A project report

on

DEDUPLICATION SCHEME FOR CLOUD DATA BASED ON CONVERGENT ENCRYPTION

Submitted in partial fulfillment of the requirements

for the award of the degree of

BACHELOR OF TECHNOLOGY

in

Computer Science & Engineering

by

Batch No: A-2

S.R.G. Gnana Deepika	**184G1A0517**
K. Lathasri	**184G1A0534**
T. Navya Deepthi	**184G1A0554**
B. Sarath Kumar	**194G5A0506**

Under the Guidance of

Dr.C.Sasikala M.Tech., Ph.D.

Associate Professor

Department of Computer Science & Engineering

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

(Affiliated to JNTUA & Approved by AICTE)

(Accredited by NAAC with 'A' Grade & Accredited by NBA(EEE, ECE & CSE))
Rotarypuram Village, B K Samudram Mandal, Ananthapuramu-515701.

2021-2022

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

(Affiliated to JNTUA & Approved by AICTE)
(Accredited by NAAC with 'A' Grade & Accredited by NBA (EEE, ECE & CSE))
Rotarypuram Village, B K Samudram Mandal, Ananthapuramu-515701.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Certificate

This is to certify that the project report entitled Deduplication Scheme For Cloud Data Based On Convergent Encryption is the bonafide work carried out by **S.R.G. Gnana Deepika** bearing Roll Number **184G1A0517, K. Lathasri** bearing Roll Number **184G1A0534, T. Navya Deepthi** bearing Roll Number **184G1A0554, B. Sarath Kumar** bearing Roll Number **194G5A0506** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering** during the academic year 2021-2022.

Signature of the Guide

Dr. C. Sasikala M.Tech., Ph.D.
Associate Professor

Head of the Department

Mr. P. Veera Prakash M.Tech. (Ph.D)
Assistant Professor & HOD

Date: **EXTERNAL EXAMINER**
Place: Rotarypuram

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned our efforts with success. It is a pleasant aspect that we have now the opportunity to express our gratitude for all of them.

It is with immense pleasure that we would like to express our indebted gratitude to our Guide **Dr. C. Sasikala** M.Tech., Ph.D**, Associate Professor**, **Computer Science & Engineering Department**, who has guided us a lot and encouraged us in every step of the project work. We thank her for the stimulating guidance, constant encouragement and constructive criticism which have made possible to bring out this project work.

We express our deep felt gratitude to **Mr. K. Venkatesh** M.Tech., **Assistant Professor, project coordinator** valuable guidance and unstinting encouragement enable us to accomplish our project successfully in time.

We are very much thankful to **Mr. P. Veera Prakash** M.Tech. (Ph.D)**, Assistant Professor & Head Of the Department, Computer Science & Engineering,** for his kind support and for providing necessary facilities to carry out the work.

We wish to convey our special thanks to **Dr. G. Bala Krishna** M.Tech, Ph.D**, Principal** of **Srinivasa Ramanujan Institute of Technology** for giving the required information in doing our project work. Not to forget, we thank all other faculty and non-teaching staff, and our friends who had directly or indirectly helped and supported us in completing our project in time.

We also express our sincere thanks to the Management for providing excellent facilities**.**

Finally, we wish to convey our gratitude to our family who fostered all the requirements and facilities that we need.

Project Associates

DECLARATION

We, Ms. S.R.G. Gnana Deepika bearing reg no: 184G1A0517, Ms. K. Lathasri bearing reg no: 184G1A0534, Ms. T. Navya Deepthi bearing reg no: 184G1A0554, Mr. B. Sarath Kumar bearing reg no: 194G5A0506, students of SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY, Rotarypuram, hereby declare that the dissertation entitled "DEDUPLICATION SCHEME FOR CLOUD DATA BASED ON CONVERGENT ENCRYPTION" embodies the report of our project work carried out by us during IV Year Bachelor of Technology under the guidance of Dr. C. Sasikala M.Tech., Ph.D, Department of CSE and this work has been submitted for the partial fulfillment of the requirements for the award of Bachelor of Technology degree.

The results embodied in this project report have not been submitted to any other Universities of Institute for the award of Degree.

S.R.G. GNANA DEEPIKA	Reg no: 184G1A0517
K. LATHASRI	Reg no: 184G1A0534
T. NAVYA DEEPTHI	Reg no: 184G1A0554
B. SARATH KUMAR	Reg no: 194G5A0506

CONTENTS

List of Figures

LIST OF ABBREVIATIONS

CSP	Cloud Storage Provider
IDC	International Data Corporation
ZB	Zetta Byte
CE	Convergent Encryption
MLE	Message Locked Encryption
OPRF	Oblivious Pseudo Random Function
J2EE	Java 2 Platform, Enterprise Edition
HTML	Hyper Text Markup Language
JVM	Java Virtual Machine
WWW	World Wide Web
JDBC	Java Data Base Connectivity
ODBC	Open Data Base Connectivity
API	Application Programming
JSP	Java Server Pages
CT	Cipher Text
UML	Unified Modeling Language
SRRS	Secure Role Re-encryption System
LAN	Local Area Network
AES	Advanced Encryption Standards
RSA	Rivest Shamir Adleman
SHA	Secured Hash Algorithm
IDE	Integrated Development Environment

ABSTRACT

Cloud computing is a powerful technology that provides a way of storing voluminous data that can easily be accessed anywhere and at any time by eliminating the need to maintain expensive computing hardware, dedicated space, and software. Addressing increasing storage needs is challenging and a time demanding task that requires large computational infrastructure to ensure successful data processing and analysis. With the continuous and exponential increase of the number of users and the size of their data, data deduplication becomes more and more a necessity for cloud storage. Rendering efficient storage and security for all data is very importantfor cloud computing. Securing and privacy preserving of data is of high priority when it comes tocloud storage. Therefore to provide efficient storage for cloud data owners and render high security for data we proposed a method called Data Deduplication.

Data Deduplication techniqueallows the cloud users to manage their cloud storage space effectively by avoiding storage of repeated data and save bandwidth. Convergent Encryption augments plain deduplication with an encryption layer that operates on the chunks before deduplication, and provides data confidentiality guarantees in deduplicated storage. This ensures that the encrypted chunks derived from duplicate chunks still have identical content, thereby being compatible with deduplication. Therefore Data deduplication eliminates excessive copies of data and significantly improves the storage space in the cloud.

Keywords: *Cloud Computing, Cloud storage, Deduplication, Convergent Encryption, Security, Data chunks.*

CHAPTER - 1
INTRODUCTION

1.1 Introduction

Cloud computing has been broadly-used in IT enterprise these days, which successfully allocates the computational sources and improves the efficiency of records storage. With the development of cloud computing, more and more customers pick to outsource their data to the Cloud Storage Provider (CSP). It doesn't rely where you are so long as you have a web and a device able to getting access to the net, you could get entry to your cloud storage providers.

Cloud storage allows keeping files in an off-website online area which you get entry to both through the general public network or a dedicated private network connection. Data that you transfer off-site for storage becomes the responsibility of a third-party cloud provider. The provider hosts, secures, manages, and maintains the servers and associated infrastructure and ensures you have access to the data whenever you need it.

According to IDC's reports, the amount of global data has reached to 4.4ZB in 2013 and will be expected to reach to 44ZB in 2020. With the rapid growth of data, a lot of redundant data are uploaded to the cloud storage servers, wasting a lot of communication bandwidth and storage space. Deduplication technology can detect and eliminate redundant data by keeping only one copy of the data, aiming to save the storage space. Research has shown that there are about 90% redundant data in cloud storage system, while the deduplication technology can save 83% storage space for backup system and 68% storage space for main memory system respectively.

Although deduplication can bring many benefits, it cannot be compatible with traditional encryption algorithms. In generally, the users can utilize different encryption scheme to ensure the security of data before outsourcing their data. As different users may choose different encryption keys, two identical data will be encrypted into different cipher texts, which make it impossible for CSP to conduct the deduplication check.

Chunk-based deduplication is widely used in modern primary and backup storage systems to achieve high storage savings. It stores only a single physical copy of duplicate chunks, while referencing all duplicate chunks to the physical copy by small- size references. Prior studies show that de duplication can effectively reduce the storage space of primary storage by 50% and that of backup storage by up to 98%. This motivates the wide deployment of deduplication in various commercial cloud storage services (e.g., Drop box, Google Drive, Bitcasa, Mozy, and Memopal) to reduce substantial storage costs.

To realize efficient deduplication check with encrypted data, proposed Convergent Encryption (CE), which adopted a deterministic encryption mechanism. Specifically, the users can utilize the convergent key to encrypt their data, where the convergent key can be obtained by computing the hash value of data itself. Finally, the users can generate the same cipher text for two identical data by CE. Therefore, CEalgorithm has been widely-used in cloud storage system supporting deduplication. Chunk-based deduplication is widely used in modern primary and backup, storage systems to achieve high storage savings. It stores only a single physical copy of duplicate chunks, while referencing all duplicate chunks to the physical copy by small-size references.

To provide confidentiality guarantees, encrypted deduplication, adds an encryption layer to deduplication, such that each chunk, before being written to deduplicated storage, is deterministically encrypted via symmetric-key encryption by a key derived from the chunk content (e.g., the key is set to be the cryptographic hash of chunk content). This ensures that duplicate chunks have identical content even after encryption, and hence we can still apply deduplication to the encrypted chunks for storage savings. Therefore Data deduplication eliminates excessive copies of data and significantly improves the storage space in the cloud.

In addition to storing non duplicate data, a deduplicated storage system needs to keep deduplication metadata. There are two types of de duplication metadata. To check if identical chunks exist, the system maintains a fingerprint index that tracks the fingerprints of all chunks that have already been stored. Also, to allow a file to be reconstructed, the system maintains a file recipe that holds the mappings from the

chunks in the file to the references of the corresponding physical copies.

Deduplication metadata is notoriously known to incur high storage overhead, especially for the highly redundant workloads (e.g., backups) as the metadata storage overhead becomes more dominant. In this work, we argue that encrypted deduplication incurs even higher metadata storage overhead, as it additionally keeps key metadata, such as the key recipes that track the chunk-to-key mappings to allow the decryption of individual files. Since the key recipes contain sensitive key information, they need to be managed separately from file recipes, encrypted by the master keys of file owners, and individually stored for different file owners. Such high metadata storage overhead can negate the storage effectiveness of encrypted deduplication in real deployment.

CHAPTER – 2

LITERATURE SURVEY

2.1 Introduction

Bhagyashree Bhoyane, Snehal Kalbhor, Sneha Chamle, Sandhya Itkapalle, P. M. Gore proposed Block-Level Message-Locked Encryption for Secure Large File Deduplication, in which MLE scheme can be extended to obtain secure de-duplication for large files, it requires a lot of metadata maintained by the end user and the cloud server. Especially, BL-MLE algorithm can achieve both file-level and block-level deduplication.

Jinbo Xiong, Yuanyuan Zhang, Shaohua Tang, Ximeng Liu, and Zhiqiang Yao proposed Secure Encrypted Data With Authorized Deduplication in Cloud, propose a novel secure role re-encryption system (SRRS), which is based on convergent encryption and the role re-encryption algorithm to prevent the privacy data leakage in cloud and it also achieves the authorized deduplication and satisfies the dynamic privilege updating and revoking.

M. Bellare. S. Keelveedhi, and T. Ristenpart, Proposed a Message-locked encryption and secure deduplication, Which will help to achieve a secured symmetric encryption scheme in which the key used for encryption and decryption is itself derived from the message.

Pasquale Puzio, Refik Molva, Melek Onen, Sergio Loureiro introduced Block-level De-duplication with Encrypted Data to the cloud storage system to store the metadata, which can achieve block-level authorized deduplication and confidentiality at the same time.

Through this literature survey helps to know about the various drawbacks present in the former deduplication methods such as transfer application metadata to block-layer deduplication, so as to accelerate the deduplication speed. Separate metadata from data to improve the storage efficiency of deduplication. While the above studies address metadata management. so as to compress deduplication

metadata, either cannot apply to the key recipe that is encrypted by the file owner's master key, or only reduce the metadata of zero chunks.

Deduplication to the keys directly to reduce the amount of key metadata. However, since the size of a key is often comparable to the size of the additional reference (both are of tens of bytes) to the corresponding physical copy, the storage saving of key metadata can be negated by such additional deduplication metadata in key-based deduplication, these systems degrade the storage efficiency achieved by deduplication. There is less efficiency on the cloud data due to High storage overhead of metadata.

2.2 Motivation

To over the above drawbacks the latter system was made where the Metadup builds on the idea of indirection. Instead of directly storing all deduplication and key metadata in both file and key recipes (both of which dominate the metadata storage overhead), we group the metadata in the form of metadata chunks that are stored in encrypted deduplication storage. Thus, both file and key recipes now store references to metadata chunks, which now contain references to data chunks (i.e., the chunks of file data). If Metadedup stores nearly identical files regularly (e.g., periodic backups), the corresponding file and key metadata are expected to have long sequences of references that are in the same order. This implies that the metadata chunks are highly redundant and hence can be effectively deduplicated.

Deduplication is a method for space-efficient data storage. It walls document facts into either fixed-size or variable-length chunks, and identifies each bite by using the cryptographic hash, called fingerprint, of the corresponding content.

Deduplication stores handiest one physical replica of replica chunks, and refers the duplicate chunks which have the equal fingerprint to the physical copy by small references. Encrypted Deduplication augments plain deduplication (i.e., deduplication without encryption) with an encryption layer that operates on the chunks before deduplication, and provides information confidentiality guarantees in deduplicated storage. It implements the encryption layer primarily based on message locked encryption (MLE), which encrypts each chunk with a symmetric key (referred to as

the MLE key) derived from the chunk content.

The system implements a Metadedup prototype and evaluates its performance in a networked setup. Compared to the network speed of our Gigabit LAN test bed, this shows that Metadedup incurs only 13.09% and 3.06% of throughput loss in writing and restoring files, respectively.

The system conducts trace-driven simulation on two real world datasets. This shows that Metadedup achieves up to 93.94% of metadata storage savings in encrypted deduplication. This also shows that Metadedup maintains the storage load balance among all servers.

Finally, by using this Convergent Encryption method the data deduplication can be achieved in an efficient way where the redundant data can be reduced and hence it improves data storage capacity.

CHAPTER – 3

Deduplication Technique Based On Convergent Encryption

This chapter discuss about the key generation and system model.

3.1 Key Generation

The system uses three types of Key Generations algorithms for generating the keys.

i. Advanced Encryption Standard (AES) algorithm.

ii. Rivest, Shamir, Adleman (RSA) algorithm.

iii. Secured Hash Algorithm (SHA1) algorithm.

A. Advanced Encryption Standard (AES) algorithm

The AES Encryption algorithm (also known as the Rijndael algorithm) is a symmetric block cipher algorithm with a block/chunk size of 128 bits. It converts these individual blocks using keys of 128, 192, and 256 bits. Once it encrypts these blocks, it joins them together to form the cipher text. The AES algorithm uses a substitution-permutation, or SP network, with multiple rounds to produce cipher text. The number of rounds depends on the key size being used. A 128-bit key size dictates ten rounds, a 192-bit key size dictates 12 rounds, and a 256-bit key size has 14 rounds. Each of these rounds requires a round key, but since only one key is inputted into the algorithm, this key needs to be expanded to get keys for each round, including round 0.

Steps in each round

Each round in the algorithm consists of four steps.

1. **Substitution of the bytes**
   In the first step, the bytes of the block text are substituted based on rules dictated by predefined S-boxes (short for substitution boxes).

2. **Shifting the rows**
   Next comes the permutation step. In this step, all rows except the first are shifted by one.

3. **Mixing the columns**

   In the third step, the Hill cipher is used to jumble up the message more by mixing the block's columns.

4. **Adding the round key**

   In the final step, the message is XORed with the respective round key.

B. Rivest, Shamir, Adleman (RSA) algorithm

RSA algorithm is asymmetric cryptography algorithm. Asymmetric actually means that it works on two different keys i.e. **Public Key** and **Private Key.** As the name describes that the Public Key is given to everyone and Private Key is kept private.

How it works

The RSA algorithm ensures that the keys, in the above illustration, are as secure as possible. The following steps highlight how it works:

1. **Generating the keys**

   Select two large prime numbers, x and y. The prime numbers need to be large so that they will be difficult for someone to figure out.

a. Calculate n = xn=x x y.

b. Calculate the ***totient*** function; ϕ(n) = (x-1) (y-1) ϕ (n) = (x−1) (y−1).

c. Select an integer e, such that e is ***co-prime*** to ϕ(n) ϕ (n) and 1 < e < ϕ(n) 1<e<ϕ (n). The pair of numbers (n,e)(n,e) makes up the public key.

d. Calculate d such that e.d = 1e.d=1 mod ϕ(n) ϕ (n).d can be found using the extended euclidean algorithm. The pair (n,d)(n,d) makes up the private key.

2. **Encryption**

   Given a plaintext PP, represented as a number, the cipher text C is calculated as: C = P^{e}C=Pe *mod* n.

3. Decryption

Using the private key (n,d)(n,d), the plaintext can be found using:

P = C^{d}P=Cd mod n.

C. Secure Hash Algorithm (SHA 1) algorithm

SHA-1 or Secure Hash Algorithm 1 is a cryptographic hash function which takes an input and produces a 160-bit (20-byte) hash value. This hash value is known as a message digest. This message digest is usually then rendered as a hexadecimal number which is 40 digits long. It is a U.S. Federal Information Processing Standard and was designed by the United States National Security Agency.

How SHA 1 works

SHA-1 works by feeding a message as a bit string of length less than 2^{64}264 bits, and producing a 160-bit hash value known as a message digest.

1) The first step is to initialize five random strings of hex characters that will serve as part of the hash function.

2) The message is then padded by appending a 1, followed by enough 0s until the message is 448 bits. The length of the message represented by 64 bits is then added to the end, producing a message that is 512 bits long.

3) The padded input obtained above, MM, is then divided into 512-bit chunks, and each chunk is further divided into sixteen 32-bit words, W_0 … W_{15}W0 …W15 . In the case of 'abc', there's only one chunk, as the message is less than 512-bits total.

4) For each chunk, begin the 80 iterations, i, necessary for hashing (80 is the determined number for SHA-1).

5) Now, store the hash values defined in step 1 in the following variables:

6) Store the result of the chunk's hash to the overall hash value of all chunks, as shown below, and proceed to execute the next chunk.

7) As a final step, when all the chunks have been processed, the message digest is represented as the 160-bit string comprised of the OR logical operator,∨, of the 5 hashed values.

3.2 System Model

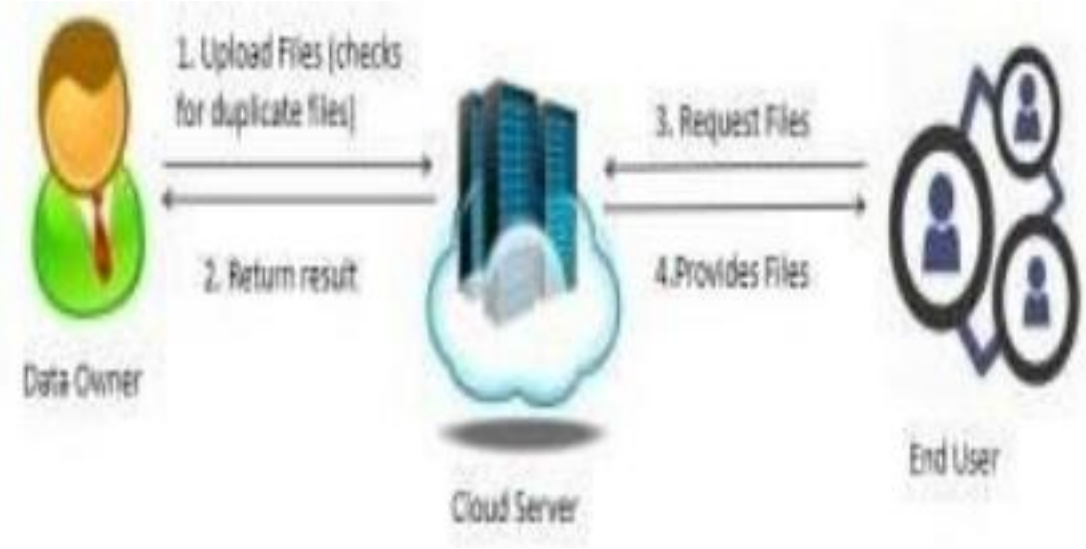


Fig 3.2.1: Working of System Model

The above figure 3.2.1 explains about different modules such as data user module, end user module, and cloud server module.

3.3 System Modules:

A. Data User Module

In this module, the data user uploads their data in the cloud server. For the security purpose the data user encrypts the data file blocks and then store in the cloud. The data user can check the duplication of the file blocks over corresponding cloud server. The Data user can have capable of manipulating the encrypted data file blocks and the data user can check the cloud data as well as the duplication of the specific file blocks and also he can create remote user with respect to registered cloud servers. The data user also checks data integrity proof on which the block is modified by the attacker.

The following are the fields of the Data User Module:

- Upload Files View Your Files
- Check Data Integrity
- Check Deduplication
- Update File

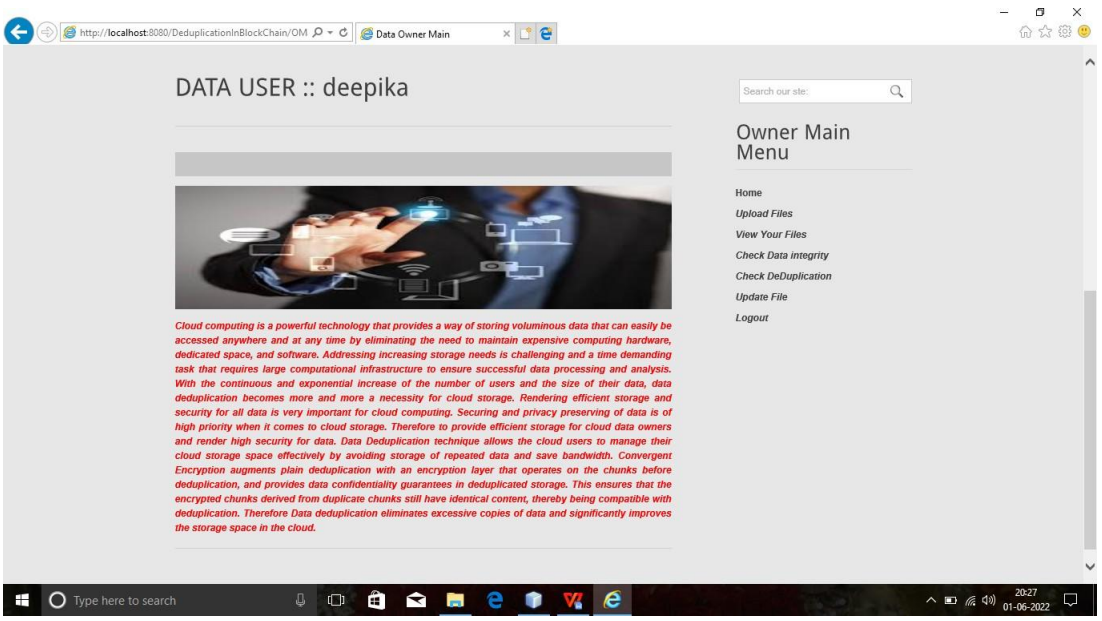


Fig 3.3.1: Data User Home Page

B. Cloud Server Module

The cloud service provider manages a cloud to provide data storage service. Data user encrypts their data files blocks and stores them in the cloud for sharing with Remote User. To access the shared data file's blocks, data consumers download encrypted data file's blocks of their interest from the cloud and then decrypt them.

The following are the fields of the Data User Module:

- View Data Users
- View End Users
- View File Requests
- View Attackers

- View Transactions
- View Blocks
- View Deduplication
- View Results
- View Throughput Results
- View Time Delay Results

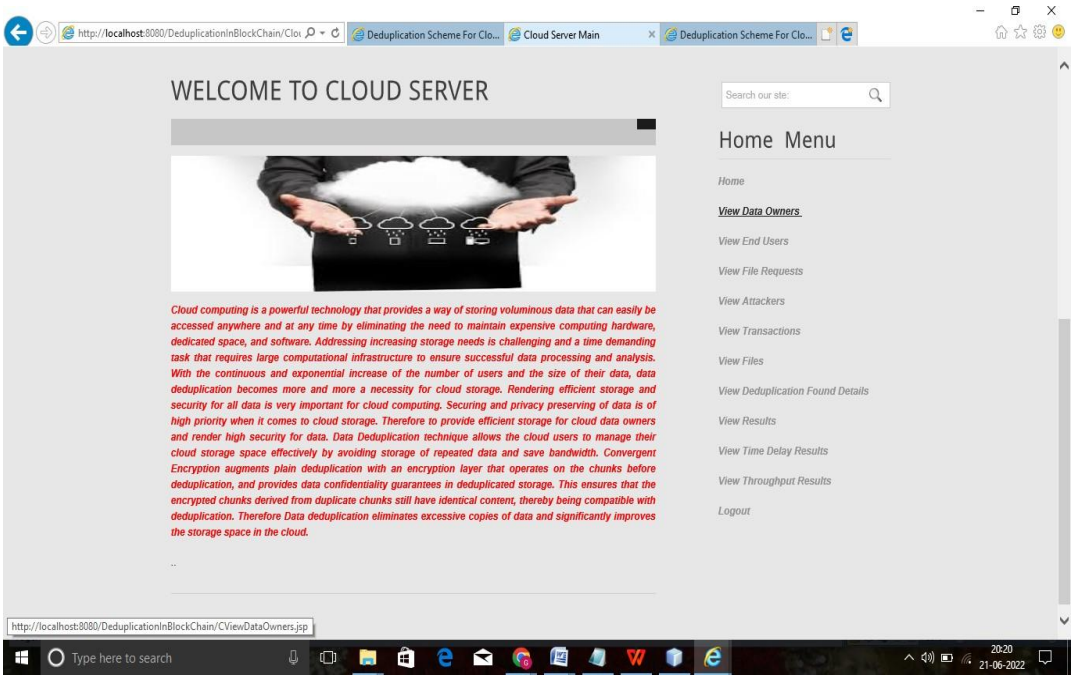


Fig 3.3.2: Cloud Server Home Page

C. End User Module

In this module, remote user logs in by using his user name and password. After he will request for secrete key of required file blocks from cloud servers, and get the secrete key. After getting secrete key he is trying to download file's blocks by entering file's blocks name and secrete key from cloud server.

The following are the fields of the Data User Module:

- Request File

- View File Response
- Download File

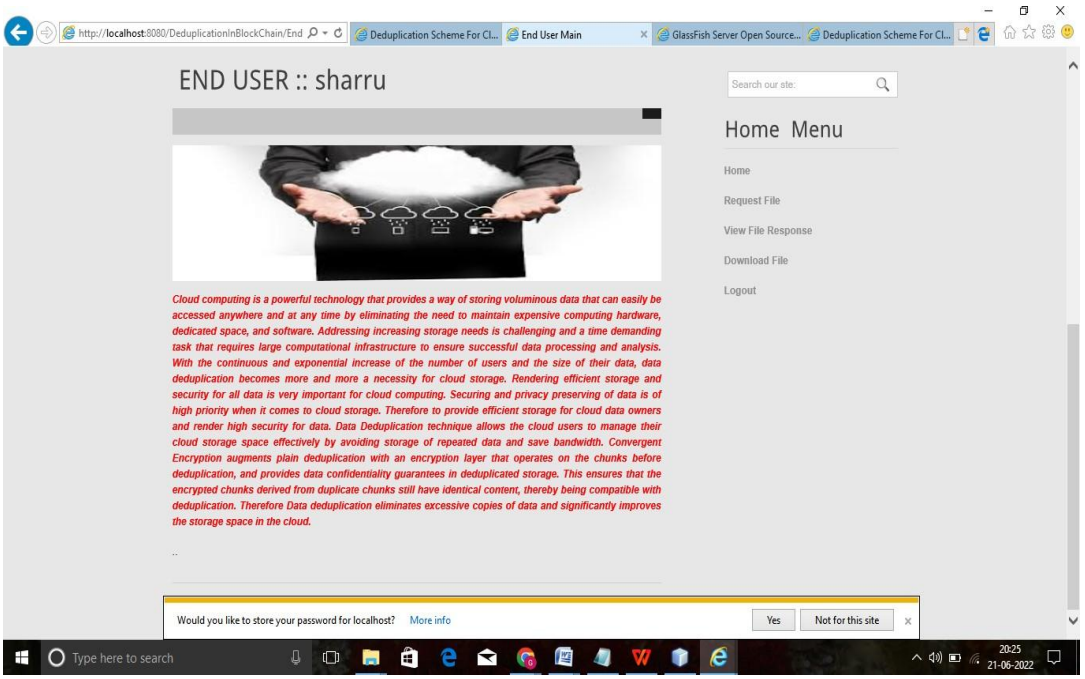


Fig 3.3.3: End User Home Page

D. Attacker Module

The user who attacks or modifies the block content called attacker. The attacker may the user who tries to access the file contents by wrong secret key from the cloud server.

CHAPTER – 4

UML DIAGRAMS

4.1 UML Introduction

The unified modeling language allows the software engineer to express an analysis model using the modeling notation that is governed by a set of syntactic, semantic and pragmatic rules. A UML system is represented using five different views that describe the system from distinctly different perspective.

UML is specifically constructed through two different domains, they are:

- UML Analysis modeling, this focuses on the user model and structural model views of the systems.
- UML Design modeling, this focuses on the behavioral modeling implementation modeling and environmental model views.

4.2 Usage of UML in Project

As the strategic value of software increases for many companies, the industry looks for techniques to automate the production of software and to improve quality and reduce cost and time to the market. These techniques include component technology, visual programming, patterns and frameworks. Additionally, the development for the World Wide Web, while making some things simpler, has exacerbated these architectural problems. The UML was designed to respond to these needs. Simply, systems design refers to the process of defining the architecture, components, modules, interfaces and data for a system to satisfy specified requirements which can be done easily through UML diagrams.

4.3 Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they

interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well.

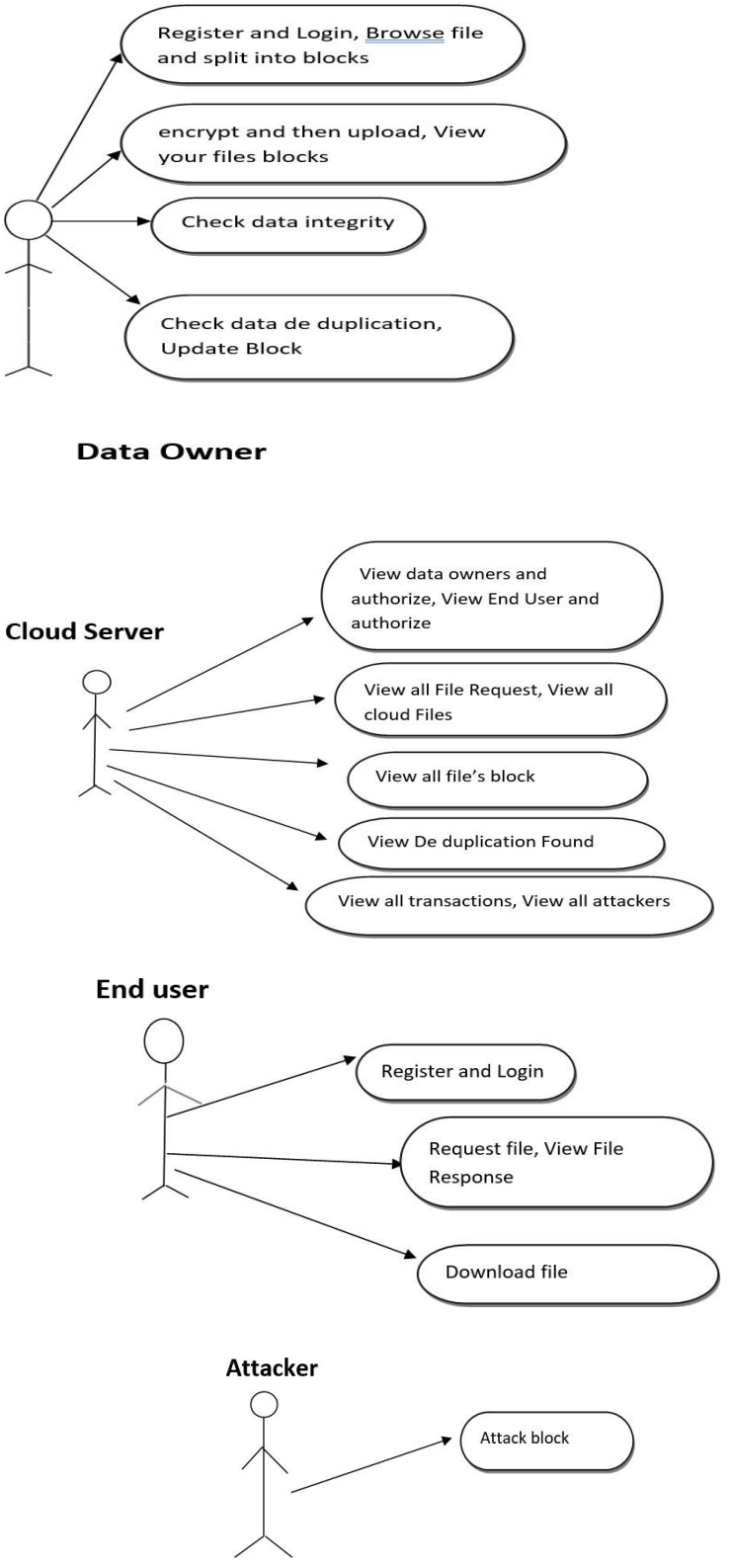


Fig 4.3.1: Use Case

4.4 Class Diagram

The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed. A class with three sections, in the diagram, classes is represented with boxes which contain three parts.

- The upper part holds the name of the class.
- The middle part contains the attributes of the class.
- The bottom part gives the methods or operations the class can take or undertake.

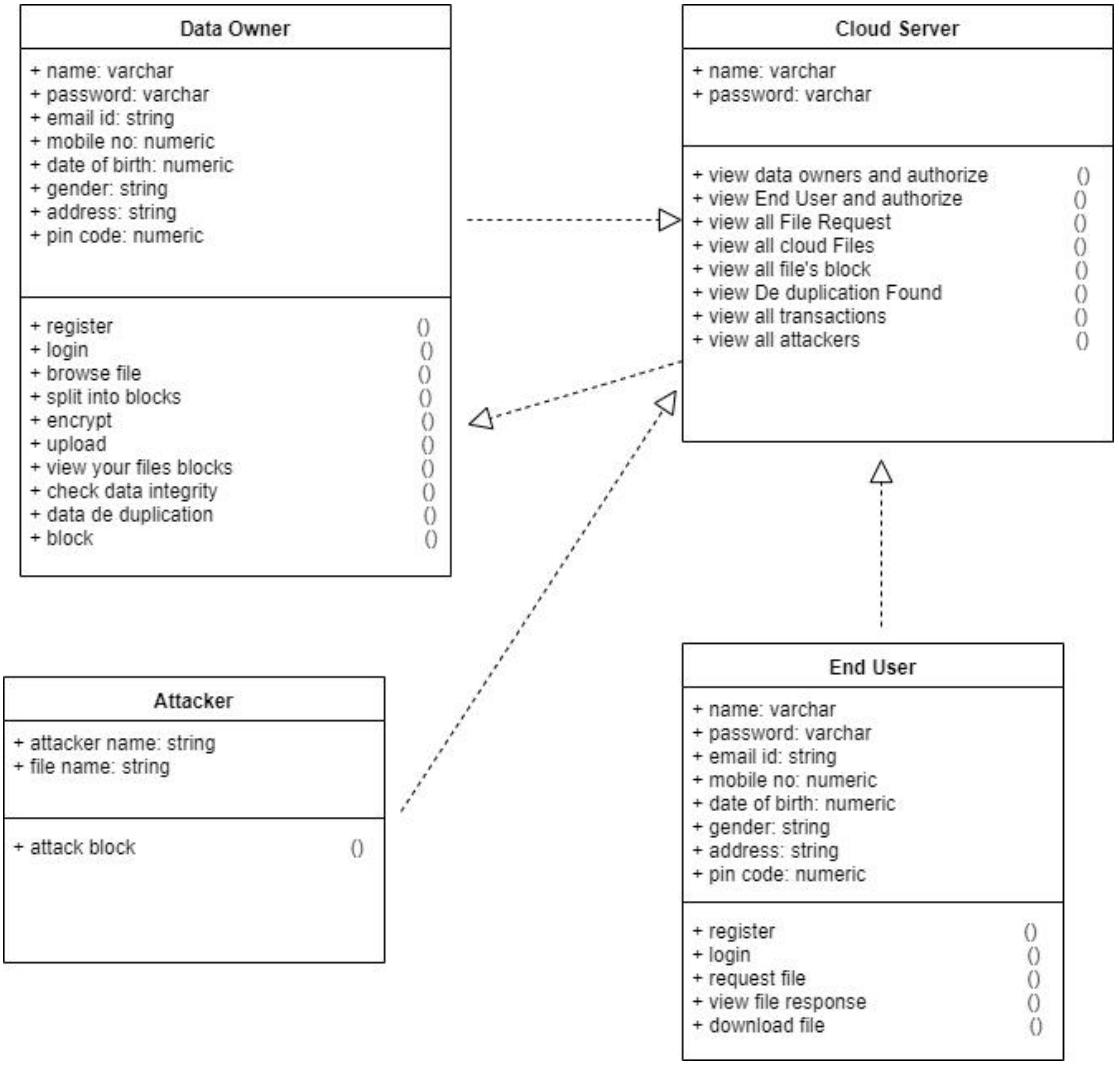


Fig 4.4.1: Class Diagram

CHAPTER – 5

IMPLEMENTATION

Encrypted deduplication combines encryption and deduplication in a seamless way to provide confidentiality guarantees for the physical data in deduplicated storage, yet it incurs substantial metadata storage overhead due to the additional storage of keys.

5.1 Data Base Connection

The MySql database is used to store the files and retrieve the files whenever needed by the users. To use the database it is mandatory to install mysql database and set a root path. Now this root path is used to connect to the Net beans Ide. The below source code shows how database is connected.

```
<%@ page import="java.sql.*"%>
<%@ page import="java.util.*" %>
<%
Connection connection = null;
        try {
        Class.forName("com.mysql.jdbc.Driver");
        connection =
DriverManager.getConnection("jdbc:mysql://localhost:3306/esse","root","root");
        String sql="";
                }
catch(Exception e)
{
        System.out.println(e);
}
%>
```

5.2 Proposed System

Metadedup builds on the idea of indirection. Instead of directly storing al deduplication and key metadata in both file and key recipes (both of which dominate themetadata storage overhead), we group the metadata in the form of metadata chunks that are stored in encrypted deduplication storage. Thus, both file and key recipes now store references to metadata chunks, which now contain references to data chunks (i.e., the chunks of file data). If Metadedup stores nearly identical files regularly (e.g., periodic backups [39]), the corresponding file and key metadata are expected to have longsequences of references that are in the same order. This implies that the metadata chunksare highly redundant and hence can be effectively deduplicated.

The system proposes a distributed key management approach that adapts Metadedup into a multi-server architecture for fault tolerant data storage. We generate the key of each data chunk from one of the servers, encode it via secret sharing, and distribute the resulting shares to remaining servers. This ensures fault-tolerant storage of data chunks, while being robust against adversarial compromise on a number of servers through our decoupled management of keys and shares.

The system implements a Metadedup prototype and evaluates its performance in a networked setup. Compared to the network speed of our Gigabit LAN tested, we show that Metadedup incurs only 13.09% and 3.06% of throughput loss in writing and restoring files, respectively.

Finally, the system conducts trace-driven simulation on two real world datasets. We show that Metadedup achieves up to 93.94% of metadata storage savings in encrypted deduplication. We also show that Metadedup maintains the storage load balance amongall servers.

Data user.jsp

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
    Transitional//EN""http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
```

```
<title>Deduplication Scheme For Cloud Data Based On Convergent Encryption </title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="css/style.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type="text/javascript" src="js/droid_sans_400-droid_sans_700.font.js">
</script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/coin-slider.min.js"></script>
<style type="text/css">
<!—
.style1 {
font-size: 24px;
color:
#FF0000; font-
weight: bold;
}
.style3 {
color:
#FF0000; font-
weight: bold;
}
.style4 {color: #0000FF}
.style5 {
color: #0000FF;
font-weight:
bold;font-style:
italic;
}
      .style6 {font-style: italic}
      .style7 {color: #FF0000}
      .style8 {font-style: italic; color: #FF0000;
      }
      -->
```

```
</style>
</head>
<body>
<div class="main">
<div class="header">
<div class="header_resize">
<div class="menu_nav">
<ul>
<li class="active"><a href="index.html"><span>Home Page</span></a></li>
<li><a href="DataUser.jsp"><span>Data User </span></a></li>
<li><a href="CloudServer.jsp"><span>Cloud Server </span></a></li>
<li><a href="EndUser.jsp"><span>End User </span></a></li>
 </ul>
 </div>
 <div class="clr"></div>
 <div class="logo">
 <h1><span class="style1"><a href="index.html" class=”style”></a>
 Deduplication Scheme For Cloud Data Based On Convergent Encryption
     </span></h1>
 </div>
 <div class="clr"></div>
 <div class="slider">
 <div id="coin-slider">
 <a href="#"><imgsrc="images/slide1.jpg" width="960"
 height="360"alt="" />
 </a><a href="#"><imgsrc="images/slide2.jpg" width="960"

 height="360" alt="" />
 </a><a href="#"><imgsrc="images/slide3.jpg"
 width=”960”height="360" alt="" /></a></div>
 <div class="clr"></div>
 </div>
 <div class="clr"></div>
 </div>
 </div>
```

```
<div class="content">
<div class="content_resize">
<div class="mainbar">
<div class="article">
<h2><span>WELCOME TO DATA USER LOGIN </span></h2>
<p class="infopost">  <a href="#" class="com"></a></p>
<div class="clr"></div>
<div class="img"></div>
<div class="post_content">
<form action="Authentication.jsp" method="post" id="leavereply">
<ol>
<li>
<span class="style8">

<label for="name"><strong>Name (required)</strong></label>
</span>
<span class="style6"><span class="style7"><strong>
<input id="name" name="userid" class="text" />
</strong></span></span></li>
<li>
<span class="style7"><em><strong>
<label for="email">Password (required)</label>
</strong></em></span><strong>
<input type="password" id="pass" name="pass" class="text" />
<label for="email"></label>
</strong>

<label for="email"></label>
</li>
<li><a href="Register.html"><strong>REGISTER</strong></a>
<input name="imageField" type="submit"
class="style3"id="imageField" value="Login" />
<input name="Reset" type="reset" class="style3" value="Reset" />
</li>
<li></li>
```

```
<li><br />
</li>
</ol>
</form>
<p class="spec"><a href="#" class="rm">..</a></p>
</div>
<div class="clr"></div>
</div>
</div>
<div class="sidebar">
<div class="searchform">
<form id="formsearch" name="formsearch"
method=”post”action="#">
<span>
<input
name="editbox_search"class="editbox_search"
id="editbox_search" maxlength="80" value="Search
ourste:" type="text" />
</span>
<input     name="button_search"     src="images/search.gif"
          class="button_search"type="image" />
</form>
</div>
<div class="clr"></div>
<div class="gadget">
<h2 class="star"><span>Home </span> Menu</h2>
<div class="clr"></div>

<ul class="sb_menu">
<li><a href="index.html">Home</a></li>
<li><a href="DataUser.jsp">Data User </a></li>
<li><a href="CloudServer.jsp">Cloud Server </a></li>
<li><a href="EndUser.jsp">End User </a></li>
</ul>
</div>
```

```
</div>
<div class="clr"></div>
</div>
</div>
<div class="fbg"></div>
<div class="footer">

</html>
```

Cloud Server Main

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN""http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Cloud Server Main</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="css/style.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type=”text/javascript” src=”js/droid_sans_400-droid_sans_700.font.js”>
</script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/coin-slider.min.js"></script>
<style type="text/css">
<!--
.style1 {
font-size: 24px;
color:
#FF0000; font-
weight: bold;
}
.style2 {
color: #FF0000;

font-style:
```

```
italic; font-
weight: bold;
}
.style3 {
color:
#FF0000; font-
weight: bold;
}
.style4 {color: #0000FF}
.style5 {
color:
#0000FF; font-
weight: bold;
font-style:
italic;
}
.style6 {
font-style:
italic;color:
#0000FF;
}
.style7 {color: #000000}
-->
</style>
</head>
<body>
<div class="main">
<div class="header">
<div class="header_resize">
<div class="menu_nav">
<ul>
<li class="active"><a href="index.html"><span>Home Page</span></a></li>
<li><a href="DataUser.jsp"><span>Data User </span></a></li>
<li><a href="CloudServer.jsp"><span>Cloud Server </span></a></li>
<li><a href="EndUser.jsp"><span>End User </span></a></li>
```

```
</ul>
</div>
<div class="clr"></div>
<div class="logo">
<h1><span class="style1"><a href="index.html" class="style1"></a> Deduplication
Scheme For Cloud Data Based On Convergent Encryption </span></h1>
</div>
<div class="clr"></div>
<div class="slider">
<div id="coin-slider"><a href="#"><imgsrc="images/slide1.jpg"
width="960"height="360" alt="" /></a>
<a href="#"><imgsrc="images/slide2.jpg" width="960" height="360" alt="" /></a>
<a href="#"><imgsrc="images/slide3.jpg" width="960" height="360"
alt=""
/></a></div>
<div class="clr"></div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="content">
<div class="content_resize">
<div class="mainbar">
<div class="article">
<h2><span>WELCOME TO CLOUD SERVER </span></h2>
<p class="infopost">  <a href="#" class="com"></a></p>
<div class="clr"></div>
<div class="img"><imgsrc="images/img1.jpg" width="620" height="154" alt=""
class="fl"
/></div>
<div class="post_content">
```

<p align="justify" class="style2">Cloud computing is a powerful technology that provides a way of storing voluminous data that can easily be accessed anywhere and at any time by eliminating the need to maintain expensive computing hardware, dedicated space, and software. Addressing increasing storage needs is challenging and a time demanding task that requires large computational infrastructure to ensure

successful data processing and analysis. With the continuous and exponential increase of the number of users and the size of their data, data deduplication becomes more and more a necessity for cloud storage. Rendering efficient storage and security for all datais very important for cloud computing. Securing and privacy preserving of data is of high priority when it comes to cloud storage. Therefore to provide efficient storage for cloud data owners and render high security for data we proposed a method called Data Deduplication. Data Deduplication technique allows the cloud users to manage their cloud storage space effectively by avoiding storage of repeated data and save bandwidth. Convergent Encryption augments plain deduplication with an encryption layer that operates on the chunks before deduplication, and provides data confidentiality guarantees in deduplicated storage. This ensures that the encrypted chunks derived from duplicate chunks still have identical content, thereby being compatible with deduplication. Therefore Data deduplication eliminates excessive copies of data and significantly improves the storage space in the cloud. </p>

```
<p class="spec"><a href="#" class="rm">..</a></p>
</div>
<div class="clr"></div>
</div>
</div>
<div class="sidebar">
<div class="searchform">
<form id="formsearch" name="formsearch" method="post" action="#">
<span>
<input name="editbox_search" class="editbox_search"
id="editbox_search"maxlength="80" value="Search our ste:" type="text"
/>
</span>
<input name="button_search" src="images/search.gif"
class="button_search"type="image" />
</form>
</div>
<div class="clr"></div>
<div class="gadget">
<h2 class="star"><span>Home </span> Menu</h2>
<div class="clr"></div>
<ul class="sb_menu style6 style3">
<li><strong><a href="CloudServerMain.jsp">Home</a></strong></li>
<li><strong><a href="CViewDataOwners.jsp">View Data Owners</a>
</strong></li>
```

```
<li><strong><a href="CViewEndUsers.jsp">View End Users</a></strong>
</li>
<li><strong><a href="CViewFileRequest.jsp">View File Requests</a>
</strong></li>
<li><strong><a href="CViewAllAttackers.jsp">View Attackers</a>
</strong></li>

<li><strong><a href="CViewAllTransactions.jsp">View Transactions</a>
</strong></li>
<li><strong><a href="CViewAllBlocks.jsp">View Blocks</a></strong></li>
<li><a href="CViewAllDeduplication.jsp">View Deduplication Found Details
</a></li>
<li><a href="CViewResults.jsp">View Results</a></li>
<li><a href="ViewTDResults1.jsp">View Time Delay Results</a></li>
<li><a href="ViewTPTResults2.jsp">View Throughput Results</a></li>
<li><strong><a href="index.html">Logout </a></strong></li>
</ul>
</div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="fbg"></div>
<div class="footer">
</html>
```

End User Module

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
    Transitional//EN""http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>End User Main</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="css/style.css" rel="stylesheet" type="text/css" />
```

```
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type="text/javascript" src="js/droid_sans_400-droid_sans_700.font.js">
</script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>

<script type="text/javascript" src="js/coin-slider.min.js"></script>
<style type="text/css">
<!—
.style1 {
    font-size: 24px;
    color:
    #FF0000; font-
    weight: bold;
}
.style2 {
    color:
    #FF0000; font-
    style: italic;
    font-weight:
    bold;
}
.style3 {
    color:
    #FF0000; font-
    weight: bold;
}
.style4 {color: #0000FF}
.style5 {
    color:
    #0000FF; font-
    weight: bold;
    font-style:
    italic;
```

```
}
-->
</style>
</head>
<body>
<div class="main">
<div class="header">
<div class="header_resize">
<div class="menu_nav">
<ul>
<li class="active"><a href="index.html"><span>Home Page</span></a></li>
<li><a href="DataUser.jsp"><span>Data User </span></a></li>
<li><a href="CloudServer.jsp"><span>Cloud Server </span></a></li>
<li><a href="EndUser.jsp"><span>End User </span></a></li>
</ul>
</div>
<div class="clr"></div>
<div class="logo">
<h1><span class="style1"><a href="index.html" class="style1"></a>Deduplication
    Scheme For Cloud Data Based On Convergent Encryption
</span></h1>
</div>
<div class="clr"></div>
<div class="slider">
<div id="coin-slider"><a href="#"><imgsrc="images/slide1.jpg"
width="960"height="360" alt="" /></a><a
href="#"><imgsrc="images/slide2.jpg" width="960" height="360" alt=""
/></a><a href="#"><img src="images/slide3.jpg" width="960"
height="360" alt="" /></a></div>
<div class="clr"></div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="content">
```

```
<div class="content_resize">
<div class="mainbar">
<div class="article">
<h2><span> END USER :: <%=application.getAttribute("uname") %>
</span></h2>
<p class="infopost">  <a href="#" class="com"></a></p>
<div class="clr"></div>
<div class="img"><imgsrc="images/img1.jpg" width="620"
height="154"alt="" class="fl" /></div>
<div class="post_content">
```

Cloud computing is a powerful technology that provides a way of storing voluminous data that can easily be accessed anywhere and at any time by eliminating the need to maintain expensive computing hardware,dedicated space, and software. Addressing increasing storage needs is challenging and a time demanding task that requires large computational infrastructure to ensure successful data processing and analysis. With the continuous and exponential increase of the number of users and the size of their data, data deduplication becomes more and more a necessity for cloud storage. Rendering efficient storage and security for all data is very important for cloud computing. Securing and privacy preserving of data is of high priority when it comes to cloud storage. Therefore to provide efficient storage for cloud data owners and render high security for data we proposed a method called Data Deduplication. Data Deduplication technique allows the cloud users to manage their cloud storage space effectively by avoiding storage of repeated data and save bandwidth. Convergent Encryption augments plain deduplication with an encryption layer that operates on the chunks before deduplication, and provides data confidentiality guarantees in deduplicated storage. This ensures that the encrypted chunks derived from duplicate chunks still have identical content, thereby being compatible with deduplication. Therefore Data deduplication eliminates excessive copies of data and significantly improves the storage space in the cloud.

```
<p class="spec"><a href="#" class="rm">..</a></p>
</div>
<div class="clr"></div>
</div>
```

```
</div>
<div class="sidebar">
<div class="searchform">
<form id="formsearch" name="formsearch" method="post" action="#">
<span>
<input name="editbox_search" class="editbox_search"
id="editbox_search"maxlength="80" value="Search our ste:" type="text"
/>
</span>
<input name="button_search" src="images/search.gif"
class="button_search"type="image" />
</form>
</div>
<div class="clr"></div>
<div class="gadget">
<h2 class="star"><span>Home </span> Menu</h2>
<div class="clr"></div>
<ul class="sb_menu style3">
<li><a href="EndUserMain.jsp">Home</a></li>
<li><a href="ERequestFile.jsp">Request File </a></li>
<li><a href="EViewFileResponse.jsp">View File Response</a></li>
<li><a href="EndDownloadFile.jsp">Download File</a></li>
<li><a href="index.html">Logout</a></li>
</ul>
</div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="fbg"></div>
<div class="footer">
</html>
```

CHAPTER - 6

RESULTS

The proposed system is implemented in Net Beans IDE using Java files. The files uploaded by the data user will split into metadata chunks and this encrypted data files will be checked for deduplication process by the cloud server. If duplicate data files are found a message, will be shown to the data user to upload another files otherwise cloud server stores the copy of the file. Likewise an end user who has authentication, can request a file to the cloud server. Upon acceptance of the cloud server an end user can get a key and decrypt to view and download a file from the cloud server.

The step by step execution of the proposed system is as shown:

The Home Screen is as shown in fig 6.1:

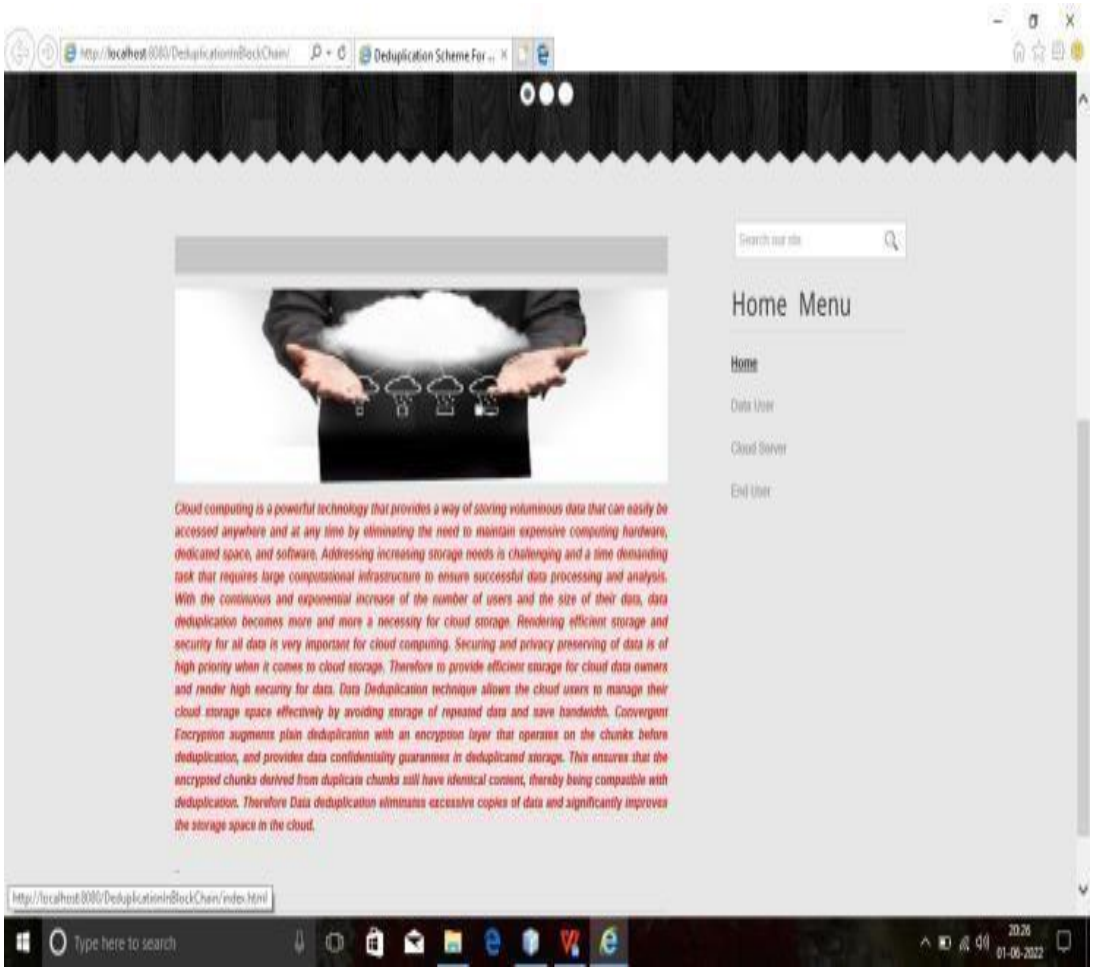


Fig 6.1: Home Screen

6.1 Data User

A. The Data user needs to register and login in order to upload the files. It is shown in fig 6.1.1

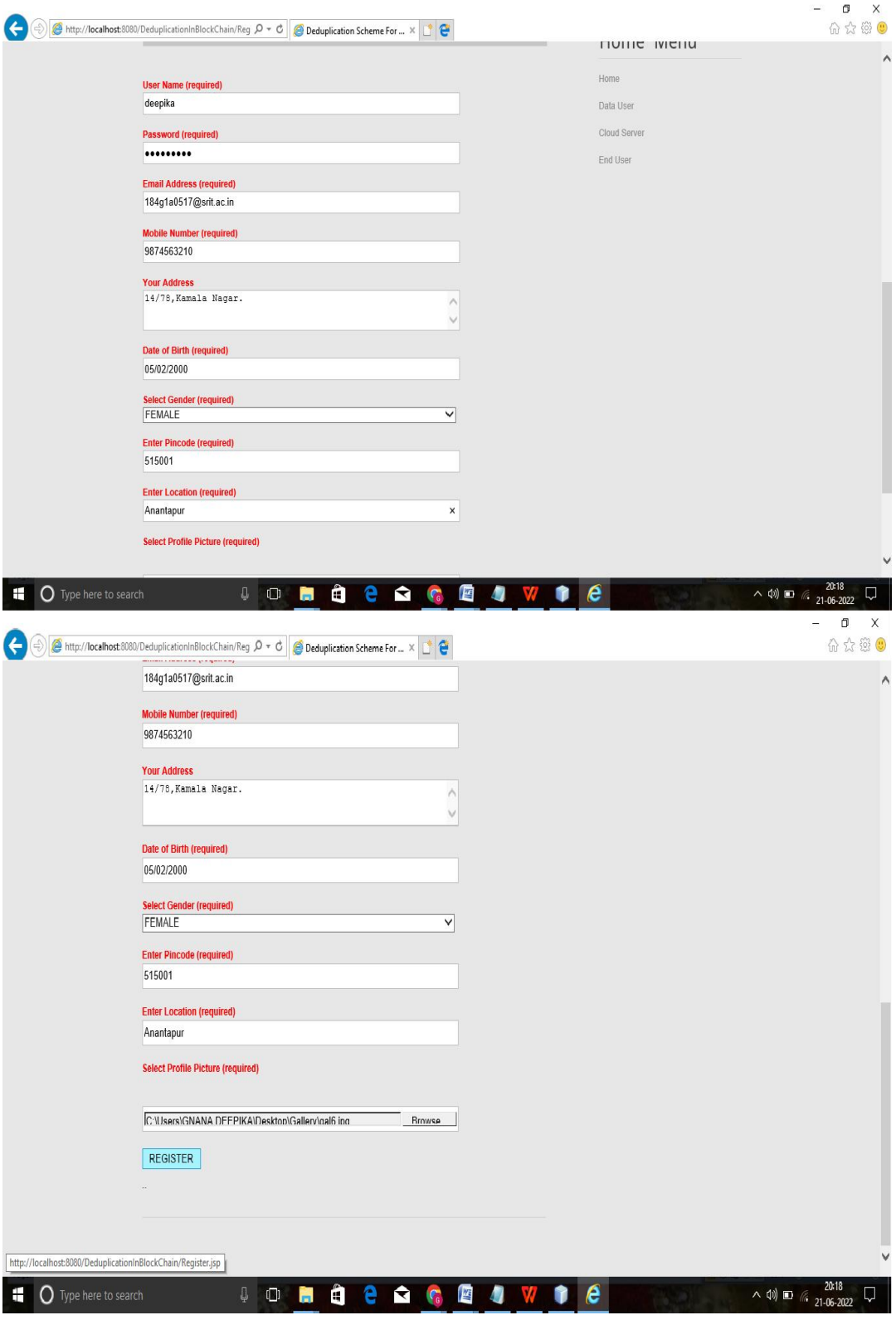


Fig 6.1.1: Data User Registration

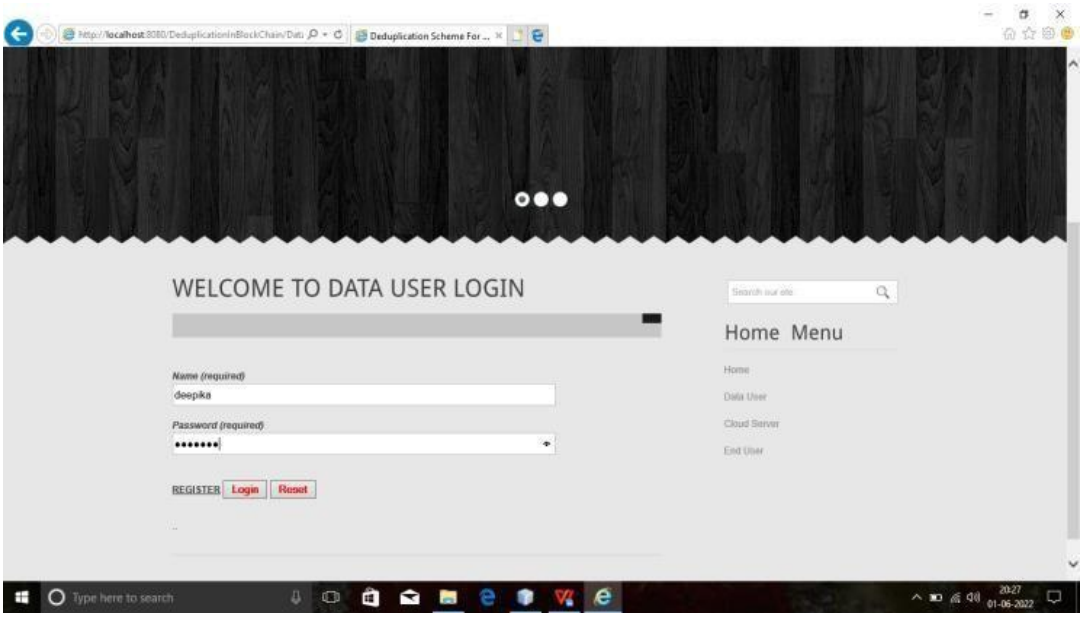


Fig 6.1.2: Data User Login

B. After successful login the data user can access all the fields that are shown in fig.

Fig 6.1.3: Data User Fields

C. For uploading the Data Files we need to select and choose the file from your device and give a file name and select encrypt button as shown in fig.

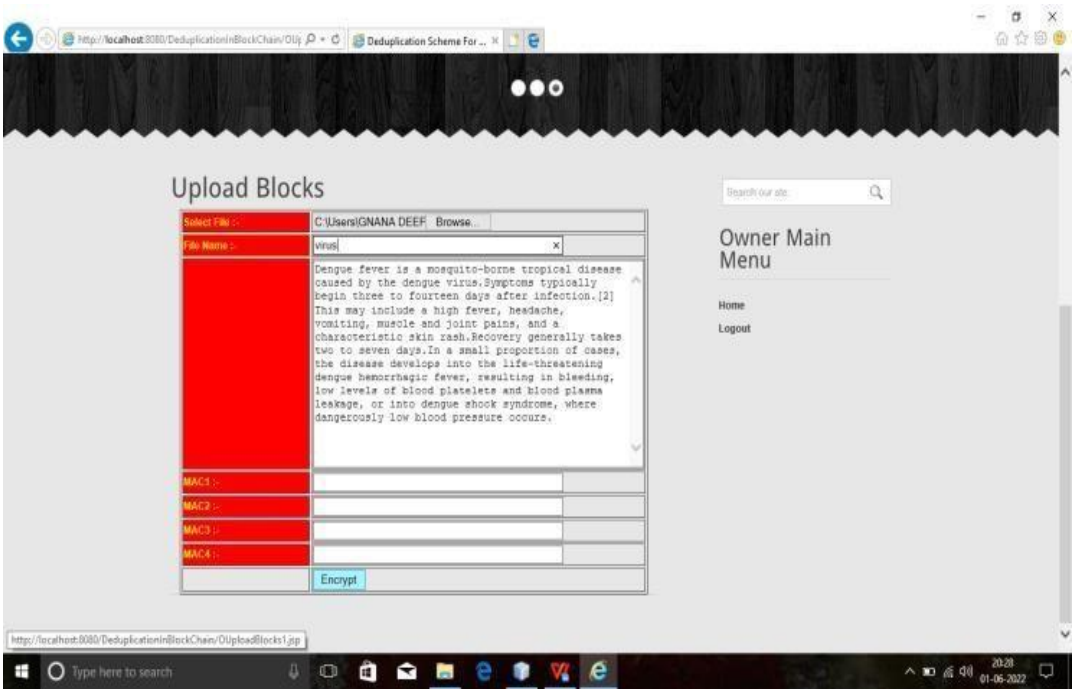
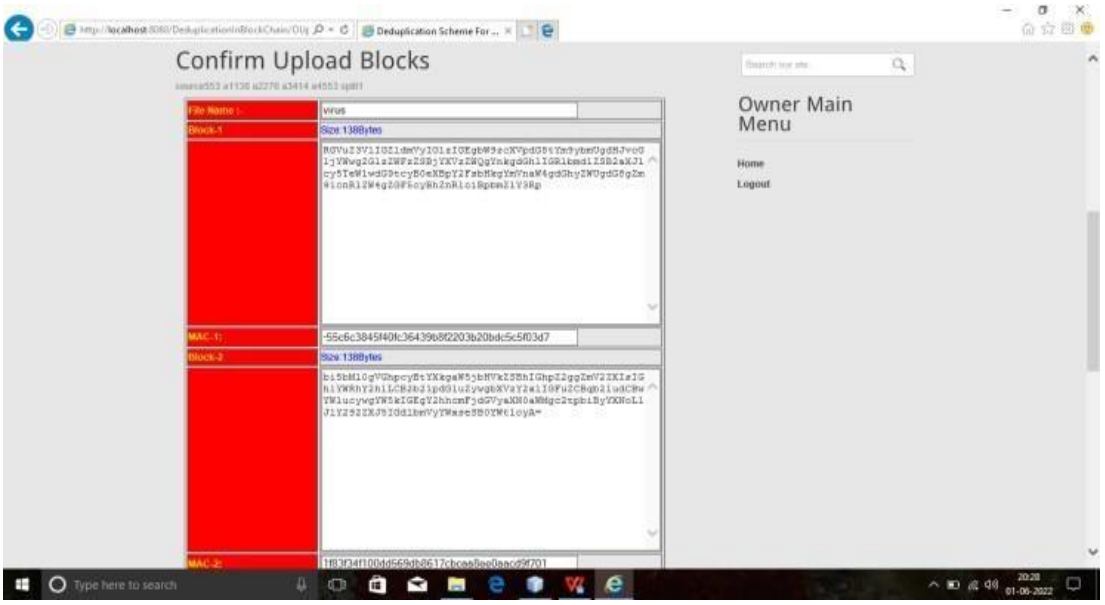


Fig 6.1.4: Uploading File

D. After encryption confirm data block chunks in order to upload and check for duplicate files as shown in fig 5.5



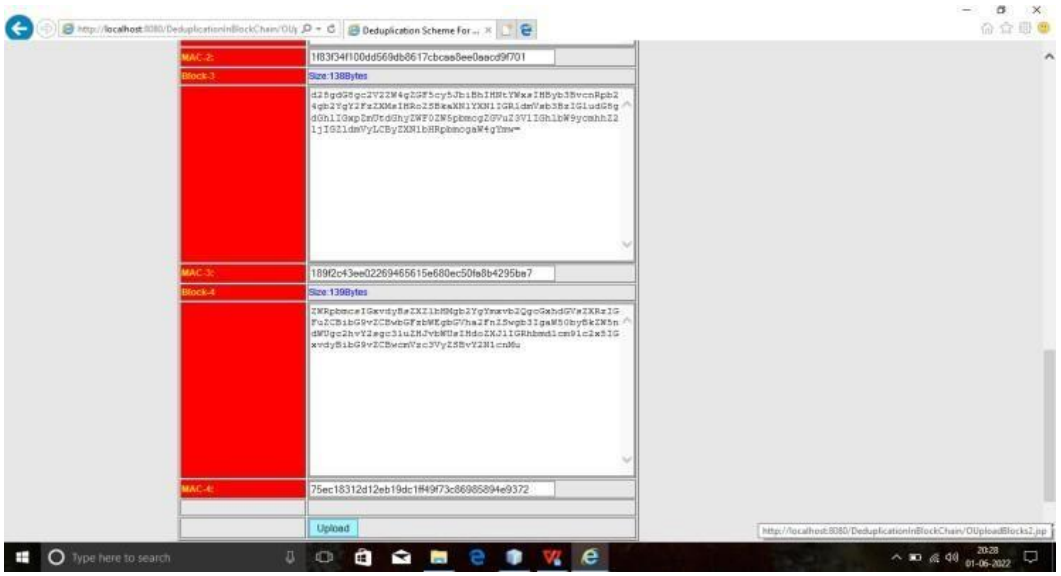


Fig 6.1.5: Confirm File Blocks

E. Upon pressing upload button the cloud server will check for duplicate files, if any duplicate files are found and a message will be displayed to the data user to upload another file as shown in fig.

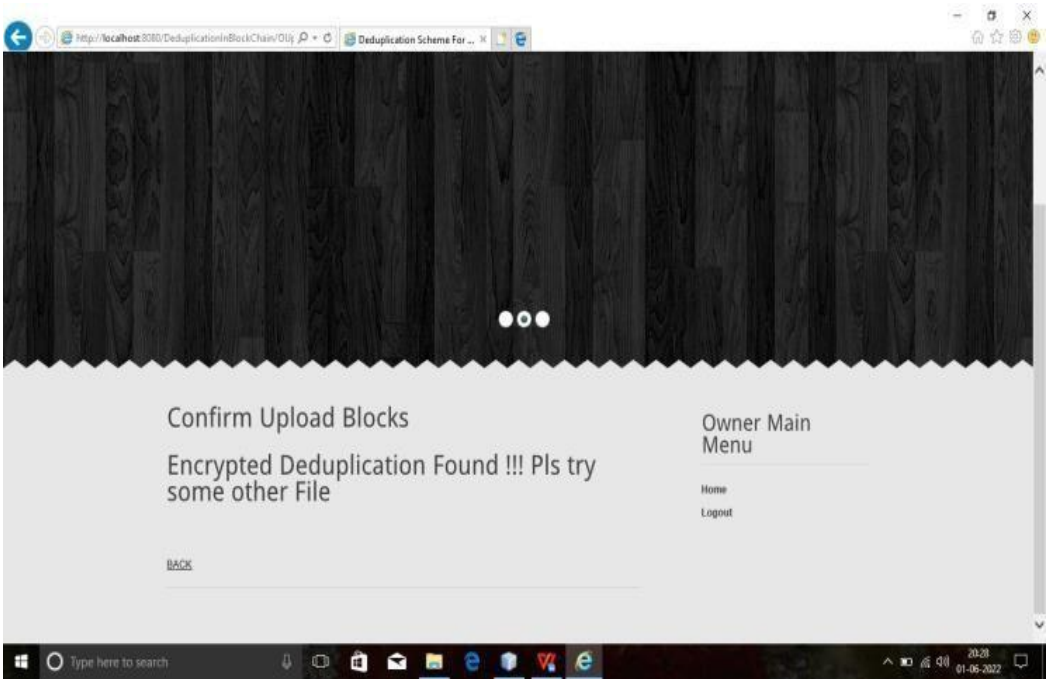


Fig 6.1.6: Duplicate File Found

F. Otherwise the files will be uploaded to the cloud server and a success message will be displayed to the data user as shown in fig 5.8

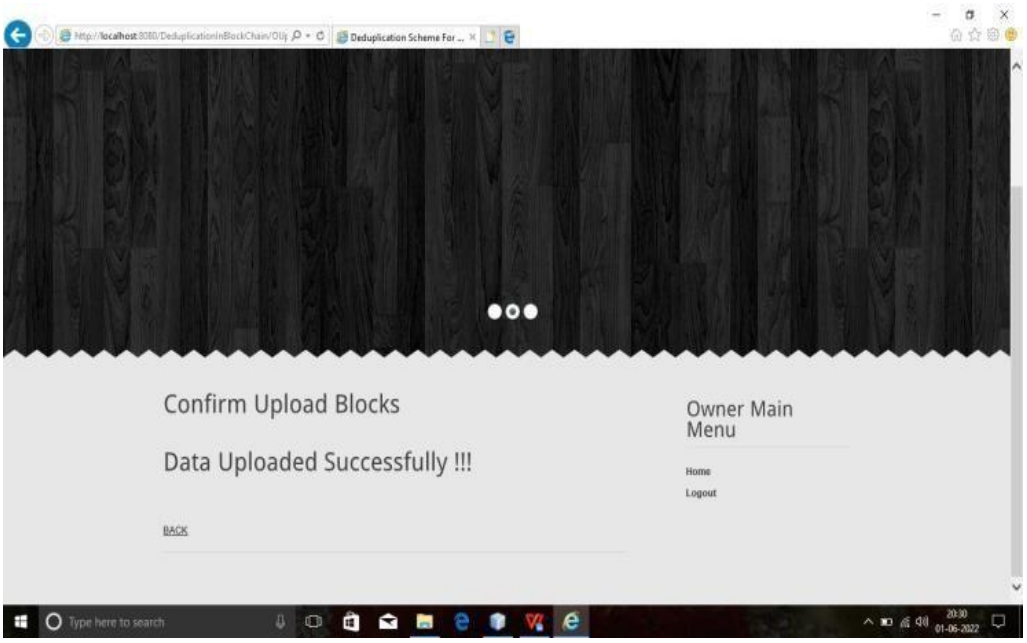


Fig 6.1.7: File Uploaded Successfully

6.2 Cloud Server

G. The Cloud Server need to login into his page as shown:

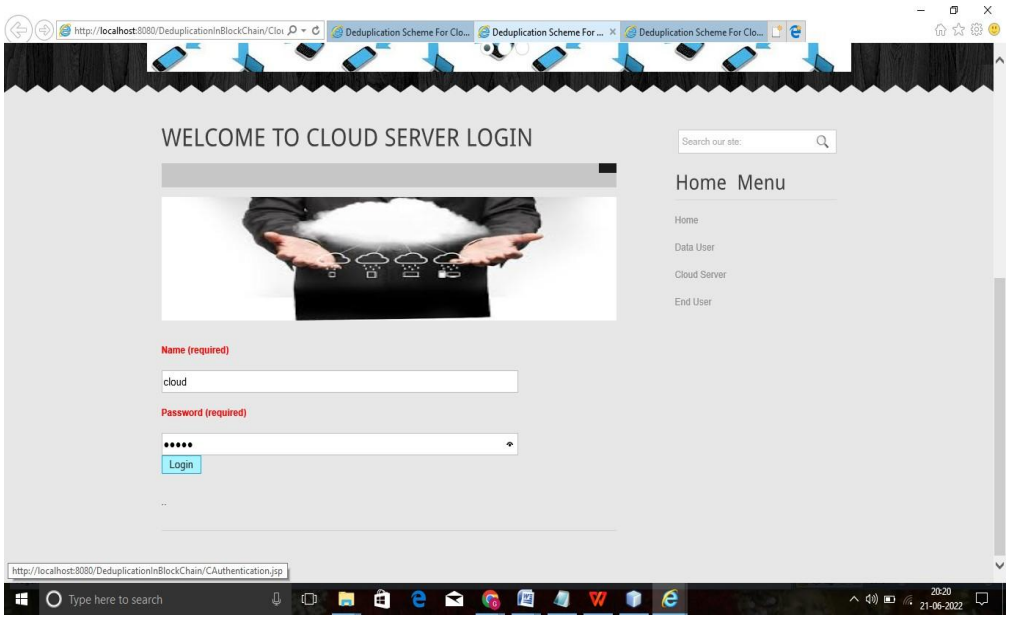


Fig 6.2.1: Cloud Server Login

H. The following are the fields in cloud server main page.

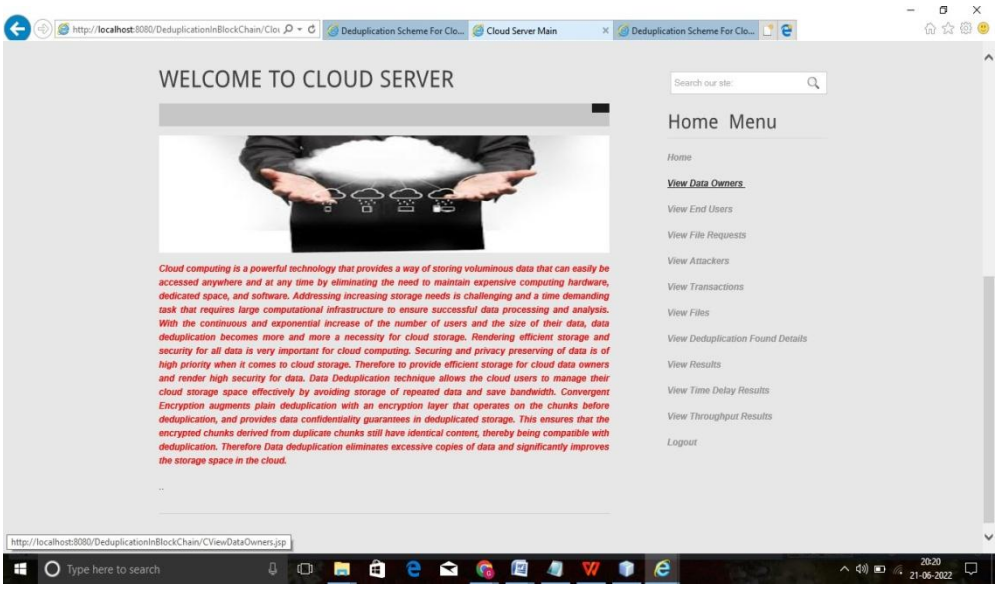


Fig 6.2.2: Cloud Server Main Page

I. The Cloud server can view all the Data Users

Fig 6.2.3: Data User Page

J. The Cloud server can view all the End Users

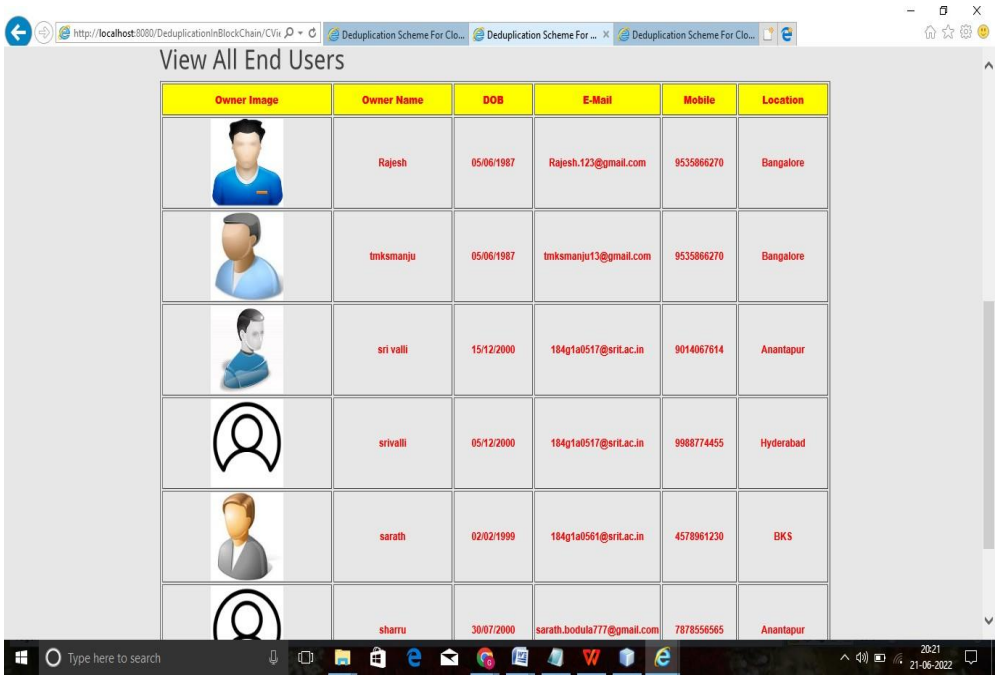


Fig 6.2.4 : End Users Page

K. The Cloud server can view all the Files.

Fig 6.2.5 : Files Page

L. The Cloud server can view all the Attackers

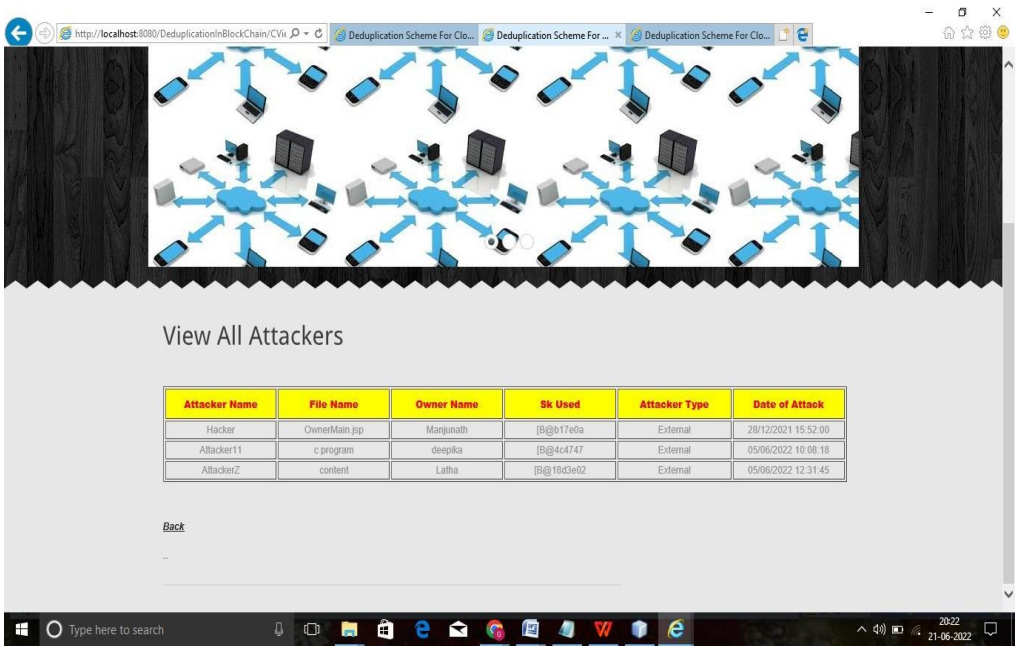


Fig 6.2.6 : Attackers Page

M. The Cloud server can view all the Transactions

Fig 6.2.7 : Transactions Page

N. The Cloud server can view all the Blocks

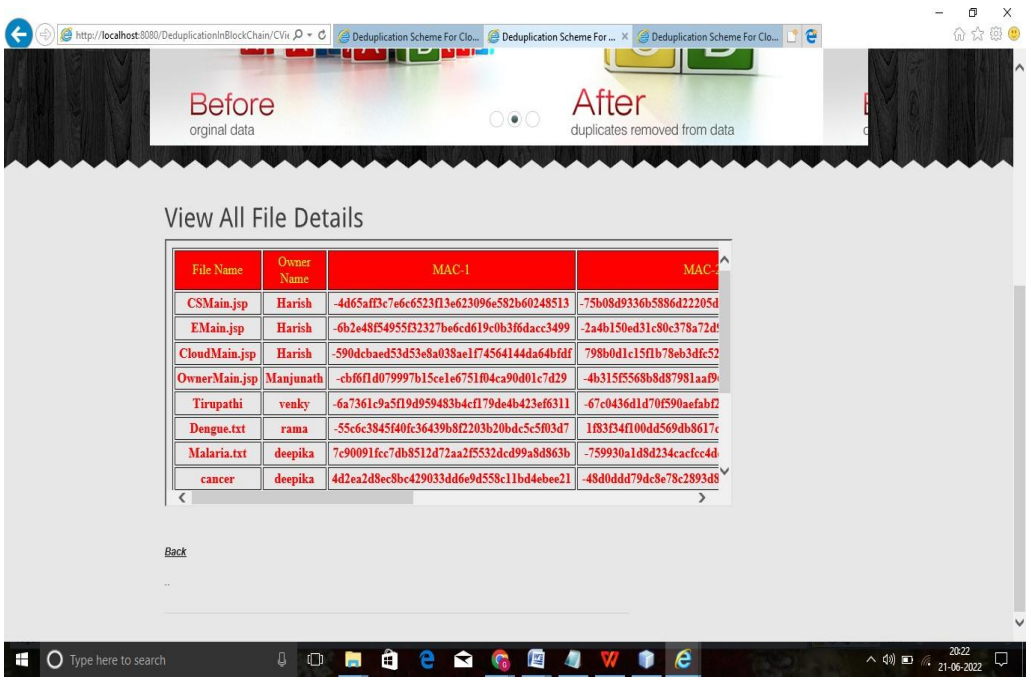


Fig 6.2.8 : All Blocks Page

O. The Cloud server can view all the Deduplication

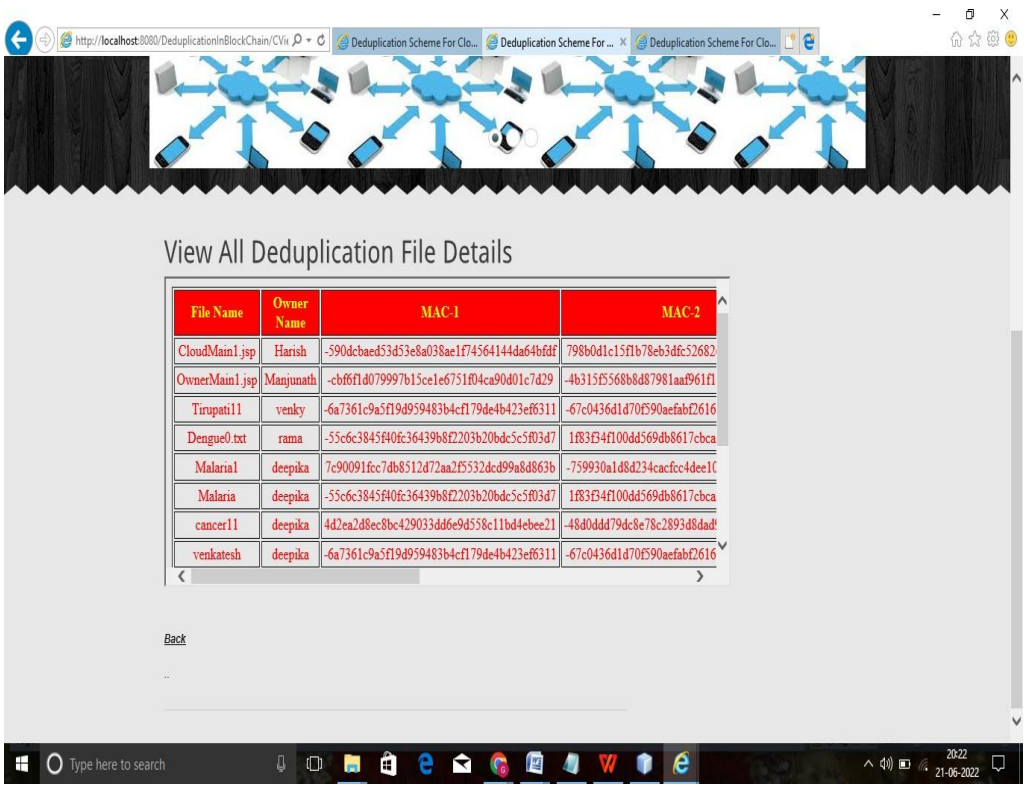


Fig 6.2.9: Deduplication Page

P. The Cloud server can view all the Results

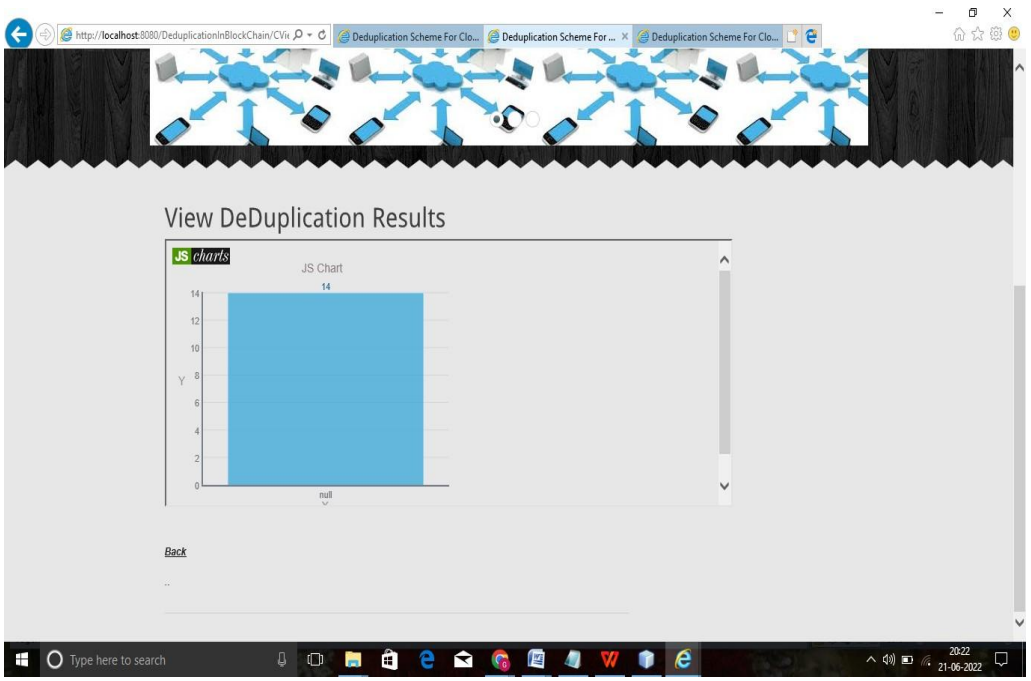


Fig 6.2.10 : Results Page

Q. The Cloud server can view all the File Request

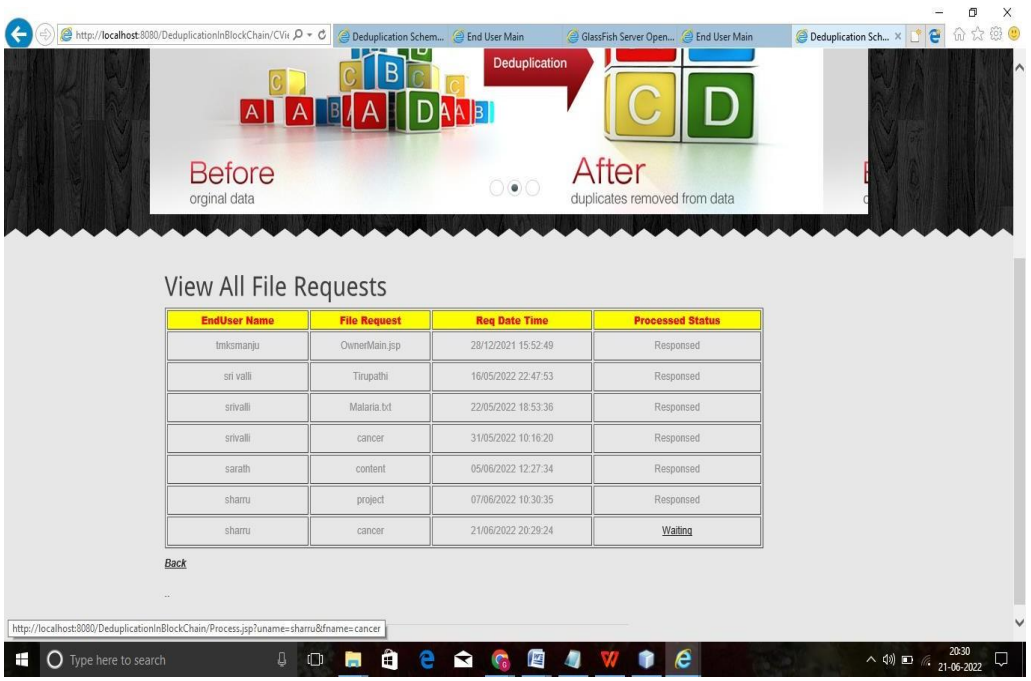


Fig 6.2.11 : File Requests Page

R. The Cloud server can view all the Time Delay Results

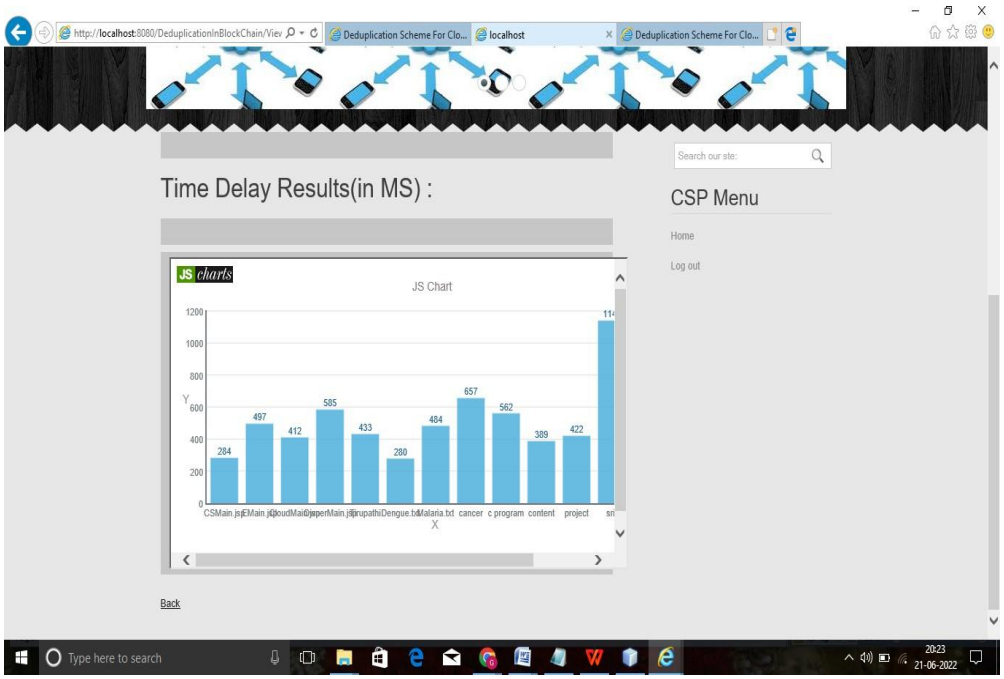


Fig 6.2.12 : Time Delay Result Page

S. The Cloud server can view all the Throughput Results

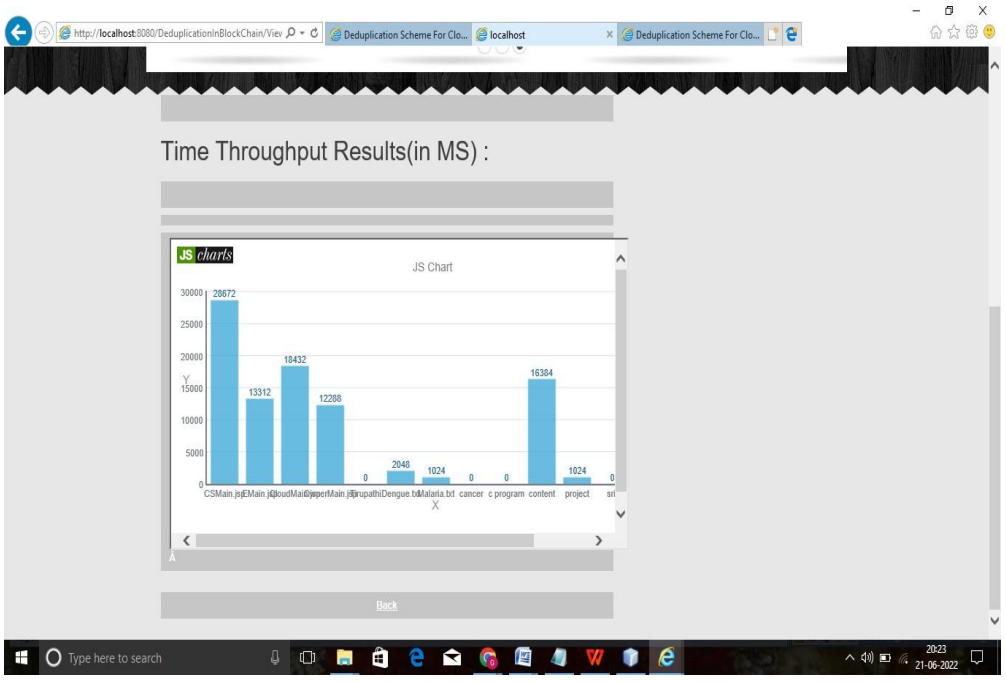


Fig 6.2.13 : Throughput Result Page

6.3 End User

T. The End User need to register and login in order to download the files.

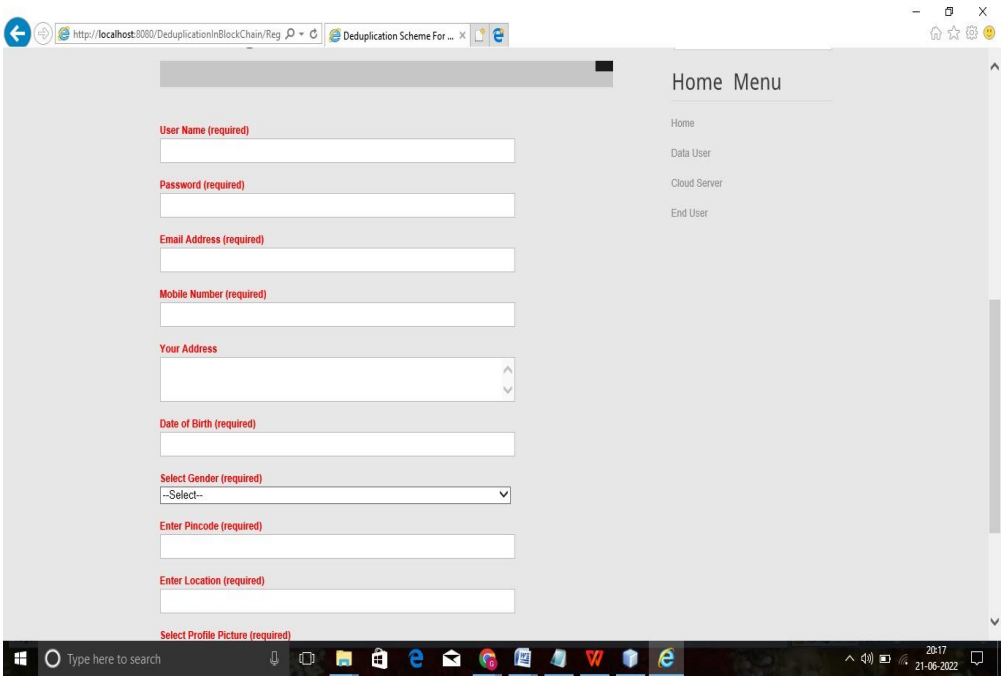


Fig 6.3.1 : End User Registration

U. The End User has login into his main page by using his credentials.

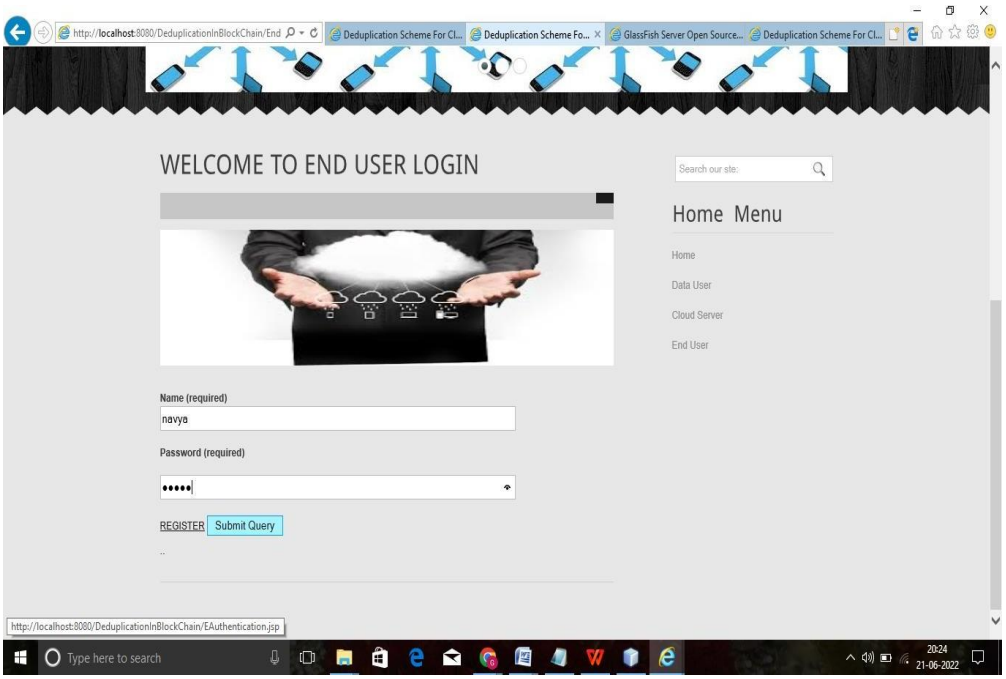


Fig 6.3.2 : End User Login

V. Now End User has requested a file to the cloud server.

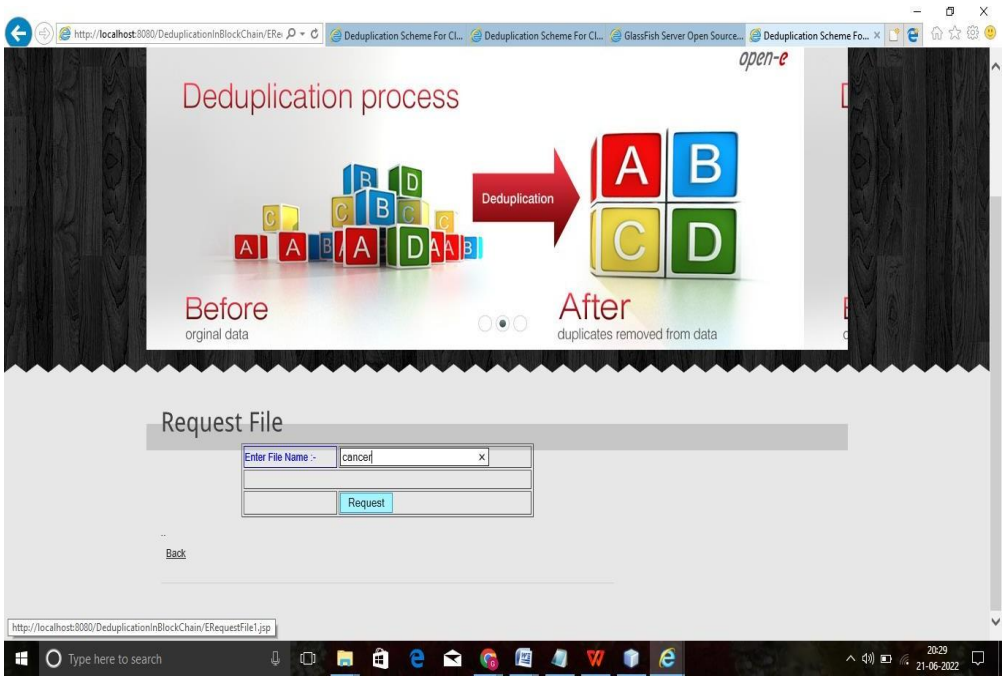


Fig 6.3.3 : Requesting File

W. End User can view the download File Response from the cloud server.

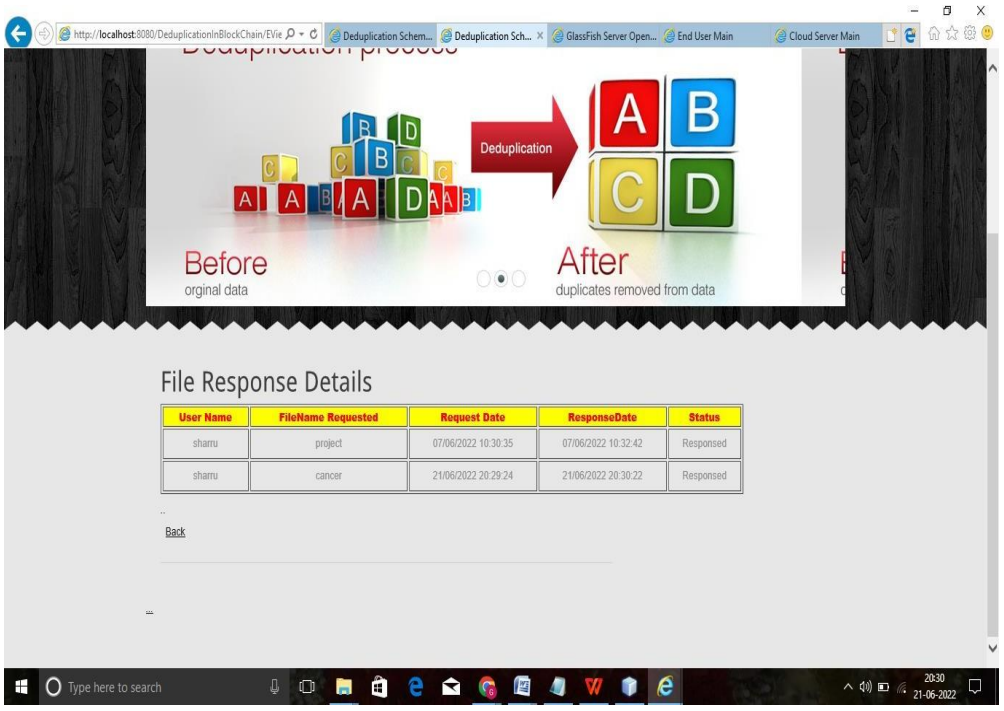


Fig 6.3.4 : File Response Page

X. Upon acceptance of the request the end user can decrypt and download the file.

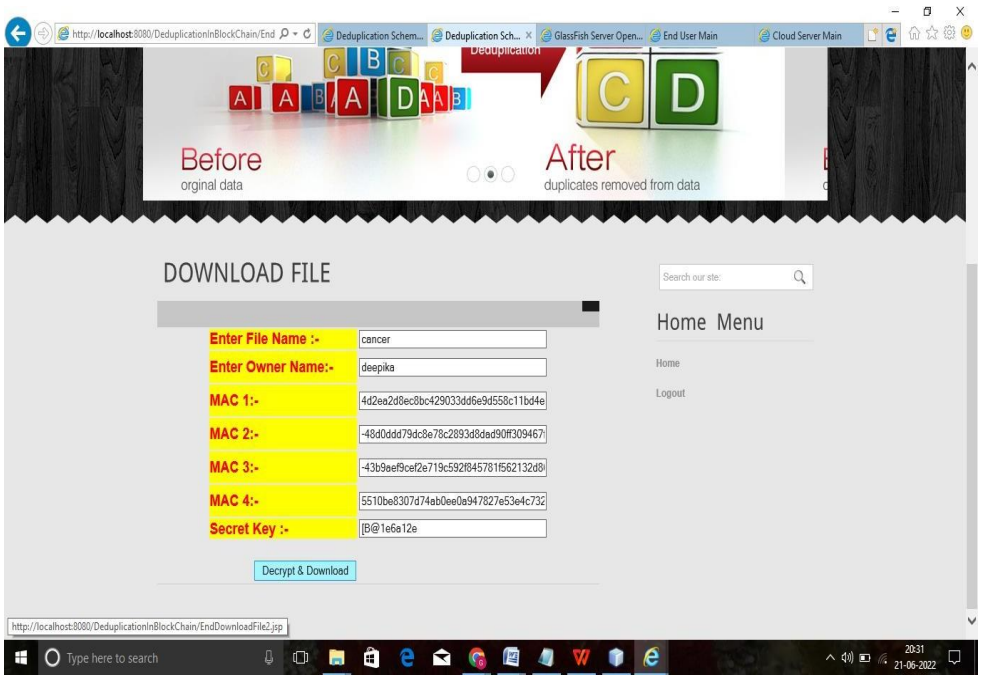


Fig 6.3.5 : Decrypting File

Y. The end user can download the file as shown in fig.

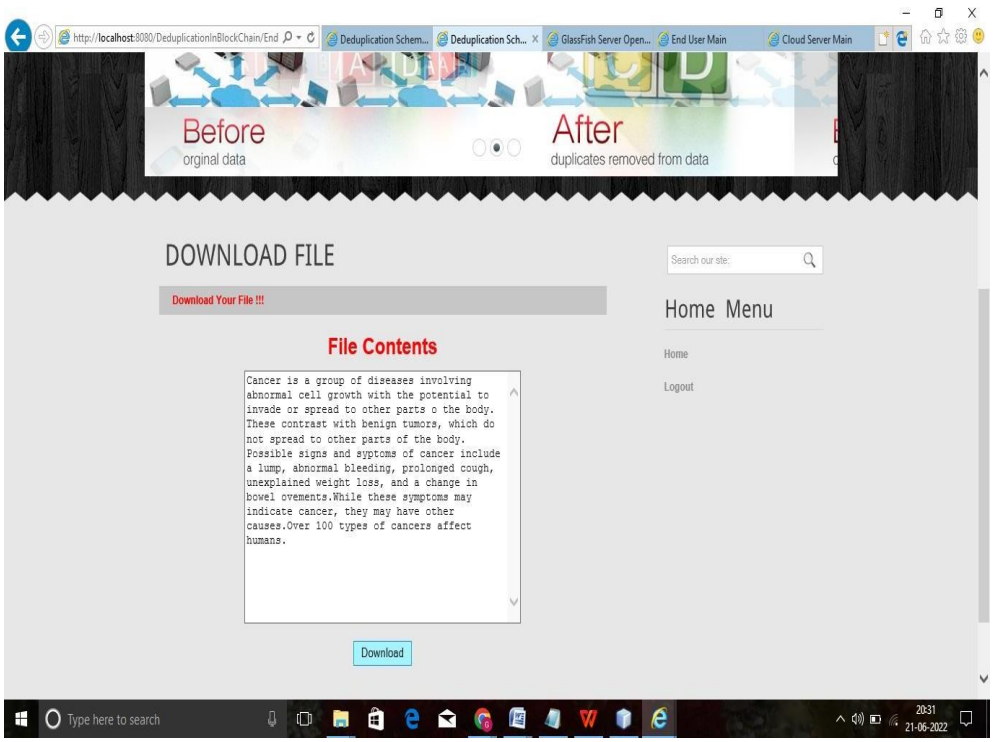


Fig 6.3.6 : Download File

CONCLUSION

The main goal is to remove the duplicate data files and improve the storage space in the cloud that is uploaded by the data user. So we proposed a Deduplication Scheme For Cloud Data Based On Convergent Encryption. Encrypted deduplication combines encryption and deduplication in a continuing way to provide confidentiality ensures for the physical data in deduplicated storage. It notably mitigates the metadata storage overhead in encrypted deduplication, even as retaining confidentiality ensures for each data and metadata.

Therefore to offer efficient storage for cloud data users and render excessive protection for information. Data Deduplication approach allows the cloud users to control their cloud storage area efficaciously by using keeping off storage of repeated information and store bandwidth. Convergent Encryption guarantees that the encrypted chunks derived from replica chunks nonetheless have identical content material, thereby being well suited with deduplication. Therefore Data deduplication eliminates excessive copies of records and significantly improves the storage area inside the cloud server. We assume that this way of storing and accessing data is much secure and have high performance. Our efforts are going on to solve the problem of security issues by storing repeated duplicate data in cloudcomputing environment.

REFERENCES

[1] Jingwei Li, Suyu Huang, Yanjing Ren, Zuoru Yang, Patrick P. C. Lee, Xiaosong Zhang, and Yao Hao, "Enabling Secure and Space-Effificient Metadata Management in Encrypted Deduplication," published by IEEE Transactions on Computers on 18.March.2021

[2] M. Bellare, S. Keelveedhi, and T. Ristenpart., "Message-locked encryption and secure deduplication," In Proc. of EUROCRYPT, 2013.

[3] M. Bellare, S. Keelveedhi, and T. Ristenpart, "DupLESS: Server-aided encryption for deduplicated storage," In Proc. Of USENIX Security, 2013.

[4] Taek -Young Youn1, Ku-Young Chang, "Authorized Client-side Deduplication Using Access Policy-based Convergent Encryption, "Journal of Internet Technology Volume 19 (2018) No.4 .

[5] D. R. Bobbarjung, S. Jagannathan, and C. Dubnicki. "Improving duplicate elimination in storage systems." ACM Transactions on Storage, ISSN: 424–448, 2006.

[6] J. Li, X. Chen, M. Li, J. Li, P. P. C. Lee, and W. Lou. "Secure deduplication with Efficient and reliable convergent key management". IEEE Transactions on Parallel Distributed Systems, ISSN:1615-1625,2014.