Chapter 1

INTRODUCTION

1.1 General:

A search engine is an indispensable tool for every internet user. In an academic environment such as ours, where a large number of users work on related material, a search engine is a useful tool. Unfortunately, no free, open source, easily deployable and customizable search engine exists for intranet users. There are a few options available like 'Google's Enterprise Search' but they are neither free nor open-source and their features are tuned for business environment and not a large academic system like ours and several other universities.

In addition to this, most search engines do not have the functionality to search files which is a specially useful in case of LAN due to high file transfer speeds. Hence, the search engine we plan to develop will have two components, one being an intranet webpage search and the other being intranet file search.

File search would support popular file sharing protocol FTP and SMB. The administrator can specify the IP range of the intranet, which will be used by the crawler to create an index of the filenames shared by FTP servers and SMB servers in that address range.

Web search component would index web pages on the intranet based on a set of starting URLs that can be specified by the network administrator at the time of deployment. A crawler will start crawling the intranet recursively from these set of starting URLs. The administrator will be able to specify patterns (regular expressions) for urls that have to be indexed (whitelist) and those that have to be ignored (blacklist) in a configuration file for the crawler. The administrator will also be able to censor content being indexed. He/she

will be able to specify certain forbidden words, which will not be indexed by the indexing component.

1.2 Objective:

To provide a solution which aims at providing the users with a system that enables them to search and access various files and web pages shared over a local area network.

- **Speed**

  The system provides speedy access to files since it is working over the wired intranet connection. The downloading speed of files can be up to a maximum of 5 to 6 mbps depending on the server configuration and network traffic.

- **Efficient**

  The system is user-friendly and the user can easily understand its contents without much effort. The search engine returns results that are displayed in a form of a table.

- **Interactive**

  The system is interactive and the user can express all his/her needs easily and precisely.

Aims:

To develop a free and open source search engine that:

- Enables user to search and download files hosted over an FTP server in an intranet.
- Enables user to search and download files hosted over SMB share in an intranet.
- Enables user to keep a track of servers for any updates through dedicated intranet mailing system.
- Enables user to search web pages hosted over an intranet.

1.3 SYSTEM ANALYSIS:

1.3.1 Identification of need:

This is the phase during which the problem is identified, alternate system solutions are studied, and recommendations are made about committing the resources required to design the system.

We did the system study for this software in our college as well as by interaction with students of other colleges. The study involved the following steps:

- **Study of Present System**

Students used the sharing facility given by the Windows OS in order to share their files over the network. The main drawback of this system was that –

- The user needed to know the IP address of the system on which the file was stored in order to gain access.
- Secondly, if any user does not know whether the file is hosted on the network or not, there was no searching interface through which the needful could be done.

- **Requirement**

Therefore, after the study of present system we thought of developing a free, open source and easily deployable search facility that will enable the user to overcome the above mentioned problems and not only share but also search for the needed files without any prior knowledge of the IP addresses of the systems over the network.

1.3.2 Preliminary Investigation:

Our project *Loogle – Intranet Search Engine* aims at developing a search engine that facilitates the searching of various files and web pages hosted over an intranet. The user would be able to access files over the local area network without knowing the exact location of the system on which the file is stored.

Reason for selection:

A search engine is an indispensable tool for every internet user. In an academic environment such as ours, where a large number of users work on related material, a search engine is a useful tool. Unfortunately, no free, open source, easily deployable and customizable search engine exists for intranet users. There are a few options available like 'Google's Enterprise Search' but they are neither free nor open-source and their

features are tuned for business environment and not a large academic system like ours and several other universities.

In addition to this, most search engines do not have the functionality to search files which is a specially useful in case of LAN due to high file transfer speeds. Hence, the search engine we plan to develop will have two components, one being an intranet webpage search and the other being intranet file search.

1.3.3 Feasibility Study:

Once the system objectives have been ascertained by initial investigation, we need to spell the various possible solutions to meet the various objectives. The feasibility study is conducted to check whether the candidate system is feasible. The system, which is selected to be the best against the criteria, is thereafter designed and developed. The feasibility study takes into consideration, the risks involved in the project development beforehand.

Feasibility study includes several distinct but interrelated type of feasibility. These are:

1.3.3.1 Technical Feasibility:

Focus is on establishing whether the technology needed for the proposed system is available and how this technology can be integrated within the organization. Technologies included are:

- Hardware
- Software
- Application development environment

Our project is technically feasible as all of the above required components were available.

1.3.3.2 Economic Feasibility:

It is concerned with the returns or benefits of the organization are likely to derive from investment in the new system. Estimated costs of new system development and operation must be balanced against projected tangible as well as intangible benefits.

1.3.3.3 Operational Feasibility:

It is an evaluation to determine whether a system is operationally acceptable. Two important dimensions to be accessed are ability and motivation to use the system.

The system is operationally feasible as it is easy to use and can be applied to all kind of institutions. The project is also technically feasible as all the tools and technology required for the project is readily available. The requirements for the system are within the resources provided and hence make the endeavor economically feasible.

Chapter 2

LITERATURE REVIEW

Google Enterprise Search

Google's universal search, powered by the Google Search Appliance, provides the ability to search all enterprise content through a single search box including intranets, file shares, databases, content management systems, and real time business data.

The Google Search Appliance crawls intranet content and creates a master index of documents that is ready for retrieval using Google's search technology, whenever a customer or employee types in a search query. The Google Search Appliance can index millions of documents, and there are some security features which ensure that users can only access the information that they have permission to view.

Microsoft Enterprise Search

Microsoft Enterprise Search connects to unstructured and structured information anywhere in an organization. Unstructured information includes content on file shares, Internet and intranet Web sites, Microsoft Exchange public folders, IBM Lotus Notes repositories, and Microsoft Office SharePoint Server or Windows sharePoint Services sites. Structured information includes information in database and line of business systems, such as ERP and CRM.

Both these search tools are not open-source and their features cater to enterprise environments. Microsoft Enterprise search is free while Google Enterprise Search is not.

Apache Lucene[2]

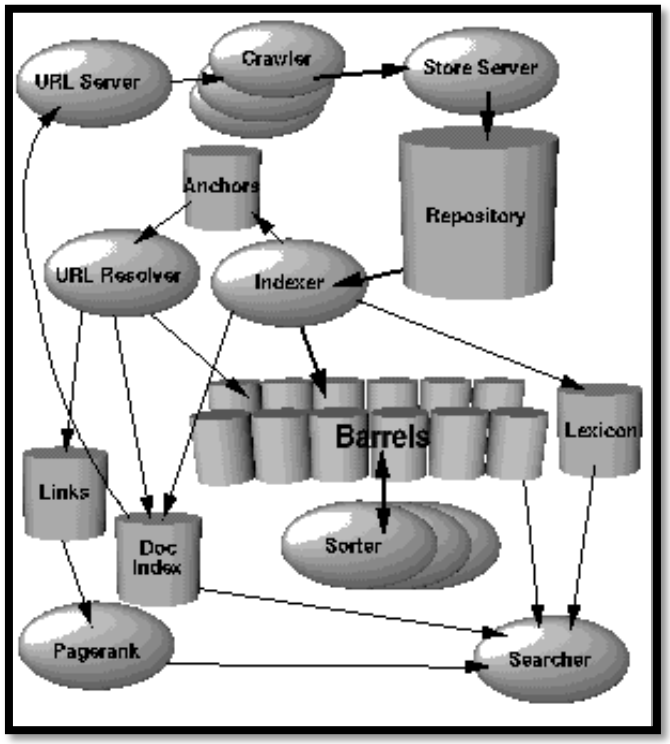
Lucene is high-performance, scalable, full-featured, open-source text search engine API. It is suitable for nearly any application that requires full-text search, especially cross-platform. Given some text lucene can create an index which can be searched later using queries.

The Anatomy of a Large-Scale Hypertextual Web Search Engine [Research Paper – Larry Page and Sergey Brin][1]

We studied this paper in order to make ourselves aware of the architecture of the most complex and large scale search engine Google.

In this paper Google is presented as a prototype of a large-scale search engine which makes heavy use of the structure present in hypertext. Google 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 different from three years ago.



This paper provides an in-depth description of a large-scale web search engine – the first such detailed public description we know of to date. 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. This paper addresses this question of how to build a practical large-scale system which can exploit the additional information present in hypertext. Also the problem of how to effectively deal with uncontrolled hypertext collections where anyone can publish anything they want is looked at.

Chapter 3

METHODOLOGY

3.1 General:

A process model for software engineering is chosen based on the nature of the project and application, the methods and tools to be used, and the controls and deliverables that are required.

Among the various available Software Process Models (Waterfall model, Spiral model, Prototype model, etc) the best suited for this project is the Incremental Model.

Incremental Model:[4]

The incremental build model is a method of software development where the model is designed, implemented and tested incrementally (a little more is added each time) until the product is finished. It involves both development and maintenance. The product is defined as finished when it satisfies all of its requirements. This model combines the elements of the waterfall model with the iterative philosophy of prototyping.

The product is decomposed into a number of components, each of which are designed and built separately (termed as builds). Each component is delivered to the client when it is complete. This allows partial utilization of product and avoids a long development time. It also creates a large initial capital outlay with the subsequent long wait avoided. This model of development also helps ease the traumatic effect of introducing complete project all at once.

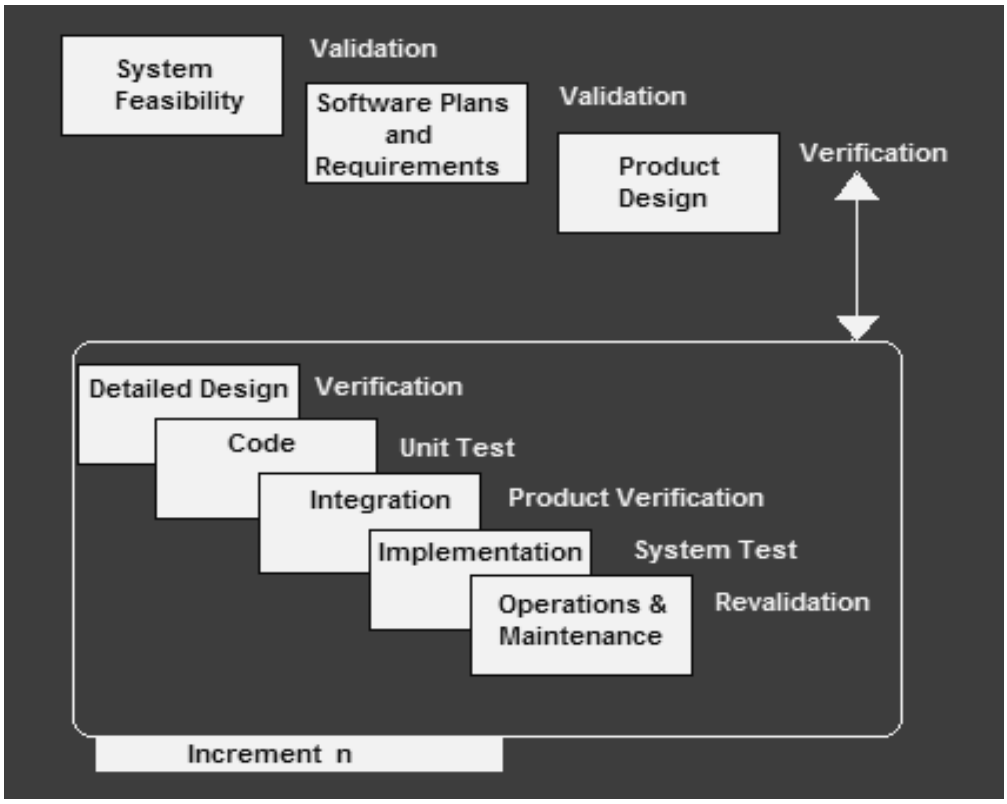


Figure 3.1: Incremental Model

Detailed Life Cycle of project:

1. **Problem Identification**

2. Preliminary investigation

3. Feasibility study

4. Determination of requirements

5. Design of system

6. Development of software

7. System testing

8. System implementation

Problem Identification:- This is the first phase of system analysis and designing. In this phase the analyst identify the organization and taking the need of system.

Preliminary investigation:-In the preliminary investigation the analysis identify the requirement through the following process that is

- **On site observation**
- **Questionnaires**
- **Documentation**
- **Conducting interview**

Feasibility study: - Once the system objectives have been ascertained by initial investigation, we need to spell the various possible solutions to meet the various objectives. The feasibility study is conducted to check whether the candidate system is feasible. The system, which is selected to be the best against the criteria, is thereafter designed and developed. The feasibility study takes into consideration, the risks involved in the project development beforehand. They are

- **Time feasibility**
- **Operational feasibility**
- **Technical feasibility**

Determination of requirements: - Requirement Analysis is the first technical step in the software engineering process. It is at this point that a general statement of software scope is refined into a concrete specification that becomes a foundation for all the software engineering activities that follow.

Analysis must focus on the informational, functional, and behavioral domains of a problem. To better understand what is required, models are created, the problem is partitioned, and representation that depict the essence of requirements and later, implementation detail, are developed.

A software requirements specification is developed as a consequence of analysis. Review is essential to ensure that developer and customer have the same perception of the

system. Unfortunately, even with best of methods, the problem is that the problem keeps changing.

Functional Requirements:

a. Data should be given in a correct form in order to avoid getting erroneous results.
b. The value of transportation cost and the assignment cost should be known prior to calculate the optimal transportation and assignment cost
c. In order to prepare the budgets the proper and the correct data is to be considered.
d. The end users should input correct figures so that the Ratio Analysis may result as per expectations.
e. Data should be checked for validity
f. Consistency between the information provided in different forms under a scheme should be checked
g. Messages should be given for improper input data, and invalid data item should be ignored.

External Interface Requirements: Functions are provided at click of the button. The output requires different type of inputs for desired result. It has been carefully analyzed that the user has to give a set of minimum commands to complete any operation.
Command names reflect the operation they perform. We have tried to make interface deal with incorrect inputs and prevent errors from entering the system where possible. It also helps the user to correct the entry. There are usually controls associated with input to ensure that no erroneous data enters the system. Similarly Output is laid out in an easy to read way.

Performance requirements: The following characteristics were taken care of in developing the system.

1. **User friendliness***:* The system is easy to learn and understand. A naive user can also use the system effectively, without any difficulty.
2. **Response time*:*** Response time of all the operations is kept low. This has been made possible by careful planning.

3. **Error handling*:*** Response to user errors and undesired situations have been taken care of to ensure that system operates without halting in case of such situation and proper error messages are given to user.
4. **Safety*:*** The system is able to avoid catastrophic behavior.
5. **Robustness*:*** The system recovers from the undesired events without human intervention.
6. **Security*:*** The system provides protection of information through the mechanism of password incorporated in it. Therefore only authorized people can access it database.
7. **Accuracy*:*** The system is highly accurate thus its utility is very high.
8. **Portability*:*** The system can be moved to new Hardware /Software after making minor changes in it.
9. **Cost Element*:*** Servicing a given demand in the system doesn't require a lot of money.

Design of system: when the analysis identifies the requirement, then designing phase is start. In this phase the flow chart and DFD are designed. Design is the most important part of the development phase for any product or system because design is the place where quality is fostered. Design is the only thing, which accurately translates a customer's requirement in to a finished software product or system. The design step produces a data design, an architectural design, and Interface Design and a Procedural Design.

System specialists often refer to this stage as logical description, in contrast to the process of developing the program software, which is referred to as physical design.

The system design describes the data to be input, calculated or stored. Individual data items and calculation procedures are written in detail. The procedures tell how to process the data and produce the output.

In the process separate tables have been created for specific details .In order to make the response time negligible all the tables have been created in the memory file system. Indexes have been created on the tables in which the number of records is very large.

Development of software: In this phase coding is started and all the designs developed are coded on the selected platform.

System testing: In this project we have performed three levels of testing: Unit testing, Integration testing and system testing.

Approach for testing: For unit testing, structural testing based on the branch coverage criterion is used. System testing is largely functional in nature. The focus was on invalid and valid cases, boundary values, and special cases. For each data entry screen , we prepared test data with extreme values & tested under all relevant test conditions. It was it was extensively tested that all forms function in the manner as expected by them and give accurate results. After my own satisfaction, we invited the concerned user to test the system.

First level of Testing: Erroneous entry is not accepted during the very time of input. To check these errors we have pressed wrong keys spontaneously and trying to trap and to check that if they lie in the acceptable range. If they are not acceptable in specified range the error message regarding particular error splashes on the screen. This strategy for whole activity is used as follows:

- To stop alphanumeric entry where only numbers are required.
- To restrict the length of an entry for a particular field so that it that input is consistent with the data structure and also lies in the acceptable range.

Second level of testing: Values are accepted as the user enters them and then checked for validity.

While making entries in a form, if the user tries to skip a field that cannot hold a null value an appropriate message is displayed conveying to the user that data must be provided.

When possible, the lists are provided to see the valid choice for the field. Once a value has been entered into a field, the cursor moves to the next field only after ensuring that the value entered in the current field is valid.

System implementation:[5] In this section, we will be discussing the plan, which will be adopted while implementation of the software in any organization. Implementation is the process of having systems personnel check out and put new equipment into use, train users, install the new application and construct any files of data needed to use it. Depending on the size of the organization that will be involved in using the application and the risk associated with its use, systems developers may choose to test the operation in only one area of the department, or with only one or two persons. Since organization systems are the business environment undergoes continual change, the information systems should keep pace. In this sense, implementation is an ongoing process. The description of the implementation plan in the department is as follows:-

In conversion from manual to computerized, the objective is to put the tested system into operation while holding cost, risks and personal irritation to a minimum. It involves creating:

1. Computer compatible files
2. Training operating staff
3. Installing terminals and hardware

A critical aspect of conversion is not disrupting the functioning of organization. Conversion should be existing, because it is the last step before the candidate system begins to show results. Unfortunately the results of conversions have been chaotic and traumatic for many firms. Unforeseen difficulties crop up as the system breaks down, data files are damaged and tempers grow short. The training package is frequently not complete and people are trying to figure out what to do. Much of these stems from poor planning or no planning at all. Therefore to avoid such problems we will be using the "parallel run approach".

The basic steps involve in the conversion are as follows:

- Conversion begins with a review of project plan, the system test documentation and the implementation plan.
- The conversion portion of implementation plan is finalized and approved.
- Files are converted.

- Parallel processing between the existing and new system is initiated.
- Results of computer runs and operations for the new system are logged on a special form.
- Assuming no problem, parallel processing is discontinued. Implementation results are documented for reference.
- Conversion is completed. Plans for the post implementation review are prepared. Following the review, the new system is officially operational.

The reason behind the use of this approach is that the software has been prepared and tested on a standalone system. So, this software will be used along with the manual system in the organization for sometimes to test its functioning under real life conditions. If there will be any problem with the software, then necessary changes will be made in the software and then it will be implemented for use. However, if there is no problem experienced by the users, then the manual system will be discontinued for further use and the department will use this software alone.

Pre Implementation Review: Pre implementation review means to check the system that whether it is working properly or not before the use of the system that is system is tested with all test cases before its implementation. If it works properly with all the test cases then only the system is implemented otherwise some amendments have to be made in the system and the system is checked again. There are many methods to perform pre implementation review. They are as under:

- Parallel system
- Pilot approach
- Phase in methods

All these approaches are followed to find out that whether the system will work out properly after it is implemented. Once these approaches satisfy the users then only the system is implemented in the department and is used by the users to perform their tasks.

In this system we have adopted the parallel run approach to find out that whether the system is giving proper result or not and we had come to the result that the system is performing all the task very efficiently and in very less time compared to the

time taken when the work is done manually. So after the satisfaction of ourselves and the user implementation of the system is done.

Post Implementation Review: After the system is implemented and conversion is completed, a review of the system is usually conducted by the users and analysts alike. Not only is this a normal practice, but it should be a formal process to determine how well the system is working, how it has been accepted, and whether adjustments are needed.

The review is all important to gather information for the maintenance of the system. Since no system is really ever complete, it will be maintained as changes are required because of internal developments, such as new users or new activities, and external developments, such as new legal requirements, new standards, or competition. The post implementation review provides the first source of information for maintenance requirements.

The most fundamental concern during post implementation review is determining whether the system has met its objective; that is, analysts want to know if the performance level of users has improved and if the system is producing the result intended.

In general, the data collection methods of questionnaire, interview, observation, sampling, and record inspection are most useful for collecting details about the new system. These review methods emphasize the importance of collecting both quantitative and subjective data to determine the suitability of the system. There is no substitute for effective review.

In this project we have done the post implementation review by inspecting the records as well as by interviewing the users of the system and at last find that this system is capable of performing all the work for which the software is made in very proper way. No information is lost by using this software and the time is also reduced to complete a task compared to the time spend when the work is done manually.

3.2 Brief plan of work:

The progress of the project throughout the academic session can be visualized through the gantt chart and pert chart given below:

3.2.1 WBS:

	Task Name	Duration	Start	Finish	Predecessors	Resource Names
1	**System Analysis**	**34 days?**	**Wed 9/1/10**	**Mon 10/18/10**		**Analyzer[25%]**
2	Study of existing system	13 days?	Wed 9/1/10	Fri 9/17/10		Analyzer
3	Explore user need	13 days?	Tue 9/7/10	Thu 9/23/10		
4	Requirement Gathering	18 days?	Thu 9/2/10	Mon 9/27/10		
5	Requirement Finalization	7 days	Wed 9/22/10	Thu 9/30/10		
6	SRS Creation	12 days	Fri 10/1/10	Mon 10/18/10	5	Computer[1]
7	**System Design**	**54 days?**	**Tue 10/12/10**	**Fri 12/24/10**		
8	Module Design	4 days	Tue 10/12/10	Fri 10/15/10		
9	Interface Design	18 days	Wed 10/27/10	Fri 11/19/10		Interface Designer[2
10	Database Design	10 days	Wed 11/24/10	Tue 12/7/10	9	Database designer[3
11	Constraint Design	5 days?	Mon 12/13/10	Fri 12/17/10	10	Computer[1]
12	Test Case Design	5 days?	Mon 12/20/10	Fri 12/24/10	11	
13	**Coding**	**47 days?**	**Wed 12/22/10**	**Thu 2/24/11**		
14	File Searching Module	26 days	Wed 12/22/10	Wed 1/26/11		Programmer[60%]
15	Crawling Module	18 days	Wed 12/22/10	Fri 1/14/11		Programmer[60%]
16	Ranking Module	15 days?	Wed 12/29/10	Tue 1/18/11		
17	String Matching Module	10 days?	Wed 1/5/11	Tue 1/18/11		
18	Web Search Module	20 days?	Wed 1/19/11	Tue 2/15/11	17	
19	Report Generation	7 days	Wed 2/16/11	Thu 2/24/11	18	Equipments[1]
20	**Testing**	**31 days**	**Fri 2/18/11**	**Fri 4/1/11**		**Tester[80%]**
21	Unit Testing	10 days	Fri 2/18/11	Thu 3/3/11		Equipments[1]
22	Integration Testing	9 days	Fri 3/4/11	Wed 3/16/11	21	
23	System Level Testing	16 days	Fri 3/11/11	Fri 4/1/11		
24	**Maintenance**	**23 days**	**Tue 4/5/11**	**Thu 5/5/11**		**Maintainer**
25	Adaptive Maintenance	7 days	Tue 4/5/11	Wed 4/13/11		Equipments[1]
26	Corrective Maintenance	8 days	Thu 4/14/11	Mon 4/25/11	25	
27	Perfective Maintenance	8 days	Tue 4/26/11	Thu 5/5/11	26	

Figure 3.2: Work Breakdown Structure

3.2.2 Gantt Chart:

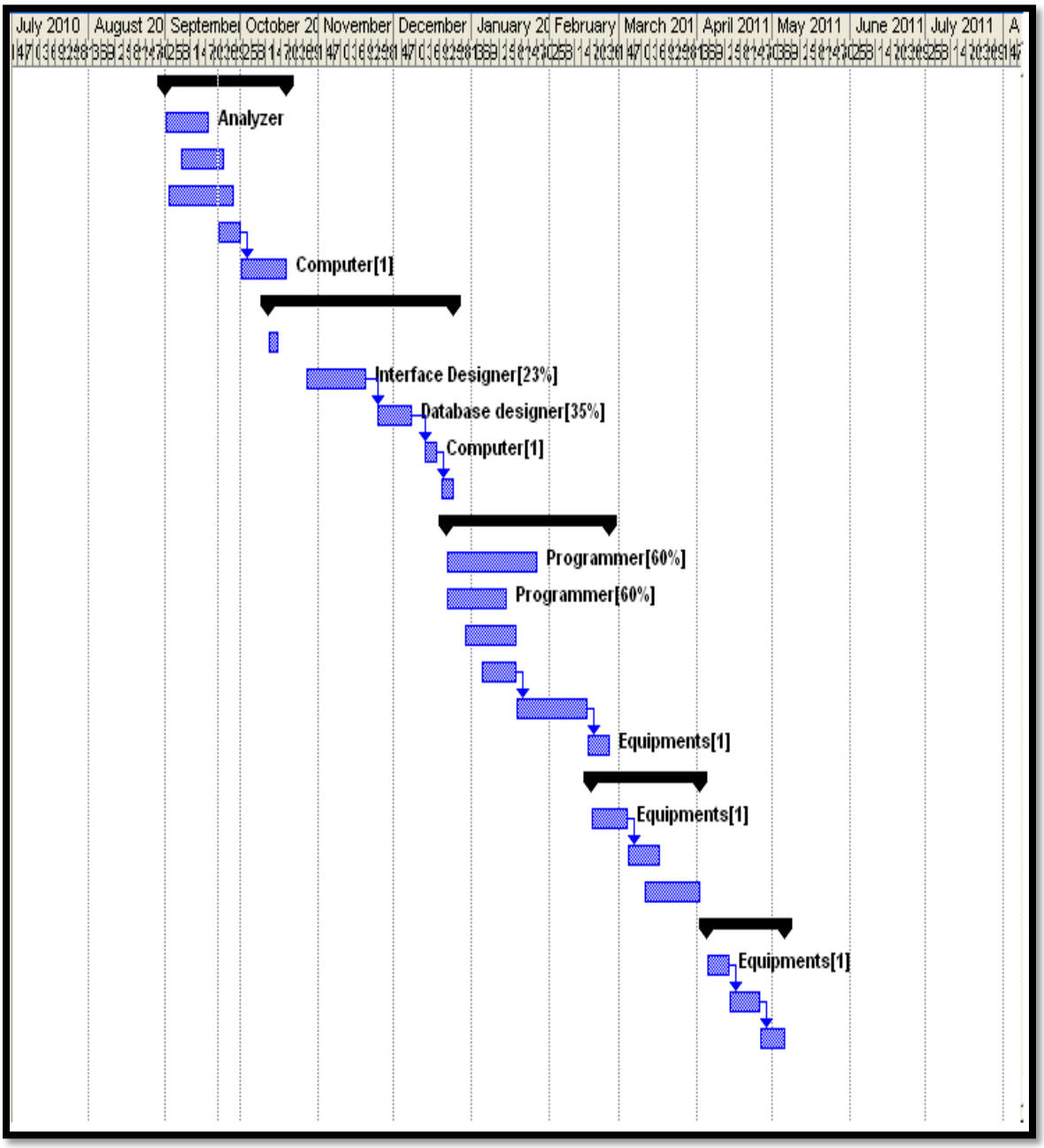


Figure 3.3: Gantt chart

3.2.3 Pert Chart:

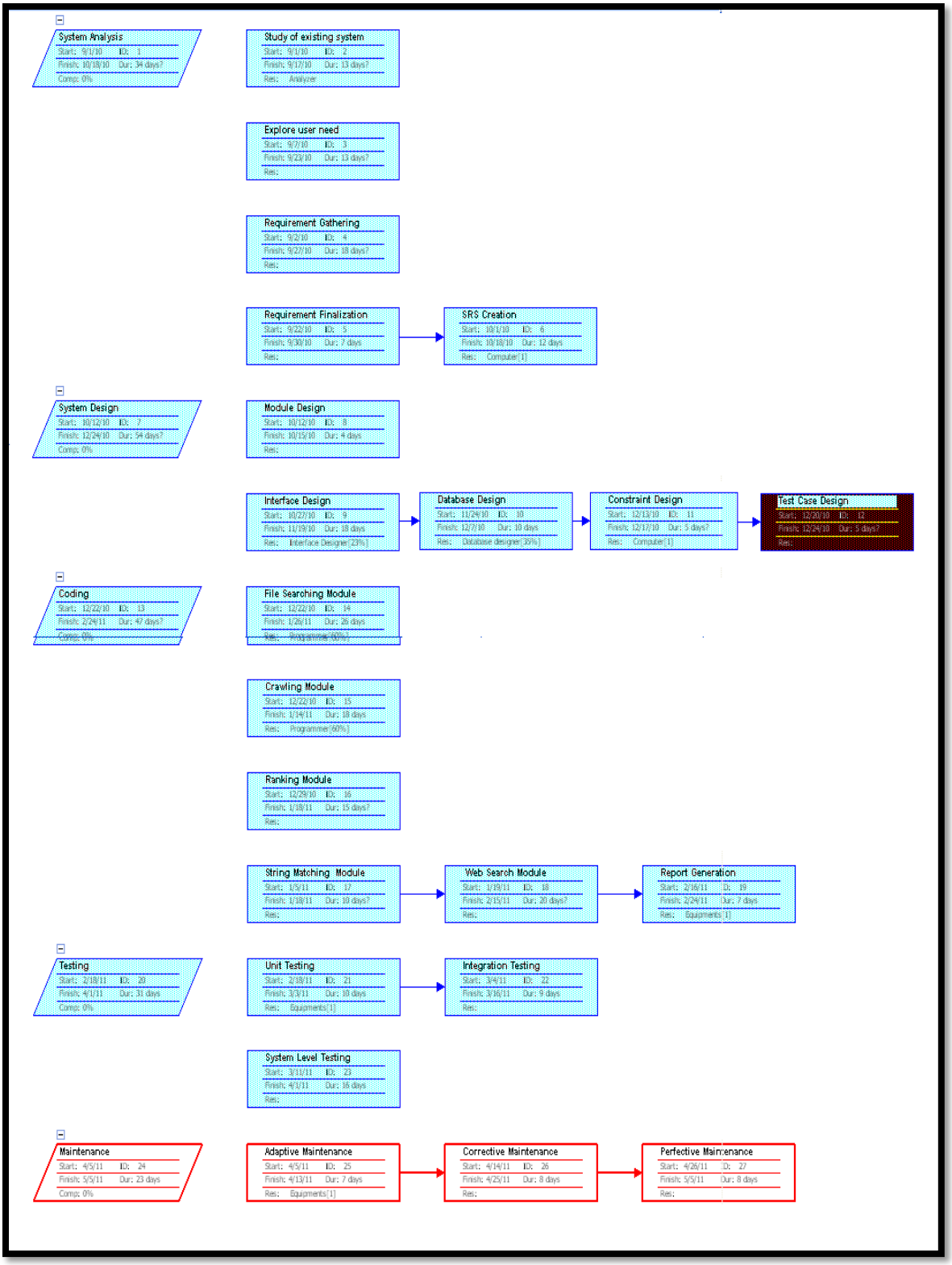


Figure 3.4: Pert chart

3.2.4 Data Flow Diagrams:

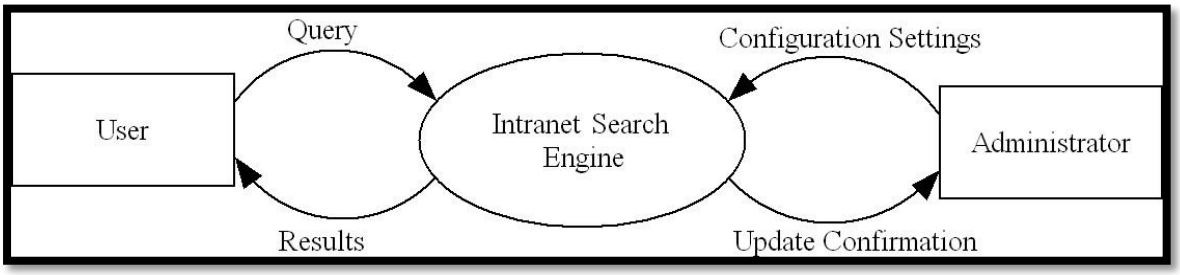


Figure 3.5: Zero Level DFD

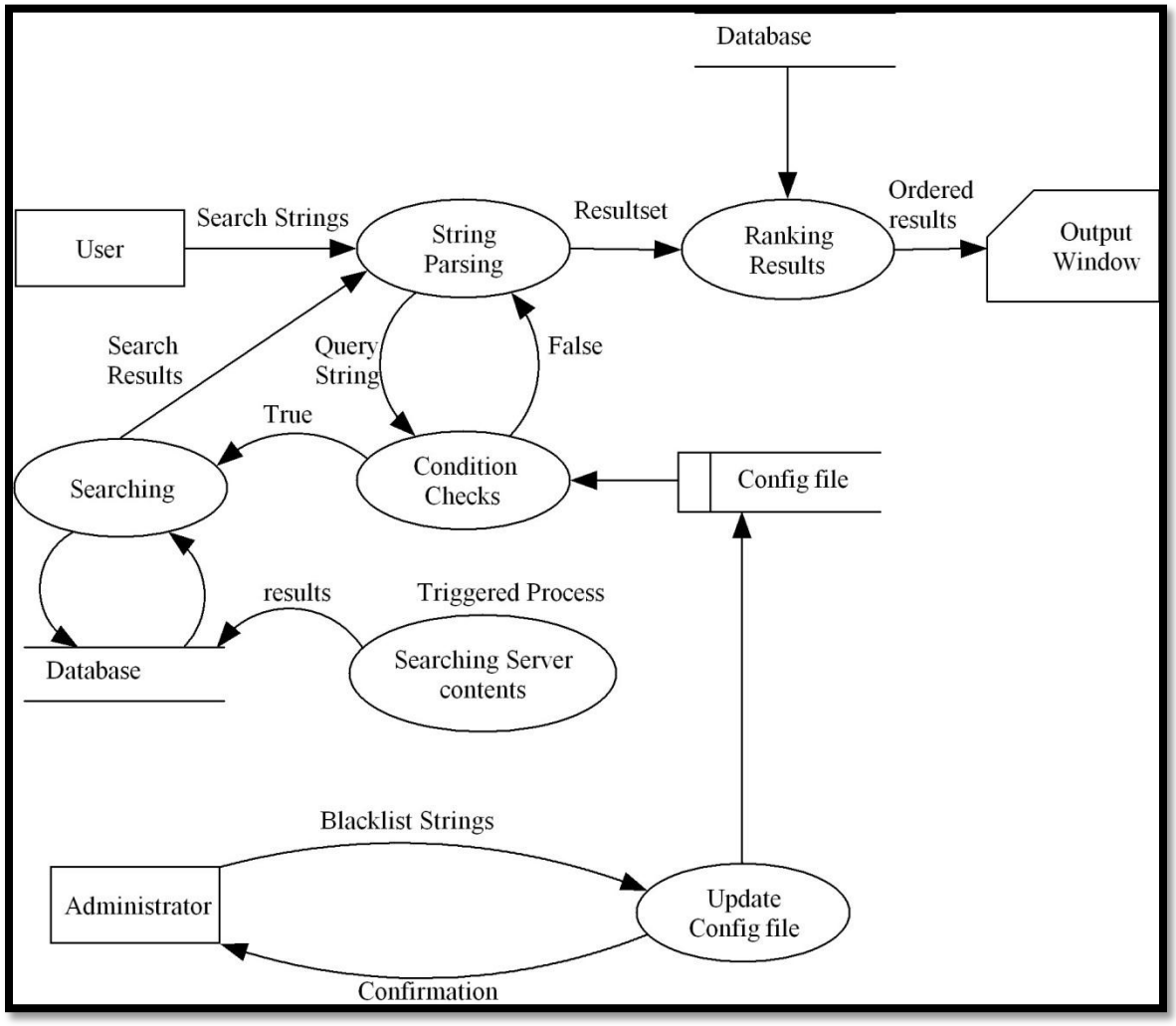


Figure 3.6: One Level DFD

Chapter 4

HARDWARE AND SOFTWARE REQUIREMENTS AND SPECIFICATIONS

4.1 Hardware (Server):

Number	Description	Type
1.	Processor	Pentium IV or higher
2.	RAM	1GB
3.	HDD	10GB
4.	Monitor	SVGA/VGA with resolution 1024 by 768 pixels
5.	LAN connection	Any compatible type

Table 4.1

4.2 Software:

Number	Description	Category
1.	RHEL 5.4	Operating System
2.	J2EE	Front end
3.	MY SQL	Back end
4.	APACHE TOMCAT	Web-server

Table 4.2

4.3 Software Engineering Paradigm Applied:

A successful software project involves a lot more than just efficient programming. Regardless of the scale or complexity of any individual application, all projects contain the same progressive core development phases during which all of the uniform and project specific tasks, issues and considerations need to be addressed in a structured and methodical way. Throughout the whole development cycle we constantly review and test the quality of the code we develop to ensure our solutions are intuitive, fault tolerant and incorporate transactional isolation.

A typical roadmap within a bespoke development project incorporates the following 6 core development phases and their associated tasks and development stages:

1. Analysis

- Project Overview
- Feasibility Study
- Initial Requirement Analysis
- Budgeting Strategy
- Resource/Responsibilities Planning
- Consultancy / Project Scoping

2. Design

- Review Project Overview
- Requirement Definition
- Data Analysis
- Software Functional Design
- Create Functional Specification Design
- Review Functional Specification
- Develop Initial Prototype Sample
- Review Project Specification
- Aesthetics Design Specification
- Specification Change Control/ Acceptance Check
- Create Final Project Specification Design
- Review Final Project Specification Design
- Project Design Completion

3. Preparation

- Specification Conformance/ Acceptance Check
- Code Modifications
- Integration/ Data Testing
- Final Code Modifications
- Code Completion Change Control
- Help/ Documentation Specification
- Help/ Documentation Production

4. Development

- Coding
- Initial Debugging
- Report Writing
- Aesthetics Design

5. Implementation

- Beta Version Deployment
- Pre Implementation Modifications
- Software Completion
- System Manager Training
- Technical Environment Deployment
- User Training
- Final Data Conversion/ Data Preparation
- Application Integration/ Data Loading
- Parallel Installation

6. Support

- Post Implementation Review
- Future Requirement Planning/ Change Control
- Technical Support
- Routine Service And Maintenance
- Future Development

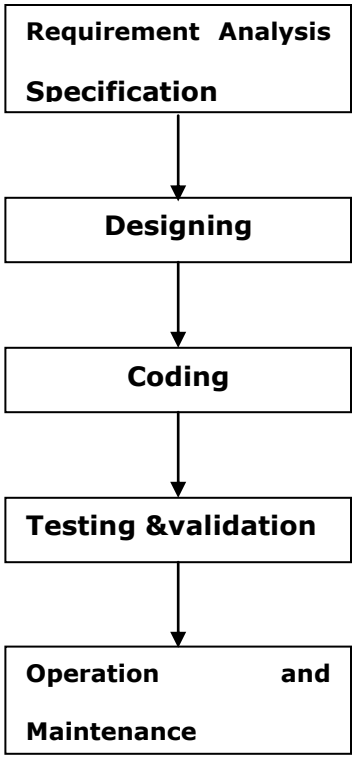


Figure 4.1: Steps used in development

To solve actual problems in an industry setting, a software engineering or a team of engineers must incorporate a development strategy that encompasses the process, method, and tools layers. This strategy is often referred to as a process model or a software engineering paradigm.

A process model or a software engineering is chosen based on the nature of the project application, the method and tools to be used, and the controls and deliverables that are required.

There are so many software paradigms some of these are:

- Liner sequential model (water fall model).
- Prototyping model.
- The RAD model.
- Incremental model

This project is built using the incremental model.

System/information engineering and modeling:

Because software is always part of larger system (or business), work begins by establishing requirements for all system element and then allocating some subset of these requirements to software. This system view is essential when software must interact with other elements such as hardware, people, and databases. System engineering and analysis encompass requirements gathering at the system level with a small amount of to level design and analysis. Information engineering encompasses requirement gathering at the strategic business level and at business area level.

Software requirements analysis:-

The requirement gathering process is intensified and focused specifically on software engineer (analyst) must understand the information domain for the software, as well as required function, behavior, performance, and interface. Requirements for both the system and the software are documented with the customer.

Design:

Software design is actually a multi step process that focuses on four distinct attributes of a program: data structure, software architecture,

Interface representations, and procedural detail. The design process translates requirements into a representation of the software that can be assessed for quality before coding begins. Like requirements, the design is documented and becomes and becomes a part of the software configuration.

Testing:

Once code has been generated, program testing begins. The testing process focuses on the logical internals of the software, ensuring that all statement have tested, and on the functional externals; that is conducting tests to uncover errors and ensure that define input will produce actual result that agree with required result.

Support:

Software will undoubtedly undergo changes after it is delivered to the customer. Changes will occur because errors have been encountered, because the software the software must be adopted to accommodate changes in its external environment, or because the customer require functional or performance enhancements. Software support/maintains reapplies each of the proceeding phases to an existing rather a new one.

Approach for software development:
In general there are three approaches to software development life cycle namely:

- **Sequential Approach**

  In this approach jumping to a later phase or coming to a previous phase is not allowed.

- **Iterative Approach**

  In an iterative approach, if there is sufficient reason to do so, one may return to previously completed step, introduce a change, and then propagate the effects of that change forward in the life cycle.

- **Recursive Approach**

  A recursive approach is that where the entire approach may be reapplied to the end products of the approach. This approach is handy in the prototype model.

We have used the iterative approach here as there were many steps that needed modification and improvements.

Chapter 5

IMPLEMENTATION

5.1 CHOICE OF PLATFORM USED:

Red Hat Enterprise Linux 5.4 as Operating System (Server)

RHEL 5.4 is used for deploying the search engine on the server, as well as for intranet mail configuration.

Java 2 Enterprise Edition as Front End:-

Java 2 enterprise edition (J2ee)[8] is a technology based on the Java language and enables the development of dynamic web sites. It was developed by Sun Microsystems to allow server side development. It allows the use of standard HTML, but adds the power and flexibility of the Java programming language. It is a presentation layer technology that allows static Web content to be mixed with Java code. It does not modify static data, so page layout and "look-and-feel" can continue to be designed with current methods. This allows for a clear separation between the page design and the application. JSP technology has facilitated the separation of the work profiles of web designer and a web developer. A web designer can design and formulate the layout for the web page by using HTML. On the other hand, a web developer working independently can use Java code and other JSP specific tags to code the business logic.

It also enables Web applications to be broken down into separate components. This allows HTML and design to be done without much knowledge of the Java code that is generating the dynamic data. JSP files are HTML files with special Tags containing Java source code that provide the dynamic content. Unlike a plain HTML page, which contains static content that always remains the same, a JSP page can change its content based on any number of variable items, including the identity of the user, the user's browser type, information provided by the user, and selections made by the user. JSP elements can be used for a variety of purposes, such as retrieving information from a

database or registering user preferences. Java's object-oriented design, platform independence, and protected-memory model allow for rapid application development.

BENEFITS FOR DEVELOPERS:

- It is easy to learn and allows developers to quickly produce web sites. JSP is based on Java, an object-oriented language. It offers a robust platform for web development.
- By having a separation of presentation and implementation, web designer's work only on the presentation and Java developers concentrate on implementing the application.
- The pages are compiled for efficient server processing.
- JSP pages can be used in combination with servlet that handle the business logic, the model supported by Java servlet engines.
- JSP is a specification, not a product. This means vendors can compete with different implementations, leading to better performance and quality.
- JSP is an integral part of J2EE. This means that JSP can play a part in the simplest applications to the most complex and demanding.
- The dynamic part is written in Java, not VBScript or another ASP-specific language, so it is more powerful and better suited to complex applications that require reusable components.
- It is portable to other operating systems and Web servers. You can take one file and move it to another platform, web server or JSP Servlet engine.

Use of struts framework:[7]

An innovative technology used in the project is the struts framework. The Struts Framework is a standard for developing well-architected Web applications. It has the following features:

- Open source
- Based on the Model-View-Controller (MVC) design paradigm, distinctly separating all three levels:
    - **Model:** application state
    - **View:** presentation of data (JSP, HTML)
    - **Controller:** routing of the application flow

- Implements the JSP Model 2 Architecture
- Stores application routing information and request mapping in a single core file, struts-config.xml

The Struts Framework, itself, only fills in the View and Controller layers. The Model layer is left to the developer.

Architecture Overview:

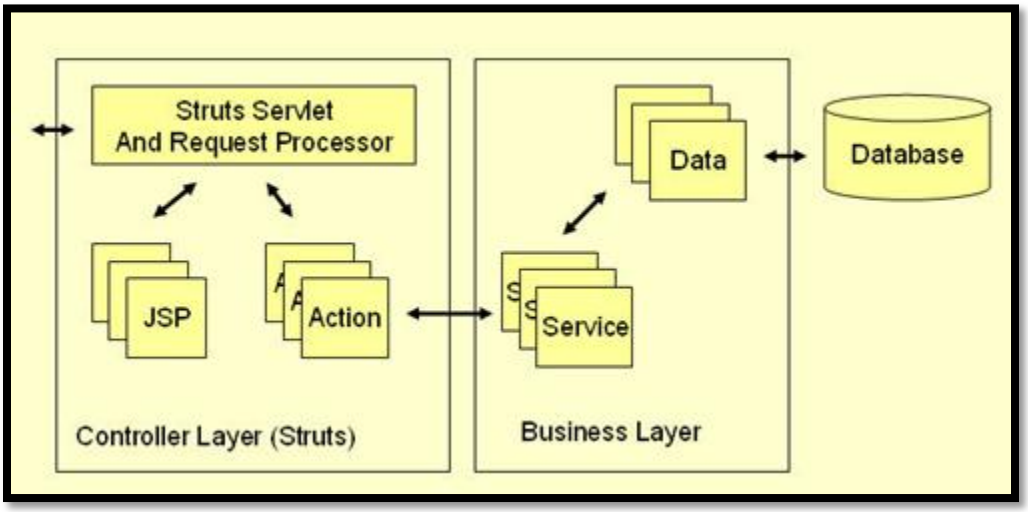


Figure 5.1: Architecture of struts framework

All incoming requests are intercepted by the Struts servlet controller. The Struts Configuration file struts-config.xml is used by the controller to determine the routing of the flow. This flow consists of an alternation between two transitions:

From View to Action	A user clicks on a link or submits a form on an HTML or JSP page. The controller receives the request, looks up the mapping for this request, and forwards it to an action. The action in turn calls a Model layer (Business layer) service or function.
From Action to View	After the call to an underlying function or service returns to the action class, the action forwards to a resource in the View layer and a page is displayed in a web browser.

Table 5.1

The diagram below describes the flow in more detail:

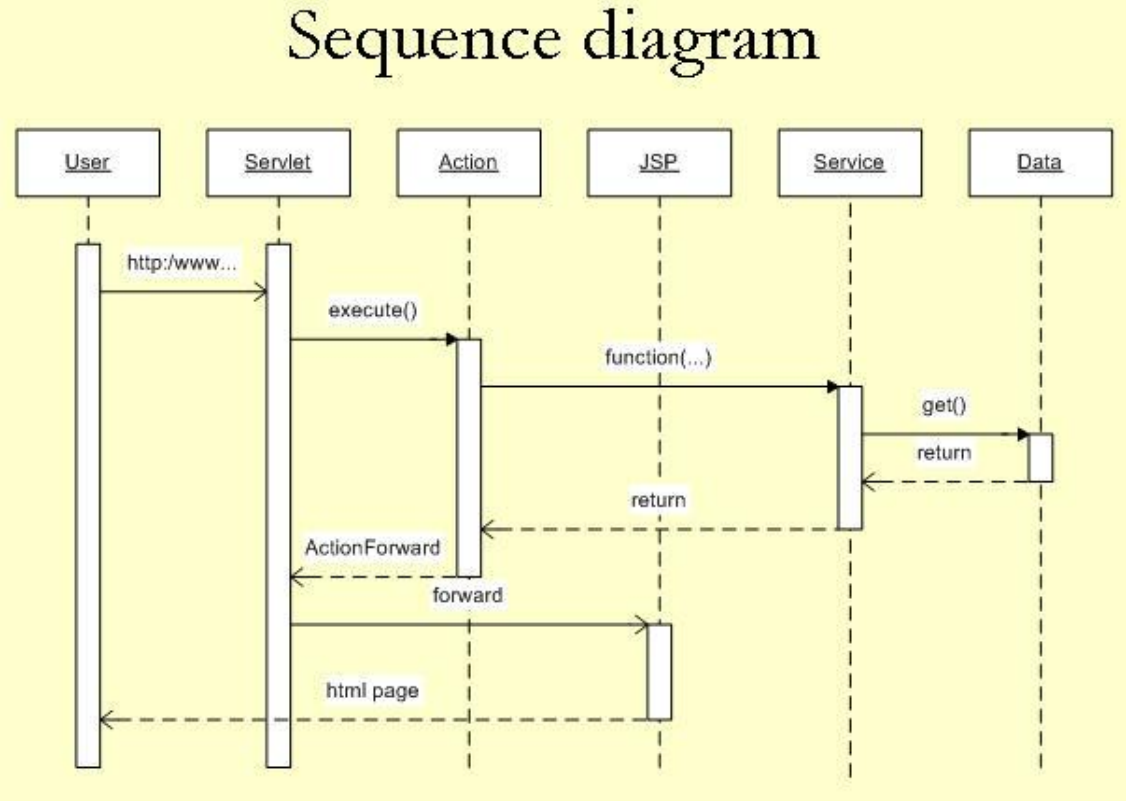


Figure 5.2: Sequence diagram of struts

1. **User** clicks on a link in an HTML page.
2. **Servlet** controller receives the request, looks up mapping information in struts-config.xml, and routes to an action.
3. **Action** makes a call to a Model layer service.
4. **Service** makes a call to the Data layer (database) and the requested data is returned.
5. **Service** returns to the action.
6. **Action** forwards to a View resource (JSP page)
7. **Servlet** looks up the mapping for the requested resource and forwards to the appropriate JSP page.
8. **JSP** file is invoked and sent to the browser as HTML.
9. **User** is presented with a new HTML page in a web browser.

Struts Components:

The Controller:

This receives all incoming requests. Its primary function is the mapping of a request URI to an action class selecting the proper application module. It's provided by the framework.

The struts-config.xml File:

This file contains all of the routing and configuration information for the Struts application. This XML file needs to be in the WEB-INF directory of the application.

Action Classes:

It's the developer's responsibility to create these classes. They act as bridges between user-invoked URIs and business services. Actions process a request and return an ActionForward object that identifies the next component to invoke. They're part of the Controller layer, not the Model layer.

View Resources:

View resources consist of Java Server Pages, HTML pages, JavaScript and Style sheet files, Resource bundles, JavaBeans, and Struts JSP tags.

Action Forms:

These greatly simplify user form validation by capturing user data from the HTTP request. They act as a "firewall" between forms (Web pages) and the application (actions). These components allow the validation of user input before proceeding to an Action. If the input is invalid, a page with an error can be displayed.

Model Components:

The Struts Framework has no built-in support for the Model layer. Struts supports any model components:

- JavaBeans
- EJB
- CORBA
- JDO
- any other

Using struts framework to develop a web project has the following benefits:

- Its open source - it's free, it's been built collaboratively and tested by many people.
- Connection pooling.
- Good tag libraries.
- Good MVC framework - you can concentrate more on business logic and less on the plumbing.
- Flexible and maintainable - XML configuration files make it easy to customize actions.
- To add functionality, you simply need to create an Action, a Form, and the JSP -- the framework takes care of the rest.
- Facilitates I18N and L10N.

MYSQL-As a Back end:

Internals and Portability:

- Written in C and C++.
- A very fast thread-based memory allocation system.
- Very fast joins using an optimized one-sweep multi-join.
- In-memory hash tables, which are used as temporary tables.

- SQL functions are implemented using a highly optimized class library and should be as fast as possible. Usually there is no memory allocation at all after query initialization.

Data Types:

- Many data types: signed/unsigned integers 1, 2, 3, 4, and 8 bytes long, FLOAT, DOUBLE, CHAR, VARCHAR, TEXT, BLOB, DATE, TIME, DATETIME, TIMESTAMP, YEAR, SET, ENUM, and OpenGIS spatial types.
- Fixed-length and variable-length records.

Security:

A privilege and password system is very flexible and secure, and allows host-based verification. Passwords are secure because all password traffic is encrypted when you connect to a server.

Scalability and Limits:

Handle large databases. We use MySQL Server with databases that contain 50 million records. Users who use MySQL Server with 60,000 tables and about 5,000,000,000 rows are not unheard of.

Connectivity:

- Clients can connect to the MySQL server using TCP/IP sockets on any platform. On Windows systems in the NT family (NT, 2000, XP, or 2003), clients can connect using named pipes. On Unix systems, clients can connect using Unix domain socket files.
- In MySQL 4.1 and higher, Windows servers also support shared-memory connections if started with the --shared-memory option. Clients can connect through shared memory by using the
  --protocol=memory option.

- The Connector/ODBC (MyODBC) interface provides MySQL support for client programs that use ODBC (Open Database Connectivity) connections. For example, you can use MS Access to connect to your MySQL server. Clients can be run on Windows or UNIX. MyODBC source is available. All ODBC 2.5 functions are supported, as are many others.

5.2 SOFTWARE DESIGN APPROACH:

The software design of the system can be best explained using the use case and the architecture diagram.

5.2.1 Use case Diagram: This use case diagram depicts the various functions users of the system can perform.

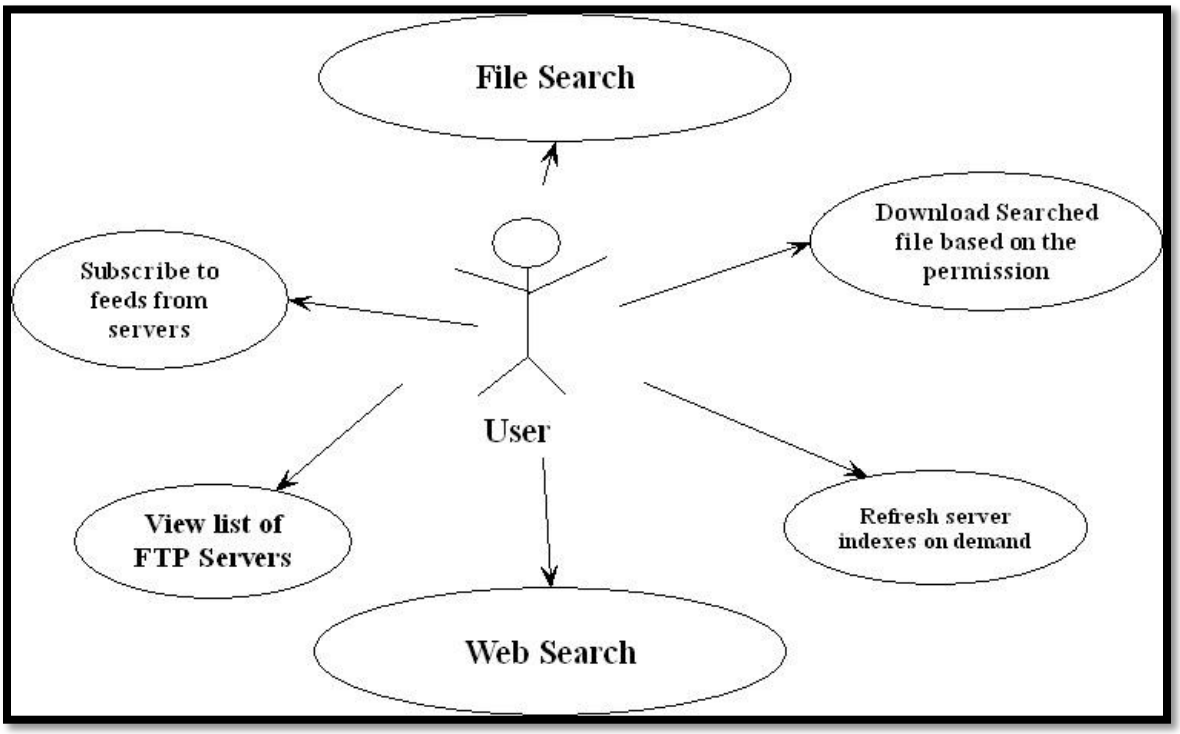


Figure 5.3: User use case diagram

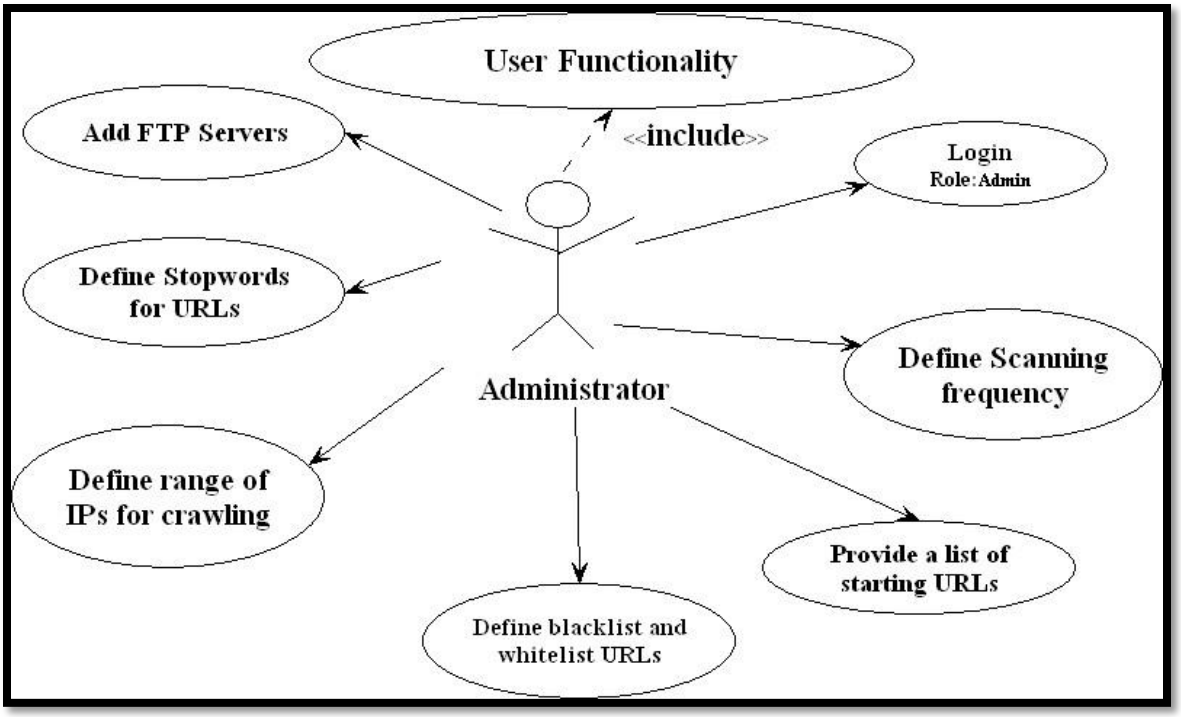


Figure 5.4: Administrator use case diagram

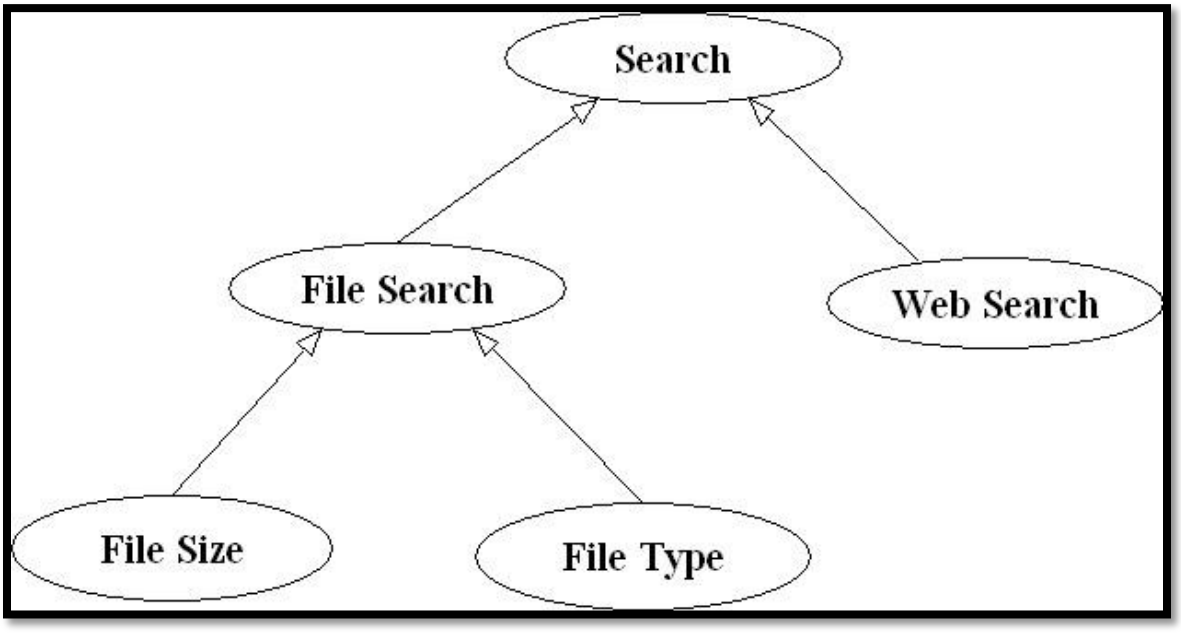


Figure 5.5: Search Hierarchy

5.2.2 <u>Sequence Architecture:</u> The following diagrams show the sequence of flow control. There is a separate diagram for each user for greater details and clarity.

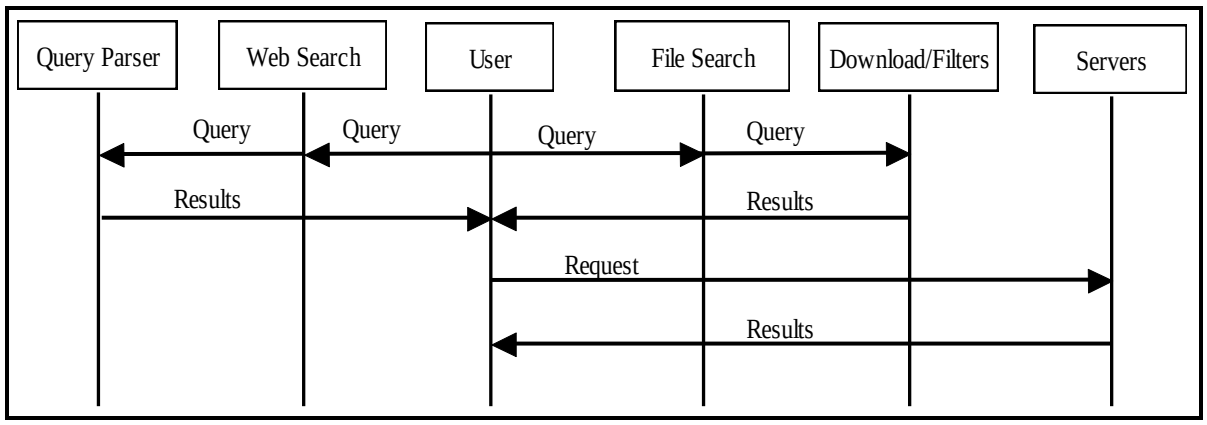


Figure 5.6: Sequence diagram for user

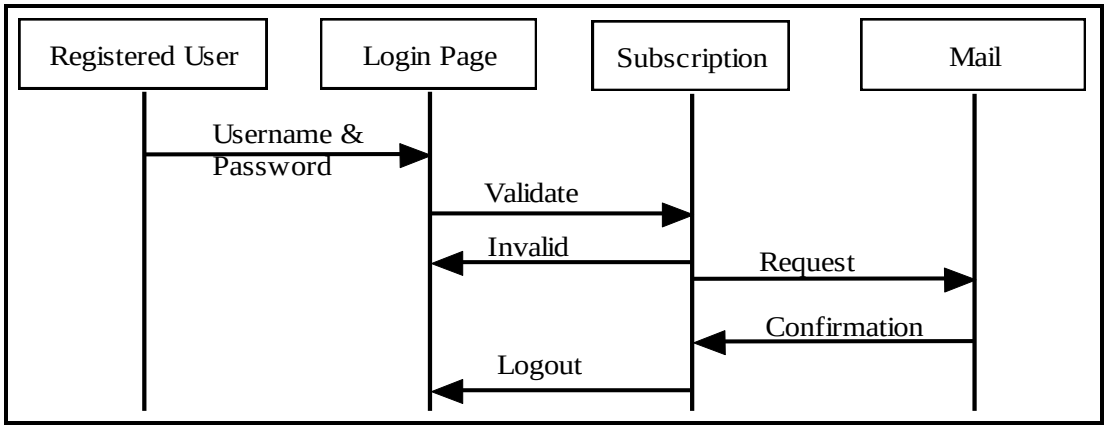


Figure 5.7: Sequence diagram for registered user

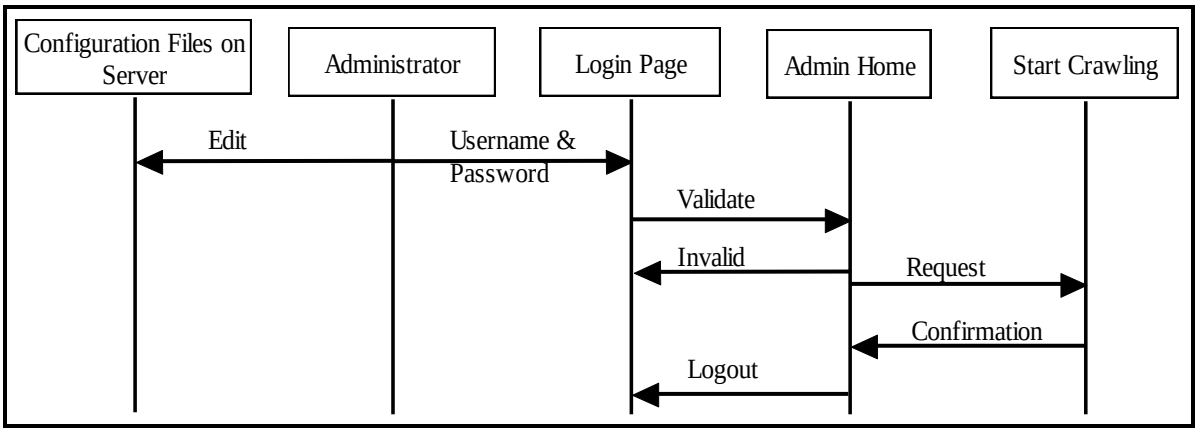


Figure 5.8: Sequence diagram for administrator

5.2.3 Database Design:[6]

A properly designed database is a model of a business, or some "thing" in the real world. Like their physical model counterparts, data models enable you to get answers about the facts that make up the objects being modeled. It's the questions that need answers that determine which facts need to be stored in the data model.

In the relational model, data is organized in tables that have the following characteristics: every record has the same number of facts; every field contains the same type of facts in each record; there is only one entry for each fact; no two records are exactly the same; the order of the records and fields is not important.

The database schema can be visualized using the following diagrams. The entity relationship diagram below shows how the various entities in the data base are related to each other and the attributes they have.

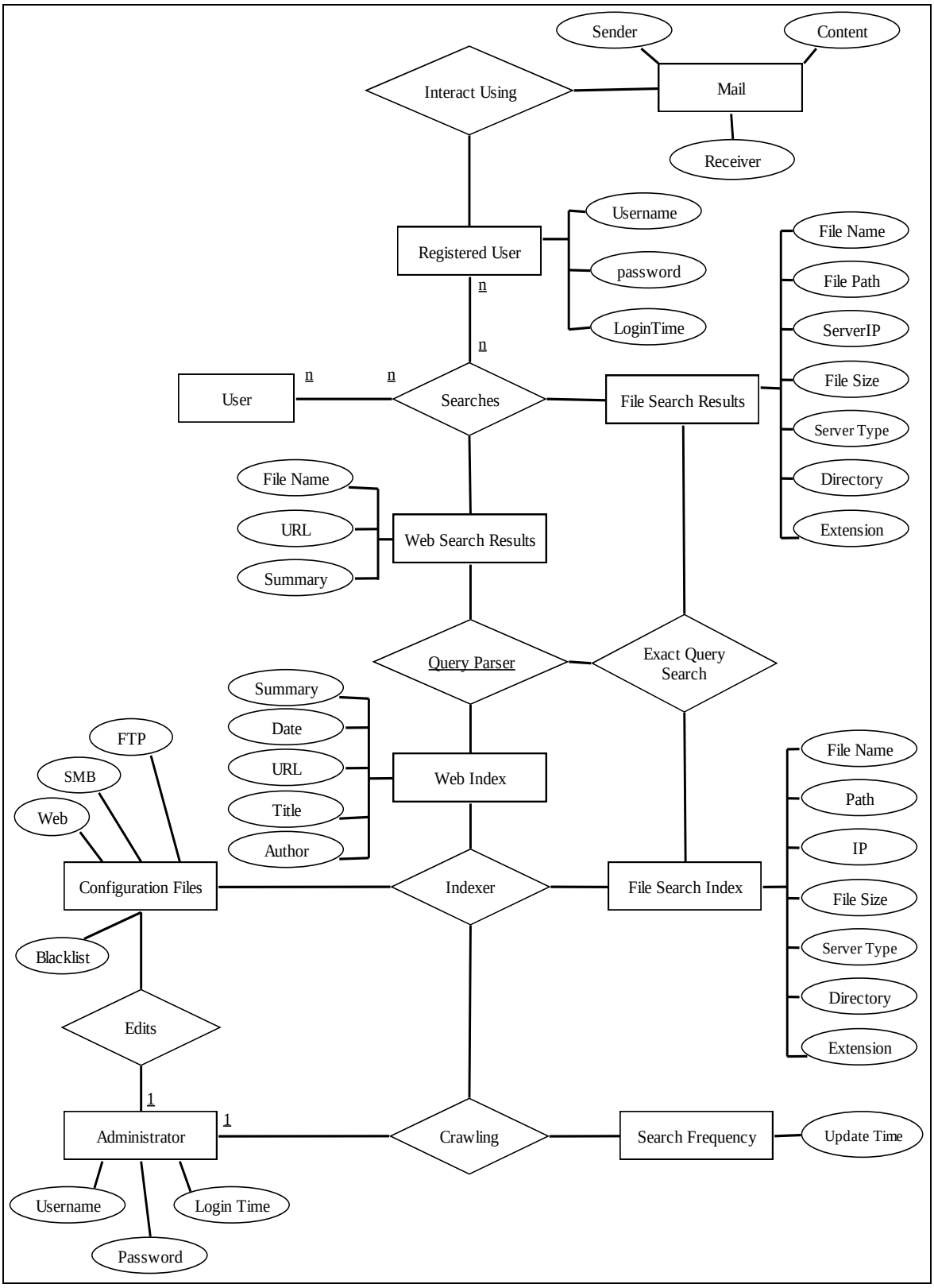


Figure 5.9: Entity relationship diagram

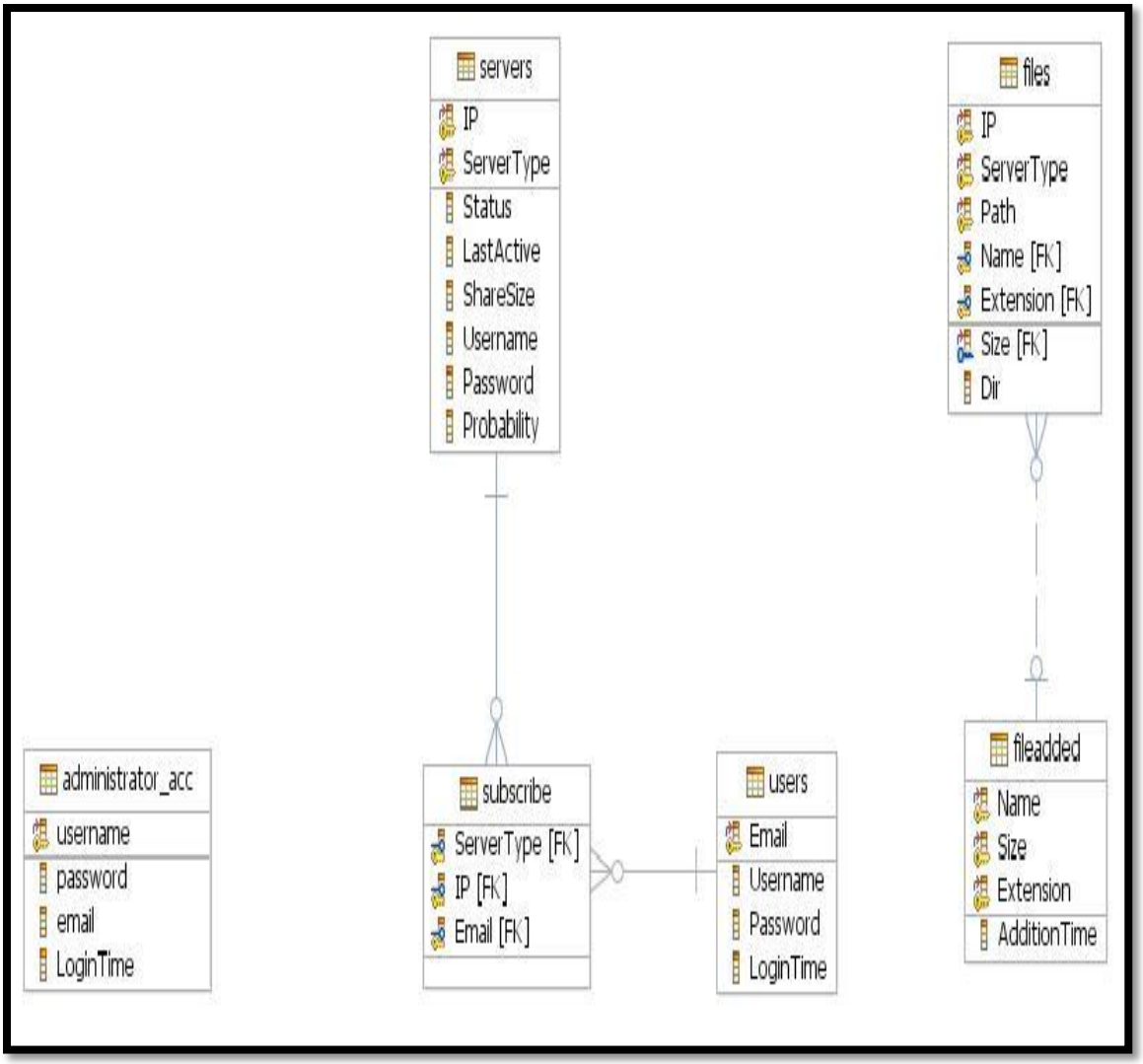


Figure 5.10: Database schematic diagram

The database consists of the following tables:

administrator_acc

Data name	Data type
username	varchar
Password	varchar
Email	varchar
LoginTime	datetime

Table 5.2

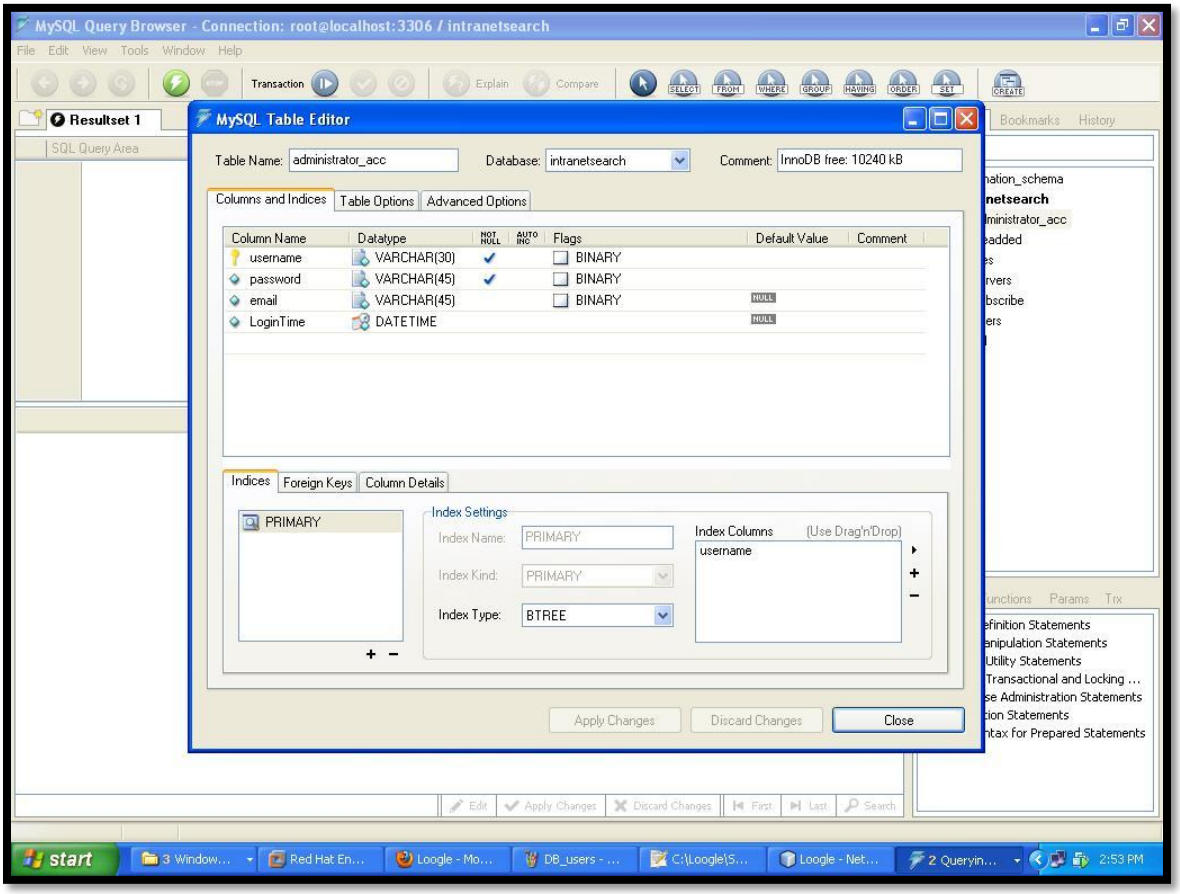


Figure 5.11: Database Table Snapshot- administrator_acc

fileadded

Data name	Data type
Name	varchar
Size	integer
AdditionTime	datetime
extension	varchar

Table 5.3

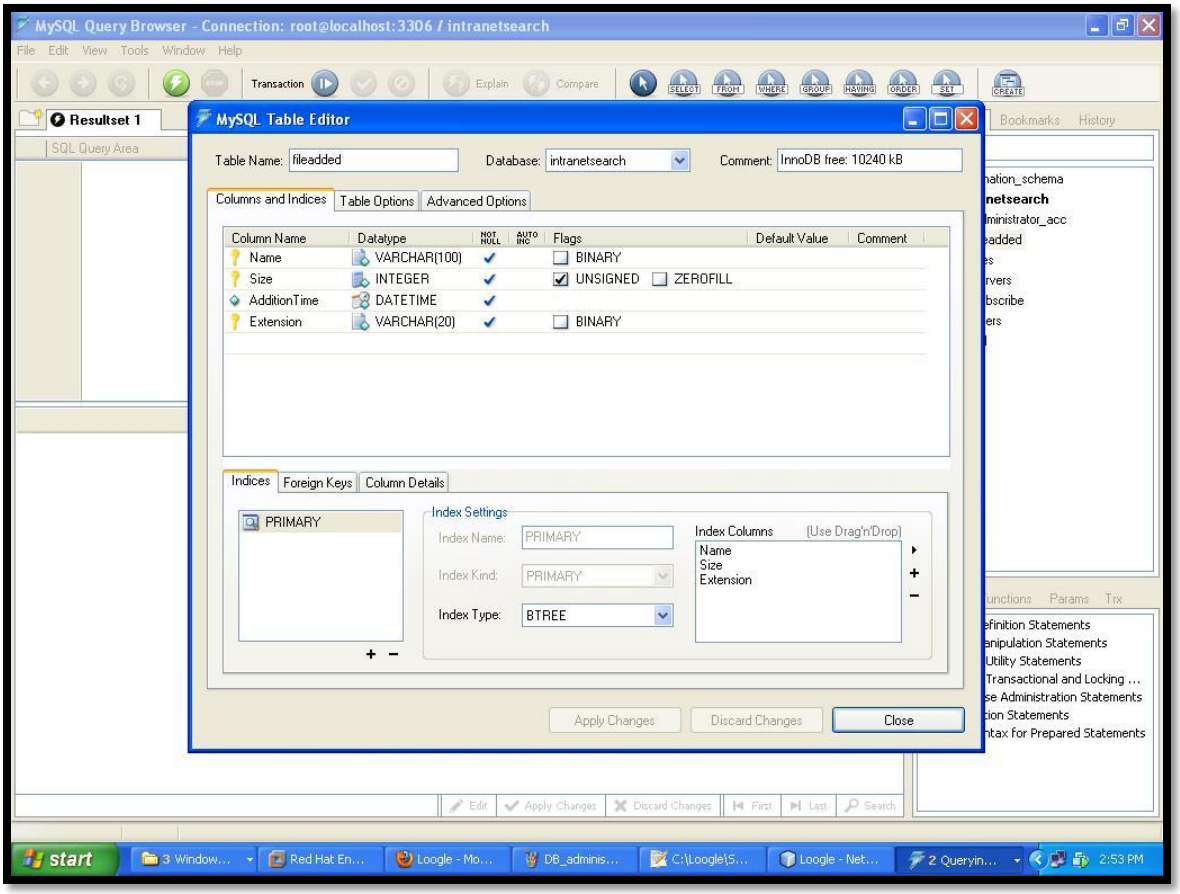


Figure 5.12: Database Table Snapshot- fileadded

files

Data name	Data type
IP	varchar
Path	varchar
Name	varchar
Extension	varchar
Size	integer
Dir	varchar
ServerType	varchar

Table 5.4

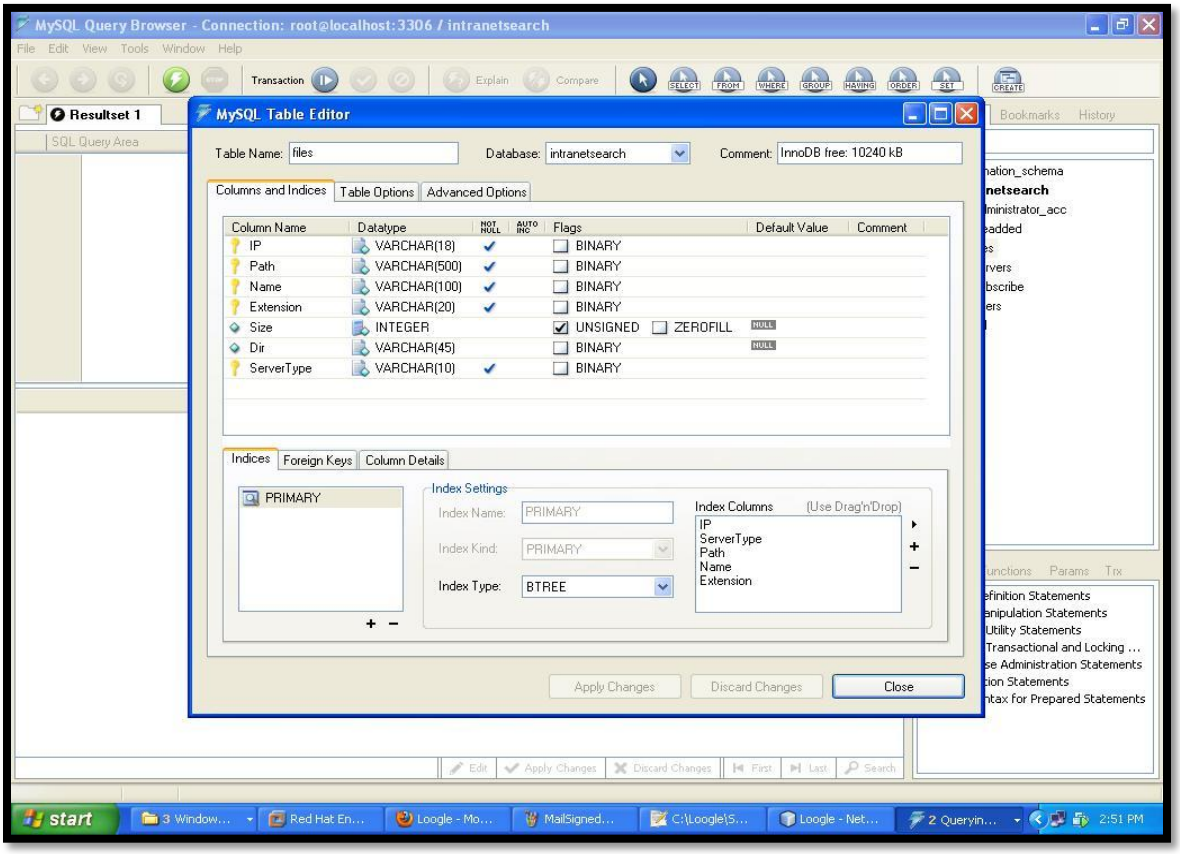


Figure 5.13: Database Table Snapshot- files

Servers

Data name	Data type
ServerType	integer
IP	varchar
Status	tinyint
LastActive	datetime
ShareSize	integer
Username	varchar
Password	varchar
Probability	integer

Table 5.5

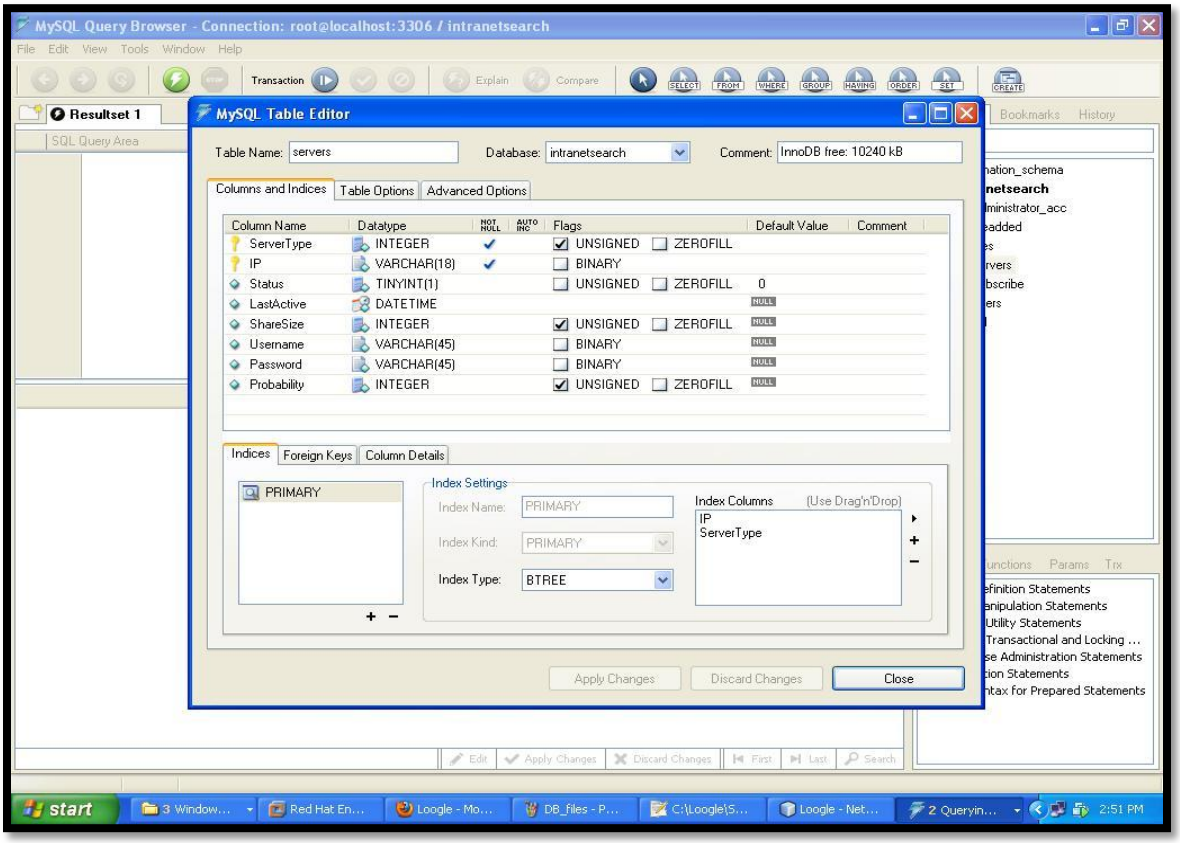


Figure 5.14: Database Table Snapshot- servers

subscribe

Data name	Data type
ServerType	Integer
IP	varchar
email	varchar

Table 5.6

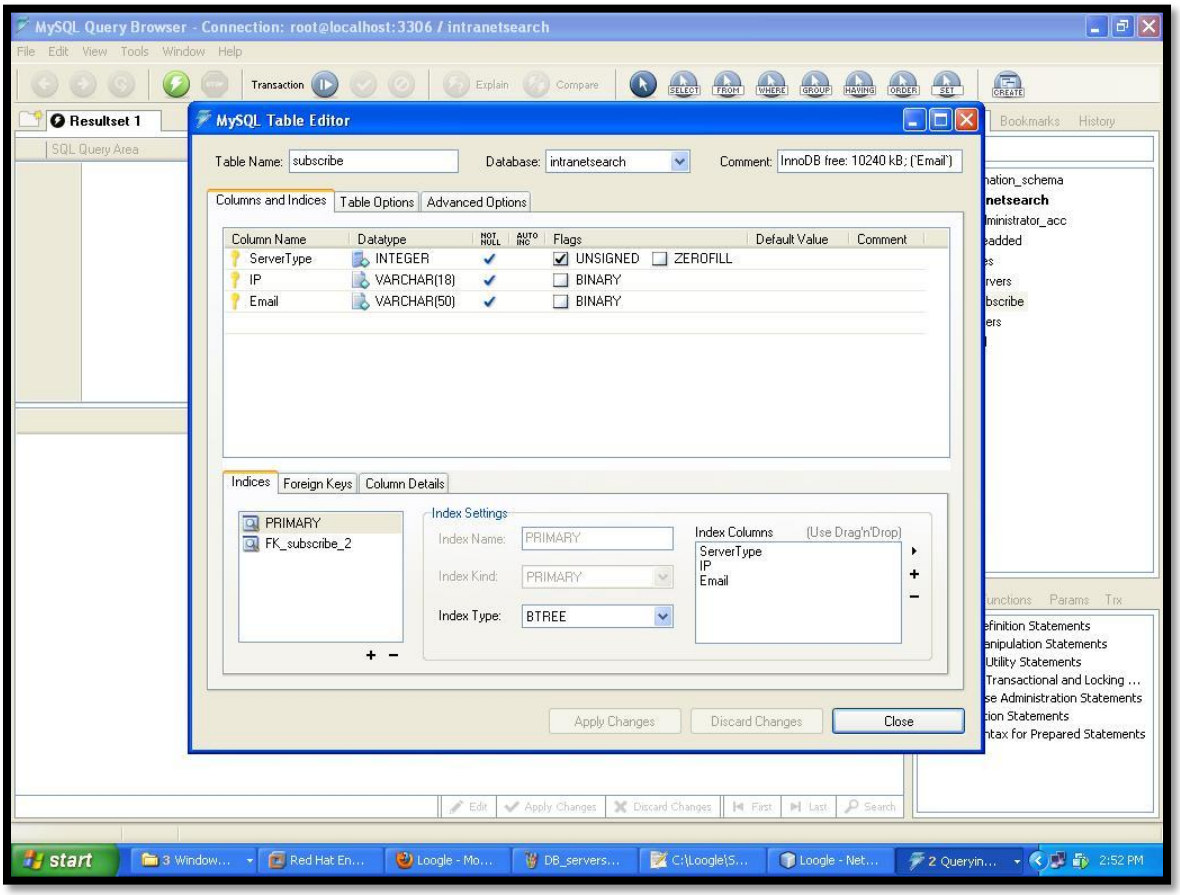


Figure 5.15: Database Table Snapshot- subscribe

users

Data name	Data type
Username	varchar
Password	varchar
Email	varchar
LoginTime	datetime

Table 5.7

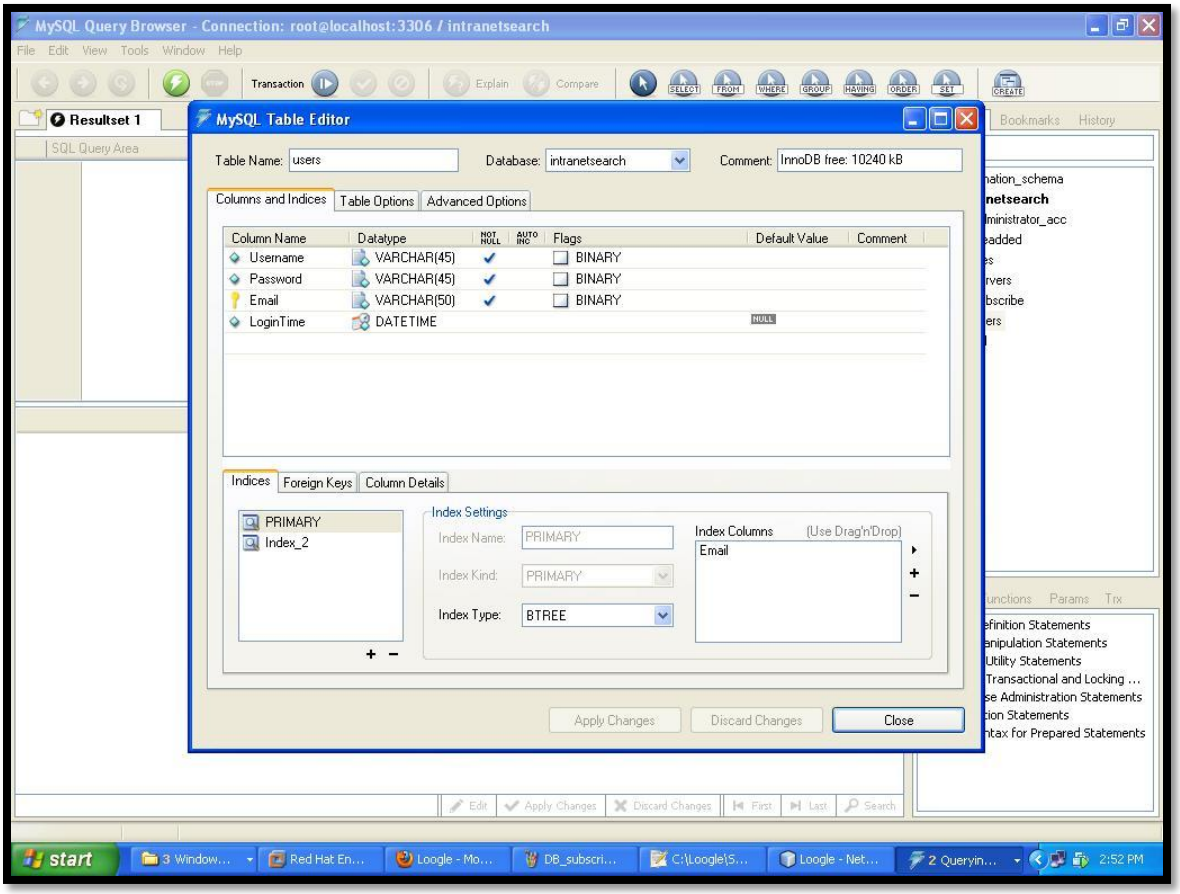


Figure 5.16: Database Table Snapshot- users

5.2.4 User Interface Design:

There are several phases and processes in the user interface design, some of which are more demanded upon than others, depending on the project. (Note: for the remainder of this section, the word *system* is used to denote any project whether it is a web site, application, or device.)

- Functionality requirements gathering – assembling a list of the functionality required by the system to accomplish the goals of the project and the potential needs of the users.
- User analysis – analysis of the potential users of the system either through discussion with people who work with the users and/or the potential users themselves. Typical questions involve:
    - What would the user want the system to do?
    - How would the system fit in with the user's normal workflow or daily activities?
    - How technically savvy is the user and what similar systems does the user already use?
    - What interface look & feel styles appeal to the user?
- Information architecture – development of the process and/or information flow of the system (i.e. for phone tree systems, this would be an option tree flowchart and for web sites this would be a site flow that shows the hierarchy of the pages).
- Prototyping – development of wireframes, either in the form of paper prototypes or simple interactive screens. These prototypes are stripped of all look & feel elements and most content in order to concentrate on the interface.
- Usability testing – testing of the prototypes on an actual user—often using a technique called think aloud protocol where you ask the user to talk about their thoughts during the experience.
- Graphic Interface design – actual look & feel design of the final graphical user interface (GUI). It may be based on the findings developed during the usability testing if usability is unpredictable, or based on communication objectives and styles that would appeal to the user. In rare cases, the graphics may drive the

prototyping, depending on the importance of visual form versus function. If the interface requires multiple skins, there may be multiple interface designs for one control panel, functional feature or widget. This phase is often a collaborative effort between a graphic designer and a user interface designer, or handled by one who is proficient in both disciplines.

User interface design requires a good understanding of user needs. The various phases of user interface design are:

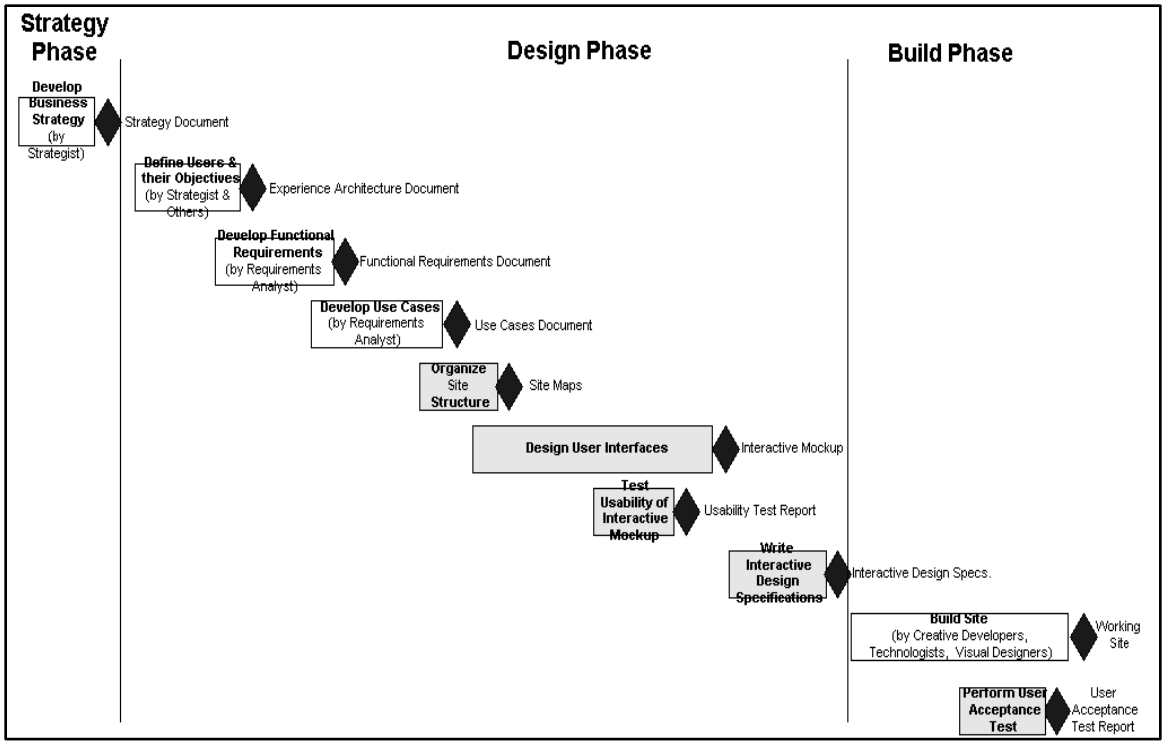


Figure 5.17: User interface development process

Requirements of a good user interface:

The dynamic characteristics of a system are described in terms of dialogue requirements contained in seven principles of part 10 of the ergonomics standard, the ISO 9241. This standard establishes a framework of ergonomic "principles" for the dialogue techniques with high-level definitions and illustrative applications and examples of the principles.

The principles of the dialogue represent the dynamic aspects of the interface and can be mostly regarded as the "feel" of the interface. The seven dialogue principles are:

- Suitability for the task: the dialogue is suitable for a task when it supports the user in the effective and efficient completion of the task.
- Self-descriptiveness: the dialogue is self-descriptive when each dialogue step is immediately comprehensible through feedback from the system or is explained to the user on request.
- Controllability: the dialogue is controllable when the user is able to initiate and control the direction and pace of the interaction until the point at which the goal has been met.
- Conformity with user expectations: the dialogue conforms with user expectations when it is consistent and corresponds to the user characteristics, such as task knowledge, education, experience, and to commonly accepted conventions.
- Error tolerance: the dialogue is error tolerant if despite evident errors in input, the intended result may be achieved with either no or minimal action by the user.
- Suitability for individualization: the dialogue is capable of individualization when the interface software can be modified to suit the task needs, individual preferences, and skills of the user.
- Suitability for learning: the dialogue is suitable for learning when it supports and guides the user in learning to use the system.

The concept of usability is defined in Part 11 of the ISO 9241 standard by effectiveness, efficiency, and satisfaction of the user. Part 11 gives the following definition of usability:

- Usability is measured by the extent to which the intended goals of use of the overall system are achieved (effectiveness).
- The resources that have to be expended to achieve the intended goals (efficiency).
- The extent to which the user finds the overall system acceptable (satisfaction).

Effectiveness, efficiency, and satisfaction can be seen as quality factors of usability. To evaluate these factors, they need to be decomposed into sub-factors, and finally, into usability measures.

The information presentation is described in Part 12 of the ISO 9241 standard for the organization of information (arrangement, alignment, grouping, labels, location), for the display of graphical objects, and for the coding of information (abbreviation, color, size, shape, visual cues) by seven attributes. The "attributes of presented information" represent the static aspects of the interface and can be generally regarded as the "look" of the interface. The attributes are detailed in the recommendations given in the standard. Each of the recommendations supports one or more of the seven attributes. The seven presentation attributes are:

- Clarity: the information content is conveyed quickly and accurately.
- Discriminability: the displayed information can be distinguished accurately.
- Conciseness: users are not overloaded with extraneous information.
- Consistency: a unique design, conformity with user's expectation.
- Detectability: the user's attention is directed towards information required.
- Legibility: information is easy to read.
- Comprehensibility: the meaning is clearly understandable, unambiguous, interpretable, and recognizable.

The user guidance in Part 13 of the ISO 9241 standard describes that the user guidance information should be readily distinguishable from other displayed information and should be specific for the current context of use. User guidance can be given by the following five means:

- Prompts indicating explicitly (specific prompts) or implicitly (generic prompts) that the system is available for input.
- Feedback informing about the user's input timely, perceptible, and non-intrusive.
- Status information indicating the continuing state of the application, the system's hardware and software components, and the user's activities.

- Error management including error prevention, error correction, user support for error management, and error messages.
- On-line help for system-initiated and user initiated requests with specific information for the current context of use.

Here we have tried to identify all these features and include them in the user interface design to make the user interface attractive, usable and error tolerant.

Screenshots:

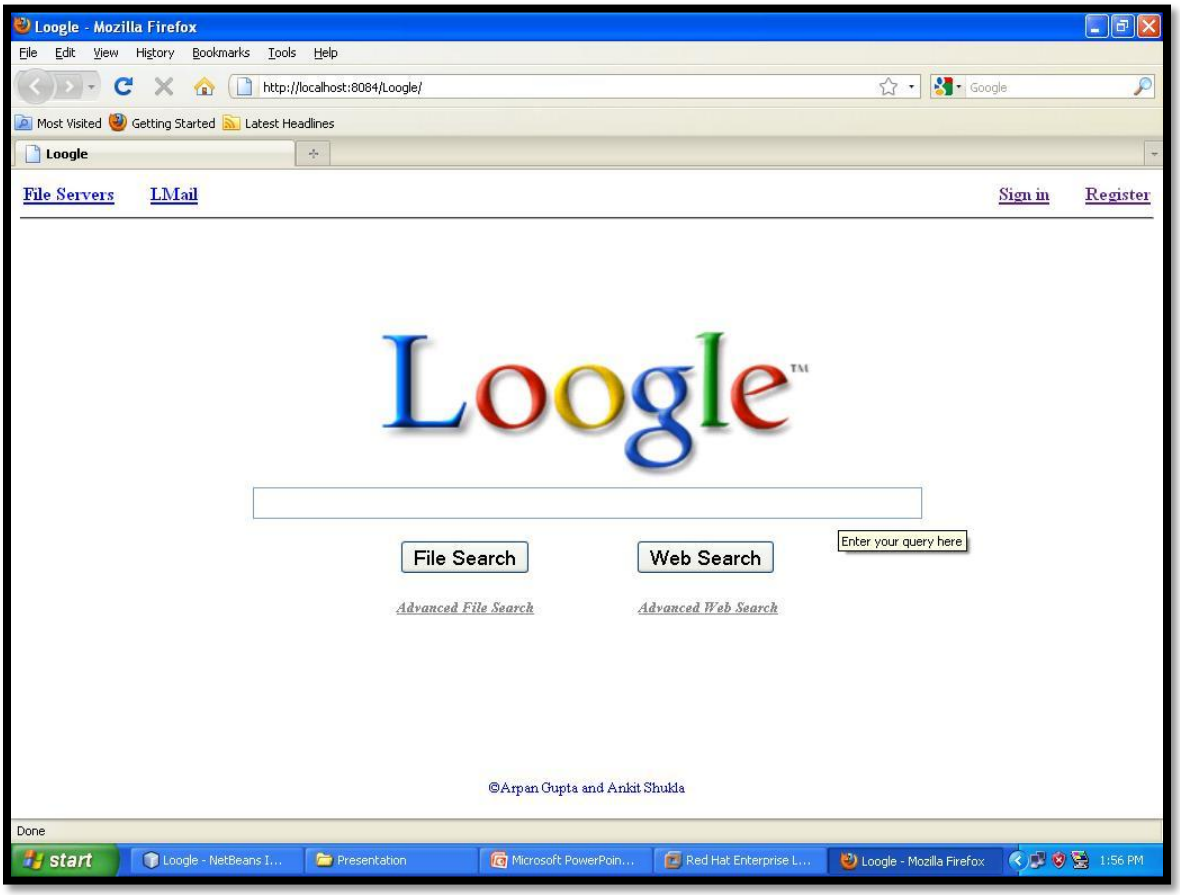


Figure 5.18: User interface screenshot- Home page

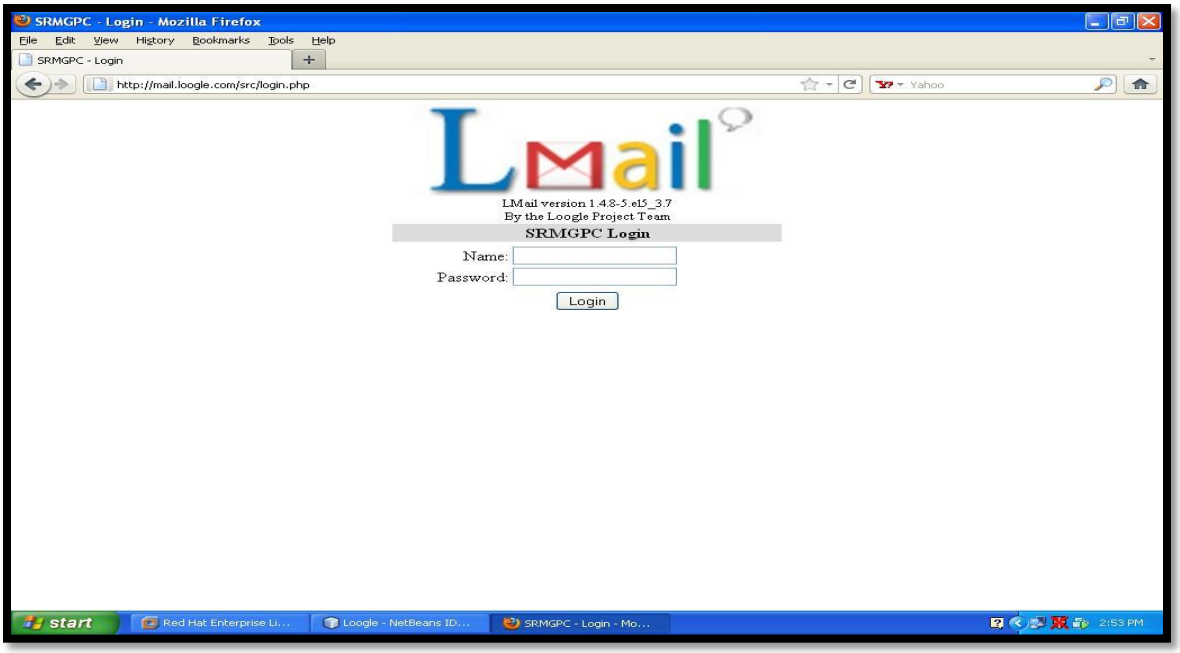


Figure 5.19: User interface screenshot- LMail Sign in

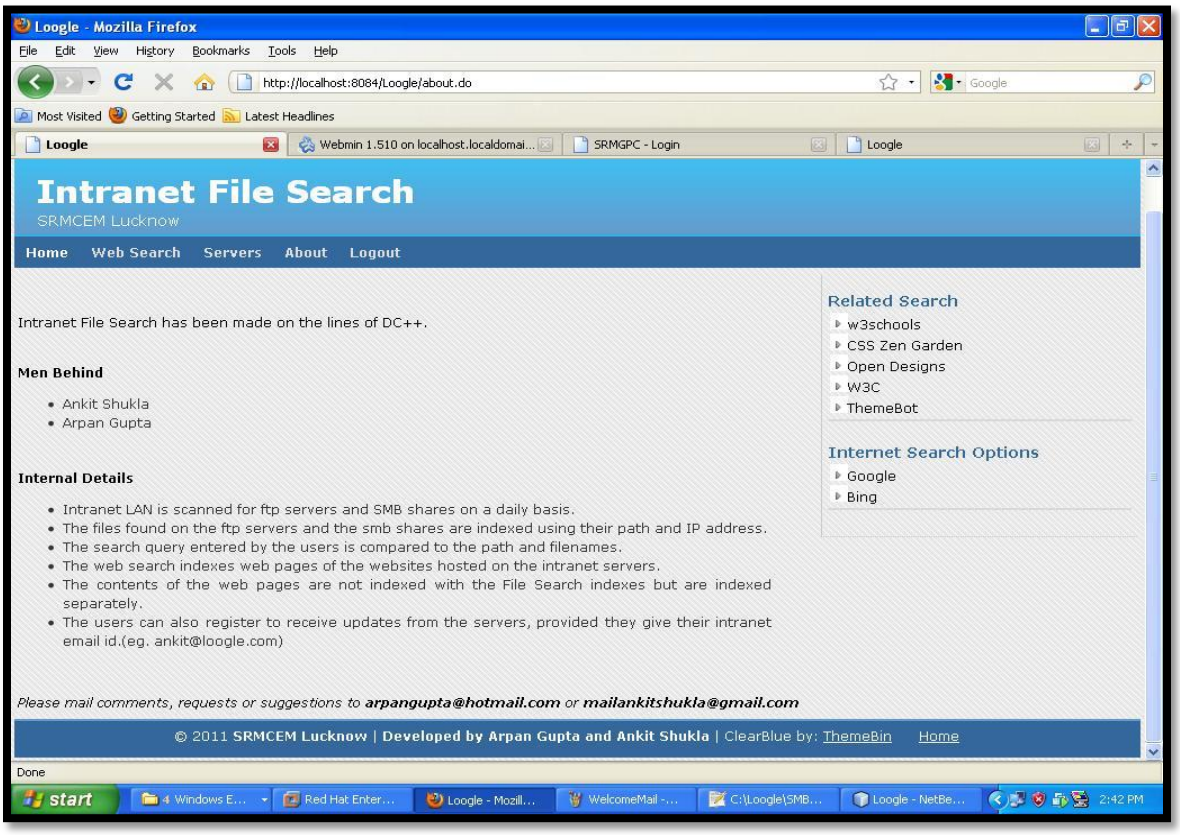


Figure 5.20: User interface screenshot- About us

5.3 MODULE DESCRIPTION:

5.3.1 File Search: The File Search module of the project deals with the following functionalities:

- The administrator supplies the IP address range of the intranet in a config file.
- Scans the range of IP addresses for FTP[14] and SMB[15] servers periodically, based on a configurable frequency (scanner frequency).
- Indexes file names of files shared on FTP and SMB servers on the intranet.
- The index is stored in a MySQL database.
- A web UI built using Java Servlets and Java Server Pages is provided to users to query for files.
- Filters are provided to users to filter search results.
- Users can view the list of FTP servers and SMB servers.
- Users can refresh server indexes on demand.
- Users can subscribe to feeds from servers.
- An admin web interface is provided to set deployment parameters.

Search Options

- ***Constraints on file size***

  This includes search on a range of file Size. For example, specifying a query with range 10-20 MB would narrow down the search domain to files with size greater than 10MB and less than 20 MB.

- ***Search on file type***

  This includes specifying options which describe the file type intended by the user. For example, the user could specify if the file is an Audio, Video, PDF, Document or a text file. User can also specify the file extension to be more specific.

- ***Search among currently active servers***

  In an intranet sharing environment, users can very easily be offline (that is, the server sharing the file may be disconnected). As such, for a user needing the file urgently at the same time may not find search results with offline servers helpful.

Hence, we have provided an option which searches for files only among currently active servers.

- ***Search in a specific server***

  If a user intends to search only in a specific server (uniquely identified by its IP), then he can specify the server IP in the search to narrow down the search to include results only from the server with the given IP.

- ***Sort the Search Results***

  We provide the option of sorting the search results by name of the file, size of the file, type of the file and server IP. The user can also specify the maximum number of search results he/she wants. In such a case, only the top N results after sorting are shown to the user.

Workflow: The diagram depicts typical workflow of the file search module:

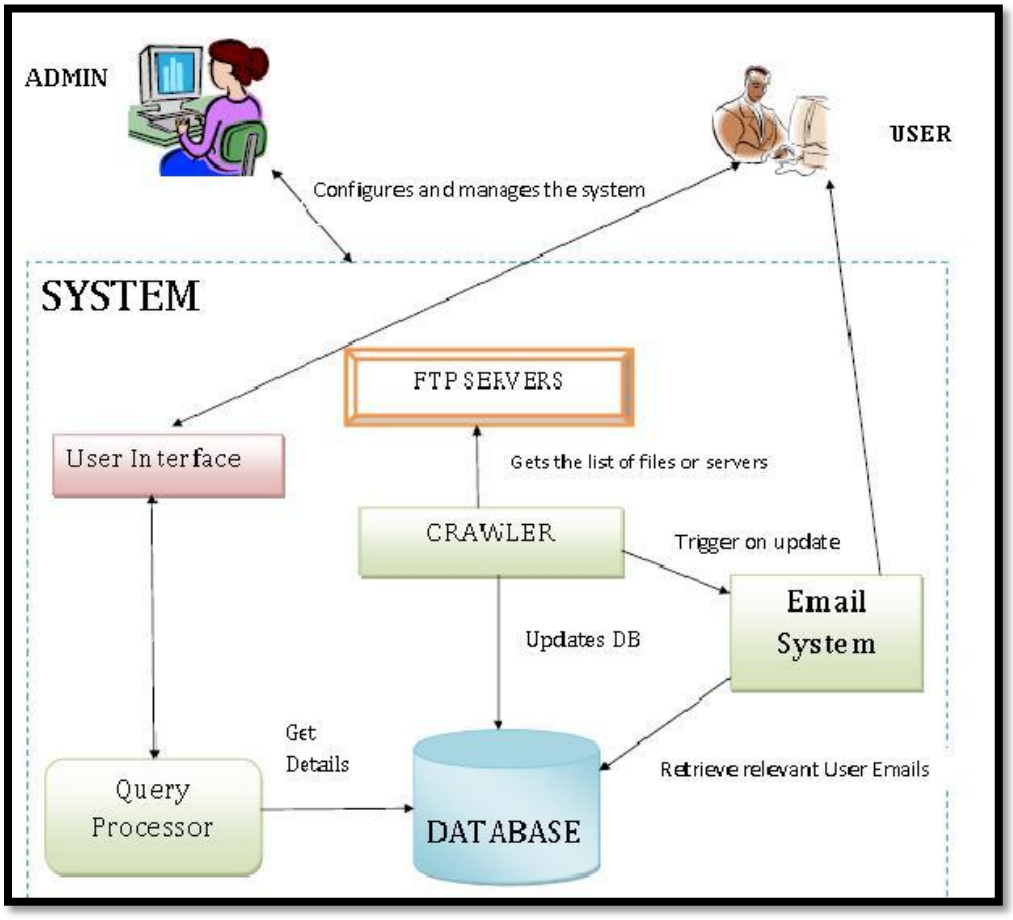


Figure 5.21: Workflow Block Diagram- File Search Module

5.3.1.1 Interfaces:

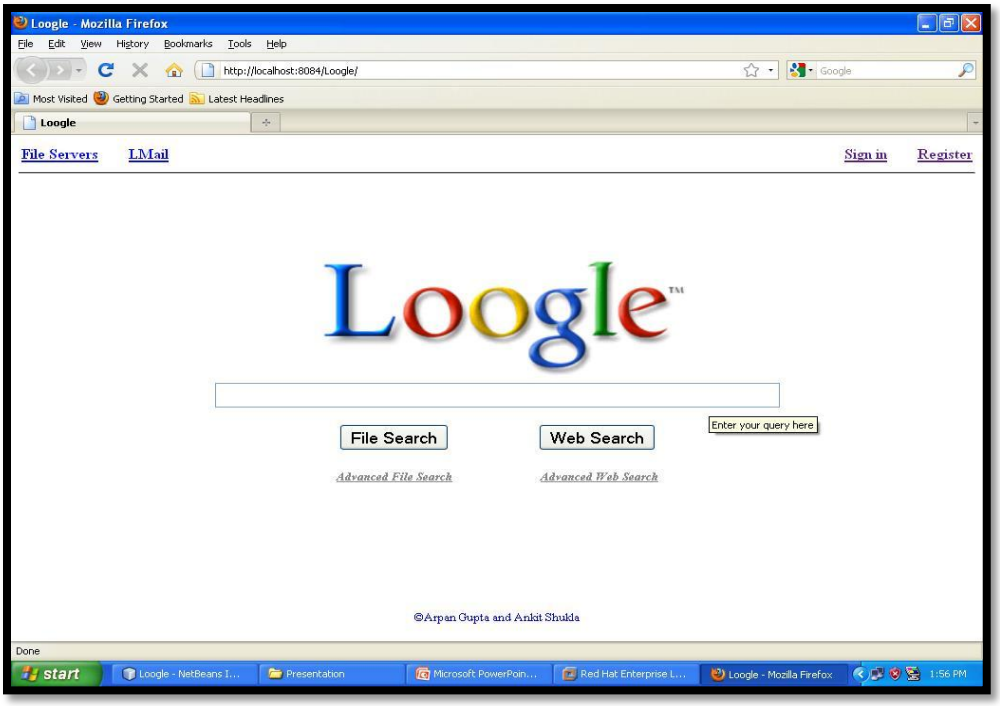


Figure 5.22: User interface screenshot- Loogle home page

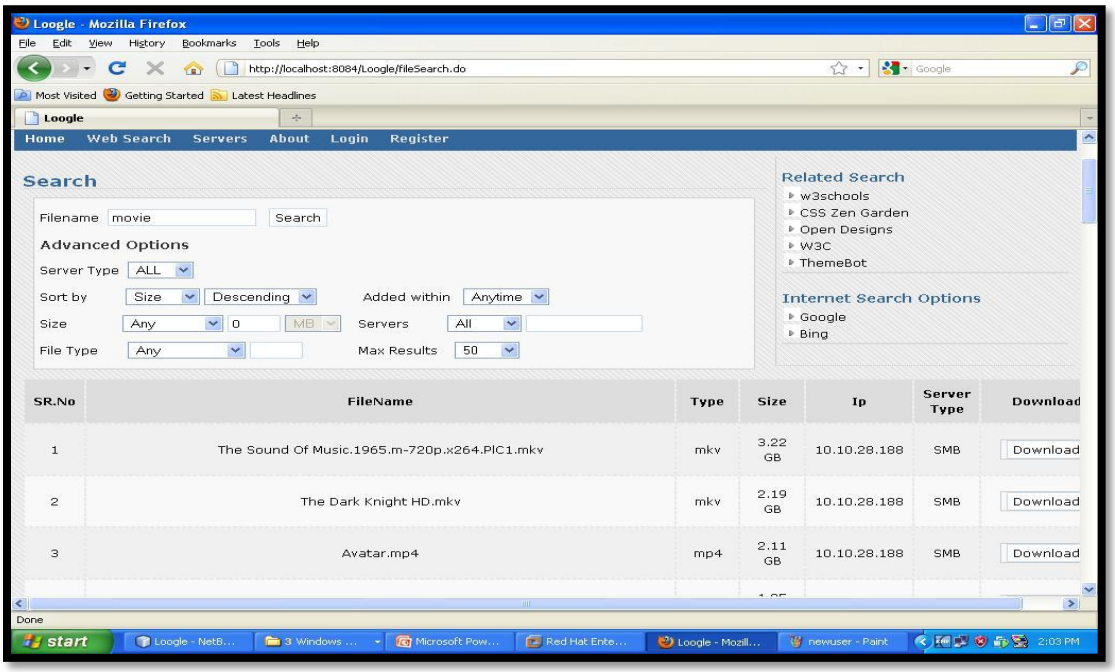


Figure 5.23: User interface screenshot- file search results

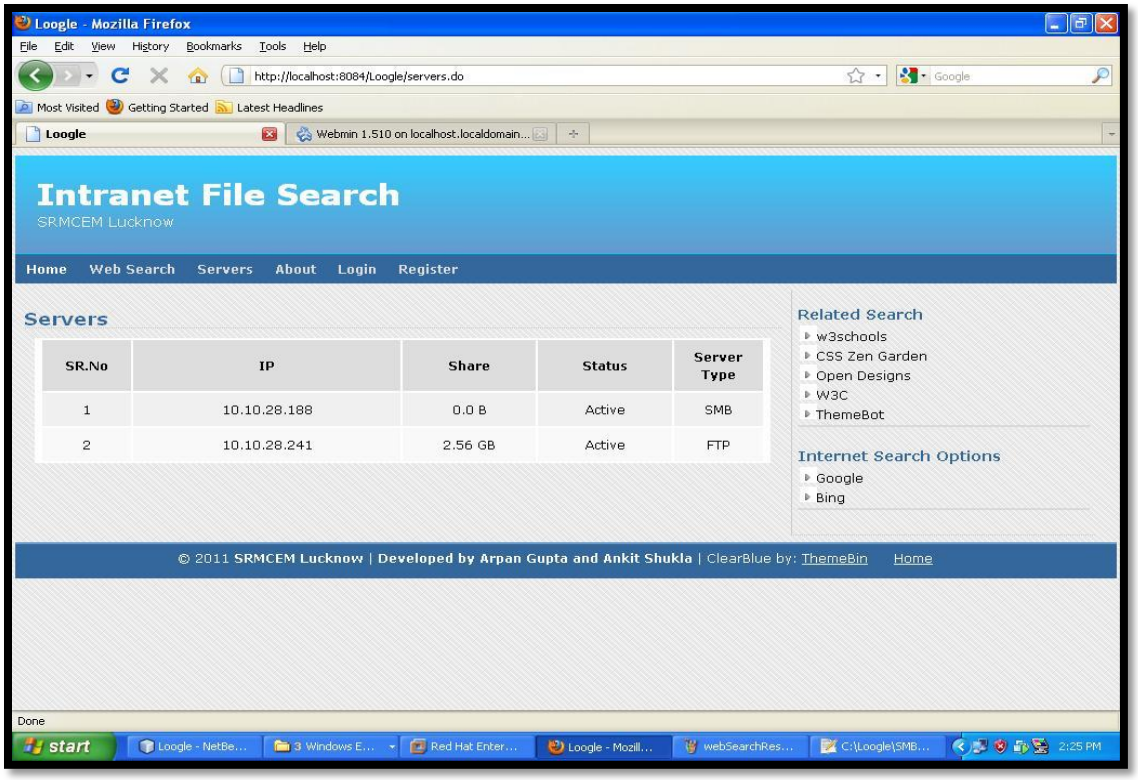


Figure 5.24: User interface screenshot- Server List

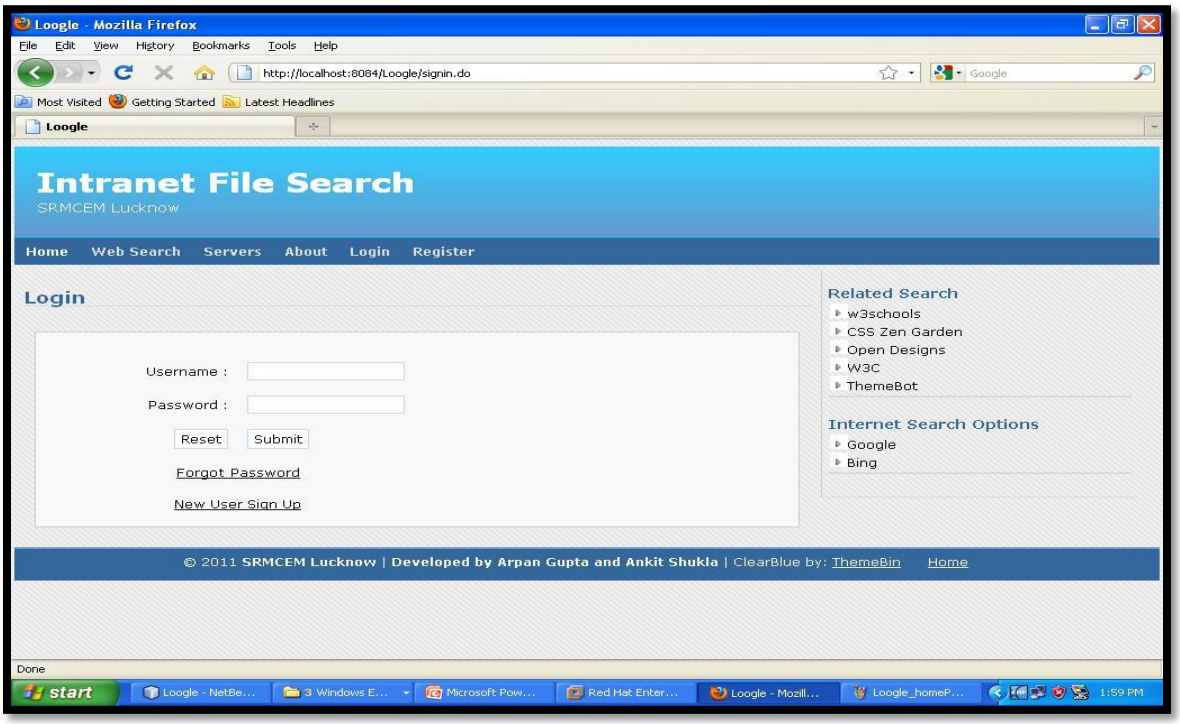


Figure 5.25: User interface screenshot- User Login

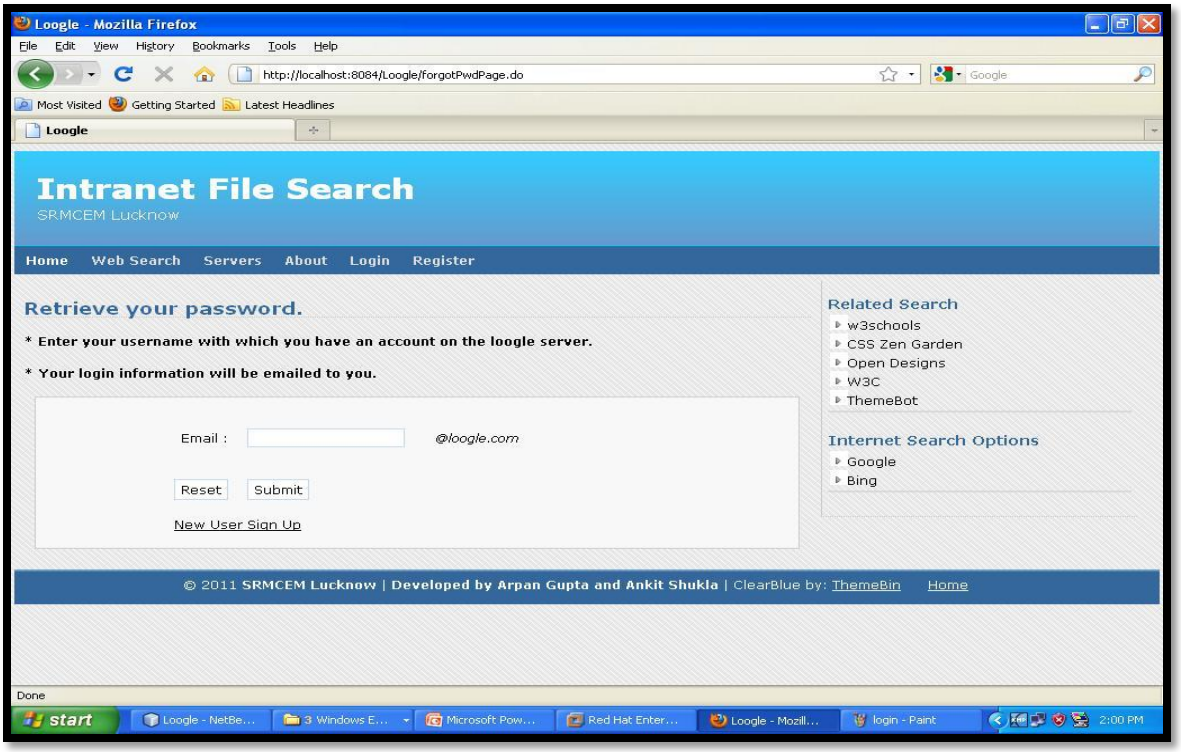


Figure 5.26: User interface screenshot- retrieve password

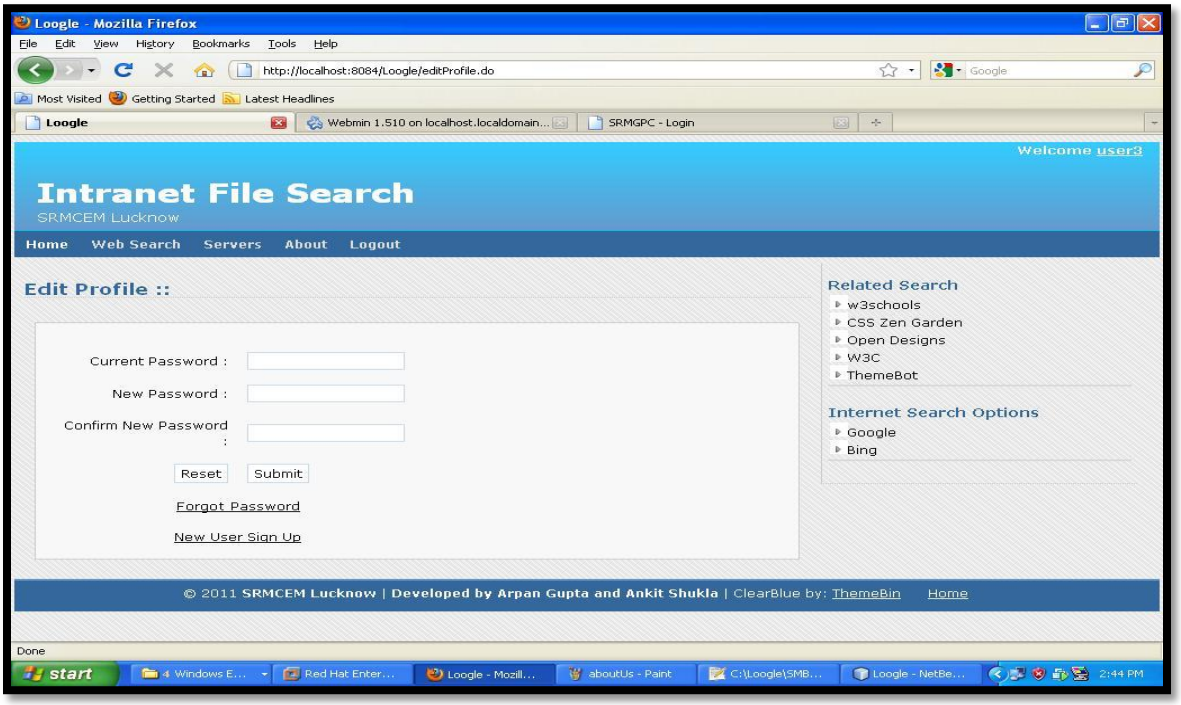


Figure 5.27: User interface screenshot- Edit Profile

5.3.1.2 Coding:

File Search :

index.jsp:

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<%@ page import="java.io.*,java.util.*,java.net.*" %>
<%@ page import="java.text.SimpleDateFormat" %>
<jsp:useBean id="queryBean" class="intranetFileSearch.Query" scope="request" />
<%!HttpSession session;
   String userID=null;
%>
<%
        session = request.getSession(true);
        if (session != null) {
           userID = (String) session.getAttribute("userID");
        }
%>
<%
        response.setHeader("Cache-Control", "no-cache"); //Forces caches to obtain a new copy
of the page from the origin server
        response.setHeader("Cache-Control", "no-store");//Directs caches not to store the page
under any circumstance
        response.setHeader("Pragma", "no-cache");//HTTP 1.0 backward compatibility
        response.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale
%>
<html>
   <head>
      <title>
         Loogle
      </title>
   </head>
   <body>
      <table width="100%" border="0">
         <tr>
            <td width="50%" align="left">
               <a href="servers.do"><font color="blue"><strong>File Servers</strong></font></a>
                     
               <a href="http://mail.loogle.com"><font
color="blue"><strong>LMail</strong></font></a>
            </td>
```

```
            <td align="right" width="50%">
               <%
               if (userID == null) {
               %>
               <font color="blue">
                  <a href="signin.do"><strong>Sign in</strong></a>
      
                  <a href="register.do"><strong>Register</strong></a>
               </font>
               <%          } else {
               %>
               <font color="blue">
                  <a href="signout.do"><strong>Logout</strong></a>
               </font>
               <%          }
               %>
            </td>
         </tr>
      </table>
      <hr style="height:auto;border-color:inherit">
      <br><br><br>
      <center>
         <!--
         <img src="images/skull1.jpg" width="50" height="80" alt="skull1"/>
         -->
         <br>
         <img src="images/loogle.png" width="400" height="150" alt="Loogle" title="Loogle"/>
         <br>
         <form action="" method="post" name="queryForm" id="queryForm">
            <input type="text" name="query" size="70" title="Enter your query here" style="font-
size:large"/>
            <br>
            <h3>
               <input type="submit" style="font-size:large" value="File Search" title="Searches the
local FTP and SMB servers"
onclick="document.getElementById('queryForm').setAttribute('action','fileSearch.do');"/>
             &
nbsp;   
               <input type="submit" style="font-size:large" value="Web Search" title="Searches
the local web servers"
onclick="document.getElementById('queryForm').setAttribute('action','webSearch.do');"/>
            </h3>
            <h5>
```

```
            <a href="fileSearch.do" style="font-size:small;color:#888"><i>Advanced File
Search</i></a>
                   
            
            <a href="webSearch.do" style="font-size:small;color:#888"><i>Advanced Web
Search</i></a>
          </h5>
        </form>
      </center>
      <br><br><br><br><br><br>
      <center>
        <font color="blue" size="2">
          &copy;Arpan Gupta and Ankit Shukla
        </font>
      </center>
  </body>
</html>
```

body.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<%@ page import="java.io.*,java.util.*,java.net.*" %>
<%@ page import="java.text.SimpleDateFormat" %>
<jsp:useBean id="queryBean" class="intranetFileSearch.Query" scope="request" />
<%
        response.setHeader("Cache-Control", "no-cache"); //Forces caches to obtain a new copy
of the page from the origin server
        response.setHeader("Cache-Control", "no-store");//Directs caches not to store the page
under any circumstance
        response.setHeader("Pragma", "no-cache");//HTTP 1.0 backward compatibility
        response.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale
%>
<%!
  String searchForm;
  List<intranetFileSearch.SearchResult> results;
  Map parameterMap;
%>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Loogle</title>
  </head>
```

```
   <body>
                    <div id="content">
                  <h1>Search</h1>
                  <%
                   out.println(searchForm);
                   //System.out.println(searchForm);
                   if(query!=null && query.compareTo("")!=0)
                   {
                      if(results==null)
                         out.println("<h3>Internal Server Error</h3>. An error occured while trying
to process your query. Please try after some time.<br />");
                       else
                       {
                         if(results.size()==0)
                            out.println("No results found.<br />");
                         else
                         {
                  %>
                  <table width=100% cellpadding=8px>
                     <tr bgcolor="#DDDDDD">
                        <th><center><b>SR.No</b></center></th>
                        <th><center><b>FileName</b></center></th>
                        <th><center><b>Type</b></center></th>
                        <th><center><b>Size</b></center></th>
                        <th><center><b>Ip</b></center></th>
                        <th><center><b>Server Type</b></center></th>
                        <th><center><b>Download</b></center></th>
                     </tr>
                        </table>
                        <%
                        }
                     }
                   }
                   %>
                     </div>
   </body>
</html>
```

header.jsp:

```
<%@ page pageEncoding="UTF-8" %>
<%@ page import="java.text.SimpleDateFormat" %>
<%@taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<%!HttpSession session;
```

```
   String userID;
%>
<%
        session = request.getSession(true);
        if (session != null) {
           userID = (String) session.getAttribute("userID");
        }
%>
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <link rel="stylesheet" href="images/style.css" type="text/css" media="all"  />
      <title>Loogle</title>
   </head>
   <body>
      <script type="text/javascript" language="javascript">
         var browserType;
         var  flag= 0;
         if (document.layers) {browserType = "nn4"}
         if (document.all) {browserType = "ie"}
         if (window.navigator.userAgent.toLowerCase().match("gecko")) {
            browserType= "gecko"
         }
         function hide2(divName) {
            if (browserType == "gecko" )
               document.poppedLayer = eval('document.getElementById(divName)');
            else if (browserType == "ie")
               document.poppedLayer = eval('document.getElementById(divName)');
            else
               document.poppedLayer = eval('document.layers[divName]');
            document.poppedLayer.style.display = "none";
         }
         function show2(divName) {
            if (browserType == "gecko" )
               document.poppedLayer = eval('document.getElementById(divName)');
            else if (browserType == "ie")
               document.poppedLayer = eval('document.getElementById(divName)');
            else
               document.poppedLayer = eval('document.layers[divName]');
            document.poppedLayer.style.display = "inline";
         }
         function toggle(divName){
            if(flag == 0) show2(divName);
            else hide2(divName);
```

```
            flag = 1-flag;
        }
        function sizeQueryTypeChange(selectBox)
        {
            if(selectBox.options[selectBox.selectedIndex].value == "any")
            {
                document.searchForm.sizeValue.disabled = true;
                document.searchForm.sizeUnit.disabled = true;
            }
            else
            {
                document.searchForm.sizeValue.disabled = false;
                document.searchForm.sizeUnit.disabled = false;
            }
        }
        function serversQueryChange(selectBox)
        {
            if(selectBox.options[selectBox.selectedIndex].value == "ip")
                document.searchForm.serversValue.disabled = false;
            else
                document.searchForm.serversValue.disabled = true;
        }
        function fileTypeChange(selectBox)
        {
            if(selectBox.options[selectBox.selectedIndex].value == "extension")
                document.searchForm.fileExtension.disabled = false;
            else
                document.searchForm.fileExtension.disabled = true;
        }
    </script>
    <div id="header">
        <%
            if(userID!=null)
            {
        %>
        <div align="right">
            <font size="2">
                <b>Welcome <a href="editProfile.do" style="color:white" title="Edit your
profile"><%=userID%></a></b>   
            </font>
        </div>
                <%--
                <a href="editProfile.do" style="color:white" title="Edit your
profile"><%=userID%></a>
```

```
            --%>
        <%
           }
        %>
        <h1 id="logo"><a href="index.jsp" style="color:white;text-decoration:none;">Intranet
File Search</a></h1>
        <div id="slogan"><a href="index.jsp" style="color:white;text-
decoration:none;">SRMCEM Lucknow</a></div>
      </div>
      <div id="nav">
        <div id="nbar">
          <ul>
            <li id="selected"><a href="fileSearch.do">Home</a></li>
            <li><a href="webSearch.do">Web Search</a></li>
            <li><a href="servers.do">Servers</a></li>
            <li><a href="about.do">About</a></li>
            <%
        if (userID == null) {
            %>
            <%--
            <li><a onclick="toggle('login')" style="cursor:pointer">Login</a></li>
            --%>
            <li><a href="signin.do">Login</a></li>
            <li><a href="register.do">Register</a></li>
            <%  } else {
            %>
            <li><a href="signout.do">Logout</a></li>
            <%          }
            %>
          </ul>
        </div>
      </div>
   </body>
</html>
```

footer.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <link rel="stylesheet" href="images/style.css" type="text/css" media="all"/>
      <title>Loogle</title>
```

```
   </head>
   <body>
      <div id="footer"> &copy; 2011 <strong>SRMCEM Lucknow</strong> |
<strong>Developed by Arpan Gupta and Ankit Shukla </strong> | ClearBlue  by: <a
href="http://www.themebin.com/">ThemeBin</a>
                     <a href="fileSearch.do">Home</a>
                    
        </div>
   </body>
</html>
```

rightmenu.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html>
   <head>
      <link rel="stylesheet" href="images/style.css" type="text/css" media="all"/>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Loogle</title>
   </head>
   <body>
            <div id="sidebar">
               <div id="left_sidebar">
                  <div class="hotlinks">
                     <h1>Related Search</h1>
                     <ul class="menu">
                        <li><a href="http://w3schools.com">w3schools</a><a
href="http://partners.ipower.com/z/57/CD5822/"></a></li>
                        <li><a href="http://csszengarden.com">CSS Zen Garden</a></li>
                        <li><a href="http://opendesigns.org/">Open Designs</a><a
href="http://www.fotolia.com/partner/114283"></a></li>
                        <li><a href="http://www.w3.com/">W3C</a><a
href="http://www.bigstockphoto.com/?refid=grKPpdNw6k"></a></li>
                        <li><a href="http://themebot.com">ThemeBot</a></li>
                     </ul>
                  </div>
                  <div class="hotlinks">
                     <h1>Internet Search Options</h1>
                     <ul class="menu">
                        <li><a href="http://www.google.com">Google</a></li>
                        <li><a href="http://www.bing.com">Bing</a></li>
                     </ul>
                  </div>
```

```
            </div>
          </div>
    </body>
</html>
```

QueryProcessor.java

```
package intranetFileSearch;

import java.io.FileNotFoundException;
import java.io.InputStream;
import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.PreparedStatement;
import java.util.Properties;
/**
 *
 * @author Arpan Gupta
 */
public class QueryProcessor {
    private String dbURL;
    private String dbDriver;
    private Connection dbCon;
    private String username;
    private String password;
    public QueryProcessor() {
        dbURL = "";
        dbDriver = "";
        username = "";
        password = "";
        dbCon = null;
    }
    public Connection getdbCon() {
        return dbCon;
    }
    public void setdbCon(final Connection con) {
        dbCon = con;
    }
    public void setdbURL(final String url) {
        dbURL = url;
    }
```

```
public void setdbDriver(final String driver) {
   dbDriver = driver;
}
public void setUsername(final String user) {
   username = user;
}
public void setPassword(final String pass) {
   password = pass;
}
public String getdbURL() {
   return dbURL;
}
public String getdbDriver() {
   return dbDriver;
}
public String getUsername() {
   return username;
}
public String getPassword() {
   return password;
}
public int updateData(final String query) throws SQLException {
   System.err.println("QueryProcessor.java : Received query : " + query);
   Statement s = dbCon.createStatement();
   int num = s.executeUpdate(query);
   s.close();
   return num;
}
public ResultSet getRecords(final String query) throws SQLException {
   Statement s = dbCon.createStatement();
   ResultSet rs = s.executeQuery(query);
   return rs;
}
public ResultSet getLoginRecords(final String query)throws SQLException
{
   PreparedStatement pstmt=dbCon.prepareStatement(query);
   ResultSet rs=pstmt.executeQuery();
   return rs;
}
public void close() throws SQLException {
   dbCon.close();
}
public static void main(String args[]) {
   String pwd = System.getProperty("user.dir");
```

```
        System.out.println("Present Working Directory : " + pwd);
        QueryProcessor qp = new QueryProcessor();
        try {
            qp.init();
        } catch (FileNotFoundException f) {
            System.err.println("QueryProcessor.java : Could not find the database parameters file\n" +
f);
        } catch (IOException ioe) {
            System.err.println("QueryProcessor.java : Could not read from database parameters
file\n" + ioe);
        }
        System.err.println(qp.dbDriver + " " + qp.dbURL + " " + qp.username + " " + qp.password);
        try {
            qp.connect();
        } catch (InstantiationException ie) {
            System.err.println("taskProcessor.java : #Could not connect to database # " + ie);
        } catch (ClassNotFoundException c) {
            System.err.println("taskProcessor.java : #Could not connect to database # " + c);
        } catch (SQLException s) {
            System.err.println("taskProcessor.java : #Could not connect to database # " + s);
        } catch (IllegalAccessException i) {
            System.err.println("taskProcessor.java : #Could not connect to database # " + i);
        }
        try {
            ResultSet rs = qp.getRecords("select * from Servers;");
            rs.close();
        } catch (SQLException s) {
            System.err.println("taskProcessor.java : #Could not connect to database # " + s);
        }
        try {
            qp.close();
        } catch (SQLException s) {
            System.err.println("taskProcessor.java : #Could not connect to database # " + s);
        }
    }
}
```

Servers.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8" import="java.util.*"%>
<%@ page import="java.io.*" %>
<%@ page import="java.net.*" %>
<%@ page import="java.text.SimpleDateFormat" %>
<%@ page import="java.util.*" %>
 <jsp:useBean id="queryBean" class="intranetFileSearch.Query" scope="request" />
```

```
<%!HttpSession session;
   String user;
%>
<%
        session = request.getSession(true);
        if (session != null) {
           user = (String) session.getAttribute("userID");
        }
%>
<%
        response.setHeader("Cache-Control", "no-cache"); //Forces caches to obtain a new copy
of the page from the origin server
        response.setHeader("Cache-Control", "no-store");//Directs caches not to store the page
under any circumstance
        response.setHeader("Pragma", "no-cache");//HTTP 1.0 backward compatibility
        response.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale
%>
<!DOCTYPEs HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <link rel="stylesheet" href="images/style.css" type="text/css" media="all"  />
      <title>Loogle</title>
   </head>
   <body>
               <div id="content" >
               <%
                  Vector <String []> results = queryBean.getServerList();
                  if(results==null)
                  {
               %>
                  <h2>Error</h2>
                  <h4> Internal Server Error </h4>
                  <p> Could not retrieve the list of servers. Please retry after some time.</p>
                  <br />
               <%
                  }
                  else
                  {
                %>
                     <h1> Servers </h1>
                     <%
                        if(results.size()==0)
```

```
    {
       out.println("<h5> No servers found.</h5><br />)");
    }
    else
    {
%>
<form action="UpdateSubscribe.do" class="special" method="post">
 <table align="center" width="100%" cellpadding="8px">
    <tr bgcolor="#DDDDDD">
       <th><center><b>SR.No</b></center></th>
       <th><center><b>IP</b></center></th>
       <th><center><b>Share</b></center></th>
       <th><center><b>Status</b></center></th>
       <th><center><b>Server Type</b></center></th>
       <%
       if(user!=null)
          {
       %>
       <th><center><b>Subscribe</b></center></th>
       <%
          }
       %>
    </tr>
    <%
       boolean colour=true;
       for(int i=0; i<results.size(); i++)
       {
          if(colour)
          {
             out.println("<tr bgcolor=\"#F0F0F0\">");
             colour=false;
          }
          else
          {
             colour=true;
             out.println("<tr bgcolor=\"#F9F9F9\">");
          }
          String []row=results.elementAt(i);
          int k=i+1;
          out.println("<td width=\"10%\"><center>"+k+" </center></td>");
          out.println("<td width=\"30%\"><center>"+row[0]+"</center></td>");
          out.println("<td width=\"15%\"><center>"+row[1]+"</center></td>");
          out.println("<td width=\"15%\"><center>"+row[2]+"</center></td>");
          out.println("<td width=\"10%\"><center>"+row[3]+"</center></td>");
```

```
                        String key=null;
                        if(row[3].compareTo("FTP")==0)
                           key=row[0]+":"+"0";
                        if(row[3].compareTo("SMB")==0)
                           key=row[0]+":"+"1";
                        //out.println("<td width=\"10%\"><center><input type=\"checkbox\"
name=\""+key+":0\" ></center></td>");
                        if(user!=null)
                           out.println("<td width=\"10%\"><center><input type=\"checkbox\"
name=\""+key+":1\" ></center></td>");
                        out.println("</tr>");
                     }
                  %>
               </table>
               <%
               if(user!=null)
                  {
               %>
               <table bgcolor="#FFFFFF" border="0" align="center" width="100%">
                  <tr>
                     <td width=50%>
                        <center>
                           <input type="reset" value="reset"/>
                         </center>
                     </td>
                     <td width=50%>
                        <center>
                            <input type="submit" value="Submit"/>
                        </center>
                     </td>
                  </tr>
                  <tr><td></td> </tr>
               </table>
               <%
                  }
               %>
            </form>
            <br />
         <%
               }
            }
         %>
         </div>
   </body>
```

```
</html>
```

loginpage.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8"
import="java.text.SimpleDateFormat"%>
<%@taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<%
        response.setHeader("Cache-Control", "no-cache"); //Forces caches to obtain a new copy
of the page from the origin server
        response.setHeader("Cache-Control", "no-store");//Directs caches not to store the page
under any circumstance
        response.setHeader("Pragma", "no-cache");//HTTP 1.0 backward compatibility
        response.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale
%>
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <link rel="stylesheet" href="images/style.css" type="text/css" media="all"  />
      <title>Loogle</title>
   </head>
   <body>
      <div id="content">
         <h1>Login</h1>
         <br>
         <html:form action="Login.do" method="post" >
            <table width="100%" cellspacing="15px">
               <tr>
                  <td>
                  </td>
               </tr>
               <tr>
                  <td align="right" width="25%">
                     Username :
                  </td>
                  <td align="left" width="25%">
                     <html:text property="username" />
                  </td>
                  <td align="left" width="50%">
                     <html:errors property="username"/>
                  </td>
               </tr>
```

```
            <tr>
              <td align="right" width="25%">
                Password :
              </td>
              <td align="left" width="25%">
                <html:password property="password" />
              </td>
              <td align="left" width="50%">
                <html:errors property="password"/>
              </td>
            </tr>
        <%
          java.util.Date today = new java.util.Date();
          SimpleDateFormat s = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
          String timestamp = s.format(today);
        %>
        <html:hidden property="timestamp" value='<%=timestamp%>' />
            <tr>
              <td align="right" width="25%">
                <html:reset value="Reset" />
              </td>
              <td align="left" width="25%">
                <html:submit value="Submit" />
              </td>
              <td width="50%"></td>
            </tr>
            <tr>
              <td align="center" colspan="2" width="25%">
                <a href="forgotPwdPage.do">
                  <font style="font-size:small">Forgot Password</font>
                </a>
              </td>
            </tr>
            <tr>
              <td align="center" colspan="2" width="25%">
                <a href="register.do">New User Sign Up</a>
              </td>
              <td width="50%"></td>
            </tr>
          </table>
    </html:form>
      </div>
    </body>
</html>
```

logoutpage.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Loogle</title>
   </head>
   <body>
      <div id="content">
      <h1>You have successfully logged out.</h1>
      </div>
   </body>
</html>
```

logoutFailPage.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Loogle</title>
   </head>
   <body>
      <div id="content">
      <h1>You are already logged out.</h1>
      </div>
   </body>
</html>
```

forgotpwd.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8"
import="java.text.SimpleDateFormat"%>
<%@taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<%
        response.setHeader("Cache-Control", "no-cache"); //Forces caches to obtain a new copy
of the page from the origin server
        response.setHeader("Cache-Control", "no-store");//Directs caches not to store the page
under any circumstance
```

```
        response.setHeader("Pragma", "no-cache");//HTTP 1.0 backward compatibility
        response.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale
%>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <link rel="stylesheet" href="images/style.css" type="text/css" media="all"  />
    <title>Loogle</title>
  </head>
  <body>
    <div id="content">
      <h1>Retrieve your password.</h1>
      <h4>* Enter your username with which you have an account on the loogle server.</h4>
      <h4>* Your login information will be emailed to you.</h4>
      <html:form action="forgotPwd.do" method="post" >
        <table width="100%" cellspacing="15px">
          <tr>
            <td>
            </td>
          </tr>
          <tr>
            <td align="right" width="25%">
              Email :
            </td>
            <td align="left" width="25%">
              <html:text property="email" />
            </td>
            <td align="left" width="50%">
              <i>@loogle.com</i>
            </td>
          </tr>
          <tr>
            <td colspan="2" width="50%" align="center">
              <html:errors property="email"/>
            </td>
          </tr>
          <tr>
            <td align="right" width="25%">
              <html:reset value="Reset" />
            </td>
            <td align="left" width="25%">
              <html:submit value="Submit" />
            </td>
            <td width="50%"></td>
```

```
            </tr>
            <tr>
               <td align="center" colspan="2" width="25%">
                  <a href="register.do">New User Sign Up</a>
               </td>
               <td width="50%"></td>
            </tr>
         </table>
   </html:form>
      </div>
   </body>
</html>
```

pwdRetrieved:

```
<%@page contentType="text/html" pageEncoding="UTF-8" %>
<%@taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Loogle</title>
   </head>
   <body>
      <div id="content">
         <table width="100%" border="0">
            <tr>
               <td>
                  <h1>Your login information was retrieved successfully.</h1>
                  <ul>
                     <li>To access your email account click here :: <a
href="http://mail.loogle.com">Lmail</a></li>
                  </ul>
               </td>
            </tr>
         </table>
      </div>
   </body>
</html>
```

pwdNotRetrieved.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8" %>
<%@taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Loogle</title>
   </head>
   <body>
      <div id="content">
         <table width="100%" border="0">
            <tr>
               <td>
                  <h2>Your Login information was not retrieved due to any of the following
reasons:</h2>
                  <ul>
                     <li>You do not have a login account on the loogle server or your account has
been deleted.</li>
                     <li>There may be database connection error.</li>
                     <li>Network Connection error.</li>
                     <li>You did not provide a valid input.</li>
                  </ul>
               </td>
            </tr>
         </table>
      </div>
   </body>
</html>
```

editProfile.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8"
import="java.text.SimpleDateFormat"%>
<%@taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<%
        response.setHeader("Cache-Control", "no-cache"); //Forces caches to obtain a new copy
of the page from the origin server
        response.setHeader("Cache-Control", "no-store");//Directs caches not to store the page
under any circumstance
        response.setHeader("Pragma", "no-cache");//HTTP 1.0 backward compatibility
        response.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale
%>
<html>
   <head>
```

```
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <link rel="stylesheet" href="images/style.css" type="text/css" media="all"  />
    <title>Loogle</title>
</head>
<body>
    <div id="content">
        <h1>Edit Profile ::</h1>
        <br>
        <html:form action="changePwd.do" method="post" >
            <table width="100%" cellspacing="15px">
                <tr>
                    <td>
                    </td>
                </tr>
                <tr>
                    <td align="right" width="25%">
                        Current Password :
                    </td>
                    <td align="left" width="25%">
                        <html:text property="currPassword" />
                    </td>
                    <td align="left" width="50%">
                        <html:errors property="currPassword"/>
                    </td>
                </tr>
                <tr>
                    <td align="right" width="25%">
                        New Password :
                    </td>
                    <td align="left" width="25%">
                        <html:password property="newPassword" />
                    </td>
                    <td align="left" width="50%">
                        <html:errors property="newPassword"/>
                    </td>
                </tr>
                <tr>
                    <td align="right" width="25%">
                        Confirm New Password :
                    </td>
                    <td align="left" width="25%">
                        <html:password property="cNewPassword" />
                    </td>
                    <td align="left" width="50%">
```

```
                <html:errors property="cNewPassword"/>
              </td>
            </tr>
            <tr>
              <td align="right" width="25%">
                <html:reset value="Reset" />
              </td>
              <td align="left" width="25%">
                <html:submit value="Submit" />
              </td>
              <td width="50%"></td>
            </tr>
            <tr>
              <td align="center" colspan="2" width="25%">
                <a href="forgotPwdPage.do">
                  <font style="font-size:small">Forgot Password</font>
                </a>
              </td>
            </tr>
            <tr>
              <td align="center" colspan="2" width="25%">
                <a href="register.do">New User Sign Up</a>
              </td>
              <td width="50%"></td>
            </tr>
          </table>
    </html:form>
      </div>
    </body>
</html>
```

pwdChangeSuccess.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8" isErrorPage="true"%>
<%@taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Loogle</title>
  </head>
  <body>
    <div id="content">
```

```
        <table width="100%" border="0">
          <tr>
            <td>
              <h1>Your Password has been changed successfully !</h1>
            </td>
          </tr>
        </table>
      </div>
    </body>
</html>
```

pwdChangeFail.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8" isErrorPage="true"%>
<%@taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Loogle</title>
   </head>
   <body>
      <div id="content">
        <table width="100%" border="0">
          <tr>
            <td>
              <h1>Error in changing password! </h1>
              <h3><a href="editProfile.do">Try again</a></h3>
            </td>
          </tr>
        </table>
      </div>
   </body>
</html>
```

downloadsuccess.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8" %>
<%@taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Loogle</title>
```

```
  </head>
  <body>
    <div id="content">
      <table width="100%" border="0">
        <tr>
          <td>
            <h1>The file has been successfully downloaded!</h1>
          </td>
        </tr>
      </table>
    </div>
  </body>
</html>
```

downloaderror.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8" isErrorPage="true"%>
<%@taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Loogle</title>
  </head>
  <body>
    <div id="content">
      <table width="100%" border="0">
        <tr>
          <td>
            <h1>The file cannot be downloaded due to one of the following reasons:</h1>
            <ul>
              <li><h4>The file has been removed from the source path.</h4></li>
              <li><h4>You may be trying to download a directory.</h4></li>
              <li><h4>Access to the file is denied.</h4></li>
              <li><h4>Network Connection problem.</h4></li>
            </ul>
          </td>
        </tr>
      </table>
    </div>
  </body>
</html>
```

5.3.2 Web Search:

The following functionalities are being provided in this module:

- Administrator supplies a set of starting URLs in a config file.
- Crawler starts crawling these URLs recursively.
- Only whitelisted URLs are crawled.
- Content of HTML files and text files files are indexed.
- Users can query the indexes using a web UI built in Java Servlets and Java Server Pages.
- Administrators can censor content using stopwords.
- Some URLs can be blacklisted. These URLs are not crawled.
- Admin can specify blacklisted and whitelisted urls in a config file.
- A scheduler takes care of periodic crawling activities.

Components

- **Crawler**[11]

  The crawler is the name of the component of the web search which is responsible for getting content out of all HTML pages specified on the intranet. The crawler is supposed to:

  - Fetch the list of hyperlinks on the page and then recursively crawl those links
  - Get the content on each page visited neglecting the html tags.
  - Maintain a list of pages visited, so that no page is crawled more than once (Not doing this may lead the crawler going into an indefinite crawl cycle).
  - The administrator is supposed to give as input list of IPs within the intranet. The crawler then recursively scans the network on the specified links to get content out of the pages.

Included Features

- Ability to configure the IPs to scan on the intranet.
- Blacklisting some IPs and keywords during configuration. The blacklisted IPs are never crawled for data and the results from blacklisted sites are never shown in the search result.

- **Indexer and Query Parser**

The indexer is the heart of a search engine. The indexer takes pages as input from the crawler and indexes them. It extracts keywords out of the page and makes index entries for that page using the keywords. Before words are stored to the index they are converted to their base form using analyzers. Query parser parses user queries and uses the index generated by the indexer to execute the query and return the results.

For indexing and parsing queries we use a popular open source information retrieval library, Apache Lucene[10].

Apache Lucene[13] is a high-performance, full-featured text search engine library written entirely in Java. It is a technology suitable for nearly any application that requires full-text search, especially cross-platform. Lucene features a powerful query syntax, fast indexing, fast searching, sorting of results by relevance or other fields, etc. It stores its index as files on the disk. These index files are used by Lucene's query processor to execute queries and return results.

Lucene uses the Vector Space Model for indexing text documents. Vector Space Model is an algebraic model for representing text documents (and any objects, in general) as vectors of identifiers, such as, for example, index terms. Each dimension in the vector corresponds to a separate term. If a term occurs in the document, its value in the vector is non-zero. Several different ways of computing these values, also known as (term) weights, have been developed. One of the best known schemes is tf-idf (term frequency - index document frequency) weighting and this is the one used in Lucene.

Relevancy rankings of documents in a keyword search can be calculated, using the assumptions of document similarities theory, by comparing the deviation of angles between each document vector and the original query vector where the query is represented as same kind of vector as the documents. Lucene itself is just an indexing and search library and does not contain crawling and HTML parsing functionality. Crawling and HTML parsing is done by our crawler.

- **UI component**

  The View of the project has been abstracted from the search component. The View (UI) part of the project only gets an object containing the list of search results or error messages and the UI component is responsible for rendering it. The UI part is easily configurable and the design of the page can be changed very easily at any point of time as it has no other component is dependent on it (UI component is completely abstracted out).

- **Scheduler**

  Our project is mostly dependent on the scheduled scanning of the intranet. For achieving the periodic scanning the **Quartz**[12] API is used and the scanning is done after specified time automatically. The automatic crawling can only be started by the administrator after he has specified the scanning frequency in the configuration file on the server.

Workflow: The workflow of the web search module can be depicted with the help of the following diagram:

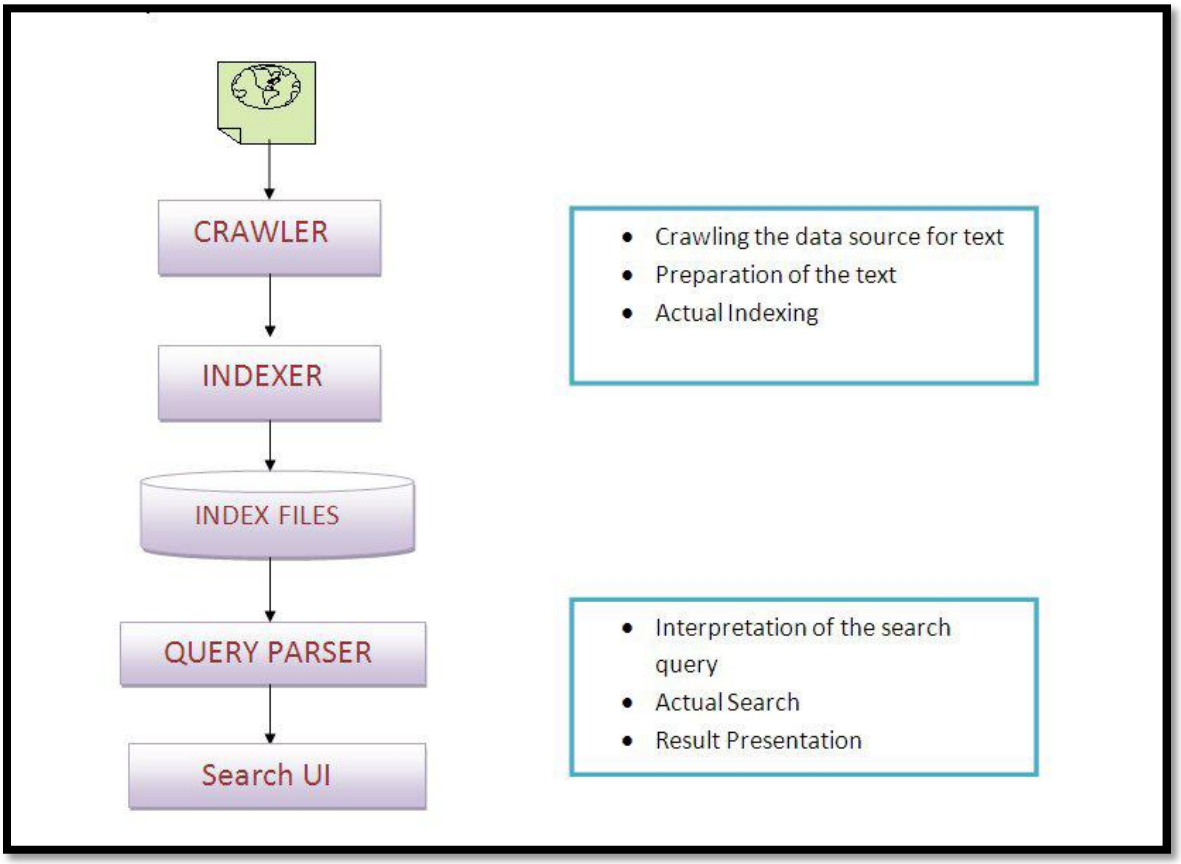


Figure 5.28: Workflow of web search module

5.3.2.1 Interfaces:

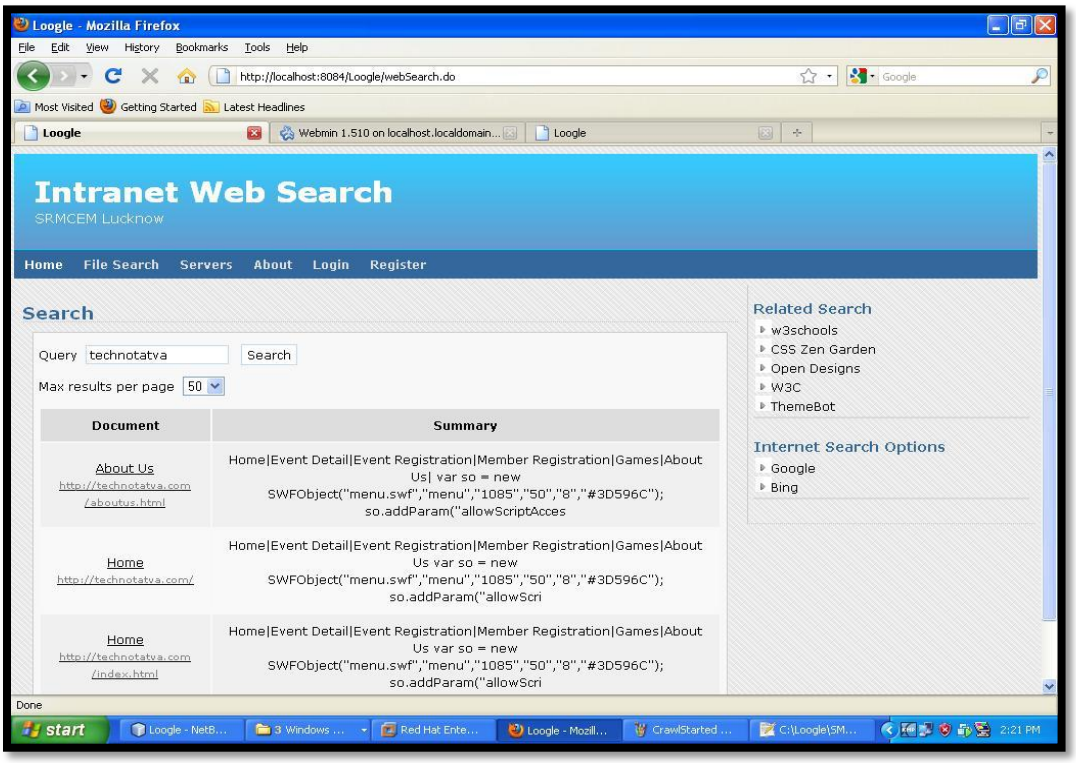


Figure 5.29: User interface screenshot- Web search results

5.3.2.2 Coding:

headerweb.jsp:

```
<%@ page pageEncoding="UTF-8" %>
<%@ page import="java.text.SimpleDateFormat" %>
<%@taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<%!HttpSession session;
   String userID;%>
<%
        session = request.getSession(true);
        if (session != null) {
           userID = (String) session.getAttribute("userID");
        }
%>
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <link rel="stylesheet" href="images/style.css" type="text/css" media="all"  />
      <title>Loogle</title>
```

```
</head>
<body>
  <script type="text/javascript" language="javascript">
    var browserType;
    var  flag= 0;
    if (document.layers) {browserType = "nn4"}
    if (document.all) {browserType = "ie"}
    if (window.navigator.userAgent.toLowerCase().match("gecko")) {
      browserType= "gecko"
    }
    function hide2(divName) {
      if (browserType == "gecko" )
        document.poppedLayer = eval('document.getElementById(divName)');
      else if (browserType == "ie")
        document.poppedLayer = eval('document.getElementById(divName)');
      else
        document.poppedLayer = eval('document.layers[divName]');
      document.poppedLayer.style.display = "none";
    }
    function show2(divName) {
      if (browserType == "gecko" )
        document.poppedLayer = eval('document.getElementById(divName)');
      else if (browserType == "ie")
        document.poppedLayer = eval('document.getElementById(divName)');
      else
        document.poppedLayer = eval('document.layers[divName]');
      document.poppedLayer.style.display = "inline";
    }
    function toggle(divName){
      if(flag == 0) show2(divName);
      else hide2(divName);
      flag = 1-flag;
    }
    function sizeQueryTypeChange(selectBox)
    {
      if(selectBox.options[selectBox.selectedIndex].value == "any")
      {
        document.searchForm.sizeValue.disabled = true;
        document.searchForm.sizeUnit.disabled = true;
      }
      else
      {
        document.searchForm.sizeValue.disabled = false;
        document.searchForm.sizeUnit.disabled = false;
```

```
            }
        }
        function serversQueryChange(selectBox)
        {
            if(selectBox.options[selectBox.selectedIndex].value == "ip")
                document.searchForm.serversValue.disabled = false;
            else
                document.searchForm.serversValue.disabled = true;
        }
        function fileTypeChange(selectBox)
        {
            if(selectBox.options[selectBox.selectedIndex].value == "extension")
                document.searchForm.fileExtension.disabled = false;
            else
                document.searchForm.fileExtension.disabled = true;
        }
    </script>
    <div id="header">
        <%
            if(userID!=null)
            {
        %>
        <div align="right">
            <font size="2"><b>Welcome <a href="editProfile.do" style="color:white" title="Edit
your profile"><%=userID%></a></b>   </font>
        </div>
        <%
            }
        %>
        <h1 id="logo"><a href="index.jsp" style="color:white;text-decoration:none;">Intranet
Web Search</a></h1>
        <div id="slogan"><a href="index.jsp" style="color:white;text-
decoration:none;">SRMCEM Lucknow</a></div>
    </div>
    <div id="nav">
        <div id="nbar">
            <ul>
                <li id="selected"><a href="webSearch.do">Home</a></li>
                <li><a href="fileSearch.do">File Search</a></li>
                <li><a href="servers.do">Servers</a></li>
                <li><a href="aboutusweb.do">About</a></li>
                <%
        if (userID == null) {
                %>
```

```
                <li><a href="signinweb.do">Login</a></li>
                <li><a href="registerweb.do">Register</a></li>
                <%  } else {
                %>
                <li><a href="signout.do">Logout</a></li>
                <%          }
                %>
              </ul>
            </div>
          </div>
    </body>
</html>
```

bodyweb.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<%@ page import="java.io.*,java.util.*,java.net.*" %>
<%@ page import="java.text.SimpleDateFormat" %>
<%@ page import = "javax.servlet.*, javax.servlet.http.*,org.apache.lucene.analysis.*,
org.apache.lucene.queryParser.*,  org.apache.lucene.demo.*,
org.apache.lucene.demo.html.Entities" %>
<jsp:useBean id="queryBean" class="intranetFileSearch.Query" scope="request" />
<%
        response.setHeader("Cache-Control", "no-cache"); //Forces caches to obtain a new copy
of the page from the origin server
        response.setHeader("Cache-Control", "no-store");//Directs caches not to store the page
under any circumstance
        response.setHeader("Pragma", "no-cache");//HTTP 1.0 backward compatibility
        response.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale
%>
<%!
   String searchForm;
   List<intranetFileSearch.SearchResult> results;
   Map parameterMap;
   String query="";
   public String escapeHTML(String s)
   {
      s = s.replaceAll("&", "&amp;");
      s = s.replaceAll("<", "&lt;");
      s = s.replaceAll(">", "&gt;");
      s = s.replaceAll("\"", "&quot;");
      s = s.replaceAll("'", "&apos;");
      return s;
```

```
   }
%>
<%
        results=null;
        parameterMap=request.getParameterMap();
        searchForm=intranetFileSearch.Search.genWebForm(parameterMap);
        query = request.getParameter("query");
%>
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Loogle</title>
   </head>
   <body>
      <div id="content">
      <h1>Search</h1>
<%
      out.println(searchForm);
      boolean error = false;              //used to control flow for error messages
      String indexName="path";  //path for the local system.
      IndexSearcher searcher = null;        //the searcher used to open/search the index
      Query queryWeb = null;                 //the Query created by the QueryParser
      Hits hits = null;                   //the search results
      int startindex = 0;                 //the first index displayed on this page
      int maxpage    = 10;                //the maximum items displayed on this page
      String queryString = null;            //the query entered in the previous page
      String startVal    = null;            //string version of startindex
      String maxresults  = null;            //string version of maxpage
      int thispage = 0;                   //used for the for/next either maxpage or
                                 //hits.length() - startindex - whichever is
                                 //less
      try
      {
        searcher = new IndexSearcher(indexName);      //create an indexSearcher for our page
                                       //NOTE: this operation is slow for large
                                       //indices (much slower than the search itself)
                                       //so you might want to keep an IndexSearcher
                                       //open

      } catch (Exception e)
      {                    //any error that happens is probably due
                                       //to a permission problem or non-existant
                                       //or otherwise corrupt index
%>
```

```
            <p>ERROR opening the Index - contact system admin!</p>

<%              error = true;                           //don't do anything up to the footer
      }
%>
<%
                                //did we open the index?
            queryString = request.getParameter("query");          //get the search criteria
            startVal    = request.getParameter("startat");        //get the start index
            maxresults  = request.getParameter("maxresults");     //get max results per page

            try
            {
                  maxpage    = Integer.parseInt(maxresults);    //parse the max results first
                  startindex = Integer.parseInt(startVal);      //then the start index
            } catch (Exception e) { } //we don't care if something happens we'll just start at 0
                                    //or end at 50
            if (queryString!=null)
            {
            Analyzer analyzer = new StandardAnalyzer();          //construct our usual analyzer
            try
            {
                  QueryParser qp = new QueryParser("contents", analyzer);
                  queryWeb = qp.parse(queryString); //parse the
            } catch (ParseException e) {                    //query and construct the Query
                                                  //object
                                                  //if it's just "operator error"
                                                  //send them a nice error HTML

               if(query.compareTo("")!=0)
               {
%>
                  <p>Error while parsing query.</p>
<%
               }
                  error = true;                           //don't bother with the rest of
                                                  //the page
            }
      }
            else
               error=true;
%>
<%
      if (error == false && searcher != null) {                 // if we've had no errors
```

```
                                        // searcher != null was to handle
                                        // a weird compilation bug
            thispage = maxpage;                     // default last element to maxpage
            hits = searcher.search(queryWeb);               // run the query
            if (hits.length() == 0) {                   // if we got no results tell the user
%>
            <p> Unable to find the query string. </p>
<%
            error = true;                       // don't bother with the rest of the
                                                // page
            }
        }
        if (error == false && searcher != null) {
%>
          <table width=100% cellpadding=8px>
            <tr bgcolor="#DDDDDD">
              <th><center><b>Document</b></center></th>
              <th><center><b>Summary</b></center></th>
            </tr>
<%
            if ((startindex + maxpage) > hits.length()) {
                thispage = hits.length() - startindex;      // set the max index to maxpage or last
            }                                   // actual search result whichever is less
            boolean colour=true;
            for (int i = startindex; i < (thispage + startindex); i++) {  // for each element

                if(colour){
                    out.println("<tr bgcolor=\"#F0F0F0\">");
                    colour=false;
                }
                else {
                    colour=true;
                    out.println("<tr bgcolor=\"#F9F9F9\">");
                }
                Document doc = hits.doc(i);             //get the next document
                String doctitle = doc.get("title");         //get its title
                String url = doc.get("path");               //get its path field
                if (url != null && url.startsWith("../webapps/")) { // strip off ../webapps prefix if
present
                    url = url.substring(10);
                }
                if ((doctitle == null) || doctitle.equals("")) //use the path if it has no title
                    doctitle = url;
                                        //then output!
```

```
                out.println("<td width=\"25%\"><center><a
href=\""+url+"\">"+doctitle+"</a><br><a href=\""+url+"\"><font size=1
color='grey'>"+url+"</font></a></center></td>");
                out.println("<td
width=\"75%\"><center>"+doc.get("summary")+"<center></td>");
                out.println("</tr>");
            }
%>
<%              if ( (startindex + maxpage) < hits.length()) {    //if there are more results...display
                                                    //the more link
                  String moreurl="webSearch.do?query=" +
                              URLEncoder.encode(queryString) +  //construct the "more" link
                              "&amp;maxresults=" + maxpage +
                              "&amp;startat=" + (startindex + maxpage);
%>
            <tr>
                  <td></td><td><a href="<%=moreurl%>">More Results>></a></td>
            </tr>
<%
            }
%>
            </table>
<%      }
        if (searcher != null)
            searcher.close();
%>
      </div>
   </body>
</html>
```

footerweb.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <link rel="stylesheet" href="images/style.css" type="text/css" media="all"/>
      <title>Loogle</title>
   </head>
   <body>
      <div id="footer"> &copy; 2011 <strong>SRMCEM Lucknow</strong> |
<strong>Developed by Arpan Gupta and Ankit Shukla </strong> | ClearBlue  by: <a
href="http://www.themebin.com/">ThemeBin</a>
```

```
                 <a href="webSearch.do">Home</a>
                
        </div>
    </body>
</html>
```

5.3.3 Registration and Mailing System:

We provide a registration system for users so that every user can have his/her own user profile. Below mentioned is the facility we provide to the registered users:

- ***Facility to subscribe to feeds from a server***

  Every user can subscribe to a subset of servers for updates. If there is an addition of file or folder on the server, a notification mail about the addition of file/folder is sent to all users who had subscribed for update feeds from that particular server. This mail will be sent after the server is scanned in the next crawl after the subscription.

5.3.3.1 Interface:

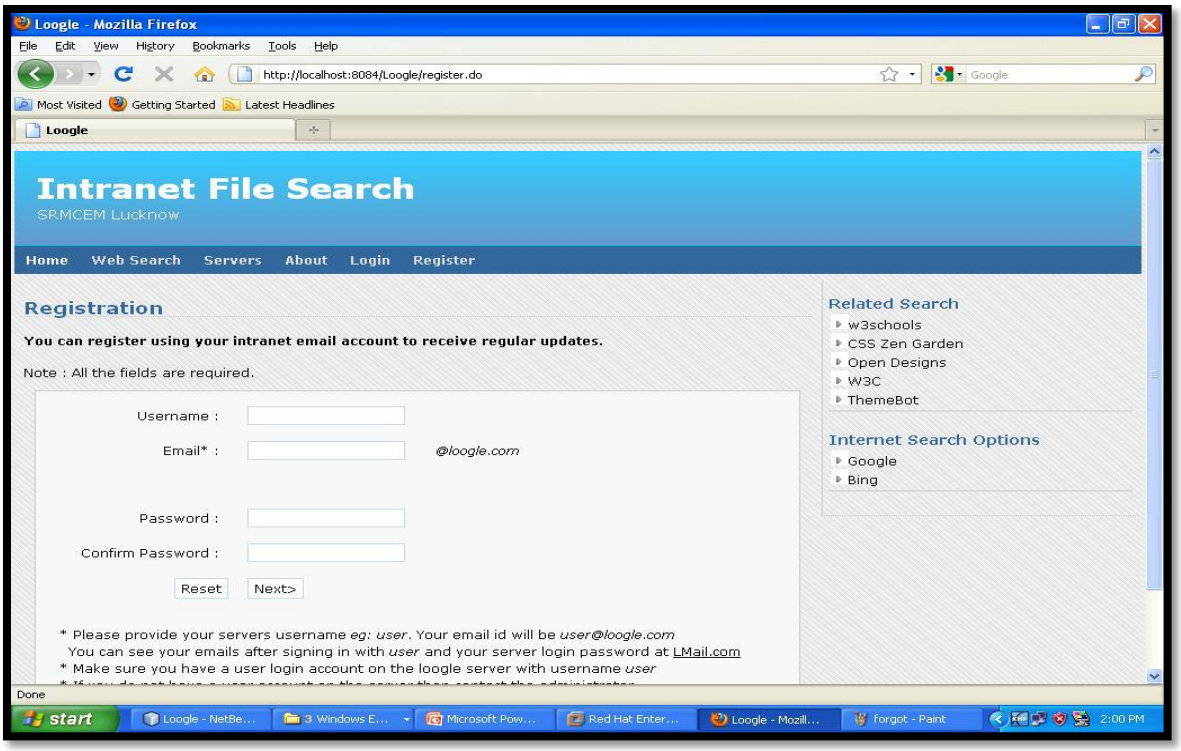


Figure 5.30: User interface screenshot- registration form

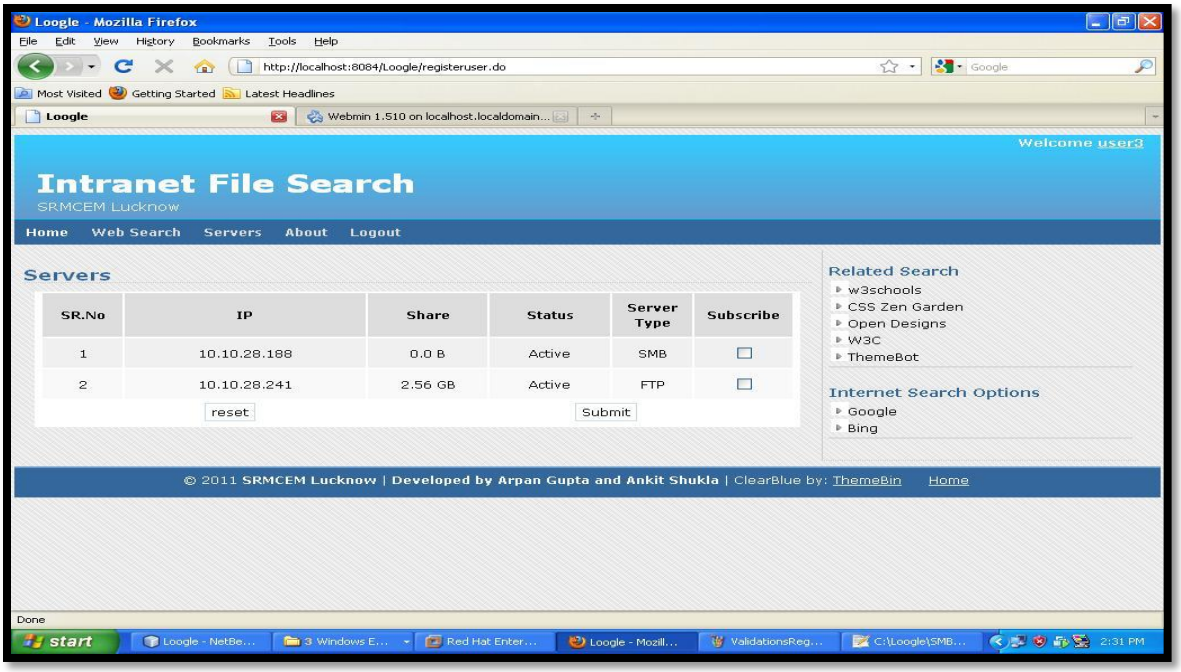


Figure 5.31: User interface screenshot- Subscription form

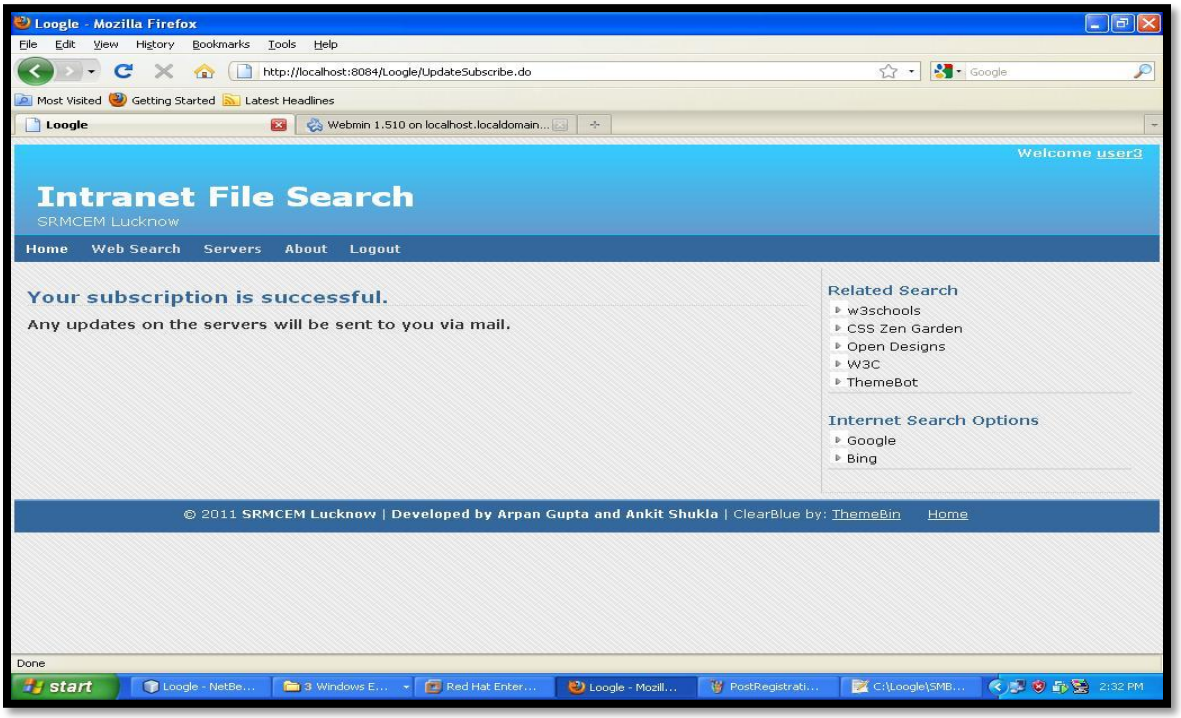


Figure 5.32: User interface screenshot- subscription success

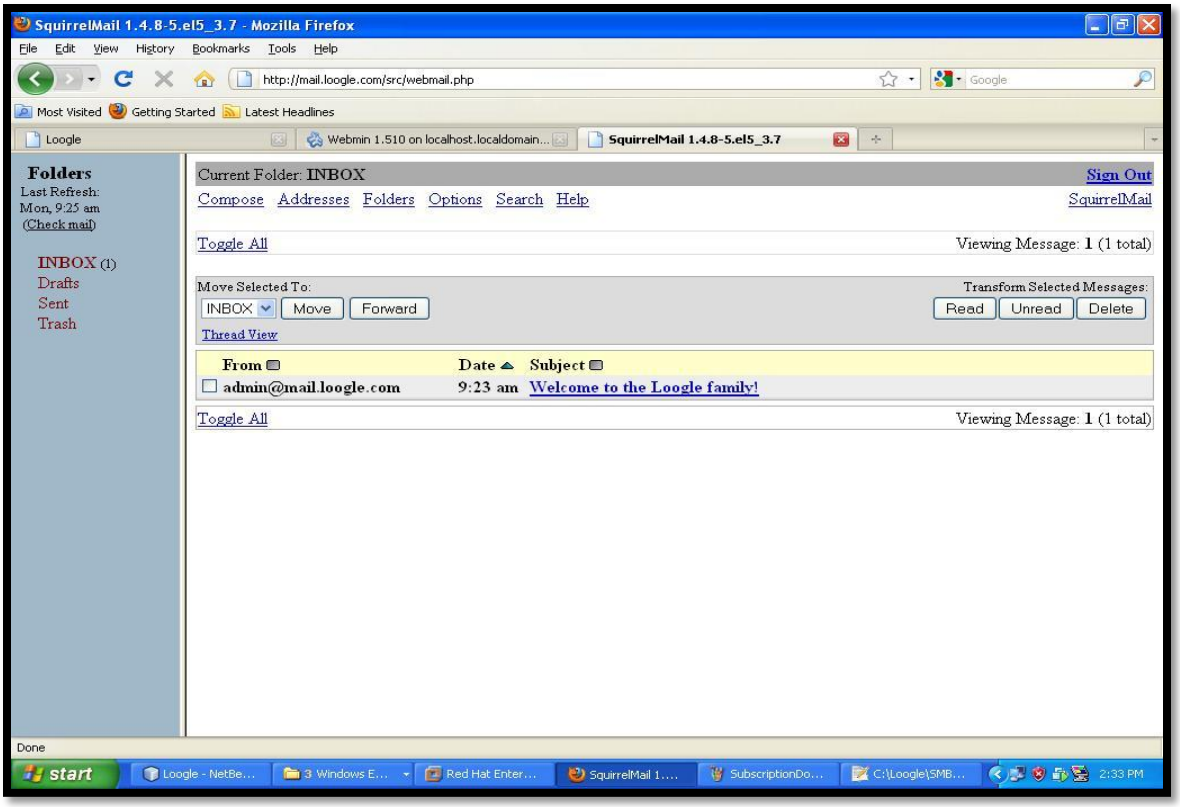


Figure 5.33: User interface screenshot- mail inbox

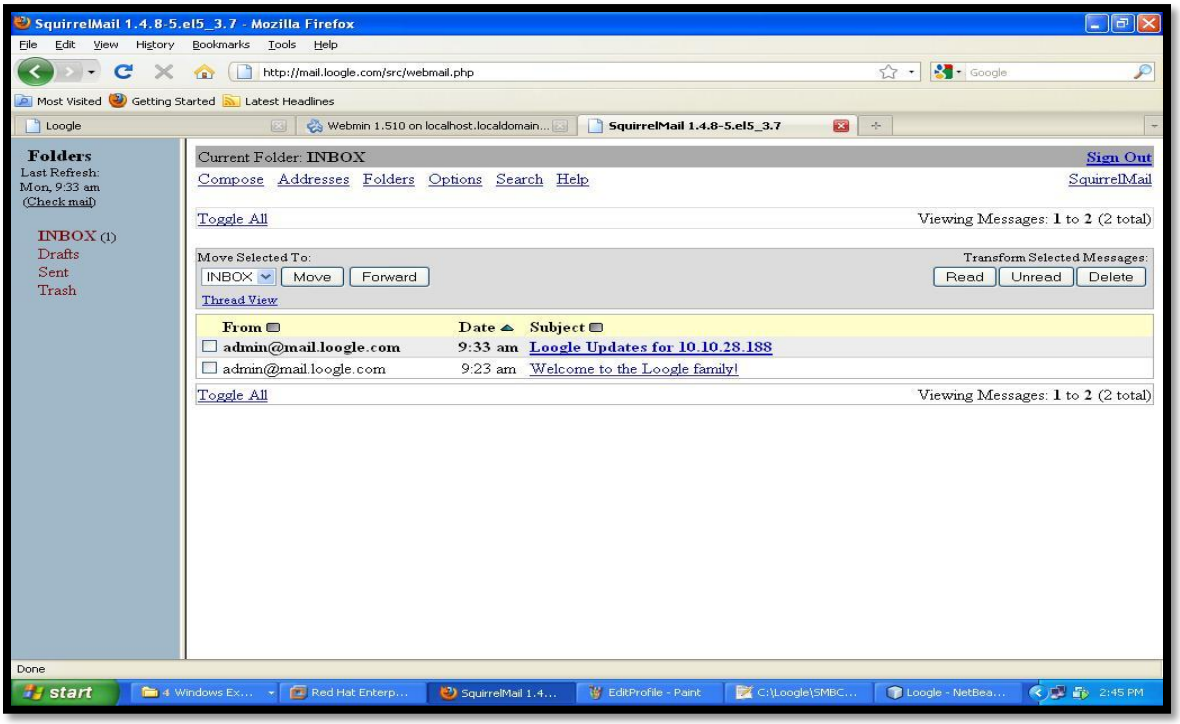


Figure 5.34: User interface screenshot- Mail inbox (updated)

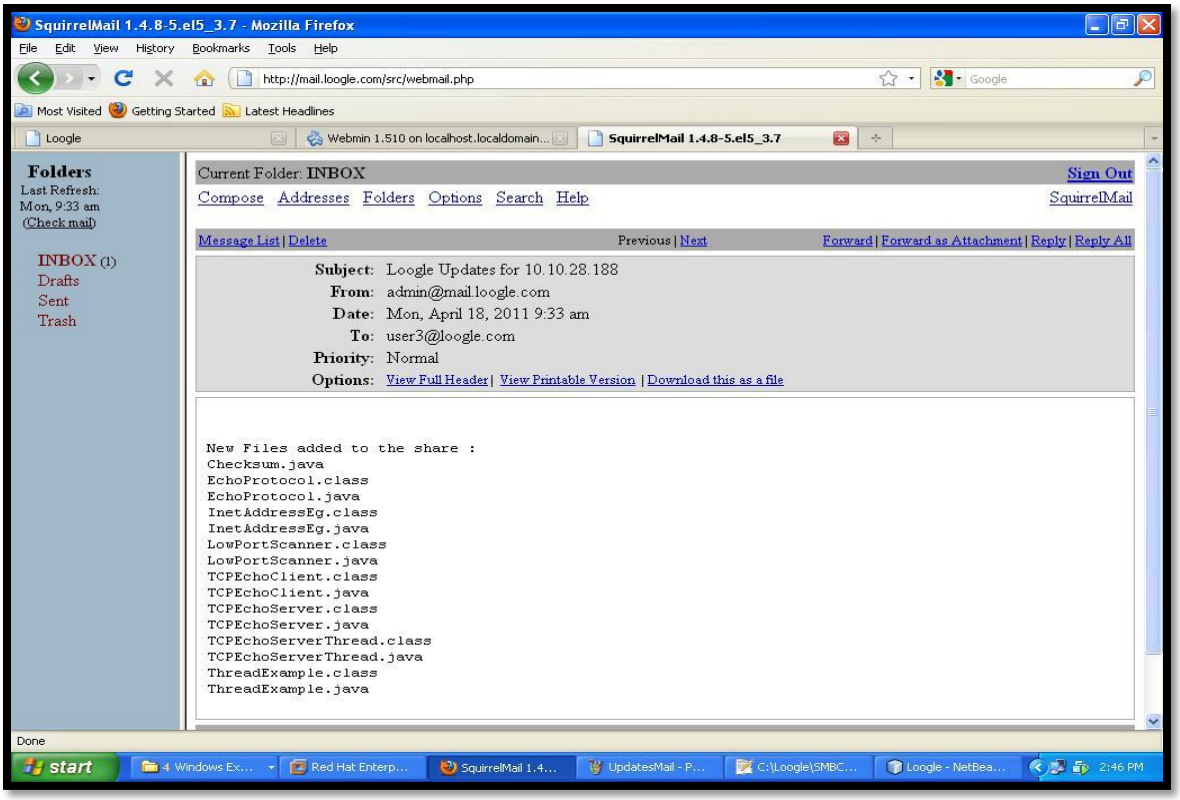


Figure 5.35: User interface screenshot- incoming mail updates

5.3.3.2 Coding:

registrationform.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8" language="java"
import="java.util.*"%>
<%@ page import="java.io.*" %>
<%@ page import="java.net.*" %>
<%@ page import="java.text.SimpleDateFormat" %>
<%@taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<%
        response.setHeader("Cache-Control", "no-cache"); //Forces caches to obtain a new copy
of the page from the origin server
        response.setHeader("Cache-Control", "no-store");//Directs caches not to store the page
under any circumstance
        response.setHeader("Pragma", "no-cache");//HTTP 1.0 backward compatibility
        response.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale
%>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
```

```
<head>
   <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
   <link rel="stylesheet" href="images/style.css" type="text/css" media="all"  />
   <title>Loogle</title>
</head>
<body>
    <div id="content">
    <h1>Registration</h1>
    <h4>You can register using your intranet email account to receive regular updates.</h4>
    Note : All the fields are required.
    <html:form action="registeruser.do" method="post" ><!--onsubmit="return
validate(this)">-->
      <table width="100%" border="0" cellspacing="15px">
        <tr>
          <td align="right" width="25%">
            Username :  
          </td>
          <td width="25%">
             <html:text property="username" maxlength="45"/>
          </td>
         <td width="50%">
            <font color="RED">
               <html:errors property="username"/>
            </font>
         </td>
        </tr>
        <tr>
          <td align="right" width="25%">
            Email* :  
          </td>
          <td width="25%">
             <html:text property="email" maxlength="45"/>
          </td>
          <td align="left">
             <i>@loogle.com</i>
          </td>
        </tr>
        <tr>
          <td width="25%" colspan="2"> </td>
          <td align="left" width="50%">
             <html:errors property="email" />
          </td>
        </tr>
        <tr>
```

```
                <td align="right" width="25%">
                  Password :  
                </td>
                <td width="25%">
                  <html:password property="password" maxlength="50"/>
                </td>
                <td>
                    <html:errors property="password" />
                </td>
              </tr>
              <tr>
                <td align="right" width="25%">
                  Confirm Password :  
                </td>
                <td width="25%">
                  <html:password property="cpassword" maxlength="50"/>
                </td>
                <td>
                  <html:errors property="cpassword" />
                </td>
              </tr>
              <tr>
                <td align="right" width="25%">
                  <html:reset value="Reset" />
                </td>
                <td align="left" width="25%">
                  <html:submit value="Next>" />
                </td>
                <td>
                </td>
              </tr>
              <tr>
                <td align="left" colspan="3">
                  <p>* Please provide your servers username <i>eg: user</i>. Your email id will be
<i>user@loogle.com</i><br/>
                      You can see your emails after signing in with <i>user</i> and
your server login password at
                    <a href="http://mail.loogle.com">LMail.com</a><br/>
                   * Make sure you have a user login account on the loogle server with username
<i>user</i><br/>
                   * If you do not have a user account on the server then contact the administrator.
                  </p>
                </td>
              </tr>
```

```
            </table>
        </html:form>
         </div>
        <script type="text/javascript">
        function validate(thisForm)
        {
           with(thisForm)
           {
           var mail=thisForm.email.value;
           if(mail==null||mail=="")
           {
              alert("Hi Value="+mail);
              return true;
           }
           else
              return checkEmail(mail);
           }
        }
        function checkEmail(email)
        {
           with(email)
           {
              alert("Value email  :: \""+email.value+"\"");
           if (/^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/.test(email.value))
           {
              return (true);
           }
           else
           {
              alert("Invalid E-mail Address! Please re-enter.");
              return (false);
           }
           }
        }
    </script>
    </body>
</html>
```

update.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
   "http://www.w3.org/TR/html4/loose.dtd">
<html>
    <head>
```

```
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Loogle</title>
  </head>
  <body>
    <div id="content">
      <table width="100%" border="0">
        <tr>
          <td>
            <h1>Your subscription is successful.</h1>
            <h2>Any updates on the servers will be sent to you via mail.</h2>
          </td>
        </tr>
      </table>
    </div>
  </body>
</html>
```

Mail System:

mailfail.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8" isErrorPage="true"%>
<%@taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Loogle</title>
  </head>
  <body>
    <div id="content">
      <table width="100%" border="0">
        <tr>
          <td>
            <h1>Unable to connect to the mail server. </h1>
            <h4>If you do not have a user account on the server, make sure that you create one
to receive the updates.</h4>
            <h2>To Subscribe --> <a href="servers.do"><i>Click Here</i></a></h2>
          </td>
        </tr>
      </table>
    </div>
  </body></html>
```

5.3.4 Administrator:

A secure login url is provided to the administrator using which he can reach the login page. After signing in he can start the automatic crawling process which uses the scheduler for periodic scanning of the intranet. He can also use the manual crawling option provided for the scanning of the FTP, SMB and Web servers individually.

Configuration files:

FTPConfigIP.txt: In this file the administrator has to define the IP addresses of the FTP servers. The IP of each server has to be given on a separate line without any special characters. Eg:

10.10.17.230

10.10.17.241

The system with the given IP will be scanned of any FTP connections and if FTP is enabled then it logs in using the *anonymous* user and *anon* password. The files available in the anonymous account will be recursively crawled and their information will be stored in the database. We are using the anonymous user as it has only read permission for the normal users.

SMBConfigIP.txt: In this file the administrator has to define the IP address or the range of IP addresses which will be scanned for the SMB shares, in the following ways:

Defining the specific IP on which the sharing is enabled. Eg: 10.10.17.43

Defining the network range by using an *(Here 254 IPs are checked for shares and scanned). Eg: 10.10.17.*

WebConfigLinks.txt: In this file the administrator has to define the URLs of the websites which are hosted in the intranet. Eg:

http://technotatva.com

http://srmcem.org

Blacklist.txt: In this file the administrator has to define the keywords which are not supposed to be crawled. Each word is specified on a separate line.

5.3.4.1 Interfaces:

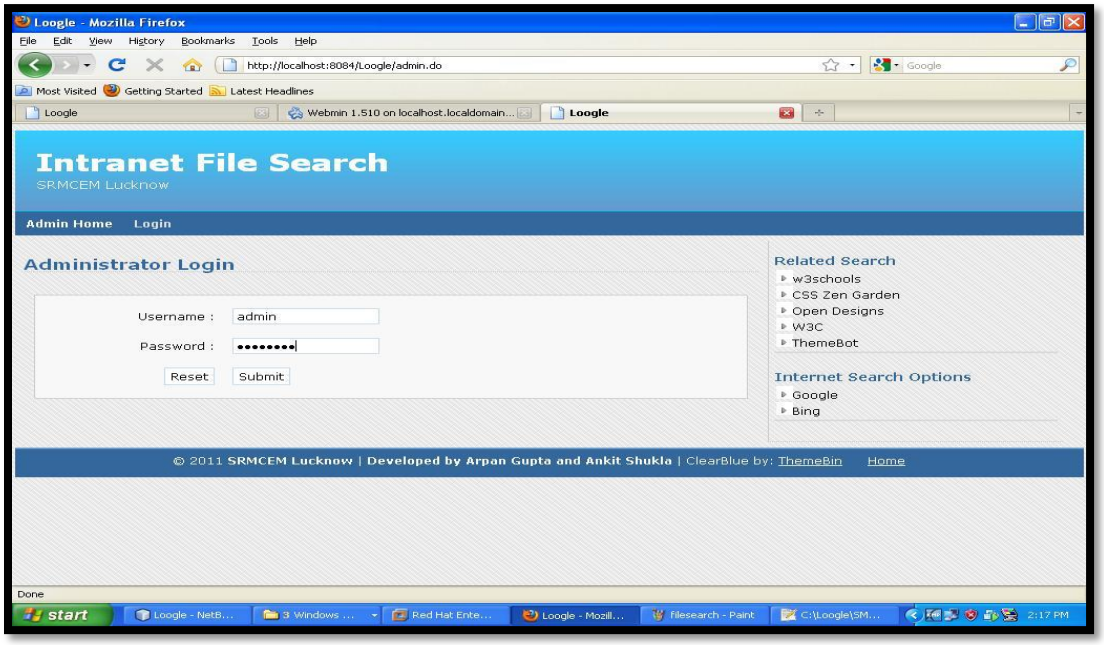


Figure 5.36: User interface screenshot- administrator login

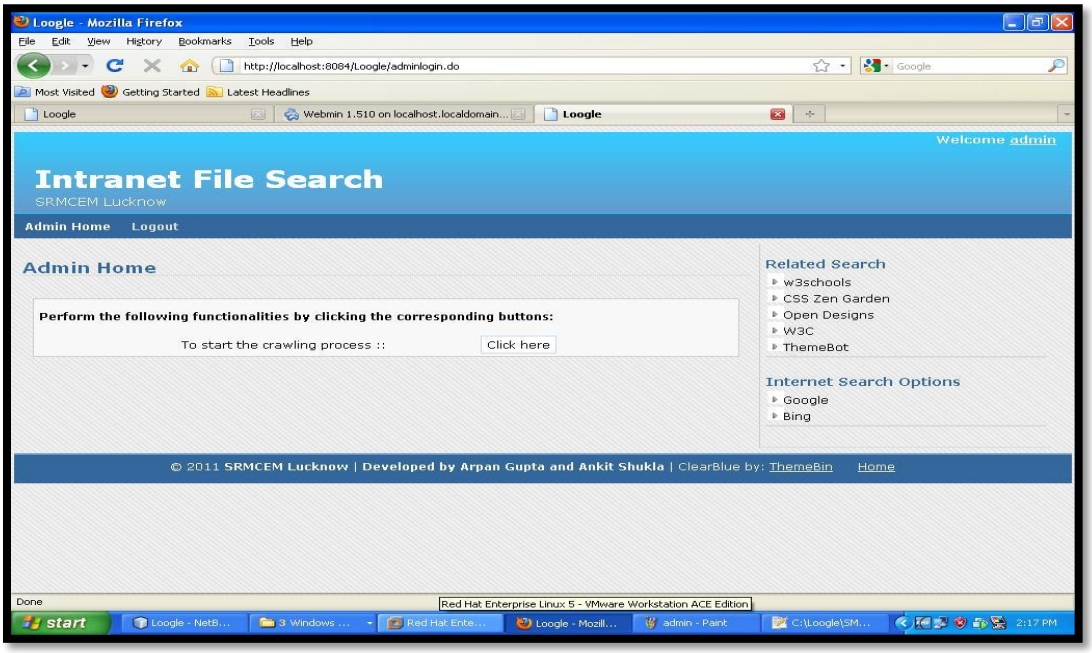


Figure 5.37: User interface screenshot- administrator home

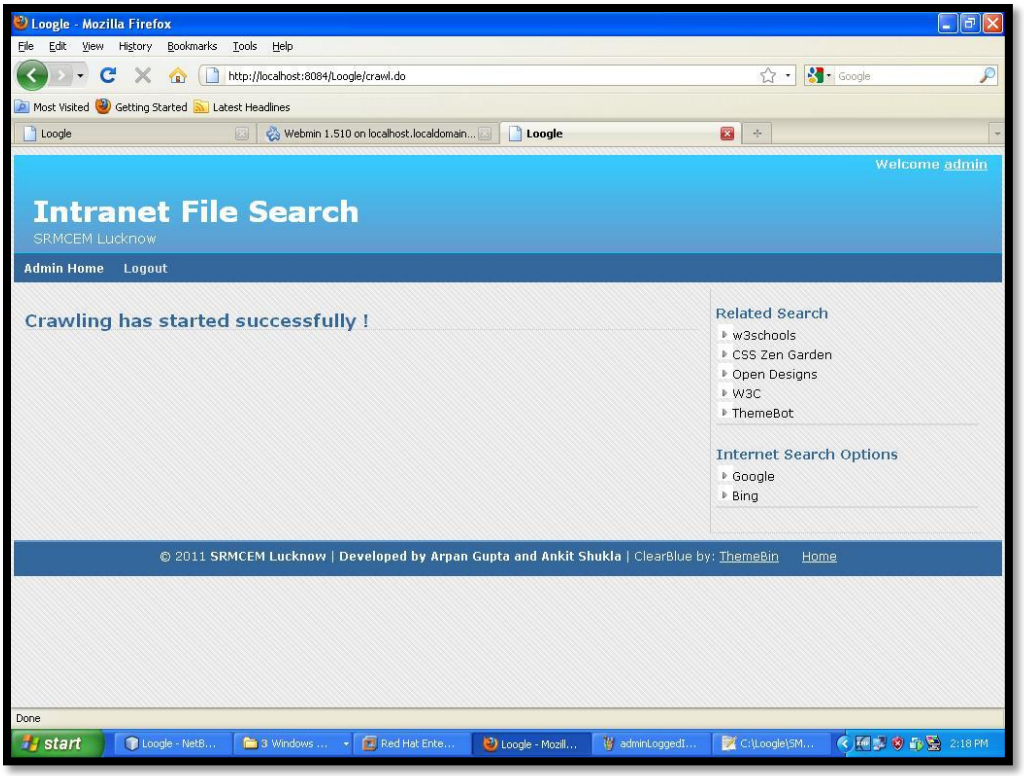


Figure 5.38: User interface screenshot- crawling start

5.3.4.2 Coding:

headeradmin.jsp:

```
  "http://www.w3.org/TR/html4/loose.dtd">
<%@taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<%@ page pageEncoding="UTF-8" %>
<%@ page import="java.text.SimpleDateFormat" %>
<%@taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<%!HttpSession session;
   String adminID;
%>
<%
        session = request.getSession(true);
        if (session != null) {
           adminID = (String) session.getAttribute("adminID");
        }
%>
<html>
   <head>
```

```
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <link rel="stylesheet" href="images/style.css" type="text/css" media="all"  />
    <title>Loogle</title>
</head>
<body>
    <script type="text/javascript" language="javascript">
        var browserType;
        var  flag= 0;
        if (document.layers) {browserType = "nn4"}
        if (document.all) {browserType = "ie"}
        if (window.navigator.userAgent.toLowerCase().match("gecko")) {
            browserType= "gecko"
        }
        function hide2(divName) {
            if (browserType == "gecko" )
                document.poppedLayer = eval('document.getElementById(divName)');
            else if (browserType == "ie")
                document.poppedLayer = eval('document.getElementById(divName)');
            else
                document.poppedLayer = eval('document.layers[divName]');
            document.poppedLayer.style.display = "none";
        }
        function show2(divName) {
            if (browserType == "gecko" )
                document.poppedLayer = eval('document.getElementById(divName)');
            else if (browserType == "ie")
                document.poppedLayer = eval('document.getElementById(divName)');
            else
                document.poppedLayer = eval('document.layers[divName]');
            document.poppedLayer.style.display = "inline";
        }
        function toggle(divName){
            if(flag == 0) show2(divName);
            else hide2(divName);
            flag = 1-flag;
        }
        function sizeQueryTypeChange(selectBox)
        {
            if(selectBox.options[selectBox.selectedIndex].value == "any")
            {
                document.searchForm.sizeValue.disabled = true;
                document.searchForm.sizeUnit.disabled = true;
            }
            else
```

```
            {
                document.searchForm.sizeValue.disabled = false;
                document.searchForm.sizeUnit.disabled = false;
            }
        }
        function serversQueryChange(selectBox)
        {
            if(selectBox.options[selectBox.selectedIndex].value == "ip")
                document.searchForm.serversValue.disabled = false;
            else
                document.searchForm.serversValue.disabled = true;
        }
        function fileTypeChange(selectBox)
        {
            if(selectBox.options[selectBox.selectedIndex].value == "extension")
                document.searchForm.fileExtension.disabled = false;
            else
                document.searchForm.fileExtension.disabled = true;
        }
    </script>
    <div id="header">
        <%
            if(adminID!=null)
            {
        %>
        <div align="right">
            <font size="2"><b>Welcome <a href="editProfile.do" style="color:white" title="Edit
your profile"><%=adminID%></a></b>   </font>
        </div>
        <%
            }
        %>
        <h1 id="logo"><a href="index.jsp" style="color:white;text-decoration:none;">Intranet
File Search</a></h1>
        <div id="slogan"><a href="index.jsp" style="color:white;text-
decoration:none;">SRMCEM Lucknow</a></div>
    </div>
    <div id="nav">
        <div id="nbar">
            <ul>
                <li id="selected"><a href="adminhome.do">Admin Home</a></li>
                <%
        if (adminID == null) {
                %>
```

```
                <li><a href="admin.do">Login</a></li>
                <%  } else {
                %>
                <li><a href="signout.do">Logout</a></li>
                <%          }
                %>
              </ul>
            </div>
          </div>
        </body>
</html>
```

adminhome.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<%@taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<%@ page import="java.text.SimpleDateFormat" %>
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Loogle</title>
   </head>
   <body>
      <div id="content">
      <h1>
         Administrator Login
      </h1>
      <br>
      <html:form action="adminlogin.do" method="post">
            <table width="100%" cellspacing="15px">
               <tr>
                  <td align="right" width="25%">
                     Username :
                  </td>
                  <td axis="left" width="25%">
                     <html:text property="username"/>
                  </td>
                  <td align="left" width="50%">
                     <html:errors property="username"/>
                  </td>
               </tr>
               <tr>
```

```
                <td align="right" width="25%">
                   Password :
                </td>
                <td align="left" width="25%">
                   <html:password property="password"/>
                </td>
                <td align="left" width="50%">
                   <html:errors property="password"/>
                </td>
      <%
         java.util.Date today = new java.util.Date();
         SimpleDateFormat s = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
         String timestamp = s.format(today);
      %>
      <html:hidden property="timestamp" value='<%=timestamp%>' />
             </tr>
             <tr>
                <td align="right">
                   <html:reset value="Reset"/>
                </td>
                <td align="left">
                   <html:submit value="Submit"/>
                </td>
             </tr>
          </table>
       </html:form>
    </div>
  </body>
</html>
```

crawlsuccess.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8" %>
<%@taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Loogle</title>
   </head>
   <body>
      <div id="content">
         <table width="100%" border="0">
            <tr>
```

```
                <td>
                    <h1>Crawling has started successfully !</h1>
                </td>
            </tr>
        </table>
    </div>
  </body>
</html>
```

crawlerror.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8" isErrorPage="true"%>
<%@taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Loogle</title>
  </head>
  <body>
    <div id="content">
      <table width="100%" border="0">
        <tr>
          <td>
            <h1>Exception occurred in starting the crawling process.</h1>
            <h2>Please ensure that you have provided the correct parameters.</h2>
          </td>
        </tr>
      </table>
    </div>
  </body>
</html>
```

adminEditProfile.jsp:

```
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <link rel="stylesheet" href="images/style.css" type="text/css" media="all"  />
    <title>Loogle</title>
  </head>
  <body>
    <div id="content">
      <h1>Edit Profile ::</h1>
      <br>
```

```
<html:form action="adminChangePwd.do" method="post" >
  <table width="100%" cellspacing="15px">
    <tr>
      <td>
      </td>
    </tr>
  <tr>
    <td align="right" width="25%">
      Current Password :
    </td>
    <td align="left" width="25%">
      <html:password property="currPassword" />
    </td>
    <td align="left" width="50%">
      <html:errors property="currPassword"/>
    </td>
  </tr>
  <tr>
    <td align="right" width="25%">
      New Password :
    </td>
    <td align="left" width="25%">
      <html:password property="newPassword" />
    </td>
    <td align="left" width="50%">
      <html:errors property="newPassword" />
    </td>
  </tr>
  <tr>
    <td align="right" width="25%">
      Retype Password :
    </td>
    <td align="left" width="25%">
      <html:password property="cNewPassword" />
    </td>
    <td align="left" width="50%">
      <html:errors property="cNewPassword"/>
    </td>
  </tr>
  <tr>
    <td align="right" width="25%">
      <html:reset value="Reset" />
    </td>
    <td align="left" width="25%">
```

```
                <html:submit value="Submit" />
              </td>
              <td width="50%"></td>
            </tr>
          </table>
    </html:form>
      </div>
    </body>
</html>
```

Chapter 6

TESTING

6.1 TESTING PERFORMED:

Software testing is a critical element of software quality assurance and represent the ultimate review of specification, design, coding. The purpose of product testing is to verify and validate the various work products viz. units, integrated unit, final product to ensure that they meet their requirements.

Testing Objectives:--

Basically, testing is done for the following purposes:

- Testing is a process of executing a program with intend of finding an error.
- A good test case is one that has a high probability of finding a yet undiscovered error.
- A successful test case is one that uncovers a yet undiscovered error.

Our objective is to design test cases that systematically uncover different classes of errors and do so with a minimum amount of time and effort. This process has two parts:--

- **Planning:** This involves writing and reviewing unit, integration, functional, validation and acceptance test plans.
- **Execution:** This involves executing these test plans, measuring, collecting data and verifying if it meets the quality criteria. Data collected is used to make appropriate changes in the plans related to development and testing.

System Testing:

Once the entire system has been built then it has to be tested against the "System Specification" to check if it delivers the features required. It is still developer focused, although specialist developers known as system testers are normally employed to do it.
In essence System testing is not about checking the individual parts of the design, but about checking the system as a whole. In effect it is one giant component. System testing can involve a number of specialist types of test to see if all the functional and non-functional requirements have been met.

Objectives of System Test: At a high level, this System Test intends to prove that :-

- The functionality, delivered by the development team, is as specified by the business in the Business Design Specification Document and the Requirements Documentation.

- The software is of high quality; the software will replace/support the intended business functions and achieves the standards required by the company for the development of new systems.

Formal Reviewing:

There will be several formal review points before and during system test. This is a vital element in achieving a quality product.

Formal Review Points:

1. Design Documentation
2. Testing Approach
3. Unit Test Plans
4. Unit Test Conditions & Results
5. System Test Conditions
6. System Test Progress
7. Post System Test Review

Testing Scope:

Outlined below are the main test types that will be performed for this release. All system test plans and conditions will be developed from the functional specification and the requirements catalogue.

Functional Testing:

The objective of this test is to ensure that each element of the application meets the functional requirements of the business as outlined in the :

- Requirements Catalogue
- Other functional documents produced during the course of the project i.e. resolution to issues/change requests/feedback.

This stage will also include **Validation Testing** - which is intensive testing of the new Front end fields and screens. Windows GUI Standards; valid, invalid and limit data input; screen & field look and appearance, and overall consistency with the rest of the application.

The third stage includes **specific functional testing** - these are low-level tests which aim to test the individual processes and data flows.

Integration Testing:

This test proves that all areas of the system interface with each other correctly and that there are no gaps in the data flow. Final Integration Test proves that system works as integrated unit when all the fixes are complete.

Business (User) Acceptance Test:

This test, which is planned and executed by the Business Representative(s), ensures that the system operates in the manner expected, and any supporting material such as procedures, forms etc. are accurate and suitable for the purpose intended. It is high level testing, ensuring that there are no gaps in functionality.

Performance Testing

These tests ensure that the system provides acceptable response times.

Penetration testing

It is to ensure that an intruder cannot break into the system using SQL injection or other such techniques. The use of prepared statement for database queries prevents attacks from SQL injection.

System Test Entrance/Exit Criteria:

Entrance Criteria

The Entrance Criteria specified by the system test controller, should be fulfilled before System Test can commence.

- All developed code must be unit tested. Unit and Link Testing must be completed and signed off by development team.
- All human resources must be assigned and in place.
- All test hardware and environments must be in place, and free for System test use.

Exit Criteria

The Exit Criteria detailed below must be achieved before the Phase 1 software can be recommended for promotion to Operations Acceptance status.

- All High Priority errors from System Test must be fixed and tested.

6.2 Validation Checks:

Various validations have been enforced to maintain the integrity of the system. Validations have been enforced in the registration forms so that the user does not enter any invalid information. He should not leave any compulsory field blank all other constraints on the information provided by the user are also maintained by these validation tests.

Validations have also been enforced to validate users before login. The

login validation is done using the username and password of the user. The details entered by the user are authenticated from the database then only the user is authorized to view the account. Here the username and password are assumed to be confidential to the user. If the username and password provided by user do not match, an error is generated.

Screen shots:

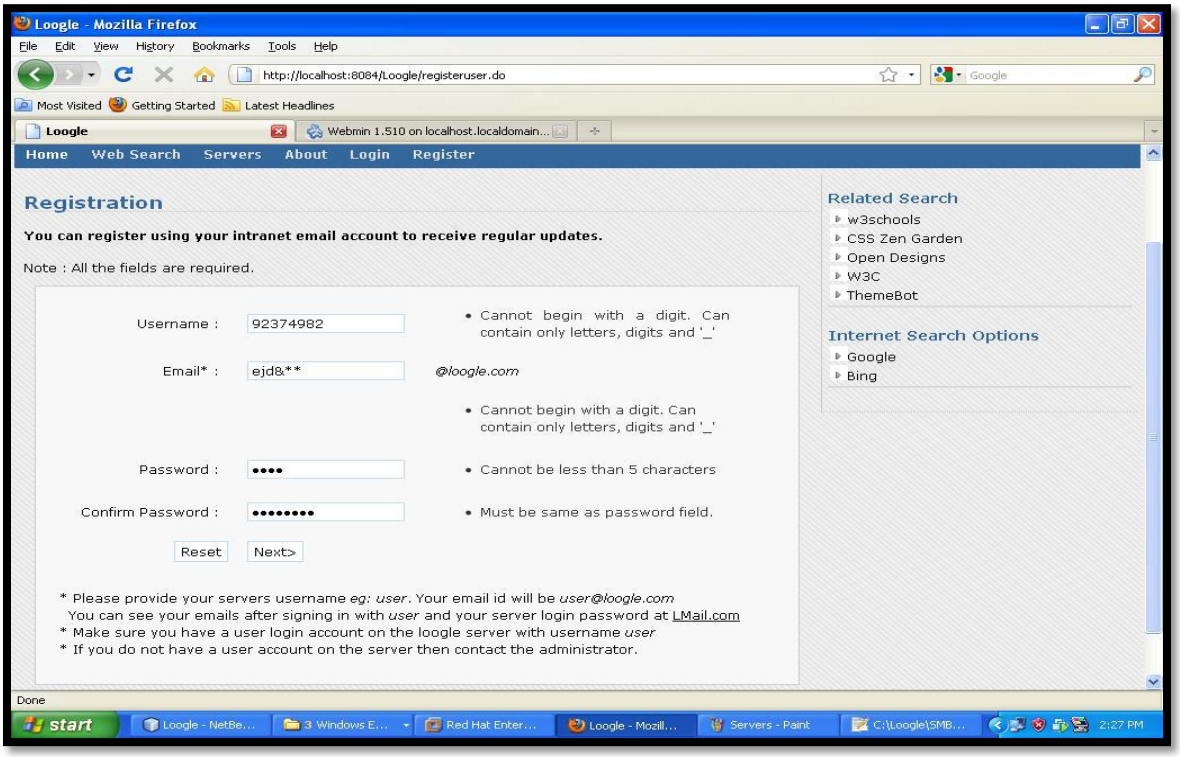


Figure 5.39: User interface screenshot- validation [registration]

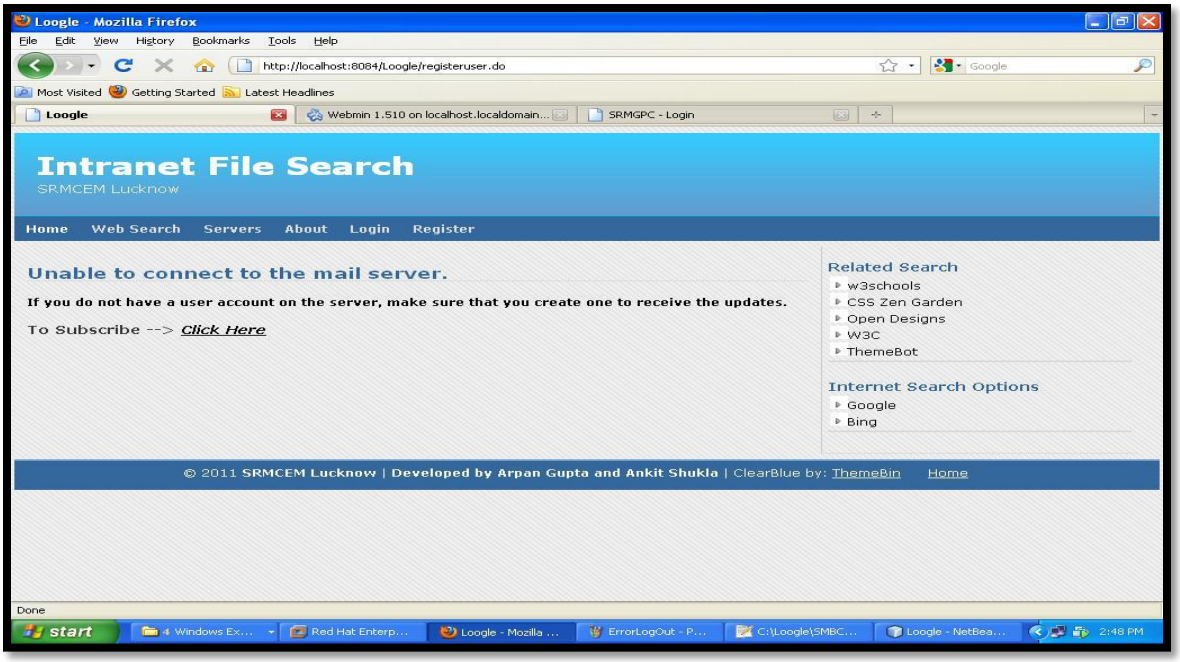


Figure 5.40: User interface screenshot- server error

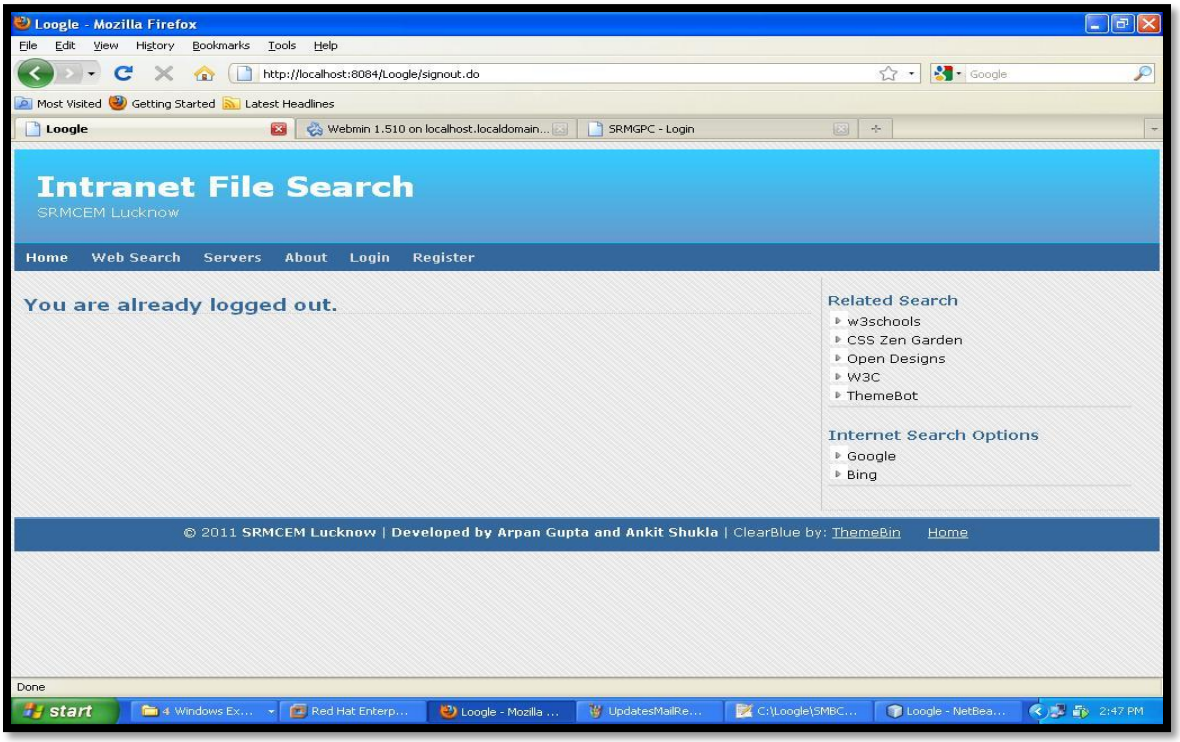


Figure 5.41: User interface screenshot- logout error

Chapter 7

RESULTS AND DISCUSSIONS

The project was successfully completed within the time constraint. All requirements were fulfilled. All the modules were completely tested and found to be fully functional and working without any errors. The products developed as a result were:

- An open source and easily deployable intranet search engine which facilitates file search and web page search over a local area network.
- An intranet mail system which can be used by users for receiving regular server updates and to communicate to the administrator.

Apart from the documents submitted regularly to depict the progress of the project the final deliverables include:

- A CD containing the following:
    - Running project
    - Complete source code
    - Backup of the database
    - Soft copy of the report
- Hard copy of the project report.

Chapter 8

ADVANTAGES AND APPLICATIONS

Advantages:

The developed search engine is advantageous in the following ways:

- It is a free, open source and easily deployable search engine.
- The search engine enables file search over LAN without having the need to know the individual IP addresses of the systems hosting the file.
- The search engine enables web page search of the websites hosted over the intranet.
- The search engine is platform independent and therefore can be used on any system irrespective of the platform running.
- Use of three tier MVC architecture makes the system secure.
- Since the system works over a wired intranet thus the speed and performance is optimized.

Applications:

- Can be used by any organization for efficient searching solution.
- User communication using the mail system.
- File search over intranet environment.
- Web page search hosted on local area network.

Chapter 9

CONCLUSION

Search engines have now become an integral part of an internet user's life. Any query related to anything can be easily answered through a search engine that returns the related page that contains the information required.

In *Loogle,* we have tried to create the same user experience for an intranet user as one experience while using search clients over internet. Keeping in mind the ease of use and basic requirements of the user, the search engine is devised in such a manner that the user experiences minimum difficulty and maximum satisfaction with search queries and results.

There are separate interfaces for the user to search for files or web pages, which enables disambiguation at user's end if he is unclear in his queries.

We have therefore tried our level best to create a system which would bring ease to users needing a system that would quench the necessity of a search engine for intranet environment

Chapter 10

FUTURE SCOPE OF THE PROJECT

The future scope includes the following functionalities which are provided by the system or can be easily added to it:

- Optimization in the crawling and scheduling process of the search engine.[3]
- Developing a ranking system to provide ranks to searched pages and files.
- Indexing the contents of the pdf and doc files.
- Generating the preview of the search results.
- Increasing the scalability.