

# **DISEASE DIAGNOSIS USING MACHINE LEARNING**

**A PROJECT REPORT**

*Submitted By*

**K.VISWANATHAN**

**K.HARI KRISHNAN**

**D.MUKESH**

*In partial fulfillment of the requirements*

*for the award of the degree*

*of*

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**



**ARIGNAR ANNA INSTITUTE OF SCIENCE & TECHNOLOGY**



**ANNA UNIVERSITY: CHENNAI 600 025**

**MAR 2021**

## **BONAFIDE CERTIFICATE**

Certified that this project report titled “**DISEASE DIAGNOSIS USING MACHINE LEARNING**” is the Bonafide work of K.VISWANATHAN (210317104301), K.HARI KRISHNAN (210317104016), D.MUKESH (210317104024) who carried out the research under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

**SIGNATURE**

**NANDAKUMAR S, M.Tech.**

**PROFESSOR,**

**HEAD OF THE DEPARTMENT,**

Computer Science and Engineering,

Arignar Anna Institute of

Science and Technology,

Sriperumbudur,

Chennai- 602 105.

**SIGNATURE**

**RAMACHANDRAN.V M.Tech.**

**SUPERVISOR,**

**ASSISTANT PROFESSOR,**

Computer Science and  
Engineering,

Arignar Anna Institute of

Science and Technology,

Sriperumbudur,

Chennai- 602 105.

Submitted for the M.E Degree Viva-Voce (Phase-II) held at Arignar Anna Institute of Science and Technology on-----

**INTERNAL EXAMINER**

**EXTERNAL EXAMINER**

## **ABSTRACT**

In the EXISTING SYSTEM, however significantly limits the usability of outsourced data due to the difficulty of searching over the encrypted data. Also data security and privacy is a major thread in the cloud computing. In the PROPOSED SYSTEM, we address this issue by developing the fine-grained file level search authorization sense only users, who are granted to access a particular file.

During the registration, every user will generate gets public key & private key. Data owner generates set of trapdoor keys and ABE key which are mailed to the user. 3 -4 Trapdoor keys are generated and everyone is a pair of keys.

When server generates 1 key user has to provide another pair of the key which is made steganography with an image & sent to the server. Server destegano the image and fetches the other pair of the trapdoor key and verifies for authentication. After verification server verifies the access policy for data access through ABE.

## ACKNOWLEDGEMENT

First I would like to take this opportunity to acknowledge my sincere thanks to my respected Chairman **Dr. Virugai G.Jayaraman** providing facilities to carry out the project work.

I am extremely grateful to my esteemed and honourable Principal **Dr.MURALIKRISHNA Ph.D** for providing opportunity to complete the project work successfully.

I express my deep gratitude and thanks to my respected Head of the Department, **Mr.NANDAKUMAR S M.Tech.,** for his much valuable support, unfledged attention and direction, which kept my project on track.

I am extremely indebted to **RAMACHANDRAN.V (M.Tech).** for his valuable suggestions and guidance in my project by spending his precious time.

I am also grateful to my parents and friends who endorsed me with encouragement and their help in the course of this project.

I also thank all the Staff Members of the Computer Science and Engineering who helped me during my project work.

## TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO
	ABSTRACT	3
	LIST OF FIGURES	8
	LIST OF ABBREVIATIONS	8
1	INTRODUCTION	9
	1.1 Domain Introduction	9
	1.2 Project Introduction	24
2	LITERATURE REVIEW	25
	2.1 Computer-assisted decision support for the diagnosis and treatment of infectious diseases in intensive care units	25
	2.2 On the Design and Analysis of the Privacy -Preserving SVM Classifier	25
	2.3 Privacy and Emergency Response In E-Healthcare Leveraging Wireless Body Sensor Network	26
	2.4 Privacy-preserving Protocol for Dynamic Medical Text Mining and Image Feature Extraction from Secure Data Aggregation in Cloud-assisted e-Healthcare Systems	26
	2.5 Smart Community: An Internet of Things Application	27

3	SYSTEM ANALYSIS	28
	3.1 EXISTING SYSTEM	28
	3.2 PROPOSED SYSTEM	28
4	SYSTEM DESIGN	30
	4.1 ARCHITECTURAL DIAGRAM	30
	4.2 DATA FLOW DIAGRAM	30
	4.3 UML DIAGRAM	31
	4.4 USE CASE DIAGRAM	33
	4.5 SEQUENCE DIAGRAM	34
	4.6 COLLABORATION DIAGRAM	35
	4.7 ACTIVITY DIAGRAM	36
	4.8 CLASS DIAGRAM	37
5	SYSTEM SPECIFICATION	40
	5.1 Hardware Environment	40
	5.2 Software Environment	41
	5.3 N-Tier Architecture	48
	5.3.1 Business Benefits	48
6	JAVA	49
	6.1 FEATURES OF JAVA	49
	6.1.1 JAVA PROGRAMMING LANGUAGE	50
	6.2 JAVA PLATFORM	52
	6.2.1 Java Virtual Machine	52
	6.2.2 Java API	53

6.3	JAVA RUNTIME ENVIRONMENT	53
6.4	JAVA SERVER PAGE	54
6.5	SWING APPPLICATION	54
6.6	JAVA PLATFORM,STANDARD EDITION	56
6.7	APPACHE TOM CAT	57
7	MODULES	60
7.1	APPLICATION OF MODULES	61
8	APPENDIX	64
8.1	SOURCE CODE	64
8.2	SCREENSHOTS	94
9	CONCLUSION & FUTURE ENHANCEMENT	101
9.1	CONCLUSON	101
9.2	FUTURE ENHANCEMENT	101
	REFERENCES	101

## **LIST OF FIGURES**

<b>FIGURE NO.</b>	<b>NAME OF THE FIGURE</b>	<b>PAGE NO.</b>
4.1	ARCHITECTURAL DIAGRAM	30
4.2	DATA FLOW DIAGRAM	31
4.3	USE CASE DIAGRAM	33
4.4	SEQUENCE DIAGRAM	34
4.5	COLLABORATION DIAGRAM	35
4.6	ACTIVITY DIAGRAM	36
4.7	CLASS DIAGRAM	37
5.3	N-Tier Architecture	48

## **LIST OF ABBREVIATIONS**

SVM	Support Vector Machine
DAS	Direct Attached Storage
SAN	Storage Area Network
NAS	Network Attached Storage
API	Application Programming Interface
JRE	Java Runtime Environment
JVM	Java Virtual Machine



# **CHAPTER 1**

## **INTRODUCTION**

### **ABSTRACT**

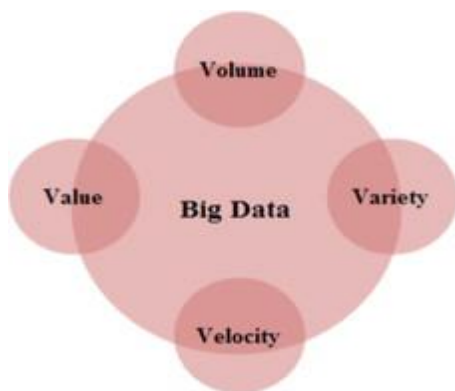
Our goal is to provide a tool to assist professionals and consumers in finding and choosing drugs. To achieve this goal, we develop an approach that allows a user to query for drugs that satisfy a set of conditions based on drug properties, such as drug indications, side effects, and drug interactions, and also takes into account patient profiles. We also analyze the disease and best drug advised to that specific patient through Big Data analysis we are implementing an application to know about disease based on our symptoms. Based on the symptoms system will predict the type of disease and suggest the specialist doctor based on the rank method. After that best drug will suggested by the system.

### **1.1 DOMAIN INTRODUCTION**

Big data is an all-encompassing term for any collection of data sets so large and complex that it becomes difficult to process using traditional data processing applications. The challenges include analysis, capture, duration, search, sharing, storage, transfer, visualization, and privacy violations. The trend to larger data sets is due to the additional information derivable from analysis of a single large set of related data, as compared to separate smaller sets with the same total amount of data, allowing correlations to be found to "spot business trends, prevent diseases, combat crime and so on. So we can implement big data in our project because every employ has instructed information so we can make analysis on this data.

## Characteristics Of Big Data

Big data is a term utilized to refer to the increase in the volume of data that are difficult to store, process, and analyze through traditional database technologies. The nature of big data is indistinct and involves considerable processes to identify and translate the data into new insights. The term “big data” is relatively new in IT and business. However, several researchers and practitioners have utilized the term in previous literature. For instance, referred to big data as a large volume of scientific data for visualization. Several definitions of big data currently exist.” Meanwhile and defined big data as characterized by three Vs: volume, variety, and velocity. The terms volume, variety, and velocity were originally introduced by Gartner to describe the elements of big data challenges. IDC also defined big data technologies as “a new generation of technologies and architectures, designed to economically extract value from very large volumes of a wide variety of data, by enabling the high velocity capture, discovery, and/or analysis.” specified that big data is not only characterized by the three Vs mentioned above but may also extend to four Vs, namely, volume, variety, velocity, and value This 4V definition is widely recognized because it highlights the meaning and necessity of big data.



*Fig Four V's Of Big Data*

**Volume** - Refers to the amount of all types of data generated from different sources and continue to expand. The benefit of gathering large amounts of data includes the creation of hidden information and patterns through data analysis. Collecting longitudinal data requires considerable effort and underlying investments. Nevertheless, such mobile data challenge produced an interesting result similar to that in the examination of the predictability of human behavior patterns or means to share data based on human mobility and visualization techniques for complex data.

**Variety**-Refers to the different types of data collected via sensors, smart phones, or social networks. Such data types include video, image, text, audio, and data logs, in either structured or unstructured format. Most of the data generated from mobile applications are in unstructured format. For example, text messages, online games, blogs, and social media generate different types of unstructured data through mobile devices and sensors. Internet users also generate an extremely diverse set of structured and unstructured data.

**Velocity** -Refers to the speed of data transfer. The contents of data constantly change because of the absorption of complementary data collections, introduction of previously archived data or legacy collections, and streamed data arriving from multiple sources

**Value**- is the most important aspect of big data; it refers to the process of discovering huge hidden values from large datasets with various types and rapid generation

### **Classification Of Big Data**

Big data are classified into different categories to better understand their characteristics shows the numerous categories of big data. The classification is important because of large-scale data in the cloud. The classification is based on five aspects: (i) data sources, (ii) content format, (iii) data stores, (iv) data staging, and (v) data processing.

Data sources include internet data, sensing and all stores of transnational information, ranges from unstructured to highly structured are stored in various formats. Most popular is the relational database that come in a large number of varieties . As the result of the wide variety of data sources, the captured data differ in size with respect to redundancy, consistency and noise, etc.

### **Big Data Storage System**

The rapid growth of data has restricted the capability of existing storage technologies to store and manage data. Over the past few years, traditional storage systems have been utilized to store data through structured RDBMS. However, almost storage systems have limitations and are inapplicable to the storage and management of big data. A storage architecture that can be accessed in a highly efficient manner while achieving availability and reliability is required to store and manage large datasets

Several storage technologies have been developed to meet the demands of massive data. Existing technologies can be classified as direct attached storage (DAS), network attached storage (NAS), and storage area network (SAN). In DAS, various hard disk drives (HDDs) are directly connected to the servers. Each HDD receives a certain amount of input/output (I/O) resource, which is managed by

individual applications. Therefore, DAS is suitable only for servers that are interconnected on a small scale. Given the aforesaid low scalability, storage capacity is increased but expandability and upgradeability are limited significantly. NAS is a storage device that supports a network. NAS is connected directly to a network through a switch or hub via TCP/IP protocols. In NAS, data are transferred as files. Given that the NAS server can indirectly access a storage device through networks, the I/O burden on a NAS server is significantly lighter than that on a DAS server. NAS can orient networks, particularly scalable and bandwidth-intensive networks. Such networks include high-speed networks of optical-fiber connections. The SAN system of data storage is independent with respect to storage on the local area network (LAN). Multipath data switching is conducted among internal nodes to maximize data management and sharing. The organizational systems of data storages (DAS, NAS, and SAN) can be divided into three parts: (i) disc array, where the foundation of a storage system provides the fundamental guarantee, (ii) connection and network subsystems, which connect one or more disc arrays and servers, and (iii) storage management software, which oversees data sharing, storage management, and disaster recovery tasks for multiple servers.

## **Hadoop Background**

Hadoop is an open-source Apache Software Foundation project written in Java that enables the distributed processing of large datasets across clusters of commodity. Hadoop has two primary components, namely, HDFS and Map Reduce programming framework. The most significant feature of Hadoop is that HDFS and Map Reduce are closely related to each other; each are co-deployed

such that a single cluster is produced. Therefore, the storage system is not physically separated from the processing system.

HDFS is a distributed file system designed to run on top of the local file systems of the cluster nodes and store extremely large files suitable for streaming data access. HDFS is highly fault tolerant and can scale up from a single server to thousands of machines, each offering local computation and storage. HDFS consists of two types of nodes, namely, a name node called “master” and several data nodes called “slaves.” HDFS can also include secondary name nodes. The name node manages the hierarchy of file systems and director namespace (i.e., metadata). File systems are presented in a form of name node that registers attributes, such as access time, modification, permission, and disk space quotas. The file content is split into large blocks, and each block of the file is independently replicated across data nodes for redundancy and to periodically send a report of all existing blocks to the name node.

Map Reduce is a simplified programming model for processing large numbers of datasets pioneered by Google for data-intensive applications. The Map Reduce model was developed based on GFS is adopted through open-source Hadoop implementation, which was popularized by Yahoo. Apart from the Map Reduce framework, several other current open-source Apache projects are related to the Hadoop ecosystem, including Hive, H base, Mahout, Pig, Zookeeper, Spark, and Avro. Twister provides support for efficient and iterative Map Reduce computations. An overview of current Map Reduce projects and related software is shown in Map Reduce allows an inexperienced programmer to develop parallel programs and create a program capable of using computers in a cloud. In most cases, programmers are required to specify two functions only: the map function

(mapper) and the reduce function (reducer) commonly utilized in functional programming. The mapper regards the key/value pair as input and generates intermediate key/value pairs. The reducer merges all the pairs associated with the same (intermediate) key and then generates an output. Summarizes the process of the map/reduce function.

## **Map Reduce In Clouds**

Map Reduce accelerates the processing of large amounts of data in a cloud; thus, Map Reduce, is the preferred computation model of cloud providers. Map Reduce is a popular cloud computing framework that robotically performs scalable distributed applications and provides an interface that allows for parallelization and distributed computing in a cluster of servers. The approach is to apply scientific computing problems to the Map Reduce framework where scientists can efficiently utilize existing resources in the cloud to solve computationally large-scale scientific data.

Currently, many alternative solutions are available to deploy Map Reduce in cloud environments; these solutions include using cloud Map Reduce runtimes that maximize cloud infrastructure services, using Map Reduce as a service, or setting up one's own Map Reduce cluster in cloud instances. Several strategies have been proposed to improve the performance of big data processing. Moreover, effort has been exerted to develop SQL interfaces in the Map Reduce framework to assist programmers who prefer to use SQL as a high-level language to express their task while leaving all of the execution optimization details to the backend

## **Research Challenges**

Although cloud computing has been broadly accepted by many organizations, research on big data in the cloud remains in its early stages. Several existing issues have not been fully addressed. Moreover, new challenges continue to emerge from applications by organization. In the subsequent sections, some of the key research challenges, such as scalability, availability, data integrity, data transformation, data quality, data heterogeneity, privacy and legal issues, and regulatory governance, are discussed.

### **Scalability**

Scalability is the ability of the storage to handle increasing amounts of data in an appropriate manner. Scalable distributed data storage systems have been a critical part of cloud computing infrastructures. The lack of cloud computing features to support RDBMSs associated with enterprise solutions has made RDBMSs less attractive for the deployment of large-scale applications in the cloud. This drawback has resulted in the popularity of NoSQL.

A No SQL database provides the mechanism to store and retrieve large volumes of distributed data. The features of No SQL databases include schema-free, easy replication support, simple API, and consistent and flexible modes. Different types of No SQL databases, such as key-value column-oriented, and document-oriented, provide support for big data. Shows a comparison of various NoSQL database technologies that provide support for large datasets.



## **Availability**

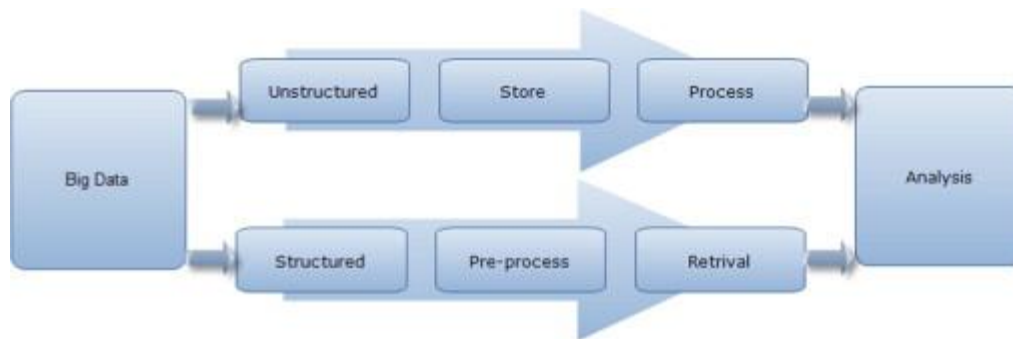
Availability refers to the resources of the system accessible on demand by an authorized individual. In a cloud environment, one of the main issues concerning cloud service providers is the availability of the data stored in the cloud. For example, one of the pressing demands on cloud service providers is to effectively serve the needs of the mobile user who requires single or multiple data within a short amount of time. Therefore, services must remain operational even in the case of a security breach. In addition, with the increasing number of cloud users, cloud service providers must address the issue of making the requested data available to users to deliver high-quality services. Lee et al. introduced a multi-cloud model called “rain clouds” to support big data exploitation. “Rain clouds” involves cooperation among single clouds to provide accessible resources in an emergency. Schroeck et al. predicted that the demand for more real time access to data may continue to increase as business models evolve and organizations invest in technologies required for streaming data and smart phones.

## **Data Integrity**

A key aspect of big data security is integrity. Integrity means that data can be modified only by authorized parties or the data owner to prevent misuse. The proliferation of cloud-based applications provides users the opportunity to store and manage their data in cloud data centers. Such applications must ensure data integrity. However, one of the main challenges that must be addressed is to ensure the correctness of user data in the cloud.

## Transformation

Transforming data into a form suitable for analysis is an obstacle in the adoption of big data. Owing to the variety of data formats, big data can be transformed into an analysis workflow in two ways



*Fig Transformation process*

## Data Quality

In the past, data processing was typically performed on clean datasets from well-known and limited sources. Therefore, the results were accurate. However, with the emergence of big data, data originate from many different sources; not all of these sources are well-known or verifiable. Poor data quality has become a serious problem for many cloud service providers because data are often collected from different sources. For example, huge amounts of data are generated from smart phones, where inconsistent data formats can be produced as a result of heterogeneous sources. The data quality problem is usually defined as “any difficulty encountered along one or more quality dimensions that render data completely or largely unfit for use”. Therefore, obtaining high-quality data from vast collections of data sources is a challenge. High-quality data in the cloud is

characterized by data consistency. If data from new sources are consistent with data from other sources, then the new data are of high quality.

## **Heterogeneity**

Variety, one of the major aspects of big data characterization, is the result of the growth of virtually unlimited different sources of data. This growth leads to the heterogeneous nature of big data. Data from multiple sources are generally of different types and representation forms and significantly interconnected; they have incompatible formats and are inconsistently represented.

In a cloud environment, users can store data in structured, semi-structured, or unstructured format. Structured data formats are appropriate for today's database systems, whereas semi-structured data formats are appropriate only to some extent. Unstructured data are inappropriate because they have a complex format that is difficult to represent in rows and columns. According to Kocarev and Jakimoski, the challenge is how to handle multiple data sources and types.

## **Privacy**

Privacy concerns continue to hamper users who outsource their private data into the cloud storage. This concern has become serious with the development of big data mining and analytics, which require personal information to produce relevant results, such as personalized and location-based services. Information on individuals is exposed to scrutiny, a condition that gives rise to concerns on profiling, stealing, and loss of control.

## **Legal/Regulatory Issues**

Specific laws and regulations must be established to preserve the personal and sensitive information of users. Different countries have different laws and regulations to achieve data privacy and protection. In several countries, monitoring of company staff communications is not allowed. However, electronic monitoring is permitted under special circumstances. Therefore, the question is whether such laws and regulations offer adequate protection for individuals' data while enjoying the many benefits of big data in the society at large.

## **Governance**

Data governance embodies the exercise of control and authority over data-related rules of law, transparency, and accountabilities of individuals and information systems to achieve business objectives. The key issues of big data in cloud governance pertain to applications that consume massive amounts of data streamed from external sources. Therefore, a clear and acceptable data policy with regard to the type of data that need to be stored, how quickly an individual needs to access the data, and how to access the data must be defined.

Big data governance involves leveraging information by aligning the objectives of multiple functions, such as telecommunication carriers having access to vast troves of customer information in the form of call detail records and marketing seeking to monetize this information by selling it to third parties.

Moreover, big data provides significant opportunities to service providers by making information more valuable. However, policies, principles, and frameworks that strike stability between risk and value in the face of increasing data size and deliver better and faster data management technology can create huge challenges.

Cloud governance recommends the use of various policies together with different models of constraints that limit access to underlying resources. Therefore, adopting governance practices that maintain a balance between risk exposure and value creation is a new organizational imperative to unlock competitive advantages and maximize value from the application of big data in the cloud.

## **Open Research Issues**

Numerous studies have addressed a number of significant problems and issues pertaining to the storage and processing of big data in clouds. The amount of data continues to increase at an exponential rate, but the improvement in the processing mechanisms is relatively slow. Only a few tools are available to address the issues of big data processing in cloud environments. State-of-the-art techniques and technologies in many important big data applications (i.e., MapReduce, Dryad, Pregel, PigLatin, MangoDB, Hbase, SimpleDB, and Cassandra) cannot solve the actual problems of storing and querying big data. For example, Hadoop and Map Reduce lack query processing strategies and have low-level infrastructures with respect to data processing and management. Despite the plethora of work performed to address the problem of storing and processing big data in cloud computing environments, certain important aspects of storing and processing big data in cloud computing are yet to be solved. Some of these issues are discussed in the subsequent subsections.

## **Data Staging**

The most important open research issue regarding data staging is related to the heterogeneous nature of data. Data gathered from different sources do not have a structured format. For instance, mobile cloud-based applications, blogs, and

social networking are inadequately structured similar to pieces of text messages, videos, and images. Transforming and cleaning such unstructured data before loading those into the warehouse for analysis are challenging tasks. Efforts have been exerted to simplify the transformation process by adopting technologies such as Hadoop and MapReduce to support the distributed processing of unstructured data formats. However, understanding the context of unstructured data is necessary, particularly when meaningful information is required. MapReduce programming model is the most common model that operates in clusters of computers; it has been utilized to process and distribute large amounts of data.

### **Distributed Storage Systems**

Numerous solutions have been proposed to store and retrieve massive amounts of data. Some of these solutions have been applied in a cloud computing environment. However, several issues hinder the successful implementation of such solutions, including the capability of current cloud technologies to provide necessary capacity and high performance to address massive amounts of data, optimization of existing file systems for the volumes demanded by data mining applications, and how data can be stored in such a manner that they can be easily retrieved and migrated between servers.

### **Data Analysis**

The selection of an appropriate model for large-scale data analysis is critical. Talia pointed out that obtaining useful information from large amounts of data requires scalable analysis algorithms to produce timely results. However, current algorithms are inefficient in terms of big data analysis. Therefore, efficient data analysis tools and technologies are required to process such data. Each algorithm

performance ceases to increase linearly with increasing computational resources. As researchers continue to probe the issues of big data in cloud computing, new problems in big data processing arise from the transitional data analysis techniques. The speed of stream data arriving from different data sources must be processed and compared with historical information within a certain period of time. Such data sources may contain different formats, which makes the integration of multiple sources for analysis a complex task .

## **Data Security**

Although cloud computing has transformed modern ICT technology, several unresolved security threats exist in cloud computing. These security threats are magnified by the volume, velocity, and variety of big data. Moreover, several threats and issues, such as privacy, confidentiality, integrity, and availability of data, exist in big data using cloud computing platforms. Therefore, data security must be measured once data are outsourced to cloud service providers. The cloud must also be assessed at regular intervals to protect it against threats. Cloud vendors must ensure that all service level agreements are met. Recently, some controversies have revealed how some security agencies use data generated by individuals for their own benefit without permission.

Therefore, policies that cover all user privacy concerns should be developed. Traditionally, the most common technique for privacy and data control is to protect the systems utilized to manage data rather than the data itself; however, such systems have proven to be vulnerable. Utilizing strong cryptography to encapsulate sensitive data in a cloud computing environment and developing a novel algorithm that efficiently allows for key management and secure key exchange are important to manage access to big data, particularly as they exist in the cloud independent of

any platform. Moreover, the issue with integrity is that previously developed hashing schemes are no longer applicable to large amounts of data. Integrity verification is also difficult because of the lack of support, given remote data access and the lack of information on internal storage.

## **OBJECTIVE OF THE PROJECT:**

- The objective of the project is to identify the disease based on the patient symptoms and suggest the best drug by English and ayurvedic medicine.
- To suggest doctor based on ranking.

## **1.2 PROJECT INTRODUCTION:**

With the rapid development of wireless sensors, smart devices and network technologies, the Internet of Things (IoT), as it can greatly improve the quality of life, has played an important role in the modern society [1]. As one of the major applications in IoT, e-healthcare has been widely researched since it has advantages in prevention and easy monitoring of diseases, ad hoc diagnosis and providing prompt medical attention in cases of accidents [2], [3]. healthcare includes many research fields, among which the extensive one is disease risk prediction as it can help to predict the disease risk and improve the diagnosis efficiency. Thus, in this paper, we focus on this popular research field.

In general, the disease risk prediction mainly consists of two phases: disease model training and remote disease prediction [4], [5]. In the phase of disease model training, a huge number of historical medical data containing patients' symptoms



and confirmed diseases are collected by the resource-abundant third party, e.g., the cloud platform, and then the training result is extracted from the collected data by means of big data mining technologies [6], [7]. After that healthcare providers, e.g., hospital or medical company, utilize the training result to predict the disease risk for undiagnosed patients based on the personal symptoms collected by medical monitoring devices or doctor visits. That is, in the whole process of disease risk prediction, confirmed patients provide their historical medical data for disease model training, while undiagnosed patients can use the disease prediction service to obtain the possible diseases by providing the collected symptoms.

Unfortunately, as shown in most e-healthcare researches [8]–[11], security and privacy issues have significantly impeded the wide adoption of e-healthcare systems, since the exposure and abuse of personal health information (PHI) would bring about serious privacy leakage, let alone the involvement of not fully-trusted third-party cloud platforms [12], [13].

## **CHAPTER 2**

### **LITERATURE SURVEY**

#### **2.1 Computer-assisted decision support for the diagnosis and treatment of infectious diseases in intensive care units**

**Author:** C A M Schurink, P J F Lucas, I M Hoepelman, M J M Bonten

##### **Abstract:**

Diagnosing nosocomial infections in critically ill patients admitted to intensive care units (ICUs) is a challenge because signs and symptoms are usually non-specific for a particular infection. In addition, the choice of treatment, or the decision not to treat, can be difficult. Models and computer-based decision-support systems have been developed to assist ICU physicians in the management of

infectious diseases. We discuss the historical development, possibilities, and limitations of various computer-based decision-support models for infectious diseases, with special emphasis on Bayesian approaches. Although Bayesian decision-support systems are potentially useful for medical decision making in infectious disease management, clinical experience with them is limited and prospective evaluation is needed to determine whether their use can improve the quality of patient care.

## **2.2 On the Design and Analysis of the Privacy-Preserving SVM Classifier**

**Author:** Keng-Pei Lin and Ming-Syan Chen

### **Abstract:**

The support vector machine (SVM) is a widely used tool in classification problems. The SVM trains a classifier by solving an optimization problem to decide which instances of the training data set are support vectors, which are the necessarily informative instances to form the SVM classifier. Since support vectors are intact tuples taken from the training data set, releasing the SVM classifier for public use or shipping the SVM classifier to clients will disclose the private content of support vectors. This violates the privacy-preserving requirements for some legal or commercial reasons. The problem is that the classifier learned by the SVM inherently violates the privacy. This privacy violation problem will restrict the applicability of the SVM. To the best of our knowledge, there has not been work extending the notion of privacy preservation to tackle this inherent privacy violation problem of the SVM classifier. In this paper, we exploit this privacy violation problem, and propose an approach to postprocess the SVM classifier to transform it to a privacy-preserving classifier which does not disclose the private content of support vectors. The postprocessed SVM classifier without exposing the private content of training data is called Privacy-Preserving SVM Classifier (abbreviated as PPSVC). The PPSVC is designed for the commonly used Gaussian kernel function. It precisely approximates the decision function of the Gaussian kernel SVM classifier without exposing the sensitive attribute values possessed by support vectors. By applying the PPSVC, the SVM classifier is able to be publicly released while preserving privacy. We prove that the PPSVC is robust against adversarial attacks. The experiments on real data sets show that the classification accuracy of the PPSVC is comparable to the original SVM classifier.

## **2.3 PRIVACY AND EMERGENCY RESPONSE IN E-HEALTHCARE LEVERAGING WIRELESS BODY SENSOR NETWORKS**

**Author:** jinyuan sun,jinyuan sun,xiaoyanzhu

### **Abstract:**

Electronic healthcare is becoming a vital part of our living environment and exhibits advantages over paper-based legacy systems. Privacy is the foremost concern of patients and the biggest impediment to e-healthcare deployment. In addressing privacy issues, conflicts from the functional requirements must be taken into account. One such requirement is efficient and effective response to medical emergencies. In this article, we provide detailed discussions on the privacy and security issues in e-healthcare systems and viable techniques for these issues. Furthermore, we demonstrate the design challenge in the fulfillment of conflicting goals through an exemplary scenario, where the wireless body sensor network is leveraged, and a sound solution is proposed to overcome the conflict.

## **2.4 Privacy-preserving Protocol for Dynamic Medical Text Mining and Image Feature Extraction from Secure Data Aggregation in Cloud-assisted Healthcare Systems**

**Author:** Jun Zhou, Zhenfu Cao Xiaolei Dong, Xiaodong Lin

### **Abstract:**

E-healthcare systems have been increasingly facilitating health condition monitoring, disease modeling and early intervention, and evidence-based medical treatment by medical text mining and image feature extraction. Owing to the resource constraint of wearable mobile devices, it is required to outsource the frequently collected personal health information (PHI) into the cloud. Unfortunately, delegating both storage and computation to the untrusted entity would bring a series of security and privacy issues. The existing work mainly focused on fine-grained privacy-preserving static medical text access and analysis, which can hardly afford the dynamic health condition fluctuation and medical image analysis. In this paper, a secure and efficient privacy-preserving dynamic medical text mining and image feature extraction scheme PPDM in cloud-assisted e-healthcare systems is proposed. Firstly, an efficient privacy-preserving fully homomorphic data aggregation is proposed, which serves the basis for our

proposed PPDM. Then, an outsourced disease modeling and early intervention is achieved, respectively by devising an efficient privacy-preserving function correlation matching PPDM1 from dynamic medical text mining and designing a privacy-preserving medical image feature extraction PPDM2. Finally, the formal security proof and extensive performance evaluation demonstrate our proposed PPDM achieves a higher security level (i.e. information-theoretic security for input privacy and adaptive chosen ciphertext attack (CCA2) security for output privacy) in the honest but curious model with optimized efficiency advantage over the state-of-the-art in terms of both computational and communication overhead.

## **2.5 Smart Community: An Internet of Things Application**

**Author:** Xu Li, Rongxing Lu, Xiaohui Liang, and Xuemin (Sherman) Shen, Jiming Chen, Xiaodong Lin

### **Abstract:**

In this article, we introduce an Internet of Things application, smart community, which refers to a paradigmatic class of cyber-physical systems with cooperating objects (i.e., networked smart homes). We then define the smart community architecture, and describe how to realize secure and robust networking among individual homes. We present two smart community applications, Neighborhood Watch and Pervasive Healthcare, with supporting techniques and associated challenges, and envision a few value added smart community services.

## **CHAPTER 3**

### **SYSTEM ANALYSIS**

#### **3.1 EXISTING SYSTEM:**

In the Existing system, more number of people were getting serious and even died because of not aware of disease. Even more number of applications are available to know about disease but they need login and patients have to give symptoms manually. Regularities in the prescriptions are for both clinical practice and the novel prescription development.

## **DISADVANTGES:**

- People are getting their medicine in emergency situation on medical shop without the knowledge of doctor
- More number of time wasted because of waiting on the hospital
- Patient are waiting for long time
- People were bought drugs from pharmacy without knowledge

## **3.2 PROPOSED SYSTEM:**

We implement a web application to know about their disease themselves using Big data. People were easily gives their symptoms by selecting symptoms form drop down box. we develop an approach that allows a user to query for drugs that satisfy a set of conditions based on drug properties, such as drug indications, side effects, and drug interactions, and also takes into account patient profiles, We also analyze the disease and best drug advised to that specific patient through Big Data analysis we are implementing an application to know about disease based on our symptoms. Based on the symptoms system will predict the type of disease and suggest the specialist doctor based on the rank method. After that best drug will suggested by the system. Based on disease we suggest the medicine in allopathic and ayurvedhic.

## **ADVANTAGES:**

- Predict the disease based on the symptoms
- Best drugs are suggested
- Best doctors were suggested based on ranking system

- Both ayurvedic and English medicine are recommended
- People can easily communicate with doctors
- Both people can gain through this application
- Time will consume

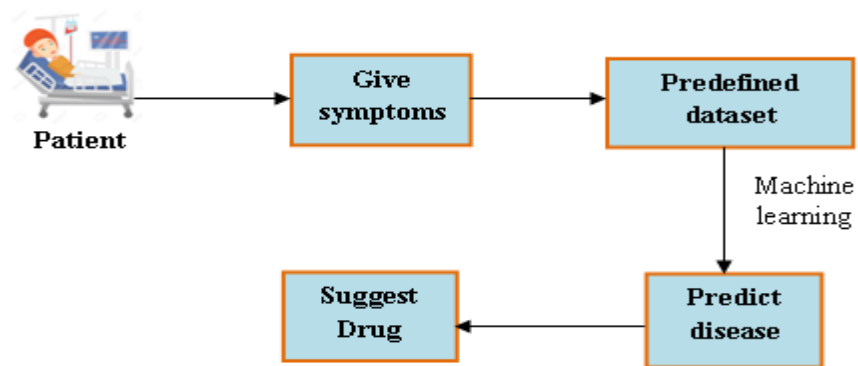
## **CHAPTER 4**

### **SYSTEM DESIGN**

#### **4.1 ARCHITECTURE DIAGRAM:**

**Fig.**

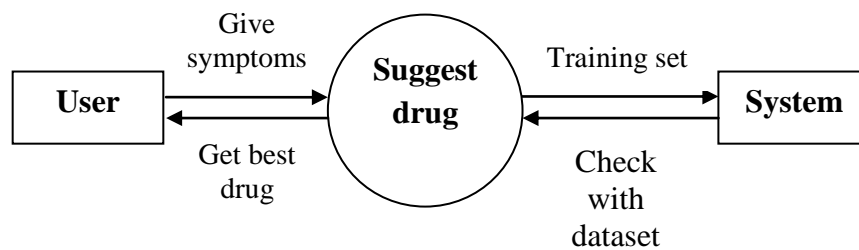
**No:4.1  
SYSTEM**



## ARCHITECTURE

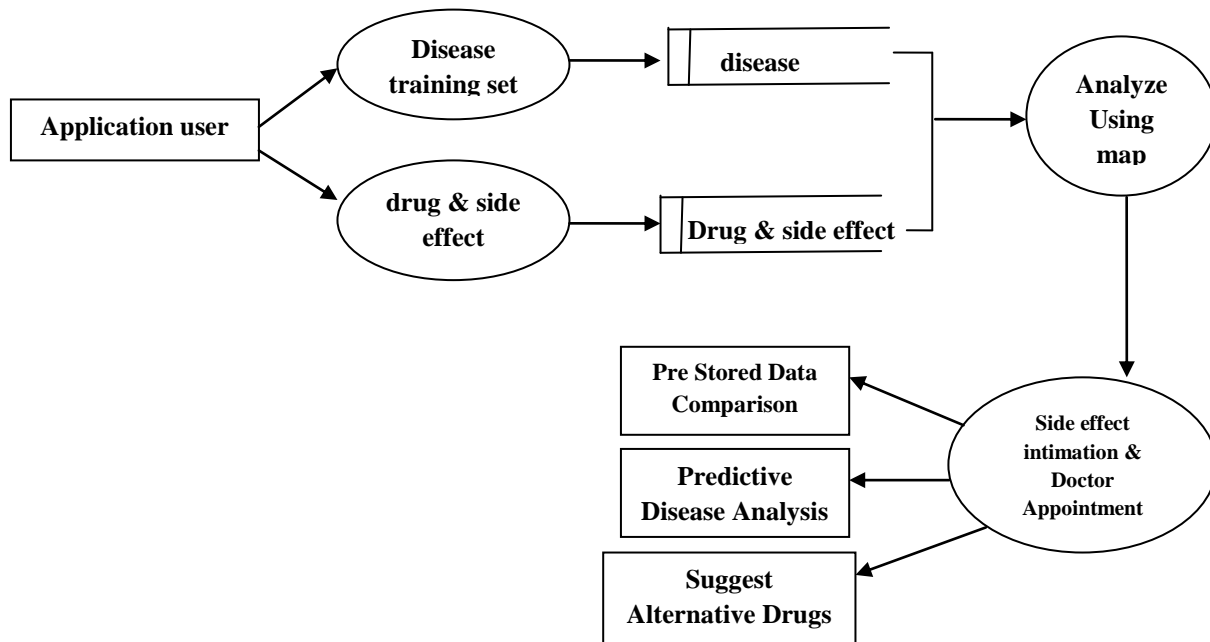
### 4.2 DATA FLOW DIAGRAM

#### 4.2.1: LEVEL 0



**Fig. No: 4.2 Data flow diagram**

### 4.2.2 LEVEL 1



### 4.3 UML DIAGRAMS:

UML is simply another graphical representation of a common semantic model. UML provides a comprehensive notation for the full lifecycle of object-oriented development.

### ADVANTAGES



- To represent complete systems (instead of only the software portion) using object oriented concepts
- To establish an explicit coupling between concepts and executable code
- To take into account the scaling factors that are inherent to complex and critical systems
- To creating a modeling language usable by both humans and machines

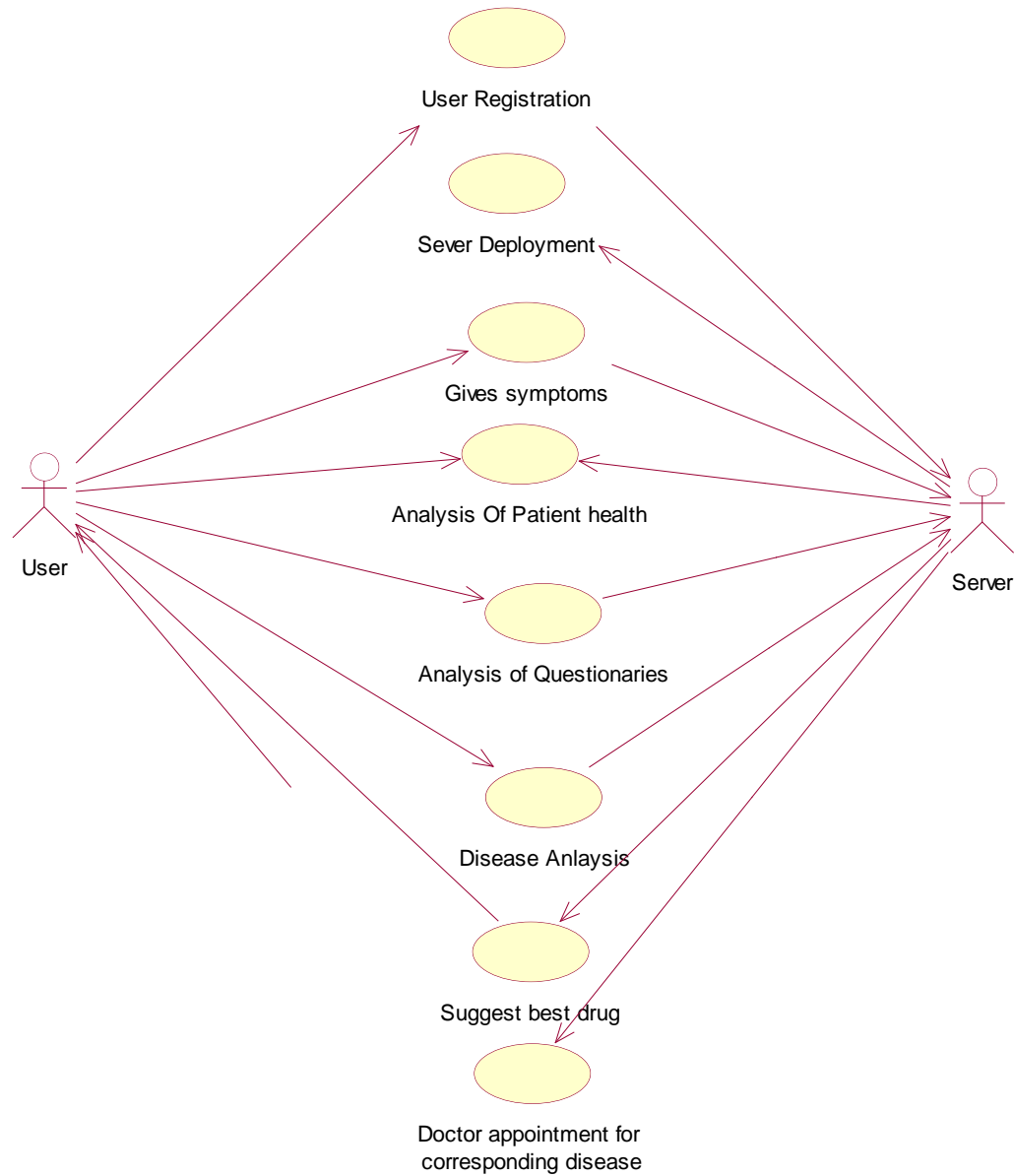
UML defines several models for representing systems

- The class model captures the static structure
- The state model expresses the dynamic behavior of objects
- The use case model describes the requirements of the user
- The interaction model represents the scenarios and messages flows
- The implementation model shows the work units
- The deployment model provides details that pertain to process allocation

## **4.4 USECASE DIAGRAM**

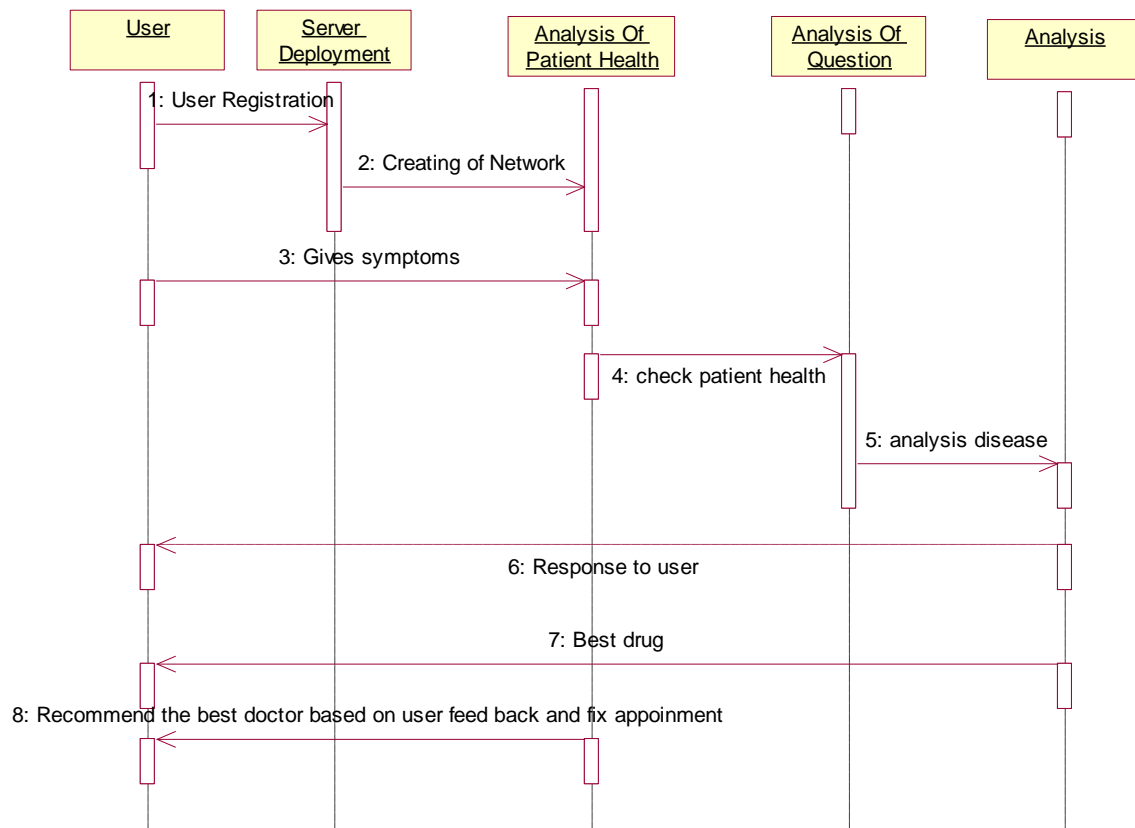
Use case diagrams overview the usage requirement for system. They are useful for presentations to management and/or project stakeholders, but for actual development you will find that use cases provide significantly more value because they describe “the meant” of the actual requirements. A use case describes a

sequence of action that provides something of measurable value to an action and is drawn as a horizontal ellipse.



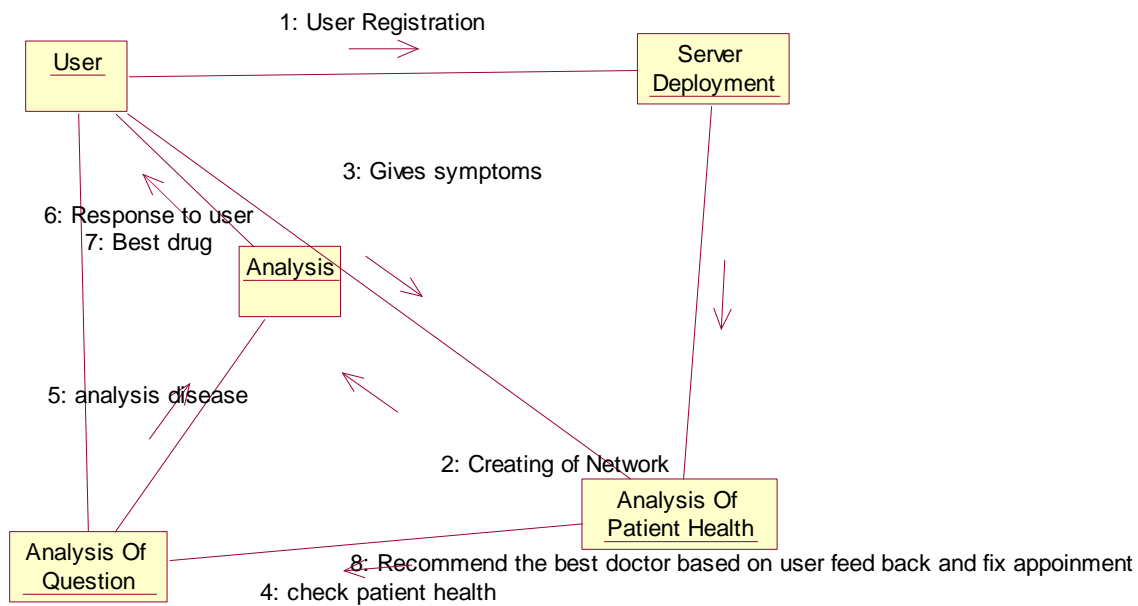
## 4.5 SEQUENCE DIAGRAM

Sequence diagram model the flow of logic within your system in a visual manner, enabling you both to document and validate your logic, and commonly used for both analysis and design purpose. Sequence diagram are the most popular UML artifact for dynamic modeling, which focuses on identifying the behavior within your system.



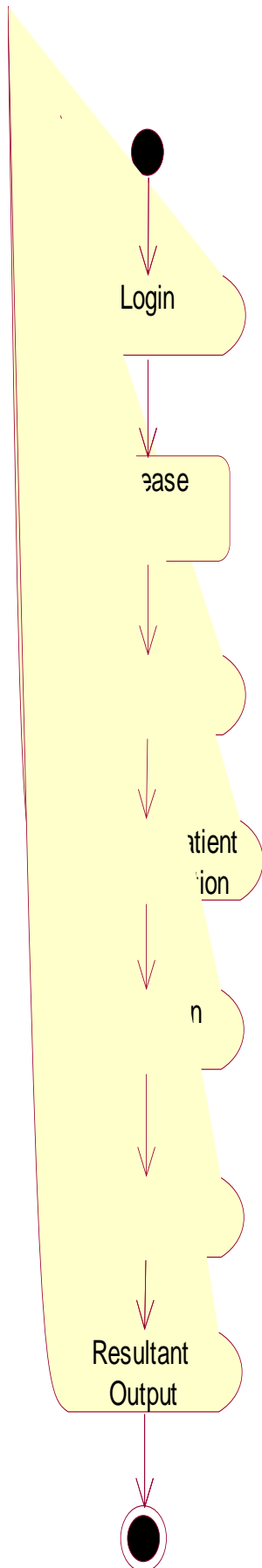
## 4.6 COLLABORATION DIAGRAM

Another type of interaction diagram is the collaboration diagram. A collaboration diagram represents a collaboration, which is a set of objects related in a particular context, and interaction, which is a set of messages exchange among the objects within the collaboration to achieve a desired outcome.

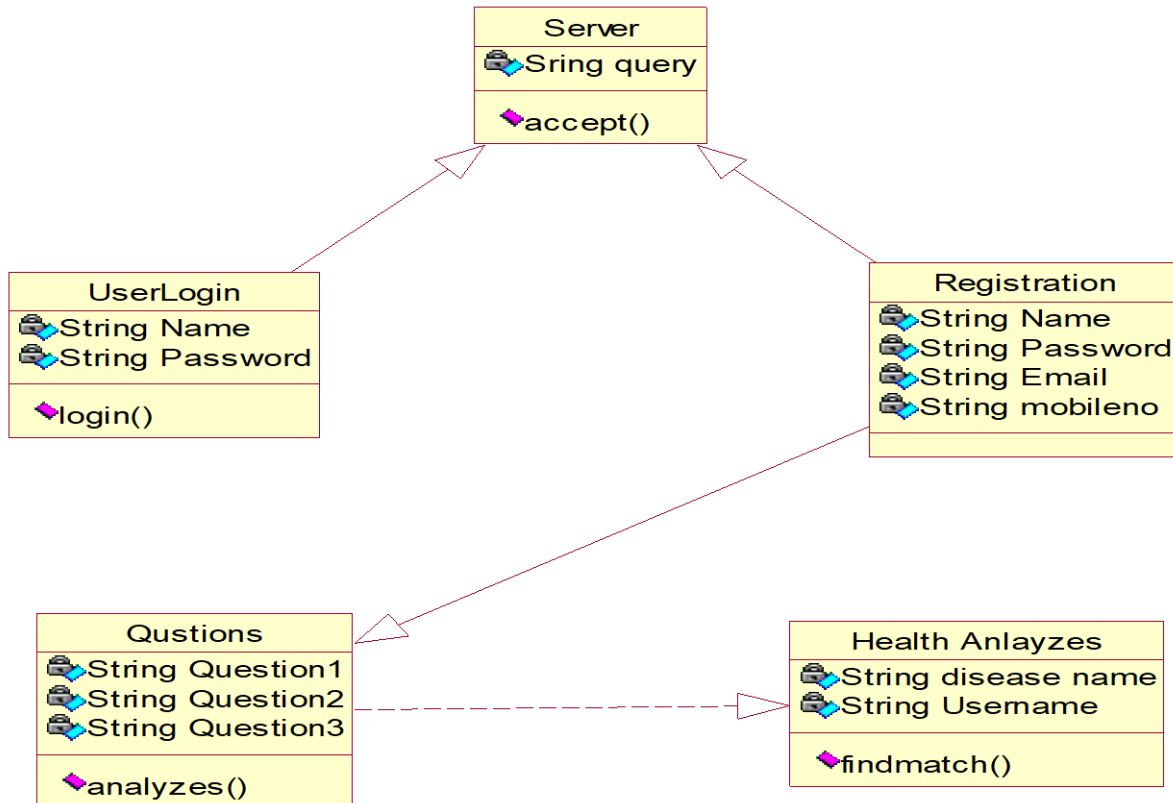


#### 4.7 ACTIVITY DIAGRAM:

Activity diagram are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. The activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. Activity diagram consist of Initial node, vity final node and activities in between.



## 4.8 CLASS DIAGRAM:



## 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.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- OPERATIONAL FEASIBILITY

### **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.

### **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.

## **OPERATIONAL 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 the system.

## **PROBLEM DEFINITION AND METHODOLOGY**

This chapter deals with the problem definition of the project and methodology used in this project. Problem definition describes the detailed information about the project. Methodology that has been adapted in this project is discussed in methodology.



## **CHAPTER 5**

### **SYSTEM SPECIFICATIONS**

#### **LANGUAGE SPECIFICATION:**

This chapter describes the requirement analysis in accordance with the input and the resources and it also describes the implementation of the project with the technology used.

#### **REQUIREMENT ANALYSIS**

Requirement analysis determines the requirements of a new system. This project analyses on product and resource requirement, which is required for this successful system. The product requirement includes input and output requirements it gives the wants in term of input to produce the required output. The resource requirements give in brief about the software and hardware that are needed to achieve the required functionality.

#### **5.1Hardware Environment**

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design. It shows what the systems do and not how it should be implemented.

- Hard disk : 500 GB
- RAM : 4GB
- Processor : Core i5

## **5.2 Software Environment**

The software requirements are the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the team's and tracking the team's progress throughout the development activity.

- Operating system : Windows 7/ 8
- Languages : Java JDK 8.1
- Data Base : Mysql
- IDE : Net Beans 8.2

## **OVERVIEW OF THE MYSQL DATABASE MANAGEMENT SYSTEM**

### **WHAT IS MYSQL?**

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation. The MySQL Web site (<http://www.mysql.com/>) provides the latest information about MySQL software.

- MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by MySQL AB. MySQL AB is a commercial company, founded in 1995 by the MySQL developers. It is a second generation Open Source company that unites Open Source values and methodology with a successful business model.
- The MySQL® software delivers a very fast, multi-threaded, multi-user, and robust SQL (Structured Query Language) database server. MySQL Server is intended for mission-critical, heavy-load production systems as well as for embedding into mass-deployed software. MySQL is a registered trademark of MySQL AB.
- The MySQL software is Dual Licensed. Users can choose to use the MySQL software as an Open Source product under the terms of the GNU General Public License or can purchase a standard commercial license from MySQL AB.

The company name was given after co-founder Monty Widenius's daughter, My, the SQL is “Structured Query Language.”, AB is Swedish for limited partnership.

## • **MYSQL IS A DATABASE MANAGEMENT SYSTEM.**

A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

## • **MySQL databases are relational. What is MySQL?**

A relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables,

views, rows, and columns, offers a flexible programming environment. You set up rules governing the relationships between different data fields, such as one-to-one, one-to-many, unique, required or optional, and “pointers” between different tables. The database enforces these rules, so that with a well-designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data. The SQL part of “MySQL” stands for “Structured Query Language”. SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax. SQL is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, “SQL-92” refers to the standard released in 1992, “SQL:1999” refers to the standard released in 1999, and “SQL:2003” refers to the current version of the standard. We use the phrase “the SQL standard” to mean the current version of the SQL Standard at any time

## MySQL Overview

### What does it do?

- MySQL is a database management system.

A database is a structured collection of data. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server.

- **MySQL is a relational database management system.**

A relational database stores data in separate tables rather than putting all the data in one big storeroom. This adds speed and flexibility. The SQL part of “MySQL” stands for “Structured Query Language.”

- **MySQL software is Open Source.**

Open Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. If you want to redistribute altered versions MySQL, other licences are available

- **MySQL Server works in client/server or embedded systems.**

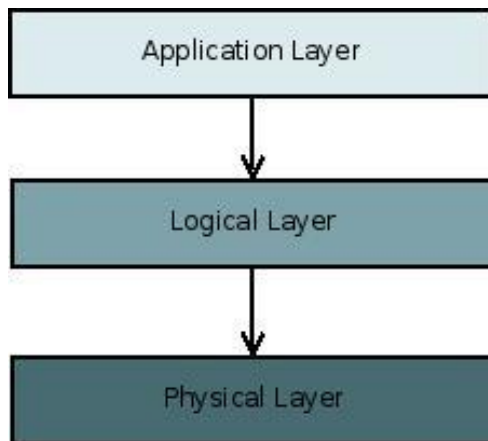
The MySQL Database Software is a client/server system that consists of a multi-threaded SQL server that supports different backends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs).

- **A large amount of contributed MySQL software is available.**

MySQL is a widely supported database, and very likely that most of your applications supports it.

## MySQL Architecture

Three layer model:

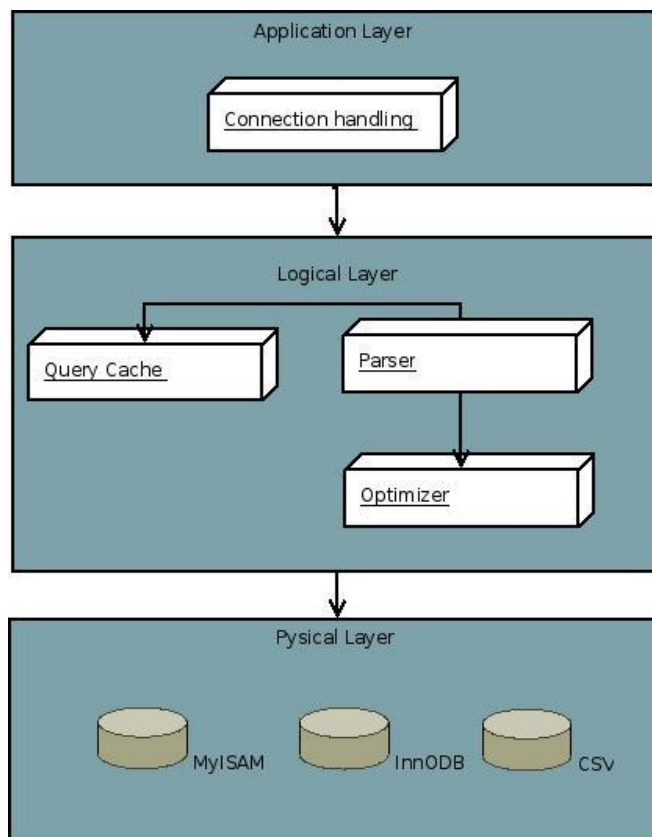


- The application layer contains common network services for connection handling, authentication and security. This layer is where different clients interact with MySQL these clients can be written in different API's: .NET, Java, C, C++, PHP, Python, Ruby, Tcl, Eiffel, etc...
- The Logical Layer is where the MySQL intelligence resides, it includes functionality for query parsing, analysis, caching and all built-in functions

(math, date...). This layer also provides functionality common across the storage engines.

- The Physical Layer is responsible for storing and retrieving all data stored in “MySQL”. Associated with this layer are the storage engines, which MySQL interacts with very basic standard API's. Each storage engine has its strengths and weaknesses, some of these engines are MyISAM, InnoDB, CSV, NDB Cluster, Falcon, etc...

### SOME INTERNAL COMPONENTS:



Each client connection gets its own thread within the server process.

When clients (applications) connect to the MySQL server, the server needs to authenticate them.

Before even parsing the query, though, the server consults the query cache, which only stores SELECT statements, along with their result sets.

The storage engine does affect how the server optimizes query.

### Storage engines:

MySQL supports several storage engines that act as handlers for different table types. MySQL storage engines include both those that handle transaction-safe tables and those that handle non-transaction-safe tables.

Feature	MyISAM	Memory	InnoDB	Archive	NDB
Storage limits	256TB	RAM	64TB	None	384EB
Transactions	No	No	Yes	No	Yes
Locking granularity	Table	Table	Row	Row	Row

### MySQL Files:

In Linux the configuration file is typically located at `/etc/mysql/my.cnf`, but may vary for the different Linux flavours, this also applies to the database files themselves which are located in the `/var/lib/MySQL`.

Regardless of the storage engine, every MySQL table you create is represented, on disk, by a `.frm` file, which describes the table's format (i.e. the table definition). The file bears the same name as the table, with a `.frm` extension. The `.frm` format is the same on all platforms but in the description of the `.frm` format that follows, the examples come from tables created under the Linux operating system.

`/var/lib/mysql/db.frm`      #Table definition

`/var/lib/mysql/db.MYD`    #MyISAM data file

`/var/lib/mysql/db.MYI`    #MyISAM Index file

/var/lib/mysql/ibdata1      #Innodb data file

## **MySQL Cluster Overview**

MySQL Cluster is a technology that enables clustering of in-memory databases in a shared-nothing system. The shared-nothing architecture enables the system to work with very inexpensive hardware, and with a minimum of specific requirements for hardware or software.

MySQL Cluster is designed not to have any single point of failure. In a shared-nothing system, each component is expected to have its own memory and disk, and the use of shared storage mechanisms such as network shares, network file systems, and SANs is not recommended or supported.

MySQL Cluster integrates the standard MySQL server with an in-memory clustered storage engine called NDB (which stands for “Network DataBase”). In our documentation, the term NDB refers to the part of the setup that is specific to the storage engine, whereas “MySQL Cluster” refers to the combination of one or more MySQL servers with the NDB storage engine. A MySQL Cluster consists of a set of computers, known as hosts, each running one or more processes. These processes, known as nodes, may include MySQL servers (for access to NDB data), data nodes (for storage of the data), one or more management servers, and possibly other specialized data access programs. The relationship of these components in a MySQL Cluster is shown here:



## 5.3 N-Tier Architecture



Figure 5.1.6 N-Tier Architecture

### 5.3.1 Business Benefits

- A richer user experience - Whether you're using a Java technology-enabled mobile phone to play a game or to access your company's network, the Java platform provides the foundation for true mobility. The unique blend of mobility and security in Java technology makes it the ideal development and deployment vehicle for mobile and wireless solutions.

- The ideal execution environment for Web services - The Java and XML languages are the two most extensible and widely accepted computing languages on the planet, providing maximum reach to everyone, everywhere, every time, to every device and platform.
- Enabling business from end to end - Java offers a single, unifying programming model that can connect all elements of a business infrastructure.

## **CHAPTER 6**

### **JAVA**

The Java platform is the ideal platform for network computing. Running across all platforms -- from servers to cell phones to smart cards -- Java technology unifies business infrastructure to create a seamless, secure, networked platform for your business. The Java platform benefits from a massive community of developers and supporters that actively work on delivering Java technology-based products and services as well as evolving the platform through an open, community-based, standards organization known as the Java Community Process program. You can find Java technology in cell phones, on laptop computers, on the Web, and even trackside at Formula One Grand Prix races. The fact is today, you can find Java technology just about everywhere!

#### **6.1 FEATURES OF JAVA**

**Java** is a programming language originally developed by James Gosling at Sun Microsystems (which is now a subsidiary of Oracle Corporation) and released

in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++ but has a simpler object model and fewer low-level facilities. Java applications are typically compiled to byte code(class file) that can run on any Java Virtual Machine (JVM) regardless of computer architecture. Java is general-purpose, concurrent, class-based, and object-oriented, and is specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere". Java is considered by many as one of the most influential programming languages of the 20th century, and widely used from application software to web application.

The original and reference implementation Java compilers, virtual machines, and class libraries were developed by Sun from 1995. As of May 2007, in compliance with the specifications of the Java Community Process, Sun relicensed most of their Java technologies under the GNU General Public License. Others have also developed alternative implementations of these Sun technologies, such as the GNU Compiler for Java and GNU Class path.

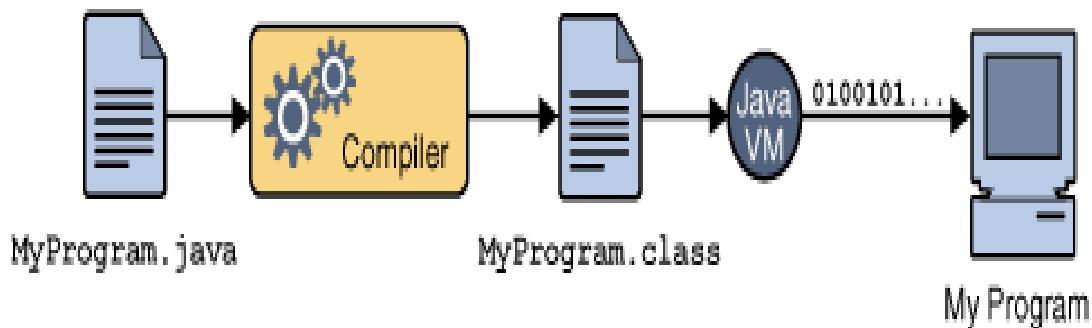
### **6.1.1 JAVA PROGRAMMING LANGUAGE**

The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

- Simple
- Architecture neutral
- Object oriented
- Portal

- Distributed
- High performance

Each of the preceding buzzwords is explained in *The Java Language Environment*, a white paper written by James Gosling and Henry Chilton. In the Java programming language, all source code is first written in plain text files ending with the `.java` extension. Those source files are then compiled into `.class` files by the `javac` compiler. A `.class` file does not contain code that is native to your processor; it instead contains *byte codes* — the machine language of the Java Virtual Machine (Java VM). The `java` launcher tool then runs your application with an instance of the Java Virtual Machine.



**Fig 7.1.1 Software Development Process.**

Because the Java VM is available on many different operating systems, the same `.class` files are capable of running on Microsoft Windows, the Solaris Operating System (Solaris OS), Linux, or Mac OS. Some virtual machines, such as the Java Hotspot virtual machine, perform additional steps at runtime to give your application a performance boost. This includes various tasks such as finding performance bottlenecks and recompiling (to native code) frequently used sections of code.

## 6.2 JAVA PLATFORM

A platform is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Microsoft Windows, Linux, Solaris OS, and Mac OS. Most platforms can be described as a combination of the operating system and underlying hardware. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.

The Java platform has two components:

- The *Java Virtual Machine*
- The *Java Application Programming Interface* (API)

### 6.2.1 JAVA VIRTUAL MACHINE

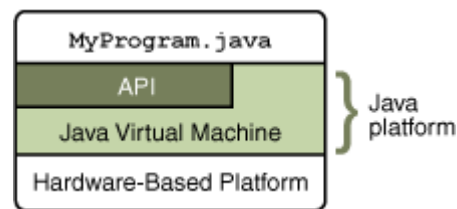
A **Java virtual machine (JVM)** is a virtual machine that can execute Java byte code. It is the code execution component of the Java software platform. A Java virtual machine is a program which executes certain other programs, namely those containing Java byte code instructions. JVMs are most often implemented to run on an existing operating system, but can also be implemented to run directly on hardware. A JVM provides an environment in which Java byte code can be executed, enabling such features as automated exception handling, which provides

root-cause debugging information for every software error (exception), independent of the source code.

A JVM is distributed along with a set of standard class libraries that implement the Java application programming interface (API). These libraries, bundled together with the JVM, form the Java Runtime Environment (JRE). JVMs are available for many hardware and software platforms. The use of the same byte code for all JVMs on all platforms allows Java to be described as a write once, run anywhere programming language, versus write once, compile anywhere, which describes cross-platform compiled languages. Thus, the JVM is a crucial component of the Java platform.

### 6.2.2 JAVA API

The API is a large collection of ready-made software components that provide many useful capabilities. It is grouped into libraries of related classes and interfaces; these libraries are known as packages.



**Fig 6.2.2 API and JVM**

As a platform-independent environment, the Java platform can be a bit slower than native code. However, advances in compiler and virtual machine technologies are bringing performance close to that of native code without threatening portability.

## 6.3 JAVA RUNTIME ENVIRONMENT

The Java Runtime Environment, or JRE, is the software required to run any application deployed on the Java Platform. End-users commonly use a JRE in software packages and Web browser plug-in. Sun also distributes a superset of the JRE called the Java 2 SDK (more commonly known as the JDK), which includes development tools such as the Java compiler, Javadoc, Jar and debugger.

One of the unique advantages of the concept of a runtime engine is that errors (exceptions) should not 'crash' the system. Moreover, in runtime engine environments such as Java there exist tools that attach to the runtime engine and every time that an exception of interest occurs they record debugging information that existed in memory at the time the exception was thrown (stack and heap values). These Automated Exception Handling tools provide 'root-cause' information for exceptions in Java programs that run in production, testing or development environments.

## 6.4 JAVA SERVER PAGE

Java Server Pages (JSPs) are server-side Java EE components that generate responses, typically HTML pages, to HTTP requests from clients. JSPs embed Java code in an HTML page by using the special delimiters `<%` and `%>`. A JSP is compiled to a Java *servlet*, a Java application in its own right, the first time it is accessed. After that, the generated servlet creates the response.

## 6.5 SWING APPLICATION

Swing is a graphical user interface library for the Java SE platform. It is possible to specify a different look and feel through the pluggable look and feel

system of Swing. Clones of Windows, GTK+ and Motif are supplied by Sun. Apple also provides an Aqua look and feel for Mac OS X. Where prior implementations of these looks and feels may have been considered lacking, Swing in Java SE 6 addresses this problem by using more native GUI widget drawing routines of the underlying platforms.

This example Swing application creates a single window with "Hello, world!" inside:

```
// Hello.java (Java SE 5)

import javax.swing.*;

public class Hello extends JFrame {

    public Hello () {

        setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);

        add(new JLabel("Hello, world!"));

        pack();

    } public static void main(String[] args) {

        new Hello().setVisible(true);

    }
}
```

The first import includes all of the public classes and interfaces from the `javax.swing` package. The `Hello` class extends the `Frame` class; the `Frame` class implements a window with a title bar and a close control. The `Hello()` constructor initializes the frame by first calling the super class constructor, passing the



parameter "hello", which is used as the window's title. It then calls the set Default Close Operation (into)method inherited from Frame to set the default operation when the close control on the title bar is selected to Window Constants. EXIT ON CLOSE this causes the JFrame to be disposed of when the frame is closed (as opposed to merely hidden), which allows the JVM to exit and the program to terminate.

Next, the layout of the frame is set to a Border Layout; this tells Swing how to arrange the components that will be added to the frame. A Label is created for the string "Hello, world!" and the add (Component) method inherited from the Container super class is called to add the label to the frame. The pack () method inherited from the Window super class is called to size the window and lay out its contents. The main () method is called by the JVM when the program starts. It instantiates a new Hello frame and causes it to be displayed by calling the set Visible (Boolean) method inherited from the Component super class with the Boolean parameter true. Once the frame is displayed, exiting the main method does not cause the program to terminate because the AWT event dispatching thread remains active until all of the Swing top-level windows have been disposed.

## **6.6 JAVA PLATFORM, STANDARD EDITION**

**Java Platform, Standard Edition** or **Java SE** is a widely used platform for programming in the Java language. It is the Java Platform used to deploy portable applications for general use. In practical terms, Java SE consists of a virtual machine, which must be used to run Java programs, together with a set of libraries

(or "packages") needed to Allow the use of file systems, networks, graphical interfaces, and so on, from within those programs. Some of the general purpose packages in J2SE are as follows

- Java.lang
- Java.io
- Java.math
- Java.net
- Java.util

## **6.7 APPACHE TOM CAT**

*Tomcat*, developed by Apache ([www.apache.org](http://www.apache.org)), is a standard reference implementation for Java servlets and JSP. It can be used standalone as a Web server or be plugged into a Web Server like Apache, Netscape Enterprise Server, or Microsoft Internet Information Server. There are many versions of Tomcat. This tutorial uses Tomcat 5.5.9 as an example. The tutorial should also apply to all later versions of Tomcat. To start Tomcat, you have to first set the JAVA\_HOME environment variable to the JDK home directory using the following command. The JDK home directory is where your JDK is stored. On my computer, it is c:\Program Files\jdk1.6.0\_24.

The apache tomcat server is an open source, java-based web application container that was created to servlets and java server page (JSP) web applications.

It was created under the apache-Jakarta subproject; however, due to its popularity, it is now hosted as separate apache project, where it is supported and enhanced by a group of volunteers from the open source java community.

## **The Tomcat Manager Web Application**

The Tomcat manager web application is packaged with the tomcat server. It is installed in the context path of manager and provides the basic functionality to manage web applications running in the tomcat server from any web browser. Some of the provided functionality includes the ability to install, start, stop, remove, and report on web applications. Covers the details of the tomat manager web application.

### **The Server**

The first container element referenced in this snippet is the `<Server>` element. It represents the entire Catalina servlet engine and is used as a top-level element for a single Tomcat instance. The `<Server>` element may contain one or more `<Service>` containers.

### **The Service**

The next container element is the `<Service>` element, which holds a collection of one or more `<Connector>` elements that share a single `<Engine>` element. N-number of `<Service>` elements may be nested inside a single `<Server>` element.

## **The Connector**

The next type of element is the `<Connector>` element, which defines the class that does the actual Handling requests and responses to and from a calling client application.

## **The Engine**

The third container element is the `<Engine>` element. Each defined `<Service>` can have only one `<Engine>` element and this single `<Engine>` component handles all requests received by all of the defined `<Connector>` components defined by a parent service.

## **The Host**

The `<Host>` element defines the virtual hosts that are contained in each instance of a Catalina `<Engine>`. Each `<Host>` can be a parent to one or more web applications, with each being represented by a `<Context>` component.

## **The Context**

The `<Context>` element is the most commonly used container in a Tomcat instance. Each `<Context>` element represents an individual web application that is running within a defined `<Host>`. There is no limit to the number of contexts that can be defined within a `<Host>`.

## **Java Web Applications**

The main function of the Tomcat server is to act as a container for Java web applications. Therefore, before we can begin our Tomcat-specific discussion, a brief introduction as to exactly what web applications are is in order. The concept of a web application was introduced with the release of the Java servlets

specification 2.2. According to this specification, “a web application is a collection of servlets, html pages, classes, and other resources that can be bundled and run on multiple containers from multiple vendors.” What this really means is that a web application is a container that can hold any combination of the following list of objects:

- Servlets
- Java Server Pages (JSPs)
- Utility classes
- Static documents, including HTML, images, JavaScript libraries, cascading style sheets, and so on
- Client-side classes
- Meta-information describing the web application

## **CHAPTER 7**

### **MODULES**

#### **MODULES:**

- A modular design reduces complexity, facilitates change (a critical aspect of software maintainability), and results in easier implementation by encouraging parallel development of different part of system. Software with effective modularity is easier to develop because function may be compartmentalized and interfaces are simplified. Software architecture

embodies modularity that is software is divided into separately named and addressable components called modules that are integrated to satisfy problem requirements.

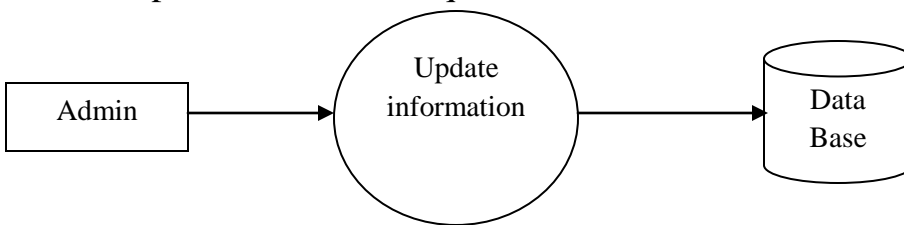
- Modularity is the single attribute of software that allows a program to be intellectually manageable. The five important criteria that enable us to evaluate a design method with respect to its ability to define an effective modular design are: Modular decomposability, Modular Comps ability, Modular Understandability, Modular continuity, Modular Protection.
- The following are the modules of the project, which is planned in aid to complete the project with respect to the proposed system, while overcoming existing system and also providing the support for the future enhancement.

## **7.1 APPLICATION MODULES:**

- **SERVER DEPLOYMENT**
- **CONSTRUCTION OF DISEASE TRAINING SET**
- **BIG DATA BASED ANALYSIS**
- **PRE STORED DATA COMPARISON**
- **PREDICTIVE DISEASE ANALYSIS**
- **DRUG AND TRAINING SET CONSTRUCTION**

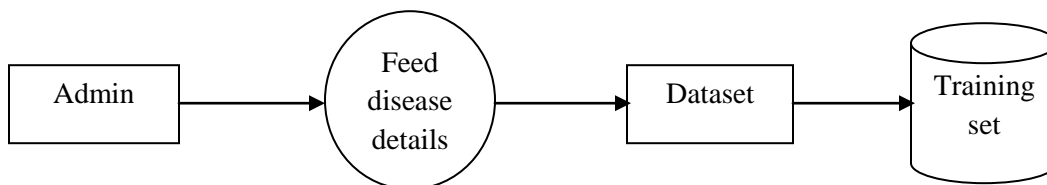
## SERVER DEPLOYMENT

Data Service Provider will contain the large amount of data in their Data Storage. Also the. The drugs and disease information will be stored in the Database . Also the Data Server will redirect the User requested job to the Resource Assigning Module to process the User requested Job.



## CONSTRUCTION OF DISEASE TRAINING SET

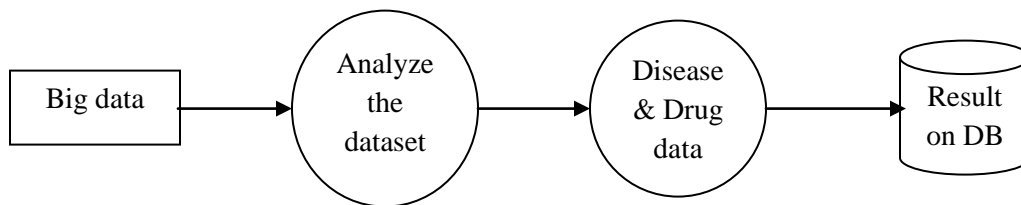
In this module we can design and implementation of train the disease to system. Server will store a set of trained dataset and its relevant diagnosis pattern. Multiple number diseases and symptoms will be collected and stored in a training set. Different type diseases and symptoms will be stored in a dataset.



## BIG DATA BASED ANALYSIS

In this module we implement big data, in this big data we will have lot or vast amount of data that may wanted or unwanted information in simple the

information in the big data are unstructured. So in this module the patient is going allow permission to access the server by the big data analyst .The big data analyst get the all the disease and drugs information which mention above and extract the information by the technique of map reducing formation to get useful information which is useful for patient.



## **PRE STORED DATA COMPARISON**

In this module doctor will import all the details about the medicine i.e. what are the symptoms, dosages and drug .And hw will store more about of the medicine so that we can make some use of it for example we can give awareness to the society .we store the all the data in the clustering format so that data can spitted and stored in the different clusters. So that it will easily to classify the data for the research.

## **PREDICTIVE DISEASE ANALYSIS**

In this module we implement predictive disease analysis system in which the data will be analysis so that we can predictive the disease based on the symptoms .This module interact with server to analysis, the analysiation is done by the



researchers. So they get the data from the server to make analysis to find the disease based on the symptoms

## **BEST DOCTOR RECOMMENDATION**

In this module we suggest the best doctor based on ranking. These ranking system is based on patient review. Patients will give review for the treatment and based on review doctors will come upon top most place.

## **DRUG AND TRAINING SET CONSTRUCTION**

In this module we will train the drugs for every disease and also train the side effects of the drugs. User will be giving their Symptoms & Diagnostic reports to the system for the diagnosis of the disease. If any side effects came by using the prescribed drug user can give the side effects on the same website and those side effects will be match the drugs and suggest the alternate drug for that disease. In this we will suggest the medicine Based upon patients willing like whether they need ayurvedic or English medicine.

## **8. APPENDIX**

### **8.1 SOURCE CODING:**

#### **USER REGISTRATION:**

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

```
*/
```

```
package com.nura.servlet;
```

```
import com.nura.dao.impl.UserDetailsDAOImpl;
```

```
import com.nura.entity.UserDetails;
```

```
import java.io.IOException;
```

```
import java.io.InputStream;
```

```
import java.io.PrintWriter;
```

```
import java.text.ParseException;
```

```
import java.text.SimpleDateFormat;
```

```
import java.util.Date;
```

```
import java.util.logging.Level;
```

```
import java.util.logging.Logger;
```

```
import javax.servlet.ServletException;
```

```
import javax.servlet.annotation.MultipartConfig;
```

```
import javax.servlet.annotation.WebServlet;
```

```
import javax.servlet.http.HttpServlet;
```

```
import javax.servlet.http.HttpServletRequest;
```

```
import javax.servlet.http.HttpServletResponse;
```

```
import javax.servlet.http.Part;
```

```
@MultipartConfig
```

```

@WebServlet(name = "UserRegistration", urlPatterns = {"/UserRegistration"})

public class UserRegistration extends HttpServlet {

    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)

        throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    try (PrintWriter out = response.getWriter()) {
        /* TODO output your page here. You may use following sample code. */
        UserDetails _userDtls = new UserDetails();
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy-mm-dd");
        Date date = null;
        try {
            date = sdf.parse(request.getParameter("dob"));
        } catch (ParseException ex) {

Logger.getLogger(UserRegistration.class.getName()).log(Level.SEVERE,    null,
ex);

        }

        _userDtls.setDateOfBirth(date);
        _userDtls.setGender(request.getParameter("gender"));
        // _userDtls.setLivesIn(request.getParameter("lives_in"));
        _userDtls.setMobileNumber(request.getParameter("mobile_no"));

```

```

_userDtls.setPassword(request.getParameter("password"));
// _userDtls.setStudiedAt(request.getParameter("studied_at"));
_userDtls.setUserName(request.getParameter("username"));
_userDtls.setSpecialist(request.getParameter("disease_name"));
_userDtls.setFirstName(request.getParameter("usr_name"));
// _userDtls.setWorkedAt(request.getParameter("worked_at"));
//String[] userLikes = request.getParameterValues("user_likes");

try {
    for (Part parts : request.getParts()) {
        System.out.println("Parts");

        InputStream inputStream = parts.getInputStream();
        byte[] getImgytes = new byte[inputStream.available()];
        inputStream.read(getImgytes);

        String fileName = getFileName(parts);
        System.out.println("File name:-" + fileName);
        if (fileName != null) {
            _userDtls.setUserImage(getImgytes);
        }
    }
} catch (Exception ex) {
    System.out.println(ex.getMessage());
}

```

```

        if (new UserDetailsDAOImpl().saveUserDtls(_userDtls)) {
            response.sendRedirect("loginPage.jsp?msg=User Details saved");
        } else {
            response.sendRedirect("loginPage.jsp?msg=Alert!Unable to
save.Contact Admin");
        }
    }
}

```

// <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the code.">

```

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse
response)

    throws ServletException, IOException {

    processRequest(request, response);
}

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse
response)

    throws ServletException, IOException {

    processRequest(request, response);
}

```

```

/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */

@Override

public String getServletInfo() {
    return "Short description";
}

// Extract file name from content-disposition header of file part
private String getFileName(Part part) {
    final String partHeader = part.getHeader("content-disposition");
    System.out.println("***** partHeader: " + partHeader);
    for (String content : part.getHeader("content-disposition").split(";")) {
        if (content.trim().startsWith("filename")) {
            return content.substring(content.indexOf('=') + 1).trim()
                .replace("\"", "");
        }
    }
    return null;
}
}

```

## **VALIDATE USER:**

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

```
package com.nura.servlet;
```

```
import com.nura.dao.impl.UserDetailsDAOImpl;
```

```
import com.nura.entity.UserDetails;
```

```
import java.io.IOException;
```

```
import java.io.InputStream;
```

```
import java.io.PrintWriter;
```

```
import java.text.ParseException;
```

```
import java.text.SimpleDateFormat;
```

```
import java.util.Date;
```

```
import java.util.logging.Level;
```

```
import java.util.logging.Logger;
```

```
import javax.servlet.ServletException;
```

```
import javax.servlet.annotation.MultipartConfig;
```

```
import javax.servlet.annotation.WebServlet;
```

```
import javax.servlet.http.HttpServlet;
```

```

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.Part;

/**
 *
 * @author ArunRamya
 */
@MultipartConfig
@WebServlet(name = "UserRegistration", urlPatterns = {"/UserRegistration"})
public class UserRegistration extends HttpServlet {

    /**
     * Processes requests for both HTTP GET and
     * POST
     * methods.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */

```



```
protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
```

```
throws ServletException, IOException {
```

```
response.setContentType("text/html;charset=UTF-8");
```

```
try (PrintWriter out = response.getWriter()) {
```

```
    /* TODO output your page here. You may use following sample code. */
```

```
    UserDetails _userDtls = new UserDetails();
```

```
    SimpleDateFormat sdf = new SimpleDateFormat("yyyy-mm-dd");
```

```
    Date date = null;
```

```
    try {
```

```
        date = sdf.parse(request.getParameter("dob"));
```

```
    } catch (ParseException ex) {
```

```
        Logger.getLogger(UserRegistration.class.getName()).log(Level.SEVERE, null,
ex);
```

```
    }
```

```
    _userDtls.setDateOfBirth(date);
```

```
    _userDtls.setGender(request.getParameter("gender"));
```

```
    // _userDtls.setLivesIn(request.getParameter("lives_in"));
```

```
    _userDtls.setMobileNumber(request.getParameter("mobile_no"));
```

```
    _userDtls.setPassword(request.getParameter("password"));
```

```
    // _userDtls.setStudiedAt(request.getParameter("studied_at"));
```

```
    _userDtls.setUsername(request.getParameter("username"));
```

```

_userDtls.setSpecialist(request.getParameter("disease_name"));

_userDtls.setFirstName(request.getParameter("usr_name"));

// _userDtls.setWorkedAt(request.getParameter("worked_at"));

//String[] userLikes = request.getParameterValues("user_likes");

try {
    for (Part parts : request.getParts()) {
        System.out.println("Parts");

        InputStream inputStream = parts.getInputStream();

        byte[] getImgytes = new byte[inputStream.available()];

        inputStream.read(getImgytes);

        String fileName = getFileName(parts);

        System.out.println("File name:-" + fileName);

        if (fileName != null) {
            _userDtls.setUserImage(getImgytes);
        }
    }
} catch (Exception ex) {
    System.out.println(ex.getLocalizedMessage());
}

if (new UserDetailsDAOImpl().saveUserDtls(_userDtls)) {
    response.sendRedirect("loginPage.jsp?msg=User Details saved");
} else {

```

```
        response.sendRedirect("loginPage.jsp?msg=Alert!Unable to  
save.Contact Admin");
```

```
    }
```

```
}
```

```
}
```

```
// <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the  
+ sign on the left to edit the code.">
```

```
@Override
```

```
protected void doGet(HttpServletRequest request, HttpServletResponse  
response)
```

```
    throws ServletException, IOException {
```

```
        processRequest(request, response);
```

```
}
```

```
@Override
```

```
protected void doPost(HttpServletRequest request, HttpServletResponse  
response)
```

```
    throws ServletException, IOException {
```

```
        processRequest(request, response);
```

```
}
```

```
@Override
```

```
public String getServletInfo() {
```

```

        return "Short description";
    } // </editor-fold>

    // Extract file name from content-disposition header of file part
    private String getFileName(Part part) {
        final String partHeader = part.getHeader("content-disposition");
        System.out.println("***** partHeader: " + partHeader);
        for (String content : part.getHeader("content-disposition").split(";")) {
            if (content.trim().startsWith("filename")) {
                return content.substring(content.indexOf('=') + 1).trim()
                    .replace("\"", "");
            }
        }
        return null;
    }
}

```

## **PATIENT APPOINTMENT:**

```

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

```

```
package com.nura.servlet;

import com.nura.dao.impl.UserDetailsDAOImpl;
import com.nura.entity.UserDetails;
import java.io.IOException;
import java.io.InputStream;
import java.io.PrintWriter;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.servlet.ServletException;
import javax.servlet.annotation.MultipartConfig;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.Part;

/**
 *
 * @author ArunRamya

```

```

*/

@MultipartConfig

@WebServlet(name = "UserRegistration", urlPatterns = {"/UserRegistration"})

public class UserRegistration extends HttpServlet {

    /**
     * Processes requests for both HTTP GET and
     * POST
     * methods.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */

    protected void processRequest(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            /* TODO output your page here. You may use following sample code. */

            UserDetails _userDtls = new UserDetails();

            SimpleDateFormat sdf = new SimpleDateFormat("yyyy-mm-dd");

```

```

Date date = null;

try {
    date = sdf.parse(request.getParameter("dob"));
} catch (ParseException ex) {

Logger.getLogger(UserRegistration.class.getName()).log(Level.SEVERE,    null,
ex);

}

_userDtls.setDateOfBirth(date);
_userDtls.setGender(request.getParameter("gender"));
// _userDtls.setLivesIn(request.getParameter("lives_in"));
_userDtls.setMobileNumber(request.getParameter("mobile_no"));
_userDtls.setPassword(request.getParameter("password"));
// _userDtls.setStudiedAt(request.getParameter("studied_at"));
_userDtls.setUserName(request.getParameter("username"));
_userDtls.setSpecialist(request.getParameter("disease_name"));
_userDtls.setFirstName(request.getParameter("usr_name"));
// _userDtls.setWorkedAt(request.getParameter("worked_at"));
//String[] userLikes = request.getParameterValues("user_likes");

try {
    for (Part parts : request.getParts()) {

```

```

        System.out.println("Parts");

        InputStream inputStream = parts.getInputStream();

        byte[] getImgytes = new byte[inputStream.available()];

        inputStream.read(getImgytes);

        String fileName = getFileName(parts);

        System.out.println("File name:-" + fileName);

        if (fileName != null) {

            _userDtls.setUserImage(getImgytes);

        }

    }

    catch (Exception ex) {

        System.out.println(ex.getLocalizedMessage());

    }

    if (new UserDetailsDAOImpl().saveUserDtls(_userDtls)) {

        response.sendRedirect("loginPage.jsp?msg=User Details saved");

    } else {

        response.sendRedirect("loginPage.jsp?msg=Alert!Unable to
save.Contact Admin");

    }

}

}

}

```



```
// <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the  
+ sign on the left to edit the code.">
```

```
/**
```

```
 * Handles the HTTP <code>GET</code> method.
```

```
 *
```

```
 * @param request servlet request
```

```
 * @param response servlet response
```

```
 * @throws ServletException if a servlet-specific error occurs
```

```
 * @throws IOException if an I/O error occurs
```

```
 */
```

```
@Override
```

```
protected void doGet(HttpServletRequest request, HttpServletResponse  
response)
```

```
    throws ServletException, IOException {
```

```
    processRequest(request, response);
```

```
}
```

```
/**
```

```
 * Handles the HTTP <code>POST</code> method.
```

```
 *
```

```
 * @param request servlet request
```

```
 * @param response servlet response
```

```
 * @throws ServletException if a servlet-specific error occurs
```

```

    * @throws IOException if an I/O error occurs
    */

    @Override

    protected void doPost(HttpServletRequest request, HttpServletResponse
response)

        throws ServletException, IOException {

        processRequest(request, response);

    }

    /**

    * Returns a short description of the servlet.

    *

    * @return a String containing servlet description

    */

    @Override

    public String getServletInfo() {

        return "Short description";

    } // </editor-fold>

    // Extract file name from content-disposition header of file part

    private String getFileName(Part part) {

        final String partHeader = part.getHeader("content-disposition");

        System.out.println("***** partHeader: " + partHeader);

```

```

    for (String content : part.getHeader("content-disposition").split(";")) {
        if (content.trim().startsWith("filename")) {
            return content.substring(content.indexOf('=') + 1).trim()
                .replace("\"", "");
        }
    }
    return null;
}
}

```

## **DISEASE UPDATION:**

```

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

```

```

package com.nura.servlet;

```

```

import com.nura.dao.impl.DiseaseDtlsDAOImpl;

```

```

import com.nura.entity.DiseaseMaster;

```

```

import java.io.IOException;

```

```

import java.io.PrintWriter;

```

```

import javax.servlet.ServletException;

```

```

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;


/**
 *
 * @author ArunRamya
 */

public class DiseaseUpdation extends HttpServlet {

    protected void processRequest(HttpServletRequest request, HttpServletResponse
response)

        throws ServletException, IOException {

        response.setContentType("text/html;charset=UTF-8");

        try (PrintWriter out = response.getWriter()) {

            /* TODO output your page here. You may use following sample code. */

            DiseaseMaster _disMaster = new DiseaseMaster();

            _disMaster.setDiseaseName(request.getParameter("disease_name"));

            _disMaster.setDiseaseDesc(request.getParameter("disease_desc"));

            boolean exist = new
DiseaseDtlsDAOImpl().diseaseExist(_disMaster.getDiseaseName());

            if (!exist) {

                if (new DiseaseDtlsDAOImpl().saveDiseaseDtls(_disMaster)) {

                    response.sendRedirect("respPage.jsp?msg=Details saved
successfully");

```

```

        } else {
            response.sendRedirect("respPage.jsp?msg=Unable to save! Contact
admin");
        }
    }else{
        response.sendRedirect("respPage.jsp?msg=Details already exist");
    }
}
}
}

```

// <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the code.">

```

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse
response)

    throws ServletException, IOException {

    processRequest(request, response);

}

```

```

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse
response)

    throws ServletException, IOException {

    processRequest(request, response);

```

```
}
```

```
@Override
```

```
public String getServletInfo() {
```

```
    return "Short description";
```

```
}// </editor-fold>
```

```
}
```

## **SYMPTOMS UPDATION:**

```
/*
```

```
* To change this license header, choose License Headers in Project Properties.
```

```
* To change this template file, choose Tools | Templates
```

```
* and open the template in the editor.
```

```
*/
```

```
package com.nura.servlet;
```

```
import com.nura.dao.impl.DiseaseSymptomsDAOImpl;
```

```
import com.nura.entity.DiseaseSymptoms;
```

```
import java.io.IOException;
```

```
import java.io.PrintWriter;
```

```
import javax.servlet.ServletException;
```

```

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

/**
 *
 * @author ArunRamya
 */

public class DiseaseSymptUpdation extends HttpServlet {

    protected void processRequest(HttpServletRequest request,
    HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            /* TODO output your page here. You may use following sample code. */

            String disName = request.getParameter("disease_name");

            String disSym = request.getParameter("disease_smp");

            DiseaseSymptomsDAOImpl _disSymImpl = new
            DiseaseSymptomsDAOImpl();

            DiseaseSymptoms _disSympt = new DiseaseSymptoms();

            _disSympt.setDiseaseName(request.getParameter("disease_name"));

            _disSympt.setDiseaseSymptoms(request.getParameter("disease_smp"));

```

```

        if(!_disSymImpl.getDiseaseSymptomBsdOnNameNSymptom(disName,
disSym)){

            response.sendRedirect("respPage.jsp?msg=Symptom  already  exist  for
the given disease");

        }else{

            if(!_disSymImpl.saveDiseaseSymptoms(_disSympt)){

                response.sendRedirect("respPage.jsp?msg=Details          saved
successfully");

            }else{

                response.sendRedirect("respPage.jsp?msg=Details          saved
successfully");

            }

        }

    }

}

```

// <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the code.">

```
/**
```

```
* Handles the HTTP <code>GET</code> method.
```

```
*
```

```
* @param request servlet request
```

```
* @param response servlet response
```

```
* @throws ServletException if a servlet-specific error occurs
```



```

    * @throws IOException if an I/O error occurs
    */

    @Override

    protected void doGet(HttpServletRequest request, HttpServletResponse
response)

        throws ServletException, IOException {

        processRequest(request, response);

    }

    /**

    * Handles the HTTP <code>POST</code> method.

    *

    * @param request servlet request

    * @param response servlet response

    * @throws ServletException if a servlet-specific error occurs

    * @throws IOException if an I/O error occurs

    */

    @Override

    protected void doPost(HttpServletRequest request, HttpServletResponse
response)

        throws ServletException, IOException {

        processRequest(request, response);

    }

```

```

/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */

@Override
public String getServletInfo() {
    return "Short description";
} // </editor-fold>

}

```

## **BEST DOCTOR FEEDBACK:**

```

/**
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

package com.nura.servlet;

import com.nura.dao.impl.DiseaseSymptomsDAOImpl;
import com.nura.entity.DiseaseSymptoms;

```

```

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;


/**
 *
 * @author ArunRamya
 */

public class DiseaseSymptUpdation extends HttpServlet {


    /**
     * Processes requests for both HTTP GET and
     POST
     * methods.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */

```

```

protected void processRequest(HttpServletRequest request,
HttpServletResponse response)

    throws ServletException, IOException {
response.setContentType("text/html;charset=UTF-8");
try (PrintWriter out = response.getWriter()) {
    /* TODO output your page here. You may use following sample code. */
    String disName = request.getParameter("disease_name");
    String disSym = request.getParameter("disease_smp");

    DiseaseSymptomsDAOImpl _disSymImpl = new
DiseaseSymptomsDAOImpl();

    DiseaseSymptoms _disSympt = new DiseaseSymptoms();
    _disSympt.setDiseaseName(request.getParameter("disease_name"));
    _disSympt.setDiseaseSymptoms(request.getParameter("disease_smp"));

    if(_disSymImpl.getDiseaseSymptomBsdOnNameNSymptom(disName,
disSym)){

        response.sendRedirect("respPage.jsp?msg=Symptom already exist for
the given disease");

    }else{

        if(_disSymImpl.saveDiseaseSymptoms(_disSympt)){

            response.sendRedirect("respPage.jsp?msg=Details saved
successfully");

        }else{

            response.sendRedirect("respPage.jsp?msg=Details saved
successfully");

        }
    }
}

```

```

    }
}
}

```

// <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the code.">

```

/**
 * Handles the HTTP <code>GET</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse
response)
    throws ServletException, IOException {
    processRequest(request, response);
}

/**
 * Handles the HTTP <code>POST</code> method.

```

```

*
* @param request servlet request
* @param response servlet response
* @throws ServletException if a servlet-specific error occurs
* @throws IOException if an I/O error occurs
*/

@Override
protected void doPost(HttpServletRequest request, HttpServletResponse
response)
    throws ServletException, IOException {
    processRequest(request, response);
}

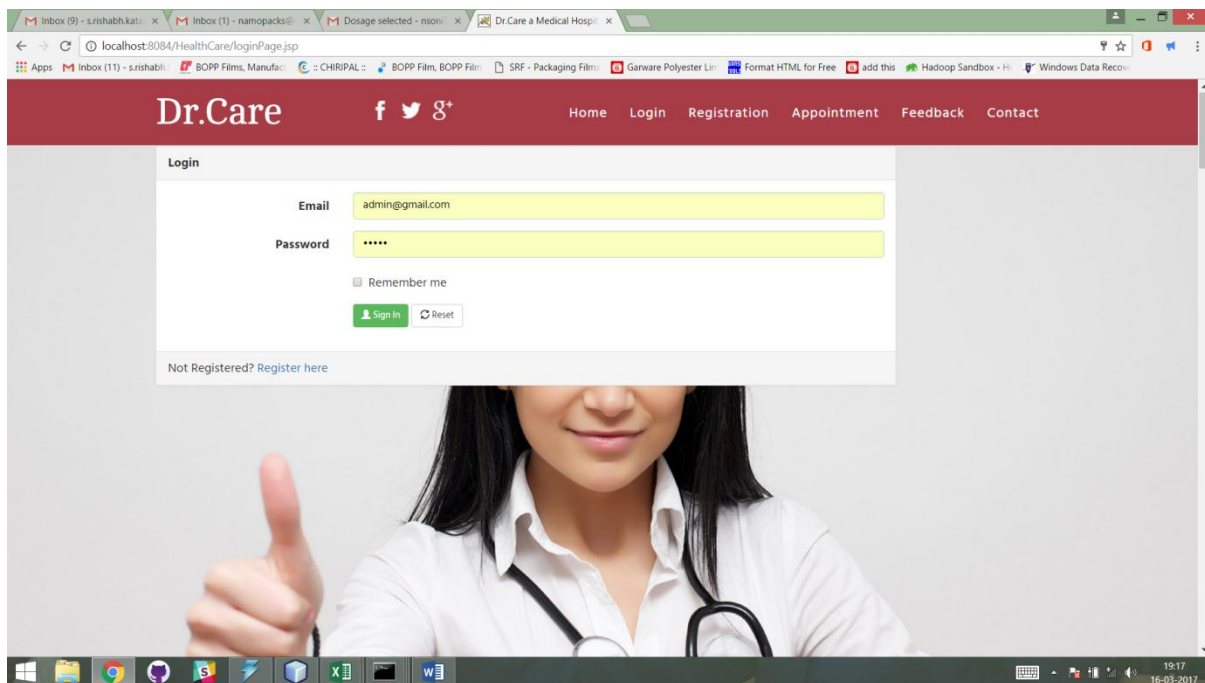
/**
* Returns a short description of the servlet.
*
* @return a String containing servlet description
*/

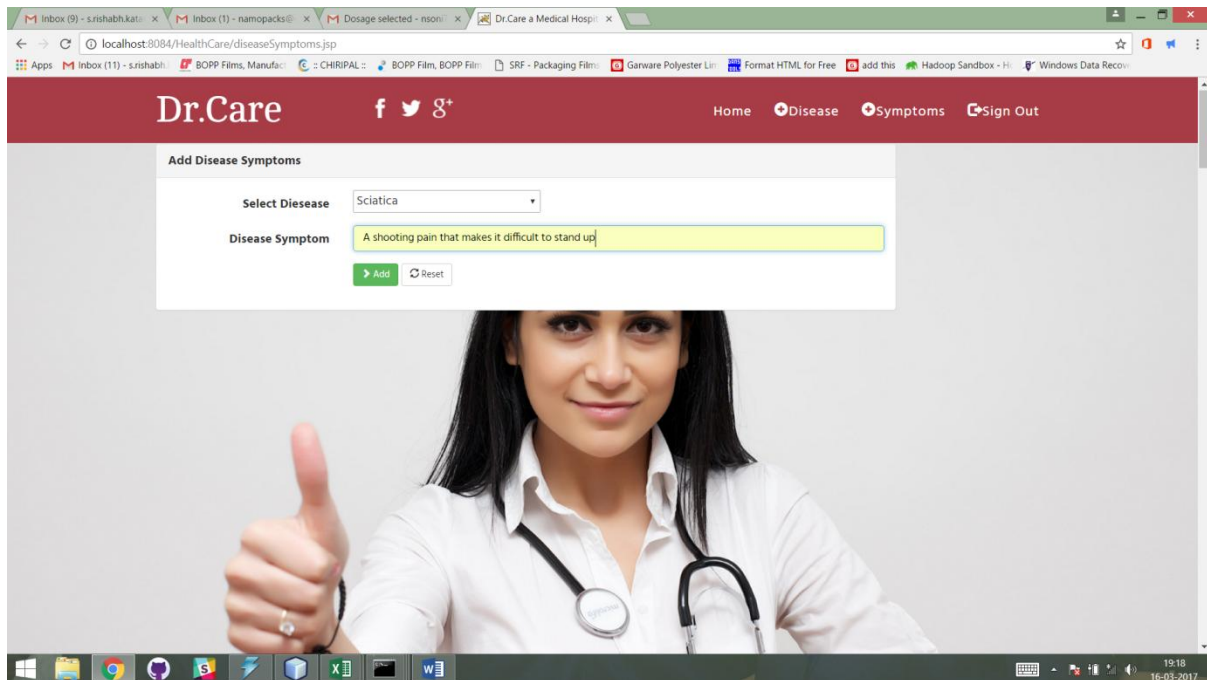
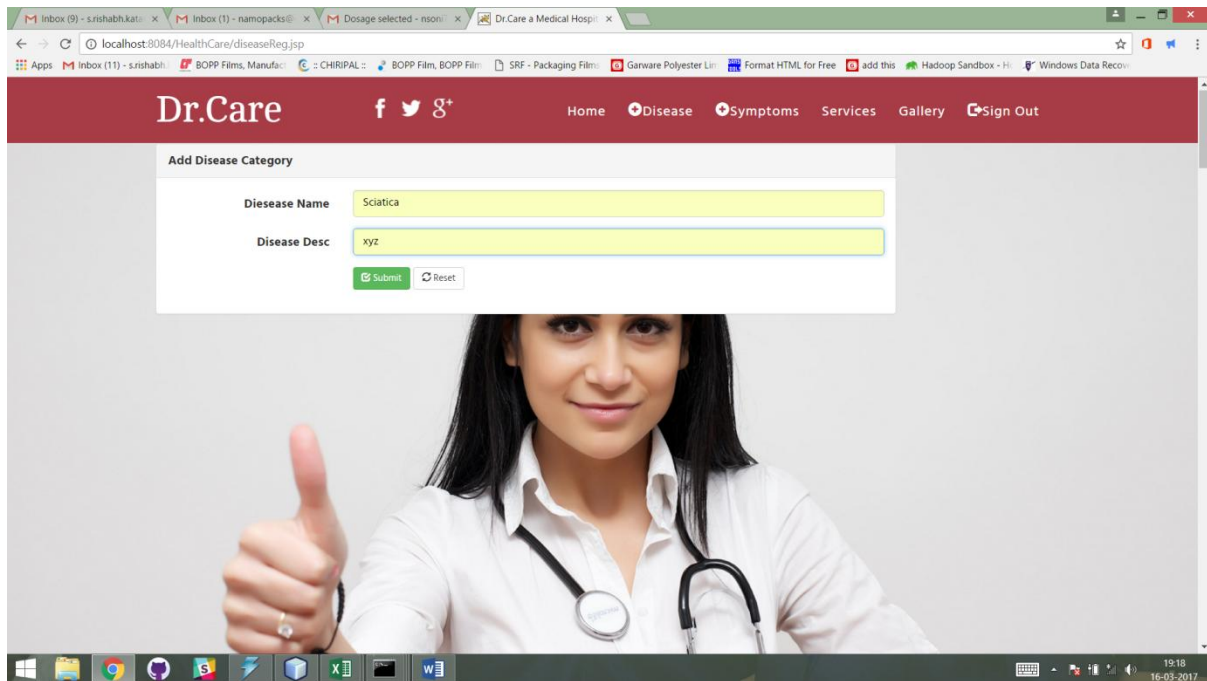
@Override
public String getServletInfo() {
    return "Short description";
} // </editor-fold>

```

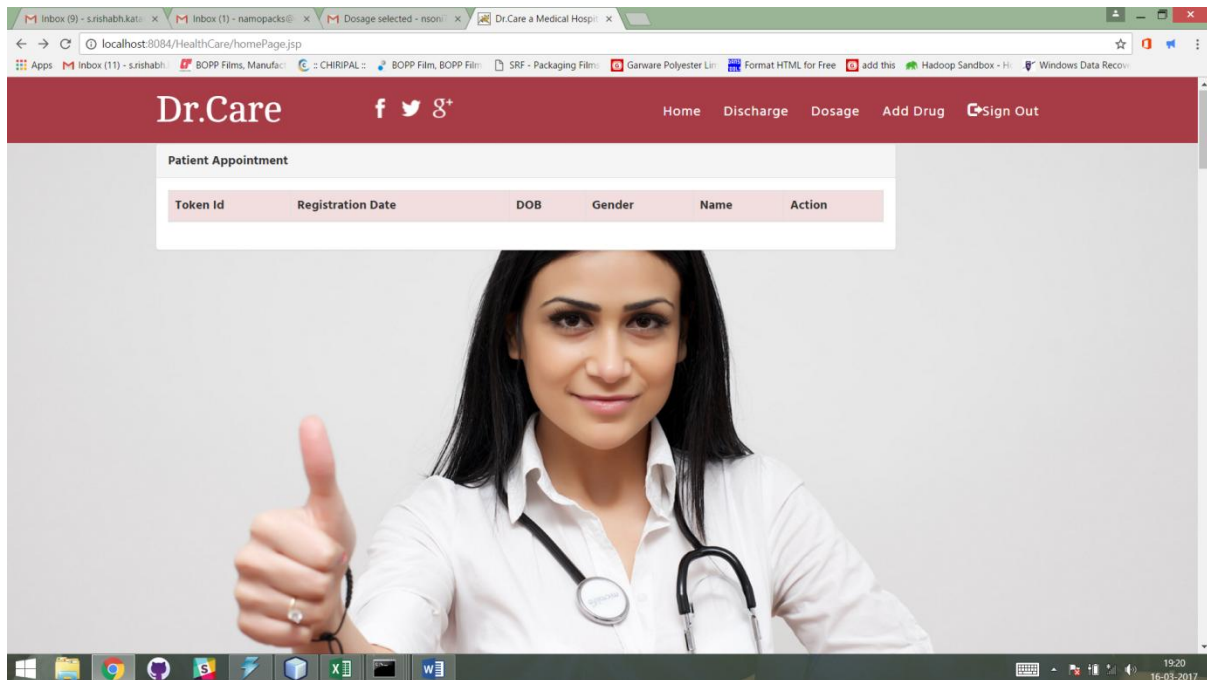
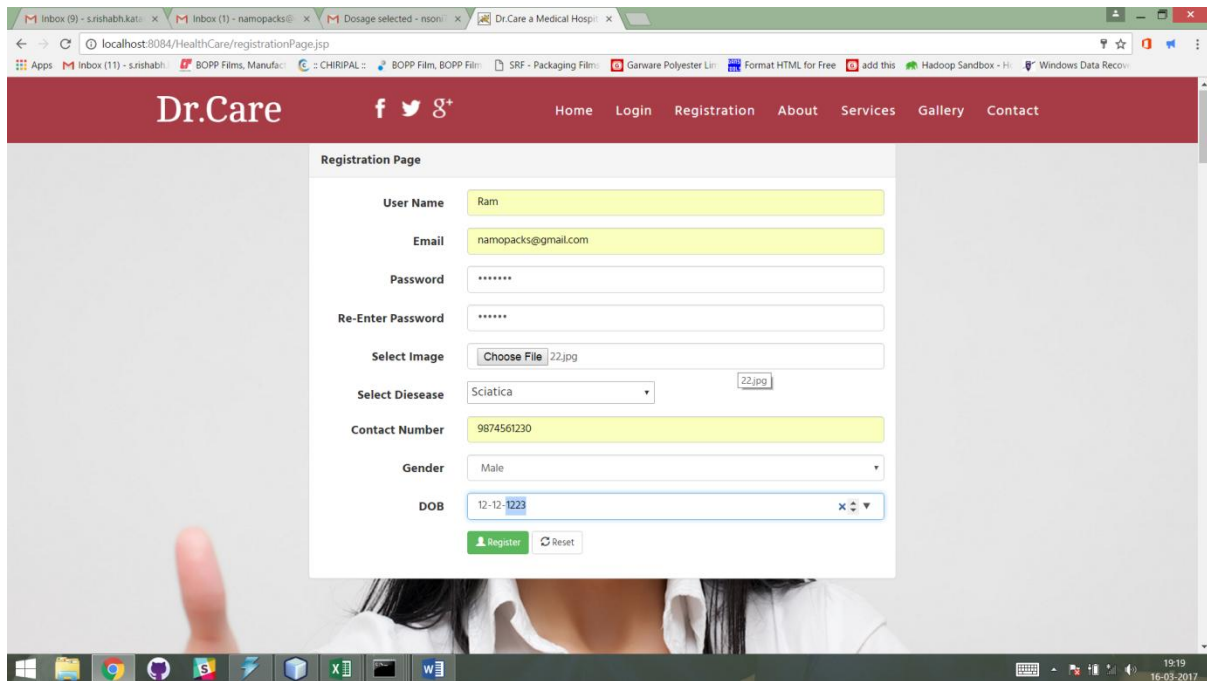
}

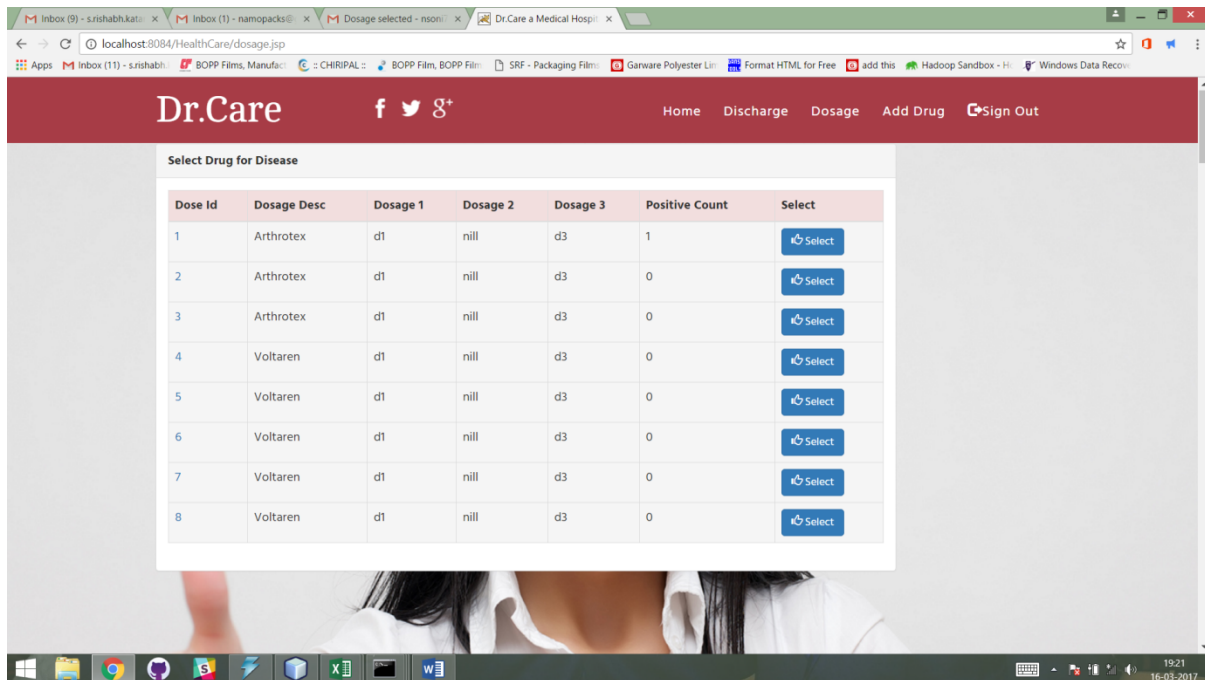
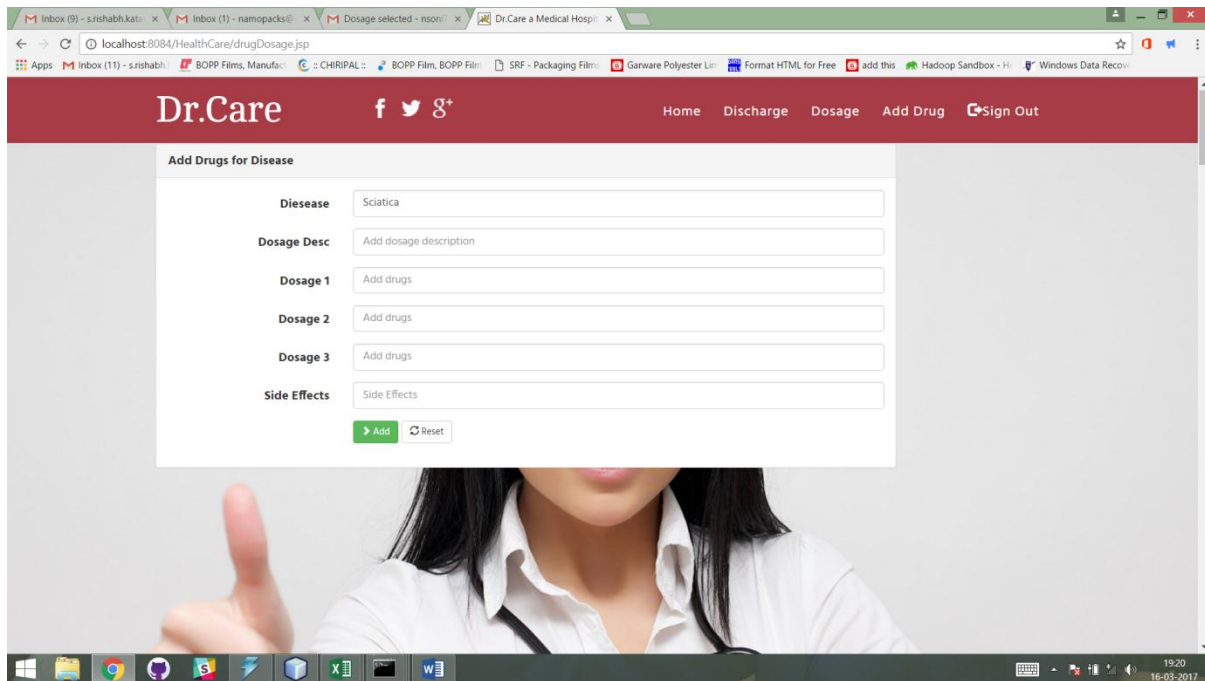
## 8.2 SNAP SHOTS:











```
Apache Hadoop Distribution - hadoop namenode
127.0.0.1 19:23:58 INFO BlockStateChange: BLOCK<addStoredBlock,blockMap updated>
162.168.2.24:19:50:10 is added to blk_1073741832_1008[blkCUSTest=UNDER_CONSTRUCTION][DfSKIDJ=a742da-8236-aa74b2774891.NORMAL.IRW]] size 0
17/03/16 19:23:58 INFO HdfsFile.StateChange: DIRM completefile: /user/rishab/input/
.h.json is closed by DFSClient_NONMAPREDUCE_-1541120673.1
17/03/16 19:23:59 INFO BlockStateChange: BLOCK<addToInvalidates: blk_1073741833_
1008 127.0.0.1:150010
17/03/16 19:24:01 INFO HdfsFile.StateChange:
80010 to delete [blk_1073741833_1008, blk_
17/03/16 19:24:01 INFO HdfsFile.StateChange:
this_pos_count desc
temporarily/attempt_local131067534_0001
192.168.2.24:1489670475531 blk_1073741833_
primaryUserDataIndex=-1, replicas=(ReplicaUnde
2da-8236-aa74b2774891.NORMAL.IRW)] si
17/03/16 19:24:03 INFO DataNode.clientreac
127.0.0.1:150010 is added to blk_1073741
M_primaryUserDataIndex=-1, replicas=(Replica
4-8236-aa74b2774891.NORMAL.IRW)] si
temporarily/attempt_local131067534_0001
ient NONMAPREDUCE_-1541120673.1
Apache Hadoop Distrib
168.2.24:1489670475531 blk_1073741832_10
entPb:1792220094 192.168.2.24:1489670475
17/03/16 19:24:01 INFO impl.FDataNodesDesu
168.2.24:1489670475531 blk_1073741833_100
entPb:1792220094 192.168.2.24:1489670475
17/03/16 19:24:02 INFO DataNode.clientreac
0.1:50093, bytes: 406, op: HDFS_READ, c
1_offset: 0, srcUID: 2066104d-9e66-40f0
4-192.168.2.24:1489670475531 blk_1073741
17/03/16 19:24:03 INFO DataNode.DataNode:
89670475531 blk_1073741833_1009 op: /12
17/03/16 19:24:03 INFO DataNode.clientreac
0.1:50010, bytes: 44, op: HDFS_WRITE, cliID: DFSclient_NONMAPREDUCE_-1541120673
0.1:50010, op: srcUID: 2066104d-9e66-40f0-21caddb82f, blockID: Pb-1792220094
192.168.2.24:1489670475531 blk_1073741833_1009 duration: 3075987
17/03/16 19:24:03 INFO DataNode.DataNode: PacketResponder: Pb-1792220094:192.168
2.24:1489670475531 blk_1073741833_1009, time:LAST_IN_PIPELNE, downstreams:0]
terminating
17/03/16 19:24:03 INFO DataNode.clientreac: op:/127.0.0.1:50010, dest:/127.0.
0.1:50011, bytes: 48, op: HDFS_READ, cliID: DFSclient_NONMAPREDUCE_-1541120673
1_offset: 0, srcUID: 2066104d-9e66-40f0-809a:21caddb82f, blockID: Pb-1792220094
192.168.2.24:1489670475531 blk_1073741833_1009, duration: 1662302
17/03/16 19:24:04 INFO DataNode.PacketValidation: Verification succeeded for
Pb-1792220094:192.168.2.24:1489670475531 blk_1073741833_1009

Administrator: Command Prompt - hadoop -jar C:\work\HealthCareHadoop...
hibernate:
select
    userdetail0_user_id as user0_,
    userdetail0_dob as dob0_,
    userdetail0_first_name as firstName3_0_,
    userdetail0_gender as gender0_,
    userdetail0_mobile_no as mobiles0_,
    userdetail0_password as password0_,
    userdetail0_specialist_in as specialist0_,
    userdetail0_user_img as user0_0,
    userdetail0_username as username0_
from
    user_dtl0 userdetail0_
where
    userdetail0_user_id=?
hibernate:
insert
into
    doc_sel
        doc_type, doc_id, doc_sel, match_count, status, count_id)
values
    (?, ?, ?, ?, ?, ?)
mail has been sent
```

Dr.Care

Home Discharge Dosage Add Drug Sign Out

Patient Appointment

Token Id	Registration Date	DOB	Gender	Name	Action
1489672289318	Thu Mar 16 19:21:29 IST 2017	1998-10-10	Male	rajesh	Treatment

Dr.Care

Home Discharge Dosage Add Drug Sign Out

Select Drug for Disease


Dose Id	Dosage Desc	Dosage 1	Dosage 2	Dosage 3	Positive Count	Select
1	Arthrotex	d1	nill	d3	1	Select
2	Arthrotex	d1	nill	d3	0	Select
3	Arthrotex	d1	nill	d3	0	Select
4	Voltaren	d1	nill	d3	0	Select
5	Voltaren	d1	nill	d3	0	Select
6	Voltaren	d1	nill	d3	0	Select
7	Voltaren	d1	nill	d3	0	Select
8	Voltaren	d1	nill	d3	0	Select

Dr.Care

Home Discharge Services Gallery Sign Out

Patient Appointment

Token Id	Registration Date	DOB	Gender	Name	Action
1489672289318	Thu Mar 16 19:21:29 IST 2017	1998-10-10	Male	rajesh	<a href="#">Treatment</a> <a href="#">Discharge</a>



Dr.Care


Home Login Registration Appointment Feedback Contact

Enter token id to provide feedback

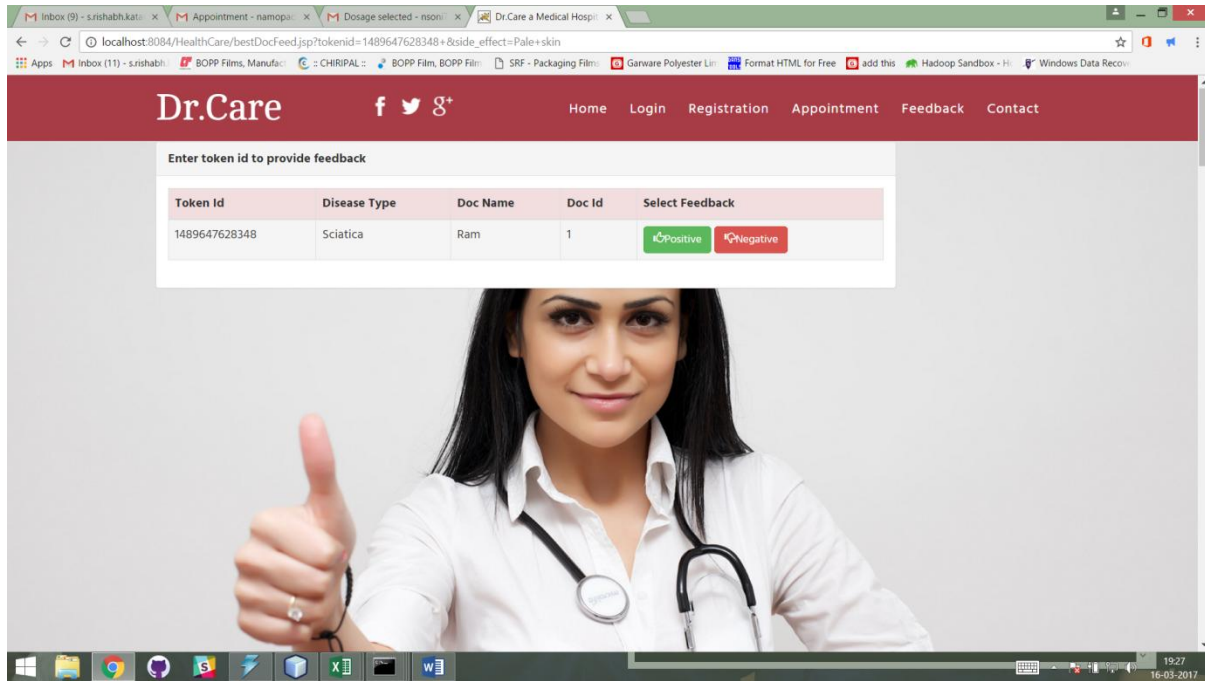
Token Id: 1489647628348

Side Effects: Pale skin

Submit Reset







## CHAPTER 9

### CONCLUSION AND

#### 9.1 CONCLUSIONS:

In this paper, we propose an approach for answering drug queries to support drug prescription. Our focus is on how to obtain and rank answers based on incomplete information and provide personalization. To cope with incomplete and noisy data, we allow both exact and close matches when answering queries. We also present an intuitive approach to display answers to users, which aims to help users to understand the ranked results and possibly refine their queries.

## 9.2 FUTURE ENHANCEMENT:

It is very important that the big data research community does not repeat the same mistake. While there is clearly an important research space examining the fundamental methods and technologies for big data analytics, it is vital to acknowledge that it is also necessary to fund domain-targeted research that allows specialised solutions to be developed for specific applications. Healthcare in general and computational biomedicine in particular, seems a natural candidate for this.

## REFERENCES

- [1] S. Khemmarat and L. Gao, “Supporting drug prescription via predictive and personalized query system,” in *PervasiveHealth*. IEEE, 2015.
- [2] C. Knox et al., “Drugbank 3.0: a comprehensive resource for omics research on drugs,” *Nucleic acids research*, vol. 39, no. suppl 1, pp. D1035–D1041, 2011.
- [3] M. Kuhn et al., “A side effect resource to capture phenotypic effects of drugs,” *Molecular systems biology*, vol. 6, no. 1, p. 343, 2010.
- [4] M. Kanehisa and S. Goto, “Kegg: kyoto encyclopedia of genes and genomes,” *Nucleic acids research*, vol. 28, no. 1, pp. 27–30, 2000.
- [5] T. Fawcett, “An introduction to roc analysis,” *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.
- [6] K. Sangkuhl et al., “Pharmgkb: understanding the effects of individual genetic variants,” *Drug Metab. Rev.*, vol. 40, no. 4, pp. 539–551, 2008.
- [7] A. Langer et al., “A text based drug query system for mobile phones,” *Int. J. Mob. Commun.*, vol. 12, no. 4, pp. 411–429, Jul. 2014.
- [8] C. Doulaverakis et al., “Panacea, a semantic-enabled drug recommendations discovery framework,” *J. Biomed. Semant.*, vol. 5, p. 13, 2014.

- [9] M. Dumontier and N. Villanueva-Rosales, “Towards pharmacogenomics knowledge discovery with the semantic web,” *Briefings in bioinformatics*, vol. 10, no. 2, pp. 153–163, 2009.
- [10] A. Ben Abacha and P. Zweigenbaum, “Medical question answering: translating medical questions into sparql queries,” in *Proceedings of the 2nd ACM SIGHIT*. ACM, 2012, pp. 41–50.
- [11] J. Jin et al., “Querying web-scale information networks through bounding matching scores,” in *WWW 2015*, 2015, pp. 527–537.
- [12] J. Jin et al., “A distributed approach for top-k star queries on massive information networks,” in *ICPADS 2014*, 2014, pp. 9–16.
- [13] A. Khan et al., “Neighborhood based fast graph search in large networks,” in *SIGMOD 2011*. ACM, 2011, pp. 901–912.
- [14] A. Khan et al., “Nema: Fast graph search with label similarity,” in *Proc. of the VLDB Endowment*, vol. 6, no. 3, 2013, pp. 181–192.
- [15] J. Sun, H. Xu, and Z. Zhao, “Network-assisted investigation of antipsychotic drugs and their targets,” *Chemistry & biodiversity*, vol. 9, no. 5, pp. 900–910, 2012.