DATA SECURITY AND PRIVACY PROTECTION FOR CLOUD STORAGE: A SURVEY

1. INTRODUCTION

1.1 PURPOSE

The abstract provides a comprehensive overview of the critical role of cloud storage in the current era, highlighting its significance in managing the exponential growth of data. It outlines the challenges and risks associated with data security and privacy in cloud storage systems and proposes a systematic review of existing literature and potential countermeasures to address these concerns.

1.2 SCOPE

The paper provides a comprehensive review of data security and privacy issues in cloud storage systems. It covers the challenges, requirements, encryption technologies, and protection methods. The scope includes an overview of cloud storage, analysis of security challenges, encryption techniques, and open research topics. The focus is on understanding and mitigating risks associated with unauthorized access, data leakage, and privacy concerns.

1.3 NEED FOR SYSTEM

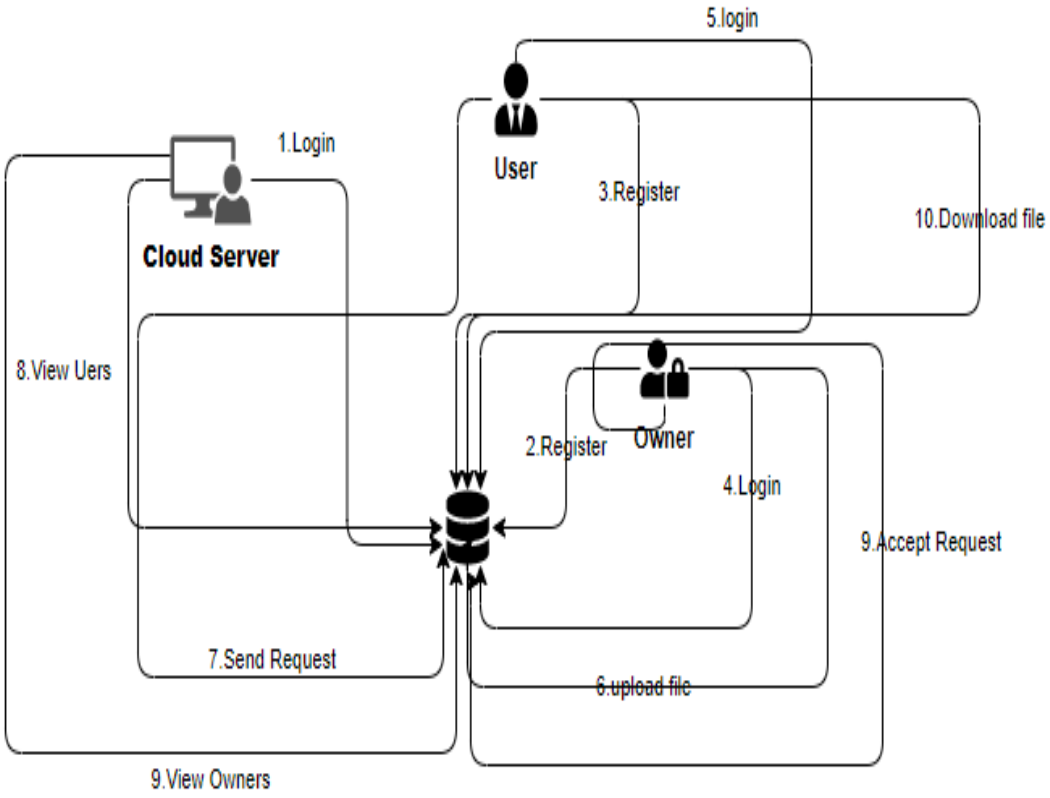
1.3.1 EXISTING SYSTEM

In an existing system there is no security for the stored data, there may be chance to stole the confidential information as well there is no privacy. Although there are some research on data security and privacy protection, there are still no comprehensive studies on the topic for cloud storage systems. As there is no usage of cryptography in early days so, anyone can readily access the information and abuse the data.

1.3.2 PROPOSED SYSTEM

To overcome the problem with an existing system here we are implementing Cryptography method for uploading data and providing data confidentiality for the user's data. Here, the cloud server have to approve the data owner and data user's registration. The data owner will upload the file. That data will be encrypted and stored in the database. Here the user will send a message to other user here we are applying IBE technique for transferring messages. The data will be secured by using IBE technique.

1.4 ARCHITECTURE



2. SOFTARE REQUIREMENT ANALYSIS AND SPECIFICATION

2.1 PRODUCT PERSPECTIVE

In response to the burgeoning demand for secure cloud storage solutions, our product offers robust data encryption and access control features. With a focus on safeguarding sensitive information, we provide a comprehensive suite of tools to mitigate risks such as unauthorized access and data leakage. Our platform prioritizes user privacy and offers scalable solutions to meet the evolving needs of enterprises and individuals in today's digital landscape.

2.2 PRODUCT FUNCTION

Introducing Vault Guard: A comprehensive cloud storage security solution. Safeguard your data with robust encryption technology and advanced access control mechanisms. Protect against unauthorized access, data leakage, and privacy breaches. Seamlessly migrate your data to the cloud while ensuring confidentiality and integrity. Vault Guard offers peace of mind in the face of evolving digital threats.

2.3 USER CHARACTERISTICS

1. Researchers and Academics: Those interested in understanding the current landscape of cloud storage systems, data security, privacy issues, and encryption technologies would find this paper valuable. It provides a comprehensive review of the literature in these areas, which could serve as a foundation for further research.

2. Government Officials and Policy Makers: Individuals responsible for crafting regulations and policies related to data protection and privacy, particularly in the context of cloud storage, could benefit from the insights presented in this paper. It highlights the challenges and requirements for ensuring data security and privacy in cloud storage systems, which can inform policymaking efforts.

3. Enterprise IT Professionals: IT professionals working in organizations that utilize cloud storage solutions would find this paper relevant. It offers insights into the potential risks associated with cloud storage, such as unauthorized access and data leakage, and discusses applicable countermeasures to mitigate these risks. This information can help them make informed decisions about data management and security strategies within their organizations.

4. Individual Users and Consumers: Individuals who use cloud storage services for personal or professional purposes may also find value in this paper. It raises awareness about the importance of data security and privacy protection in cloud storage systems and provides information about encryption technologies and protection methods that can help users safeguard their data.

2.4 MODULES

This project contains 3 modules namely Data owner, Data user, Cloud server.

Operations of modules explained below.

Data Owner:

Register: Data owner should register into the application with required details.

Login: Data owner must login with valid credentials.

Upload file: Data owner will upload the file.

View file: Data owner can view uploaded files.

View User Request: Data owner can view the requests from users to download the file.

Accept Request: After receiving the request from users data owner will accept the request.

Logout: Finally, data owner can logout from the application.

Data User:

Register: Data user should register into the application with required details.

Login: Data user must login with valid credentials.

View files: Data user can search the file with the help of keyword and can view the files related to their search those are uploaded by data owner.

Send request: After selecting, the file data user can send the request to data owner to get an access for downloading that particular file.

View status: Data user can check their status after sending request to data owner.

Download file: Once after accepting the request by data owner, data user can download the file.

Logout: Finally, data user can logout from the application.

Cloud Server:

Login: Cloud server must login with default valid credentials.

New registered users: The cloud servers have to approve the users' registrations.

New registered users: The cloud server have to approve the Owners registrations.

View data owners: Cloud server can view the all data owner's details.

View data user: Cloud server can view the all data user's details.

Logout: Finally, Cloud can logout from the application.

2.5 FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS

2.5.1 FUNCTIONAL REQUIREMENTS

1. Data storage and management capabilities within the cloud storage system.

2. Ability to migrate data from various sources to the cloud securely.

3. Implementation of data security measures such as encryption and access control.

4. Capability to handle massive data volumes generated in the digital era.

5. Provision of comprehensive data security and privacy protection mechanisms.

6. Support for various encryption technologies and protection methods.

7. Continuous monitoring and detection of unauthorized access attempts.

8. Flexibility to adapt to evolving security threats and vulnerabilities.

9. Integration with existing enterprise systems and applications.

10. Ability to scale storage resources dynamically based on demand.

2.5.2 NON-FUNCTIONAL REQUIREMENTS

1. Security: Ensuring confidentiality, integrity, and availability of stored data.

2. Privacy: Protection of sensitive information from unauthorized access and disclosure.

3. Performance: Efficient data storage, retrieval, and processing capabilities.

4. Scalability: Ability to accommodate growing data volumes and user demands.

5. Reliability: Ensuring data is consistently available and accessible.

6. Compliance: Adherence to industry regulations and standards for data security and privacy.

7. Usability: Intuitive user interfaces for data management and access control.

8. Interoperability: Compatibility with different cloud platforms and storage solutions.

9. Resilience: Ability to withstand and recover from system failures or cyber attacks.

10. Cost-effectiveness: Optimization of resource utilization and operational expenses.

2.6 SYSTEM SPECIFICATION

2.6.1 HARDWARE SPECIFICATION

- Processor - I3/Intel Processor
- RAM - 4GB (min)
- Hard Disk - 160GB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - SVGA

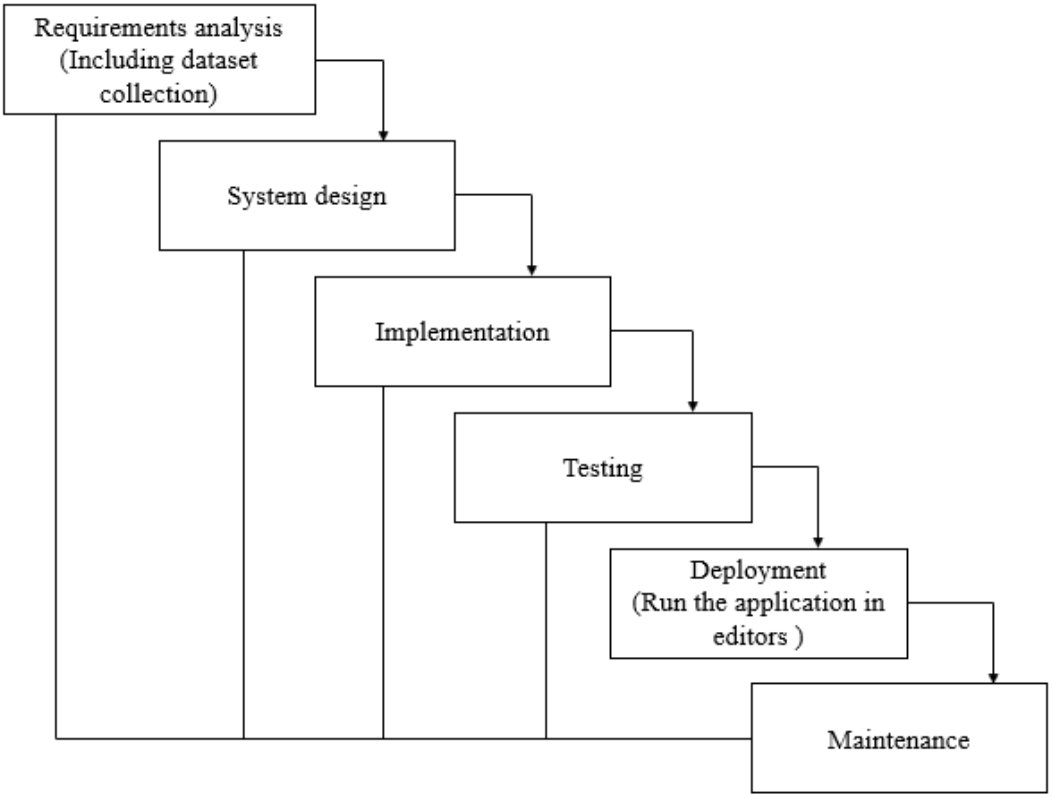
2.6.2 SOFTWARE SPECIFICATION

- Operating System : Windows 7/8/10
- Application Server : Tomcat 7.0
- Front End : HTML, JSP
- Scripts : JavaScript.
- Server side Script : Java Server Pages.
- Database : My SQL 6.0
- Database Connectivity : JDBC

2.7 SOFTWARE DEVELOPMENT LIFE CYCLE

2.7.1 WATERFALL MODEL

In our project, we use waterfall model as our software development cycle because of its systematic procedure while implementing.



Waterfall Model

- **Requirement Gathering and analysis** − all possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** − the requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** − with inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing** − All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

- **Deployment of system** − Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** − There are some issues which come up in the client environment. To fix those issues, patches are released. Also, to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

2.8 SYSTEM STUDY

FEASIBILITY STUDY

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

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

Economic feasibility:

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

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

Social feasibility:

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

2.9 METHODOLOGY AND ALGORITHM

1. User Registration and Authentication:

- Methodology: Cloud server verifies the identity of both the data owner and data user during registration to ensure only authorized users can access the system.

- Algorithms: This process may involve techniques such as password-based authentication, two-factor authentication, or biometric authentication for user verification.

2. Data Upload and Encryption:

- Methodology: The data owner uploads the file to the cloud storage system. Before storing the data in the database, it is encrypted to ensure confidentiality.

- Algorithms: Advanced encryption algorithms like AES (Advanced Encryption Standard) are commonly used for encrypting the data. Additionally, hybrid encryption schemes combining symmetric and asymmetric encryption may be employed for efficiency and security.

3. Message Transfer Using IBE:

- Methodology: When a user wants to send a message to another user, the sender encrypts the message using the recipient's identity, such as their email address, which is converted into a public key by the Key Generation Centre (KGC).

- Algorithms: Identity-Based Encryption (IBE) is utilized for encrypting and decrypting messages. The algorithms involved in IBE include:

- Setup Algorithm: Generates system parameters and master keys.

- Key Extraction Algorithm: Converts user identities into public keys.

- Encryption Algorithm: Encrypts messages using the recipient's public key.

- Decryption Algorithm: Decrypts messages using the recipient's private key, which is generated by the KGC.

4. Data Retrieval and Decryption:

- **Methodology:** When a user requests access to their encrypted data, the cloud server retrieves the encrypted file and decrypts it using the appropriate decryption key.

- **Algorithms:** The decryption algorithm corresponding to the encryption algorithm used during data upload is applied to decrypt the data. Key management techniques may be employed to securely manage and distribute decryption keys.

5. Security Measures:

- **Methodology:** Various security measures are implemented to protect against attacks such as unauthorized access, data leakage, and privacy disclosure.

- **Algorithms:** Access control mechanisms, authentication protocols, and cryptographic primitives are utilized to ensure data security and privacy protection.

2.10 TECHNOLOGIES USED

1. Data Encryption Technology: This includes various encryption algorithms and methods to secure data stored in the cloud. Examples include:

- **AES (Advanced Encryption Standard):** A symmetric encryption algorithm widely used for securing sensitive data.

- **Elliptic Curve Cryptography (ECC):** Another asymmetric encryption technique known for its efficiency and smaller key sizes compared to RSA.

- **Homomorphic Encryption:** Enables computations to be performed on encrypted data without decrypting it first, thus preserving privacy.

2. Access Control Mechanisms: These are used to regulate and manage access to data stored in the cloud, ensuring that only authorized users can access it. Examples include:

- **Role-Based Access Control (RBAC):** Assigns permissions to users based on their roles within an organization.

3. Privacy Protection Measures: These technologies aim to safeguard sensitive information and prevent unauthorized disclosure. Examples include:

- **Data Masking/Anonymization:** Techniques to replace sensitive data with fictitious but realistic values to protect privacy.

- **Data Loss Prevention (DLP):** Software tools and policies designed to prevent the unauthorized transmission of sensitive information.

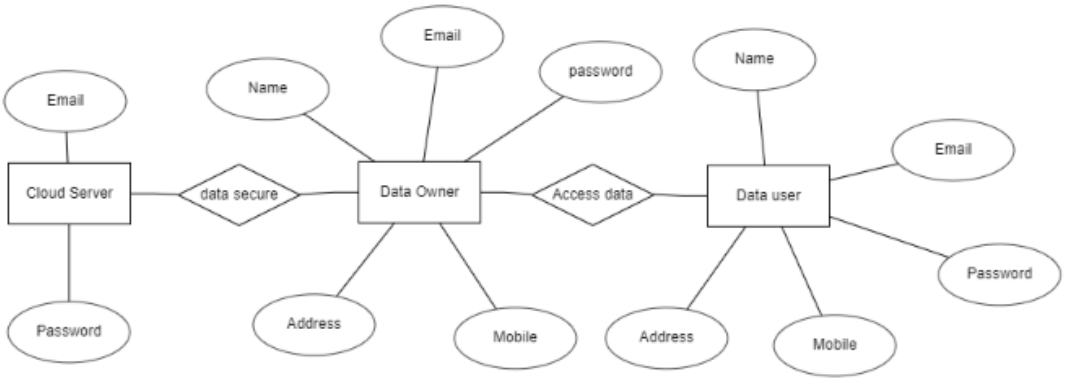
- **Privacy-Preserving Data Mining:** Methods to perform data mining and analysis while protecting the privacy of individuals' data.

3. SYSTEM DESIGN

3.1ER DIAGRAM:

An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.

An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database. Let’s have a look at a simple ER diagram to understand this concept.



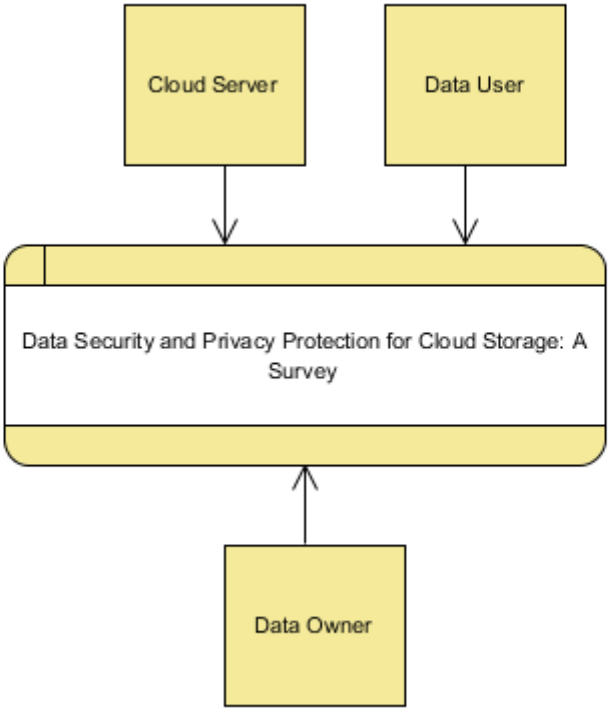
3.2NORMALIZATION

The surge in development trends like Internet of Things, smart cities, digital enterprise transformation, and the global digital economy has propelled the storage market's rapid growth due to immense data generation. Cloud storage systems, offering data storage and management, have become essential in this era, with governments, enterprises, and individuals actively migrating data to the cloud. However, this poses significant risks such as unauthorized access, data leakage, and privacy breaches. This paper conducts a comprehensive review of literature on data security and privacy issues in cloud storage, encompassing encryption technologies and countermeasures.

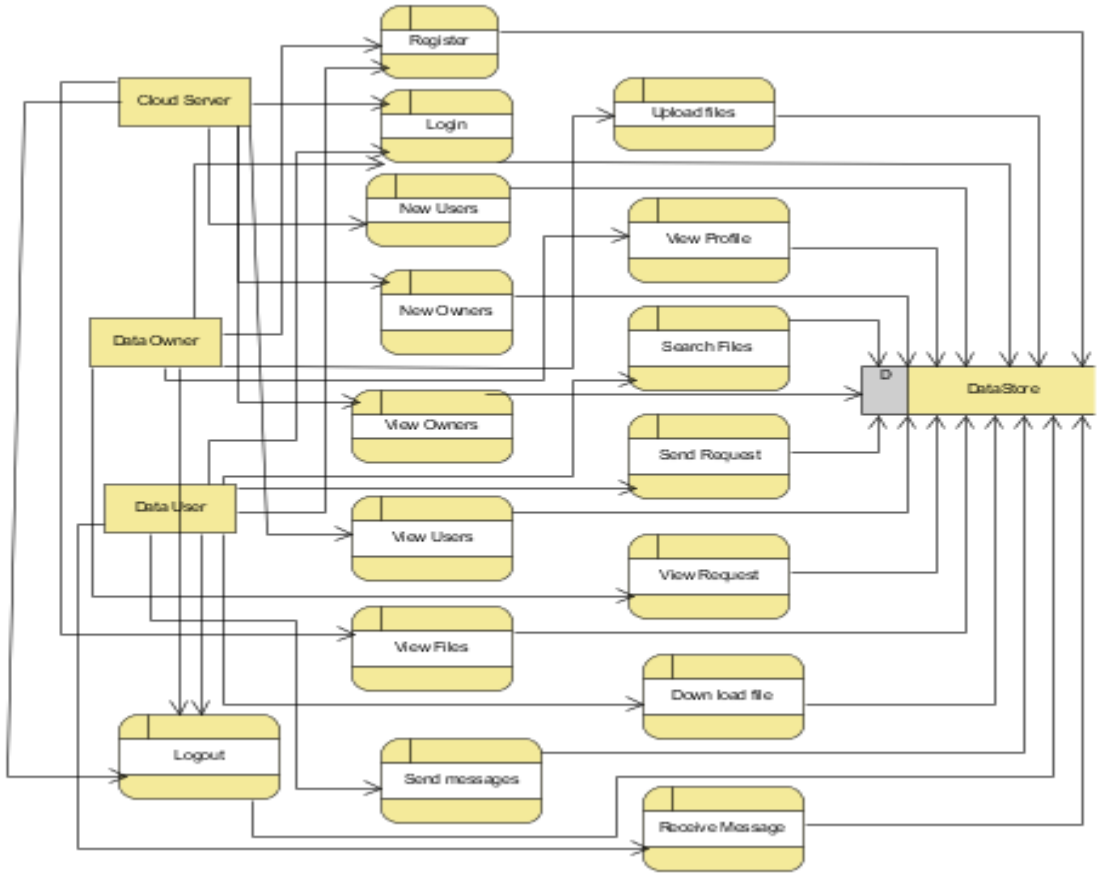
3.3DFD DIAGRAM:

A Data Flow Diagram (DFD) is a traditional way to visualize the information flows within a system. A neat and clear DFD can depict a good amount of the system requirements graphically. It can be manual, automated, or a combination of both. It shows how information enters and leaves the system, what changes the information and where information is stored. The purpose of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communications tool between a systems analyst and any person who plays a part in the system that acts as the starting point for redesigning a system.

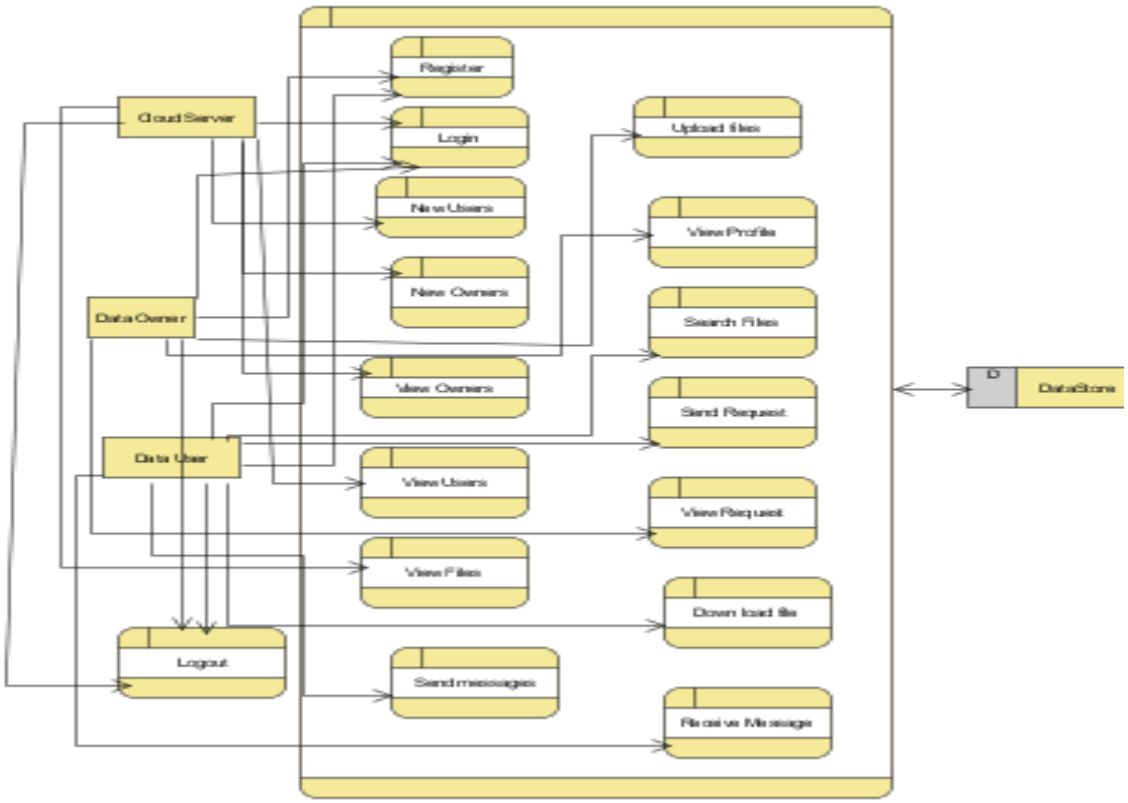
Context level diagram:



Level 1 diagram:



Level 2 diagram:



3.4 UML DIAGRAMS

UML stands for Unified Modelling Language. UML is a standardized general-purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modelling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modelling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

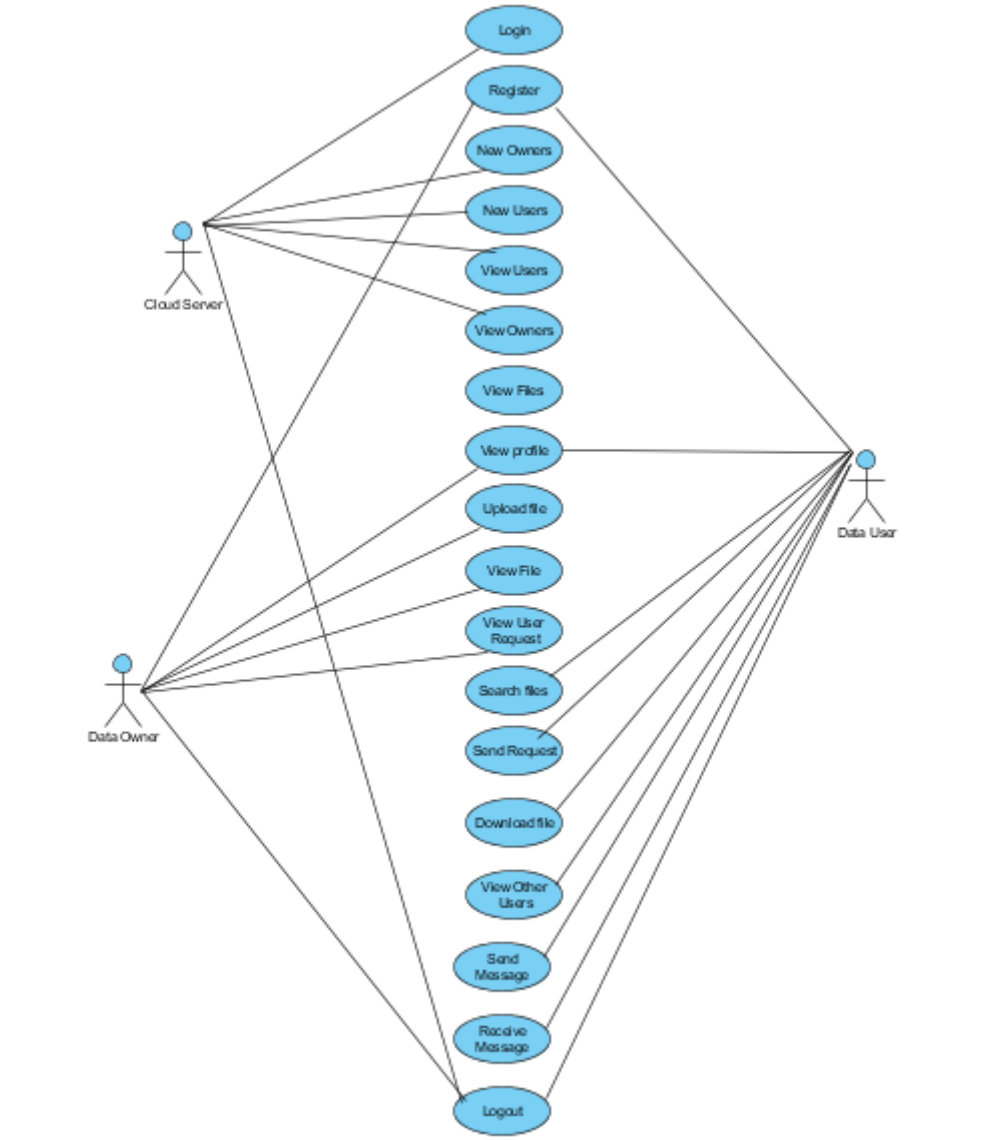
GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modelling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modelling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

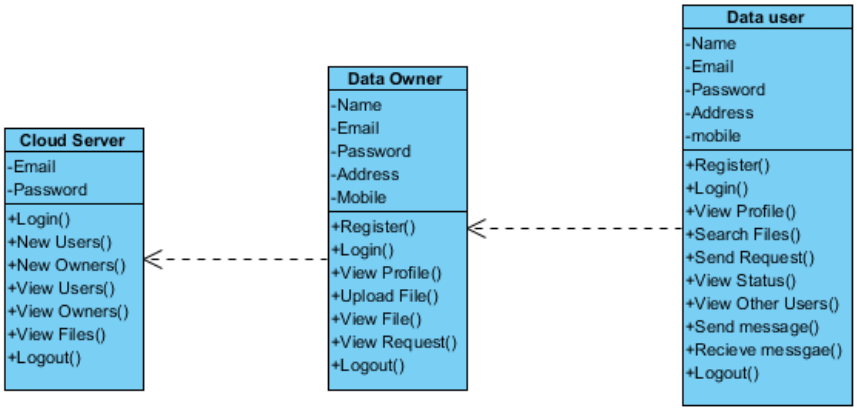
USE CASE DIAGRAM

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



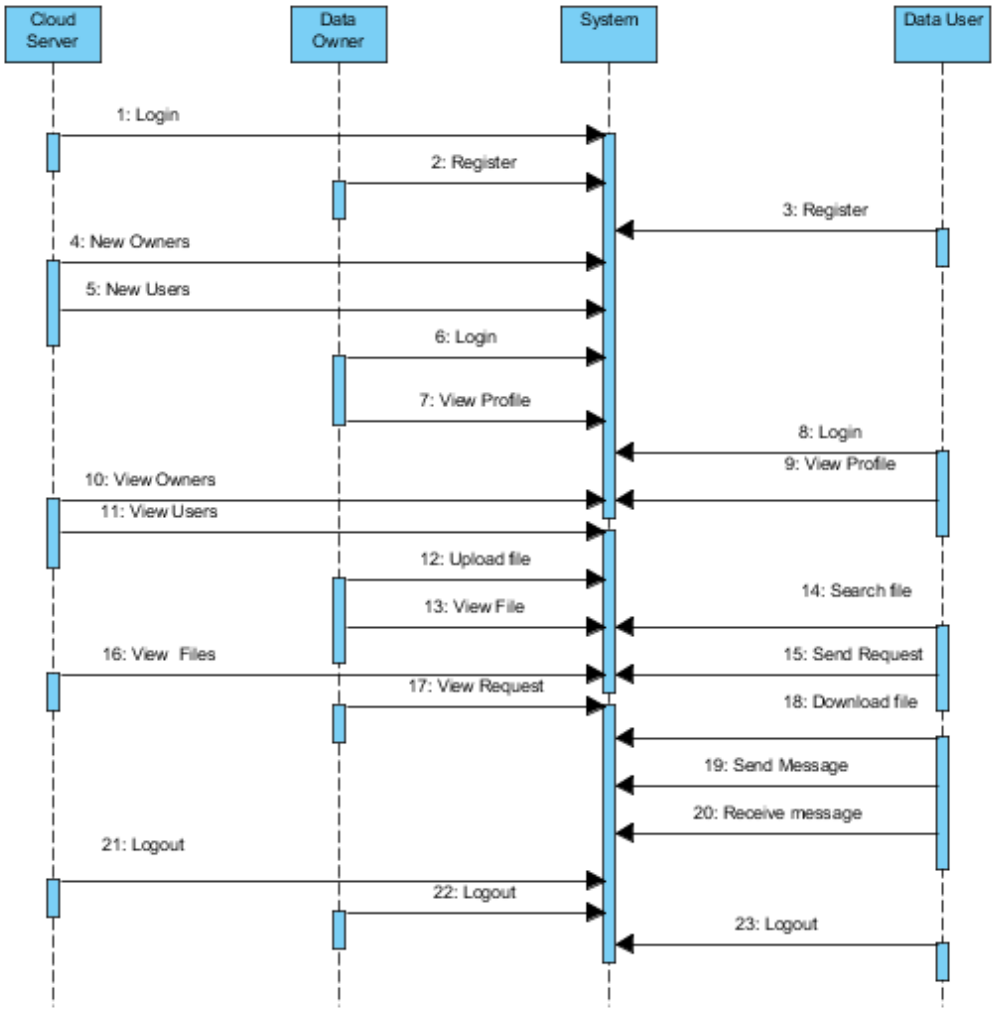
CLASS DIAGRAM

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



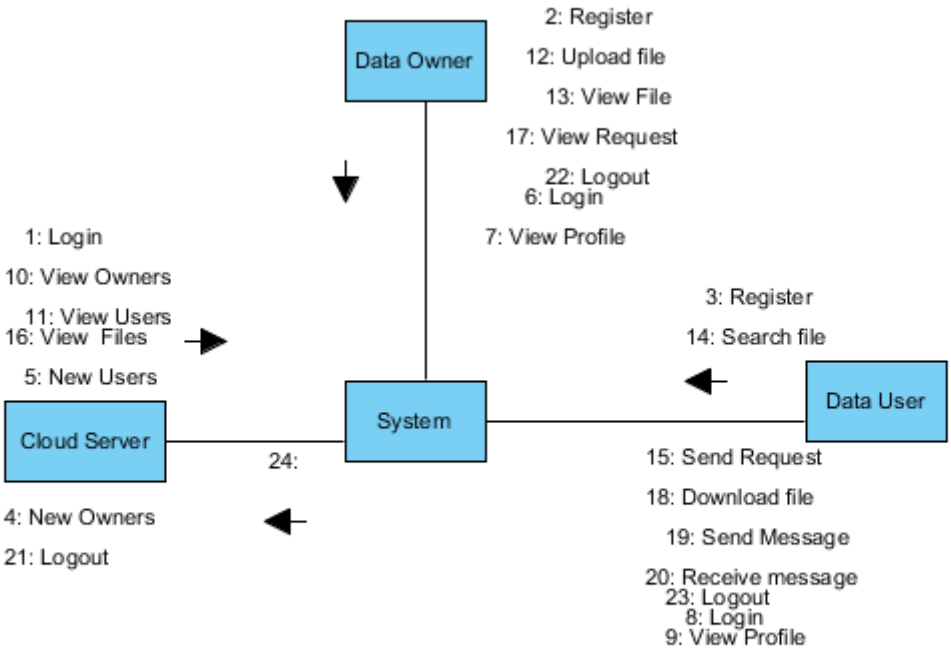
SEQUENCE DIAGRAM

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



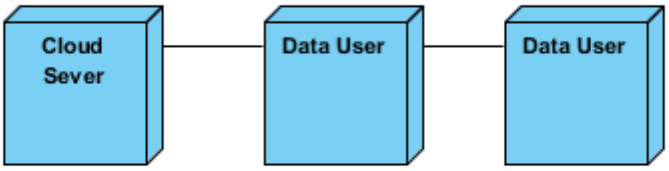
COLLABORATION DIAGRAM:

In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization.



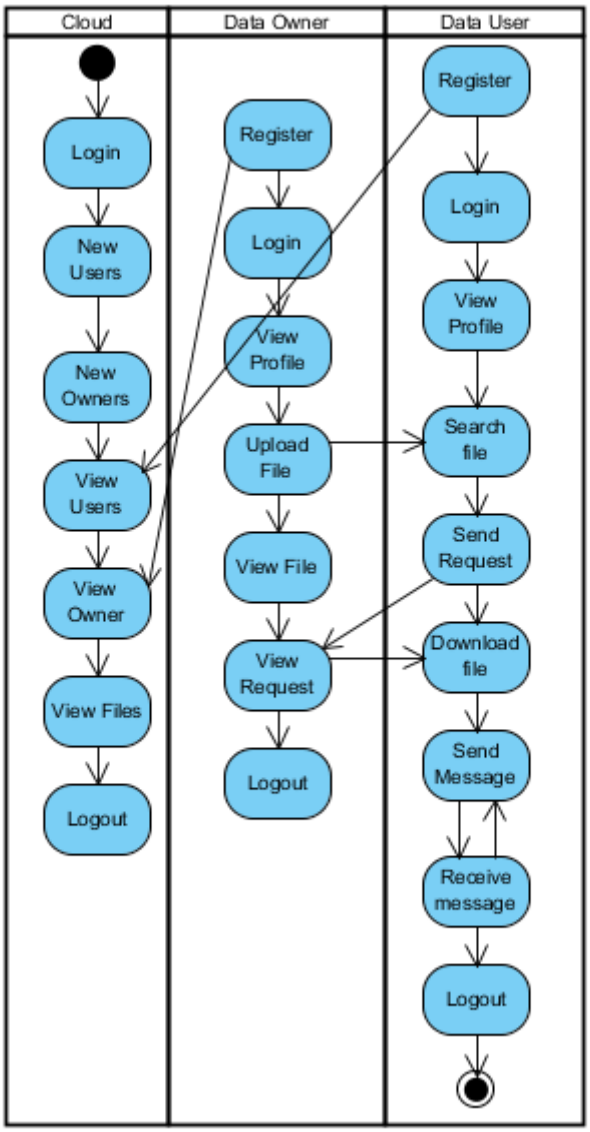
DEPLOYMENT DIAGRAM

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware's used to deploy the application.



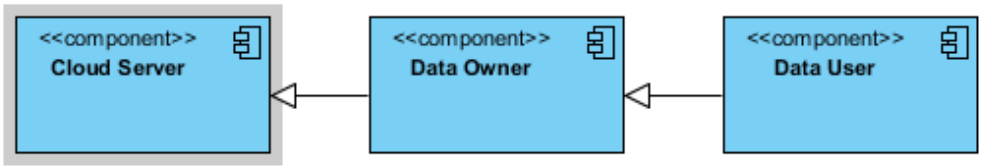
ACTIVITY DIAGRAM:

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



COMPONENT DIAGRAM:

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical **c**omponents in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required function is covered by planned development.



4. TESTING

4.1 Introduction:

Testing is a critical process in software development that ensures the quality, functionality, and reliability of software applications. It involves executing a program or system with the intent of finding errors and verifying that it meets specified requirements. Testing encompasses various techniques and methodologies tailored to uncover defects, assess performance, and validate functionality across different stages of the software development lifecycle.

In today's dynamic and competitive digital landscape, the importance of testing has escalated, driven by increasing complexity in software systems, rapid release cycles, and evolving user expectations. Effective testing practices not only enhance the overall quality of software but also mitigate risks associated with defects, security vulnerabilities, and compliance issues.

This introduction provides an overview of the fundamental concepts and significance of testing in software development, laying the groundwork for exploring its diverse aspects, methodologies, and emerging trends.

4.2 Types of Testing

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform

basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

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

Functional testing is centered on the following items:

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

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

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test.

System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. You cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

4.3 TESTING METHODOLOGIES

Testing methodologies are systematic approaches to planning, executing, and evaluating tests throughout the software development lifecycle. These methodologies help ensure that software meets quality standards, functions as expected, and fulfils user requirements. Here are some common testing methodologies:

1. Waterfall Model: In this traditional sequential approach, testing occurs after the development phase is complete. Each stage, such as requirements, design, implementation, testing, and maintenance, is completed sequentially before moving to the next.

2. V-Model: The V-Model is an extension of the waterfall model where testing activities are aligned with each stage of the development process. For every phase of development, there is a corresponding testing phase, forming a V-shape structure.

3. Agile Testing: Agile methodologies, such as Scrum or Kanban, emphasize iterative development and collaboration between cross-functional teams. Testing occurs continuously throughout the development process, with frequent releases and feedback loops.

4. DevOps Testing: DevOps integrates development and operations teams to enable faster and more reliable software delivery. Testing is automated and integrated into the continuous integration/continuous deployment (CI/CD) pipeline, allowing for rapid feedback and deployment.

5. Exploratory Testing: This methodology relies on testers' intuition, experience, and creativity to uncover defects. Testers explore the software without predefined test cases, focusing on understanding the system and identifying potential issues.

6. Risk-Based Testing: Risk-based testing prioritizes test activities based on the likelihood and impact of potential failures. Tests are designed to mitigate the highest risks first, ensuring that critical functionality is thoroughly tested.

7. Behavior-Driven Development (BDD): BDD focuses on defining desired system behavior using natural language specifications called "user stories" or "scenarios." Tests are written in a human-readable format and serve as both requirements documentation and executable tests.

4.4 Test Cases

S.NO	Test cases	I/O	Expected O/T	Actual O/T	P/F
1	Data Owner register	Owner details	If email not exist registration done else email already exists	If email not exist registration done else email already exists	P
2	Data Owner login	Valid/invalid credentials	Login success/ login failed	Login success/ login failed	P
3	Data User register	User details	If email not exist registration done else email already exists	If email not exist registration done else email already exists	P
4	Data User login	Valid/invalid credentials	Login success/ login failed	Login success/ login failed	P

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

5. IMPLEMENTATION

5.1 SAMPLE OUTPUTS

HOMEPAGE:

CLOUD LOGIN:

CLOUD HOME:

New Registered Owners:

New Registered Users:

View data owners:

View data users:

View Files:

Data Owner Login:

Homepage:

View profile:

Upload File:

View File:

View requests:

User Login:

User Homepage:

User View Profile:

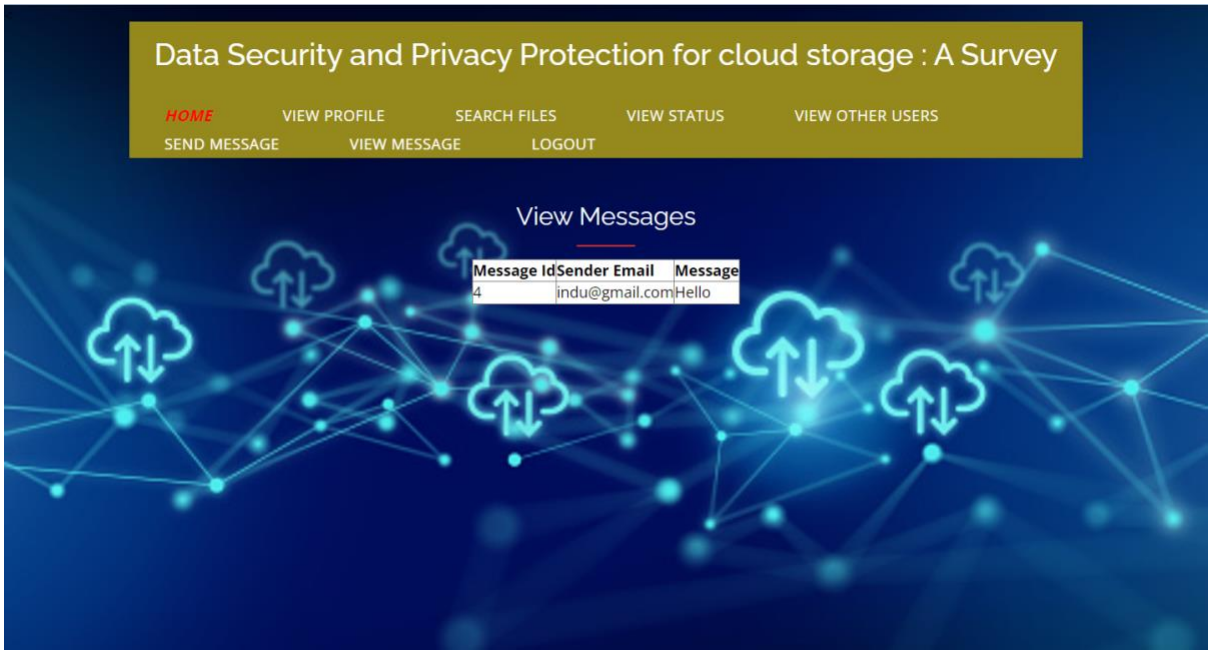
Search Files:

View Status

View Other Users

Send Messages:

View Message:



CONCLUSION

In this paper, we give a detail survey on data security and privacy preservation in cloud storage system. First of all, from the outstanding performance of cloud in the digital economy, enterprise digital transformation, Internet of things and other fields, we confirm that cloud computing and cloud storage will still be the mainstream. We first analyze eight elements of data security in cloud storage system: data confidentiality, data integrity, data availability, fine-grained access control, secure data sharing.

References

[1] A. Akavia, S. Goldwasser, and V. Vaikuntanathan, ‘‘Simultaneous hardcore bits and cryptography against memory attacks,’’ in Proc. TCC, San Francisco, CA, USA, 2009, pp. 474–495.

[2] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, ‘‘Provable data possession at untrusted stores,’’ in Proc. ACM-CSS, New York, NY, USA, 2007, pp. 598–609.

[3] N. Attrapadung and H. Imai, ‘‘Attribute-based encryption supporting direct/indirect revocation modes,’’ in Proc. IMACC, Cirencester, U.K., Dec. 2009, pp. 278–330.

[4] J. Baek, R. Safavi-Naini, and W. Susilo, ‘‘Public key encryption with keyword search revisited,’’ in Proc. ICCSA, Perugia, Italy, 2008, pp. 1249–1259.

[5] M. Bellare, A. Boldyreva, and A. O’Neill, ‘‘Deterministic and efficiently searchable encryption,’’ in Proc. CRYPTO, Santa Barbara, CA, USA, 2007, pp. 535–552.

[6] F. Berti, O. Pereira, T. Peters, and F. X. Standaert, ‘‘On leakage-resilient authenticated encryption with decryption leakages,’’ IACR Trans. Symmetric Cryptol., vol. 2017, no. 3, pp. 271–293, 2017.

[7] J. Bethencourt, A. Sahai, and B. Waters, ‘‘Ciphertext-policy attributebased encryption,’’ in Proc. IEEE Symp. Secur. Privacy, Berkeley, CA, USA, May 2007, pp. 321–334.

[8] T. Bhatia and A. K. Verma, ‘‘Data security in mobile cloud computing paradigm: A survey, taxonomy and open research issues,’’ J. Supercomput., vol. 73, no. 6, pp. 2558–2631, Jun. 2017.

[9] A. Boldyreva, V. Goyal, and V. Kumar, ‘‘Identity-based encryption with efficient revocation,’’ in Proc. ACM CCS, Alexandria, VA, USA, 2008, pp. 417–426.

[10] D. Boneh and X. Boyen, ‘‘Efficient selective-ID secure identity based encryption without random oracles,’’ in Proc. Adv. Cryptol. (Eurocrypt), Interlaken, Switzerland, vol. 3027, 2004, pp. 223–238. [11] D. Boneh and X. Boyen, ‘‘Secure identity-based encryption without random oracles,’’ in Proc. CRYPTO, vol. 3152. Berlin, Germany: Springer, 2004, pp. 443–459.

[12] D. Boneh and M. Franklin, ‘‘Identity-based encryption from the Weil pairing,’’ in Proc. CRYPTO, vol. 2139. Berlin, Germany: Springer, 2001, pp. 213–229.

[13] D. Boneh, G. Di Crescenz, R. Ostrovsky, and G. Persiano, ‘‘Public key encryption with keyword search,’’ in Proc. EUROCRYPT, vol. 3027. Berlin, Germany: Springer, 2004, pp. 506–522.

[14] Z. Brakerski, ‘‘Fully homomorphic encryption without modulus switching from classical GapSVP,’’ in Proc. CRYPTO, Santa Barbara, CA, USA, 2012, pp. 868–886.

[15] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, ‘‘(Leveled) fully homomorphic encryption without bootstrapping,’’ ACM Trans. Comput. Theory, vol. 6, no. 3, pp. 1–36, Jul. 2014.

[16] Z. Brakerski and V. Vaikuntanathan, ‘‘Efficient fully homomorphic encryption from (standard) LWE,’’ SIAM Journal on Computing, vol. 43, no. 2, pp. 831–871, Jan. 2014.

[17] Z. Brakerski, Y. T. Kalai, J. Katz, and V. Vaikuntanathan, ‘‘Overcoming the hole in the bucket: Public-key cryptography resilient to continual memory leakage,’’ in Proc. IEEE 51st Annu. Symp. Found. Comput. Sci. (FOCS), Las Vegas, NV, USA, Oct. 2010, pp. 501–510.

[18] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, ‘‘Privacy-preserving multikeyword ranked search over encrypted cloud data,’’ IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 1, pp. 222–233, Jan. 2014. [19] B. Casemore, ‘‘Network modernization: Essential for digital transformation and multicloud,’’ IDC, Framingham, MA, USA, White Paper US45603019, Nov. 2019.

[20] J. H. Cheon, A. Kim, M. Kim, and Y. Song, ‘‘Homomorphic encryption for arithmetic of approximate numbers,’’ in Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur., Hong Kong, 2017, pp. 409–437.