**CHAPTER 1**

**INTRODUCTION**

**1.1 OVERVIEW OF THE PROJECT:**

Now a day’s visual data is responsible for one of the largest shares of global Internet traffic in both corporate and personal use scenarios. The amounts of images, graphics, and photos being generated and shared every day, especially through mobile devices, is growing at an everincreasing rate. The storage needs for such large amounts of data in resource-constrained mobile devices has been a driving factor for data outsourcing services such as the ones leveraging Cloud Storage and computing solutions. Such services (e.g. Instagram and Flickr) have been reported to be among the largest growing internet services. Additionally, the availability of large amounts of images in public and private repositories also leads to the need for contentbased search and retrieval solutions.

**OBJECTIVE**

The scope of the project is retrieving the images from the repositories based on content search. The repositories will maintain their unique keys and the images in there positories also maintain their unique keys for the purpose of security.

**1.2 REQUIREMENTS OF THE PROJECT**

**1.2.1 HARDWARE REQUIREMENTS**

The hardware requirements may serve as the basis for a contract for the Implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design. It shows what the system does and not how it should be implemented.

**HARDWARE**

* PROCESSOR : PENTIUM IV 2.6 GHz, Intel Core 2 Duo.
* RAM : 4GB DD RAM
* MONITOR : 15” LCDLED MONITOR
* HARD DISK : 40 GB
  + 1. **SOFTWARE REQUIREMENTS**

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the teams and tracking the team’s progress throughout the development activity.

**SOFTWARE**

Front End : Core Java, J2EE (Servlets, Jsp)

Back End : My sql 5.5

Operating System : Windows 7

IDE : Eclipse

**1.2.3 FUNCTIONAL REQUIREMENTS**

A functional requirement defines a function of a software-system or its component. A function is described as a set of inputs, the behavior, and outputs. The proposed system is achieved by suppression-based and generalization-based k-anonymous and confidential databases. The protocols rely on well-known cryptographic assumptions, and we provide theoretical analyses to proof their soundness and experimental results to illustrate their efficiency.

**1.2.4 NON-FUNCTIONAL REQUIREMENTS**

**EFFICIENCY**

To address the scalability issue, we propose an edge-centric clustering scheme to extract sparse social dimensions. In sparse social dimensions, the social dimensionbased approach can efficiently handle networks of millions of actors while demonstrating comparable prediction performance as other non-scalable methods.

Despite the fact that data outsourcing, especially to cloud computing infrastructures, seems a natural solution to support large scale image storage and retrieval systems, it also raises new challenges in terms of data privacy control. Privacy should not be expected to be preserved by cloud providers. Furthermore, malicious or simply careless system administrators working for the providers have full access to data on the hosting cloud machines.Finally, external hackers can exploit software vulnerabilities to gain unauthorized access to servers.

* 1. **INTRODUCTION TO EXISTING DESIGN**

An outsourcing servers to generate and update an index used to efficiently process and reply to queries, a task that in manystate of art solutions must be managed by client devices. As we show further ahead in the paper, our new methodology leads to optimized computation and communication overheads with non-negligible impact on system performance and mobile battery consumption. Outsourced image storage, search, and retrieval framework by leveraging IES-CBIR to avoid most heavy computations to be performed by the client (i.e. indexing of dynamically added/updated images).

**1.4 NEED OF PROPOSED SYSTEM**

**1.4.1 Proposed System Advantages**

* Privacy preserving outsourced storage, search, and retrieval of large-scale, dynamically updated image repositories.
* To upload their endless data.

**1.5 SCOPE OF THE PROJECT**

* The scope of the project is retrieving the images from the repositories based on content search.
* The repositories will maintain their unique keys and the images in the repositories also maintain their unique keys for the purpose of security.

**CHAPTER 2**

**LITERATURE SURVEY**

**Literature Survey:**

Class-tested and coherent, this textbook teaches classical and web information retrieval, including web search and the related areas of text classification and text clustering from basic concepts. It gives an up-to-date treatment of all aspects of the design and implementation of systems for gathering, indexing, and searching documents; methods for evaluating systems; and an introduction to the use of machine learning methods on text collections. All the important ideas are explained using examples and figures, making it perfect for introductory courses in information retrieval for advanced undergraduates and graduate students in computer science. Based on feedback from extensive classroom experience, the book has been carefully structured in order to make teaching more natural and effective. Slides and additional exercises (with solutions for lecturers) are also available through the book's supporting website to help course instructors prepare their lectures.3

Cloud computing is clearly one of today’s most enticing technology areas due, at least in part, to its cost-efficiency and flexibility. However, despite the surge in activity and interest, there are significant, persistent concerns about cloud computing that are impeding momentum and will eventually compromise the vision of cloud computing as a new IT procurement model. In this paper, we characterize the problems and their impact on adoption. In addition, and equally importantly, we describe how the combination of existing research thrusts has the potential to alleviate many of the concerns impeding adoption. In particular, we argue that with continued research advances in trusted computing and computation-supporting encryption, life in the cloud can be advantageous from a business intelligence standpoint over the isolated alternative that is more common today.4

Contrary to popular assumption, DRAMs used in most modern computers retain their contents for several seconds after power is lost, even at room temperature and even if removed from a motherboard. Although DRAMs become less reliable when they are not refreshed, they are not immediately erased, and their contents persist sufficiently for malicious (or forensic) acquisition of usable full-system memory images. We show that this phenomenon limits the ability of an operating system to protect cryptographic key material from an attacker with physical access. We use cold reboots to mount successful attacks on popular disk encryption systems using no special devices or materials. We experimentally characterize the extent and predictability of memory remanence and report that remanence times can be increased dramatically with simple cooling techniques. We offer new algorithms for finding cryptographic keys in memory images and for correcting errors caused by bit decay. Though we discuss several strategies for partially mitigating these risks, we know of no simple remedy that would eliminate them.8

The paper describes the design, implementation, and evaluation of Depot, a cloud storage system that minimizes trust assumptions. Depot tolerates buggy or malicious behavior by any number of clients or servers, yet it provides safety and liveness guarantees to correct clients. Depot provides these guarantees using a two-layer architecture. First, Depot ensures that the updates observed by correct nodes are consistently ordered under Fork-JoinCausal consistency (FJC). FJC is a slight weakening of causal consistency that can be both safe and live despite faulty nodes. Second, Depot implements protocols that use this consistent ordering of updates to provide other desirable consistency, staleness, durability, and recovery properties. Our evaluation suggests that the costs of these guarantees are modest and that Depot can tolerate faults and maintain good availability, latency, overhead, and staleness even when significant faults occur.11

We describe a working implementation of leveled homomorphic encryption (without bootstrapping) that can evaluate the AES-128 circuit in three different ways. One variant takes under over 36 hours to evaluate an entire AES encryption operation, using NTL (over GMP) as our underlying software platform, and running on a large-memory machine. Using SIMD techniques, we can process over 54 blocks in each evaluation, yielding an amortized rate of just under 40 minutes per block. Another implementation takes just over two and a half days to evaluate the AES operation, but can process 720 blocks in each evaluation, yielding an amortized rate of just over five minutes per block. We also detail a third implementation, which theoretically could yield even better amortized complexity, but in practice turns out to be less competitive forour implementations we develop both AES-specific optimizations as well as several “generic” tools for FHE evaluation. These last tools include (among others) a different variant of the Brakerski-Vaikuntanathan key-switching technique that does not require reducing the norm of the ciphertext vector, and a method of implementing the Brakerski-Gentry-Vaikuntanathan modulus-switching transformation on ciphertexts in CRT representation.12

**CHAPTER 3**

**SYSTEM ANALYSIS**

Despite the fact that data outsourcing, especially to cloud computing infrastructures, seems a natural solution to support large scale image storage and retrieval systems, it also raises new challenges in terms of data privacy control. Privacy should not be expected to be preserved by cloud providers. Furthermore, malicious or simply careless system administrators working for the providers have full access to data on the hosting cloud machines, Finally, external hackers can exploit software vulnerabilities to gain unauthorized access to servers.

**3.1 EXISTING SYSTEM**

**3.1.1** **EXISTINGSYSTEM DISADVANTAGES**

* Malicious or simply careless system administrators working for the providers have full access to data on the hosting cloud machines.
* The risk of information leakage, a user should obtain authorization from the data owner for accessing the encrypted data.

An outsourcing servers to generate and update an index used to efficiently process and reply to queries, a task that in many states of art solutions must be managed by client devices. As we show further ahead in the paper, our new methodology leads to optimized computation and communication overheads with non-negligible impact on system performance and mobile battery consumption. Outsourced image storage, search, and retrieval framework by leveraging IES-CBIR to avoid most heavy computations to be performed by the client (i.e. indexing of dynamically added/updated images).

**3.2 PROPOSED SYSTEM**

**3.2.1 PROPOSED SYSTEM ADVANTAGES**

* Privacy preserving outsourced storage, search, and retrieval of large-scale, dynamically updated image repositories.
* To upload their endless data.

**3.3 FEASIBILITY STUDY OF THE PROPOSED SYSTEM**

Now a day’s visual data is responsible for one of the largest shares of global Internet traffic in both corporate and personal use scenarios. The amounts of images, graphics, and photos being generated and shared every day, especially through mobile devices, is growing at an ever increase rate. The storage needs for such large amounts of data in resource-constrained mobile devices has been a driving factor for data outsourcing services such as the ones leveraging Cloud Storage and computing solutions. Such services (e.g. Instagram and Flickr) have been reported to be among the largest growing internet services. Additionally, the availability of large amounts of images in public and private repositories also leads to the need for contentbased search and retrieval solutions.

**3.4 SYSTEM REQUIREMENT SPECIFICATION**

**FUNCTIONAL REQUIREMENTS**

A functional requirement defines a function of a software-system or its component. A function is described as a set of inputs, the behavior, and outputs. The proposed system is achieved by suppression-based and generalization-based k-anonymous and confidential databases. The protocols rely on well-known cryptographic assumptions, and we provide theoretical analyses to proof their soundness and experimental results to illustrate their efficiency.

**NON-FUNCTIONAL REQUIREMENTS**

* **EFFICIENCY**

To address the scalability issue, we propose an edge-centric clustering scheme to extract sparse social dimensions. In sparse social dimensions, the social dimensionbased approach can efficiently handle networks of millions of actors while demonstrating comparable prediction performance as other non-scalable methods.

* **RELIABILITY**

The dynamic nature of networks entails efficient update of the model for collective behavior prediction.

**3.5** **SCOPE OF THE PROJECT**

* The scope of the project is retrieving the images from the repositories based on content search.
* The repositories will maintain their unique keys and the images in the repositories also maintain their unique keys for the purpose of security.

**CHAPTER 4**

**PROPOSED SYSTEM DESIGN**

**4.1 SYSTEM ARCHITECTURE**

Design Engineering deals with the various UML [Unified Modeling language] diagrams for the implementation of project. Design is a meaningful engineering representation of a thing that is to be built. Software design is a process through which the requirements are translated into representation of the software. Design is the place where quality is rendered in software engineering. Design is the means to accurately translate customer requirements into finished product.

Search by content

Related images

Request to image owner

Get image key as response

Download image

Login

Register

Create repository

Request to admin

Add images

Key maintenance

Register

Login

Repository

Key as response

Add images to repository

Request to repository owner

Get repository key as response

Key maintenance

Register

Login

**Fig 4.1: Architecture Diagram for Repository, Image, Third Party Owner**

**EXPLANATION:**

The system architecture represents the proposed flow of the application. In this project first the data owners had right to utilize the cloud space for uploading their files but those are outsourced in encryption format i.e., cipher text. The cipher text splits and stored in different blocks not in a single file due to this we are providing more confidentiality. The Data user first get permission from the authorities of the application then only able to access files from the cloud space of this application otherwise redirected to flaws .If datautilizer i.e., user able to search then he/she can get an encrypted file but with only encryption key, file cannot be decrypted .For decrypting the file we need an instant key i.e., symmetric key which will relates to server who maintains file blocks. These two keys can retrieve the original content of the file without any leakages.

**4.2 ALGORITHMS**

**ALGORITHM USED**

Storage requirements for visual data have been increasing in recent years, following the emergence of many new highly interactive multimedia services and applications for both personal and corporate use. Outsourced and distributed privacy preserving storage and retrieval in large image repositories. Our proposal is based on a novel cryptographic scheme, named IES-CBIR, specifically designed for media image data. Our solution enables both encrypted storage and querying using Content Based Image Retrieval (CBIR) while preserving privacy.

Image Encryption Scheme (IES) with Content-Based Image Retrieval (CBIR):

**4.3 FLOW CHART**

****

**Fig 4.3(A): Activity Diagram for Repository, Image Owner, Third Party User**

**EXPLANATION:**

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

****

**Fig 4.3(B): Use Case Diagram for Repository, Image, Third Party User.**

**EXPLANATION:**

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. The above diagram consists of user as actor. Each will play a certain role to achieve the concept.

**4.4 DATAFLOW DIAGRAM**

Repository Owner

Image Owner

TPU

Register

Login

Verify login

Database

Home Page

Error Page

**Fig 4.4(B): Data Flow Diagram of Level 0**

Create repository by sending request to admin

Admin response

(Accept/decline)

Add images

Key maintenance

Add images to repository

Request to repository owner

Get repository key as response

Key maintenance

Search by content

Related images

Request to image owner

Get image key as response

Download image

Database

**Fig 4.4(B): Data flow Diagram for Level 1**

**Explanation**

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects, often they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design). A DFD shows what kinds of data will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of processes, or information about whether processes will operate in sequence or in parallel.

****

**Fig 4.4(C): Class Diagram for Repository Owner, Admin, Image Owner, TPU.**

**EXPLANATION:**

In this class diagram represents how the classes with attributes and methods are linked together to perform the verification with security. From the above diagram shown the various classes involved in our project.

****

**Fig 4.4(D): Deployment Diagram for Database**

Create repository by sending request to admin

Admin response

(Accept/decline)

Add images

Key maintenance

Add images to reposry

Request to repository owner

Get repository key as response

Key maintenance

Search by content

Related images

Request to image owner

Get image key as response

Download image

Database

Register

Login

Register

Register

Login

Login

**Fig 4.4(E): Entity Relation(E-R) for Modules**

**EXPLANATION:**

In software engineering, an entity-relationship model (ERM) is an abstract and conceptual representation of data. Entity-relationship modeling is a database modeling method, used to produce a type of conceptual schema or semantic data model of a system, often a relational database, and its requirements in a top-down fashion. Diagrams created by this process are called entity-relationship diagrams, ER diagrams, or ERDs. User gives main query and it converted into sub queries and sends through data dissemination to data aggregators. Results are to be showed to user by data aggregators.

****

**Fig 4.4(F): State Diagram for Database**

**Explanation**

State diagram are a loosely defined diagram to show workflows of stepwise activities and actions, with support for choice, iteration and concurrency. UML, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. UML activity diagrams could potentially model the internal logic of a complex operation. In many ways UML activity diagrams are the object-oriented equivalent of flow charts and data flow diagrams (DFDs) from structural development.



**Fig 4.4(G): Component Diagram for Database**

**Explanation**

In the Unified Modeling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems. User gives main query and it converted into sub queries and sends through data dissemination to data aggregators. Results are to be showed to user by data aggregators. All boxes are components and arrow indicatedependencies.

**CONCLUSION**

In this project we addressed the problem of securing data outsourced to the cloud against an adversary which has access to the encryption key. For that purpose, we introduced a novel security definition that captures data confidentiality against the new adversary. We then proposed Bastion, a scheme which ensures the confidentiality of encrypted data even when the adversary has the encryption key, and all but *two* cipher text blocks. Bastion is most suitable for settings where the cipher text blocks are stored in multi-cloud storage systems. In these settings, the adversary would need to acquire the encryption key as well as instant server key which is related to different blocks for accessing the encrypted file to decrypt.

**CHAPTER 5**

**IMPLEMENTATION OF PROPOSED SYSTEM**

**5.1 TECHNOLOGIES USED**

This chapter is about the software language and the tools used in the development of the project. The platform used here is JAVA. The Primary languages are JAVA,J2EE and J2ME. In this project J2EE is chosen for implementation.

**5.1.1 THE JAVA FRAMEWORK**

**Java** is a [programming language](http://en.wikipedia.org/wiki/Programming_language) originally developed by [James Gosling](http://en.wikipedia.org/wiki/James_Gosling) at Microsystems and released in 1995 as a core component of Sun Microsystems' [Java platform](http://en.wikipedia.org/wiki/Java_(software_platform)). The language derives much of its [syntax](http://en.wikipedia.org/wiki/Syntax_(programming_languages)) from [C](http://en.wikipedia.org/wiki/C_(programming_language)) and [C++](http://en.wikipedia.org/wiki/C%2B%2B) but has a simpler [object model](http://en.wikipedia.org/wiki/Object_model) and fewer [low-level](http://en.wikipedia.org/wiki/Low-level_programming_language) facilities. Java applications are typically [compiled](http://en.wikipedia.org/wiki/Compiler) to [bytecode](http://en.wikipedia.org/wiki/Java_bytecode) that can run on any [Java Virtual Machine](http://en.wikipedia.org/wiki/Java_Virtual_Machine) (JVM) regardless of [computer architecture](http://en.wikipedia.org/wiki/Computer_architecture). Java is general-purpose, concurrent, class-based, and object-oriented, and is specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere".

Java is considered by many as one of the most influential programming languages of the 20th century, and is widely used from application software to web applications. The java framework is a new platform independent that simplifies application development internet. Java technology's versatility, efficiency, platform portability, and security make it the ideal technology for network computing. From laptops to datacenters, game consoles to scientific supercomputers, cell phones to the Internet, Java is everywhere!

**5.1.2 OBJECTIVES OF JAVA**

To see places of Java in Action in our daily life, explore java.com.

**Why Software Developers Choose Java:** Java has been tested, refined, extended, and proven by a dedicated community. And numbering more than 6.5 million developers, it's the largest and most active on the planet. With its versatility, efficiency, and portability, Java has become invaluable to developers by enabling them to:

* Write software on one platform and run it on virtually any other platform
* Create programs to run within a Web browser and Web services
* Develop server-side applications for online forums, stores, polls, HTML forms processing, and more
* Combine applications or services using the Java language to create highly customized applications or services
* Write powerful and efficient applications for mobile phones, remote processors, low-cost consumer products, and practically any other device with a digital heartbeat

**Some Ways Software Developers Learn Java:**Today, many colleges and universities offer courses in programming for the Java platform. In addition, developers can also enhance their Java programming skills by reading Sun's java.sun.com Web site, subscribing to Java technology-focused newsletters, using the Java Tutorial and the New to Java Programming Center, and signing up for Web, virtual, or instructor-led courses.

**Object Oriented** To be an ObjectOriented language, any language must follow at least the four characteristics.

1. Inheritance: It is the process of creating the new classes and using the behavior of the existing classes by extending them just to reuse the existing code and adding addition a feature’s as needed.

2. Encapsulation: It is the mechanism of combining the information and providing the abstraction.

3. Polymorphism: As the name suggest one name multiple form, Polymorphism is the way of providing the different functionality by the functions having the same name based on the signatures of the methods.

4. Dynamic binding: Sometimes we don't have the knowledge of objects about their specific types while writing our code. It is the way of providing the maximum functionality to a program about the specific type at runtime.

**5.1.3Java Server Pages - An Overview**

Java Server Pages or JSP for short is Sun's solution for developing dynamic web sites. JSP provide excellent serverside scripting support for creating database driven web applications. JSP enable the developers to directly insert java code into jsp file, this makes the development process very simple and its maintenance also becomes very easy.JSP pages are efficient, it loads into the web server’s memory on receiving the request very first time and the subsequent calls are served within a very short period of time.

In today's environment most web sites server’s dynamic pages based on user request. Database is very convenient way to store the data of users and other things. JDBC provide excellent database connectivity in heterogeneous database environment. Using JSP and JDBC it very easy to develop database driven web application.Java is known for its characteristic of "write once, run anywhere." JSP pages are flat Java Server Pages.Java Server Pages (JSP) technology is the Java platform technology for delivering dynamic content to web clients in a portable, secure and well-defined way. The Java Server Pages specification extends the Java Servlet API to provide web application developers with a robust framework for creating dynamic web content on the server using HTML, and XML templates, and Java code, which is secure, fast, and independent of server platforms.JSP has been built on top of the Servlet API and utilizes Servlet semantics. JSP has become the preferred request handler and response mechanism. Although JSP technology is going to be a powerful successor to basic Servlets, they have an evolutionary relationship and can be used in a cooperative and complementary manner.Servlets are powerful and sometimes they are a bit cumbersome when it comes to generating complex HTML. Most servlets contain a little code that handles application logic and a lot more code that handles output formatting. This can make it difficult to separate and reuse portions of the code when a different output format is needed. For these reasons, web application developers turn towards JSP as their preferred servlets environment.

**5.1.4 Evolution of Web Applications**

Over the last few years, web server applications have evolved from static to dynamic applications. This evolution became necessary due to some deficiencies in earlier web site design. For example, to put more of business processes on the web, whether in business-to-consumer (B2C) or business-to-business (B2B) markets, conventional web site design technologies are not enough. The main issues, every developer faces when developing web applications, are:

1. Scalability - a successful site will have more users and as the number of users is increasing fastly, the web applications have to scale correspondingly.

2. Integration of data and business logic - the web is just another way to conduct business, and so it should be able to use the same middle-tier and data-access code.

3. Manageability - web sites just keep getting bigger and we need some viable mechanism to manage the ever-increasing content and its interaction with business systems.

4. Personalization - adding a personal touch to the [web page](http://www.roseindia.net/jsp/javaserverpagestutorial.shtml) becomes an essential factor to keep our customer coming back again. Knowing their preferences, allowing them to configure the information they view, remembering their past transactions or frequent search keywords are all important in providing feedback and interaction from what is otherwise a fairly one-sided conversation.

Apart from these general needs for a business-oriented web site, the necessity for new technologies to create robust, dynamic and compact server-side web applications has been realized. The main characteristics of today's dynamic web server applications are as follows:

1. Serve HTML and XML, and stream data to the web client

2. Separate presentation, logic and data

3. Interface to databases, other Java applications, CORBA, directory and mail services

4. Make use of application server middleware to provide transactional support.

5. Track client sessions.

**5.1.5 Benefits of JSP**

One of the main reasons why the Java Server Pages technology has evolved into what it is today and it is still evolving is the overwhelming technical need to simplify application design by separating dynamic content from static template display data. Another benefit of utilizing JSP is that it allows to more cleanly separating the roles of web application/HTML designer from a software developer. The JSP technology is blessed with a number of exciting benefits, which are chronicled as follows:

1. The JSP technology is platform independent, in its dynamic web pages, its web servers, and its underlying server components. That is, JSP pages perform perfectly without any hassle on any platform, run on any web server, and web-enabled application server. The JSP pages can be accessed from any web server.

2. The JSP technology emphasizes the use of reusable components. These components can be combined or manipulated towards developing more purposeful components and page design. This definitely reduces development time apart from the at development time, JSPs are very different from Servlets, however, they are precompiled into Servlets at run time and executed by a JSP engine which is installed on a Web-enabled application server such as BEA WebLogic and IBM WebSphere.

**5.1.6 Servlets**

Earlier in client- server computing, each application had its own client program and it worked as a user interface and need to be installed on each user's personal computer. Most web applications use HTML/XHTML that are mostly supported by all the browsers and web pages are displayed to the client as static documents.A web page can merely display’s static content and it also lets the user navigate through the content, but a web application provides a more interactive experience.Any computer running Servlets or JSP needs to have a container. A container is nothing but a piece of software responsible for loading, executing and unloading the Servlets and JSP. While servlets can be used to extend the functionality of any Java- enabled server.They are mostly used to extend web servers, and are efficient replacement for CGI scripts. CGI was one of the earliest and most prominent serverside dynamic content solutions, so before going forward it is very important to know the difference between CGI and the Servlets.

**5.1.7 Java Servlets**

Java Servlet is a generic server extension that means a java class can be loaded dynamically to expand the functionality of a server. Servlets are used with web servers and run inside a Java Virtual Machine (JVM) on the server so these are safe and portable.Unlike applets they do not require support for java in the web browser. Unlike CGI, servlets don't use multiple processes to handle separate request. Servets can be handled by separate threads within the same process. Servlets are also portable and platform independent.A web server is the combination of computer and the program installed on it. Web server interacts with the client through a web browser. It delivers the [web pages](http://www.roseindia.net/servlets/IntroductionToWebServer.shtml)to the client and to an application by using the web browser and  he HTTP protocols respectively.The define the web server as the package of  large number of programs installed on a computer connected to Internet or intranet for downloading the requested files using [File Transfer](http://www.roseindia.net/servlets/IntroductionToWebServer.shtml) Protocol, serving e-mail and building and publishing web pages. A web server works on a client server model.

**Conclusion**

JSP and Servlets are gaining rapid acceptance as means to provide dynamic content on the Internet. With full access to the Java platform, running from the server in a secure manner, the application possibilities are almost limitless. When JSPs are used with Enterprise JavaBeans technology, e-commerce and database resources can be further enhanced to meet an enterprise's needs for web applications providing secure transactions in an open platform. J2EE technology as a whole makes it easy to develop, deploy and use web server applications instead of mingling with other technologies such as CGI and ASP. There are many tools for facilitating quick web software development and to easily convert existing server-side technologies to JSP and Servlets.

**IMPLEMENTAION OR CODING**

**index.html**

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=ISO-8859-1"*>

<title>Insert title here</title>

<link rel=*"stylesheet"* href=*"w3.css"*>

<link rel=*"stylesheet"* href=*"win8.css"*>

<style>

*.mySlides* {display:*none*;}

</style>

</head>

<body class=*"w3-win8-steel"*>

<h1 STYLE="font-size: *55px*;font-weight:*900* ;height:*200px* ;background-image: *url(top.jpg)*;background-position: *right*;" class=*"w3-center w3-text-blue-gray"*>PRACTICAL PRIVACY-PRESERVING CONTENT BASED

RETRIEVAL IN CLOUD IMAGE REPOSITORIES

</h1>

<div class=*"w3-bar w3-center"*>

<a href=*index.html* class=*"w3-button w3-teal"* style="width:*300px*;">Home</a>

<a href=*reg.jsp* class=*"w3-button w3-teal"*style="width:*300px*;">Register</a>

</div>

<div style="right: *10px*;position: *absolute*; ">

<h1 CLASS=*" w3-grey w3-center"* style="font-size:*27px*; width:*50px*;display: *inline-block*;">L<br>O<br>G<br>I<br>N<br><br>H<br>E<br>R<br>E</h1>

<div style="width:*400px*;display: *inline-block*;">

<form action=*"Login"* method=*"get"*>

<table class=*"w3-table"* style="height: *400px*">

<tr><td><input type=*radio* class=*"w3-radio"* name=*utype* value=*user*> User</td></tr>

<tr><td><input type=*radio* class=*"w3-radio"* name=*utype* value=*admin*> Admin</td></tr>

<tr><td><strong class=*"w3-text-red"*>Enter UserID</strong><br><input class=*"w3-input w3-win8-steel"* type=*text* name=*uname*></td></tr>

<tr><td><strong class=*"w3-text-red"*> Enter Password</strong><br><input class=*"w3-input w3-win8-steel"* Type=*password* name=*pass*></td></tr>

<tr><td><input type=*submit* value=*Login* class=*"w3-button w3-teal"* style="width:*100%*"></td></tr>

<tr><td><input Type=*reset* value=*clear* class=*"w3-button w3-teal"* style="width:*100%*"></td></tr>

</table>

</form></div>

</div>

<div class=*"w3-content w3-section"* style="max-width:*850px*;left:*10px*;position:*absolute*;max-height: *500px*;">

<img class=*"mySlides"* src=*"pics/img1.jpg"* style="width:*850px*;height: *400px*;">

<img class=*"mySlides"* src=*"pics/pic2.jpg"* style="width:*850px*;height: *400px*;">

<img class=*"mySlides"* src=*"pics/pic3.jpg"* style="width:*850px*;height: *400px*;">

</div>

<script>

**var** myIndex = 0;

carousel();

**function** carousel() {

**var** i;

**var** x = document.getElementsByClassName("mySlides");

**for** (i = 0; i < x.length; i++) {

x[i].style.display = "none";

}

myIndex++;

**if** (myIndex > x.length) {myIndex = 1}

x[myIndex-1].style.display = "block";

setTimeout(carousel, 2000); // Change image every 2 seconds

}

</script>

</body>

</html>

**5.2 MODULES**

1. User Interface
2. Repository Owner
3. Image Owner
4. Third party
5. Admin

**MODULE DESCRIPTION**

1. **User Interface:**

Firstwe are differentiating the users depending on access permissions. The following are the different type of user’s each user must possess the authentication process for the purpose of providing security to the accounts and for maintaining the application safe.

1. Repository Owner
2. Image Owner
3. Third party

Login

Register

Login verify

Database

Home

Page

Error/Login page

Third party

Repository owner

Image owner

Access permissions

**Fig 5.2(A): User Interface Module Diagram**

**2. Repository Owner:**

Repository owner means he is creating the repository. And he had the key for that repository. If any want to add the images into his repository the user must need key.

Following are the operations for Repository owner.

* Register.
* Login.
* Send repository creating request for admin.
* Store repository key which he created.
* Adding the images into repository.
* Share key with other users.
* Logout.

Request to admin

Admin process

Home page

Create

Repository

Request

(Repository key/image key)

Response

(Repository key/image key)

Add images

Key maintenance

Logout

**Fig 5.2(B): Repository Owner Module Diagram**

**3. Image Owner:**

Image owner will store the images into different repositories by getting the keys from that particular repository owner. And he had the keys of images which he stored.

Following are the operations for image owner.

* Register.
* Login.
* View repositories.
* Send Repository Key request for respected Repository owner.
* Get the Repository Key from repository owner.
* Upload images into repository.
* Store the image keys which he uploaded.
* Share image keys with different users.
* Logout.

Home page

Request

(Repository Key/Image Key)

Response

(Repository Key/Image Key)

Key Maintenance

Add Image to Repository

Logout

Repository List

**Fig 5.2(C): Image Owner Module Diagram**

**4. Third Party User:**

The thirdparty users have the permissions for searching the images and get images and he don’t have permission like add images.

* Register.
* Login.
* Searching image by content based.
* Select image.
* Send image key request for image owner.
* Get image key from respected owner.
* View the image.
* Download the image.
* Create repository.
* Logout.

Search with Content

Key request to image owner

Response

Related Images

Home Page

Download Image

Create Repository

Logout

**Fig 5.2(D): Third Party User (TPU) Module Diagram**

**5. Admin:**

Admin have the following operations to do.

* Login.
* View repository requests.
* Give response (Accept/decline) the repository request.
* View repository keys.
* View image keys.

He had his unique username and password apart from those he can’t be able to perform any operation why because he can’t get into his home page where these operations are maintained.

Login

Home Page

Response

(Accept/delete)

Repository requests

View image keys

View repository keys

**Fig 5.2(E): Admin Module Diagram**

**CHAPTER 6**

**TESTING AND VALIDATION**

**6.1 SOFTWARE TESTING**

**GENERAL**

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.

**DEVELOPING METHODOLOGIES**

The test process is initiated by developing a comprehensive plan to test the general functionality and special features on a variety of platform combinations. Strict quality control procedures are used.

The process verifies that the application meets the requirements specified in the system requirements document and is bug free. The following are the considerations used to develop the framework from developing the testing methodologies.

**6.2 TYPES OF TESTS**

**6.2.1 Unit testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program input produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**6.2.2 Functional test**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Procedures : interfacing systems or procedures must be invoked.

**6.2.3 System Test**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configurationoriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**6.2.4 Performance Test**

The Performance test ensures that the output be produced within the time limits and the time taken by the system for compiling, giving response to the users and request being send to the system for to retrieve the results.

**6.2.5 Integration Testing**

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**6.2.6 Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Acceptance testing for Data Synchronization:**

* The Acknowledgements will be received by the Sender Node after the Packets are received by the Destination Node
* The Route add operation is done only when there is a Route request in need
* The Status of Nodes information is done automatically in the Cache Updation process

**6.2.7 Build the Test Plan**

Any project can be divided into units that can be further performed for detailed processing. Then a testing strategy for each of this unit is carried out. Unit testing helps to identity the possible bugs in the individual component, so the component that has bugs can be identified and can be rectified from errors.

**6.3 TEST CASES AND SCENARIOS**

**User Interface:**

**Input:** submit username and password as login inputs

**Output:** ifvalid homepage otherwise error page.

**Repository Owner:**

**Input:** create repository by providing repository name

**Output:** repository key will be generated and distributed to repository owner through admin.

**Image Owner:**

**Input:** send request to repository owner for repository key

**Output:** repository owner sent repository key and allow to upload images to the repository.

**Third party:**

**Input:** search the images with the help of content

**Output:** getting related images which are in encrypted format.

**Admin:**

**Input:** properly login to the account which is centralized

**Output:** view repository and image keys as well as proceed the repository creation requests from the repository owners.

**CHAPTER 7**

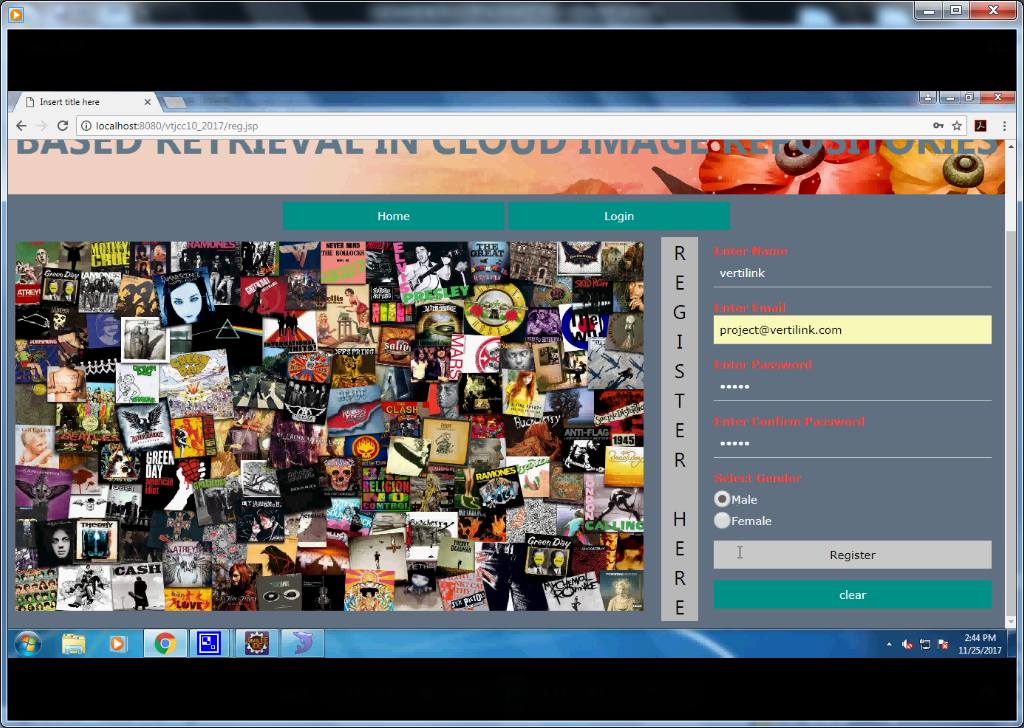
**RESULTS, CONCLUSIONS AND FUTURE SCOPE**

**7.1 RESULTS**

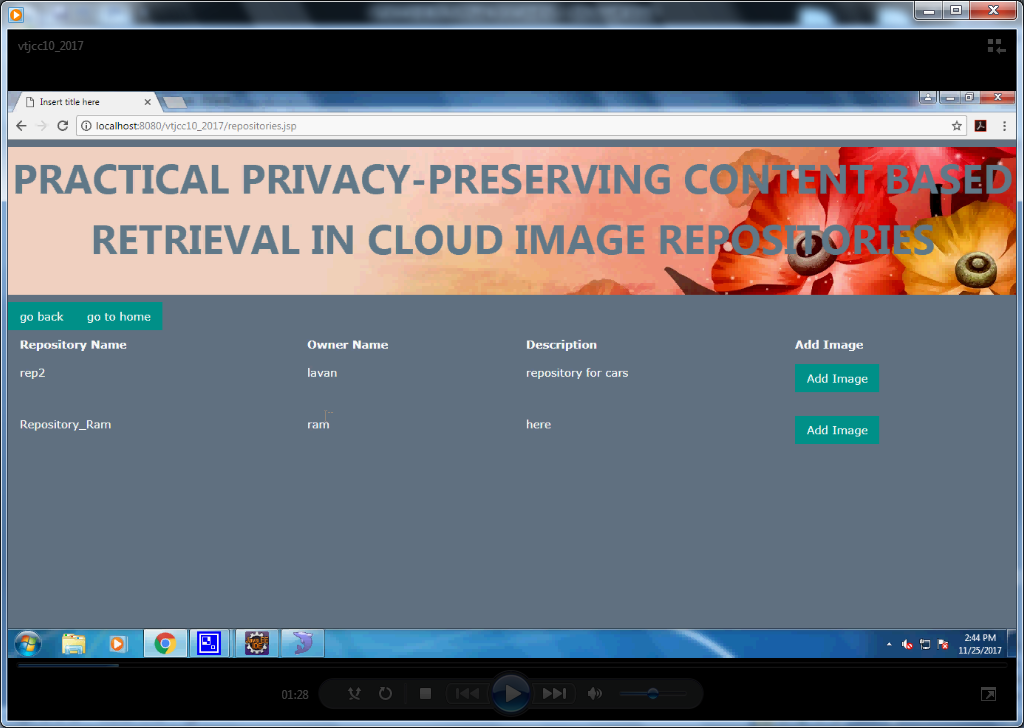
Snapshot is nothing but every moment of the application while running. It gives the clear elaborated of application. It will be useful for the new user to understand for the future steps.

****

**Fig 7.1(A): Web Interface Screenshot**

****

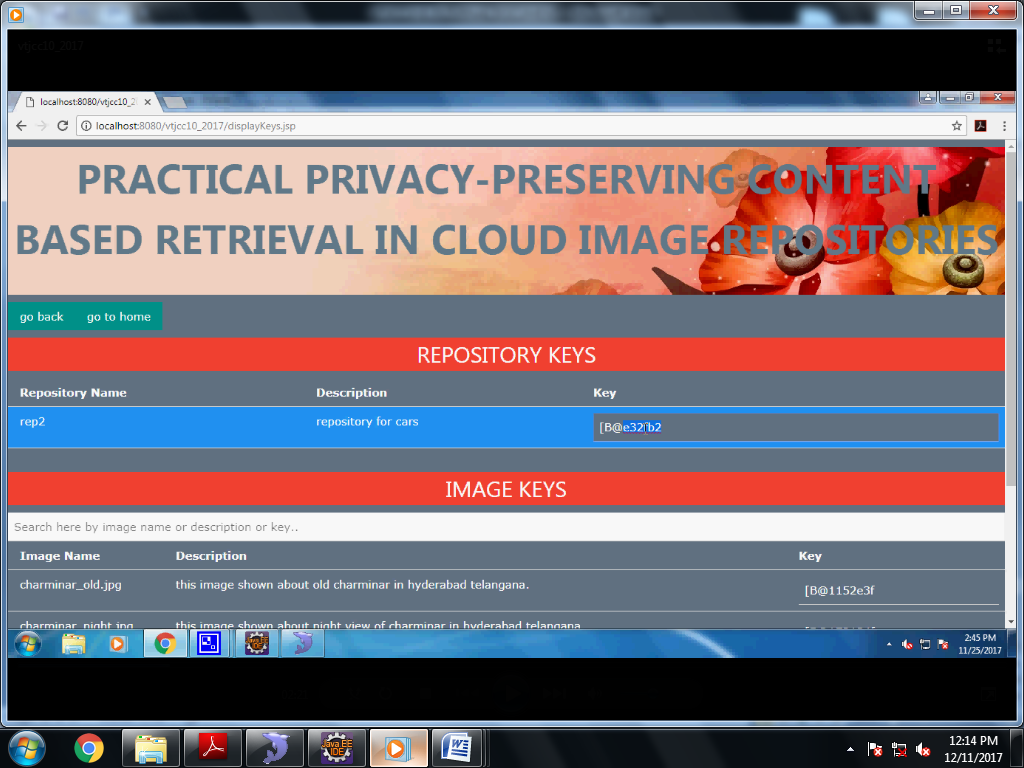
**Fig 7.1(B): Registration for Repository Owner, Image Owner**

****

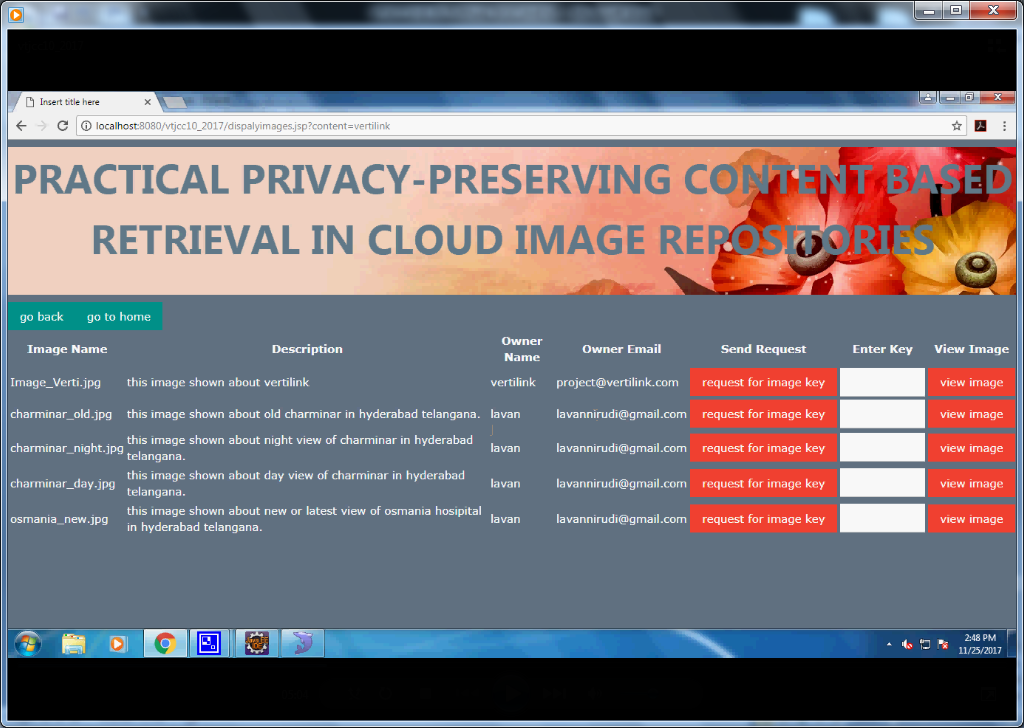
**Fig 7.1(C): Repositories Screenshot**

****

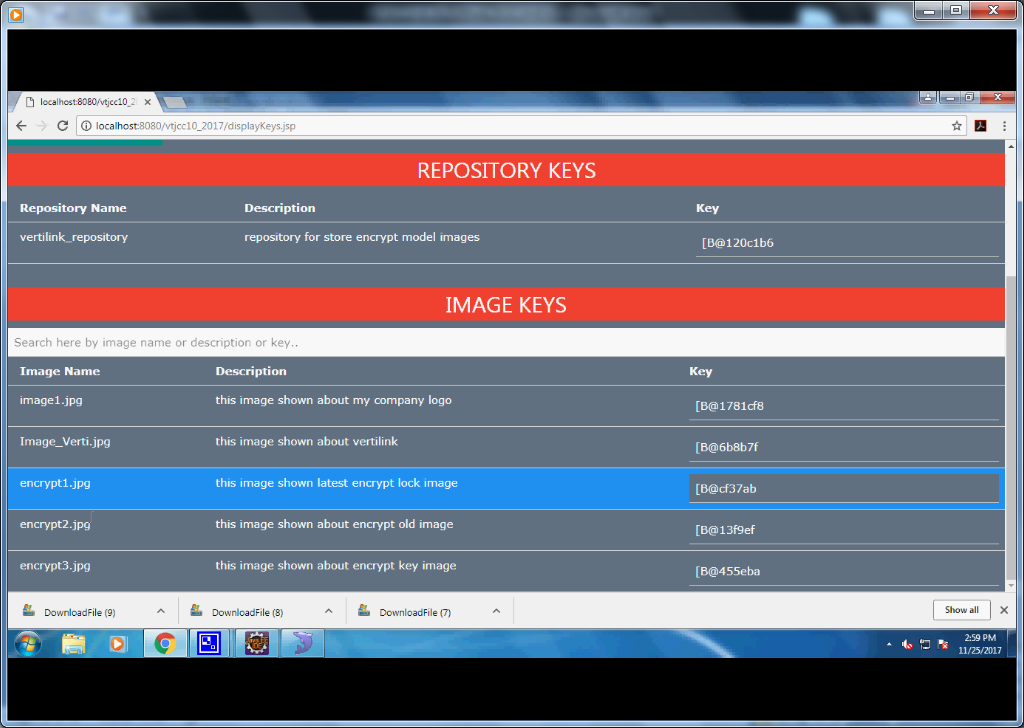
**Fig 7.1(D): Repository Main Page**

****

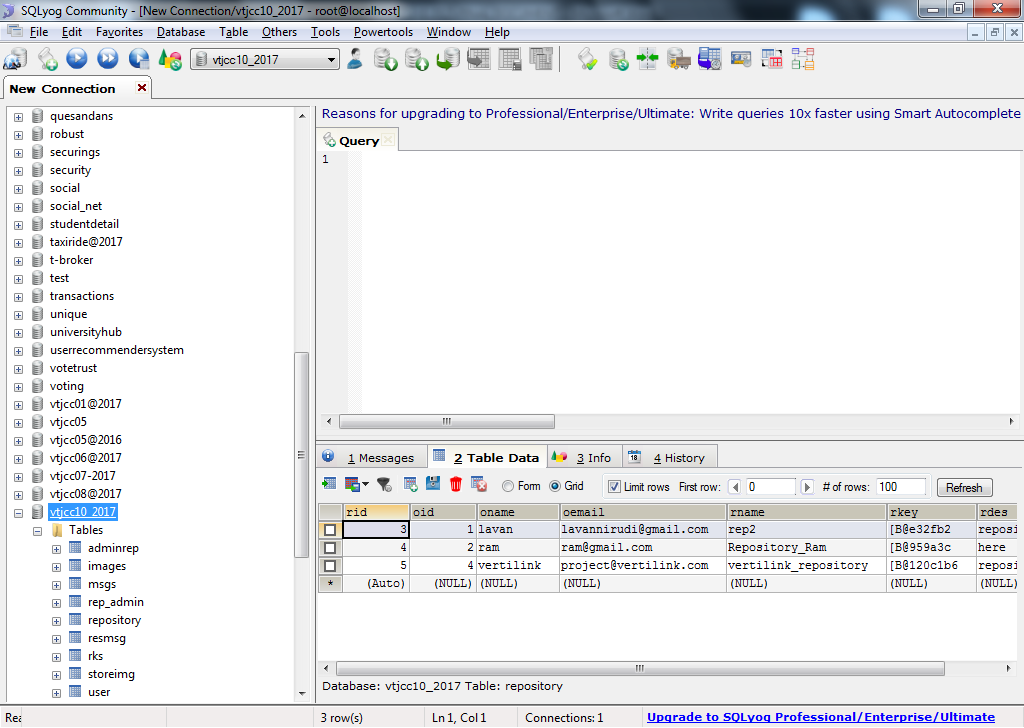
**Fig 7.1(E): Keys for Repository, Image Owner**

****

**Fig 7.1(F): Keys for Image Owner, Repository by Admin**

****

**Fig 7.1(F): Repository Keys Request**

****

**Fig 7.1(G): Querying in MYSQL**

**7.2 CONCLUSION:**

We have proposed a new secure framework for the privacy-preserving outsourced storage, search, and retrieval of large-scale, dynamically updated image repositories, where the reduction of client overheads is a central aspect. In the basis of our framework is a novel cryptographic scheme, specifically designed for images, named IES-CBIR. Key to its design is the observation that in images, color information can be separated from texture information, enabling the use of different encryption techniques with different properties for each one, and allowing privacy preserving Content-Based Image Retrieval to be performed by third-party, untrusted cloud servers.

**7.3 FUTURE SCOPE:**

Additional experimental evaluation of implemented prototypes revealed that our approach achieves an interesting trade-off between precision and recall in CBIR, while exhibiting high performance and scalability when compared with alternative solutions. An interesting future work direction is to investigate the applicability of our methodology - i.e. the separation of information contexts when processing data (color and texture in this contribution) - in other domains beyond image data.

**7.4 REFERENCES**

[1] M. Meeker, “Internet Trends 2015,” in Code Conf., 2015.

[2] Global Web Index, “Instagram tops the list of social network growth,” http://tinyurl.com/hnwwlzm, 2013.

[3] C. D. Manning, P. Raghavan, and H. Sch ¨ utze, An Introduction to Information Retrieval. Cambridge University Press, 2009, vol. 1.

[4] R. Chow, P. Golle, M. Jakobsson, E. Shi, J. Staddon, R. Masuoka, and J. Molina, “Controlling data in the cloud: outsourcing computation without outsourcing control,” in CCSW’09, 2009.

[5] D. Rushe, “Google: don’t expect privacy when sending to Gmail,” http://tinyurl.com/kjga34x, 2013.

[6] G. Greenwald and E. MacAskill, “NSA Prism program taps in to user data of Apple, Google and others,” http://tinyurl.com/oea3g8t, 2013.

[7] A. Chen, “GCreep: Google Engineer Stalked Teens, Spied on Chats,” http://gawker.com/5637234, 2010.

[8] J. Halderman and S. Schoen, “Lest we remember: cold-boot attacks on encryption keys,” in Commun. ACM, vol. 52, no. 5, 2009.

[9] National Vulnerability Database, “CVE Statistics,” http://web.nvd.nist.gov/view/vuln/statistics, 2014.

[10] D. Lewis, “iCloud Data Breach: Hacking and Celebrity Photos,”

https://tinyurl.com/nohznmr, 2014.

[11] P. Mahajan, S. Setty, S. Lee, A. Clement, L. Alvisi, M. Dahlin, and M. Walfish, “Depot: Cloud Storage with Minimal Trust,” ACM Trans. Comput. Syst., vol. 29, no. 4, pp. 1–38, dec 2011.

[12] C. Gentry, S. Halevi, and N. P. Smart, “Homomorphic evaluation of the AES circuit,” in CRYPTO’12. Springer, 2012, pp. 850–867.

[13] P. Paillier, “Public-key cryptosystems based on composite degree residuosity classes,” in EUROCRYPT’99, 1999, pp. 223–238.

[14] T. ElGamal, “A public key cryptosystem and a signature scheme based on discrete logarithms,” in Adv. Cryptol. Springer, 1985.

[15] C.-Y. Hsu, C.-S. Lu, and S.-c. Pei, “Image Feature Extraction in Encrypted Domain with Privacy-Preserving SIFT,” IEEE Trans. Image Process., vol. 21, no. 11, pp. 4593–4607, 2012.