An Industry Oriented Mini Project Report(CS704PC)

On

“PROFIT MAXIMIZATION FOR CLOUD ADOPTION AND MIGRATION IN CLOUD COMPUTING”

Submitted in partial fulfilment of the requirements

for the award of the degree of

Bachelor of Technology

in

Computer Science and Engineering

by

Mr.P.SHIVA RAMA KRISHNA(18261A05G3)

Under the Guidance of

Ms. VIJAYALAXMI C HANDARAGALL

(Assistant Professor)

DEPT. OF COMPUTER SCIENCE AND ENGINEERING

MAHATMA GANDHI INSTITUTE OF TECHNOLOGY

(Affiliated to Jawaharlal Nehru Technological University

Hyderabad) Kokapet (V), Gandipet (M), Hyderabad.

Telangana -500 075. (India)

2021-2022

MAHATMA GANDHI INSTITITE OF TECHNOLOGY

(Affiliated to Jawaharlal Nehru Technological University Hyderabad)

GANDIPET, HYDERABAD – 500 075. Telangana

CERTIFICATE

This is to certify that the industry oriented mini project (CS704PC) entitled **PROFIT MAXIMIZATION FOR CLOUD ADOPTION AND MIGRATION IN CLOUD COMPUTING** is being submitted by **P.SHIVA RAMA KRISHNA** in partial fulfillment for the award of **B.Tech** in **Computer Science and Engineering** to **Jawaharlal Nehru Technological University Hyderabad** is a record of bonafide work carried out by him under the guidance of **MS.Vijayalaxmi C Handaragall, assistant professor, department of cse.**

The results embodied in this project have not been submitted to any other University or Institute for the award of any degree or diploma.

Project Guide	**Project Coordinator**	**Head of the Department**
Ms.Vijayalaxmi.C. Handaragall	Dr. M.Rama Bai	Dr.C.R.K. Reddy
Assistant Professor	Professor, Dept. of CSE	Professor, Dept. of CSE

External Examiner

DECLARATION

This is to certify that the work reported in this project titled **“PROFIT MAXIMIZATION FOR CLOUD ADOPTION AND MIGRATION IN CLOUD COMPUTING**” is a record of work done by us in the Department of Computer Science and Engineering, Mahatma Gandhi Institute of Technology, Hyderabad.

No part of the work is copied from books/journals/internet and wherever the portion is taken, the same has been duly referred in the text. The report is based on the work done entirely by us and not copied from any other source.

P.SHIVA RAMA KRISHNA
(18261A05G3)

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without introducing the people who made it possible and whose constant guidance and encouragement crowns all efforts with success. They have been a guiding light and source of inspiration towards the completion of the project.

I would like to express my sincere thanks to **Dr. K. Jaya Sankar, Principal MGIT**, for providing the work facilities in the college.

I am also thankful to **Dr. C.R.K Reddy, Professor & Head of Department**, Dept. of Computer Science and Engineering for providing excellent infrastructure and a conducive atmosphere for completing this project successfully.

I am also extremely thankful to my Project Coordinators, **Dr. M.RAMABHAI, Professor & Head of Department, Dept. of CSE, Mr.R.Srinivas, Sr. Assistant Professor, Dept. of CSE and Ms. K.SUNITHA, Assistant Professor, Dept. of CSE**, for their valuable suggestions and interest throughout the course of this project.

I would like to express my sincere gratitude and indebtedness to my Project Guide, **Ms.Vijayalaxmi C Handaragall, Assistant Professor, Dept. of CSE** who has supported me throughout the project with patience and knowledge.
I convey my heartfelt thanks to the lab staff for allowing me to use the required equipment whenever needed.
Finally, I would like to take this opportunity to thank my family for their support throughout the work. I sincerely acknowledge and thank all those who gave directly their support in completion of this work.

P.SHIVA RAMA KRISHNA(18261A05G3)

TABLE OF CONTENTS

LIST OF TABLE

LIST OF FIGURE

ABSTRACT

Along with the development of cloud computing, more and more applications are migrated into the cloud. An important feature of cloud computing is pay-as-you-go. However, most users always should pay more than their actual usage due to the one-hour billing cycle. In addition, most cloud service providers provide a certain discount for long-term users, but short-term users with small computing demands cannot enjoy this discount. To reduce the cost of cloud users, a new role is introduced, which is cloud broker. A cloud broker is an intermediary agent between cloud providers and cloud users. It rents a number of reserved VMs from cloud providers with a good price and offers them to users on an on-demand basis at a cheaper price than that provided by cloud providers. Besides, the cloud broker adopts a shorter billing cycle compared with cloud providers. By doing this, the cloud broker can reduce a great amount of cost for user. In addition to reduce the user cost, the cloud broker also could earn the difference in prices between on-demand and reserved VMs. In this project, the focus is on how to configure a cloud broker and how to price its VMs such that its profit can be maximized on the premise of saving costs for users. Profit of a cloud broker is affected by many factors such as the user demands, the purchase price and the sales price of VMs, the scale of the cloud broker, etc.. Moreover, these factors are affected mutually, which makes the analysis on profit more complicated. In this project, firstly a synthetically analysis is given on all the affecting factors, and define an optimal multi server configuration and VM pricing problem which is modeled as a profit maximization problem. Secondly, combining the partial derivative and bisection search method, a heuristic method is proposed to solve the optimization problem. The near-optimal solutions can be used to guide the configuration and VM pricing of the cloud broker. Moreover, a series of comparisons are given which show that a cloud broker can save a considerable cost for user.

CHAPTER 1

INTRODUCTION

Over the past few years, cloud computing has experienced tremendous development. More and more cloud providers have jumped on the cloud bandwagon, and they centrally manage a variety of resources such as hardware and software and deliver them over the internet in the form of services to customers on demand. Thanks to unique properties such as elasticity, ﬂexibility, apparently unlimited computational power, and pay-as-you-use pricing model, cloud computing can reduce the requirement of clients for large capital outlays for hardware necessary to deploy service and the human expenses to operate it [4]. Hence, an increasing number of clients are transferring their business to the cloud.

One important feature of cloud computing is pay-as you-use which contains two meanings. First, according to the customer resource demand such as CPU, memory, etc., the physical machines are dynamically segmented using virtualization technologies and provided to customers in the form of virtual machines (VMs), and customers pay according to the amount of resources they actually consumed. Second, the VMs can be dynamically allocated and de allocate data anytime, and customers should pay based on how long the resources are actually used. Nevertheless, the pay-as-you-use pricing model is presently only conceptual due to the extreme complexity in monitoring and auditing resource usage [8], and cloud providers usually adopt an hourly billing scheme; in other words, the Billing Time Unit (BTU) of the cloud providers is one hour, for instance, Amazon EC2 [9]. Therefore, the customers should pay for the resources by the hour even if they do not actually utilize the allocated resources in the whole billing horizon [10]. This leads to a waste of resources and raises the cost of customers to a certain degree. Network security algorithm is used in this project to provide security to the user data .the algorithm that is user is advanced encryption standard.

1.1 Problem Definition

Profit maximization function is defined to find an optimal combination of the server size R and the queue capacity k such that the profit is maximized. However, this strategy has further implication other than just losing the revenue from some services, because it also implies loss of reputation Therefore loss of future customers. In treating a cloud service platform as an

M/M/m model, and the problem of optimal multi server configuration for profit maximization was formulated and solved. This work is the most relevant work to ours, but it adopts a single renting scheme to configure a multi-server system which cannot adopt to the varying market demand and lead to low service quality and great resource waste. To overcome this weakness another resource management strategy is used in, which is cloud federation. Using federation different providers running services that have complementary resource requirement over time can mutually collaborate to share their respective resources in order to fulfill each ones demand. However, providers should make an intelligent decision about utilization of the federation (either as a contributor or as a customer of resources) depending on different conditions that they might face.

1.2 Existing System

Along with the development of cloud computing, more and more applications are migrated into the cloud. An important feature of cloud computing is pay-as-you-go. However, most users always should pay more than their actual usage due to the one-hour billing cycle. In addition, most cloud service providers provide a certain discount for long-term users, but short-term users with small computing demands cannot enjoy this discount. To reduce the cost of cloud users, a new role is introduced, which is cloud broker. A cloud broker is an intermediary agent between cloud providers and cloud users. It rents a number of reserved VMs from cloud providers with a good price and offers them to users on an on-demand basis at a cheaper price than that provided by cloud providers .Besides, the cloud broker adopts a shorter billing cycle compared with cloud providers. By doing this, the cloud broker can reduce a great amount of cost for user. In addition to reduce the user cost, the cloud broker also could earn the difference in prices between on-demand and reserved VMs.

1.3 Proposed System

In this project, the focus is on how to configure a cloud broker and how to price its VMs such that its profit can be maximized on the premise of saving costs for users. Profit of a cloud broker is affected by many factors such as the user demands, the purchase price and the sales price of VMs, the scale of the cloud broker, etc.. Moreover, these factors are affected mutually, which makes the analysis on profit more complicated. In this project, firstly a synthetically analysis is given on all the affecting factors, and define an optimal multi server configuration and VM pricing problem which is modeled as a profit maximization problem. Secondly, combining the partial derivative and bisection search method, a heuristic method is proposed to solve the optimization problem. The near-optimal solutions can be used to guide the configuration and VM pricing of the cloud broker. Moreover, a series of comparisons are given which show that a cloud broker can save a considerable cost for users.

1.4 Requirement Specification

1.4.1 Hardware Requirements

• Hardware	:	Pentium Dual Core
• Speed	:	2.80GHz
• RAM	:	1GB
• Hard Disk	:	20 GB

1.4.2 Software Requirements

• Operating System	:	Windows 10
• Technology	:	Java8 andJ2EE
• Web Technologies	:	Html, JavaScript, CSS
• IDE	:	Net beans
• Web Server	:	Tomcat
• Database	:	My SQL

1.5 Technology

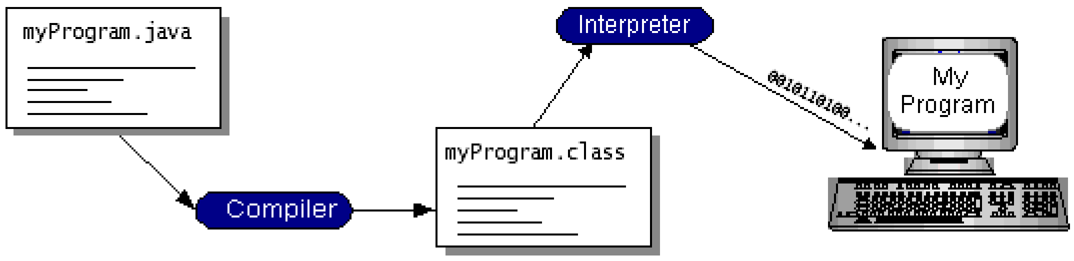
1.5.1 Java

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

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

- Interpreted
- Multithreaded
- Robust
- Dynamic
- Secure

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



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

1.5.2 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 runon.

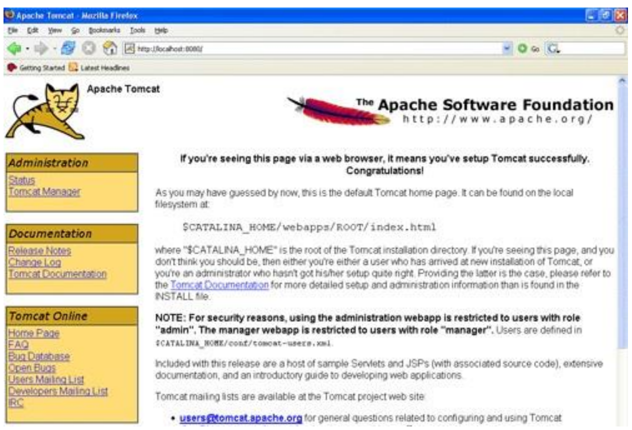
To gain a wider acceptance of JDBC, Sun based JDBC’s framework on ODBC. As you discovered earlier in this chapter, ODBC has widespread support on a variety of platforms. Basing JDBC on ODBC will allow vendors to bring JDBC drivers to market much faster than developing a completely new connectivity solution.

JDBC was announced in March of 1996. It was released for a 90 day public review that ended June 8, 1996. Because of user input, the final JDBC v1.0 specification was released soon after.

The remainder of this section will cover enough information about JDBC for you to know what it is about and how to use it effectively. This is by no means a complete overview of JDBC. That would fill an entire book.

1.5.3 Tomcat 6.0 webserver

Tomcat is an open source web server developed by Apache Group. Apache Tomcat is the servlet container that is used in the official Reference Implementation for the Java Servlet and JavaServer Pages technologies. The Java Servlet and JavaServer Pages specifications are developed by Sun under the Java Community Process. Web Servers like Apache Tomcat support only web components while an application server supports web components as well as business components (BEAs Weblogic, is one of the popular application server).To develop a web application with jsp/servlet install any web server like JRun, Tomcat etc to run yourApplication.



If you're seeing this page via a web browser, it means you've setup Tomcat successfully. Congratulations!

As you may have guessed by now, this is the default Tomcat home page. It can be found on the local filesystem at:

```
$CATALINA_HOME/webapps/ROOT/index.html
```

where "$CATALINA_HOME" is the root of the Tomcat installation directory. If you're seeing this page, and you don't think you should be, then either you're either a user who has arrived at new installation of Tomcat, or you're an administrator who hasn't got his/her setup quite right. Providing the latter is the case, please refer to the Tomcat Documentation for more detailed setup and administration information than is found in the INSTALL file.

NOTE: For security reasons, using the administration webapp is restricted to users with role "admin". The manager webapp is restricted to users with role "manager". Users are defined in `$CATALINA_HOME/conf/tomcat-users.xml`.

Included with this release are a host of sample Servlets and JSPs (with associated source code), extensive documentation, and an introductory guide to developing web applications.

Tomcat mailing lists are available at the Tomcat project web site:

- **users@tomcat.apache.org** for general questions related to configuring and using Tomcat

Tomcatwebserver

CHAPTER 2

LITERATURE SURVEY

Table 2.1 Literature Survey

Sno	**Year of Publish**	**Title**	**Methodology/ Technology**	**Advantages**	**Disadvantages**
1	2020	Transferable Knowledge for low-cost decision making in cloud environment	Random forest, support vector regression	TL scheme increases overall efficiency.	Taking large amount of data to train
2	2018	Profit maximization for cloud brokers in cloud computing	Queue model optimization technique	Cloud brokers adopts a shorter billing cycle compared with cloud providers	Many factors effects such as user demands, purchase price of vm scale of brokers.
3	2017	Customer satisfaction aware optimal multi-server configuration for profit maximization in cloud Computing	Queuing model	Service-level agreement, renting price, energy consumption.	Affects the quality of services.

4	2017	Maximizing the profit of cloud broker with priority aware pricing	Priority aware, pricing scheme, cost effective, reservation algorithm	Brokers will charge user with different prices based on priorities	Ignoring the pricing issues
5	2015	Profit maximization scheme with guaranteed quality of service in cloud computing	Queue model, Optimization technique	Double Quality guaranteed scheme	If there is no guarantee of service quality leads to serious resource wastage

CHAPTER 3

METHODOLOGY FOR PROFIT MAXIMIZATION

3.1 System Architecture

System architecture that defines the structure, behavior and more views of a system. A system architecture can consist of system components and the sub-systems developed, that will work together to implement the overall system.

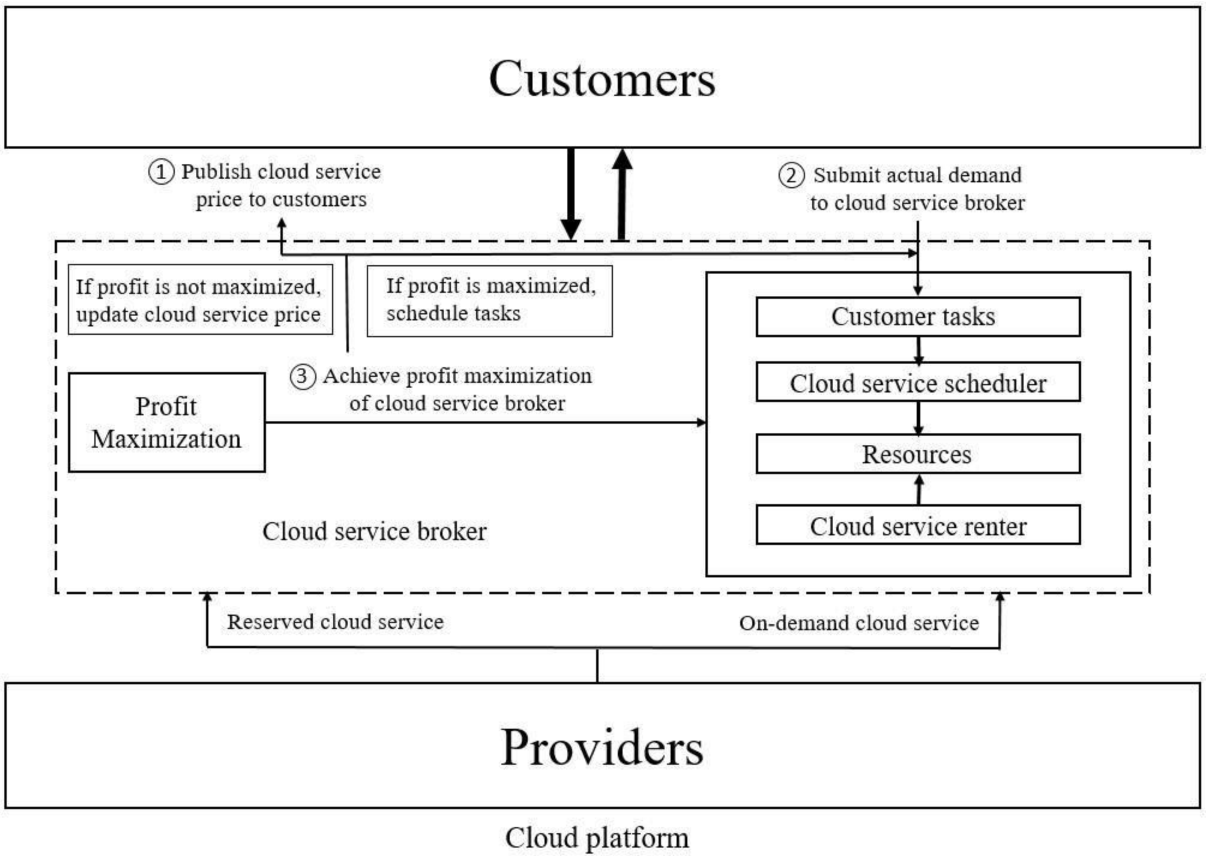

Customers
① Publish cloud service price to customers
② Submit actual demand to cloud service broker
If profit is not maximized, update cloud service price
If profit is maximized, schedule tasks
Customer tasks
Cloud service scheduler
Resources
Cloud service renter
Profit Maximization
③ Achieve profit maximization of cloud service broker
Cloud service broker
Reserved cloud service
On-demand cloud service
Providers
Cloud platform

Figure 3.1 System Architecture

The above figure 3.1 shows that the cloud broker is the mediator between customer and provider. Cloud broker will rent the on demand cloud services from different cloud providers and provide services to the cloud user. Profit will become maximum when the user

Rents the services from broker the broker will schedule the cloud services to the user based

On the requirements.

3.2 System Design

3.2.1 Use Case Diagram For Profit Maximization

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.

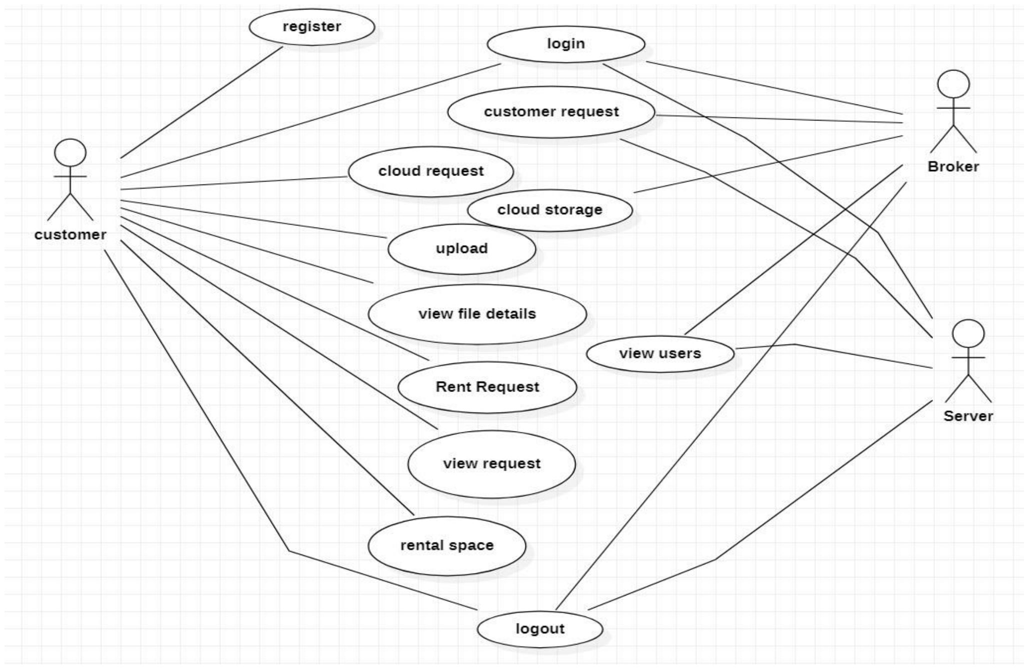


Figure 3.2 Use Case Diagram

In figure 3.2 the customer has to login successfully then the user has to send Request to the cloud so that the broker has to accept the request send by the customer. Then that can be stored in cloud storage.

3.2.2 Sequence Diagram For Profit Maximization

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.

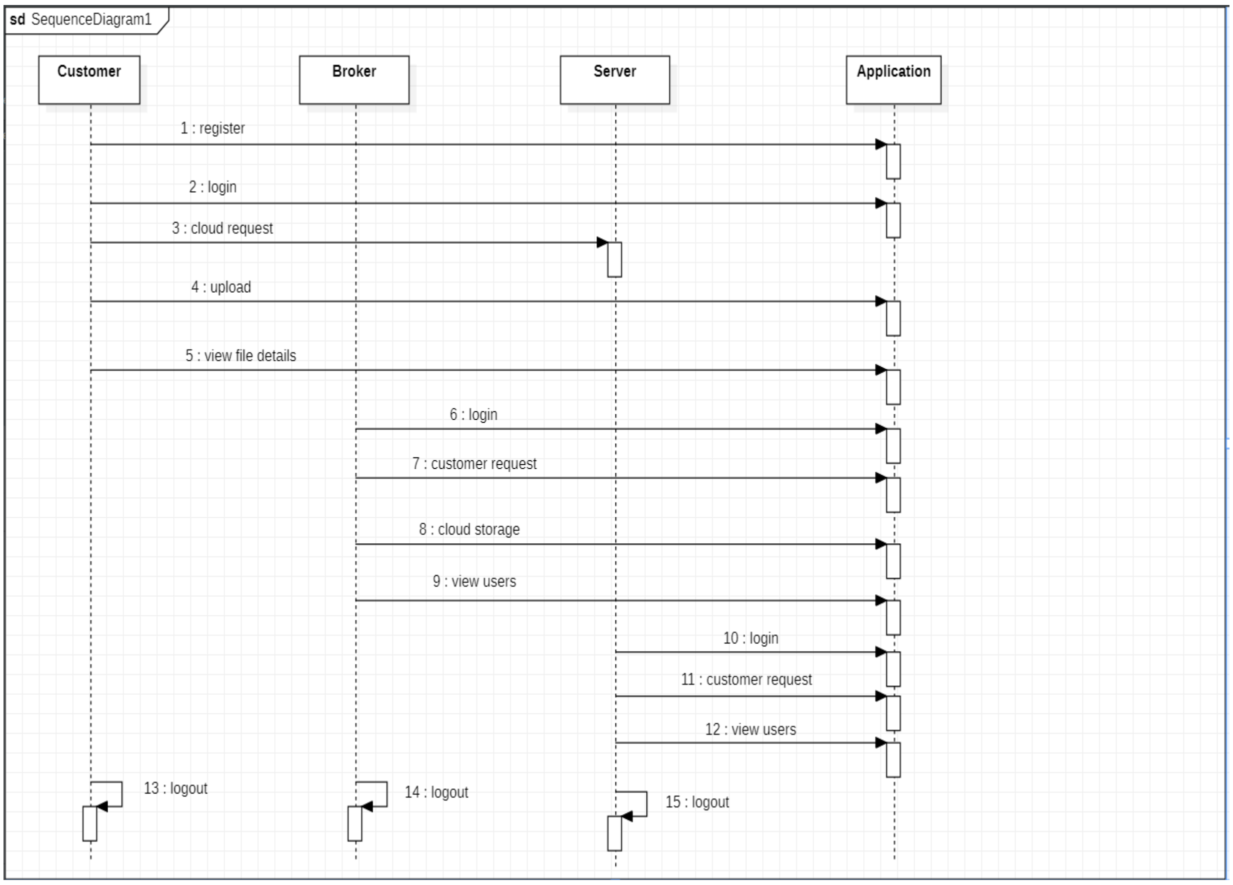


Figure 3.3 Sequence Diagram

Figure 3.3 shows the sequence followed in the project, where the customer has to upload the text for cloud space after that same customer has to send the rent request to the another customer .He has to login and check the view request that should be accepted so the rent requested to customer is successful.

3.3 Modules

1. **Customer**

   Here the customer has to register with the application then only the customer has to login with the application after successful login the customer can perform some actions such as cloud request, upload, view file details, Rent Request, view request, rental space, logout.

2. **Broker**

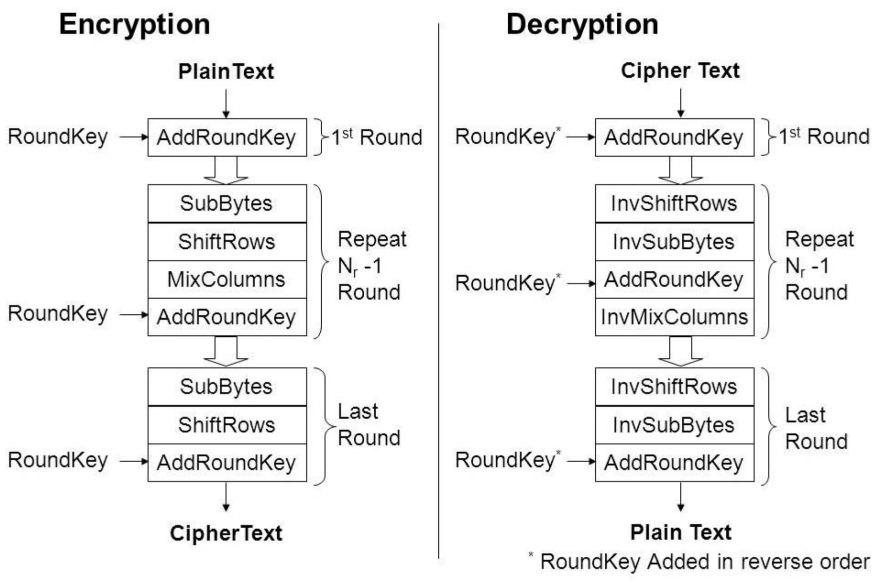
   Here the broker no need not to register with the application directly can login, after successful login the broker can perform some actions such as customer request, cloud storage, view users, logout

3. **Server**

   Here the server no need not to register with the application directly can login, after successful login the server can perform some actions such as customer request, view users, logout.

3.4 Algorithm

3.4.1 Implementation



3.4.2 Working

Advanced Encryption Standard (AES) is a specification for the encryption of electronic data established by the U.S National Institute of Standards and Technology (NIST) in 2001. AES is widely used today as it is a much stronger than DES and triple DES despite being harder to implement.

Points to remember

- AES is a block cipher.
- The key size can be 128/192/256bits.
- Encrypts data in blocks of 128 bits each.

That means it takes 128 bits as input and outputs 128 bits of encrypted cipher text as output. AES relies on substitution-permutation network principle which means it is performed usingaseriesoflinkedoperationswhichinvolvesreplacingandshufflingoftheinputdata.

Working of the cipher

AES performs operations on bytes of data rather than in bits. Since the block size is 128 bits, the cipher processes 128 bits (or 16 bytes) of the input data at a time.

The number of rounds depends on the key length as follows:

- 128 bit key – 10rounds
- 192 bit key – 12rounds
- 256 bit key – 14rounds

Creation of Round keys

A Key Schedule algorithm is used to calculate all the round keys from the key. So the initial key is used to create many different round keys which will be used in the corresponding round of the encryption.

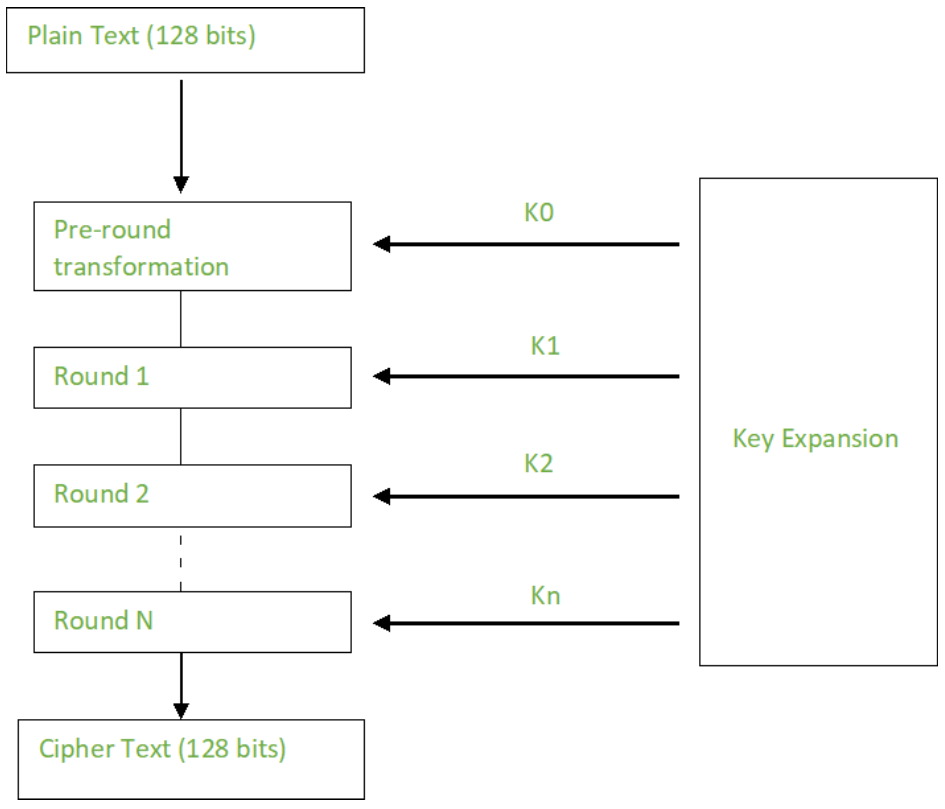


Figure 3.4 Round keys

Encryption

AES considers each block as a 16 byte (4 byte x 4 byte = 128) grid in a column major arrangement.

[b0 | b4 | b8 | b12|

| b1 | b5 | b9 | b13|

| b2 | b6 | b10| b14|

| b3 | b7 | b11| b15]

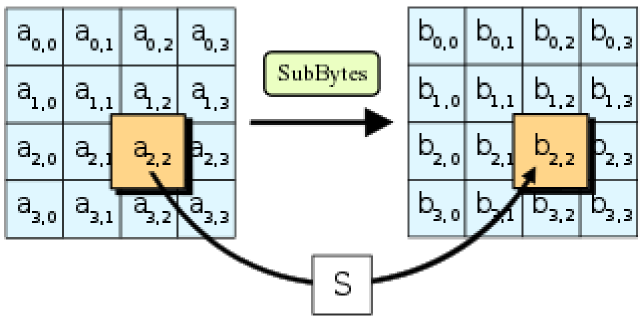
Each round comprises of 4 steps :

- Sub Bytes
- Shift Rows
- Mix Columns
- Add Round Key

The last round doesn't have the Mix Columns round.

The Sub Bytes does the substitution and Shift Rows and Mix Columns performs the permutation in the algorithm.

Sub Bytes



$a_{0,0}$ $a_{0,1}$ $a_{0,2}$ $a_{0,3}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$ $a_{1,3}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$ $a_{2,3}$
$a_{3,0}$ $a_{3,1}$ $a_{3,2}$ $a_{3,3}$
SubBytes
$b_{0,0}$ $b_{0,1}$ $b_{0,2}$ $b_{0,3}$
$b_{1,0}$ $b_{1,1}$ $b_{1,2}$ $b_{1,3}$
$b_{2,0}$ $b_{2,1}$ $b_{2,2}$ $b_{2,3}$
$b_{3,0}$ $b_{3,1}$ $b_{3,2}$ $b_{3,3}$
S

Figure 3.5 Sub Bytes

This step implements the substitution. In this step each byte is substituted by another byte.(Its performed using a lookup table also called the S-box. This substitution is done in a way that a byte is never substituted by itself and also not substituted by another byte which is a compliment of the current byte .The result of this step is a 16 byte (4 x 4) matrix like before.

The next two steps implement the permutation.

Shift Rows

This step is just as it sounds. Each row is shifted a particular number of times.

- The first row is not shifted
- The second row is shifted once to the left.
- The third row is shifted twice to the left.
- The fourth row is shifted thrice to the left.

(A left circular shift is performed.)

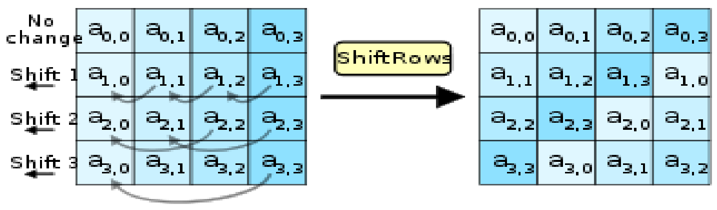


Figure 3.6 Shift Rows

Mix Columns

This step is basically a matrix multiplication. Each column is multiplied with a specific matrix and thus the position of each byte in the column is changed as a result.

This step is skipped in the last round.

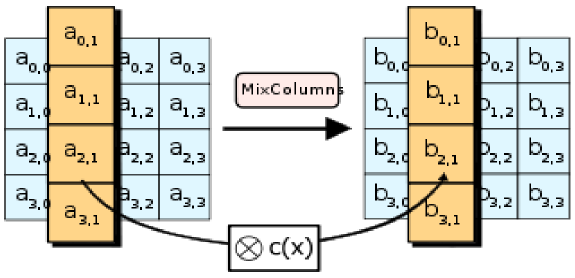


Figure 3.7 Mix Columns

Add Round Keys

Now the resultant output of the previous stage is XOR-ed with the corresponding round key. Here, the 16 bytes is not considered as a grid but just as 128 bits of data.

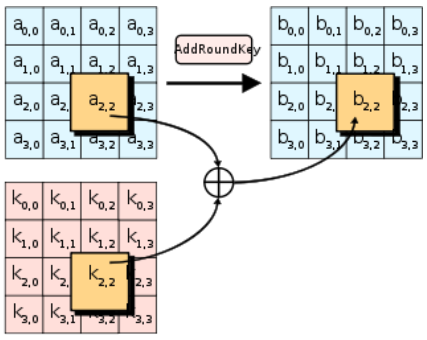


Figure 3.8 Add Round Key

After all these rounds 128 bits of encrypted data is given back as output. This process is repeated until all the data to be encrypted undergoes this process.

Decryption

The stages in the rounds can be easily undone as these stages have an opposite to it which when performed reverts the changes .Each 128 blocks goes through the 10,12 or 14 rounds depending on the key size.

The stages of each round in decryption is as follows :

- Add roundkey
- Inverse MixColumns
- ShiftRows
- Inverse SubByte

The decryption process is the encryption process done in reverse so i will explain the steps with notable differences.

Inverse MixColumns

This step is similar to the Mix Columns step in encryption, but differs in the matrix used to carry out the operation.

$$\begin{bmatrix} b0 \\ b1 \\ b2 \\ b3 \end{bmatrix} = \begin{bmatrix} 14 & 11 & 13 & 9 \\ 9 & 14 & 11 & 13 \\ 13 & 9 & 14 & 11 \\ 11 & 13 & 9 & 14 \end{bmatrix} \begin{bmatrix} c0 \\ c1 \\ c2 \\ c3 \end{bmatrix}$$

Inverse Sub Bytes

Inverse S-box is used as a lookup table and using which the bytes are substituted during decryption.

AES instruction set is now integrated into the CPU (offers throughput of several GB/s)to improve the speed and security of applications that use AES for encryption and decryption . Even though its been 20 years since its introduction, everyone has failed to break the AES algorithm as it is infeasible even with the current technology.

CHAPTER 4

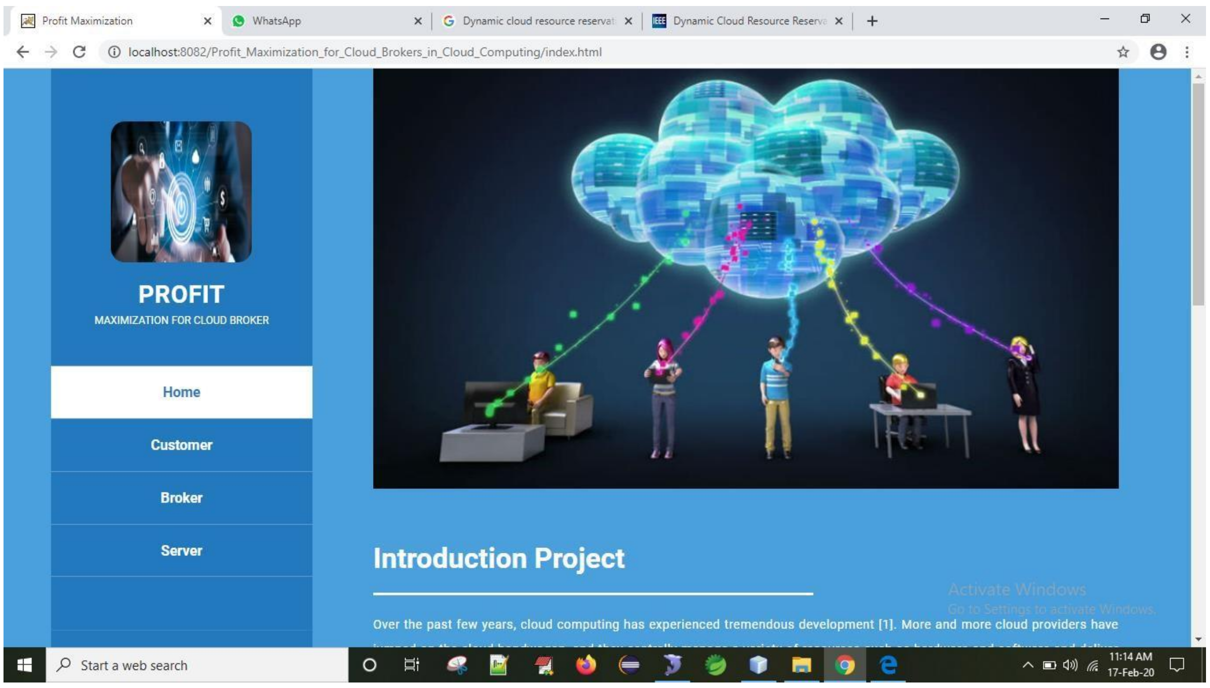
TESTING AND RESULTS

4.1 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.

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

4.2 Results


Profit Maximization
WhatsApp
Dynamic cloud resource reservat
Dynamic Cloud Resource Reserv
localhost:8082/Profit_Maximization_for_Cloud_Brokers_in_Cloud_Computing/index.html
PROFIT
MAXIMIZATION FOR CLOUD BROKER
Home
Customer
Broker
Server
Introduction Project
Start a web search
11:14 AM
17-Feb-20

Over the past few years, cloud computing has experienced tremendous development [1]. More and more cloud providers have

Figure 4.1 Home screen of the web Application

The above figure 4.1 shows that the home screen of the Profit Maximization for Cloud Adoption and Migration in Cloud Computing

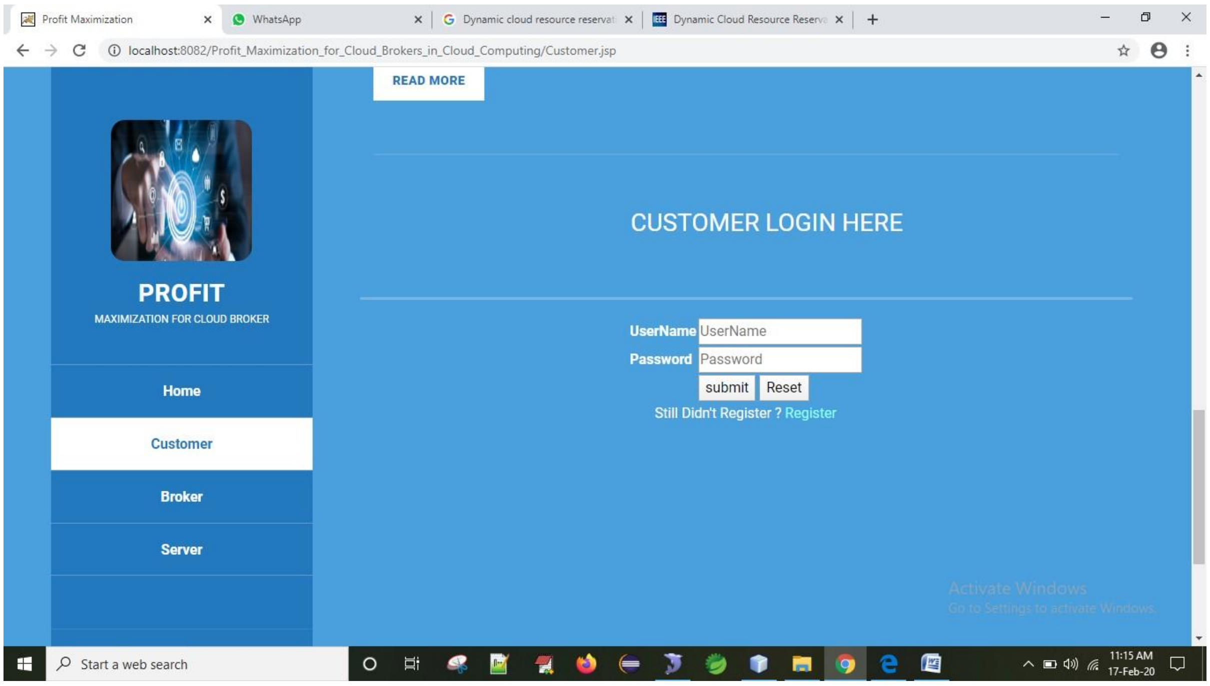
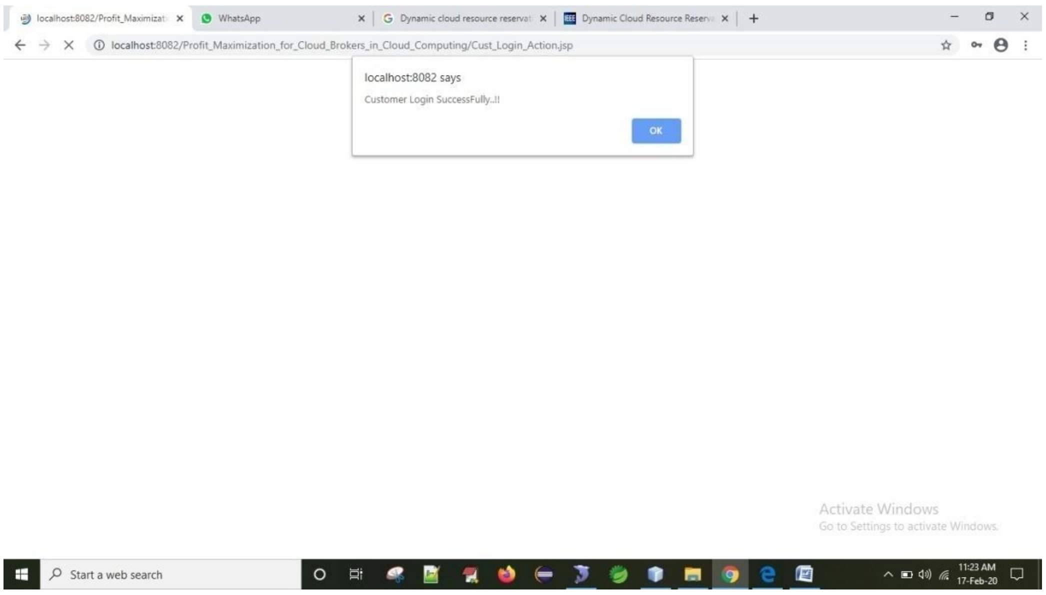


Figure 4.2 Interface Of Customer log in

The above figure 4.2 shows the interface of customer log in which contains the fields like username and password.


localhost:8082 says

Customer Login SuccessFully..!!

Figure 4.3 Customer login Success Page

The above figure 4.3 shows the customer log in success page of the Profit Maximization for Cloud Adoption and Migration in Cloud Computing.

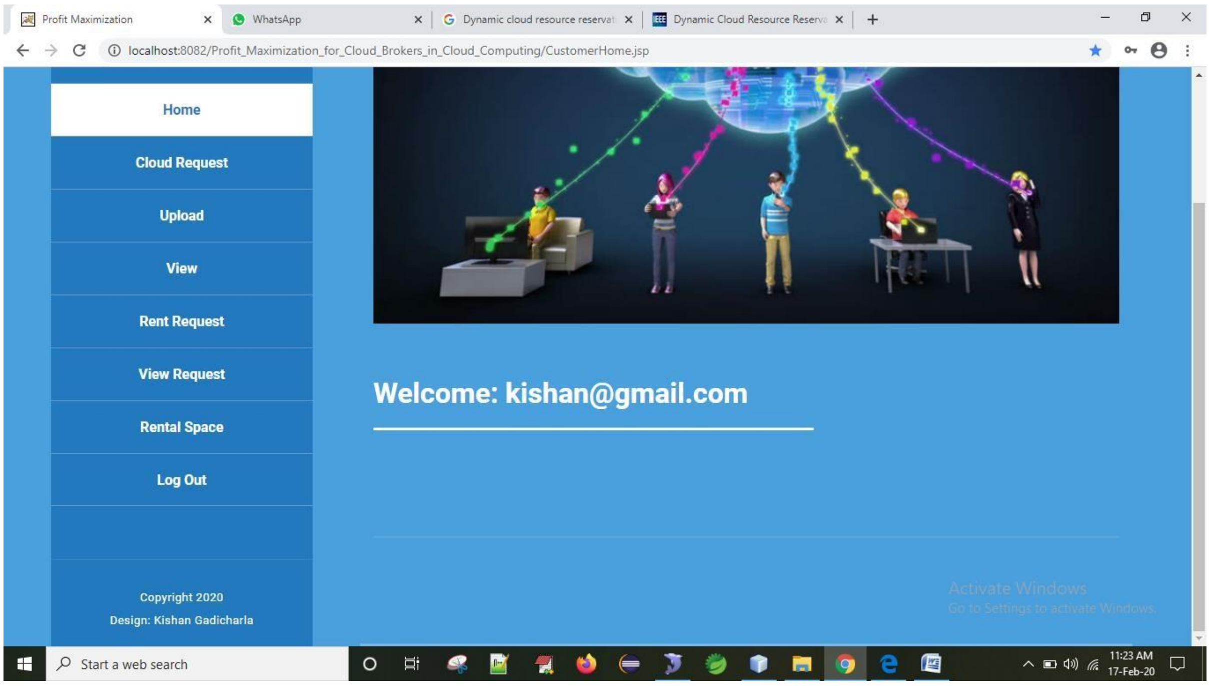
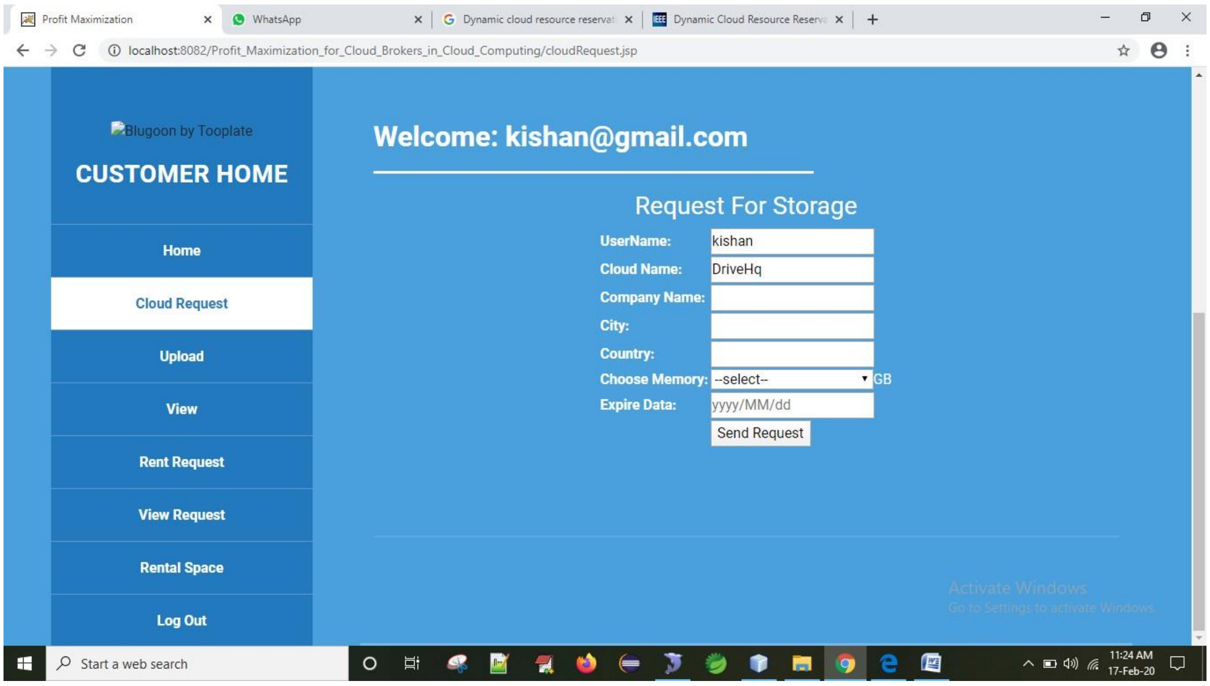

Profit Maximization
WhatsApp
Dynamic cloud resource reservat
Dynamic Cloud Resource Reserva
localhost:8082/Profit_Maximization_for_Cloud_Brokers_in_Cloud_Computing/CustomerHome.jsp
Home
Cloud Request
Upload
View
Rent Request
View Request
Rental Space
Log Out
Copyright 2020
Design: Kishan Gadicharla
Welcome: kishan@gmail.com
Start a web search
11:23 AM
17-Feb-20

Figure 4.4 Customer Home Page

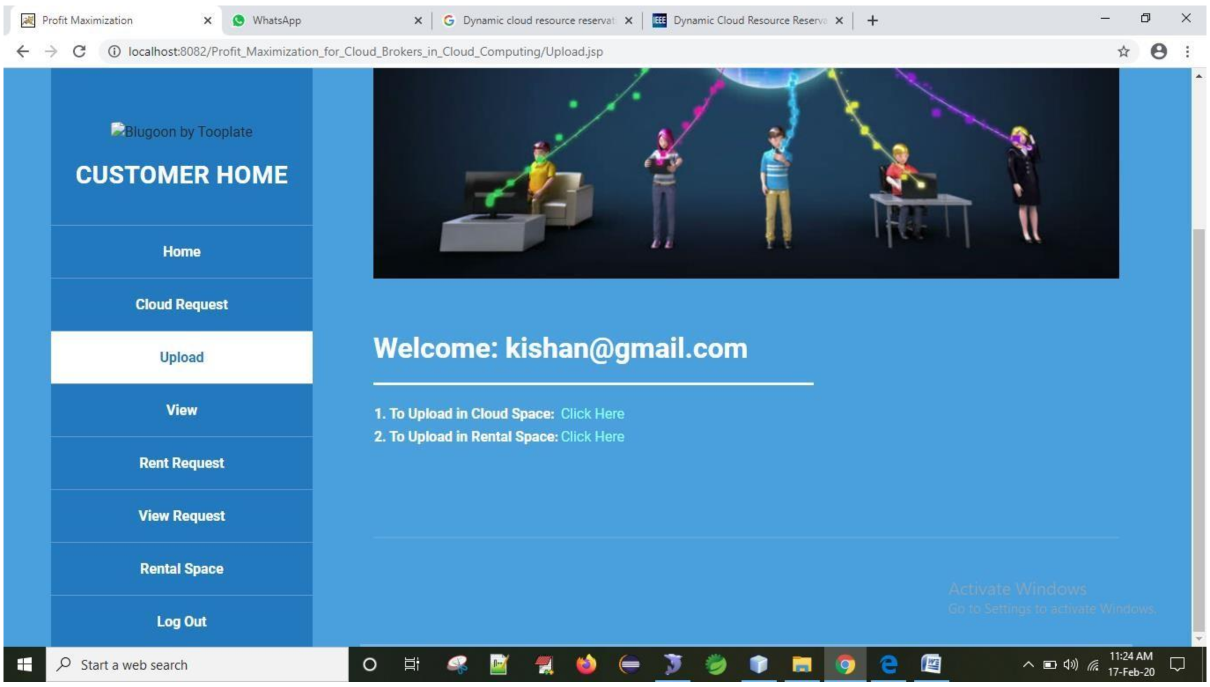
The above figure 4.4 shows the customer home page of the Profit Maximization for Cloud Adoption and Migration in cloud computing.



UserName:	kishan
Cloud Name:	DriveHq
Company Name:	
City:	
Country:	
Choose Memory:	--select-- GB
Expire Data:	yyyy/MM/dd

Figure 4.5 Request To The Cloud

In the above figure 4.5 shows the portal to request to the cloud in the Profit Maximization for cloud adoption and Migration in the Cloud Computing.

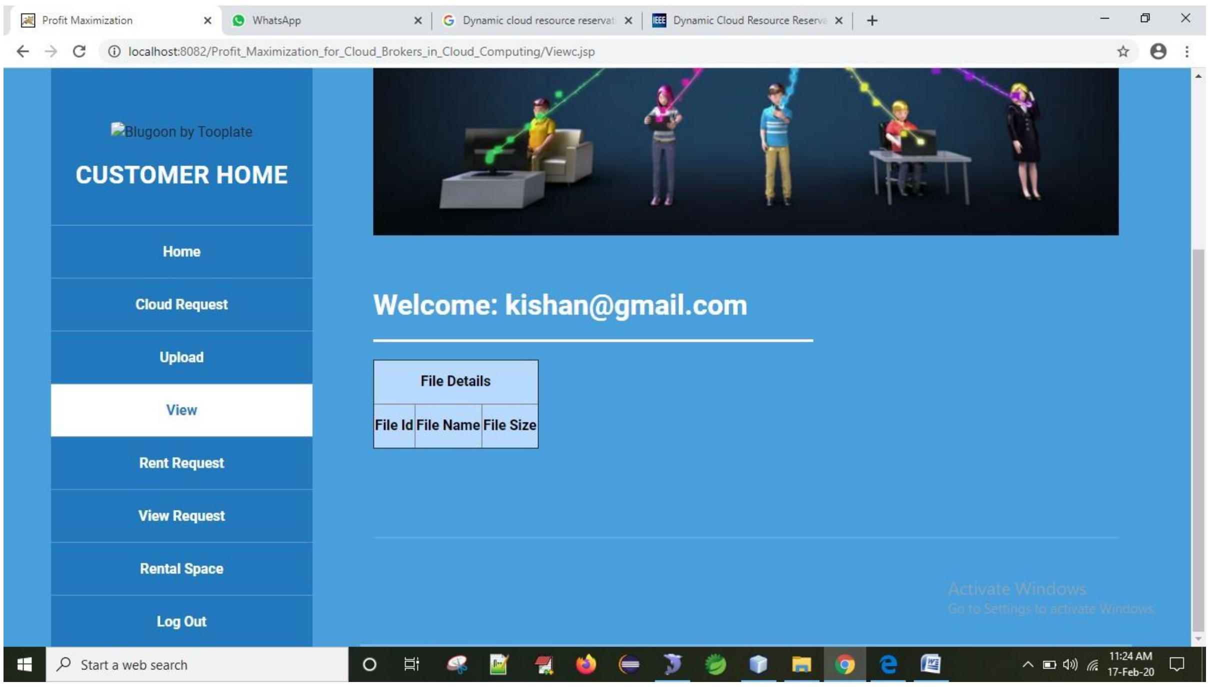


Welcome: kishan@gmail.com

1. To Upload in Cloud Space: Click Here
2. To Upload in Rental Space: Click Here

Figure 4.6 Portal To Upload The Text File

The above figure 4.6 shows the portal to upload the text file in the cloud space.

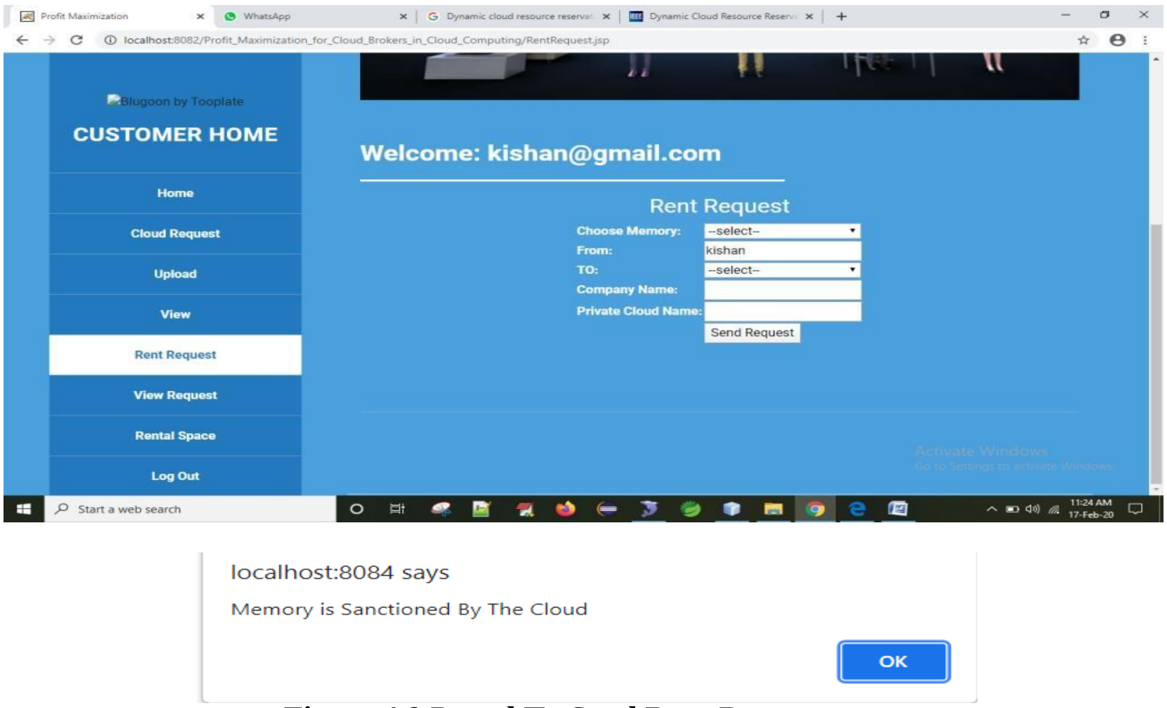


Welcome: kishan@gmail.com

File Details		
File Id	File Name	File Size

Figure 4.7 Portal To View The File Details

The above figure 4.7 depicts the web application portal to view the file details in the profit maximization for cloud adoption and migration in the cloud computing.



localhost:8082/Profit_Maximization_for_Cloud_Brokers_in_Cloud_Computing/RentRequest.jsp

CUSTOMER HOME

Welcome: kishan@gmail.com

Rent Request

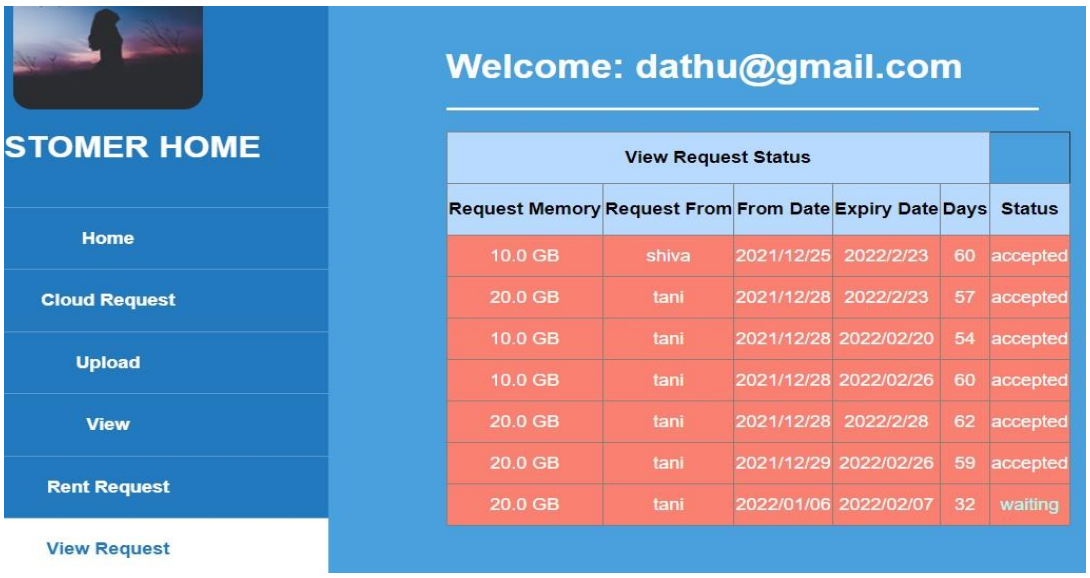
Choose Memory:	--select--
From:	kishan
TO:	--select--
Company Name:	
Private Cloud Name:	
	Send Request

localhost:8084 says

Memory is Sanctioned By The Cloud

Figure 4.8 Portal To Send Rent Request

The above figure 4.8 depicts the web application portal to send rent request to user



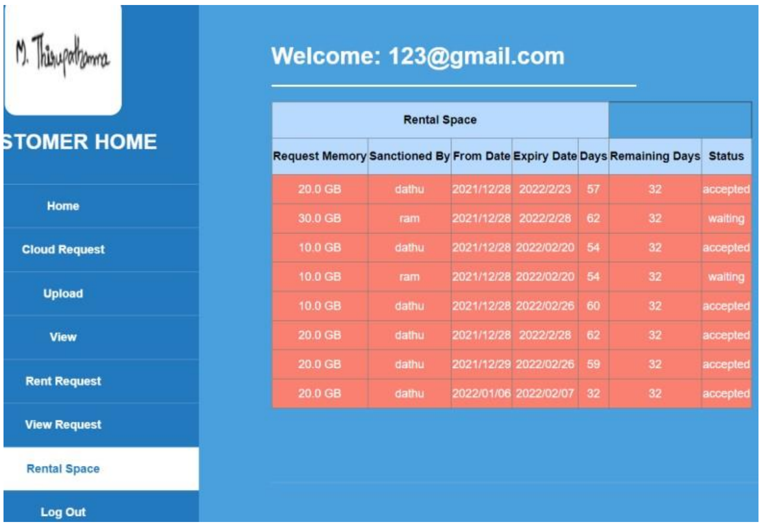
STOMER HOME

Welcome: dathu@gmail.com

View Request Status					
Request Memory	Request From	From Date	Expiry Date	Days	Status
10.0 GB	shiva	2021/12/25	2022/2/23	60	accepted
20.0 GB	tani	2021/12/28	2022/2/23	57	accepted
10.0 GB	tani	2021/12/28	2022/02/20	54	accepted
10.0 GB	tani	2021/12/28	2022/02/26	60	accepted
20.0 GB	tani	2021/12/28	2022/2/28	62	accepted
20.0 GB	tani	2021/12/29	2022/02/26	59	accepted
20.0 GB	tani	2022/01/06	2022/02/07	32	waiting

Figure 4.9 View Request

The above figure 4.9 depicts the web application portal to view request status

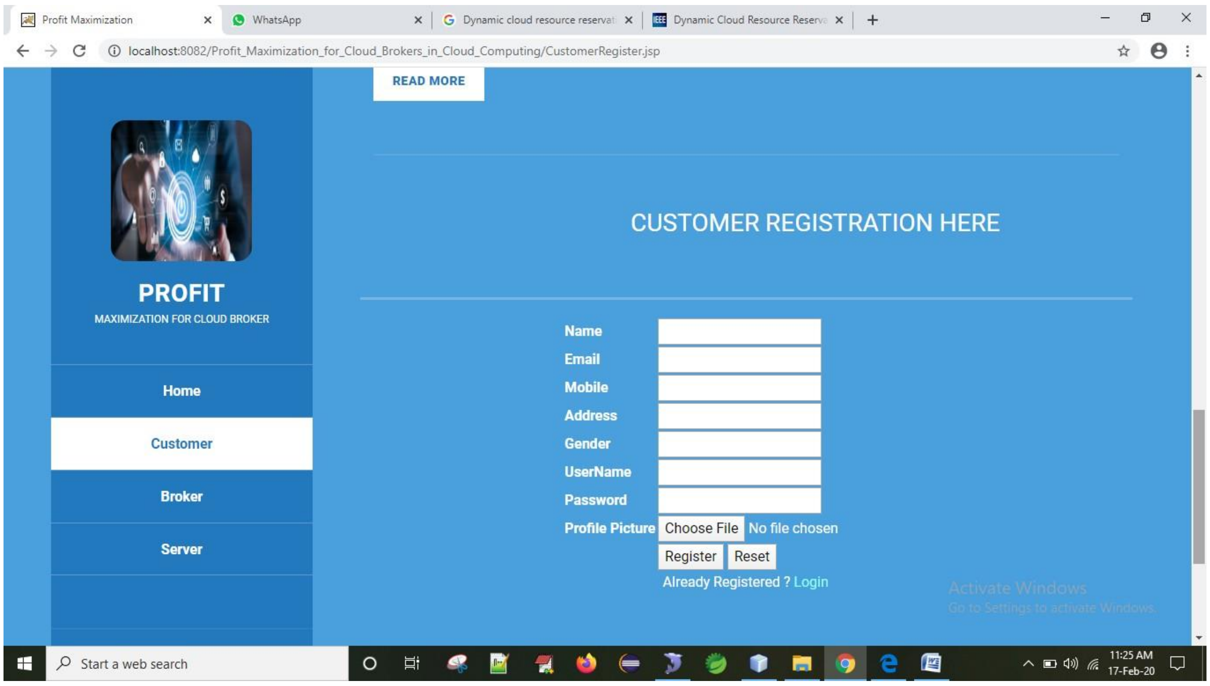


Welcome: 123@gmail.com

Rental Space						
Request Memory	Sanctioned By	From Date	Expiry Date	Days	Remaining Days	Status
20.0 GB	dathu	2021/12/28	2022/2/23	57	32	accepted
30.0 GB	ram	2021/12/28	2022/2/28	62	32	waiting
10.0 GB	dathu	2021/12/28	2022/02/20	54	32	accepted
10.0 GB	ram	2021/12/28	2022/02/20	54	32	waiting
10.0 GB	dathu	2021/12/28	2022/02/26	60	32	accepted
20.0 GB	dathu	2021/12/28	2022/2/28	62	32	accepted
20.0 GB	dathu	2021/12/29	2022/02/26	59	32	accepted
20.0 GB	dathu	2022/01/06	2022/02/07	32	32	accepted

Figure 4.10 Rental Space

The above figure 4.10 depicts the web application portal to rental space for Profit maximization for cloud adoption and migration in cloud computing


Profit Maximization
WhatsApp
Dynamic cloud resource reservati
Dynamic Cloud Resource Reserve
localhost:8082/Profit_Maximization_for_Cloud_Brokers_in_Cloud_Computing/CustomerRegister.jsp
READ MORE
PROFIT
MAXIMIZATION FOR CLOUD BROKER
Home
Customer
Broker
Server
Choose File
No file chosen
Register
Reset
Activate Windows
Go to Settings to activate Windows
Start a web search
11:25 AM
17-Feb-20

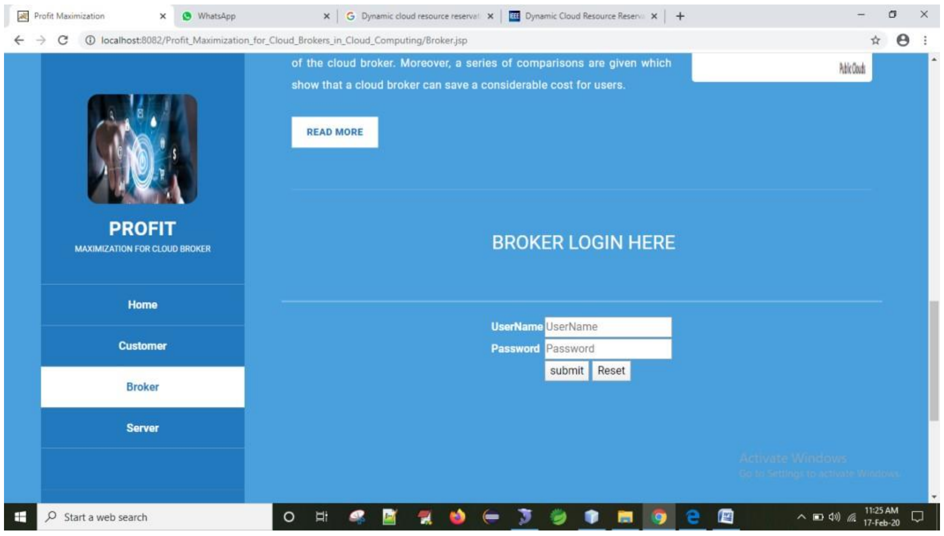
CUSTOMER REGISTRATION HERE

Name	
Email	
Mobile	
Address	
Gender	
UserName	
Password	
Profile Picture	Choose File No file chosen

Already Registered ? Login

Figure 4.11 Customer Registration Page

The above figure 4.11 depicts the web application portal to customer registration page for Profit maximization for cloud adoption and migration in cloud computing

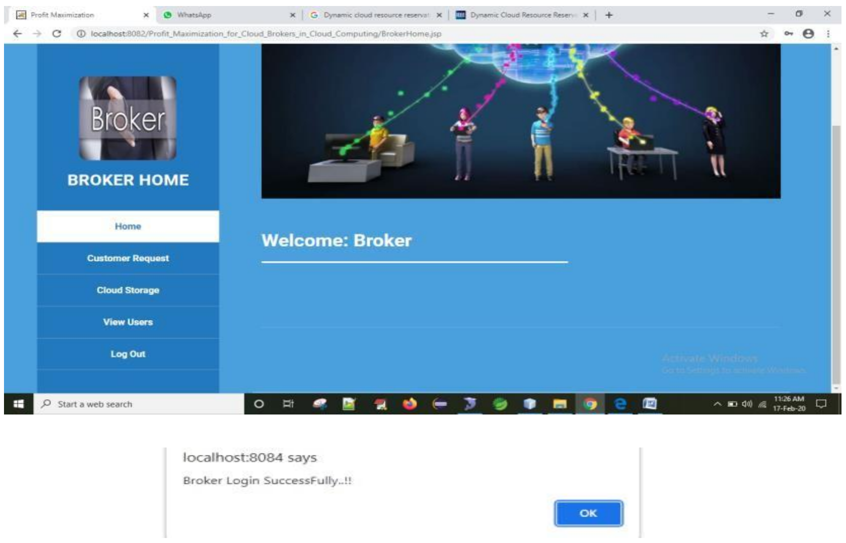


localhost:8082/Profit_Maximization_for_Cloud_Brokers_in_Cloud_Computing/Broker.jsp
PROFIT
MAXIMIZATION FOR CLOUD BROKER
Home
Customer
Broker
Server
READ MORE
BROKER LOGIN HERE
UserName
Password
submit
Reset

of the cloud broker. Moreover, a series of comparisons are given which show that a cloud broker can save a considerable cost for users.

Figure 4.12 Screen broker login

The above figure 4.12 depicts the web application portal to broker login screen for Profit maximization for cloud adoption and migration in cloud computing



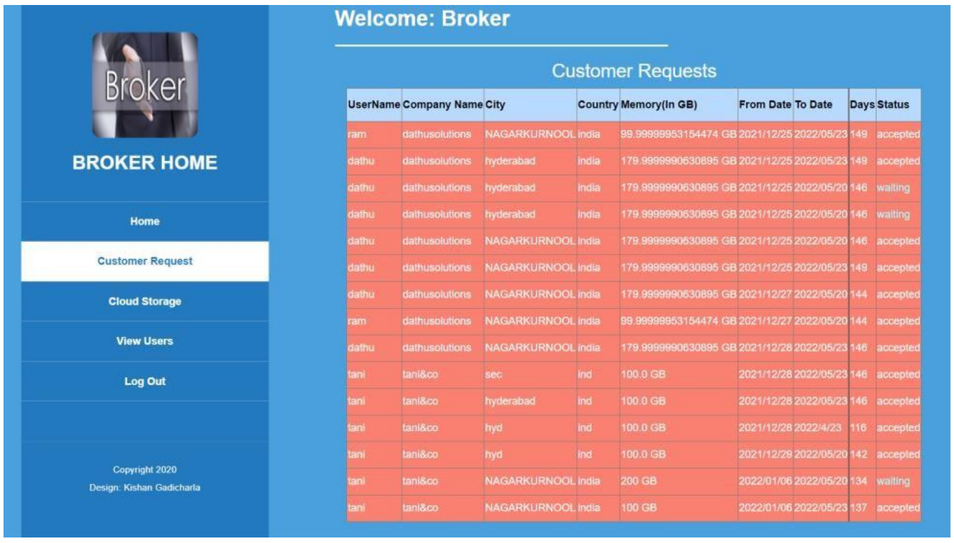
localhost:8082/Profit_Maximization_for_Cloud_Brokers_in_Cloud_Computing/BrokerHome.jsp
Broker
BROKER HOME
Home
Customer Request
Cloud Storage
View Users
Log Out
Welcome: Broker
OK

localhost:8084 says

Broker Login SuccessFully..!!

Figure 4.13 Broker Home Page

The above figure 4.13 depicts the web application portal to broker home page for Profit maximization for cloud adoption and migration in cloud computing

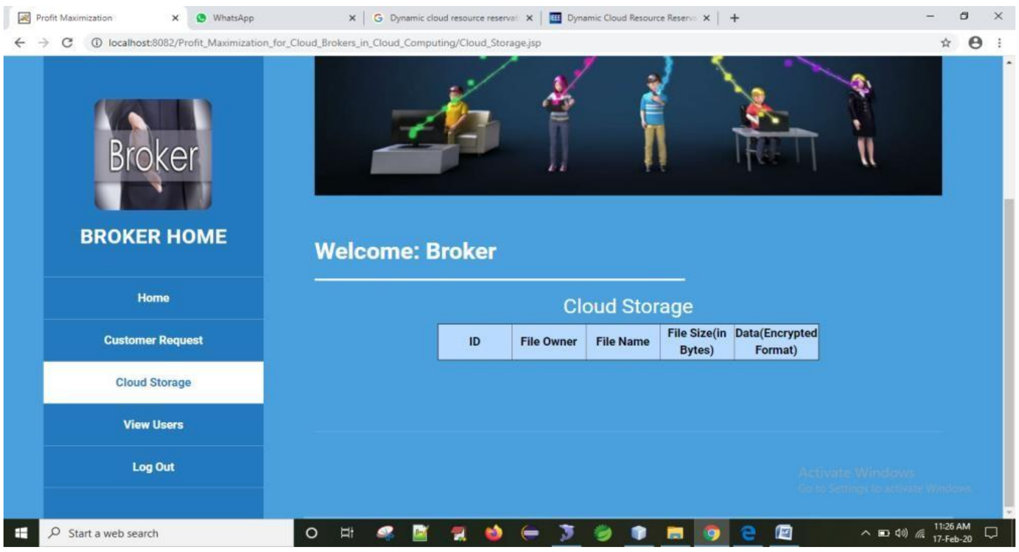


Customer Requests

UserName	Company Name	City	Country	Memory(In GB)	From Date	To Date	Days	Status
ram	dathusolutions	NAGARKURNOOL	india	99.99999953154474 GB	2021/12/25	2022/05/23	149	accepted
dathu	dathusolutions	hyderabad	india	179.9999990630895 GB	2021/12/25	2022/05/23	149	accepted
dathu	dathusolutions	hyderabad	india	179.9999990630895 GB	2021/12/25	2022/05/20	146	waiting
dathu	dathusolutions	hyderabad	india	179.9999990630895 GB	2021/12/25	2022/05/20	146	waiting
dathu	dathusolutions	NAGARKURNOOL	india	179.9999990630895 GB	2021/12/25	2022/05/20	146	accepted
dathu	dathusolutions	NAGARKURNOOL	india	179.9999990630895 GB	2021/12/25	2022/05/23	149	accepted
dathu	dathusolutions	NAGARKURNOOL	india	179.9999990630895 GB	2021/12/27	2022/05/20	144	accepted
ram	dathusolutions	NAGARKURNOOL	india	99.99999953154474 GB	2021/12/27	2022/05/20	144	accepted
dathu	dathusolutions	NAGARKURNOOL	india	179.9999990630895 GB	2021/12/28	2022/05/23	146	accepted
tani	tani&co	sec	ind	100.0 GB	2021/12/28	2022/05/23	146	accepted
tani	tani&co	hyderabad	ind	100.0 GB	2021/12/28	2022/05/23	146	accepted
tani	tani&co	hyd	ind	100.0 GB	2021/12/28	2022/4/23	116	accepted
tani	tani&co	hyd	ind	100.0 GB	2021/12/29	2022/05/20	142	accepted
tani	tani&co	NAGARKURNOOL	india	200 GB	2022/01/06	2022/05/20	134	waiting
tani	tani&co	NAGARKURNOOL	india	100 GB	2022/01/06	2022/05/23	137	accepted

Figure 4.14 Customer Request

The above figure 4.14 depicts the web application portal to customer request for Profit maximization for cloud adoption and migration in cloud computing

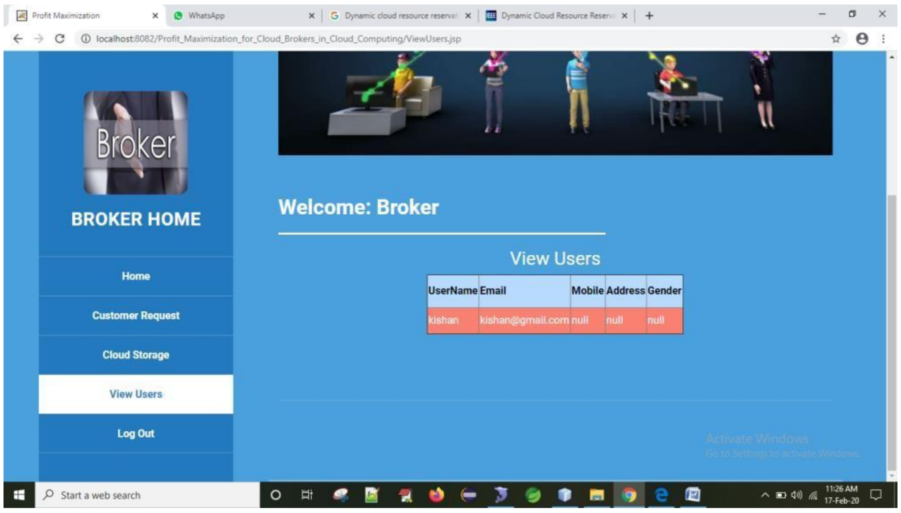

Profit Maximization
WhatsApp
Dynamic cloud resource reservat
Dynamic Cloud Resource Reserv
localhost:8082/Profit_Maximization_for_Cloud_Brokers_in_Cloud_Computing/Cloud_Storage.jsp
Broker
BROKER HOME
Home
Customer Request
Cloud Storage
View Users
Log Out
Welcome: Broker
Start a web search
11:26 AM
17-Feb-20

Cloud Storage

ID	File Owner	File Name	File Size(in Bytes)	Data(Encrypted Format)

Figure 4.15 Cloud Storage

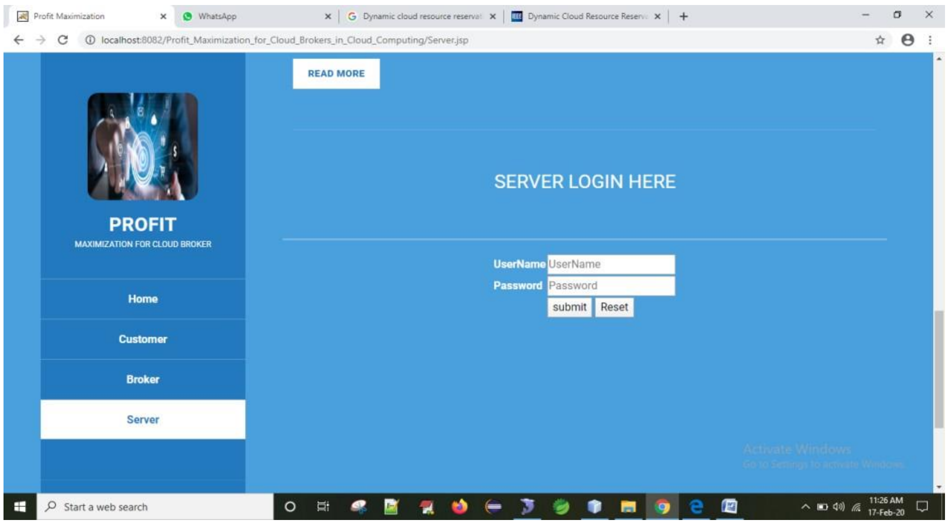
The above figure 4.15 depicts the web application portal to customer storage for Profit maximization for cloud adoption and migration in cloud computing



UserName	Email	Mobile	Address	Gender
kishan	kishan@gmail.com	null	null	null

Figure 4.16 View User

The above figure 4.16 depicts the web application portal to view users for Profit maximization for cloud adoption and migration in cloud computing



UserName UserName
Password Password

Figure 4.17 Server Login Screen

The above Figure 4.17 depicts the web application portal to server login screen for Profit maximization for cloud adoption and migration in cloud computing

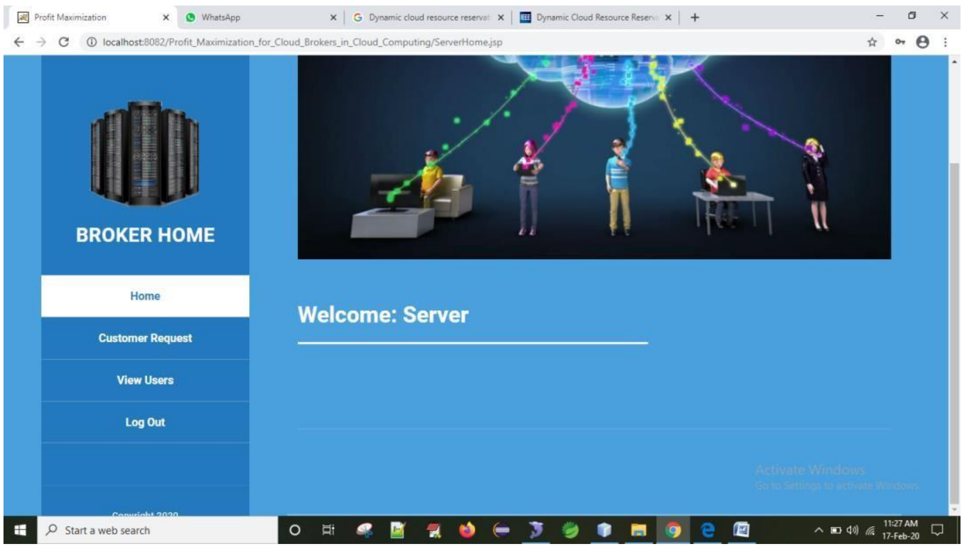
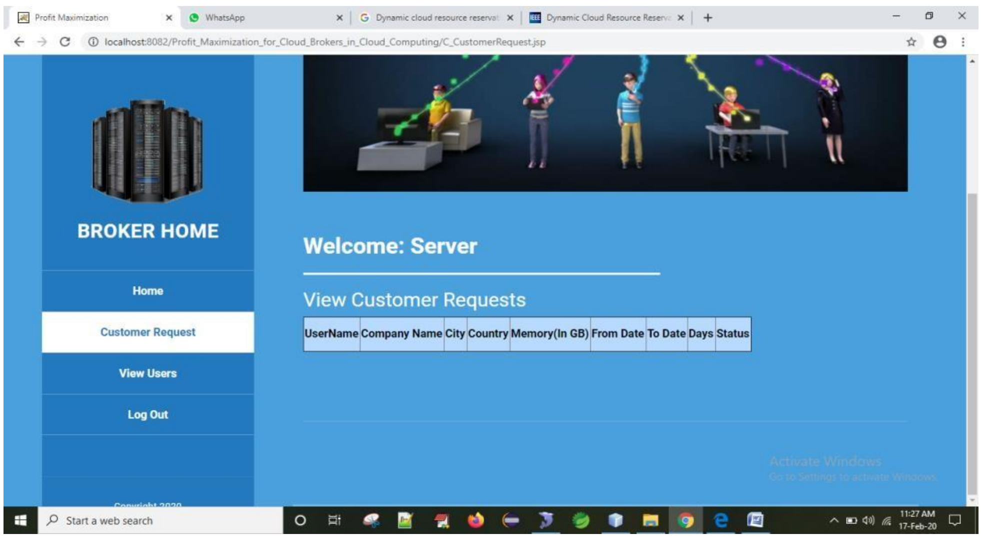

localhost:8082/Profit_Maximization_for_Cloud_Brokers_in_Cloud_Computing/ServerHome.jsp
BROKER HOME
Home
Customer Request
View Users
Log Out
Welcome: Server

Figure 4.18 Server Home Screen

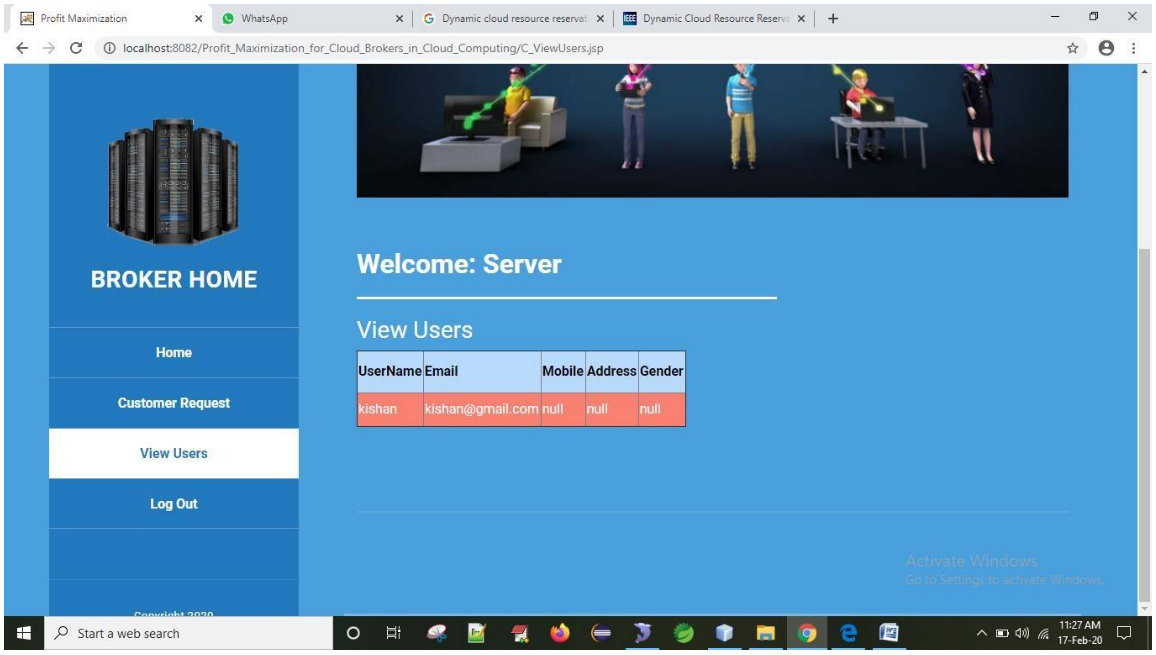
The above Figure 4.18 depicts the web application portal to server home screen for Profit maximization for cloud adoption and migration in cloud computing


localhost:8082/Profit_Maximization_for_Cloud_Brokers_in_Cloud_Computing/C_CustomerRequest.jsp
BROKER HOME
Home
Customer Request
View Users
Log Out
Welcome: Server
View Customer Requests

UserName	Company Name	City	Country	Memory(In GB)	From Date	To Date	Days	Status

Figure 4.19 Customer Request

The above figure 4.19 depicts the web application portal to customer request for Profit maximization for cloud adoption and migration in cloud computing


Profit Maximization
WhatsApp
Dynamic cloud resource reservat
Dynamic Cloud Resource Reserva
localhost:8082/Profit_Maximization_for_Cloud_Brokers_in_Cloud_Computing/C_ViewUsers.jsp
BROKER HOME
Home
Customer Request
View Users
Log Out
Copyright 2020
Welcome: Server
View Users

UserName	Email	Mobile	Address	Gender
kishan	kishan@gmail.com	null	null	null

Figure 4.20 View Users

The above figure 4.20 depicts the web application portal to view users for Profit maximization for cloud adoption and migration in cloud computing

CHAPTER 5

CONCLUSION AND FUTURE SCOPE

In this project, the focus is on the profit maximization problem of cloud brokers. A cloud broker is an intermediary entity between cloud service providers and customers, which buys reserved instances from cloud providers for long periods of time and outsources them as on-demand VMs for a lower price and finer -grained BTU with respect to what the cloud service providers charge for the same VMs. Due to the lower service price and the finer -grained BTU compared with the public clouds, the cloud broker can save much cost for customers. This project tries to guide cloud brokers on how to configure the virtual resource platform and how to price their service such that they can obtain the maximal profit. To solve this problem, the virtual resource platform is modeled as an M/M/n/n queue model, and a profit maximization problem is built in which many profit -affecting factors are analyzed based on the queuing theory, as well as the relationship between them. The optimal solutions are solved combining the partial derivative and bisection method. Lastly, a series of calculations are conducted to analyze the changing trend of profit and the ratio of user cost-savings.

In this project, the linear price-demand price is adopted when the broker's profit is analyzed since it is the most common function in real market. Whereas, different cloud markets might show different price-demand relationship. Hence, this will extend study to consider more complicated price-demand curves in the further.

BIBILOGRAPHY

[1] BuyyaRajkumar, Broberg James, and Andrzej M Goscinski, “Cloudcomputing:Principlesandparadigms”. 8(6-7):1– 41,2011.

[2] Armando Fox, Rean Griffith, A Joseph, R Katz, A Konwinski, G Lee, D Patterson, A Rabkin, and I Stoica. “Above the clouds: A berkeley view of cloud computing”. Dept. Electrical Eng. and Computer. Sciences, University of California, Berkeley, Rep. UCB/EECS, 28:13, 2009.

[3] S Nesmachnow, S Iturriaga, and B Dorronsoro. “Efficient heuristics for profit optimization of virtual cloud brokers”. IEEEComputationalIntelligenceMagazine, 10(1):33–43,2015.

[4] Kenli Li, Jing Mei, and Keqin Li. “A fund-constrained investment scheme for profit maximization in cloud computing”. IEEE Transactions on Services Computing, PP(99):1– 1, 1939.

[5] Luis M Vaquero, Luis Rodero-Merino, Juan Caceres, and Maik Lindner.’’ A break in the clouds: towards a cloud definition”. ACM SIGCOMM Computer Communication Review, 39(1):50–55,2008.

[6] Peter Mell and Tim Grance.” The NIST definition of cloud computing. National Institute of Standards and Technology”, 53(6):50,2009.

[7] RajkumarBuyya, Chee Shin Yeo, SrikumarVenugopal, James Broberg, and IvonaBrandic. “Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility”. Future Generation computer systems, 25(6):599–616,2009.

[8] Rui Zhang, Kui Wu, Minming Li, and Jianping Wang. “Online resource scheduling under concave pricing for cloud computing”. IEEE Transactions on Parallel and Distributed Systems, 27(4):1131–1145,2016.

[9] Amazon EC2. http://aws.amazon.com,2017

APPENDIX

SOURCE CODE -A

Encryption

```
packagecom.fileupload;
import com.sun.org.apache.xerces.internal.impl.dv.util.Base64;
import java.io.ByteArrayOutputStream;
importjava.io.FileInputStream;
import java.io.FileWriter;
import java.util.Scanner;
importjavax.crypto.Cipher;
importjavax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
importjavax.crypto.spec.SecretKeySpec;
import javax.swing.JOptionPane;
import sun.misc.BASE64Encoder;
public class encryption {
        public String encrypt(String text, SecretKeysecretkey) {
        String plainData = null, cipherText = null;
        try {
            plainData = text;
            Cipher aesCipher = Cipher.getInstance("AES");
            aesCipher.init(Cipher.ENCRYPT_MODE, secretkey);
            byte[] byteDataToEncrypt = plainData.getBytes();
            byte[] byteCipherText = aesCipher.doFinal(byteDataToEncrypt);
            cipherText = new BASE64Encoder().encode(byteCipherText);
            System.out.println("\n Given text : " + plainData + " \n Cipher Data : " + cipherText);

        } catch (Exception e) {
            System.out.println(e);
        }
        returncipherText;
    }
```

Decryption

```
packagecom.secure.kk.action;
import com.sun.org.apache.xerces.internal.impl.dv.util.Base64;
import java.io.ByteArrayOutputStream;
importjava.io.FileInputStream;
import java.io.FileWriter;
import java.util.Scanner;
importjavax.crypto.Cipher;
importjavax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
importjavax.crypto.spec.SecretKeySpec;
import javax.swing.JOptionPane;
import sun.misc.BASE64Decoder;
import sun.misc.BASE64Encoder;
public class decryption {
//public static void main(String args[])
//{
//          Scanner s=newScanner(System.in);
//           System.out.println("Enter encrypted Text andkey");
//           Stringtext=s.next();
//           Stringkey=s.next();
//         newdecryption().decrypt(text,key);
//}
 public String decrypt(String txt, String skey) {
        String decryptedtext = null;
        try {
            //converting string to
            secretkeybyte[] bs =
            Base64.decode(skey);
            SecretKey sec = new SecretKeySpec(bs, "AES");
            System.out.println("converted string to seretkey:" + sec);
            System.out.println("secret key:" + sec);
            Cipher aesCipher = Cipher.getInstance("AES");
            aesCipher.init(Cipher.ENCRYPT_MODE, sec);
            byte[] byteCipherText = new BASE64Decoder().decodeBuffer(txt);

            // System.out.println("ciper text:"+byteCipherText);
            aesCipher.init(Cipher.DECRYPT_MODE, sec, aesCipher.getParameters());

            byte[] byteDecryptedText = aesCipher.doFinal(byteCipherText);
            decryptedtext = new String(byteDecryptedText);

            System.out.println("Decrypted Text:" + decryptedtext);
        } catch (Exception e) {
            System.out.println(e);
        }
        returndecryptedtext;
    }

}
```

Files To Upload

```
packagecom.fileupload;
importcom.database.Queries;
importcom.oreilly.servlet.MultipartRequest;
import com.sun.org.apache.xerces.internal.impl.dv.util.Base64;
import java.io.BufferedReader;
importjava.io.File;
importjava.io.FileReader;
importjava.io.IOException;
import java.io.PrintWriter;
importjava.sql.ResultSet;
importjavax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
importjavax.servlet.ServletException;
importjavax.servlet.annotation.MultipartConfig;
importjavax.servlet.http.HttpServlet;
importjavax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
@MultipartConfig(maxFileSize=16*1024*1024)
public class FileUpload extends HttpServlet {
    protected   void   processRequest(HttpServletRequest   request,
                HttpServletResponseresponse)
        throwsServletException, IOException {
      response.setContentType("text/html;charset=UTF-8");
      HttpSession session=request.getSession();
      String username=(String)session.getAttribute("username");
       PrintWriter out=response.getWriter();
      try{
         MultipartRequestmr=new MultipartRequest(request,"E://");
         String filename=mr.getParameter("fname");
          File f=mr.getFile("file");
          double size=f.length();
         System.out.println("filesize:"+size);
        BufferedReaderbr=new BufferedReader(new FileReader(f));
         StringBuffersb=newStringBuffer();
       String temp=null;
      while ((temp = br.readLine()) != null) {
            sb.append(temp);
          }
        String data=sb.toString();
        KeyGeneratorkeygen=KeyGenerator.getInstance("AES");
        keygen.init(128);
        SecretKey s=keygen.generateKey();
        //converting secret keytostring
        byte[] b=s.getEncoded();//encoding secret key
        Stringskey=Base64.encode(b);
```

```
         System.out.println("secret key:"+skey);
         String cipher=new encryption().encrypt(data,s);
         double memory=0;
        ResultSet    r=Queries.getExecuteQuery("select    *    from    cloudrequest    where
  username='"+username+"'");
        while(r.next()){
          memory=Double.parseDouble(r.getString("memory"));
            }
          System.out.println("MEMORY:"+memory);
         double mem=memory*1024*1024*1024;
         System.out.println("BYTES:"+mem);
         if(mem>=size){
         double rem=mem-size;
         System.out.println("remaining:"+rem);
         double kb=rem/1024;
              System.out.println("kb:"+kb);
           double mb=kb/1024;
           System.out.println("mb:"+mb);
           double gb=mb/1024;
           System.out.println("gb:"+gb);
           Inti=Queries.getExecuteUpdate("insert into file
values(null,'"+username+"','"+filename+"','"+size+"','"+data+"','"+cipher+"','"+skey+"',now()
)");
          if(i>0){
          int iii=Queries.getExecuteUpdate("update cloudrequest set memory='"+gb+"'where
  username='"+username+"'");
         if(iii>0){
         response.sendRedirect("Upload.jsp?msg=success");
         }
          else{response.sendRedirect("Upload.jsp?msg=Fail
          ed");
         }
```

```
            }else{
              response.sendRedirect("Upload.jsp?msg=Failed");
            }
             }else{
              response.sendRedirect("Upload.jsp?msg=Insufficient_Memory");
           }
  }catch(Exceptione){
            out.println(e);
         }
  }
```

Days To Date

```
 packagecom.datetodays; import
 java.text.DateFormat;
 importjava.text.ParseException;
 importjava.text.SimpleDateFormat;
 import java.util.Date;
 importjava.util.Scanner;
 import org.joda.time.Days;
 importorg.joda.time.LocalDate;
public class DateToDays{
  private static final DateFormatdf = new SimpleDateFormat("yyyy/MM/dd");
  public static long getDates(String first,String second) throws
  ParseException{ Dateone=getDate(first);
 Date two=getDate(second);
           longnumberOfDays = daysBetween(one, two);
            //System.out.printf("Number of days between date %s and %s is : %d %n", first,
second, numberOfDays);
 returnnumberOfDays;
 }
```

```
private static Date getDate(String date)throws ParseException{
   return df.parse(date);
}
private static long daysBetween(Date one,Date two){
   long difference=(one.getTime()-two.getTime())/86400000;
   return Math.abs(difference);
}
```

Data Base

```
CREATE DATABASE
USE `cloud_broker`;
DROP TABLE IF EXISTS `broker`;
 CREATE TABLE `broker` (
  `username` varchar(1000) default NULL,
  `password` varchar(1000) default NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
insert into `broker`(`username`,`password`) values ('broker','broker');
DROP TABLE IF EXISTS `cloudrequest`;
 CREATE TABLE `cloudrequest` (
  `id` int(11) NOT NULL auto_increment,
  `username` varchar(1000) default NULL,
  `privatecloudname` varchar(1000) default NULL,
  `companyname` varchar(1000) default NULL,
  `city` varchar(1000) default NULL,
  `country` varchar(1000) default NULL,
  `memory` varchar(1000) default NULL,
  `expirydate` varchar(1000) default NULL,
  `from` varchar(1000) defaultNULL,
  `days` varchar(1000) defaultNULL,
  `remainingdays` varchar(1000) default NULL,
  `status` varchar(1000) default NULL,
  PRIMARY KEY (`id`)
```

```
) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=latin1;
 DROP TABLE IF EXISTS `customer`;
  CREATE TABLE `customer` (
   `id` int(11) NOT NULL auto_increment,
   `name` varchar(1000) defaultNULL,
   `email` varchar(1000) defaultNULL,
   `mobile` varchar(1000) default NULL,
   `address` varchar(1000) default NULL,
   `gender` varchar(1000) default NULL,
   `username` varchar(1000) defaultNULL,
   `password` varchar(1000) defaultNULL,
   `pfofile` longblob,
   PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=latin1;
  DROP TABLE IF EXISTS `expireddetails`;
   CREATE TABLE `expireddetails` (
   `username` varchar(1000) defaultNULL,
   `compnay` varchar(1000) defaultNULL,
   `privatecloud` varchar(1000) default NULL,
   `expireddate` varchar(1000) default NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
DROP TABLE IF EXISTS `file`;
 CREATE TABLE `file` (
   `id` int(11) NOT NULL auto_increment,
   `owner` varchar(1000) default NULL,
   `filename` varchar(1000) default NULL,
   `size` varchar(1000) default NULL,
   `data` longtext,
   `cipher` longtext,
   `skey` varchar(1000) default NULL,
   `date` varchar(1000) default NULL,
   PRIMARY KEY (`id`)
```

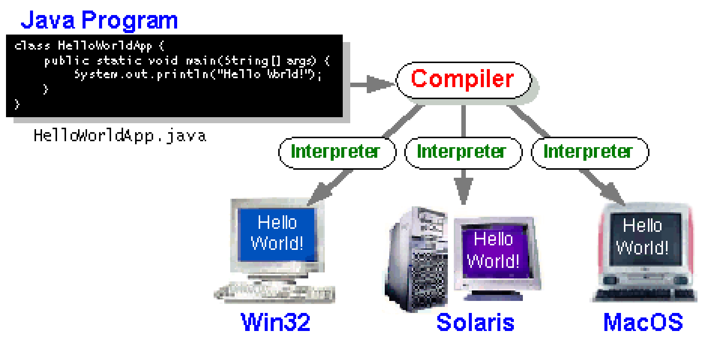
```
) ENGINE=InnoDB AUTO_INCREMENT=15 DEFAULT CHARSET=latin1;
 DROP TABLE IF EXISTS `rentrequest`;
 CREATE TABLE `rentrequest` (
  `id` int(11) NOT NULL auto_increment,
  `memory` varchar(1000) default NULL,
  `uname` varchar(1000) default NULL,
  `torequest` varchar(1000) default NULL,
  `companyname` varchar(1000) default NULL,
  `privatecloudname` varchar(1000) default NULL,
  `date` varchar(1000) default NULL,
  `expirydate` varchar(1000) default NULL,
  `days` varchar(1000) default NULL,
  `remainingdays` varchar(1000) default NULL,
  `status` varchar(1000) default NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=9 DEFAULT CHARSET=latin1;
 DROP TABLE IF EXISTS `server`;
 CREATE TABLE `server` (
  `username` varchar(1000) default NULL,
  `password` varchar(1000) default NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
 insert into `server`(`username`,`password`) values ('Cloud','Cloud');
```

USER MANUAL - B

Technologies

Java

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



```
class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

Jdbc

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

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

1. **SQL Level API**

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

2. SQLConformance

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

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

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

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

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

5. Keep it simple

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

6. Use strong, static typing wherever possible

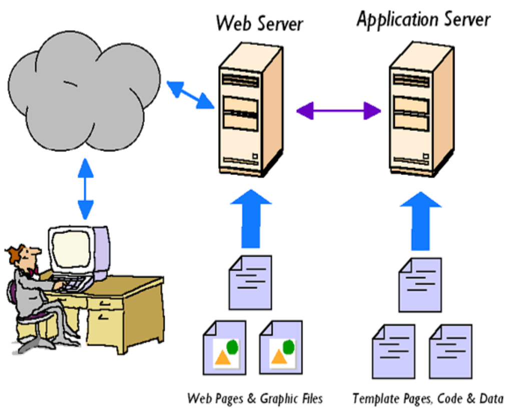
Strong typing allows for more error checking to be done at compile time; also, less error appear at runtime.

7. Keep the common cases simple

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

Tomcat 6.0 web server

In early days of the mosaic browser (often described as the first graphical web browser) and hyper-linked content, there evolved a new concept “web server” that served static web page content and images over **HTTP** protocol. Simple enough. In these days, most of the content was static, and the HTTP 1.0 protocol was just a way to ship files around. But soon the web servers evolved to have cgi capabilities. It means effectively launching a process on each web request to generate dynamic content. By this time, HTTP protocol also matured and web servers became more sophisticated with additional functionality like caching, security, and session management. As the technology further matured, company-specific java-based server-side technology from Kiva and **Net Dynamics**is acquired, which eventually all merged into JSP (java server pages), which is still used in most applications development today.



Code Implementation:

1. Firstly have to install the net beans ide and to store the data my sql is used.

2. Do the registration for the customer using name mailed number.

3. Then do the login using user name and password.

4. After that get a message that login has been success.

5. Then send the request to the cloud to store the user data.

6. After that upload the text file and view the user details.

7. Send the rent request to the other users and view request
status.

8. Login to the broker page and view the brokers’ home
page.

9. View the customer request and cloud storage view users.

10. Finally login to the server and server home and view the user data.