1. INTRODUCTION

1.1 General

In this work, we aim to creating a in this project we have successfully built a prototype supporting live migration of VMs between any two nodes on the Internet (even though they are behind different Networks). In the future, we will study fault-tolerance support for a SOC system; we will also conduct sensitivity analysis of how violation of our model assumptions would impact the optimal resource allocation.

Cloud computing has recently emerge (come into view) as a computing paradigm in which storage and computation (calculation) can be outsourced from organizations to next generation data centers hosted. In this paper to provide scalability and elasticity (flexibility), cloud services often make heavy use of replication to ensure consistent performance and availability this consistency model is a variant of weak consistency that allows data to be inconsistent among some replicas during the update process, but ensures that updates will eventually be propagated to all replicas. This makes it difficult to strictly maintain the ACID guarantees; consistency part of ACID is sacrificed to provide reasonable availability Interesting consistency problems can arise as transactional database systems are deployed in cloud environments and use policy-based authorization systems to protect sensitive resources.

1.2 Objective

We aim as a fundamental difference to existing approaches, we formulate such a resource allocation problem to be a convex optimization problem. Given a task with its resource requirements and a budget, we first prove that the optimal resource allocation on a qualified node that can minimize a task's execution time does exist. We further show that it is nontrivial to solve such a convex optimization problem directly via a brute-force strategy and the interior point method.

1.3 Existing System

In existing, In this system are used on transaction in cloud environment and also have drawback in policy-based authorization systems they are consistency

problems can arise in transaction database also its easily affected by unauthorized person the data and Incremental Punctual the worst performance especially when frequent policy updates occur. Interesting consistency problems can arise as transactional database systems are deployed in cloud environments and use
policy-based authorization systems to protect sensitive resources.

Existing System Drawbacks:

- It stores without encrypt data, so unknown person can access it.
- Number of request from client to server, which will be burden.

1.4 Proposed System

In proposed system, we are using two phase commit protocol in this protocol is also used to store the database in cloud environment. Also we verify the person, whether he is valid or not and we achieve the atomic commit protocol. We are using Two-Phase Validation Commit protocol, by using the algorithm to give more secure transaction , it identifies both the trusted person and achieves the acid properties. Before Authorization they send prepare-to-commit technique , which they used for identifying the person.

In this project, what we achieving to give the user more secure transaction as possible And also the notion of trusted transactions when dealing with proofs of authorization. Accordingly, we propose several increasingly stringent levels of policy consistency constraints, and present different enforcement approaches to guarantee the trustworthiness of transactions executing on cloud servers

Proposed System advantages:

- The ‘n’ number client can access server without burdening it by using 2PVC protocol.
- It stores secure data by using encryption.

2. LITERATURE REVIEW

TITLE 1 : Building Castles out of Mud: Practical Access Pattern Privacy and Correctness on Untrusted Storage

AUTHOR : Zhi-Dan Zhao, Lei Rao, Xue Liu

YEAR : Feburary, 2013.

DESCRIPTION :

We introduce a new practical mechanism for remote data storage with efficient access pattern privacy and correctness. A storage client can deploy this mechanism to issue encrypted reads, writes, and inserts to a potentially curious and malicious storage service provider, without revealing information or access patterns. The provider is unable to establish any correlation between successive accesses, or even to distinguish between a read and a write. Moreover, the client is provided with strong correctness assurances for its operations − illicit provider behavior does not go undetected.

TITLE 2 : HAIL: A High-Availability and Integrity Layer for Cloud Storage

AUTHOR : Xiwang Yang, Yang Guo, and Yong Liu

YEAR : May ,2012.

DESCRIPTION:

We introduce HAIL (High-Availability and Integrity Layer), a distributed cryptographic system that permits a set of servers to prove to a client that a stored file is intact and retrievable. HAIL strengthens, formally unifies, and streamlines distinct approaches from the cryptographic and distributed-systems communities. Proofs in HAIL are efficiently computable by servers and highly compact typically tens or hundreds of bytes, irrespective of file size. HAIL cryptographically verifies and reactively reallocates file shares. It is robust against an active, mobile adversary, i.e., one that may progressively corrupt the full set of servers.

TITLE 3 : Provable Data Possession at Untrusted Stores∗

AUTHOR : Stavros Giuseppe Ateniese,Randal Burns, Reza Curtmola

YEAR : March, 2012.

DESCRIPTION :

We introduce a model for provable data possession (PDP) that allows a client that has stored data at an un trusted server to verify that the server possesses the original data without retrieving it. The model generates probabilistic proofs of possession by sampling random sets of blocks from the server, which drastically reduces I/O costs. The client maintains a constant amount of metadata to verify the proof In particular, the overhead at the server is low (or even constant), as opposed to linear in the size of the data. Experiments using our implementation verify the practicality of PDP and reveal that the performance of PDP is bounded by disk I/O and not by cryptographic computation.

TITLE 4 : Consistency Rationing in the Cloud: Pay only when it matters

AUTHOR : Kleanthi Lakiotaki1, Nikolaos F. Matsatsinis and Alexis

YEAR : October,2011.

DESCRIPTION:

Cloud storage solutions promise high scalability and low cost. Existing solutions, however, differ in the degree of consistency they provide. Our experience using such systems indicates that there is a non-trivial trade-off between cost, consistency and availability. High consistency implies high cost per transaction and, in some situations, reduced availability. Low consistency is cheaper but it might result in higher operational cost because of, e.g., overselling of products in a Web shop. In this paper, we present a new transaction paradigm, that not only allows designers to define the consistency guarantees on the data instead at the transaction level, but also allows to automatically switch consistency guarantees at runtime. We demonstrate the feasibility and potential of the ideas through extensive experiments on a first prototype implemented on Amazon's S3 and running the TPC-W benchmark.

TITLE 5 : Relaxed Currency and Consistency: How to Say "Good Enough" in SQL
AUTHOR : Hongfei Guo, Per-Åke Larson
YEAR : March,2011.
DESCRIPTION:

Despite the widespread and growing use of asynchronous copies to improve scalability, performance and availability, this practice still lacks a firm semantic foundation. Applications are written with some understanding of which queries can use data that is not entirely current and which copies are "good enough"; however, there are neither explicit requirements nor guarantees. We propose to make this knowledge available to the DBMS through explicit currency and consistency (C&C) constraints in queries and develop techniques so the DBMS can guarantee that the constraints are satisfied.

In this paper we describe our model for expressing C&C constraints, define their semantics, and propose SQL syntax. onsistency constraints are enforced at compile time while currency constraints are enforced at run time by dynamic plans that check the currency of each local replica before use and select sub-plans accordingly. This approach makes optimal use of the cache DBMS while at the same time guaranteeing that applications always get data that is "good enough" for their purpose.

TITLE 6 : Enforcing Policy and Data Consistency of Cloud Transactions
AUTHOR : Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl.
YEAR : June,2010.
DESCRIPTION:

In distributed transactional database systems deployed over cloud servers, entities cooperate to form proofs of authorizations that are justified by collections of certified credentials. These proofs and credentials may be evaluated and collected over extended time periods under the risk of having the underlying authorization policies or the user credentials being in inconsistent states. It therefore becomes possible for a policy-based authorization systems to make unsafe decisions that might threaten sensitive resources. In this paper, we highlight the criticality of the problem. We then present the first formalization of the concept of trusted transactions when dealing with proofs of authorizations. We propose a Two-Phase Validation Commit protocol as a solution, that is a modified version of the basic Two-Phase Commit protocols. We finally provide performance analysis of the different presented approaches to guide the decision makers in which approach to use.

TITLE 7 : Above the Clouds: A Berkeley View of Cloud Computing

AUTHOR : Michael Isard, Mihai Budiu.

YEAR : January,2010.

DESCRIPTION :

Cloud Computing, the long-held dream of computing as a utility, has the potential to transform a large part of the IT industry, making software even more attractive as a service and shaping the way IT hardware is designed and purchased. Developers with innovative ideas for new Internet services no longer require the large capital outlays in hardware to deploy their service or the human expense to operate it. They need not be concerned about over provisioning for a service whose popularity does not meet their predictions, thus wasting costly resources, or under provisioning for one that becomes wildly popular, thus missing potential customers and revenue. Moreover, companies with large batch-oriented tasks can get results as quickly as their programs can scale, since using 1000 servers for one hour costs no more than using one server for 1000 hours. This elasticity of resources, without paying a premium for large scale, is unprecedented in the history of IT.

3. DESIGN.

3.1 Modules

- User interface
- Quality of service
- Authorization policies
- Distributed transactions
- Certificate Authorities

Module Description and Diagrams:

User Interface Design:

To connect with server user must give their username and password then only they can able to connect the server. If the user already exits directly can login into the server else user must register their details such as username, password and Email id, into the server. Server will create the account for the entire user to maintain upload and download rate.

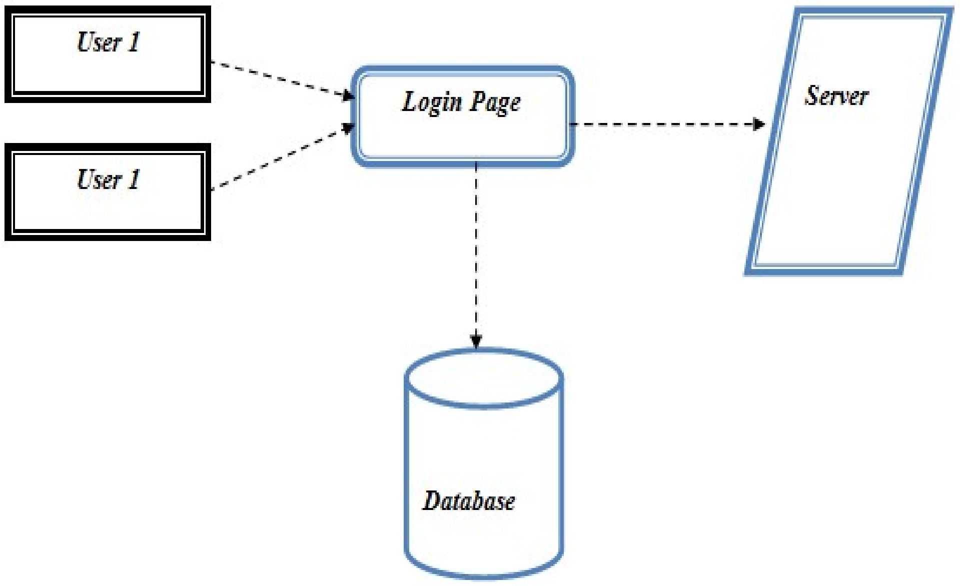


Fig:3.1 User Interface Design

Quality Of Service:

In this module, the data is given by customer requests arrive at each front-end proxy server. After the receiving the data it sense automatically to check the whether the server the total number of Server. We assume that there exists a proxy/DNSserver collocated with each request source.

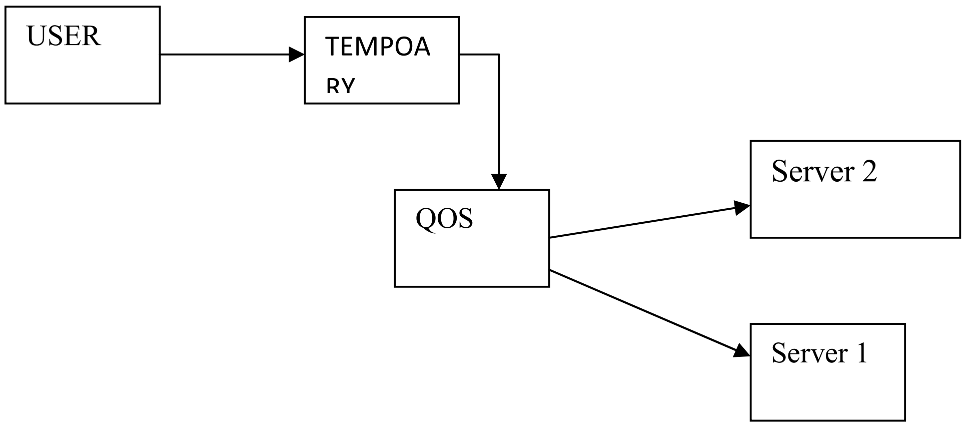


Fig:3.2 Quality of Service

Authorization Policies:

In this model, once user add the document in the cloud server and its compute the file and also analysis the user. It’s also analysis file belong to which category and pass the file in to particular database in cloud environment.

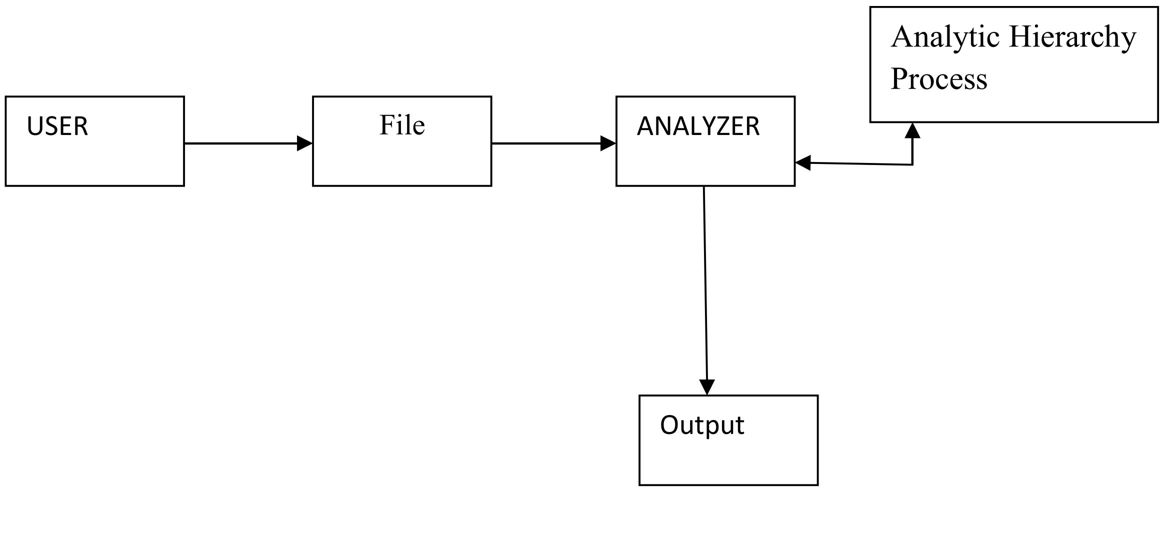


Fig:3.3 Authorization Policies

Distributed Transactions:

In this model we going to get value from authorization model and we will store the data in data in particular cloud database in cloud environment using transaction management in the system. Transaction management used for safe transaction process in file transfer system.It’s using a commit or rollback in transaction system.

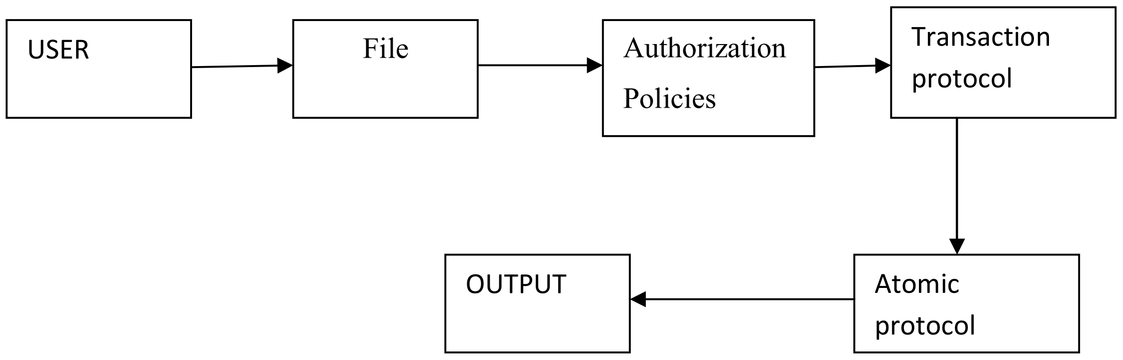


Fig:3.4 Distributed Transactions

Certificate Authorities:

In this model we are going to analysis the file transfer system are monitor by a third party person .see entire transaction those who are pass the file in cloud they are file are belong to the particular database or un wanted are store in any database..

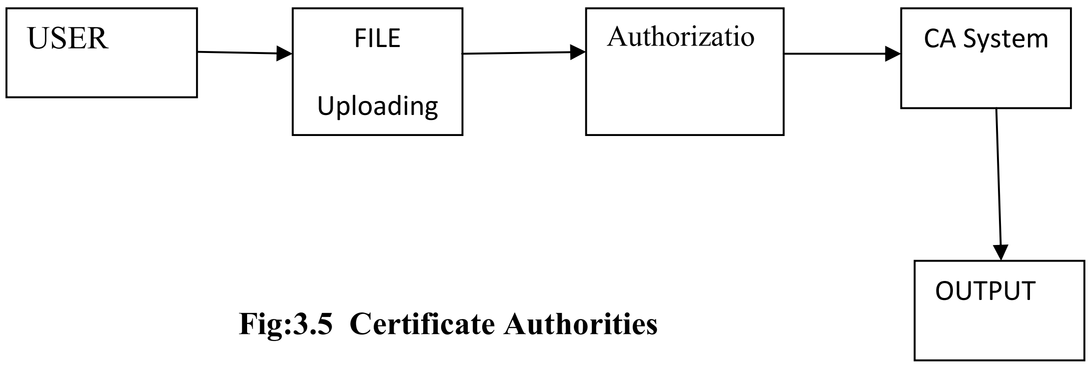
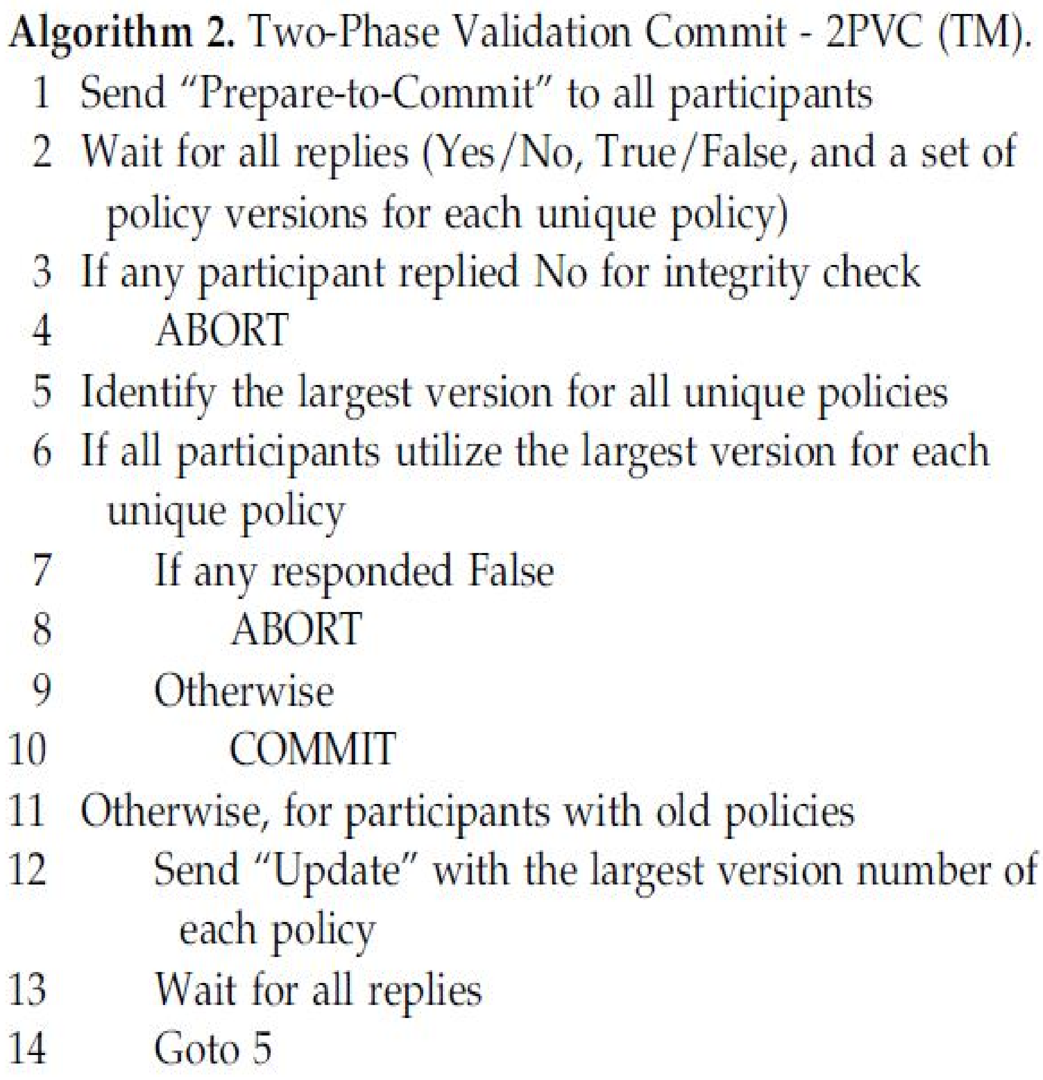


Fig:3.5 Certificate Authorities

Algorithm used-2PVC

System Techniques

The two-phase commit protocol (2PC) is a type of atomic commitment protocol (ACP). It is a distributed algorithm that coordinates all the processes that participate in a distributed atomic transaction on whether to commit or *abort* (roll back) the transaction (it is a specialized type of consensus protocol. However, it is not resilient to all possible failure configurations, and in rare cases user (e.g., a system's administrator) intervention is needed to remedy an outcome.



Algorithm 2. Two-Phase Validation Commit - 2PVC (TM).

```
1  Send "Prepare-to-Commit" to all participants
2  Wait for all replies (Yes/No, True/False, and a set of
     policy versions for each unique policy)
3  If any participant replied No for integrity check
4        ABORT
5  Identify the largest version for all unique policies
6  If all participants utilize the largest version for each
     unique policy
7        If any responded False
8              ABORT
9        Otherwise
10             COMMIT
11 Otherwise, for participants with old policies
12       Send "Update" with the largest version number of
           each policy
13       Wait for all replies
14       Goto 5
```

3.2 System Design

System Architecture:

The systems architect establishes the basic structure of the system, defining the essential core design features and elements that provide the framework for all that follows, and are the hardest to change later. The systems architect provides the architects view of the users' vision for what the system needs to be and do, and the paths along which it must be able to evolve, and strives to maintain the integrity of that vision as it evolves during detailed design and implementation.

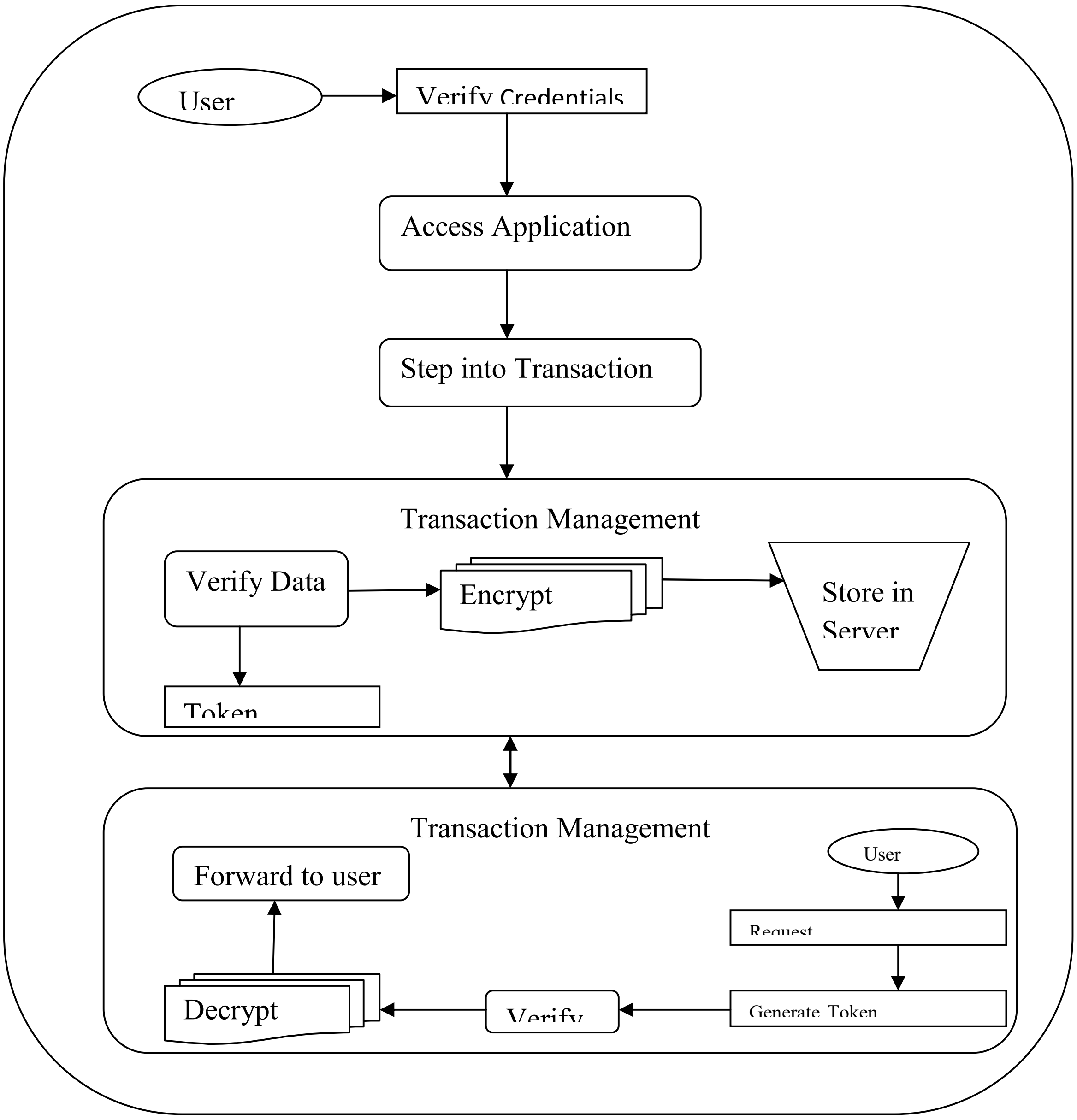


Fig:3.6 System Architecture

Architecture Diagram: The systems architect establishes the basic structure of the system, defining the essential core design features and elements that provide the framework for all that follows, and are the hardest to change later. The systems architect provides the architects view of the users' vision for what the system needs to be and do, and the paths along which it must be able to evolve, and strives to maintain the integrity of that vision as it evolves during detailed design and implementation.

3.3 UML Diagram

UML Diagrams:

A diagram is the symbolic representation of information according to some visualization technique. In other words ,it is the graphical presentation of a set of elements most often referred to connected graph of vertices(things) and arcs(relationships)

UML Introduction:

The Unified Modeling language is a standard general purpose modeling language in the field of object-oriented software engineering.UML includes a set of graphic notation techniques to create visual models of object-oriented software systems. UML combines techniques of data modeling, business modeling, object modeling and component modeling and can be used throughout the software development life-cycle and across different implementation technologies.

UML specification:

There are four parts to the UML 2.x specification:

1. The Superstructure that defines the notation and semantics for diagrams and their model elements

2. The Infrastructure that defines the core metamodel on which the Superstructure is based

3. The Object Constraint Language(OCL) for defining rules for model elements

4. The UML Diagram Interchange that defines how UML 2 diagram layouts are exchanged.

The current versions of these standards follow:

UML Superstructure version 2.4.1

UML Infrastructure version 2.4.1

OCL version 2.3.1

UML Interchange version 1.0

Design:

UML offers a way to visualize a system's architectural blueprints in a diagram, including elements such as:

- Any activities (jobs);
- Individual component of the system;
- And how they interact with other software components;
- How the system will run;
- How entities interact with others;
- External interface

Modeling:

It is important to distinguish between the UML model and the set of diagrams of the system. A diagram is apartial graphic representation of a system's model.The model may also contain documentation that drives the model elements and diagrams

UML diagrams represent two different views of a system model.

- **Static (or Structural) view :** Emphasizes the static structure of the system using objects , attributes ,operations and relationships.

It includes: Class Diagrams , Composite Structure diagrams , Object Diagrams , Deployment Diagrams, Component Diagrams.

- **Dynamic (or Behavioural) view:** Emphasizes the dynamic behavior of the system by showing collaborations among objects and changes to the internal states of objects.

It includes: Sequence diagram , Activity diagrams , State machine diagrams ,use case diagrams, Interaction Diagram.

Activity Diagram:

Activity diagram are a loosely defined diagram to show workflows of stepwise activities and actions, with support for choice, iteration and concurrency. UML, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. UML activity diagrams could potentially model the internal logic of a complex operation. In many ways UML activity diagrams are the object-oriented equivalent of flow charts and data flow diagrams (DFDs)from structural development.

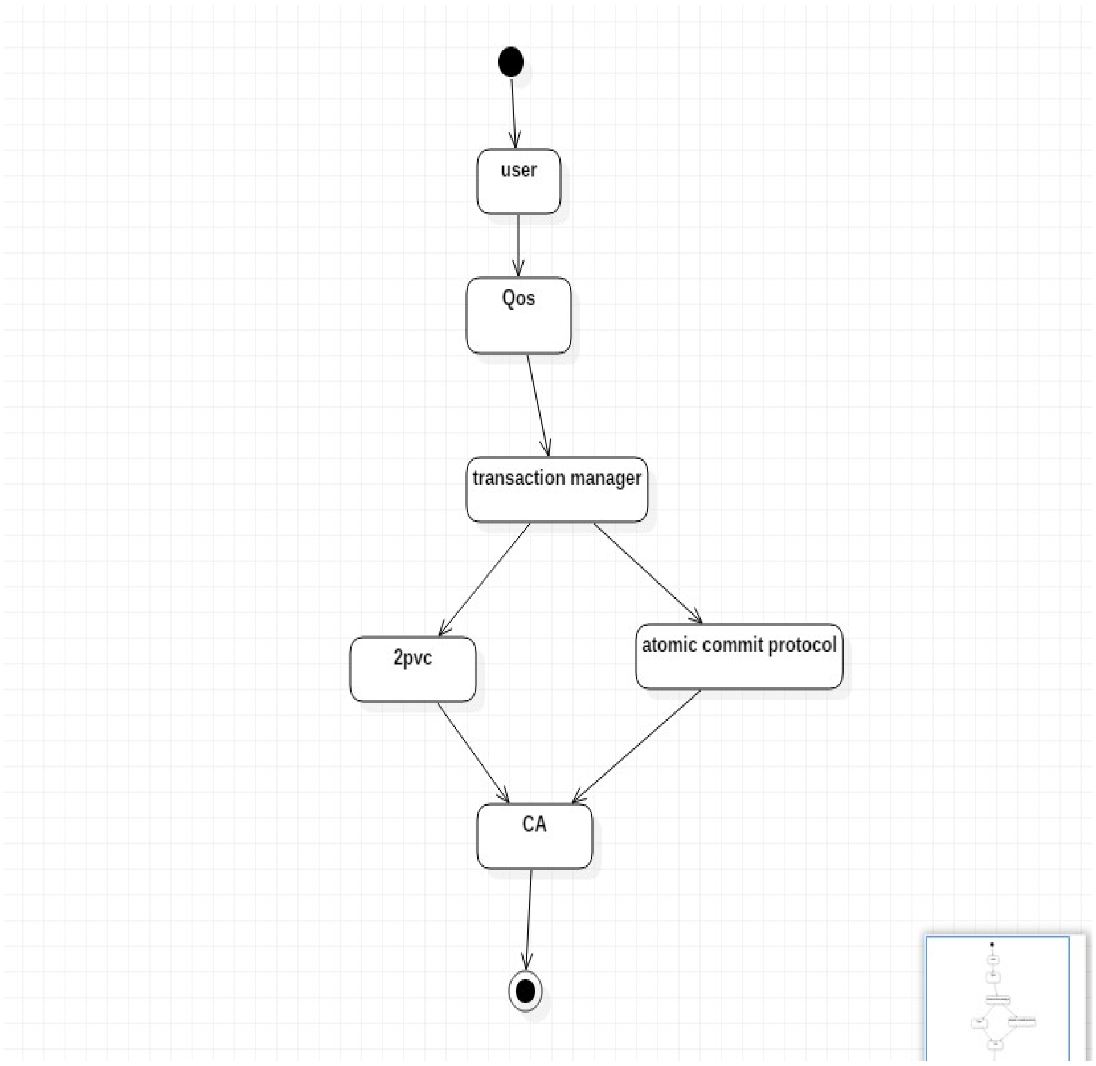


Fig:3.7 Activity Diagram

Use case Diagram:

A use case diagram is a type of behavioral diagram created from a Use-case analysis. The purpose of use case is to present overview of the functionality provided by the system in terms of actors, their goals and any dependencies between those use cases. In the below diagram eleven use cases are depicted. They are used to search result using CST methods.

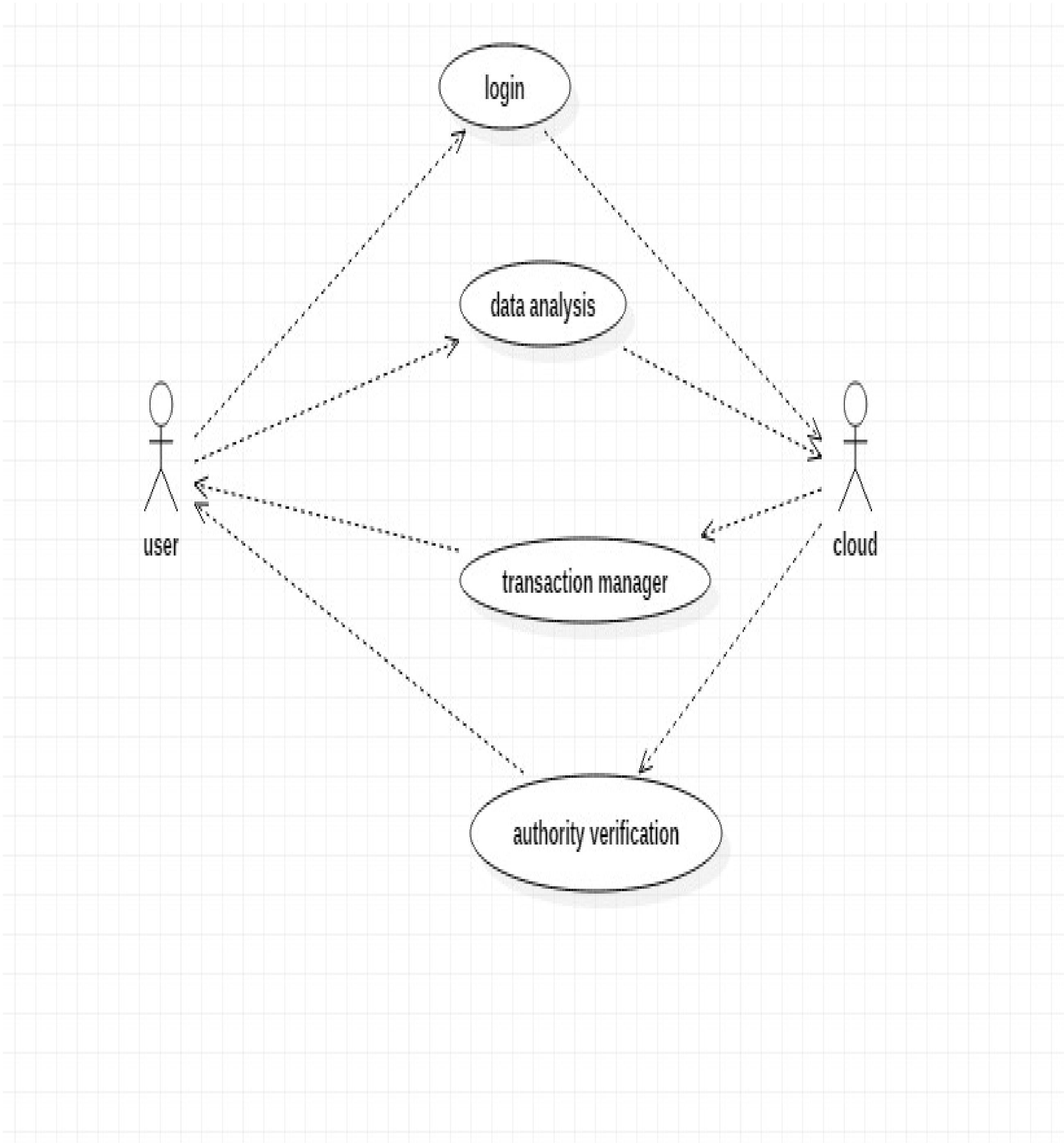


Fig:3.8 Use Case Diagram

Data Flow Diagram:

Data flow diagrams can be used to provide a clear representation of any business function. The technique starts with an overall picture of the business and continues by analyzing each of the functional areas of interest.

Level-0:

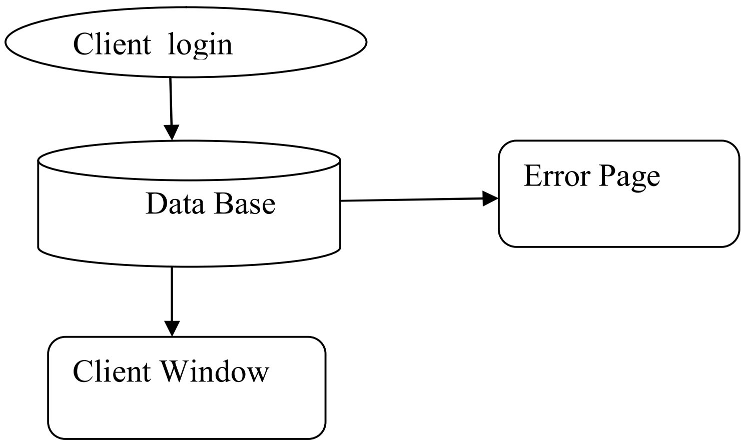


Fig:3.9(a) Data Flow Diagrams(level-0)

Level-1:

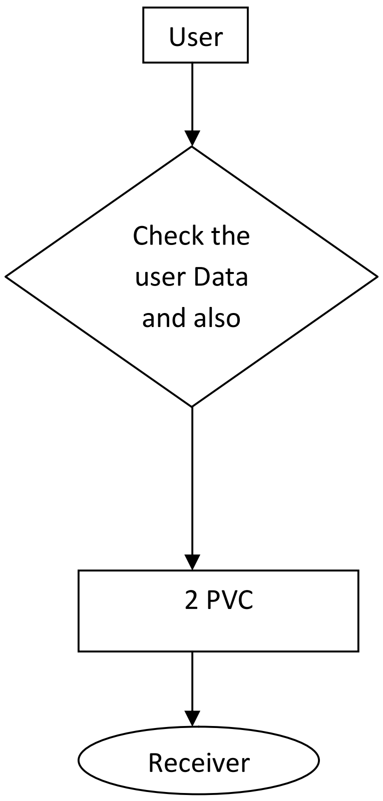


Fig:3.9(b) Data Flow Diagrams(level-1)

Sequence Diagram:

A sequence diagram in 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-trace diagrams, event scenarios, and timing diagrams. These diagrams must tells about each and actions performed by the particular users. Their shows the operation performed systems.

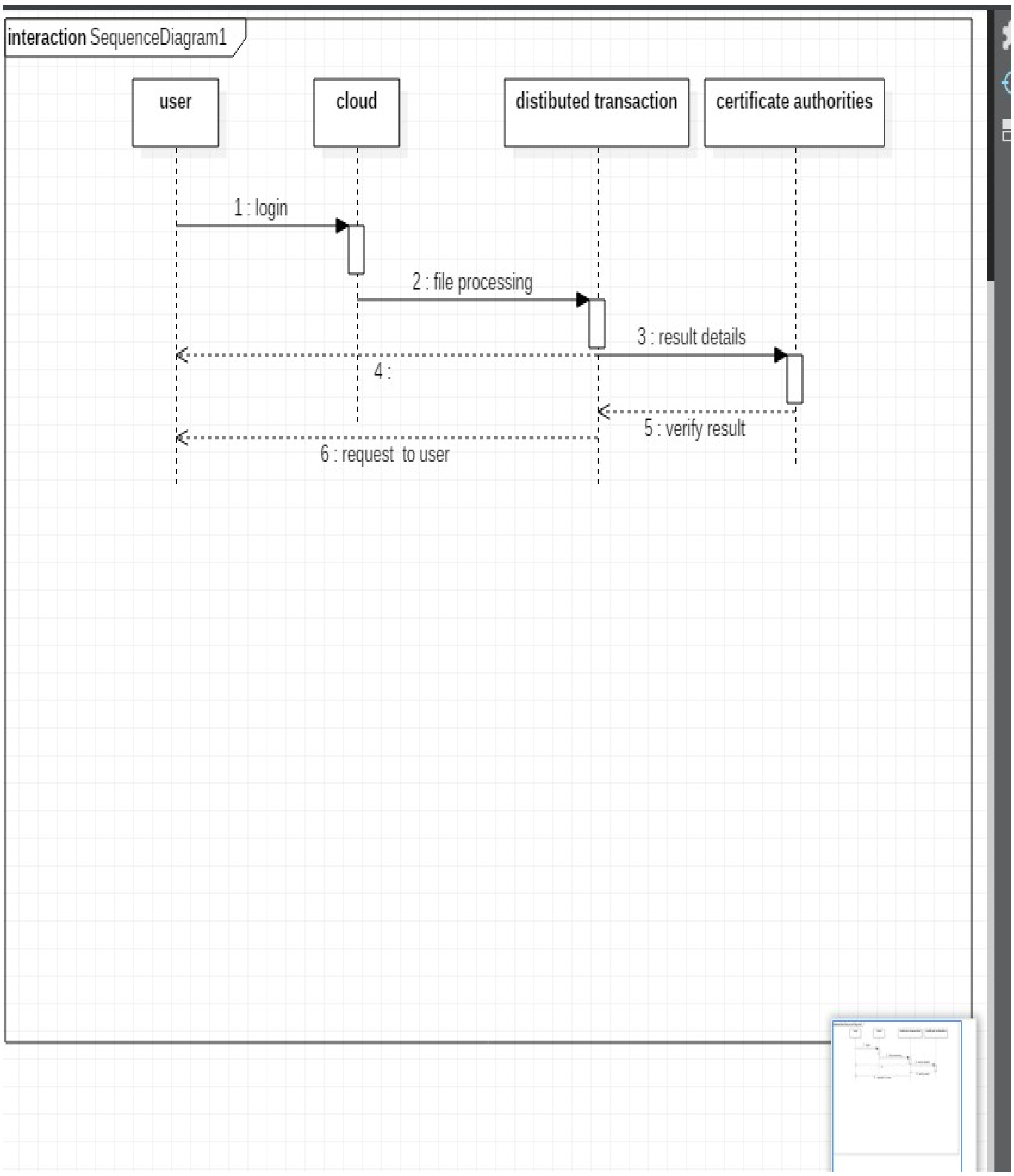


Fig:3.10 Sequence Diagram

Collaboration Diagram:

A collaboration diagram show the objects and relationships involved in an interaction, and the sequence of messages exchanged among the objects during the interaction. The collaboration diagram can be a decomposition of a class, class diagram , or part of a class diagram. it can be the decomposition of a use case, use case diagram, or part of a use case diagram. The collaboration diagram shows messages being sent between classes and object(instances). A diagram is created for each system operation that relates to the current development cycle(iteration).

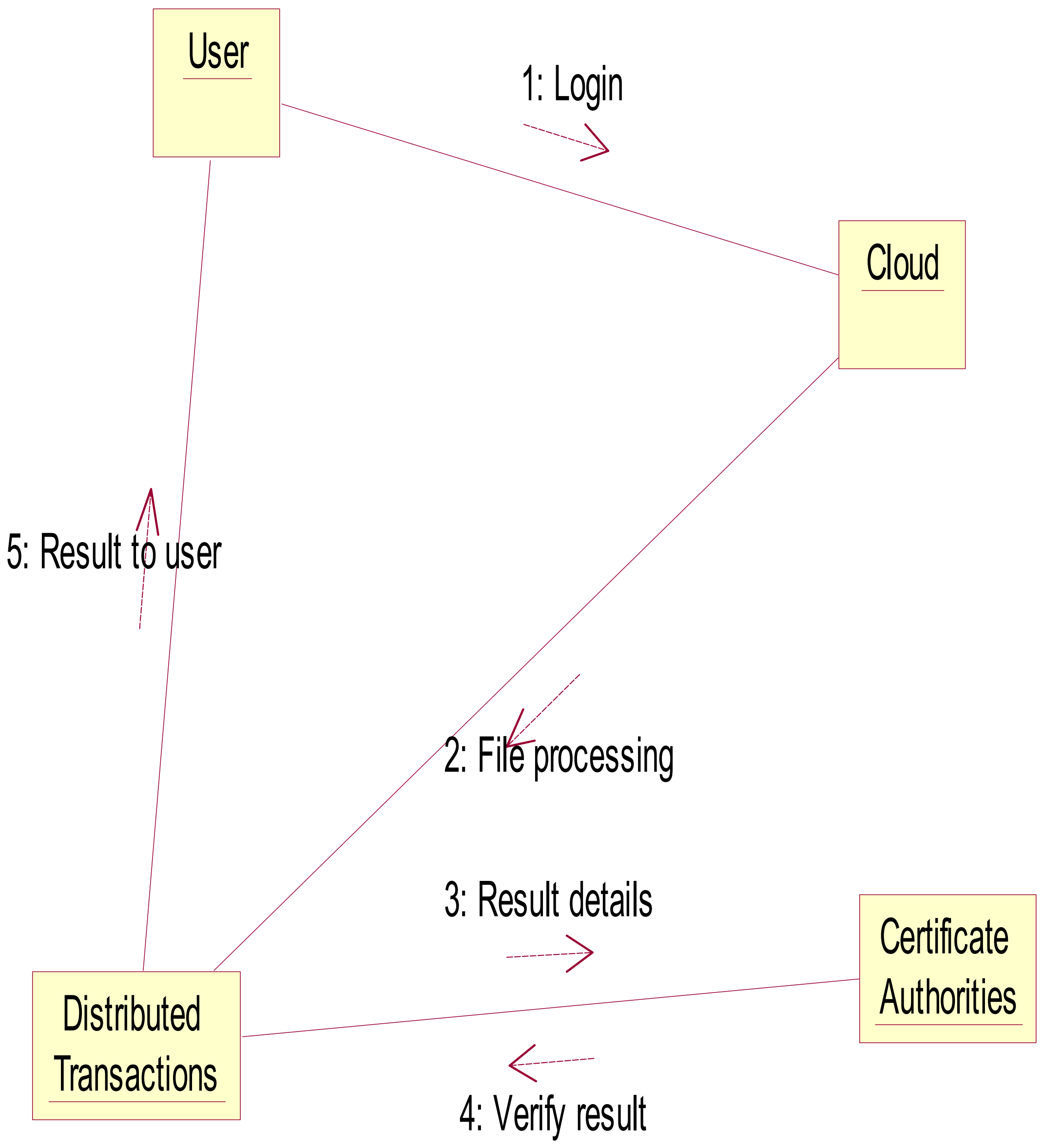


Fig:3.11 Collaboration Diagram

Class Diagram:

A class diagram in the UML, is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, and the relationships between the classes. Private visibility hides information from anything outside the class partition. Protected visibility allows child classes to access information they inherited from a parent class

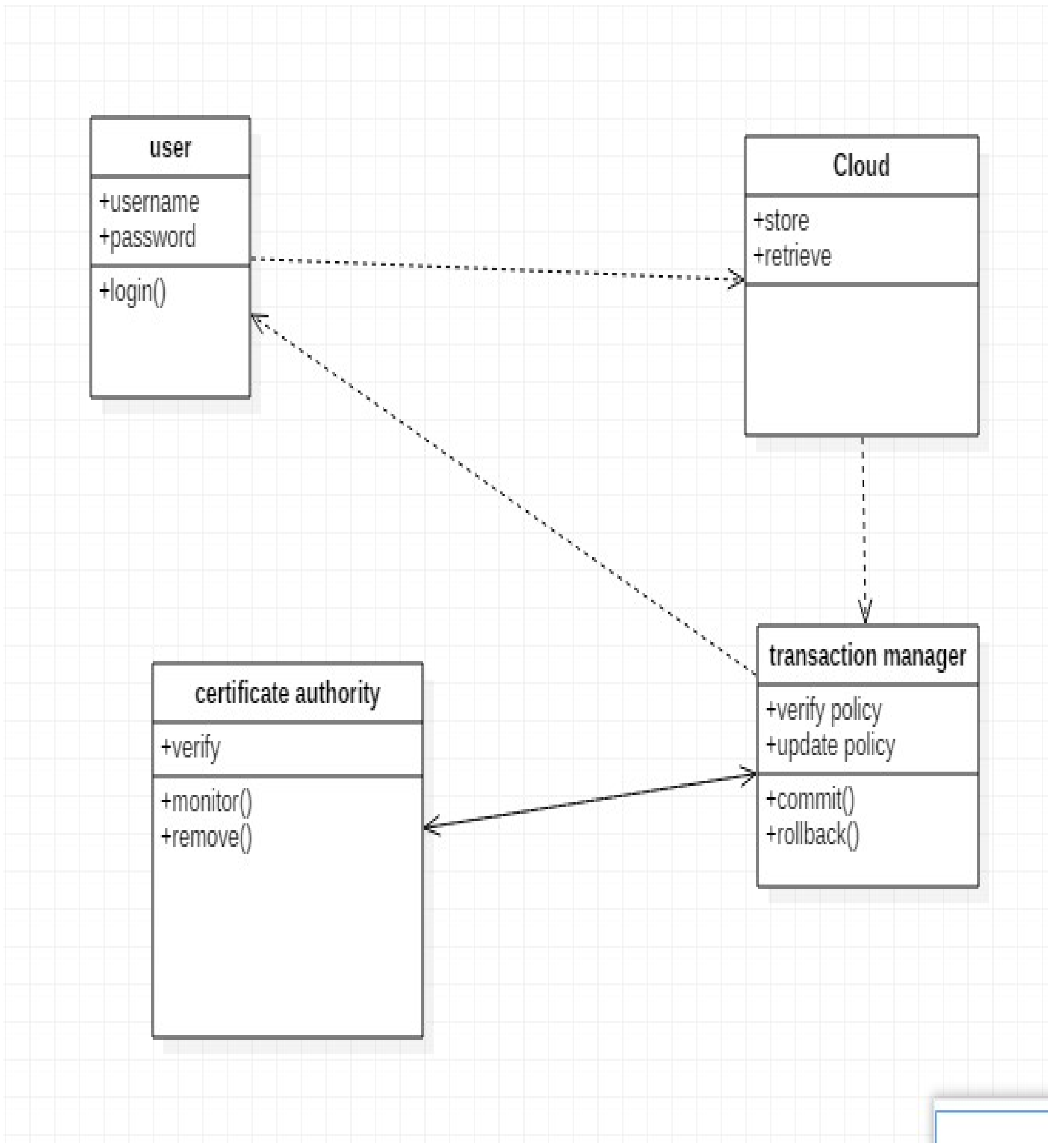


Fig:3.12 Class Diagram

E-R Diagram:

An entity relationship diagram shows the reationnships of entity sets stored in a database . An entity in this context is a component of data. In other words ,ER diagrams illustrate the logical structure of databases.

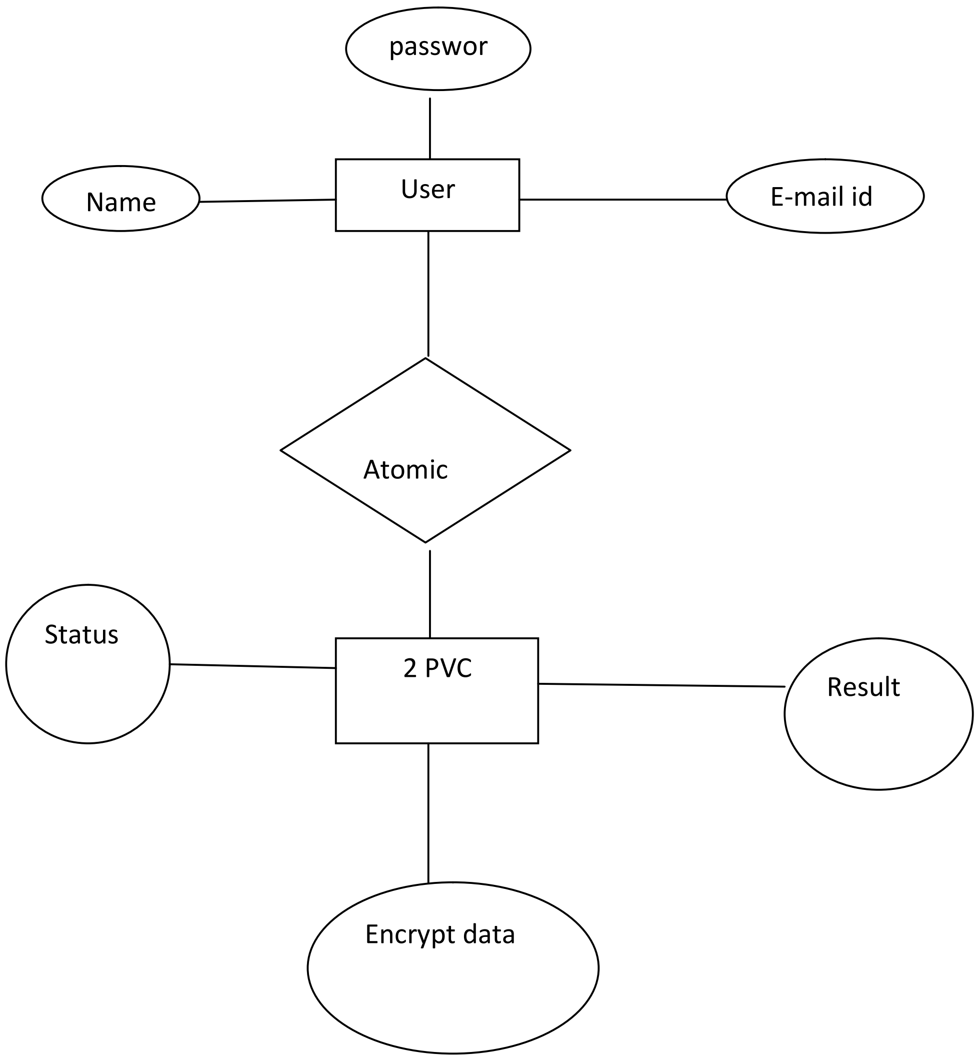


Fig:3.13 Entity-Relationship Diagram

Gantt Chart:

NAME OF THE TASK:

Task1: Paper Analysis & Discussion and Module Separation & GUI Design

Task2: Module Implementation

Task3: 3rd and 4th modules implementation

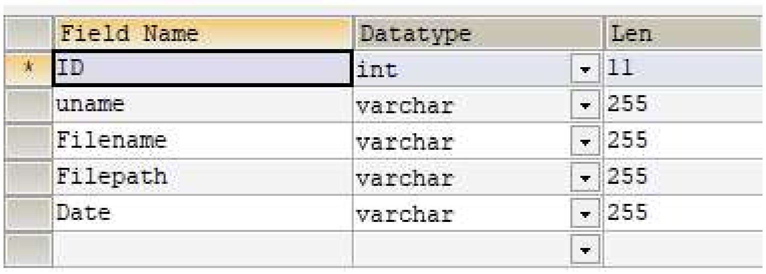
Task4: 5th and future enhancement implementation

	AUG-SEP	SEP-NOV	NOV-JAN	FEB-MAR	MAR
TASK1	🟩				
TASK2		🟩			
TASK3			🟨		
TASK4				🟨	
TASK5					🟨

Task5: Document Preparation and Debugging

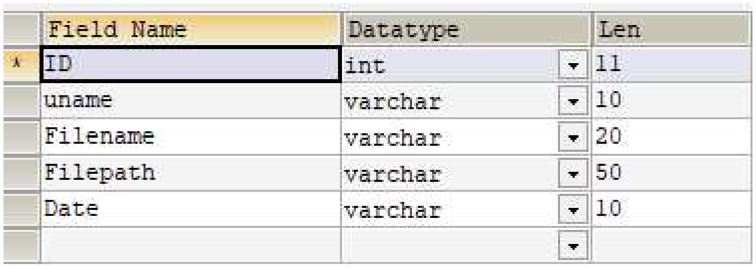
Fig:3.14 Gantt chart

3.4 Database Tables:



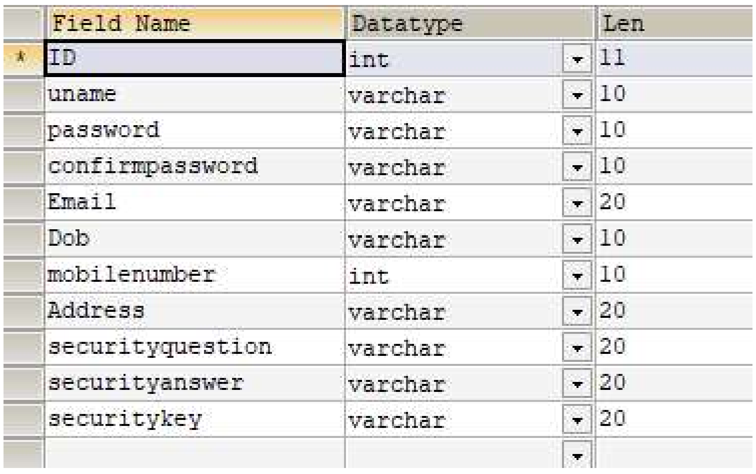
	Field Name	Datatype	Len
*	ID	int	11
	uname	varchar	255
	Filename	varchar	255
	Filepath	varchar	255
	Date	varchar	255

Fig:3.15 Admin data in admin cloud



	Field Name	Datatype	Len
*	ID	int	11
	uname	varchar	10
	Filename	varchar	20
	Filepath	varchar	50
	Date	varchar	10

Fig:3.16 (home/life/medial)databases in transactions



	Field Name	Datatype	Len
*	ID	int	11
	uname	varchar	10
	password	varchar	10
	confirmpassword	varchar	10
	Email	varchar	20
	Dob	varchar	10
	mobilenumber	int	10
	Address	varchar	20
	securityquestion	varchar	20
	securityanswer	varchar	20
	securitykey	varchar	20

Fig:3.17 login in transactions

4. ANALYSIS

4.1 System Requirements

4.1.1 General:

In this project we have successfully built a prototype supporting live migration of VMs between any two nodes on the Internet (even though they are behind different networks). Cloud computing has recently emerge (come into view) as a computing paradigm in which storage and computation (calculation) can be outsourced from organizations to next generation data centers hosted. In this paper to provide scalability and elasticity (flexibility), cloud services often make heavy use of replication to ensure consistent performance and availability this consistency model is a variant of weak consistency that allows data to be inconsistent among some replicas during the update process, but ensures that updates will eventually be propagated to all replicas

In the future, we will study fault-tolerance support for a SOC system; we will also conduct sensitivity analysis of how violation of our model assumptions would impact the optimal resource allocation.

4.1.2 Hardware Requirements:

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design. It shows what the system does and not how it should be implemented.

PROCESSOR	:	PENTIUM IV 2.6 GHz, Intel Core 2 Duo.
RAM	:	512 MB DD RAM
MONITOR	:	15" COLOR
HARD DISK	:	40 GB
CDDRIVE	:	LG 52X
KEYBOARD	:	STANDARD 102 KEYS
MOUSE	:	3 BUTTONS

4.1.3 Software Requirements:

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the team's and tracking the team's progress throughout the development activity.

FRONT END	:	JSP
BACK END	:	My SQL SERVER

4.1.4 Functional Requirements:

A functional requirement defines a function of a software-system or its component. A function is described as a set of inputs, the behavior, and outputs. The proposed system is achieved by suppression-based and generalization-based k-anonymous and confidential databases. The protocols rely on well-known cryptographic assumptions, and we provide theoretical analyses to proof their soundness and experimental results to illustrate their efficiency.

4.2 Software Description

4.2.1 General

This chapter is about the software language and the tools used in the development of the project. The platform used here is JAVA. The Primary languages are JAVA,J2EE and J2ME. In this project J2EE is chosen for implementation.

4.2.2 Features of Java

The features of java is mainly oops concepts

Inheritance: It is the process of creating the new classes and using the behavior of the existing classes by extending them just to reuse the existing code and adding addition a features as needed.

Encapsulation: It is the mechanism of combining the information and providing the abstraction.

Polymorphism: As the name suggest one name multiple form, Polymorphism is the way of providing the different functionality by the functions having the same name based on the signatures of the methods.

Dynamic binding: Sometimes we don't have the knowledge of objects about their specific types while writing our code. It is the way of providing the maximum functionality to a program about the specific type at runtime.

, Java has become invaluable to developers by enabling them to

- Write software on one platform and run it on virtually any other platform
- Create programs to run within a Web browser and Web services
- Develop server-side applications for online forums, stores, polls, HTML forms processing, and more
- Combine applications or services using the Java language to create highly customized applications or services
- Write powerful and efficient applications for mobile phones, remote processors, low-cost consumer products, and practically any other device with a digital heartbeat

4.2.3 The Java Framework

Java is a programming language originally developed by James Gosling at Sun Microsystems and released in 1995 as a core component of Sun Microsystems' Java platform.

The language derives much of its syntax from C and C++ but has a simpler object model and fewer low-level facilities. Java applications are typically compiled to byte code that can run on any Java Virtual Machine (JVM) regardless of computer architecture. Java is general-purpose, concurrent, class-based, and object-oriented, and is specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere".

Java is considered by many as one of the most influential programming languages of the 20th century, and is widely used from application software to web applications The java framework is a new platform independent that simplifies application development internet .Java technology's versatility, efficiency, platform portability, and security make it the ideal technology for network computing. From laptops to datacenters, game consoles to scientific supercomputers, cell phones to the Internet, Java is everywhere!

4.2.4 Objectives of Java

Why Software Developers Choose Java

Java has been tested, refined, extended, and proven by a dedicated community. And numbering more than 6.5 million developers, it's the largest and most active on the planet. With its versatilty, efficiency, and portability, Java has become invaluable to developers by enabling them to

- Write software on one platform and run it on virtually any other platform
- Create programs to run within a Web browser and Web services
- Develop server-side applications for online forums, stores, polls, HTML forms processing, and more
- Combine applications or services using the Java language to create highly customized applications or services
- Write powerful and efficient applications for mobile phones, remote processors, low-cost consumer products, and practically any other device with a digital heartbeat

Some Ways Software Developers Learn Java

Today, many colleges and universities offer courses in programming for the Java platform. In addition, developers can also enhance their Java programming skills by reading Sun's java.sun.com Web site, subscribing to Java technology-focused newsletters, using the Java Tutorial and the New to Java Programming Center, and signing up for Web, virtual, or instructor-led courses.

Object Oriented

To be an Object Oriented language, any language must follow at least the four characteristics.

1.Inheritance :It is the process of creating the new classes and using the behavior of the existing classes by extending them just to reuse the existing code and adding addition a features as needed.

2.Encapsulation: It is the mechanism of combining the information and providing the abstraction.

3.Polymorphism: As the name suggest one name multiple form, Polymorphism is the way of providing the different functionality by thefunctions having the same name based on the signatures of the methods.

4.Dynamic binding : Sometimes we don't have the knowledge of objects about their specific types while writing our code. It is the way of providing the maximum functionality to a program about the specific type at runtime.

4.2.5 Java Server Pages - An Overview

Java Server Pages or JSP for short is Sun's solution for developing dynamic web sites. JSP provide excellent server side scripting support for creating database driven web applications. JSP enable the developers to directly insert java code into jsp file, this makes the development process very simple and its maintenance also becomes very easy.

JSP pages are efficient, it loads into the web servers memory on receiving the request very first time and the subsequent calls are served within a very short period of time. In today's environment most web sites servers dynamic pages based on user request.

Database is very convenient way to store the data of users and other things. JDBC provide excellent database connectivity in heterogeneous database environment. Using JSP and JDBC its very cceasy to develop database driven web application.Java is known for its characteristic of "write once, run anywhere." JSP pages are platfJavaServer Pages

JavaServer Pages (JSP) technology is the Java platform technology for delivering dynamic content to web clients in a portable, secure and well-defined way.

the JavaServer Pages specification extends the Java Servlet API to provide web application developers with a robust framework for creating dynamic web content on the server using HTML, and XML templates, and Java code, which is secure, fast, and independent of server platforms.

JSP has been built on top of the Servlet API and utilizes Servlet semantics. JSP has become the preferred request handler and response mechanism. Although JSP technology is going to be a powerful successor to basic Servlets, they have an evolutionary relationship and can be used in a cooperative and complementary manner.

Servlets are powerful and sometimes they are a bit cumbersome when it comes to generating complex HTML. Most servlets contain a little code that handles application logic and a lot more code that handles output formatting. For these reasons, web application developers turn towards JSP as their preferred servlet environment.

4.2.6 Evolution of Web Applications

Over the last few years, web server applications have evolved from static to dynamic applications. This evolution became necessary due to some deficiencies in earlier web site design. For example, to put more of business processes on the web, whether in business-to-consumer (B2C) or business-to-business (B2B) markets, conventional web site design technologies are not enough. The main issues, every developer faces when developing web applications, are:

1. **Scalability** - a successful site will have more users and as the number of users is increasing fastly, the web applications have to scale correspondingly.
2. **Integration of data and business logic** - the web is just another way to conduct business, and so it should be able to use the same middle-tier and data-access code.
3. **Manageability** - web sites just keep getting bigger and we need some viable mechanism to manage the ever-increasing content and its interaction with business systems.
4. **Personalization** - adding a personal touch to the web page becomes an essential factor to keep our customer coming back again. Knowing their preferences, allowing them to configure the information they view, remembering their past transactions or

frequent search keywords are all important in providing feedback and interaction from what is otherwise a fairly one-sided conversation. The main characteristics of today's dynamic web server applications are as follows:

1. Serve HTML and XML, and stream data to the web client
2. Separate presentation, logic and data
3. Interface to databases, other Java applications, CORBA, directory and mail services
4. Make use of application server middleware to provide transactional support.
5. Track client sessions .

4.2.7 Benefits of JSP

One of the main reasons why the JavaServer Pages technology has evolved into what it is today and it is still evolving is the overwhelming technical need to simplify application design by separating dynamic content from static template display data. Another benefit of utilizing JSP is that it allows to more cleanly separate the roles of web application/HTML designer from a software developer.

The JSP technology is blessed with a number of exciting benefits, which are chronicled as follows:

The JSP technology is platform independent, in its dynamic web pages, its web servers, and its underlying server components.These components can be combined or manipulated towards developing more purposeful components and page design. This definitely reduces development time apart from the At development time, JSPs are very different from Servlets, however, they are precompiled into Servlets at run time and executed by a JSP engine which is installed on a Web-enabled application server such as BEA WebLogic and IBM WebSphere.

3.2.8 Servlets

Earlier in client- server computing, each application had its own client program and it worked as a user interface and need to be installed on each user's personal computer. Most web applications use HTML/XHTML that are mostly supported by all the browsers and web pages are displayed to the client as static documents.

A web page can merely displays static content and it also lets the user navigate through the content, but a web application provides a more interactive experience. Any computer running Servlets or JSP needs to have a container. A container is nothing but a piece of software responsible for loading, executing and unloading the Servlets and JSP. While servlets can be used to extend the functionality of any Java-enabled server. They are mostly used to extend web servers, and are efficient replacement for CGI scripts. CGI was one of the earliest and most prominent server side dynamic content solutions, so before going forward it is very important to know the difference between CGI and the Servlets.

4.2.9 Java Servlets

Java Servlet is a generic server extension that means a java class can be loaded dynamically to expand the functionality of a serverServlets are used with web servers and run inside a Java Virtual Machine (JVM) on the server so these are safe and portable. Servlets are also portable and platform independent.A web server is the combination of computer and the program installed on it.

. A web page can merely displays static content and it also lets the user navigate through the content, but a web application provides a more interactive experience Web server interacts with the client through a web browser. It delivers the web pages to the client and to an application by using the web browser and he HTTP protocols respectively.The define the web server as the package of large number of programs installed on a computer connected to Internet or intranet for downloading the requested files using File Transfer Protocol, serving e-mail and building and publishing web pages. A web server works on a client server model.

4.2.10 Conclusion

JSP and Servlets are gaining rapid acceptance as means to provide dynamic content on the Internet. With full access to the Java platform, running from the server in a secure manner, the application possibilities are almost limitless. J2EE technology as a whole makes it easy to develop, deploy and use web server applications instead of mingling with other technologies such as CGI and ASP

5. IMPLEMENTATION AND TESTING

5.1 General

This chapter describes implementation Creating a Contrary to existing solutions which often generate bulky messages per request, our protocol produces only one lightweight query message per task on the Content Addressable Network. It works effectively to find for each task its qualified resources under a randomized policy that mitigates the contention among requesters. We show the SOC with our optimized algorithms can make an improvement by increase than the existing percentage in system throughput than a P2P Grid model.

5.2 Coding

Login page:

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>


<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
 <title>ARaynorDesign Template</title>
  <meta name="description" content="free website template" />
  <meta name="keywords" content="enter your keywords here" />
  <meta http-equiv="content-type" content="text/html; charset=utf-8" />
  <link rel="stylesheet" type="text/css" href="css/style.css" />
  <script type="text/javascript" src="js/jquery.min.js"></script>
  <script type="text/javascript" src="js/jquery.easing.min.js"></script>
  <script type="text/javascript" src="js/jquery.nivo.slider.pack.js"></script>
  <script type="text/javascript">
  $(window).load(function() {
     $('#slider').nivoSlider();
  });
  </script>
</head>

<body>
  <div id="main">

                 <div id="header">
                   <div id="menubar">
                     <div id="welcome">
                       <h1><a href="#">Welcome To Clouds</a></h1>
                     </div><!--close welcome-->
        <div id="menu_items">
                       <ul id="menu">
            <li class="current"><a href="index.jsp">Home</a></li>
            <li><a href="admin.jsp">Admin</a></li>
            <li><a href="testimonials.jsp">Testimonials</a></li>
            <li><a href="projects.jsp">Projects</a></li>
            <li><a href="contact.jsp">Contact Us</a></li>
          </ul>
        </div><!--close menu-->
      </div><!--close menubar-->
                 </div><!--close header-->
```

```
                <div id="site_content">


                 <div id="banner_image">
                   <div id="slider-wrapper">
         <div id="slider" class="nivoSlider">
           <img src="images/home_1.jpg" alt="" />
           <img src="images/home_2.jpg" alt="" />
           <img src="images/home_3.jpg" alt="" />
           <img src="images/home_4.jpg" alt="" />
                      </div><!--close slider-->
                    </div><!--close slider_wrapper-->
                 </div><!--close banner_image-->


                 <div class="sidebar_container">
                    <div class="sidebar">
         <div class="sidebar_item">


         </div><!--close sidebar_item-->
       </div><!--close sidebar-->


     </div><!--close sidebar_container-->


<div id="content">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
```

```
<title>Insert title here</title>

<script >
function validateForm()
{
   if(document.myform.uname.value=="")
   {
     alert("User Name should be left blank");
     document.myform.uname.focus();
     return false;
   }
   else if(document.myform.pwd.value=="")
   {
     alert("Password should be left blank");
     document.myform.pwd.focus();
     return false;
   }
}
```

Register page:

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
              pageEncoding="ISO-8859-1"%>
           <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
           "http://www.w3.org/TR/html4/loose.dtd">
           <html>
           <head>
           <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-
           1">
```

```
 <title>ARaynorDesign Template</title>
  <meta name="description" content="free website template" />
  <meta name="keywords" content="enter your keywords here" />
  <meta http-equiv="content-type" content="text/html; charset=utf-8" />
  <link rel="stylesheet" type="text/css" href="css/style.css" />
  <script type="text/javascript" src="js/jquery.min.js"></script>
  <script type="text/javascript" src="js/jquery.easing.min.js"></script>
  <script type="text/javascript" src="js/jquery.nivo.slider.pack.js"></script>
  <script type="text/javascript">
    $(window).load(function() {
        $('#slider').nivoSlider();
    });
  </script>

  <script>
function validateForm()
{



    if(document.frm.rname.value=="")
    {
      alert("Name should be left blank");
      document.frm.rname.focus();
      result=false;

    }
    else if(document.frm.runame.value=="")
    {
      alert("User Name should be left blank");
      document.frm.runame.focus();
      result=false;

    }
    else if(document.frm.rpassword.value=="")
```

```
{
  alert("Password should be left blank");
  document.frm.rpassword.focus();
  result=false;

}
else if(document.frm.rcpassword.value=="")
{
  alert("confirm Password should be left blank");
  document.frm.rcpassword.focus();
  result=false;

}
/*  else if(document.frm.pwd!=document.frm.cpwd.value)
{
  alert("confirm Password should be equal to password");
  document.frm.cpwd.focus();
  result=false;

}
else if(document.frm.age.value=="")
{
  alert("age should not be left blank");
  document.frm.age.focus();
  result=false;

}
else if(document.frm.age.value>=100)
{
  alert("age should  be less than hundred years");
  document.frm.age.focus();
  result=false;

}
```

```
else if(document.frm.age.value<=0)
{
  alert("age should not be negative values");
  document.frm.age.focus();
  result=false;

}*/
else if(document.frm.rmobile.value=="")
{
  alert("mobile number should not be blank");
  document.frm.rmobile.focus();
  result=false;

}
else if(isNaN(document.frm.rmobile.value))
{
  alert("mobile number should  be Number");
  document.frm.rmobile.focus();
  result=false;

}
```

Implementation of Optimal Resources.

```
package user;

import java.io.IOException;
import java.util.ArrayList;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
```

```
/**
 * Servlet implementation class Admincloud
 */
public class Admincloud extends HttpServlet {
	private static final long serialVersionUID = 1L;

   /**
    * @see HttpServlet#HttpServlet()
    */
   public Admincloud() {
      super();
      // TODO Auto-generated constructor stub
   }

	/**
	 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse
response)
	 */
	protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
		// TODO Auto-generated method stub
	}

	/**
	 * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse
response)
	 */
	protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
		// TODO Auto-generated method stub
		ArrayList al=new ArrayList();
		al.add(request.getParameter("aname"));
		al.add(request.getParameter("apass"));
		System.out.println(al);
```

```
                String adminname=al.get(0).toString();
                String adminpassword=al.get(1).toString();


        if(adminname.equalsIgnoreCase("admin")&&adminpassword.equalsIgnoreCa
se("admin"))
                {
                        response.sendRedirect("welcome.jsp");
                }
                else
                {
                        response.sendRedirect("admin.jsp");
                }

        }

}
```

Cloud resource management

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">

<head>
  <title>ARaynorDesign Template</title>
  <meta name="description" content="free website template" />
  <meta name="keywords" content="enter your keywords here" />
  <meta http-equiv="content-type" content="text/html; charset=utf-8" />
  <link rel="stylesheet" type="text/css" href="css/style.css" />
  <script type="text/javascript" src="js/jquery.min.js"></script>
  <script type="text/javascript" src="js/jquery.easing.min.js"></script>
  <script type="text/javascript" src="js/jquery.nivo.slider.pack.js"></script>
  <script type="text/javascript">
    $(window).load(function() {
      $('#slider').nivoSlider();
    });
  </script>
```

Optimal Resource allocation

```
package user;


import java.io.File;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.ArrayList;


import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;


/**
 * Servlet implementation class Client1Search
 */
public class Client1Search extends HttpServlet {
        private static final long serialVersionUID = 1L;


   /**
    * @see HttpServlet#HttpServlet()
    */
   public Client1Search() {
      super();
      // TODO Auto-generated constructor stub
   }


        /**
```

```
         * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
         */
        protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
                // TODO Auto-generated method stub
        }

        /**
         * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
         */
        protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

     /*String query=request.getParameter("query");

                String res=dbpack.Client1DB.fileSearch(query);

                if(res.equals("file found"))
                {
                        response.sendRedirect("results.jsp");
                }
                else
                        response.sendRedirect("welcome.jsp");
                */
```

DataBase:

```
package dbpack;

import java.sql.Connection;

import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;

public class Client1DB {
        static Statement st=null;
        public static Connection con()
        {
                Connection dbcon = null;
                try{

                Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

dbcon=DriverManager.getConnection("Jdbc:Odbc:SelfOrganizing");
                /* st=dbcon.createStatement();*/

                }
                catch(Exception e)
                {e.printStackTrace();}

                 return dbcon;
        }

        public static int        register(ArrayList           al)           throws
ClassNotFoundException, SQLException
        {
        ResultSet rs;
        String rname=al.get(0).toString();
        String runame=al.get(1).toString();
        String rpwd=al.get(2).toString();
        String rcpwd=al.get(3).toString();
        String rgender=al.get(4).toString();
        String rage=al.get(5).toString();
        String rmobile=al.get(6).toString();
String remail=al.get(7).toString();
int i=0;
try
{
        rs=con().createStatement().executeQuery("select username
from client1registration where username='"+runame+"'");
        if(rs.next())
        {System.out.println("avoid duplicates");}
        else{
 PreparedStatement ps=con().prepareStatement("insert into
client1registration values(?,?,?,?,?,?,?,?)");
 ps.setString(1, rname);
 ps.setString(2, runame);
 ps.setString(3, rpwd);
 ps.setString(4, rcpwd);
 ps.setString(5, rgender);
 ps.setString(6, rage);
 ps.setString(7, rmobile);
 ps.setString(8, remail);
 i=ps.executeUpdate();

}
}catch(Exception e)
{
        e.printStackTrace();
}




return i;
}
public static int authentic(ArrayList al)
{
```

```
                    int count=0;
                    String runame=al.get(0).toString();
                    String rpwd=al.get(1).toString();

                    try{
                            Statement st=con().createStatement();
                            ResultSet        rs=st.executeQuery("select      *      from
client1registration where username='"+runame+"' and password='"+rpwd+"'");

                    while(rs.next())
                            {
                    count++;
                    }

                }
                    catch(Exception e)
                    {
                            e.printStackTrace();
                    }

                    return count;
            }

            public static String fileSearch(String filename)
            {
                    String res="file not found";
                    try{
                            Statement st=con().createStatement();
                            ResultSet rs=st.executeQuery("select * from cloud_data
where filename='"+filename+"' ");

                    while(rs.next())
                            {
                    res="file found";
                    }

                }
                    catch(Exception e)
                    {
                            e.printStackTrace();
                    }
            return res;

            }

  }
```

5.3 Testing

5.3.1 General

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.

5.3.2 Developing Methodologies

The test process is initiated by developing a comprehensive plan to test the general functionality and special features on a variety of platform combinations. Strict quality control procedures are used.

The process verifies that the application meets the requirements specified in the system requirements document and is bug free. The following are the considerations used to develop the framework from developing the testing methodologies.

5.3.3 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 input 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.

. 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

.

Functional Testing:

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.

System Testing:

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results.

An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

Performance Testing:

The Performance test ensures that the output be produced within the time limits,and the time taken by the system for compiling, giving response to the users and request being send to the system for to retrieve the results.

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.

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.

Acceptance testing for Data Synchronization:

- The Acknowledgements will be received by the Sender Node after the Packets are received by the Destination Node
- The Route add operation is done only when there is a Route request in need
- The Status of Nodes information is done automatically in the Cache Updation process

Build the test plan:

Any project can be divided into units that can be further performed for detailed processing. Then a testing strategy for each of this unit is carried out.

Unit testing helps to identity the possible bugs in the individual component, so the component that has bugs can be identified and can be rectified from errors.

Table:5.1 Test Case For login:

Test case id	Test Scenario	Test Steps	Expected Result	Result

TC1	Check login name with invalid data	Enter
		1.Username
		2.password
		3.Enter key
			Username=user1
			Password=user1
			Enter key=12345
				Enter Valid Details

TC2	Check login name with invalid data	Enter
		1.Username
		2.password
		3.Enter key
			Username=user1
			Password=user1
			Enter
			key=5752807
				Successful

Table:5.2 Test Case For Admin:

Test case id	Test states	Test data	Result

TC1	Enter
	1.Username
	2.Password
		Username=admin
		Password=aadmin1
			Enter
			valid
			details

TC2	Enter
	1.Username
	2.Password
		Username=admin
		Password=admin
			successful

Table:5.2 Test Case For Registration:

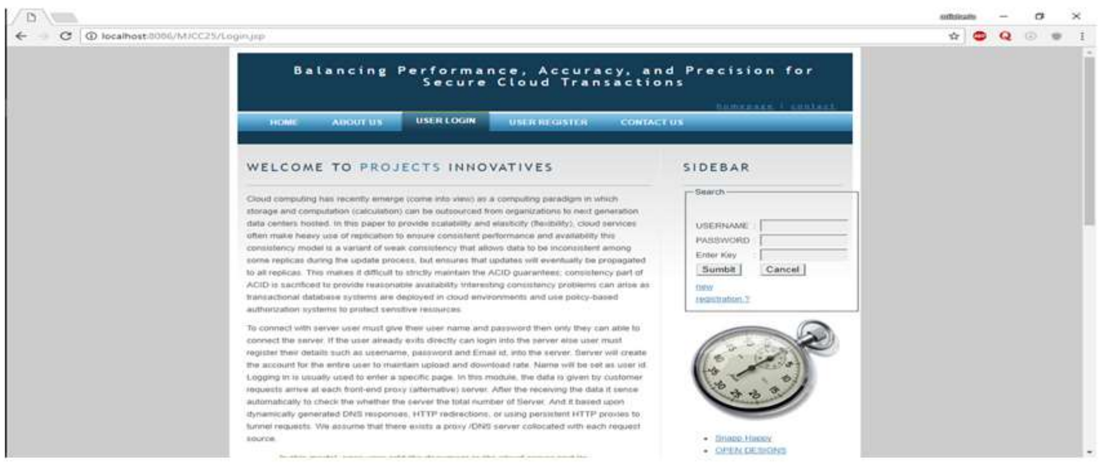
Test scenario	Test steps	Test data	Expected result
Check user registration with vaild input	Enter the details 1.Username 2.Password 3.Conform 4.Email 5.D0B 6.Security question 7.Enteranswr 8.Mobile no 9.address	1.Abc 2.13450 3.13450 4.abc@gmail.com 5.8 -10-11 6.What is Favorite Sports 7. Crckt 8. 9900889900 9.hyd	Your securitykey *******
Checkuser registration with invaild input	Enter the details 1.Username 2.Password 3.Conform 4.Email 5.D0B 6.Security question 7.Enteranswr 8.Mobile no 9.address	1.Abc 2.abcabc 3.abcab 4.abjkc@gmail.com 5.8 -10-9 6.What is Favorite Sports 7. Crckt 8. 9900889900 9.hyd	Login failed..

6. RESULTS

6.1 General

Snapshot is nothing but every moment of the application while running. It gives the clear elaborated of application. It will be useful for the new user to understand for the future steps

6.2 Various Screenshots


localhost:8086/MJCC25/Login.jsp

Balancing Performance, Accuracy, and Precision for Secure Cloud Transactions

WELCOME TO PROJECTS INNOVATIVES

Cloud computing has recently emerge (come into view) as a computing paradigm in which storage and computation (calculation) can be outsourced from organizations to next generation data centers hosted. In this paper to provide scalability and elasticity (flexibility), cloud services often make heavy use of replication to ensure consistent performance and availability this consistency model is a variant of weak consistency that allows data to be inconsistent among some replicas during the update process, but ensures that updates will eventually be propagated to all replicas. This makes it difficult to strictly maintain the ACID guarantees; consistency part of ACID is sacrificed to provide reasonable availability interesting consistency problems can arise as transactional database systems are deployed in cloud environments and use policy-based authorization systems to protect sensitive resources.

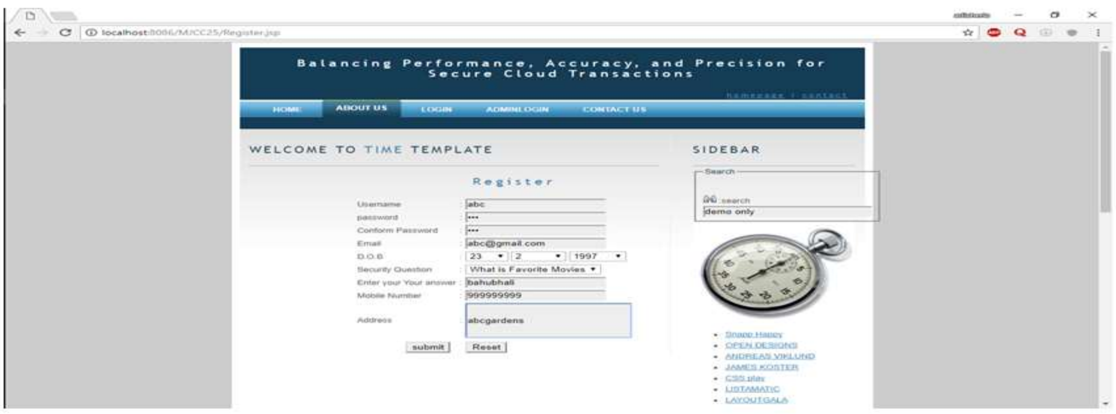
To connect with server user must give their user name and password then only they can able to connect the server. If the user already exits directly can login into the server else user must register their details such as username, password and Email id, into the server. Server will create the account for the entire user to maintain upload and download rate. Name will be set as user id. Logging in is usually used to enter a specific page. In this module, the data is given by customer requests arrive at each front-end proxy (alternative) server. After the receiving the data it sence automatically to check the whether the server the total number of Server. And it based upon dynamically generated DNS responses, HTTP redirections, or using persistent HTTP proxies to tunnel requests. We assume that there exists a proxy /DNS server collocated with each request source.

SIDEBAR

Search

USERNAME :
PASSWORD :
Enter Key :

Fig:6.1 Login page-Screen Shot


localhost:8086/MJCC25/Register.jsp

Balancing Performance, Accuracy, and Precision for Secure Cloud Transactions

WELCOME TO TIME TEMPLATE

Register

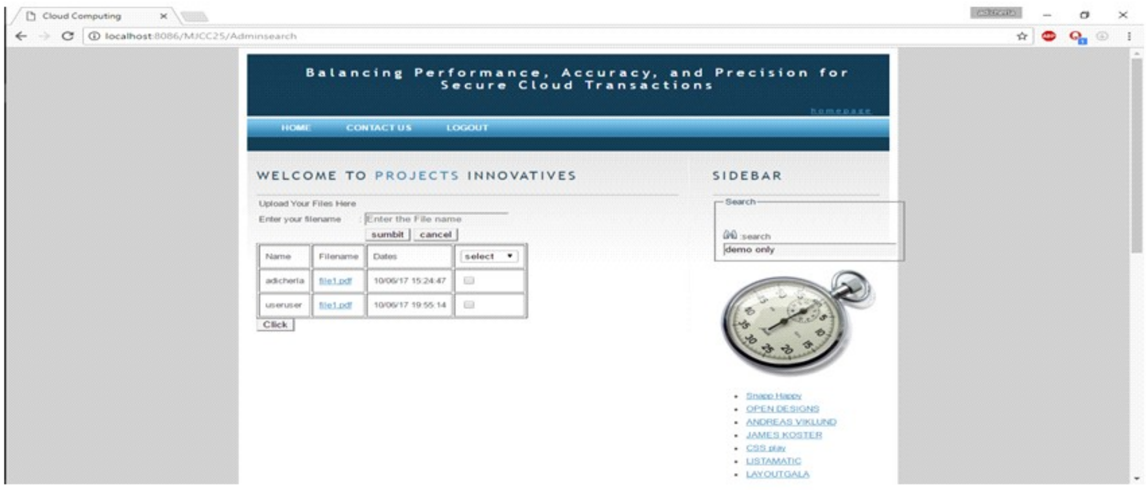
Username	abc
password	***
Conform Password	***
Email	abc@gmail.com
D.O.B	23 ▾ 2 ▾ 1997 ▾
Security Question	What is Favorite Movies ▾
Enter your Your answer	bahubhali
Mobile Number	999999999
Address	abcgardens

SIDEBAR

Search

demo only

Fig:6.2 Registration page-Screen Shot



Cloud Computing

localhost:8086/MJCC25/Adminsearch

Balancing Performance, Accuracy, and Precision for Secure Cloud Transactions

homepage

HOME CONTACT US LOGOUT

WELCOME TO PROJECTS INNOVATIVES

Upload Your Files Here

Enter your filename : Enter the File name

sumbit cancel

Name	Filename	Dates	select
adicherla	file1.pdf	10/06/17 15:24:47	☐
useruser	file1.pdf	10/06/17 19:55:14	☐

Click

SIDEBAR

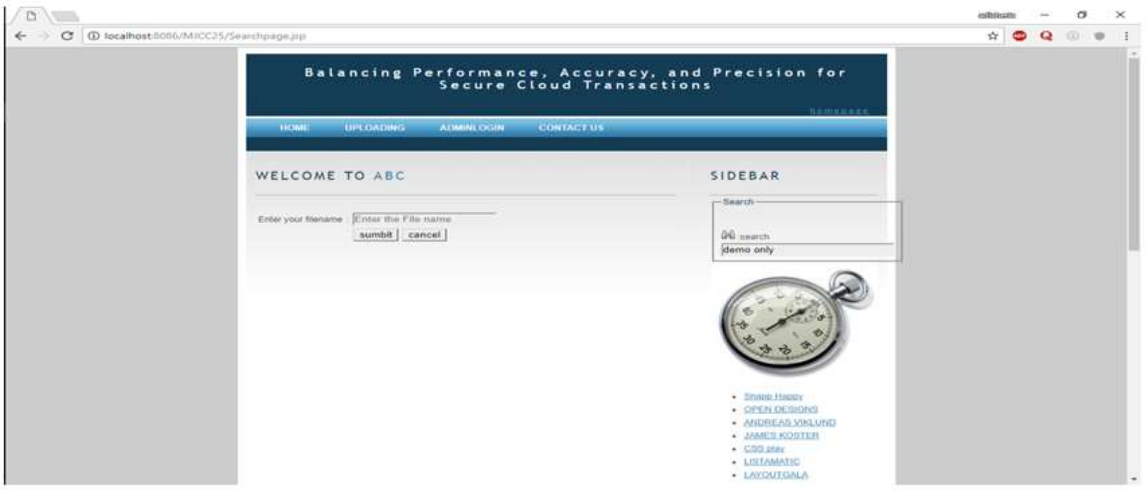
Search

search

demo only

- Snapp Happy
- OPEN DESIGNS
- ANDREAS VIKLUND
- JAMES KOSTER
- CSS play
- LISTAMATIC
- LAYOUTGALA

Fig:6.3 Admin login page-Screenshot



localhost:8086/MJCC25/Searchpage.jsp

Balancing Performance, Accuracy, and Precision for Secure Cloud Transactions

homepage

HOME UPLOADING ADMINLOGIN CONTACT US

WELCOME TO ABC

Enter your filename Enter the File name

sumbit cancel

SIDEBAR

Search

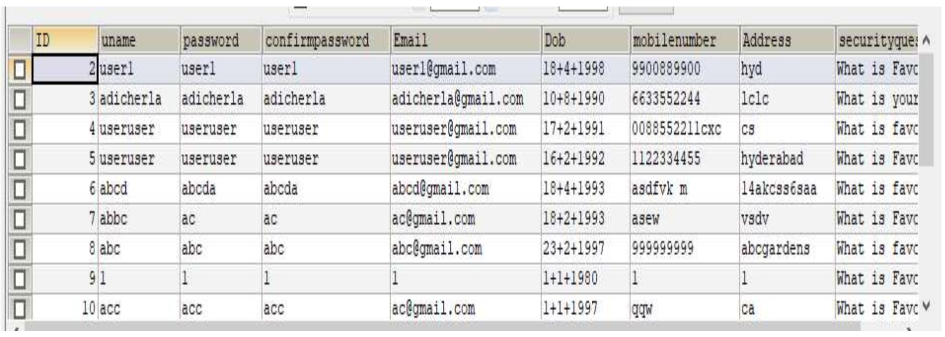
search

demo only

- Snapp Happy
- OPEN DESIGNS
- ANDREAS VIKLUND
- JAMES KOSTER
- CSS play
- LISTAMATIC
- LAYOUTGALA

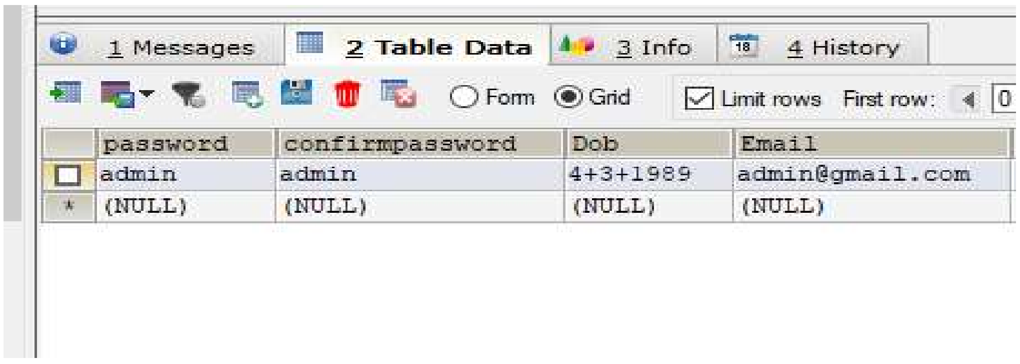
Fig:6.4 User file search page-Screenshot

Database screenshots:



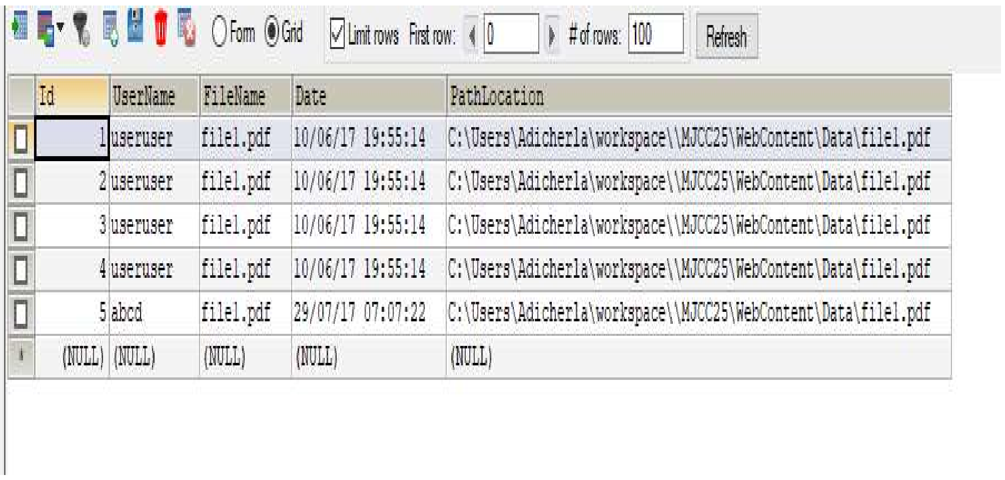
ID	uname	password	confirmpassword	Email	Dob	mobilenumber	Address	securityques
2	user1	user1	user1	user1@gmail.com	18+4+1998	9900889900	hyd	What is Favo
3	adicherla	adicherla	adicherla	adicherla@gmail.com	10+8+1990	6633552244	lclc	What is your
4	useruser	useruser	useruser	useruser@gmail.com	17+2+1991	008855221lcxc	cs	What is favo
5	useruser	useruser	useruser	useruser@gmail.com	16+2+1992	1122334455	hyderabad	What is Favo
6	abcd	abcda	abcda	abcd@gmail.com	18+4+1993	asdfvk m	14akcss6saa	What is favo
7	abbc	ac	ac	ac@gmail.com	18+2+1993	asew	vsdv	What is Favo
8	abc	abc	abc	abc@gmail.com	23+2+1997	999999999	abcgardens	What is favo
9	1	1	1	1	1+1+1980	1	1	What is Favo
10	acc	acc	acc	ac@gmail.com	1+1+1997	qqw	ca	What is Favo

Fig:6.5 login database-Screen shot



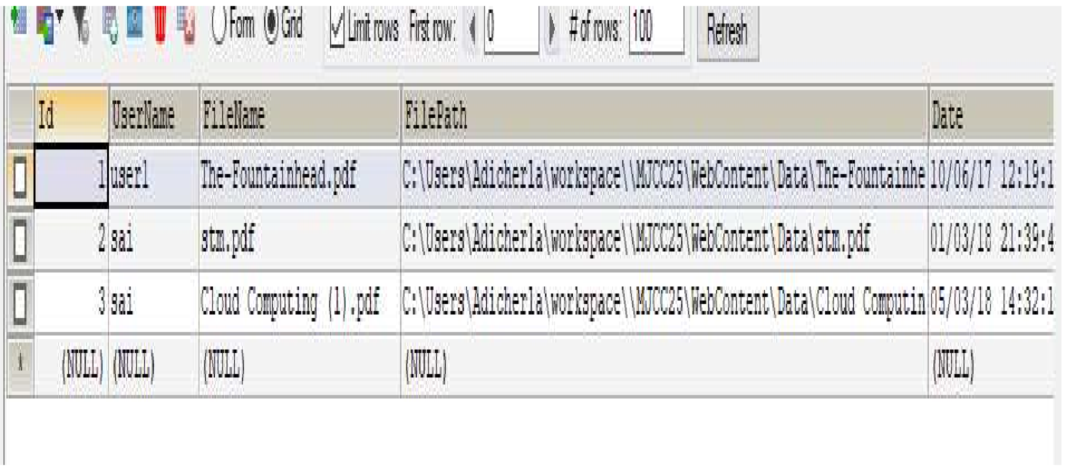
password	confirmpassword	Dob	Email
admin	admin	4+3+1989	admin@gmail.com
(NULL)	(NULL)	(NULL)	(NULL)

Fig:6.6 Admin database-Screen shot



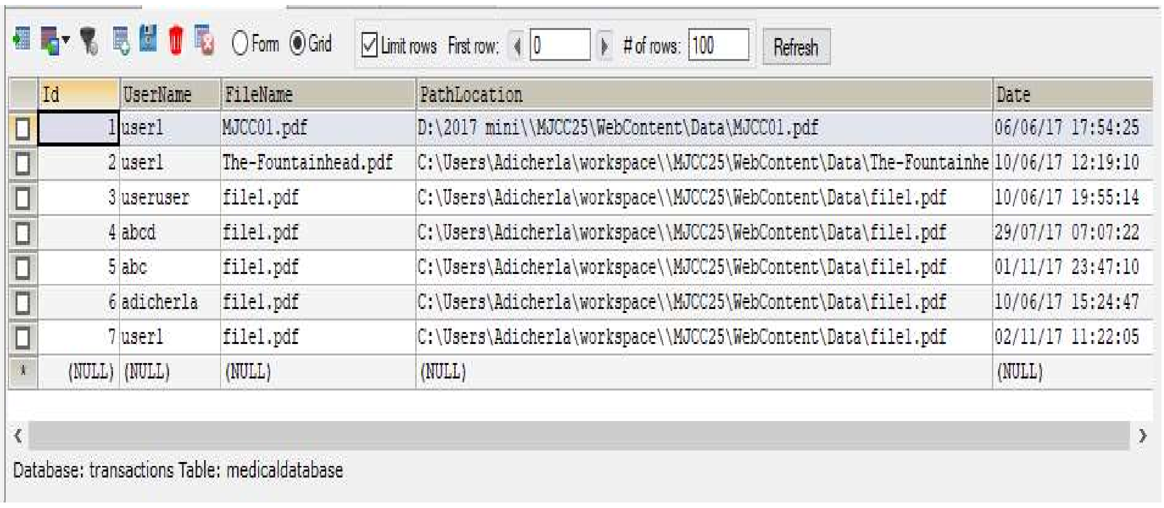
Id	UserName	FileName	Date	PathLocation
1	useruser	file1.pdf	10/06/17 19:55:14	C:\Users\Adicherla\workspace\\MJCC25\WebContent\Data\file1.pdf
2	useruser	file1.pdf	10/06/17 19:55:14	C:\Users\Adicherla\workspace\\MJCC25\WebContent\Data\file1.pdf
3	useruser	file1.pdf	10/06/17 19:55:14	C:\Users\Adicherla\workspace\\MJCC25\WebContent\Data\file1.pdf
4	useruser	file1.pdf	10/06/17 19:55:14	C:\Users\Adicherla\workspace\\MJCC25\WebContent\Data\file1.pdf
5	abcd	file1.pdf	29/07/17 07:07:22	C:\Users\Adicherla\workspace\\MJCC25\WebContent\Data\file1.pdf
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)

Fig:6.7 Life database-Screen shot



Id	UserName	FileName	FilePath	Date
1	user1	The-Fountainhead.pdf	C:\Users\Adicherla\workspace\\MJCC25\WebContent\Data\The-Fountainhe	10/06/17 12:19:1
2	sai	stm.pdf	C:\Users\Adicherla\workspace\\MJCC25\WebContent\Data\stm.pdf	01/03/18 21:39:4
3	sai	Cloud Computing (1).pdf	C:\Users\Adicherla\workspace\\MJCC25\WebContent\Data\Cloud Computin	05/03/18 14:32:1
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)

Fig:6.8 Home database-Screen shot



Form [✓ Grid] [✓ Limit rows] First row: 0 # of rows: 100 Refresh

Id	UserName	FileName	PathLocation	Date
1	user1	MJCC01.pdf	D:\2017 mini\\MJCC25\WebContent\Data\MJCC01.pdf	06/06/17 17:54:25
2	user1	The-Fountainhead.pdf	C:\Users\Adicherla\workspace\\MJCC25\WebContent\Data\The-Fountainhe	10/06/17 12:19:10
3	useruser	file1.pdf	C:\Users\Adicherla\workspace\\MJCC25\WebContent\Data\file1.pdf	10/06/17 19:55:14
4	abcd	file1.pdf	C:\Users\Adicherla\workspace\\MJCC25\WebContent\Data\file1.pdf	29/07/17 07:07:22
5	abc	file1.pdf	C:\Users\Adicherla\workspace\\MJCC25\WebContent\Data\file1.pdf	01/11/17 23:47:10
6	adicherla	file1.pdf	C:\Users\Adicherla\workspace\\MJCC25\WebContent\Data\file1.pdf	10/06/17 15:24:47
7	user1	file1.pdf	C:\Users\Adicherla\workspace\\MJCC25\WebContent\Data\file1.pdf	02/11/17 11:22:05
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)

Database: transactions Table: medicaldatabase

Fig:6.9 Medical database-Screen shot

7. CONCLUSION AND FUTURE ENHANCEMENT

7.1 Conclusion

In this work, we study the information to the different from most existing multi-attribute range query solutions that require propagating multiple sub queries along multiple dimensions in parallel. To mitigate the contention problem due to analogous queries in CAN, our range query protocol proactively diffuses resource indexes over the network and randomly route query messages among nodes to locate qualified ones that satisfy tasks' minimal demands.

7.2 Application

VM-Ware Resource Allocation:

To build a true private cloud, automated workload balancing within shared pools of resources delivers optimized resource usage. This ensures that companies get full resource utilization and the highest availability as needed by application and service loads. Continuously monitors utilization across resource pools and intelligently aligns resources with business needs, enabling you to:

- Dynamically allocate IT resources to the highest priority applications. Create rules and policies to prioritize how resources are allocated
- Give IT autonomy to business organizations. Provide dedicated IT infrastructure to business units while still achieving higher hardware utilization through resource pooling.
- Empower business units to build and manage virtual machines within their resource pool while giving central IT control over hardware resources.

7.3 Future Enhancements

We identified several consistency problems that can arise during cloud-hosted transaction processing using weak consistency models, particularly if policy-based authorization systems are used to enforce access controls. We used workloads to experimentally evaluate implementations of our proposed consistency models relative to three core metrics: transaction processing performance, accuracy, and precision. We found that high performance comes at a cost: Deferred and Punctual proofs had minimal overheads, but failed to detect certain types of consistency problems. And proofs were faster than view consistency ones in the few cases when the latest policy happens to match the policy used by all participating servers and as a result all servers skip the reevaluation step of 2PVC.

8. REFERENCES

[1] A.J. Lee and M. Winslett, "Safety and Consistency in Policy-Based Authorization Systems," Proc. 13th ACM Conf. Computer and Comm. Security (CCS '06), 2006.

[2] D.J. Abadi, "Data Management in the Cloud: Limitations and Opportunities," IEEE Data Eng. Bull., vol. 32, no. 1, pp. 3-12, Mar. 2009.

[3] D.Cooper et al., "Internet x.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," RFC 5280, http://tools.ietf.org/html/rfc5280, May 2008.

[4] E. Rissanen, "Extensible Access Control Markup Language (Xacml) Version 3.0," http://docs.oasis-open.org/xacml/3.0/ xacml-3.0-core-spec-os-en.html, Jan. 2013.

[5] F. Chang et al., "Bigtable: A Distributed Storage System for Structured Data," Proc. Seventh USENIX System.

[6] G. DeCandia et al., "Dynamo: Amazons Highly Available Key- Value Store," Proc. 21st ACM SIGOPS Symp. Operating Systems Principles (SOSP '07), 2007.

[7] J. Li, N. Li, and W.H. Winsborough, "Automated Trust Negotiation Using Cryptographic Credentials," Proc. 12th ACM Conf. Computer and Comm. Security (CCS '05), Nov. 2005.

[8] J. Li and N. Li, "OACerts: Oblivious Attribute Based Certificates," IEEE Trans. Dependable and Secure Computing, vol. 3, no. 4, pp. 340- 352, Oct.-Dec. 2006.

[9] J. Camenisch and A. Lysyanskaya, "An Efficient System for Non- Transferable Anonymous Credentials with Optional Anonymity Revocation," Proc. Int'l Conf.

Theory and Application of Cryptographic Techniques: Advances in Cryptology (EUROCRYPT '01), 2001

[10] L. Bauer et al., "Distributed Proving in Access-Control Systems," Proc. IEEE Symp. Security and Privacy, May 2005.

[11] M. Armbrust et al., "Above the Clouds: A Berkeley View of Cloud Computing," technical report, Univ. of California, Feb. 2009.

[12] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - Ocsp," RFC 2560, http://tools.ietf.org/html/rfc5280, June 1999.

[13] M.K. Iskander, D.W. Wilkinson, A.J. Lee, and P.K. Chrysanthis, "Enforcing Policy and Data Consistency of Cloud Transactions," Proc. IEEE Second Int'l Workshop Security and Privacy in Cloud Computing (ICDCS-SPCCICDCS-SPCC), 2011.

[14] P.K. Chrysanthis, G. Samaras, and Y.J. Al-Houmaily, "Recovery and Performance of Atomic Commit Processing in Distributed Database Systems," Recovery Mechanisms in Database Systems, Prentice Hall PTR, 1998

[15] S. Das, D. Agrawal, and A.E. Abbadi, "Elastras: An Elastic Transactional Data Store in the Cloud," Proc. Conf. Hot Topics in Cloud Computing (USENIX HotCloud '09), 2009.