**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**

**JNANASANGAMA, BELAGAVI-590018**

# PROJECT REPORT

**ON**

## "A BIOMETRICS-BASED SAFE SYSTEM FOR ENCRYPTION"

**Submitted in Partial fulfilment of the Requirements for the VIII Semester of the Degree of**

## BACHELOR OF ENGINEERING

**in**

## Information Science and Engineering


*Submitted by*

**PRANJALI S, (1CR19IS107)**
**S DISHA ADIGA, (1CR19IS123)**
**VANI B TELAGADE, (1CR19IS170)**

*Under the Guidance of*

**Internal Guide**
**Dr. Srividya R**
**Associate Professor**
**Dept. of ISE, CMRIT**

**DEPARTMENT OF INFORMATION SCIENCE& ENGINEERING**

**CMR INSTITUTE OF TECHNOLOGY**

#132, AECS LAYOUT, IT PARK ROAD, KUNDALAHALLI, BANGALORE-560037


**2022-2023**

## DEPT. OF INFORMATION SCIENCE & ENGINEERING

# *Certificate*

This is to Certified that the project work entitled **"A biometrics-based safe system for encryption"** carried out by **Pranjali S (1CR19IS107), S Disha Adiga (1CR19IS123) and Vani B Telagade (1CR19IS170)** in partial fulfillment for the award of **Bachelor of Engineering** in **Information Science & Engineering** of the Visvesvaraya Technological University, Belgaum during the year **2022-2023**. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library. The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.

| | | |
|---|---|---|
| ---------- | ---------- | ---------- |
| **Signature of Guide** | **Signature of HOD** | **Signature of Principal** |
| **Dr. Srividya R** | **Dr. M. Farida Begam** | **Dr. Sanjay Jain** |
| **Associate Professor** | **Professor & HOD** | **Principal** |
| **Department of ISE** | **Department of ISE** | **CMRIT, Bengaluru** |
| **CMRIT, Bengaluru** | **CMRIT, Bengaluru** | |

**External Viva**

| | **Name of Examiners** | **Institution** | **Signature with Date** |
|---|---|---|---|
| **1.** | ---------- | ---------- | ---------- |
| **2.** | ---------- | ---------- | ---------- |

# ACKNOWLEDGEMENT

Any work of significance requires a great deal of effort and time put into it. But a factor of even greater importance is efficient guidance and encouragement. In spite of all my dedicated work, this internship would not have been possible without continuous help and guidance provided by people who gave their unending support right from when this idea was conceived.

I would like to thank to **Dr. Sanjay Jain,** Principal, CMRIT, Bangalore, for his constant co-operation and support throughout this Technical Seminar tenure.

I would like to thank **Dr. M Farida Begam, Professor & Head ,** Department of Information Science and Engineering, CMRIT for her constant guidance and support during this Technical Seminar period.

I would like to thank my guide, **Dr. Srividya R, Associate Professor,** Department of Information Science and Engineering, CMRIT for her constant guidance that helped me in completing the Technical Seminar work successfully.

Last but definitely not the least I would like to thank **my family** and **friends** who have always supported me in every path of the technical seminar work.

**Pranjali S**
**S Disha Adiga**
**Vani B Telagade**

# DECLARATION

We, **Pranjali S (1CR19IS107), S Disha Adiga (1CR19IS123) and Vani B Telagade (1CR19IS170**) bonafide students of **CMR Institute of Technology, Bengaluru**, hereby declare that the dissertation entitled, "**A biometrics-based safe system for encryption**" has been carried out by us under the guidance of **Dr. Srividya R**, Associate Professor, CMRIT, Bengaluru, in partial fulfillment of the requirements for the award of the degree of Bachelor of Engineering in Information Science & Engineering, of the Visvesvaraya Technological University, Belgaum during the academic year 2022-2023. The work done in this dissertation report is original and it has not been submitted for any other degree in any university.

Place: Bengaluru

**Pranjali S**

**S Disha Adiga**

**Vani B Telagade**

# ABSTRACT

Privacy of user data is a matter of concern which can be observed as applications of big data tend to increase in these days. The implementations of the emerging technologies have improved service models and performance of applications. But the remarkably growing volumes of data sizes have given rise to many challenges.

During data transmission and processing, time taken for executing the encryption of data results in an issue.

In order to handle the concerns regarding privacy and to attain adoptive performance level, many applications are not encrypting the data.

This project focuses on privacy and hence proposes an encryption algorithm which is considered to be an innovative idea.

It is designed in order to maximize the protection of privacy by making use of a selective strategy for encryption which makes use of the time requirements for execution in an optimized way.

# TABLE OF CONTENTS


**Abstract** i

**Table of contents** ii-iii

**List of tables** iv

**List of figures** v

**Chapter 1 – Introduction** 1-2

**Chapter 2 – Literature survey** 3-4

**Chapter 3 – System requirements**

3.1 Hardware requirements 5

3.2 Software requirements 5

**Chapter 4 – Proposed system**

4.1 Encryption using DNA 6

4.2 Proposed Design 7

4.3 Procedure for encryption 7-8

4.4 Procedure for decryption 9

**Chapter 5 – System design**

5.1 Logical design 9

5.2 Design goals 9

5.3 The Model View Controller Architecture 11

5.4 Struts in Java 12

5.5 Software architecture 13

5.6 Class diagram 14

5.7 Use case diagram 14

5.8 Data flow diagram 16

5.9 Sequence diagram 17

5.10 Context analysis diagram 18

**Chapter 6 – Implementation**

6.1 User homepage 16

6.2 Admin homepage 16

6.3 Packages for file upload 17

6.4 Read contents of file 17

6.5 Remove unnecessary information 17

6.6 Split files into blocks 18

6.7 Preprocess files 18

6.8 Files classified as sensitive or not 19

6.9 HTML 19

6.10 JavaScript 20

6.11 Servlet 21-22

**Chapter 7 – Testing**

7.1 Definition 23

7.2 Types 23

**Chapter 8 – Implementation** **24-28**

**Chapter 9 – Conclusion** **29**

**REFERENCES**

# LIST OF TABLES

**Table** **Page No**

Table 8.1: m_admin ..... 32
Table 8.2: m_blocks ..... 32
Table 8.3: m_cluster ..... 32
Table 8.4: m_files ..... 32

# LIST OF FIGURES

**Figure** **Page No**


Figure 1.1: Balance between protection and efficiency ..... 1
Figure 4.1a: Binary mapping of DNA nucleotides ..... 6
Figure 4.1b: A general representation of DNA cryptography ..... 6
Figure 4.3: Biometrics based algorithm for encryption ..... 8
Figure 4.4: Biometrics based algorithm for encryption ..... 9
Figure 5.3: J2EE using MVC Architecture ..... 12
Figure 5.5 : System Architecture ..... 13
Figure 5.6: Class diagram for user ..... 14
Figure 5.7a: User use case diagram ..... 15
Figure 5.7b: Admin use case diagram ..... 15
Figure 5.8a: Dataflow diagram for user ..... 16
Figure 5.8b: Admin dataflow diagram ..... 16
Figure 5.9: Sequence diagram for the system ..... 17
Figure 5.10: Context Analysis diagram ..... 18
Figure 6.1: Code for user homepage ..... 19
Figure 6.2: Code for admin homepage ..... 19
Figure 6.3: Code for importing packages ..... 20
Figure 6.4: Read contents of file ..... 20
Figure 6.5: Removal of unnecessary information ..... 20
Figure 6.6: Code to split files ..... 21
Figure 6.7 : Upload and preprocess file ..... 21
Figure 6.8a: Code for sensitive file ..... 22
Figure 6.8b: Code for non-sensitive file ..... 22
Figure 6.11a: Servlet ..... 25
Figure 6.11b: Servlet interface ..... 25
Figure 6.11c: JSP Model ..... 26
Figure 8.1: User registration ..... 28
Figure 8.2: User login ..... 28
Figure 8.3: Admin login ..... 29
Figure 8.4: Admin profile ..... 29
Figure 8.5: Upload files ..... 29
Figure 8.6: Download options ..... 30
Figure 8.7: View user details ..... 30
Figure 8.8: Detect state of file ..... 30
Figure 8.9: Display status of file ..... 31
Figure 8.10 : Files stored as blocks ..... 31
Figure 8.11 : File download ..... 31

# Chapter 1

# INTRODUCTION

The Internet has connected the world like never before, and mobile computing is the future of technology as it allows us to access the Internet and data on-the-go. Mobile cloud computing techniques have enabled diverse applications, but concerns remain about data security and privacy. One of the key privacy concerns is caused by unencrypted data transmissions due to the large volume of data. This can result in privacy leakage issues, as plaintext data is easily captured by adversaries through methods such as jamming, monitoring, and spoofing. To address this problem, we have used an encryption-decryption algorithm which uses a DNA strand as a key. This approach also makes use of the base-pairing concept in DNA.

Figure 1.1 shows the high level architecture for addressing privacy protections which illustrates the addressing of privacy issues.

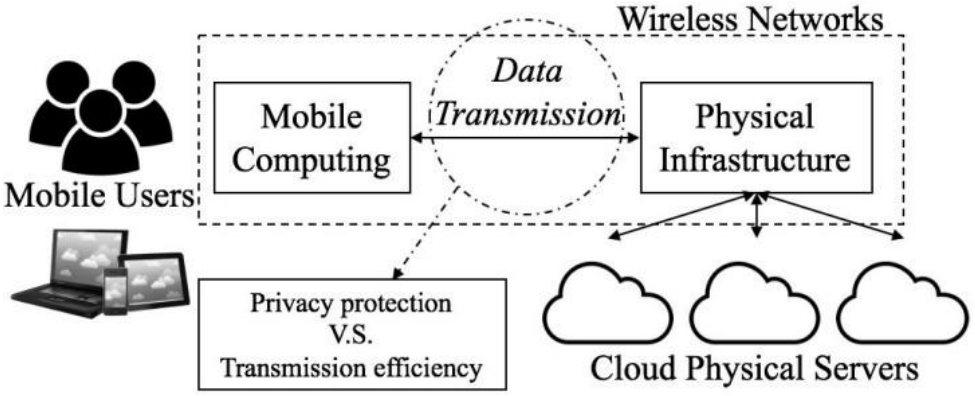



**Fig. 1.1.** Balance between protection and efficiency

The main issue is that most wireless transmissions carry plain text and the implementation of big data stops the transmission from carrying cipher texts. The target protection location represented as broken line box in figure 1.1 shows that the transmissions between mobile computing and physical infrastructure.

In the proposed model, the user is given the opportunity to upload files to the cloud server. Based on a set of pre-defined terms related to military security, the uploaded files are checked. If the uploaded files contains any term present in the already defined file, the file will be classified as sensitive.

If the file is classified as sensitive, first, we compute the size of the file. The logical based address will be calculated to decide how many blocks the file will be divided into. Then, the file is encrypted and uploaded to the cloud server.

Techniques that D2ES uses are firstly, classification of data packages into sensitive and non-sensitive and secondly, determining if encryption can be done.

The main contributions of this work are threefold: First, the proposed approach selectively encrypts data packages so that time can be reduced in encrypting unnecessary data.

Two working modes are considered when creating the transmission strategy, including encryption and non-encryption modes. Second, the proposed algorithm offers an optimal solution providing the maximum value of total privacy weights. Two involved constraints are execution time and privacy levels.

Overall, this project proposes a novel and effective approach for safeguarding privacy of user data. By implementing this model, data privacy and security can be significantly enhanced without compromising performance.

# Chapter 2

# LITERATURE SURVEY

Many applications disregard data encryption to achieve compliance. This work expresses privacy concerns about data and proposes Dynamic Data Encryption Strategy (DDES), a revolutionary data encryption method. The goal of this addition is to broaden the breadth of privacy protection while minimising time limitations[1].

Numerous cutting-edge applications in Cyber-Physical Systems (CPSs), such as Intelligent Transportation Systems (ITSs), are being driven by the anticipated enhanced network explorations and the rising need for mobile data sharing and transferring. However, the tensions between security and communication effectiveness limit the use of ITS today. The Security Aware Efficient Data Sharing and Transferring (SA-EAST) paradigm, which is intended for safeguarding cloud-based ITS deployments, is proposed in this article as a solution to this problem. We want to achieve safe real-time multimedia data sharing and transferring by using this method. Our experimental assessment has demonstrated that our suggested paradigm performs well at safeguarding communications for ITS[3].

In the data from fusion reactors, a brand-new technique and its application for real-time blob filament detection and tracking are provided. Numerous additional uses, such as ignition kernels in combustion and tumour cells in a diagnostic picture, depend on similar temporal properties. The method for obtaining these characteristics is presented in this study by breaking down the total process into three steps: local feature cell verification, grouping feature cells into extended characteristics, and monitoring feature motion through spatial overlaying. [4]

DDoS attack source traceback is an unsolved and difficult issue. A straightforward and efficient traceback approach is deterministic packet marking (DPM), however existing DPM-based traceback schemes are impractical because of their scaling limitations. The fact that just a few computers and routers are participating in an assault session is something we discovered. Therefore, instead of designating every Internet node as the current techniques do, we simply need to label specific involved nodes for traceback purposes. We suggest a novel marking on demand (MOD) traceback approach based on the DPM mechanism in light of this discovery. We must use the conventional DPM method to designate these implicated ingress routers in order to identify the involved attack source.[5]

The idea of DNA computing has led to the development of several algorithms. In the past 20 years, DNA computing has been used to tackle challenging problems like SAT and DES. DNA computing has been used by Lipton [6] to solve the NP-complete SAT problem. He also developed a DNA-based encoding system, and the author used the fewest possible variables to answer the SAT issue. Data Encryption Standard (DES) cracking was proposed by Boneh et al [7] using DNA computing.

With their suggested approach, every cryptosystem might be broken with a key size of only 64 bits or less. Additionally, its encryption circuit is compact and can be broken in 916 steps. Each phase is equivalent to 32 extractions, and processing takes a minimum of one full day for each of the 10 extractor stages. To break DES using their suggested technique would

take at least four months. The DNA computing approaches developed by Chen et al. [8] might be used to address challenging problems. With the use of DNA technology, they also cracked DES. The predicted algorithm by Chen et al. has three different functions: 1) The key space should be initialised with every possible using the initial function. 2) The encryption procedure; and 3) Finding the appropriate related key. The molecular sticker algorithm also includes tubes and short memory strands.

Symmetric-key MingXin et al. [9] proposed the DNA cryptosystem, which incorporates DNA biotechnology with cryptography techniques. This technique for encryption and decryption was developed with the use of DNA probes, and the ciphertext was then included into the DNA chip (microarray). The development of DNA chips results in the method's privacy. A suggested approach for the effective, dependable, and secure transfer of data was recently made by Kar et al. [10]. The method makes use of DNA keys, which comprised three keys: two for encryption and one for communicating the encryption key to the recipient. In this approach, the text itself will be transformed to a binary sequence through a process called string to binary transformation.

The massive DNA pattern is converted into binary data using a binary coding algorithm as the encryption key. Performing XOR and adding procedures to the 64-bit raw and 64-bit key would provide the ciphertext. Shiu et al. has made several cryptographic technique suggestions [11]. The following are the cryptographic methods that employ the DNA method: a) Insertion technique Methods two and three are replacement and complementing pair. They also demonstrated that the replacement strategy is more effective than the other two.

By encrypting ciphertext in a word file with the DNA sequence, Liu et al. [12] put out a data concealing technique. Their approach turns the plaintext into a DNA sequence by using DNA coding. Chebyshev maps construct one time key using the desired DNA sequence key and two pseudorandom DNA sequences, XXOR and YPrimer. The YPrimer and the XXOR are included in the output, respectively, for the DNA messaging sequence. They might be turned into ciphertext by using the shift procedures on the result that had been generated earlier. Following the encryption of the ciphertext into the word file, the word file might be transformed to PDF.

The created PDF will be delivered to the recipient. Mandge et al. [13] had suggested a powerful ciphertext method that combined key generation technology with matrix insertion modification in the encryption process. The several shifting and XOR operations will first turn the plaintext into a mini-cipher. It is claimed that because this procedure incorporates safe key creation, if it were applied to the plaintext each time, we might acquire different miniciphers for the same plaintext. After using a number of biotechnologies, including DNA primers, to transform this minicipher into a final ciphertext. The table lookup substitution method (TLSM), put out by Taur et al. [14], has improved the substitution approach using the single bit complementary rule of Shiu et al. They expanded to include the replacement method's two bits complementing rule. TLSM, which makes use of lookup tables [15][16], converts the two-bit message to the matching letter. Shiu et al.'s rule, on the other hand, could only translate one bit at a time. Additionally, the TLSM technique uses highly good ciphertext compression [17].

# Chapter 3

# SYSTEM REQUIREMENT SPECIFICATION

The below sections specify the hardware and software requirements that is needed for implementing the project

## 3.1 Hardware Requirements

The term "hardware requirements" describes the precise physical elements or gadgets required to carry out a certain operation or support a particular piece of software or system.

- **Processor** : 733
- **Keyboard** : 104 keys
- **Floppy drive** : 1.44 MB MHz Pentium III
- **RAM** : 128 MB
- **Hard Disk** : 10 GB
- **Monitor** : 14”VGA Color
- **Mouse** : Logitech serial mouse
- **Disk space** : 1 GB

## 3.2 Software Requirements

In order to satisfy the expectations of its users, a software system or application must include a number of specialised features and functionalities.

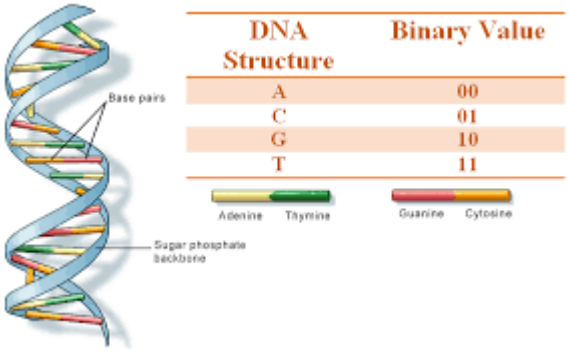
- **Operating system** : Win 2000/XP
- **Technologies used** : Java, Servlets, JDBC and JSP
- **JDK** : Version 1.4
- **Database** : MySQL 5.0

# Chapter 4

# PROPOSED SYSTEM

## 4.1 Encryption using DNA

A relatively new field called DNA cryptography makes use of the characteristics of DNA molecules to develop safe cryptographic methods. Adenine, Cytosine, Guanine, and Thymine are the four nucleotides that serve as the binary alphabet (0 and 1) for the purpose of encoding information into DNA sequences, which is the fundamental concept underpinning DNA cryptography. This is represented in Fig 4.1a. A string of bits that can be used to represent data, such as text, images, or other kinds of digital information, can be represented by the sequence of nucleotides.




| DNA Structure | Binary Value |
|---|---|
| A | 00 |
| C | 01 |
| G | 10 |
| T | 11 |

Fig 4.1a Binary mapping of DNA Nucleotides

By exploiting the special qualities of DNA molecules to encode digital information, DNA can be utilised to encrypt and decrypt files. Digital data is transformed into a DNA sequence through the process, which may then be sent, stored, and retrieved as required.

In the encryption process using DNA, a binary representation of digital data using 0s and 1s is created at first. This binary data is then separated into manageable chunks. Using an encoding technique, map each segment to a corresponding DNA sequence. Finally, a single DNA sequence that represents the encrypted data by joining the resulting DNA sequences is created. This entire process is represented in Fig 4.1b.

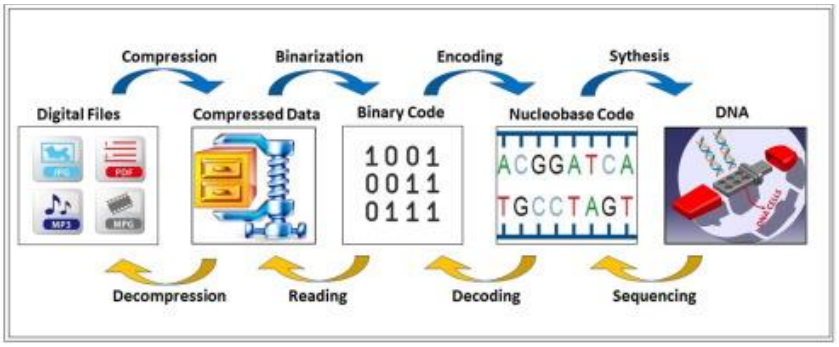



Fig 4.1b A general representation of DNA cryptography

## 4.2 Proposed design

**Phase1: Encryption**

**Step one**: Convert binary data to DNA sequences.

A=00,
T=01,
C=10, and
G=11

**Step two**: Complementary pair rule.
Complementary pair rule is a unique equivalent pair which is assigned to every nucleotides base pair.
Example:

Complementary rule:((AC)(CG)(GT)(TA))
DNA strand: AATGCT

Applying complementary rule on DNA strand: CCATGA

**Step three**: Representing DNA sequences as numeric data.
We extract the index of each couple nucleotides in DNA reference sequence, numerically.

Example:
Assume the reference sequence to be CT1GA2TC3CC4GC5AT6TT7.
Then the numerical representation will be 040602.

**Phase 2 : Decryption :**

**Step One**: Convert numeric data to DNA sequences.
We extract the couple nucleotides in DNA reference sequence according to the index read from the file.

**Step two**: Complementary pair rule.
Complementary pair rule is a unique equivalent pair which is assigned to every nucleotides base pair.

**Step three**: Convert DNA sequences to binary data.

When the user is downloading a file from the web server the corresponding file has to fetch from cloud server 2 and it will send to DNA decryption service which is running in cloud server 1. Once the file is decrypted it will be downloaded to the user machine [1].

In this implementation, the admin assigns a unique DNA sequence and a unique key to every user which is later used to create the DNA reference sequence for the user using rand() function and hash set constructs.

## 4.3 Procedure for Encryption

Consider X to be the original information that the user wishes to encrypt. This data has been identified as sensitive after comparison with the set of keywords. The original information is then converted into binary format. Let the obtained binary format be termed as X'. We then apply the complementary base pairing rules on X'. The complementary base pairing rule used here is ((AC)(CG)(GT)(TA)). A represents Adenine, C represents Cytosine, G represents

Guanine and T represents Thymine.

X'' is the form obtained after applying the base pairing rules. We then find the index of each couple of nucleotides in the DNA reference sequence that is specified explicitly to generate the secret data X'''.

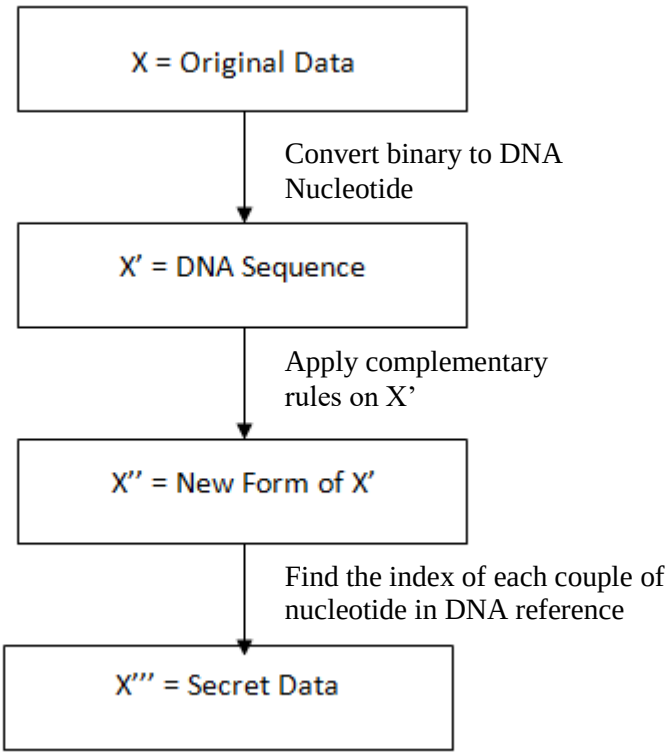



Fig 4.3 Biometrics based algorithm for encryption

In the developed web application, the customer chooses to send the original information M to cloud computing settings across the internet. To offer the ultimate version of M, which is M'", and upload it to the internet, there are 3 channel-phases. The data M is translated into binary form after it hasbeen read as an integers.

The base pairing rules must be used in order to translate binary information into amino acids represented as the sequence of DNA. In the actual world of biology, nucleotide synthesis follows set guidelines.

Example:
Assume original data X=01000001(A) should be uploaded to the cloud
• DNA Reference Sequence:
[TG, TA, AT, GC, CT, GA, CA, AC, AA, GT, CG, AG, CC, TT, TC, GG]
[TG00,TA01,AT02,GC03,CT04,GA05,CA06,AC07,AA08,GT09,CG10,AG11,CC12,TT13, TC14,GG15]

- X=01000001 (Original data).
- Sub-phase1 (Base pairing rule) (A= 00, T= 01, C= 10, G= 11): X´= TAAT
- Sub-phase2 (Applying complimentary rule) ((AC) (CG) (GT) (TA)): X´´= ACCA
- Sub-phase3 (Indexes): X´´´= 0706(Encrypted data)

## 4.4 Procedure for Decryption

The secret data obtained in the previous step is the input for decryption. X''' is the secret data. Find the index of each couple of nucleotides in the DNA reference sequence. Now we obtain X'' which is the previous form of X'. When complementary rules are applied on X''', the DNA sequence X' is obtained. This X' is in its binary form. Upon conversion of this binary data into nucleotide, original data X is obtained.

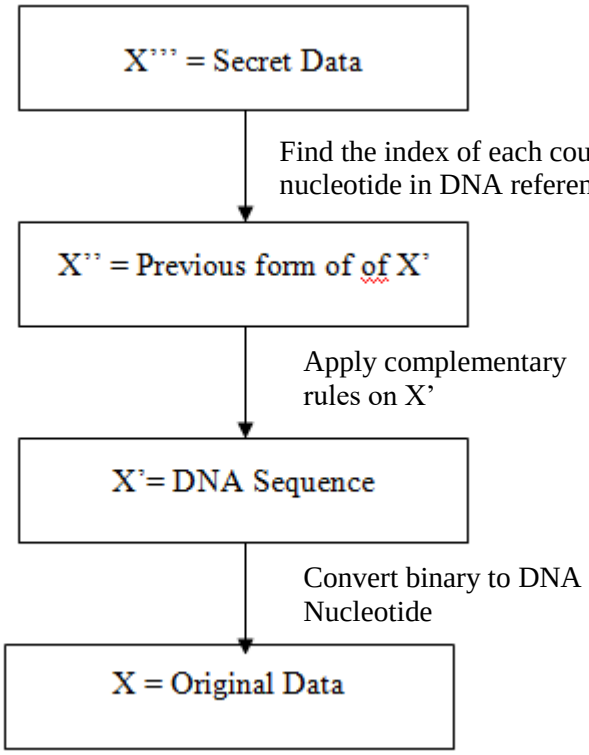



Fig 4.4 Biometrics based algorithm for decryption

While a user uploads a file from his local computer to a website, the file must first be forwarded to the DNA Encryption Service, which runs on a cloud server 1. Cloud server 1 will encrypt the file, and cloud the computer 1 has to transmit the encrypted file to cloud server 2, using the internet service notion, where it will be stored.

Example:
DNA Reference sequence:
[TG00,TA01,AT02,GC03,CT04,GA05,CA06,AC07,AA08,GT09,CG10,AG11,CC12,TT13, TC14, GG15]
[TG, TA, AT, GC, CT, GA, CA, AC, AA, GT, CG, AG, CC, TT, TC, GG]

- M´´´=0706 (Input)
- By referring the DNA sequence:
  Sub-phase1 (Indexes): M´´= ACCA.
- By using Complementary rule:
  Sub-phase2 ((AC) (CG) (GT) (TA)):M´= TAAT
- By using Base Pair Rule:
  Sub-phase3 (A= 00, T= 01, C= 10, G= 11): M=01000001 (A)(Output)

# Chapter 5

# SYSTEM DESIGN

## 5.1 Logical Design

A computer system or software program is designed logically during this stage, which places less emphasis on the actual implementation details and more on the functionality, features, and information flow of the system.

It entails employing models like data flow diagrams, entity-relationship diagrams, and other modeling approaches to provide a high-level representation of the system's architecture, parts, and relationships. Creating a functional plan for the system that both technical and non-technical stakeholders can understand is the aim of logical design.

The physical design stage, when the system's architecture is transformed into an actual implementation that can be run on hardware and software, is a crucial one in the system development life cycle because it serves as the foundation for it.

Logical design is an essential step in the system development life cycle, as it forms the basis for the physical design phase, where the system's architecture is translated into an actual implementation that can be executed on hardware and software platforms. By defining the system's logical structure early in the development process, potential design flaws or issues can be identified and addressed before implementation, saving time and resources in the long run.

## 5.2 Design Goals

Regardless of the application domain, size, or complexity, design objectives are applicable to all web apps. The following are some characteristics that are applicable to web applications.

1. Simplicity
2. Complexity
3. Identity
4. Visual appeal
5. Compatibility

The design process involves several activities, which can vary depending on the nature of the project and the design methodology used. However, the following activities are commonly involved in the design process:

Requirement gathering: In this activity, the design team gathers and analyzes the requirements and specifications for the system or product to be designed. This involves understanding the needs of the end-users, the business objectives of the project, and any technical constraints or limitations.

Research and analysis: This activity involves researching and analyzing various design options and alternatives based on the requirements gathered in the previous step. The team may conduct user research, competitor analysis, and other studies to inform their design decisions.

Conceptual design: In this activity, the design team creates and evaluates different conceptual designs that meet the requirements and objectives of the project. This may involve creating sketches, wireframes, or other low-fidelity prototypes to explore different design ideas.

Detailed design: Once a conceptual design has been selected, the team moves on to detailed design. This involves creating detailed specifications, design documents, and high-fidelity prototypes that provide a complete picture of the final product or system.

Implementation: In this activity, the design is translated into a working prototype or system. This may involve programming, coding, or building physical prototypes.

Testing and evaluation: Once the implementation is complete, the design team tests the system or product to ensure that it meets the requirements and objectives of the project. Any issues or bugs are identified and addressed during this phase.

Iteration: The design process is often iterative, meaning that the design team goes through several cycles of refining the design based on feedback and testing results. This may involve revisiting earlier stages of the design process to make adjustments and improvements.

Documentation and maintenance: Finally, the design team documents the design process and the final product or system. Maintenance and support plans are put in place to ensure that the system or product remains functional and up-to-date over time.

## 5.3 The Model View Controller Architecture

In software development, the Model-View-Controller (MVC) architecture pattern is frequently used to divide the application functionality into three interconnected parts: the Model, the View, and the Controller.

The application's data and business logic are represented by the model. It is in charge of processing all data-related processes as well as maintaining and managing the data. The User Interface and Presentation Layer are not intended to affect the Model in any way.

The View is in charge of providing the user with an intuitive display of the data. In order to prevent any changes to the View from affecting the functionality of the program, it is intended to be independent from the Model and the Controller.

The Model and the View are connected through the Controller. It interacts with the Model to process user input that it receives from the View. The operation's results are then updated in the View. The Controller is in charge of carrying out the application logic and directing how the Model and View communicate with one another.

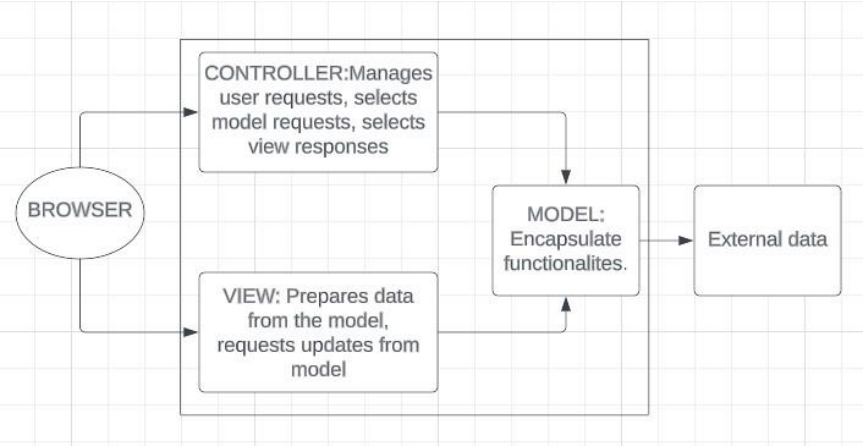



**Fig. 5.3.** J2EE uses MVC Architecture

The MVC architecture has several benefits for software development, including:

Separation of concerns: The application logic is divided into three separate parts, which makes the code simpler to read, modify, and maintain.

Reusability: The components can be used in other applications with different views and controllers because they are independent.

Testability: The ability to test the Model, View, and Controller independently makes it simpler to find and address problems.

Scalability: To accommodate the needs of various projects, the architecture can be scaled up or down.

## 5.4 Struts in Java

A well-liked web application framework for creating Java-based enterprise applications is called Struts. It offers a suite of tools and components for creating scalable and maintainable online applications and is based on the Model-View-Controller (MVC) architectural paradigm.

The presentation layer in Struts is represented by the view, the business logic and data access layer by the model, and the controller is implemented by the action classes. ActionServlet is a servlet controller provided by Struts that receives and controls all incoming client requests. These requests must be processed by the Action classes, which then send them to the appropriate resources for processing. The JSPs that make up Struts View are responsible for serving data to the client. Struts and form beans can be used in conjunction with JSPs to provide interactive user interfaces that communicate with the server-side model.

Several technologies, like JDBC, Hibernate, or JPA, can be used to create the Struts Model in order to manage the data and offer access to the database. Struts offers a collection of helper classes, such ActionForm, to make communication between the Model and the Controller easier.

## 5.5 Software Architecture

Figure 5.5 depicts the system architecture.

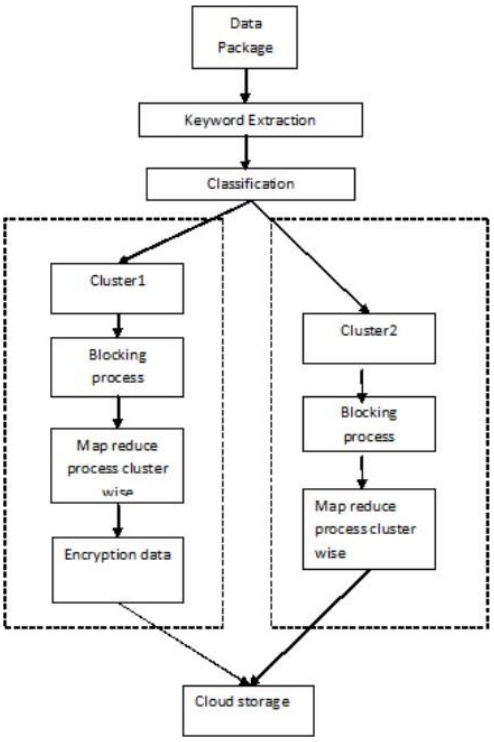



**Fig. 5.5.** System architecture

In figure 5.5 :-

Data package :- Consists of all the files that needs sensitive data to be encrypted.

Keyword extraction :- Sensitive data which if leaked will be a threat to privacy will be extracted by the algorithm.

Classification :- Sensitive data extracted is classified by dividing it into two clusters – cluster 1 and cluster 2.

Cluster 1 :- blocking process is carried out . Then map reduce process cluster wise is carriedout and data is encrypted and stored in cloud.

Cluster 2 :- Blocking process is carried out. Then Map Reduce process is carried out cluster wise and data is stored in cloud without encryption.

Then the data stored in cloud from both clusters is compared and evaluated.

## 5.6 Class Diagrams

A sort of UML (Unified Modelling Language) diagram called a class diagram is used to illustrate the classes and connections among them in an object-oriented software system. A class diagram illustrates the classes, their properties, methods, and connections between them in order to depict the structure of the system.

A class is a description of the characteristics and actions of things that fall under that class. A class is shown as a rectangle with the class name written inside it in a class diagram. Following the class name is a list of the class's attributes, followed by a list of its methods.

Figure 5.4 depicts the class diagram which portrays the functionalities available to the user accessing the website.

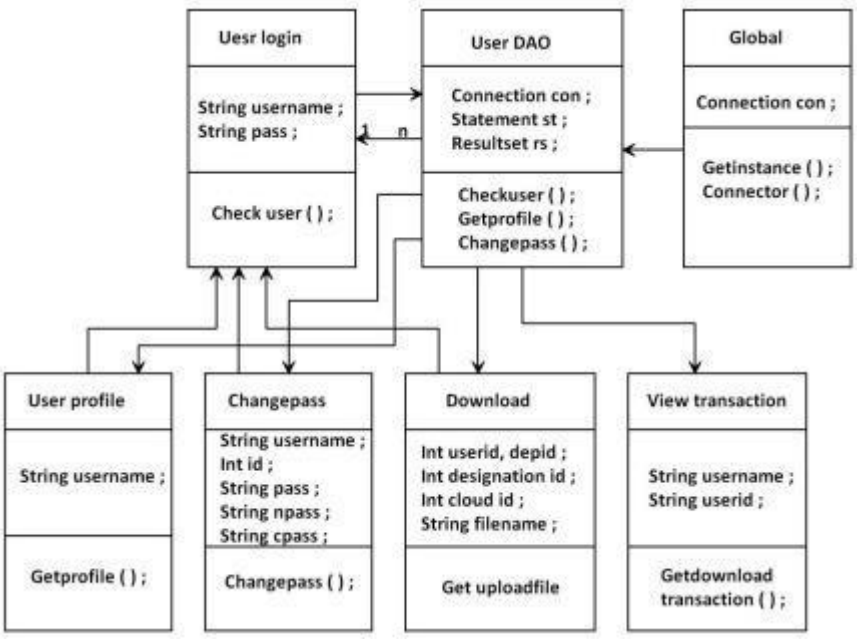



**Fig. 5.6.** Class diagram for user

## 5.7 Use Case Diagrams

Use case diagrams are a subset of UML (Unified Modelling Language) diagrams that are used to show how users (actors) interact with a system or software programme. By displaying the many use cases and system actors, a use case diagram offers a high-level picture of the functioning of the system.

A use case represents a particular activity or feature that the system must carry out in order to accomplish a particular objective. Each use case outlines the stages or actions the system must take to complete a certain activity or objective. The people, systems, or other entities that interact with the system are represented by actors, on the other hand.

Each use case is depicted in a use case diagram as an oval shape with a caption that details the

particular job or objective. On the outside of the use case diagram, actors are shown as stick figures or other shapes. Arrows show the relationships between the actors and the use cases they are associated with.
Figures 5.7a and 5.7b depicts the use case diagrams for user and admin respectively.

The user first registers before logging in. The file to be uploaded is selected. The file is encrypted if it contains sensitive data. While downloading, the file is decrypted. The file is also split into blocks and uploaded to the FTP cloud server.

The admin can keep a track of the registered users after logging in.

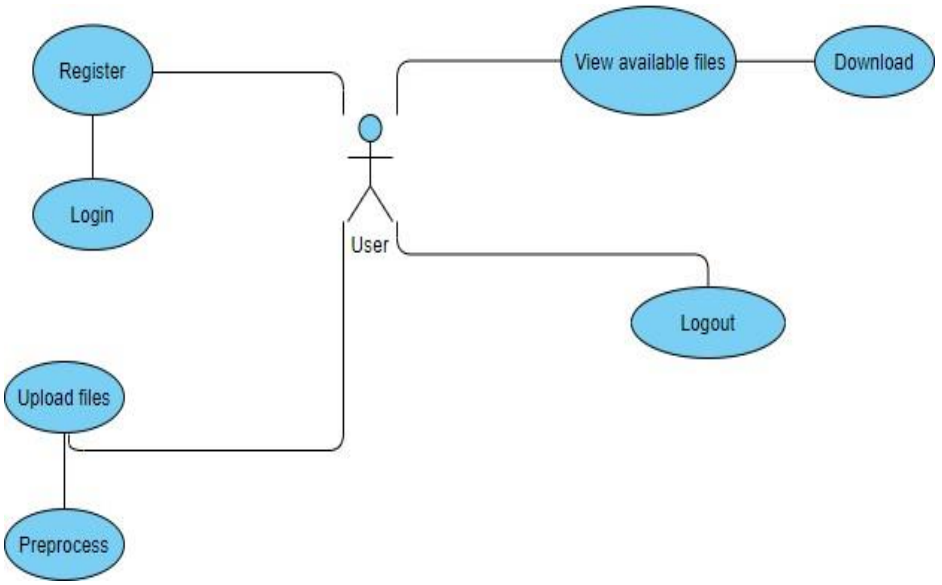



**Fig. 5.7a.** Use case diagram for user

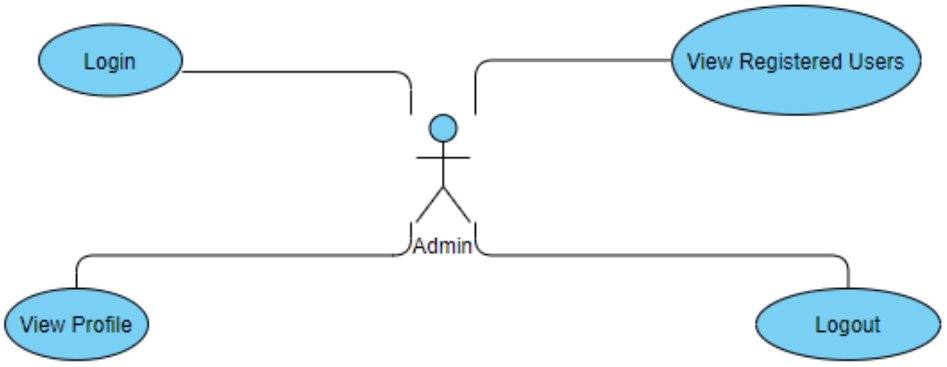



**Fig. 5.7b.** Use case diagram for admin

## 5.8 Dataflow Diagrams

A data flow diagram (DFD) is a picture that shows how information moves through a computer program or system. It is a graphical representation of the system's operations, data sources, data flows, and external partners.

Processes in a DFD stand in for the actions or duties carried out by the system, including data input, data processing, and data output. Data stores, like a database or a file, are locations where data is kept. The transfer of data between processes, data stores, and external entities is represented by data flows. The sources or destinations of data that are external to the system, such as users, clients, or other systems, are represented by external entities.

Figure 5.6a and 5.6b gives the dataflow diagram for the user.

The user first registers before logging in. The file to be uploaded is selected. The file is encrypted if it contains sensitive data. While downloading, the file is decrypted. The file is also split into blocks and uploaded to the FTP cloud server.

The admin can keep a track of the registered users after logging in.

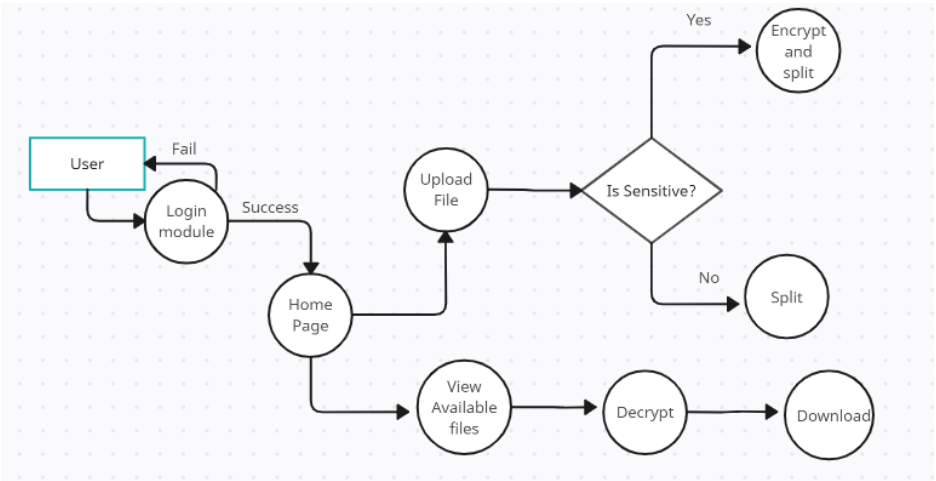



**Fig. 5.8a.** Dataflow diagram for user

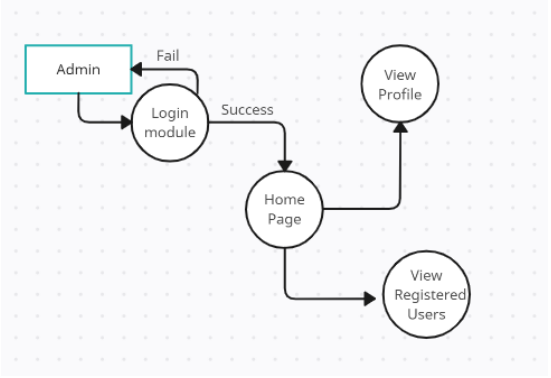



**Fig. 5.8b** Dataflow diagram for admin

## 5.9 Sequence Diagram

A sequence diagram is a form of UML (Unified Modelling Language) diagram that depicts the communications and interactions that take place within a software programme or system. It displays the direction of communication as well as the sequence in which messages are passed back and forth among various system components.

Vertical lines, known as lifelines, are frequently used in sequence diagrams to indicate the various system objects or components. Arrows that indicate the direction of communication between the lifelines serve as a representation of messages. Each message is identified by its name and any parameters that were supplied along with it.

Sequence diagrams are frequently used to represent intricate relationships between items or system components. They can be used to find potential design flaws or to test the system's performance in various scenarios.

Arrows in a sequence diagram represent the communication path between system elements or objects. The label on the arrow identifies the kind of communication being transmitted, and the direction of the arrow indicates the direction of message flow.
Sequence diagrams frequently use three types of arrows:
The arrowhead: The arrowhead indicates the direction of the message flow and identifies the sending and receiving objects or components for each message.
The vertical line: The object or component receiving the message's lifeline is symbolised by the vertical line. It displays the order in which things happen or what is done during the message exchange.
The message's heading: The message type is specified by the label on the arrow.

Figure 5.9 gives the sequence diagram of the system. This sequence diagram portrays the order of events that take place when a user is accessing the system. Starting from login to downloading the file, all the events that occur are given in the form of a sequence diagram.

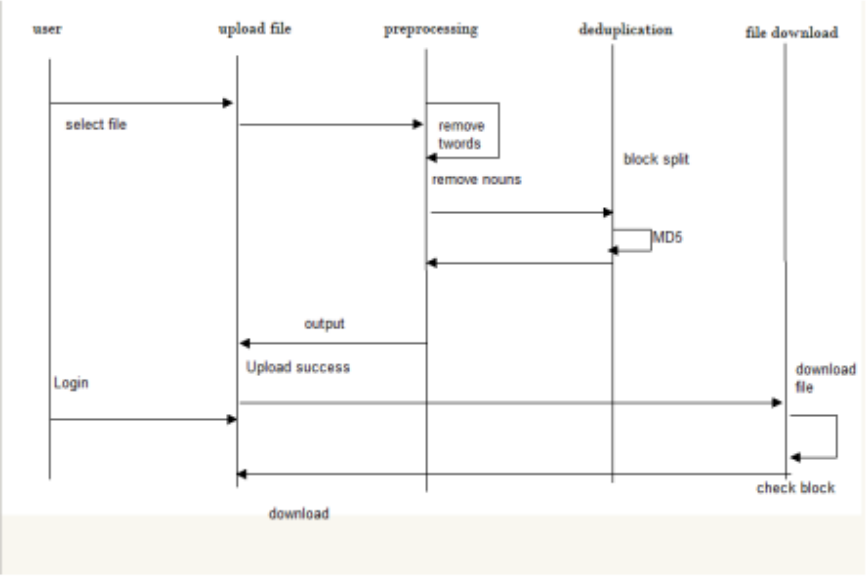



**Fig. 5.9.** Sequence diagram for the system

## 5.10 Context Analysis Diagram

A particular kind of diagram used in software development and systems engineering to show the connections between a system and its surrounding environment or external entities is a context analysis diagram. A context analysis diagram serves the objective of identifying and analysing the external elements, such as users, other systems, and physical or legal restrictions, that have an impact on the system.

A box in the centre of a context analysis diagram represents the system being analysed, and boxes on either side of it represent external entities or environmental elements that affect the system. The boxes are linked together by lines or arrows that represent the movement of materials or information between the system and its surroundings.

Figure 5.10 gives the context analysis diagram.

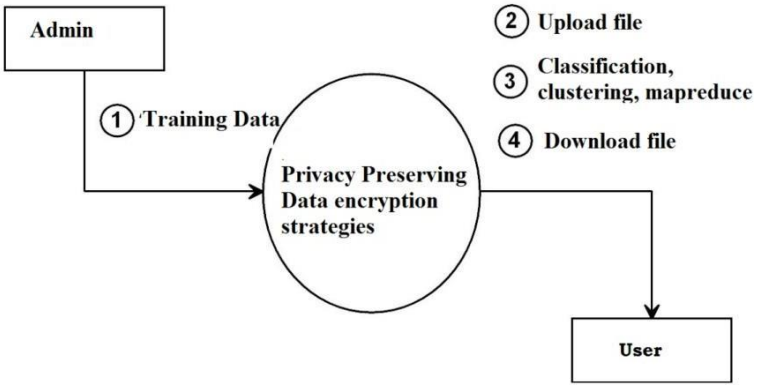



**Fig. 5.10.** Context Analysis Diagram

# Chapter 6

# IMPLEMENTATION

## 6.1 User Homepage

Figure 6.1 depicts the page through which users will interact with.Developed using HTML , CSS and Bootstrap.

```
<nav class="navbar navbar-default" role="navigation">
    <div class="container">
        <div class="navbar-header">
            <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#bs-example-navbar-collapse-1">
                <span class="sr-only">Toggle navigation</span>
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
            </button>
            <!-- navbar-brand is hidden on larger screens, but visible when the menu is collapsed -->
            <a class="navbar-brand" href="index.html">PPDES</a>
        </div>
        <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
            <ul class="nav navbar-nav">
                <li>
                    <a href="<%=request.getContextPath()%>/Profile" target="myframe">View Profile</a>
                </li>

                <li>
                   <a href="<%=request.getContextPath() %>/Files/jsp/upload_file.jsp" target="myframe">Upload File</a>
                </li>
                <li>
                  <a href="<%=request.getContextPath() %>/DownloadFile?submit=get"  target="myframe">Download_file</a>
                </li>
                 <li>
                  <a href="<%=request.getContextPath() %>/ChangePass?id=<%=id%>&no=1" target="myframe">Change Password</a>
                </li>
                 <li>
                  <a href="<%=request.getContextPath()%>/SignOut?no=1">LogOut</a>
                </li>
            </ul>
        </div>
```

**Fig. 6.1.** Code for user homepage

## 6.2 Admin Homepage

Figure 6.2 depicts the page through which admin will interact with .Developed using HTML CSS and Bootstrap.

```
<div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
    <ul class="nav navbar-nav">
        <li>
            <a href="<%=request.getContextPath()%>/AdminProfile" target="myframe">View Profile</a>
        </li>

         <li>
            <a href="<%=request.getContextPath() %>//Files/jsp/admin/users.jsp" target="myframe">View Users List</a>
        </li>


         <li>
          <a href="<%=request.getContextPath() %>/AdminChangePass?id=<%=id%>&no=1" target="myframe">Change Password</a>
        </li>
         <li>
          <a href="<%=request.getContextPath()%>/SignOut?no=1">LogOut</a>
        </li>
    </ul>
</div>
```

**Fig.6.2.** Code for admin homepage

## 6.3 Packages for file upload

Figure 6.3 depicts the packages that contain methods that are needed to upload files are imported.

```
import org.apache.commons.fileupload.FileItem;
import org.apache.commons.fileupload.FileItemFactory;
import org.apache.commons.fileupload.disk.DiskFileItemFactory;
import org.apache.commons.fileupload.servlet.ServletFileUpload;
import org.apache.commons.fileupload.util.Streams;
import org.apache.commons.io.FilenameUtils;
```

**Fig. 6.3.** Code for importing packages

## 6.4 Read contents of file

Figure 6.4 has the code that reads the contents of the file.

```
ArrayList<String> key_data = ReadFile.readfile(fileName);
System.out.println("Arraylist Size :"+key_data.size());
 String counts=TextSearch.text_count(key_data,file.getName());
 String counters[]=counts.split("~~");
```

**Fig. 6.4.** Code for reading contents

## 6.5 Removal of unnecassary information

Figure 6.5 has the code to remove stop words in the content of the file.

```
for(int i=0;i<s.length;i++)
{

    if(s[i].trim()!=null)
    {
        String[] word = s[i].split("~");

        for(int j=0;j<word.length;j++)
        {

            String search = word[j].trim();
            search.trim();

            System.out.println("Keyword :"+search);

            if(search.trim().equals("@") || search.trim().equals("#") || search.equals("&") ||search.equals("%") || search.equals("^"))
            {

                System.out.println(" Unnessary Data 1: "+search);
            }
            else if(search.equals(".") ||search.equals(",") || search.equals("*")||search.equals("~") ||search.equals("!") )
            {

                System.out.println(" Unnessary Data : "+search);
            }
            else
            {
                boolean f = UserDAO.check(search);
```

**Fig. 6.5.** Code to remove unwanted info

## 6.6 Split files into blocks before encryption

Figure 6.6 contains the code to split files into blocks.

```
try
    {
     filePart = new FileOutputStream(new File(root2+"\\"+fno+"blk_"+Integer.toString(nChunks - 1)));
     file5=new File(root2+"\\"+fno+"\\blk_"+Integer.toString(nChunks - 1));
    System.out.println("new filepart is "+filePart);
    String filepathh=root2+"\\"+fno+"blk_"+Integer.toString(nChunks - 1);
    filePart.write(byteChunkPart);
    filePart.flush();
    filePart.close();
    byteChunkPart = null;
    filePart = null;
    String hashcode=MD5.MD(new File(filepathh));
    String filename2= file5.getName();
    String blockname1 = filename2.replaceAll("[0-9]","");
    String blockname2 = blockname1.replaceAll("_","");
        // String blockname2=newFileName;
    System.out.println("===========Perferct Block name======"+blockname2);
    String blkname=fno+"blk_"+Integer.toString(nChunks - 1);
    System.out.println("<<<<<<<<<<+blkname"+blkname);

      String blockidandflag=ClassifyDAO.insertblockname(fno,blkname,hashcode);
      String a[]=blockidandflag.split("~~");
      int blockid=Integer.parseInt(a[0]);
      String flag7=a[1];
                DNA_Main.DNA_Encrypt_Main(root1,filepathh);
                File file1=new File(filepathh);
                FileUpload.upload(ftpserver,ftpusername,ftppassword,blkname,file1,dirToUploadFile);
     sb.append(blockid);
     sb.append("-");

    }
```

**Fig. 6.6.** Code to split files

## 6.7 Upload and preprocess file

Figure 6.7 has the code to upload the files to preprocess it. Once the user upload the file, it should be checked to see if it contains sensitive data.

```
<div style="position: absolute;top: 200px;left: 250px;">
  <input type="submit" value="Preprocess" onclick="return checkPassword()" class="submit"/></div>
</form>

<%
    if(Utility.parse(request.getParameter("no"))==1)
    {%>
        <div class="success" id="message" style="position:absolute;top:-10px;font-size: 20px;color:#994c00;font-family: monotype corsiva;">
            <p>Preprocessing Completed!</p>
        </div>
    <%}
    if(Utility.parse(request.getParameter("no"))==2)
    {%>
        <div class="success" id="message" style="position:absolute;top:-10px;font-size: 20px;color: #994c00;font-family: monotype corsiva;">
            <p>Oops something went wrong.....!</p>
        </div>
    <%
    }
    if(Utility.parse(request.getParameter("no"))==3)
    {%>
        <div class="success" id="message" style="position:absolute;top:-10px;font-size: 20px;color: #994c00;font-family: monotype corsiva;">
            <p>File Already Exist!</p>
        </div>
    <%
    }
    if(Utility.parse(request.getParameter("no"))==5)
    {%>
        <div class="success" id="message" style="position:absolute;top:-10px;font-size: 20px;color: #994c00;font-family: monotype corsiva;">
            <p>This File does not belong to Particular Category!!!!</p>
        </div>
    <%
    }
    %>
```

**Fig. 6.7.** Code for uploading files

## 6.8 Files are classified as sensitive and non sensitive

Figure 6.8a and Figure 6.8b contain the code which classifies data to sensitive or non-sensitive

If data consists of sensitive words it is treated as a sensitive file which needs to be encrypted. If the file does not contain sensitive words it is treated as a non sensitive file which needs no encryption.

```
if(count1>count2)

{
    sensitive = request.getRealPath("") + "/Uploaded_Files/Cluster_1/"+ file.getName();

    Fileblockpath = request.getRealPath("") + "/File_Blocks/";

    System.out.println("fileName CAR===="+sensitive);

   OutputStream outputStream1 = new FileOutputStream(sensitive);
   InputStream inputStream1 = file.getInputStream();

   int readBytes1 = 0;
   byte[] buffer1 = new byte[10000];
   while ((readBytes1 = inputStream1.read(buffer1, 0, 10000)) != -1)
   {
       outputStream1.write(buffer1, 0, readBytes1);
   }
   outputStream1.close();
   inputStream1.close();

   ff=new File(sensitive);
   int clusterid=1;
   int f_no=ClassifyDAO.insertinto_m_file_new(filename,clusterid);
   System.out.println("fno"+f_no);
   catname =ClassifyDAO.getClustername(clusterid);
   System.out.println("<<<<<<<<<<<<<<"+filename);
   str = Integer.toString(f_no);
   System.out.println("str:"+str);
}
```

**Fig. 6.8a.** Code for sensitive file

```
if(count2>count1)
{
    String nonsensitive = request.getRealPath("") + "/Uploaded_Files/Cluster_2/"+ file.getName();
    String Fileblockpath = request.getRealPath("") + "/File_Blocks/";
    System.out.println("fileName digestive===="+nonsensitive);

       OutputStream outputStream1 = new FileOutputStream(nonsensitive);
       InputStream inputStream1 = file.getInputStream();

       int readBytes1 = 0;
       byte[] buffer1 = new byte[10000];
       while ((readBytes1 = inputStream1.read(buffer1, 0, 10000)) != -1)
       {
           outputStream1.write(buffer1, 0, readBytes1);
       }
       outputStream1.close();
       inputStream1.close();

       int clusterid=2;
       ff=new File(nonsensitive);
       int f_no=ClassifyDAO.insertinto_m_file_new(filename,clusterid);
       catname =ClassifyDAO.getClustername(clusterid);
       Packet2.formPacket(ff, nonsensitive,Fileblockpath,f_no);
}
```

**Fig. 6.8b.** Code for non sensitive file

## 6.9 HTML

A markup language called HTML, or HyperText Markup Language, is used to build online pages and web applications. Using a number of tags and attributes that specify the different aspects of the page, such as headings, paragraphs, images, links, and forms, it enables developers to organise content on a web page.

Early in the 1990s, HTML was initially introduced, and since then it has developed into a widely accepted standard for building websites and software. HTML is a vital tool for web development nowadays and is supported by all current web browsers and system.

**Advantages :-**

1. Simple to learn: Even for novices without any prior coding knowledge, HTML is comparatively simple to learn and understand.

2. All current web browsers and operating systems support HTML, making it a flexible and frequently used markup language.

3. Compatibility with other technologies: Dynamic and interactive web applications can be made by combining HTML with other web technologies like CSS and JavaScript.

4. Accessibility: HTML comes with built-in accessibility features that make it simpler to develop websites and software that is usable by those with disabilities.

5. SEO: HTML offers a structured and organised approach to deliver content, which can help web pages be more visible and rank higher in search results

## 6.10 JavaScript

It is a computer language with a scripting foundation that Netscape Communication Corporation created. Both client and server components of Web-based applications can be created using JavaScript.

It can be used to create client-side software that is performed by a web browser as part of a web page. It can be used to create Web server programs on the server side that take information provided by a web browser, process it, and then modify the browser's display as necessary.

JavaScript is generally used for client-side scripting, which implies that instead of running on the web server, it does so on the user's browser. This enables the creation of dynamic, interactive web pages that may react in real time to user inputs.

Cross-platform compatibility: All current web browsers as well as a wide range of non-browser environments, including desktop and mobile applications, support JavaScript. As a result, it is a flexible language that may be applied in a variety of contexts.

Programming in an object-oriented manner: Because JavaScript is an object-oriented language, programmers can construct reusable code in the form of objects and classes. Code may become more modular and easier to maintain as a result.

Programming in the background: JavaScript supports programming in the background.

**Syntax :-**

<Script>..... </Script>

<Script Language = “JavaScript”>

JavaScript statements

</Script>

## 6.11 Servlets

A servlet is a server-side Java technology that makes it possible to create dynamic web applications. It is a Java class that is in charge of handling client requests provided via the HTTP protocol and producing responses to them.

Typically, servlets work in tandem with a web server like Apache Tomcat, which accepts incoming requests and routes them to the proper servlet for processing. The suitable answer is subsequently produced by the servlet and sent back to the client.

The following are some of the main benefits and characteristics of using servlets in Java:

Platform independence: Because Servlets are built in Java, they can run on any system that has Java Virtual Machine (JVM) support.

High performance: Servlets are made to be small and effective, making them suitable for managing large numbers of requests.

Reusability: By making it simple to reuse Servlets in other applications, new application development time and effort are reduced.

Security: By incorporating security features like authentication and access control, servlets offer a secure environment for web applications.

Scalability: Servlets can accommodate a high volume of users and requests since they are scalable by design. They can be used to create intricate web applications with high scalability needs.

Development of complicated web applications is made simple by the ease with which Servlets may be connected with other Java technologies, such as JavaServer Pages (JSP) and JavaBeans.

Figure 6.11a depicts the diagram of the servlets. They have no graphical user interface and can be embedded in many different servers because it assumes nothing about the server's protocol or environment.

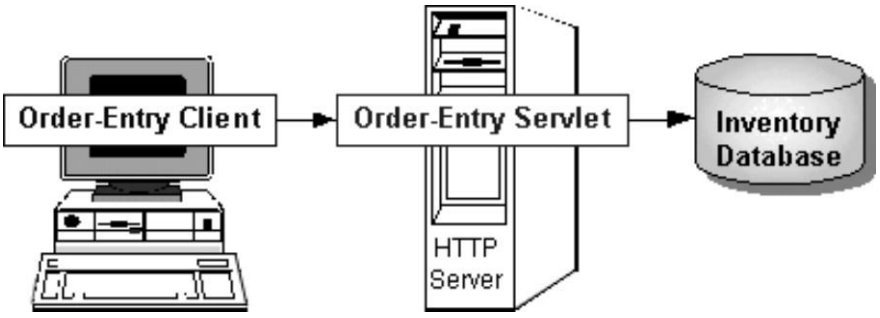



**Fig 6.11a Servlets**

A GenericServlet is an abstract class in the Java language that offers a straightforward implementation of the Servlet interface. Regardless of the specific functionality of each servlet, it defines a set of methods and properties that are shared by all servlets.

The init, service, and destroy methods are provided with default implementations by the GenericServlet class, which also implements the Servlet interface. Additionally, it defines additional methods for setting and retrieving initialization settings as well as for learning more about the context of the servlet.

Developers can design unique servlets with a specific set of features by extending the GenericServlet class and customising its methods. To generate HTML output depending on data from a database or another data source, for instance, a custom servlet might implement the service function

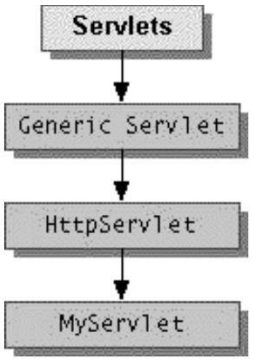



**Fig 6.11b Servlet interface**

## Java server pages :-

The JSP model includes the following components:

JSP pages: JSP pages are HTML pages that contain embedded Java code. They are processed by a web server to generate dynamic content.

Servlets: Servlets are Java components that handle business logic and generate dynamic content. They can interact with JSP pages to retrieve or process data.

JavaBeans: JavaBeans are reusable Java components that encapsulate data and functionality. They can be used in JSP pages or servlets to perform specific tasks.

Tag libraries: Tag libraries provide a set of custom tags that can be used in JSP pages to perform common tasks, such as iterating over data or formatting output.

Figure 6.11c shows the JSP Model. The presentation logic should be implemented in JSPpage and business logic as a part of java bean.

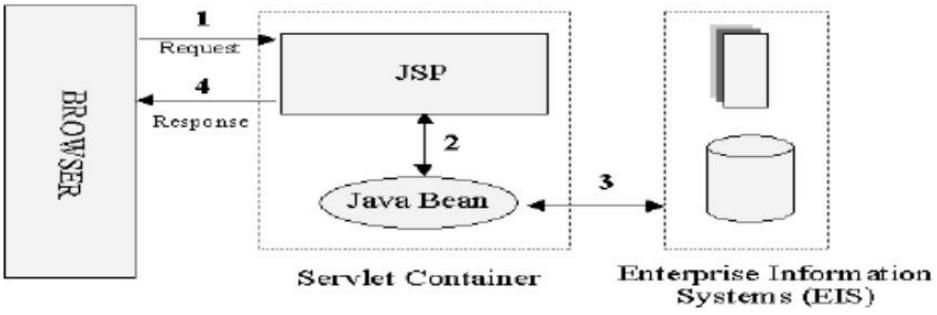



**Fig. 6.11c.** JSP Model

# Chapter 7

# TESTING

## 7.1 Definition

It is the process of evaluating and verifying that a software product does what it is supposed to do. It prevents bugs, reduces development costs and improves performance. Tests are simple short tests that test the functionality of a particular unit or module of their code such as class or function.

## 7.2 Types

- Validation testing: This makes that the application complies with its requirements.
- System testing involves putting the program through its paces in a real-world setting.
- Regression testing: To make sure that the necessary functionalities of earlier versions are still functional in the current version, the program is tested using an automated test harness.
- Recovery testing: To make sure the right procedures for retrieving any information that is lost will work, the program is purposefully disrupted in a variety of ways.
- Security testing is carried out to prevent any unauthorised attempts to use the program, specific features of it, view the data, or damage the program's code or hardware.
- Stress testing: This is when the software is subjected to unusual demands by being required to receive or output information at faster rates than usual.
- Software is tested for performance to make sure the performance criteria are met.
- Usability testing: While the software was being built, usability prototypes were evaluated; nonetheless, validation of the completed product is always necessary.
- Alpha & beta evaluation: This is the stage at which the software is made available to real customers. Alpha releases may be offered to a small group of users who will be asked to report any issues and other specific findings to the team in charge of production.
- Bottom-up integration testing: Each module is tested using a test harness in this method. Once a group of individual modules have undergone testing, they are assembled into builds, and they are then subjected to additional testing by a different test harness. This procedure may be carried out repeatedly until the full application is built.
- Top-down integration testing: In this type of testing, the complete application is examined first. Then it is broken down into smaller modules, and each is examined separately.
- Unit testing: This method focuses on testing a whole unit. This would restrict the test to one unit while testing the interactions of several functions. A unit's precise scope is up to interpretation. Additional test code, sometimes referred to as scaffolding, could be required to

# Chapter 8

# RESULTS

The following snapshots define the results or outputs that we will get after step by step execution of all the modules of the system.


**Fig. 8.1.** User registration

Figure 8.1 allows new users to register using their credentials.


**Fig. 8.2.** User login

Figure 8.2 allows the users to login to their account using their login credentials.


**Fig. 8.3.** Admin login

Figure 8.3 allows admins to log into their account using their login credentials.


**Fig. 8.4.** Admin profile

Figure 8.4 allows admins to view all their information.


**Fig. 8.5.** Upload files

Figure 8.5 shows that the files can be uploaded for encryption.

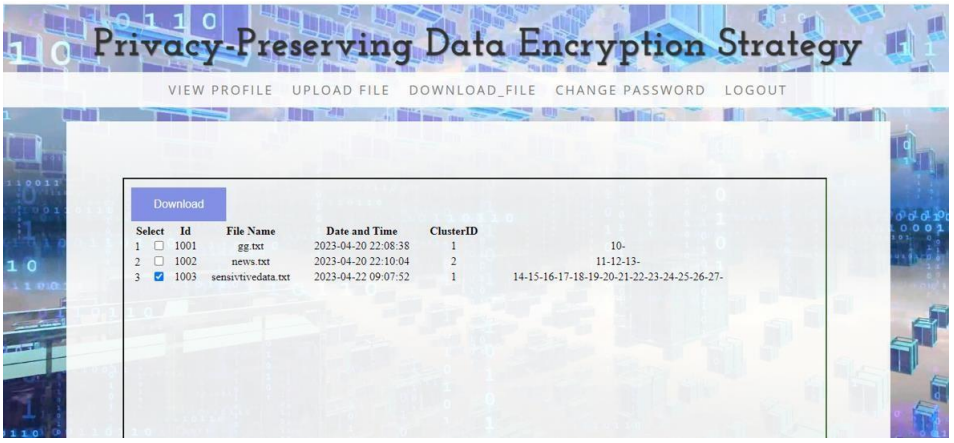



**Fig. 8.6.** Download options

Figure 8.6 depicts that the users can download the files and decrypt it to read its content.

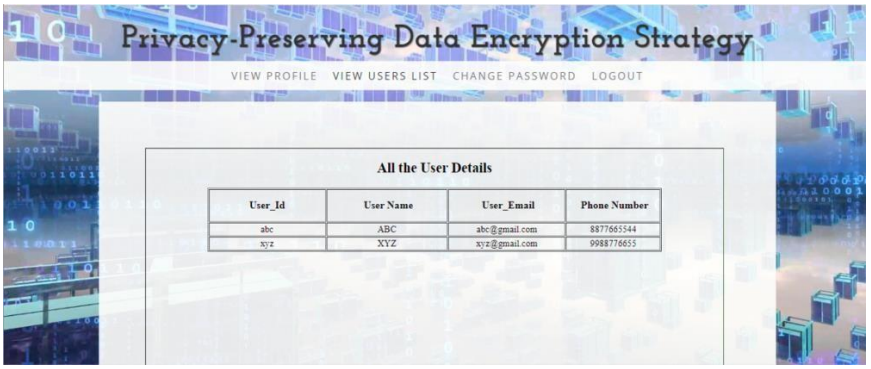



**Fig. 8.7.** User details

Figure 8.7 allows the admin to see all the users who have registered.

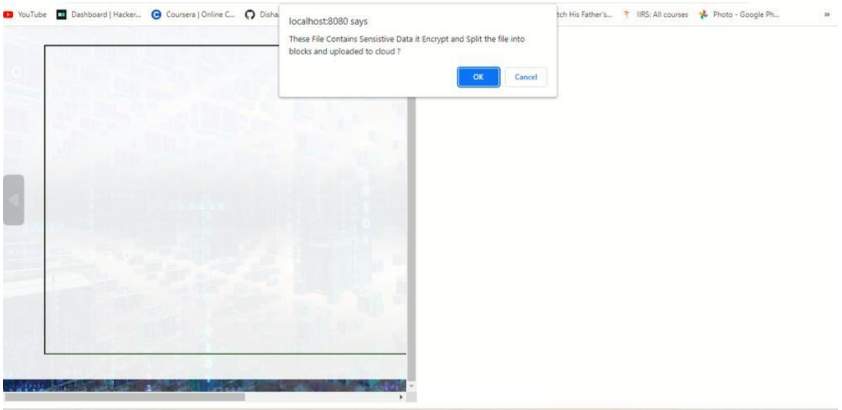



**Fig. 8.8.** Detect state of file

Figure 8.8 detects if the uploaded file is sensitive or not and asks if it should be split and encrypted and stored in cloud.

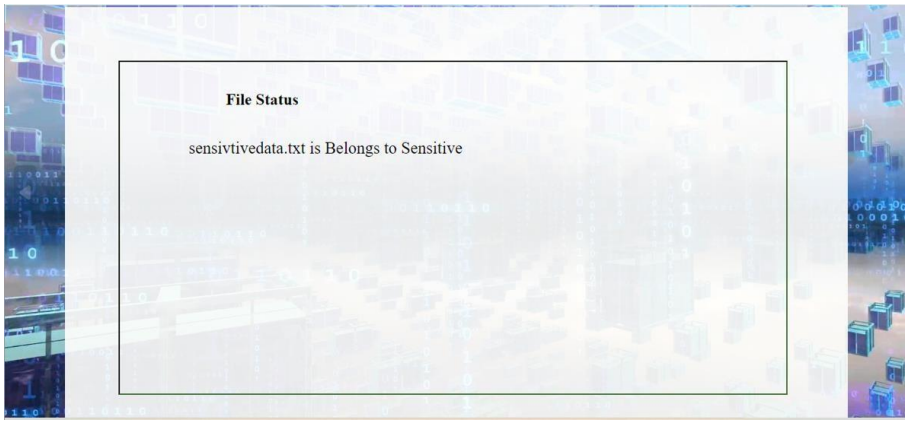



**Fig. 8.9.** Display status of file

Figure 8.9 displays if file is sensitive or non sensitive.

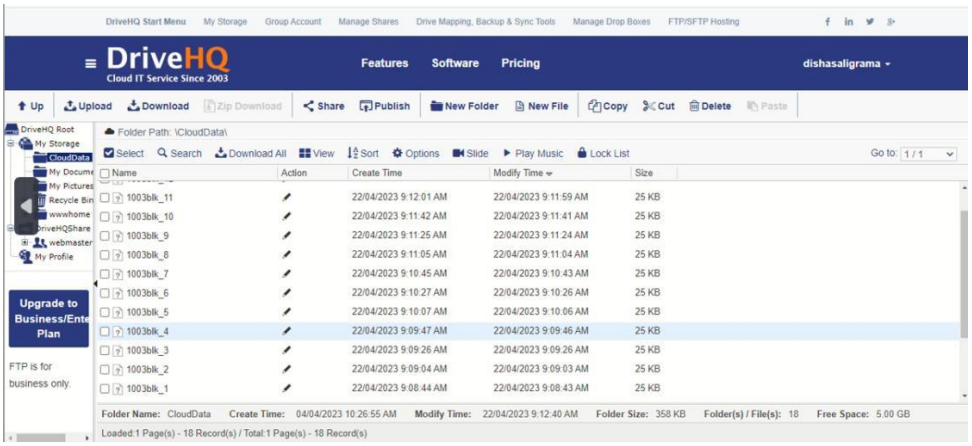



**Fig. 8.10.** Files stored as blocks

Figure 8.10 shows the files are stored as blocks in DriveHQ cloud.

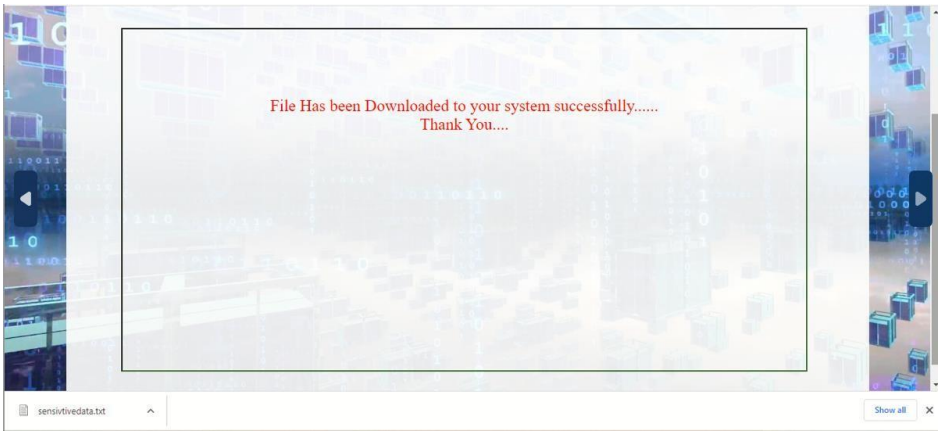



**Fig. 8.11.** File downloaded

Figure 8.11 shows the file has been downloaded successfully.

Table 8.1 contains the list of details on how blocks are made.

**Table 8.1 : -** m_admin

| Column | Type | Default Value | Nullable | Character Set | Collation | Privileges |
|---|---|---|---|---|---|---|
| s_no | int | | NO | | | select,insert,update,references |
| admin_id | varchar(20) | | YES | latin1 | latin1_swedish_ci | select,insert,update,references |
| admin_pwd | varchar(20) | | YES | latin1 | latin1_swedish_ci | select,insert,update,references |
| admin_name | varchar(20) | | YES | latin1 | latin1_swedish_ci | select,insert,update,references |
| email_id | varchar(20) | | YES | latin1 | latin1_swedish_ci | select,insert,update,references |
| phone | varchar(20) | | YES | latin1 | latin1_swedish_ci | select,insert,update,references |

Table 8.2 contains the list of details on how blocks are made.

**Table 8.2 :-** m_blocks

| Column | Type | Default Value | Nullable | Character Set | Collation | Privileges |
|---|---|---|---|---|---|---|
| id | int | | NO | | | select,insert,update,references |
| f_no | int | | YES | | | select,insert,update,references |
| blocks | varchar(50) | | YES | latin1 | latin1_swedish_ci | select,insert,update,references |
| hash_code | varchar(50) | | YES | latin1 | latin1_swedish_ci | select,insert,update,references |
| instance | int | | YES | | | select,insert,update,references |

Table 8.3 contains list of details related to clusters.

**Table8. 3 :-** m_cluster

| Column | Type | Default Value | Nullable | Character Set | Collation | Privileges |
|---|---|---|---|---|---|---|
| id | int | | NO | | | select,insert,update,references |
| f_no | int | | YES | | | select,insert,update,references |
| key_word | varchar(200) | | YES | latin1 | latin1_swedish_ci | select,insert,update,references |
| key_rank | int | | YES | | | select,insert,update,references |
| cluster_id | int | | YES | | | select,insert,update,references |

Table 8.4 contains the file related details.

**Table8. 4 :-** m_files

| Column | Type | Default Value | Nullable | Character Set | Collation | Privileges |
|---|---|---|---|---|---|---|
| f_no | int | | NO | | | select,insert,update,references |
| f_name | varchar(50) | | YES | latin1 | latin1_swedish_ci | select,insert,update,references |
| date_and_time | varchar(100) | | YES | latin1 | latin1_swedish_ci | select,insert,update,references |
| cluster_id | int | | YES | | | select,insert,update,references |
| lba | varchar(100) | | YES | latin1 | latin1_swedish_ci | select,insert,update,references |

# Chapter 9

# CONCLUSION

This project has demonstrated the potential of DNA cryptography as a secure and efficient method of data encryption. Through the use of DNA sequences as keys, information can be encoded in a way that is virtually unbreakable, providing an additional layer of security that could have significant implications for the protection of sensitive data. Although there are still limitations and challenges to be addressed, such as the cost and time required for DNA sequencing and synthesis, the future of DNA cryptography looks promising. As technology advances and more research is conducted in this area, we can expect to see further development and refinement of this innovative method of encryption.

There are numerous areas for future work in DNA cryptography. One potential avenue for research is the development of more efficient and cost-effective methods for DNA synthesis and sequencing. This could help to reduce the barriers to entry for this technology and make it more widely accessible. Additionally, further studies could be conducted to explore the potential of DNA-based encryption for large-scale applications, such as in cloud computing or the internet of things. As this technology continues to evolve and mature, it has the potential to revolutionize the field of cryptography and enhance the security of our digital world. Ultimately, the future of DNA cryptography holds great promise and will be an exciting area to watch in the years to come.

.

# REFERENCES


[1] Privacy-Preserving Data Encryption Strategy for Big Data in Mobile Cloud Computing Environment Sumit Vikram Tripathi , Ritukar 2, Prof. Murthy B3, Dr. K. S. Jagadish Gowda U.G. Student, Department of Computer Science & Engineering,Sri Krishna Institute of Technology, Bengaluru, India1 U.G. Student, Department of Computer Science & Engineering,Sri Krishna Institute of Technology, Bengaluru, India2 Assistant Professor, Department of Computer & Engineering,Sri Krishna Institute of Technology, Bengaluru, India3 Professor, Department of Computer & Engineering,Sri Krishna Institute of Technology, Bengaluru, India

[2] Vikram, A., Kalaivani, S., & Gopinath, G. (2019). A Novel Encryption Algorithm based on DNA Cryptography. 2019 International Conference on Communication and Electronics Systems (ICCES). doi:10.1109/icces45898.2019.90023

[3] Gai, H. Zhao, M. Qiu, L. Qiu, and M. Chen. SA-EAST is an effective data transfer method for ITS in mobile heterogeneous cloud computing that is privacy conscious. 2017; 16(2):60, ACM Transactions on Embedded Computing Systems.

[4] A. Sim, M. Churchill, J. Choi, A. Stathopoulos, C. Chang, and S. Klasky are Wu, K. Wu, and other names. Blob-filaments in fusing plasma: approaching spatial characteristic tracing and immediate identification. 2016 issue of IEEE Transactions on Big Data.

[5] S. Guo, M. Guo, W. Zhou, and Yu. a workable architecture for IP traceback using dynamically stochastic session tagging. :1418–1427, IEEE Transactions on Computers, 2016.

[6] R.J. Lipton, "DNA solution of hard computational problems," Science, New Series, 268(5210), 542-545, 1995.

[7] D. Boneh, C. Dunworth, and R.J. Lipton, "Breaking DES Using a Molecular Computer," DIMACS workshop on DNA computing, 27, 37-51, 1995.

[8] Z. Chen, X. Geng, and J. Xu, "Efficient DNA Sticker Algorithms for DES," IEEE 3rd International Conference on Bio-Inspired Computing (BICTA), 15- 22, 2008.

[9] L. MingXin, L. XueJia, X. GuoZhen, and Q. Lei, "Symmetric-key cryptosystem with DNA technology," Science in China Series F: Information Sciences, 50(3), 324-333, 2007.

[10] N. kar, A. Majumder, A. Saha, A. Jamatia, K. Chakma, and M. C. Pal, "An Improved Data Security using DNA Sequencing," Proceedings of the 3rd ACM MobiHoc workshop on Pervasive wireless healthcare, 13-18, 2013.

[11] H.J. Shiu, K.L. Ng, J.F. Fang, R.C.T. Lee, and C.H. Huang, "Data hiding methods based upon DNA sequences," Information Sciences, 180, 2196-2208, 2010.

[12] H. Liu, D. Lin, A. Kadir, "A novel data hiding method based on deoxyribonucleic acid coding," Computers and Electrical Engineering, 39, 1164-1173, 2013.

[13] T. Mandge and V. Choudhary, "A DNA Encryption Technique
Based on Matrix Manipulation and Secure key Generation Scheme," IEEE International conference on Information Communication and Embedded Systems (ICICES), 47-52, 2013.

[14] J.S. Taur, H.Y. Lin, H.L. Lee and C.W. Tao, "Data hiding in DNA sequences based on Table Lookup Substitution," International Journal of Innovative Computing, Information and Control, 8(10), 6585-6598, 2012.

[15] Y.H. Huang, C.C. Chang, C.Y. Wu, "A DNA-based data hiding technique with low modification rates," Multimedia Tools Applications, 1-13, 2012.

[16] M.R. Abbasy, P. Nikfard, A. Ordi, M.R.N. Torkaman, "DNA Base Data Hiding Algorithm," International Journal on New Computer Architectures and Their Applications (IJNCAA), 2(1), 183192, 2012.

[17] O. Tornea, M. Antonini, M. Borda, "Multimedia Data Compression and Encryption using DNA Cipher", Communications Department, Technical University of ClujNapoca, winner of 2nd Prize, album SSET 2013.

[18] W.M. Shih, J.D. Quispe, G.F. Joyce, "1.7-kilobase singlestranded DNA that folds into a nanoscale octahedron", Nature, Vol. 427, pp. 618-621, 2004.

[19] H. Shiu, K. Ng, J.F. Fang, et al., "Data hiding methods based upon DNA sequences", Elsevier Inc., pp. 2196–2208,2010.

[20] M. Reza Najaf Torkaman et al, "Innovative Approach to Improve Hybrid Cryptography by using DNA Steganography", IJNCAA 2012

[21] Zhihua Chen et el. "Efficient DNA Sticker Algorithms for DES", IEEE 3rd international conference on Bio-inspired computing, 2012

[22] A.P. Thiruthuvadoss, "Comparison and Performance Evaluation of Modern Cryptography and DNA Cryptography," M.S. Thesis, Royal Institute of Technology, 2012.

[23] G. Cui, L. Qin, Y. Wang, X. Zhang, "Information Security Technology Based on DNA Computing," International Workshop on Anti-counterfeiting, Security, Identification, 288-291, 2007

[24] M. Babaei, "A novel text and image encryption method based on chaos theory and DNA computing," Natural computing, 12(1), 101-

[25] SCLPV: Secure Certificateless Public Verification for Cloud-Based Cyber-Physical-Social Systems Against Malicious Auditors Yuan Zhang, Student Member, IEEE, ChunxiangXu, Member, IEEE, Shui Yu, Senior Member, IEEE, Hongwei Li, Member, IEEE, and Xiaojun Zhang.