

**BIGDATA ANALYTICS OF EFFECTIVE DISEASE
DIAGNOSIS USING MACHINE LEARNING
WITH DRUG PRESCRIPTION**

Submitted in partial fulfillment of the required for the award of

**Bachelor of computer science
J.SHARMILA (40290092)**



**DEPARTMENT OF COMPUTER SCIENCE
SCHOOL OF SCIENCE AND HUMANITIES**

**SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade "A" by NAAC | 12B Status by UGC |
Approved by AICTE**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI - 600 119**

MAY - 2023



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **J.SHARMILA**, Register Number **(40290092)** who carried out the Project entitled "**BIGDATA ANALYTICS OF EFFECTIVE DISEASE DIAGNOSIS USING MACHINE LEARNING WITH DRUG PRESCRIPTION**" under the supervision from JANUARY 2023 TO MAY 2023.

Internal Guide

Ms. AISHWARYA , MCA., Assistant Professor

DEAN AND HEAD OF THE DEPARTMENT

Dr. REKHA CHAKRAVARTHI M.E., Ph.D.

Submitted for Viva voce examination held on 8.05.2023

Internal Examiner
External Examiner

DECLARATION

I SHARMILA.J (40290092) hereby declare that the Project work entitled **“BIGDATA ANALYTICS OF EFFECTIVE DISEASE DIAGNOSIS USING MACHINE LEARNING WITH DRUG PRESCRIPTION”** done by us under the guidance of **MS.AISHWARYA.B** MSc, Assistant professor Department of computer science at **SATHYABAMA INSITIUTE OF SCIENCE AND TECHNOLOGY** is submitted in partial fulfillment of the requirements for the award of Bachelor of science degree in computer science.



DATE: 09/05/2023

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of Management of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. REKHA CHAKRAVARTHI M.E., Ph.D., Dean – Arts and science** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **MS.AISHWARYA.B** Msc., Assistant professor Department of computer science for Their valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the Department of computer science who were helpful in many ways for the completion of the training.

A handwritten signature in black ink, appearing to read 'Sharmila', is enclosed in a light gray rectangular box.

SHARMILA

TABLE OF CONTENT

S.NO	TITLE	PAGE NO
1.	Abstract	7
2.	Chapter-1 Introduction	8-12
3.	Chapter-2 2.1 Existing system 2.2 Proposed system	13
4.	Chapter-3 3.1 Architecture diagram 3.2 Data flow diagram	14
	3.2.1 Level 1	15
	Use case diagram	16
	Sequence diagram	17
	Collaboration diagram	18
	Activity diagram	19
	Class diagram	20
	Feasiblity study	20
5.	Chapter -4 4.1- Hardware environment 4.2- Software environment	21
	Over view of My SQL Database system	22
	SQL Architecture	23

S.NO	TITLE	PAGE NO
6.	Chapter 5 Features of Java 5.1- Java virtual Machine	24
	5.2 - Java API 5.3 - Java Runtime Environment	25
	5.4 - Java Server page	26
	5.5 - Introduction to apache TOM CAT	27-28
7.	Chapter- 6 Modules	29-31
8.	Chapter -7 7.1 - Appendix(source coding)	32-55
	7.2 - Snapshots	57-63
9.	Chapter - 8 8.1 - Conclusion 8.2 - Future enhancement	64
10.	Chapter-9 9.1 - Reference	65

ABSTRACT

In the Existing system, Big data mining-driven disease risk prediction has become one of the important topics in the field of e-healthcare. However, without the security and privacy assurances, disease risk prediction cannot continue to flourish. In the Proposed system, an efficient and privacy-preserving disease risk prediction scheme for e-healthcare is proposed, hereafter referred to as EPDP. Compared with the up-to-date works, the proposed EPDP comprehensively achieves two phases of disease risk prediction, i.e., disease model training and disease prediction, while ensuring the privacy preservation. In the Modification process, which is our Implementation 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. User can post the query through system or through Android Application also. We also arrange appointment to the Best Doctor for the consultation.

CHAPTER 1

INTRODUCTION

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.

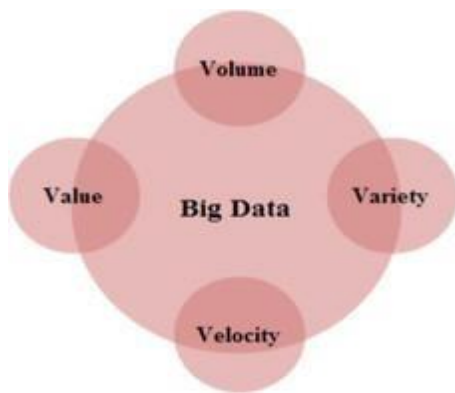


Fig Four V's Of Big Data

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

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 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 datanodes called “slaves.” HDFS can also include secondary name nodes. The namenode manages the hierarchy of file systems and director namespace (i.e., metadata). File systems are presented in a form of namenode 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.

computing problems to the MapReduce framework where scientists can efficiently utilize existing resources in the cloud to solve computationally large-scale scientific data.

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.

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.

Data Quality

In the past, data processing was typically performed on clean datasets from well-known and limited sources. Therefore, the results were accurate. 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”.

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.

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.

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 Security

Although cloud computing has transformed modern ICT technology, several unresolved security threats exist in cloud computing. 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.

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.

CHAPTER 2

SYSTEM ANALYSIS

2.1 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

2.2 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 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 3

SYSTEM DESIGN

3.1 ARCHITECTURE DIAGRAM:

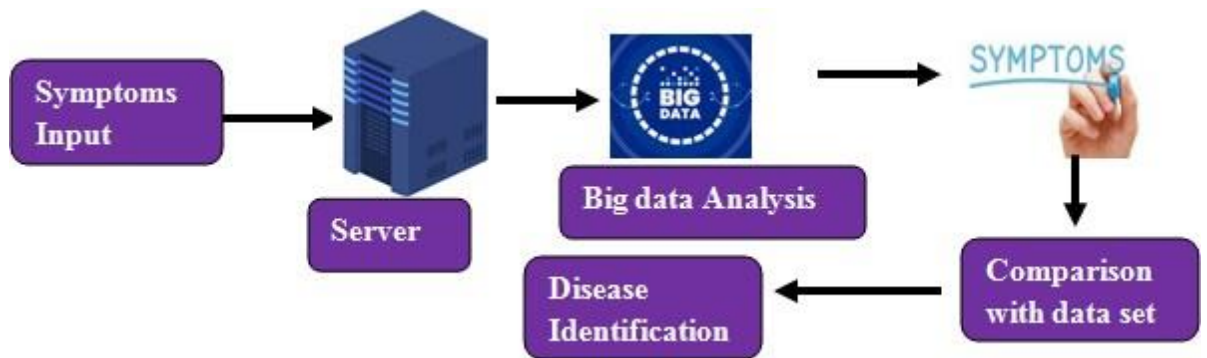


Fig. No:3.1 SYSTEM ARCHITECTURE

3.2 DATA FLOW DIAGRAM

3.2.1: LEVEL 0

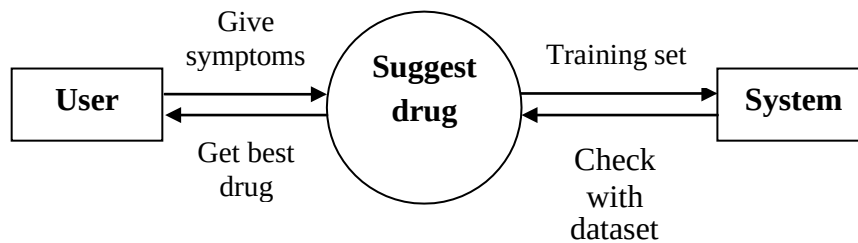
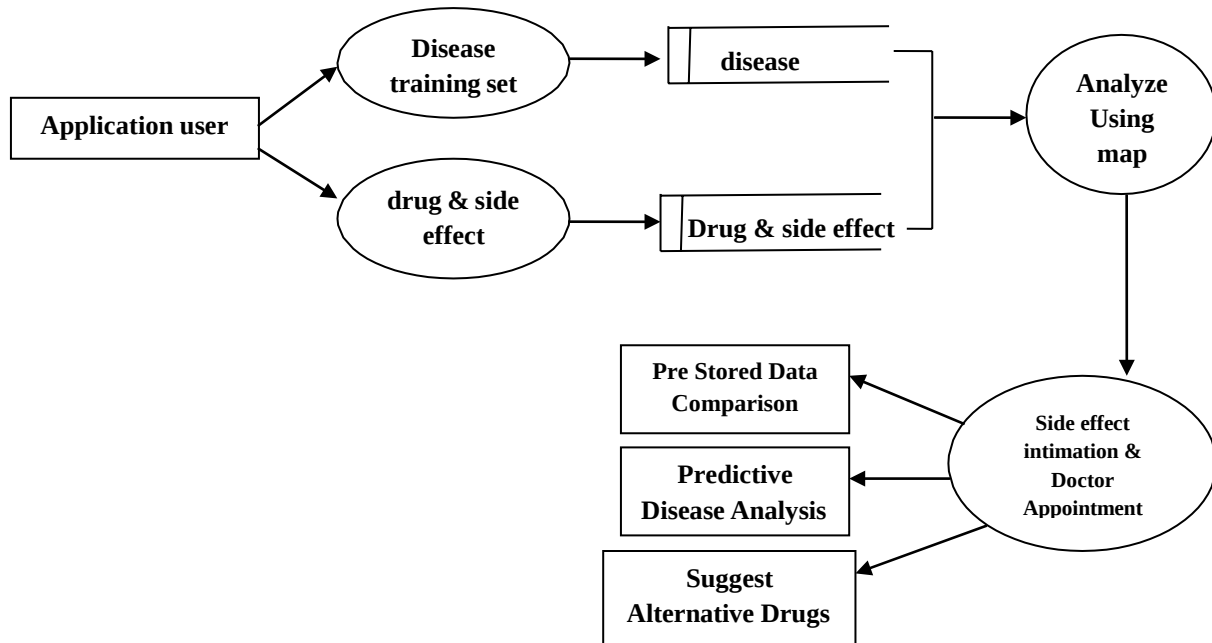


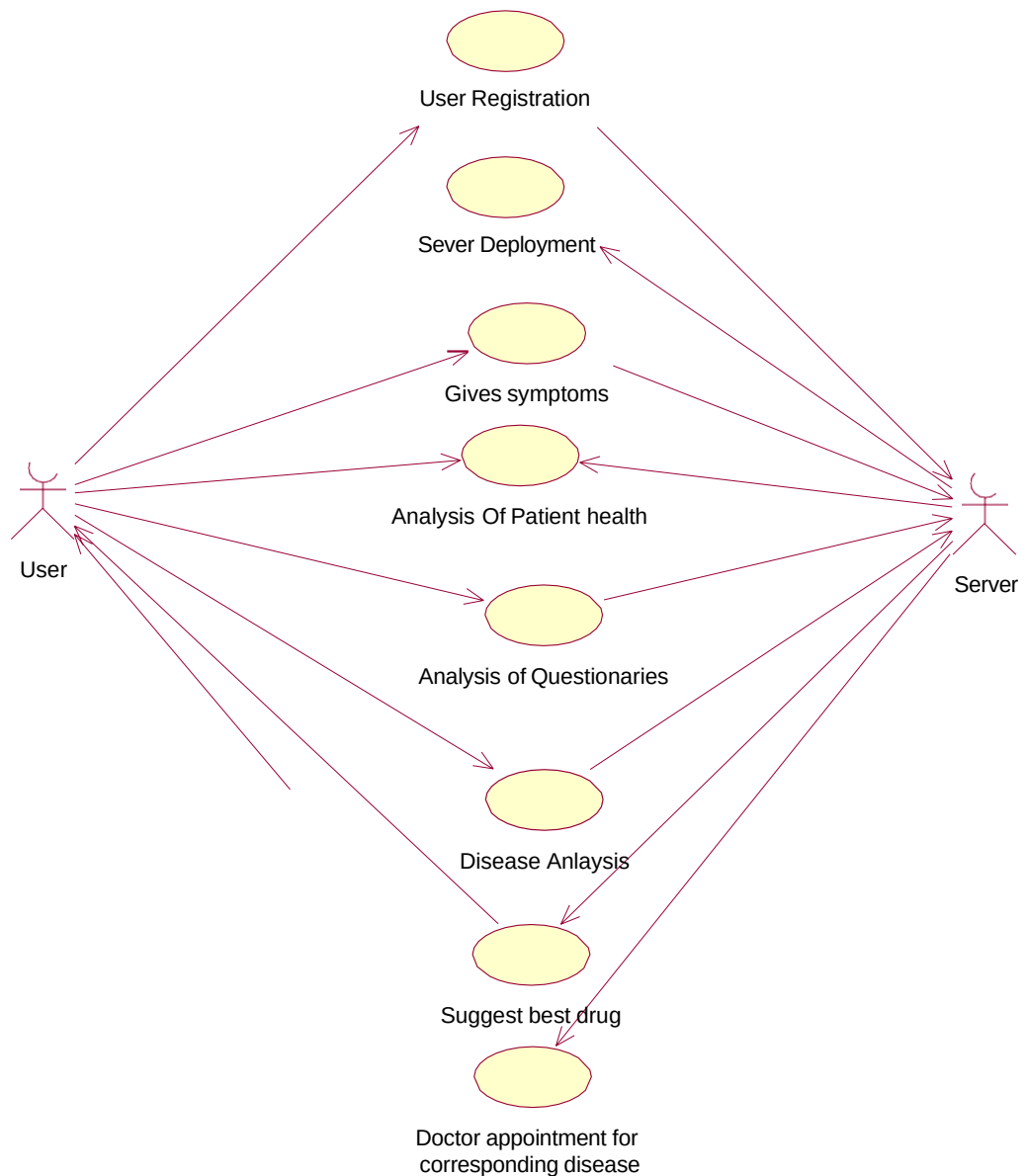
Fig. No: 3.2 Data flow diagram

3.2.2 LEVEL 1



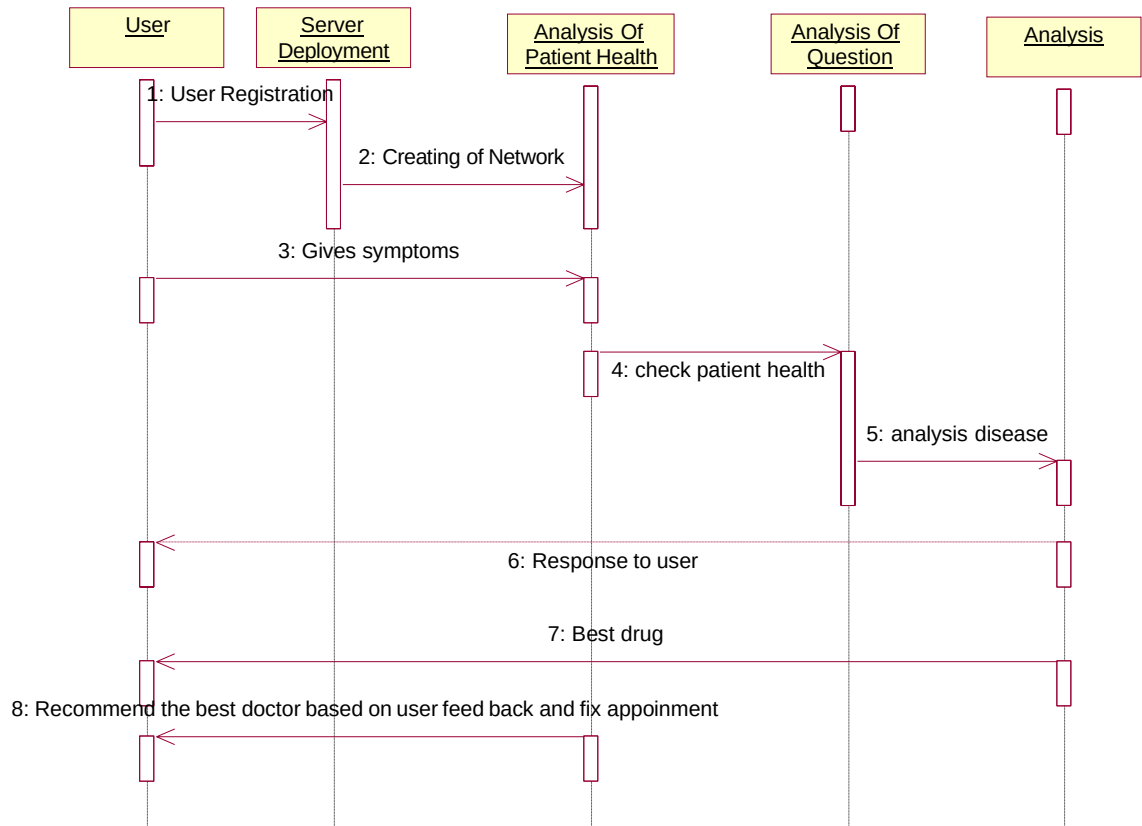
USECASE DIAGRAM

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.



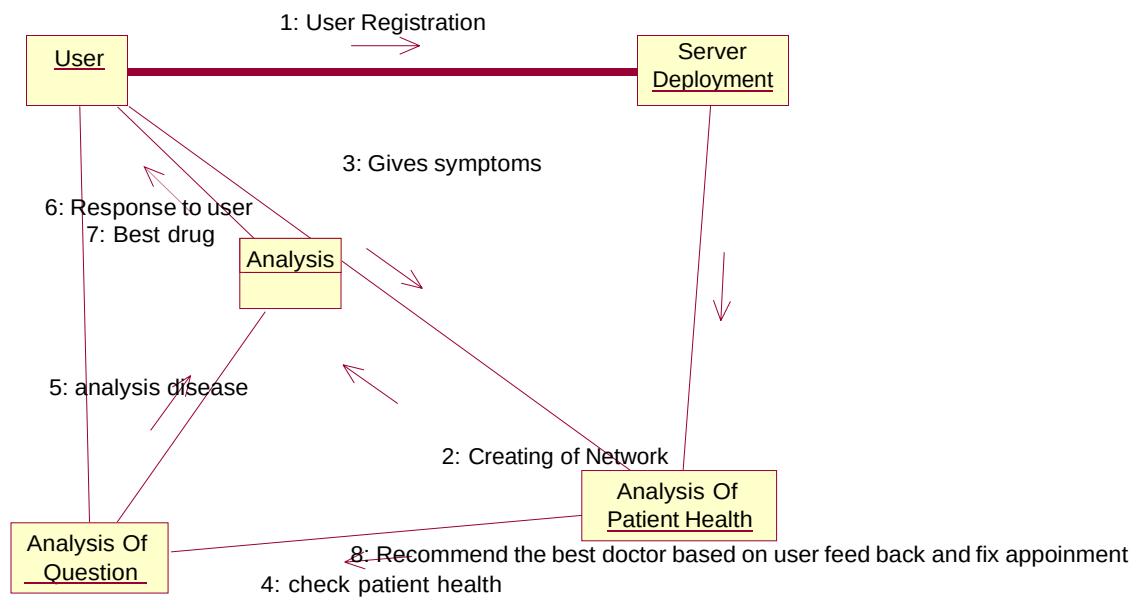
SEQUENCE DIAGRAM

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.



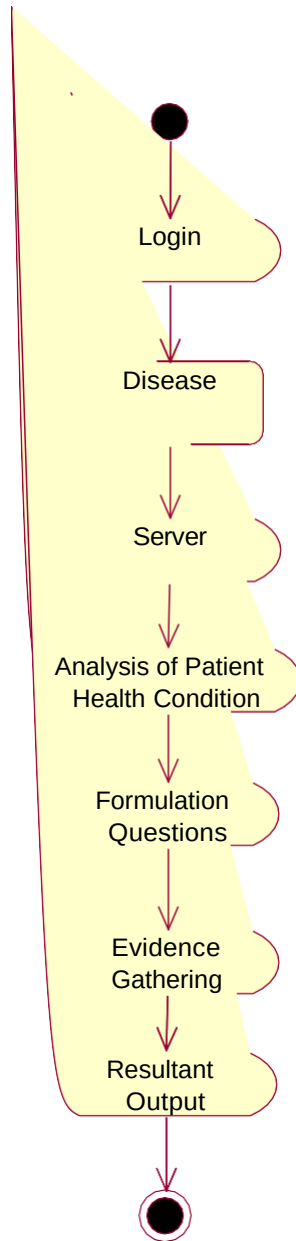
COLLABORATION DIAGRAM

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.

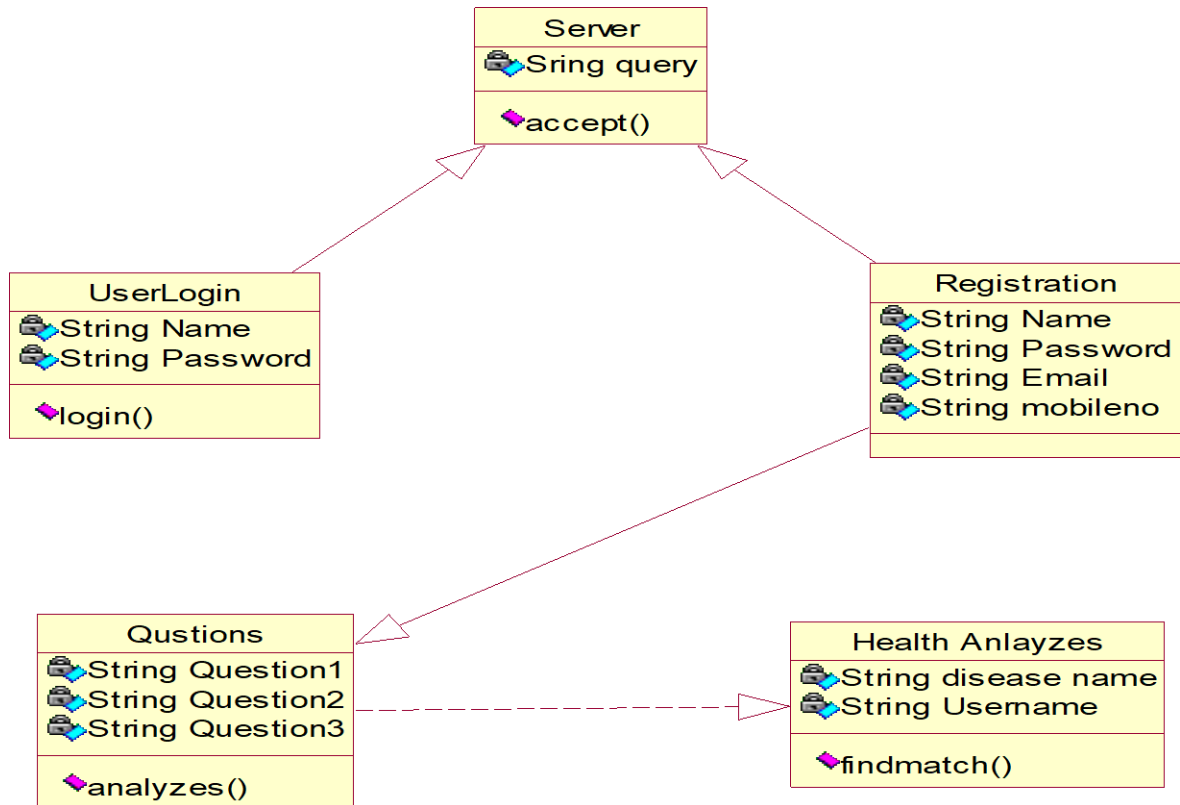


ACTIVITY DIAGRAM:

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, activity final node and activities in between.



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

CHAPTER 4

SYSTEM SPECIFICATIONS

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.

4.1 Hardware 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

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

4.2 Software Environment

The software requirements are the specification of the system. 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.

- 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 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.

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.

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.**

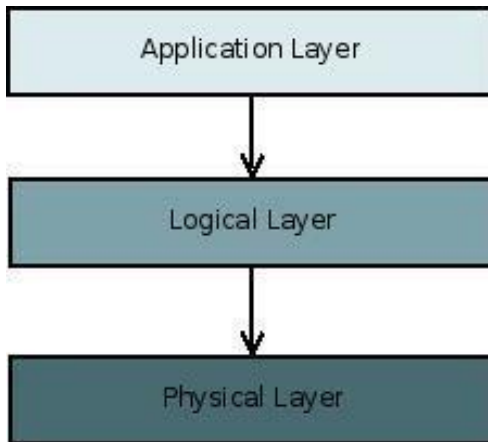
A relational database stores data in separate tables rather than putting all the data in one big storeroom. This adds speed and flexibility. The SQL part of “MySQL” stands for “Structured Query Language.”

- **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 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 APIs: .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 APIs. Each storage engine has its strengths and weaknesses, some of these engines are MyISAM, InnoDB, CSV, NDB Cluster, Falcon, etc...

4.3 Java Platform

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.

CHAPTER-5

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.

5.1 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 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.

5.2 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*.

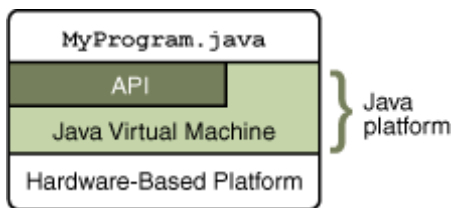


Fig 5.2 API and JVM

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.

5.3 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.

5.4 JAVASERVER 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.

5.5 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.

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. 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.

5.6 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

5.7 INTRODUCTION TO APACHE TOM CAT

Tomcat, developed by Apache (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 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. 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 6

MODULES

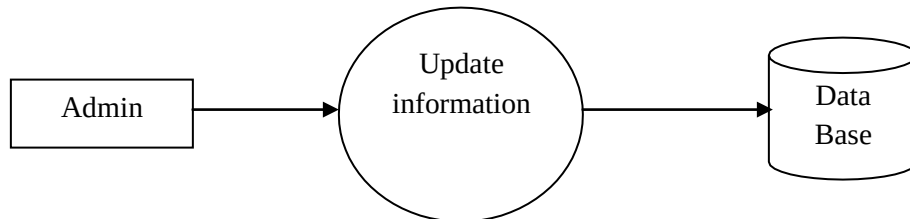
- A modular design reduces complexity, facilitates change (a critical aspect of software maintainability), and results in easier implementation by encouraging parallel development of different part of system. Software with effective modularity is easier to develop because function may be compartmentalized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

6.1 MODULES:

- 1. SERVER DEPLOYMENT**
- 2. CONSTRUCTION OF DISEASE TRAINING SET**
- 3. BIG DATA BASED ANALYSIS**
- 4. PRE STORED DATA COMPARISON**
- 5. PREDICTIVE DISEASE ANALYSIS**
- 6. DRUG AND TRAINING SET CONSTRUCTION**

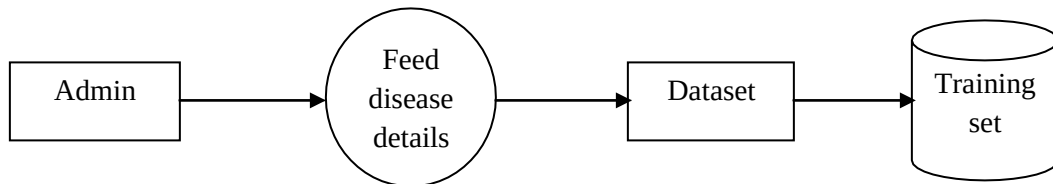
SERVER DEPLOYMENT

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.



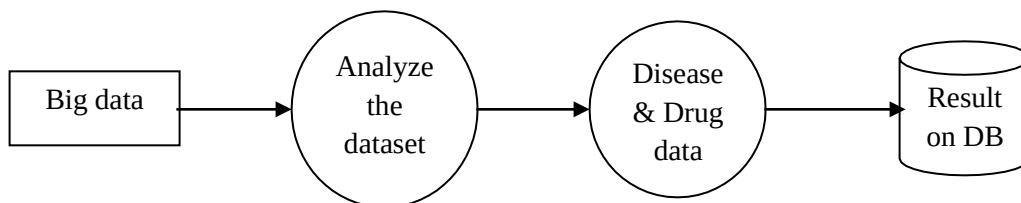
CONSTRUCTION OF DISEASE TRAINING SET

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 .



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.

CHAPTER-7

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.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);
```

```
}
```

```
/**
```

```
 * 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;
}
}

```

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

 */

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

public class UserRegistration extends HttpServlet {

/**

 * Processes requests for both HTTP <code>GET</code> and <code>POST</code>

 * 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[] getImgbytes = new byte[inputStream.available()];

inputStream.read(getImgbytes);

String fileName = getFileName(parts);

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

if (fileName != null) {

_userDtls.setUserImage(getImgbytes);

}

}

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

/**
 * Processes requests for both HTTP <code>GET</code> and <code>POST</code>
 * 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 <code>GET</code> and <code>POST</code>
 * 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);
}

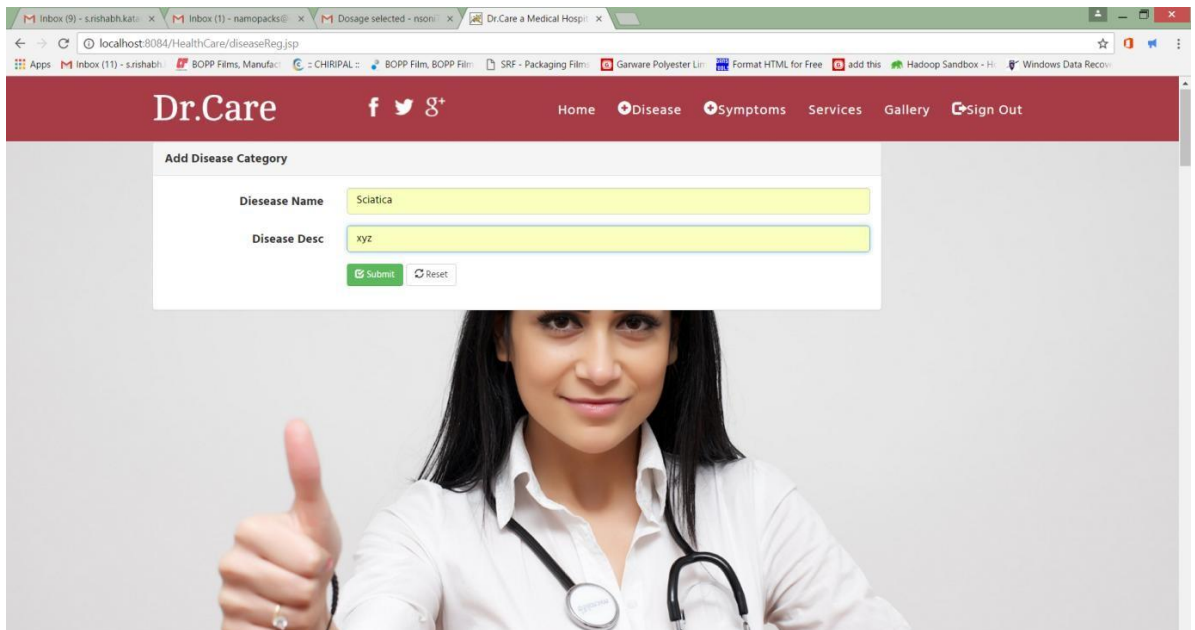
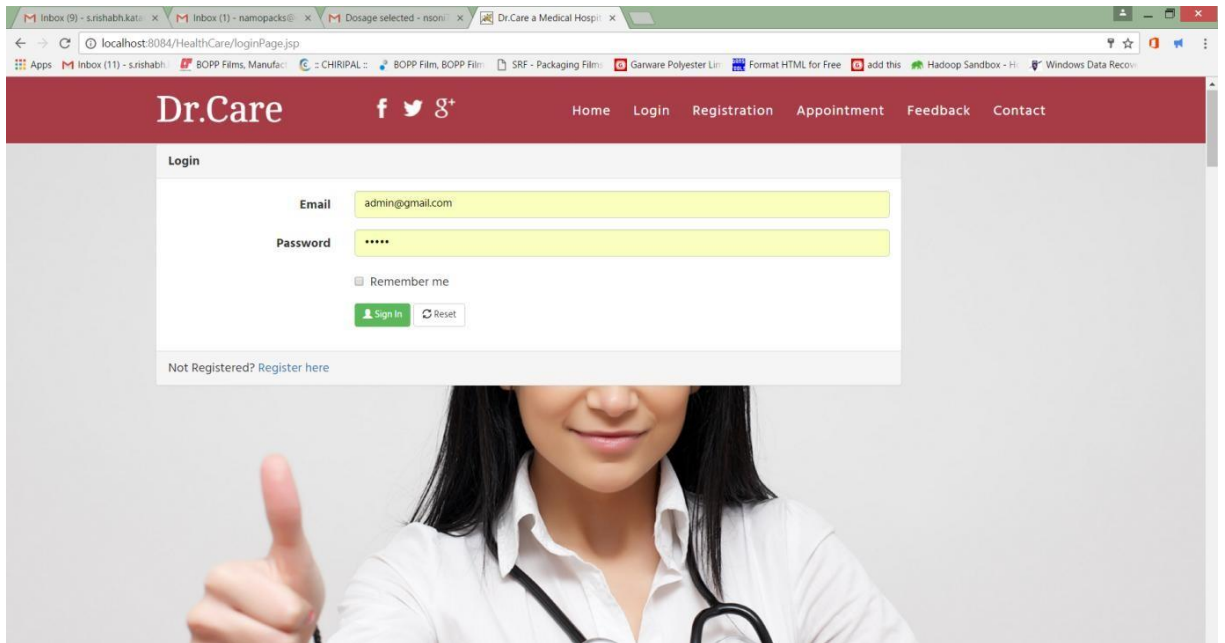
/**
 * Handles the HTTP <code>POST</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse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

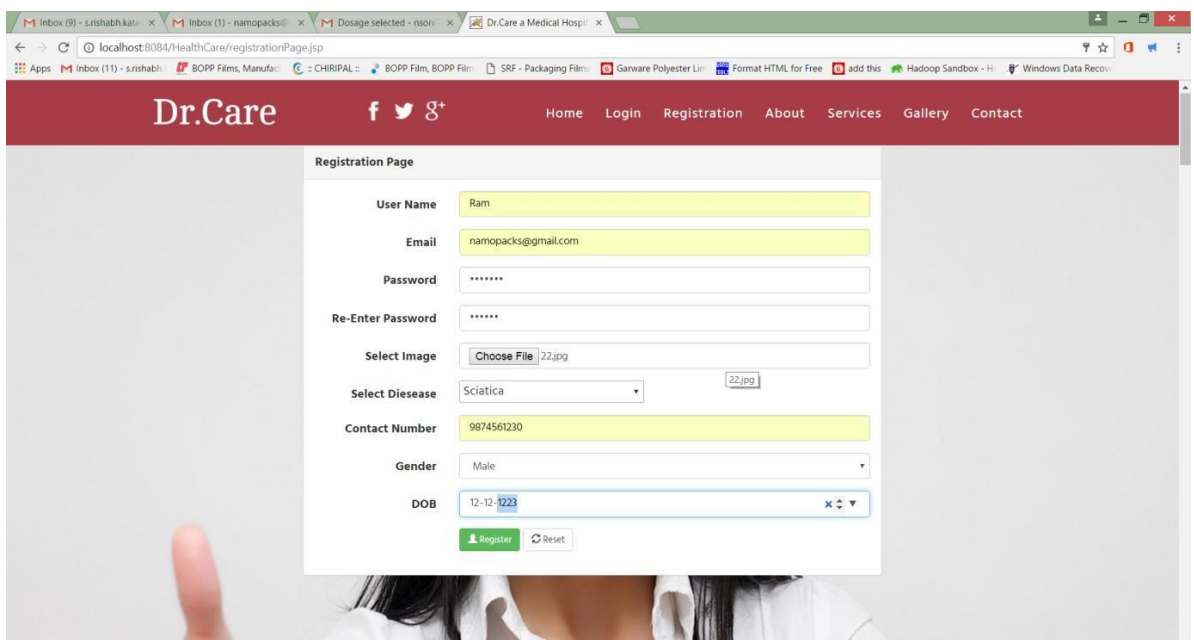
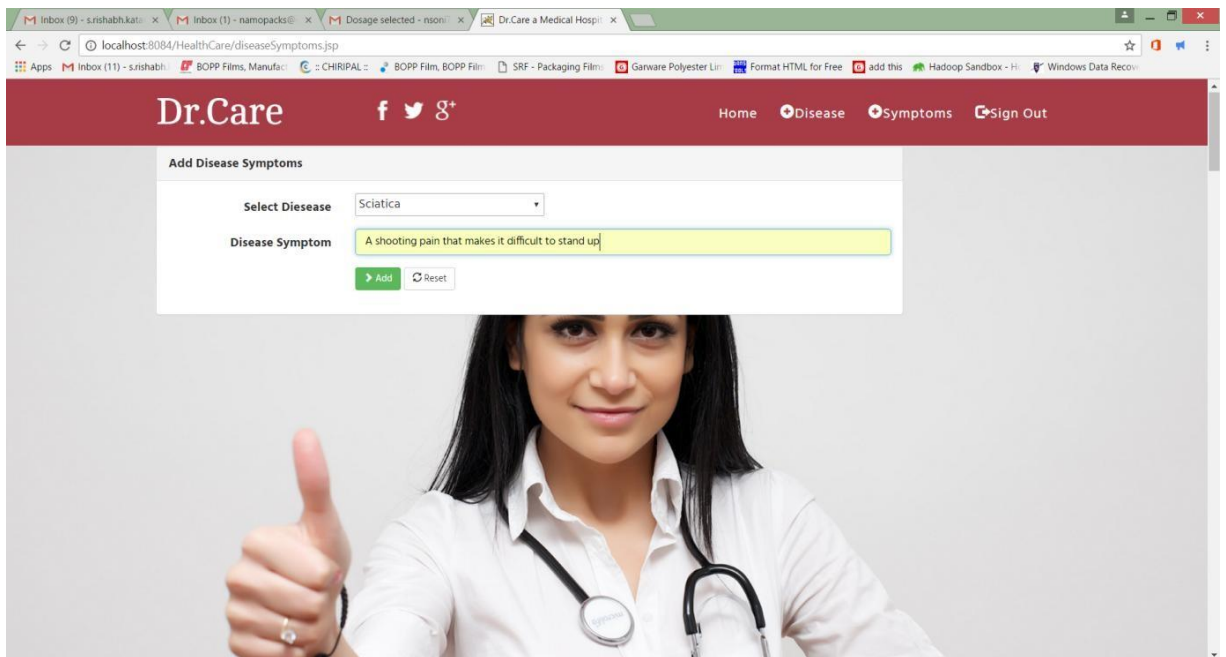
/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
@Override
public String getServletInfo() {
    return "Short description";
} // </editor-fold>

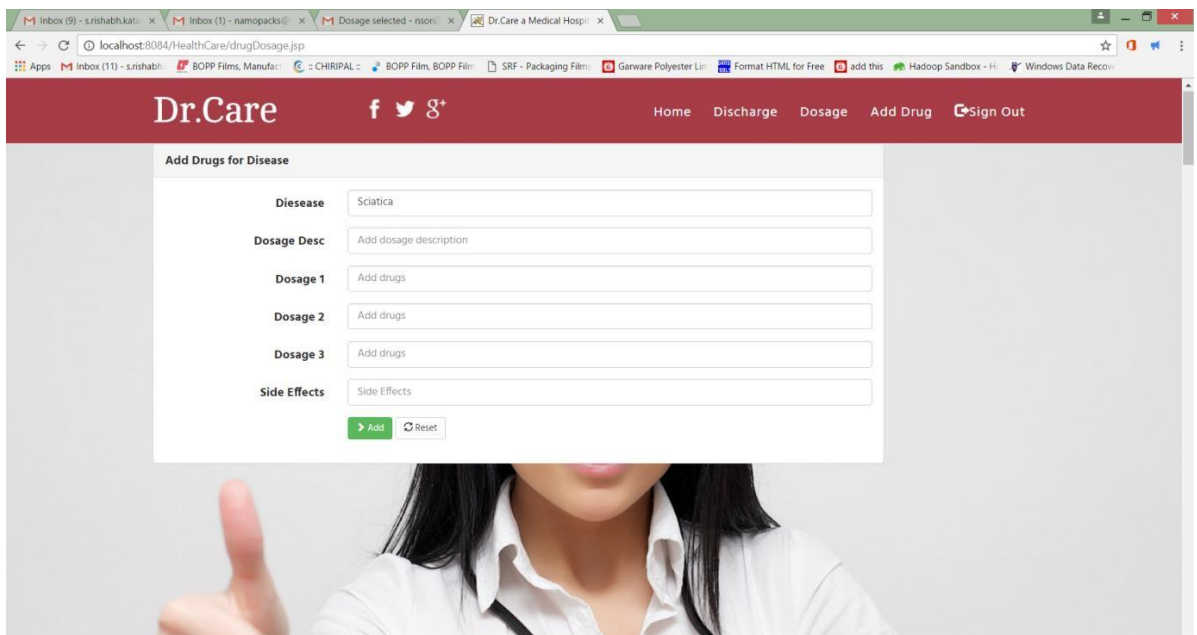
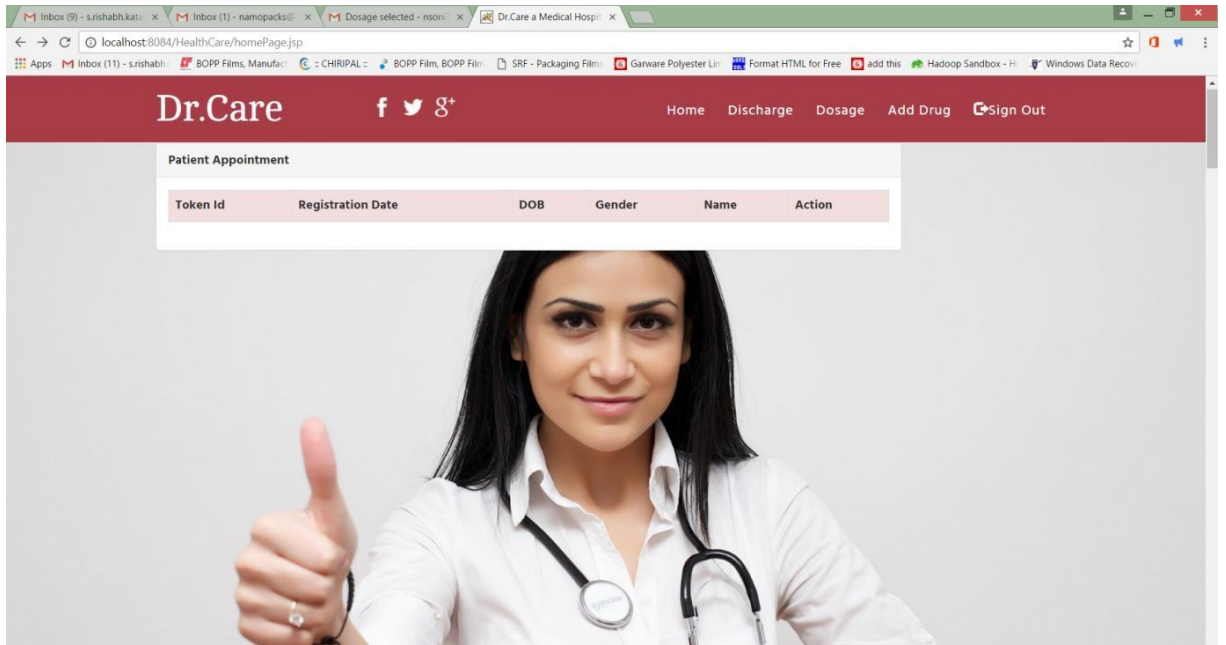
}

```

7.2 SCREEN SHOTS







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 Login Registration About Services Gallery Contact

Patient info gathering

Token Id:- 1489672289318

Date of Registration Thu Mar 16 19:21:29 IST 2017

User Name rajesh

Email nsoni739@gmail.com

Contact Number 1234567890

Gender Male

DOB 10-10-1998

Select the Symptoms

Pain & throbbing sensation in the rear of leg
Hip pain
Pain worsens during sitting or moving the affected limb
A shooting pain that makes it difficult to stand up

Add Reset

Dr.Care f t 8+ 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

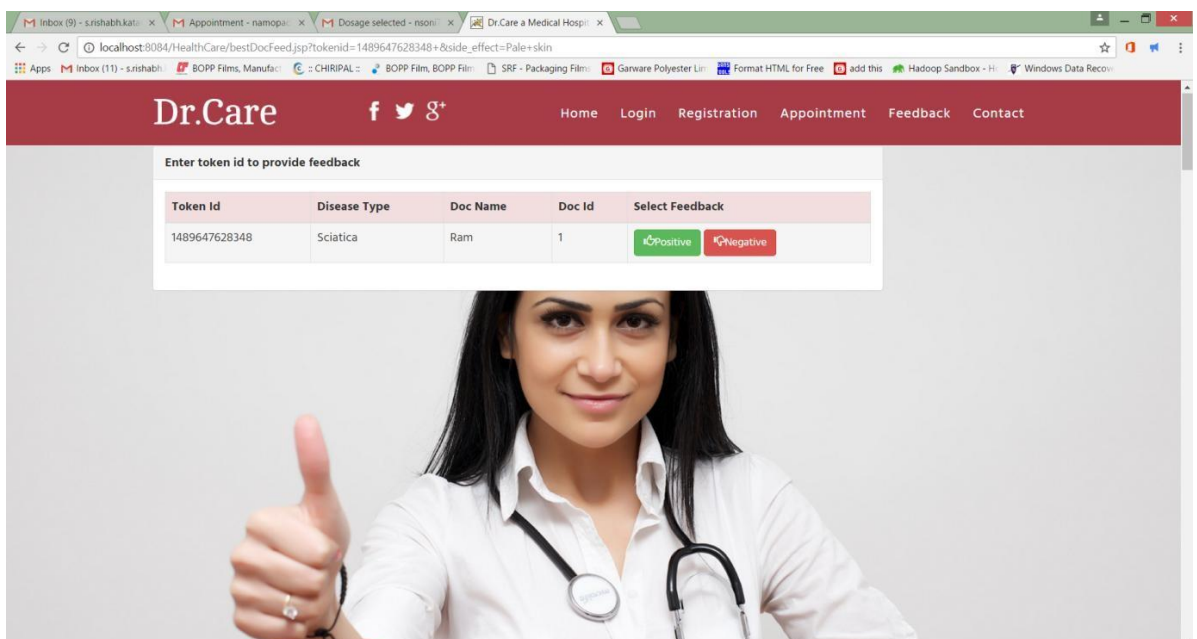
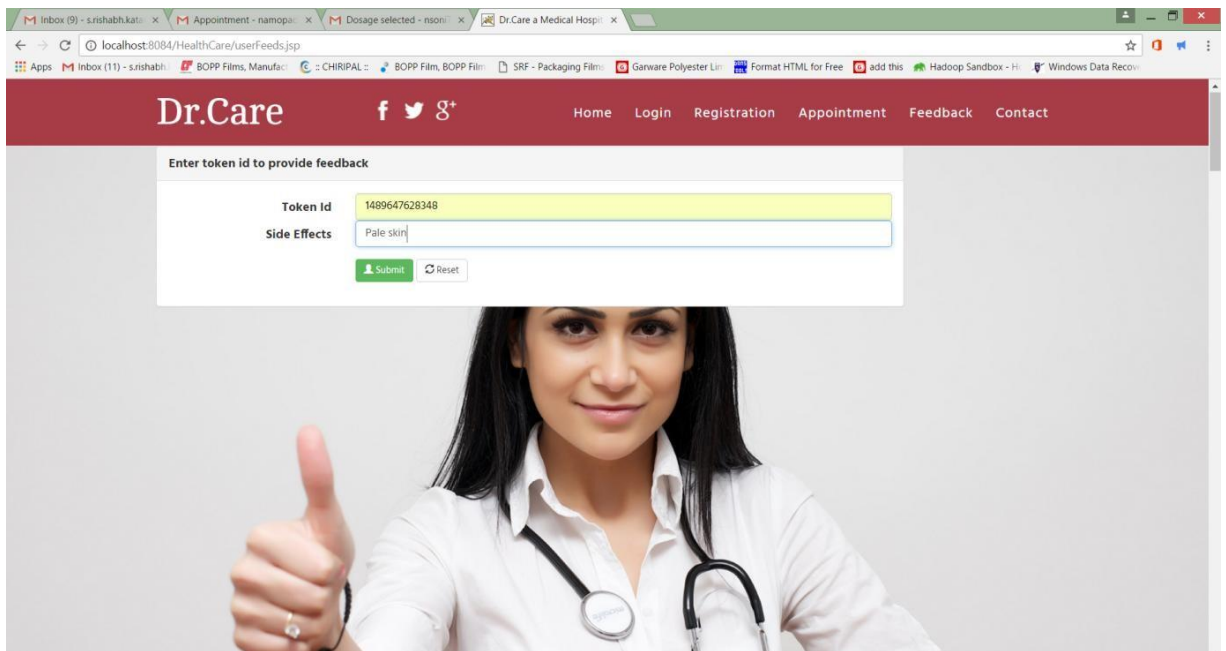


Dr.Care f t 8+ Home Discharge Services Gallery Sign Out

Patient Appointment

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





CHAPTER 8

8.1 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.

8.2 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.

CHAPTER 9

9.1 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.