A
Mini Project Report
On

CLOUD COMPUTING SECURE KEYWORD SEARCH AND DATA SHARING

Submitted to

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY

in partial fulfillment of the requirement
for the award of the degree of

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING (AI & ML)

SREE CHAITANYA
EDUCATIONAL INSTITUTIONS

SUBMITTED BY:

MUTHYAM PRANAVI (22TR1A6639)
NAGUNURI DEEKSHITHA (22TR1A6640)
PATHURI HARIKESHA (22TR1A6645)
MARRI YASWANTH (22TR1A6634)

Under the Guidance of
K. SIREESHA
ASSISTANT PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (AI & ML)
SREE CHAITANYA INSTITUTE OF TECHNOLOGICAL SCIENCES
LMD COLONY, KARIMNAGAR-505527
(Approved by AICTE, Affiliated to JNTUH, Hyderabad)

JUNE-2025

SREE CHAITANYA INSTITUTE OF TECHNOLOGICAL SCIENCES
LMD COLONY, KARIMNAGAR-505527
(Approved by AICTE, Affiliated to JNTUH)

Certificate

SREE CHAITANYA
EDUCATIONAL INSTITUTIONS

Certified that this Mini Project Report entitled, **"CLOUD COMPUTING SECURE KEYWORD SEARCH AND DATA SHARING"** is the bonafide work of **MUTHYAM PRANAVI (H.T.No.22TR1A6639), NAGUNURI DEEKSHITHA (H.T.No. 22TR1A6640), PATHURI HARIKESHA (H.T.No. 22TR1A6645)** and **MARRI YASHWANTH (H.T.No. 22TR1A6634)** **of III Year, CSE (AI & ML)** in the year 2025 in partial fulfillment of the requirements to award the Degree of Bachelor of Technology in **COMPUTER SCIENCE AND ENGINEERING (AI & ML)** of Sree Chaitanya Institute of Technological Sciences, Karimnagar.

K. SIREESHA
Assistant Professor
Project Guide

Dr. CHADA SAMPATH REDDY
Professor
Head of the Department

External Examiner

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to our Project Guide, **Mrs. K. SIREESHA, Assistant Professor** whose knowledge and guidance has motivated us to achieve goals we never thought possible. The time we have spent working under his supervision has truly been a pleasure.

We are thankful to **Mrs. SHAGUFTHA BASHEER, Assistant Professor & Project Coordinator** of AIML Department for her effort, guidance and all faculty members of AIML Department for their help during my course. Thanks to programmers and non-teaching staff of AIML Department, Sree Chaitanya Institute of Technological Sciences.

We also thank **Dr. CHADA SAMPATH REDDY, HOD & Professor of CSE (AI&ML) Department** for providing seamless support and knowledge for the entire project work and also for providing right suggestions at every phase of the development of the project. He has consistently been a source of motivation, encouragement, and inspiration.

It is a great pleasure to convey our thanks to our principal **Dr. A. PRASAD RAJU, Principal**, Sree Chaitanya Institute of Technological Sciences and the College Management for permitting us to undertake this project and providing excellent facilities to carry out our project work.

Finally Special thanks to our parents, sisters and brothers for their support and encouragement throughout my life and this course. Thanks to all our friends and well wishers for their constant support.

DECLARATION

We hereby declare that the work which is being presented in this report entitled, "**CLOUD COMPUTING SECURE KEYWORD SEARCH AND DATA SHARING"**", submitted towards the partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **CSE (AI&ML)**, Sree Chaitanya Institute of Technological Sciences, Karimnagar is an authentic record of our own work carried out under the supervision of **Mr. GUIDE NAME, Assistant Professor, Department of CSE (AI&ML),** Sree Chaitanya Institute of Technological Sciences, Karimnagar.

To the best of our knowledge and belief, this project bears no resemblance with any report submitted to Sree Chaitanya Institute of Technological Sciences or any other University for the award of any degree or diploma.

Date:
Place: SCITS, KARIMNAGAR

MUTHYAM PRANAVI
H.T.No: 22TR1A6639
NAGUNURI DEEKSHITHA
H.T.No: 22TR1A6640
PATHURI HARIKESHA
H.T.No: 22TR1A6645
MARRI YASHWANTH
H.T.No: 22TR1A6634

ABSTRACT

The emergence of cloud infrastructure has significantly reduced the costs of hardware and software resources in computing infrastructure. To ensure security, the data is usually encrypted before it's outsourced to the cloud. Unlike searching and sharing the plain data, it is challenging to search and share the data after encryption. Nevertheless, it is a critical task for the cloud service provider as the users expect the cloud to conduct a quick search and return the result without losing data confidentiality. To overcome these problems, we propose a ciphertext-policy attribute-based mechanism with keyword search and data sharing (CPAB-KSDS) for encrypted cloud data. The proposed solution not only supports attribute-based keyword search but also enables attribute-based data sharing at the same time, which is in contrast to the existing solutions that only support either one of two features. Additionally, the keyword in our scheme can be updated during the sharing phase without interacting with the PKG. In this project, we describe the notion of CPAB-KSDS as well as its security model. Besides, we propose a concrete scheme and prove that it is against chosen ciphertext attack and chosen keyword attack secure in the random oracle model. Finally, the proposed construction is demonstrated practical and efficient in the performance and property comparison..

LIST OF FIGURES

TABLE OF CONTENTS

CHAPTER-1

INTRODUCTION

Cloud computing has been the remedy to the problem of personal data management and maintenance due to the growth of personal electronic devices. It is because users can outsource their data to the cloud with ease and low cost. The emergence of cloud computing has also influenced and dominated Information Technology industries. It is unavoidable that cloud computing also suffers from security and privacy challenges.

Encryption is the basic method for enabling data confidentiality and attribute-based encryption is a prominent representative due to its expressiveness in user's identity and data. After the attribute-based encrypted data is uploaded in the cloud, authorized users face two basic operations: data searching and data sharing. Unfortunately, traditional attribute- based encryption just ensures the confidentiality of data. Hence, it does not support searching and sharing. Suppose in a Person Health Record (PHR) system, a group of patients store their encrypted personal health reports Enc;;Enc in the cloud, where Enc is an attribute-based encryption of the health report Di under an access policy Pi and a keyword KWi. Doctors satisfying the policy Pi can recover the record Di. However, they could not retrieve the specific record by simply typing the keyword. Instead, a doctor Alice needs to first download and decrypt the encrypted records. After decryption, she can use the keyword to search the specific one from a bunch of the decrypted health records. Another inconvenient scenario is that Alice attempts to share a record with her colleague, in the case like she needs to consult the report with a specialist. In this situation, she must download the encrypted files, then decrypt them. Then, after she has acquired the underlying record, she encrypts the record using the policy of the specialist. As a result, this system is very inefficient in terms of searching and sharing.

Additionally, the traditional attribute-based encryption (ABE) technology used in the current PHR systems might cause another issue for keyword maintenance because the ABE algorithm could not scale well for keyword updates once the number of the records significantly increases. For example, after reviewing a health report with the patient selfmarked “contagious” tag, Alice from hospital A confirmed it is not the contagious condition and corrected the tag to “non- contagious”. In order for Alice to share a health report that is encrypted with a tag “contagious” with another doctor from hospital B, she need to change the tag as “non-contagious” without decrypting the report. As the traditional attribute-based encryption with keyword search cannot support keyword updating, Alice has to generate a new tag for all shared ciphertexts so as to keep the privacy of the keyword. From above scenarios, the traditional attribute-based encryption is not flexible for data searching and sharing. Additionally, attribute-based encryption is not well scaled when there is an update request to the keyword. In order to search and share a specific record, Alice downloads and decrypts the ciphertexts. However, this process is impractical to Alice especially when there is a tremendous number of ciphertexts. The worse situation is the data owner Alice should stay online all the time because Alice needs to provide her private key for the data decryption. Thus, ABE solution does not take the advantages of cloud computing.

An alternative method is to delegate a third party to do the search, re-encrypt and keyword update work instead of Alice. Alice can store her private key in the third party’s storage, and thus the third party can do the heavy job on behalf of Alice. In such an approach, however, we need to fully trust the third party since it can access to Alice’s private key. If the third party is compromised, all the user data including sensitive privacy will be leaked as well. It would be a severe disaster to the us

CHAPTER 2

PROJECT ANALYSIS

2.1REQUIREMENT SPECIFICATIONS

The software requirements specification specifies the function requirements and non-functional requirements. Functional requirements refer to how the system is going to react according to the input provided and how it is going to behave in particular situations and non-functional requirements refer to Usability, Availability, Reliability, Performance, Security, Supportability, Interface

SYSTEM REQUIREMENTS

- **H/W System Configuration: -**
- Processor - Pentium –IV
- RAM - 4 GB
- Hard Disk - 20 GB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - SVGA

Software Requirements:

- Operating System - Windows XP
- Coding Language - Java/J2EE(JSP,Servlet)
- Front End - J2EE
- Back End - MySQL

2.2 EXISTING SYSTEM

In an ABE, the users' identities are described by a list of attributes. After ABE's pioneering work, several scholars extended the notion of ABE. For example, key policy attribute-based encryption (KP-ABE), where the private key of a user is related to an access policy and the cipher text corresponds to an attribute set. In contrast, there is another example called cipher text-policy attribute-based encryption

(CP-ABE). where the private key is generated with an attribute set and the cipher text is related to an access policy. In both KP-ABE and CP-ABE, the cipher text length is linear with the size of the access policy. To reduce the cipher text length, Emura et al. Proposed a cipher text-policy attribute-based encryption scheme with constant cipher text length. Although it supports the AND-gates on multi attributes, it doesn't support the monotonic express on attributes. After that, a number of constructions have come out to enhance the efficiency, security and expressiveness. To illustrate the ABE's application, Li et al. Adopted the notion of attribute-based encryption in the PHR system to achieve fine grained access control on personal health records.

A cipher text policy attribute-based encryption with hidden policy was proposed to hide the access policy which may leak the user's privacy in the PHR system. The concept of outsourcing decryption attribute-based encryption was introduced to enable a computation-constrained mobile device to outsource most of the decryption work to a service provider. However, there is no guarantee that the service provider could return the correct partial decryption cipher text. To overcome this issue, Lai and Li proposed attribute-based encryption with verifiable outsourced decryption schemes respectively.

Proxy re-encryption was designed to delegate the decryption. Prior work has focused on the scheme's functionality, efficiency, and

security model. Later, Liang et al. Presented an attribute-based proxy re-encryption (AB-PRE) scheme by using proxy re-encryption to a attribute based setting. Meanwhile, another AB-PRE scheme was proposed to support "AND" gates on positive and negative attributes. Following their work, Liang et al. Proposed a cipher text-policy attribute-based proxy re-encryption (CPABPRE) scheme supporting a monotonic access formula in the selective model. Later, the security has been improved in an adaptive model. Ge et al. presented two KPABE schemes that are secure in the selective and adaptive model respectively. Liang et al. Proposed a deterministic finite automata(DFA) based PRE scheme, where the access policy is viewed as a DFA. Unfortunately, the privacy could not be preserved in keyword search in all of these schemes.

Disadvantages

- In the existing work, the system does not provide **Data integrity proof**.
- This system is less performance due to lack of strong encryption techniques.

2.3 PROPOSED SYSTEM

The proposed system first introduces a ciphertext-policy attribute-based mechanism with keyword search and data sharing (CPAB-KSDS) for encrypted cloud data. The searching and sharing functionality are enabled in the ciphertext-policy setting. Furthermore, our scheme supports the keyword to be updated during the sharing phase. After presenting the construction of our mechanism, we proof its chosen ciphertext attack (CCA) and chosen keyword attack (CKA) security in the random oracle model. The proposed construction is demonstrated practical and efficient in the performance and property comparison.

Advantages

1) allows the data owner to search and share the encrypted health report without the unnecessary decryption process.
2) supports keyword updating during the data sharing phase. 3) more importantly, does not need the exist of the PKG, either in the phase of data sharing or keyword updating.
3) the data owner can fully decide who could access the data he encrypted

2.4 PRELIMINARY INVESTIGATION

The first and foremost strategy for development of a project starts from the thought of designing a mail enabled platform for a small firm in which it is easy and convenient of sending and receiving messages, there is a search engine, address book and also including some entertaining games. When it is approved by the organization and our project guide the first activity, ie. preliminary investigation begins. The activity has three parts:

- **Request Clarification**
- **Feasibility Study**
- **Request Approval**

REQUEST CLARIFICATION

After the approval of the request to the organization and project guide, with an investigation being considered, the project request must be examined to determine precisely what the system requires.

Here our project is basically meant for users within the company whose systems can be interconnected by the Local Area Network (LAN). In today's busy schedule man need everything should be provided in a readymade manner. So taking into consideration of the

vastly use of the net in day to day life, the corresponding development of the portal came into existence.

2.5 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must

not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

2.6 FEASIBILITY ANALYSIS

An important outcome of preliminary investigation is the determination that the system request is feasible. This is possible only if it is feasible within limited resource and time. The different feasibilities that have to be analyzed are

- **Operational Feasibility**
- **Economic Feasibility**
- **Technical Feasibility**

Operational Feasibility

Operational Feasibility deals with the study of prospects of the system to be developed. This system operationally eliminates all the tensions of the admin and helps him in effectively tracking the project

progress. This kind of automation will surely reduce the time and energy, which previously consumed in manual work. Based on the study, the system is proved to be operationally feasible.

Economic Feasibility

Economic Feasibility or Cost-benefit is an assessment of the economic justification for a computer based project. As hardware was installed from the beginning & for lots of purposes thus the cost on project of hardware is low. Since the system is a network based, any number of employees connected to the LAN within that organization can use this tool from at anytime. The Virtual Private Network is to be developed using the existing resources of the organization. So the project is economically feasible.

Technical Feasibility

According to Roger S. Pressman, Technical Feasibility is the assessment of the technical resources of the organization. The organization needs IBM compatible machines with a graphical web browser connected to the Internet and Intranet. The system is developed for platform Independent environment. Java Server Pages, JavaScript, HTML, SQL server and WebLogic Server are used to develop the system. The technical feasibility has been carried out. The system is technically feasible for development and can be developed with the existing facility.

CHAPTER-3

PROJECT DESIGN

3.1 INPUT DESIGN

Input Design plays a vital role in the life cycle of software development, it requires very careful attention of developers. The input design is to feed data to the application as accurate as possible. So inputs are supposed to be designed effectively so that the errors occurring while feeding are minimized. According to Software Engineering Concepts, the input forms or screens are designed to provide to have a validation control over the input limit, range and other related validations.

This system has input screens in almost all the modules. Error messages are developed to alert the user whenever he commits some mistakes and guides him in the right way so that invalid entries are not made. Let us see deeply about this under module design.

Input design is the process of converting the user created input into a computer-based format. The goal of the input design is to make the data entry logical and free from errors. The error is in the input are controlled by the input design. The application has been developed in user-friendly manner. The forms have been designed in such a way during the processing the cursor is placed in the position where must be entered. The user is also provided with in an option to select an appropriate input from various alternatives related to the field in certain cases.

Validations are required for each data entered. Whenever a user enters an erroneous data, error message is displayed and the user can move on to the subsequent pages after completing all the entries in the current page.

3.2 OUTPUT DESIGN

The Output from the computer is required to mainly create an efficient method of communication within the company primarily among the project leader and his team members, in other words, the administrator and the clients. The output of VPN is the system which allows the project leader to manage his clients in terms of creating new clients and assigning new projects to them, maintaining a record of the project validity and providing folder level access to each client on the user side depending on the projects allotted to him. After completion of a project, a new project may be assigned to the client. User authentication procedures are maintained at the initial stages itself. A new user may be created by the administrator himself or a user can himself register as a new user but the task of assigning projects and validating a new user rests with the administrator only.

The application starts running when it is executed for the first time. The server has to be started and then the internet explorer in used as the browser. The project will run on the local area network so the server machine will serve as the administrator while the other connected systems can act as the clients. The developed system is highly user friendly and can be easily understood by anyone using it even for the first time.

3.3 SYSTEM ARCHITECTURE

An architectural diagram is a diagram of a system that is used to abstract the overall outline of the software system and the relationships, constraints, and boundaries between components. It is an important tool as it provides an overall view of the physical deployment of the software system and its evolution roadmap. An architecture diagram is a graphical representation of a set of concepts, that are part of an architecture, including their principles, elements and components.

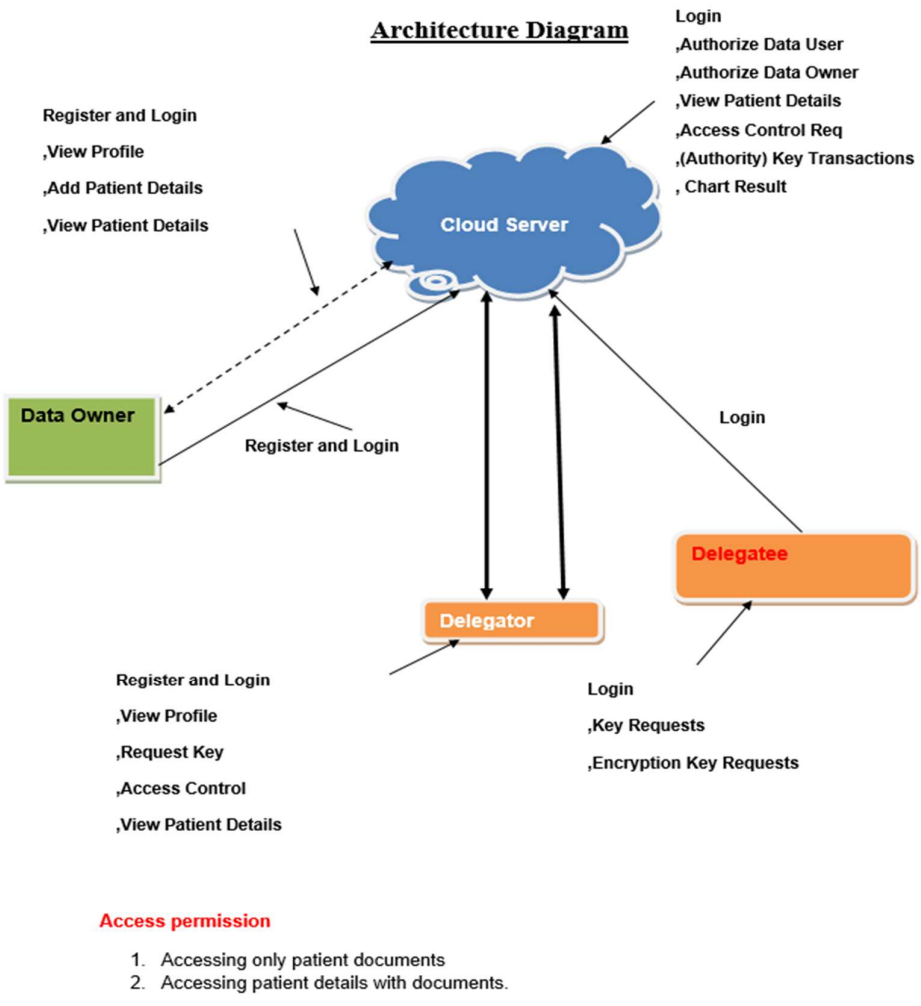


Fig.No:3.3: Architecture Diagram

3.4 Data Flow Diagram

The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of the input data to the system, various processing carried out on these data, and the output data is generated by the system**.**

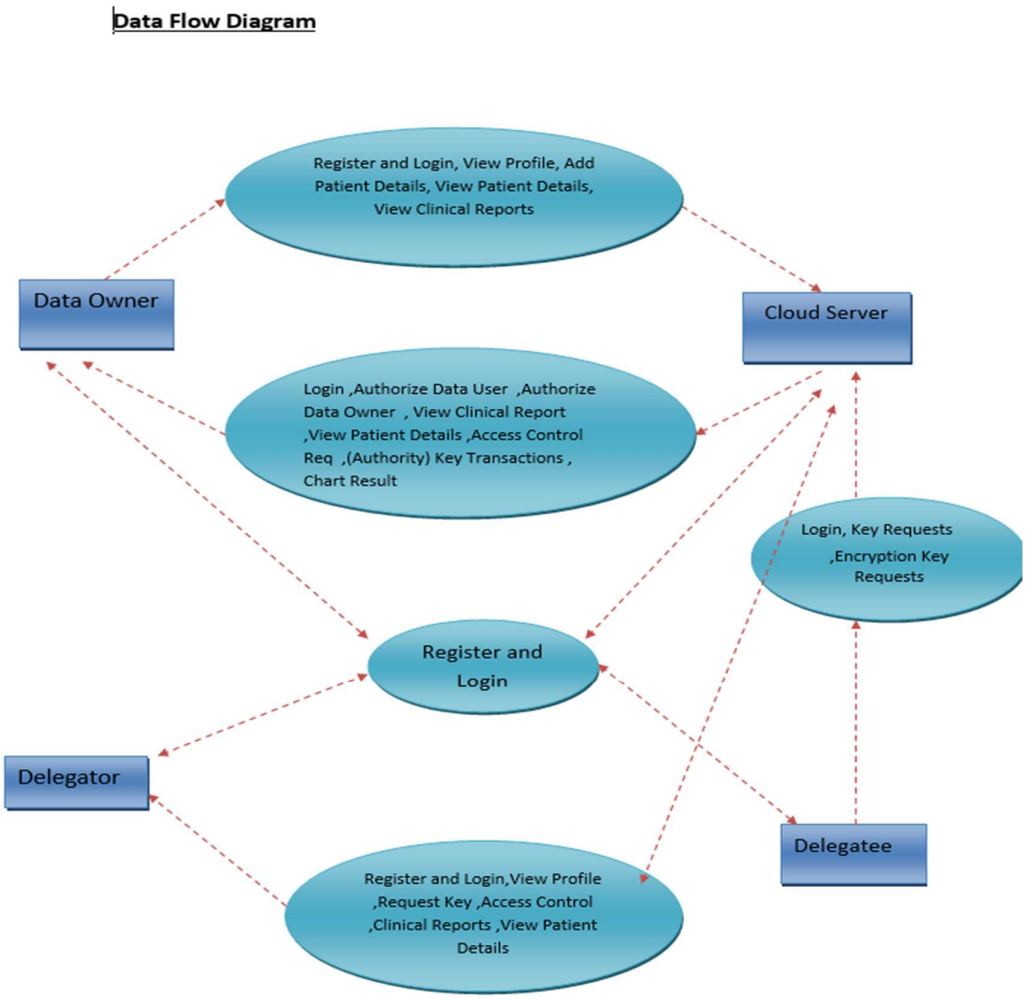


Fig.No:3.4 Data Flow Diagram

3.4.1 Flowchart

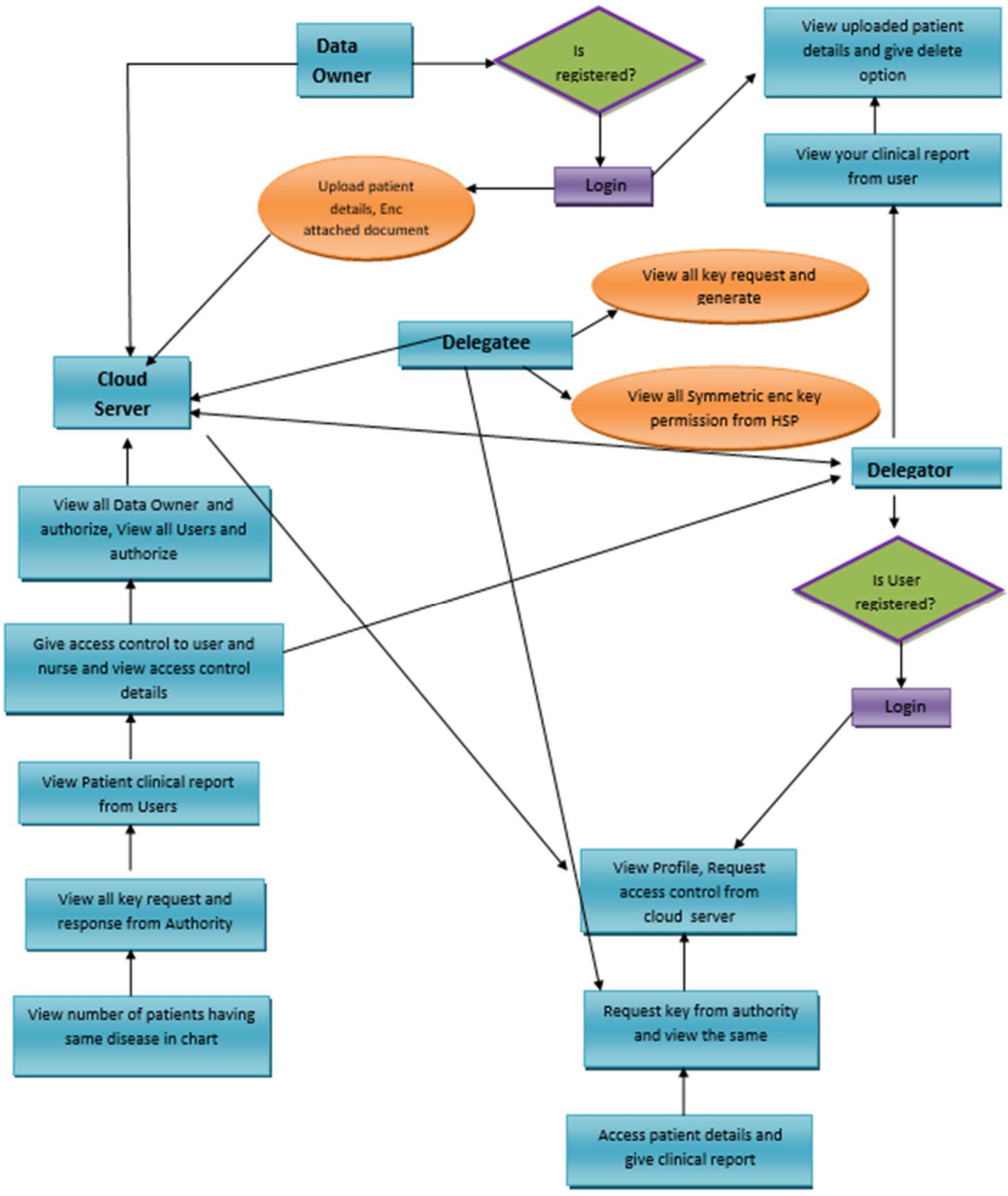


Fig.No:3.4.1 Flowchart

-

3.5 UNIFIED MODELLING LANGUAGE

UML stands for Unified Modeling Language. UML is a standardized general purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML. The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems. The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

UML DIAGRAMS:

A UML diagram is a diagram based on the UML (Unified Modeling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system.

3.5.1. 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. The main purpose of a use case diagram is to show what system functions are performed for which actor.

Roles of the actors in the system can be depicted.

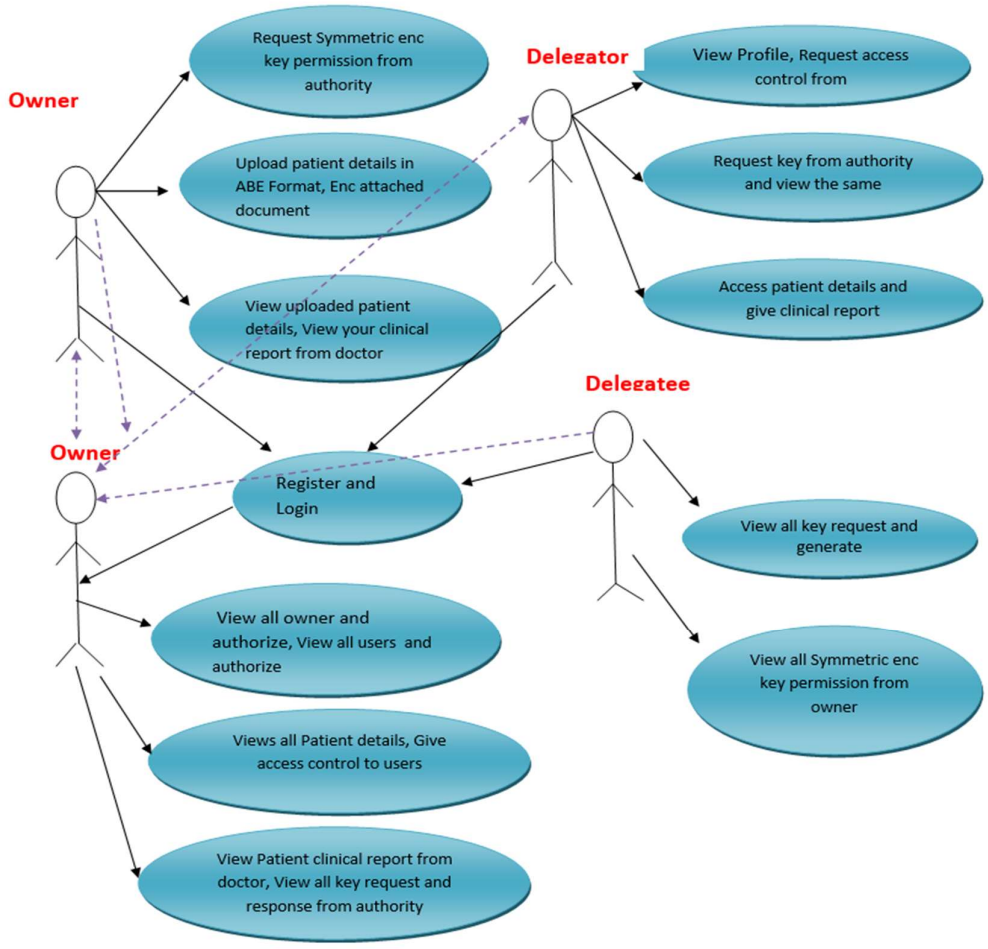


Fig.No:3.5.1. USE CASE DIAGRAM

3.5.2. CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes.

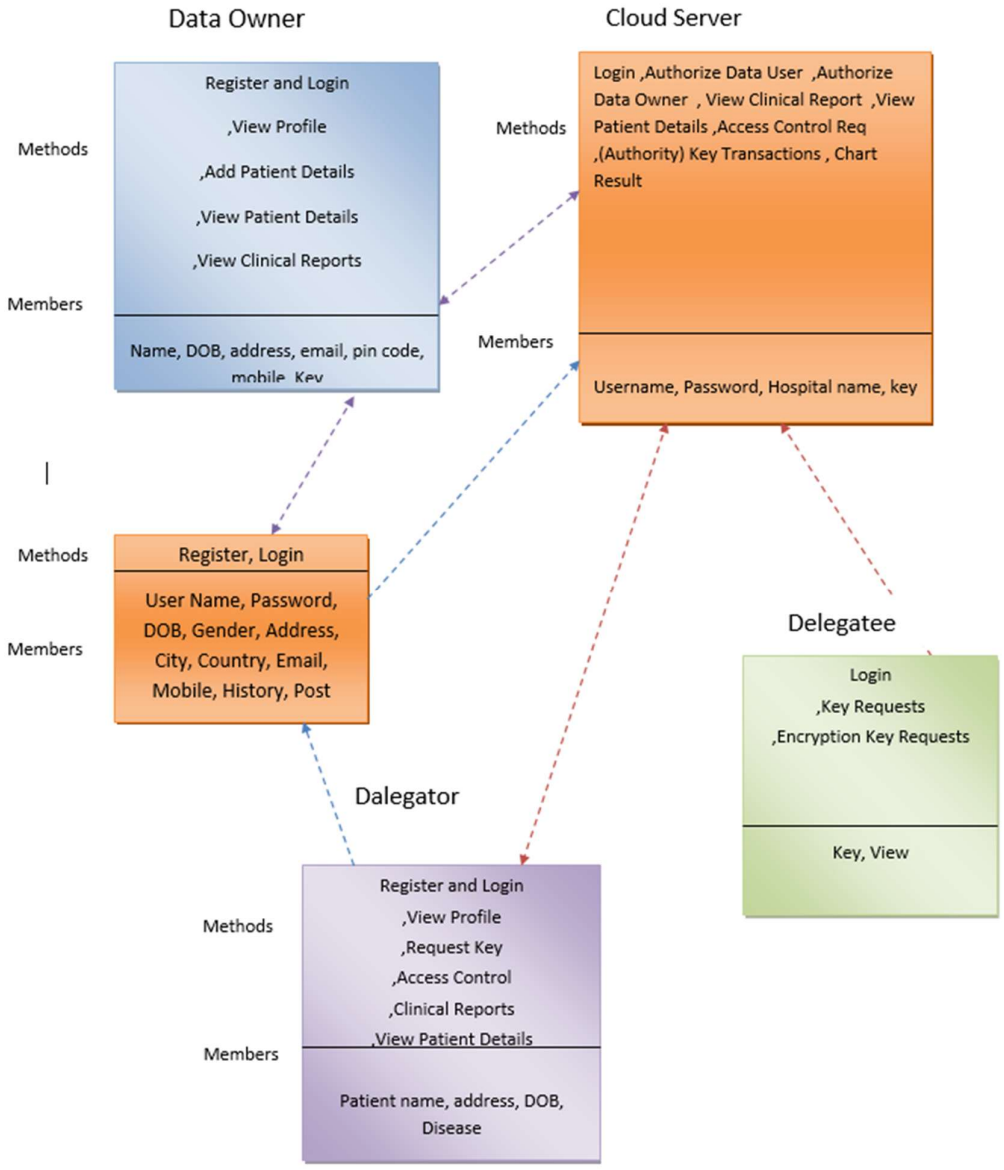


Fig.No:3.5.2. CLASS DIAGRAM

The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed.

In the diagram, classes are represented with boxes which contain three parts

- The upper part holds the name of the class
- The middle part contains the attributes of the class
- The bottom part gives the methods or operations the class can take or undertake

In the design of a system, a number of classes are identified and grouped together in a class diagram which helps to determine the static relations between those objects. With detailed modeling, the classes of the conceptual design are often split into a number of subclasses.

3.5.3 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 diagrams are sometimes called event diagrams, event scenarios.

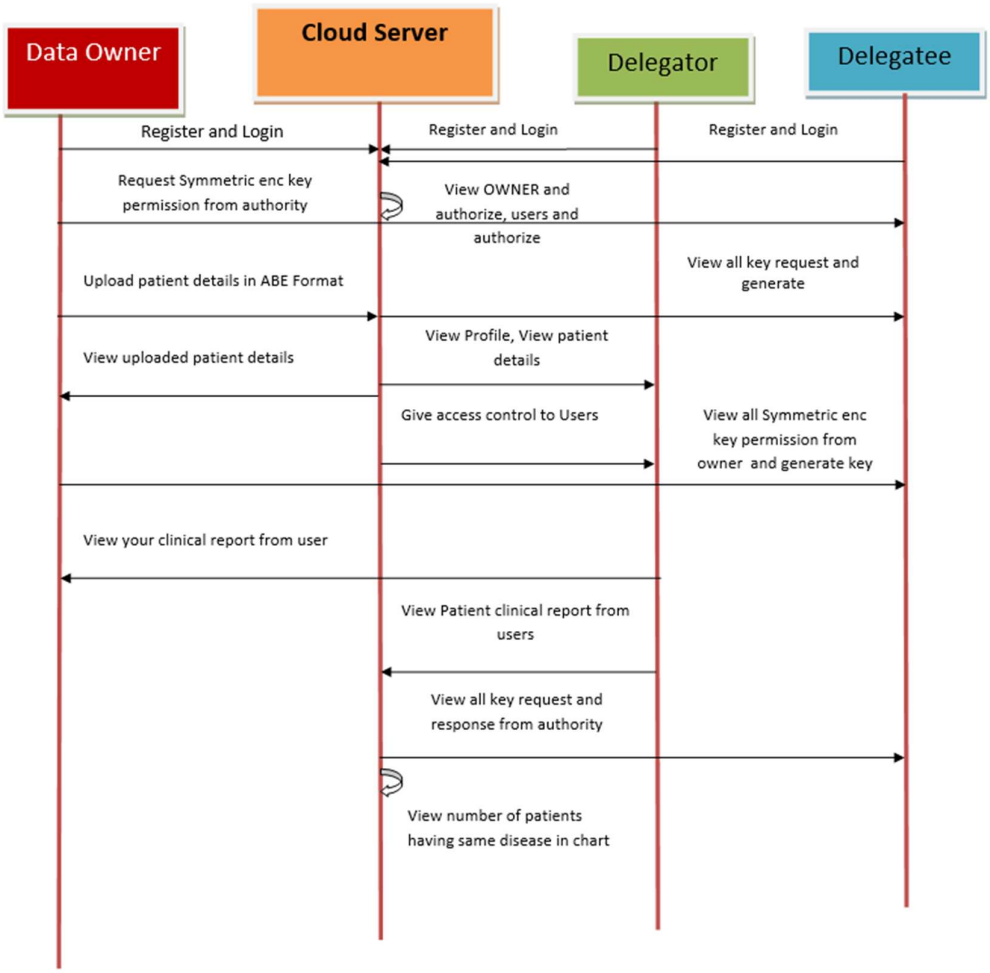


Fig.No:3.5.3 SEQUENCE DIAGRAM:

CHAPTER 4

IMPLEMENTATION

Data Owner

In this module, the provider requests for symmetric encryption key permission from OWNER and upload the patient details in ABE with the key. View & delete the uploaded patient details, and view the clinical report from the user.

Delegator

In this module, Delegator register and logs in and request access control from the healthcare server and view the access control (1-access only the patient details and 2-accessing both patient details with the document), if the user has both the access permissions, user can provide the clinical report for the corresponding patient details.

CLOUD SERVER

The Cloud Server authorizes both user and owner, view all the uploaded patient details and give the access control permissions to the corresponding requested user. View the response from the OWNER about the key requested. After the clinical report is generated by the user forward it to the corresponding patient. And view the patient disease in chart.

DELEGATEE

In this module, the Delegatee will generate the key requested by User. And also generates the symmetric encryption key and provides permission requested by the users.

CHAPTER-5

RESULTS

Fig.No:5.1 Home Page

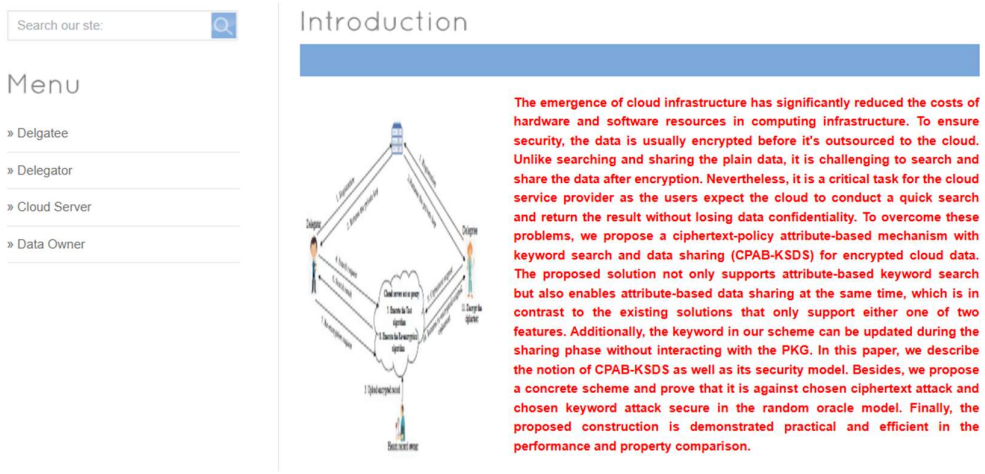


Fig.No:5.2 Introduction

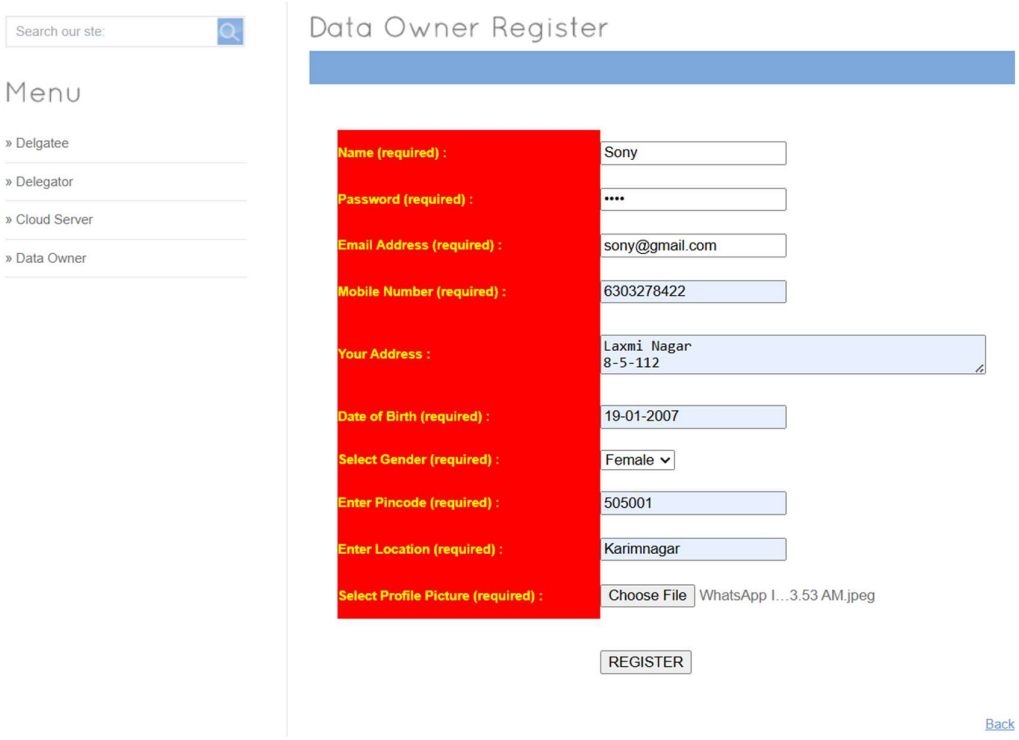


Fig.No:5.3 Data Owner Register

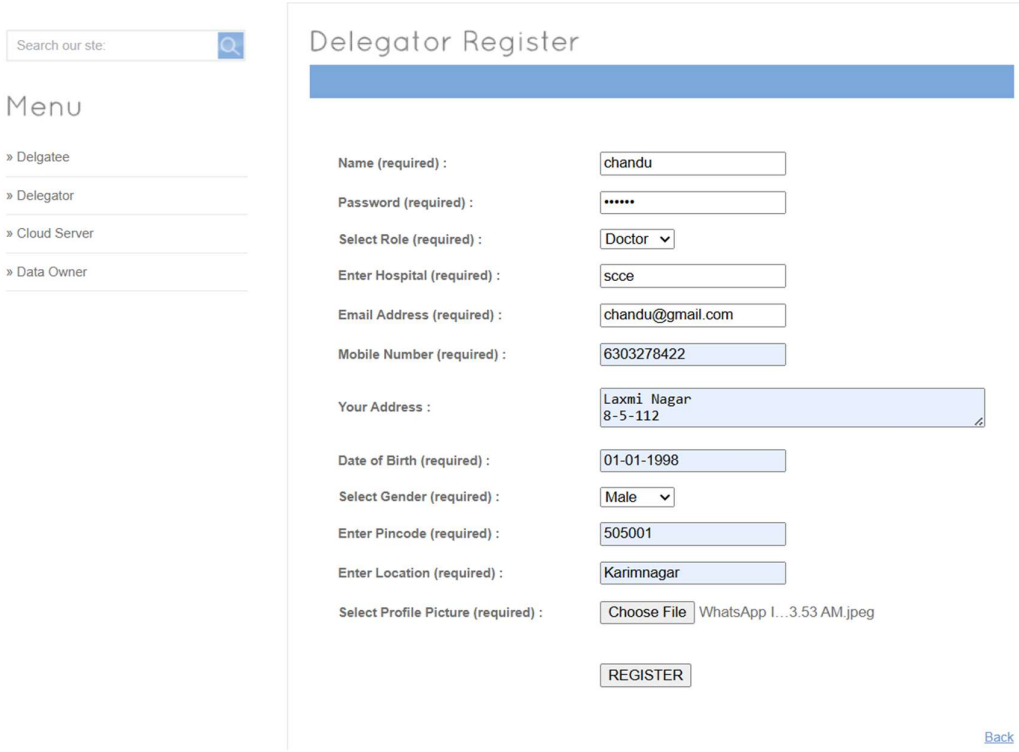


Fig.No:.4 Delegatoor Register

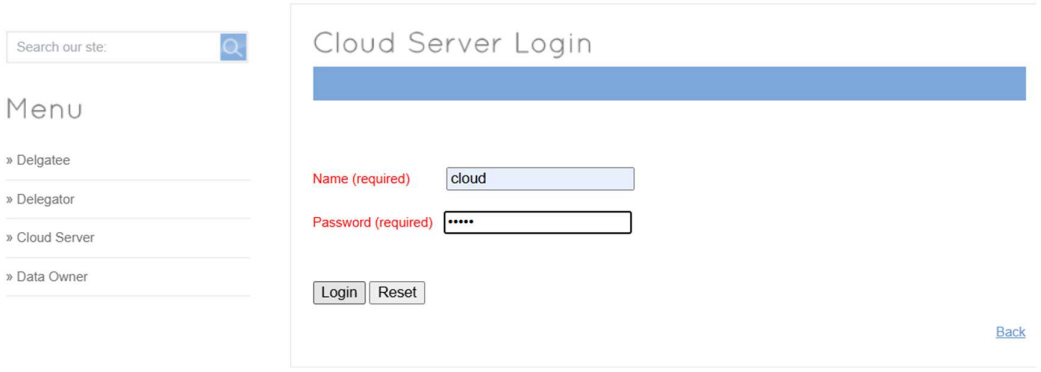


Fig.No:5.5 Cloud

Fig.No:5.6 Authorize Delegatoor

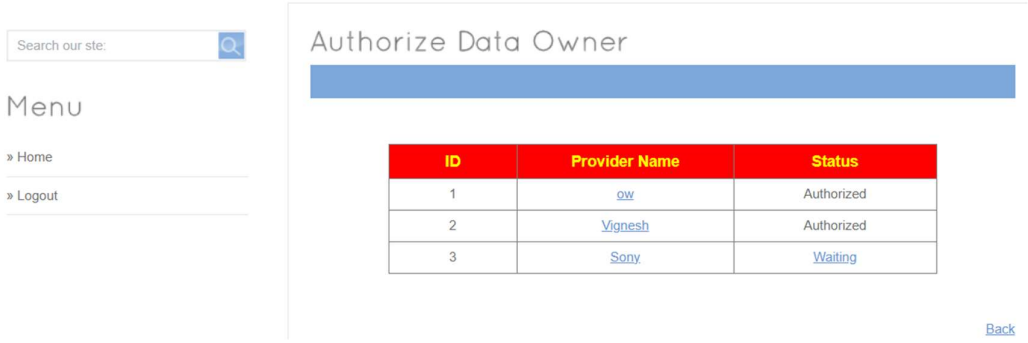
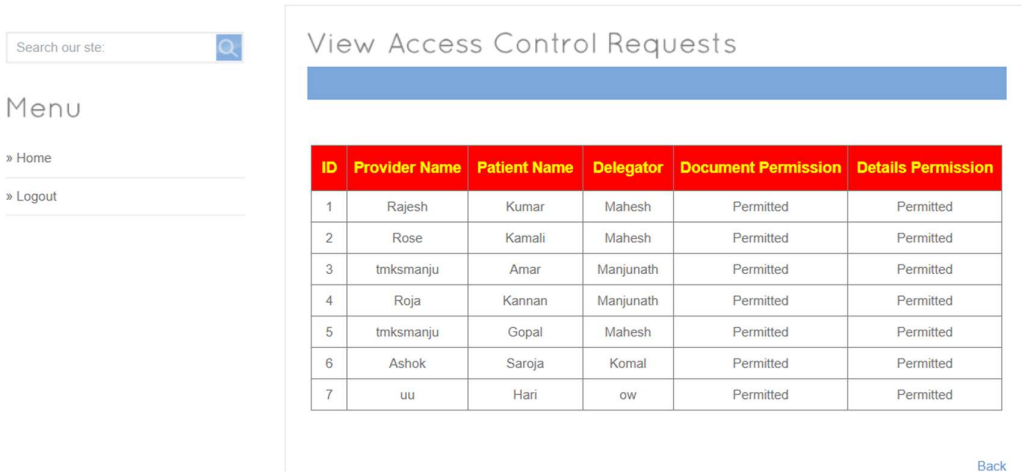
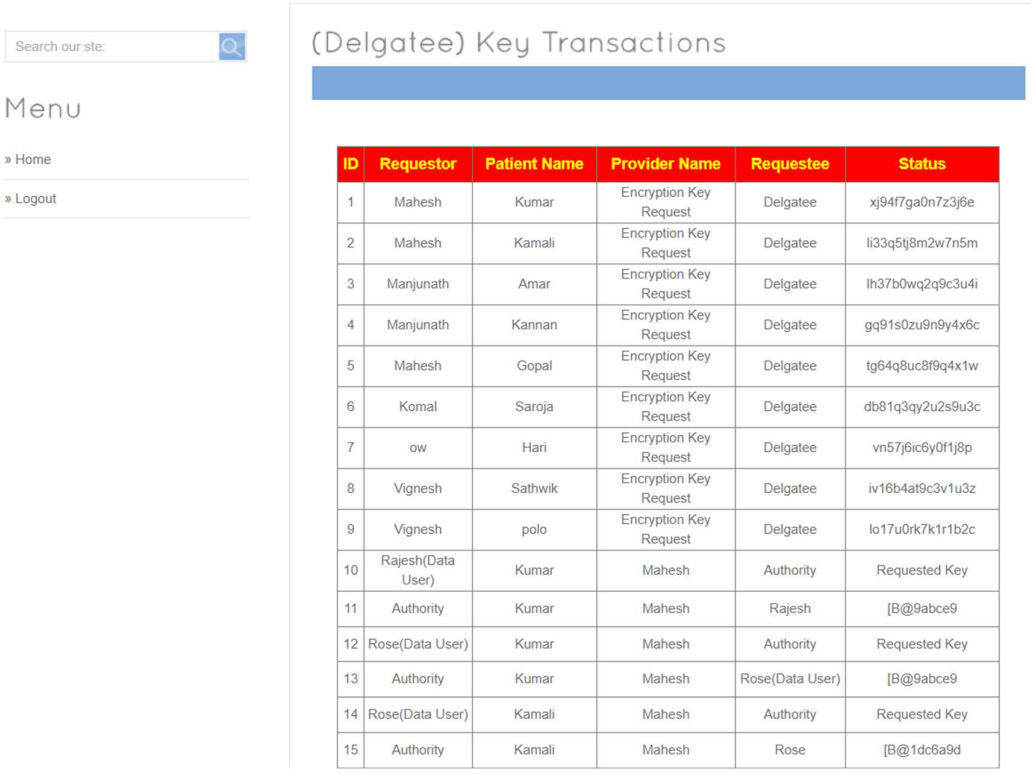


Fig.No:5.7 Authorize Data Onwer



ID	Provider Name	Patient Name	Delegator	Document Permission	Details Permission
1	Rajesh	Kumar	Mahesh	Permitted	Permitted
2	Rose	Kamali	Mahesh	Permitted	Permitted
3	tmksmanju	Amar	Manjunath	Permitted	Permitted
4	Roja	Kannan	Manjunath	Permitted	Permitted
5	tmksmanju	Gopal	Mahesh	Permitted	Permitted
6	Ashok	Saroja	Komal	Permitted	Permitted
7	uu	Hari	ow	Permitted	Permitted

Fig.No:5.8 View Access Control Request



ID	Requestor	Patient Name	Provider Name	Requestee	Status
1	Mahesh	Kumar	Encryption Key Request	Delgatee	xj94f7ga0n7z3j6e
2	Mahesh	Kamali	Encryption Key Request	Delgatee	li33q5tj8m2w7n5m
3	Manjunath	Amar	Encryption Key Request	Delgatee	lh37b0wq2q9c3u4i
4	Manjunath	Kannan	Encryption Key Request	Delgatee	gq91s0zu9n9y4x6c
5	Mahesh	Gopal	Encryption Key Request	Delgatee	tg64q8uc8f9q4x1w
6	Komal	Saroja	Encryption Key Request	Delgatee	db81q3qy2u2s9u3c
7	ow	Hari	Encryption Key Request	Delgatee	vn57j6ic6y0f1j8p
8	Vignesh	Sathwik	Encryption Key Request	Delgatee	iv16b4at9c3v1u3z
9	Vignesh	polo	Encryption Key Request	Delgatee	lo17u0rk7k1r1b2c
10	Rajesh(Data User)	Kumar	Mahesh	Authority	Requested Key
11	Authority	Kumar	Mahesh	Rajesh	[B@9abce9
12	Rose(Data User)	Kumar	Mahesh	Authority	Requested Key
13	Authority	Kumar	Mahesh	Rose(Data User)	[B@9abce9
14	Rose(Data User)	Kamali	Mahesh	Authority	Requested Key
15	Authority	Kamali	Mahesh	Rose	[B@1dc6a9d

Fig.No:5.9 (Delegatee) Key Transactions

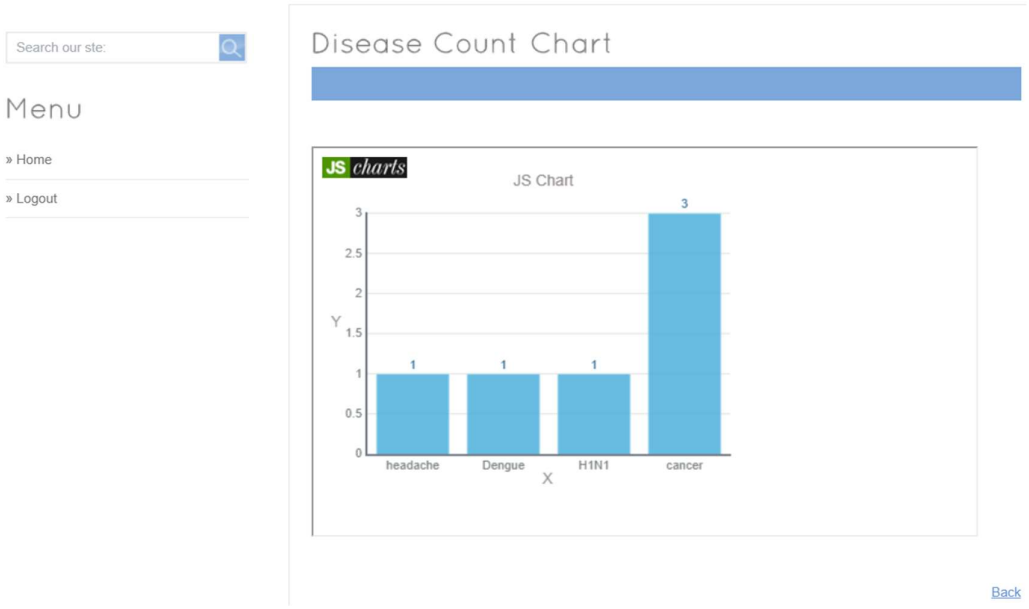


Fig.No:5.10 Disease Count Chart

Fig.No:5.11 Add Patient

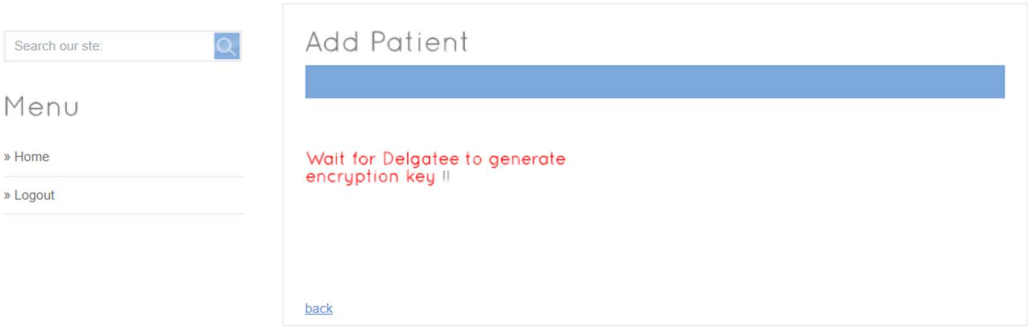


Fig.No:5.11.1 Add Patient

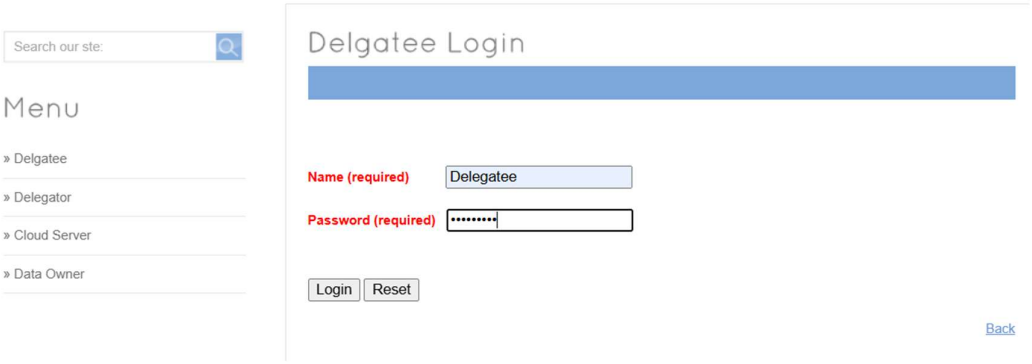


Fig.No:5.12 Delegatee Login

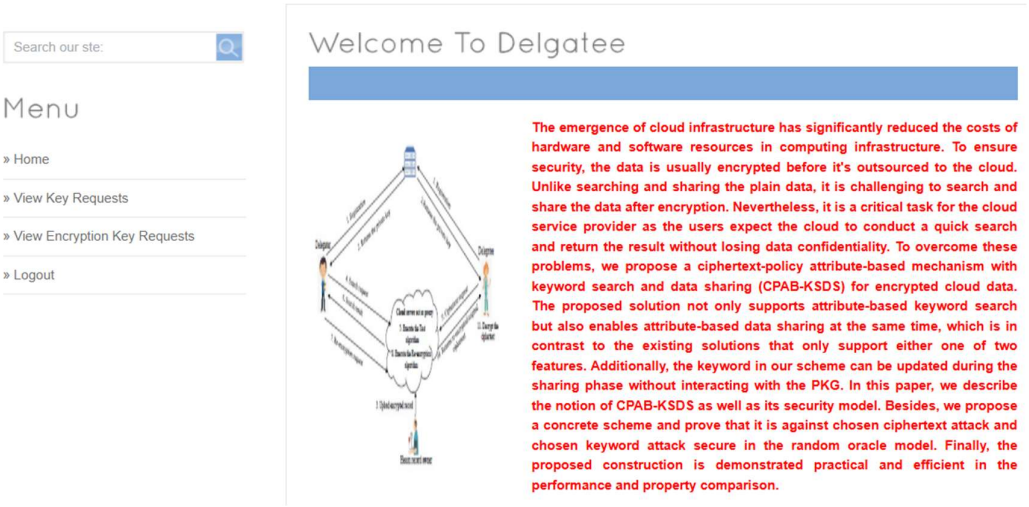


Fig.No:5.13 Welcome To Delegatee

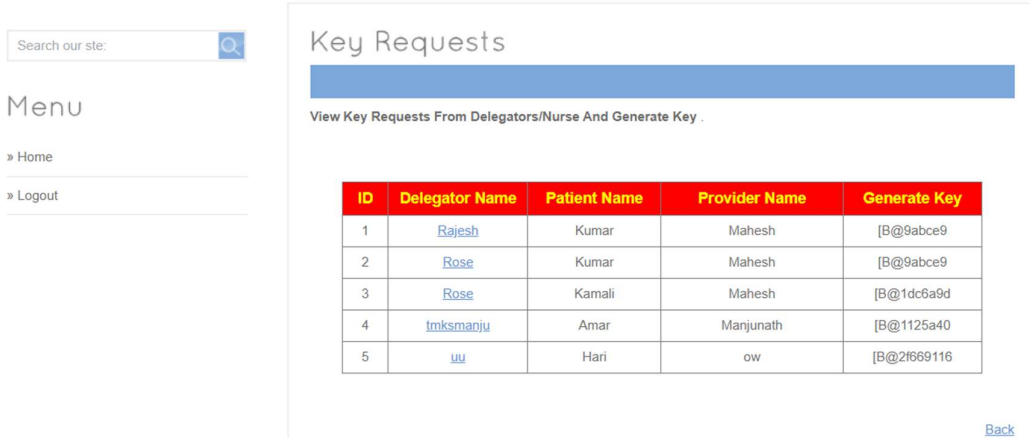


Fig.No:5.14 Key Request

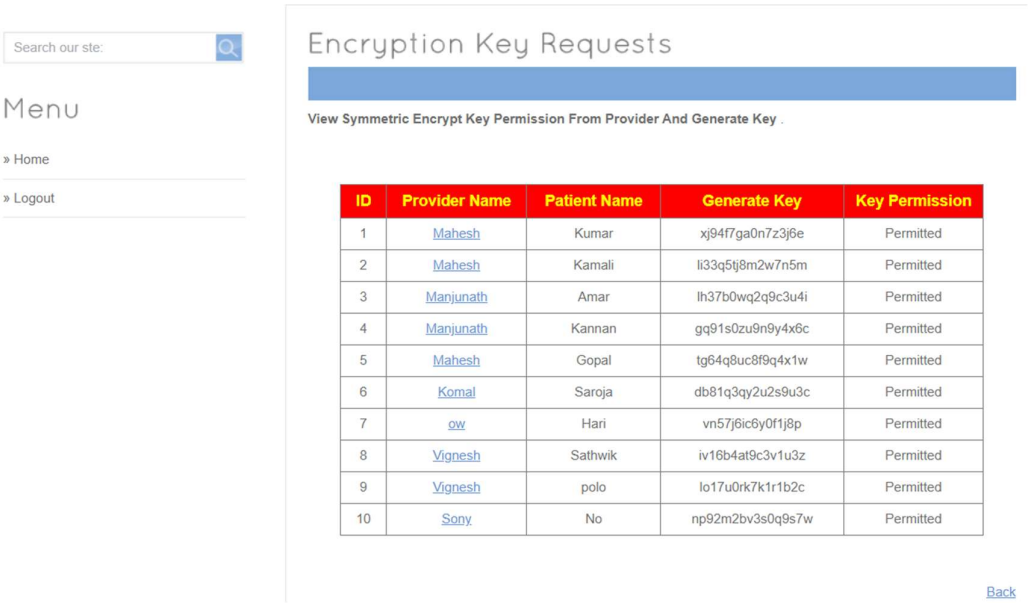


Fig.No:5.15 Encryption Key Request

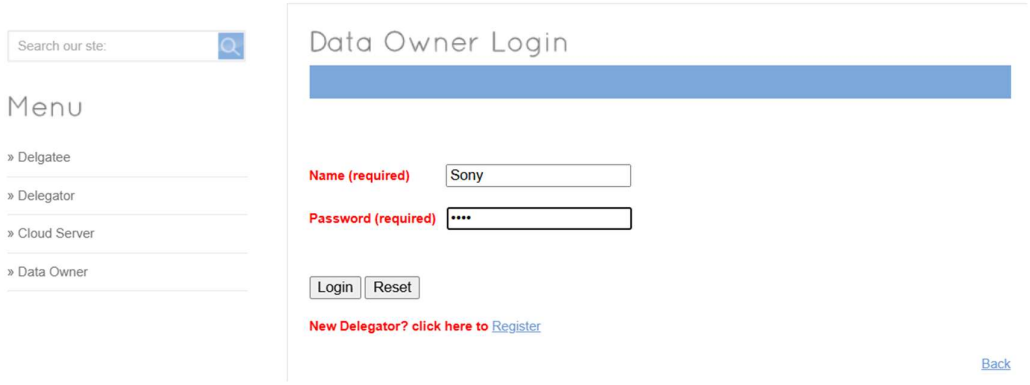


Fig.No:5.16 Data Owner Login

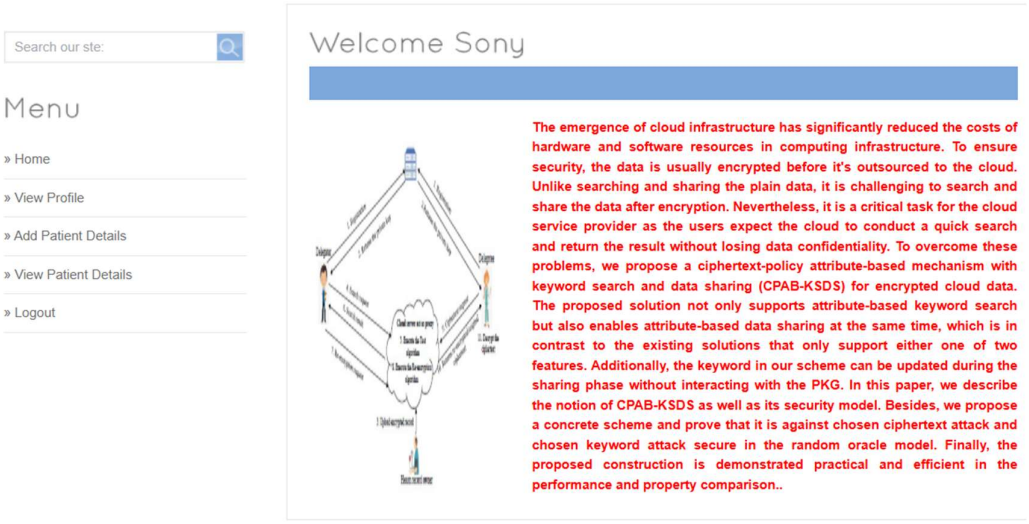


Fig.No:5.17 Welcome

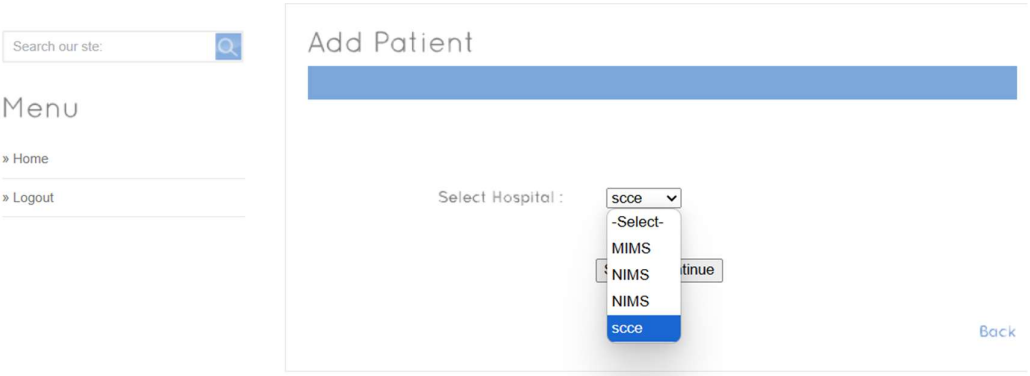


Fig.No:5.18 Add Patient

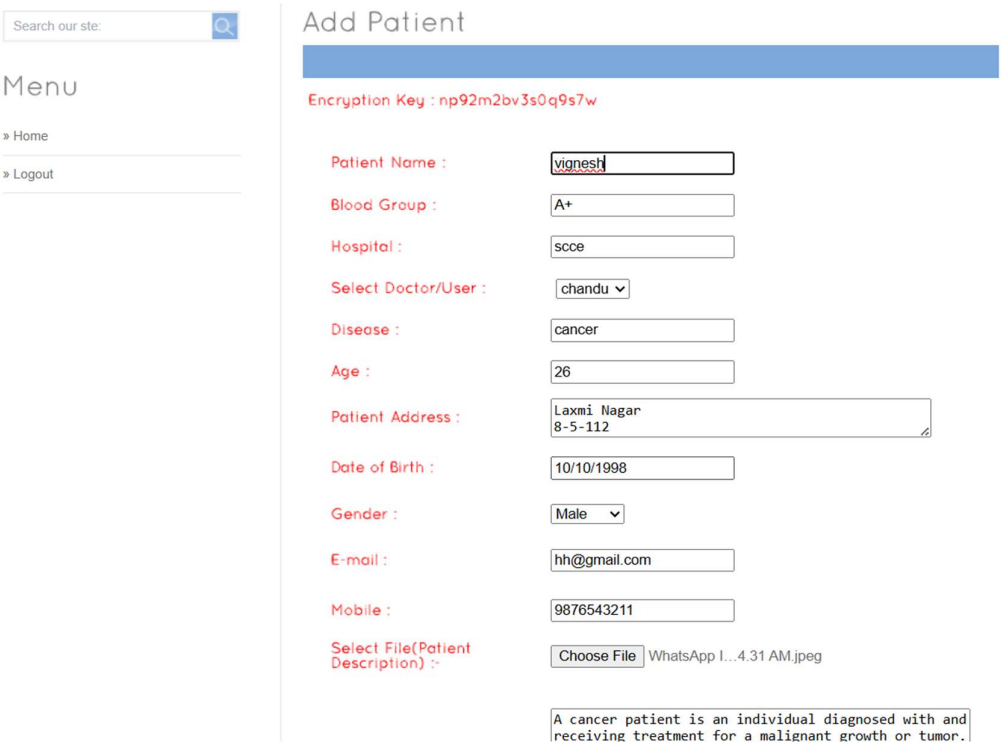


Fig.No:5.18.1 Add Patient

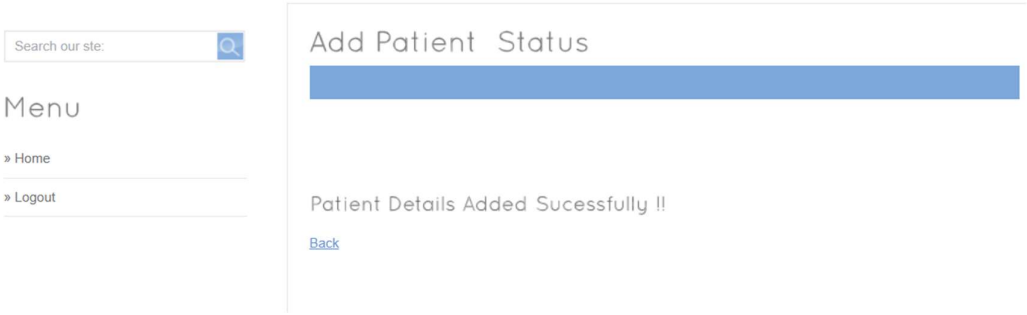


Fig.No:5.19 Add Patient Status

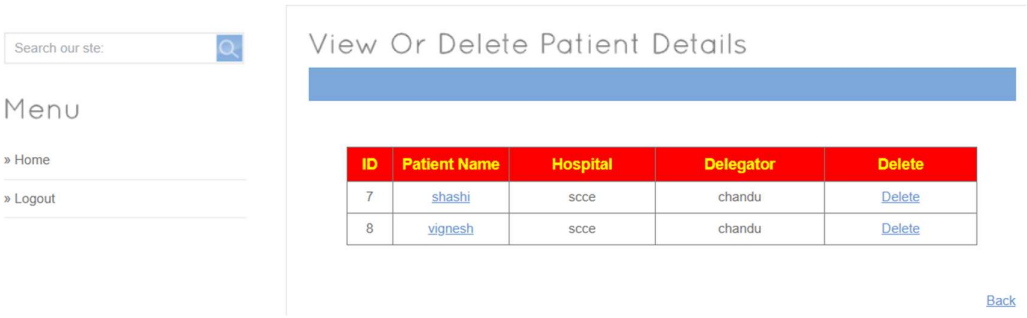


Fig.No:5.20 View Or Delete Patient Details

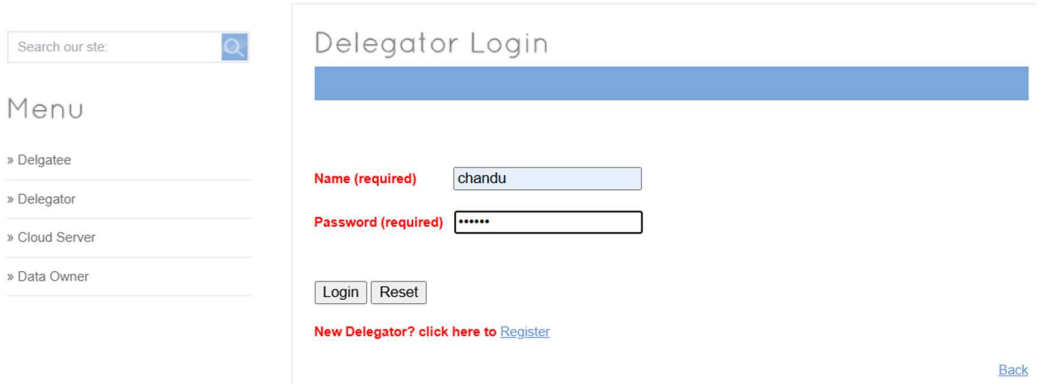


Fig.No:5.21 Delegatoor Login

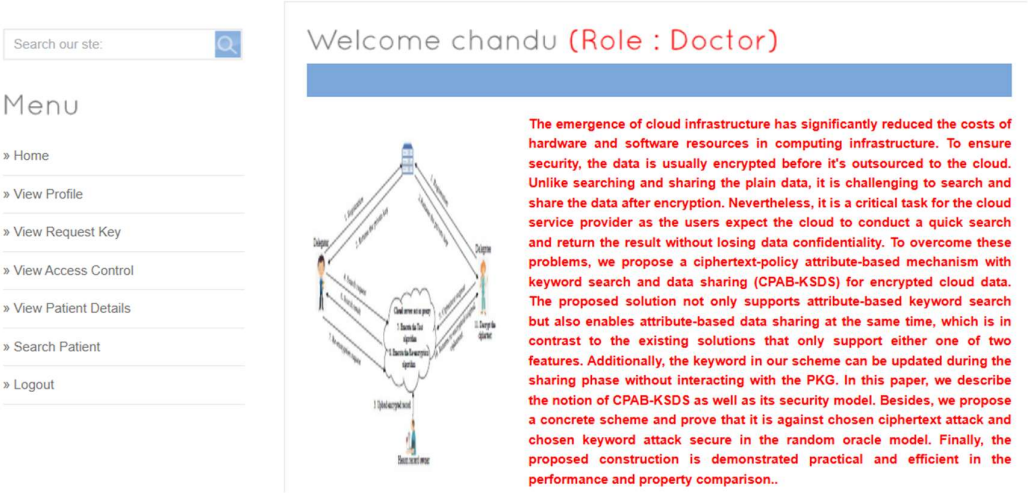


Fig.No:5.22 Welcome Chandu

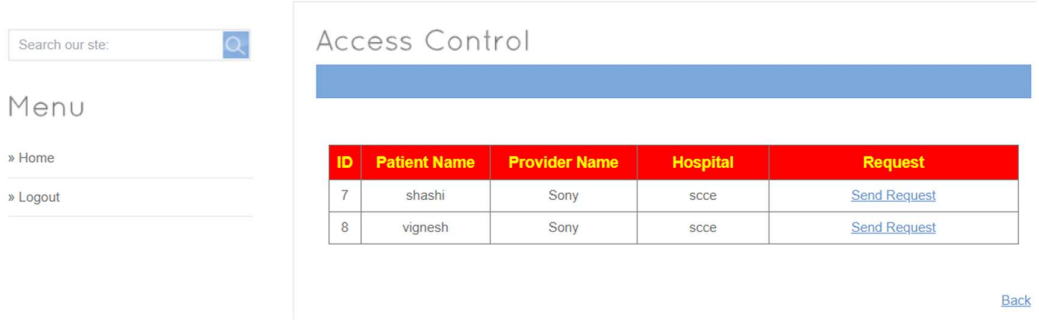


Fig.No:5.23 Access Control

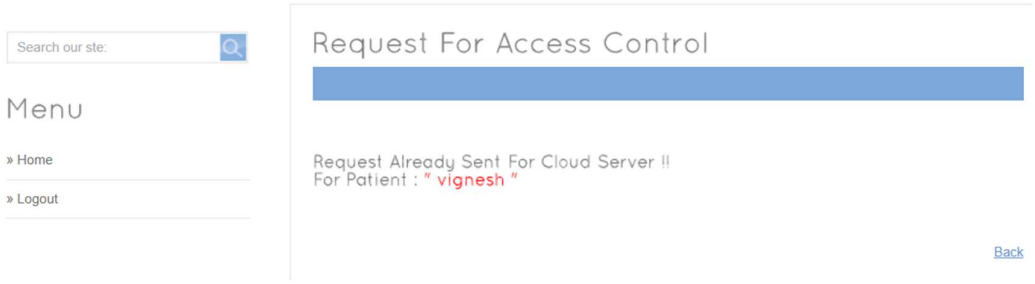


Fig.No:5.24 Request For Access Control

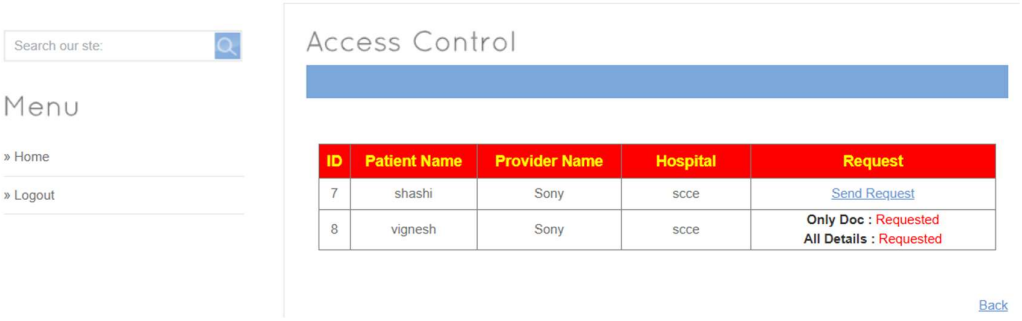


Fig.No:5.25 Access Control

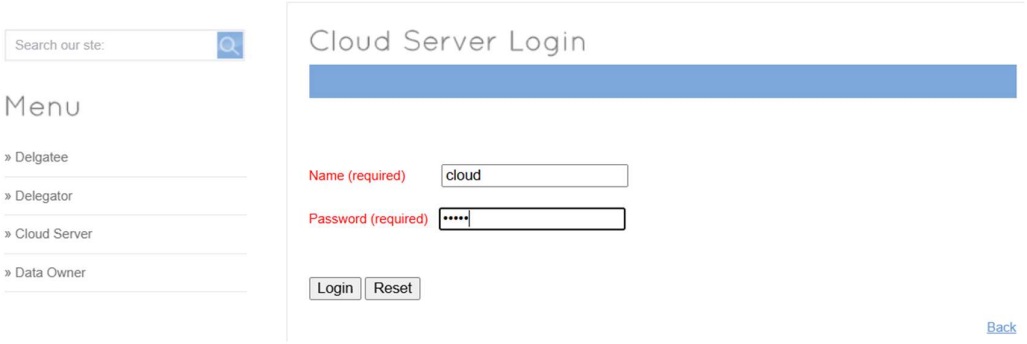


Fig.No:5.26 Cloud Server Login

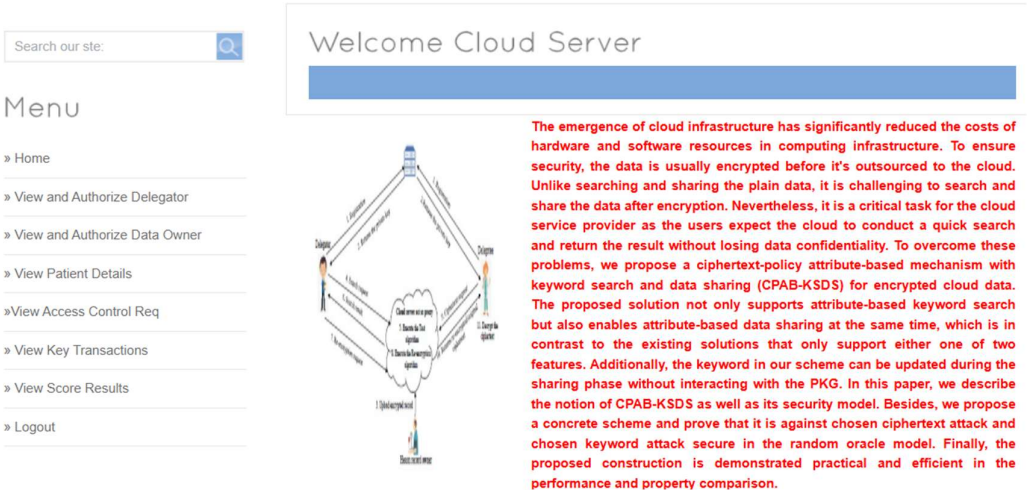


Fig.No:5.27 Welocome Cloud Server

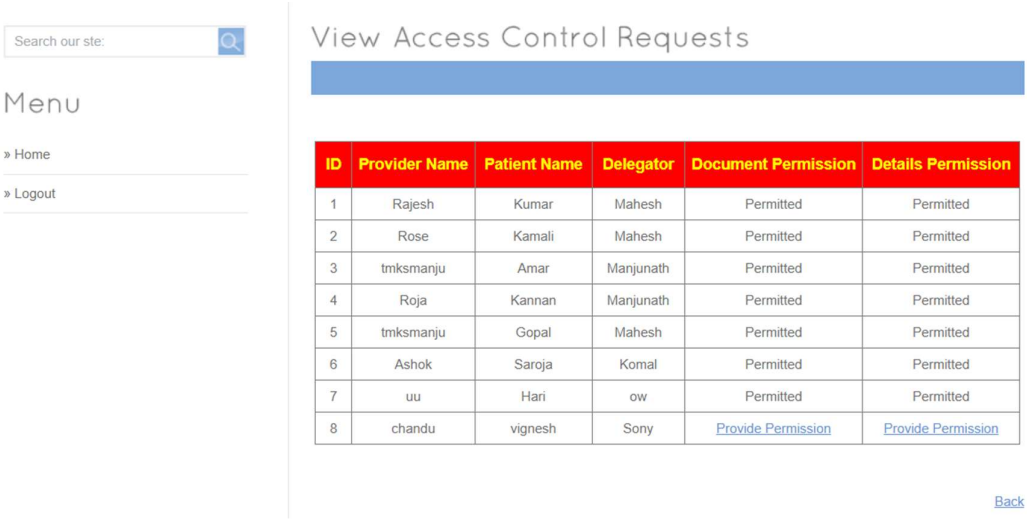


Fig.No:5.28 View Access Coontrol Request

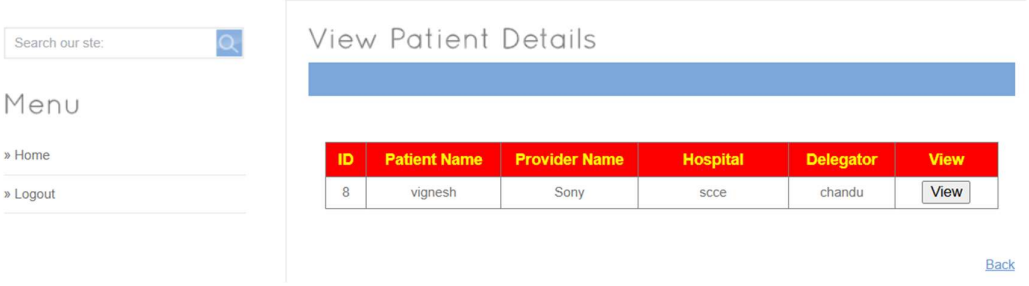


Fig.No:5.29 View Patient Details

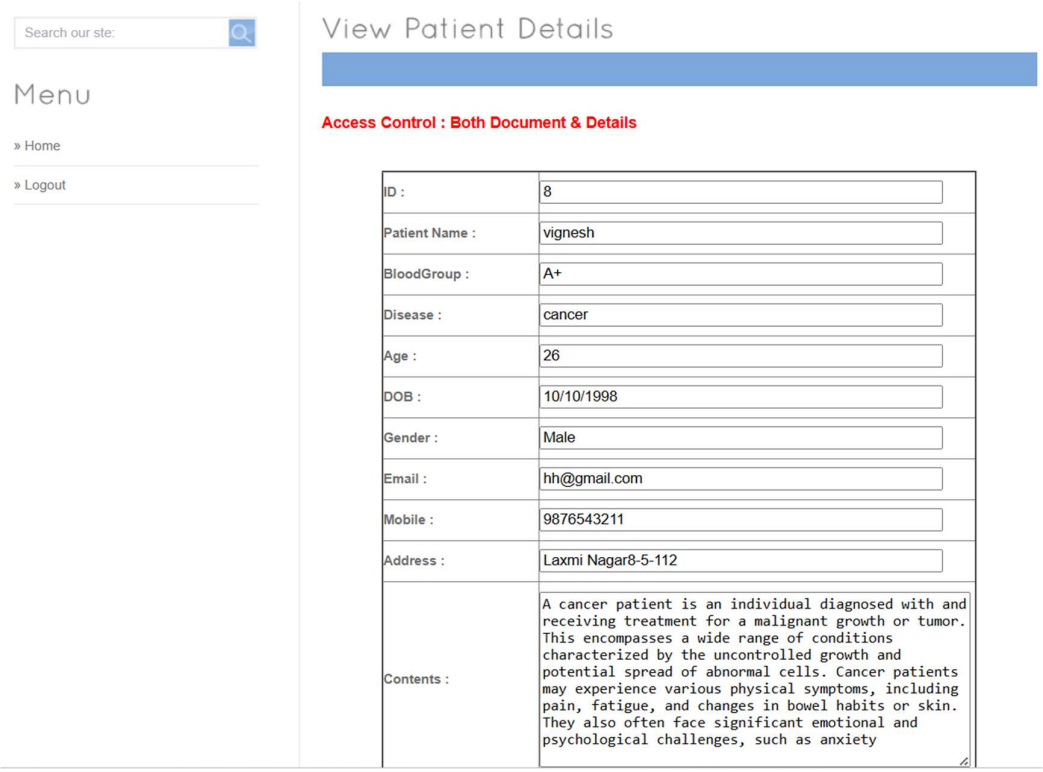


Fig.No:5.29.1 View Patient Details

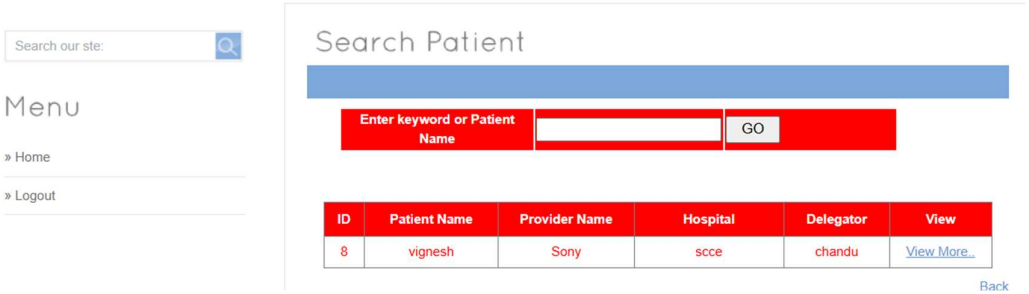


Fig.No:5.30 Search Patient

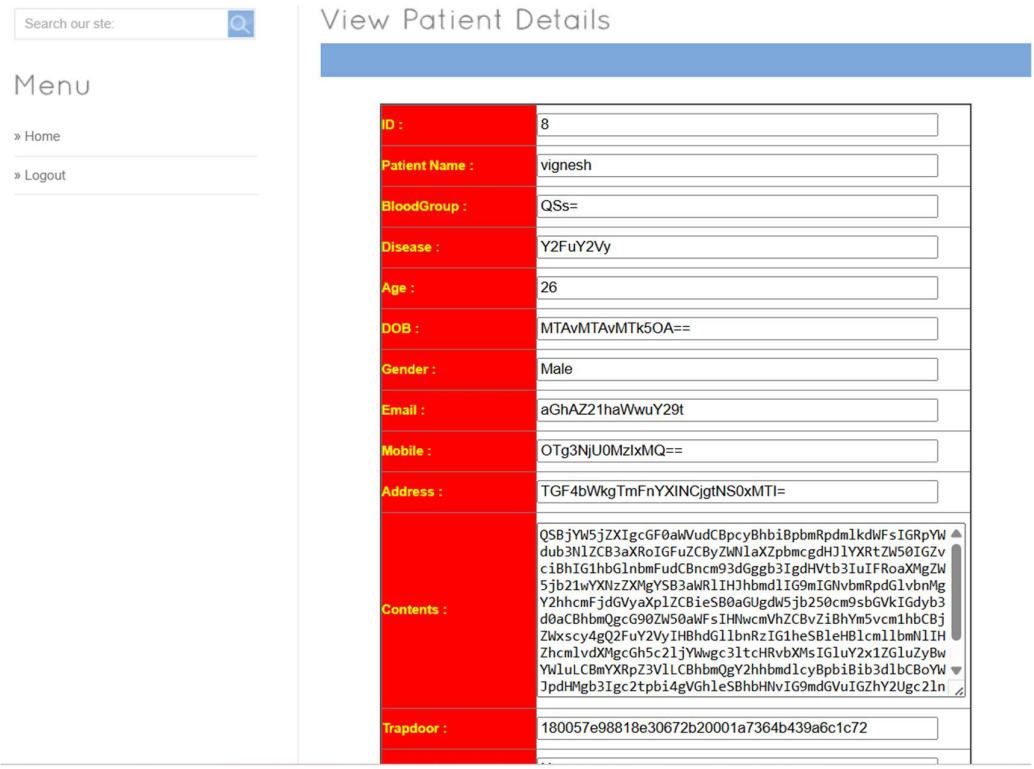


Fig.No:5.31 View Patient Details

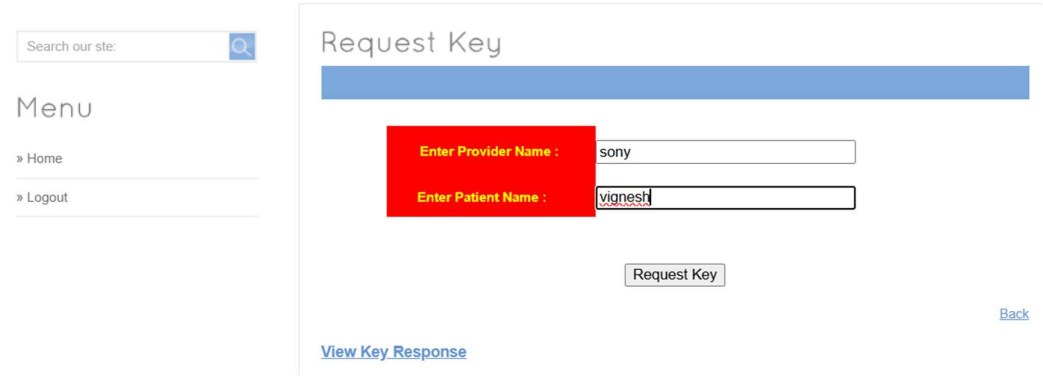


Fig.No:5.32 Request Key

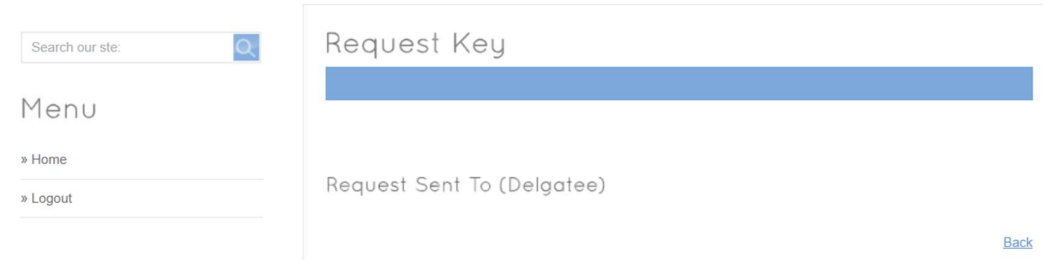


Fig.No:5.32.1 Request Key

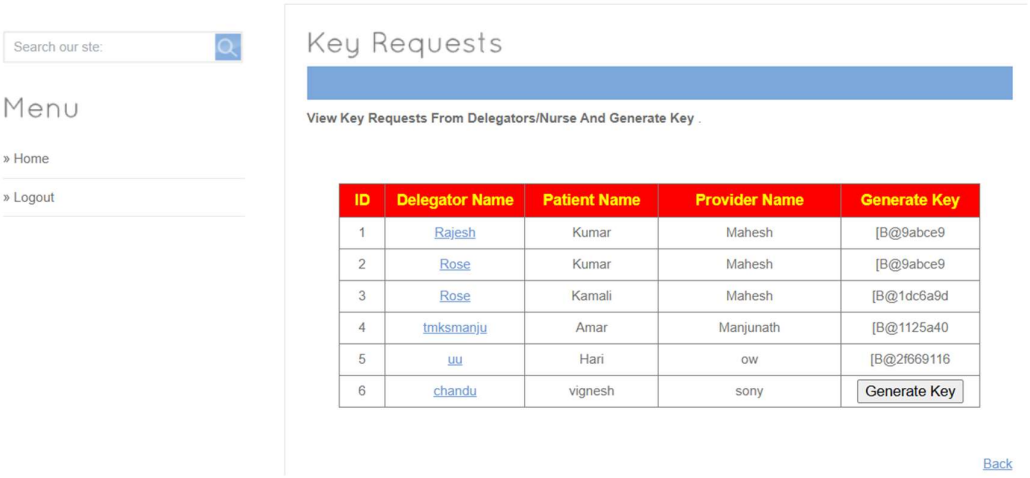


Fig.No:5.33 Key Requests

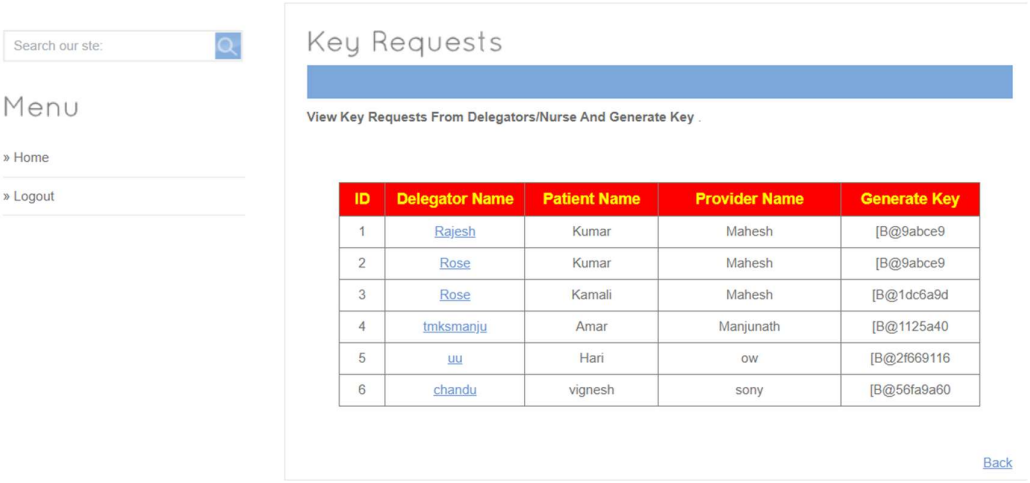


Fig.No:5.33.1 Key Request

CHAPTER 6

TESTING

6.1 SYSTEM TESTING

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

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

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

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 satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

6.2.3: Functional test

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

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

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

6.2.4: System Test

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

6.2.5: White Box Testing

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

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.

6.3: TESTING METHODOLOGIES

The following are the Testing Methodologies:

- Unit Testing.
- Integration Testing.
- User Acceptance Testing.
- Output Testing.
- Validation Testing.

Unit Testing

Unit testing focuses verification effort on the smallest unit of Software design that is the module. Unit testing exercises specific paths in a module's control structure to ensure complete coverage and maximum error detection. This test focuses on each module individually, ensuring that it functions properly as a unit. Hence, the naming is Unit Testing.

During this testing, each module is tested individually and the module interfaces are verified for the consistency with design specification. All important processing path are tested for the expected results. All error handling paths are also tested.

Integration Testing

Integration testing addresses the issues associated with the dual problems of verification and program construction. After the software has been integrated a set of high order tests are conducted. The main objective in this testing process is to take unit tested modules and builds a program structure that has been dictated by design.

The following are the types of Integration Testing:

1. **Top Down Integration**

This method is an incremental approach to the construction of program structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main program

module. The module subordinates to the main program module are incorporated into the structure in either a depth first or breadth first manner.

In this method, the software is tested from main module and individual stubs are replaced when the test proceeds downwards.

2. Bottom-up Integration

This method begins the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated from the bottom up, processing required for modules subordinate to a given level is always available and the need for stubs is eliminated. The bottom up integration strategy may be implemented with the following steps:

- The low-level modules are combined into clusters into clusters that perform a specific Software sub-function.
- A driver (i.e.) the control program for testing is written to coordinate test case input and output.
- The cluster is tested.
- Drivers are removed and clusters are combined moving upward in the program structure

The bottom up approaches tests each module individually and then each module is module is integrated with a main module and tested for functionality.

OTHER TESTING METHODOLOGIES

User Acceptance Testing

User Acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required.

The system developed provides a friendly user interface that can easily be understood even by a person who is new to the system.

Output Testing

After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in the specified format. Asking the users about the format required by them tests the outputs generated or displayed by the system under consideration. Hence the output format is considered in 2 ways – one is on screen and another in printed format.

Validation Checking

Validation checks are performed on the following fields.

Text Field:

The text field can contain only the number of characters lesser than or equal to its size. The text fields are alphanumeric in some tables and alphabetic in other tables. Incorrect entry always flashes and error message.

Numeric Field:

The numeric field can contain only numbers from 0 to 9. An entry of any character flashes an error messages. The individual modules are checked for accuracy and what it has to perform. Each module is subjected to test run along with sample data. The individually tested modules are integrated into a single system. Testing involves executing the real data information is used in the program the existence of any program defect is inferred from the output. The testing should be planned so that all the requirements are individually tested.

A successful test is one that gives out the defects for the inappropriate data and produces and output revealing the errors in the system.

Preparation of Test Data

Taking various kinds of test data does the above testing. Preparation of test data plays a vital role in the system testing. After preparing the test data the system under study is tested using that test data. While testing the system by using test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

Using Live Test Data:

Live test data are those that are actually extracted from organization files. After a system is partially constructed, programmers or analysts often ask users to key in a set of data from their normal activities. Then, the systems person uses this data as a way to partially test the system. In other instances, programmers or analysts extract a set of live data from the files and have them entered themselves.

It is difficult to obtain live data in sufficient amounts to conduct extensive testing. And, although it is realistic data that will show how the system will perform for the typical processing requirement, assuming that the live data entered are in fact typical, such data generally will not test all combinations or formats that can enter the system. This bias toward typical values then does not provide a true systems test and in fact ignores the cases most likely to cause system failure.

Using Artificial Test Data:

Artificial test data are created solely for test purposes, since they can be generated to test all combinations of formats and values. In other words, the artificial data, which can quickly be prepared by a data generating utility program in the information systems department, make possible the testing of all login and control paths through the program.

The most effective test programs use artificial test data generated by persons other than those who wrote the programs. Often, an

independent team of testers formulates a testing plan, using the systems specifications.

The package "Virtual Private Network" has satisfied all the requirements specified as per software requirement specification and was accepted.

USER TRAINING

Whenever a new system is developed, user training is required to educate them about the working of the system so that it can be put to efficient use by those for whom the system has been primarily designed. For this purpose the normal working of the project was demonstrated to the prospective users. Its working is easily understandable and since the expected users are people who have good knowledge of computers, the use of this system is very easy.

MAINTAINENCE

This covers a wide range of activities including correcting code and design errors. To reduce the need for maintenance in the long run, we have more accurately defined the user's requirements during the process of system development. Depending on the requirements, this system has been developed to satisfy the needs to the largest possible extent. With development in technology, it may be possible to add many more features based on the requirements in future. The coding and designing is simple and easy to understand which will make maintenance easier.

TESTING STRATEGY :

A strategy for system testing integrates system test cases and design techniques into a well planned series of steps that results in the successful construction of software. The testing strategy must co-operate test planning, test case design, test execution, and the resultant data collection and evaluation. A strategy for software testing must

accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high level tests that validate major system functions against user requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing represents an interesting anomaly for the software. Thus, a series of testing are performed for the proposed system before the system is ready for user acceptance testing.

CHAPTER 7

CONCLUSION

In this project new notion of cipher text-policy attribute- based mechanism (CPAB-KSDS) is introduced to support keyword searching and data sharing. A concrete CPAB-KSDS scheme has been constructed in this project and we prove its CCA security in the random oracle model. The proposed scheme is demonstrated efficient and practical in the performance and property comparison. This project provides an affirmative answer to the open challenging problem pointed 96 out in the prior work, which is to design an attribute based encryption with keyword searching and data sharing without the PKG during the sharing phase. Furthermore, our work motivates interesting open problems as well including designing CPAB-KSDS scheme without random oracles or proposing a new scheme to support more expressive keyword search.

REFERENCES

[1] A. Sahai and B. Waters, “Fuzzy identity-based encryption,” in Annual
International Conference on the Theory and Applications of Crypto982 graphic Techniques, pp. 457–473, Springer, 2005.

[2] V. Goyal, O. Pandey, A. Sahai, and B. Waters, “Attribute-based encryp984
tion for fine-grained access control of encrypted data,” in Proceedings of the 13th ACM conference on Computer and communications security,
pp. 89–98, Acm, 2006.

[3] J. Bethencourt, A. Sahai, and B. Waters, “Ciphertext-policy attribute988
based encryption,” in Security and Privacy, 2007. SP’07. IEEE Sympo989
sium on, pp. 321–334, IEEE, 2007.

[4] B. Waters, “Ciphertext-policy attribute-based encryption: An expressive,
efficient, and provably secure realization,” in International Workshop on
Public Key Cryptography, pp. 53–70, Springer, 2011.

[5] H. Qian, J. Li, Y. Zhang, and J. Han, “Privacy-preserving personal health
record using multi-authority attribute-based encryption with revocation,”
International Journal of Information Security, vol. 14, no. 6, pp. 487–96 497, 2015.

[6] J. Liu, X. Huang, and J. K. Liu, “Secure sharing of personal health records in cloud computing: Ciphertext-policy attribute-based signcryp999
tion,” Future Generation Computer Systems, vol. 52, pp. 67–76, 2015.

[7] L. Fang, W. Susilo, C. Ge, and J. Wang, “Interactive conditional proxy
re-encryption with fine grain policy,” Journal of Systems and Software, vol. 84, no. 12, pp. 2293–2302, 2011.
[8] K. Emura, A. Miyaji, A. Nomura, K. Omote, and M. Soshi, “A
ciphertext-policy attribute-based encryption scheme with constant ci1005
phertext length,” in International Conference on Information Security
Practice and Experience, pp. 13–23, Springer, 2009.
[9] S. Hohenberger and B. Waters, “Attribute-based encryption with fast
decryption,” in Public-Key Cryptography–PKC 2013, pp. 162–179,
Springer, 2013.
[10] A. Lewko and B. Waters, “New proof methods for attribute-based
encryption: Achieving full security through selective techniques,” in
Advances in Cryptology–CRYPTO 2012, pp. 180–198, Springer, 2012.
[11] M. Li, S. Yu, Y. Zheng, K. Ren, and W. Lou, “Scalable and secure
sharing of personal health records in cloud computing using attribute1015
based encryption,” IEEE transactions on parallel and distributed sys1016
tems, vol. 24, no. 1, pp. 131–143, 2012.
[12] L. Zhang, G. Hu, Y. Mu, and F. Rezaeibagha, “Hidden ciphertext policy
attribute-based encryption with fast decryption for personal health record
system,” IEEE Access, vol. 7, pp. 33202–33213, 2019.
[13] M. Green, S. Hohenberger, B.Waters, et al., “Outsourcing the decryption
of abe ciphertexts.,” in USENIX Security Symposium, vol. 2011, 2011.
[14] J. Lai, R. H. Deng, C. Guan, and J. Weng, “Attribute-based encryption

with verifiable outsourced decryption," IEEE Transactions on informa1024
tion forensics and security, vol. 8, no. 8, pp. 1343–1354, 2013.
[15] J. Li, X. Huang, J. Li, X. Chen, and Y. Xiang, "Securely outsourcing attribute-based encryption with checkability," IEEE Transactions on Parallel and Distributed Systems, vol. 25, no. 8, pp. 2201–2210, 2013.
[16] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic 1028
proxy cryptography," in International Conference on the Theory and 1029
Applications of Cryptographic Techniques, pp. 127–144, Springer, 1998. 1030
[17] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy 1031
re-encryption schemes with applications to secure distributed storage," 1032
ACM Transactions on Information and System Security (TISSEC), vol. 9, 1033
no. 1, pp. 1–30, 2006. 1034
[18] B. Libert and D. Vergnaud, "Unidirectional chosen-ciphertext secure 1035
proxy re-encryption," IEEE Transactions on Information Theory, vol. 57, 1036
no. 3, pp. 1786–1802, 2011. 1037
[19] M. Green and G. Ateniese, "Identity-based proxy re-encryption," in 1038
Applied Cryptography and Network Security, pp. 288–306, Springer, 1039
2007. 1040
[20] C. Ge, W. Susilo, J. Wang, and L. Fang, "Identity-based conditional 1041

proxy re-encryption with fine grain policy," Computer Standards & 1042
Interfaces, vol. 52, pp. 1–9, 2017. 1043
[21] X. Liang, Z. Cao, H. Lin, and J. Shao, "Attribute based proxy re- 1044
encryption with delegating capabilities," in Proceedings of the 4th In- 1045
ternational Symposium on Information, Computer, and Communications 1046
Security, pp. 276–286, ACM, 2009. 1047
[22] S. Luo, J. Hu, and Z. Chen, "Ciphertext policy attribute-based proxy 1048
re-encryption," in International Conference on Information and Com- 1049
munications Security, pp. 401–415, Springer, 2010. 1050
[23] K. Liang, L. Fang, W. Susilo, and D. S. Wong, "A ciphertext-policy 1051
attribute-based proxy re-encryption with chosen-ciphertext security," in 1052
Intelligent Networking and Collaborative Systems (INCoS), 2013 5th 1053
International Conference on, pp. 552–559, IEEE, 2013.