A Security Data Dynamics and Public Auditing Scheme for Cloud Storage

#### A Minor Project Report

*Submitted by*

#### Doma Bhargav [Reg. No.: RA2011028010069] Upanshu Bharadwaj [Reg. No.: RA2011028010083] *Under the Guidance of*

**Dr. M Saravanan**

Associate Professor, Department of Networking and Communications

*In partial fulfilment of the requirements for the degree of*

### BACHELOR OF TECHNOLOGY

**in**

### COMPUTER SCIENCE AND ENGINEERING

#### with specialization in Cloud Computing



**DEPARTMENT OF NETWORKING AND COMMUNICATIONS SCHOOL OF COMPUTING**

**COLLEGE OF ENGINEERING AND TECHNOLOGY SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

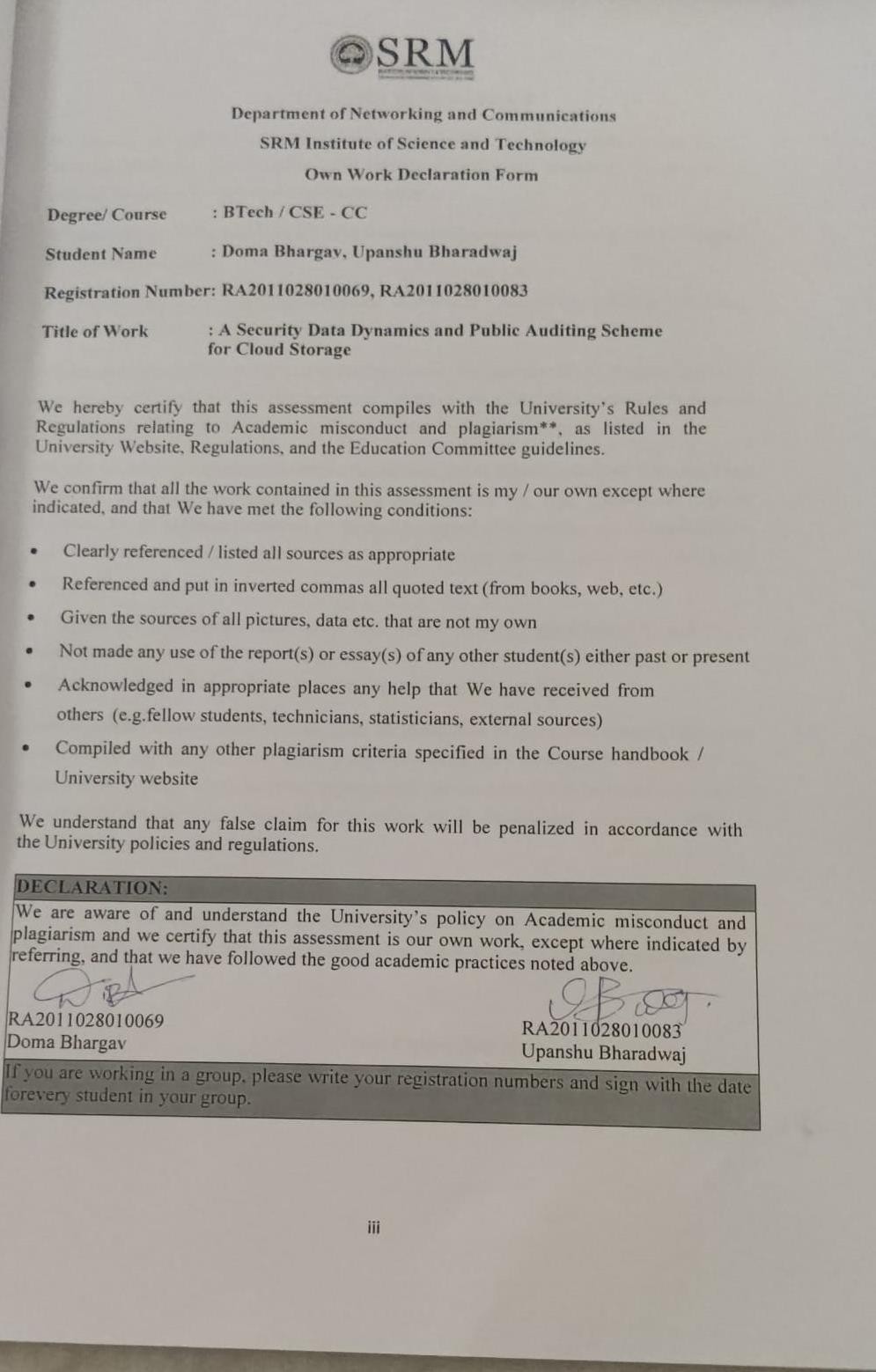
**(Under Section 3 of UGC Act, 1956)**

SRM NAGAR, KATTANKULATHUR – 603 203 CHENGALPATTU DISTRICT

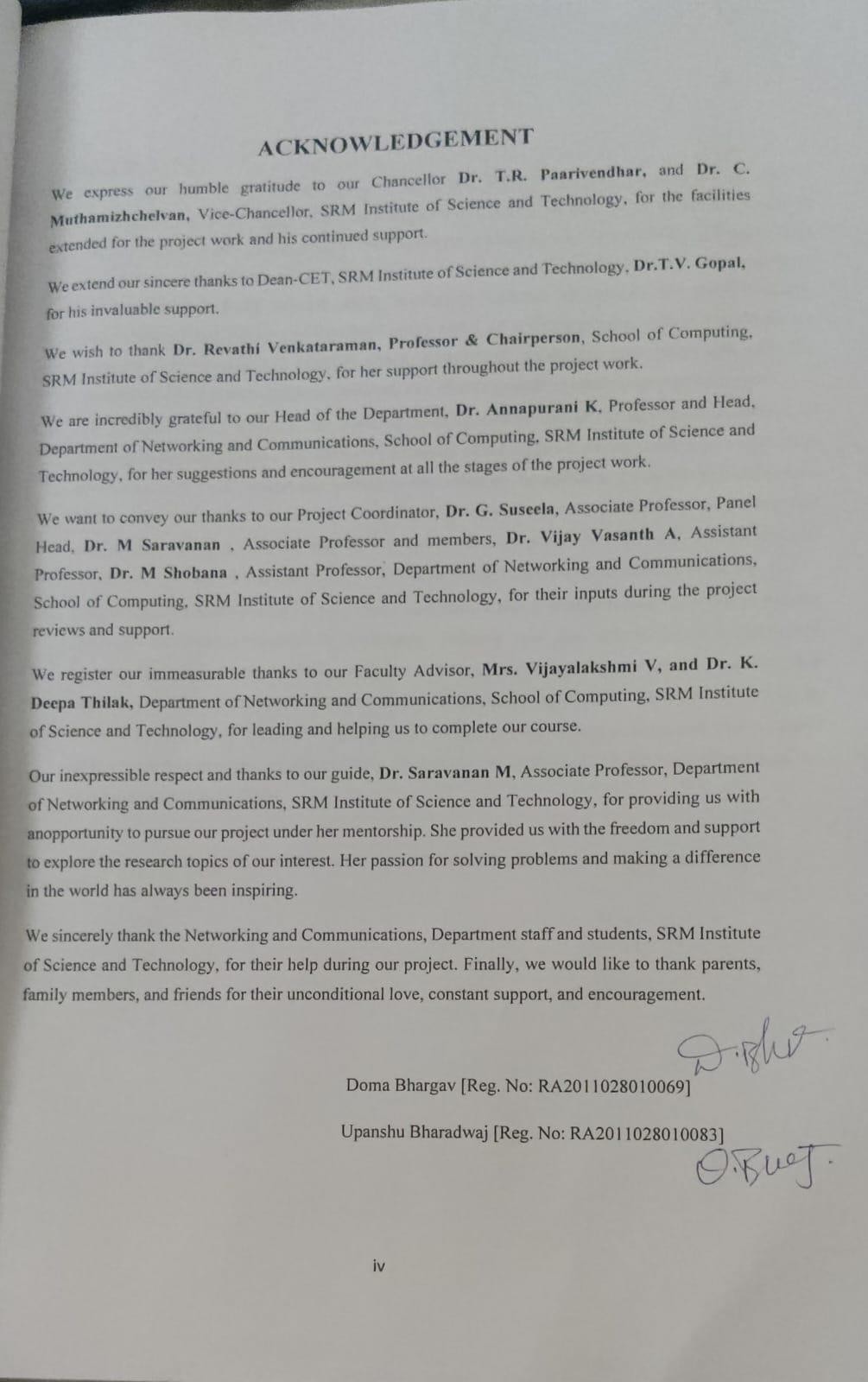
**NOVEMBER 2023**



ii



iii



iv

# ABSTRACT

* It is extremely quick and inexpensive computer services as well as data storage and a fast developing technology. Cloud service providers handle all data saved on servers based on their capacity as dataowners who worried about the legitimacy and dependability of the data. Has the ability to misuse or modify data for any unapproved user or individual. This study proposes a secure public auditing scheme using third-party auditors to validate the confidentiality, dependability, and integrity of data used in the cloud. In auditing scheme several algorithms are included such as SHA-512 integrity check, RSA-15360 public key encryption, and AES-256 encryption.To deal with addition, removal, and alteration data dynamics operations are used.
* To strengthen data security, the auditing scheme makes use of an advanced collection of cryptographic algorithms. The SHA-512 integrity check offers a strong method of verification and acts as a sentinel against any breach in data integrity. In addition, the public key encryption RSA-15360 enhances security by using cutting-edge encryption methods to guarantee that confidential data can only be accessed by those who are permitted. AES-256 encryption also strengthens the security infrastructure as a whole, protecting data from possible breaches.
* Essentially, this study presents a novel approach that surpasses traditional data security protocols. Through the implementation of third-party auditing and the utilization of sophisticated cryptographic algorithms, it offers a resilient structure for guaranteeing the privacy, stability, and accuracy of data within the constantly changing and growing domain of cloud computing. In addition to addressing immediate issues, this strategy presents itself as a preventative measure against possible dangers, promoting an environment of dependability and trust throughout the cloud computing ecosystem.

**Key words:** data owner, TPA, and CSS Open the file, Grant Request, Create Key, Download Document.

**CHAPTER NO**

### TABLE OF CONTENTS

**TITLE PAGE NO**

**ABSTRACT v**

LIST OF FIGURES viii

[LIST OF ABBREVIATIONS ix](#_TOC_250003)

1. [INTRODUCTION 10](#_bookmark0)
   1. [GENERAL 10](#_bookmark1)
   2. [PURPOSE 12](#_TOC_250002)
   3. [PROPOSED MEDTHOD 13](#_bookmark2)
   4. [SCOPE 14](#_bookmark3)
   5. MOTIVATION 14
   6. [CLOUD SERVICE PROVIDERS 15](#_bookmark4)
2. LITERATURE REVIEW 16
3. [PROPOSED METHOGOLOGY OF THE PROJECT 19](#_bookmark5)
   1. ARCHITECTURE 19
   2. USECASE DIAGRAM 20
   3. [ER DIAGRAM 21](#_bookmark6)
   4. ALGORITHMS 23
4. [TECHNICAL REQUIREMENTS 26](#_bookmark7)
   1. [FUNCTIONAL REQUIREMENTS 26](#_TOC_250001)
   2. [NON FUNCTIONAL REQUIREMENTS 26](#_TOC_250000)
   3. [HARDWARE REQUIREMENTS 27](#_bookmark8)
   4. [SOFTWARE REQUIREMENTS 27](#_bookmark9)
5. [IMPLEMENTATION AND RESULTS 28](#_bookmark10)
6. **CONCLUSION AND FUTURE SCOPE 32**
   1. CONCLUSION 32
   2. FUTURE SCOPE 33

**REFERENCES 34**

**APPENDIX A 36**

**APPENDIX B 49**

**PAPER PUBLICATION STATUS 58**

**PLAGIARISM REPORT 59**

|  |  |  |
| --- | --- | --- |
|  | **LIST OF FIGURES** |  |
| **Figure No.** | **Title** | **Page No.** |
| 1.1 | Cloud storage system design | 11 |
| 1.2 | Project flow | 13 |
| 3.1 | Architecture diagram | 24 |
| 3.2 | Use case diagram for Cloud Service Provider | 20 |
| 3.3 | ER Diagram of total mechanism | 21 |
| 3.4 | Class Diagram | 22 |
| 3.5 | Data Encryption and Decryption | 24 |
| 5.1 | Data Flow Diagram | 29 |
| 5.2 | Data Owner Reg Table | 31 |
| 5.3 | File Upload Status | 31 |
| 5.4 | File Request Status | 31 |

## LIST OF ABBREVIATIONS

AES - Advance Encryption Standard

TPA - Third Party Administrator

SHA - Secure Hash Algorithm

RSA - Rivest, Shamir, Adleman

HTML - Hypertext Markup Language

AWS - Amazon Web Service

GCP - Google Cloud Provider

CSP - Cloud Service Provider

ER - Entity Relation

# CHAPTER 1 INTRODUCTION

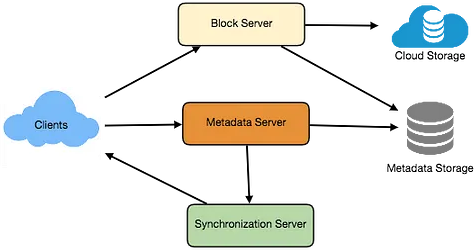
Data Owners can store and retrieve data and files online thanks to a technology calledcloud storage, which is usually provided by third-party service providers who look after data centres and storage infrastructure. It offers several advantages compared to traditional on-premisesstorage solutions[1].

#### General

According to research and market analysis, the global air conditioning market is expected to reach US$ 624 billion in 2023, from US$ 272 billion in 2018, with an annual growth rate of 18%. In today's world, cloud computing is an advanced technology that everyone uses, whether internally or outside.For computation and storage it uses cutting-edge, quickly developing technologies.For the lowest possible cost,compute and storage as a service is used. Three fundamental services were offered by the service model: software as a service (SaaS), platform as a service (PaaS), and infrastructure as a service (IaaS).

One essential cloud computing service is cloud storage. They include data availability, privacy, protection, and location; secure transmission is one important development in cloud security.There are some security challenges like threats, data loss, degradation, outside malicious attack, and multi-tenancy.By preserving the stored information cloud system maintains the integrity. Unauthorized users shouldn't have access to inaccurate activties. The provider truely maintains data reliability and integrity.Users store their private and confidential information on the cloud to ensure authentication and access control policies.Moreover, this all stored on cloud platforms.

The presentation of a data auditing scheme includes a secure system for storage.For checking info TPAs can conduct audits.Data on cloud is updated to date to increase several constraints. Private audibility is one of the two components of TPA management which enables the data owner to assess the integrity. Nobody is qualified to question the server about how the data is being handled.



**Figure 1.1 Cloud storage system design**

The figure 1.1 represents cloud storage system design which tells the relation between client and cloud server through meta data server.

###### Purpose

The aim of this project is to provide a useful and powerful global analysis and powerful data for cloud storage. The program strives to protect the information while ensuring that users have the authority to modify stored data. The main purpose is to conduct public audits so that independent auditors can verify the integrity of the data without accessing the actual data. By achieving these goals, the plan will reduce significant concerns about data security and privacy in the cloud while also improving the reliability and availability of cloud storage services.

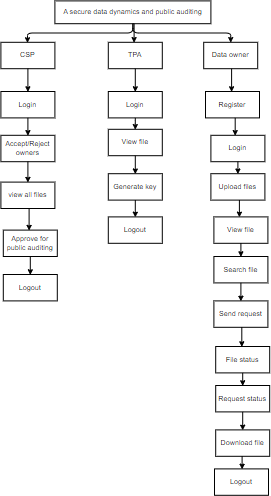
When it comes to cloud storage, where a great deal of private data is kept in remote locations, data security and integrity must be guaranteed. The goal of the project is to create a comprehensive framework that allows independent data integrity verification without direct access to the actual data by utilizing Third-Party Auditors (TPA) to conduct public audits. As a result, users and stakeholders will feel more confident in the transparent and accountable cloud storage ecosystem.

An essential component of data confidentiality in the cloud storage system is the use of the Advanced Encryption Standard (AES) algorithm. The project intends to encrypt data while it is in transit and at rest by putting AES into practice, protecting it from breaches or unwanted access. By doing this, issues with data security in a cloud environment are addressed and the overall privacy of user information is improved. The project also aims to integrate dynamic data operations, giving users the ability to safely edit the data that is stored. This preserves the integrity of the system as a whole while adding a layer of flexibility and user control. Users can interact with their data in real-time because data operations are dynamic, and the TPA makes it easier for audits to verify that changes are compliant with security guidelines.

By fulfilling these objectives, the project improves the dependability and accessibility of cloud storage services in addition to helping to lessen serious worries about data security and privacy in the cloud. By offering consumers a transparent and safe platform to store and manage their data, this creative plan hopes to increase user confidence in cloud-based storage options. The establishment of a public auditing scheme and secure data dynamics is an essential first step in strengthening the confidentiality, integrity, and user-centric control of data in the cloud, especially as the digital landscape develops further.

###### PROPOSED METHOD

This recommended testing method includes public key encryption (RSA-15360), integrity checking (SHA- 512), and encryption using AES-256 technology. And perform dynamic operations on data, often including insertion, deletion, and modification.



**Figure 1.2: Project flow**

###### SCOPE

It aims to provide a strong system that protects sensitive data and allows for limited, private, and accessible access. It entails putting the data in the file via encryption and decrypting it. Here, the framework seeks to offer clients safe cloud storage. The primary function of the AES algorithm is to generate a key that only the data owners will be able to access the data stored in the cloud.

###### Motivation

Building trust with consumers and businesses that use cloud storage services is one of the initiative's main goals. Even though there are many advantages to cloud storage in terms of affordability, scalability, and accessibility, worries about the security and privacy of private information continue to exist. By adding dynamic data security measures, the suggested solution seeks to close this confidence gap. Furthermore, the requirement for accountability and transparency in cloud storage systems is addressed by the establishment of public monitoring programs. Third-party entities conducting independently verified audits can provide benefits to various stakeholders, including users and regulatory bodies. This is a preventative step against future security breaches in addition to guaranteeing adherence to data protection laws. As a result of the transparency brought about by public monitoring, users can feel confident that their data is being handled carefully and in compliance with established security protocols.

The motivation behind cloud storage data security dynamics and public monitoring programs is to ensure data security, privacy, ownership, and accountability while complying with regulations and adapting to changing cloud security threats. This solution can provide users and organizations with the confidence and tools needed to reap the benefits of cloud storage

###### Cloud Service Providers:

Cloud service providers (CSPs) are companies that offer a variety of computing services and resources over the internet. These services can contains computing power, networking, storage, databases, analytics, software and more. Instead of organizations managing their own physical hardware and infrastructure, they can use the resources provided by these cloud service providers on a pay-as-you-go basis. Some well-known cloud service providers include:

 **Amazon Web Services (AWS):** A comprehensive and widely used cloud platform byAmazon.

 **Microsoft Azure:** Microsoft's cloud computing platform that provides a wide range ofservices and solutions.

 **Google Cloud Platform (GCP):** Google's suite of cloud computing services, includinginfrastructure, storage, machine learning, and data analytics.

 **IBM Cloud:** IBM's cloud offering, providing services such as virtual servers, storage, andapplication services.

 **Alibaba Cloud:** A leading cloud provider in China, offering a range of cloud computingservices.

 **Oracle Cloud:** Oracle's cloud services, providing databases, applications, andinfrastructure.

 **Digital Ocean:** Known for its simplicity and developer-friendly approach, Digital Oceanoffers cloud services with a focus on simplicity and ease of use.

 **Salesforce:** While primarily known for its customer relationship management (CRM)software, Salesforce also offers a cloud platform for building and deploying applications.

These providers have data centers located around the world, allowing users to access their services from virtually anywhere.

# CHAPTER 2 LITERATURE REVIEW

In this chapter, we discuss the previous research that has been conducted in this domain.

1. **Vinaya Sarmalkar; Yatin Gandhi; R. Patil Rashmi "Cloud Storage Security Using Deduplication Method 2020"**

In this paper, we identify several problems, such as the inability to assign a new hash tag to client-side

deduplication after file updates. Thus, the tools utilized here are the RDPC protocol, C++, homomorphic hash algorithm, and dynamic ownership, which fails.

In brief: We present an innovative statistically sound plan, guarantee that the problems we observed earlier will be resolved, and prevent the formation of duplicates. We encrypt the data in the file using the AES and SHA algorithms. The data owner's stored data is not accessible to unauthorized parties.

1. **Hemraj Lamkuche, Sunil Kumar, and Dilip Kumar "Cloud Computing Application: Secure- Drive 2021 for Secure File Storage"**

We identify a few problems in this paper, which are Safe Data Transfer utilizing secure communication

protocols, such as HTTPS, to protect files during upload and download, and making sure that data is kept private while in transit. Here, HTML, CSS, Blowfish, Triple DES, and Phyton are the tools that are used. Cloud-based algorithms were employed.

In brief: the primary benefits of our systems are that they Thus, we offer algorithms that get around these problems and function better. In this case, they merely use a few apps to secure the data, but we use the AES algorithm and TPA to create files that are more secure and prevent other data owners from accessing the data without a key. CSS also plays a significant part in approving and rejecting the owners' requests.

1. **P. Baby Shamini; K. S. Megavarshini; D. C. Joy Winnie Wise A QR Code-Based Real-Time Auditing System for Safe Cloud Storage in 2020 Concurrency Management:**

A Look at Mobile Devices These are some of the issues in this paper; while using QR codes to secure the data is an excellent method, there are still a lot of issues that need to be resolved, such as the camera scanner not working properly and the need to take a long time but efficiently to send some keys to a respected mail address. To address these issues, the authors use HTML, CSS, Python, and Apache MariaDB among other

tools. MariaDB is the database, and HTML and CSS are used for the website.

In this study, we tackle three important problems related to concurrency control and mobile devices. Thus, the data is not secure because they did not even employ encryption algorithms; instead, it is secured by a single security mechanism known as QR.

1. **T. Longbin; Dai Wenyun. A Framework for Safe and Expandable Key-Value Stores, 2017.**

It is not possible to transfer queries from one database to another since key-value databases lack a querying language. In this paper, we addressed this issue, which is entirely based on the question. One data base to another receives the data transfer. A key value will be used to access the data, which is primarily focused on scalability. They employed cutting-edge technologies like Google Bigtable, Amazon Dynamo, Python, and flash memory.

Summary: As cloud computing has become more widely used, worries about the security of the data being

outsourced have unavoidably increased. Because of the resource limitations of mobile devices, security solutions must send all computing-related tasks to the cloud for execution.

1. **Tanuj Jain, Nidhi Nair, and Mihir Gada Cloud Computing Application for Secure File Storage:** Secure-Drive 2021The literature review emphasizes how important safe file storage is when it comes to cloud computing. It examines the fundamental idea of secure transmission as well as the instruments and technologies used to accomplish this. Through an analysis of an application such as "Secure-Drive," which makes use of web development technologies, encryption algorithms, and secure communication protocols, this survey adds to a better understanding of the steps taken to guarantee that data in cloud -based file storage systems is secure and confidential.

Summary: The literature review is lacking in depth, organization, citations, and accurate definitions of technical terms, among other problems. There is a typographical error and an ambiguity regarding the authors' roles. Additionally, a section on future research directions is absent, as are the most recent research updates. To create a better survey, these problems must be fixed.

1. **S P. Calista Bebe, Akila D. Orchini Comparative User Authentication-Based Streebog Hash Function**

for Safe Cloud 2020 Data Preservation An enhanced comprehension of user authentication and data

security in relation to cloud storage is made possible by this survey of the literature. The "Orchini"paper

aims to give readers important insights into this developing field by examining the difficulties, datasets, and methodologies used in the study.

Summary: Crucial facets of cloud data security are covered in the paper "Orchini: User Authentication and Streebog Hash Function for Secure Cloud Data Storage" by P. Calista Bebe and Akila D. Nonetheless, it has problems with the intricacy of its title, the need for a more thorough examination of problems and solutions, and the need to define the significance of datasets and acronyms. To improve the overall quality and impact of the paper, additional clarity regarding the methodology and specific contributions is needed.

1. **IK Meenakshi and Sudha George. Employing TPA for Cloud Server Storage Security.2014 issues of International Journal of Advanced Research in Computer Science and Technology.**

The owner can always rely on the cloud server to store their data and retrieve it when required. Data integrity in cloud storage is a critical concern since many users store their data there. The owner hopes that after transferring the datato the cloud, their apps and data will be protected. Nevertheless, there's a chance that the owner's data will be lost or changed. For the purpose of validation in this case, the user needs to download the data.

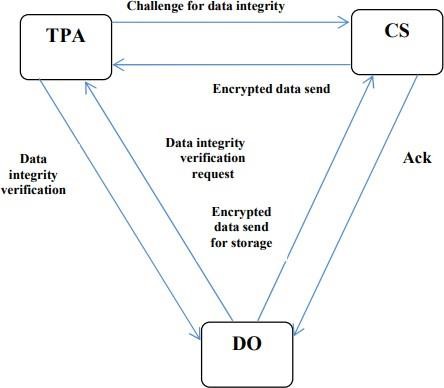
1. **An effective auditing scheme for cloud data storage security was developed by J Agarkhed and R Ashalatha.[ICCPCT, 2017].**

The requirement to store and oversee a show is satisfied by cloud computing.For information access and security remarkable innovations, such as parallel and guide diminish.The owner also gains from the release of the load of maintaining and storing native data, as well as from no longer having to worry about security and storage. A creative approach is required to guarantee the on-demand data's accuracy is modified and that the cloud owners have had a chance to verify it. A cutting-edge secure cryptography hashing algorithm is used for the encryption and file division. A private and public key are given to the user for reliable data file retrieval after the data is uploaded. A suggested modified RSA cryptosystem algorithm is used to generate those keys. Occasionally, an efficient audit of the data file is conducted using a multilevel hash tree algorithm. The efficacy of the suggested model in are demonstrated by the implementation and the ensuing outcomes.

# CHAPTER 3

This is the Architecture of the project the key features are TPA and CS it shows the encryption process and the data flow process.

###### Architecture Design

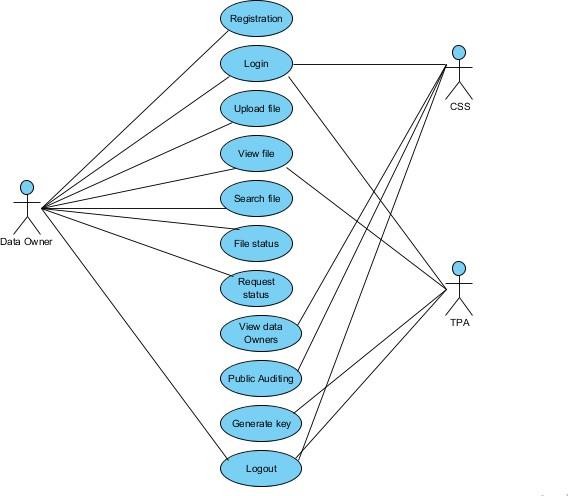


**3.1-** Architecture diagram

This figure, explains about the architecture of the project what the TPA will able to do and CS do data encreption and all.

#### Use case Diagrams:

* + - According to the Unified Modeling Language (UML), a use case diagram is a particular kind of behavioral diagram that is produced from and defined by a use case analysis. Its objective is to provide a graphical summary of the functionality that a system offers in terms of actors, use cases (representations of their goals), and any interdependencies among those use cases. A use case diagram's primary goal is to display which actors receive which system functions. It is possible to illustrate the roles of the system's actors.



**Figure 3.2 Use Case Diagram**

###### ER Diagram:

An entity-relationship model, or ER model, uses an entity relationship diagram (ER Diagram) to illustrate how a database is structured. An ER model is a database's design or blueprint that can be used when the database is eventually put into use. The entity set and relationship set are the two primary parts of the E-R model.

An entity-relationship diagram (ER diagram) illustrates this. A collection of related entities that may or may not have attributes is called an entity set. An ER diagram illustrates the entire logical structure of a database by illustrating the relationships between tables and their attributes. In terms of DBMS, an entity is a table or an attribute of a table in a database.

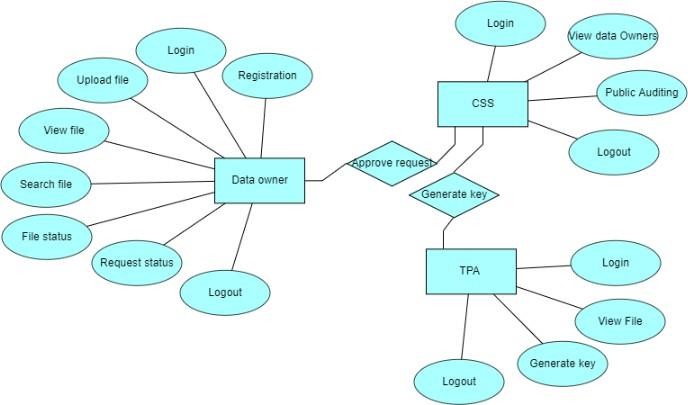
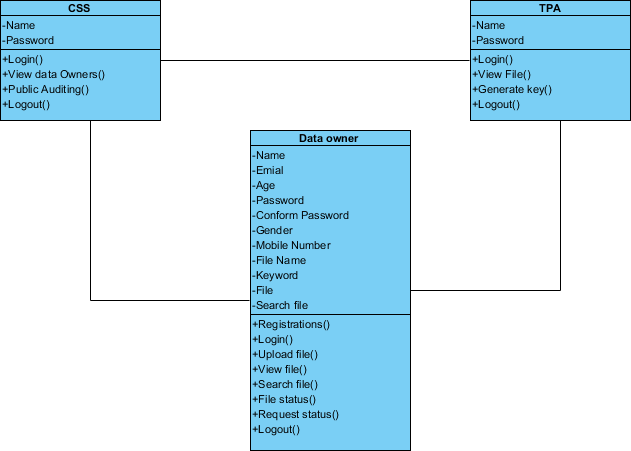


Figure 3.3 ER diagram

**Class Diagram:** A class diagram in software engineering is a sort of static structure diagram that

displays the classes, attributes, operations (or methods), and other components of a system. It is created using the Unified Modelling Language (UML).



**Figure 3.4**: Class Diagram

### ALGORITHM

* 1. **AES Algorithm:**

An important component of data security, especially for cloud storage, is the symmetric encryption algorithm known as the Advanced Encryption Standard (AES). The National Institute of Standards and Technology (NIST) adopted AES as a standard in 2001 after a rigorous selection procedure.

AES inn Cloud Storage:

AES is frequently used by cloud storage services to encrypt data in order to guarantee its integrity and confidentiality. Data stored in the cloud may be vulnerable to threats and travel across multiple networks. By guaranteeing that only authorized users possessing the proper decryption key can access and decipher the stored information, AES encryption helps reduce the risk of unauthorized access.

Key Management in Cloud Storage:

Key management is a crucial component of using AES in cloud storage. The confidentiality of the encryption keys determines the security of the encrypted data. To protect encryption keys, cloud service providers usually put strong key management procedures in place. To guarantee that only approved people or systems can handle or access the keys, this includes safe key storage, key rotation, and access controls.

Performance Consideration:

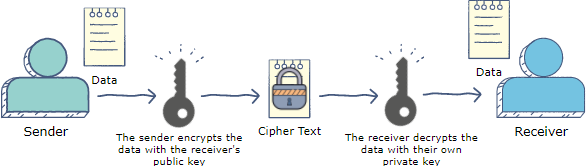
AES is made to offer a decent trade-off between performance and security. Its computational efficiency, which makes it appropriate for use in resource-constrained environments like cloud storage systems, contributes to its widespread adoption. For data to be encrypted and decrypted swiftly and without having a major negative influence on the overall performance of cloud storage services, the algorithm's efficiency is essential.

private key confidential, and authorised donors can either participate with or obtain the public key.

**AES Algorithm**

The National Institute of Standards and Technology (NIST) of the United States created the Advance Encryption Standard (AES) in 2001 as a specification for the encryption of electronic data. Even though AES requires more care to develop, it is currently widely used due to its superior

strength compared to DES and triadic DES. AES is a block cypher. One can achieve a critical sizeof 128/192/256 bits. Data translation is done using 128-bit GBTs.



**Figure 3.5** Data encryption and Decryption

* 1. **Rivest-Shamir-Adleman:**

In order to safeguard encryption keys and secure communication channels, RSA is a popular asymmetric encryption algorithm used for digital signatures and key exchange.Forests.

The RSA (Rivest-Shamir-Adleman) asymmetric encryption algorithm is essential to a cloud storage project. It fulfills a number of purposes that are crucial to the integrity and security of data. It first establishes secure communication channels by facilitating secure key exchange. Furthermore, it facilitates the creation of digital signatures, thereby authenticating and maintaining the integrity of data. Last but not least, RSA improves privacy and data security in dynamic data operations. In conclusion, RSA is essential for guaranteeing data authenticity and secrecy in cloud storage projects that involve safe data dynamics and open auditing.

* 1. **Secure Hash Algorithm 256-bit:**

Data checksums or hashes are computed using the cryptographic hashing algorithm SHA-256 to ensure data integrity. SHA-256, or Secure Hash Algorithm 256-bit, is an essential part of a secure data dynamic and public auditing scheme for a cloud storage project. Since it creates distinct digital fingerprints, or hashes, for data, SHA-256 is essential for guaranteeing data authenticity and integrity. Particularly in dynamic cloud storageenvironments where data is constantly changing, these hashes are used to confirm data integrity. SHA-256is a vital tool for preserving the dependability and credibility of data throughout its lifecycle in the cloud because of its capacity to offer real-time verification of data integrity and its participation in public auditing schemes.

**CHAPTER 4**

# TECHNICAL REQUIREMENTS

#### Functional Requirements:

These are the specific demands made by the end stoner for essential features that the system must give. These functionalities must be an integral part of the contractual agreement. They're generally expressed in terms of inputs to be handed to the system, the operations to be performed, and the anticipated labors. Functional conditions are basically the stoner's unequivocal requests that are directly observable in the final product, in discrepancy tonon-functional conditions. exemplifications of functional conditions include Authentication of druggies whenever they log into the system. Initiating a system arrestment in response to a cyber-attack. transferring a verification dispatch to druggies when they register for the first time on a software system.

#### Non Functional Requirements:

Non-functional conditions Pertain to the quality constraints that the system must cleave to as quested in the design contract. The degree of emphasis or perpetration of these factors can vary from one design to another. They are also appertained to as non-behavioral conditions and primarily address issues similar as Portability, Security, Maintainability, and trustability.

* + - Scalability
    - Performance
    - Reusability
    - Inflexibility

The emplifications of non-functional conditions include, Emails must be transferred with a quiescence not exceeding 12 hours from the circumstance of a affiliated exertion. Each request's processing should be completed within a 10-alternate timeframe. The website must load within 3 seconds when the number of concurrent druggies exceeds 10,000.

###### Hardware Requirements:

* Processor - I3/Intel Processor
* RAM - 8GB and above
* Hard Disk - 128 GB
* Key Board - Standard Windows Keyboard
* Mouse - Two or Three Button Mouse

###### Software Requirements:

* Operating System : Windows 10 and above
* Server-side Script : Python 3.6
* Libraries Used : MYSQL
* Framework : Flask.

Front End: HTML, JSP; Application Server: Tomcat 7.0 Java Server Pages with server-side scripting.

JDBC-based database connectivity

# CHAPTER 5 IMPLEMENTATION AND RESULTS

1. **Data Owner:**

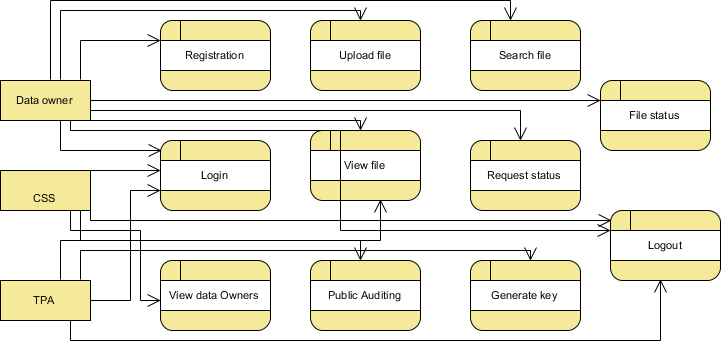
The data owner is essential in guaranteeing the security and integrity of their data stored in the cloud under a secure data dynamic and public auditing scheme. There are numerous obligations such as Data Import It is the data owner's responsibility to safely upload their data to the cloud storage service. To safeguard the data while it is in storage, this may entail access control and encryption techniques. Establish policies and access controls to limit who can view and alter data stored in the cloud. Defining user roles and permissions, audit preparation to create and maintain metadata, and audit information for the data may all be part of this. Over time, the integrity of the data can be confirmed using this metadata. Procedures to verify the accuracy of the data that has been stored are started by routine auditing. As a response to audit challenges, this may entail comparing the metadata created or the stored data with the original data. If a third party or cloud service contests the data's integrity, the data owner must present proof of data integrity derived from the metadata that has been stored. Important Prior to now, management Oversee and rotate the encryption keys that are used to protect the data, keeping an eye on them and sending alerts when necessary. Keep an eye out for any unusual activity or illegal access to the cloud storage service. And it has different module.

Register: The owner of the data may register and log in using proper credentials. File Upload: The data supplier is able to upload the file.

View File: The data owner can check if a file was uploaded correctly by viewing it once.

File Search: A user can perform a keyword-based search to find a file. If the file is found, the user can view it and send a request to the cloud to download it.

View File Status: The individual submitting the request has the ability to see the file's status. Request Status: After the cloud and TPA have accepted the request.



**Figure 5.1**: Data Flow Diagram

1. Cloud provider login:

It requires multiple steps. I will explain in detail below:

Lists of controls over access (ACLs) Control who can read, write, and alter data by implementing access control lists. As access rights are modified, these lists should be periodically checked and updated.

Versioning of your data can help you monitor changes over time. Make sure this feature is enabled. When

updating dynamic data, this is crucial. Ensuring data integrity can be achieved through the use of cryptographic hashes or checksums. To find any illegal modifications, periodically compare data to these integrity checks.

Digital Signatures: To ensure the authenticity of data, use digital signatures. In order to make sure that the data hasn't been altered during transmission, this is essential.

Continual Evaluations: Audit your cloud storage on a regular basis to look for any irregularities or

unapproved access. External auditors as well as your company are capable of doing this.

Secure Communication: Make sure that your systems and the cloud provider are communicating in a secure manner at all times. To secure data while it's in transit, use VPNs and HTTPS protocols. Install security updates on a regular basis to minimize potential vulnerabilities in your systems and cloud services.

1. Third party authority:

When it comes to confirming the accuracy and integrity of data stored in the cloud, a third -party authority usually plays a pivotal role in a secure data dynamic and public auditing scheme. which usually is a separate organization that the user and the cloud provider trust, becomes involved. Public keys or user and cloud provider credentials may be in the possession of this authority. Additionally, it has various modules: Authorize owners by logging in.

Generate key to user: An owner's specific key is generated by authority.

Algorithm Implementation:

A round key is a special type of encryption used in encryption. These are used for a data array containing exactly one block of data (data to be encrypted) as other methods. We call this sequence the sequence of events.

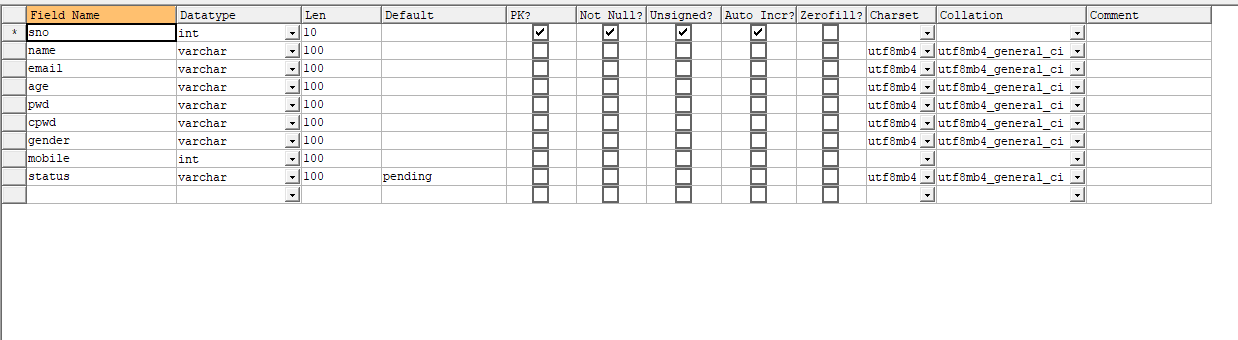
The following AES encryption method is used for 128-bit blocks: The round key is determined using the shortcut key.

Initialization block data (white text) is in the event string. Add the first key to the first state array. Nine state update iterations are performed. Be the last and tenth of the state's business. Copy the encrypted data (ciphertext) from the final state array. The action required for the tenth round is different from the first nine rounds, so it is listed as "the last tenth round after the nine rounds."

Encrypt 128-bit blocks. Since AES works in byte ranges, we first convert 128 bits to 16 bytes. Although we use the word "modified" most of the information will be saved this way. Four rows and four columns form a two-dimensional byte array for RSN/AES operations. First, the 16-byte file is encrypted.

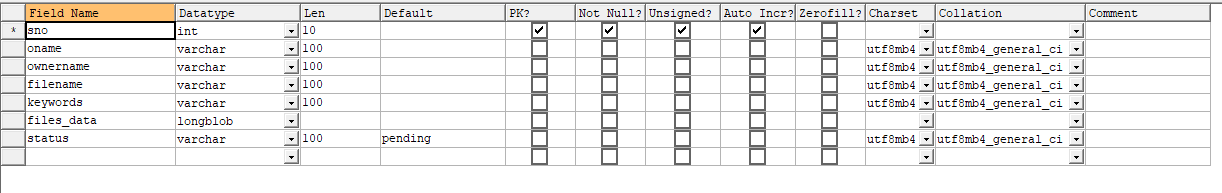
**STEPS FOR EXECUTING THE PROJECTS**

1. Install the necessary packages first.
2. Specifying the links between databases.
3. Establishing each database table that will be utilized for this project.
4. Update the HTML and CSS pages.
5. Utilize the components to build the project.
6. Execute the Python file (app.py). Copy the URL link. Paste it into any browser. Then, adhere to the screen shots below.

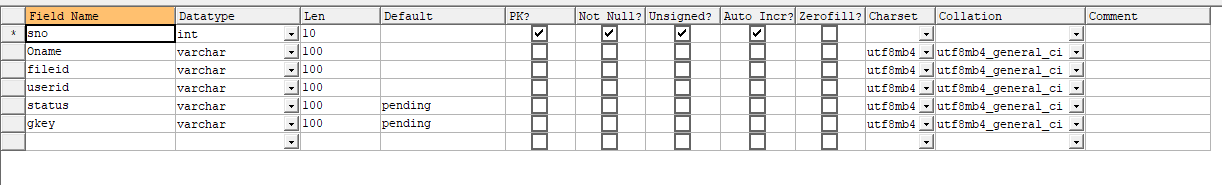


**Figure 5.2**. Data Owner Registraion table:

The Figure 5.2 shows the Login details of the Data Owners.



**Figure 5.3 Fil**e upload status The Figure 5.3 shows the file upload status of the data owners



**Figure 5.4** File request status The Figure 5.4 shows the file request status by the data owner

**CHAPTER 6 CONCLUSION AND FUTURE SCOPE**

This chapter explains the conclusion of our work and the possible research works that can be done in the same domain.

#### Conclusion:

In conclusion, our study gave the idea on, Storing the data securely on the cloud is one way to conduct a secure audit. The various techniques are used by the prospective to ensure that TPA cannot access content related to the robustness.An operation is suggested for information dynamics that deals with insertion, deletion, and modification. Then, the CSS should grant the data owner's request. The key that the owner needs to access the data is generated by the TPA. There is potential use for this application in the software industry.Integrating data security and general monitoring solutions into cloud storage systems is an important step in building trust and confidence in cloud-based systems. However, it is important to ensure that these solutions are carefully designed and implemented with a focus on security, efficiency and users. Products want to take full advantage of this technology. This will help organizations and individuals feel confident with cloud storage when managing their data. This application is available for commercial software. He presented the idea that one way to manage security is to store data securely in the cloud. The system uses SHA-512, RSA-15360 and AES-256 algorithms to ensure that TPA cannot access any information related to the audit process. We recommend dynamic addition, deletion and replacement operations for files. The data owner's request must be approved by CSS. The application will be useful in the software industry.

#### Future Scope

We hope to be able to check data in bulk in the future. Future improvements canbe made in various areas to further improve its results. These may include using simpler access control systems to prevent unauthorized access, integrating complex encryption algorithms to improve data protection, using technology learning techniques for invisible detection and intrusion prevention, and using distributed devices to improve crime prevention and scalability. Integration of blockchain technology can also be beneficial for transparency and accountability by providing an immutable, fair decision-making process.

Distributed storage system: Distributed storage system can increase scalability and security. This reduces the possibility of data loss due to hardware failure or outage due to data being constantly stored on multiple servers or locations. While the server is down, data can be easily retrieved from another network node, ensuring data availability and durability.

Advanced encryption algorithms: Various encryption algorithms can be integrated into the system to improve data protection. This includes techniques such as homomorphic encryption, which allows encrypted devices to operate without beingdecrypted, and post-quantumencryption, whichprevents future transactions fromaffectingthe current encryption process.

Machine Learning for Anomaly Detection: Machine learning techniques can be used to instantly detect anomalies and potential vulnerabilities. By analyzing user behavior and input patterns, machine learning algorithms can detect differences in specific messages and trigger alerts or responses. Automatic security when activity is detected. This approach helps prevent infiltration.

# References

1. R. Patil Rashmi; Yatin Gandhi; Vinaya Sarmalkar Secure Cloud Storage with Deduplication Technique 2020
2. Nidhi Nair; Tanuj Jain; Mihir Gada Secured File Storage in Cloud Computing

Application: Secure-Drive 2021

1. Sunil Kumar, Dilip Kumar, Hemraj Lamkuche TPA Auditing to Enhance the Privacy and Security in Cloud Systems
2. P. Baby Shamini; D. C. Joy Winnie Wise; K. S. Megavarshini A Real Time Auditing System using QR Code for Secure Cloud Storage 2020
3. Longbin Chen; Wenyun Dai. A Design for Scalable and Secure Key-Value Stores. 2017
4. P. Calista Bebe, Akila D. Orchini Similarity User Authentication Based Streebog Hash Function for Secured Data Storage in Cloud 2020
5. The global cloud computing market report 2019.
6. J Agarkhed, R Ashalatha-” An efficient auditing scheme for data storage security in cloud”.2017[ICCPCT].
7. B. L Adokshaja, and S.J.Saritha,”Third Party Public Auditing on Cloud Storage using the Cryptographic Algorithm”ICECDS-2017.
8. SK Saroj, G Noida, SKChauhan, AK Sharma “Threshold cryptography based data security in cloud computing”.S Vats-2015.
9. AbhishekMohta, Ravi Kant Sahu, Lalit Kumar. “Robust Data Security for Cloud while using Third Party Auditor”. International journal of advanced research in CSE (IJARCSE), Volume 2, Issue 2, February 2012.
10. IK Meenakshi and Sudha George. Cloud Server Storage Security using TPA.International Journal of Advanced Research in Computer Science and Technology (IJARCST) ISSN: 2347- 9817, 2014.
11. Qian Wang, Cong Wang, KuiRen, and Wenjing Lou, and Jin Li. Enabling Public Auditability and Data Dynamics for Security in Cloud Computing. Parallel and Distributed Systems, IEEE Transactions on,22(5):847-859,2011.
12. KanYang,XiaohuaJia.”An Efficient and Secure Dynamic Auditing Protocol for Data

Storage in Cloud Computing, IEEE Transaction on (1- 10), 2012.

1. W.Stalling,”Cryptography and network security,”LPE Sixth Edition,ISBN-978-013-335- 4690.
2. Kerry Maletsky,”RSA vs ECC comparison for embedded system”Atmel8951.
3. MeilianaSumagita,ImamRiadi,”Analysis of secure hash algorithm(SHA) 512 for encryption process on web based application”IJCSDF-7(4):373- 381.

## Appendix A Sample Code

**Python code:**

from flask import Flask, render\_template,request,session,redirect,url\_for import pymysql

import pandas as pd

from random import \*

app=Flask(\_name\_) app.config['SECRET\_KEY']='f9bf78b9a18ce6d46a0cd2b0b86df9da'

db = pymysql.connect(host='localhost',user='root',password='',db='secure\_data\_dynamics',port=3307) cursor=db.cursor()

app=Flask(\_name\_)

@app.route("/") def index():

return render\_template("index.html")

@app.route("/home") def home():

return render\_template("Home.html")

@app.route("/dataowner",methods=["POST","GET"]) def dataowner():

if request.method == "POST": name = request.form["name"] email = request.form["email"] age = request.form["age"] pwd = request.form["pwd"] cpwd = request.form["cpwd"]

gender = request.form["gender"]

mobile = request.form["mobile"]

sql = "insert into dataowner (name,email,age,pwd,cpwd,gender,mobile) values (%s,%s,%s,%s,%s,%s,%s)"

values = (name, email, age, pwd, cpwd, gender, mobile) cursor.execute(sql, values)

db.commit()

return render\_template("dataowner.html", ms="success") return render\_template("dataowner.html")

@app.route("/dataownerlogin",methods=['POST','GET']) def dataownerlogin():

if request.method=='POST': name = request.form["name"] pwd = request.form["pwd"]

sql="select \* from dataowner where name=%s and pwd=%s and status='accepted'" val=(name,pwd)

cursor.execute(sql,val)

Results=cursor.fetchall() if len(Results)>0:

session["dataownerloginid"]=Results[0][0] session["dataownername"]=Results[0][1]

return render\_template("dataownerhome.html",msg="sucess") else:

return render\_template("dataownerlogin.html",msg="Details not found")

return render\_template("dataownerlogin.html")

@app.route("/css",methods=["POST","GET"]) def css():

if request.method=="POST":

name=request.form["name"] pwd=request.form["pwd"]

if name=="CSS" and pwd=="CSS":

return render\_template("csshome.html",msg="success") return render\_template("css.html")

@app.route("/View\_data\_owners") def View\_data\_owners():

sql = "select \* from dataowner where status='pending'" data = pd.read\_sql\_query(sql, db)

print(data)

data.drop(["cpwd","status"],inplace=True,axis=1) data["Action"]="Action" print(data.values.tolist())

return render\_template("View\_data\_owners.html",row\_val=data.values.tolist())

@app.route("/add\_delete/<s1>") def add\_delete(s1=0):

print(s1)

sql = "update dataowner set status='%s' where sno='%s' " % ("accepted", s1) cursor.execute(sql)

db.commit()

return redirect(url\_for('View\_data\_owners'))

@app.route("/delete\_delete/<s1>") def delete\_delete(s1=0):

print(s1)

sql = "update dataowner set status='%s' where sno='%s' " % ("rejected", s1) cursor.execute(sql)

db.commit()

return redirect(url\_for('View\_data\_owners'))

@app.route("/uplodfiles",methods=["POST","GET"]) def uplodfiles():

if request.method == 'POST':

FileName = request.form["FileName"] Keywords = request.form["Keywords"] Files = request.form["Files"]

# session["dcb"] = "No Request Recieved"

dd = "uploadFiles/" + Files f = open(dd, "r")

data = f.read()

sql = "insert into files\_upload (oname,ownername,FileName,Keywords,files\_data) values (%s,%s,%s,%s,AES\_ENCRYPT(%s,'rupesh'))"

values = (session["dataownername"],session["dataownerloginid"], FileName, Keywords, data)

cursor.execute(sql, values) db.commit()

return render\_template("uplodfiles.html", msg="success", files=Files) return render\_template("uplodfiles.html")

@app.route("/viewfiles") def viewfiles():

sql = "select \* from files\_upload where ownername='%s'" % (session["dataownerloginid"]) data = pd.read\_sql\_query(sql, db)

data.drop(["status"],axis=1,inplace=True) print(data)

return render\_template("viewfiles.html",row\_val=data.values.tolist())

@app.route("/publicauditing") def publicauditing():

sql = "select sno,oname,ownername,filename,keywords, files\_data from files\_upload where status='%s'" % ("pending")

data = pd.read\_sql\_query(sql, db)

return render\_template("publicauditing.html",row\_val=data.values.tolist())

@app.route("/updaterequest/<s1>") def updaterequest(s1=0):

sql = "update files\_upload set status='accepted' where sno='%s' " % (s1)

cursor.execute(sql) db.commit()

return redirect(url\_for('publicauditing'))

@app.route("/reject/<s1>")

def reject(s1=0):

sql = "update files\_upload set status='rejected' where sno='%s' " % (s1) cursor.execute(sql)

db.commit()

return redirect(url\_for('publicauditing'))

#searching files @app.route("/SearchFiles",methods=['POST','GET']) def SearchFiles():

if request.method=='POST':

Name=request.form['Keywords'] try:

sql = "select \* from files\_upload where Keywords='%s' " % (Name) # print(X[0][0])#important

results=pd.read\_sql\_query(sql,db) db.commit()

print(results) results.drop(["status"],axis=1,inplace=True) results["action"]=""

print(results)

return render\_template("SearchFilesDisplay.html", col\_name=results.columns.values,row\_val=results.values.tolist())

except:

return render\_template("SearchFiles.html",msg="not found") return render\_template("SearchFiles.html")

@app.route('/fgh/<s1>/<s2>') def fgh(s1=0,s2=0):

print(s1,s2)

sql="insert into filerequest (Oname,fileid,userid) values('%s','%s','%s')"%(session["dataownername"],s1,s2)

cursor.execute(sql)

db.commit()

return redirect(url\_for('SearchFiles'))

@app.route("/filestatus") def filestatus():

sql="select \* from filerequest" data=pd.read\_sql\_query(sql,db) data.drop(["gkey"],axis=1,inplace=True) print(data)

return render\_template("filestatus.html",row\_val=data.values.tolist())

@app.route('/TPA') def TPA():

return render\_template("TPA.html")

@app.route("/tpalogin",methods=["POST","GET"]) def tpalogin():

if request.method=="POST": name=request.form["name"] pwd=request.form["pwd"]

if name=="tpa" and pwd=="tpa":

return render\_template("tpahome.html",msg="success") return render\_template("TPA.html")

@app.route('/TPAFiles') def TPAFiles():

sql="select \* from filerequest where status='pending'"

data=pd.read\_sql\_query(sql,db) data.drop(["gkey"],axis=1,inplace=True) print(data)

return render\_template("TPAFiles.html",row\_val=data.values.tolist())

@app.route("/genearatekey/<s1>") def genearatekey(s1=0):

otp = randint(123593489, 600876509)

print(type(s1),otp)

sql = "update filerequest set status='%s', gkey='%s' where sno='%s' " % ("accepted",otp,s1) cursor.execute(sql)

db.commit()

return render\_template('genearatekey.html')

@app.route("/downloadfile") def downloadfile():

sql = "select \* from filerequest where status='accepted'" data = pd.read\_sql\_query(sql, db)

print(data)

data["Action"]='Action'

return render\_template("downloadfile.html",row\_val=data.values.tolist())

@app.route('/filesf/<s1>') def filesf(s1=0):

sql = "select AES\_DECRYPT(files\_data,'rupesh') from files\_upload where sno='%s' " % (s1) cursor.execute(sql)

data = pd.read\_sql\_query(sql, db)

print(data)

return render\_template("filesf.html", row\_val=[[data.values[0][0].decode('utf8')]]) # return render\_template(".html")

if(\_name)==("main\_"):

app.secret\_key="f9bf78b9a18ce6d46a0cd2b0b86df9da" app.run(debug=True

**SQL Data Base:**

/\*

SQLyog Enterprise - MySQL GUI v6.56

MySQL - 11.2.0-MariaDB : Database - secure\_data\_dynamics

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

/\*!40101 SET NAMES utf8 \*/;

/!40101 SET SQL\_MODE=''/;

/\*!40014 SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0 \*/;

/\*!40101 SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='NO\_AUTO\_VALUE\_ON\_ZERO' \*/;

CREATE DATABASE /!32312 IF NOT EXISTS/`secure\_data\_dynamics` /\*!40100 DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4\_general\_ci \*/;

USE `secure\_data\_dynamics`;

/\*Table structure for table `dataowner` \*/ DROP TABLE IF EXISTS `dataowner`;

CREATE TABLE `dataowner` (

`sno` int(10) unsigned NOT NULL AUTO\_INCREMENT,

`name` varchar(100) DEFAULT NULL,

`email` varchar(100) DEFAULT NULL,

`age` varchar(100) DEFAULT NULL,

`pwd` varchar(100) DEFAULT NULL,

`cpwd` varchar(100) DEFAULT NULL,

`gender` varchar(100) DEFAULT NULL,

`mobile` varchar(100) DEFAULT NULL,

`status` varchar(100) DEFAULT 'pending', PRIMARY KEY (`sno`)

) ENGINE=InnoDB AUTO\_INCREMENT=5 DEFAULT CHARSET=utf8mb4

COLLATE=utf8mb4\_general\_ci;

/\*Data for the table `dataowner` \*/

insert into `dataowner`(`sno`,`name`,`email`,`age`,`pwd`,`cpwd`,`gender`,`mobile`,`status`) values (1,'preeti','preeti@gmail.com','25','1234','1234','Female','6363379953','accepted'),(2,'nakku','nakku@gma

il.com','12','1234','1234','Female','9685745892','rejected'),(3,'malli','malli@gmail.com','26','1234','1234','

Male','7896523658','accepted'),(4,'kumar','kumar@gmail.com','26','1234','1234','Male','9632574563','reje cted');

/\*Table structure for table `filerequest` \*/ DROP TABLE IF EXISTS `filerequest`;

CREATE TABLE `filerequest` (

`sno` int(10) unsigned NOT NULL AUTO\_INCREMENT,

`Oname` varchar(100) DEFAULT NULL,

`fileid` varchar(100) DEFAULT NULL,

`userid` varchar(100) DEFAULT NULL,

`status` varchar(100) DEFAULT 'pending',

`gkey` varchar(100) DEFAULT 'pending', PRIMARY KEY (`sno`)

) ENGINE=InnoDB AUTO\_INCREMENT=2 DEFAULT CHARSET=utf8mb4

COLLATE=utf8mb4\_general\_ci;

/\*Data for the table `filerequest` \*/

insert into `filerequest`(`sno`,`Oname`,`fileid`,`userid`,`status`,`gkey`) values (1,'malli','1','1','accepted','218946897');

/\*Table structure for table `files\_upload` \*/ DROP TABLE IF EXISTS `files\_upload`;

CREATE TABLE `files\_upload` (

`sno` int(10) unsigned NOT NULL AUTO\_INCREMENT,

`oname` varchar(100) DEFAULT NULL,

`ownername` varchar(100) DEFAULT NULL,

`filename` varchar(100) DEFAULT NULL,

`keywords` varchar(100) DEFAULT NULL,

`files\_data` longblob DEFAULT NULL,

`status` varchar(100) DEFAULT 'pending', PRIMARY KEY (`sno`)

) ENGINE=InnoDB AUTO\_INCREMENT=2 DEFAULT CHARSET=utf8mb4

COLLATE=utf8mb4\_general\_ci;

/\*Data for the table `files\_upload` \*/

insert into `files\_upload`(`sno`,`oname`,`ownername`,`filename`,`keywords`,`files\_data`,`status`) values (1,'preeti','1','file','123','\Za.\*v(o','accepted');

/\*!40101 SET SQL\_MODE=@OLD\_SQL\_MODE \*/;

/\*!40014 SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS \*/

**Workspace XML:**

xml version="1.0" encoding="UTF-8"?>

<project version="4">

<component name="ChangeListManager">

<list default="true" id="1dff9e5f-840e-4851-9e6d-8f89a021003e" name="Default Changelist" comment="" />

<option name="SHOW\_DIALOG" value="false" />

<option name="HIGHLIGHT\_CONFLICTS" value="true" />

<option name="HIGHLIGHT\_NON\_ACTIVE\_CHANGELIST" value="false" />

<option name="LAST\_RESOLUTION" value="IGNORE" />

</component>

<component name="ProjectId" id="2R5xKuLwlgJQ2aWJM8zVyffp05g"

<component name="ProjectViewState">

<option name="hideEmptyMiddlePackages" value="true" />

<option name="showExcludedFiles" value="true" />

<option name="showLibraryContents" value="true" />

</component>

<component name="PropertiesComponent">

<property name="RunOnceActivity.ShowReadmeOnStart" value="true" />

<property name="last\_opened\_file\_path" value="W:/2021-2022/PROJECTS/PYTHON 2022 TITLE ID FOLDERS/APPLICATIONS/TCMAPY710-ONLINE SUPER MARKET/CODE/supermarket" />

</component>

<component name="ServiceViewManager">

<option name="viewStates">

<list>

<serviceView>

<treeState>

<expand />

<select />

</treeState>

</serviceView>

</list>

</option>

</component>

<component name="SvnConfiguration">

<configuration />

</component>

<component name="TaskManager">

<task active="true" id="Default" summary="Default task">

<changelist id="1dff9e5f-840e-4851-9e6d-8f89a021003e" name="Default Changelist" comment=""

/>

<created>1686553682255</created>

<option name="number" value="Default" />

<option name="presentableId" value="Default" />

<updated>1686553682255</updated>

</task>

<servers />

</component>

<component name="WindowStateProjectService">

<state x="463" y="120" key="FileChooserDialogImpl" timestamp="1686561705037">

<screen x="0" y="0" width="1366" height="728" />

</state>

<state x="463" y="120" key="[FileChooserDialogImpl/0.0.1366.728@0.0.1366.728"](mailto:FileChooserDialogImpl/0.0.1366.728@0.0.1366.728) timestamp="1686561705037" />

<state x="379" y="189" key="com.intellij.ide.util.TipDialog" timestamp="1686561678728">

<screen x="0" y="0" width="1366" height="728" />

</state>

<state x="379" y="189" key="[com.intellij.ide.util.TipDialog/0.0.1366.728@0.0.1366.728"](mailto:com.intellij.ide.util.TipDialog/0.0.1366.728@0.0.1366.728) timestamp="1686561678728" />

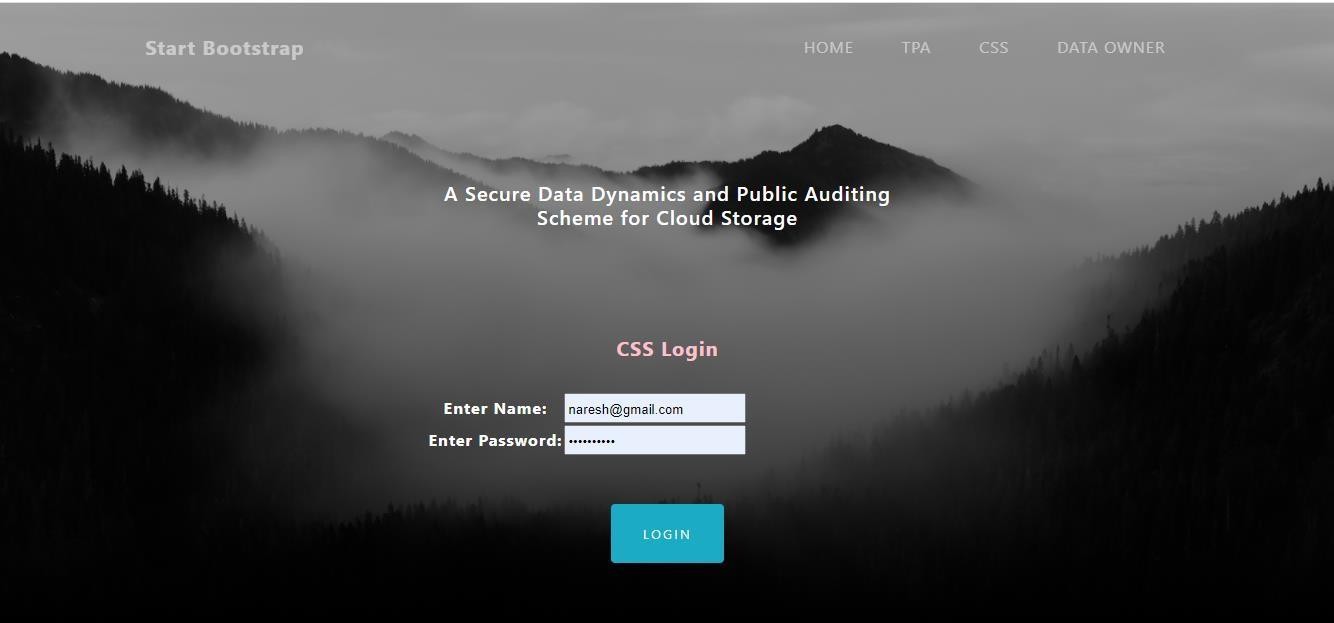
</component>

</project>

## Appendix B Screenshots

**Home page:**

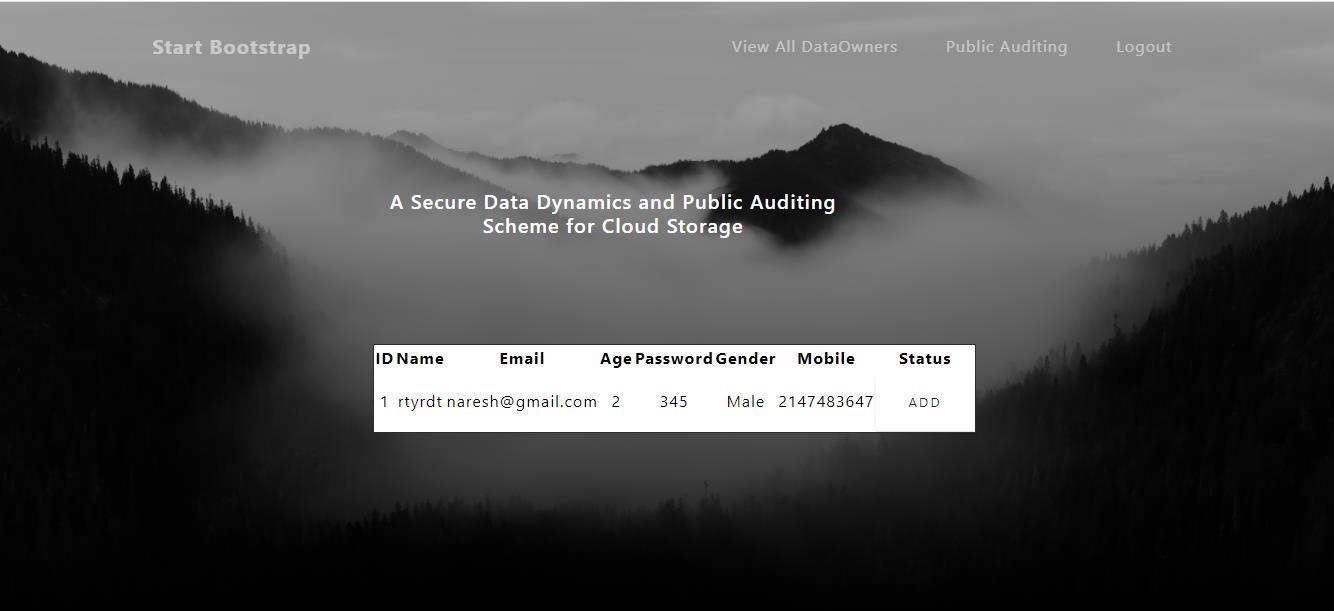
**CSS Login:**



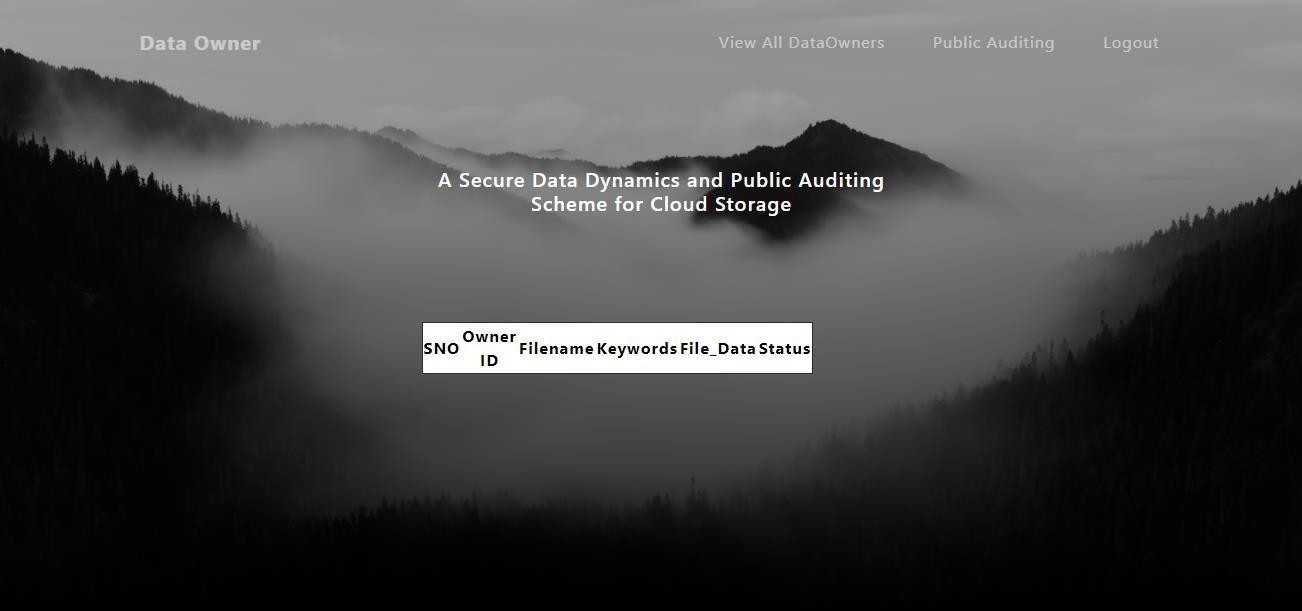
**CSS Home page:**



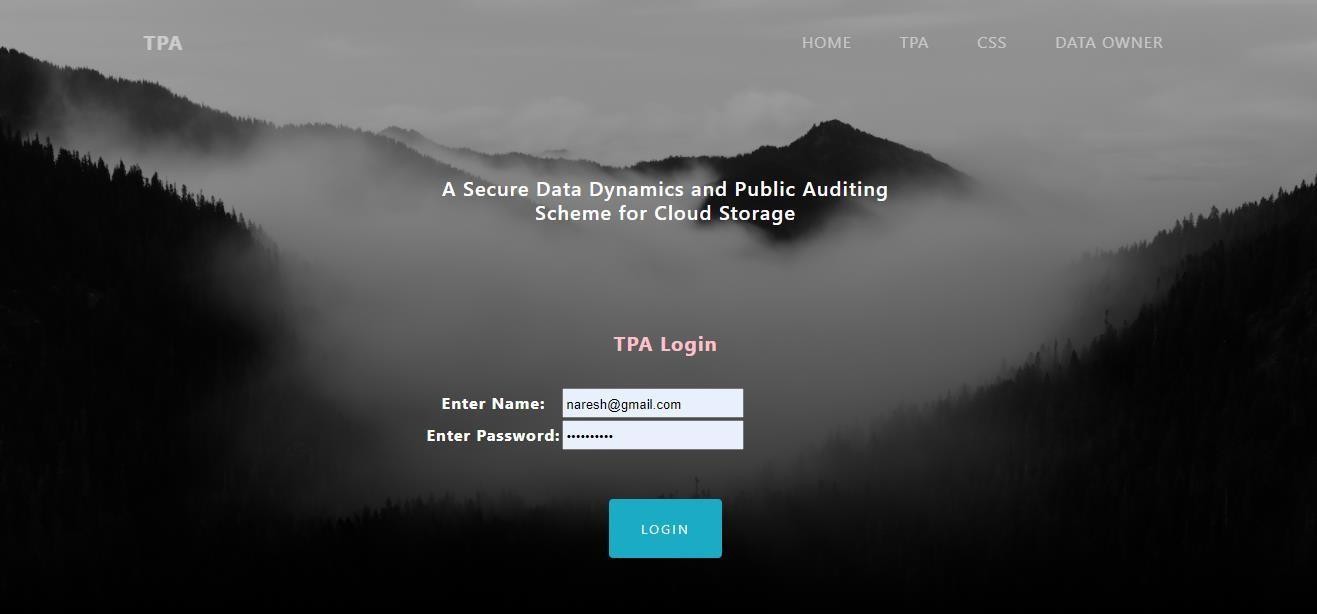
**View all Data Owners :**



**Public Auditing:**



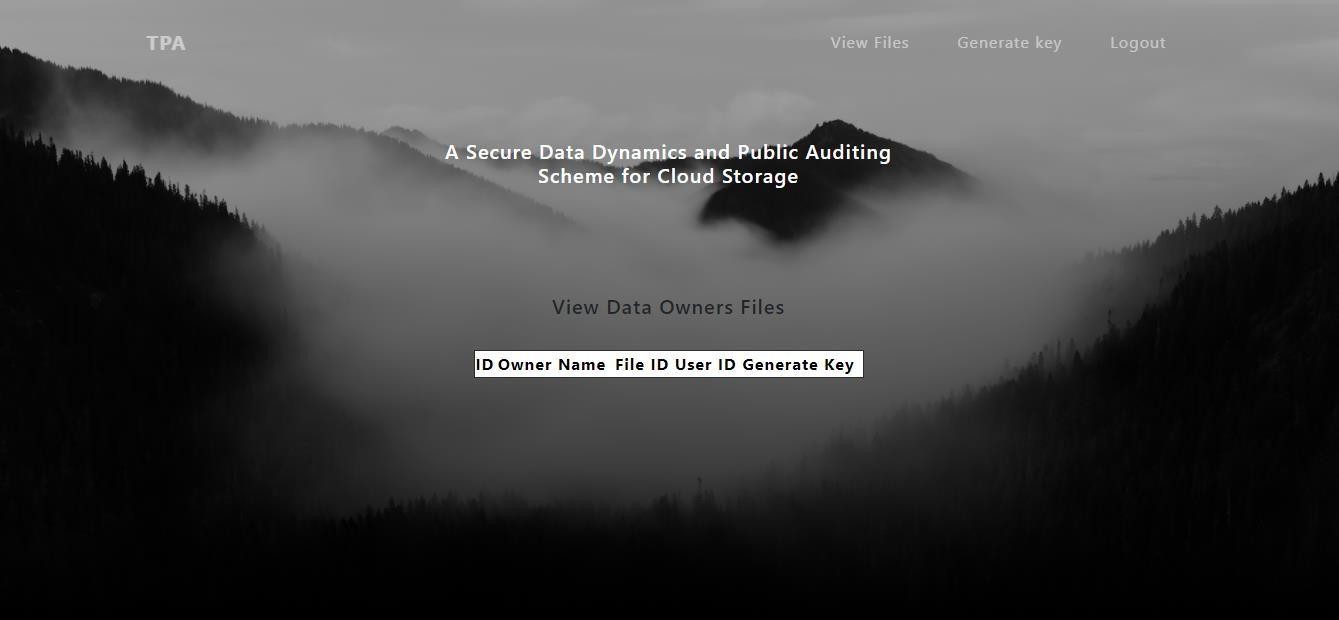
**TPA Login Page:**



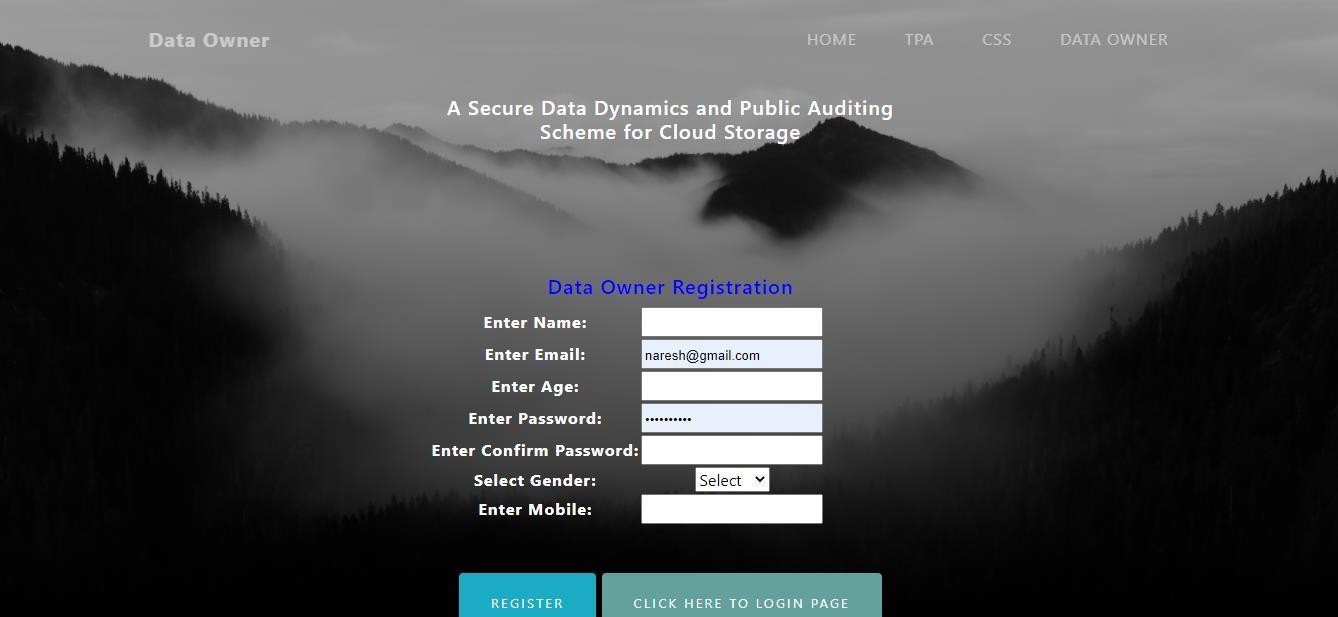
##### TPA Home page



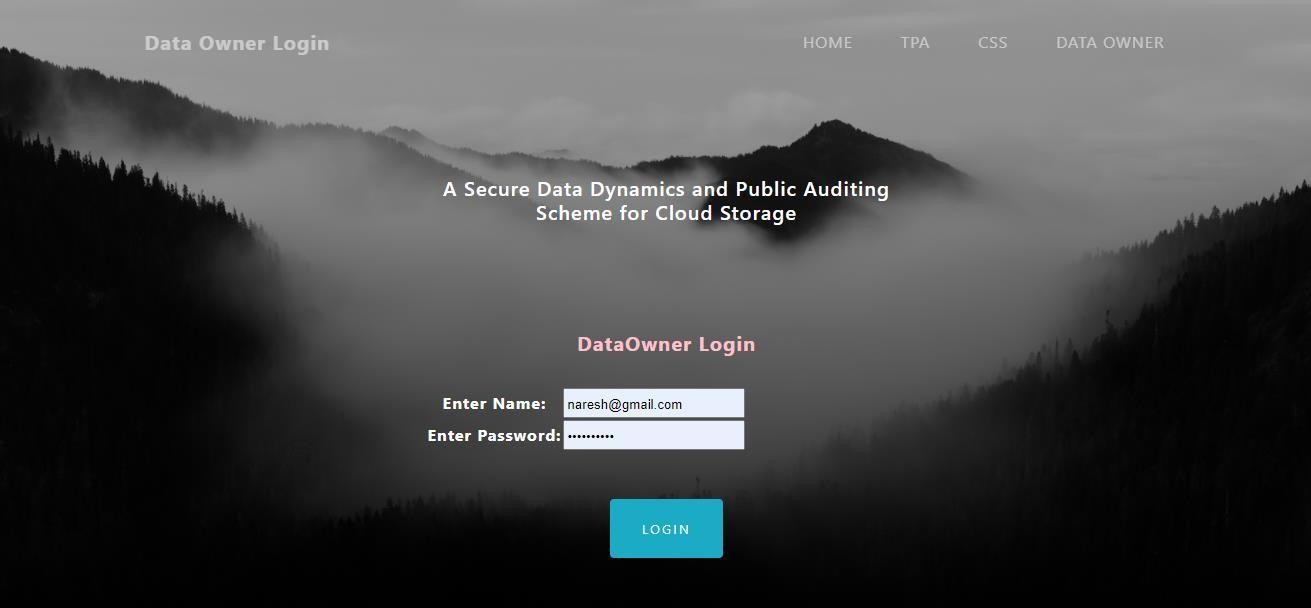
View Files:



**Data Owner Reg Page:**



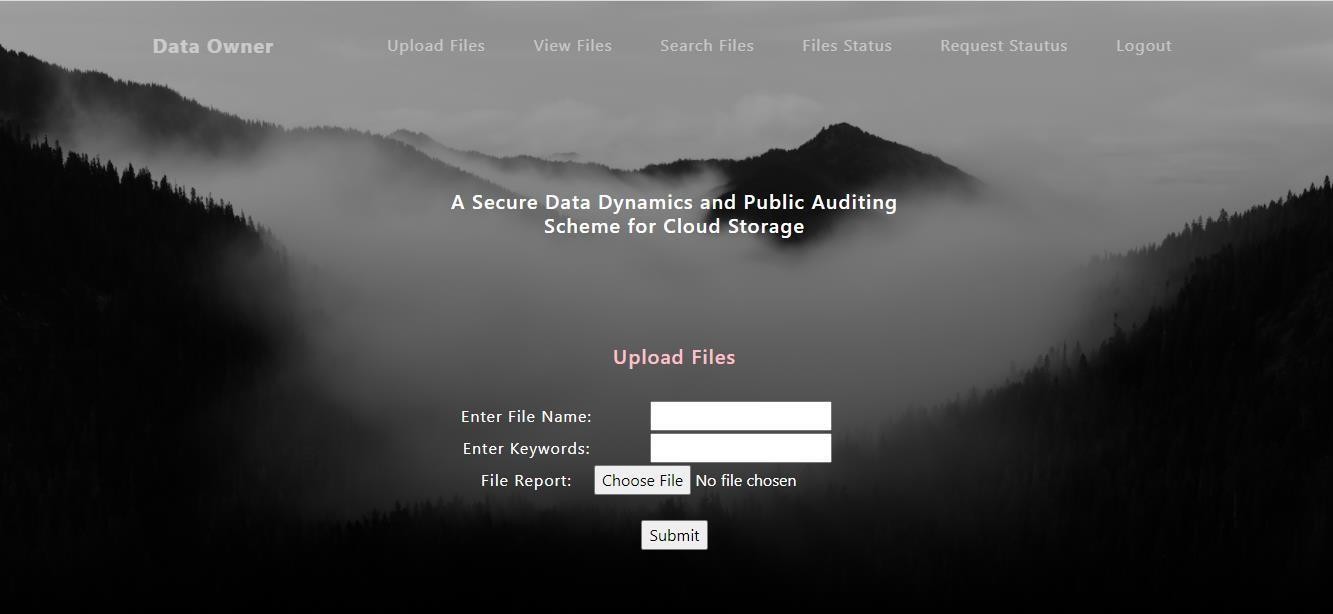
**Data Owner login Page:**



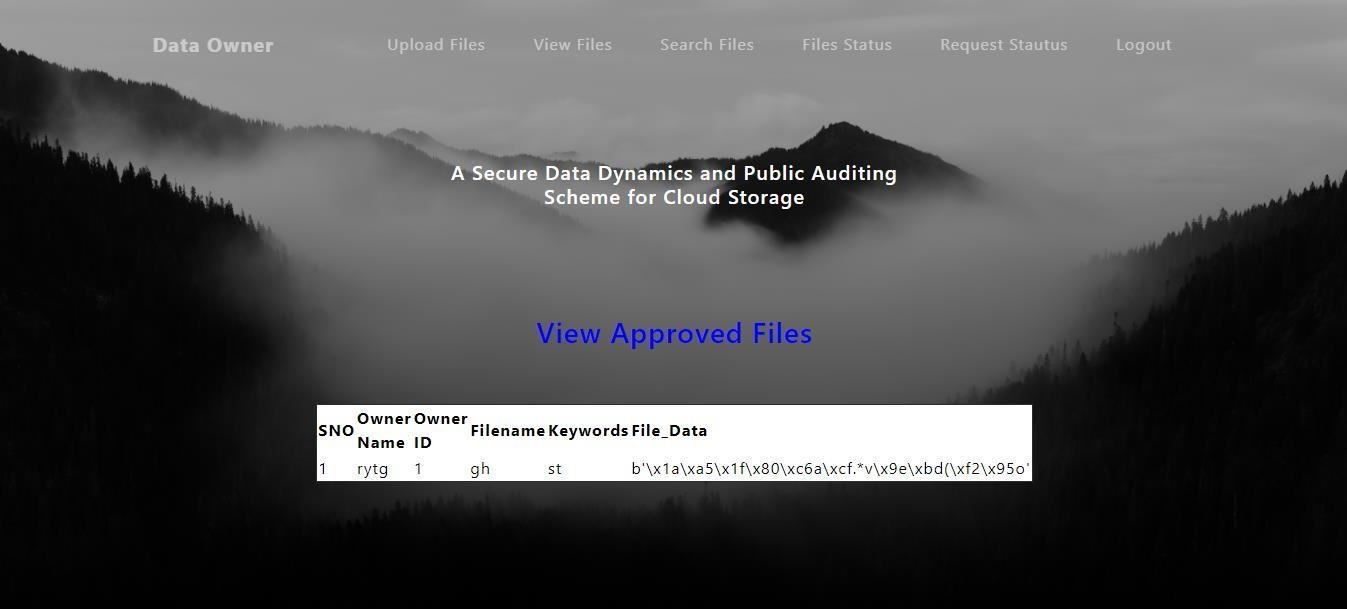
**Owner Home page:**



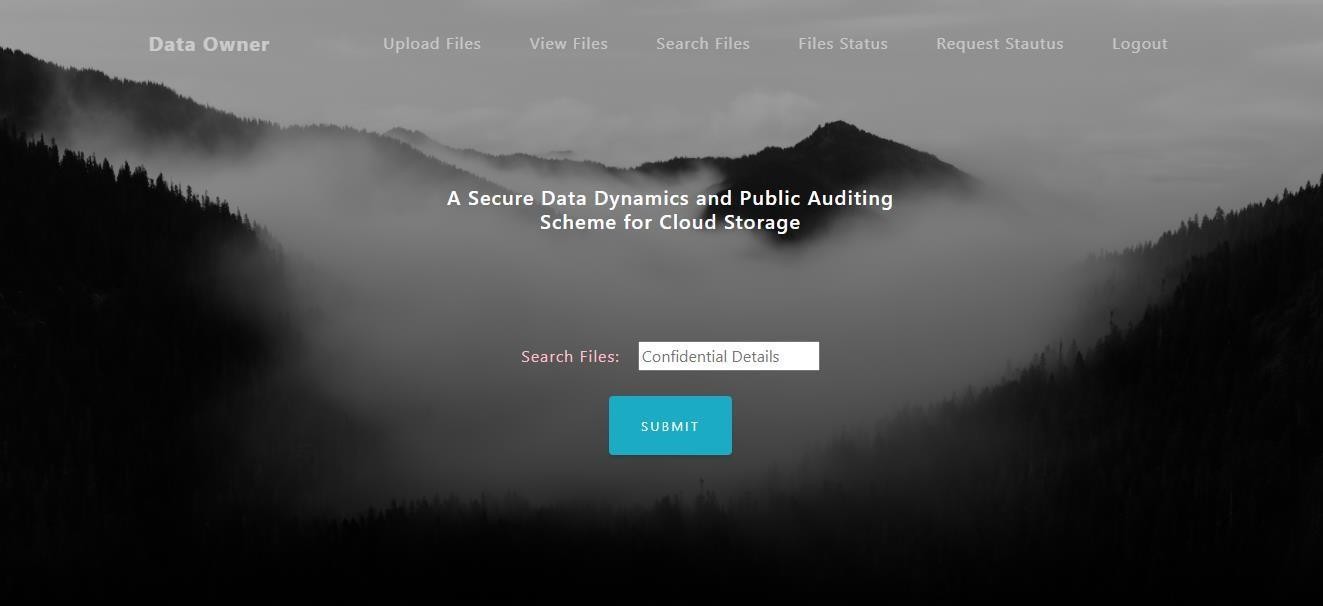
**Upload Data:**



**View Files:**



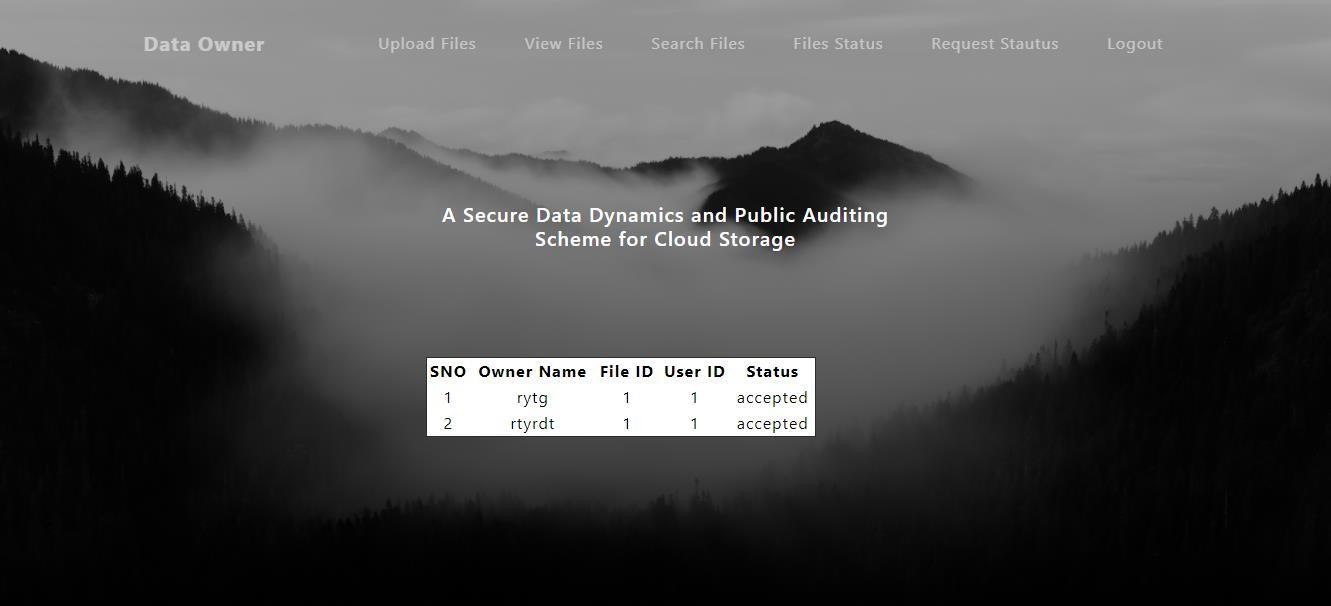
**Search Files:**



**View Data:**



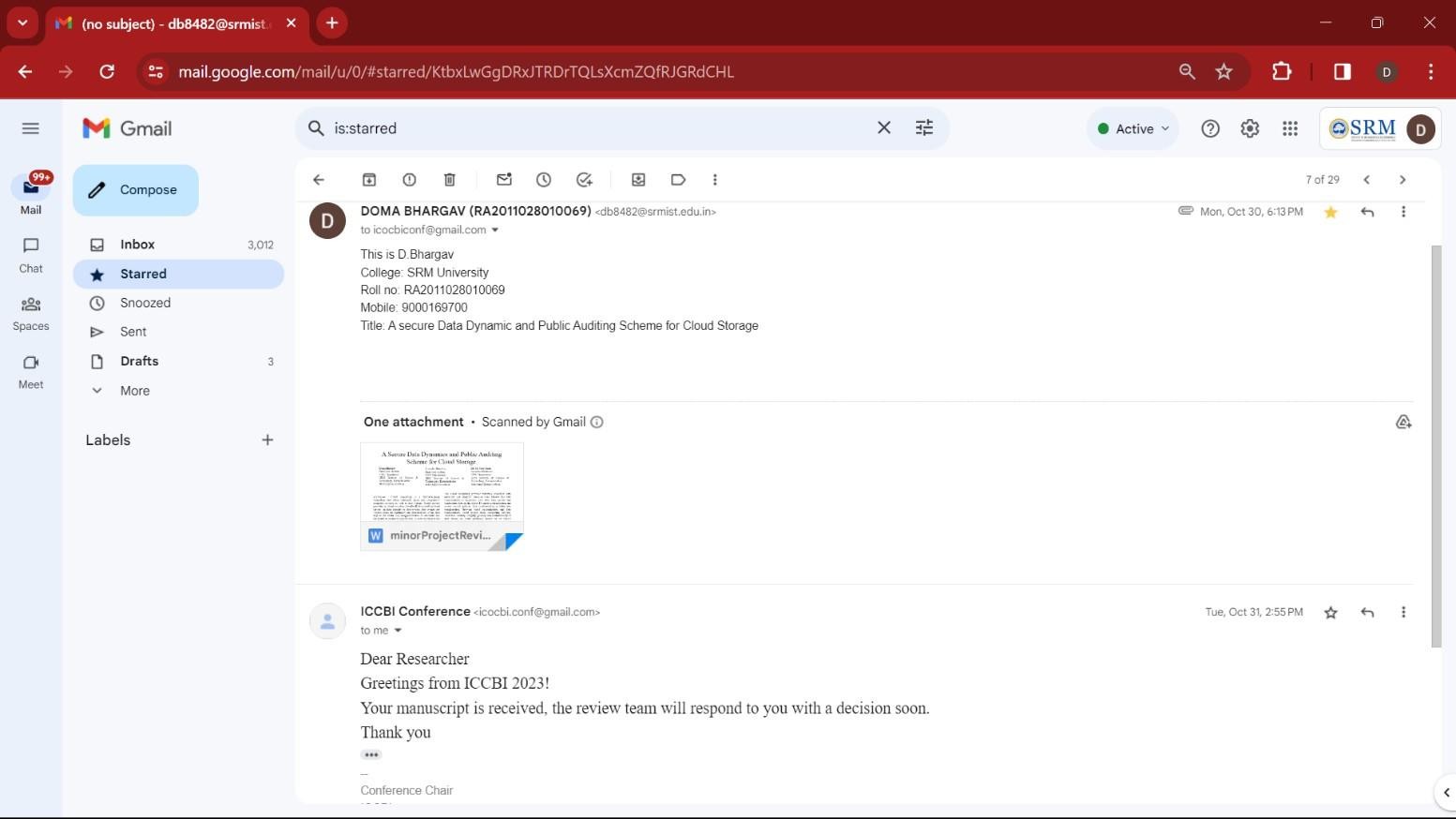
##### File Status:

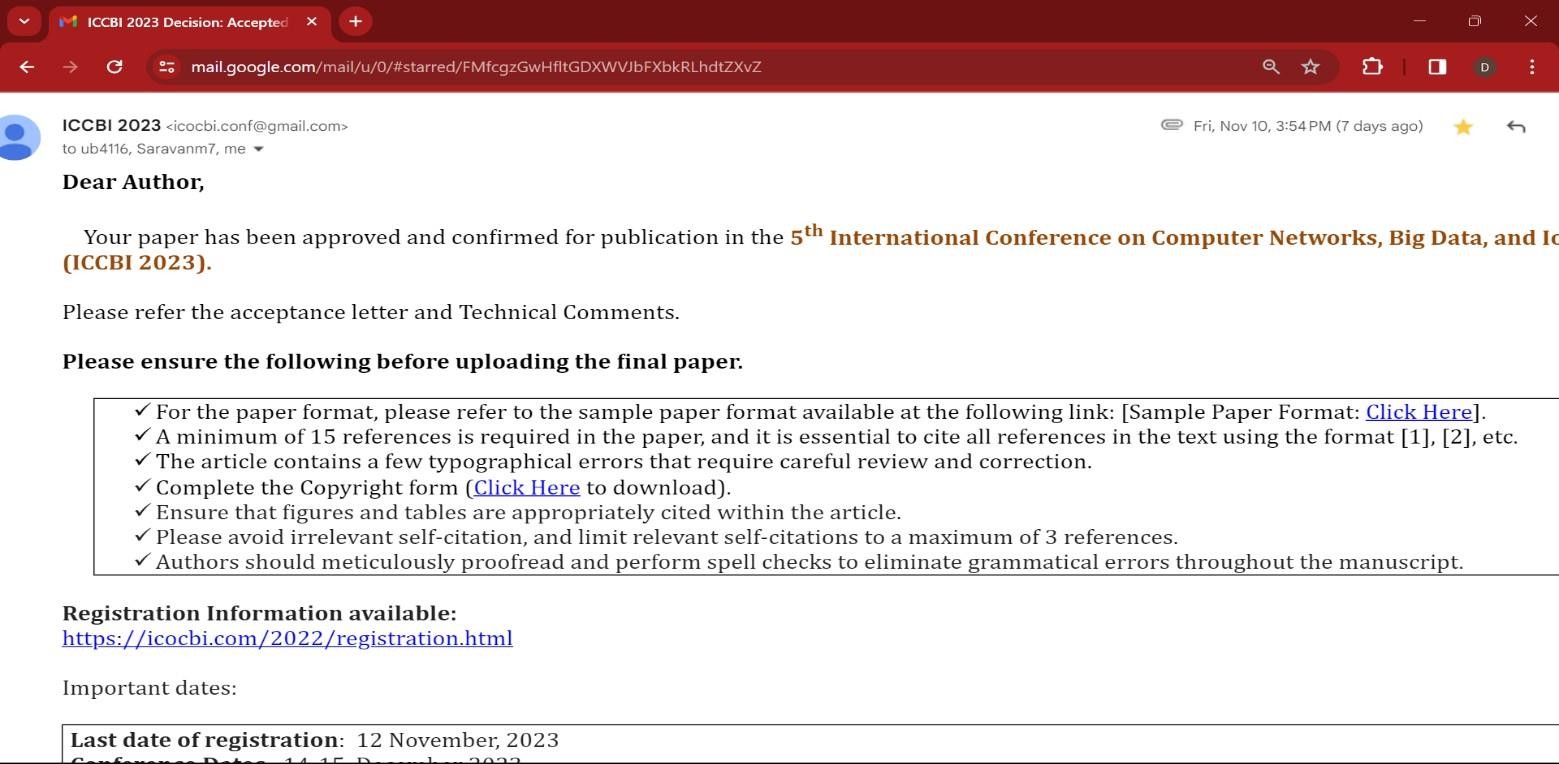


**Request Status :**



## PAPER PUBLICATION STATUS





**PLAGIARISM REPORT**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S R | M I N S | T I T U | T E O F S C I E N C E A N D T E C H N O L O G Y  **(Deemed to be University u/ s 3 of UGC Act, 1956)** | |
| **Office of Controller of Examinations** | | | | |
| REPORT FOR PLAGIARISM CHECK ON THE DISSERTATION/PROJECT REPORTS FOR UG/PG PROGRAMMES  **(To be attached in the dissertation/ project report)** | | | | |
| 1 | Name of the Candidate **(IN BLOCK LETTERS)** | | | DOMA BHARGAV UPANSHU BHARADWAJ |
| 2 | Address of the Candidate | | | SRM Nagar, Kattankulathur - 603 203 Chengalpattu District, Tamil Nadu  SRM Nagar, Kattankulathur - 603 203 Chengalpattu District, Tamil Nadu |
| 3 | Registration | Number | | RA2011028010069 RA2011028010083 |
| 4 | Date of Birth | | | 13-10-2002  15-11-2003 |
| 5 | Department | | | Networking and Communications |
| 6 | Faculty | | | Engineering and Technology |
| 7 | Title of the Dissertation/Project | | | Secure Data Dynamics and Public Auditing Scheme for Cloud Storage |
| 8 | Whether the above project /dissertation is done by | | | ~~Individua~~l or group : (Strike whichever is not applicable)   1. If the project/ dissertation is done in group, then how many students together completed the project 2 2. Mention the Name & Register number of other candidates :   Doma Bhargav RA2011028010069 Upanshu Bharadwj RA2011028010083 |
| 9 | Name and address of the Supervisor / Guide | | | Dr. M Saravanan  SRM Nagar, Kattankulathur - 603 203 Chengalpattu District, Tamil Nadu  Mail ID: [Saravanm@srmist.edu.i](mailto:Saravanm@srmist.edu.in)n Mobile Number: 897305939 |
| 10 | Name and address of Co-Supervisor / Co- Guide (if any) | | | Dr. Vijay Vasanth  SRM Nagar, Kattankulathur - 603 203 Chengalpattu District, Tamil Nadu  Dr. M . Shobana M  SRM Nagar, Kattankulathur - 603 203 Chengalpattu District, Tamil Nadu |

