

**KIGALI INDEPENDENT UNIVERSITY ULK
SCHOOL OF SCIENCE AND TECHNOLOGY
DEPARTEMENT OF COMPUTER SCIENCE
P.O BOX: 2280 KIGALI**

**SECURE FILE STORAGE ON THE CLOUD USING
ASYMMETRIC CRYPTOGRAPHY
CASE STUDY: Small and Medium Enterprises in DRC**

Done by: KEBO WANY Gustave

Roll Number: 202111063

Supervised by: Mr. BYIRINGIRO Eric

**Dissertation Submitted in Partial Fulfilment of the Requirements for the Award of
Bachelor's Degree in Computer Science**

Kigali, September 2024

DECLARATION

This dissertation titled “secure file storage on the cloud using asymmetric cryptography, a case study of small and medium enterprises” is my original work, it has never been submitted before for any other degree award to any other University or for any other institution or publication.

KEBO WANY Gustave

Signature:

Date: / / 2024

APPROVAL

This is to certify that the dissertation titled “**secure file storage on the cloud using asymmetric cryptography**” has been submitted by **KEBO WANY Gustave**, Roll number: 202111063 and done under my supervision for examination with my approval.

Supervisor: Mr. BYIRINGIRO Eric

Signature:

Date: / / 2024

DEDICATION

To the Almighty God,

To my beautiful mother BILINGI BITISHO Marie-Louise,

To my Dad my mentor WANZA BILINGI,

To my beautiful fiancée Linda Lucie WABENGA,

To everyone who prayed for me and supported me;

I dedicate this work to you.

ACKNOWLEDGEMENT

I sincerely thank the following people for their help and support, without which this research would not have been possible. My deepest gratitude goes out to **Prof. Dr. RWIGAMBA BALINDA**, the distinguished President and Founder of our acclaimed University, for creating this learning environment that has fostered my intellectual development.

I owe a special debt of gratitude to **Mr BYIRINGIRO Eric**, my supervisor, whose advice, perceptive observations, and tolerance got me through every stage of my project.

This research came to be in large part because of his guidance and dedication to greatness. I am extremely grateful to my parents for their consistent financial and emotional support, which has enabled me to pursue my academic goals. Their support served as the cornerstone upon which this study was constructed.

In addition, I would want to thank my brothers, sisters, and extended family for their encouragement and belief in my skills, which gave me the determination to keep going.

Thank you to those who are close to me for your support, insightful conversations, and constructive criticism; without them, I could not have finished this work successfully. Lastly, I would like to express my gratitude to everyone who helped make this research a reality, whether directly or indirectly. Your assistance is greatly appreciated, and I will always be appreciative.

KEBO WANY Gustave

TABLE OF CONTENTS

DECLARATION	i
APPROVAL	ii
DEDICATION	iii
ACKNOWLEDGEMENT	iv
LIST OF TABLES	d
LIST OF FIGURES	e
ABBREVIATIONS AND ACRONYMS	g
ABSTRACT.....	i
CHAPTER ONE: GENERAL INTRODUCTION	1
1.1. Introduction.....	1
1.2. Background of the study	1
1.3. Problem statement.....	2
1.4. Research Objectives.....	3
1.4.1. General objective	3
1.4.2. Specifics objectives.....	3
1.4.3. Research Questions	3
1.5. Scope of the study	4
1.5.1. Scope of the project	4
1.5.2. Content Scope	4
1.5.4. Time Scope	4
1.6. Significance of the study.....	5
1.6.1. Personal Interest.....	5
1.7. Project Methodology.....	5
1.8. Organisational of the projet.....	6
1.9. Limitations	7
CHAPTER TWO: LITERATURE REVIEW	8

2.1. Introduction.....	8
2.2. Conceptual Review	8
2.2.1. Cloud	8
2.2.2. Storage	8
2.2.3. Asymmetric	8
2.2.4. Cryptography	9
2.3. Theoretical Review	9
2.4. Review of Related Literature	10
2.5. Conceptual Frame work	12
2.5.1. Relationships among Concepts	13
2.5.2. Conclusion	15
CHAPTER THREE: SYSTEM ANALYSIS AND DESIGN	16
3.1. Introduction.....	16
3.2. Analysis of the current system	16
3.2.1. Problem of the current system	16
3.2.1.1. Security Vulnerabilities	16
3.3. Analysis of the new system.....	17
3.3.1. Introduction	17
3.3.2. System requirements.....	17
3.3.3. Functional Diagram	22
3.3.4. Methodological approach	24
CHAPTER FOUR: SYSTEM IMPLEMENTATION.....	31
4.1. Implementing and coding	31
4.1.2. Description of Implementation Tools and Technology	31
4.1.3. Screenshots and source codes	32
4.2. Testing.....	35
4.2.1. Introduction	35
4.2.2. Unit Testing Outputs	36

4.2.3. Validation Testing Outputs.....	36
4.2.4. Integration Testing Outputs	36
4.2.5. Functional and System Testing.....	36
4.2.6. Acceptance Testing Report.....	36
CONCLUSION AND RECOMMENDATIONS.....	39
CONCLUSION	39
RECOMMENDATIONS.....	40
SUMMARY.....	41
REFERENCES	42
APPENDICES	46

LIST OF TABLES

Table 1. User authentication mechanisms comparison table	18
Table 2. Key management techniques table	19
Table 3: system requirement table	20
Table 4: Performance Testing Results Table	37
Table 5: Time Frame.....	46

LIST OF FIGURES

Figure 1: Asymmetric encryption Algorithm.....	9
Figure 2: Dropbox security Architecture	11
Figure 3: Conceptual framework	14
Figure 4: Dropbox Logo.....	21
Figure 5: Functional Diagram	22
Figure 6: Flowchart diagram.....	23
Figure 7: Agile Model iterations.....	26
Figure 8: Use case Diagram.....	28
Figure 9: Class diagram.....	29
Figure 10: sequence Diagram.....	29
Figure 11: Activity diagram.....	30
Figure 21: Login page interface.....	32
Figure 13: Register interface.....	32
Figure 14: Main page.....	33
Figure 15: Report menu.....	33
Figure 16: Settings Option	34
Figure 17: Edit profile	34
Figure 18: Dropbox console for application creation.....	35
Figure 19: Uploading file and encrypted	37
Figure 20: Decrypted file and download	37
Figure 21: Running the flask app from VScode.....	46
Figure 22: Private Key generated	47
Figure 23: Public key generated	47

ABBREVIATIONS AND ACRONYMS

- 2FA: Two-Factor Authentication
- AES: Advanced Encryption Standard
- APIs: Application Programming Interfaces
- AWS: Amazon web Service
- CPU: Central Process Unity
- CRUD: Create, Read, Update, Delete
- CS: Computer Science
- CSPs: Cloud Service Providers
- CSS: Cascading Style Sheets
- DES: Data Encryption Standard
- DFD: Data Flow Diagram
- DRC: Democratic Republic of the Congo
- DSA: Digital Signature Algorithm
- ECC: Elliptic Curve Cryptography
- ERD: Entity Relationship Diagram
- GB: Giga Byte
- GCP: Google Cloud Platform
- GDPR: General Data Protection Regulation
- HIPAA: Health Insurance Portability and Accountability Act
- HTML: Hypertext Mark-up Language
- HTTPS: Hypertext Transfer Protocol Secure
- IDE: Integrated Development Environment
- IT: Information Technology

JS: JavaScript

MFA: Multi-Factor Authentication

MVC: Model-View-Controller

MySQL: My Structured Query Language

OOSADM: Object-Oriented Systems Analysis and Design Methodology

PHP: Hypertext Pre-processor

RAM: Random Access Memory

RBAC: Role-Based Access Control

RSA: Rivest-Shamir-Adleman

SMEs: Small and Medium Enterprises

SQL: Structured Query Language

SSADM: Structured Systems Analysis and Design Methodology

SSD: Solid-State-Drive

SSH: Secure Shell

UAT: User Acceptance Test

UI: User Interface

ULK: Kigali Independent University

UX: User Experience

ABSTRACT

In order to create and evaluate an innovative security framework that uses asymmetric cryptography to improve security in cloud-based file storage systems, this exploratory study uses a mixed-methods approach. The main objective was to create a customized security framework that works well with current cloud infrastructures and addresses security issues without negatively impacting system performance.

The creation of the security architecture, its deployment across several cloud storage platforms, and performance testing in actual use cases were among the goals of the research. The process comprised a conceptual design, which was then put into practice, rigorously tested, and included user test scenarios, expert evaluations, and simulated attacks. Important performance indicators were assessed, including access times, the speed at which data is encrypted and decrypted, and the susceptibility to security lapses.

According to the findings, integrating asymmetric cryptography greatly increased security by lowering the possibility of unwanted access. Crucially, retrieval times were not significantly impacted by this improvement, indicating that it had no effect on system performance.

According to the study's findings, asymmetric cryptography can be a useful way to improve cloud storage security without compromising user experience. It highlights the necessity of regular framework updates to keep up with changing security risks and technical advancements, and it advises cloud service providers to implement such cryptographic procedures. It is also advised that the framework be scalable across many platforms and data volumes to guarantee its applicability and efficacy in the ever-changing world of cloud computing.

Key words: security, storage, encryption and cloud.

CHAPTER ONE: GENERAL INTRODUCTION

1.1. Introduction

Ensuring data security and privacy is crucial since cloud services are becoming indispensable for exchanging and storing sensitive information. The goal of this project is to create an asymmetric cryptography-based secure cloud-based file storage system in which files are encrypted with a public key and decrypted with a private key. The technology provides strong protection by encrypting files before uploading and limiting access to decryption to authorized users only (Stallings W. , *Cryptography and Network Security: Principles and Practice*, 2020).

While security, performance, scalability, and data protection compliance are non-functional criteria, the project solves major functional requirements like safe authentication, simple file management, and efficient key management. While HTML, CSS, and JavaScript are used on the front end, MySQL and PHP are used on the back end for secure user authentication and key management. Python is used for both encryption and decoding. The project is driven by agile methodology, which allows for iterative development and ongoing input and improvement. This strategy makes sure the system lives up to user expectations and changes with the demands of the users.

An examination of the literature reveals weaknesses in the cloud security solutions now in use, which this project aims to fill by fusing robust encryption techniques with an intuitive user interface. The ultimate objective is to develop a scalable, secure, and effective cloud storage solution that offers consumers improved data security (Schwaber, 2021).

1.2. Background of the study

The ability to store, manipulate, and retrieve data with previously unheard-of ease and scalability has completely transformed thanks to the cloud. These days, cloud computing services are an essential component of contemporary IT infrastructures, enabling businesses and private users to store enormous amounts of data without having to purchase actual hard drives. According to a report by the International Data Corporation (IDC), the global spending on public cloud services and infrastructure is expected to reach \$500 billion by 2023 (Corporation(IDC), 2020). However, these advantages have also given birth to a major security and privacy challenge as sensitive information migrate to the cloud. Probably the most critical issue with cloud storage us how to protect data against unauthorized access, breaches, and other

types of cyber threats. Traditional encryption methods, though effective up to a multitenant cloud environment where data can be susceptible during both transmission and at rest. Asymmetric cryptography, more popularly referred to as public-key cryptography, seems to be a strong solution to these problems. Unlike symmetric cryptography, relying on using one key for encryption and for decryption, asymmetric cryptography uses a pair. This approach enhances security by ensuring that even if the public key is compromised, the private key remains secure, thus protecting the data (Lindell, 2020).

A software development technique that can adjust to the ever-changing requirements and security concerns is essential for the creation of safe cloud storage systems. Agile approaches iterative, incremental approach and emphasis on flexibility and cooperation have led to their widespread adoption in the software development community. According to (Martin, 2022), the Agile methodology places a strong emphasis on the continuous delivery of functional software, enabling regular testing and the inclusion of the security features.

1.3. Problem statement

Clouds have been demonstrating the trend of increasing dependency on cloud-based storage systems. Therefore, there have arisen a number of important questions concerning the security of the data stored in this ever-advancing era of cloud computing. The system has very good scalability and access capability, but sensitive critical information is vulnerable because of several vulnerabilities in traditional methods of encryption. Current security measures against sophisticated attacks can't tend to do care, so speak, thus leading to breaches and violations of privacy and loss of data. What is quite urgently required are strong security solutions specially adapted to cloud environments in order to try and regain confidence for organisation and people that their data are well protected against emerging cyber threats. The present investigation will deal with the critical gap to assess asymmetric cryptography for practical implementation in enhancing the security architecture of cloud-based file storage systems. In this regard, the methodology will be done based on the reliance on public cryptographic techniques which have advantages in key management and encryption in devising a single framework that perhaps can mitigate the risk of unauthorized access and breach in cloud storage. Empirical data is analysed and critically evaluated in an attempt to extract pragmatic insights and recommendations that should help improving the security of cloud storage systems, where one can have greater confidence in the integrity and confidentiality of stored data.

1.4. Research Objectives

1.4.1. General objective

The main objective of this research project is to apply asymmetric cryptography in the process of uploading and downloading files to and from cloud storage, through a framework that we will develop and ensure file security.

1.4.2. Specifics objectives

- i. Use role-based access control (RBAC) to manage user permissions and establish user authentication and authorization to create a capable security structure where access to the system can only be granted to authorize users.
- ii. Allows for the safe upload and download of files that are encrypted on the client side before being sent to the cloud. The files are then decrypted on the client side after being downloaded, ensuring data security even while they are being transmitted.
- iii. Secure the files using asymmetric cryptography, where the key pairs (public and private key) will intervene. Users must be able to upload files with encryption permissions, and they must also be able to get files that have been decrypted.
- iv. Create and implement an intuitive file management interface with features for organizing, searching, downloading, and uploading files. Make sure that both technical and non-technical users can easily understand and utilize the user interface.
- v. Optimize the system to rapidly manage massive file uploads and downloads, reducing latency and guaranteeing seamless encryption and decryption procedures without experiencing any performance snags.
- vi. Ensure that the system supports standard APIs for third-party service and application integration to provide easy connection with other systems and services.

1.4.3. Research Questions

- i. How can the security of files stored on cloud servers be successfully guaranteed by the use of asymmetric cryptography?
- ii. What are the best practices for controlling public and private keys in a cloud-based system to improve security and provide users more authority?

- iii. What effects does asymmetric cryptography's encryption and decryption of huge data have on system performance, and how may this be optimized?
- iv. How can asymmetric encryption be combined with role-based access control (RBAC) and user authentication to stop illegal access to files stored in the cloud?
- v. When using asymmetric encryption for cloud storage, what scalability and reliability issues come up, and how may these be resolved to keep the system operating at peak efficiency?
- vi. How can standard APIs be used to integrate third - party services with the cloud storage system to facilitate interoperability with other applications?

1.5. Scope of the study

1.5.1. Scope of the project

This study's time period, geographic reach, theoretical reach, and content reach all contribute to its stated scope, which guarantees a thorough and targeted examination of the security of asymmetric cryptography-based cloud storage systems.

1.5.2. Content Scope

The features and functionalities that the system will implement are covered by the content scope. These comprise user permission and authentication, file management, key management, encryption and decryption, audit and logging, user interface, security and compliance, scalability and performance, redundancy and reliability, usability and maintainability, and interoperability.

1.5.3. Geographical Scope

This study project's geographic scope is mostly concentrated on implementation within the Republic of Rwanda. On the other hand, the system's scalability allows for future development into other nations and areas. A particular Kigali organization or group of organizations will serve as the testing ground for the initial deployment. To make sure local laws are followed and to get preliminary user input. The method can be modified for usage in different regions based on the results, taking into account local data privacy laws and regulations

1.5.4. Time Scope

This research project's time frame is set from January 2024 to September 2024. The secure file storage system's planning, development, testing, and deployment processes are all included in this time frame.

1.6. Significance of the study

The significance of this research contribute to personal interest, public interest, and academic interest.

1.6.1. Personal Interest

The initiative offers a strong way for individual users to secure their personal data that is kept on cloud servers. Strong data protection techniques are essential given the growing reliance on digital storage for sensitive information, including financial records, private messages, and personal papers. By ensuring that files are encrypted before upload and that only authorized users may decrypt them, this initiative guards against potential breaches and unauthorized access to sensitive data

1.6.2. Institutional Interests

This study furthers the academic domains of cryptography and data security which of course are part of computer science field by investigating the application of asymmetric encryption in cloud storage systems. It offers perceptions into the difficulties and remedies related to cloud data security, which can be used as a guide for further studies and innovations. Additionally, the project enhances our understanding of data protection rules compliance, secure key management procedures, and secure application user interface design

1.6.3. Public Interests

In an era where data breaches and cyberattacks are becoming more common, the initiative addresses broader issues about data security and privacy from the standpoint of the public interest. The solution reduces risks related to cloud storage by incorporating secure user authentication, authorization, and key management. This is especially crucial for governmental agencies, healthcare organizations, and educational institutions, among other public bodies and organizations that manage substantial amounts of sensitive data. The project's emphasis on compliance with data protection regulations like GDPR and HIPAA further ensures that it meets the highest standards of data security and privacy (Voigt, 2017).

1.7. Project Methodology

The methodology for this project describes the methodical steps involved in creating a secure file storage system based on asymmetric cryptography. These steps include system design, implementation, testing, and evaluation. In order to uncover gaps in the current solutions, an exploratory research phase is conducted to study existing cryptographic protocols and file storage systems. This phase draws on foundational works such as Diffie and Hellman's 1976 "New Directions in Cryptography". The next step is a descriptive phase that uses best practices

for software development and cryptographic security to identify the functional and non-functional needs of the system the technique guarantees a methodical and empirically-based approach to accomplish the project's goals.

1.8. Organisational of the projet

1.8.1. Chapter one: General Introduction

The goal of this project is to create a secure cloud-based file storage system that uses asymmetric cryptography, which encrypts data using a public key and decrypts it using a private key to provide strong security. Along with non-functional elements like security, performance, and compliance, it solves critical functional requirements like safe authentication, effective file management, and efficient key management. Python is used for encryption and decryption, and the system combines front-end technology (HTML, CSS, and JavaScript) with back-end tools (PHP, MySQL). The project's agile methodology guarantees iterative development and user-needs adaption. It aims to solve security vulnerabilities in current cloud security solutions by fusing robust encryption with an easy-to-use interface to improve data security.

1.8.2. Chapter two: Literature review

With its scalable and affordable solutions, cloud computing has completely changed the way that data is stored and managed. However, it also brings up serious security issues, especially with regard to data integrity and confidentiality. Secure data encryption and decryption are made possible by asymmetric cryptography, also known as public-key cryptography, which provides a workable solution. In order to solve these security issues, this literature review examines important ideas, describes how asymmetric cryptography-based cloud file storage works, and assesses pertinent research.

1.8.3. Chapter three: System Analysis and Design

A comprehensive system analysis and design that prioritizes user needs above data security and efficiency are necessary to develop a safe cloud file storage system. In this phase, the system is evaluated, possible problems are identified, and solutions that support organizational and user objectives are designed. With a thorough investigation and design of the suggested solution, the research attempts to develop a system that protects data kept on cloud platforms.

1.8.4. Chapter four: System Implementation

System implementation is the process of using code to transform an idea into a working system. The procedures used to put the cloud-based file management system into place are described in this part, with an emphasis on user authentication and safe file transfer via asymmetric

cryptography. In order to provide the required functionality, the system integrates both frontend and backend components, making use of a variety of tools and technologies.

1.8.5. Conclusion and Recommendations

The goal of the research project was to use asymmetric cryptography to create a secure cloud storage system that would address the usability, security, and performance problems with current options. The system combined key management, RSA encryption, user authentication, and compliance with privacy rules, using agile methodology and Object-Oriented Systems Analysis and Design (OOSADM) to construct a scalable and dependable design. The system's efficiency and functioning were confirmed by extensive testing, which also revealed the need to integrate machine learning, improve usability, and fortify encryption. Future studies will concentrate on enhancing mobile usability, encryption techniques, and regulatory compliance.

1.9. Limitations

The study's limitations cover a wide range of variables that could influence or limit the research methodology and data analysis. These restrictions are unavoidable during the research process and are not within the researcher's power.

CHAPTER TWO: LITERATURE REVIEW

2.1. Introduction

Cloud computing, with its scalable, flexible, and affordable solutions, has completely changed the way businesses handle and keep data. But switching to cloud storage raises serious security issues, especially with regard to data integrity and confidentiality. These worries are effectively addressed by asymmetric cryptography, commonly referred to as public-key cryptography, which permits safe data encryption and decryption. This review of the literature looks at essential terminology, investigates the ideas and workings of asymmetric cryptography-based cloud file storage, and evaluates relevant publications.

2.2. Conceptual Review

Definition of key concepts:

2.2.1. Cloud

Within the field of Information Technology (IT), "cloud" refers to cloud computing, which is the internet-based delivery of computing services, such as servers, storage, databases, networking, software, and analytics, and is sometimes called "the cloud." These services, which are accessible from any location with an internet connection, let users access and store data and apps on remote servers as opposed to local devices. (Thomas Erl, 2023)

2.2.2. Storage

The technologies and systems used to handle, retrieve, and store digital data are referred to as storage in the Information technologies (IT) domain. In computer systems, storage is essential because it allows data to be retained for a variety of lengths of time by applications, users, and organizations. Storage is subdivided into three types where we have, primary storage (volatile storage), secondary storage (non-volatile) and tertiary storage (offline storage)

Cloud storage offers great scalability, flexibility, and accessibility by enabling users to store and retrieve data from a distant system. Any internet-connected device can access data that is kept on dispersed servers. The ability to store files and data on remote servers run by cloud service providers (CSPs) like Microsoft Azure, Google Cloud Platform (GCP), and Amazon Web Services (AWS) is a fundamental feature of cloud computing (Poulton, 2021).

2.2.3. Asymmetric

Asymmetric generally refers to public-key cryptography or asymmetric cryptography in information technology. Two keys are used in this kind of encryption: a public key and a private key. As they serve distinct purposes, the mathematically related keys are "asymmetric." This

technique is frequently used to secure communications, particularly when it comes to digital signatures, email encryption, and cloud storage (Mao, 2022).

2.2.4. Cryptography

The science and art of information security known as cryptography involves encrypting data to make it unreadable and preventing unauthorized access. It uses mathematical methods to secure data integrity, confidentiality, and authenticity in communication. Numerous industries, including financial transactions, data storage, and online communication, depend on cryptography. Different Cryptography Types include AES and DES which are examples of symmetric cryptography, in which the same key is used for both encryption and decryption. Asymmetric cryptography, often known as public key cryptography, uses two distinct but linked keys a public key for encryption and a private key for decryption for example, RSA and ECC. Hashing Operations is one-way functions (e.g., SHA-256, MD5) that transform data into a fixed-size hash value are frequently utilized for data integrity (Stallings W. , Cryptography and Network Security: Principles and Practice, 2020).

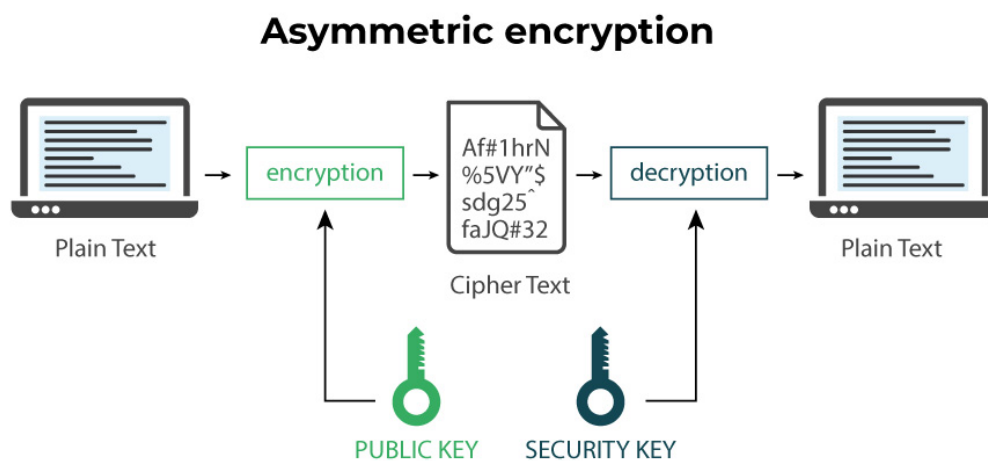


Figure 1: Asymmetric encryption algorithm

2.3. Theoretical Review

This study brings together several key theories in cryptography, access security, and cloud security. Theoretical models such as public key cryptography (RSA), role-based access control, and client - side encryption provide the fundamental concepts that support the project goals. In addition, the literature review highlights existing gaps in the current knowledge framework that

this study seeks to fill, particularly the implementation of user - friendly and secure file storage solutions for small and medium - sized enterprises. The research questions and methods of the project are based on these theoretical models, ensuring that the system meets modern security standards and is practical for real-world use (Tim Mather, 2020).

2.4. Review of Related Literature

The evaluation of relevant literature highlights gaps in coverage, context, chronology, and methodology by concentrating on empirical studies of secure file storage utilizing asymmetric cryptography. User authentication, file encryption, key management, and cloud security are the main topics of interest. In order to create a reliable and secure file storage system, these areas are essential (Kaur, 2021).

a. User Authentication

The foundation of any safe system is user authentication. Numerous authentication strategies have been thoroughly studied in studies by Bonneau and all. These mechanisms range from conventional password-based systems to biometric and multi-factor authentication. In 2012, Bonneau and Stajano emphasized the shortcomings of password-based authentication and the demand for more reliable, approachable substitutes investigated the efficacy of multi-factor authentication (MFA) in augmenting security, observing obstacles in user acceptance because to usability concerns. Notwithstanding these developments, there is still a problem with smoothly incorporating MFA into cloud-based file storage systems, especially when it comes to striking a balance between security and usability (Al Zahrani, 2021).

b. File Encryption

Protecting the confidentiality and integrity of data requires the encryption of files. Developed by Rivest, Shamir, and Adelman, the RSA algorithm is a fundamental part of asymmetric cryptography. Recent research by Sun, Chang, and GAO (Sun, Chang, & Gao, 2018) and Kumar and Mittal (Kumar & Mittal, 2020) has examined encryption algorithm upgrades as a means of improving security and performance. To optimize effectiveness and security, GAO, Sun, and Chang proposed a hybrid encryption scheme that combines AES and RSA. However, there are challenges in implementing such hybrid models in real-world cloud systems, particularly concerning key management and performance optimization. Kumar and Mittal stressed the need for efficient encryption methods that don't compromise system performance, especially for large file transfers in cloud storage (GAO, 2020).

c. Key Management

Cryptographic systems must be kept secure through proper key management. Li and al. (Li, Chen, & Lee, 2019) and Housley (Housley, 2018) have both studied the development of secure key management strategies. Housley highlighted the role of hardware security modules (HSMs) in his explanation of the importance of safe key distribution, generation, and storage. Li and colleagues proposed a decentralized key management system using block chain technology to increase security and transparency. Despite these advancements, there are still few comprehensive solutions that ensure cloud-based storage systems are scalable and easy to use by incorporating these crucial management procedures (Hassan, 2023).

d. Cloud Security

The area of cloud security is vast and intricate, with many studies focusing on different facets of data protection in cloud systems. Subashini and Kavitha gave a thorough rundown of cloud computing security difficulties, emphasizing issues with compliance, insider threats, and data breaches. Suggested reading Hashizume and all; Ali and all; Hashizume, Rosado, Fernández-Medina, & Fernandez. Recent work has concentrated on creating security frameworks and best practices for cloud settings. A layered security approach was proposed by Hashizume and friends in 2013 to solve a variety of cloud security issues. Ali and friends underlined how crucial it is to follow laws like GDPR and HIPAA in order to secure personal information. Nevertheless, there appears to be a deficiency in the attention these studies pay to the specific application of asymmetric cryptography to cloud file storage security (Vines, 2022).

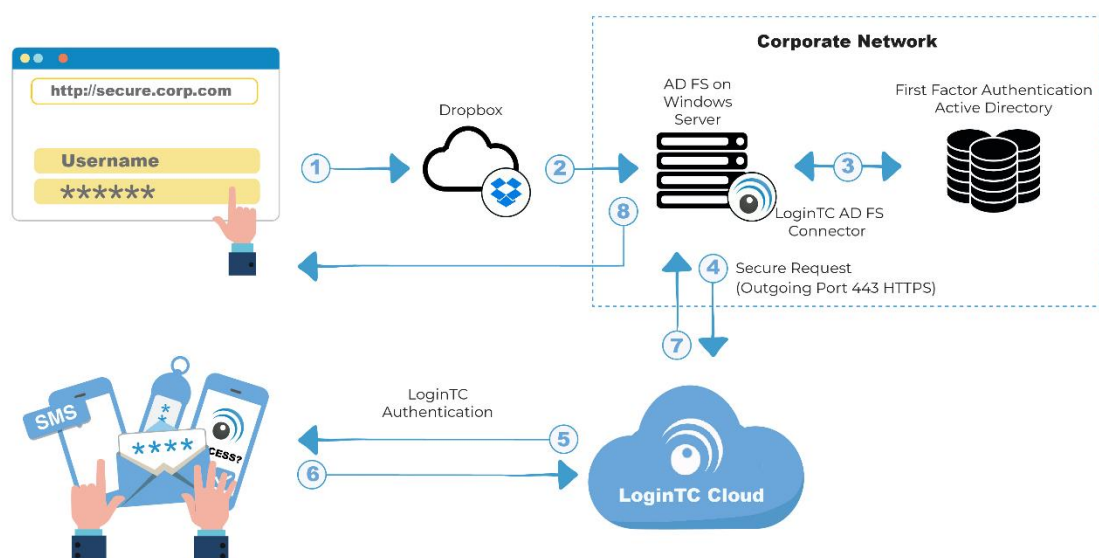


Figure 2: Dropbox Security architecture

2.2.1. Identified Gaps

There are still a few gaps in user authentication, file encryption, key management, and cloud security (Gupta, 2023), despite significant progress in these areas:

2.2.1.1.Coverage gap

Numerous studies concentrate on certain areas of security but do not offer all-encompassing solutions that combine cloud security, key management, encryption, and user authentication into a single system.

2.2.1.2.Context gap

Research on the particular difficulties and remedies associated with putting these security measures into place in cloud-based file storage systems, especially for small and medium-sized enterprises (SMEs), is limited.

2.2.1.3.Timing gap

Current study is necessary due to the cloud's rapid growth and the rise in cyber dangers. Some research that have already been done may be out of date and do not cover the most recent security issues.

2.2.1.4.Methodology gap

Even though a lot of studies suggest theoretical frameworks and models, not much practical research has been done to validate these ideas in actual cloud systems. Further investigation utilizing experiments and case studies is required to evaluate the feasibility and efficacy of suggested remedies.

2.2.2. Conclusion

In order to create a reliable file storage system, the literature study emphasizes the significance of secure user authentication, file encryption, key management, and cloud security. There are still gaps in the integration of these components into a cohesive, workable system for cloud-based file storage, even with major developments. More safe and effective cloud storage systems will be developed as a result of filling in these gaps with thorough, current, and empirically supported research.

2.5. Conceptual Frame work

The purpose of this research project's conceptual framework is to demonstrate the essential elements and how they function together to create a safe file storage system that uses asymmetric cryptography in a cloud setting. This framework, which identifies the key components, their purposes, and the relationships between them, offers a transparent

framework that directs the investigation. This framework's main ideas include cloud security, key management, file encryption and decryption, user authentication, and user interface design.

2.5.1. Relationships among Concepts

2.5.1.1. User Authentication and Authorization ↔ User Interface

The system's user interface makes it easier for users to register and log in, guaranteeing that only authorized users can access it (Babar, 2022).

2.5.1.2. User Authentication and Authorization ↔ Key Management

The key management system generates and gives the user the appropriate encryption keys upon user authentication (M. Alzain, 2021).

2.5.1.3. File Encryption and Decryption ↔ Key Management

The public-private key pairs needed for file encryption and decryption are generated by the key management system (Soni, 2023).

2.5.1.4. File Encryption and Decryption ↔ Cloud Security

Data protection is ensured during storage and transit by cloud security mechanisms, which protect encrypted files (Shukla, 2023).

2.5.1.5. User Interface ↔ File Encryption and Decryption

The user interface (UI) offers ways for users to download and upload files, starting the encryption and decryption procedures (Ahmed, 2022).

2.5.1.6. File management ↔ access control

To guarantee that only authorized users can access, alter, or remove files, file management and access control must work closely together in a secure file storage system. Policies that establish user rights based on roles and attributes are enforced by access control techniques like Attribute-Based Access Control (ABAC) and Role-Based Access Control (RBAC). This guarantees that only authorized individuals can access sensitive files. Furthermore, multi-factor authentication (MFA) improves security by confirming users' identities prior to allowing access to files. In order to prevent unauthorized users from accessing or changing encrypted data,

access control is crucial for maintaining the cryptographic keys required for file encryption and decryption. Cloud storage systems can preserve data security and stop unwanted access by coordinating file management and access control (Chen, 2022).

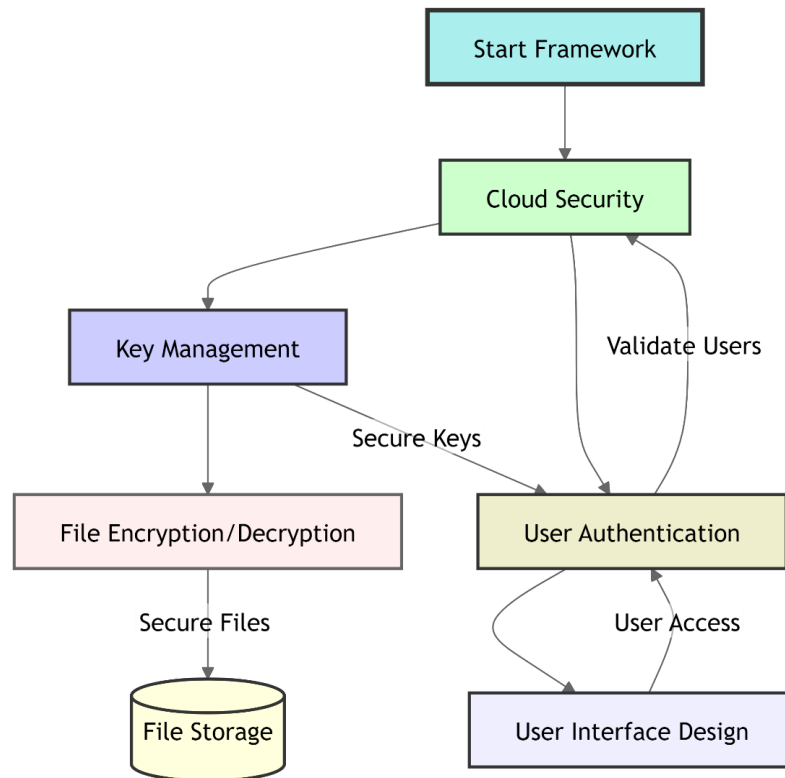


Figure 3: Conceptual Framework

Secure file management and access controls are supported by the user interface (UI), which adds to the overall security of the cloud. The purpose of this project's conceptual framework is to provide an illustration of the fundamental ideas, procedures, and connections that guide the creation and operation of the system. This framework includes all of the essential components needed to manage, encrypt, and store user files securely on a cloud platform.

The connections between these ideas are essential to the overall security and functionality of the system. The first line of defence is user authentication and authorization, which guarantee that only authorized users can access the system. After authenticating, users can upload, download, and safely arrange files using an intuitive interface that facilitates file management system interaction (Cooper, 2023).

Encryption and decryption with asymmetric keys are essential for preserving file secrecy. When files are uploaded, public keys encrypt them so that only the matching private keys may decode them when they are downloaded. Strong key management procedures that guarantee the safe generation, storage, and maintenance of encryption keys serve as the foundation for this procedure. Users may handle their files while on the road because to the system's mobile accessibility, and audit and logging systems keep track of everything to provide a thorough security audit trail. Every functionality has security safeguards in place to guarantee data safety and adherence to legal requirement (Rivera, 2022). Through the interconnection of these ideas, the conceptual framework covers both functional and non-functional needs, guaranteeing a unified and safe system for asymmetric cryptography cloud-based file storage (Carter, 2022).

2.5.2. Conclusion

By creating a safe file storage system that combines effective user identification, reliable key management, and asymmetric cryptography for efficient file encryption and decryption, and extensive cloud security protocols; this research seeks to close some of these gaps. In order to ensure that users can safely manage their files in a cloud environment without experiencing serious performance concerns, the system will be built to strike a balance between security and usability (Roy, 2023).

In conclusion, even though the fields of user authentication, file encryption, key management, and cloud security have made huge advances, more study is still required to solve the lingering issues and gaps. In order to support this endeavour, this research project develops a safe, effective, and user-friendly file storage system that makes use of contemporary cloud security techniques and asymmetric cryptography's advantages (Nguyen, 2022).

CHAPTER THREE: SYSTEM ANALYSIS AND DESIGN

3.1. Introduction

Developing a safe cloud file storage system that satisfies user demands and guarantees data security and efficiency requires careful consideration of system analysis and design. This stage includes analysing the existing system, identifying issues, and creating solutions that meet user needs and organizational goals. The goal of our research is to create a system that will guarantee the security of data kept on cloud platforms. This section offers an in-depth investigation and design of the proposed system.

3.2. Analysis of the current system

3.2.1. Problem of the current system

There are various issues with the current system that uses asymmetric cryptography to secure file storage on the cloud. First, there are significant security vulnerabilities because a lot of cloud storage services use server-side encryption, which gives the cloud provider authority over the encryption keys. This puts user data at risk of security breaches. Users can manage their encryption keys and increase security by switching to client-side encryption. Additionally, non-technical users find it challenging to handle encryption keys due to the complexity of the present systems' user interfaces. This project intends to address these issues by creating a user-friendly interface that makes cryptographic processes simpler.

3.2.1.1. Security Vulnerabilities

Because so many cloud storage options still use antiquated or inadequate encryption, data is vulnerable to cyberattacks. This issue is made worse by weak key management procedures, which may result in data breaches and illegal access. Furthermore, data is only secured during transmission and not at rest because certain services lack end-to-end encryption, which raises the danger of exposure.

3.2.1.2. Complex User Interfaces

Most online storage services have very complex and perplexing user interfaces, which makes it challenging for non-technical users to securely and effectively manage their information. Due of its complexity, user mistakes can happen, including incorrect handling of confidential information.

3.2.1.3. Performance Issues

Current systems frequently experience performance problems in their encryption and decryption procedures, which results in noticeable delays when uploading and downloading files. This has an effect on overall productivity and user satisfaction, particularly for those who routinely handle and access massive amounts of data.

3.2.1.4. Limited Accessibility

For users who need to access their files while on the road, many cloud storage options have poorly optimized mobile interfaces or insufficient support for mobile devices. This limitation can be a major annoyance in the mobile-first world of today.

3.2.1.5. Scalability and Reliability Concerns

Many current systems suffer with scalability as the number of users and stored data develops, leading to interruptions in service and performance deterioration. In addition, frequent failures and interruptions cast doubt on the reliability of these systems and run the risk of resulting in lost or illegally accessible data.

3.2.1.6. Compliance and Legal Issues

A lot of cloud storage providers don't follow data protection laws like GDPR or HIPAA, which can result in legal issues and large fines for companies that store personal or sensitive data. In addition, unreadable data handling practices create privacy concerns and damage consumer trust in the cloud storage provider.

3.3. Analysis of the new system

3.3.1. Introduction

In order to provide a more secure, effective, and user-friendly cloud storage service, the new system for our project was created to address the critical issues found in the current cloud storage solutions. It does this by using powerful encryption techniques, improving user experience, and ensuring regulatory compliance. This section examines the new system, emphasizing its main features and components as well as how it resolves the issues with the current system.

3.3.2. System requirements

Functional requirements, non-functional requirements, software requirements, and hardware requirements are the four main categories of system requirements for the proposed secure file storage on the cloud using asymmetric cryptography.

3.3.2.1. Functional requirements

Functional requirements define the exact performance and functions that comprise the system.

The functional requirements need to be clear, simple, and unambiguous (Nuclino, 2024).

To satisfy user and business needs, the system must accomplish these fundamental functions:

- **User Authentication and Authorization**

Mechanisms for secure user authorization and authentication must be provided by the system. Users need to use a secure authentication method (such two-factor authentication) to register and log in. It is crucial to use role-based access control, or RBAC, to make sure that only people who have permission have access to specific functions.

Table 1. User authentication mechanisms comparison table

Mechanism	Security level	Ease of implementation	User experience
Password-based	Medium	High	Medium
Two-factor authentication	High	Medium	High

- **File Upload and Download**

File uploading and downloading from cloud storage must be safe for users. Before being uploaded, files must be encrypted on the client side, and once downloaded, they must be decrypted on the client side. Various file sizes and types must be supported by the system.

- **Asymmetric Encryption and Decryption**

File security requires the system to use asymmetric encryption, such as RSA. Every user will have their own pair of public and private keys. Prior to uploading, files will be encrypted using the user's public key, which will then be decrypted upon download using the user's private key.

- **Key Management**

Security key management is a requirement for the system. Encryption keys should be generated, stored, and distributed by a secure key management service. To prevent unauthorized access, keys have to be kept in a secure location.

Table 2. Key management techniques table

Technique	Security	Storage method	Implementation complexity
Hardware security module(HSM)	High	Physical device	High
Software-based key storage	Medium	Secure server	medium

1. User Interface (UI)

A user-friendly file management interface is a need of the system. File uploading, downloading, organizing, and searching should all be possible through the user interface. Both technical and non-technical people should find it intuitive and simple to use.

2. Audit and Logging

All user activities need to be recorded by the system for auditing purposes. Information about file uploads, downloads, key management operations, and user authentication events should all be recorded in reports. Only authorized administrators should have access to securely stored information.

3.3.2.2. Non-Functional Requirements

In software engineering, features of a software system that are unrelated to particular functionality or behaviour are referred to as non-functional requirements. Instead of focusing on what the system should accomplish, they explain how it should operate (GeeksforGeeks, July,2024). It describes the functional capabilities and quality aspects of the system. These specifications ensure the system's dependability, security, and efficiency.

- Security

The system has to provide an elevated level of security. This includes secure key management, secure user authentication, authorization procedures, and encryption of data both in transit and at rest. Also, the system must abide by all applicable data protection laws (such as HIPAA and GDPR).

- Performance

The system needs to work well under a range of loads. It should have the least amount of delay when handling big file uploads and downloads. Performance constraints should be avoided by optimizing the encryption and decryption operations.

- **Scalability**

For the system to deal with increasing number of users and data volumes, it must be adaptable. To be able to accommodate growing demand, the design should facilitate horizontal scalability, which enables the simple addition of more resources.

- **Reliability**

There should be no interruptions in the reliability of the system. To provide high availability, duplication and failover methods must be utilized. To prevent data loss, regular backups and data resilience methods should be used.

- **Usability**

The system needs to be accessible and easy to use. The user interface ought to be simple to use and intuitive. It is important to regularly collect user feedback and use them to improve the UI/UX.

- **Compliance**

The system needs to abide by privacy and data protection laws. To comply with legal standards, it should include tools for audit trails, user consent management, and data anonymization.

- **Maintainability**

The system needs to be simple to update and maintain. Updates and maintenance ought to be simple due to the system architecture. Comprehensive documentation is required to support troubleshooting and enhancements to the system.

- **Interoperability**

The system needs to be able to easily interface with other services and systems. Standard APIs should be supported by the system in order to integrate it with apps and services from other parties.

Table 3: system requirement table

Requirement Type	Requirements Description	Priority Level
Functional requirements	User authentication and authorization;	High
Non-functional requirements	Compliance with GDPR	Medium

3.3.2.2.1. Software Requirements

The technology and software required to create and operate the system are specified in the software requirements. It outlines the different types of software that will be utilized to create the system overall.

- a. **Frontend Technologies**
 - i. HTML, CSS, JavaScript.
 - ii. Frontend frameworks/libraries such as React.js, Angular, or Vue.js for building a responsive and interactive UI.
- b. **Backend Technologies**
 - i. PHP and MySQL for server-side scripting and database management.
 - ii. Python for implementing encryption and decryption algorithms.
 - iii. Web server software like Apache or Nginx.
- c. **Development Tools**
 - i. Integrated Development Environment (IDE) such as Visual Studio Code, PyCharm.
 - ii. Version control system like Git.
 - iii. APIs.
- d. **Database Management**
 - i. MySQL or any other relational database management system (RDBMS).
- e. **Other Software**
 - i. Key management service (e.g., AWS KMS, Azure Key Vault) optional.
 - ii. Cloud storage service (e.g., AWS S3, Google Cloud Storage, Dropbox...).



Figure 4: Dropbox Logo

3.3.3. Functional Diagram

The relationship between the primary parts of the secure file storage system and the way operations move within it is depicted visually in the functional diagram. The system's handling of file management (upload and download), user authentication, and encryption which uses asymmetric cryptography to ensure safe file storage is depicted in this diagram.

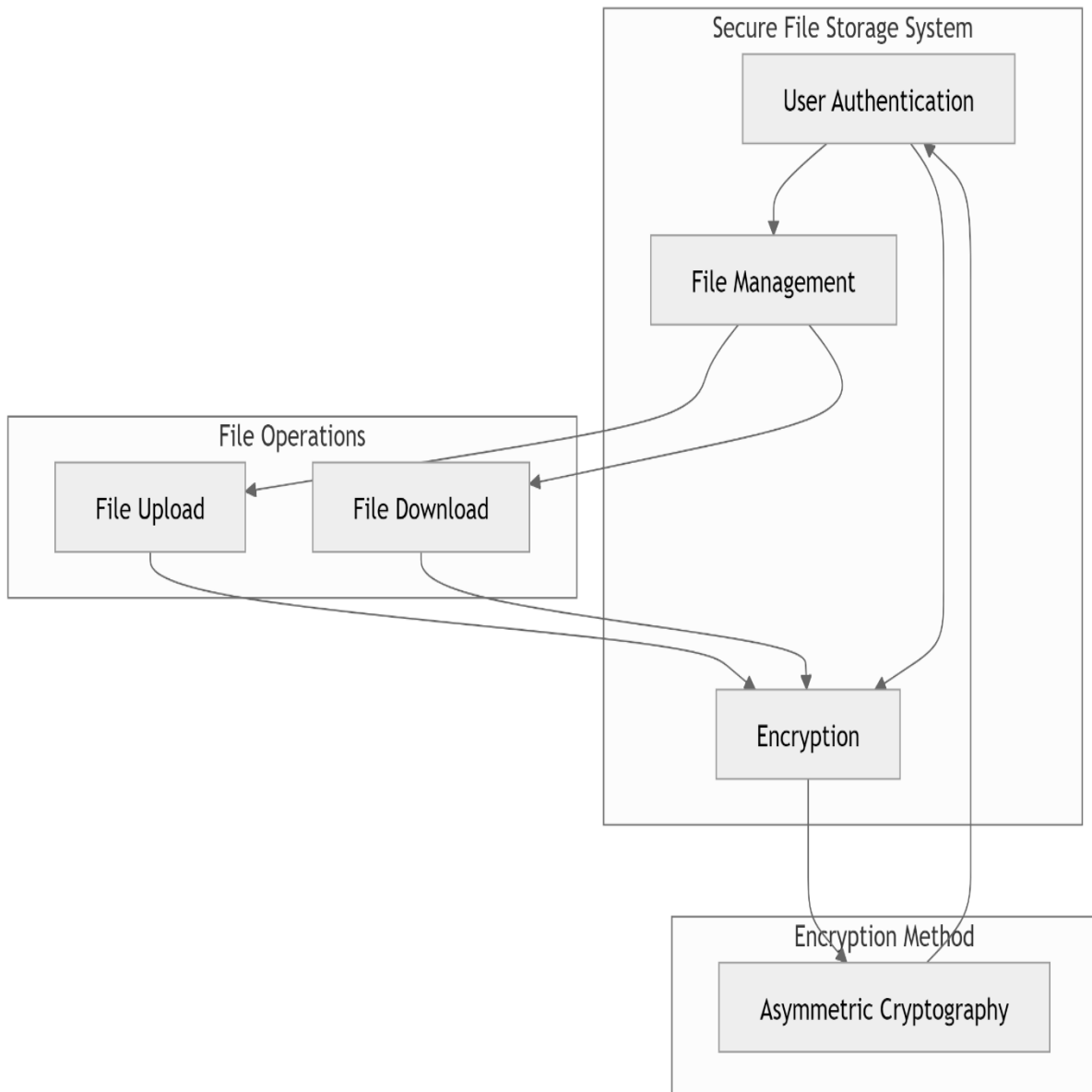


Figure 5: Functional diagram

The functional diagram for the secure file storage system using asymmetric cryptography (RSA) demonstrates key interactions between system components. It starts with user authentication and role-based access control (RBAC), ensuring that only authorized users can

upload or download files. Files are encrypted on the client side using RSA, where the public key secures the data before uploading it to the cloud. These encrypted files are stored securely, protecting them from unauthorized access. When users download files, they are decrypted locally using the corresponding private key. A logging system tracks user actions such as uploads, downloads, and file access for auditing purposes

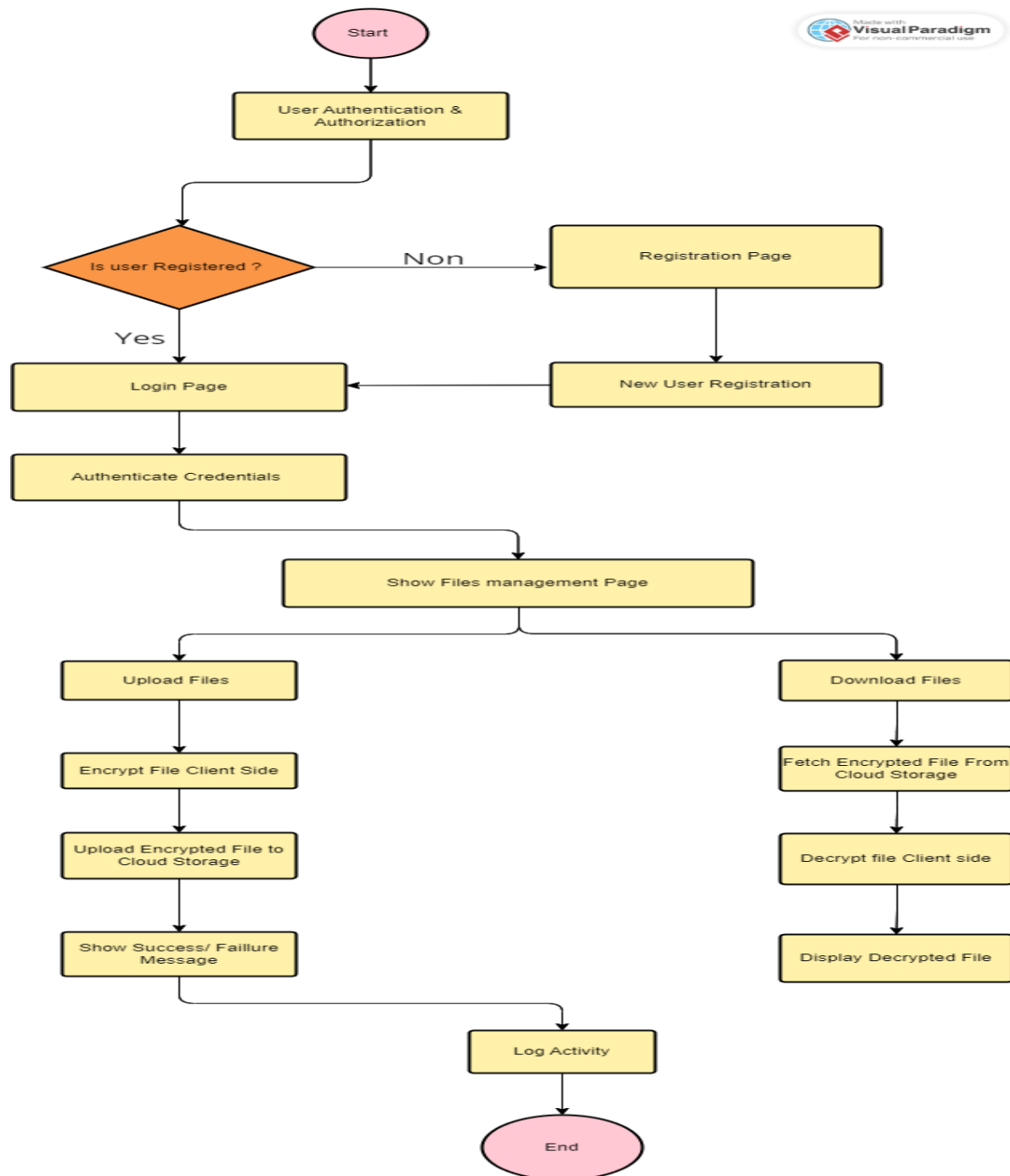


Figure 6: Flow chart Diagram

Based on the functional diagram that describes the overall architecture of the system, showing components such as user interface, user authentication, encryption, file storage, key management and audit and logging by showing how they interact each other, and on the other hand it is the flow chart diagram which represent the sequences of actions taken by the user when they interact with the system.

3.3.4. Methodological approach

3.3.4.1. Data collection techniques

The main methods of collecting information for this study will be documentation and observation. These techniques will provide the thorough and precise information required to comprehend the security of cloud storage as it stands today, the challenges that have faced, and the efficacy of available cryptographic solutions.

Since research approaches and designs are generally orthogonal to data collecting techniques, every kind of data collection technique may theoretically be used with any sort of research strategy. Nonetheless, some methods of gathering data are more frequently used with experimental techniques. Some are more frequently encountered in qualitative research, whereas others are more frequently encountered in comparative or associational (survey) methodologies (Patel, 2021).

3.3.4.1.1. Observation

It requires methodically observing and documenting actions and occurrences as they take place in their natural environments; by using this method the researcher can collect empirical data objectively, gaining a better understanding of the procedures and practices in place, independent of participant self-reports. By observing how cloud storage systems are implemented and used in the real world, one can gain a clear understanding of security procedures, user interactions, and any potential weaknesses or inefficiencies.

3.3.4.1.2. Documentation

In order to complete the documentation process, records and papers pertaining to cryptography and cloud storage security must be gathered and examined. This method offers contextual and historical data that can support and validate results obtained through other methods. Examining documentation serves to acquire a wide range of background data, comprehend industry norms, and examine documented issues and solutions pertaining to encryption and cloud storage.

3.3.4.2. Software development methodology

3.3.4.2.1. Introduction

The Agile technique is a popular approach to software development that places a strong emphasis on flexibility, collaboration, and iterative development. Agile approaches put an emphasis on making tiny, gradual improvements to a project, which enables teams to swiftly adjust to changes and continuously deliver value to stakeholders. The Agile model was selected for this research project on secure cloud file storage using asymmetric cryptography because it can adapt to changing requirements and places a strong emphasis on user feedback and collaboration.

3.3.4.2.2. Agile Methodology Overview

Sprints, or iterative cycles, are what define agile methods and usually last one to four weeks. Producing a potentially shippable product increment is the goal of each sprint. Agile principles place more emphasis on working with customers, adapting quickly to changes, and delivering working software than they do on extensive documentation.

3.3.4.2.3. Key Phases of Agile Methodology

a. Initiation and Planning

Collaborate with stakeholders to collect both functional and non-functional requirements, sort requirements into priority using a product backlog. Sprint planning by choosing items from the backlog to define the first sprint's scope and establish a sprint backlog and specific objectives.

b. Design and Prototyping

System design involves Establish the system architecture, encompassing the database schema, encryption methods, and user interfaces. Utilize UML diagrams to see the various components. Develop prototypes for important system features such as file upload/download, user authentication, and encryption/decryption procedures.

c. Implementation

Execute a sprint by iteratively developing features. Coding, testing, and integration are all part of each cycle. Make sure that at the conclusion of each sprint, working software is delivered. Conduct regular informal discussion to talk about the day's plans, obstacles, and progress.

d. Testing

Unit testing involves evaluating the functionality and dependability of individual components; integration testing verifies that various system modules operate together seamlessly; validation testing compares the system to user requirements and confirms that security criteria are met.

e. Review and Retrospective

Review sprint presents complete features to stakeholders and get their feedback then integrate feedback into the product backlog. Review the sprint procedure, find areas that require work, and make improvements for the upcoming sprint.

f. Deployment

Plan the release of the finished product, making sure that every feature has been tested and considered acceptable.

Continuous Integration and Deployment (CI/CD) use these methods to automate deployment and testing, enabling an easy release process.

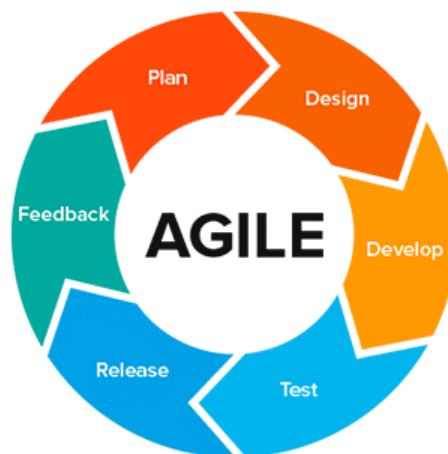


Figure 7: Agile model iterations

3.3.4.3. System Design Methodology

3.3.4.3.1. Introduction

The Object-Oriented System Analysis and Design methodology (OOSADM) is the system design approach used in this project. Using this process, an accurate representation of the system is created by classifying and identifying each of its components, specifying their connections, and using a variety of diagrams to show how the system behaves (S. Bennett, 2022).

System analysis in the context of a secure cloud file storage system entails a thorough investigation of current file storage and management procedures. This includes using techniques like surveys, interviews, and observation to get specific requirements from stakeholders. Understanding the security, usability, and performance limits of the current systems is the main goal. For instance, problems like insufficient user authentication, insufficient encryption methods, and ineffective file management procedures are recognized and recorded (Lewis, 2021).

3.3.4.3.2. Steps Involved in System Design

3.3.4.3.2.1. Solution Design

Developing a solution design based on the requirements acquired during the analysis phase is the first step in the design process. In order to do this, one must comprehend the current system, if any, and use system sequence diagrams to lay out a new system architecture.

3.3.4.3.2.2. Object Identification and Classification

Determine which objects are present in the system and classify them according to shared characteristics and behaviours. This facilitates the logical and effective arrangement of the system's components.

3.3.4.3.2.3. Defining Class Hierarchy and Relationships

Define the relationships between the classes and establish the hierarchy among them. This involves figuring out which classes are dependent on which classes, how classes interact with one another, and the inheritance hierarchy.

3.3.4.3.2.4. Application Framework Definition

Create an application framework that describes the system's general architecture. This framework guarantees that all of the software's components are seamlessly integrated and acts as a roadmap for its construction.

3.3.4.3.3. Tools Used in Object-Oriented System Analysis and Design Methodology

3.3.4.3.3.1. Use Case Diagram

The interactions between users, or actors, and the system are illustrated in the use case diagram. It displays the connections between actors and use cases as well as the different use cases (functionalities) that the system will handle.

Use Cases (such as login, register, upload, and download files), Actors are users, and Relationships (such as associations between actors and use cases) are all included.

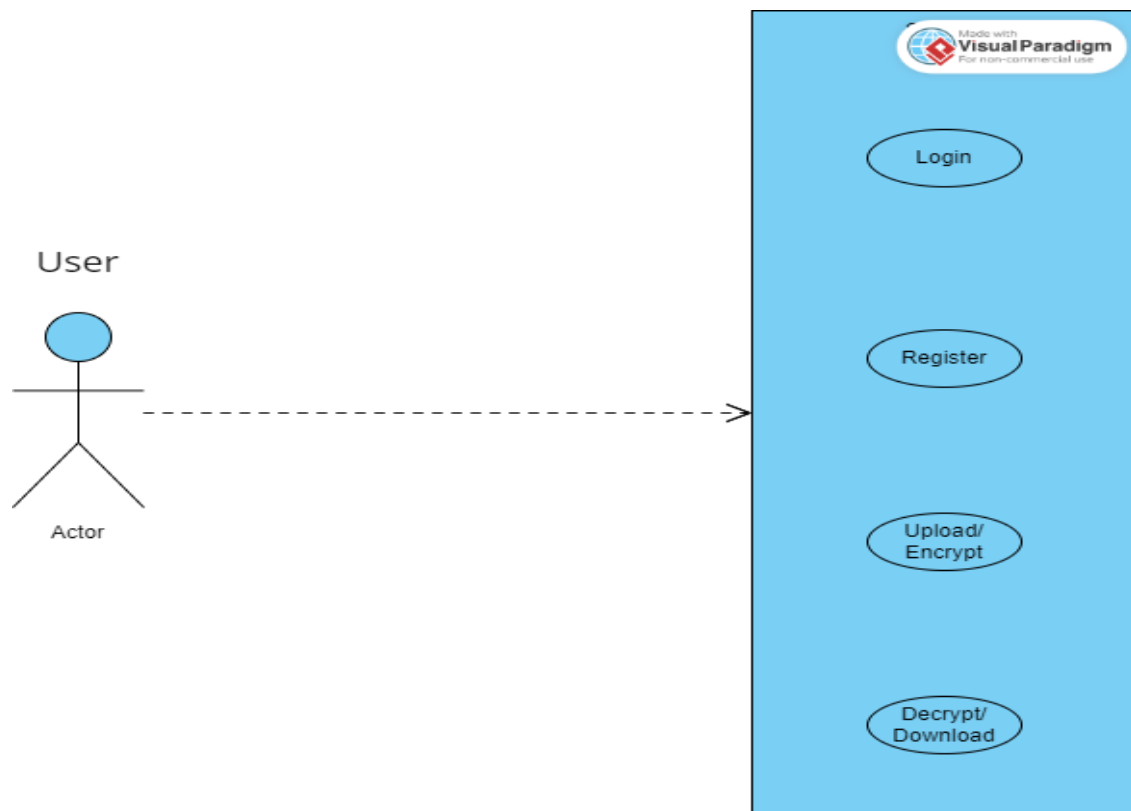


Figure 8: Use Case Diagram

3.3.4.3.3.2. Class Diagram

The system's classes, their properties, methods, and the connections between them are all represented in the class diagram, which offers a static representation of the system.

The terms "classes" refers to user, file, secure file; "attributes" to filename, password, file path; "methods" to login, register, upload, and download files; and "relationships" to associations and generalizations.

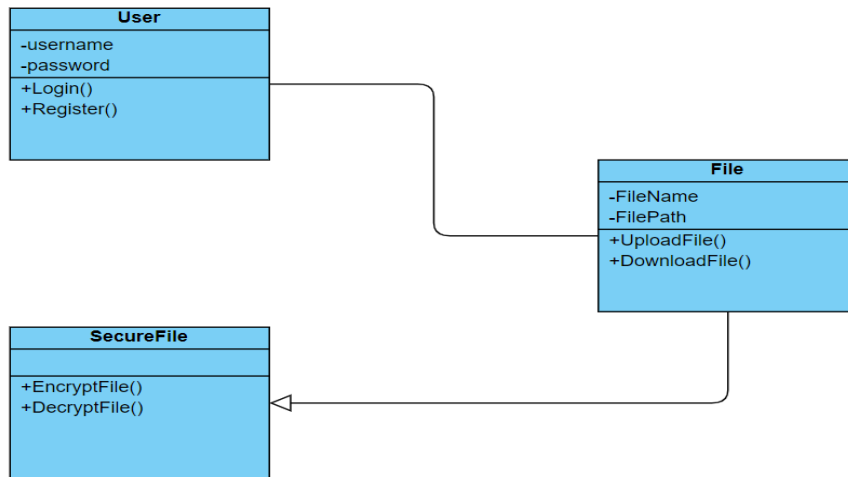


Figure 9: Class Diagram

3.3.4.3.3. Sequence Diagram

The message flow between objects in a particular circumstance is illustrated in the sequence diagram. It shows how various objects interact in a specific order to produce a desired result. Messages (such as login request, authentication response), Activation bars (such as the duration an object is active), Objects (such as User, System, Database), and Lifelines (such as the amount of time of object existence).

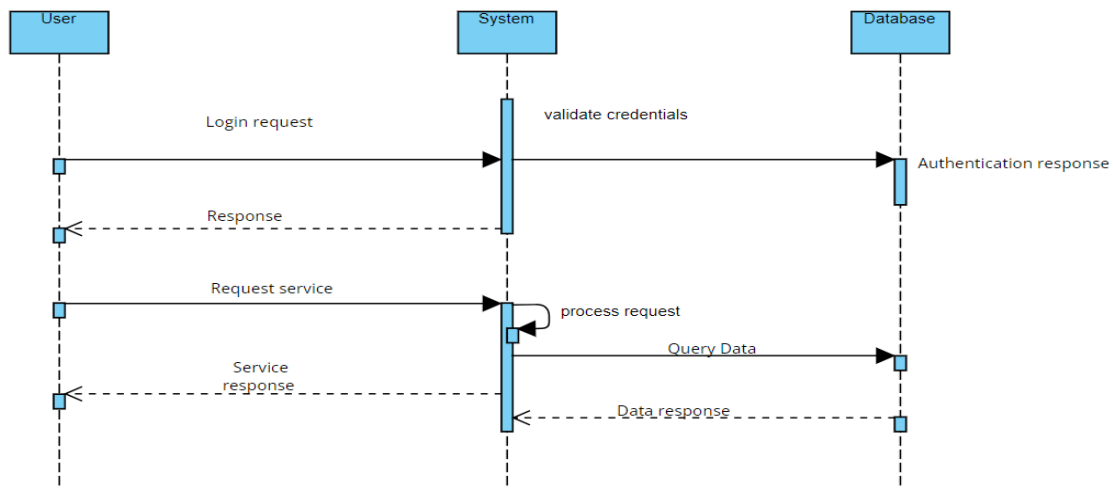


Figure 10: Sequence Diagram

3.3.4.3.4. Activity Diagram

The activity diagram illustrates how decisions and activities are made, capturing the dynamic aspects of the system. Modelling the logic of intricate processes and workflows is one of its uses. Transitions (such as moving from one activity to another), Start and End points, Activities (such as uploading, downloading, logging in, registering), and Decisions (such as is the user authenticated?).

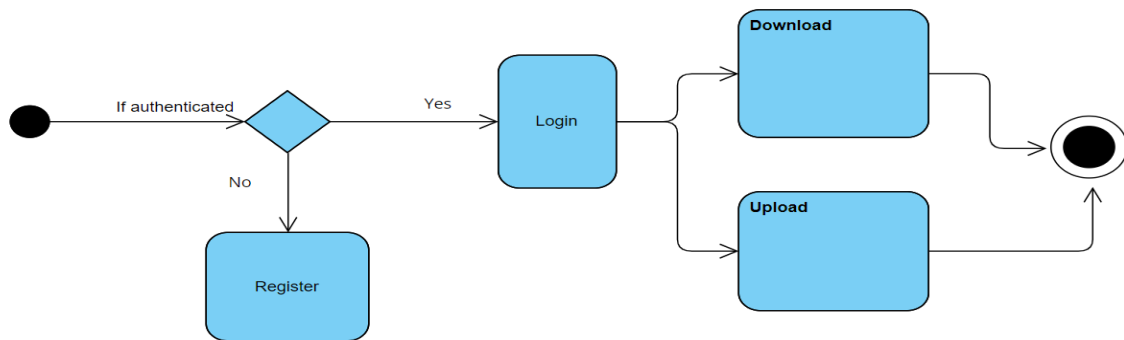


Figure 11: Activity Diagram

The systematic and comprehensive system design process is carried out by the use of the Object-Oriented System Analysis and Design Methodology (OOSADM). It involves categorizing things and relationships, identifying them, and using different diagrams to show how the system behaves. This approach makes that the system is scalable, modular, and well-organized, which makes development, maintenance, and future improvements easier. The design process is made more visible and comprehensible by using tools like use case diagrams, class diagrams, sequence diagrams, and activity diagrams. This promotes efficient communication and teamwork within the development team.

CHAPTER FOUR: SYSTEM IMPLEMENTATION

4.1. Implementing and coding

4.1.1. Introduction

The process of using code to transform a concept into a functional system is known as system implementation. This chapter describes the steps involved in putting our cloud-based file management system into operation, including user authentication and asymmetric cryptography-enabled secure file transfer. In order to accomplish the intended functionality, the system combines frontend and backend components and makes use of a variety of tools and technologies.

4.1.2. Description of Implementation Tools and Technology

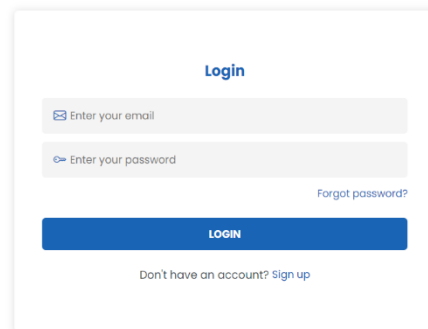
4.1.2.1. Frontend Technologies

The framework of the web pages, including the login, registration, and file management interfaces, are created using HTML while CSS is used to style web pages so that the user interface is both visually pleasing and responsive and finally is JavaScript Used for handling form validations, user interactions, and dynamic content updates on the client-side of scripting.

4.1.2.2. Backend Technologies

- ✓ PHP: This programming language is used to handle user authentication, registration, and file management tasks through server-side scripting. It also helps to communicate between the frontend and the database.
- ✓ Python: This language is used to implement asymmetric cryptography, which guarantees safe file uploads and downloads.
- ✓ MySQL: This relational database management system stores user data, file metadata, and other relevant information.
- ✓ Python programming language will employ the RSA Algorithm for file encryption and decryption to protect data during storage and transmission.

4.1.3. Screenshots and source codes

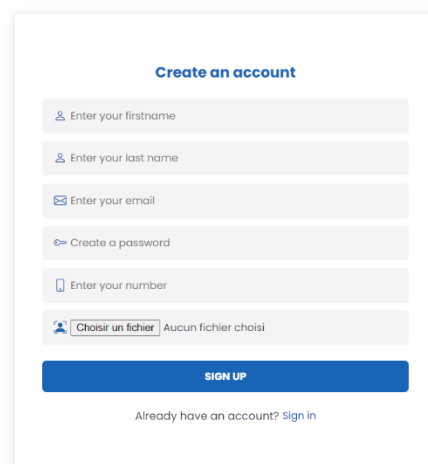


The screenshot shows a login form with the following elements:

- Title: **Login**
- Input field: Enter your email (with an envelope icon)
- Input field: Enter your password (with an eye icon)
- Link: [Forgot password?](#)
- Button: **LOGIN**
- Text: Don't have an account? [Sign up](#)

Figure 12: Login Page interface

The login interface is where the user is prompted to enter his or her e-mail and password the access the system with the corrected credentials.



The screenshot shows a registration form with the following elements:

- Title: **Create an account**
- Input field: Enter your firstname (with a person icon)
- Input field: Enter your last name (with a person icon)
- Input field: Enter your email (with an envelope icon)
- Input field: Create a password (with an eye icon)
- Input field: Enter your number (with a phone icon)
- Input field: [Choisir un fichier](#) Aucun fichier choisi (with a file icon)
- Button: **SIGN UP**
- Text: Already have an account? [Sign in](#)

Figure 13: Register Page

The register page it is set for new users to register themselves into the system by provided different elements so that once the success to register they will be able to access the system easily when they will need to login.

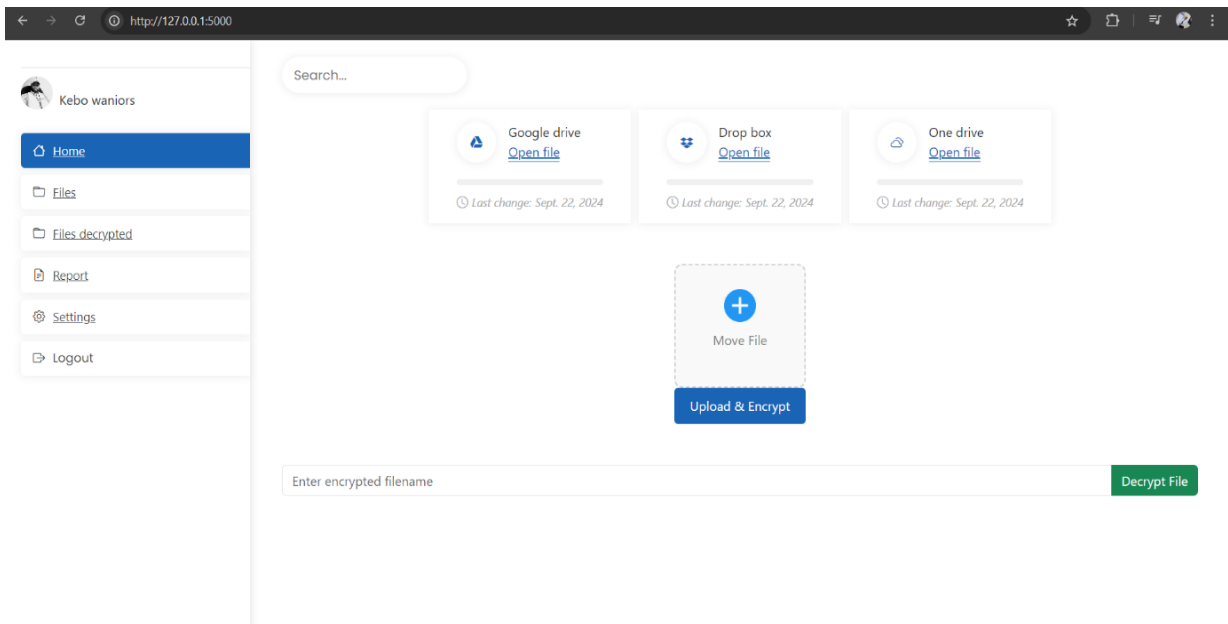


Figure 14: Main Page

The main page interface is first page once the user login into the system and I where all the operations are shown so that the user can choose one of them to navigate into the system by clicking a given button.

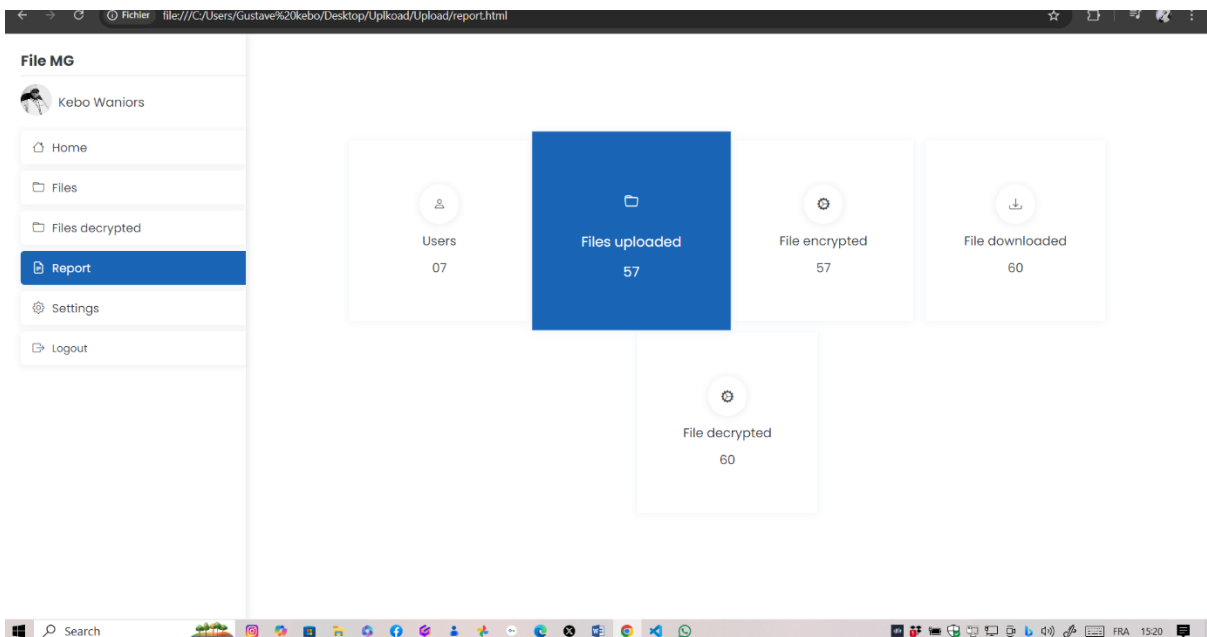


Figure 15: Report Menu

This page elaborates the operations which have been done in the system where it shows the number of users that were connected to the system, the number of files that are been decrypted, encrypted, downloaded and uploaded.

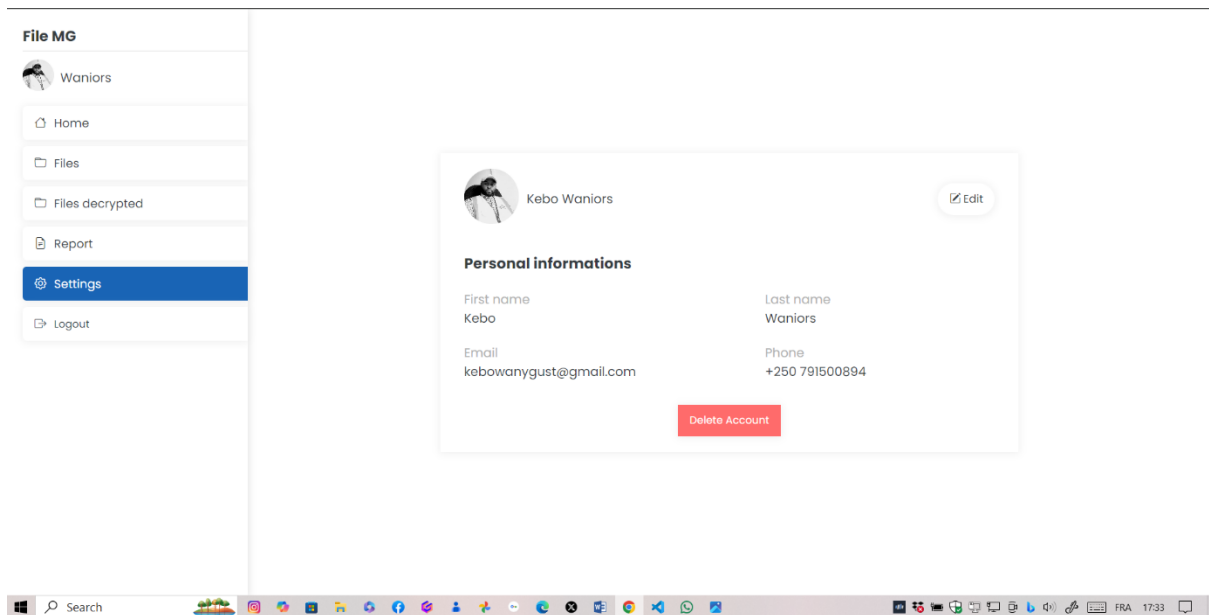


Figure 16: Setting option

The setting option give the user the possibility of either delete his account or edit it.

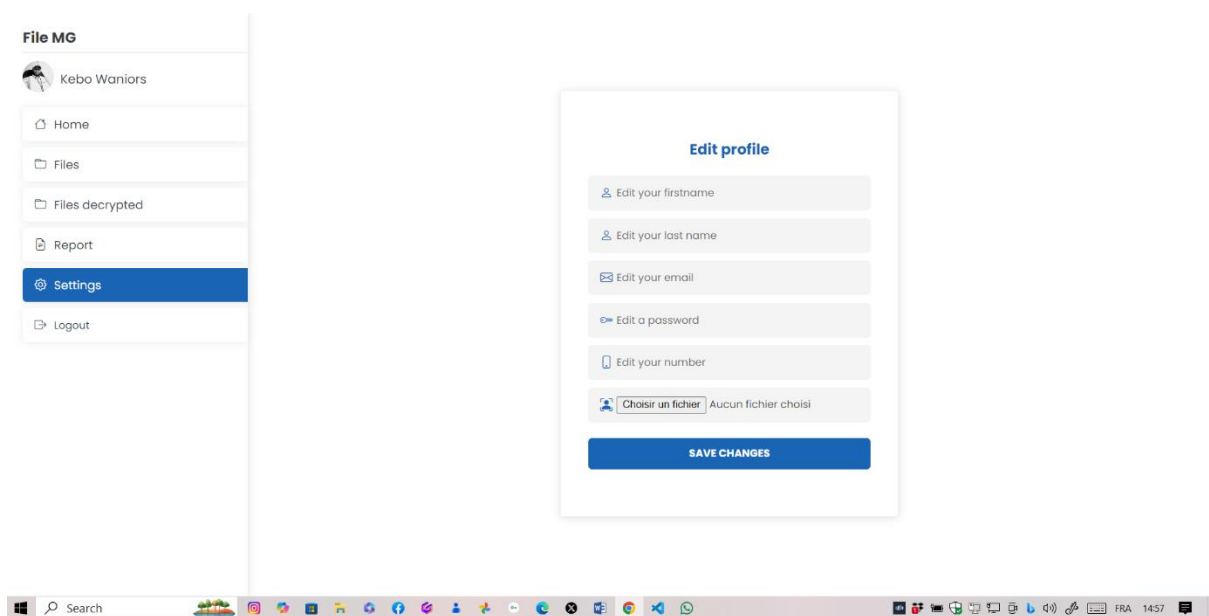


Figure 17: Edit Profile Option

This option is set to help user to edit his or her profile instead of delete it.

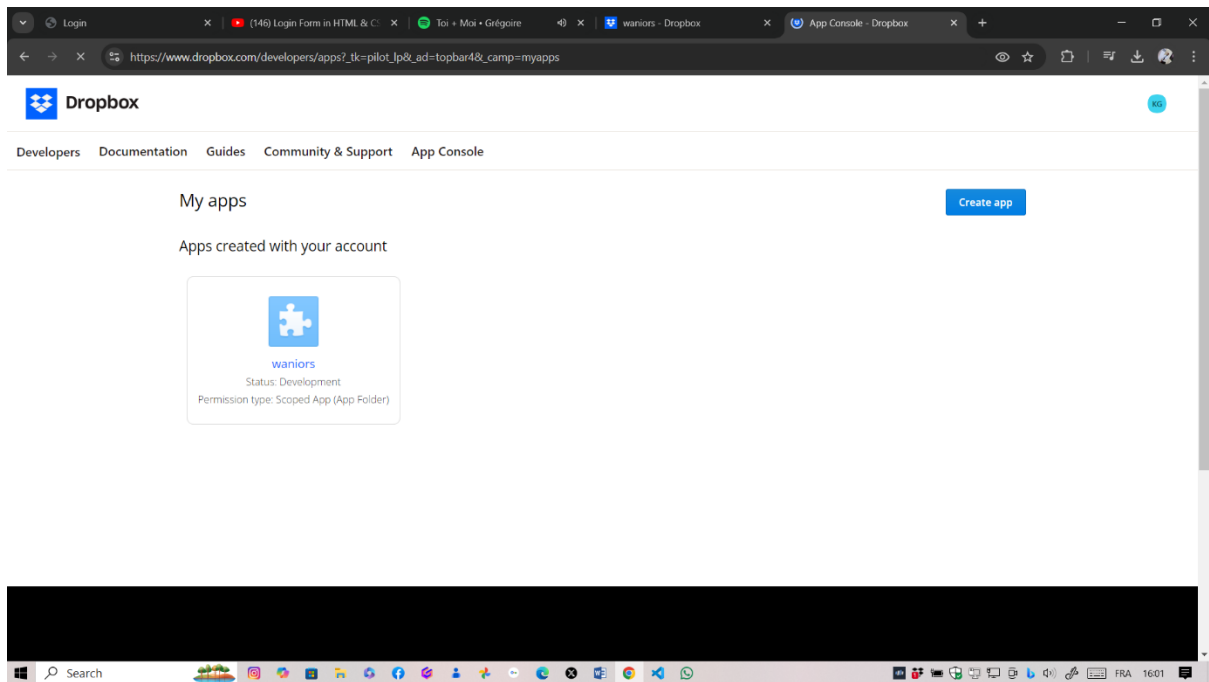


Figure 18: Dropbox console for application creation

To create an application in the Dropbox console, it is necessary to register in the Dropbox developer portal (<https://www.dropbox.com/developers>). Once logged in, select “Create App” and choose the type of app you want to create, such as full Dropbox access or specific access to app folders. Choose the permissions your app needs, such as uploading or downloading files. After you configure your app settings, Dropbox creates an App Key and App Secret, which are required for API calls. These credentials can now be used to connect your app to the Dropbox API, allowing it to interact with features like file storage, retrieval, and sharing.

4.2. Testing

4.2.1. Introduction

A crucial stage of the software development lifecycle, testing looks for and fixes bugs, confirms that the program satisfies requirements, and makes sure the system works as expected. The primary objective of testing is to validate the quality of software systems by the methodical execution of the software under carefully regulated conditions (Luo).

The several testing methods used on the cloud-based file management system are described in this section. These methods include unit testing, validation testing, integration testing, functional and system testing, and acceptance testing.

Each object that people encounter undergoes inspection for quality. In the field of software development, where it is essential to properly inspect the software system at various testing

stages, the notion of quality has also been introduced. Because of the intense competition these days and the rapid changes in platforms and business requirements, software needs to be updated and supported according to the latest requirements in order to be stable and in use for an extended period of time.

One of the general tasks carried out at any company to assure the value and quality of software products and their life on the market is software testing (Honest, May 2019).

4.2.2. Unit Testing Outputs

Unit testing ensures that each unit of the system functions as planned by testing it separately. It concentrates on confirming the functionality of individual parts or units.

4.2.3. Validation Testing Outputs

Validation testing certifies that the system fulfils the stakeholder-established requirements and specifications. It verifies that the system carries out the desired tasks properly.

4.2.4. Integration Testing Outputs

Integration testing confirms how the system's various components interact with one another. It guarantees that integrated units function as intended.

4.2.5. Functional and System Testing

Functional testing compares the system's working to the functional specifications. System testing examines how well the entire system complies with the given specifications.

4.2.6. Acceptance Testing Report

To determine whether the system satisfies the acceptance criteria and is set up for deployment, acceptance testing is carried out. Stakeholders or end users must test the system to ensure that it is functioning and user-friendly.

4.2.6.1. Summary of Acceptance Tests

The goal is to verify that the system satisfies all requirements and is easy to use. Examining every feature, such as user registration, login, file upload, encryption, download, and decryption, is the scope of the testing.

4.2.6.2. Tests Cases Performed

4.2.6.2.1. Encryption, uploading

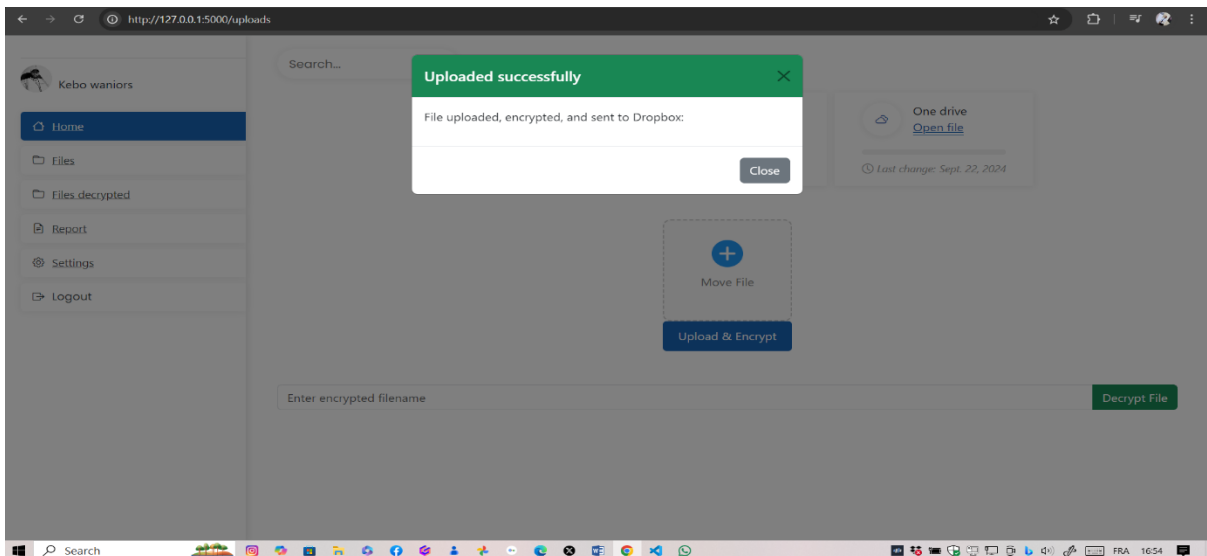


Figure 19: Uploading file and Automatically Encrypted

Here the user drag or click the + icon to insert file that he or she want to upload to any cloud service provide, for our case we use the drop box, after the user is insert the file he will just click the upload & encrypt button then the file will automatically upload and the encrypted format will be store in a given folder into the drop box.

4.2.6.2.2. Downloading, and decryption of files

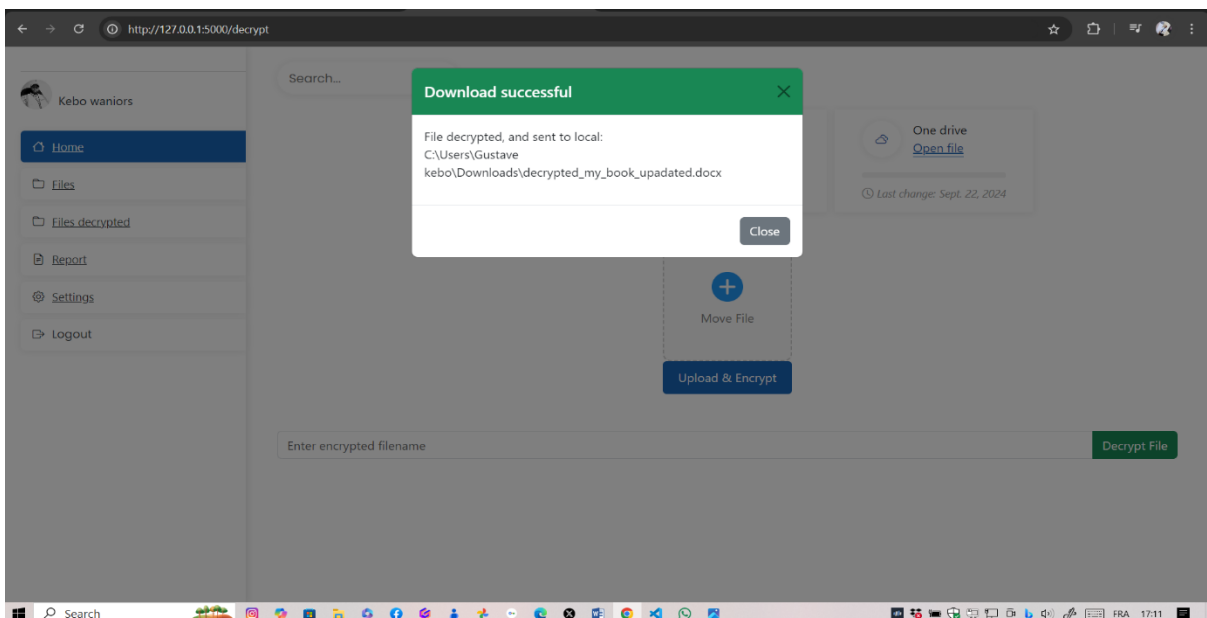


Figure 20: Downloading Decrypted file.

For this operation, the user will enter the name of the file into the research bar and he will make sur that the name is exactly the same with the name he want to decrypt and download from

cloud (drop box). Once he done the operation, the decrypted file will automatically save in local disk of his computer and the Downloads folder.

Table 4: Performance Testing Results Table

Test scenario	File size	Encryption time	Decryption time	Upload time	Download time
Scenario 1	3.2MB	25secs	8Secs	38secs	5secs
Scenario 2	1.1MB	8sec	3secs	14secs	3secs
Scenario 3	6.3MB	56secs	25secs	30secs	19secs

4.2.6.2.3. Control of access and security inspections

Every acceptance test case was completed properly, as expected, users were able to upload and download data, register, log in, and execute encryption and decryption. Significant issues did not arise, and small issues found during testing were quickly fixed.

CONCLUSION AND RECOMMENDATIONS

CONCLUSION

By using asymmetric cryptography to address the important problems of secure cloud file storage, this research project has successfully handled the crucial areas of data security, user authentication, and key management. The system makes sure that sensitive files are encrypted before being transferred to the cloud and can only be decoded by the intended recipient by utilizing RSA asymmetric encryption. This method offers strong security against unwanted access, which is a big improvement in cloud data security.

The system has passed the acceptance testing phase and is seemed ready for deployment. Users have validated the functionality and are satisfied with the system's performance and usability.

Furthermore, it has been shown that the system's user authentication mechanism which combines role-based access control and multi-factor authentication is essential for avoiding unauthorized access. These elements further improve the overall security framework by guaranteeing that users are only allowed access according to their designated roles, in addition to providing account security. Keeping the integrity of encryption keys, which are necessary for the system to work, has also been made possible by the inclusion of a secure key management service. The distribution, storage, and creation of keys have all been done according to the correct procedures, guaranteeing that they are safe from abuse or unwanted access.

Additionally, the project was successful in increasing accessibility and usability thanks to the creation of a user interface that is simple to use and intuitive for both technical and non-technical people. Users may safely access their files from a variety of platforms thanks to the system's responsiveness on mobile devices, which improves accessibility overall. In addition, the system's architecture has been designed to facilitate scalability, meaning that speed won't be compromised as the number of users and data quantities increase. In order to maintain compliance with legal and regulatory requirements, the system also complies with pertinent data protection legislation.

To sum up, this research effort has advanced the creation of a safe, expandable, and intuitive cloud file storage system. Strong security measures combined with an emphasis on compliance and usability create a strong foundation for future developments in safe cloud storage solutions.

RECOMMENDATIONS

Even though secure cloud storage has advanced significantly with the existing system, ongoing research and development are still necessary to handle new issues, strengthen security, and increase user experience. To ensure that the system continues to be reliable, user-friendly, and compliant with changing technology and regulatory environments, future research should concentrate on the topics listed below:

- **Improve encryption and key management algorithms:** Even if the system securely protects files with RSA encryption, more sophisticated encryption algorithms and approaches should be investigated in the future to further improve data security. As technology develops, researching encryption techniques resistant to quantum errors may become essential.
- **Enhance Usability and User Interface Design:** More research might concentrate on improving the user experience, particularly for non-technical users, even though the current system has an intuitive interface. This can entail creating more user-friendly navigational tools and providing customisation choices for various user roles.
- **Integrate Machine Learning for Security:** In the future, research may focus on integrating machine learning algorithms to instantly identify and stop possible security risks. To find illegal access attempts or questionable activity, this could involve anomaly detection and predictive analysis.
- **Increase Platform Cross-Platform Interoperability and Mobile Accessibility:** Though more study might focus on optimizing the system for a larger range of devices and operating systems, the technology presently supports mobile accessibility. Users would be able to safely access their data from any location and on any device thanks to this.
- **Respect of New Regulations:** Continuous research is required to make sure the system complies with new legal requirements when data protection legislation change. Subsequent research endeavours may examine the incorporation of compliance automation mechanisms to facilitate smooth adherence to these requirements by enterprises.
- **Studies on Security Awareness and User Behaviour:** Examining user interactions with security features and their comprehension of data protection might provide important information for improving the security features of the system. Programs for system integration-based user education and awareness may be necessary for this.

- **Scalability and Performance Optimization:** Since the system is made to be scalable, more study may be done to find ways to maximize performance in large-scale deployments and under severe loads. To increase productivity, methods like cloud-native architectures and distributed computing could be investigated.
- **Examine Multi-Cloud Redundancy Strategies:** Investigating the utilization of multi-cloud strategies may be helpful to further improve reliability and lower the danger of data loss. This would entail sharing data among several cloud service providers in order to guarantee availability and redundancy.
- **Additional Research on User Authentication Techniques:** To add more security layers, future studies may look into more sophisticated user authentication techniques such as biometric authentication or decentralized identity systems.

SUMMARY

The goal of this research project was to create a secure cloud storage system that protects data by utilizing asymmetric cryptography. The study started by listing the limitations of the available cloud storage options, including performance problems, complicated user interfaces, and vulnerabilities in security. In order to solve these problems, the system was built with capabilities including key management, RSA file encryption, user authentication, and compliance monitoring with privacy regulations. In order to ensure flexibility and continual improvement during the system's development process, the project also explored the adoption of the agile methodology. Object-Oriented Systems Analysis and Design (OOSADM) was the methodology used in the system design, with the goal of developing a dependable and scalable architecture.

Phases of testing were carried out to verify the functionality, overall performance, and integration of the system. In order to provide real-time security, the project's conclusion emphasized the significance of strengthening encryption techniques, expanding usability, and incorporating cutting-edge technology like machine learning.

Future study directions were suggested, encompassing investigating more sophisticated encryption methods, maximizing mobile usability, and guaranteeing adherence to new laws. The project made clear how important it is to continue conducting research in order to handle new problems and enhance the system's usability, security features, and resilience.

REFERENCES

- Ahmed, H. A. (2022). Designing Secure User Interfaces for Data Encryption and Decryption.
- Al Zahrani, M. A. (2021). Multi-Factor Authentication: A Comprehensive Review.
- Albrecht, J., & Binns, R. V. (2021;2018). The GDPR and Data Protection: Principles and Practices;The Ethical Implications of GDPR: Lessons from the US and UK. Oxford University Press;Cambridge University Press.
- Anderson, J. (2020). Continuous Integration and Continuous Deployment. Tech Press.
- Anderson, R. (2018). Security Engineering: A Guide to Building Dependable Distributed Systems. Wiley.
- AppViewX, T. (2024). Asymmetric Encryption. Retrieved from The AppViewX: <https://www.appviewx.com/education-center/asymmetric-encryption/>
- Babar, D. S. (2022). User Authentication: The Key to Securing Cloud Services.
- Bell, J. (2019). Doing Your Research Project. McGraw-Hill Education.
- Carter, N. G. (2022). Asymmetric Cryptography in Secure Cloud Environments: Frameworks and Best Practices.
- Chen, S. C. (2022). Access Control Models and Security Policies for Cloud-Based File Systems.
- Cooper, Y. R. (2023). Cloud Computing Security: Foundations and Innovations.
- Corporation(IDC), I. D. (2020). Worldwide Public Cloud Services Spending Forecast to Reach \$500 Billion in 2023.
- Foundation., P. S. (2021). PyCryptodome Documentation. Retrieved from pycryptodome: <https://www.pycryptodome.org/src/overview>
- GAO, Y. S. (2020). A Novel Hybrid Encryption Scheme Based on AES and RSA for Cloud Data Security.
- GeeksforGeeks. (July,2024). Non-Functional Requirements in Software Engineering. software engineering . Retrieved from <https://www.geeksforgeeks.org/non-functional->

requirements-in-software-
engineering/?itm_source=auth&itm_medium=contributions&itm_campaign=articles

- Gikow, R. (2017). *Ethical Guidelines for Conducting Research*. Routledge.
- Gupta, J. P. (2023). *User Authentication and Data Encryption in Cloud Computing: A Review*.
- Hassan, A. A. (2023). *A Survey on Cloud Storage Management: Trends and Challenges*.
- Honest, N. (May 2019). *Role of Testing in Software Development Life Cycle* . *International Journal of Computer Sciences and Engineering Open Access* , 886-889.
- Housley, R. (2018). *Practical Cryptography and Key Management: Securing Digital Communication*.
- Jhanjhi, N. Z. (2019). *A Review Paper on Security in Cloud Computing*. *International Journal of Computer Science and Network Security*, 63-71.
- Kaur, P. K. (2021). *Secure File Storage in Cloud Computing Using Asymmetric Cryptography*.
- Kumar, P., & Mittal, N. (2020). *Enhancing cloud security using hybrid encryption*. *Journal of Cloud Computing: Advances, Systems and Applications*, 1-18.
- kv, s. (). *Structured Systems Analysis and Design Methodology*. ITC Infotech India Ltd, 1-7.
- Lewis, R. W. (2021). *System Analysis and Design in Cloud Environments*.
- Li, J. L., Chen, X. L., & Lee, P. P. (2019). *A hybrid cloud approach for secure authorized deduplication*. *EEE Transactions on Parallel and Distributed Systems*, 1206-1216.
- Lindell, J. K. (2020). *Introduction to Modern Cryptography*.
- Luo, L. (n.d.). *Software Testing Techniques, Technology Maturation and Research Strategies*. Institute for Software Research International, 1.
- M. Alzain, B. Z. (2021). *Key Management in Cloud Computing: A Survey*.
- Mao, W. (2022). *Modern Cryptography: Principles and Practice*.
- Martin, R. C. (2022). *Agile Software Development: Principles, Patterns, and Practices*.
- Nguyen, J. P. (2022). *Asymmetric Encryption and Cloud Security Frameworks: Balancing Usability and Protection*.

- Nuclino. (2024). A Guide to Functional Requirements (with Examples). Retrieved from <https://www.nuclino.com/articles/functional-requirements#what-are-functional-requirements>
- Patel, J. A. (2021). Research Design and Methodology: Security and Privacy in Cloud Computing.
- Poulton, N. (2021). Data Storage Networking: Real World Skills for the CompTIA Storage+ Certification and Beyond.
- Rivera, H. F. (2022). Asymmetric Encryption and Key Management in Cloud Storage.
- Roy, S. D. (2023). Cloud Storage Security: Algorithms and Methods for Data Protection.
- S. Bennett, R. F. (2022). Object-Oriented Systems Analysis and Design Using UML.
- Schwaber, K. (2021). Agile Project Management with Scrum.
- Shukla, R. K. (2023). Security and Privacy Issues in Cloud Computing: A Survey.
- Sivankalai, S. (2021). The Impact of Cloud Computing on Academic Libraries. Library Philosophy and Practice.
- Soni, N. K. (2023). Secure File Encryption and Decryption: A Key Management Perspective.
- Stallings, W. (2017). Cryptography and Network Security: Principles and Practice. USA.
- Stallings, W. (2017). Cryptography and Network Security: Principles and Practice.
- Stallings, W. (2020). Cryptography and Network Security: Principles and Practice.
- Stallings, W. (2020). Cryptography and Network Security: Principles and Practice.
- Sun, D., Chang, G. S., & Gao, X. (2018). A hybrid encryption scheme for data security in cloud computing. *Procedia Computer Science*, 314-320.
- Thary, H. H. (April 2019). ASYMMETRIC CRYPTOGRAPHY. Iraqi University.
- Thomas Erl, R. P. (2023). Cloud Computing: Concepts, Technology & Architecture.
- Tim Mather, S. K. (2020). Cloud Security and Privacy: An Enterprise Perspective on Risks and Compliance.
- Vines, R. L. (2022). Cloud Security: A Comprehensive Guide to Secure Cloud Computing.

Voigt, P. &. (2017). he EU General Data Protection Regulation (GDPR).

Wazid, M. (2018). Design of Secure User Authenticated Key Management Protocol for Generic IoT Networks. IEEE Internet of Things Journal , 253.

Wazid, M. D. (February 2018). Design of Secure User Authenticated Key Management Protocol for Generic IoT Networks. IEEE Internet of Things Journal, 269 - 282.

Yin, R. K. (2018). Case Study Research and Applications: Design and Methods.

APPENDICES

Table 5: Time frame

Tasks	Start-date	End-date	Observations
Planning	29 January	16 February	18 days
Analysis and designing	17 February	2 March	14 days
Developing	3 March	15 July	134 days
Testing	16 July	29 July	13 days
Deploying	5 August	10 August	5 days
Review	25 August	15 September	21 days

```

27
28 # RSA key file paths
29 rsa_public_key_file = 'public.pem'
30 rsa_private_key_file = 'private.pem'
31
32 def download_from_dropbox(dropbox_path, local_path):
33     dbx = dropbox.Dropbox(DROPBOX_ACCESS_TOKEN)
34     try:
35         metadata, response = dbx.files_download(path=dropbox_path)
36         with open(local_path, 'wb') as f:
37             f.write(response.content)
38         print(f"Downloaded {dropbox_path} to {local_path}")
39     except dropbox.exceptions.ApiError as e:
40         print(f"Dropbox API error: {e}")
41     except Exception as e:
42         print(f"Error downloading file from Dropbox: {e}")
43
44 def generate_rsa_key_pair():
45     private_key = rsa.generate_private_key(
46         public_exponent=65537,
47         key_size=2048,
48         backend=default_backend()
49     )
50     public_key = private_key.public_key()
51

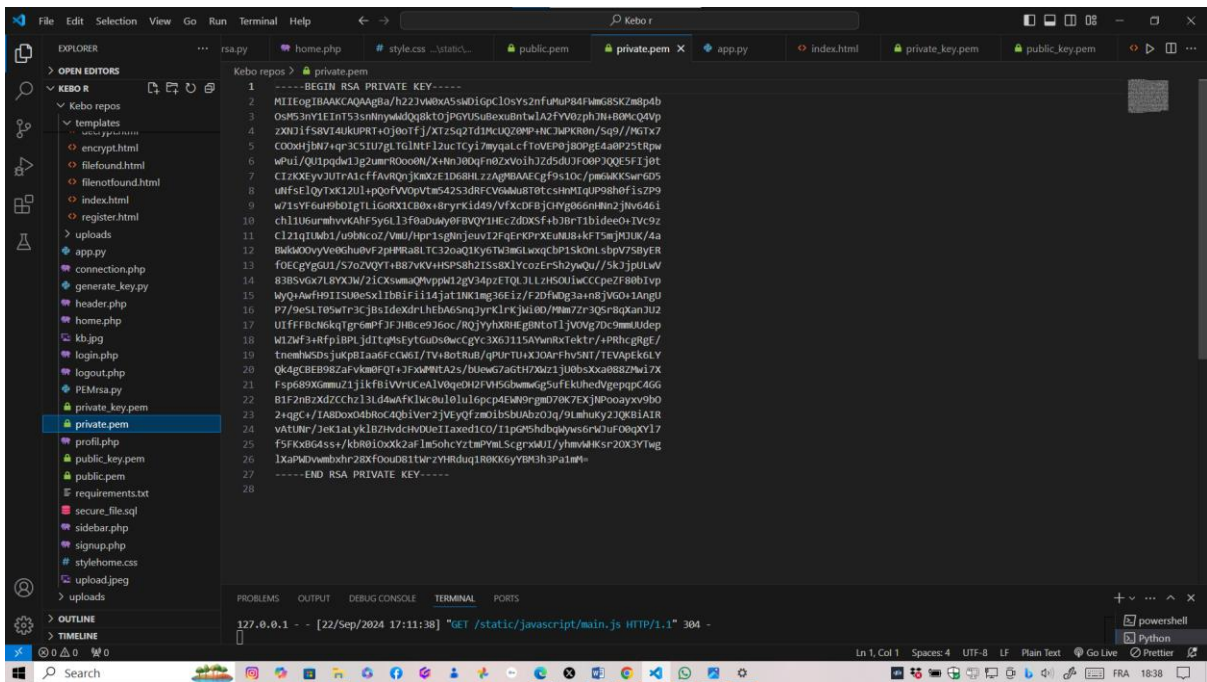
```

```

PS C:\xampp\htdocs\kebo r> & "C:\Users\gustave.kebo\AppData\Local\Programs\Python\Python310\python.exe" "c:\xampp\htdocs\kebo r\kebo repos/app.py"
* Serving Flask app 'app'
* Debug mode: on
* Restarting with stat
* Debugger is active!
* Debugger PIN: 136-779-375

```

Figure 21: Running the Python application from Visual studio code



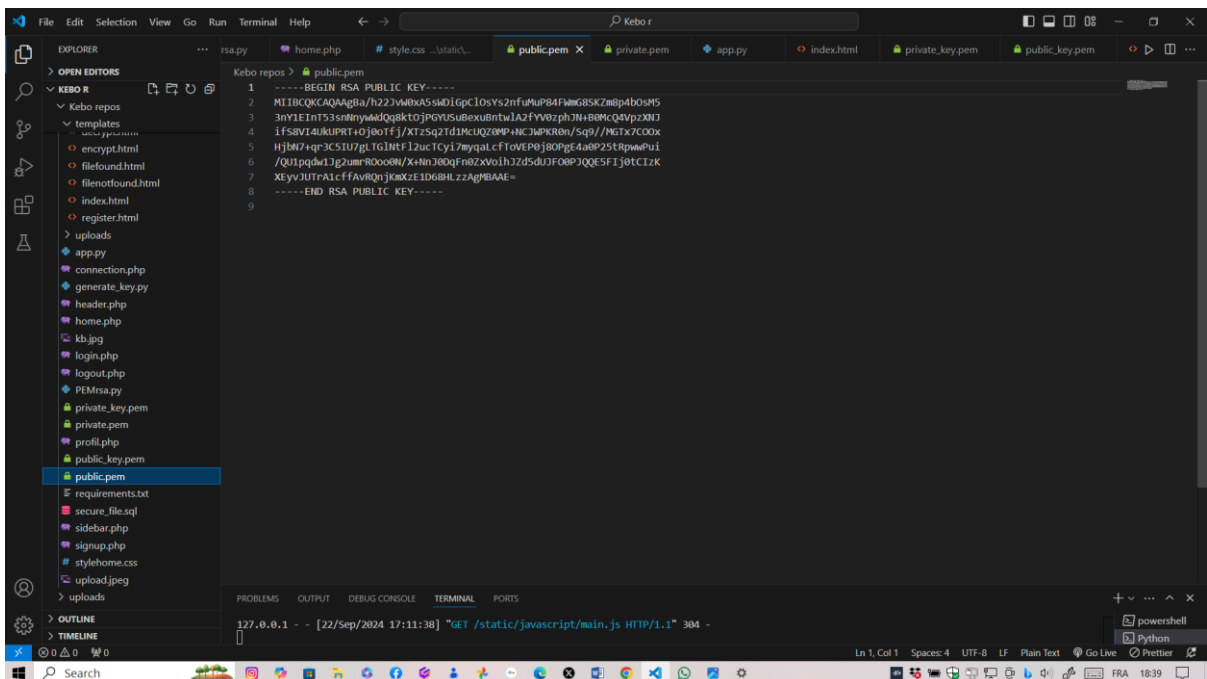
The screenshot shows the Visual Studio Code interface with the file explorer on the left and the code editor in the center. The file explorer shows a project named 'Kebo' with various files and folders. The code editor displays the content of 'private.pem', which is a PEM-formatted RSA private key. The key is displayed as a single line of text, starting with '-----BEGIN RSA PRIVATE KEY-----' and ending with '-----END RSA PRIVATE KEY-----'. The key is surrounded by a grey selection box. The terminal at the bottom shows the command 'GET /static/javascript/main.js HTTP/1.1' 304 -.

```

1 -----BEGIN RSA PRIVATE KEY-----
2 MIIEFgIBAAKCAQAgBa/h22jvWbXASwDIgpCLOsY2nfuP84FmG8SKZm8p4b
3 Osm53nY1E1T53snMnywMqQ8kT0jPGVUSuBexuBntwIA2FYV0zphJH+B8mCQ4Vp
4 2X0D1fS8V14UKUPRT+0j0oTfj/XT2sq2T1dMCUQZ0MP+HCIMPKR0N/Sq9//MGTx7
5 CO0xHjM7+qr3C5U17gLTG1ntF12ucTCy17myqALcFtoVEP0jB0PpE4a0P25TRp
6 wPu1/QU1pqdw1Jg2umR00o8M/X+HnJ0dQFno2Xvo1hJ2d5dUJF00P3Q0E5F1j0t
7 CTzKXeyvJUTRA1cFFAVRQnjKwzE1D68HlzzAgMBAE=
8 uNFsE1QyTK12U1+pQoFV0pVtw54253dRfCV6m8u8T0tcsHnM1qUP98Hf1sZP9
9 w715YF6uH9b01gTL1GoRk1CB0x+8ryrK1d49/vfXcDFBJGH9666nN2jNv6461
10 ch11U6umhvKAhF5y6L13f0aduY0FBVQY1HEczD0XSf+h3BRT1b1deeo+IV9z
11 C121q1Umb1/u9bnc0z/VmU/hpr1sglnjeuv12f-qErKPrXEAmu8+kFT5wJDUK/4a
12 BwK00yV99hu0VfzjPRAbLTC2oaQ21kyeTm36CkxqC0P15K0n1sIpV758yER
13 FOEGp9G1J570Z0Q7f4B87KvH8PS8h21SS8X1YeezEh32y0k//5k1jpdUw
14 B3BS6X7L8YK2N/21CkSmwQWypk12gV4p4ETOL3LLH8D0UwCCpeZf88b1Vp
15 WYQAwfH0TISUBesX1Ib8if114j3at1HK1mg36Ez/fzDf0g3aVn8jVGO+1AngU
16 P7/9eSLT95wTr3CjBS1DexdrlHEB65nqjYrK1KjM180/Mm0Zr3Q5r8QxanJ02
17 UIFFBCh6kqTgr8ePFJf3H8ce9J6oc/RQYyYbX8HEgBnt01jYVOVg7Dc9mUIdep
18 W1ZwF3+rFk1BPLjdt1qM5EYgUdSwecGyc3X6J115AYmArTektR/+PRHcgrGE/
19 tnmhWSDs+juKPIAa6FcW6I/TV+BotRub/qPurTUxJOArFhv5NT/TEVApE6LY
20 Qk4gCBE98ZafVkm8FQT+JF0a#ntA2s/bluwG7aGTH70z1J08sXcaB88Zw17X
21 Fsp69XGmmuZ1jKfB1VvUceAlV0q8H2FVH5GbwmmG5uFEkUhdVgepapCAGG
22 B1F2hZxdZCchz13ld4wAfk1wbu01u16pc4EwN9rgm078K7EXJPOoayxv9B0
23 2+qg+/IABD0x04Bt0c4Qb1Ver2jVeYQfzm01sbuUabz03q/9Lmhuky2JQkBIAR
24 vAUNr/3ek1aLyk1BZHvdcHvDUE1TaxedCO/11pG5Hdbqyus6WJuf0BqY17
25 TSfKX8Gass+/K8r10xXZaf1m5ohcyztmPvLScgrxMUI/yhmWkRS20X3YTwg
26 lXaP0vmbxhR28Xfo0d81tWzYHRduq1R0KXyYBh3P1aM+
27 -----END RSA PRIVATE KEY-----
28

```

Figure 22: Private Key generated



The screenshot shows the Visual Studio Code interface with the file explorer on the left and the code editor in the center. The file explorer shows a project named 'Kebo' with various files and folders. The code editor displays the content of 'public.pem', which is a PEM-formatted RSA public key. The key is displayed as a single line of text, starting with '-----BEGIN RSA PUBLIC KEY-----' and ending with '-----END RSA PUBLIC KEY-----'. The key is surrounded by a grey selection box. The terminal at the bottom shows the command 'GET /static/javascript/main.js HTTP/1.1' 304 -.

```

1 -----BEGIN RSA PUBLIC KEY-----
2 MIIBCCQCAQAgBa/h22jvWbXASwDIgpCLOsY2nfuP84FmG8SKZm8p4b0S4S
3 3nY1E1T53snMnywMqQ8kT0jPGVUSuBexuBntwIA2FYV0zphJH+B8mCQ4Vp2X0D
4 1fS8V14UKUPRT+0j0oTfj/XT2sq2T1dMCUQZ0MP+HCIMPKR0N/Sq9//MGTx7CO0x
5 HjM7+qr3C5U17gLTG1ntF12ucTCy17myqALcFtoVEP0jB0PpE4a0P25TRpWpu1
6 /QU1pqdw1Jg2umR00o8M/X+HnJ0dQFno2Xvo1hJ2d5dUJF00P3Q0E5F1j0tCTzK
7 XeyvJUTRA1cFFAVRQnjKwzE1D68HlzzAgMBAE=
8 -----END RSA PUBLIC KEY-----
9

```

Figure 23: Public Key generated

