A major project report on

Cloud secure storage using block chain system

submitted in partial fulfillment of the requirements for the award of degree of

Bachelor of Technology

Submitted by

M Sathvika Angel

(20eg105630)

P Manikanta Reddy

(20eg105654)

J. Sumanth

(20eg105707)

Under the guidance of

Dr A Jyothi

Assistance Professor

Department of CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ANURAG UNIVERSITY

VENKATAPUR– 500088

TELANGANA

Year 2023-2

DECLARATION

We hereby declare that the Report entitled "**Cloud secure storage using blockchain system**" submitted for the award of Bachelor of technology Degree is our original work and the Report has not formed the basis for the award of any degree, diploma, associate ship or fellowship of similar other titles. It has not been submitted to any other University or Institution for the award of any degree or diploma.

Place: Anurag University, Hyderabad

M Sathvika Angel (20eg105630)

P Manikanta Reddy (20eg105654)

J. Sumanth (20eg105707)

Date:20-04-2024

CERTIFICATE

This is to certify that the Report entitled with "**Cloud secure storage using blockchain system"**.that is being submitted by **M Sathvika Angel**bearing roll number **20eg105630 ,P Manikanta Reddy** bearing roll number **20eg105654 and J Sumanth** bearing roll number **20eg105707** in partial fulfillment for the award of B.Tech in to the Anurag University is a record of bonafide work carried out by them under our guidance and supervision.

The results embodied in this Report have not been submitted to any other University or Institute for the award of any degree or diploma.

Dr. G. Vishnu Murthy	**Signature of Supervisor**
Head of the Department	**Dr A Jyothi**
DEPT OF CSE	**Assistance Professor**
	Department of CSE

ACKNOWLEDGMENT

we would like to express our sincere thanks and deep sense of gratitude to project supervisor Dr **A Jyothi** for her constant encouragement and inspiring guidance without which this project could not have been completed. Her critical reviews and constructive comments improved our grasp of the subject and steered to the fruitful completion of the work. her patience, guidance and encouragement made this project possible.

We would like express our special thanks to **Dr. V. Vijaya Kumar**, Dean School of Engineering , Anurag University, for their encouragement and timely support in our B.Tech program. We would like acknowledge our sincere gratitude for the support extended by **Dr. G. Vishnu Murthy** , Dean, Dept. of CSE ,Anurag University. We also express our deep sense of gratitude to **Dr. V V S S S Balaram** ,Acadamic co-ordinator ,**Dr .Pallam Ravi** , Project in-Charge , Dr **A Jyothi** .Project Co-ordinator and Project review committee members, whose research expertise and commitment to the highest standards continuously motivated us during the crucial stage in our project work.

ABSTRACT

The rapid development of cloud computing has been greatly aided by the popularity and power of cloud storage. Nonetheless, there are still serious security incidents that, as aconsequence of malicious attack and management negligence, cause widespread sensitive data exposures at the cloud storagetier. To preserve the privacy of cloud data, this study created(CSSM). By combining data dispersion with dispersed storage,CSSM was able to provide encrypted, chucked, and scattered storage, preventing data intrusions at the storage layer. In addition, CSSM used a system of management levels and combined user passwords with secret sharing to avoid any unauthorized access to cryptographic materials. The experimental results show the proposed method can successfully store enormous amounts of cloud data without introducing a substantial latency cost, and is also suitable for securing data at the storage layer from leaking.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLESLIST OF TABLES

LIST OF GRAPHS

1.INTRODUCTION

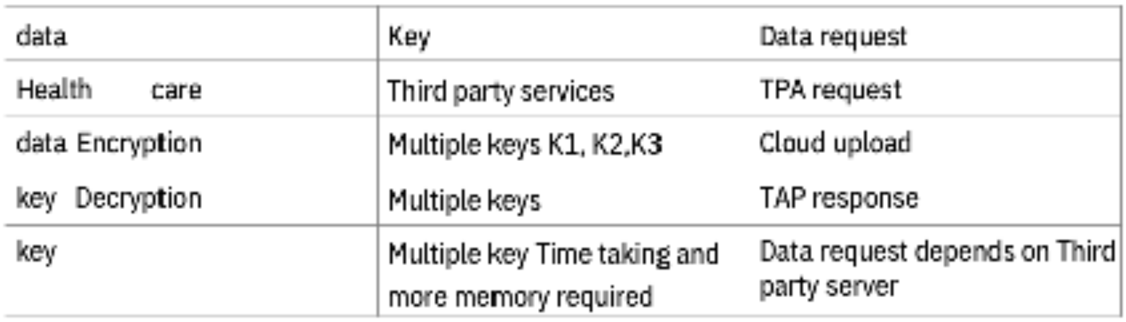
In recent decades, cloud computing has made major advancements. The main focus is on storage as a service, which forms the basis for several applications including pattern recognition, picture forensics, and forgery detection . A collection of tools to get you started is provided below. In the cloud, Amazon Web Service (AWS) has emerged as the de facto standard. One of the most well-liked cloud storage systems is Swift, a fundamental OpenStack component that complies with this standard. Although delivering useful services, the Open stack Swift method nevertheless faces several actual security vulnerabilities. According to the Cloud Security Alliance's top threat case analysis study issued in 2018, two-thirds of the incidents would result in customer data loss, mostly as a result of management incompetence and hostile assaults. For example, the OpenStack Swift mechanism often saves data in plaintext in its default configuration for performance reasons. As a result, unauthorised access to user data occurs at the storage layer. Additionally, the Swift mechanism may leak, according to the Open stack Vulnerability Management Team VMT's Security Report. M. Anwar Hossain was the assistant editor in responsibility of organizing the evaluation and approval of this article for publication. Because to security flaws, user data or configuration information may be compromised.To improve security and stop data leakage, a cloud-based data security storage method has been suggested for the Apache Spark framework. In order to protect user data in the cloud and stop data breaches during machine learning processes, several encryption techniques have been put into place. Yet, priorre search emphasizes the significance of safe key management strategies to guard against the compromise of cryptographic materials. Using the Hadoop system, Zeroes and colleagues have created a secure distributed storage solution that protects the privacy of cloud data through data dispersion and encryption. Before reading the data, this solution decrypts and assembles it. A key cache server is required, and all keys should only be kept in memory, in order to prevent key theft while utilizing this method. To achieve the high demands on big data storage, the cloud computing mechanism offers a solution because it provides controlled and flexible data processing and exchange mechanisms as well as their respective storage spaces . The increased interest has expanded in the field of health, including medical and research institutions and their cooperation. But despite the advantages that cloud computing offers, it lacks the functionality associated with data exchange due to the risks involved in exposing its content. For data proprietors, there’s a risk that the data collected will end in the hands of malevolent users. In this context, the fear of violating the regulations and the exploitation of data creates an atmosphere of mistrust that does not ensure the implementation of data exchange. Our major project, "Cloud Secure Storage Using Blockchain, " provides cloud storage security by implementing a two-layer defense – RSA encryption for data protection and blockchain system for decentralized access control, ensuring robust safeguarding against data breaches and unauthorized access.

1.1 PROBLEM STATEMENT

The existing secure distributed storage system based on Hadoop, aimed at preserving cloud data confidentiality through data dispersion and encryption, encounters a critical challenge in its key management strategy. Relying on a key cache server and storing all encryption keys in memory poses a significant security risk, potentially leading to key exposure and compromise. The vulnerability introduced by this approach, combined with the assumption of trusted third parties, raises concerns about the overall robustness and security of the system in real-world cloud storage scenarios. A more resilient key management solution is imperative to address these vulnerabilities and enhance the system's security posture.

1.2 PROBLEM ILLUSTRATION

Existing methods has two issues in cloud dispersion mainly more memory user as time taken for key generation from multiple servers.



data	Key	Data request
Health care	Third party services	TPA request
data Encryption key Decryption	Multiple keys K1, K2,K3 Multiple keys	Cloud upload TAP response
key	Multiple key Time taking and more memory required	Data request depends on Third party server

Table 1.2.1 problem illustration by table

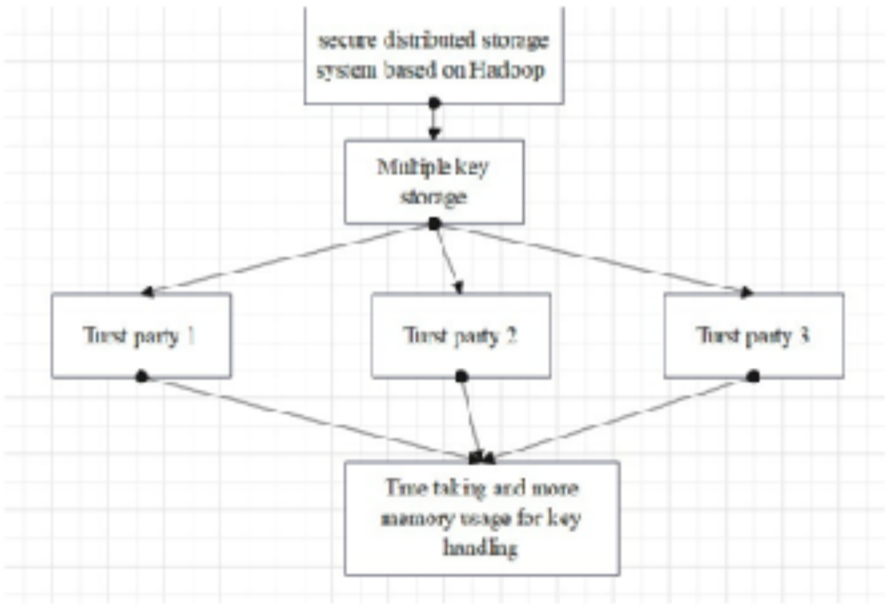

secure distributed storage system based on Hadoop
Multiple key storage
Trust party 1
Trust party 2
Trust party 3
Time taking and more memory usage for key handling

Fig 1.2.1 Problem illustration by flow map

1.3 OBJECTIVE

Traditional cloud storage methods can be susceptible to breaches and unauthorized access. This project aims to significantly improve security by implementing a two-layer defense system. By combining RSA encryption for data confidentiality and a blockchain system for decentralized access control, the project strives to create a robust shield against data breaches. Ultimately, this enhanced security aims to address the concerns around data exchange in healthcare, fostering trust and enabling secure collaboration and research.

2 LITERATURE SURVEY

Shah et al.proposed a cloud-oriented data security storage mechanism under the framework of Apache Spark, which prevents data leakage and improves the security of Apache Spark framework. constructed a secure distributed storagesystem based on Hadoop system, which keep the confiden-tiality of cloud data through data dispersion and encryption. It performs the data decryption and assembly tasks before reading data. To prevent the keys from being stolen, this method requires key cache server and all keys should bestowed in memory only. Some approaches [intro-duced independent third party to manage the key. It is assumed that third parties stay trusted. However, the assumption cloud not always exists in the real cloud storage environments .Wang et al. presented a data privacy preserving scheme for sensor-cloud system, based on edge computing and differential storage method. In this scheme, user data .

The use of the blockchain technology is relative concept in the research community. However, researchers have focused on the utilization of the most aspects of this innovative technology, and one of the most promising areas of research is the combination of the access control mechanisms with the blockchain.. They introduced smart contracts as a way to implement contextual access control restrictions and make authorization decisions. They also used blockchain to enforce access policies in dispersed situations where there is no central authority and to ensure that policies are enforced correctly and uniformly.

Also, Macías and Guitart proposed using blockchain technology as an access control tool for representing and transferring resource access rights from one user to another. They advocated storing the representation of these rights in the form of transactions on the blockchain. They also employed attribute-based access control (ABAC) policies, which combine a collection of rules expressing conditions over a set of attributes associated with the subject, resource, or environment. In addition, Uchibeke et al. in 2018 implemented identity-based access control (IBAC) and role-based access control (RBAC) on the Hyperledger Fabric blockchain, a private and permissioned scheme led by IBM, to achieve access control methods for big data (RBAC).

3.ANALYSIS

3.1 EXISTING SYSTEM

AES (Advanced Encryption Standard) has become the encryption algorithm of choice for governments, financial institutions, and security-conscious enterprises around the world. Because the computational requirements of this approach are low, AES can be used with consumer computing devices such as laptops and smartphones, as well as for quickly encrypting large amounts of data. For example, the IBM z14 mainframe series uses AES to enable pervasive encryption in which all the data in the entire system, whether at rest or in transit, is encrypted. AES is a symmetric algorithm which uses the same 128, 192, or 256 bit key for both encryption and decryption (the security of an AES system increases exponentially with key length). With even a 128-bit key, the task of cracking AES by checking each of the 2128 possible key values (a "brute force" attack) is so computationally intensive that even the fastest supercomputer would require, on average, more than 100 trillion years to do it.

3.1.1 DISADVANTAGES OF EXISTING SYSTEM:

A major issue with AES is that, as a symmetric algorithm, it requires that both the encryptor and the decrypt or use the same key. This gives rise to a crucial key management issue – how can that all-important secret key be distributed to perhaps hundreds of recipients around the world without running a huge risk of it being carelessly or deliberately compromised.

3.2 PROPOSED SYSTEM

In proposed system data is secured to store in cloud using two layer security block chian and encryption for encryption RSA algorithm is used and for Block chain hash bocks are generated with hash key. It is an asymmetric algorithm that uses a publicly known key for encryption, but requires a different key, known only to the intended recipient, for decryption. In this system, appropriately called public key cryptography (PKC), the public key is the product of multiplying two huge prime numbers together. Only that product, 1024, 2048, or 4096 bits in length, is made public. But RSA decryption requires knowledge of the two prime factors of that product. Because there is no known method of calculating the prime factors of such large numbers, only the creator of the public key can also generate the private key required for decryption.To overcome the challenge of key management and data memory usage in cloud secure records we use bock chain technology to reduce multiple key management and use encryption key and block chain to access data and use only cloud server to save data and secure with single key.

3.2.1 ADVANTAGES PROPOSED SYSTEM

Two layer authentication method is used for data security , Encryption and block chain system.

Providing encryption and authentication for e-mail and file storage applications across multiple platforms and it uses RSA algorithm for its key transportation.

3.3 SYSTEM REQUIREMENTS

3.3.1 HARDWARE REQUIREMENTS:

• System	: Pentium IV 2.4 GHz.
• Hard Disk	: 200 GB.
• Floppy Drive.	: 1.44 Mb.
• Monitor	: 15 VGA Colour.
• Ram	: 1 GB.

3.3.2 SOFTWARE REQUIREMENTS:

• Operating system	:	Windows XP/7/10.
• Coding Language	:	Java
• Tool	:	Netbeans
• Database	:	MYSQL
• Cloud	:	Drive HQ

4 IMPLEMENTATION

4.1MODULES:

4.1.1Owner model:

owner will register into the application by providing all the necessary details and therefore he can login into the application using username and password and user can upload the files to application and share with the other registered users. He can also view the files uploaded by him and can also view the requests for secret key and block chain key from the other users and we can respond and the key and block chain will be sent to user by mail. Using that key, he can download the file and view the information

4.1.2User Module:

user will register with application and get user name and password. Owner can see all encrypted files uploaded by all users and send request to respective user and get approval to download data and block chain hash and secuirty keys for RSA are shared to owner email which can be used for owner download.

4.1.3Cloud Module:

Using this module cloud can register with application and store information of each user and owner who uploads and requests for data. Details which are stored in cloud are upload to Drive hq cloud.

4.2 TECHNOLOGIES USED

4.2.1Java Technology

Java technology is both a programming language and a platform.

The Java Programming Language

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

- 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.

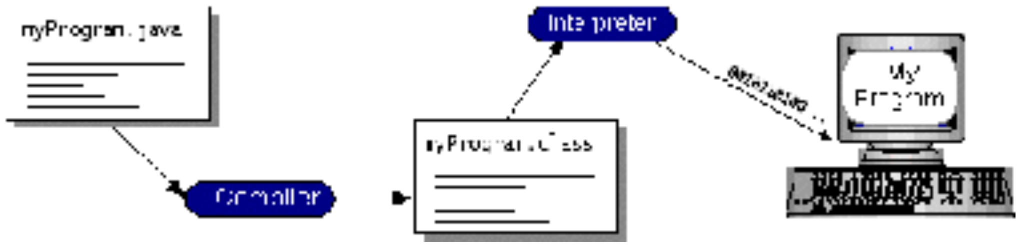


Fig 4.2.1.1 Illustration of how program works

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

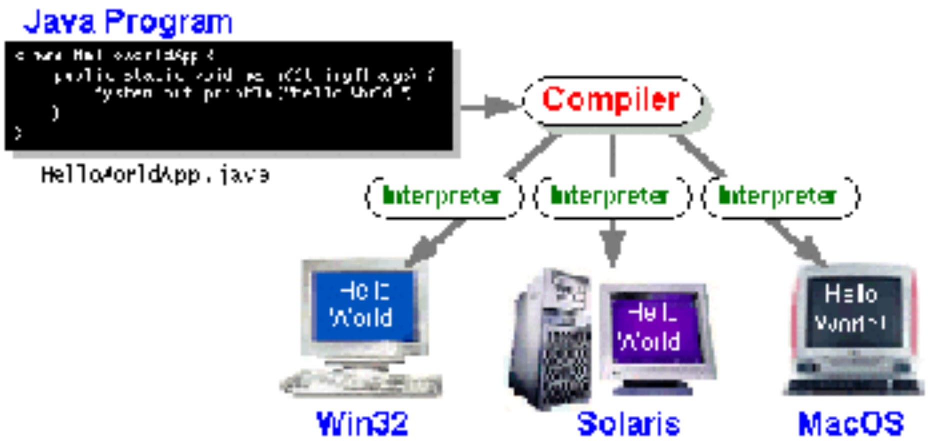


Fig 4.2.1.2 Working Java VM

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.

What Can Java Technology Do?

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

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

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

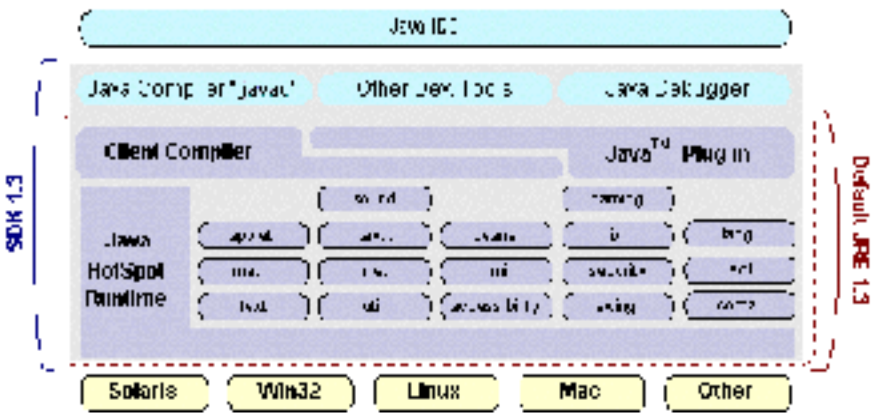


Fig 4.2.1.4 Java 2 SDK

Java Web Development

Web development is known as website development or web application development. The web development creates, maintains, and updates web development applications using a browser. This web development requires web designing, backend programming, and database management. The development process requires software technology.

Web development creates web applications using servers. We can use a web server or machine server like a CPU. The Web server or virtual server requires web application using technology. Web development requires server-side programming language or technology. Mostly Java, PHP, and other server-side languages require for web development.Java web development creates a server-side website and web application. The majority of Java web apps do not execute on the server directly. A web container on the server hosts Java web applications.

For Java web applications, the container acts as a runtime environment. What the Java Virtual Machine is for locally running Java applications, the container is for Java web applications. JVM is used to run the container itself.

Java distinguishes between two types of containers: web and Java EE. Additional functionality, such as server load distribution, can be supported by a container. A web container supports Java servlets and JSP (JavaServer Pages). In Java technology, Tomcat is a common web container.

Functions of Java Web Development

Java web development creates applications and websites using static and dynamic resources. The static resource refers to HTML pages with images, and a dynamic resource refers to classes, jars, Servlet, and JSP. Java web development uses several packages, files, and online links. Java web development requires web archive files known as a WAR files.

Java web development works on three main factors. These development factors show below

- Front-end web development using Java technology.
- Backend web development using Java server technology.
- Database management using Java database driver.

The above three factors create, update, remove, display and operate data or information.

Front-end web development: The front-end technology interacts with the user and Java interface. It helps to insert and submit data. Java web development uses JavaServer Pages or JSP for the front-end form or table.

Backend web development: The backend technology maintains and updates data of the database. Java uses Servlet, spring, and other advanced technology.

Database management handles or fetches data from the database using the Java database driver. The Java technology uses JDBC, Hibernate to handle the database.

Types of the Java Web Technologies

- Servlet API
- JSP (JavaServer page)
- JDBC Driver
- JAVA Persistence
- JavaServer Faces (JSF)
- JSTL
- JAVA Message Service API

Servlet API (JAVA Web application programming interface)

Servlet, filter, filter chain, servlet config, and other interfaces are available in the javax. Servlet package. The capabilities of servers that host apps are increased by using Servlet.

JSP (JavaServer Page Web application programming technology)

Developers employ JavaServer Pages or JSP technology to quickly produce platform- and server-independent online content. Normally, the developer works on separate Common Gateway Interface files to embed dynamic elements in HTML pages. Java JSP technology can be used, as it has access to the whole Java API family.

The JSP technology pieces code to control web information and moves dynamically. A JSP page comprises static data written in HTML, WML, XML, and other markup languages. Special JSP tags simplify Java code into HTML pages, making web development user-friendly.

JDBC Driver or Java Database Connectivity

JDBC Driver is a connector between database and Java web application. Java database connectivity helps to update and modify data using queries. The jdbc driver is an essential part of Java web development. This driver helps to send data to the database and retrieve data from the database.

Within a Java program, the JDBC driver allows to perform the following tasks:

- Make a data source connection
- To the data source, send queries and update statements
- Displays require data from a database.
- Organize application information.

JDBC is a set of methods and queries for accessing databases written in Java. Clients can use web applications using JDBC drivers to update any information in the database.

JDBC drivers connect to databases in four ways: JDBC-ODBC Bridge Driver, Network Protocol Driver, Native Driver, and Thin Driver.

Persistence API for Java

JDBC Goals

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

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

1. **SQL Level API**

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

2. **SQL Conformance**

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

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

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

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

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

5. **Keep it simple**

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

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

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

Technology of the JavaServer Faces

JavaServer Faces is called a JSF Technology. This technology provides a framework for developing web-based interfaces. JSF provides a simple model for components in various scripting or markup languages.

The data sources and server-side event handlers are coupled to the User Interface widgets. JSF aids in the creation and maintenance of web applications by minimizing the time and effort required.

◦ Construct Java web development pages.
◦ Drop components on a web page by adding component tags to a web page.
◦ Connect Java web development page components to server-side data.
◦ Connect component-generated events to application code running on the server.
◦ Extend the life of server requests by storing and restoring the application state.

4.2.2 CRYPTOGRAPHY

Cryptography is technique of securing information and communications through use of codes so that only those person for whom the information is intended can understand it and process it. Thus preventing unauthorized access to information. The prefix “crypt” means “hidden” and suffix “graphy” means “writing”. In Cryptography the techniques which are use to protect information are obtained from mathematical concepts and a set of rule based calculations known as algorithms to convert messages in ways that make it hard to decode it. These algorithms are used for cryptographic key generation, digital signing, verification to protect data privacy, web browsing on internet and to protect confidential transactions such as credit card and debit card transactions.

Techniques used For Cryptography: In today’s age of computers cryptography is often associated with the process where an ordinary plain text is converted to cipher text which is the text made such that intended receiver of the text can only decode it and hence this process is known as encryption. The process of conversion of cipher text to plain text this is known as decryption.

Features Of Cryptography are as follows:

Confidentiality: Information can only be accessed by the person for whom it is intended and no other person except him can access it.

Integrity: Information cannot be modified in storage or transition between sender and intended receiver without any addition to information being detected.

Non-repudiation: The creator/sender of information cannot deny his intention to send information at later stage.

Authentication: The identities of sender and receiver are confirmed. As well as destination/origin of information is confirmed.

Types Of Cryptography: In general there are three types Of cryptography:

Symmetric Key Cryptography: It is an encryption system where the sender and receiver of message use a single common key to encrypt and decrypt messages. Symmetric Key Systems are faster and simpler but the problem is that sender and receiver have to somehow exchange key in a secure manner. The most popular symmetric key cryptography system are Data Encryption System(DES) and Advanced Encryption System(AES).

Hash Functions: There is no usage of any key in this algorithm. A hash value with fixed length is calculated as per the plain text which makes it impossible for contents of plain text to be recovered. Many operating systems use hash functions to encrypt passwords.

Asymmetric Key Cryptography: Under this system a pair of keys is used to encrypt and decrypt information. A receiver's public key is used for encryption and a receiver's private key is used for decryption. Public key and Private Key are different. Even if the public key is known by everyone the intended receiver can only decode it because he alone know his private key. The most popular asymmetric key cryptography algorithm is RSA algorithm.

Applications Of Cryptography:

Computer passwords: Cryptography is widely utilized in computer security, particularly when creating and maintaining passwords. When a user logs in, their password is hashed and compared to the hash that was previously stored. Passwords are hashed and encrypted before being stored. In this technique, the passwords are encrypted so that even if a hacker gains access to the password database, they cannot read the passwords.

Digital Currencies: To safeguard transactions and prevent fraud, digital currencies like Bitcoin also use cryptography. Complex algorithms and cryptographic keys are used to safeguard transactions, making it nearly hard to tamper with or forge the transactions.

Secure web browsing: Online browsing security is provided by the use of cryptography, which shields users from eavesdropping and man-in-the-middle assaults. Public key cryptography is used by the Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols to encrypt data sent between the web server and the client, establishing a secure channel for communication.

Electronic signatures: Electronic signatures serve as the digital equivalent of a handwritten signature and are used to sign documents. Digital signatures are created using cryptography and can be validated using public key cryptography. In many nations, electronic signatures are enforceable by law, and their use is expanding quickly.

Authentication: Cryptography is used for authentication in many different situations, such as when accessing a bank account, logging into a computer, or using a secure network. Cryptographic methods are employed by authentication protocols to confirm the user's identity and confirm that they have the required access rights to the resource.

Cryptocurrencies: Cryptography is heavily used by cryptocurrencies like Bitcoin and Ethereum to safeguard transactions, thwart fraud, and maintain the network's integrity. Complex algorithms and cryptographic keys are used to safeguard transactions, making it nearly hard to tamper with or forge the transactions.

End-to-End Encryption: End-to-end encryption is used to protect two-way communications like video conversations, instant messages, and email. Even if the message is encrypted, it assures that only the intended receivers can read the message. End-to-end encryption is widely used in communication apps like WhatsApp and Signal, and it provides a high level of security and privacy for users

4.2.2.1 RSA

The RSA name is given by their inventors which is used to encrypt the text with high security. The RSA technique is one of the most used techniques to encrypt text, as it is the asymmetric encryption algorithm. It encrypts the text by utilizing the mathematical properties of the prime numbers.

In the RSA algorithm, the sender and receiver have private keys. Also, a common public key exists, which the sender shares with the receiver. The sender encrypts the plain text using their own public and private key, and the receiver decrypts the message using their private and public common key.

RSA SAMPLE CODE

```
import java.math.*;
import java.util.*;
public class Main {
  public static int getGCD(int mod, int num) {
    // If the mod is zero, return the num
    if (mod == 0)
      return num;
    else
      // recursive function call
      return getGCD(num % mod, mod);
  }
  public static void main(String args[]) {
    int d = 0, e; // Intialization
    int message = 32; // number message
    int prime1 = 5; // 1st prime number p
    int prime2 = 7; // 2nd prime number q
    int primeMul = prime1 * prime2; // performing operations
    int primeMul1 = (prime1 - 1) * (prime2 - 1);
```

```
System.out.println("primeMul1 is equal to : " + primeMul1 + "\n");
// Finding the valid public key
for (e = 2; e < primeMul1; e++) {
  // Here e is a public key
  if (getGCD(e, primeMul1) == 1) {
    break;
  }
}
// Printing the public key
System.out.println("Public key e is = " + e);
// Calculating the private key
for (int m = 0; m <= 9; m++) {
  // get the value of temp
  int temp = 1 + (m * primeMul1);
  // private key
  if (temp % e == 0) {
    d = temp / e;
    break;
  }
}
System.out.println("d is : " + d);
double cipher;
BigInteger d_message;
// getting the cipher text
cipher = (Math.pow(message, e)) % primeMul;
System.out.println("Cipher text is : " + cipher);
// Int to BigInteger
BigInteger bigN = BigInteger.valueOf(primeMul);
// Float to bigINt
BigInteger bigC = BigDecimal.valueOf(cipher).toBigInteger();
// decrypting the message
```

4.2.3 BLOCKCHAIN SYSTEM

Blockchain technology is a database mechanism that stores data in blocks linked together in a chain. Each block contains a timestamp, transaction data, and a cryptographic hash of the previous block. The blocks are linked together to form a chain that is chronologically consistent because the chain cannot be modified or deleted without consensus from the network.

4.2.4 CLOUD

Imagine a vast digital warehouse, accessible from anywhere. That's cloud data storage in a nutshell. Instead of bulky hard drives, your files reside on remote servers managed by companies like Google Drive or Dropbox.

This remote access is key. With a web browser or app, you can upload, download, edit, and share documents, photos, videos – anything digital – from any device. No more lugging around external drives!

Cloud storage offers scalability too. Need more space? Simply upgrade your plan. No need to constantly buy new hardware. Businesses love this flexibility, easily storing massive datasets or collaborating on projects in real-time.

Security is paramount. Cloud providers employ robust encryption and access controls to keep your data safe from prying eyes. Automatic backups ensure peace of mind – even if your local device fails, your precious files are secure in the cloud.

But the cloud isn't perfect. A strong internet connection is crucial for accessing your data. Outages or slow speeds can be frustrating. Additionally, depending on the amount of data stored and the plan chosen, costs can add up.

For businesses, choosing the right cloud provider is vital. Public clouds offer flexibility and affordability, but some companies might prefer the increased control of a private cloud. Hybrid solutions combine both for a customized approach.

Overall, cloud data storage is a game-changer. It's convenient, secure, and scalable, making it a powerful tool for individuals and businesses alike. As technology evolves,

Fig 4.2.4.1 Cloud computing

cloud storage will continue to shape the way we store, access, and manage our ever-growing digital footprint.

4.2.4.1 DRIVE HQ

DriveHQ stands out as a veteran provider with a strong focus on business needs. Established in 2003, they offer a secure and feature-rich file server accessible from anywhere via web browsers and mobile apps. Their core strength lies in keeping your data safe, utilizing encryption and access controls to give you peace of mind. But DriveHQ goes beyond simple storage, offering automatic backups, controlled file sharing, and even cloud-to-cloud backup solutions for added redundancy. For businesses with specific file transfer needs, DriveHQ even provides FTP server hosting. One of the best things about DriveHQ is its focus on enterprise functionality. They offer user and group management, allowing for easy control over access, and integrate seamlessly with Active Directory for existing business infrastructure. This, combined with support for large-scale deployments, makes DriveHQ a compelling option for companies seeking a robust cloud storage solution.

4.3 SAMPLE CODE

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="utf-8">

  <meta content="width=device-width, initial-scale=1.0" name="viewport">

  <title>Cloud Dispersion</title>

  <meta content="" name="description">

  <meta content="" name="keywords">

  <!-- Favicons -->

  <link href="assets/img/favicon.png" rel="icon">

  <link href="assets/img/apple-touch-icon.png" rel="apple-touch-icon">

  <!-- Google Fonts -->

  <link href="https://fonts.googleapis.com/css?
family=Open+Sans:300,300i,400,400i,600,600i,700,700i|
Raleway:300,300i,400,400i,500,500i,600,600i,700,700i|
Poppins:300,300i,400,400i,500,500i,600,600i,700,700i" rel="stylesheet">

  <!-- Vendor CSS Files -->

  <link href="assets/vendor/aos/aos.css" rel="stylesheet">

  <link href="assets/vendor/bootstrap/css/bootstrap.min.css" rel="stylesheet">

  <link href="assets/vendor/bootstrap-icons/bootstrap-icons.css" rel="stylesheet">

  <link href="assets/vendor/boxicons/css/boxicons.min.css" rel="stylesheet">

  <link href="assets/vendor/glightbox/css/glightbox.min.css" rel="stylesheet">

  <link href="assets/vendor/swiper/swiper-bundle.min.css" rel="stylesheet">

  <!-- Template Main CSS File -->

  <link href="assets/css/style.css" rel="stylesheet">
```

```
 <!--  * Template Name: Cloud Dispersion

  * Updated: Sep 18 2023 with Bootstrap v5.3.2

  * Template URL: https://bootstrapmade.com/Cloud Dispersion-free-onepage-
bootstrap-theme/

  * Author: BootstrapMade.com

  * License: https://bootstrapmade.com/license/

 </head>

<body>

  <!-- ======= Header ======= -->

  <header id="header" class="fixed-top d-flex align-items-center">

    <div class="container d-flex justify-content-between">

      <div class="logo">

        <h1><a href="index.html">Cloud Dispersion</a></h1>

        <!-- Uncomment below if you prefer to use an image logo -->

        <!-- <a href="index.html"><img src="assets/img/logo.png" alt="" class="img-
fluid"></a>-->

      </div>



      <nav id="navbar" class="navbar">

        <ul>

          <li><a class="nav-link scrollto " href="index.html">Home</a></li>

          <li><a class="nav-link scrollto " href="owner.jsp">Owner</a></li>

          <li><a class="nav-link scrollto active" href="cloud.jsp">Cloud</a></li>

          <li><a class="nav-link scrollto " href="user.jsp">User</a></li>

        </ul>
        <i class="bi bi-list mobile-nav-toggle"></i>

      </nav><!-- .navbar -->

    </div>

  </header><!-- End Header -->

  <!-- ======= Hero Section ======= -->

  <section id="hero" class="d-flex flex-column justify-content-center align-items-
center">

    <div class="container text-center text-md-left" data-aos="fade-up"

         <center><br>

                         <h6 style="color:white">Cloud Login</h6><br>



                         <form style="width: 20%;margin-bottom:  40%;"
action="cact.jsp">

  <!-- Email input -->

  <div class="form-outline mb-4">

     <input type="text" name="username" id="form2Example1"
placeholder="Username" required="" class="form-control" />

  </div>

  <!-- Password input -->

  <div class="form-outline mb-4">

     <input type="password" name="password" id="form2Example2"
placeholder="password" required="" class="form-control" />

  </div>

  <!-- 2 column grid layout for inline styling -->

  <!-- Submit button -->

 <button type="submit" class="btn btn-primary">Submit</button>
```

5 SYSTEM DESIGN

System Design Introduction:

The System Design Document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

5.1 SYSTEM ARCHITECTURE

Architecture Flow:

Below architecture diagram represents mainly flow of request from the users to database through servers. In this scenario overall system is designed in three tiers separately using three layers called presentation layer, business layer, data link layer. This project was developed using 3-tier architecture.

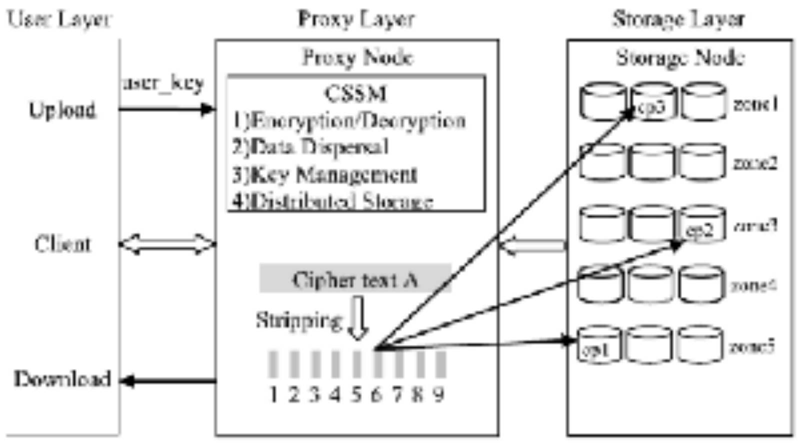


Figure 5.1.1: Architecture diagram

3-Tier Architecture:

The three-tier software architecture (a three layer architecture) emerged in the 1990s to overcome the limitations of the two-tier architecture. The third tier (middle tier server) is between the user interface (client) and the data management (server) components. This middle tier provides process management where business logic and rules are executed and can accommodate hundreds of users (as compared to only 100 users with the two tier architecture) by providing functions such as queuing, application execution, and database staging.

The three tier architecture is used when an effective distributed client/server design is needed that provides (when compared to the two tier) increased performance, flexibility, maintainability, reusability, and scalability, while hiding the complexity of distributed processing from the user. These characteristics have made three layer architectures a popular choice for Internet applications and net-centric information systems

Advantages of Three-Tier:

- Separates functionality from presentation.
- Clear separation – better understanding.
- Changes limited to well define components.
- Can be running on WWW.
- Effective network performance.

5.2 UML DIAGRAMS

Global Use Case Diagrams:

Identification of actors:

Actor: Actor represents the role a user plays with respect to the system. An actor interacts with, but has no control over the use cases.

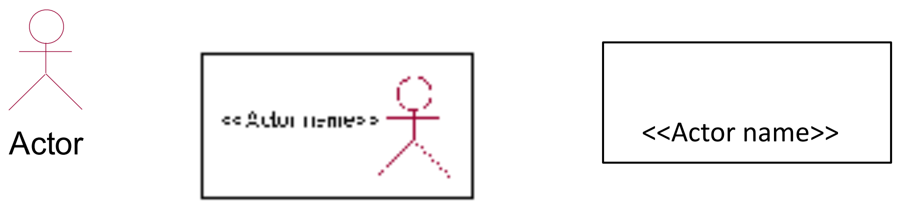


Fig 5.2.1 Graphical representation:

An actor is someone or something that:

Interacts with or uses the system.

- Provides input to and receives information from the system.
- Is external to the system and has no control over the use cases.

Actors are discovered by examining:

- Who directly uses the system?
- Who is responsible for maintaining the system?
- External hardware used by the system.
- Other systems that need to interact with the system.

Questions to identify actors:

- Who is using the system? Or, who is affected by the system? Or, which groups need help from the system to perform a task?
- Who affects the system? Or, which user groups are needed by the system to perform its functions? These functions can be both main functions and secondary functions such as administration.
- Which external hardware or systems (if any) use the system to perform tasks?
- What problems does this application solve (that is, for whom)?
- And, finally, how do users use the system (use case)? What are they doing with the system?

The actors identified in this system are:

a. System Administrator
b. Customer
c. Customer Care

Identification of usecases:

Usecase: A use case can be described as a specific way of using the system from a user's (actor's) perspective.

Graphical representation:

A more detailed description might characterize a use case as:

- Pattern of behavior the system exhibits
- A sequence of related transactions performed by an actor and the system
- Delivering something of value to the actor

Use cases provide a means to:

- capture system requirements
- communicate with the end users and domain experts
- test the system

Use cases are best discovered by examining the actors and defining what the actor will be able to do with the system.

5.3 **Flow of Events**

A flow of events is a sequence of transactions (or events) performed by the system. They typically contain very detailed information, written in terms of what the system should do, not how the system accomplishes the task. Flow of events are created as separate files or documents in your favorite text editor and then attached or linked to a use case using the Files tab of a model element.

A flow of events should include:

- When and how the use case starts and ends
- Use case/actor interactions
- Data needed by the use case
- Normal sequence of events for the use case
- Alternate or exceptional flows

5.3.1Construction of Usecase diagrams:

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

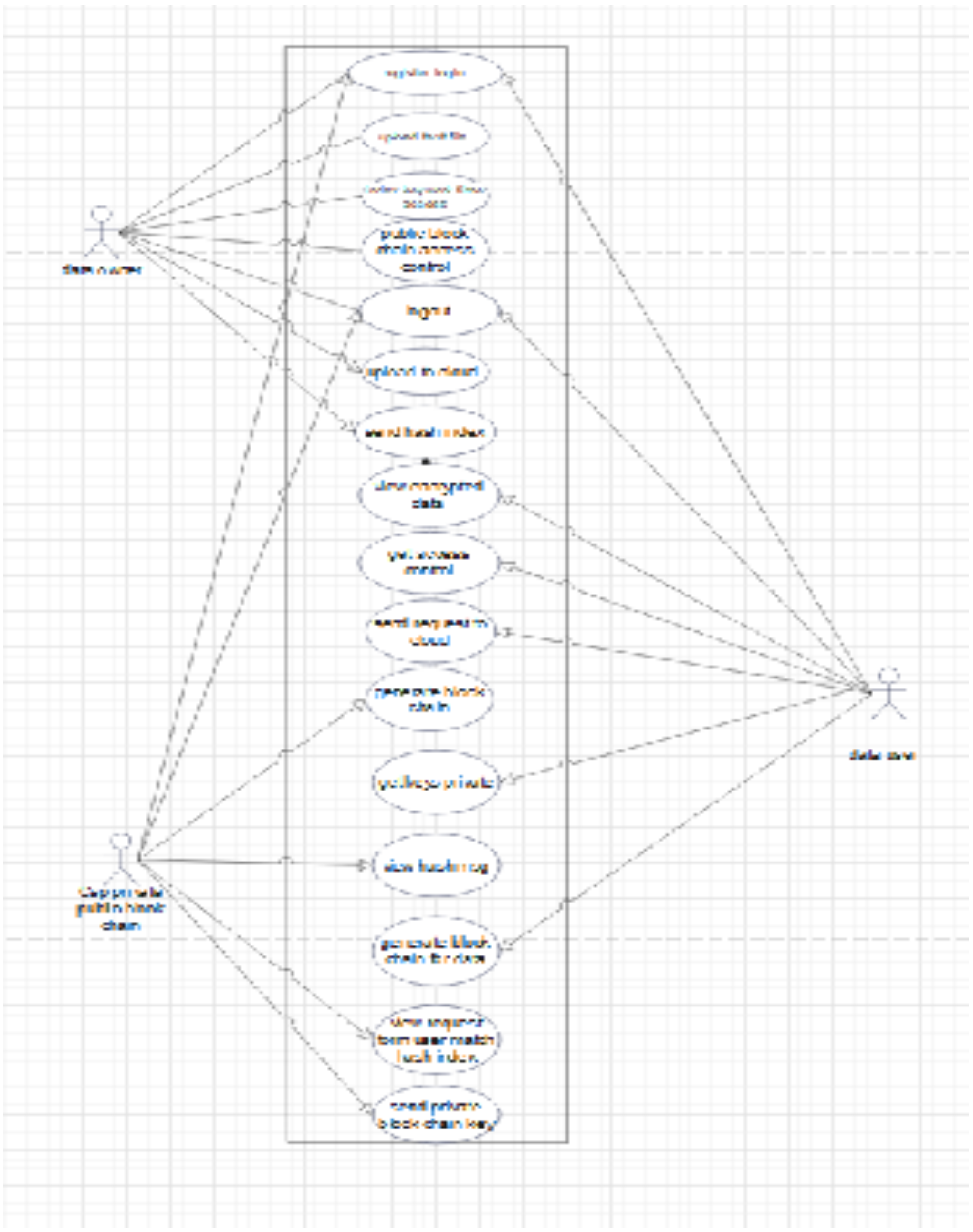


Figure 5.3.1.1 Use Case Diagram

5.3.2 SEQUENCE DIAGRAMS:

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

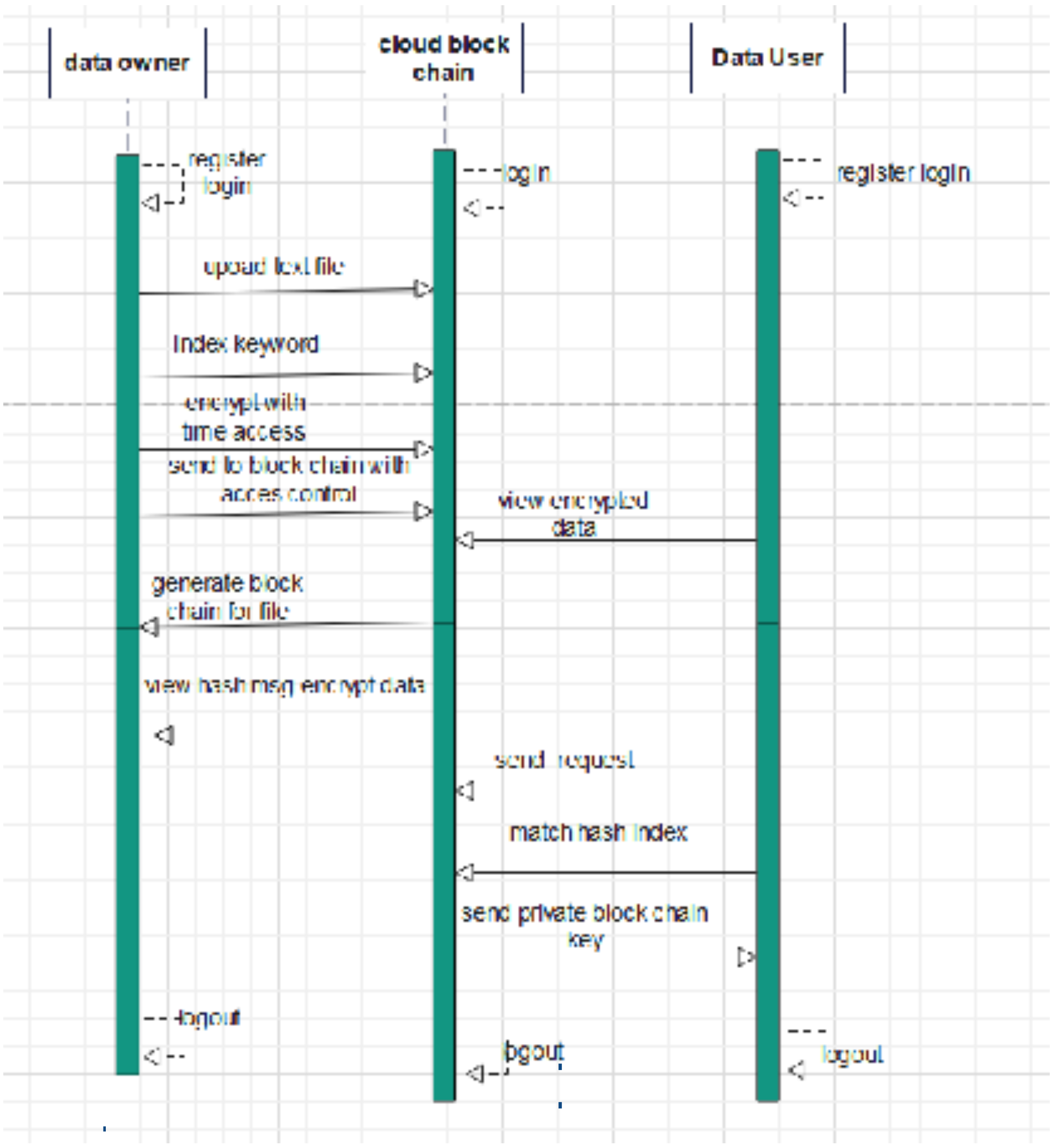


Figure 5.3.2.1 Sequence diagram

5.3.3. CLASS DIAGRAM:

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

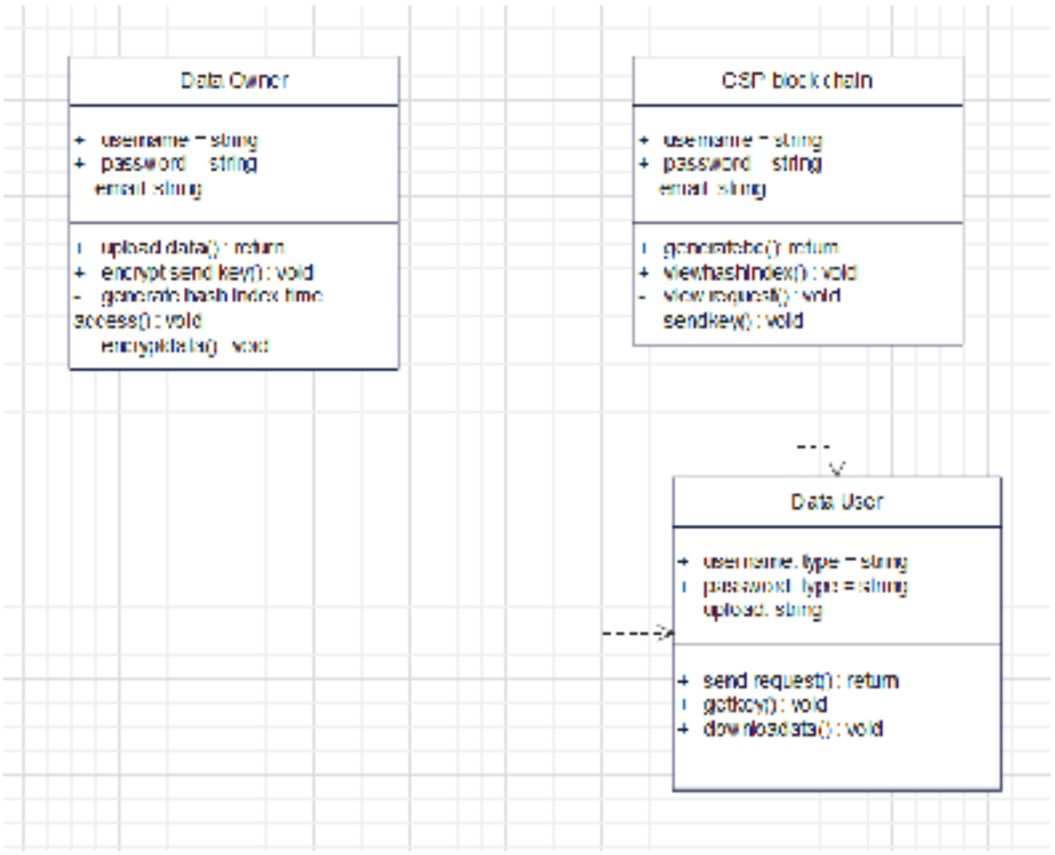


Figure 5.3.3.1: Class Diagram

5.3.4 ACTIVITY DIAGRAM:

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.

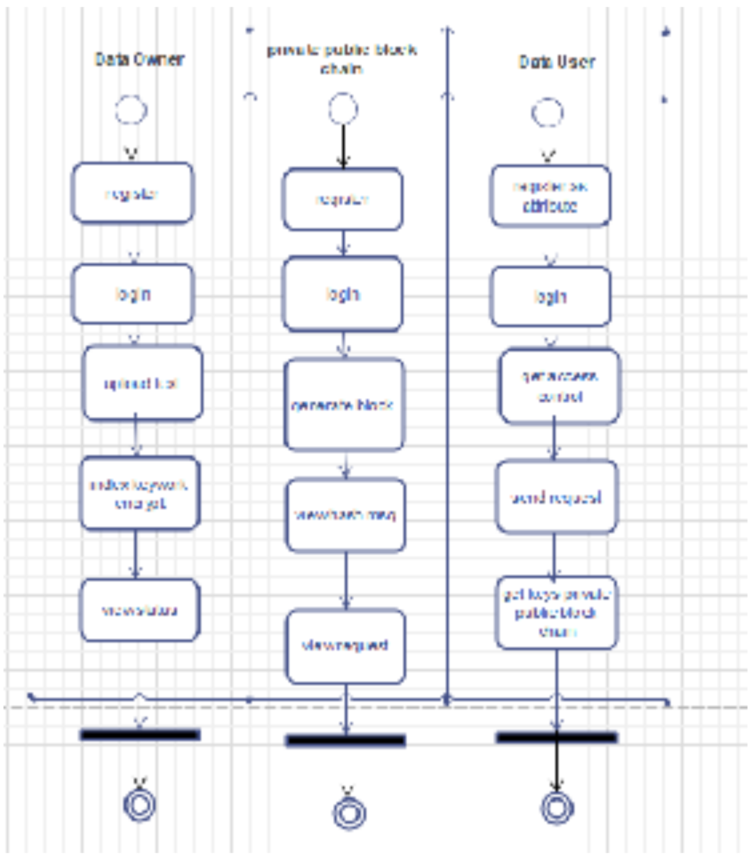


Figure 5.3.4.1: Activity Diagram

6 EXPERIMENT RESULTS

6.1 INTRODUCTION:

Testing is the debugging program is one of the most critical aspects of the computer programming triggers, without programming that works, the system would never produce an output of which it was designed. Testing is best performed when user development is asked to assist in identifying all errors and bugs. The sample data are used for testing. It is not quantity but quality of the data used the matters of testing. Testing is aimed at ensuring that the system was accurately an efficiently before live operation commands.

Testing objectives:

The main objective of testing is to uncover a host of errors, systematically and with minimum effort and time. Stating formally, we can say, testing is a process of executing a program with intent of finding an error.

1. A successful test is one that uncovers an as yet undiscovered error.
2. A good test case is one that has probability of finding an error, if it exists.
3. The test is inadequate to detect possibly present errors.
4. The software more or less confirms to the quality and reliable standards.

6.2. Levels of Testing:

Code testing:

This examines the logic of the program. For example, the logic for updating various sample data and with the sample files and directories were tested and verified.

Specification Testing:

Executing this specification starting what the program should do and how it should performed under various conditions. Test cases for various situation and combination of conditions in all the modules are tested.

Unit testing:

In the unit testing we test each module individually and integrate with the overall system. Unit testing focuses verification efforts on the smallest unit of software design in the module. This is also known as module testing. The module of the system is tested separately. This testing is carried out during programming stage itself. In the testing step each module is found to work satisfactorily as regard to expected output from the module. There are some validation checks for fields also. Each Module can be tested using the following two Strategies:

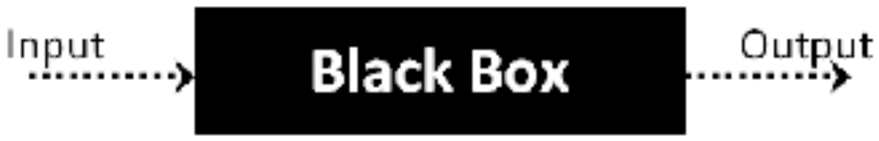
1. Black Box Testing
2. White Box Testing

BLACK BOX TESTING

What is Black Box Testing?

Black box testing is a software testing techniques in which functionality of the software under test (SUT) is tested without looking at the internal code structure, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on the software requirements and specifications.

In Black Box Testing we just focus on inputs and output of the software system without bothering about internal knowledge of the software program.



The above Black Box can be any software system you want to test. For example : an operating system like Windows, a website like Google ,a database like Oracle or even your own custom application. Under Black Box Testing , you can test these applications by just focusing on the inputs and outputs without knowing their internal code implementation.

Black box testing - Steps

Here are the generic steps followed to carry out any type of Black Box Testing.

- Initially requirements and specifications of the system are examined.

- Tester chooses valid inputs (positive test scenario) to check whether SUT processes them correctly. Also some invalid inputs (negative test scenario) are chosen to verify that the SUT is able to detect them.
- Tester determines expected outputs for all those inputs.
- Software tester constructs test cases with the selected inputs.
- The test cases are executed.
- Software tester compares the actual outputs with the expected outputs.
- Defects if any are fixed and re-tested.

WHITE BOX TESTING

White Box Testing is the testing of a software solution's internal coding and infrastructure.It focuses primarily on strengthening security, the flow of inputs and outputs through the application, and improving design and usability.White box testing is also known as clear, open, structural, and glass box testing.

It is one of two parts of the "box testing" approach of software testing. Its counter-part, blackbox testing, involves testing from an external or end-user type perspective. On the other hand, Whitebox testing is based on the inner workings of an application and revolves around internal testing. The term "whitebox" was used because of the see-through box concept.

What do you verify in White Box Testing ?

White box testing involves the testing of the software code for the following:

- Internal security holes
- Broken or poorly structured paths in the coding processes
- The flow of specific inputs through the code
- Expected output
- The functionality of conditional loops
- Testing of each statement, object and function on an individual basis

How do you perform White Box Testing?

To give you a simplified explanation of white box testing, we have divided it into two basic steps.

STEP 1) UNDERSTAND THE SOURCE CODE

The first thing a tester will often do is learn and understand the source code of the application. Since white box testing involves the testing of the inner workings of an application, the tester must be very knowledgeable in the programming languages used in the applications they are testing. The tester should be able to find security issues and prevent attacks from hackers and naive users who might inject malicious code into the application either knowingly or unknowingly.

Step 2) CREATE TEST CASES AND EXECUTE

The second basic step to white box testing involves testing the application's source code for proper flow and structure. One way is by writing more code to test the application's source code. The tester will develop little tests for each process or series of processes in the application.

6.3 PRAMETERS CONSIDERED

File Download:

Parameter: Secure File Transfer Protocol

Formula: Transmit(DecryptedData) - Implement a secure protocol for downloading the file.

FileUpload :

Data Encryption:

Parameter: Encryption Algorithm (e.g., AES)

Formula: EncryptedData = Encrypt(FileData, EncryptionKey)

Audit Trail:

Parameter: Logging Mechanism

Formula: LogEntry = CreateLogEntry(User, Action, Timestamp)

Preferred Max Block Size: Measured in kilobytes, this is the size at which the ordering service creates a new block regardless of other factors

Max Block Size = Size (Ordering service)

7 DISCUSSION OF RESULTS

7.1 Test Cases

Below stable shows the test case for the admin. The details if admin are stored in the database such as admin name, admin name etc. Admin is responsible for handling the transactions. The admin sets password and user name. Weather all the passwords and usernames are correct not is checked. Because by using these details the user will login in into the cloud and can view the details of file stored.

Test case 1

The test case 1 tests the login of admin’s. The passwords and username is given if the correct password and usernames are entered the login will be successful. If any wrongin passwords and username the login will be denied.

Sl # Test Case : -	UTC-1
Name of Test: -	**owner login**
Items being tested: -	**Validation for owner login**
Sample Input: -	**Fill form**
Expected output: -	**Details stored in database if wrong details are given check validation**
Actual output: -	**Validation verified details stored in db**
Remarks: -	Pass.

Table 7.1.1 Test Case1

Test case 2

The test case 2 is for checking the users. Again for the user’s password and user name is given based on only the correct input the users will be able to login

Sl # Test Case : -	UTC-2
Name of Test: -	**upload data to cloud**
Items being tested: -	**Data encrypted or not**
Sample Input: -	**Text file**
Expected output: -	**encryption key generated and stored in cloud**
Actual output: -	**encryption can be viewed by server**
Remarks: -	sucess.

Table 7.1.2 Test Case2

Test case 3

The test case 3 shows for the file upload and it is being stored in different cloud which is called block generation and storing and hence this file is divided and stored .If the storage and block generation is not successful than we can store the file

Sl # Test Case : -	ITC-1
Name of Test: -	**File request**
Item being tested: -	**Request sent to owner**
Sample Input: -	**Encrypted Text data**
Expected output: -	**Data stored in server, and database with confirmation**

Actual output: -	**Request sent success.**
Remarks: -	Pass.

Table 7.1.3 Test Case 3

Test case 4

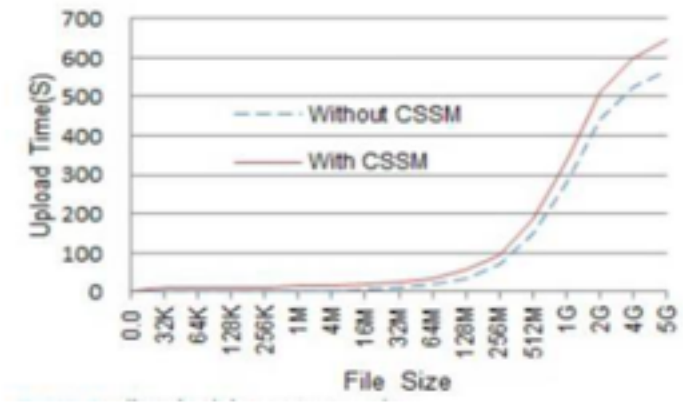
Test case 4 is for file upload where the uploaded is downloaded. During downloadif any one of the cloud fails the recovery should start automatic. If the recovery is successful all the blocks will be generated and merged together.

Sl # Test Case : -	STC-1
Name of Test: -	**receiver completed download**
Item being tested: -	**Check tasks upload by cloud and complete download**
Sample Input: -	**Encrypted data**
Expected output: -	**Data decrypted send to receiver**
Actual output: -	**receivers can download data**
Remarks: -	Pass

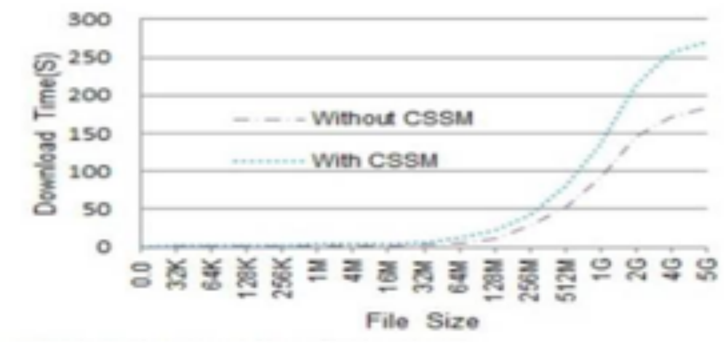
Table 7.1.4 Test Case 4

7.2 PERFORMANCE COMPARISION

We examine the time overhead in Swift system upload and download operations with and without CSSM in order to assess CSSM's performance. Le sizes typically vary from 32KB to 5GB. We make 10 tries to upload and download the same data because the time needed for each operation varies, and the time taken is the average. It takes approximately the same amount of time to download and upload. The increased time needed for upload and download activities is depicted in The outcomes of the experiment allow us to deduce the following conclusions.



Graph 7.2.1 File Upload Comparison with and without CSSM



Graph 7.2.2 File Dwonload Comparison with and without CSSM

CSSM'S EFFECT ON SYSTEM OVERHEAD

The test results show that as file size rises, raising the CSSM shortens the amount of time needed for uploading and downloading. Yet, as compared to upload operations, download operations have additional runtime overhead. For files with a size of 5G, the upload operation took 78 seconds longer and the download operation took 85 seconds longer when comparing the increased working time between the two points in time.

7.3 RESULT SCREEN SHOTS

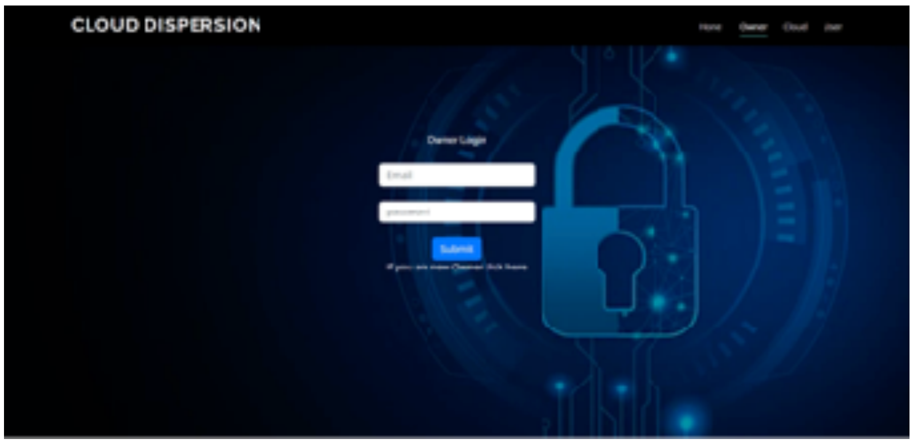


Fig 7.3.1 Owner Login page

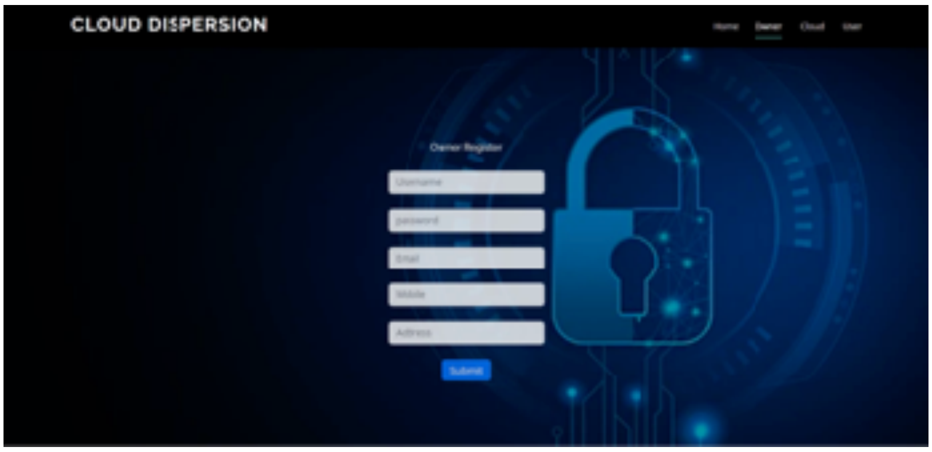


Fig 7.3.2 Owner Registration page

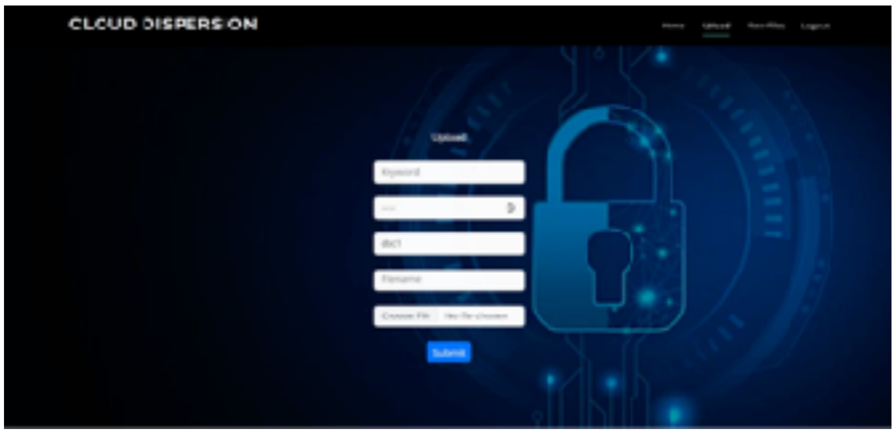
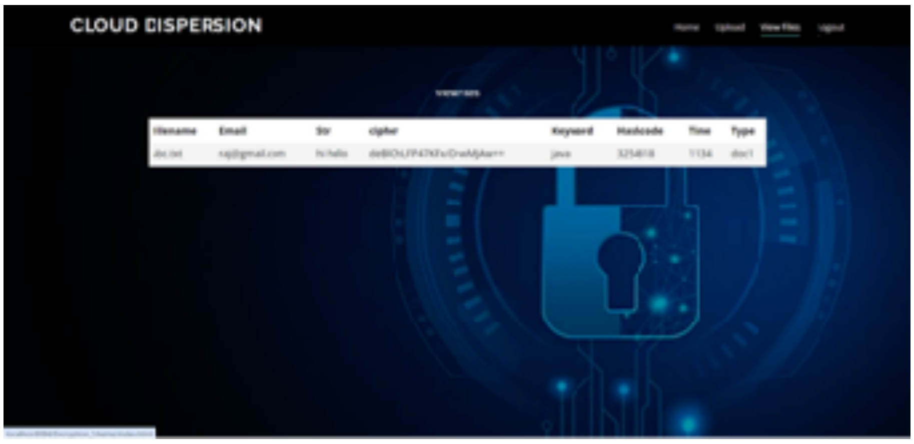


Fig 7.3.3 Owner file upload page



Filename	Email	Str	cipher	Keyword	Hashcode	Time	Type
abc.txt	raj@gmail.com	hi hello	deBIOhJFP47KFx/DraMjAw==	java	325418	1134	doc1

Fig 7.3.4 All uploaded documents

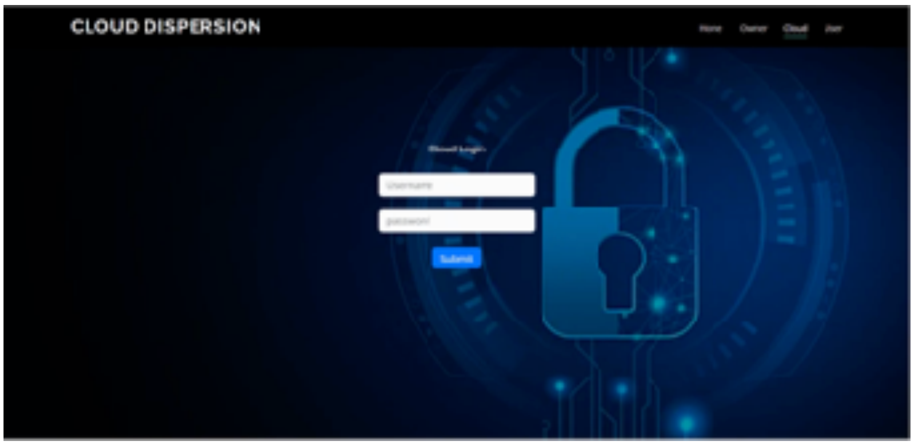
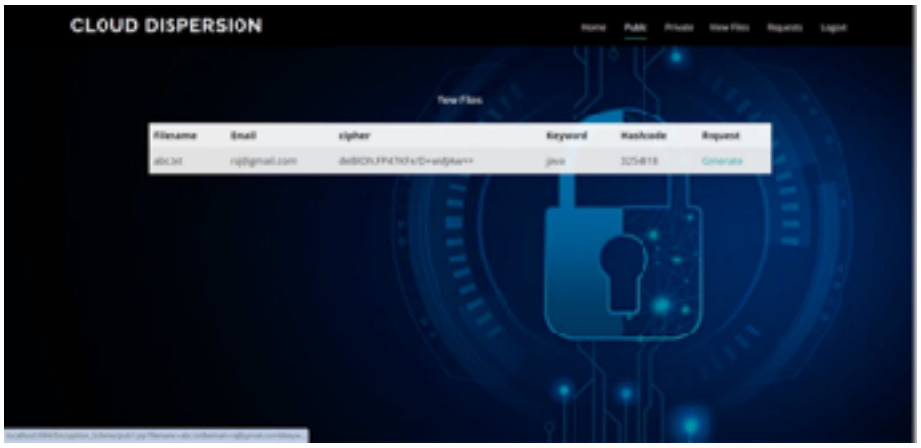


Fig 7.3.5 Cloud login page



Filename	Email	cipher	Keyword	Hashcode	Request
abc.txt	raj@gmail.com	deBIOhJFP47KFx/D+widjAw==	java	325418	Generate

Fig 7.3.6 Cloud page

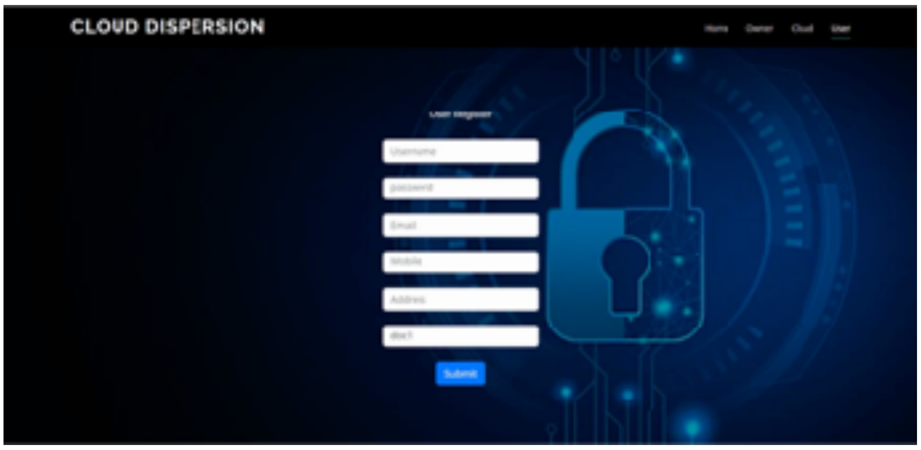


Fig 7.3.7 User registration page

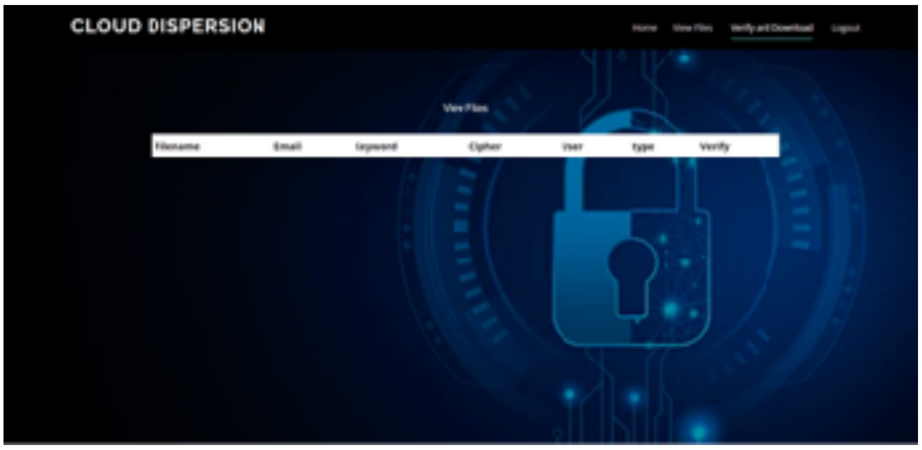
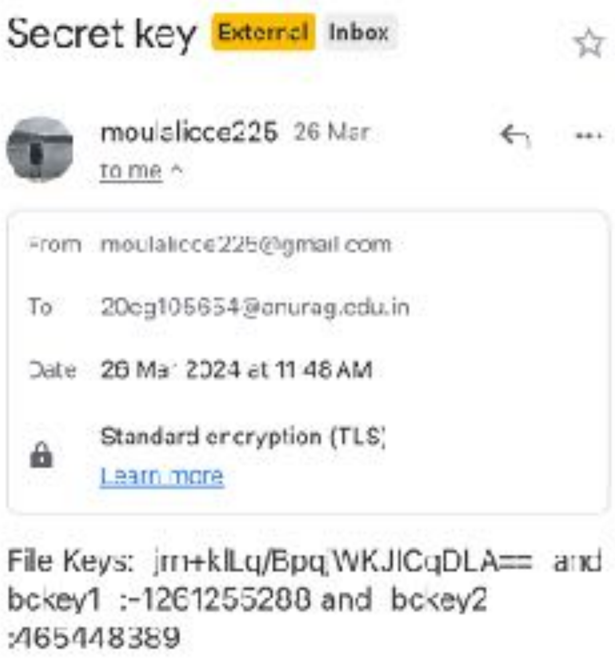


Fig 7.3.8 Document Requested

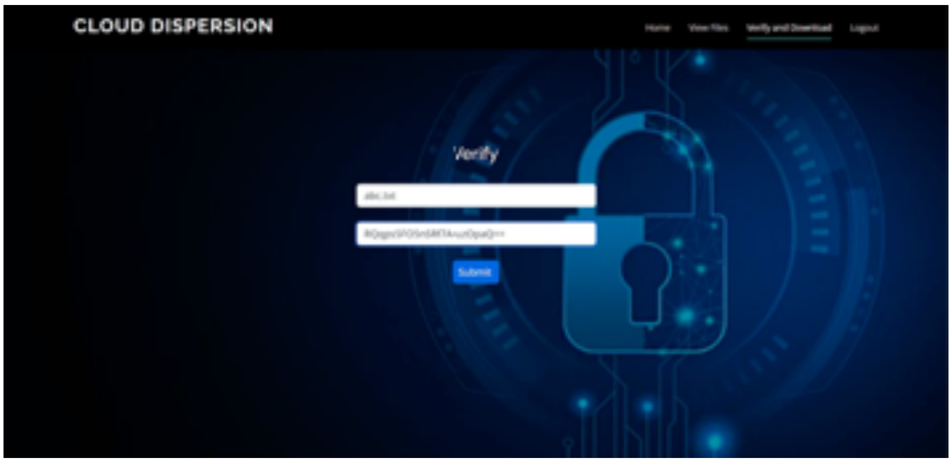


moulalicce225 26 Mar
to me

From moulalicce225@gmail.com
To 20eg105654@anurag.edu.in
Date 26 Mar 2024 at 11 48 AM
Standard encryption (TLS)
Learn more

File Keys: jm+klLq/BpqWKJICqDLA== and bckey1 :-1261255288 and bckey2 :465448389

Fig 7.3.9 Block keys and secret key sent to user



abc.txt

[illegible]

Fig 7.3.10 user verification level 1(block keys)

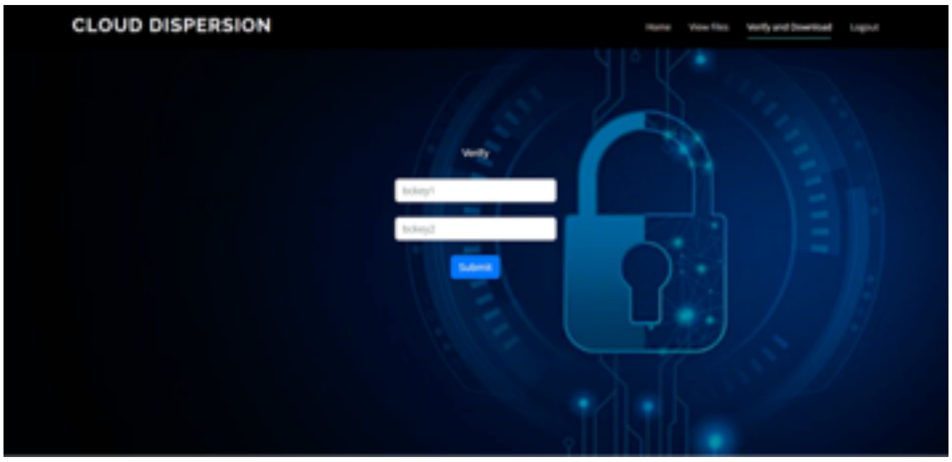
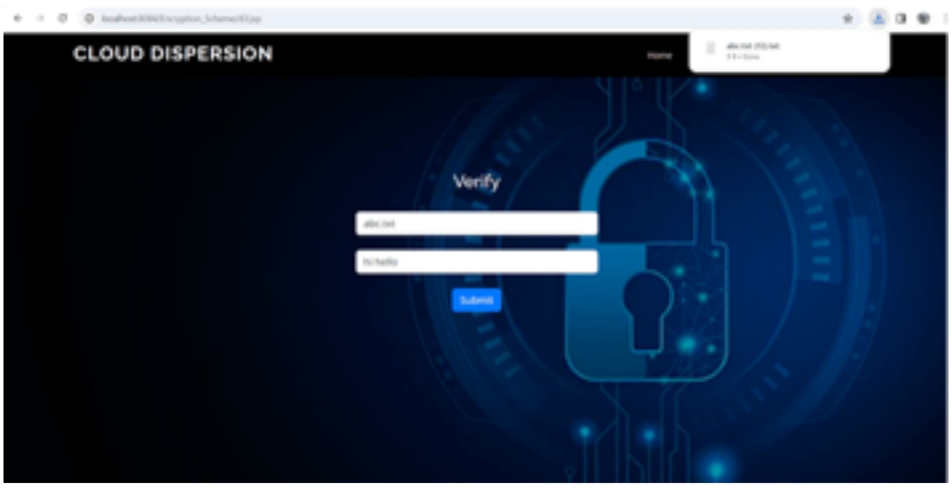


Fig 7.3.11 User verification level 2(Secret key)



abc.txt

hi hello

Fig 7.3.12 File downloaded

8 CONCLUSION

For the issue of cloud data leakage caused by management negligence and malicious attack at storage layer, we proposed CSSM, a cloud secure storage mechanism. CSSM adopted a combined approach of blockchain system and encryption technologies, which can improve the data security and pre-vent attackers from stealing user data. The experimental results show that CSSM can effectively prevent user data leakage at cloud storage layer. In terms of performance, the increased time overhead of CSSM is acceptable to users. This paper provides a feasible approach to solve the stor-age security problem, especially prevention from user data leakage at cloud storage layer. CSSM could also effectively protect cryptographic materials from storage perspective.

9 FUTURE SCOPE

While the current system using RSA encryption and blockchain for cloud storage security is promising, there's room for improvement. Double encryption with a symmetric key can be added for extra protection. Blockchain's potential can be maximized by storing encryption keys on it as well. Utilizing future-proof hashing algorithms like SHA-3 ensures long-term security. Looking ahead, integrating homomorphic encryption allows encrypted data manipulation and attribute-based encryption provides finer access control. Decentralized storage networks can further enhance security by distributing data storage. As quantum computers become a reality, post-quantum cryptography can be incorporated for ultimate protection. Remember, scalability, performance optimization, and adherence to emerging standards are crucial for real-world implementation. By continuously innovating and adapting, this system has the potential to revolutionize secure cloud storage.

10 REFERENCES

[1] H. Pagnia and F. C. Gartner, "On the impossibility of fair exchange without a trusted third party," Darmstadt Univ. Technol., Darmstadt, Germany, Tech. Rep. TUD-BS-1999-02.

[2] A. Küpçü and A. Lysyanskaya, "Usable optimistic fair exchange," in Proc. Cryptographers' Track RSA Conf. Cham, Switzerland: Springer, 2010, pp. 252–267.

[3] X. Chen, J. Li, and W. Susilo, "Efficient fair conditional payments for outsourcing computations," IEEE Trans. Inf. Forensics Security, vol. 7, no. 6, pp. 1687–1694, Dec. 2012.

[4] N. Asokan, M. Schunter, and M. Waidner, "Optimistic protocols for fair exchange," in Proc. 4th ACM Conf. Comput. Commun. Secur., 1997, pp. 7–17.

5] Q. Huang, G. Yang, D. S. Wong, and W. Susilo, "Ambiguous optimistic fair exchange," in Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur. Cham, Switzerland: Springer, 2008, pp. 74–89.

[6] M. T. Dashti, "Efficiency of optimistic fair exchange using trusted devices," ACM Trans. Auto. Adapt. Syst., vol. 7, no. 1, pp. 1–18, Apr. 2012.

[7] Q. Huang, G. Yang, D. S. Wong, and W. Susilo, "Ambiguous optimistic fair exchange: Definition and constructions," Theor. Comput. Sci., vol. 562, pp. 177–193, Jan. 2015.

[8] H. Huang, X. Chen, Q. Wu, X. Huang, and J. Shen, "Bitcoin-based fair payments for outsourcing computations of fog devices," Future Gener. Comput. Syst., vol. 78, pp. 850–858, Jan. 2018.

[9] L. Eckey, S. Faust, and B. Schlosser, "OptiSwap: Fast optimistic fair exchange," in Proc. 15th ACM Asia Conf. Comput. Commun. Secur., Oct. 2020, pp. 543–557.

[10] Y. Zhang, R. H. Deng, X. Liu, and D. Zheng, "Outsourcing service fair payment based on blockchain and its applications in cloud computing," IEEE Trans. Services Comput., vol. 14, no. 4, pp. 1152–1166, Jul. 2021.