

A FAST NEAREST NEIGHBOR SEARCH SCHEME OVER OUTSOURSED ENCRYPTED MEDICAL IMAGES

*A Project report submitted to Jawaharlal Nehru
Technological University, Kakinada, In the partial
fulfilment for the award of the Degree in
BACHELOR OF TECHNOLOGY*

**IN
COMPUTER SCIENCE & ENGINEERING
(ARTIFICIAL INTELIGENCE)**

Submitted by

KUNCHALA POOJA (20F91A4330)

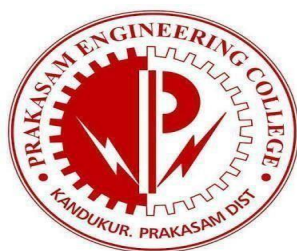
MANNEM NANDINI (20F914332)

THATIKONDA PRATHYUSHA (20F91A4361)

VAKA MANEESHA (20F91A4362)

Under the Noble Guidance of

N.MADHAVA RAO M.Tech



PRAKASAM ENGINEERING COLLEGE

(An ISO 9001-2008 & NAAC Accredited Institution)

(Affiliated to Jawaharlal Nehru Technological University, Kakinada)

O.V.ROAD, KANDUKUR-523105, A.P.

A.Y:2020-2024

PRAKASAM ENGINEERING COLLEGE

(An ISO 9001-2008 & NAAC Accredited Institution) (Affiliated to Jawaharlal Nehru Technological University, Kakinada) **O.V.ROAD, KANDUKUR-523105, A.P.**



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that the project report entitled “**A FAST NEAREST NEIGHBOR SEARCH SCHEME OVER OUTSOURCED ENCRYPTED MEDICAL IMAGES**” is a bonafide work **KUNCHALA POOJA(20F91A4330), MANNEM NANDINI(20F91A4332), THATIKONDA PRATHYUSHA(20F91A4361), VAKA MANEESHA (20F91A4362)**, in the partial fulfilment of the requirement for the award of the degree in Bachelor of **Technology in COMPUTER SCIENCE & ENGINEERING (ARTIFICIAL INTELLIGENCE)** for the academic year 2023-2024. This work is done under my supervision.

Signature of the Guide

Mr. N. MADHAVARAO

M.Tech

Signature of the HOD

Mr. K. SUBBA REDDY

M.Tech(Ph.D)

Signature of External Examiner

DECLARATION

We do hereby declare that the project report -1entitled“**A FAST NEAREST NEIGHBOR SEARCH SCHEME OVER OUTSOURCED ENCRYPTED MEDICAL IMAGES**” is a genuine work carried out by us under the guidance of **Mr.N.MADHAVA RAO M.Tech** in partial fulfillment for the award of the degree of “**Bachelor of Technology in Computer Science& Engineering(Artificial intelligence)**”of **Jawaharlal Nehru Technological University, Kakinada.**

K.POOJA (20F91A4330)

M.NANDINI (20F91A4332)

T.PRATYUSHA (20F91A4361)

V.MANEESHA (20F91A4362)

ACKNOWLEDGEMENT

We feel to render our thankful acknowledgement to the following distinguished personalities, who stretched their helping hand to us, in completing our mini project work.

We are very grateful and our sincere thanks to our secretary & correspondent **Dr.K.RAMAIAH** of **PRAKASAM ENGINEERING COLLEGE** for giving this opportunity.

We hereby, express our regards and extend our gratitude to our **PRINCIPAL, Dr.CH. RAVI KUMAR** , for giving this opportunity to do the thesis as a part of our course.

We express our deep sense of gratitude to **Mr.K.SUBBAREDDY, M.Tech(Ph.d)**, Head of the Department, Department of CSE for having shown keen interest at every stage of development of our thesis and guiding us in every aspect.

And We are thankful to our guide **Mr.N.MADHAVA RAO M.Tech**, who has channeled our thoughts and timely suggestions.

We would also like to thank all our Faculties in Prakasam Engineering College for their constant encouragement and for being a great group of knowledgeable and cooperative people to work with

K.POOJA (20F91A4330)

M.NANDINI (20F91A4332)

T.PRATHYUSHA (20F91A4361)

V.MANEESHA (20F91A4362)

TABLE OF THE CONTENT

CHAPTER PAGE NO

ABSTRACT	I
LIST OF FIGURES	II-III
CHAPTER 1: INTRODUCTION	1-9
1.1 Existing Systems	4
1.1.1 Disadvantages of existing systems	6
1.2 Proposed Systems	7
1.2.1 Advantages of Proposed Systems	9
CHAPTER 2: LITERATURE SURVEY	10-13
CHAPTER 3: SYSTEM ANALYSIS	14-54
3.1 Feasibility Study	14
3.1.1 Economical Feasibility	14
3.1.2 Technical Feasibility	14
3.1.3 Social Feasibility	15
3.2 System Requirements	15
3.3 Module Description	16
3.4 Software Environment	17
3.4.1 Java technology	17
3.4.2 How Will Java Technology Change My Life?	22
3.4.3 JDBC Goals	25
3.4.4 Networking	28
3.4.5 JFree Chart	31
3.4.6 J2ME(Java 2 Micro Edition)	33
3.4.7 J2EE Software Environment	36
3.4.8 About Java	39

3.4.9 Javascript	43
3.4.10 Hyper Text Markup Language	45
3.4.11 Java Database Connectivity	47
 CHAPTER 4: SYSTEM DESIGN	 55-63
4.1 System Architecture	55
4.2 Dataflow Diagram	56
4.3 Flowchart	58
Use case Diagrams	59
4.5 Class Diagram	60
4.6 Sequence Diagram	61
4.7 Input Design and Output Design	62
CHAPTER 5: IMPLEMENTATION	64-74
5.1 code	64
 CHAPTER 6: SYSTEM TESTING	 75-79
6.1 Testing methodologies	75
6.2 Test Strategy	79
 CHAPTER 7: RESULTS	 80-89
 CHAPTER 8: CONCLUSION	 90
 CHAPTER 9: FUTURE ENHANCEMENT	 91
 REFERENCES	 92

ABSTRACT

Medical imaging is crucial for medical diagnosis, and the sensitive nature of medical images necessitates rigorous security and privacy solutions to being place. In a cloud-based medical system for Healthcare Industry 4.0, medical images should be encrypted prior to being outsourced. However, processing queries over encrypted data without first executing the decryption operation is challenging and impractical at present. In the paper, we propose a secure and efficient scheme to find the exact nearest neighbor over encrypted medical images. Instead of calculating the Euclidean distance, we reject candidates by computing the lower bound of Euclidean distance that is related to the mean and standard deviation of data. Unlike most existing schemes, our scheme can obtain the exact nearest neighbor rather than an approximate result. We then evaluate our proposed approach to demonstrate its utility.

LIST OF FIGURES

SL.NO	FIGURE NO	FIGURE NAME	PAGE NO
1	3.4.1(a)	Working of java program	18
2	3.4.1(b)	Java interpreter	19
3	3.4.1(c)	Figure on java program	20
4	3.4.1(d)	Java 2 SDK	22
5	3.4.4	TCP/IP stack	
6	3.4.6	General J2ME architecture	33
7	3.4.8	Compiling and interpreting Java source code	42
8	4.1	Architecture Diagram	55
9	4.2	Data flow diagram	56
10	4.3	Flow chart	58
11	4.4	Usecase diagram	59
12	4.5	Class diagram	60
13	4.6	Sequence diagram	61
14	7.1	Main page	80
15	7.2	Server login	81
16	7.3	Server1 page	82
17	7.4	Medical data	83

18	7.5	All users page	84
19	7.6	Attacter page	84
20	7.7	Server 2	85
21	7.8	User login page	86
22	7.9	Data owner	87
23	7.10	User page	87
24	7.11	View All image Rank chart Result	88
25	7.12	View All image Attack count chart	89

III

CHAPTER-1

INTRODUCTION

Cloud computing is becoming a norm in our society [1], and in such deployment, the data owner can outsource databases and management functionalities to the cloud server. The latter stores the databases and supplies access mechanisms to query and manage the outsourced database. This allows data owners to reduce data management expenses and improve quality of service. However, the cloud may not be fully trusted because it may leak sensitive information to unauthorized entities (e.g., compromised) or foreign government agencies [2].

The rapid evolution of cloud computing is revolutionizing e-Health and the whole Industry 4.0 in the field of healthcare. The cloud-based electronic health care system is one popular application for Healthcare Industry 4.0. A well- design electronic healthcare system can obviously improve the quality of access and experience of healthcare users. In recent years, applications for cloud computing and big-data computing in the Healthcare Industry 4.0 have been enthusiastically discussed [3], [4], [5]. In an electronic healthcare system, patients' medical images may be outsourced to a third-party [6], such as a professional community health care cloud server. Within the healthcare industry, many forms of medical imaging play a crucial role in diagnosis and quality of care, including magnetic resonance imaging (MRI), ultrasonic, computed tomography (CT) and computed radiography. In addition, for the healthcare industry, prior or archival records (e.g., disease out breaks) can have significant reference value. When a new patient is seen, doctors can Efficiently and accurately make a diagnosis and develop the appropriate treatment.

programs by finding similar cases in the database and analyzing them. For example, hospitals or related medical institutions can store patients' medical images in a professional and secure database of a Practical electronic healthcare system. If doctors acquire medical images from a new patient, they may find similar images in the outsourced database to be used as a reference. This is particularly crucial when dealing with symptoms from rare diseases such as those listed.

Nearest neighbour search, therefore, can be used in such a scenario. However, the sensitivity and privacy of medical images require that the security and privacy of such images be ensured and preserved.

Encrypting data by the data owner is a naive method to ensure privacy [7], while it ensures the secrecy of the Out sourced data from the cloud and unauthorized users. Additionally, to protect query privacy, permitted users should send their requests to the cloud for evaluation after encryption. However, by analyzing the data access patterns, the cloud (or a malicious insider) can derive private information about the real data items even though the data and queries are encrypted [8], [9]. In detail, the access pattern includes not only the content of the data block accessed by the user but also the way how the user accesses the data block, such as frequency, location, order, habit and so on. By statistical analysis, data mining or other techniques, the cloud server can infer user's type, hobbies and the frequency of accessing specific content. For example, traffic analysis technique can obtain some sensitive information about access pattern. When a user searches through a service provider such as Google, the search history will leak the user's search habits even the identity. Also, the frequency of the search may leak the popularity of the retrieved data.

Additionally, the cloud server can establish some correlation among successive accesses.

In other words, we need to ensure secrecy of the outsourced data and a user's query record insecure query processing, as well as hiding data access patterns. In order to be practical and efficient, the proposed scheme needs to reduce the computation cost on the end-user as much as possible. The nearest neighbour search is a vital operation in data mining, machine learning, and information retrieval, and more recently the healthcare industry as well. Recently, due to the emergence of high-dimensional medical images, the importance of having an efficient and effective nearest neighbour search algorithm (in terms of speed and space) has become more pronounced.

Generally, it is a difficult task to process encrypted data without first executing the decryption operation. The challenge is how a cloud server can process the queries over encrypted medical images. Motivated by this challenge, we propose an efficient and effective scheme to search for the exact nearest neighbour over outsourced encrypted medical images.

Specifically, in the paper, we discuss the problem of exact nearest neighbour search scheme encrypted medical images propose a secure and efficient solution. Our scheme supports dynamic updates. It allows data users to easily add or delete medical images whenever necessary.

In the next section, we will introduce related work on nearest neighbour search. In Section III, we discuss relevant background materials including the system and the design goals. The proposed scheme is explained in Section IV. Our security and performance analysis are outlined in Section V, and in the last section.

1.1 EXSISTING SYSTEM

Nearest neighbour search, first studied by Knuth [10] in 1973, has been the focus of ongoing research. In the literature, there are a large number of methods relating to the query process over encrypted data, such as searchable encryption [11]. In 2006, Curtmola et al. presented a new searchable symmetric encryption (SSE) construction to resist chosen keyword attacks [12]. Kamara et al. proposed a dynamic SSE scheme and a new security framework to optimize dynamic operations [13]. In 2013, Kamara et al. proposed a new dynamic SSE scheme, which is designed to support parallelizable search [14].

However, these schemes focus on keyword search, rather than the nearest neighbour search. We refer interested readers to [15] for a survey of existing SSE schemes. One could also use sequential scan (brute-force search), which sequentially calculates and compares the Euclidean distance between the query item and every record in the encrypted database. However, this scheme needs significant time and space requirements, which are proportional to the number and dimension of the data. Thus, such a scheme is not appropriate for dealing with large-scale and high dimensional Data despite the existence of methods to minimize the costs. In 2009, wongetal.

[16] Suggested searching for nearest neighbours over encrypted data using the a symmetric scalar product preserving encryption (ASPE) scheme. However, using the scheme requires linear search time relating to the number of data records. Two years later in 2011, Hu et al. [17] introduced a scheme with tree-based data structures and ASPE, which results in faster search time.

Unfortunately, in this scheme the client who wishes to execute queries needs to perform numerous interactions with the server. Also, the need to maintain a local index will incur (significant) local storage costs. In 2006, Zhu et al. [18] designed a safer scheme which can resist potential attacks of cloud server and avoid key-sharing. Zhu et al.

[19] in 2017 proposed an efficient scheme for k-nearest neighbour search, which enhances the protection of the decryption key and reduces the burden on data.

There are a number of efficient schemes to find the approximate nearest neighbour, designed to improve efficiency in space and time at the cost of accuracy. The scheme based on Locality.

Sensitive Hashing (LSH) is a well-known and effective method that solves the nearest neighbour query problem in high-dimensional space. The LSH scheme [20] embeds data in low-dimensional subspaces and utilizes hash tables to improve efficiency. In 2004, Data et al. [21] provided a basic LSH based on the original LSH scheme, which exploited the property of p-stable distribution to generalize the LSH method from the Hamming space to the Euclidean space. This scheme, however, incurs significant overhead in space. In 2007, And on et al. [22] introduced the Leech lattice into the LSH scheme of [20], which reduces the query time and memory consumption. Hashing algorithms are useful to deal with high-dimension and large-scale data, but they require.

Additional cost when implemented in the exact nearest neighbour search. Due to the need for accuracy in the healthcare industry, these LSH-based schemes are not suitable for solving the medical images problems.

Researchers have also proposed methods using tree-based data structures to efficiently find the nearest neighbour in the plaintext domain.

As early as 1975, Bentley [23] proposed KD-tree and used this special data structure to store information that can be retrieved by associative searches. Many query types can be efficiently handled by this single data structure. Three decades later, Jagadish et al.

[24] Presented an efficient B+-tree structure to address the K-nearest neighbour (KNN) search problem involving high-dimensional data.

Other typical examples include R-tree variations [25], [26], [27] and Cover tree [28]. These schemes, which are based on special data structures can reduce the search time cost because of the tree-structured organization of data. However, such data structures require large pre process time and memory space, and so they are not appropriate for high dimensional and large-scale data.

In 2012, Ahn et al. [27] proposed a nearest neighbour search algorithm for plaintext. This algorithm reduces dimensions of data points by embedding them in low dimensional space. Non-nearest neighbours are eliminated via a comparison of the distances in this space. Due to this property, this algorithm is suitable for processing.

In particular, unlike most existing algorithms, this method can obtain the exact nearest neighbour rather than an approximate one. The beauty of this method is simplicity, in the sense of simple pre processing without involving complex data structures.

1.1.1 Disadvantages

- 1) The system's problem is the problem of exact nearest neighbour search over encrypted medical images and proposes a secure and efficient solution.
- 2) The system is less security due to lack of Locality Sensitive Hashing (LSH).

1.2 PROPOSEDSYSTEM

In the proposed system, the system proposed scheme has good scalability performance. And the dynamic changes to the database have almost no impact on our algorithm. The scalability is mainly reflected in two aspects. On the one hand, the owner Alice wants to add one or more medical images to the data base that has been encrypted and out sourced.

In this case, Alice first computes the mean and standard deviation of these new images, then she encrypts and uploads these images and related means, standard deviations and IDs to the cloud server.

1) Accuracy: To meet the need for accuracy in the healthcare industry, the proposed scheme achieves a precise nearest neighbour search instead of an approximation.

2) Security: Healthcare Industry 4.0 requires high security and privacy. Our scheme is designed to ensure the confidentiality of all related medical images. To ensure query privacy, the database and query need to be encrypted before sending to the cloud server. The cloud servers can infer nothing useful when analyzing the data flow in the query process.

3) Efficiency: Efficiency is a remarkable feature of Healthcare Industry 4.0. Our scheme has significantly less computational expense and resource consumption than existing schemes. In our scheme, we do not need to compute the Euclidean distance for all data, and the query process incurs low computational overhead on the user.

4) Dynamic: Our scheme supports dynamic updates. Data users can easily add or delete medical images that have been outsourced to the cloud server.

Certainly! Here's a proposed system for a fast nearest neighbour search scheme over outsourced encrypted medical images:

1. Encryption of Medical Images: Before outsourcing the medical images, they are encrypted using a secure encryption scheme such as homo morphic encryption or fully homo morphic encryption (FHE). This ensure that the images remain confidential and secure even while being processed by third- party servers.

2. *Indexing Scheme: Develop an indexing scheme that allows for efficient searching over the encrypted data.

This might involve techniques like locality sensitive hashing (LSH) or tree-based structures adapted for encrypted data, such as encrypted search trees or encrypted spatial indexes.

3. Search Nearest Neighbour Search Algorithm: Design a secure nearest neighbour search algorithm that operates directly on the encrypted data. This algorithm should be able to find the nearest neighbour of a query image without decryption the entire dataset. Techniques such as secure a k-nearest neighbour (k-NN) search or secure similarity search can be explored.

4. Query Processing: When a query for nearest neighbour is received, the encrypted query image is processed using the secure search algorithm to identity the nearest neighbours within the encrypted dataset. Only the encrypted Identifiers or indices of the nearest neighbours are returned to the users.

5. *Decryption of Nearest Neighbour *: Once the encrypted identifiers of the nearest neighbour are obtained, they can be decrypted locally by the user using the corresponding decryption keys. This allows the users to access the actual images of the nearest neighbours while preserving the confidentiality of the dataset.

6. Performance Optimization: Optimize the performance of the system by fine-tuning parameters, improving algorithms, and possibly parallelizing computations to achieve faster search times while maintaining security.

7. Evaluation and Testing*: Evaluate the proposed system using benchmark datasets of medical images to assess its performance in terms of search accuracy, efficiency, and security. Conduct thorough testing to validate the effectiveness of the system under various scenarios and workload conditions.

8. Privacy and Security Analysis: Perform a comprehensive privacy and security analysis to identify potential vulnerabilities and ensure that the system provides strong protection against unauthorized access, data breaches, and privacy violations.

9. * Integration and Deployment*: Integrate the proposed system into existing medical imaging systems or platform, ensuring seamless compatibility and ease of deployment. Provide documentation and support for users to facilitate adoption and usage of the system.

10. Continuous improvement: Continuously monitor and update the system to address any emerging security threats, improve performance, and incorporate feedback from users and stakeholders for ongoing refinement and enhancement.

1.2.1 Advantages

1.) The system is more effective since it is implemented using the scheme (FNN) which is very fast and secure.

2.) The system is accurate since the data search technique is Nearest Neighbour.

CHAPTER-2

LITERATURE SURVEY

“Computation partitioning for mobile cloud computing in big data environment,”

Proposed two secure indexing schemes, both of them can achieve the similarity of two images by computing the Jac card index between their encrypted feature vectors. Meanwhile, Lu et al. also combined signal processing (e.g., bit-planer and omization, random projection, and randomized unary encoding) and cryptographic techniques (e.g., XOR and random permutation) to investigate three feature protection schemes, which can be used as building blocks to achieve PPCBIR over large encrypted image databases .

“Cloud computing: Challenges and future directions,”

Proposed a privacy-preserving SIFT method based on homomorphic encryption to address the secure problem encountered in the outsourced environment. After that, the improved homomorphic encryption techniques are employed in to encrypt indexes, although they achieved more efficient retrieval than conventional privacy preserving schemes, the computation overhead is too high to be suitable for practical applications. Compared with homomorphic encryption-based techniques the work improved that the feature/index randomization-based techniques perform better in term of computational and communication cost.

“Access pattern disclosure on searchable encryption: Ramification, attack and mitigation,”.

first designed an efficient PPCBIR scheme in the cloud computing scenario byintroducing a secure KNN technique to encrypt indexes, and then constructed pre-filtortablestoimprovetheretrieval efficiencybasedonLocalitySensitiveHashing(LSH)tech nique.WithsecureKNN,Xiaetal.presentedanotherefficientandcopy-deterrence PPCBIR scheme similar to which utilizes watermark extraction totraceunlawfulqueryusertoimprovethesecurity.

“A health- IoT platform based on the integration of intelligent packaging, unobtrusive bio-sensor, and intelligent medicine box,”

extracted Fisher vector as indexes and used K-means clustering to narrow the search range, they achieved a lightweight PPCBIR scheme over encrypted data based on secure KNN, too. In addition, PPCBIR schemes were also proposed based on secure KNN. employed a deep learning model, Convolutional Neural Networks (CNN), to improve the accuracy of retrieval, and constructed an encrypted hierarchical index tree to speed up the query phase.

“Private and Secured Medical Data Transmission and Analysis for Wireless Sensing Healthcare System,”

presented a verifiable fine-grained encrypted image retrieval scheme, which is capable of supporting efficient fine-grained access control and result verification simultaneously. However, secure KNN has been proven is insecure under the threat model in which the adversary only knows cipher text.

“Multidimensional binary search trees used for associative searching,”

The growth of Mobile cloud computing (MCC) is challenged by the need to adapt to the resources and environment available for mobile clients while working with the dynamic changes of network bandwidth. In this project, propose a model of computation partitioning for state full data in the dynamic environment that will improve performance. First, constructed a model of state full data streaming and studied the method of computation partitioning in a dynamic environment and developed a definition of direction and calculation of the segmentation scheme, including single frame data flow, task scheduling and executing efficiency. Second, we proposed a computation partitioning method for Single frame data flow and determined the data parameters of the application model, the computation partitioning scheme, and the task and work order data stream model.

“Computation partitioning for mobile cloud computing in big data environment,”

In modern hospitals, patients are treated using a wide array of medical devices that are increasingly interacting with each other over the network, thus offering a perfect example of a cyber-physical system and study the safety of a medical device system for the physiologic closed-loop control of drug infusion.

The main contribution of the project is the verification approach for the safety properties of closed-loop medical device systems and demonstrates, using a case study, that the approach can be applied to a system of clinical importance. Our method combines simulation based analysis of a detailed model of the system that contains continuous patient dynamics with model checking of a more abstract timed automata model.

“Ubiquitous data accessing method in IoT-based information system for emergency medical services,”

The rapid development of Internet of things (IoT) technology makes it possible for connecting various smart objects together through the Internet and providing more data interoperability methods for application purpose. Recent research shows more potential applications of IoT in information intensive industrial sectors such as healthcare services. However, the diversity of the objects in IoT causes the heterogeneity problem of the data format in IoT platform. Meanwhile, the use of IoT technology in applications has spurred the increase of real-time data, which makes the information storage and accessing more difficult and challenging. In this research, first a semantic data model is proposed to store and interpret IoT data. Then a resource-based data accessing method (UDA-IoT) is designed to acquire and process IoT data ubiquitously to improve the accessibility to IoT data resources.

“Toward privacy-assured cloud data services with flexible search functionalities,”

The provided citation refers to a paper titled "Toward privacy-assured cloud data services with flexible search functionalities" authored by M. Li, S. Yu, W. Lou, and Y. T. How. It was presented at the ICDCSW (International Conference on Distributed Computing Systems Workshops), likely in 2012, held in Macau, China. The paper is published in the proceedings of the conference, and it discusses the development of cloud data services that prioritize privacy while offering flexible search functionalities.

“Building castles out of mud: practical access pattern privacy and correctness on un trusted storage,”

The citation provided refers to a paper titled "Building castles out of mud: practical access pattern privacy and correctness on un trusted storage," authored by P. Williams,

R. Sion, and B. Carbunar. It was presented at the CCS (Conference on Computer and Communications Security), likely in 2008, held in Alexandria, VA, USA. The paper is published in the proceedings of the conference by the Association for Computing Machinery (ACM). It appears to focus on practical techniques for maintaining access pattern privacy and data correctness in scenarios where storage is untrusted.

“Secure KNN Computation on Encrypted Databases,”

The provided citation refers to a paper titled "Secure KNN Computation on Encrypted Databases," authored by W. Wong, D. W. Cheung, B. Kao, and N. Mamoulis. It was presented at ACM SIGMOD (Special Interest Group on Management of Data) conference, likely in 2009, held in Providence, Rhode Island, USA. The paper is published in the proceedings of the conference by the Association for Computing Machinery (ACM). The paper likely discusses techniques for performing secure k-nearest neighbors (KNN) computation on encrypted databases, ensuring privacy while still enabling useful data analysis.

CHAPTER-3

SYSTEM ANALYSIS

3.1 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**

3.1.1 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.

3.1.2 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.

3.1.3 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.

3.2 SYSTEM REQUIREMENTS

H/W System Configuration:-

- Processor - Pentium-IV
- RAM - 4 GB(min)
- 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

3.3 Modules

Data Owner

In this module, the data provider uploads their encrypted data in the Cloud server. For the security purpose the data owner encrypts the data file and then store in the server. The Data owner can have capable of manipulating the encrypted data file and performs the following operations 1. Register and Login with username and password and Repository key (give option to key get repository key by entering password) Create Repository (image title, Image Tag, Image Desc (EncDesc), Medical Image Uses, attach image, generate image key using RSA, uploaded Date and time), View all Secret Key Medical images with keys and date and time, view all users with medical image key request and give access permission.

Cloud Server

The Cloud server manages which is to provide data storage service for the Data Owners. Data owners encrypt their data files and store them in the Server for sharing with data consumers and performs the following operations such as View all Medical images, View Users Reposted keys, view all Medical Image access details with date and time, view all attackers those who used wrong key to access image, View Image rank in chart, View count of attacked same file in chart.

Data User

In this module, the user can only access the data file with the secret key. The user can search the file for a specified keyword. The data which matches for a particular keyword will be indexed in the cloud server and then response to the end user and can do the following operations like Search Medical images by content keyword, Request Medical image key to corresponding user, View Image key response and decrypt image contents and view medical image with its details.

Key Distribution Service

In this module, the KDS performs the following operations such as View all Users and generate Repository key for each and every user, View Repository key Access details with Date and Time, view all Image key access from one to another users with date and time, View all Medical images with keys.

3.4 SOFTWARE ENVIRONMENT

3.4.1 Java Technology

Java technology is both a programming language and a platform.

The Java Programming Language

The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

- Simple
- Architecture neutral
- Object oriented
- Portable
- Distributed

- High performance
- Interpreted
- Multithreaded
- Robust
- Dynamic
- Secure

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called *Java byte codes* —the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.

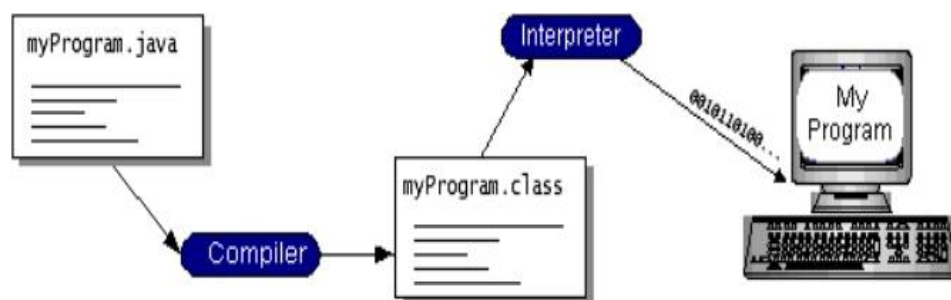


FIG 3.4.1(a) working of java program

You can think of Java byte codes as the machine code instructions for the JavaVirtual Machine (Java VM). Every Java interpreter, whether it's a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java byte codes help make “write once, run anywhere” possible. You can compile your program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the Java VM.

That means that as long as a computer has a Java VM, the same program written in the Java programming language can run on Windows 2000, a Solaris workstation, or on an iMac.

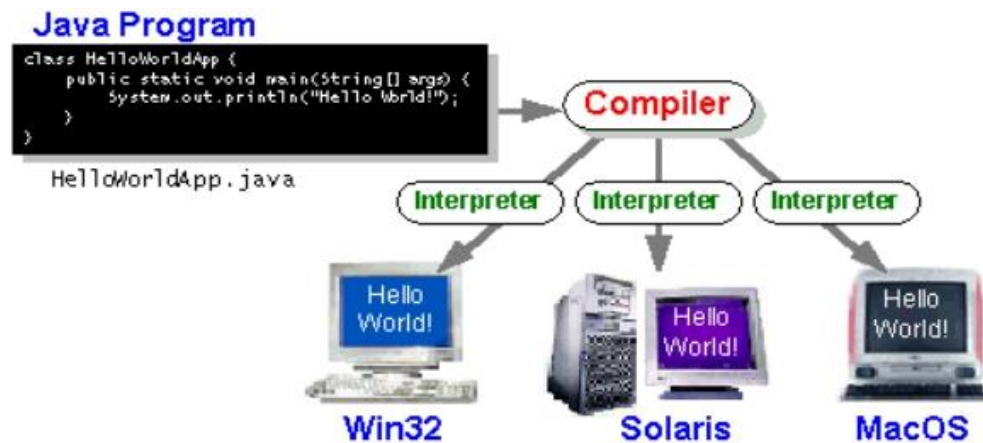


Fig 3.4.1(b) Java interpreter

The Java Platform

A platform is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and Mac OS. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.

The Java platform has two components:

- The Java Virtual Machine (Java VM)
- The Java Application Programming Interface (Java API)

You've already been introduced to the Java VM. It's the base for the Java platform and is ported onto various hardware-based platforms.

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets.

The Java API is grouped into libraries of related classes and interfaces; these libraries are known as *packages*. The next section, What Can Java Technology Do? Highlights what functionality some of the packages in the Java API provide.

The following figure depicts a program that's running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware.

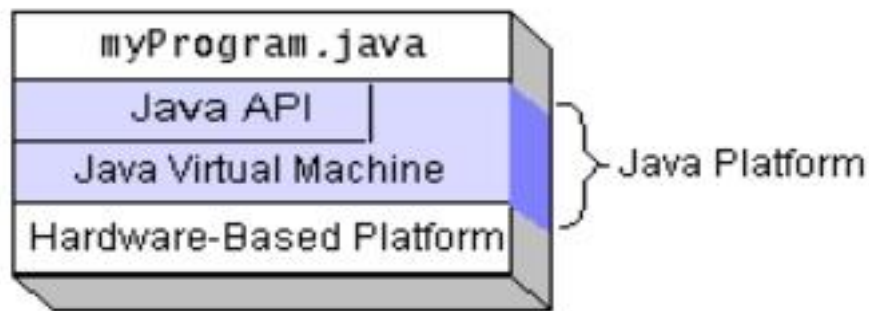


Fig 3.4.1(c) Figure on Java Program

Native code is code that after you compile it, the compiled code runs on a specific hardware platform. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time byte code compilers can bring performance close to that of native code without threatening portability.

What Can Java Technology Do?

The most common types of programs written in the Java programming language are applets and applications. If you've surfed the Web, you're probably already familiar with applets. An applet is a program that adheres to certain conventions that allow it to run within a Java-enabled browser.

However, the Java programming language is not just for writing cute, entertaining applets for the Web. The general-purpose, high-level Java programming language is also a powerful software platform. Using the generous API, you can write many types of programs.

An application is a standalone program that runs directly on the Java platform. A special kind of application known as a *server* serves and supports clients on a network. Examples of servers are Web servers, proxy servers, mail servers, and print servers. Another specialized program is a servlet. A servlet can almost be thought of as an applet that runs on the server side. Java Servlets are a popular choice for building interactive web applications, replacing the use of CGI scripts. Servlets are similar to applets in that they are runtime extensions of applications. Instead of working in browsers, though, servlets run within Java Web servers, configuring or tailoring the server.

How does the API support all these kinds of programs? It does so with packages of software components that provides a wide range of functionality. Every full implementation of the Java platform gives you the following features:

- **The essentials:** Objects, strings, threads, numbers, input and output, data structures, system properties, date and time, and so on.
- **Applets:** The set of conventions used by applets.
- **Networking:** URLs, TCP (Transmission Control Protocol), UDP (User Datagram Protocol) sockets, and IP (Internet Protocol) addresses.
- **Internationalization:** Help for writing programs that can be localized for users worldwide. Programs can automatically adapt to specific locales and be displayed in the appropriate language.
- **Security:** Both low level and high level, including electronic signatures, public and private key management, access control, and certificates.
- **Software components:** Known as Java Beans TM, can plug into existing component architectures.
- **Object serialization:** Allows lightweight persistence and communication via Remote Method Invocation (RMI).
- **Java Database Connectivity (JDBCTM):** Provides uniform access to a wide range of relational databases.

The Java platform also has APIs for 2D and 3D graphics, accessibility, servers, collaboration, telephony, speech, animation, and more.

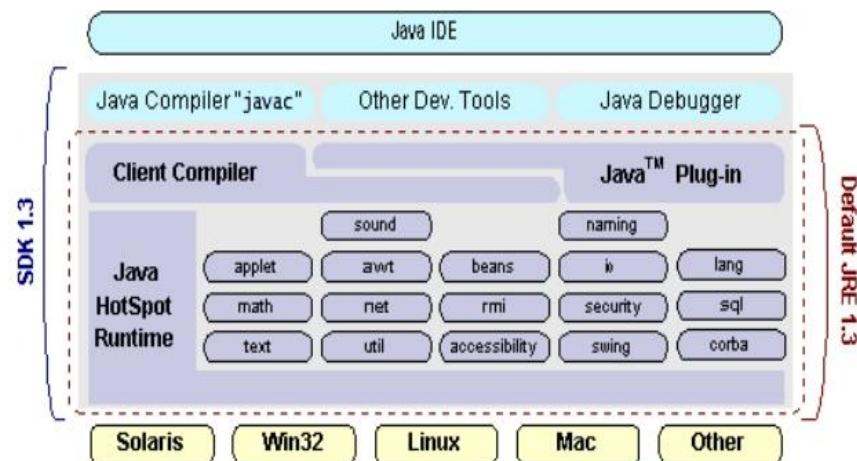


Fig 3.4.1(d) Java 2 SDK

3.4.2 How Will Java Technology Change My Life?

We can't promise you fame, fortune, or even a job if you learn the Java programming language. Still, it is likely to make your programs better and requires less effort than other languages. We believe that Java technology will help you do the following:

- **Get started quickly:** Although the Java programming language is a powerful object-oriented language, it's easy to learn, especially for programmers already familiar with C or C++.
- **Write less code:** Comparisons of program metrics (class counts, method counts, and so on) suggest that a program written in the Java programming language can be four times smaller than the same program in C++.
- **Write better code:** The Java programming language encourages good coding practices, and its garbage collection helps you avoid memory leaks. Its object orientation, its JavaBeans component architecture, and its wide-ranging, easily extendible API let you reuse other people's tested code and introduce fewer bugs.
- **Develop programs more quickly:** Your development time may be as much as twice as fast versus writing the same program in C++. Why? You write fewer lines of code and it is a simpler programming language than C++.

- **Avoid platform dependencies with 100% Pure Java:** You can keep your program portable by avoiding the use of libraries written in other languages. The 100% Pure Java TM Product Certification Program has a repository of historical process manuals, white papers, brochures, and similar materials online.
- **Write once, run anywhere:** Because 100% Pure Java programs are compiled into machine-independent byte codes, they run consistently on any Java platform.
- **Distribute software more easily:** You can upgrade applets easily from a central server. Applets take advantage of the feature of allowing new classes to be loaded “on the fly,” without recompiling the entire program.

ODBC

Microsoft Open Database Connectivity (ODBC) is a standard programming interface for application developers and database systems providers. Before ODBC became a *de facto* standard for Windows programs to interface with database systems, programmers had to use proprietary languages for each database they wanted to connect to. Now, ODBC has made the choice of the database system almost irrelevant from a coding perspective, which is as it should be. Application developers have much more important things to worry about than the syntax that is needed to port their program from one database to another when business needs suddenly change.

Through the ODBC Administrator in Control Panel, you can specify the particular database that is associated with a data source that an ODBC application program is written to use. Think of an ODBC data source as a door with a name on it. Each door will lead you to a particular database. For example, the data source named Sales Figures might be a SQL Server database, whereas the Accounts Payable data source could refer to an Access database. The physical database referred to by a data source can reside anywhere on the LAN.

The ODBC system files are not installed on your system by Windows 95. Rather, they are installed when you setup a separate database application, such as SQL Server Client or Visual Basic 4.0.

When the ODBC icon is installed in Control Panel, it uses a file called ODBCINST.DLL. It is also possible to administer your ODBC data sources through a stand-alone program called ODBCADM.EXE

There is a 16-bit and a 32-bit version of this program and each maintains a separate list of ODBC data sources.

From a programming perspective, the beauty of ODBC is that the application can be written to use the same set of function calls to interface with any data source, regardless of the database vendor. The source code of the application doesn't change whether it talks to Oracle or SQL Server. We only mention these two as an example. There are ODBC drivers available for several dozen popular database systems. Even Excel spreadsheets and plain text files can be turned into data sources. The operating system uses the Registry information written by ODBC Administrator to determine which low-level ODBC drivers are needed to talk to the data source (such as the interface to Oracle or SQL Server). The loading of the ODBC drivers is transparent to the ODBC application program. In a client/server environment, the ODBC API even handles many of the network issues for the application programmer.

The advantages of this scheme are so numerous that you are probably thinking there must be some catch. The only disadvantage of ODBC is that it isn't as efficient as talking directly to the native database interface. ODBC has had many detractors make the charge that it is too slow. Microsoft has always claimed that the critical factor in performance is the quality of the driver software that is used. In our humble opinion, this is true. The availability of good ODBC drivers has improved a great deal recently. And anyway, the criticism about performance is somewhat analogous to those who said that compilers would never match the speed of pure assembly language. Maybe not, but the compiler (or ODBC) gives you the opportunity to write cleaner programs, which means you finish sooner. Meanwhile, computers get faster every year.

JDBC

In an effort to set an independent database standard API for Java; Sun Microsystems developed Java Database Connectivity, or JDBC.

JDBC offers a generic SQL database access mechanism that provides a consistent interface to a variety of RDBMSs. This consistent interface is achieved through the use of “plug-in” database connectivity modules, or *drivers*.

If a database vendor wishes to have JDBC support, he or she must provide the driver for each platform that the database and Java run on.

To gain a wider acceptance of JDBC, Sun based JDBC’s framework on ODBC. As you discovered earlier in this chapter, ODBC has widespread support on a variety of platforms. Basing JDBC on ODBC will allow vendors to bring JDBC drivers to market much faster than developing a completely new connectivity solution.

JDBC was announced in March of 1996. It was released for a 90 day public review that ended June 8, 1996. Because of user input, the final JDBC v1.0 specification was released soon after. The remainder of this section will cover enough information about JDBC for you to know what it is about and how to use it effectively. This is by no means a complete overview of JDBC. That would fill an entire book.

3.4.3 JDBC Goals

Few software packages are designed without goals in mind. JDBC is one that, because of its many goals, drove the development of the API. These goals, in conjunction with early reviewer feedback, have finalized the JDBC class library into a solid framework for building database applications in Java.

The goals that were set for JDBC are important. They will give you some insight as to why certain classes and functionalities behave the way they do. The eight design goals for JDBC are as follows:

1.SQL Level API

The designers felt that their main goal was to define a SQL interface for Java. Although not the lowest database interface level possible, it is at a low enough level for higher-level tools and APIs to be created. Conversely, it is at a high enough level for application programmers to use it confidently.

Attaining this goal allows for future tool vendors to “generate” JDBC code and to hide many of JDBC’s complexities from the end user.

2.SQL Conformance

SQL syntax varies as you move from database vendor to database vendor. In an effort to support a wide variety of vendors, JDBC will allow any query statement to be passed through it to the underlying database driver. This allows the connectivity module to handle non-standard functionality in a manner that is suitable for its users.

3. JDBC must be implemental on top of common database interfaces

The JDBC SQL API must “sit” on top of other common SQL level APIs. This goal allows JDBC to use existing ODBC level drivers by the use of a software interface. This interface would translate JDBC calls to ODBC and vice versa.

4.Provide a Java interface that is consistent with the rest of the Java system

Because of Java’s acceptance in the user community thus far, the designers feel that they should not stray from the current design of the core Java system.

5. Keep it simple

This goal probably appears in all software design goal listings. JDBC is no exception. Sun felt that the design of JDBC should be very simple, allowing for only one method of completing a task per mechanism. Allowing duplicate functionality only serves to confuse the users of the API.

6. Use strong, static typing wherever possible

Strong typing allows for more error checking to be done at compile time; also, less error appear at runtime.

7. Keep the common cases simple

Because more often than not, the usual SQL calls used by the programmer are simple SELECT’s, INSERT’s, DELETE’s and UPDATE’s, these queries should be simple to perform with JDBC. However, more complex SQL statements should also be possible.

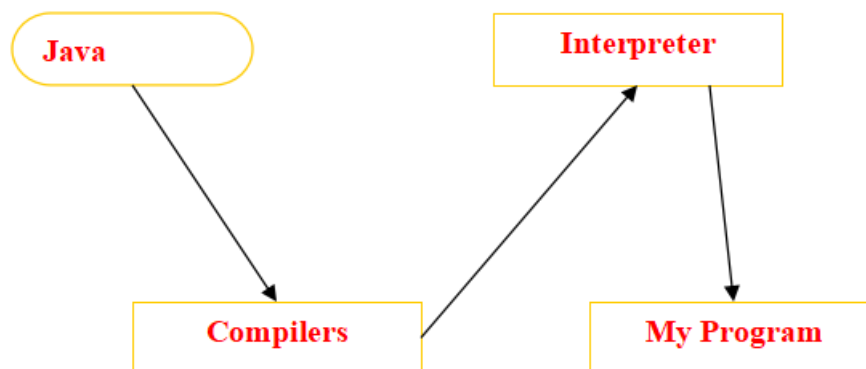
Finally we decided to proceed the implementation using Java Networking. And for dynamically updating the cache table we go for MS Access database. Java has two things: a programming language and a platform.

Java is a high-level programming language that is all of the following :

Simple	Architecture-neutral
Object-oriented	Portable
Distributed	High-performance
Interpreted	multithreaded
Robust	Dynamic
Secure	

Java is also unusual in that each Java program is both compiled and interpreted. With a compile you translate a Java program into an intermediate language called Java byte codes the platform-independent code instruction is passed and run on the computer.

Compilation happens just once; interpretation occurs each time the program is executed. The figure illustrates how this works.



You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM).

Every Java interpreter, whether it's a Java development tool or a Web browser that can run Java applets, is an implementation of the Java VM. The Java VM can also be implemented in hardware.

Java byte codes help make “write once, run anywhere” possible. You can compile your Java program into byte codes on my platform that has a Java compiler. The byte codes can then be run any implementation of the Java VM. For example, the same Java program can run Windows NT, Solaris, and Macintosh.

3.4.4 Networking

TCP/IP stack

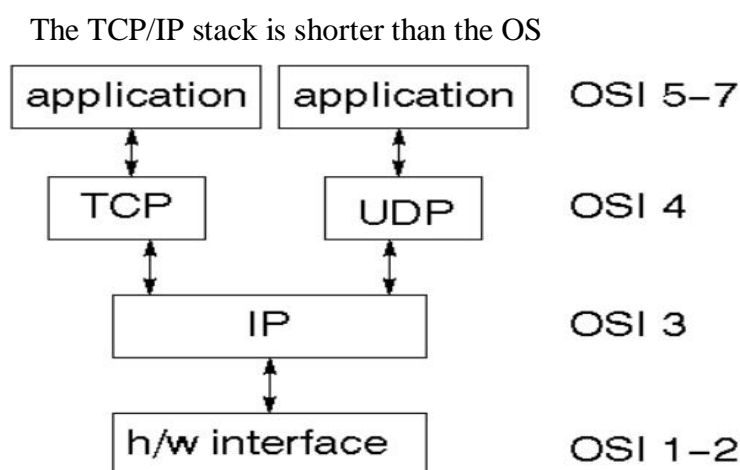


Fig 3.4.4 TCP/IP stack

TCP is a connection-oriented protocol; UDP (User Datagram Protocol) is a connectionless protocol.

IP datagram's

The IP layer provides a connectionless and unreliable delivery system. It considers each datagram independently of the others. Any association between datagram must be supplied by the higher layers. The IP layer supplies a checksum that includes its own header. The header includes the source and destination addresses. The IP layer handles routing through an Internet.

It is also responsible for breaking up large datagram into smaller ones for transmission and reassembling them at the other end.

UDP

UDP is also connectionless and unreliable. What it adds to IP is a checksum for the contents of the datagram and port numbers. These are used to give a client/server model - see later.

TCP

TCP supplies logic to give a reliable connection-oriented protocol above IP. It provides a virtual circuit that two processes can use to communicate.

Internet addresses

In order to use a service, you must be able to find it. The Internet uses an address scheme for machines so that they can be located. The address is a 32 bit integer which gives the IP address. This encodes a network ID and more addressing.

Network address

Class A uses 8 bits for the network address with 24 bits left over for other addressing. Class B uses 16 bit network addressing. Class C uses 24 bit network addressing and class D uses all 32.

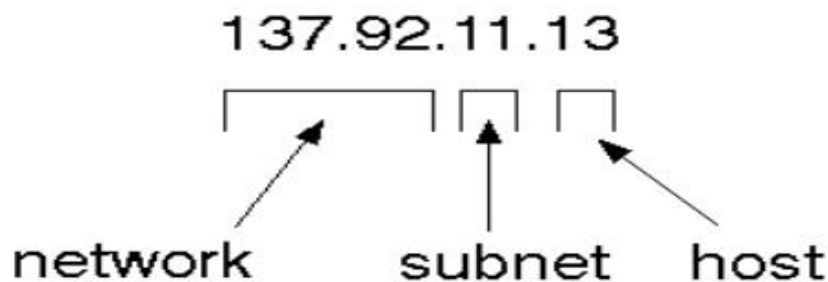
Subnet address

Internally, the UNIX network is divided into sub networks. Building 11 is currently on one sub network and uses 10-bit addressing, allowing 1024 different hosts.

Host address

8 bits are finally used for host addresses within our subnet. This places a limit of 256 machines that can be on the subnet.

Total address



The 32 bit address is usually written as 4 integers separated by dots.

Port addresses

A service exists on a host, and is identified by its port. This is a 16 bit number. To send a message to a server, you send it to the port for that service of the host that it is running on. This is not location transparency! Certain of these ports are "well known".

Sockets

A socket is a data structure maintained by the system to handle network connections. A socket is created using the call `socket`. It returns an integer that is like a file descriptor. In fact, under Windows, this handle can be used with `Read File` and `Write File` functions.

```
#include <sys/ types. h>
#include <sys/socket. h>
int socket(int family, int type, int protocol);
```

Here "family" will be `AF_INET` for IP communications, protocol will be zero, and type will depend on whether TCP or UDP is used. Two processes wishing to communicate over a network create a socket each. These are similar to two ends of a pipe - but the actual pipe does not yet exist.

3.4.5 J Free Chart

J Free Chart is a free 100% Java chart library that makes it easy for developers to display professional quality charts in their applications. J Free Chart's extensive feature set includes:

A consistent and well-documented API, supporting a wide range of chart types;
A flexible design that is easy to extend, and targets both server-side and client-side applications;

Support for many output types, including Swing components, image files (including PNG and JPEG), and vector graphics file formats (including PDF, EPS and SVG);

J Free Chart is "open source" or, more specifically, free software. It is distributed under the terms of the GNU Lesser General Public Licence (LGPL), which permits use in proprietary applications.

1. Map Visualizations

Charts showing values that relate to geographical areas. Some examples include: (a) population density in each state of the United States, (b) income per capita for each country in Europe, (c) life expectancy in each country of the world. The tasks in this project include:

Sourcing freely redistributable vector outlines for the countries of the world, states/provinces in particular countries (USA in particular, but also other areas);

Creating an appropriate dataset interface (plus default implementation), a rendered, and integrating this with the existing XY Plot class in J Free Chart; Testing, documenting, testing some more, documenting some more.

2. Time Series Chart Interactivity

Implement a new (to J Free Chart) feature for interactive time series charts --- to display a separate control that shows a small version of ALL the time series data, with a sliding "view" rectangle that allows you to select the subset of the time series data to display in the main chart.

3. Dashboards

There is currently a lot of interest in dashboard displays. Create a flexible dashboard mechanism that supports a subset of J Free Chart types (dials, pies, thermometers, bars, and lines/time series) that can be delivered easily via both Java Web Start and an applet.

4. Property Editors

The property editor mechanism in J Free Chart only handles a small subset of the properties that can be set for charts. Extend (or re implement) this mechanism to provide greater end-user control over the appearance of the charts.

3.4.6 J2ME (Java 2 Micro edition):-

Sun Microsystems defines J2ME as "a highly optimized Java run-time environment targeting a wide range of consumer products, including pagers, cellular phones, screen-phones, digital set-top boxes and car navigation systems." Announced in June 1999 at the Java One Developer Conference, J2ME brings the cross-platform functionality of the Java language to smaller devices, allowing mobile wireless devices to share applications. With J2ME, Sun has adapted the Java platform for consumer products that incorporate or are based on small computing devices.

3.4.6. General J2ME architecture

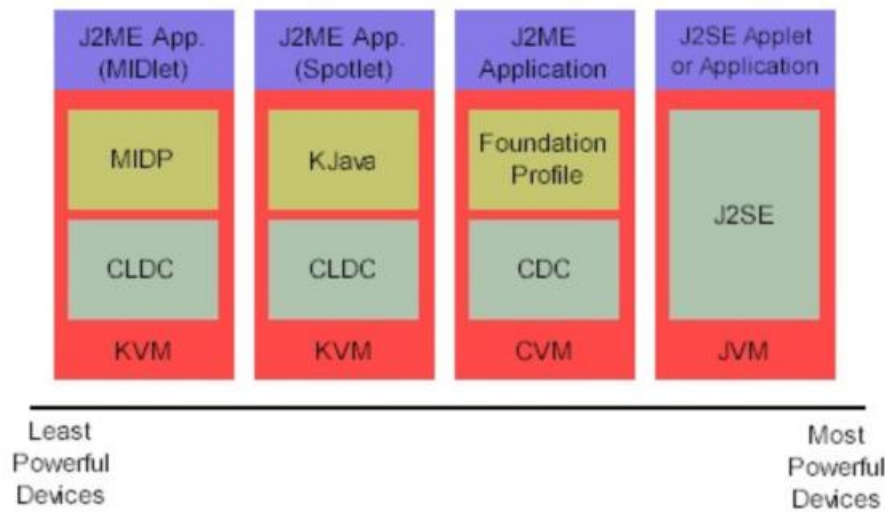


Fig 3.4.6 General J2ME architecture

J2ME uses configurations and profiles to customize the Java Runtime Environment (JRE). As a complete JRE, J2ME is comprised of a configuration, which determines the JVM used, and a profile, which defines the application by adding domain-specific classes. The configuration defines the basic run-time environment as a set of core classes and a specific JVM that run on specific types of devices. We'll discuss configurations in detail in the profile defines the application; specifically, it adds domain-specific classes to the J2ME configuration to define certain uses for devices. We'll cover profiles in depth in the following graphic depicts the relationship between the different virtual machines, configurations, and profiles. It also draws a parallel with the J2SE API and its Java virtual machine. While the J2SE virtual machine is generally referred to as a JVM, the J2ME virtual machines, KVM and CVM, are subsets of JVM. Both KVM and CVM can be thought of as a kind of Java virtual machine -- it's just that they are shrunken versions of the J2SE JVM and are specific to J2ME.

2. Developing J2ME applications

Introduction In this section, we will go over some considerations you need to keep in mind when developing applications for smaller devices.

We'll take a look at the way the compiler is invoked when using J2SE to compile J2ME applications. Finally, we'll explore packaging and deployment and the role pre verification plays in this process.

3.Design considerations for small devices

Developing applications for small devices requires you to keep certain strategies in mind during the design phase. It is best to strategically design an application for a small device before you begin coding. Correcting the code because you failed to consider all of the "go chats" before developing the application can be a painful process. Here are some design strategies to consider:

- * Keep it simple. Remove unnecessary features, possibly making those features a separate, secondary application.
- * Smaller is better. This consideration should be a "no brainer" for all developers. Smaller applications use less memory on the device and require shorter installation times. Consider packaging your Java applications as compressed Java Archive (jar) files.
- * Minimize run-time memory use. To minimize the amount of memory used at run time, use scalar types in place of object types. Also, do not depend on the garbage collector. You should manage the memory efficiently yourself by setting object references to null when you are finished with them. Another way to reduce run-time memory is to use lazy instantiation, only allocating objects on an as-needed basis. Other ways of reducing overall and peak memory use on small devices are to release resources quickly, reuse objects, and avoid exceptions.

1. Configurations overview

The configuration defines the basic run-time environment as a set of core classes and a specific JVM that run on specific types of devices. Currently, two configurations exist for J2ME, though others may be defined in the future:

- * **Connected Limited Device Configuration (CLDC)** is used specifically with the KVM for 16-bit or 32-bit devices with limited amounts of memory. This is the configuration (and the virtual machine) used for developing small J2ME applications.
-

Its size limitations make CLDC more interesting and challenging (from a development point of view) than CDC. CLDC is also the configuration that we will use for developing our drawing tool application. An example of a small wireless device running small applications is a Palm hand-held computer.

* **Connected Device Configuration (CDC)** is used with the C virtual machine (CVM) and is used for 32-bit architectures requiring more than 2 MB of memory. An example of such a device is a Net TV box.

5.J2ME profiles

What is a J2ME profile?

As we mentioned earlier in this tutorial, a profile defines the type of device supported. The Mobile Information Device Profile (MIDP), for example, defines classes for cellular phones. It adds domain-specific classes to the J2ME configuration to define uses for similar devices. Two profiles have been defined for J2ME and are built upon CLDC: KJava and MIDP. Both KJava and MIDP are associated with CLDC and smaller devices. Profiles are built on top of configurations. Because profiles are specific to the size of the device (amount of memory) on which an application runs, certain profiles are associated with certain configurations.

A skeleton profile upon which you can create your own profile, the Foundation Profile, is available for CDC.

Profile 1: K Java

K Java is Sun's proprietary profile and contains the K Java API. The K Java profile is built on top of the CLDC configuration. The K Java virtual machine, KVM, accepts the same byte codes and class file format as the classic J2SE virtual machine. K Java contains a Sun-specific API that runs on the Palm OS. The K Java API has a great deal in common with the J2SE Abstract Windowing Toolkit (AWT). However, because it is not a standard J2ME package, its main package is `com.sun.kjava`. We'll learn more about the K Java API later in this tutorial when we develop some sample applications.

Profile 2: MIDP

MIDP is geared toward mobile devices such as cellular phones and pagers. The MIDP, like K Java, is built upon CLDC and provides a standard run-time environment that allows new applications and services to be deployed dynamically on end user devices. MIDP is a common, industry-standard profile for mobile devices that is not dependent on a specific vendor. It is a complete and supported foundation for mobile application development. MIDP contains the following packages, the first three of which are core CLDC packages, plus three MIDP-specific packages.

- * java.lang
- * java.io
- * java.util
- * javax.microedition.io
- * javax.microedition.lcdui
- * javax.microedition.midlet
- * javax.microedition.rms

3.4.7 J2EE SOFTWARE ENVIRONMENT

Client Server

Overview:

With the varied topic in existence in the fields of computers, Client Server is one, which has generated more heat than light, and also more hype than reality. This technology has acquired a certain critical mass attention with its dedication conferences and magazines. Major computer vendors such as IBM and DEC, have declared that Client Servers is their main future market. A survey of DBMS magazine revealed that 76% of its readers were actively looking at the client server solution.

The growth in the client server development tools from \$200 million in 1992 to more than \$1.2 billion in 1996.

Client server implementations are complex but the underlying concept is simple and powerful. A client is an application running with local resources but able to request the database and relate the services from separate remote server. The software mediating this client server interaction is often referred to as MIDDLEWARE.

The typical client either a PC or a Work Station connected through a network to a more powerful PC, Workstation, Midrange or Main Frames server usually capable of handling request from more than one client. However, with some configuration server may also act as client. A server may need to access other server in order to process the original client request.

The key client server idea is that client as user is essentially insulated from the physical location and formats of the data needs for their application. With the proper middleware, a client input from or report can transparently access and manipulate both local database on the client machine and remote databases on one or more servers. An added bonus is the client server opens the door to multi-vendor database access indulging heterogeneous table joins.

What is a Client Server

Two prominent systems in existence are client server and file server systems. It is essential to distinguish between client servers and file server systems. Both provide shared network access to data but the comparison ends there! The file server simply provides a remote disk drive that can be accessed by LAN applications on a file by file basis. The client server offers full relational database services such as SQL-Access, Record modifying, Insert, Delete with full relational integrity backup/ restore performance for high volume of transactions, etc. the client server middleware provides a flexible interface between client and server.

Why Client Server

Client server has evolved to solve a problem that has been around since the earliest days of computing: how best to distribute your computing, data generation and data storage resources in order to obtain efficient, cost effective departmental and enterprise wide data processing.

During mainframe era choices were quite limited. A central machine housed both the CPU and DATA (cards, tapes, drums and later disks). Access to these resources was initially confined to batched runs that produced departmental reports at the appropriate intervals. A strong central information service department ruled the corporation. The role of the rest of the corporation limited to requesting new or more frequent reports and to provide hand written forms from which the central data banks were created and updated. The earliest client server solutions therefore could best be characterized as “SLAVE- MASTER”. Time-sharing changed the picture. Remote terminal could view and even change the central data, subject to access permissions. And, as the central data banks evolved in to sophisticated relational database with non-programmer query languages, online users could formulate adhoc queries and produce local reports without adding to the MIS applications software backlog. However remote access was through dumb terminals, and the client server remained subordinate to the Slave\Master.

Front end or User Interface Design

The entire user interface is planned to be developed in browser specific environment with a touch of Intranet-Based Architecture for achieving the Distributed Concept.

The browser specific components are designed by using the HTML standards, and the dynamism of the designed by concentrating on the constructs of the Java Server Pages.

Communication or Database Connectivity

The Communication architecture is designed by concentrating on the Standards of Servlets and Enterprise Java Beans. The database connectivity is established by using the Java Data Base Connectivity. The standards of three-tire architecture are given major concentration to keep the standards of higher cohesion and limited coupling for effectiveness of the operations.

Features of the Language Used

In my project, I have chosen *Java* language for developing the code.

3.4.8 About Java

Initially the language was called as “oak” but it was renamed as “Java” in 1995. The primary motivation of this language was the need for a platform-independent (i.e., architecture neutral) language that could be used to create software to be embedded in various consumer electronic devices.

- Java is a programmer’s language.
- Java is cohesive and consistent.
- Except for those constraints imposed by the Internet environment, Java gives the programmer, full control.

Finally, Java is to Internet programming where C was to system programming.

Importance of Java to the Internet

Java has had a profound effect on the Internet. This is because; Java expands the Universe of objects that can move about freely in Cyberspace. In a network, two categories of objects are transmitted between the Server and the Personal computer. They are: Passive information and Dynamic active programs. The Dynamic, Self-executing programs cause serious problems in the areas of Security and probability. But, Java addresses those concerns and by doing so, has opened the door to an exciting new form of program called the Applet.

Java can be used to create two types of program

Applications and Applets: An application is a program that runs on our Computer under the operating system of that computer. It is more or less like one creating using C or C++. Java’s ability to create Applets makes it important. An Applet is an application designed to be transmitted over the Internet and executed by a Java – compatible web browser. An applet is actually a tiny Java program, dynamically downloaded across the network, just like an image. But the difference is, it is an intelligent program, not just a media file. It can react to the user input and dynamically change.

Features of Data

Security

Every time you that you download a “normal” program, you are risking a viral infection. Prior to Java, most users did not download executable programs frequently, and those who did scanned them for viruses prior to execution. Most users still worried about the possibility of infecting their systems with a virus. In addition, another type of malicious program exists that must be guarded against. This type of program can gather private information, such as credit card numbers, bank account balances, and passwords. Java answers both these concerns by providing a “firewall” between a network application and your computer.

When you use a Java-compatible Web browser, you can safely download Java applets without fear of virus infection or malicious intent.

Portability

For programs to be dynamically downloaded to all the various types of platforms connected to the Internet, some means of generating portable executable code is needed .As you will see, the same mechanism that helps ensure security also helps create portability. Indeed, Java’s solution to these two problems is both elegant and efficient.

The Byte code

The key that allows the Java to solve the security and portability problems is that the output of Java compiler is Byte code. Byte code is a highly optimized set of instructions designed to be executed by the Java run-time system, which is called the Java Virtual Machine (JVM). That is, in its standard form, the JVM is an interpreter for byte code.

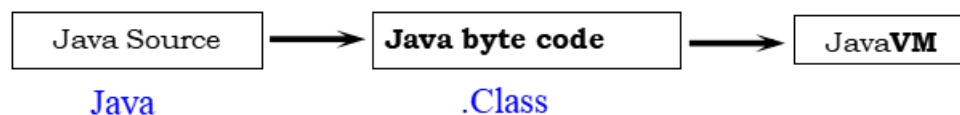
Translating a Java program into byte code helps makes it much easier to run a program in a wide variety of environments. The reason is, once the run-time package exists for a given system, any Java program can run on it.

Although Java was designed for interpretation, there is technically nothing about Java that prevents on-the-fly compilation of byte code into native code. Sun has just completed its Just In Time (JIT) compiler for byte code. When the JIT compiler is a part of JVM, it compiles byte code into executable code in real time, on a piece-by-piece, demand basis. It is not possible to compile an entire Java program into executable code all at once, because Java performs various run-time checks that can be done only at run time. The JIT compiles code, as it is needed, during execution.

Java Virtual Machine (JVM)

Beyond the language, there is the Java virtual machine. The Java virtual machine is an important element of the Java technology. The virtual machine can be embedded within a web browser or an operating system. Once a piece of Java code is loaded onto a machine, it is verified. As part of the loading process, a class loader is invoked and does byte code verification makes sure that the code that's has been generated by the compiler will not corrupt the machine that it's loaded on. Byte code verification takes place at the end of the compilation process to make sure that is all accurate and correct. So byte code verification is integral to the compiling and executing of Java code.

Overall Description



Picture showing the development process of JAVA Program

Java programming uses to produce byte codes and executes them. The first box indicates that the Java source code is located in a. Java file that is processed with a Java compiler called javac. The Java compiler produces a file called a. class file, which contains the byte code. The. Class file is then loaded across the network or loaded locally on your machine into the execution environment is the Java virtual machine, which interprets and executes the byte code.

Java Architecture

Java architecture provides a portable, robust, high performing environment for development. Java provides portability by compiling the byte codes for the Java Virtual Machine, which is then interpreted on each platform by the run-time environment. Java is a dynamic system, able to load code when needed from a machine in the same room or across the planet.

Compilation of code

When you compile the code, the Java compiler creates machine code (called byte code) for a hypothetical machine called Java Virtual Machine (JVM). The JVM is supposed to execute the byte code. The JVM is created for overcoming the issue of portability. The code is written and compiled for one machine and interpreted on all machines. This machine is called Java Virtual Machine.

Compiling and interpreting Java Source Code

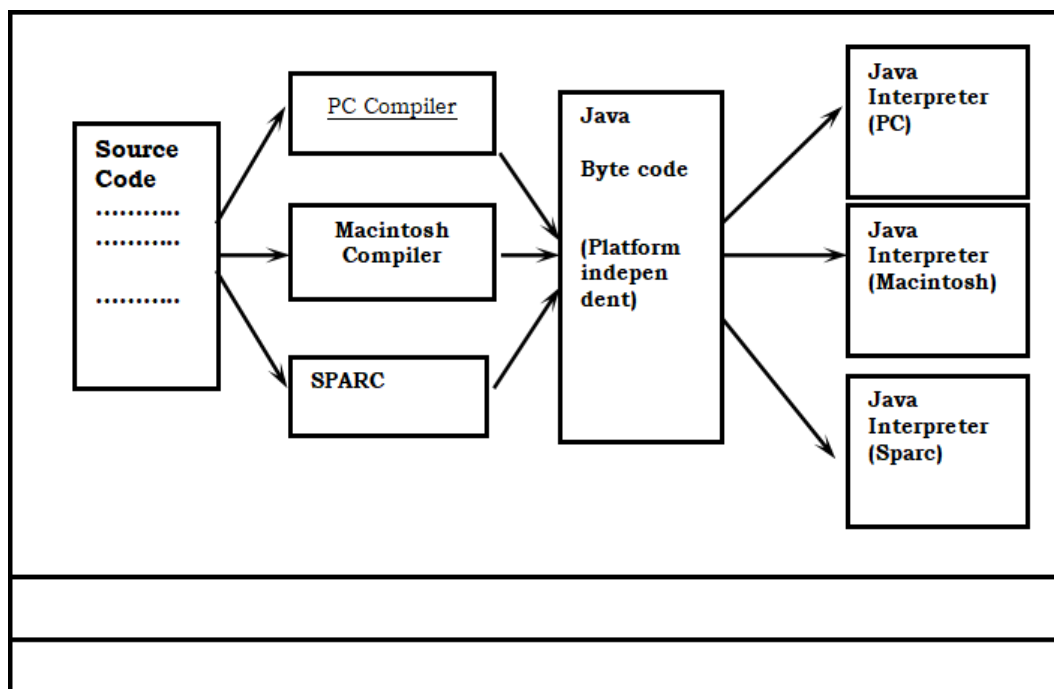


Fig 3.4.8 Compiling and interpreting Java Source Code

Java Source Code

During run-time the Java interpreter tricks the byte code file into thinking that it is running on a Java Virtual Machine.

Simple

Java was designed to be easy for the Professional programmer to learn and to use effectively. If you are an experienced C++ programmer, learning Java will be even easier. Because Java inherits the C/C++ syntax and many of the object oriented features of C++. Most of the confusing concepts from C++ are either left out of Java or implemented in a cleaner, more approachable manner. In Java there are a small number of clearly defined ways to accomplish a given task.

Object-Oriented

Java was not designed to be source-code compatible with any other language. This allowed the Java team the freedom to design with a blank slate. One outcome of this was a clean usable, pragmatic approach to objects. The object model in Java is simple and easy to extend, while simple types, such as integers, are kept as high-performance non-objects.

Robust

The multi-platform environment of the Web places extraordinary demands on a program, because the program must execute reliably in a variety of systems. The ability to create robust programs was given a high priority in the design of Java. Java is strictly typed language; it checks your code at compile time and run time.

Java virtually eliminates the problems of memory management and de allocation, which is completely automatic. In a well-written Java program, all run time errors can –and should –be managed by your program.

3.4.9 JAVASCRIPT

JavaScript is a script-based programming language that was developed by Netscape Communication Corporation. JavaScript was originally called Live Script and renamed as JavaScript to indicate its relationship with Java.

JavaScript supports the development of both client and server components of Web-based applications. On the client side, it can be used to write programs that are executed by a Web browser within the context of a Web page. On the server side, it can be used to write Web server programs that can process information submitted by a Web browser and then updates the browser's display accordingly.

Even though JavaScript supports both client and server Web programming, we prefer JavaScript at Client side programming since most of the browsers supports it. JavaScript is almost as easy to learn as HTML, and JavaScript statements can be included in HTML documents by enclosing the statements between a pair of scripting tags<SCRIPTS>..
</SCRIPT>.

```
<SCRIPT LANGUAGE = "JavaScript">
```

```
JavaScript statements
```

```
</SCRIPT>
```

Here are a few things we can do with JavaScript :

- Validate the contents of a form and make calculations.
- Add scrolling or changing messages to the Browser's status line.
- Animate images or rotate images that change when we move the mouse over them.
- Detect the browser in use and display different content for different browsers.
- Detect installed plug-ins and notify the user if a plug-in is required.

We can do much more with JavaScript, including creating entire application.

Java script Vs Java

Java JavaScript and Java are entirely different languages. A few of the most glaring differences are:

- applets are generally displayed in a box within the web document; JavaScript can affect any part of the Web document itself.

- While JavaScript is best suited to simple applications and adding interactive features to Web pages; Java can be used for incredibly complex applications.

There are many other differences but the important thing to remember is that JavaScript and Java are separate languages. They are both useful for different things; in fact they can be used together to combine their advantages.

ADVANTAGES

- JavaScript can be used for Sever-side and Client-side scripting.
- It is more flexible than VBScript.
- JavaScript is the default scripting languages at Client-side since all the browsers supports it.

3.4.10 Hyper Text Mark up Language

Hypertext Mark up Language (HTML), the languages of the World Wide Web (WWW), allows users to produces Web pages that include text, graphics and pointer to other Web pages (Hyperlinks).

HTML is not a programming language but it is an application of ISO Standard 8879, SGML (Standard Generalized Mark up Language), but specialized to hypertext and adapted to the Web. The idea behind Hypertext is that instead of reading text in rigid linear structure, we can easily jump from one point to another point. We can navigate through the information based on our interest and preference. A mark up language is simply a series of elements, each delimited with special characters that define how text or other items enclosed within the elements should be displayed. Hyperlinks are underlined or emphasized works that load to other documents or some portions of the same document.

HTML can be used to display any type of document on the host computer, which can be geographically at a different location. It is a versatile language and can be used on any platform or desktop. HTML provides tags (special codes) to make the document look attractive. HTML tags are not case-sensitive. Using graphics, fonts, different sizes, colour, etc., can enhance the presentation of the document. Anything that is not a tag is part of the document itself.

Basic HTML Tags :

<!-- -->	Specifies comments
<A>.....	Creates hypertext links
.....	Formats text as bold
<BIG>.....</BIG>	Formats text in large font.
<BODY>...</BODY>	Contains all tags and text in the HTML document
<CENTER>...</CENTER>	Creates text
<DD>...</DD>	Definition of a term
<DL>...</DL>	Creates definition list
...	Formats text with a particular font
<FORM>...</FORM>	Encloses a fill-out form
<FRAME>...</FRAME>	Defines a particular frame in a set of frames
<H#>...</H#>	Creates headings of different levels
<HEAD>...</HEAD>	Contains tags that specify information about a document
<HR>...</HR>	Creates a horizontal rule
<HTML>...</HTML>	Contains all other HTML tags
<META>...</META>	Provides meta-information about a document
<SCRIPT>...</SCRIPT>	Contains client-side or server-side script
<TABLE>...</TABLE>	Creates a table
<TD>...</TD>	Indicates table data in a table
<TR>...</TR>	Designates a table row
<TH>...</TH>	Creates a heading in a table

ADVANTAGES

- A HTML document is small and hence easy to send over the net. It is small because it does not include formatted information.
- HTML is platform independent.
- HTML tags are not case-sensitive.

3.4.11 Java Database Connectivity

What Is JDBC?

JDBC is a Java API for executing SQL statements. (As a point of interest, JDBC is a trademarked name and is not an acronym; nevertheless, JDBC is often thought of as standing for Java Database Connectivity. It consists of a set of classes and interfaces written in the Java programming language. JDBC provides a standard API for tool/database developers and makes it possible to write database applications using a pure Java API.

Using JDBC, it is easy to send SQL statements to virtually any relational database. One can write a single program using the JDBC API, and the program will be able to send SQL statements to the appropriate database. The combinations of Java and JDBC lets a programmer write it once and run it anywhere.

What Does JDBC Do?

Simply put, JDBC makes it possible to do three things:

- Establish a connection with a database
- Send SQL statements
- Process the results.

JDBC versus ODBC and other APIs

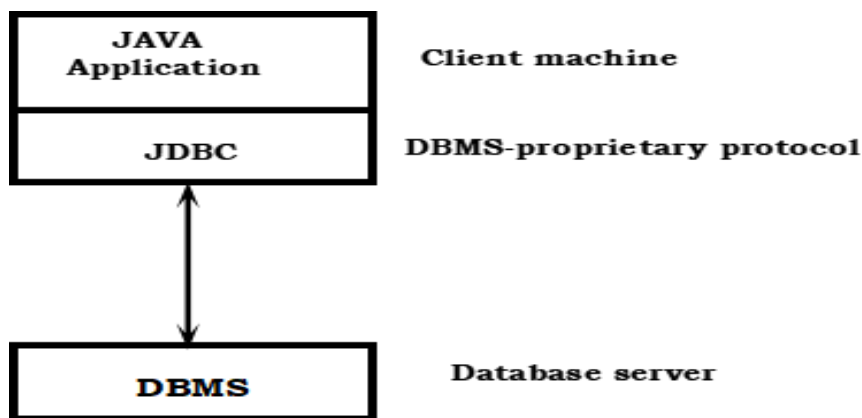
At this point, Microsoft's ODBC (Open Database Connectivity) API is that probably the most widely used programming interface for accessing relational databases. It offers the ability to connect to almost all databases on almost all platforms.

So why not just use ODBC from Java? The answer is that you can use ODBC from Java, but this is best done with the help of JDBC in the form of the JDBC-ODBC Bridge, which we will cover shortly. The question now becomes "Why do you need JDBC?" There are several answers to this question:

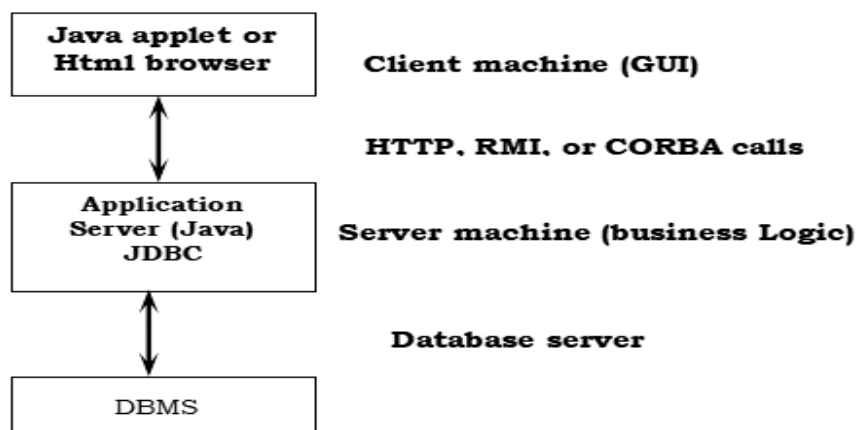
1. ODBC is not appropriate for direct use from Java because it uses a C interface. Calls from Java to native C code have a number of drawbacks in the security, implementation, robustness, and automatic portability of applications.
2. A literal translation of the ODBC C API into a Java API would not be desirable. For example, Java has no pointers, and ODBC makes copious use of them, including the notoriously error-prone generic pointer "void *". You can think of JDBC as ODBC translated into an object-oriented interface that is natural for Java programmers.
3. ODBC is hard to learn. It mixes simple and advanced features together, and it has complex options even for simple queries. JDBC, on the other hand, was designed to keep simple things simple while allowing more advanced capabilities where required.
4. A Java API like JDBC is needed in order to enable a "pure Java" solution. When ODBC is used, the ODBC driver manager and drivers must be manually installed on every client machine. When the JDBC driver is written completely in Java, however, JDBC code is automatically installable, portable, and secure on all Java platforms from network computers to mainframes.

Two-tier and Three-tier Models

The JDBC API supports both two-tier and three-tier models for database access.



In the two-tier model, a Java applet or application talks directly to the database. This requires a JDBC driver that can communicate with the particular database management system being accessed. A user's SQL statements are delivered to the database, and the results of those statements are sent back to the user. The database may be located on another machine to which the user is connected via a network. This is referred to as a client/server configuration, with the user's machine as the client, and the machine housing the database as the server. The network can be an Intranet, which, for example, connects employees within a corporation, or it can be the Internet.



In the three-tier model, commands are sent to a "middle tier" of services, which then send SQL statements to the database. The database processes the SQL statements and sends the results back to the middle tier, which then sends them to the user. MIS directors find the three-tier model very attractive because the middle tier makes it possible to maintain control over access and the kinds of updates that can be made to corporate data.

Another advantage is that when there is a middle tier, the user can employ an easy-to-use higher-level API which is translated by the middle tier into the appropriate low-level calls.

Finally, in many cases the three-tier architecture can provide performance advantages.

Until now the middle tier has typically been written in languages such as C or C++, which offer fast performance. However, with the introduction of optimizing compilers that translate Java byte code into efficient machine-specific code, it is becoming practical to implement the middle tier in Java. This is a big plus, making it possible to take advantage of Java's robustness, multithreading, and security features. JDBC is important to allow database access from a Java middle tier.

JDBC Driver Type

The JDBC drivers that we are aware of at this time fit into one of four categories:

- JDBC-ODBC bridge plus ODBC driver
- Native-API partly-Java driver
- JDBC-Net pure Java driver
- Native-protocol pure Java driver

JDBC-ODBC Bridge

If possible, use a Pure Java JDBC driver instead of the Bridge and an ODBC driver. This completely eliminates the client configuration required by ODBC. It also eliminates the potential that the Java VM could be corrupted by an error in the native code brought in by the Bridge (that is, the Bridge native library, the ODBC driver manager library, the ODBC driver library, and the database client library).

What Is the JDBC- ODBC Bridge?

The JDBC-ODBC Bridge is a JDBC driver, which implements JDBC operations by translating them into ODBC operations. To ODBC it appears as a normal application program. The Bridge implements JDBC for any database for which an ODBC driver is available. The Bridge is implemented as the sun.

jdbc. odbc Java package and contains a native library used to access ODBC. The Bridge is a joint development of Inter solv and Java Soft.

Java Server Pages (JSP)

Java server Pages is a simple, yet powerful technology for creating and maintaining dynamic-content web pages. Based on the Java programming language, Java Server Pages offers proven portability, open standards, and a mature re-usable component model .The Java Server Pages architecture enables the separation of content generation from content presentation. This separation not eases maintenance headaches, it also allows web team members to focus on their areas of expertise. Now, web page designer can concentrate on layout, and web application designers on programming, with minimal concern about impacting each other's work.

Features of JSP

Portability:

Java Server Pages files can be run on any web server or web-enabled application server that provides support for them. Dubbed the JSP engine, this support involves recognition, translation, and management of the Java Server Page lifecycle and its interaction components.

Components

It was mentioned earlier that the Java Server Pages architecture can include reusable Java components. The architecture also allows for the embedding of a scripting language directly into the Java Server Pages file. The components current supported include Java Beans, and Servlets.

Processing

A Java Server Pages file is essentially an HTML document with JSP scripting or tags. The Java Server Pages file has a JSP extension to the server as a Java Server Pages file. Before the page is served, the Java Server Pages syntax is parsed and processed into a Servlet on the server side. The Servlet that is generated outputs real content in straight HTML for responding to the client.

Access Models:

A Java Server Pages file may be accessed in at least two different ways. A client's request comes directly into a Java Server Page. In this scenario, suppose the page accesses reusable Java Bean components that perform particular well-defined computations like accessing a database. The result of the Beans computations, called result sets is stored within the Bean as properties. The page uses such Beans to generate dynamic content and present it back to the client.

In both of the above cases, the page could also contain any valid Java code. Java Server Pages architecture encourages separation of content from presentation.

Steps in the execution of a JSP Application:

1. The client sends a request to the web server for a JSP file by giving the name of the JSP file within the form tag of a HTML page.
2. This request is transferred to the Java Web Server. At the server side Java Web Server receives the request and if it is a request for a jsp file server gives this request to the JSP engine.
3. JSP engine is program which can understands the tags of the jsp and then it converts those tags into a Servlet program and it is stored at the server side. This Servlet is loaded in the memory and then it is executed and the result is given back to the Java Web Server and then it is transferred back to the result is given back to the Java Web Server and then it is transferred back to the client.

JDBC connectivity

The JDBC provides database-independent connectivity between the J2EE platform and a wide range of tabular data sources. JDBC technology allows an Application Component Provider to:

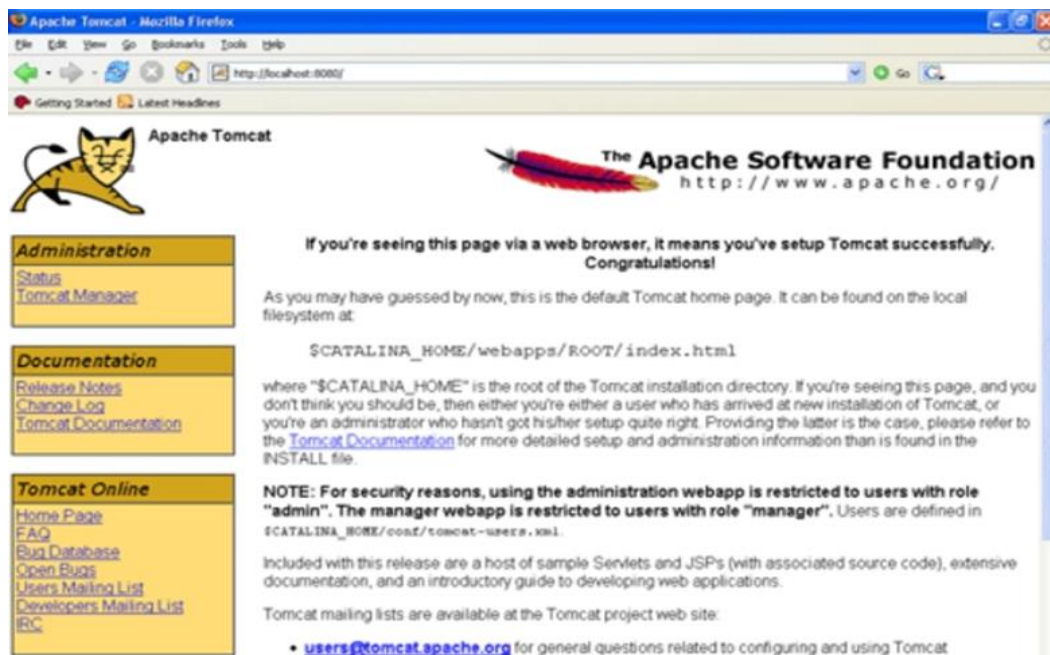
- Perform connection and authentication to a database server
- Manager transactions
- Move SQL statements to a database engine for pre processing and execution.

- Execute stored procedures
- Inspect and modify the results from Select statements.

Tomcat 6.0 web server

Tomcat is an open source web server developed by Apache Group. Apache Tomcat is the servlet container that is used in the official Reference Implementation for the Java Servlet and Java Server Pages technologies. The Java Servlet and Java Server Pages specifications are developed by Sun under the Java Community Process. Web Servers like Apache Tomcat support only web

components while an application server supports web components as well as business components (BEAs Web logic, is one of the popular application server). To develop a web application with jsp /servlet install any web server like J Run, Tomcat etc to run your application.



Bibliography:

References for the Project Development were taken from the following Books and Web sites

PL/SQL Programming by Scott Ur man

SQL complete reference by Livion

JAVA Technologies

JAVA Complete Reference

Java Script Programming by Yehuda Shiran

Mastering JAVA Security

JAVA2 Networking by Pistoria

JAVA Security by Scotl oaks

Head First EJB Sierra Bates

J2EE Professional by Shadab siddiqui

JAVA server pages by Larne Pekowsley

JAVA Server pages by Nick Todd

HTML - HTML Black Book by Holzner

JDBC

Java Database Programming with JDBC by Patel moss.

Software Engineering by Roger Pressman

CHAPTER-4

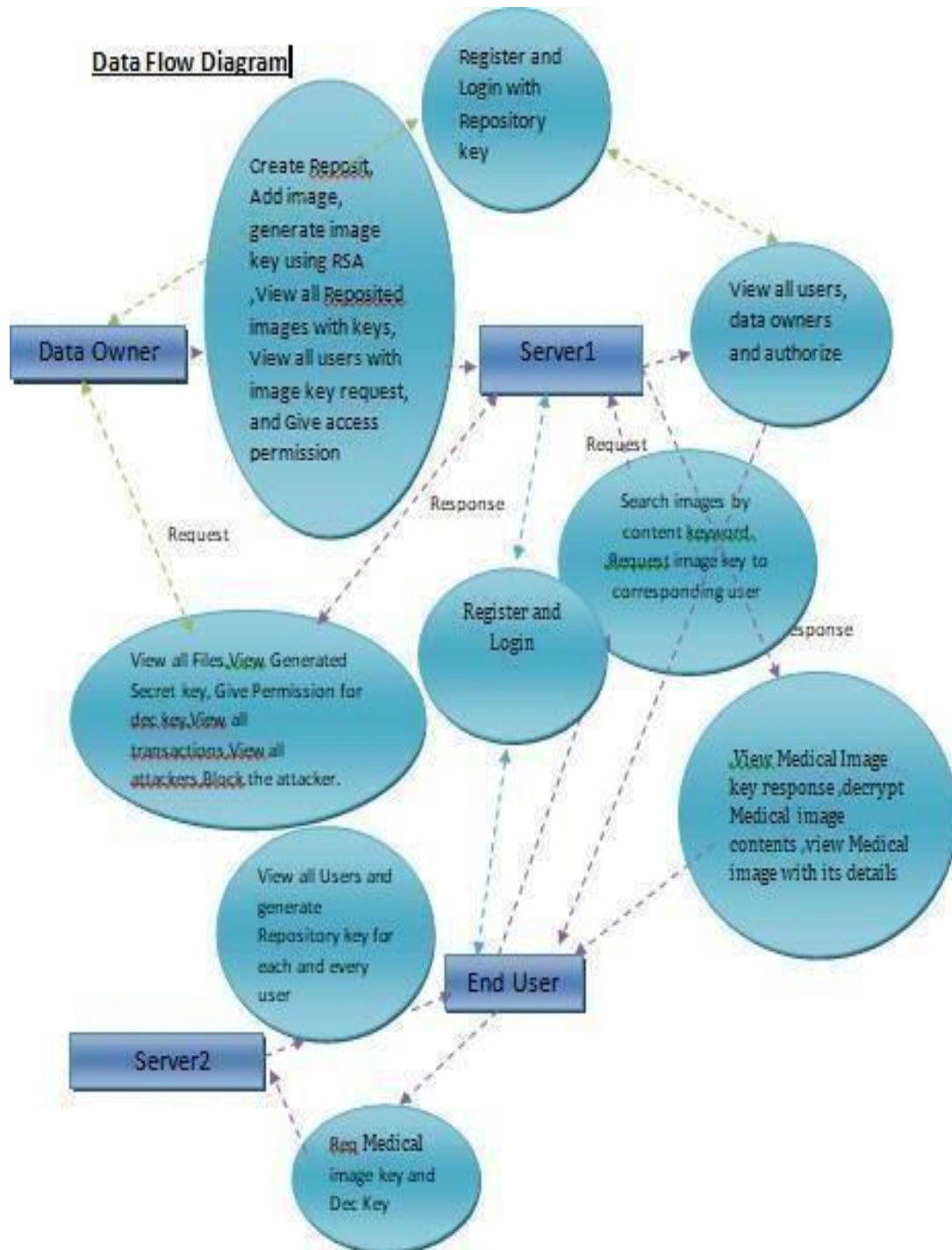
SYSTEM DESIGN

4.1 ArchitectureDiagram

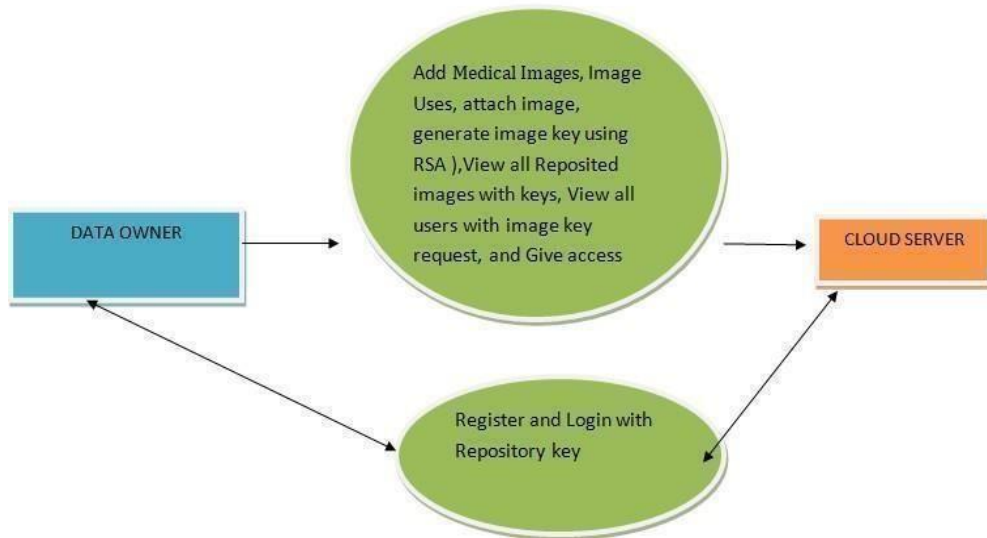


FIG: 4.1 Architecture Diagram

4.2 DataFlowDiagram



Level-0



LEVEL 1

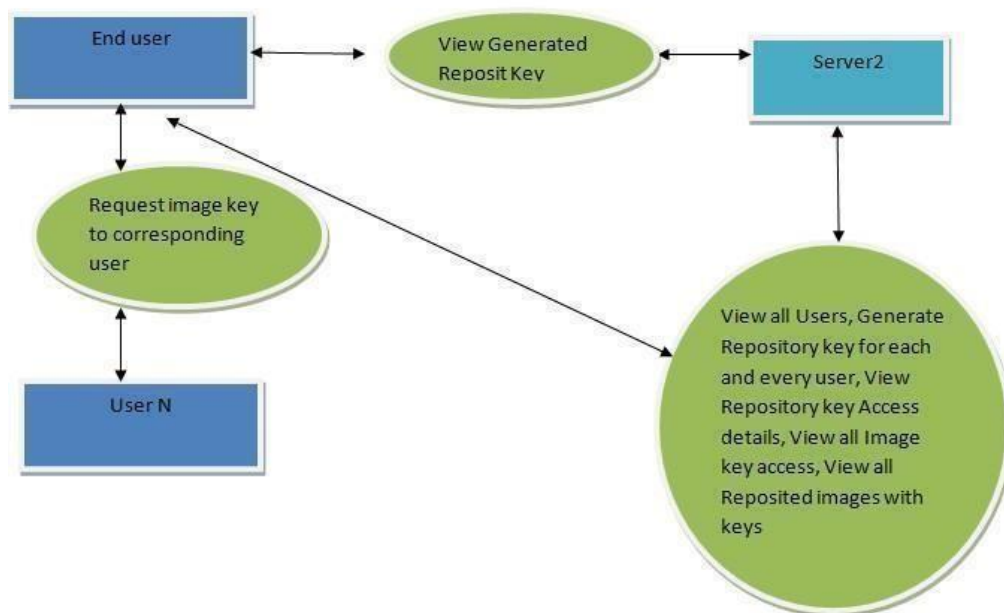


FIG-4.2:DataFlowDiagram

4.3Flowchart

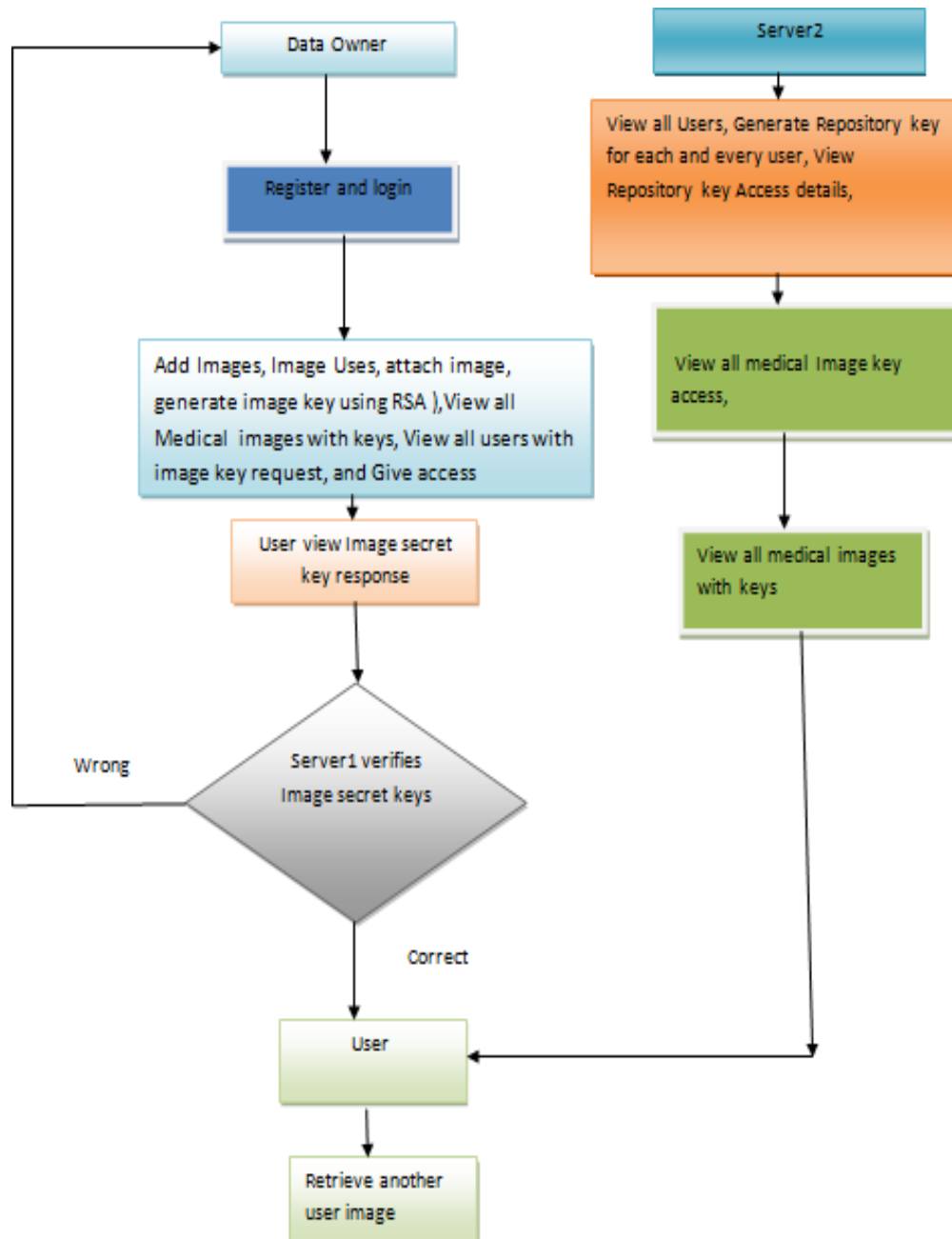


FIG-4.3:Flowchart

4.4 Usecase Diagram

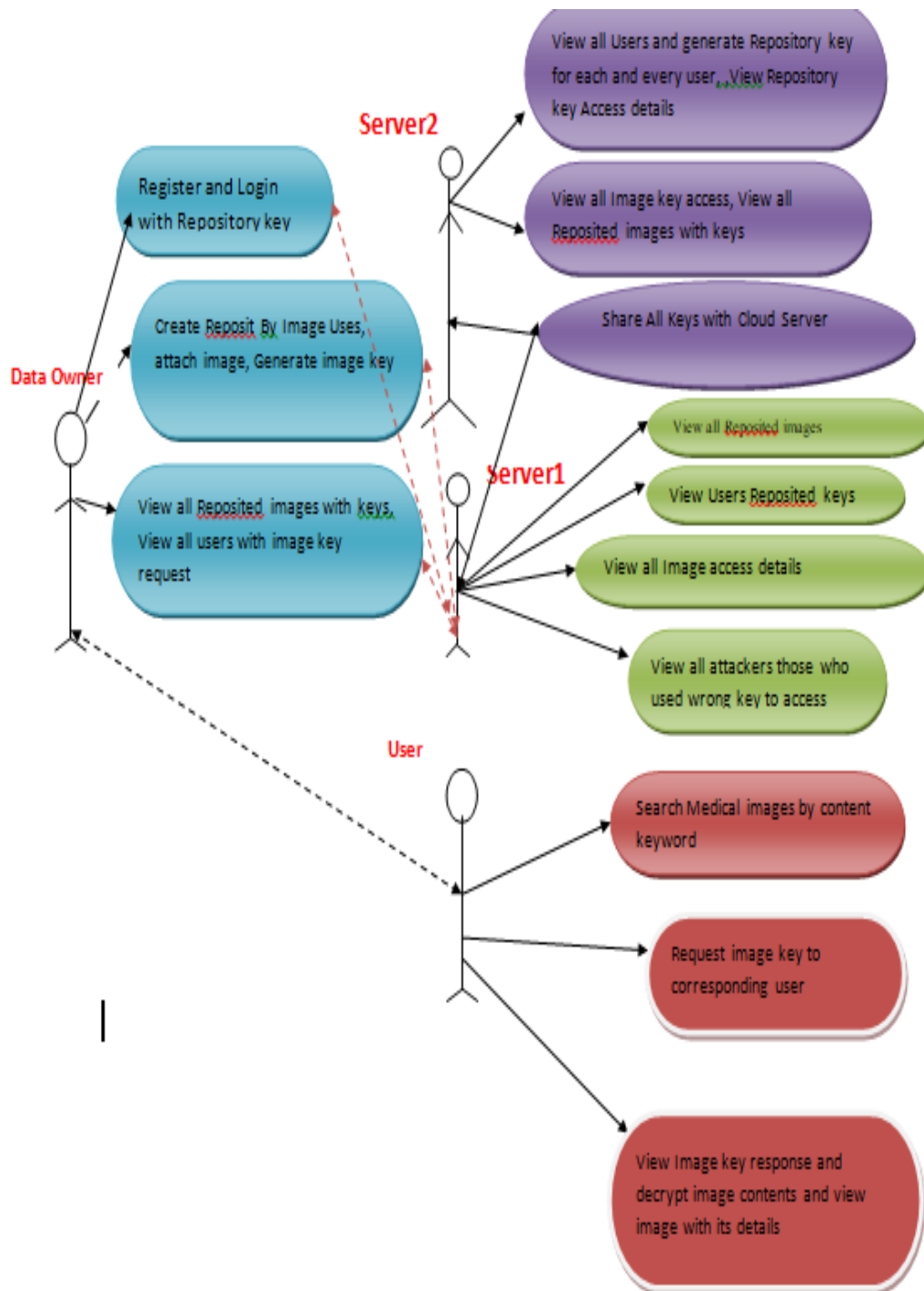


FIG-4.4 :UsecaseDiagram

4.5 Class Diagram

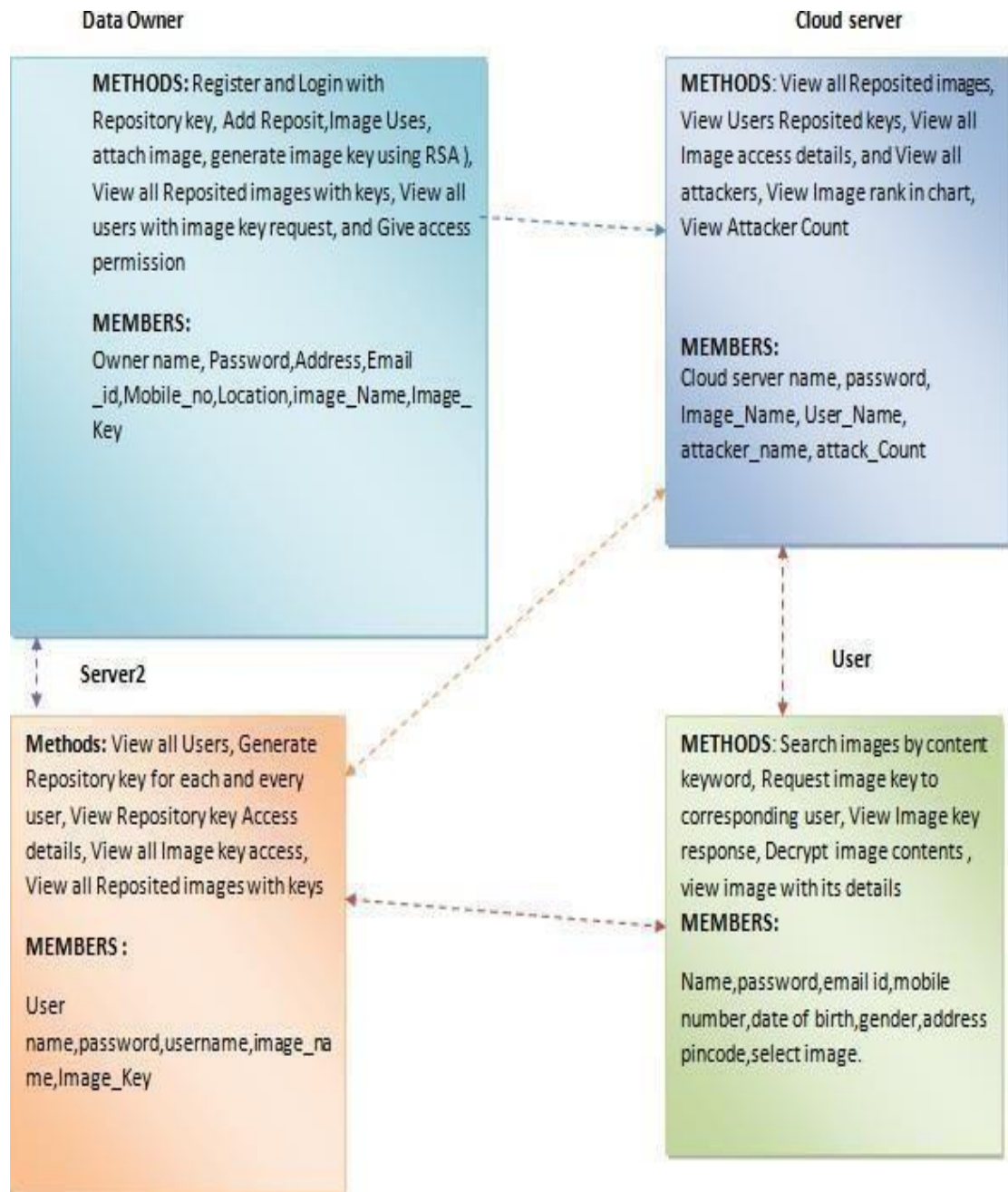


FIG-4.5: Class Diagram

4.6 SequenceDiagram

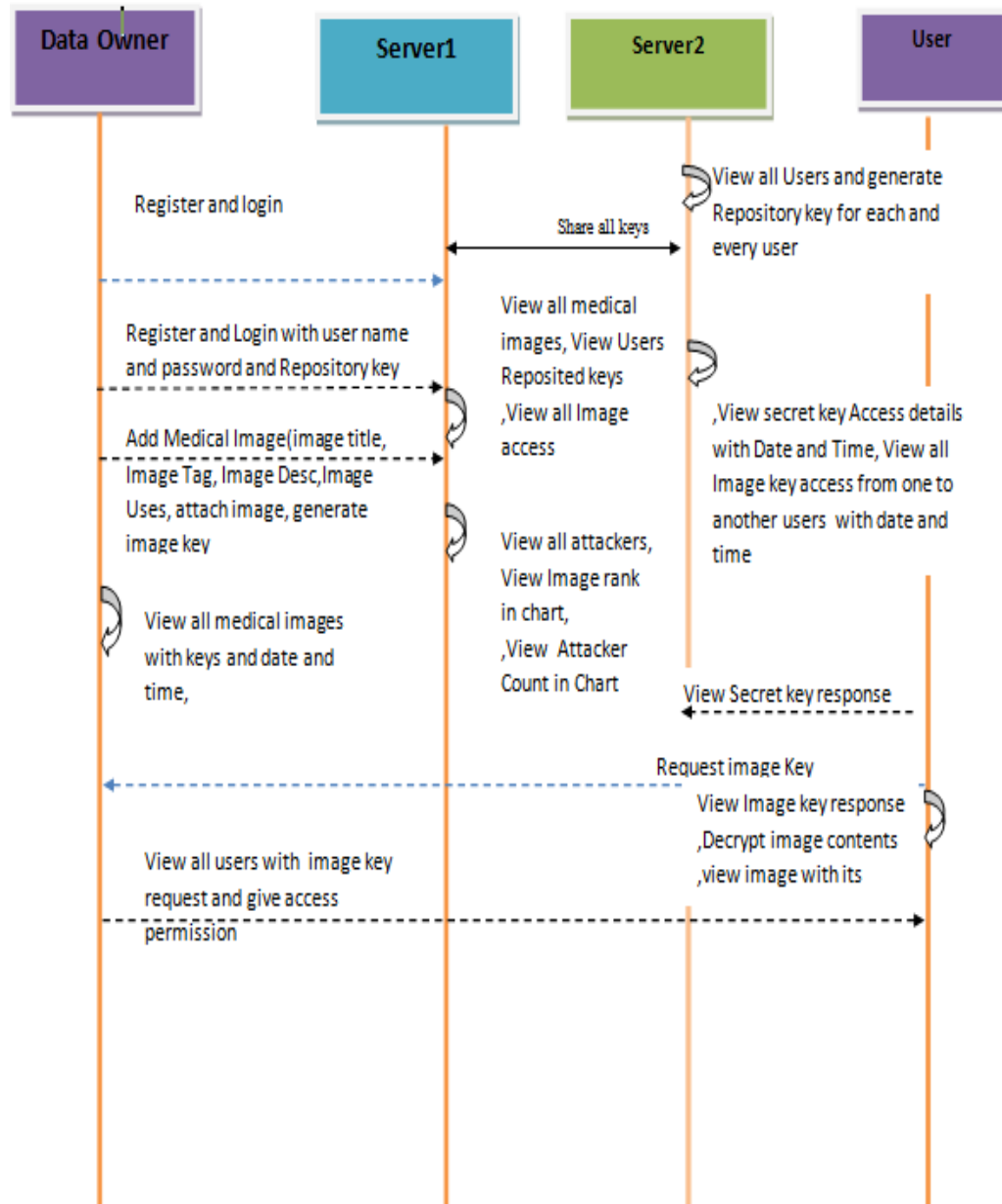


FIG-4.6:SequenceDiagram

4.7 INPUT DESIGN AND OUTPUT DESIGN

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 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 away 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.

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 tests 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 is 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.

CHAPTER-5

IMPLEMENTATION

5.1 CODE

HTML.INDEX

```
<!DOCTYPEhtmlPUBLIC"-//W3C//DTD XHTML 1.0
Strict//EN""http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!--

Design by Free CSS Templates

http://www.freecsstemplates.org

Released for free under a CreativeCommonsAttribution2.5 License

Name :Green Forest

Description:Atwo-column,fixed-widthdesignwithdarkcolorscheme.

Version : 1.0

Released 20110306

-->

<htmlxmlns="http://www.w3.org/1999/xhtml">

<head>

<metaname="keywords"content=""/>

<metaname="description"content=""/>

<metahttp-equiv="content-type"content="text/html;charset=utf-8"/>

<title>HomePage... </title>
```

```

<linkhref="style.css"rel="stylesheet"type="text/css"media="screen"/><scripttype="te
xt/javascript"src="js/cufon-yui.js"></script>

<scripttype="text/javascript"src="js/cufon-ptsans.js"></script>

<scripttype="text/javascript"src="js/jquery-1.4.2.min.js"></script>

<scripttype="text/javascript"src="js/script.js"></script>

<scripttype="text/javascript"src="js/coin-slider.min.js"></script>

<styletype="text/css">

<!--

.style3{

color: #FF0000;

font-weight:bold

}

.style4{color:#FF0000;font-weight:bold;font-size:18px;}-->

</style>

</head>

<body>

<divid="wrapper">

<divid="menu">

<ul>

<liclass="current_page_item"><a href="index.html">Home</a></li>

<li><a href="C_Login.jsp">SERVER1</a></li>

<li><a href="KDC_Login.jsp">SERVER2</a></li>

<li><a href="U_Login.jsp">DATAOWNERORDATA

```

```

USER</a></li>

</ul>

</div>

<!--end #menu-->

<div="header">

  <div="logo">

<h1><a href="#">A FastNearest Neighbor Search Scheme over
Outsourced Encrypted Medical Images</a></h1>

<p>&nbsp;</p>

</div>

</div>

<!--end#header-->

<div="page">

<div="page-bgtop">

<div="page-bgbtm">

<div="content">

<divclass="post">

<divstyle="clear:both;">&nbsp;

<pclass="style4">A Fast Nearest Neighbor Search Scheme over
Outsourced Encrypted MedicalImages</p>

<p>&nbsp;</p>

</div>

<divclass="entry">

```

<p align="justify"class="style3">

Medical imaging is crucial for medical diagnosis, and the sensitive nature of medical Images necessitates rigorous security and privacy solutions to be in place. In a cloud based medical system for Healthcare Industry 4.0, medical images should be encrypted prior to being outsourced. However, processing queries over encrypted data without first executing the decryption operation is challenging and impractical at present. In the paper, we propose a secure and efficient scheme to find the exact nearest neighbour over encrypted medical images. Instead of calculating the Euclidean distance, we reject candidates by computing the lower bound of Euclidean distance that is related to the mean and standard deviation of data. Unlike most existing schemes, our scheme can obtain the exact nearest neighbour rather than an approximate result. We then evaluate our proposed approach to demonstrate its utility... </p>

</div>

</div>

<divclass="post"></div>

</div>

<!--end#conetet<divid="sidebar">

<h2>Menu</h2>

<liclass="current_page_item">HOME

SERVER1

```

<li><a href="KDC_Login.jsp">SERVER2</a></li>

<li><a href="U_Login.jsp">DATAOWNERORUSER</a></li>

</ul>

</li>

</ul>


</div>

</div>

</div>

</div>

<!--end#sidebar-->

<div style="clear: both;">&nbsp;</div>

<!--end#page-->

</div>

<!--end#footer-->

<div align="center"></div>

</body>

</html>

```

CONNECT.JSP

```

<% @page import="java.sql.*"%>C

connection connection = null;

    try        {

        Class.forName("com.mysql.jdbc.Driver");

```

connection

```
DriverManager.getConnection("jdbc:mysql://localhost:3306/fnns","root","");

    String sql="";

}

    catch(Exceptione){

        System.out.println(e);

    }

    %>
```

LOGIN.JSP

```
<title>AuthenticationPage</title>

<% @includefile="connect.jsp"%>

<% @pageimport="java.util.Date"%>

<%

    Stringname=request.getParameter("userid");

    String pass=request.getParameter("pass");

    try{

        andpass="" +pass+"";

        String sql="SELECT * FROM kdc where user="" +name+""Statementstmt =

        connection.createStatement();

        ResultSetsrs=stmt.executeQuery(sql); String utype="";

        if(rs.next()){

            response.sendRedirect("KDC_Main.jsp");
```

```

        }

        else

        {

            response.sendRedirect("KDC_Wrong_Login.jsp");}

        }

catch(Exceptione){

out.print(e);

}

%>

```

MAIN.JSP

```

<!DOCTYPEhtmlPUBLIC"-//W3C//DTD   XHTML   1.0

Strict//EN""http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<htmlxmlns="http://www.w3.org/1999/xhtml">

<head>

<metaname="keywords"content=""/>

<metaname="description"content=""/>

<metahttp-equiv="content-type"content="text/html;charset=utf-8"/>

<title>SERVER1MAIN</title>

<linkhref="style.css"rel="stylesheet"type="text/css"media="screen"/><scripttype="te
xt/javascript"src="js/cufon-yui.js"></script>

<scripttype="text/javascript"src="js/cufon-ptsans.js"></script>

<scripttype="text/javascript"src="js/jquery-1.4.2.min.js"></script>

```

```
<scripttype="text/javascript"src="js/script.js"></script>

<scripttype="text/javascript"src="js/coin-slider.min.js"></script>

<styletype="text/css">

<!--

.style3{color:#FF00FF}

-->

</style>

</head>

<body>

<divid="wrapper">

<divid="menu">

<ul>

<li><ahref="index.html">Home</a></li>

<liclass="current_page_item"><ahref="C_Login.jsp">SERVER1</a></li>

<li><ahref="KDC_Login.jsp">SERVER2</a></li>

<li><ahref="U_Login.jsp">DataOwnerORDATA
USERS</a></li>

</ul>

</div>

<!--end#menu-->

<divid="header">

<divid="logo">
```


<h1>A FastNearest Neighbor Search Schemeover Outsourced Encrypted
Medical Images</h1>

</div>

</div>

<!--end#header-->

<div id="page">

<div id="page-bgtop">

<div id="page-bgbtm">

<div id="content">

<div class="post">

<h2 class="title

style3">Welcometo

Server1</h2>

<p class="title

style3"></p>

</div>

</div>

<!--end#content>

<div id="sidebar">

<h2>Menu</h2>


```

<li>

<ul>

<li><a href="C_AuthorizeUsers.jsp">View All users
and
<li><a href="C_ViewImages.jsp">View All Medical
authorize</a></li>Images
<li><a href="C_UsersWithReposit.jsp">View Users
</a></li>Reposited keys
<li><a href="C_ViewAccessDetail.jsp">View All
</a></li>
Medical
60
Image access details</a></li>
<li><a href="C_AllAttacker.jsp">View All
attackers</a></li>
<li><a href="C_AllImagesRank.jsp">View Medical
Image rank in chart</a></li>at
<li><a href="C_AttackedNo.jsp">View count of
tacked chart</a></li>
<li><a href="C_Longin.jsp">Logout</a></li><br/>
</ul>
</li>

```

```
</ul>

</div>

<!--end#sidebar-->

</div>

</div>

</div>

<!--end#page-->

</div>

<!--end#footer-->

<div align=center></div>

</body>

</html>
```

CHAPTER-6

SYSTEM TESTING

6.1 TESTING METHODOLOGIES

The following are the Testing Methodologies' Unit Testing.

- o Integration Testing.
- o User Acceptance Testing.
- o Output Testing.
- o 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 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 individuals' 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 test each module individually and then each 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.

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 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 an error message.

Numeric Field:

The numeric field can contain only numbers from 0 to 9. An entry of any character flashes an error message. 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 inappropriately at a and produces an output revealing the errors in the system.

preparing of Test Data

Taking various kinds of test data does the above testing. Preparation of test data plays a virtual role in testing.

After preparing the test data the system under study is test dusting 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 system 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.

Them 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.

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 Demon started 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.

6.2 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.

CHAPTER-7

RESULTS

7.1 Main page:

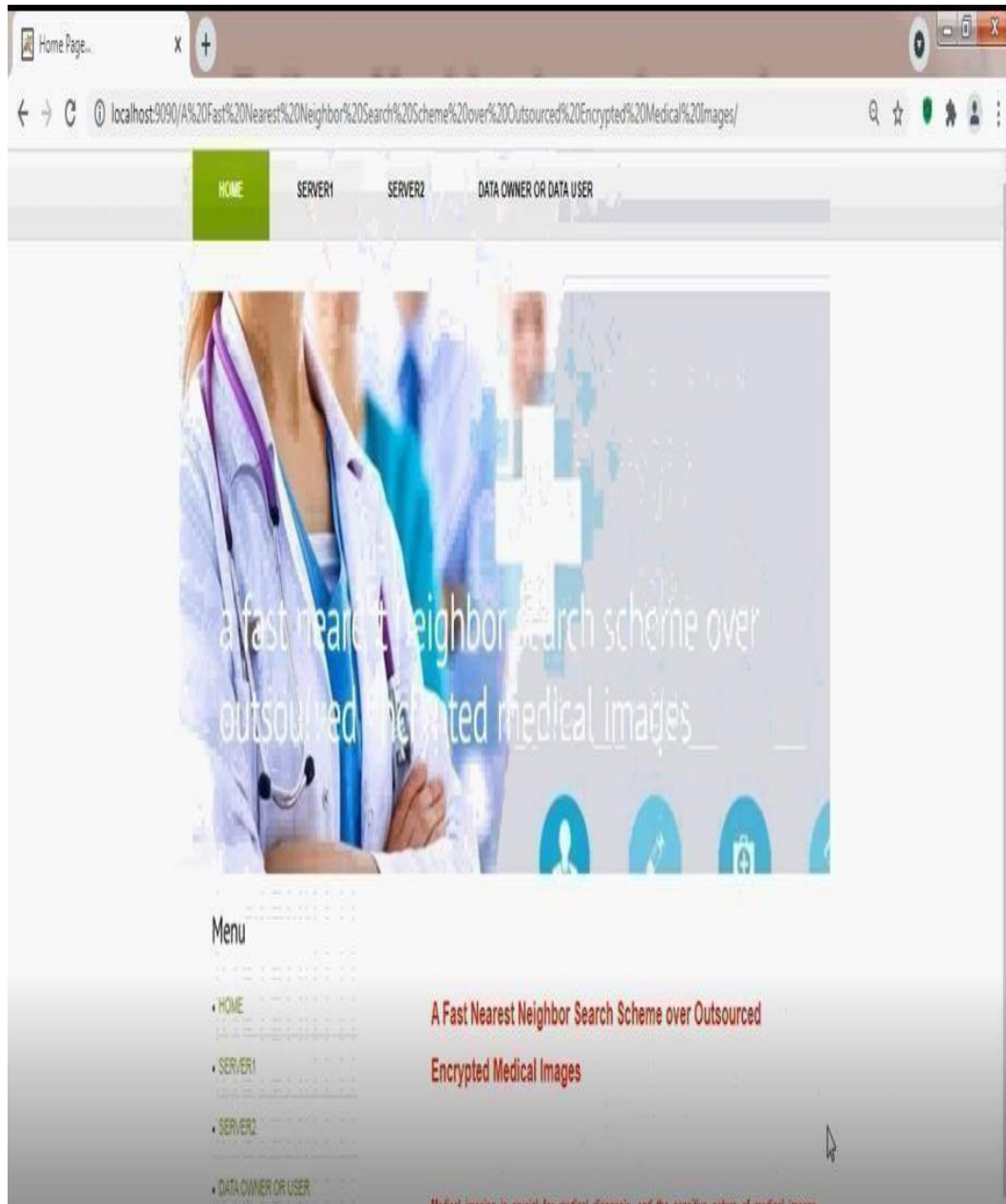


Fig-7.1:Main page

7.2 Server login

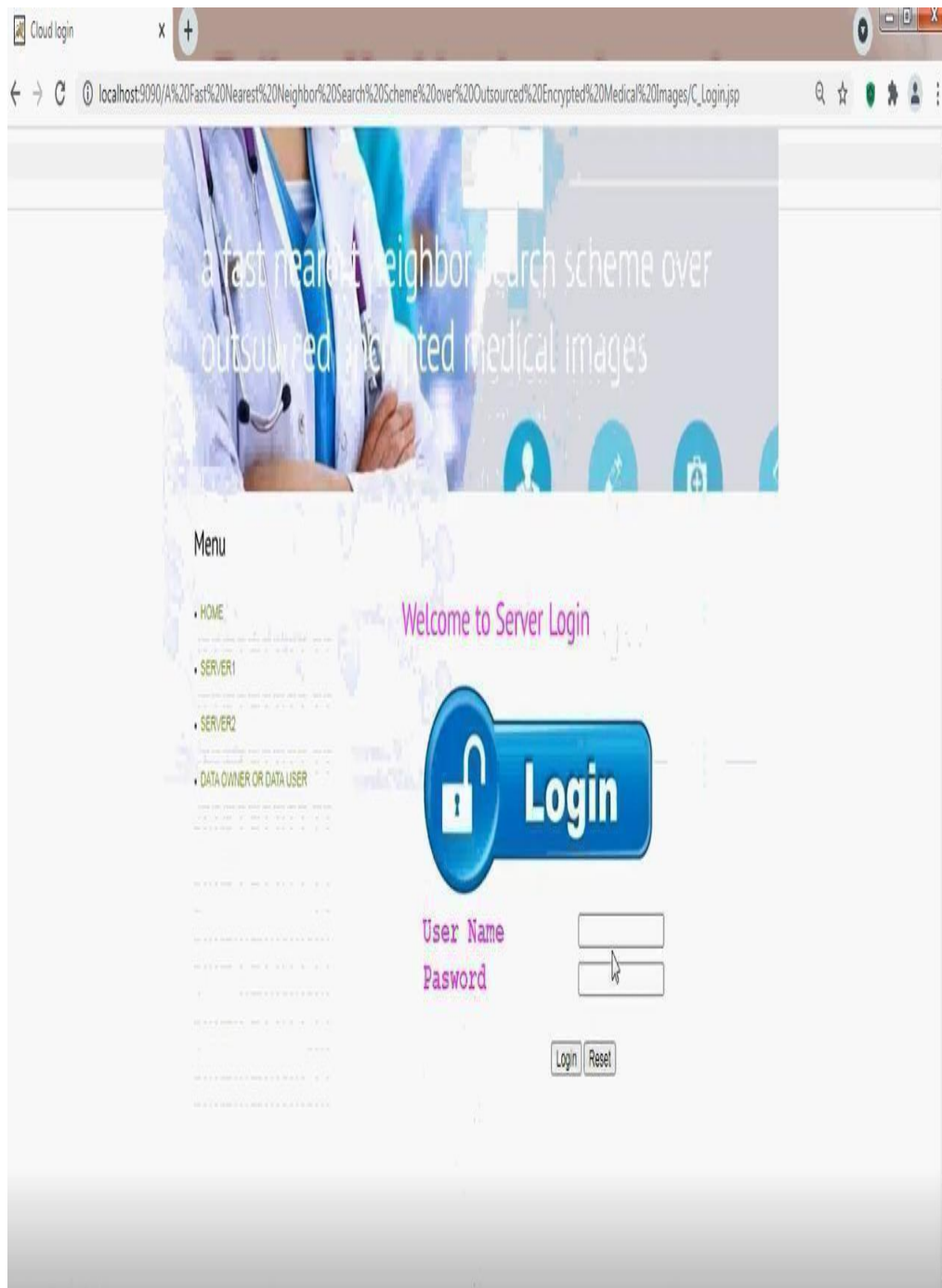


Fig-7.2:Server login

7.3Server1page

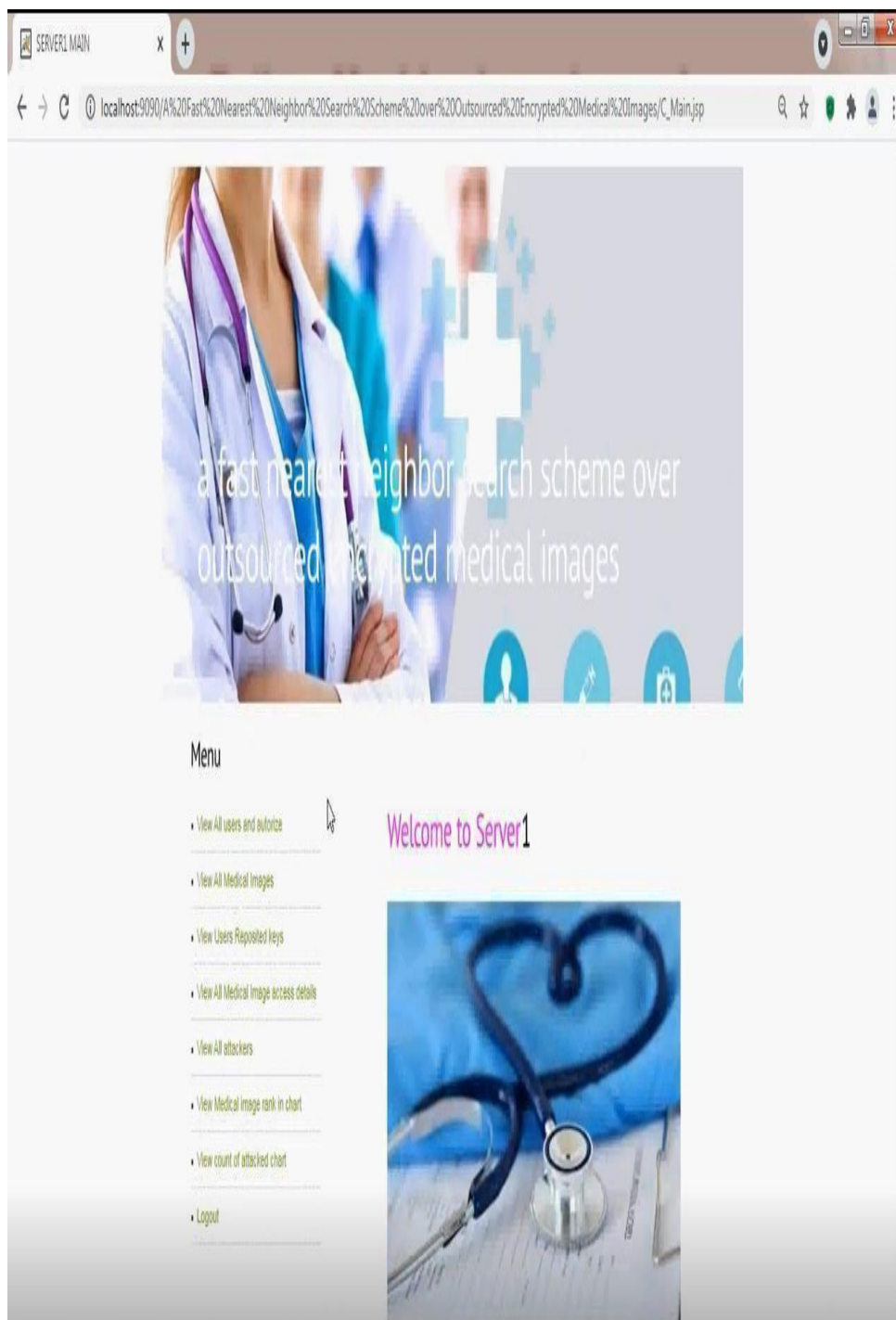





Fig-7.3:server1 page

7.4 Medical data

Cloud View Images

localhost:9090/A%2Ffast%20Nearest%20Neighbor%20Search%20Scheme%20over%20Outsourced%20Encrypted%20Medical%20Images/C_ViewImages.jsp

View All Medical Images

Image	Image Title	Image Owner	Image Tag	Description	Image Uses	date	Rank
	Heart	Gopi	Heart Block	V6hpcy8tZURpY2FsIG1tY W61IHuob3dtIHRoZS8oZU	to know about heart block	15/09/2021 15:36:52	4
	Brain	Gopi	Blood Clot in Brain	V6hpcy8tZURpY2FsIG1tY W61IHuob3dtIHRoZS8oZU	to know about blood clot in the brain	15/09/2021 15:14:19	0
	Liver	Seshank	Liver Cell Damage	V6hpcy8tZURpY2FsIG1tY W61IHuob3dtIHRoZS8oZU	to know about liver damage	15/09/2021 15:18:26	3

Rank

Fig-7.4:Medical page

7.5 Alluserspage

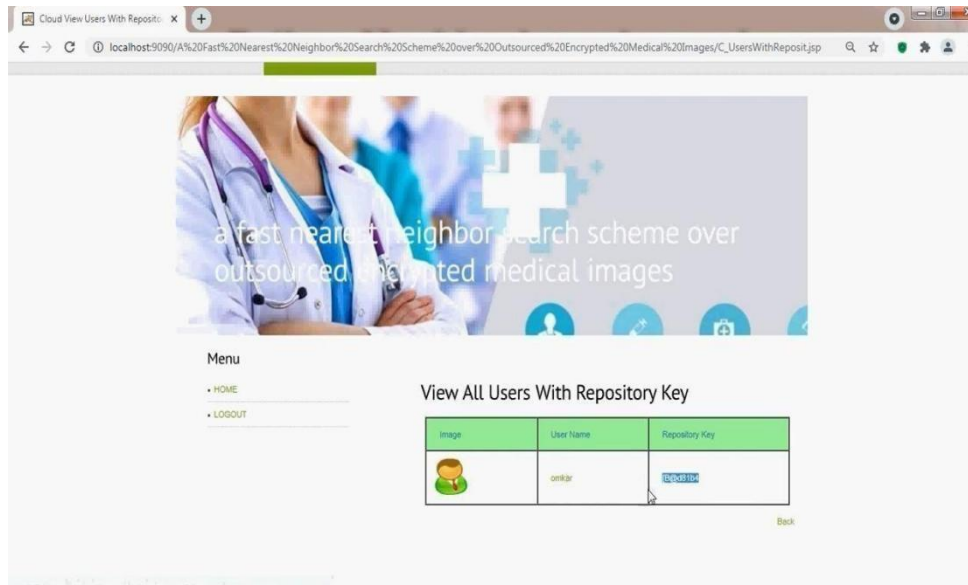


Fig-7.5:All user page

7.6:Attacker page

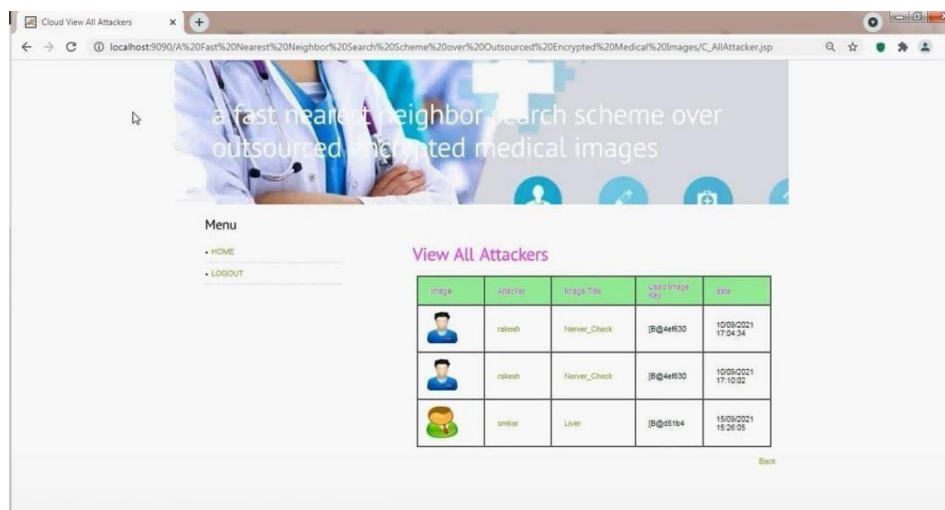


Fig-7.6:Attacker page

7.7 server 2

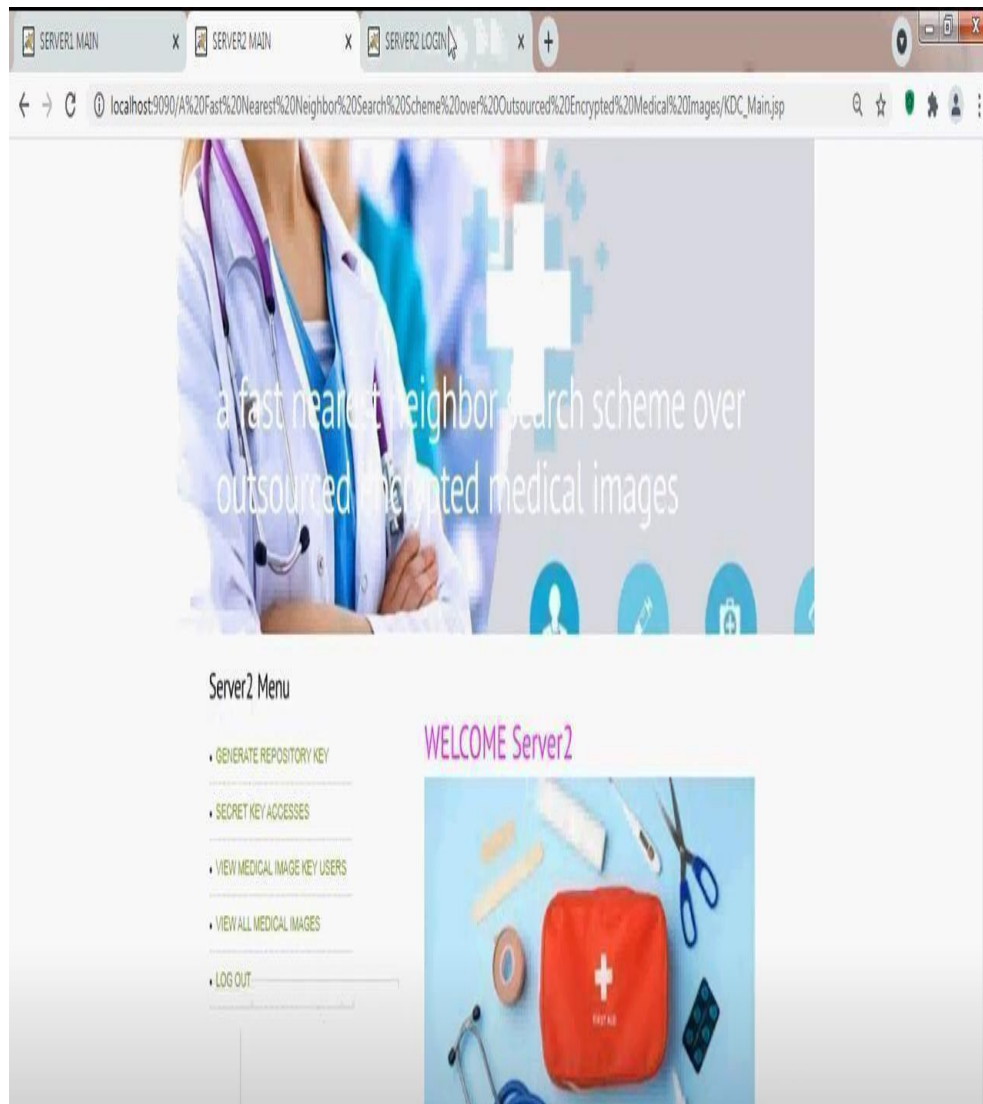


FIG-7.7: Server 2

7.8 User login page



Fig-7.8 :user login page

7.9 Dataowner



Fig-7.9:Data owner

7.10 User page

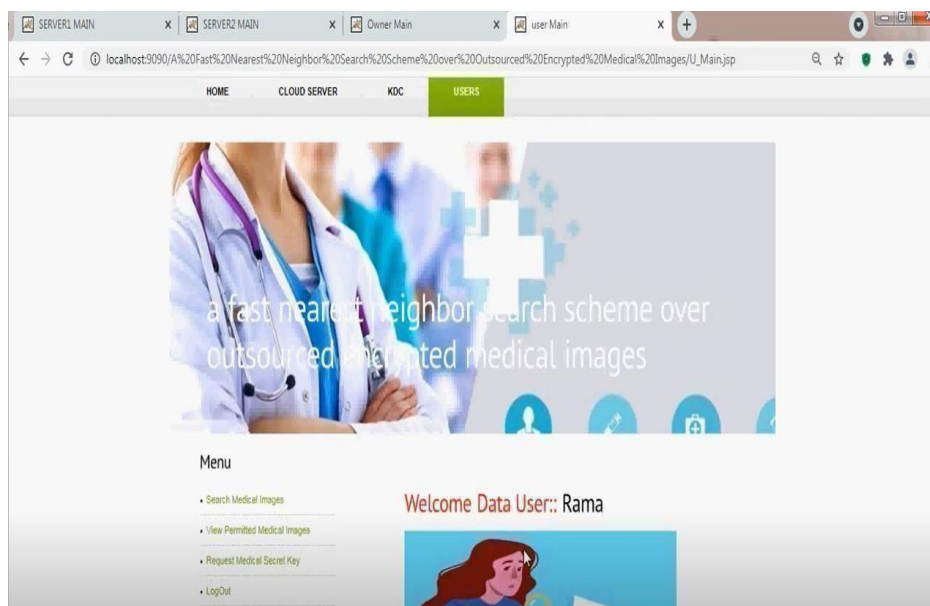


Fig-7.10 User page

7.11 View AllImage RankChart Results

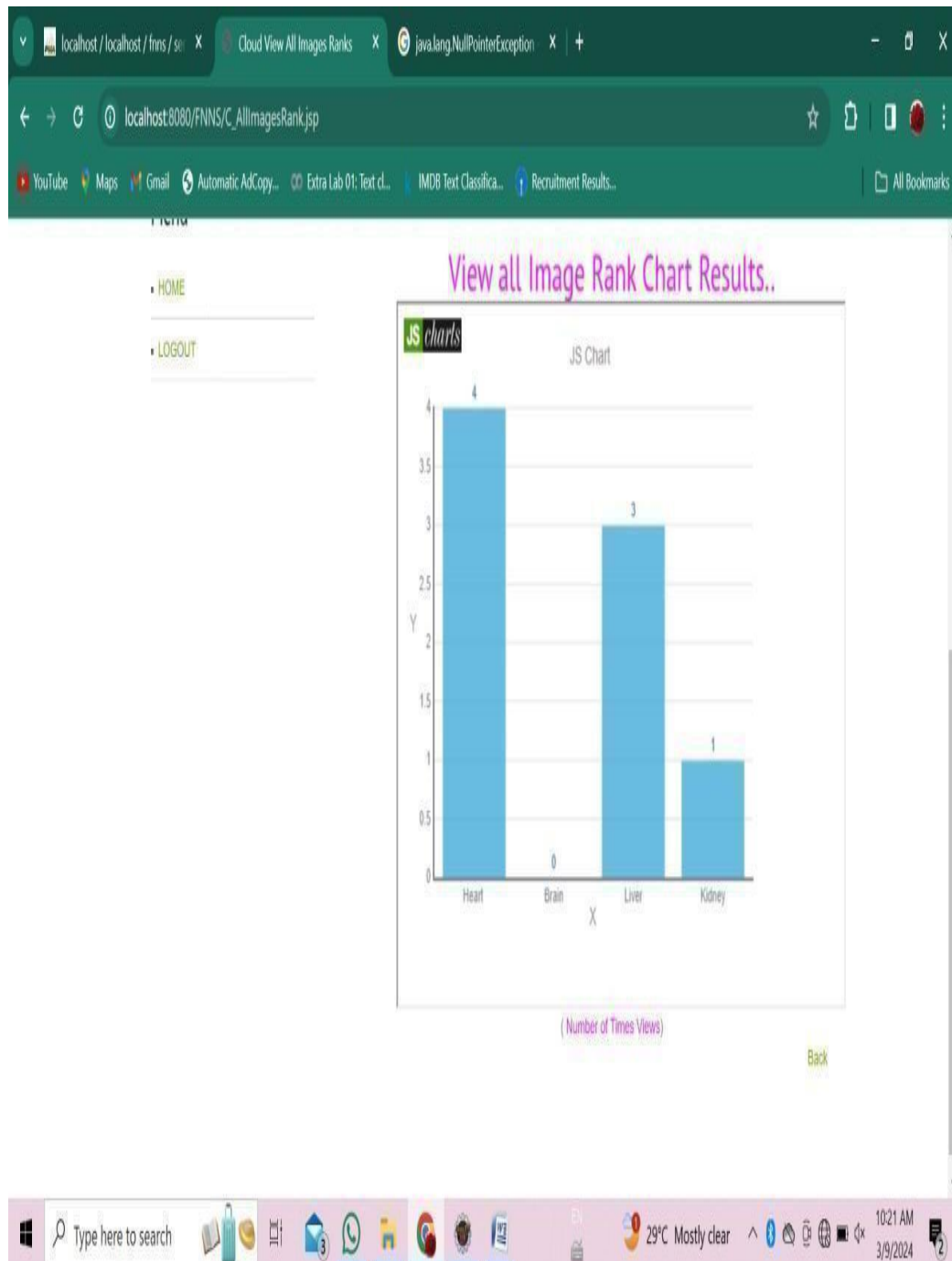


Fig-7.11:ViewAll Image Rank Chart Result

7.12.ViewallimageAttackedcountchart

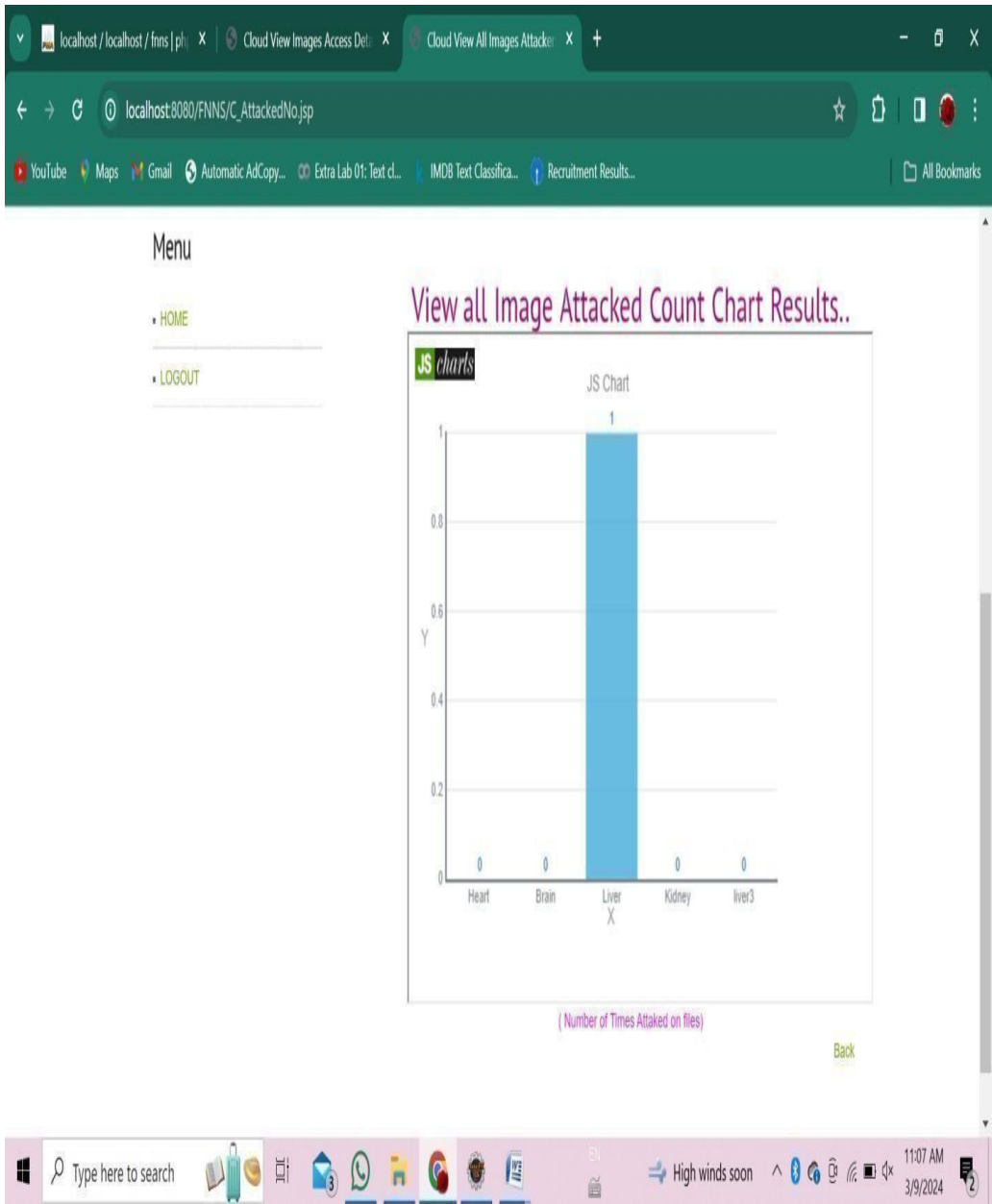


Fig-8.12:ViewallimageAttackedcountchart

CHAPTER-8

CONCLUSION

Cloud-based electronic healthcare systems will be increasingly popular, particularly due to the capability to share and access data in real-time across organizations (e.g., between medical practitioners and healthcare providers) and countries. One process becomes challenging, if not impractical.

In the paper, we presented a secure and efficient scheme to locate the exact nearest neighbour over encrypted medical images stored in the remote cloud server. For the purpose of rejecting candidate data points, our scheme securely computes the lower bound of the squared Euclidean distance between a data point in the database and the query submitted by a legitimate user. The performance of our scheme is evaluated using real-world medical images.

Future research includes finding a real-world healthcare organization to design and implement a prototype of our proposed approach. This will allow us to evaluate the real-world utility of the proposed system as well as its scalability in practice. In addition, it will also allow us to identify weaknesses / limitations, if any, that we are not aware of.

CHAPTER-9

FUTURE ENHANCEMENT

Some potential future enhancements for our project could include:

1 *Improved Efficiency*: Continuously optimizing the algorithm to make the nearest neighbor search even faster and more resource-efficient.

2 *Enhanced Security Measures*: Implementing additional security measures to further protect the encrypted medical images and the search process, such as advanced encryption techniques or multi-factor authentication.

3 *Scalability*: Designing the system to handle a larger volume of medical images and user queries as the database grows over time.

4 *User Interface Enhancements*: Creating a user-friendly interface for easier interaction with the system, allowing healthcare professionals to search for and retrieve medical images more intuitively.

5 *Integration with Other Healthcare Systems*: Integrating your system with existing healthcare databases or systems to facilitate seamless access to medical images and improve interoperability within the healthcare ecosystem.

6 *Machine Learning Integration*: Exploring the integration of machine learning techniques to enhance the accuracy of image similarity matching and provide more relevant search results.

7 *Support for Additional Data Types*: Extending the system to support other types of medical data beyond images, such as patient records or diagnostic reports, to provide a more comprehensive healthcare management solution.

8 *Collaboration Tools*: Adding collaboration features that allow multiple healthcare professionals to securely share and annotate medical images, fostering better communication and collaboration in diagnosis and treatment planning.

REFERENCES

- [1] J. Li, L. Huang, Y. Zhou, S. He, Z. Ming, "Computation partitioning for mobile cloud computing in big data environment," *IEEE Trans. Ind. Informat.*, vol. 13, no. 4, pp. 2009–2018, Feb. 2017.
- [2] K.-K. R. Choo, "Cloud computing: Challenges and future directions," *Trends & Issues in Crime and Criminal Justice*, vol. 400, no. 400, pp. 1–6, Oct. 2010.
- [3] M. Pajic, R. Mangharam, O. Sokolsky, D. Arney, J. M. Goldman, and I. Lee, "Model-driven safety analysis of closed-loop medical systems," *IEEE Trans. Ind. Informat.*, vol. 10, no. 1, pp. 3–16, Feb. 2014.
- [4] B. Xu, L. D. Xu, H. Cai, C. Xie, J. Hu, and F. Bu, "Ubiquitous data accessing method in IoT-based information system for emergency medical services," *IEEE Trans. Ind. Informat.*, vol. 10, no. 2, pp. 1578–1586, May. 2014.
- [5] G. Yang *et al.*, "A health-IoT platform based on the integration of intelligent packaging, unobtrusive bio-sensor, and intelligent medicine box," *IEEE Trans. Ind. Informat.*, vol. 10, no. 4, pp. 2180–2191, Nov. 2014.
- [6] H. Huang, T. Gong, N. Ye, R. Wang, and Y. Dou, "Private and Secured Medical Data Transmission and Analysis for Wireless Sensing Healthcare System," *IEEE Trans. Ind. Informat.*, vol. 13, no. 3, pp. 1227–1237, June. 2017.
- [7] M. Li, S. Yu, W. Lou, and Y. T. Hou, "Toward privacy-assured cloud data services with flexible search functionalities," in *Proc. ICDCSW/IEEE*, Macau, CHN, 2012, pp. 466–470.
- [8] P. Williams, R. Sion, and B. Car bunar, "Building castles out of mud: practical access pattern privacy and correctness on untrusted storage," in *Proc. CCS. ACM*, Alexandria, VA, USA, 2008, pp. 139–148.
- [9] M. S. Islam, M. Kuzu, and M. Kantarcioglu, "Access pattern disclosure on searchable encryption: Ramification, attack and mitigation," in *NDSS*, San Diego, CA, USA, 2012.
- [10] D. E. Knuth, "Sorting and searching," in *The art of computer programming*, vol. 3, Boston, USA: Addison-Wesley, 1973.
- [11] D. Song, D. Wagner, and A. Perrig, "Practical Techniques for Searches on Encrypted Data," in *Proc. of IEEE S&P*, DC, USA, 2000, pp. 44–55.

- [12] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: improved definitions and efficient constructions," *J. Comput. Secur.*, vol.19, no.5, pp.895-934, 2011.
- [13] S. Kamara, C. Papamanthou, and T. Roeder, "Dynamic Searchable Symmetric Encryption," in *Proc. of ACM CCS*, Raleigh, NC, USA, 2012, pp. 965–976.
- [14] S. Kamara, C. Papamanthou, "Parallel and dynamic searchable symmetric encryption," *Financial Cryptography and Data Security*, Springer Berlin Heidelberg, 2013, pp.258-274.
- [15] G.S.Poh, J.-J.Chin, W.-C.Yau, K.-K.R.Choo, and M.S. Mohamad, "Searchable Symmetric Encryption: Designs and Challenges," *ACM Comput. Surv.* vol.50, no.3, pp.40:1-40:37, 2017.
- [16] W. Wong, D. W. Cheung, B. Kao, and N. Mamoulis, "Secure KNN Computation on Encrypted Data bases," in *Proc. ACM SIGMOD*, Providence, RI, USA, 2009, pp.139–152.
- [17] H. Hu, J. Xu, C. Ren, and B. Choi, "Processing Private Queries over Untrusted Data Cloud through Privacy Homomorphism," in *Proc. IEEE ICDE*, Hannover, NI, GER, 2011, pp.601–612.
- [18] Y. Zhu, Z. Huang, and T. Takagi, "Secure and controllable k-NN query over encrypted cloud data with key confidentiality," *Journal of Parallel & Distributed Computing*, vol.89, pp.1-12, Mar.2016.
- [19] L. Zhou, Y. Zhu, and A. Castiglione, "Efficient k-NN query over encrypted data in cloud with limited key-disclosure and offline data owner," *Computers & Security*, vol.69, pp.84-96, Aug.2017.
- [20] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *Proc. STOC*, Dallas, TX, USA, 1998, pp.604–613.
- [21] M. Datar, N. Immorlica, P. Indyk, and V.S. Mirrokni, "Locality sensitive hashing scheme based on p-stable distributions," in *Proc. SCG.ACM*, Brooklyn, NY, USA, 2004, pp.253–262.
- [22] A. Andoni, P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," *IEEE Symposium on Foundations of Computer Science*, Berkeley, CA, USA, 2006, pp.459-468.
- [23] J.L.Bentley, "Multidimensional binary search trees used for associative searching," *Communications of ACM*, vol.18, no.9, pp.509–517, Sep.1975.
-

- [24] H.V.Jagadish, B.C.Ooi, K.-L.Tan, C.Yu, and R. Zhang, "IDistance: An adaptive B+-tree based indexing method for nearest neighbor search," *ACM Trans. Database Syst.*, vol.30, no.2, pp.364–397, June.2005.
- [25] S.Arya, D.Mount, N. Netanyahu, R.Silverman, and A.Wu, "A noptimal algorithm for approximate nearest neighbor searching fixed dimensions," *Journal of the ACM*, vol.45, no.6, 1998, pp.891–923.
- [26] S. Bercht old, D. A. Keim, and H. P. Kriegel, "The x-tree: An index structure for high-dimensional data," in *Proc. VLDB Conf.*, Mumbai (Bombay), India, 1996, pp. 28–39.