Savitribai Phule Pune University

A PRELIMINARY PROJECT REPORT

ON

"Hypertextual Search Engine"

SUBMITTED BY

MS.SHEETAL KOLEKAR	**B120324301**
MR.NIKHIL LOHAKARE	**B120324305**
MS.PRIYANKA RAUT	**B120324360**

Under the guidance of

Prof. Sneha Pisey

Department of Computer Engineering

Modern Education Society's

College Engineering, Pune-411001

[2017-18]

MODERN EDUCATION SOCIETY'S

College of Engineering, Pune 01

C E R T I F I C A T E

This is to certify that the Project Entitled

"Hypertextual Search Engine"

SUBMITTED BY

MS.SHEETAL KOLEKAR	**B120324301**
MR.NIKHIL LOHAKARE	**B120324305**
MS.PRIYANKA RAUT	**B120324360**

Is a bonafide work carried out by students under the supervision of Prof. Sneha Pisey and it is submitted towards the partial fulfilment of the requirement of Bachelor of Engineering (Computer Engineering).

Prof. S.H. Pisey
Internal Guide
Dept. of Computer Engg.

Dr.N.F.Shaikh
H.O.D
Dept. of Computer Engg.

Signature of Internal Examiner

Signature of External Examiner

ABSTRACT

In this report, we present a prototype of a large-scale search engine which makes heavy use of the structure present in hypertext. Our search engine is designed to crawl and index the web efficiently and produce much more satisfying search results than existing systems. To engineer a search engine is a challenging task. Search engines index tens to hundreds of millions of web pages involving a comparable number of distinct terms. They answer tens of millions of queries every day. Despite the importance of large-scale search engines on the web, very little academic research has been done on them.Furthermore, due to rapid advance in technology and web proliferation, creating a web search engine today is very difficult. We are providing an in-depth description of our large-scale web search engine. Apart from the problems of scaling traditional search techniques to data of this magnitude, there are new technical challenges involved with using the additional information present in hypertext to produce better search results. We address this question of how to build a practical large-scale system which can exploit the additional information present in hypertext. Also we look at the problem of how to effectively deal with uncontrolled hypertext collections where anyone can publish anything they want.

ACKNOWLEDGEMENT

It gives me great pleasure and satisfaction in presenting this seminar on “Hypertextual Search Engine”. I would like to express my deep sense of gratitude towards the Principal Dr. A.A Keste and HOD of Computer Department Prof N.F. Shaikh. Special thanks to my seminar guide Prof. Sneha Pisey for her valuable support.

I thank with all my heart and express my honour and deep gratitude for all staff members of Computer Department for helping me in every way possible. I would like to thank all those,who have directly or indirectly helped me for the completion of the work during this seminar.

Ms. Sheetal Kolekar
Mr. Nikhil Lohakare
Ms.Priyanka Raut
(B.E. Computer Engg.)

INDEX

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

SYNOPSIS

1.1 PROJECT TITLE

Hypertextual search engine

1.2 PROJECT OPTION

Internal Project

1.3 INTERNAL GUIDE

Prof.Sneha Pisey

1.4 TECHNICAL KEYWORDS

1. Information retrival
2. Distributed systems
3. Index
4. Crawler
5. World Wide Web
6. Parser
7. Inverted index
8. PageRank

1.5 PROBLEM STATEMENT

The problem of information overload has brought about the challenge of how to find the relevant information in user friendly manner, in language that user can really understand. English is not native language of India, in finding particular information on the internet and to interpret that information user need to know english very well, as it is difficult to people who live in rural areas. Focusing on small index size, crawling speed and language we are developing hypertextual search engine.

1.6 ABSTRACT

In this report, we present a prototype of a large-scale search engine which makes heavy use of the structure present in hypertext. Our search engine is designed to crawl and index the web efficiently and produce much more satisfying search results than existing systems. To engineer a search engine is a challenging task. Search engines index tens to hundreds of millions of web pages involving a comparable number of distinct terms. They answer tens of millions of queries every day. Despite the importance of large-scale search engines on the web, very little academic research has been done on them.Furthermore, due to rapid advance in technology and web proliferation, creating a web search engine today is very difficult. We are providing an in-depth description of our large-scale web search engine. Apart from the problems of scaling traditional search techniques to data of this magnitude, there are new technical challenges involved with using the additional information present in hypertext to produce better search results. We address this question of how to build a practical large-scale system which can exploit the additional information present in hypertext. Also we look at the

problem of how to effectively deal with uncontrolled hypertext collections where anyone can publish anything they want.

1.7 GOALS AND OBJECTIVES

- Scaling with the Web-
- Improved Search Quality
- Academic Search Engine Research
- Crawling and Parsing the Web Efficiently
- Reduced Index Size
- Giving local searches importacne
- Use social signals to improve rankings

1.9 RELEVANT MATHEMATICS ASSOCIATED WITH THE PROJECT

Backend Process :

- Input: intitial URL for crawler to start crawling eg. mescoepune.org
- Output: indexed pages of crawled web

Front end :

- Input: Search query - eg- mescoe
- Output: Search engine ranking of page based on core algorithm
- Data structures: Inverted index typically in the form of a hash table or binary tree.
- Constraint: They rank websites in part according to concepts such as intrinsic authority, which in many cases are flawed, and which allow rankings to be manipulated.

1.10 NAMES OF CONFERENCES / JOURNALS WHERE PAPERS CAN BE PUBLISHED

International World Wide Web Conference ,IEEEE

CHAPTER 2

TECHNICAL KEYWORDS

2.1 AREA OF PROJECT

Web Search Engine

2.2 TECHNICAL KEYWORDS

1. Information retrival
2. Distributed systems
3. Index
4. Crawler
5. World Wide Web
6. Parser
7. Inverted index
8. PageRank

CHAPTER 3

INTRODUCTION

3.1 PROJECT IDEA

Our idea is to Scale the Web by Crawling and Parsing the Web Efficiently and Building a much smaller index.We will do it with help of hypertexual structure of world wide web.This idea of propagating anchor text to the page it refers to was implemented in the World Wide Web especially because it helps search non-text information, and expands the search coverage with fewer downloaded documents. We use anchor propagation mostly because anchor text can help provide better quality results. Using anchor text efficiently is technically difficult because of the large amounts of data which must be processed. We also intent to improved Search Quality by using various refining criteria and search algorithm. Academic citation literature has been applied to the web, largely by counting citations or backlinks to a given page. This gives some approximation of a page's importance or quality.We can use this idea for imroving search results.

3.2 MOTIVATION OF THE PROJECT

The motivation for doing this project was primarily an interest in undertaking a challenging project in an interesting area of research. The opportunity to learn about a new area of computing not covered in lectures was appealing. The World Wide Web creates many new challenges for information retrieval. It is very large and heterogeneous. Current estimates are that there are over 130 trillion web pages with a doubling life of less than one year. More importantly, the web pages are extremely diverse, ranging from "What is Joe having for lunch today?" to journals about information retrieval. In addition to these major challenges, search engines on the Web must also contend with inexperienced users and pages engineered to manipulate search engine ranking functions. However, unlike "flat" document collections, the World Wide Web is hypertext and provides considerable auxiliary information on top of the text of the web pages, such as link structure and link text. In this project, we take advantage of the link structure of the Web to produce a global importance ranking of every web page.

3.3 LITERATURE SURVEY

Sergey Brin, Rajeev Motwani, Lawrence Page, Terry Winograd [1] "What can you do with a web in your pocket". In Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, 1998. described "A repository of Web pages such as the WebBase is an excellent research tool, enabling experiments that would otherwise be impossible to perform efficiently. And, of course, it can be crucial to the development of better search engines.An important lesson we have learned from these experiments is that size does matter. The extraction experiment would likely have failed if the WebBase had been one third of itscurrent size. Furthermore, the hardware cost of a large WebBase is quite reasonable and trends in disk capacity and computing power make it very likely that many more applications involving a local Web repository will become practical in the

near future.In analyzing the Web, we have found that it is important to look beyond just the text. The extraction experiment made heavy use of formatting and URL's. PageRank takes advantage of the link structure. Google makes use of anchor text and font information. Much of the information on the Web is not in the plain text and many applications can achieve great gains by leveraging it."

Sergey Brin, Rajeev Motwani, Lawrence Page, Terry Winograd [2] , "The PageRank citation ranking: Bringing order to the web," Stanford InfoLab, Stanford, CA, USA, Tech. Rep., 1999. described "PageRank could be used to separate out a small set of commonly used documents which can answer most queries. The full database only needs to be consulted when the small database is not adequate to answer a query. Finally, PageRank may be a good way to help find representative pages to display for a cluster center. We have found a number of applications for PageRank in addition to search which include traffic estimation, and user navigation."

Sergey Brin and Larry Page [3] "The anatomy of a large-scale hypertextual web search engine". In To Appear: Proceedings of the Seventh International Web Conference proposed that "Google is designed to be a scalable search engine. The primary goal is to provide high quality search results over a rapidly growing World Wide Web. Google employs a number of techniques to improve search quality including page rank, anchor text, and proximity information. Furthermore, Google is a complete architecture for gathering web pages, indexing them, and performing search queries over them. " The biggest problem facing users of web search engines today is the quality of the results they get back. While the results are often amusing and expand users' horizons, they are often frustrating and consume precious time. For example, the top result for a search for "Bill Clinton" on one of the most popular commercial search engines was the Bill Clinton Joke of the Day: April 14, 1997. Google and PageRank algorithm is designed to provide higher quality search so as the Web continues to grow rapidly, information can be found easily. In order to accomplish this Google makes heavy use of hypertextual information consisting of link structure and link (anchor) text. Google also uses proximity and font information. While evaluation of a search engine is difficult, we have subjectively found that Google returns higher quality search results than current commercial search engines. The analysis of link structure via PageRank allows Google to evaluate the quality of web pages. The use of link text as a description of what the link points to helps the search engine return relevant (and to some degree high quality) results. Finally, the use of proximity information helps increase relevance a great deal for many queries.

CHAPTER 4

PROBLEM DEFINITION AND SCOPE

4.1 PROBLEM STATEMENT

The problem of information overload has brought about the challenge of how to find the relevant information in user friendly manner, in language that user can really understand. English is not native language of India, in finding particular information on the internet and to interpret that information user need to know english very well, as it is difficult to people who live in rural areas. Focusing on small index size, crawling speed and language we are developing hypertextual search engine.

4.1.1 Goals and objectives

- Scaling with the Web
- Improved Search Quality
- Academic Search Engine Research
- Crawling and Parsing the Web Efficiently
- Reduced Index Size
- Giving local searches importacne
- Use social signals to improve rankings

4.1.2 Statement of scope

The problem of information overload has brought about the challenge of how to Find the relevant information in user friendly manner, in language that user can Really understand. English is not native language of India, in finding particular Information on the internet and to interpret that information user need to know English very well, as it is difficult to people who live in rural areas. Focusing on Small index size, crawling speed, language we are developing hypertextual search engine.

4.2 SOFTWARE CONTEXT

We can use our project software in world wide web search engine industry .Which is thriving today .This project aims to serve our internet customers to search the web more efficiently .the predominant business model for commercial search engines is advertising.

4.3 MAJOR CONSTRAINT

Building a web index is major constraint in our project. Web is biggest database human has ever created and it is increasing day by day .Challenges in handling this humongous amount of data are immense .The Biggest Search index contains hundreds of billions of webpages and is well over 100,000,000 gigabytes in size .This can be major constraint in building our project.

4.4 METHODOLOGIES OF PROBLEM SOLVING AND EFFICIENCY ISSUES

First one is conventional method in which We use software known as web crawlers to discover publicly available webpages. Crawlers look at webpages and follow links on those pages, much like you would if you were browsing content on the web .They go from link to link and bring data about those webpages back to our database .But his method has many efficiency issues as it takes huge amount of space and time to build an index.

The other method in which we will use web spider which will crawl and parse the web at same time that will increase speed of our process .Moreover we intent to decrease size of our index this method allows us to do so by retrieving only important parameter from source .So this method is more efficient.

4.5 SCENARIO IN WHICH MULTI-CORE, EMBEDDED AND DISTRIBUTED COMPUTING USED

In case of crawling the web we can use distributed computing system to increase the speed and efficiency of our crawler .Multicore computing can be USED for crawling as well. We can use to extra cores to increase speed of retrieving information.

4.6 OUTCOME

This project will result into a web based search engine than can handle the user queries and present them with accurate answer. This project will result in a runtime system that gets user queries, retrieves the results out of the index from the right machine, and re-ranks them according to the query. User will get results in user friendly manner.

4.7 APPLICATION

- Searching the world wide web
- Academic and Scientific Research
- Large data with lots of patterns
- Helping Business Grow

4.8 HARDWARE RESOURCES REQUIRED

Sr. No.	Parameter	Minimum Requirement	Justification
1	CPU Speed	2 GHz	For faster crawling and parsing
2	RAM	60 GB	For parsing and handling queries
3	Architecture	64-bit	For parsing
4	Internet speed	28 Mbps	For faster crawling and bandwidth
5	Hard disk	80 GB	To build searchable index

Table 4.1: Hardware Requirements

4.9 SOFTWARE RESOURCES REQUIRED

Platform :

1. Operating System - Linux

2. IDE – PYCham , Notepad ++.

3. Programming Languages – Python ,Html ,PHP ,MySql

CHAPTER 5

PROJECT PLAN

5.1 PROJECT ESTIMATES

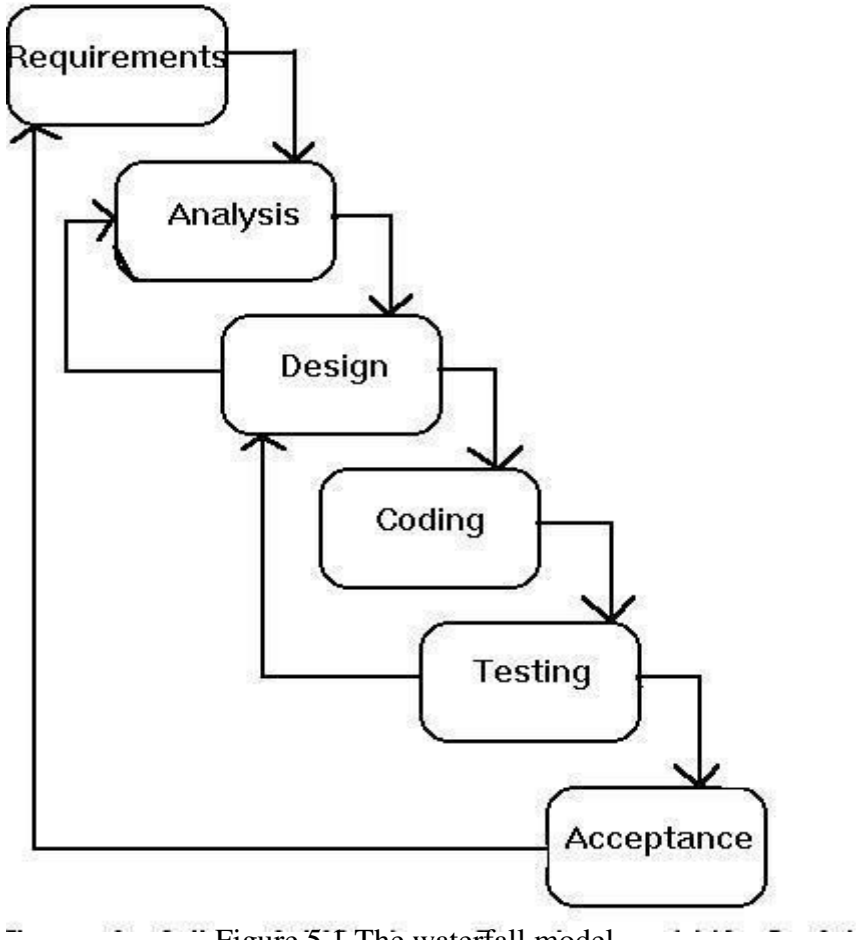


Figure 5.1 The waterfall model

5.1.1 Reconciled Estimates

5.1.1.1 Cost Estimate

To crawl million wed pages we need powerful machine, which is not possible for normal pc or laptop .As College is providing us access of PARAM supercomputer, our cost has decreased.

5.1.1.2 Time Estimates

Spider rate is around 8 pages per second which is 691,200 pages per day .We will need 360 machine working/crawling hours to index our target amount of pages that is 10 million pages.

5.1.2 Project Resources

With access to PARAM supercomputer with hardware configuration as 64 GB RAM, 9TB HDD, 28 MBPS internet speed, 64 BIT, Ubuntu. We are using IDE such as PYcharm for our python coding and Notepad ++ for font end programming.

5.2 RISK MANAGEMENT W.R.T. NP HARD ANALYSIS

Time is the risk that is associated with our project .As we have discussed above if we fail to index our targeted page in given time that means time will be a big factor that can cause failure of this project .To manage this risk we are working on our back end, so that when we start indexing our pages we can achieve our intended target without error.

5.2.1 Risk Identification

As discussed above, before risk identification is necessary before we can manage that risk .As in our project case we have to identify our problems related to our project .In our case one such identified problem is time and speed .By identifying this problem and analyzing this problem we can be sure that risk does not lead to failure.

5.2.2 Risk Analysis

The risks for the Project can be analyzed within the constraints of time and quality

ID	Risk Description	Probability	Impact		
			Schedule	Quality	Overall
1	Time	High	High	Low	Medium
2	Disk Size	Medium	Low	High	High

Table 5.1: Risk Table

Probability	Value	Description
High	Probability of occurrence is	> 50%
Medium	Probability of occurrence is	26−75%

Low	Probability of occurrence is	< 25%

Table 5.2: Risk Probability definitions

Impact	Value	Description
Low	5−10%	Run out of disk space
Medium	< 5%	Time management

Table 5.3: Risk Impact definitions

5.2.3 Overview of Risk Mitigation, Monitoring, Management

Following are the details for each risk.

Risk ID	1
Risk Description	Crawling
Category	Development Environment.
Source	Software requirement Specification document.
Probability	Low
Impact	High
Response	Mitigate
Strategy	Strategy
Risk Status	Occurred
Risk ID	2
Risk Description	Indexing
Category	Requirements

Source	Software Design Specification documentation review.
Probability	Low
Impact	High
Response	Mitigate
Strategy	Better testing will resolve this issue.
Risk Status	Identified

5.3 PROJECT SCHEDULE

5.3.1 Project task set

Major tasks in project stages are:

- Task 1: Crawling
- Task 2: Indexing
- Task 3: Searching

Risk ID	3
Risk Description	Searching
Category	Technology
Source	This was identified during early development and testing.
Probability	Low
Impact	Very High
Response	Accept
Strategy	Example Running Service Registry behind proxy balancer
Risk Status	Identified

5.3.2 Timeline Chart

Please refer Annex C for the planner.

5.4 TEAM ORGANIZATION

5.4.1 Team structure

Roles are defined and distributed in our team. As there are two stages of project, front end and back end. Two members are working on front end and other two handling the process of back end.

5.4.2 Project implementation

1) The crawler- This is the part that goes through the web, grabs the pages, and stores information about them into some central data store. In addition to the text itself, our project will need things like links , titles etc. The crawler needs to be smart enough to know how often to hit certain domains, to obey the robots.txt convention, etc**.**

2) The parser- This reads the data fetched by the crawler, parses it, saves whatever metadata it needs to, throws away junk, and possibly makes suggestions to the crawler on what to fetch next time around.

3) The indexer- Reads the stuff the parser parsed, and creates inverted indexes into the terms found on the webpages. It can be as smart as we want it to be -- we can apply NLP techniques to make indexes of concepts, cross-link things, throw in synonyms, etc.

4) The ranking engine- Given a few thousand URLs matching "apple", how do we decide which result is the best? Just the index doesn't give you that information. So we will need to analyze the text, the linking structure, and whatever other pieces we want to look at, and create some scores. This may be done completely on the fly, or based on some pre-computed notions of "experts".

5) The front end- The front end needs to receive user queries, hit the central engine, and respond; this The front end needs to be smart about caching results, possibly mixing in results from other sources, etc. It has its own set of problems.

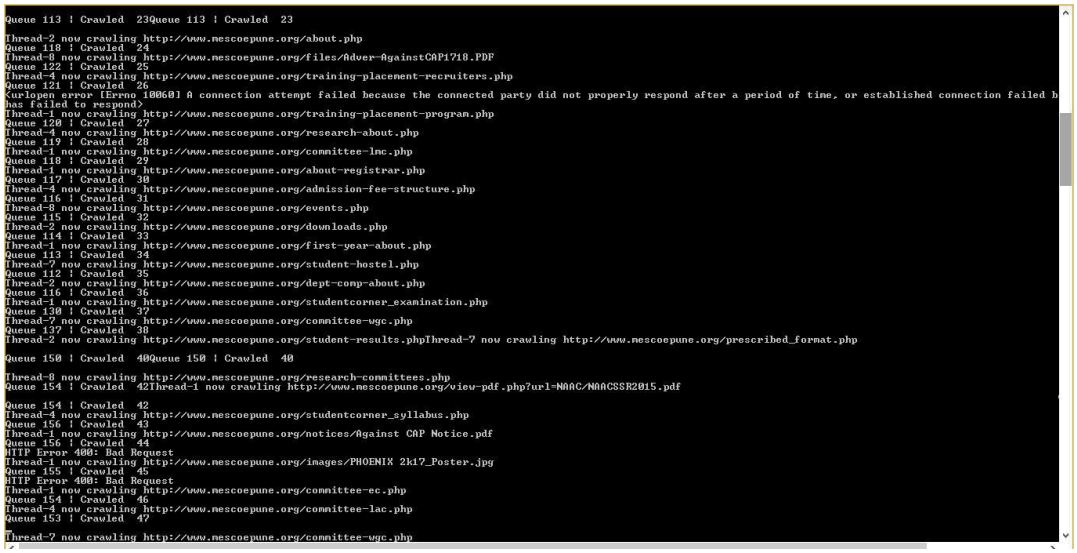


Figure 5.2 Crawler running on PARAM terminal

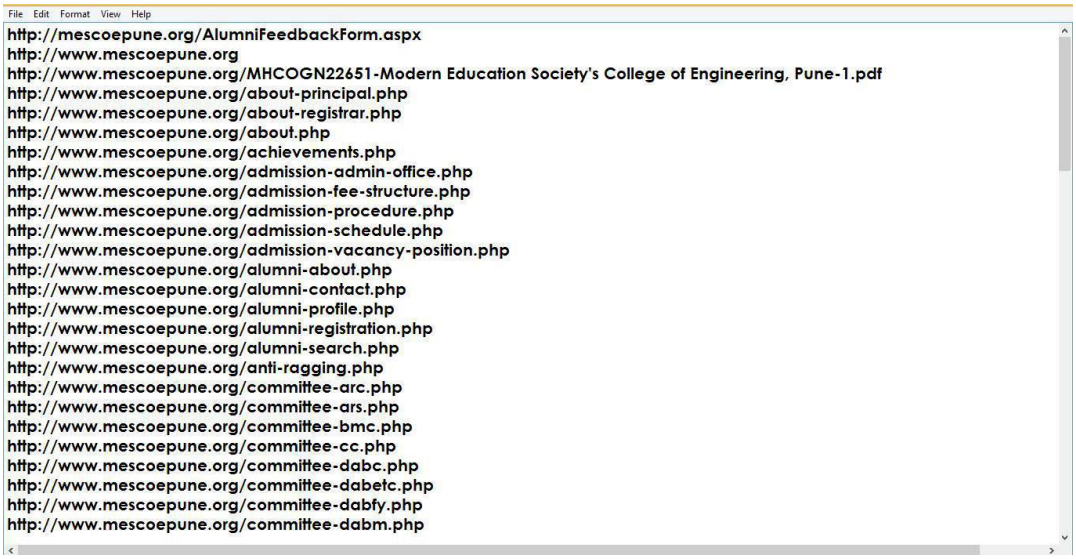


Figure 5.3 Crawled links

5.4.3 Management reporting and communication

The team members hold a weekly report about project progress reporting, this is how our mechanism for progress reporting works. Our team communication is identified as per assessment sheet and lab time table.

CHAPTER 6

SOFTWARE REQUIREMENT SPECIFICATION (SRS IS TO BE PREPARED USING RELEVANT MATHEMATICS DERIVED AND SOFTWARE ENGG. INDICATORS IN ANNEX A AND B)

6.1 INTRODUCTION

6.1.1 Purpose and Scope of Document

The purpose of SRS is to build the reliable and efficient crawler for search engine .We are working on IDE environment. For coding purpose we are using python language and running the code in Pycharm. For front end design we are using Notepad++ Editor to design the front end and programming languages like html, javascript and css.

6.1.2 Overview of responsibilities of Developer

Developer plays a vital role in our project. Developer has responsibilities to build the Front End Program as well as Back End Program. Developer is working with many challenges like the program must be space complexity as well as Time complexity to gain the speed of crawler and after the crawler crawls the web its should store the retrieval page in minimum amount of memory space. Core algorithm, Conduct end-to-end analysis that includes data gathering and requirements specification, processing, analysis, ongoing deliverables, and presentations.

6.2 USAGE SCENARIO

User will interact with the front end system i.e web Browser. User access the search engine which has adequate features like Language select option accordingly the virtual keyboard will popup, Videos, Images, News relevant data to the query given by the user.User can search the query in any language over all the globe and best relevant result will display on the user screen.

6.2.1 User profiles

The User who has access to the internet can interact with our system. Any user can use the search engine.

6.2.2 Use-cases

Sr. No	User	Description	Use case
1.	Scientist	Related content retrieval to research	Research
2.	Public	Information retrieval over the web	Searching
3.	Student	Knowledge base search	browsing
4.	Doctor	Medical Base search	Research

Table 6.1: Use Case

6.2.3 Use Case View

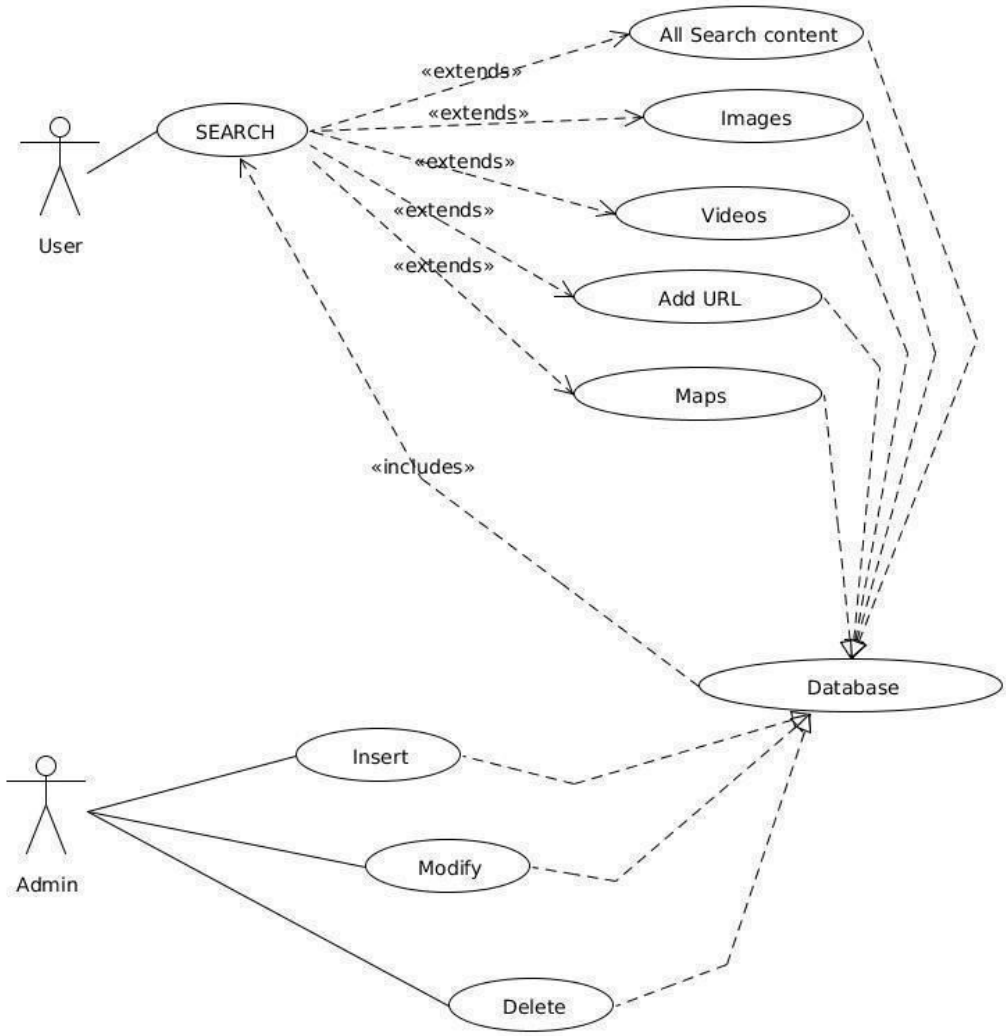


Figure 6.1: Use case diagram

6.3 DATA MODEL AND DESCRIPTION

6.3.1 Data Description

Data Model Design is an important foundation for the building and maintenance of high quality search systems, or business insight applications based on unstructured big data. Database attributes are Title of the webpage, URL of the page , Metadata Description, Text on the webpage, Links on the webpage and so on.

6.3.2 Data objects and Relationships

Data objects has various relationship between the attributes. Entity-Relation objects like User, Images, Videos, All search content, Languages, News etc. User is an entity which is related to the Images, Videos, All query content. When user is searching for the query in the web browser.

The Image has consists of Format, ID, Size and Keywords. Likewise Videos also has format, ID, keywords and so on.

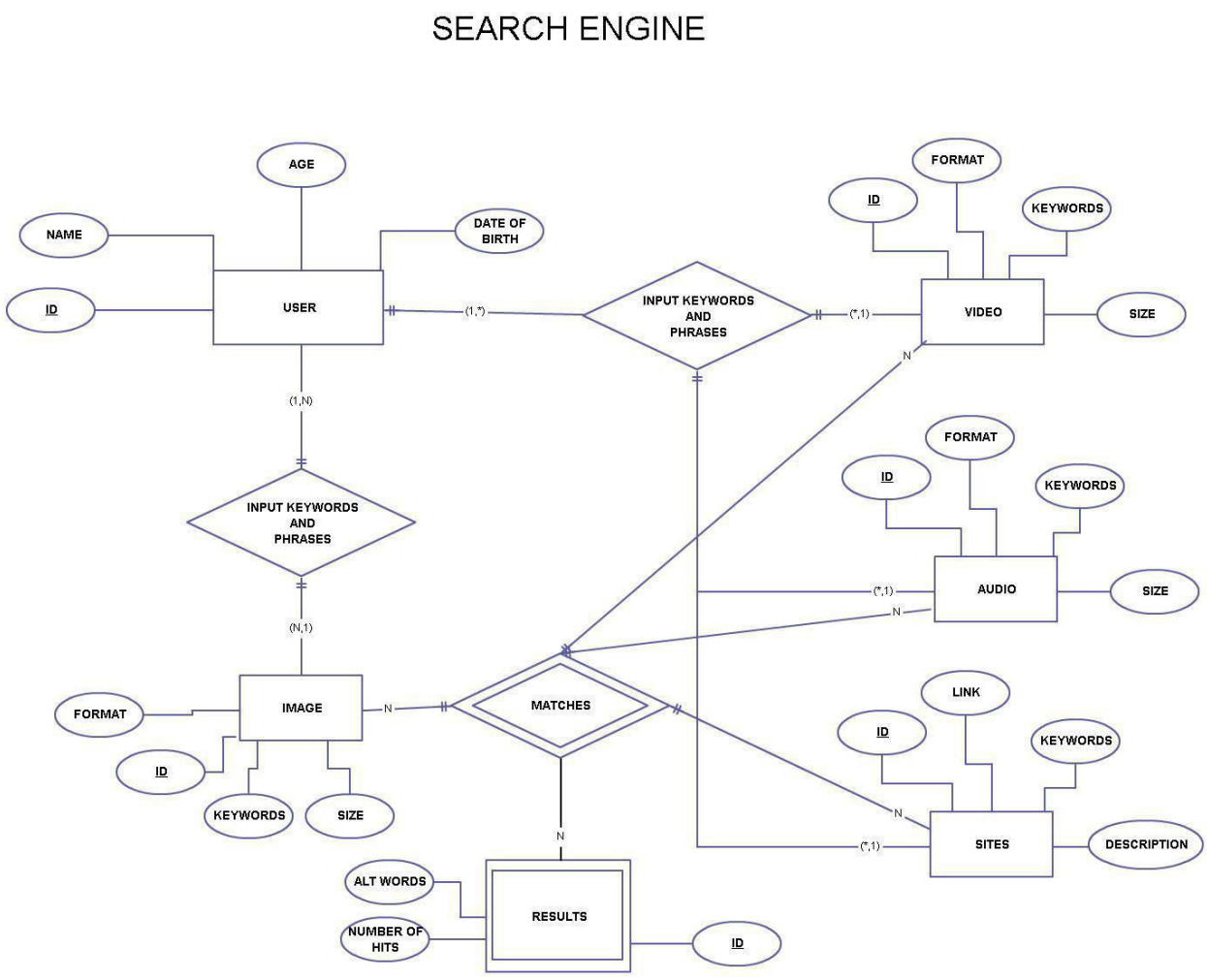


Figure 6.2: E-R Diagram of Search engine

6.4 FUNCTIONAL MODEL AND DESCRIPTION

6.4.1 Data Flow Diagram

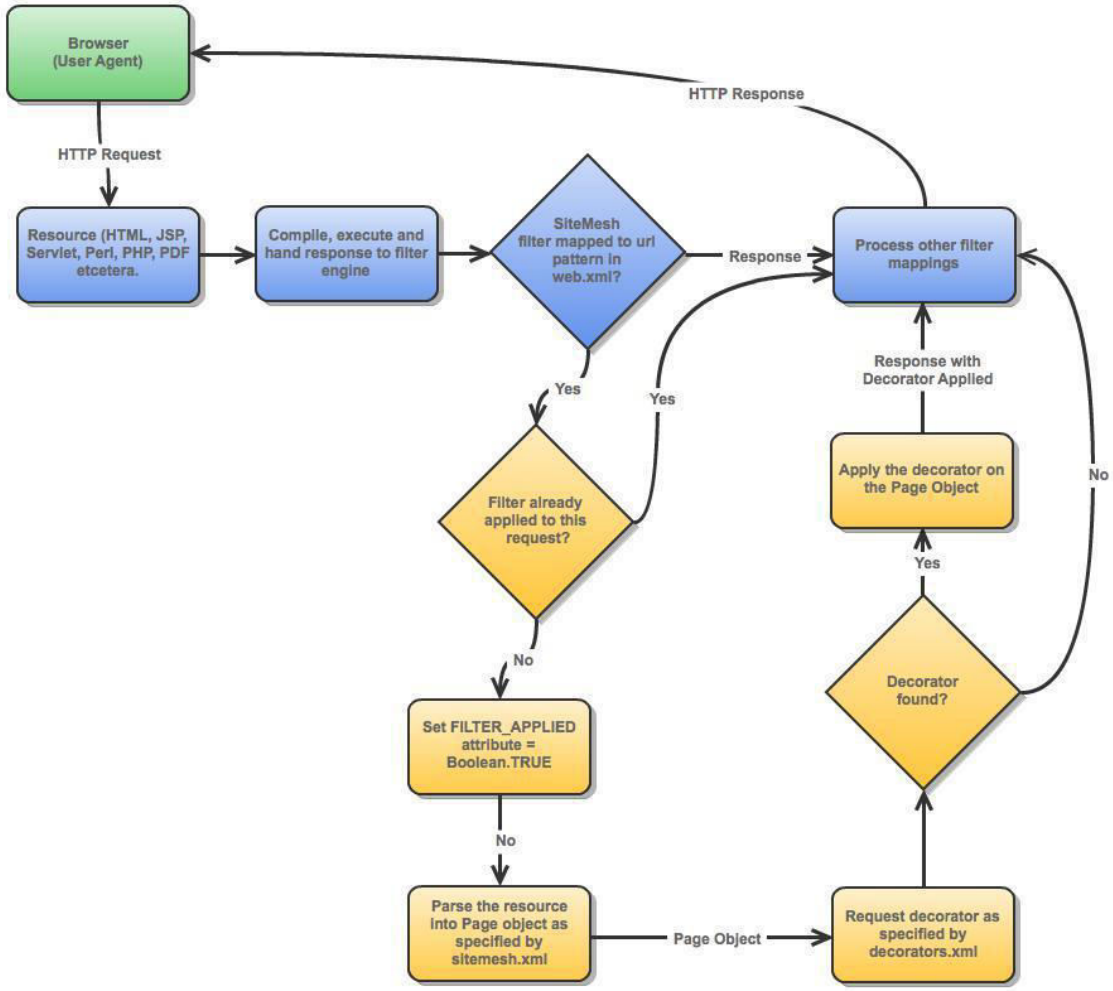


Figure 6.3: Data Flow Diagram of Search Engine

6.4.2 Activity Diagram:

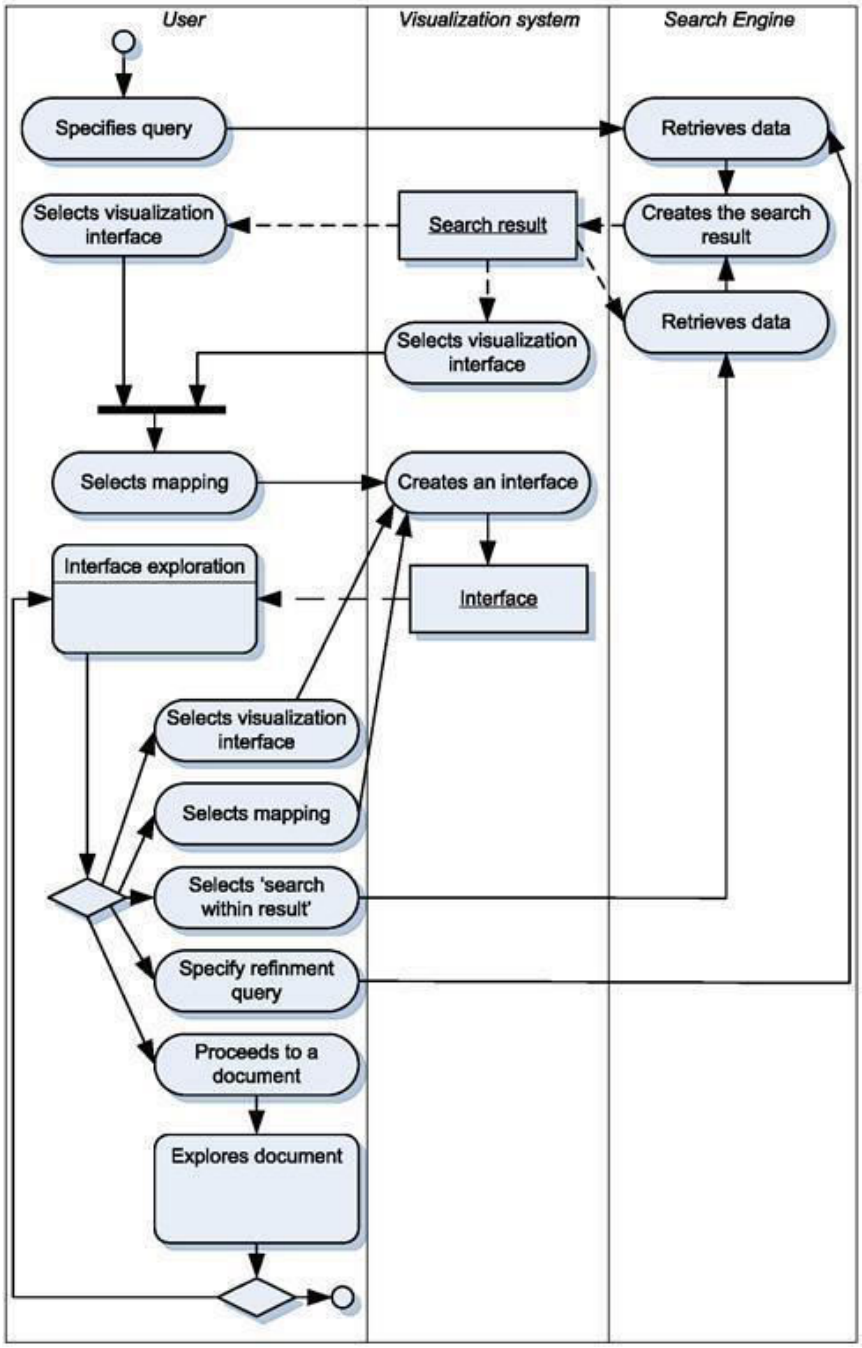


Figure 6.4: Activity Diagram

6.4.3 Non Functional Requirements:

6.4.3.1 Interface Requirements

Interface should be clean and understandable .There should be contrast in between colors used on user interface page.

6.4.3.2 Performance Requirements

We Should build the runtime system that gets users' queries, retrieves the results out of the index from the right machine(s), and re-ranks them according to the query.

6.4.3.3 Software quality attributes

Our software should be mobile responsive and can be used on any of Operating System .So our software should be portable, scalable, with good performance on various platforms.

6.4.4 State Diagram

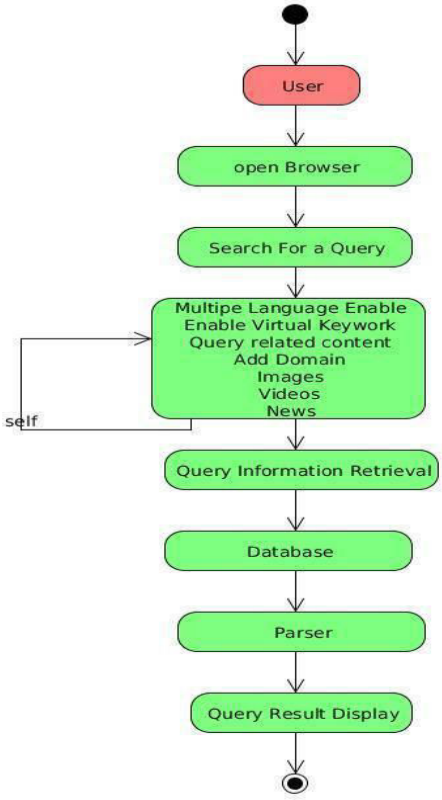


Figure 6.5: State diagram

6.4.5 Design Constraints

As our system is strong so no constraint can hit our system.

6.4.6 Software Interface Description

Our Software Interface should be clean and understandable. There should be contrast in between colors used on user interface page

CHAPTER 7

DETAILED DESIGN DOCUMENT

USING APPENDIX A AND B

7.1 INTRODUCTION

7.1.1 Front End

Front End side is visible to the user. Where user can search the query, retrieval information from the web.

7.1.2 Back End

Back End is a database side. Where the crawler retrieval the web pages all over the globe and store in the database in the structural database respectively.

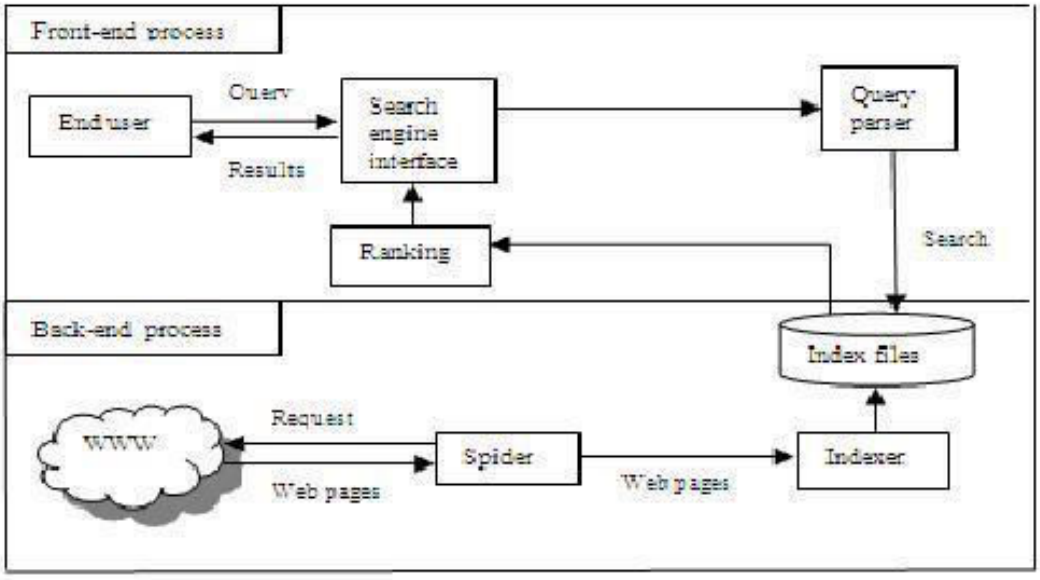


Figure 7.1 Back end and front end process

7.2 ARCHITECTURAL DESIGN

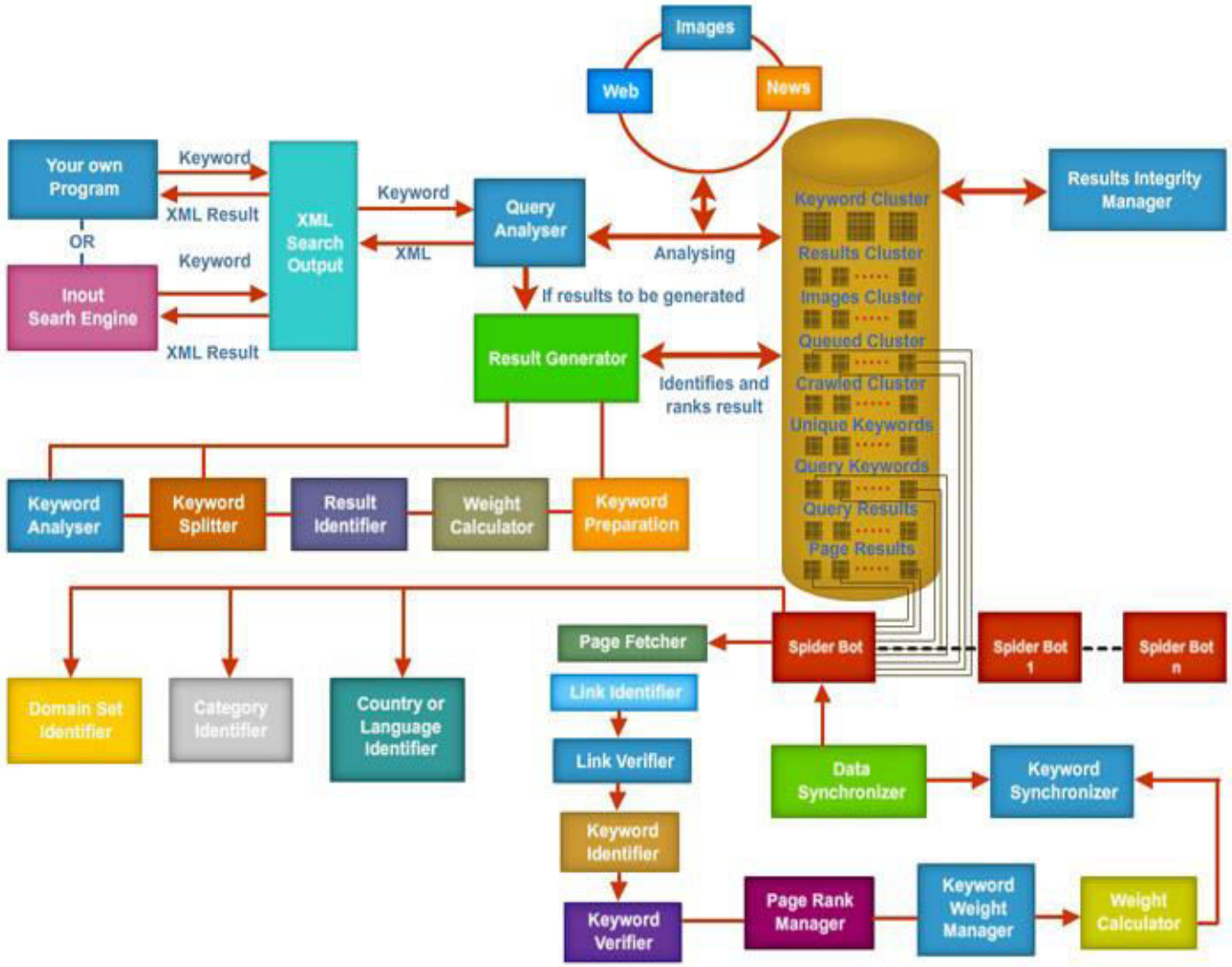


Figure 7.2 Search engine architecture

7.3 DATA DESIGN

7.3.1 Internal software data structure

To characterize an internal software-level resource container abstraction, we need additional internal variables.

7.3.2 Global data structure

To fully characterize a system-level resource container abstraction, we need additional global variables such as:

1. se_ root:

This is the root container. All the resources in the system are allocated to the root container and hence to the hierarchy rooted here. Any container or task that has to get a resource has to be in the tree rooted here.

2. se_ index:

This is the index container. All the resources in the database are allocated to the index container and hence to the hierarchy rooted here. Any container or task that has to access a index has to be in the tree rooted here.

7.3.3 Database description

Database which holds index of crawled data is created which contains all required things from a web page entity. It include table which holds- Title, Keyword, Description, Metadata, URL, Links and Text Content on the webpage are created in an database.

7.4 COMPONENT DESIGN

7.4.1 Class Diagram of Search Engine

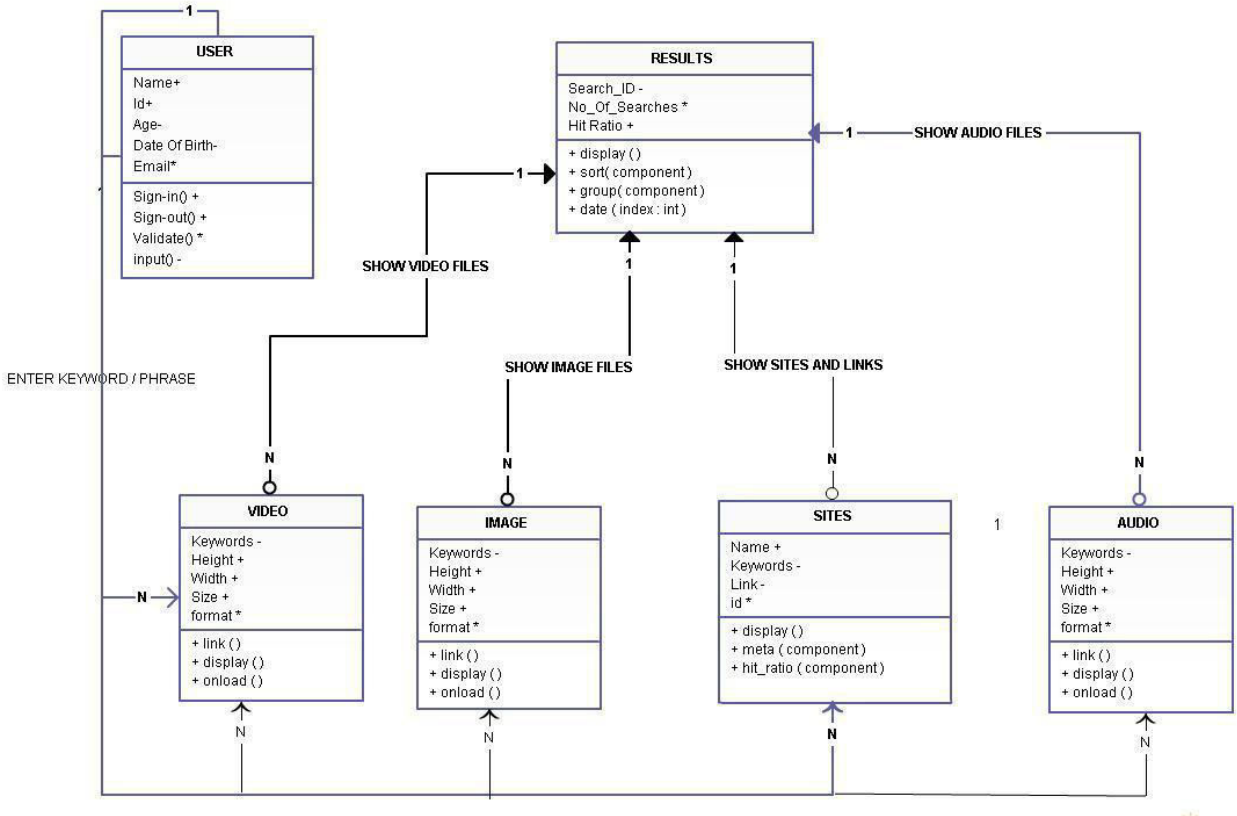


Figure 7.3: Class Diagram

CHAPTER 8

SUMMARY AND CONCLUSION

So now we take a look at summary report of the project. This project will result into a web based search engine than can handle the user queries and present them in the form relevant links to the query.

In this report, we have taken on the audacious task of condensing every page on the World Wide Web into a single number, given by calculating its value. This is a global ranking of all web pages, regardless of their content, based solely on their location in the Web's graph structure. Using it, wear able to order search results so that more important and central Web pages are given preference.

In experiments, this turns out to provide higher quality search results to users. The intuition behind it is that it uses information which is external to the Web pages themselves - their back links, which provide a kind of peer review. Furthermore, back links from "important" pages are more significant than back links from average pages. This is encompassed in the recursive definition of value. This value could be used to separate out a small set of commonly used documents which can answer most queries. The full database only needs to be consulted when the small database is not adequate to answer a query. Finally, this may be a good way to help and representative pages to display for a cluster center. We have found a number of applications for this project in addition to search which include traffic estimation, and user navigation.

References:

[1] Best of the Web 1994 – Navigators http://botw.org/1994/awards// navigators.html

[2]Bill Clinton Joke of the Day: April 14, 1997. http://www.io.com / cjburke/clinton/970414.html.

[3] Mauldin, Michael L. Lycos Design Choices in an Internet Search Service, IEEE Expert Interview http://www.computer.org/pubs/expert/ 1997/trends/x1008/mauldin.htm

[4] MCho 98] Junghoo Cho, Hector Garcia-Molina, Lawrence Page. Efficient Crawling Through URL Ordering. Seventh International Web Conference (WWW 98). Brisbane, Australia, April 14-18, 1998.

[5] Use Upon Driver Attention http:// www.webfirst.com/aaa/text/cell/cell0toc.htm

[6] Exclusion Protocol: http://info.webcrawler.com/mak/ projects/robots/exclusion.htm

[7] [Abiteboul 97] Serge Abiteboul and Victor Vianu, Queries and Computation on the Web. Proceedings of the International Conference on Database Theory. Delphi, Greece 1997.

[8] [Gravano 94] Luis Gravano, Hector Garcia-Molina, and A. Tomasic. The Effectiveness of GlOSS for the Text-Database Discovery Problem. Proc. of the 1994 ACM SIGMOD International Conference On Management Of Data, 1994.

[9] [McBryan 94] Oliver A. McBryan. GENVL and WWWW: Tools for Taming the Web. First International Conference on the World Wide Web. CERN, Geneva (Switzerland), May 25-26-27 1994. http:// www.cs.colorado.edu/home/mcbryan/mypapers/www94.ps

[10] [Pinkerton 94] Brian Pinkerton, Finding What People Want: Experiences with the WebCrawler. The Second International WWW Conference Chicago, USA, October 17-20, 1994. http://info.webcrawler. com/bp/WWW94.html

[11] [TREC 96] Proceedings of the fifth Text REtrieval Conference (TREC-5). Gaithersburg, Maryland, November 20-22, 1996. Publisher: Department of Commerce, National Institute of Standards and Technology. Editors: D. K. Harman and E. M. Voorhees. Full text at:

ANNEXURE A

LABORATORY ASSIGNMENTS ON PROJECT ANALYSIS OF ALGORITHMIC DESIGN

- To develop the problem under consideration and justify feasibility using concepts of knowledge canvas and IDEA Matrix. Refer [?] for IDEA Matrix and Knowledge canvas model. Case studies are given in this book. IDEA Matrix is represented in the following form. Knowledge canvas represents about identification of opportunity for product. Feasibility is represented w.r.t. business perspective.

I	D	E	D
Increase: Speed of crawler.	Drive: Surf all over the web.	Educate:Give relevant result to user.	Accelerate: Apply algorithms to get relevant results.
Improve: The performance of parser.	Deliver: Relevant pages to user query.	Evaluate: Time for crawling.	Associate: User-friendly search engine.
Ignore: Stop words.	Decrease: Size of index.	Eliminate: Unnecessary information while parsing.	Avoid: Irrelevant web pages

Table B.1 IDEA Matrix

- **PROJECT PROBLEM STATEMENT FEASIBILITY ASSESSMENT USING NP-HARD, NP-COMPLETE OR SATISFIABILITY ISSUES USING MODERN ALGEBRA AND/OR RELEVANT MATHEMATICAL MODELS.**

- **Feasibility Study:**
  Feasibility study is the test of a system proposal according to its workability, impact on the organization, ability to meet user needs, and effective use of recourses. It focuses on the evaluation of existing system and procedures analysis of alternative candidate system cost estimates. Feasibility analysis was done to determine whether the system would be feasible. The development of a computer based system or a product is more likely plagued by resources and delivery dates. Feasibility study helps the analyst to decide whether or not to proceed, amend, postpone or cancel the project, particularly important when the project is large, complex and costly. Once the analysis of the user requirement is complement, the system has to check for the compatibility and feasibility of the software package that is aimed at. An important outcome of the preliminary investigation is the determination that the system requested is feasible.

- **Technical Feasibility:**

  The technology used can be developed with the current equipments and has the technical capacity to hold the data required by the new system. This technology supports the modern trends of technology. Easily accessible, more secure technologies. Technical feasibility on the existing system and to what extend it can support the proposed addition.

- **Operational Feasibility:**

  This proposed system can easily implemented, as this is based on Swing scoding. The database created is with MySQL server which is more secure and easy to handle. The resources that are required to implement/install these are available. The personal of the organization already has enough exposure to computers. So the project is operationally feasible.

- **Economical Feasibility:**

  Economic analysis is the most frequently used method for evaluating the effectiveness of a new system. More commonly known cost/benefit analysis, the procedure is to determine the benefits and savings that are expected from a candidate system and compare them with costs. If benefits outweigh costs, then the decision is made to

design and implement the system. An entrepreneur must accurately weigh the cost versus benefits before taking an action. This system is more economically feasible which assess the brain capacity with quick online test. So it is economically a good project.

- **NP-hard, NP-Complete Analysis:**

When solving problems we have to decide the difficulty level of our problem.
There are three types of classes provided for that. These are as follows:

1) P Class

2) NP-hard Class

3) NP-Complete Class

- **P Class:**

The class of polynomially solvable problems, P contains all sets in which membership may be decided by an algorithm whose running time is bounded by a polynomial. Besides containing all of what we have decided to consider practical computational tasks, the class P has another attractive attribute. Its use allows us to not worry about our machine model since all reasonable models of computation (including programs and Turing machines) have time complexities, which are polynomially related.

- **NP Class:**

Informally the class P is the class of decision problems solvable by some algorithm within a number of steps bounded by some fixed polynomial in the length of the input. Turing was not concerned with the efficiency of his machines, but rather his concern was whether they can simulate arbitrary algorithms given sufficient time. However it turns out Turing machines can generally simulate more efficient computer models (for example machines equipped with many tapes or an unbounded random access memory) by at most squaring or cubing the computation time. Thus P is a robust class and has equivalent definitions over a large class of computer models. Here we follow standard practice and define the class P in terms of Turing machines.

- **NP Hard:**

A problem is NP-hard if solving it in polynomial time would make it possible to solve all problems in class NP in polynomial time. Some NP-hard problems are also in NP (these are called "NP-complete"), some are not. If you could reduce an NP problem to an NP-hard problem and then solve it in polynomial time, you could solve all NP

problems. Also, there are decision problems in NP-hard but are not NP-complete, such as the infamous halting problem.

- **NP-complete:**

  A decision problem L is NP-complete if it is in the set of NP problems so that any given solution to the decision problem can be verified in polynomial time, and also in the set of NP hard problems so that any NP problem can be converted into L by a transformation of the inputs in polynomial time. The complexity class NP-complete is the set of problems that are the hardest problems in NP, in the sense that they are the ones most likely not to be in P. If you can find a way to solve an NP-complete problem quickly, then you can use that algorithm to solve all NP problems quickly.

- **Conclusion:**

  Thus we can conclude that our topic comes under NP class because the steps required to find the relevant pages based on user query can be complete after following some fixed algorithmic steps. The steps may vary according to the query keywords. So our topic comes only in NP class.

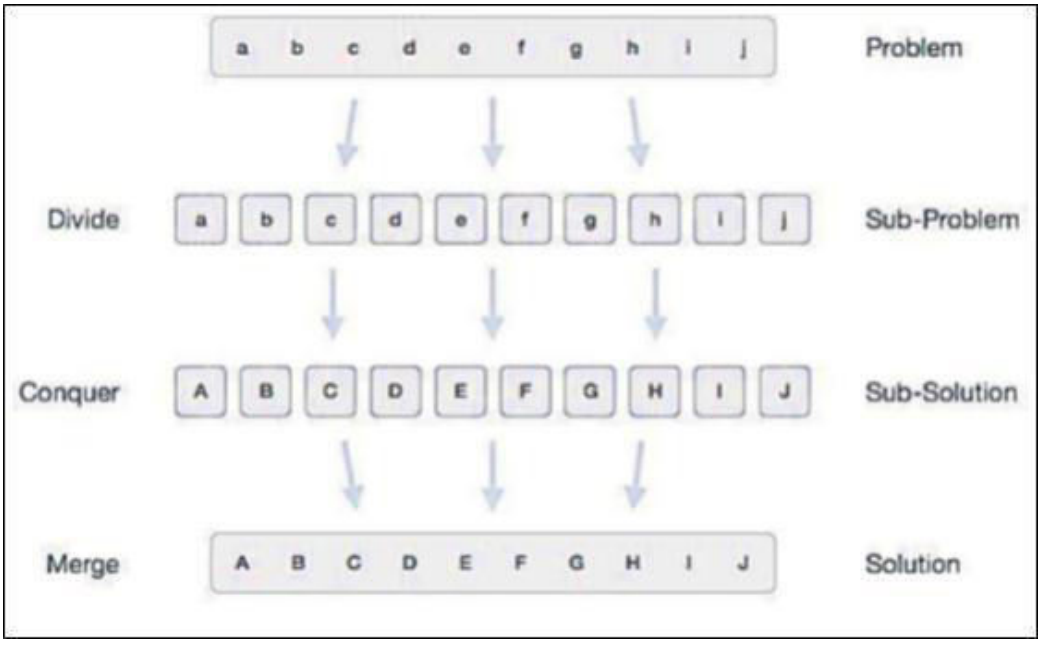
ANNEXURE B

LABORATORY ASSIGNMENTS ON PROJECT QUALITY AND RELIABILITY TESTING OF PROJECT DESIGN

B.1 USE OF DIVIDE AND CONQUER STRATEGIES TO EXPLOIT DISTRIBUTED/ PARALLEL / CONCURRENT PROCESSING OF THE ABOVE TO IDENTIFY OBJECT, MORPHISMS, OVERLOADING IN FUNCTIONS (IF ANY), AND FUNCTIONAL RELATIONS AND ANY OTHER DEPENDENCIES (AS PER REQUIREMENTS). IT CAN INCLUDE VENN DIAGRAM, STATE DIAGRAM, FUNCTION RELATIONS, I/O RELATIONS; USE THIS TO DERIVE OBJECTS MORPHISM ,OVERLOADING

Use of divide and conquer strategies to exploit distributed/parallel/concurrent processing.

Divide and conquer (D&C) is an algorithm design paradigm based on multi-branched recursion. A divide and conquer algorithm works by recursively breaking down a problem into two or more sub-problems of the same (or related) type (divide), until these become simple enough to be solved directly (conquer). The solutions to the sub-problems are then combined to give a solution to the original problem.



B.2 USE THE ABOVE TO DRAW FUNCTIONAL DEPENDENCY GRAPHS AND RELEVANT SOFTWARE MODELLING METHODS, TECHNIQUES INCLUDING UML DIAGRAMS OR OTHER NECESSITIES USING APPROPRIATE TOOLS.

UML Overview:

UML (Unified Modeling Language) is a general purpose modeling language. UML provides elements and components to support the requirement of complex systems. UML follows the object oriented concepts and methodology. UML diagrams are drawn from different perspectives like design, implementation, deployment etc. At the conclusion UML can be defined as a modeling language to capture the architectural, behavioral and structural aspects of a system. The UML has an important role in this Object Oriented analysis and design, The UML diagrams are used to model the design. So the UML has an important role to play.

UML notations:

UML notations are the most important elements in modeling. Efficient and appropriate use of notations is very important for making a complete and meaningful model. The model is useless unless its purpose is depicted properly. So learning notations should be emphasized from the very beginning. Different notations are available for things and relationships. And the UML diagrams are made using the notations of things and relationships. Extensibility is another important feature which makes UML more powerful and flexible

UML Diagrams:

Diagrams are the heart of UML. These diagrams are broadly categorized as structural and behavioral diagrams.

- Structural diagrams are consists of static diagrams like class diagram, object diagram etc.
- Behavioral diagrams are consists of dynamic diagrams like sequence diagram, collaboration diagram etc.

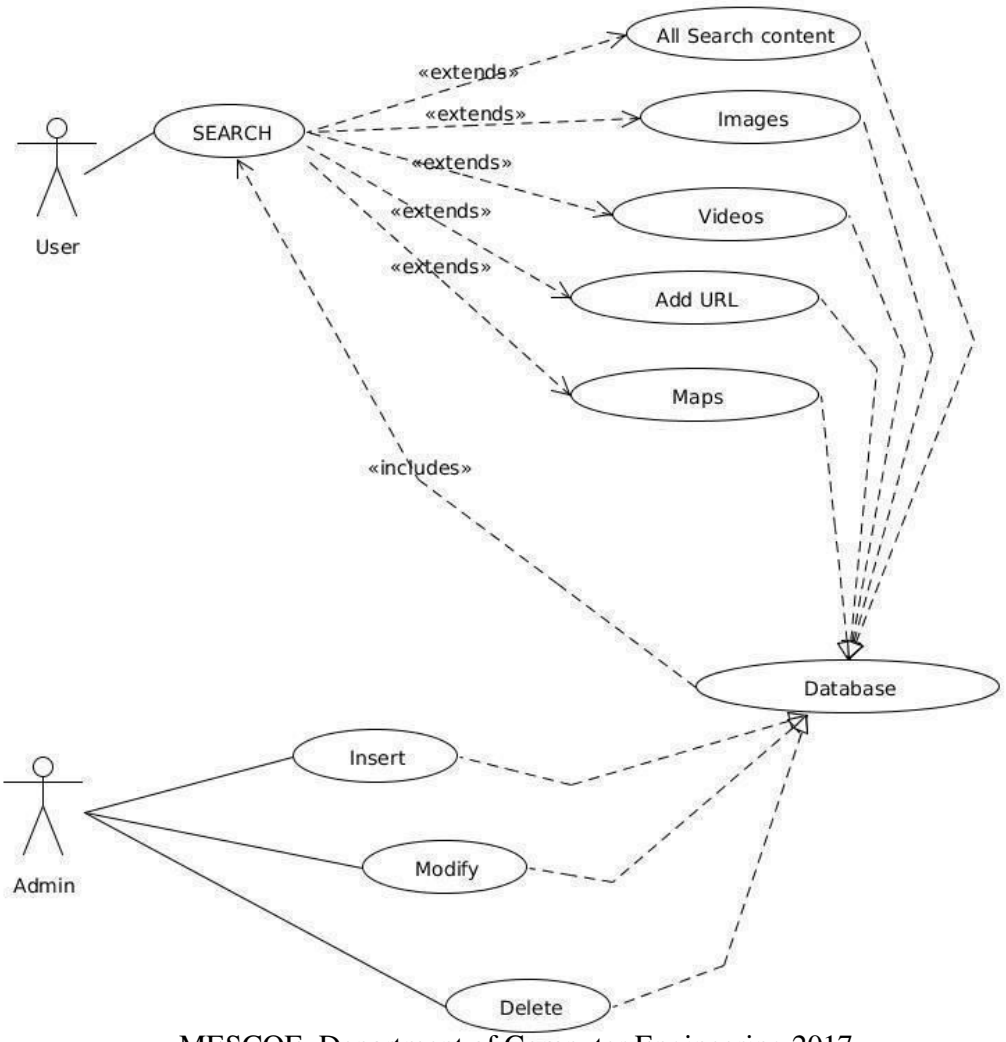
The static and dynamic nature of a system is visualized by using these diagrams

UML Tools:

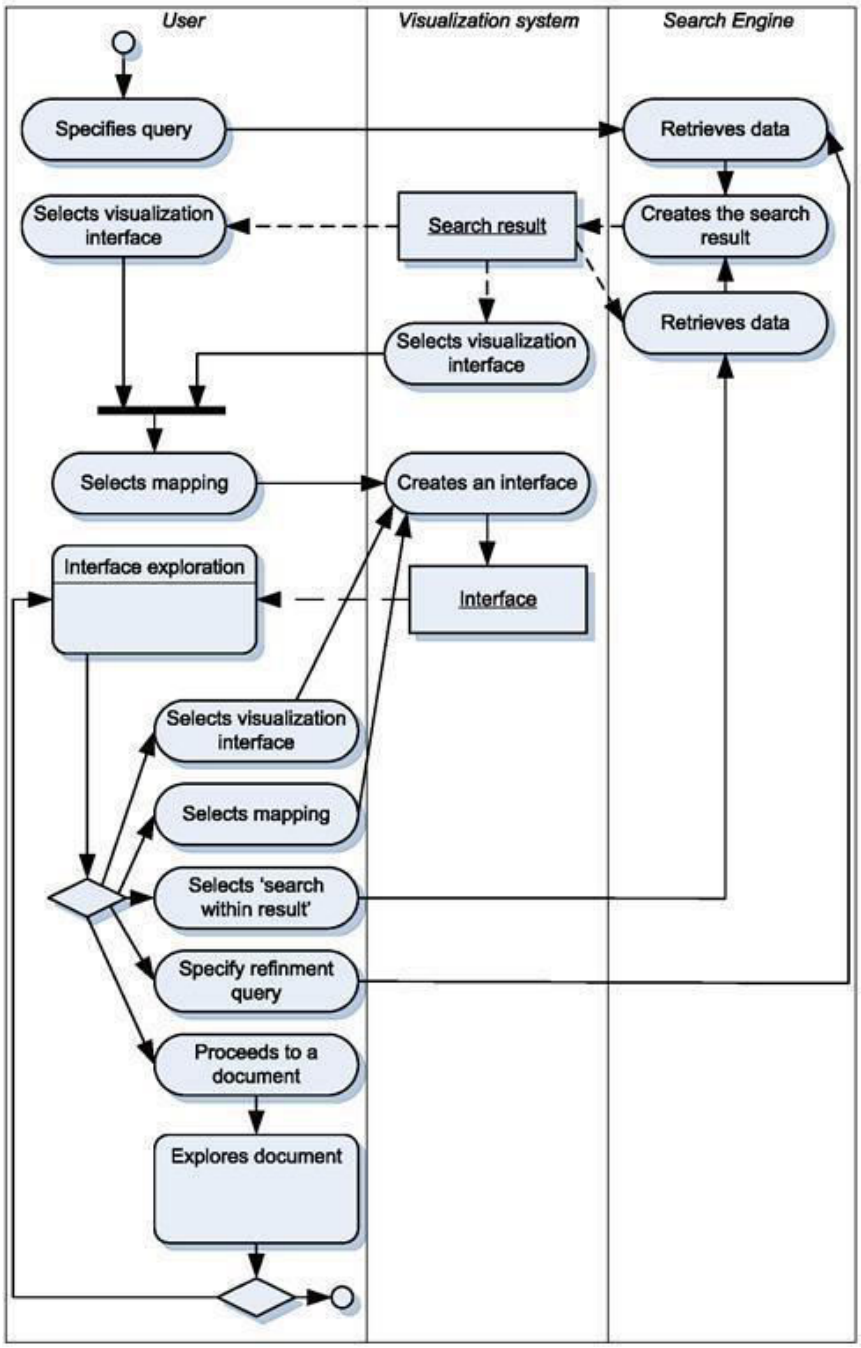
- StarUML - StarUML is an open source project to develop fast, flexible, extensible, featureful, and freely-available UML/MDA platform running on Win32 platform.
- ArgoUML - ArgoUML is the leading open source UML modeling tool and includes support for all standard UML diagrams.
- Umbrello UML Modeller - Umbrello UML Modeller is a Unified Modelling Language diagram program for KDE.
- Acceleo - Acceleo is easy to use. It provides off the shelf generators (JEE, .Net, php...) and template editors for Eclipse.
- GenMyModel - An online UML modeling tool.

UML Diagrams:

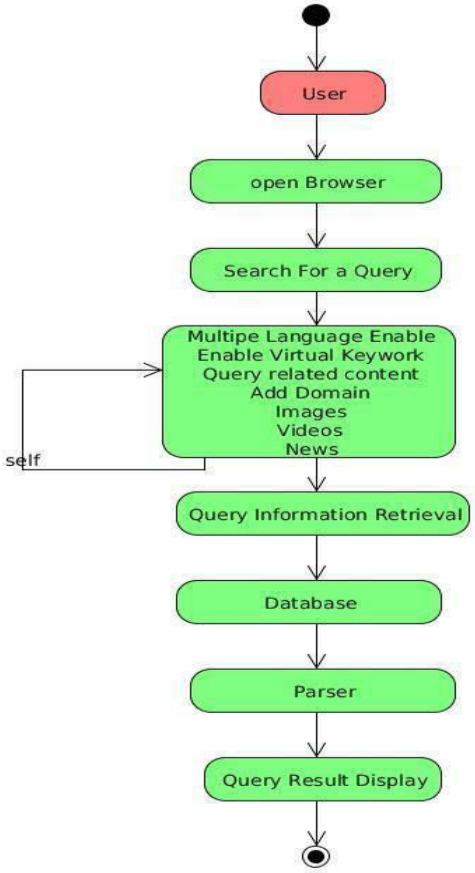
1. **Use case diagram**



2. Activity Diagram



3. State diagram



SOFTWARE MODEL:

Waterfall Model:

The Waterfall Model was first Process Model to be introduced. It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed fully before the next phase can begin. This type of model is basically used for the for the project which is small and there are no uncertain requirements. At the end of each phase, a review takes place to determine if the project is on the right path and whether or not to continue or discard the project. In this model the testing starts only after the development is complete. In waterfall model phases do not overlap.

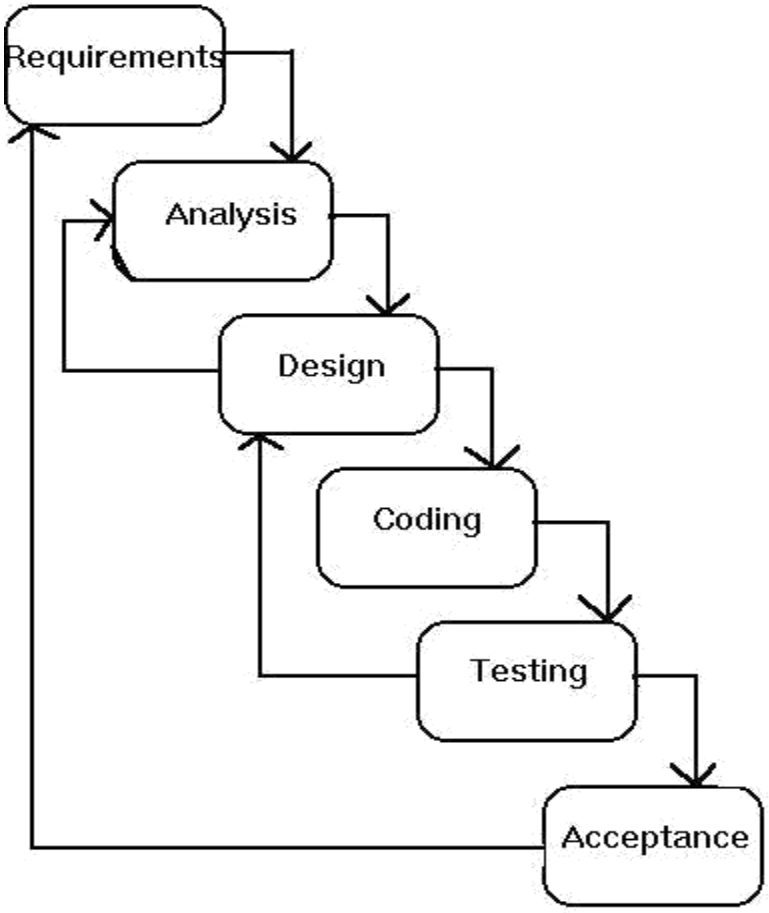


Figure. The waterfall model

Advantages:

- This model is simple and easy to understand and use.
- It is easy to manage due to the rigidity of the model – each phase has specific deliverables and a review process.
- In this model phases are processed and completed one at a time. Phases do not overlap.
- Waterfall model works well for smaller projects where requirements are very well understood

B.2 TESTING OF PROJECT PROBLEM STATEMENT USING GENERATED TEST DATA (USING MATHEMATICAL MODELS , GUI , FUNCTION TESTING PRINCIPLES , IF ANY) SELECTION AND APPROPRIATE USE OF TESTING TOOLS, TESTING OF UML DIAGRAM'S RELIABILITY. WRITE ALSO TEST CASES [BLACK BOX TESTING] FOR EACH IDENTIFIED FUNCTIONS.

SOFTWARE TESTING

Software testing is an activity aimed at evaluating an attribute or capability of a program or system and determining that it meets its required results. It is more than just running a program with the intention of finding faults. Every project is new with different parameters. No single yardstick maybe applicable in all circumstances. This is a unique and critical area with altogether different problems. Although critical to software quality and widely deployed by programs and testers. Software testing steel remains an art, due to limited understanding of principles of software. The difficulty stems from complexity of software. The purpose of software testing can be quality assurance, verification and validation or reliability estimation. Testing can be used as a generic metric as well. Software testing is a trade-off between budget, time and quality.

TESTING PROCCESS:

It is an important phase. We execute the program with given inputs and note down the outputs. These are compared with expected outputs. If matched then the program is said to be as per user specification else there is something wrong.

LEVELS OF TESTING:

1) Debug: It is defined as the successful correction of a failure

2) Demonstrate: The process of finding major features work with typical inputs.

3) VERIFY: It is the process of finding faults in the requirements, Design.

4) Validate: It is the process of finding as many faults in the requirement and design.

5) Prevent: To avoid errors in the development of requirements, design and implementation.

PRINCIPLES OF TESTING:

1) Testing should be based on user requirements.
2) Testing time and recourses are limited.
3) Exhaustive time is impossible.
4) Use effective resources to test.
5) Test planning should be done early.
6) Testing should begin in small and progress towards the testing in large.

VALIDATION AND VERIFICATION:

1. Software Verification:

It is the process of evaluation a system or component to determine whether the product of given development phase satisfy the condition imposed at the start of the phase.

1) It is a static process
2) It does not involve any code and is human based checking.
3) It uses methods like inspections, walk through, desk checking etc.
4) It can catch errors that validations cannot.

2. Software Validation:

It is the process of evaluation a system or component during or at the end of the development process to determine to determine whether it specifies the specified requirements. It involves executing the actual software. it is a computer based testing process.

1) It is a dynamic process.
2) It involves executing of code as well as human based execution of program.
3) It uses methods like Black box and white box testing.
4) It can catch errors that verification cannot catch.

Note that verification and validation (V&V) are complementary to each other.

3. Software Verificatin And Validation(V&V):

V&V is a technical discipline of system engineering. Software V&V is a system engineering process employing a rigorous methodology for evaluating the correctness and quality of software product through the software life cycle.

TYPES OF SOFTWARE TESTING:

1. Black Box Testing:

The term black box refers to the software which is treated as a black box. By treating it as a black box, we mean that the system or the source code is not checked at all. It is done from customers view point. The test engineer in black box testing only knows the set of inputs and the expected outputs and is unaware how those inputs are transformed into outputs by the software.

2. White Box Testing:

White box testing is a way of testing the external functionality of the code by examining and testing the program code that realizes the external functionality. It is a methodology to design test cases that uses the control structure of the application to design the test. White box testing is used to test the program code, code structure and internal design flow.

White box testing types:
1) Static white box testing
2) Dynamic white box testing.

3. Gray Box Testing:

Gray box testing consists of methods and tools derived from the knowledge of the application internals and the environment with which it interacts, that can be applied in black box testing to enhance testing productivity, bug finding and bug analyzing efficiency. It incorporates the elements of both black box as well as white box testing. It considers the outcome on the user end, system specific knowledge and the operating system.

Unit Testing:

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration Testing:

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration testing is specifically aimed at exposing the problems that arise from the Combination of components.

System Testing:

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results.

ANNEXURE C

PROJECT PLANNER

Work task	**Description**	**Duration**
Detail Design	Create Detailed design for project	1 week
Planning and Dataset	Modeling and dataset searching or creation	3 weeks
Algorithms	Detail study of algorithms	2 weeks
PageRank	Detail study of algorithm	2 weeks
Implementation	Divided into phases	
Phase A,B,C…….	Implementation of modules	10 weeks
System Testing	Test system quality , fix errors if any and improved if needed. Test system for different datasets.	3 weeks
Final Report	Prepare and upload initial report	2 weeks

ANNEXURE D

REVIEWERS COMMENTS OF PAPER SUBMITTED

(At-least one technical paper must be submitted in Term-I on the project design in the conferences/workshops in IITs, Central Universities or UoP Conferences or equivalent International Conferences Sponsored by IEEE/ACM)

1. Paper Title:
2. Name of the Conference/Journal where paper submitted :
3. Paper accepted/rejected :
4. Review comments by reviewer :
5. Corrective actions if any :

ANNEXURE E

PLAGIARISM REPORT