A Proxy Re-Encryption Approach to Secure Data Sharing in the Internet of Things Based on Blockchain

**A Proxy Re-Encryption Approach to Secure Data Sharing in the Internet of Things Based on Blockchain**

**A Project Report submitted in partial fulfillment of the degree of the   
Bachelor of Technology in Computer Science and Engineering**

**By**

**Mateti Meghana 20C45A0515 (Matetimeghana2001@gmail.com)**

**Pendam Yashonath 19C41A05H1 (pyashonath3@gmail.com)**

**Vanga Vishwa Teja 19C41A05E8 (vishwateja2026@gmail.com**

**Pendli Mahendher 19C41A05H3 (mahenderpendli26@gmail.com)**

**Under the Guidance of**

**N. Shiva Prasad**

**Assistant Professor**



Department of Computer Science and Engineering

**Jayamukhi institute of technological Sciences**

**Narsampet, warangal-506 332**

**(Affiliated to JNTUH, Accredited by NAAC ‘A’ Grade)**

**2022-2023**

**Jayamukhi institute of technological Sciences**

**Narsampet, warangal – 506 332**

**(Affiliated to JNTUH, Accredited by NAAC ‘A’ Grade)**

**CERTIFICATE**



This is to certify that the Project Report entitled “**A Proxy Re-Encryption Approach to Secure Data Sharing in the Internet of Things Based on Blockchain**” is a bona fide work of the students **Mateti Meghana, Pendam Yashonath, Vanga Vishwa Teja, Pendli Mahendher** bearing Roll No.s **20C45A0515, 19C41A05H1, 19C41A05E8, 19C41A05H3** submitted in partial fullfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering** during the academic year **2022-23.**

**Guide Head of the Department**

**Principal**

**ACKNOWLEDGEMENT**

We express our immense gratitude and sincere thanks to **Prof. V. JANAKI**, principal, Jayamukhi Institute of Technological Sciences, Narsampet and Warangal for her constant and gracious guidance throughout the work.

We express our gratitude and sincere thanks to **Dr. P. SRINIVAS RAO**, Head of the Department Of CSE, Jayamukhi Institute of Technological Sciences for his constant motivation and support.

We are grateful to our Project Guide **Mr. N.SHIVA PRASAD**, Assistant professor for his Valuable guidance and giving us the opportunity to do our work in a better manner.

We are also thankful to our Management for providing all the facilities in completing the project.

**ABSTRACT**

The evolution of the Internet of Things has seen data sharing as one of its most useful applications in cloud computing. As eye-catching as this technology has been, data security remains one of the obstacles it faces since the wrongful use of data leads to several damages. In this article, we propose a proxy re-encryption approach to secure data sharing in cloud environments. Data owners can outsource their encrypted data to the cloud using identity-based encryption, while proxy re-encryption construction will grant legitimate users access to the data. With the Internet of Things devices being resource-constrained, an edge device acts as a proxy server to handle intensive computations. Also, we make use of the features of information-centric networking to deliver cached content in the proxy effectively, thus improving the quality of service and making good use of the network bandwidth. Further, our system model is based on blockchain, a disruptive technology that enables decentralization in data sharing. It mitigates the bottlenecks in centralized systems and achieves fine-grained access control to data. The security analysis and evaluation of our scheme show the promise of our approach in ensuring data confidentiality, integrity, and security.

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| **1.** | **INTRODUCTION** | **1-2** |
| **2.** | **LITERATURE REVIEW** | **3-8** |
| **3.** | **EXISTING SYSTEM** | **9** |
| **4.** | **PROPOSED SYSTEM** | **10** |
| **5.** | **SYSTEM ANALYSIS** | **11** |
| **6.** | **SYSTEM ENVIRONMENT** | **12-28** |
| **7.** | **SYSTEM STUDY**  7.1 Feasibility Study  7.2 Economical Feasibility  7.3 Technical Feasibility | **29-30** |
| **8.** | **SYSTEM DESIGN**  8.1 System Architecture  8.2 Data Flow Diagram  8.3 Uml Diagrams  8.4 Use Case Diagram  8.5 Class Diagram  8.6 Sequence Diagram  8.7 Activity Diagram | **31-39** |
| **9.** | **SYSTEM REQUIREMENTS**  9.1 Hardware Requirements  9.2 Software Requirements | **40** |
| **10.** | **SYSTEM TESTING**  10.1 Unit Testing 10.2 Integration Testing 10.3 Acceptance Testing  10.4 Test Results | **41-43** |
| **11.** | **IMPLEMENTATION** | **44** |
| **12.** | **INPUT & OUTPUT SYSTEM**  12.1 Input Design  12.2 Output Design | **45-46** |
| **13.** | **USER MANUAL AND OUTPUT SCREENS** | **47-56** |
| **14.** | **CONCLUSION** | **57** |
| **15.** | **BIBLIOGRAPHY** | **58-59** |

**1. INTRODUCTION**

**1.1 Introduction to Proxy Re-encryption Approach For Secure Data Sharing in IOT using Block Chain**

The Internet of Things (IoT) has emerged as a technology that has great significance to the world nowadays and its utilization has given rise to an expanded growth in network traffic volumes over the years. It is expected that a lot of devices will get connected in the years ahead. Data is a central notion to the IoT paradigm as the data collected serves several purposes in applications such as healthcare, vehicular networks, smart cities, industries, and manufacturing, among others [1]. The sensors measure a host of parameters that are very useful for stakeholders involved. Consequently, as enticing as IoT seems to be, its advancement has introduced new challenges to security and privacy. IoT needs to be secured against attacks that hinder it from providing the required services, in addition to those that pose threats to the confidentiality, integrity, and privacy of data. A viable solution is to encrypt the data before outsourcing to the cloud servers. Attackers can only see the data in its encrypted form when traditional security measures fail. In data sharing, any information must be encrypted from the source and only decrypted by authorized users in order to preserve its protection. Conventional encryption techniques can be used, where the decryption key is shared among all the data users designated by the data owner. The use of symmetric encryption implies that the same key is shared between the data owner and users, or at least the participants agree on a key. This solution is very inefficient. Furthermore, the data owners do not know in advance who the intended data users are, and, therefore, the encrypted data needs to be decrypted and subsequently encrypted with a key known to both the data owner and the users. This decrypt-and-encrypt solution means the data owner has to be online all the time, which is practically not feasible. The problem becomes increasingly complex when there are multiple pieces of data and diverse data owners and users. Although simple, the traditional encryption schemes involve complex key management protocols and, hence, are not apt for data sharing. Proxy re-encryption (PRE), a notion first proposed by Blaze et al. [2], allows a proxy to transform a file computed under a delegator’s public key into an encryption intended for a delegate. Let the data owner be the delegator and the data user be the delegate. In such a scheme, the data owner can send encrypted messages to the user temporarily without revealing his secret key. The data owner or a trusted third party generates the re-encryption key. A proxy runs the re-encryption algorithm with the key and revamps the ciphertext before sending the new ciphertext to the user. An intrinsic trait of a PRE scheme is that the proxy is not fully trusted (it has no idea of the data owner’s secret key). This is seen as a prime candidate for delegating access to encrypted data in a secured manner, which is a crucial component in any data-sharing scenario. In addition, PRE allows for encrypted data in the cloud to be shared to authorized users while maintaining its confidentiality from illegitimate parties.

### 2. LITERATURE REVIEW

**LITERATURE[1]** : Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications.

**AUTHOR’S**: Ala Al-Fuqaha, M. Guizani , M.Mohammadi, M. Aledhari, M.Ayyash.

**YEAR OF PUBLISH** : OCT/DEC 2015.

**DESCRIPTION :**

The IoT is enabled by the latest developments in RFID, smart sensors, communication technologies, and Internet protocols. The basic premise is to have smart sensors collaborate directly without human involvement to deliver a new class of applications. The current revolution in Internet, mobile, and machine-to-machine (M2M) technologies can be seen as the first phase of the IoT. In the coming years, the IoT is expected to bridge diverse technologies to enable new applications by connecting physical objects together in support of intelligent decision making. This paper starts by providing a horizontal overview of the IoT. Then, we give an overview of some technical details that pertain to the IoT enabling technologies, protocols, and applications. Compared to other survey papers in the field, our objective is to provide a more thorough summary of the most relevant protocols and application issues to enable researchers and application developers to get up to speed quickly on how the different protocols fit together to deliver desired functionalities without having to go through RFCs and the standards specifications. We also provide an overview of some of the key IoT challenges presented in the recent literature and provide a summary of related research work. Moreover, we explore the relation between the IoT and other emerging technologies including big data analytics and cloud and fog computing. We also present the need for better horizontal integration among IoT services. Finally, we present detailed service use-cases to illustrate how the different protocols presented in the paper fit together to deliver desired IoT services.

**LITERATURE[2]:** Developing Information Networking Further: From PSIRP To PURSUIT.

**AUTHOR’S:** N Fotiou, P Nikander, D Trossen , George C. Polyzos.

**YEAR OF PUBLISH** : Oct .2010.

**DESCRIPTION :**

PSIRP (Publish-Subscribe Internet Routing Paradigm) is an EU FP7 funded project that has developed a clean-slate architecture for the future Internet, based on the publish-subscribe primitives (rather than the send-receive ones), all the way down to the core networking functions. The PSIRP vision is a pure information-centric Internet architecture, possibly providing remedies to many of the current Internet problems. In PSIRP, all is information and everything is about information. Content-based identities, recursive application of ideas, cryptographic techniques, and the Trust-to-Trust principle are all extensively used to achieve the design goals. Furthermore, incentive compatibility and socio-economic considerations are guiding the design from the outset, to ground the project in reality and to provide credible and viable potential deployment paths. The project has developed, implemented, and preliminarily evaluated solutions for rendezvous, topology formation and routing, and information forwarding, with ongoing work currently focusing in experimenting. A new (also EU FP7 funded) follow-on project, PURSUIT (PublishSubscribe Internet Technologies), will refine and further explore and expand PSIRP’s vision. We believe that this will eventually lead to a more complete architecture and protocol suite, thereby providing for more extensive performance evaluation and investigations on scalability. This paper provides an overview of the PSIRP concepts and the developed architecture, along with some key results, and outlines the research directions of the PURSUIT project, focusing on the project goals and its expected outcomes.

**LITERATURE[3] :** Secure Naming For A Network Of Information.

**AUTHOR’S:** C Dannewitz , J Golic , B Ohlman, B Ahlgren.

**YEAR OF PUBLISH :** 2010

**DESCRIPTION :**

Several projects propose an information-centric approach to the network of the future. Such an approach makes efficient content distribution possible by making information retrieval host-independent and integrating into the network storage for caching information. Requests for particular content can, thus, be satisfied by any host or server holding a copy. The current security model based on host authentication is not applicable in this context. Basic security functionality must instead be attached directly to the data and its naming scheme. A naming scheme to name content and other objects that enables verification of data integrity as well as owner authentication and identification is here presented. The naming scheme is designed for flexibility and extensibility, e.g., to integrate other security properties like access control. At the same time, the naming scheme offers persistent IDs even though the content, content owner and/or owner's organizational structure, or location change. The requirements for the naming scheme and an analysis showing how the proposed scheme fulfills them are presented. Experience with prototyping the naming scheme is also discussed. The naming scheme builds the foundation for a secure information-centric network infrastructure that can also solve some of the main security problems of today's Internet.

**LITERATURE[4]**: Building An Encrypted And Searchable Audit Log.

**AUTHOR’S:** B.R.Waters , D Balfanz , G Durfee , D. K. Smetters.

**Y EAR OF PUBLISH :** Feb 2004.

**DESCRIPTION :**

Audit logs are an important part of any secure system, and they need to be carefully designed in order to give a faithful representation of past system activity. This is especially true in the presence of adversaries who might want to tamper with the audit logs. While it is important that auditors can inspect audit logs to assess past system activity, the content of an audit log may contain sensitive information, and should therefore be protected from unauthorized parties. Protecting the contents of audit logs from unauthorized parties (i.e., encrypting it), while making it efficiently searchable by authorized auditors poses a problem. We de- scribe an approach for constructing searchable encrypted audit logs which can be combined with any number of existing approaches for creating tamper-resistant logs. In particular, we implemented an audit log for database queries that uses hash chains for integrity protection and identity- based encryption with extracted keywords to enable searching on the encrypted log. Our technique for keyword search on encrypted data has wide application beyond searchable audit logs.

**LITERATURE [5]:** Identity-Based Cryptosystems And Signature Schemes.

**AUTHOR :** A Shamir.

**YEAR OF PUBLISH** : Aug 1984.

**DESCRIPTION :**

We introduce a novel type of cryptographic scheme, which enables any pair of users to communicate securely and to verify each other’s signatures without exchanging private or public keys, without keeping key directories, and without using the services of a third party. The scheme assumes the existence of trusted key generation centers, whose sole purpose is to give each user a personalized smart card when he first joins the network. The information embedded in this card enables the user to sign and encrypt the messages he sends and to decrypt and verify the messages he receives in a totally independent way, regardless of the identity of the other party. Previously issued cards do not have to be updated when new users join the network, and the various centers do not have to coordinate their activities or even to keep a user list. The centers can be closed after all the cards are issued, and the network can continue to function in a completely decentralized way for an indefinite period.

**LITERATURE[6] :** Chosen-Ciphertext Security From Identity-Based Encryption.

**AUTHOR’S :** R Canetti, S Halevi , J Katz.

**YEAR OF PUBLISH :** 2004

**DESCRIPTION :**

We propose a simple and efficient construction of a CCAsecure public-key encryption scheme from any CPA-secure identity-based encryption (IBE) scheme. Our construction requires the underlying IBE scheme to satisfy only a relatively “weak” notion of security which is known to be achievable without random oracles; thus, our results provide a new approach for constructing CCA-secure encryption schemes in the standard model. Our approach is quite different from existing ones; in particular, it avoids non-interactive proofs of “well-formedness” which were shown to underlie most previous constructions. Furthermore, applying our conversion to some recently-proposed IBE schemes results in CCA-secure schemes whose efficiency makes them quite practical. Our technique extends to give a simple and reasonably efficient method for securing any binary tree encryption (BTE) scheme against adaptive chosen-ciphertext attacks. This, in turn, yields more efficient CCA-secure hierarchical identity-based and forward-secure encryption schemes in the standard model.

**LITERATURE[7] :** Improving Security And Efficiency In Attribute-Based Data Sharing.

**AUTHOR :** Junbeom Hur

**YEAR OF PUBLISH :** 05 April 2011

**DESCRIPTION :**

With the recent adoption and diffusion of the data sharing paradigm in distributed systems such as online social networks or cloud computing, there have been increasing demands and concerns for distributed data security. One of the most challenging issues in data sharing systems is the enforcement of access policies and the support of policies updates. Ciphertext policy attribute-based encryption (CP-ABE) is becoming a promising cryptographic solution to this issue. It enables data owners to define their own access policies over user attributes and enforce the policies on the data to be distributed. However, the advantage comes with a major drawback which is known as a key escrow problem. The key generation center could decrypt any messages addressed to specific users by generating their private keys. This is not suitable for data sharing scenarios where the data owner would like to make their private data only accessible to designated users. In addition, applying CP-ABE in the data sharing system introduces another challenge with regard to the user revocation since the access policies are defined only over the attribute universe. Therefore, in this study, we propose a novel CP-ABE scheme for a data sharing system by exploiting the characteristic of the system architecture. The proposed scheme features the following achievements:

1) the key escrow problem could be solved by escrow-free key issuing protocol, which is constructed using the secure two-party computation between the key generation center and the data-storing center, and

2) fine-grained user revocation per each attribute could be done by proxy encryption which takes advantage of the selective attribute group key distribution on top of the ABE. The performance and security analyses indicate that the proposed scheme is efficient to securely manage the data distributed in the data sharing system

**LITERATURE[8] :** Time-based proxy re-encryption scheme for secure data sharing in a cloud environment

**AUTHOR’S :** Qin Liu, Guojun Wang, Jie Wu.

**YEAR OF PUBLISH :** February 2014

**DESCRIPTION :**

A fundamental approach for secure data sharing in a cloud environment is to let the data owner encrypt data before out souring. To simultaneously achieve fine-grained access control on encrypted data and scalable user revocation, existing work combines attribute-based encryption (ABE) and proxy re-encryption (PRE) to delegate the cloud service provider (CSP) to execute re-encryption. However, the data owner should be online in order to send the PRE keys to the CSP in a timely fashion, to prevent the revoked user from accessing the future data. The delay of issuing the PRE keys may cause potential security risks. In this paper, we propose a time-based proxy re-encryption (TimePRE) scheme to allow a user's access right to expire automatically after a predetermined period of time. In this case, the data owner can be offline in the process of user revocations. The basic idea is to incorporate the concept of time into the combination of ABE and PRE. Specifically, each data is associated with an attribute-based access structure and an access time, and each user is identified by a set of attributes and a set of eligible time periods which denote the period of validity of the user's access right. Then, the data owner and the CSP are required to share a root secret key in advance, with which CSP can automatically update the access time of the data with the time that it receives a data access request. Therefore, given the re-encrypted ciphertext, only the users whose attributes satisfy the access structure and whose access rights are effective in the access time can recover corresponding data.

**3.EXISTING SYSTEM**

The problem becomes increasingly complex when there are multiple pieces of data and diverse data owners and users Data is a central notion to the IoT paradigm as the data collected serves several purposes in applications such as healthcare, vehicular networks …etc and manufacturing, among others. The sensors measure a host of parameters that are very useful for stakeholders involved. Consequently, as enticing as IoT seems to be, its advancement has introduced new challenges to security and privacy. IoT needs to be secured against attacks that hinder it from providing the required services, in addition to those that pose threats to the confidentiality, integrity, and privacy of data.

**4.PROPOSED SYSTEM**

The traditional encryption schemes involve complex key management protocols and, hence, are not apt for data sharing. Proxy re-encryption (PRE), a notion first proposed by Blaze et al., allows a proxy to transform a file computed under a delegator’s public key into an encryption intended for a delegate. Let the data owner be the delegator and the data user be the delegate. In such a scheme, the data owner can send encrypted messages to the user temporarily without revealing his secret key. The data owner or a trusted third party generates the re-encryption key. A proxy runs the re-encryption algorithm with the key and revamps the ciphertext before sending the new ciphertext to the user. An intrinsic trait of a PRE scheme is that the proxy is not fully trusted (it has no idea of the data owner’s secret key). This is seen as a prime candidate for delegating access to encrypted data in a secured manner, which is a crucial component in any data-sharing scenario. In addition, PRE allows for encrypted data in the cloud to be shared to authorized users while maintaining its confidentiality from illegitimate parties.

Motivated by this scenario, this article proposes an improvement in IoT data sharing by combining PRE with identity-based encryption (IBE), information-centric networking (ICN), and blockchain technology. Shamir first presented the notion of IBE, in which a sender encrypts a message to a recipient using the identity (email ) as the public key. It is a very powerful primitive used to combat numerous key distribution problems and has consented to the development of several cryptographic protocols, including public-key searchable encryption , secret handshakes , and chosen ciphertext attack (CCA) secure public-key encryption . IBE is preferred over attribute-based encryption (ABE) because ABE involves heavy computations on data encryption, decryption, and key management, and these processes are not convenient for the resource-constrained IoT devices.

**5.SYSTEM ANALYSIS**

A viable solution is to encrypt the data before outsourcing to the cloud servers. Attackers can only see the data in its encrypted form when traditional security measures fail. In data sharing, any information must be encrypted from the source and only decrypted by authorized users in order to preserve its protection. Conventional encryption techniques can be used, where the decryption key is shared among all the data users designated by the data owner. The use of symmetric encryption implies that the same key is shared between the data owner and users, or at least the participants agree on a key. This solution is very inefficient. Furthermore, the data owners do not know in advance who the intended data users are, and, therefore, the encrypted data needs to be decrypted and subsequently encrypted with a key known to both the data owner and the users. This decrypt-and-encrypt solution means the data owner has to be online all the time, which is practically not feasible. The problem becomes increasingly complex when there are multiple pieces of data and diverse data owners and users. Although simple, the traditional encryption schemes involve complex key management protocols and, hence, are not apt for data sharing. Proxy re-encryption (PRE), a notion first proposed by Blaze et al. [2], allows a proxy to transform a file computed under a delegator’s public key into an encryption intended for a delegate. Let the data owner be the delegator and the data user be the delegate. In such a scheme, the data owner can send encrypted messages to the user temporarily without revealing his secret key. The data owner or a trusted third party generates the re-encryption key. A proxy runs the re-encryption algorithm with the key and revamps the ciphertext before sending the new ciphertext to the user. An intrinsic trait of a PRE scheme is that the proxy is not fully trusted (it has no idea of the data owner’s secret key). This is seen as a prime candidate for delegating access to encrypted data in a secured manner, which is a crucial component in any data-sharing scenario. In addition, PRE allows for encrypted data in the cloud to be shared to authorized users while maintaining its confidentiality from illegitimate parties.

**6.SOFTWARE ENVIRONMENT**

**Java Technology**

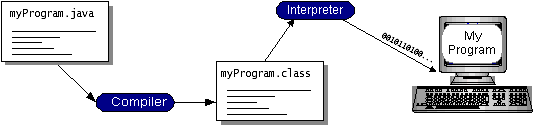
Java technology is both a programming language and a platform.

**The java programming language**

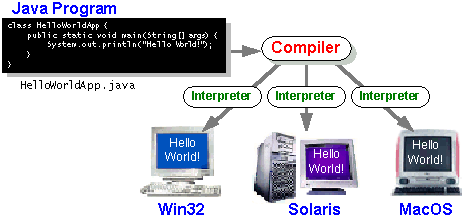
The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

* + - 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.



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 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.



**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 MacOS. Most platforms can be described as 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.

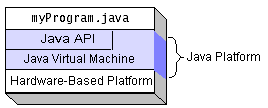
The Java platform has two components:

* 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.



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?**

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.

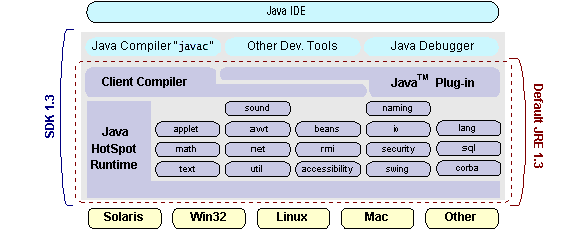
However, the Java programming language is not just for writing cute, entertaining applets for the Web. The general-purpose, high-level Java programming language is also a powerful software platform. Using the generous API, you can write many types of programs.

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 Data gram 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 JavaBeansTM, 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. The following figure depicts what is included in the Java 2 SDK.

****

**How will java 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 JavaTM 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.

### 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.

1. **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.

1. **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.
2. **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.

1. **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.

1. **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.

1. **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.

**Java Program**

**Compilers**

**Interpreter**

**My Program**

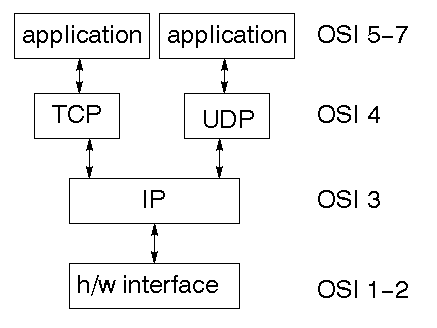
You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it’s a Java development tool or a Web browser that can run Java applets, is an implementation of the Java VM. The Java VM can also be implemented in hardware.

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.

**Networking**

**TCP/IP stack**

The TCP/IP stack is shorter than the OSI one:



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

**IP layers**

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 address**

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. The network ID falls into various classes according to the size of the network address.

**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.

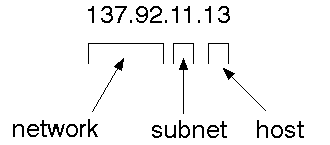
**Sub address**

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 address**

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". A service exists on a host, and is identified by its port. This is a 16 bit number.

**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.

**Jfree chart:**

JFreeChart is a free 100% Java chart library that makes it easy for developers to display professional quality charts in their applications. JFreeChart'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);

JFreeChart 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 XYPlot class in JFreeChart;

Testing, documenting, testing some more, documenting some more.

## 2. Time Series Chart Interactivity

Implement a new (to JFreeChart) 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 JFreeChart 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 JFreeChart only handles a small subset of the properties that can be set for charts. Extend (or reimplement) this mechanism to provide greater end-user control over the appearance of the charts.

**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 JavaOne 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.

**1. 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 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 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 reverification 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 "gotchas" 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.

**4. 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: KJava**

KJava is Sun's proprietary profile and contains the KJava API. The KJava profile is built on top of the CLDC configuration. The KJava virtual machine, KVM, accepts the same byte codes and class file format as the classic J2SE virtual machine. KJava contains a Sun-specific API that runs on the Palm OS. The KJava 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 KJava 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 KJava, 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

**7. SYSTEM STUDY**

### 7.1 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

### 7.2 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.

### 7.3 TECHNICAL FEASIBILITY :

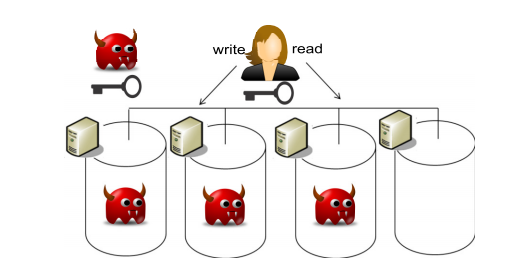
This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

### 7.4 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.

**8.SYSTEM DESIGN**

**8.1 SYSTEM ARCHITECTURE**:

****

**8.2 DATA FLOW DIAGRAM:**

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

Unauthorized Owner

Owner login

Check

No

Yes

Signature and private key

File Upload

polynomial-time

View & download

End process

Unauthorized User

User login

Check

No

Yes

Authentication Key

Search key

File search

View and download

End process

Unauthorized Auditor

Cloud login

Check

No

Yes

Activate User & owner

Respond to User

Key send

End process

Unauthorized User

User login

Check

No

Yes

Authentication Key

Search key

File search

View and download

End process

Unauthorized Auditor

Cloud login

Check

No

Yes

Activate User & owner

Respond to User

Key send

End process

### 8.3 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

**GOALS:**

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

### 8.4 USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

Consumer

Owner

owner

cloud

owner

### 8.5 CLASS DIAGRAM:

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

Data Owner

Login

Secure Authentication ()

Polynomial-time-key ()

Key verification ()

View & download ()

View profile ()

Search key request()

File search ()

View & Download ()

Users

Login

Activate User &owner ()

Key respond ()

Monitoring ()

cloud

Login

### 8.6 SEQUENCE DIAGRAM:

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

Database

cloud

Owner

Registration

User

Login

Verification



File upload

View & down

Search Key

request

Respond \_key

File search

View files

Activate Users

Security

### 8.7 ACTIVITY DIAGRAM:

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

Start

Cloud

Consumer

Owner

View download

Login

Login

Login

View download

Secure Authentication

View download

Activate users

Secure Authentication

Respond request

View & download

Search request

File upload

Key send

Search file

Polynomial time

File upload

Verification

View & download

File upload

View download

Secure Authentication

Polynomial time

File upload

Secure Authentication

Search request

Search file

Activate users

Respond request

Verification

View download

View & download

Key send

**9.SYSTEM REQUIREMENTS**

### 9.1 HARDWARE REQUIREMENTS:

* System : Pentium IV 2.4 GHz.
* Hard Disk : 40 GB.
* Floppy Drive : 1.44 Mb.
* Monitor : 15 VGA Colour.
* Mouse : Logitech.
* Ram : 512 Mb.

**9.2 SOFTWARE REQUIREMENTS:**

* Operating system : - Windows XP/7.
* Coding Language : JAVA/J2EE
* Data Base : MYSQL

View download

### 10.SYSTEM TESTING

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

**TYPES OF TESTS**

**Unit testing**

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

**Integration testing**

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

**Functional test**

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

Functional testing is centered on the following items:

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

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

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

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

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

**System Test**

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

**White Box Testing**

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

**Black Box Testing**

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

**10.1 Unit Testing:**

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

**Test strategy and approach**

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

**Test objectives**

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

**Features to be tested**

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

# **10.2 Integration Testing:**

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

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

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

**10.3 Acceptance Testing:**

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

**10.4 Test Results:**

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

**11.IMPLEMENTATION**

**MODULES:**

1. Data Owner
2. Data User
3. Admin

**MODULES DESCRIPTION:**

**Data Owner:**

In Data Owner module, Initially Data Owner must have to register their detail and admin will approve the registration by sending signature key and private key through email. After successful login he/she have to verify their login by entering signature and private key. Then data Owner can upload files into cloud server with Polynomial key generation. He/she can view the files that are uploaded in cloud by entering the secret file key.

**Data User:**

In Data User module, Initially Data Users must have to register their detail and admin will approve the registration by sending signature key and private key through email. After successful login he/she have to verify their login by entering signature and private key. Data Users can search all the files upload by data owners. He/she can send search request to admin then admin will send the search key. After entering the search key he/she can view the file

**Admin:**

In Admin module, Admin can view all the Data owners and data user’s details. Admin will approve the users and send the signature key and private key to the data owners and data users. Also admin will send the search request key to the users. Admin can able see the files in cloud uploaded by the data owners.

**12.INPUT DESIGN AND OUTPUT DESIGN**

**12.1 INPUT DESIGN**

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

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

**OBJECTIVES**

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

### 12.2 OUTPUT DESIGN

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

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2. Select methods for presenting information.

3. Create document, report, or other formats that contain information produced by the system.

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

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

**13.USER MANUAL AND OUTPUT SCREENS**

**Table of Contents**

1. Introduction

2. System Requirements

3. Installation

4. Configuration

5. Data Sharing Workflow

6. Security Considerations

7. Troubleshooting

8. Frequently Asked Questions (FAQs)

9. Conclusion

10. Appendix: Glossary of Terms

**1. Introduction**

Welcome to the user manual for the Proxy Re-Encryption approach for secure data sharing in IoT devices using blockchain. This manual provides step-by-step instructions to help you understand and implement this approach effectively.

Proxy Re-Encryption (PRE) is a cryptographic technique that allows a trusted proxy to convert and transfer encrypted data from one user to another without decrypting the data. Blockchain technology provides a decentralized and immutable platform to store access control policies and manage data sharing transactions securely in an IoT environment.

**2. System Requirements**

To implement the Proxy Re-Encryption approach, ensure that your system meets the following requirements:

- IoT devices running compatible operating systems (e.g., Linux, Windows, or IoT-specific OS)

- Internet connectivity for IoT devices

- Access to a blockchain network (e.g., Ethereum, Hyperledger Fabric)

- Software development tools (e.g., IDE, programming languages)

- Basic understanding of cryptography and blockchain concepts.

**3. Installation**

To install and set up the Proxy Re-Encryption approach, follow these steps:

1. Install the required software development tools on your development environment.

2. Clone or download the Proxy Re-Encryption code repository from the designated source (e.g., GitHub).

3. Set up the necessary dependencies and libraries as specified in the repository documentation.

4. Configure the blockchain network connection parameters in the code, including the network address and credentials.

5. Build and deploy the smart contracts required for data sharing on the blockchain network.

6. Install and configure the necessary libraries and dependencies on your IoT devices.

**4. Configuration**

To configure the Proxy Re-Encryption approach, follow these steps:

1. Specify the access control policies and rules for data sharing in the blockchain smart contracts.

2. Define the encryption and decryption keys for each user or device involved in the data sharing process.

3. Configure the Proxy Re-Encryption proxy server with the appropriate settings, including network address, port, and security configurations.

4. Establish the connection between the IoT devices and the Proxy Re-Encryption proxy server using secure communication protocols.

**5. Data Sharing Workflow**

The Proxy Re-Encryption approach follows a typical workflow for secure data sharing in IoT devices. The following steps outline the process:

1. User A initiates a data sharing request with User B through the IoT application or interface.

2. The IoT application verifies User A's credentials and sends the request to the blockchain network.

3. The blockchain network checks the access control policies defined in the smart contracts to ensure User A has permission to share the data with User B.

4. Upon successful verification, the blockchain network generates a re-encryption key and sends it to the Proxy Re-Encryption proxy server.

5. The Proxy Re-Encryption proxy server uses the re-encryption key to convert the encrypted data from User A into a format that User B can decrypt.

6. The Proxy Re-Encryption proxy server securely transmits the re-encrypted data to User B's IoT device.

7. User B's IoT device decrypts the re-encrypted data using its private key, which was securely stored and managed by the device.

8. User B can now access and utilize the shared data securely.

**6. Security Considerations**

When implementing the Proxy Re-Encryption approach for secure data sharing in IoT devices using blockchain, consider the following security measures:

- Ensure strong encryption algorithms are used for data encryption and decryption.

- Protect private keys associated with IoT devices from unauthorized access or theft.

- Regularly update and patch software and firmware on IoT devices to address any security vulnerabilities.

- Implement secure communication protocols (e.g., TLS) between IoT devices and the Proxy Re-Encryption proxy server.

- Periodically audit and review access control policies defined in the blockchain smart contracts to ensure they align with the intended security requirements.

**7. Troubleshooting**

If you encounter any issues during the implementation or usage of the Proxy Re-Encryption approach, consider the following troubleshooting steps:

1. Verify that all required dependencies and libraries are installed correctly.

2. Check the network connectivity between the IoT devices, the blockchain network, and the Proxy Re-Encryption proxy server.

3. Review the configuration settings to ensure they are accurate and up to date.

4. Consult the documentation and resources provided with the Proxy Re-Encryption code repository for known issues and solutions.

5. Seek assistance from the developer community or support channels associated with the Proxy Re-Encryption approach.

**8. Frequently Asked Questions (FAQs)**

Q1: Can the Proxy Re-Encryption approach be used with any blockchain network?

A1: Yes, the Proxy Re-Encryption approach is blockchain-agnostic and can be implemented with various blockchain networks such as Ethereum or Hyperledger Fabric.

Q2: How are access control policies managed in the Proxy Re-Encryption approach?

A2: Access control policies are defined as smart contracts on the blockchain network. These policies determine who can share data and with whom, based on predefined rules.

Q3: What happens if the Proxy Re-Encryption proxy server is compromised?

A3: If the proxy server is compromised, an attacker may gain unauthorized access to the re-encryption keys. It is crucial to implement robust security measures to protect the proxy server.

**9. Conclusion**

Congratulations! You have successfully completed the user manual for the Proxy Re-Encryption approach for secure data sharing in IoT devices using blockchain. This approach provides a powerful and secure method for sharing data in IoT environments, ensuring privacy and data integrity. Remember to follow best practices for security and keep the system up to date.

**10. Appendix: Glossary of Terms**

- Proxy Re-Encryption (PRE): A cryptographic technique that allows a trusted proxy to convert and transfer encrypted data from one user to another without decrypting the data.

- Blockchain: A decentralized and immutable distributed ledger technology that stores and manages transactions and data in a transparent and secure manner.

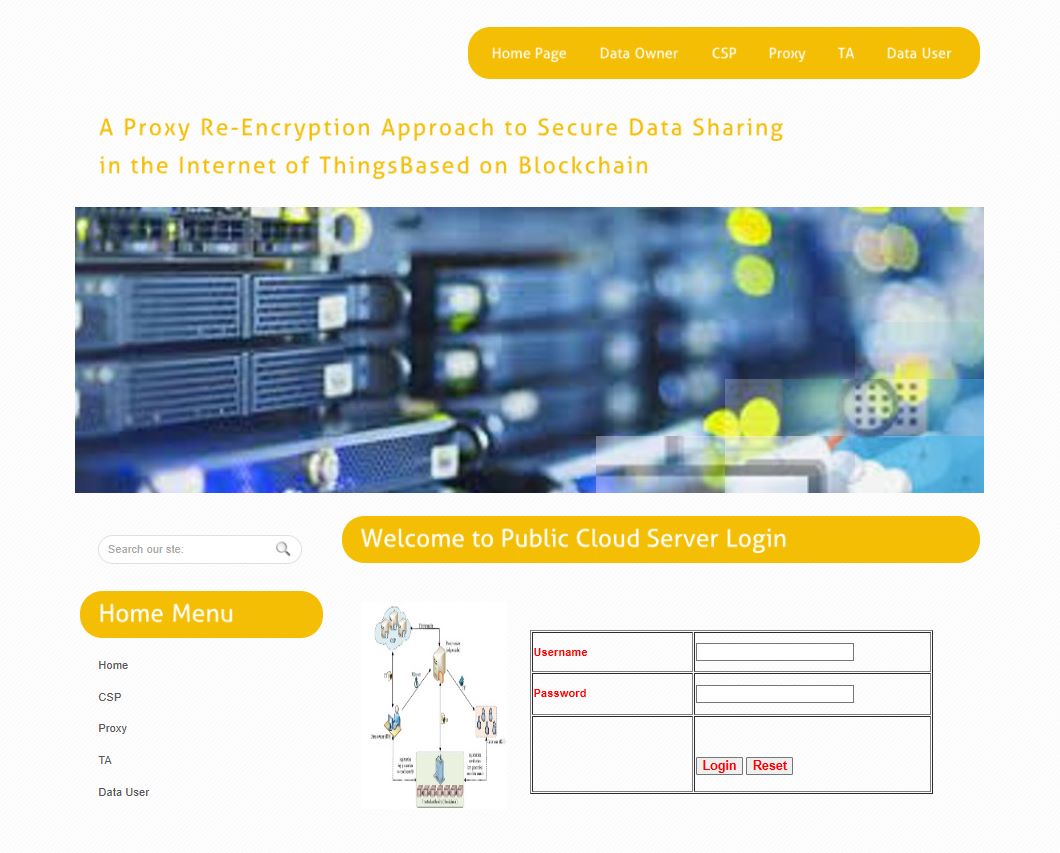
- IoT: Internet of Things - A network of interconnected physical devices that collect and exchange data over the internet.

- Smart Contracts: Self-executing contracts with the terms of the agreement directly written into code, deployed and executed on a blockchain network.

### OUTPUT SCREENS



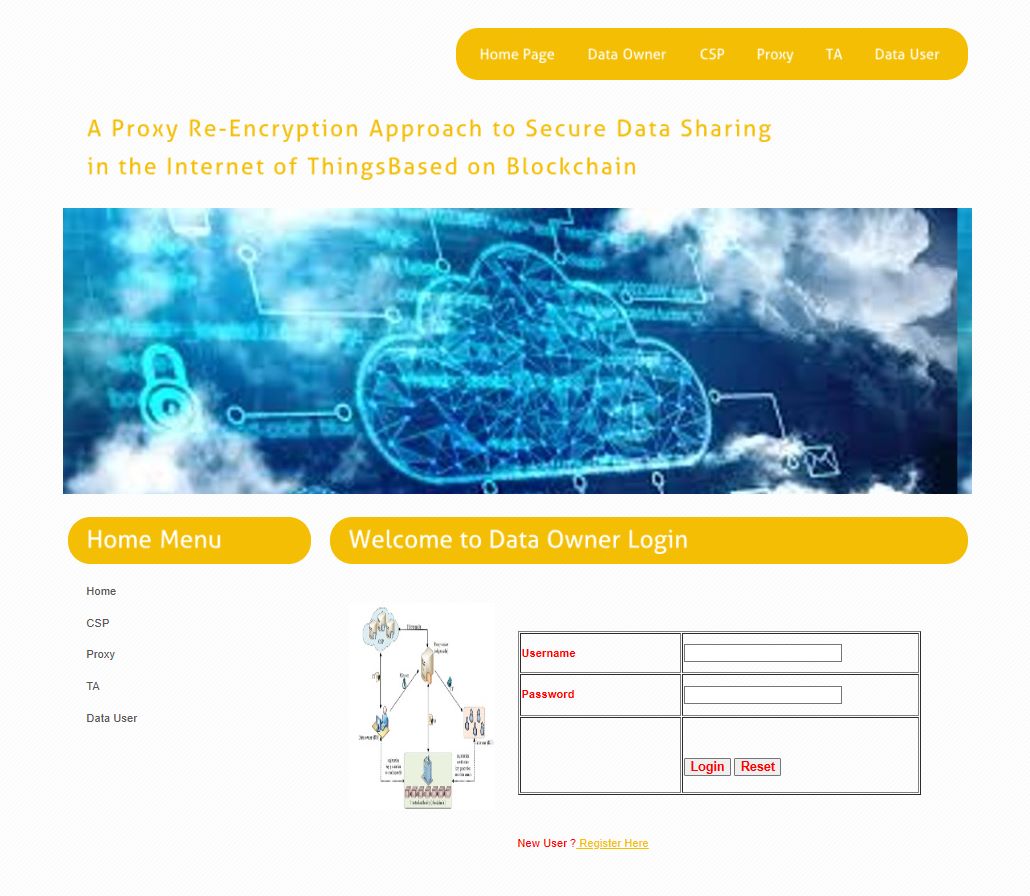
**Fig 13.1.1:- Home Page**



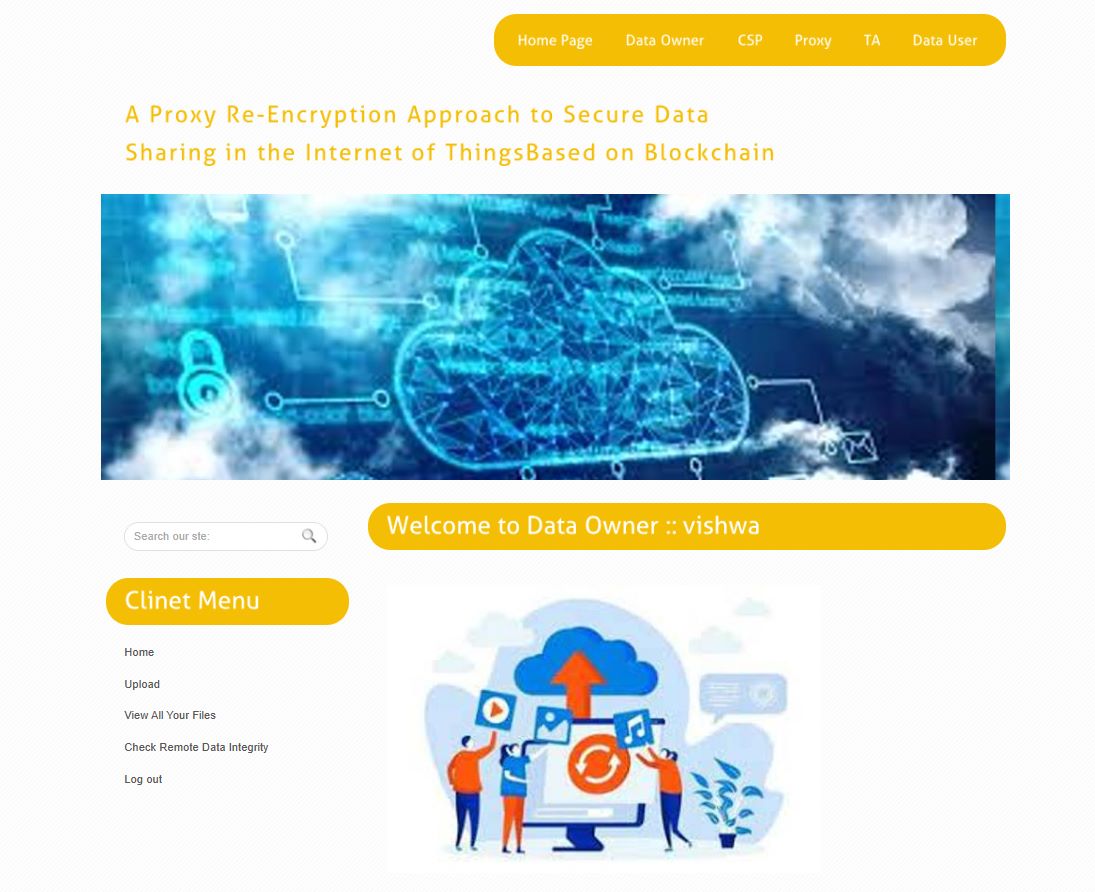
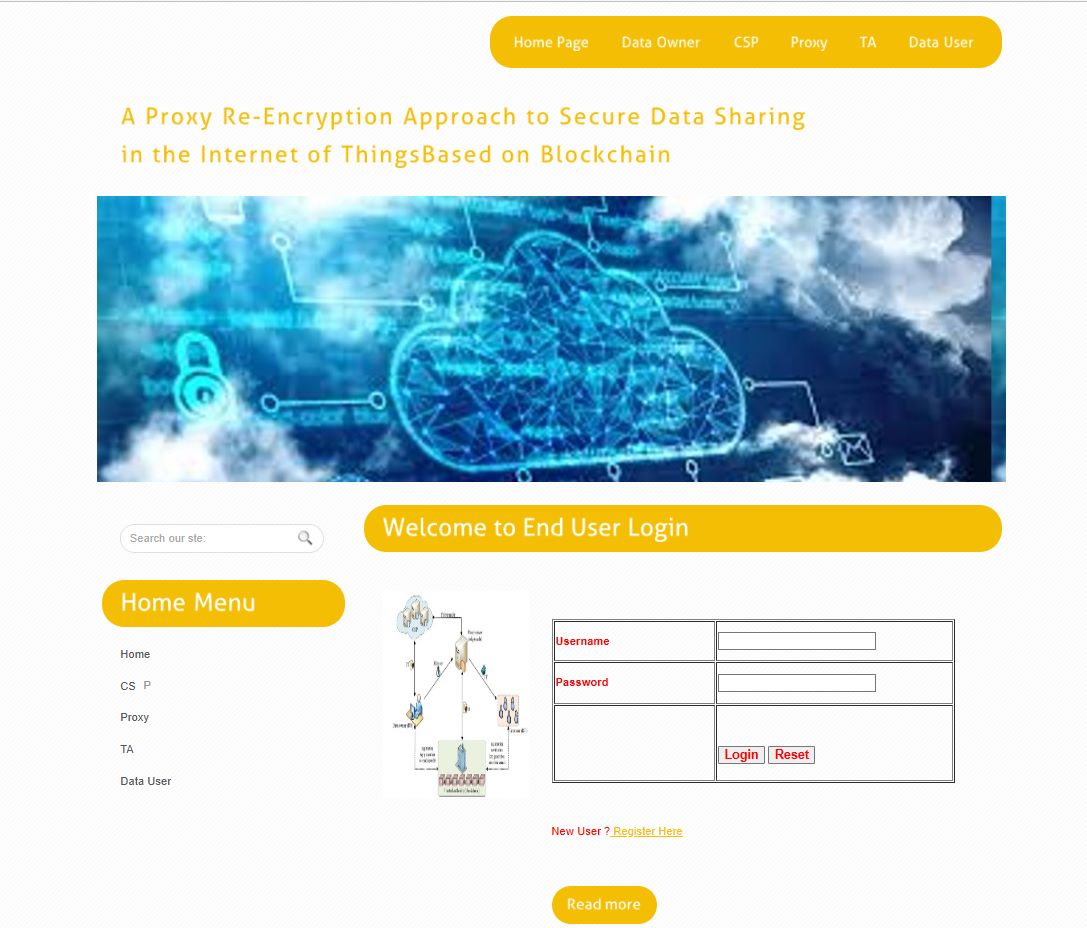
**Fig 13.1.2 :- Cloud Service Login Page**



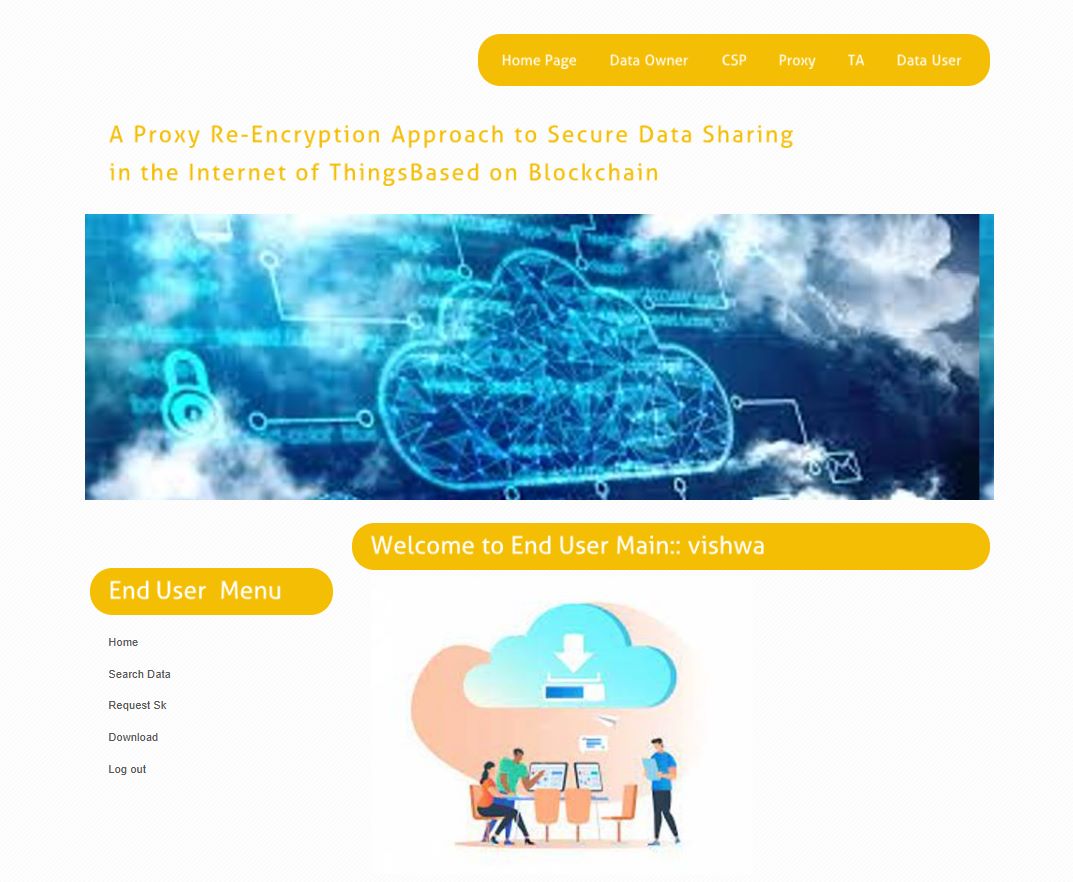
**Fig 13.1.3 :- Csp Main Page**

****

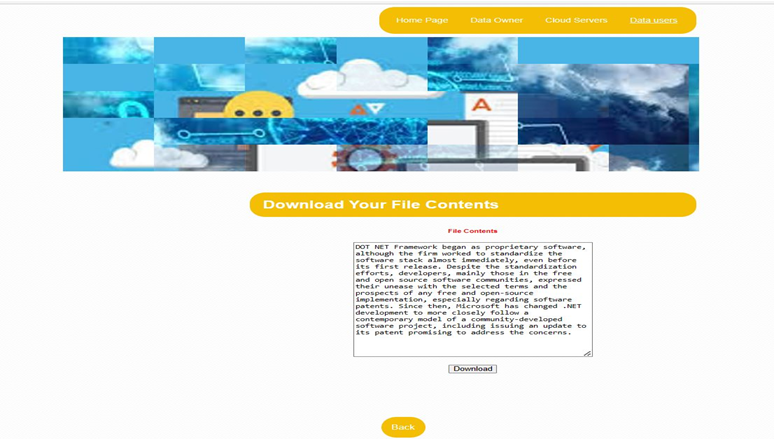
**Fig 13.2.4 :- Data Owner Login Page**

**Fig 13.2.5 :-** **Data Owner Main Page**

**Fig 13.2.6 :- End User Login Page**

****

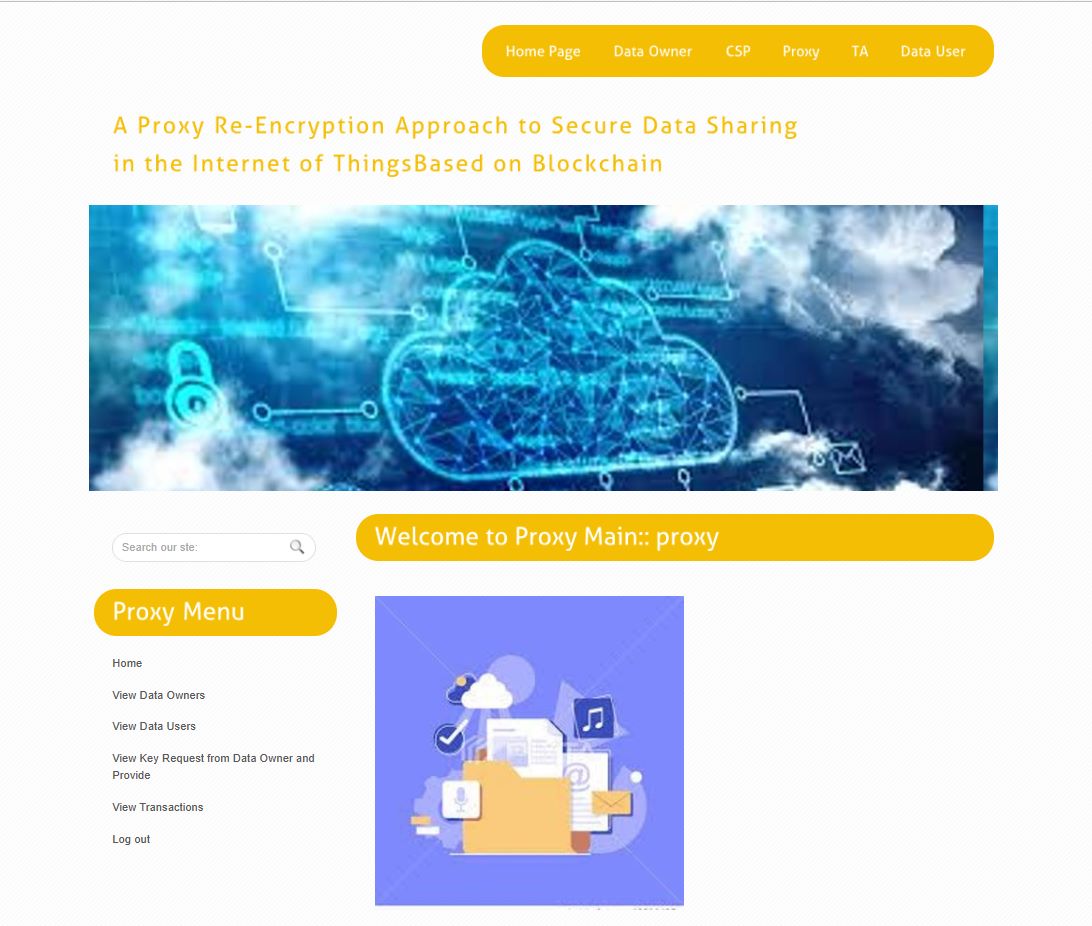
**Fig 13.1.7 :- End User Main Page**

****

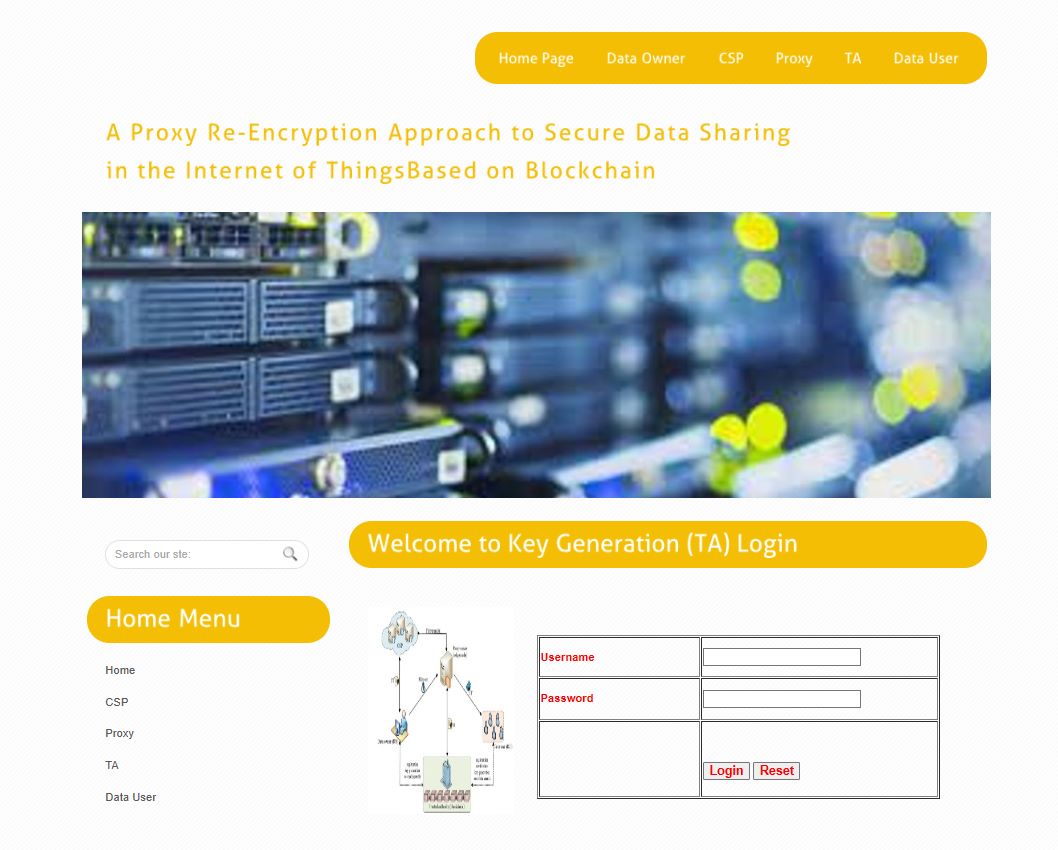
**Fig 13.1.8 :-** **End User Download Page**

****

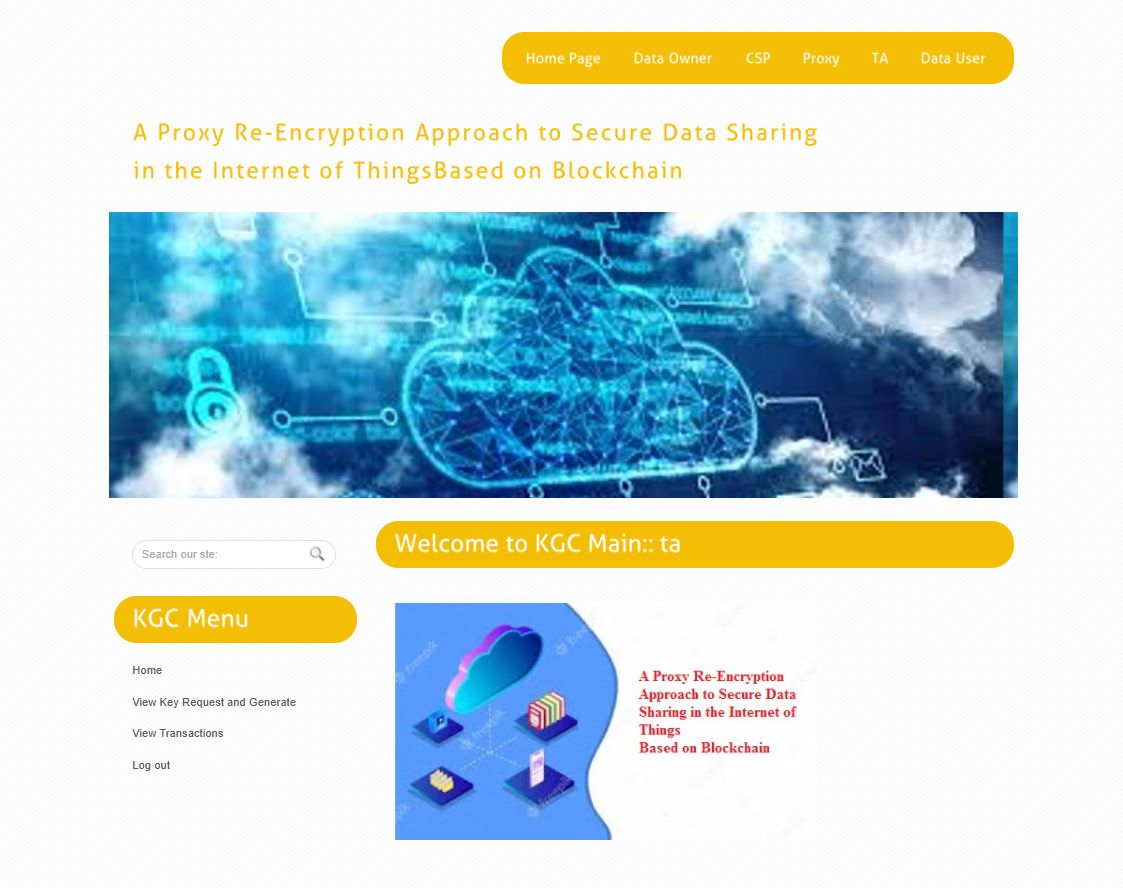
**Fig 10.1.9 :- Fig 13.1.8 :-** **Proxy Login Page**

****

**Fig 13.1.10 :- Proxy Main Page**

****

**Fig 13.1.11:-** **TA Login page**

****

**Fig 13.1.12:** **TA Main Page**

**14.CONCLUSION**

The emergence of the IoT has made data sharing one of its most prominent applications. To guarantee data confidentiality, integrity, and privacy, we propose a secure identity-based PRE data-sharing scheme in a cloud computing environment. Secure data sharing is realized with IBPRE technique, which allows the data owners to store their encrypted data in the cloud and share them with legitimate users efficiently. Due to resource constraints, an edge device serves as the proxy to handle the intensive computations. The scheme also incorporates the features of ICN to proficiently deliver cached content, thereby improving the quality of service and making great use of the network bandwidth. Then, we present a blockchain-based system model that allows for flexible authorization on encrypted data. Fine grained access control is achieved, and it can help data owners achieve privacy preservation in an adequate way. The analysis and results of the proposed model show how efficient our scheme is, compared to existing schemes.

**15.BIBLIOGRAPHY**

[1] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, “Internet of Things: A survey on enabling technologies, protocols, and applications,” IEEE Commun. Surveys Tut., vol. 17, no. 4, pp. 2347–2376, Oct./Dec. 2015.

[2] M. Blaze, G. Bleumer, and M. Strauss, “Divertible protocols and atomic proxy cryptography,” in Proc. Int. Conf. Theory Appl. Cryptographic Techn., Springer, May 1998, pp. 127–144.

[3] A. Shamir, “Identity-based cryptosystems and signature schemes,” in Proc. Workshop Theory Appl. Cryptographic Techn., Springer, Aug. 1984, pp. 47–53.

[4] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, “Public key encryption with keyword search,” in Proc. Int. Conf. Theory Appl. Cryptographic Techn., Springer, May 2004, pp. 506–522.

[5] B. R. Waters, D. Balfanz, G. Durfee, and D. K. Smetters, “Building an encrypted and searchable audit log,” in NDSS, vol. 4. Citeseer, Feb. 2004, pp. 5–6.

[6] D. Balfanz et al., “Secret handshakes from pairing-based key agreements,” in Proc. IEEE, Symp. Secur. Privacy, 2003, pp. 180–196.

[7] R. Canetti, S. Halevi, and J. Katz, “Chosen-ciphertext security from identity-based encryption,” in Proc. Int. Conf. Theory Appl. Cryptographic Techn., Springer, 2004, pp. 207–222.

[8] T. Koponen et al., “A data-oriented (and beyond) network architecture,” in Proc. Conf. Appl., Techn., Architectures, Protoc. Comput. Commun., Aug. 2007, pp. 181–192.

[9] N. Fotiou, P. Nikander, D. Trossen, and G. C. Polyzos, “Developing information networking further: From PSIRP to pursuit,” in Proc. Int. Conf. Broadband Commun., Netw. Syst., Springer, Oct. 2010, pp. 1–13.

[10] C. Dannewitz, J. Golic, B. Ohlman, and B. Ahlgren, “Secure naming for a network of information,” in Proc. INFOCOM IEEE Conf. Comput. Commun. Workshops,2010, pp. 1–6.

[11] A. Carzaniga, M. J. Rutherford, and A. L. Wolf, “A routing scheme for content-based networking,” in Proc. IEEE INFOCOM 2004, vol. 2, 2004, pp. 918–928.

[12] I. Psaras, W. K. Chai, and G. Pavlou, “Probabilistic in-network caching for information-centric networks,” in Proc. 2nd ed. ICN Workshop Inform.- Centric Netw., Aug. 2012, pp. 55–60.

[13] Y. Sun et al., “Trace-driven analysis of ICN caching algorithms on video on-demand workloads,” in Proc. 10th ACM Int. Conf. Emerging Netw. Exp. Technol., Dec. 2014, pp. 363–376.

[14] S. Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System, vol. 4. Bitcoin.org, 2008. Available: https://bitcoin. org/bitcoin. pdf

[15] S. Yu, C. Wang, K. Ren, and W. Lou, “Achieving secure, scalable, and fine-grained data access control in cloud computing,” in Proc. IEEE INFOCOM, Mar. 2010, pp. 1–9.

[16] N. Park, “Secure data access control scheme using type-based re encryption in cloud environment,” in Semantic Methods Knowledge Management and Communications. Berlin, Germany: Springer, 2011, pp. 319– 327.

[17] G. Wang, Q. Liu, J. Wu, and M. Guo, “Hierarchical attribute-based encryption and scalable user revocation for sharing data in cloud servers,” Comput. Secur., vol. 30, no. 5, pp. 320–331, Jul. 2011. [18] J. Hur, “Improving security and efficiency in attribute-based data sharing,” IEEE Trans. Knowl. Data Eng., vol. 25, no. 10, pp. 2271–2282, Apr. 2011.

[19] P. K. Tysowski and M. A. Hasan, “Hybrid attribute-and re-encryption based key management for secure and scalable mobile applications in clouds,” IEEE Trans. Cloud Comput., vol. 1, no. 2, pp. 172–186, Nov. 2013.

[20] Q. Liu, G. Wang, and J. Wu, “Time-based proxy re-encryption scheme for secure data sharing in a cloud environment,” Inform. Sci., vol. 258, pp. 355–370, Feb. 2014.

[21] J. Han, W. Susilo, and Y. Mu, “Identity-based data storage in cloud computing,” Future Gener. Comput. Syst., vol. 29, no. 3, pp. 673–681, Mar. 2013.

[22] H.-Y. Lin, J. Kubiatowicz, and W.-G. Tzeng, “A secure fine-grained access control mechanism for networked storage systems,” in Proc. IEEE 6th Int. Conf. Softw. Secur. Rel., Jun. 2012, pp. 225–234.

[23] Y. Zhou et al., “Identity-based proxy re-encryption version 2: Making mobile access easy in cloud,” Future Gener. Comput. Syst., vol. 62, pp. 128–139, Sep. 2016.

[24] X. A. Wang, J. Ma, F. Xhafa, M. Zhang, and X. Luo, “Cost-effective secure e-health cloud system using identity based cryptographic techniques,” Future Gener. Comput. Syst., vol. 67, pp. 242–254, Feb. 2017.

[25] J. Shao, G. Wei, Y. Ling, and M. Xie, “Identity-based conditional proxy re-encryption,” in Proc. IEEE Int. Conf. Commun., Jun. 2011, pp. 1–5