**CHAPTER-1: SYSTEM ANALYSIS**

**1.1 Existing System**

The existing system, as mentioned in the abstract, likely refers to the current state of ranked keyword search schemes in cloud computing environments. Here's a brief overview of what the existing systems might encompass based on the information provided:

Focus on Search Efficiency or Functionality: Existing systems primarily concentrate on improving the efficiency or functionality of ranked keyword search in cloud environments. These systems might employ techniques such as indexing, keyword matching, or ranking algorithms to enhance search performance.

Lack of Efficient Access Control: While existing systems may excel in search efficiency, they often lack efficient access control mechanisms. This means that users may not have

fine-grained control over who can access specific data or documents stored in the cloud.

Limited Security Analysis: Furthermore, the existing systems may not undergo thorough formal security analysis to ensure the confidentiality of outsourced data and the privacy of index and tokens. This could potentially leave data vulnerable to unauthorized access or breaches.

Overall, the existing systems in this context are characterized by their focus on search efficiency or functionality, coupled with deficiencies in providing efficient access control and comprehensive security analysis. These limitations prompt the need for a more robust solution like the proposed Multi-keyword Ranked Search scheme with Fine-grained access control (MRSF).

**1.1.1 Disadvantages of Existing System**

**Computational Overhead:** Existing systems often suffer from high computational overhead due to the need for complex cryptographic operations such as homomorphic encryption or secure multi-party computation. These operations can significantly impact the efficiency and responsiveness of search queries.

**Limited Search Functionality**: Many existing systems support only basic keyword-based searches and lack advanced search features such as fuzzy matching, synonym search, or natural language processing. This limitation restricts the usefulness and flexibility of the search functionality.

**Access Control Complexity**: Implementing access control mechanisms in existing systems can be complex and cumbersome, especially when dealing with large datasets and dynamic access control policies. Managing and enforcing fine-grained access control rules may require significant administrative overhead.

**1.2 Proposed System**

In the proposed system, MRSF (Multi-Keyword Ranked Searchable Encryption) and fine-grained access control mechanisms are integral components to enhance search functionality and enforce precise access policies. Here’s how they are integrated into the proposed system:

**MRSF (Multi-Keyword Ranked Searchable Encryption):**

MRSF allows users to perform searches using multiple keywords and receive ranked search results based on relevance. In the proposed system, MRSF techniques are employed to enable efficient multi-keyword searches over encrypted data stored in the cloud. Encrypted keyword indexes are

generated during the indexing phase, facilitating quick retrieval of relevant documents based on multiple search terms. This capability enhances the search functionality of the system, enabling users to find the most relevant information efficiently.

**Fine-Grained Access Control:**

Fine-grained access control mechanisms enable data owners to define granular access policies that specify which users or user groups have permission to access individual documents or subsets of data. In the proposed system, fine-grained access control is implemented to ensure that access to encrypted documents is restricted according to the data owner’s specified policies. This level of control allows data owners to enforce precise access restrictions, such as restricting access to sensitive documents to only a subset of authorized users or requiring different levels of authorization for different types of data.

Fine-grained access control enhances the security of the system by limiting access to sensitive information and preventing unauthorized users from accessing confidential data.

By incorporating MRSF and fine-grained access control mechanisms into the proposed system, it offers advanced search functionality and robust access control capabilities. Users can efficiently search for relevant information using multiple keywords while ensuring that access to sensitive data is strictly controlled according to predefined access policies. This combination of features enhances the security, usability, and efficiency of the system, making it well-suited for practical applications in cloud environments.

**1.2.1 Advantages of Proposed System**

**Enhanced Security**: The proposed system ensures robust security by encrypting data before storing it in the cloud, preventing unauthorized access to sensitive information. By employing strong encryption mechanisms and access control policies, it mitigates the risk of data breaches and unauthorized data access.

**Confidentiality Preservation**: Through encryption techniques, the proposed system maintains the confidentiality of data stored in the cloud. Even though search operations are performed on encrypted data, the system ensures that plaintext information is never exposed to unauthorized parties, preserving data privacy.

**1.3 Introduction**

The introduction provides context on the security concerns associated with storing data in cloud computing environments and the common practice of encrypting data before outsourcing it to cloud servers. Here's a breakdown of the key points in the introduction:

Cloud Computing: Cloud computing offers convenient access to computation and storage resources, leading to widespread adoption by enterprises and individuals. However, outsourcing data to the cloud means relinquishing direct control over that data, raising security concerns, particularly for sensitive data like medical records or financial documents.

Data Encryption: To mitigate the risk of data leakage, data owners typically encrypt their data before storing it in the cloud. However, conventional encryption schemes prevent authorized operations on the encrypted data, hindering efficient information retrieval over outsourced data.

Searchable Symmetric Encryption (SSE): SSE is presented as a promising solution to balance data utilization and confidentiality. SSE-based designs, including Boolean keyword search schemes, enable keyword searches over encrypted data. However, existing schemes do not adequately support ranked search, and their complexity limits their application to large-scale cloud data.

Ranked Search Challenges: Existing ranked search schemes are limited in their functionalities

While some support single keyword search, others enable multi-keyword search but require rebuilding the keyword dictionary when new keywords are added.

Access Control: In addition to keyword search functionality, access control over encrypted cloud data is essential, especially for scenarios involving sensitive information like healthcare data.

Compliance with regulations such as HIPAA necessitates robust access control mechanisms.

Overall, the introduction highlights the need for efficient data retrieval mechanisms while ensuring data security and access control in cloud computing environments, setting the stage for the proposed research on a Multi-keyword Ranked Search scheme with Fine-grained access control (MRSF).

**CHAPTER-2: LITERATURE SURVEY**

**C. Wang, N. Cao, J. Li, K. Ren, and W. Lou [1]** states that “Secure ranked

keyword search over encrypted cloud data” : As Cloud Computing becomes prevalent, sensitive information are being increasingly centralized into the cloud. For the protection of data privacy, sensitive data has to be encrypted before outsourcing, which makes effective data utilization a very challenging task. Although traditional searchable encryption schemes allow users to securely search over encrypted data through keywords, these techniques support only boolean search, without capturing any relevance of data files. This approach suffers from two main drawbacks when directly applied in the context of Cloud Computing. On the one hand, users, who do not necessarily have pre-knowledge of the encrypted cloud data, have to post process every retrieved file in order to find ones most matching their interest, On the other hand, invariably retrieving all files containing the queried keyword further incurs unnecessary network traffic, which is absolutely undesirable in today's pay-as-you-use cloud paradigm.

**L. Zhang, Y. Zhang, and H. Ma [2]** states that “Privacy-preserving and dynamic

multi-attribute conjunctive keyword search over encrypted cloud

data” : With the increasing popularity of cloud computing, a growing data owners are motivated to outsource their huge data to cloud servers in order to facilitate access and save data management cost. To protect user privacy and data security, sensitive data should be encrypted before outsourced to the cloud server, which obsoletes data utilization like efficient search over encrypted data. In this paper, we present a privacy-preserving conjunctive keyword search scheme over encrypted cloud data, which simultaneously supports dynamic update operations.

**N. Cao, C. Wang, M. Li, K. Ren, and W. Lou [3]** states that “Privacy-preserving

multi-keyword ranked search over encrypted cloud data” : With the advent of cloud computing, data owners are motivated to outsource their complex data management systems from local sites to the commercial public cloud for great flexibility and economic savings. But for protecting data privacy, sensitive data have to be encrypted before outsourcing, which obsoletes traditional data utilization based on plaintext keyword search. Thus, enabling an encrypted cloud data search service is of paramount importance. Considering the large number of data users and documents in the cloud, it is necessary to allow multiple keywords in the search request and return documents in the order of their relevance to these keywords.

**D. X. Song, D. Wagner, and A. Perrig [4]** states that “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. In this paper, 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.

**R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky [5]** states that “Searchable

symmetric encryption: improved definitions and efficient constructions” : Private-key storage outsourcing [30, 4, 33] allows clients with either limited resources or limited expertise to store and distribute large amounts of symmetrically encrypted data at low cost. Since regular private-key encryption prevents one from searching over encrypted data, clients also lose the ability to selectively retrieve segments of their data. To address this, several techniques have been proposed for provisioning symmetric encryption with search capabilities [40, 23, 10, 18]; the resulting construct is typically called searchable encryption. The area of searchable encryption has been identified by DARPA as one of the technical advances that can be used to balance the need for both privacy and national security in information aggregation systems [1]. One approach to provisioning symmetric encryption with search capabilities is with a so-called secure index [23]. An index is a data structure that stores document collections while supporting efficient keyword search, i.e., given a keyword, the index returns a pointer to the documents that contain it. Informally, an index is “secure” if the search operation for a keyword w can only be performed by users that possess a “trapdoor” for w and if the trapdoor can only be generated with a secret key.

**CHAPTER-3: SYSTEM DESIGN**

**3.1 System Architecture**

View all end users and data owners

, View all cloud files

, Capture all attackers

, View all attackers

, View all key attackers

, View all transactions

, View all search requests

, View File Rank Result

, View Time Delay Results

, View Throughput Results

**CLOUD SERVER**

**Register and Login  
, Upload files  
, View all your uploaded files  
, Verify your secret key  
, Verify your Trapdoor un likability  
, View all search request**

**1) Registers & Logins**

****

**1) Injects False Data**

**Attacker**

**1) Registers & Login**

**End User**

**Register and Login**

**, Request File Search and View Response**

**, Search Files by Multi keyword**

**, Download File**

**Fig-1 Architecture Diagram**

**3.1.a**  **Flow Chart 1 : End User**

**User**

User Register

LOGIN

Yes No

Request File Search

Username & Password Wrong

YES NO

No Response

View Response

Search Files by Multikeyword

**YES**  NO

No data found

View All Corresponding Files

Download File

**Fig-2 Flow Chart 1: End User**

**3.1.b** **Flow Chart 2 : Cloud Server**

**Start**

**Cloud Server**

**Cloud Server**

Cloud Login

YES No

**No Username Exists**

**View all end users and data owners**

View all cloud files

Capture all attackers

**View all attackers**

**View all key attackers**

**View all transactions**

**View all search requests**

**View File Rank Result,** **View Time Delay Results,** **View Throughput Results**

**3.1.c**  **Flow Chart 3 : Data Owner**

**Start**

**Data Owner**

**Register and Login**

Owner Login

YES No

**Upload files**

**No Owner Exists**

View all your uploaded files

Verify your secret key

**Verify your Trapdoor unlinkability**

**View all search request**

**Fig-4 Flow Chart 3: Data Owner**

**3.1.2 Class Diagram**

Cloud Server

Data Owner

End User

Methods

Members

Login, Register, Reset

User Name, Password

Members

Methods

Login

Members

Methods

Members

Methods

,View all end users and data owners ,View all cloud files ,Capture all attackers,View all attackers,View all key attackers,View all transactions,View all search requests,View File Rank Result,View Time Delay Results,View Throughput Results

File Name, Owner Name, Trapdoor,User Name,Secret key,Date and Time,Attacker\_Name,Attacked\_Date

Register and Login,Upload files ,View all your uploaded files ,Verify your secret key ,Verify your Trapdoor unlinkbility ,View all search request

File Name, Owner Name, Trapdoor,User Name,Secret key,Date and Time

Register

Register, Reset

Name, Password, DOB, Gender, Address, City, Country, Email, Mobile

Methods

Members

Register and Login, Request File Search and View Response, Search Files by Multikeyword, Download File

File Name, Owner Name, Trapdoor,User Name,Secret key,Date and Time

**Fig-5 Class Diagram**

**3.1.3 Sequence Diagram**

**End user**

**Cloud server**

**Data Owner**

Browse & Encrypt uploads file

File sent confirmation

Request secret key

Verify your secret key

Gives secret key

View all your uploaded files

Searching file based on Multi Keyword

Verify your Trapdoor unlinkability

View all end users and data owners

View all search request

Request Access Permission

Access permission response

View all cloud files, Capture all attackers, View all attackers

Request SK

Download File

Download file

View all key attackers, View all transactions, View all search requests

Save files

View File Rank Result, View Time Delay Results, View Throughput Results

**Fig-6 Sequence Diagram**

**3.1.4 Data Flow Diagram**

Data Flow diagram

**ATTACKER**

**Data Owner**

**CLOUD   
 SERVER**

**Attacker**

**END USER**

**Fig-7 Data Flow Diagram**

**3.1.5 Use Case Diagram**

**Use Case Diagram**

Encrypt File

Upload Response

**Cloud Server**

Browse File

Store Encrypted Data

asdfa

**Data Owner**

View Attackers

asdfa

View Owner Files

asdfa

View all attackers, View all key attackers

Upload File

View all transactions

**END** **USER**

**Searches for files based on Content’s keyword**

**Requests for Skey**

**Requests for downloading files**

Retrieve and store data

Authorize user

**Fig-8 Use Case Diagram**

**3.1.6 Modules**

* **Data Owner**

In this module, the data provider uploads their encrypted data in the Cloud server. For the security purpose the data owner encrypts the data file and then store in the server. The Data owner can have capable of manipulating the encrypted data file and performs the following operations Browse and enc and Uploads files ,View all your uploaded files, Verify your secret key ,Verify your Trapdoor Unlinkability, View all search request.

* **Cloud Server**

The Cloud server manages which is to provide data storage service for the Data Owners. Data owners encrypt their data files and store them in the Server for sharing with data consumers. To access the shared data files, data consumers download encrypted data files of their interest from the Server and then Server will decrypt them. The server will generate the aggregate key if the end user requests for file authorization to access and performs the following operations such as View all cloud files ,Capture all attackers, View all attackers, View all key attackers, View all transactions, View all search requests, View File Rank Result, View Time Delay Results, View Throughput Results

* **END User**

In this module, the user can only access the data file with the secret key. The user can search the file for a specified keyword. The data which matches for a particular keyword will be indexed in the cloud server and then response to the end user and can do the following operations like Register and Login, Request File Search and View Response, Search Files by Multi-keyword, Download File.

**3.2 System Requirements**

##### **3.2.1 Hardware Requirements**:

* + - * Processor: i3 and above
      * RAM: 4 GB
      * Space on Hard Disk: 20 GB

##### **3.2.2 Software Requirements**:

* + - * Eclipse IDE
      * JDK 1.8
      * SQL YOG
      * MYSQL
      * TOMCAT

##### **Operating Systems Supported:**

* + - * Windows 7
      * Windows 10
      * Windows 11

##### **Technologies and Languages used to Develop:**

* + - * JAVA
      * J2EE (JSP, Servlet)
      * CSS
      * HTML
      * JAVA SCRIPT
      * MySQL

##### **Debugger and Emulator:**

Eclipse

**CHAPTER-4: INPUT AND OUTPUT DESIGN**

**4.1 Input Design**

Input Design plays a vital role in the life cycle of software development, it requires very careful attention of developers. The input design is to feed data to the application as accurate as possible. So inputs are supposed to be designed effectively so that the errors occurring while feeding are minimized. According to Software Engineering Concepts, the input forms or screens are designed to provide to have a validation control over the input limit, range and other related validations.

This system has input screens in almost all the modules. Error messages are developed to alert the user whenever he commits some mistakes and guides him in the right way so that invalid entries are not made. Let us see deeply about this under module design.

Input design is the process of converting the user created input into a computer-based format. The goal of the input design is to make the data entry logical and free from errors. The error is in the input are controlled by the input design. The application has been developed in user-friendly manner. The forms have been designed in such a way during the processing the cursor is placed in the position where must be entered. The user is also provided with in an option to select an appropriate input from various alternatives related to the field in certain cases.

Validations are required for each data entered. Whenever a user enters an erroneous data, error message is displayed and the user can move on to the subsequent pages after completing all the entries in the current page.

**4.2 Output Design**

The Output from the computer is required to mainly create an efficient method of communication within the company primarily among the project leader and his team members, in other words, the administrator and the clients. The output of VPN is the system which allows the project leader to manage his clients in terms of creating new clients and assigning new projects to them, maintaining a record of the project validity and providing folder level access to each client on the user side depending on the projects allotted to him. After completion of a project, a new project may be assigned to the client. User authentication procedures are maintained at the initial stages itself. A new user may be created by the administrator himself or a user can himself register as a new user but the task of assigning projects and validating a new user rests with the administrator only.

The application starts running when it is executed for the first time. The server has to be started and then the internet explorer in used as the browser. The project will run on the local area network so the server machine will serve as the administrator while the other connected systems can act as the clients. The developed system is highly user friendly and can be easily understood by anyone using it even for the first time.

**CHAPTER-5: SYSTEM ENVIRONMENT**

### 5.1 Java Technology:

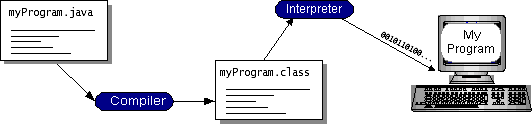
Java technology is both a programming language and a platform.

### 5.1.1 The Java Programming Language:

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

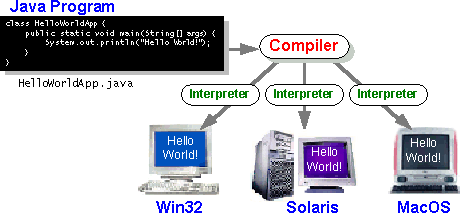
* + - * Simple
      * Architecture neutral
      * Object oriented
      * Portable
      * Distributed
      * High performance
      * Interpreted
      * Multithreaded
      * Robust
      * Dynamic
      * Secure

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.



##### Fig-7 Execution process

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.



**Fig-8 JVM**

### 5.1.2 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

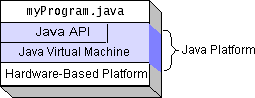
operating system and hardware. The Java platform differs from most other platforms in that it’s a software-only platform that runs on top of other hardware-based platforms.

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.

##### Fig-9 Java Platform

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.

**5.1.3 Java Database Connectivity**

**What Is JDBC?**

JDBC is a Java API for executing SQL statements. (As a point of interest, JDBC is a trademarked name and is not an acronym; nevertheless, JDBC is often thought of as standing for Java Database Connectivity. It consists of a set of classes and interfaces written in the Java programming language. JDBC provides a standard API for tool/database developers and makes it possible to write database applications using a pure Java API.

Using JDBC, it is easy to send SQL statements to virtually any relational database. One can write a single program using the JDBC API, and the program will be able to send SQL statements to the appropriate database. The combinations of Java and JDBC lets a programmer write it once and run it anywhere.

What Does JDBC Do?

Simply put, JDBC makes it possible to do three things:

* Establish a connection with a database
* Send SQL statements
* Process the results.

**JDBC versus ODBC and other APIs**

At this point, Microsoft's ODBC (Open Database Connectivity) API is that probably the most widely used programming interface for accessing relational databases. It offers the ability to connect to almost all databases on almost all platforms.

So why not just use ODBC from Java? The answer is that you can use ODBC from Java, but this is best done with the help of JDBC in the form of the JDBC-ODBC Bridge, which we will cover shortly. The question now becomes "Why do you need JDBC?" There are several answers to this question:

1. ODBC is not appropriate for direct use from Java because it uses a C interface. Calls from Java to native C code have a number of drawbacks in the security, implementation, robustness, and automatic portability of applications.
2. A literal translation of the ODBC C API into a Java API would not be desirable. For example, Java has no pointers, and ODBC makes copious use of them, including the notoriously error-prone generic pointer "void \*". You can think of JDBC as ODBC translated into an object-oriented interface that is natural for Java programmers.
3. ODBC is hard to learn. It mixes simple and advanced features together, and it has complex options even for simple queries. JDBC, on the other hand, was designed to keep simple things simple while allowing more advanced capabilities where required.
4. A Java API like JDBC is needed in order to enable a "pure Java" solution. When ODBC is used, the ODBC driver manager and drivers must be manually installed on every client machine. When the JDBC driver is written completely in Java, however, JDBC code is automatically installable, portable, and secure on all Java platforms from network computers to mainframes.

**Two-tier and Three-tier Models**

The JDBC API supports both two-tier and three-tier models for database access.

In the two-tier model, a Java applet or application talks directly to the database. This requires a JDBC driver that can communicate with the particular database management system being accessed. A user's SQL statements are delivered to the database, and the results of those statements are sent back to the user. The database may be located on another machine to which the user is connected via a network. This is referred to as a client/server configuration, with the user's machine as the client, and the machine housing the database as the server. The network can be an Intranet, which, for example, connects employees within a corporation, or it can be the Internet.

In the three-tier model, commands are sent to a "middle tier" of services, which then send SQL statements to the database. The database processes the SQL statements and sends the results back to the middle tier, which then sends them to the user. MIS directors find the three-tier model very attractive because the middle tier makes it possible to maintain control over access and the kinds of updates that can be made to corporate data. Another advantage is that when there is a middle tier, the user can employ an easy-to-use higher-level API which is translated by the middle tier into the appropriate low-level calls. Finally, in many cases the three-tier architecture can provide performance advantages.

**Java applet or**

**Html browser**

**Application**

**Server (Java)**

**JDBC**

## **DBMS**

**Client machine (GUI)**

**HTTP, RMI, or CORBA calls**

**Server machine (business Logic)**

**DBMS-proprietary protocol**

**Database server**

**JAVA**

**Application**

### JDBC

### DBMS

**Client machine**

**DBMS-proprietary protocol**

**Database server**

Until now the middle tier has typically been written in languages such as C or C++, which offer fast performance. However, with the introduction of optimizing compilers that translate Java byte code into efficient machine-specific code, it is becoming practical to implement the middle tier in Java. This is a big plus, making it possible to take advantage of Java's robustness, multithreading, and security features. JDBC is important to allow database access from a Java middle tier.

**JDBC Driver Types**

The JDBC drivers that we are aware of at this time fit into one of four categories:

* JDBC-ODBC bridge plus ODBC driver
* Native-API partly-Java driver
* JDBC-Net pure Java driver
* Native-protocol pure Java driver

**JDBC-ODBC Bridge**

If possible, use a Pure Java JDBC driver instead of the Bridge and an ODBC driver. This completely eliminates the client configuration required by ODBC. It also eliminates the potential that the Java VM could be corrupted by an error in the native code brought in by the Bridge (that is, the Bridge native library, the ODBC driver manager library, the ODBC driver library, and the database client library).

**What Is the JDBC- ODBC Bridge?**

The JDBC-ODBC Bridge is a JDBC driver, which implements JDBC operations by translating them into ODBC operations. To ODBC it appears as a normal application program. The Bridge implements JDBC for any database for which an ODBC driver is available. The Bridge is implemented as the

sun.jdbc.odbc Java package and contains a native library used to access ODBC. The Bridge is a joint development of Intersolv and JavaSoft.

**5.1.4 Java Server Pages (JSP)**

Java server Pages is a simple, yet powerful technology for creating and maintaining dynamic-content web pages. Based on the Java programming language, Java Server Pages offers proven portability, open standards, and a mature re-usable component model .The Java Server Pages architecture enables the separation of content generation from content presentation. This separation not eases maintenance headaches, it also allows web team members to focus on their areas of expertise. Now, web page designer can concentrate on layout, and web application designers on programming, with minimal concern about impacting each other’s work.

**Features of JSP**

**Portability:**

Java Server Pages files can be run on any web server or web-enabled application server that provides support for them. Dubbed the JSP engine, this support involves recognition, translation, and management of the Java Server Page lifecycle and its interaction components.

**Components**

It was mentioned earlier that the Java Server Pages architecture can include reusable Java components. The architecture also allows for the embedding of a scripting language directly into the Java Server Pages file. The components current supported include Java Beans, and Servlets.

**Processing**

A Java Server Pages file is essentially an HTML document with JSP scripting or tags. The Java Server Pages file has a JSP extension to the server as a Java Server Pages file. Before the page is served, the Java Server Pages syntax is parsed and processed into a Servlet on the server side. The Servlet that is generated outputs real content in straight HTML for responding to the client.

**Access Models:**

A Java Server Pages file may be accessed in at least two different ways. A client’s request comes directly into a Java Server Page. In this scenario, suppose the page accesses reusable Java Bean components that perform particular well-defined computations like accessing a database. The result of the Beans computations, called result sets is stored within the Bean as properties. The page uses such Beans to generate dynamic content and present it back to the client.

In both of the above cases, the page could also contain any valid Java code. Java Server Pages architecture encourages separation of content from presentation.

**Steps in the execution of a JSP Application:**

1. The client sends a request to the web server for a JSP file by giving the name of the JSP file within the form tag of a HTML page.
2. This request is transferred to the JavaWebServer. At the server side JavaWebServer receives the request and if it is a request for a jsp file server gives this request to the JSP engine.
3. JSP engine is program which can understands the tags of the jsp and then it converts those tags into a Servlet program and it is stored at the server side. This Servlet is loaded in the memory and then it is executed and the result is given back to the JavaWebServer and then it is transferred back to the result is given back to the JavaWebServer and then it is transferred back to the client.

**5.1.5 Hyper Text Markup Language**

Hypertext Markup Language (HTML), the languages of the World Wide Web (WWW), allows users

to produces Web pages that include text, graphics and pointer to other Web pages (Hyperlinks).

HTML is not a programming language but it is an application of ISO Standard 8879, SGML (Standard Generalized Markup Language), but specialized to hypertext and adapted to the Web. The idea behind Hypertext is that instead of reading text in rigid linear structure, we can easily jump from one point to another point. We can navigate through the information based on our interest and preference. A markup language is simply a series of elements, each delimited with special characters that define how text or other items enclosed within the elements should be displayed. Hyperlinks are underlined or emphasized works that load to other documents or some portions of the same document.

HTML can be used to display any type of document on the host computer, which can be geographically at a different location. It is a versatile language and can be used on any platform or desktop.

HTML provides tags (special codes) to make the document look attractive. HTML tags are not case-sensitive. Using graphics, fonts, different sizes, color, etc., can enhance the presentation of the document. Anything that is not a tag is part of the document itself.

**Basic HTML Tags :**

**<!-- -->** Specifies comments

**<A>……….</A>** Creates hypertext links

**<B>……….</B>** Formats text as bold

**<BIG>……….</BIG>** Formats text in large font.

**<BODY>…</BODY>** Contains all tags and text in the HTML document

**<CENTER>...</CENTER>** Creates text

**<DD>…</DD>** Definition of a term

**<DL>...</DL>**  Creates definition list

**<FONT>…</FONT>** Formats text with a particular font

**<FORM>...</FORM>** Encloses a fill-out form

**<FRAME>...</FRAME>** Defines a particular frame in a set of frames

**<H#>…</H#>** Creates headings of different levels

**<HEAD>...</HEAD>** Contains tags that specify information about a document

**<HR>...</HR>** Creates a horizontal rule

**<HTML>…</HTML>** Contains all other HTML tags

**<META>...</META>** Provides meta-information about a document

**<SCRIPT>…</SCRIPT>** Contains client-side or server-side script

**<TABLE>…</TABLE>**  Creates a table

**<TD>…</TD>** Indicates table data in a table

**<TR>…</TR>** Designates a table row

**<TH>…</TH>** Creates a heading in a table

**ADVANTAGES**

* A HTML document is small and hence easy to send over the net. It is small because it does not include formatted information.
* HTML is platform independent.
* HTML tags are not case-sensitive.

**5.1.6 JavaScript**

JavaScript is a script-based programming language that was developed by Netscape

Communication Corporation. JavaScript was originally called Live Script and renamed as

JavaScript to indicate its relationship with Java. JavaScript supports the development of

both client and server components of Web-based applications. On the client side, it can be

used to write programs that are executed by a Web browser within the context of a Web

page. On the server side, it can be used to write Web server programs that can process

information submitted by a Web browser and then updates the browser’s display

accordingly

Even though JavaScript supports both client and server Web programming, we prefer JavaScript at Client side programming since most of the browsers supports it. JavaScript is almost as easy to learn as HTML, and JavaScript statements can be included in HTML documents by enclosing the statements between a pair of scripting tags

<SCRIPTS>..</SCRIPT>.

<SCRIPT LANGUAGE = “JavaScript”>

JavaScript statements

</SCRIPT>

Here are a few things we can do with JavaScript:

* Validate the contents of a form and make calculations.
* Add scrolling or changing messages to the Browser’s status line.
* Animate images or rotate images that change when we move the mouse over them.
* Detect the browser in use and display different content for different browsers.
* Detect installed plug-ins and notify the user if a plug-in is required.

We can do much more with JavaScript, including creating entire application.

JavaScript vs Java:

JavaScript and Java are entirely different languages. A few of the most glaring differences are:

* Java applets are generally displayed in a box within the web document; JavaScript can affect any part of the Web document itself.
* While JavaScript is best suited to simple applications and adding interactive features to Web pages; Java can be used for incredibly complex applications.

There are many other differences but the important thing to remember is that JavaScript and Java are separate languages. They are both useful for different things; in fact they can be used together to combine their advantages.

Advantages:

* JavaScript can be used for Sever-side and Client-side scripting.
* It is more flexible than VBScript.
* JavaScript is the default scripting languages at Client-side since all the browsers supports it.

**5.1.7 Client Server**

#### **Over view:**

With the varied topic in existence in the fields of computers, Client Server is one, which has generated more heat than light, and also more hype than reality. This technology has acquired a certain critical mass attention with its dedication conferences and magazines. Major computer vendors such as IBM and DEC, have declared that Client Servers is their main future market. A survey of DBMS magazine revealed that 76% of its readers were actively looking at the client server solution. The growth in the client server development tools from $200 million in 1992 to more than $1.2 billion in 1996.

Client server implementations are complex but the underlying concept is simple and powerful. A client is an application running with local resources but able to request the database and relate the services from separate remote server. The software mediating this client server interaction is often referred to as MIDDLEWARE.

The typical client either a PC or a Work Station connected through a network to a more powerful PC, Workstation, Midrange or Main Frames server usually capable of handling request from more than one client. However, with some configuration server may also act as client. A server may need to access other server in order to process the original client request.

The key client server idea is that client as user is essentially insulated from the physical location and formats of the data needs for their application. With the proper middleware, a client input from or report can transparently access and manipulate both local database on the client machine and remote databases on one or more servers. An added bonus is the client server opens the door to multi-vendor database access indulging heterogeneous table joins.

#### **What is a Client Server**

Two prominent systems in existence are client server and file server systems. It is essential to distinguish between client servers and file server systems. Both provide shared network access to data but the comparison dens there! The file server simply provides a remote disk drive that can be accessed by LAN applications on a file by file basis. The client server offers full relational database services such as SQL-Access, Record modifying, Insert, Delete with full relational integrity backup/ restore performance for high volume of transactions, etc. the client server middleware provides a flexible interface between client and server, who does what, when and to whom.

#### **Why Client Server**

Client server has evolved to solve a problem that has been around since the earliest days of computing: how best to distribute your computing, data generation and data storage resources in order to obtain efficient, cost effective departmental an enterprise wide data processing. During mainframe era choices were quite limited. A central machine housed both the CPU and DATA (cards, tapes, drums and later disks). Access to these resources was initially confined to batched runs that produced departmental reports at the appropriate intervals. A strong central information service department ruled the corporation. The role of the rest of the corporation limited to requesting new or more frequent reports and to provide hand written forms from which the central data banks were created and updated. The earliest client server solutions therefore could best be characterized as “SLAVE-MASTER”.Time-sharing changed the picture. Remote terminal could view and even change the central data, subject to access permissions. And, as the central data banks evolved in to sophisticated relational database with non-programmer query languages, online users could formulate adhoc queries and produce local reports without adding to the MIS applications software backlog. However remote access was through dumb terminals, and the client server remained subordinate to the Slave\Master.

# Tomcat 9.0 web server

Tomcat is an open source web server developed by Apache Group. Apache Tomcat is the servlet container that is used in the official Reference Implementation for the Java Servlet and Java Server Pages technologies. The Java Servlet and Java Server Pages specifications are developed by Sun under the Java Community Process. Web Servers like Apache Tomcat support only web components while an application server supports web components as well as business components (BEAs Web logic, is one of the popular application server).To develop a web application with jsp/servlet install any web server like JRun, Tomcat etc. to run your application.



**CHAPTER-6: SYSTEM STUDY**

**6.1 FEASIBILITY STUDY**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

* ECONOMICAL FEASIBILITY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

**6.1.1 ECONOMICAL FEASIBILITY**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

### 6.1.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

**6.1.3 SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

**CHAPTER-7: SYSTEM TESTING**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

### 7.1 TESTING METHODOLOGIES

The following are the Testing Methodologies:

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

**7.1.1 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.

**7.1.2 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.

**7.1.3 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.

**7.1.4 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.

**7.1.5 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 and 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.

**7.2 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.

**7.3** **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.

**7.4 TESTING STRATEGY:**

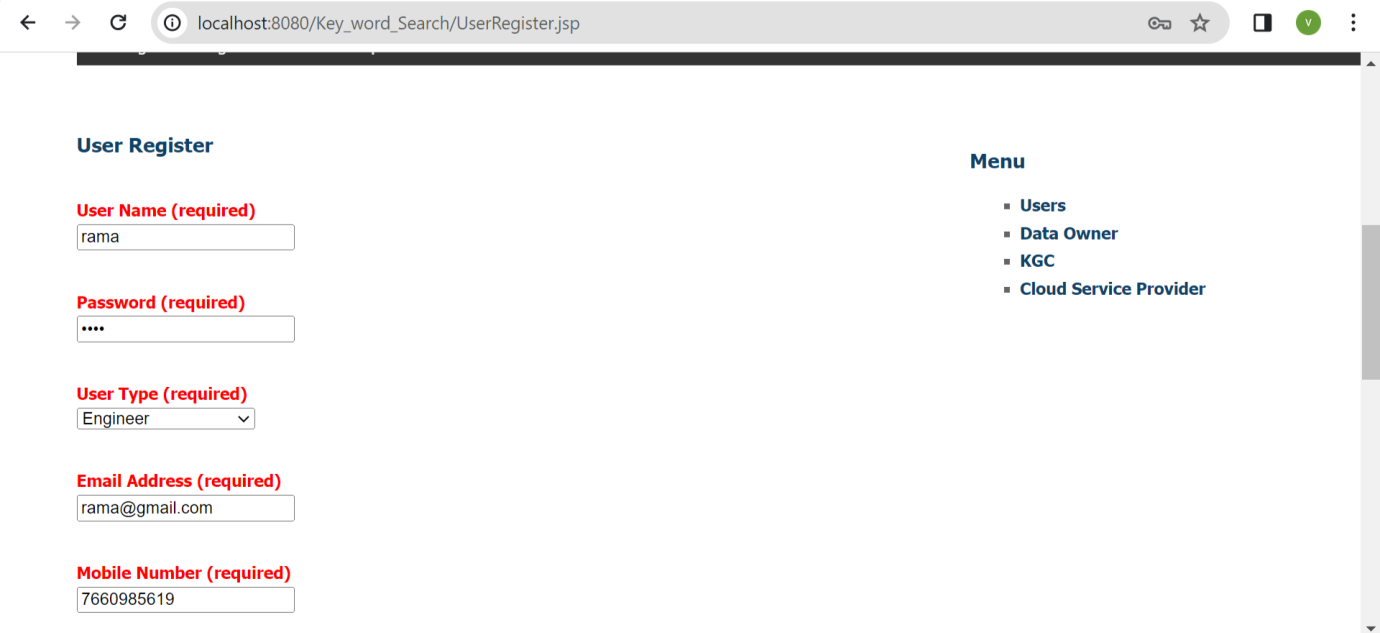
A strategy for system testing integrates system test cases and design techniques into a Well planned series of steps that results in the successful construction of software. The testing strategy must co-operate test planning, test case design, test execution, and the resultant data collection and evaluation .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.

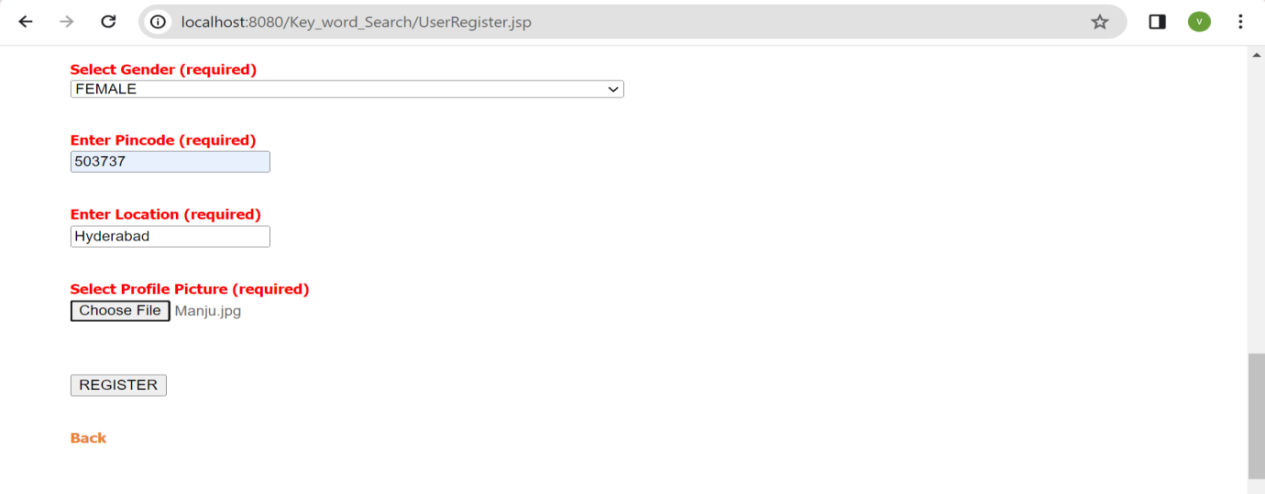
**CHAPTER-8: RESULTS**



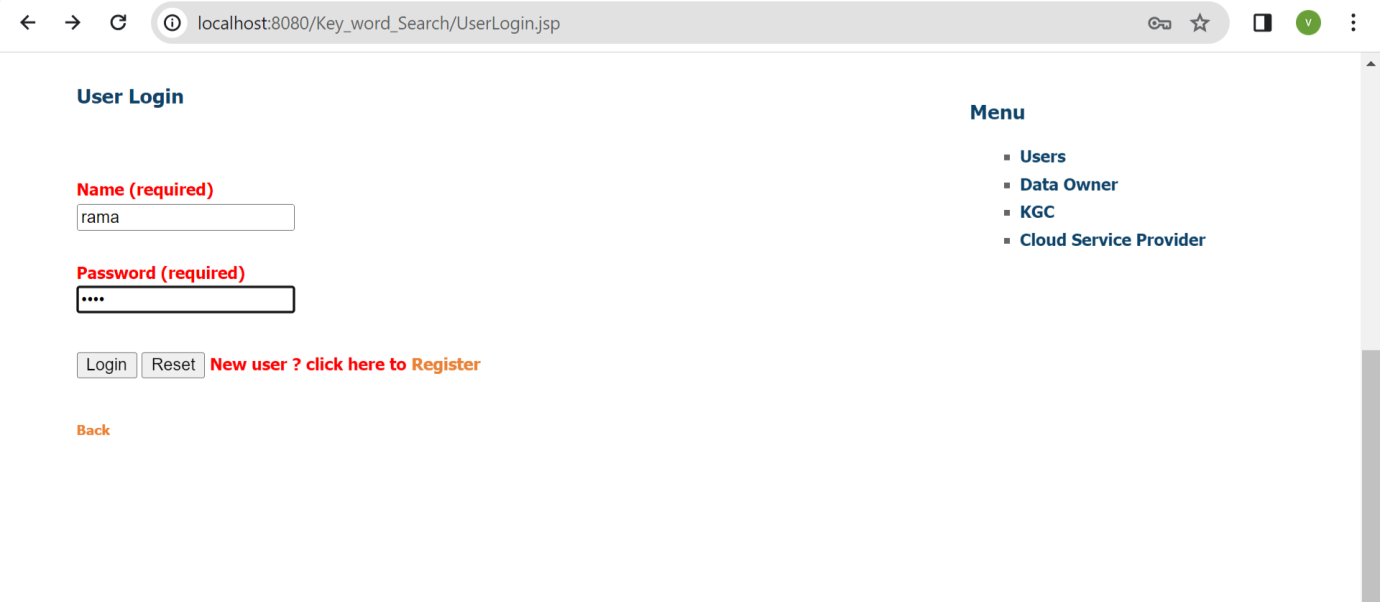
**Fig 8.1: Screenshot of Result (1)**: Home page.



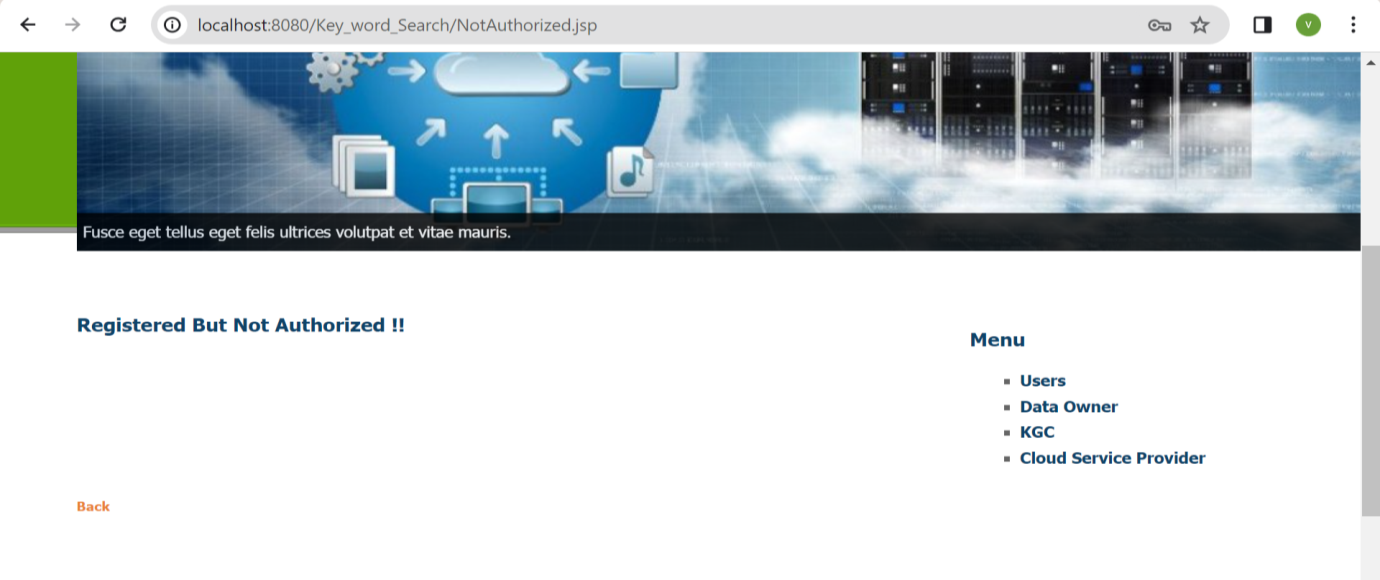
**Fig 8.2: Screenshot of Result (2)**: User registration.



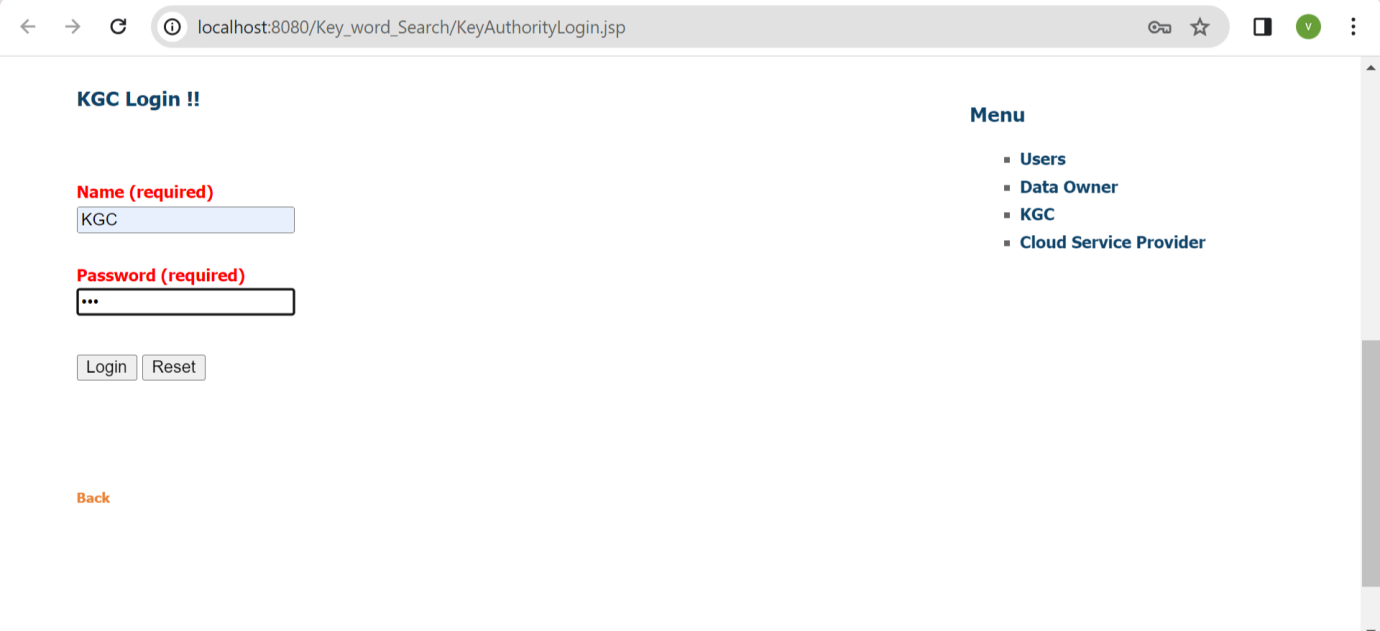
**Fig 8.3: Screenshot of Result (3):** User registration.

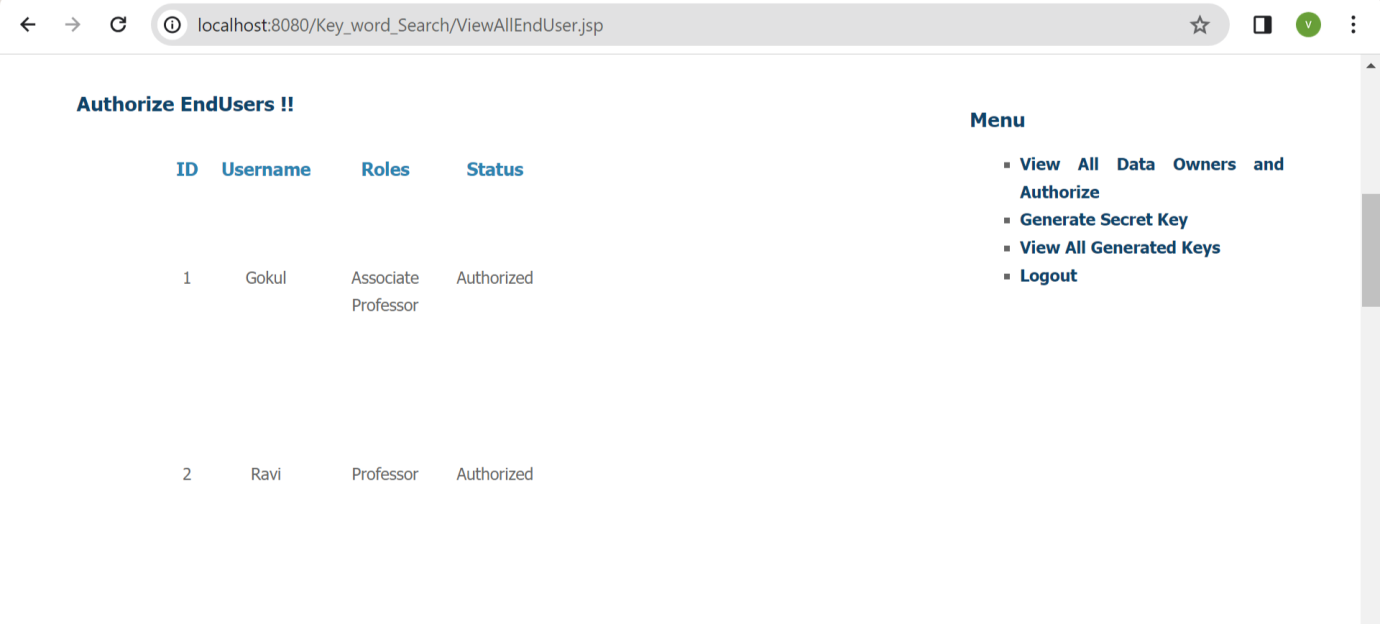


**Fig 8.4: Screenshot of Result (4)**: User login.

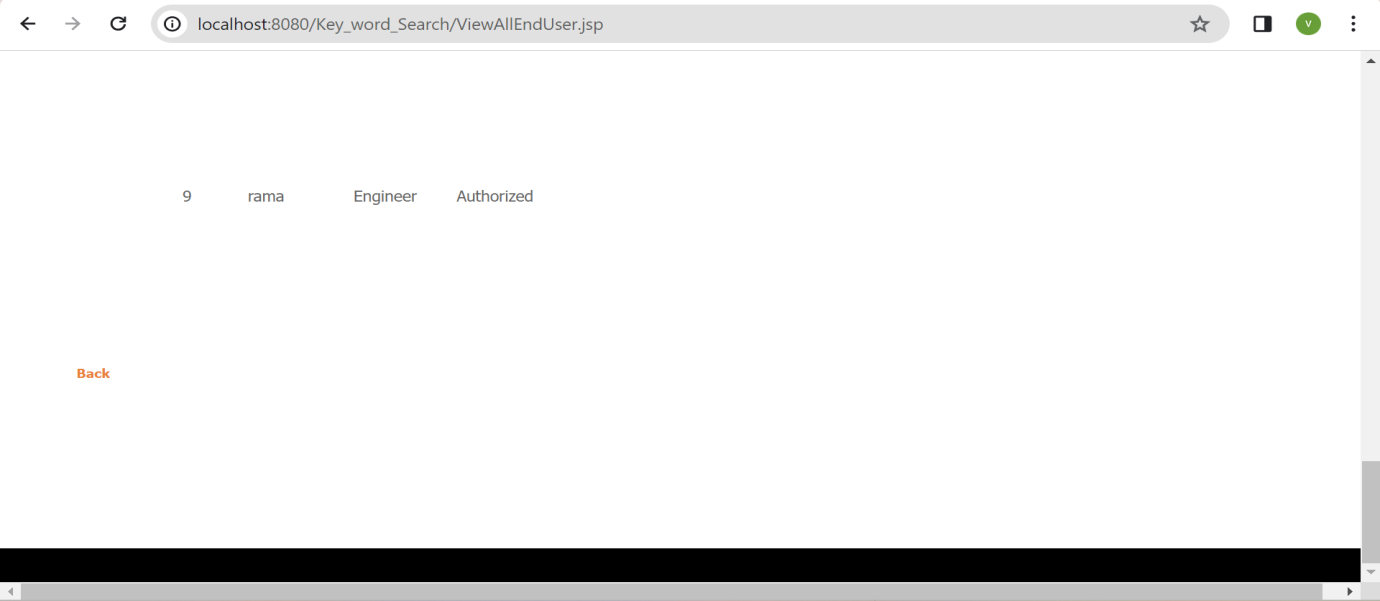


**Fig 8.5: Screenshot of Result (5)**: User registered but not authorized, KGC authorizes the end users.

 **Fig 8.6: Screenshot of Result (6)**: KGC Login details.



**Fig 8.7: Screenshot of Result (7)**: Authorization of End user.



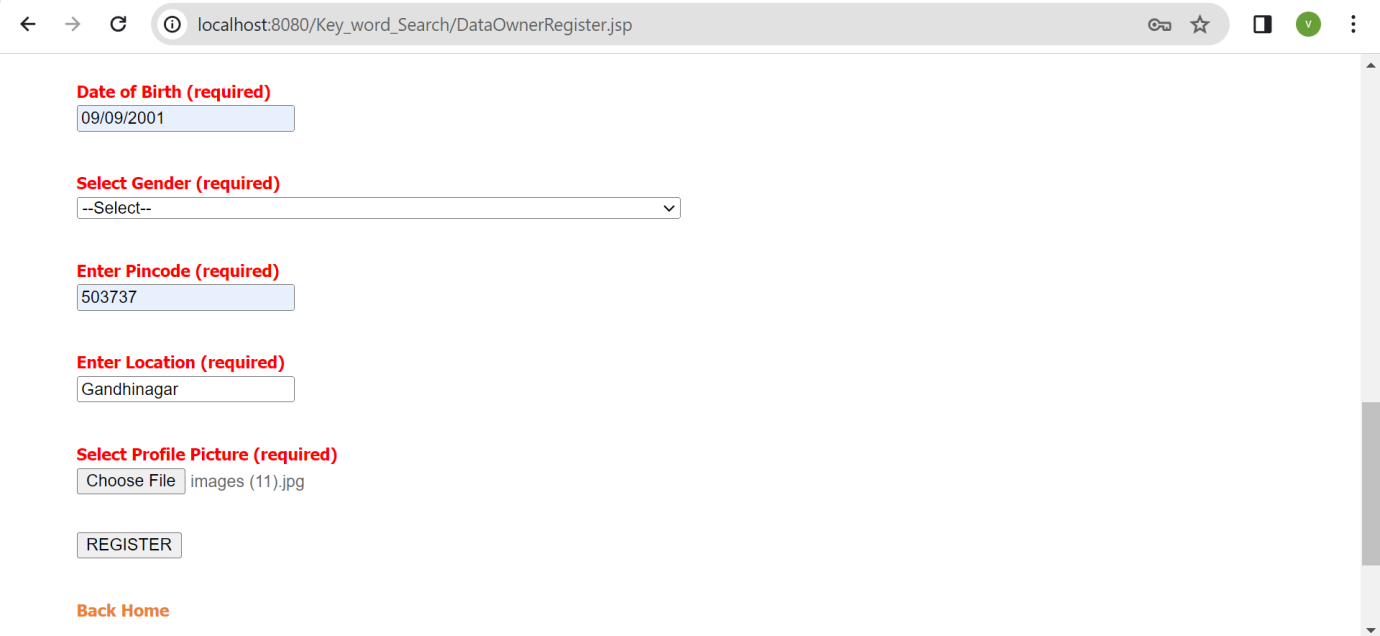
**Fig 8.8: Screenshot of Result (8)**: Authorization of End user.



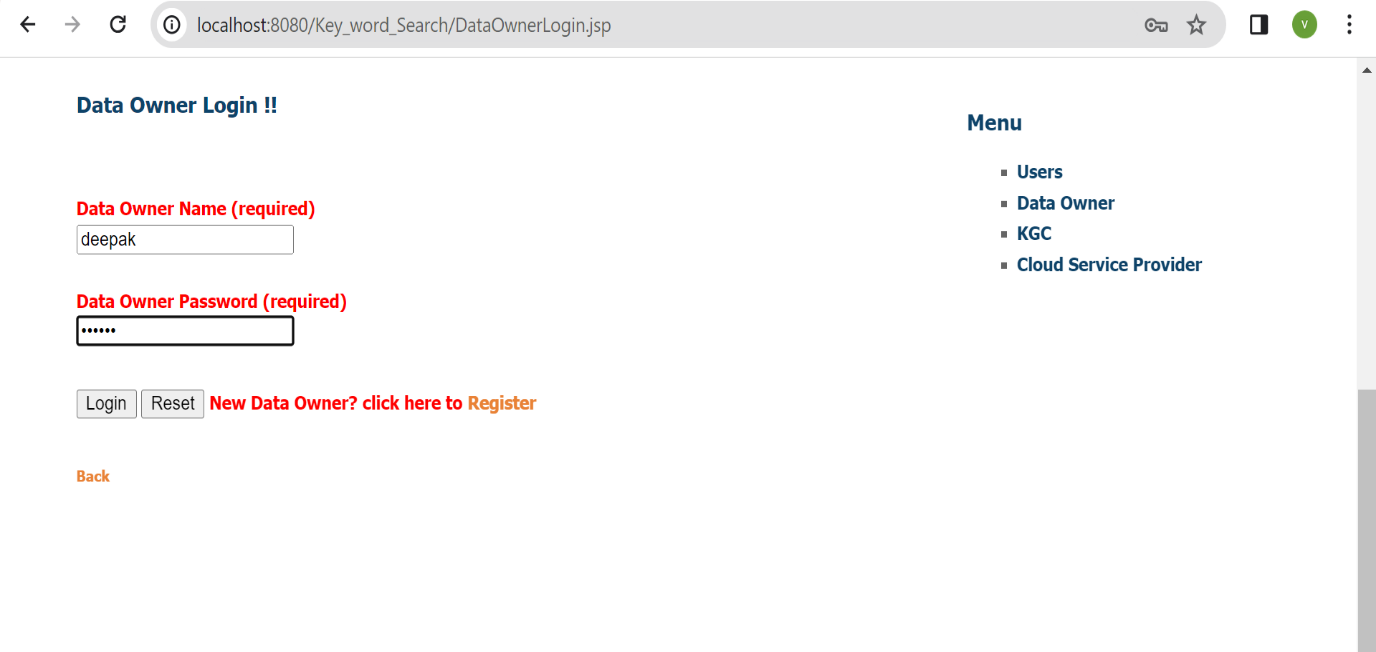
**Fig 8.9: Screenshot of Result (9)**: User home page.



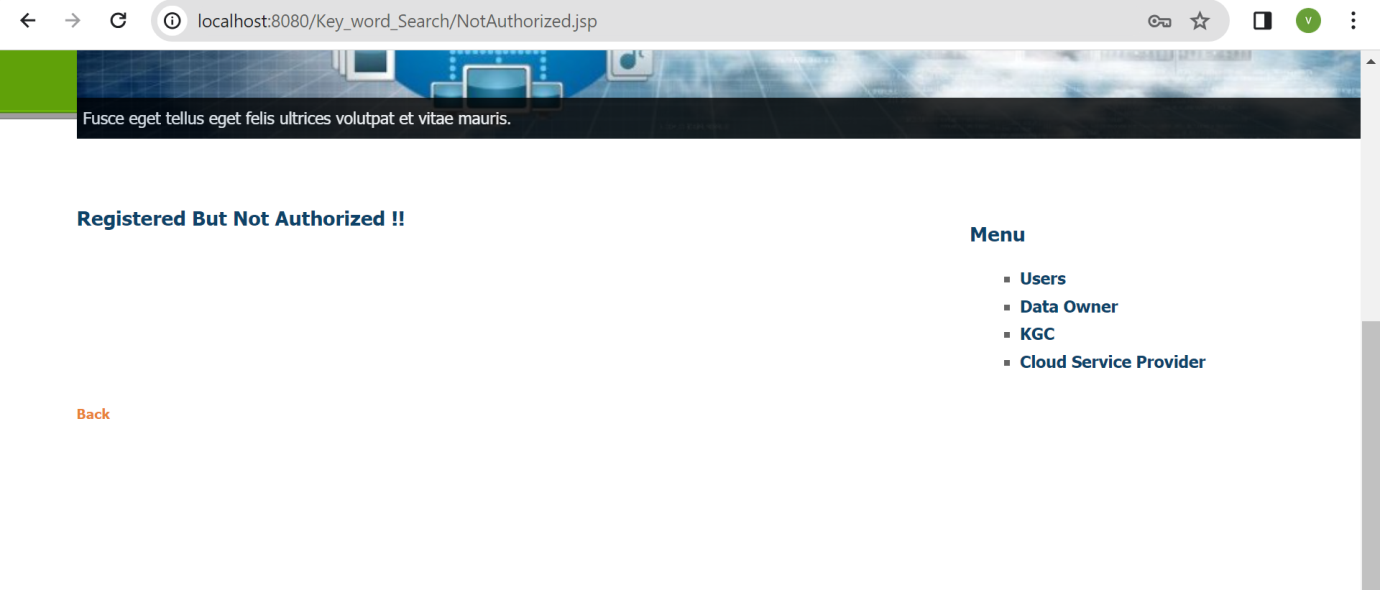
**Fig 8.10: Screenshot of Result (10)**: Data owner registration.

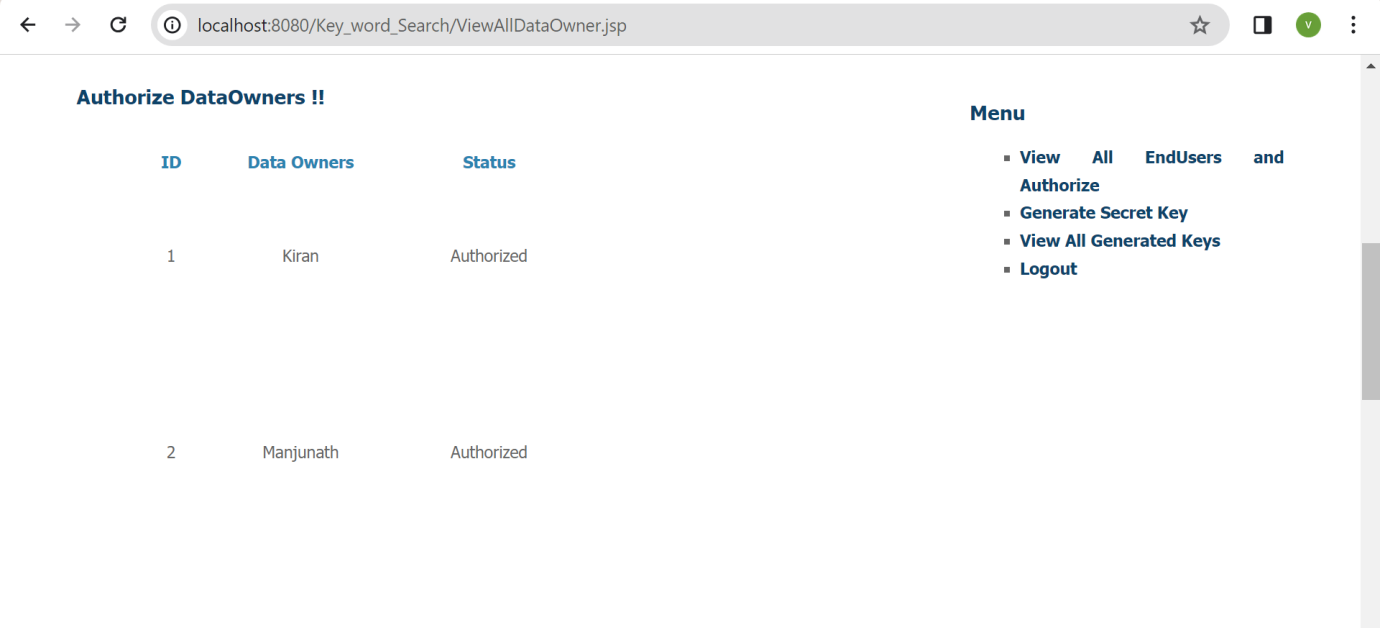


**Fig 8.11: Screenshot of Result (11)**: Data owner registration.

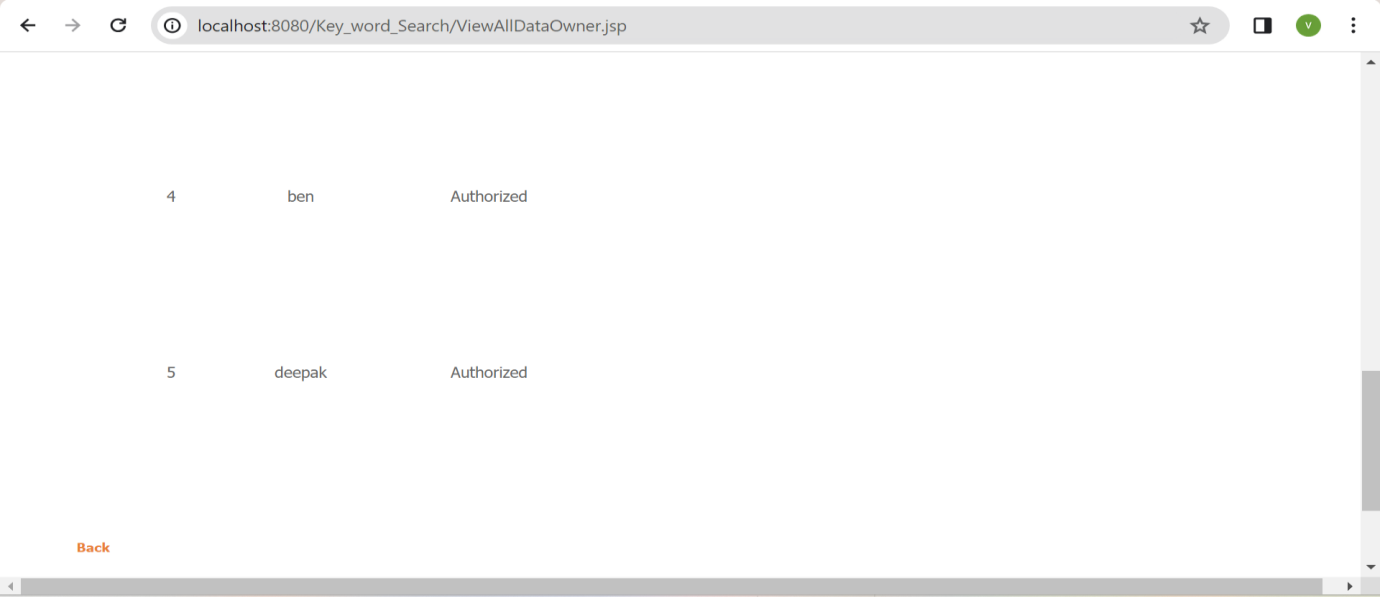


**Fig 8.12: Screenshot of Result (12)**: Data owner login.

**Fig 8.13: Screenshot of Result (13)**: Data owner registered but not authorized, KGC authorizes the data owner.



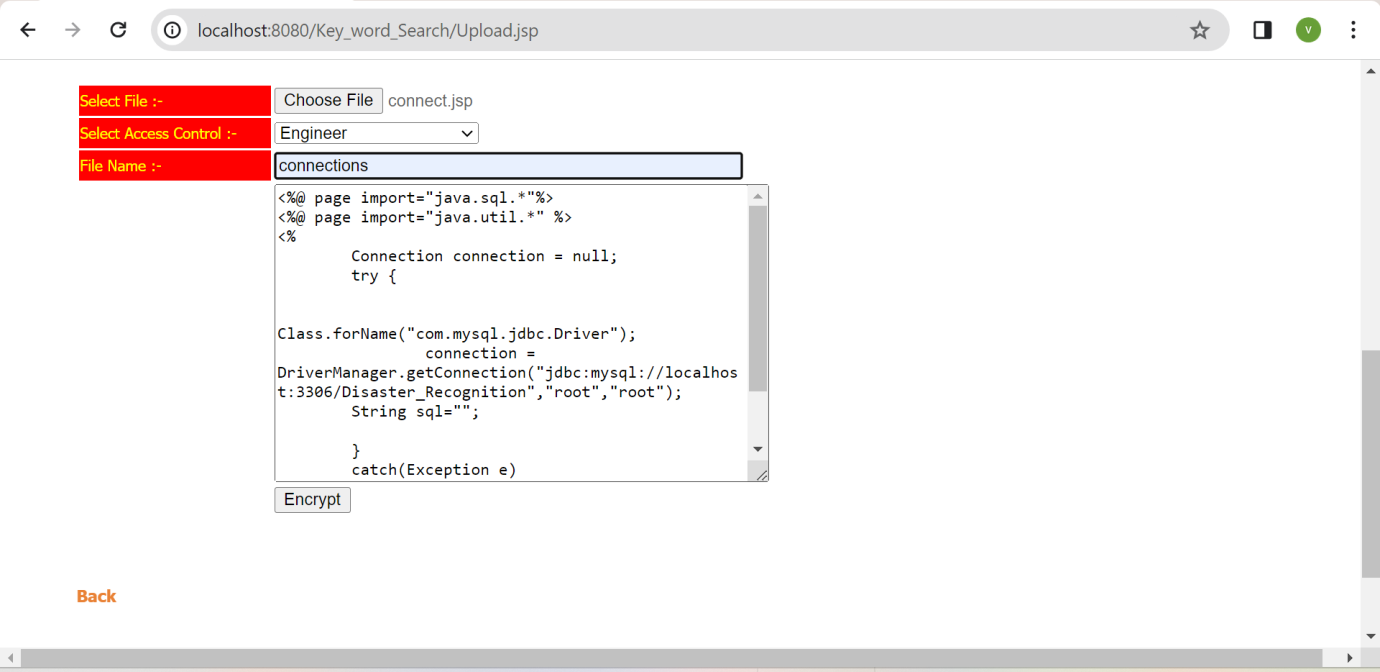
**Fig 8.14: Screenshot of Result (14)**: Data owner authorization.



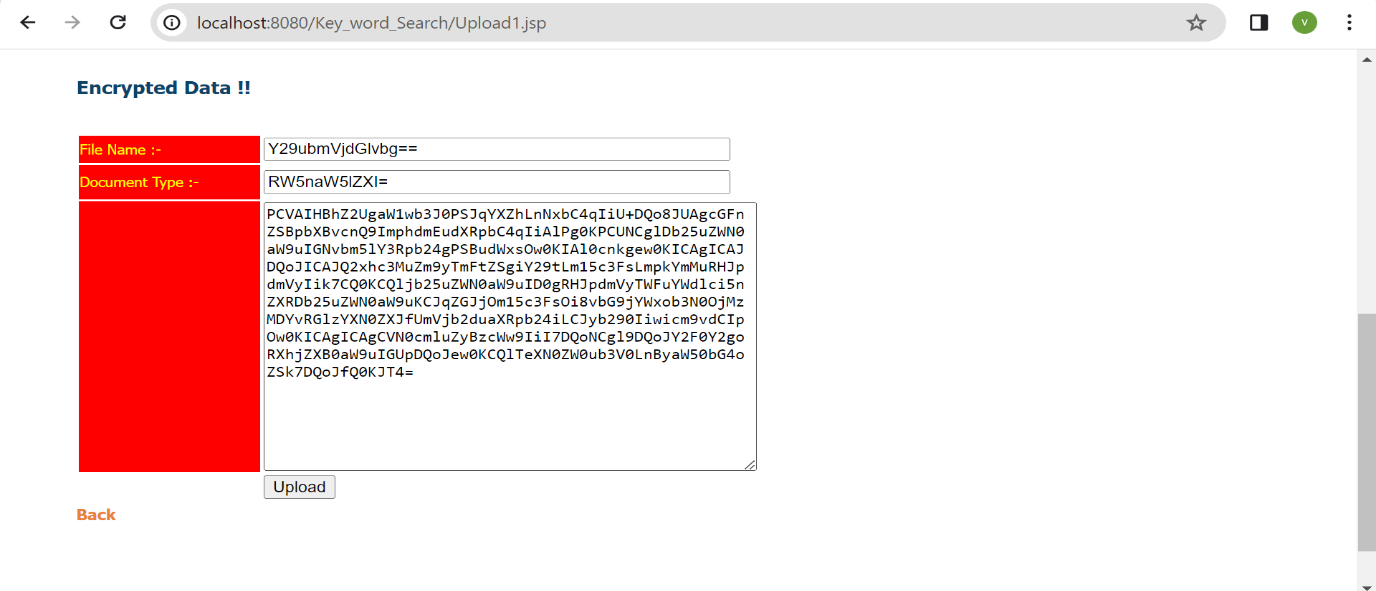
**Fig 8.15: Screenshot of Result (15)**: Data owner authorization.



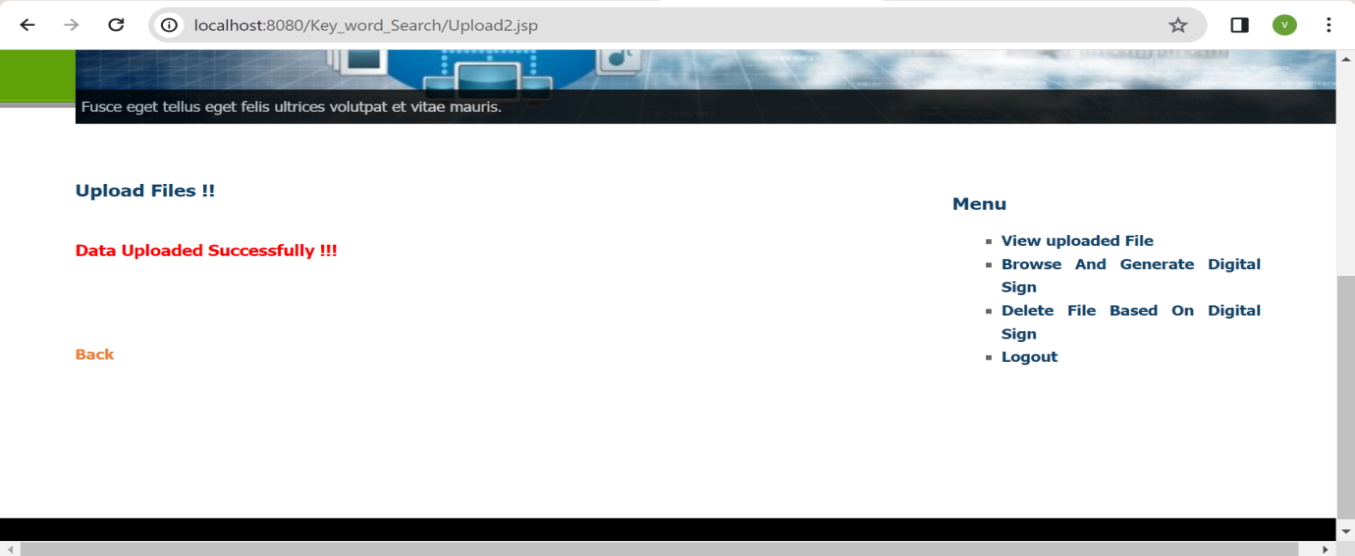
**Fig 8.16: Screenshot of Result (16)**: Data owner home page.



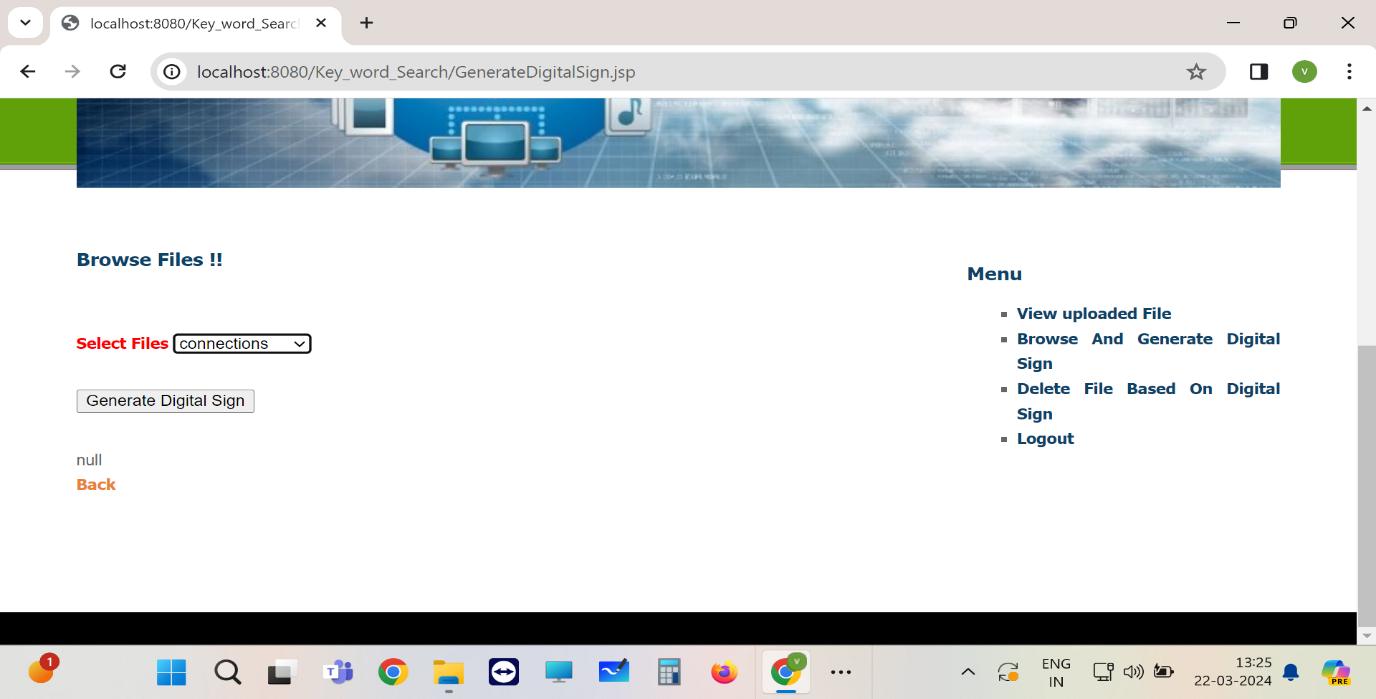
**Fig 8.17: Screenshot of Result (17)**: Data owner giving access control and encrypts the data.



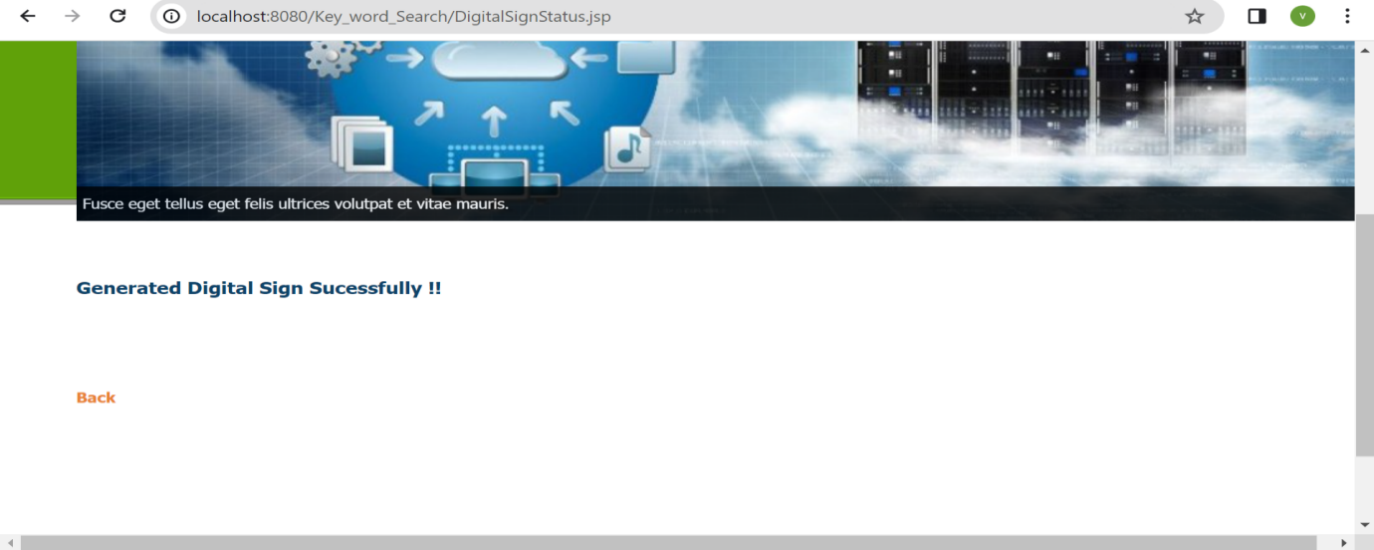
**Fig 8.18: Screenshot of Result (18)**: Encrypted data is uploaded to the cloud.



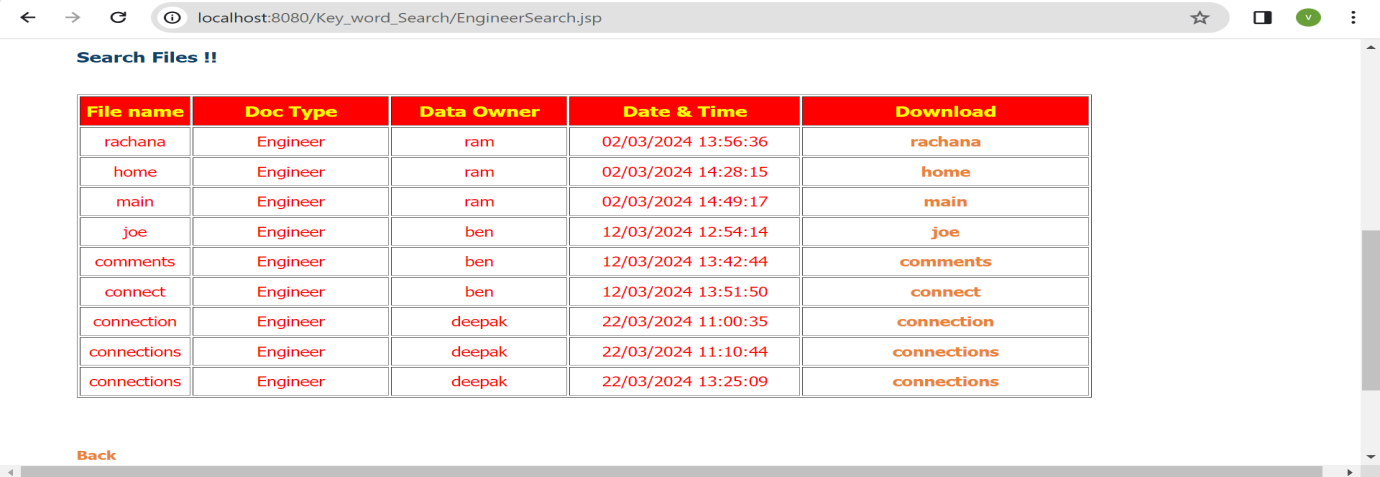
**Fig 8.19: Screenshot of Result (19)**: Data uploaded successfully.



**Fig 8.20: Screenshot of Result (20)**: Browse and generate digital sign.

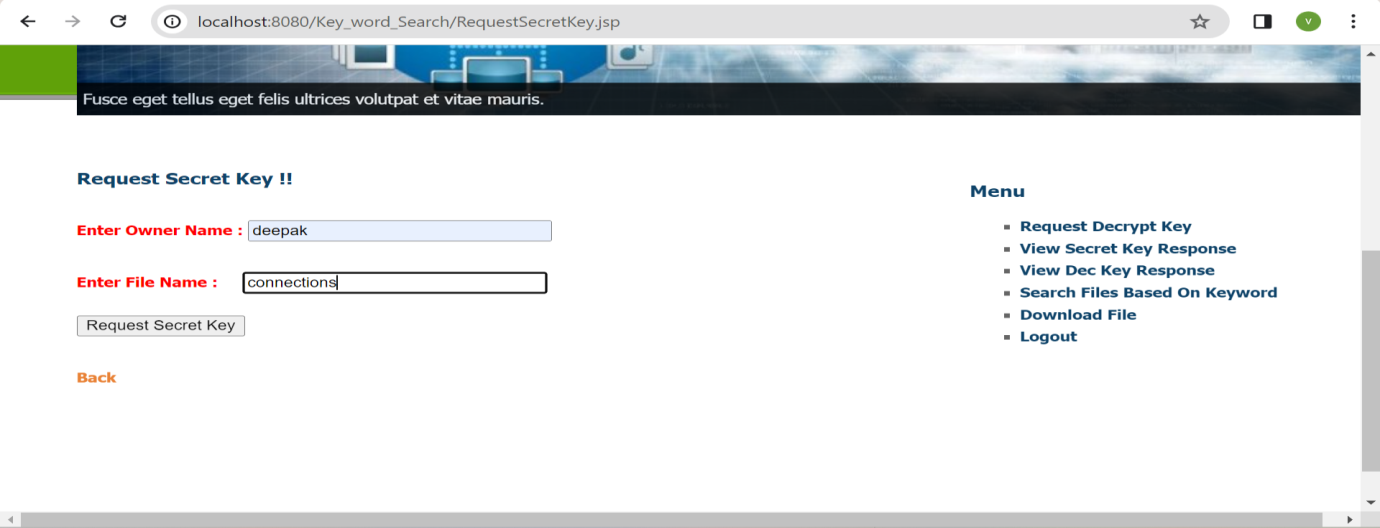


**Fig 8.21: Screenshot of Result (21)**: Generated digital sign successfully.

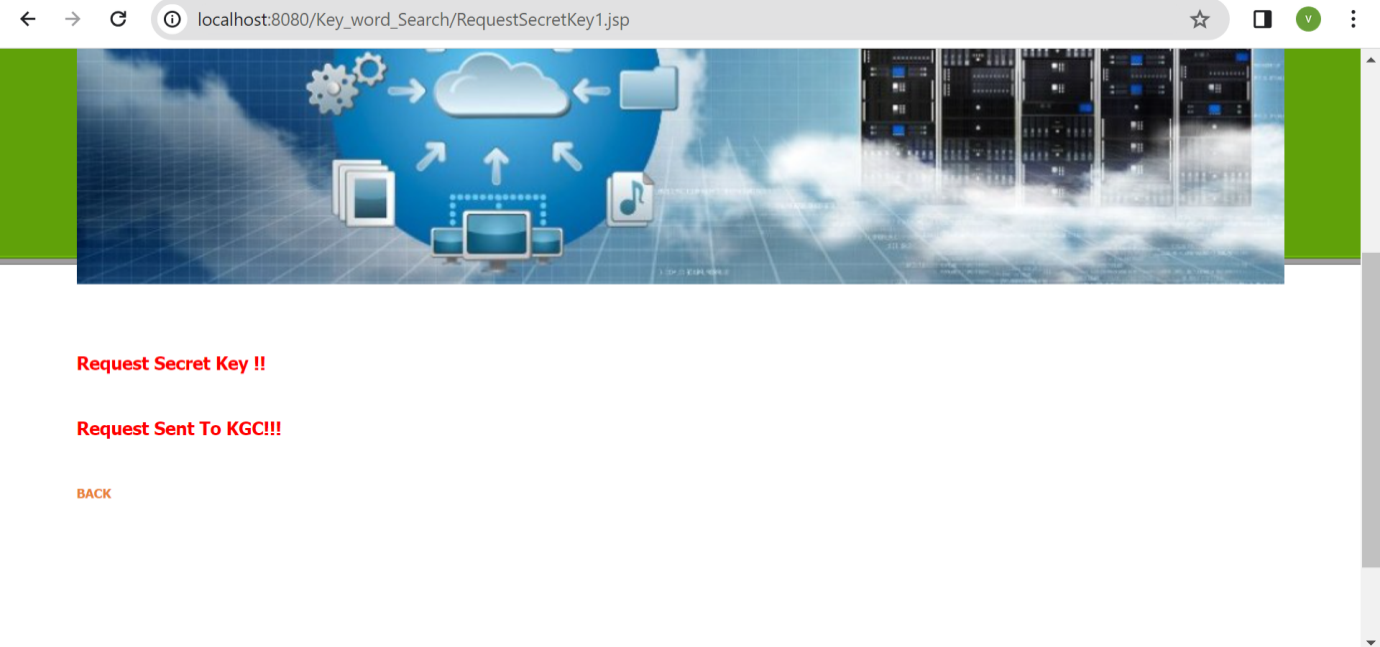
****

**Fig 8.22: Screenshot of Result (22)**: Search file based on keywords, this figure shows

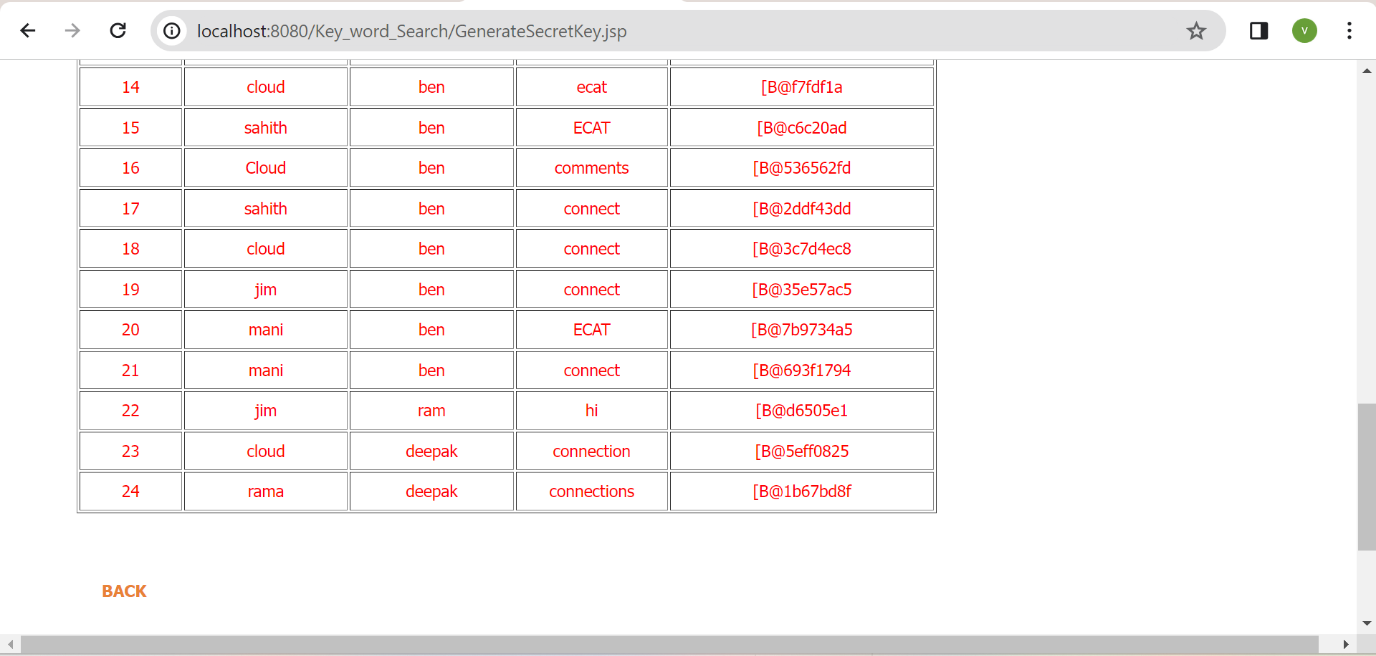
Search files.



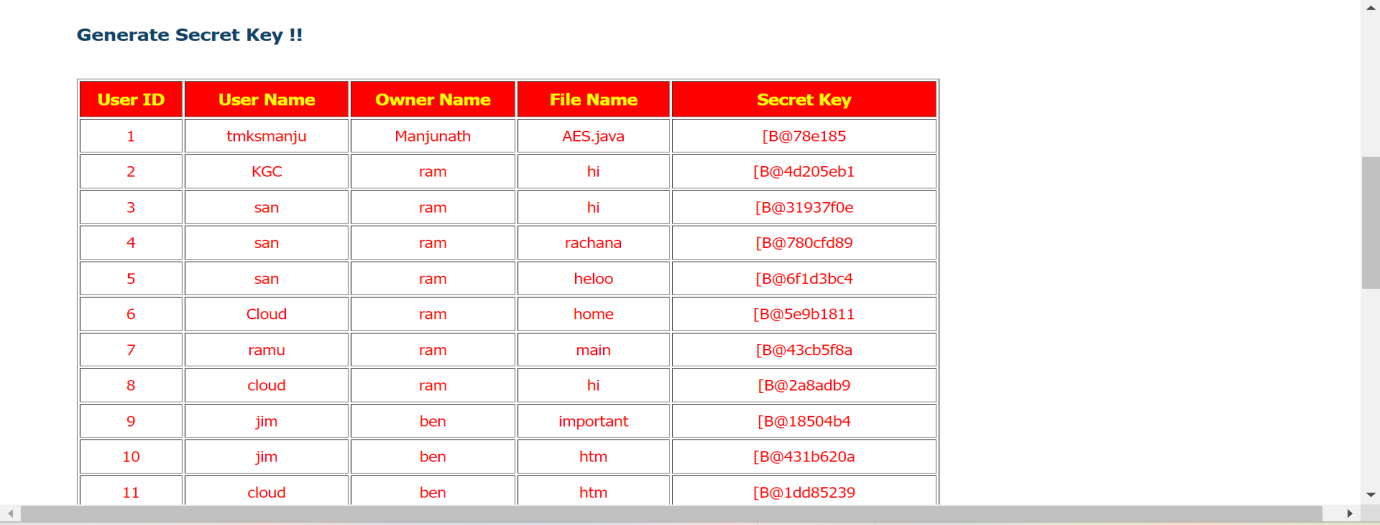
**Fig 8.23: Screenshot of Result (23)**: Request secret key for respective file for accessing the file.



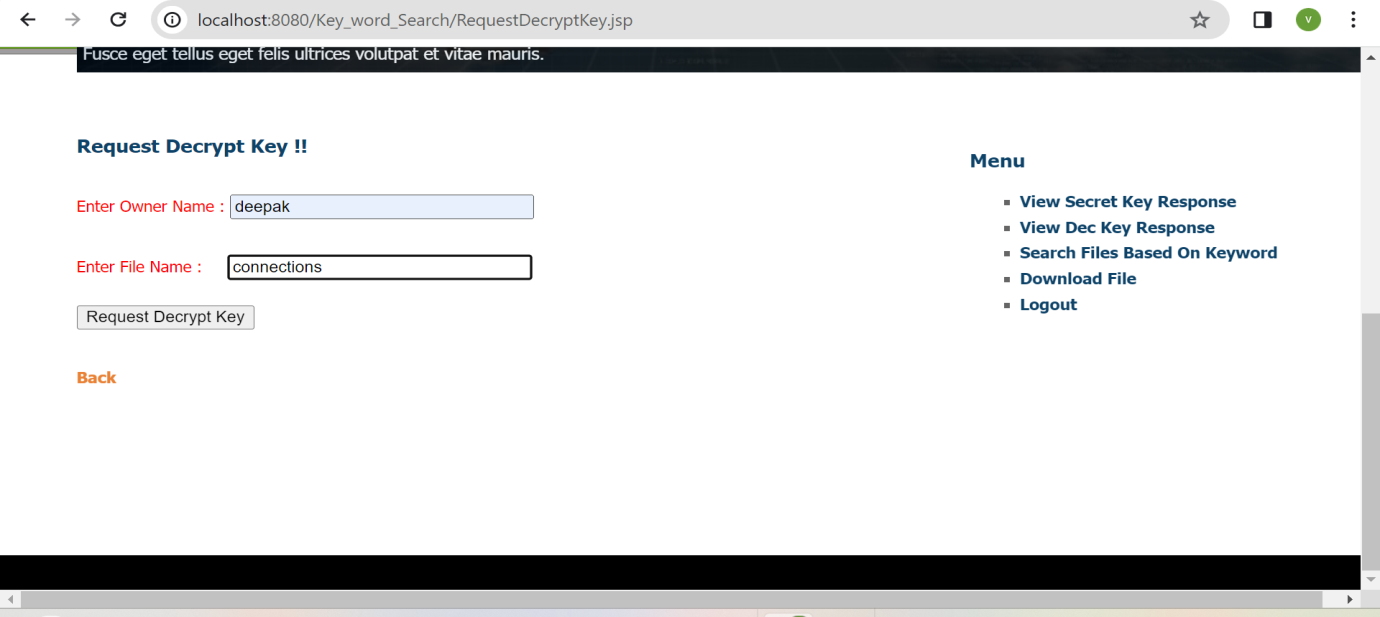
**Fig 8.24: Screenshot of Result (24)**: Request sent to KGC.



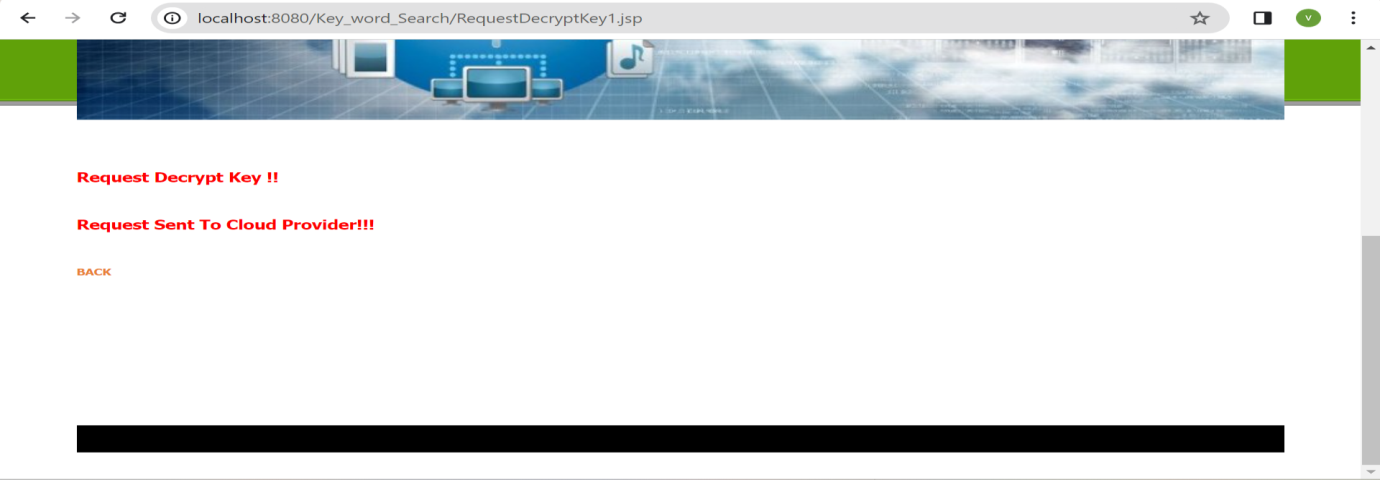
**Fig 8.25: Screenshot of Result (25)**: Viewing the generated secret key.



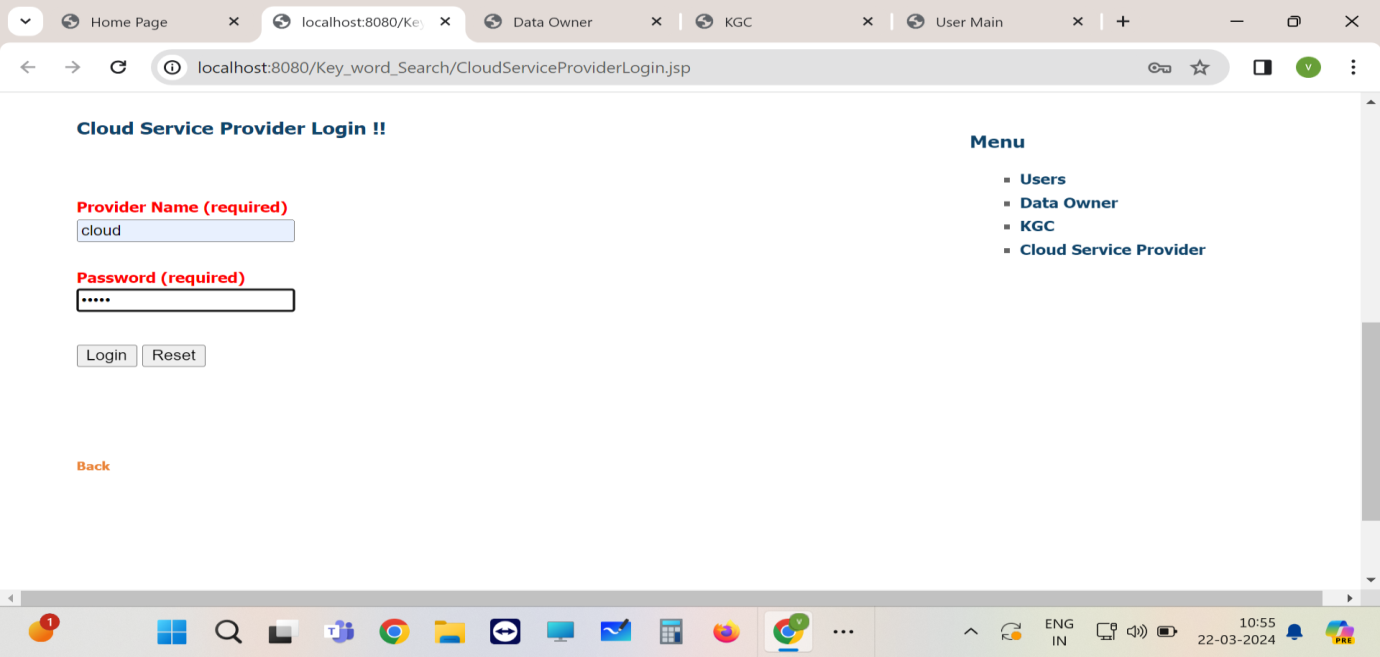
**Fig 8.26: Screenshot of Result (26)**: Viewing the generated secret key.



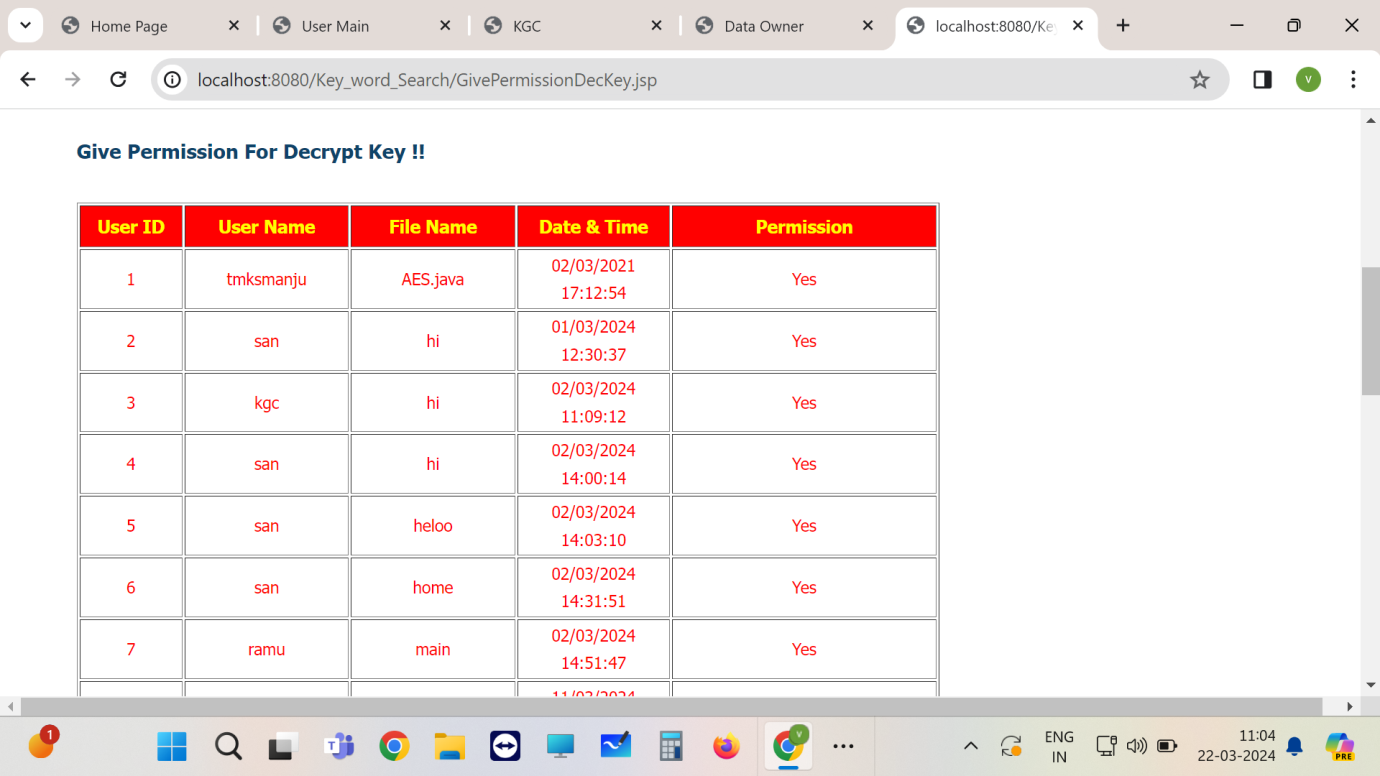
**Fig 8.27: Screenshot of Result (27)**: User requesting for respective file decrypt key.



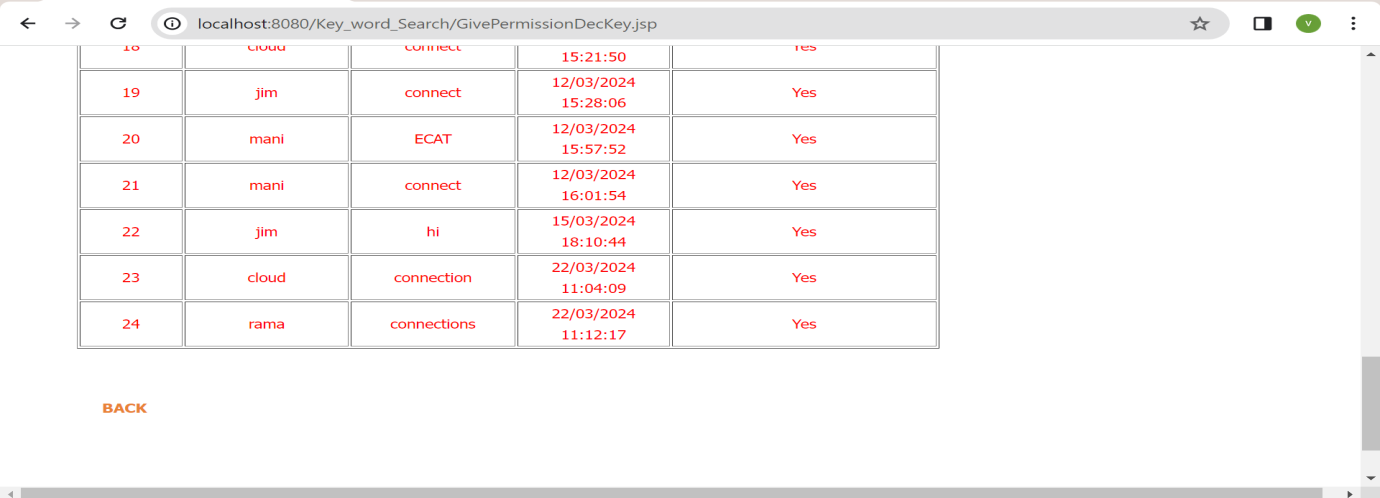
**Fig 8.28: Screenshot of Result (28)**: Request sent to cloud provider.



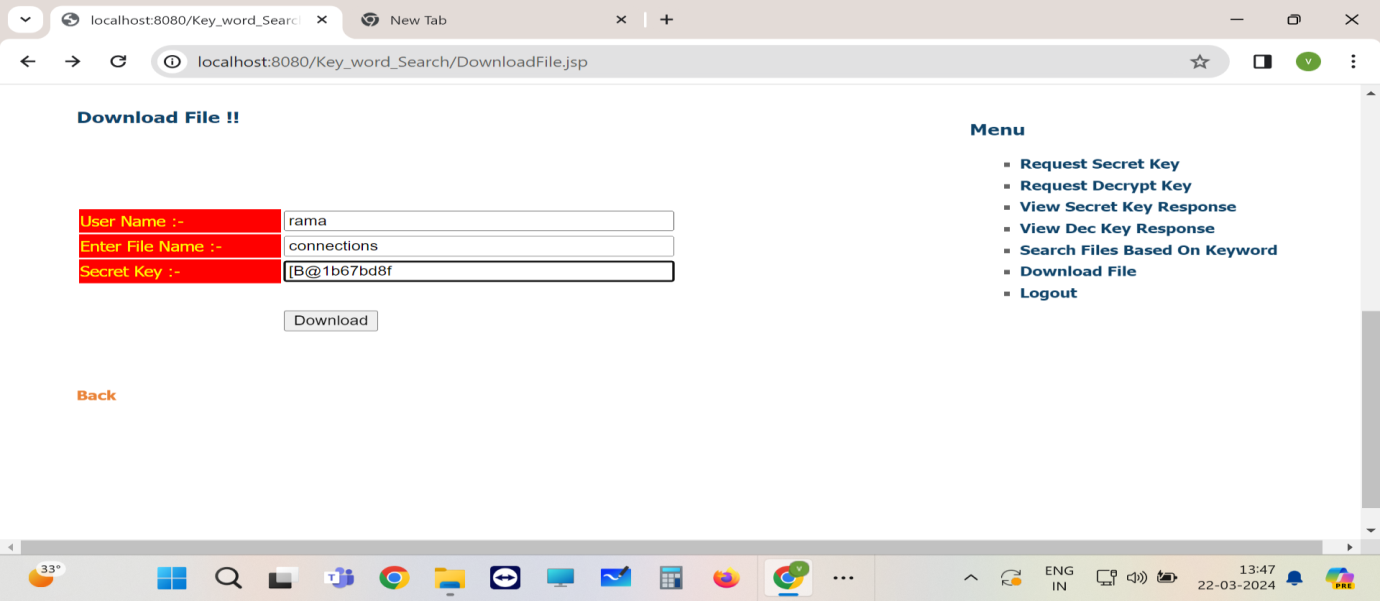
**Fig 8.29: Screenshot of Result (29)**: Cloud provider login.



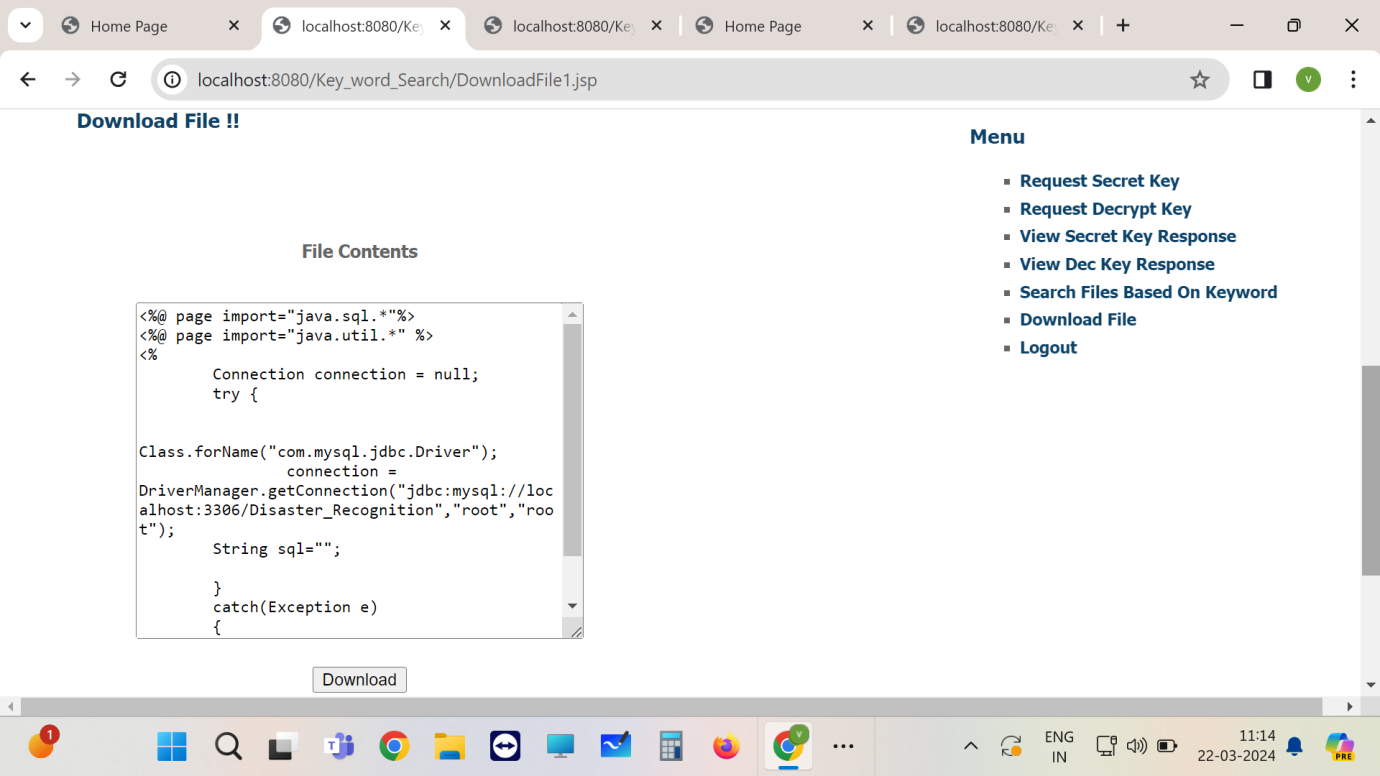
**Fig 8.30: Screenshot of Result (30)**: Cloud provides permission for decrypt key.



**Fig 8.31: Screenshot of Result (31)**:Cloud provides permission for decrypt key.



**Fig 8.32: Screenshot of Result (32)**: User has to enter secret key to download the respective file.



**Fig 8.33: Screenshot of Result (33)**:Click on download button to download the respective file locally.

**CHAPTER-9:CONCLUSION & FUTURE ENHANCEMENT**

**9.1 Conclusion:**

our research presents a comprehensive solution for practical multi-keyword ranked search with access control over encrypted cloud data. By leveraging homomorphic encryption, efficient index structures, and access control mechanisms, we have addressed the critical challenges of privacy-preserving search and access control in cloud storage environments.

Through extensive experimentation and evaluation, we have demonstrated the effectiveness and efficiency of our proposed solution. Our system achieves secure and efficient search operations while preserving the confidentiality and integrity of the data stored in the cloud. Furthermore, our approach offers fine-grained access control, enabling organizations to enforce access policies based on user attributes or roles.

Our work contributes to advancing the state-of-the-art in secure cloud computing, offering a practical solution for protecting sensitive data in the cloud. By providing a secure and efficient means of searching and accessing encrypted data, our solution paves the way for the widespread adoption of cloud storage in sensitive domains such as healthcare, finance, and government.

Moving forward, we envision further research in enhancing the scalability and usability of our solution, as well as exploring new avenues for improving the security guarantees and performance optimizations. By continuing to innovate in the field of secure cloud computing, we can empower organizations to harness the benefits of cloud storage while mitigating the risks associated with data privacy and security.

**9.2 Future Enhancement:**

Future enhancements to the proposed system encompass scalability, usability, and performance optimization, along with advanced access control policies and heightened security measures. Additionally, support for various data types, integration with privacy-preserving technologies, and real-world deployment for validation are crucial for ensuring its effectiveness and applicability in practical scenarios.

**CHAPTER 10: BIBLIOGRAPHY**

1. F.-H. Liu, H.-F. Lo, L.-C. Chen, and W.-T. Lee, “Comprehensive security integrated model and ontology within cloud computing,” J. Internet Technol., vol. 14, no. 6, pp. 935–946, 2013.

2. M. J. Atallah, Algorithms and Theory of Computation Handbook. Boca Raton, FL, USA: CRC Press, 1998.

3. M. Armbrust et al., “A view of cloud computing,” Commun. ACM, vol. 53, no. 4, pp. 50–58, Apr. 2010.

4. B. Bloom, “Space/time trade-offs in hash coding with allowable errors,” Commun. ACM, vol. 13, no. 7, pp. 422–426, Jul. 1970.

5. B. Schneier, Applied Cryptography: Protocols, Algorithms, Source Code in C. Hoboken, NJ, USA: Wiley, 1996.

6. M. Bellare, A. Boldyreva, and A. O’Neill, “Deterministic and efficiently searchable encryption,” in Proc. Annu. CRYPTO, 2007, pp. 535–552.

7. D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, “Public key encryption with keyword search,” in Proc. Int. Conf. Theory Appl. EUROCRYPT, 2004, pp. 506–522.

8. L. Ballard, S. Kamara, and F. Monrose, “Achieving efficient conjunctive keyword searches over encrypted data,” in Proc. ICICS, 2005, pp. 414–426.

9. A. Broder and M. Mitzenmacher, “Network applications of bloom filters: A survey,” Internet Math., vol. 1, pp. 485–509, 2005.

10. D. Boneh and B.Waters, “Conjunctive, subset, range queries on encrypted data,” in Proc. Theory of Cryptography Conf. TCC, 2007, pp. 535–554.