A

PROJECT REPORT

On

DESIGN AND IMPLEMENTATION OF WEB CRAWLER BASED SEARCH ENGINE

submitted in partial fulfillment of the requirements

for the award of degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

BY

BHAVYA KUMARI (1114310045)

LAVANYA GUPTA (1114310077)

PARISHA AGARWAL (1114310100)

SANCHIT SAXENA (1114310139)

Department Of Computer Science & Engineering

IMS Engineering College, Ghaziabad (U.P.) India

(Affiliated to U.P. Technical University, Lucknow, India)

April, 2015

DESIGN AND IMPLEMENTATION OF WEB CRAWLER BASED SEARCH ENGINE

B.Tech Project Report

Submitted By

BHAVYA KUMARI (1114310045)

LAVANYA GUPTA (1114310077)

PARISHA AGARWAL (1114310100)

SANCHIT SAXENA (1114310139)

Under the Guidance of

Ms. Arti Patle

Assistant Professor, IMS Engineering College

Department Of Computer Science & Engineering

IMS Engineering College, Ghaziabad (U.P.) India

(Affiliated to U.P. Technical University, Lucknow, India)

April, 2015

CERTIFICATE

This is to certify that Ms. Bhavya Kumari, Ms. Lavanya Gupta, Ms. Parisha Agarwal and Mr. Sanchit Saxena of Computer Science department of IMS Engineering College has interacted with me for some technical inputs in regard of his/her project work. I would like to continue in providing such technical inputs for student projects in future as well.

I wish them the very best in all their future endeavors.

Name:

Designation:

Email_id:

Contact No:

Signature:

CERTIFICATE

This is to certify that this project report entitled “Design and Implementation of Web Crawler Based Search Engine” by Bhavya Kumari (Roll No. 1114310045), Lavanya Gupta (Roll No. 1114310077), Parisha Agarwal (Roll No. 1114310100) and Sanchit Saxena (Roll No. 1114310139), submitted in partial fulfillment of the requirements for the degree of Bachelor of Technology in Computer Science & Engineering of IMS Engineering College, Ghaziabad during the academic year 2014-15, is a bonafide record of work carried out under guidance and supervision.

Mrs. Arti Patle
(Mentor)

Mr. Naveen Kumar
(Project Coordinator)

Dr. Pankaj Agarwal
(HOD)

(External Examiner)
Name:
Designation:
Affiliation:

Prof. (Dr.) S.P Pandey
(Director, IMSEC)

ACKNOWLEDGEMENT

We would like to express our deep felt gratitude to our advisor, Prof. Arti Patle, for giving us the opportunity to work on this project and for the valuable advices, encouragement and constant support. It has been both privilege and rewarding experience working with her.

We want to thank Dr. Pankaj Agarwal (HOD) for providing us all facilities and valuable guidance to develop this project.

We want to thank our project coordinator, Mr. Naveen Kumar, for his daily inspection and monitoring our project so that we are able to make a real time user friendly project. We also want to give credit to the faculty members of Computer Science department and the staff for providing us the means to accomplish our project.

Bhavya Kumari	1114310045	4th Year	…………………….
Lavanya Gupta	1114310077	4th Year	...…………………..
Parisha Agarwal	1114310100	4th Year	…………………….
Sanchit Saxena	1114310139	4th Year	…………………….

ABSTRACT

In today’s world, technology has touched every aspect of life but it is sad to hear that millions of people die every year not because of lack of medical facilities but because of lack of knowledge about the available medical facilities. The right to good health is of paramount importance. India has the most inequitable healthcare scenario feasible. On one hand, our country is becoming the hub for medical tourism where people from other countries flock to get good quality medical treatment. On the other hand, most of these facilities seem to be unavailable to the natives. The reason is the poor financial condition of the people. Many government policies and NGOs help such people but knowing about them is still a challenge for the people. One of the ways to make this search easy is by a search system.

A search system is a software system that is designed to search for information on World Wide Web. The project aims at providing a search system which can be used to make searches related to medical fields including detailed information about the diseases, hospitals and NGOs providing cure for the same. The search system consists of various modules like a web crawler, an indexer and a ranker.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter-1

INTRODUCTION

1. Introduction

Health care is a major business in many countries. As has been reported in, 16% of the Gross Domestic product (GDP) of the United States came from the health care sector in year 2004. The statistics for many other countries are similar: 10.9% in Switzerland, 10.7% in Germany, 9.7% in Canada, and 9.5% in France. As the baby boomer generation reaches their retirement age and health care becomes more expensive, the percentage of GDP spent on health care continues to increase. A large part of health care is related to the management and retrieval of medical information. The widespread use of the Web has radically changed the way people acquire medical information.

1.1. Project Overview

The project is a *web crawler based search system* which is used to search the queries related to medical field. Health issues are a major concern in India and generally people could not get proper treatment on time due to lack of knowledge about the symptoms of diseases, lack of knowledge of nearest hospitals and lack of money. Several government policies and NGOs have actively played a major role in providing treatment to those who lack in money. Today is the era of internet and most of the people are connected to the World Wide Web in some way. This search engine is an attempt to provide the solution to all of the above mentioned problems by providing the details of diseases and its symptoms, nearby hospitals and related NGOs on the same platform.

The search system provides results related to the information about the queried disease, hospitals where the treatment of queried disease is available and NGOs which help the people in treatment of that disease. All the three results are viewed on the same platform in a single search. The three types of results are viewed in separate tabs, namely, Home, Hospitals and NGOs. The system also provides the maps service to the user. In maps, user selects the disease and city and a list of hospitals and NGOs is viewed in separate tabs. When the user clicks on any hospital or NGO, it gets pinned on the map.

1.2. Objective

1. **To design and implement a web crawler:** The crawler crawls the web pages related to the queried disease as well as the hospitals and NGOs working for the same.

2. **To design and implement an indexer:** The indexer indexes all the fetched relevant data (keywords) from the web pages. Inverted Indexing is used to index the keywords.

3. **To design and implement a ranker:** The ranker adds a rank to the indexed pages so that most relevant results appear on the top of the search results.

4. **To design and implement auto complete module:** The auto complete module helps the user to type a word quickly by providing a suggestion of similar words.

5. **To provide the map service to hospitals and NGOs:** This service provides maps to reach the nearest hospitals and NGOs. The users can enter the city and the nearest hospitals and NGOs are provided as a result and they can be pinned on the map.

6. **To design a front end application for the user to interact with the system:** The front end contains a search bar and a search button in which the query is entered by the user. The results related to disease, hospitals and NGOs are viewed in separate tabs respectively.

7. **To integrate all the above modules:** The final objective is to integrate all the above modules and implement a searching system.

Chapter-2

LITERATURE REVIEW

2. Literature Review

The project is developed with the help of several proven theories given by researchers all over the world. Several research papers led to the literature study and comparison of various algorithms of crawling and ranking used in this project, and also helped in understanding the basic working and functionalities of the existing systems.

2.1. Web Crawler Strategies

- **Breadth First Search Algorithm**

This algorithm aims in the uniform search across the neighbor nodes. It starts at the root node and searches all the neighbor nodes at the same level. If the objective is reached, then it is reported as success and the search is terminated. If it is not, it proceeds down to the next level sweeping the search across the neighbor nodes at the level and so on until the objective is reached. When all the nodes are searched but the objective is not met, then it is reported as failure.

Breadth first is well suited for situations where the objective is found on the shallower parts in a deeper tree. It will not perform so well when the branches are so many in a game tree especially like chess game and also when all the paths lead to the same objective with the same length of the path [1] [2]. Andy Yoo et al. [3] proposed a distributed BFS for numerous branches Poisson random graphs and achieved high scalability through a set of clever memory and communication optimizations.

- **Depth First Search Algorithm**

This is a powerful technique of systematically traverse through the search by starting at the root node and traverse deeper through the child node. If there is more than one child, then priority is given to the leftmost child and traverse deep until no more child is available. It is backtracked to the next unvisited node and then continues in a similar manner [4].

This algorithm makes sure that all the edges are visited once breadth [5]. It is well suited for search problems, but when the branches are large then this algorithm might end up in an infinite loop [2].

2.2. Ranker Strategies

- **HITS Algorithm**

This algorithm's main technique is Web Structure Mining, Web Content Mining. It computes the hubs and authority of the relevant pages. It ranks at top, the relevant as well as important page as the result. The input parameters are the Content, Backward and Forward links. The relevancy of results is more than PageRank algorithm (since this algorithm uses the hyperlinks so according to Henzinger, 2001 it will give good results and also consider the content of the page). Limitations of the algorithm are Topic drift, efficiency problem and quality of result which is less than PageRank [6].

2.3. Examples of Medical Search Engine

- **Bing Health**

Bing Health (previously *Live Search Health*) is a health-related search service as part of Microsoft's Bing search engine. It is a search engine specifically for health-related information through a variety of trusted and credible sources, including Medstory, Mayo Clinic, and National Institutes of Health's Medline Plus.

Search results in Live Search Health were presented in a three-column layout with health-related articles from the trusted sources in the left, web search results in the middle, and sponsored results on the right. One particular feature for Live Search Health is that all health search queries and responses were encrypted to provide a measure of privacy and security when dealing with health issues.

However, on June 3, 2009, the Live Search Health front-end became fully integrated into Bing search results, accessible only via the "Explorer pane" on the left when the contextual search engine detects a health-related search query entered.

On January 10, 2010, Bing Health search results got an upgrade. Typing in a specific illness will now highlight important information such as related conditions, and common medications to reduce symptoms. In addition reference materials and documentation about the disease and its history can be shown.

- **MedSearch**

MedSearch is a specialized medical Web search engine, to address these challenges. Existing Web search engines often cannot handle medical search well because they do not consider its special requirements. Often a medical information searcher is uncertain about his exact questions and unfamiliar with medical terminology. Therefore, he sometimes prefers to pose long queries, describing his symptoms and situation in plain English, and receive comprehensive, relevant information from search results.

MedSearch uses several key techniques to improve its usability and the quality of search results. First, it accepts queries of extended length and reforms long queries into shorter queries by extracting a subset of important and representative words. This not only significantly increases the query processing speed but also improves the quality of search results. Second, it provides diversified search results. Lastly, it suggests related medical phrases to help the user quickly digest search results and refine the query. We evaluated MedSearch using medical questions posted on medical discussion forums. The results show that MedSearch can handle various medical queries effectively and efficiently [16].

Chapter-3

SYSTEM ANALYSIS

3. System Analysis

The system analysis details about the existing systems that are both present in theory or market, and their drawbacks which led to the proposed system. It also showcases all the details like technical, operational and economic feasibility, too.

3.1. Existing System

There are several online portals and websites helping people these days to find doctors, hospitals and blood banks in their city and nearby localities. Few examples of these are *doctorbabu.com, medindia.net*, etc. They even help the common people calculate diabetes risk assessment, blood sugar conversion online, help them find medical colleges, resources, specialty services, directories, various articles and drugs for specific diseases and pain.

- **MedSearch** is a specialized medical Web search engine, to address these challenges. MedSearch uses several key techniques to improve its usability and the quality of search results. First, it accepts queries of extended length and reforms long queries into shorter queries by extracting a subset of important and representative words. This not only significantly increases the query processing speed but also improves the quality of search results. Second, it provides diversified search results. Lastly, it suggests related medical phrases to help the user quickly digest search results and refine the query. We evaluated MedSearch using medical questions posted on medical discussion forums. The results show that MedSearch can handle various medical queries effectively and efficiently.

- **Bing Health** (previously *Live Search Health*) is a health-related search service as part of Microsoft's Bing search engine. It is a search engine specifically for health-related information through a variety of trusted and credible sources. Search results in Live Search Health were presented in a three-column layout with health-related articles from the trusted sources in the left, web search results in the middle, and sponsored results on the right.

3.1.1. Drawbacks

Information available on portals about drugs online might help people in emergency at times but can even worsen the situation if the drugs are taken without any doctor's consent. Since not all are aware of the salts they are allergic to or know the proper dosage of the medicine. So, probably only the information about the salts of the drugs online might cause a chaos. Other search engines, like Bing Health only provide articles and information about the disease and the symptoms which can be generally found on any general search engine does not prove to be so effective to medical problems. And another engine, MedSearch do not provide best results when it comes to the doctors and hospitals as they are working commercially and ranks the sites in any order they want.

3.2. Proposed System

This project is about a platform that consolidates all the information about the disease, its symptoms, the concerned hospitals and NGOs providing treatment for the same. There is an additional feature of the map, which helps users pin the address of hospitals and NGOs on it to mark the area where they are situated and verify if it actually exists or not.

3.3. Feasibility Study

3.3.1. Economic Feasibility

- **Cost/Benefits Analysis:** The cost/benefits factor is quite low which means that the cost incurred to build the system is low as compared to the benefits this system provides to the user.

3.3.2. Operational Feasibility

- **Reliability:** Any kind of information that is generated in the result is reliable as the admin is responsible to feed in only reliable and trusted websites for crawling.
- **Maintainability:** The system is easy to maintain since only crawled data and the ranked results have to be updated time to time. And any other maintenance of hardware is easy, too.
- **Usability:** The system being user-friendly helps the user search any query on the search engine easily and also provides tabs to display information in a distinguished manner.

- **Affordability:** To use the system, a user only needs an internet connection with a good bandwidth and nothing else which makes it quite affordable.

3.3.3. Technical Feasibility

- **Hardware Feasibility:** The system requires a server, mainly to keep the process of crawling ongoing for several hours as per the need.
- **Software Feasibility:** The software required by the system are a database to index all the crawled data, Eclipse IDE and Apache Tomcat to build the project and few Java tools like JDK, etc. to implement java language required for coding.

Chapter-4

SYSTEM SPECIFICATION

4. System Specification

The system specification consists of hardware and software requirements as well as the minimum requirements used to run this system. It also includes the features of the front end and back end of the system.

4.1. Hardware Requirements

The successful implementation of the project requires:

- A high configuration laptop or personal computer to act as a server for crawling as well as for other modules.
- Internet Connection is required to run the project.

Table 4.1.1: Minimum Hardware Requirements

Hardware	**Specification**
Processor	Intel core 2 duo @ 2GHz
Ram	1 GB
Hard disk	20 GB
Internet Connection	Bandwidth @ 1Mbps

4.2. Software Requirements

The software requirements for the project are as follows:

- **Ubuntu:** Ubuntu is an open source software platform that runs everywhere from the smartphone, the tablet and the PC to the server and the cloud. It is a complete desktop Linux operating system, freely available with both community and professional support.

  The Ubuntu community is built on the ideas enshrined in the Ubuntu Manifesto: that software should be available free of charge, that software tools should be usable by people in their local language and despite any disabilities, and that people should have the freedom to customize and alter their software in whatever way they see fit. "Ubuntu" is an ancient African word, meaning "humanity to others". The Ubuntu distribution brings the spirit of Ubuntu to the software world. Any version of Ubuntu can be used to run this project.

- **Java Runtime Environment:** The Java Runtime Environment (JRE), also known as Java Runtime, is part of the Java Development Kit (JDK), a set of programming tools for developing Java applications. The Java Runtime Environment provides the minimum requirements for executing a Java application; it consists of the Java Virtual Machine (JVM), core classes, and supporting files.

- **Apache Tomcat:** Apache Tomcat is an open-source web server and servlet container developed by the Apache Software Foundation (ASF). Tomcat implements several Java EE specifications including Java Servlet, Java Server Pages (JSP), Java EL, and Web Socket, and provides a "pure Java" HTTP web server environment for Java code to run in. Apache is developed and maintained by an open community of developers under the auspices of the Apache Software Foundation, released under the Apache License 2.0 license, and is open-source software.

- **MYSQL Database Management System:** MySQL is a relational database management system (RDBMS), and ships with no GUI tools to administer MySQL databases or manage data contained within the databases. Users may use the included command line tools, or use MySQL "front-ends", desktop software and web applications that create and manage MySQL databases, build database structures, back up data, inspect status, and work with data records. The official set of MySQL front-end tools, MySQL Workbench is actively developed by Oracle, and is freely available for use.

Table 4.2.1: Minimum Software Requirements

Software	**Specification**
Linux OS	Any distribution. Preferred - Ubuntu 14.04
Apache tomcat server	Apache tomcat 7
MySQL database	MySQL server 5.5
Java Runtime environment	Java 7 Oracle JRE

4.3. Front End

The front end of the project is designed using HTML, CSS, JavaScript, JQuery and Bootstrap. The user can interact with the system through front end. In the front end, three user interfaces are designed namely *main page*, *maps* and *admin panel*. It also includes designing the logo for the project.

1. **User**

   1. **Home.html**

      This is an html file, in which the user finds the link for the Maps service. Once the user clicks on the link, the page is redirected to another html page named maplist.html.

      The *main page* contains a search bar, a search button and a link for maps. As soon as the user clicks in the search bar, it shifts at the top of the page and a menu bar appears which contains three tabs namely home, hospitals and NGOs. Below the three tabs is the area in which the search results appear. The user enters the query in the search bar and clicks the search button. As soon as the query in entered in the search bar, the results start appearing in the targeted area. By default, the user gets the results in home tab in which links related to the disease description appears. To get the links of hospitals and NGOs, user has to click on the respective tabs. At a time, ten search results are viewed on the page. To view further results, user has to click the *next* button. To view the maps, the user has to click on the *maps* link which is present at the top right corner of the main page.

   2. **Maplist.html**

      This html file gives user an interface to select the city and press the go button which enlists the hospitals and NGOs that are stored in the database. The results are displayed in different tabs for hospitals and NGOs respectively. Once the user clicks on any of the hospital or NGO's name, the address is pinned on the map for the user.

      The *maps* page is divided in two parts: one part contains two drop-down menus to select the city and treatment and a *Go* button along with the map canvas and the other part contains the area where nearby hospitals and NGOs are shown. Firstly, the user selects the treatment and city from the drop-down menu and clicks the *Go* button. Then the list of hospitals and NGOs is viewed in the right half of the page. By default, hospitals are viewed which contains the list of hospitals. To view NGOs, the user has to click on the *NGOs* tab. When the user clicks on the name of any hospital on NGO from the list, it is pinned on the map present on the map canvas at the left half of the page. The page also contains a link name *Back to Searching*. When the user clicks on that link, the search page re-appears.

2. Admin

1. AdminHome.html

This is an html file that serves as a homepage for the admin of the maps module. Only admin is authorized to login to this page and feed the addresses of hospitals and NGOs manually. After the admin logs in, this html page is redirected to a Login.java page.

2. AdminPanel.jsp

This is a JSP page in which the admin is welcomed if the login is successful. The admin selects from a drop down menu for which type of location has to be stored, i.e., Hospital's address has to be fed in the database or any NGO's. Then he feeds in the city and complete address and clicks the locate button which calls a function to verify the address by pinning it on the map. If the address is pinned on the map, then a request is sent to the database to store the fed data.

The *admin panel* is a page which is used by the admin of the maps module. The page contains two text boxes: *username* and *password* and a *Login* button. The admin enters the username and password and clicks the login button.

4.3.1. Features

- All the results are viewed in separate tabs.
- The results are provided in well-structured form which makes easy for the user to access the system.
- The user can search the nearby hospitals and NGOs from the maps as well by providing treatment and city as input.
- The hospitals and NGOs are pinned on the map when user clicks it.
- The links of maps and main page are present on the pages which makes it easy for the user to switch between the two pages.
- A large number of results can be viewed by clicking on the *next* button.

4.4. Back End

1. Crawler

At the back end, a main method is called for starting the crawler. It also checks for validation of arguments passed. The arguments passed are the location of seeds.xml file and max depth up to which the crawler would go. It also updates seeds in seeds_table in database and it calls the functions to crawl the pages. Another function checks if the *seeds.xml* exists at the given location or not. The program will proceed only if the file exists. Seeds.xml file is read and then the functions of InjectSeed.java are called to insert those seeds in database.

Database.java contains functions to return the database connections wherever it is needed in the program. *MaintainIndex.java* contains method to check whether a page is fetched in the previous runs of crawler or not. This function also can clear the indexes of any website if needed.

CrawlPages.java is the main file of the program responsible for crawling the pages and sending the content to PreIndexer.java. The class has functions that crawl the pages, adds the fetched pages to database, maintains ranking factors in the database, check that page has been crawled or not etc. This is the main controlling file of the whole module. It also simultaneously sends the crawled pages for indexing purpose. *PreIndexer.java* has functions which fetches the contents from the crawled page according to pre-defined tags. It tokenizes the string and also removes the stop words. Then it sends the remaining content to WordIndexer.java which contains methods that inserts the words with the page number they are present in the database. It is responsible for maintaining all of the word to page mapping table in the database. The searching takes place in this word to page mapping table only.

2. Auto Complete

Auto complete module helps to bring on suggestions as the user types the queries. This module takes every string of letters that the user types, find the words that start from that string, sorts the words in order of decreasing frequency and picks up the top four results and present them as suggestions to the user. The frequency here is maintained as the number of times a particular word has been searched by the users. When a user types his query and clicks on the “Enter” or “Go” button, that particular query is split by ”space” and the resulting words’ frequencies are increased by one in the database.

3. Maps

In the back end of this module, the location which user inputs is updated on the map by the process of geo-location and pinned on it. Another JSP page is used as an interface between the database and the admin, through which the admin can feed in the data like address of hospitals and NGOs specializing in particular treatments. A mapping function is also used to map the hospitals with their specialized treatments that are offered and remove redundancy.

4.4.1. Features

- The main feature of the crawler is that it is fast when the data is on smaller-scale i.e. it crawls ten pages per minute on an average.
- The crawler is memory efficient and also indexes the data to the database quickly.
- The process of feeding the seeds via the XML file is easy.
- The auto complete module displays fast results out of most frequently searched words.
- The map module pins the searched location of the hospitals and NGOs on the map.
- The map module provides an interface to add treatments that are offered by the hospital and create the database with non-redundant data.
- The map module helps update the auto complete file.
- The map module provides a user interface feature for the user to choose the city and treatment from a list of non-redundant options.

Chapter-5
PROJECT DESCRIPTION

5. Project Description

Project description describes all the modules of the system along with the diagrams like DFD, E-R diagram and database design diagram. The input output designs of the system are also included in this.

5.1. Problem Definition

Scenarios related to health have always been a major issue of concern in India since years before the independence. A majority of population has been affected since then due to malnutrition or denial to the access of health care facilities. But as the years passed by, researchers developed new technologies which led to significant advancement in science and medicine. In the current scenario, India is becoming a hub of advanced healthcare facilities with prominent medical institutions and hospitals. NGOs have been an active part in the medical field and government policies have also been in the favour of good health. But, even today we encounter a large number of deaths due to critical diseases like Cancer, Hepatitis, AIDS, Jaundice and many more. The major reason behind this is that people are not aware of their nearest healthcare facilities.

In this project, the main focus is on the development of such a system which provides all-round information to the users related to their query.

5.2. Overview of the Project

The project is a search system which is dedicated to the medical field. It has a friendly Google-like interface, and helps to solve user’s queries related to health problems. The results are separated into three different categories- Informative, Hospitals, and NGO’s with easy navigation between each one of them. The project also supports Map service, which helps users to find the nearest Hospitals and NGO’s working for the cause.

5.3. Module Description

There are 3 main modules in the project- crawler, autocomplete and maps.

5.3.1. Modules

1. Crawler Module

a) StartRun.java

This file contains main method for starting the crawler. Its checks for validation of arguments passed. The arguments passed are the location of seeds.xml file and max depth up to which the crawler would go. It also updates seeds in seeds_table in database and it calls the functions to crawl the pages.

b) LocationCheck.java

This file contains method for checking that seeds.xml exists at the given location or not. The program will proceed only if the file exists.

c) UpdateSeeds.java

This file contains functions to read the seeds.xml file and calls the functions of InjectSeed.java to insert those seeds in database.

d) InjectSeed.java

This file contains methods that perform the function of inserting the seeds from seeds.xml file to database. This also performs the functions to clear the outdated seeds from the database.

e) Database.java

This file contains functions to return the database connections wherever it is needed in the program.

f) MaintainIndex.java

This file contains method to checked weather a page is fetched in the previous runs of crawler or not. This function also can clear the indexes of any website if needed.

g) Crawlpages.java

This is the main file of the program responsible for crawling the pages and sending the content to PreIndexer.java. The class has functions that crawl the pages, adds the fetched pages to database, maintains ranking factors in the database, check that page has been crawled or not etc. This is the main controlling file of the whole module. It also simultaneously sends the crawled pages for indexing purpose.

h) PageInfoBean.java

This is a java bean responsible for holding three variables for a particular page, the page's readable status, number of hyperlinks in a page, and an array of hyperlinks in a page. The object of this class is needed in the internal working of the crawling module.

i) PreIndexer.java

This file has functions which fetches the contents from the crawled page according to pre-defined tags. It tokenizes the string and also removes the stop words. Then it sends the remaining content to WordIndexer.java.

j) ProcessStopWords.java

This file contains a method that is responsible for removing the stop words form the fetched Strings. The stop words are like a, an, the etc.

k) WordIndexer.java

This file contains methods that inserts the words with the page number they are present in the database. It is responsible for maintaining the all of word to page mapping table in the database. The searching takes place in this word to page mapping table only.

2. Auto Complete Module

a) XMLWriter.java

This is a java class that maintains a connection to the database and retrieves the words with higher frequencies from database and writes them in an XML file “transfer.xml”.

b) ReadXML.java

This is a java class that reads the XML file “transfer.xml” and returns a Document object “doc”.

c) Autocomplete.js

This is a JavaScript file that picks the data from “home.html” page and sends the data to two different servlets- “UpdateFrequency.java” and “Autocomplete.java”.

d) Autocomplete.java

This is a servlet which receives each string of data typed by the user through an Ajax request. It further sends the data to a java class “Matcher.java”. It is also the main servlet that returns the suggestions to the “home.html” page.

e) <u>UpdateFrequency.java</u>

This is a servlet which receives each word typed by the user, through an Ajax request. It increases the frequency of each received word in the database.

f) <u>Matcher.java</u>

This is a java class that receives the string of data from “Autocomplete.java” and searches the “doc” for words that match the last word of the received string. Then, it sorts the results based on the frequency and returns top 4 frequently used words to “Autocomplete.java”.

3. Map Module

I. User

a) <u>Home.html</u>

This is an html file, in which the user finds the map link on the homepage of the spider search engine. Once the user clicks the link, the page is redirected to another html page named maplist.html.

b) <u>Maplist.html</u>

This html file gives user an interface to select the city and press the go button which enlists the hospitals and NGOs that are stored in our database. The results are displayed in different tabs for hospitals and NGOs respectively. Once the user clicks on any of the hospital or NGO's name, the address is pinned on the map for the user.

Maplist.html has following functions

i. *function init()*
A function that initializes the map by a city with the help of the latitude and longitude values (geocoding) predefined by the admin, i.e., Ghaziabad here.

ii. *function updateMap()*
It is a function that updates the map by pinning the address of any hospital or NGO that the user wants to see on the map. And also calls another function named ShowLocation().

iii. *function showLocation()*
This function pins the address on the map as well as helps the user zoom in and convert the view type of map from terrain view to maybe a satellite one.

iv. *Back to Searching*

This is another link on the maplist.html page which directs the user back to the homepage of the spider search engine.

II. Admin

a) <u>AdminHome.html</u>

It is an html file that serves as a homepage for the admin of the maps module. Only admin is authorized to login to this page and feed the addresses of hospitals and NGOs manually. After the admin logs in, this html page is redirected to a Login.java page.

b) <u>Login.java</u>

It is a servlet in which the login of the user is checked by validate.java class which welcomes the user if login is successful otherwise returns to the AdminHome.html page.

c) <u>AdminPanel.jsp</u>

A JSP page in which the admin is welcomed if the login is successful. The admin selects from a drop down menu for which type of location has to be stored, i.e., Hospital's address has to be fed in the database or any Ngo's. Then he feeds in the city and complete address and clicks the locate button which calls a function to verify the address by pinning it on the map. If the address is pinned on the map, then a request is sent to the database to store the fed data. AdminPanel.jsp has following buttons:

a. *Update Auto Complete*

This button calls a JavaScript function updateAutocomplete() which makes an Ajax call to XMLWriter.java. This servlet updates the transfer.xml file.

b. *Get Frequency*

This button calls a JavaScript function getFrequency() which makes an Ajax call to GetFerquency.java. This servlet returns the sum of frequencies of the words and also tells weather the updation of auto complete xml file is recommended or not.

c. *Clear Frequency*

This button calls a JavaScript function clearFrequency() which makes an Ajax call to ClearFrequency.java. This servlet sets the frequencies of all the words in the database to one. This is done so that the frequencies do not go to very high integer values.

5.4. Data Flow Diagram

1. Crawler and Indexer

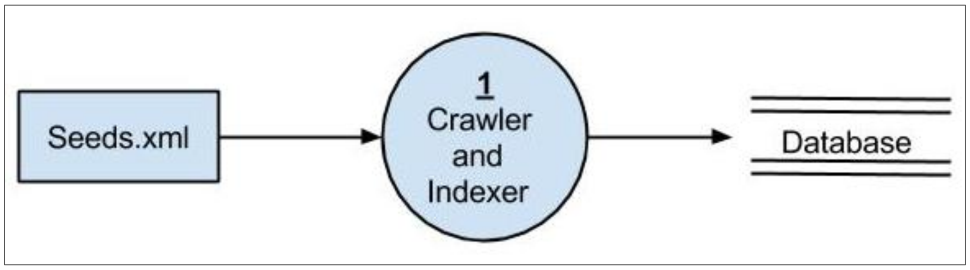


Fig.5.4.1: 0 Level DFD for Crawler and Indexer

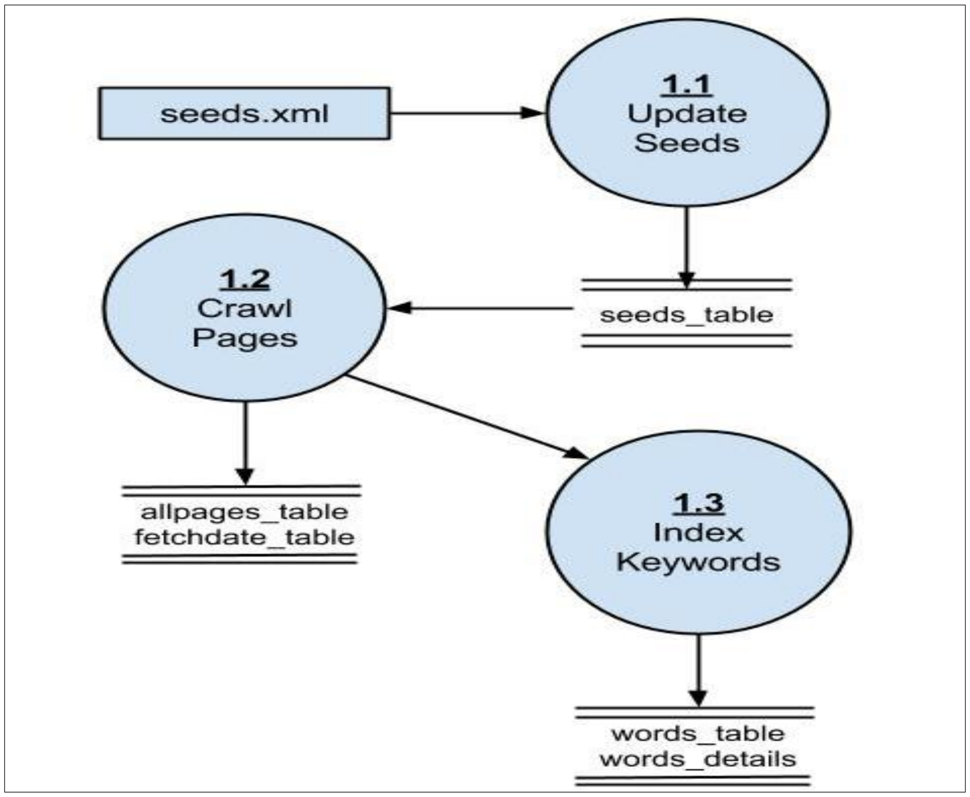


Fig. 5.4.2: 1 Level DFD for crawler and Indexer

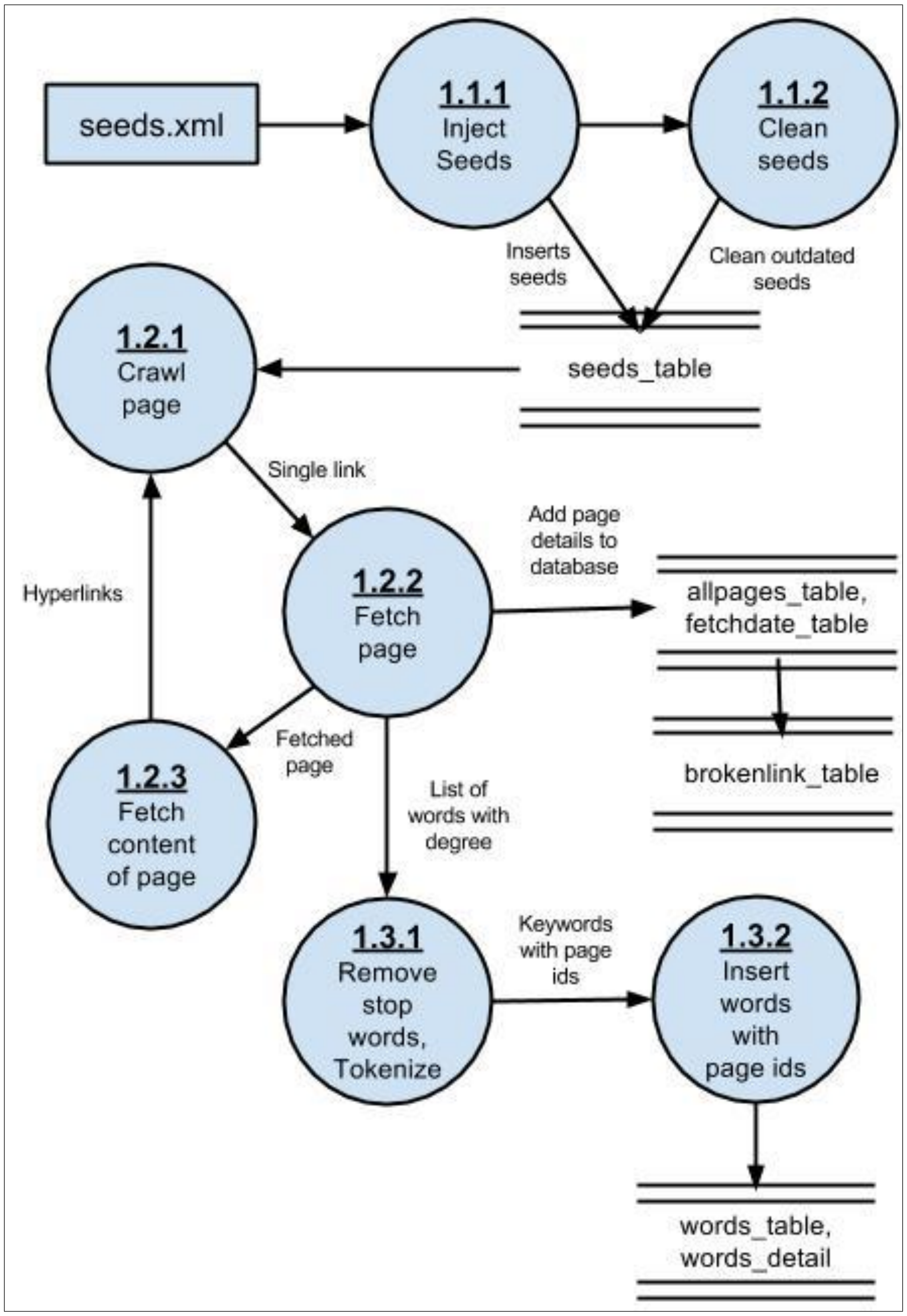

seeds.xml
1.1.1 Inject Seeds
1.1.2 Clean seeds
Inserts seeds
Clean outdated seeds
seeds_table
1.2.1 Crawl page
Single link
Hyperlinks
1.2.2 Fetch page
Add page details to database
allpages_table, fetchdate_table
brokenlink_table
Fetched page
1.2.3 Fetch content of page
List of words with degree
1.3.1 Remove stop words, Tokenize
Keywords with page ids
1.3.2 Insert words with page ids
words_table, words_detail

Fig. 5.4.3: 2 Level DFD for Crawler and Indexer

2. Searcher and Ranker

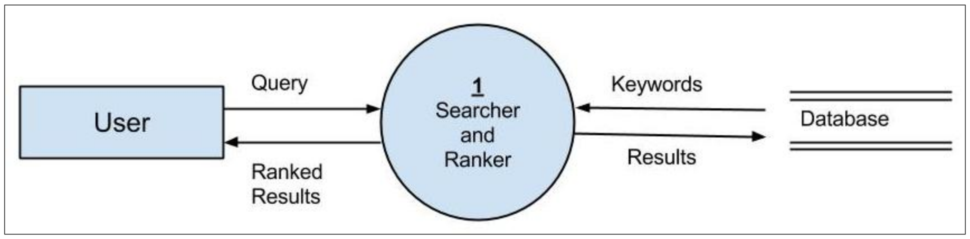


Fig. 5.4.4: 0 Level DFD for Searcher and Ranker

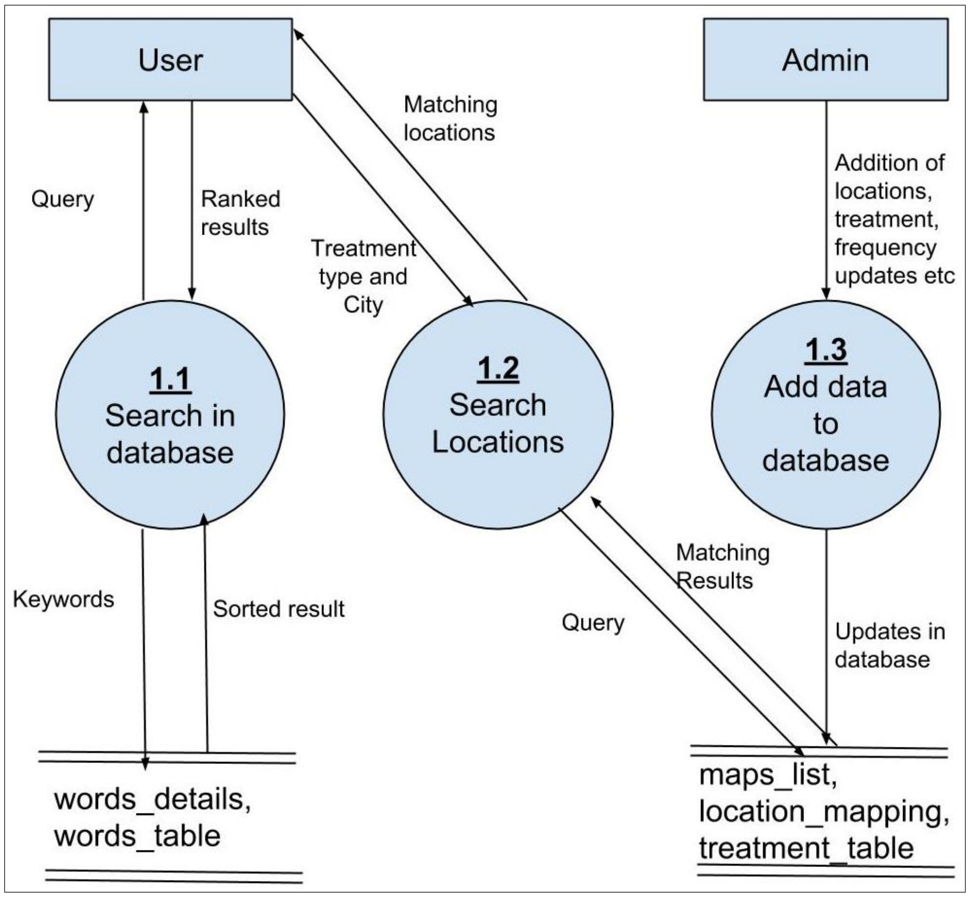


Fig. 5.4.5: 1 Level DFD for Searcher and Ranker

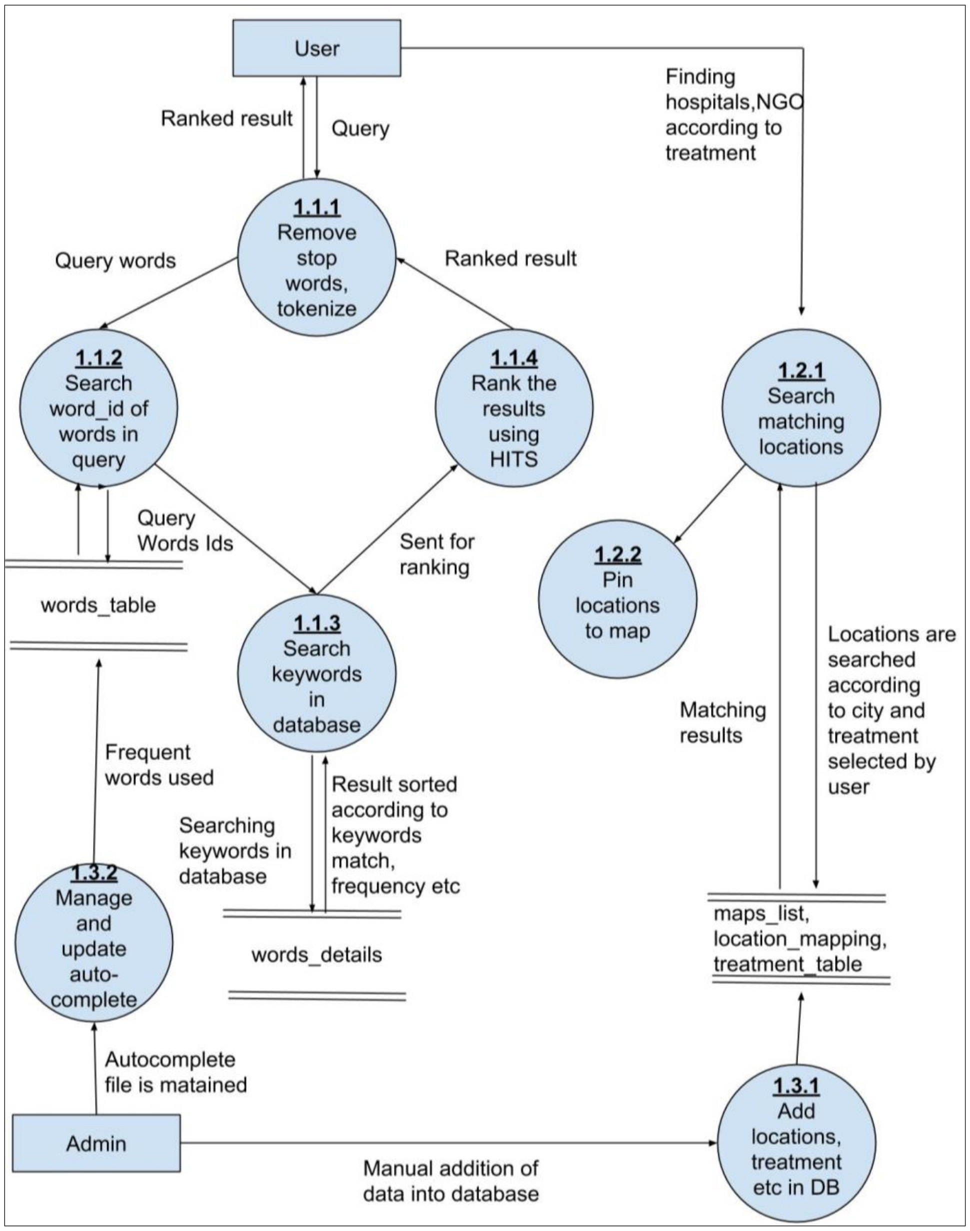

User
Ranked result
Query
Finding hospitals,NGO according to treatment
1.1.1 Remove stop words, tokenize
Query words
Ranked result
1.1.2 Search word_id of words in query
1.1.4 Rank the results using HITS
1.2.1 Search matching locations
Query Words Ids
Sent for ranking
words_table
1.2.2 Pin locations to map
1.1.3 Search keywords in database
Locations are searched according to city and treatment selected by user
Matching results
Frequent words used
Result sorted according to keywords match, frequency etc
Searching keywords in database
1.3.2 Manage and update auto-complete
words_details
maps_list, location_mapping, treatment_table
Autocomplete file is matained
1.3.1 Add locations, treatment etc in DB
Admin
Manual addition of data into database

Fig. 5.4.6: 2 Level DFD for Searcher and Ranker

5.5. E-R Diagram

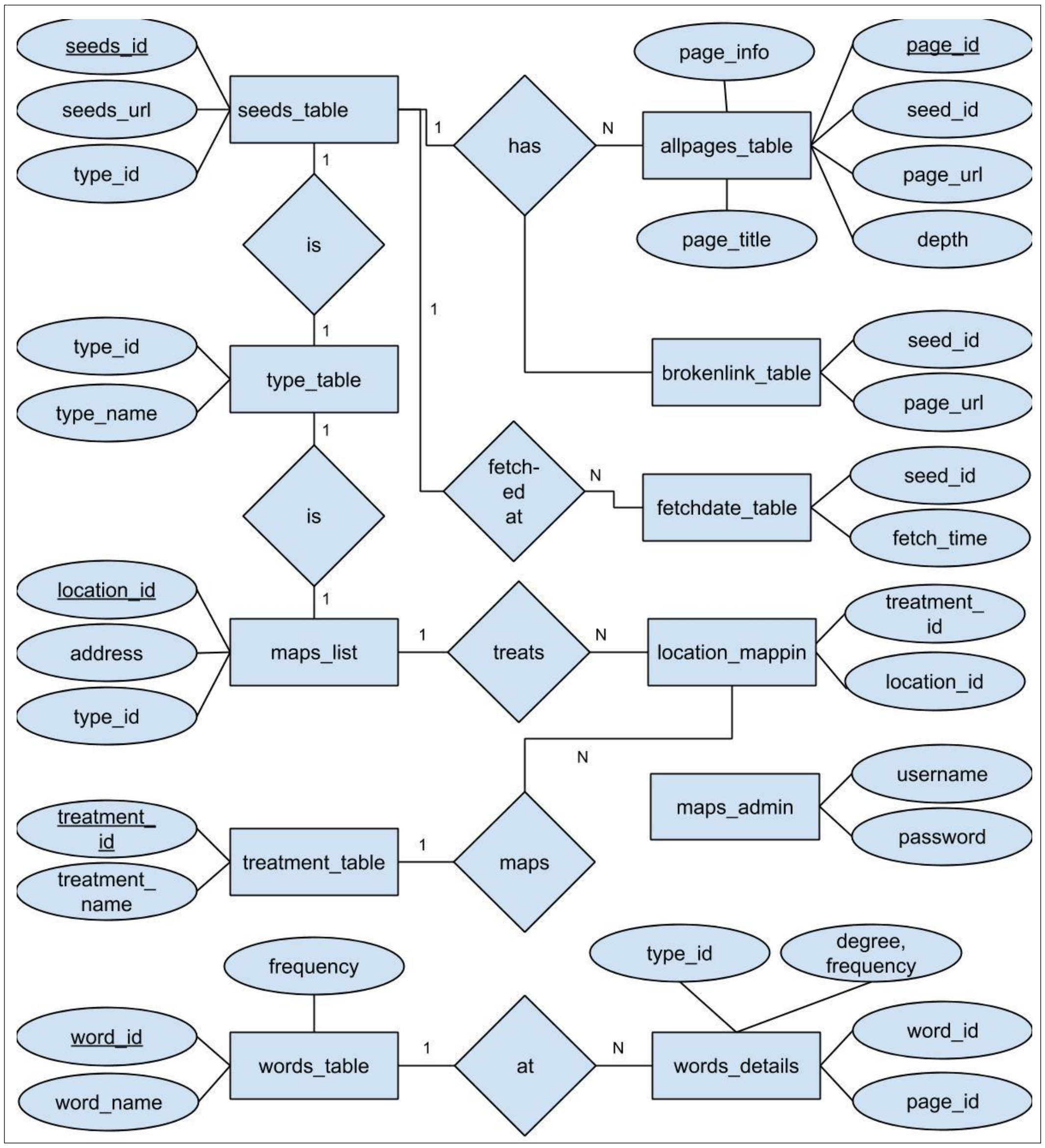


Fig. 5.5.1: E-R Diagram

5.6. Database Design

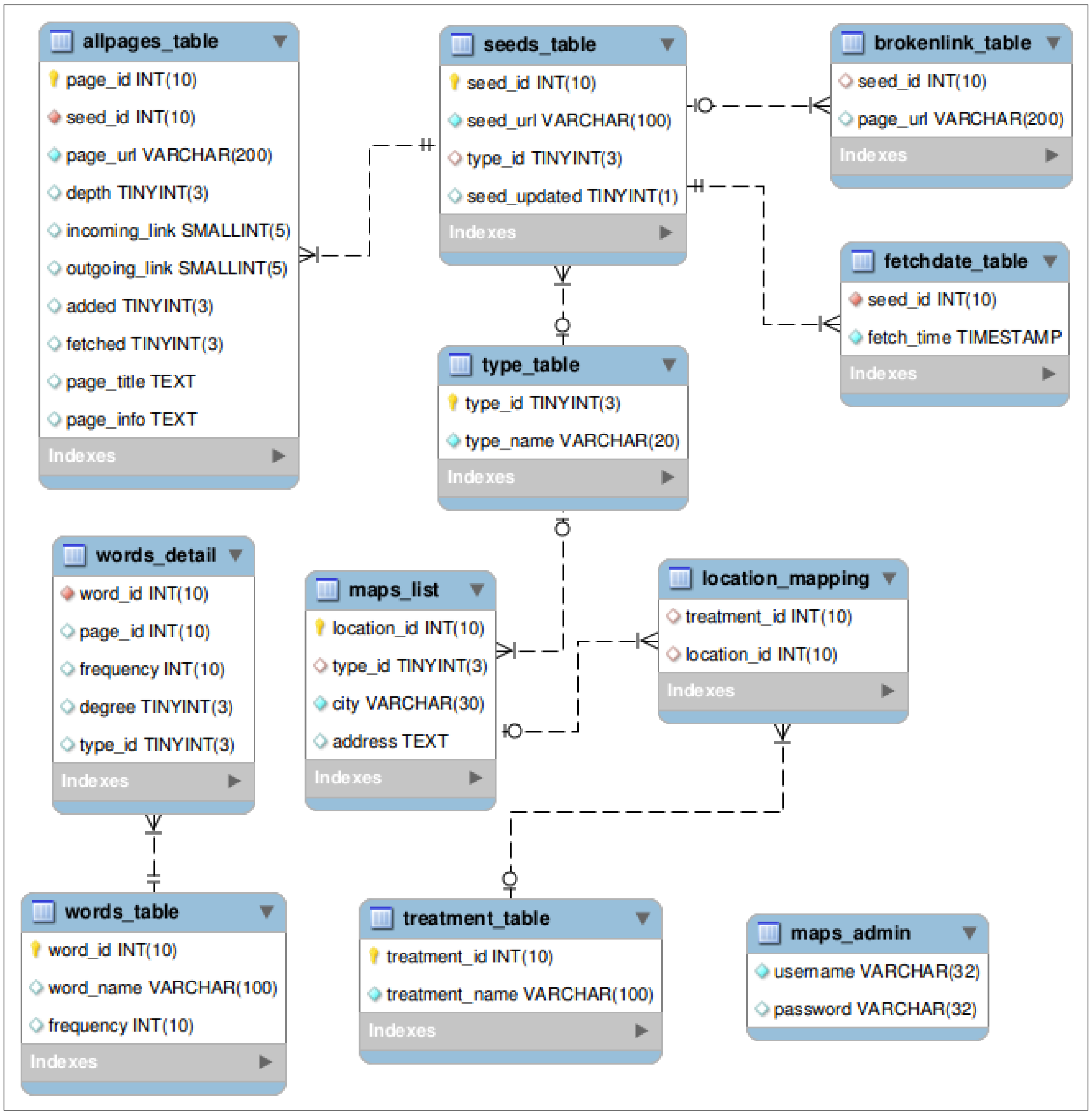
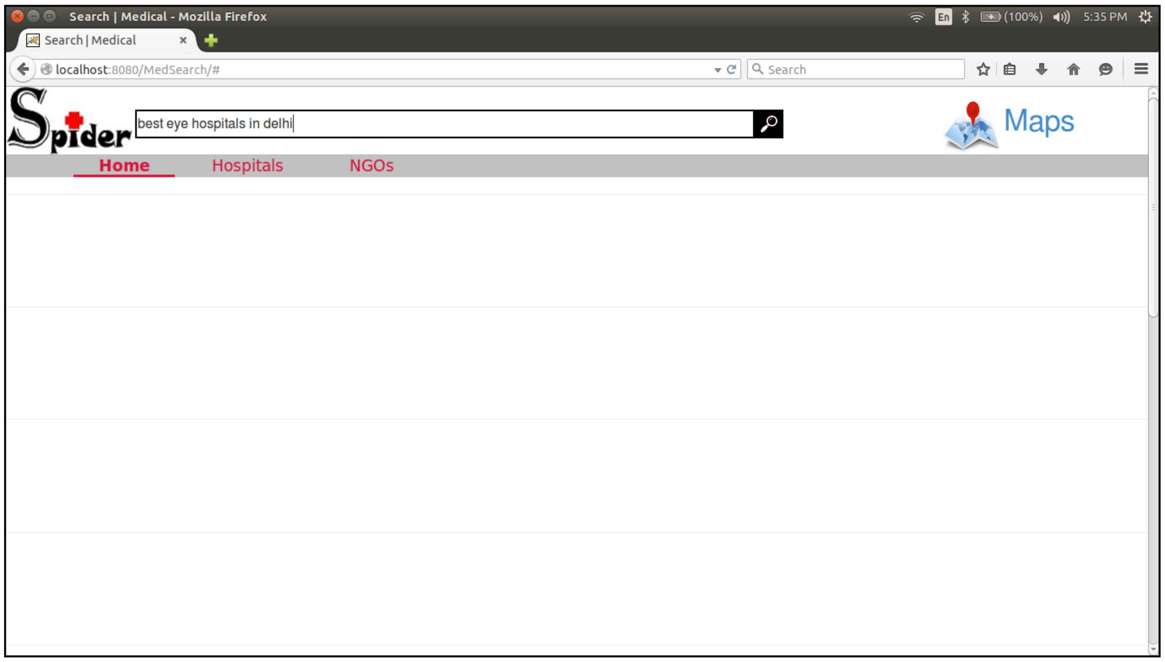


Fig. 5.6.1: Database Diagram

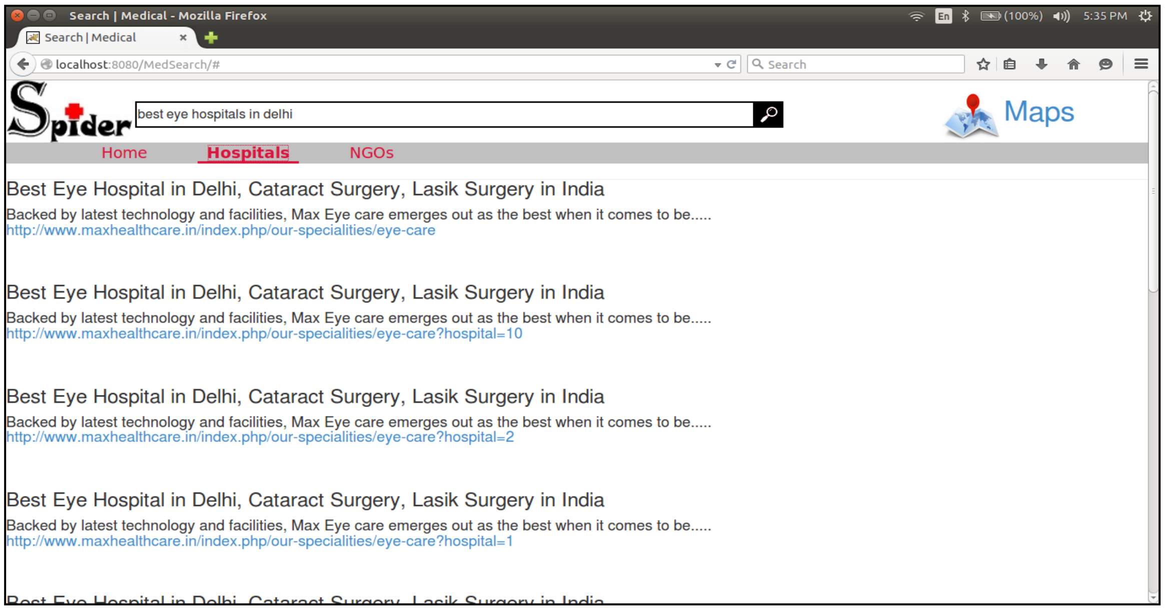
5.7. Input Design


Search | Medical - Mozilla Firefox
Search | Medical
localhost:8080/MedSearch/#
Search
Spider
Maps
Home
Hospitals
NGOs

best eye hospitals in delhi

Fig. 5.7.1: Entering input in the search bar

5.8. Output Design


Search | Medical - Mozilla Firefox
Search | Medical
localhost:8080/MedSearch/#
Search
Spider
Maps
Home
Hospitals
NGOs

best eye hospitals in delhi

Best Eye Hospital in Delhi, Cataract Surgery, Lasik Surgery in India
Backed by latest technology and facilities, Max Eye care emerges out as the best when it comes to be.....
http://www.maxhealthcare.in/index.php/our-specialities/eye-care

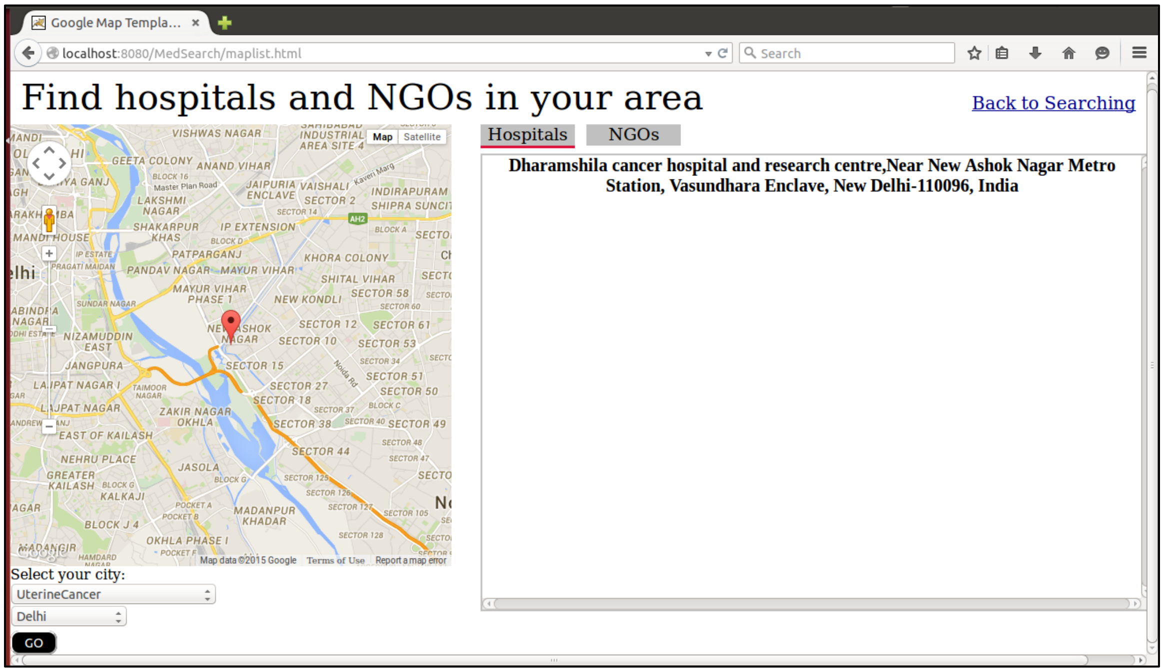
Best Eye Hospital in Delhi, Cataract Surgery, Lasik Surgery in India
Backed by latest technology and facilities, Max Eye care emerges out as the best when it comes to be.....
http://www.maxhealthcare.in/index.php/our-specialities/eye-care?hospital=10

Best Eye Hospital in Delhi, Cataract Surgery, Lasik Surgery in India
Backed by latest technology and facilities, Max Eye care emerges out as the best when it comes to be.....
http://www.maxhealthcare.in/index.php/our-specialities/eye-care?hospital=2

Best Eye Hospital in Delhi, Cataract Surgery, Lasik Surgery in India
Backed by latest technology and facilities, Max Eye care emerges out as the best when it comes to be.....
http://www.maxhealthcare.in/index.php/our-specialities/eye-care?hospital=1

Best Eye Hospital in Delhi, Cataract Surgery, Lasik Surgery in India

Fig. 5.8.1: Search results



Find hospitals and NGOs in your area

Back to Searching

Dharamshila cancer hospital and research centre,Near New Ashok Nagar Metro Station, Vasundhara Enclave, New Delhi-110096, India

Select your city:

UterineCancer

Delhi

Fig. 5.8.2: Search results of maps

Chapter-6

SYSTEM TESTING

6. System Testing

Different types of testing like unit testing, acceptance testing are performed with the help of suitable test cases in this module.

6.1. Unit Testing

In this phase, each module and its sub-modules has been tested independently.

- The Crawler module is able to successfully crawl the websites from the web and index it accordingly.
- The AutoComplete module can successfully read the indexed words from the database and sort the words based on frequency.
- The Map module successfully pins the requested landmarks on the map based on the location selected.

6.2. Acceptance Testing

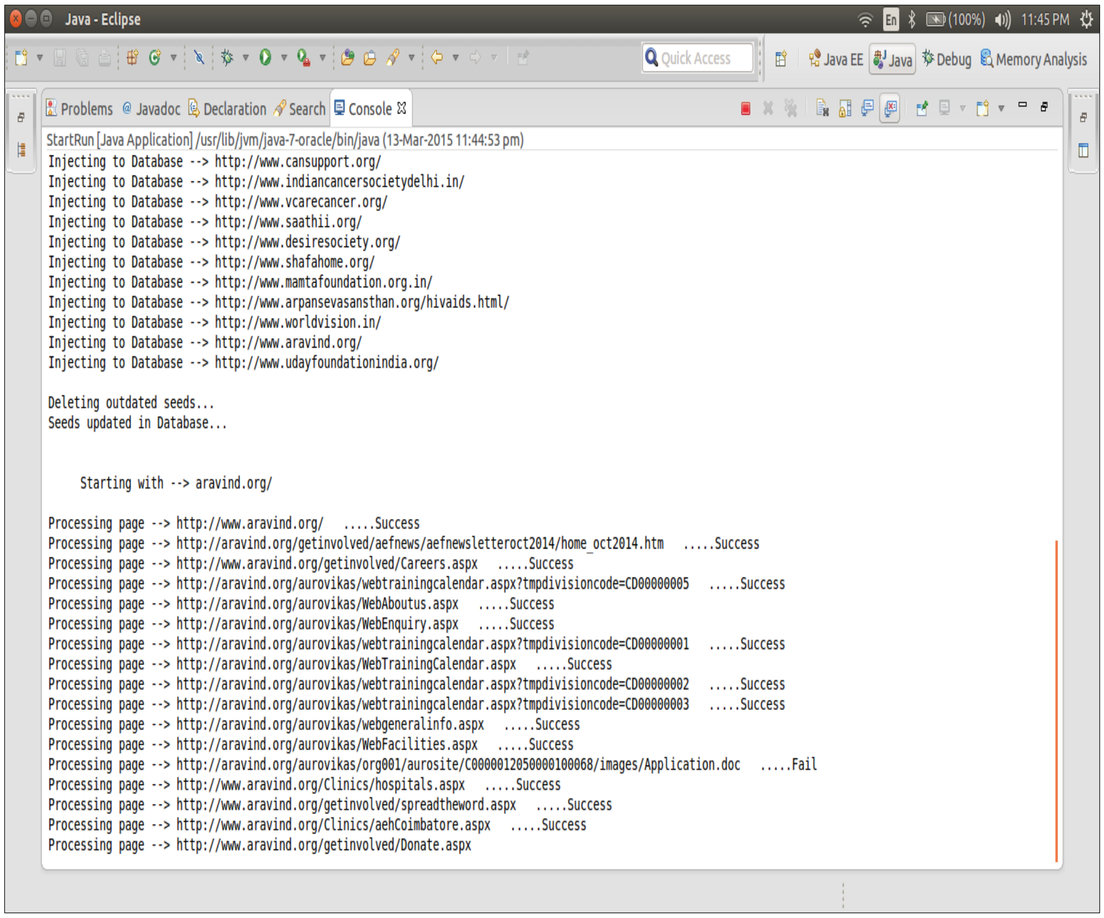
The entire project has been tested by some random users from their terminal.

- The user-interface has been thoroughly checked for its ease of use and navigation.
- The users have validated how the project gathers and presents results based on the search query.
- The presentation of the results in the three categories has also been verified.
- The results shown to the user have been validated for their decreasing order of relevance.
- The suggestions provided by the Auto complete module have also been validated.
- The Map module has been validated for its accuracy to pin-point correct landmarks, and also their relation to the disease queried.
- The behaviour of the system in case of no-input has also been tested.
- The behaviour of the system in the case of non-relevant query has been tested.

6.3 Test Cases

Test Case 1:

Input: URLs in seeds.xml.
Expected Output: Crawling starts, with successfully crawled sites indicating Success, else fail.
Actual Output: Crawling starts, and successfully crawled sites indicate Success, else fail.

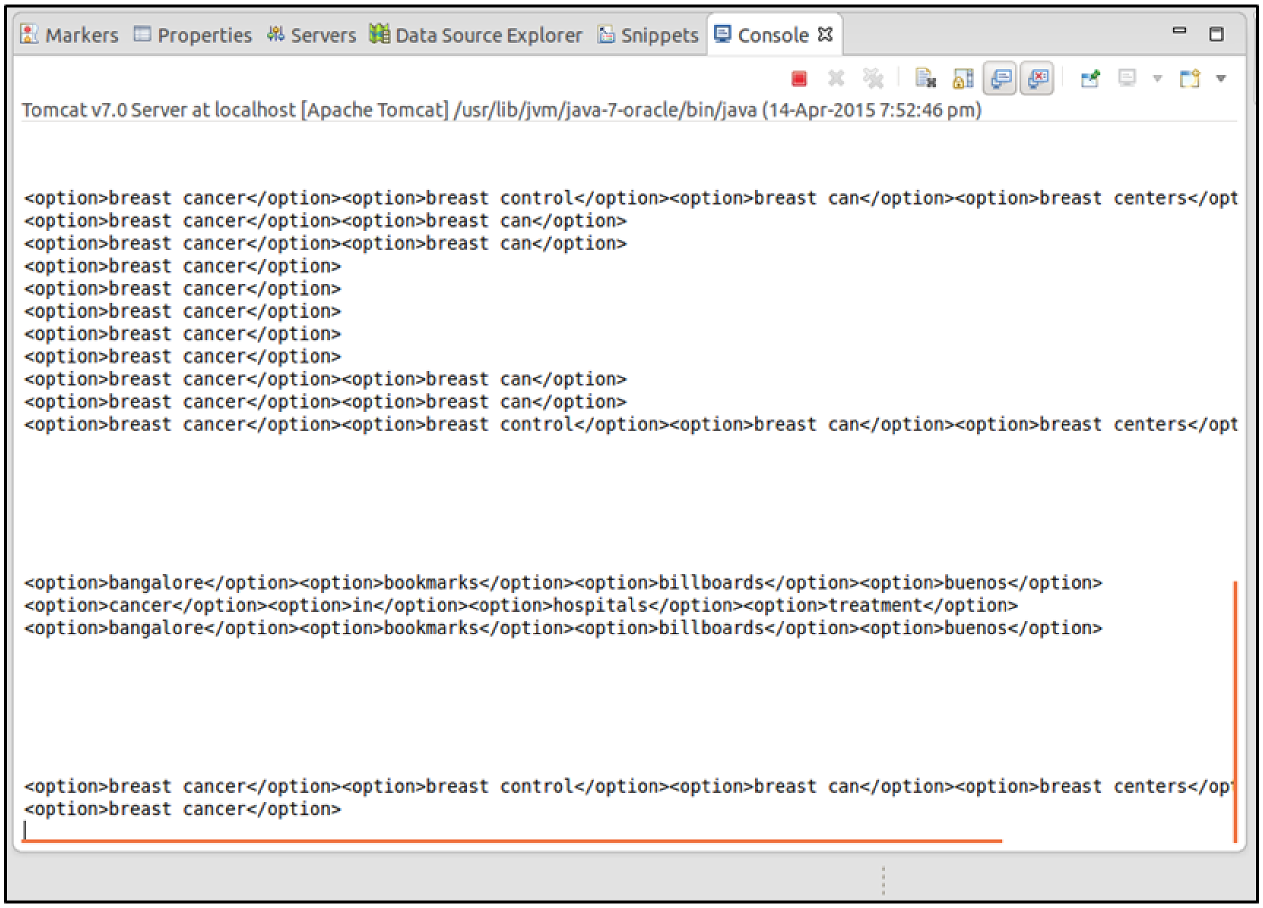


```
StartRun [Java Application] /usr/lib/jvm/java-7-oracle/bin/java (13-Mar-2015 11:44:53 pm)
Injecting to Database --> http://www.cansupport.org/
Injecting to Database --> http://www.indiancancersocietydelhi.in/
Injecting to Database --> http://www.vcarecancer.org/
Injecting to Database --> http://www.saathii.org/
Injecting to Database --> http://www.desiresociety.org/
Injecting to Database --> http://www.shafahome.org/
Injecting to Database --> http://www.mamtafoundation.org.in/
Injecting to Database --> http://www.arpansevasansthan.org/hivaids.html/
Injecting to Database --> http://www.worldvision.in/
Injecting to Database --> http://www.aravind.org/
Injecting to Database --> http://www.udayfoundationindia.org/

Deleting outdated seeds...
Seeds updated in Database...


    Starting with --> aravind.org/

Processing page --> http://www.aravind.org/   .....Success
Processing page --> http://aravind.org/getinvolved/aefnews/aefnewsletteroct2014/home_oct2014.htm   .....Success
Processing page --> http://www.aravind.org/getinvolved/Careers.aspx   .....Success
Processing page --> http://aravind.org/aurovikas/webtrainingcalendar.aspx?tmpdivisioncode=CD00000005   .....Success
Processing page --> http://aravind.org/aurovikas/WebAboutus.aspx   .....Success
Processing page --> http://aravind.org/aurovikas/WebEnquiry.aspx   .....Success
Processing page --> http://aravind.org/aurovikas/webtrainingcalendar.aspx?tmpdivisioncode=CD00000001   .....Success
Processing page --> http://aravind.org/aurovikas/WebTrainingCalendar.aspx   .....Success
Processing page --> http://aravind.org/aurovikas/webtrainingcalendar.aspx?tmpdivisioncode=CD00000002   .....Success
Processing page --> http://aravind.org/aurovikas/webtrainingcalendar.aspx?tmpdivisioncode=CD00000003   .....Success
Processing page --> http://aravind.org/aurovikas/webgeneralinfo.aspx   .....Success
Processing page --> http://aravind.org/aurovikas/WebFacilities.aspx   .....Success
Processing page --> http://aravind.org/aurovikas/org001/aurosite/C0000012050000100068/images/Application.doc   .....Fail
Processing page --> http://www.aravind.org/Clinics/hospitals.aspx   .....Success
Processing page --> http://www.aravind.org/getinvolved/spreadtheword.aspx   .....Success
Processing page --> http://www.aravind.org/Clinics/aehCoimbatore.aspx   .....Success
Processing page --> http://www.aravind.org/getinvolved/Donate.aspx
```

Fig.6.3.1: Crawler

Test Case 2:

Input: Breast c
Expected Output: Words with “Breast c” as the prefix in sorted order.
Actual Output: Words with “Breast c” as the prefix in sorted order.



```
Tomcat v7.0 Server at localhost [Apache Tomcat] /usr/lib/jvm/java-7-oracle/bin/java (14-Apr-2015 7:52:46 pm)


<option>breast cancer</option><option>breast control</option><option>breast can</option><option>breast centers</opt
<option>breast cancer</option><option>breast can</option>
<option>breast cancer</option><option>breast can</option>
<option>breast cancer</option>
<option>breast cancer</option>
<option>breast cancer</option>
<option>breast cancer</option>
<option>breast cancer</option>
<option>breast cancer</option><option>breast can</option>
<option>breast cancer</option><option>breast can</option>
<option>breast cancer</option><option>breast control</option><option>breast can</option><option>breast centers</opt






<option>bangalore</option><option>bookmarks</option><option>billboards</option><option>buenos</option>
<option>cancer</option><option>in</option><option>hospitals</option><option>treatment</option>
<option>bangalore</option><option>bookmarks</option><option>billboards</option><option>buenos</option>






<option>breast cancer</option><option>breast control</option><option>breast can</option><option>breast centers</op
<option>breast cancer</option>
```

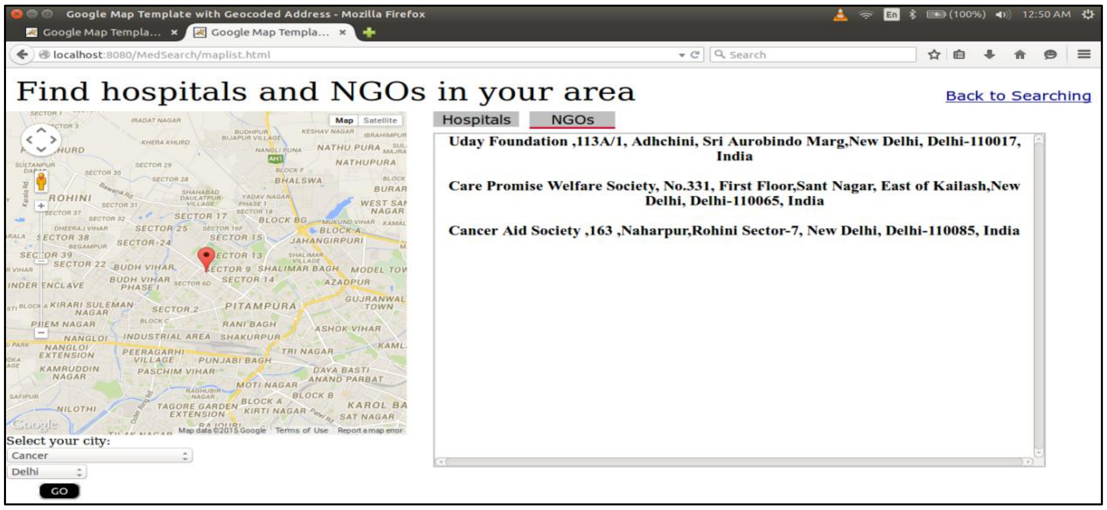
Fig. 6.3.2: Auto Complete

Test Case 3:

Input: City, treatment

Expected Output: List of Hospitals and NGOs, and on clicking on a result, it shows its location on the map.

Actual Output: List of Hospitals and NGOs, and on clicking on a result, it shows its location on map.



Find hospitals and NGOs in your area

Back to Searching

Uday Foundation ,113A/1, Adhchini, Sri Aurobindo Marg,New Delhi, Delhi-110017, India

Care Promise Welfare Society, No.331, First Floor,Sant Nagar, East of Kailash,New Delhi, Delhi-110065, India

Cancer Aid Society ,163 ,Naharpur,Rohini Sector-7, New Delhi, Delhi-110085, India

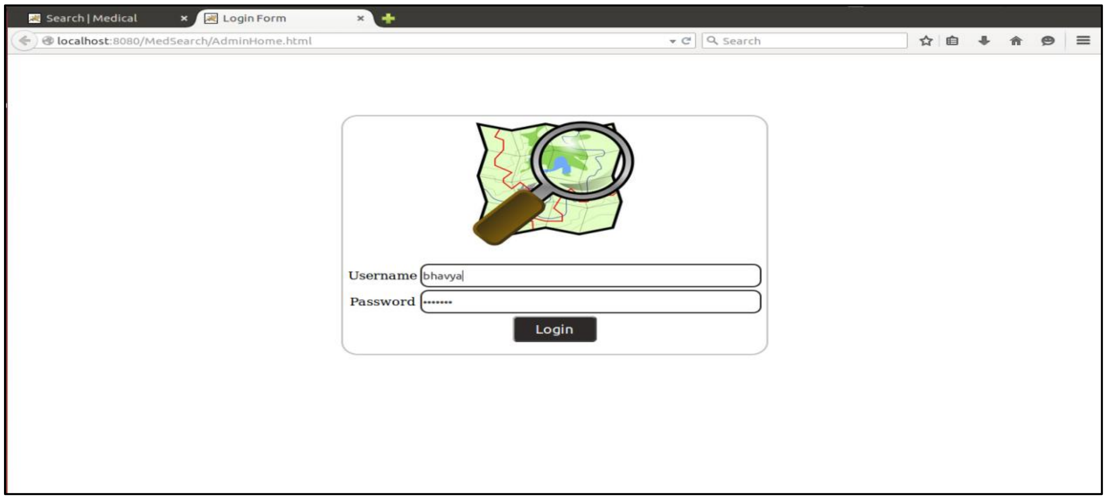
Fig. 6.3.3: Pinning location on map by clicking on its name

Test Case 4:

Input: Correct Username and Password

Expected Output: Google Map template with Geocoded address page.

Actual Output: Google Map template with Geocoded address page.



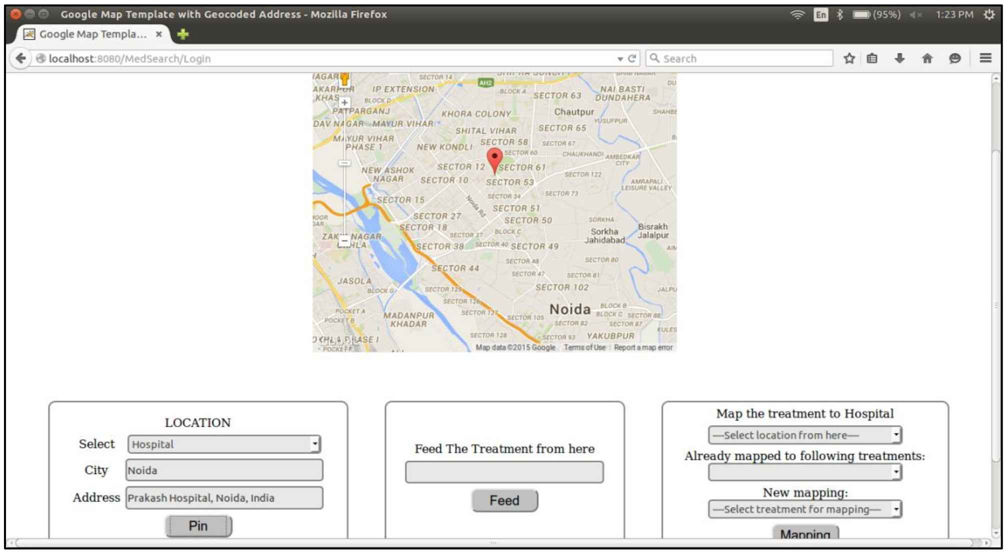
Username bhavya

Password •••••••

Fig. 6.3.4: Admin Panel Login

Test Case 5:

Input: Location
Expected Output: Successful pin on the map.
Actual Output: Successful pin on the map.


Google Map Template with Geocoded Address - Mozilla Firefox

localhost:8080/MedSearch/Login

LOCATION
Select: Hospital
City: Noida
Address: Prakash Hospital, Noida, India
Pin

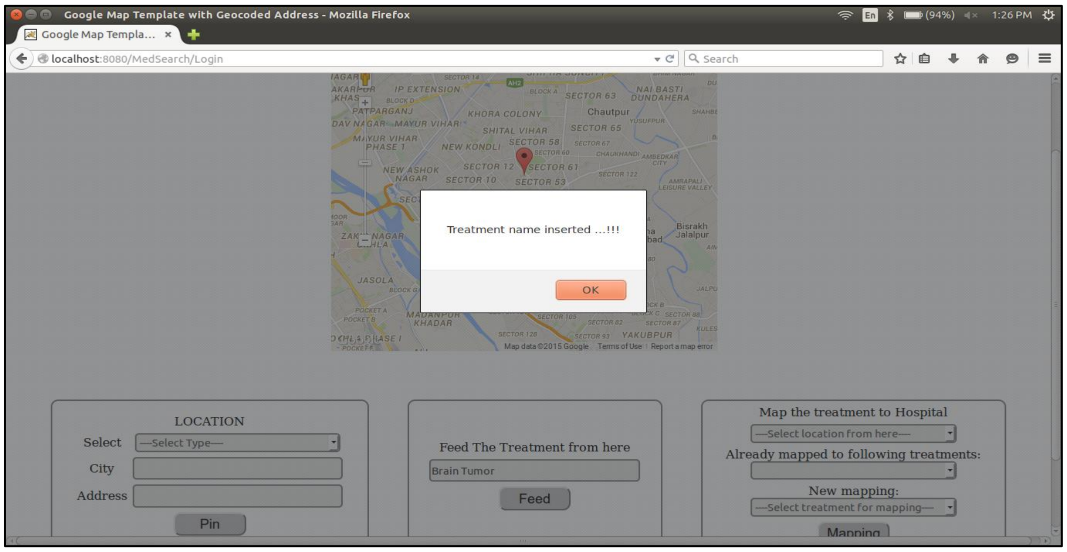
Feed The Treatment from here
Feed

Map the treatment to Hospital
—Select location from here—
Already mapped to following treatments:
New mapping:
—Select treatment for mapping—
Mapping

Fig. 6.3.5: Pinning location on map while entering in database

Test Case 6:

Input: Any treatment name
Expected Output: Treatment name inserted.
Actual Output: Treatment name inserted.


Google Map Template with Geocoded Address - Mozilla Firefox

localhost:8080/MedSearch/Login

Treatment name inserted ...!!!
OK

LOCATION
Select: —Select Type—
City:
Address:
Pin

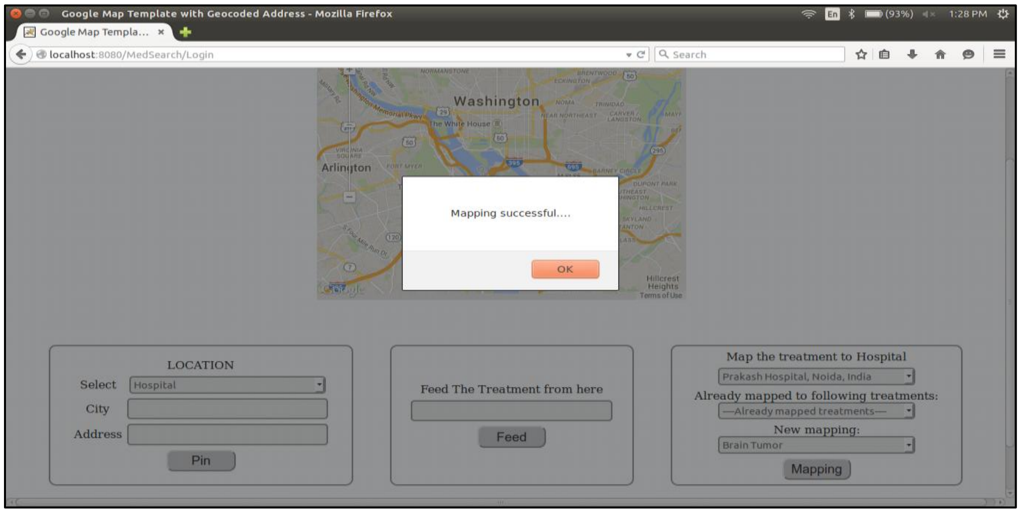
Feed The Treatment from here
Brain Tumor
Feed

Map the treatment to Hospital
—Select location from here—
Already mapped to following treatments:
New mapping:
—Select treatment for mapping—
Mapping

Fig. 6.3.6: Successful insertion of treatment name

Test Case 7:

Input: Hospital, Treatment in “Map the treatment to Hospital”
Expected Output: Mapping Successful.
Actual Output: Mapping Successful.


Google Map Template with Geocoded Address - Mozilla Firefox
Google Map Templa...
localhost:8080/MedSearch/Login
Search
Washington
Arlington
OK
Pin
Feed
Mapping

Mapping successful....

LOCATION
Select: Hospital
City:
Address:

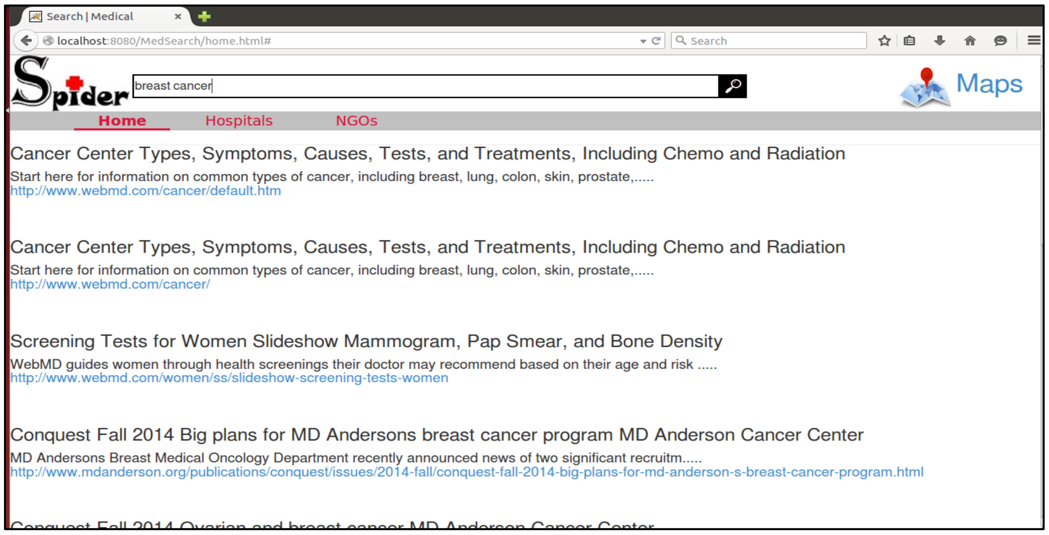
Feed The Treatment from here

Map the treatment to Hospital: Prakash Hospital, Noida, India
Already mapped to following treatments: —Already mapped treatments—
New mapping: Brain Tumor

Fig. 6.3.7: Successful mapping

Test Case 8:

Input: Breast Cancer
Expected Output: Series of results including Title, Information and Link.
Actual Output: Series of results including Title, Information and Link.



breast cancer

Cancer Center Types, Symptoms, Causes, Tests, and Treatments, Including Chemo and Radiation
Start here for information on common types of cancer, including breast, lung, colon, skin, prostate,.....
http://www.webmd.com/cancer/default.htm

Cancer Center Types, Symptoms, Causes, Tests, and Treatments, Including Chemo and Radiation
Start here for information on common types of cancer, including breast, lung, colon, skin, prostate,.....
http://www.webmd.com/cancer/

Screening Tests for Women Slideshow Mammogram, Pap Smear, and Bone Density
WebMD guides women through health screenings their doctor may recommend based on their age and risk .....
http://www.webmd.com/women/ss/slideshow-screening-tests-women

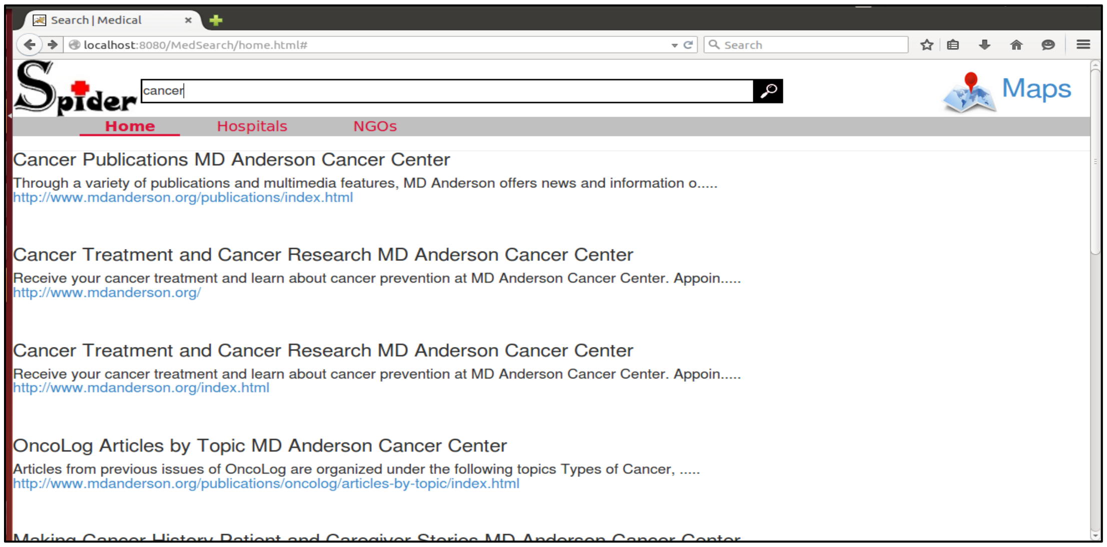
Conquest Fall 2014 Big plans for MD Andersons breast cancer program MD Anderson Cancer Center
MD Andersons Breast Medical Oncology Department recently announced news of two significant recruitm.....
http://www.mdanderson.org/publications/conquest/issues/2014-fall/conquest-fall-2014-big-plans-for-md-anderson-s-breast-cancer-program.html

Conquest Fall 2014 Ovarian and breast cancer MD Anderson Cancer Center

Fig. 6.3.8: Searching a query

Test Case 9:

Input: Cancer, Home tab
Expected Output: List of informative results.
Actual Output: List of informative results.



Cancer Publications MD Anderson Cancer Center
Through a variety of publications and multimedia features, MD Anderson offers news and information o.....
http://www.mdanderson.org/publications/index.html

Cancer Treatment and Cancer Research MD Anderson Cancer Center
Receive your cancer treatment and learn about cancer prevention at MD Anderson Cancer Center. Appoin.....
http://www.mdanderson.org/

Cancer Treatment and Cancer Research MD Anderson Cancer Center
Receive your cancer treatment and learn about cancer prevention at MD Anderson Cancer Center. Appoin.....
http://www.mdanderson.org/index.html

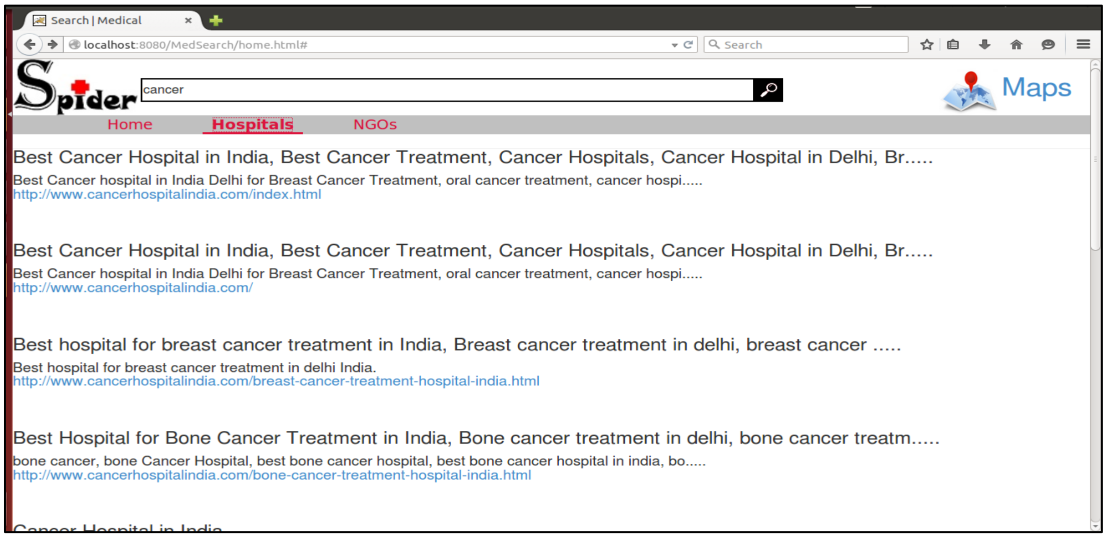
OncoLog Articles by Topic MD Anderson Cancer Center
Articles from previous issues of OncoLog are organized under the following topics Types of Cancer, ...
http://www.mdanderson.org/publications/oncolog/articles-by-topic/index.html

Making Cancer History Patient and Caregiver Stories MD Anderson Cancer Center

Fig. 6.3.9: Disease information in home tab

Test Case 10:

Input: Cancer, Hospitals tab
Expected Output: List of sites which provide hospitals for the disease.
Actual Output: List of sites which provide hospitals for the disease.



Best Cancer Hospital in India, Best Cancer Treatment, Cancer Hospitals, Cancer Hospital in Delhi, Br.....
Best Cancer hospital in India Delhi for Breast Cancer Treatment, oral cancer treatment, cancer hospi.....
http://www.cancerhospitalindia.com/index.html

Best Cancer Hospital in India, Best Cancer Treatment, Cancer Hospitals, Cancer Hospital in Delhi, Br.....
Best Cancer hospital in India Delhi for Breast Cancer Treatment, oral cancer treatment, cancer hospi.....
http://www.cancerhospitalindia.com/

Best hospital for breast cancer treatment in India, Breast cancer treatment in delhi, breast cancer .....
Best hospital for breast cancer treatment in delhi India.
http://www.cancerhospitalindia.com/breast-cancer-treatment-hospital-india.html

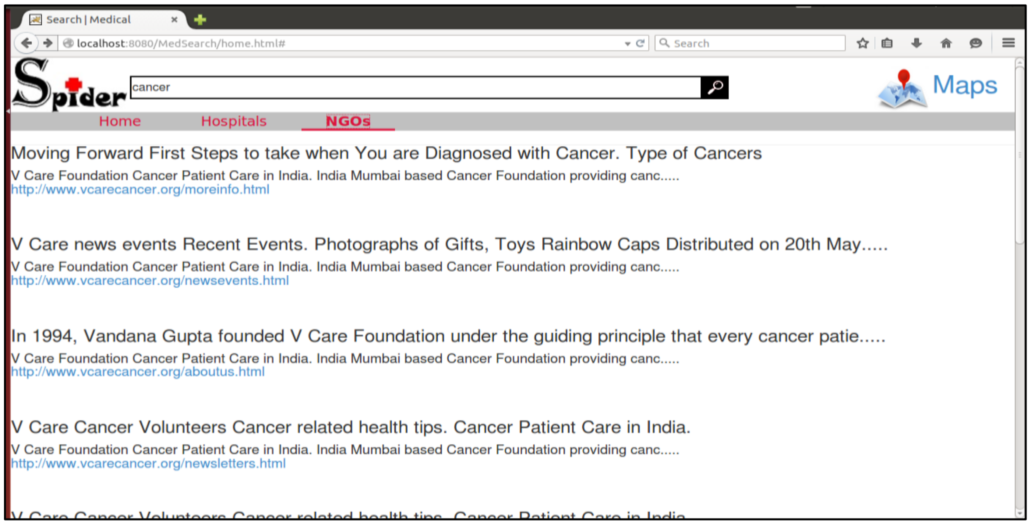
Best Hospital for Bone Cancer Treatment in India, Bone cancer treatment in delhi, bone cancer treatm.....
bone cancer, bone Cancer Hospital, best bone cancer hospital, best bone cancer hospital in india, bo.....
http://www.cancerhospitalindia.com/bone-cancer-treatment-hospital-india.html

Cancer Hospital in India

Fig. 6.3.10: List of hospitals in Hospitals tab

Test Case 11:

Input: Cancer, NGOs tab
Expected Output: List of sites which provide NGOs working for the disease queried.
Actual Output: List of sites which provide NGOs working for the disease queried.



Moving Forward First Steps to take when You are Diagnosed with Cancer. Type of Cancers
V Care Foundation Cancer Patient Care in India. India Mumbai based Cancer Foundation providing canc.....
http://www.vcarecancer.org/moreinfo.html

V Care news events Recent Events. Photographs of Gifts, Toys Rainbow Caps Distributed on 20th May.....
V Care Foundation Cancer Patient Care in India. India Mumbai based Cancer Foundation providing canc.....
http://www.vcarecancer.org/newsevents.html

In 1994, Vandana Gupta founded V Care Foundation under the guiding principle that every cancer patie.....
V Care Foundation Cancer Patient Care in India. India Mumbai based Cancer Foundation providing canc.....
http://www.vcarecancer.org/aboutus.html

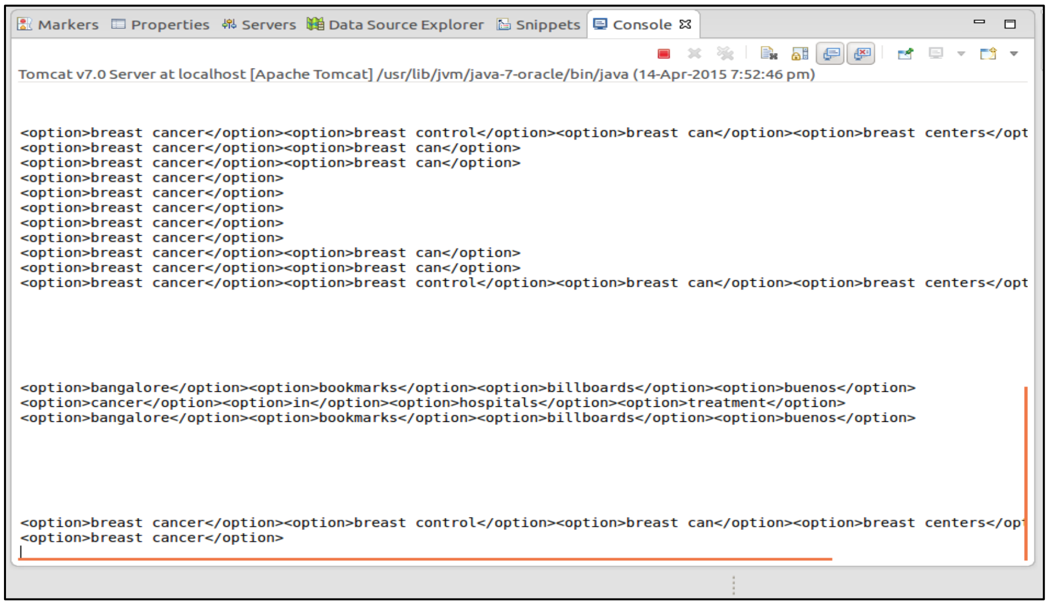
V Care Cancer Volunteers Cancer related health tips. Cancer Patient Care in India.
V Care Foundation Cancer Patient Care in India. India Mumbai based Cancer Foundation providing canc.....
http://www.vcarecancer.org/newsletters.html

V Care Cancer Volunteers Cancer related health tips. Cancer Patient Care in India.

Fig. 6.3.11: List of NGOs in NGOs tab

Test Case 12:

Input: b
Expected Output: List of words beginning with ‘b’.
Actual Output: List of words beginning with ‘b’.



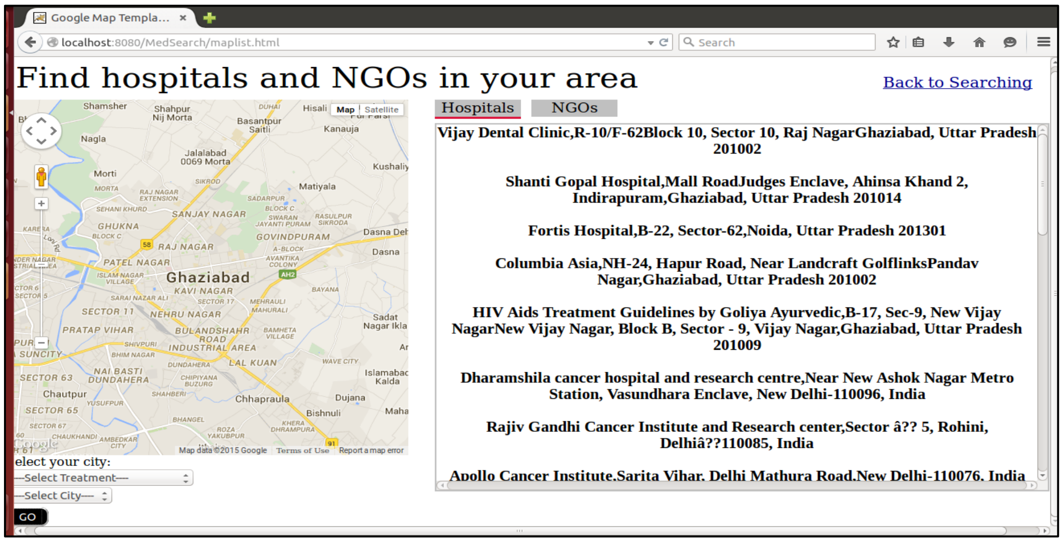
Tomcat v7.0 Server at localhost [Apache Tomcat] /usr/lib/jvm/java-7-oracle/bin/java (14-Apr-2015 7:52:46 pm)

```
<option>breast cancer</option><option>breast control</option><option>breast can</option><option>breast centers</opt
<option>breast cancer</option><option>breast can</option>
<option>breast cancer</option><option>breast can</option>
<option>breast cancer</option>
<option>breast cancer</option>
<option>breast cancer</option>
<option>breast cancer</option>
<option>breast cancer</option>
<option>breast cancer</option><option>breast can</option>
<option>breast cancer</option><option>breast can</option>
<option>breast cancer</option><option>breast control</option><option>breast can</option><option>breast centers</opt



<option>bangalore</option><option>bookmarks</option><option>billboards</option><option>buenos</option>
<option>cancer</option><option>in</option><option>hospitals</option><option>treatment</option>
<option>bangalore</option><option>bookmarks</option><option>billboards</option><option>buenos</option>



<option>breast cancer</option><option>breast control</option><option>breast can</option><option>breast centers</op
<option>breast cancer</option>
```

Fig. 6.3.12: Auto complete

Test Case 13:

Input: None
Expected Output: List of all Hospitals and NGOs.
Actual Output: List of all Hospitals and NGOs.



Find hospitals and NGOs in your area

Back to Searching

Hospitals NGOs

Vijay Dental Clinic,R-10/F-62Block 10, Sector 10, Raj NagarGhaziabad, Uttar Pradesh 201002

Shanti Gopal Hospital,Mall RoadJudges Enclave, Ahinsa Khand 2, Indirapuram,Ghaziabad, Uttar Pradesh 201014

Fortis Hospital,B-22, Sector-62,Noida, Uttar Pradesh 201301

Columbia Asia,NH-24, Hapur Road, Near Landcraft GolflinksPandav Nagar,Ghaziabad, Uttar Pradesh 201002

HIV Aids Treatment Guidelines by Goliya Ayurvedic,B-17, Sec-9, New Vijay NagarNew Vijay Nagar, Block B, Sector - 9, Vijay Nagar,Ghaziabad, Uttar Pradesh 201009

Dharamshila cancer hospital and research centre,Near New Ashok Nagar Metro Station, Vasundhara Enclave, New Delhi-110096, India

Rajiv Gandhi Cancer Institute and Research center,Sector â?? 5, Rohini, Delhiâ??110085, India

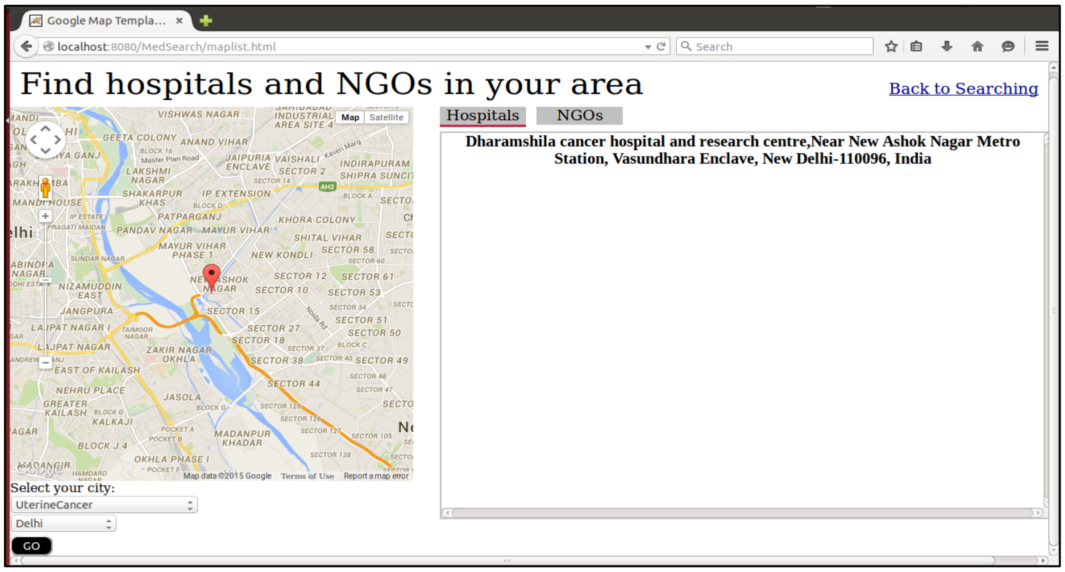
Apollo Cancer Institute,Sarita Vihar, Delhi Mathura Road,New Delhi-110076, India

elect your city:
—Select Treatment—
—Select City—
GO

Fig. 6.3.13: List of all hospitals and NGOs in maps

Test Case 14:

Input: Uterine Cancer, Delhi
Expected Output: List of all Hospitals concerned with the same.
Actual Output: List of all Hospitals concerned with the same.



Find hospitals and NGOs in your area

Back to Searching

Hospitals NGOs

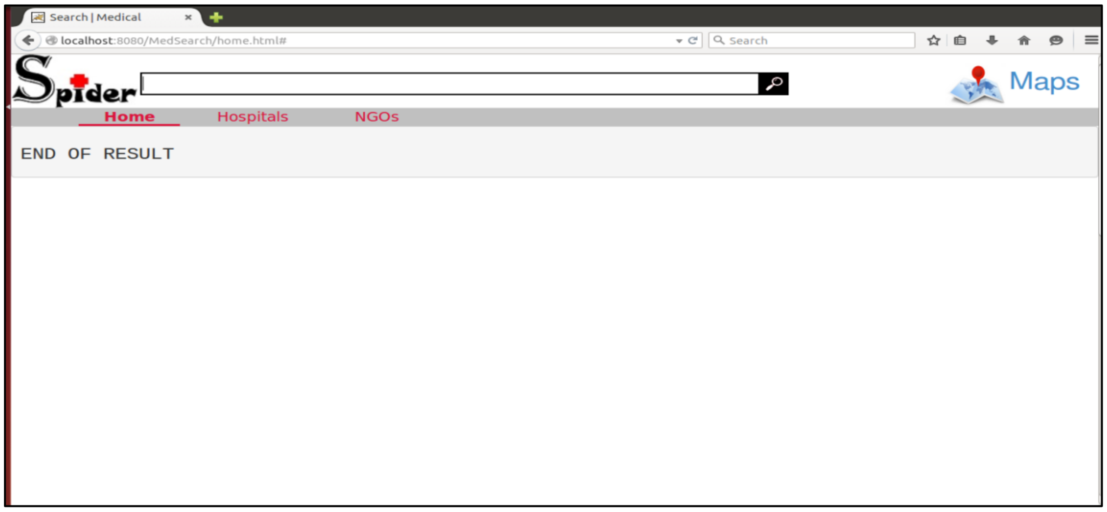
Dharamshila cancer hospital and research centre,Near New Ashok Nagar Metro Station, Vasundhara Enclave, New Delhi-110096, India

Select your city:
UterineCancer
Delhi
GO

Fig. 6.3.14: List of hospitals in particular city providing treatment of particular disease

Test Case 15:

Input: None
Expected Output: End of Result.
Actual Output: End of Result.

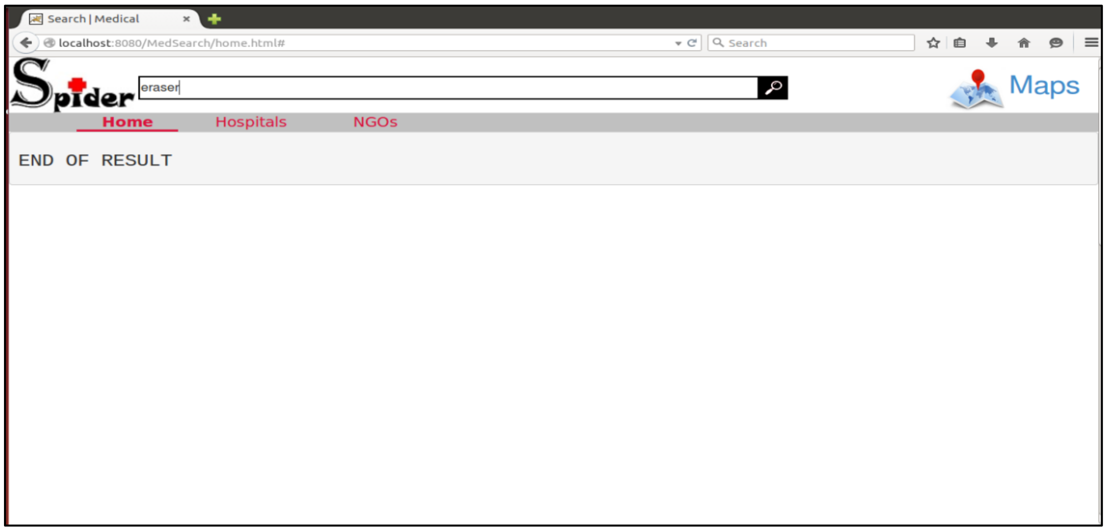


END OF RESULT

Fig. 6.3.15: End of result for blank input

Test Case 16:

Input: Eraser
Expected Output: End of Result.
Actual Output: End of Result.



END OF RESULT

Fig. 6.3.16: End of result for irrelevant input

Chapter-7
SYSTEM IMPLEMENTATION

7. System Implementation

System Implementation comprises of the algorithm of the functions that different module perform in the system. The algorithm is a step by step procedure of how the modules work.

1. Crawler

1. Read seeds from seeds.xml and insert them to database.
2. Read seeds from the database one by one and pass them to crawler function.
3. The crawler function will fetch the page if it hasn't been fetch.
4. It sends the page to indexer as document object.
5. It also extracts the hyperlink from the page and sends them to the crawler function in step 2.

2. Indexer

1. The indexer receives the page.
2. Fetches textual content from it.
3. Remove the stop words.
4. Tokenizes the content.
5. Insert those words with their page ids and tag information.

3. Auto Complete

```
1. Write the indexed words from database into an XML file and set the initial
   frequency of each word to 1.
2. Read the XML file into a Document doc.
3. As soon as the user types some string in the input box,
   if (string has no space)
   {
       Pick it through AJAX,
       Search Doc for words that have this string as prefix.
       From the list of matched words, display the top 4 results which have highest
       frequency.
   }
```

```
else
{
        Pick that string using AJAX,
        Spilt the string based on “spaces”,
        Pick the last word from the string,
        Search for words that start with the last word in the Document doc,
        Now, among the list of all the words that start with that string, sort the list in
        decreasing order based on frequency,
        Concatenate top 4 results with the results of previous words,
        Display them to the user.
}
```

4. When the user types some string and presses “Enter” or clicks on “Go”,
    - Split the string using space,
    - Remove stop words,
    - Pick each word,
    - Find that word in the Doc, and
    - Increase its frequency in the XML file and Database by 1.
5. For each string entered thereafter, repeat steps 3-4.

4. Map Module

1. Take user input.
2. User selects name of city and treatment.
3. The treatment and city data is fed to the SQL query being processed to display the results.
4. The address of the hospitals and NGOs are displayed in the respective tags.
5. When the user clicks the address of any hospital or NGO, it gets pinned on the map displayed on the page.
6. The user can go back to searching by clicking the link on the top right corner of the page.

5. Admin Panel

1. The admin panel is authenticated by two fields, the username and password.
2. If the username and password matches the data previously stored in the database, the authentication process is complete and the login is successful.
3. The admin can feed the information by selecting a hospital or NGO.
4. Then feeds the address and city of the hospital or NGO, respectively. And click the submit button to insert new data in database.

5. Admin feeds the treatments and then maps the treatment with their corresponding hospitals and NGOs. The mapping function helps avoid redundancy in the records of the database.
6. Get Frequency and Clear Frequency functions counts the number of words searched and clears the xml file if the words searched increases a limit of 100 words, respectively.
7. Update AutoComplete function updates the frequency of words in an xml file after reading the frequencies from the database.

Chapter-8

RESULTS AND CONCLUSION

8. Results, Discussions, Conclusion and Future Scope of Study

The results describe the outcome of all the proposed objectives described previously and their successful completion. Discussions are made to enlighten all the outcomes we derive from this project.
Conclusion concludes all the work that was done in the project with their suitable outcomes, and future scope tells us about different fields where the system can be used in further and what all developments are possible with this developed project.

8.1. Results

- The web crawler is fully implemented and crawls almost ten pages per minute.
- The indexer is completely designed and implemented and works hand in hand with the crawler.
- The ranker is fully implemented with the help of HITS algorithm, which ranks the pages crawled for better ranked results.
- The auto-complete module and the map module are completely implemented.
- The front end is designed and implemented along with the integration of all the other modules of the project.

8.2. Discussions

In response to the huge market need, Health line, a popular Web search engine for medical information, came into existence in October 2005. Shortly thereafter, Google announced its own medical Web search engine, Google Health, in May 2006. There are also several other medical Web search engines. While these systems have their own merits, they mostly treat medical search in much the same way as traditional Web search. Medical search has several unique requirements that distinguish itself from traditional Web search. A common scenario in which a person performs medical search is that he feels uncomfortable but is uncertain about his exact medical problems. In this case, the searcher usually prefers to learn all kinds of knowledge that is related to his situation. However, existing medical Web search engines are optimized for precision and concentrate their search results on a few topics. This lack of diversity problem is aggravated by the nature of medical Web pages. When discussing a medical topic, many medical Web sites use similar, but not identical, descriptions by paraphrasing contents in medical textbooks and research papers. Hence, search results provided by existing medical Web search engines often contain much semantic redundancy, which cannot be easily handled by existing methods for identifying near-duplicate documents or result relevancy. To find useful medical information, the searcher often has to go through a large number of Web pages laboriously. This system is used to search the queries related to medical field and provide relevant results like information about the disease, hospitals and NGOs offering treatment for the same. An additional feature of the system is the maps service through which user can pin the hospitals and NGOs of a particular city on the map and find places where the treatment is offered.

8.3. Conclusion

People are thirsty for medical information. Existing Web search engines often cannot handle medical search well because they do not consider its special requirements. Often a medical information searcher is uncertain about his exact questions and unfamiliar with terminology. Therefore, he sometimes prefers to pose long queries, describing his symptoms and situation in plain English, and receive comprehensive, irrelevant information from search results. The project presents a specialized Web search engine for medical information retrieval to overcome these challenges. It can help ordinary Internet users throughout the entire process of medical treatment. The design of this system takes into consideration the unique requirements of medical search. It supports queries written in plain English, provides relevant search results, and suggests related Hospitals, NGOs on the maps, with proper ranking and annotation. These features are attractive to ordinary Internet users who have little medical knowledge and are unfamiliar with medical terminology. Consumer-centric medical search is the main aim of this project. This system is used to search the queries related to medical field and provide relevant results in three tabs namely, Home, Hospitals and NGOs. It also provides maps service through which user can choose his treatment and city corresponding to which the results of hospital's address and Ngo's address are displayed in a tab. The address can be pinned on the map to verify its existence. The system provides a better view of all the consolidated data that a user requires for medical treatment.

8.4. Future Study

- The current project is entirely based on medical field, be it the queries, maps, or results. In the future, the scope of this search engine can be extended to other fields like:

    - Tourism
    - Education
    - Transport, etc.

- In the current project, keyword-based searching is implemented, i.e., searching solely depends on the keywords that the user types in the text box. The results presented as an output may not concern the same interpretation. Thus, in the future, Artificial Intelligence can be used to give relevant results for context-based searching in the system.

- Word association can be implemented in this project in future.

- The present system crawls only websites and does not include files. In future, the crawler can crawl files in PDF, PPT, DOCX and many other formats. Also it can include images in search results.

- Auto correction is a feature which enables the user to retrieve the correct search results even for incorrectly spelled query.

- The present map module pins the hospitals and NGOs on the map but does not provide the direction to each the destination. In future, map direction module can also be implemented.

- IP Address based searching can be implemented in future.

- Voice based searching is searching the results via voice input instead of providing the result through the keyboard. The benefits of voice searching are that it makes input easier and faster.

APPENDIX A
SOURCE CODE

(It includes important source code only.)

A.1 Crawler & Indexer

1. The seeds in seeds.xml file is feed into processController() function of crawl pages CrawlPages.java. This is a recursive function which sends each link it gets into the crawling function.

```
//This method will insert the seed and will recursively find all its pages at different depth.
Public static void processController(int seed_id, String page_url ,int depth) {
        if(depth<=maxdepth) {
                if(page_url.length()<200) {
                        boolean isUnique = false;
                        boolean isfetched = false;
                        isUnique = isUnique(page_url);
                        if(!isUnique)
                        {
                                isfetched = fetched(returnPageId(page_url));
                        }
                        if(!isfetched)
                        {
                                PageInfoBean page_object = pageInfo(seed_id, page_url,
depth,isUnique);
                                if(page_object.isReadStatus())
                                {
                                        for(String url: page_object.getLinks()) {
                                                if(!isBroken(url))processController(seed_id,url,
depth+1);
                                        }
                                }
                        page_object = null;
                        }
                                elseincreaseLink(returnPageId(page_url), depth);
                        }
                }
        }
}
```

2. The fetched page is passed to writeToFile() function of PreIndexer.java class in the form of document object. This class contains functions for fetching the specific content from the page.

```
static String[] tofetch= {"h1","h2","h3","h4","h5","h6","em","strong","b"};
Elements elements = null;
//Replace all symbols, stopwords and fetches the data between the above tags.
Public void writeToFile(int page_id, String page_url, int type_id, Document doc) {
        ArrayList<String> array = new ArrayList<String>();
        for(Element meta : doc.select("meta"))
        {
                String text = meta.attr("content").replaceAll("\\[.*?\\]", "");
                array = new ProcessStopWords().removeStopWords(text);
                for(String word : array)
                {
                        WordIndexer.wordIndexer(word, page_id, 1, type_id);
                }
        }
        for(String tag: tofetch)
        {
                elements = doc.select(tag);
                for(Element e: elements)
                {
                        array = new ProcessStopWords().removeStopWords(e.text());
                        for(String word : array)
                        {
                                WordIndexer.wordIndexer(word, page_id,degreeReturn(tag),type_id);
                        }
                }
        }
        array = null;
}
```

3. The method below adds an entry of word to page mapping for the existing words in the database.

```
//This method adds a page entry for a particular word.
Public static boolean addPageEntry(int word_id, int page_id, int degree, int type_id)
{
        boolean add = false;
        try
        {
                PreparedStatement stmt = conn.prepareStatement("INSERT INTO words_detail
(word_id, page_id, frequency, degree, type_id) VALUES (?,?,?,?,?)");
                stmt.setInt(1, word_id);
                stmt.setInt(2, page_id);
                stmt.setInt(3, 1);
                stmt.setInt(4, degree);
                stmt.setInt(5, type_id);
                int status = stmt.executeUpdate();
                if(status != 0 ) add = true;
                stmt.close();
        }
        catch (SQLException e)
        {
                System.out.println("Error: SQLException occurred while adding an pageentry to 
database.");
        }
        return add;
}
```

A.2 Searcher& Ranker

Now after the indexes are made in the database in the "words_details" table, the system is ready for searching. The searching query is entered from the Home page of the project, and the words are matched with the words in the database, the best page links are fetched in return which are ranked high according to ranking algorithms.

1. This function has the search query used in searching and ranking. It makes the query ready by adding the word_id string, limits of the result and type of the result in the query.

```
//The SQL query used for searching and ranking.
protected String SQLQuery(String array, String type, int page)
{
        int lowerlimit = (page*10)-10;
        String query = "SELECT B.page_id FROM allpages_table AS A INNER JOIN \n" +"(SELECT 
page_id, COUNT(page_id) AS matchCount, AVG(degree) AS avgDeg, SUM(frequency) as sumFreq \n" 
+ "FROM words_detail WHERE type_id="+type+" and word_id IN ### \n" + "GROUP BY page_id 
ORDER BY matchCount DESC, avgDeg, sumFreq DESC limit "+lowerlimit+",10) AS B \n" + "ON 
A.page_id=B.page_id ORDER BY (A.incoming_link + A.outgoing_link)/2.0 DESC;";
        query = query.replaceAll("###", array);
        return query;
}
```

A.3 Auto Complete

The codes below are related to **Auto Completion** system that is used in Searching interface. The Auto Completion system fetches the most searched words which match with the word the user is typing. It uses the functions below.

The incomplete words that are in process of being typed are sent to the AutoComplete.java which receives the search string with incomplete word and match the incomplete word with the most searched words in the transfer.xml file. The matching is done with the help of match() function in Matcher.java class.

1. The searching for AutoComplete system is done in the transfer.xml file and the top four results with highest frequencies are returned back to the home page and put in the drop down for user convenience. The matching and sorting according to frequencies are done with the help of code below.

```
//Receives the string and returns the matching results.
public String match(String string, File xmlfile) throws ParserConfigurationException, SAXException,
IOException
{
        String result="";
        String[] words = string.split("\\s+");
        int arraysize = words.length;
        string = words[arraysize-1];
        Document doc = newReadXML().read(xmlfile);
        NodeList first = doc.getDocumentElement().getChildNodes();
        HashMap<String, Integer> hmap = new HashMap<String, Integer>();
        String freq = null;
        for(int i=0;i<first.getLength();i++)
        {
                if(first.item(i).getTextContent().startsWith(string))
                {
                        freq=first.item(i).getAttributes().getNamedItem("freq").getNo eValue();
                        int intfreq=Integer.parseInt(freq);
                        hmap.put(first.item(i).getTextContent(), intfreq);
                }
        }
        String prevwords="";
        for(int i=0;i<(arraysize-1);i++)
        {
                prevwords = prevwords.concat(words[i]);
                prevwords = prevwords.concat(" ");
```

```
            }
            Map<String, Integer> map = sortByValues(hmap);
            Set<Entry<String, Integer>> set = map.entrySet();
            Iterator<Entry<String, Integer>> iterator2 = set.iterator();
            int count=1;
            while(iterator2.hasNext() && count<=4)
            {
                        Entry<String, Integer> me = iterator2.next();
                        result=result.concat("<option>"+prevwords+""+me.getKey()+"</option>");
                        count++;
            }
            return result;
}
```

A.4 Maps

The Search Engine is equipped with Maps feature which help the user to search for hospitals and NGOs for particular treatment and city. These Hospitals and treatments are manually fed by the admin by an admin panel.

1. The Maps page of user takes the input of treatment and city from the user. The both parameters are sent to Location.java servlet. The following function receives it, fetches the matching results and sends it back to the user.

```
//doGet method of Location.java. Resposible for searching matching Hospitals and NGOs according  to
users requirement.
Protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
                        response.setContentType("text/html");
                        PrintWriter out= response.getWriter();
                         String city = request.getParameter("city");
                         String treatment = request.getParameter("treatment");
                         String result="";
                         String type = request.getParameter("type");
                        if(treatment.equals("0")) treatment = "treatment_id";
                        if(city.equals("0")) city = "B.city";
                        else {
                                     city = "'".concat(city);
                                    city = city.concat("'");
                         }
                        try{
                                    Connection con= newDatabase().returnConnection();
```

```
                              Statement stmt = con.createStatement();
                              ResultSet rs= stmt.executeQuery("SELECT DISTINCT
B.address,B.type_id,B.location_id FROM maps_list AS B INNER JOIN (SELECT location_id FROM
location_mapping where treatment_id="+treatment+") AS A ON A.location_id=B.location_id WHERE
B.city="+city+" and B.type_id="+type+"");
                              while(rs.next())  {
                                        result=result.concat("<div style='cursor:pointer;'
id='"+rs.getInt("B.location_id")+"'
onclick='UpdateMap(this)'>"+rs.getString("B.address")+"</div><br>"); }
                              if(result.equals("")) result = result.concat("NO RESULTS FOUND !!!");
                    out.println(result);
                              }catch(Exception e){
                                        System.out.println(e); }
}
```

REFERENCES

[1] Skiena, S. S. (2008). Introduction to Algorithm Design. In *The Algorithm Design Manual* (pp. 3-30). Springer London.

[2] Coppin, B. (2004). *Artificial intelligence illuminated*. Jones & Bartlett Learning.

[3] Yoo, A., Chow, E., Henderson, K., McLendon, W., Hendrickson, B., & Catalyurek, U. (2005, November). *A scalable distributed parallel breadth-first search algorithm on BlueGene/L*. In *Supercomputing, 2005. Proceedings of the ACM/IEEE SC 2005 Conference* (pp. 25-25). IEEE.

[4] Shen, A. (2011). *Algorithms and programming: problems and solutions*. Springer Science & Business Media.

[5] Deo, N. (2004). *Graph theory with applications to engineering and computer science*. PHI Learning Pvt. Ltd..

[6] Anderson, S. S. (1970). *Graph theory and finite combinatorics*. Markham Pub. Co.

[7] Bodino, G. A. (1962). *Economic applications of the theory of graphs* (Vol. 1). Routledge.

[8] Behzad, M., & Chartrand, G. (1972). *Introduction to the Theory of Graphs*. Allyn and Bacon.

[9] Chen, W. K. (2012). *Applied graph theory* (Vol. 13). Elsevier.

[10] Nilsson, N. J. (1998). *Artificial intelligence: a new synthesis*. Morgan Kaufmann.

[11] Hersh, WilliamR,.(2002).Information Retrieval. Springer.

[12] Jackson, P. C. (1985). *Introduction to artificial intelligence*. Courier Corporation.

[13] Luger, G. F. (2005). *Artificial intelligence: structures and strategies for complex problem solving*. Pearson education.

[14] Minsky, M. (1995). Computation & intelligence: Collected readings. *Luger, GF*.

[15] Minton, S. (1988). *Learning search control knowledge: An explanation-based approach* (Vol. 61). Springer Science & Business Media.

[16] Luo, G., Tang, C., Yang, H., & Wei, X. (2008, October). MedSearch: a specialized search engine for medical information retrieval. In *Proceedings of the 17th ACM conference on Information and knowledge management* (pp. 143-152). ACM.