International Journal of Information Technology & Computer Engineering

ISSN 2347–3657

Volume 14, Issue 2, 2026

Privacy-Preserving Searchable Encryption For Web Services

B Anitha[1], Anjali Pothula[2], Poojitha Banoth[3], Archana Thokala[4]

[1]Associate Professor; Department Of Information Technology Bhoj Reddy Engineering College For Women Hyderabad India.

[2,3,4]B.Tech Students; Department Of Information Technology Bhoj Reddy Engineering College For Women Hyderabad India.

Mail Id; anjalipothula10@gmail.com, banothpoojitha15@gmail.com, archanakmp43@gmail.com

Abstract

Cloud-based web platforms routinely manage large collections of confidential information, making secure storage and controlled retrieval essential requirements. Conventional encryption techniques protect files during storage, but they often prevent direct keyword searching unless the data is first decrypted, which may increase privacy risks and computational overhead. To overcome this limitation, this study presents a Privacy-Preserving Searchable Encryption (PPSE) framework that enables secure keyword-based retrieval over encrypted documents.

In the proposed system, files are encrypted prior to cloud storage, while document keywords are converted into protected searchable indexes using a Public Key Encryption with Keyword Search (PEKS) mechanism. This approach allows authorized users to search encrypted content without disclosing the original keywords or document contents to the storage server. The framework is implemented as a web-based application incorporating secure user authentication, cryptographic key management, and a MySQL database for encrypted data storage.

For data confidentiality, the system employs AES-256-CBC encryption for document protection and an elliptic curve cryptography (ECC)-based PEKS scheme for trapdoor generation and keyword matching. Search requests are processed through secure trapdoors, enabling efficient comparison operations while preserving privacy. Experimental evaluation indicates that the framework achieves reliable search accuracy, low processing overhead, and scalable performance with growing document volumes.

The proposed solution is particularly suitable for domains requiring strict confidentiality, including healthcare repositories, legal archives, and enterprise document management systems. The study demonstrates that privacy-preserving searchable encryption can provide a practical balance between security, usability, and retrieval efficiency in modern cloud environments.

Keywords: *Privacy-Preserving Searchable Encryption, Cloud Security, Public Key Encryption with Keyword Search, Secure Retrieval, Trapdoor Search, Encrypted Data Management.*

Introduction

The rapid expansion of cloud-based services has transformed the way organizations and individuals store digital information. Cloud platforms offer advantages such as scalability, remote accessibility, reduced infrastructure cost, and simplified data management. Despite these benefits, outsourcing data storage to third-party servers introduces serious concerns regarding confidentiality, unauthorized access, and privacy protection. Sensitive information such as medical records, financial reports, legal files, and enterprise documents often resides on remote infrastructure, making secure handling essential.

Conventional encryption methods are widely used to protect outsourced data. By converting plaintext into ciphertext, encryption prevents unauthorized users from reading stored content. However, traditional encryption creates a practical limitation: once data is encrypted, servers cannot directly process search queries or perform keyword-based retrieval without first decrypting the content. Requiring decryption during search operations increases computational overhead and may expose confidential information. This challenge becomes more significant when managing large-scale datasets in dynamic cloud environments.

Privacy-Preserving Searchable Encryption (PPSE) has emerged as an effective solution to this problem. PPSE allows users to search encrypted data while ensuring that document contents and search queries remain hidden from the storage server. Instead of revealing keywords in plaintext, cryptographic techniques generate secure searchable representations that enable matching operations without disclosing sensitive information.

In this project, a PPSE framework based on Public Key Encryption with Keyword Search (PEKS) is developed for secure document storage and retrieval. Documents are encrypted using strong symmetric cryptography before storage, while keywords are transformed into protected searchable ciphertexts. During retrieval, authorized users generate a trapdoor corresponding to the requested keyword. The server uses this trapdoor to compare encrypted indexes and identify matching files without learning the actual keyword or document contents.

Existing System

In existing cloud storage systems, files are commonly protected using standard encryption methods before being uploaded to remote servers. While this ensures confidentiality during storage, it prevents direct keyword-based searching over encrypted files. Users must download encrypted documents, decrypt them locally, and manually inspect their contents to locate required information. This process is inefficient, time-consuming, and unsuitable for large datasets. It also increases the possibility of exposing confidential information during retrieval. Since the server cannot interpret encrypted content, it cannot assist in performing intelligent or efficient search operations.

Proposed System

The proposed system introduces a Privacy-Preserving Searchable Encryption model that enables secure storage and efficient keyword retrieval over encrypted data. Documents are encrypted using AES-based symmetric encryption before being stored in the database. Keywords extracted from each document are converted into searchable ciphertexts using a PEKS-based mechanism.

Survey

Searchable encryption has become a major research area in secure cloud computing because users increasingly rely on third-party servers to store confidential information. Although encryption protects privacy, it limits the ability to perform efficient retrieval operations. To address this issue, early research introduced methods for searching encrypted data without revealing plaintext contents. A significant milestone was the Public Key Encryption with Keyword Search (PEKS) model proposed by Boneh et al. This approach allows a server to test whether an encrypted keyword matches a user-generated trapdoor without learning the keyword itself. PEKS established the foundation for secure keyword search in public-key environments.

Later studies focused on Searchable Symmetric Encryption (SSE), where search operations are performed using symmetric keys. SSE methods provide faster performance and lower computational cost compared to public-key techniques. They are highly efficient in single-owner storage systems but face challenges in multi-user scenarios due to key distribution and access control requirements.

Researchers also introduced dynamic searchable encryption schemes that support insertion, deletion, and modification of encrypted files. These methods improve practicality for real-world systems where datasets continuously evolve. Additional concepts such as forward privacy were developed to prevent servers from linking newly added files to earlier search queries.

Recent work has explored advanced retrieval capabilities including multi-keyword search, ranked search, semantic search, and fuzzy keyword matching. Multi-keyword search allows simultaneous querying of multiple terms, ranked retrieval improves relevance ordering, and fuzzy search handles spelling variations or typing mistakes. While these features enhance usability, they often increase computational overhead and may reveal limited access patterns.

To further strengthen privacy, techniques such as Oblivious RAM (ORAM) and Private Information Retrieval (PIR) have been proposed. These methods conceal user access behavior and query patterns from the server. However, their implementation complexity and high resource consumption make them less suitable for lightweight web applications.

Overall, prior research provides strong theoretical and practical foundations for searchable encryption. Nevertheless, balancing privacy, scalability, and efficiency remains an open challenge. The PPSE system presented in this project builds upon PEKS-based searchable encryption and integrates document encryption, secure indexing, and trapdoor search into a practical web-based application for confidential data retrieval.

Architecture

System Architecture

The system architecture of the Privacy-Preserving Searchable Encryption (PPSE) framework is designed as a secure multi-layer structure that protects both stored data and user search queries. The architecture consists of three primary components: the User (Client), the Application Server, and the Database Server. These components work together to provide secure document storage and confidential retrieval services. The user interacts with the system through a web-based interface for registration, login, document upload, and keyword-based search operations. During the upload process, the selected document is encrypted using the AES-256-CBC algorithm to preserve confidentiality. At the same time, relevant keywords associated with the document are converted into searchable ciphertexts using a Public Key Encryption with Keyword Search (PEKS) mechanism. The encrypted document and protected keyword indexes are securely transmitted to the server and stored in the database.

During the search process, the user enters a keyword query through the interface. Instead of transmitting the actual keyword, the system generates a secure trapdoor using the user's cryptographic credentials. This trapdoor is sent to the server, where matching operations are performed against the stored encrypted indexes using the Test() function. The server can identify relevant encrypted files without learning the original keyword or accessing the contents of the stored documents. Only the matched encrypted documents are returned to the user, who can then decrypt them using authorized keys. This

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 2, 2026

architecture ensures that all sensitive information remains protected throughout storage and retrieval, while the server processes only encrypted data.

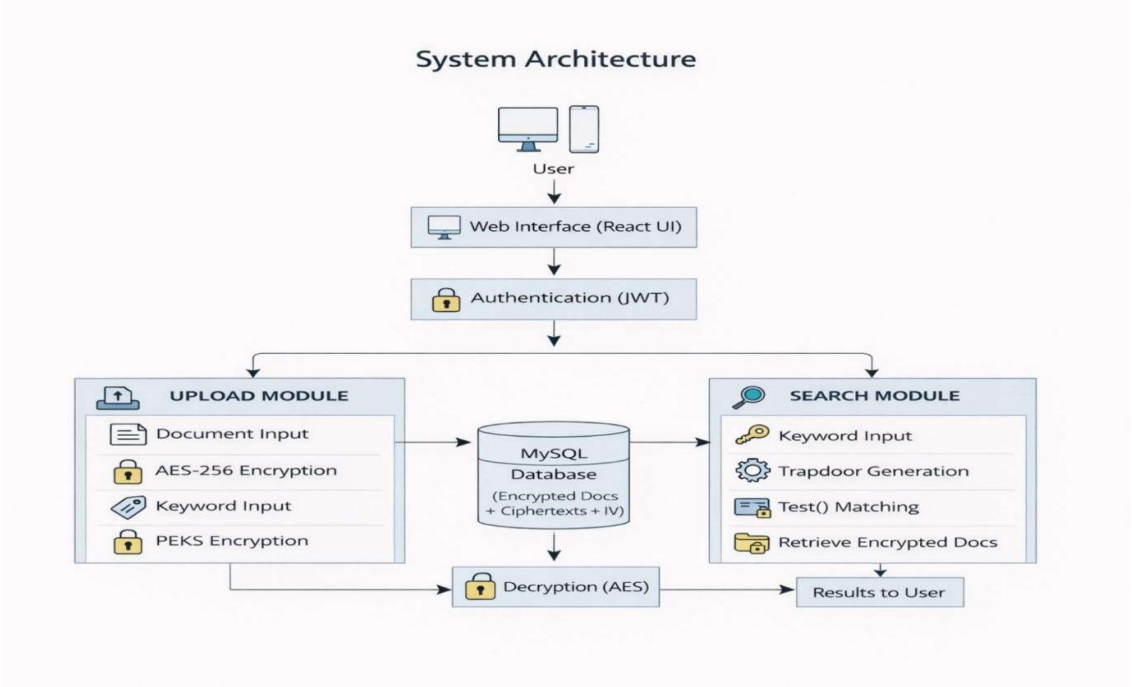


Fig. 1 System Architecture

Technical Architecture

The technical architecture of the PPSE system follows a three-tier layered model consisting of the frontend layer, backend layer, and database layer. The frontend layer is developed using React.js and provides an interactive user interface for registration, login, document upload, document management, and secure keyword search. It communicates with backend services through secure API requests. The backend layer is implemented using Node.js and Express.js, which manage application logic such as authentication, file processing, encryption, keyword indexing, trapdoor generation, and search execution. This layer integrates cryptographic modules including AES-256-CBC for document encryption and an elliptic curve cryptography (ECC)-based PEKS mechanism for searchable encryption.

The database layer uses MySQL for persistent storage of encrypted files, initialization vectors, encrypted session keys, searchable ciphertext indexes, and user account information. Plaintext files and original keywords are never stored in the database, which significantly improves confidentiality. JWT-based authentication is used to verify user identity and secure communication between the frontend and backend. This layered technical architecture ensures efficient interaction among system components, supports scalability for larger datasets, and maintains strong privacy by ensuring that sensitive data remains encrypted during all operations.

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 2, 2026

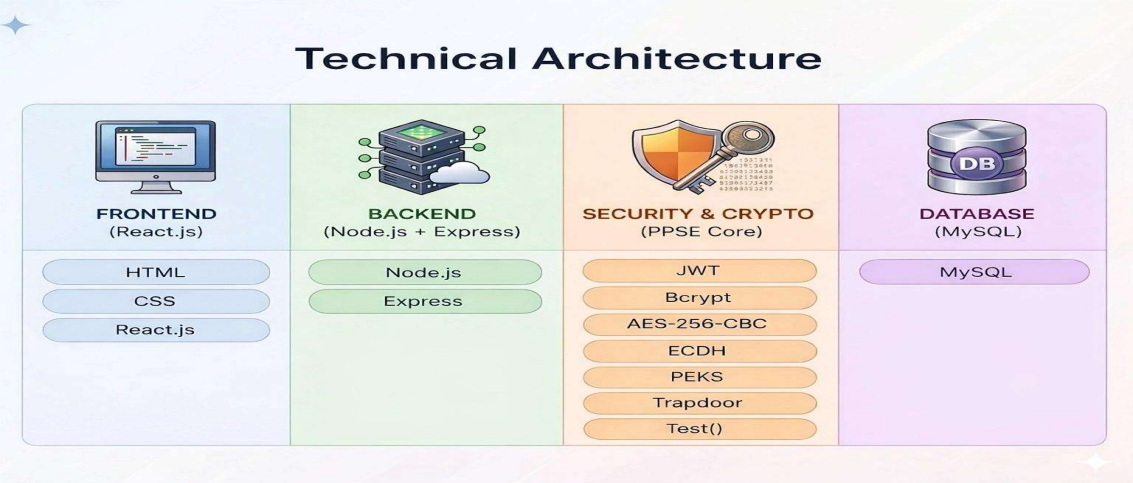


Fig. 2 Technical Architecture

Workflow Diagram

The workflow of the PPSE system begins with user authentication, where registered users log in using valid credentials and receive a JWT token for secure access to authorized services. After successful login, the user can choose either the upload operation or the search operation. In the upload workflow, the user selects a document to be stored in the system. The application extracts relevant keywords from the document or its metadata. A random AES secret key and initialization vector are then generated, and the document is encrypted using AES-256-CBC. The extracted keywords are converted into searchable ciphertexts using the PEKS algorithm. Finally, the encrypted document, initialization vector, encrypted key, and searchable indexes are stored in the database.

In the search workflow, the user enters a keyword query through the web interface. The system converts this keyword into a secure trapdoor instead of sending the plaintext term. The trapdoor is forwarded to the server, where it is matched against the stored PEKS indexes using the Test() function. When matching records are identified, the corresponding encrypted documents are retrieved from the database and sent to the user. The authorized user then decrypts the files locally and accesses the required information. This workflow ensures that documents remain encrypted during storage and retrieval, while search queries are also protected, thereby preserving privacy and confidentiality throughout the entire process.

Methodology

The Privacy-Preserving Searchable Encryption (PPSE) system is developed to provide a secure and efficient method for storing and retrieving encrypted information without exposing confidential data. The methodology follows a structured workflow that combines user authentication, document encryption, secure keyword indexing, trapdoor generation, and privacy-preserving search operations. The system begins with user interaction through a web-based frontend developed using React.js, where users can perform operations such as registration, login, document upload, and keyword search. To ensure secure access, user authentication is managed using JSON Web Tokens (JWT), allowing only authorized users to access protected resources.

When a user uploads a document, the file is first encrypted using the AES-256 algorithm to preserve confidentiality before storage or transmission. Simultaneously, keywords associated with the document are processed and transformed into encrypted searchable indexes using a Public Key Encryption with Keyword Search (PEKS)-inspired mechanism. Both the encrypted document and encrypted keyword indexes are then transmitted to the backend server, which is implemented using Node.js and Express.js. These encrypted components are securely stored in a MySQL database without revealing plaintext document contents or original keywords.

During the retrieval process, the user enters a search keyword through the interface. Instead of sending the plaintext keyword to the server, the system generates a cryptographic trapdoor using authorized credentials. This trapdoor is submitted to the backend, where the server performs a secure comparison using the Test() function against stored encrypted keyword indexes. If matching records are found, the corresponding encrypted documents are retrieved from the database and returned to the user. To maintain end-to-end confidentiality, decryption is performed only at the client side using the appropriate AES key. As a result, the server never gains access to the original file contents or user search terms. Communication between frontend, backend, and database components is secured through standard HTTP/HTTPS protocols to preserve integrity and privacy. The integration of cryptographic methods, modular software components, and secure storage mechanisms creates a scalable architecture suitable for real-world privacy-sensitive environments such as healthcare systems, enterprise repositories, and legal document management platforms.

Modules

The Privacy-Preserving Searchable Encryption (PPSE) system is organized into multiple functional modules, each responsible for a specific task related to security, storage, retrieval, and system management. The modular design improves maintainability, scalability, and overall performance while ensuring secure coordination among all components.

User Authentication Module

The User Authentication Module is responsible for verifying user identity and controlling access to the system. New users can register by providing details such as name, email address, and password. Passwords are securely processed and stored in the database. Registered users can log in using valid credentials, after which a JWT token is generated to establish a secure session. Only authenticated users are permitted to upload files, perform searches, and retrieve documents.

Document Encryption Module

The Document Encryption Module secures files before they are stored in the system. When a user uploads a document, the file is encrypted using the AES-256 algorithm. This encryption process ensures that sensitive information remains confidential during transmission and storage. Only encrypted versions of the files are stored in the database, preventing unauthorized access to original content.

Keyword Encryption (PEKS) Module

The Keyword Encryption Module enables searchable encryption by transforming user-provided keywords into secure encrypted indexes. Keywords related to uploaded documents are processed and encrypted using a PEKS-based public key cryptographic technique. These encrypted keywords are stored alongside the encrypted files and later used for privacy-preserving search operations.

Trapdoor Generation Module

The Trapdoor Generation Module is responsible for protecting search queries. When a user enters a keyword, the system generates a secure trapdoor using authorized cryptographic credentials. The trapdoor acts as a protected representation of the keyword and is sent to the server instead of the plaintext search term. This prevents disclosure of user queries during search operations.

Implementation

The implementation of the Privacy-Preserving Searchable Encryption (PPSE) system is carried out as a secure full-stack web application that enables encrypted document storage and privacy-preserving keyword retrieval. The system integrates modern web technologies with cryptographic techniques to provide confidentiality, efficiency, and user-friendly operation. The implementation consists of frontend development, backend services, database management, secure keyword indexing, retrieval mechanisms, and authentication controls.

The frontend layer is developed using React.js, which provides a responsive and interactive user interface for users. It includes pages such as registration, login, dashboard, document upload, and keyword search. Through this interface, users

can upload files, submit search queries, and access retrieved documents. Client-side operations such as initiating AES encryption and generating trapdoors are also integrated into the frontend. Communication between the frontend and backend is performed using HTTP requests through Axios or the Fetch API.

The backend layer is implemented using Node.js with Express.js and is responsible for managing core application functions. It provides RESTful APIs for user authentication, document upload, encrypted search, and file retrieval. The backend validates requests, processes encrypted data, performs secure keyword matching, and coordinates communication between the frontend and database layers. It also executes the Test() function to compare encrypted indexes with search trapdoors during retrieval.

The system uses MySQL as the database management platform for storing encrypted data securely. The database stores encrypted files, PEKS ciphertext keywords, user credentials, and metadata. No plaintext documents or original keywords are stored in the database, thereby ensuring complete confidentiality. Proper schema design is adopted to improve storage efficiency and search performance.

The keyword encryption component is based on a PEKS-inspired cryptographic method. During document upload, user-provided keywords are transformed into secure ciphertext using a public key and stored as searchable indexes. During the search process, the user enters a keyword that is converted into a trapdoor using the corresponding private key. The trapdoor is sent to the server instead of the plaintext keyword. The backend compares the trapdoor with stored encrypted indexes using the Test() function and identifies matching records without revealing sensitive information.

Once matching files are found, the encrypted.

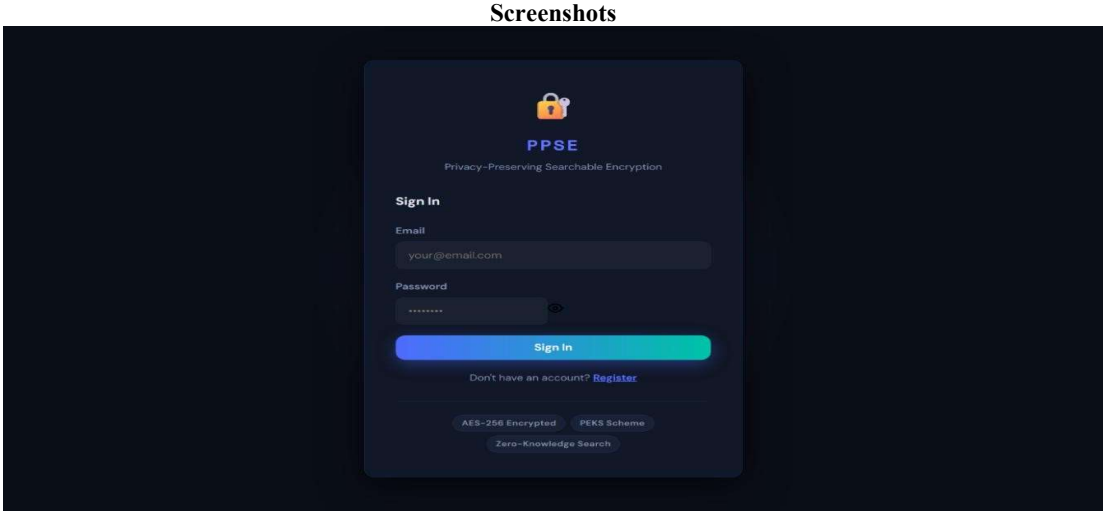
Pseudo Code

The upload process begins when a user selects a file and provides an associated keyword. The system first derives a public and private key pair for the user. A trapdoor is then generated using the private key and supplied keyword in order to check whether the same keyword already exists for previously stored files. Existing encrypted keyword indexes are retrieved from the database, and the Test() function is used to compare them with the generated trapdoor. If a duplicate match is found, the upload process is terminated. Otherwise, the keyword is encrypted using the PEKS algorithm, while both the file and keyword are encrypted using AES. The encrypted keyword, searchable ciphertext, and encrypted file are then stored securely in the database, and the upload is completed successfully.

The search process begins when the user enters a keyword into the search interface. The system derives the private key associated with the user and generates a trapdoor corresponding to the entered keyword. All encrypted keyword indexes belonging to the user are then retrieved from the database. The server uses the Test() function to compare each stored searchable ciphertext with the trapdoor. Whenever a match is identified, the corresponding encrypted document is selected. These matched files are then decrypted using AES and added to the result list. Finally, the system returns all matching documents to the authorized user.

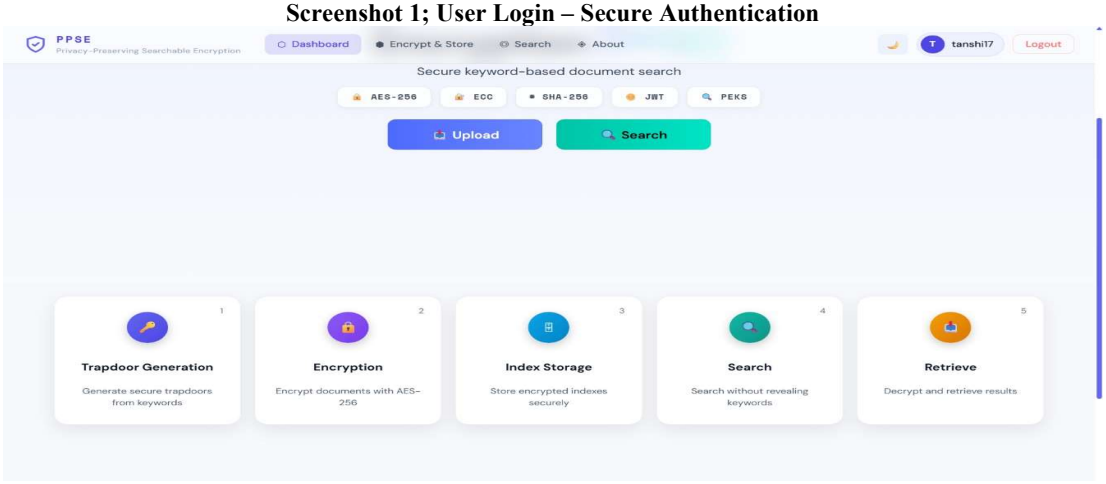
The PPSE system relies on several supporting cryptographic functions. The PEKS() function is used to encrypt keywords into searchable ciphertext. The Trapdoor() function generates secure search tokens from user keywords. The Test() function compares encrypted indexes with trapdoors to determine matches without exposing the original keyword. AES_Encrypt() is used to protect files and sensitive values before storage, while AES_Decrypt() restores the original data after retrieval.

Screenshots

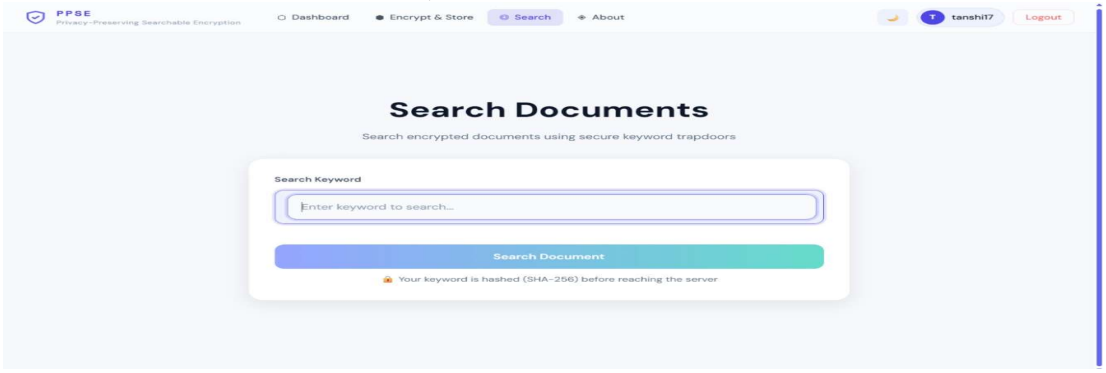


International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 2, 2026

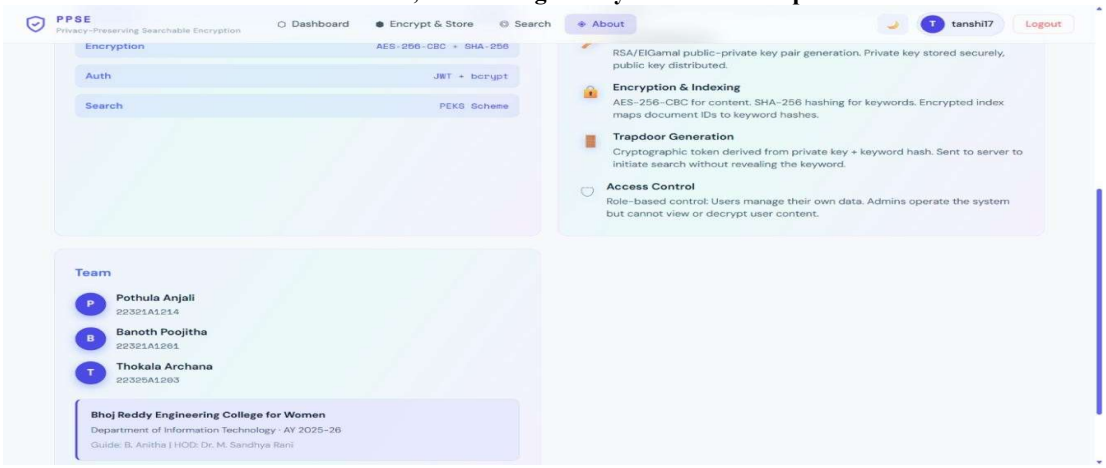
Screenshot 1; User Login – Secure Authentication



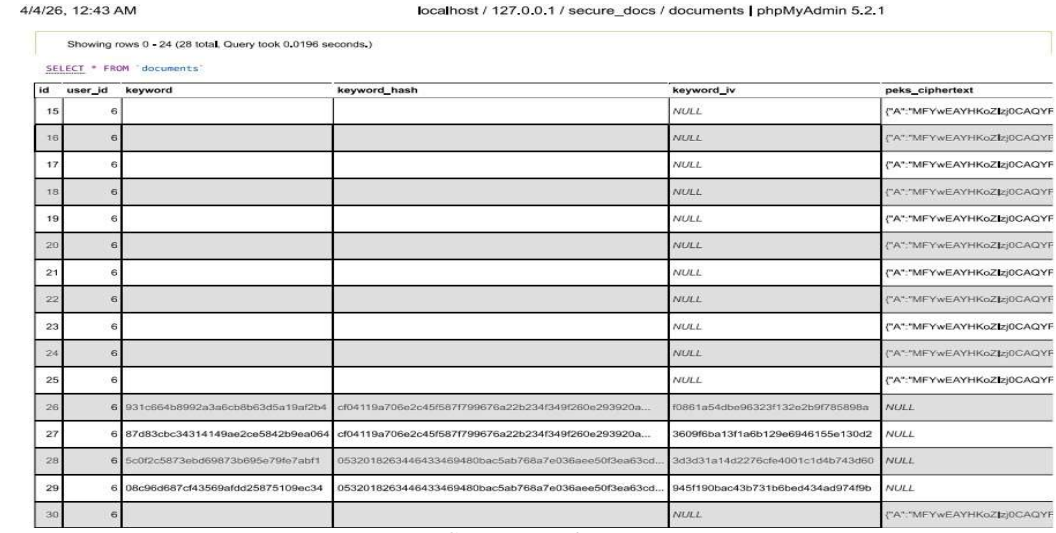
Screenshot2; Dashboard – User Home Interface



Screenshot 3; Search Page – Keyword Search Input



International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 2, 2026



Screenshot 4 Database

Conclusion

The project titled **Privacy-Preserving Searchable Encryption (PPSE) System** successfully demonstrates a secure and practical solution for storing and retrieving sensitive information in cloud-based environments. With the rapid growth of outsourced data storage, maintaining confidentiality while preserving usability has become a major challenge. Traditional encryption methods effectively protect stored data, but they often restrict direct search functionality over encrypted files. This project addresses that limitation by integrating searchable encryption techniques that allow secure keyword retrieval without revealing document contents or user queries.

The proposed system employs AES encryption to secure uploaded documents and a PEKS-inspired searchable encryption mechanism to protect associated keywords. Through the use of cryptographic trapdoors and the Test() function, the server can perform keyword matching over encrypted indexes without learning the original keyword. As a result, both stored data and search requests remain confidential, even when processed on untrusted servers. This significantly reduces privacy risks commonly associated with conventional cloud storage models.

The implementation of the system using React.js, Node.js, Express.js, and MySQL demonstrates that privacy-preserving encryption techniques can be effectively integrated into modern web applications. The platform supports secure user registration, authenticated access, encrypted document upload, confidential keyword search, and controlled retrieval of files. Its modular architecture improves maintainability, scalability, and real-world usability in multi-user environments.

The project also highlights the practical importance of searchable encryption in domains where confidentiality is critical, such as healthcare systems, legal organizations, financial institutions, educational repositories, and enterprise document management. By preventing unauthorized disclosure while maintaining efficient retrieval, the PPSE system offers a strong balance between security and functionality.

Future Scope

The Privacy-Preserving Searchable Encryption (PPSE) system can be extended in several directions to improve functionality, scalability, and security. Future enhancements may include support for multi-keyword search, allowing users to retrieve documents based on multiple terms simultaneously. Ranked search techniques can also be introduced so that search results are presented according to relevance, improving user experience and retrieval efficiency.

Another promising enhancement is the integration of fuzzy search mechanisms, which would allow the system to handle spelling mistakes, partial matches, and keyword variations. This would make the platform more flexible and user-friendly, especially for large datasets and non-technical users.

Advanced cryptographic methods such as Attribute-Based Encryption (ABE), Functional Encryption, and Homomorphic Encryption can be incorporated to provide fine-grained access control and secure computations over encrypted data. These techniques would strengthen privacy guarantees while enabling more advanced operations without exposing sensitive information.

The system can also be deployed on public or hybrid cloud platforms to improve scalability and support large volumes of encrypted documents. Enhanced key management solutions, automated key rotation, secure backup recovery, and distributed trust models can further improve system reliability and security.

Machine learning and intelligent indexing methods may be applied to optimize search speed, predict

user behavior, and improve retrieval accuracy while preserving privacy. Additionally, extending the application to mobile platforms would increase accessibility and allow secure document management from smartphones and tablets.

References

[1] Boneh, D., Di Crescenzo, G., Ostrovsky, R., & Persiano, G. (2004). *Public Key Encryption with Keyword Search*. EUROCRYPT.
[2] Song, D. X., Wagner, D., & Perrig, A. (2000). *Practical Techniques for Searches on Encrypted Data*. IEEE Symposium on Security and Privacy.
[3] Curtmola, R., Garay, J., Kamara, S., & Ostrovsky, R. (2006). *Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions*. ACM CCS.
[4] Goh, E. J. (2003). *Secure Indexes*. IACR Cryptology ePrint Archive.
[5] Cash, D., Jarecki, S., Jutla, C., Krawczyk, H., Rosu, M., & Steiner, M. (2013). *Highly-Scalable Searchable Symmetric Encryption with Support for Boolean Queries*. CRYPTO.
[6] Kamara, S., & Papamanthou, C. (2013). *Parallel and Dynamic Searchable Symmetric Encryption*. Financial Cryptography Conference.
[7] Chase, M., & Kamara, S. (2010). *Structured Encryption and Controlled Disclosure*. ASIACRYPT.
[8] Node.js Documentation. (2026). *Official Documentation*. https://nodejs.org
[9] React Documentation. (2026). *Official Documentation*. https://react.dev
[10] MySQL Documentation. (2026). *Reference Manual*. https://dev.mysql.com/doc
[11] Mozilla Developer Network (MDN). (2026). *Web Security Concepts*. https://developer.mozilla.org
[12] Stallings, W. (2017). *Cryptography and Network Security: Principles and Practice*. Pearson.