MAHARAJA INSTITUTE OF TECHNOLOGY MYSORE

Belawadi, SrirangapatnaTq, Mandya-571477

DEPARTMENT OF CSE (Artificial Intelligence)

Assignment – Project Report

2025-26

Subject Name: Cloud Computing

Subject Code: M23BCS504

Semester: 5th sem

Submitted by:

Sl. No.	Student Name	USN.	CO's Mapping					Total	Scaled to
			CO1	CO2	CO3	CO4	CO5		
1	**Rithu Prakash R**	**4MH23CA045**							

Verified and Approved by:

Faculty Name: Prof. M J Yogesh

Signature:

Date:

Cloud Data Security Using Encryption Techniques

TABLE OF CONTENTS

1. INTRODUCTION

Cloud computing has become one of the most important technologies in modern information systems. It allows users to store, access, and manage data over the internet without depending on local storage devices. Services such as Google Drive, Dropbox, OneDrive, and Amazon S3 provide users with easy access to their data from anywhere in the world. While cloud computing offers flexibility, scalability, and cost efficiency, it also introduces serious security challenges, especially related to data privacy and confidentiality.

In traditional systems, data is stored locally on personal computers or organizational servers. These systems provide a certain level of control over data access and security. However, when data is stored on cloud servers, users lose direct control over their files. Cloud service providers manage the infrastructure, storage, and access mechanisms. This raises concerns about unauthorized access, data leakage, insider threats, and cyber-attacks.

One of the major security concerns in cloud computing is data confidentiality. Sensitive information such as personal documents, academic records, financial files, and business data must be protected from unauthorized users. Even though cloud providers implement security mechanisms, storing data in plaintext format can still pose risks. If an attacker gains access to the storage system, unencrypted data can be easily read and misused.

To address these issues, encryption is widely used as an effective security mechanism. Encryption converts readable data (plaintext) into an unreadable format (ciphertext) using cryptographic algorithms. Only authorized users with the correct decryption key can restore the original data. This ensures that even if encrypted data is accessed by an attacker, it remains useless without the key.

This project focuses on implementing a Cloud Encryption Mini Project using Python and the Fernet symmetric encryption algorithm. The main objective of the project is to encrypt files before uploading them to the cloud and decrypt them after downloading. By doing this, data confidentiality is maintained even if the cloud storage is compromised.

The project demonstrates a simple yet effective approach to securing cloud data using client-side encryption. The encryption and decryption processes are performed locally

on the user's system, ensuring that the cloud never stores the original unencrypted data.

2. LITERATURE SURVEY

With the rapid growth of cloud computing, researchers and organizations have proposed various methods to secure data stored in cloud environments. Early cloud security models relied heavily on trust between the user and the cloud service provider. However, this approach proved insufficient due to increasing cyber threats and data breaches.

Several studies emphasize the importance of **client-side encryption**, where data is encrypted before being uploaded to the cloud. According to research conducted on cloud security models, encrypting data at the client side significantly reduces the risk of data exposure, even if cloud servers are compromised. Symmetric encryption algorithms such as AES (Advanced Encryption Standard) are commonly used due to their efficiency and strong security guarantees.

Other research works explore hybrid encryption models combining symmetric and asymmetric cryptography. While these methods enhance security, they often increase computational complexity and implementation difficulty. For small-scale applications and academic projects, symmetric encryption provides a good balance between security and simplicity.

Fernet encryption, which is part of the Python cryptography library, is a widely accepted symmetric encryption method. It ensures data confidentiality, integrity, and authentication using AES encryption and HMAC verification. Several researchers highlight Fernet as a reliable option for file encryption applications due to its ease of use and strong cryptographic standards.

Existing cloud storage solutions typically rely on server-side encryption, where the cloud provider encrypts data after it is uploaded. While this approach provides protection against external attacks, it does not fully protect against insider threats or unauthorized access within the cloud provider's infrastructure.

This project adopts the concept of **end-to-end encryption**, ensuring that data is encrypted before entering the cloud and decrypted only after retrieval by the authorized user. The approach aligns with modern security practices recommended in cloud security literature.

3. EXISTING SYSTEM

In the existing cloud storage system, users upload files directly to cloud servers without performing encryption at the client side. Most cloud providers apply server-side encryption, which means that the encryption keys are managed by the service provider. Although this approach offers convenience, it introduces several security risks.

One major drawback of the existing system is the lack of full control over encryption keys. Since cloud providers manage the keys, users must trust the provider to handle their data securely. In cases of data breaches, insider attacks, or legal access requests, sensitive user data may be exposed.

Additionally, plaintext data transmission or weak encryption methods can lead to vulnerabilities during file uploads and downloads. Attackers may exploit network-level weaknesses to intercept data. Even if encryption is applied, improper key management can compromise security.

The existing system also lacks transparency, as users are not always aware of how their data is encrypted or who has access to the encryption keys. This raises concerns about privacy, especially for sensitive personal or academic data.

4. PROPOSED SYSTEM

The proposed system introduces a secure file storage approach using **client-side encryption**. In this system, files are encrypted locally using a secret key before being uploaded to cloud storage. The cloud server stores only encrypted data, ensuring that the original content remains confidential.

The encryption and decryption processes are handled entirely by the user's system. A symmetric encryption key is generated once and securely stored. This key is used for both encryption and decryption operations. Only users with access to the key can decrypt the encrypted files.

The proposed system ensures:

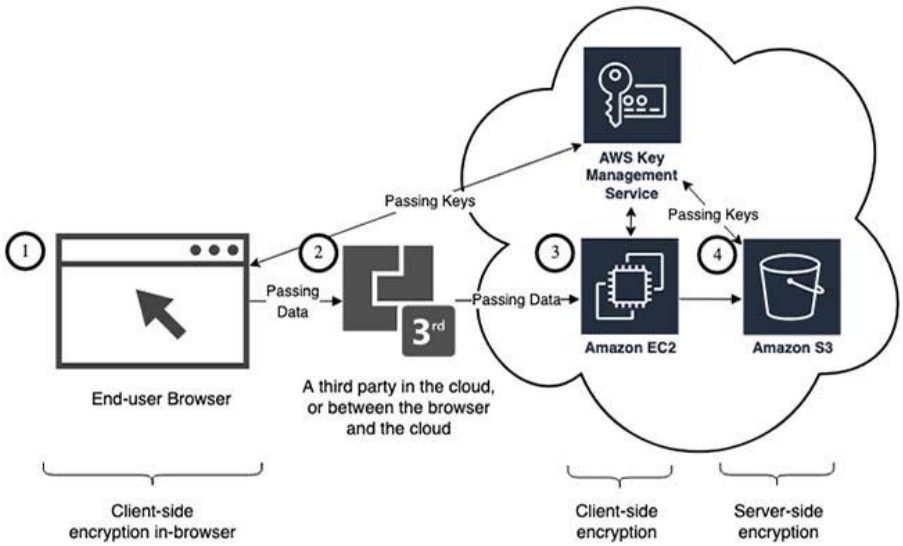
- Confidentiality of cloud-stored data
- Protection against unauthorized access
- Reduced trust dependency on cloud providers
- Secure file sharing through encrypted files

The system is simple, lightweight, and suitable for academic and small-scale applications. It demonstrates practical implementation of encryption concepts in cloud computing environments.

5. SYSTEM ARCHITECTURE

The system architecture consists of the following components:

1. **User Interface (CLI)**
   Provides a menu-driven interface for users to generate keys, encrypt files, and decrypt files.
2. **Key Management Module**
   Responsible for generating and loading encryption keys securely.
3. **Encryption Module**
   Encrypts files using the Fernet encryption algorithm.
4. **Cloud Storage**
   Stores encrypted files without access to plaintext data.
5. **Decryption Module**
   Decrypts encrypted files after downloading from the cloud.

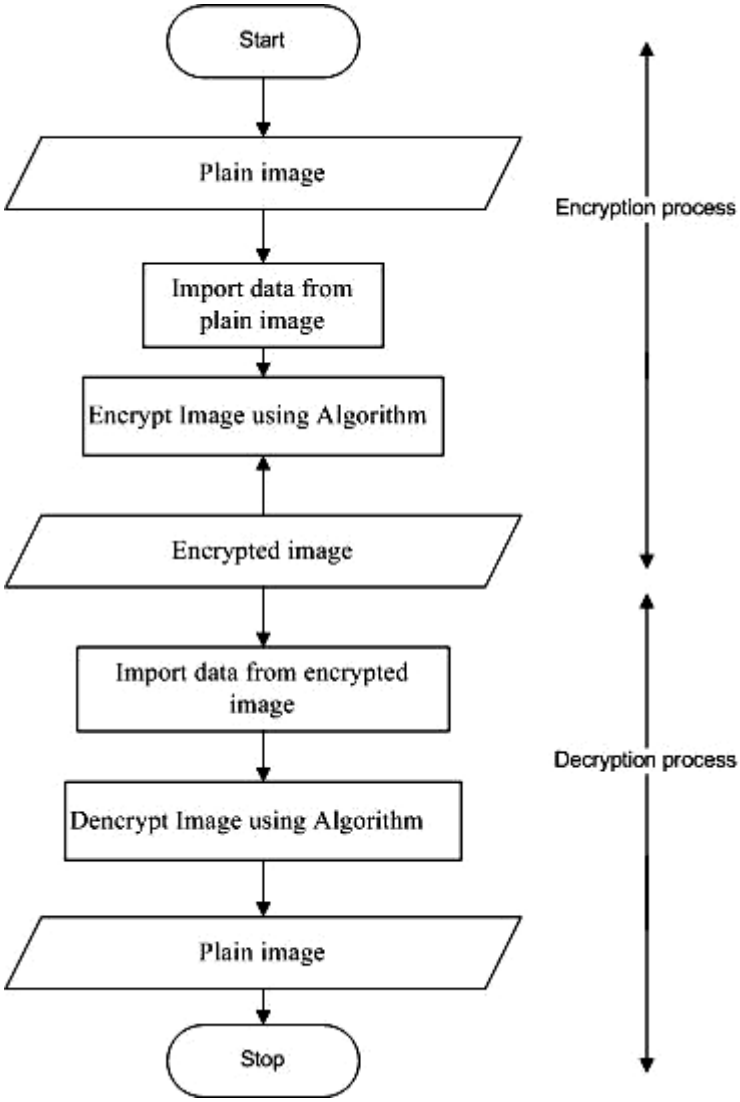


METHODOLOGY

The methodology followed in this project involves a step-by-step approach to ensure secure cloud data storage. Initially, a secret encryption key is generated using the Fernet algorithm. This key is stored securely and reused for all encryption and decryption operations.

Once the key is generated, the user selects a file for encryption. The file is read in binary mode, encrypted using the secret key, and saved with a .enc extension. The encrypted file is then uploaded to cloud storage.

When the user wants to retrieve the file, the encrypted file is downloaded from the cloud. The decryption process uses the same secret key to restore the original file content.



IMPLEMENTATION DETAILS

The project is implemented using Python programming language. The cryptography library is used to perform encryption and decryption. The Fernet module ensures secure symmetric encryption with built-in integrity verification.

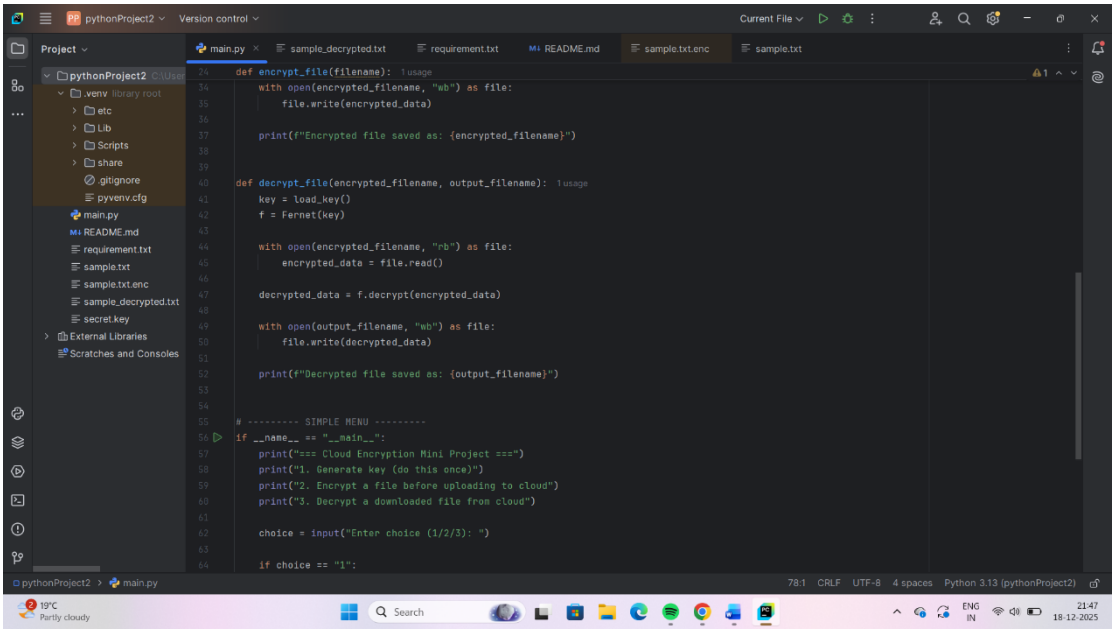
The implementation includes functions for key generation, encryption, and decryption. File handling operations are performed in binary mode to support all file types, including text files, PDFs, and images.

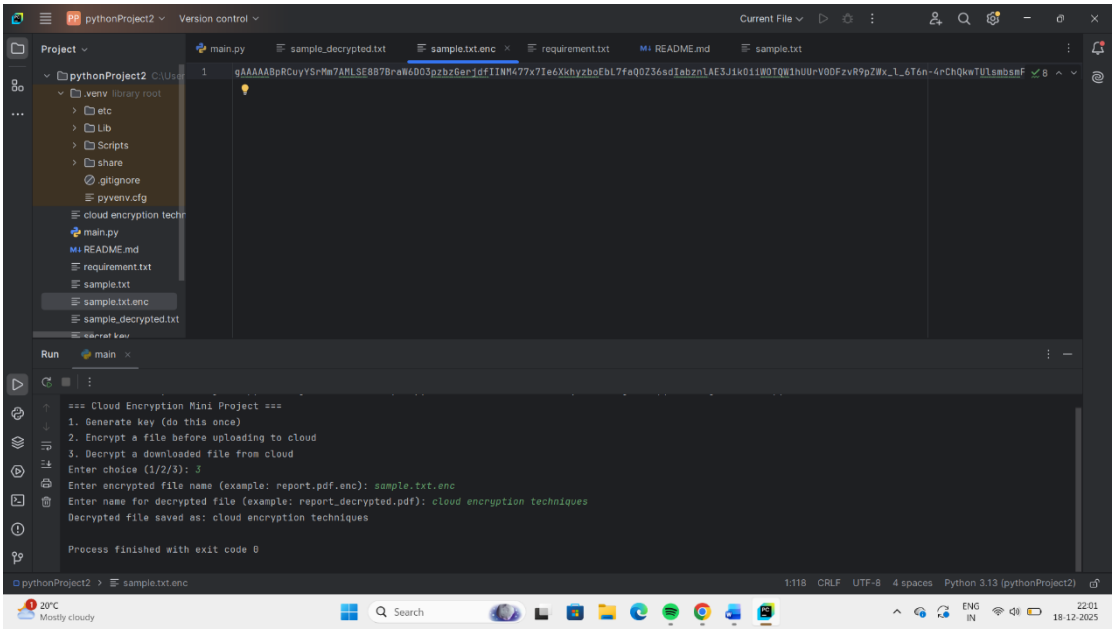
The menu-driven interface allows users to interact with the system easily. The simplicity of the implementation makes it suitable for beginners while still demonstrating strong security concepts.

RESULTS AND OUTPUT

The system successfully encrypts files before uploading them to cloud storage. The encrypted files appear unreadable and cannot be opened without decryption. After downloading and decrypting, the original file is restored without any data loss.

Screenshots of encrypted files, cloud uploads, and decrypted outputs confirm the effectiveness of the system.





ADVANTAGES

- Ensures data confidentiality
- Protects against cloud data breaches
- Simple and efficient implementation
- Supports multiple file formats
- Lightweight and fast encryption

LIMITATIONS

- Key loss results in permanent data loss
- No automated key backup
- Manual cloud upload and download

FUTURE ENHANCEMENTS

Future enhancements may include:

- GUI-based interface
- Cloud API integration
- Multi-user key management
- Hybrid encryption models

CONCLUSION

This project demonstrates the importance of encryption in cloud computing. By encrypting files before uploading them to the cloud, data confidentiality is ensured even in untrusted environments. The use of Fernet encryption provides strong security with minimal complexity, making this project suitable for academic purposes and real-world applications.