DYNAMIC RESOURCE ASSIGNMENT AND MIGRATION FOR EFFECTIVE CLOUD UTILIZATION

A PROJECT REPORT

Submitted by

JAYASHRI.M.G [REGISTER NO: 211417104091]

KAVITHA.E [REGISTER NO: 211417104114]

AMRUTHA.R [REGISTER NO: 211417104014]

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE, CHENNAI-600123.

ANNA UNIVERSITY: CHENNAI 600 025

AUGUST 2021

BONAFIDE CERTIFICATE

Certified that this project report **“ DYNAMIC RESOURCE ASSIGNMENT AND MIGRATION FOR EFFECTIVE CLOUD UTILIZATION”** is the bonafide work of “ JAYASHRI.M.G[REGISTER NO:211417104091],KAVITHA.E[REGISTER NO:211417104114],AMRUTHA.R[REGISTER NO:211417104014]**”** who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr.S. MURUGAVALLI, M.E., Ph.D.,	**Dr. KAVITHA SUBRAMANI, M.E., Ph.D.,**
HEAD OF THE DEPARTMENT	**SUPERVISOR**
DEPARTMENT OF CSE,	DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,	PANIMALAR ENGINEERING COLLEGE,
NASARATHPETTAI,	NASARATHPETTAI,
POONAMALLEE,	POONAMALLEE,
CHENNAI-600 123	CHENNAI-600 123
.	

Certified that the above candidate(s) was/ were examined in the Anna University Project Viva-Voce Examination held on **05-08-2021**

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We would like to extend our heartfelt and sincere thanks to our Directors **Tmt.C.VIJAYARAJESWARI**, **Dr.C.SAKTHIKUMAR, M.E., Ph.D.,** and **Tmt. SARANYASREE, B.E., M.B.A.,** for providing us with the necessary facilities for completion of this project.

We also express our gratitude to our Principal **Dr.K.Mani, M.E., Ph.D.** for his timely concern and encouragement provided to us throughout the course.

We thank the HOD of CSE Department, **Dr. S.MURUGAVALLI, M.E., Ph.D.,** for the support extended throughout the project.

We would like to thank our guide Dr. **KAVITHA SUBRAMANI, M.E., Ph.D.,** and all the faculty members of the Department of CSE for their advice and suggestions for the successful completion of the project.

NAME OF THE STUDENTS
JAYASHRI.M.G
KAVITHA.E
AMRUTHA.R

TABLE OF CONTENTS

ABSTRACT

In the Existing system, the infrastructure resources in distributed green cloud data centers (DGCDCs) are shared by multiple heterogeneous applications to provide flexible services to Global users in a high-performance and low-cost way. In the Proposed system, task Scheduling and resource optimization (STSRO) method to minimize the total cost of their Provider by cost-effectively scheduling all arriving tasks of heterogeneous applications to meet tasks' delay-bound constraints. In the Modification Process, they are three types of systems. 1. Hot Machines that can handle the current job. 2. Warm Machines are kept in the idle state until job is assigned. 3. Cold machines are also kept idle until warm machines are busy with their jobs. Each machine deploys with three Virtual servers. 1^{st} Job is assigned to the Hot machine's 1^{st} Virtual machine and in the same way other jobs are assigned to the remaining VMs. If the VMs in the hot machine are filled then the jobs are assigned in the Warm machines VMs. In the same way jobs are assigned to the cold machines. Automatic migration of job is implemented, so as to transfer the load to the Hot VM from Warm VM once it has completed the job and cold VM to warm VM. We also implemented cache mechanism.

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

JDK	Java Development Kit
VM	Virtual Machine
HTTP	Hyper Text Transfer Protocol
SaaS	Software as a Service
IaaS	Infrastructure as a Service
PaaS	Platform as a Service
HaaS	Hardware as a Service
DGCDC	Distributed Green Cloud Data Centre

CHAPTER 1

1. INTRODUCTION:

1.1 Overview

Cloud computing is the on-demand availability of computer system resources, especially data storage and computing power, without direct active management by the user. There are three types of service models. 1. SaaS 2. IaaS 3. PaaS. Software as a service (SaaS) is a software distribution model in which a cloud provider hosts applications and makes them available to end users over the internet. In this model, an independent software vendor (ISV) may contract a third-party cloud provider to host the application. Or, with larger companies, such as Microsoft, the cloud provider might also be the software vendor. Iaas is also known as Hardware as a Service (HaaS)**.** It is one of the layers of the cloud computing platform. It allows customers to outsource their IT infrastructures such as servers, networking, processing, storage, virtual machines, and other resources. Customers access these resources on the Internet using a pay-as-per use model. Platform as a Service (PaaS) provides a runtime environment. It allows programmers to easily create, test, run, and deploy web applications. You can purchase these applications from a cloud service provider on a pay-as-per use basis and access them using the Internet connection. In PaaS, back end scalability is managed by the cloud service provider, so end- users do not need to worry about managing the infrastructure. PaaS includes infrastructure (servers, storage, and networking) and platform (middleware, development tools, database management systems, business intelligence, and more) to support the web application life cycle.Now we are using IaaS (Infrastructure as a service). Infrastructure as a service are online services that provides high-level APIs used to dereference various low-level details of underlying network infrastructure like physical computing resources, location, data partitioning, scaling, security, backup etc. Green networking is the practice of selecting energy-efficient networking technologies and products, and minimizing resource use whenever possible. Green computing is an effective approach towards designing, manufacturing, using and disposing of computers and its resources with minimal or no impact on environment. The goals of Green computing is to manage the power and energy efficiency, choice of ecofriendly hardware and software, and recycling the material to increase the product's life. Green computing benefits the environment. Reduced energy usage from green techniques translates into lower carbon dioxide emissions, stemming from a reduction in the fossil fuel used in power plants and transportation. Simply, saving energy and resources saves money. A virtual machine (VM) is a virtual environment that works like a computer within a computer. It runs on an isolated partition of its host computer with its own resources of CPU power, memory, an operating system (e.g. Windows, Linux, macOS), and other resources. Hackers are incorporating virtual machine detection into their Trojans, worms and other malware in order to thwart antivirus vendors and virus researchers, according to a note published this week by the SANS Institute Internet Storm Center. Researchers often use virtual machines to detect hacker activities. Dropbox is a modern workspace designed to reduce busywork-so you can focus on the things that matter. Sign in and put your creative energy to work. Dropbox is used to upload file in the cloud. Initially, we need to create a login ID to upload files in the cloud using dropbox. Dropbox is a file hosting service operated by the American company Dropbox, Inc., that offers cloud storage, file synchronization, personal cloud, and client software.

1.2 Problem definition

The main concept of our project is to reduce the space usage where we store data in cloud that needs more money. By minimizing the space, we can reduce the cost and we have also have another benefit that we can implement green computing method in our project. That is while processing to store data we run all systems that are connected to cloud which consumes more power. But using our project we can control the running systems so that power consumption will be reduced. In our project, migration process is also implemented. Migration process is the process in which the job from one virtual machine is automatically moved to another virtual machine which completed its assigned job. Using this migration process while moving the job of one’s virtual machine to another, the other virtual machines that are connected to cloud and in the power on state, the virtual machines power supply will be turned off until the work is assigned to the particular. So that limited power is consumed and wastage of power is avoided. In the previous projects, migration process is not implemented which resulted in high space usage and power consumption. Due to high power consumption, heat is produced to the surrounding which resulted in global warming. With the help of our project, we can reduce power so that global warming is avoided. As we said earlier about migration process, once the job in the 1st VM is completed, the job from the next VM that is doing its job will automatically migrated and the 2nd VM will be in power off position till another job is assigned. All these processes are implemented using dropbox to the cloud server. In the dropbox, we can upload the file and at the same time we can also delete the uploaded files that are used or unnecessary for future. At the beginning we are using NetBeans and SQL servers to register, login and upload the jobs to the virtual machines in the SQL to the cloud. Overall, we are implanting the green computing method using our project. At present, our project is not implemented worldwide but it is under process.

CHAPTER 2

2. LITERATURE SURVEY:

[1] A major challenging problem for cloud providers is designing efficient mechanisms for virtual machine (VM) provisioning and allocation. Recently, cloud providers have introduced auction-based models for VM provisioning and allocation which allow users to submit bids for their requested VMs. We then design truthful greedy and optimal mechanisms for the problem such that the cloud provider provisions VMs based on the requests of the winning users and determines their payments. We show that the proposed mechanisms are truthful, that is, the users do not have incentives to manipulate the system by lying about their requested bundles of VM instances and their valuations.

[2] Cloud providers provision their heterogeneous resources such as CPUs, memory, and storage in the form of virtual machine (VM) instances which are then allocated to the users. We address the problem of autonomic VM provisioning and allocation for the auction-based model considering multiple types of resources by designing an approximation mechanism. This problem is computationally intractable, and our proposed mechanism is by far the strongest approximation result that can be achieved for this problem. Furthermore, our proposed mechanism drives the system into an equilibrium in which the users do not have incentives to manipulate the system by untruthfully reporting their VM bundle requests and valuations.

[3] This work, to the authors' knowledge, represents the first online combinatorial auction designed for the cloud computing paradigm, which is general and expressive enough to both: 1) optimize system efficiency across the temporal domain instead of at an isolated time point; The final result is an online auction framework that is truthful, computationally efficient, and guarantees a competitive ratio ≈ 3.30 in social welfare in typical scenarios. The framework consists of three main steps: 1) a tailored primal-dual algorithm that decomposes the long-term optimization into a series of independent one-shot optimization problems, with a small additive loss in competitive ratio; 2) a randomized sub framework that applies primal-dual optimization for translating a centralized cooperative social welfare approximation algorithm into an auction mechanism, retaining the competitive ratio while adding truthfulness.

[4] We study the multi-resource allocation problem in cloud computing systems where the resource pool is constructed from a large number of heterogeneous servers, representing different points in the configuration space of resources such as processing, memory, and storage. We design a multi-resource allocation mechanism, called DRFH, that generalizes the notion of Dominant Resource Fairness (DRF) from a single server to multiple heterogeneous servers. Large-scale simulations driven by Google cluster traces show that DRFH significantly outperforms the traditional slot-based scheduler, leading to much higher resource utilization with substantially shorter job completion times.

[5] Clouds have become an attractive computing platform which offers on-demand computing power and storage capacity. However, challenges arise when considering computing instance non-deterministic acquisition time, multiple VM instance types, unique cloud billing models and user budget constraints. In this paper, we present a cloud auto-scaling mechanism to automatically scale computing instances based on workload information and performance desire. It enables cloud applications to finish submitted jobs within the deadline by controlling underlying instance numbers and reduces user cost by choosing appropriate instance types. Results show that our cloud auto-scaling mechanism can meet user specified performance goal with less cost.

[6] Today Cloud computing is on demand as it offers dynamic flexible resource allocation, for reliable and guaranteed services in pay-as-you-use manner, to Cloud service users. This resource provision is done by considering the Service Level Agreements (SLA) and with the help of parallel processing. Hence by considering multiple SLA parameter and resource allocation by preemption mechanism for high priority task execution can improve the resource utilization in Cloud. In this paper we propose an algorithm which considered Preemptable task execution and multiple SLA parameters such as memory, network bandwidth, and required CPU time.

[7] Current cloud providers use fixed-price based mechanisms to allocate Virtual Machine (VM) instances to their users. The fixed-price based mechanisms do not provide an efficient allocation of resources and do not maximize the revenue of the cloud providers. In this PhD dissertation we will design, study and implement combinatorial auction-based mechanisms for efficient provisioning and allocation of VM instances in cloud computing environments.

[8] Although the resource elasticity offered by Infrastructure-as-a-Service (IaaS) clouds opens up opportunities for elastic application performance, it also poses challenges to application management. Instead of burdening cloud users with complex management, we move the task of determining the optimal resource configuration for cluster applications to cloud providers. We find that a structural reorganization of multi-tier websites, by adding a caching tier which runs on resources debited from the original resource budget, significantly boosts application performance and reduces resource usage. Experiment results show that V-Cache outperforms a representative capacity management scheme and a cloud-cache based resource provisioning approach by at least 15% in performance, and achieves at least 11% and 21% savings on CPU and memory resources, respectively.

[9] Although MapReduce, the core technology of cloud computing, lowers the barriers to enter the parallel computing, it introduces the other challenging research issue of improving its performance via properly resource provisioning. In this paper, this optimization problem, called Node Capability-aware Provisioning Problem (NCPP), is first formulated as a mathematical model. Moreover, the node Capability-Aware Resource Provisioner (CARP) is proposed based on Apache Hadoop to show its feasibility to solve NCPP in a systematic way.

[10] Cloud computing is the promising key technology to build future architecture of massive IT systems and one of key benefits of cloud computing is to provide its customers with elastic resources according to the fluctuation of request workloads. Adaptive resource management architecture has been proposed, and we divide resource management into two parts, resource provision and job scheduling. Simulation evaluation has been set up with realistic grid workload, and results show that our provisioning model gives elastic resource provisioning for dynamic workload and FCFS achieves better performance compared with other scheduling policies.

CHAPTER 3

3. SYSTEM ANALYSIS:

Systems design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could see it as the application of systems theory to product development

3.1 EXISTING SYSTEM:

In the **EXISTING SYSTEM,** the infrastructure resources in distributed green cloud data centers (DGCDCs) are shared by multiple heterogeneous applications to provide flexible services to global users in a high-performance and low-cost way.

DISADVANTAGES:

- Congestion occurring
- More power consumption
- Waiting time is increased
- Unreliable
- Low data transmission rate
- replicate request

3.2 PROPOSED SYSTEM:

We deploy two types of systems. 1. Hot Machines can handle the current job. 2. Warm Machines are kept idle state until job is assigned. We deploy three Virtual servers for every machine. 1st Job is assigned to the hot machine 1st Virtual machine and same way following jobs are assigned to other VMs. Now jobs are assigned to the Warm machines once all the VMs of Hot category have occupied with the jobs. Automatic migration of job is implemented, so as to transfer the load to the Hot VM from Warm VM once it has completed the job. We also implemented cache mechanism.

ADVANTAGES:

- Avoid congestion
- Less power consumption
- Waiting time is decreased
- Reliable
- High data transmission rate
- Avoid replicate request

3.3 REQUIREMENT ANALYSIS AND SPECIFICATION:

3.3.1 INPUT REQUIREMENTS:

1. Create or register new account
2. Give user ID and password
3. Login and assign jobs to VM's
4. Upload files in Dropbox.

3.3.2 OUTPUT REQUIREMENTS:

1. Processing the job.
2. Uploading the files in cloud.

3.3.3 FUNCTIONAL REQUIREMENTS:

In software engineering and systems engineering, a functional requirement defines a function of a system or its component, where a function is described as a specification of behavior between inputs and outputs. As defined in requirements engineering, functional requirements specify particular results of a system.

A Functional Requirement (FR) is a description of the service that the software must offer. It describes a software system or its component. A function is nothing but inputs to the software system, its behavior, and outputs. It can be a calculation, data manipulation, business process, user interaction, or any other specific functionality which defines what function a system is likely to perform. Functional Requirements are also called Functional Specification.

1. **Registration functions:**

   The user will be registered based upon his username and password provided.
2. **Authorization levels:**

   The user will be authorized based upon his username and password provided.
3. **Selecting location**:

   Once logged in, we need to set the location where the file needs to be processed.
4. **Uploading file:**

   User need to upload the files and the virtual machine starts processing the jobs assigned.
5. **Algorithms:**

   Resource Migration Algorithm.
6. **Backup and recovery:**

   In case the file uploaded by the user has been corrupted, then the automatic recovery of the corrupted files will be initiated.

3.4 HARDWARE ENVIRONMENT:

• Processor	:	Core i3/i5/i7
• RAM	:	2-4GB
• HDD	:	500 GB

3.5 Software Requirements

• Operating system	:	Windows 7/8
• Front End	:	Java-JDK1.7
• Back End	:	MYSQL
• Cloud	:	Drop Box
• SQL Query Browser		
• NetBeans		

3.5.1 JAVA

3.5.1.1 INTRODUCTION TO JAVA

Java is an object-oriented, cross platform, multi-purpose programming language produced by Sun Microsystems. First released in 1995, it was developed to be a machine independent web technology. It was based on C and C++ syntax to make it easy for programmers from those communities to learn. Since then, it has earned a prominent place in the world of computer programming.

Java has many characteristics that have contributed to its popularity:

- **Platform independence** - Many languages are compatible with only one platform. Java was specifically designed so that it would run on any computer, regardless if it was running Windows, Linux, Mac, UNIX or any of the other operating systems.
- **Simple and easy to use** - Java's creators tried to design it so code could be written efficiently and easily.
- **Multi-functional** - Java can produce many applications from command-line programs to applets to Swing windows (basically, sophisticated graphical user interfaces).

Java does have some drawbacks. Since it has automated garbage collection, it can tend to use more memory than other similar languages. There are often implementation differences on different platforms, which have led to Java being described as a "write once, test everywhere" system. Lastly, since it uses an abstract "virtual machine", a generic Java program doesn't have access to the Native API's on a system directly. None of these issues are fatal, but it can mean that Java isn't an appropriate choice for a particular piece of software.

3.5.1.2 JAVA PLATFORM

One thing that distinguished Java from some other languages is its ability to run the same compiled code across multiple operating systems.

In other languages, the source code (code that is written by the programmer), is compiled by a compiler into an executable file. This file is in machine language, and is intended for a single operating system/processor combination, so the programmer would have to re-compile the program separately for each new operating system/processor combination.

Java is different in that it does not compile the code directly into machine language code. Compilation creates bytecode out of the source code. Bytecode generally looks something like this:

```
a7 f4 73 5a 1b 92 7d
```

When the code is run by the user, it is processed by something called the Java Virtual Machine (JVM). The JVM is essentially an interpreter for the bytecode. It goes through the bytecode and runs it. There are different versions of the JVM that are compatible with each OS and can run the same code. There is virtually no difference for the end-user, but this makes it a lot easier for programmers doing software development.

3.5.1.3 INSTALLING JDK

Before installing the Java Development Kit (JDK), you should probably know what it is. It is distributed by Oracle. It contains the core libraries and compiler required to develop Java. The JDK should not be confused with the JRE (Java Runtime Environment). The JRE is a JVM for running, as opposed to compiling, Java programs.

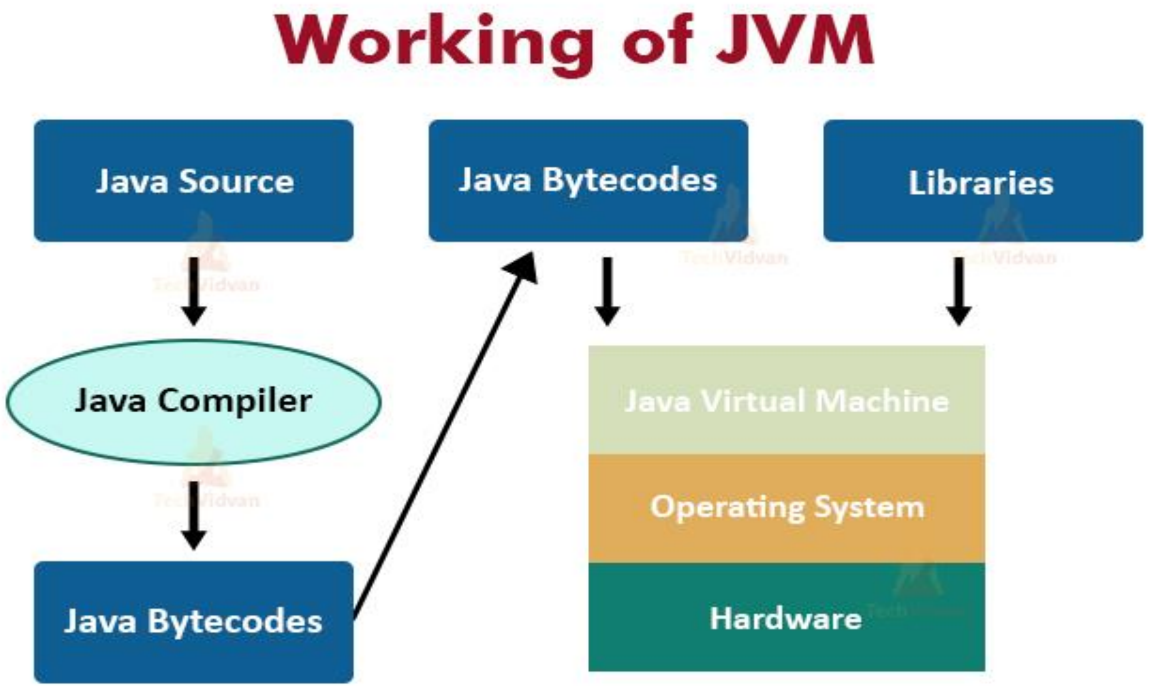


Figure 1.1 Working of JVM

3.5.1.4 Downloading and Installing

To download the JDK, go to http://www.oracle.com/technetwork/java/javase/downloads/index.html.

Click on "JDK with NetBeans Bundle". Follow the instructions for downloading the JDK installation file.

Windows: If you are running Windows, simply run the executable file and follow the installation instructions.

UNIX, Solaris, or Linux: For Linux and UNIX, download the "jdk1 6.0" for Linux systems. Save the downloaded file in any drive. Once you have saved the file, extract it to a place that you can remember, by using Terminal or by double clicking on the file. When you have finished extracting the file, copy the JDK 1.6.0 folder and paste it in the user/local (To paste to the user/local directory, you have to be in root) so that every user can use the java files. You can delete the downloaded zip file so that it doesn't take up space on your drive.

Macintosh: The latest available JDK is automatically installed by the operating system. Because Java for Macintosh is developed and maintained by Apple, in coordination with Sun, the current version on the Macintosh may not be the current version that is available from Sun.

3.5.1.5 WORKING OF JAVA

For those who are new to object-oriented programming, the concept of a class will be new to you. Simplistically, a class is the definition for a segment of code that can contain both data (called attributes) and functions (called methods). When the interpreter executes a class, it looks for a particular method by the name of main, which will sound familiar to C programmers. The main method 27 is passed as a parameter an array of strings (similar to the argv [] of C), and is declared as a static method. To output text from the program, we execute the println method of System.out, which is java's output stream. UNIX users will appreciate the theory behind such a stream, as it is actually standard output. For those who are instead used to the Wintel platform, it will write the string passed to it to the user's program. Java consists of two things:

- Programming language
- Platform

3.5.1.6 JAVA PROGRAMMING LANGUAGE

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

- Simple
- Object-oriented

- ➢ Distributed
- ➢ Interpreted
- ➢ Secure
- ➢ Architecture-neutral
- ➢ Portable
- ➢ High-performance
- ➢ Multithreaded
- ➢ Dynamic

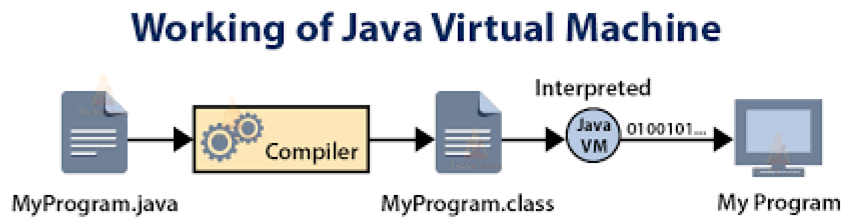


Figure 1.2 JVM

3.5.2 DROPBOX

3.5.2.1 INTRODUCTION TO DROPBOX

Dropbox is a file hosting service operated by the American company Dropbox, Inc., headquartered in San Francisco, California, that offers cloud storage, file synchronization, personal cloud, and client software. Dropbox was founded in 2007 by MIT students Drew Houston and Arash Ferdowsi as a startup company, with initial funding from seed accelerator Y Combinator.

Dropbox brings files together in one central place by creating a special folder on the user's computer. The contents of these folders are synchronized to Dropbox's servers and to other computers and devices where the user has installed Dropbox, keeping the same files up-to-date on all devices. Dropbox uses a freemium business model, where users are offered a free account with a set storage size, with paid subscriptions available that offer more capacity and additional features. Dropbox Basic users are given two gigabytes of free storage space. Dropbox Plus users are given two terabytes of storage space, as well as additional features, including advanced sharing controls, remote wipe, and an optional Extended Version History add-on. Dropbox offers computer apps for Microsoft Windows, Apple macOS, and Linux computers, and mobile apps for iOS, Android, and Windows Phone smartphones and tablets. In March 2013, the company acquired Mailbox, a popular email app, and in April 2014, the company introduced Dropbox Carousel, a photo and video gallery app. Both Mailbox and Carousel were shut down in December 2015, with key features from both apps implemented into the regular Dropbox service. In October 2015, it officially announced Dropbox Paper, its collaborative document editor, in a reported effort to expand its operations towards businesses. In March 2016, Dropbox had 500 users.

3.5.2.2 DROPBOX PLATFORM

Dropbox has computer apps for Microsoft Windows, Apple macOS, and Linux computers, and mobile apps for iOS, Android, and Windows Phone smartphones and tablets. It also offers a website interface. As part of its partnership with Microsoft, Dropbox announced a universal Windows 10 app in January 2016.

Dropbox's apps offer an automatic photo uploading feature, allowing users to automatically upload photos or videos from cameras, tablets, SD cards, or smartphones to a dedicated "Camera Uploads" folder in their Dropbox. Users are given 500 megabytes of extra space for uploading their first photo, and are given up to 3 gigabytes of extra space if users continue using the method for more photos.

In July 2014, Dropbox introduced "streaming sync" for its computer apps. Streaming sync was described as a new "supercharged" synchronization speed for large files that improves the upload or download time by up to 2 times.

In August 2015, Dropbox announced the availability of "Universal 2nd Factor" USB security keys, providing two-factor authentication for logging into its services.

3.5.3 NETBEANS

NetBeans is an integrated development environment (IDE) for Java. NetBeans allows applications to be developed from a set of modular software components called *modules*. NetBeans runs on Windows, macOS, Linux and Solaris. In addition to Java development, it has extensions for other languages like PHP, C, C++, HTML5,] and JavaScript. Applications based on NetBeans, including the NetBeans IDE, can be extended by third party developers.

3.5.3.1 NETBEANS IDE

NetBeans IDE is an open-source integrated development environment. NetBeans IDE supports development of all Java application types (Java SE (including JavaFX), Java ME, web, EJB and mobile applications) out of the box. Among other features are an Ant-based project system, Maven support, refactoring, version control (supporting CVS, Subversion, Git, Mercurial and Clearcase).

Modularity: All the functions of the IDE are provided by modules. Each module provides a well-defined function, such as support for the Java language, editing, or support for the CVS versioning system, and SVN. NetBeans contains all the modules needed for Java development in a single download, allowing the user to start working immediately. Modules also allow NetBeans to be extended. New features, such as support for other programming languages, can be added by installing additional modules. For instance, Sun Studio, Sun Java Studio Enterprise, and Sun Java Studio Creator from Sun Microsystems are all based on the NetBeans IDE.

License: The IDE is licensed under the Apache License 2.0. Previously, from July 2006 through 2007, NetBeans IDE was licensed under Sun's Common Development and Distribution License (CDDL), a license based on the Mozilla Public License (MPL). In October 2007, Sun announced that NetBeans would henceforth be offered under a dual license of the CDDL and the GPL version 2 licenses, with the GPL linking exception for GNU Classpath. Oracle has donated NetBeans Platform and IDE to the Apache Foundation where it underwent incubation and graduated as a top level project in April 2019.

3.5.4 SQL QUERY BROWSER

MySQL is an open-source relational database management system (RDBMS). Its name is a combination of “My”, the name of co-founders Michael Widenius's daughter, and "SQL", the abbreviation for Structured Query Language. A relational database organizes data into one or more data tables in which data types may be related to each other; these relations help structure the data. SQL is a language programmers use to create, modify and extract data from the relational database, as well as control user access to the database. In addition to relational databases and SQL, an RDBMS like MySQL works with an operating system to implement a relational database in a computer's storage system, manages users, allows for network access and facilitates testing database integrity and creation of backups.

3.5.4.1 FEATURES OF DROPBOX

- A broad subset of ANSI SQL 99, as well as extensions
- Cross-platform support
- Stored procedures, using a procedural language that closely adheres to SQL/PSM[79]
- Triggers
- Cursors
- Updatable views
- Online Data Definition Language (DDL) when using the InnoDB Storage Engine.
- Information schema
- Performance Schema that collects and aggregates statistics about server execution and query performance for monitoring purposes.
- A set of SQL Mode options to control runtime behavior, including a strict mode to better adhere to SQL standards.
- X/Open XA distributed transaction processing (DTP) support; two phase commit as part of this, using the default InnoDB storage engine

CHAPTER 4
SYSTEM DESIGN

4.1. SYSTEMFLOW DIAGRAM:

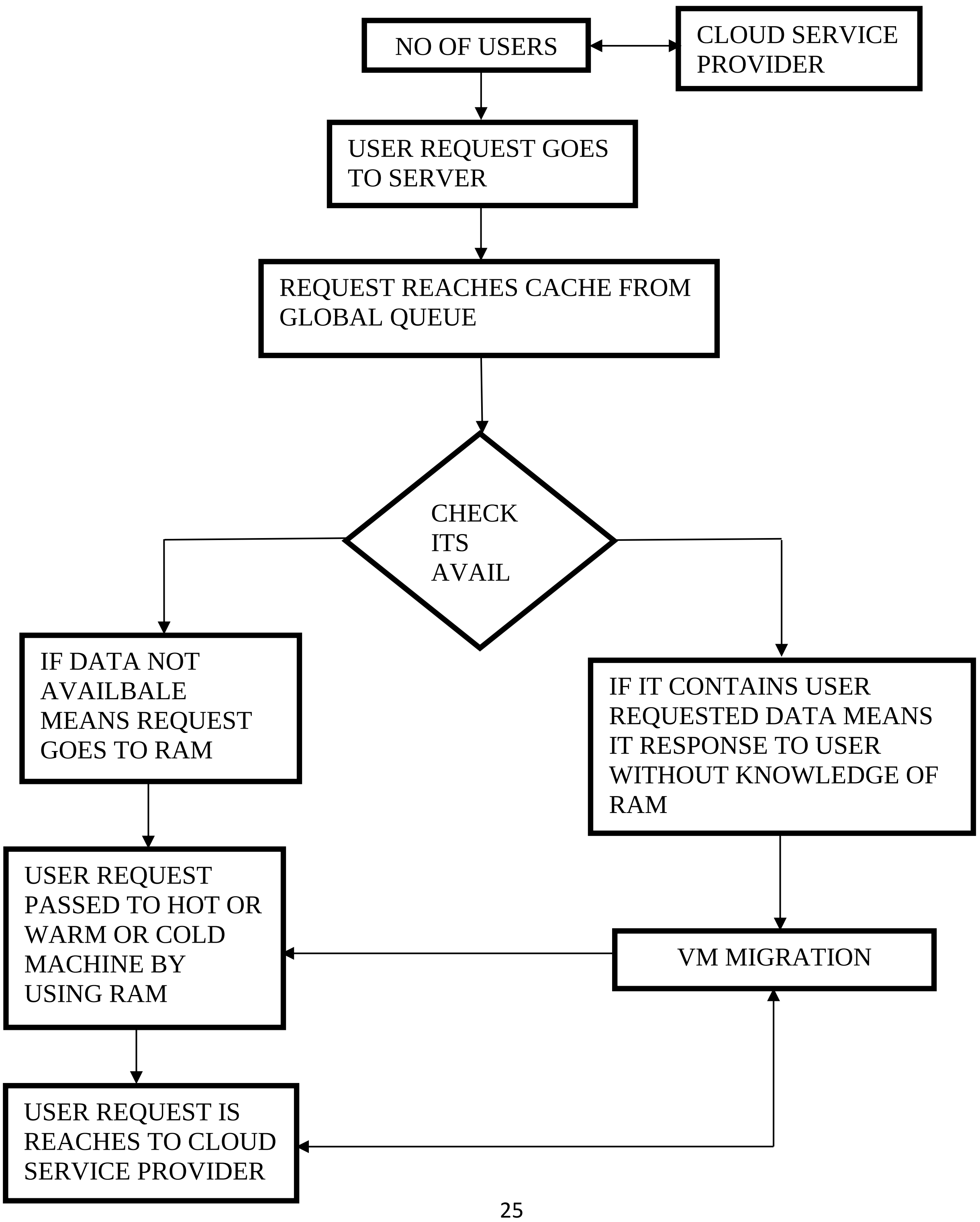

NO OF USERS
CLOUD SERVICE PROVIDER
USER REQUEST GOES TO SERVER
REQUEST REACHES CACHE FROM GLOBAL QUEUE
CHECK ITS AVAIL
IF DATA NOT AVAILBALE MEANS REQUEST GOES TO RAM
IF IT CONTAINS USER REQUESTED DATA MEANS IT RESPONSE TO USER WITHOUT KNOWLEDGE OF RAM
USER REQUEST PASSED TO HOT OR WARM OR COLD MACHINE BY USING RAM
VM MIGRATION
USER REQUEST IS REACHES TO CLOUD SERVICE PROVIDER

Figure 1.3 System Flow Diagram

4.5 UML DIAGRAM:

4.5.1 USECASE DIGRAM

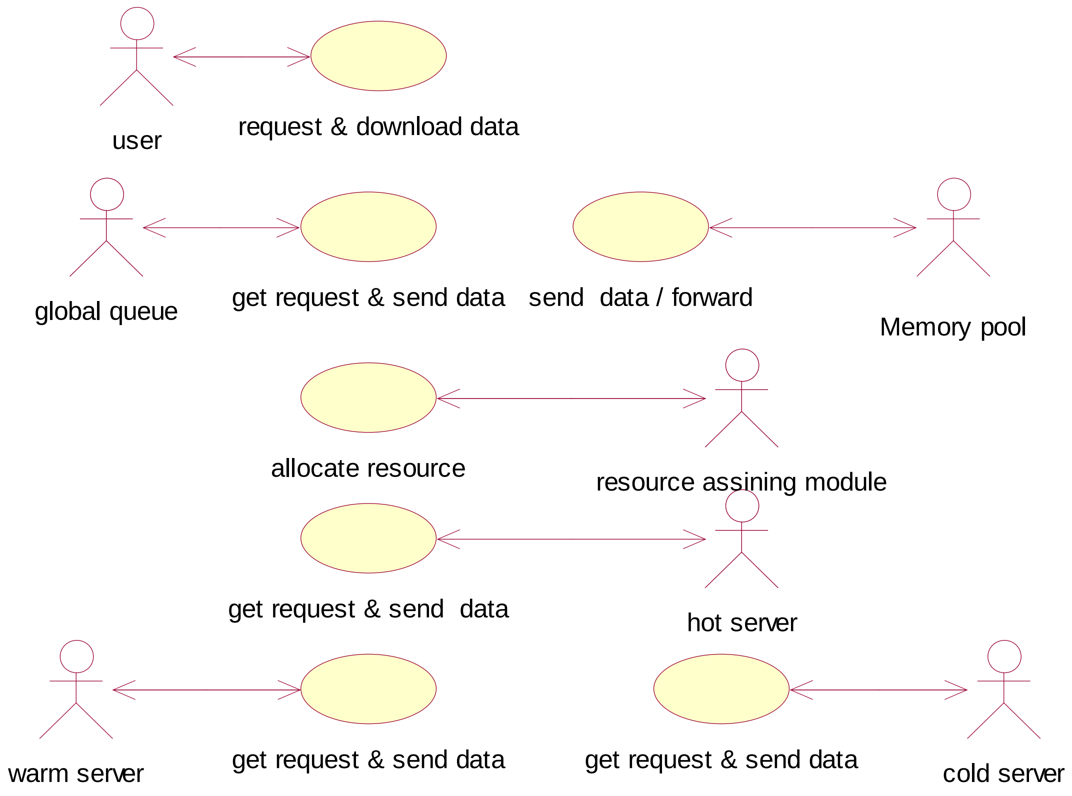


Figure 1.4 Use case Diagram

4.5.2 SEQUENCE DIAGRAM:

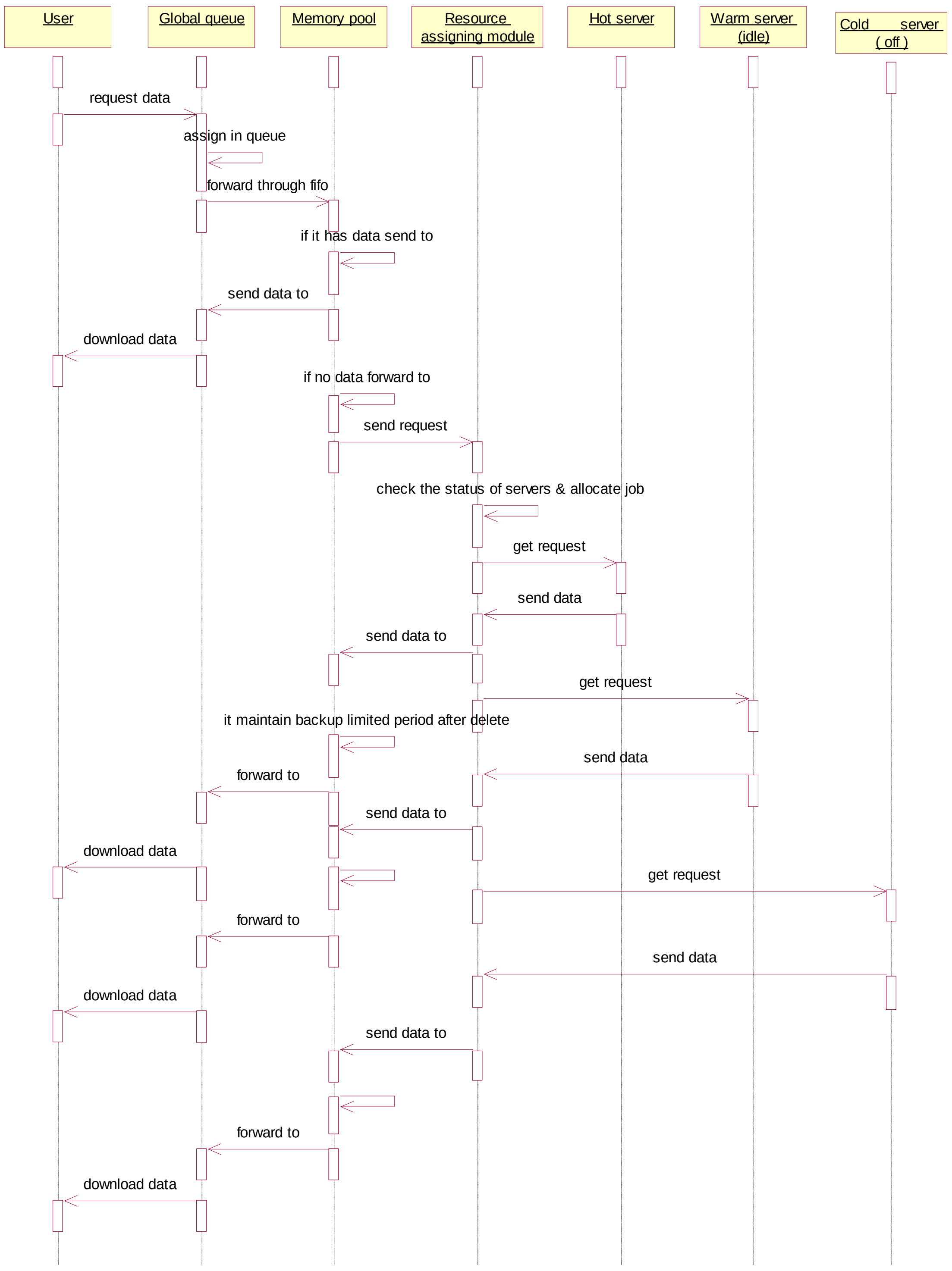


Figure 1.5 Sequence Diagram

4.5.3 COLLABORATION DIAGRAM:

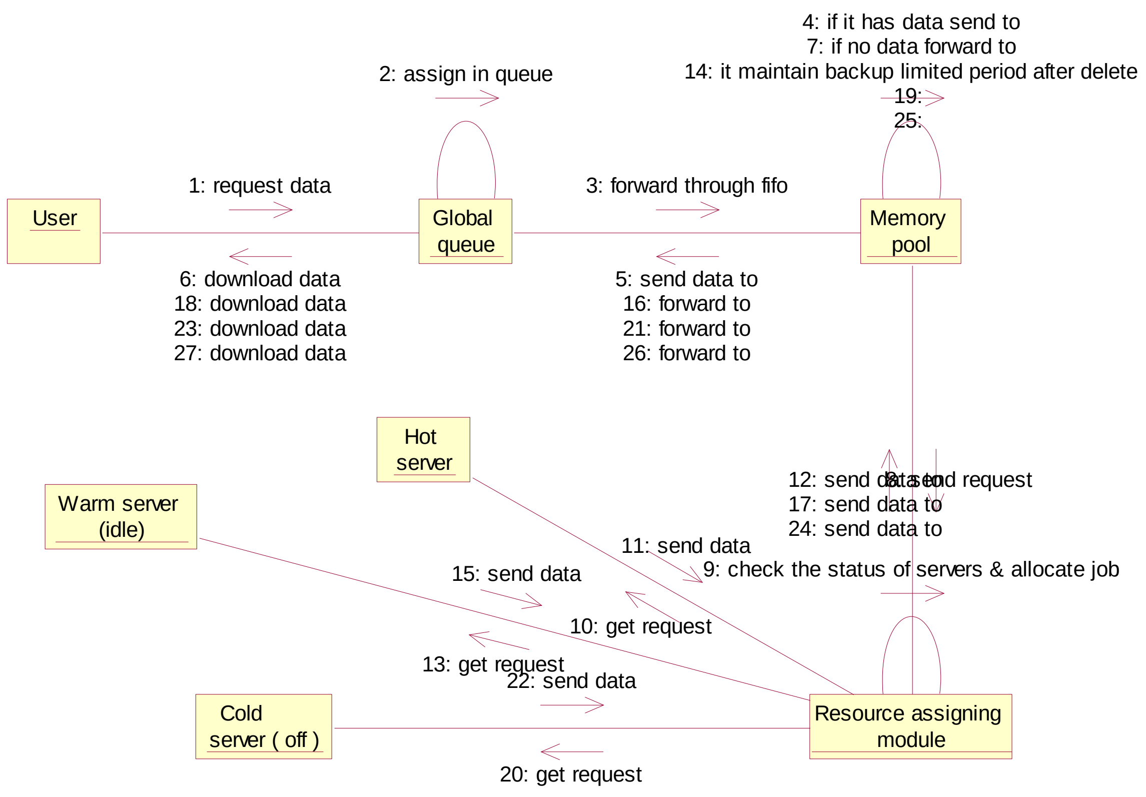


Figure 1.6 Collaboration Diagram

4.5.4 ACTIVITY DIGRAM:

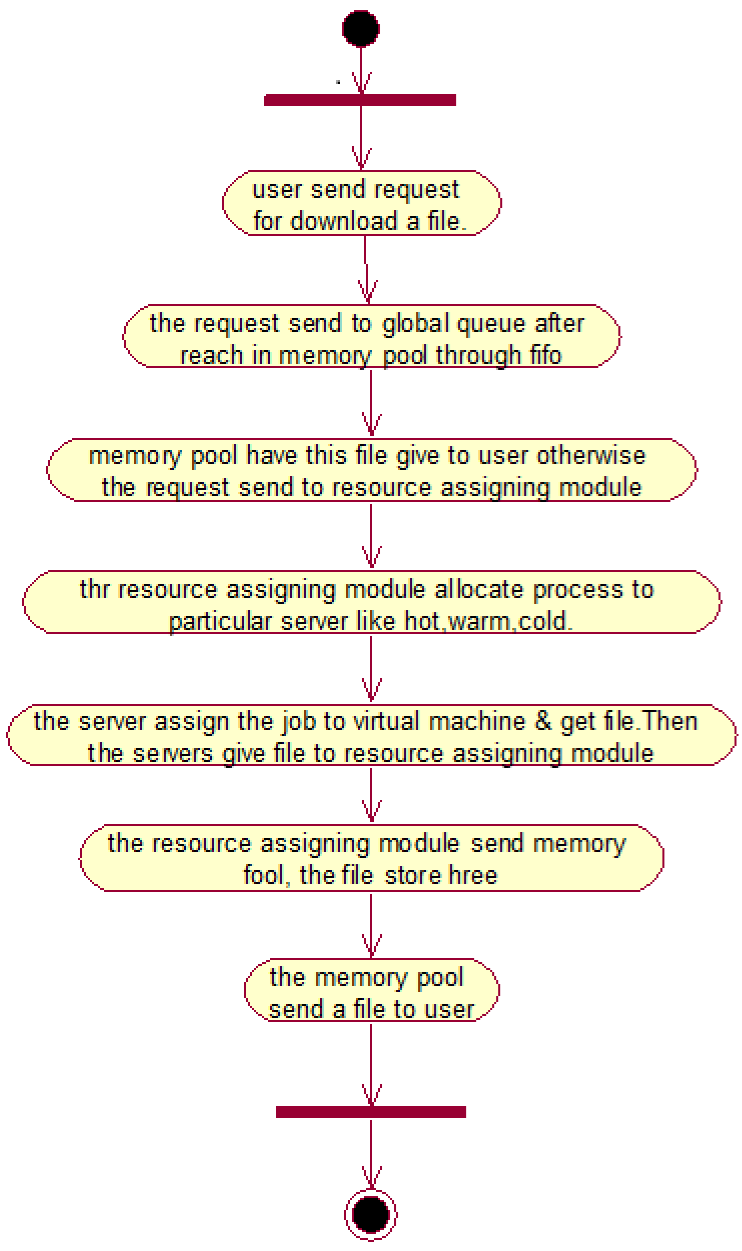

user send request for download a file.
the request send to global queue after reach in memory pool through fifo
memory pool have this file give to user otherwise the request send to resource assigning module
thr resource assigning module allocate process to particular server like hot,warm,cold.
the server assign the job to virtual machine & get file.Then the servers give file to resource assigning module
the resource assigning module send memory fool, the file store hree
the memory pool send a file to user

Figure 1.7 Activity Diagram

4.5.5 CLASS DIAGRAM:

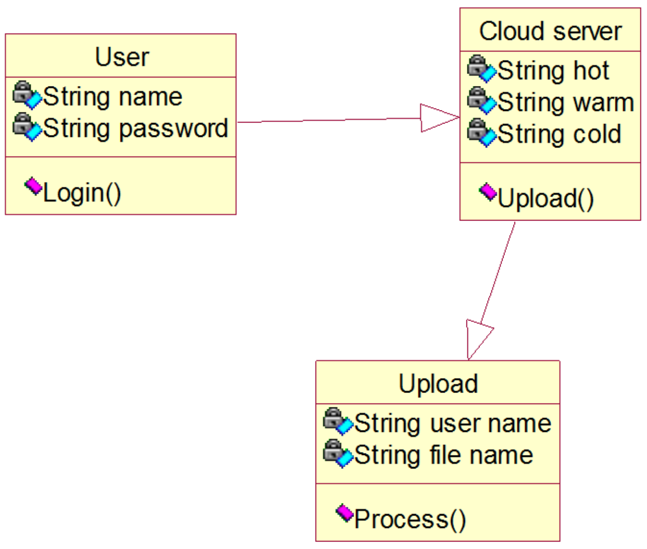


Figure 1.8 Class Diagram

CHAPTER 5

5. SYSTEM ARCHITECTURE:

5.1 ARCHITECTURE OVERVIEW:

From the below architecture diagram, we can understand that initially virtual machine divides the job and assigns it according to the number of job divided. Hot machine are the one that start to execute the job, if some more jobs are need to be executed then it is allocated to warm and cold machine. If job in hot machine is completed then the job from warm machine will automatically migrate to the hot machine.

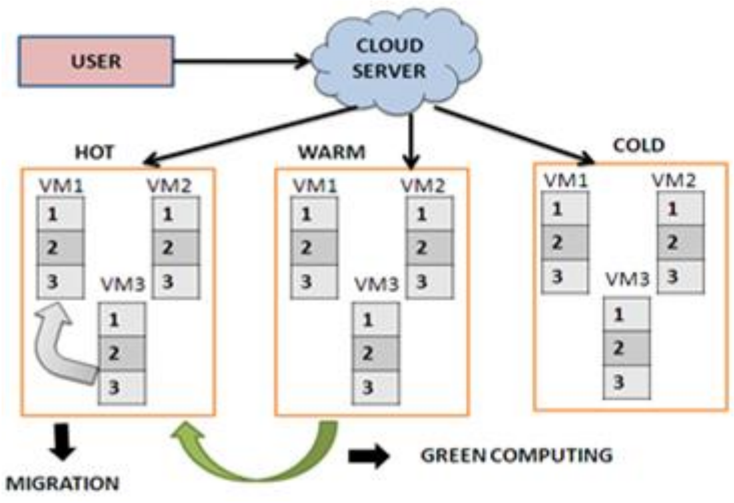


Figure 1.9 System Architecture diagram

5.2 MODULE DESIGN SPECIFICATION:

1. User Registration
2. Cloud Server Deployment
3. Intermediate Server Deployment
4. Green Computing Setup
5. Migration of Virtual Server
6. Cache Server Implementation

5.1.1 USER REGISTRATION

In this module we are going to create a User application by which the User is allowed to access the data from the Server of the Cloud Service Provider. Here first the User wants to create an account and then only they are allowed to access the Network. Once the User creates an account, they are to login into their account and request the Job from the Cloud Service Provider. Based on the User's request, the Cloud Service Provider will process the User requested Job and respond to them. All the User details will be stored in the Database of the Cloud Service Provider. In this Project, we will design the User Interface Frame to Communicate with the Cloud Server through Network Coding using the programming Languages like Java/ .Net. By sending the request to Cloud Server Provider, the User can access the requested data if they authenticated by the Cloud Service Provider.

5.1.2 CLOUD SERVER DEPLOYMENT

Cloud Service Provider will contain the large amount of data in their Data Storage. Also the Cloud Service provider will maintain the all the User information to authenticate the User when are login into their account. The User information will be stored in the Database of the Cloud Service Provider. Also the Cloud Server will redirect the User requested job to the Resource Assigning Module to process the User requested Job. The Request of all the Users will process by the Resource Assigning Module. To communicate with the Client and the with the other modules of the Cloud Network, the Cloud Server will establish connection between them. For this Purpose we are going to create a User Interface Frame. Also the Cloud Service Provider will send the User Job request to the Resource Assign Module in Fist in First out (FIFO) manner.

5.1.3 INTERMEDIATE SERVER DEPLOYMEN

By implementing Intermediate Server the Job Processing Scheme, we can effectively process the User Requested Job and efficiently maintains the Resources of the Cloud Server. So that we can save the Energy of the Resources when they are not process the Job.

5.1.4 GREEN COMPUTING SETUP

Green computing is the term used to denote efficient use of resources in computing. It is also known as Green IT. In this Module, we will Process the User requested Job. The User requested Job will redirect to the RAM of the Cloud Server. The RAM will contain three Types of the Physical Servers. 1. HOT Server, WARM Server and COLD Server. These

Physical Servers will contain ‘n’ number of virtual Server to process the User requested Job. So that the Job can be efficiently processed.

5.1.5 MIGRATION OF VIRTUAL SERVER

In this module we create the migration server, main use of migration to migrate the job form on virtual serve to another server, so that the energy can be reduce and work load of the server is balanced, by using the Migration we can shift the process from one VM to anther VM without loss of data.

5.1.6 CACHE SERVER IMPLEMENTATION

As a modification in this Project, we are creating a Cache Memory in the User requested job will be stored for the period time. If another User requests the same Job to the Server of the Cloud Service Provider (CSP), the Server will check in the Cache Memory first. So that we can reduce the Job Processing Time. If the request Data is presented, then the Server will provide the Data to the User immediately. If the request Data is not in the Cache Memory, then the Server process the User requested Job by transferring it to the RAM.

5.2 PROGRAM DESIGN LANGUAGE

RESOURCE MIGRATION ALGORITHM

Using this Algorithm, Once the job is allotted to a particular Server, assuming that this current job proposing server has completed half of the work, by the time the another already running server has completed the running job, then this running job will be dynamically moved to that current running server so that the job is migrated dynamically.

CHAPTER 6

6. SYSTEM IMPLEMENTATION

6.1 CLIENT SIDE CODING:

```
import java.sql.*;
import java.io.*;
import java.net.*;
import java.util.*;
public class dataserver
{

        private dbase db;
        ResultSet ls;
        Connection cs;
        Statement st;
        Vector v,v1,allfile;
        static ServerSocket socte;
        static Socket soc;
        ObjectInputStream dis;
        ObjectOutputStream dos;
        InputStream is;
        OutputStream os;
  String n="",ip="",s,name,nodes;
  static dataserver sol;
  int polte=4003,na;
static String ipa="";
static int poct =0;
        public dataserver()throws Exception
        {
                super();
                db = new dbase();
                st=db.dbcon();
                 s="delete  from userdetails";
                db.delete(s);
        }
        public void listen()throws Exception
         {

   dis=new ObjectInputStream(soc.getInputStream());
         String lname=(String)dis.readObject();



          if(lname.equals("user"))
                {
                        String output="";
dis=new ObjectInputStream(soc.getInputStream());
         String nname2=(String)dis.readObject();
         String ay[]=nname2.split("&");
         int l=Integer.parseInt(ay[1]);

         s="select * from userdetails where username='"+ay[0]+"' and ip='"+ay[2]+"'";
         if(db.check(s))
                        {
      output="exist";
                s="delete from userdetails where username=''";

 db.delete(s);


                        }
                        else
                        {
         s="update userdetails set  username='"+ay[0]+"' where portno='"+l+"'";
         db.insert(s);
         s="update userdetails set  password='"+ay[3]+"' where portno='"+l+"'";
         db.insert(s);
```

```
                    }

dos=new ObjectOutputStream(soc.getOutputStream());
 dos.writeObject(output);

                    }
                    else if(lname.equals("validuser"))
          {
                              String oup1e="",status1="";
dis=new ObjectInputStream(soc.getInputStream());
           String nname12=(String)dis.readObject();
           String ay1[]=nname12.split("&");
           int l1=Integer.parseInt(ay1[1]);

          s="select status from userdetails where username='"+ay1[0]+"'";
           ls=st.executeQuery(s);
           if(ls.next())
                    {
                     status1=ls.getString(1);
                    }
                    if(status1.equals("disable"))
                    {
                    s="select password from userdetails where  ip='"+ay1[2]+"' and
username='"+ay1[0]+"'";

           String pwo=db.select1(s);
                    s="select username from userdetails where  ip='"+ay1[2]+"' and
username='"+ay1[0]+"'";
           String usm=db.select1(s);

           if(pwo.equals(ay1[3]))
                    {
```

```
 if(usm.equals(ay1[0]))
                                {
              oup1e="ok";
                s="update userdetails set  portno='"+l1+"' where username='"+ay1[0]+"'";
              db.insert(s);
                s="update userdetails set  status='enable' where username='"+ay1[0]+"'";
              db.insert(s);

                                }
                        else
                                {
                                oup1e="ok1";

}
                                }

                                else
                                {
                                        System.out.println("pwo:"+pwo);

                                        oup1e="ok1";

                                }
                        }
                        else
                        {
                                        oup1e="ok1";

                        }
dos=new ObjectOutputStream(soc.getOutputStream());
 dos.writeObject(oup1e);
```

```
            }

                        else
                        {
String nameay[]=lname.split("&");
dos=new ObjectOutputStream(soc.getOutputStream());
polte=polte+1;
String p=String.valueOf(polte);
 System.out.println("p:"+p);
 s="Insert into userdetails
values('','"+nameay[0]+"','"+polte+"','disable','"+nameay[1]+"','','')";
 db.insert(s);
            dos.writeObject(p);

System.out.println("send");

                        }
            }

           public static void main(String[] alg)
           {

                       try
                       {

                                   socte=new ServerSocket(4000);
                                   sol=new dataserver();

                                   while(true)
                                   {
                                   soc=socte.accept();
                           sol.listen();
                        }
                }
                catch (Exception ex)
                {
                        System.out.println("Failed loading L&F: ");
                        System.out.println(ex);
                }
        }
 }
 import java.awt.*;
 import java.awt.event.*;
 import javax.swing.*;
 import javax.swing.event.*;
import java.io.*;
import java.net.*;
import java.util.*;
import java.sql.*;
import java.lang.*;
import java.util.Vector;
import java.awt.Toolkit;
import java.util.Timer;
import java.util.TimerTask;
import java.util.StringTokenizer;
import java.util.concurrent.*;

public class server extends JFrame implements Runnable
{
    int change;
        Vector system=new Vector();
        int ucl,thtime,losme;
         int sto,ttime=0;
         String mesg,plocess,file,sysname,s,jpot,nameu="",tcnj,dl;
        public ResultSet ns,ls,csl;
         public Connection con,col,co;
         public Statement st,sl,sc;
         DefaultListModel model1;
         private static JLabel jLabel1;
         private static JLabel jLabel2;
         private static JLabel jLabel3;
         private static JLabel jLabel4;
         private static JTextArea jList1;
         private static JScrollPane jscle;
         private static JTextArea jTextAea1;
         private static JScrollPane jscne;

         private JPanel contentPane;
         String logsys,log,sname,all,dosn,smesg,toype="";
         static Thread t1;
         String socue,doste,state,smsg,slcne,msgtime,ht,mcj,ncj,ipj,pcj,clj,flj,tlj;
         FileOutputStream output,op1;
         String fd;
    boolean sta=true,stak;
    int end;
         public static Socket  cs,cs1,cs2;
         ObjectInputStream in,in1,in2;
         ObjectOutputStream ois1,ois2,ois3;
         public static server sn,c,f;
         String los1;
         int  staa=0,che,flage=0;
        // String ht;
    Vector v1,v2;
         int pocne,tyl=0;

         Vector p2=new Vector();
         Vector tcxoe=new Vector();
         Vector alfie=new Vector();
```

```
            public static ServerSocket mst;
                        public server()throws Exception
                        {
                          super();
                        System.out.println(">");
                        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
con = DriverManager.getConnection("jdbc:odbc:server");
st=con.createStatement();
col = DriverManager.getConnection("jdbc:odbc:server");
sl=col.createStatement();
co = DriverManager.getConnection("jdbc:odbc:server");
sc=co.createStatement();
            s="delete from filetable";
             st.executeUpdate(s);
              s="update list set status='0'";
             st.executeUpdate(s);
             System.out.println(">>>");

                        }
public void intz()throws Exception
           {
                          System.out.println("[");
                                   JLabel bg;
                                   bg = new JLabel(new ImageIcon("predictionserver.jpg"));
                                   pocne=6666;
                                   v1=new Vector();
                                   JLabel jLabel7=new JLabel();
                                   jLabel1 = new JLabel();
                                   jLabel2 = new JLabel();
                                   jLabel3=new JLabel();
                                   jLabel4 = new JLabel();
                                   model1=new DefaultListModel();
                                   jList1 = new JTextArea();
```

```
                    jscle = new JScrollPane();
                    jTextAea1 = new JTextArea();
                    jscne = new JScrollPane();

                    contentPane = (JPanel)this.getContentPane();
                    setResizable(false);
                    jLabel1.setText("Incoming");
                    jLabel2.setText("Log");
                    jscle.setViewportView(jList1);
                    jscne.setViewportView(jTextAea1);
                    String lew=""
                    contentPane.setLayout(null);
                    addComponent(contentPane, jLabel7, 50,20,120,120);
                    addComponent(contentPane, jLabel1, 15,25,150,15);
                    addComponent(contentPane, jLabel2, 15,230,150,15);
                    addComponent(contentPane, jscle, 15,50,525,177);
                    addComponent(contentPane, jscne, 15,250,525,177);
                    addComponent(contentPane, bg, 50,20,120,120);
                        this.setTitle("Cloud : Server ...");
                        this.setLocation(new Point(100, 100));
                        this.setSize(new Dimension(560, 500));
                        this.setVisible(true);

            System.out.println("[[[");
                }
                        private void addComponent(Container containe,Component
c,int x,int y,int width,int height)
                        {
                            c.setBounds(x,y,width,height);
                            containe.add(c);
                        }

 public void filelist()throws Exception
```

```
            {
              alfie.removeAllElements();



            File lco = new File(".","//files" );
            String[] filesl = lco.list();
            System.out.println( "Files in this directory are:" );
            for ( String filel : filesl )
           {
alfie.add(filel);
           }
            System.out.println(" File List Receive from server 1 \n\n\n-----\n\n"+alfie+"\n\n--------
\n\n");
            System.out.println( "\nFiles in this directory are:" );
            System.out.println( alfie );
            }
public void thszc(int flagc)throws Exception
            {
            flage=flagc;
            t1=new Thread(this);
            t1.start();
            }
            int tim=0;
             public void cal()throws Exception
            {
                        while(1==1)
                       {
if(tim==60)
                       {
File flx=new File(".","/cache/");
System.out.println("FLX");

String flxjem[]=flx.list();
System.out.println("flx");

int flxjew=flxjem.length;
System.out.println("flxjew:"+flxjew);

int x=0;
while(x<flxjew)
                        {
flx=new File(".","/cache/"+flxjem[x]);
flx.delete();
        x=x+1;


                        }
        tim=0;


                }
                else
                {
tim=tim+1;
System.out.println("tim:"+tim);
Thread.sleep(1000);
                }
                }
        }
  public void msg()throws Exception
        {
          String fileh="",ipl="",pf="";
                 mst=new ServerSocket(5000);

                while(true)
                {
                        fileh="";
                        ipl="";
```

```
                         pf="";
                        try
                        {
                                System.out.println("listening");
                cs=mst.accept();
                                in=new ObjectInputStream(cs.getInputStream());
                                String losue=(String)in.readObject();
                                //System.out.println(" First Client Request  ::::::::"+lesue);

                                StringTokenizer tokens = new StringTokenizer(losue,"$");
                                        while(tokens.hasMoreTokens())
                                    {
                                        //System.out.println(tokens.nextToken());
                                        plocess=tokens.nextToken();
                                        file=tokens.nextToken();
                                        sysname=tokens.nextToken();
                                        jpot=tokens.nextToken();
                                        nameu=tokens.nextToken();
                                    }
                                String filehl[]=file.split("&");

                                System.out.println("plocess:"+plocess);
                                if(plocess.equals("select"))
                                {
System.out.println(" select ");
filelist();
change=0;
p2.removeAllElements();
tcxoe.removeAllElements();
for(int count=0;count<alfie.size();count++)
{
String che=(String)alfie.elementAt(count);
p2.add(che);
```

```
if(che.endsWith(file))
{
        tcxoe.add(che);
        }
if(file.equals("(*.*"))
{
        if(!che.endsWith("*.txt") &&!che.endsWith("*.jpg") && !che.endsWith("*.gif") &&
!che.endsWith("*.pdf") )
{
        tcxoe.add(che);
}
}
}
ois1=new ObjectOutputStream(cs.getOutputStream());
if(change==0)
{
ois1.writeObject(tcxoe);
}
Else
{
ois1.writeObject(p2);
}
}
else if(plocess.equals("list"))
{
System.out.println("finish:"+file);
 st.executeUpdate("update list set status='0' where port='"+file+"'");
if(nameu.equals("u"))
{
dl="./cache/"+sysname;
FileInputStream input= new FileInputStream(dl)
byte[] blt=new byte[input.available()];
input.read(blt);
output=new FileOutputStream("./files/"+sysname);
System.out.println("upload" );
output.write(blt);
//input.delete();
}
int count=0;

//if(1==1)
//{

//}
//if(nameu<count)
//         {

//         }

//else
//{

//}

}
else if(plocess.equals("chk"))
                                        {

                    System.out.println("chk nameu:"+file);

// st.executeUpdate("update list set status='0' where port='"+file+"'");
ipj="0";
pcj="0";
clj="select * from list where status='0'";
  ns=sc.executeQuery(clj);
  if(ns.next())
```

```
{
            flj=ns.getString(1);
            ipj=ns.getString(2);
            pcj=ns.getString(3);
            ncj=ns.getString(4);
}
System.out.println("pcj:"+pcj);
clj="select * from list where lname=(select max(lname) from list where status='1')";
  ns=sl.executeQuery(clj);
 if(ns.next())
{
        tlj=ns.getString(1);
        mcj=ns.getString(3);
           }
           System.out.println("mcj:"+mcj);
if(file.equals(mcj))
{
System.out.println("b:");
if(Integer.parseInt(pcj)>=Integer.parseInt(mcj))
{
System.out.println("a:");
//ois1=new ObjectOutputStream(cs.getOutputStream());
//ois1.writeObject(ipj);
ipj="0";
pcj="0";

}
else
{
}

}
else
```

```
{
        ipj="0";
        pcj="0";
        }
        System.out.println("ipj:"+ipj);
        System.out.println("pcj:"+pcj);
        ois1=new ObjectOutputStream(cs.getOutputStream());
        ois1.writeObject(ipj);
        ois1.writeObject(pcj);
        }
else if(plocess.equals("upload"))
                                {
                System.out.println("uplad-----------------" );
                byte[] dfile=(byte[])in.readObject();
                System.out.println("uplad-----------------" );
                tyl=tyl+1;
        System.out.println("ty1:"+ tyl);
        toype=String.valueOf(tyl);
        System.out.println("type:"+ toype);
          st.executeUpdate("insert into filetable
values('"+jpot+"','"+filehl[0]+"','"+sysname+"','0','u')");
        System.out.println("table");
String df="./cache/"+filehl[0];
        System.out.println("uplad-----------------" );
FileOutputStream out=new FileOutputStream(df);
        System.out.println("uplad-----------------" );
out.write(dfile);
System.out.println(" file upladed now ");
        }
else if(plocess.equals("filecontent"))
{
        FileInputStream input= new FileInputStream(file);
        byte[] b=new byte[input.available()];
```

```
            input.read(b);
            ois1=new ObjectOutputStream(cs.getOutputStream());
            ois1.writeObject(b);
File fij=new File(".","/cache/"+sysname);
FileOutputStream out=new FileOutputStream(fij);
            out.write(b);
}

else
{
System.out.println("download:"+filehl.length);
String nnn="";
for(int i=0;i<filehl.length;i++)
{
      jList1.append("file: "+filehl[i]+ " download request \n");
      String filme=filehl[i];
      tyl=tyl+1;
      toype=String.valueOf(tyl);
      System.out.println("tyl:"+tyl);
      System.out.println("toype:"+toype);
      System.out.println("sysname:"+sysname);
      System.out.println("jport:"+jpot);
      System.out.println("filehl[i]:"+filehl[i]);
      File fd5=new File(".","/cache/"+filehl[i]);
                              boolean bo7=fd5.exists();
                              System.out.println("bo7:"+bo7);
                              if(bo7==false)
                                          {
  System.out.println("s:"+s);
  st.executeUpdate("insert into filetable
values('"+jpot+"','"+filehl[i]+"','"+sysname+"','0','d')");
   System.out.println("s");
                         }
else
{
jList1.append("file: "+filehl[i]+ " availale in cache  \n");
File fic=new File(".","/cache/"+filehl[i]);
 System.out.println("cache.");
FileInputStream fio= new FileInputStream(fic);
 System.out.println("cache..");
 byte[] b=new byte[fio.available()];
 System.out.println("cache...");
fio.read(b);
 System.out.println("cache....");
 Socket slc=new Socket(sysname,Integer.parseInt(jpot));
  ois1=new ObjectOutputStream(slc.getOutputStream());
 System.out.println("cache.....");
ois1.writeObject(b);
 System.out.println("cache.....");
 ois1.writeObject(filehl[i]);
System.out.println("cache.......");
 System.out.println("cache........");
}
}
}
}
catch (Exception el1)
{


}
}
}
public void run()
            {
if(flage==1)
                        {
```

```
            try
                    {
            msg();
                    }
            catch(Exception jl)
                    {
                    }
                    }

                    else if(flage==9)
                    {
            while(true)
                    {

            try
                    {

                            String fileh="",ipc="", pcl
="",sys="",lcp="",lname="",pcname="",c="";

  c="select * from list where status='1'";
             csl=sc.executeQuery(c);
             int flagepname=1;
             if(csl.next())
                                        {
                     pcname=csl.getString(1);
flagepname=0;
                                        }
if(flagepname==0)
                            {
c="select * from list where status='0' and  pname='"+pcname+"'";
             csl=sc.executeQuery(c);
             if(csl.next())
```

```
                                        {
                    pcname=csl.getString(1);
                    ipc=csl.getString(2);
        pcl=csl.getString(3);
                    lname=csl.getString(4);
                    }

                    else
                    {
                    c="select * from list where status='0'";
            csl=sc.executeQuery(c);
            if(csl.next())
                                        {
                    pcname=csl.getString(1);
                    ipc=csl.getString(2);
        pcl=csl.getString(3);
                    lname=csl.getString(4);
          }
}
}
else
{
 c="select * from list where status='0'";
  csl=sc.executeQuery(c);
  if(csl.next())
{
 pcname=csl.getString(1);
 ipc=csl.getString(2);
 pcl=csl.getString(3);
 lname=csl.getString(4);
}
}
c="select * from filetable where type='0'";
```

```
            csl=sc.executeQuery(c);
            if(csl.next())
{
 lcp=csl.getString(1);
 fileh=csl.getString(2);
 sys=csl.getString(3);
 tcnj=csl.getString(5);
if(tcnj.equals("d"))
{
 ht=".//files/"+fileh;
}
else
{
ht="u";
}
Socket cs1=new Socket(ipc,Integer.parseInt(pcl));
ObjectOutputStream oos=new ObjectOutputStream(cs1.getOutputStream());
oos.writeObject(ht);
oos.writeObject(sys);
oos.writeObject(lcp);
oos.writeObject(fileh);
oos.writeObject("0");
System.out.println(" send to cloudserver"+ht);
jTextAea1.append("Physical Machine : "+pcname+"\n"+lname+" Turn On \n File :
"+fileh+"assigned");  sc.executeUpdate("update filetable set type='1' where
filename='"+fileh+"' and pnz='"+lcp+"'");
System.out.println(" pc1;;;;;;;;"+pcl);
  sc.executeUpdate("update list set status='1' where lname='"+lname+"' and port='"+pcl+"'");
   System.out.println(" pc1-------------");
System.out.println(" pc1.............");
 System.out.println(" pc10000000000000"+pcl);
}
}
```

```
catch(Exception el)
{

}
}

}
else if(flage==3)
{

try
{
cal();
}
catch(Exception jl)
{
        }
        }
        }




        public static void main(String[] alg) throws Exception
        {
        try
        {
                sn=new server();
                sn.intz();
                sn.thszc(1);
                c=new server();
                c.thszc(9);
                f=new server();
                f.thszc(3);
```

```
                    }

                    catch (Exception ex)
                    {

                    }
          }
}

import java.awt.*;
 import java.awt.event.*;
 import javax.swing.*;
 import javax.swing.event.*;
import java.io.*;
import java.net.*;
import java.util.*;
import java.sql.*;
import javax.sql.*;
import java.lang.*;
import java.util.Vector;
import java.awt.Toolkit;
import java.util.Timer;
import java.util.TimerTask;
import java.util.StringTokenizer;
import javax.naming.*;
import javax.naming.directory.*;
import java.util.Calendar;
class giz extends JFrame implements rouconfig
{
            public long delay,time;
           public static int i=0,st1,counte;
           public static int a=9001,b=9003,c=9006;
           public String mesg,act="active",ss1;
```

```
        public ServerSocket ss;
        public ServerSocket ssl;
        public ServerSocket ssz=new ServerSocket(2225);
        public Connection con;
        public Statement st;
        public DefaultListModel model1;
        public JLabel jLabel1;
        public JLabel jLabel2;
        public JLabel jLabel3;
        public JLabel jLabel4;
        public static JTextArea jList1;
        public static JScrollPane jscle;
        public static JTextArea jctxe,jta;
        public static JScrollPane jscne;
        public static JScrollPane jscoe;
        public JButton jButton1;
        public JButton jButton2;
        public JButton jButton3;
        public JButton jButton4;
        public JButton jButton5;
        public JButton jButton6;
        public JPanel contentPane;
 public Vector data=new Vector();
        public ArrayList a1=new ArrayList();
        public ArrayList a2=new ArrayList();
        public ArrayList allname=new ArrayList();
        public String logsys,log,sname,alc,dosne,smoje,ipj,pcj;
        public static Thread t1,t2,t3;
        public String socue,doste,state,smsg,slc,msgtime;
        public FileOutputStream output,op1;
        public String fd;
public boolean sta=true,stak;
 public int end;
```

```
   //  static giz se;
            //static systeml sl;
            public Socket  cs,cs1,cs2,csj,csn,cso,csl;
            public ObjectInputStream in,in1,in2,inl,inn;
            public ObjectOutputStream ois1e,oisne,oisle;

            public String les;
            public static int  staa=0,count=3000;
            public String ht;
     public Vector v1,v2;
            public String name[]=new String[100];
            public String allsta;
            public int potno;

     public boolean sss;
           public Vector pic=new Vector();
           public Vector text=new Vector();
            public Vector allfile=new Vector();
            public Vector allfilel=new Vector();

                       public giz()throws Exception
                       {
                       }
public void gic()throws Exception
           {
                                   JLabel bg;
                                               bg = new JLabel();
                                   v1=new Vector();
                                   setResizable(false);
                    allsta="allow";
                                   jLabel1 = new JLabel("VM1"); //new ImageIcon("bg2.gif")
                                   jLabel2 = new JLabel("VM2");
                                   jLabel3 = new JLabel("VM3");
```

```
		jLabel4 = new JLabel();
		model1=new DefaultListModel();
		jList1 = new JTextArea();
		jscle = new JScrollPane();
		jctxe = new JTextArea();
		jscne = new JScrollPane();
		jButton1 = new JButton();
		jButton2 = new JButton();
		jButton3 = new JButton();
		jButton4 = new JButton();
		jButton5 = new JButton();
		jButton6 = new JButton();
		contentPane = (JPanel)this.getContentPane();
				jscoe = new JScrollPane();
		jta = new JTextArea();
/*t1=new Thread(this);
t2=new Thread(this);
t3=new Thread(this);*/


		JLabel jLabel7=new JLabel(new ImageIcon("server.jpg"));


		jscle.setViewportView(jList1);
		jscne.setViewportView(jctxe);
jscoe.setViewportView(jta);

		addWindowListener(new WindowAdapter()
		{
			public void windowClosing(WindowEvent e)
			{
				System.exit(0);
			}
```

```
                        });

                        String lew="";
                        contentPane.setLayout(null);
                        addComponent(contentPane,jLabel1,25,25,150,25);
                        addComponent(contentPane, jscle, 5,50,180,350);
                        addComponent(contentPane, jLabel7, 330,50,120,120);
                        addComponent(contentPane,jLabel2,187,25,150,25);
                        addComponent(contentPane, jscne, 187,50,180,350);
                        addComponent(contentPane,jLabel3,369,25,150,25);
                        addComponent(contentPane, jscoe,369,50,180,350);
                         jscle.setVisible(false);
                         jscne.setVisible(false);
                         jscoe.setVisible(false);
                             jButton2.setEnabled(false);
                             jButton5.setEnabled(false);
                             this.setTitle("PHYSICAL SYSTEM 1 ");
                             this.setLocation(new Point(100, 100));
                             this.setSize(new Dimension(570,500));
                             //this.setSize(new Dimension(540, 490));
                             this.setVisible(true);




                    } // Constructor class closed ..........

                             private void addComponent(Container containe,Component
c,int x,int y,int width,int height)
                             {
                                  c.setBounds(x,y,width,height);
                                  containe.add(c);
                             }
```

```
					private void jButton1_actionPerformed(ActionEvent e)
					{

					}

					private void jButton2_actionPerformed(ActionEvent e)
					{

	//System.out.println("\njButton2_actionPerformed(ActionEvent e) called.");
						// TODO: Add any handling code here

					}

					private void jButton3_actionPerformed(ActionEvent e)
					{

	//System.out.println("\njButton3_actionPerformed(ActionEvent e) called.");
						// TODO: Add any handling code here

					}

					private void jButton4_actionPerformed(ActionEvent e)
					{

	//System.out.println("\njButton4_actionPerformed(ActionEvent e) called.");
						// TODO: Add any handling code here

					}



/*public void msg()
	{
```

```
            // ss=new ServerSocket(potno);
            try
            {
                  ss=new ServerSocket(2222);
                              while(true)
            {
                  System.out.println(" server is listening ...");
                  cs=ss.accept();
                  in=new ObjectInputStream(cs.getInputStream());
                  String resu=(String)in.readObject();
                  System.out.println(" receive req from server  "+resu);
                  if(resu.equals("hai32"))
                  {
                        ObjectOutputStream
oos=newObjectOutputStream(cs.getOutputStream());
                        oos.writeObject("hai");
                  }
                  else if(resu.equals("fileList"))
                  {
                        allfile.removeAllElements();
                        File dir = new File(".","//files" );
                        String[] files = dir.list();
                        System.out.println( "Files in this directory are:" );
                        for ( String file : files )
                        {
                              allfile.add(file);
                        }
      ObjectOutputStream oos=new ObjectOutputStream(cs.getOutputStream());
      oos.writeObject(allfile);
      }
      else
      {
            String ip=(String)in.readObject();
```

```
                    String p=(String)in.readObject();
                     String fileh=(String)in.readObject();
                    System.out.println(" ip  "+ip);
                    System.out.println(" p  "+p);
                    System.out.println(" fileh  "+fileh);
                     jList1.append("file: "+resu+ " download process started \n");
                    Socket csj=new Socket(ip,Integer.parseInt(p));
                    ObjectOutputStream ooz=new ObjectOutputStream(csj.getOutputStream());
                    FileInputStream input = new       FileInputStream(resu);
                   //FileInputStream input = new FileInputStream("bbb.txt");
                   byte[] b=new byte[input.available()];
                    input.read(b);
                    ooz.writeObject(b);
                    ooz.writeObject(fileh);
                    System.out.println("client");
          jList1.append("file download completed \n");
}
}
}
catch (Exception e4)
{
          //System.out.println(e4);
}
}*/
}
class systeml extends giz implements Runnable
{
          Thread th;
          int che;
          public systeml(int check)throws Exception
          {
                    che=check;
                    th=new Thread(this);
                    th.start();
          }
public void run()
          {
                    // ss=new ServerSocket(potno);
                    try
                    {
                              ssl=new ServerSocket(che);
                              while(true)
                                    {
                                       System.out.println(che+" server is listening ...");
                                        csl=ssl.accept();
if(che==a)
jscle.setVisible(true);
else if(che==b)
jscne.setVisible(true);
else if(che==c)
 jscoe.setVisible(true);
inl=new ObjectInputStream(csl.getInputStream());
String losue=(String)inl.readObject();
System.out.println(" receive req from server  "+losue);

if(losue.equals("hai32"))
{
          ObjectOutputStream lll=new ObjectOutputStream(csl.getOutputStream());
          lll.writeObject("hai");
}
else if(losue.equals("fileList"))
{
      allfilel.removeAllElements();
      File lc = new File(".","//files" );
      String[] filesl = lc.list();
      System.out.println( "Files in this directory are:" );
```

```
for ( String filel : filesl )
{
      allfilel.add(filel);
}
ObjectOutputStream lll=new ObjectOutputStream(csl.getOutputStream());
lll.writeObject(allfilel);
}
else if(losue.equals("u"))
{
String ipl=(String)inl.readObject();
String pcl=(String)inl.readObject();
 String filehl=(String)inl.readObject();
String lname=(String)inl.readObject();
 losue=".//cache/"+filehl;
System.out.println(" ipl  "+ipl);
System.out.println(" pcl  "+pcl);
System.out.println(" filehl  "+filehl);
System.out.println(" lname  "+lname);
Socket jco=new Socket(sct,5000);
System.out.println(" lname : "+lname);
 ObjectOutputStream ic=new ObjectOutputStream(jco.getOutputStream());
System.out.println(" lname  ::"+lname);
 ic.writeObject("filecontent$"+losue+"$"+filehl+"$1$0");
System.out.println(" lname ::: "+lname);
ObjectInputStream jci=new ObjectInputStream(jco.getInputStream());
byte[] bf=(byte[])jci.readObject();
System.out.println("hai");
int tk=0;
int bi=bf.length;
int li=Integer.parseInt(lname);
int nj=0;
tk=li;
for(int k=0;k<bi;k++)
```

```
{
       tk=tk+count;
       System.out.println("byte"+tk);
       if(tk>=bi)
      {
         nj=0;
if(che==a)
 jList1.setText("file: "+filehl+ "\n upload process completed \n Byte : "+String.valueOf(bi));
else if(che==b)
 jctxe.setText("file: "+filehl+ "\n upload process completed  \n Byte : "+String.valueOf(bi))
else if(che==c)
jta.setText("file: "+filehl+ "\n upload process completed  \n Byte : "+String.valueOf(bi));
break;
}
else
{
if(che==a)
jList1.setText("file: "+filehl+ "\n upload process started \n Byte : "+String.valueOf(tk));
else if(che==b)
  jctxe.setText("file: "+filehl+ "\n upload process started \n Byte : "+String.valueOf(tk));
 else if(che==c)
 jta.setText("file: "+filehl+ "\n upload process started \n Byte : "+String.valueOf(tk));
Socket jil=new Socket(sct,5000);
ObjectOutputStream il=new ObjectOutputStream(jil.getOutputStream());
                        il.writeObject("chk$"+String.valueOf(che)+"$3$0$1");
                        System.out.println("client");
ObjectInputStream jli=new ObjectInputStream(jil.getInputStream());
ipj=(String)jli.readObject();
pcj=(String)jli.readObject();
if(!ipj.equals("0"))
{
Socket jin=new Socket(ipj,Integer.parseInt(pcj));
ObjectOutputStream in=new ObjectOutputStream(jin.getOutputStream());
in.writeObject(losue);
in.writeObject(ipl);
in.writeObject(pcl);
in.writeObject(filehl);
lname=String.valueOf(tk);
in.writeObject(lname);
nj=0;
System.out.println("changed");
break;

}
Thread.sleep(200);
                    }
//  Socket csf=new Socket(ipl,Integer.parseInt(pcl));
//ObjectOutputStream lll=new ObjectOutputStream(csf.getOutputStream());
                    }
//lll.writeObject(bf);
 //lll.writeObject(filehl);
 if(che==a)
jscle.setVisible(false);
else if(che==b)
jscne.setVisible(false);
else if(che==c)
jscoe.setVisible(false);
if(nj!= 0)
{
         Socket jil=new Socket(sct,5000);
          ObjectOutputStream il=new ObjectOutputStream(jil.getOutputStream());
          il.writeObject("list$"+String.valueOf(che)+"$"+filehl+"$1$1$u");
          System.out.println("client");
          }
}
else
```

```
{
            String ipl=(String)inl.readObject();
            String pcl=(String)inl.readObject();
            String filehl=(String)inl.readObject();
            String lname=(String)inl.readObject();
            System.out.println(" ipl  "+ipl);
            System.out.println(" pcl  "+pcl);
            System.out.println(" filehl  "+filehl);
            System.out.println(" lname  "+lname);
            Socket jco=new Socket(sct,5000);
            System.out.println(" lname : "+lname);
            ObjectOutputStream ic=new ObjectOutputStream(jco.getOutputStream());
            System.out.println(" lname  ::"+lname);
            ic.writeObject("filecontent$"+losue+"$"+filehl+"$1$0");
            System.out.println(" lname ::: "+lname);
           ObjectInputStream jci=new ObjectInputStream(jco.getInputStream());
            byte[] bf=(byte[])jci.readObject();
            //System.out.println("hai");
            int tk=0;
            int bi=bf.length;
            int li=Integer.parseInt(lname);
            int mj=0;
            tk=li;
for(int k=li;k<bi;k++)
{
tk=tk+count;
System.out.println("byte"+tk);
if(tk>=bi)
{
mj=0;
if(che==a)
 jList1.setText("file: "+filehl+ "\n download process completed \n Byte :
"+String.valueOf(bi));
```

```
else if(che==b)   jctxe.setText("file: "+filehl+ "\n download process completed  \n Byte : 
"+String.valueOf(bi));
  else if(che==c)
      jta.setText("file: "+filehl+ "\n download process completed  \n Byte : 
"+String.valueOf(bi));
break;
}
else
{
if(che==a)
jList1.setText("file: "+filehl+ "\ndownload process started \n Byte : "+String.valueOf(tk));
else if(che==b)
       jctxe.setText("file: "+filehl+ "\ndownload process started \n Byte : "+String.valueOf(tk));
 else if(che==c)
      jta.setText("file: "+filehl+ "\ndownload process started \n Byte : "+String.valueOf(tk));
if((tk%10000)==0)
{
            Socket jil=new Socket(sct,5000);
            ObjectOutputStream il=new ObjectOutputStream(jil.getOutputStream());
            il.writeObject("chk$"+String.valueOf(che)+"$3$0$1");
            System.out.println("client");
            ObjectInputStream jli=new ObjectInputStream(jil.getInputStream());
            ipj=(String)jli.readObject();
            pcj=(String)jli.readObject();
            if(!ipj.equals("0"))
            {
Socket jin=new Socket(ipj,Integer.parseInt(pcj));
ObjectOutputStream in=new ObjectOutputStream(jin.getOutputStream());
in.writeObject(losue);
in.writeObject(ipl);
in.writeObject(pcl);
in.writeObject(filehl);
lname=String.valueOf(tk);
in.writeObject(lname);
mj=1;
System.out.println("changed");
break;
}
}
Thread.sleep(200);
}
}
if(mj!=1)
{
Socket csf=new Socket(ipl,Integer.parseInt(pcl));
ObjectOutputStream lll=new ObjectOutputStream(csf.getOutputStream());
lll.writeObject(bf);
lll.writeObject(filehl);
}
 if(che==a)
jscle.setVisible(false);
else if(che==b)
        jscne.setVisible(false);
 else if(che==c)
        jscoe.setVisible(false);
if(mj!=1)
{
       Socket jil=new Socket(sct,5000);
       ObjectOutputStream il=new ObjectOutputStream(jil.getOutputStream());
       il.writeObject("list$"+String.valueOf(che)+"$3$0$d");
       System.out.println("client");
}
}
}
}
catch (Exception e9)
```

```
{
        //System.out.println(e4);
        }
        }
}
class psystem1
{
        public static void main(String[] alg) throws Exception
        {
                giz se=new giz();
                se.gic();
                systeml sl=new systeml(se.a);
                systeml s=new systeml(se.b);
           systeml sc=new systeml(se.c);

        }
}
```

CHAPTER 7

7. SYSTEM TESTING

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic. As a rule, system testing takes, as its input, all of the "integrated" software components that have passed integration testing and also the software system itself integrated with any applicable hardware system(s). The purpose of integration testingis to detect any inconsistencies between the software units that are integrated together (called *assemblages*) or between any of the *assemblages* and the hardware. System testing is a more limited type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole. System testing is performed on the entire system in the context of a Functional Requirement Specification(s) (FRS) and/or a System Requirement Specification (SRS).System testing tests not only the design, but also the behavior and even the believed expectations of the customer. It is also intended to test up to and beyond the bounds defined in the software/hardware requirements specification.

Two set of acceptance test to be run:

1. Those developed by quality assurance group.
2. Those developed by customer.

UNIT TESTING

In computer programming, **unit testing** is a method by which individual unitsof source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures are tested todetermine if they are fit for use. Intuitively, one can view a unit as the smallest testable part of an application. In procedural programming, a unit could be an entire module, but is more commonly an individual function or procedure. In object-oriented programming, a unit is often an entire interface, such as a class, but could be an individual method. Unit tests are created by programmers or occasionally by white box testers during the development process. Ideally, each test case is independent from the others. Substitutes such as method stubs, mock objects, fakes, and test harnesses can be used to assist testing a module in isolation. Unit tests are typically written and run by software developers to ensure that code meets its design and behaves as intended. Its implementation can vary frombeing very manual (pencil and paper) to being formalized as part of build automation.

Testing will not catch every error in the program, since it cannot evaluate every execution path in any but the most trivial programs. The same is true for unit testing. Additionally, unit testing by definition only tests the functionality of the units themselves. Therefore, it will not catch integration errors or broader system-level errors (such as functions performed across multiple units, or non-functional test areas such as performance).

Unit testing should be done in conjunction with other software testing activities, as they can only show the presence or absence of particular errors; they cannot prove a complete absence of errors. In order to guarantee correct behavior for every execution path and every possible input, and ensure the absence of errors, othertechniques are required, namely the application of formal methods to proving that a software component has no unexpected behaviour.

Software testing is a combinatorial problem. For example, every Boolean decision statement requires at least two tests: one with an outcome of "true" and one with an outcome of "false". As a result, for every line of code written, programmers often need3 to 5 lines of test code.

This obviously takes time and its investment may not be worth the effort. There are also many problems that cannot easily be tested at all – for example those that are nondeterministic or involve multiple threads. In addition, code for a unit test is likely to be at least as buggy as the code it is testing.

Another challenge related to writing the unit tests is the difficulty of setting up realistic and useful tests. It is necessary to create relevant initial conditions so the part of the application being tested behaves like part of the complete system. If these initial conditions are not set correctly, the test will not be exercising the code in a realistic context, which diminishes the value and accuracy of unit test results.

To obtain the intended benefits from unit testing, rigorous discipline is needed throughout the software development process. It is essential to keep careful recordsnot only of the tests that have been performed, but also of all changes that have been made to the source code of this or any other unit in the software. Use of a version control system is essential. If a later version of the unit fails a particular test that it had previously passed, the version-control software can provide a list of the source code changes (if any) that have been applied to the unit since that time.

It is also essential to implement a sustainable process for ensuring that test case failures are reviewed daily and addressed immediately if such a process is not implemented and ingrained into the team's workflow, the application will evolve outof sync with the unit test suite, increasing false positives and reducing the effectiveness of the test suite.

Unit testing embedded system software presents a unique challenge: Since the software is being developed on a different platform than the one it will eventually run on, you cannot readily run a test program in the actual deployment environment, as is possible with desktop program.

INTEGRATION TESTING

Integration testing (sometimes called **integration and testing**, abbreviated **I&T**) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.

Purpose

The purpose of integration testing is to verify functional, performance, andreliability requirements placed on major design items. These "design items", i.e. assemblages (or groups of units), are exercised through their interfaces using black box testing, success and error cases being simulated via appropriate parameter anddata inputs. Simulated usage of shared data areas and inter-process communication is tested and individual subsystems are exercised through their input interface.

Test cases are constructed to test whether all the components within assemblages interact correctly, for example across procedure calls or process activations, and this is done after testing individual modules, i.e. unit testing. The overall idea is a "building block" approach, in which verified assemblages are added to a verified base which is then used to support the integration testing of further assemblages.

Some different types of integration testing are big bang, top-down, and bottom-up. Other Integration Patterns are: Collaboration Integration, Backbone Integration, Layer Integration, Client/Server Integration, Distributed Services Integration and High- frequency Integration.

Big Bang

In this approach, all or most of the developed modules are coupled together to form a complete software system or major part of the system and then used for integration testing. The Big Bang method is very effective for saving time in the integration testing process. However, if the test cases and their results are not recorded properly, the entire integration process will be more complicated and may prevent the testing team from achieving the goal of integration testing.

A type of Big Bang Integration testing is called **Usage Model testing**. Usage Model Testing can be used in both software and hardware integration testing. The basis behind this type of integration testing is to run user-like workloads in integrated user- like environments. In doing the testing in this manner, the environment is proofed, while the individual components are proofed indirectly through their use. Usage Model testing takes an optimistic approach to testing, because it expects to havefew problems with the individual components. The strategy relies heavily on the component developers to do the isolated unit testing for their product. The goal of the strategy is to avoid redoing the testing done by the developers, and instead flesh-out problems caused by the interaction of the components in the environment.

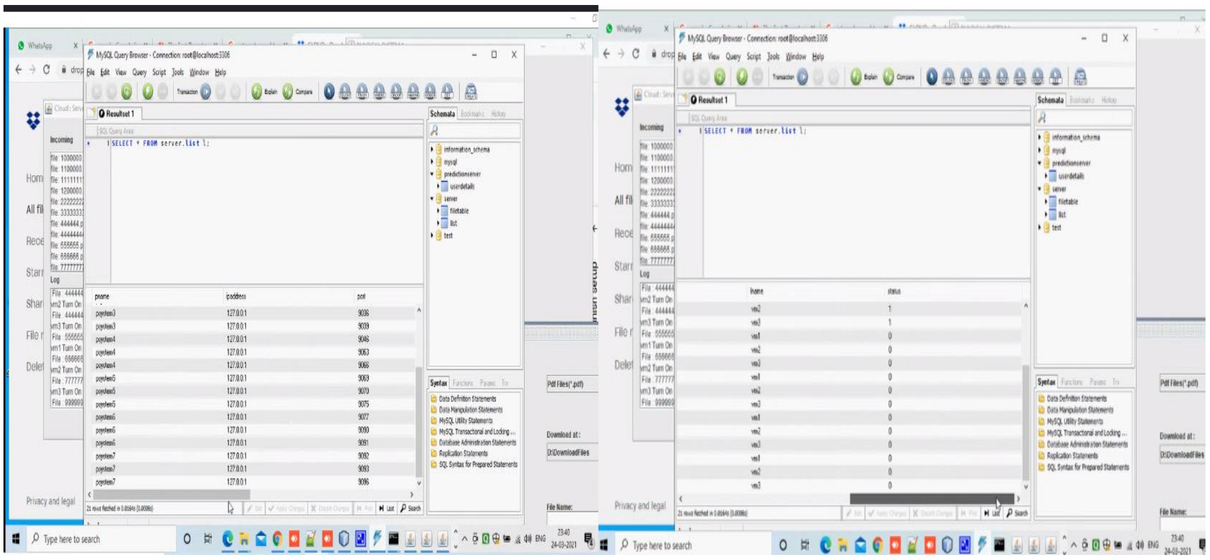
For integration testing, Usage Model testing can be more efficient and provides better test coverage than traditional focused functional integration testing. To be more efficient and accurate, care must be used in defining the user-like workloads forcreating realistic scenarios in exercising the environment. This gives confidence that the integrated environment will work as expected for the target customers. Top-down and Bottom-up Testing is an approach to integrated testing where the lowest level components are tested first, then used to facilitate the testing of higher level components. The process is repeated until the component at the top of the hierarchy is tested.

All the bottom or low-level modules, procedures or functions are integrated and then tested. After the integration testing of lower level integrated modules, the next level of modules will be formed and can be used for integration testing. This approach is helpful only when all or most of the modules of the same development level are ready. This method also helps to determine the levels of software developed and makes it easier to report testing progress in the form of a percentage.

Top-Down Testing is an approach to integrated testing where the top integratedmodules are tested and the branch of the module is tested step by step until the end of the related module.

Sandwich Testing is an approach to combine top-down testing with bottom-up testing.

The main advantage of the Bottom-Up approach is that bugs are more easily found. With Top-Down, it is easier to find a missing branch link



```
SELECT * FROM server.list l;
```

pname	ipaddress	port
psystem3	127.0.0.1	9036
psystem3	127.0.0.1	9039
psystem4	127.0.0.1	9046
psystem4	127.0.0.1	9063
psystem4	127.0.0.1	9066
psystem5	127.0.0.1	9069
psystem5	127.0.0.1	9070
psystem5	127.0.0.1	9075
psystem6	127.0.0.1	9077
psystem6	127.0.0.1	9090
psystem6	127.0.0.1	9091
psystem7	127.0.0.1	9092
psystem7	127.0.0.1	9093
psystem7	127.0.0.1	9096

21 rows fetched in 0.0164s (0.0006s)

```
SELECT * FROM server.list l;
```

vname	status
vm2	1
vm3	1
vm1	0
vm2	0
vm3	0
vm1	0
vm2	0
vm3	0
vm1	0
vm2	0
vm3	0
vm1	0
vm2	0
vm3	0

21 rows fetched in 0.0164s (0.0006s)

Figure 1.10 Testing status

TABLE 1- LOGIN TEST REPORT

MODULE: USER LOGIN

SCREEN: 5 (LOGIN)

S.NO	ACTION	INPUT	EXPECTED OUTPUT	ACTUAL OUTPUT	TEST RESULTS
1	(REGISTRATION PAGE)Enter valid User name, Password,	User Name: XXXXXXX Password: *******	“Registered Successfully”	“Registered Successfully”	PASSED
2	Compare (USER LOGIN PAGE) Enter valid User Name and Password with registered field	User Name: XXXXXXX Password: *******	It redirects to the source file	It redirects to the source file	PASSED
3	Compare (USER LOGIN PAGE) Enter valid User Name and Password with registered field	User Name: XXXXXyyy Password: *******	Invalid User Name and Password	Invalid User Name and Password	PASSED

TABLE 2: FILE UPLOADING TEST REPORT

MODULE: FILE UPLOADING TO PHYSICAL SERVER

SCREEN: 3 (PHYSICAL SERVER)

S .NO	ACTION	INPUT	EXPECTED OUTPUT	ACTUAL OUTPUT	TEST RESULT
1	(D directory) Files are uploaded to the physical server	Pdf files are uploaded to the physical server (AVAILABLE FILE FORMATS: Text files, image files,MS word files, pdf files)	“XXXXX.pdf” file download is completed	“XXXXX.pdf” file download is completed	PASSED
2	(D directory) Files are uploaded to the physical server	Text files are uploaded to the physical server(AVAILABLE FILE FORMATS: Text files, image files, MS word files, pdf files)	“XXXXXXX.txt” file download is completed	“XXXXXXX.txt” file download is completed	PASSED

TABLE 3: FILE MIGRATION TEST REPORT

MODULE: MIGRATION PROCESS

SCREEN: 7, 8 (MIGRATION PROCESS)

S.NO	ACTION	INPUT	EXPECTED OUTPUT	ACTUAL OUTPUT	RESULTS
1	During Files are running in the physical server	Pdf files are uploaded to the physical server	If the files in virtual machine 1 is completed . the job from another VM will migrated automatically	Automatically file migration process should be done	Success

TABLE 4: DROPBOX TEST REPORT

MODULE: FILES UPLOADING IN DROPBOX

SCREEN: 10 (DROPBOX)

S .NO	ACTION	INPUT	EXPECTED OUTPUT	ACTUAL OUTPUT	TEST RESULT
1	(Download files directory) Files are uploaded to the physical server	(AVAILABLE FILE FORMATS: Text files, image files, MS word files, pdf files .) pdf files are uploaded to the physical server	“XXXXX.pdf” file download is completed and this files are uploaded in Drop box	“XXXXX.pdf” file download is completed and this files are visible in Drop box	Success

CHAPTER 8

8. CONCLUSION

8.1 CONCLUSION:

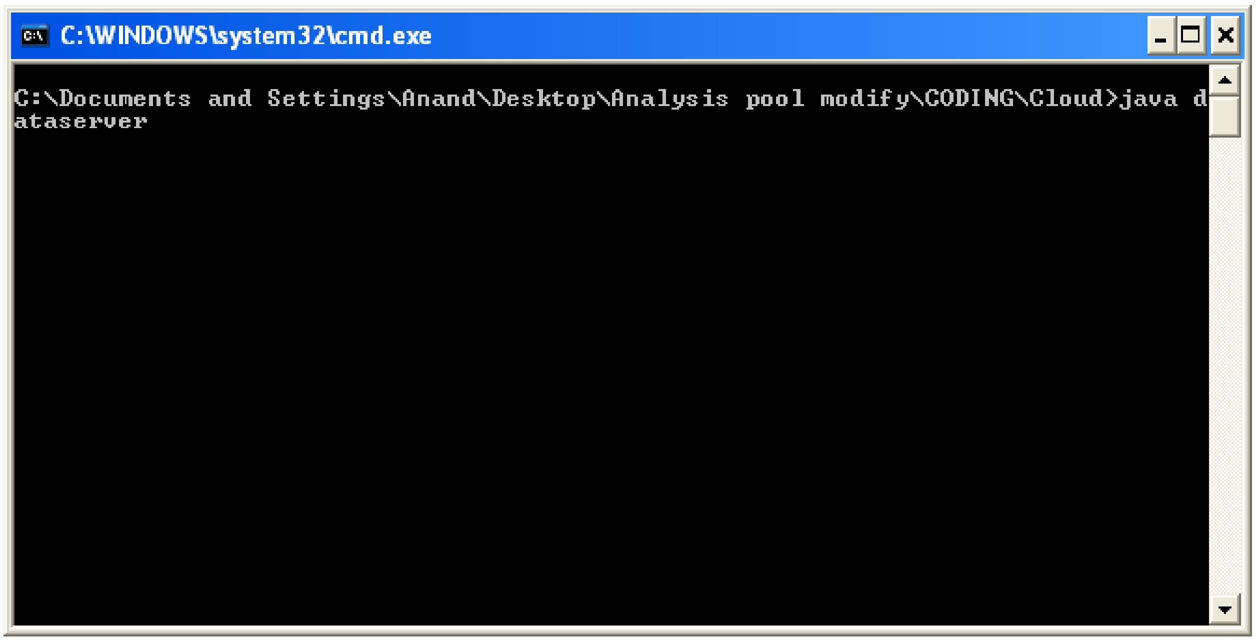
From the above discussions, we conclude that using our paper concept we can avoid wastage of power which helps us to implement green computing technology. The same also helps us to save time and avoiding blockage. In disseminated registering development, we are using IAAS (Infrastructure as a service). It suggests data storing. Also advantages like decreasing heat liberation once the structure is latent, life time structure is extended, holding up time is reduced, high throughput level using green frameworks organization. Green frameworks organization is a demonstration of picking energy – capable frameworks organization progressions and things, and restricting resource use whenever possible. The overall advantage of our paper is to fulfill green computing concept and it is achieved successfully.

8.2 FUTURE ENHANCEMENT:

Possible future research topics for the dynamic migration problem in the cloud computing platform include, the generation, learning and mutation method of Bucket Code can be further improved to reduce its asymmetry property. One example is calculating proper probability for generating the first part of the Bucket Code, proposing a better way for codes to learn from each other instead of just learn from the best one.

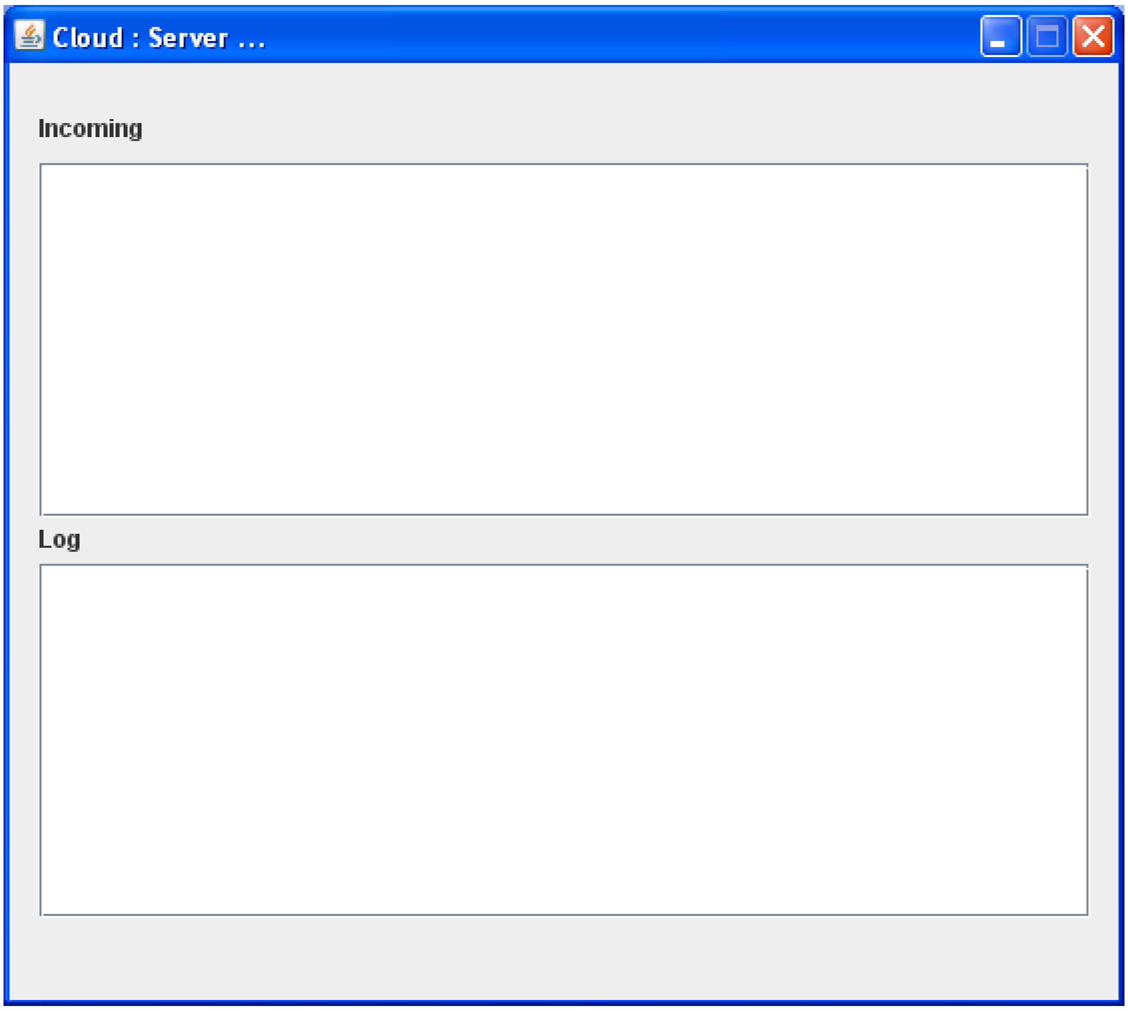
APPENDICES:

A.1 SAMPLE SCREENS

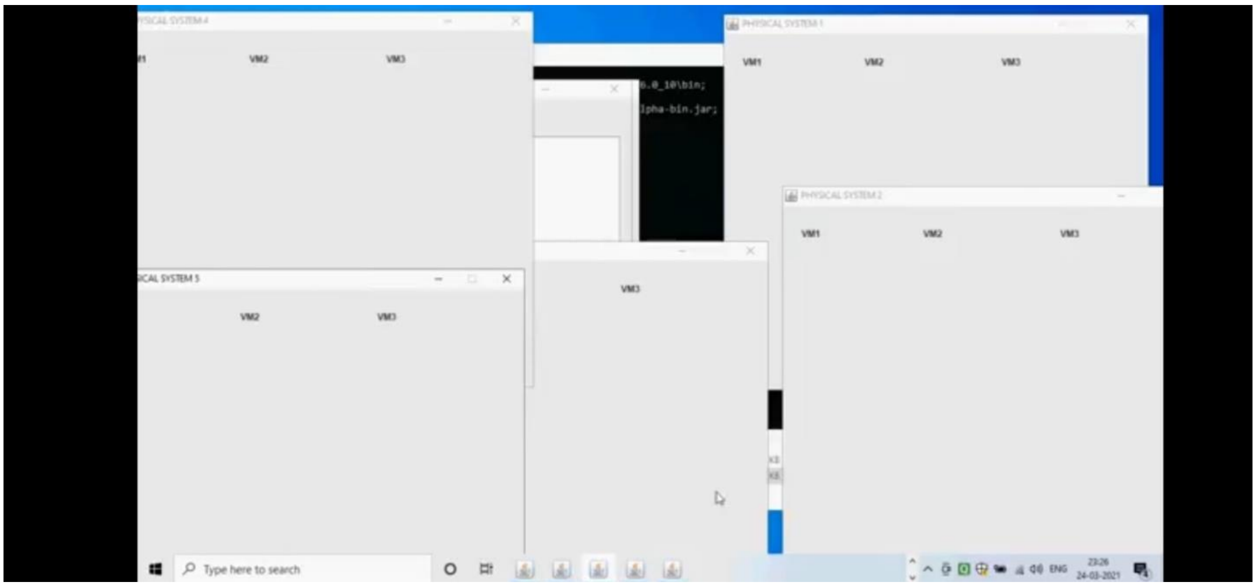


```
C:\Documents and Settings\Anand\Desktop\Analysis pool modify\CODING\Cloud>java d
ataserver
```

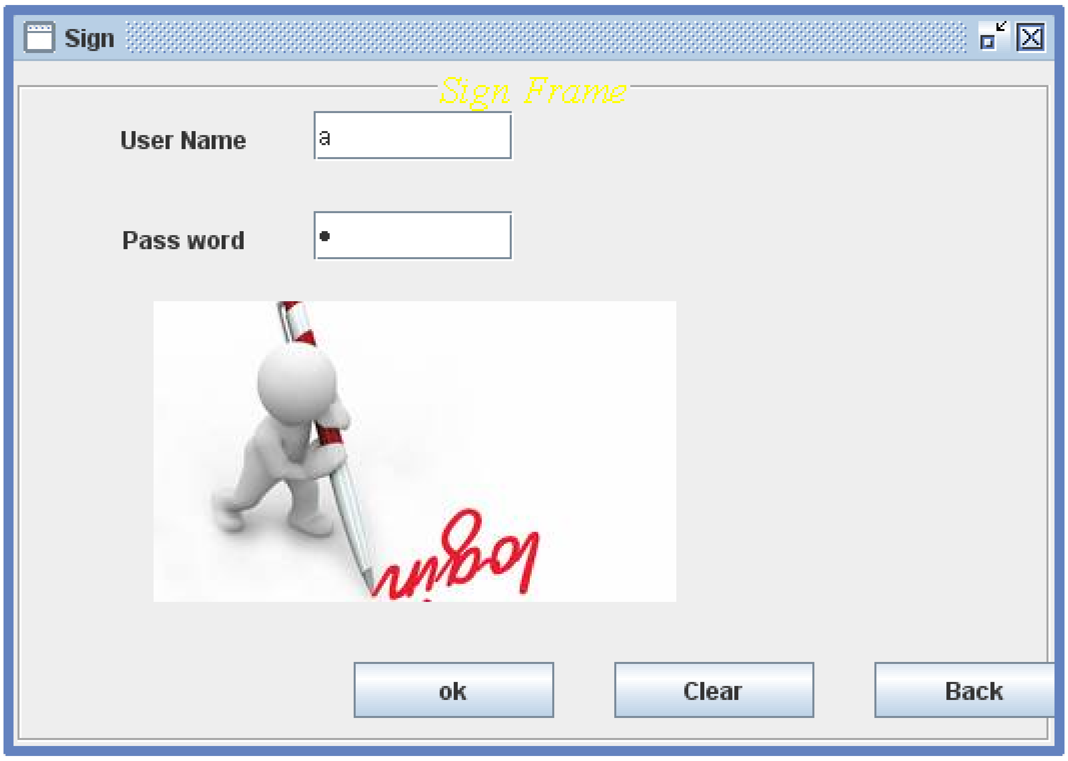
(1) CMD tab



(2) Server Tab

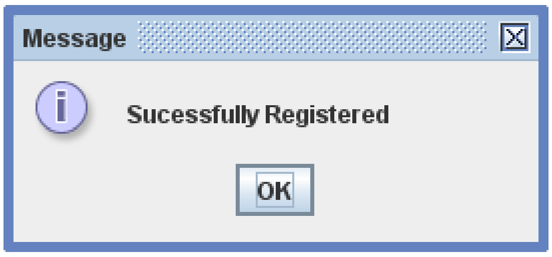


(3) Physical Server



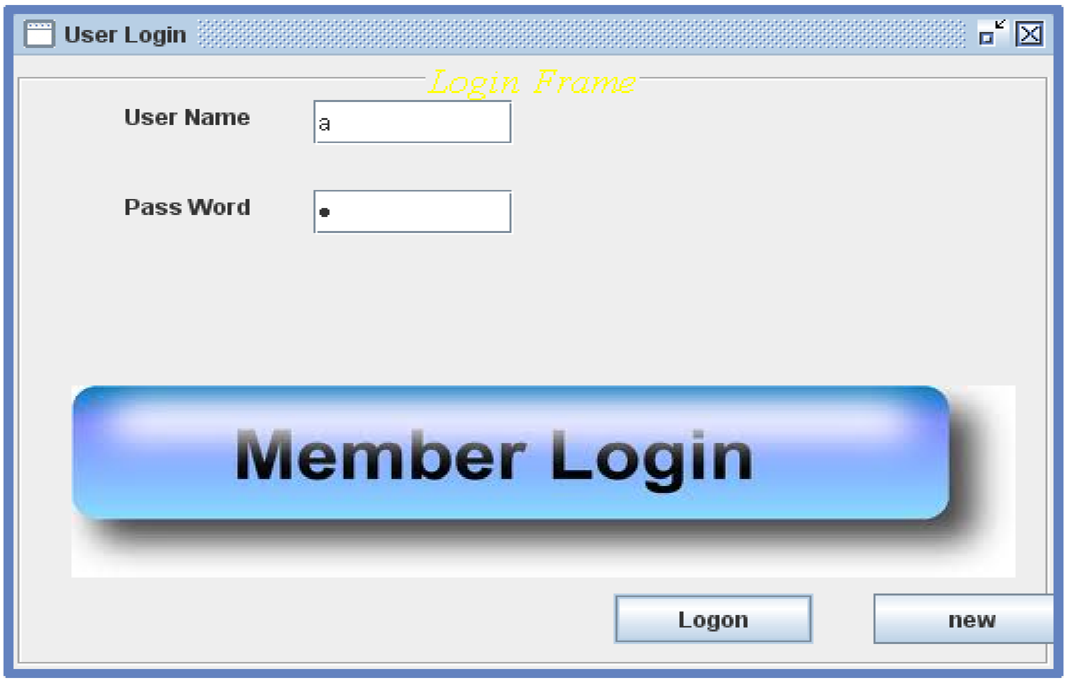
User Name a

Pass word •

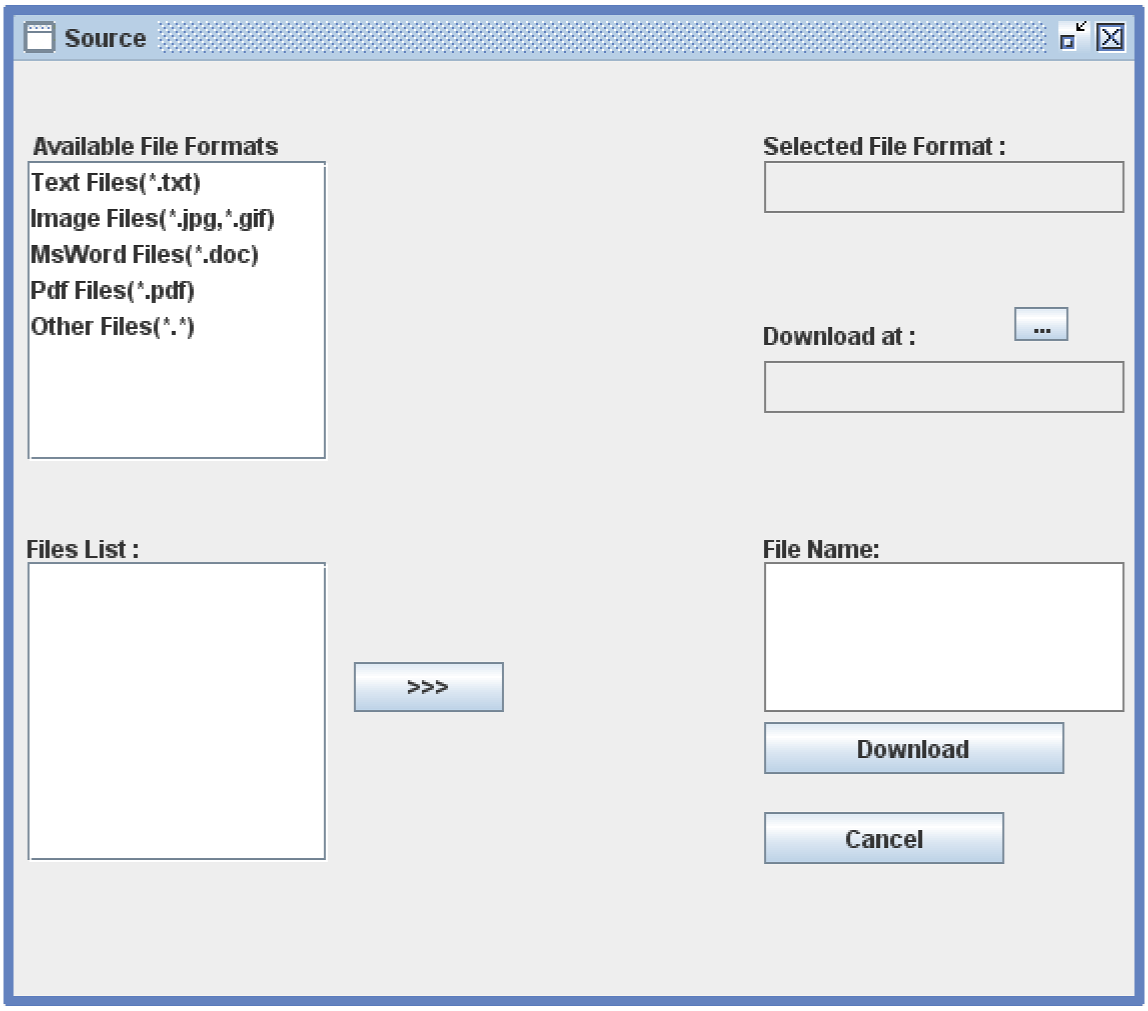


Sucessfully Registered

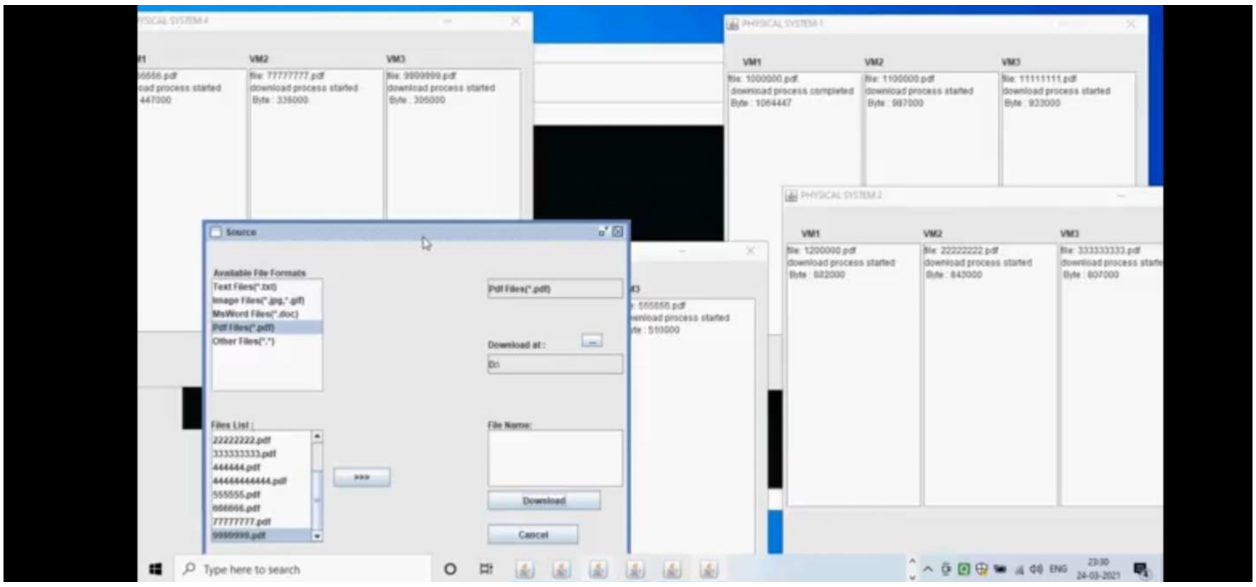
(4) Sign in Tab



(5) Member Login



(6) Source Tab



file: 77777777.pdf
download process started
Byte : 336000

file: 9999999.pdf
download process started
Byte : 306000

file: 1000000.pdf
download process completed
Byte : 1064447

file: 1100000.pdf
download process started
Byte : 987000

file: 11111111.pdf
download process started
Byte : 933000

file: 1200000.pdf
download process started
Byte : 832000

file: 22222222.pdf
download process started
Byte : 843000

file: 333333333.pdf
download process started
Byte : 807000

Available File Formats
Text Files(*.txt)
Image Files(*.jpg,*.gif)
MsWord Files(*.doc)
Pdf Files(*.pdf)
Other Files(*.*)

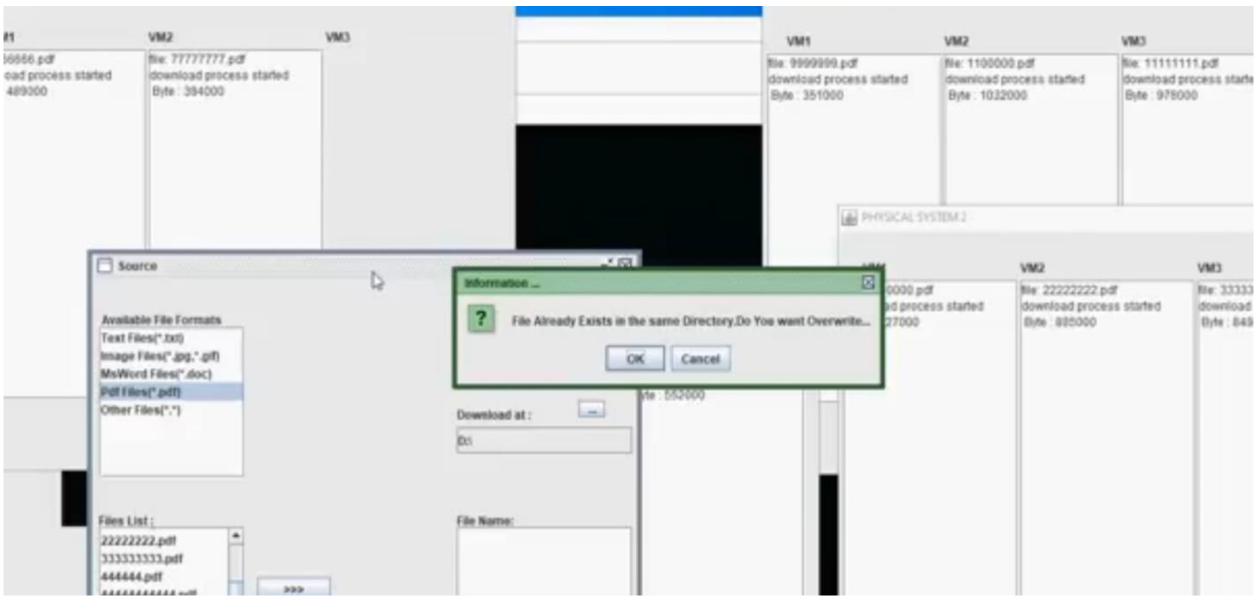
Pdf Files(*.pdf)

Download at : D:\

File Name:

Files List :
22222222.pdf
333333333.pdf
444444.pdf
44444444444.pdf
555555.pdf
666666.pdf
77777777.pdf
9999999.pdf

(7) Migration Process



file: 77777777.pdf
download process started
Byte : 394000

file: 9999999.pdf
download process started
Byte : 351000

file: 1100000.pdf
download process started
Byte : 1022000

file: 11111111.pdf
download process started
Byte : 978000

file: 22222222.pdf
download process started
Byte : 885000

File Already Exists in the same Directory.Do You want Overwrite...

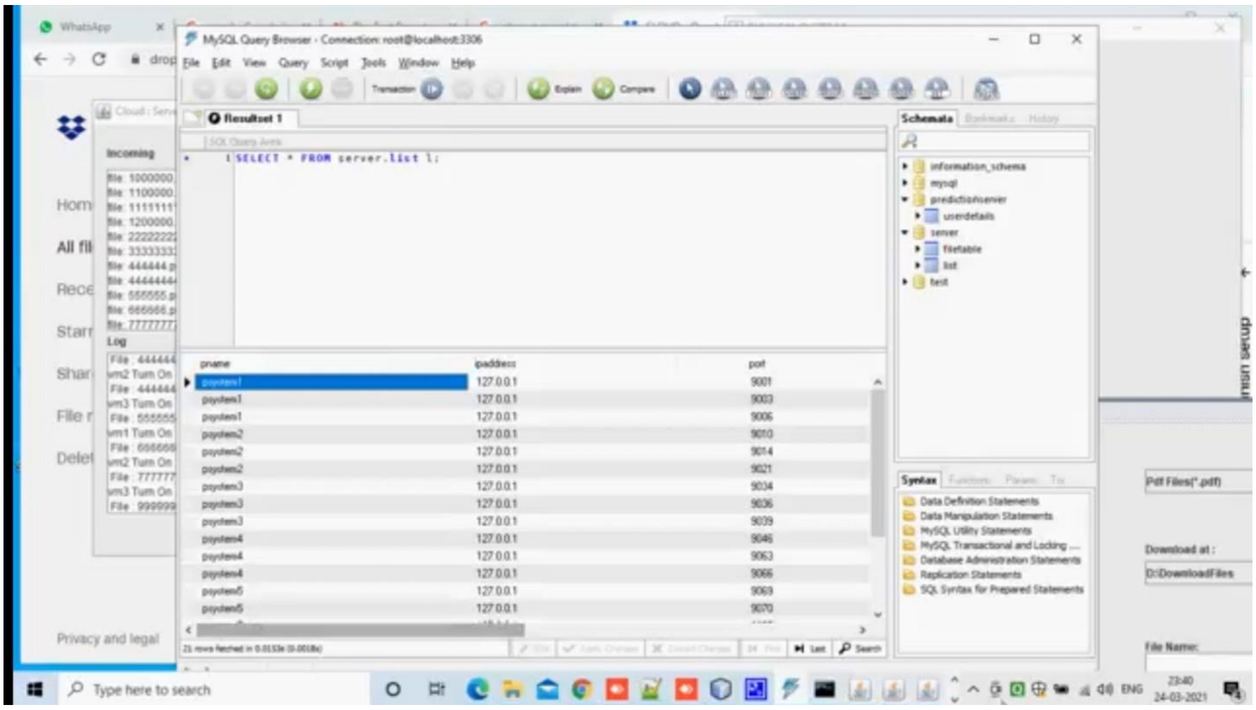
Available File Formats
Text Files(*.txt)
Image Files(*.jpg,*.gif)
MsWord Files(*.doc)
Pdf Files(*.pdf)
Other Files(*.*)

Download at : D:\

File Name:

Files List :
22222222.pdf
333333333.pdf
444444.pdf

(8) Migration Process

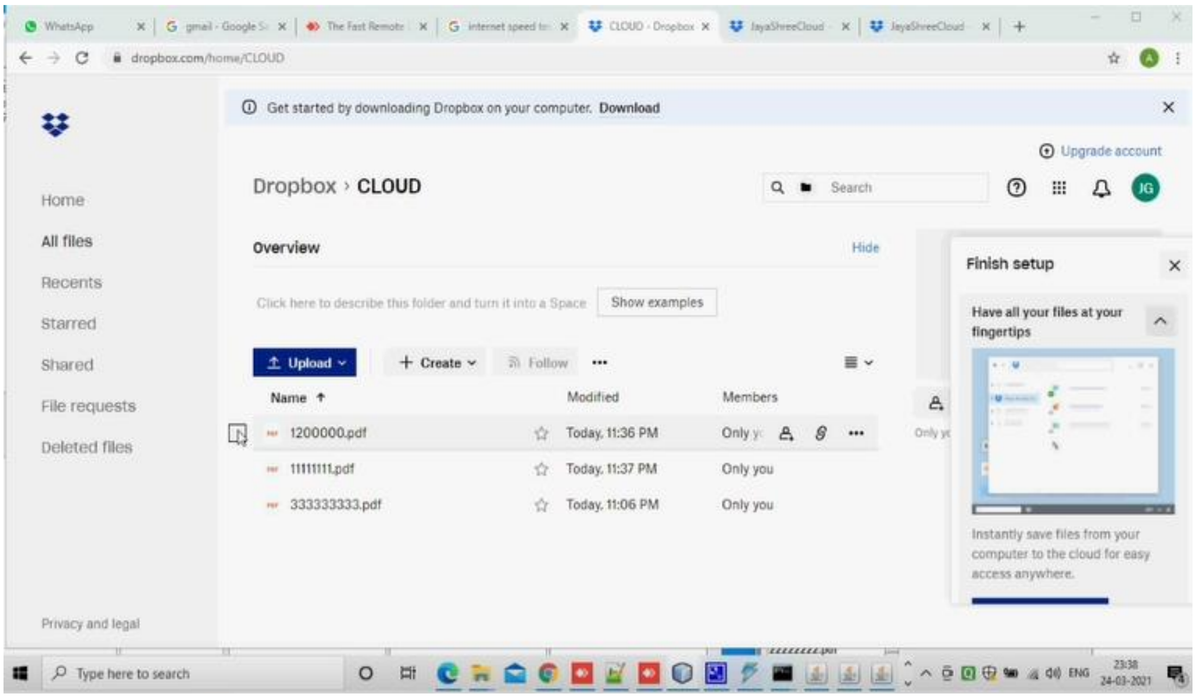


```
SELECT * FROM server.list l;
```

pname	ipaddress	port
psystem1	127.0.0.1	9001
psystem1	127.0.0.1	9003
psystem1	127.0.0.1	9006
psystem2	127.0.0.1	9010
psystem2	127.0.0.1	9014
psystem2	127.0.0.1	9021
psystem3	127.0.0.1	9034
psystem3	127.0.0.1	9036
psystem3	127.0.0.1	9039
psystem4	127.0.0.1	9046
psystem4	127.0.0.1	9063
psystem4	127.0.0.1	9066
psystem5	127.0.0.1	9069
psystem5	127.0.0.1	9070

```
21 rows fetched in 0.0153s (0.0018s)
```

(9) Testing Status


dropbox.com/home/CLOUD
Get started by downloading Dropbox on your computer. Download
Home
All files
Recents
Starred
Shared
File requests
Deleted files
Privacy and legal
Upgrade account
Dropbox › CLOUD
Search
Overview
Hide
Click here to describe this folder and turn it into a Space
Show examples
Upload
Create
Follow
Finish setup
Have all your files at your fingertips
Instantly save files from your computer to the cloud for easy access anywhere.
Type here to search

Name	Modified	Members
1200000.pdf	Today, 11:36 PM	Only y
11111111.pdf	Today, 11:37 PM	Only you
333333333.pdf	Today, 11:06 PM	Only you

(10) Dropbox

A.2 PUBLICATION:

M G JAYASHRI, E KAVITHA, R AMRUTHA, KAVITHA SUBRAMANI “Dynamic resource allocation and migration for effective cloud utilization”, **International Journal of Emerging Technologies and Research (Volume 8| issue 4| April 2021).**

Available at: https://www.jetir.org/view?paper=JETIRES06014

REFERENCES:

[1]Mahyar Movahed Nejad; Lena Mashayekhy; Daniel Grosu, “Truthful Greedy Mechanisms for Dynamic Virtual Machine Provisioning and Allocation in Clouds”,IEEE Transactions on Parallel and Distributed Systems (Volume: 26, Issue: 2, Feb. 2015).

[2] Lena Mashayekhy; Mahyar Movahed Nejad; Daniel Grosu, “A PTAS Mechanism for Provisioning and Allocation of Heterogeneous Cloud Resources”, IEEE Transactions on Parallel and Distributed Systems (Volume: 26, Issue: 9, Sept. 1 2015).

[3] Weijie Shi; Linquan Zhang; Chuan Wu; Zongpeng Li; Francis C. M. Lau, “An Online Auction Framework for Dynamic Resource Provisioning in Cloud Computing”, IEEE/ACM Transactions on Networking (Volume: 24, Issue: 4, Aug. 2016).

[4] Wei Wang; Ben Liang; Baochun Li, “Multi-Resource Fair Allocation in Heterogeneous Cloud Computing System”, IEEE Transactions on Parallel and Distributed Systems (Volume: 26, Issue: 10, Oct. 1 2015).

[5] Ming Mao; Jie Li; Marty Humphrey, “Cloud auto-scaling with deadline and budget constraints”, 2010 11th IEEE/ACM International Conference on Grid Computing.

[6] W.-T. Su and S.-M. Wu, “Node capability aware resource provisioning in a heterogeneous cloud,” in 2012 1st IEEE International Conference on Communications in China (ICCC). IEEE, 2012, pp. 46–50.

[7] G. Le, K. Xu, and J. Song, “Dynamic resource provisioning and scheduling with deadline constraint in elastic cloud,” in 2013 International Conference on Service Sciences (ICSS). IEEE, 2013, pp. 113–117.

[8] Y. Guo, P. Lama, J. Rao, and X. Zhou, “V-cache: Towards flexible resource provisioning for multi-tier applications in IaaS clouds,” in Parallel & Distributed Processing (IPDPS), 2013 IEEE 27th International Symposium on. IEEE, 2013, pp. 88–99.

[9] C. S. Pawar and R. B. Wagh, “Priority based dynamic resource allocation in cloud computing,” in Cloud and Services Computing (ISCOS), 2012 International Symposium on. IEEE, 2012, pp. 1–6.

[10] S. Zaman and D. Grosu, “Combinatorial auction-based mechanisms for VM provisioning and allocation in clouds,” in Cluster, Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM International Symposium on. IEEE, 2012, pp. 729–734.