**Major Project Report**

**On**

**Best Peer++: A Peer-to –Peer Based Large- Scale Data Processing Platform**

Submitted to the Faculty of Engineering of

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY**

**HYDERABAD**

In partial fulfillment of the requirements of the award of Degree of

**Bachelor of Technology**

In

**COMPUTER SCIENCE AND ENGINEERING**

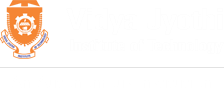
By

M RISHIKESH REDDY (13911A0599)

G AKHIL REDDY (13911A0578)

J SAI TEJA (13911A0585)

J PRANAY BHASKAR (13911A0586)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEE**

**Vidya Jyothi Institute of Technology**

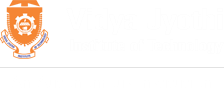
**(Approved by AICTE, Affiliated to JNTUH, HYDERABAD)**

**VIDYA JYOTHI INSTITUTE OF TECHNOLOGY**

**(Approved by AICTE, Affiliated to JNTUH, HYDERABAD)**

**Aziz Nagar Gate, C.B. Post, Hyderabad**

**Department Of Computer Science and Engineering**

****

**Certificate**

This is to certify that the Seminar Report entitled **“ Best Peer++: A Peer-to –Peer Based Large- Scale Data Processing Platform ”** a bonafide record of work carried out by M.RISHIKESH REDDY (13911A0599), G.AKHIL REDDY (13911A0578), J.SAITEJA (13911A0585), J.PRANAY BHASKAR (13911A0586) in partial fulfillment of the requirement for the award of the degree of Bachelor of Technology in **Computer Science and Engineering** of **Jawaharlal Nehru Technological University, Hyderabad** during the academic year 2015-2016.

**PRJOECT GUIDE HEAD OF THE DEPARTEMENT**

*Y. PRAVEEN KUMAR MR.VIJAY KUMAR*

*ASSISTANT PROFESSOR, PROFESSOR,*

*DEPARTEMENT OF CSE. DEPARTEMENT OF CSE*

Acknowledgement

We take this opportunity to express our/my profound sincere gratitude to all those who helped us/me to carry out this project successfully.

We owe our sincere gratitude to our guide ***Y.PRAVEEN KUMAR ASSISANT PROFESSOR,* Dept. of Computer Science & Engineering,** for giving us an opportunity to do this project.

We are highly indebted to **Mr.** *VIJAY KUMAR* **Head of the Department, Dept. of Computer Science & Engineering.** We would like to convey our sincere thanks for providing us/me with all needed inputs, and all necessary facilities and helpful suggestions. We also thank for his esteemed guidance encouragement and valuable advices given to us/me in all aspects of the project

Finally, we thank all the respective faculty of **Computer Science & Engineering,** our parents and friends who helped us, either directly or indirectly for the completion of the project work.

With Sincere Regards

M RISHIKESH REDDY (13911A0599),

G AKHIL REDDY (13911A0578),

J SAI TEJA (13911A0585),

J PRANAY BHASKAR (13911A0586),

Abstract

* The corporate network is often used for sharing information among the participating companies and facilitating collaboration in a certain industry sector where companies share a common interest.
* It can effectively help the companies to reduce their operational costs and increase the revenues. However, the inter-company data sharing and processing poses unique challenges to such a data management system including scalability, performance, throughput, and security.
* By integrating cloud computing, database, and P2P technologies into one system, Best Peer++provides an economical, flexible and scalable platform for corporate network applications and delivers data sharing services to participants based on the widely accepted pay-as-you-go business model.
* We evaluate Best Peer++ on Amazon EC2 Cloud platform. The benchmarking results show that Best Peer++ outperforms HadoopDB, a recently proposed large-scale data processing system, in performance when both systems are employed to handle typical corporate network workloads.
* The benchmarking results also demonstrate that Best Peer++ achieves near linear scalability for throughput with respect to the number of peer node

CONTENTS

1. INTRODUCTION 1

* Objective 1
* Existing System 1
* Proposed System 2

2. REQUIRMENTS SPECIFICATION 3

2.1. Introduction 3

2.2 .Functional Requirements 3

2.3. Non-functional Requirements 5

2.4.1 Software Requirements 6

Hardware Requirements 6

**3. LITERATURE SURVEY 8**

4. SYSTEM ANALYSIS 9

4.1 Introduction 9

4.2 Modules 9

4.2.1 Admin 10

Customer 11

5. **SYSTEM DESIGN 12**

5.1 Introduction 12

5.2 UML Diagrams

5.2.1 Use Case Diagram

5.2.2 Class Diagram

5.2.3 Sequence Diagram

5.2.4 Collaboration Diagram

5.2.5 Activity Diagram

5.2.6 Component Diagram

**6. IMPLEMETATION**

**7. TESTING 54**

7.1 Introduction 54

7.2 Levels of Testing 54

7.3 Maintenance and Implementation 54

7.4 Criteria Satisfied by Test Cases 55

* **SCREEN SHOTS 57**
* **CONCLUSION 61**

**BIBLIOGRAPHY**

**FUTURE ENHANCEMENTS**

**1. Introduction**

1.1 Objective

* *We present Best Peer++, a system which delivers elastic data sharing services for corporate network applications in the cloud based on Best Peer—a peer-to-peer (P2P) based data management platform.*
* *We evaluate Best Peer++ on Amazon EC2 Cloud platform.*
* *The benchmarking results show that Best Peer++ outperforms HadoopDB, a recently proposed large-scale data processing system, in performance when both systems are employed to handle typical corporate network workloads.*

**1.2 SYSTEM ANALYSIS**

**Systems analysis** is a problem-solving method that involves looking at the wider system, breaking apart the parts and figuring out how it works in order to achieve a particular goal. But before we get into detail about how that works, we should probably first answer the question: what is a system?

A **system** is in general a set of parts, steps, or components that are connected to form a more complex whole. For example, a computer system contains processors, memory, electrical pathways, a power supply, etc. For a very different example, a business is a system made up of methods, procedures, and routines.

1.2.1 Existing System

* *Such a warehousing solution has some deficiencies in real deployment.*
* *First, the corporate network needs to scale up to support thousands of participants, while the installation of a large-scale centralized data warehouse system entails nontrivial costs including huge hardware/software investments and high maintenance cost .*
* *In the real world, most companies are not keen to invest heavily on additional information systems until they can clearly see the potential return on investment (ROI).*
* *Second, companies want to fully customize the access control policy to determine which business partners can see which part of their shared data.*

Disadvantages:

* *Most of the data warehouse solutions fail to offer such flexibilities.*
* *Solution has not been designed to handle such dynamicity.*

1.2.2 Proposed System

* *Our system can efficiently handle typical workloads in a corporate network and can deliver near linear query throughput as the number of normal peers grows.*
* *Best Peer++ adopts the pay-as-you-go business model popularized by cloud computing.*
* *The total cost of ownership is therefore substantially reduced since companies do not have to buy any hardware/software in advance.*
* *Best Peer++ extends the role-based access control for the inherent distributed environment of corporate networks*

Advantages:

* *Best peer++ employs P2P technology to retrieve data between business partners.*
* *Best Peer++ is a promising solution for efficient data sharing within corporate networks.*
* *A test packet generation algorithm.*
* *A fault localization algorithm to isolate faulty devices and rules.*
* *ATPG (Automatic Test Pattern Generation and Automatic Test Pattern Generator) use cases for functional and performance testing.*
* *Evaluation of a prototype ATPG system using rule sets collected from the Stanford and Internet2 backbones.*

2. REQUIREMENTS SPECIFICATION

**2.1 Introduction:**

Software Requirements Specification plays an important role in creating quality software solutions. Specification is basically a representation process. Requirements are represented in a manner that ultimately leads to successful software implementation.

Requirements may be specified in a variety of ways. However there are some guidelines worth following: -

• Representation format and content should be relevant to the problem

• Information contained within the specification should be nested

• Diagrams and other notational forms should be restricted in number and consistent in use.

• Representations should be revisable.

The software requirements specification is produced at the culmination of the analysis task. The function and performance allocated to the software as a part of system engineering are refined by establishing a complete information description, a detailed functional and behavioral description, and indication of performance requirements and Design constraints, appropriate validation criteria and other data pertinent to requirements.

*2.2 SOFTWARE REQUIREMENTS:*

* *Operating system : Windows XP/7.*
* *Coding Language : JAVA/J2EE*
* *IDE : Net beans 7.4*
* *Database : MYSQL*

*2.3HARDWARE REQUIREMENTS*

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

**3. LITERATURE SURVEY**

**1 “A Comparative Analysis of Methodologies for Database Schema Integration,”**

**AUTHORS:** C. Batini, M. Lenzerini, and S. Navathe

One of the fundamental principles of the database approach is that a database allows a no redundant, unified representation of all data managed in an organization. This is achieved only when methodologies are available to support integration across organizational and application boundaries. Methodologies for database design usually perform the design activity by separately producing several schemas, representing parts of the application, which are subsequently merged. Database schema integration is the activity of integrating the schemas of existing or proposed databases into a global, unified schema. The aim of the paper is to provide first a unifying framework for the problem of schema integration, then a comparative review of the work done thus far in this area. Such a framework, with the associated analysis of the existing approaches, provides a basis for identifying strengths and weaknesses of individual methodologies, as well as general guidelines for future improvements and extensions.

2. “**A Comparative Analysis of Methodologies for Database Schema Integration**,”

**AUTHORS:** C. Batini, M. Lenzerini, and S. Navathe

One of the fundamental principles of the database approach is that a database allows a no redundant, unified representation of all data managed in an organization. This is achieved only when methodologies are available to support integration across organizational and application boundaries. Methodologies for database design usually perform the design activity by separately producing several schemas, representing parts of the application, which are subsequently merged. Database schema integration is the activity of integrating the schemas of existing or proposed databases into a global, unified schema. The aim of the paper is to provide first a unifying framework for the problem of schema integration, then a comparative review of the work done thus far in this area. Such a framework, with the associated analysis of the existing approaches, provides a basis for identifying strengths and weaknesses of individual methodologies, as well as general guidelines for future improvements and extensions.

3. **“Adaptive Multi-Join Query Processing in PDBMS**,”

**AUTHORS:** S. Wu, Q.H. Vu, J. Li, and K.-L. Tan

Traditionally, distributed databases assume that the (small) set of nodes participating in a query is known apriori, the data is well placed, and the statistics are readily available. However, these assumptions are no longer valid in a peer-based database management system (PDBMS). As such, it is a challenge to process and optimize queries in a PDBMS. In this paper, we present our distributed solution to this problem for multi-way join queries. Our approach first processes a multi-way join query based on an initial query evaluation plan (generated using statistical data that may be obsolete or inaccurate); as the query is being processed, statistics obtained on-the-fly are used to (continuously) refine the current plan dynamically into a more effective one. We have conducted an extensive performance study which shows that our adaptive query processing strategy can reduce the network traffic significantly.

4**. “Data Sharing in the Hyperion Peer Database System,”**

**AUTHORS:** P. Rodrıguez-Gianolli, M. Garzetti**,**

This demo presents Hyperion, a prototype system that supports data sharing for a network of independent Peer Relational Database Management Systems (PDBMSs). The nodes of such a network are assumed to be autonomous PDBMSs that form acquaintances at run-time, and manage mapping tables to define value correspondences among different databases. They also use distributed Event-Condition-Action (ECA) rules to enable and coordinate data sharing. Peers perform local querying and update processing, and also propagate queries and updates to their acquainted peers. The demo illustrates the following key functionalities of Hyperion: (1) the use of (data level) mapping tables to infer new metadata as peers dynamically join the network, (2) the ability to answer queries using data in acquaintances, and (3) the ability to coordinate peers through update propagation.

**4. SYSTEM ANALYSIS**

**MODULES:**

* Peer++ Processing Approach
* Parallel P2P Processing
* Implementing Map Reduce
* Adaptive Query Processing

**MODULES DESCRIPTION:**

**4.1 Peer++ Processing Approach:**

* Best Peer++ employs two query processing approaches: basic processing and adaptive processing. The basic query processing strategy is similar to the one adopted in the distributed databases domain.
* Overall, the query submit-ted to a normal peer P is evaluated in two steps: fetching and processing. In the fetching step, the query is decomposed into a set of sub-queries which are then sent to the remote normal peers that host the data involved in the query (the list of these normal peers is determined by searching the indices stored in BATON).
* The sub query is then processed by each remote normal peer and the intermediate results are shuffled to the query submitting peer P. In the processing step, the normal peer P first collects all the required data from the other participating normal peers
* . To reduce I/O, the peer P creates a set of Mem Tables to hold the data retrieved from other peers and bulk inserts these data into the local MySQL when the Mem Table is full. After receiving all the necessary data, the peer P finally evaluates the submitted query.

**4.2 Parallel P2P Processing:**

* For each join, instead of forwarding all tuples into a single processing node, we disseminate them into a set of nodes, which will process the join in parallel. We adopt the conventional replicated join approach.
* Namely, the small table will be replicated to all processing nodes and joined with a partition of the large table.

**4.3 Implementing Map Reduce:**

* The main difference between Map Reduce method and native P2P method comes from the join processing. In Map Reduce method, instead of doing replicate joins, the symmetric-hash join approach is adopted.
* Each mapper reads in its local data and shuffles the intermediate tuple according to the hash value of the join key. Therefore, each tuple only needs to be shuffled once on each level.
* Note that the configuration and launch of a Map Reduce job also incurs certain overhead, which, can be measured in the runtime, is a constant value.

**4.4 Adaptive Query Processing:**

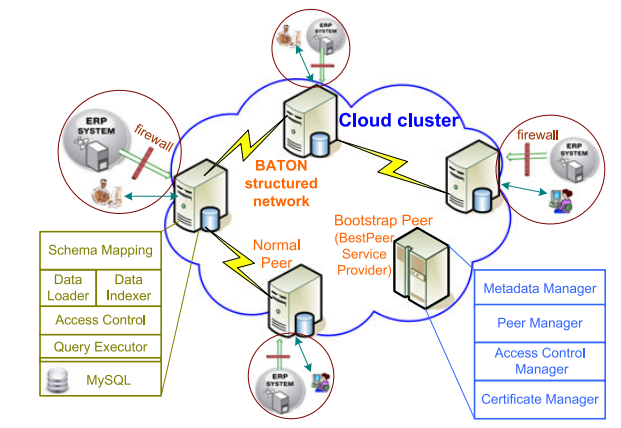
* For small jobs, the P2P engine performs better than the Map Reduce engine, as it does not incur initialization cost and database join algorithms have been well optimized.
* However, for large-scale data analytic jobs, the Map Reduce engine is more scalable, as it does not incur recursive data replications.
* Based on the above-mentioned cost models, we propose our adaptive query processing approach.
* When a query is submitted, the query planner retrieves related histogram and index information from the bootstrap node, analyzes the query and constructs a processing graph for the query then the costs of both the P2P engine and Map Reduce engine are predicted based on the histograms and runtime parameters of the cost models.
* The query planner compares the costs between two methods and executes the one with lower cost.

5. SYSTEM DESIGN

**5.1 Introduction**

* The most creative and challenging phase of the life cycle is system design. The term design describes a final system and the process by which it is developed. It refers to the technical specifications that will be applied in implementations of the candidate system. The design may be defined as “the process of applying various techniques and principles for the purpose of defining a device, a process or a system with sufficient details to permit its physical realization”.
* The designer’s goal is how the output is to be produced and in what format. Samples of the output and input are also presented. Second input data and database files have to be designed to meet the requirements of the proposed output.
* The processing phases are handled through the program Construction and Testing. Finally, details related to justification of the system and an estimate of the impact of the candidate system on the user and the organization are documented and evaluated by management as a step toward implementation.
* The importance of software design can be stated in a single word “Quality*”*. Design provides us with representations of software that can be assessed for quality.
* Design is the only way where we can accurately translate a customer’s requirements into a complete software product or system. Without design we risk building an unstable system that might fail if small changes are made.
* It may as well be difficult to test, or could be one who’s quality can’t be tested. So it is an essential phase in the development of a software product.

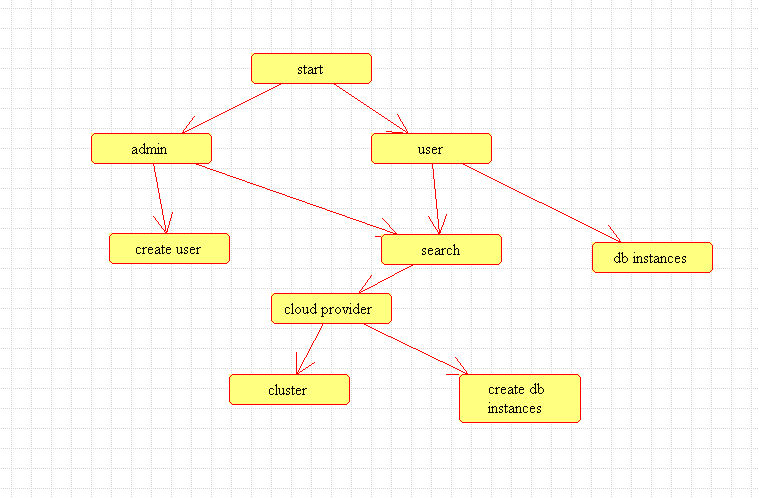
*5.2 SYSTEM ARCHITECTURE:*



**5.3 UML Diagrams:**

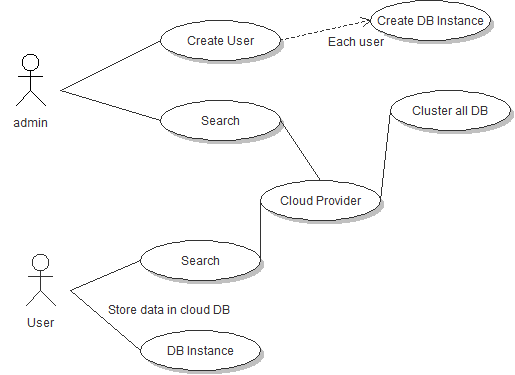
* UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.
* The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.
* The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.
* The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

**5.3.1 DATA FLOW DIAGRAM:**



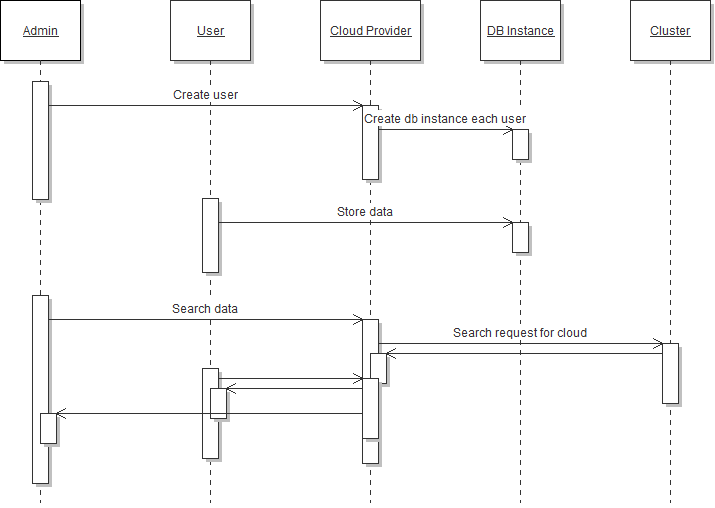
**5.3 .2 USE CASE DIAGRAM:**

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



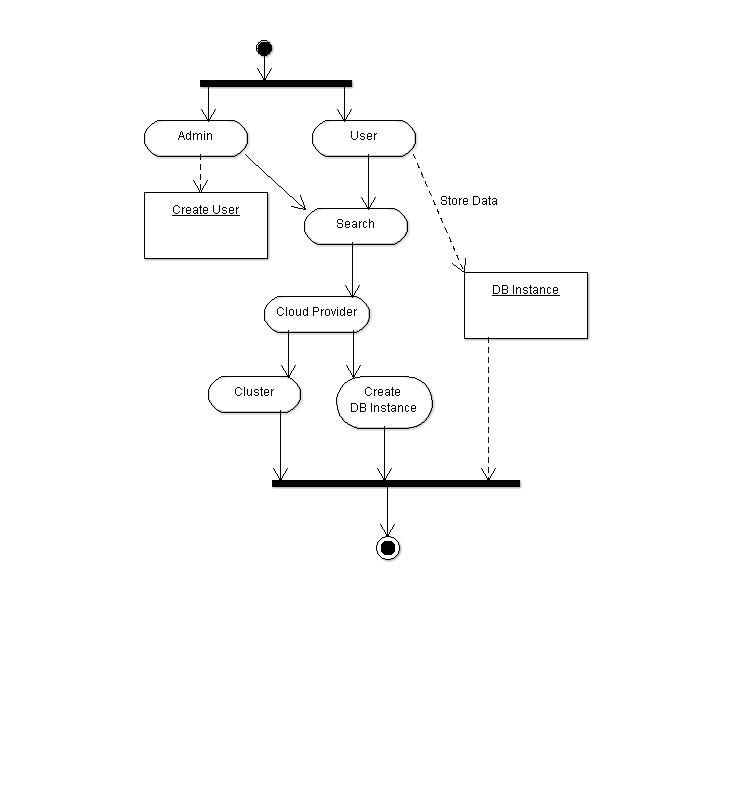
**5.3. 3 SEQUENCE DIAGRAM:**

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



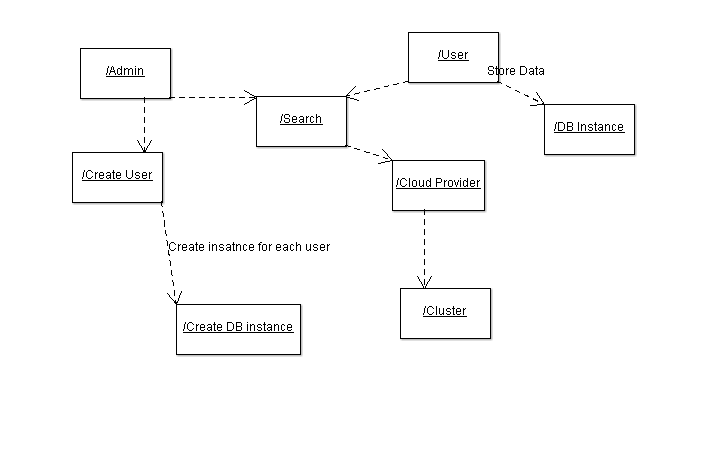
**5.3.4 ACTIVITY DIAGRAM:**

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



**5.3.5 COLLOBRATION DIAGRAM:**

* A collaboration diagram, also called a communication diagram or interaction diagram, is an illustration of the relationships and interactions among [software](http://searchsoa.techtarget.com/definition/software) [object](http://searchsoa.techtarget.com/definition/object)s in the Unified Modeling Language (UML).
* A collaboration diagram resembles a [flowchart](http://whatis.techtarget.com/definition/flowchart) that portrays the roles, functionality and behavior of individual objects as well as the overall operation of the system in [real time](http://searchcio-midmarket.techtarget.com/definition/real-time). Objects are shown as rectangles with naming labels inside. ‘
* These labels are preceded by colons and may be underlined. The relationships between the objects are shown as lines connecting the rectangles.
* The [message](http://whatis.techtarget.com/definition/message)s between objects are shown as arrows connecting the relevant rectangles along with labels that define the message



**6. SYSTEM STUDY**

**6.1 FEASIBILITY STUDY**

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

Three key considerations involved in the feasibility analysis are

* ECONOMICAL FEASIBILITY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

**6.2 ECONOMICAL FEASIBILITY**

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

**6.3 TECHNICAL FEASIBILITY**

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

**6.4 SOCIAL FEASIBILITY**

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

**7. IMPLEMENTATION**

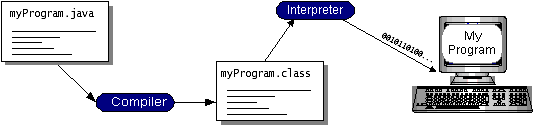
**7.1 Java Technology**

Java technology is both a programming language and a platform.

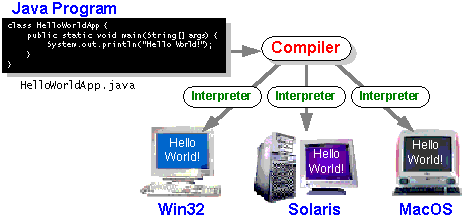
**The Java Programming Language**

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

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



* You can think of Java byte codes as the machine code instructions for the *Java Virtual Machine* (Java VM). Every Java interpreter, whether it’s a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java byte codes help make “write once, run anywhere” possible.
* You can compile your program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the Java VM. That means that as long as a computer has a Java VM, the same program written in the Java programming language can run on Windows 2000, a Solaris workstation

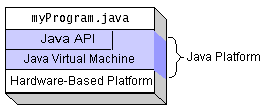


**The Java Platform**

A *platform* is the hardware or software environment in which a program runs. We’ve already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and MacOS.

The Java platform has two components:

* 1. The *Java Virtual Machine* (Java VM)
  2. The *Java Application Programming Interface* (Java API)
* You’ve already been introduced to the Java VM. It’s the base for the Java platform and is ported onto various hardware-based platforms.
* The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries of related classes and interfaces; these libraries are known as *packages*. The next section, What Can Java Technology Do? Highlights what functionality some of the packages in the Java API provide.
* The following figure depicts a program that’s running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware.

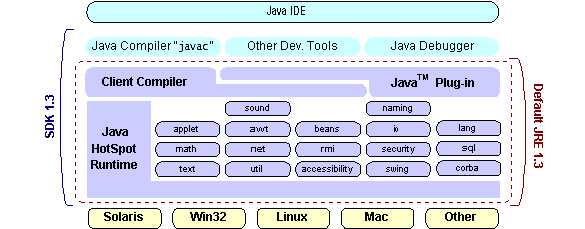


* Native code is code that after you compile it, the compiled code runs on a specific hardware platform. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time byte code compilers can bring performance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

***What Can Java Technology Do?***

Every full implementation of the Java platform gives you the following features:

* **The essentials**: Objects, strings, threads, numbers, input and output, data structures, system properties, date and time, and so on.
* **Applets**: The set of conventions used by applets.
* **Networking**: URLs, TCP (Transmission Control Protocol), UDP (User Data gram Protocol) sockets, and IP (Internet Protocol) addresses.
* **Internationalization**: Help for writing programs that can be localized for users worldwide. Programs can automatically adapt to specific locales and be displayed in the appropriate language.
* **Security**: Both low level and high level, including electronic signatures, public and private key management, access control, and certificates.
* **Software components**: Known as JavaBeansTM, can plug into existing component architectures.
* **Object serialization**: Allows lightweight persistence and communication via Remote Method Invocation (RMI).
* **Java Database Connectivity (JDBCTM)**: Provides uniform access to a wide range of relational databases.
* The Java platform also has APIs for 2D and 3D graphics, accessibility, servers, collaboration, telephony, speech, animation, and more. The following figure depicts what is included in the Java 2 SDK.



***How Will Java Technology Change My Life?***

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

* **Get started quickly**
* **Write less code**
* **Write better code**
* **Develop programs more quickly**
* **Avoid platform dependencies with 100% Pure Java**
* **Write once, run anywhere**
* **Distribute software more easily**

ODBC:

* Microsoft Open Database Connectivity (ODBC) is a standard programming interface for application developers and database systems providers. Before ODBC became a *de facto* standard for Windows programs to interface with database systems, programmers had to use proprietary languages for each database they wanted to connect to.
* Now, ODBC has made the choice of the database system almost irrelevant from a coding perspective, which is as it should be. Application developers have much more important things to worry about than the syntax that is needed to port their program from one database to another when business needs suddenly change.
* Through the ODBC Administrator in Control Panel, you can specify the particular database that is associated with a data source that an ODBC application program is written to use. Think of an ODBC data source as a door with a name on it. Each door will lead you to a particular database. For example, the data source named Sales Figures might be a SQL Server database, whereas the Accounts Payable data source could refer to an Access database. The physical database referred to by a data source can reside anywhere on the LAN.

**JDBC**

* In an effort to set an independent database standard API for Java; Sun Microsystems developed Java Database Connectivity, or JDBC. JDBC offers a generic SQL database access mechanism that provides a consistent interface to a variety of RDBMSs. This consistent interface is achieved through the use of “plug-in” database connectivity modules, or *drivers*. If a database vendor wishes to have JDBC support, he or she must provide the driver for each platform that the database and Java run on.
* To gain a wider acceptance of JDBC, Sun based JDBC’s framework on ODBC. As you discovered earlier in this chapter, ODBC has widespread support on a variety of platforms. Basing JDBC on ODBC will allow vendors to bring JDBC drivers to market much faster than developing a completely new connectivity solution.
* JDBC was announced in March of 1996. It was released for a 90 day public review that ended June 8, 1996. Because of user input, the final JDBC v1.0 specification was released soon after.

Java ha two things: a programming language and a platform. Java is a high-level programming language that is all of the following

Simple Architecture-neutral

Object-oriented Portable

Distributed High-performance

Interpreted multithreaded

Robust Dynamic

***SQL Level API:***

* The designers felt that their main goal was to define a SQL interface for Java. Although not the lowest database interface level possible, it is at a low enough level for higher-level tools and APIs to be created.
* Conversely, it is at a high enough level for application programmers to use it confidently. Attaining this goal allows for future tool vendors to “generate” JDBC code and to hide many of JDBC’s complexities from the end user.

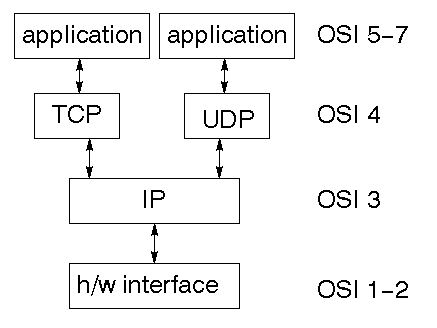
***SQL Conformance:***

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

**Networking:**

**TCP/IP stack**

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



**JavaScript and Ajax Development:**

* JavaScript is an object-oriented scripting language primarily used in client-side interfaces for web applications.
* Ajax (Asynchronous JavaScript and XML) is a Web 2.0 technique that allows changes to occur in a web page without the need to perform a page refresh. JavaScript toolkits can be leveraged to implement Ajax-enabled components and functionality in web pages.

**Web Server and Client:**

* Web Server is software that can process the client request and send the response back to the client. For example, Apache is one of the most widely used web server. Web Server runs on some physical machine and listens to client request on specific port.
* A web client is software that helps in communicating with the server. Some of the most widely used web clients are Firefox, Google Chrome, Safari etc. When we request something from server (through URL), web client takes care of creating a request and sending it to server and then parsing the server response and present it to the user.

**HTML and HTTP:**

* Web Server and Web Client are two separate softwares, so there should be some common language for communication. HTML is the common language between server and client and stands for **H**yper**T**ext **M**arkup **L**anguage.
* Web server and client needs a common communication protocol, HTTP (**H**yper**T**ext **T**ransfer **P**rotocol) is the communication protocol between server and client. HTTP runs on top of TCP/IP communication protocol.

Some of the important parts of HTTP Request are:

* **HTTP Method** – action to be performed, usually GET, POST, PUT etc.
* **URL** – Page to access
* **Form Parameters** – similar to arguments in a java method, for example user,password details from login page.

Sample HTTP Request:

|  |  |
| --- | --- |
| 1  2  3 | GET /FirstServletProject/jsps/hello.jsp HTTP/1.1  Host: localhost:8080  Cache-Control: no-cache |

Some of the important parts of HTTP Response are:

* **Status Code** – an integer to indicate whether the request was success or not. Some of the well known status codes are 200 for success, 404 for Not Found and 403 for Access Forbidden.
* **Content Type** – text, html, image, pdf etc. Also known as MIME type
* **Content** – actual data that is rendered by client and shown to user.

**MIME Type or Content Type**:

If you see above sample HTTP response header, it contains tag “Content-Type”. It’s also called MIME type and server sends it to client to let them know the kind of data it’s sending. It helps client in rendering the data for user. Some of the mostly used mime types are text/html, text/xml, application/xml etc.

**Understanding URL:**

URL is acronym of Universal Resource Locator and it’s used to locate the server and resource. Every resource on the web has it’s own unique address. Let’s see parts of URL with an example.

[**http://localhost:8080/FirstServletProject/jsps/hello.jsp**](http://localhost:8080/FirstServletProject/jsps/hello.jsp):

**http://** – This is the first part of URL and provides the communication protocol to be used in server-client communication.

**localhost** – The unique address of the server, most of the times it’s the hostname of the server that maps to unique IP address. Sometimes multiple hostnames point to same IP addresses and web server virtual host takes care of sending request to the particular server instance.

**8080** – This is the port on which server is listening, it’s optional and if we don’t provide it in URL then request goes to the default port of the protocol. Port numbers 0 to 1023 are reserved ports for well known services, for example 80 for HTTP, 443 for HTTPS, 21 for FTP etc.

**FirstServletProject/jsps/hello.jsp** – Resource requested from server. It can be static html, pdf, JSP, servlets, PHP etc.

**Why we need Servlet and JSPs?**

* Web servers are good for static contents HTML pages but they don’t know how to generate dynamic content or how to save data into databases, so we need another tool that we can use to generate dynamic content. There are several programming languages for dynamic content like PHP, Python, Ruby on Rails, Java Servlets and JSPs.
* Java Servlet and JSPs are server side technologies to extend the capability of web servers by providing support for dynamic response and data persistence.

**Web Container:**

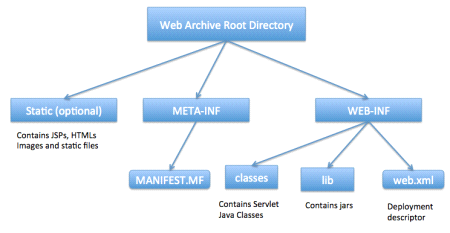
* Tomcat is a web container, when a request is made from Client to web server, it passes the request to web container and it’s web container job to find the correct resource to handle the request (servlet or JSP) and then use the response from the resource to generate the response and provide it to web server. Then web server sends the response back to the client.
* When web container gets the request and if it’s for servlet then container creates two Objects HTTPServletRequest and HTTPServletResponse. Then it finds the correct servlet based on the URL and creates a thread for the request. Then it invokes the servlet service() method and based on the HTTP method service() method invokes doGet() or doPost() methods. Servlet methods generate the dynamic page and write it to response. Once servlet thread is complete, container converts the response to HTTP response and send it back to client.

Some of the important work done by web container are:

* **Communication Support** – Container provides easy way of communication between web server and the servlets and JSPs. Because of container, we don’t need to build a server socket to listen for any request from web server, parse the request and generate response. All these important and complex tasks are done by container and all we need to focus is on our business logic for our applications.
* **Lifecycle and Resource Management** – Container takes care of managing the life cycle of servlet. Container takes care of loading the servlets into memory, initializing servlets, invoking servlet methods and destroying them. Container also provides utility like JNDI for resource pooling and management.
* **Multithreading Support** – Container creates new thread for every request to the servlet and when it’s processed the thread dies. So servlets are not initialized for each request and saves time and memory.
* **JSP Support** – JSPs doesn’t look like normal java classes and web container provides support for JSP. Every JSP in the application is compiled by container and converted to Servlet and then container manages them like other servlets.
* **Miscellaneous Task** – Web container manages the resource pool, does memory optimizations, run garbage collector, provides security configurations, support for multiple applications, hot deployment and several other tasks behind the scene that makes our life easier.

**Web Application Directory Structure**

* Java Web Applications are packaged as Web Archive (WAR) and it has a defined structure. You can export above dynamic web project as WAR file and unzip it to check the hierarchy. It will be something like below image.



**Deployment Descriptor**

**Web.xml** file is the deployment descriptor of the web application and contains mapping for servlets (prior to 3.0), welcome pages, security configurations, session timeout settings etc.

Thats all for the java web application startup tutorial, we will explore Servlets and JSPs more in future posts.

**7.2 MySQL:**

* MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation.
* The MySQL Web site (<http://www.mysql.com/>) provides the latest information about MySQL software.

**MySQL is a database management system.**

* A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network.To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server.
* Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.
* **MySQL databases are relational.**
* A relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables, views, rows, and columns, offers a flexible programming environment. The database enforces these rules, so that with a well-designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data.
* The SQL part of “MySQL” stands for “Structured Query Language”. SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax.
* SQL is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, “SQL-92” refers to the standard released in 1992, “SQL:1999” refers to the standard released in 1999, and “SQL:2003” refers to the current version of the standard. We use the phrase “the SQL standard” to mean the current version of the SQL Standard at any time.

FEATURES OF SQL:

* **MySQL software is Open Source.**
* **The MySQL Database Server is very fast, reliable, scalable, and easy to use**
* **MySQL Server works in client/server or embedded systems.**

**.**

**8. SYSTEM TESTING**

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

**8.1 TYPES OF TESTS**

**Unit testing:**

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

**Integration testing**

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

**Functional test:**

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

Functional testing is centered on the following items:

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

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

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

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

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

**System Test**

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

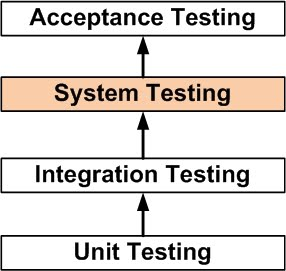
**White Box Testing**

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

**Black Box Testing**

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

8.2 SYSTEM TESTING Fundamentals



8.2.1 System Testing:

* **System Testing**is a [level of the software testing](http://softwaretestingfundamentals.com/software-testing-levels/) where a complete and integrated software is tested.
* The purpose of this test is to evaluate the system’s compliance with the specified requirements.
* **System Testing** is a level of the software **testing** where a complete and integrated software is tested.
* The purpose of this **test** is to evaluate the **system's** compliance with the specified requirements. Definition by ISTQB. **system testing**: The process of **testing** an integrated **system** to verify that it meets specified.

**8.2.2Unit Testing:**

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

***Test strategy and approach***

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

**Test objectives**

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

**Features to be tested**

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

**8.2 .3 Integration Testing**

* Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

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

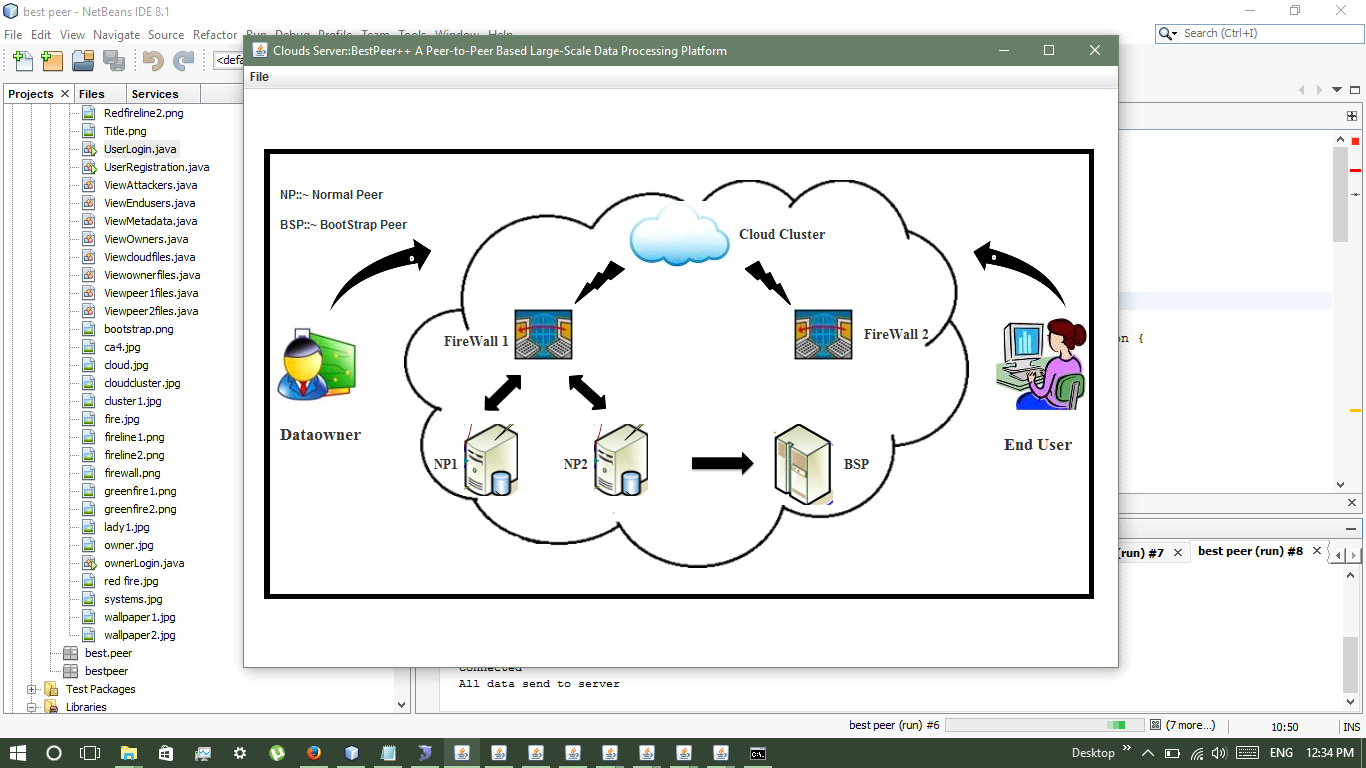
**8.2.4 Acceptance Testing**

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

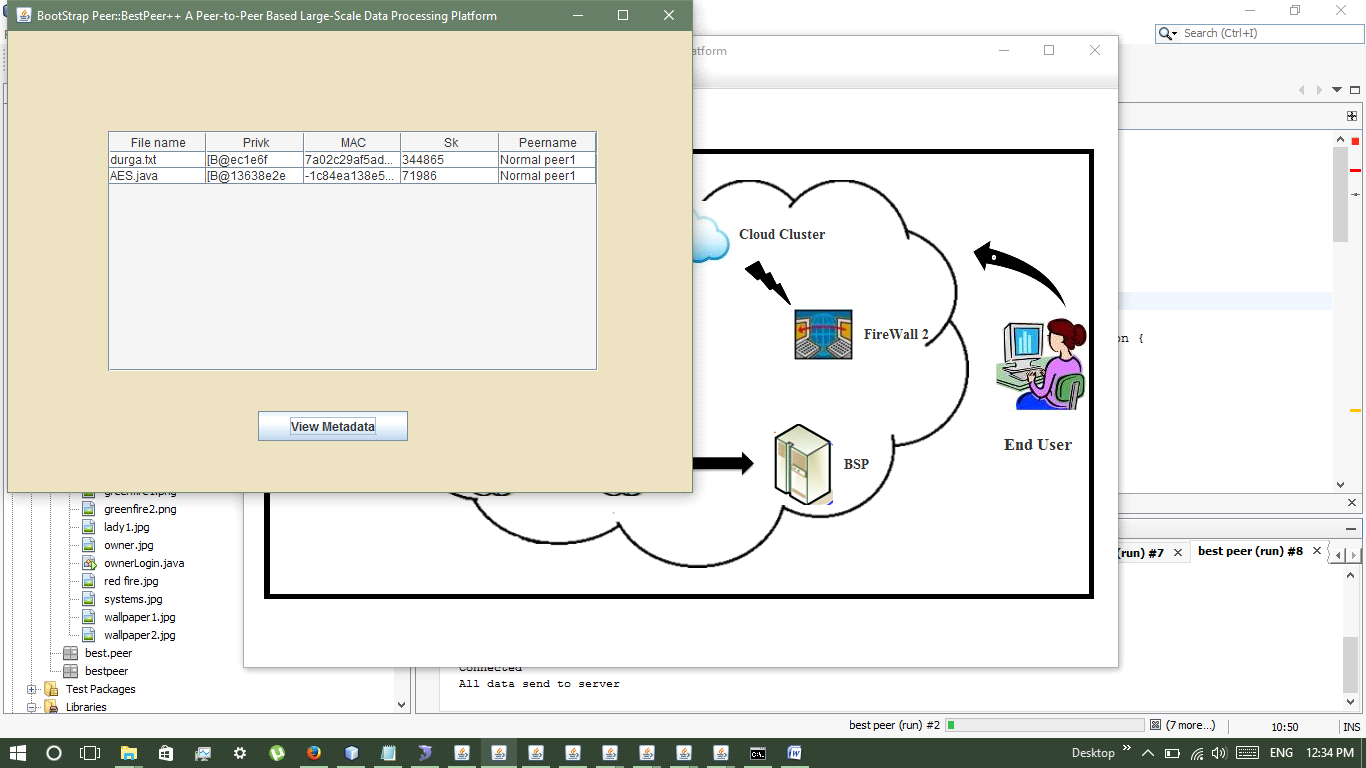
**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

9. Screenshots:

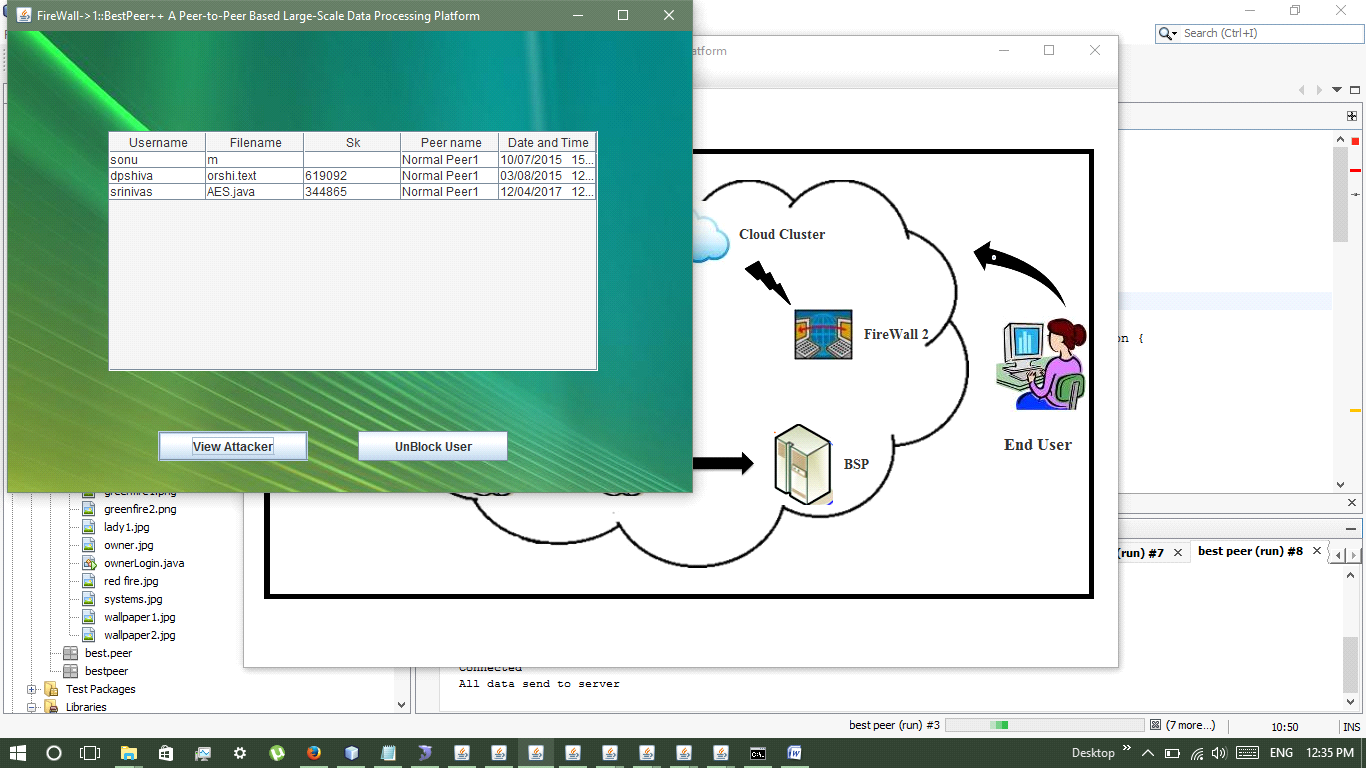
1. Clo ud.java:



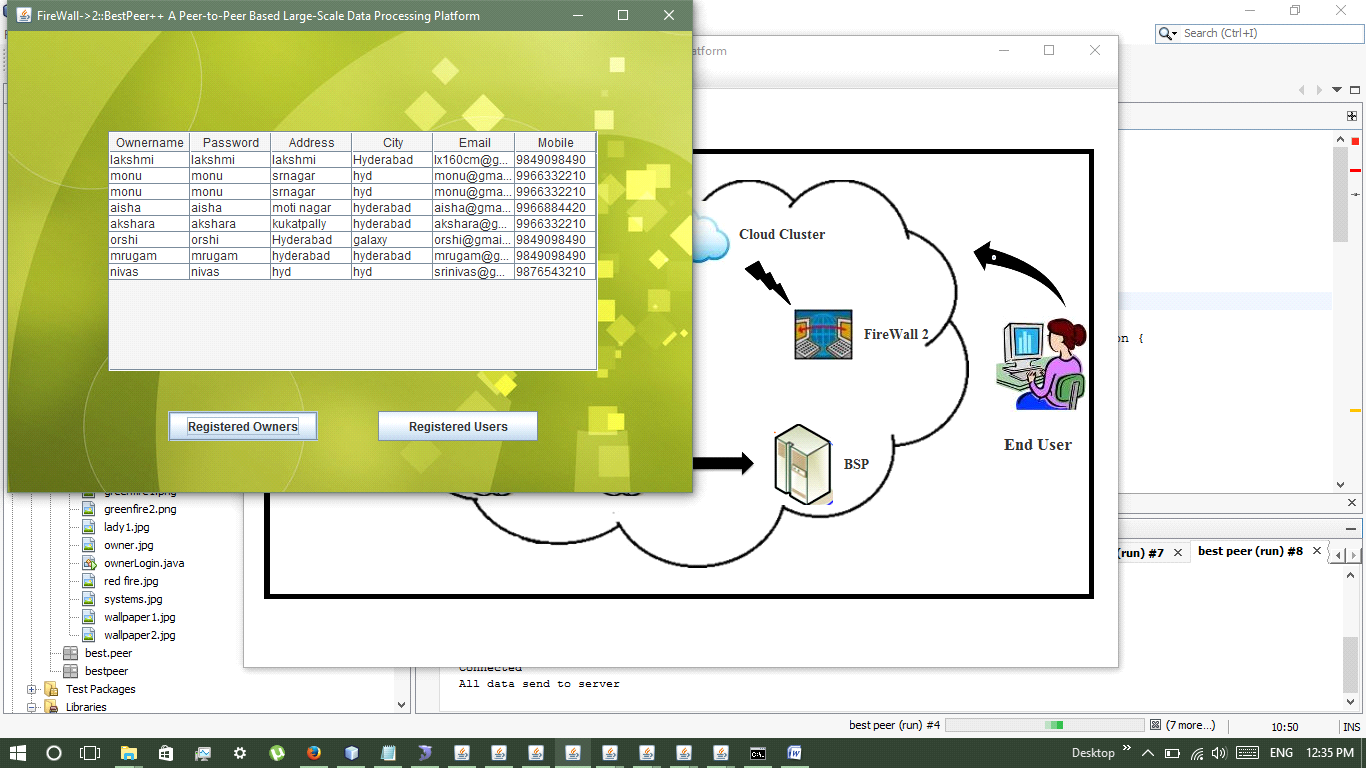
2. Booster.java:



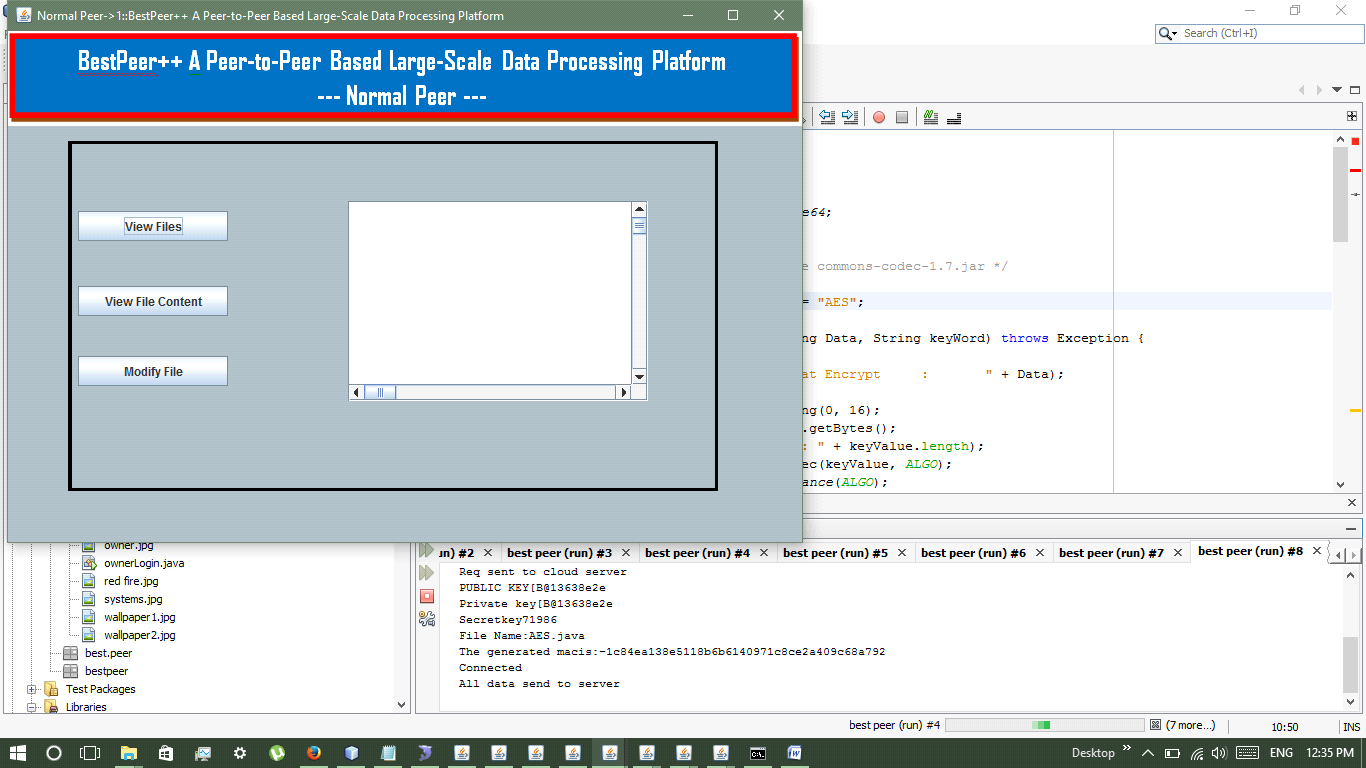
3. Firwall.java:



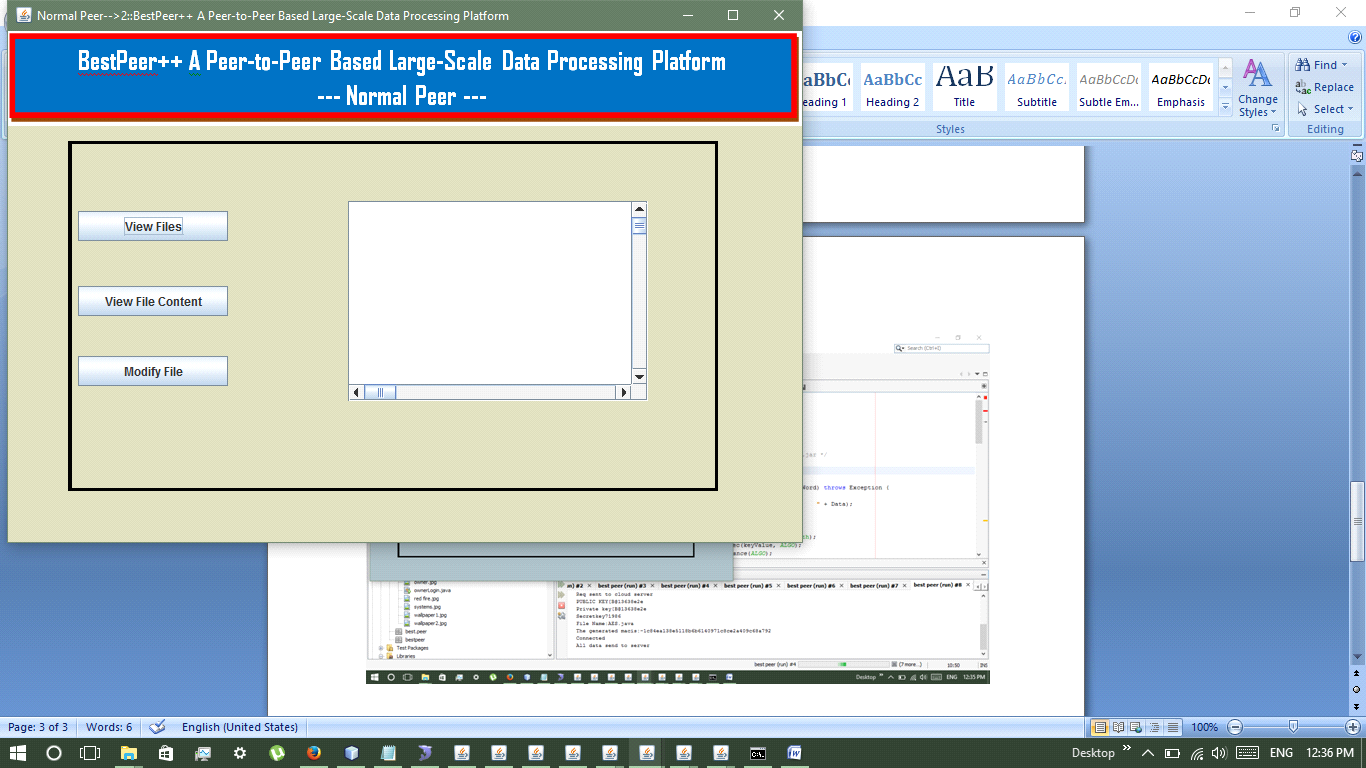
4. Firewall2.java:



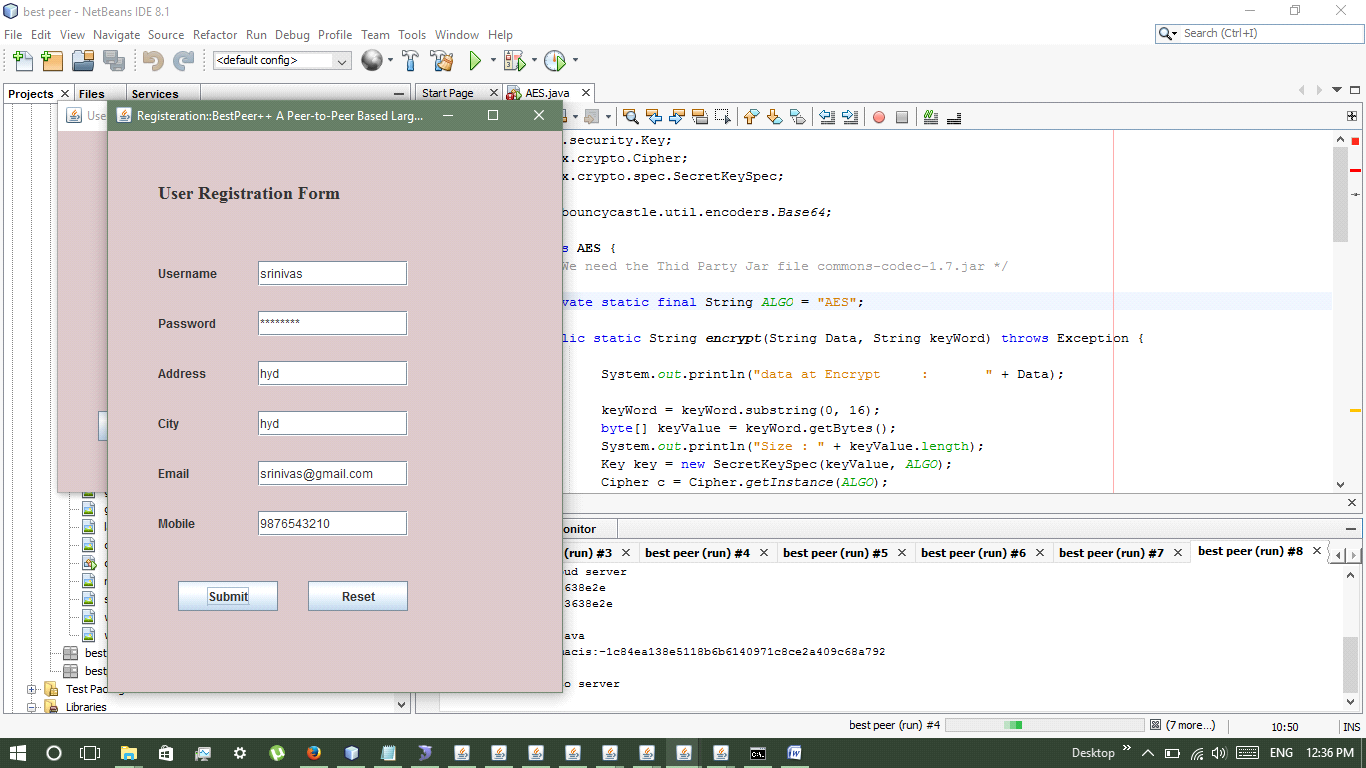
5. Normal peer 1:



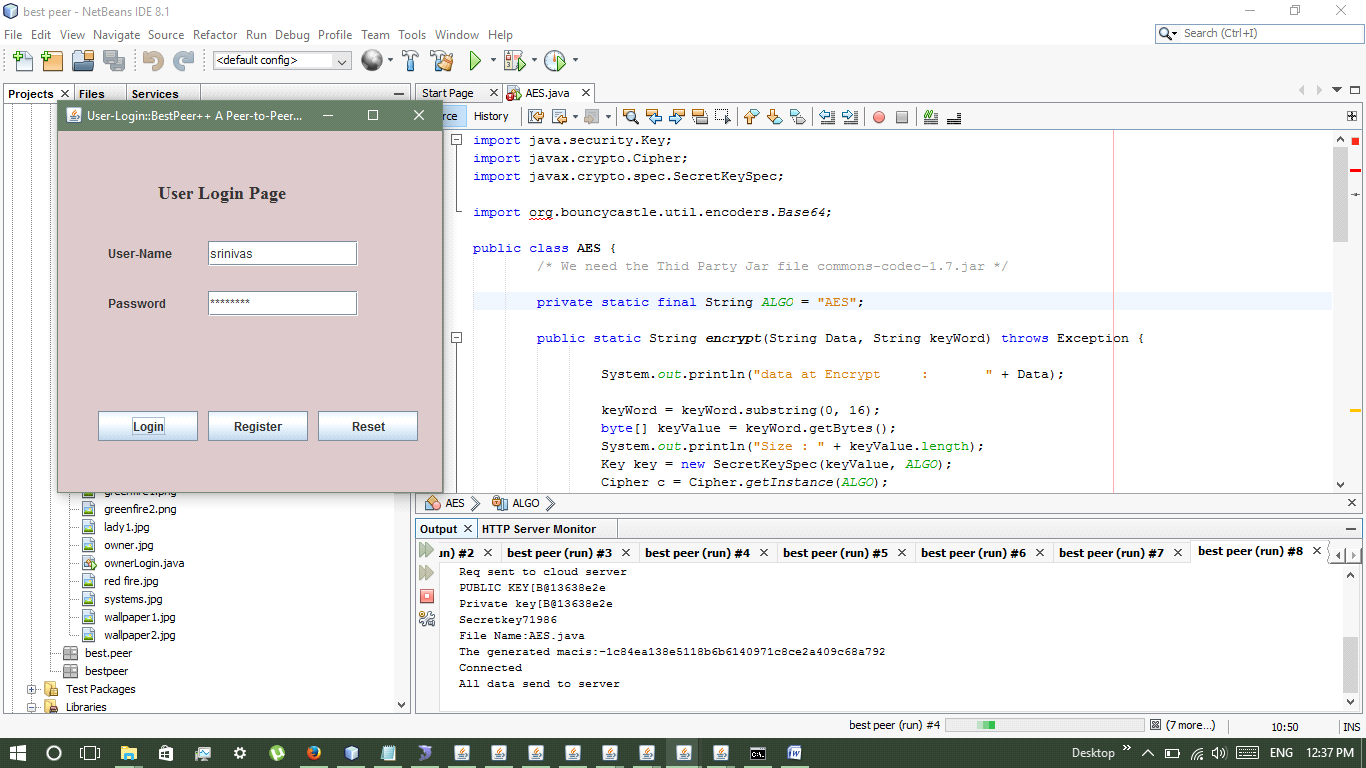
6.Normal peer 2:



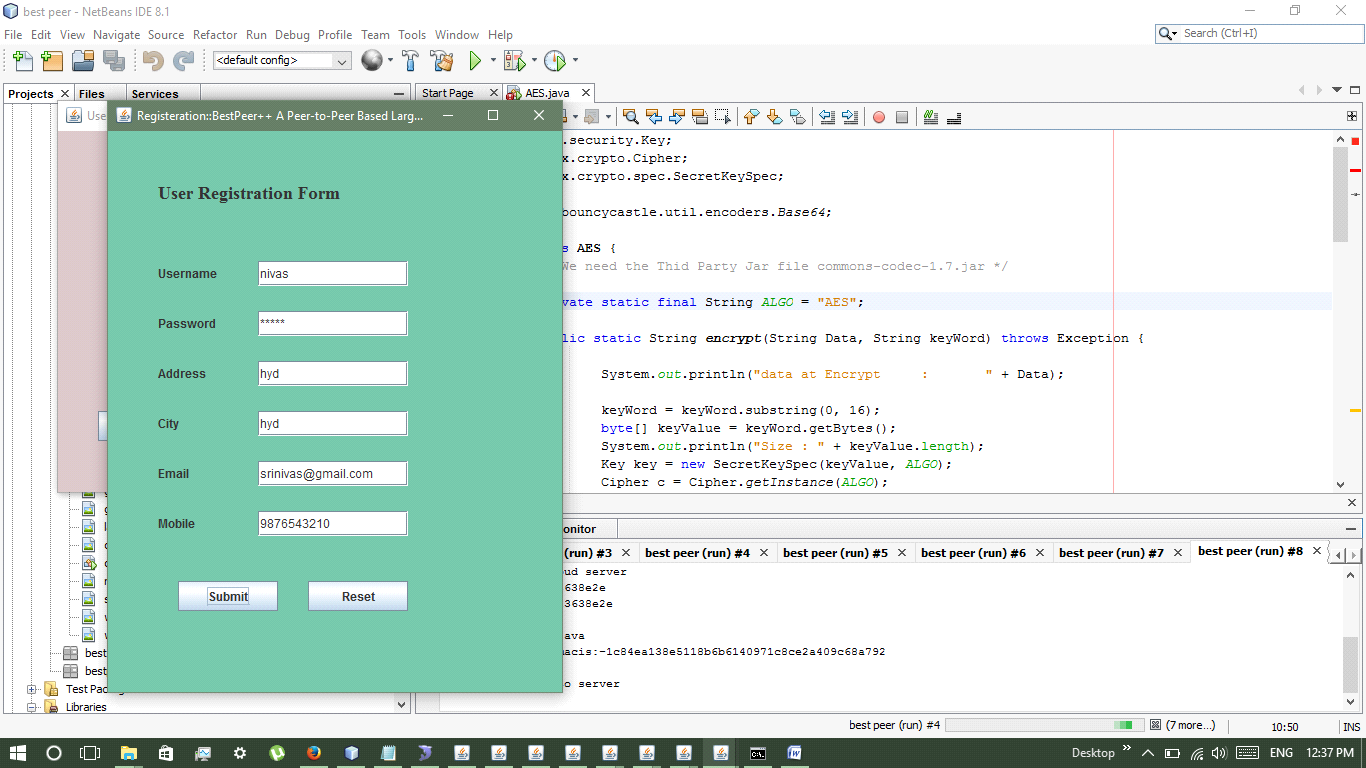
7. User register:



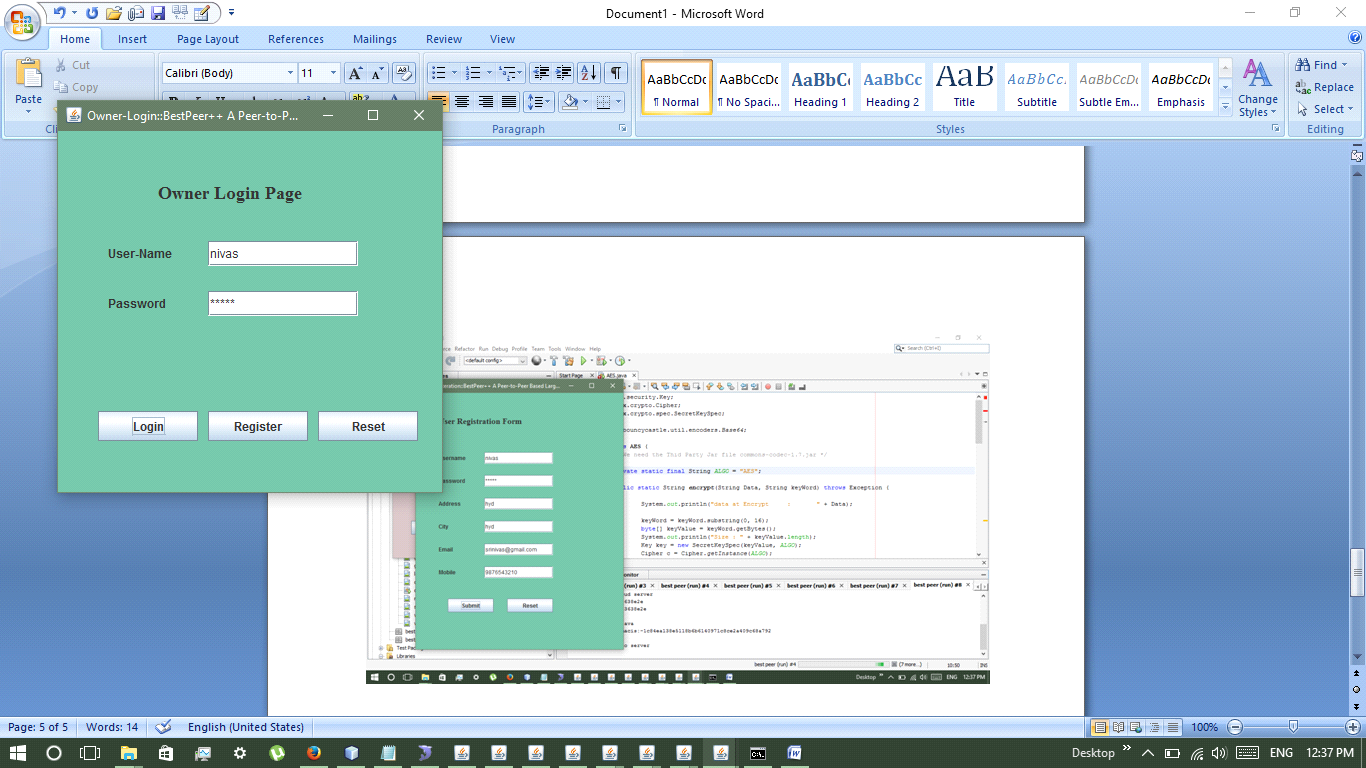
8. User login:



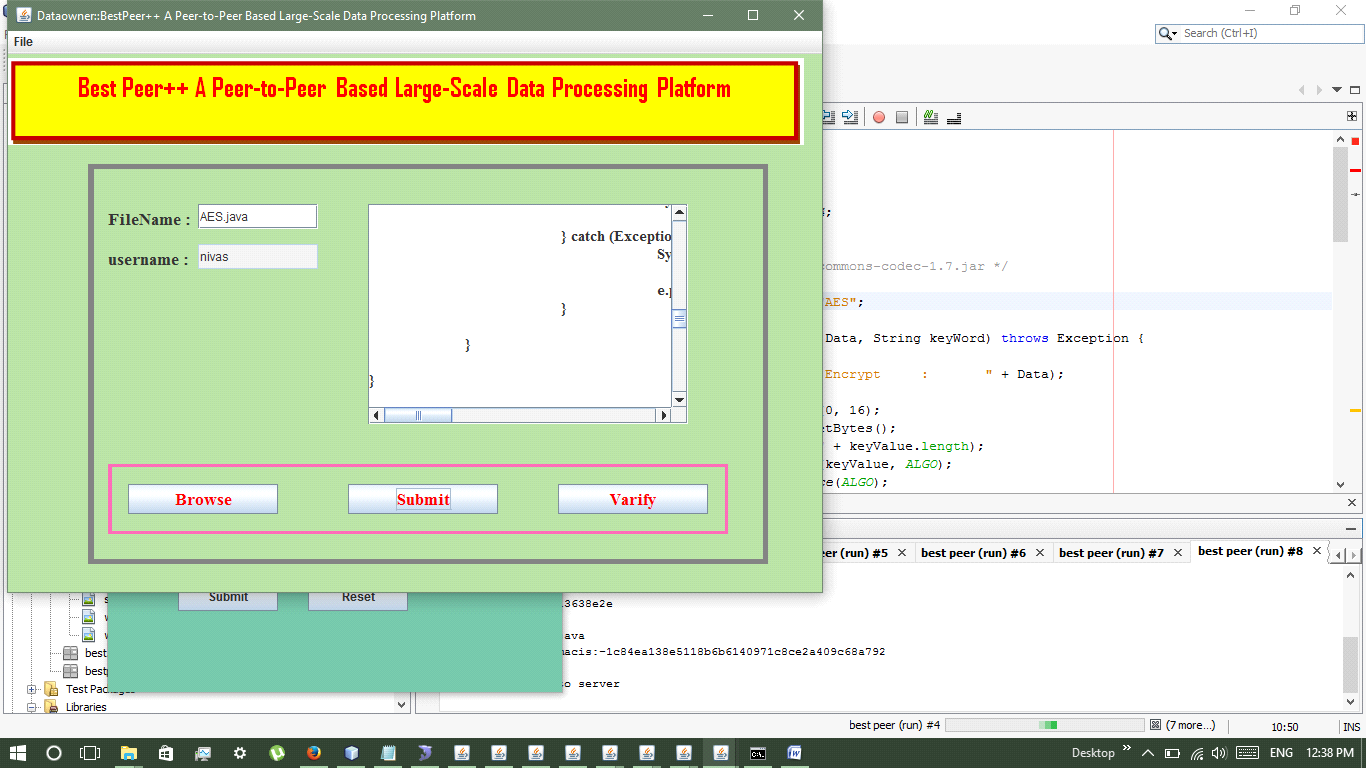
9. Owner registers:



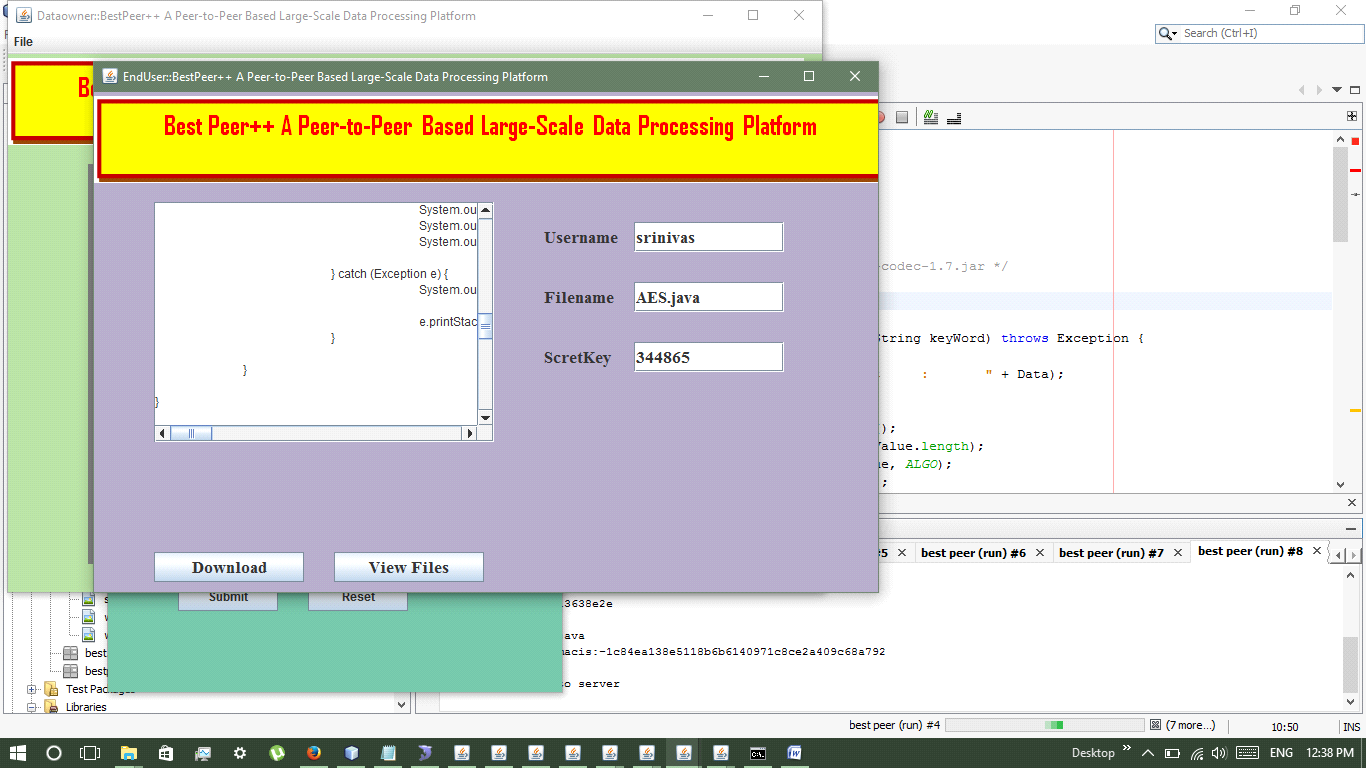
10. Owner login:



11. Upload file:



12. User downloads:



**10. BIBLIOGRAPHY**

**REFERENCES:**

[1] K. Aberer, A. Datta, and M. Hauswirth, “Route Maintenance Overheads in DHT Overlays,” in 6th Workshop Distrib. Data Struct., 2004.

[2] A. Abouzeid, K. Bajda-Pawlikowski, D.J. Abadi, A. Rasin, and A. Silberschatz, “HadoopDB: An Architectural Hybrid of MapReduce and DBMS Technologies for Analytical Workloads,” Proc. VLDB Endowment, vol. 2, no. 1, pp. 922-933, 2009.

[3] C. Batini, M. Lenzerini, and S. Navathe, “A Comparative Analysis of Methodologies for Database Schema Integration,” ACM Computing Surveys, vol. 18, no. 4, pp. 323-364, 1986.

[4] D. Bermbach and S. Tai, “Eventual Consistency: How Soon is Eventual? An Evaluation of Amazon s3’s Consistency Behavior,” in Proc. 6th Workshop Middleware Serv. Oriented Comput. (MW4SOC ’11), pp. 1:1-1:6, NY, USA, 2011.

[5] B. Cooper, A. Silberstein, E. Tam, R. Ramakrishnan, and R. Sears, “Benchmarking Cloud Serving Systems with YCSB,” Proc. First ACM Symp. Cloud Computing, pp. 143-154, 2010.

[6] H.V. Jagadish, B.C. Ooi, K.-L. Tan, C. Yu, and R. Zhang, “iDistance: An Adaptive B+-Tree Based Indexing Method for Nearest Neighbor Search,” ACM Trans. Database Systems, vol. 30, pp. 364-397, June 2005.

[6] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels, “Dynamo: Amazon’s Highly Available Key-Value Store,” Proc. 21st ACM SIGOPS Symp. Operating Systems Principles (SOSP ’07), pp. 205-220, 2007.

[7] J. Dittrich, J. Quian\_e-Ruiz, A. Jindal, Y. Kargin, V. Setty, and J. Schad, “Hadoop++: Making a Yellow Elephant Run Like a Cheetah (without it Even Noticing),” Proc. VLDB Endowment, vol. 3, no. 1/2, pp. 515-529, 2010.

[8] H. Garcia-Molina and W.J. Labio, “Efficient Snapshot Differential Algorithms for Data Warehousing,” technical report, Stanford Univ., 1996.

[9] Google Inc., “Cloud Computing-What is its Potential Value for Your Company?” White Paper, 2010.

[10] R. Huebsch, J.M. Hellerstein, N. Lanham, B.T. Loo, S. Shenker, and I. Stoica, “Querying the Internet with PIER,” Proc. 29th Int’l Conf. Very Large Data Bases, pp. 321-332, 2003.

[11] H.V. Jagadish, B.C. Ooi, K.-L. Tan, Q.H. Vu, and R. Zhang, “Speeding up Search in Peer-to-Peer Networks with a Multi-Way Tree Structure,” Proc. ACM SIGMOD Int’l Conf. Management of Data, 2006.

[12] H.V. Jagadish, B.C. Ooi, K.-L. Tan, C. Yu, and R. Zhang, “iDistance: An Adaptive B+-Tree Based Indexing Method for Nearest Neighbor Search,” ACM Trans. Database Systems, vol. 30, pp. 364-397, June 2005.

[13] H.V. Jagadish, B.C. Ooi, and Q.H. Vu, “BATON: A Balanced Tree Structure for Peer-to-Peer Networks,” Proc. 31st Int’l Conf. Very Large Data Bases (VLDB ’05), pp. 661-672, 2005.

[14] A. Lakshman and P. Malik, “Cassandra: Structured Storage System on a P2P Network,” Proc. 28th ACM Symp. Principles of Distributed Computing (PODC ’09), p. 5, 2009.

[15] W.S. Ng, B.C. Ooi, K.-L. Tan, and A. Zhou, “PeerDB: A P2P-Based System for Distributed Data Sharing,” Proc. 19th Int’l Conf. Data Eng., pp. 633-644, 2003.

[16] Oracle Inc., “Achieving the Cloud Computing Vision,” White Paper, 2010.

[17] V. Poosala and Y.E. Ioannidis, “Selectivity Estimation without the Attribute Value Independence Assumption,” Proc. 23rd Int’l Conf. Very Large Data Bases (VLDB ’97), pp. 486-495, 1997.

[18] M.O. Rabin, “Fingerprinting by Random Polynomials,” Technical Report TR-15-81, Harvard Aiken Computational Laboratory, 1981.

[19] E. Rahm and P. Bernstein, “A Survey of Approaches to Automatic Schema Matching,” The VLDB J., vol. 10, no. 4, pp. 334-350, 2001.

[20] P. Rodr\_ıguez-Gianolli, M. Garzetti, L. Jiang, A. Kementsietsidis, I. Kiringa, M. Masud, R.J. Miller, and J. Mylopoulos, “Data Sharing in the Hyperion Peer Database System,” Proc. Int’l Conf. Very Large Data Bases, pp. 1291-1294, 2005.

[21] Saepio Technologies Inc., “The Enterprise Marketing Management Strategy Guide,” White Paper, 2010.

[22] I. Tatarinov, Z.G. Ives, J. Madhavan, A.Y. Halevy, D. Suciu, N.N. Dalvi, X. Dong, Y. Kadiyska, G. Miklau, and P. Mork, “The Piazza Peer Data Management Project,” SIGMOD Record, vol. 32, no. 3, pp. 47-52, 2003.

[23] A. Thusoo, J. Sarma, N. Jain, Z. Shao, P. Chakka, S. Anthony, H. Liu, P. Wyckoff, and R. Murthy, “HIVE: A Warehousing Solution over a Map-Reduce Framework,” Proc. VLDB Endowment, vol. 2, no. 2, pp. 1626-1629, 2009.

[24] H.T. Vo, C. Chen, and B.C. Ooi, “Towards Elastic Transactional Cloud Storage with Range Query Support,” Proc. VLDB Endowment, vol. 3, no. 1, pp. 506-517, 2010.

[25] S. Wu, S. Jiang, B.C. Ooi, and K.-L. Tan, “Distributed Online Aggregation,” Proc. VLDB Endowment, vol. 2, no. 1, pp. 443-454, 2009.

**11. FUTURE ENHANCEMENT**

* Open nature of peer-to-peer systems exposes them to malicious activity. Building trust relationships among peers can mitigate attacks of malicious peers.
* This paper presents distributed algorithms that enable a peer to reason about trustworthiness of other peers based on past interactions and recommendations.
* Peers create their own trust network in their proximity by using local information available and do not try to learn global trust information.
* Two contexts of trust, service, and recommendation contexts are defined to measure trustworthiness in providing services and giving recommendations.
* Interactions and recommendations are evaluated based on importance, recentness, and peer satisfaction parameters. Additionally, recommender’s trustworthiness and confidence about a recommendation are considered while evaluating recommendations.
* Simulation experiments on a file sharing application show that the proposed model can mitigate attacks on 16 different malicious behavior models.
* In the experiments, good peers were able to form trust relationships in their proximity and isolate malicious peers.

**CONCLUSION:**

* We have discussed the unique challenges posed by sharing and processing data in an inter-businesses environment and proposed Best Peer++, a system which delivers elastic data sharing services, by integrating cloud computing, database, and peer-to-peer technologies.
* The benchmark conducted on Amazon EC2 cloud platform shows that our system can efficiently handle typical workloads in a corporate network and can deliver near linear query throughput as the number of normal peers grows.
* Therefore, BestPeer++ is a promising solution for efficient data sharing within corporate networks.