# TRANSPARENT ENCLAVE EXECUTION

Submitted in partial fulfillment of the requirements for the award of

Bachelor of Engineering degree in Computer Science and Engineering

By

**GUNDA YUVA NAGA SAI (Reg. No - 39110356)**
**VINEETH SARADA (Reg. No – 39111106)**


**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**SCHOOL OF COMPUTING**

# SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY**
**(DEEMED TO BE UNIVERSITY)**
**Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE**
**JEPPIAAR NAGAR, RAJIV GANDHISALAI,**
**CHENNAI - 600119**

**APRIL - 2023**

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**BONAFIDE CERTIFICATE**

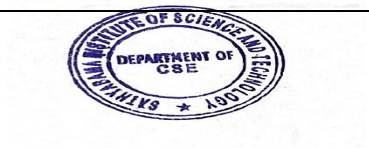
This is to certify that this Project Report is the bonafide work of **Gunda Yuva Naga Sai (Reg.No - 39110356) and Vineeth Sarada(Reg.No - 39111106)** who carried out the Project Phase-2 entitled **"TRANSPARENT ENCLAVE EXECUTION"** under my supervision from January 2023 to April 2023.

**Internal Guide**

**Dr. M. MAHESWARI, M.E., Ph.D**

**Head of the Department**

**Dr. L. LAKSHMANAN, M.E., Ph.D.**




**Submitted for Viva voce Examination held on 24.04.2023**

**Internal Examiner** **External Examiner**

## DECLARATION

I, **Gunda Yuva NagaSai (Reg.No - 39110356),** hereby declare that the Project Phase-2 Report entitled **"TRANSPARENT ENCLAVE EXECUTION"** done by me under the guidance of **Dr. M. Maheswari, M.E., Ph.D** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in **Computer Science and Engineering**.

**DATE:24.04.2023**

**PLACE: Chennai**

**SIGNATURE OF THE CANDIDATE**

## ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph. D**, **Dean**, School of Computing, **Dr. L. Lakshmanan M.E., Ph.D,** Head of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.M.Maheswari M.E.,Ph.D,** for her valuable guidance, suggestions and constant encouragement paved the way for the successful completion of my phase-2 project work.

I wish to express my thanks to all the Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

# ABSTRACT

Today's internet users place a premium on keeping their personal information safe. Cloud-based data storage service has drawn increasing interests from both academic and industry in the recent years due to its efficient and low-cost management. Since it provides services in an open network, it is urgent for service providers to make use of secure data storage and sharing mechanism to ensure data confidentiality and service user privacy. To protect sensitive data from being compromised, the most widely used method is encryption. However, simply encrypting data cannot fully address the practical need of data management. Besides, an effective access control over download request also needs to be considered so that economic denial of sustainability (EDoS) attacks cannot be launched to hinder users from enjoying service. In this paper we consider the transparent enclave execution, in the context of cloud-based storage, in the sense that we design a control mechanism over the both data access and download request without loss of security and efficiency. Transparent Enclave Execution is designed in this paper, where this is for a distinct design setting. The security and experimental analysis for the systems are also presented. Transparent enclave execution refers to a security mechanism that allows software programs to execute within a secure enclave without requiring the program to be modified. Enclaves are isolated regions of memory within a computer's hardware that are designed to provide a secure environment for executing sensitive code.

Transparent enclave execution enables software developers to protect their applications against external attacks, such as hacking and tampering, by executing them within an enclave. This security measure is achieved through a process known as dynamic binary translation, which translates the program's instructions into an enclave-compatible format at runtime. By leveraging this technology, developers can ensure the confidentiality and integrity of their software, as well as protect the sensitive data it handles. Transparent enclave execution is increasingly being used in a variety of industries, including finance, healthcare, and government, where data security is paramount.

| | TABLE OF CONTENTS | | |
|---|---|---|---|
| **Chapter No** | **TITLE** | | **Page No.** |
| | **ABSTRACT** | | v |
| | **LIST OF FIGURES** | | vii |
| 1 | **INTRODUCTION** | | 1 |
| 2 | **LITERATURE SURVEY** | | |
| | 2.1 | Inferences from Literature Survey | 3 |
| | 2.2 | Open Problems in Existing System | 6 |
| 3 | **REQUIREMENTS ANALYSIS** | | |
| | 3.1 | Feasibility Studies/Risk Analysis of the Project | 8 |
| | 3.2 | Software Requirements Specification Document | 9 |
| | 3.3 | System Use case | 9 |
| | 3.4 | Activity overflow | 10 |
| 4 | **DESCRIPTION OF PROPOSED SYSTEM** | | |
| | 4.1 | Selected Methodology or process model | 11 |
| | 4.2 | Architecture / Overall Design of Proposed System | 12 |
| | 4.3 | Use case diagram | 13 |
| | 4.4 | System Study | 14 |
| | 4.5 | Security Model | 15 |
| | 4.6 | Authentication Encryption with associated data | 16 |
| **5** | **IMPLEMENTATION DETAILS** | | |
| | 5.1 | Implementation and Specific Requirements | 17 |
| | 5.2 | Deployment Setup | 18 |
| | 5.3 | Data Flow Model | 18 |
| | 5.4 | Collaboration Model | 19 |

.

# LIST OF FIGURES

# CHAPTER-1
# INTRODUCTION

Cloud-based storage service has attracted considerable attention from both academia and industries. It may be widely used in many Internet-based commercial applications (e.g., Apple iCloud) due to its long-list benefits including access flexibility and free of local data management. The increasing number of individuals and companies nowadays prefer to outsource their data to the remote cloud in such a way that they may reduce the cost of upgrading their local data management facilities/devices.

However, the worry of security breaches over outsourced data may be one of the main obstacles hindering Internet users from widely using cloud-based storage services. In many practical applications, outsourced data may need to be further shared with others. For example, a Dropbox user Alice may share photos with her friends. Without using data encryption, prior to sharing the photos, Alice needs to generate a sharing link and further share the link with friends. Although guaranteeing some level of access control over unauthorized users (e.g., those are not Alice's friends), the sharing link may be visible within the Dropbox administration level (e.g., the administrator could reach the link). Since the cloud (which is deployed in an open network) is not be fully trusted, it is generally recommended to encrypt the data prior to being uploaded to the cloud to ensure data security and privacy. One of the corresponding solutions is to directly employ an encryption technique (e.g., AES) on the outsourced data before uploading to cloud, so that only specified cloud user (with a valid decryption key) can gain access to the data via valid decryption. To prevent shared photos being accessed by the "insiders" of the system, a straightforward way is to designate the group of authorized data users prior to encrypting the data.

In some cases, nonetheless, Alice may have no idea who the photo receivers/users will be. It is possible that Alice only has knowledge of attributes w.r.t. photo receivers. In this case, traditional public key encryption (e.g., Paillier Encryption), which requires the encryptor to know who the data receiver is in

advance, cannot be leveraged. Providing policy-based encryption mechanism over the outsourced photos is therefore desirable, so that Alice makes use of the mechanism to define access policy over the encrypted photos to guarantee only a group of authorized users is able to access the photos. In a cloud-based storage service, there exists a common attack that is well-known as resource-exhaustion attack. Since a (public) cloud may not have any control over download request (namely, a service user may send unlimited numbers of download request to cloud server), a malicious service user may launch the denial-of-service (DoS)/distributed denial-of-service (DDoS) attacks to consume the resource of cloud storage service server so that the cloud service could not be able to respond honest users' service requests. As a result, in the "pay-as-you-go" model, economic aspects could be disrupted due to higher resource usage. The costs of cloud service users will rise dramatically as the attacks.

In the paper, a new mechanism, dubbed dual access control, to tackle the above two problems. To secure data in cloud-based storage service, attribute-based encryption (ABE) is one of the promising candidates that enables the confidentiality of out sourced data as well as fine-grained control over the outsourced data. Ciphertext-Policy ABE (CP-ABE) provides an effective way of data encryption such that access policies, defining the access privilege of potential data receivers, can be specified over encrypted data. Note that we consider the use of CP-ABE in our mechanism in this paper. Nevertheless, simply employing CP-ABE technique is not sufficient to design an elegant mechanism guaranteeing the control of both data access and download request.

# CHAPTER-2

# LITERATURE SURVEY

[1] Alexandros Bakas and Antonis Michalas. Modern Family: A revocable hybrid encryption scheme based on attribute-based encryption, symmetric searchable encryption and SGX. In SecureComm 2019, pages 472–486, 2019.

Secure cloud storage is considered as one of the most important issues that both businesses and end-users consider before their private data to the cloud. Lately, we have seen some interesting approaches that are based either on the promising concept of Symmetric Searchable Encryption (SSE) or on the well-studied field of Attribute-Based Encryption (ABE). In this paper, we propose a hybrid encryption scheme that combines both SSE and ABE by utilizing the advantages of both these techniques. In contrast to many approaches, we design a revocation mechanism that is completely separated from the ABE scheme and solely based on the functionality offered by SGX.

[2] Antonis Michalas. The lord of the shares: combining attribute-based encryption and searchable encryption for flexible data sharing. In SAC 2019, pages 146–155, 2019.

Secure cloud storage is considered one of the most important issues that both businesses and end-users are considering before moving their private data to the cloud. Lately, we have seen some interesting approaches that are based either on the promising concept of Symmetric Searchable Encryption (SSE) or on the well-studied field of Attribute-Based Encryption (ABE). In the first case, researchers are trying to design protocols where users' data will be protected from both *internal* and *external* attacks without paying the necessary attention to the problem of user revocation. On the other hand, in the second case existing approaches address the problem of revocation. However, the overall efficiency of these systems is compromised since the proposed protocols are solely based on ABE schemes and the size of the produced ciphertexts and the time required to decrypt grows with the complexity of the access formula. In this paper, we propose a protocol that combines *both* SSE and ABE in a way that the main advantages of

each scheme are used. The proposed protocol allows users to directly search over encrypted data by using an SSE scheme while the corresponding symmetric key that is needed for the decryption is protected via a Ciphertext-Policy Attribute-Based Encryption scheme.

[3] G. Wang, C. Liu, Y. Dong, P. Han, H. Pan, and B. Fang, “Decrypt: A multi-user searchable symmetric encryption scheme for cloud applications,” IEEE Access, vol. 6, pp. 2908–2921, 2018.

Searchable Encryption (SE) has been extensively examined by both academic and industry researchers. While many academic SE schemes show provable security, they usually expose some query information (e.g., search and access patterns) to achieve high efficiency. However, several inference attacks have exploited such leakage, e.g., a query recovery attack can convert opaque query trapdoors to their corresponding keywords based on some prior knowledge. On the other hand, many proposed SE schemes require significant modification of existing applications, which makes them less practical, weak in usability, and difficult to deploy. In this paper, we introduce a secure and practical searchable symmetric encryption scheme with provable security strength for cloud applications, called IDCrypt, which improves the search efficiency, and enhances the security strength of SE using symmetric cryptography. We further point out the main challenges in securely searching on multiple indexes and sharing encrypted data between multiple users. To address the above issues, we propose a token-adjustment search scheme to preserve the search functionality among multi-indexes, and a key-sharing scheme that combines identity-based encryption and public-key encryption. Our experimental results show that the overhead of the key-sharing scheme is low.

[4] Kaiping Xue, Weikeng Chen, Wei Li, Jianan Hong, and Peilin Hong. Combining data owner-side and cloud-side access control for encrypted cloud storage. IEEE Transactions on Information Forensics and Security, 2018.

People endorse the great power of cloud computing, but cannot fully trust the cloud providers to host privacy-sensitive data, due to the absence of user-to-cloud

controllability. To ensure confidentiality, data owners outsource encrypted data instead of plaintexts. To share the encrypted files with other users, ciphertext-policy attribute-based encryption (CP-ABE) can be utilized to conduct fine-grained and owner-centric access control. But this does not sufficiently become secure against other attacks. Many previous schemes did not grant the cloud provides the capability to verify whether a downloader can decrypt. Therefore, these files should be available to everyone accessible to the cloud storage. A malicious attacker can download thousands of files to launch economic denial of sustainability (EDoS) attacks, which will largely consume the cloud resource. The payer of the cloud service bears the expense. Besides, the cloud provider serves both as the accountant and the payee of resource consumption fees, lacking transparency to data owners. These concerns should be resolved in real-world public cloud storage. In this paper, we propose a solution to secure encrypted cloud storages from EDoS attacks and provide resource consumption accountability. It uses CP-ABE schemes in a black-box manner and complies with arbitrary access policy of the CP-ABE. We present two protocols for different settings, followed by performance and security analysis

[5] Jianting Ning, Zhenfu Cao, Xiaolei Dong, Kaitai Liang, Hui Ma, and Lifei Wei. Auditable σ-time outsourced attribute-based encryption for access control in cloud computing. IEEE Transactions on Information Forensics and Security, 13(1):94–105, 2018.

As a sophisticated mechanism for secure fine-grained access control over encrypted data, ciphertext-policy attribute-based encryption (CP-ABE) is one of the highly promising candidates for cloud computing applications. However, there exist two main long-lasting open problems of CP-ABE that may limit its wide deployment in commercial applications. One is that decryption yields expensive pairing cost which often grows with the increase of access policy size. The other is that one is granted access privilege for unlimited times as long as his attribute set satisfies the access policy of a given ciphertext. Such powerful access rights, which are provided by CP-ABE, may be undesirable in real-world applications (e.g., pay-as-you-use). To address the above drawbacks, in this paper, we propose a new notion called auditable σ-time outsourced CF-ABE, which is believed to be

applicable to cloud computing. In our notion, an expensive pairing operation incurred by decryption is offloaded to the cloud and meanwhile, the correctness of the operation can be audited efficiently. Moreover, the notion provides σ-time fine-grained access control. The cloud service provider may limit a particular set of users to enjoy access privilege for at most σ times within a specified period. As of independent interest, the notion also captures key-leakage resistance. The leakage of a user's decryption key does not help a malicious third party in decrypting the ciphertexts belonging to the user. We design a concrete construction (satisfying our notion) in the key encapsulation mechanism setting based on Rouselakis and Waters (prime order) CP-ABE, and further present security and extensive experimental analysis to highlight the scalability and efficiency of our construction

## 2.1 Open Problems in Existing System

Transparent enclave execution is a relatively new technology that has been developed to enable the secure execution of code within trusted hardware enclaves. While the technology has shown promise in addressing various security concerns related to cloud computing and data privacy, there are still some open problems that need to be addressed.
Here are a few open problems in existing systems for transparent enclave execution:

***2.1.1 Limited functionality:*** One of the biggest limitations of existing systems for transparent enclave execution is their limited functionality. While these systems can protect the confidentiality and integrity of code and data, they often lack support for complex computations, which can limit their usefulness.

***2.1.2 Performance overhead:*** Transparent enclave execution systems typically come with a significant performance overhead, which can impact their scalability and cost-effectiveness. Improving the performance of these systems is a major challenge that needs to be addressed.

***2.1.3 Attack surface:*** Enclave execution systems are only as secure as the hardware and software they run on. Attackers can exploit vulnerabilities in the underlying system to bypass the security measures provided by the enclave. This means that there is a significant attack surface that needs to be addressed to make these systems truly secure.

***2.1.4 Trustworthiness of the hardware:*** Hardware security is critical to the success of transparent enclave execution systems. However, ensuring the trustworthiness of hardware components can be challenging. This is particularly true for commodity hardware, where it can be difficult to verify that the hardware has not been tampered with or compromised.

***2.1.5 Interoperability:*** Existing systems for transparent enclave execution often lack interoperability with other systems, which can limit their usefulness. For example, an enclave application developed for one platform may not be compatible with another platform, which can lead to vendor lock-in and hinder innovation.

Addressing these challenges will require ongoing research and development efforts in the areas of hardware security, software design, and system architecture. As transparent enclave execution continues to gain popularity, it is likely that these challenges will be addressed over time.

# CHAPTER-3

# REQUIREMENT ANALYSIS

## 3.1 Feasibility Studies/Risk Analysis of the Project:

Here are some potential risks associated with transparent enclave execution:

***3.1.1 Hardware vulnerabilities:*** Transparent enclave execution relies on trusted hardware to ensure the security of the execution environment. However, hardware components can still have vulnerabilities that can be exploited by attackers. Mitigating this risk involves ensuring that the hardware components used in the system are thoroughly tested and certified before deployment.

***3.1.2 Software vulnerabilities:*** The software used to implement and manage the enclave execution environment can also have vulnerabilities that can be exploited by attackers. Mitigating this risk involves ensuring that the software is designed and implemented using secure development practices, and that it is regularly updated to address known vulnerabilities.

***3.1.3 Performance overhead:*** Transparent enclave execution systems can have significant performance overhead, which can impact the usability and cost-effectiveness of the system. Mitigating this risk involves designing the system to minimize performance overhead and ensuring that the system is properly scaled to meet the needs of the users.

***3.1.4 Limited functionality:*** Transparent enclave execution systems may have limited functionality compared to traditional computing environments, which can limit their usefulness for certain types of applications. Mitigating this risk involves carefully assessing the requirements of the application and ensuring that the system can meet those requirements.

***3.1.5 Compliance risks:*** Transparent enclave execution systems may be subject to various legal and regulatory requirements related to data privacy and security. Mitigating this risk involves ensuring that the system is designed and operated in

compliance with these requirements.

### 3.2 Software Requirements Specification Document:

- Operating System : Windows 7/8/10
- IDE : Py charm
- Server-side scripts : HTML, CSS, Js
- Libraries Used : Pandas, Smtplib, Flask
- Technology : Python 3.6+

### 3.3 System Use Case:

***3.3.1 Scenario:*** A company wants to process sensitive data in the cloud, but is concerned about data privacy and security.

***3.3.2 Solution:*** The company can use transparent enclave execution to securely process sensitive data in a cloud environment. The system architecture would include a trusted hardware enclave that provides a secure execution environment for the code and data. The enclave would be hosted on a cloud provider's infrastructure, with the provider managing the underlying hardware and software.

***3.3.3 Workflow:*** The company would develop its application code and data processing logic in a standard programming language, such as C or Python. The code would be compiled into an enclave executable that can be loaded into the trusted hardware enclave. The enclave executable would be encrypted and signed to ensure its integrity and confidentiality.

***3.3.4 Data Flow:*** The sensitive data would be encrypted before it is sent to the enclave for processing. Once the data is inside the enclave, it would be decrypted and processed by the enclave application. The results of the processing would be encrypted and sent back to the company for further analysis or storage.

### 3.4 ACTIVITY WORKFLOW

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

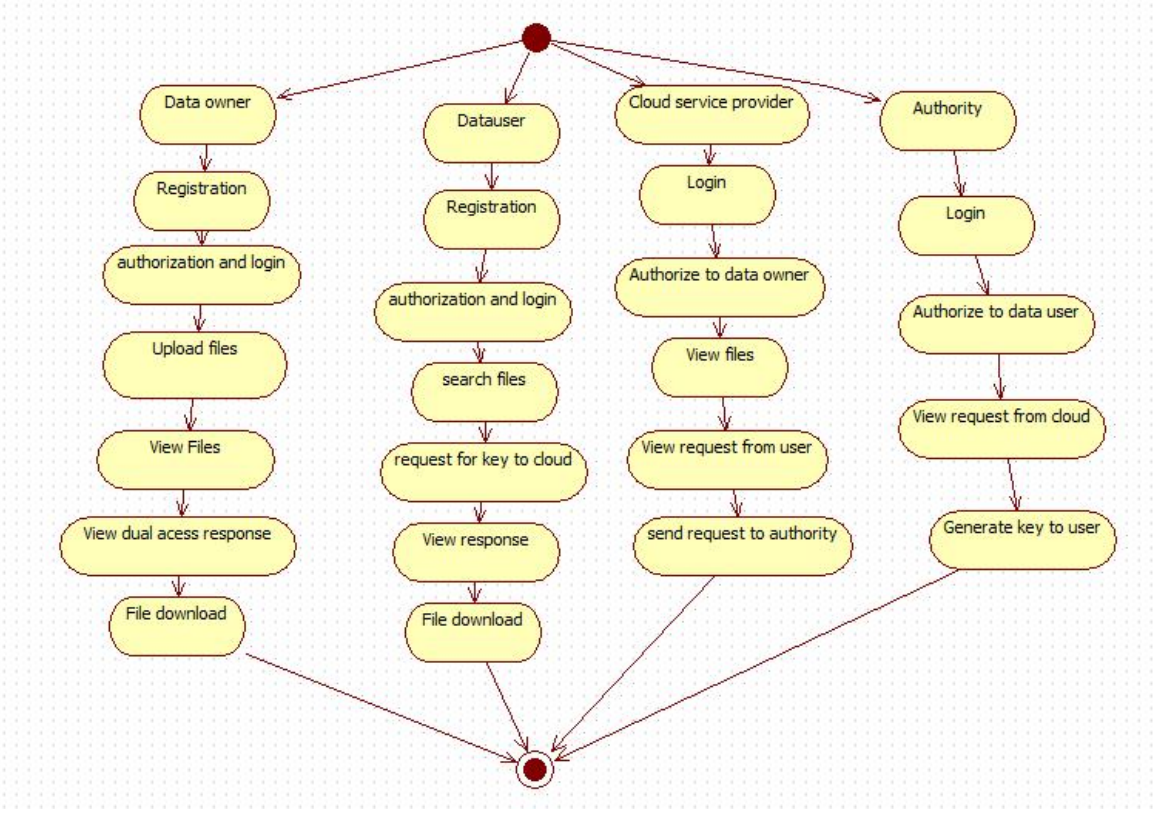



**Fig: 3.1 Activity Diagram**

# CHAPTER-4

# DESCRIPTION OF THE PROPOSED SYSTEM

## 4.1 METHODOLOGY

***4.1.1 Designing the enclave:*** The first step in implementing a transparent enclave system is to design the enclave itself. This involves identifying the sensitive computations that need to be executed within the enclave, and determining the appropriate security measures that should be used to protect the data processed within the enclave.

***4.1.2 Developing the enclave:*** Once the design is complete, the next step is to develop the enclave. This involves using enclave development tools, such as SDKs and compilers, to create the enclave binary. The enclave should be designed to accept inputs and produce outputs in a way that is compatible with the application it is being integrated with.

***4.1.3 Integrating the enclave with the application:*** The enclave must be integrated with the application in a way that enables the application to securely communicate with the enclave. This involves modifying the application to include functionality for securely communicating with the enclave, and designing protocols for exchanging data between the application and the enclave.

***4.1.4 Deploying the enclave:*** Once the enclave is complete and integrated with the application, the next step is to deploy the enclave in a production environment. This involves using deployment and management tools to configure the enclave and ensure that it is running correctly.

***4.1.5 Monitoring and auditing enclave activities:*** Once the enclave is deployed, it is important to monitor its activities to ensure that it is processing data correctly and securely. This involves implementing monitoring and auditing tools that can be used to detect any anomalies or suspicious activities within the enclave.

***4.1.6 Updating and patching the enclave:*** Over time, the enclave may need to

be updated or patched to address new security vulnerabilities or to add new functionality. This involves using deployment and management tools to update the enclave and ensure that it is running correctly.

In summary, the methodology of a transparent enclave system involves designing and developing an enclave, integrating it with an application, deploying it in a production environment, monitoring its activities, and updating it as necessary. Each of these steps is critical for ensuring the security and privacy of the data processed within the enclave, and must be carefully managed and monitored to minimize the risk of security breaches or other security incidents"

## 4.2 ARCHITECTURE OF PROPOSED SYSTEM

The model describes the structure of a database with the help of a diagram. The model is a design or blueprint of a database that can later be implemented as a database. The diagram shows the relationship among entity sets. It 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, it shows the complete logical structure of a database.

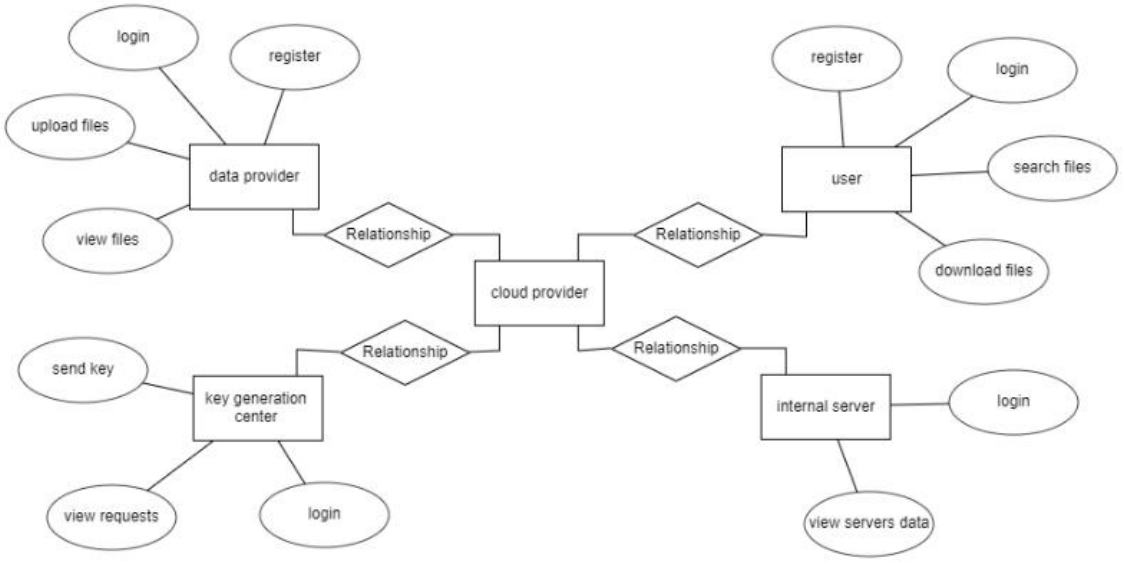



***Fig: 4.1 Architecture diagram***

## 4.3 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.

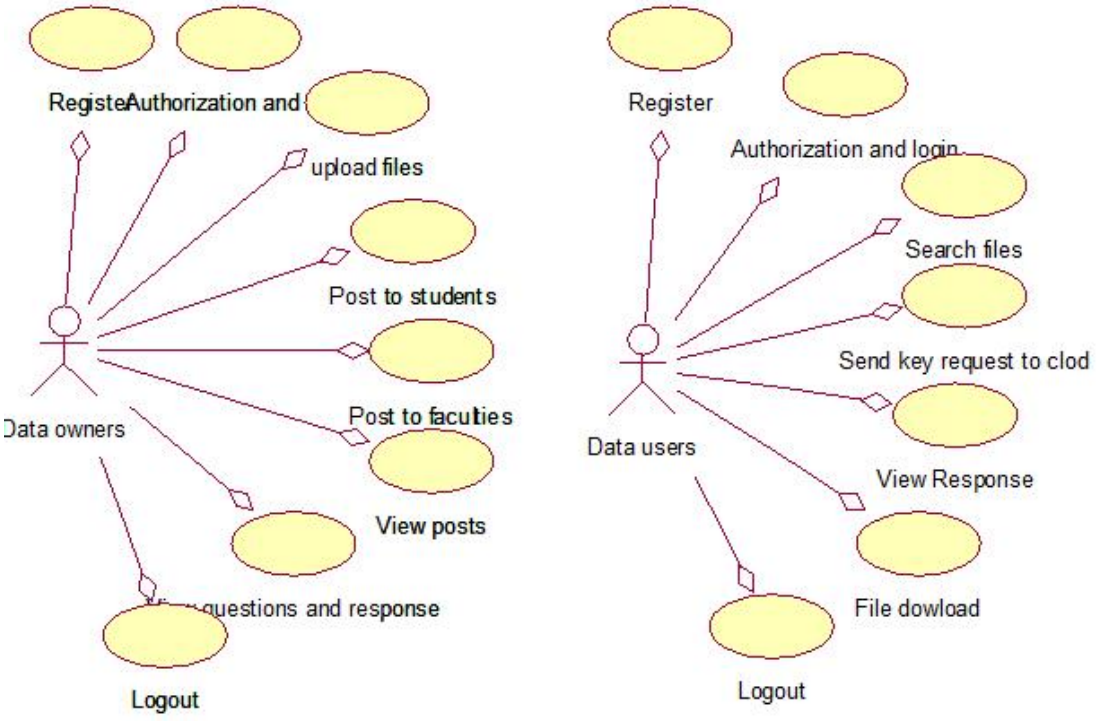



**Fig: 4.2 USE CASE DIAGRAM OF PROPOSED MODEL**

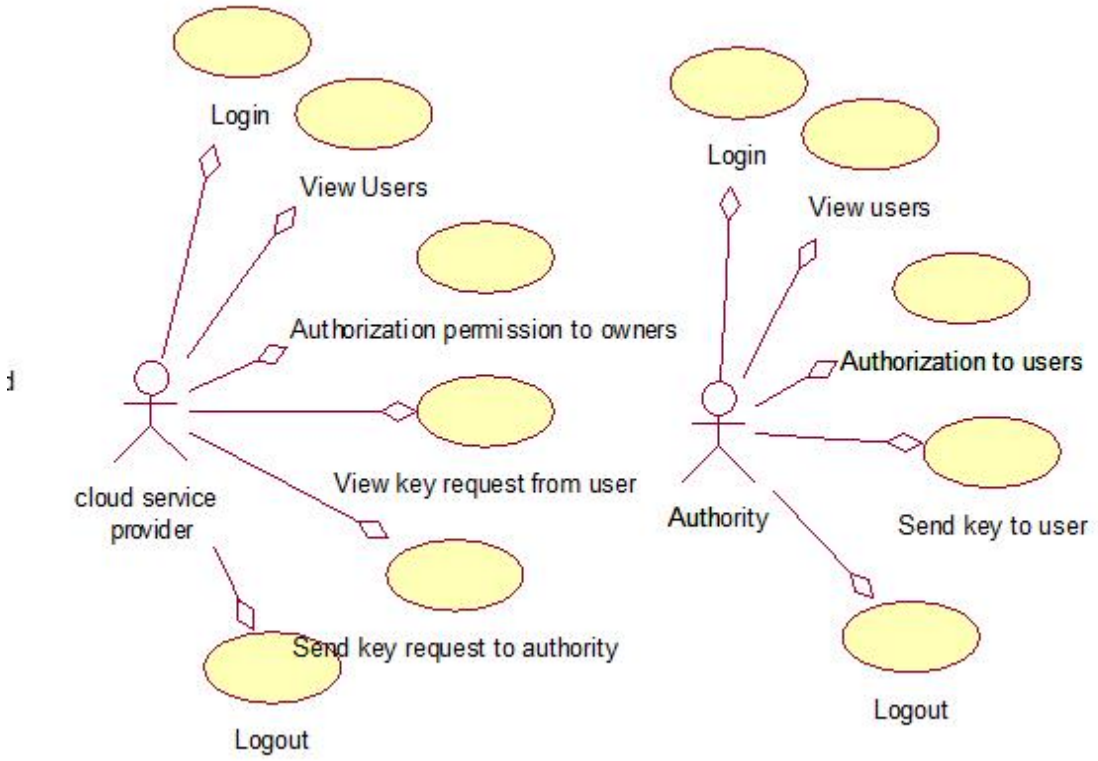



***Fig : 4.3 USE CASE DIAGRAM***

## 4.4 SYSTEM STUDY

### 4.4.1 FEASIBILITY STUDY:

The feasibility of the project is analyzed 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:

(i) ECONOMICAL FEASIBILITY

(ii) TECHNICAL FEASIBILITY

(iii) SOCIAL FEASIBILITY

#### *4.4.1.1 ECONOMICAL 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.

#### *4.4.1.2 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.

#### *4.4.1.3 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.

### 4.5 SYSTEM ARCHITECTURE AND SECURITY MODEL

The architectures of our dual access control systems for cloud data sharing. Concretely, the systems consist of the following entities:

i. Authority is responsible for initializing system parameters and data user registration. Also, it handles the call request from the cloud in the first proposed construction.

ii. Data owner holds the data and wants to outsource his data to the cloud. Data owners (only) want to share their data with those who satisfy certain conditions (e.g., professors or associate professors).

iii. Data user wants to download and decrypt the encrypted data shared in the cloud. Those who are authorized can download the encrypted file and further decrypt it to access the plaintext.

iv. Cloud provides convenient storage service for data owners and data users. Specifically, it stores the outsourced data from data users and handles the download requests sent by data users.

v. Enclave handles the call request from the cloud (used in the second system). The description of workflow is introduced as follows. Data owners encrypt their data under the access policies chosen by themselves and upload the encrypted data to the cloud. Authorized data users can download the shared data by sending a download request to the cloud. Upon receiving a download request from an authorized data user, the cloud does as follow.

vi. For our basic system, the cloud sends a call request to the authority. After receiving a response from the authority, the cloud sends a response back to the data user for our enhanced system, the cloud sends a call request to the enclave. After receiving a response from the enclave, the cloud sends a response back to the data user.

## 4.6 AUTHENTICATED ENCRYPTION WITH ASSOCIATED DATA

Authenticated encryption with associated data (AEAD) is a form of symmetric-key encryption that simultaneously provides confidentiality as well as integrity. A symmetric-key encryption scheme SE mainly consists of the following two PPT algorithms:

i. On input of a message m and a symmetric key it outputs a ciphertext.

ii. On input a symmetric key and a ciphertext it outputs a message m.

A symmetric-key encryption scheme SE should be semantically secure under a chosen plaintext attack.

# CHAPTER-5

# IMPLEMENTATION DETAILS

## 5.1 IMPLEMENTATION AND SPECIFIC REQUIREMENTS

Transparent Enclave Execution requires a comprehensive understanding of enclave-based security and the specific requirements of the project. Here is an overview of the steps involved in building a project that uses Transparent Enclave Execution:

i. Determine the requirements of the project: Before implementing any security measure, it is important to determine the requirements of the project. Identify the sensitive data that needs to be protected, the potential threats to the system, and any regulatory or compliance requirements.

ii. Design the enclave: Once the requirements are determined, design the enclave that will be used to protect the sensitive data. Determine the size of the enclave, the type of security measures to be used, and the cryptographic primitives to be employed.

iii. Develop the application code: Develop the application code that will be executed within the enclave. This code should only access the sensitive data and perform the necessary computations within the enclave.

iv. Compile the enclave: Compile the enclave code using an appropriate compiler that supports enclave-based security, such as the Intel SGX SDK.

v. Initialize the enclave: Initialize the enclave by loading the enclave binary into memory and creating an enclave instance. This process typically involves establishing a secure channel between the application and the enclave.

vi. Execute the code within the enclave: Once the enclave is initialized, execute the application code within the enclave. Any sensitive data

processed by the code will be protected by the enclave's security measures.

## 5.2 DEPLOYMENT SETUP

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.

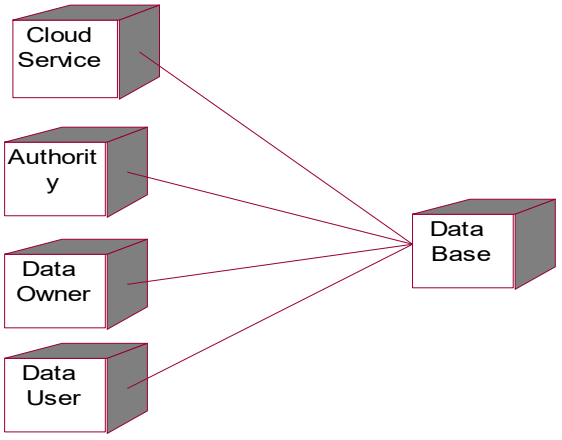



**Fig: 5.1 Deployment diagram**

## 5.3 DATA FLOW MODEL

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.

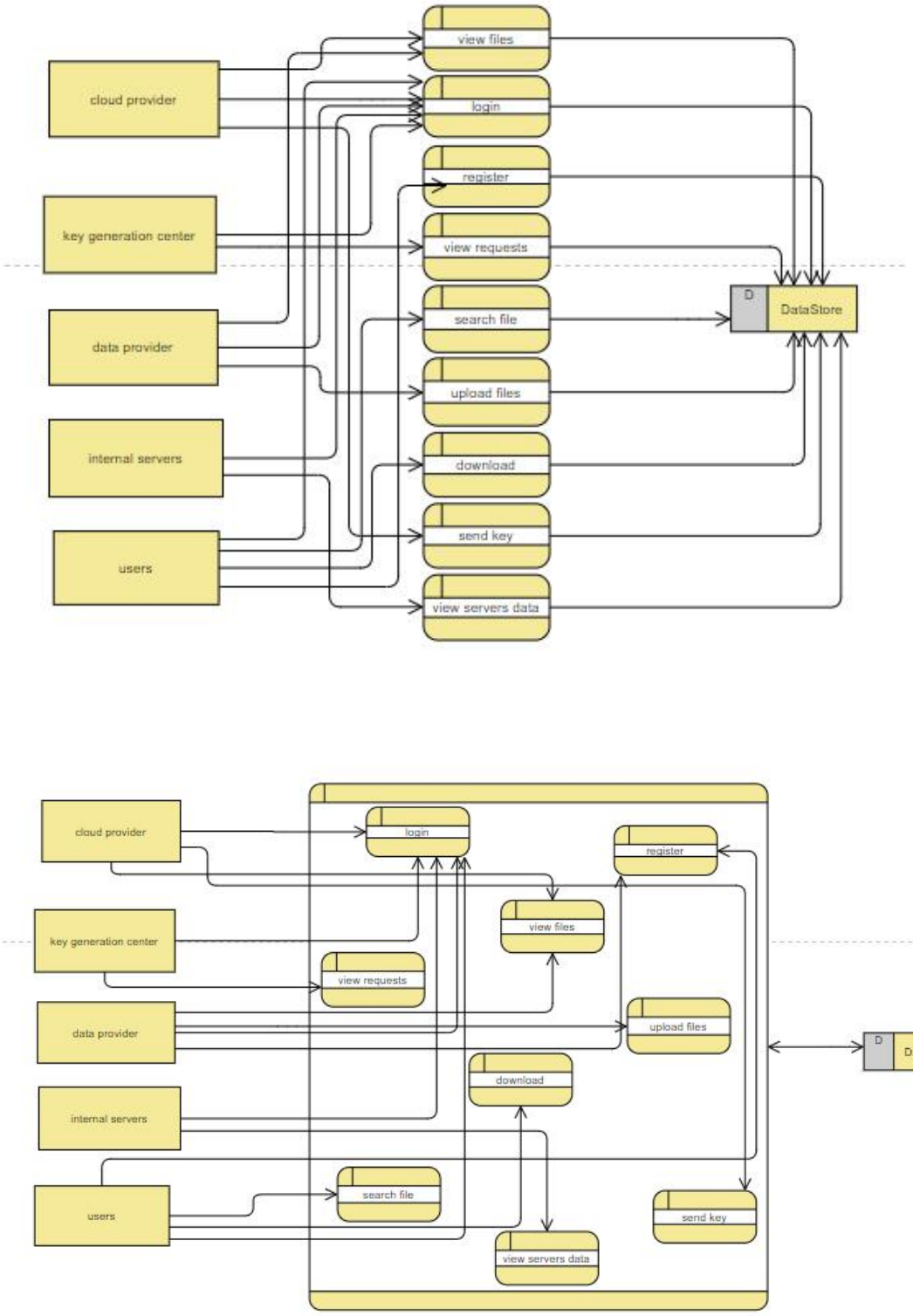



**Fig: 5.2 DFD diagram**

## 5.4 COLLABORATION SEQUENCE

In collaboration sequence 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.

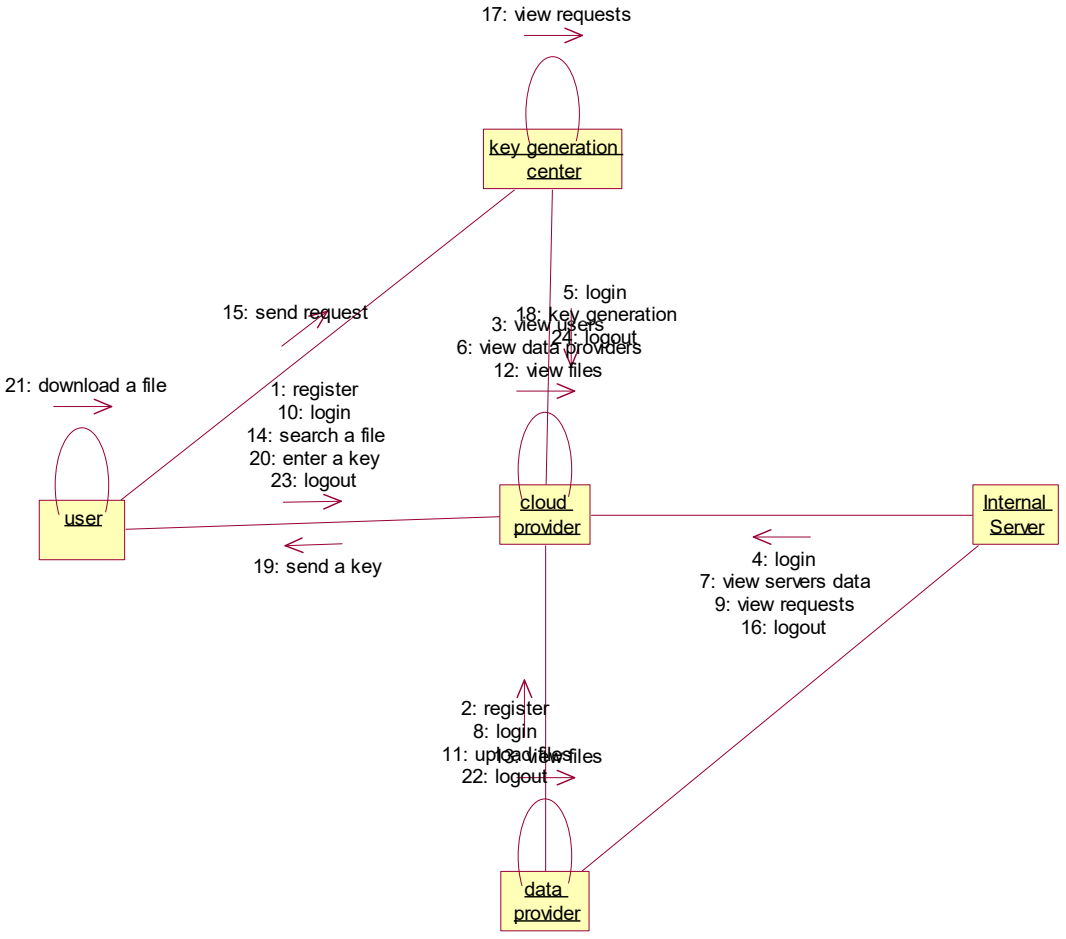



**Fig: 5.3 Collaboration diagram**

## 5.5 SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

### TYPES OF TESTS:

#### 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.

## 5.6 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.

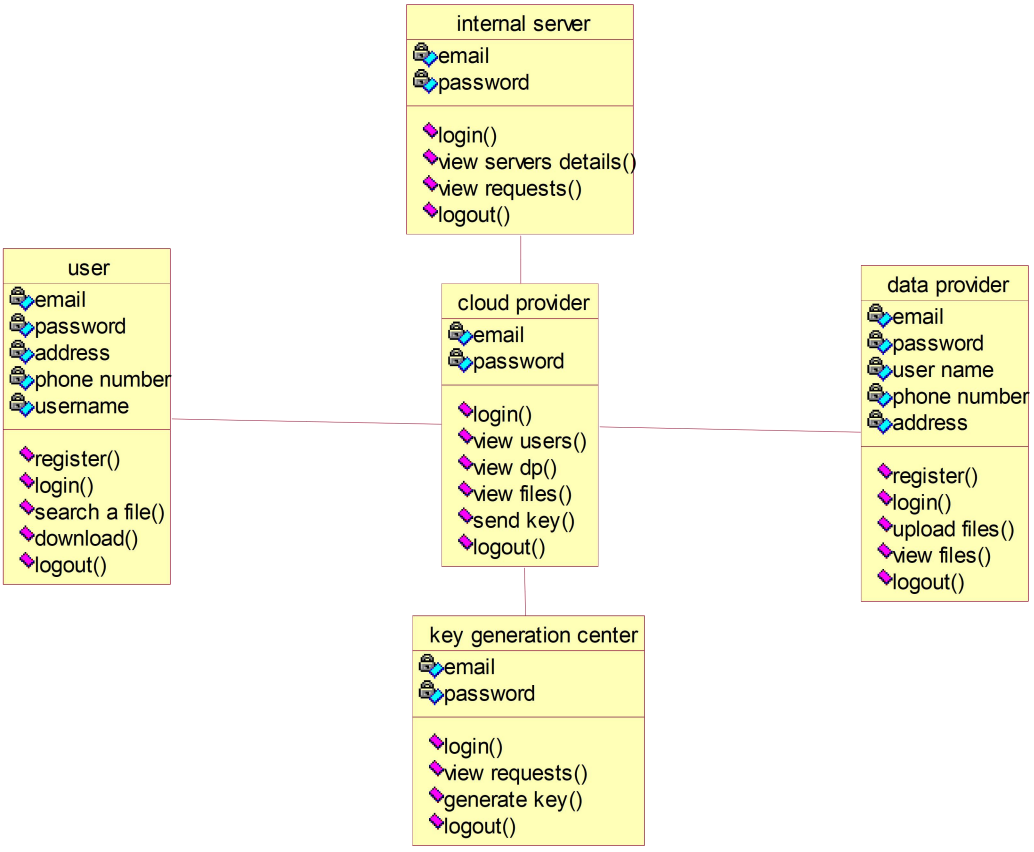



**Fig: 5.4 Class diagram of the model**

# CHAPTER-6
# RESULTS AND DISCUSSION

Thus far, we have seen how authenticated and integrity protected execution enables efficient constructions of sealed glass proofs. The transparent enclave execution model is highly conservative: it assumes complete leakage of a program's state. The systems we propose are with the following distinct features:

i. Confidentiality of outsourced data. In our proposed systems, the outsourced data is encrypted prior to being uploaded to cloud. No one can access them without valid access rights.

ii. Anonymity of data sharing. Given an outsourced data, cloud server cannot identify data owner, so that the anonymity of owner can be guaranteed in data storage and sharing.

iii. Fine-grained access control over outsourced (encrypted) data. Data owner keeps controlling his encrypted data via access policy after uploading the data to cloud. A data owner can encrypt his outsourced data under a specified access policy such that only a group of authorized data users, matching the access policy, can access the data.

iv. Control over anonymous download request and EDoS attacks resistance. A cloud server is able to control the download request issued by any system user, where the download request can set to be anonymous. With the control over download request, we state that our systems are resistant to EDoS attacks.

v. High efficiency. Our proposed systems are built on the top of the CP-ABE system. Compared with they do not incur significant additional computation and communication overhead. This makes the systems feasible for real-world applications.

# CHAPTER-7
# CONCLUSION

## 7.1 CONCLUSION:

Transparent Enclave is a security mechanism used to protect data in cloud data storage. This mechanism provides two layers of access control, where one layer is managed by the cloud service provider, and the other layer is managed by the data owner. This control mechanism ensures that only authorized personnel can access the data stored in the cloud. Transparent Enclave is a security feature that allows applications to execute in a secure, isolated environment called an enclave. Enclaves are protected from unauthorized access, tampering, and observation by other processes running on the same system.

Overall, the use of Transparent control in cloud data storage is an effective security measure that helps to protect sensitive data. However, it is important to note that this security measure should be combined with other security measures such as encryption, regular backups, and strong passwords to provide comprehensive data protection. It provides strong security guarantees for sensitive computations, such as those involving encryption keys, user credentials, and other confidential data. However, it is important to note that it is not a silver bullet solution and has its own limitations and potential vulnerabilities that need to be carefully considered and addressed. We showed two dual access control systems and addressed an intriguing and pervasive issue with cloud-based data sharing. DDoS/EDoS assaults cannot be used against the suggested systems. We claim that different CP-ABE constructions may "transplant" the method utilized to accomplish the feature of control on download request.

The suggested solutions don't incur a large computational or communication overhead, according to the findings of our experiments (compared to its underlying CP-ABE building block). We take use of the fact that the secret information entered the enclave cannot be recovered in our improved system. The memory access patterns or other comparable side-channel attacks reveal that enclave may,

nevertheless, leak part of its secret(s) to a hostile host. Thus, introduces the transparent enclave execution approach.

In conclusion, Transparent Enclave Execution is an essential security mechanism for cloud data storage that helps to safeguard sensitive information. Its implementation is critical in ensuring the confidentiality, integrity, and availability of data stored in the cloud.

### 7.2 FUTURE WORK:

Transparent Enclave Execution is a rapidly evolving technology, and there are several areas of future work that could potentially enhance its effectiveness and usability. Some of these areas of research and development include:

Improved performance: One major challenge of enclave-based security is the performance overhead that can be incurred by executing code within an enclave. Future work could focus on developing more efficient implementations of enclaves, such as optimized compilers and hardware acceleration, to reduce this overhead.

Broader hardware support: Currently, enclave-based security is only supported by a limited number of hardware platforms, such as Intel SGX. Future work could focus on expanding support for enclave-based security to a wider range of hardware platforms, such as ARM-based processors, to increase its applicability and accessibility.

Enhanced security guarantees: While enclave-based security provides strong security guarantees for sensitive computations, there is still potential for side-channel attacks and other vulnerabilities that could compromise the security of the enclave. Future work could focus on developing more robust security measures, such as secure hardware-based random number generators and side-channel-resistant encryption schemes, to further enhance the security of enclaves.

Standardization and interoperability: Currently, enclave-based security is implemented in a vendor-specific manner, which can limit interoperability and

hinder the development of standardized security protocols. Future work could focus on developing standards for enclave-based security that promote interoperability and enable secure communication between enclaves implemented by different vendors.

### 7.3 IMPLEMENTATION ISSUES:

Implementation issues of transparent enclave execution:

Transparent Enclave Execution (TEE) is a security technology that enables secure execution environments for sensitive software applications. TEEs are implemented using hardware support, such as Intel SGX, AMD SEV, or ARM Trust Zone, and software frameworks, such as Open Enclave or Fortanix SDK. However, the implementation of TEEs also faces several issues that need to be addressed. In this answer, I will discuss some of the implementation issues of TEEs in detail.

Limited resources: TEEs operate in a restricted environment with limited resources, including memory and CPU cycles. Therefore, implementing TEEs requires optimization and efficient use of available resources.

Platform-specific implementation: The implementation of TEEs depends on the underlying hardware and software platform, and the implementation may vary from one platform to another. This may result in platform-specific implementations that may not be interoperable with other platforms.

Compatibility: TEEs need to be compatible with the software and hardware platforms they are deployed on. This requires careful consideration of compatibility issues during the design and implementation of TEEs.

Integration with existing systems: TEEs need to be integrated with existing systems, including operating systems, virtualization technologies, and software applications. Integration with existing systems may require modifications to the existing software and hardware architecture, which may be challenging and time-consuming.

Trustworthiness of the TEE: The trustworthiness of the TEE is crucial since it

serves as a trusted execution environment. Any flaw in the TEE can compromise the security of the system. Therefore, the TEE must be secure and reliable, and free of vulnerabilities that can be exploited by attackers.

Performance overhead: TEEs impose significant performance overhead due to the use of encryption, decryption, and integrity checking mechanisms. The overhead can impact the usability and adoption of TEEs, particularly in resource-constrained environments.

Enclave lifecycle management: TEEs require proper management throughout their lifecycle, including creation, deployment, and decommissioning. Proper management is crucial to ensure the security of the enclave and prevent unauthorized access.

Multi-tenancy: multi-tenancy is a critical issue in TEEs since multiple users may share the same enclave. Researchers need to ensure that the enclave's resources are properly allocated and isolated to prevent one user from compromising the confidentiality and integrity of another user's data.

Standardization: Standardization of TEEs is essential to ensure that they are interoperable across different systems and architectures. Standardization can also help to improve the security and reliability of TEEs by providing a common set of guidelines and best practices.

### 7.4 RESEARCH ISSUES:

Transparent Enclave Execution (TEE) is a security technology that aims to provide secure execution environments for sensitive software applications. TEEs are designed to protect the confidentiality and integrity of the code and data that run within them, even from privileged system software and hardware.

TEEs face several research issues related to their transparent execution.
Performance overhead: TEEs impose significant performance overhead due to the

use of encryption, decryption, and integrity checking mechanisms. The overhead can impact the usability and adoption of TEEs, particularly in resource-constrained environments.

Compatibility and Interoperability: TEEs need to be compatible and interoperable with existing software and hardware architectures. Ensuring compatibility and interoperability is essential to the successful integration of TEEs into existing systems and the wider ecosystem.

User Authentication and Authorization: TEEs require robust user authentication and authorization mechanisms to ensure that only authorized users can access the protected resources within the enclave. Researchers need to ensure that the authentication and authorization mechanisms are secure and reliable, and do not compromise the confidentiality and integrity of the enclave.

Multi-tenancy: multi-tenancy is a critical issue in TEEs since multiple users may share the same enclave. Researchers need to ensure that the enclave's resources are properly allocated and isolated to prevent one user from compromising the confidentiality and integrity of another user's data.

Standardization: TEEs are a promising technology that can provide secure execution environments for sensitive software applications. However, researchers need to address several research issues related to their transparent execution.

# REFERENCES

[1] Alexandros Bakas and Antonis Michalas. Modern family: A revocable hybrid encryption scheme based on attribute-based encryption, symmetric searchable encryption and SGX. In SecureComm 2019, pages 472–486, 2019.

[2] Antonis Michalas. The lord of the shares: combining attribute-based encryption and searchable encryption for flexible data sharing. In SAC 2019, pages 146–155, 2019.

[3] Ben Fisch, Dhinakaran Vinayagamurthy, Dan Boneh, and Sergey Gorbunov. IRON: functional encryption using intel SGX. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, pages 765–782, 2017.

[4] Christofer Hoff. Cloud computing security: From ddos (distributed denial of service) to edos (economic denial of sustainability). http://www. rationalsurvivability.com/blog/?p=66.

[5] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In Advances in Cryptology-CRYPTO 1999, pages 537–554. Springer, 1999.

[6] F. Tramer, F. Zhang, H. Lin, J.-P. Hubaux, A. Juels, and E. Shi, “Sealedglass proofs: Using transparent enclaves to prove and sell knowledge,” in Proc. IEEE Eur. Symp. Security Privacy, 2017, pp. 19–34.

[7] G. Wang, C. Liu, Y. Dong, P. Han, H. Pan, and B. Fang, “Decrypt: A multi-user searchable symmetric encryption scheme for cloud applications,” IEEE Access, vol. 6, pp. 2908–2921, 2018.

[8] J. Li, Y. Wang, Y. Zhang, and J. Han, “Full verifiability for outsourced decryption in attribute-based encryption,” IEEE Trans. Services Comput., vol. 13, no. 3, pp. 478–487, May/Jun. 2020.

[9] Jianting Ning, Zhenfu Cao, Xiaolei Dong, Kaitai Liang, Hui Ma, and Lifei Wei. Auditable σ-time outsourced attribute-based encryption for access control in cloud computing. IEEE Transactions on Information Forensics and Security, 13(1):94–105, 2018.

[10] Kaiping Xue, Weikeng Chen, Wei Li, Jianan Hong, and Peilin Hong. Combining data owner-side and cloud-side access control for encrypted cloud storage. IEEE Transactions on Information Forensics and Security, 2018.

# APPENDIX

## A. SOURCE CODE

Authority:

```
<!DOCTYPE html>
<html lang="en">
   <head>
      <meta charset="utf-8" />
      <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-
fit=no" />
      <meta name="description" content="" />
      <meta name="author" content="" />
      <title>Creative - Start Bootstrap Theme</title>
      <!-- Favicon-->
      <link rel="icon" type="image/x-icon" href="static/assets/img/favicon.ico" />
      <link href="static/css/styles.css" rel="stylesheet" />
   </head>
   <body id="page-top">
      <nav class="navbar navbar-expand-lg navbar-light fixed-top py-3" id="mainNav">
         <div class="container">
               <ul class="navbar-nav ml-auto my-2 my-lg-0">
                  <li class="nav-item"><a class="nav-link js-scroll-trigger"
href="/home">Home</a></li>
                  <li class="nav-item"><a class="nav-link js-scroll-trigger"
href="/AuthorityViewUsers">View Users requests and generating key </a></li>
                  <li class="nav-item"><a class="nav-link js-scroll-trigger"
href="/Viewusersforauthorization">View Users for authorization</a></li>

<!--                  <li class="nav-item"><a class="nav-link js-scroll-trigger"
href="#services">View All Attackers</a></li>-->

                  <li class="nav-item"><a class="nav-link js-scroll-trigger"
href="/Authority">Logout</a></li>
               </ul>
            </div>
```

```
        </nav>
        <!-- Masthead-->
        <header class="masthead">
              <div class="row h-100 align-items-center justify-content-center text-center">
                 <h1 class="text-uppercase text-white font-weight-bold">Transparent Enclave
 Execution</h1>
                 {% if msg=="success" %}
 <!--                    <h2 style="color:red"></h2><br><br>-->
                    <br><h3 style="color:pink"> Login Successfully Completed, Welcome
 {{ Name }}!</h3>
                 {% endif %}
                 {% if msg=="Failure" %}
                     <h2 style="color:red">Invalid Details</h2><br><br>
                 {% endif %}
                   <div class="col-lg-10 align-self-end"></div>
                 <div class="col-lg-8 align-self-baseline"></div>
              </div>
        </header>
        <script src="static/js/scripts.js"></script>
     </body>
 </html>
```

Cloud Service Provider:

```
<!DOCTYPE html>
<html lang="en">
   <head>
      <meta charset="utf-8" />
      <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no" />
      <meta name="description" content="" />
      <meta name="author" content="" />
      <title>Creative - Start Bootstrap Theme</title>
      <!-- Favicon-->
      <link rel="icon" type="image/x-icon" href="static/assets/img/favicon.ico" />
      <link href="static/css/styles.css" rel="stylesheet" />
   </head>
   <body id="page-top">
      <nav class="navbar navbar-expand-lg navbar-light fixed-top py-3" id="mainNav">
         <div class="container">
               <ul class="navbar-nav ml-auto my-2 my-lg-0">
                  <li class="nav-item"><a class="nav-link js-scroll-trigger"
href="/homes">Home</a></li>
                  <li class="nav-item"><a class="nav-link js-scroll-trigger"
href="/ViewOwnersAndAuthorization">View Owners and Authorization </a></li>
                  <li class="nav-item"><a class="nav-link js-scroll-trigger" href="/ViewUsers">View
Users</a></li>
<!--                  <li class="nav-item"><a class="nav-link js-scroll-trigger" href="viewallfiles">View
All Files</a></li>-->
                  <li class="nav-item"><a class="nav-link js-scroll-trigger"
href="/ViewUsersrequest">View Users Requests</a></li>
<!--                  <li class="nav-item"><a class="nav-link js-scroll-trigger" href="#contact">View all
Attackers</a></li>-->
<!--                  <li class="nav-item"><a class="nav-link js-scroll-trigger" href="#contact">View
Rank Results</a></li>-->
                  <li class="nav-item"><a class="nav-link js-scroll-trigger" href="/csp">Logout</a></li>

               </ul>
            </div>
```

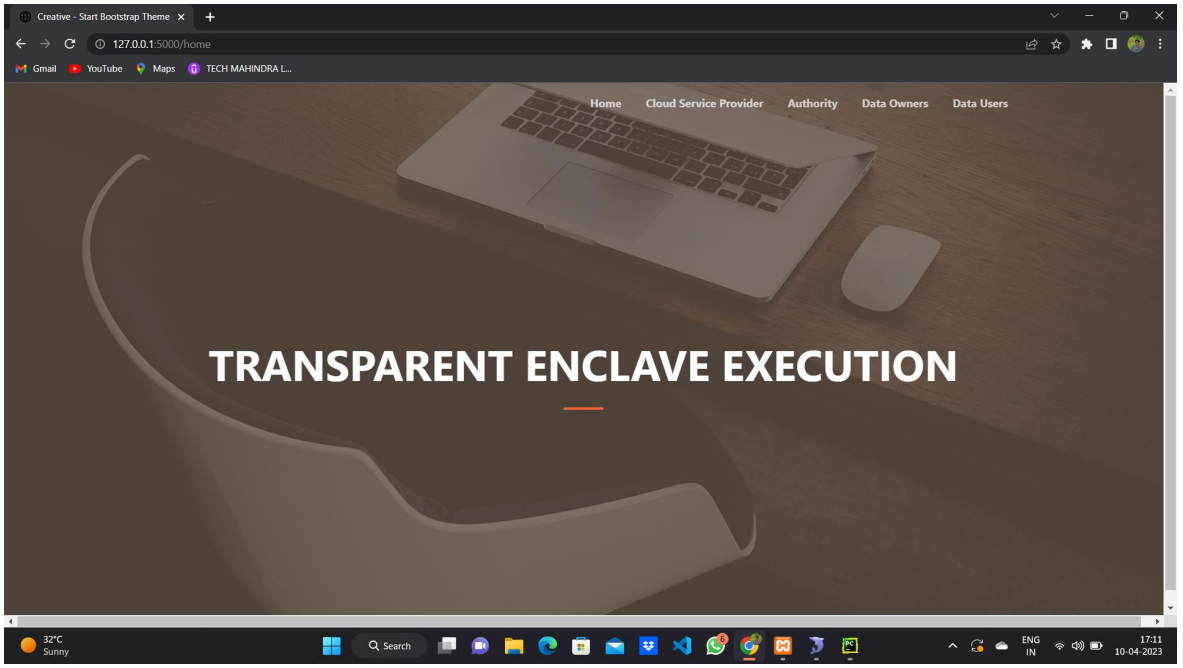
```
      </nav>
      <!-- Masthead-->
      <header class="masthead">
            <div class="row h-100 align-items-center justify-content-center text-center">
               <h1 class="text-uppercase text-white font-weight-bold">Transparent Enclave
Execution</h1>
               {% if msg=="success" %}
                  <h2 style="color:red">Login Successfully Completed,</h2><br><br>
                  <br><h3 style="color:pink"> Welcome {{ Name }}!</h3>
               {% endif %}
               {% if msg=="Failure" %}
                  <h2 style="color:pink">Invalid Details</h2>
               {% endif %}
                 <div class="col-lg-10 align-self-end"></div>
               <div class="col-lg-8 align-self-baseline"></div>
            </div>
      </header>
      <script src="static/js/scripts.js"></script>
   </body>
</html>
```

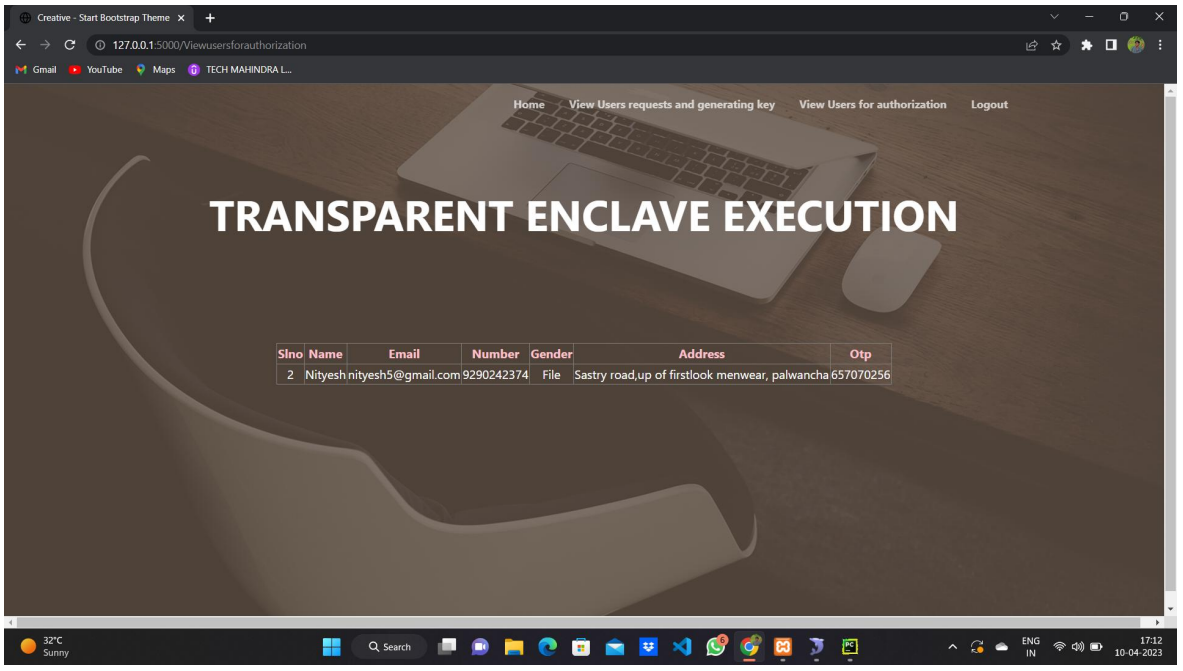
Data Owner HTML:

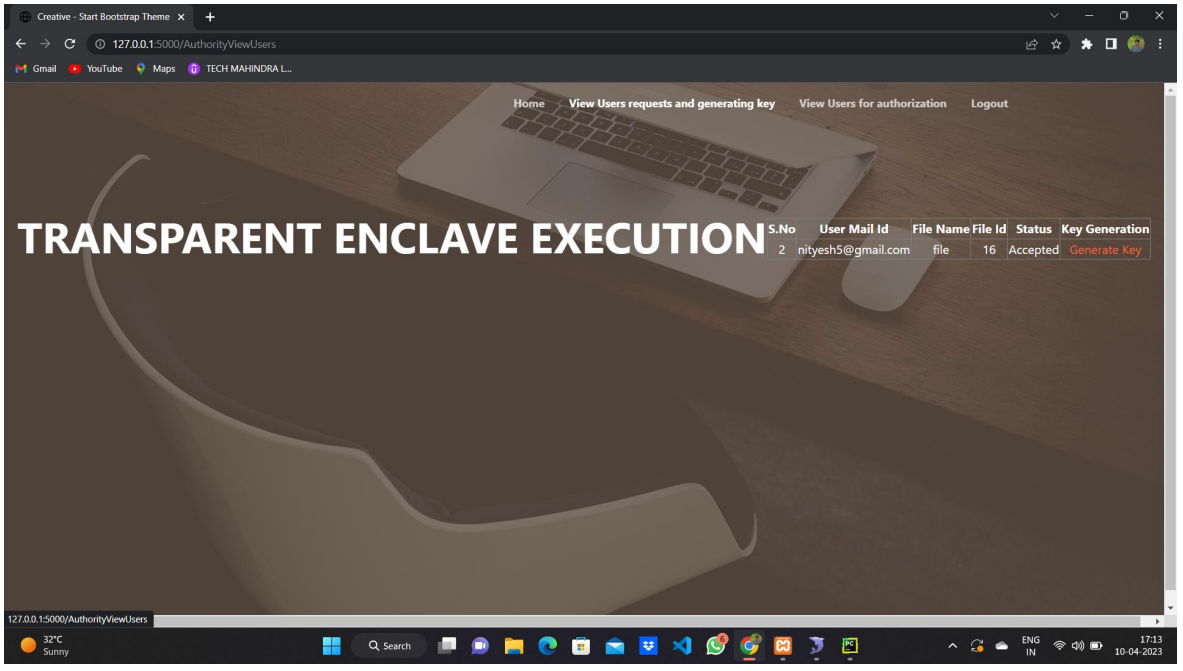
```
<!DOCTYPE html>
<html lang="en">
   <head>
      <meta charset="utf-8" />
      <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no" />
      <meta name="description" content="" />
      <meta name="author" content="" />
      <title>Creative - Start Bootstrap Theme</title>
      <!-- Favicon-->
      <link rel="icon" type="image/x-icon" href="static/assets/img/favicon.ico" />
      <link href="static/css/styles.css" rel="stylesheet" />
   </head>
   <body id="page-top">
      <nav class="navbar navbar-expand-lg navbar-light fixed-top py-3" id="mainNav">
         <div class="container">
               <ul class="navbar-nav ml-auto my-2 my-lg-0">
                  <li class="nav-item"><a class="nav-link js-scroll-trigger"
href="home">Home</a></li>
                  <li class="nav-item"><a class="nav-link js-scroll-trigger"
href="DataOwnersUploadFiles">Upload Files</a></li>
                  <li class="nav-item"><a class="nav-link js-scroll-trigger"
href="DataOwnersViewFiles">View Files</a></li>
                  <li class="nav-item"><a class="nav-link js-scroll-trigger"
href="DataOwnersViewAllFiles">View All Files and Request dual access</a></li>
                  <li class="nav-item"><a class="nav-link js-scroll-trigger"
href="Viewdualaccessresponse">View Dual Access Response</a></li>

                  <li class="nav-item"><a class="nav-link js-scroll-trigger"
href="Viewdualaccessrequests">View Dual Access Requests</a></li>

                  <li class="nav-item"><a class="nav-link js-scroll-trigger" href="/csp">Logout</a></li>

               </ul>
            </div>
      </nav>
      <!-- Masthead-->
      <header class="masthead">
```

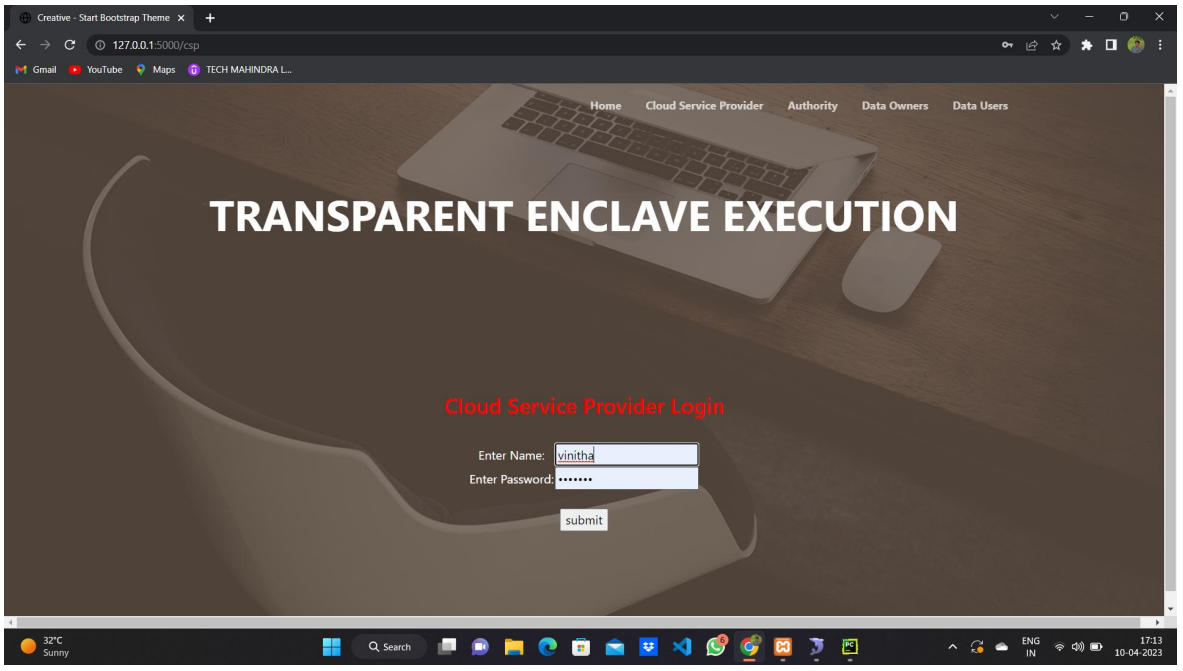
```
            <div class="row h-100 align-items-center justify-content-center text-center">
                <h1 class="text-uppercase text-white font-weight-bold">Transparent Enclave
Execution</h1>
                {% if msg=="sucess" %}
                    <h2 style="color:red">Login Successfully Completed, {{ name }}!</h2><br><br>
<!--                    <br><h3 style="color:pink"> Welcome {{ name }}!</h3>-->
                {% endif %}

                  <div class="col-lg-10 align-self-end"></div>
                <div class="col-lg-8 align-self-baseline"></div>
            </div>
        </header>
        <script src="static/js/scripts.js"></script>
    </body>
</html>
```

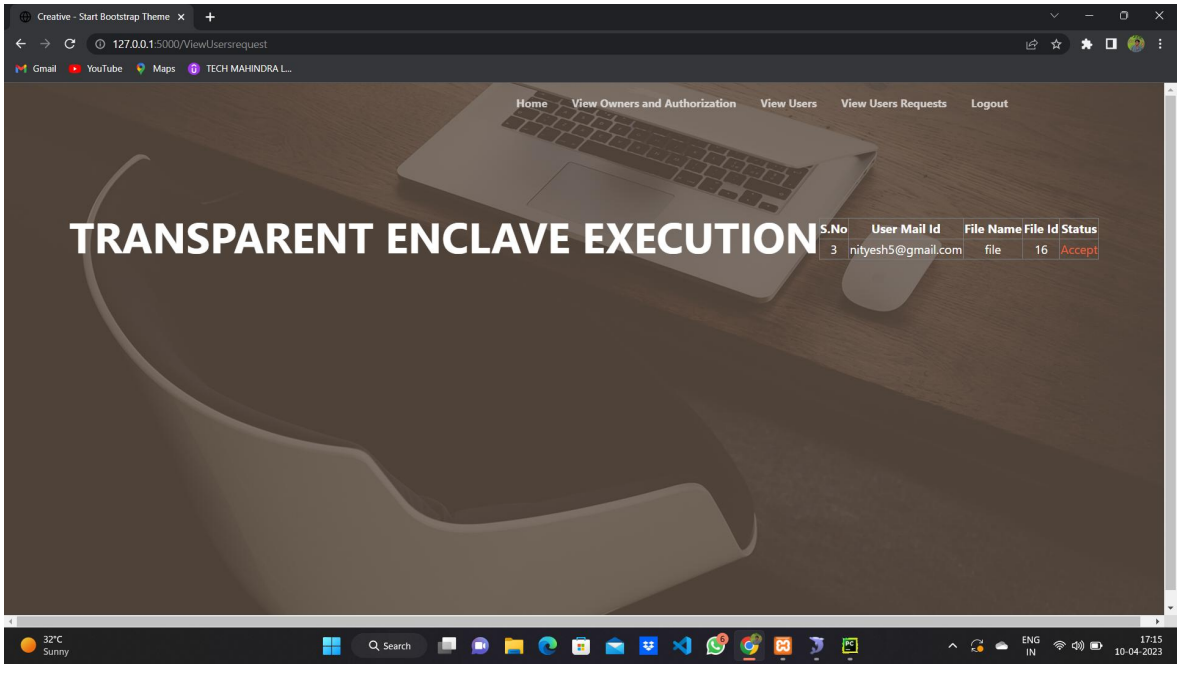
## SCREEN SHOTS

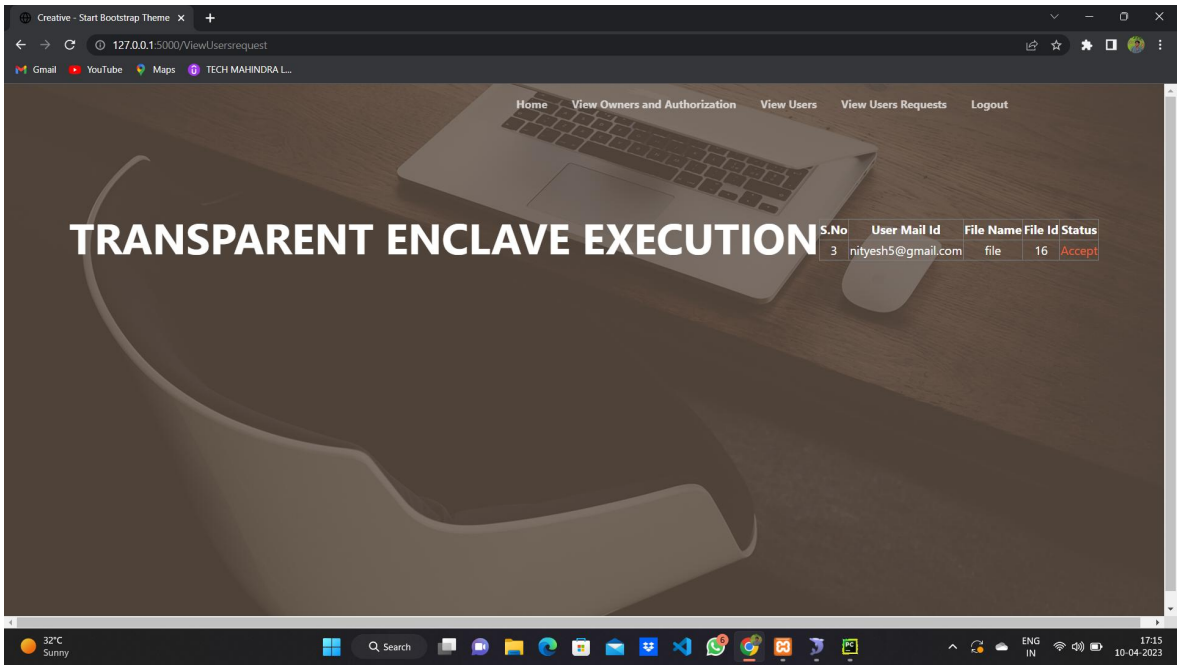


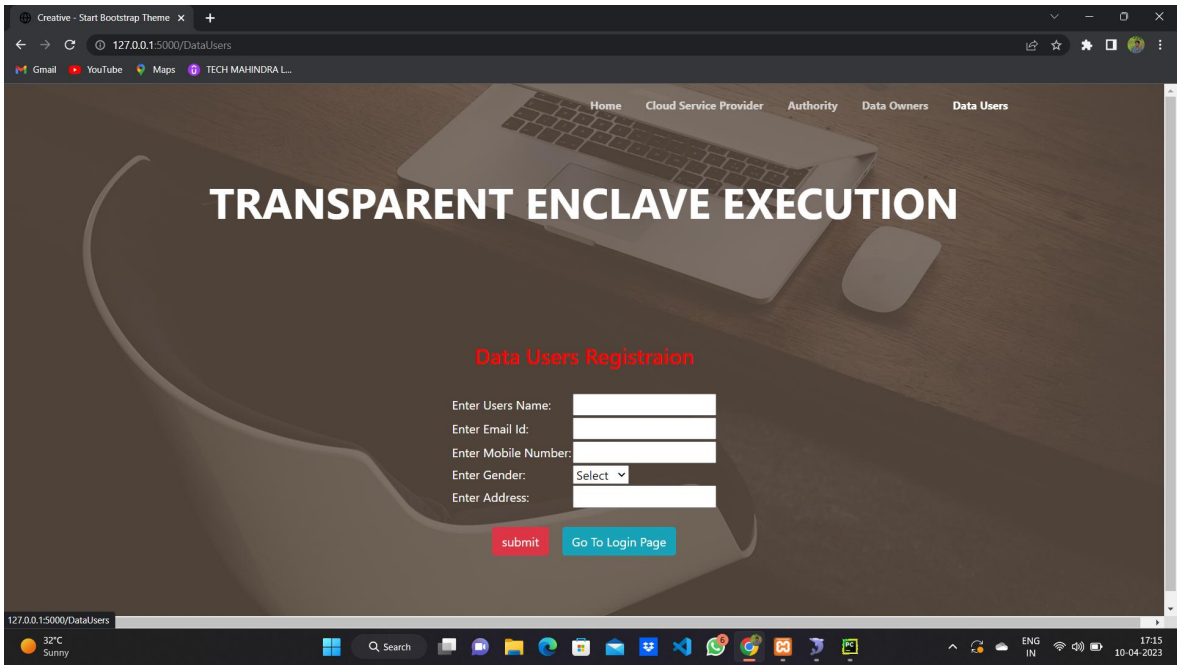

Creative - Start Bootstrap Theme
127.0.0.1:5000/Viewusersforauthorization
Gmail
YouTube
Maps
TECH MAHINDRA L...
Home
View Users requests and generating key
View Users for authorization
Logout
TRANSPARENT ENCLAVE EXECUTION
Slno Name Email Number Gender Address Otp
2 Nityesh nityesh5@gmail.com 9290242374 File Sastry road,up of firstlook menwear, palwancha 657070256
32°C
Sunny
Search
ENG
IN
17:12
10-04-2023

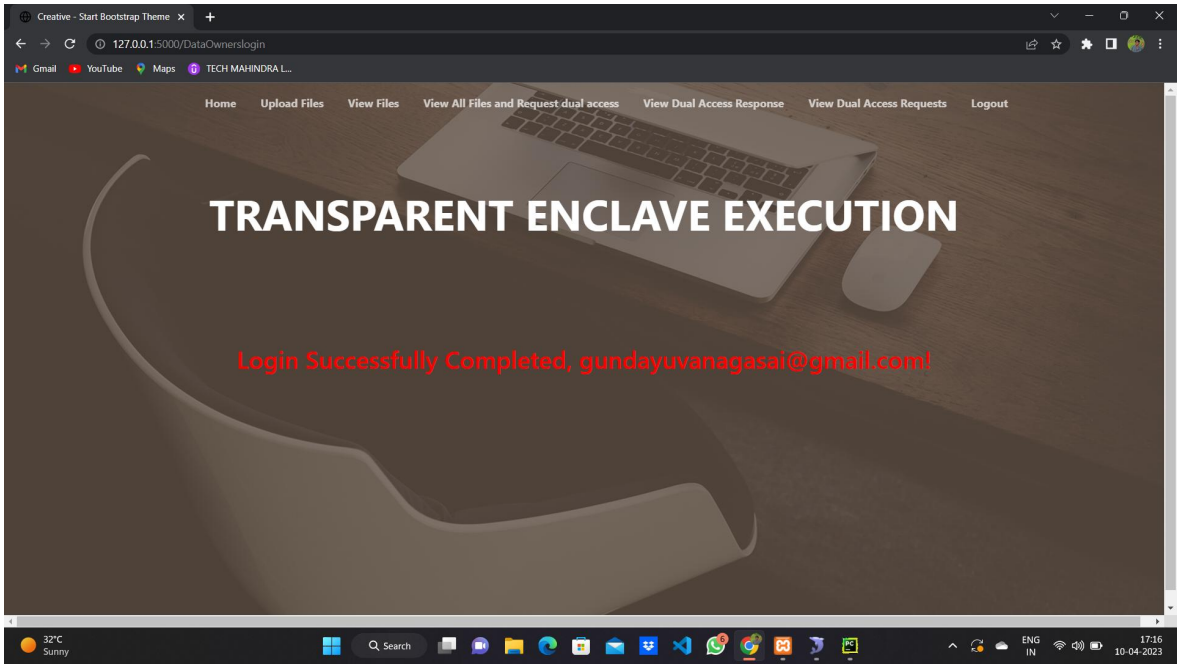

Creative - Start Bootstrap Theme
127.0.0.1:5000/AuthorityViewUsers
Gmail
YouTube
Maps
TECH MAHINDRA L...
Home
View Users requests and generating key
View Users for authorization
Logout
TRANSPARENT ENCLAVE EXECUTION
S.No User Mail Id File Name File Id Status Key Generation
2 nityesh5@gmail.com file 16 Accepted Generate Key
127.0.0.1:5000/AuthorityViewUsers
32°C
Sunny
Search
ENG
IN
17:13
10-04-2023

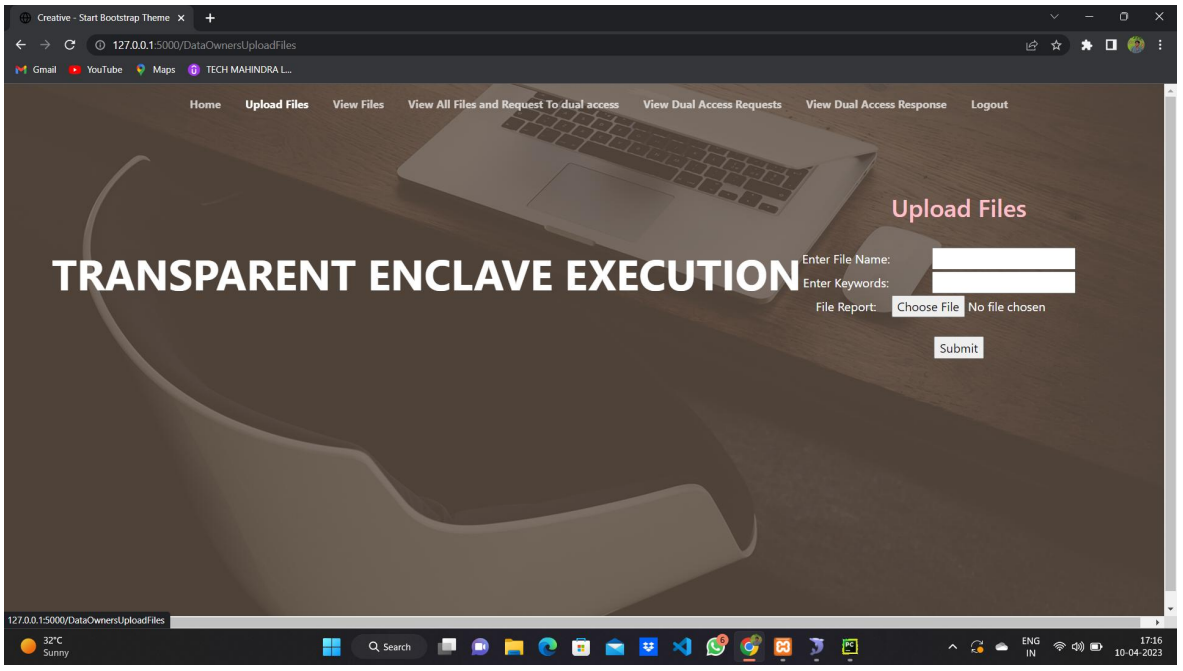

Creative - Start Bootstrap Theme
127.0.0.1:5000/csp
Gmail
YouTube
Maps
TECH MAHINDRA L...
Home
Cloud Service Provider
Authority
Data Owners
Data Users
TRANSPARENT ENCLAVE EXECUTION
Cloud Service Provider Login
Enter Name:
vinitha
Enter Password:
submit
32°C
Sunny
Search
ENG
IN
17:13
10-04-2023


Creative - Start Bootstrap Theme
127.0.0.1:5000/ViewUsersrequest
Gmail
YouTube
Maps
TECH MAHINDRA L...
Home
View Owners and Authorization
View Users
View Users Requests
Logout
TRANSPARENT ENCLAVE EXECUTION
S.No
User Mail Id
File Name
File Id
Status
3
nityesh5@gmail.com
file
16
Accept
32°C
Sunny
Search
ENG
IN
17:15
10-04-2023


Creative - Start Bootstrap Theme
127.0.0.1:5000/ViewUsersrequest
Gmail
YouTube
Maps
TECH MAHINDRA L...
Home
View Owners and Authorization
View Users
View Users Requests
Logout
TRANSPARENT ENCLAVE EXECUTION
S.No
User Mail Id
File Name
File Id
Status
3
nityesh5@gmail.com
file
16
Accept
32°C
Sunny
Search
ENG
IN
17:15
10-04-2023


Creative - Start Bootstrap Theme
127.0.0.1:5000/DataUsers
Gmail
YouTube
Maps
TECH MAHINDRA L...
Home
Cloud Service Provider
Authority
Data Owners
Data Users
TRANSPARENT ENCLAVE EXECUTION
Data Users Registraion
Enter Users Name:
Enter Email Id:
Enter Mobile Number:
Enter Gender:
Select
Enter Address:
submit
Go To Login Page
127.0.0.1:5000/DataUsers
32°C
Sunny
Search
ENG
IN
17:15
10-04-2023


Creative - Start Bootstrap Theme
127.0.0.1:5000/DataOwnerslogin
Gmail
YouTube
Maps
TECH MAHINDRA L...
Home
Upload Files
View Files
View All Files and Request dual access
View Dual Access Response
View Dual Access Requests
Logout
TRANSPARENT ENCLAVE EXECUTION
Login Successfully Completed, gundayuvanagasai@gmail.com!
32°C
Sunny
Search
ENG
IN
17:16
10-04-2023


Creative - Start Bootstrap Theme
127.0.0.1:5000/DataOwnersUploadFiles
Gmail
YouTube
Maps
TECH MAHINDRA L...
Home
Upload Files
View Files
View All Files and Request To dual access
View Dual Access Requests
View Dual Access Response
Logout
TRANSPARENT ENCLAVE EXECUTION
Upload Files
Enter File Name:
Enter Keywords:
File Report:
Choose File
No file chosen
Submit
127.0.0.1:5000/DataOwnersUploadFiles
32°C
Sunny
Search
ENG
IN
17:16
10-04-2023