DATA SECURITY AND PRIVACY PROTECTION FOR CLOUD STORAGE

Submitted in partial fulfillment of the

requirements for the award of

Bachelor of Engineering degree in Computer Science and Engineering

By

NARU SAI LOHITH (39110681)
PEMMARAJU BHANU BHARADWAJ (39110762)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHISALAI,
CHENNAI - 600119
APRIL – 2023

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with —A‖ grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Naru Sai Lohith (39110681) and Pemmaraju Bhanu Bharadwaj (39110762)** who carried out the Project Phase-1 entitled **"DATA SECURITY AND PRIVACY PROTECTION FOR CLOUD STORAGE"** under my supervision from Jan 2023 to April 2023.

[illegible]

Internal Guide
Dr. N. SRINIVASAN, M.E., Ph.D.

[illegible]

Head of the Department
Dr. L. LAKSHMANAN, M.E., Ph.D.

Submitted for Viva-voce Examination held on 20.04.2023

[illegible] [illegible]

Internal Examiner **External Examiner**

DECLARATION

I, **Naru Sai Lohith (39110681),** hereby declare that the Project Phase-1 Report entitled **"DATA SECURITY AND PRIVACY PROTECTION FOR CLOUD STORAGE"** done by me under the guidance of **Dr. N. Srinivasan, M.E., Ph.D.** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in **Computer Science and Engineering**.

[illegible]

DATE: 07/04/2023

PLACE: Chennai **SIGNATURE OF THE CANDIDATE**

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to the **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala, M.E., Ph. D**, **Dean**, School of Computing, and **Dr. L. Lakshmanan, M.E., Ph.D., and Dr. S. Vigneshwari, M.E., Ph.D.,** Heads of the Department of Computer Science and Engineering for providing me with the necessary support and details at the right time during the progressive projects.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. N. Srinivasan, M.E., Ph.D.,** for his valuable guidance, suggestions, and constant encouragement that paved way for the successful completion of my phase-1 project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Cloud storage technology is receiving greater attention and better development as a result of the rapid expansion of unstructured data and the development of cloud computing technology. Regarding the data and information saved and maintained globally via the cloud, the cloud provider has no recommendations. The majority of privacy protection strategies rely on encryption technology. A fog-based, three-layer storage architecture. The suggested architecture may both fully utilize cloud storage and safeguard data privacy. Here, we're employing the hash-Solomon algorithm, which is made to separate data into sections. Typically, encryption technology is the foundation of privacy protection strategies. To prevent data from being stored in the cloud, there are several privacy-preserving techniques available. We suggest a three-layer fog computing-based storage architecture. The suggested architecture may both fully utilize cloud storage and safeguard data privacy. The Hash-Solomon coding method is being used in this instance to segment the data. If one data component is absent, the data information is lost. Using bucket concept-based algorithms, we secure the data in this framework so that it can demonstrate the security and effectiveness of our plan. Additionally, this approach can calculate the distribution proportion saved on the cloud, fog, and local computers, respectively, based on computational intelligence.

Keywords: Cloud computing, Fog computing, Hash-Solomon code, Bucket framework, Advanced Encryption Standard (AES), and Secured Hash Algorithm (SHA).

TABLE OF CONTENTS

Chapter No	TITLE	Page No
	ABSTRACT	v
	LIST OF FIGURES	ix
1	INTRODUCTION	1
	1.2 Scope and Objective	2
	LITERATURE SURVEY	
2	2.1 Inference from the Literature Survey	4
	2.2 Open Problems in the Existing System	9
	REQUIREMENT ANALYSIS	
	3.1 Objective	10
	3.2 Software Requirements Specification Document	10
3	3.2.1 Hardware Requirements	
	3.2.2 Software Requirements	
	DESCRIPTION OF THE PROPOSED SYSTEM	
	4.1 Description of the Proposed System	11
	4.1.1 Advantages of Proposed System	
	4.2 Selected Methodology of Process Model	11
	4.2.1 Hash-Solomon Code	
	4.2.2 Bucket	
	4.2.3 BCH Code Algorithm	
	4.2.4 Advanced Encryption Standard (AES)	
	4.2.5 Secure Hash Algorithm (SHA)	
4	4.3 Architecture/ Overall Design of the Proposed System	14
	4.4 Description of Software for Implementation and Testing Plan of the Proposed Model/System	16
	4.4.1 Visual Studio	17

	4.4.1.1 Architecture of Visual Studio	
	4.4.1.2 Features of Visual Studio	
	4.4.2 SQL Server	22
	4.4.2.1 SQL Server Management Studio	
	4.4.2.2 Microsoft SQL SMS	
	4.4.3 .NET	26
	4.4.3.1 The .NET Framework	
	4.4.3.2 Features of .NET	
	4.5 Project Management Plan/Modules	31
	4.5.1 Login Module	
	4.5.2 Registration Module	
	4.5.3 Storage Module	
	4.5.4 Recovery Module	
	4.5.5 Download Module	
	SYSTEM ANALYSIS	
	5.1 Purpose	35
	5.2 Scope	35
5	5.3 Existing System	35
	5.3.1 Existing System Dis-advantages	
	5.4 Proposed System	36
	5.4.1 Proposed System Advantages	
	SYSTEM DESIGN	
	6.1 Input Design	37
6	6.1.1 Input Design Objectives	
	6.2 Output Design	38
	6.2.1 Output Design Objectives	
	SYSTEM TESTING	
	7.1 Overview	39
	7.2 Types of Tests	39
	7.3 Test Strategy and Approach	41
7	7.3.1 Test Objectives	
	7.3.2 Features to be Tested	42
	7.4 Requirement Analysis	42

	7.5 Functional Requirements	
	CONCLUSION	
8	8.1 Conclusion	44
	8.2 Future Enhancement	44
	REFERENCES	45
	APPENDIX	
	A. SOURCE CODE	47
	B. SCREENSHOTS	62
	C. RESEARCH PAPER	66

LIST OF FIGURES

Figure No	Figure Name	Page Number
4.2.3	Binary BCH Code	12
4.2.4	AES Algorithm	13
4.2.5	SHA Algorithm	14
4.3.1	System Architecture - 1	14
	System Architecture - 2	15
4.4.1.1	Visual Studio Architecture	18
4.4.2	SQL Server Management Studio	23
4.4.2.1	Microsoft SQL Server Management Studio	25
4.4.2.2	Database Creation	25
4.4.3	.NET Architecture	28
4.4.3.3	.NET Framework	31
4.5.3	Storage Procedure	33
4.5.5	Download Procedure	34

CHAPTER 1
INTRODUCTION

Cloud computing in computer science refers to a sort of outsourcing of computer services. It is easily usable by users. They do not have to be concerned with the production, distribution, or origin of the electricity. They pay for what they consumed each month. Similarly, to this, cloud computing allows users to access resources like storage, processing power, or custom development environments without having to worry about how they are managed internally. Internet-based computing is the norm for cloud computing. Based on how the internet is depicted in computer network diagrams, the cloud is a metaphor for the internet; as such, it is an abstraction that conceals the intricate infrastructure of the internet. This form of computing enables consumers to receive technology-enabled services through the Internet ("in the cloud") without being aware of or having any control over the underlying technologies. Fog computing, which refers to the increasing challenges in gaining objective access to information, can be seen in both massive data structures and large cloud systems. As a result, the obtained content is of poor quality. Fog computing may have a variety of different consequences on cloud computing and big data systems. A common characteristic that can be extracted, nevertheless, is the difficulty in accurately distributing content. This problem has been addressed by the development of metrics that aim to increase accuracy. A data plane and a control plane make up for networking. For instance, fog computing on the data plane enables computing services to live at the network's edge rather than on servers in a data center. Fog computing, as opposed to cloud computing, emphasizes proximity to end users and client objectives, dense geographical distribution and local resource pooling, latency reduction and backbone bandwidth savings to achieve a better quality of service (QoS), edge analytics/stream mining, leading to superior user-experience and redundancy in case of a failure while it is also capable of being used in AAL scenarios.

We recommend a TLS framework based on the fog computing model to ensure user privacy. The TSL framework efficiently protects user privacy while granting users some degree of administrative power. The inner invasion is challenging to fend off, as was already stated. Traditional methods are effective at addressing external attacks, but they are useless when CSP is experiencing issues. In contrast to conventional methods, our strategy uses encoding technology to partition user input into three segments of varying sizes. Each of them will be missing some crucial information necessary for secrecy. The three data components will be stored in the cloud server, the fog server, and the user's local workstation in that order, from large to small, by the fog computing concept. Even if the attacker obtains all the data from one server, he will not be able to reconstruct the user's original data using this technique. Because the fog server and local PC are both under human control, the CSP is also unable to obtain any relevant information without the data saved on those two devices.

1.2 SCOPE AND OBJECTIVE

SCOPE

A Three-Layer Privacy Preserving Cloud Storage Scheme Based on Computational Intelligence in Fog Computing**.** The privacy preservation is our focus, some active attacks are beyond the scope of this work. The three-layer cloud storage stores in to the three different parts of data parts. If the one data part missing, we lost the data information. In this proposed framework using the bucket concept-based algorithms. We are using a BCH code algorithm. It’s High flexible.

OBJECTIVE

Cloud storage also causes a series of secure problems. When using cloud storage, users do not actually control the physical storage of their data and it results in the separation of ownership and management of data. In order to solve the problem of privacy protection in cloud storage, we propose a TLS framework based on fog computing model and design a Hash-Solomon algorithm. Through the theoretical safety analysis, the scheme is proved to be feasible. By allocating the ratio of data blocks stored in different servers reasonably, we can ensure the privacy of data in each server. On another hand, cracking the encoding matrix is impossible theoretically. Besides, using hash transformation can protect the fragmentary information. Through the experiment test, this scheme can efficiently complete encoding and decoding without influence of the cloud storage efficiency. The three-layer cloud storage stores in to the three different parts of data parts. If the one data part missing, we lost the data information. In this proposed framework using the bucket concept-based algorithms.

CHAPTER 2
LITERATURE SURVEY

2.1 INFERENCES FROM LITERATURE SURVEY

Title 1: Privacy-preserving security solution for cloud services

Author: L. Malina*, J. Hajny, P. Dzurenda and V. Zeman

An innovative security solution for cloud services that protect privacy. Our approach is based on an effective nonlinear group signing method that offers anonymous access to cloud services and shared storage servers. The inventive approach provides registered users with anonymous authentication. As a result, users' personal information (age, valid registration, and successful payment) may be verified without disclosing the users' identities, and they can utilize cloud services without fear of having their behavior profiled. However, a user's access privileges are suspended if they violate the provider's policies. Our technology guarantees data transmission secrecy, un-link ability, and anonymous access. We put our idea into practice as a proof-of-concept application and then report the outcomes of our experiments. We also examine group signature schemes, which are fundamental components of privacy-enhancing cloud service solutions, and existing privacy-preserving solutions for cloud services. We assess how well our solution performs in comparison to similar solutions and schemes.

Title 2: A secure data privacy preservation for on-demand cloud service

Author: Chandramohan Dhasarathan *, Vengattaraman Thirumal, Dhavachelvan Ponnurangam

This essay focuses on concerns of intellectual property, confidentiality, and privacy that belong to the insurance and banking industries. If authoritarians misuse secret knowledge in the modern corporate world, privacy danger. disruptions caused by software when stealing digital data for external services. For the sake of business continuity, liability in digital secrecy In the cloud, where a vast amount of data is stored and kept up-to-date, isolation, mismanagement resulting in privacy breaches nearby, and its preventative phenomena are strictly regulated. Although cloud computing has altered the computer industry by improving its effectiveness, efficiency, and optimization of the service environment, among other things, cloud users' data and their identity, dependability, maintainability, and privacy may differ for various CPs (cloud providers). With today's technology, CP makes sure that the user's confidential information is kept more discreetly. The fact that not even the cloud provider has recommendations for the information and digital data saved and maintained internationally in the cloud is even more astonishing. One of the necessary research questions in cloud computing is the system that is being suggested. To prevent digital data loss in the cloud, we proposed the privacy-preserving model (PPM–DDLC). This suggestion enables the CR (cloud requester/users) to have confidence in the confidential data and information they save in the cloud.

Title 3: A Survey on Secure Storage Services in Cloud Computing

Author: Ms. B.Tejaswi, Dr. L.V.Reddy & Ms. M.Leelavathi

Cloud computing is a new technology that solely relies on the internet and its surroundings. Users can make use of a variety of services, including Software as a Service (SaaS), Platform as a Service (PaaS), Infrastructure as a Service (IaaS), and Storage as a Service (SaaS). Users and companies who utilize storage-as-a-service can store their data remotely, which introduces additional security concerns to the accuracy of data in the cloud. There are several methods for achieving secure cloud storage, including flexible distributed storage integrity auditing mechanisms, distributed erasure-coded data, Merkle Hash Tree (MHT) building, etc. These methods help to protect and effectively store dynamic data in the cloud. This study also discusses designs for managing security and privacy in the context of cloud storage.

Title 4: Survey on Privacy-Preserving Methods for Storage in Cloud Computing

Author: Neethu Mariam Joseph, Esther Daniel, N. A. Vasanthi, Ph.D

The human race is currently relying more and more on a variety of online storage facilities to back up our data or to use it in real-time, providing access from anywhere, at any time. Since the user's data is stored and managed outside of the user's premises, all of these services raise issues about security and privacy flaws for all the services they provide. This essay explores the different privacy concerns that arise when user data is stored by third-party service providers, sometimes known as cloud services. The term "cloud computing" refers to the basic framework for an emerging service delivery model that has the advantage of reducing costs by pooling computer and storage resources, along with an on-demand provisioning mechanism based on demand.

TITLE 5: Survey on Secure Storage in Cloud Computing

AUTHOR: **A. Rajathi1* and N. Saravanan2**

Cloud computing is a setting where information and resources are made available to end users over the Internet as a service. Thus, the cloud provides advantages in the form of online storage services and allows customers to access their data from any location at any time. Better performance, lower storage costs, and scalability are all benefits of using a cloud storage service rather than investing in expensive software and staff upkeep. However, securely maintaining saved data in a cloud environment is not a simple process, especially given the possibility that the stored data may not be entirely reliable. Cloud services are delivered via the internet, increasing their exposure to storage security flaws. However, one of the main issues keeping many large firms from utilizing cloud computing is security. This article projected several cloud storage systems, approaches, and their benefits, and disadvantages, and also discussed the difficulties involved in implementing safe cloud data storage. The results of this survey aid in determining potential areas for future study and strategies for addressing the problems now present.

TITLE 6: A Secure Cloud-assisted Urban Data Sharing Framework for Ubiquitous-cities

AUTHOR: Jian Shena,b,c,_, Dengzhi Liuc, Jun Shenc, Qi Liua,c, Xingming Suna,c

More and more individuals are moving into urban areas as urbanization picks up speed. New information and communication technologies are used to process urban data, which makes it easier to handle, in order to deal with the enormous data that are generated by citizens and public city departments. A cutting-edge form of computation is cloud computing. Numerous cloud-based applications have been developed since the commercialization of cloud computing. The cloud is partially trustworthy because a third party offers the cloud service. There are numerous security risks in cloud computing as a result of its features. A potential cryptography method that can be applied in the cloud to address a variety of security challenges is attribute-based encryption (ABE). In this research, we use attribute-based cryptography to present a system for exchanging urban data. We expand our scheme to support dynamic operations in order to make it suitable for the exploitation of omnipresent cities in the actual world. It can be deduced, in particular, from the performance analysis portion of the analysis, that our strategy is secure and can withstand potential attacks. Additionally, comparisons and experimental findings demonstrate that our technique is more effective in terms of computing.

TITLE 7: Security and Privacy Preservation Scheme of Face Identification and Resolution Framework Using Fog Computing in the Internet of Things

AUTHOR: Yanna Wang, and Xuanxia Yao

Technology for face resolution and recognition is essential to maintaining human identity both offline and online. The rise in applications dependent on face identification and resolution in the contemporary Internet of Things (IoT) and big data environment increases the demands on computing, communication, and storage resources. To increase processing power and conserve bandwidth, we have suggested the fog computing-based face detection and resolution framework. However, the characteristics of fog computing-based frameworks have some security and privacy challenges. To address the aforementioned problems, we suggest a security and privacy preservation approach in this work. We explain the foundation for face identification and resolution based on fog computing and briefly discuss security and privacy concerns. Then it is suggested that data encryption, data integrity checking, and authentication and session key agreement schemes be used to address the problems with confidentiality, integrity, and availability in the face identification and face resolution procedures. Finally, we put a prototype system into place to assess how security measures affect system performance. In the meantime, we assess and examine the suggested scheme's security characteristics from the perspective of logical formal proof and the CIA (confidentiality, integrity, availability) characteristics of information security. The findings show that the suggested plan can successfully satisfy the needs for security and privacy preservation.

Title 8: On a Relation Between Verifiable Secret Sharing Schemes and a Class of Error-Correcting Codes

Author: Ventzislav Nikov and Svetla Nikova

We attempt to provide a fresh perspective on Verifiable Secret Sharing Schemes (VSS). We start by creating a new "metric" (with slightly different properties than the standard Hamming metric). Based on a set of banned distances that is a monotonic decreasing set, we use this metric to design a very specific class of codes that we refer to as error-set correcting codes. Next, we adapt the packing problem to the new conditions and broaden the idea of error-set correcting codes' error-correcting abilities (taking into account the new metric and the new packing). Then, we take into account burst-error interleaving codes that propose an effective burst-error correcting technique, which is actually the well-known pair-wise checking protocol of Distributed Commitments (DC) and VSS, and we demonstrate the error-correcting ability of the error-set correcting interleaving codes.

Title 6: Security and Privacy Preservation Scheme of Face Identification and Resolution Framework Using Fog Computing in Internet of Things

Author: P. Hu, H. Ning, Y. Wang, X.

Face identification and resolution technology is crucial to ensure the identity consistency of humans in physical space and cyber space. In current Internet of Things (IoT) and big data situation, the increase of applications based on face identification and resolution raises the demands of computation, communication and storage capabilities. Therefore, we have proposed the fog computing based face identification and resolution framework to improve processing capacity and save the bandwidth. However, there are some security and privacy issues brought by the properties of fog computing based framework. In this paper, we propose a security and privacy preservation scheme to solve above issues. We give an outline of the fog computing based face identification and resolution framework, and summarize the security and privacy issues. Then the authentication and session key agreement scheme, data encryption scheme, and data integrity checking scheme are proposed to solve the issues of confidentiality, integrity, and availability in the processes of face identification and face resolution. Finally, we implement a prototype system to evaluate the influence of security scheme on system performance. Meanwhile, we also evaluate and analyze the security properties of proposed scheme from the viewpoint of logical formal proof and the CIA (confidentiality, integrity, availability) properties of information security. The results indicate that the proposed scheme can effectively meet the requirements for security and privacy preservation.

2.2 OPEN PROBLEMS IN THE EXISTING SYSTEM

- Chances in the understanding of risk as a result of extending the data center into the cloud.
- In this storage schema, the user’s data is totally stored in cloud servers. If the user loses their right to control data and faces privacy risks.
- The privacy protection schemes are usually based on encryption technology. These kinds of methods cannot effectively resist attacks from the inside of cloud servers.
- Data Integrity.
- Low latency.
- Location awareness.

CHAPTER 3
REQUIREMENT ANALYSIS

3.1 OBJECTIVE

A number of security issues are also brought on by cloud storage. Users that use cloud storage actually have no control over how their data is physically stored, which separates ownership and management of the data. We develop a Hash-Solomon algorithm and propose a TLS framework based on a fog computing model to address the issue of privacy protection in cloud storage. The approach is demonstrated to be practical through the theoretical safety analysis. We can guarantee the privacy of data in each server by distributing the ratio of data blocks stored on various servers properly. On the other side, it is theoretically impossible to crack the encoding matrix. Additionally, fragmentary information can be protected by applying the hash transformation. This approach successfully completed the experiment test's encoding and decoding without affecting the effectiveness of cloud storage. The three-layer cloud storage system stores data in three separate portions. We lose the data information if one data component is absent. Algorithms based on the bucket notion are used in the suggested framework.

3.2 SOFTWARE REQUIREMENTS SPECIFICATION DOCUMENT

3.2.1 HARDWARE REQUIREMENTS

System:	Intel i5 11Gen
Speed:	2.4GHZ
Hard Disk:	1 TB HDD, 256GB SSD
Monitor:	14” LED
Input Devices:	Keyboard, Mouse
Ram:	8GB

3.2.2 SOFTWARE REQUIREMENTS

Operating System:	Windows 11 Home
Coding Language:	Dot Net (.net)
IDE:	Visual Studio

Database:	SQL Server

CHAPTER 4

DESCRIPTION OF THE PROPOSED SYSTEM

4.1 DESCRIPTION OF THE PROPOSED SYSTEM

- The framework can take full of cloud storage and protect the privacy of data.
- Here cloud computing has attracted great attention from different sectors of society.
- The three-layer cloud storage stores into three different parts of data parts. If one data part missing, we lost the data information. This proposed framework uses bucket concept-based algorithms.

4.1.1 ADVANTAGES OF THE PROPOSED SYSTEM

- In our system we use a bucket concept to reduce data wastage and reduce process timings.
- We are using a BCH (Bose–Chaudhuri–Hocquenghem) code algorithm. It’s High flexible.
- BCH codes are used in many communications applications and low amount of redundancy.

4.2 SELECTED METHODOLOGY OR PROCESS MODEL

4.2.1 HASH-SOLOMON CODE

It is the process of splitting the data into three different parts based on the data. Mainly, the three different parts are the Cloud server, the Fog server, and the local host to ensure data integrity and security. Additionally, based on the Hash-Solomon code's characteristic, the technique may guarantee that the original data cannot be reconstructed by partial data. On the other hand, using the Hash-Solomon code results in some duplicated data blocks that are employed in the decoding process. Although increasing the amount of redundant blocks results in more data storage, it can also improve storage dependability. Our plan can effectively secure the privacy of user data by an appropriate allotment of the data. Complex calculations are required for the Hash-Solomon code, and computational intelligence can help (CI).

4.2.2 BUCKET

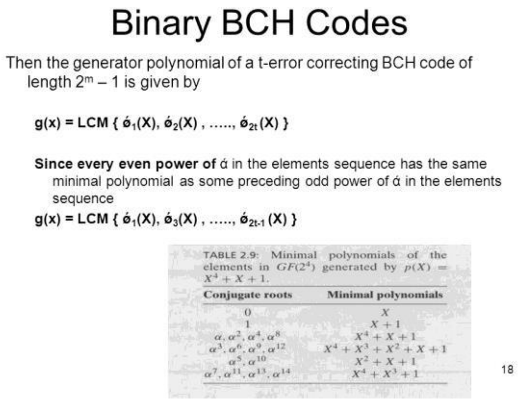
• The Access Control Lists (ACLs) for buckets inside Google Cloud Storage are represented by the Bucket Access Controls resource. You may control who gets access to your data and to what extent using ACLs.

• The three-layer cloud storage system divides data into three distinct pieces. We lose the data information if one data component is absent.

• The algorithms in this suggested framework are based on the bucket notion.

4.2.3 BCH CODE ALGORITHM

The Bose, Chaudhuri, and Hocquenghem (BCH) codes are a noteworthy expansion of the Hamming code for multiple-error correction, forming a huge class of potent random error-correcting cyclic codes. Hocquenghem discovered binary BCH codes in 1959, while Bose and Chaudhuri separately did the same in 1960. We are only considering Binary BCH code in this project.



Binary BCH Codes

Then the generator polynomial of a t-error correcting BCH code of length $2^m - 1$ is given by

$$g(x) = \text{LCM} \{ \acute{ø}_1(X), \acute{ø}_2(X), \ldots\ldots, \acute{ø}_{2t}(X) \}$$

Since every even power of ά in the elements sequence has the same minimal polynomial as some preceding odd power of ά in the elements sequence

$$g(x) = \text{LCM} \{ \acute{ø}_1(X), \acute{ø}_3(X), \ldots\ldots, \acute{ø}_{2t-1}(X) \}$$

TABLE 2.9: Minimal polynomials of the elements in $GF(2^4)$ generated by $p(X) = X^4 + X + 1$.

Conjugate roots	Minimal polynomials
0	X
1	$X + 1$
$\alpha, \alpha^2, \alpha^4, \alpha^8$	$X^4 + X + 1$
$\alpha^3, \alpha^6, \alpha^9, \alpha^{12}$	$X^4 + X^3 + X^2 + X + 1$
α^5, α^{10}	$X^2 + X + 1$
$\alpha^7, \alpha^{11}, \alpha^{13}, \alpha^{14}$	$X^4 + X^3 + 1$

18

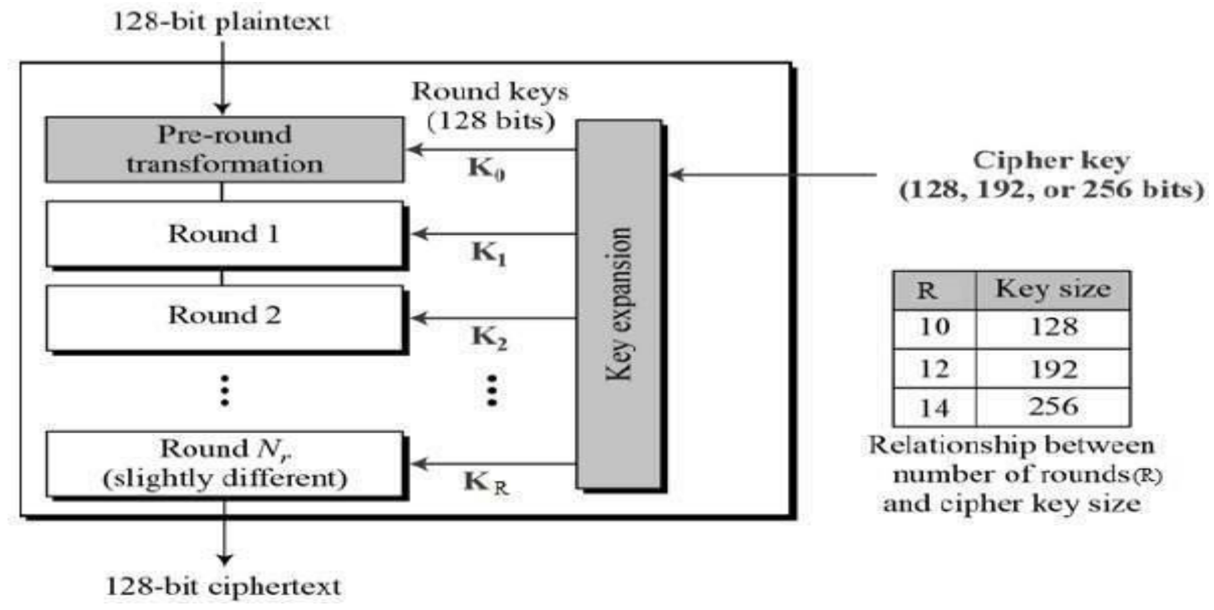
Fig 4.2.3: Binary BCH Code

4.2.4 ADVANCED ENCRYPTION STANDARD (AES)

The Advanced Encryption Standard is the more common and commonly used symmetric encryption algorithm that is likely to be used nowadays (AES). Compared to triple DES, it can be found at least six times faster.

As DES's key size was too small, a replacement was required. It was thought to be vulnerable to an exhaustive key search assault as processing power increased. To address this issue, Triple DES was created, however, it was proven to be slower.

- It encrypts the plain text into ciphertext and sends it to the storage and inversely, decrypts the ciphertext in the storage and then transmits data to the host as plain text.



R	Key size
10	128
12	192
14	256

Fig 4.2.4: AES Algorithm

4.2.5 SECURED HASH ALGORITHM(SHA)

The Secure Hash Algorithm, or SHA, is a family of algorithms that includes the SHA 256 algorithm. The NSA and NIST collaborated to publish it in 2001 as a replacement for the SHA 1 family, which was gradually becoming less resistant to brute-force attacks.

The 256 in the name refers to the final hash digest value, meaning that regardless of the amount of plaintext or cleartext present, the hash value will always be 256 bits.

<u>The following are some of the SHA algorithm's distinguishing qualities:</u>

- Cleartext Length: The cleartext should be no longer than 264 bits. To keep the digest as random as possible, the size must be in the comparison area.
- Digest Length: The length of the hash digest for the SHA-256 algorithm should be 256 bits, for the SHA-512 algorithm 512 bits, and so forth. Larger digests typically imply a lot more calculations at the expense of performance and storage.
- Irreversible: All hash functions, including the SHA 256, are intended to be irreversible. If you already have the digest, you shouldn't receive the plaintext,

and if you run the digest through the hash function once more, it shouldn't return the original value.

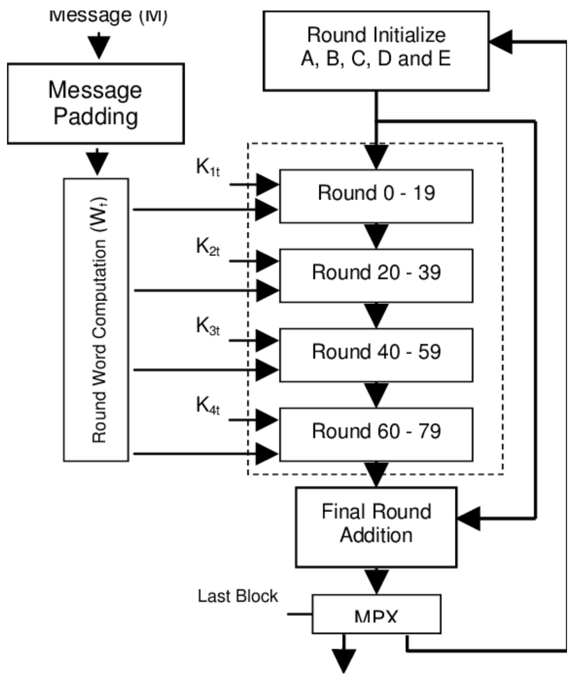


Fig 4.2.5 SHA Algorithm

Steps in SHA-256 Algorithm:

- Padding Bits
- Padding Length
- Initializing the Buffers
- Compression Functions
- Output

4.3 ARCHITECTURE / OVERALL DESIGN OF THE PROPOSED SYSTEM

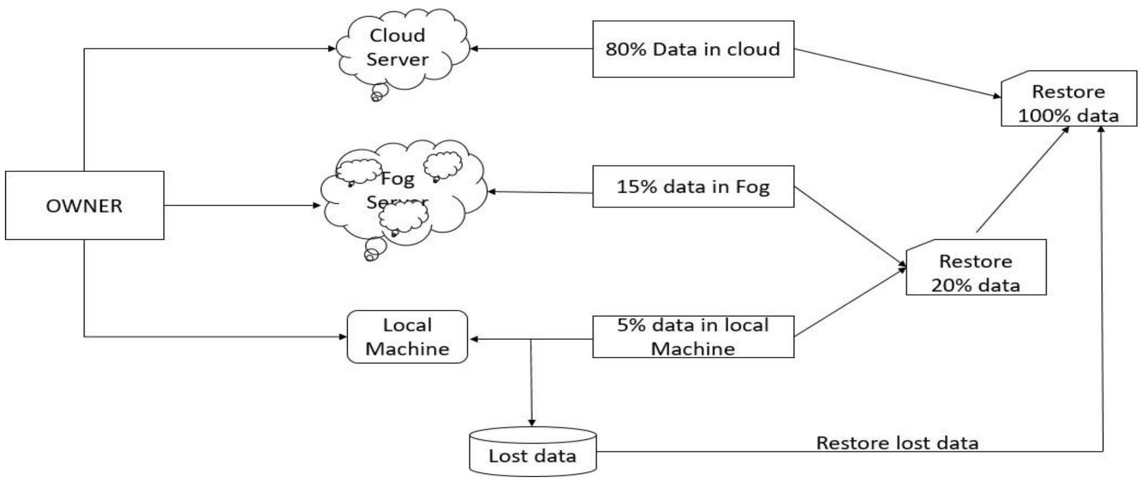


Fig 4.3.1: System Architecture-1

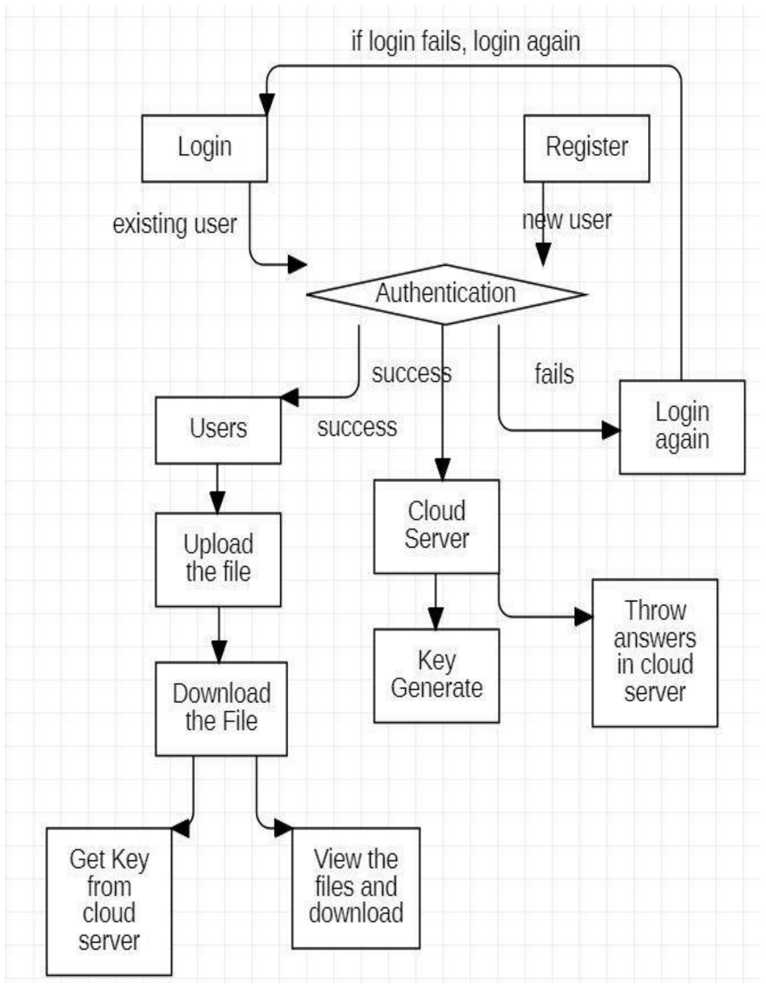


Fig 4.3.2: System Architecture-2

The block diagram of the proposed system has been shown in the above figures. We create a three-layer fog computing-based storage framework. The suggested architecture may both fully utilize cloud storage and safeguard data privacy. The Hash-Solomon coding method is being used in this instance to partition data into various pieces. We lose the data information if one data component is absent. We safeguard the data information in this framework using bucket concept-based algorithms, which can then demonstrate the security and effectiveness of our plan. Additionally, this technique can determine the distribution proportion of data kept in the cloud (80%), fog (15%), and local machine (5%), all based on computational intelligence. The framework can utilize all cloud storage while preserving data privacy. In this case, cloud computing has garnered a lot of interest from several societal sectors. The three-layer cloud storage system stores data in three separate portions. We lose the data information if one data component is absent. Algorithms based on the bucket notion are used in the suggested framework. In our system, we employ the bucket concept to cut down on data loss and speed up processing. The Bose-Chaudhuri-Hocquenghem (BCH) code algorithm is what we are employing. Highly flexible. There is little redundancy in BCH codes, which are employed in many communications applications.

4.4 DESCRIPTION OF SOFTWARE FOR IMPLEMENTATION AND TESTING PLAN OF THE PROPOSED MODEL/SYSTEM

4.4.1 VISUAL STUDIO

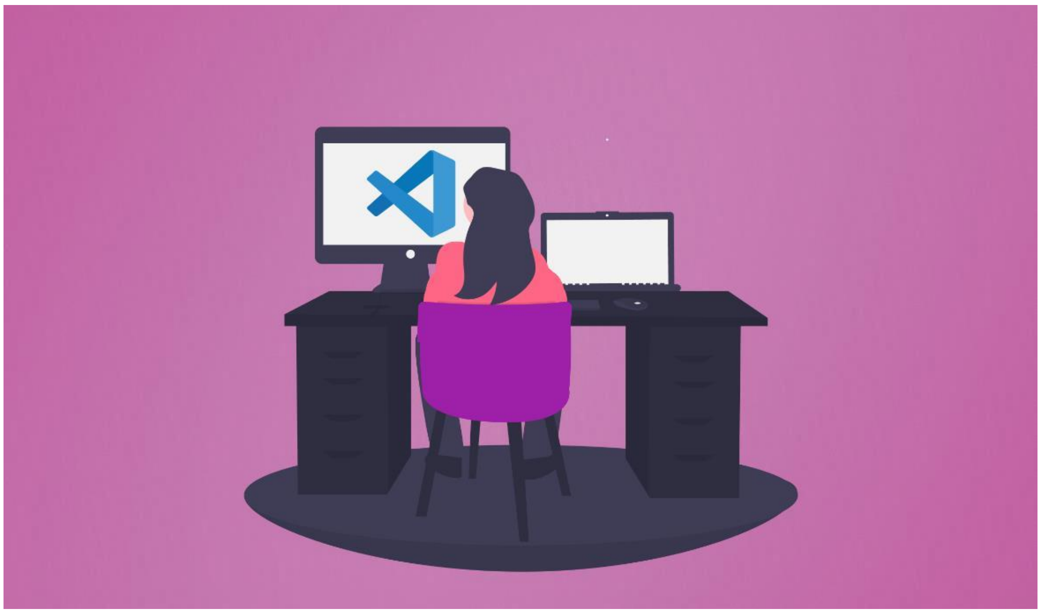
On your desktop, Visual Studio Code is a quick yet effective source code editor that runs on Windows, macOS, and Linux. It contains support for JavaScript, TypeScript, and Node.js built-in, as well as a robust ecosystem of extensions for additional languages and runtimes (including C++, C#, Java, Python, PHP, Go, and.NET). Debugging support, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git are among the features. The theme, keyboard shortcuts, options, and extensions that offer more functionality can all be changed by users.

On April 29, 2015, Microsoft made its initial announcement of Visual Studio Code at the 2015 Build conference. Soon later, a project build was made available.

The MIT License was applied to the release of Visual Studio Code's source code on November 18, 2015, and it was made accessible on GitHub. Support for extensions was also declared. The public project phase of Visual Studio Code ended on April 14, 2016, and it was made available online. While Microsoft's releases are private freeware, the majority of Visual Studio Code's source code has been made available on GitHub under the permissive MIT License.

Microsoft's Visual Studio is an integrated development environment (IDE). Computer programmer, including as websites, web applications, online services, and mobile applications, are developed using it. Microsoft's software development platforms, including Windows Store, Windows Presentation Foundation, Windows API, and Windows Forms, are used by Visual Studio. Both native and managed code can be generated by it.

A code editor that supports code refactoring and IntelliSense (the code completion feature) is a feature of Visual Studio. Both a source-level debugger and a machine-level debugger can be used with the integrated debugger. A code profiler, designer for creating GUI apps, web designer, class designer, and database schema designer are further built-in tools.



It accepts plug-ins that increase functionality at almost every level, such as adding support for source control programmer (such as Subversion and Git) and new toolkits, such as editors and visual designers for programming languages with specialized functions, or toolkits for different stages of the software development lifecycle (like the Azure DevOps client: Team Explorer).

If a language-specific service is available, Visual Studio enables the code editor and debugger to support (to varied degrees) practically every programming language. It currently supports 36 distinct programming languages. C, C++, C++/CLI, Visual Basic.NET, C#, F#, JavaScript, TypeScript, XML, XSLT, HTML, and CSS are just a few of the built-in languages. Using plug-ins, more languages can be supported, including Python, Ruby, Node.js, and M, among others. In the past, Java (and J#) were supported.

4.4.1.1 Architecture of Visual Studio

Visual Studio does not support any programming language, solution, or tool intrinsically; instead, it allows the plugging of functionality coded as a VSPackage.

When installed, the functionality is available as a Service. The IDE provides three services: SVsSolution, which provides the ability to enumerate projects and solutions; SVsUIShell, which provides windowing and UI functionality (including tabs, toolbars, and tool windows); and SVsShell, which deals with registration of VSPackages. In addition, the IDE is also responsible for coordinating and enabling communication between services. All editors, designers, project types and other tools are implemented as VSPackages. Visual Studio uses COM to access the VSPackages. The Visual Studio SDK also includes the Managed Package Framework (MPF), which is a set of managed wrappers around the COM-interfaces that allow the Packages to be written in any CLI compliant language. However, MPF does not provide all the functionality exposed by the Visual Studio COM interfaces. The services can then be consumed for creation of other packages, which add functionality to the Visual Studio IDE.

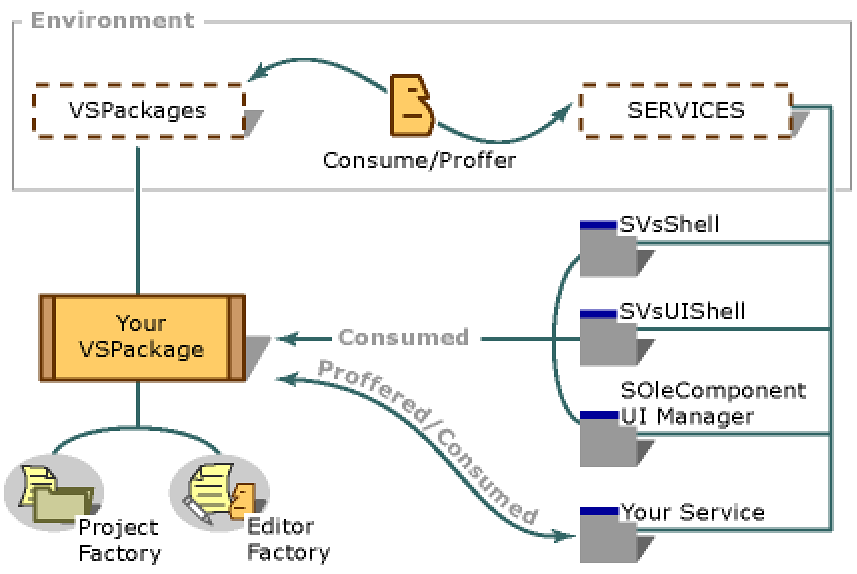


Fig 4.4.1.1 Visual Studio Architecture

Support for programming languages is added by using a specific VSPackage called a Language Service. A language service defines various interfaces which the VSPackage implementation can implement to add support for various functionalities. Functionalities that can be added this way include syntax coloring, statement completion, brace matching, parameter information tooltips, member lists, and error markers for background compilation. If the interface is implemented, the functionality will be available for the language. Language services are implemented on a per-language basis. The implementations can reuse code from the parser or the compiler for the language. Language services can be implemented either in native code or managed code.

For native code, either the native COM interfaces or the Babel Framework (part of Visual Studio SDK) can be used. For managed code, the MPF includes wrappers for writing managed language services.

Visual Studio does not include any source control support built in but it defines two alternative ways for source control systems to integrate with the IDE. A Source Control VSPackage can provide its own customized user interface. In contrast, a source control plugin using the MSSCCI (Microsoft Source Code Control Interface) provides a set of functions that are used to implement various source control functionality, with a standard Visual Studio user interface. MSSCCI was first used to integrate Visual SourceSafe with Visual Studio 6.0 but was later opened up via the Visual Studio SDK. Visual Studio .NET 2002 used MSSCCI 1.1, and Visual Studio .NET 2003 used MSSCCI 1.2. Visual Studio 2005, 2008, and 2010 use MSSCCI Version 1.3, which adds support for rename and delete propagation, as well as asynchronous opening.

Visual Studio supports running multiple instances of the environment (each with its own set of VSPackages). The instances use different registry hives (see MSDN's definition of the term "registry hive" in the sense used here) to store their configuration state and are differentiated by their AppId (Application ID). The instances are launched by an AppId-specific .exe that selects the AppId, sets the root hive, and launches the IDE. VSPackages registered for one AppId are integrated with other VSPackages for that AppId. The various product editions of Visual Studio are created using the different AppIds. The Visual Studio Express edition products are installed with their own AppIds, but the Standard, Professional, and Team Suite products share the same AppId. Consequently, one can install the Express editions side-by-side with other editions, unlike the other editions which update the same installation. The professional edition includes a superset of the VSPackages in the standard edition, and the team suite includes a superset of the VSPackages in both other editions. The AppId system is leveraged by the Visual Studio Shell in Visual Studio 2008.

4.4.1.2 Features of Visual Studio

1. Code editor

Visual Studio includes a code editor that supports syntax highlighting and code completion using IntelliSense for variables, functions, methods, loops, and LINQ queries. IntelliSense is supported for the included languages, as well as for XML, Cascading Style Sheets, and JavaScript when developing web sites and web applications. Autocomplete suggestions appear in a modeless list box over the code editor window, in proximity of the editing cursor. In Visual Studio 2008 onwards, it can be made temporarily semi-transparent to see the code obstructed by it. The code editor is used for all supported languages.

The Visual Studio Code Editor also supports setting bookmarks in code for quick navigation. Other navigational aids include collapsing code blocks and incremental search, in addition to normal text search and regex search. The code editor also includes a multi-item clipboard and a task list. The code editor supports code snippets, which are saved templates for repetitive code and can be inserted into code and customized for the project being worked on. A management tool for code snippets is built in as well. These tools are surfaced as floating windows which can be set to automatically hide when unused or docked to the side of the screen. The Visual Studio code editor also supports code refactoring including parameter reordering, variable and method renaming, interface extraction, and encapsulation of class members inside properties, among others. An online version of Visual Studio Code is available at Visual Studio Code.

2. Debugger

Visual Studio includes a debugger that works both as a source-level debugger and as a machine-level debugger. It works with both managed code as well as native code and can be used for debugging applications written in any language supported by Visual Studio. In addition, it can also attach to running processes, monitor, and debug those processes. If source code for the running process is available, it displays the code as it is being run. If source code is not available, it can show the disassembly. The Visual Studio debugger can also create memory dumps as well as load them later for debugging. Multi-threaded programs are also supported. The debugger can be configured to be launched when an application running outside the Visual Studio environment crashes.

The Visual Studio Debugger allows setting breakpoints (which allow execution to be stopped temporarily at a certain position) and watches (which monitor the values of variables as the execution progresses). Breakpoints can be conditional, meaning they get triggered when the condition is met. Code can be stepped over, i.e., run one line (of source code) at a time. It can either step into functions to debug inside it, or step over it, i.e., the execution of the function body isn't available for manual inspection. The debugger supports Edit and Continue, i.e., it allows code to be edited as it is being debugged. When debugging, if the mouse pointer hovers over any variable, its current value is displayed in a tooltip ("data tooltips"), where it can also be modified if desired. During coding, the Visual Studio debugger lets certain functions be invoked manually from the Immediate tool window. The parameters to the method are supplied at the Immediate window.

3. Designer

Visual Studio includes a host of visual designers to aid in the development of applications. These tools include:

Windows Forms Designer

The Windows Forms designer is used to build GUI applications using Windows Forms. Layout can be controlled by housing the controls inside other containers or locking them to the side of the form. Controls that display data (like textbox, list box and grid view) can be bound to data sources like databases or queries. Data-bound controls can be created by dragging items from the Data Sources window onto a design surface. The UI is linked with code using an event-driven programming model. The designer generates either C# or VB.NET code for the application.

WPF Designer

The WPF designer, codenamed Cider, was introduced with Visual Studio 2008. Like the Windows Forms designer it supports the drag and drop metaphor. It is used to author user interfaces targeting Windows Presentation Foundation. It supports all WPF functionality including data binding and automatic layout management. It generates XAML code for the UI. The generated XAML file is compatible with Microsoft Expression Design, the designer-oriented product. The XAML code is linked with code using a code-behind model.

Web designer/development

Visual Studio also includes a web-site editor and designer that allows web pages to be authored by dragging and dropping widgets. It is used for developing ASP.NET applications and supports HTML, CSS and JavaScript. It uses a code-behind model to link with ASP.NET code. From Visual Studio 2008 onwards, the layout engine used by the web designer is shared with the discontinued Expression Web. There is also ASP.NET MVC support for MVC technology as a separate download and ASP.NET Dynamic Data project available from Microsoft.

Class designer

The Class Designer is used to author and edit the classes (including its members and their access) using UML modeling. The Class Designer can generate C# and VB.NET code outlines for the classes and methods. It can also generate class diagrams from hand-written classes.

Data designer

The data designer can be used to graphically edit database schemas, including typed tables, primary and foreign keys and constraints. It can also be used to design queries from the graphical view.

Mapping designer

From Visual Studio 2008 onwards, the mapping designer is used by LINQ to SQL to design the mapping between database schemas and the classes that encapsulate the data. The new solution from ORM approach, ADO.NET Entity Framework, replaces and improves the old technology.

4.4.2 SQL SERVER

SQL (Structured Query Language) is used for managing data stored in a relational database management system (RDBMS) or for stream processing in a relational data stream management system, programmers utilize SQL, and RDSMS. It is especially helpful when managing structured data, or data that includes relationships between entities and variables.

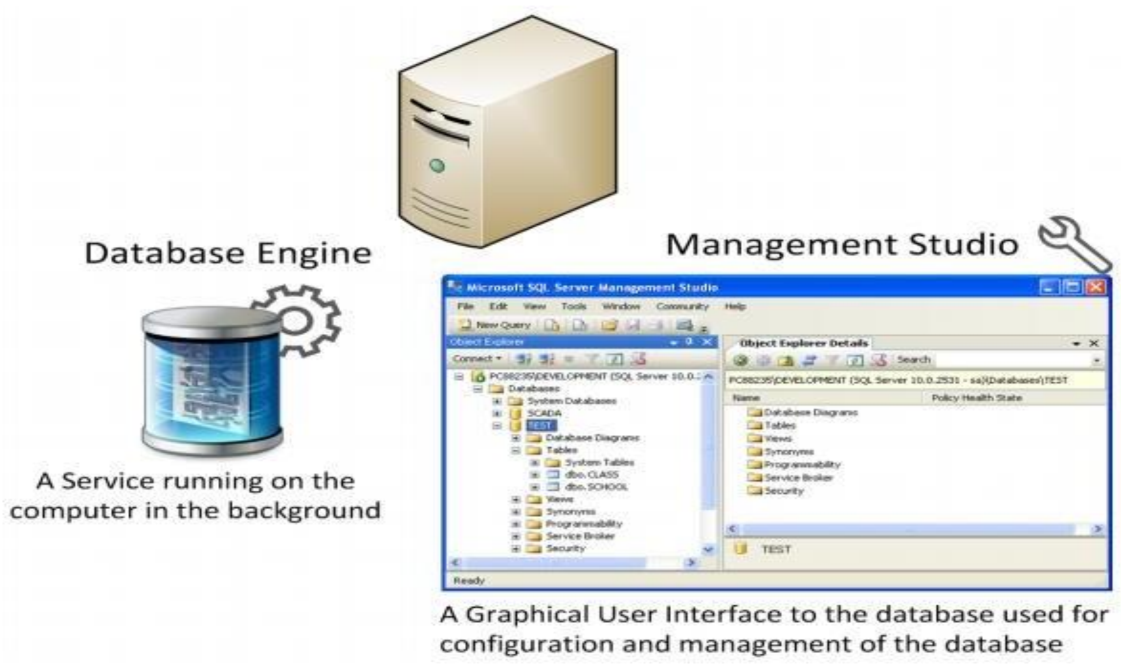

Database Engine
A Service running on the computer in the background
Management Studio
A Graphical User Interface to the database used for configuration and management of the database

Fig 4.4.2 SQL Server Interface

Data Query Language (DQL), Data Definition Language (DDL), Data Control Language (DCL), and Data Manipulation Language (DML) are some of the common sublanguages of SQL. SQL was originally based on relational algebra and tuple relational calculus. Data query, data manipulation (insert, update, and delete), data definition (schema development and change), and data access control are all included in the scope of SQL. Despite being primarily a declarative language (4GL), SQL also has procedural components.

Edgar F. Codd's relational model was one of the first commercial languages to be used. His seminal 1970 project, "A Relational Model of Data for Large Shared Data Banks," defined the model. It became the most used database language despite not entirely adhering to Codd's relational paradigm.

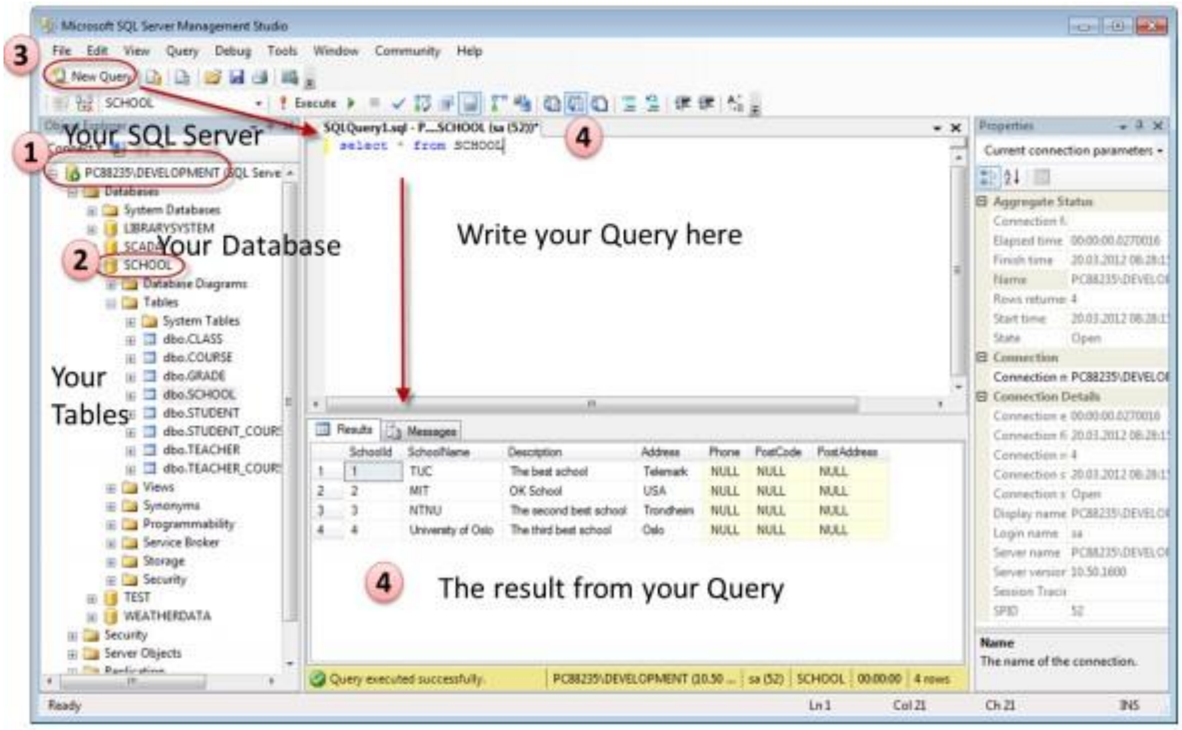
In 1986, the American National Standards Institute (ANSI) and the International Organization for Standardization (ISO) recognized SQL as a standard. Since then, a wider range of functionality has been added to the standard. Despite the fact that there are standards, the majority of SQL code needs to be modified at least slightly before being transferred to different database systems.

4.4.2.1 SQL SERVER MANAGEMENT STUDIO

SQL Server Management Studio is a GUI tool included with SQL Server for configuring, managing, and administering all components within Microsoft SQL Server. The tool includes both script editors and graphical tools that work with objects and features of the server. As mentioned earlier, version of SQL Server Management Studio is also available for SQL Server Express Edition, for which it is known as SQL Server Management Studio Express.

A central feature of SQL Server Management Studio is the Object Explorer, which allows the user to browse, select, and act upon any of the objects within the server. It can be used to visually observe and analyze query plans and optimize the database performance, among others. SQL Server Management Studio can also be used to create a new database, alter any existing database schema by adding or modifying tables and indexes, or analyze performance. It includes the query windows which provide a GUI based interface to write and execute queries.

When creating SQL commands and queries, the “Query Editor” (select “New Query” from the Toolbar) is used (shown in the figure above). With SQL and the “Query Editor” we can do almost everything with code, but sometimes it is also a good idea to use the different Designer tools in SQL to help us do the work without coding (so much).



```
select * from SCHOOL
```

	SchoolId	SchoolName	Description	Address	Phone	PostCode	PostAddress
1	1	TUC	The best school	Telemark	NULL	NULL	NULL
2	2	MIT	OK School	USA	NULL	NULL	NULL
3	3	NTNU	The second best school	Trondheim	NULL	NULL	NULL
4	4	University of Oslo	The third best school	Oslo	NULL	NULL	NULL

Query executed successfully. | PC88235\DEVELOPMENT (10.50 ... | sa (52) | SCHOOL | 00:00:00 | 4 rows

Fig 4.4.2.1 Microsoft SQL Server Management Studio

4.4.2.2 Create a new Database

It is quite simple to create a new database in Microsoft SQL Server. Just right-click on the “Databases” node and select “New Database…”

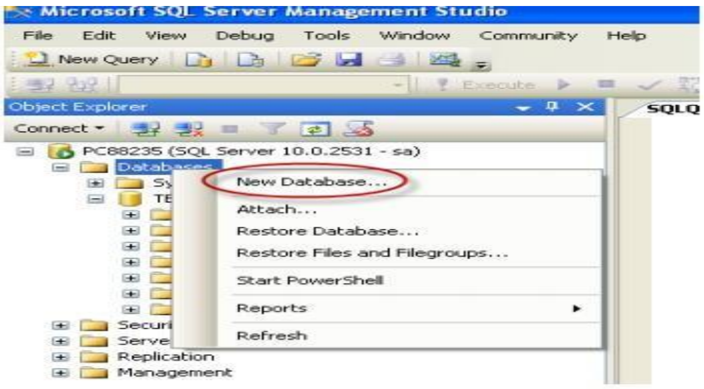


Fig 4.4.2.2 Database Creation

The OLAP Services feature available in SQL Server version 7.0 is now called SQL Server 2000 Analysis Services. The term OLAP Services has been replaced with the term Analysis Services. Analysis Services also includes a new data mining component. The Repository component available in SQL Server version 7.0 is now called Microsoft SQL Server 2000 Meta Data Services. References to the component now use the term Meta Data Services. The term repository is used only in reference to the repository engine within Meta Data Services

SQL-SERVER database consists of six type of objects,

They are,

1. TABLE
2. QUERY
3. FORM
4. REPORT
5. MACRO

Table

A database is a collection of data about a specific topic.

Views of Table

We can work with a table in two types,

1. Design View
2. Datasheet View

Design View

To build or modify the structure of a table we work in the table design view. We can specify what kind of data will be hold.

Datasheet View

To add, edit or analyses the data itself we work in tables datasheet view mode.

Query

A query is a question that must be asked the data. Access gathers data that answers the question from one or more table. The data that make up the answer is either dynast (if you edit it) or a snapshot (it cannot be edited). Each time we run query; we get latest information in the dynast. Access either displays the dynast or snapshot for us to view or perform an action on it, such as deleting or updating.

4.4.3 .NET

.NET is a free, open-source, cross-platform platform for developers that may be used to create a wide range of applications. With.NET this, you can create applications for the web, mobile, desktop, gaming, IoT, and more using a variety of languages, editors,

and libraries.
ASP.NET Core web apps, command-line/console apps, libraries, and Universal Windows Platform apps are the four cross-platform use cases that.NET enables. Windows Forms and Windows Presentation Foundation (WPF), which produce the typical GUI for desktop software on Windows, were not implemented prior to.NET Core 3.0. However, Windows Forms, WPF, and Universal Windows Platform are now supported by.NET Core 3. (UWP).
NuGet packages can be used with.NET. .NET uses its package manager to download updates, as opposed to the.NET Framework, which is updated through Windows Update. But starting in December 2020, .NET upgrades were also made available through Windows Update.
The two main.NET components are CoreCLR and CoreFX, which are equivalent to the.NET Framework's Common Language Infrastructure (CLI) implementation's Common Language Runtime (CLR) and Framework Class Library (FCL). As a just-in-time compiler known as RyuJIT, CoreCLR serves as a complete runtime and virtual machine for managed execution of CLI applications as a CLI implementation of a Virtual Execution System (VES). Additionally, CoreRT, the.NET Native runtime optimized for use in AOT-generated native binaries, is part of .NET Core. In addition to sharing a portion of the.NET Framework APIs with other CLI implementations of the core Standard Libraries, CoreFX also has its own APIs that are independent of the.NET Framework. UWP makes use of a.NET library variation.
Developer services like compilation and package management are offered by the.NET command-line interface, which also offers an execution entry point for operating systems.

Objectives of .Net Framework

1. To provide a consistent object-oriented programming environment whether object codes are stored and executed locally on Internet-distributed, or executed remotely.
2. To provide a code-execution environment to minimize software deployment and guarantee the safe execution of code.
3. Eliminates performance problems.

There are different types of applications, such as Windows-based applications and Web-based applications

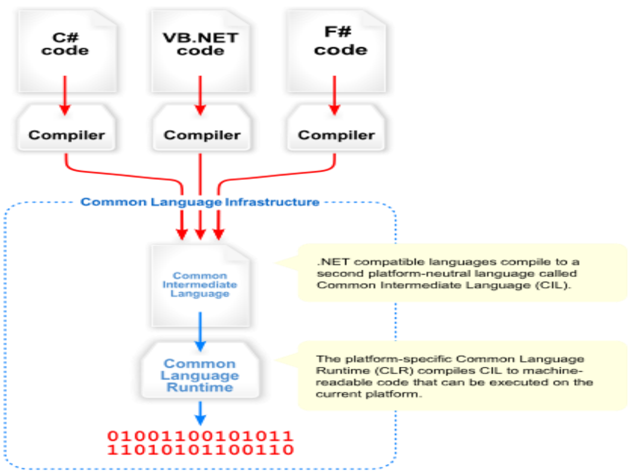

C# code
VB.NET code
F# code
Compiler
Compiler
Compiler
Common Language Infrastructure
Common Intermediate Language
.NET compatible languages compile to a second platform-neutral language called Common Intermediate Language (CIL).
Common Language Runtime
The platform-specific Common Language Runtime (CLR) compiles CIL to machine-readable code that can be executed on the current platform.
01001100101011
11010101100110

Fig 4.4.3: .NET Architecture

4.4.3.1 THE .NET FRAMEWORK

The .NET Framework has two main parts:

1. The Common Language Runtime (CLR).
2. A hierarchical set of class libraries.

The CLR is described as the “execution engine” of .NET. It provides the environment within which programs run. The most important features are

- Conversion from a low-level assembler-style language, called Intermediate Language (IL), into code native to the platform being executed on.
- Memory management, notably including garbage collection.
- Checking and enforcing security restrictions on the running code.
- Loading and executing programs, with version control and other such features.
- The following features of the .NET framework are also worth description:

Managed Code

The code that targets .NET, and which contains certain extra Information - “metadata” - to describe itself. Whilst both managed and unmanaged code can run in the runtime, only managed code contains the information that allows the CLR to guarantee, for instance, safe execution and interoperability.

Managed Data

With Managed Code comes Managed Data. CLR provides memory allocation and Deal location facilities, and garbage collection. Some .NET languages use Managed Data by default, such as C#, Visual Basic.NET and JScript.NET, whereas others, namely C++, do not. Targeting CLR can, depending on the language you're using, impose certain constraints on the features available. As with managed and unmanaged code, one can have both managed and unmanaged data in .NET applications - data that doesn't get garbage collected but instead is looked after by unmanaged code.

Common Type System

The CLR uses something called the Common Type System (CTS) to strictly enforce type-safety. This ensures that all classes are compatible with each other, by describing types in a common way. CTS define how types work within the runtime, which enables types in one language to interoperate with types in another language, including cross-language exception handling. As well as ensuring that types are only used in appropriate ways, the runtime also ensures that code doesn't attempt to access memory that hasn't been allocated to it.

Common Language Specification

The CLR provides built-in support for language interoperability. To ensure that you can develop managed code that can be fully used by developers using any programming language, a set of language features and rules for using them called the Common Language Specification (CLS) has been defined. Components that follow these rules and expose only CLS features are considered CLS-compliant.

4.4.3.2 FEATURES OF .NET

C#.NET is also compliant with CLS (Common Language Specification) and supports structured exception handling. CLS is set of rules and constructs that are supported by the CLR (Common Language Runtime). CLR is the runtime environment provided by the .NET Framework; it manages the execution of the code and also makes the development process easier by providing services C#.NET is a CLS-compliant language. Any objects, classes, or components that created in C#.NET can be used in any other CLS-compliant language. In addition, we can use objects, classes, and components created in other CLS-compliant languages in C#.NET .The use of

CLS ensures complete interoperability among applications, regardless of the languages used to create the application.

Constructors and Destructors

Constructors are used to initialize objects, whereas destructors are used to destroy them. In other words, destructors are used to release the resources allocated to the object. In C#.NET the sub finalize procedure is available. The sub finalize procedure is used to complete the tasks that must be performed when an object is destroyed. The sub finalize procedure is called automatically when an object is destroyed. In addition, the sub finalize procedure can be called only from the class it belongs to or from derived classes.

Garbage Collection

Garbage Collection is another new feature in C#.NET. The .NET Framework monitors allocated resources, such as objects and variables. In addition, the .NET Framework automatically releases memory for reuse by destroying objects that are no longer in use. In C#.NET, the garbage collector checks for the objects that are not currently in use by applications. When the garbage collector comes across an object that is marked for garbage collection, it releases the memory occupied by the object.

Overloading

Overloading is another feature in C#. Overloading enables us to define multiple procedures with the same name, where each procedure has a different set of arguments. Besides using overloading for procedures, we can use it for constructors and properties in a class.

Multithreading

C#.NET also supports multithreading. An application that supports multithreading can handle multiple tasks simultaneously, we can use multithreading to decrease the time taken by an application to respond to user interaction.

Structured Exception Handling

C#.NET supports structured handling, which enables us to detect and remove errors at runtime. In C#.NET, we need to use Try…Catch…Finally statements to create exception handlers. Using Try…Catch…Finally statements, we can create robust and effective exception handlers to improve the performance of our application.

4.4.3.3 The .Net Framework

The .NET Framework is a new computing platform that simplifies application development in the highly distributed environment of the Internet.

ASP.NET XML WEB SERVICES	Windows Forms
Base Class Libraries	
Common Language Runtime	
Operating System	

Fig 4.4.3.3 .Net Framework

Objectives of .Net Framework

1. To provide a consistent object-oriented programming environment whether object codes are stored and executed locally on Internet-distributed, or executed remotely.
2. To provide a code-execution environment to minimizes software deployment and guarantees safe execution of code.
3. Eliminates the performance problems.

There are different types of application, such as Windows-based applications and Web-based applications.

4.5 PROJECT MANAGEMENT PLAN

MODULES

- Login
- Registration
- Storage Scheme
- Recovery Scheme
- Download Scheme

4.5.1 Login Module

When a user opens the website, this is the first action that appears. In order to access the website, the user must input a valid phone number and a password that they created upon enrolling. If the user's information matches the data in the database table, the user is able to successfully log into the website; otherwise, a message indicating that the login attempt failed is displayed, and the user must enter their information correctly again. For the purpose of registering new users, a link to the registered activity is also supplied and also an OTP verification code is also generated and sent to the user’s registered mobile number or Email address. Thus, we ensure a secure platform for the user to upload their data.

4.5.2 Registration Module

Before logging in, a new user must first register on the website. The registered activity is opened by clicking the register button in the login action. When registering, a new user must provide their full name, password, and phone number. For confirmation, the user must reenter their password in the confirm password textbox. When a person fills out all of the text boxes when they click the register button, the data is sent to the database and they are then redirected to the login process. Then, to access the website, a registered user must log in. All textboxes have validations applied to ensure that the website functions properly. Each textbox, whether it be for a name, contact information, password, or confirm password, must have information in it in order for you to register. The app will notify users that each textbox needs to have information in it if any of them is empty. For registration to be successful, the information in the password and confirm password sections must also match. Another need is that the contact number be a genuine one with 10 digits. Registration will fail if any of these checks are met, at which point the user must re-register. When a field is empty, a message will be displayed on the website. If all of this information is accurate, the user will be sent to the login page so they can access the website.

4.5.3 Storage Module

The user can upload their data to three separate storage servers with this module. After the data is posted to the cloud, the owner no longer has control over it. The original data are encrypted into three separate layers in this module. Before being stored in the cloud, the data in each layer can be encrypted using a variety of cryptographic algorithms and encryption keys. Cloud servers, fog servers, and local servers are storage servers. 80% of the data is stored on cloud servers. 15% of the data on the fog server is stored sensitively. 5% of the data is stored on a local server.

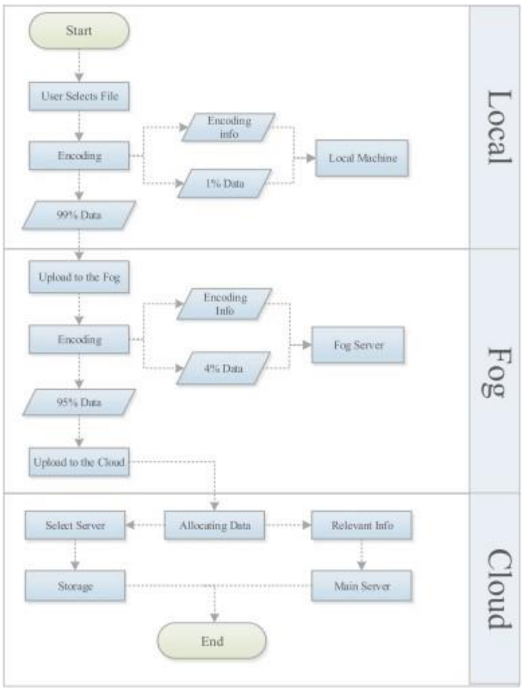


Fig 4.5.3: Storage Procedure

4.5.4 Recovery Module

This module allows the user to retrieve their files from three by employing the BCH algorithm, if the storage server's data will be matched with these three layers of data if the information was compromised in any way, it was kept in the bucket. The user can quickly retrieve it from the bucket of the layer's framework.

4.5.5 Download Module

The process is outlined in when a user wishes to download a file from a cloud server. The user's request is first received by the cloud server, which then integrates the data across various distributed servers. 95% of the data is sent to the fog server by the cloud server after integration. The data is then transferred from the cloud server to the fog server.

We can retrieve 99% of the data by combining it with the 4% of the fog server's data blocks and the encoding details. The user is then sent 99% of the data via the fog server. Third, the user gets the information from the fog server. By repeating the processes, the user can obtain all the data.

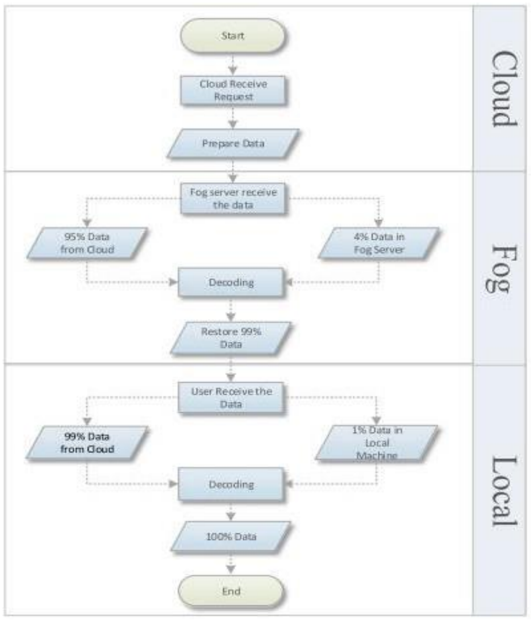


Fig 4.5.5: Download Procedure

CHAPTER 5

SYSTEM ANALYSIS

5.1 PURPOSE

We design a three-layer storage framework based on fog computing. The proposed framework can both take full advantage of cloud storage and protect the privacy of data. Here we are using Hash-Solomon code algorithm is designed to divide data into different parts. If the one data part missing, we lost the data information. In this framework we are using bucket concept-based algorithms and secure the data information and then it can show the security and efficiency in our scheme. Moreover, based on computational intelligence, this algorithm can compute the distribution proportion stored in cloud 80% of data, and fog 15% of data, and local machine 5% of data, respectively. The framework can take full of cloud storage and protect the privacy of data. Here the cloud computing has attracted great attention from different sector of society. The three-layer cloud storage stores in to the three different parts of data parts. If the one data part missing, we lost the data information. In this proposed framework using the bucket concept-based algorithms. In our system we using a bucket concept so reduce the data wastages and reduce the process timings. We are using a BCH (Bose–Chaudhuri–Hocquenghem) code algorithm. It’s High flexible. BCH code are used in many communications’ application and low amount of redundancy.

5.2 SCOPE

A Three-Layer Privacy Preserving Cloud Storage Scheme Based on Computational Intelligence in Fog Computing**.** The privacy preservation is our focus, some active attacks are beyond the scope of this work. The three-layer cloud storage stores in to the three different parts of data parts. If the one data part missing, we lost the data information. In this proposed framework using the bucket concept-based algorithms. We are using a BCH code algorithm. It’s High flexible.

5.3 EXISTING SYSTEM

- Recent years witness the development of cloud computing technology. With the explosive growth of unstructured data, cloud storage technology gets more attention and better development.

- The computer technology has developed rapidly. Cloud computing has gradually matured through so many people efforts.

- In current storage schema, the user’s data is totally stored in cloud servers. If the user loses their right of control on data and face privacy risk.

- The privacy protection schemes are usually based on encryption technology. These kinds of methods cannot effectively resist attack from the inside of cloud server.

5.3.1 Existing System Dis-advantages:

- Changes in the understanding of risk as a result of extending the datacenter into the cloud.
- Low latency and location awareness.

5.4 PROPOSED SYSTEM

- The framework can take full of cloud storage and protect the privacy of data.
- Here the cloud computing has attracted great attention from different sector of society.
- The three-layer cloud storage stores in to the three different parts of data parts. If the one data part missing, we lost the data information. In this proposed framework using the bucket concept-based algorithms.

5.4.1 Proposed System Advantages:

- In our system we using a bucket concept so reduce the data wastages and reduce the process timings.
- We are using a BCH (Bose–Chaudhuri–Hocquenghem) code algorithm. It’s High flexible.
- BCH code are used in many communications’ application and low amount of redundancy.

CHAPTER 6
SYSTEM DESIGN

6.1 INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

6.1.1 Input Design Objectives:

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.
2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.
3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus, the objective of input design is to create an input layout that is easy to follow.

6.2 OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

6.2.2 Output Design Objectives:

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.
2. Select methods for presenting information.
3. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

- ❖ Convey information about past activities, current status or projections of the
- ❖ Future.
- ❖ Signal important events, opportunities, problems, or warnings.
- ❖ Trigger an action.
- ❖ Confirm an action.

CHAPTER 7

SYSTEM TESTING

7.1 OVERVIEW

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

7.2 TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g., components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input	: identified classes of valid input must be accepted.
Invalid Input	: identified classes of invalid input must be rejected.
Functions	: identified functions must be exercised.
Output	: identified classes of application outputs must be exercised.
Systems/Procedures:	interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure, and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

7.3 Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

7.3.1 Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

7.3.2 Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

7.4 REQUIREMENT ANALYSIS

Requirement analysis, also called requirement engineering, is the process of determining user expectations for a new modified product. It encompasses the tasks that determine the need for analysing, documenting, validating and managing software or system requirements. The requirements should be documentable, actionable, measurable, testable and traceable related to identified business needs or opportunities and define to a level of detail, sufficient for system design.

7.5 FUNCTIONAL REQUIREMENTS

It is a technical specification requirement for the software products. It is the first step in the requirement analysis process which lists the requirements of particular software systems including functional, performance and security requirements. The function of the system depends mainly on the quality hardware used to run the software with given functionality.

Usability

It specifies how easy the system must be use. It is easy to ask queries in any format which is short or long, porter stemming algorithm stimulates the desired response for user.

Robustness

It refers to a program that performs well not only under ordinary conditions but also under unusual conditions. It is the ability of the user to cope with errors for irrelevant queries during execution.

Security

The state of providing protected access to resource is security. The system provides good security and unauthorized users cannot access the system there by providing high security.

Reliability

It is the probability of how often the software fails. The measurement is often expressed in MTBF (Mean Time Between Failures). The requirement is needed in order to ensure that the processes work correctly and completely without being aborted. It can handle any load and survive and survive and even capable of working around any failure.

Compatibility

It is supported by version above all web browsers. Using any web servers like localhost makes the system real-time experience.

Flexibility

The flexibility of the project is provided in such a way that is has the ability to run on different environments being executed by different users.

Safety

Safety is a measure taken to prevent trouble. Every query is processed in a secured manner without letting others to know one’s personal information.

7.6 NON- FUNCTIONAL REQUIREMENTS

Portability

It is the usability of the same software in different environments. The project can be run in any operating system.

Performance

These requirements determine the resources required, time interval, throughput and everything that deals with the performance of the system.

Accuracy

The result of the requesting query is very accurate and high speed of retrieving information. The degree of security provided by the system is high and effective.

Maintainability

Project is simple as further updates can be easily done without affecting its stability. Maintainability basically defines that how easy it is to maintain the system. It means that how easy it is to maintain the system, analyze, change and test the application. Maintainability of this project is simple as further updates can be easily done without affecting its stability.

CHAPTER 8
CONCLUSION

8.1 CONCLUSION

The development of cloud computing brings us a lot of benefits. Cloud storage is a convenient technology which helps users to expand their storage capacity. However, cloud storage also causes a series of secure problems. When using cloud storage, users do not actually control the physical storage of their data and it results in the separation of ownership and management of data. In order to solve the problem of privacy protection in cloud storage, we propose a TLS framework based on fog computing model and design a BCH Code algorithm. Through the theoretical safety analysis, the scheme is proved to be feasible. By allocating the ratio of data blocks stored in different servers reasonably, we can ensure the privacy of data in each server. On another hand, cracking the encoding matrix is impossible theoretically. Besides, using hash transformation can protect the fragmentary information. Through the experiment test, this scheme can efficiently complete encoding and decoding without influence of the cloud storage efficiency. Furthermore, we design a reasonable comprehensive efficiency index, in order to achieve the maximum efficiency, and we also find that the Cauchy matrix is more efficient in coding process.

8.2 FUTURE ENHANCEMENT

In future, we are going to implement real-time cloud in this concept like amazon web services for additional security.

REFERENCES

[1] P. Mell and T. Grance, “The NIST definition of cloud computing,” Nat. Inst. Stand. Technol., vol. 53, no. 6, pp. 50–50, 2009.

[2] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, “A survey of mobile cloud computing: Architecture, applications, and approaches,” Wireless Commun. Mobile Comput., vol. 13, no. 18, pp. 1587–1611, 2013.

[3] J. Chase, R. Kaewpuang, W. Yonggang, and D. Niyato, “Joint virtual machine and bandwidth allocation in software defined network (sdn) and cloud computing environments,” in Proc. IEEE Int. Conf. Commun., 2014, pp. 2969–2974.

[4] H. Li, W. Sun, F. Li, and B. Wang, “Secure and privacy-preserving data storage service in public cloud,” J. Comput. Res. Develop., vol. 51, no. 7, pp. 1397–1409, 2014.

[5] Y. Li, T.Wang, G.Wang, J. Liang, and H. Chen, “Efficient data collection in sensor-cloud system with multiple mobile sinks,” in Proc. Adv. Serv. Comput., 10th Asia-Pac. Serv. Comput. Conf., 2016, pp. 130–143.

[6] L. Xiao, Q. Li, and J. Liu, “Survey on secure cloud storage,” J. Data Acquis. Process., vol. 31, no. 3, pp. 464–472, 2016.

[7] R. J. McEliece and D. V. Sarwate, “On sharing secrets and reed-solomon codes,” Commun. ACM, vol. 24, no. 9, pp. 583–584, 1981.

[8] J. S. Plank, “T1: Erasure codes for storage applications,” in Proc. 4th USENIX Conf. File Storage Technol., 2005, pp. 1–74.

[9] R. Kulkarni, A. Forster, and G. Venayagamoorthy, “Computational intelligence

in wireless sensor networks: A survey," IEEE Commun. Surv. Tuts.,
vol. 13, no. 1, pp. 68–96, First Quarter 2011.

[10] Z. Xia, X. Wang, L. Zhang, Z. Qin, X. Sun, and K. Ren, "A privacypreserving
and copy-deterrence content-based image retrieval scheme in
cloud computing," IEEE Trans. Inf. Forensics Security, vol. 11, no. 11,
pp. 2594–2608, Nov. 2016.

[11] J. Shen, D. Liu, J. Shen, Q. Liu, and X. Sun, "A secure cloud-assisted
urban data sharing framework for ubiquitous-cities," Pervasive Mobile
Comput., vol. 41, pp. 219–230, 2017.

[12] Z. Fu, F. Huang, K. Ren, J.Weng, and C.Wang, "Privacy-preserving smart
semantic search based on conceptual graphs over encrypted outsourced
data," IEEE Trans. Inf. Forensics Security, vol. 12, no. 8, pp. 1874–1884,
Aug. 2017.

[13] J. Hou, C. Piao, and T. Fan, "Privacy preservation cloud storage architecture
research," J. Hebei Acad. Sci., vol. 30, no. 2, pp. 45–48, 2013.

APPENDIX

A. Source Code

Download Page Code

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.SqlClient;
using System.IO;
using System.Linq;
using System.Text;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class download : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
      //  Label1.Text = Convert.ToString(Session["name"]);
        SqlConnection con = new SqlConnection("Data Source=SAINARU;Initial
Catalog=cloud1;Integrated Security=True");
        string strSQL = "Select * from file_m where uname='" +
Session["mailid"].ToString() + "'";
        SqlDataAdapter dt = new SqlDataAdapter(strSQL, con);
        DataSet ds = new DataSet();
        dt.Fill(ds, "uname");
        con.Close();
        GridView1.DataSource = ds;
        GridView1.DataBind();
    }
    protected void GridView1_SelectedIndexChanged (object sender, EventArgs e)
    {
        string f_name = Convert.ToString(GridView1.SelectedRow.Cells[1].Text);
        string dt,dt1,dt2;
        var fileStream = new FileStream(Server.MapPath("~/cloud/"+f_name),
FileMode.Open, FileAccess.Read);
        using (var streamReader = new StreamReader(fileStream, Encoding.UTF8))
        {
            dt = streamReader.ReadToEnd();
        }
        var fileStream1 = new FileStream(Server.MapPath("~/cloud1/" + f_name),
FileMode.Open, FileAccess.Read);
        using (var streamReader = new StreamReader(fileStream1, Encoding.UTF8))
        {
            dt1 = streamReader.ReadToEnd();
        }
```

Home Page HTML Code

```
<%@ Master Language="C#" AutoEventWireup="true" CodeFile="Home.master.cs"
Inherits="Home" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
          <charset="utf-8">
          <meta name="viewport" content="width=device-width, initial-scale=1">
          <title>Lumino UI Elements</title>
          <link href="css/bootstrap.min.css" rel="stylesheet">
          <link href="css/font-awesome.min.css" rel="stylesheet">
          <link href="css/datepicker3.css" rel="stylesheet">
          <link href="css/styles.css" rel="stylesheet">
          <!--Custom Font-->
          <link
href="https://fonts.googleapis.com/css?family=Montserrat:300,300i,400,400i,500,500i
,600,600i,700,700i" rel="stylesheet">
          <!--[if lt IE 9]>
```

```
		<script src="js/html5shiv.js"></script>
		<script src="js/respond.min.js"></script>
		<![endif]-->
</head>
<body>
		<nav class="navbar navbar-custom navbar-fixed-top" role="navigation">
				<div class="container-fluid">
						<div class="navbar-header">
								<button type="button" class="navbar-toggle collapsed"
data-toggle="collapse" data-target="#sidebar-collapse"><span class="sr-
only">Toggle navigation</span>
										<span class="icon-bar"></span>
										<span class="icon-bar"></span>
										<span class="icon-bar"></span></button>
								<a class="navbar-brand"
href="#"><span>Cloud</span>Admin</a>

						</div>
				</div><!-- /.container-fluid -->
		</nav>
		<div id="sidebar-collapse" class="col-sm-3 col-lg-2 sidebar">
				<div class="profile-sidebar">
						<div class="profile-userpic">
								<img src="http://placehold.it/50/30a5ff/fff" class="img-
responsive" alt="">
						</div>
						<div class="profile-usertitle">
								<div class="profile-usertitle-name">
	<asp:Label ID="Label1" runat="server" Text="Label"></asp:Label></div>
								<div class="profile-usertitle-status"><span
class="indicator label-success"></span>Online</div>
						</div>
						<div class="clear"></div>
```

```
                    </div>
                    <div class="divider"></div>
                             <div class="form-group">
                                        <input type="text" class="form-control"
placeholder="Search">
                              </div>
                    <ul class="nav menu">
                              <li><a href="Default.aspx"><em class="fa fa-
dashboard">&nbsp;</em> Home</a></li>
                              <li><a href="upload.aspx"><em class="fa fa-
calendar">&nbsp;</em> Upload</a></li>
                              <li><a href="view.aspx"><em class="fa fa-bar-
chart">&nbsp;</em> View</a></li>
                              <li><a href="download.aspx"><em class="fa fa-toggle-
off">&nbsp;</em> Download</a></li>
                              <li><a href="Login.aspx"><em class="fa fa-power-
off">&nbsp;</em> Logout</a></li>
                    </ul>
          </div><!--/.sidebar-->
          <div class="col-sm-9 col-sm-offset-3 col-lg-10 col-lg-offset-2 main">
                    <div class="row">
                              <ol class="breadcrumb">
                                        <li><a href="#">
                                                  <em class="fa fa-home"></em>
                                        </a></li>
                                        <li class="active">Forms</li>
                              </ol>
                    </div><!--/.row-->

                    <div class="row">
                              <div class="col-lg-12">
                                        <h1 class="page-header"></h1>
                              </div>
```

```
                </div><!--/.row-->
                <div class="row">
                        <div class="col-lg-12">
                                <div class="panel panel-default">
                                        <div class="panel-heading">Forms</div>
                                        <div class="panel-body">
                                                <div class="col-md-6">

                                                 <form id="form1" runat="server">
   <div>
      <asp:ContentPlaceHolder id="ContentPlaceHolder1" runat="server">

      </asp:ContentPlaceHolder>
   </div>
   </form>
                                                                </div>


                                                </div>
                                        </div>
                                </div><!-- /.panel-->
                        </div><!-- /.col-->
                        <div class="col-sm-12">
                                <p class="back-link"> <a href="#">Cloud</a></p>
                        </div>
                </div><!-- /.row -->
        </div><!--/.main-->
<script src="js/jquery-1.11.1.min.js"></script>
        <script src="js/bootstrap.min.js"></script>
        <script src="js/chart.min.js"></script>
        <script src="js/chart-data.js"></script>
        <script src="js/easypiechart.js"></script>
        <script src="js/easypiechart-data.js"></script>
        <script src="js/bootstrap-datepicker.js"></script>
        <script src="js/custom.js"></script>
</body>
</html>
```

Register Page Code

```
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
public partial class Register : System.Web.UI.Page
{
    SqlConnection con;
    SqlDataReader rs;
    string conn = "Data Source=SAINARU;Initial Catalog=cloud1;Integrated
Security=True";
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    protected void Button2_Click(object sender, EventArgs e)
    {
        if (TextBox1.Text == "" || TextBox2.Text == "" || TextBox1.Text == "" ||
TextBox2.Text == "" || TextBox2.Text == "")
        {
            MsgBox.Show("Fill All Datas");
        }
        else if (TextBox2.Text != TextBox5.Text)
        {
            MsgBox.Show("Password not matching");
        }
        else
        {
            con = new SqlConnection(conn);
            string s1 = "insert into login1 values('" + TextBox1.Text + "','" + TextBox3.Text
+ "','" + TextBox4.Text + "','" + TextBox2.Text + "')";
            con.Open();
            SqlCommand cmd = new SqlCommand(s1, con);
            cmd.ExecuteNonQuery();
            TextBox1.Text = "";
            TextBox2.Text = "";
            TextBox3.Text = "";
            TextBox4.Text = "";
            TextBox5.Text = "";
            Response.Redirect("Login.aspx");
        }
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        Response.Redirect("Login.aspx");
    }
}
```

Upload Page Code

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Security.Cryptography;
using System.Text;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data.SqlClient;

public partial class upload : System.Web.UI.Page
{
    public static string dt;
    public static string conn = "Data Source=SAINARU;Initial
Catalog=cloud1;Integrated Security=True";
    SqlConnection con;
    SqlCommand  cmd;
    SqlDataReader rs;
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        if (FileUpload1.HasFile)
        {
            try
            {
                if (FileUpload1.PostedFile.ContentType == "text/plain")
                {
                    if (FileUpload1.PostedFile.ContentLength < 102400)
                    {
                        string filename = Path.GetFileName(FileUpload1.FileName);
                        FileUpload1.SaveAs(Server.MapPath("~/temp1/") + filename);
                        var fileStream = new FileStream(Server.MapPath("~/temp1/") +
filename, FileMode.Open, FileAccess.Read);
                        using (var streamReader = new StreamReader(fileStream,
Encoding.UTF8))
                        {
                            dt = streamReader.ReadToEnd();
                        }
```

```
                      string path = Server.MapPath("~/temp1/") + filename;
                      FileInfo file = new FileInfo(path);
                      if (file.Exists)//check file exsit or not
                      {
                          file.Delete();

                      }
                      Cipher c = new Cipher();
                     string c_text= Convert.ToString( c.Encrypt(dt, "p@SSword"));
                     int a = c_text.Length;
                     int x =(int) a / 3;
                     string sub = c_text.Substring(0, x);
                     string sub1 = c_text.Substring(x, x);
                     int x1 = a-(x + x);
                     string sub2 = c_text.Substring(x+x, x1);
                     string cipherText1 = CryptorEngine.Encrypt(sub, true);
                     string cipherText2 = CryptorEngine.Encrypt(sub1, true);
                     string cipherText3 = CryptorEngine.Encrypt(sub2, true);
                     File.WriteAllText(Server.MapPath("~/cloud/") + filename,
String.Empty);
                     using (StreamWriter writer = new
StreamWriter(Server.MapPath("~/cloud/") + filename))
                     {
                         writer.Write(cipherText1);

                     }
                     File.WriteAllText(Server.MapPath("~/cloud1/") + filename,
String.Empty);
                     using (StreamWriter writer = new
StreamWriter(Server.MapPath("~/cloud1/") + filename))
                     {
                         writer.Write(cipherText2);
```

```
                  }
                  File.WriteAllText(Server.MapPath("~/temp/") + filename, String.Empty);
                  using (StreamWriter writer = new
StreamWriter(Server.MapPath("~/temp/") + filename))
                  {
                     writer.Write(cipherText3);

                  }
                  File.WriteAllText(Server.MapPath("~/bin/") + filename, String.Empty);
                  using (StreamWriter writer = new
StreamWriter(Server.MapPath("~/bin/") + filename))
                  {
                     writer.Write(cipherText3);

                  }
                  con = new SqlConnection(conn);
                  con.Open();
                  cmd = new SqlCommand("insert into file_m values('" +
Session["mailid"].ToString() + "','" + filename + "')", con);
                  cmd.ExecuteNonQuery();
                  // FileUpload1.SaveAs(Server.MapPath("~/") + filename);
                   StatusLabel.Text = "Upload status: File uploaded!";
                }
                else
                   StatusLabel.Text = "Upload status: The file has to be less than 100
kb!";
             }
             else
                StatusLabel.Text = "Upload status: Only Text files are accepted!";
          }
          catch (Exception ex)
          {
                StatusLabel.Text = "Upload status: The file could not be uploaded. The
following error occured: " + ex.Message;
            }
        }
    }
}
```

View Page Code

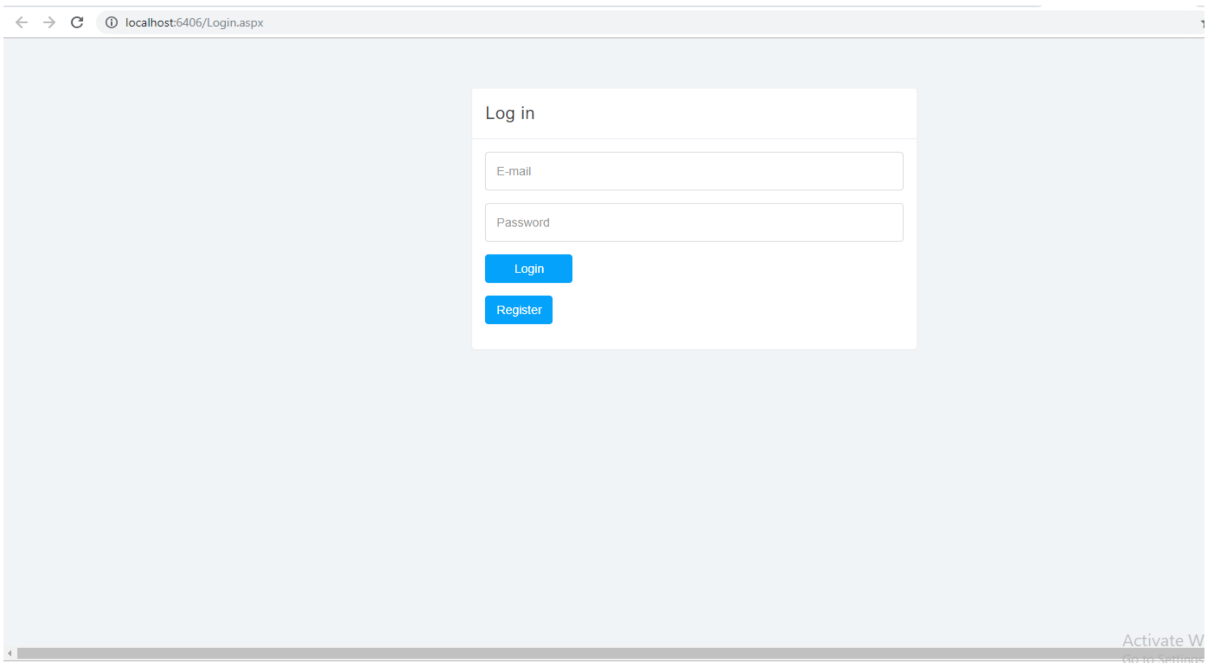
```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.SqlClient;
using System.IO;
using System.Linq;
using System.Text;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
public partial class view : System.Web.UI.Page
{
    public static int xa = 0;
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            if (string.IsNullOrEmpty(Session["mailid"] as string))
            {
                Response.Redirect("Login.aspx");
            }
            else
            {
                SqlConnection con = new SqlConnection("Data Source=SAINARU;Initial
Catalog=cloud1;Integrated Security=True");
```

```
                string strSQL = "Select * from file_m where uname='" +
Session["mailid"].ToString() + "'";
                SqlDataAdapter dt = new SqlDataAdapter(strSQL, con);
                DataSet ds = new DataSet();
                dt.Fill(ds, "uname");
                con.Close();
                GridView1.DataSource = ds;
                GridView1.DataBind();
            }
        }

    }
    protected void GridView1_SelectedIndexChanged(object sender, EventArgs e)
    {
        TextBox1.Text = "";

        TextBox4.Text  =  "";
        TextBox5.Text  =  "";
        TextBox6.Text  =  "";
        TextBox7.Text  =  "";
        TextBox8.Text  =  "";
        TextBox9.Text  =  "";
        TextBox10.Text = "";
        string f_name = Convert.ToString(GridView1.SelectedRow.Cells[1].Text);
        string dt, dt1, dt2;
        var fileStream = new FileStream(Server.MapPath("~/cloud/" + f_name),
FileMode.Open, FileAccess.Read);
        using (var streamReader = new StreamReader(fileStream, Encoding.UTF8))
        {
            dt = streamReader.ReadToEnd();
        }
        var fileStream1 = new FileStream(Server.MapPath("~/cloud1/" + f_name),
FileMode.Open, FileAccess.Read);
```

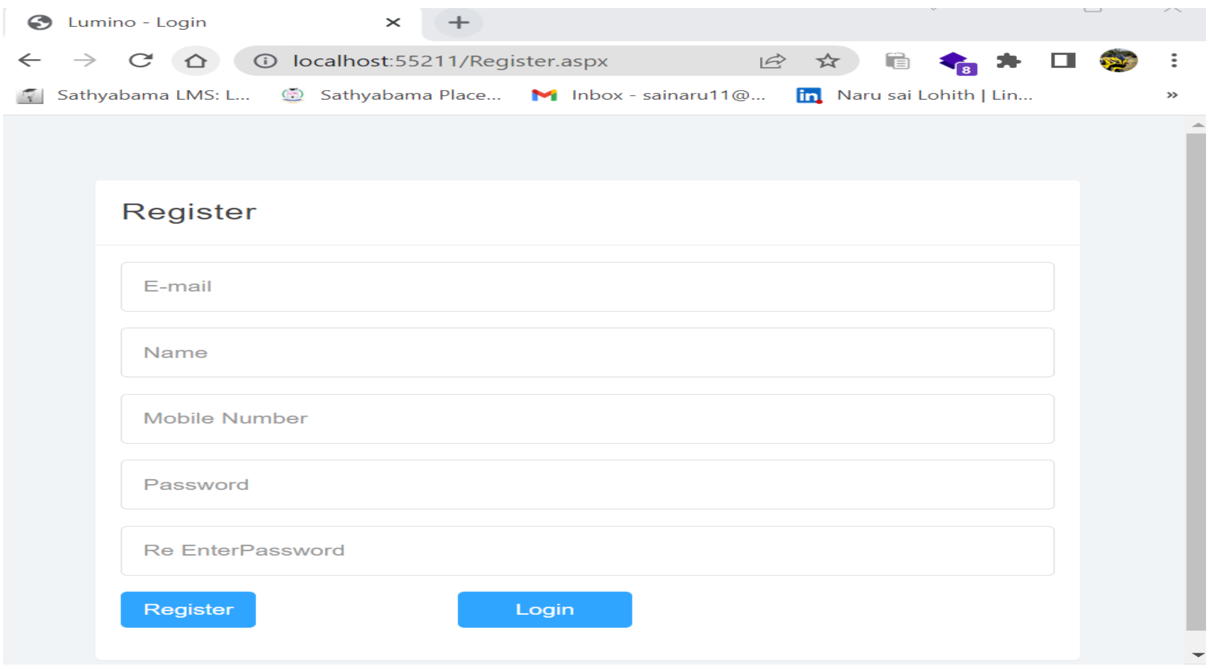
```
        using (var streamReader = new StreamReader(fileStream1, Encoding.UTF8))
        {
            dt1 = streamReader.ReadToEnd();
        }
        string path1 = Server.MapPath("~/temp/" + f_name);
        FileInfo file1 = new FileInfo(path1);
        if (file1.Exists)//check file exsit or not
        {
            var fileStream2 = new FileStream(Server.MapPath("~/temp/" + f_name),
FileMode.Open, FileAccess.Read);
            using (var streamReader = new StreamReader(fileStream2, Encoding.UTF8))
            {
                dt2 = streamReader.ReadToEnd();
            }

        }
        else
        {
            var fileStream2 = new FileStream(Server.MapPath("~/bin/" + f_name),
FileMode.Open, FileAccess.Read);
            using (var streamReader = new StreamReader(fileStream2, Encoding.UTF8))
            {
                dt2 = streamReader.ReadToEnd();
            }
        }
        TextBox1.Text = dt;
        TextBox4.Text = dt1;
        TextBox5.Text = dt2;
        }
    protected void Button4_Click(object sender, EventArgs e)
    {
        string plain1 = CryptorEngine.Decrypt(TextBox1.Text, true);
        TextBox6.Text = plain1;
        xa = xa + 1;
        if(xa==3)
        {
            xa = 0;
            Button6.Visible = true;
        }
    }
    protected void Button2_Click(object sender, EventArgs e)
    {
        string plain1 = CryptorEngine.Decrypt(TextBox4.Text, true);
        TextBox7.Text = plain1;
        xa = xa + 1;
        if (xa == 3)
        {
            xa = 0;
            Button6.Visible = true;
        }
    }
    protected void Button3_Click(object sender, EventArgs e)
    {
        string plain1 = CryptorEngine.Decrypt(TextBox5.Text, true);
        TextBox8.Text = plain1;
        xa = xa + 1;
        if (xa == 3)
        {
            xa = 0;
            Button6.Visible = true;
        }
    }
    protected void Button6_Click(object sender, EventArgs e)
    {
        string txt = TextBox6.Text + TextBox7.Text + TextBox8.Text;
```

```
        TextBox9.Text = txt;
    }
    protected void Button5_Click(object sender, EventArgs e)
    {
        Cipher c = new Cipher();
        string p_text = Convert.ToString(c.Decrypt(TextBox9.Text, "p@SSword"));
        TextBox10.Text = p_text;
    }
}
```

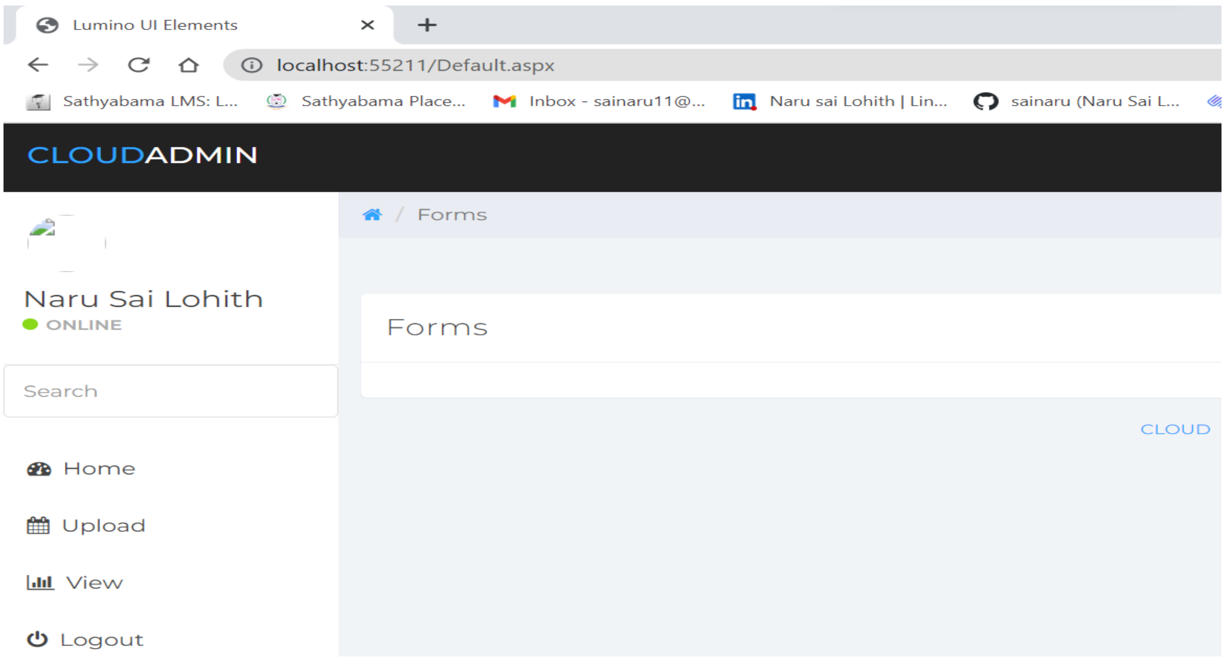
B. Screenshots


localhost:6406/Login.aspx

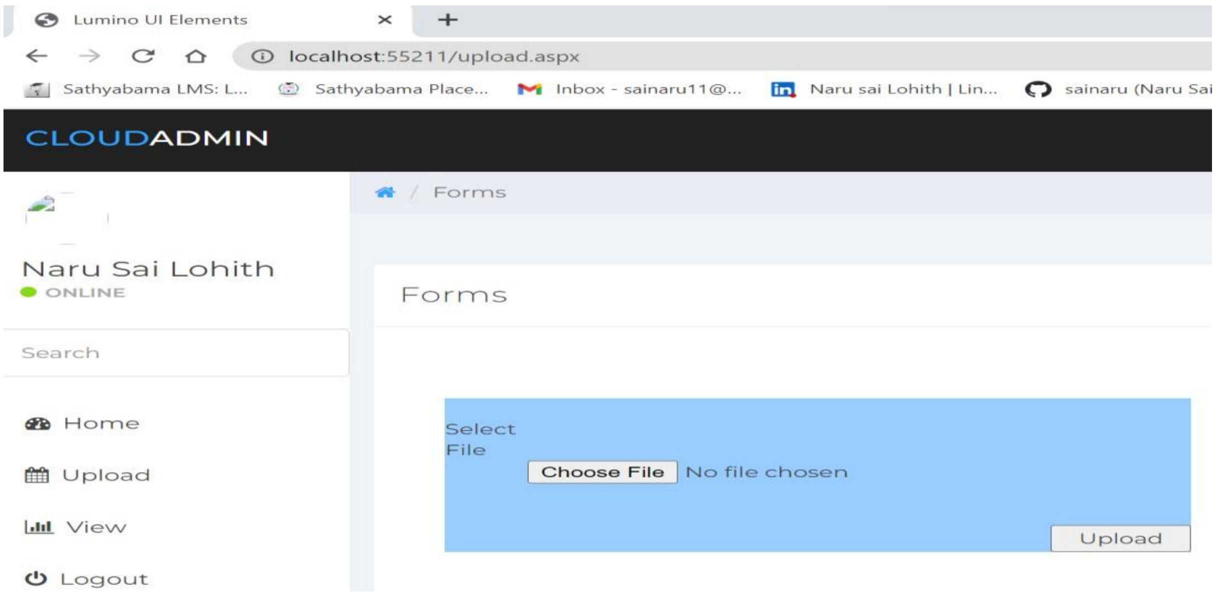
B.1. Login Page


localhost:55211/Register.aspx

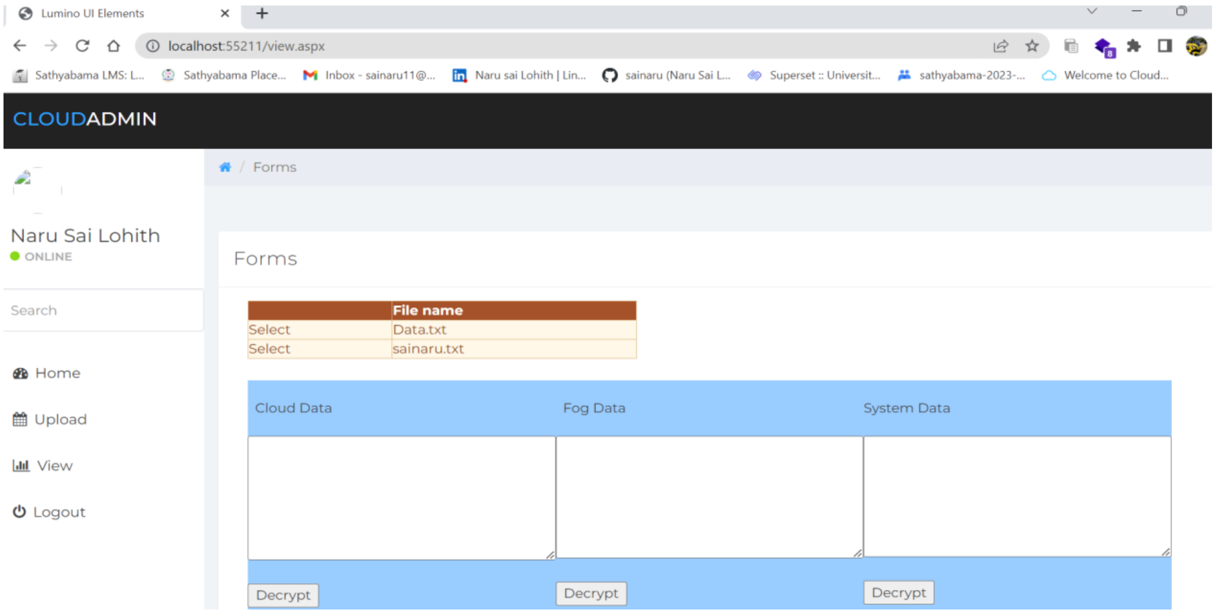
B.2. Register Page



B.3. Home Page

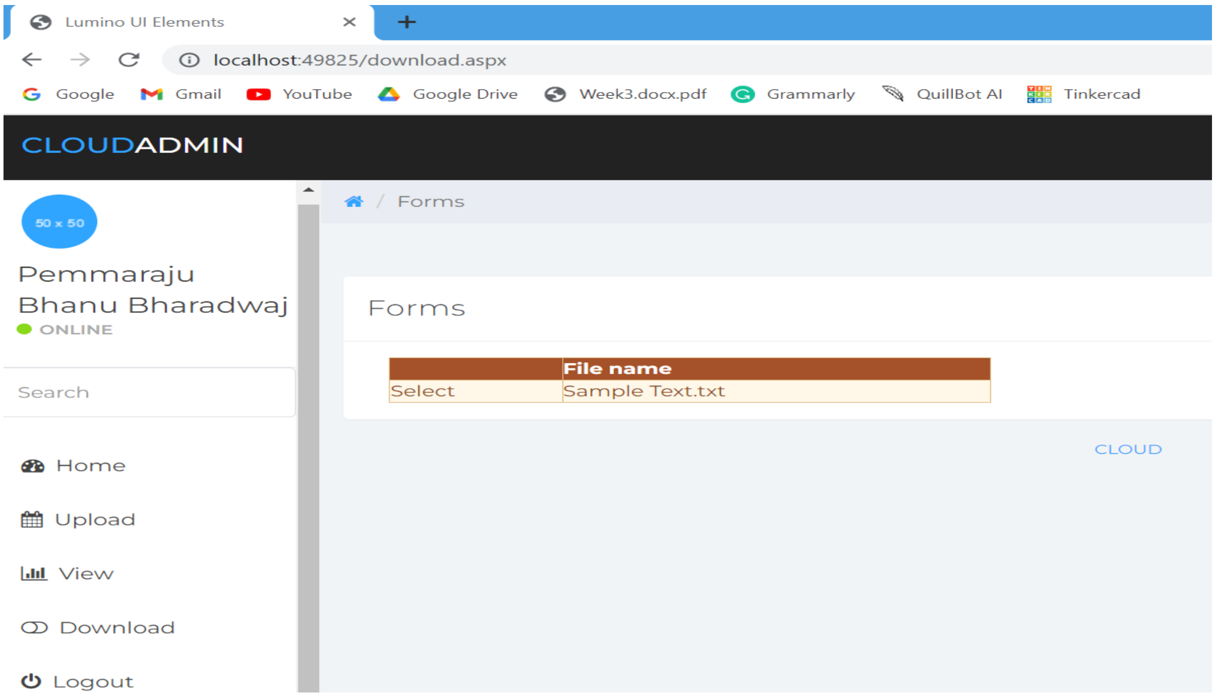


B.4. Upload Page



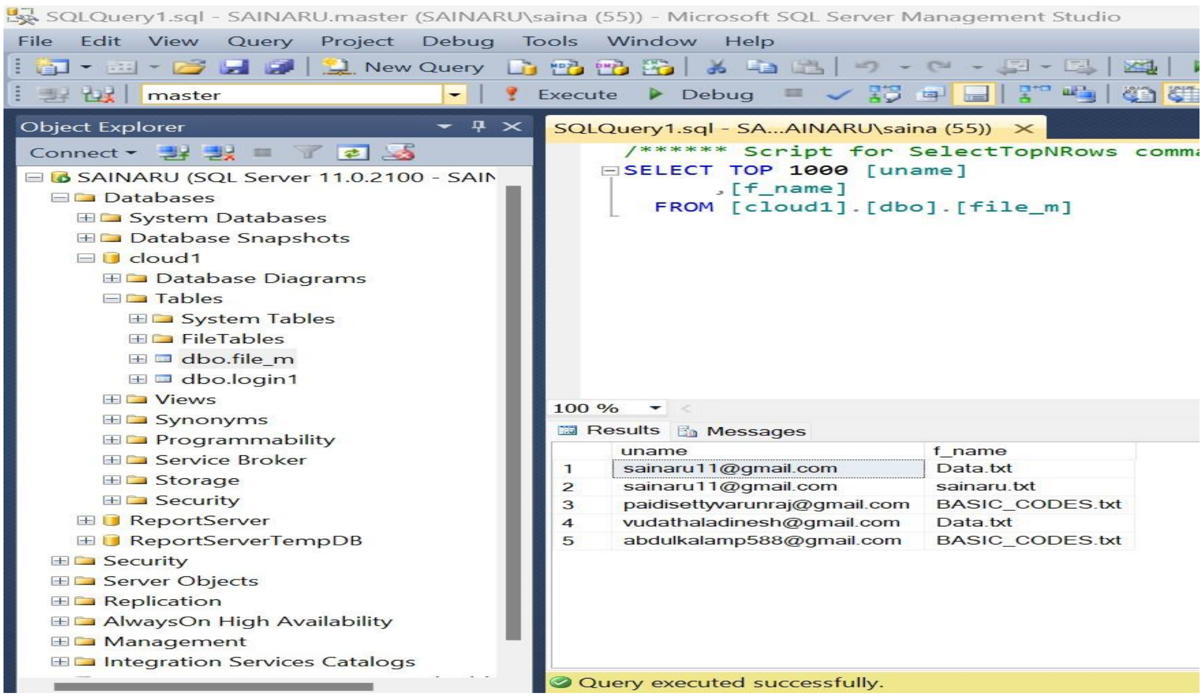
	File name
Select	Data.txt
Select	sainaru.txt

B.5. View Page



	File name
Select	Sample Text.txt

B.6. Download Page

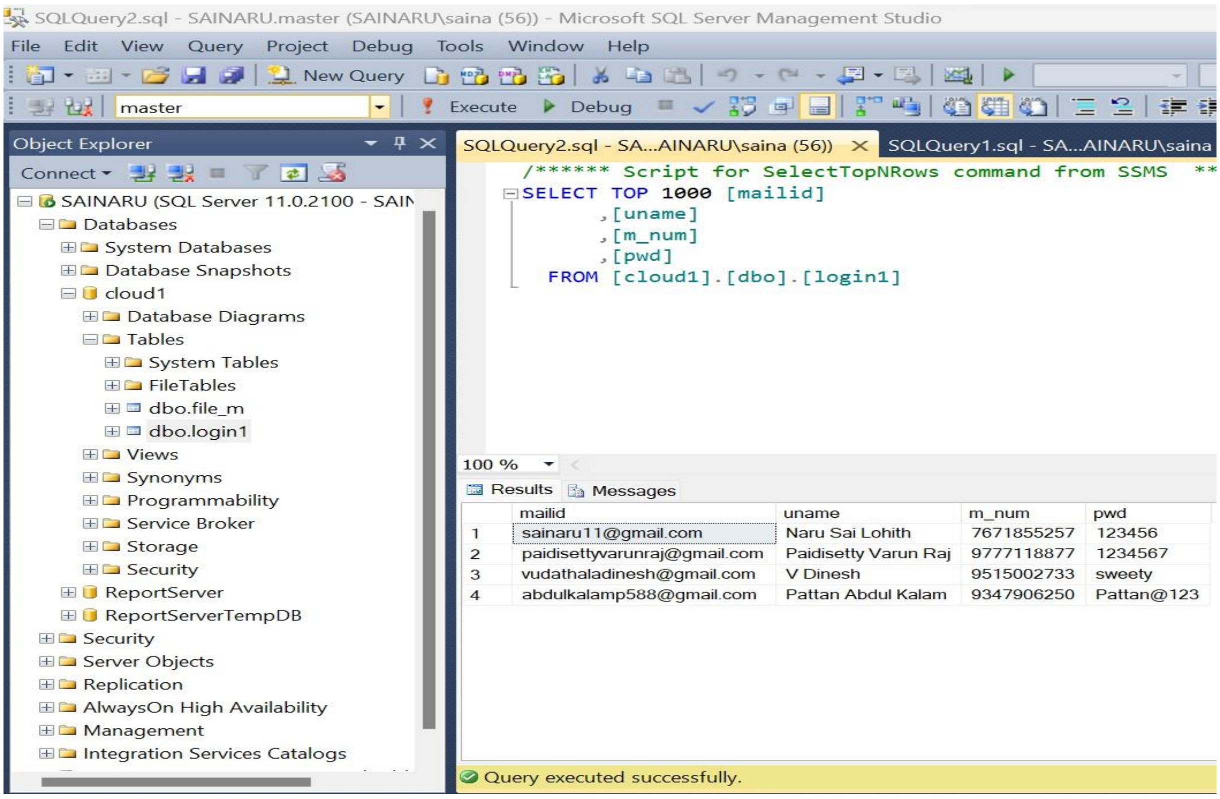


```
/****** Script for SelectTopNRows comma
SELECT TOP 1000 [uname]
      ,[f_name]
  FROM [cloud1].[dbo].[file_m]
```

	uname	f_name
1	sainaru11@gmail.com	Data.txt
2	sainaru11@gmail.com	sainaru.txt
3	paidisettyvarunraj@gmail.com	BASIC_CODES.txt
4	vudathaladinesh@gmail.com	Data.txt
5	abdulkalamp588@gmail.com	BASIC_CODES.txt

Query executed successfully.

B.7. Database Upload Files



```
/****** Script for SelectTopNRows command from SSMS  **
SELECT TOP 1000 [mailid]
      ,[uname]
      ,[m_num]
      ,[pwd]
  FROM [cloud1].[dbo].[login1]
```

	mailid	uname	m_num	pwd
1	sainaru11@gmail.com	Naru Sai Lohith	7671855257	123456
2	paidisettyvarunraj@gmail.com	Paidisetty Varun Raj	9777118877	1234567
3	vudathaladinesh@gmail.com	V Dinesh	9515002733	sweety
4	abdulkalamp588@gmail.com	Pattan Abdul Kalam	9347906250	Pattan@123

Query executed successfully.

B.8. Database Login Users

C. Research Paper

Data Security and Privacy Protection for Cloud Storage

Pemmaraju Bhanu Bharadwaj
Department of CSE
Sathyabama Institute of Science and Technology, Chennai

Naru Sai Lohith
Department of CSE
Sathyabama Institute of Science and Technology, Chennai

Dr. N Srinivasan
Department of CSE
Sathyabama Institute of Science and Technology, Chennai

Abstract

The rapid expansion of unstructured data and the rise of cloud computing have led to increased interest in and improved development of cloud storage technologies. The cloud service provider provides no advice for how to best store and keep data and information internationally. Encryption technology is used in the majority of privacy protection measures. An architectural design with three layers based on fog. Both cloud storage and data privacy can be used to their full potential with the recommended design. The hash-Solomon algorithm, which is used in this case to divide data into parts, is being used. For the most part, privacy protection measures are built on encryption technology. Several privacy-preserving strategies are available to prevent data from being saved in the cloud. Our recommended storage design is based on a three-layer fog computing system. Both cloud storage and data privacy can be used to their full potential with the recommended design. In this case, data segmentation is done using the Hash-Solomon coding technique. Data information is lost if even one of its components is missing. We safeguard the data in this framework with bucket concept-based algorithms to show the security and effectiveness of our plan. The distribution fraction that is kept on local, cloud, and fog computers, respectively, may also be found using this approach, according to computational intelligence.

Keywords: Cloud computing, Fog computing, Hash-Solomon code, Bucket framework, Advanced Encryption Standard (AES), and Secured Hash Algorithm (SHA).

INTRODUCTION

A form of outsourcing of computer services is referred to as cloud computing in computer science. The user interface is simple. They do not need to be worried about how the power is generated, distributed, or where it comes from. Each month, they pay for the food and drink they consumed. In a similar vein, cloud computing enables users to access resources such as storage, processing power, or unique development environments without worrying about how they are handled inside. The majority of cloud computing involves online activities. The cloud is an abstraction that conceals the intricate internet architecture because it is a metaphor for the internet in the same manner that computer network diagrams show it. Thanks to this sort of computing, consumers can access technology-enabled services via the Internet without being aware of or in charge of the underlying technologies. Massive data structures and expansive cloud systems are examples of fog computing, which describes the growing difficulties in obtaining unbiased access to information. As a result, the content that is produced is of low quality. Cloud computing and big data systems may be affected by fog computing in a number of different ways. However, a common trait that may be drawn is the challenge of correctly dispersing material. This problem has been resolved by developing policies intended to increase accuracy. A data plane and a control plane are the building blocks of networking. For instance, fog computing on the data plane enables the relocation of computing services from servers in data centers to the network's edge. We advise using a fog computing model-based TLS framework to protect user privacy. While giving users some degree of administrative control, the TSL architecture effectively preserves user privacy. As was already mentioned, it is difficult to resist the inner assault. While conventional techniques are useful for stopping external assaults, they are ineffective when CSP is having problems. Instead of using traditional approaches, our strategy divides user input into three chunks of different sizes using encoding technology. They will all be lacking specific vital details that are required for concealment. The user's local workstation, the cloud server, and the fog server will store the three data components in that order-from large to tiny-as part of the fog computing concept. Even if he is able to obtain all the data from a single server, the attacker will not be able to recreate the user's original data using this technique. Likewise, the local PC and fog server data are both managed by humans, preventing the CSP from accessing any relevant information without the two devices' data.

Literature Survey:

INFERENCES FROM THE LITERATURE SURVEY

Title 1: Privacy-preserving security solution for cloud services
Author: L. Malina*, J. Hajny, P. Dzurenda and V. Zeman

A front-line security method for cloud services that maintain privacy. Our strategy allows anonymous access to shared storage servers and cloud services thanks to an efficient nonlinear group signing technique. Registered users can authenticate anonymously thanks to the creative technique. Because of this, users' personal information, such as their age and the validity of their registration and payments, perhaps confirmed without disclosing the user identities, they may utilize cloud services without concern that their behavior would be monitored. Access privileges may be suspended for a user who violates the provider's policies. Our system ensures the confidentiality of data transfer, unlinkability, and anonymity of access. We put our idea into practice as a proof-of-concept program, after which we provide the findings of our investigations. We also look at existing privacy-preserving cloud service solutions as well as group signature schemes, which are key elements of privacy-enhancing cloud service solutions. In contrast to other similar solutions and schemes, we evaluate how well our solution performs.

Proposed Method

Fog computing is an expanded kind of cloud computing that consists of several fog nodes. These nodes feature a particular amount of processing power and storage. As part of our plan, we divide each of three sections of the user's data and save each one independently on a local Computer for the user, a cloud server, and a fog server. Another advantage of the Hash-Solomon code is its capacity to ensure that the original data cannot be recreated from insufficient data. However, utilizing Hash-Solomon code results in certain duplicate data blocks that are employed in the decoding process. Although increasing the amount of redundant blocks results in more data storage, it can also improve storage dependability. Our plan can effectively secure the privacy of user data through the sensible allocation of data. Complex calculations are required for the Hash Solomon code, and computational intelligence can help (CI). Our plan can offer a higher level of privacy protection from the inside, particularly from the CSPs, when compared to conventional ways.

Advantages of the Proposed system:
We employ a bucket idea in our system to cut down on process times and data waste. BCH (Bose-Chaudhuri-Hocquenghem) code is the algorithm we are utilizing. Extremely flexible. There is little redundancy in BCH codes, which are employed in many communications applications.

Existing System

The expansion of the data center onto the cloud presents opportunities for improvement in risk perception. The user's information is kept on cloud servers using this storage structure. If the user's right to data control is violated and their privacy is at stake. Most privacy protection strategies rely on encryption technology. These techniques are unable to successfully fend against assaults that originate on cloud servers.

Problems in the Existing system:

- Data Integrity.
- Low latency.
- Location awareness.

Title 2: A secure data privacy preservation for on-demand cloud service

Author: Chandramohan Dhasarathan *, Vengattaraman Thirumal, Dhavachelvan Ponnurangam.

The article focuses on issues related to intellectual property, secrecy, and privacy that affect the banking and insurance sectors. Privacy risk if authoritarians in the contemporary business environment exploit hidden knowledge. Software disruptions brought on by data theft for third-party services. Maintaining business continuity requires liability in digital secrecy. Isolation, improper management leading to privacy breaches nearby, and its preventative phenomena are rigorously regulated where a sizable amount of data is kept current and saved on the cloud. Despite the fact that cloud computing has changed the computer industry, The identification, dependability, maintainability, and privacy of cloud users' data as well as other factors like boosting efficacy, efficiency, and service environment optimization may differ for different cloud providers. According to CP, contemporary technologies enable more covert storage of the user's private data. Even more astounding is the lack of rules for digital and informational data kept and maintained internationally in the cloud, not even by the cloud provider. The suggested approach deals with one of the important cloud computing research issues. To avoid data leakage in the virtual servers, we proposed the privacypreserving paradigm. This suggestion fosters confidence mostly in CR's (cloud requester/users) ability to store confidential data and information.

Title 3: Survey on Privacy-Preserving Methods for Storage in Cloud Computing
Author: Neethu Mariam Joseph, Esther Daniel, N. A. Vasanthi, Ph.D

The human race currently depends increasingly, we rely on a variety of online storage services to restore our data or utilize it immediately, allowing access from anywhere, at any time. Since the user's data is controlled and stored off-site, all of these services raise concerns about security and privacy weaknesses across the board. The many privacy issues are examined in this article when user data is held by third-party service providers, sometimes referred to as cloud services. The phrase "cloud computing" refers to the fundamental structure of a developing service delivery paradigm by combining processing and storage resources and utilizing an on-demand provisioning method that adjusts capacity in response to demand, which has the advantage of lowering costs.

Title 4: A Survey on Secure Storage Services in Cloud Computing

Author: Ms. B.Tejaswi, Dr. L.V.Reddy & Ms. M.Leelavathi

Cloud computing is a relatively recent invention that only makes use of the internet and its surroundings. Users can choose from a variety of services, including storage as a service, Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). The potential for clients and businesses utilizing storage as a service to store their data offshore creates more concerns about the security and dependability of cloud-based data. A number of methods, distributed erasure-coded data, the development of Merkle Hash Trees (MHT), and adaptability distributed storage integrity auditing systems, etc., can be used to provide secure cloud storage. These methods help to adequately secure and archive dynamic data in the cloud. In the context of cloud storage, designs for controlling security and privacy are also covered in this study.

Title 5: On a Relation between Verifiable Secret Sharing Schemes and a Class of Error-Correcting Codes
Author: Ventzislav Nikov and Svetla Nikova

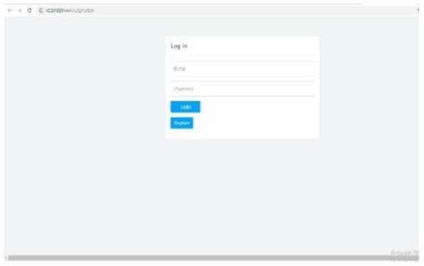
We aim to provide a new viewpoint on Verifiable Secret Sharing Schemes in our debate. We start by developing a fresh "metric". Based on a set of banned lengths that is a set that reduces monotonically, we use this metric to construct an incredibly specialized class of codes that we term error-set correcting codes. The packing problem is then modified to consider the altered situation and the enhanced error-set correcting coding abilities. Following that, we employ burst-error interleaving codes, which demonstrate the ability of error-set correcting interleaving codes to correct burst errors and are in fact the very well pair-wise checking protocol utilized by Distributed Commitments (DC) and VSS.

Modules

- Login Module
- Register Module
- Upload Module
- Recovery Module
- Download Module

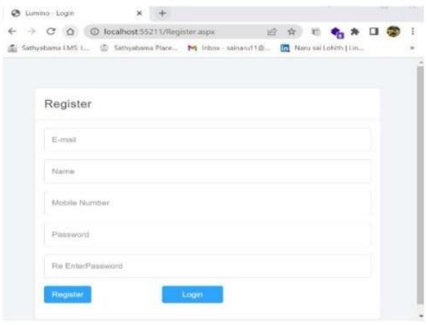
Login Module:

When someone accesses a website, they see this right away. The user must provide their registered password together with a working phone number in order to access the website. The user can access the website successfully if their information matches that in the database table; otherwise, the message "login failed" is displayed, and they must reenter the necessary information. A link to the registered activity is also made available during the registration of new users.

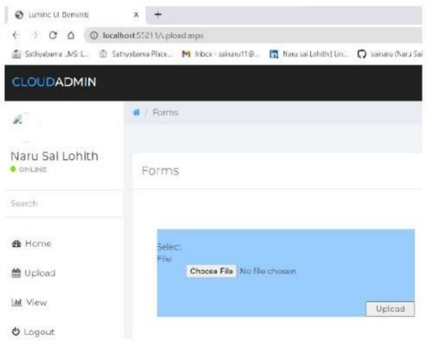


Register Module:

A new user must register in order to use the website before logging in. By selecting the register button under the login activity, the registered activity is launched. The entire name, password, and phone number of a new user are entered during registration. The user is taken back to the login screen after filling out all text boxes and clicking the register button, which sends the information to the database. An authenticated user must log in before they may access the website. For the website to operate properly, validations are applied to each textbox. Each textbox must have information; hence, the text fields for name, contact information, password, and confirm password cannot be left empty during registration. The notification that information must be entered into each textbox will be shown if any such textbox is left empty.



Upload Module:
This module allows the user to upload data to three different storage sites. The owner loses control of the data once it is uploaded to the cloud. In this module, three independent levels of encryption are applied to the original data. The data in each layer can be encrypted beforehand using several cryptographic techniques and encryption keys. Storage servers include local servers, cloud servers, and fog servers. Cloud servers house 80% of the world's data. 15% of the data kept on the fog server is sensitive data. On a local server, 5% of the data is kept.

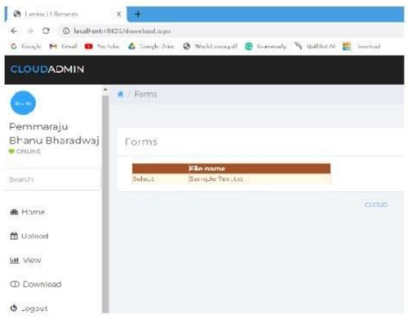


Recovery Module:

By using the BCH method, this module enables users to recover files from three levels of storage if the data on the storage server can be matched with the three layers of data. If the data was compromised in any manner, it was maintained in the bucket. The bucket of the layer's framework allows the user to rapidly get it.

Download Module:

Customers are given instructions on how to obtain files from cloud servers. After receiving the user's request, the cloud server integrates the data from several distant servers. The first portion of the data is obtained from the virtual server (cloud), the second portion of the data is obtained by a virtual server (fog) and the remaining data is accumulated with both data and sent to the local storage, where the original data is obtained. By repeating the aforementioned steps, the user can obtain all the data.



Conclusion:

There are many benefits to the expansion of cloud computing. With the use of the useful technology referred to as cloud storage, users can boost their storage capacity. However, cloud storage also plays a role in a variety of security problems. As they lack command over the physical storage of their data, users who use cloud storage face a dissociation of ownership and administration of data. In order to tackle the problem of privacy preservation in cloud storage, we design a TLS framework based on the fog concept and a BCH Code algorithm. The theoretical safety research shows the validity of the scheme. By evenly spreading the amount of data blocks kept on other servers, we can protect the confidentiality of information on each server.

The encoding matrix cannot theoretically be decoded. Moreover, it is possible to use hash transformation to protect the partial information. The experiment test revealed that this technique of decoding and encoding could be accomplished without impairing the efficiency of cloud storage. We also develop a logical, all-encompassing efficiency measure to achieve maximum efficiency. We also find that coding is improved by the Cauchy matrix.

REFERENCES:

[1] Privacy-preserving security solution for cloud services
Author: L. Malina*, J. Hajny, P. Dzurenda and V. Zeman

[2] A secure data privacy preservation for on-demand cloud service
Author: Chandramohan Dhasarathan *, Vengattaraman Thirumal, Dhavachelvan Ponnurangam

[3] A Survey on Secure Storage Services in Cloud Computing
Author: Ms. B.Tejaswi, Dr. L.V.Reddy &amp; Ms. M.Leelavathi

[4] On a Relation Between Verifiable Secret Sharing Schemes and a Class of Error-
Correcting Codes
Author: Ventzislav Nikov and Svetla Nikova