

Data searching mechanism over encrypted cloud data using Matching Ranked Search

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Technology degree in Information Technology

by

Y.Santhosh (37120086)



**DEPARTMENT OF INFORMATION TECHNOLOGY
SCHOOL OF COMPUTING**

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

MARCH - 2021

DECLARATION

IY.santhosh(37120086), hereby declare that the Project Report entitled "**Data searching mechanism over encrypted cloud data using Matching Ranked search**" done by me under the guidance of **Mr.P.Saravanan M.E.**, is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology degree in Information Technology.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with "A" grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in



DEPARTMENT OF INFORMATION TECHNOLOGY

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Y.santhosh (37120086)**, who carried out the project entitled **"DATA SEARCHING MECHANISM OVER ENCRYPTED CLOUD DATA USING MATCHING RANKED SEARCH"** under my supervision from December 2020 to March 2021

Internal Guide

Mr. P. Saravanan M.E.,

Head of the Department

DR.R SUBHASHINI M.E,Ph.D;

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

ACKNOWLEDGEMENT

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

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D, Dean**, School of Computing , **Dr. R.Subhashini M.E., Ph.D. , Head** of the Department of Information Technology for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Mr.P.Sarvanan M.E.**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

ABSTRACT

Encrypted search technology has been studied extensively in recent years. With more and more information being stored in cloud, creating indexes with independent keywords has resulted in enormous storage cost and low search accuracy, which has become an urgent problem to be solved. Thus, in this paper, we propose a new feature matching ranked search mechanism (FMRS) for encrypted cloud data. This mechanism uses feature score algorithm (FSA) to create indexes, which allows multi-keywords which are extracted from a document as a feature to be mapped to one dimension of the index. Thus, the storage cost of indexes can be reduced and the efficiency of encryption can be improved. Moreover, FMRS uses a matching score algorithm (MSA) in generating trapdoor process. With the help of FSA, the matching score algorithm can rank the search results according to the type of match and the number of matching keywords, and therefore it is able to return results with higher ranking accuracy. Comprehensive analysis prove that our mechanism is more feasible and effective.

TABLE OF CONTENT

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	6
1	INTRODUCTION	7
2	Literature Survey	9
2.1	Privacypreservingkeyword searches on remote encrypted data.	10
2.2	Practical techniques for searches on encrypted data.	10
2.3	SSARES,Secure searchable automated remote email storage.	11
2.4	Secure anonymous database search	11
2.5	Privacy-enhanced searches using broom	12

3. Methodology

3.1 SYSTEM ANALYSIS	15
3.2 IMPLEMENTATION	16
3.3SYSTEM REQUIREMENTS	17
3.4Software Environment	18
3.5 SYSTEM DESIGN	23
3.6	29
DESIGN	

4.

RESULT&DISCUSSION	35
4.1 HOME SCREEN	37
4.2 OWNER LOGIN	38
4.3 LOGIN STATUS	39
4.4 UPLOAD FILE	40
4.5 ENCRYPTED DATA	40
4.6 UPLOAD STATUS	41
4.7 VIEW ALL UPLOADS	41
4.8 CLOUD LOGIN	42
4.9 VIEW OWNERS	43
5. CONCLUSION	46

1. INTRODUCTION

In order to save local storage cost, more and more people are willing to store their private data in cloud servers. However, there are potential risks in cloud storage environments, since data leakage events have happened more and more frequently in recent years. Many people have to encrypt their data before uploading to the cloud to ensure the security of private information. For the untrustworthy cloud environment, many scholars have proposed their solutions for different cloud storage problems [1]– [4]. Their works focus on designing a searchable encrypted index that hides document information from cloud server and can only be computed through specific trapdoors. The searchable encrypted index can be a bloom filter [5]– [7] generated by mapping keywords, or an index vector [8], [9] reflecting the importance of keywords. However, these schemes not only have large storage cost but also have low search accuracy.

In encrypted search schemes, it is particularly difficult to find a solution that can satisfy the user's accurate query. Although there are many searchable methods that support multi-keyword search [10], [11], they do not consider the relationship between extracted keywords. In addition, existing methods which focus on judging the importance of a keyword for a document are not sufficiently, and when a large number of keywords are extracted from the document, it will inevitably lead to a huge storage cost. In order to address these missing or incomplete search problems, many scholars have proposed fuzzy keyword search and ranking search [12]– [17]. These solutions can greatly enrich query results, but they often cannot meet users' actual search requirements.

As we all know, in plain document query, phrase search method is widely used and has achieved better effects. Recently, many scholars have begun to use the method of phrase or conjunctive keyword search in encrypted search system. Comparing with single keyword or multi-keyword search, the phrase search method can obtain higher query precision. However, when the number of phrases which is extracted from documents increases, the phrase search will result in a huge computational cost, and this cost tends to increase with the number of phrases [18]. Therefore, it is urgent to find a feasible and effective search solution that cannot only reduce computational burden but also return results accurately.

In this paper, we propose a new feature matching ranked search mechanism (FMRS) for encrypted cloud data. In the process of creating indexes, we first propose a feature score algorithm (FSA) that can map multiple keywords which are extracted from a document to one dimension of the index. Therefore, comparing with creating indexes with independent keywords, this mechanism can not only reduce index dimensions, but also improve the efficiency of encryption. During the generating trapdoor process of FMRS, we design a novel matching score algorithm (MSA). This algorithm cannot only focus on the matching relationship between query keywords and features, but also comprehensively consider each matching type to return results that are closer to user's real request. The contributions of this paper are summarized as follows.

- (1) For the first time, we propose a novel feature score algorithm to create indexes, which allows multiple keywords extracted from a document as a feature tube mapped to one dimension of the index to achieve the purpose of accelerating the encryption process and reducing storage cost.
- (2) In the generating trapdoor process of FMRS, a matching score algorithm is designed. With the help of FSA, this algorithm can rank search results according to the matching types and the number of matching keywords, and therefore the returned results have a higher-ranking accuracy.

The rest of this paper is arranged as follows. In section II, we introduce related work of encrypted search. Section III describes the system model, the threat model, and the design goals. Also, in this section, we describe some related terms used in this paper in detail. Then we present the proposed feature matching ranked search mechanism in section IV. In section V, we analyze the performance of our mechanism.

2. Literature Survey

Verifiable symmetric searchable encryption for semi-honest-but-curious cloud servers

Outsourcing data to cloud servers, while increasing service availability and reducing users' burden of managing data, inevitably brings in new concerns such as data privacy, since the server may be honest-but-curious. To mediate the conflicts of data usability and data privacy in such a scenario, research of searchable encryption is of increasing interest. Motivated by the fact that a cloud server, besides its curiosity, may be selfish in order to save its computation and/or download bandwidth, in this paper, we investigate the searchable encryption problem in the presence of a semi-honest-but-curious server, which may execute only a fraction of search operations honestly and return a fraction of search outcome honestly. To fight against this strongest adversary ever, a verifiable SSE (VSSE) scheme is proposed to offer verifiable searchability in addition to the data privacy, both of which are further confirmed by our rigorous security analysis. Besides, we treat the practicality/efficiency as a central requirement of a searchable encryption scheme as well. To this end, we implemented and tested the proposed VSSE, with real world data sets, on a laptop (serve as the server) and a mobile phone running Android 2.3.4 (serve as the user). The experimental results optimistically suggest that the proposed scheme satisfies all of our design goals.

Designated cloud server public key encryption with keyword search from lattice in the standard model.

A public key encryption scheme with keyword search capabilities is proposed using lattices for applications in multiuser environments. The proposed scheme enables a cloud server to check if any given encrypted data contains certain keywords specified by multiple users, but the server would not have knowledge of the keywords specified by the users or the contents of the encrypted data, which provides data privacy as well as privacy for user queries in multiuser environments. It can be proven secure under the standard learning with errors assumption in the random oracle model.

2.1 Privacy preserving keyword searches on remote encrypted data.

We consider the following problem: a user U wants to store his files in an encrypted form on a remote file server S . Later the user U wants to efficiently retrieve some of the encrypted files containing (or indexed by) specific keywords, keeping the keywords themselves secret and not jeopardizing the security of the remotely stored files. For example, a user may want to store old e-mail messages encrypted on a server managed by Yahoo or another large vendor, and later retrieve certain messages while travelling with a mobile device. In this paper, we offer solutions for this problem under well-defined security requirements. Our schemes are efficient in the sense that no public-key cryptosystem is involved. Indeed, our approach is independent of the encryption method chosen for the remote files. They are also incremental, in that U can submit new files which are totally secure against previous queries but still searchable against future queries.

2.2 Practical techniques for searches on encrypted data.

It is desirable to store data on data storage servers such as mail servers and file servers in encrypted form to reduce security and privacy risks. But this usually implies that one has to sacrifice functionality for security. For example, if a client wishes to retrieve only documents containing certain words, it was not previously known how to let the data storage server perform the search and answer the query, without loss of data confidentiality. We describe our cryptographic schemes for the problem of searching on encrypted data and provide proofs of security for the resulting crypto systems. Our techniques have a number of crucial advantages. They are provably secure: they provide provable secrecy for encryption, in the sense that the untrusted server cannot learn anything about the plaintext when only given the ciphertext; they provide query isolation for searches, meaning that the untrusted server cannot learn anything more about the plaintext than the search result; they provide controlled searching, so that the untrusted server cannot search for an arbitrary word without the user's authorization; they also support hidden queries, so that the user may ask the untrusted server to search for a secret word without revealing the word to the server. The algorithms presented are simple, fast (for a document of length n , the encryption and search algorithms only need $O(n)$ stream

cipher and block cipher operations), and introduce almost no space and communication overhead, and hence are practical to use today.

2.3 SSARES: Secure searchable automated remote email storage.

The increasing centralization of networked services places user data at considerable risk. For example, many users store email on remote servers rather than on their local disk. Doing so allows users to gain the benefit of regular backups and remote access, but it also places a great deal of unwarranted trust in the server. Since most email is stored in plaintext, a compromise of the server implies the loss of confidentiality and integrity of the email stored therein. Although users could employ an end-to-end encryption scheme (e.g., PGP), such measures are not widely adopted, require action on the sender side, only provide partial protection (the email headers remain in the clear), and prevent the users from performing some common operations, such as server-side search. To address this problem, we present secure searchable automated remote email storage (SSARES), a novel system that offers a practical approach to both securing remotely stored email and allowing privacy-preserving search of that email collection. Our solution encrypts email (the headers, body, and attachments) as it arrives on the server using public-key encryption. SSARES uses a combination of identity based encryption and bloom filters to create a searchable index. This index reveals little information about search keywords and queries, even against adversaries that compromise the server. SSARES remains largely transparent to both the sender and recipient.

2.4 Secure anonymous database search.

There exist many large collections of private data that must be protected on behalf of the entities that hold them or the clients they serve. However, there are also often many legitimate reasons for sharing that data in a controlled manner. How can two parties decide to share data without prior knowledge of what data they have? For example, two intelligence agencies might be willing to cooperate by sharing documents about a specific case, and need a way of determining which documents might be of interest to each other. We introduce and address the problem of allowing such entities to search each other's data securely and anonymously. We aim to protect the content of the queries, as well as the content of documents unrelated to those queries, while concealing the identity of the participants. Although there exist

systems for solving similar problems, to our knowledge we are the first to address this specific need and also the first to present a secure anonymous search system that is practical for real-time querying. In order to achieve this in an efficient manner, we make use of Bloom filters [5], definitions of security for deterministic encryption [22] that we adapt and instantiate in the private key setting and of a novel encryption primitive, reroutable encryption.

2.5 Privacy-enhanced searches using encrypted Bloom filters.

It is often necessary for two or more or more parties that do not fully trust each other to share data selectively. For example, one intelligence agency might be willing to turn over certain documents to another such agency, but only if the second agency requests the specific documents. The problem, of course, is finding out that such documents exist when access to the database is restricted. We propose a search scheme based on Bloom filters and group ciphers such as Pohlig-Hellman encryption. A semi-trusted third party can transform one party's search queries to a form suitable for querying the other party's database, in such a way that neither the third party nor the database owner can see the original query. Furthermore, the encryption keys used to construct the Bloom filters are not shared with this third party. Multiple providers and queriers are supported; provision can be made for third-party "warrant servers", as well as "censorship sets" that limit the data to be shared.

Enabling personalized search over encrypted outsourced data with efficiency improvement.

In cloud computing, searchable encryption scheme over outsourced data is a hot research field. However, most existing works on encrypted search over outsourced cloud data follow the model of "one size fits all" and ignore personalized search intention. Moreover, most of them support only exact keyword search, which greatly affects data usability and user experience. So how to design a searchable encryption scheme that supports personalized search and improves user search experience remains a very challenging task. In this paper, for the first time, we study and solve the problem of personalized multi-keyword ranked search over encrypted data

(PRSE) while preserving privacy in cloud computing. With the help of semantic ontology WordNet, we build a user interest model for individual user by analyzing the user's search history, and adopt a scoring mechanism to express user interest smartly. To address the limitations of the model of "one size fit all" and keyword exact search, we propose two PRSE schemes for different search intentions. Extensive experiments on real-world dataset validate our analysis and show that our proposed solution is very efficient and effective.

Multi-keyword ranked search supporting synonym query over encrypted data in cloud computing.

Cloud computing becomes increasingly popular. To protect data privacy, sensitive data should be encrypted by the data owner before outsourcing, which makes the traditional and efficient plaintext keyword search technique useless. The existing searchable encryption schemes support only exact or fuzzy keyword search, not support semantics-based multi-keyword ranked search. In the real search scenario, it is quite common that cloud customers' searching input might be the synonyms of the predefined keywords, not the exact or fuzzy matching keywords due to the possible synonym substitution (reproduction of information content) and/or her lack of exact knowledge about the data. Therefore, synonym-based multi-keyword ranked search over encrypted cloud data remains a very challenging problem. In this paper, for the first time, we propose an effective approach to solve the problem of synonym-based multi-keyword ranked search over encrypted cloud data. We make contributions mainly in two aspects: synonym-based search for supporting synonym query and multi-keyword ranked search for achieving more accurate search result. Two secure schemes are proposed to meet privacy requirements in two threat models of known ciphertext model and known background model. In enhanced scheme, the sensitive frequency information can be well protected by introducing some dummy keywords, which is not adopted in basic scheme. We give security analysis to justify the correctness and privacy-preserving guarantee of the proposed schemes. Extensive experiments on real-world dataset validate our analysis and show that our proposed solution is very efficient and effective in supporting synonym-based searching.

Secure conjunctive multi-keyword search for multiple data owners in cloud computing.

Recently, secure search over encrypted cloud data has become a hot research spot and challenging task. Some secure search schemes have been proposed to try to meet this challenge. In this paper, we propose a conjunctive multi-keyword secure search scheme for multiple data owners. To guarantee data security and system flexibility in the multiple data owners environment, we design an ingenious secure query scheme that allows each data owner to adopt randomly chosen temporary keys to build secure indexes for different data files. An authorized data user does not need to know these temporary keys of constructing indexes and can instead randomly choose another temporary query keys to encrypt query keywords while the cloud can correctly perform keywords matching over encrypted data files. Extensive experiments demonstrate the correctness and practicality of the proposed scheme.

3. Methodology

3.1 System Analysis

EXISTING SYSTEM

We believe that the cloud server is “honestbut curious” [39], which means that the cloud server willhonestly perform search operations according to the submit-ted trapdoor. However, at the same time, the cloud servermay also infer the plain data from existing information. Theexisting information can be encrypted documents, encryptedindexes and the trapdoor which must be submitted to thecloud server, or some background knowledge about encryp-tion. We believe that the general goal of the server’s attackis to infer secret keys or plain data. he cloud server may judgewhether some query keywords are in a trapdoor by combiningexisting trapdoors and query results, or it may use encryptedbackground information to infer secret keys.

PROPOSED SYSTEM

proposed several schemes over encrypted clouddata, which also supports dynamic update operations, suchas adding or deleting files.Although the above related work has improved the search-able encryption technology from different aspects, the fol-lowing problems still need to be solved: (1) The amountof encryption calculation increases with the size of dictio-nary, and when a large number of keywords are extracted,it will inevitably lead to huge computational cost; (2) Thesearch results of existing schemes depend on similari-tyscores or independent keyword matching relationships. Theydo not comprehensively consider the number of matchingkeywords and the overall matching relationship, which willresult in low *query accuracy*.

3.2 Implementation

MODULES

1. DataUser
2. DataOwner
3. Cloud

Module Description

Data Owner

In this application owner should register with the application and after his/her registration owner should be authorized by the cloud, then only the owner can able to login after the owner successful login owner can perform some operations such as upload document and view documents.

Data User

In this application user should register with the application and after his/her registration user should be authorized by the cloud, then only the user can able to login after the user successful login user can perform some operations such as generate trapdoor request, cloud response and all downloads.

Cloud

In this application cloud can directly login with the application after cloud successful login cloud can do some operations such as view owners and authorize, view users and authorize, view all documents, trapdoor request, matching Ranked.

3.3 SYSTEM REQUIREMENTS:

HARDWARE REQUIREMENTS:

- System : Pentium Dual Core.
- Hard Disk : 120 GB.
- Monitor : 15" LED
- Input Devices : Keyboard, Mouse
- Ram : 1 GB

SOFTWARE REQUIREMENTS:

- Operating system : Windows 7.
- Coding Language : JAVA/J2EE

- Tool : Eclipse
- Database : MYSQL

3.4 Software Environment

Java Technology

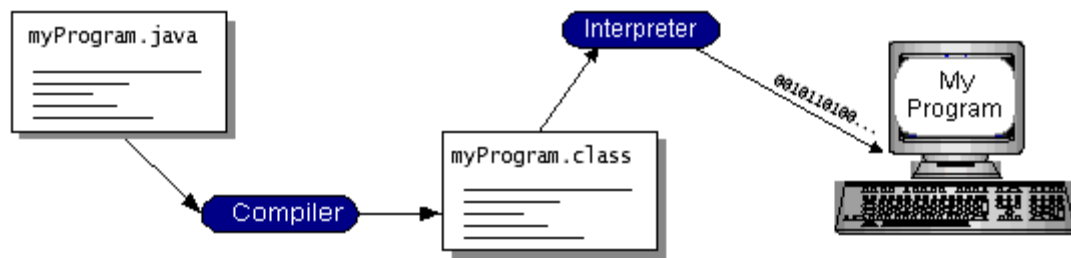
Java technology is both a programming language and a platform.

The Java Programming Language

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

- Simple
- Architecture neutral
- Object oriented
- Portable
- Distributed
- High performance
- Interpreted
- Multithreaded
- Robust
- Dynamic
- Secure

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called *Java byte codes* —the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.



You can think of Java byte codes as the machine code instructions for the *Java Virtual Machine* (Java VM). Every Java interpreter, whether it's a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java byte codes help make "write once, run anywhere" possible. You can compile your program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the Java VM. That means that as long as a computer has a Java VM, the same program written in the Java programming language can run on Windows 2000, a Solaris workstation, or on an iMac.



The Java Platform

A *platform* is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and MacOS. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.

The Java platform has two components:

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

You've already been introduced to the Java VM. It's the base for the Java platform and is ported onto various hardware-based platforms.

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries of related classes and interfaces; these libraries are known as *packages*. The next section, What Can Java Technology Do? Highlights what functionality some of the packages in the Java API provide.

The following figure depicts a program that's running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware.



Native code is code that after you compile it, the compiled code runs on a specific hardware platform. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time byte code compilers can bring performance close to that of native code without threatening portability.

What Can Java Technology Do?

The most common types of programs written in the Java programming language are *applets* and *applications*. If you've surfed the Web, you're probably already familiar with applets. An applet is a program that adheres to certain conventions that allow it to run within a Java-enabled browser.

However, the Java programming language is not just for writing cute, entertaining applets for the Web. The general-purpose, high-level Java programming language is also a powerful software platform. Using the generous API, you can write many types of programs.

An application is a standalone program that runs directly on the Java platform. A special kind of application known as a *server* serves and supports clients on a network. Examples of servers are Web servers, proxy servers, mail servers, and print servers. Another specialized program is a *servlet*. A servlet can almost be thought of as an applet that runs on the server side. Java Servlets are a popular choice for building interactive web applications, replacing the use of CGI scripts. Servlets are similar to applets in that they are runtime extensions of applications. Instead of working in browsers, though, servlets run within Java Web servers, configuring or tailoring the server.

How does the API support all these kinds of programs? It does so with packages of software components that provides a wide range of functionality. Every full implementation of the Java platform gives you the following features:

- **The essentials:** Objects, strings, threads, numbers, input and output, data structures, system properties, date and time, and so on.
- **Applets:** The set of conventions used by applets.
- **Networking:** URLs, TCP (Transmission Control Protocol), UDP (User Datagram Protocol) sockets, and IP (Internet Protocol) addresses.
- **Internationalization:** Help for writing programs that can be localized for users worldwide. Programs can automatically adapt to specific locales and be displayed in the appropriate language.
- **Security:** Both low level and high level, including electronic signatures, public and private key management, access control, and certificates.
- **Software components:** Known as JavaBeans™, can plug into existing component architectures.
- **Object serialization:** Allows lightweight persistence and communication via Remote Method Invocation (RMI).
- **Java Database Connectivity (JDBC™):** Provides uniform access to a wide range of relational databases.

The Java platform also has APIs for 2D and 3D graphics, accessibility, servers, collaboration, telephony, speech, animation, and more. The following figure depicts what is included in the Java 2 SDK.



JDBC Goals

Few software packages are designed without goals in mind. JDBC is one that, because of its many goals, drove the development of the API. These goals, in conjunction with early reviewer feedback, have finalized the JDBC class library into a solid framework for building database applications in Java.

The goals that were set for JDBC are important. They will give you some insight as to why certain classes and functionalities behave the way they do. The eight design goals for JDBC are as follows:

1. SQL Level API

The designers felt that their main goal was to define a SQL interface for Java. Although not the lowest database interface level possible, it is at a low enough level for higher-level tools and APIs to be created. Conversely, it is at a high enough level for application programmers to use it confidently. Attaining this goal allows for future tool vendors to “generate” JDBC code and to hide many of JDBC’s complexities from the end user.

2. SQL Conformance

SQL syntax varies as you move from database vendor to database vendor. In an effort to support a wide variety of vendors, JDBC will allow any query statement to be passed through it to the underlying database driver. This allows the connectivity module to handle non-standard functionality in a manner that is suitable for its users.

3. **JDBC must be implemental on top of common database interfaces**

The JDBC SQL API must “sit” on top of other common SQL level APIs. This goal allows JDBC to use existing ODBC level drivers by the use of a software interface. This interface would translate JDBC calls to ODBC and vice versa.

4. **Provide a Java interface that is consistent with the rest of the Java system**

Because of Java’s acceptance in the user community thus far, the designers feel that they should not stray from the current design of the core Java system.

5. **Keep it simple**

This goal probably appears in all software design goal listings. JDBC is no exception. Sun felt that the design of JDBC should be very simple, allowing for only one method of completing a task per mechanism. Allowing duplicate functionality only serves to confuse the users of the API.

6. ***Use strong, static typing wherever possible***

Strong typing allows for more error checking to be done at compile time; also, less error appear at runtime.

7. **Keep the common cases simple**

Because more often than not, the usual SQL calls used by the programmer are simple SELECT’s, INSERT’s, DELETE’s and UPDATE’s, these queries should be simple to perform with JDBC. However, more complex SQL statements should also be possible.

Finally we decided to proceed the implementation using Java [Networking](#).

And for dynamically updating the cache table we go for MS [Access](#) database.

Java ha two things: a programming language and a platform.

Java is a high-level programming language that is all of the following

Simple

Architecture-neutral

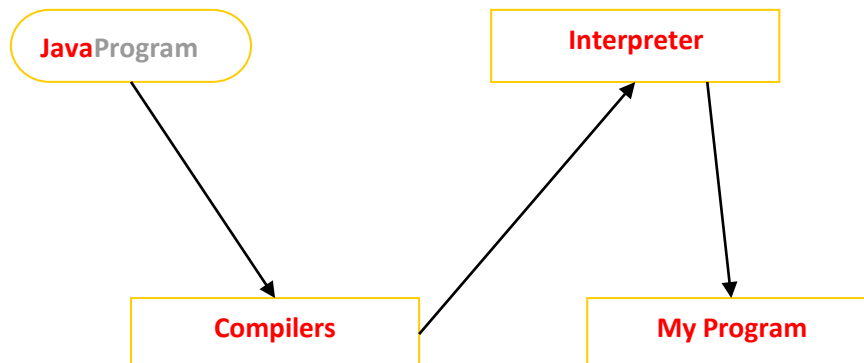
Object-oriented

Portable

Distributed	High-performance
Interpreted	multithreaded
Robust	Dynamic
Secure	

Java is also unusual in that each Java program is both compiled and interpreted. With a compiler you translate a Java program into an intermediate language called Java byte codes the platform-independent code instruction is passed and run on the computer.

Compilation happens just once; interpretation occurs each time the program is executed. The figure illustrates how this works.



You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it's a Java development tool or a Web browser that can run Java applets, is an implementation of the Java VM. The Java VM can also be implemented in hardware.

Java byte codes help make “write once, run anywhere” possible. You can compile your Java program into byte codes on my platform that has a Java compiler. The byte codes can then be run any implementation of the

Java VM. For example, the same Java program can run Windows NT, Solaris, and Macintosh.

Networking

TCP/IP stack

The TCP/IP stack is shorter than the OSI one:



TCP is a connection-oriented protocol; UDP (User Datagram Protocol) is a connectionless protocol.

IP datagram's

The IP layer provides a connectionless and unreliable delivery system. It considers each datagram independently of the others. Any association between datagram must be supplied by the higher layers. The IP layer supplies a checksum that includes its own header. The header includes the source and destination addresses. The IP layer handles routing through an Internet. It is also responsible for breaking up large datagram into smaller ones for transmission and reassembling them at the other end.

Port addresses A service exists on a host, and is identified by its port. This is a 16 bit number. To send a message to a server, you send it to the port for that service of the host that it is running on. This is not location transparency! Certain of these ports are "well known"

Sockets

A socket is a data structure maintained by the system to handle network connections. A socket is created using the call `socket`. It returns an integer that is like a file descriptor. In fact, under Windows, this handle can be used with Read File and Write File functions.

```
#include <sys/types.h>
#include <sys/socket.h>
int socket(int family, int type, int protocol);
```

Here "family" will be `AF_INET` for IP communications, protocol will be zero, and type will depend on whether TCP or UDP is used. Two processes wishing to communicate over a network create a socket each. These are similar to two ends of a pipe - but the actual pipe does not yet exist.

JFree Chart

JFreeChart is a free 100% Java chart library that makes it easy for developers to display professional quality charts in their applications. JFreeChart's extensive feature set includes:

A consistent and well-documented API, supporting a wide range of chart types;

A flexible design that is easy to extend, and targets both server-side and client-side applications;

Support for many output types, including Swing components, image files (including PNG and JPEG), and vector graphics file formats (including PDF, EPS and SVG);

JFreeChart is "open source" or, more specifically, [free software](#). It is distributed under the terms of the [GNU Lesser General Public Licence](#) (LGPL), which permits use in proprietary applications.

1. Map Visualizations

Charts showing values that relate to geographical areas. Some examples include: (a) population density in each state of the United States, (b) income per capita for each country in Europe, (c) life expectancy in each country of the world. The tasks in this project include:

Sourcing freely redistributable vector outlines for the countries of the world, states/provinces in particular countries (USA in particular, but also other areas);

Creating an appropriate dataset interface (plus default implementation), a rendered, and integrating this with the existing XYPlot class in JFreeChart;

Testing, documenting, testing some more, documenting some more.

2. Time Series Chart Interactivity

Implement a new (to JFreeChart) feature for interactive time series charts --- to display a separate control that shows a small version of ALL the time series data, with a sliding "view" rectangle that allows you to select the subset of the time series data to display in the main chart.

3. Dashboards

There is currently a lot of interest in dashboard displays. Create a flexible dashboard mechanism that supports a subset of JFreeChart chart types (dials, pies, thermometers, bars, and lines/time series) that can be delivered easily via both Java Web Start and an applet.

4. Property Editors

The property editor mechanism in JFreeChart only handles a small subset of the properties that can be set for charts. Extend (or reimplement) this mechanism to provide greater end-user control over the appearance of the charts.

J2ME (Java 2 Micro edition):-

Sun Microsystems defines J2ME as "a highly optimized Java run-time environment targeting a wide range of consumer products, including pagers, cellular phones, screen-phones, digital set-top boxes and car navigation systems." Announced in June 1999 at the JavaOne Developer Conference, J2ME brings the cross-platform functionality of the Java language to smaller devices, allowing mobile wireless devices to share applications. With J2ME, Sun has adapted the Java platform for consumer products that incorporate or are based on small computing devices.

* **Connected Limited Device Configuration (CLDC:)** is used specifically with the KVM for 16-bit or 32-bit devices with limited amounts of memory. This is the configuration (and the virtual machine) used for developing small J2ME applications. Its size limitations make CLDC more interesting and challenging (from a development point of view) than CDC. CLDC is also the configuration that we will use for developing our drawing tool application. An example of a small wireless device running small applications is a Palm hand-held computer.

* **Connected Device Configuration (CDC)** is used with the C virtual machine (CVM) and is used for 32-bit architectures requiring more than 2 MB of memory. An example of such a device is a Net TV box.

* java.lang

* java.io

* java.util

* javax.microedition.io

* javax.microedition.lcdui

* javax.microedition.midlet

* javax.microedition.rms

3.5 SYSTEM DESIGN

UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

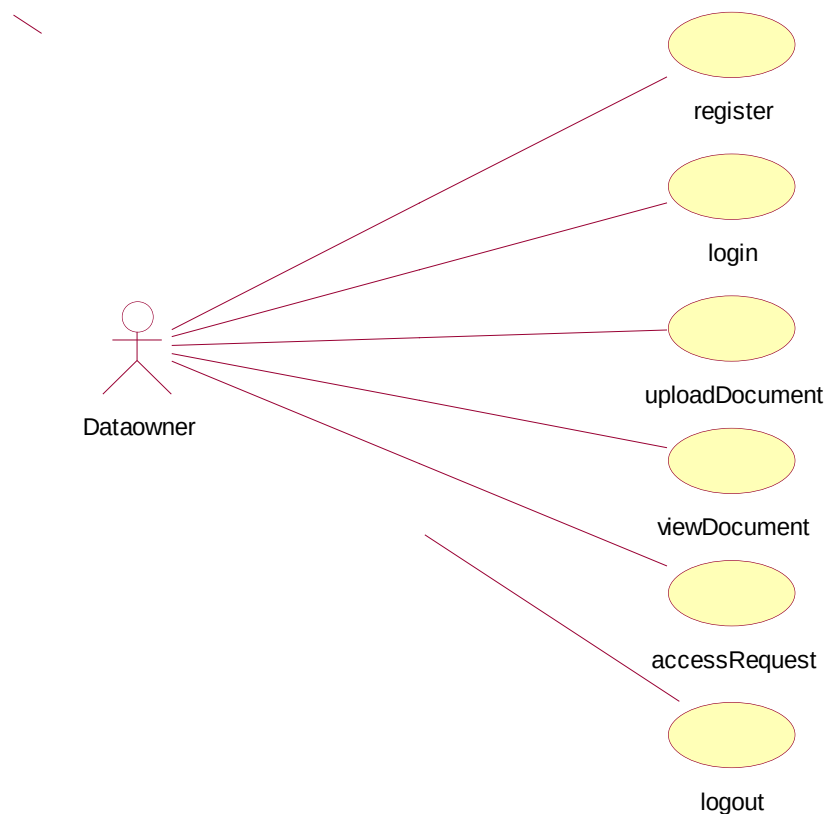
GOALS:

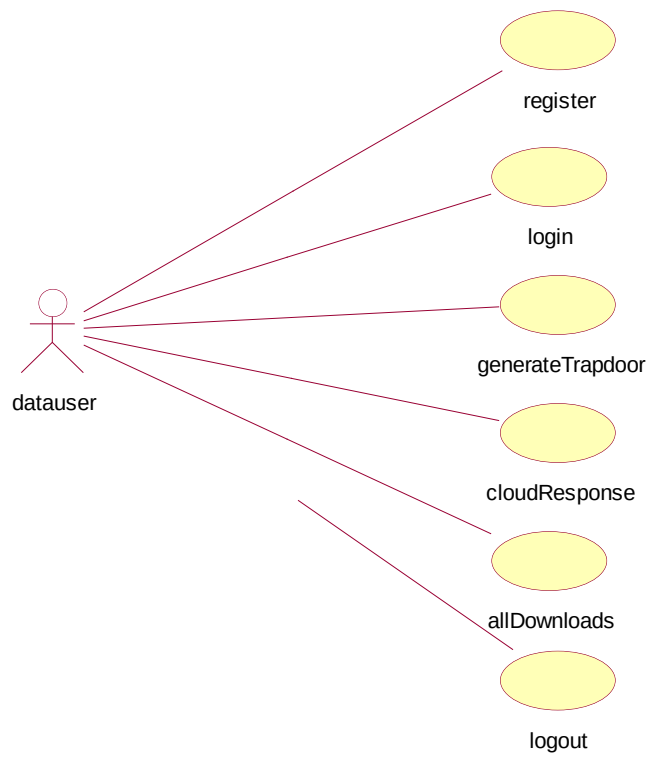
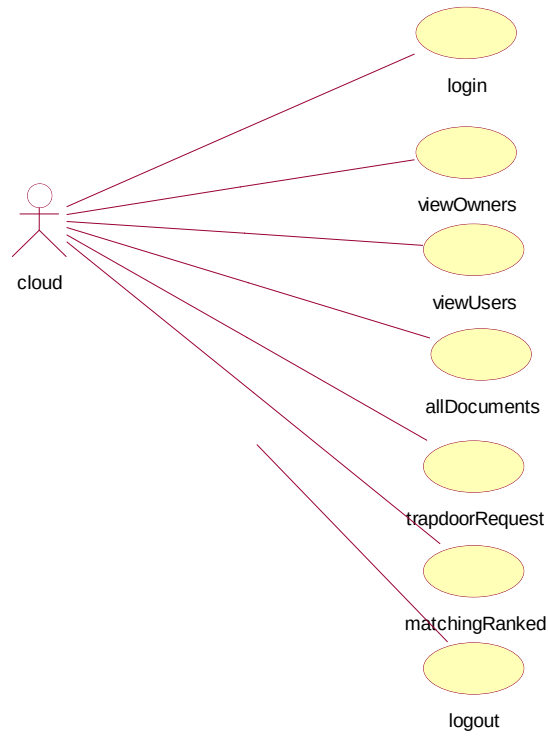
The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

USE CASE DIAGRAM:

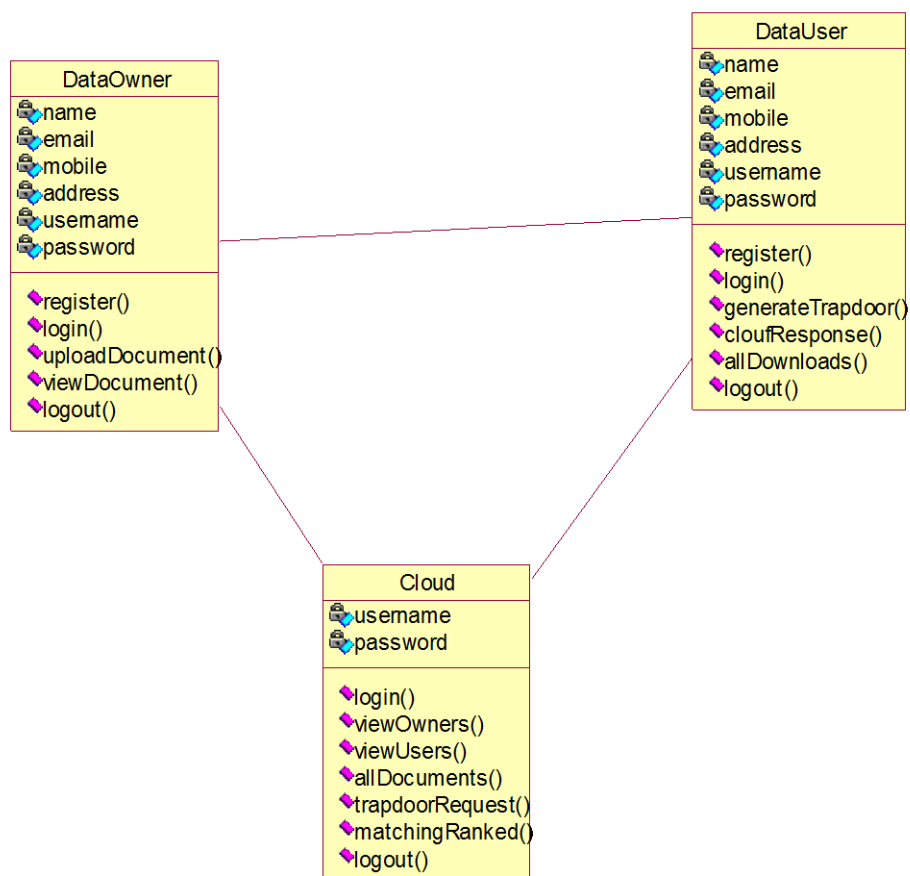
A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.





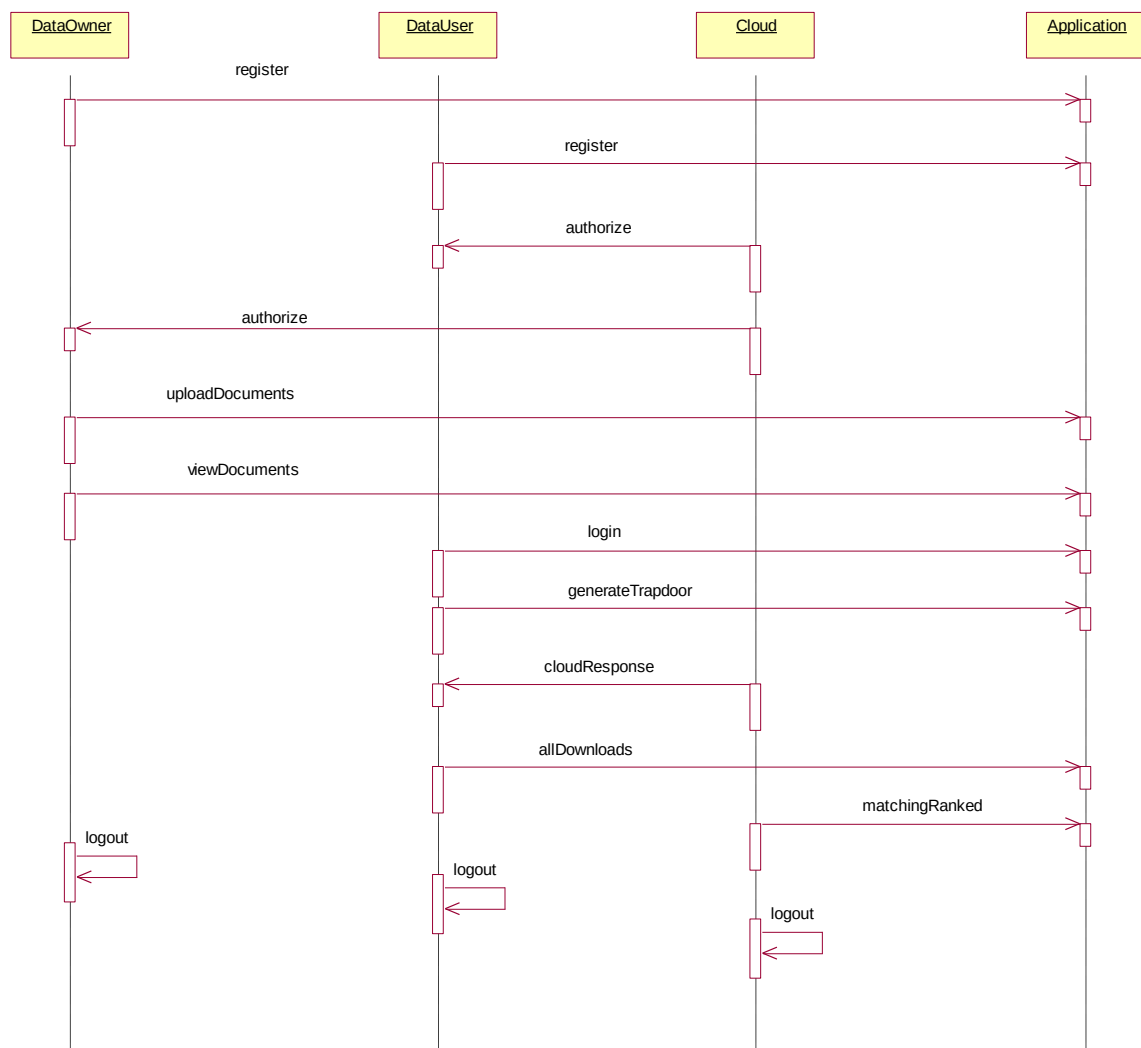
CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

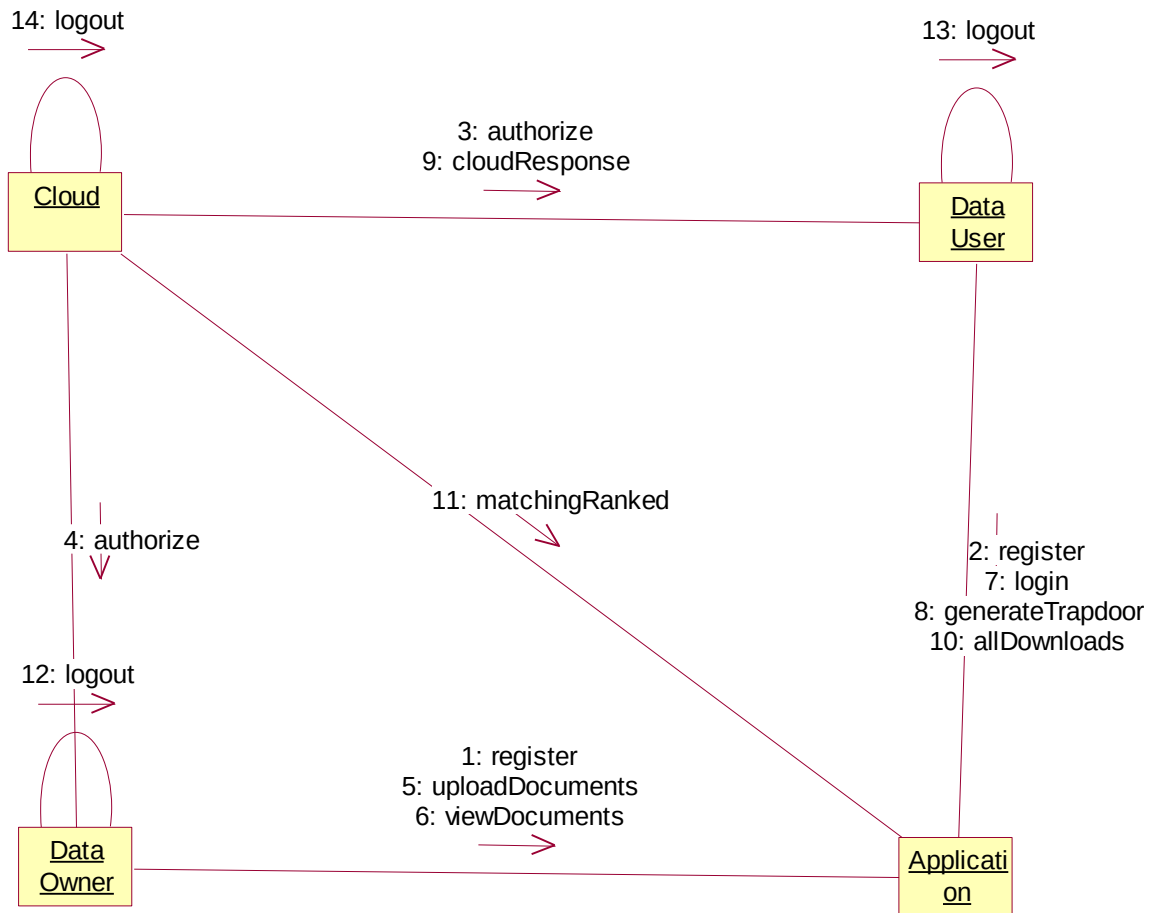


SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



Collaboration



3.6 SYSTEM TESTING

TESTING METHODOLOGIES

The following are the Testing Methodologies:

- **Unit Testing.**
- **Integration Testing.**
- **User Acceptance Testing.**
- **Output Testing.**
- **Validation Testing.**

Unit Testing

Unit testing focuses verification effort on the smallest unit of Software design that is the module. Unit testing exercises specific paths in a module's control structure to ensure complete coverage and maximum error detection. This test focuses on each module individually, ensuring that it functions properly as a unit. Hence, the naming is Unit Testing.

During this testing, each module is tested individually and the module interfaces are verified for the consistency with design specification. All important processing path are tested for the expected results. All error handling paths are also tested.

Integration Testing

Integration testing addresses the issues associated with the dual problems of verification and program construction. After the software has been integrated a set of high order tests are conducted. The main objective in this testing process is to take unit tested modules and builds a program structure that has been dictated by design.

The following are the types of Integration Testing:

1)Top Down Integration

This method is an incremental approach to the construction of program structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main program module. The module subordinates to the main program module are incorporated into the structure in either a depth first or breadth first manner.

In this method, the software is tested from main module and individual stubs are replaced when the test proceeds downwards.

2. Bottom-up Integration

This method begins the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated from the bottom up,

processing required for modules subordinate to a given level is always available and the need for stubs is eliminated. The bottom up integration strategy may be implemented with the following steps:

- The low-level modules are combined into clusters into clusters that perform a specific Software sub-function.
- A driver (i.e.) the control program for testing is written to coordinate test case input and output.
- The cluster is tested.
- Drivers are removed and clusters are combined moving upward in the program structure

The bottom up approaches tests each module individually and then each module is module is integrated with a main module and tested for functionality.

OTHER TESTING METHODOLOGIES

User Acceptance Testing

User Acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required. The system developed provides a friendly user interface that can easily be understood even by a person who is new to the system.

Output Testing

After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in the specified format. Asking the users about the format required by them tests the outputs generated or displayed by the system under consideration. Hence the output format is considered in 2 ways – one is on screen and another in printed format.

Validation Checking

Validation checks are performed on the following fields.

Text Field:

The text field can contain only the number of characters lesser than or equal to its size. The text fields are alphanumeric in some tables and alphabetic in other tables. Incorrect entry always flashes and error message.

Numeric Field:

The numeric field can contain only numbers from 0 to 9. An entry of any character flashes an error messages. The individual modules are checked for accuracy and what it has to perform. Each module is subjected to test run along with sample data. The individually tested modules are integrated into a single system. Testing involves executing the real data information is used in the program the existence of any program defect is inferred from the output. The testing should be planned so that all the requirements are individually tested.

A successful test is one that gives out the defects for the inappropriate data and produces an output revealing the errors in the system.

Preparation of Test Data

Taking various kinds of test data does the above testing. Preparation of test data plays a vital role in the system testing. After preparing the test data the system under study is tested using that test data. While testing the system by using test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

Using Live Test Data:

Live test data are those that are actually extracted from organization files. After a system is partially constructed, programmers or analysts often ask users to key in a set of data from their normal activities. Then, the systems person uses this data as a way to partially test the system. In other instances, programmers or analysts extract a set of live data from the files and have them entered themselves.

It is difficult to obtain live data in sufficient amounts to conduct extensive testing. And, although it is realistic data that will show how the system will perform for the typical processing requirement, assuming that the live data entered are in fact typical, such data generally will not test all combinations or formats that can enter the system. This bias toward typical values then does not provide a true systems test and in fact ignores the cases most likely to cause system failure.

Using Artificial Test Data:

Artificial test data are created solely for test purposes, since they can be generated to test all combinations of formats and values. In other words, the artificial data, which can quickly be prepared by a data generating utility program in the information systems department, make possible the testing of all login and control paths through the program.

The most effective test programs use artificial test data generated by persons other than those who wrote the programs. Often, an independent team of testers formulates a testing plan, using the systems specifications.

The package "Virtual Private Network" has satisfied all the requirements specified as per software requirement specification and was accepted.

USER TRAINING

Whenever a new system is developed, user training is required to educate them about the working of the system so that it can be put to efficient use by those for whom the system has been primarily designed. For this purpose the normal working of the project was demonstrated to the prospective users. Its working is easily understandable and since the expected users are people who have good knowledge of computers, the use of this system is very easy.

MAINTAINENCE

This covers a wide range of activities including correcting code and design errors. To reduce the need for maintenance in the long run, we have more accurately defined the user's requirements during the process of system development. Depending on

the requirements, this system has been developed to satisfy the needs to the largest possible extent. With development in technology, it may be possible to add many more features based on the requirements in future. The coding and designing is simple and easy to understand which will make maintenance easier.

TESTING STRATEGY :

A strategy for system testing integrates system test cases and design techniques into a well planned series of steps that results in the successful construction of software. The testing strategy must co-operate test planning, test case design, test execution, and the resultant data collection and evaluation .A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high level tests that validate major system functions against user requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing represents an interesting anomaly for the software. Thus, a series of testing are performed for the proposed system before the system is ready for user acceptance testing.

SYSTEM TESTING:

Software once validated must be combined with other system elements (e.g. Hardware, people, database). System testing verifies that all the elements are proper and that overall system function performance is achieved. It also tests to find discrepancies between the system and its original objective, current specifications and system documentation.

UNIT TESTING:

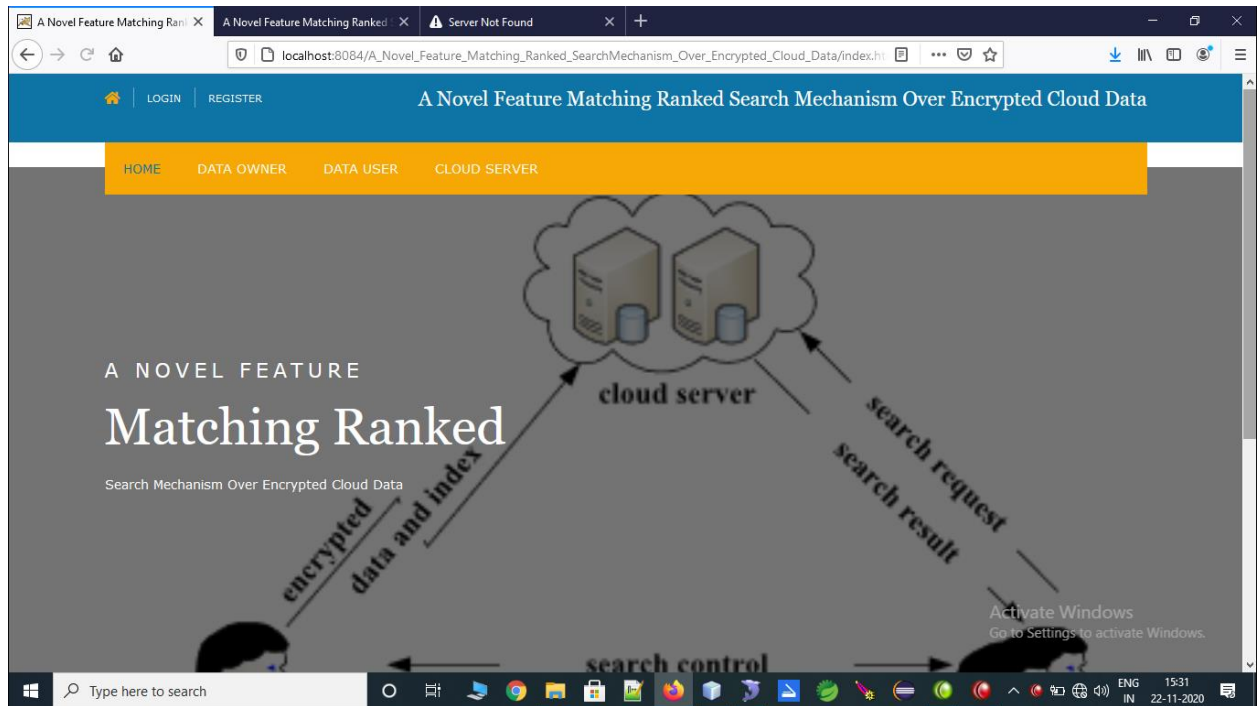
In unit testing different are modules are tested against the specifications produced during the design for the modules. Unit testing is essential for verification of the code produced during the coding phase, and hence the goals to test the internal logic of the modules. Using the detailed design description as a guide, important Conrail paths are tested to uncover errors within the boundary of the modules. This testing is carried out during the programming stage itself. In this type of testing step, each

module was found to be working satisfactorily as regards to the expected output from the module.

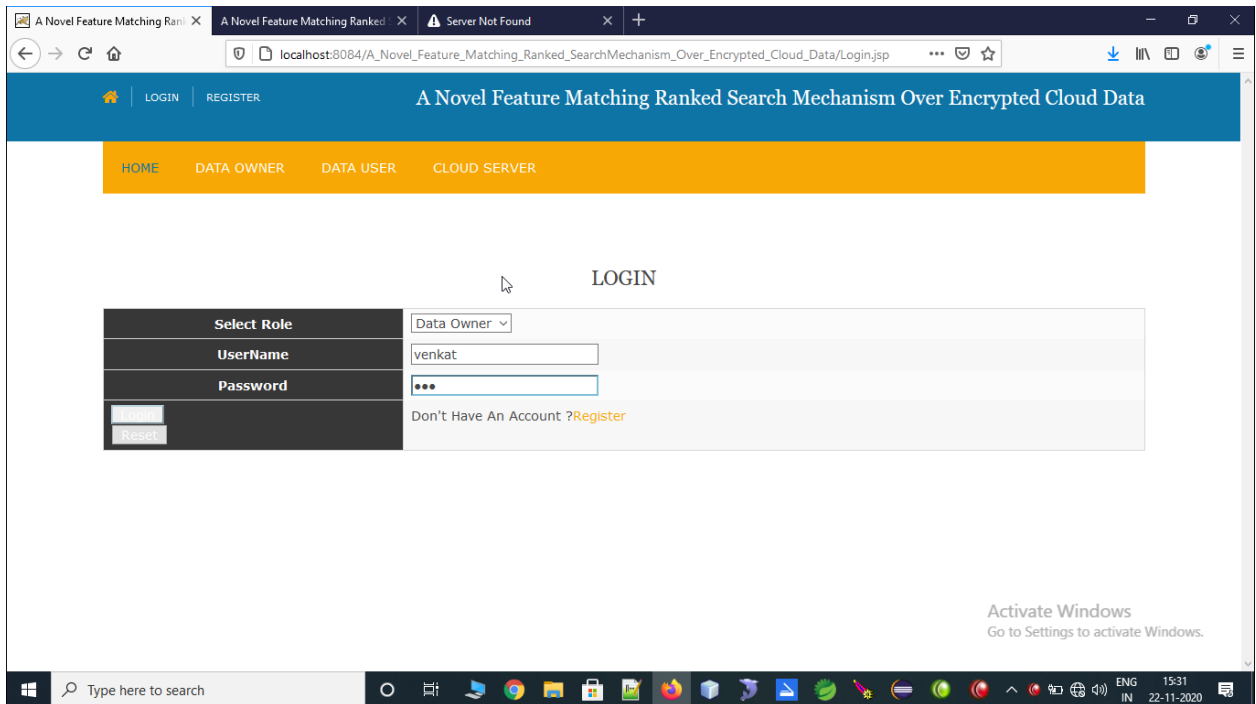
In Due Course, latest technology advancements will be taken into consideration. As part of technical build-up many components of the networking system will be generic in nature so that future projects can either use or interact with this. The future holds a lot to offer to the development and refinement of this project.

4. Results & Discussion

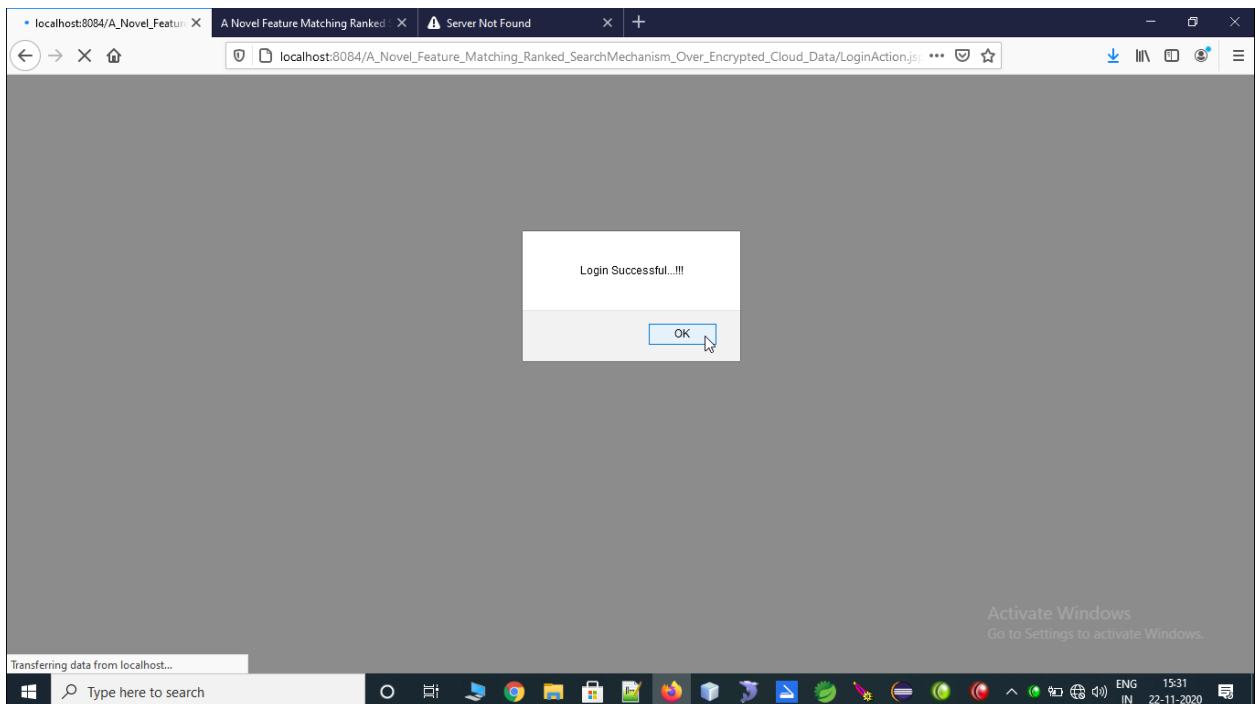
4.1 Home screen



4.2 Owner Login



4.3 Login status



4.4 Upload file

A Novel Feature Matching Ranked Search Mechanism Over Encrypted Cloud Data

HOME UPLOAD DOCUMENT VIEW DOCUMENTS LOGOUT

UPLOAD DOCUMENT

Document	Browse... malaria.txt
Document Name	malaria.txt
Document Index	malaria, I
Choose Date	dd / mm / yyyy
Document Data	1-8-59/B,Chikkadpally Stock and Sales report for the company GLENMARK between 01/08/2020 and 31/08/2020 Page - 1----- -----Product Name Packing Opening Stock Purchases Sales Closing Stock Qty Value Qty Value Qty

Activate Windows
Go to Settings to activate Windows.

4.5 Encrypted data

A Novel Feature Matching Ranked Search Mechanism Over Encrypted Cloud Data

HOME UPLOAD DOCUMENT VIEW DOCUMENTS LOGOUT

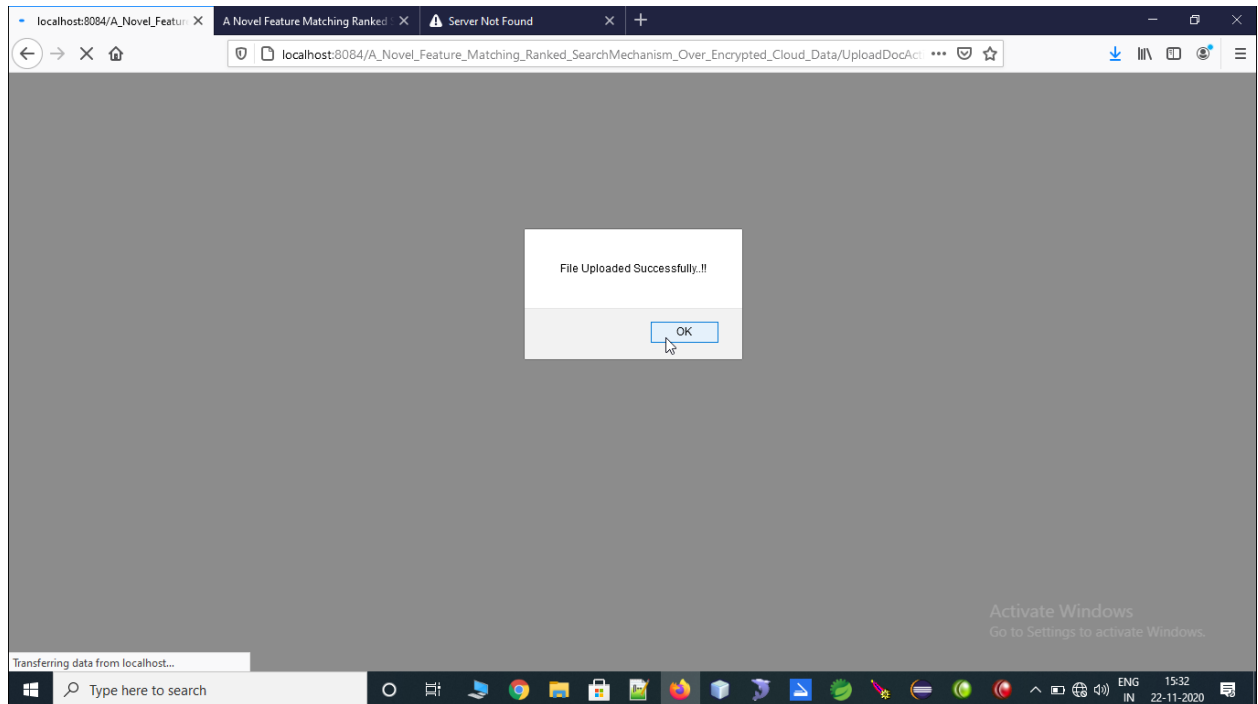
UPLOAD DOCUMENT

Upload Date	2020-11-22
Document Name	yYmJaXUUMiaqRjSIu3f2Vw=
Document Index	aaandZHU94nFBCXy2RCV2ZlpVh5jY0kSjyZ EshldDSHk=
Document Data	mMqrywH9NphokFb6CizQAudSkICEVS09 wn4iZFSwg9kEKfggbW/Q7a1jIbKRwmqG kgt/FcC78Pny rrXHO9ZV4BLObKwBeX0b5zTXRldUScn C7SrqpS275WuIb3spVTDxxhiHKkzA1u8 Ysib+W2J0EJAS 0JWR2MFONLxIRj5X9/XlqjUsgHnub+bmU 9ZCa9P+659N /SKpWmX1hrh4UD99gUAP /nAxqz8QD7ss gQH73d5UDXd3lxsE5LYG457PpxlpVA13

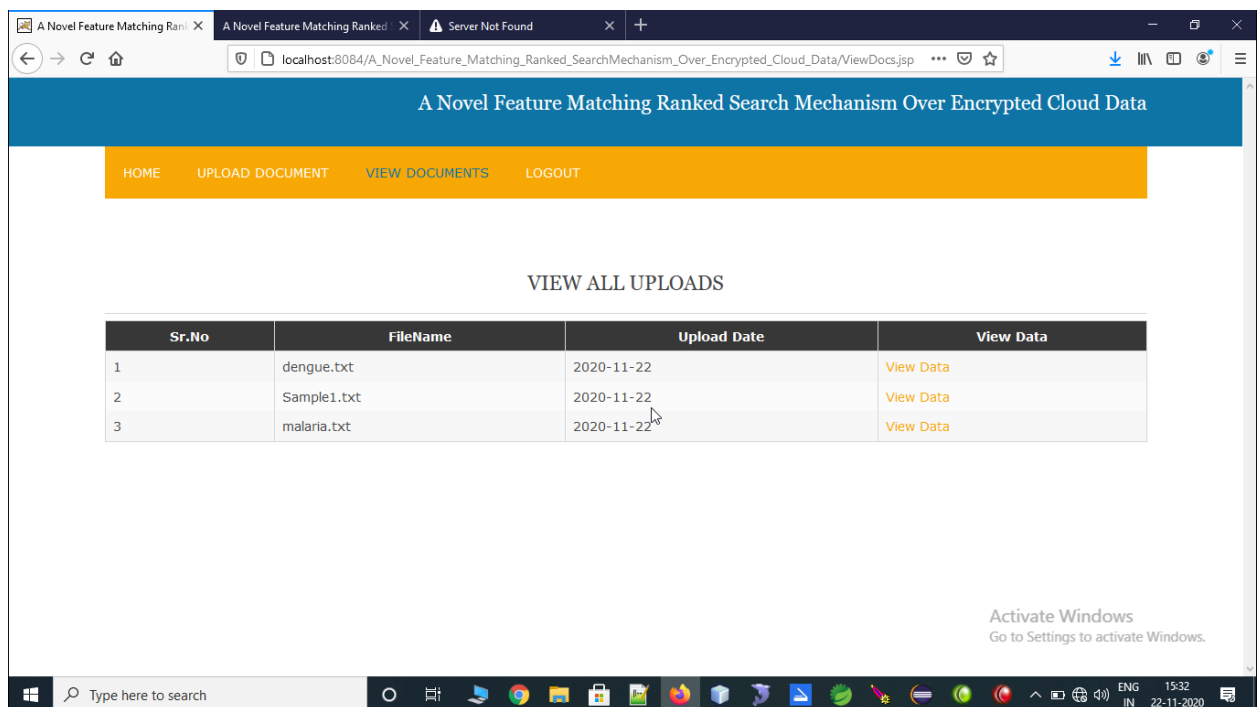
Upload

Activate Windows
Go to Settings to activate Windows.

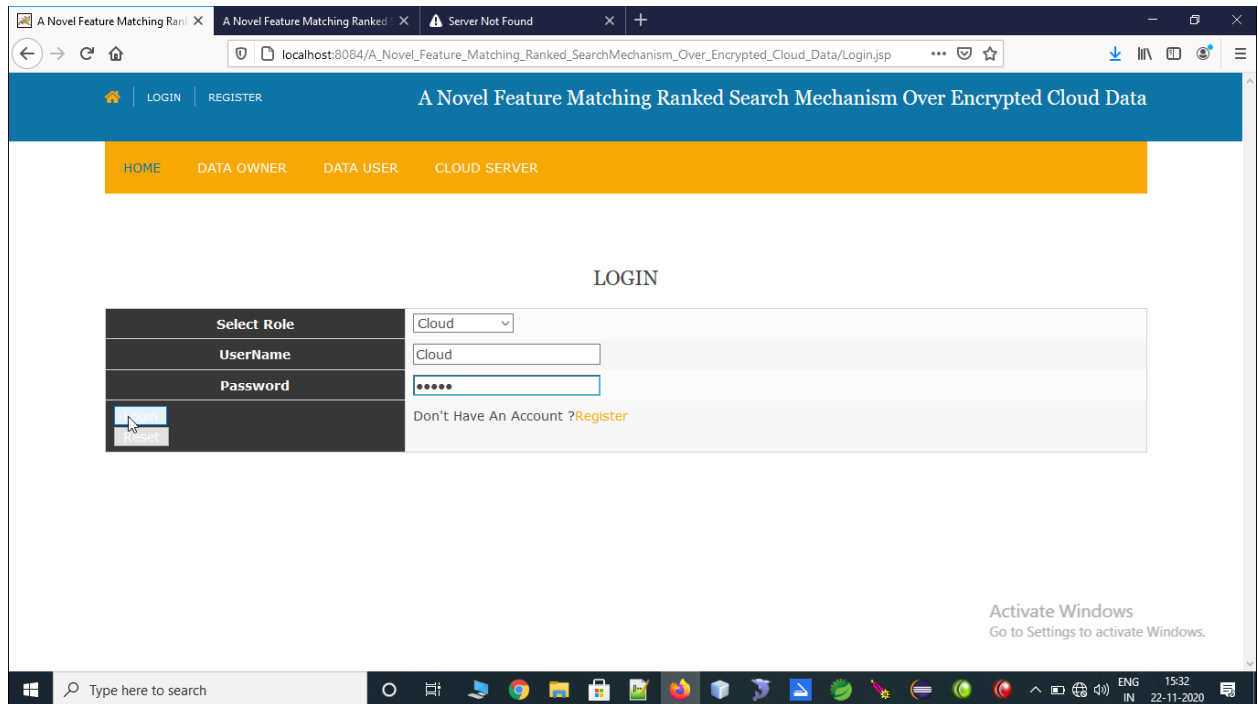
4.6 Upload status



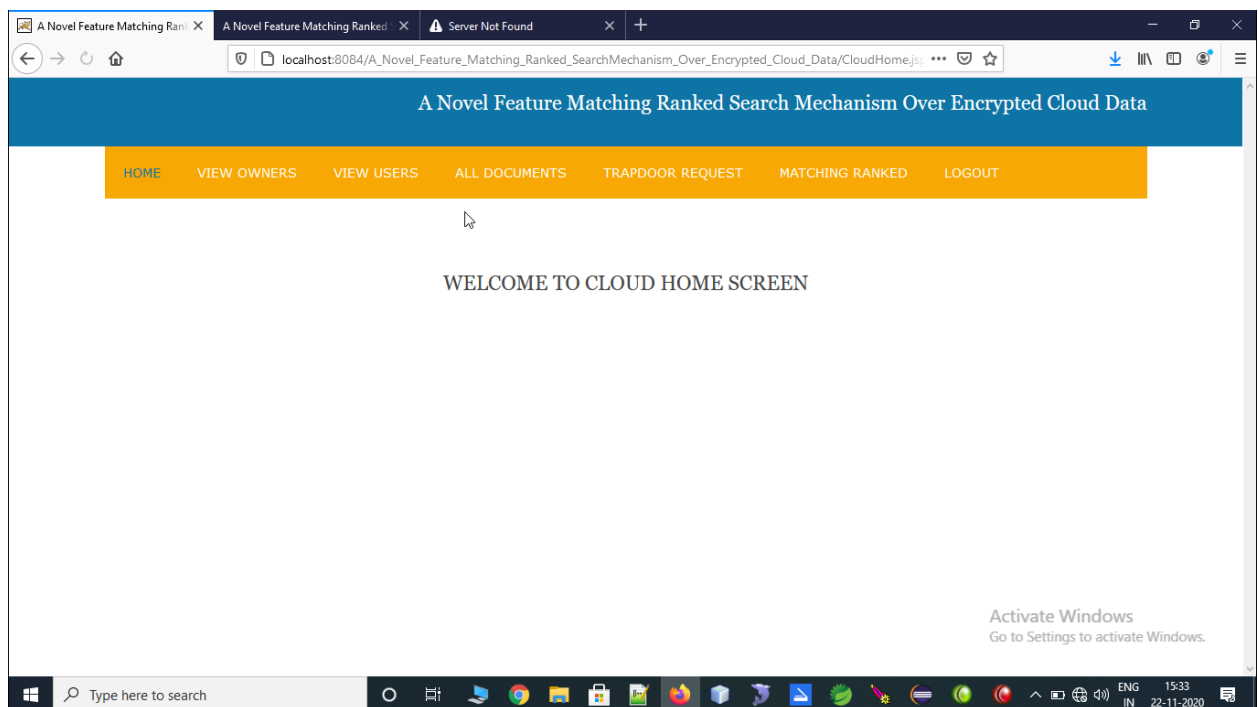
4.7 View all uploads



4.8 Cloud login



Cloud home



4.9 View owners

A Novel Feature Matching Ranked Search Mechanism Over Encrypted Cloud Data

HOME VIEW OWNERS VIEW USERS ALL DOCUMENTS TRAPDOOR REQUEST MATCHING RANKED LOGOUT

ALL REGISTERED DATAOWNERS

NAME	EMAIL	MOBILE	GENDER	ADDRESS	STATUS
venkat	venkat@gmail.com	9640340708	Male	hyd	Authorized

Activate Windows
Go to Settings to activate Windows.

All user details

All documents

A Novel Feature Matching Ranked Search Mechanism Over Encrypted Cloud Data

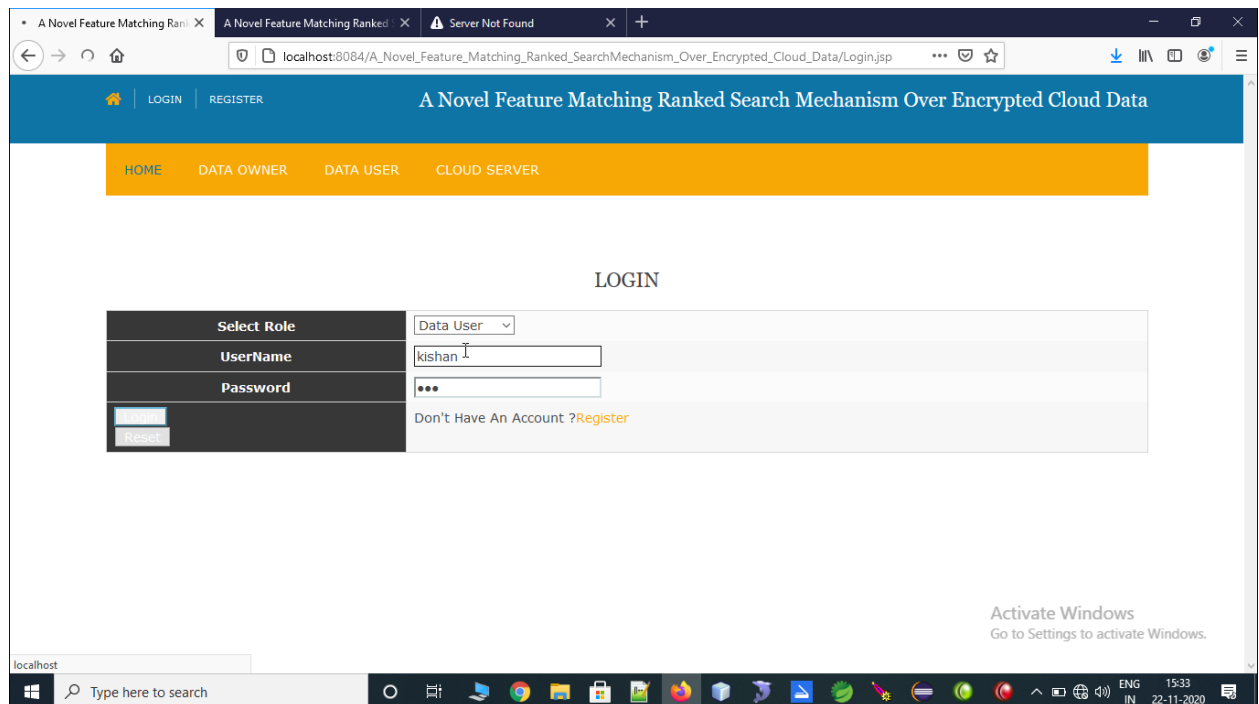
VIEW OWNERS VIEW USERS ALL DOCUMENTS TRAPDOOR REQUEST MATCHING RANKED LOGOUT

ALL DOCUMENTS

Sr.No	OwnerName	FileName	Upload Date	View Data
1	venkat	dengue.txt	2020-11-22	View Data
2	venkat	Sample1.txt	2020-11-22	View Data
3	venkat	malaria.txt	2020-11-22	View Data

Activate Windows
Go to Settings to activate Windows.

Data user login

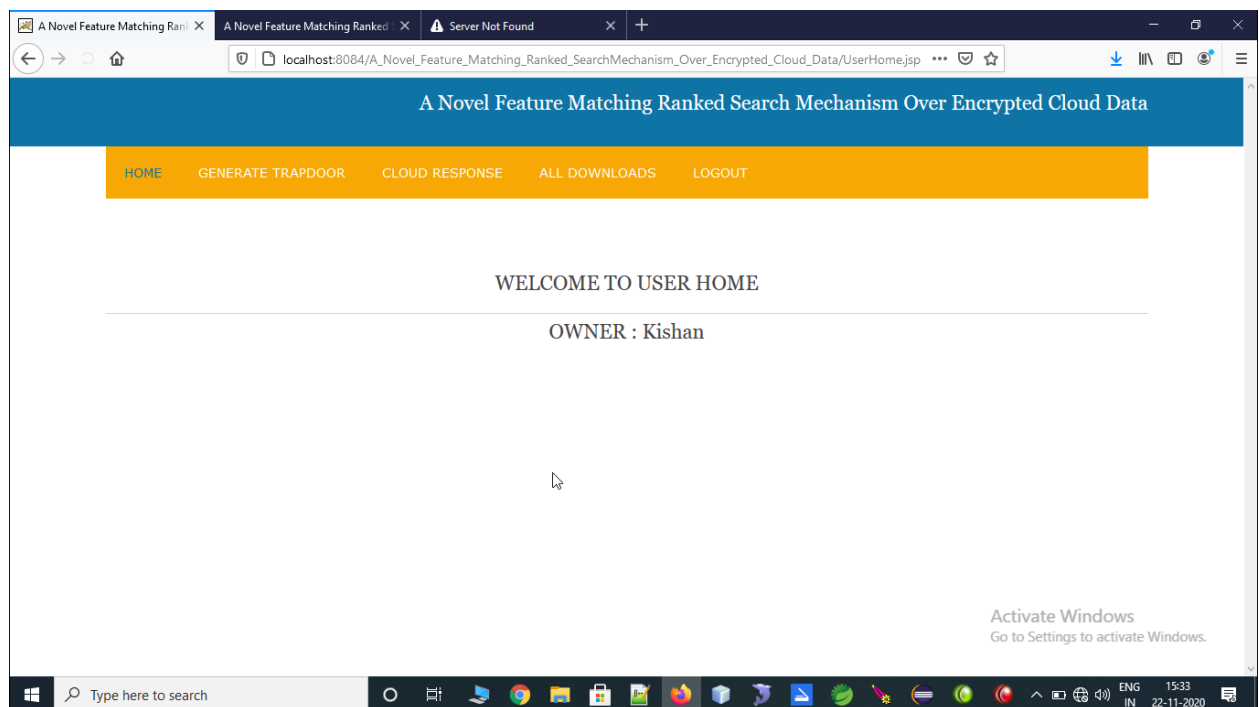


The screenshot shows a web browser window with the URL `localhost:8084/A_Novel_Feature_Matching_Ranked_SearchMechanism_Over_Encrypted_Cloud_Data/Login.jsp`. The page has a blue header with the title "A Novel Feature Matching Ranked Search Mechanism Over Encrypted Cloud Data" and navigation links for LOGIN and REGISTER. Below the header is an orange navigation bar with links for HOME, DATA OWNER, DATA USER, and CLOUD SERVER. The main content area is titled "LOGIN" and contains a form with the following fields:

Select Role	Data User
UserName	kishan
Password	...
Don't Have An Account ?Register	

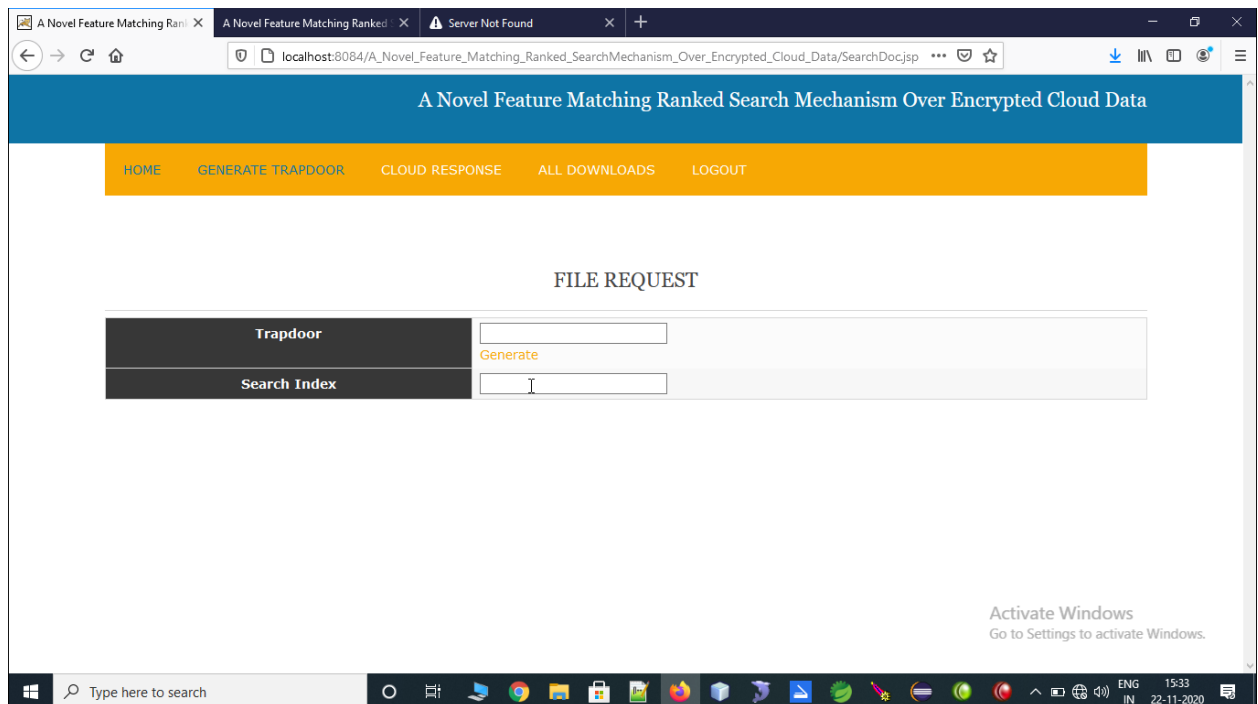
At the bottom right, there is an "Activate Windows" watermark with the text "Go to Settings to activate Windows." The Windows taskbar at the bottom shows the time as 15:33 on 22-11-2020.

User home

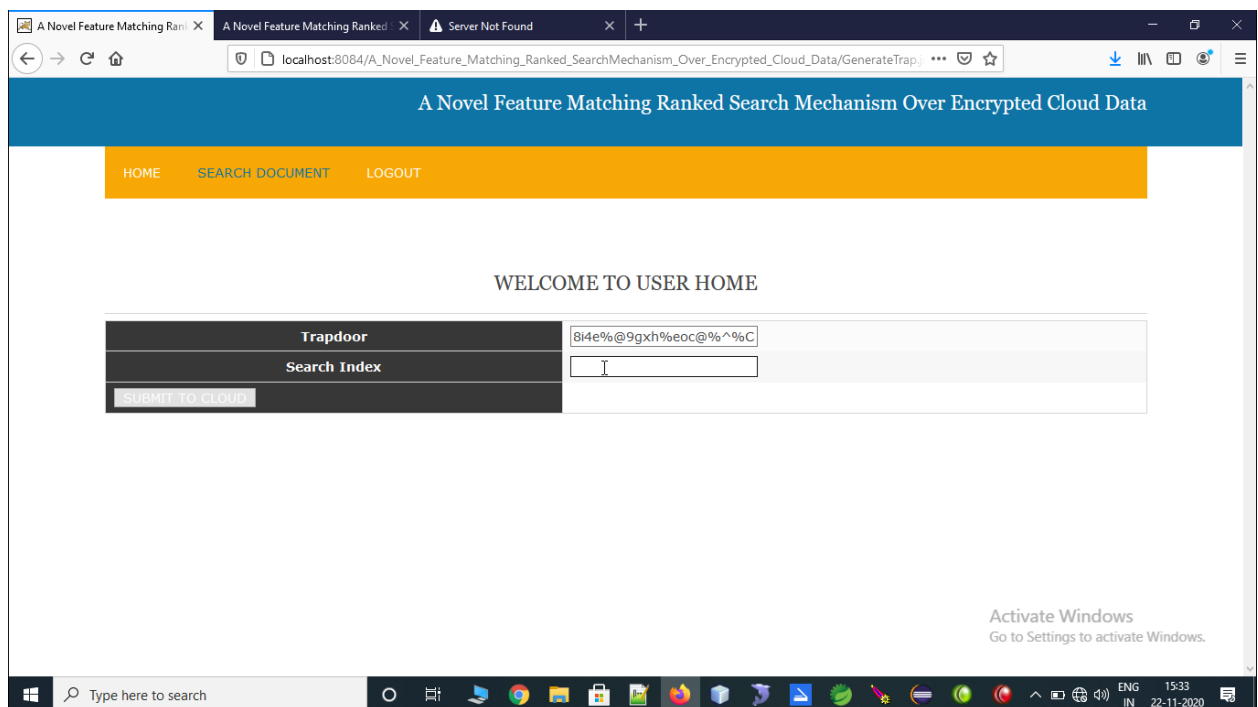


The screenshot shows the same web browser window, but the URL is `localhost:8084/A_Novel_Feature_Matching_Ranked_SearchMechanism_Over_Encrypted_Cloud_Data/UserHome.jsp`. The page has the same blue header and orange navigation bar. The main content area is titled "WELCOME TO USER HOME" and displays the text "OWNER : Kishan". At the bottom right, there is an "Activate Windows" watermark with the text "Go to Settings to activate Windows." The Windows taskbar at the bottom shows the time as 15:33 on 22-11-2020.

File request



Trapdoor generated



Cloud response

A Novel Feature Matching Ranked Search Mechanism Over Encrypted Cloud Data

HOME GENERATE TRAPDOOR CLOUD RESPONSE ALL DOWNLOADS LOGOUT

WELCOME TO USER HOME

File Name	Transfer Status	Document	Access Request
dengue.txt	Transferred	B7RhJtWgDqe6QdlkVqXan5AbxSV0VDYrOyUiroFAQRm6Fdkqc54 B5whCxi6ZuJtK7c5WxQyWn OpS0vP1QD+6ax8ugs/99uifXPnXWd4RkITT0XDvyVuudHqYkfL12v 6XnxbdaaOzDfo5wP2QuPEmU na7VgAwQFQNZ3pDK3P12R2sL8wXHAKO/vB2XZJS6SdQJN1+Om wvYEbYV7nJrtIKc3fXbn28Ts1Ni tWwlmz12zMoc3bf4t8bw/aJh8ICaxCbD84Rkqiltgw84kErXm0ZUw5 6QG/IZDCoxh4H07etkBRPj gdP5o+M2epKYD+wxYAaxMJ2zE8IRjrNihmrEgviIv35QkGwW/7ymL 9YlvivHENmFlixwpiB3uHoXD DSi3CcuXYBgstyKucLPB11BAIbQU/QyZwAxbNOMETWkv2Ja /fkmnxABZhKA/cKUtktZch9rTw79 LFr+2xE9nFcPUVVrkvhSx0fTa6YDaSebiG3nOQ49sX607PgTpLfizVP bqOV7tgH/BfuO/+GJJsMd KQ3ZTjJqXhwXnwxIRbDyV3ZHTGL4XM60fLAmxAF10d9YEKBK2zN IN/0RRmRUwQLtfZzaSbmAaUtr m3uQ80MStfs7dQCQXdzC2quFpb4PntxqBP/dtrSdbpd/HnEGaRLni /QECgWC3qd1oB3lDkRjTe/8 6k1Xb7vfxeoIglR12ltQZYfaAP2XZebAITFRn8X4P2cU9rtSqt3S612	Request Verified

localhost:8084/A_Novel_Feature_Matching_Ranked_SearchMechanism_Over_Encrypted_Cloud_Data/CloudResp.jsp

Activate Windows
Go to Settings to activate Windows.

Access request at owner

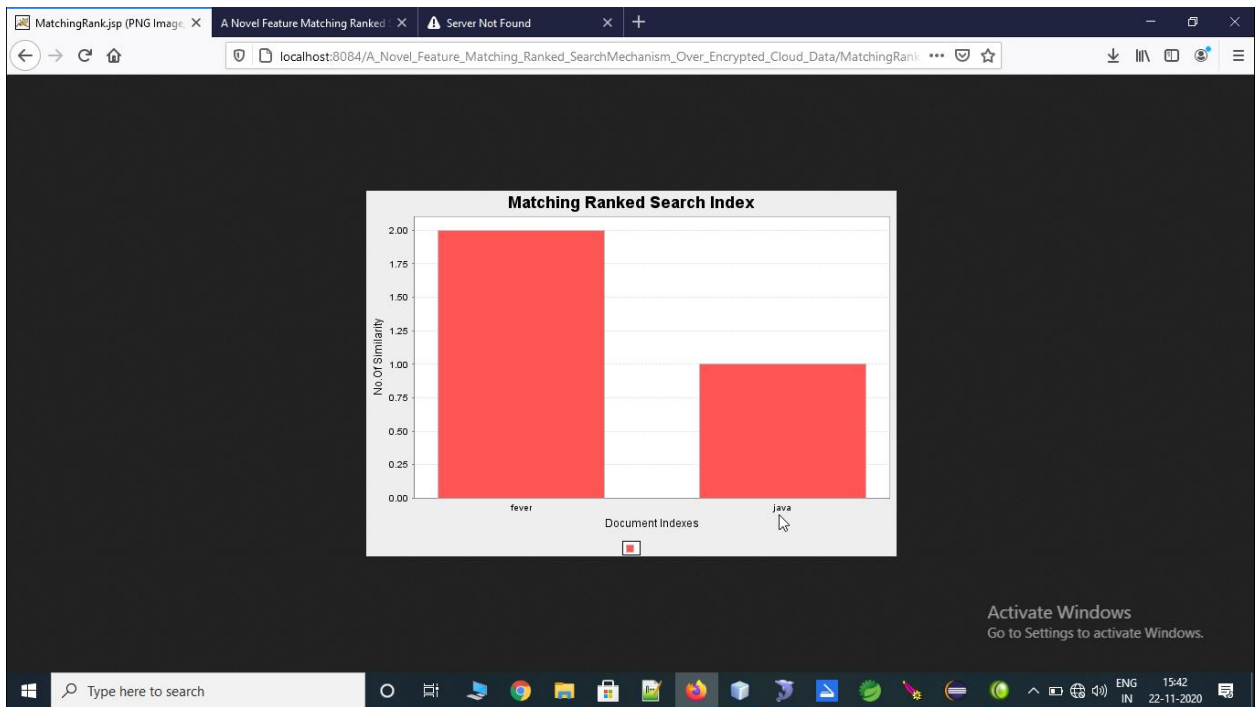
A Novel Feature Matching Ranked Search Mechanism Over Encrypted Cloud Data

HOME UPLOAD DOCUMENT VIEW DOCUMENTS ACCESS REQUEST LOGOUT

ACCESS REQUEST

Sr.No	File ID	Status
1	4	access_verifies
2	5	access_verifies
3	6	Accept Request

Activate Windows
Go to Settings to activate Windows.



4. CONCLUSION

In this paper, we propose a novel feature matching ranked search mechanism for encrypted cloud data. In this mechanism, a feature score algorithm is used to create indexes so that a plurality of keywords extracted from a document are only mapped to one dimension of the index. Comparing with creating indexes with independent keywords, this mechanism can reduce the index dimension. In addition, a matching score algorithm is designed in the generating trapdoor process of FMRS. This algorithm can give the query request an accurate score based on the type of match and the number of matching keywords, so that the query results are more inline with users' actual search requests. It can be seen from the experiment results that our mechanism can speed up the creation of index, the generation of trapdoor, and the search process. Moreover, our mechanism can save storage cost and improve the ranking accuracy.

5. References

- [1] Q. Chai and G. Gong, "Verifiable symmetric searchable encryption for semi-honest-but-curious cloud servers," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Ottawa, ON, Canada, Jun. 2012, pp. 917–922.
- [2] X. Zhang, C. Xu, R. Xie, and C. Jin, "Designated cloud server public key encryption with keyword search from lattice in the standard model," *Chin. J. Electron.*, vol. 27, no. 2, pp. 304–309, Mar. 2018.
- [3] Y. Chang and M. Mitzenmacher, "Privacy preserving keyword searches on remote encrypted data," in *Proc. Int. Conf. Appl. Cryptogr. Netw. Secur.*, vol. 5. Berlin, Germany: Springer, 2005, pp. 442–455.
- [4] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searching on encrypted data," in *Proc. IEEE Symp. Secur. Privacy (S&P)*, Berkeley, CA, USA, May 2000, pp. 44–55.
- [5] A. J. Aviv, M. E. Locasto, S. Potter, and A. D. Keromytis, "SSARES: Secure searchable automated remote email storage," in *Proc. 23rd Annu. Comput. Secur. Appl. Conf. (ACSAC)*, Miami Beach, FL, USA, Dec. 2007, pp. 129–139.
- [6] M. Raykova, B. Vo, S. M. Bellovin, and T. Malkin, "Secure anonymous database search," in *Proc. ACM Workshop Cloud Comput. Secur. (CCSW)*, New York, NY, USA, 2009, pp. 115–126.
- [7] S. M. Bellovin and W. R. Cheswick, "Privacy-enhanced searches using encrypted Bloom filters," *IACR Cryptol. ePrint Arch., Tech. Rep.*, 2004, vol. 22.
- [8] Z. Fu, K. Ren, J. Shu, X. Sun, and F. Huang, "Enabling personalized search over encrypted outsourced data with efficiency improvement," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 9, pp. 2546–2559, Sep. 2016.
- [9] Z. Fu, X. Sun, Z. Xia, L. Zhou, and J. Shu, "Multi-keyword ranked search supporting synonym query over encrypted data in cloud computing," in *Proc. IEEE 32nd Int. Perform. Comput. Commun. Conf. (IPCCC)*, San Diego, CA, USA, Dec. 2013, pp. 1–8.
- [10] H. Yin, Z. Qin, J. Zhang, W. Li, L. Ou, Y. Hu, and K. Li, "Secure conjunctive multi-keyword search for multiple data owners in cloud computing," in *Proc. IEEE 22nd Int. Conf. Parallel Distrib. Syst. (ICPADS)*, Wuhan, China, Dec. 2016, pp. 276–286.

- [11] Z. Fu, X. Wu, C. Guan, X. Sun, and K. Ren, "Toward efficient multi-keyword fuzzy search over encrypted outsourced data with accuracyimprovement," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 12, pp. 2706–2716, Dec. 2016.
- [12] Z. Guo, H. Zhang, C. Sun, Q. Wen, and W. Li, "Secure multi-keywordranked search over encrypted cloud data for multiple data owners," *J. Syst.Softw.*, vol. 137, pp. 380–395, Mar. 2018.
- [13] X. Jiang, J. Yu, J. Yan, and R. Hao, "Enabling efficient and verifiable multi-keyword ranked search over encrypted cloud data," *Inf. Sci.*, vols. 403–404, pp. 22–41, Sep. 2017.
- [14] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou, "Fuzzy keywordsearch over encrypted data in cloud computing," in *Proc. IEEE INFO-COM*, San Diego, CA, USA, Mar. 2010, pp. 1–5
- [15] M. A. M. Ahsan, F. Z. Chowdhury, M. Sabilah, A. W. B. A. Wahab, and M. Y. I. B. Idris, "An efficient fuzzy keyword matching technique forsearching through encrypted cloud data," in *Proc. Int. Conf. Res. Innov.Inf. Syst. (ICRIIS)*, Langkawi, Malaysia, Jul. 2017, pp. 1–5.
- [16] X. Zhu, Q. Liu, and G. Wang, "A novel verifiable and dynamic fuzzykeyword search scheme over encrypted data in cloud computing," in *Proc. IEEE Trustcom/BigDataSE/ISPA*, Tianjin, China, Aug. 2016, pp. 845–851.
- [17] H. Zhu, Z. Mei, B. Wu, H. Li, and Z. Cui, "Fuzzy keyword search andaccess control over ciphertexts in cloud computing," in *Proc. Australas.Conf. Inf. Secur. Privacy*. Cham, Switzerland: Springer, 2017, pp. 248–265.
- [18] Y. Tang, D. Gu, N. Ding, and H. Lu, "Phrase search over encrypted datawith symmetric encryption scheme," in *Proc. 32nd Int. Conf. Distrib.Comput. Syst. Workshops*, Macau, China, 2012, pp. 471–480.
- [19] E.-J. Goh, "es," *IACR Cryptol. ePrint Arch.*, Tech. Rep., 2003, p. 216.
- [20] D. Boneh, E. Kushilevitz, R. Ostrovsky, and W. Skeith, III, "Public keyencryption that allows PIR queries," in *Proc. Annu. Int. Cryptol. Conf.* Berlin, Germany: Springer, 2007, pp. 50–67.