Endovault - A Novel Efficient Document Management in Cloud Storage with Secure, Structured and Keyword Driven Retrieval System

A PROJECT REPORT

Submitted by

A.Buvaneshkumaar (202009009)
P.Vimal (202009253)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

MEPCO SCHLENK ENGINEERING COLLEGE, SIVAKASI
(An Autonomous Institution affiliated to Anna University Chennai)

APRIL 2024

BONAFIDE CERTIFICATE

Certified that this project report **"Endovault - A Novel Efficient Document Management in Cloud Storage with Secure,Structured and Keyword-Driven Retrieval System"** is the bonafide work of "**A.Buvaneshkumaar (202009009) , P.Vimal (202009253)"** who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

HEAD OF THE DEPARTMENT	**SUPERVISOR**
Dr. J. Angela jennifa sujana,	**Mrs. P. Swathika, M.E., (Ph.D.,)**
M. Tech., Ph.D.,	**Assistant Professor (Sr. G),**
Professor and Head,	Department of Artificial Intelligence
Department of Artificial Intelligence	and Data Science,
and Data Science,	Mepco Schlenk Engineering College,
Mepco Schlenk Engineering College,	Sivakasi
Sivakasi.	

Submitted for Viva-Voce Examination held at **Mepco Schlenk Engineering College (Autonomous), Sivakasi** on ______/______/20____.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

Our cutting-edge web application designed to revolutionize document management with heightened security measures and an intuitive interface and also incorporated an innovative feature for keyword-based document filtering and ranking.. The application offered a comprehensive suite of services to meet diverse user needs, allowing for the effortless upload of documents to a secure cloud storage area, ensuring accessibility and reliability. Robust encryption methodologies, including a hybrid encryption strategy, reinforced document confidentiality, while advanced hashing techniques verified document integrity.

A pivotal aspect of the application was its content originality checks, empowering users to validate the authenticity of document content. The addition of keyword-based document filtering and ranking further enhanced the user experience by providing a tailored approach to document retrieval and document's organization. This feature enabled users to efficiently locate and prioritize documents based on relevant keywords, streamlining information retrieval processes.

The culmination of these services resulted in a powerful tool for comprehensive document management and security. By encapsulating the development, implementation, and performance evaluation of these services, our work highlighted the application's effectiveness in safeguarding document integrity, elevating user experience, and responding to contemporary challenges in document management. The holistic approach presented in the application reflected a paradigm shift in document management, integrating advanced technologies and methodologies to enhance both functionality and security.

ACKNOWLEDGEMENT

First and foremost, we would like to thank the **LORD ALMIGHTY** for his abundant blessings that is showered upon our past, present and future successful endeavors.

We extend our sincere gratitude to our college management and principal **Dr. S. Arivazhagan M.E., Ph.D.,** for providing sufficient working environment such as systems and library facilities.

We would like to extend our heartfelt gratitude to our Head of the Artificial Intelligence and Data Science Department, **Dr.J.Angela Jennifa Sujana B.E, M.Tech, Ph.D.,** for giving us the golden opportunity to undertake the project of this nature and for her most valuable guidance.

We would also like to extend our gratitude to **Mrs.L.Prasika B.E, M.E, (Ph.D), Assistant Professor** , Department of Artificial Intelligence and Data Science, for being our project coordinator and directing us throughout our project.

We would also like to extend our gratitude and sincere thanks to **Mrs.P.Swathika B.Tech, M.E, (Ph.D),** Assistant Professor (Sr.Grade) Department of Artificial Intelligence and Data Science for being our project guide and for her moral support and suggestions. She has put her valuable experience and expertise in directing suggesting and supporting us throughout the project to bring our best.

Our sincere thanks to our revered faculty members, **lab technicians and beloved family and our friends** for their help at right time for making this project a successful one.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CMS	**C**entral **M**anagement **S**erver
CK	**C**loud **K**eepers
CC	**C**loud **C**ustomers
DN	**D**omain **N**ame
URD	**U**ser's **R**ecognition **ID**
DSF	Designated **D**ocument **S**torage **F**ield
APrk	**A**symmetric **Pr**ivate **k**ey
APuk	**A**symmetric **Pu**blic **k**ey
SMk	**S**ymmetric **M**ono **k**ey
C_{AES}	**C**ipher format **{AES}**
C_{RSA}	**C**ipher format **{RSA}**
Doc_{SCF}	**Doc**ument_{ **S**ecure **C**ypher **F**ormat **}**
HSM	**H**ash **S**torage **M**anager

CHAPTER 1

INTRODUCTION

1.1 AIM

To Develop an Document Management System with Structured,Verification and Keyword-Driven Retrieval functionalities in Cloud Storage Environment with enhanced Security Measures with AI enabled futuristic behaviours.

1.2 MOTIVATION

The motivation behind the development of our web application system from the crucial need for seamless, secure, and efficient document management in an increasingly dynamic digital landscape. As users, we often find ourselves constrained by the limitations of traditional file access, unable to retrieve needed documents anytime, anywhere. The paramount concern is not only accessibility but also the assurance of safety and security, safeguarding sensitive information from potential intruders. Moreover, the escalating volume of stored documents compounds the challenge of effective organization and retrieval. The inability to search for documents based on their content adds a layer of complexity to information retrieval, hindering productivity.

The motivation further deepens when considering the vulnerability of document integrity, the uncertainty of whether the content has been altered by unauthorized individuals. In response to these pressing concerns, our web application was conceived as a comprehensive solution. By allowing users to effortlessly upload documents to a secure cloud storage area, ensuring accessibility without compromising security, the application addresses the need for any time, anywhere access.

1.3 OUTLINE OF THE PROJECT

Our web application project encompasses a comprehensive array of services designed to address the intricate challenges of modern document management. At its core lies the crucial need for seamless and secure handling of digital files, a necessity stemming from the limitations of traditional methods. First and foremost, user verification through email validation serves as the initial layer of security, ensuring that only authorized individuals gain access to the system. This safeguards sensitive data but also establishes a foundation of trust for users. One of the key functionalities of the application is the extraction of frequency from files, enabling users to analyze and understand the distribution of data within their documents. This feature enhances data comprehension and aids in decision-making processes.Hybrid encryption further fortifies the security of our system by employing a combination of symmetric and asymmetric encryption techniques. This ensures that data remains protected while uploading and mitigating the risk of unauthorized access.

The ability to upload files seamlessly to a secure cloud storage area is paramount for users seeking anytime, anywhere access to their documents. Our application provides a user-friendly interface for file uploading, facilitating effortless document management.The File Storage feature offers users a centralized hub for organizing and accessing their uploaded documents, streamlining the document retrieval process and enhancing overall productivity. Hash generation for files adds an additional layer of security by creating unique fingerprints for each document. This enables users to verify the integrity of their files and detect any unauthorized alterations.

Lossless file compression service optimizes storage space without compromising the quality or integrity of the documents. This feature is particularly beneficial for users dealing with large volumes of data, as it reduces storage costs and improves efficiency.A comprehensive report generation and analysis tool empowers users to gain insights into their document management practices. By analyzing various metrics such as file usage patterns and access frequency, users can make informed decisions to optimize their workflow.Finally, AI-enabled features and interactive mediums enhance the user experience by facilitating seamless communication within the application and also for the learning perspective.

The web application addresses the need for efficient, secure, and intelligent document management solutions in today's digital landscape. By integrating a diverse range of functionalities, we aim to revolutionize the way users interact with and manage their digital assets, ultimately enhancing productivity and data security.

1.4 OVERVIEW OF THE PROJECT

The genesis of our innovative web application project emerged from a critical need in the contemporary digital landscape – the demand for a seamless, secure, and efficient document management system. Traditional modes of file access often confined users, limiting their ability to retrieve crucial documents at any time and from any location. However, our project was fueled by a more profound concern - not just the accessibility of documents but the assurance of their safety and security, guarding sensitive information against potential intruders. In response to these multifaceted challenges, our web application was conceived as a comprehensive solution. Our approach was centered around empowering users with a seamless and secure document management experience, transcending the conventional constraints of traditional file access. By facilitating the effortless uploading of a diverse range of documents to a secure cloud storage area, our system not only ensured ubiquitous accessibility but also placed a paramount emphasis on security, fulfilling the imperative need for anytime access.The incorporation of robust encryption methodologies, advanced hashing techniques, and meticulous content originality checks served as a concrete manifestation of our unwavering commitment to fortifying document confidentiality and integrity.In addition to these features, we implemented an AI interactive model for learning and content development, further enhancing the user experience. This model enabled users to interact with the system for learning purposes, as well as to facilitate content development in a more efficient and personalized manner.Through the "Document Upload" feature, users could seamlessly upload a variety of documents, fostering convenience and accessibility within a secure cloud storage environment. Our "Document Encryption" service employed state-of-the-art techniques, creating a formidable barrier against unauthorized access and potential data breaches. The integration of "Hybrid Encryption" combined both symmetric and asymmetric encryption, ensuring a multi-layered security approach for robust protection."Document Content Hashing" was implemented to facilitate efficient and secure document retrieval, significantly reducing the complexity of information searches. The "Check Originality" feature added an extra layer of security by verifying the authenticity of documents, protecting against unauthorized alterations. Leveraging "File Compression," our system optimized storage space and accelerated file transfers, enhancing overall system efficiency.Empowering users with control, the "File Management" functionality allowed for the convenient display and removal of files, contributing to an enhanced overall user experience. Lastly, "Report Generation" enhanced transparency and oversight by generating detailed reports about stored documents in the database, facilitating effective management and monitoring.

1.5 OBJECTIVE OF THE PROJECT

Our web application project addresses the need for streamlined document management in today's digital age. It offers a suite of features designed to enhance accessibility, security, and efficiency throughout the document lifecycle.

- To ensure that all stored documents remain unchanged and authentic over time through robust verification mechanisms.

- To implement encryption techniques to safeguard user’s data during transmission and storage, ensuring confidentiality and integrity.

- To optimize storage space without compromising the quality of documents, to ensure efficient resource utilization.

1.6 OUTCOME OF THE PROJECT

- Ensured the integrity of stored documents by detecting unauthorized alterations, assuring authenticity, and preventing data tampering or corruption

- Ensured confidentiality during transmission and storage, and preventing unauthorized access or data breaches, thus maintained the privacy and security of user documents.

- Implemented efficient compression techniques to reduce storage space to maintain document quality and focused on document maintenance by integrating automated features to ensure ongoing optimization and reliability of stored data.

1.7 ORGANIZATION OF THE PROJECT

The overview of the subject described in the underlying chapters is given below

Chapter 2: This section encompasses the prior research conducted preceding the development of the proposed system. Additionally, it outlines the methodology employed by the current system along with its limitations.

Chapter 3: Provides an overview of both the design of the current system and the proposed system, as well as the formulations utilized in the system's implementation.

Chapter 4: Describes the structure and elucidates the approach employed in the proposed system.

Chapter 5: Delves into the implementation of the proposed system, detailing the interaction among its formulated modules and elucidating how the methodology executes to yield the desired outcomes.

Chapter 6: Provides an overview of the dataset utilized and explains the experiments conducted to demonstrate the efficacy of the system.

Chapter 7: Summarizes the findings and outcomes of the system and also explores other potential avenues for future enhancements and improvements.

Chapter 8: Covers the social impact and applications of the proposed system.

1.8 SUMMARY

The web application project offers a comprehensive solution for modern document management challenges. It addresses the need for organized document storage and retrieval by providing users with a structured environment for systematic organization. The application ensures document integrity over time through robust verification mechanisms, including hash generation and comparison. Security is further enhanced through the use of hybrid encryption, which combines symmetric and asymmetric techniques for secure data transmission and storage.

Efficient information retrieval is facilitated through keyword-based document search, with results ranked based on relevance. Storage optimization is achieved through lossless compression, reducing document size without compromising quality. Users can gain valuable insights into their document usage patterns and access frequencies through comprehensive reporting functionalities.

Moreover, the application offers an engaging user experience with an AI-powered chat model, allowing users to interact naturally and access relevant information seamlessly. Overall, the web application aims to revolutionize document management practices, providing users with the tools they need to efficiently and securely manage their digital documents in today's dynamic digital landscape.

CHAPTER - 2

LITERATURE SURVEY

2.1 Secure File Storage on Cloud Computing Using Cryptographic Algorithm - Shreya Sambhaji Ranadive , Harshada Sanjay Sawant , Jayesh Ekanath Pinjarkar - July , 2017

Shreya Sambhaji Ranadive , Harshada Sanjay Sawant , Jayesh Ekanath Pinjarkar [13] proposed an approach and it depicted that cloud computing is a rapidly growing technology that offers scalability and adaptability for computing services. However, security issues are crucial for its development, as cloud computing shares disseminated resources through networks. This research explored data security in cloud computing by creating unique keys and encryption using them. This discussesed the concept of cryptography, which is defined as "an art of writing a secret law." This research also reviewed the literature on advanced encryption algorithms, focusing on the elliptic curve cryptography (AES) algorithm. AES requires medium memory size and is considered excellent in terms of security. It consumes less time for encryption than RSA and DSA, making it better than RSA and DSA. This work concludes that AES algorithm provides better security than RSA and DSA due to its lower time for encryption and decryption.

2.2 Achieving Efficient Conjunctive Keyword Searches over Encrypted Data - Lucas Ballard, Seny Kamara, and Fabian Monrose - October , 2018

Lucas Ballard, Seny Kamara, and Fabian Monrose [14] proposed a method and it presented two provably secure and efficient schemes for performing conjunctive keyword searches over symmetrically encrypted data. The first scheme is based on Shamir Secret Sharing and provides the most efficient search technique in this context to date. Although the size of its trapdoors is linear in the number of documents being searched, the overhead remains reasonable in practice. To address this limitation, an alternative based on bilinear pairings is provided that yields constant size trapdoors. This latter construction is not only asymptotically more efficient than previous secure conjunctive keyword search schemes in the symmetric setting but incurs significantly less storage overhead. Additionally, unlike most previous work, the constructions are proven secure in the standard model.Remote and untrusted storage systems allow clients with limited resources to store and distribute large amounts of data at low cost. However, encryption restricts a client's ability to selectively access segments of their data, especially when they wish to retrieve specific content related to a given keyword.

2.3 Lossless data compression techniques and their performance - Komal Sharma, Kunal Gupta - July , 2018

Komal Sharma,Kunal Gupta proposed an approach [15] and it explored the lossless data compression techniques and assesses their performance in various contexts. It provides an in-depth analysis of compression algorithms, emphasizing their ability to reduce data size without any loss of information. The research evaluates the effectiveness of well-established methods such as Huffman coding, Lempel-Ziv algorithms, and Burrows-Wheeler transform, while also considering newer approaches. This discussesed the performance metrics such as compression ratio, speed, and memory requirements, offering a comparative study to aid in selecting the most suitable technique for specific applications. Advantages of this work include its comprehensive coverage of both traditional and contemporary compression methods, serving as a valuable resource for researchers, data scientists, and practitioners in need of informed decision-making on compression strategies.

2.4 Lossless data compression technique with encryption based approach - September , 2018

Komal Sharma; Kunal Gupta [16] proposed an approach and it presented an approach to lossless data compression by integrating encryption techniques into the compression process. This method aims to not only reduce data size without loss of information but also enhance the security of compressed data through encryption. This research explores the synergy between compression and encryption algorithms, evaluating their combined performance in terms of data reduction and confidentiality. Advantages of this work include its innovative integration of compression and encryption, providing a dual-layered solution for efficient data management and enhanced security. This approach is particularly relevant in scenarios where both compression and encryption are crucial, such as secure transmission or storage of sensitive information. However, limitations may arise in terms of computational overhead due to the combined processes, potentially impacting the speed of compression. Additionally, careful consideration is needed to strike a balance between compression ratio and encryption strength, as overly aggressive compression might compromise security. Continuous evaluation and adaptation of this technique is to evolving encryption standards and compression algorithms are also essential for sustained effectiveness.

2.5 Structured Document Model and Its Secure Access Control in Cloud Computing - Jinbo Xiong, Zhiqiang Yao,Jun Ma, Ximeng Liu, Qi Li - October,2019

Jinbo Xiong, Zhiqiang Yao, Jun Ma, Ximeng Liu, Qi Li [17] proposed an approach and it depicted a new structured document model that addresses the complex creation, access, and management issues of structured documents in cloud computing. The model leverages multilevel security and employs identity-based encryption to protect sensitive and privacy information. The model is suitable for the special live characteristics of structured documents, such as multi-user participation, multimedia interaction, and multilevel security. However, most existing document models become inapplicable in cloud computing due to static content, document-centricity, simple structure, and security levels.It focuses on the problems faced by structured documents in cloud computing, such as the need for secure access control schemes. It proposed a fine-grained access control scheme to protect sensitive and privacy information, ensuring that only authorized users can access the corresponding elements of the structured document. It highlights the importance of a secure access control scheme for structured documents in cloud computing environments.

2.6 A User-based Document Management Mechanism in Cloud - Guozhen Shi , Mang Su , Fenghua Li , Jiapeng Lou, Qiong Huang - November , 2019

Guozhen Shi , Mang Su , Fenghua Li , Jiapeng Lou, Qiong Huang [18] proposed an approach and depicted a user-centric approach for interactive and efficient document management within a cloud storage environment. The mechanism aims to safeguard data and counter replay attacks in cloud storage while ensuring the confidentiality and integrity of documents. The focus is on addressing challenges related to managing the uncertainty and variability of users in the cloud, emphasizing the need for robust security measures. This work also acknowledges the complexities associated with ensuring complete security and integrity of documents in the cloud and highlights challenges in implementing multi-factor access control. It proposed a user-based document management mechanism, the research contributes to enhancing security and efficiency in cloud-based document storage and access.This addressed the issues related to diverse user behaviors, access patterns, and dynamic user requirements within the cloud storage environment.

2.7 Secure and Efficient Data Retrieval in Cloud Computing - Anuradha N. M,G. A. Patil - November , 2019

Anuradha N. M,G. A. Pati [19] proposed an approach and it introduced an efficient data retrieval system with advanced accessing strategies within a cloud storage environment. The system is designed to enhance usability and improve file retrieval accuracy, particularly beneficial for collaborative work scenarios. Emphasizing robust security measures, this research addresses concerns related to the confidentiality of user data during retrieval, aiming to mitigate the risk of privacy leaks when accessing plaintext. However, the research acknowledges certain challenges, including the absence of a strong authorization mechanism, the potential susceptibility to statistical attacks on keywords, and the complexity of key management, which may pose operational challenges. Despite these considerations, this work contributes to the development of secure and efficient data retrieval systems in cloud computing, offering a balance between usability and privacy protection.

2.8 Secure Ranked Keyword Search over Encrypted Cloud Data - Cong Wang, Ning Cao, Jin Li, Kui Ren, and Wenjing Lou - November, 2019

Cong Wang, Ning Cao, Jin Li, Kui Ren, and Wenjing Lou proposed an approach[20] which mainly focused that the cloud computing has become increasingly prevalent, with sensitive information being centralized into the cloud. To protect data privacy, sensitive data must be encrypted before outsourcing, making effective data utilization a challenging task. Traditional searchable encryption schemes support only boolean search, without capturing any relevance of data files. This approach suffers from two main drawbacks when directly applied in the context of Cloud Computing: users must postprocess every retrieved file to find ones most matching their interest, and retrieving all files containing the queried keyword incurs unnecessary network traffic. It defines and solves the problem of effective yet secure ranked keyword search over encrypted cloud data. Ranked search greatly enhances system usability by returning matching files in a ranked order regarding certain relevance criteria, making one step closer to the practical deployment of privacy-preserving data hosting services in Cloud Computing. This proposed works defines "as-strong-as-possible" security guarantee compared to previous SSE schemes, while correctly realizing the goal of ranked keyword search.

2.9 Efficient Multi-Keyword Ranked Query on Encrypted Data in the Cloud - Zhiyong Xu1, Wansheng Kang, Ruixuan Li, KinChoong Yow, and Cheng-Zhong Xu - July , 2019

Zhiyong Xu1, Wansheng Kang, Ruixuan Li, KinChoong Yow, and Cheng-Zhong Xu [21] proposed an appropach and it focused particularly for large-scale data management systems. However, data owners must outsource their data onto public cloud servers, which are not within their trusted domains. This poses a significant challenge in terms of data security and privacy. To prevent information disclosure, sensitive data must be encrypted before uploading onto cloud servers, making plain text keyword queries impossible. As the total amount of data stored in public clouds accumulates exponentially, it is challenging to support efficient keyword-based queries and rank matching results on encrypted data. Most current works only consider single keyword queries without appropriate ranking schemes. A novel approach, called Multi-Keyword Ranked Query (MKQE), addresses these issues by introducing new trapdoor generation and scoring algorithms to make in-order query results. Keyword access frequency is considered to select an adequate matching file set.

2.10 Secure File Storage on Cloud Using Cryptography - Nandini K , Faisal. S , Shailendra B , Shree Vallabha S , Pavan B - December , 2019

Nandini K , Faisal. S , Shailendra B , Shree Vallabha S , Pavan B [22] proposed an approach and it discussed the use of cryptography and image steganography to secure file storage on the cloud. Cloud computing is used in various sectors, including business, colleges, and universities, to store large amounts of data. However, transferring secure information through the web has become difficult due to security concerns. To address this, symmetrical key cryptography algorithmic rules and steganography techniques are used. The proposed system uses AES, blowfish, RC6, and 3DES algorithms to provide block-wise security to data. Key data contains encryption of a part of the file using the algorithmic rule and key. The file is split into eight components, and all components are encrypted simultaneously using multithreading techniques. Encoding Keys is inserted into a cloud image using LSB technique. Steganography images are sent to a valid receiver using email. For file secret writing purposes, the reverse method of cryptography is applied.

2.11 Secure Hashing Algorithms and Their Comparison - Rohit,Sagar Kamra,Manu sharma, Alka Leekha - March , 2019

Rohit,Sagar Kamra,Manu sharma,Alka Leekha [23] proposed a method and it explored various secure hashing algorithms, analyzing their strengths and weaknesses. It provides a comprehensive overview of prominent hashing algorithms such as MD5, SHA-1, SHA-256, and SHA-3, evaluating their performance, security features, and applications. The research aims to assist in selecting the most suitable hashing algorithm based on specific security requirements. Advantages of this approach include its systematic approach to algorithm comparison, offering valuable insights into algorithmic characteristics and their implications for secure data hashing. However, limitations arised from the dynamic nature of cryptographic research, as new vulnerabilities and attacks can emerge over time. Additionally, the research's effectiveness depends on the comprehensiveness of the algorithms covered and the relevance of its findings to real-world security scenarios.

2.12 Implementation of Enhanced Secure Hash Algorithm Towards a Secured Web Portal - Froilan E. De Guzman; Bobby D. Gerardo; Ruji P. Medina - September , 2019

Froilan E. De Guzman; Bobby D. Gerardo; Ruji P. Medina [24] propsed a method and it focused on the practical application of an enhanced secure hash algorithm in the context of securing a web portal. It likely outlines the implementation details, performance metrics, and security enhancements brought about by the proposed algorithm. The research addressed the challenges specific to web portal security and demonstrate how the enhanced secure hash algorithm contributes to mitigating potential risks such as unauthorized access, data tampering, or other security threats. Advantages of this research includes a tangible and applicable solution to web portal security issues, with potential improvements in data integrity and confidentiality. However, limitations arised from the scalability of the proposed algorithm, compatibility with existing systems, and the real-world feasibility of its implementation. Additionally, the effectiveness of this solution depend on the specific use case and the evolving landscape of cybersecurity threats.

2.13 A Secured Cryptographic Hashing Algorithm - Rakesh Mohanty , Niharjyoti Sarangi, Sukant Kumar Bishi - November , 2019

Rakesh Mohanty , Niharjyoti Sarangi, Sukant Kumar Bishi [25] introduced an approach and it depicted the secure cryptographic hashing algorithm designed to address the vulnerabilities present in existing hash functions. The algorithm employs advanced techniques to ensure data integrity and confidentiality by generating a fixed-size hash value that is unique to each input. Its security is bolstered through a combination of key factors such as resistance to collision attacks, robustness against preimage attacks, and efficient computational performance. The algorithm demonstrates a high level of resistance to various cryptographic attacks, making it suitable for safeguarding sensitive information in digital applications.

Advantages of this algorithm include its robustness against common cryptographic attacks, efficient computational performance, and suitability for diverse applications requiring secure hashing. Furthermore, the algorithm exhibits resistance to collision attacks, ensuring the integrity of hashed data. However, limitations included the potential vulnerabilities that could emerge over time with advancements in cryptanalysis, requiring periodic updates and evaluations to maintain its security posture.

2.14 An Analysis on Different Document Keyword Extraction Methods - M.G. Thushara; S. Anjali, M. Meera Nai - October , 2019

M.G. Thushara; S. Anjali, M. Meera Nai [26] proposed an approach and it conducted a comprehensive analysis of various document keyword extraction methods, aiming to provide insights into their effectiveness and applicability. It scrutinizes diverse techniques, including statistical, linguistic, and machine learning-based approaches, evaluating their performance in extracting meaningful keywords from documents.This research systematically compares the strengths and weaknesses of each method, considering factors such as accuracy, scalability, and adaptability to different types of content. Advantages of this research lie in its thorough examination of multiple keyword extraction methodologies, offering a valuable resource for researchers, practitioners, and developers seeking to implement or improve document analysis systems. However, limitations may arise in the rapidly evolving field of natural language processing, and the effectiveness of the methods may vary depending on the nature and complexity of the documents being analyzed.

2.15 Keyword Extraction: A Modern Perspective - Tadashi Nomoto - September , 2019

Tadashi Nomoto [27] introduced an approach of modern perspective on keyword extraction, offering a comprehensive analysis of contemporary techniques and methodologies in the field. It explores recent advancements in natural language processing, machine learning, and data analytics to identify and extract keywords from textual data. The research delves into the challenges and opportunities presented by diverse document types, languages, and contextual nuances, provides an understanding of the complexities involved in keyword extraction. Advantages of this research include its up-to-date insights into cutting-edge technologies and methodologies, making it a valuable resource for researchers and practitioners interested in optimizing keyword extraction processes. However, limitations arised in the dynamic nature of the field, as new technologies and approaches continue to emerge, necessitating ongoing updates to maintain the research's relevance. Additionally, the effectiveness of the discussed techniques vary based on the specific characteristics of the data being analyzed, requiring users to consider the applicability of the methods to their particular use cases.

2.16 An Enchanced Document Verification Strategy using Digital Identification - Madura Rajapashea , Muammar Adnanb , Ashen Dissanayakac, Dasith Guneratned , Kavinga Abeywardanee - December , 2020

Madura Rajapashea , Muammar Adnanb , Ashen Dissanayakac , Dasith Guneratned , Kavinga Abeywardanee [28] introduced an enhanced document verification strategy utilizing a digital signature-based mechanism. The system aims to deliver fast and reliable verification processes with heightened security, presenting a comprehensive solution that promotes transparency and integrity. Notably, the system provides support for multi-format documents, enhancing its versatility. However, this research acknowledges the limitations such as the system's limited support for single-page documents, potential issues in the scalability and efficiency of decentralized storage, and challenges associated with content extraction. Despite these concerns, this research contributes to advancing document verification mechanisms, offering a multifaceted solution with improved security and efficiency, particularly for multi-format documents.

2.17 Development of a Document Management System for Private Cloud Environment - Chia Hung Kao, Shin Tzu Liu - March ,2020

Chia Hung Kao, Shin Tzu Liu [29] proposed an approach and it focused on the efficient management and maintenance of documents within a private cloud setting. The system is designed to facilitate efficient document sharing and collaboration, ensuring consistency across the stored data, and providing support for heterogeneous devices. While emphasizing user-friendly features, this research recognizes the challenges such as the requirement for specific IT knowledge to operate the data storage system and potential conflicts in handling large volumes of data. Despite these challenges, this research contributes to the development of a document management system tailored for private cloud environments, aiming to enhance collaboration and consistency while addressing specific usability and scalability considerations.This likely involves a balance between providing user-friendly features and ensuring the system can scale to meet the demands of growing data and user bases.

2.18 Comparison of Lossless Data Compression Techniques - **Athira Gopinath,M. Ravishankar** - October , 2020

Athira Gopinath,M. Ravishankar [30] proposed a methodology which involved a systematic examination of multiple lossless data compression techniques, considering factors such as compression ratios, processing speeds, and the adaptability of each method to different types of data. This research likely involves the use of benchmark datasets to perform rigorous testing and evaluation, comparing the performance of various compression algorithms in real-world scenarios.This involve assessing how well each method handles different types of data, such as text, images, or binary files. The evaluation could also extend to examining the impact of compression on computational resources, including processing time and memory usage. This involves considering factors like the speed of compression versus the achieved compression ratio and the computational complexity of each algorithm.This research is poised to contribute valuable insights into the landscape of lossless data compression, offering a comparative analysis that aids in understanding the strengths and limitations of different techniques.

2.19 Efficient File Upload and Mutli-Keyword Search over Encrypted Cloud Data - **M. Bhavya; C.N. Pushpa J. Thriveni K.R. Venugopal** - December , 2020

M. Bhavya; C.N. Pushpa J. Thriveni K.R. Venugopal [31] proposesd an efficient solution for secure file upload and multi-keyword search over encrypted cloud data. This system focuses on preserving data confidentiality while allowing users to upload and search for files without revealing their content to the cloud service provider. It employs cryptographic techniques such as homomorphic encryption to enable computation on encrypted data, ensuring that sensitive information remains private during both upload and search operations. This approach also integrates an efficient indexing mechanism for multi-keyword search, facilitating quick and accurate retrieval of relevant encrypted files without compromising security.Advantages of this approach include robust data confidentiality through encryption, efficient file upload procedures, and a fast and secure multi-keyword search capability. This system allows users to harness the benefits of cloud storage and retrieval while maintaining control over their data's privacy. However, limitations arised in terms of the computational overhead associated with homomorphic encryption, potentially impacting system performance.The practicality and scalability of this solution is influenced by factors such as the size of the dataset.

2.20 Development of online cloud storage technology - Shan-Hua Zou; Ning-Sheng Fang; Wei-Jie Gao - September , 2020

Shan-Hua Zou; Ning-Sheng Fang; Wei-Jie Gao [32] proposed an approach and it explored the advancements in online cloud storage technology, aiming to provide a comprehensive overview of the current state of the field. It delves into various aspects such as storage infrastructure, security protocols, data accessibility, and performance optimization in the context of cloud-based storage solutions. This research addressed the emerging challenges and proposes innovative approaches to enhance the efficiency and reliability of online cloud storage. Advantages of this research include its thorough examination of contemporary technologies, offering insights into the potential for improved storage capabilities, security measures, and user accessibility. However, limitations arised in the rapid evolution of cloud technologies, making it challenging to keep the information up-to-date. Additionally, this research's applicability is depend on the specific cloud storage platforms and technologies.

2.21 Authorized Private Keyword Search over Encrypted Data in Cloud Computing - Ming Li ,Shucheng Yu, Ning Cao, Wenjing Lou - September , 2021

Ming Li ,Shucheng Yu, Ning Cao, Wenjing Lou [33] proposed an approach and it focused in cloud computing, clients often outsource their data to cloud storage servers to reduce management costs. However, these servers cannot be fully trusted in protecting sensitive personal information. Encryption is a promising way to protect the confidentiality of outsourced data, but it also introduces difficulty in performing effective searches over encrypted information. Most existing works do not support efficient searches with complex query conditions, and care needs to be taken when using them due to potential privacy leakages about the data owners to the data users or the cloud server. It uses online Personal Health Record (PHR) as a case study to show the necessity of search capability authorization that reduces privacy exposure resulting from search results. They establish a scalable framework for Authorized Private Keyword Search (APKS) over encrypted cloud data and propose two novel solutions based on Hierarchical Predicate Encryption (HPE). This solutions enable efficient multi-dimensional keyword searches with range query, allow delegation and revocation of search capabilities, and enhance query privacy by hiding users's query keywords against the server.

2.22 Security and Security and Privacy Privacy Privacy Issues in Cloud Computing - Jaydip Sen - October , 2021

Jaydip Sen [34] proposed an approach and it depicted that the cloud computing is transforming the way IT is consumed and managed, offering improved cost efficiencies, accelerated innovation, faster time-to-market, and the ability to scale applications on demand. However, legal/contractual, economic, service quality, interoperability, security, and privacy issues still pose significant challenges. This research described various service and deployment models of cloud computing and identifies major challenges. In particular, three critical challenges: regulatory, security, and privacy issues in cloud computing. Solutions to mitigate these challenges are also proposed along with a brief presentation on future trends in cloud computing deployment. Ensuring the confidentiality of data during transmission and storage is critical. Proper encryption mechanisms must be employed to safeguard data from unauthorized access.

2.23 Practical Multi-Keyword Ranked Search With Access Control Over Encrypted Cloud Data - Iayi Li, Jianfeng Ma, Yinbin Miao, Ruikang Yang, Ximeng Liu, Kim-Kwang Raymond Choo - 3, July-September 2022

Iayi Li, Jianfeng Ma, Yinbin Miao, Ruikang Yang, Ximeng Liu, Kim-Kwang Raymond Choo [35] proposed an approach and it focused on enhancing the security and efficiency of cloud data search through the implementation of multi-keyword ranked search strategies with access control over encrypted content. This approach employs Fine-grained multi-keyword search (FMS) schemes, incorporating Cipher-Policy Attribute-Based Encryption (CP-ABE) and Cipher-Policy Attribute-Based Keyword Search (CP-ABKS). The primary goals include improving search accuracy while addressing vulnerabilities, known-sample attacks, and privacy implications. However, challenges such as computation and time-consuming access control methods, as well as the complexity associated with these mechanisms, are acknowledged. This research strives to strike a balance between bolstering security measures and overcoming the potential drawbacks, contributing to the advancement of privacy-preserving techniques in cloud data management.

2.24 Privacy-Preserving Multi-keyword Ranked Search over Encrypted Cloud Data - Ning Cao , Cong Wang, Ming Li, Kui Ren, and Wenjing Lou - September , 2022

Ning Cao , Cong Wang, Ming Li, Kui Ren, and Wenjing Lou discussed[36] that the importance of privacy-preserving multi-keyword ranked search over encrypted cloud data

(MRSE) in the context of cloud computing. This research depicted that for protecting data privacy, sensitive data must be encrypted before outsourcing to the cloud, which replaces traditional data utilization based on plaintext keyword search. They propose a set of strict privacy requirements for such a secure cloud data utilization system and propose two significantly improved MRSE schemes to achieve various stringent privacy requirements in two different threat models. This research also conducted a thorough analysis of the privacy and efficiency guarantees of the schemes and show that experiments on a real-world dataset show that the schemes introduced low overhead on computation and communication. This research concluded that the MRSE can be a valuable tool for storing and retrieving sensitive data in the cloud, but it requires careful consideration of performance, system usability, and scalability.

2.25 Modern Lossless Compression Techniques: Review, Comparison and Analysis - Apoorv Gupta , Aman Bansal , Vidhi Khanduja - November , 2022

Apoorv Gupta , Aman Bansal , Vidhi Khanduja [37] proposed an approach and it depicted the the size and amount of data shared over the internet increasing daily. To facilitate fast and efficient sharing of data over the network, efficient compression algorithms are required. This research discussesed the algorithms of widely used traditional and modern compression techniques, comparing them on the Silesia corpus. The researchers found that PPMonstr provides the best compression ratio, while Deflate is the fastest algorithm in terms of compression and decompression speed but provides a low compression ratio. Bzip2 and PPMd provide moderate compression speed and good compression ratio, making them suitable for applications dependent on both compression ratio and speed. Experimental results indicate that LZMA is an ideal algorithm for applications requiring high compression ratio along with fast decompression speed, as it provides moderate decompression speed coupled with high compression ratio. This research highlights the need for efficient compression algorithms to reduce network bandwidth and storage space while ensuring efficient data sharing.

CHAPTER 3

SYSTEM DESIGN

3.1 OVERVIEW

In this chapter, the system design as well as the modules involved in the proposed system is discussed.

3.2 SYSTEM ARCHITECTURE DESIGN

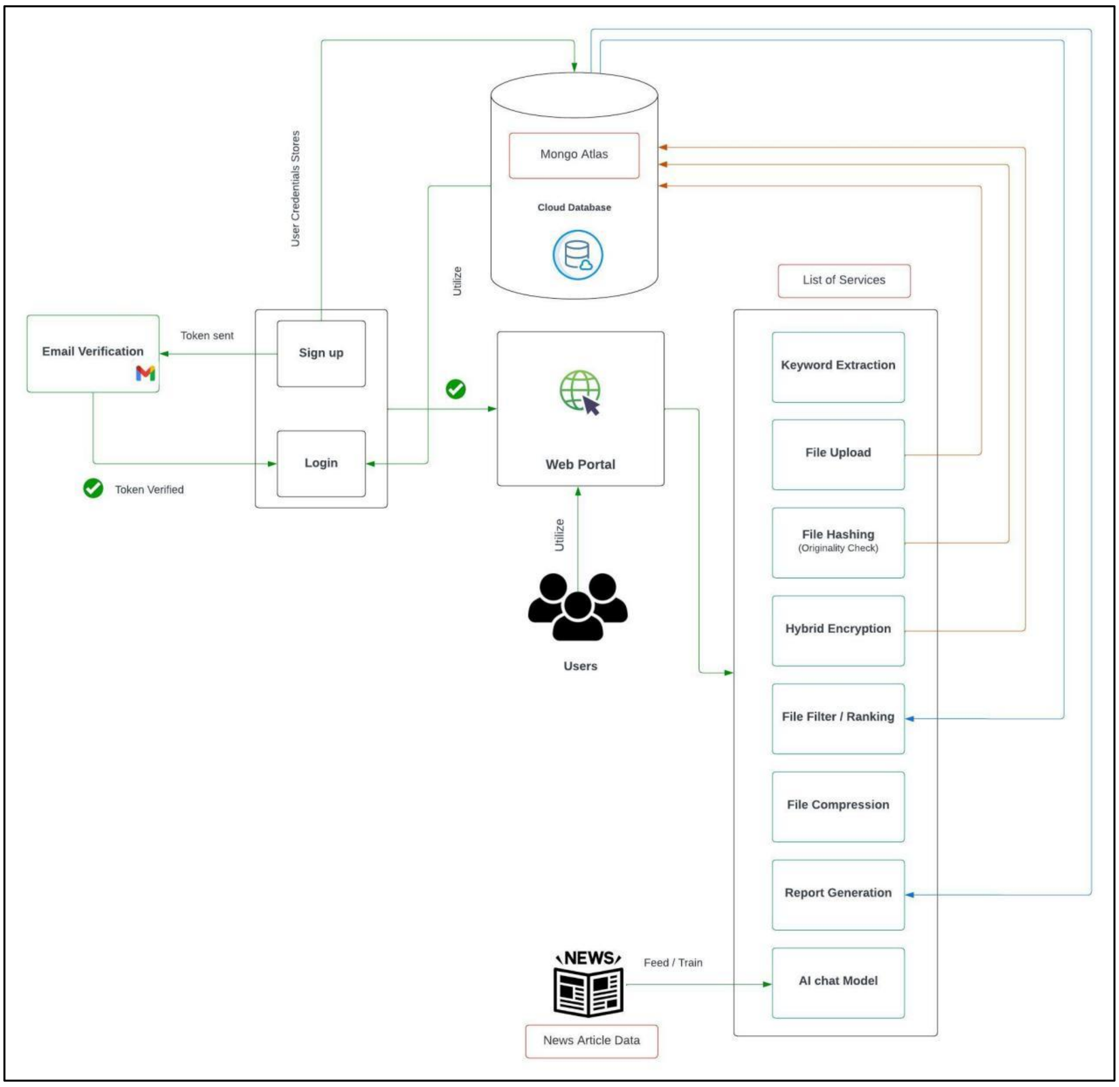


Figure 3.1 - Overall System Design

Our proposed work is an innovative cloud-based document management system designed to streamline document handling processes with ease and efficiency which is illustrated in Figure 3.1. Its core functionality revolves around a user-friendly web portal, offering seamless uploading, secure storage, and convenient access to documents from any device with internet connectivity. At the heart of Our proposed work's operation are several integrated services meticulously orchestrated to ensure optimal performance and reliability. Through its robust architecture, It empowers users with the ability to organize, search, and retrieve documents effortlessly, enhancing productivity and fostering collaboration across teams. With Our proposed work, businesses can entrust their document management needs to a dependable solution engineered for modern workflows.

3.3 MODULES DESCRIPTION

The modules in the proposed system are:

1. User Registration and Verification
2. Strucuted Document Management
3. Document Digital Signing
4. Document Encryption and Decryption
5. Document Upload
6. Keywords Extraction and Retrieval & Ranking
7. File Compression
8. AI Interactive Model

3.3.1 User Registration and Verification:

The sequence of steps adhered for user registration and verification is as follows:

i) Submission of Email and Password
ii) Email Verification
iii) User activation in Cloud Storage Environment
iv) Secure Identity Verification

i) Submission of Email and Password

The user journey begins with account creation. Here, users provide their email address and set a password. This initial step establishes the foundation for user identity within the system, enabling them to interact with its features and functionalities going forward.

ii) Email Verification

To safeguard the system and ensure users have control over their accounts, an email verification process follows registration. A verification email is sent to the provided address. This email contains a unique link or code that the user must click or enter to confirm ownership of the email address. This step prevents unauthorized registrations and guarantees legitimate users have access to their accounts.

iii) User activation in Cloud Storage Environment

Following successful email verification, the user's account is activated within the cloud storage environment. This critical step unlocks the full potential of the system. With their account active, users gain authorization to upload, store, and manage their documents securely. This activation essentially grants them the keys to their own digital vault within the cloud.

iv) Secure Identity Verification (User Recognition ID - URD)

Even after email verification and account activation, ensuring user identity remains paramount for system security. This is where the User Recognition ID (URD) comes into play. The URD acts as a secure identity verification mechanism within the cloud storage environment. It grants access to cloud services only to authorized users, significantly reducing the risk of unauthorized logins and protecting the integrity of the system. Think of the URD as a digital key that unlocks the user's designated space within the cloud storage, safeguarding valuable documents and information.

v) Storage Maintenance and Security

The foundation of secure cloud storage lies in the user registration and verification process. This rigorous approach establishes a trusted user base, acting as the first line of defense against unauthorized access. By requiring email verification and setting up User Recognition IDs (URDs), the system ensures that only authorized individuals can interact with and store data. This significantly reduces the risk of malicious attempts, safeguarding the integrity of the cloud storage environment and keeping the valuable information protected.

The user registration and verification process is divided into two methods , such as :

A) Token - Based Authentication

B) User Management

each serving the dual purposes of security enhancement and administrative management.

A) Token-Based Authentication

Token-based authentication encompasses various methods and techniques , such as :

i) Enhanced Security Measure

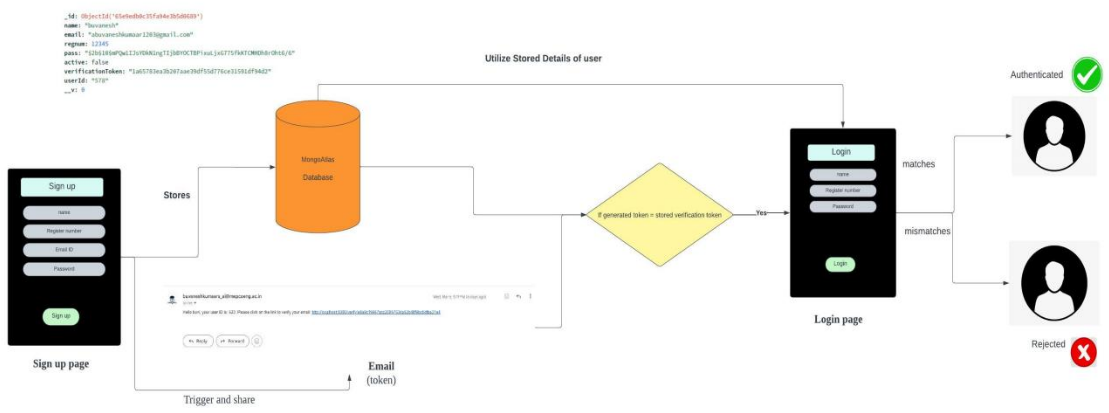
Our security measures are beyond the traditional username and password combinations. To further bolster protection, we leverage token-based authentication. This approach ditches storing passwords directly on the system. Instead, upon successful login or email verification, a unique token is dynamically generated which is illustrated in Figure 3.2. This token acts as a temporary key, granting access to the cloud storage for a predetermined timeframe. This significantly reduces the risk of unauthorized access, even if a hacker were to intercept login credentials. It offering an extra layer of security for the valuable data.

ii) Minimizing Risk

The implementation of token-based authentication significantly reduces the risk associated with storing sensitive information. Unlike traditional methods that store passwords directly, this approach prioritizes user security. Even in the unfortunate event of a system breach, hackers would not gain access to actual passwords. Instead, they would encounter encrypted tokens that expire after a set period, rendering them useless. This multi-layered approach minimizes the potential for unauthorized access and safeguards the integrity of user accounts within the cloud storage environment.

iii) Temporary Access Keys

Our system leverages tokens as temporary access keys, adding another shield against unauthorized access. These tokens function like digital keys, but unlike physical keys that can be lost or stolen, tokens are temporary and can be revoked at any time. This means even if someone were to intercept a token, it would only grant access for a limited duration, significantly reducing the window of vulnerability. This approach strengthens the overall security posture of the system and ensures the data remains protected within the cloud storage environment.



```
_id: ObjectId('65e9edb0c35fa94e3b5d0689')
name: "buvanesh"
email: "abuvaneshkumaar1203@gmail.com"
regnum: 12345
pass: "$2b$10$mPQwlIJsYDkNlngTIjbBYOCTBPixuLjxG775fkKTCMHDh8rOhtG/6"
active: false
verificationToken: "1a65783ea3b207aae39df55d776ce31591df94d2"
userId: "578"
__v: 0
```

Figure 3.2 - Registration and Login

B) User Management

The user management employs various techniques , such as :

i) Seamless Account Creation

At the core of our system lies a user management system designed for seamless account creation. We have prioritized a smooth registration process that incorporates email verification. This ensures a positive user experience while upholding security.The verification step safeguards against illegitimate registrations and guarantees that only authorized individuals can access the system and its functionalities which is illustrated in Figure 3.2. This streamlined approach provides a secure and user-friendly gateway to the cloud storage experience.

ii) Controlled Access

The user account functionalities extend beyond simple logins.This work incorporate a granular access control system that grants users controlled access to specific features within the cloud storage environment. This approach enhances overall security by ensuring users only interact with functionalities relevant to their assigned roles and permissions. users are equipped with unique keys, granting access to designated sections. This dual functionality ensures the security of sensitive information while enabling users to explore and utilize the features they require. This controlled access system fosters a secure environment and personalizes the user experience within the cloud storage system.

iii) Comprehensive User Information

The backbone of our secure document management platform lies in comprehensive user profiles. Each profile acts as a digital identity card, containing essential information such as a unique user ID, name, email address, and securely encrypted passwords. This data is not only crucial for user authentication but also allows for personalized interactions within the platform. Imagine a system that remembers the name and preferences, offering a tailored experience that caters to the specific needs.

iv) Prioritizing Security and Usability

By implementing advanced user management features, we prioritize both security and usability. This translates to a seamless and secure environment for document handling and collaboration.

- **Enhanced Security:** Robust user authentication protocols like token-based authentication and controlled access minimize the risk of unauthorized access and safeguard sensitive information.

- **Streamlined Usability:** We prioritize a user-friendly registration process and intuitive interface, ensuring a positive experience from day one.

3.3.2 Structured Document Management

Structured document management encompasses two methods, such as

A) Structured Management

B) Standardized Naming Conventions

each fulfilling the dual objectives of organizing and segregating files, while also facilitating convenient access.

A) Structured Management

Structured document management is essential for maintaining an organized and efficient system for handling documents. It encompasses several key components aimed at facilitating convenient access, easy maintenance, and robust security measures.structured document management optimizes document handling processes by providing convenient access, easy maintenance, and robust security measures with several segregation types which is illustrated in Figure 3.3. By adhering to standardized naming conventions and organizing documents systematically, the system enhances clarity, uniformity, and efficiency in document management operations.

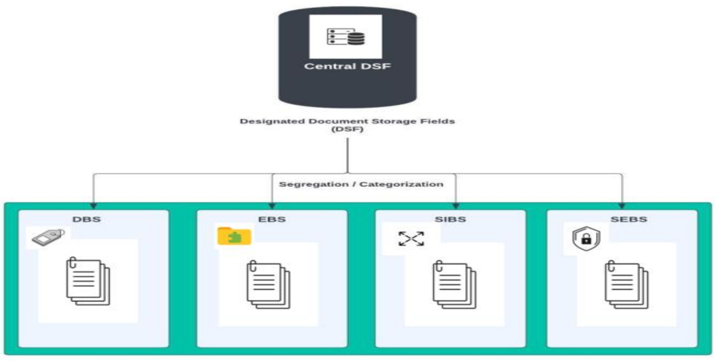

Central DSF
Designated Document Storage Fields (DSF)
Segregation / Categorization
DBS
EBS
SIBS
SEBS

Figure 3.3 - Structured Document Management

B) Standardized Naming Convention

In document management systems, easy identification and efficient management of documents are crucial aspects. Such systems enhance clarity and uniformity throughout the document handling process by ensuring consistent and systematic arrangement of documents. Key naming attributes play a vital role in this process, which include the domain name, file name, and the User's Identification ID (URD) which is illustrated in Figure 3.4. Incorporating these attributes enables streamlined organization and retrieval of documents, facilitating smooth workflow operations and reducing the likelihood of errors or confusion. By implementing a structured approach to document naming and management, organizations can optimize their document handling processes, improve productivity, and ensure a more efficient and transparent workflow.

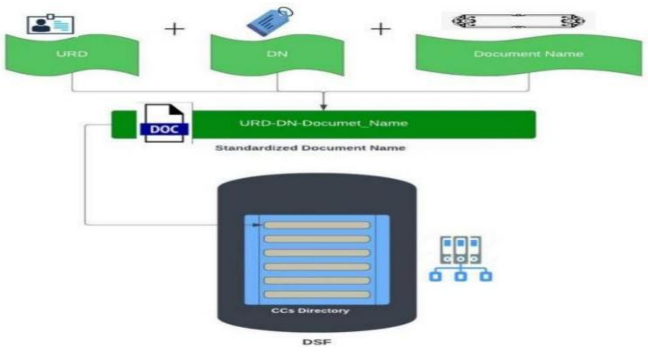


Figure 3.4 - Standardized Naming Conventions

Structured document management offers numerous advantages, such as :

i) Convenient Access

ii) Easy Maintenance

iii) Aligned Security Measures

iv) Easy Identification and Management of Documents

i) Convenient Access

Structured document management ensures that users can easily access the documents they need, when they need them. By organizing documents in a systematic manner, users can quickly locate and retrieve relevant information, improving overall productivity.

ii) Easy Maintenance

Maintaining documents becomes simpler with structured document management. Documents are segregated and organized based on standardized naming conventions, making it easier to update, modify, or archive them as needed. This streamlined maintenance process helps ensure that documents remain up-to-date and relevant.

iii) Aligned Security Measures

Security is a top priority in structured document management. By implementing aligned security measures, such as access controls and encryption, the system ensures that sensitive documents are protected from unauthorized access or tampering. This instills confidence in users regarding the confidentiality and integrity of their documents.

iv) Easy Identification and Management of Documents

Structured document management enables easy identification and management of documents through standardized naming conventions. Users can quickly locate specific documents by referencing key attributes such as domain name, file name, and URD. This simplifies document management tasks and reduces the risk of errors or confusion.

v) Consistent and Systematic Arrangement of Documents

Documents are arranged in a consistent and systematic manner within the structured document management system. This ensures that documents are organized logically, facilitating efficient navigation and retrieval. Users can rely on the structured arrangement of documents to locate information quickly and effectively.

3.3.3 Document Digital Signing

In our proposed system the file hashing and digital signing are pivotal security measures employed to ensure the authenticity and integrity of documents which is illustrated in Figure 3.5.

The sequence of methods employed for digital document signing is as follows:

i) File Hashing - SHA 256 Algorithm

ii) Digital Signing Mechanism

i) File Hashing - SHA 256 Algorithm

When a file is uploaded to the system, it undergoes a series of hashing processes to create unique fingerprints for each document. Utilizing the SHA-256 algorithm, a member of the SHA-2 cryptographic hash function family, the system hashes the complete content of the file. SHA-256's deterministic nature ensures consistent hash values for identical inputs, making it ideal for verifying data integrity. This resulting hash value acts as a distinct identifier for the document, effectively identifying any modifications made after uploading.SHA-256 operates on input messages of varying lengths, producing a fixed-size hash value of 256 bits or 64 characters in hexadecimal format. The algorithm involves padding the input, initializing hash values, processing input blocks via a compression function, and ultimately generating the final hash value. Its deterministic behavior and cryptographic robustness make it suitable for a range of security applications, including data integrity verification and password storage.

Through the integration of SHA-256 hashing and digital signing mechanisms, the system ensures the authenticity and originality of documents throughout their lifecycle. These security measures are integral to maintaining the integrity of the document management system, instilling confidence in users regarding the reliability and accuracy of their stored documents.

ii) Digital Signing Mechanism

Digital signing further enhances the integrity of uploaded documents by storing the hash value of the file's complete content in the database. This hash value serves as a reference point for subsequent originality checks. The integrity verification process involves re-computing the hash (Current Hash) using the Unique Reference Document (URD) and the document's current content. The system then compares the Current Hash with the Complete Hash stored in the Hash Storage Management (HSM), ensuring that any deviation between them indicates modifications to file content.

The document digital signing strategy yields numerous benefits , such as :

- **Data Integrity:** Consumers can trust that their uploaded documents remain unchanged. If the document is altered, its hash value changes.
- **Security:** SHA-256 provides robust security, resisting attacks. Consumers are confident their documents are safe from unauthorized changes.
- **Reliability:** SHA-256's consistency ensures the same input always yields the same hash value. This reliability ensures accurate document verification.
- **User Trust:** Implementing these measures builds user trust. Consumers feel confident in the system's reliability, fostering positive experiences and long-term relationships.
- **Enhanced Compliance:** By implementing robust security measures like SHA-256 and ensuring data integrity, businesses can better comply with industry regulations and standards related to document handling and privacy.

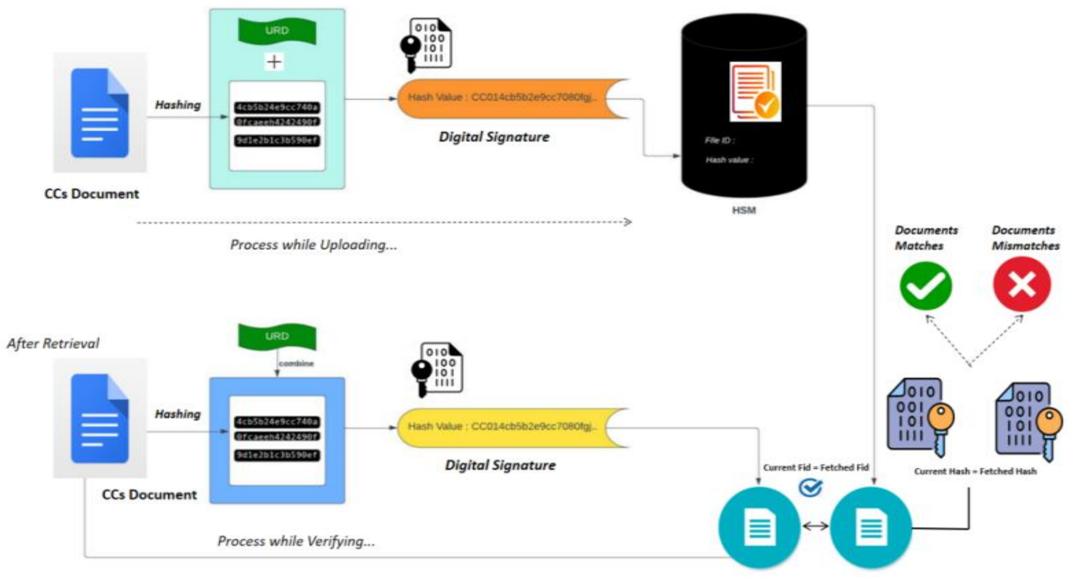


Figure 3.5 - Document Digital Signing and Verification

Algorithm 3.1 : Hashing

```
Input: User inputs

User_Recognition_ID (URD)
File_ID (Current_Document_File_ID)
Current_Document_Content


Process: Document Hashing
Current_Hash=Compute_Hash(URD+Hash_Current_Document_Content)

 Information Stored: URD, Stored_File_ID

Verification:

Stored_File_ID=Retrieve_File_ID_From_CMS_Database(URD)
Stored_Hash = Retrieve_Hash_From_HSM(URD, Stored_File_ID)

Returns the original file - checks the originality:

if Current_Document_File_ID == Stored_File_ID:
if Current_Hash == Stored_Hash:
Print("Document integrity verified. Original document.") else:
Print("Document integrity verification failed. Possible tampering.")
else:
Print("File ID mismatch. Document integrity verification aborted.")
```

Steps for Digital Signature and Verification Process which is depicted in algorithm 3.1 is explained as follows:

Step 1 : Initialization

1.1 Upon file upload, initiate hashing process.

Step 2 : Hashing with SHA-256

2.1 The content of the input file is processed through the SHA-256 algorithm.
2.2 The resulting hash value is generated as the output

Step 3 : Deterministic Behavior

3.1 Ensure same input always produces same hash value such that for each time the same input is processed, it consistently generates the identical hash value.

Step 4 : Unique Identifier - Digital Signature

4.1 The hash value functions as a distinct marker for each document.

4.2 It effectively identifies any modifications made after the document has been uploaded.

Step 5 : Digital Signing Mechanism:

5.1 Enhance integrity of uploaded documents.

Step 6 : Storing Hash Value:

6.1 Store the hash value of file's complete content in database.

6.2 Serve as reference point for subsequent originality checks.

Step 7: Integrity Verification:

7.1 Re-compute hash (Current Hash) using URD and document's current content.

7.2 Upon receiving the input of the document's content and URD, the system generates the current hash.

7.3 This hash is then compared with the complete hash stored in the Hash Storage Manager (HSM). Any deviation between the two hashes indicates modifications to the file content.

3.3.4 Document Encryption and Decryption

In the proposed work, a sophisticated hybrid encryption method ensures the secure transmission of uploaded files, striking a balance between efficiency and robust security. This two-layered encryption strategy combines symmetric and asymmetric cryptography for optimal protection which is illustrated in Figure 3.6 and Figure 3.7.

The cryptography mechanism involves two primary methods, such as :

A) Encryption

B) Decryption

A) Cryptographic Mechanism - Hybrid Encryption:

The sequence of steps followed by the hybrid encryption strategy is as follows:

i) AES (Advanced Encryption Standard): (256 bit)

Utilizing symmetric encryption with the same key - Symmetric Mono Key (SMk) ensures robust confidentiality for documents through a 256-bit AES (Advanced Encryption Standard) key (C_{AES}). This method employs a single key for both encryption and decryption, simplifying the process while maintaining high security standards. With AES's 256-bit key length, the encryption is exceptionally strong, providing a formidable barrier against unauthorized access. By applying this method, documents remain safeguarded, ensuring that sensitive information remains confidential and protected from unauthorized interception or tampering.

ii) RSA (Rivest-Shamir-Adleman): (256 bit)

RSA, as an asymmetric algorithm, relies on a pair of keys: a public key and a private key, which is kept confidential. The public key serves as a widely distributed cipher used for encrypting data securely. It allows anyone to encrypt messages intended for the owner of the corresponding private key. However, decryption can only be performed using the private key, ensuring that only the intended recipient, possessing the private key, can decipher the encrypted data. This ingenious approach not only enables secure data transmission but also facilitates the creation of digital signatures. RSA's functionality include its resistance to brute force attacks due to the complexity of factoring large prime numbers, its versatility in various cryptographic applications beyond encryption, such as digital certificates and secure authentication protocols, and its widespread adoption as a standard encryption method in various industries, reaffirming its reliability and effectiveness in securing sensitive data.

iii) Two-Layered Encryption Approach - Hybrid Secure Strategy

To ensure maximum security, documents go through a two-step encryption process. Initially, the Advanced Encryption Standard (AES) safeguards the data itself. Then, another layer of protection is added using RSA encryption.

The hybrid encryption strategy described combines AES (Advanced Encryption Standard) and RSA (Rivest-Shamir-Adleman) encryption methods to ensure robust security for document protection. In the first layer, documents undergo encryption using AES with a 256-bit key (SMK), a symmetric encryption method. AES encryption (C_{AES}) generates a securely encrypted version of the document (Doc_{SCF}), utilizing the same key for both encryption and decryption.

To further enhance security, the AES key (SMK) undergoes encryption using RSA encryption. This involves encrypting the AES key (SMK) with the recipient's public key (APuK), resulting in a separate cipher (C_{RSA}). This additional step ensures that the AES key is safeguarded during transmission or storage, as only the recipient possessing the corresponding private key (APrK) can decrypt and retrieve the AES key, thereby ensuring end-to-end security for document transmission and storage.

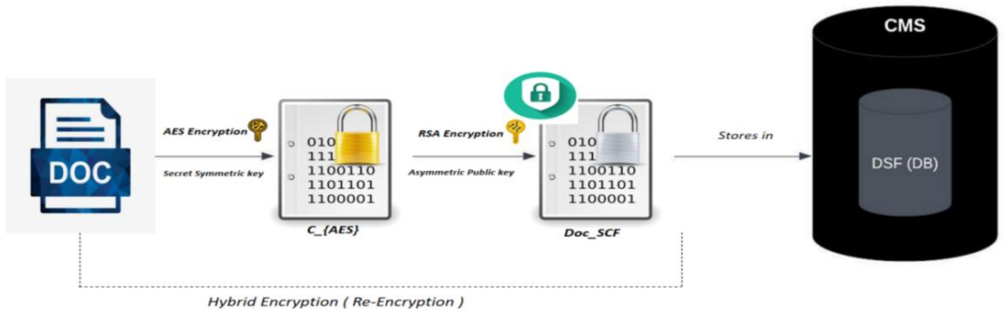


Figure 3.6 - Encryption Process

Algorithm 3.2 - Cryptography - Encryption

```
Encryption:
Step 1:
    ● Input ( Document, SMK, APuK )
Step 2:
    ● Encryption with AES using Symmetric Key (SMK)
    ● C_AES = AES_Encrypt(Document, SMK)
Step 3:
    ● Encryption of AES Symmetric Key with RSA Public Key (APuK)
    ● Doc_SCF = RSA_Encrypt(C_AES, APuK)
Step 4:
    ● Output Output: Doc_SCF
```

B) Cryptographic Mechanism - Decryption:

The sequence of steps followed by the hybrid decryption strategy and its outcomes is as follows:

i) Unlocking the Secret Key:

The encrypted document initially appears scrambled. The recipient's Asymmetric private key (APrK) acts like a master key. It is used to decrypt a specific portion of the encrypted document format (Doc_ {SCF}) which is considered as an complete encypted format. Then the symmetric key, called as Symmetric Mono Key (SMK) and typically uses the Advanced Encryption Standard (AES) algorithm which is considered to be in layer 2 of decryption.

ii) Content Decryption Takes Over:

The process begins with decrypting the RSA by using RSA's Private key and then decrypting AES by using AES key using SMK, thus enabling AES to decrypt the document's actual content. This two-step approach capitalizes on the unique strengths of both asymmetric and symmetric cryptography: RSA ensures the secure transmission of the AES key, while AES efficiently manages the bulk decryption of the document. In essence, RSA aims to break the initial encryption barrier, following with AES to step in and decrypt the document with precision. This collaborative effort ensures a robust encryption-decryption cycle, reinforcing the security and integrity of the document transmission process.

iii) RSA Decryption

The process of retrieving the key for decryption utilizes a two-step approach. The encrypted document (Doc_{SCF}) contains the scrambled encyrpted data that's essential for unlocking the document's content. To access this key, the recipient's private key (APrK) is used. Imagine the private key as a master key specifically designed to open a hidden compartment within the encrypted document. This compartment holds the RSA's Private key, which will be used in the following step to decrypt the actual content of the document in layer 1 which breaking the complete encrypting content.

iv) AES Decryption

After the RSA of layer 1 decryption compelete then the SMK (Symmetric Mono Key) retrieves the AES key from the encrypted document.It used for the layer 2 of decryption. With the AES key in hand, the decryption process takes place. The AES algorithm, known for its efficiency, takes center stage. The AES key acts like a key fitting perfectly into a complex lock, unlocking the encrypted document content (C_{AES}). Finally, the original document content is revealed.

v) Final User-Readable File Format

The result of the hybrid decryption process is the user - readable file format, which is the original document content decrypted from its encrypted state which is illustrated in Figure 3.3. This file format can be accessed and used by the authorized recipient or user, ensuring confidentiality and security of the document during transmission and storage.

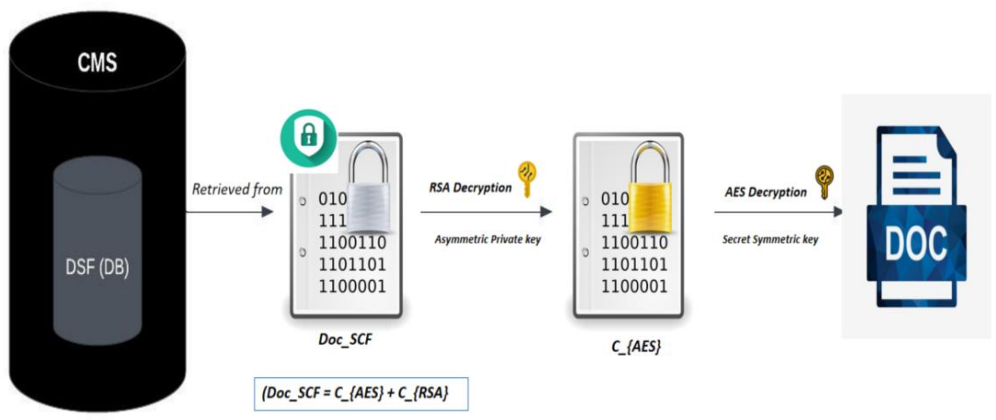


Figure 3.7 - Decryption Process

Algorithm 3.3 - Cryptographic Decryption

Decryption:

Step 1:

- *Input (Doc_SCF, APrK)*

Step 2:

- *Decryption of AES Symmetric Key with RSA Private Key (APrK)*
- *C_AES = RSA_Decrypt(Doc_SCF, APrK)*

Step 3:

- *Decryption of Document using AES Symmetric Key*
- *Document = AES_Decrypt(C_AES, SMK)*

Step 4:

- *Output: Document*

Decryption initiates by inputting the securely encrypted document (Doc_SCF) and the RSA private key (APrK). Then, utilizing RSA decryption, the AES symmetric key (SMK) is extracted, enabling the subsequent AES decryption of the document, resulting in the retrieval of the original document which is illustrated in Algorithm 3.3.

3.3.5 Secure Document Upload

In the proposed system, ensuring the secure upload of files is paramount, achieved through meticulous actions performed by the system to uphold both security and manageability standards. Users interact with a user-friendly web portal to upload documents, providing essential details including User ID, Filename, Domain, File Description, Keywords, and the file itself.

Details Required for Upload:

- **User ID:** Unique identifier for the user uploading the document.
- **Filename:** Name of the file being uploaded.
- **Domain:** Category or area to which the file belongs.
- **File Description:** Brief description or summary of the file's content.
- **Keywords:** Relevant keywords associated with the file for searchability and categorization.
- **File:** The actual document file being uploaded.

To enhance document security during uploading, a two-tier encryption process is implemented

i) Level 1 : AES Encryption (256 bit)

ii) Level 2 : RSA Encryption (256 bit)

i) Level 1: AES Encryption (256-bit)

At Level 1 of encryption, the system employs the Advanced Encryption Standard (AES) with a formidable 256-bit key to secure files. This process involves scrambling the document contents with an intricate lock, rendering unauthorized access nearly impossible. The robust 256-bit key acts as a formidable barrier, safeguarding data during transmission and storage within the system. This AES encryption method serves as a potent defense mechanism against potential threats, providing users with confidence in the security of their uploaded files.

ii) Level 2: RSA Encryption (using .pem file)

The commitment to data security extends beyond a singular layer of protection, as evidenced by Level 2 encryption within the system. This additional safeguard is tailored for highly sensitive documents and employs RSA public key encryption, facilitated with a unique key stored in a .pem file. Access to this vault is restricted to authorized recipients possessing the corresponding private key. Through this dual encryption approach, which combines AES and RSA, a formidable barrier against unauthorized access is erected. Encrypting the AES key itself with RSA ensures heightened confidentiality and integrity for the most critical data, further fortifying the system's security measures.

Numerous advantages exist when securing files during the uploading process ,such as:

- **Robust Protection:** Utilizing AES encryption with a 256-bit key ensures formidable security for files, effectively preventing unauthorized access during transmission and storage.

- **Tailored Security:** The use of RSA encryption at Level 2, complemented by .pem files, offers tailored security for highly sensitive documents. This additional layer of protection restricts access to authorized recipients holding the corresponding private key.

- **Enhanced Confidentiality:** Encrypting the AES key with RSA at Level 2 significantly enhances the confidentiality of critical data. This dual encryption method strengthens the system's security measures, mitigating potential threats and unauthorized access.

- **User Confidence:** By employing advanced encryption techniques such as AES and RSA, the system inspires user confidence in the security of their uploaded files. This holistic approach to data security assures users that their sensitive information is well-protected within the system.

3.3.6 Keywords Extraction and Retrieval & Ranking

The process of keyword extraction, retrieval, and ranking comprises two primary stages namely,

A) Keyword Extraction Process

B) Retrieval and Ranking Process

A) Keyword Extraction Process

Enhancing searchability is another feature built into the system. Upon uploading the document to the Central Management Server (CMS), the system automatically performs a keyword extraction process. Imagine the system intelligently scanning the document to identify and extract the most relevant keywords. These keywords become tags associated with the document, making it significantly easier to find later. This guarantees that the documents remain accessible and can be easily retrieved using relevant keywords within the CMS, preventing them from getting lost in the digital void.

Steps to process and extract the keywords:

- **Step 1 : Tokenization:** The text is segmented into individual tokens (words or phrases).
- **Step 2 : Lowercasing:** Standardizing tokens to a uniform lowercase format to avoid case discrepancies.
- **Step 3 : Stopword Removal:** Common and less meaningful words (stopwords) are eliminated to focus on significant terms.
- **Step 4 : Stemming and Lemmatization:** Words are reduced to their base or root forms to normalize variations.

Keywords are extracted based on their frequency values within the document, resulting in a list of extracted keywords represented as:

$$\boxed{keywords = [kw1, kw2, ..., kwn]}$$

B) Retrieval and Ranking

The retrieval and ranking process encompasses various methods and strategies, such as,

a. Query Processing Techniques

b. Keyword Matching and Ranking

c. Keyword Matching and Ranking

d. File Filtering/Ranking Functionality

e. Document Similarity

f. Retrieval - Finding Relevant Documents

g. Ranking - Prioritizing the Most Relevant Documents

h. Ranking Algorithm - BM25

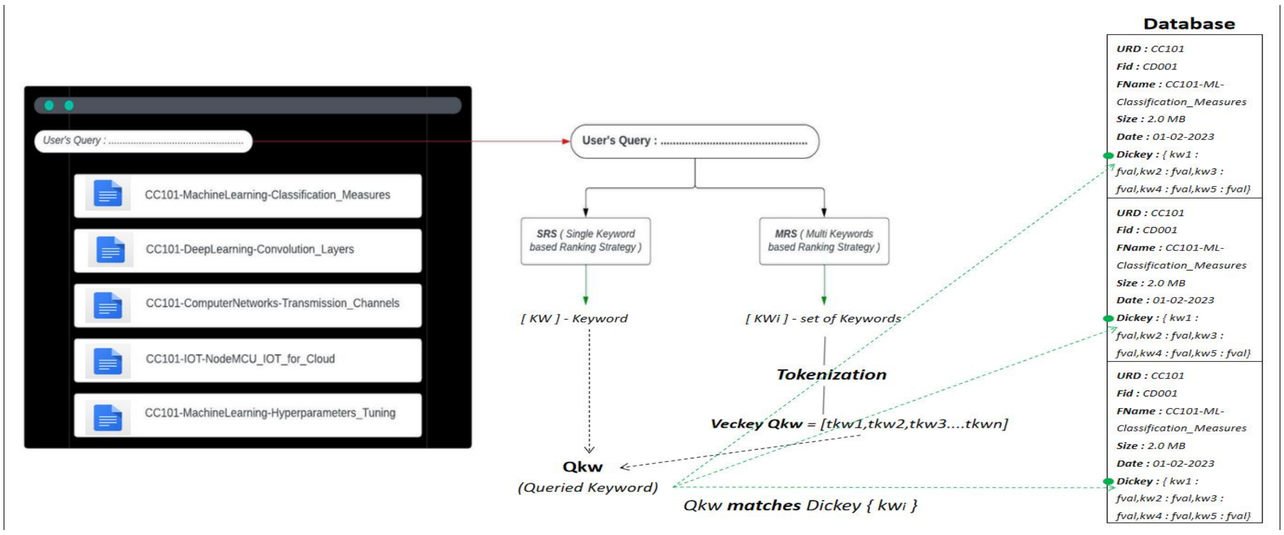

Database
URD : CC101
Fid : CD001
FName : CC101-ML-Classification_Measures
Size : 2.0 MB
Date : 01-02-2023
Dickey : { kw1 : fval,kw2 : fval,kw3 : fval,kw4 : fval,kw5 : fval}
User's Query :
CC101-MachineLearning-Classification_Measures
CC101-DeepLearning-Convolution_Layers
CC101-ComputerNetworks-Transmission_Channels
CC101-IOT-NodeMCU_IOT_for_Cloud
CC101-MachineLearning-Hyperparameters_Tuning
SRS (Single Keyword based Ranking Strategy)
MRS (Multi Keywords based Ranking Strategy)
[KW] - Keyword
[KWi] - set of Keywords
Tokenization
Veckey Qkw = [tkw1,tkw2,tkw3....tkwn]
Qkw
(Queried Keyword)
Qkw matches Dickey { kwi }

Figure 3.8 - Retrieval and Ranking

a) Query processing Techniques

The system is used to securely store the documents and it also makes them easy to find. When the user enter a search query, the proposed system’s application goes into action and thus process it. It utilizes sophisticated techniques to analyze the query and identify the documents most relevant to the needs.

Steps to process the user’s query:

- **Tokenization:** The user query is tokenized into individual terms.
- **Lowercasing:** Tokens from the query are standardized to lowercase.
- **Stopword Removal:** Common stopwords are removed from the query terms.
- **Stemming and Lemmatization:** Query terms are reduced to their base forms for normalization and consistency.

b) Keyword Matching and Ranking

The user's query is matched against the stored array of document keywords and their associated probability values. The documents are then ranked based on the relevance of their keywords to the query, considering the extracted probability values.

c) File Filtering/Ranking Functionality

Additionally, the system incorporates functionalities for filtering or ranking uploaded files based on specific criteria or attributes. While specific details may vary, the system can categorize or prioritize documents based on user-defined criteria such as file type, domain, relevance, or other metadata. This capability enhances user experience by facilitating efficient retrieval and ranking of documents based on user preferences or search criteria.

d) Document Similarity

Techniques like TF-IDF (Term Frequency-Inverse Document Frequency) can measure the relative importance of terms within a document and across the entire document collection which is illustrated in Figure 3.8. This helps identify documents that discuss similar concepts or ideas, even if they do not use the exact same keywords.

e) Retrieval - Finding Relevant Documents

When a user enters a search query using keywords or phrases, the retrieval process get started.The system searches through the collection of uploaded documents, focusing on the extracted keywords associated with each document.Documents containing keywords that match or are similar to the user's search query are identified as potential results.

f) Ranking - Prioritizing the Most Relevant Documents

It is mentioned that not all documents with matching keywords might be equally relevant to the user's intent.Ranking algorithms analyze the retrieved documents and prioritize them based on their relevance to the search query. This ensures the most relevant documents appear at the top of the search results, saving users time by surfacing the information.

e) Ranking Algorithm - BM25

The BM25 algorithm is a powerful tool for ranking documents in information retrieval systems. It incorporates various components to accurately assess the relevance of documents to a given query. At its core, BM25 operates within the probabilistic retrieval framework which is illustrated in Figure 3.9, assuming that relevant and non-relevant documents follow distinct statistical distributions which is depicted in algorithm 3.4.

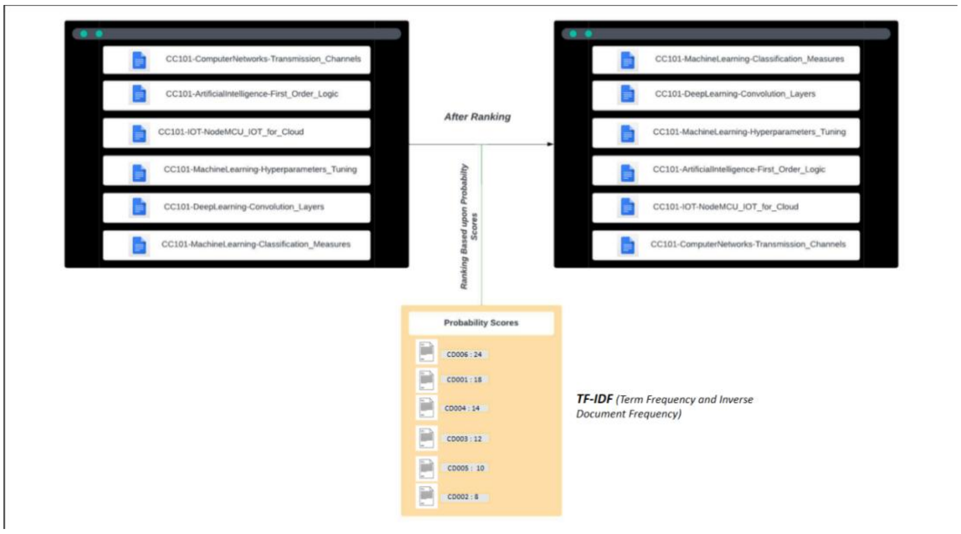

CC101-ComputerNetworks-Transmission_Channels
CC101-ArtificialIntelligence-First_Order_Logic
CC101-IOT-NodeMCU_IOT_for_Cloud
CC101-MachineLearning-Hyperparameters_Tuning
CC101-DeepLearning-Convolution_Layers
CC101-MachineLearning-Classification_Measures
After Ranking
Ranking Based upon Probability Scores
CC101-MachineLearning-Classification_Measures
CC101-DeepLearning-Convolution_Layers
CC101-MachineLearning-Hyperparameters_Tuning
CC101-ArtificialIntelligence-First_Order_Logic
CC101-IOT-NodeMCU_IOT_for_Cloud
CC101-ComputerNetworks-Transmission_Channels
Probability Scores
CD006 : 24
CD001 : 18
CD004 : 14
CD003 : 12
CD005 : 10
CD002 : 8
TF-IDF (Term Frequency and Inverse Document Frequency)

Figure 3.9 - Documents's Probability Scores - Ranking

The BM25 algorithm comprises several essential components such as :

i) Term Frequency (TF):

BM25, or Best Matching 25, is a ranking function used in information retrieval to determine the relevance of documents to a given search query. Unlike traditional term frequency schemes, BM25 employs a modified approach that addresses the saturation effects caused by heavily repeated terms. In essence, it prevents these overly frequent terms from disproportionately influencing the relevance score of a document. This modification helps to ensure that the ranking reflects the true relevance of the document to the query, rather than being skewed by the prevalence of certain terms. By considering the frequency of terms in a more nuanced manner, BM25 enhances the accuracy and effectiveness of information retrieval systems, ultimately providing users with more relevant and useful results.

ii) Inverse Document Frequency (IDF):

IDF, or Inverse Document Frequency, plays a crucial role in information retrieval by quantifying the significance of a term across the entire corpus. How common a term is across the entire document collection. Less common terms (higher IDF) carry more weight in the ranking, as they might be more specific indicators of document relevance.High Term Frequency (TF) within a document indicates a focus on that term. However, if the term is very frequent across many documents, it might not be as specific or informative for identifying relevant documents. IDF helps address this by penalizing overly common terms.

iii) Document Length Normalization:

BM25, in its quest for fair and accurate document ranking, incorporates a crucial feature known as document length normalization. This mechanism is implemented to counteract the bias that arises from differences in document lengths. Longer documents naturally tend to have more occurrences of terms, potentially skewing the relevance scores in their favor. However, with document length normalization, BM25 levels the playing field by adjusting the relevance scores to account for document length. This means that longer documents are not unfairly favored in the ranking process, ensuring that each document's relevance is evaluated impartially and accurately. By applying this normalization technique, BM25 enhances the reliability and fairness of the ranking algorithm, resulting in more equitable and relevant search results for users.

iv) Query Term Saturation:

BM25 incorporates a mechanism designed to tackle the challenge posed by excessively high term frequencies in queries. When certain terms occur too frequently in a query, they can disproportionately influence the relevance ranking, potentially skewing the results. BM25 employs a function that dampens the impact of such excessively high frequencies. By moderating the influence of these terms.

Several advantages exist for the retrieval and ranking process, sucha as:

- Efficient Search
- Improved User Experience
- Enhanced Discoverability
- Increased Productivity

- **Increased Productivity**

  With quick access to relevant documents, users can complete tasks more efficiently, leading to improved productivity and workflow optimization.

- **Efficient Search**

  Ranking algorithms prioritize documents based on their relevance, ensuring users find the information they need quickly and easily.

- **Improved User Experience**

  Effective retrieval and ranking lead to a more efficient search experience, saving users time and frustration by surfacing the most relevant documents first.

- **Enhanced Discoverability**

  Ranking can help users discover documents they might not have found by browsing alone, especially if the documents contain relevant concepts or ideas expressed with different keywords.

Algorithm 3.4 : Ranking - BM25

Step 1 :

Calculate IDF for each term in the query

- ***For each term t in the query***
- ***Calculate IDF using the formula***

$$IDF(t) = \log(\frac{N - n(t) + 0.5}{n(t) + 0.5})$$

Where:

✧ *N is the total number of documents in the corpus.*

✧ *n(t) is the number of documents containing term t.*

Step 2 :

Calculate TF for each term in each document

- ***For each term t in each document d***
- ***Calculate TF using the formula***

$$TF(t,d) = \frac{f_{t,d}}{\sum_{t' \in d}^{i} f_{t'd}}$$

Step 3 :

Combine TF and IDF to compute the relevance score for each document

- ***For each document d***
- ***For each term t in the query***
- ***Calculate the product of IDF and TF***

$$IDF(t).TF_{t,d}$$

Sum up the products for all query terms to get the relevance score for document d

$$score(D,Q) = \sum_{i=1}^{n} IDF(q_i).\frac{f(q_i,D).(k_1+1)}{f(q_i,D)+k_1.(1-b+b.\frac{|D|}{avgdl})}$$

Where :

✧ *D*	*Document*
✧ *Q*	*Query*
✧ *n*	*Number of unique terms*
✧ *Q i*	*each unique term in query*
✧ *F(qi ,D)*	*term frequency of term*
✧ *|D|*	*Document Length*
✧ *Avgdl*	*Average document length*
✧ *IDF (qi)*	*Inverse Docuemt Frequency*

3.3.7 File Compression

The system employes the Brotli lossless compression for its effectiveness in achieving the objectives which is illustrated in Figure 3.10. Brotli offers high compression ratios while maintaining data integrity, making it suitable for various applications, including critical ones. By preserving the original data without any loss of information during compression and decompression, Brotli ensures data integrity, which is essential for maintaining the accuracy and reliability of critical application

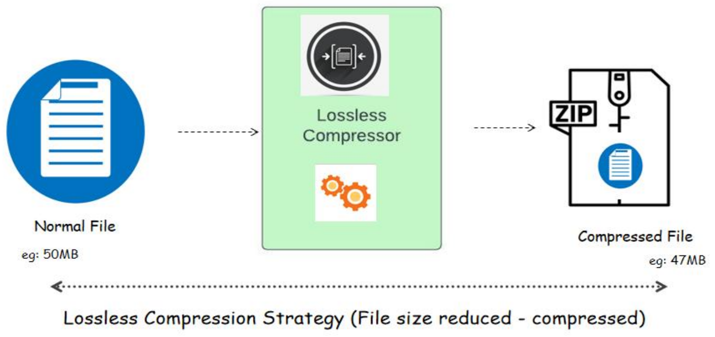


Figure 3.10 - Lossless Compression Strategy

Various needs exist for the compression strategy, such as :

i) Reduced Storage Requirements:

Compression helps in minimizing the amount of storage space needed for storing documents. By reducing the size of files, we can optimize storage utilization and accommodate more data within limited storage capacities.

ii) Faster Data Processing:

Compressed files require less data to be processed, resulting in faster processing times. This is particularly beneficial for applications that involve large volumes of data, as it reduces the time required for data transfer and manipulation.

iii) Improved Performance:

With reduced file sizes and faster data processing, overall system performance is enhanced. Users experience quicker access to documents and smoother system operations, leading to improved productivity and user satisfaction.

iv) Enhanced Security:

Compression can contribute to enhanced security by reducing the exposure of sensitive data. Smaller file sizes make it easier to transfer encrypted data securely, minimizing the risk of unauthorized access or interception during transmission.

Several advantages exist for the lossless compression strategy,such as :

i) Exceptional Performance:

Brotli compression algorithm excels in achieving high compression ratios and rapid decompression speeds, specifically tailored for text-based content like HTML, CSS, and JavaScript files.

ii) Superior Efficiency:

Compared to other compression methods, Brotli demonstrates superior efficiency, surpassing many traditional techniques in reducing file sizes.

iii) Optimal Balance:

Brotli strikes an optimal balance between compression ratios and decompression speeds, making it ideal for scenarios where reducing data size while ensuring quick access to compressed data is essential.

iv) Lossless Compression:

A distinguishing feature of Brotli is its focus on lossless compression, ensuring the preservation of the original data's integrity, crucial for applications where accuracy and fidelity are paramount.

v) Versatility and Practical Application:

Brotli's practical application extends beyond web development, showcasing its versatility in various systems to enhance overall performance.Real-world examples consistently demonstrate Brotli's effectiveness in reducing load times and optimizing bandwidth usage.

3.3.8 AI Interactive Model

An AI interactive model of the system consists of two approaches , such as :

i) User’s Document Interaction

ii) Interactive model - learning

i) User’s Document Interaction:

Langchain is an AI strategical model which simplifies the development of document interaction applications powered by large language models (LLMs) through pre-built components like document loaders, prompt templates, and output parsers which is illustrated in 3.11. This structured workflow involves users loading their documents, crafting prompts for LLM interactions (e.g., asking questions or requesting summaries), Langchain facilitating the interaction using the document and prompt, and output parsers interpreting the LLM's responses based on the prompt's intent. This approach enables developers to create user-friendly applications where documents are easily engaged with, leveraging Langchain's LLM for efficient processing and presentation of information.

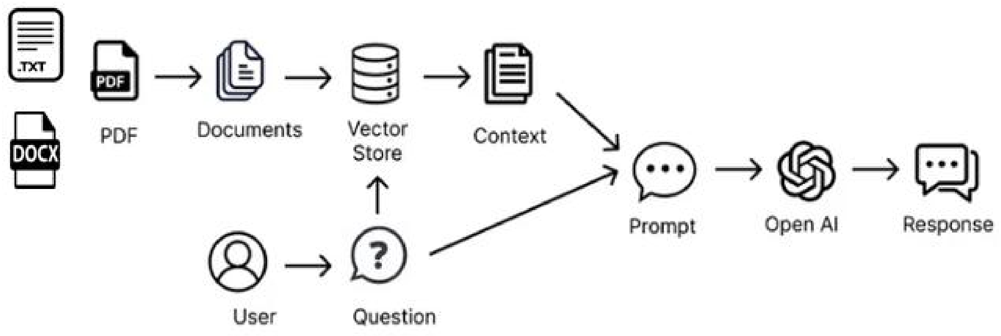


Figure 3.11 - Langchain - Document Interaction LLM

ii) Interactive Model - Learning

The interactive model encompasses various methodologies and strategies, such as

a) Large Language Models (LLMs)

Large Language Models (LLMs) like Llama-2 are the driving force behind innovative document interaction systems which is illustrated in Figure 3.12. These powerful AI models are trained on massive amounts of text data, allowing them to understand and generate human-like language. In the context of document chatting, LLMs analyze the custom dataset, grasping the unique vocabulary, relationships, and content within the documents. This empowers them to respond to the chat prompts by summarizing key points, identifying connections between documents, and answering factual questions directly from the data.

b) Llama 2 and LORA

The combination of Llama-2 and LORA models unlocks a powerful approach for interaction through a chat interface using the custom dataset which is entirely defines and depicted in algorithm 3.5.

c) Llama-2 as the Foundation

Llama-2, a large language model by Meta, excels at understanding and generating human-like text. It serves as the backbone for processing the customized data and responding to the chat prompts.

d) Custom Dataset Integration

The dataset is preprocessed and fed into Llama-2. This allows Llama-2 to learn the specific vocabulary, relationships, and context within the documents.

e) LORA for Fine-Tuning:

LORA (Locally Reparametrized Optimizer) acts as a fine-tuning tool. It adapts the pre-trained Llama-2 model specifically to the dataset which is illustrated in Figure 3.13. This improves the model's ability to understand and respond to queries.

$$\boxed{\textit{Fine-tuned Llama-2} = \textit{Llama-2 (Base Parameters)} + \textit{LORA}}$$

The base Llama-2 parameters as a large, complex function. LORA introduces smaller adjustments based on the dataset, effectively shaping the function to perform better on the specific documents.This document chat system powered by Llama-2 and LORA empowers you to access and analyze information with ease.The user simply ask questions in a natural conversation, and the system leverages the custom dataset to deliver insightful responses. LORA's fine-tuning keeps things relevant by focusing on the specific data, eliminating irrelevant information. This intuitive chat interface makes exploring the documents as easy as having a conversation.

Alogrithm 3.5 : AI Interactive Model

```
LLAMA and QLORA for Learning of News and Social Impact

Input: Preprocessed student answer text (answer_text)
Output: Graded score (graded_score), Feedback (feedback)

  Start
Step 1: Initialize LLAMA model (LLAMA) and QLORA model (QLORA).
Step 2: Load pre-trained LLAMA model and QLORA model with specified configurations.
Step 3: Preprocess the News Article data.
Step 4: Encode the question and reference using the LLAMA model to obtain their representations.
Step 5: Encode the News and Social impact text using the LLAMA model to obtain its representation.
Step 6: Combine the representations of the question and reference  with the News and Social impact representation.
Step 7: Reduce the dimensionality of the combined representation using the QLORA model.
Step 8: Predict the News and Social impact using a fully connected layer applied to the reduced representation.
Step 9: Generate feedback based on the reduced representation.
Step 10: Set the training parameters for fine-tuning the LLAMA and QLORA models.
Step 11: Train the LLAMA and QLORA models
Step 12: Save the fine-tuned LLAMA and QLORA models for future use.
    End
```

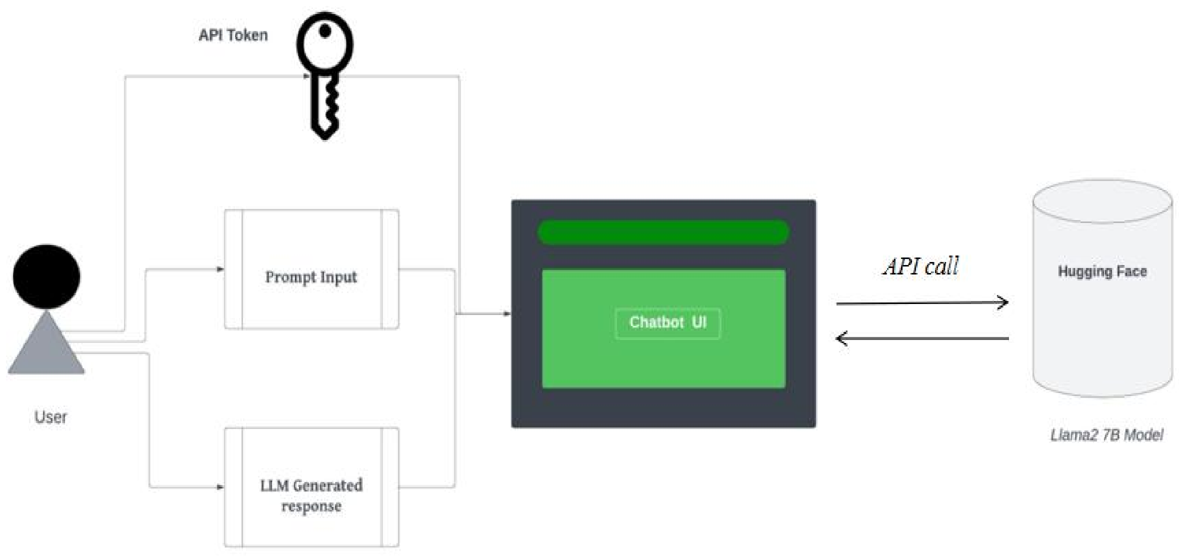


Figure 3.12 - System Model - AI Interaction model

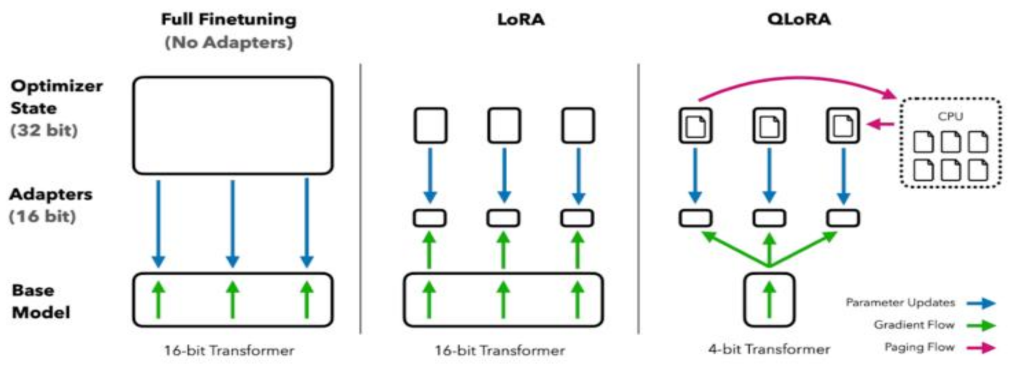


Figure 3.13 - Fine tuned LoRA Model

3.4 SUMMARY

Structured document management offers significant advantages for both users and organizations, streamlining access to information and enhancing productivity. By organizing documents systematically, users can swiftly locate the data they require. Additionally, maintenance is simplified through standardized categorization and naming conventions, facilitating updates, modifications, and archiving. Aligned security measures, such as access controls and encryption, prioritize document confidentiality and integrity, ensuring data remains secure.

In the proposed system, file hashing and digital signing play pivotal roles in ensuring document authenticity and integrity. The hash value serves as a unique identifier, aiding in detecting any alterations post-upload, while the digital signing mechanism stores and compares hash values for integrity verification. Leveraging the SHA-256 algorithm's reliability strengthens trust in the system's accuracy and reliability throughout the document lifecycle.The hybrid decryption strategy employed combines RSA decryption for retrieving the AES symmetric key and AES decryption for decrypting the document content, facilitating the recovery of the original user-readable file format from its securely encrypted state.Through advanced encryption methodologies like AES and RSA, user trust and confidence are bolstered.

This approach enhances data security but also upholds user privacy and data protection standards, fortifying the overall security posture of the document upload system, thereby ensuring sensitive information remains protected against unauthorized access and threats.Furthermore, the hybrid encryption strategy combines the efficiency of symmetric encryption (AES) with the enhanced security of asymmetric encryption (RSA), leveraging the strengths of both encryption methods to achieve a secure and efficient document encryption solution.The utilization of the Brotli compression algorithm further enhances system efficiency by achieving high compression ratios and rapid decompression speeds, especially for text-based content like HTML, CSS, and JavaScript files. Its focus on lossless compression ensures data integrity, crucial for accuracy-sensitive applications, setting a new efficiency standard in contemporary applications.

Lastly, Langchain simplifies document interaction by leveraging large language models (LLMs) like Llama-2 and LoRa. Users can load documents, craft prompts, and receive tailored responses, enhancing document exploration through intuitive conversation-like interactions, thus facilitating a deeper understanding of the content.

CHAPTER 4

SYSTEM IMPLEMENTATION

4.1 OVERVIEW

The system implementation phase is a crucial stage in the software development lifecycle where the theoretical design is transformed into a working system. This system provides the steps taken to implement the system, including the tools, technologies, methodologies, and resources utilized throughout the process.

4.2 SOFTWARE SETUP

The software setup for the Endovault - A Novel Efficient Document Management in Cloud Storage with Secure, Structured and Keyword-Driven Retrieval System comprises various components, including the backend implemented using MongoDB Atlas and Microsoft Azure, frontend developed using React.js and AI Model by using LLM transformers.

4.3 BACKEND IMPLEMENTATION

The backend implementation primarily revolves around leveraging MongoDB Atlas for database management and Microsoft Azure for hosting and infrastructure support.

A strong backend is essential for the success of the proposed work, and MongoDB Atlas coupled with Microsoft Azure Storage can offer a potent solution. MongoDB Atlas, a cloud-based document database, provides a flexible and scalable platform for storing critical application data, such as user information and document references. Its schemaless architecture enables effortless adaptation to evolving data requirements, ensuring agility as the project evolves. Microsoft Azure Storage, on the other hand, seamlessly integrates with the backend, managing uploads, storage, and delivery of various media files. It optimizes media for diverse devices and guarantees secure storage, aligning perfectly with the project's needs.Together, MongoDB Atlas and Microsoft Azure Storage form a robust backend infrastructure capable of scaling the system. By efficiently managing both data and media files, this partnership enhances user experience, ensuring that the application remains responsive and reliable even as it grows in complexity and scale

There are numerous advantages associated with integrating the backend implementation of the system, offering significant benefits , such as :

- **Scalability**

  It is easy to scales up or down based on the application's data storage needs.

- **Flexibility**

  It is easy adapts to changing data models and requirements without rigid schema restrictions.

- **Cloud-Based**

  It offers a convenient and reliable cloud-hosted solution, eliminating the need for managing the own database infrastructure.

- **Integration**

  It integrates well with various programming languages and frameworks, simplifying development.

4.4 FRONTEND IMPLEMENTATION

The proposed system leverages React.js, a popular JavaScript library, to build a user-friendly and interactive frontend for the file storage system.The component of the frontend implemetation is as follows :

- User Login and Signup Forms
- File Upload Section with Drag-and-Drop Functionality
- File List and Management Section with Sorting/Filtering Options
- Progress Bars for Uploads and Encryption
- Visualization Components for AI-powered News Feed or Results
- State Management: React allows for effective state management. Libraries like Redux or MobX can be integrated to manage user information, upload progress, file

data, and data retrieved from the AI model. This ensures a consistent state across the application.

4.5 AI INTERACTIVE MODEL

Various implementation packages are utilized in the development of AI interactive models, with some such as :

i) Pandas

Pandas serves as a cornerstone in Python's data manipulation and analysis arsenal, facilitating seamless handling of diverse datasets. Its core features, including DataFrames and Series, offer structured representation and manipulation capabilities. Whether it's cleaning messy data or performing complex calculations, Pandas excels with its intuitive interface. By seamlessly integrating with visualization libraries like Matplotlib and Seaborn, Pandas enhances data exploration and presentation. Its benefits extend to simplifying coding efforts, ensuring high performance even with sizable datasets, and empowering robust data analysis across various domains. From loading data from different sources to conducting exploratory data analysis, Pandas stands as a versatile tool in the data scientist's toolkit.

ii) NumPy

NumPy forms the backbone of numerical computing in Python, offering efficient array manipulation and mathematical operations. Its multidimensional arrays enable rapid numerical computations, crucial for scientific and engineering applications. With built-in functions for linear algebra, statistics, and random number generation, NumPy provides a comprehensive suite of tools for numerical analysis. Its seamless integration with other scientific computing libraries amplifies its utility in tasks ranging from image processing to machine learning. As a fundamental building block, NumPy empowers researchers and developers to tackle complex numerical challenges with ease and efficiency.

iii) AI Transformers

AI Transformers represents the cutting edge of deep learning in natural language processing (NLP), delivering state-of-the-art performance across a plethora of tasks. Built on the foundation of encoder-decoder architectures and attention mechanisms, these models excel at understanding and generating human-like text. Pre-trained models such as BERT, GPT-3, and T5 offer unparalleled capabilities, with the flexibility to fine-tune for specific applications. From sentiment analysis to text summarization, AI Transformers revolutionize how we interact with and understand textual data. With open-source implementations readily available, these models democratize access to advanced NLP techniques, empowering developers to build sophisticated language-driven applications.

4.6 PACKAGES USED

Various packages have been employed in the development of the web-based system, tailored to the requirements of each individual module , such as :

A) User Registration and Verification

Some of the packages used for the implmentation of user registration and verification such as :

i) Nodejs - Nodemailer

Nodemailer stands out as a favored Node.js module, streamlining the otherwise intricate process of sending emails from Node.js applications. With its versatile and user-friendly API, Nodemailer offers developers a seamless experience for email transmission. The module supports multiple transport methods, including SMTP, sendmail, and direct transport, allowing users to choose the most suitable option for their application's requirements. This flexibility enables effortless integration of email sending functionality into Node.js applications, whether for transactional messages, notifications, or other communication needs. Nodemailer's widespread adoption and robust capabilities make it an indispensable tool for developers seeking efficient email handling within their Node.js projects.

B) Document Digital Signing and Verification

Some of the packages used for the implmentation of document digital signing and verification such as :

i) Node.js - Crypto

The crypto module serves as a fundamental tool for implementing SHA-256 hashing, a widely utilized cryptographic function essential for various tasks, notably generating hash values for text content in files. SHA-256 hashing, a member of the Secure Hash Algorithm (SHA-2) family, converts input data into a fixed-size string of characters, known as a hash value, with a length of 256 bits. This process is integral for ensuring data integrity, as even a minor change in the input data results in a significantly different hash value. By employing the crypto module, developers can seamlessly integrate SHA-256 hashing into their applications, providing a robust mechanism for verifying the integrity and authenticity of textual content within files.

ii) Node.js - Buffer

Node.js provides the Buffer class, allowing developers to efficiently handle raw binary data within their applications. This capability is particularly useful for tasks such as reading from or writing to files, where data is often represented in its binary form. Buffers in Node.js provide a fixed-size chunk of memory, which can be manipulated to store and manipulate binary data. Whether it's parsing files, working with network streams, or dealing with cryptographic operations, the Buffer class empowers developers with the tools they need to manage binary data effectively in their Node.js applications.

C) Document Encryption

Some of the packages used for the implmentation of document encryption
such as :

i) Nodejs - nodersa

The nodersa library in Node.js serves as a powerful toolset for handling RSA encryption and decryption tasks within applications. With nodersa, developers gain access to a range of functionalities essential for secure communication and data integrity. This includes the ability to generate RSA key pairs for asymmetric encryption, enabling secure transmission of sensitive information. Additionally, nodersa facilitates signing and verifying digital signatures, crucial for ensuring the authenticity and integrity of data exchanged between parties. By leveraging nodersa, Node.js developers can implement robust encryption mechanisms to protect their applications and data from unauthorized access and tampering, thereby enhancing overall security posture.

D) Document Upload

Some of the packages used for the implmentation of document upload such as :

i) Nodejs - express

Express is a widely-used web application framework for Node.js, prized for its minimalist design and remarkable flexibility. It offers developers a rich toolkit to craft both web and mobile applications effortlessly. With Express, building dynamic websites becomes a streamlined process, thanks to its comprehensive set of features. From routing and middleware integration to template engines and database connectivity, Express empowers developers to create robust, scalable, and efficient applications. Its simplicity and versatility make it a top choice for projects of all sizes, enabling developers to focus on building outstanding user experiences without getting bogged down by unnecessary complexity.

ii) Nodejs - Multer

Multer stands out as a crucial middleware within the Node.js ecosystem, specifically tailored for efficiently managing file uploads in web applications. Seamlessly integrated with Express.js, Multer simplifies the process of handling multipart/form-data, a common format used for file uploads. Its intuitive design and robust functionality enable developers to effortlessly handle file uploads, whether it's storing images, videos, documents, or any other type of file. Multer streamlines the parsing and handling of incoming file uploads, providing developers with a reliable solution for managing user-generated content. By leveraging Multer, Node.js developers can enhance the functionality of their web applications, offering users a seamless and responsive experience when uploading files.

iii) Nodejs - multer-gridfs-storage

Multer-gridfs-storage serves as a valuable extension to Multer, seamlessly integrating with MongoDB's GridFS storage system. This extension enhances Multer's capabilities by offering efficient storage engine support tailored specifically for handling large files within MongoDB. By leveraging multer-gridfs-storage, developers can streamline the process of storing and retrieving large files, such as images, videos, and documents, directly within their MongoDB database. This integration not only simplifies the management of file uploads but also ensures optimal performance and scalability for applications dealing with substantial amounts of user-generated content. With multer-gridfs-storage, Node.js developers can leverage the power of MongoDB's GridFS to efficiently handle file storage requirements, enhancing the robustness and scalability of their applications.

E) Keywords Extraction

Some of the packages used for the implmentation of keywords extraction such as :

i) Nodejs - stopwords

The stopwords package in Node.js serves as a valuable resource for natural language processing (NLP) tasks by offering a curated collection of common English stop words. These stop words, such as "the," "and," and "but," are ubiquitous in language but typically lack significant semantic meaning in the context of text analysis. By leveraging the stopwords package, developers can efficiently filter out these non-informative words from text data, enabling more accurate and focused analysis. Whether it's sentiment analysis, text classification, or keyword extraction, integrating stopwords into NLP pipelines helps streamline the process and enhances the quality of insights derived from textual data. With stopwords, Node.js developers can leverage a standardized approach to handling common linguistic constructs, facilitating more effective and insightful text analysis tasks.

iv) Nodejs - mammoth

The mammoth package in Node.js offers a streamlined solution for processing .docx files, enabling developers to effortlessly extract both content and formatting from Microsoft Word documents. By integrating mammoth into their applications, developers gain access to a powerful toolset for parsing and analyzing .docx files, allowing for seamless extraction of text, images, tables, and more. Additionally, mammoth preserves the formatting present in the original Word document, ensuring that the structure, styles, and layout are accurately retained during the extraction process. This capability proves invaluable for a wide range of applications, from content management systems and document conversion tools to data analysis pipelines. With mammoth, Node.js developers can efficiently handle .docx files, unlocking new possibilities for text processing and document management within their applications.

F) Document Compression - Lossless Compression

Some of the packages used for the implmentation of document compression of lossless compression such as :

i) Nodejs - zlib

The zlib module, an integral part of Node.js, provides powerful features for data compression and decompression. Leveraging zlib, developers can efficiently compress uploaded files before storing them in MongoDB Atlas. By reducing the size of files through compression, this approach offers benefits such as decreased storage requirements and minimized bandwidth usage, which can lead to cost savings and improved performance. However, it's essential to consider potential trade-offs, especially for certain file types where compression may not yield significant benefits or could even introduce overhead.

ii) Nodejs - stream

The Node.js stream module offers a robust solution for handling data in manageable chunks, presenting an efficient approach for processing large files. Particularly valuable during tasks such as uploads, encryption, or decryption, streams enable developers to manipulate data without the need to load the entire file into memory simultaneously. This capability not only conserves memory resources but also enhances the performance and scalability of applications dealing with substantial amounts of data. By leveraging streams, developers can implement more responsive and resource-efficient solutions for a variety of data processing tasks, ensuring optimal performance even when working with large files.

4.7 Functional and Non - Functional Requirements

A) Functional Requirements

The proposed system fulfills numerous functional requirements , such as :

i) User Management

- Users should be able to sign up for the system using a username, email address, and password.
- Users should be able to log in using their username or email address and password.

- The system should implement secure password hashing and storage.
- Implement functionalities like password reset or email verification.

ii) File Storage and Management

- Users should be able to upload files to the system.
- The system should support various file formats (e.g., documents, images, audio, video).
- Users should be able to view a list of their uploaded files with details like filename, size, upload date, and status (e.g., uploaded, verified, encrypted).
- Users should be able to download their uploaded files.

iii) Originality Check

- Users should be able to choose to perform an originality check on their uploaded files.
- The system should compare the uploaded file against a database of existing files or utilize an online service to identify potential plagiarism.
- The system should provide feedback to the user regarding the originality of the uploaded file.

iv) Encryption

- Users should be able to choose to encrypt their uploaded files for additional security.
- The system should implement a secure encryption algorithm (e.g., AES-256).
- Users should be able to set a password for decryption

v) AI News Feed/Train

- This feature could allow users to interact with an AI model in various ways:
- AI News Feed: Users should be able to personalize their news feed based on their preferences (e.g., selecting preferred topics). The AI model would then curate news articles relevant to these interests.

B) Non-Functional Requirements

The proposed system contains some non - functional requirements , such as :

i) Performance

- The system should be responsive and handle user requests efficiently.
- File upload and download times should be optimized, especially for larger files.

ii) Security

- The system should implement secure user authentication and authorization mechanisms.
- User data and uploaded files should be encrypted at rest and in transit.
- Regular backups should be performed to prevent data loss.

iii) Scalability

- The system should be able to accommodate a growing number of users and uploaded files.
- The backend infrastructure should be scalable to handle increased load.

iv) Usability

- The user interface should be intuitive and easy to use.
- Users should be able to easily navigate the system and find the functionalities they need.
- Clear instructions and feedback should be provided throughout the user experience.

CHAPTER 5
RESULTS AND DISCUSSION

5.1 DATASET DESCRIPTION

Two datasets utilized in the proposed system include the BBC news article dataset comprising text documents and a custom-designed dataset focusing on news and social impact.

5.1.1 Dataset - Text documents Ranking

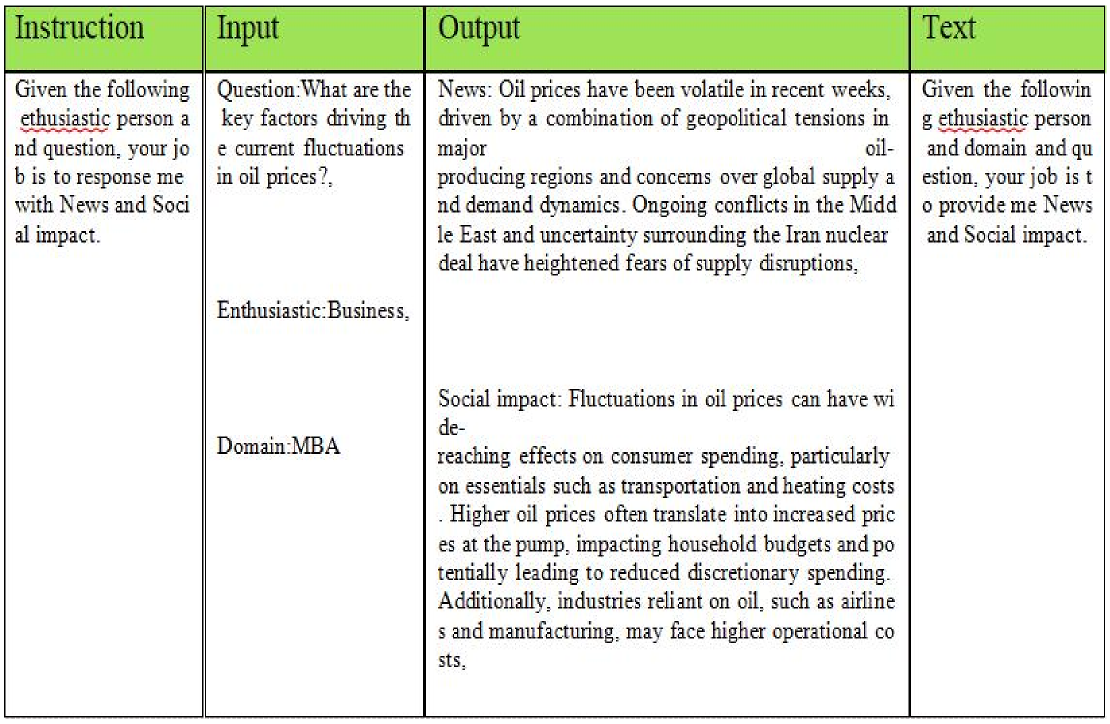
The BBC News dataset [12] is a highly regarded resource widely employed in natural language processing, particularly for text classification endeavors. It encompasses a collection of news articles meticulously categorized into five distinct classes: Business, Entertainment, Politics, Sport, and Tech. Each article is systematically labeled with its corresponding category, offering a diverse array of texts characterized by varying content and writing styles.

This dataset stands as an invaluable asset for both training and evaluating machine learning models tailored for text classification, sentiment analysis, and numerous language-related tasks. Its comprehensive nature facilitates the development and assessment of models capable of accurately categorizing news articles into their respective topics. Notably, the dataset comprises 2,225 documents sourced from the BBC news website, encompassing stories published across five topical areas from 2004 to 2005. Its natural class distribution is notably balanced, ensuring an equitable representation of articles across categories.

This balanced distribution proves instrumental in fostering fair and unbiased model training, preventing any inherent bias toward a specific class. The dataset's composition into five natural classes - business, entertainment, politics, sport, and tech - further amplifies its relevance and usability. This diversity allows for robust model training and evaluation across distinct categories, reflecting real-world scenarios and enhancing the model's adaptability to different types of news content.

5.1.2 Custom Designed Dataset

Table 5.1 - Custom Designed Dataset



Instruction	Input	Output	Text
Given the following ethusiastic person and question, your job is to response me with News and Social impact.	Question:What are the key factors driving the current fluctuations in oil prices?, Enthusiastic:Business, Domain:MBA	News: Oil prices have been volatile in recent weeks, driven by a combination of geopolitical tensions in major oil-producing regions and concerns over global supply and demand dynamics. Ongoing conflicts in the Middle East and uncertainty surrounding the Iran nuclear deal have heightened fears of supply disruptions, Social impact: Fluctuations in oil prices can have wide-reaching effects on consumer spending, particularly on essentials such as transportation and heating costs. Higher oil prices often translate into increased prices at the pump, impacting household budgets and potentially leading to reduced discretionary spending. Additionally, industries reliant on oil, such as airlines and manufacturing, may face higher operational costs,	Given the following ethusiastic person and domain and question, your job is to provide me News and Social impact.

When prompted with a question from the user, the AI model processes the input and generates outputs in the form of news articles, detailing the social impact of the subject matter. This innovative tool not only streamlines the news writing process but also ensures that the resulting paragraphs are coherent, informative, and relevant to the interests of the intended audience. By harnessing the power of artificial intelligence, journalists can produce compelling stories that resonate with readers, addressing pressing issues and shedding light on important developments across diverse domains.

5.2 EVALUATION METRICS

Multiple evaluation metrics are employed to calibrate the proposed system such as:

i) Precision

Precision measures the accuracy of the positive predictions made by a model.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \quad - (5.1)$$

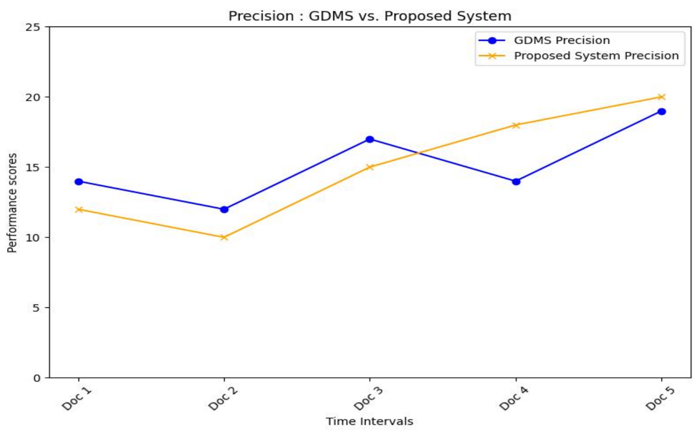


Figure 5.1 - Precision Scores Result

The visual comparison of the precision scores of the GDMS and the proposed system over the specified time intervals which is illustrated in Figure 5.1. The plot showcases how their performance varies or compares across different document IDs or time periods. Interpretation of this plot could help in assessing the effectiveness or improvements of the proposed system over the existing GDMS across various time intervals in document management tasks.

ii) Recall (Sensitivity)

Recall measures the proportion of actual positives that were correctly identified.

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives} \qquad - (5.2)$$

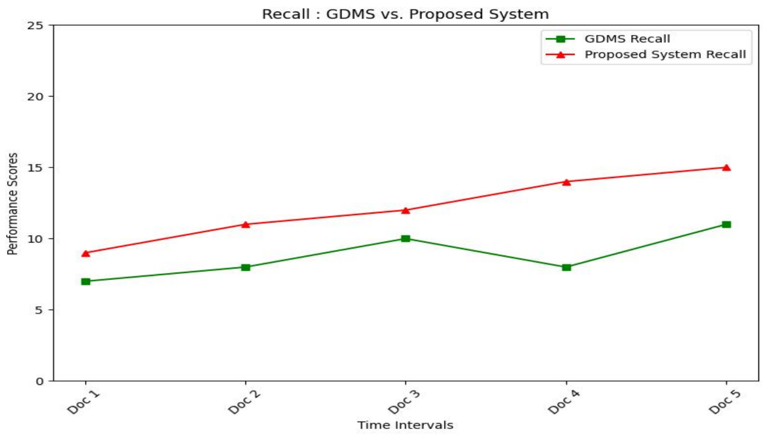


Figure 5.2 - Recall Scores Result

The visual showcases of the recall scores of both systems across different time intervals which is illustrated in Figure 5.2. Analyzing this plot helps in understanding how effectively each system retrieves relevant documents over time or document IDs. It enables comparison and evaluation of the systems' retrieval capabilities, assisting in determining which system performs better in terms of recall for document management tasks over the specified intervals

iii) F1 Score

The F1 score is the harmonic mean of precision and recall, providing a single score that balances both metrics.

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \qquad \text{- (5.3)}$$

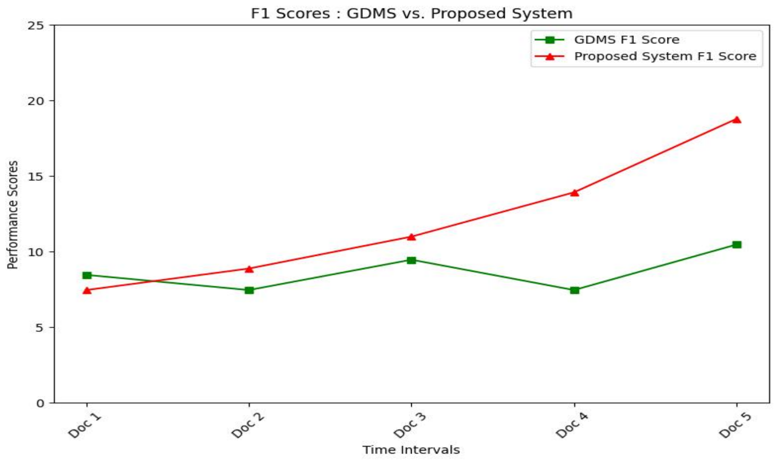


Figure 5.3 - F1 Scores Result

The visual comparison of the F1 scores of both systems across different time intervals which is illustrated in Figure 5.3. This comparison aids in evaluating the overall performance balance between precision and recall for each system over time. Analyzing this plot helps in understanding which system maintains a better balance between precision and recall, crucial for effective document management across various time intervals.

iv) Mean Reciprocal Rank (MRR)

MRR measures the rank of the first correct result averaged across multiple queries.

$$MRR = \frac{1}{Number\ of\ Queries} \sum_{i=1}^{Number\ of\ queries} \frac{1}{Rank_i} \qquad \text{- (5.4)}$$

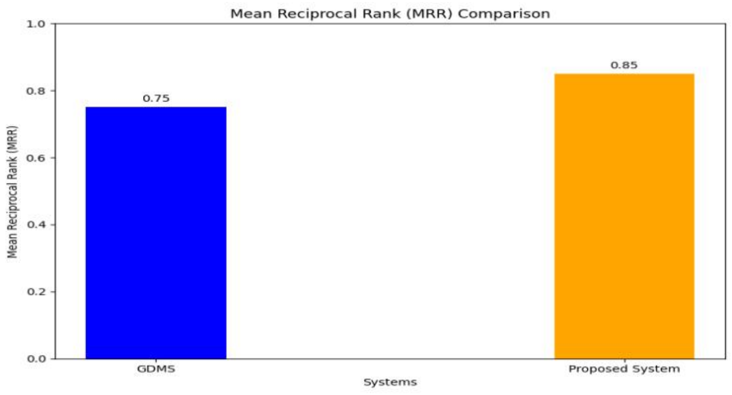


Figure 5.4 - Mean Reciprocal Result

The visual comparison of the MRR values between GDMS and the Proposed System which is illustrated in Figure 5.4. Analyzing this plot allows for a direct comparison of the ranking quality between the two systems. In this case, the Proposed System exhibits a higher MRR value compared to GDMS, indicating potentially better document retrieval ranking in the proposed system

v) Time Complexity

Time complexity quantifies the amount of time an algorithm takes to run as a function of the input size.

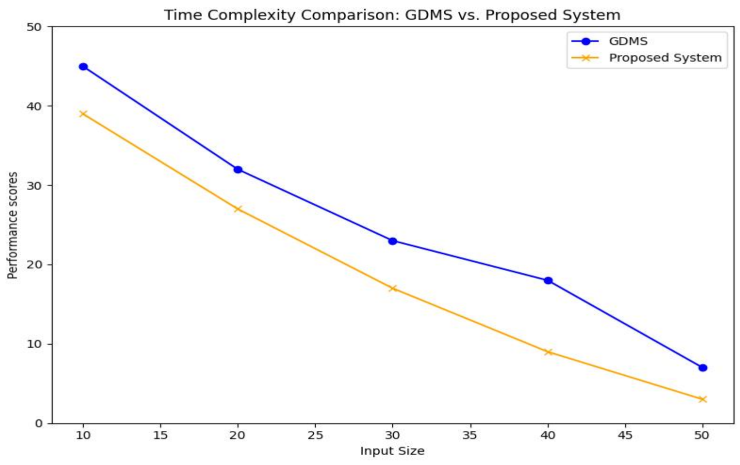


Figure 5.5 - Time Complexity Result

The visual representation of time complexity trends for both systems, specifically focusing on document processing, illuminates how the time required for operations evolves with varying input sizes which is illustrated in Figure 5.5. This plot serves as a valuable tool for understanding the efficiency and scalability of both systems in managing document tasks. By scrutinizing these trends, it becomes possible to compare and evaluate the performance scalability of each system, specifically in the context of document processing. This comparison aids in determining which system proves more efficient or scalable for handling document-related tasks across varying input sizes.

vi) Speed of Retrieval

It measures the time taken to retrieve or access information/data.It can be measured in terms of time (e.g., seconds) taken to retrieve a set of documents or information.

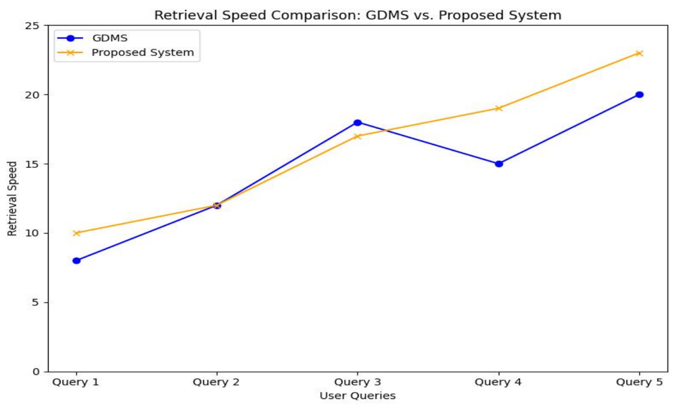


Figure 5.6 - Retrieval Speed

The visual representation of the retrieval speeds of both systems across different user queries which is illustrated in Figure 5.6. This comparison aids in understanding how efficiently each system retrieves information in response to various user queries. Analyzing this plot helps in evaluating and comparing the retrieval speed performance of the systems, allowing for insights into which system might perform better in terms of speed when handling different user queries in a document management context.

vii) Error Rate

The error rate measures the accuracy or deviation of predicted values from the true values.

$$Error\ Rate = \frac{Number\ of\ errors}{Total\ Number\ of\ samples} \qquad \text{- (5.5)}$$

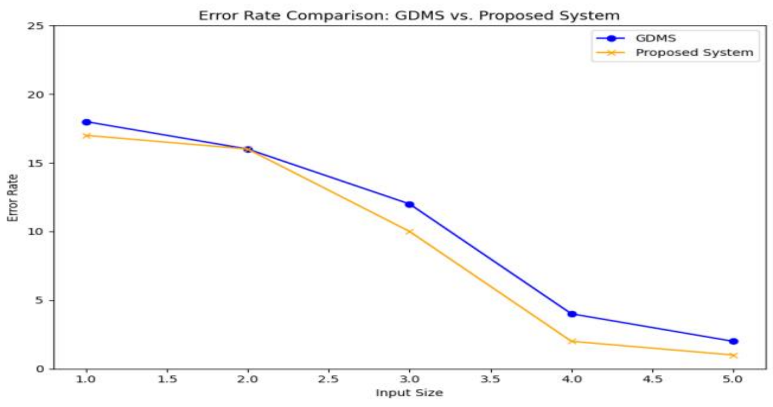


Figure 5.7 - Error Rate

The visual comparison of the error rates of both systems across different input sizes which illustrated in Figure 5.7. This comparison aids in evaluating and comparing the predictive accuracy or performance of GDMS and the Proposed System, helping to understand which system demonstrates better accuracy in predicting values closer to the true values across various input sizes.

viii) Compression Speed:

Compression speed represents the rate at which a lossless compression algorithm processes input data and produces compressed output. It is typically measured in terms of data throughput or processing time.

$$\text{Compression Speed} = \frac{\text{Size of data before compression} - \text{Size of data after compression}}{\text{Time taken for Compression}} \qquad \text{- (5.6)}$$

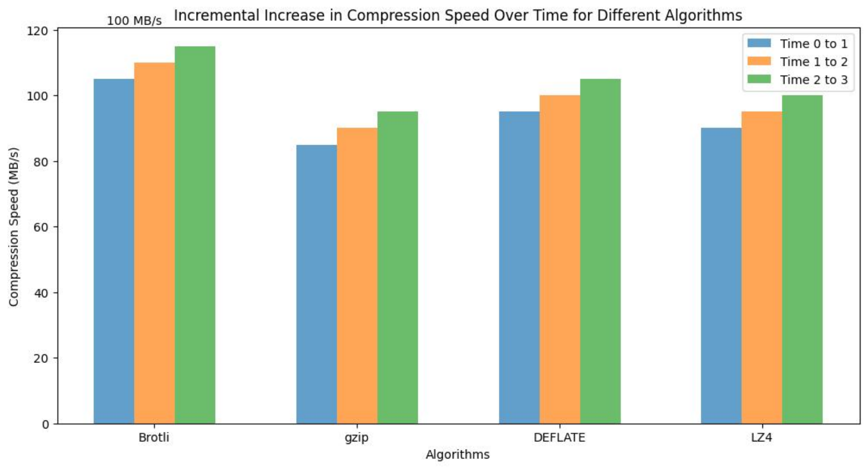


Figure 5.8 - Comparison of Compression speed

The visual comparison of the compression speeds of different lossless compression algorithms across various input sizes of time(seconds) which is illustrated in Figure 5.8. This comparative analysis serves as a valuable tool for evaluating and contrasting the performance of each algorithm in terms of processing speed. By examining the compression speed curves of each algorithm, one can gain insights into their efficiency and effectiveness across different data sizes.

ix) Compression Ratio rate:

The compression ratio rate comparison assesses the efficiency of various lossless compression algorithms in terms of their ability to achieve data reduction while maintaining information integrity.

$$\text{Compression Ratio rate} = \frac{\text{Original data size}}{\text{Compressed data size}} \qquad \text{- (5.7)}$$

$$\text{Compression Percentage} = \left(1 - \frac{\text{Compressed file size}}{\text{Original file size}}\right) \times 100 \qquad \text{- (5.8)}$$

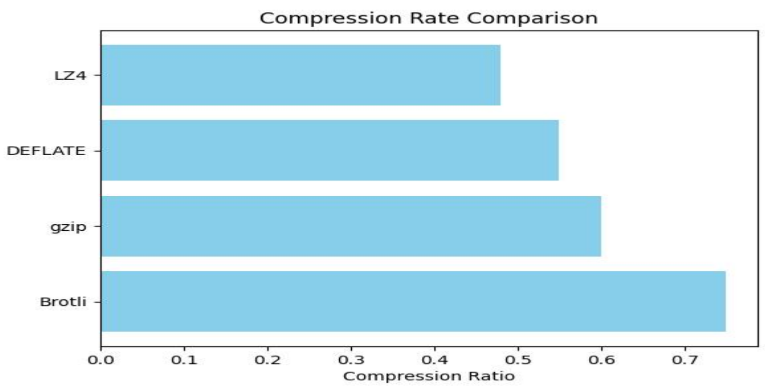


Figure 5.9 - Comparison of Compression ratio rate

The visual depiction provides a comprehensive comparison of the compression ratio rates among distinct lossless compression algorithms across varying input sizes measured in time (seconds) which is illustrated in Figure 5.9. This graphical representation serves as a crucial analytical tool, facilitating the assessment and differentiation of each algorithm's performance in terms of data reduction efficacy. By scrutinizing the compression ratio rate curves of individual algorithms, observers can glean valuable insights into the efficiency and effectiveness exhibited across diverse data sizes.

5.3 COMPARISON - RANKING ALGORITHMS

Table 5.2 - Comparison of Ranking Algorithms

Algorithms	Ranking Percentage (%)
Modern BM25	85
Okapi BM25	70
TF-IDF (Term Frequency-Inverse Document Frequency)	62
Divergence from Randomness (DFR) Models	57

5.4 COMPARISON - GDMS vs PROPOSED SYSTEM

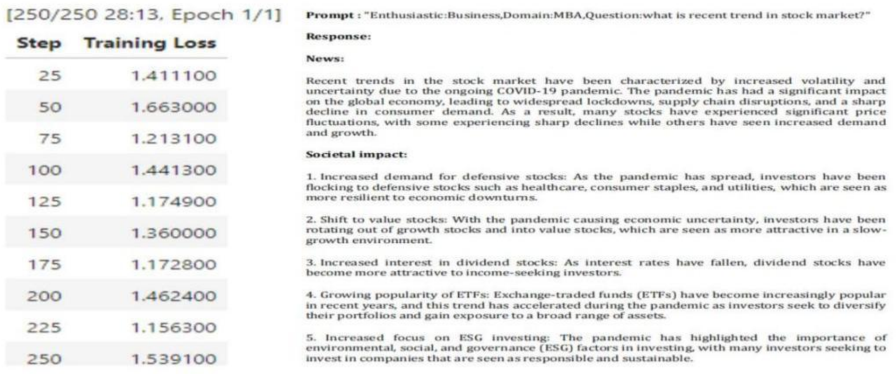
In the domain of document management systems, the General Document Management System (GDMS) plays a pivotal role, offering essential functionalities for structured data management, retrieval, and security. With a comprehensive suite of features, GDMS ensures organized storage and systematic handling of documents, prioritizing data security to safeguard sensitive information from unauthorized access or breaches. The proposed system closely mirrors GDMS in various aspects, demonstrating commendable performance in structured management, document handling, and security protocols. Designed to emulate GDMS's core functionalities, the system emphasizes accuracy and reliability in data and document management, maintaining a comparable level of security infrastructure to protect critical information. Comparing GDMS with the proposed system, both exhibit moderate yet accurate performances across essential functionalities crucial for system evaluation. The system, like GDMS, delivers consistent and reliable functionalities in structured management, document handling, and verification, ensuring moderate but accurate management of data and documents. Both systems maintain a comparable level of security, prioritizing the protection of sensitive information. Moreover, the access speed of the proposed system, similar to GDMS, is noted to be moderate, highlighting a balanced efficiency without compromising accuracy in managing and handling documents. While neither system excels in rapidity, the dependable and accurate performance of the proposed system signifies steady reliability in its operations across these categories.

5.5 AI INTERACTIVE MODEL

The evaluation metrics and a sample outcome for the AI interactive model are outlined below.

i) Training Loss and Sample Interation

The evaluation metrics for the AI model encompass both the training loss pertaining to the model's output and describe the user's interaction with the trained AI model.



[250/250 28:13, Epoch 1/1]

Step	Training Loss
25	1.411100
50	1.663000
75	1.213100
100	1.441300
125	1.174900
150	1.360000
175	1.172800
200	1.462400
225	1.156300
250	1.539100

Prompt : "Enthusiastic:Business,Domain:MBA,Question:what is recent trend in stock market?"

Response:

News:

Recent trends in the stock market have been characterized by increased volatility and uncertainty due to the ongoing COVID-19 pandemic. The pandemic has had a significant impact on the global economy, leading to widespread lockdowns, supply chain disruptions, and a sharp decline in consumer demand. As a result, many stocks have experienced significant price fluctuations, with some experiencing sharp declines while others have seen increased demand and growth.

Societal impact:

1. Increased demand for defensive stocks: As the pandemic has spread, investors have been flocking to defensive stocks such as healthcare, consumer staples, and utilities, which are seen as more resilient to economic downturns.

2. Shift to value stocks: With the pandemic causing economic uncertainty, investors have been rotating out of growth stocks and into value stocks, which are seen as more attractive in a slow-growth environment.

3. Increased interest in dividend stocks: As interest rates have fallen, dividend stocks have become more attractive to income-seeking investors.

4. Growing popularity of ETFs: Exchange-traded funds (ETFs) have become increasingly popular in recent years, and this trend has accelerated during the pandemic as investors seek to diversify their portfolios and gain exposure to a broad range of assets.

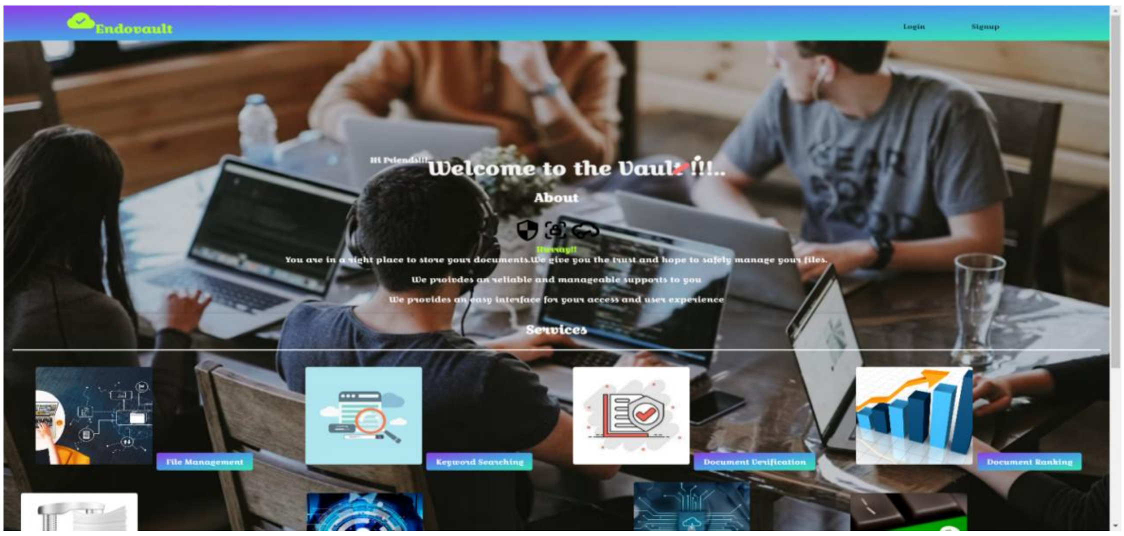
5. Increased focus on ESG investing: The pandemic has highlighted the importance of environmental, social, and governance (ESG) factors in investing, with many investors seeking to invest in companies that are seen as responsible and sustainable.

Figure 5.10 - Training Loss and Sample Interaction

The above Figure 5.10 depicts the training losss variation for each and every 25 epochs and also the sample result of interaction of predicted frame of answer from the AI chat Llama model .

5.6 WEB APPLICATION

The proposed system offers a variety of available set of services.



You are in a right place to store your documents.We give you the trust and hope to safely manage your files.

We proivdes an reliable and manageable supports to you

We provides an easy interface for your access and user experience

Figure 5.11 - It represents the Welcome page of the Web Application

The platform is designed to make the experience seamless and enjoyable which is illusrated in Figure 5.11.The web application gives the better user experience and it also help the user to store and maintains their valuable source of documents. With robust storage and maintenance capabilities, you can securely store, organize, and access the files anytime, anywhere.

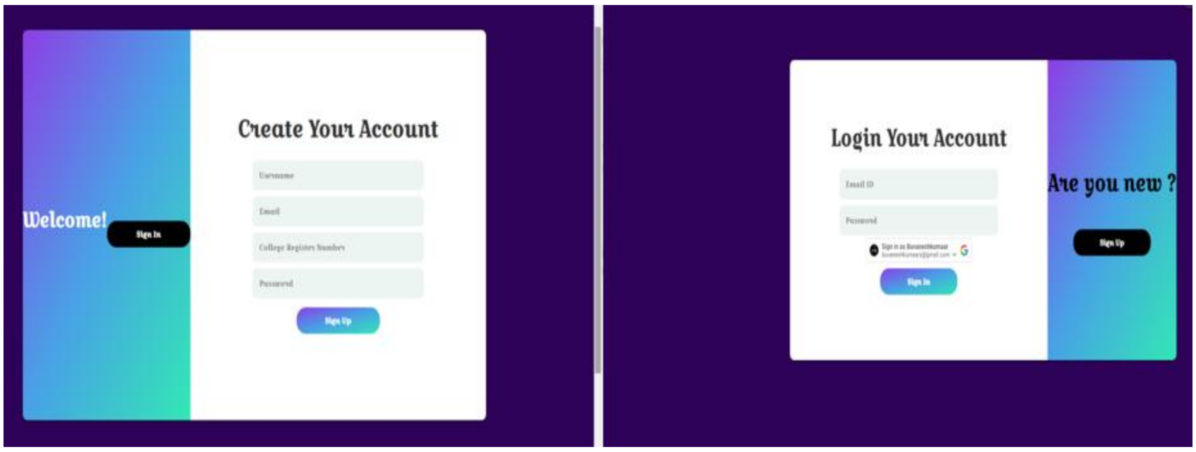
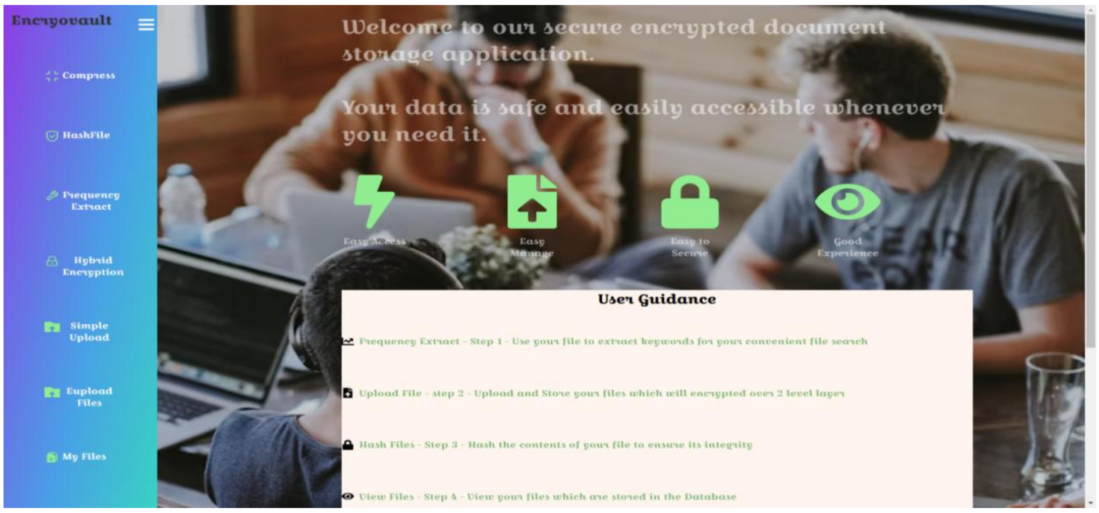


Figure 5.12- It represents the Authorization & Authentication of the Web Application

To prioritize the security and privacy of the data above all else. the platform employs state-of-the-art authentication mechanisms to ensure that only authorized users gain access to sensitive information which is illusrated in Figure 5.12.Rest assured that the personal and confidential data is safeguarded with the utmost diligence. It uses the highest standards of security and trust in every interaction.



Welcome to our secure encrypted document storage application.

Your data is safe and easily accessible whenever you need it.

User Guidance

Frequency Extract - Step 1 - Use your file to extract keywords for your convenient file search

Upload File - step 2 - Upload and Store your files which will encrypted over 2 level layer

Hash Files - Step 3 - Hash the contents of your file to ensure its integrity

View Files - Step 4 - View your files which are stored in the Database

Figure 5.13 - It represents the List of Services in the Web Application

The array of services offered by the Web Application, tailored to meet the diverse needs. From advanced keyword extraction to seamless storage solutions, It covered at every step which is illusrated in Figure 5.13. The encryption and hashing services ensure the information remains confidential and tamper-proof. The full range of services is tailored to enhance your operations, whether it's document management, trend analysis, or protecting sensitive data, ensuring an elevated experience and empowered operations.

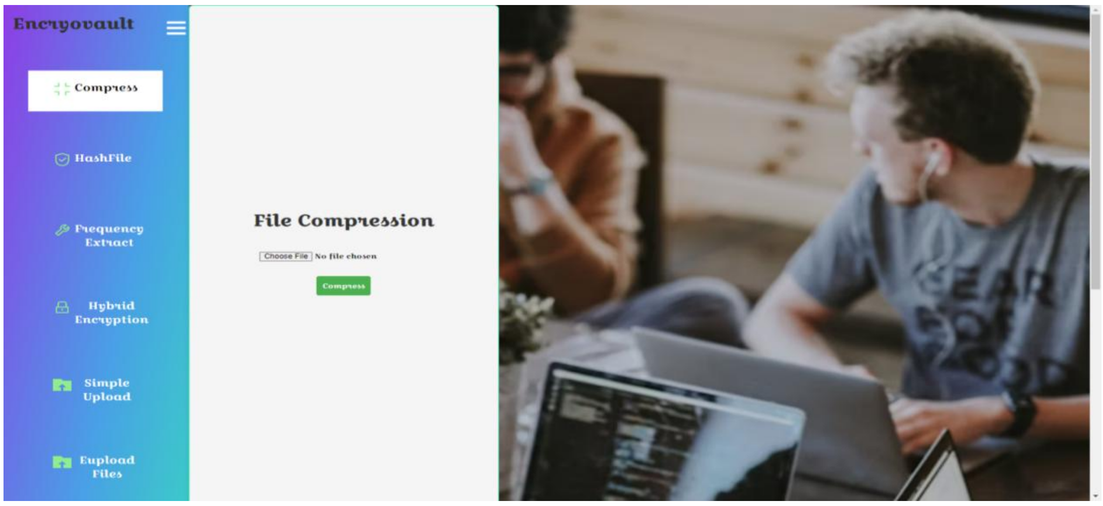


Figure 5.14 - It represents the File Compression Service in the Web Application

Introducing the File Compression Service, designed to streamline the data management experience. With the advanced compression algorithms, we ensure that the files are efficiently condensed without compromising on quality which is illusrated in Figure 5.14. Experience faster uploads, smoother downloads, and optimized storage utilization with the File Compression Service.

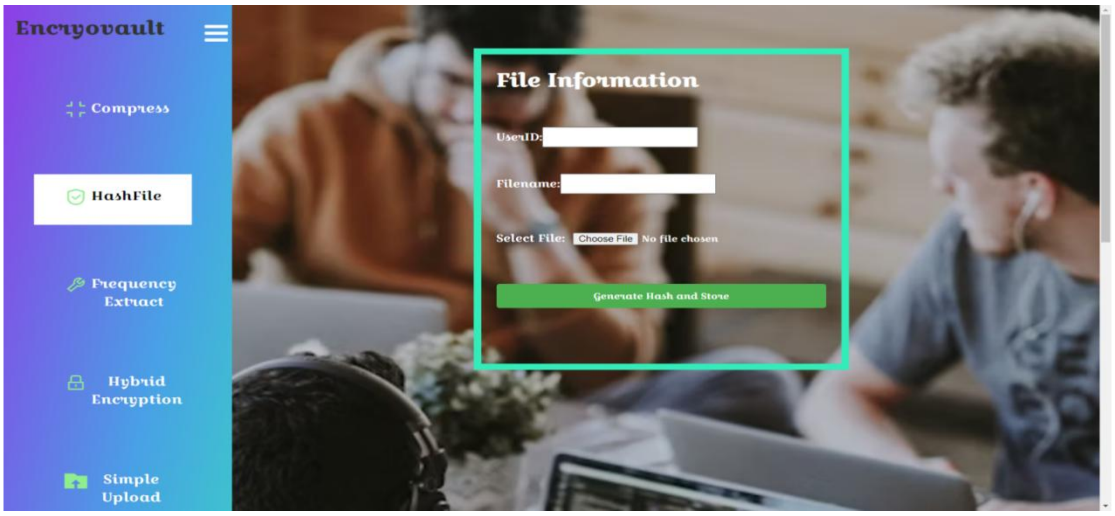
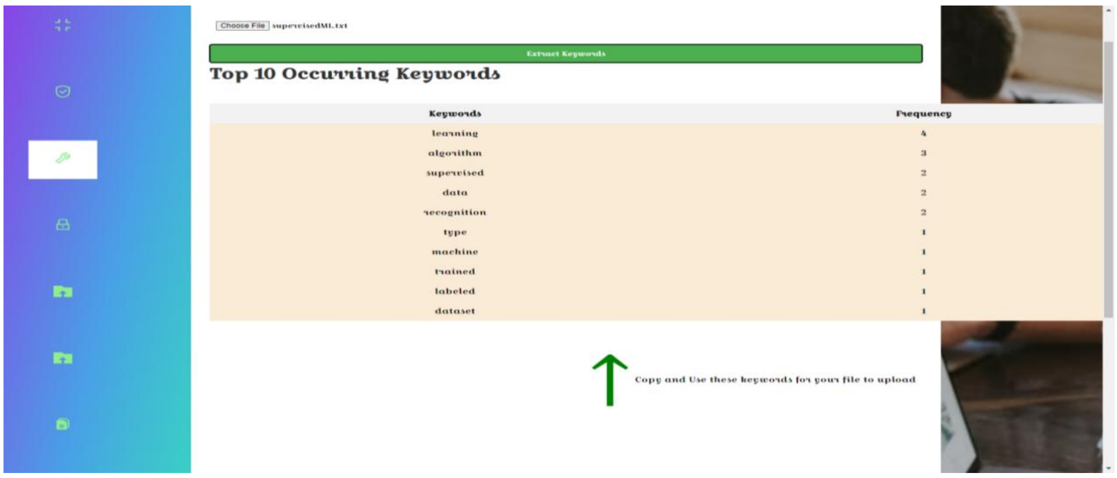


Figure 5.15 - It represents the File Hashing Service in the Web Application

With the robust hashing algorithms, we generate unique cryptographic hash values for the files, providing a digital fingerprint that ensures data authenticity and integrity which is illusrated in Figure 5.15. Whether you're sharing sensitive documents or verifying the integrity of critical files, the service offers peace of mind through tamper-proof identification. Simply upload the files, and let the system generate secure hash values that serve as a digital seal of trust.



Top 10 Occurring Keywords

Keywords	Frequency
learning	4
algorithm	3
supervised	2
data	2
recognition	2
type	1
machine	1
trained	1
labeled	1
dataset	1

Copy and Use these keywords for your file to upload

Figure 5.16 - It represents the keyword Extraction Service in the Web Application

Discover the Keyword Extraction Service, a powerful tool to unlock insights from the textual data which is illusrated in Figure 5.16. Using advanced natural language processing techniques, we automatically identify and extract key terms and phrases from the documents, enabling to uncover valuable patterns and trends.With the intuitive interface and customizable options, extracting meaningful insights has never been easier. Unleash the potential of the textual data with the Keyword Extraction Service.

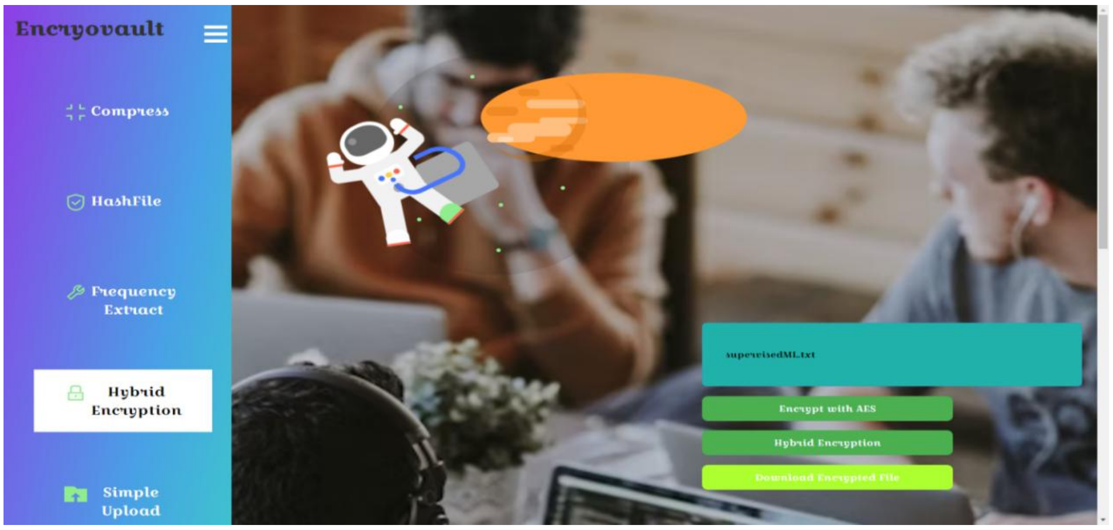


Figure 5.17 - It represents the Hybrid Encryption Service in the Web Application

Introducing the Hybrid Encryption Service, a cutting-edge solution for safeguarding the data with the utmost security which is illusrated in Figure 5.17. Combining the strengths of both symmetric and asymmetric encryption, the service offers unparalleled protection for the sensitive information. With symmetric encryption handling the bulk of the data and asymmetric encryption managing the keys, we ensure a perfect balance of efficiency and security.

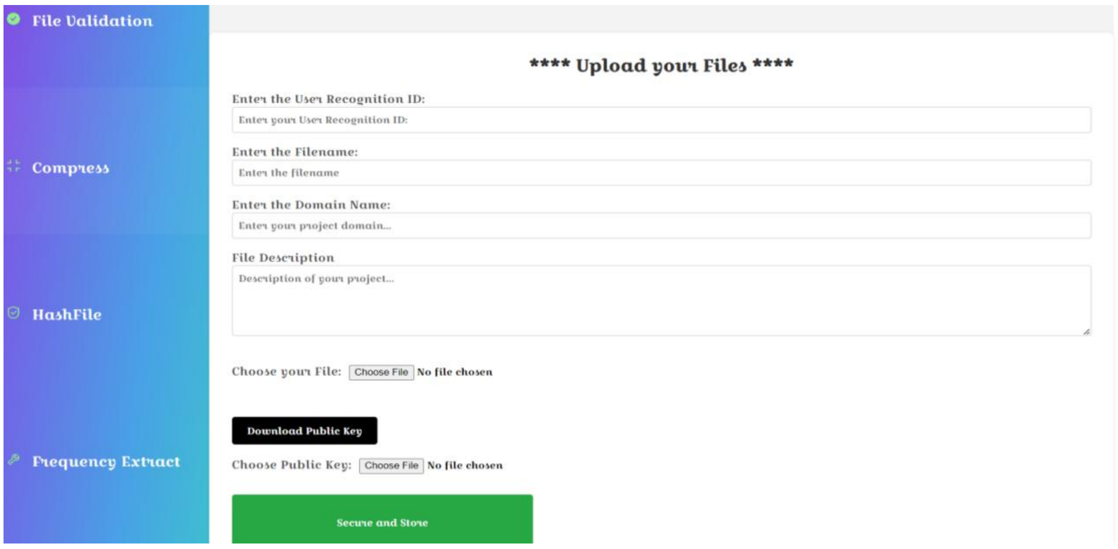
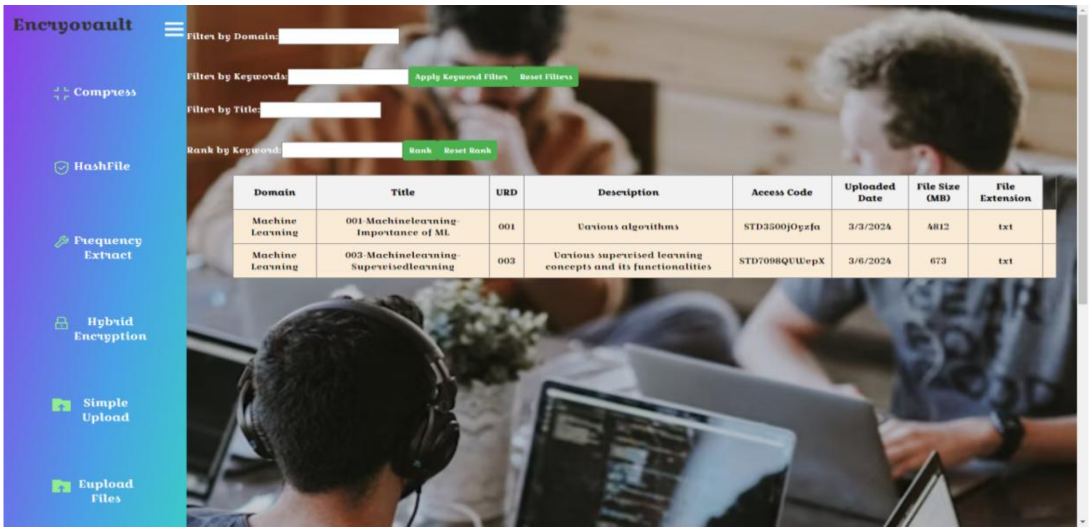

File Validation
Compress
HashFile
Frequency Extract
**** Upload your Files ****
Enter the User Recognition ID:
Enter your User Recognition ID:
Enter the Filename:
Enter the filename
Enter the Domain Name:
Enter your project domain...
File Description
Description of your project...
Choose your File: Choose File No file chosen
Download Public Key
Choose Public Key: Choose File No file chosen
Secure and Store

Figure 5.18 - It represents the File Upload Service in the Web Application

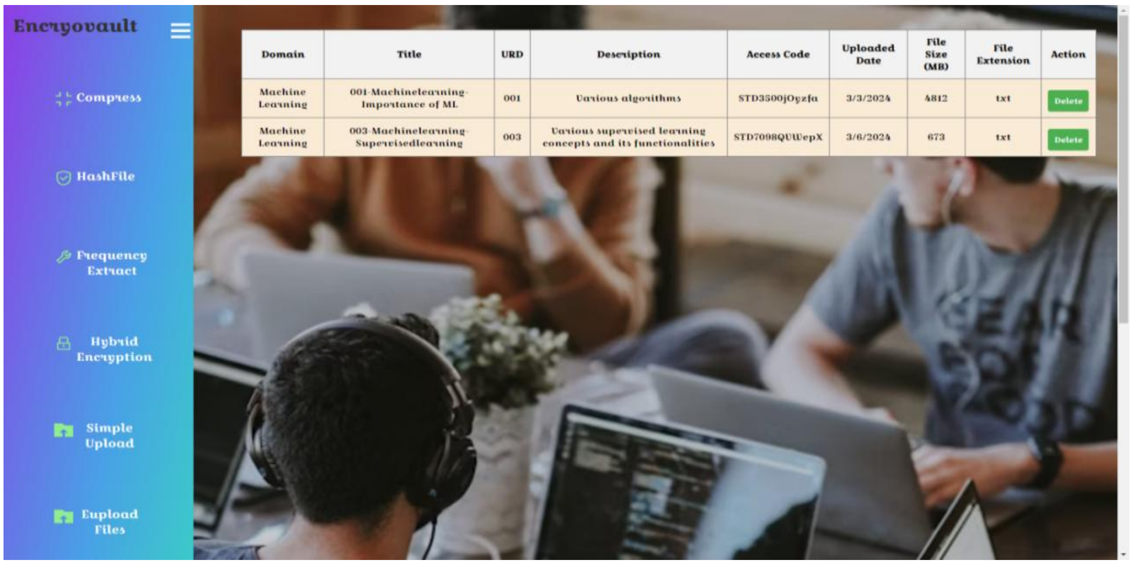
Simplifying the process of transferring the files securely which is illusrated in Figure 5.18. With the intuitive interface, you can effortlessly upload documents, images, videos, and more, directly to the platform. With robust security measures in place, the uploads are encrypted and protected from unauthorized access. Experience convenience and peace of mind as you effortlessly transfer the files with the File Upload Service.



Domain	Title	URD	Description	Access Code	Uploaded Date	File Size (MB)	File Extension
Machine Learning	001-Machinelearning-Importance of ML	001	Various algorithms	STD3500jOyzfa	3/3/2024	4812	txt
Machine Learning	003-Machinelearning-Supervisedlearning	003	Various supervised learning concepts and its functionalities	STD7098QUWepX	3/6/2024	673	txt

Figure 5.19 - It represents the List of Files Stored in the Cloud Server

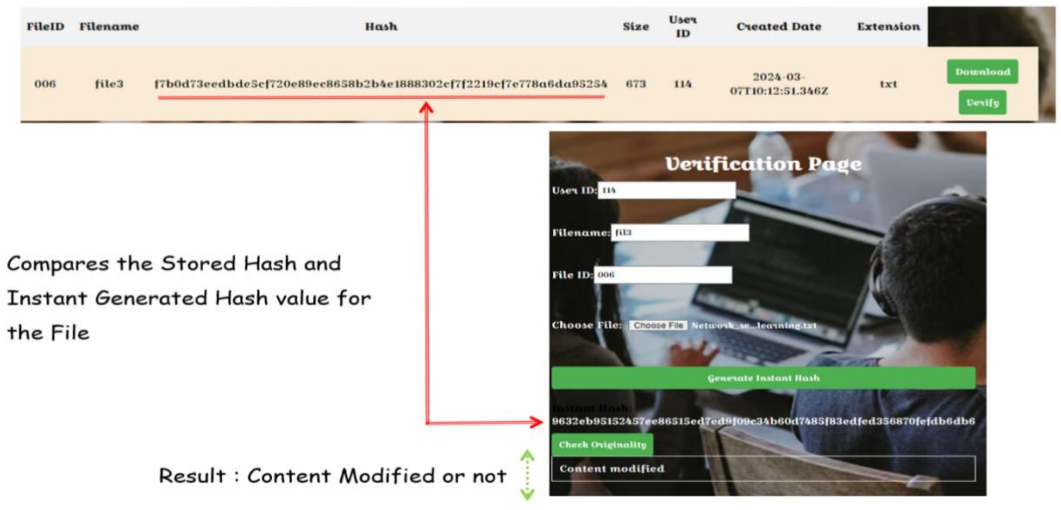
The extensive array of files securely stored on the Cloud Server which is illusrated in Figure 5.19. With robust security measures and reliable data backup protocols in place, you can trust that the files are safe and accessible whenever you need them. Experience the convenience of centralized storage and access the files from anywhere, at any time, with the Cloud Server.



Domain	Title	URD	Description	Access Code	Uploaded Date	File Size (MB)	File Extension	Action
Machine Learning	001-Machinelearning-Importance of ML	001	Various algorithms	STD3500jOyzfa	3/3/2024	4812	txt	Delete
Machine Learning	003-Machinelearning-Supervisedlearning	003	Various supervised learning concepts and its functionalities	STD7098QUWepX	3/6/2024	673	txt	Delete

Figure 5.20 - It represents the List of Files Stored in the Cloud Server - Remove Files if not needed

Discover the flexibility of the Cloud Server's file management capabilities, allowing to curate and optimize the storage space effortlessly which is illusrated in Figure 5.20. remove files that are no longer needed. Whether it is outdated documents, redundant files, or temporary data, the platform empowers you to efficiently clean up the storage environment



FileID	Filename	Hash	Size	User ID	Created Date	Extension	
006	file3	f7b0d73eedbde5cf720e89ec8658b2b4e1888302cf7f2219cf7e778a6da95254	673	114	2024-03-07T10:12:51.346Z	txt	Download Verify

Verification Page

User ID: 114

Filename: fil3

File ID: 006

Choose File: Choose File Network_se...learning.txt

Generate Instant Hash

Instant Hash:
9632eb95152457ee86515ed7ed9f09e34b60d7485f83edfed356870fefdb6db6

Check Originality

Content modified

Compares the Stored Hash and
Instant Generated Hash value for
the File
Result : Content Modified or not

Figure 5.21 - It represents the Checking of File Content's Originality

The power of the Originality Checking service, designed to ensure the authenticity and integrity of the file content which is illusrated in Figure 5.21. Using advanced algorithms and comparison techniques, we meticulously analyze the documents to detect any instances modification.Trust in the Originality Checking service to uphold the highest standards of credibility and authenticity in the content.

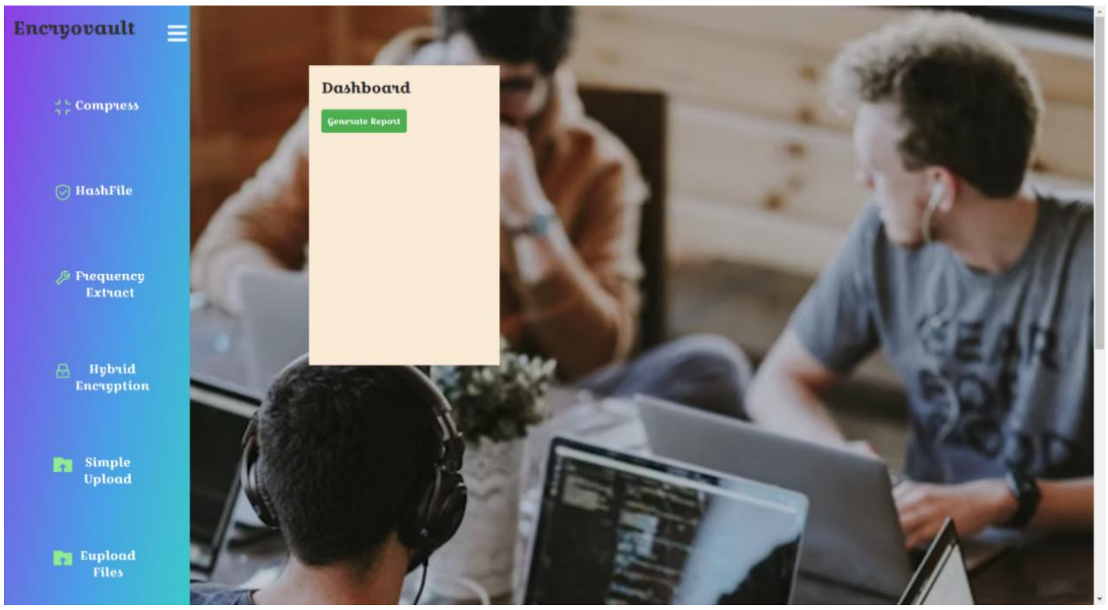


Figure 5.22 - It represents the report generation page

By analyzing data trends, or summarizing key findings, the intuitive interface empowers to generate professional reports with ease which is illusrated in Figure 5.22.

CHAPTER 6

CONCLUSION

The harmonious collaboration between cloud computing and AI-driven chat models marks a pivotal moment in the evolution of document management, ushering in a wave of transformative capabilities that bolster scalability, flexibility, and accessibility. By capitalizing on standardized formats and structured data organization within cloud environments, businesses can optimize operational efficiency to unprecedented levels. Moreover, the implementation of advanced digital signature techniques and AI-powered verification mechanisms ensures the integrity and security of documents, instilling confidence in their authenticity.

The integration of AI chat models further enhances this paradigm shift by enriching user experiences with dynamic organization, retrieval, and content creation functionalities. This fusion of technologies not only streamlines document management processes but also elevates user interaction to new heights, fostering a more intuitive and engaging experience.

As a result, this amalgamation of cloud computing and AI-driven chat models redefines traditional document management practices, offering unparalleled efficiency, security, and user-centric functionalities. By embracing these innovations, businesses are propelled into a new era of productivity and competitiveness, equipped with the tools necessary to thrive in today's dynamic digital landscape.

CHAPTER 7

SOCIAL IMPACT

Remote learning accessibility for students has become increasingly important in today's educational landscape. It involves several key aspects:

- **Domain-based Collaborative Learning Opportunities:** Students benefit from collaborative learning experiences tailored to specific domains. Whether it's science, literature, or mathematics, remote learning platforms provide opportunities for students to engage with peers and educators, fostering a deeper understanding of the subject matter.

- **Efficient Document Management:** Remote learning necessitates effective document management systems. These systems ensure that educational materials, assignments, and resources are organized and easily accessible to students and educators alike, streamlining the learning process.

- **Enhanced Data Security and Authenticity:** With the transition to digital learning environments, ensuring the security and authenticity of data and assessments is paramount. Robust data security measures protect students' personal information and academic integrity, fostering trust in remote learning platforms.

- **Scalability for Growing Educational Needs:** Remote learning platforms must be scalable to accommodate growing educational needs. As more students and institutions adopt remote learning, platforms should be able to scale resources and services accordingly to ensure uninterrupted access to education.

- **Useful for Researchers and Content Creators:** Remote learning platforms are not only beneficial for students but also for researchers and content creators. These platforms provide valuable insights into learning trends and preferences, enabling researchers to enhance educational methodologies. Additionally, content creators leverage these platforms to develop engaging educational materials tailored to remote learning environments.

APPENDIX I

The successful implementation of the project relies on specific software and hardware components to ensure smooth operation and optimal performance. This section outlines the necessary system requirements, including software dependencies, hardware specifications, and deployment considerations.

i) Hardware Requirements

The project requires the following hardware specifications:

1. Processor: Inter Core I5

2. Ram: 16GB

3. Storage: 100Gb free disk space

ii) Software Requirements

The following software components are required for the development and execution of the project:

1. **React 16.8:** JavaScript library for building the project's frontend.

2. Node JS: A JavScript runtime environment for backend data processing and storage

3. Python 3.11 or higher: Programming language for development and also includes transformers in LLM (Llama and LoRa - Hugging Face Importation)

APPENDIX II

SOURCE CODES:

Server.js:

```
const express = require('express');
const cors = require('cors');
const mongoose = require('mongoose');
const fileUpload = require('express-fileupload');
const Grid = require('gridfs-stream');
const zlib = require('zlib');
const multer = require('multer');const crypto = require('crypto');
const forge = require('node-forge');
const app = express();
app.use(cors());
app.use(express.json());
app.use(fileUpload()); Use express-fileupload middleware
multer().single('file');
mongoose.set('strictQuery', 'true');
let gfs;
mongoose.connect("mongodb+srv:sihworks001:schlenkians@cluster0.fbg8aof.mongodb.net/
endovault?re
tryWrites=true&w=majority&appName=AtlasApp", { useNewUrlParser: true,
useUnifiedTopology: true });
const connection = mongoose.connection;
connection.once('open', () => {
console.log("Database gets connected");
gfs = Grid(connection.db, mongoose.mongo);
gfs.collection('Documents');
console.log('GridFS is ready');
});
const PORT = process.env.PORT || 8080;
const MAX_FILE_SIZE_BYTES = 10 * 1024 * 1024; 10 MB
const RSA_PUBLIC_KEY = `
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAvpG3s2nHh0yK8OmFUY
WZ
MF+aDtKf0Ei/0i9t0Xo2kw/7VfIhqjN/MG8p3ANoSY2qNz93Ssf0Kw6GyUP9pmwF
XblyS8rY8fl1Oxlq+XDSjO4s+U6t/2WwDpRqJArU/KUt61y3A6LJgdZaX5iFz5jl
4VRz/B/3ouY9X7/9hC6HKUJssu9rhFrLlL5PIRVlfYvNjy0u8jOaVTNGDF5HV3A7
rg8YLG4F7fs8lcM2v55BW3K2O/2LlSyOk6lDsfgI+JAvxWV+nExZJ8pNnygEnVsI
uDfbJl76fFTT5rLnpV6oOzHmN0CLHEaQr3/FRb7dbzMIhBwTbZgV4bxEJ+wfylUe
iQIDAQAB
-----END PUBLIC KEY-----
`;
COMPRESSION - AREA
app.post('/compress', (req, res) => {
try {
if (!req.files || !req.files.file) {
console.log("No file received on the server");
return res.status(400).json({ error: 'File is required.' });
}
const file = req.files.file;
if (file.size > MAX_FILE_SIZE_BYTES) {
```

```
console.log("File size exceeds the limit");
return res.status(400).json({ error: 'File size exceeds the limit.' });}
console.log("File received on the server", file);
Set the filename in the Content-Disposition header
res.setHeader('Content-Disposition', `attachment; filename=${file.name.replace(/\.[^/.]+$/,
"")}.gz`);
res.setHeader('Content-Type', file.mimetype); Set the original file type
Compress the file using zlib and send it as a stream
const compressedStream = zlib.createGzip();
const readStream = new stream.PassThrough();
readStream.end(file.data);
readStream.pipe(compressedStream).pipe(res);
console.log("Compression done and sent!");
} catch (error) {
console.error('Error compressing file', error);
res.status(500).json({ error: 'Internal Server Error' });
}
});
AES - ENCRYPT AREA
app.post('/aesencrypt', async (req, res) => {
try {
if (!req.files || !req.files.file) {
console.log("No file received on the server");
return res.status(400).json({ error: 'File is required.' });
}
const file = req.files.file;
if (file.size > MAX_FILE_SIZE_BYTES) {
console.log("File size exceeds the limit");
return res.status(400).json({ error: 'File size exceeds the limit.' });
}
console.log("File received on the server", file);
Convert the file data to a buffer
const fileData = Buffer.from(file.data);
Generate a random 256-bit key
const aesKey = crypto.randomBytes(32); 256 bits / 8 = 32 bytes
Create an AES-256-CFB cipher using the key
const cipher = crypto.createCipheriv('aes-256-cfb', aesKey, Buffer.alloc(16, 0)); Initialization
Vector (IV)
is set to zero for simplicity
Encrypt the file
const encryptedFile = Buffer.concat([cipher.update(fileData),
cipher.final()]);console.log("Encrypted file:", encryptedFile);
Convert key and encrypted file to Base64 for JSON response
const aesKeyData = aesKey.toString('base64');
const encryptedFileData = encryptedFile.toString('base64');
res.json({ file: encryptedFileData, key: aesKeyData });
} catch (err) {
console.error("Encryption failed:", err);
res.status(500).json({ error: "Internal Server Error" });
}
});
RSA - ENCRYPT AREA
const handleServerHybridEncryption = async (req, res) => {
try {
const aesKey = req.body.key;
```

```
if (!aesKey) {
console.log("AES key not provided");
return res.status(400).json({ error: 'AES key is required.' });
}
RSA encryption of AES key
const rsaKey = forge.pki.publicKeyFromPem(RSA_PUBLIC_KEY);
const encryptedAesKey = rsaKey.encrypt(aesKey, 'RSA-OAEP');
Convert the encrypted key to Base64
const base64EncryptedAesKey = forge.util.encode64(encryptedAesKey);
res.json({ encryptedAesKey: base64EncryptedAesKey });
} catch (error) {
console.error('RSA failed (from server):', error);
res.status(500).json({ error: 'Internal Server Error' });
}
};
app.post('/rsaserverencrypt', handleServerHybridEncryption);
const cloudroute = require('./routes/mroute');
const cloudverify = require('./routes/mverify');
const fupload=require('./routes/uploadroute');
const hashfile=require('./routes/hashstore');
app.use('/Endosignup', cloudroute);
app.use('/Endoverify', cloudverify);
app.use('/Endovault',fupload);
app.use('/Endohash',hashfile);
app.listen(PORT, () => console.log(`Server is running on port ${PORT}`));
```

User Signup - Schema:

```
const mongoose = require("mongoose");
const Schema = mongoose.Schema;
const bcrypt = require('bcrypt');
const CCsignup = new Schema(
{
name: { type: String, required: true },
email: { type: String, required: true },
regnum: { type: Number, required: true },
pass: { type: String, required: true },
active: { type: Boolean, default: false },
verificationToken: { type: String },
},
{ collection: 'signup' }
);
CCsignup.pre('save', function (next) {
if (this.pass && this.isModified('pass')) {
bcrypt.hash(this.pass, 10, (err, hashed) => {
if (err) return next(err);
this.pass = hashed;
next();
});
} else {
next();
}
});
CCsignup.methods.checkpass = function (pass, cb) {
```

```
bcrypt.compare(pass, this.pass, (err, result) => {
return cb(err, result);
});
};
const Userdetails = mongoose.model("Signup", CCsignup);
module.exports = Userdetails;
```

Email sending :

```
const express = require('express');
const router = express.Router();
const CCsignup = require('../models/signup');
const nodemailer = require('nodemailer');
const crypto = require('crypto');
const transporter = nodemailer.createTransport({
service: 'gmail',
auth: {
user: 'buvaneshkumaars_ai@mepcoeng.ac.in',
pass: 'Buvi@123',
},
});
router.post('/signup', async (req, res) => {try {
const { name, email, regnum, pass } = req.body;
const existingUser = await CCsignup.findOne({ email });
if (existingUser) {
return res.status(400).json({ error: 'Email already exists' });
}
const token = crypto.randomBytes(20).toString('hex');
const usersignup = new CCsignup({
name,
email,
regnum,
pass,
active: false,
verificationToken: token, });
await usersignup.save();
const mailOptions = {
from: 'buvaneshkumaars_ai@mepcoeng.ac.in',
to: email,
subject: 'Email Verification',
text: `Hello ${name}, please click on the link to verify your email:
http:localhost:8080/verify/${token}`,
};
await transporter.sendMail(mailOptions);
res.status(201).json({ message: 'Data user signup details saved successfully' });
} catch (error) {
console.error(error);
res.status(500).json({ error: 'Internal server error' });
}
});
module.exports = router;
```

Mail verification:

```
const express = require('express');
const router = express.Router();
const CCsignup = require('../models/signup');
router.get('/verify/:token', async (req, res) => {
try {
const token = req.params.token;
const user = await CCsignup.findOne({ verificationToken: token });
if (!user) {
return res.status(404).json({ error: 'Invalid or expired verification token' });
}
console.log(user);console.log("token verified!");
user.active = true;
console.log("activated!");
user.verificationToken = '';
await user.save();
console.log(res);
} catch (error) {
console.error(error);
res.status(500).json({ error: 'Internal server error' });
}
});
module.exports = router;
```

File upload Shcema:

```
const mongoose = require('mongoose');
const fileSchema = new mongoose.Schema({
URD: {
type: String
},
code: {
type: String,
unique: true
},
filename: {
type: String,
},
domain: {
type: String,
required: true
},
uploadDate: {
type: Date,
default: Date.now
},
contentType: {
type: String
},
fileData: {
type: Buffer
},
fileSize: {
type: Number
},
fileExtension: {
type: String
},
descrip: {
type: String,required: true
},
});
function generateRandomCode(length) {
const prefix = 'STD';
const numbers = '0123456789';
let randomNumbers = '';
Generate four random numbers
for (let i = 0; i < 4; i++) {
randomNumbers += numbers.charAt(Math.floor(Math.random() * numbers.length));
}
const characters = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz';
let randomChars = '';
Generate random characters with the given length
for (let i = 0; i < length - 4; i++) {
randomChars += characters.charAt(Math.floor(Math.random() * characters.length));
}
Combine prefix, random numbers, and random characters
return prefix + randomNumbers + randomChars;
}
fileSchema.pre('save', function(next) {
this.code = generateRandomCode(10);
next();
});
const File = mongoose.model('File', fileSchema,'Documents');
module.exports = File;
```

Upload File route:

```
const express=require('express');
const multer=require('multer');
const router=express.Router();
const File=require('../models/fupload');
const Grid=require('gridfs-stream');
const {GridFsStorage}=require('multer-gridfs-storage');
const storage=multer.memoryStorage();
const upload=multer({storage:storage,limits:{fileSize:200 * 1024 * 1024}});
router.post('/submit',upload.single('upload'),async(req,res)=>{
const file=req.file;
const {domain,descrip,urd}=req.body;
console.log(req.body);if(!file)
{
return res.status(400).send('file not upload');
}
const newFile=new File({
filename:file.originalname,
contentType:file.mimetype,
fileData:file.buffer,
domain:domain,
descrip:descrip,
urd:urd,
uploadDate:new Date(),
});
try
{
await newFile.save();
console.log(`File ${newFile.filename} has been saved to MongoDB Atlas.`);
res.send('File uploaded successfully');
}catch(err)
{
console.log(err.message);
res.status(500).send('Internal server error');
}
});
module.exports=router;
```

Hash route:

```
const express = require('express');
const multer = require('multer');
const crypto = require('crypto');
const HFile = require('../models/fhash');
const router = express.Router();
const storage = multer.memoryStorage();
const upload = multer({ storage: storage });
router.use(express.json());
router.post('/generateHash', upload.single('file'), (req, res) => {
console.log('Route hit');
console.log("req",req);
console.log('Request body:', req.body);
console.log('Request file:', req.file);
console.log('File content:', req.file.buffer.toString('utf-8'));
if (!req.file) {
return res.status(400).json({ error: 'No file provided' });
}
console.log("File received in the server");
const filebuffer = req.file.buffer;
Generate hash
const hash = crypto.createHash('sha256').update(filebuffer).digest('hex');
const size = req.file.size;
Respond with success message only
res.json({ message: 'Hash generated successfully', hash, size });
});
router.post('/generateHash', upload.single('file'), async (req, res) => {
try {
if (!req.file) {
return res.status(400).json({ error: 'No file provided' });
}
console.log("File received in the server");
const fileBuffer = req.file.buffer;
const hash = crypto.createHash('sha256').update(fileBuffer).digest('hex');
const { originalname: filename, size } = req.file;
const userId = req.body.userId; Make sure the userId is sent from the front end
Save file information to the database
const newFile = new HFile({
filename,hash,
size,
userId,
});
await newFile.save();
Respond with success message and file information
res.json({ message: 'Hash generated and file saved successfully', hash, size, filename,
userId });
} catch (error) {
console.error('Error generating hash:', error);
res.status(500).json({ error: 'Internal server error' });
}
});
router.post('/storeFile', upload.single('file'), async (req, res) => {
if (!req.file) {
return res.status(400).json({ error: 'No file provided' });
}
const filename = req.body.filename;
const file = req.file.buffer;
const userId = req.body.userId;
Generate hash
const hash = crypto.createHash('sha256').update(file).digest('hex');
const size = req.file.size;
Store the data in MongoDB collection 'hfiles'
const newHFile = new HFile({
filename,
hash,
size,
userId,
});
newHFile.save()
.then(() => res.json({ message: 'File stored successfully' }))
.catch(error => res.status(500).json({ error: 'Internal Server Error' }));
});
module.exports = router;
```

Frontend Code

Sign up:

```
import React, { useState } from 'react';
import styles from './styles.module.css';
import axios from 'axios';
import { Link } from 'react-router-dom';
import { ToastContainer, toast } from 'react-toastify';
import 'react-toastify/dist/ReactToastify.css';

const Signup = () => {
  const [name, setName] = useState('');
  const [email, setEmail] = useState('');
  const [regnum, setRegnum] = useState('');
  const [pass, setPass] = useState('');

  const handleSignup = async (e) => {
    e.preventDefault();

    try {
      const response = await axios.post('http://localhost:8080/Endosignup/signup', {
```

```
      name,
      email,
      regnum: parseInt(regnum),
      pass,

    },
    {
      headers: {
        'Content-Type': 'application/json',
      },
    });

    console.log(response.data);

    toast.success('Successfully registered! Please check your email for verification.', {
      position: 'top-right',
      autoClose: 5000,
      hideProgressBar: false,
      closeOnClick: true,
      pauseOnHover: true,
      draggable: true,
    });

    setName('');
    setEmail('');
    setRegnum('');
    setPass('');
  } catch (error) {
    if (error.response && error.response.status === 409) {
      const errorMessage = error.response.data.error;

      // Display different alert messages based on the server response
      if (errorMessage.includes('Email, Register Number, and Password')) {
        alert('Email, Register Number, and Password already exist!');
      } else if (errorMessage.includes('Email and Register Number')) {
        alert('Email and Register Number already exist!');
      } else if (errorMessage.includes('Email and Password')) {
        alert('Email and Password already exist!');
      } else if (errorMessage.includes('Register Number and Password')) {
        alert('Register Number and Password already exist!');
      } else if (errorMessage.includes('Email')) {
        alert('Email already exists!');
      } else if (errorMessage.includes('Register Number')) {
        alert('Register Number already exists!');
      } else if (errorMessage.includes('Password')) {
        alert('Password already exists!');
      }
    } else {
      console.error(error);
    }
  }
};

return (
  <div className={styles.signup_container}>
```

```
<div className={styles.signup_form_container}>
 <ToastContainer />

 <div className={styles.left}>
  <h1>Welcome!</h1>
  <Link to="/login">
   <button type="button" className={styles.white_btn}>
    Sign In
   </button>
  </Link>
 </div>

 <div className={styles.right}>
  <form className={styles.form_container} onSubmit={handleSignup}>
   <h1>Create Your Account</h1>

   <input
    type="text"
    placeholder="Username"
    name="name"
    value={name}
    onChange={(e) => setName(e.target.value)}
    required
    className={styles.input}
   />

   <input
    type="email"
    placeholder="Email"
    name="email"
    value={email}
    onChange={(e) => setEmail(e.target.value)}
    required
    className={styles.input}
   />

   <input
    type="text"
    placeholder="College Register Number"
    name="reg_num"
    value={regnum}
    onChange={(e) => setRegnum(e.target.value)}
    required
    className={styles.input}
   />

   <input
    type="password"
    placeholder="Password"
    name="pass"
    value={pass}
    onChange={(e) => setPass(e.target.value)}
    required
    className={styles.input}
   />
```

```
          <button type="submit" className={styles.green_btn}>
            Sign Up
          </button>
        </form>
      </div>
    </div>
  </div>
 );
};

export default Signup;
```

Login:

```
import React from 'react';
import { useState } from 'react';

import { Link } from 'react-router-dom';
import axios from 'axios';
import styles from './styles.module.css';
import { useNavigate } from 'react-router-dom';
import { GoogleOAuthProvider } from '@react-oauth/google';
import { GoogleLogin } from '@react-oauth/google';
import jwt_decode from 'jwt-decode';

const Login = () => {
  const [email, setEmail] = useState("");
  const [pass, setPass] = useState("");
  const navigate = useNavigate();

  const handleSubmit = async (e) => {
    e.preventDefault();
    try {
      const response = await axios.post("http://localhost:8080/Endoauth/login", {
        email: email,
        pass: pass,
      });

      console.log(response);

      if (response.status === 200) {
        alert("Login successful");
        navigate("/show");
        setEmail("");
        setPass("");
      } else if (response.status === 401) {
        alert("Invalid password");
      } else if (response.status === 404) {
        alert("User not found");
      } else {
        alert("Login failed");
      }
    } catch (err) {
      console.log(err);
```

```
        alert("Check the login credentials!!!");
      }
    };

    return (

      <div className={styles.login_container}>
        <div className={styles.login_form_container}>
          <div className={styles.left}>
            <form className={styles.form_container} onSubmit={handleSubmit}>
              <h1>Login Your Account</h1>



              <input
                type="text"
                placeholder='Email ID'
                name='email'
                onChange={(e) => {
              // Clear all fields when any field changes (this may not be the desired behavior)
                  setEmail(e.target.value); // Set only the 'name' field
                }}
                value={email}
                required
                className={styles.input} />

              <input
                type="text"
                placeholder='Password'
                name='pass'
                onChange={(e) => {
              // Clear all fields when any field changes (this may not be the desired behavior)
                  setPass(e.target.value); // Set only the 'name' field
                }}
                value={pass}
                required
                className={styles.input} />

              {/* google_login */}

              <GoogleOAuthProvider clientId="913271782373-
ph3nudp7k9djgg49vq8v9deafpokg72d.apps.googleusercontent.com">
                <GoogleLogin
                  onSuccess={credentialResponse => {
                    const details = jwt_decode(credentialResponse.credential);
                    console.log(details);
                    console.log(credentialResponse);
                    navigate('/show');
                  }}
                  onError={() => {
                    console.log('Login Failed');
                  }}
                />
              </GoogleOAuthProvider>
```

```
                <button type="submit" className={styles.green_btn}>
                    <span>Sign In </span>
                </button>

            </form>
        </div>
        <div className={styles.right}>
            <h1>Are you new ?</h1>
            <Link to="/signup">
                <button type='button' className={styles.white_btn}>
                    Sign Up
                </button>
            </Link>


        </div>
    </div>
  </div> )};
export default Login;
```

File Upload:

```
import React, { useState, useRef} from 'react';
import './uploadcss.css';
import { ImFolderUpload } from 'react-icons/im';
import { PiFilesFill } from 'react-icons/pi';
import { ImDownload3 } from 'react-icons/im';
import { TbLogout2 } from 'react-icons/tb';
import { CgProfile } from 'react-icons/cg';
import { FaBars } from 'react-icons/fa';
import { NavLink } from 'react-router-dom';
import './uploadcss.css';
import { FaHandshake } from 'react-icons/fa';
import 'react-toastify/dist/ReactToastify.css';
function Uploadfiles() {
consat fileInput = useRef();
const [fname, setFname] = useState('');const [domain, setDomain] = useState('');
const [descrip, setDesc] = useState('');
const [urd, setUrd] = useState('');
const [sec, setSec] = useState('')
const [isOpen, setIsOpen] = useState(false);
const [showFileUpload, setShowFileUpload] = useState(false);
const [selectedFile, setSelectedFile] = useState(null);
const toggle = () => setIsOpen(!isOpen);
const menuItems = [
{
path: '/myfiles',
name: 'My Files',
icon: <PiFilesFill />,
},
{
path: '/downloadfiles',
name: 'Download Files',
icon: <ImDownload3 />,
},
{
path: '/profile',
name: 'My Profile',
icon: <CgProfile />,
},
{
path: '/',
name: 'Logout',
icon: <TbLogout2 />,
},
];
const handleFileChange = (e) => {
const file = e.target.files[0];
setSelectedFile(file);
};
const onUpload = async (e) => {
e.preventDefault();prevent Default submission
if (selectedFile) {
const formData = new FormData();
formData.append('file', selectedFile);
console.log("File selected!");
}
const handleItemClick = (index) => {
if (index === 0) {
setShowFileUpload(true); Display FileUpload component when Upload Files is clicked
} else {setShowFileUpload(false);
}
}
const navigate=useNavigate();
const onSubmit = async (e) => {
e.preventDefault();
console.log("Form Data enters!");
const formData = new FormData();
formData.append('upload', selectedFile);
formData.append('Urd', urd);
formData.append('name', fname);
formData.append('domain', domain);
formData.append('descrip', descrip);
console.log('URD', urd);
console.log("Name", fname);
console.log("Domain:", domain);
console.log("Description:", descrip);
console.log("File Details:",selectedFile);
try {
const response = await fetch('http:localhost:8080/Endovault/submit', {
method: 'POST',
body: formData,
});
return (
<> <div className='container'>
<div style={{ width: isOpen ? '300px' : '50px' }} className='sidebar'>
<div className='top_section'>
<h1 style={{ display: isOpen ? 'block' : 'none' }} className='logo'>
Encryovault
```

```
</h1>
<div style={{ marginLeft: isOpen ? '50px' : '0px' }} className='bars'>
<FaBars onClick={toggle} />
</div>
</div>
{menuItems.map((item, index) => (
<div key={index} className='link'>
<NavLink
to={item.path}
onClick={() => handleItemClick(index)} Handle menu item click
className='link'
activeClassName='active'
>
<div className='icon'>{item.icon}</div>
<div style={{ display: isOpen ? 'block' : 'none' }} className='link_text'>
{item.name}
</div>
</NavLink>
{showFileUpload && index === 0 && ( Conditional rendering for FileUpload
<div className='fileUploadContainer'>{item.component}</div>)}
</div> ))}
</div>
<div className="grand-page">
<div className="form-container">
<h1 className="upload-heading">**** Upload your Files ****</h1>
<form encType="multipart/form-data">
<label className="input-label">Enter the User Recognition ID:</label>
<input
type="text"
className="input-field"
placeholder="Enter your User Recognition ID:"
name="urd"
value={urd}
onChange={(e) => setUrd(e.target.value)}
/>
<label className="input-label">Enter the Filename:</label>
<input
type="text"
aclassName="input-field"
placeholder="Enter the filename"
name="fname"
value={fname}
onChange={(e) => setFname(e.target.value)} />
<label className="input-label">Enter the Domain Name:</label>
<input
type="text"
className="input-field"
placeholder="Enter your project domain..."
name="domain"
value={domain}
onChange={(e) => setDomain(e.target.value)}
/>
<label className="input-label">File Description</label>
<textarea
type="text"
```

```
className="input-field description"
name="descrip"
onChange={(e) => {
setDesc(e.target.value);
}}
value={descrip}
placeholder="Description of your project..."
rows={4}
cols={40}
/>
<label className="input-label">Choose your File:</label>
<input type="file" name="file" onChange={handleFileChange} ref={fileInput} />
<button type="submit" onClick={onSubmit} className="upload-button">
Upload here!!!</button>
</form>
</div>
</div>
</div> </>
)}
export default Uploadfiles;
```

File Compression:

```
import React, { useState } from 'react';
import './com.css';
import { ImFolderUpload } from 'react-icons/im';
import { ImDownload3 } from 'react-icons/im';
import { TbLogout2 } from 'react-icons/tb';
import { CgProfile } from 'react-icons/cg';
import { FaBars } from 'react-icons/fa';
import { NavLink } from 'react-router-dom';
function Compress() {
const [isOpen, setIsOpen] = useState(false);
const [showFileUpload, setShowFileUpload] = useState(false);
const [file, setFile] = useState(null);
const [download, setDownload] = useState(null);
const handleItemClick = (index) => {
if (index === 0) {
setShowFileUpload(true);
} else {
setShowFileUpload(false);
}
};
const toggle = () => setIsOpen(!isOpen);
const menuItems = [
{
path: '/fileupload',
name: 'Upload Files',
icon: <ImFolderUpload />,
},
{
path: '/downloadfiles',
name: 'Download Files',
icon: <ImDownload3 />,
},
{
path: '/profile',
name: 'My Profile',
icon: <CgProfile />,
},
{path: '/',
name: 'Logout',
icon: <TbLogout2 />,
},
];
const handleFileChange = (event) => {
setFile(event.target.files[0]);
};
const handleCompress = async () => {
try {
if (!file) {
console.error('No file selected.');
return;
}
console.log('File gets selected');
const formData = new FormData();
formData.append('file', file,file.name);
console.log("File",file);
const response = await fetch('http:localhost:8080/compress', {
method: 'POST',
body: formData,
});
if (!response.ok) {
console.error('Error compressing file:', response.statusText);
return;
}
const compressedData = await response.blob();
const downloadUrl = URL.createObjectURL(compressedData);
setDownload(downloadUrl);
} catch (error) {
console.error('Error compressing file:', error);
}
};
return (
<>
<div className="container">
<div style={{ width: isOpen ? '300px' : '50px' }} className="sidebar">
<div className="top_section">
<h1 style={{ display: isOpen ? 'block' : 'none' }} className="logo">
Encryovault
</h1>
<div style={{ marginLeft: isOpen ? '50px' : '0px' }} className="bars">
<FaBars onClick={toggle} />
</div>
</div>
{menuItems.map((item, index) => (<div key={index} className="link">
<NavLink
to={item.path}
onClick={() => handleItemClick(index)}
className="link"
activeClassName="active"
>
<div className="icon">{item.icon}</div>
<div style={{ display: isOpen ? 'block' : 'none' }} className="link_text">
{item.name}
</div>
</NavLink>
{showFileUpload && index === 0 && (
<div className="fileUploadContainer">{item.component}</div>
)}
</div>
))}
</div>
<div className="comp">
<form className="fbody">
<h1>File Compression</h1>
<input type="file" accept=".txt,.docx,.pdf" name="file" onChange={handleFileChange} />
<button type="button" onClick={handleCompress}>
Compress
</button>
</form>
{download && (
<a href={download} download={file.name}>
Download Compressed File
</a>
)}
</div>
</div>
</>
);
}
export default Compress;
```

File Encryption:

```
import React, { useRef, useState } from "react";
import "./encry.css";
import axios from "axios";
import { toast } from "react-toastify";
import "react-toastify/dist/ReactToastify.css";
const Encryfile = () => {
const inputRef = useRef();
const [selectedFile, setSelectedFile] = useState(null);const [aesEncryptionFile,
setAesEncryptionFile] = useState(null);
const [rsaEncryptionFile, setRsaEncryptionFile] = useState(null);
const [aesKeyForEncryption, setAesKeyForEncryption] = useState(null);
const handleAESClick = async () => {
if (!selectedFile) {
return;
}
const formData = new FormData();
formData.append('file', selectedFile);
try {
Send the file to the server for encryption
const response = await axios.post("http:localhost:8080/aesencrypt", formData, {
```

```
headers: {
'Content-Type': 'multipart/form-data',
},
});
if (response.data.error) {
console.error("Encryption failed:", response.data.error);
toast.error("Encryption failed");
return;
}
Check if the response data includes the required properties
if (!response.data.file || !response.data.key) {
console.error("Invalid response data:", response.data);
toast.error("Invalid response data");
return;
}
console.log("AES Key:", response.data.key);
Update the state with the encrypted file and key
setAesEncryptionFile({
file: response.data.file,
key: response.data.key,
});
console.log("Encryption successful");
toast.success("Encryption successful");
After AES encryption, trigger RSA encryption
setAesKeyForEncryption(response.data.key);
Set the AES key in the component state
} catch (err) {
console.error("Encryption failed:", err);
toast.error("Encryption failed");
}
};const handleServerHybridEncryptionClick = async () => {
if (!aesKeyForEncryption) {
return;
}
console.log("AES key received!");
try {
Convert the AES key to Base64
const encoder = new TextEncoder();
const base64AesKey = btoa(encoder.encode(aesKeyForEncryption));
const response = await axios.post("http:localhost:8080/rsaserverencrypt", {
key: base64AesKey,
}, {
responseType: 'arraybuffer',
transformRequest: [(data) => data],
});
const encryptedAesKey = new Uint8Array(response.data);
setRsaEncryptionFile({ encryptedAesKey });
toast.success("RSA Encryption successful");
} catch (err) {
console.error("RSA encryption failed (from front):", err);
toast.error("RSA Encryption Failed");
}
};
const handleDownloadClick = () => {
if (rsaEncryptionFile) {
```

```
const blob = new Blob([rsaEncryptionFile.file], {
type: "application/octet-stream",
});
const link = document.createElement("a");
link.href = URL.createObjectURL(blob);
link.download = "encrypted_file";
link.click();
}
};
const handleFileChange = (event) => {
if (event.target.files && event.target.files.length > 0) {
setSelectedFile(event.target.files[0]);
}
};
const onChooseFile = () => {
inputRef.current.click();
};
const clearFileInput = () => {
inputRef.current.value = "";
setSelectedFile(null);}; return (
<div>
<input
ref={inputRef}
type="file"
onChange={handleFileChange}
style={{ display: "none" }}
/>
{/* Button to trigger the file input dialog */}
{!selectedFile && (
<button className="file-btn" onClick={onChooseFile}>
<span className="material-symbols-outlined"></span> Upload File
</button>
)}
{selectedFile && (
<>
<div className="file-card">
<span className="material-symbols-outlined icon"></span>
<div className="file-info">
<div style={{ flex: 1 }}>
<h6>{selectedFile?.name}</h6>
</div>
</div>
</div>
<button className="encrypt-btn" onClick={handleAESClick}>
Encrypt with AES
</button>
{/* Button to trigger Hybrid encryption */}
{aesEncryptionFile && (
<button
className="encrypt-btn"
onClick={handleServerHybridEncryptionClick}
>
Hybrid Encryption
</button>
)}
{/* Button to trigger download */}
{rsaEncryptionFile && (
<button className="download-btn" onClick={handleDownloadClick}>
Download Encrypted File
</button>
)}
</>
)}
</div>
);
};export default Encryfile;
```

File Decryption:

```
import React, { useState, useEffect } from 'react';
import Sidebar from "../sidebar2/sidebar2";
import 'react-toastify/dist/ReactToastify.css';

function Downloadfiles() {
  const [files, setFiles] = useState([]);
  const [filter, setFilter] = useState('');
  const [domainFilter, setDomainFilter] = useState('');
  const [keywordsFilter, setKeywordsFilter] = useState('');
  const [accessCodeFilter, setAccessCodeFilter] = useState(''); // New state for access code
filter
  const [rankKeyword, setRankKeyword] = useState('');
  const [rankedFiles, setRankedFiles] = useState([]);
  const [documentScores, setDocumentScores] = useState([]);

  useEffect(() => {
    fetch('http://localhost:8080/Simupload/upload1')
      .then(response => response.json())
      .then(data => setFiles(data))
      .catch(error => console.error(error));

    if (rankKeyword.trim() !== '') {
      fetch(`http://localhost:8080/Simupload/rankFiles?keyword=${rankKeyword}`)
        .then(response => response.json())
        .then(scores => setDocumentScores(scores))
        .catch(error => console.error(error));
    }
  }, [rankKeyword]);

  const rankFiles = async () => {
    try {
      const response = await
fetch(`http://localhost:8080/Simupload/rankFiles?keyword=${rankKeyword}`);
      const rankedFiles = await response.json();
      setRankedFiles(rankedFiles);
    } catch (error) {
      console.error(error);
    }
  };

  const resetRank = () => {
    setRankKeyword('');
    setRankedFiles([]);
    setDocumentScores([]);
  };

  const filterFiles = async () => {
    try {
      const response = await
fetch(`http://localhost:8080/Simupload/filterByKeywords?keyword=${keywordsFilter}`);
      const filteredFiles = await response.json();
      setFiles(filteredFiles);
      setDocumentScores([]);
    } catch (error) {
      console.error(error);
    }
  };

  const filterByAccessCode = async () => {
    try {
      const response = await
fetch(`http://localhost:8080/Simupload/filterByAccessCode?accessCode=${accessCodeFilter}
`);
      const filteredFiles = await response.json();
      setFiles(filteredFiles);
      setDocumentScores([]);
    } catch (error) {
      console.error(error);
    }
  };

  const filterByTitle = async () => {
    try {
      const response = await
fetch(`http://localhost:8080/Simupload/filterByTitle?title=${filter}`);
      const filteredFiles = await response.json();
      setFiles(filteredFiles);
      setDocumentScores([]);
    } catch (error) {
      console.error(error);
    }
  };

  const resetFilters = async () => {
    setDomainFilter('');
    setFilter('');
    setKeywordsFilter('');
    setAccessCodeFilter(''); // Reset access code filter
    try {
      const response = await fetch('http://localhost:8080/Simupload/upload1');
      const allFiles = await response.json();
      setFiles(allFiles);
      setDocumentScores([]);
    } catch (error) {
      console.error(error);
    }
  };

  const downloadFile = async (file) => {
    try {
      const response = await fetch(`http://localhost:8080/Simupload/downloadFile/${file._id}`,
{
        method: 'GET',
      });

      if (response.ok) {
        const blob = await response.blob();
        const url = window.URL.createObjectURL(blob);
        const a = document.createElement('a');
        a.href = url;
        a.download = file.filename;
        document.body.appendChild(a);
        a.click();
        document.body.removeChild(a);
      } else {
        console.error('Error downloading file');
      }
    } catch (error) {
      console.error(error);
    }
  };

  return (
    <>
      <div style={{ display: 'flex' }}>
        <Sidebar />
        <div>
          <div className="filters-container">
            <br></br>
            <br></br>

            <label>Filter by Domain:</label>
            <input type="text" value={domainFilter} onChange={(e) =>
setDomainFilter(e.target.value)} />
            <br></br>
            <br></br>

            <label>Filter by Keywords:</label>
            <input type="text" value={keywordsFilter} onChange={(e) =>
setKeywordsFilter(e.target.value)} />
            <button onClick={filterFiles}>Apply Keyword Filter</button>
            <button onClick={resetFilters}>Reset Filters</button>
            <br></br>
            <br></br>

            <label>Filter by Access Code:</label>
            <input type="text" value={accessCodeFilter} onChange={(e) =>
setAccessCodeFilter(e.target.value)} />
            <button onClick={filterByAccessCode}>Search by Access Code</button>
            <br></br>
            <br></br>
            <label>Filter by Title:</label>
            <input type="text" value={filter} onChange={(e) => setFilter(e.target.value)} />
            <button onClick={filterByTitle}>Search by Title</button>
            <br></br>
            <br></br>
            <button onClick={resetFilters}>Reset Filters</button>
            {/* <button onClick={() => { setAccessCodeFilter('');
filterByAccessCode(); }}>Reset Access Code Filter</button>
            <button onClick={() => { setFilter(''); filterByTitle(); }}>Reset Title Filter</button>
*/}
            <br>
            </br>
            <br></br>
            <label>Rank by Keyword:</label>
            <input type="text" value={rankKeyword} onChange={(e) =>
setRankKeyword(e.target.value)} />
            <button onClick={rankFiles}>Rank</button>
            <button onClick={resetRank}>Reset Rank</button>
            <br></br>
            <br></br>

            <div className='container'>
              <table border='1'>
                <thead>
                  <tr>
                    <th>Domain</th>
                    <th>Title</th>
                    <th>URD</th>
                    <th>Description</th>
                    <th>Access Code</th>
                    <th>Uploaded Date</th>
                    <th>File Size (MB)</th>
                    <th>File Extension</th>
                    <th>Actions</th> {/* New column for actions */}
                  </tr>
                </thead>

                <tbody>
                  {(rankedFiles.length > 0 ? rankedFiles : files).map((file, index) => (
                    <tr key={index}>
                      <td>{file.domain}</td>
                      <td>{file.filename}</td>
                      <td>{file.urd}</td>
                      <td>{file.descrip}</td>
                      <td>{file.code}</td>
                      <td>{new Date(file.uploadDate).toLocaleDateString()}</td>
                      <td>{file.fileSize}</td>
                      <td>{file.fileExtension}</td>
                      <td>
                        <button onClick={() => downloadFile(file)}>Decrypt and Download</button>
                      </td>
                    </tr>
                  ))}
                </tbody>
          </table>
         </div>
        </div>
       </div>
      </div>
    </>
  );
}

export default Downloadfiles;
```

Remove Files:

```
import React, { useState, useEffect } from 'react';
import Sidebar from "../sidebar2/sidebar2";
import 'react-toastify/dist/ReactToastify.css';
import { ToastContainer, toast } from 'react-toastify';

function Removefiles() {
  const [files, setFiles] = useState([]);
  const [filter, setFilter] = useState('');
  const [domainFilter, setDomainFilter] = useState('');
  const [keywordsFilter, setKeywordsFilter] = useState('');
  const [rankKeyword, setRankKeyword] = useState('');
  const [rankedFiles, setRankedFiles] = useState([]);
  const [documentScores, setDocumentScores] = useState([]);
  useEffect(() => {
    // Fetch files from the backend API
    fetchFiles();
  }, []);

  const fetchFiles = async () => {
    try {
      const response = await fetch('http://localhost:8080/Simupload/upload1');
      const allFiles = await response.json();
      setFiles(allFiles);
      setDocumentScores([]);
    } catch (error) {
      console.error(error);
    }
  };

  const deleteFile = async (fileId) => {
    // Display a confirmation dialog
    const userConfirmed = window.confirm('Do you really want to delete the file?');

    if (!userConfirmed) {
      // User canceled the deletion
      return;
    }

    try {
      await fetch(`http://localhost:8080/Simupload/deleteFile/${fileId}`, {
        method: 'DELETE',
      });
    toast.success('File deleted successfully', { position: toast.POSITION.TOP_RIGHT });
    // Update the file list after deletion
    fetchFiles();
  } catch (error) {
    console.error(error);
    toast.error('Error deleting file', { position: toast.POSITION.TOP_RIGHT });
  }
};

return(
<>
{/* <ToastContainer /> */}
  <div style={{ display: 'flex' }}>
    <Sidebar />
    <div>

      <div className="filters-container">
        <br></br>
        <br>
        </br>
        <div className='container'>
          <table border='1'>
            <thead>
              <tr>
                <th>Domain</th>
                <th>Title</th>
                <th>URD</th>
                <th>Description</th>
                <th>Access Code</th>
                <th>Uploaded Date</th>
                <th>File Size (MB)</th>
                <th>File Extension</th>
                <th>Action</th>
              </tr>
            </thead>
            <tbody>
              {(rankedFiles.length > 0 ? rankedFiles : files).map((file, index) => (
                <tr key={index}>
                  <td>{file.domain}</td>
                  <td>{file.filename}</td>
                  <td>{file.urd}</td>
                  <td>{file.descrip}</td>
                  <td>{file.code}</td>
                  <td>{new Date(file.uploadDate).toLocaleDateString()}</td>
                  <td>{file.fileSize}</td>
                  <td>{file.fileExtension}</td>
                  <td>
                    <button onClick={() => deleteFile(file._id)}>Delete</button>
                  </td>
                </tr>
              ))}
            </tbody>
          </table>
        </div>
        </div>
      </div>
    </div>
  </>
 );
}
export default Removefiles;
```

Keyword Extract:

```
import React, { useState ,useEffect} from 'react';
import axios from 'axios';
import Sidebar from '../sidebar2/sidebar2';

const FileUploadForm = () => {
 const [file, setFile] = useState();
 const [keywords, setKeywords] = useState([]);
 const [blinkArrow, setBlinkArrow] = useState(true);


 useEffect(() => {
  const intervalId = setInterval(() => {
   setBlinkArrow((prev) => !prev);
  }, 500); // Blink every 500ms

  return () => clearInterval(intervalId);
 }, []);

 const onChangeHandler = (event) => {
  setFile(event.target.files[0]);
 };

 const onClickHandler = () => {
  const formData = new FormData();
  formData.append('file', file);

  axios.post('http://localhost:8080/processfile', formData)
   .then(response => {
    setKeywords(response.data);
   })
   .catch(error => {
    console.error(error);
   });
 };

 return (
  <>
   <div style={{ display: 'flex' }}>
    <Sidebar />
    <div className="main-container">
     <h2>Frequency Extractor</h2>
     <div className="form-container">
      <form>
       <label htmlFor="fileInput">Choose a file:</label>
```

```
          <input type="file" id="fileInput" name="file" onChange={onChangeHandler} />
          <button type="button" onClick={onClickHandler}>Extract Keywords</button>
          <h1>Top 10 Occurring Keywords</h1>
          <br></br>

          {keywords && Object.keys(keywords).length > 0 && (
            <div className="result-container">
              <table>
                <thead>
                  <tr>
                    <th>Keywords</th>
                    <th>Frequency</th>
                  </tr>
                </thead>
                <tbody >
                  {Object.entries(keywords).map(([word, count]) => (
                    <tr key={word}>
                      <td>{word}</td>
                      <td>{count}</td>
                    </tr>
                  ))}
                </tbody>
              </table>

              <div style={{ marginTop: '20px' }}>
                <div style={{ display: 'flex', alignItems: 'center', justifyContent: 'flex-end' }}>
                  <div style={{ display: 'flex', alignItems: 'center', paddingRight: '10px' }}>
                    <div style={{ fontSize: '150px', color: 'green', marginRight: '5px', visibility:
blinkArrow ? 'visible' : 'hidden' }}>
                      &#x2191;
                    </div>
                    <div style={{ fontSize: '16px' }}>Copy and Use these keywords for your file
to upload</div>
                  </div>
                </div>
              </div>
            </div>
          )}

        </form>
      </div>
    </div>
  </div>
  </>
 );
};
export default FileUploadForm;
```

BM25 Ranking Strategy:

```
from google.colab import drive
from google.colab.output import eval_js
from IPython.display import display, Javascript
from google.colab import drive

drive.mount('/content/drive')
!pip install python-docx==0.8.10
!pip install --upgrade python-docx
import os
import math
from collections import Counter
from docx import Document

class MyCounter(Counter):
def __init__(self, *args, **kwargs):
super(MyCounter, self).__init__(*args, **kwargs)
class BM25:
def __init__(self, documents):
self.documents = documents
self.document_count = len(documents)
self.avg_document_length = sum(len(doc) for doc in documents) / self.document_count
self.term_counts = self.calculate_term_counts()
self.k1 = 1.2
self.b = 0.75

def calculate_term_counts(self):
        term_counts = MyCounter()
        for document in self.documents:
        term_counts.update(document)
        return term_counts
        def calculate_idf(self, term):
        document_with_term_count = self.term_counts[term]
        return math.log((self.document_count - document_with_term_count + 0.5) /
(document_with_term_count + 0.5))
        def calculate_bm25_score(self, query, document):
        score = 0.0
        document_length = len(document)
        query_terms = Counter(query)
        for term in query_terms:
        if term not in document:
        continue
        idf = self.calculate_idf(term)
        term_frequency = document.count(term)
        numerator = term_frequency * (self.k1 + 1)
        denominator = term_frequency + self.k1 * (1 - self.b + self.b * (document_length /
        self.avg_document_length))
        score += idf * (numerator / denominator)
        return score

def rank_documents(self, query):
        document_scores = []
        for document in self.documents:
        score = self.calculate_bm25_score(query, document)
        document_scores.append((document, score))
        print(document_scores)
        ranked_documents = sorted(document_scores, key=lambda x: x[1], reverse=True)
        return ranked_documents

def read_text_from_docx(docx_path):
        doc = Document(docx_path)
        text = [paragraph.text for paragraph in doc.paragraphs]
        return text

def read_documents_from_directory(directory_path):
        documents = []
        for filename in os.listdir(directory_path):
        if filename.endswith(".txt"):
        file_path = os.path.join(directory_path, filename)
        with open(file_path, 'r', encoding='utf-8') as file:
        document_text = file.read().split()
        documents.append((filename, document_text))
        elif filename.endswith(".docx"):
        file_path = os.path.join(directory_path, filename)
        document_text = read_text_from_docx(file_path)
        documents.append((filename, document_text))
        return documents
def get_input():
        js_code = '''
        prompt("Enter the directory path:")
        '''
        return eval_js(js_code)
        directory_path = get_input()
        query = input("Enter the query terms separated by space: ").split()
        # documents = read_documents_from_directory(directory_path)
        # bm25 = BM25([doc[1] for doc in documents])
        # ranked_documents = bm25.rank_documents(query)
        # for idx, (document, score) in enumerate(ranked_documents, start=1):
        # if any(term in document for term in query):
        # print(f"Document_{idx:03}: {document[0]}, Score: {score}")

documents = read_documents_from_directory(directory_path)
bm25 = BM25([doc[1] for doc in documents])
ranked_documents = bm25.rank_documents(query)
for idx, (document, score) in enumerate(ranked_documents[:25], start=1):
if any(term in document for term in query):
print(f"Document_{idx:3}: {documents[idx-1][0]}, Score: {score}")
```

AI Interactive Model

```
!pip install pypdf
!pip install python-dotenv
!pip install -q accelerate==0.21.0 peft==0.4.0 bitsandbytes==0.40.2 transformers==4.31.0

trl==0.4.7
import os
import torch
```

```
from datasets import load_dataset
from transformers import (
AutoModelForCausalLM,
AutoTokenizer,
BitsAndBytesConfig,
HfArgumentParser,
TrainingArguments,
pipeline,
logging,
)
from peft import LoraConfig, PeftModel
from trl import SFTTrainer
import pandas as pd

# The model that you want to train from the Hugging Face hub
model_name = "NousResearch/Llama-2-7b-chat-hf"
# The instruction dataset to use
df = pd.read_csv("/content/bbc_llamma_data.csv")
# Fine-tuned model name
new_model = "/content/Llama-2-7b-chat-finetune-buvi-docchat"

###############################################################################
#####
# QLoRA parameters
###############################################################################
#####
# LoRA attention dimension
lora_r = 64
# Alpha parameter for LoRA scaling
lora_alpha = 16
# Dropout probability for LoRA layers
lora_dropout = 0.1

###############################################################################
#####
# bitsandbytes parameters
###############################################################################
#####
# Activate 4-bit precision base model loading
use_4bit = True
# Compute dtype for 4-bit base models
bnb_4bit_compute_dtype = "float16"
# Quantization type (fp4 or nf4)
bnb_4bit_quant_type = "nf4"
# Activate nested quantization for 4-bit base models (double quantization)
use_nested_quant = False

###############################################################################
#####
# TrainingArguments parameters
###############################################################################
#####
# Output directory where the model predictions and checkpoints will be stored
output_dir = "/content/results"
# Number of training epochs
```

```
num_train_epochs = 1

# Enable fp16/bf16 training (set bf16 to True with an A100)
fp16 = False
bf16 = False
# Batch size per GPU for training
per_device_train_batch_size = 4
# Batch size per GPU for evaluation
per_device_eval_batch_size = 4
# Number of update steps to accumulate the gradients for
gradient_accumulation_steps = 1

# Enable gradient checkpointing
gradient_checkpointing = True
# Maximum gradient normal (gradient clipping)
max_grad_norm = 0.3
# Initial learning rate (AdamW optimizer)
learning_rate = 2e-4
# Weight decay to apply to all layers except bias/LayerNorm weights
weight_decay = 0.001

# Optimizer to use
optim = "paged_adamw_32bit"
# Learning rate schedule
lr_scheduler_type = "cosine"
# Number of training steps (overrides num_train_epochs)
max_steps = -1
# Ratio of steps for a linear warmup (from 0 to learning rate)
warmup_ratio = 0.03
# Group sequences into batches with same length
# Saves memory and speeds up training considerably
group_by_length = True
# Save checkpoint every X updates steps
save_steps = 0
# Log every X updates steps
logging_steps = 25

################################################################################
#####
# SFT parameters
################################################################################
#####
# Maximum sequence length to use
max_seq_length = None
# Pack multiple short examples in the same input sequence to increase efficiency
packing = False
# Load the entire model on the GPU 0
device_map = {"": 0}

# # Load dataset (you can process it here)
# dataset = load_dataset(dataset_name, split="train")
# Load tokenizer and model with QLoRA configuration
compute_dtype = getattr(torch, bnb_4bit_compute_dtype)
bnb_config = BitsAndBytesConfig(
load_in_4bit=use_4bit,
bnb_4bit_quant_type=bnb_4bit_quant_type,
bnb_4bit_compute_dtype=compute_dtype,
bnb_4bit_use_double_quant=use_nested_quant,
)
# Check GPU compatibility with bfloat16
if compute_dtype == torch.float16 and use_4bit:
major, _ = torch.cuda.get_device_capability()
if major >= 8:
print("=" * 80)
print("Your GPU supports bfloat16: accelerate training with bf16=True")
print("=" * 80)
# Load base model

model = AutoModelForCausalLM.from_pretrained(
model_name,
quantization_config=bnb_config,
device_map=device_map
)

model.config.use_cache = False
model.config.pretraining_tp = 1

# Load LLaMA tokenizer
tokenizer = AutoTokenizer.from_pretrained(model_name, trust_remote_code=True)
tokenizer.pad_token = tokenizer.eos_token
tokenizer.padding_side = "right" # Fix weird overflow issue with fp16 training
# Load LoRA configuration

peft_config = LoraConfig(
lora_alpha=lora_alpha,
lora_dropout=lora_dropout,
r=lora_r,
bias="none",
task_type="CAUSAL_LM",
)

# Set training parameters
training_arguments = TrainingArguments(
output_dir=output_dir,
num_train_epochs=num_train_epochs,
per_device_train_batch_size=per_device_train_batch_size,
gradient_accumulation_steps=gradient_accumulation_steps,
optim=optim,
save_steps=save_steps,
logging_steps=logging_steps,
learning_rate=learning_rate,
weight_decay=weight_decay,
fp16=fp16,
bf16=bf16,
max_grad_norm=max_grad_norm,
max_steps=max_steps,
warmup_ratio=warmup_ratio,
group_by_length=group_by_length,
lr_scheduler_type=lr_scheduler_type,
report_to="tensorboard"
```

```
)

# Set supervised fine-tuning parameters
trainer = SFTTrainer(
model=model,
train_dataset=dataset,
peft_config=peft_config,
dataset_text_field="text",
max_seq_length=max_seq_length,
tokenizer=tokenizer,
args=training_arguments,
packing=packing,
)

# Train model
trainer.train()
trainer.model.save_pretrained(new_model)
%reload_ext tensorboard
%tensorboard --logdir results/runs

prompt_template = """### Instruction:
Given the following ethusiastic person and domain and question, your job is to provide me
news,social impact.

### Input:
{}

### Response:
"""

sentence ="Enthusiastic:Business,Domain:MBA,Question:what is recent trend in stock
market?"

input_sentence = prompt_template.format(sentence.strip())
pipe = pipeline(task="text-generation", model=model, tokenizer=tokenizer, max_length=1000)
result = pipe(input_sentence)
print(result)
```

REFERENCES

[1] Guozhen Shi1 , Mang Su2 , Fenghua Li3 , Jiapeng Lou1 , Qiong Huang1 - “A User-based Document Management Mechanism in Cloud” - 2014 Tenth International Conference on Computational Intelligence and Security

[2] Jinbo Xiong, Zhiqiang Yao ,Jun Ma, Ximeng Liu, Qi Li -”Structured Document Model and Its Secure Access Control in Cloud Computing “-2013 International Conference on Cloud Computing and Big Data

[3] Cong Wang, Ning Cao, Jin Li, Kui Ren, and Wenjing Lou - “Secure Ranked Keyword Search over Encrypted Cloud Data” - 2010 International Conference on Distributed Computing Systems

[4] Ning Cao , Cong Wang , Ming Li , Kui Ren , and Wenjing Lou - “Privacy-Preserving Multi-keyword Ranked Search over Encrypted Cloud Data “-2010 International Conference on Distributed Computing Systems

[5] Madura Rajapashea , Muammar Adnanb , Ashen Dissanayakac , Dasith Guneratned , Kavinga Abeywardane - “Multi-Format Document Verification System” - American Scientific Research Journal for Engineering, Technology, and Sciences · December 2020

[6] “Modern Lossless Compression Techniques: Review, Comparison and Analysis “-Apoorv Gupta1 , Aman Bansal1 , Vidhi Khanduja-February 2020
[7] Jiayi Li , Jianfeng Ma Yinbin Miao , Ruikang Yang , Ximeng Liu , and Kim-Kwang Raymond Choo , - “Practical Multi-Keyword Ranked Search With Access Control Over Encrypted Cloud Data”- IEEE transactions on cloud computing, vol. 10, no. 3, july-september 2022

[8] Ming Li, Shucheng Yu, Ning Cao and Wenjing Lou - “Authorized Private Keyword Search over Encrypted Data in Cloud Computing” - 2011 31st International Conference on Distributed Computing Systems

[9] Chia Hung Kao, Shin Tzu Liu - “Development of a Document Management System for Private Cloud Environment” - The 2nd International Conference on Integrated Information

[10] Anuradha N. M,G. A. Patil - “Secure and Efficient Data Retrieval in Cloud Computing “- International Journal of Engineering Research & Technology (IJERT) ISSN: 2278-0181 Vol. 4 Issue 04, April-2015

[11] Nandini K1 , Faisal. S2 , Shailendra B3 , Shree Vallabha S4 , Pavan B5 - “Secure File Storage on Cloud Using Cryptography” - International Journal for Research in Applied Science & Engineering Technology (IJRASET)

[12] BBC News Dataset :
(https://www.kaggle.com/datasets/shivamkushwaha/bbc-full-text-document-classification)

[13] Secure File Storage on Cloud Computing Using Cryptographic Algorithm - July , 2017

[14] Achieving Efficient Conjunctive Keyword Searches over Encrypted Data - October , 2018

[15] Lossless data compression techniques and their performance - July , 2018

[16] Lossless data compression technique with encryption based approach - September , 2018

[17] Structured Document Model and Its Secure Access Control in Cloud Computing - October,2019

[18] A User-based Document Management Mechanism in Cloud - November , 2019

[19] Secure and Efficient Data Retrieval in Cloud Computing - November , 2019

[20] Secure Ranked Keyword Search over Encrypted Cloud Data - November, 2019

[21] Efficient Multi-Keyword Ranked Query on Encrypted Data in the Cloud - July , 2019

[22] Secure File Storage on Cloud Using Cryptography - December , 2019

[23] Secure Hashing Algorithms and Their Comparison - March , 2019

[24] Implementation of Enhanced Secure Hash Algorithm Towards a Secured Web Portal - September , 2019

[25] A Secured Cryptographic Hashing Algorithm - November , 2019

[26] An Analysis on Different Document Keyword Extraction Methods - October , 2019

[27] Keyword Extraction: A Modern Perspective - September , 2019

[28] Multi-Format Document Verification System - December , 2020

[29] Development of a Document Management System for Private Cloud Environment - March ,2020

[30] Comparison of Lossless Data Compression Techniques - October , 2020

[31] Efficient File Upload and Mutli-Keyword Search over Encrypted Cloud Data - December , 2020

[32] Development of online cloud storage technology - September , 2020

[33] Authorized Private Keyword Search over Encrypted Data in Cloud Computing - September , 2021

[34] Security and Security and Privacy Privacy Privacy Issues in Cloud Computing - October , 2021

[35] Practical Multi-Keyword Ranked Search With Access Control Over Encrypted Cloud Data - 3, July-September 2022

[36] Privacy-Preserving Multi-keyword Ranked Search over Encrypted Cloud Data - September , 2022

[37] Modern Lossless Compression Techniques: Review, Comparison and Analysis - November , 2022

[38] Venkata Sravan Kumar Maddineni Shivashanker Ragi-Venkata Sravan Kumar Maddineni Shivashanker Ragi-Master Thesis Electrical Engineering November 2011

[39] Xiuxiu Jiang,Jingbo Yan,Rong Hao-Enabling efficient and verifiable multi-keyword ranked search over encrypted cloud data-Information Sciences-Volumes 403–404, September 2017

[40] D. Shanmuga Priya, T. S. Mahima, C. Priyanga and M. Geetha- "Practical Multi-Keyword Ranked Search with TimeBased Access Control over Encrypted Cloud Data " Volume 5, Issue 2, May 2021

[41] Tarasvi lakum, 1 prof.b. tirapathi reddyan "Efficient file access control technique for shared cloud data security through key signatures search scheme" - Journal of Theoretical and Applied Information Technology 15th January 2022. Vol.100. No 1

[42] Xin Li-Access Control Strategy Based on Trust under Cloud Computing Platform-2018 International Conference on Virtual Reality and Intelligent Systems

[43] J. Persis Jessintha, Dr. R. Anbuselvi- "An Analysis on Access Control Mechanisms in Cloud Environment" -NCICN – 2015 (Volume 3 – Issue 07)-IJERT

[44] Vincent C. Hu Michaela Iorga Wei Bao Ang Li Qinghua Li Antonios Gouglidis- "General Access Control Guidance for Cloud Systems" -NIST Special Publication 800-210

[45] Yogesh M. Gajmal1, Dr. K. P. Thooyamani-"Data Access control schemes in cloud computing: a review-International Research Journal of Engineering and Technology (IRJET)"

[46] Greeshma N. Gopal, Mahendra Pratap Singh- "Secure Similarity Based Document Retrieval System in Cloud"-National institute of technology,April 16,2021

[47] S.Annal Ezhil Selvi1, Dr. R. Anbuselvi-"Ranking Algorithm Based on File's Accessing Frequency for Cloud Storage System-International Journal of Advanced Research Trends in Engineering and Technology (IJARTET) "
Vol. 4, Issue 9, September 2017

[48] Vimmi Makkar 2, Sandeep Dalal- "Ranked Keyword Search in Cloud Computing": An Innovative Approach-International Journal of Computational Engineering Research-Vol, 03

[49] Xuelong Dai,Hua Da"Enhanced Semantic-Aware Multi-Keyword Ranked Search Scheme Over Encrypted Cloud Data" - December 2022

[50] Y.Li,J.Liu,andW.Lou""A Secure Document Ranking Scheme in Cloud Computing"- January 2022

[51] A.Jain,V.Shrivastava,andS.Agarwal- "Access Control of Documents in Cloud Computing Environments" - January 2022

[52] S.Singh and R.Gupta- "Efficient Document Processing in Cloud Computing"- December 2022

[53]X.Zhang,Y.ZhangandQ.Cheng"Privacy-Preserving Document Ranking in Cloud-Based Information Retrieval"January 2021

[54] P.Kumar,Q.Zhao,R.Singh "Secure Document Ranking and Access Control in Cloud Environments"- October 2019

[55] D.Brown,E.Martinez,F.Kim"Advanced Access Control Policies for Cloud-based Document Management"- Novemebr 2019

[56]P.Gupta,S.Patel,T.Nguyen"Scalable Document Processing Framework for Cloud Computing Infrastructures"-December 2019

[57] A.Lee,S.Kumar,J.Wang "Fine-Grained Access Control Techniques for Documents in Cloud Storage"- October 2017

[58] L.Chen,M.Kim,N.Tan "Dynamic Access Control and Ranking in Cloud-based Document Repositories"- March 2019

[60] P.Sharma,R.Gupta,S.Zhang"Privacy-Aware Document Processing and Access Control in Cloud Environments"July 2019

[61] A.Smith,B.Johnson,C.Wang"Enhanced Access Control Mechanisms for Cloud Document Repositories"January 2023

[62] X.Chen,Y.Liu,Z.Zhang"Efficient Document Processing Techniques for Cloud-based Environments"- March 2022

[63] L.Gupta,N.Patel,O.Lee "Privacy-Preserving Document Ranking in Cloud-based Information Retrieval Systems" -Nocember 2022

[64] L.Lee,S.Gupta,C.Wang "Blockchain-Based Access Control for Cloud Document Sharing Platforms"June 2020

[65] R. Aravind- “An Efficient Search in Cloud Computing Using Ranked Keyword Search Algorithm”-International Journal of Science and Research (IJSR)-2012

[66] “An Overview and Analysis of Hybrid Encryption: The Combination of Symmetric Encryption and Asymmetric Encryption” -2021 2nd International Conference on Computing and Data Science (CDS)-28-29 Jan. 2021

PUBLICATIONS

CONFERENCE PUBLICATION

AUTHORS :

A.Buvaneshkumaar

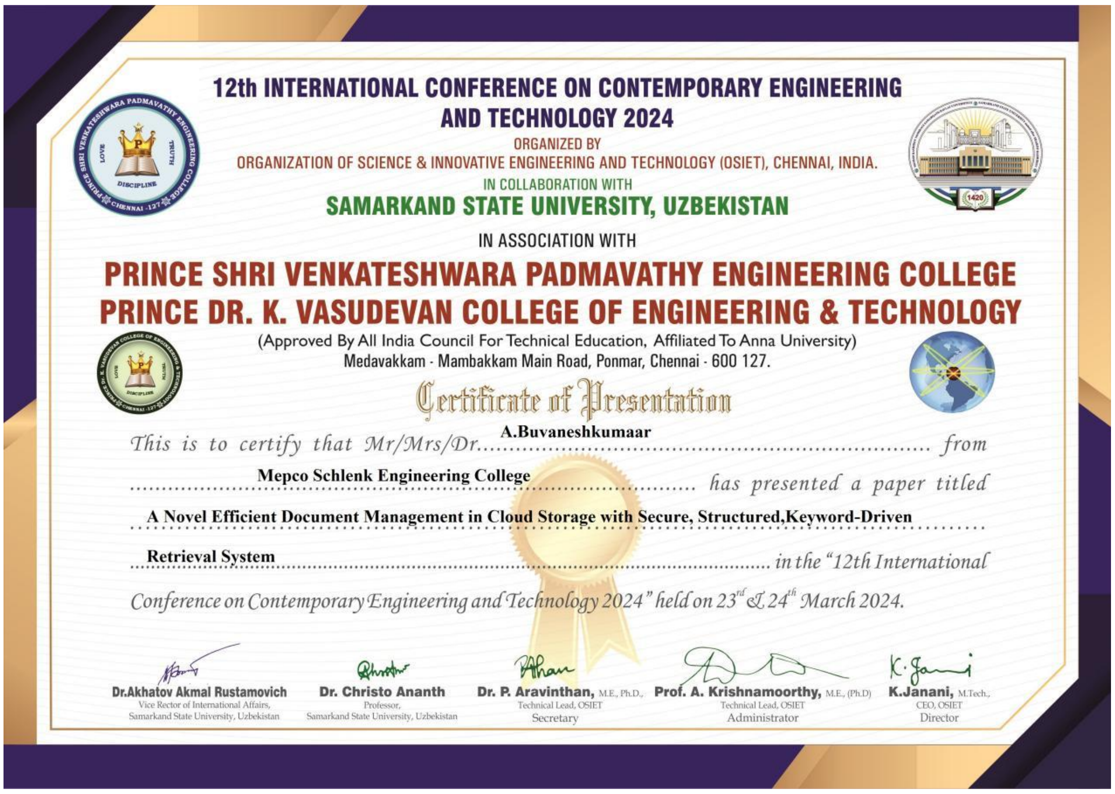
P.Vimal

CONFERENCE NAME : 12th International Conference on Contemporary Engineering and Technology 2024

RESEARCH PAPER NAME : A Novel Efficient Document Management in Cloud Storage with Secure, Structured, Keyword - Driven Retrieval System.

PROGRESS : Submitted (Certified)

CERTIFICATES



12th INTERNATIONAL CONFERENCE ON CONTEMPORARY ENGINEERING AND TECHNOLOGY 2024

ORGANIZED BY

ORGANIZATION OF SCIENCE & INNOVATIVE ENGINEERING AND TECHNOLOGY (OSIET), CHENNAI, INDIA.

IN COLLABORATION WITH

SAMARKAND STATE UNIVERSITY, UZBEKISTAN

IN ASSOCIATION WITH

PRINCE SHRI VENKATESHWARA PADMAVATHY ENGINEERING COLLEGE

PRINCE DR. K. VASUDEVAN COLLEGE OF ENGINEERING & TECHNOLOGY

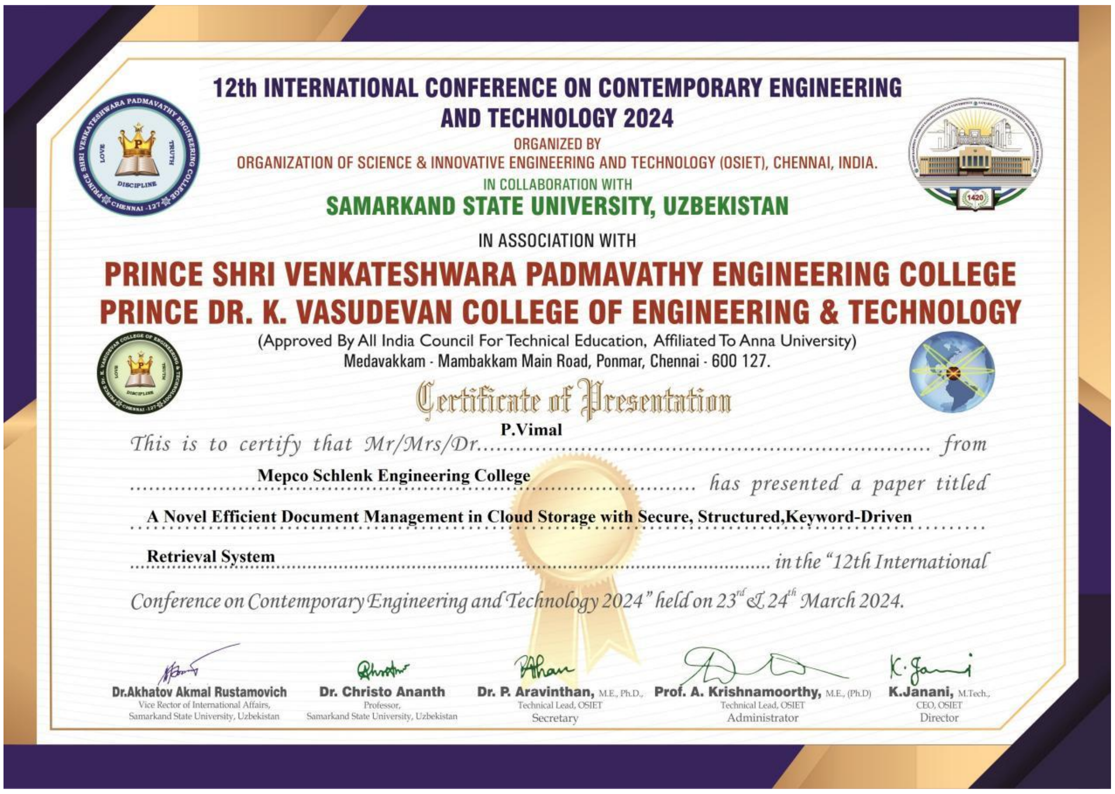
(Approved By All India Council For Technical Education, Affiliated To Anna University)

Medavakkam - Mambakkam Main Road, Ponmar, Chennai - 600 127.

Certificate of Presentation

This is to certify that Mr/Mrs/Dr. **A.Buvaneshkumaar** *from* **Mepco Schlenk Engineering College** *has presented a paper titled* **A Novel Efficient Document Management in Cloud Storage with Secure, Structured,Keyword-Driven Retrieval System** *in the "12th International Conference on Contemporary Engineering and Technology 2024" held on 23rd & 24th March 2024.*

[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Dr.Akhatov Akmal Rustamovich	Dr. Christo Ananth	Dr. P. Aravinthan, M.E., Ph.D.,	Prof. A. Krishnamoorthy, M.E., (Ph.D)	K.Janani, M.Tech.,
Vice Rector of International Affairs, Samarkand State University, Uzbekistan	Professor, Samarkand State University, Uzbekistan	Technical Lead, OSIET Secretary	Technical Lead, OSIET Administrator	CEO, OSIET Director



12th INTERNATIONAL CONFERENCE ON CONTEMPORARY ENGINEERING AND TECHNOLOGY 2024

ORGANIZED BY

ORGANIZATION OF SCIENCE & INNOVATIVE ENGINEERING AND TECHNOLOGY (OSIET), CHENNAI, INDIA.

IN COLLABORATION WITH

SAMARKAND STATE UNIVERSITY, UZBEKISTAN

IN ASSOCIATION WITH

PRINCE SHRI VENKATESHWARA PADMAVATHY ENGINEERING COLLEGE

PRINCE DR. K. VASUDEVAN COLLEGE OF ENGINEERING & TECHNOLOGY

(Approved By All India Council For Technical Education, Affiliated To Anna University)

Medavakkam - Mambakkam Main Road, Ponmar, Chennai - 600 127.

Certificate of Presentation

This is to certify that Mr/Mrs/Dr. **P.Vimal** *from* **Mepco Schlenk Engineering College** *has presented a paper titled* **A Novel Efficient Document Management in Cloud Storage with Secure, Structured,Keyword-Driven Retrieval System** *in the "12th International Conference on Contemporary Engineering and Technology 2024" held on 23rd & 24th March 2024.*

[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Dr.Akhatov Akmal Rustamovich	Dr. Christo Ananth	Dr. P. Aravinthan, M.E., Ph.D.,	Prof. A. Krishnamoorthy, M.E., (Ph.D)	K.Janani, M.Tech.,
Vice Rector of International Affairs, Samarkand State University, Uzbekistan	Professor, Samarkand State University, Uzbekistan	Technical Lead, OSIET Secretary	Technical Lead, OSIET Administrator	CEO, OSIET Director

JOURNAL SUBMISSION

AUTHORS :

Mrs.P.Swathika

A.Buvaneshkumaar

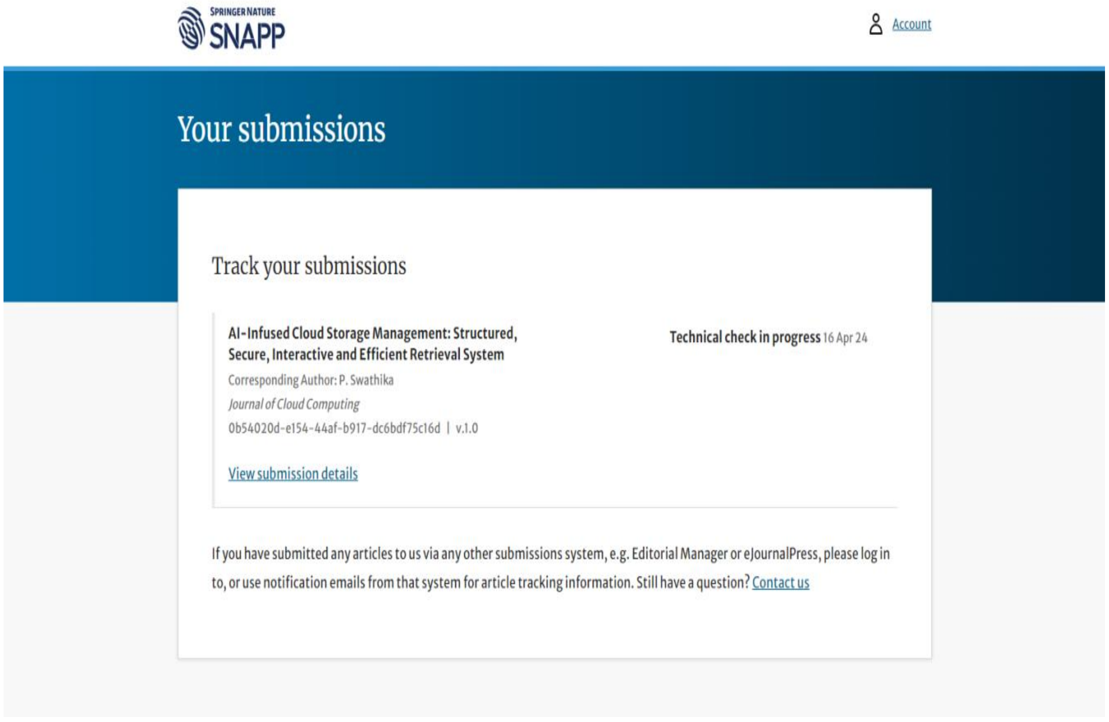
P.Vimal

JOURNAL NAME : Journal of Cloud Computing

RESEARCH PAPER NAME : AI-Infused Cloud Storage Management: Structured, Secure, Interactive and Efficient Retrieval System

PROGRESS : Submitted (Ongoing)

SUBMISSION PROOF 1



SPRINGER NATURE SNAPP

Account

Your submissions

Track your submissions

AI-Infused Cloud Storage Management: Structured, Secure, Interactive and Efficient Retrieval System

Corresponding Author: P. Swathika

Journal of Cloud Computing

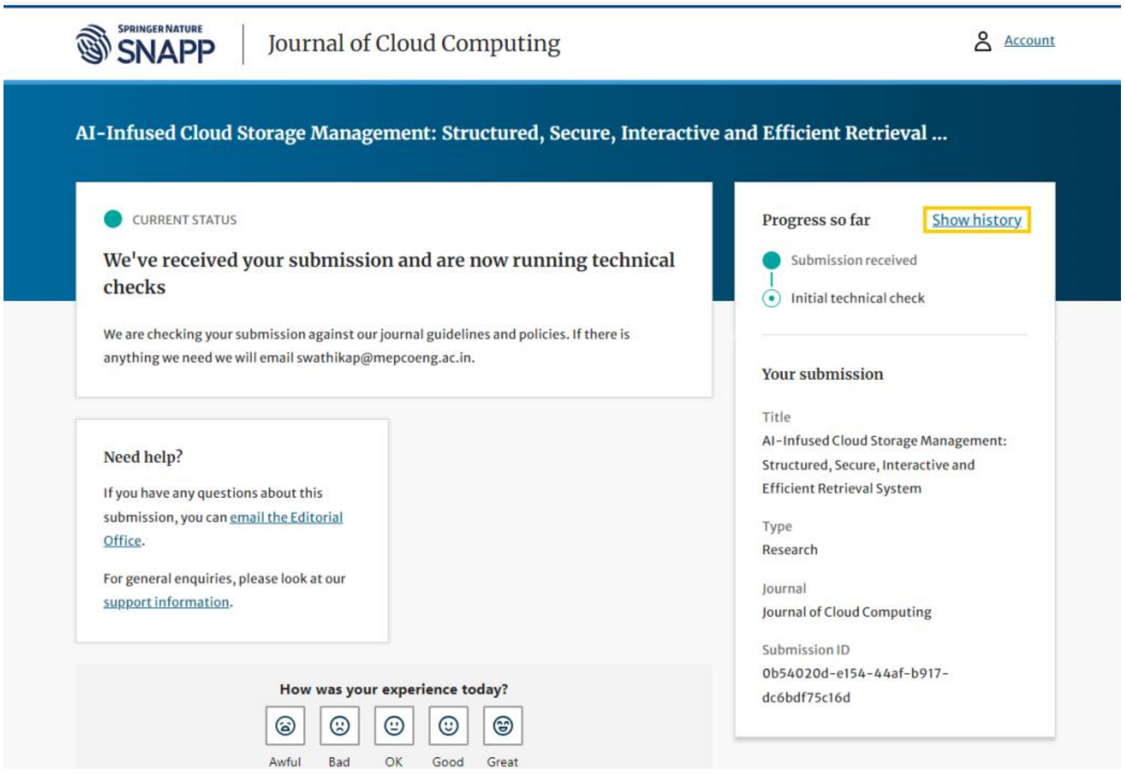
0b54020d-e154-44af-b917-dc6bdf75c16d | v.1.0

View submission details

Technical check in progress 16 Apr 24

If you have submitted any articles to us via any other submissions system, e.g. Editorial Manager or eJournalPress, please log in to, or use notification emails from that system for article tracking information. Still have a question? Contact us

SUBMISSION PROOF 2



SPRINGER NATURE SNAPP | Journal of Cloud Computing

Account

AI-Infused Cloud Storage Management: Structured, Secure, Interactive and Efficient Retrieval ...

CURRENT STATUS

We've received your submission and are now running technical checks

We are checking your submission against our journal guidelines and policies. If there is anything we need we will email swathikap@mepcoeng.ac.in.

Need help?

If you have any questions about this submission, you can email the Editorial Office.

For general enquiries, please look at our support information.

How was your experience today?

Awful Bad OK Good Great

Progress so far Show history

Submission received

Initial technical check

Your submission

Title
AI-Infused Cloud Storage Management: Structured, Secure, Interactive and Efficient Retrieval System

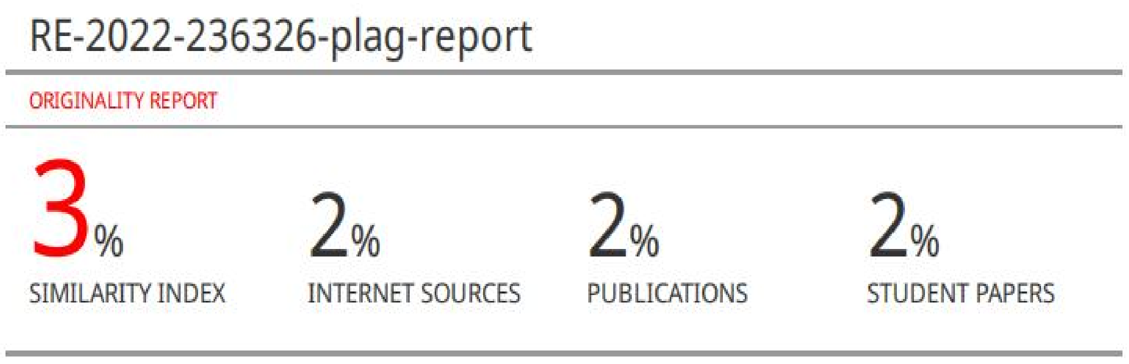
Type
Research

Journal
Journal of Cloud Computing

Submission ID
0b54020d-e154-44af-b917-dc6bdf75c16d

PLAGIARISM REPORT

JOURNAL PAPER

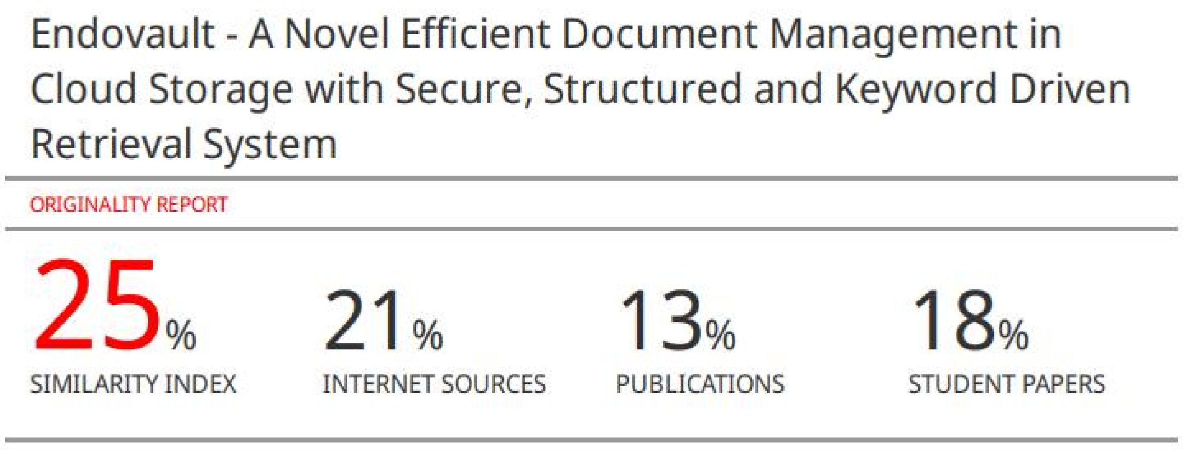


RE-2022-236326-plag-report

ORIGINALITY REPORT

3%	2%	2%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

FINAL PROJECT REPORT



Endovault - A Novel Efficient Document Management in Cloud Storage with Secure, Structured and Keyword Driven Retrieval System

ORIGINALITY REPORT

25%	21%	13%	18%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS