" Thal" required="Required" TextMode="SingleLine"
Width="100%"></asp:TextBox>
</td>
<td width="33%" class="style14">
&nbsp;</td>
</tr>
<tr>
<td align="right" class="style13">
&nbsp;</td>
<td>
&nbsp;</td>
<td width="33%" class="style14">
&nbsp;</td>
</tr>
<tr>
<td align="right" class="style13">
&nbsp;</td>
<td>
<asp:TextBox ID="TextBox10" runat="server" CssClass="style12"
placeholder=" Trest Blood Pressure" required="Required" TextMode="SingleLine"
Width="100%"></asp:TextBox>
</td>
<td width="33%" class="style14">
&nbsp;</td>
</tr>
<tr>
<td align="right" class="style13">
&nbsp;</td>
<td>
&nbsp;</td>
<td width="33%" class="style14">
&nbsp;</td>
```

```
</tr>
<tr>
<td align="right" class="style13">
&nbsp;</td>
<td>
<asp:TextBox ID="TextBox11" runat="server" CssClass="style12"
placeholder=" Serum Cholesterol" required="Required" TextMode="SingleLine"
Width="100%"></asp:TextBox>
</td>
<td width="33%" class="style14">
&nbsp;</td>
</tr>
<tr>
<td align="right" class="style13">
&nbsp;</td>
<td>
&nbsp;</td>
<td width="33%" class="style14">
&nbsp;</td>
</tr>
<tr>
<td align="right" class="style13">
&nbsp;</td>
<td>
<asp:TextBox ID="TextBox12" runat="server" CssClass="style12"
placeholder=" Maximum Heart Rate achieved(Thalach)" required="Required"
TextMode="SingleLine" Width="100%"></asp:TextBox>
</td>
<td width="33%" class="style14">
&nbsp;</td>
</tr>
<tr>
<td align="right" class="style13">
```



```
&nbsp;</td>
<td>
&nbsp;</td>
<td width="33%" class="style14">
&nbsp;</td>
</tr>
<tr>
<td align="right" class="style13">
&nbsp;</td>
<td>
<asp:TextBox ID="TextBox13" runat="server" CssClass="style12"
placeholder=" ST Depression Induced by Exercise(Oldpeak)" required="Required"
TextMode="SingleLine" Width="100%"></asp:TextBox>
</td>
<td width="33%" class="style14">
&nbsp;</td>
</tr>
<tr>
<td align="right" class="style13">
&nbsp;</td>
<td>
&nbsp;</td>
<td width="33%" class="style14">
&nbsp;</td>
</tr>
<tr>
<td colspan="3" class="auto-style1">
</td>
</tr>
<tr>
<td align="center" colspan="3">
<asp:Button ID="btnAnalyse" runat="server" Text="Analyse Heart"
```

```
CssClass="style10"
Height="38px" Width="200px" Font-Size="X-Large" ForeColor="#333333"
BorderColor="#009933" OnClick="btnAnalyse_Click"/>

</td>
</tr>
<tr>
<td align="right" class="style13">
&nbsp;  </td>
<td>
<asp:Label ID="lblAge" runat="server" Visible="False"></asp:Label>
<asp:Label ID="lblSex" runat="server" Visible="False"></asp:Label>
</td>
<td width="20%">
&nbsp;  </td>
</tr>
</table>
</asp:Content>
```

CHAPTER 6

OUTPUT SNAPSHOTS

6.1 Result Snapshots

The Fig. 6.1 shows home page of the application which consists of login options for admin and user. The admin logs in using admin login and user logs in using the user login.

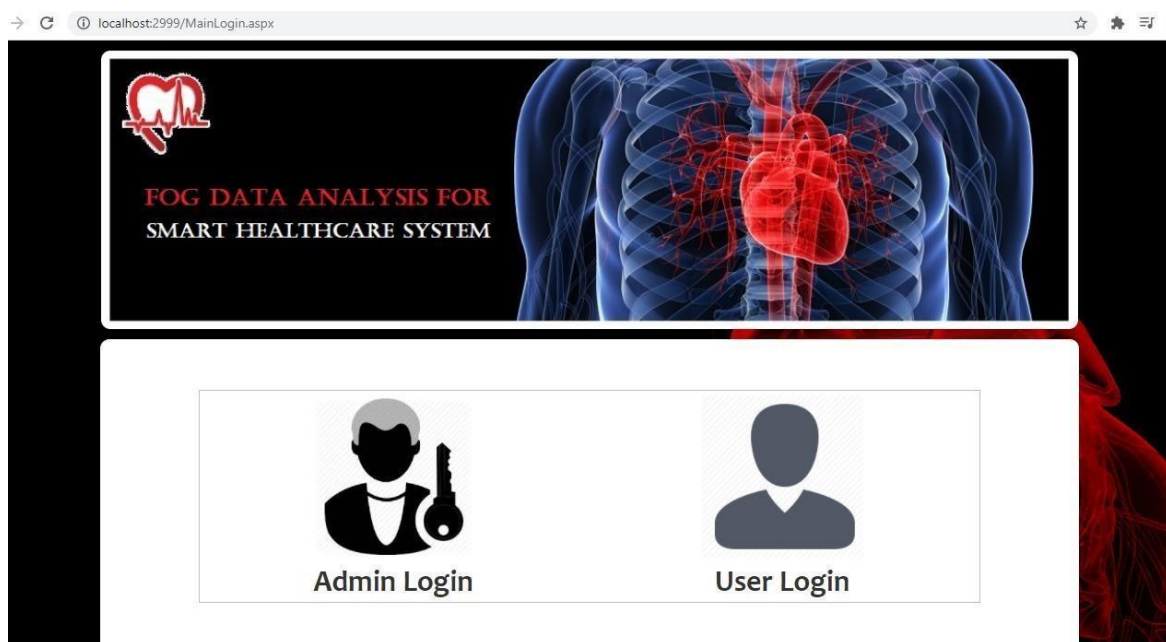
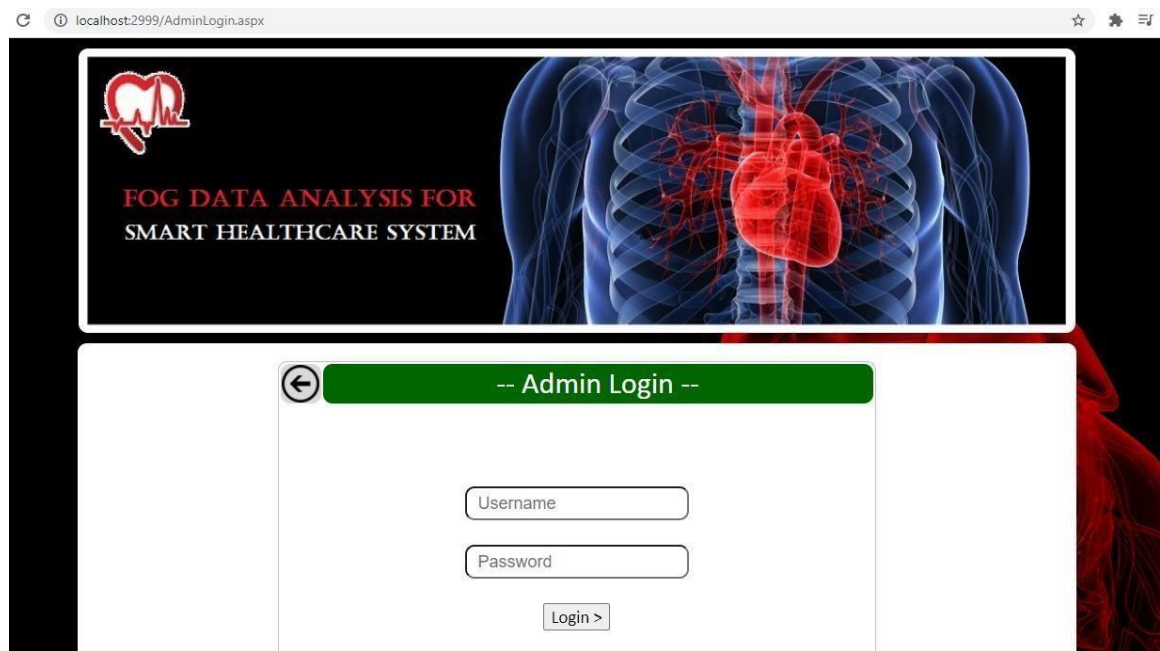


Fig. 6.1.1 Home page

Once the admin login icon is clicked the admin login page will appear. Fig. 6.2 is the snapshot of the admin page. the admin may login using his credentials.



localhost:2999/AdminLogin.aspx

FOG DATA ANALYSIS FOR
SMART HEALTHCARE SYSTEM

-- Admin Login --

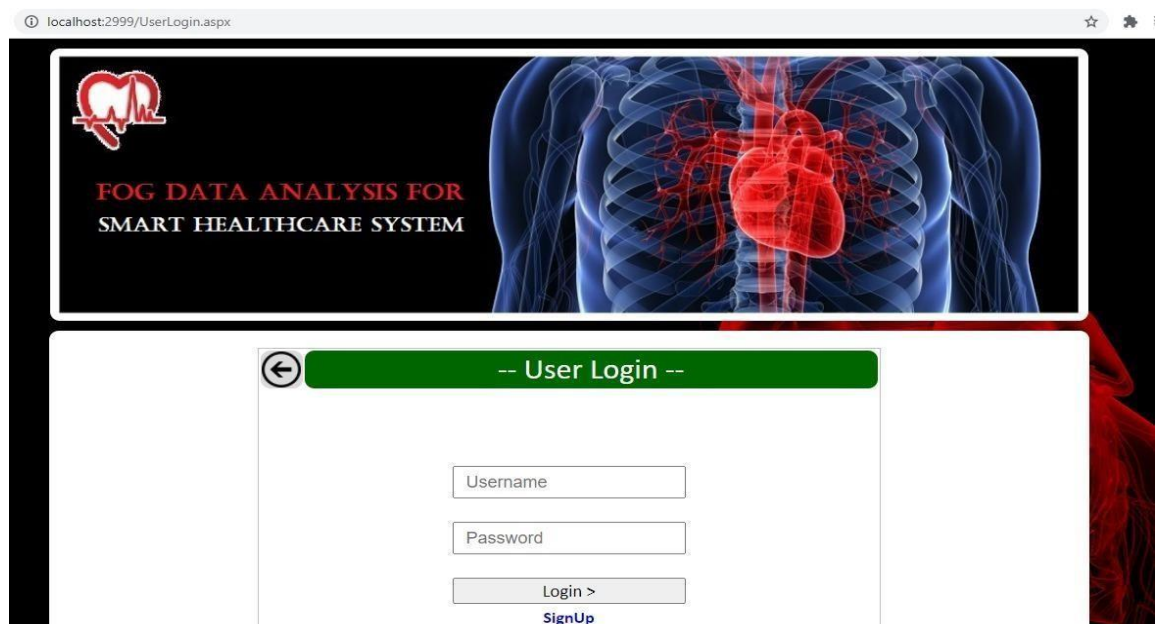
Username

Password

Login >

Fig. 6.1.2 Admin Login

The registered user has to enter his details like username and password and can login. If the user is not registered he can register in the new user option using sign up. Fig. 6.3 is the user login page.



localhost:2999/UserLogin.aspx

FOG DATA ANALYSIS FOR
SMART HEALTHCARE SYSTEM

-- User Login --

Username

Password

Login >
SignUp

Fig. 6.1.3 User Login

The new user has to enter details like name, address, mobile no., email id, age, gender and can give a password and register. the Fig. 6.4 is the user registration page.

localhost:2999/Register.aspx

-- Register --

Id : 1004

Name

Address

Mobile No.

Email Id

Your Age

Gender : ☐ Male ☐ Female

Password

Submit

Fig. 6.1.4 New User Registration Page(Sign Up Page)

The Fig. 9.5 shows the example for new user registration.

localhost:2999/Register.aspx

-- Register --

Id : 1004

John

Bangalore

8856231067

john111@gmail.com

48

Gender : ☒ Male ☐ Female

....

Submit

Fog Data Analysis For Smart HealthCare System || Your Name

Fig. 6.1.5 New User Registration Page(Sign Up Page)

The Fig. 6.1.6 shows the doctor's details page.

localhost:2999/ViewTrainingData.aspx

Add DoctorAdd Training DataView UserView Training DataView FeedbackLogout

-- Doctor Details --

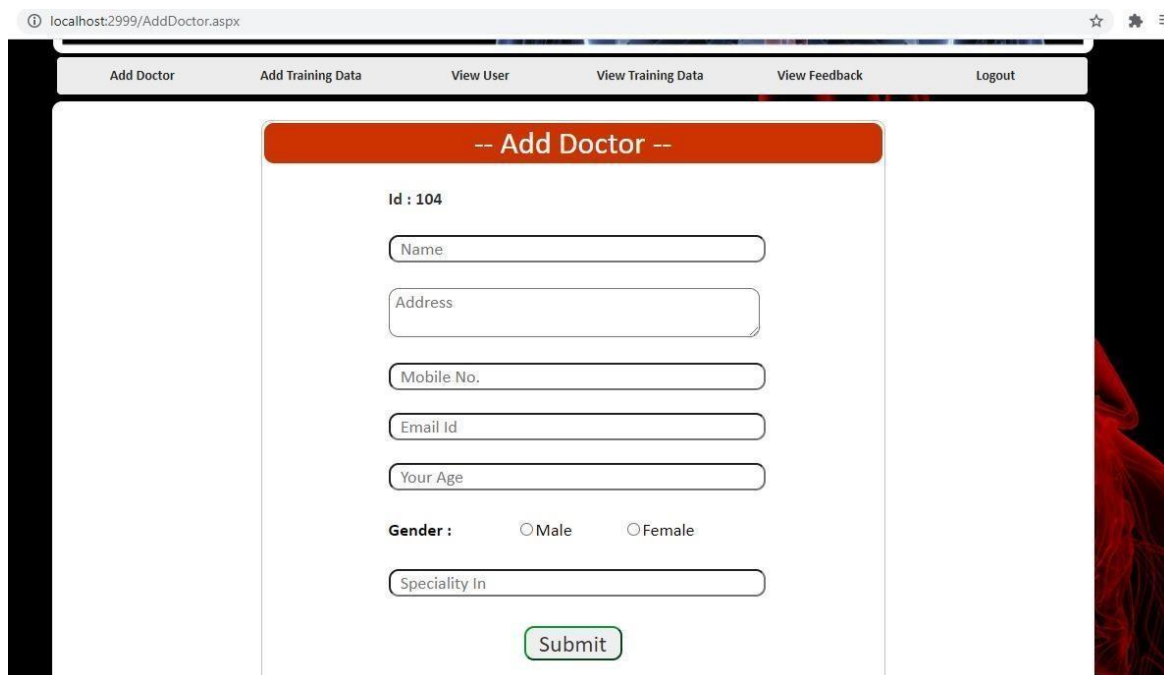
Disease	Age	Gender	Chest Pain	Sugar	Restecg	Exang	Slope	CA	Thal	BP	Cholesterol	Thalach	Old Peak
Coronary Artery Disease	25	0	2	100	2	1	2	3	3	120	50	80	100
High Blood Pressure	40	0	3	120	1	0	3	0	7	300	70	150	80
Congestive Heart Failure	60	1	1	110	0	1	1	2	6	150	40	98	120
Stroke	75	1	3	165	2	0	2	0	3	132	55	88	121
Congestive Heart Failure	60	1	1	110	0	1	1	2	6	150	40	98	120
Congestive Heart Failure	40	1	0	80	1	3	2	0	3	110	150	70	110
Congestive Heart Failure	22	1	3	105	0	7	0	1	2	125	115	92	99
Congestive Heart Failure	30	0	0	84	0	0	2	3	3	111	80	64	88
Stroke	55	1	2	54	2	5	1	1	1	170	101	67	165
Congestive Heart Failure	70	0	3	45	1	7	0	0	0	80	79	54	84
Stroke	80	0	1	54	2	3	2	3	2	88	158	89	65
Congestive Heart Failure	27	1	3	56	1	7	1	0	3	125	89	78	179
Stroke	69	1	2	125	2	5	0	1	0	166	156	92	122
Congestive Heart Failure	48	0	0	88	0	3	2	1	1	98	147	100	110

Fog Data Analysis For Smart HealthCare System || Your Name

Fig. 6.1.6: Doctor Information Page

And we can also add a new doctor to the list, by giving details like name , address, mobilenumber, email id , age , gender, and filling the speciality of the field handled by the doctor.

Fig. 6.1.7 is the add doctor page.



-- Add Doctor --

Id : 104

Name

Address

Mobile No.

Email Id

Your Age

Gender : ☐ Male ☐ Female

Speciality In

Submit

Fig. 6.1.7 New Doctor Registration Page

We can directly add training data to the database through the add training data page.

Fig. 6.1.8 shows the add training data page.

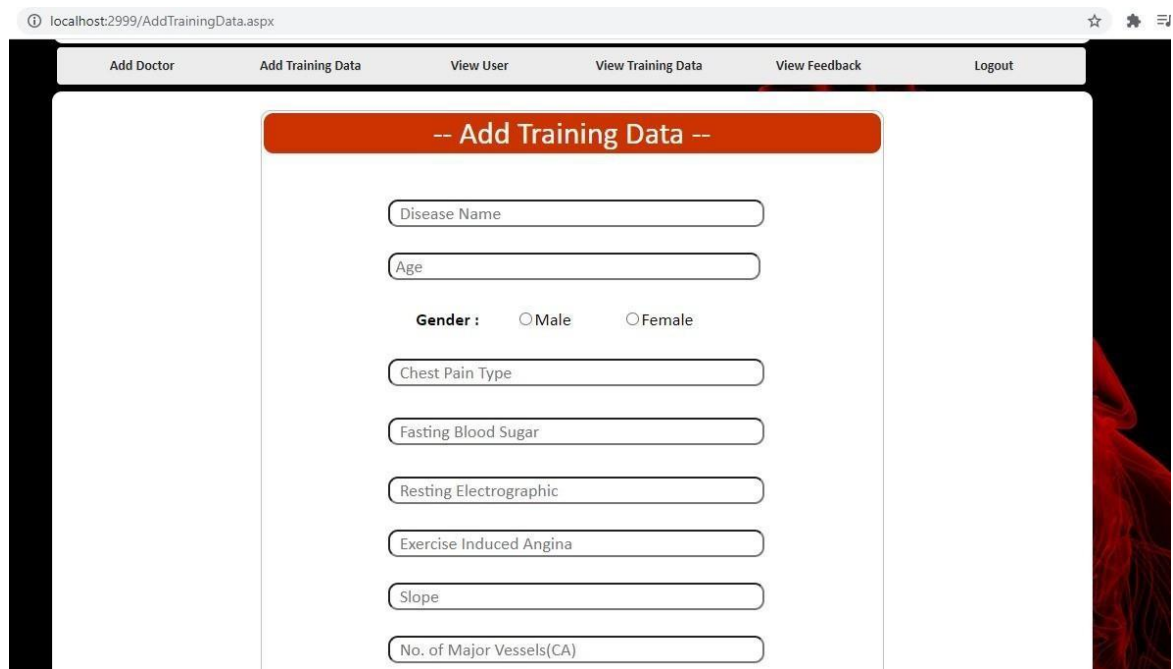


Fig. 6.1.8 Adding new data to database

Fig. 6.1.9 shows the output for given test data.

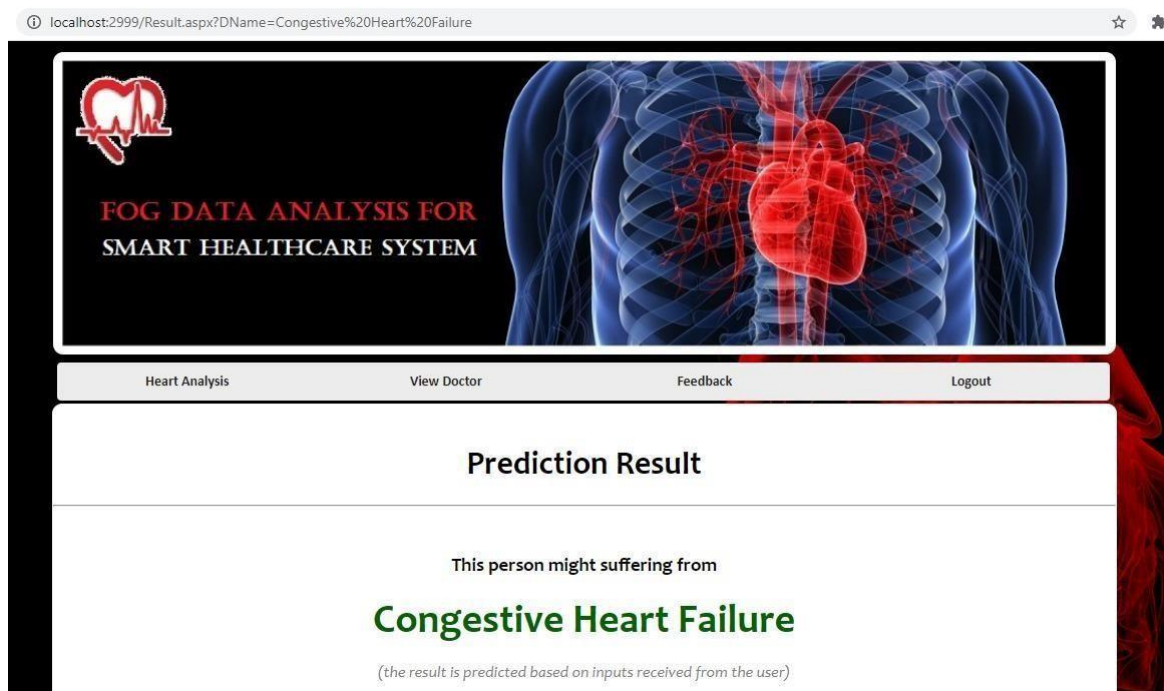


Fig. 6.1.9 Analysed Result Page

We can see the disease result from the input given, and also symptoms and treatment given by the system. Fig. 6.1.10 is the snapshot of the same.

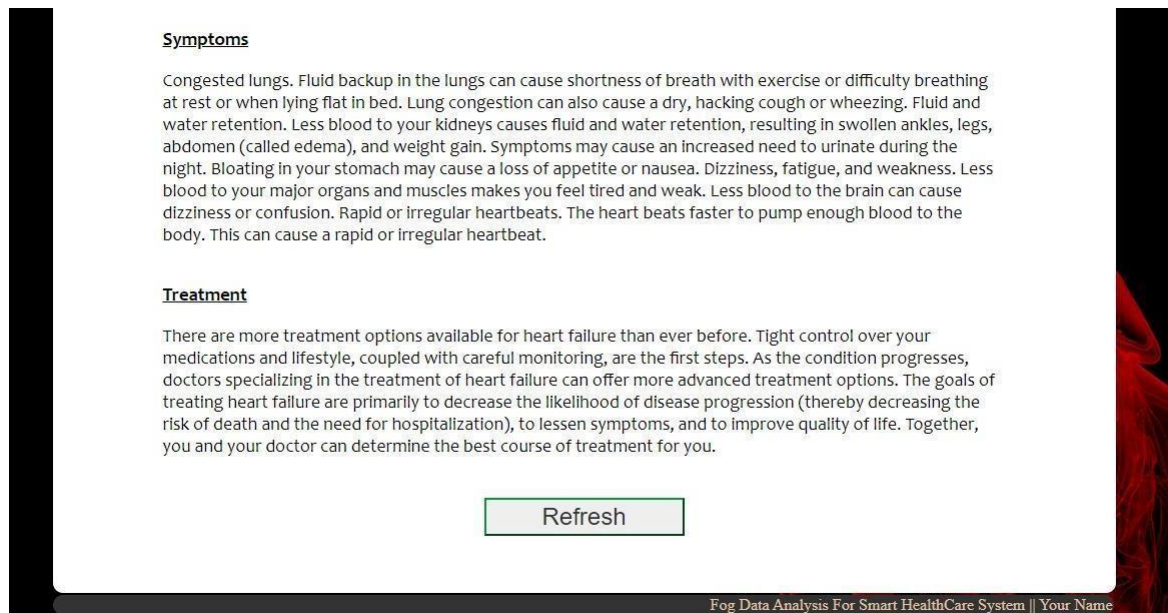


Fig. 6.1.10 Analysed Result Page

Disease results will be sent as notification to the slave node (User's android application) from the master node. Fig. 6.1.11 is the picture of the notification in the mobile application.

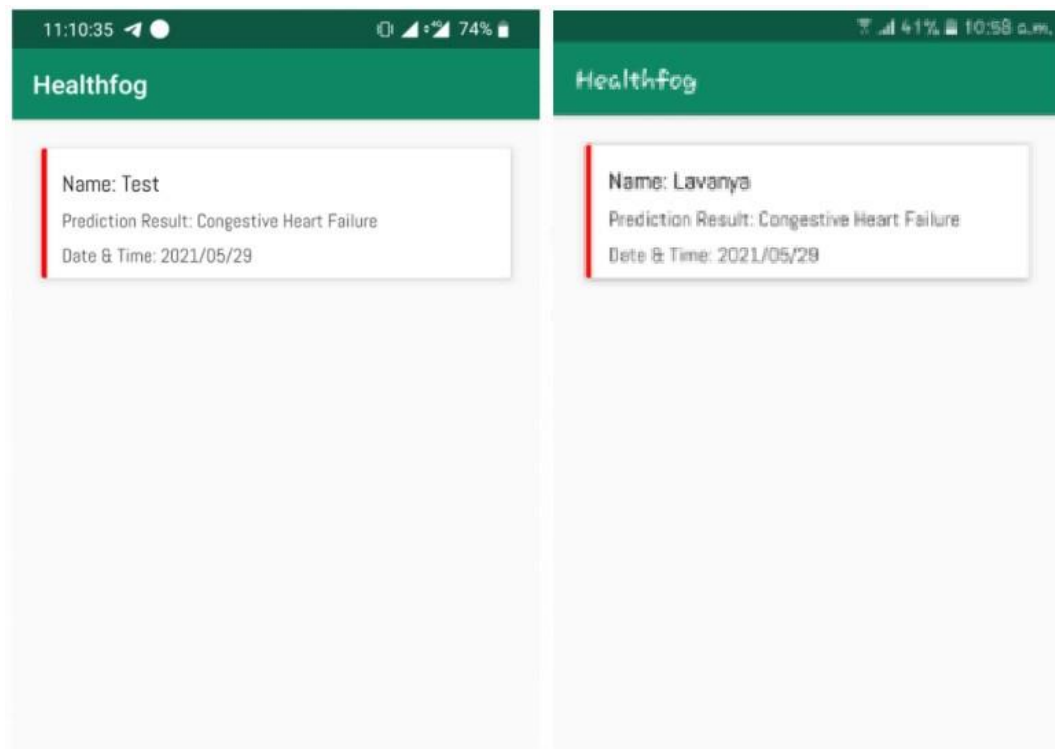


Fig. 6.1.11 View of the Slave node (User's android application)

6.2 Comparison Table

Confusion matrix drawn for a random sample: This is used to find the performance of any classification model. It is carried out over the test data set where we are aware of the true values as shown in Table 9.1 and Table 9.2.

Here for a known sample of 36 random samples from the train data set we have drawn a confusion matrix and found out the accuracy. We see that in machine learning, when it is required to solve any problem that includes statistical classification, this confusion matrix would be described as an error matrix. This table helps us to analyse the performance of any algorithm required.

True data V/s Predicted data

	RHE	HBP	CHF	STR	MHA	HVD
RHE	3	0	4	0	0	0
HBP	0	4	0	4	0	0
CHF	1	0	8	0	0	0
STR	0	0	4	0	0	0
MH A	0	0	0	0	4	0
HVD	0	0	0	0	0	4

Table 6.2.1 Confusion matrix

Random samples taken , $N=36$

Accuracy is given by the formula $(TP+TN)/N * 100$

disease	TP	TN	FP	FN	ACCURACY
RHE	3	28	4	1	0.8811
HBP	4	28	4	0	0.8888
CHF	8	27	1	0	0.9722
STR	4	28	0	4	0.8888
MHA	4	32	0	0	0.1
HVD	4	32	0	0	0.1

Table 6.2.2 Accuracy calculation

Average accuracy is 0.9351

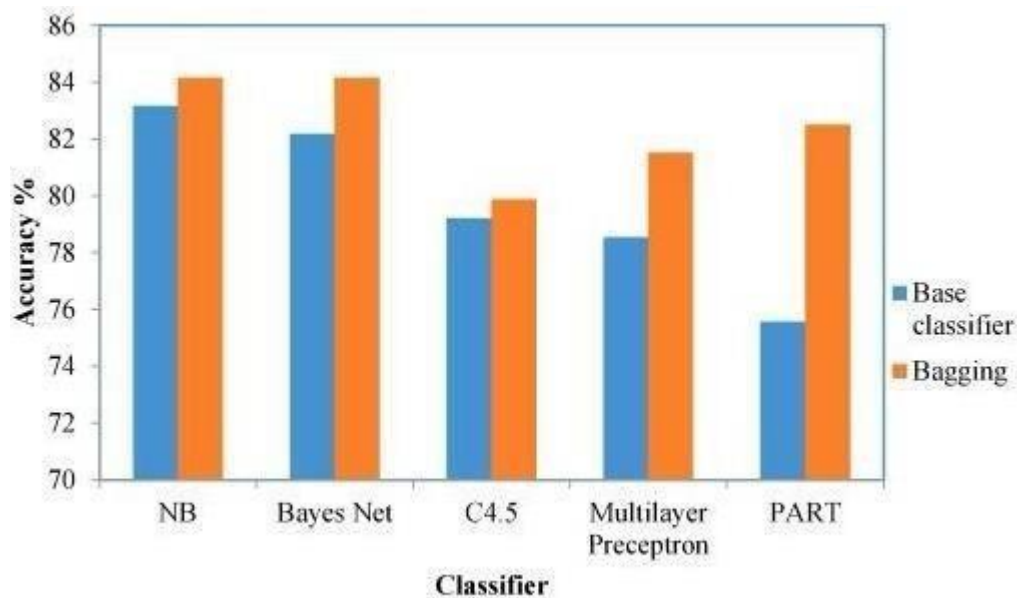


Fig. 6.2.1 Comparison of the Naive Bayes algorithm to other algorithms.[2]

When naive bayes is compared to various other algorithms we see that it has better accuracy than others.

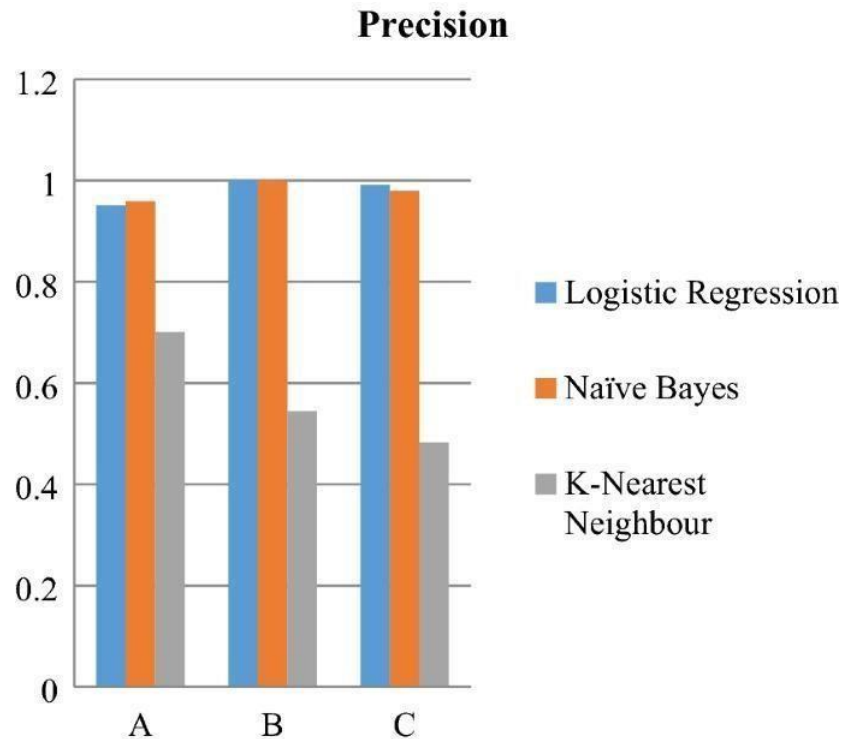


Fig. 6.2.2 Comparison of the Naive Bayse algorithm to other algorithms. [2]

The above graph shows the comparison of naive bayes with knn and logistic regression on their precision.

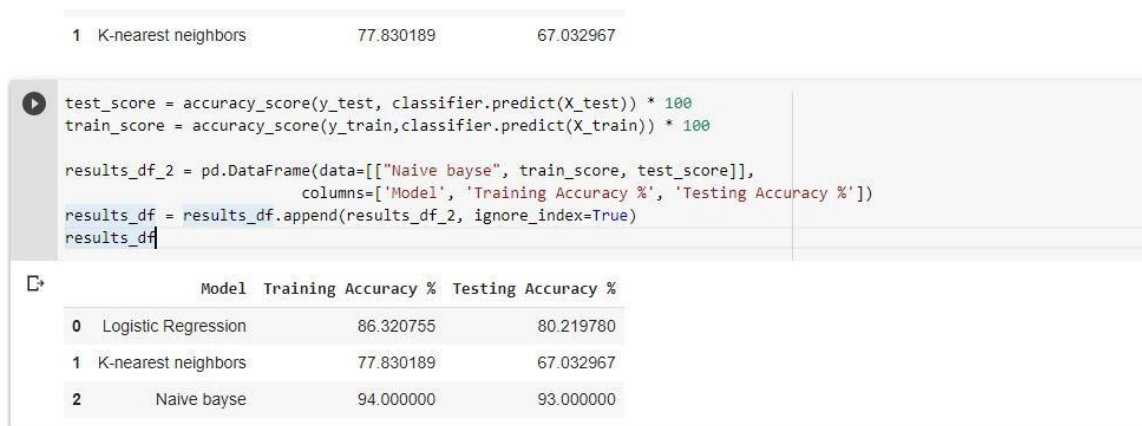


Fig. 6.2.3 Comparison of the Naive Bayse algorithm to other algorithms in Google Colabusing the Heart data set.

After applying Logistic regression, K-nearest neighbours and Naive Bayse algorithm, we have achieved a testing accuracy of 80%,67% and 93% respectively and thus we conclude that NaiveBayse algorithm has a better accuracy.

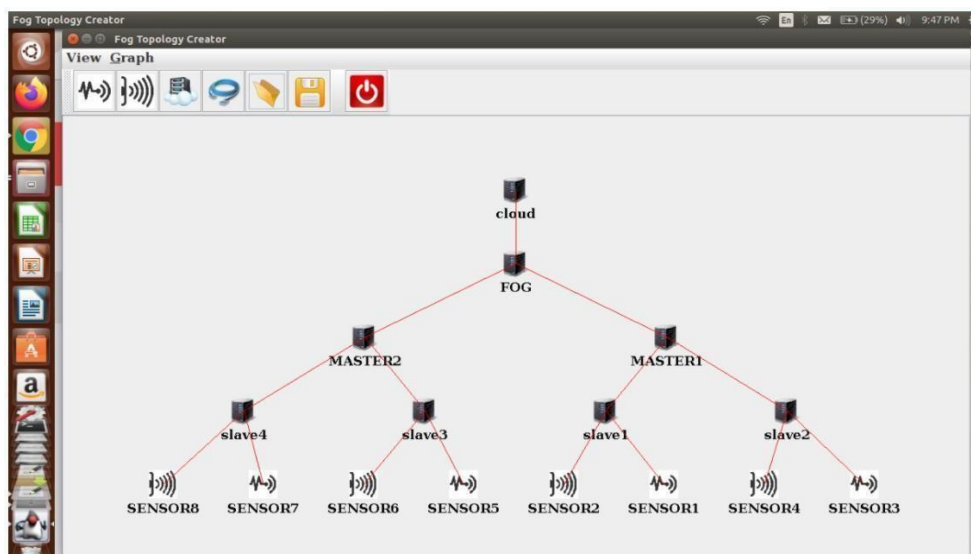


Fig. 6.2.4 Example for Topology simulation

We have considered different topologies to draw the conclusion. The topology seen is one of the examples. Since we have implemented the project on a master and a slave node, and we don't have resources to extend it to more nodes we have simulated the same and drawn conclusion. Topologies Considered to draw conclusion: to measure the accuracy we have used the dataset for other components like power consumption, latency we have used the storage capacity (occupied by dataset) to measure the load.

Here in the above topology the data is being collected by the sensors and is being delivered to the fog nodes [slave, master] for the computation purpose, and to further optimize the implemented system the data can be sent to the cloud architecture so that the output can be sent as a notification to the sufferer.

Prediction accuracy: The chosen dataset for this model comprises 1027 examples. To improve on the testing accuracy and not to cause much delay to the training time, we must divide the available training data equally across all the master nodes as shown in Fig. 6.16 and Fig. 6.17.

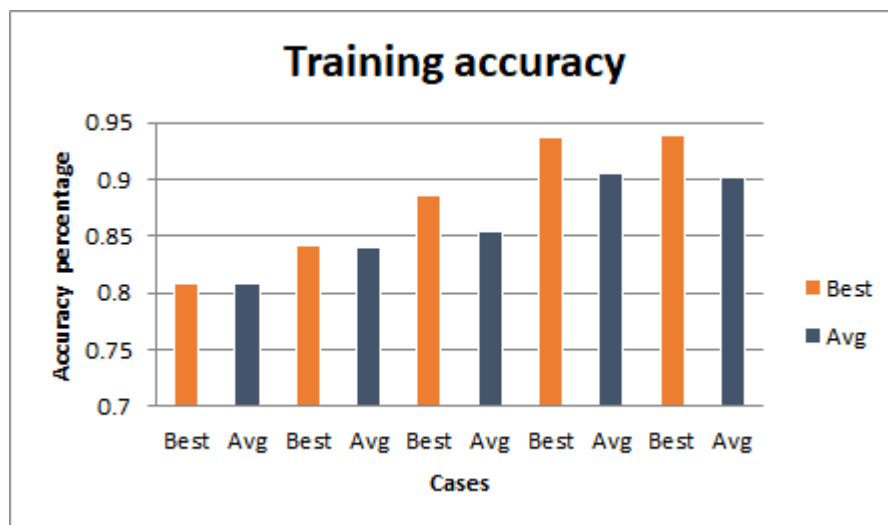


Fig. 6.2.5 Graph showing accuracy when algorithm is applied on different sizes of trainingdataset [2].

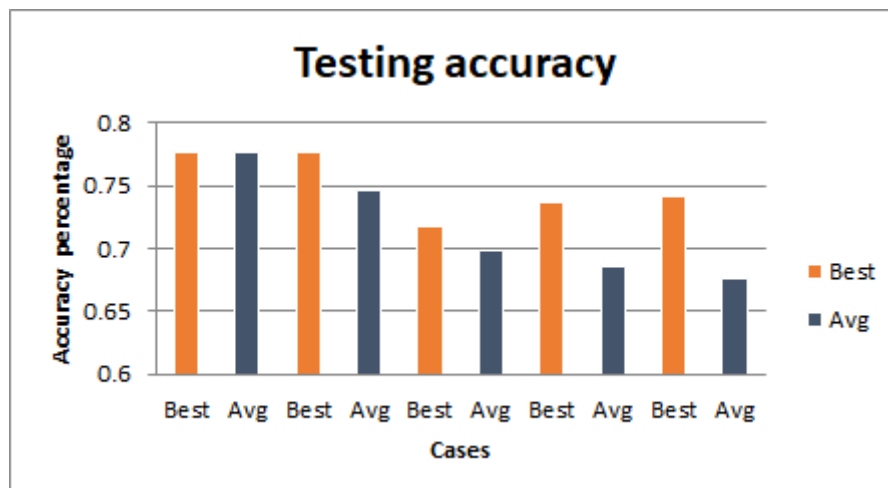


Fig. 6.2.6 Graph showing accuracy when algorithm is applied on different sizes of testingdataset [2].

As the calculation is applied to various sizes of dataset, we can see that the preparation and testing precision increases with the expansion of the dataset. Here we have taken best and normal perceptions of exactness for each dataset size and plotted the diagram to demonstrate that as the size of the dataset builds the calculation's exhibition turns out to be better.

Time characteristics: Low latency compared to cloud, because in fog communication single hop data transfer takes place while in cloud there is multi hop data transfer which leads to high latency as shown in Fig. 6.18.

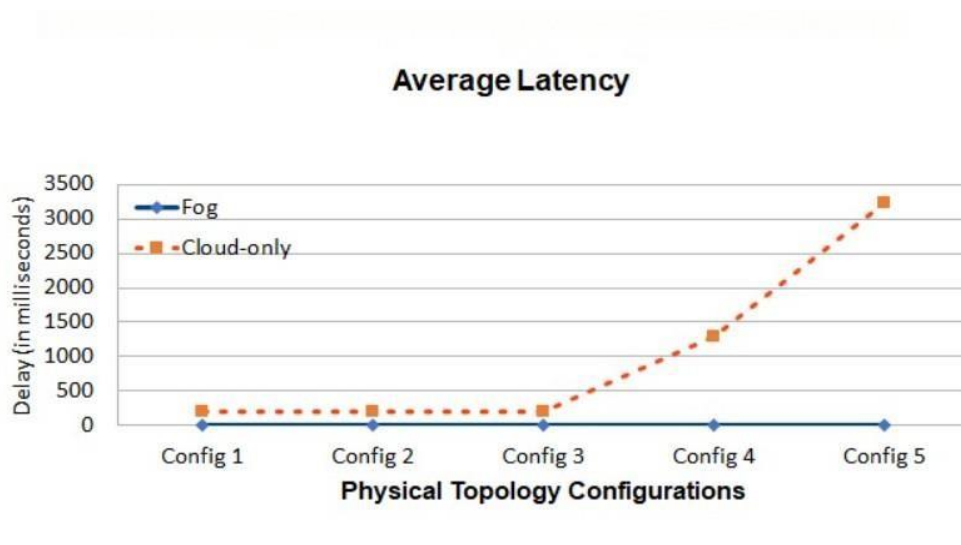


Fig. 6.2.7 Graph showing variation of latency [2].

Network usage: By the study of network usage in different cases and cloud, we can see that the fog uses less bandwidth when compared to the cloud, in fog there is a chance that the network bandwidth can increase when the worker node increases as shown in Fig. 6.2.7.

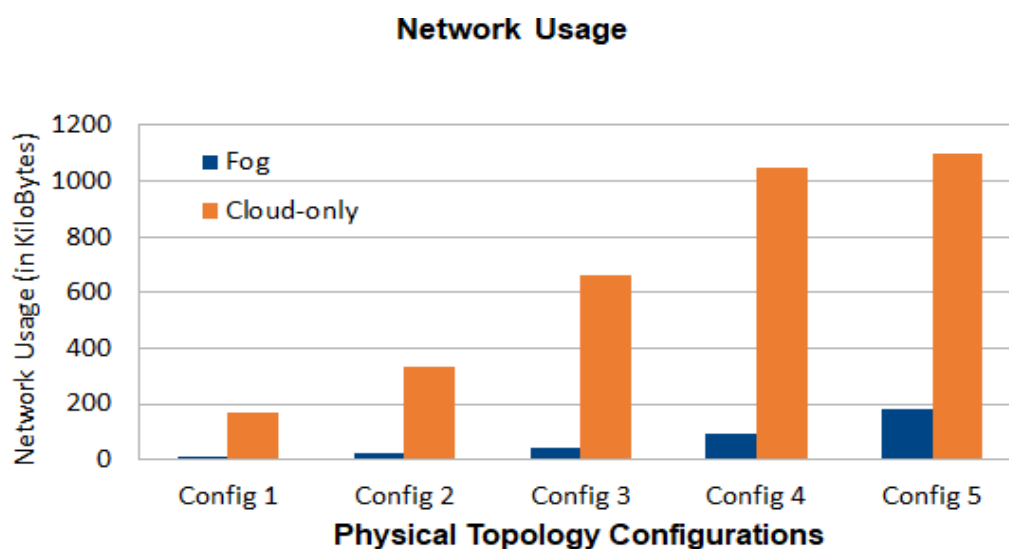


Fig. 6.2.8 Graph showing variation of network usage based on parameters such as bandwidth.[2]

Power consumption: With the simulation observations The power consumption in the fog node is less but in cloud is more because of the high capacity servers, but in fog there is no need for the complex infrastructure and the setup is very simple when compared to cloud as shown in Fig. 6.2.8.

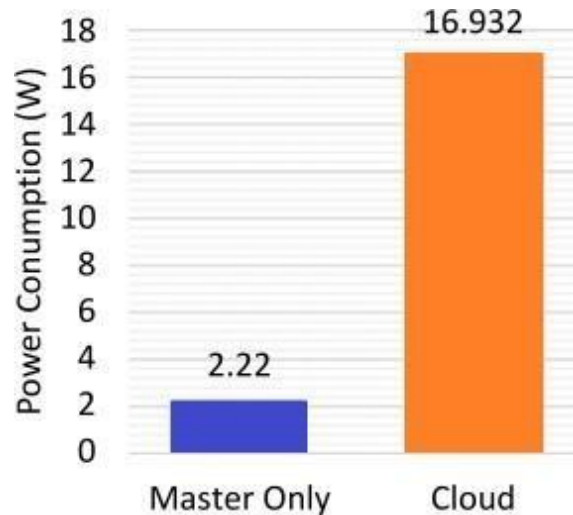


Fig. 6.2.9 Graph showing variation of power consumption. [2]

After analyzing these parameters for result analysis we found that fog is better than cloud and we move ahead with fog architecture for the implementation of smart health care systems.

6.3 Contribution To The Existing System

- In our model which we have implemented follows the slave [replica] concept such that don't have to store the results in the master node but sent to the replica so there is a better resource management and at the slave node we have achieved the resource management as we have used azure cloud integrator in the fog architecture for forwarding the result.

- We have achieved improvements in the properties like time delay or latency, power consumption, energy consumption and network bandwidth with the help of simulation.
- We have achieved an accuracy of 93% which is more than the existing system in terms of prediction algorithm.

7.1 TESTING

Software testing establishes whether or not a piece of software is correct, complete, and of good quality. Validation is the process of verifying that the generated software satisfies the user's expectations. The actions that take place during the testing phase basically, determine whether the system's capabilities fit the criteria. The most important is the goal of software testing is to find bugs in the program. Errors occur when a component fails. It is discovered that parts of the developed system are erroneous, incomplete, or inconsistent. The procedure where software is executed with the purpose of detecting software bugs is one example of a test technique (errors or other defects). It entails the process of evaluating one or more properties of a software component or system interest.

- Meet the criteria that guided the design and development of the product.
- Responds to all types of inputs accurately,
- It completes its tasks in a reasonable amount of time.
- Is it useable in a reasonable amount of time?
- It can be installed and run in the contexts it was designed for.
- Obtains the desired general outcome from the stakeholders

Given the virtually limitless ways possible for the number of tests for even simple components, all software testing employs some approach to pick tests which are practicable given the time and things available. Therefore, software testing is commonly (but not always) performed. Solely tries to run an application with the goal of getting to know anything such as bugs in software. It provides objective, unbiased results, impartial information to users about the quality of software and the danger of failure as well as sponsors. As soon as executable software is available, software testing can begin. In a staged approach, for example, the majority of testing occurs after the system

Requirements have been specified and then implemented. Program that can be tested Requirements, programming, and testing are all handled differently in an agile approach. Testing is frequently carried out in parallel. Both valid and invalid inputs are chosen by the test designer. It calculates the proper output without knowing anything about the internals of the test object Structure.

Unit Testing

Single units of source code sets of one or more program modules, as well as accompanying control data, processes, and procedures are examined to see whether they are suitable for the use of unit testing.

Integration Testing

Individual software segments are joined and assessed as a group during the integration testing phase of software testing. Integration testing is used to assess a system's or component's compliance with stated functional requirements. It happens between unit testing and validation testing.

System Testing

This is a type of testing that is done on an entire integrated system to see if it meets its criteria. All the components that have successfully cleared the integration testing are given into system testing.

Since the project is on a large scale, developers would require testing to make it successful. In the case where each piece works appropriately in every regard and provides expected output for a wide range of data sources then the task is required to be efficient. So if the destination is unsuccessful, this ought to be attempted.

Testing Tools and Environment

For the sake of removing errors that are in various stages The topic of levels of testing comes into the picture. The normal stages of testing are as follows:

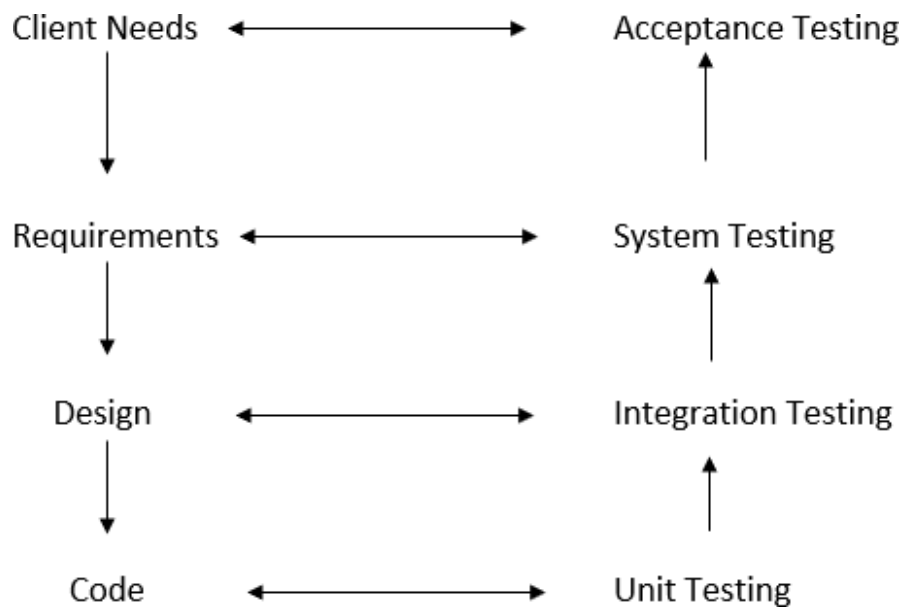


Fig. 7.1 Testing Process

A long sequence of tests was done on the proposed system as shown in Table 8.1 before it was all set to take user acceptance testing.

7.2 TEST CASES

Since the project is on a large scale, developers would require testing to make it successful. In the case where each piece works appropriately in every regard and provides expected output for a wide range of data sources then the task is required to be efficient. So if the destination is unsuccessful, this ought to be attempted.

Use Case	Test Case	Description	Procedure	Expected Result
UC01: Client wants to register	TC01: Client sends the registration request	Client submits the registration POST request with UserID and other components.	Client supplies it's UserID and other attributes to the body of the POST request is constructed and it is updated in the database	The client is successfully registered..
UC02: Client wants to connect a slave node to the master node.	TC02: Client sends the mapping request to Fog node.	Client registers in the master node.	the client downloads the apk file into android app and authenticates using the login credentials.	The android app receives notification when executed.

Table 7.1 Test Cases

CHAPTER 8

CONCLUSION

8.1 Findings and Suggestions

- In this research work, we are focusing on the healthcare aspects for heart patients by proposing a novel Fog based Smart Healthcare System for Automatic Diagnosis of Heart Diseases using machine learning.
- Here we are using decision tree and naive bayes techniques for prediction. Health Fog provides healthcare as a fog service and efficiently manages the data of heart patients which is coming from different IoT devices.
- Here we are reducing latency or delay in communication and improving efficiency.

8.2 Significance of the Proposed Research Work

- Our model which we have implemented follows the slave [replica] concept such that we don't have to store the results in the master node but send them to the replica so there is better resource management.
- Fog has features like low latency, low energy consumption and low network usage. (From simulation result)
- Fog has high efficiency compared to cloud due to its features. (From simulation result)
- Naive bayes algorithm is a fast, highly scalable algorithm,
- Doctors get more clients online and instant diagnosis (instead of waiting for the lab results we can directly check in the implemented system).

8.3 Limitations of this Research Work

- The system is not fully automated, it needs data from the user or Doctor for full diagnosis.
- An internet connection is compulsory when running a web application. Still there are many parts of the world where the internet is not accessible.
- Since in-fog environment controllers and storages are distributed across various locations in the network, it needs more maintenance.
- Browser Support – Unfortunately, we don't all use the same browser. This means during development we have to ensure that all features have to be supported across a variety of browsers.

8.4 Directions for the Future Works

In this research work, the essence being to develop a novel fog-based healthcare system and this is created with a fog environment with one master to many slave architectures for Mechanized prediction of diseases related to the heart using machine learning algorithms with input being taken from several IoT devices. Here we use decision trees and Naive Bayes techniques for prediction. Fog environment is very flexible and it can be used to provide healthcare as a service and coherently manage the input of the sufferer which is imminent from different IoT devices. Here we are reducing latency or delay in communication and improving efficiency. In the future we would like to develop a one to one communication between the master node and the slave node. And for further increase in security we would like to implement the system with block-chain.

REFERENCES

- [1] Tuli, S., Mahmud, R., Tuli, S., & Buyya, R. (2019). FogBus: A blockchain-based lightweight framework for edge and fog computing. *Journal of Systems and Software*, 154, 22-36.
- [2] Tuli, S., Basumatary, N., Gill, S. S., Kahani, M., Arya, R. C., Wander, G. S., & Buyya, R. (2020). HealthFog: An ensemble deep learning based Smart Healthcare System for Automatic Diagnosis of Heart Diseases in integrated IoT and fog computing environments. *Future Generation Computer Systems*, 104, 187-200.
- [3] Margariti, S. V., Dimakopoulos, V. V., & Tsoumanis, G. (2020). Modeling and Simulation Tools for Fog Computing—A Comprehensive Survey from a Cost Perspective. *Future Internet*, 12(5), 89.
- [4] Kumar, Y., & Mahajan, M. (2019). Intelligent behavior of fog computing with IOT for the healthcare system. *Int. J. Sci. Technol. Res*, 8(07).
- [5] Machado, J. D. S., Moreno, E. D., & Ribeiro, A. D. R. L. (2020). A survey on Fog Computing and its research challenges. *International Journal of Grid and Utility Computing*, 11(4), 486-495
- [6] Marques, B., Coelho, I.M., da Costa Sena, A. and Castro, M.C. (2019) ‘_A network coding protocol for wireless sensor fog computing’, *Int. J. Grid and Utility Computing*, Vol. 10, No. 3, pp.224–234.
- [7] Adel, A. (2020). Utilizing technologies of fog computing in educational IoT systems: privacy, security, and agility perspectives. *Journal of Big Data*, 7(1), 1-29
- [8] Gupta, H., Vahid Dastjerdi, A., Ghosh, S. K., & Buyya, R. (2017). iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments. *Software: Practice and Experience*, 47(9), 1275-1296.
- [9] Kraemer, F. A., Braten, A. E., Tam Kittikhun, N., & Palma, D. (2017). Fog computing in

healthcare—a review and discussion. *IEEE Access*, 5, 9206-9222.

[10] Monteiro, A. (2016). Smart Fog Computing Interface for Healthcare Domain Applications.

[11] Bermbach, D., Pallas, F., Pérez, D. G., Plebani, P., Anderson, M., Kat, R., & Tai, S. (2017, November). A research perspective on fog computing. In the International Conference on Service-Oriented Computing (pp. 198-210). Springer, Cham.

[12] Abbasi, B. Z., & Shah, M. A. (2017, September). Fog computing: Security issues, solutions and robust practices. In 2017 23rd International Conference on Automation and Computing (ICAC) (pp. 1-6). IEEE.

[13] Bonomi, F., Milito, R., Zhu, J., & Addepalli, S. (2012, August). Fog computing and its role in the internet of things. In Proceedings of the first edition of the MCC workshop on Mobile cloud computing (pp. 13-16).

[14] Gia, T. N., Jiang, M., Rahmani, A. M., Westerlund, T., Liljeberg, P., & Tenhunen, H. (2015, October). Fog computing in healthcare internet of things: A case study on ecg extraction of features.

[15] Paul, A., Pinjari, H., Hong, W. H., Seo, H. C., & Rho, S. (2018). Fog computing-based IoT for health monitoring systems. *Journal of Sensors*, 2018.

[16] Naas, M. I., Boukhobza, J., Parvedy, P. R., & Lemarchand, L. (2018, May). An extension to ifogsim to enable the design of data placement strategies. In 2018 IEEE 2nd International Conference on Fog and Edge Computing (ICFEC) (pp. 1-8). IEEE.

[17] Rao, T. V. N., Khan, A., Maschendra, M., & Kumar, M. K. (2015). A paradigm shift from cloud to fog computing. *International Journal of Science, Engineering and Computer Technology*, 5(11), 385.

[18] Menon, V. G., & Prathap, P. J. (2017). Moving from vehicular cloud computing to vehicular fog computing: Issues and challenges. *International Journal on Computer Science and Engineering*, 9(2), 14-18.

[19] Plebani, P., Garcia Perez, D., Anderson, M., Bermbach, D., Cappiello, C., Kat, R. I., ... & Vitali, M. (2017). Information logistics and fog computing: The DITAS* approach. In Forum and Doctoral Consortium Papers Presented at the 29th International Conference on Advanced

[20] Abdelaziz, J., Adda, M., & Mcheick, H. (2018). An Architectural Model for Fog Computing. *J. Ubiquitous Syst. Pervasive Networks*, 10(1), 21-25.

[21] AN ENHANCED CLOUD STORAGE METHOD: FOG COMPUTING Sirisha Sanatha, Swathi Amancha B.Tech Student (CSE), SMEC, Hyderabad, Telangana.

[22] Rao, Y. S., & Sree, K. B. (2018). A review on fog computing: conceptual live Vm migration framework, issues, applications and its challenges. *Int J Sci Res Comput Sci Eng InfTechnol*, 3(1), 1175-1184.

[23] Wang, Z., Guo, Y., Gao, Y., Fang, C., Li, M., & Sun, Y. (2020). Fog-Based Distributed Networked Control for Connected Autonomous Vehicles. *Wireless Communications and Mobile Computing*, 2020.

[24] Amor, A. B., Abid, M., & Meddeb, A. (2020). Secure fog based e-learning scheme. *IEEE Access*, 8, 31920-31933.

[25] Hu, P., Ning, H., Qiu, T., Zhang, Y., & Luo, X. (2016). Fog computing based face identification and resolution scheme in the internet of things. *IEEE transactions on industrial informatics*, 13(4), 1910-1920.

[26] Ijaz, M., Li, G., Lin, L., Cheikhrouhou, O., Hamam, H., & Noor, A. (2021). Integration and Applications of Fog Computing and Cloud Computing Based on the Internet of Things for Provision of Healthcare Services at Home. *Electronics*, 10(9), 1077.

INTRODUCTION

LITERATURE SURVEY

SYSTEM REQUIREMENT SPECIFICATION

SYSTEM DESIGN AND ARCHITECTURE

IMPLEMENTATION

OUTPUT SNAPSHOTS

TESTING AND RESULT

CONCLUSION

REFERENCES

BASE PAPER