

# **CLOUD COMPUTING SECURITY CHALLENGES, THREATS AND VULNERABILITIES**

*A project Report submitted  
in partial fulfillment for the award of the Degree of*

**Bachelor of Technology in  
Computer Science and Engineering  
by**

<b>THIRULOKESH.N</b>	<b>(U18CS502)</b>
<b>DIVAGAR.P</b>	<b>(U18CS709)</b>
<b>INBA ELAVARASAN.RV</b>	<b>(U18CN257)</b>
<b>SURYA.KS</b>	<b>(U18CS504)</b>

*Under the guidance of*  
**Mrs.: K. ANURANJANI ME., (Ph.D.)**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING SCHOOL  
OF COMPUTING  
BHARATH INSTITUTE OF HIGHER EDUCATION AND RESEARCH  
(Deemed to be University Estd u/s 3 of UGC Act, 1956)**

**CHENNAI 600 073, TAMILNADU, INDIA  
April, 2022**

## CERTIFICATE

This is to certify that the project report entitled Security Challenges and Threats and Vulnerabilities submitted by Thirulokesh.N, Divagar.P, Inba Elavarasan .RV and Surya. KS to the Department of Computer Science and Engineering, Bharath Institute of Higher Education and Research, in partial fulfillment for the award of the degree of **B. Tech in (Computer Science and Engineering)** is a bona fide record of project work carried out by them under my supervision. The contents of this report, in full or in parts, have not been submitted to any other Institution or University for the award of any other degree.

Signature of Supervisor

**Mrs.: K. Anuranjani**

Assistant Professor,

Department of Computer Science & Engineering,

School of Computing, Bharath Institute of Higher Education and Research,

April, 2022

Signature of Head of the Department

**Dr. B. Persis Urbana Ivy**

Professor & Head of the Department of Computer Science

& Engineering,

School of Computing,

Bharath Institute of Higher Education and Research,

April, 2022

INTERNAL EXAMINER

EXTERNAL EXAMINER

## DECLARATION

We declare that this project report titled **Security Challenges and Threats and Vulnerabilities** submitted in partial fulfillment of the degree of **B. Tech in (Computer Science and Engineering)** is a record of original work carried out by us under the supervision of Mrs. Anuranjani and has not formed the basis for the award of any other degree or diploma, in this or any other Institution or University. In keeping with the ethical practice in reporting scientific information, due acknowledgements have been made wherever the findings of others have been cited.

Signature

THIRULOKESH.N

U18CS502

Signature

DIVAGAR.P

U18CS709

Signature

INBA ELAVARASAN.R

U18CN257

Signature

SURYA.KS

U18CS504

Chennai,

Date:

## ACKNOWLEDGMENTS

First, we wish to thank the almighty who gave us good health and success throughout our project work.

We express our deepest gratitude to our beloved President **Dr. J. Sundeeep Aanand**, and Managing Director **Dr.E. Swetha Sundeeep Aanand** for providing us the necessary facilities for the completion of our project.

We take great pleasure in expressing sincere thanks to Vice Chancellor (I/C) **Dr. K. Vijaya Baskar Raju**, Pro Vice Chancellor (Academic) **Dr. M. Sundararajan**, Registrar **Dr. S. Bhuminathan** and Additional Registrar **Dr. R. Hari Prakash** for backing us in this project. We thank our Dean Engineering **Dr. J. Hameed Hussain** for providing sufficient facilities for the completion of this project.

We express our immense gratitude to our Academic Coordinator **Mr. G. Krishna Chaitanya** for his eternal support in completing this project.

We thank our Dean, School of Computing **Dr. S. Neduncheliyan** for his encouragement and the valuable guidance.

We record indebtedness to our Head, Department of Computer Science and Engineering **Dr. B. Persis Urbana Ivy** for immense care and encouragement towards us throughout the course of this project.

We also take this opportunity to express a deep sense of gratitude to our Supervisor **Mrs. K. Anuranjani** for his cordial support, valuable information and guidance, he helped us in completing this project through various stages.

We thank our department faculty, supporting staff and friends for their help and guidance to complete this project.

<b>THIRULOKESH.N</b>	<b>(U18CS502)</b>
<b>DIVAGAR.P</b>	<b>(U18CS709)</b>
<b>INBA ELAVARASAN.RV</b>	<b>(U18CN257)</b>
<b>SURYA.KS</b>	<b>(U18CS504)</b>

## TABLE OF CONTENTS

	DESCRIPTION	PAGE NO
	<b>CERTIFICATE</b>	iii
	<b>DECLARATION</b>	iv
	<b>ACKNOWLEDGEMENT</b>	v
	<b>ABSTRACT</b>	vii
	<b>LIST OF FIGURES</b>	xi
CHAPTER	TITLE	PAGE NO
	<b>1 INTRODUCTION</b>	
1	1.0 Cloud Computing	01
	1.1 Contributions	01
	1.2 Aim & Objective	02
	1.3 Characteristics	02
	<b>2 LITERATURE SURVEY</b>	03
	2.0 Literature review	04
2	2.1 Efficient and verifiable outsourcing Scheme of sequence comparisons	04
	2.2 Secure outsourcing of Sequence Comparisons	04
	2.3 Secure and private of Sequence Comparisons	04
	2.4 New algorithm for Secure Outsourcing Modular Exponentiations	05
	<b>3 PROBLEM ANALYSIS</b>	05
3	3.1 Existing System	06
	3.1.1 Disadvantages	06
	3.2 Proposed System	06
	3.2.1 Advantages of Proposed System	06
	<b>4 PROJECT DESCRIPTION</b>	07
	4.1 System Architecture	07
	4.2 Modules	08
	4.2.1 Login Module	08
	4.2.2 Registration Module	09
	4.2.3 Creation Storage and Instance	09
4	4.2.4 Find Collusion Module	10
	4.2.5 Find Third-Party Module	10
	4.3 Data Flow Diagram	11
	4.3.1 DFD-Level 0	11
	4.3.2 DFD-Level 1	12
	4.3.3 DFD-Level 2	12
	4.4 System Diagram	13

	4.4.1 Use case Diagram	13
	4.4.2 Class Diagram	14
	4.4.3 Sequence Diagram	15
	4.4.4 Activity Diagram	16
	4.4.5 Component Diagram	17
	4.5 Requirement Specifications	18
	4.5.1 Hardware Requirements	18
	4.5.2 Software Requirements	18
	<b>5 SYSTEM DESIGN</b>	19
5	5.1 Input Design	19
	5.2 Output Design	19
	<b>6 SYSTEM DESIGN</b>	20
	6.1 Testing Process	20
	6.2 Types of Tests	20
	6.2.1 Unit Testing	21
	6.2.2 Integration Testing	21
	6.2.3 Functional Testing	21
6	6.2.4 System Testing	22
	6.2.5 White Box	22
	Testing	
	6.2.6 Black Box Texting	22
	6.2.7 Integration Testing	23
	6.2.8 Acceptance Testing	23
	6.2.9 Alpha Testing	24
	6.2.10 Beta Testing	24
	<b>7 Conclusion &amp; Future Enhancement</b>	25
	7.1 Conclusion	25
	7.2 Future Enhancement	25
7	<b>REFERENCES</b>	27
	<b>APPENDICES</b>	27
	<b>APPENDIX 1 (SOURCE CODE)</b>	
	<b>APPENDIX 2 (SCREENSHOTS)</b>	

## **ABSTRACT**

Today data sharing and maintaining its security is major challenge. User in the data sharing system upload their file with the encryption using private key. This property is especially important to any large-scale data sharing system, as any user leak the key information then it will become difficult for the data owner to maintain security of the information. In this paper provide a concrete and efficient instantiation of scheme, prove its security and provide an implementation to show its practicality. There are lots of challenges for data owner to share their data on servers or cloud. There are different solutions to solve these problems. These techniques are very much critical to handle key shared by the data owner. This paper will introduce the trusted authority to authenticate user those who have the access to the data on cloud.

SHA algorithm is used by the trusted authority to generate the key and that key will get share to user as well as the owner. The trusted authority module receives encrypted file using AES Algorithm from the data owner and computes hash value using MD-5 algorithm. It stores key in its database which will be used during the dynamic operations and to determine the cheating party in the system. Trusted authority send file to CSP module to store on cloud. The resulting key sets are shown to have a number of desirable properties that ensure the confidentiality of communication sessions against collusion attacks by another network node

<b>FIGURE NO</b>	<b>NAME</b>	<b>PAGE NO</b>
4.1	System Architecture	8
4.2	Login Module	9
4.3	Registration Module	10
4.4	Creation Storage and Instance	11
4.5	Find Collusion Module	11
4.6	Find Third-Party Module	12
4.7	DFD-Level 0: Data Owner	12
4.8	DFD-Level 1:	13
4.9	DFD-Level 2:	13



# **CHAPTER 1**

## **INTRODUCTION**

### **1.0 Cloud Computing**

One possible solution is to migrate character sequences to public cloud computing platforms and to request that Cloud Service Provider process sequence comparisons. At present, primary sequence comparison algorithms are deployed as a universal outsourcing service on public clouds. But at the same time, its security and privacy issues are increasingly emerging.

The outsourced data stored as plaintext could easily be exposed to malicious external intruders and internal attackers in the CSP, and the individual private information carried by character sequences (e.g., personal identification, financial transaction records, genetic markers for some diseases, information that is used to identify paternity or maternity, etc.) could more or less be disclosed or abused. Therefore, secure outsourcing is designed to protect the privacy of character sequences, and to ensure that the scheduled computing requests are normally performed on the cloud servers.

### **1.1 Contributions**

Our scheme is easy in deployment, efficient in processing and controllable in overhead. The contributions of this paper mainly in the following four aspects.

Based on the universal model of a public cloud outsourcing, we propose an overall architecture for E-SC. This architecture is built on the end user and the unmodified CSP. Its overall system model, which has been demonstrated to be secure under the threat model, is user-friendly and implementation-friendly.

A salted hash algorithm is improved to hash the character sequences and the indexes of cost matrices, so as to defend against statistical attacks.

An additive order preserving encryption algorithm is designed to encrypt the elements of cost matrices. Also, this algorithm can achieve an indistinguishability under additive ordered chosen-plaintext attack with linear time complexity.

A single cloud server works for the first time to provide a privacy-preserving computable outsourcing service to effectively resist collusion attacks from the cloud. With per-processing modules of padding, partition and expansion, there is no need to decrypt any outsourced data in the non-interactive sequence comparison stage.

Simulation results show that the overall execution performance of our E-SC is negatively correlated with its security.

## **1.2 Aim & Objective**

The main objective of this system is, SHA algorithm is used by the trusted authority to generate the key and that key will get share to user as well as the owner. The trusted authority module receives encrypted file using AES Algorithm from the data owner and computes hash value using MD-5 algorithm.

## **1.3 Characteristics**

The main characteristic of a cloud computing includes

- High reliability
- More flexibility
- Low cost
- Provide security
- User comfortable

## CHAPTER 2

### 2.0 LITERATURE REVIEW

S. No	Topic	Author(S)	Focus
1.	Efficient And Verifiable Outsourcing Scheme Of Sequence Comparisons	Y. Feng, H. Ma, and X. Chen	In this paper, we solve the problem of verifiable outsourcing computation of sequence
2.	Secure Outsourcing of Sequence Comparisons	M.J. Atallah and J. Li	We tackle the problem by integrating the technique of garbled circuit with homomorphic encryption
3.	Secure And Private Sequence Comparisons	M.J. Atallah, F. Kerschbaum and W. Du	The similarity between two sequences arises in a large number of applications
4.	New Algorithm for Secure Outsourcing of Modular Exponentiations	X. Chen, J. Li, J. Ma, Q. Tang and W. Lou	Moreover, we prove that both the algorithms can achieve the desired security notions

## **2.1 Efficient and Verifiable Outsourcing scheme of Sequence Comparisons**

With the rapid development of cloud computing, the techniques for securely outsourcing prohibitively expensive computations are getting widespread attentions in the scientific community. In the outsourcing computation paradigm, the clients with resource-constrained abilities can outsource the heavy computation workloads into the cloud server and enjoy unlimited computing resources in a pay-per-use manner. One of the most critical functionalities in outsourcing computation is the verifiability of the result.

## **2.2 Secure Outsourcing of Sequence Comparisons**

One of the most critical functionalities in data outsourcing is verifiability. However, there is very few secure outsourcing schemes for sequence comparisons that the clients can verify whether the servers honestly execute a protocol or not. In this paper, we tackle the problem by integrating the technique of garbled circuit with homomorphic encryption. As compared to existing schemes, our proposed solution enables clients to efficiently detect the dishonesty of servers.

## **2.3 Secure and Private Sequence Comparisons**

The amount of communication done by our protocol is proportional to the time complexity of the best-known algorithm for performing the sequence comparison. The problem of determining the similarity between two sequences arises in a large number of applications, in particular in bioinformatics. In these application areas, the edit distance is one of the most widely used notions of sequence similarity: It is the least-cost set of insertions, deletions, and substitutions required to transform one string into the other.

## **2.4 New Algorithm for Secure Outsourcing Modular Exponentiations**

Modular exponentiations have been considered the most expensive operation in discrete-logarithm based cryptographic protocols. In this paper, we propose a new secure outsourcing algorithm for exponentiation modular a prime in the one-malicious model. Compared with the state-of-the-art algorithm, the proposed algorithm is superior in both efficiency and check ability. We then utilize this algorithm as a subroutine to achieve outsource-secure Cramer-Shoup encryptions and Schnorr signatures. Besides, we propose the first outsource-secure and efficient algorithm for simultaneous modular exponentiations.

## **CHAPTER 3**

### **3. PROBLEM ANALYSIS**

#### **3.1 Existing System**

Large-scale problems in the physical and life sciences are being revolutionized by Internet computing technologies, like grid computing, that make possible the massive cooperative sharing of computational power, bandwidth, storage, and data.

A weak computational device, once connected to such a grid, is no longer limited by its slow speed, small amounts of local storage, and limited bandwidth: It can avail itself of the abundance of these resources that is available elsewhere on the network. Without revealing to the remote agents whose computational power is being used, either one's data or the outcome of the computation on the data.

##### **3.1.1 Disadvantages**

- Secure outsourcing for widely applicable sequence comparison problems
- Risk of Leak of Secret Information

#### **3.2 Proposed System**

We propose a secure data sharing scheme, which can achieve secure key distribution and data sharing for dynamic group. We provide a secure way for key distribution without any secure communication channels. The users can securely obtain their private keys from group manager without any Certificate Authorities due to the verification for the public key of the user.

Our scheme can achieve fine-grained access control, with the help of the group user list, any user in the group can use the source in the cloud

and revoked users cannot access the cloud again after they are revoked.

We propose a secure data sharing scheme which can be protected from collusion attack. The revoked users can not be able to get the original data files once they are revoked even if they conspire with the untrusted cloud.

### **3.2.1 Advantages of Proposed System**

- Power Means of Persuasion and control
- More Reliable
- It's more secure and efficient.
- Data confidentiality

## CHAPTER 4

### 4.PROJECT DESCRIPTION

#### 4.1 System Architecture

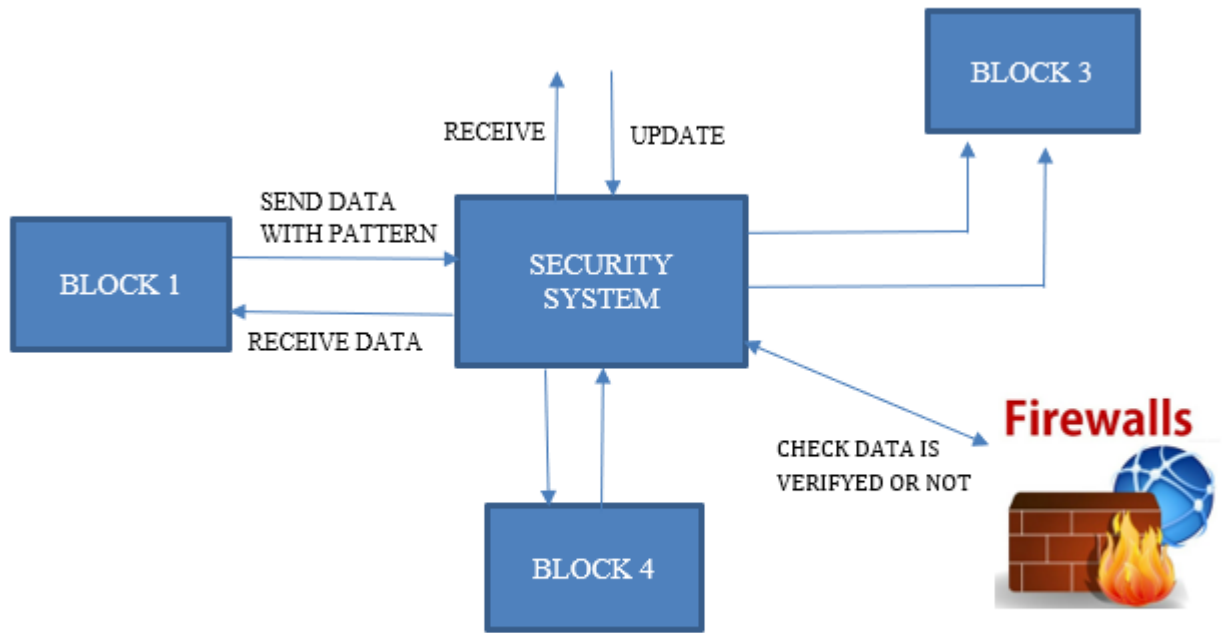


Fig 4.1 System Architecture

The above figure 4.1 shows the overall architecture of our system.

We consider a cloud system composed of three major entities that are cloud server, data owner and the multiple users. Block 1 send the data to security system and then received the data. Security system provide security using firewall. Firewall is used to check data is verified or not.

The data owner has not control over the data after it is uploaded on cloud. In this module, the original data get encrypted into two different values.



## 4.2 Modules

There are Used five Different Modulus

### 4.2.1 Login Module

### 4.2.2 Registration Module

### 4.2.3 Creation Storage and Instance

### 4.2.4 Find collusion Module

### 4.2.5 Find Third-Party Module

#### 4.2.1 Login Module

This is the first activity, User needs to provide a correct contact number and a password, which user enters while registering, in order to login into the app. If information provided by the user matches with the data in the database table, then user successfully login into the app else message of login failed is displayed and user need to re-enter correct information. A link to the register activity is also provided for registration of new users.

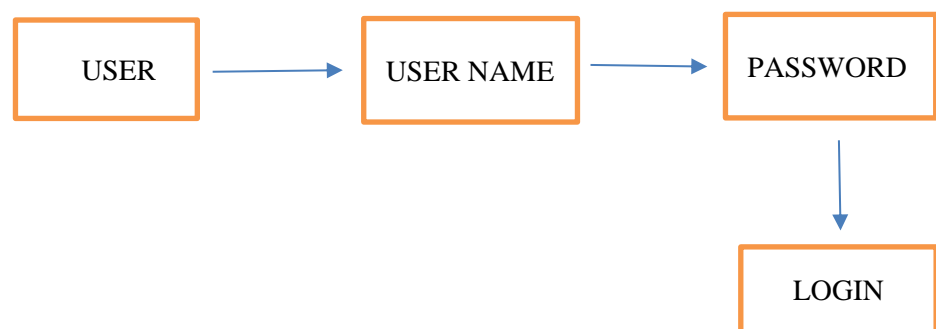


Fig 4.2 Login Module

**INPUT:** User Name and Password

**OUTPUT:** Admin Login

### 4.2.2 Registration Module

A new user who wants to access the app needs to register first before login. By clicking on register button in login activity, the register activity gets open. A new user registers by entering full name, password and contact number. A user needs to enter password again in confirm password textbox for confirmation. When user enters the information in all textboxes, on the click of register button, the data is transferred to database and user is directed to login activity again.

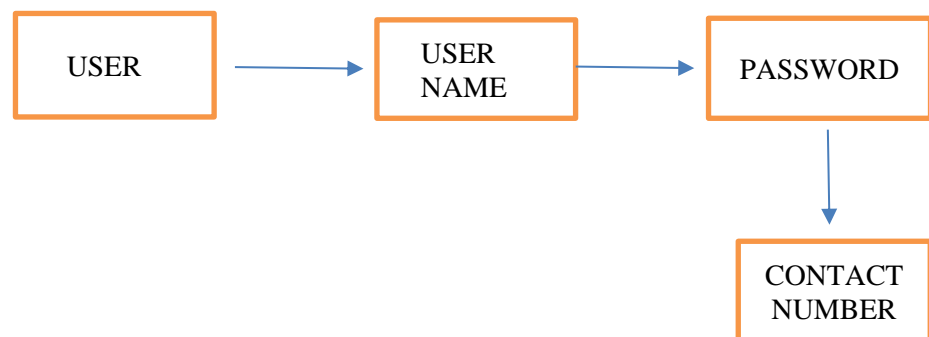


Fig 4.3 Registration Module

**INPUT:** User Name and Password

**OUTPUT:** Database

### 4.2.3 Creation Storage and Instance

The data owner has not control over the data after it is uploaded on cloud. In this module, the original data get encrypted into two different values. The data in each slice can be encrypted by using different cryptographic algorithms and encryption key before storing them in the Cloud.

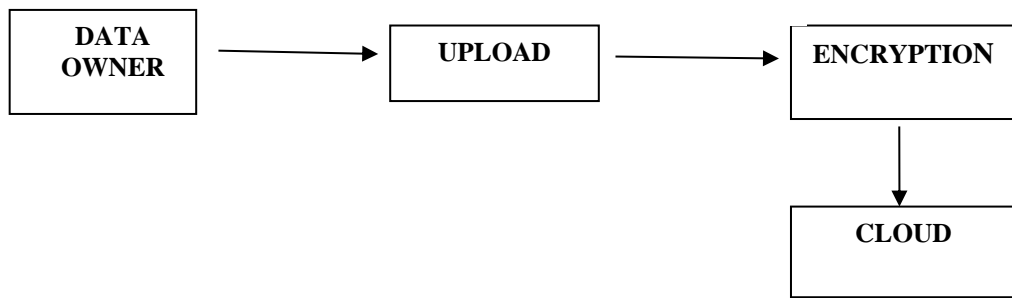


Fig 4.4 Creation Storage and Instance

**INPUT:** User Name and Password

**OUTPUT:** data uploaded

#### 4.2.4 Find Collusion Module

In this Module, Receiver can find collusion occurring or not using calculating a distance.

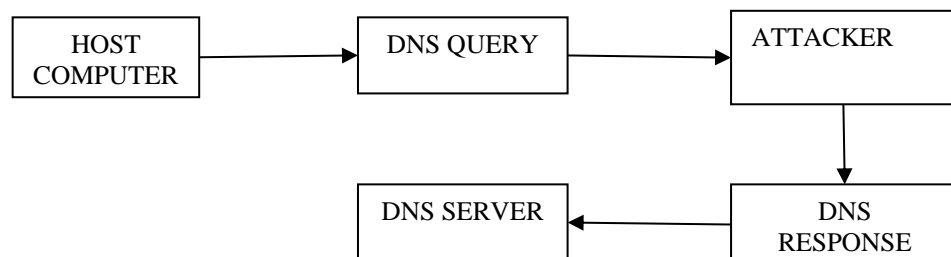


Fig 4.5 Find Collusion Module

**INPUT:** User Name and Password

**OUTPUT:** Database

#### 4.2.5 Find Third-Party Module

In this Module, receiver can also find third-parties. Third party refers to another company making software for the original vendor's product

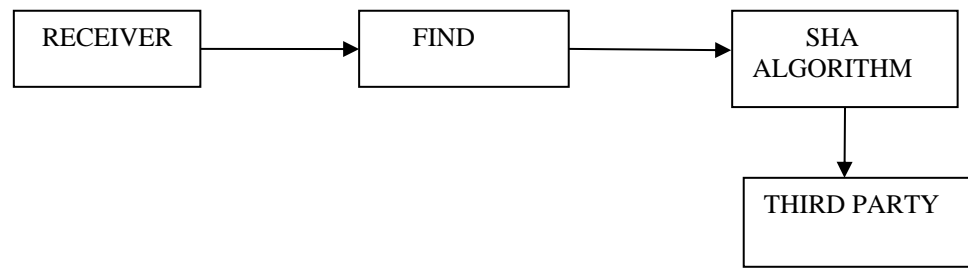


Fig 4.6 Find Third-Party Module

**INPUT:** User Name and Password

**OUTPUT:** find third-parties

### 4.3 Data Flow Diagram

A Data Flow Diagram (DFD) is a graphical representation of the “flow” of data through an information system, modeling its process aspects. Often, they are a preliminary step used to create an overview of the system which can later be elaborated. DFD’s can also be used for the visualization of data processing (structured design).

#### 4.3.1 DFD-Level 0: Data Owner

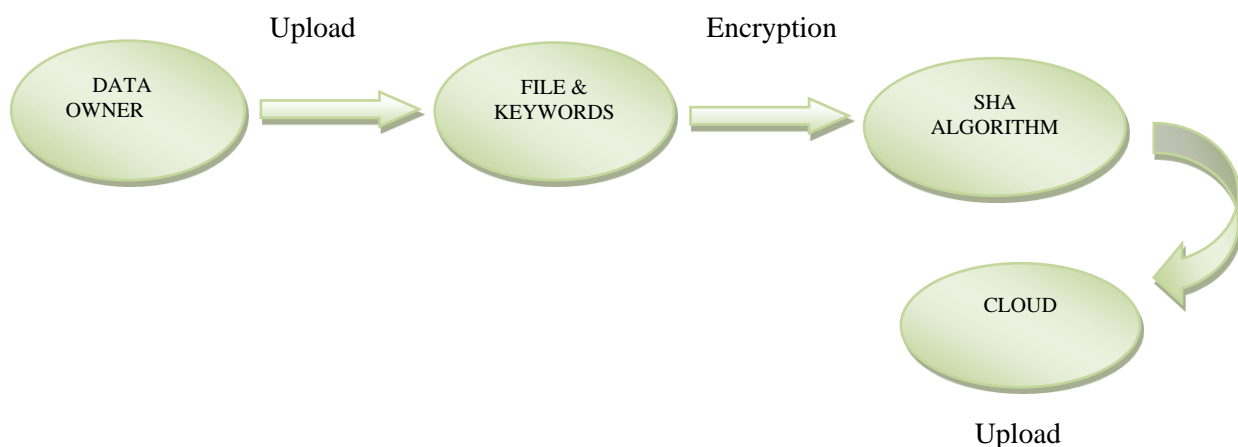


Fig 4.7 DFD-Level 0: Data Owner

#### 4.3.2 DFD-Level 1:

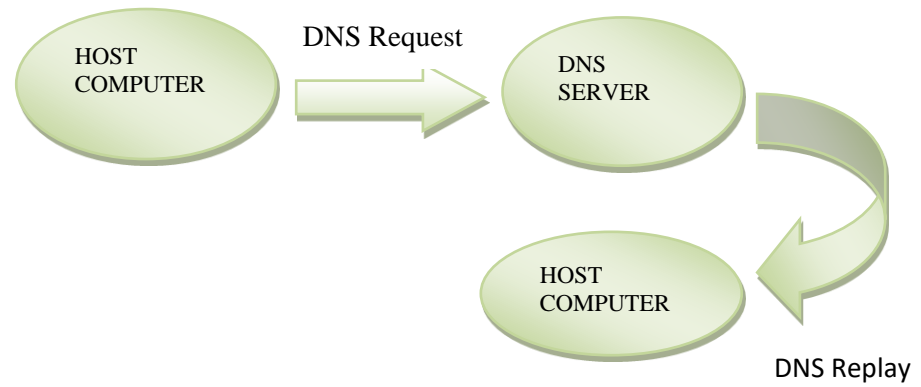


Fig 4.8 DFD-Level 1:

#### 4.3.3 DFD-Level 2:

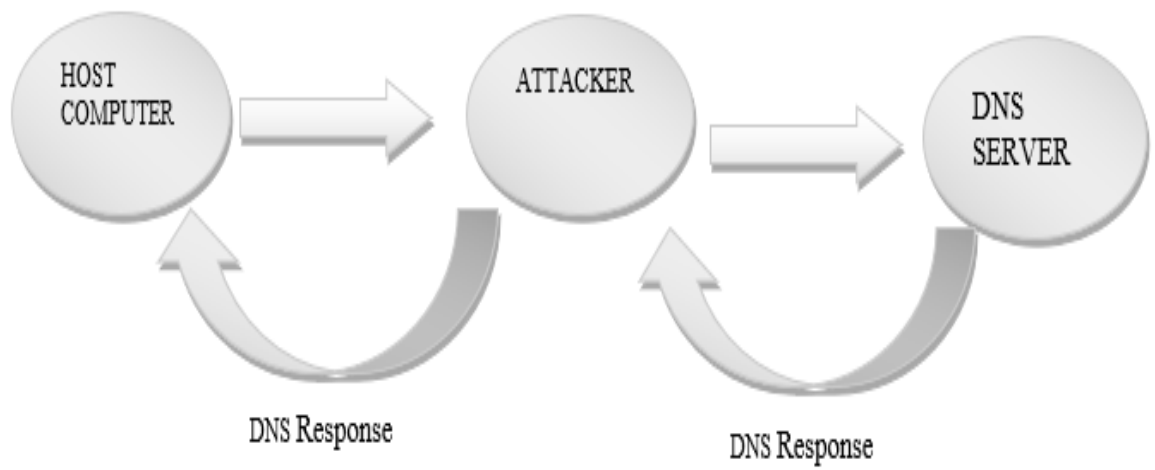


Fig 4.9 DFD-Level 2:

## 4.4 SYSTEM DIAGRAM

### 4.4.1 Use case Diagram

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.

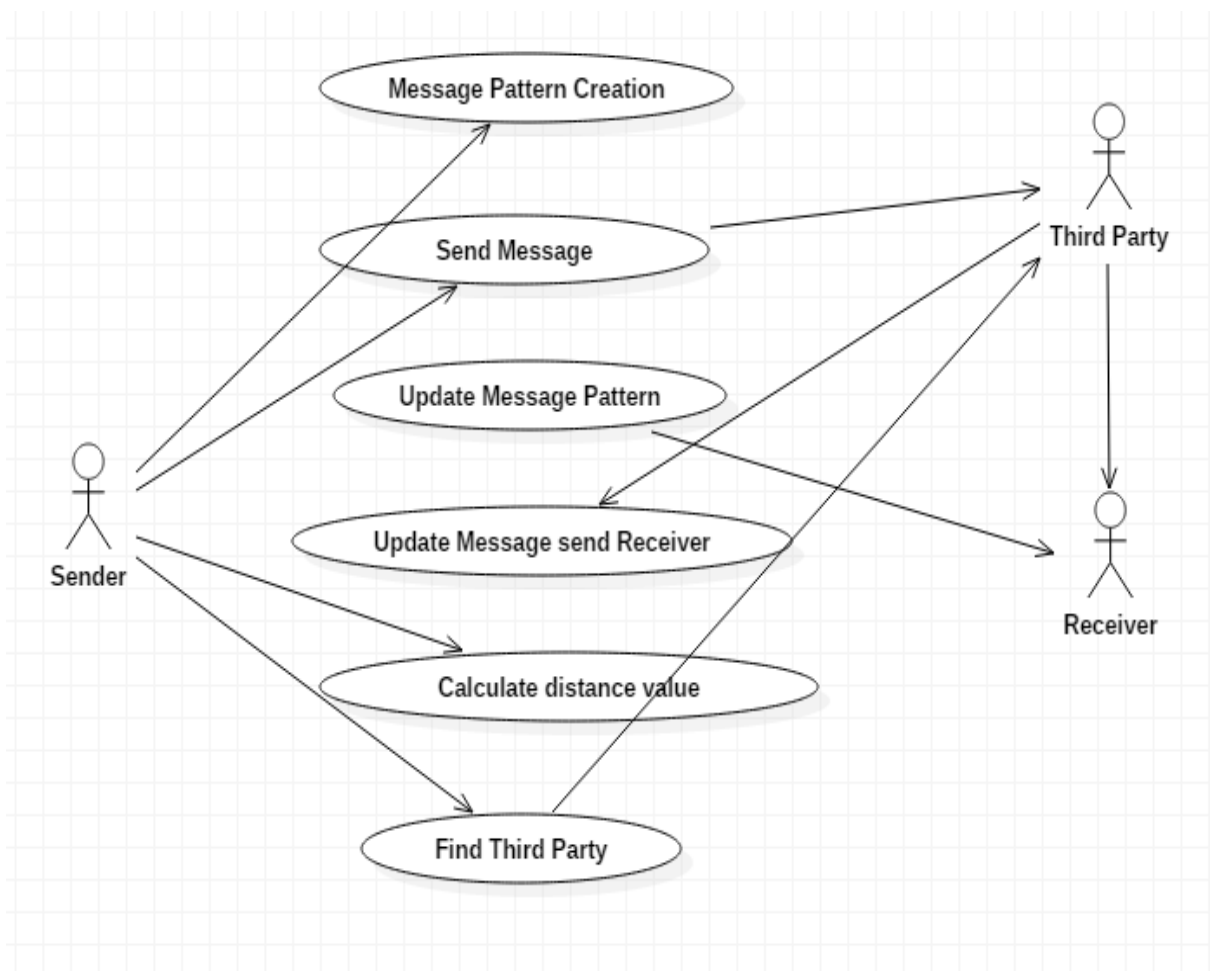


Fig 4.10 Use Case Diagram.

#### 4.4.2 Class Diagram

A class is the main build in block of object-oriented modeling. It is used for general conceptual modeling of the systematic of the application and for detailed modeling translated the models into the program code.

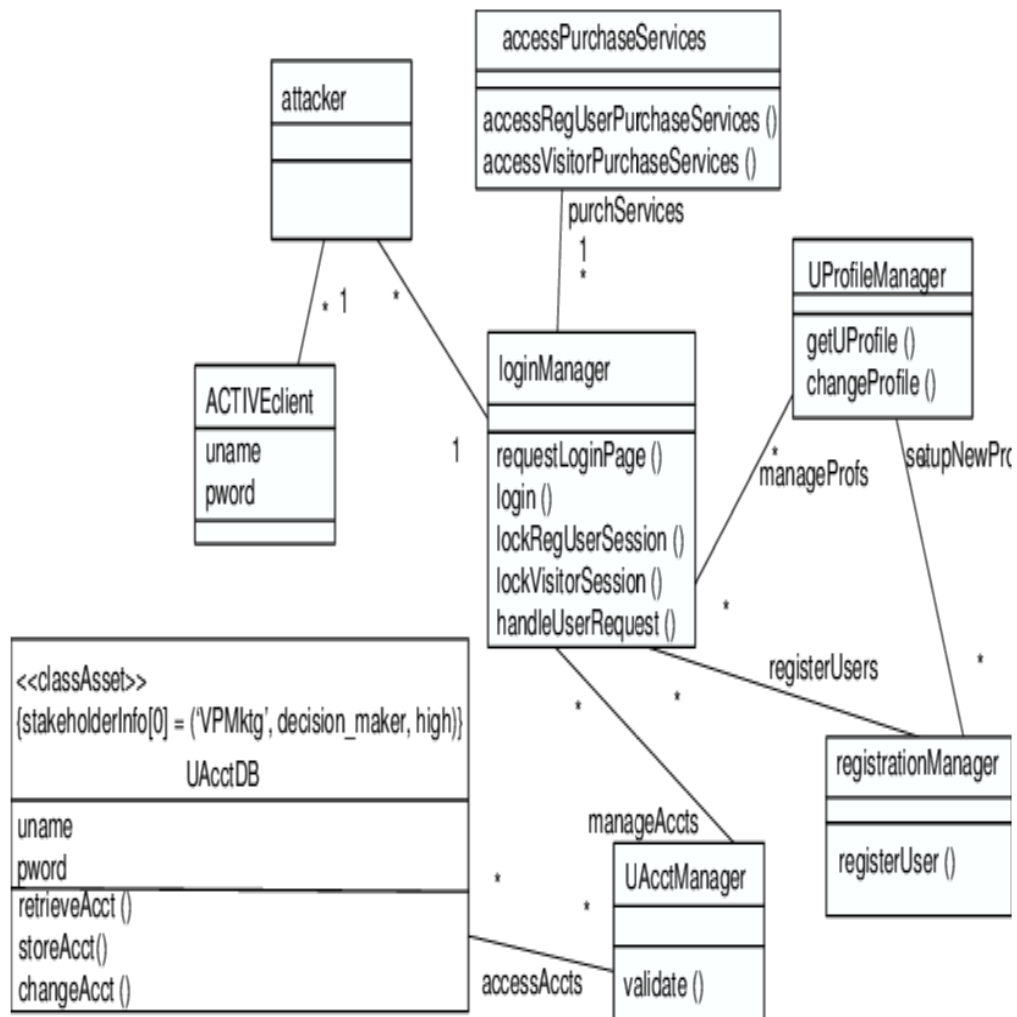


Fig.4.11 Class Diagram

### 4.4.3 Sequence Diagram

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a message Sequence Chart. Sequence diagram are sometime called event diagrams, and timing diagram.

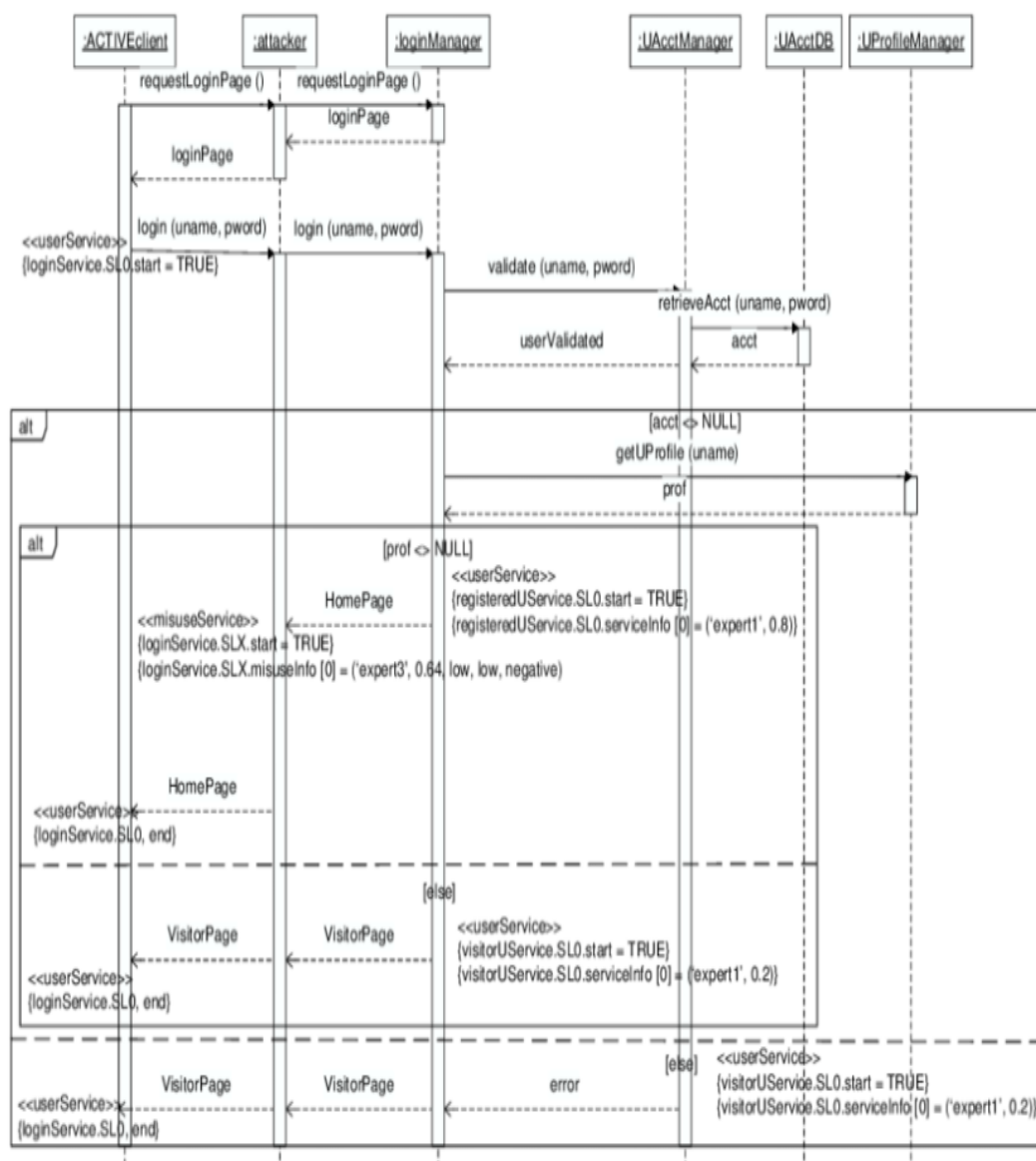


Fig: 4.12 Sequence Diagram



#### 4.4.4 Activity Diagram

Activity diagrams are graphical representations of workflow of stepwise activity and action with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagram can be used to describe the business and operational step by step workflow of components in a system. An activity diagram shows the overflow of control.

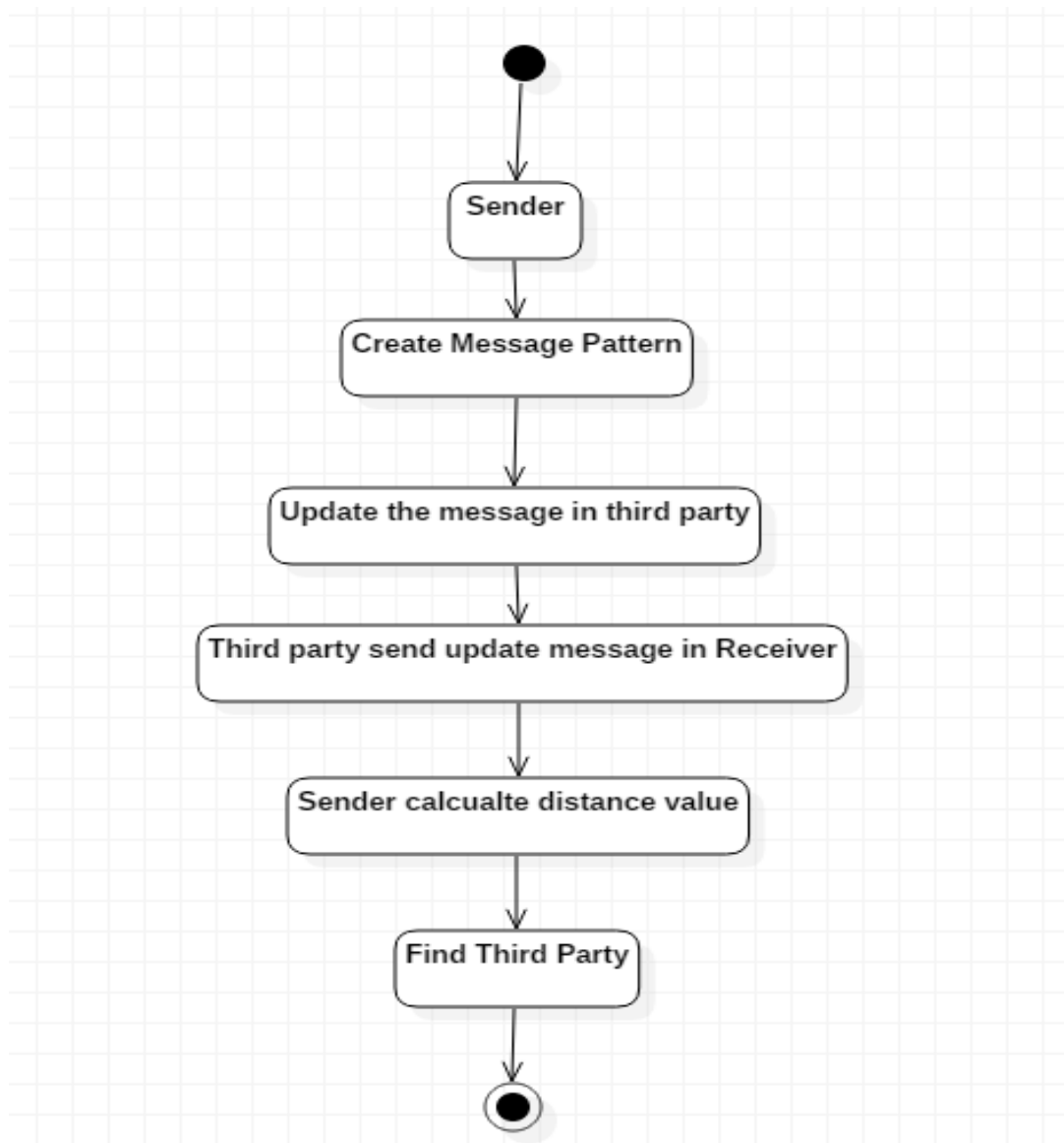


Fig: 4.13 Activity Diagram

#### 4.4.5 Component Diagram

A component diagram is user to visualize organization and relationship among them. The systems are more component useful while making the executable system. The user, master user and Third-party auditor are executable parts of the system.

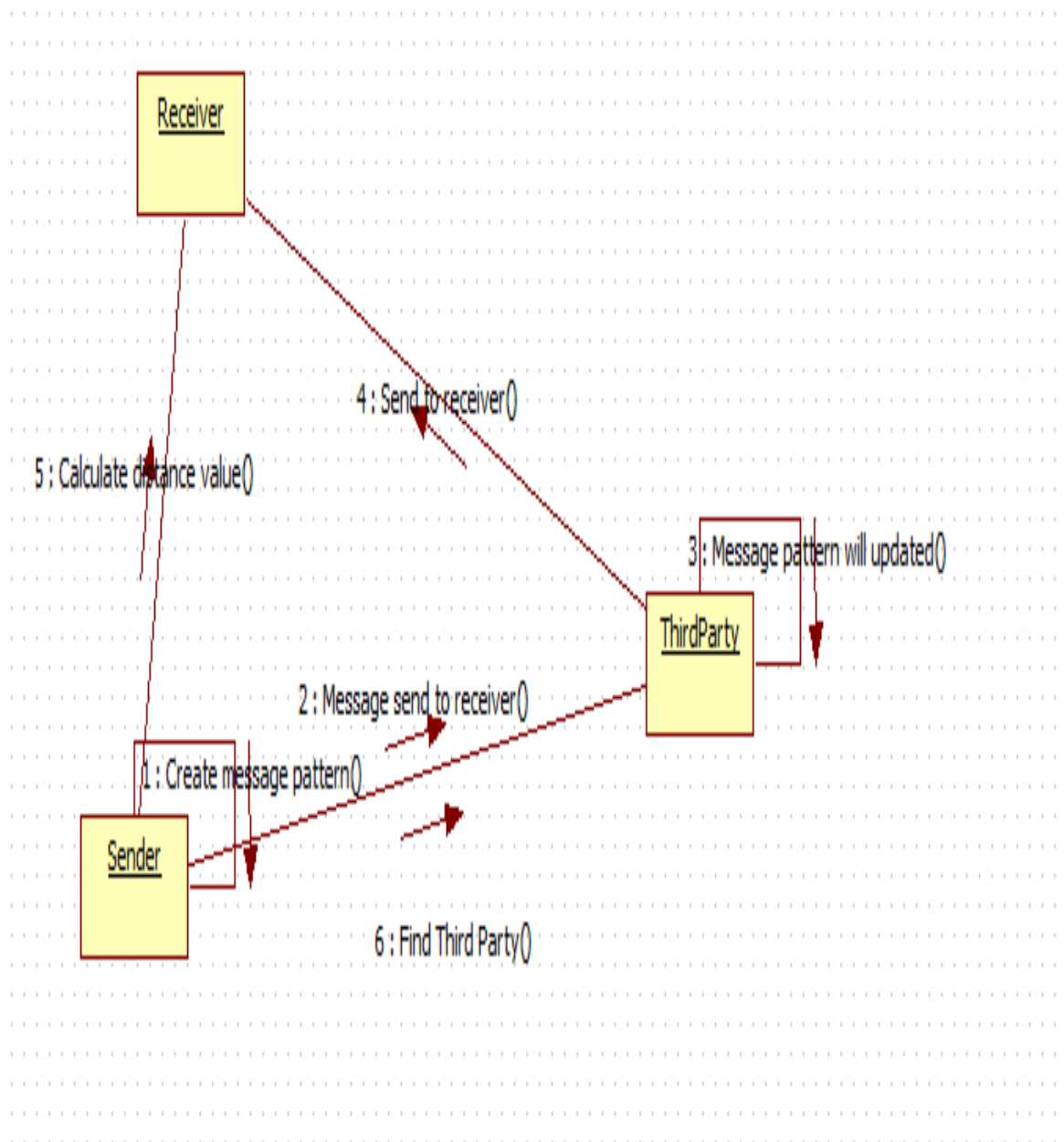


Fig: 4.14 Component Diagram

## 4.5 Requirement Specification

### 4.5.1 Hardware Requirements

System	Pentium IV
Hard Disk	40 GB
Speed	2.4GHZ
Monitor	15 VGA colour
RAM	512 MB

Table 2: Hardware Requirements

### 4.5.2 Software Requirements

Operating system	Windows XP
Coding Language	JAVA
IDE	Net beans
Data Base	MYSQL

Table 3: Software Requirements

## **CHAPTER 5**

### **5. SYSTEM DESIGN**

#### **5.1 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.

#### **5.2 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.

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 6**

### **6. System Testing**

#### **6.1 Testing Process**

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.

#### **6.2. Types of Tests**

1. White Box Testing
2. Black Box Testing
3. Unit Testing
4. Integration Testing
5. Alpha Testing
6. Beta Testing

##### **6.2.1 Unit Testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program input produces 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 business process performs accurately

to the documented specifications and contains clearly defined inputs and expected results.

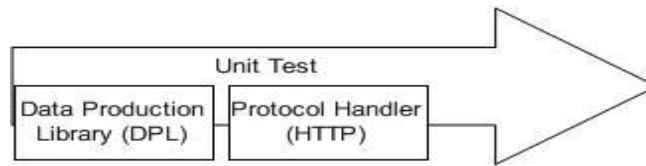


Fig 6.3 Unit Testing

### 6.2.2 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 satisfactory, as shown by successful 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.

### 6.2.3 Functional Testing

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

**Functional testing is centered on the following items:**

- **Valid Input** is used to identify classes of valid input that must be accepted.
- **Invalid Input** is used to identify classes of invalid input that must be rejected.
- **Functions** is used to identify functions that must be exercised.
- **Output** is used to identify classes of application outputs.
- **Systems/Procedures** is used to identify systems or procedures that 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 identifying 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.

#### **6.2.4 System Testing**

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.

#### **6.2.5 White Box Testing**

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

#### **6.2.6 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.

#### **Test Strategy and Approach**

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



## **Test Objectives**

- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.
- 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.

### **6.2.7 Integration Testing**

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.

### **6.2.8 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.

### **6.2.9 Alpha Testing**

In software development, alpha test will be a test among the teams to confirm that your product works. Originally, the term alpha test meant the first phase of testing in a software development process. The first phase includes unit testing, component testing, and system testing. It also enables us to test the product on the lowest common denominator machines to make sure download times are acceptable and pre loaders work.

### **6.2.10 Beta Testing**

In software development, a beta test is the second phase of software testing in which a sampling of the intended audience tries the product out. Beta testing can be considered "pre-release testing." Beta test versions of software are now distributed to curriculum specialists and teachers to give the program a "real world" test.

## **CHAPTER 7**

### **CONCLUSION & FUTURE ENHANCEMENT**

#### **7.1 Conclusion**

Through the above summary, due to the problems about the collusion attacks that are widespread in the secure outsourcing of sequence comparison algorithms, this paper will introduce the trusted authority to authenticate user those who have the access to the data on cloud. SHA algorithm is used by the trusted authority to generate the key and that key will get share to user as well as the owner. The trusted authority module receives encrypted file using AES Algorithm from the data owner and computes hash value using MD-5 algorithm. It stores key in its database which will be used during the dynamic operations and to determine the cheating party in the system. Trusted authority send file to CSP module to store on cloud. The resulting key sets are shown to have a number of desirable properties that ensure the confidentiality of communication sessions against collusion attacks by other network nodes.

#### **7.2 Future Enhancement**

It is somewhat hard to extend the work in our paper to certain applications with multi-data source. Firstly, two-character sequences from different sources should be encrypted respectively with different keys. Secondly, three cost matrices should be encrypted together after being constructed by the negotiation between both sides. The security target is to complete sequence comparison on a single cloud server in the way of privacy preservation and to ensure that the string typed data of the end user on any side will not be arbitrarily stolen by the other user or the CSP.

## REFERENCES

- [1] Y. Feng, H. Ma, and X. Chen, “Efficient and verifiable out sourcing scheme of sequence comparisons,” *Intel. Autom. Soft Comput.*, vol. 21, no. 1, pp. 51–63, Jan. 2015.
- [2] M. J. Atallah and J. Li, “Secure outsourcing of sequence comparisons,” in *Proc. Int. Workshop Privacy Enhancing Technol. (PET)*, Toronto, ON, Canada, 2004, pp. 63–78.
- [3] M. J. Atallah, F. Kerschbaum, and W. Du, “Secure and private sequence comparisons,” in *Proc. ACM Workshop Privacy Electron. Soc. (WPES)*, Washington, DC, USA, 2003, pp. 39–44.
- [4] D. Szajda, M. Pohl, J. Owen, and B. Lawson, “Toward a practical data privacy scheme for a distributed implementation of the Smith-Waterman genome sequence comparison algorithm,” in *Proc. Netw. Distrib. Syst. Secur. Symp. (NDSS)*, San Diego, CA, USA, 2006, pp. 253–265.
- [5] X. Chen, J. Li, J. Ma, Q. Tang, and W. Lou, “New algorithms for secure outsourcing of modular exponentiations,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 9, pp. 2386–2396, Sep. 2014.
- [6] R. Akimana, O. Markowitch, and Y. Roggeman, “Secure outsourcing of DNA sequences comparisons in a Grid environment,” *WSEAS Trans. Comput. Res.*, vol. 2, no. 2, pp. 262–269, Feb. 2007.
- [7] M. Blanton, M. J. Atallah, K. B. Frikken, and Q. Malluhi, “Secure and efficient outsourcing of sequence comparisons,” in *Proc. Eur. Symp. Res. Comput. Secure. (ESORICS)*, Pisa, Italy, 2012, pp. 505–522.
- [8] Y. Feng, H. Ma, X. Chen, and H. Zhu, “Secure and verifiable outsourcing of sequence comparisons,” in *Proc. Int. Conf. Inf. Commun. Technol. (ICT-EurAsia)*, Yogyakarta, Indonesia, 2013, pp. 243–252.
- [9] S. Salinas, X. Chen, J. Li, and P. Li, “A tutorial on secure outsourcing of large-scale computations for big data,” *IEEE Access*, vol. 4, pp. 1406–1416, Apr. 2016.
- [10] X. Chen, J. Li, J. Weng, J. Ma, and W. Lou, “Verifiable computation over large database with incremental updates,” *IEEE Trans. Comput.*, vol. 65, no. 10, pp. 3184–

3195, Oct. 2016.

[11] B. Berger, N. M. Daniels, and Y. W. Yu, “Computational biology in the 21st century: Scaling with compressive algorithms,” *Commun. ACM*, vol. 59, no. 8, pp. 72–80, Aug. 2016.

[12] E. Ayday, J. L. Raisaro, J.-P. Hubaux, and J. Rougemont, “Protecting and evaluating genomic privacy in medical tests and personalized medicine,” in *Proc. ACM Workshop Privacy Electron. Soc. (WPES)*, Berlin, Germany, 2013, pp. 95–106.

[13] E. D. Cristofaro, S. Faber, and G. Tsudik, “Secure genomic testing with size-and position-hiding private substring matching,” in *Proc. ACM Workshop Privacy Electron. Soc. (WPES)*, Berlin, Germany, 2013, pp. 107–117.

[14] J. H. Cheon, M. Kim, and K. Lauter, “Homomorphic computation of edit distance,” in *Proc. Int. Conf. Financial Cryptogr. Data Secur. (FC)*, Puerto Rico, 2015, pp. 194–212.

[15] K. Hua, Q. Yu, and R. Zhang, “A guaranteed similarity metric learning framework for biological sequence comparison,” *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 13, no. 5, pp. 868–877, Sep. 2016.

[16] N. S. Vo, Q. Tran, N. Niraula, and V. Phan, “RandAL: A randomized approach to aligning DNA sequences to reference genomes,” *BMC Genomics*, vol. 15, p. S2, Jul. 2014.

[17] V. Palazón-González and A. Marzal, “Speeding up the cyclic edit distance using LAESA with early abandon,” *Pattern Recognit. Lett.*, vol. 62, pp. 1–7, Sep. 2015.

[18] K. Lauter, A. López-Alt, and M. Naehrig, “Private computation on encrypted genomic data,” in *Proc. Int. Conf. Cryptol. Inf. Secur. Latin Amer. (LATINCRYPT)*, Florianópolis, Brazil, 2015, pp. 3–27.

[19] K. Shimizu, K. Nuida, and G. Rätsch, “Efficient privacy-preserving string search and an application in genomics,” *Bioinformatics*, vol. 32, no. 11, pp. 1652–1661, Jun. 2016.

[20] J. Katz and Y. Lindell, *Introduction to Modern Cryptography*. Boca Raton, FL, USA: CRC Press, 2014, pp. 241–284.

## APPENDICES

### APPENDIX 1

#### SOURCE CODE

##### Coding

##### AES Encrypter

```
import java.io. *;
import java.security.Key;
import javax.crypto.Cipher;
import javax.crypto.SecretKey;
import javax.crypto.spec.IvParameterSpec;
import javax.crypto.CipherInputStream;
import javax.crypto.CipherOutputStream;
import javax.crypto.KeyGenerator;
import java.security.spec.AlgorithmParameterSpec;
import javax.crypto.spec.SecretKeySpec;
public class AESEncrypter
{
    Cipher ecipher;
    Cipher dcipher;
    public AESEncrypter(SecretKey key)
    {
        byte[] iv = new byte[]
        {0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x09, 0x0a, 0x0b, 0x0c,
        0x0d, 0x0e, 0x0f};
        AlgorithmParameterSpec paramSpec = new IvParameterSpec(iv);
        try{
            ecipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
            dcipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
            ecipher.init(Cipher.ENCRYPT_MODE, key, paramSpec);
            dcipher.init(Cipher.DECRYPT_MODE, key, paramSpec);
        }
        catch (Exception e){
            e.printStackTrace();
        }
    }
    byte[] buf = new byte[1024];
    public void encrypt(InputStream in, OutputStream out)
    {try{
        out = new CipherOutputStream(out, ecipher);
        int numRead = 0;
        while ((numRead = in.read(buf)) >= 0)
        {
            out.write(buf, 0, numRead);
        }
        out.close();
    }
    catch (java.io.IOException e)
    {
```

```

}}
public void decrypt(InputStream in, OutputStream out)
{try{
in = new CipherInputStream(in, dcipher);
int numRead = 0;
while ((numRead = in.read(buf)) >= 0)
{
out.write(buf, 0, numRead);
}out.close();
}catch (java.io.IOException e)
{
}}
private static Key generateKey() {
String keyValue="TheBestSecretKey";
Key key = new SecretKeySpec(keyValue.getBytes(), "AES");
return key;
}
public static void main(String args[])
{try{
Key key1 = generateKey();
/*KeyGenerator      kgen=KeyGenerator.getInstance("AES");
kgen.init(128);
SecretKey key=kgen.generateKey();
System.out.println("*/
AESEncrypter encrypter = new AESEncrypter((SecretKey)key1);
encrypter.encrypt(new FileInputStream("D:\\Documents and Settings\\JAVA-
SAN\\My Documents\\My Pictures\\2.JPG"),new
FileOutputStream("C:/Encrypted.jpg"));
FileOutputStream("C:/Decrypted.jpg"));;
}
catch (Exception e)
{
e.printStackTrace();
}}}}

```

## **Admin Servlet**

```

import com.commondb.Common_DB;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.ResultSet;
import java.util.ArrayList;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
/**
*
* @author sentamilpandi.m

```

```

*/
@WebServlet(name = "AdminServlet", urlPatterns = {"/AdminServlet"})
public class AdminServlet extends HttpServlet {
/**
 * Processes requests for both HTTP
 * <code>GET</code> and
 * <code>POST</code> methods.*
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
throws ServletException, IOException {
response.setContentType("text/html;charset=UTF-8");
PrintWriter out = response.getWriter();
try {
/*
 * TODO output your page here. You may use following sample code.
 */
out.println("<html>");
out.println("<head>");
out.println("<title>Servlet AdminServlet</title>");
out.println("</head>");
out.println("<body>");
out.println("<h1>Servlet AdminServlet at " + request.getContextPath() + "</h1>");
out.println("</body>");
out.println("</html>");
} finally {
out.close();
}}
/**
 * Handles the HTTP
 * <code>GET</code> method.
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
processRequest(request, response);
}
/**
 * Handles the HTTP
 * <code>POST</code> method.
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs

```



```

*/
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
try {
String adminuser=request.getParameter("adminuser");
String adminpass=request.getParameter("adminpass");
ResultSet rs=Common_DB.LoginCheck("mona","adminlogin", "Username",
"password", adminuser, adminpass);
if(rs.next()) {
ArrayList<String> list=new ArrayList<String>();
HttpSession ses1=request.getSession(true);
System.out.println("????????");
ResultSet rr=Common_DB.ViewTable("mona","adminlogin");
while(rr.next())
{
String n=rr.getString(1);
System.out.println("????????"+n);
list.add(n);
}
ses1.setAttribute("groupname", list);
response.sendRedirect("AdminLinks.jsp");
}
else {
response.sendRedirect("Error.jsp");
}} catch (Exception ex) {
Logger.getLogger(AdminServlet.class.getName()).log(Level.SEVERE, null, ex);
}}
/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
@Override
public String getServletInfo() {
return "Short description";
}}

```

### Approved Servlet

```

import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import java.sql.*;
/**
 *
 * @author sentamilpandi.m
 */
public class ApprovedServlet extends HttpServlet {

```

```

/**
 * Processes requests for both HTTP
 * <code>GET</code> and
 * <code>POST</code> methods.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
throws ServletException, IOException {
response.setContentType("text/html;charset=UTF-8");
PrintWriter out = response.getWriter();
try {
/*
 * TODO output your page here. You may use following sample code.
 */
out.println("<html>");
out.println("<head>");
out.println("<title>Servlet ApprovedServlet</title>");
out.println("</head>");
out.println("<body>");
out.println("<h1>Servlet ApprovedServlet at " + request.getContextPath() + "</h1>");
out.println("</body>");
out.println("</html>");
} finally {
out.close();
}}
/**
 * Handles the HTTP
 * <code>GET</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
processRequest(request, response);
}
/**
 * Handles the HTTP
 * <code>POST</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs

```

```

*/
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    HttpSession session=request.getSession();
    String UserName=session.getAttribute("username").toString();
    String group=session.getAttribute("group").toString();
    String uname = request.getParameter("slist1");
    String filename = request.getParameter("slist3");
    System.out.println("&&& "+filename+"&&&&& "+uname);
    String Approved="Approved";
    int counting=0;
    Connection con=null;
    Statement st=null;
    ResultSet rs=null;
    try{
        Class.forName("com.mysql.jdbc.Driver");
        con=DriverManager.getConnection("jdbc:mysql://localhost:3306/mona","root","root")
        ;
        st=con.createStatement();
        String qry = "select * from othergroup where username='"+uname+"' &&
        groupname='"+group+"' && filename='"+filename+"'";
        rs=st.executeQuery(qry);
        rs=st.executeQuery("select usercount from othergroup where groupname='"+group+"'
        and username='"+UserName+"' ");
        if(rs.next())
        {
            counting=Integer.parseInt(rs.getString("usercount"));
        }
        System.out.println("@ @ @ @ @"+counting);
        intrs1=st.executeUpdate("updateothergroupSET
        usercount='"+(counting+1)+"',status='"+Approved+"'whereusername='"+uname+"'&&
        filename='"+filename+"'");
        if(rs1>0)
        {
            response.sendRedirect("AppSuccess.jsp");
        }else{
            response.sendRedirect("AppFail.jsp");
        }
    }
    catch(Exception ex)
    {
        ex.printStackTrace();
    }
}
/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
@Override
public String getServletInfo() {
    return "Short description";
}
}

```

## Delete Servlet

```
import com.commondb.Common_DB;
import static com.mona.DownloadServlet.filename;
import java.io.*;
import java.io.IOException;
import java.io.PrintWriter;
import java.security.Key;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.Properties;
import java.util.Random;
import java.util.Scanner;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;
import javax.mail.Message;
import javax.mail.PasswordAuthentication;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;
import javax.servlet.RequestDispatcher;
import javax.servlet.http.HttpSession;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
/**
 *
 * @author sentamilpandi.m
 */
public class DeleteServlet extends HttpServlet {
    Connection con=null;
    Statement st=null;
    ResultSet rs=null;
    RequestDispatcher rd=null;
    static Properties properties=new Properties();
    static
    {
        properties.put("mail.smtp.host", "smtp.gmail.com");
        properties.put("mail.smtp.socketFactory.Fort", "465");
        properties.put("mail.smtp.socketFactory.class", "javax.net.ssl.SSLSocketFactory");
        properties.put("mail.smtp.auth", "true");
        properties.put("mail.smtp.Fort", "465");
    }
    private static Key generateKey(String keyvalidation) {
        String keyValue=keyvalidation;
        Key key = new SecretKeySpec(keyValue.getBytes(), "AES");
        return key;
    }
}
```

```

}
/**
 * Processes requests for both HTTP
 * <code>GET</code> and
 * <code>POST</code> methods.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
throws ServletException, IOException {
response.setContentType("text/html;charset=UTF-8");
PrintWriter out = response.getWriter();
try {
/*
 * TODO output your page here. You may use following sample code.
 */
out.println("<html>");
out.println("<head>");
out.println("<title>Servlet DeleteServlet</title>");
out.println("</head>");
out.println("<body>");
out.println("<h1>Servlet DeleteServlet at " + request.getContextPath() + "</h1>");
out.println("</body>");
out.println("</html>");
} finally {
out.close();
}}
// <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the +
sign on the left to edit the code.">
/**
 * Handles the HTTP
 * <code>GET</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
processRequest(request, response);
}
/**
 * Handles the HTTP
 * <code>POST</code> method.
 *
 * @param request servlet request

```

```

* @param response servlet response
* @throws ServletException if a servlet-specific error occurs
* @throws IOException if an I/O error occurs
*/
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
try
{
HttpSession session1=request.getSession();
String UserName=session1.getAttribute("username").toString();
String list=session1.getAttribute("filename").toString();
String group=session1.getAttribute("group").toString();
String fname=request.getParameter("filename");
String TempDownloadDirectory="D:/temp1/";
String ln="";
String key = null;
try{
Class.forName("com.mysql.jdbc.Driver");
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/mona","root","password");
st=con.createStatement();
rs=st.executeQuery("Select product from groupname where
groupname='"+group+"'");
if(rs.next())
{
key=rs.getString(1);
}}
catch(Exception ex)
{
ex.printStackTrace();
}
Key key1 = generateKey(key);
"+TempDownloadDirectory+filename+"????? "+group);
AesEncrypter Decrypter = new AesEncrypter((SecretKey)key1);
Decrypter.decrypt(new FileInputStream("D:"+group+"/"+fname),new
FileOutputStream(TempDownloadDirectory+fname));
System.out.println("D:"+group+"/"+fname);
File ff=new File(TempDownloadDirectory+fname);
Scanner sc=new Scanner(ff);
while(sc.hasNextLine())
{
ln=ln+sc.nextLine().toString();
}
session1.setAttribute("data", ln);
final String from="javaredquene@gmail.com";
final String password="mona1234";
final String to="javaredquene@gmail.com";
Session session = Session.getInstance(properties, new javax.mail.Authenticator(){
protected PasswordAuthentication getPasswordAuthentication() {
return new PasswordAuthentication(from, password);
}});
}

```

```

Random generator = new Random();
int r = generator.nextInt(999999);
String ran12=new Integer(r).toString();
Message message = new MimeMessage(session);
message.setFrom(new InternetAddress(from));
message.setRecipients(Message.RecipientType.TO,
InternetAddress.parse(to));
message.setSubject("OTP FOR YOU");
message.setText("otp for this session is :"+ran12);
Transport.send(message);
System.out.println("Email Sent Successfully");
response.sendRedirect("Delete.jsp");
}catch(Exception ex){
ex.printStackTrace();
}}
/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
@Override
public String getServletInfo() {
return "Short description";
}}

```

## Download Servlet

```

import com.commondb.Common_DB;
import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;
import java.io.*;
import javax.servlet.ServletOutputStream;
import java.security.Key;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.commons.fileupload.FileItem;
import org.apache.commons.fileupload.FileUploadException;
import org.apache.commons.fileupload.disk.DiskFileItemFactory;
import org.apache.commons.fileupload.servlet.ServletFileUpload;
/**
 *
 * @author sentamilpandi.m

```

```

*/
@WebServlet(name = "DownloadServlet", urlPatterns = { "/DownloadServlet" })
public class DownloadServlet extends HttpServlet {
    static String filename;
    private static Key generateKey(String keyvalidation) {
        String keyvalue=keyvalidation;
        Key key = new SecretKeySpec(keyvalue.getBytes(), "AES");
        return key;}
    private static final String dCase = "abcdefghijklmnopqrstuvwxyz";
    private static final String uCase = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
    private static final String sChar = "!@#$%^&*";
    private static final String intChar = "0123456789";
    /**
     * Processes requests for both HTTP
     * <code>GET</code> and
     * <code>POST</code> methods.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request, HttpServletResponse
    response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            /*
             * TODO output your page here. You may use following sample code.
             */
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet DownloadServlet</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Servlet DownloadServlet at " + request.getContextPath() + "</h1>");
            out.println("</body>");
            out.println("</html>");
        } finally {
            out.close();
        }
    }
    /**
     * Handles the HTTP
     * <code>GET</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override

```



```

protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
processRequest(request, response);
}
/**
 * Handles the HTTP
 * <code>POST</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
try {
HttpSession session1=request.getSession(true);
String keyvalidation=request.getParameter("keyvalidation");
String group=session1.getAttribute("group").toString();
String fn = request.getParameter("slist3");
filename=request.getParameter("filename");
String TempDownloadDirectory="D:/temp1/";
File file=new File(TempDownloadDirectory);
if(!(file.exists())) {
file.mkdir();
}
Key key1 = generateKey(keyvalidation);
System.out.println(keyvalidation+"#####"+TempDownloadDirectory+filename+
"????? "+group);
AESEncrypter Decrypter = new AESEncrypter((SecretKey)key1);
Decrypter.decrypt(new FileInputStream("D:"+group+"/"+filename),new
FileOutputStream(TempDownloadDirectory+filename));
if(ServletFileUpload.isMultipartContent(request)){
try {
String nn="";
List<FileItem> multipart = new ServletFileUpload(
new DiskFileItemFactory()).parseRequest(request);
for(FileItem item : multipart){
if(!item.isFormField()){
String name = new File(item.getName()).getName();
item.write( new File(TempDownloadDirectory+name));
nn=name;
}}
request.setAttribute("message", "File download Successfully");
System.out.println("???????");
} catch (Exception ex) {
request.setAttribute("message", "File download Failed due to " + ex);
}}
Key key1 = generateKey(keyvalidation);
System.out.println("#####"+TempDownloadDirectory+filename);
AESEncrypter Decrypter = new AESEncrypter((SecretKey)key1);

```

```

Decryper.decrypt(new FileInputStream("D:"+group+"/"+filename),new
FileOutputStream(TempDownloadDirectory+filename));
String filepath1=TempDownloadDirectory+filename;
System.out.println(filepath1+"???????" +filename+"....."+TempDownloadDirectory);
System.out.println("***** "+fn);
FileInputStream filetoDownload=new FileInputStream(filepath1);
ServletOutputStream output=response.getOutputStream();
response.setContentType("images/jpg");
response.addHeader("Content-Disposition","attachment;filename="+filename);
response.setContentLength(filetoDownload.available());
int readBytes=0;
byte[] buffer=new byte[1024];
while(filetoDownload.available()>0)
{
output.write(filetoDownload.read());
}
session1.setAttribute("keyvalidation1", keyvalidation);
output.close();
filetoDownload.close();
} catch (Exception ex) {
ex.printStackTrace();
}}
/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
@Override
public String getServletInfo() {
return "Short description";
}}

```

## Download the Servlet

```

import com.commondb.Common_DB;
import java.io.*;
import java.sql.*;
import javax.servlet.http.HttpSession;
import java.security.Key;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;
import javax.servlet.ServletException;
import javax.servlet.ServletOutputStream;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
/**
 *
 * @author sentamilpandi.m
 */
public class DownloadotherServlet extends HttpServlet {
private static Key generateKey(String keyvalidation) {
String keyValue=keyvalidation;

```

```

Key key = new SecretKeySpec(keyValue.getBytes(), "AES");
return key;
}
private static final String dCase = "abcdefghijklmnopqrstuvwxyz";
private static final String uCase = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
private static final String sChar = "!@#$%^&*";
private static final String intChar = "0123456789";
/**
 * Processes requests for both HTTP
 * <code>GET</code> and
 * <code>POST</code> methods.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
throws ServletException, IOException {
response.setContentType("text/html;charset=UTF-8");
PrintWriter out = response.getWriter();
try {
/*
 * TODO output your page here. You may use following sample code.
 */
out.println("<html>");
out.println("<head>");
out.println("<title>Servlet DownloadotherServlet</title>");
out.println("</head>");
out.println("<body>");
out.println("<h1>Servlet DownloadotherServlet at " + request.getContextPath() +
"</h1>");
out.println("</body>");
out.println("</html>");
} finally {
out.close();
}
/**
 * Handles the HTTP
 * <code>GET</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
processRequest(request, response);
}

```

```

/**
 * Handles the HTTP
 * <code>POST</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
try{
Connection con=null;
Statement st=null;
ResultSet rs=null;
HttpSession session=request.getSession(true);
String UserName=session.getAttribute("username").toString();
String group=session.getAttribute("group").toString();
String Requestgroup=session.getAttribute("Requestgroup").toString();
String filename=request.getParameter("filename");
System.out.println("*****"+filename);
String TempDownloadotherDirectory="D:/temp2/";
System.out.println("D:/temp2/");
File file=new File(TempDownloadotherDirectory);
if(!(file.exists()))
{file.mkdir();
}
String filename1=filename;
filename1=filename.substring(0,filename.indexOf("("));
System.out.println("uuuuuuuuu"+filename1);
String filename2=filename;
String keyvalidation = session.getAttribute("keyvalidation").toString();
System.out.println("MMMMMMMMM"+keyvalidation);
String RequestGroup=filename.substring(filename.indexOf("(")+1,(filename.length()-1));
System.out.println("ZZZZZZZZZZ"+RequestGroup);
System.out.println("IIIIIIII"+filename1+filename2);
Key key1 = generateKey(keyvalidation);
System.out.println("#####"+TempDownloadotherDirectory+filename1);
AESEncrypter Decrypter = new AESEncrypter((SecretKey)key1);
Decrypter.decrypt(new FileInputStream("D:/"+RequestGroup+"/"+filename1),new
FileOutputStream(TempDownloadotherDirectory+filename1));
Class.forName("com.mysql.jdbc.Driver");
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/mona","root","pass
word");
st=con.createStatement();
Stringqry="selectRequestgroupfromothergroupwhereusername='"+UserName+"'
Requestgroup='"+Requestgroup+"'";
rs=st.executeQuery(qry);
String reqname="";
while(rs.next()){

```

```

reqname=rs.getString("Requestgroup");
}
System.out.println("^^^^^^"+qry);
String count="select count(*) from login where groupname='"+reqname+"'";
rs=st.executeQuery(count);
int gcount=0;
while(rs.next()){
gcount=rs.getInt(1);
}
System.out.println("QQQQQQQQQQQQ"+gcount);
String count1="select usercount from othergroup where username='"+UserName+"'
and Requestgroup='"+reqname+"'";
rs=st.executeQuery(count1);
int acount=0;
while(rs.next()){
acount=rs.getInt(1);
}
System.out.println("RRRRRRRRRRRR"+acount);
if(acount==gcount){
String filepath1=TempDownloadotherDirectory;
System.out.println(filepath1+"?????" +filename1+"....."+TempDownloadotherDirect
ory);
FileInputStreamfiletoDownload=new FileInputStream("D:/temp2/"+filename1);
ServletOutputStream output=response.getOutputStream();
System.out.println(filepath1+"SSSSSSSSSSSSSSSSSS");
response.setContentType("images/jpg");
response.addHeader("Content-Disposition","attachment;filename="+filename1);
response.setContentLength(filetoDownload.available());
int readBytes=0;
byte[] buffer=new byte[1024];
while(filetoDownload.available()>0)
{ output.write(filetoDownload.read());
}
output.close();
filetoDownload.close();
response.sendRedirect("");
}
else{
response.sendRedirect("pending.jsp");
}}
catch(Exception ex)
{
ex.printStackTrace();
}}
/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
@Override
public String getServletInfo() {
return "Short description";

```

```
}}
```

## **Edcrypt**

```
import java.io.*;
import java.security.*;
import java.security.spec.EncodedKeySpec;
import java.security.spec.PKCS8EncodedKeySpec;
import java.security.spec.X509EncodedKeySpec;
import javax.crypto.*;
import org.apache.commons.codec.binary.Hex;
import org.bouncycastle.jce.provider.BouncyCastleProvider;
import sun.misc.BASE64Decoder;
import sun.misc.BASE64Encoder;
public class EDCRYPT {
    String ALGORITHM_USED = "RSA";
    String PROVIDER = "BC";
    private KeyPair key;
    public EDCRYPT() throws NoSuchAlgorithmException
    {
        this.init();
        this.generateKey();
    }
    public void init()
    {
        Security.addProvider(new BouncyCastleProvider());
    }
    public KeyPair generateKey() throws NoSuchAlgorithmException
    {
        KeyPairGenerator keyGen = null;
        try {
            keyGen = KeyPairGenerator.getInstance(ALGORITHM_USED, PROVIDER);
        } catch (NoSuchProviderException e) { e.printStackTrace(); }
        keyGen.initialize(1024);
        key = keyGen.generateKeyPair();
        return key;
    }
    public PublicKey getpublickey()
    {
        return key.getPublic();
    }
    public PrivateKey getprivatekey()
    {
        return key.getPrivate();
    }
    public byte[] encrypt(byte[] text, PublicKey key) throws Exception
    {
        byte[] cipherText = null;
        try
        {
            Cipher cipher = Cipher.getInstance("RSA/ECB/PKCS1Padding", PROVIDER);
            cipher.init(Cipher.ENCRYPT_MODE, key);
            cipherText = cipher.doFinal(text);
        }
    }
}
```

```

    }catch (Exception e){throw e;}
    return cipherText;
    }
    public String encrypt(String text, PublicKey key) throws Exception
    {
        String encryptedText;
        try
        {
            byte[] cipherText = encrypt(text.getBytes(),key);
            encryptedText = encodeToBASE64(cipherText);
        }catch (Exception e){throw e;}return encryptedText;
        }
    public byte[] decrypt(byte[] text, PrivateKey key) throws Exception
    {byte[] dectyptedText = null;
    try{
        Cipher cipher = Cipher.getInstance("RSA/ECB/PKCS1Padding",PROVIDER);
        cipher.init(Cipher.DECRYPT_MODE,key);
        dectyptedText = cipher.doFinal(text);
    }catch (Exception e){throw e;}
    return dectyptedText;
    }
    public String decrypt(String text, PrivateKey key) throws Exception
    {
        String result;
        try{
            byte[] dectyptedText = decrypt(decodeToBASE64(text),key);
            result = new String(dectyptedText);
        }catch (Exception e){throw e;}
        return result;
    }
    public String getKeyAsString(Key key)
    {
        byte[] keyBytes = key.getEncoded();
        BASE64Encoder b64 = new BASE64Encoder();
        return b64.encode(keyBytes);
    }
    public PrivateKey getPrivateKeyFromString(String key) throws Exception
    {
        KeyFactory keyFactory = KeyFactory.getInstance(ALGORITHM_USED);
        BASE64Decoder b64 = new BASE64Decoder();
        EncodedKeySpecprivateKeySpec=new
        PKCS8EncodedKeySpec(b64.decodeBuffer(key));
        PrivateKey privateKey = keyFactory.generatePrivate(privateKeySpec);
        return privateKey;
    }
    public PublicKey getPublicKeyFromString(String key) throws Exception
    {
        BASE64Decoder b64 = new BASE64Decoder();
        KeyFactory keyFactory = KeyFactory.getInstance(ALGORITHM_USED);
        EncodedKeySpecpublicKeySpec=new
        X509EncodedKeySpec(b64.decodeBuffer(key));
        PublicKey publicKey = keyFactory.generatePublic(publicKeySpec);
        return publicKey;
    }

```

```

    }
    private String encodeToBASE64(byte[] bytes)
    {
        BASE64Encoder b64 = new BASE64Encoder();
        return b64.encode(bytes);
    }
    private byte[] decodeToBASE64(String text) throws IOException
    {
        BASE64Decoder b64 = new BASE64Decoder();
        return b64.decodeBuffer(text);
    }
    public static void main(String[] args) throws Exception {
        EDCRYPT rsa= new EDCRYPT();
        String pub=rsa.getKeyAsString(rsa.getpublickey());
        PublicKey key=rsa.getPublicKeyFromString(pub);
        System.out.println("public key ???"+pub);
        String encry=rsa.encrypt("hello world is the first java program for the java
        beginners",key);
        System.out.println("cipher text :"+encry);
        String pri=rsa.getKeyAsString(rsa.getprivatekey());
        System.out.println("private key "+pri);
        String decry=rsa.decrypt(encry,rsa.getPrivateKeyFromString(pri));
        System.out.println("!!!!!!"+decry);
    }}

```

## Group Name Servlet

```

import com.commondb.Common_DB;
import java.io.File;
import java.sql.*;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
/**
 *
 * @author sentamilpandi.m
 */
public class GroupnameServlet extends HttpServlet {
    /**
     * Processes requests for both HTTP
     * <code>GET</code> and
     * <code>POST</code> methods.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request, HttpServletResponse
    response)

```



```

throws ServletException, IOException {
response.setContentType("text/html;charset=UTF-8");
PrintWriter out = response.getWriter();
try {
/*
* TODO output your page here. You may use following sample code.
*/
out.println("<html>");
out.println("<head>");
out.println("<title>Servlet GroupnameServlet</title>");
out.println("</head>");
out.println("<body>");
out.println("<h1>Servlet GroupnameServlet at " + request.getContextPath() +
"</h1>");
out.println("</body>");
out.println("</html>");
} finally {
out.close();
}
}
/**
* Handles the HTTP
* <code>GET</code> method.
*
* @param request servlet request
* @param response servlet response
* @throws ServletException if a servlet-specific error occurs
* @throws IOException if an I/O error occurs
*/
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
processRequest(request, response);
}
/**
* Handles the HTTP
* <code>POST</code> method.
*
* @param request servlet request
* @param response servlet response
* @throws ServletException if a servlet-specific error occurs
* @throws IOException if an I/O error occurs
*/
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
String groupName=request.getParameter("groupname");
String key=request.getParameter("key");
Connection con=null;
Statement st=null;
try{
Class.forName("com.mysql.jdbc.Driver");
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/mona","root","pass

```

```

word");
st=con.createStatement();
intrs=st.executeUpdate("INSERTINTOgroupname(groupname,product)VALUES
('"+groupname1+"','"+key1+"')");
if(rs>0)
{
File file=new File("D:/"+groupname1);
System.out.println(""+groupname1);
if(!(file.exists()))
{
file.mkdir();
}
response.sendRedirect("AdminLinks.jsp");
}
else{
response.sendRedirect("Error.jsp");
}}
catch(Exception ex)
{
ex.printStackTrace();
}}
/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
@Override
public String getServletInfo() {
return "Short description";
}}

```

## Login Servlet

```

import com.commondb.Common_DB;
import java.io.File;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.RequestDispatcher;
import javax.servlet.http.HttpSession;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.logging.Level;
import java.util.logging.Logger;
import java.util.ArrayList;
/**
 *

```

```

* @author sentamilpandi.m
*/
public class LoginServlet extends HttpServlet {
String UserName="";
String Password="";
String Requestgroup;
String filename;
String Email="";
String group="";
Connection con=null;
Statement st=null;
ResultSet rs=null;
RequestDispatcher rd=null;
/**
* Processes requests for both HTTP
* <code>GET</code> and
* <code>POST</code> methods.
*
* @param request servlet request
* @param response servlet response
* @throws ServletException if a servlet-specific error occurs
* @throws IOException if an I/O error occurs
*/
protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
throws ServletException, IOException {
response.setContentType("text/html;charset=UTF-8");
PrintWriter out = response.getWriter();
try {
/*
* TODO output your page here. You may use following sample code.
*/
out.println("<html>");
out.println("<head>");
out.println("<title>Servlet LoginServlet</title>");
out.println("</head>");
out.println("<body>");
out.println("<h1>Servlet LoginServlet at " + request.getContextPath() + "</h1>");
out.println("</body>");
out.println("</html>");
} finally {
out.close();
}}
/**
* Handles the HTTP
* <code>GET</code> method.
*
* @param request servlet request
* @param response servlet response
* @throws ServletException if a servlet-specific error occurs
* @throws IOException if an I/O error occurs
*/

```

```

@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

processRequest(request, response);
}
/**
 * Handles the HTTP
 * <code>POST</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    HttpSession session1=request.getSession(true);
    UserName=request.getParameter("UserName");
    Password=request.getParameter("Password");
    Requestgroup=request.getParameter("Requestgroup");
    filename=request.getParameter("filename");
    try {
        ArrayList list=new ArrayList();
        ResultSet rs1=Common_DB.LoginCheck("mona", "Login",
        "UserName","Password",UserName, Password);
        if(rs1.next()) {
            String group=rs1.getString("groupname");
            System.out.println(">>>>>>>>>" +group);
            File file=new File("D:/" +group);
            if(!(file.exists()))
            {
                file.mkdir();
            }
            File[] files=new File("D:/" +group).listFiles();
            System.out.println(">>>>>>>>>" +files.length);
            for(int i=0;i<files.length;i++) {
                String filename=files[i].getName();
                list.add(filename);
            }
            session1.setAttribute("group", group);
            session1.setAttribute("filename", list);
            session1.setAttribute("username", UserName);
            response.sendRedirect("download.jsp");
            Class.forName("com.mysql.jdbc.Driver");
            con=DriverManager.getConnection("jdbc:mysql://localhost:3306/mona","root","password");
            st=con.createStatement();
            rs=st.executeQuery("select * from login where UserName='"+UserName+"' and
            Password='"+Password+"' and groupname='"+group+"'");
            if(rs.next())

```

```

{
System.out.println("COMING");
response.sendRedirect("download.jsp");
}
else
{
response.sendRedirect("download1.jsp");
}}
else
{
response.sendRedirect("Error.jsp");
}}
catch (Exception ex)
{
ex.printStackTrace();
}}
/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
@Override
public String getServletInfo() {
return "Short description";
}}

```

### Other Group Servlet

```

import com.commondb.Common_DB;
import java.sql.*;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import javax.servlet.RequestDispatcher;
import javax.servlet.annotation.WebServlet;
import java.util.Random;
import javax.naming.*;
import javax.mail.*;
import javax.mail.internet.*;
import com.sun.mail.smtp.*;
import java.util.Properties;
/**
 *
 * @author sentamilpandi.m
 */
@WebServlet(name="OthergroupServlet",urlPatterns= {"/OthergroupServlet"})
public class OthergroupServlet extends HttpServlet {
Connection con=null;

```

```

Statement st=null;
ResultSet rs=null;
RequestDispatcher rd=null;
String group="";
String pending="pending";
protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
throws ServletException, IOException {
response.setContentType("text/html;charset=UTF-8");
PrintWriter out = response.getWriter();
try {
/*
* TODO output your page here. You may use following sample code.
*/
out.println("<html>");
out.println("<head>");
out.println("<title>Servlet OthergroupServlet</title>");
out.println("</head>");
out.println("<body>");
out.println("<h1>Servlet OthergroupServlet at " + request.getContextPath() +
"</h1>");
out.println("</body>");
out.println("</html>");
} finally {
out.close();
}}
/**
* Handles the HTTP
* <code>GET</code> method.
*
* @param request servlet request
* @param response servlet response
* @throws ServletException if a servlet-specific error occurs
* @throws IOException if an I/O error occurs
*/
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
processRequest(request, response);
}
/**
* Handles the HTTP
* <code>POST</code> method.
*
* @param request servlet request
* @param response servlet response
* @throws ServletException if a servlet-specific error occurs
* @throws IOException if an I/O error occurs
*/
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

```

```

HttpSession session1=request.getSession(true);
String Requestgroup=request.getParameter("Requestgroup");
String filename=request.getParameter("filename");
String UserName=session1.getAttribute("username").toString();
String group=session1.getAttribute("group").toString();
String regemail=(String)session1.getAttribute("regemail");
System.out.println(""+regemail);
Connection con=null;
Statement st=null;
ResultSet rs1=null;
int counting=0;
try {
Class.forName("com.mysql.jdbc.Driver");
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/mona","root","pass
word");
st=con.createStatement();
rs1=st.executeQuery("Select count(*) from login where
groupname='"+Requestgroup+"'");
while(rs1.next())
{
counting=rs1.getInt(0);
}
System.out.println(" @ @ @ @ @ @ @ @ "+rs1);
intrs=Common_DB.InsertTable("mona","INSERTINTO
othergroup(username,groupname,Requestgroup,filename,Status,usercount)
VALUES('"+UserName+"','"+group+"','"+Requestgroup+"','"+filename+"','"+pending
+"','"+counting+"')");
if(rs>0)
{
response.sendRedirect("success1.jsp");
}
else{
response.sendRedirect("Error.jsp");
}
session1.setAttribute("Requestgroup", Requestgroup);
}
catch (Exception ex)
{
ex.printStackTrace();
}
}
/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
@Override
public String getServletInfo() {
return "Short description";
}
}

```

## Reg Servlet

```
import com.commondb.Common_DB;
```

```

import java.io.IOException;
import java.io.PrintWriter;
import java.security.PublicKey;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Properties;
import javax.mail.*;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;
import javax.mail.internet.MimeMultipart;
import javax.management.Query;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.RequestDispatcher;
import javax.servlet.http.HttpSession;
import java.util.ArrayList;
/**
 *
 * @author sentamilpandi.m
 */
@WebServlet(name = "RegServlet", urlPatterns = {"/RegServlet"})
public class RegServlet extends HttpServlet {
    String alive="alive";
    static Properties properties = new Properties();
    static
    {
        properties.put("mail.smtp.host", "smtp.gmail.com");
        properties.put("mail.smtp.socketFactory.port", "465");
        properties.put("mail.smtp.socketFactory.class",
            "javax.net.ssl.SSLSocketFactory");
        properties.put("mail.smtp.auth", "true");
        properties.put("mail.smtp.port", "465");
    }
    /**
     * Processes requests for both HTTP
     * <code>GET</code> and
     * <code>POST</code> methods.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request, HttpServletResponse
    response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");

```



```

PrintWriter out = response.getWriter();
try {
/*
 * TODO output your page here. You may use following sample code.
 */
out.println("<html>");
out.println("<head>");
out.println("<title>Servlet RegServlet</title>");
out.println("</head>");
out.println("<body>");
out.println("<h1>Servlet RegServlet at " + request.getContextPath() + "</h1>");
out.println("</body>");
out.println("</html>");
} finally {
out.close();
}}
/**
 * Handles the HTTP
 * <code>GET</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
processRequest(request, response);
}
/**
 * Handles the HTTP
 * <code>POST</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
HttpSession session1=request.getSession();
try
{
String reguser=request.getParameter("reguser");
String regpass=request.getParameter("regpass");
String regemail=request.getParameter("regemail");
String group=request.getParameter("groupname");
System.out.println("????????"+reguser+","+regpass+","+regemail);
session1.setAttribute("regemail", regemail);
int k=Common_DB.InsertTable("mona","INSERT INTO

```

```

login(UserName,Password,Email,groupname)
VALUES(""+reguser+"",""+regpass+"",""+regemail+"",""+group+"");
if(k>0)
{
System.out.println(query);
int q1=Common_DB.InsertTable("mona","USE mona");
int q=Common_DB.InsertTable("mona",query);
System.out.println("*****"+q);
if(q==0)
{
ResultSet
gp=Common_DB.ViewParticularData("mona","groupname","groupname",group);
if(gp.next()){
String key=gp.getString(2);

final String from="ranjeshsamy@gmail.com";
final String password="@ @rajesh89@ @";
final String to="ranjeshsamy@gmail.com";
Session session = Session.getInstance(properties, new javax.mail.Authenticator()
{
protected PasswordAuthentication getPasswordAuthentication() {
return new PasswordAuthentication(from, password);
}});
Message message = new MimeMessage(session);
message.setFrom(new InternetAddress(from));
message.setRecipients(Message.RecipientType.TO,
InternetAddress.parse(regemail));
message.setSubject("public key" + "Private key");
message.setText("public key is "+"\\n"+key + "\\n");
Multipart multipart = new MimeMultipart
Transport.send(message);
System.out.println("Email sent successfully");
response.sendRedirect("Login.jsp");
}
else
{
response.sendRedirect("Error.jsp");
}}
else{
response.sendRedirect("userexist.jsp");
}}
catch(Exception ex)
{
ex.printStackTrace();
response.sendRedirect("userexist.jsp");
}}
/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
@Override

```

```

public String getServletInfo() {
return "Short description";
}

```

## Upload Servlet

```

import com.commondb.Common_DB;
import java.io.*;
import java.security.Key;
import java.sql.*;
import java.sql.ResultSet;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.ServletConfig;
import javax.servlet.ServletException;
import javax.servlet.http.HttpSession;
import org.apache.commons.fileupload.FileItem;
import org.apache.commons.fileupload.FileUploadException;
import org.apache.commons.fileupload.disk.DiskFileItemFactory;
import org.apache.commons.fileupload.servlet.ServletFileUpload;
import org.apache.commons.io.output.*;
/**
 *
 * @author sentamilpandi.m
 */
public class UploadServlet extends HttpServlet {
Connection con=null;
Statement st=null;
ResultSet rs=null;
RequestDispatcher rd=null;
private static Key generateKey(String group) throws Exception
{
String keyValue="";
ResultSet rs1=Common_DB.ViewParticularData("mona", "groupname",
"groupname",group);
String group2="";
if(rs1.next())
{
group2=rs1.getString(2);
}
Key key = new SecretKeySpec(group2.getBytes(), "AES");
return key;
}
/**

```

```

* Processes requests for both HTTP
* <code>GET</code> and
* <code>POST</code> methods.
*
* @param request servlet request
* @param response servlet response
* @throws ServletException if a servlet-specific error occurs
* @throws IOException if an I/O error occurs
*/
protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
throws ServletException, IOException {
response.setContentType("text/html;charset=UTF-8");
PrintWriter out = response.getWriter();
try {
/*
* TODO output your page here. You may use following sample code.
*/
out.println("<html>");
out.println("<head>");
out.println("<title>Servlet UploadServlet</title>");
out.println("</head>");
out.println("<body>");
out.println("<h1>Servlet UploadServlet at " + request.getContextPath() + "</h1>");
out.println("</body>");
out.println("</html>");
} finally {
out.close();
}
}
/**
* Handles the HTTP
* <code>GET</code> method.
*
* @param request servlet request
* @param response servlet response
* @throws ServletException if a servlet-specific error occurs
* @throws IOException if an I/O error occurs
*/
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
}
* Handles the HTTP
* <code>POST</code> method.
*
* @param request servlet request
* @param response servlet response
* @throws ServletException if a servlet-specific error occurs
* @throws IOException if an I/O error occurs
*/
@Override

```

```

protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
String name = null;
HttpSession session1=request.getSession(true);
String UserName=session1.getAttribute("username").toString();
String list=session1.getAttribute("filename").toString();
String group=session1.getAttribute("group").toString();
final String UPLOAD_DIRECTORY = "D:"+group;
String TempUploadDirectory="D:/temp";
System.out.println("?????????" + TempUploadDirectory);
File file=new File(TempUploadDirectory);
if(!(file.exists())) {
file.mkdir();
}
if(ServletFileUpload.isMultipartContent(request)){
try {
String nn="";
List<FileItem> multipart = new ServletFileUpload(
new DiskFileItemFactory()).parseRequest(request);
for(FileItem item : multipart){
if(!item.isFormField()){
name = new File(item.getName()).getName();
item.write( new File(TempUploadDirectory + File.separator + name));
nn=name;
}}
Key key1 = generateKey(group);
AESEncrypter encrypter = new AESEncrypter((SecretKey)key1);
encrypter.encrypt(new FileInputStream(TempUploadDirectory + File.separator +
nn),new FileOutputStream(UPLOAD_DIRECTORY+ File.separator+nn));
request.setAttribute("message", "File Uploaded Successfully");
} catch (Exception ex) {
request.setAttribute("message", "File Upload Failed due to " + ex);
}
}else{
request.setAttribute("message",
"Sorry this Servlet only handles file upload request");
}
try {
int rs=Common_DB.InsertTable("mona", "INSERT INTO
userprofile(UserName,groupname,filename)
VALUES('"+UserName+"','"+group+"','"+name+"')");
if(rs>0)
{
}
} catch (Exception ex) {
ex.printStackTrace();
}
request.getRequestDispatcher("/resultupload.jsp").forward(request, response);
}
/**
* Returns a short description of the servlet.
*

```

```

* @return a String containing servlet description
*/
@Override
public String getServletInfo() {
return "Short description";
}}

```

## User Delete Servlet

```

import com .commondb.Common_DB;
import java.sql.*;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
*
* @author java
*/
public class UserDeleteServlet extends HttpServlet {
**
* Processes requests for both HTTP
* <code>GET</code> and
* <code>POST</code> methods.
*
* @param request servlet request
* @param response servlet response
* @throws ServletException if a servlet-specific error occurs
* @throws IOException if an I/O error occurs
*/
protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
throws ServletException, IOException {
response.setContentType("text/html;charset=UTF-8");
PrintWriter out = response.getWriter();
try {
/*
* TODO output your page here. You may use following sample code.
*/
out.println("<html>");
out.println("<head>");
out.println("<title>Servlet UserDeleteServlet</title>");
out.println("</head>");
out.println("<body>");
out.println("<h1>Servlet UserDeleteServlet at " + request.getContextPath() +
"</h1>");
out.println("</body>");
out.println("</html>");
} finally {
out.close();
}
}
}

```

```

}}
/**
 * Handles the HTTP
 * <code>GET</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
}
/**
 * Handles the HTTP
 * <code>POST</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    HttpSession session=request.getSession(true);
    String gname=request.getParameter("gname");
    String groupkey=request.getParameter("groupkey");
    Connection con=null;
    Statement st=null;
    ResultSet rs=null;
    try{
        Class.forName("com.mysql.jdbc.Driver");
        con=DriverManager.getConnection("jdbc:mysql://localhost:3306/mona","root","pass
word");
        st=con.createStatement();
        String qry="select * groupname where groupname='"+gname+"' and
product='"+groupkey+"'";
        rs=st.executeQuery(qry);
        System.out.println("AAAAAAAAAAAA"+qry);
        ResultSet rs1=Common_DB.LoginCheck("mona", "groupname",
"groupname","product",gname,groupkey);
        System.out.println("AAAAAAAAAAAA"+gname+groupkey);
        if(rs1.next())
        {
            response.sendRedirect("userrevocation.jsp");
        }else{
            response.sendRedirect("userrevocationfail.jsp");
        }
        session.setAttribute("gname", gname);
        session.setAttribute("groupkey", groupkey);

```

```

    }
    catch(Exception ex)
    {
        ex.printStackTrace();
    }
}
/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
@Override
public String getServletInfo() {
    return "Short description";
}}

```

## User Revocation Servlet

```

import com.commondb.Common_DB;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import java.sql.*;
/**
 *
 * @author java
 */
public class UserRevocationServlet extends HttpServlet {

    /**
     * Processes requests for both HTTP
     * <code>GET</code> and
     * <code>POST</code> methods.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request, HttpServletResponse
    response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            /**
             * TODO output your page here. You may use following sample code.
             */
            out.println("<html>");

```



```

out.println("<head>");
out.println("<title>Servlet UserRevocationServlet</title>");
out.println("</head>");
out.println("<body>");
out.println("<h1>Servlet UserRevocationServlet at " + request.getContextPath() +
"</h1>");
out.println("</body>");
out.println("</html>");
} finally {
out.close();
}}
/**
 * Handles the HTTP
 * <code>GET</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
processRequest(request, response);
}
/**
 * Handles the HTTP
 * <code>POST</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
HttpSession session=request.getSession();
String username=request.getParameter("username");
System.out.println("ZZZZZZZZZZ"+username);
Connection con=null;
Statement st=null;
ResultSet rs=null;
try{
Class.forName("com.mysql.jdbc.Driver");
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/mona","root","pass
word");
st=con.createStatement();
String qry="Delete from login where UserName='"+username+"'";
rs=st.executeQuery(qry);
System.out.println("JJJJJJJJJJ"+qry);
Common_DB.FreeQuery("mona","delete from login where

```

```

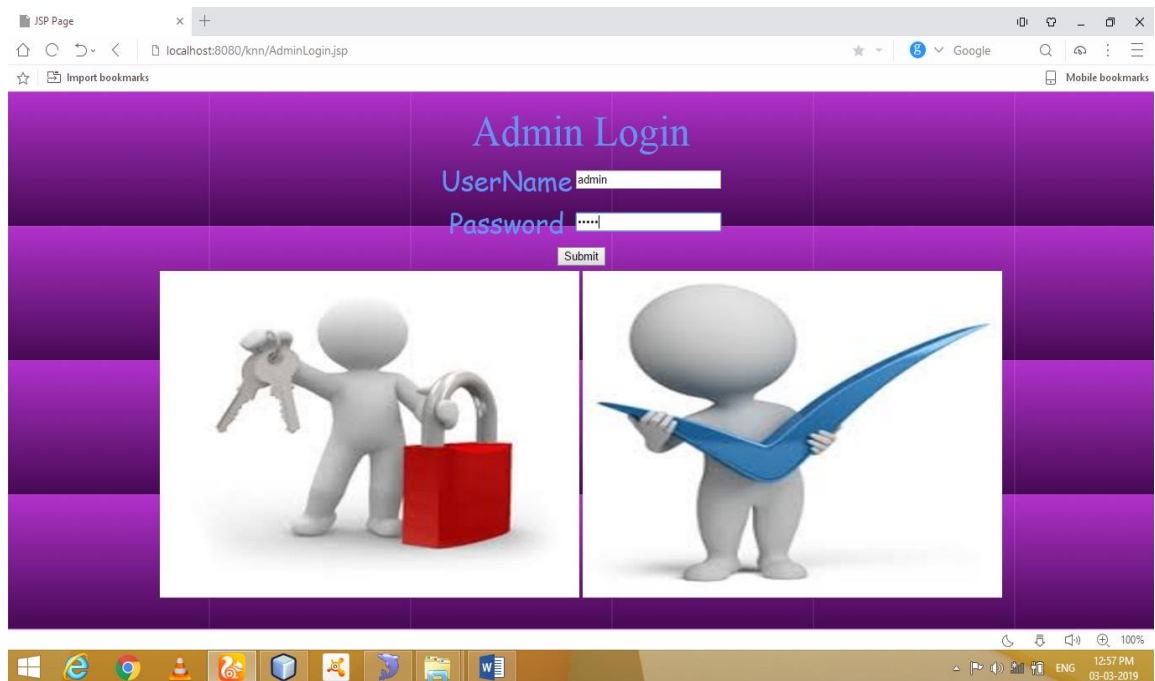
username="" + username + "");
Common_DB.FreeQuery("mona","delete from userprofile where
username="" + username + "");
Common_DB.FreeQuery("mona","delete from othergroup where
username="" + username + "");
if(rs.next())
{
}
response.sendRedirect("Revsuccess.jsp");
}
catch(Exception ex){
}}
/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
@Override
public String getServletInfo() {
return "Short description";
}}

```

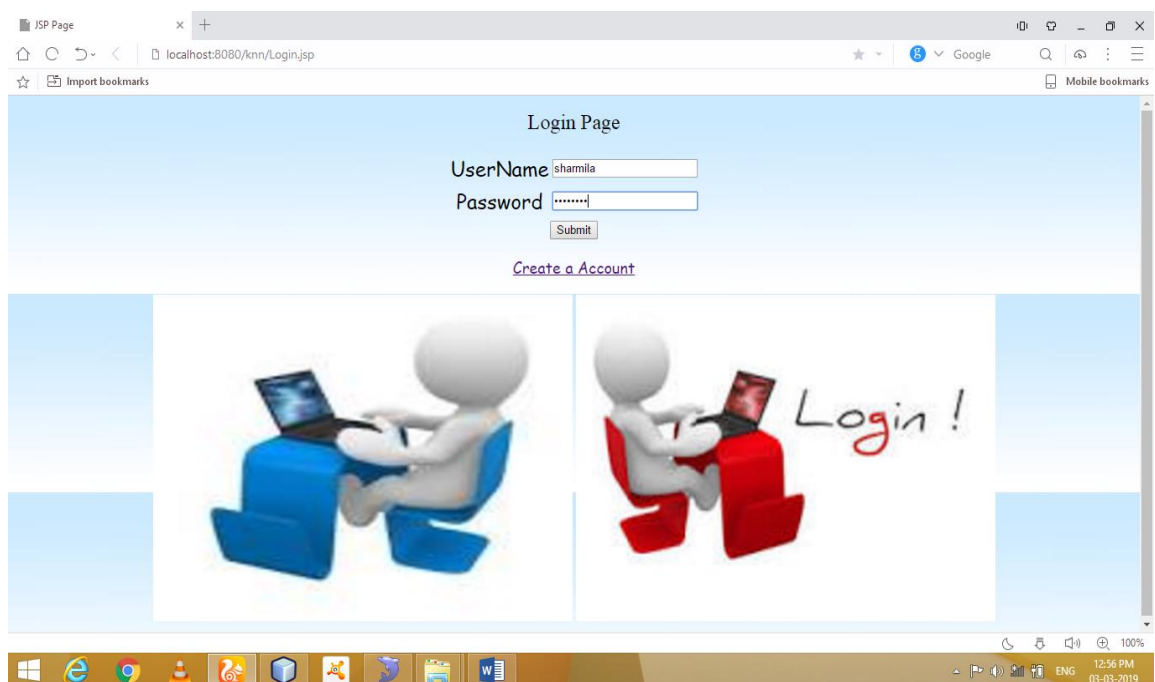
## APPENDIX 2

### SCREENSHOTS

#### Admin Login



#### User Login



## Registration

JSP Page x +

localhost:8080/knn/Reg.jsp

☆ Import bookmarks

Mobile bookmarks

### User Registration Details


UserName:

Password:

Confirm Password:

Email:

Group Name:



Windows taskbar: 12:59 PM 03-03-2019

## Create New Group

JSP Page x +

localhost:8080/knn/groupname.jsp


☆ Import bookmarks

Mobile bookmarks

### GroupName Registration

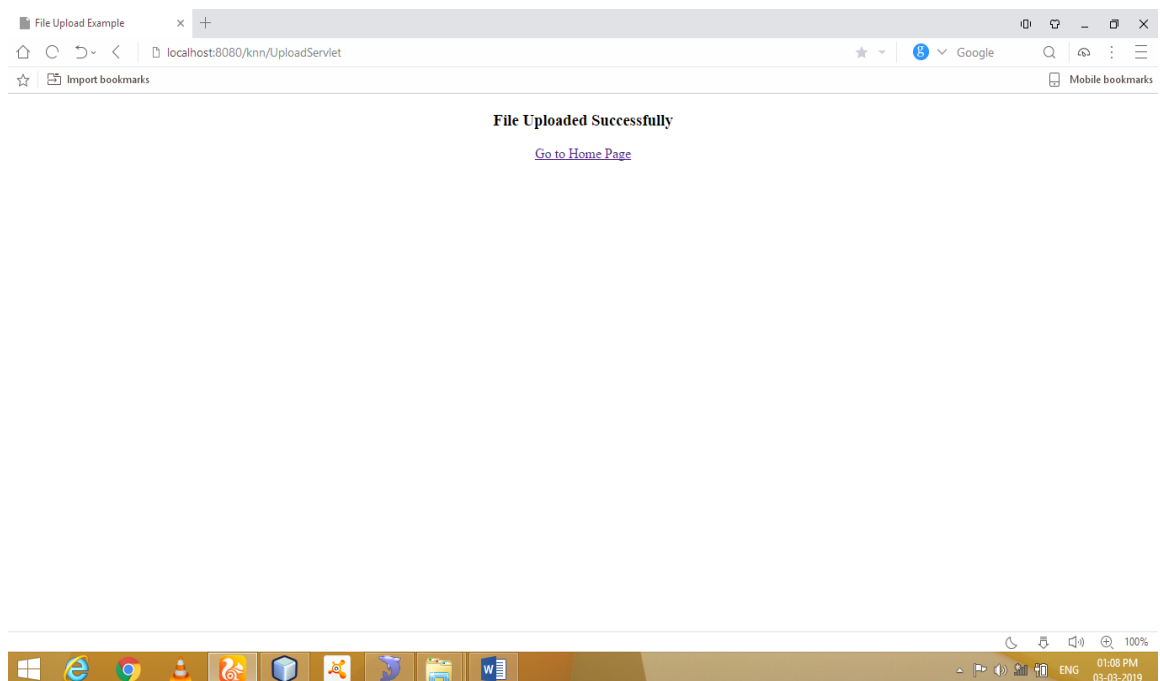
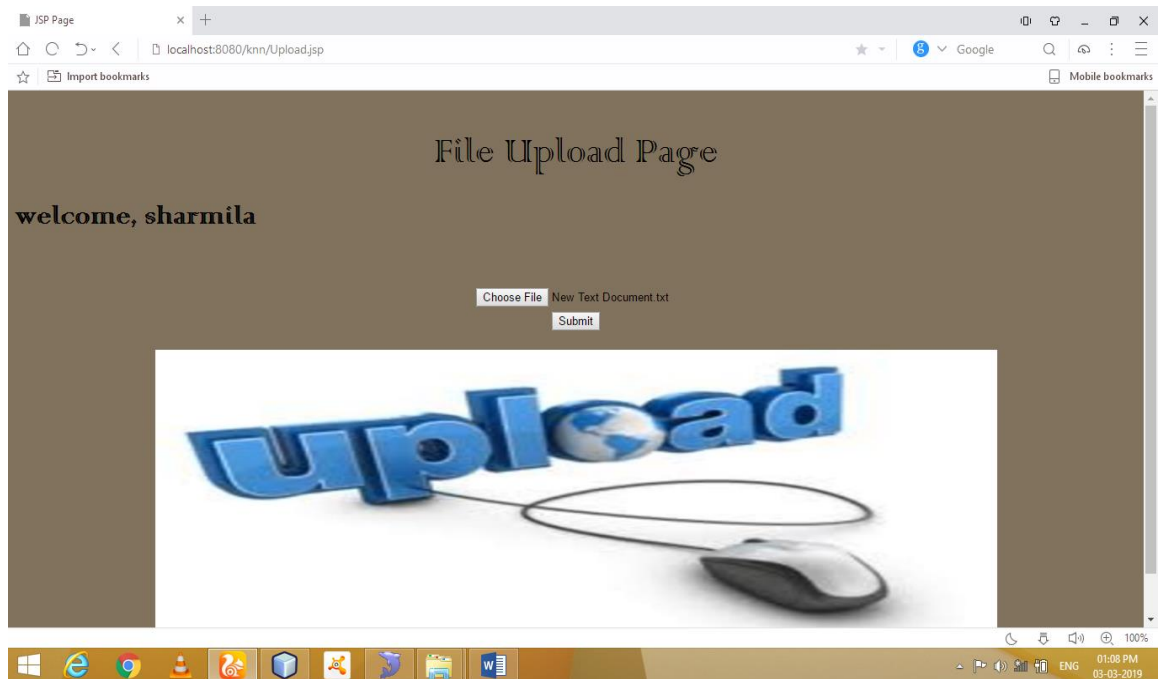
GroupName:

GroupKey:

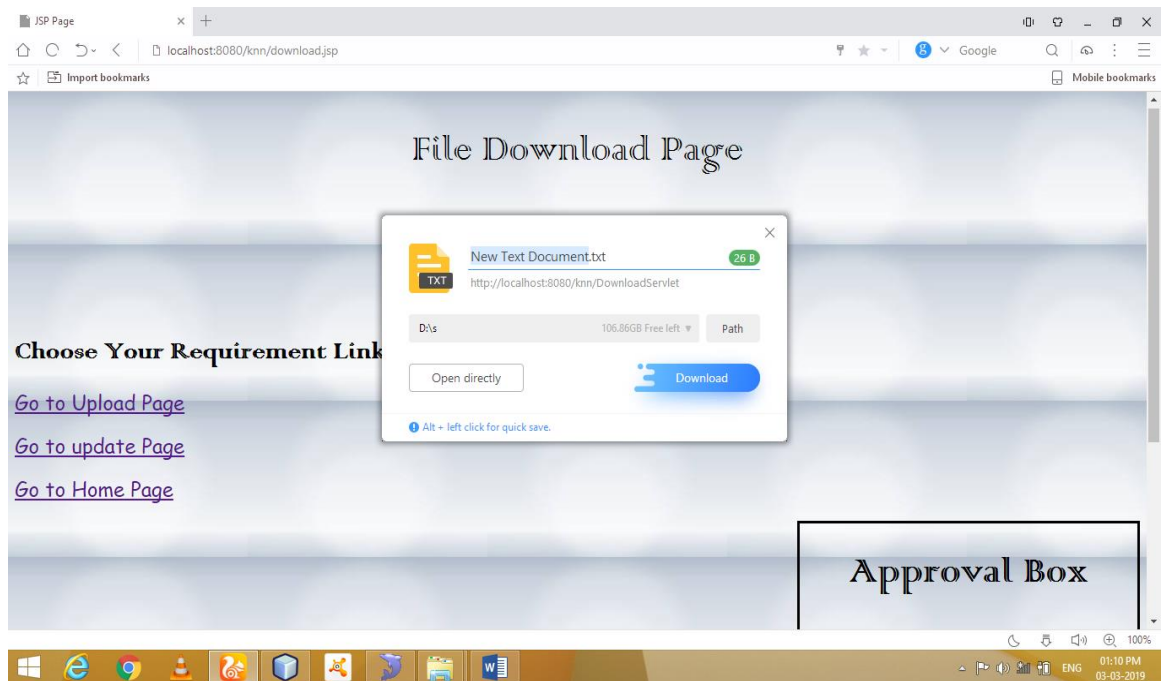
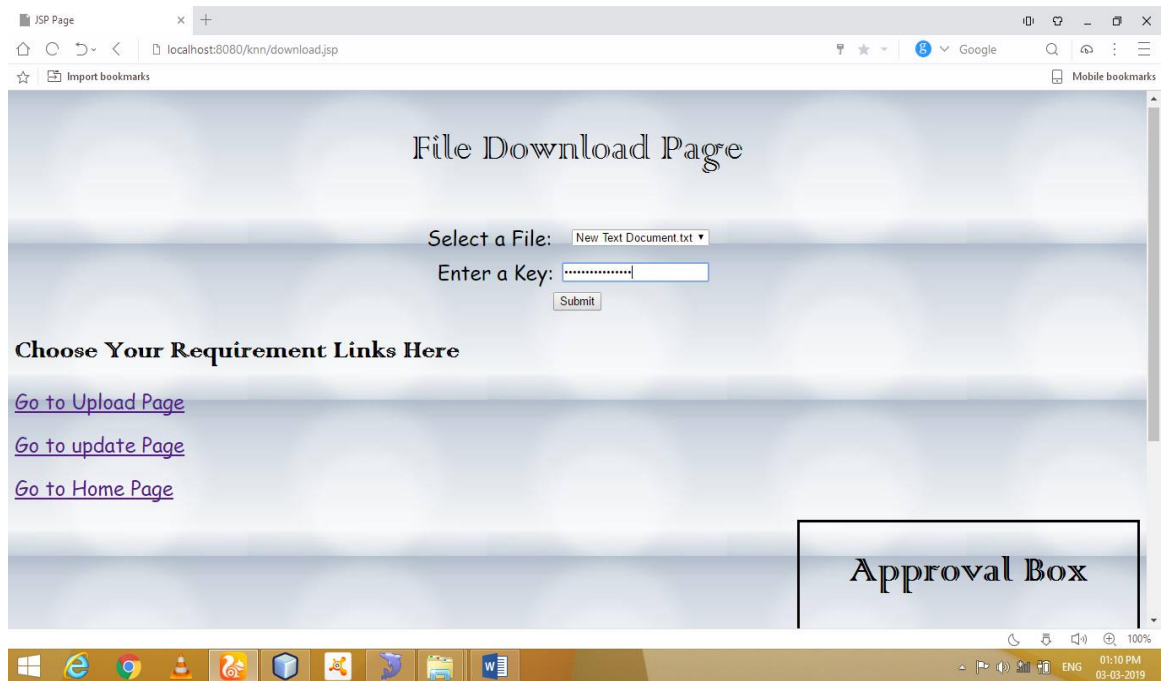


Windows taskbar: 01:01 PM 03-03-2019

# Data Upload Page

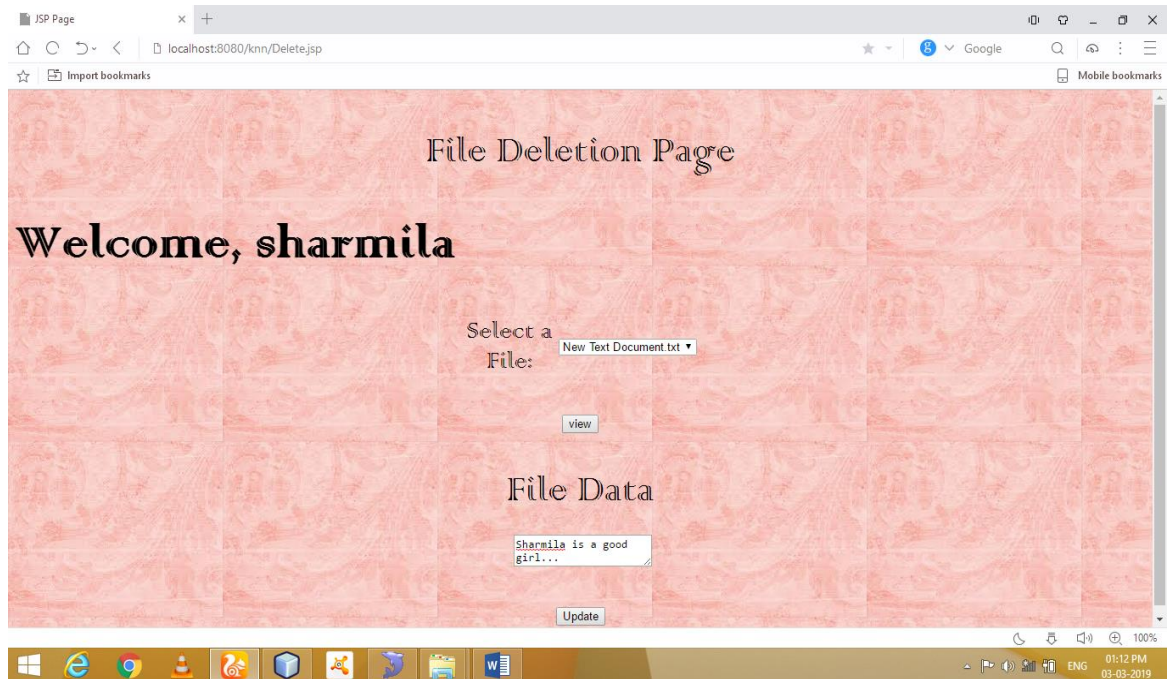


# File Download Page

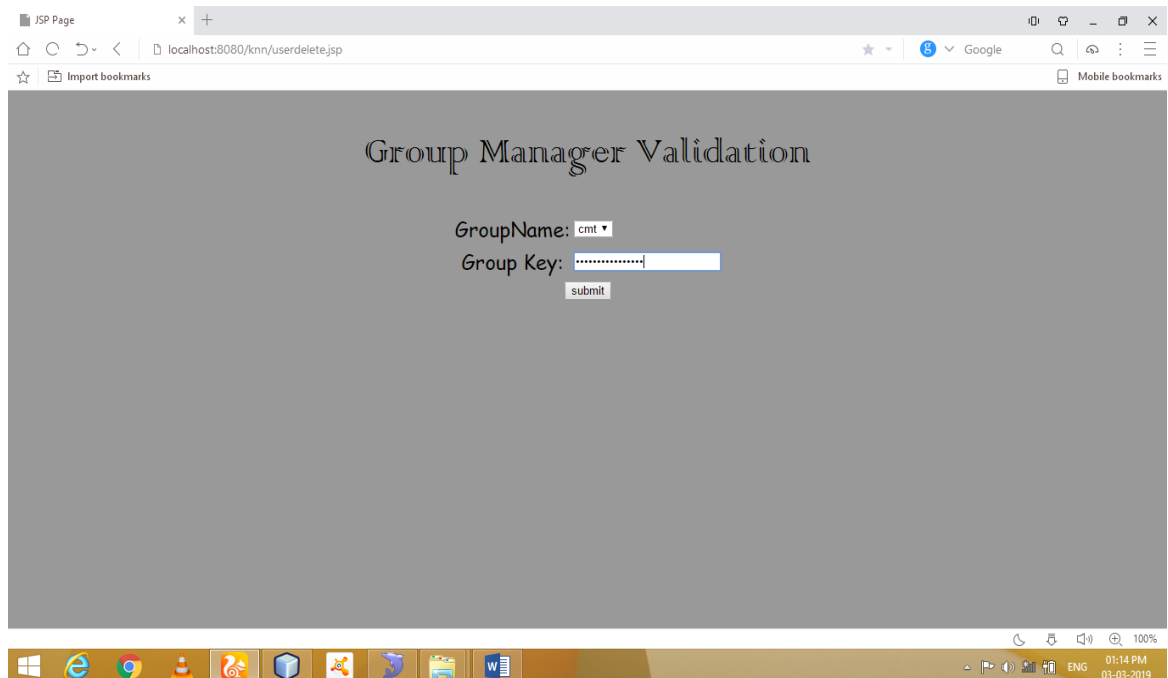




## Update Page



## User Revocation Details



# Find Third Party

localhost:8080/knn/userpage.jsp

Google

Mobile bookmarks

Unknown user

User Name
benin
benin
raji
sharmila

Windows Taskbar

01:15 PM 03-03-2019

# Performance Analysis

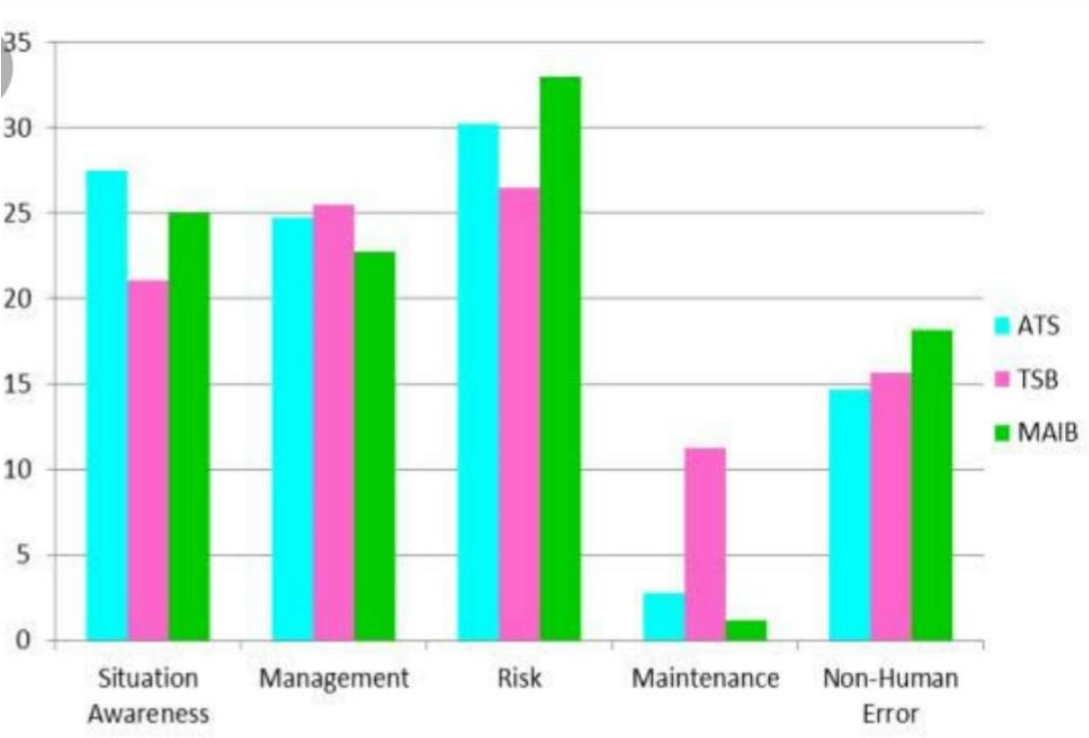


Fig 4.15 computational outsourcing