CHAPTER 1

INTRODUCTION

Cloud storage offers a cost-effective way to store big data and deal with the rapid growth of data variety, volume, and velocity. Although cloud storage has become entrenched in our digitalized society, there are challenges that have yet to be resolved. For example, both the cloud server and data owners are typically not in the same trusted domain, and hence security and privacy risks remain issues that need to be considered when utilizing cloud storage solutions. Some solutions have been presented for different security risks. For example, to ensure data utilization over encrypted cloud data, one typical solution is to utilize searchable encryption (SE), which enables the cloud to retrieve encrypted data based on a set of keywords on behalf of data users. So we introduce a basic Verifiable Searchable Encryption Framework (VSEF) against insider KGA, by extending the public auditing technique to SE scheme. In the basic VSEF, the costly correctness verification tasks are assigned to a fully-trusted third-party auditor, and in turn, the auditor honestly reports the auditing results to cloud clients. We further enhance the basic VSEF to support other features such as multi-keyword search, multi-key encryption, dynamic update (e.g., file modification, file insertion, file deletion), so that the enhanced VSEF can be widely applied in practice.

1.1 CONTRIBUTION

1.1.1 VERIFIABLE KEYWORD SEARCH:

Different from traditional private verification solutions using Merkle hash tree, hash chain or accumulator, the basic (or enhanced) VSEF allows the third-party auditor to test the correctness of search results, which prevents the malicious cloud server from forging, tampering or discarding stored cloud data. Furthermore, it significantly reduces the workload of cloud clients, relating to result verification.

1.1.2 RESISTING INSIDER KGA:

Our proposed schemes focus on the insider KGA rather than the external KGA which can be solved by building secure channel or delegating designated test. In the basic (or enhanced) VSEF, the insider cloud server is not capable of producing the valid keyword ciphertext or trapdoor without the secret key of data owner or data user, respectively. As the malicious cloud cannot encrypt each candidate keyword and test keyword ciphertext (or index) with the submitted trapdoor, it cannot launch the insider KGA with public keys.

1.1.3 MULTI-KEYWORD SEARCH:

The single keyword search in the basic VSEF produces many irrelevant search results, which incurs unnecessary wastage of storage and computation resources for data users. Hence, the enhanced VSEF is designed to allow a data user to submit multiple keywords at one time, which also significantly avoids time delay in real-time scenarios with highly efficient retrieval.

1.1.4 MULTI-KEY ENCRYPTION:

Different from the traditional multikey encryption scenarios in which data users are required to submit multiple trapdoors in the ciphertext retrieval process

and interact multiple rounds with the data owner. The enhanced VSEF enables a data user to submit a single trapdoor but allows the cloud server to retrieve the records encrypted by different keys. This feature not only avoids multiple interactions between the data owner and data user but also has a constant overhead of obtaining the decryption key for each search result.

1.1.5 DYNAMIC UPDATE:

Apart from supporting static data, the enhanced VSEF also provides dynamic data operations (e.g., modification, insertion and deletion). Besides, the enhanced VSEF achieves dynamic result verification by introducing an index switcher, which stores the map between record indices and tag indices. In comparison with tree-based index structures (e.g., Merkle hash tree or accumulation tree needs to update additional nodes.), our enhanced VSEF decreases the costs of ciphertext update.

1.2 SYSTEM MODEL

As illustrated in Fig. 1.1, we consider a scenario in which cloud clients would like to share some sensitive records with authorized entities by employing public cloud storage services. The system model3 of basic VSEF mainly consists of four entities, namely Trusted Authority (TA), Data Owner (DO), Cloud Server (CS), Data User (DU), Third-Party Auditor (TPA). The TA is in charge of system initialization involving the generation and distribution of public parameters and secret keys. The DO who possesses a large amount of records shifts the local data storage and computation tasks to CS. Besides, through delegating auditing tasks to TPA, the high result verification overhead on DO can be further offloaded. The CS, which has adequate storage and computation capabilities, is responsible for storing and managing stored records (e.g., searching ciphertexts, responding to result verification). The TPA with expertise and capabilities verifies whether the search results are accurate or not. The

authorized DU can retrieve encrypted records of interest. Next, we show the high-level representation of the system model as follows.

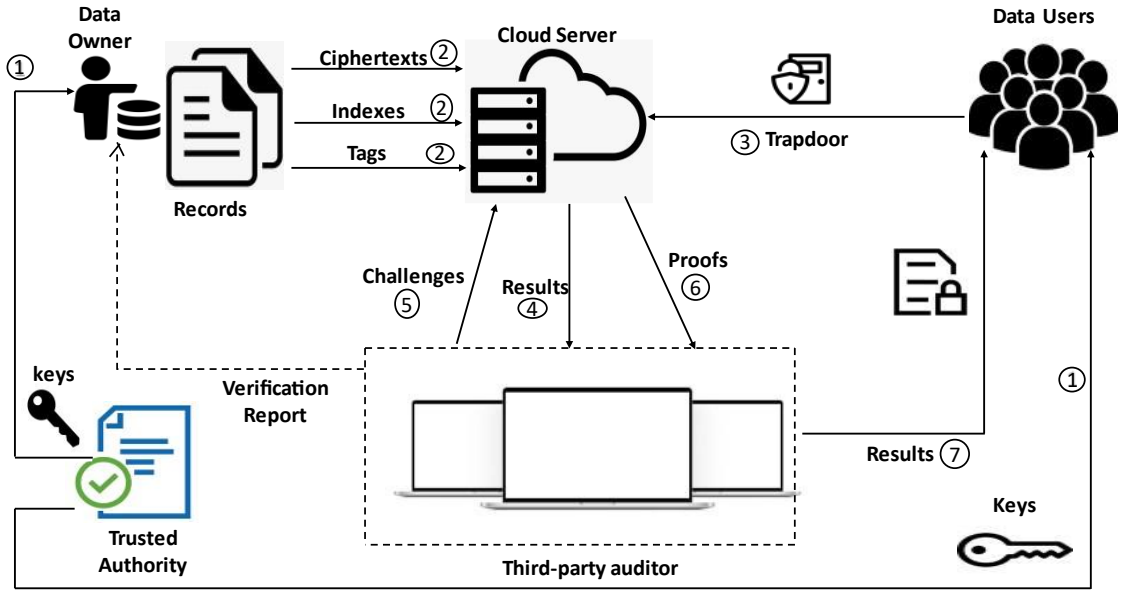

Fig 1 .1 The System model of VSEF

In the system initialization phase, the TA generates the public/secret key pairs for both DO and DUs (step 1). Before outsourcing data to CS, the DO first extracts keywords from records, then encrypts the original records and keywords to output record ciphertexts and indexes, respectively, note that each record is also labelled with a tag. Finally, the DO uploads the indexes, record ciphertexts along with associated tags to CS (step 2). When the DU intends to seek the encrypted records containing a particular keyword, he/she first outputs a trapdoor based on his secret key and then sends it to CS (step 3). The CS returns the matched results if the trapdoor satisfies the indexes (step 4). However, the CS may return a fraction of incorrect search results for various incentives (e.g., saving storage space, hiding data loss accidents). Thus, the TA employs the result verification mechanism on behalf of DO to check whether the search results are accurate by sending challenging information to CS (step 5). It requires

that the CS should response with proof information (step 6). If the proof information passes the result verification mechanism, the TPA forwards the results to DU (step 7); otherwise, it outputs “Null”. Note that the TPA also passes the verification report to DO, which is shown by the dashed line in Fig. 1.1.

1.3 SECURITY MODEL

Different from the PEKS scheme [5], the adversary A in the basic (or enhanced) VSEF can access not only the trapdoor generation oracle but also the index generation oracle in Game 1 and Game 2, respectively. Hence, our proposed VSEF should guarantee that A cannot distinguish the trapdoors and indexes in the following two games performed between A and a challenger C.

- The process of Game 1, which aims to guarantee the trapdoor privacy.
- The process of Game 2, which protects the index indistinguishability.

1.4 OBJECTIVES

The main objective of this project is to implement a Verifiable Searchable Encryption Framework. It is used to resist the insider Keyword Guessing Attack which is a scenario where the insider employee or the contractor removes or steals information from the organization for personal financial or other form of gain.

1.5 SCOPE

Cloud storage has become entrenched in our digitalized society, there are challenges that have yet to be resolved. The main scope of the project is to resist insider Keyword Guessing Attack and it reduces the security and privacy risks to the data owners.

CHAPTER 2
LITERATURE SURVEY

2.1 A search optimized blockchain-based verifiable searchable symmetric encryption framework.

AUTHOR: Mu Han , Shuai Wu (2022).

Outsourcing storage and computation to cloud servers have become a trend. Although searchable symmetric encryption (SSE) had handled the data privacy issue caused by honest-but-curious servers, a semi-honest server may return incomplete or incorrect results when users search for their encrypted data. To against such servers, scholars have recently used blockchain/Ethereum-based SSE schemes which utilize the public, that is, active nodes, to verify the search process. However, the search operation in existing schemes is very expansive in terms of fee and time. In this paper, we propose a new blockchain-based searchable encryption framework with search optimized, that is, free of charge, quicker, and more private, at the cost of some extra storage. Besides, we design a general and efficient verification algorithm for our framework, which makes the search verifiable. In addition, we deploy an instance of our framework on an official Ethereum test network, and the experimental results and evaluations demonstrate the advantage of our framework.

2.2 Verifiable Attribute-Based Keyword Search Scheme over Encrypted Data for Personal Health Records in Cloud.

AUTHOR: Yuqin Sun, Lidong Han, Jingguo Bi , Xiao Tan and Xie Qi (2021).

Personal health record (PHR) is a medical model in which patients can upload their medical records and define the access control by themselves. Since the limited local storage and the development of cloud computing, many PHR services have been outsourced to the cloud. In order to ensure the privacy of electronic medical records, patients intend to encrypt their health records before uploading them. However, encrypted PHR can not be accessed directly and not be retrieved by the legitimate users. To solve these issues, in this article we propose a new searchable encryption scheme with ciphertext-policy attributes, which achieves fine-grained access control and exact keyword search over encrypted PHRs. Moreover, in our proposed scheme, the receiver can verify the integrity of search result which the cloud server returns. Finally, we simulate our scheme, and the experiments show that our scheme has high practicability for cloud-based healthcare systems and has high efficiency in aspects of keyword search and results verification.

2.3 A Secure Searchable Encryption Framework for Privacy-Critical Cloud Storage Services

AUTHOR : Thang Hoang, Attila A.Yavuz and Jorge Guajardo (2019).

Searchable encryption has received a significant attention from the research community with various constructions being proposed, each achieving asymptotically optimal complexity for specific metrics (e.g., search, update). Despite their elegance, the recent attacks and deployment efforts have shown that the optimal asymptotic complexity might not always imply practical performance, especially if the application demands a high privacy. In this article, we introduce a novel Dynamic Searchable Symmetric Encryption (DSSE) framework called Incidence Matrix (IM)-DSSE, which achieves a high level of privacy, efficient search/update, and low client storage with actual deployments on real cloud settings. We harness an incidence matrix along with two hash tables to create an encrypted index, on which both search and update operations can be performed effectively with minimal information leakage. This simple set of data structures surprisingly offers a high level of DSSE security while achieving practical performance. Specifically, IM-DSSE achieves forward-privacy, backward-privacy and size-obliviousness simultaneously. We also create several DSSE variants, each offering different trade-offs that are suitable for different cloud applications and infrastructures. We fully implemented our framework and evaluated its performance on a real cloud system (Amazon EC2). We have released IM-DSSE as an open-source library for wide development and adaptation.

2.4 Enabling verifiable multiple keywords search over encrypted cloud data.

AUTHOR : Yinbin Miao, Jian Weng , Ximeng Liu , Kim- Kwang Raymond Choo, Zhiquan Liu , Hongwei Li (2018).

Searchable Encryption (SE) enables a user to search over encrypted data, such as data stored in a remote cloud server. Existing certificate-, identity-, and attribute-based SE schemes suffer from certificate management or key escrow limitations. Furthermore, the semi-honest-but-curious cloud may conduct partial search operations and return a fraction of the search results (i.e., incomplete results) in order to reduce costs. In this paper, we present a secure cryptographic primitive, Verifiable Multiple Keywords Search (VMKS) over ciphertexts, which leverages the Identity-Based Encryption (IBE) and certificateless signature techniques. The VMKS scheme allows the user to verify the correctness of search results and avoids both certificate management or key escrow limitations. We then demonstrate the security of proposed VMKS scheme (i.e., the scheme achieves both ciphertext indistinguishability and signature unforgeability). We also use a real-world dataset to evaluate its feasibility and efficiency.

2.5 An Efficient Public-Key Searchable Encryption Scheme Secure against Inside Keyword Guessing Attacks.

AUTHOR : Qiong Huang, Hongbo Li (2017).

With the rapid development of cloud computing technology, a large amount of data now has been stored onto the cloud. Since the data owner loses its control of the data, several security and privacy issues arise in cloud storage service, among which data privacy is a very sensitive problem. Encryption is an effective way to protect the data from being leaked. However, traditional search mechanisms do not work for encrypted data. How to efficiently search over encrypted data thus becomes an important and interesting problem, and has attracted many researchers' attention [7, 23, 42, 38, 31, 16, 34, 21, 17, 15, 32, 40]. In 2004, Boneh et al. introduced the notion of Public Key Encryption with Keyword Search (PEKS), integrating keyword search functionality into public key encryption. They proposed the first PEKS scheme (denoted by BDOP-PEKS hereafter) [7] and showed that it is secure based on the bilinear Diffie-Hellman assumption in the random oracle model. The framework of PEKS is illustrated in Figure 1. There are three parties involved, a data sender called Alice, a data receiver called Bob and a cloud server. Alice has a bunch of sensitive documents {Fi} to share with his friend Bob. First, Alice extracts keywords {wi,j} from each document Fi , and encrypts the keywords using the PEKS scheme. Besides, Alice encrypts each document with a (possibly another) encryption scheme. Let the ciphertexts be {Cwi,j } and {Ci}, respectively. Alice uploads all the ciphertexts onto the cloud server. To search over the encrypted documents whether there is any one containing some keyword w, Bob computes a trapdoor Tw for w using his secret key, and gives it to the cloud server (via a secure channel).

CHAPTER 3
SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

In most existing system, verifiable keyword search schemes are susceptible to keyword-guessing attacks (KGA). This is because the keywords are always restricted to low-entropy keyword space, which makes it possible to infer what keywords are included in the stored records by exhaustively guessing the candidate keywords and to further obtain the sensitive information in these records. When an external attacker obtains the trapdoors, he/she can launch the external KGA. And also the existing system arises some risks such as those due to hardware failures, software bugs, network attacks, and so on, yet remain. As data owners lose direct physical control over their data, the malicious cloud may discard rarely or never accessed data, or even conceal data loss incidents. In other words, the malicious cloud yields a fraction of inaccurate search results.

3.2 DISADVANTAGES

- In the existing system, the VSEF supports Insider keyword attack.
- It provides inaccurate search results.
- To the best of our knowledge, there is no single DSSE scheme that outperforms all the other alternatives in terms of all the aforementioned metrics: privacy (e.g., information leakage), performance (e.g., search, update delay), storage efficiency and functionality.
- Most of the existing system only provide a theoretical asymptotic analysis and, in some cases, merely a prototype implementation.

3.3 PROPOSED SYSTEM

To overcome the disadvantages of existing system, we first introduce a basic Verifiable Searchable Encryption Framework (VSEF) against insider KGA, by extending the public auditing technique to SE scheme. In the basic VSEF, the costly correctness verification tasks are assigned to a fully-trusted third-party auditor, and in turn, the auditor honestly reports the auditing results to cloud clients. Thus, we further enhance the basic VSEF to support other features such as multi-keyword search, multi-key encryption, dynamic update (e.g., file modification, file insertion, file deletion), so that the enhanced VSEF can be widely applied in practice.

3.4 ADVANTAGES

The main advantages of the enhanced VSEF are,

- Verifiable keyword search
- Resisting insider KGA
- Multi-keyword search
- Multi-key encryption
- Dynamic update.

CHAPTER 4

SYSTEM REQUIREMENTS

4.1 HARDWARE REQUIREMENTS

- Processor : Intel core processor 2.60 GHZ
- RAM : 8 GB
- Hard disk : 512 GB
- Compact Disk : 650 Mb
- Keyboard : Standard keyboard
- Monitor : 15 inch color monitor

4.2 SOFTWARE REQUIREMENTS

• Operating system: Windows OS

• Software: C #

• Visual studio(Community version 2018)

• Mongo database(version 5.0.2)

• FTP cloud.

CHAPTER 5
SOFTWARE DESCRIPTION

5.1 SOFTWARE:C#

C# (pronounced "C sharp") is a general-purpose, object-oriented programming language that was developed by Microsoft as part of the .NET framework. C# is designed to be simple, modern, and easy to use, while also providing powerful features for developers. C# is similar in syntax to other C-style languages, such as C++ and Java, and is often used to develop Windows desktop applications, web applications, and games. It is a statically typed language, meaning that variables must be declared with a specific data type before they can be used.

Some of the key features of C# include garbage collection, automatic memory management, and support for object-oriented programming concepts such as classes, interfaces, and inheritance. C# also includes advanced features such as LINQ (Language-Integrated Query) for data querying and manipulation, and asynchronous programming support for creating responsive and scalable applications. C# is widely used in enterprise software development, and is also used by many game developers due to its performance and ease of use. C# code is compiled into Microsoft Intermediate Language (MSIL), which can then be executed on the .NET runtime environment.

One of the key features of C# is the .NET Framework, which provides a set of libraries and tools for developing Windows applications, web applications, and services. C# can be used with a variety of development tools, including Microsoft Visual Studio, Visual Studio Code, and others.The latest version of C# was released in 2021.

5.1.1 FEATURES OF C#

C# is a modern, powerful, and versatile programming language that has a wide range of features. Here are some of the key features of C#:

1. Object-Oriented Programming: C# is an object-oriented language, which means that it allows developers to create classes and objects to represent real-world entities and interact with each other.
2. Cross-Platform: C# can be used to develop applications for a variety of platforms, including Windows, Linux, macOS, Android, and iOS, using the .NET framework or .NET Core.
3. Type-Safe: C# is a strongly-typed language, which means that all variables and data types must be explicitly defined before they can be used. This helps to prevent common programming errors such as type mismatches and null reference errors.
4. Memory Management: C# has a garbage collector that automatically manages memory allocation and deallocation, making it easier for developers to write code without having to worry about memory management.
5. Exception Handling: C# has a robust exception handling mechanism that allows developers to handle runtime errors and prevent application crashes.
6. LINQ: C# includes Language-Integrated Query (LINQ), which provides a powerful and concise syntax for querying data from various sources such as arrays, collections, and databases.
7. Asynchronous Programming: C# has built-in support for asynchronous programming, which allows developers to write code that can run concurrently without blocking the main thread.

8. Delegates and Events: C# supports delegates and events, which are powerful mechanisms for creating and handling custom events and call backs.
9. Attributes: C# includes attributes, which are metadata tags that can be added to code elements to provide additional information or functionality.

Overall, C# is a versatile and powerful language that offers a wide range of features for developers to build modern, high-performance applications for various platforms.

5.1.2 USES OF C#

- Desktop Application Development: C# is commonly used for developing Windows desktop applications using the .NET framework or .NET Core. It provides a rich set of libraries and tools for building modern and responsive user interfaces.
- Web Application Development: C# can be used to build web applications and services using ASP.NET or ASP.NET Core. C# is well-suited for building server-side applications and provides a robust and secure environment for web development.
- Game Development: C# is widely used in the game development industry, particularly for building games using the Unity game engine. C# provides a powerful and easy-to-use programming language for creating games that run on a variety of platforms.
- Mobile Application Development: C# can be used to build mobile applications for Android and iOS platforms using Xamarin, a popular mobile app development platform that allows developers to write native apps using C#.

- Machine Learning and Data Science: C# can be used for building machine learning models and data analysis using libraries such as ML.NET, which provides a powerful and easy-to-use set of APIs for building custom ML models and analyzing data.
- Internet of Things (IoT): C# can be used for building IoT applications and devices using platforms such as .NET IoT Core and Azure IoT. C# provides a robust and secure environment for building and deploying IoT solutions.

5.2 MONGO DATABASE

MongoDB is a popular NoSQL database that is designed to handle large amounts of unstructured or semi-structured data. Unlike traditional SQL databases, MongoDB stores data in documents, which are similar to JSON objects. This allows developers to store complex data structures in a flexible and scalable manner.

MongoDB is highly scalable and provides features such as automatic sharding and horizontal scaling that allow it to handle large amounts of data and high traffic loads. It also provides high availability features such as automatic failover and replica sets that ensure that data is always available and accessible.

MongoDB provides a number of features that make it a flexible and powerful database solution, including indexing, a powerful aggregation framework, ad-hoc queries, and support for multiple programming languages. It is widely used in a variety of applications and industries, including e-commerce, social media, gaming, and more.

5.2.1 FEATURES OF MONGO DATABASE

MongoDB is a popular NoSQL database that provides a number of features that make it a flexible and powerful database solution. Here are some of the key features of MongoDB:

1. Document-oriented: MongoDB stores data in documents, which are similar to JSON objects. This allows developers to store complex data structures in a flexible and scalable manner.
2. Schema-less: MongoDB is schema-less, which means that documents in a collection can have different structures and fields. This makes it easy to change the data model as requirements change.
3. Indexing: MongoDB supports multiple indexing options, including text, geospatial, and hashed indexing, which makes it easy to query and analyze data efficiently.
4. High performance: MongoDB is designed for high performance, with features such as in-memory computing and automatic sharding that allow it to handle large amounts of data and high traffic loads.
5. High availability: MongoDB provides high availability features such as automatic failover and replica sets that ensure that data is always available and accessible.
6. Aggregation framework: MongoDB provides a powerful aggregation framework that allows developers to perform complex data analysis and transformations on the data.
7. Horizontal scalability: MongoDB is highly scalable, with automatic sharding and horizontal scaling features that allow it to handle large amounts of data and high traffic loads.
8. Ad-hoc queries: MongoDB supports ad-hoc queries, which means that developers can query data without having to pre-define how the data is structured.

9. Flexibility: MongoDB provides a lot of flexibility for developers to customize the database to their specific needs, such as using a different data model or defining custom indexes.
10. Support for multiple programming languages: MongoDB provides drivers for a wide range of programming languages, including Java, Python, Node.js, and more.

Overall, MongoDB is a flexible and powerful database solution that provides many features that make it a popular choice for a variety of applications and industries.

5.2.2 USES OF MONGO DATABASE

1. Content Management Systems: MongoDB can be used to store and manage content for websites and other digital media, such as text, images, and videos.
2. E-commerce Applications: MongoDB can be used to store product catalog data, customer information, and order details for e-commerce applications.
3. Social Media Applications: MongoDB can be used to store user profiles, social network data, and messaging data for social media applications.
4. Gaming Applications: MongoDB can be used to store game state data, user information, and other game-related data for gaming applications.
5. Internet of Things (IoT) Applications: MongoDB can be used to store and analyze data from sensors and other IoT devices.
6. Big Data Analytics: MongoDB can be used as a data store for big data analytics platforms, providing a flexible and scalable way to store and analyze large amounts of data.
7. Mobile Applications: MongoDB can be used as a backend data store for mobile applications, providing a scalable and flexible solution for storing user data and other application data.

5.3 FTP CLOUD

FTP (File Transfer Protocol) is a standard network protocol used for transferring files between a client and a server over the internet or a network. It is often used for transferring large files, such as video, audio, and software.FTP cloud refers to a file transfer service that uses the FTP protocol and is hosted on a cloud-based server. This means that users can access and transfer files from any device that is connected to the internet, without having to install any software or hardware.

FTP cloud services typically offer a range of features, such as user authentication, encryption, file sharing, and collaboration tools, making it easy for users to share and collaborate on files securely. They can also provide scalable storage options and automatic backup and recovery services, ensuring that files are always available and protected .FTP cloud services can be used by individuals and businesses for a wide range of purposes, such as file sharing, remote file access, data backup and recovery, and content distribution. They are particularly useful for businesses that need to share large files with clients or partners or need to access files remotely from different locations.

5.3.1 FEATURES OF FTP CLOUD

FTP cloud services provide a range of features that make it easy to transfer, store, and manage files securely. Here are some of the key features of FTP cloud:

1. File transfer: FTP cloud allows users to transfer files of any size securely between a client and a server using the FTP protocol.
2. Scalable storage: FTP cloud services typically offer scalable storage options, allowing users to store large amounts of data and expand storage capacity as needed.

3. Secure data transfer: FTP cloud services use encryption and user authentication to ensure that files are transferred securely and are only accessible by authorized users.
4. Collaboration tools: FTP cloud services often include collaboration tools, such as file sharing and team folders, making it easy for users to work together on files.
5. Remote file access: FTP cloud allows users to access files from anywhere with an internet connection, making it easy to work remotely or on-the-go.
6. Automatic backup and recovery: FTP cloud services often include automatic backup and recovery features, ensuring that files are always protected and can be easily restored in case of data loss.
7. Integration with other tools: FTP cloud services can be integrated with other tools, such as content management systems, project management software, and workflow tools, making it easy to manage files and collaborate with team members.

Overall, FTP cloud services provide a range of features that make it easy to transfer, store, and manage files securely, making them a popular choice for businesses and individuals who need to share and collaborate on files.

5.3.2 USES OF FTP CLOUD

FTP (File Transfer Protocol) is a standard protocol used to transfer files over the internet. FTP cloud refers to the use of FTP to transfer files to or from cloud-based storage services. Here are some of the common uses of FTP cloud:

1. Website hosting: FTP cloud can be used to upload and manage website files on a web server. Many web hosting companies offer FTP access to their customers to make it easy to upload and manage their website files.

2. Backup and storage: FTP cloud can be used to backup and store files securely in the cloud. This provides an easy and secure way to store and access files from anywhere with an internet connection.
3. Collaboration: FTP cloud can be used for collaboration between team members working on a project. Files can be uploaded and downloaded by team members from a central location, making it easy to share files and collaborate on projects.
4. File sharing: FTP cloud can be used for sharing files with others. Files can be uploaded to a central location and shared with others using a link or by providing them with FTP access.
5. Automated file transfers: FTP cloud can be used to automate file transfers between different systems or applications. This can be useful for businesses that need to transfer large volumes of data between different systems or applications on a regular basis.

Overall, FTP cloud provides a convenient and secure way to transfer and manage files in the cloud, making it a useful tool for businesses and individuals alike.

CHAPTER 6
SYSTEM DESIGN

6.1 SYSTEM ARCHITECTURE

System architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system.

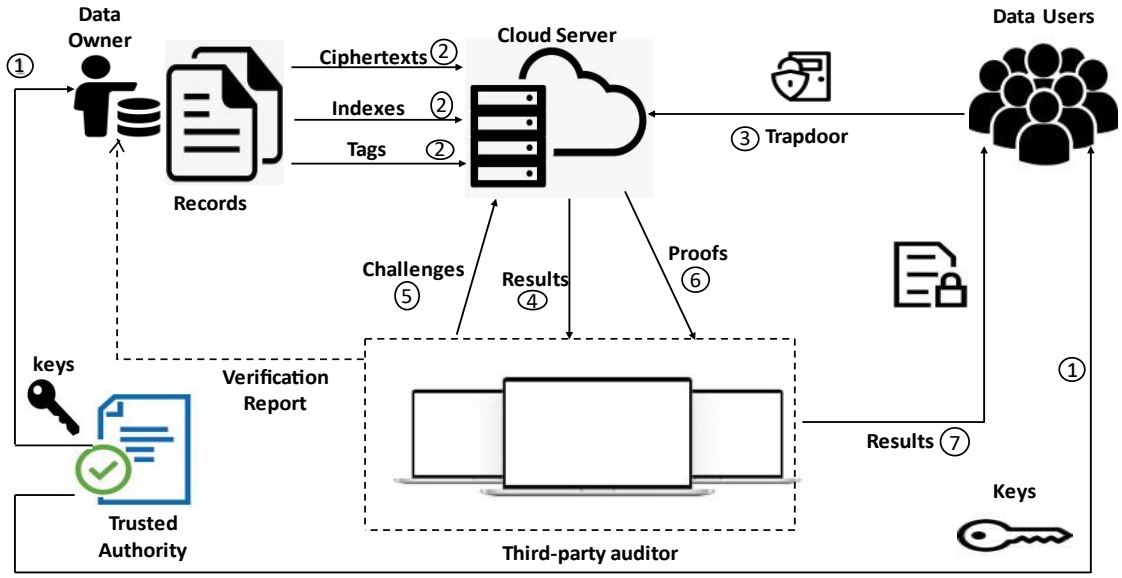

Fig 6.1 The System architecture of VSEF

6.2 SEQUENCE DIAGRAM

A sequence diagram is a type of interaction diagram in Unified Modeling Language (UML) that depicts the interactions between objects in a specific scenario or use case. It visualizes the flow of messages between objects and shows the sequence of actions that occur in the system.

In a sequence diagram, each object is represented as a vertical line, called a lifeline, which represents the lifespan of the object during the scenario. Messages are represented as arrows that indicate the direction of communication between objects. The order of messages on the diagram reflects the chronological order of events in the scenario.

Sequence diagrams can be used to model various scenarios, such as system behavior, business processes, and software architecture, to name a few. They are useful in analyzing the interactions between different parts of a system and can help identify potential problems and improve system performance.

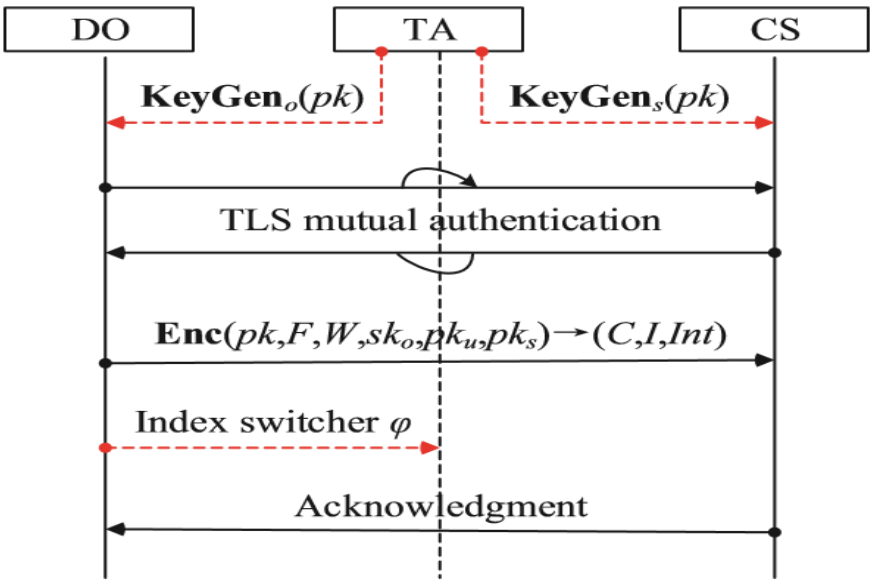

Fig 6.2 Data Uploading

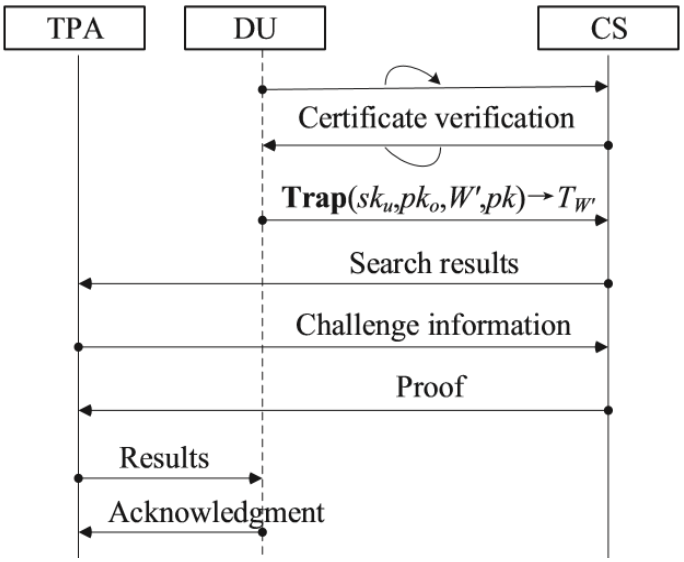

Fig 6.3 Result Verification

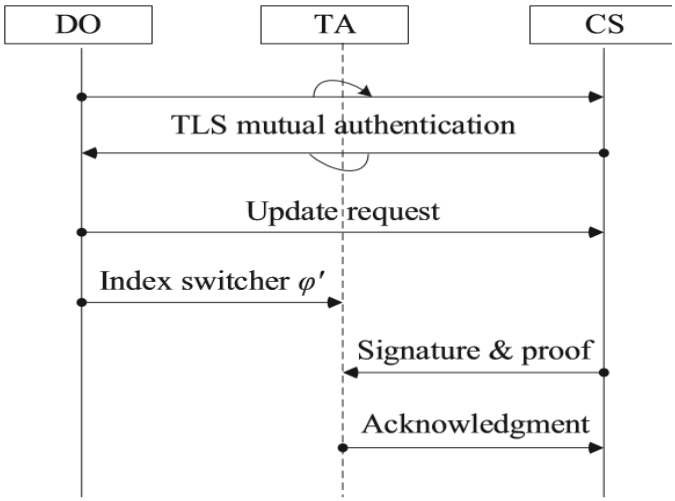

Fig 6.4 Data Update

CHAPTER 7

SYSTEM IMPLEMENTATION

In this project, we created a verifiable searchable encryption framework using ECC and SSL algorithm. It is used to reduce the insider Keyword Guessing Attacks. To start the system, first we have to register in VSEF webpage using username, password, user Id and phone number. After registration, we have to login into the system using username and password which we register. Once we logged in successfully, the dashboard page is opened. Then the data owner has certain data and upload the data in the form of ciphertext that is encrypted data into the cloud. Next, the trusted authority generates a key and distributes it to the data owner and data user. The generated key is valid only for 10 minutes. Whenever the data user needs the data, they have to generate a key and send the request to the cloud. The cloud verifies the user and allow them to access the data. Now, the third party auditor sends the decrypted data to data user and sends verification report to data owner.

7.1 ELLIPTICAL CURVE CRYPTOGRAPHY (ECC)

It is used to create a faster, smaller and more efficient cryptographic keys. It is a key-based technique for encrypting data. ECC focuses on pairs of public and private keys for decryption and encryption of web traffic. We used this algorithm to keys. It is as simple as securely generating a random integer in certain range, so it is extremely fast. Any number within the range is valid ECC private key.

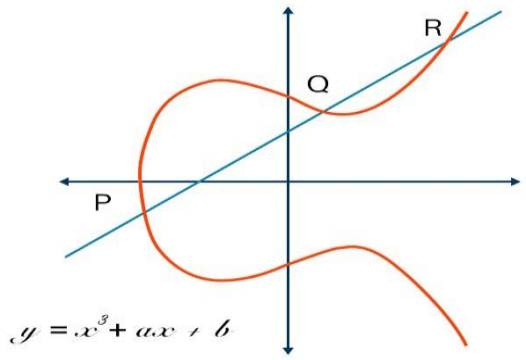

Fig 7.1 Simple elliptical curve

7.2 SECURE SOCKET LAYER (SSL)

It uses asymmetric cryptography to initiate communication which is known as SSL handshake. It protects the confidentiality and integrity of the data-in-transit.We used this SSL for certification to connect computer website to webserver.

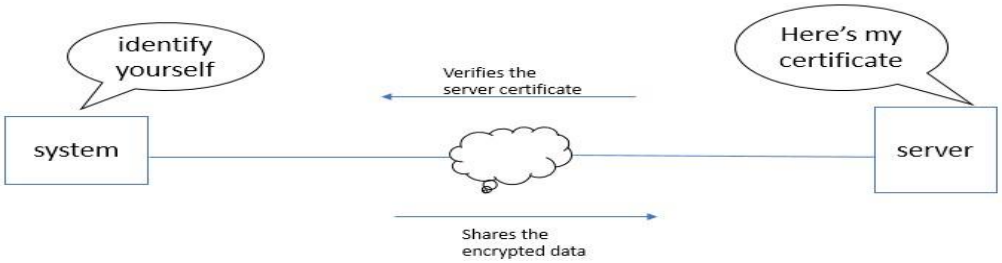

Fig 7.2 Secure Socket Layer (SSL)

CHAPTER 8
SYSTEM TESTING

Software testing is a method of assessing the functionality of a software program. There are many different types of software testing but the two main categories are dynamic testing and static testing. Dynamic testing is an assessment that is conducted while the program is executed; static testing, on the other hand, is an examination of the program's code and associated documentation. Dynamic and static methods are often used together.

Testing is a set activity that can be planned and conducted systematically. Testing begins at the module level and work towards the integration of entire computers based system. Nothing is complete without testing, as it is vital success of the system.

Testing Objectives:

There are several rules that can serve as testing objectives, they are

1. Testing is a process of executing a program with the intent of finding an error

2. A good test case is one that has high probability of finding an undiscovered error.

3. A successful test is one that uncovers an undiscovered error.

If testing is conducted successfully according to the objectives as stated above, it would uncover errors in the software. Also testing demonstrates that software functions appear to the working according to the specification, that performance requirements appear to have been met.

There are three ways to test a program

1. For Correctness
2. For Implementation efficiency
3. For Computational Complexity.

Tests for correctness are supposed to verify that a program does exactly what it was designed to do. This is much more difficult than it may at first appear, especially for large programs.

Tests used for implementation efficiency attempt to find ways to make a correct program faster or use less storage. It is a code-refining process, which re-examines the implementation phase of algorithm development. Tests for computational complexity amount to an experimental analysis of the complexity of an algorithm or an experimental comparison of two or more algorithms, which solve the same problem.

The data is entered in all forms separately and whenever an error occurred, it is corrected immediately. A quality team deputed by the management verified all the necessary documents and tested the Software while entering the data at all levels. The development process involves various types of testing. Each test type addresses a specific testing requirement. The most common types of testing involved in the development process are:

- Unit Test.
- Functional Test.
- Integration Test.

8.1 UNIT TEST

The first test in the development process is the unit test. The source code is normally divided into modules, which in turn are divided into smaller units called units. These units have specific behavior. The test done on these units of code is called unit test. Unit test depends upon the language on which the project is developed. Unit tests ensure that each unique path of the project performs accurately to the documented specifications and contains clearly defined inputs and expected results.

8.2 FUNCTIONAL TESTING

Functional test can be defined as testing two or more modules together with the intent of finding defects, demonstrating that defects are not present, verifying that the module performs its intended functions as stated in the specification and establishing confidence that a program does what it is supposed to do.

8.3 INTEGRATION TESTING

In integration testing modules are combined and tested as a group. Modules are typically code modules, individual applications, source and destination applications on a network, etc. Integration Testing follows unit testing and precedes system testing. Testing after the product is code complete. Betas are oftenwidely distributed or even distributed to the public at large in hopes that they will buy the final product when it is released.

CHAPTER 9

SYSTEM STUDY

SOURCE CODE

Program for Cloud Storage

```
using BL.security;
using Microsoft.AspNetCore.Hosting;
using Microsoft.AspNetCore.Http;
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Net;
using System.Text;
using System.Threading.Tasks;
namespace BL.CloudStorage
{
 public class StoreDataToCLoud
 {
 static string ftpServer = "ftp.drivehq.com"; // you can also use
proftp.drivehq.com if you have a paid account.
 static string username = "guna1991"; // Your DriveHQ username
 static string password = "Ashok@1991"; // Your DriveHQ password
 /// <summary>
 /// Upload a file to the FTP server.
 /// </summary>
 /// <param name="srcFilePath">a local file path</param>
 /// <param name="destFilePath">the path on the FTP server, such as
```

```
/temp/filename</param>
 /// <returns></returns>
 public static bool UploadFile(string srcFilePath, string destFilePath = null)
 {
 if (String.IsNullOrWhiteSpace(srcFilePath))
 throw new ArgumentNullException("Source FilePath.");
 if (String.IsNullOrWhiteSpace(destFilePath))
 destFilePath = Path.GetFileName(srcFilePath);
 Uri serverUri = GetUri(destFilePath);
 // the serverUri should start with the ftp:// scheme.
 if (serverUri.Scheme != Uri.UriSchemeFtp)
return false;
 FtpWebRequest request = CreateFtpRequest(serverUri,
WebRequestMethods.Ftp.UploadFile);
 // read file content into byte array
 FileStream sourceStream = new FileStream(srcFilePath, FileMode.Open,
FileAccess.Read);
 byte[] fileContent = new byte[sourceStream.Length];
 sourceStream.Read(fileContent, 0, fileContent.Length);
 sourceStream.Close();
 // var data = CryptoService.Encrypt(fileContent, "sblw-3hn8-sqoy19");
 // send the file content to the FTP server
 request.ContentLength = fileContent.Length;
 Stream requestStream = request.GetRequestStream();
 requestStream.Write(fileContent, 0, fileContent.Length);
 requestStream.Close();
 FtpWebResponse response = (FtpWebResponse)request.GetResponse();
 Console.WriteLine("Response status: {0} - {1}", response.StatusCode,
```

```
 response.StatusDescription);
  return true;
  }
  /// <summary>
  /// Download srcFilePath on the FTP server to the destFilePath. If null, save to
 the current folder)
  /// </summary>
  /// <param name="srcFilePath"></param>
  /// <param name="destFilePath"></param>
  /// <returns></returns>
  public static bool DownloadFile(string srcFilePath, I Hosting Environment
 Env, string destFilePath = null)
  {
  if (String.IsNullOrWhiteSpace(srcFilePath))
  throw new ArgumentNullException("Source FilePath.");
  if (String.IsNullOrWhiteSpace(destFilePath))
destFilePath = Path.Combine(Env.WebRootPath,"Downloads",srcFilePath);
Uri serverUri = GetUri(srcFilePath);
 //// the serverUri should start with the ftp:// scheme.
 if (serverUri.Scheme != Uri.UriSchemeFtp)
 return false;
 FtpWebRequest request = CreateFtpRequest(serverUri,
WebRequestMethods.Ftp.DownloadFile);
 FtpWebResponse response = (FtpWebResponse)request.GetResponse();
 Stream responseStream = response.GetResponseStream();
 FileStream fileStream = new FileStream(destFilePath,
FileMode.OpenOrCreate, FileAccess.Write);
 byte[] buffer = new byte[32 * 1024];
 while (true)
```

```
 {
 int bytesRead = responseStream.Read(buffer, 0, buffer.Length);
 if (bytesRead == 0)
 break;
fileStream.Write(buffer, 0, buffer.Length);
 }
 fileStream.Close();
 Console.WriteLine("Response status: {0} - {1}", response.StatusCode,
response.StatusDescription);
 return true;
 }
 /// <summary>
 /// List an FTP folder's content
 /// </summary>
/// <param name="path"></param>
 /// <returns></returns>
 public static string List(string path)
 {
 Uri serverUri = GetUri(path);
 //// the serverUri should start with the ftp:// scheme.
 if (serverUri.Scheme != Uri.UriSchemeFtp)
 return "";
 FtpWebRequest request = CreateFtpRequest(serverUri,
WebRequestMethods.Ftp.ListDirectory);
 FtpWebResponse response = (FtpWebResponse)request.GetResponse();
 StreamReader sr = new StreamReader(response.GetResponseStream());
 string result = sr.ReadToEnd();
 Console.WriteLine("Response status: {0} - {1}", response.StatusCode,
response.StatusDescription + "\r\n" + result);
```

```
return result;
}
private static FtpWebRequest CreateFtpRequest(Uri serverUri, string method)
{
FtpWebRequest request = (FtpWebRequest)WebRequest.Create(serverUri);
request.EnableSsl = true;
request.UsePassive = true;
request.UseBinary = true;
request.KeepAlive = true;
request.Credentials = new NetworkCredential(username, password);
request.Method = method;
return request;
}
private static Uri GetUri(string remoteFilePath)
{
Uri ftpServerUri = new Uri("ftp://" + ftpServer+ "/Uploads/");
return new Uri(ftpServerUri, remoteFilePath);
}
}
}
```

Program for Admin Schema

```
using GSchema;
using System;
using System.Collections.Generic;
using System.Text;
namespace BL.SchemaModel
{
```

```
public class AdminSchema
{
[GSchema("uname", "User Name", "string", true, getHtmlClass = "col-md6")]
public string uname { get; set; }
[GSchema("password", "Password", "password", true, getHtmlClass = "colmd
6")]
public string password { get; set; }
[GSchema("name", "Name", "string", true, getHtmlClass = "col-md-6")]
public string name { get; set; }
[GSchema("zone", "Zone", "string", true, getEnumVal = "zone", getHtmlClass
= "col -md-6", getfieldHtmlClass = "select2 selectfield")]
public string zone { get; set; }
[GSchema("address", "Address", "string", true, getHtmlClass = "col-md-12")]
public string address { get; set; }
[GSchema("email", "Email", "email", true, getHtmlClass = "col-md-6")]
public string email { get; set; }
[GSchema("phone", "Phone", "number", true, getHtmlClass = "col-md-6")]
public string phone { get; set; }
[GSchema("status", "Status", "string", true, getEnumVal = "status",
getHtmlClass = "col-md-6", getfieldHtmlClass = "select2 selectfield")]
public string status { get; set; }
[GSchema("gst", "GST No", "string", true, getHtmlClass = "col-md-6")]
public string gst { get; set; }
[GSchema("photo", "Photo", "file", false, getHtmlClass = "col-md-6")]
public byte[] photo { get; set; }
[GSchema("remarks", "Remarks", "string", false, getHtmlClass = "col-md6")]
public string remarks { get; set; }
}
}
```

Program for User Schema

```
using GSchema;
using System;
using System.Collections.Generic;
using System.Text;
namespace BL.SchemaModel
{
 public class UserSchema
 {
 [GSchema("name", "Name", "string", true, getHtmlClass = "col-md-6")]
 public string name { get; set; }
 [GSchema("email", "Email", "email", true, getHtmlClass = "col-md-6")]
 public string email { get; set; }
 [GSchema("phone", "Phone", "number", true, getHtmlClass = "col-md-6")]
 public string phone { get; set; }
 [GSchema("type", "Role", "string", true, getEnumVal = "role", getHtmlClass =
"col-md-6", getfieldHtmlClass = "select2 selectfield")]
 public string type { get; set; }
 [GSchema("uname", "User Name", "string", true, getHtmlClass = "col-md6")]
 public string uname { get; set; }
 [GSchema("password", "Password", "password", true, getHtmlClass = "colmd-
6")]
 public string password { get; set; }
 [GSchema("zone", "Zone", "string", true, getEnumVal = "zone", getHtmlClass
= "col-md-6", getfieldHtmlClass = "select2 selectfield")]
 public string zone { get; set; }
 [GSchema("address", "Address", "textarea", true, getHtmlClass = "col-md6")]
 public string address { get; set; }
```

```
[GSchema("area", "Area", "string", true, getEnumVal = "area", getHtmlClass =
"col-md-6", getfieldHtmlClass = "select2 selectfield")]
 public string area { get; set; }
 [GSchema("remarks", "Remarks", "textarea", true, getHtmlClass = "col-md6")]
 public string remarks { get; set; }
 }
}
```

Program for Security CryptoService

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Security.Cryptography;
using System.Text;
using System.Threading.Tasks;
namespace BL.security
{
 public class CryptoService
 {
 public static byte[] Encrypt(byte[] input, string key)
 {
 TripleDESCryptoServiceProvider tripleDES = new
TripleDESCryptoServiceProvider();
 tripleDES.Key = UTF8Encoding.UTF8.GetBytes(key);
 tripleDES.Mode = CipherMode.ECB;
 tripleDES.Padding = PaddingMode.PKCS7;
 ICryptoTransform cTransform = tripleDES.CreateEncryptor();
 byte[] resultArray = cTransform.TransformFinalBlock(input, 0, input.Length);
 tripleDES.Clear();
```

```
 return resultArray;
 }
 public static byte[] Decrypt(byte[] input, string key)
 {
TripleDESCryptoServiceProvidertripleDES=newTripleDESCryptoService
Provider();
 tripleDES.Key = UTF8Encoding.UTF8.GetBytes(key);
 tripleDES.Mode = CipherMode.ECB;
 tripleDES.Padding = PaddingMode.PKCS7;
 ICryptoTransform cTransform = tripleDES.CreateDecryptor();
 byte[] resultArray = cTransform.TransformFinalBlock(input, 0, input.Length);
 tripleDES.Clear();
 return resultArray;
 }
 }
}
```

Hash Service

```
using System;
using System.Collections.Generic;
using System.Security.Cryptography;
using System.Text;
namespace BL.security
{
 public class HasherService
 {
 public HasherService()
 {
 }
```

```
public static byte[] GenerateSalt()
{
const int saltLength = 32;
using (var randomNumberGenerator = new RNGCryptoServiceProvider())
{
var randomNumber = new byte[saltLength];
randomNumberGenerator.GetBytes(randomNumber);
return randomNumber;
}
}
private static byte[] Combine(byte[] first, byte[] second)
{
var ret = new byte[first.Length + second.Length];
Buffer.BlockCopy(first, 0, ret, 0, first.Length);
Buffer.BlockCopy(second, 0, ret, first.Length, second.Length);
return ret;
}
public static byte[] HashPasswordWithSalt(byte[] toBeHashed, byte[] salt)
{
using (var sha256 = SHA256.Create())
{
var combinedHash = Combine(toBeHashed, salt);
return sha256.ComputeHash(combinedHash);
}
}
}
}
```

Program for Key Generator

```
using Microsoft.AspNetCore.Http;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Security.Cryptography;
using System.Text;
using System.Threading.Tasks;
namespace BL.VSEF
{
 public static class KeyGenerator
 {
 public static string ToGeneratePublicSearchableKey(this string Id,HttpContext
context)
 {
 string productName = typeof(KeyGenerator).Assembly.GetName().Name;
 string clientIdentificationId = context.Request.Host.Value;
 string versionNumber = "01.00.00";
 string productIdentifier = (productName + "-" + clientIdentificationId + "-"
+Id+"-"+ versionNumber).ToLower();
return GenerateLicenseKey(productIdentifier);
 }
 static string GenerateLicenseKey(string productIdentifier)
 {
 return FormatLicenseKey(GetMd5Sum(productIdentifier));
 }
 static string GetMd5Sum(string productIdentifier)
 {
 System.Text.Encoder enc = System.Text.Encoding.Unicode.GetEncoder();
```

```
byte[] unicodeText = new byte[productIdentifier.Length * 2];
enc.GetBytes(productIdentifier.ToCharArray(), 0, productIdentifier.Length,
unicodeText, 0, true);
MD5 md5 = new MD5CryptoServiceProvider();
byte[] result = md5.ComputeHash(unicodeText);
StringBuilder sb = new StringBuilder();
for (int i = 0; i < result.Length; i++)
{
sb.Append(result[i].ToString("X2"));
}
return sb.ToString();
}
static string FormatLicenseKey(string productIdentifier)
{
productIdentifier = productIdentifier.Substring(0, 28).ToUpper();
char[] serialArray = productIdentifier.ToCharArray();
StringBuilder licenseKey = new StringBuilder();
int j = 0;
for (int i = 0; i < 28; i++)
{
for (j = i; j < 4 + i; j++)
{
licenseKey.Append(serialArray[j]);
}
if (j == 28)
{
break;
}
else
```

```
 {
 i = (j) - 1;
 licenseKey.Append("-");
 }
 }
 return licenseKey.ToString();
 }
 }
}
```

Program for Encryption Search

```
<Project Sdk="Microsoft.NET.Sdk.Web">
 <PropertyGroup>
 <TargetFramework>net5.0</TargetFramework>
 </PropertyGroup>
 <ItemGroup>
<ContentInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\
bower.json" />
<ContentInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\
dist\css\bootstrap-theme.css" />
<ContentInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\
dist\css\bootstrap-theme.min.css" />
<ContentInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\
dist\css\bootstrap.css" />
<ContentInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\
dist\css\bootstrap.min.css" />
<ContentInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\
dist\fonts\glyphicons-halflings-regular.eot" />
<ContentInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\
```

```
dist\fonts\glyphicons-halflings-regular.ttf" />
<ContentInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\
dist\fonts\glyphicons-halflings-regular.woff" />
<ContentInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\
bower.json" />
<ContentInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\MIT
LICENSE.txt" />
<ContentInclude="wwwroot\vendors\parsleyjs\bower_components\mocha\
bower.json" />
<ContentInclude="wwwroot\vendors\parsleyjs\bower_components\mocha\
mocha.css" />
<ContentInclude="wwwroot\vendors\parsleyjs\bower_components\sinonjs\
bower.json" />
<ContentInclude="wwwroot\vendors\parsleyjs\bower_components\uwidget\
bower.json" />
<ContentInclude="wwwroot\vendors\parsleyjs\bower_components\uwidget\
uwidget.css" />
 </ItemGroup>
 <ItemGroup>
 <None Include="wwwroot\js\site.js" />
 <None Include="wwwroot\lib\bootstrap\dist\css\bootstrap-grid.css.map" />
 <None Include="wwwroot\lib\bootstrap\dist\css\bootstrap-grid.min.css.map"
/>
 <None Include="wwwroot\lib\bootstrap\dist\css\bootstrap-reboot.css.map" />
 <NoneInclude="wwwroot\lib\bootstrap\dist\css\bootstrap-reboot.min.css.map"
/>
 <None Include="wwwroot\lib\bootstrap\dist\css\bootstrap.css.map" />
 <None Include="wwwroot\lib\bootstrap\dist\css\bootstrap.min.css.map" />
 <None Include="wwwroot\lib\bootstrap\dist\js\bootstrap.bundle.js" />
```

<None Include="wwwroot\lib\bootstrap\dist\js\bootstrap.bundle.js.map" />
<None Include="wwwroot\lib\bootstrap\dist\js\bootstrap.bundle.min.js" />
<None Include="wwwroot\lib\bootstrap\dist\js\bootstrap.bundle.min.js.map"/>
<None Include="wwwroot\lib\bootstrap\dist\js\bootstrap.js" />
<None Include="wwwroot\lib\bootstrap\dist\js\bootstrap.js.map" />
<None Include="wwwroot\lib\bootstrap\dist\js\bootstrap.min.js" />
<None Include="wwwroot\lib\bootstrap\dist\js\bootstrap.min.js.map" />
<None Include="wwwroot\lib\bootstrap\LICENSE" />
<NoneInclude="wwwroot\lib\jqueryvalidationunobtrusive\jquery.validate.unobtrusive.js" />
<NoneInclude="wwwroot\lib\jqueryvalidationunobtrusive\jquery.validate.unobtrusive.min.js" />
<None Include="wwwroot\lib\jquery-validation\dist\additional-methods.js" />
<NoneInclude="wwwroot\lib\jquery-validation\dist\additional-methods.min.js" />
<None Include="wwwroot\lib\jquery-validation\dist\jquery.validate.js" />
<None Include="wwwroot\lib\jquery-validation\dist\jquery.validate.min.js" />
<None Include="wwwroot\lib\jquery-validation\LICENSE.md" />
<None Include="wwwroot\lib\jquery\dist\jquery.js" />
<None Include="wwwroot\lib\jquery\dist\jquery.min.js" />
<None Include="wwwroot\lib\jquery\dist\jquery.min.map" />
<None Include="wwwroot\vendors\bootstrap\.github\CODEOWNERS" />
<None Include="wwwroot\vendors\bootstrap\.github\CONTRIBUTING.md" />
<NoneInclude="wwwroot\vendors\bootstrap\.github\ISSUE_TEMPLATE\bug md" />
<NoneInclude="wwwroot\vendors\bootstrap\.github\ISSUE_TEMPLATE\bug_report.md" />
<NoneInclude="wwwroot\vendors\bootstrap\.github\ISSUE_TEMPLATE\

feature.md" />
<NoneInclude="wwwroot\vendors\bootstrap\.github\ISSUE_TEMPLATE\
feature_request.md" />
<None Include="wwwroot\vendors\bootstrap\.github\SUPPORT.md" />
<None Include="wwwroot\vendors\Chart.js\.github\ISSUE_TEMPLATE.md"
/>
<NoneInclude="wwwroot\vendors\Chart.js\.github\PULL_REQUEST_
TEMPLATE.md" />
<None Include="wwwroot\vendors\echarts\.github\CONTRIBUTING.md" />
<None Include="wwwroot\vendors\echarts\.github\ISSUE_TEMPLATE.md"
/>
<None Include="wwwroot\vendors\Flot\.circleci\config.yml" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\dist\
fonts\glyphicons-halflings-regular.svg" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\dist\
js\bootstrap.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\dist\
js\bootstrap.min.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\
DOCS-LICENSE" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\js\af
fix.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\js\
alert.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\js\
button.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\js\
carousel.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\js\

collapse.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\js\
dropdown.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\js\
modal.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\js\
popover.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\js\
scrollspy.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\js\
tab.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\js\
tooltip.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\js\
transition.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less
\alerts.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less
\badges.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less
\bootstrap.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less
\breadcrumbs.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less
\buttongroups.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less
\buttons.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less
\carousel.less" />

<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\close.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\code.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\component-animations.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\dropdowns.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\forms.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\glyphicons.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\grid.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\inputgroups.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\jumbotron.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\labels.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\listgroup.less" />
<NoneSInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\media.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\mixins.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\modals.less" />

<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\navbar.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\navs.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\normalize.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\pager.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\pagination.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\panels.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\popovers.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\print.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\progress-bars.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\responsive-utilities.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\scaffolding.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\tables.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\theme.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\thumbnails.less" />

<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\tooltip.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\type.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\utilities.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\variables.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\less\wells.less" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\LICENSE" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\bootstrap\LICENSEMIT" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\expect.js\index.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\expect.js\Makefile" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\expect.js\support\jquery.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\dist\jquery.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\dist\jquery.min.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\dist\jquery.min.map" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\ajax.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\

ajax\jsonp.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\
ajax\load.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\
ajax\parseJSON.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\
ajax\parseXML.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\
ajax\script.js"/>
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\
ajax\var\nonce.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\
ajax\var\rquery.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\
ajax\xhr.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\
attributes.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\
attributes\attr.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\
attributes\classes.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\
attributes\prop.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\
attributes\support.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\
attributes\val.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\call
backs.js" />

<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\core.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\core\access.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\core\init.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\core\parseHTML.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\core\ready.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\core\var\rsingleTag.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\css.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\css\addGetHookIf.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\css\curCSS.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\css\defaultDisplay.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\css\hiddenVisibleSelectors.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\css\support.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\css\swap.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\css\var\cssExpand.js" />

<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\css\var\getStyles.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\css\var\isHidden.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\css\var\rmargin.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\css\var\rnumnonpx.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\data.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\data\accepts.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\data\Data.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\data\var\data_priv.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\data\var\data_user.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\deferred.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\deprecated.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\dimensions.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\effects.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\effects\animatedSelector.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\

effects\Tween.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\event.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\event\ajax.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\event\alias.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\event\support.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\exports\amd.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\exports\global.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\intro.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\jquery.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\manipulation.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\manipulation\support.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\manipulation\var\rcheckableType.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\manipulation_evalUrl.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\offset.js" />

<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\outro.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\queue.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\queue\delay.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\selectornative.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\selectorsizzle.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\selector.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\serialize.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\sizzle\dist\sizzle.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\sizzle\dist\sizzle.min.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\sizzle\dist\sizzle.min.map" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\traversing.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\traversing\findFilter.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\traversing\var\rneedsContext.js" />

<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\var\arr.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\var\class2type.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\var\concat.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\var\hasOwn.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\var\indexOf.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\var\pnum.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\var\push.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\var\rnotwhite.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\var\slice.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\var\strundefined.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\var\support.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\var\toString.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\jquery\src\wrap.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\mocha\LICENSE" />

<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\mocha\media\logo.svg" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\mocha\mocha.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\mocha\scripts\ensurecompatible-npm.sh" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\sinonjs\sinon.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\uwidget\LICENSE" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\uwidget\uwidget.js" />
<NoneInclude="wwwroot\vendors\parsleyjs\bower_components\uwidget\uwidget.less" />
</ItemGroup>
<ItemGroup>
<PackageReference Include="Microsoft.AspNetCore.Mvc.NewtonsoftJson" Version="5.0.3" />
<PackageReferenceInclude="Microsoft.VisualStudio.Web.CodeGeneration.Design" Version="5.0.2" />
</ItemGroup>
<ItemGroup>
<ProjectReference Include="..\BL\BL.csproj" />
</ItemGroup>
<ItemGroup>
<Folder Include="wwwroot\Downloads\" />
<Folder Include="wwwroot\Uploads\" />
</ItemGroup>
</Project>

Program for Mongodb.Identity.Core

```
using Microsoft.AspNetCore.Identity;
using Microsoft.Extensions.DependencyInjection;
using MongoDb.Identity.Core.Identity;
using MongoDb.Identity.Core.Identity.Stores;
using MongoDb.Identity.Core.Models;
using MongoDB.Bson;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace MongoDb.Identity.Core
{
 public static class StartUpExtentions
 {
 public static IServiceCollection MongoIdentityService(this IServiceCollection
services)
 {
 services.AddTransient<MongoTablesFactory>();
 var lockoutOptions = new LockoutOptions()
 {
 AllowedForNewUsers = true,
 DefaultLockoutTimeSpan = TimeSpan.FromMinutes(10),
 MaxFailedAccessAttempts = 3
 };
 services.AddDefaultIdentity<ApplicationUser>(opt =>
 {
 opt.Password.RequireDigit = false;
```

```
opt.Password.RequiredLength = 6;
 opt.Password.RequireLowercase = false;
 opt.Password.RequireNonAlphanumeric = false;
 opt.Password.RequireUppercase = false;
 opt.Lockout = lockoutOptions;
 opt.SignIn.RequireConfirmedAccount = false;
 opt.User.RequireUniqueEmail = true;
 })
 .AddRoles<ApplicationRole>()
 .AddDefaultTokenProviders()
 .AddRoleStore<MongoRoleStore<ApplicationRole, ObjectId>>()
 .AddUserStore<MongoUserStore<ApplicationUser, ObjectId>>();
 services.AddScoped<IUserClaimsPrincipalFactory<ApplicationUser>,
 AdditionalUserClaimsPrincipalFactory>();
 return services;
 }
 }
}
```

CHAPTER 10

SCREENSHOTS

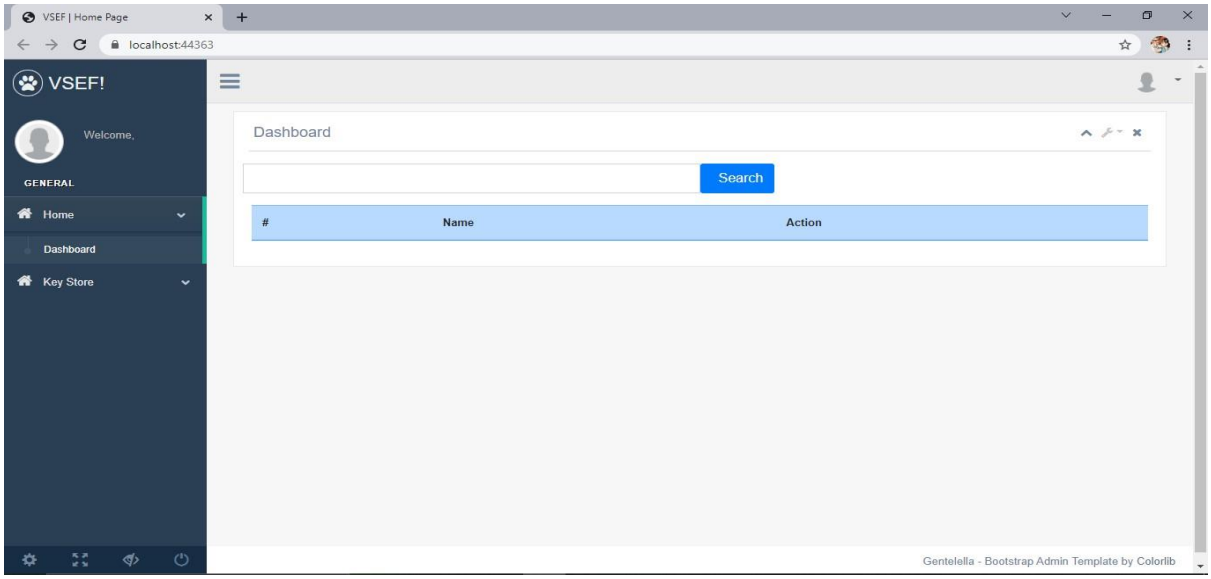

Fig 10.1 Dashboard

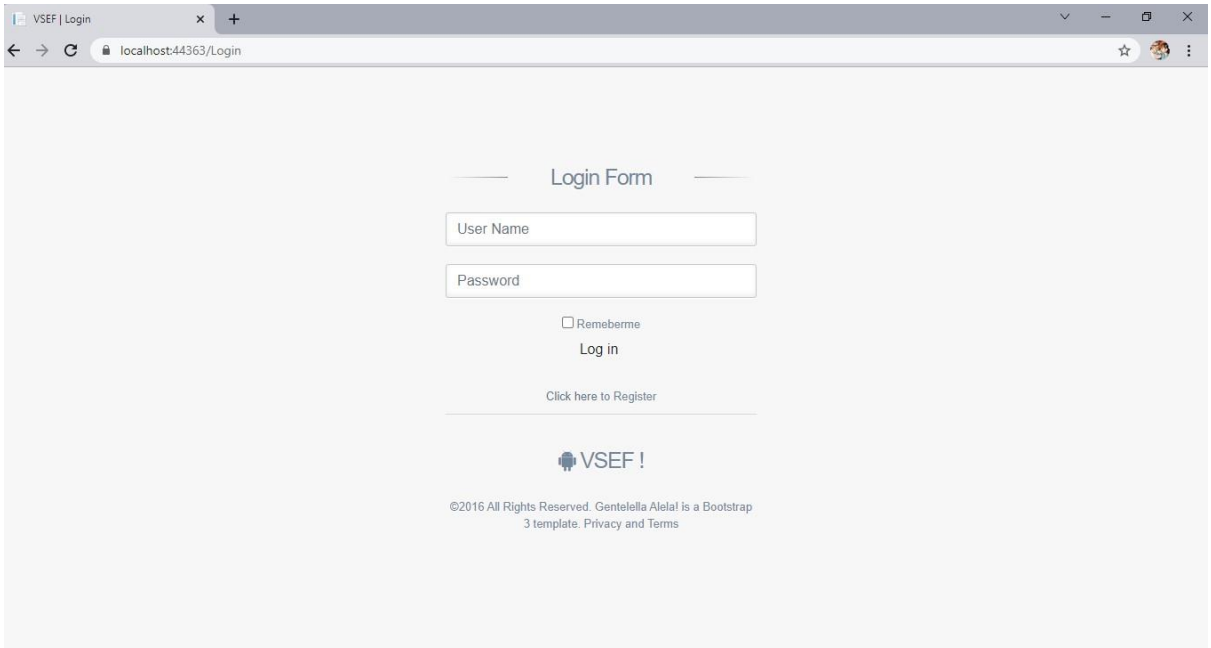

Fig 10.2 Login Page

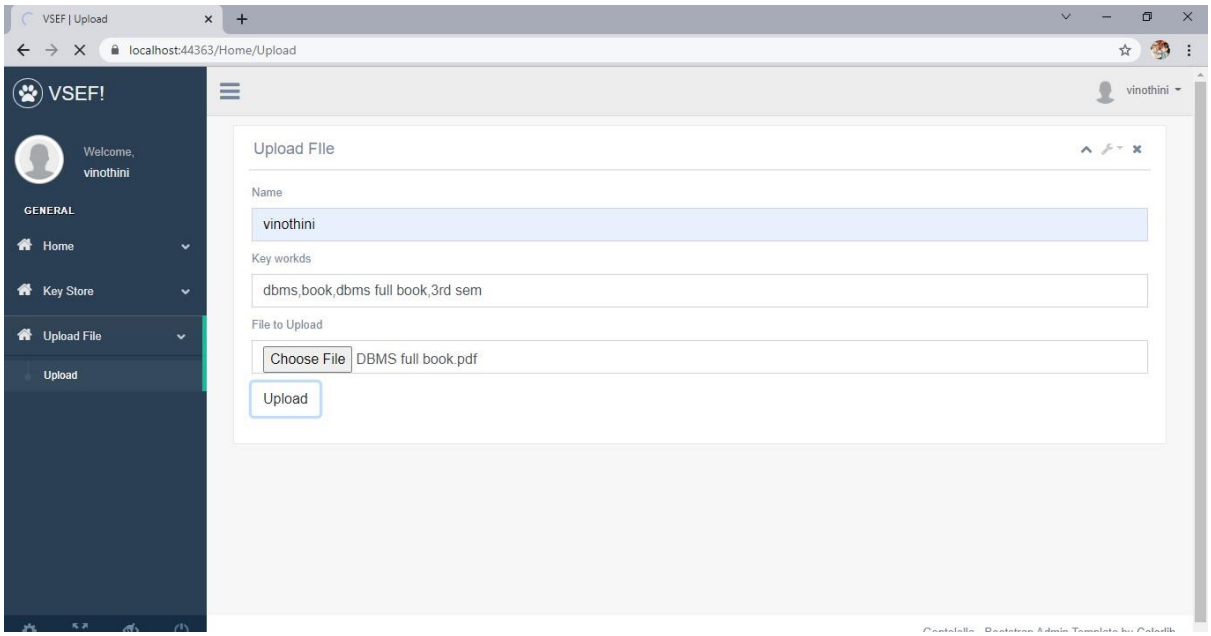

Fig 10.3 File Uploading

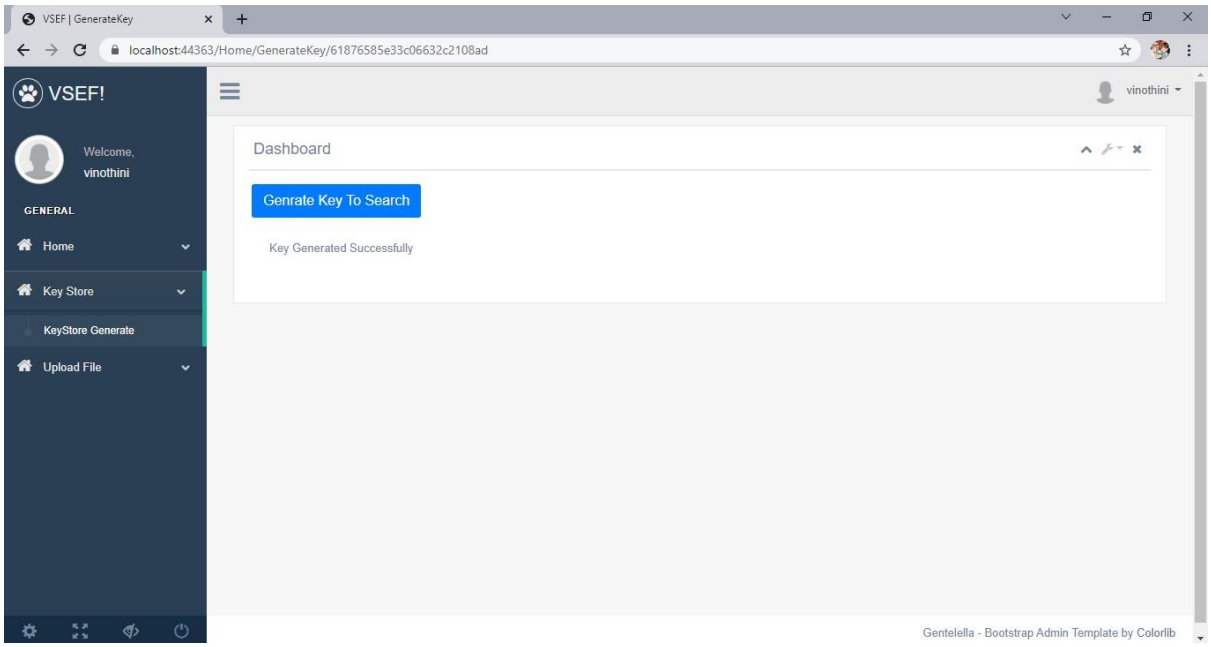

Fig 10.4 Key Generation

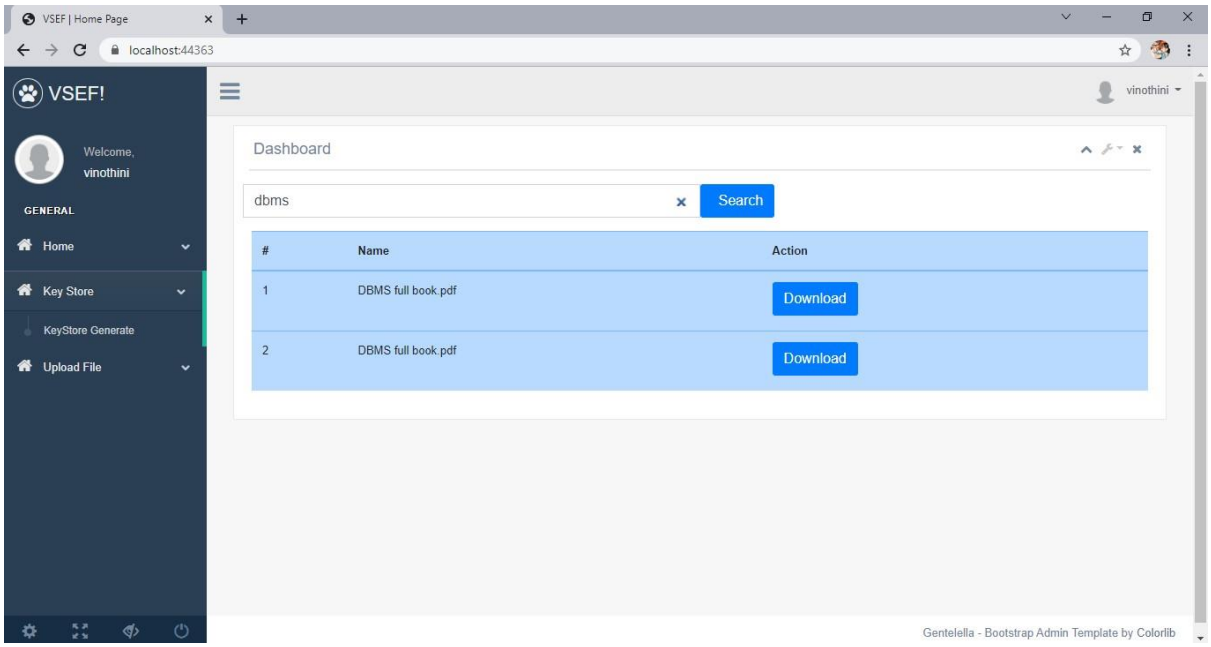

Fig 10.5 File Downloading

CHAPTER 11
CONCLUSION AND FUTURE ENHANCEMENT

11.1 CONCLUSION

The basic VSEF which mitigates the risk of inaccurate search results returned by the malicious CS and resists insider KGA was first proposed. Then, the basic VSEF was improved to support multi-keyword search, multi-key encryption and dynamic update at the same time in enhanced VSEF. We proved that basic or enhanced VSEF is semantically secure against the insider KGA, and achieves both correctness and soundness.Verifiable searchable encryption (VSE) is a cryptographic technique that allows a user to store encrypted data on an untrusted server while still being able to search and retrieve specific pieces of information. VSE provides an additional layer of security by allowing the user to verify that the search results are correct, without revealing any information to the server.

VSE frameworks use a combination of techniques, including symmetric and asymmetric encryption, hash functions, and zero-knowledge proofs, to ensure the privacy and authenticity of the data and search results. One of the most significant benefits of VSE is that it allows for efficient searching and retrieval of data, while maintaining the privacy and integrity of the data.VSE has numerous applications, including in healthcare, finance, and other fields where sensitive information must be stored and accessed securely. VSE also provides a solution to the growing need for privacy-preserving search functionality in cloud storage and computing.Overall, VSE is a powerful tool for protecting sensitive data in cloud environments, and its continued development and refinement will likely have significant implications for the future of data privacy and security.

11.2 FUTURE ENHANCEMENT

- Efficiency improvements: One of the main challenges of VSE schemes is their high computational and communication overheads. Researchers can work on developing more efficient VSE schemes that reduce the time and resources required for keyword search operations.

- Support for complex queries: Current VSE schemes only support simple keyword searches, which can limit their usefulness in practical scenarios. Researchers can work on developing VSE schemes that support more complex queries, such as boolean operators and proximity searches.

- Better security guarantees: VSE schemes rely on cryptographic primitives, such as hash functions and symmetric encryption, to ensure the confidentiality and integrity of the data. However, the security of these primitives can be compromised by attacks such as collision attacks and brute-force attacks. Researchers can work on developing VSE schemes that provide better security guarantees against such attacks.

- Interoperability: VSE schemes from different vendors are not currently interoperable, meaning that data encrypted using one VSE scheme cannot be searched using another VSE scheme. Researchers can work on developing interoperable VSE schemes that allow data to be searched across multiple VSE schemes.

- Scalability: VSE schemes are currently designed for small-scale deployments and may not be suitable for large-scale systems. Researchers can work on developing VSE schemes that can scale to handle large amounts of data and support high query rates.

REFERENCES

[1] Efficient Verifiable Searchable Encryption for Big Data" by Pengfei Hao, Zheli Liu, and Jianfeng Ma (IEEE Transactions on Dependable and Secure Computing, 2021).

[2] Verifiable Searchable Symmetric Encryption with Efficient Search Pattern Hiding" by Jian Liu, Wei Shi, Mingzhong Wang, and Yuanyuan Zhang, published in IEEE Transactions on Information Forensics and Security, Vol. 16, pp. 2324-2337, 2021.

[3] Verifiable Searchable Encryption for Outsourced Databases" by Rui Zhang, Chunxiao Li, and Ximeng Liu (ACM Transactions on Storage, 2021).

[4] Verifiable Searchable Symmetric Encryption with Efficient Updates" by Fangguo Zhang, Guojun Wang, and Wei Wu (IEEE Transactions on Information Forensics and Security, 2021).

[5] Verifiable Searchable Encryption with Practical Efficiency" by Xiuzhen Cheng, Chen Chen, and Xinyi Huang (IEEE Transactions on Information Forensics and Security, 2021).

[6] Efficient Verifiable Searchable Encryption with Security Against Insider Attacks" by Zhou et al. (2021).

[7] Verifiable Conjunctive Keyword Search Over Encrypted Data with User Revocation" by Yu et al. (2021).

[8] Privacy-Preserving Verifiable Searchable Encryption with Efficient Revocation" by Zhang et al. (2021).

[9] Verifiable Multi-Keyword Search over Encrypted Data with User Revocation" by Wang et al. (2020).

[10] Efficient Verifiable Boolean Keyword Search over Encrypted Data with Fine-Grained Access Control" by Ma et al. (2020).