

An Industry Oriented Project Report on

“PROFIT MAXIMIZATION FOR CLOUD BROKERS IN CLOUD COMPUTING”

Submitted to the

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD

In partial fulfilment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY IN

COMPUTER SCIENCE AND ENGINEERING

BY

B. VENNELA (18WJ1A0552)

ALASYAM SUSHMA SRI (18WJ1A0514)

BHAVANA REDDY MARAM (18WJ1A0549)

Under the Esteemed Guidance

Of Mr. B. LALU,

Associate Professor, CSE Dept.



GURU NANAK INSTITUTIONS TECHNICAL CAMPUS(AUTONOMUS)

School of Engineering and Technology

Ibrahimpattanam, R.R District-501506

2021-2022



Department of Computer Science and Engineering

CERTIFICATE

This is to certify that this project report entitled “**PROFIT MAXIMIZATION FOR CLOUD BROKERS IN CLOUD COMPUTING**” by **ALASAYAM SUSHMA SRI (18WJ1A0514), BHAVANA REDDY MARAM (18WJ1A0549), BHUSANI VENNELA (18WJ1A0552)** submitted in partial fulfilment of the requirements for the degree of **Bachelor of Technology in Computer Science and Engineering** of the **Jawaharlal Nehru Technological University Hyderabad** during the academic year 2021-2022, is a bona fide record of work carried out under our guidance and supervision.

INTERNAL GUIDE

Mr. B. LALU

PROJECT CO-ORDINATOR

Mr. SAMIRANA ACHARYA

HOD CSE

Dr. J. RAJESHWAR

EXTERNALEXAMINER

**RAM Innovative Infotech**

M : +91 9581 012 012

E : raminnovativeinfotech@gmail.com

Flat No.#309, Amrutha Ville,
Opp: Yashoda Hospital, Somajiguda,
Hyderabad-82, Telangana, India

www.raminnovativeinfotech.webs.com

PROJECT COMPLETION CERTIFICATE

This is to certify that the following students of final year B.Tech, Department of Computer Science Engineering - Guru Nanak Institutions Technical Campus (GNITC) have completed their training and project at GNITC successfully.

STUDENT NAME:
ALASYAM SUSHMA**ROLL NO:**

1. SRIBHAVANA MARAM
2. BHUSANI VENNELA
3. _____
4. _____

18WJ1A051418WJ1A054918WJ1A0552

The training was conducted on JAVA Technology for the completion of the project titled “PROFIT MAXIMIZATION FOR CLOUD BROKERS IN CLOUD COMPUTING”

in RAM INNOVATIVE INFOTECH. The project has been completed in all aspects.

ACKNOWLEDGEMENT

We wish to express our sincere thanks to **Dr. Rishi Sayal**, Associate Director, GNITC for providing us with the conducive environment for carrying through our academic schedules and Project with ease.

We would like to say our sincere thanks to **Dr. J. Rajeshwar, Head of the Department of CSE**, GNITC for providing seamless support and right suggestions are given in the development of the project.

We would like to say our sincere thanks to **Mr. Samirana Acharya, Assistant. Professor**, Department of CSE, Minor Project Coordinator, for providing seamless support and right suggestions are given in the development of the project.

We have been truly blessed to have a wonderful internal guide **Mr. B. Lalu, Associate. Professor** of CSE, GNITC for guiding us to explore the ramification of our work and we express our sincere gratitude towards him for leading us through the completion of Project.

Finally, we would like to thank our family members for their moral support and encouragement to achieve goals.

ALASYAM SUSHMA SRI (18WJ1A0514)

BHAVANA REDDY MARAM (18WJ1A0549)

BHUSANI VENNELA (18WJ1A0552)

PROFIT MAXIMIZATION FOR CLOUD BROKERS IN CLOUD COMPUTING.

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, we introduce a new role, 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 paper, we focus 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. In this paper, we firstly give a synthetically analysis on all the affecting factors, and define an optimal multiserver configuration and VM pricing problem which is modeled as a profit maximization problem. Secondly, combining the partial derivative and bisection search method, we propose a heuristic method 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.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	i
	LIST OF FIGURES	v
	LIST OF SYMBOLS	vii
	LIST OF ABBREVIATIONS	xi
	LIST OF TABLES	xii
1.	CHAPTER 1 : INTRODUCTION	
	1.1 GENERAL	1
	1.2 OBJECTIVE	2
	1.3 EXISTING SYSTEM	2
	1.3.1EXISTINGSYSTEM DISADVANTAGES	2
	1.3.2 LITERATURE SURVEY	3-8
	1.4 PROPOSED SYSTEM	9
	1.4.1 PROPOSED SYSTEM ADVANTAGES	9
2.	CHAPTER 2 :PROJECT DESCRIPTION	
	2.1 GENERAL	10
	2.2 METHODOLOGIES	10
	2.2.1 MODULES NAME	10
	2.2.2 MODULES EXPLANATION	10-13
	2.2.3 MODULE DIAGRAM	
	2.2.4GIVEN INPUTAND EXPECTED OUTPUT	14
	2.3 TECHNIQUE OR ALGORITHM	15
3.	CHAPTER 3 : REQUIREMENTS	
	3.1 GENERAL	16
	3.2 HARDWARE REQUIREMENTS	16
	3.3 SOFTWARE REQUIREMENTS	17
4.	CHAPTER 4 :SYSTEM DESIGN	
	4.1 GENERAL	19
	4.1.1 USE CASE DIAGRAM	19
	4.1.2 CLASS DIAGRAM	20

	4.1.3 OBJECT DIAGRAM 4.1.4 COMPONENT DIAGRAM 4.1.5 DEPLOYMENT DIAGRAM 4.1.6 SEQUENCE DIAGRAM 4.1.7 COLLABORATION DIAGRAM 4.1.8 STATE DIAGRAM 4.1.9 ACTIVITY DIAGRAM 4.1.10 DATAFLOW DIAGRAM 4.1.11 E-R DIAGRAM 4.1.12 SYSTEM ARCHITECTURE	21 22 23 24 25 26 27 28-30 31 32
5.	CHAPTER 5 :SOFTWARE SPECIFICATION 5.1 GENERAL FRONT END 5.2 FEATURES OF JAVA 5.2.1 THE JAVA FRAMEWORK 5.2.2 OBJECTIVE OF JAVA 5.2.3 JAVA SWING OVERVIEW 5.2.4 EVOLUTION OF COLLECTION FRAMEWORK 5.3 CONCLUSION	34 34 34 35 37 40
6.	CHAPTER 6 :IMPLEMENTATION 6.1 GENERAL	40-68

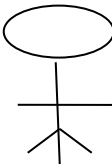
--	--	--

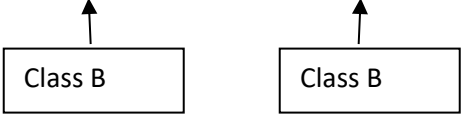
7.	CHAPTER 7 :SNAPSHOTS 7.1 GENERAL 7.2 VARIOUS SNAPSHOTS	69 69-77
8.	CHAPTER 8 :SOFTWARE TESTING 8.1 GENERAL 8.2 DEVELOPING METHODOLOGIES 8.3 TYPES OF TESTING	78 78 78-80
9.	CHAPTER 9 : APPLICATIONS AND FUTURE ENHANCEMENT 9 GENERAL	81
10	CHAPTER 10 : 10.1CONCLUSION 10.2 REFERENCES	82-83 83

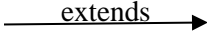


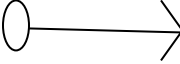
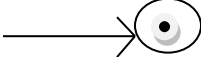
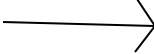
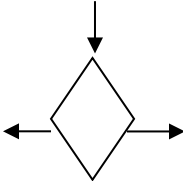
LIST OF FIGURES

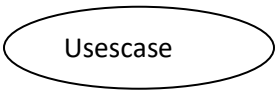
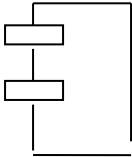
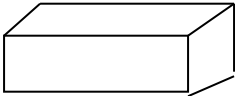
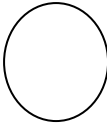
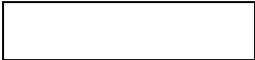

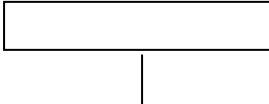
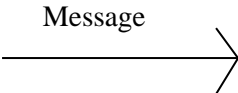
FIGURE NO	NAME OF THE FIGURE	PAGE NO.
4.1	System architecture	32
4.2	Activity Diagram	27
4.3	Use case Diagram	19
4.4	Data flow diagram	28-30
4.5	Sequence diagram	24
4.6	Collaboration diagram	25
4.7	Class diagram	20
4.8	Activity Diagram	27
4.9	State Diagram	26
4.1	Component Diagram	22
4.12	E-R Diagram	31

LIST OF SYMBOLS

S.NO	NOTATION NAME	NOTATION	DESCRIPTION
1.	Class	<div style="display: flex; justify-content: space-around; align-items: center;"> <div style="border: 1px solid black; padding: 5px; text-align: center;"> <i>+ public</i> <i>-private</i> </div> <div style="border: 1px solid black; padding: 5px; text-align: center;"> <i>Class Name</i> <i>-attribute</i> <i>-attribute</i> </div> </div>	Represents a collection of similar entities grouped together.
2.	Association	<div style="display: flex; justify-content: space-around; align-items: center;"> <div style="border: 1px solid black; padding: 5px;">Class A</div> <div style="border: 1px solid black; padding: 5px;">NAME</div> <div style="border: 1px solid black; padding: 5px;">Class B</div> </div> <div style="display: flex; justify-content: space-around; align-items: center; margin-top: 10px;"> <div style="border: 1px solid black; padding: 5px;">Class A</div> <div style="border: 1px solid black; padding: 5px;">Class B</div> </div>	Associations represents static relationships between classes. Roles represents the way the two classes see each other.
3.	Actor		It aggregates several classes into a single classes.
		<div style="display: flex; justify-content: space-around; align-items: center;"> <div style="border: 1px solid black; padding: 5px;">Class A</div> <div style="border: 1px solid black; padding: 5px;">Class A</div> </div>	

4.	Aggregation		Interaction between the system and external environment
----	-------------	--	---

5.	Relation (uses)	uses	Used for additional process communication.
6.	Relation (extends)		Extends relationship is used when one use case is similar to another use case but does a bit more.
7.	Communication		Communication between various use cases.
8.	State		State of the process.
9.	Initial State		Initial state of the object
10.	Final state		Final state of the object
11.	Control flow		Represents various control flow between the states.
12.	Decision box		Represents decision making process from a constraint

13.	Usecase		Interact ion between the system and external environment.
14.	Component		Represents physical modules which are a collection of components.
15.	Node		Represents physical modules which are a collection of components.
16.	Data Process/State		A circle in DFD represents a state or process which has been triggered due to some event or acion.
17.	External entity		Represents external entities such as keyboard,sensors,etc.
18.	Transition		Represents communication that occurs between processes.
19.	Object Lifeline		Represents the vertical dimensions that the object communications.
20.	Message		Represents the message exchanged.

LIST OF ABBREVIATION

S.NO	ABBREVIATION	EXPANSION
1.	DB	DataBase
2.	JVM	Java Virtual Machine
3.	JSP	Java Server Page
4.	PWS	Personalised Web Search
5.	UPS	User Personalised Search
6.	JRE	Java Runtime Environment

CHAPTER 1

INTRODUCTION

1.1 GENERAL

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. 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-allocated at any time, 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 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. 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 . This leads to a waste of resources and raises the cost of customers to a certain degree. In addition, almost all cloud providers provide two main ways to pay for their instances: On-Demand and Reserved Instances . With On-Demand instances, users pay for compute capacity by per hour depending on which instances they run, and they are recommended for the applications with short-term workloads. Reserved Instances provide users with a significant discount (up to 75% in Amazon EC2) compared to On-Demand instance pricing, but customers should rent instances for long periods, e.g., from six months to several years, according to the current plans offered by real cloud providers such as Amazon and Microsoft Azure . Obviously, this discount cannot be enjoyed by the short-term customers.

1.2 OBJECTIVE

The objective of this project to 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

1.3 EXISTING SYSTEM

- An important feature of cloud computing is pay-as-you-go.
- However, in exiting system 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.

EXISTING SYSTEM DISADVANTAGES

- Service performances, total cost, and load balancing are not addressed.
- Proper allocation of the requested services is not guaranteed.

1.4 LITERATURE SURVEY

TITLE :Maximizing profit of cloud brokers under quantized billing cycles: a dynamic pricing strategy based on ski-rental problem

AUTHOR :Gourav Saha and Ramkrishna Pasumathy.

YEAR :2015

DESCRIPTION :

In cloud computing, users scale their resources (computational) based on their need. There is massive literature dealing with such resource scaling algorithms. These works ignore a fundamental constraint imposed by all Cloud Service Providers (CSP), i.e. one has to pay for a fixed minimum duration irrespective of their usage. Such quantization in billing cycles poses a problem for users with sporadic workload. In recent literature, Cloud Broker (CB) has been introduced for the benefit of such users. A CB rents resources from CSP and in turn provides service to users to generate profit. Contract between CB and user is that of pay-what-you-use/pay-per-use. However CB faces the challenge of Quantized Billing Cycles as it negotiates with CSP. We design two algorithms, one fully online and the other partially online, which maximize the profit of the CB. The key idea is to regulate user demand using dynamic pricing. Our algorithm is inspired by the Ski-Rental problem. We derive competitive ratio of these algorithms and also conduct simulations using real world traces to prove the efficiency of our algorithm.

TITLE :Cloud brokeringarchitecture for dynamic placement of virtual machines.

AUTHOR :Dheeraj Rane , Abhishek Srivastava

YEAR :2015

DESCRIPTION :

Numerous service providers in the cloud market provide diverse services, at distinct price and level of competence. However, the concept of cloud computing is yet to reach many organizations and to the end users as well. With perspective of these organizations and end users, it is very difficult to choose one among many service providers. Further, adhering to only one provider will deprive these users of benefits from market diversity and fluctuating rates. Instead, having a broker be aware of the provider's service offerings and consumer's requirements, and managing the assignment under economic and technical constraints will benefit both ends of the service viz consumers and providers. In this work, a cloud resource broker is proposed that will govern the assignment of providers' resources to consumer dynamically. The proposed broker uses various requirements and constraints specified by the consumer in the requirement description template as input, to calculate aggregated requirements, using an aggregation algorithm. Further, the service scheduling algorithm is defined to find an optimized match between the aggregated requirements with the providers offerings. Thereafter, this algorithm is executed frequently, based on a strategy for dynamic scheduling to benefit consumers on account of introduction of new provider or some good offerings. Results show that the solution provided by broker proves to be a win-win situation for the consumer with respect to cost as well as performance.

TITLE :Green cloud broker:On-line dynamic virtual machine placement across multiplecloud providers.

AUTHOR :Federico Larumbe and Brunilde Sans.

YEAR :2016

DESCRIPTION :

Cloud computing controls the increasing energy consumption of applications by consolidating them in shared servers through virtualization. This technique can be greatly improved and complemented by choosing an optimal data center for each Virtual Machine (VM). That dynamic optimization problem was stated as a Mixed Integer Linear Programming (MILP) model that minimizes operational expenditures, while respecting constraints on Quality of Service (QoS), power consumption, and CO₂ emissions. Geographically distributed users experience varying response times depending on where users and VMs are located. Users closer to VMs experience a shorter response time, thus distributing VMs close to users improves the QoS. On the other hand, the increasing energy consumption of the cloud raised concerns about the impact on CO₂ emissions and global warming. Placing VMs in data centers that use green energy sources is an important way to mitigate this problem. A comprehensive optimization modelling framework and an efficient tabu search heuristic were developed to handle applications with dynamic demands in real time. Test cases had more than 1,000 nodes, 650 applications, and 6,500 VMs. Results report that the communication delay is reduced up to 6 times, 30% of power consumption is saved, and the CO₂ emissions can be reduced up to 60 times. The framework also allows to carefully assess the trade-offs between delay, cost, CO₂ emissions and power consumption.

TITLE :Online resource scheduling under concave pricing for cloud computing.

AUTHOR :Rui Zhang, Kui Wu, Minming Li, and Jianping Wang.

YEAR :2016

DESCRIPTION :

With the booming growth of cloud computing industry, computational resources are readily and elastically available to the customers. In order to attract customers with various demands, most Infrastructure-as-a-service (IaaS) cloud service providers offer several pricing strategies such as pay as you go, pay less per unit when you use more (so called volume discount), and pay even less when you reserve. The diverse pricing schemes among different IaaS service providers or even in the same provider form a complex economic landscape that nurtures the market of cloud brokers. By strategically scheduling multiple customers' resource requests, a cloud broker can fully take advantage of the discounts offered by cloud service providers. In this paper, we focus on how a broker may help a group of customers to fully utilize the volume discount pricing strategy offered by cloud service providers through cost-efficient online resource scheduling. We present a randomized online stack-centric scheduling algorithm (ROSA) and theoretically prove the lower bound of its competitive ratio. Our simulation shows that ROSA achieves a competitive ratio close to the theoretical lower bound under a special case cost function. Trace driven simulation using Google cluster data demonstrates that ROSA is superior to the conventional online scheduling algorithms in terms of cost saving.

TITLE :Efficientheuristics for profit optimization of virtual cloud brokers.

AUTHOR :S Nesmachnow, S Iturriaga, and B Dorransoro.

YEAR :2015

DESCRIPTION :

This article introduces a new kind of broker for cloud computing, whose business relies on outsourcing virtual machines (VMs) to its customers. More specifically, the broker owns a number of reserved instances of different VMs from several cloud providers and offers them to its customers in an on-demand basis, at cheaper prices than those of the cloud providers. The essence of the business resides in the large difference in price between on-demand and reserved VMs. We define the Virtual Machine Planning Problem, an optimization problem to maximize the profit of the broker. We also propose a number of efficient smart heuristics (seven two-phase list scheduling heuristics and a reordering local search) to allocate a set of VM requests from customers into the available pre-booked ones, that maximize the broker earnings. We perform experimental evaluation to analyze the profit and quality of service metrics for the resulting planning, including a set of 400 problem instances that account for realistic workloads and scenarios using real data from cloud providers.

TITLE :A variable service broker routing policy for datacenter selection in cloud analyst

AUTHOR :Ahmad M. Manasrah, Tariq Smadi, and Ammar Almomani

YEAR :2016

DESCRIPTION

Cloud computing depends on sharing distributed computing resources to handle different services such as servers, storage and applications. The applications and infrastructures are provided as pay per use services through data center to the end user. The data centers are located at different geographic locations. However, these data centers can get overloaded with the increase number of client applications being serviced at the same time and location; this will degrade the overall QoS of the distributed services. Since different user applications may require different configuration and requirements, measuring the user applications performance of various resources is challenging. The service provider cannot make decisions for the right level of resources. Therefore, we propose a Variable Service Broker Routing Policy-VSBRP, which is a heuristic-based technique that aims to achieve minimum response time through considering the communication channel bandwidth, latency and the size of the job. The proposed service broker policy will also reduce the overloading of the data centers by redirecting the user requests to the next data center that yields better response and processing time. The simulation shows promising results in terms of response and processing time compared to other known broker policies from the literature.

1.4 PROPOSED SYSTEM

- To reduce the cost of cloud users, in this paper we introduce a new role, 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

PROPOSED SYSTEM ADVANTAGE

- Lower service price and the finer-grained output.
- Guides cloud brokers on how to configure the resource platform properly

CHAPTER 2

PROJECT DESCRIPTION

2.1 GENERAL

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-allocated at any time, and customers should pay based on how long the resources are actually used.

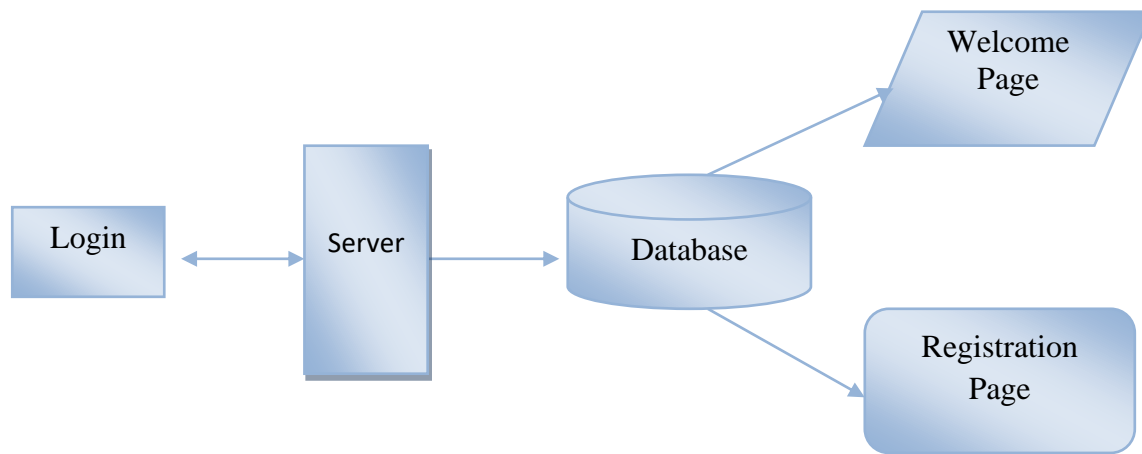
2.2 METHODOLOGIES

2.2.1 MODULES NAME:

1. User Interface Design
2. Admin
3. Cloud
4. Virtual Machine
5. User

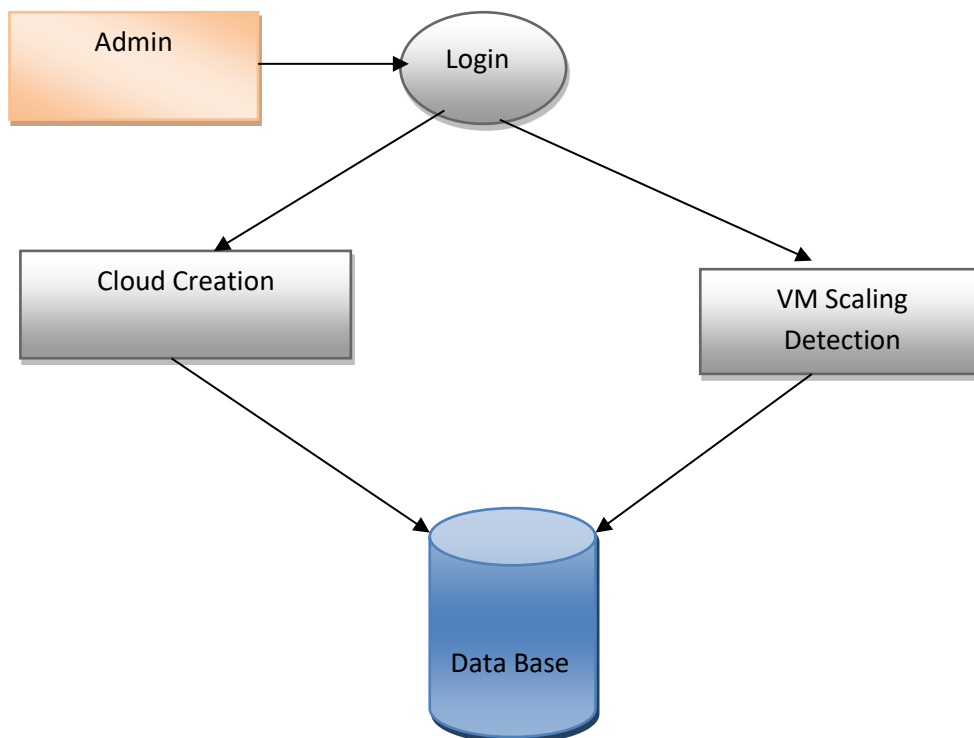
1. User Interface Design

In this module we design the windows for the project. These windows are used for secure login for all users. To connect with server user must give their username and password then only they can be able to connect the server. If the user already exists directly can login into the server else user must register their details such as username, password and Email id, into the server. Server will create the account for the entire user to maintain upload and download rate. Name will be set as user id. Logging in is usually used to enter a specific page.



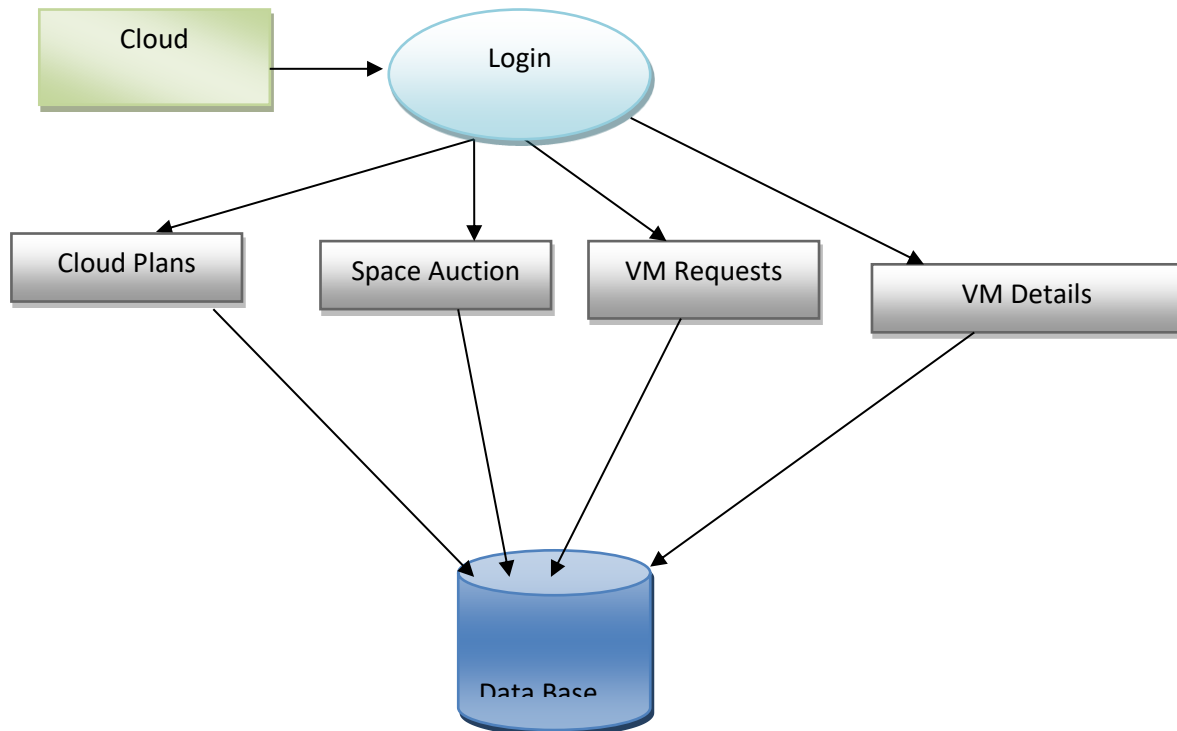
2. Admin

This is the first module of this project. In this module initially admin need to login. Then admin will create the cloud with some space. And he will manage the cloud details like how many VM instances create to a cloud and how many resources that are providing by the cloud. And admin will manage the VM scaling details. That means how many users are available and how they are utilizing the VM.



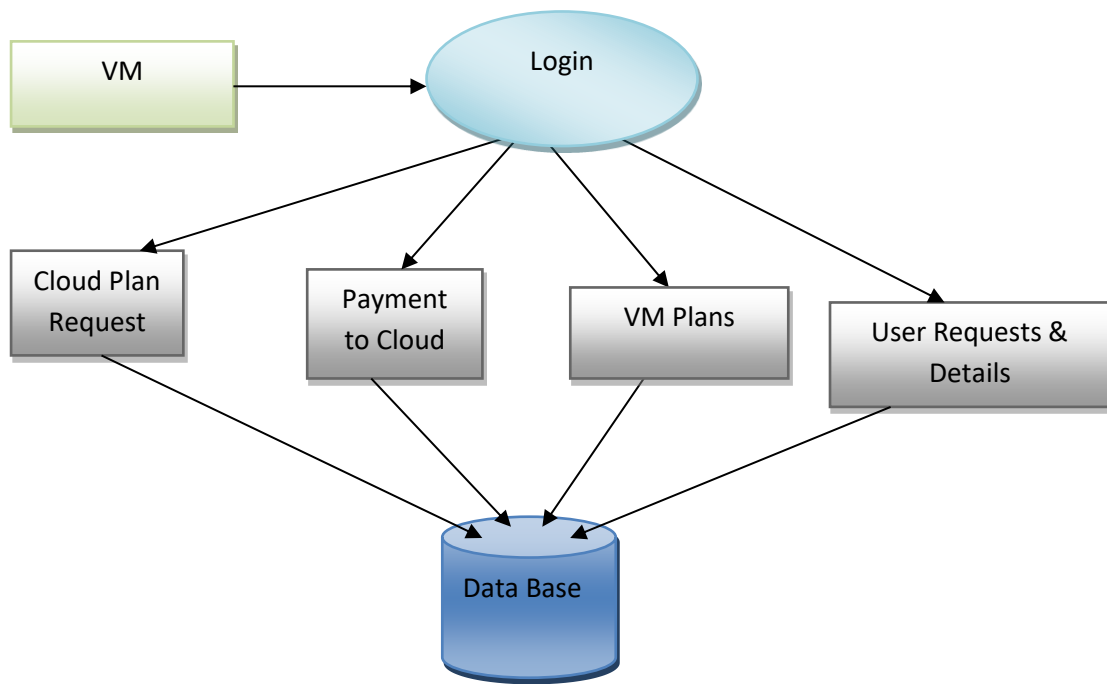
3. Cloud

This is the second module of this project. In this module initially cloud owner need to login. Then he will verify the VM requests to allocate the space and resources. If any VM user wants upgrade their plans that will approval. And online auction for increasing the space of VM between different users and then who will pay more amounts then space will allocate those VM users. And VM can be removed if the expiry date completed to utilize the space in that particular cloud.



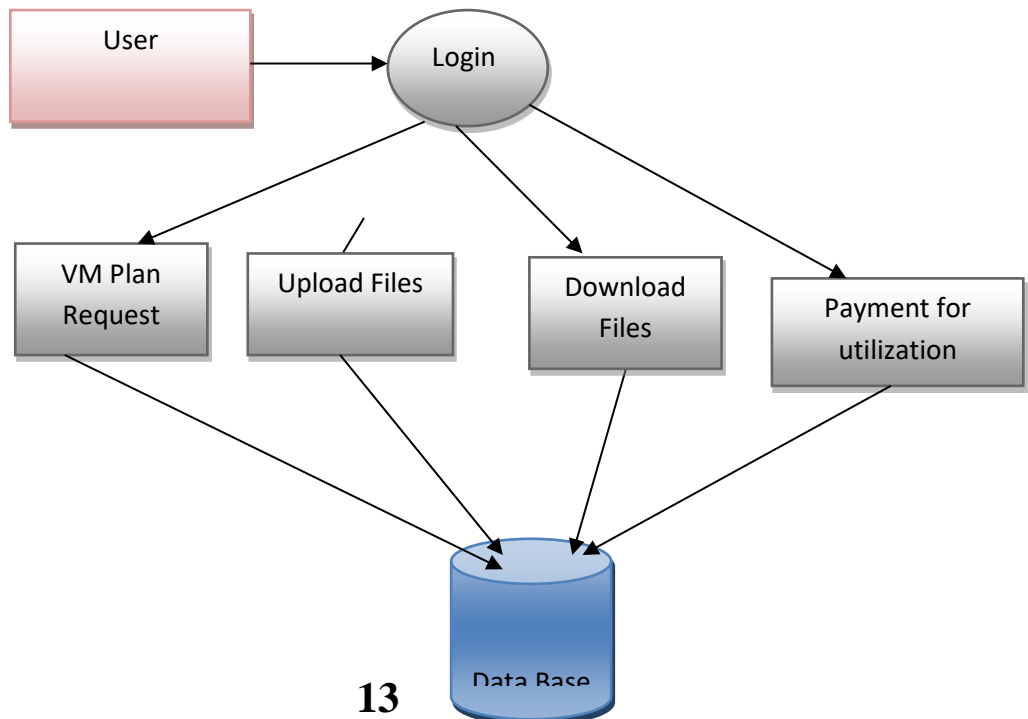
4. Virtual Machine

This is the third module of this project. In this module initially VM owner need to login. Then he will verify the request of resource from different users while they are registering. If he approved then user can utilize the space by paying the money based on his utilization. If the space or expiry date of VM occurs it need to reclaim his space by online auction. And any user bills need to maintain. And resource up gradation needs to maintain.



5. User

This is the fourth module in this project. In this module initially user need to login. Then user can share store the data based on the size. Depending the resource user need to pay the amount to the VM Owner. If user wants to upgrade his resource he can upgrade by paying required amount. If user wants change the VM resource he can his resource to other VM which will provide better resources.



GIVEN INPUT EXPECTED OUTPUT:

➤ User Interface Design

Input : Enter Login name and Password

Output : If valid user name and password then directly open the home page otherwise show error message and redirect to the registration page.

➤ Admin

Input : Admin Login name and Password

Output: If valid user name and password then directly open the admin home page otherwise show error message and redirect to the admin login page.

➤ Cloud

Input : Enter the user name and password

Output : If valid user name and password then directly open the cloud home page otherwise show error message and redirect to the cloud login page.

➤ Virtual Machine

Input : Enter the user name and password

Output : If valid user name and password then directly open the VM home page otherwise show error message and redirect to the VM login page.

➤ **User**

Input : Enter the user name and password

Output: If valid user name and password then directly open the user home page otherwise show error message and redirect to the user login page.

2.3 TECHNIQUE USED OR ALGORITHM USED

M/M/n/n queue model

In this M/M/n/n queuing model, the arrival of VM requests is assumed to be a Poisson stream with arrival rate λ (measured by the number of requests per unit of time).

CHAPTER 3

REQUIREMENTS ENGINEERING

3.1 GENERAL

We have conducted experiments on our collected dataset and extensive results have demonstrated that our model outperforms all other existing models. In the future, we will investigate more tasks under this framework, such as event summarization and event attribute mining in social media.

3.2 HARDWARE REQUIREMENTS

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

- PROCESSOR : DUAL CORE 2 DUOS.
- RAM : 2GB DD RAM
- HARD DISK : 250 GB

3.3 SOFTWARE REQUIREMENTS

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

SOFTWARE REQUIREMENTS

- FRONT END : J2EE (JSP, SERVLET)

- BACK END : MY SQL 5.5
- OPERATING SYSTEM : WINDOWS 7
- IDE : ECLIPSE

3.4 FUNCTIONAL REQUIREMENTS

A functional requirement defines a function of a software-system or its component. A function is described as a set of inputs, the behavior, Firstly, the system is the first that achieves the standard notion of semantic security for data confidentiality in attribute-based deduplication systems by resorting to the hybrid cloud architecture.

3.5 NON-FUNCTIONAL REQUIREMENTS

EFFICIENCY

Our multi-modal event tracking and evolution framework is suitable for multimedia documents from various social media platforms, which can not only effectively capture their multi-modal topics, but also obtain the evolutionary trends of social events and generate effective event summary details over time. Our proposed mmETM model can exploit the multi-modal property of social event, which can effectively model social media documents including long text with related images and learn the correlations between textual and visual modalities to separate the visual-representative topics and non-visual-representative topics.

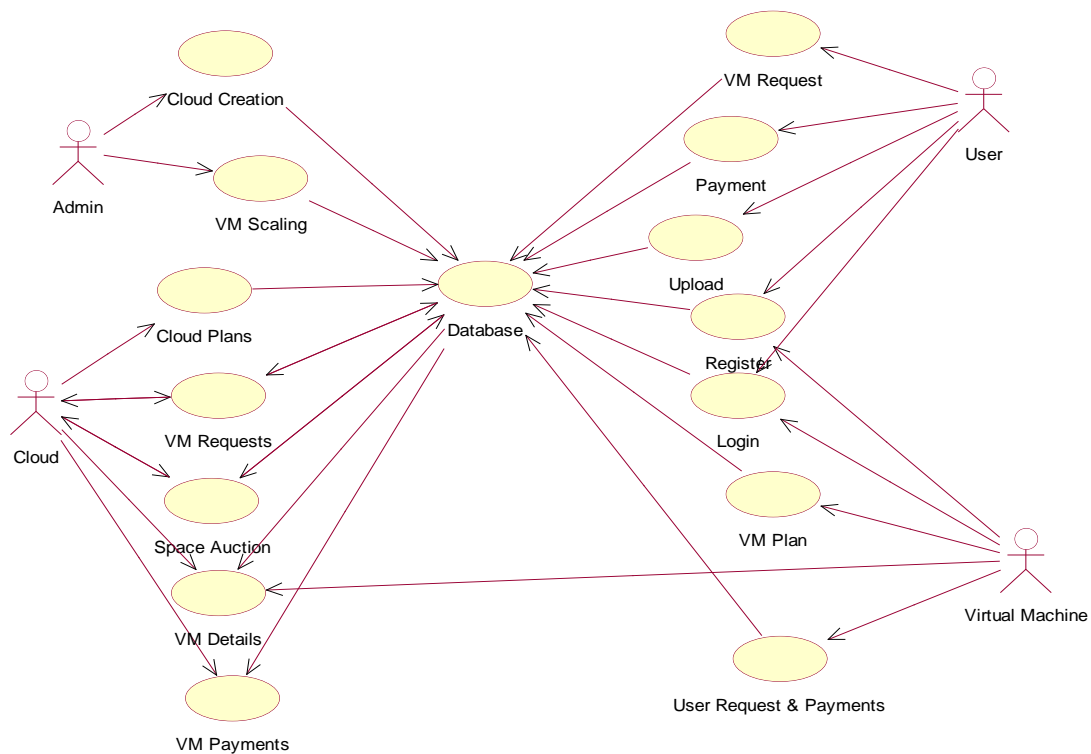
CHAPTER 4

DESIGN ENGINEERING

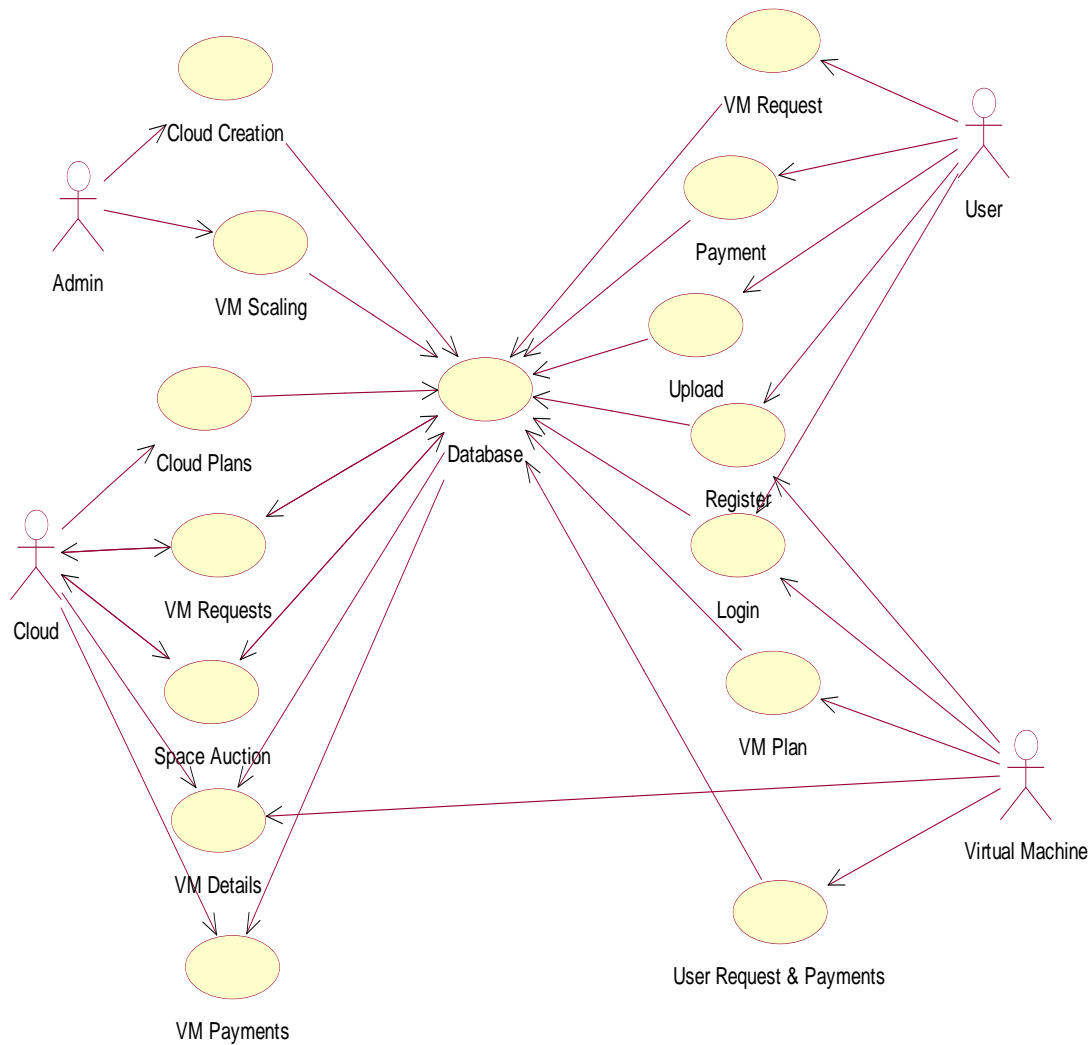
4.1 GENERAL

Design Engineering deals with the various UML [Unified Modeling language] diagrams for the implementation of project. Design is a meaningful engineering representation of a thing that is to be built. Software design is a process through which the requirements are translated into representation of the software. Design is the place where quality is rendered in software engineering. Design is the means to accurately translate customer requirements into finished product.

UMLDIAGRAMS



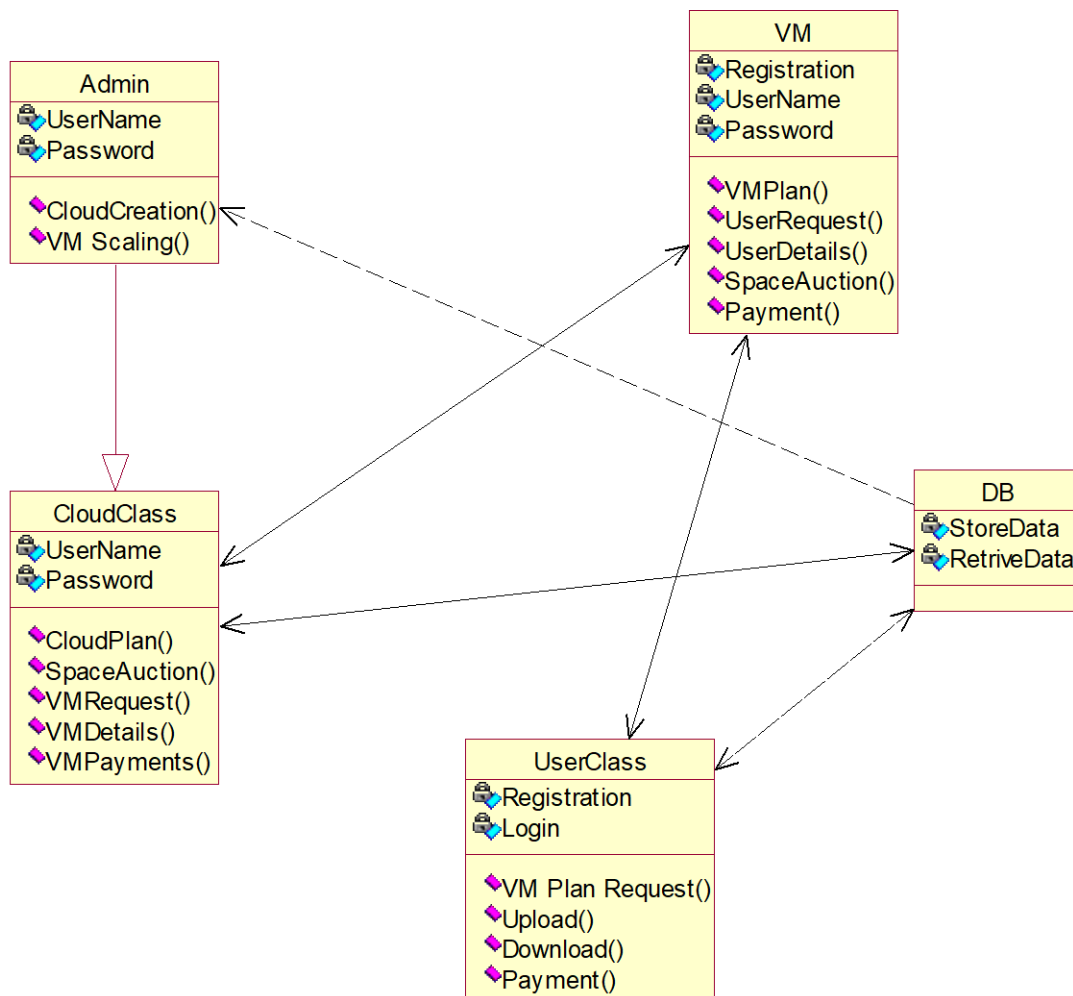
USE CASE DIAGRAM



EXPLANATION

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. The above diagram consists of user as actor. Each will play a certain role to achieve the concept.

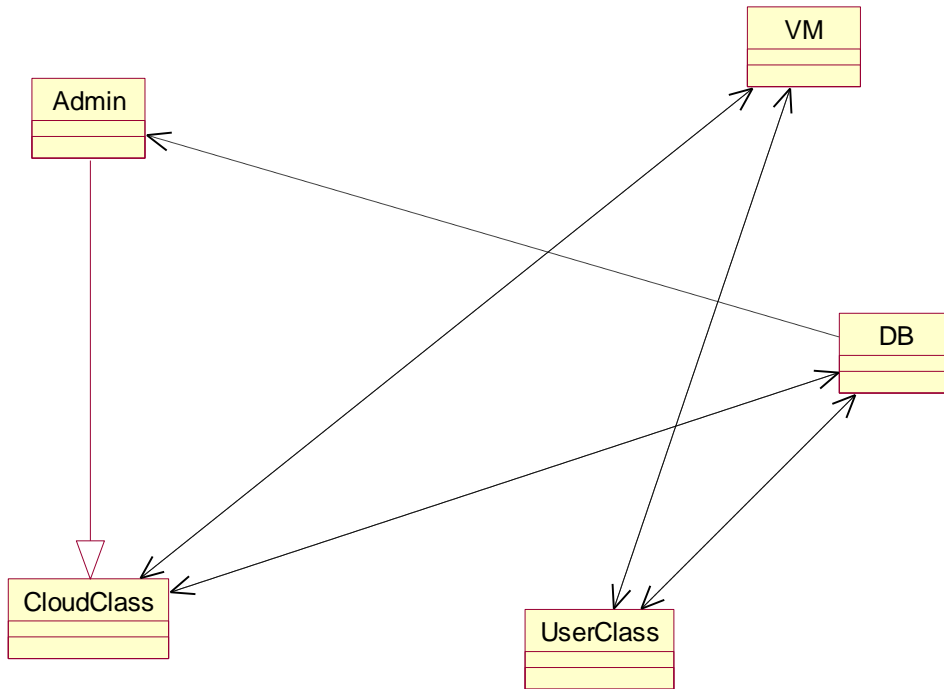
CLASS DIAGRAM



EXPLANATION

In this class diagram represents how the classes with attributes and methods are linked together to perform the verification with security. From the above diagram shown the various classes involved in our project.

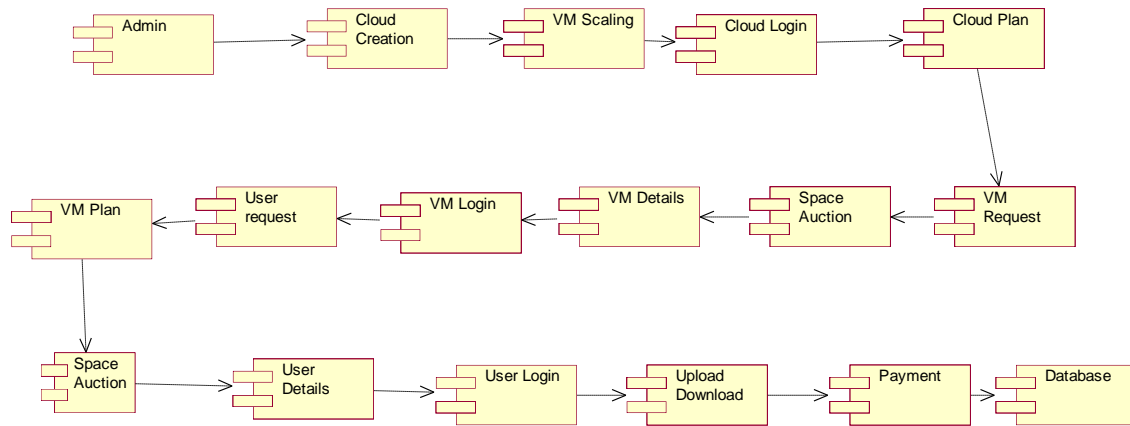
OBJECT DIAGRAM



EXPLANATION

In the above diagram tells about the flow of objects between the classes. It is a diagram that shows a complete or partial view of the structure of a modeled system. In this object diagram represents how the classes with attributes and methods are linked together to perform the verification with security.

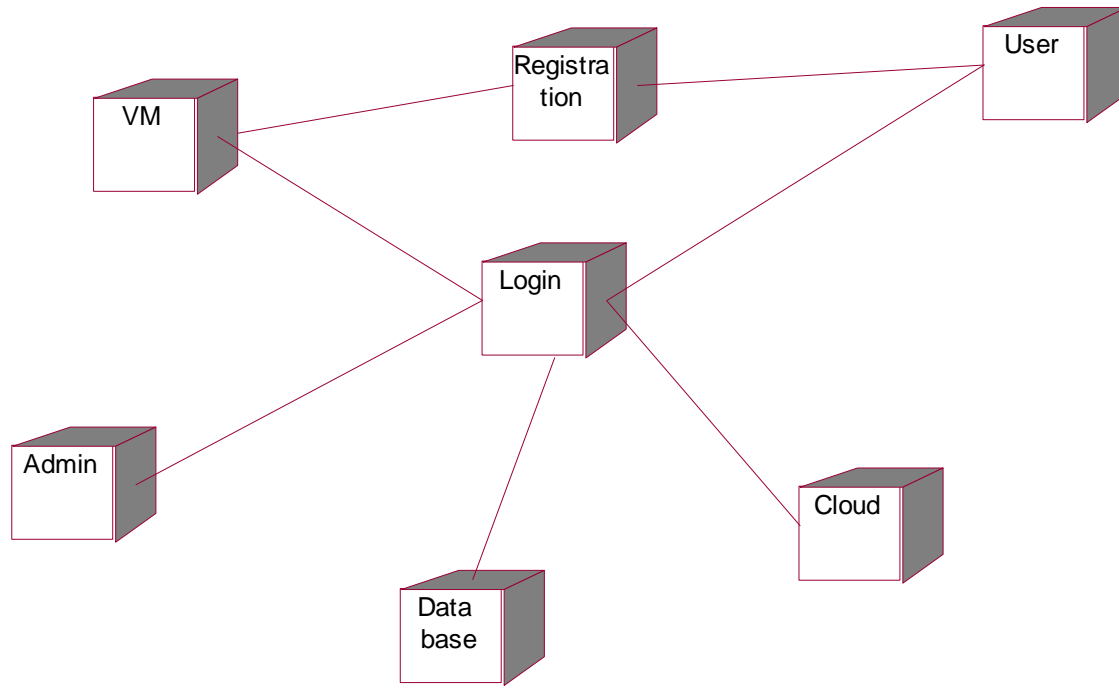
COMPONENT DIAGRAM



EXPLANATION

Component diagrams are essentially class diagrams that focus on a system's components that often used to model the static implementation view of a system.

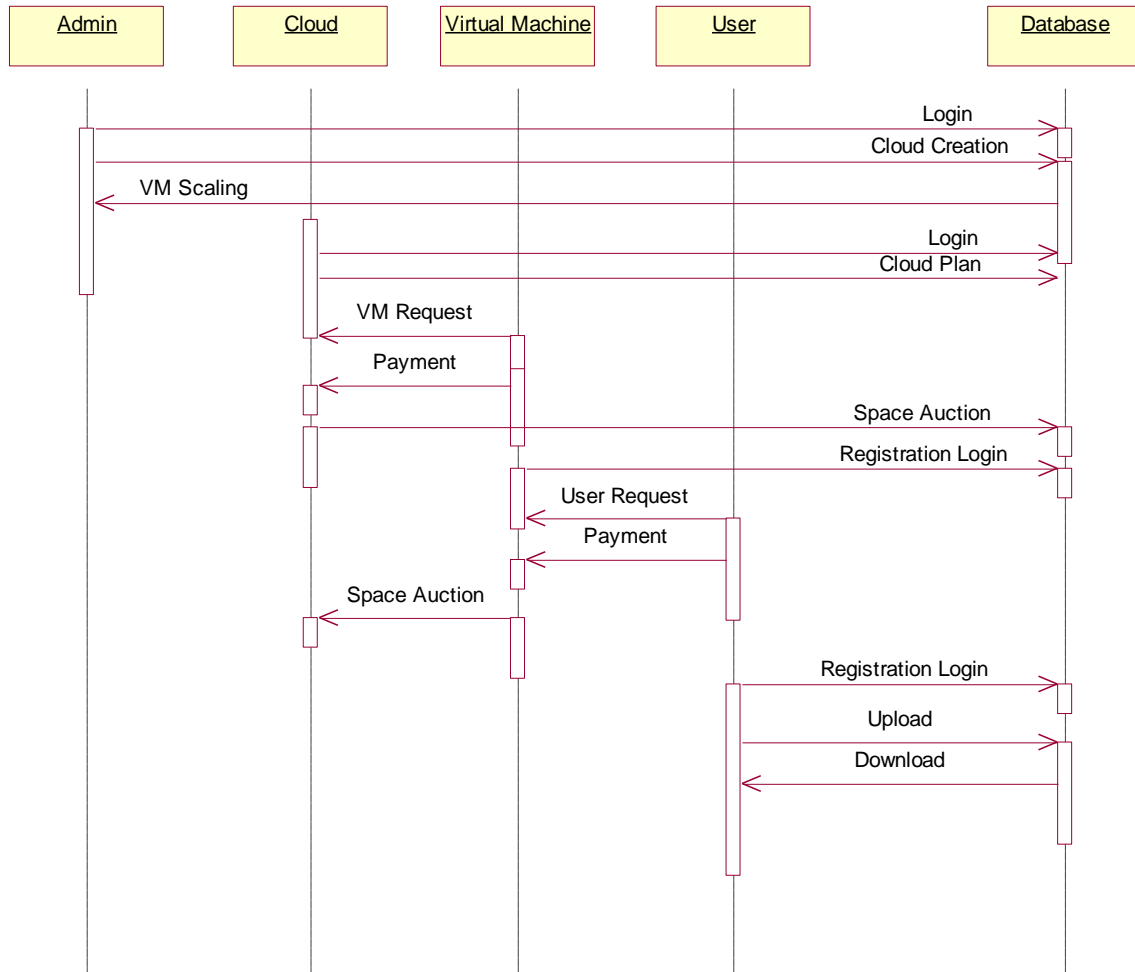
DEPLOYMENT DIAGRAM



EXPLANATION

Deployment diagrams is a kind of structure diagram used in modeling the physical aspects of an object-oriented system. They are often be used to model the static deployment view of a system (topology of the hardware).

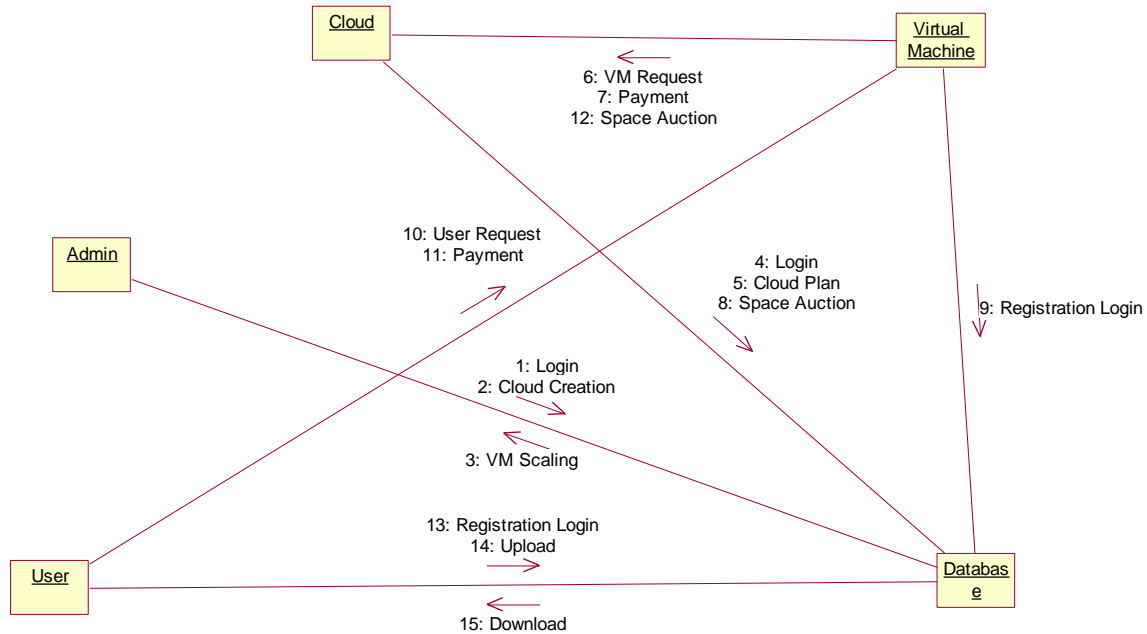
SEQUENCE DIAGRAM



EXPLANATION

Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration. Sequence Diagrams are time focus and they show the order of the interaction visually by using the vertical axis of the diagram to represent time what messages are sent and when.

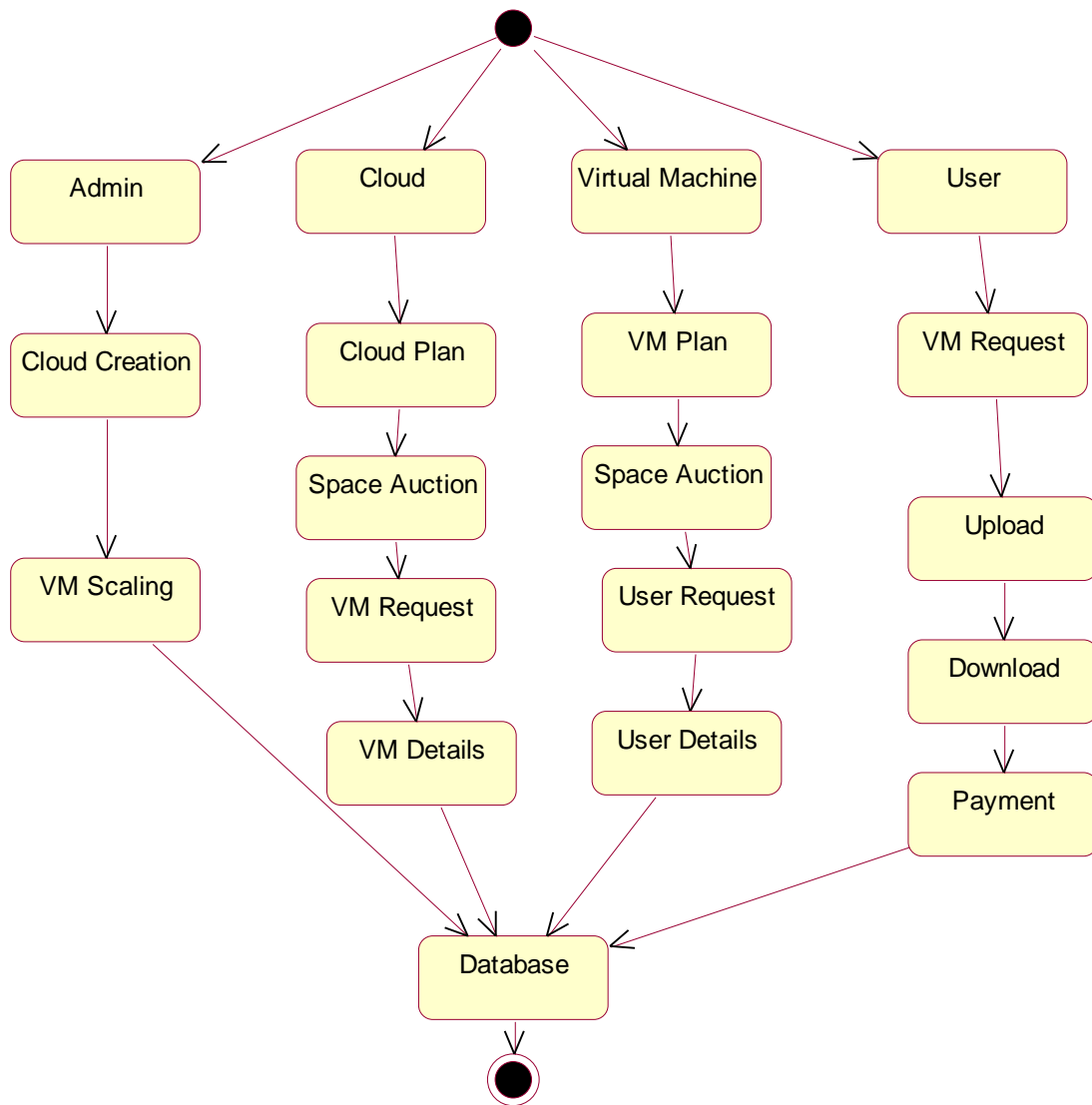
COLLABORATION DIAGRAM



EXPLANATION

A collaboration diagram, also called a communication diagram or interaction diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML). The concept is more than a decade old although it has been refined as modeling paradigms have evolved.

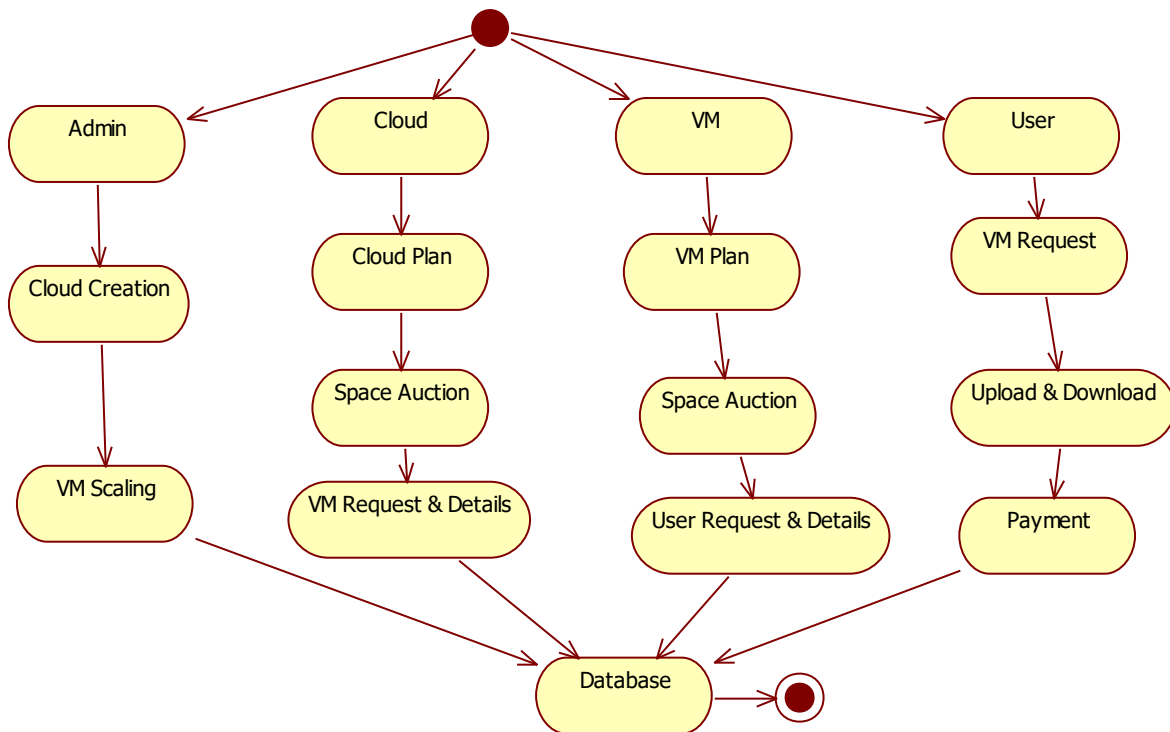
STATE DIAGRAM



EXPLANATION

State diagram are a loosely defined diagram to show workflows of stepwise activities and actions, with support for choice, iteration and concurrency. State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction. Many forms of state diagrams exist, which differ slightly and have different semantics.

ACTIVITY DIAGRAM

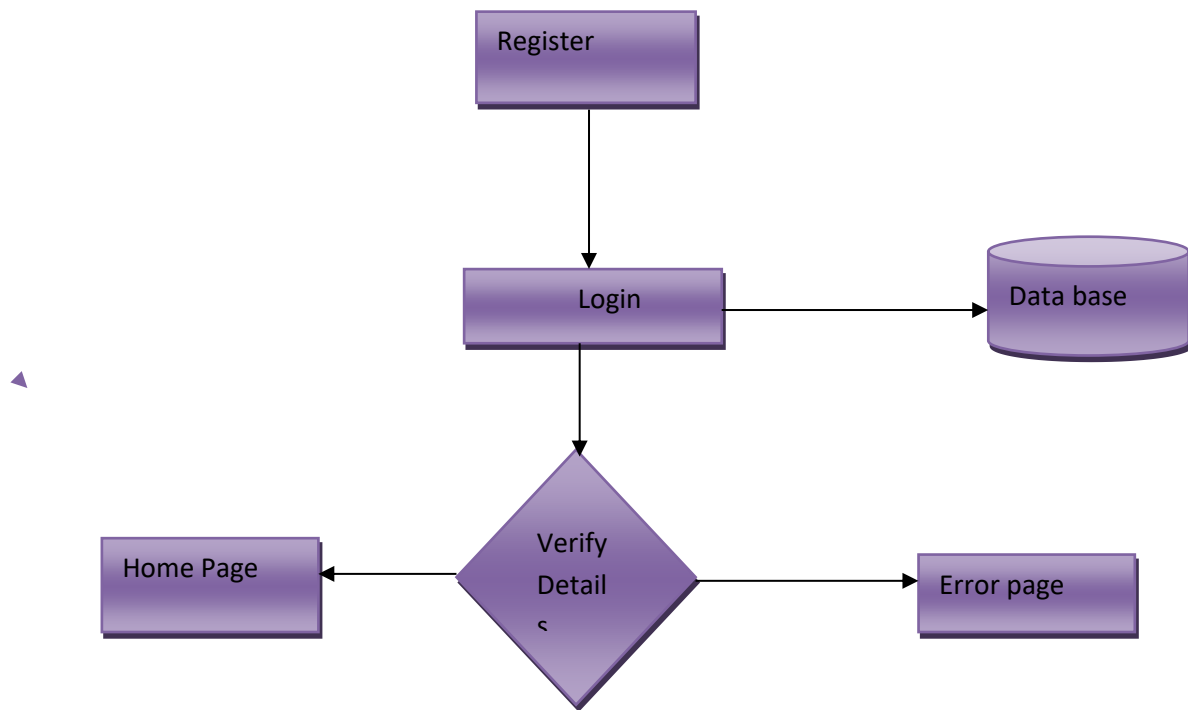


EXPLANATION

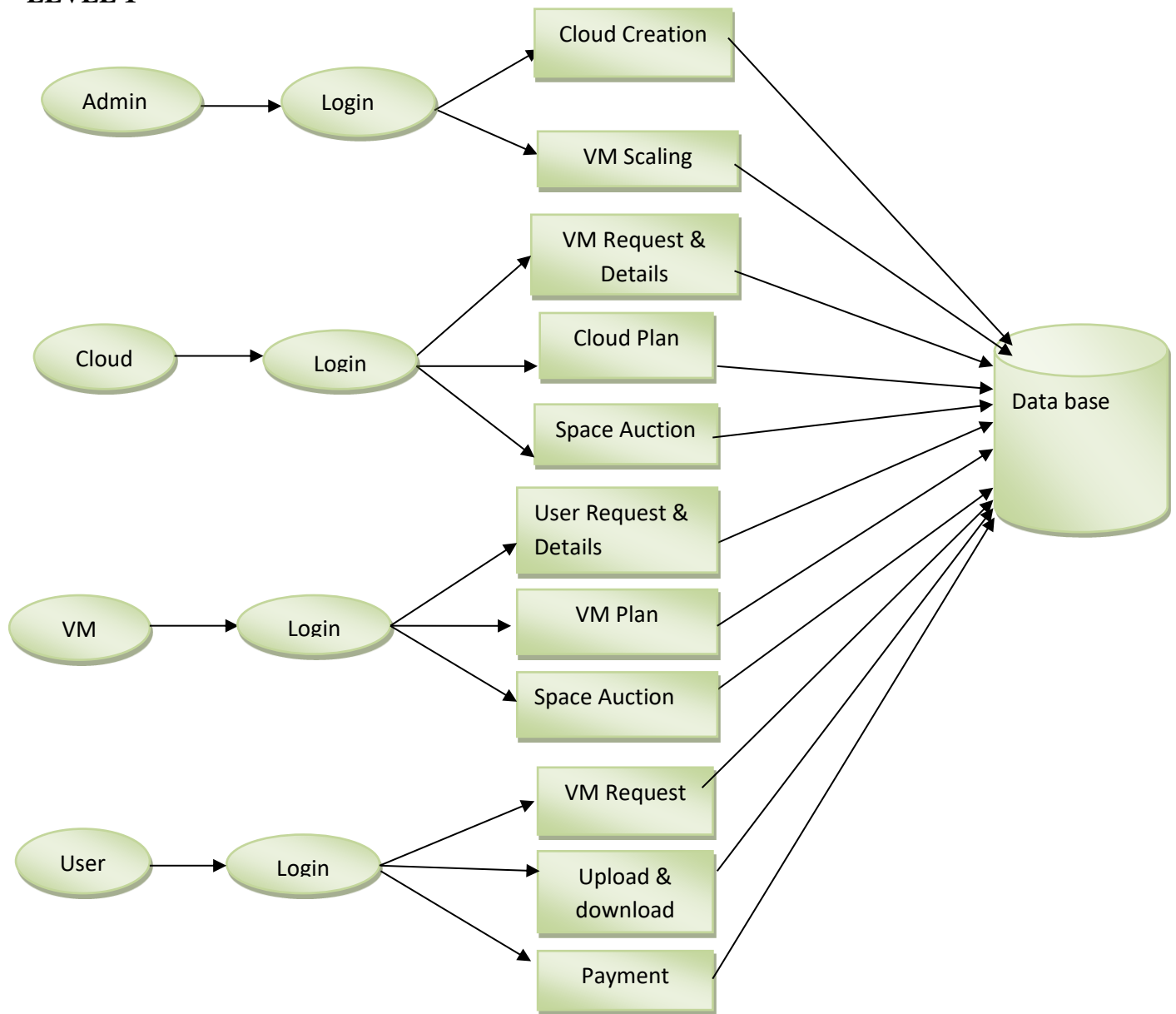
Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system.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

DATA FLOW DIAGRAM

LEVEL 0



LEVEL 1

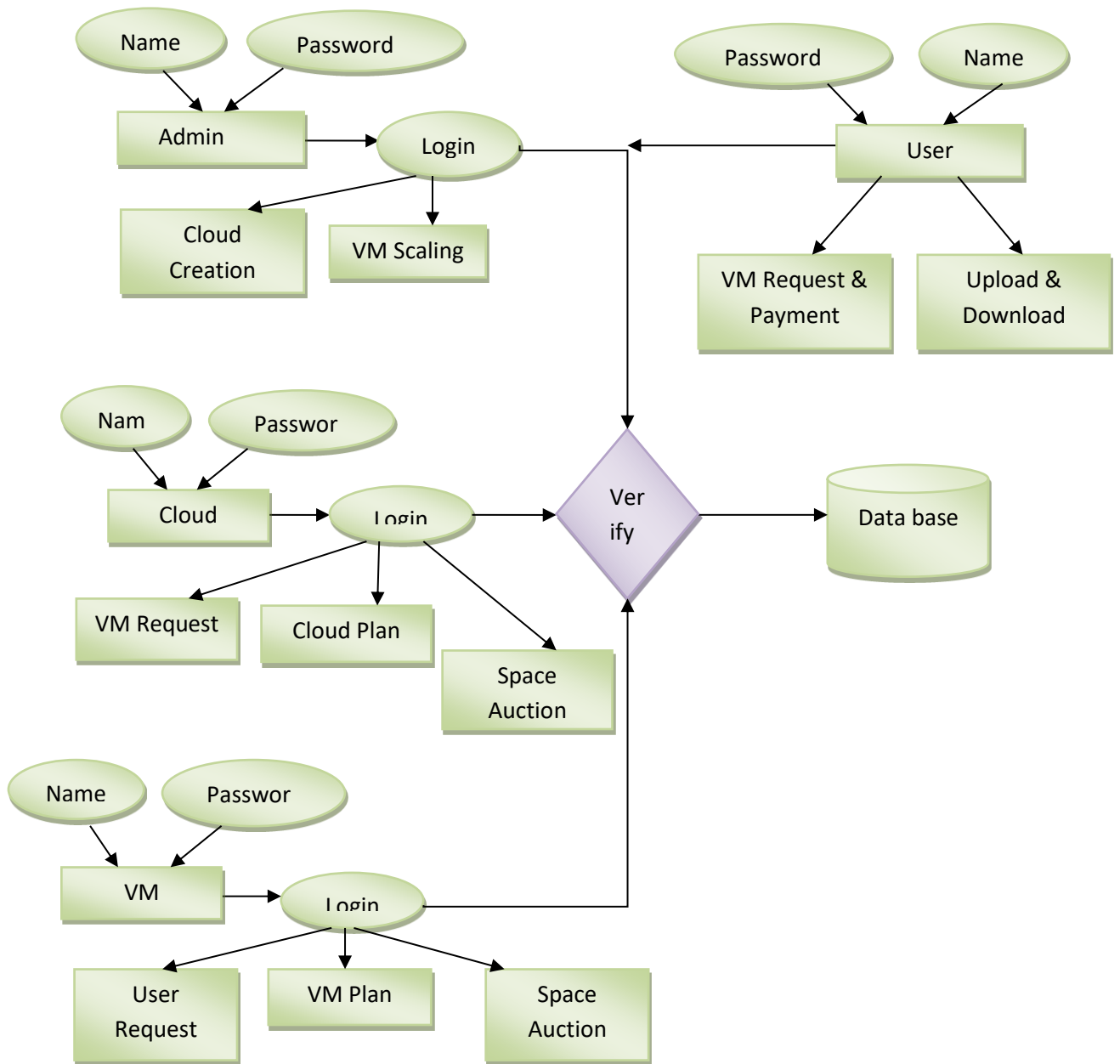


EXPLANATION

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. Often they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).

A DFD shows what kinds of data will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of processes, or information about whether processes will operate in sequence or in parallel.

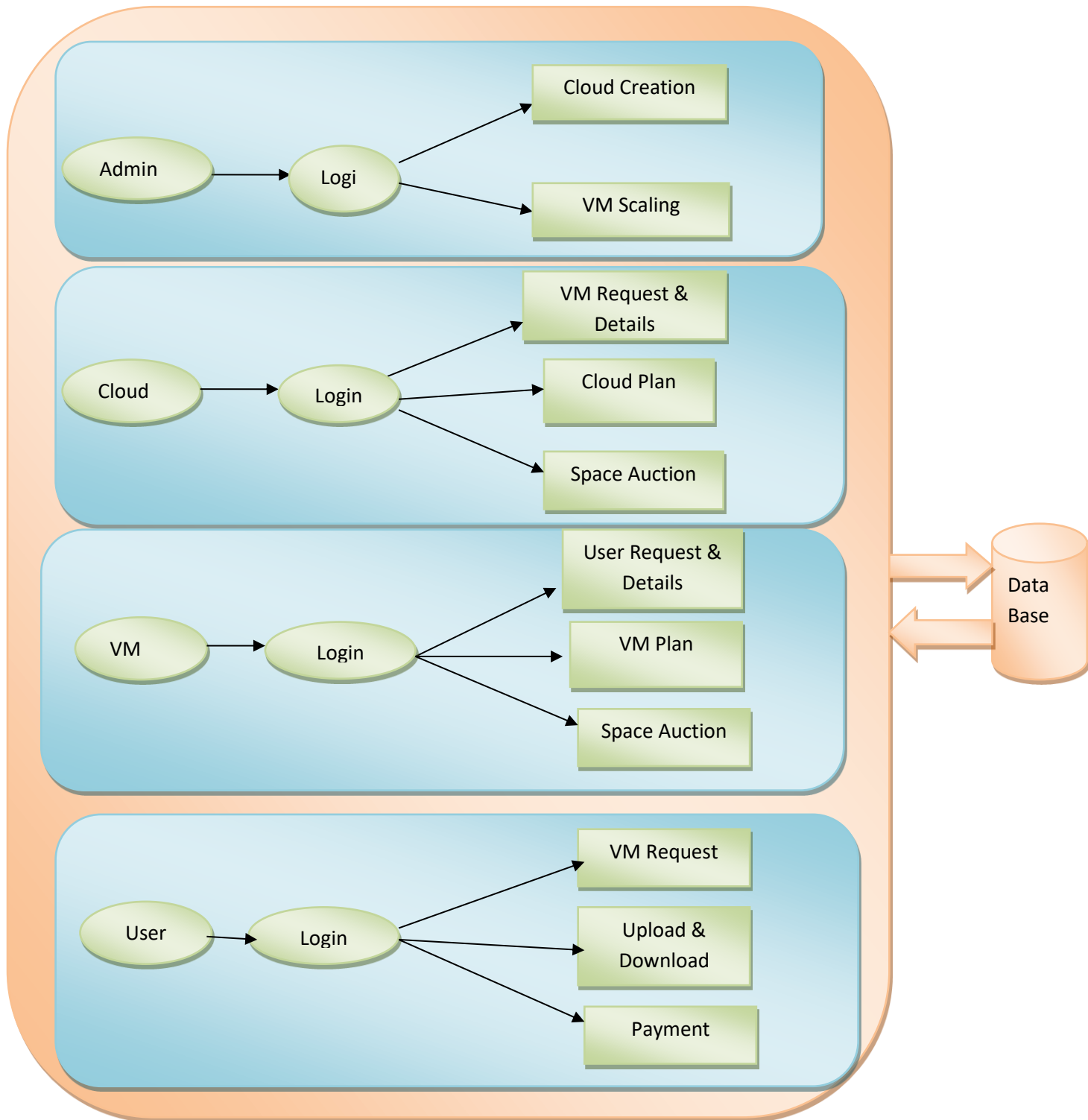
E-R DIAGRAM



EXPLANATION

Entity-Relationship Model (ERM) is an abstract and conceptual representation of data. Entity-relationship modeling is a database modeling method, used to produce a type of conceptual schema or semantic data model of a system, often a relational database.

SYSTEM ARCHITECTURE



EXPLANATION

In this Project, admin will create the cloud with some space. Cloud owner will derive their own plan for attracting the Virtual Machine owners. VM owners will buy the space from how much they required by paying the money to utilize the space. User will register and request the virtual machine to store the data in the cloud space. Based on the plan of VM the amount needs to pay for the storage of data. And cloud will provide some space for auction. Based on the price the VM owners will quote the amount for that space. Which VM owner will quote the maximum price for that space will get the space.

CHAPTER 5

DEVELOPMENT TOOLS

1.1 GENERAL

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

5.2 FEATURES OF JAVA

5.2.1 THE JAVA FRAMEWORK

Java is a programming language originally developed by James Gosling at Microsystems and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++ but has a simpler object model and fewer low-level facilities. Java applications are typically compiled to byte code that can run on any Java Virtual Machine (JVM) regardless of computer architecture. Java is general-purpose, concurrent, class-based, and object-oriented, and is specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere".

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

5.2.2 OBJECTIVES OF JAVA

To see places of Java in Action in our daily life, explore java.com.

Why Software Developers Choose Java

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

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

Some Ways Software Developers Learn Java

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

ObjectOriented

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

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

2. Encapsulation: It is the mechanism of combining the information and providing the abstraction.
3. Polymorphism: As the name suggest one name multiple form, Polymorphism is the way of providing the different functionality by the functions having the same name based on the signatures of the methods.
4. Dynamic binding: Sometimes we don't have the knowledge of objects about their specific types while writing our code. It is the way of providing the maximum functionality to a program about the specific type at runtime.

5.2.3 JAVA SWING OVERVIEW

Abstract Window Toolkit (AWT) is cross-platform

Swing provides many controls and widgets to build user interfaces with. Swing class names typically begin with a J such as JButton, JList, JFrame. This is mainly to differentiate them from their AWT counterparts and in general is one-to-one replacements. Swing is built on the concept of Lightweight components vs AWT and SWT's concept of Heavyweight components. The difference between the two is that the Lightweight components are rendered (drawn) using purely Java code, such as drawLine and drawImage, whereas Heavyweight components use the native operating system to render the components. Some components in Swing are actually heavyweight components. The top-level classes and any derived from them are heavyweight as they extend the AWT versions. This is needed because at the root of the UI, the parent windows need to be provided by the OS. These top-level classes include JWindow, JFrame, JDialog and JApplet. All Swing components to be rendered to the screen must be able to trace their way to a root window of one of those classes.

Note: It generally it is not a good idea to mix heavyweight components with lightweight components (other than as previously mentioned) as you will encounter layering issues, e.g., a lightweight component that should appear "on top" ends up being obscured by a

heavyweight component. The few exceptions to this include using heavyweight components as the root pane and for popup windows. Generally speaking, heavyweight components will render on top of lightweight components and will not be consistent with the look and feel being used in Swing. There are exceptions, but that is an advanced topic. The truly adventurous may want to consider reading this article from Sun on mixing heavyweight and lightweight components.

5.2.4 Evolution of Collection Framework:

Almost all collections in Java are derived from the **java.util.Collection** interface. Collection defines the basic parts of all collections. The interface states the `add()` and `remove()` methods for adding to and removing from a collection respectively. Also required is the `toArray()` method, which converts the collection into a simple array of all the elements in the collection. Finally, the `contains()` method checks if a specified element is in the collection. The Collection interface is a subinterface of **java.util.Iterable**, so the `iterator()` method is also provided. All collections have an iterator that goes through all of the elements in the collection. Additionally, Collection is a generic. Any collection can be written to store any class. For example, `Collection<String>` can hold strings, and the elements from the collection can be used as strings without any casting required.

There are three main types of collections:

- Lists: always ordered, may contain duplicates and can be handled the same way as usual arrays
- Sets: cannot contain duplicates and provide random access to their elements
- Maps: connect unique keys with values, provide random access to its keys and may host duplicate values

LIST

Lists are implemented in the JCF via the `java.util.List` interface. It defines a list as essentially a more flexible version of an array. Elements have a specific order, and duplicate elements are allowed. Elements can be placed in a specific position. They can also be searched for within the list. Two concrete classes implement List. The first is `java.util.ArrayList`, which implements the list as an array. Whenever functions specific to a list are required, the class moves the elements around within the array in order to do it. The other implementation is `java.util.LinkedList`. This class stores the elements in nodes that each have a pointer to the previous and next nodes in the list. The list can be traversed by following the pointers, and elements can be added or removed simply by changing the pointers around to place the node in its proper place.

SET

Java's `java.util.Set` interface defines the set. A set can't have any duplicate elements in it. Additionally, the set has no set order. As such, elements can't be found by index. Set is implemented by `java.util.HashSet`, `java.util.LinkedHashSet`, and `java.util.TreeSet`. `HashSet` uses a hash table. More specifically, it uses a `java.util.HashMap` to store the hashes and elements and to prevent duplicates. `java.util.LinkedHashSet` extends this by creating a doubly linked list that links all of the elements by their insertion order. This ensures that the iteration order over the set is predictable. `java.util.TreeSet` uses a red-black tree implemented by a `java.util.TreeMap`. The red-black tree makes sure that there are no duplicates. Additionally, it allows Tree Set to implement `java.util.SortedSet`.

The `java.util.Set` interface is extended by the `java.util.SortedSet` interface. Unlike a regular set, the elements in a sorted set are sorted, either by the element's `compareTo()` method, or a method provided to the constructor of the sorted set. The first and last elements of the sorted set can be retrieved, and subsets can be created via minimum and maximum values, as well as beginning or ending at the beginning or ending of the sorted set. The `SortedSet` interface is implemented by `java.util.TreeSet`

`java.util.SortedSet` is extended further via the `java.util.NavigableSet` interface. It's similar to `SortedSet`, but there are a few additional methods. The `floor()`, `ceiling()`, `lower()`, and `higher()` methods find an element in the set that's close to the parameter. Additionally, a descending iterator over the items in the set is provided. As with `SortedSet`, `java.util.TreeSet` implements `NavigableSet`.

MAP

Maps are defined by the `java.util.Map` interface in Java. Maps are simple data structures that associate a key with a value. The element is the value. This lets the map be very flexible. If the key is the hash code of the element, the map is essentially a set. If it's just an increasing number, it becomes a list. Maps are implemented by `java.util.HashMap`, `java.util.LinkedHashMap`, and `java.util.TreeMap`. `HashMap` uses a hash table. The hashes of the keys are used to find the values in various buckets. `LinkedHashMap` extends this by creating a doubly linked list between the elements. This allows the elements to be accessed in the order in which they were inserted into the map. `TreeMap`, in contrast to `HashMap` and `LinkedHashMap`, uses a red-black tree. The keys are used as the values for the nodes in the tree, and the nodes point to the values in the map

THREAD

Simply put, a *thread* is a program's path of execution. Most programs written today run as a single thread, causing problems when multiple events or actions need to occur at the same time. Let's say, for example, a program is not capable of drawing pictures while reading keystrokes. The program must give its full attention to the keyboard input lacking the ability to handle more than one event at a time. The ideal solution to this problem is the seamless execution of two or more sections of a program at the same time.

CREATING THREADS

Java's creators have graciously designed two ways of creating threads: implementing an interface and extending a class. Extending a class is the way Java inherits methods and variables from a parent class. In this case, one can only extend or inherit from a single parent class. This limitation within Java can be overcome by implementing interfaces, which is the most common way to create threads.

(Note that the act of inheriting merely allows the class to be run as a thread. It is up to the class to `start()` execution, etc.)

Interfaces provide a way for programmers to lay the groundwork of a class. They are used to design the requirements for a set of classes to implement. The interface sets everything up, and the class or classes that implement the interface do all the work. The different set of classes that implement the interface have to follow the same rules.

5.5 CONCLUSION

Swing's high level of flexibility is reflected in its inherent ability to override the native host operating system (OS)'s GUI controls for displaying itself. Swing "paints" its controls using the Java 2D APIs, rather than calling a native user interface toolkit. The Java thread scheduler is very simple. All threads have a priority value which can be changed dynamically by calls to the threads `setPriority()` method. Implementing the above concepts in our project to do the efficient work among the Server.

CHAPTER 6

IMPLEMENTATION

6.1 GENERAL

CODING

1.Index.Jsp

```
<!DOCTYPEHTML>
<html>
  <head>
    <title>VTJCC17_2019</title>
    <meta charset="utf-8"/>
    <meta name="viewport" content="width=device-width, initial-scale=1"/>
    <link rel="stylesheet" href="assets/css/main.css"/>
  </head>
  <body>

    <!-- Header -->
    <header id="header" class="alt">
      <div class="Logo"><a href="index.jsp" style="font-size:
30px;">Profit Maximization for Cloud Brokers in Cloud
Computing</a></div>
      <a href="#menu" class="toggle"><span>Menu</span></a>
    </header>

    <!-- Nav -->
    <nav id="menu">
      <ul class="links">
        <li><a href="index.jsp">Home</a></li>
        <li><a href="admin.jsp">Admin</a></li>
        <li><a href="cloud.jsp">Cloud</a></li>
        <li><a href="vm.jsp">VM</a></li>
        <li><a href="user.jsp">User</a></li>
      </ul>
    </nav>

    <!-- Banner -->
    <!--
      To use a video as your background, set data-video to the name of
your video without
      its extension (eg. images/banner). Your video must be available
in both .mp4 and .webm
      formats to work correctly.
    -->
```

```

<sectionid="banner"data-video="images/banner">
  <divclass="inner">
    <h1style="margin-top: 50px;">Abstract</h1>
    <pstyle="text-align: justify; margin-left: 30px;
margin-right: 20px;">

```

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, we introduce a new role, 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 paper, we focus 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 paper, we firstly give a synthetically analysis 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, we propose a heuristic method 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.</p>

```

</div>
</section>

```

```

<!-- Scripts -->
<scriptsrc="assets/js/jquery.min.js"></script>
<scriptsrc="assets/js/jquery.scrolly.min.js"></script>
<scriptsrc="assets/js/jquery.scrollex.min.js"></script>
<scriptsrc="assets/js/skel.min.js"></script>
<scriptsrc="assets/js/util.js"></script>
<scriptsrc="assets/js/main.js"></script>

```

```

</body>

```

```

</html>

```


2.Admin.jsp

```
<!DOCTYPEHTML>
<html>
  <head>
    <title>VTJCC17_2019</title>
    <metacharset="utf-8"/>
    <metaname="viewport"content="width=device-width, initial-scale=1"/>
    <linkrel="stylesheet"href="assets/css/main.css"/>
    <linkrel="stylesheet"href="assets/css/w3.css"/>
  </head>
  <body>

    <!-- Header -->
    <headerid="header"class="alt">
      <divclass="Logo"><a href="index.jsp" style="font-size:
30px;"> Profit Maximization for Cloud Brokers in Cloud
Computing</a></div>
      <a href="#menu"class="toggle"><span>Menu</span></a>
    </header>

    <!--Nav -->
    <navid="menu">
      <ulclass="Links">
        <li><a href="index.jsp">Home</a></li>
        <li><a href="cloud.jsp">Cloud</a></li>
        <li><a href="vm.jsp">VM</a></li>
        <li><a href="user.jsp">User</a></li>
      </ul>
    </nav>

    <!-- Banner -->
    <!--
      To use a video as your background, set data-video to the name of
your video without
      its extension (eg. images/banner). Your video must be available
in both .mp4 and .webm
      formats to work correctly.
    -->
    <sectionid="banner"data-video="images/banner">
      <divclass="inner">
        <h1style="margin-top: 50px;">Admin Login</h1>
        <formaction="Admin"method="post">
          <tableclass="w3-table w3-text-white w3-
black"style="width: 750px;">

            <tr><td>UserId</td><td><inputtype="text"name="uid"class="w3-input"></td></tr>

            <tr><td>Password</td><td><inputtype="password"name="pwd"class="w3-
input"></td></tr>

          </table>
          <inputtype="submit"value="Login"class="w3-button w3-
white">
        </form>
      </div>
    </section>
```

```
<!-- Scripts -->
    <scriptsrc="assets/js/jquery.min.js"></script>
    <scriptsrc="assets/js/jquery.scrollly.min.js"></script>
    <scriptsrc="assets/js/jquery.scrollex.min.js"></script>
    <scriptsrc="assets/js/skel.min.js"></script>
    <scriptsrc="assets/js/util.js"></script>
    <scriptsrc="assets/js/main.js"></script>

</body>

</html>
```

3. AdminLogin.java

```
package com.control;

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;
```

```

/**

 * Servlet implementation class Admin

 */

@WebServlet("/Admin")

public class Admin extends HttpServlet {

    private static final long serialVersionUID = 1L;


    /**

     * @see HttpServlet#HttpServlet()

     */

    public Admin() {

super();

        // TODO Auto-generated constructor stub

    }


    /**

     * @see HttpServlet#doGet(HttpServletRequest request,
HttpServletResponse response)

     */

```

```

        protected void doGet(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {

            // TODO Auto-generated method stub

        }

        /**

        * @see HttpServlet#doPost(HttpServletRequest request,
        HttpServletResponse response)

        */

        protected void doPost(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {

            // TODO Auto-generated method stub

            PrintWriter o = response.getWriter();

            String uid = request.getParameter("uid");

            String pwd = request.getParameter("pwd");

            if(uid.equals("admin") &&pwd.equals("admin")){

                response.sendRedirect("ahome.jsp");

            }else{

```

```

o.println("<script type=\"text/javascript\">");

o.println("alert('Please enter valid Details');");

o.println("window.location='admin.jsp';</script>");

}

}

}

```

4.AdminHome.jsp

```

<!DOCTYPEHTML>
<html>
  <head>
    <title>VTJCC17_2019</title>
    <meta charset="utf-8"/>
    <meta name="viewport" content="width=device-width, initial-scale=1"/>
    <link rel="stylesheet" href="assets/css/main.css"/>
    <link rel="stylesheet" href="assets/css/w3.css"/>
  </head>
  <body>

    <!-- Header -->
    <header id="header" class="alt">
      <div class="logo"><a href="index.jsp" style="font-size:
30px;"> Profit Maximization for Cloud Brokers in Cloud
Computing</a></div>
      <a href="#menu" class="toggle"><span>Menu</span></a>
    </header>

    <!-- Nav -->
    <nav id="menu">
      <ul class="Links">
        <li><a href="ahome.jsp">Home</a></li>
        <li><a href="cloudreg.jsp">Cloud Creation</a></li>
        <li><a href="cloudscaling.jsp">VM Scaling</a></li>
        <li><a href="index.jsp">Logout</a></li>
      </ul>
    </nav>

```

```

        <!-- Banner -->
        <!--
            To use a video as your background, set data-video to the name of
your video without
            its extension (eg. images/banner). Your video must be available
in both .mp4 and .webm
            formats to work correctly.
        -->
        <sectionid="banner"data-video="images/banner">
            <divclass="inner">
                <h1style="margin-top: 50px;">Welcome To Admin</h1>
            </div>
        </section>

        <!-- Scripts -->
        <scriptsrc="assets/js/jquery.min.js"></script>
        <scriptsrc="assets/js/jquery.scrolly.min.js"></script>
        <scriptsrc="assets/js/jquery.scrollex.min.js"></script>
        <scriptsrc="assets/js/skel.min.js"></script>
        <scriptsrc="assets/js/util.js"></script>
        <scriptsrc="assets/js/main.js"></script>

    </body>

</html>

```

5.Reg.jsp

```

<!DOCTYPEHTML>
<html>
    <head>
        <title>VTJCC17_2019</title>
        <metacharset="utf-8"/>
        <metaname="viewport"content="width=device-width, initial-scale=1"/>
        <linkrel="stylesheet"href="assets/css/main.css"/>
        <linkrel="stylesheet"href="assets/css/w3.css"/>
    </head>
    <body>

        <!-- Header -->
        <headerid="header"class="alt">
            <divclass="Logo"><a href="index.jsp" style="font-size:
30px;"> Profit Maximization for Cloud Brokers in Cloud
Computing</a></div>
            <a href="#menu"class="toggle"><span>Menu</span></a>
        </header>

        <!-- Nav -->
        <navid="menu">
            <ulclass="links">
                <li><a href="ahome.jsp">Home</a></li>
                <li><a href="cloudreg.jsp">Cloud Creation</a></li>
                <li><a href="cloudscaling.jsp">VM Scaling</a></li>
                <li><a href="index.jsp">Logout</a></li>
            </ul>
        </nav>

```

```

<!-- Banner -->
<!--
    To use a video as your background, set data-video to the name of
your video without
    its extension (eg. images/banner). Your video must be available
in both .mp4 and .webm
    formats to work correctly.
-->
<sectionid="banner"data-video="images/banner">
    <divclass="inner">
        <h1style="margin-top: 50px;">Cloud Creation</h1>
        <formaction="CloudReg"method="post">
            <tableclass="w3-table w3-black"style="width:
750px;">

                <tr><td>Name</td><td><inputtype="text"name="name"class="w3-input"></td></tr>

                <tr><td>Email</td><td><inputtype="text"name="uid"class="w3-input"></td></tr>

                <tr><td>Password</td><td><inputtype="password"name="pwd"class="w3-
input"></td></tr>

                <tr><td>Space</td><td><inputtype="text"name="mob"class="w3-input"></td></tr>
            </table>
            <inputtype="submit"value="Register"class="w3-button
w3-white">

        </form>
    </div>
</section>

<!-- Scripts -->
    <scriptsrc="assets/js/jquery.min.js"></script>
    <scriptsrc="assets/js/jquery.scrolly.min.js"></script>
    <scriptsrc="assets/js/jquery.scrollex.min.js"></script>
    <scriptsrc="assets/js/skel.min.js"></script>
    <scriptsrc="assets/js/util.js"></script>
    <scriptsrc="assets/js/main.js"></script>

</body>

</html>

```

6.UserReg.java

```
package com.control;
```

```
import java.io.IOException;
```

```
import java.io.PrintWriter;
```

```
import javax.servlet.ServletException;
```

```
import javax.servlet.annotation.WebServlet;
```

```
import javax.servlet.http.HttpServlet;
```

```
import javax.servlet.http.HttpServletRequest;
```

```
import javax.servlet.http.HttpServletResponse;
```

```
import com.bean.UserBean;
```

```
import com.dao.Dao;
```



```

/**

 * Servlet implementation class CloudReg

 */

@WebServlet("/CloudReg")

public class CloudReg extends HttpServlet {

    private static final long serialVersionUID = 1L;

    /**

     * @see HttpServlet#HttpServlet()

     */

    public CloudReg() {

super();

        // TODO Auto-generated constructor stub

    }

```

```

/**

    * @see HttpServlet#doGet(HttpServletRequest request,
HttpServletResponse response)

    */

    protected void doGet(HttpServletRequest request,
HttpServletResponse response) throws ServletException, IOException {

        // TODO Auto-generated method stub

    }

/**

    * @see HttpServlet#doPost(HttpServletRequest request,
HttpServletResponse response)

    */

    protected void doPost(HttpServletRequest request,
HttpServletResponse response) throws ServletException, IOException {

        // TODO Auto-generated method stub

        PrintWriter o = response.getWriter();

        String name = request.getParameter("name");

        String email = request.getParameter("uid");

```

```
String pwd = request.getParameter("pwd");

String mob = request.getParameter("mob");

UserBeanub = new UserBean();

ub.setName(name);

ub.setEmail(email);

ub.setPwd(pwd);

ub.setMob(mob);

String sql = "insert into cloud values(?,?,?,?)";

int i = Dao.vm(sql, ub);

if(i> 0){

    o.println("<script type=\"text/javascript\">");

    o.println("alert('Cloud Created Successfully...');");

o.println("window.location='ahome.jsp';</script>");

}else{

    o.println("<script type=\"text/javascript\">");

    o.println("alert('Cloud not created');");
```

```
o.println("window.location='cloudreg.jsp';</script>");
```

```
}
```

```
}
```

```
}
```

7.User.jsp

```
<!DOCTYPEHTML>
<html>
  <head>
    <title>VTJCC17_2019</title>
    <meta charset="utf-8"/>
    <meta name="viewport" content="width=device-width, initial-scale=1"/>
    <link rel="stylesheet" href="assets/css/main.css"/>
    <link rel="stylesheet" href="assets/css/w3.css"/>
  </head>
  <body>

    <!-- Header -->
    <header id="header" class="alt">
      <div class="Logo"><a href="index.jsp" style="font-size:
30px;"> Profit Maximization for Cloud Brokers in Cloud
Computing</a></div>
      <a href="#menu" class="toggle"><span>Menu</span></a>
    </header>

    <!-- Nav -->
    <nav id="menu">
      <ul class="links">
        <li><a href="index.jsp">Home</a></li>
        <li><a href="admin.jsp">Admin</a></li>
        <li><a href="cloud.jsp">Cloud</a></li>
        <li><a href="vm.jsp">VM</a></li>
        <li><a href="userreg.jsp">User Registration</a></li>
      </ul>
    </nav>
```

```

        <!-- Banner -->
        <!--
            To use a video as your background, set data-video to the name of
your video without
            its extension (eg. images/banner). Your video must be available
in both .mp4 and .webm
            formats to work correctly.
        -->
        <sectionid="banner"data-video="images/banner">
            <divclass="inner">
                <h1style="margin-top: 50px;">User Login</h1>
                <formaction="User"method="post">
                    <tableclass="w3-table w3-black"style="width:
750px;">

                        <tr><td>Email</td><td><inputtype="text"name="uid"class="w3-input"></td></tr>

                        <tr><td>Password</td><td><inputtype="password"name="pwd"class="w3-
input"></td></tr>

                                </table>
                                <inputtype="submit"value="Login"class="w3-button w3-
white">

                                    </form>
                                </div>
                            </section>

        <!-- Scripts -->
        <scriptsrc="assets/js/jquery.min.js"></script>
        <scriptsrc="assets/js/jquery.scrolly.min.js"></script>
        <scriptsrc="assets/js/jquery.scrollex.min.js"></script>
        <scriptsrc="assets/js/skel.min.js"></script>
        <scriptsrc="assets/js/util.js"></script>
        <scriptsrc="assets/js/main.js"></script>

    </body>

</html>

```

8.Login.java

```
package com.control;

import java.io.IOException;

import java.io.PrintWriter;


import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;


import com.dao.Dao;

/**

 * Servlet implementation class User

 */
```

```

@WebServlet("/User")

public class User extends HttpServlet {

    private static final long serialVersionUID = 1L;

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

    /**
     * @see HttpServlet#doGet(HttpServletRequest request,
    HttpServletResponse response)
     */

```

```

        protected void doGet(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {

            // TODO Auto-generated method stub

        }

        /**

        * @see HttpServlet#doPost(HttpServletRequest request,
        HttpServletResponse response)

        */

        protected void doPost(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {

            // TODO Auto-generated method stub

            PrintWriter o = response.getWriter();

            String uid = request.getParameter("uid");

            String pwd = request.getParameter("pwd");

            String sql = "select * from user where email='"+uid+"'
and password='"+pwd+"' and status1='Approved'";

            boolean b = Dao.login(sql);

            HttpSession session = request.getSession();

```



```

        if(b == true){

            session.setAttribute("email", uid);

            sql = "select name from user where
email='"+uid+"'";

            String name = Dao.getName(sql);

            session.setAttribute("name", name);

            response.sendRedirect("uhome.jsp");

        }else{

            o.println("<script type=\"text/javascript\">");

            o.println("alert('Please enter valid Details/
Register');");

            o.println("window.location='user.jsp';</script>");

        }

    }

}

```

9.VM.jsp

```
<!DOCTYPEHTML>
<html>
  <head>
    <title>VTJCC17_2019</title>
    <metacharset="utf-8"/>
    <metaname="viewport"content="width=device-width, initial-scale=1"/>
    <linkrel="stylesheet"href="assets/css/main.css"/>
    <linkrel="stylesheet"href="assets/css/w3.css"/>
  </head>
  <body>

    <!-- Header -->
    <headerid="header"class="alt">
      <divclass="Logo"><a href="index.jsp" style="font-size:
30px;"> Profit Maximization for Cloud Brokers in Cloud
Computing</a></div>
      <a href="#menu"class="toggle"><span>Menu</span></a>
    </header>

    <!-- Nav -->
    <navid="menu">
      <ulclass="Links">
        <li><a href="index.jsp">Home</a></li>
        <li><a href="admin.jsp">Admin</a></li>
        <li><a href="cloud.jsp">Cloud</a></li>
        <li><a href="vmreg.jsp">VM Registration</a></li>
        <li><a href="user.jsp">User</a></li>
      </ul>
    </nav>

    <!-- Banner -->
    <!--
      To use a video as your background, set data-video to the name of
your video without
      its extension (eg. images/banner). Your video must be available
in both .mp4 and .webm
      formats to work correctly.
    -->
    <sectionid="banner"data-video="images/banner">
      <divclass="inner">
        <h1style="margin-top: 50px;">VM Login</h1>
        <formaction="VM"method="post">
          <tableclass="w3-table w3-black"style="width:
750px;">
            <tr><td>Email</td><td><inputtype="text"name="uid"class="w3-input"></td></tr>
            <tr><td>Password</td><td><inputtype="password"name="pwd"class="w3-
input"></td></tr>
          </table>
          <inputtype="submit"value="Login"class="w3-button w3-
white">
        </form>
      </div>
    </section>
```

```

        <!-- Scripts -->
        <scriptsrc="assets/js/jquery.min.js"></script>
        <scriptsrc="assets/js/jquery.scrolly.min.js"></script>
        <scriptsrc="assets/js/jquery.scrollex.min.js"></script>
        <scriptsrc="assets/js/skel.min.js"></script>
        <scriptsrc="assets/js/util.js"></script>
        <scriptsrc="assets/js/main.js"></script>

    </body>

</html>

```

10.MediaHome.jsp

```

<!DOCTYPEHTML>
<html>
    <head>
        <title>VTJCC17_2019</title>
        <metacharset="utf-8"/>
        <metaname="viewport"content="width=device-width, initial-scale=1"/>
        <linkrel="stylesheet"href="assets/css/main.css"/>
        <linkrel="stylesheet"href="assets/css/w3.css"/>
    </head>
    <body>

        <!-- Header -->
        <headerid="header"class="alt">
            <divclass="logo"><a href="index.jsp"style="font-size:
30px;"> Profit Maximization for Cloud Brokers in Cloud
Computing</a></div>
            <a href="#menu"class="toggle"><span>Menu</span></a>
        </header>

        <!-- Nav -->
        <navid="menu">
            <ulclass="Links">
                <li><a href="chome.jsp">Home</a></li>
                <li><a href="vmrequest.jsp">VM Request</a></li>
                <li><a href="vmospace.jsp">Space Auction</a></li>
                <li><a href="vmdetails.jsp">VM Details</a></li>
                <li><a href="cloudplans.jsp">Cloud Plan</a></li>
                <li><a href="logout.jsp">Logout</a></li>
            </ul>
        </nav>

        <!-- Banner -->
        <!--
            To use a video as your background, set data-video to the name of
your video without
            its extension (eg. images/banner). Your video must be available
in both .mp4 and .webm
            formats to work correctly.
        -->

```

```

    <%
    session = request.getSession(false);
    String name = (String) session.getAttribute("name");
    %>
        <sectionid="banner" data-video="images/banner">
            <divclass="inner">
                <h1style="margin-top: 50px;">Welcome To <%=name
%></h1>
            </div>
        </section>

    <!-- Scripts -->
        <scriptsrc="assets/js/jquery.min.js"></script>
        <scriptsrc="assets/js/jquery.scrolly.min.js"></script>
        <scriptsrc="assets/js/jquery.scrollex.min.js"></script>
        <scriptsrc="assets/js/skel.min.js"></script>
        <scriptsrc="assets/js/util.js"></script>
        <scriptsrc="assets/js/main.js"></script>

    </body>

</html>

```

```

package com.control;

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import com.bean.UserBean;

import com.dao.Dao;

```

```

/**
 * Servlet implementation class VMReg
 */
@WebServlet("/VMReg")

public class VMReg extends HttpServlet {

    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public VMReg() {
super();

        // TODO Auto-generated constructor stub
    }

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

        // TODO Auto-generated method stub
    }

```

```

/**
 * @see HttpServlet#doPost(HttpServletRequest request,
HttpServletResponse response)

 */

protected void doPost(HttpServletRequest request,
HttpServletResponse response) throws ServletException, IOException {

    // TODO Auto-generated method stub

    PrintWriter o = response.getWriter();

    String name = request.getParameter("name");

    String email = request.getParameter("uid");

    String pwd = request.getParameter("pwd");

    String mob = request.getParameter("mob");

    UserBeanub = new UserBean();

    ub.setName(name);

    ub.setEmail(email);

    ub.setPwd(pwd);

    ub.setMob(mob);

    String sql = "insert into vm values(?,?,?,?)";

    int i = Dao.vm(sql, ub);

    if(i> 0){

        o.println("<script type=\"text/javascript\">");

        o.println("alert('Register Successfully...');");

        o.println("window.location='vm.jsp';</script>");

```

```

        }else{

            o.println("<script type=\"text/javascript\">");

            o.println("alert('Please enter valid Details');");

            o.println("window.location='vmreg.jsp';</script>");

        }

    }

}

<!DOCTYPEHTML>
<%@pageimport="java.util.Iterator"%>
<%@pageimport="com.dao.Dao"%>
<%@pageimport="java.util.List"%>
<html>
    <head>
        <title>VTJCC17_2019</title>
        <metacharset="utf-8"/>
        <metaname="viewport"content="width=device-width, initial-scale=1"/>
        <linkrel="stylesheet"href="assets/css/main.css"/>
        <linkrel="stylesheet"href="assets/css/w3.css"/>
    </head>
    <body>

        <!-- Header -->
        <headerid="header"class="alt">
            <divclass="Logo"><a href="index.jsp" style="font-size:
30px;">Profit Maximization for Cloud Brokers in Cloud
Computing</a></div>
            <a href="#menu"class="toggle"><span>Menu</span></a>
        </header>

        <!-- Nav -->
        <navid="menu">
            <ulclass="links">
                <li><a href="vmhome.jsp">Home</a></li>
                <li><a href="space.jsp">Space Auction</a></li>
                <li><a href="userreq.jsp">User</a></li>
                <li><a href="userdet.jsp">User Details</a></li>
                <li><a href="logout.jsp">Logout</a></li>
            </ul>
        </nav>

```

```

<!-- Banner -->
    <!--
        To use a video as your background, set data-video to the name of
        your video without
        its extension (eg. images/banner). Your video must be available
        in both .mp4 and .webm
        formats to work correctly.
    -->
    <%
        session = request.getSession(false);
        String name = (String) session.getAttribute("name");
        String sql = "select * from message where
vmname='"+session.getAttribute("email")+"'";
        List<String>lt = Dao.getPlan(sql);
        Iterator<String>itr = lt.iterator();
    %>
        <sectionid="banner" data-video="images/banner">
            <divclass="inner">
                <h1style="margin-top: 50px;">Cloud Plans</h1>
                <tableclass="w3-table w3-black w3-text-
white"style="width: 850px;">
                    <tr><th>Cloud</th><th>Message</th><th>Pay</th></tr>
                    <%
                        while(itr.hasNext()){
                            String cn = itr.next();
                            String p = itr.next();
                            String a = itr.next();
                        %>
                    <tr><td><%=cn%></td><td><%=a %></td>
                    <td><a href="pay.jsp" class="w3-button w3-
white">Pay</a></td></tr>
                    <%} %>
                </table>
            </div>
        </section>

    <!-- Scripts -->
        <scriptsrc="assets/js/jquery.min.js"></script>
        <scriptsrc="assets/js/jquery.scrolly.min.js"></script>
        <scriptsrc="assets/js/jquery.scrollex.min.js"></script>
        <scriptsrc="assets/js/skel.min.js"></script>
        <scriptsrc="assets/js/util.js"></script>
        <scriptsrc="assets/js/main.js"></script>

    </body>

</html>

<!DOCTYPEHTML>
<html>
    <head>
        <title>VTJCC17_2019</title>
        <meta charset="utf-8"/>
        <meta name="viewport" content="width=device-width, initial-scale=1"/>
        <link rel="stylesheet" href="assets/css/main.css"/>
        <link rel="stylesheet" href="assets/css/w3.css"/>
    </head>

```



```

<body>

    <!-- Header -->
        <headerid="header"class="alt">
            <divclass="Logo"><a href="index.jsp" style="font-size:
30px;">Profit Maximization for Cloud Brokers in Cloud
Computing</a></div>
            <a href="#menu"class="toggle"><span>Menu</span></a>
        </header>

    <!-- Nav -->
        <navid="menu">
            <ulclass="Links">
                <li><a href="uhome.jsp">Home</a></li>
                <li><a href="vmdet.jsp">VM Details</a></li>
                <li><a href="store.jsp">Store Data</a></li>
                <li><a href="download.jsp">Download</a></li>
                <li><a href="logout.jsp">Logout</a></li>
            </ul>
        </nav>

    <!-- Banner -->
    <!--
        To use a video as your background, set data-video to the name of
your video without
        its extension (eg. images/banner). Your video must be available
in both .mp4 and .webm
        formats to work correctly.
    -->
    <%
session = request.getSession(false);
String name = (String) session.getAttribute("name");
%>
        <sectionid="banner"data-video="images/banner">
            <divclass="inner">
                <h1style="margin-top: 50px;">Welcome To <%=name
%></h1>
            </div>
        </section>

    <!-- Scripts -->
        <scriptsrc="assets/js/jquery.min.js"></script>
        <scriptsrc="assets/js/jquery.scrolly.min.js"></script>
        <scriptsrc="assets/js/jquery.scrollex.min.js"></script>
        <scriptsrc="assets/js/skel.min.js"></script>
        <scriptsrc="assets/js/util.js"></script>
        <scriptsrc="assets/js/main.js"></script>

</body>

</html>

<!DOCTYPEHTML>
<%@pageimport="java.util.Iterator"%>
<%@pageimport="com.dao.Dao"%>
<%@pageimport="java.util.List"%>
<html>

```

```

<head>
    <title>VTJCC17_2019</title>
    <meta charset="utf-8"/>
    <meta name="viewport" content="width=device-width, initial-scale=1"/>
    <link rel="stylesheet" href="assets/css/main.css"/>
    <link rel="stylesheet" href="assets/css/w3.css"/>
</head>
<body>

    <!-- Header -->
    <header id="header" class="alt">
        <div class="Logo"><a href="index.jsp" style="font-size:
30px;">Profit Maximization for Cloud Brokers in Cloud
Computing</a></div>
        <a href="#menu" class="toggle"><span>Menu</span></a>
    </header>

    <!-- Nav -->
    <nav id="menu">
        <ul class="links">
            <li><a href="chome.jsp">Home</a></li>
            <li><a href="vmrequest.jsp">VM Request</a></li>
            <li><a href="vmospace.jsp">Space Auction</a></li>
            <li><a href="vmdetails.jsp">VM Details</a></li>
            <li><a href="logout.jsp">Logout</a></li>
        </ul>
    </nav>

    <%
    session = request.getSession(false);
    String name = (String) session.getAttribute("name");
    String sql = "select * from cloudspace where
Cloudname='"+session.getAttribute("email")+"'";
    List<String>lt = Dao.getPlan(sql);
    Iterator<String>itr = lt.iterator();
    %>
        <section id="banner" data-video="images/banner">
            <div class="inner">
                <h1 style="margin-top: 50px;">Cloud Plan</h1>
                <table class="w3-table w3-black w3-text-
white" style="width: 850px;">

                    <tr><th>Cloud</th><th>Plan</th><th>Amount</th><th>Edit</th></tr>
                    <%
                    while(itr.hasNext()){
                        String cn = itr.next();
                        String p = itr.next();
                        String a = itr.next();
                        %>
                        <tr><td><%=cn%></td><td><%=p %></td><td><%=a %></td>
                        <td><a href="edit.jsp?cn=<%=cn%>&&p=<%=p
%>" class="w3-button w3-white">Edit</a></td></tr>
                        <%} %>
                    </table>
                    <a href="newplan.jsp">Add Plan</a>
                </div>
            </section>

```

```
<!-- Scripts -->
  <scriptsrc="assets/js/jquery.min.js"></script>
  <scriptsrc="assets/js/jquery.scrolly.min.js"></script>
  <scriptsrc="assets/js/jquery.scrollex.min.js"></script>
  <scriptsrc="assets/js/skel.min.js"></script>
  <scriptsrc="assets/js/util.js"></script>
  <scriptsrc="assets/js/main.js"></script></body>

</html>
```

CHAPTER 7

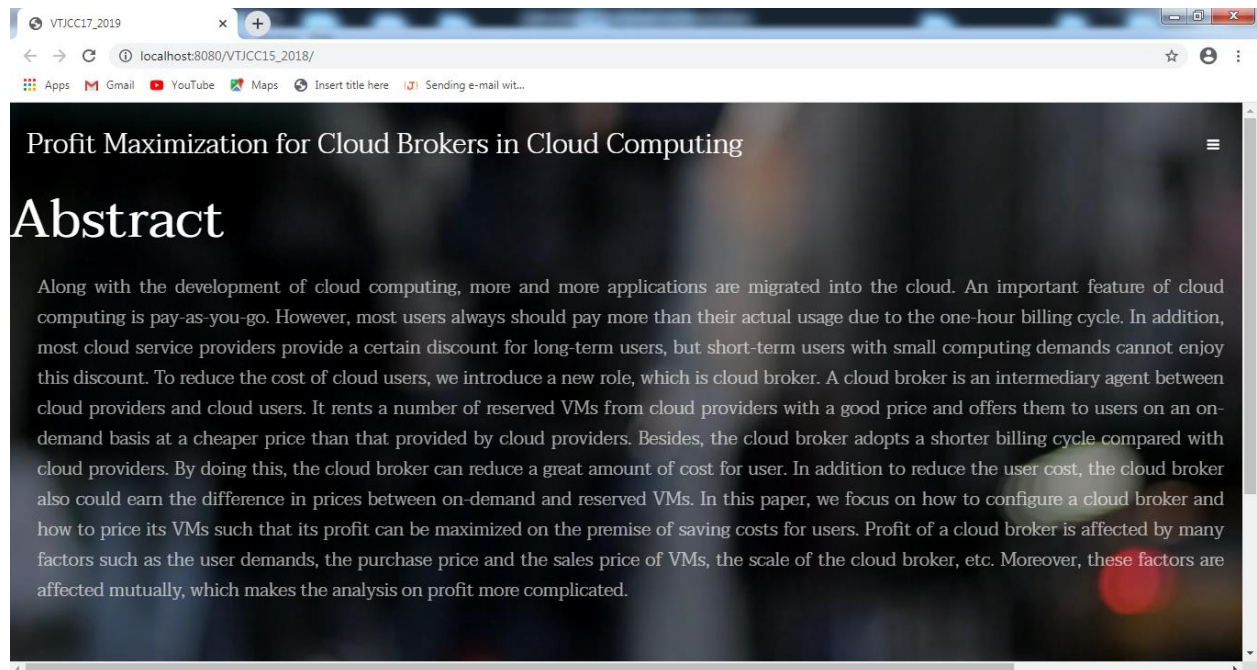
SNAPSHOTS

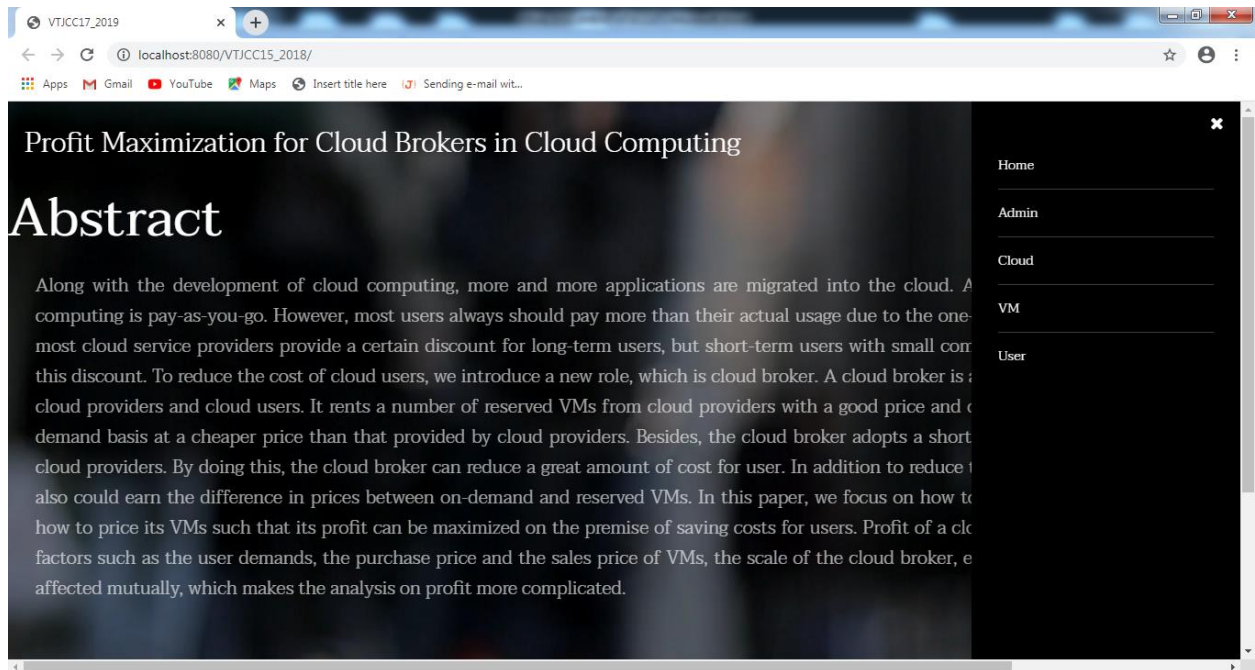
GENERAL

This project is implements like web application using COREJAVA and the Server process is maintained using the SOCKET & SERVERSOCKET and the Design part is played by Cascading Style Sheet.

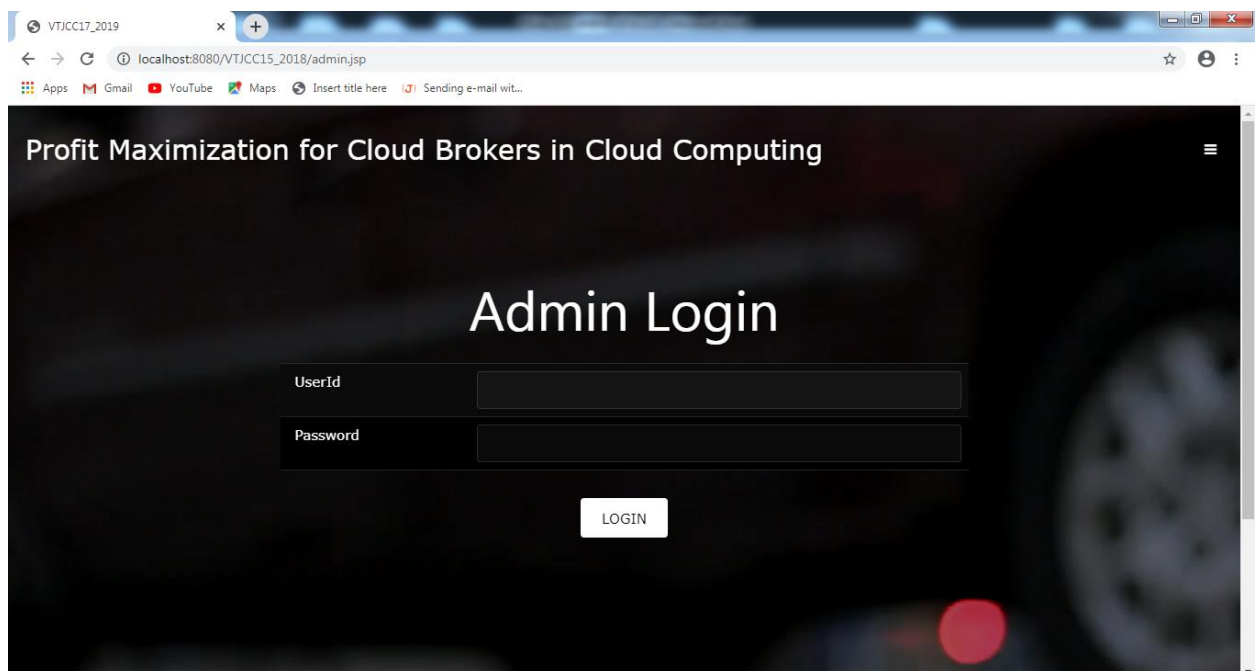
SNAPSHOTS

Index.jsp

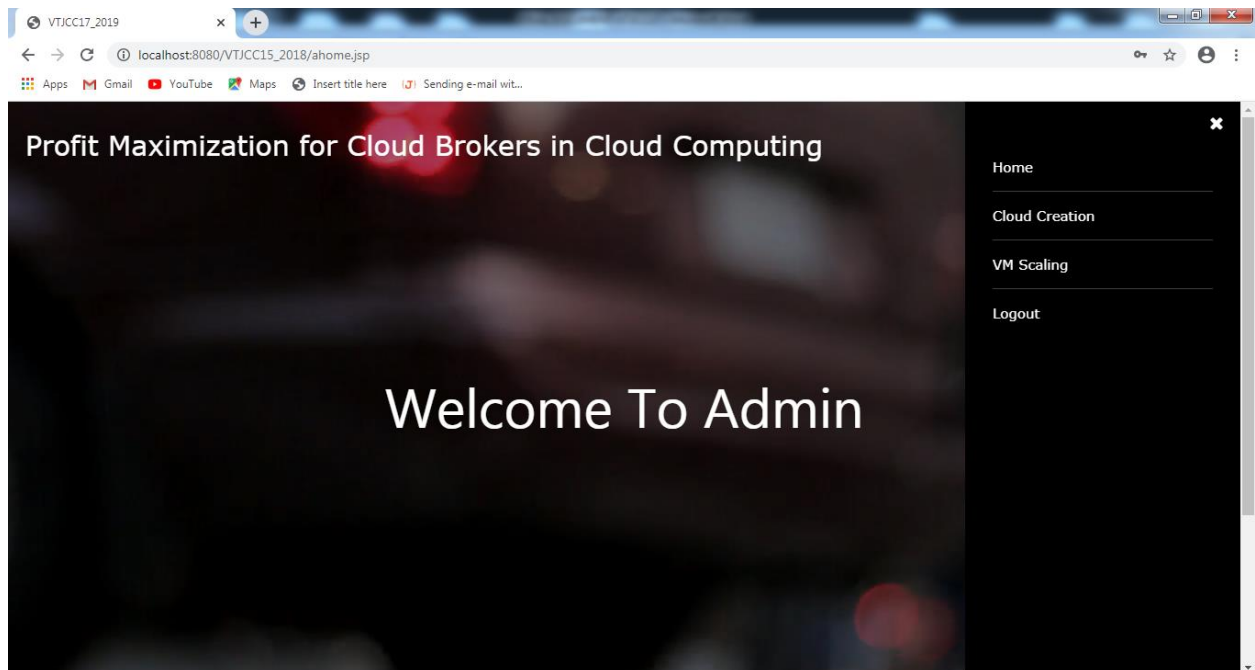




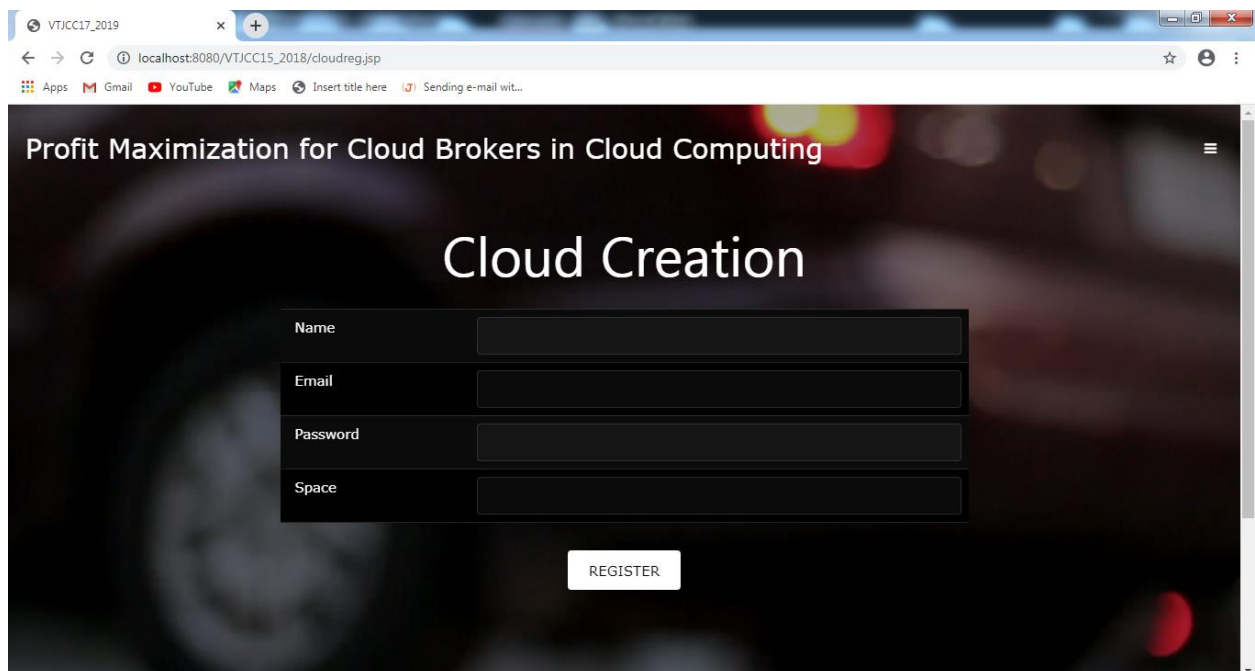
AdminLogin.jsp



AdminHome.jsp



CloudCreation.jsp



CloudLogin.jsp

Profit Maximization for Cloud Brokers in Cloud Computing

Cloud Login

Email

Password

LOGIN

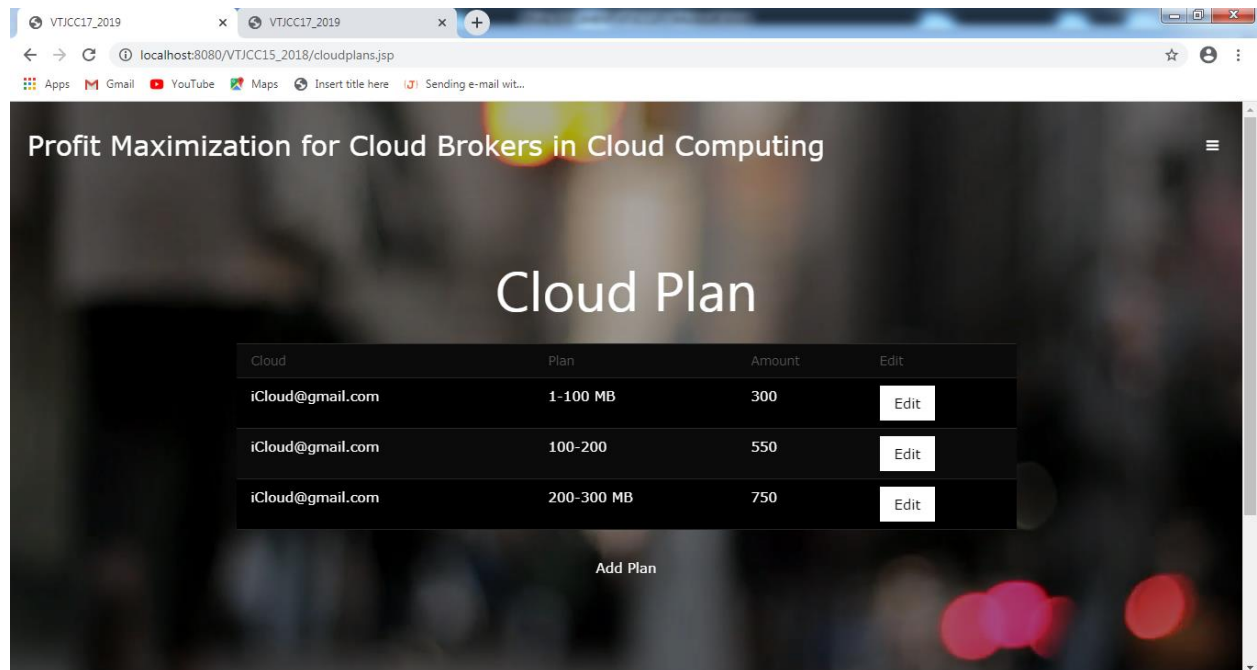
CloudHome.jsp

Profit Maximization for Cloud Brokers in Cloud Computing

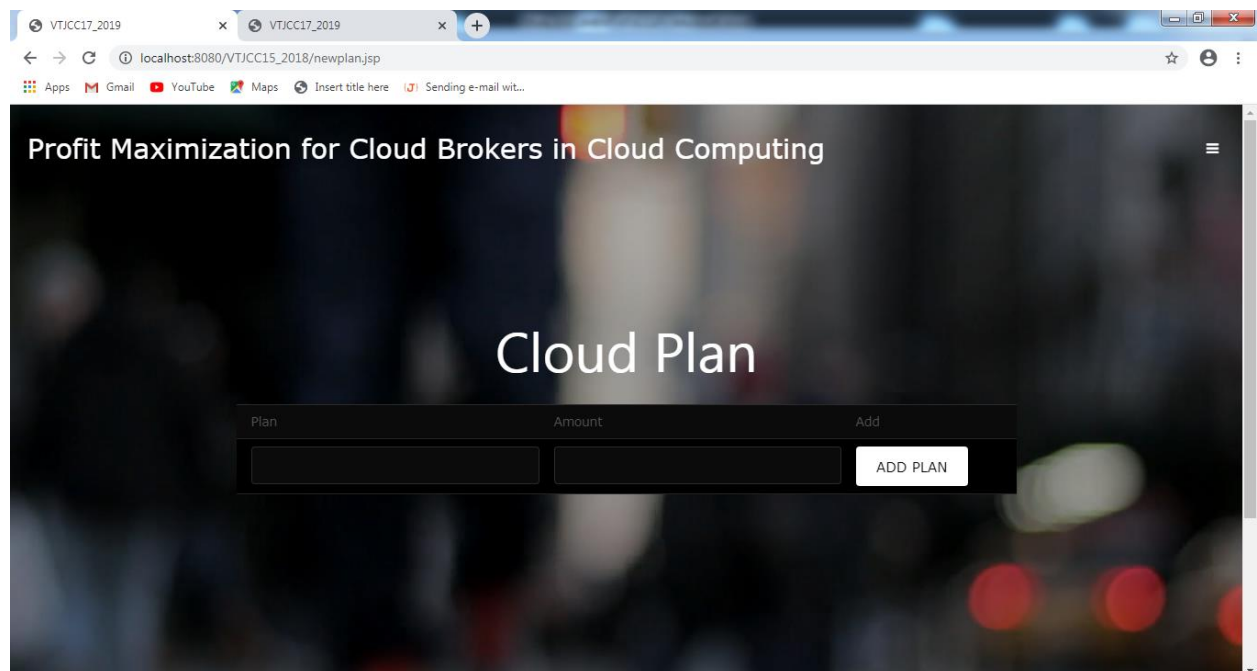
Welcome To iCloud

- Home
- VM Request
- Space Auction
- VM Details
- Cloud Plan
- Logout

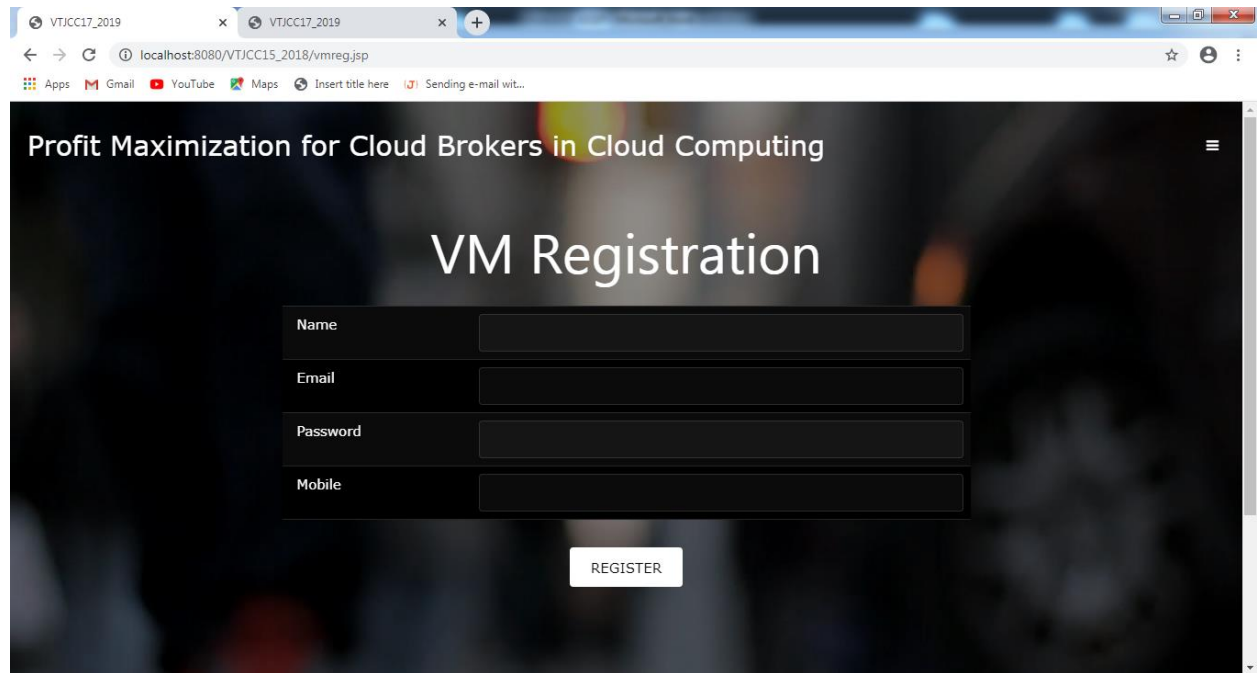
CloudPlan.jsp



CloudAddPlan.jsp



VMRegistration.jsp



The screenshot shows a web browser window with two tabs. The active tab is titled 'VTJCC17_2019' and the address bar shows 'localhost:8080/VTJCC15_2018/vmreg.jsp'. The page has a dark background with a blurred image of people. At the top, the text 'Profit Maximization for Cloud Brokers in Cloud Computing' is displayed. Below this, the title 'VM Registration' is centered. A registration form is centered on the page, consisting of four input fields labeled 'Name', 'Email', 'Password', and 'Mobile'. Below the form is a white button labeled 'REGISTER'.

Profit Maximization for Cloud Brokers in Cloud Computing

VM Registration

Name

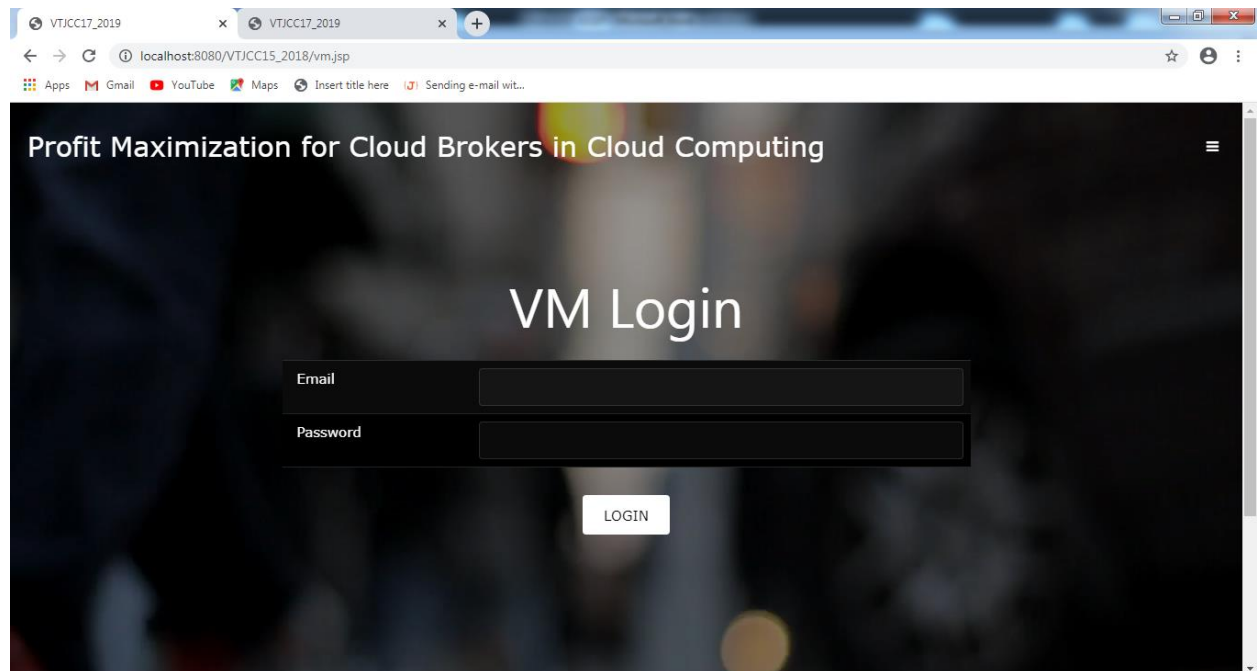
Email

Password

Mobile

REGISTER

VMLogin.jsp



The screenshot shows a web browser window with two tabs. The active tab is titled 'VTJCC17_2019' and the address bar shows 'localhost:8080/VTJCC15_2018/vm.jsp'. The page has a dark background with a blurred image of people. At the top, the text 'Profit Maximization for Cloud Brokers in Cloud Computing' is displayed. Below this, the title 'VM Login' is centered. A login form is centered on the page, consisting of two input fields labeled 'Email' and 'Password'. Below the form is a white button labeled 'LOGIN'.

Profit Maximization for Cloud Brokers in Cloud Computing

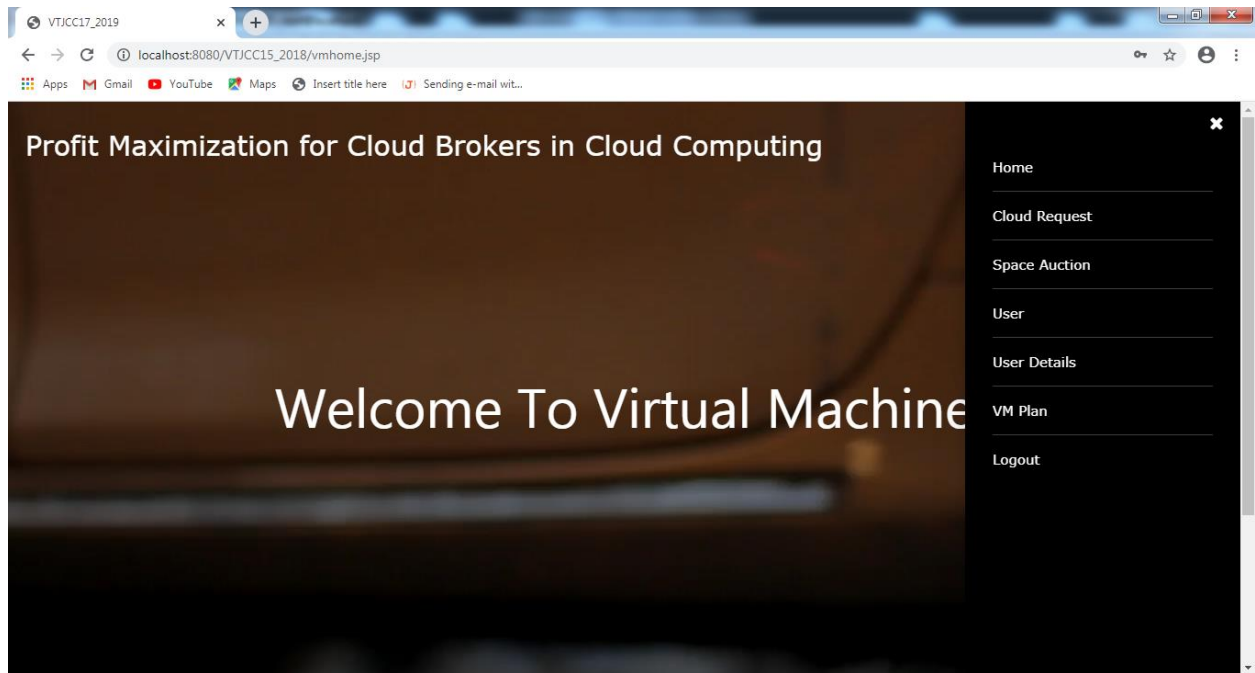
VM Login

Email

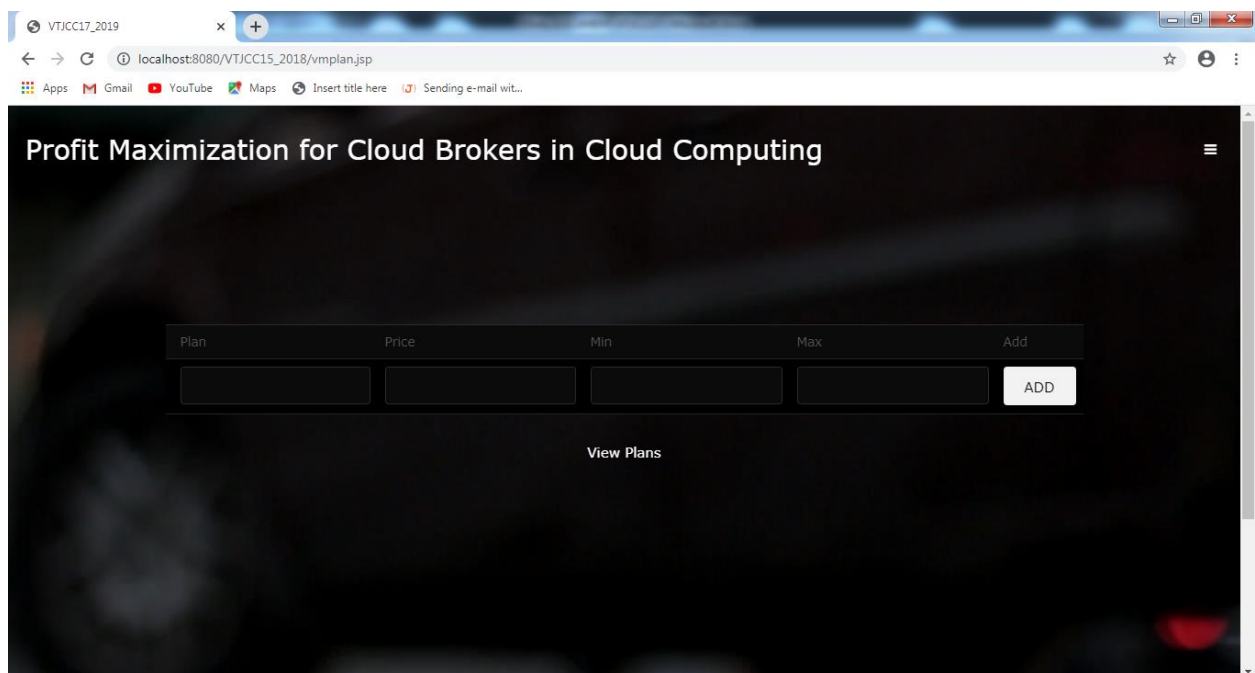
Password

LOGIN

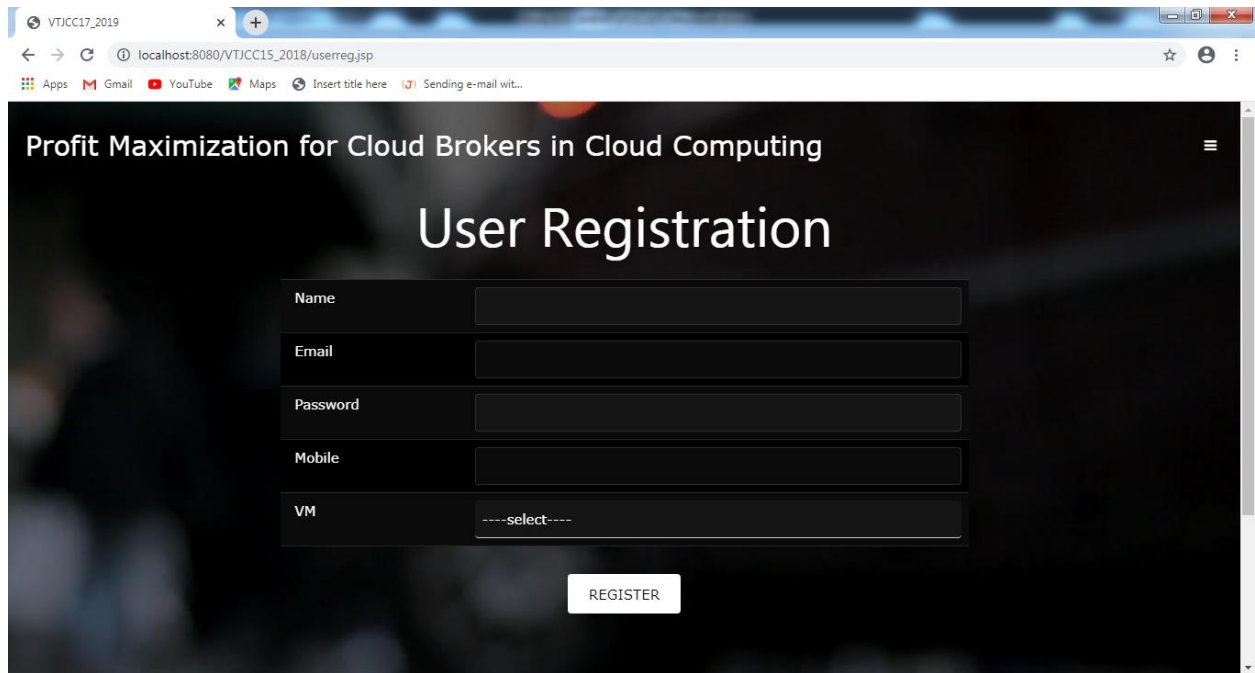
VMHome.jsp



VMPlan.jsp



UserRegistration.jsp



The screenshot shows a web browser window with the address bar displaying 'localhost:8080/VTJCC15_2018/userreg.jsp'. The page has a dark background with a blurred image of a car wheel. At the top, the text 'Profit Maximization for Cloud Brokers in Cloud Computing' is visible. Below this, the title 'User Registration' is centered in a large white font. A registration form is centered on the page, consisting of five input fields: 'Name', 'Email', 'Password', 'Mobile', and 'VM'. The 'VM' field is a dropdown menu with '----select----' as the placeholder. Below the form is a white 'REGISTER' button.

Profit Maximization for Cloud Brokers in Cloud Computing

User Registration

Name

Email

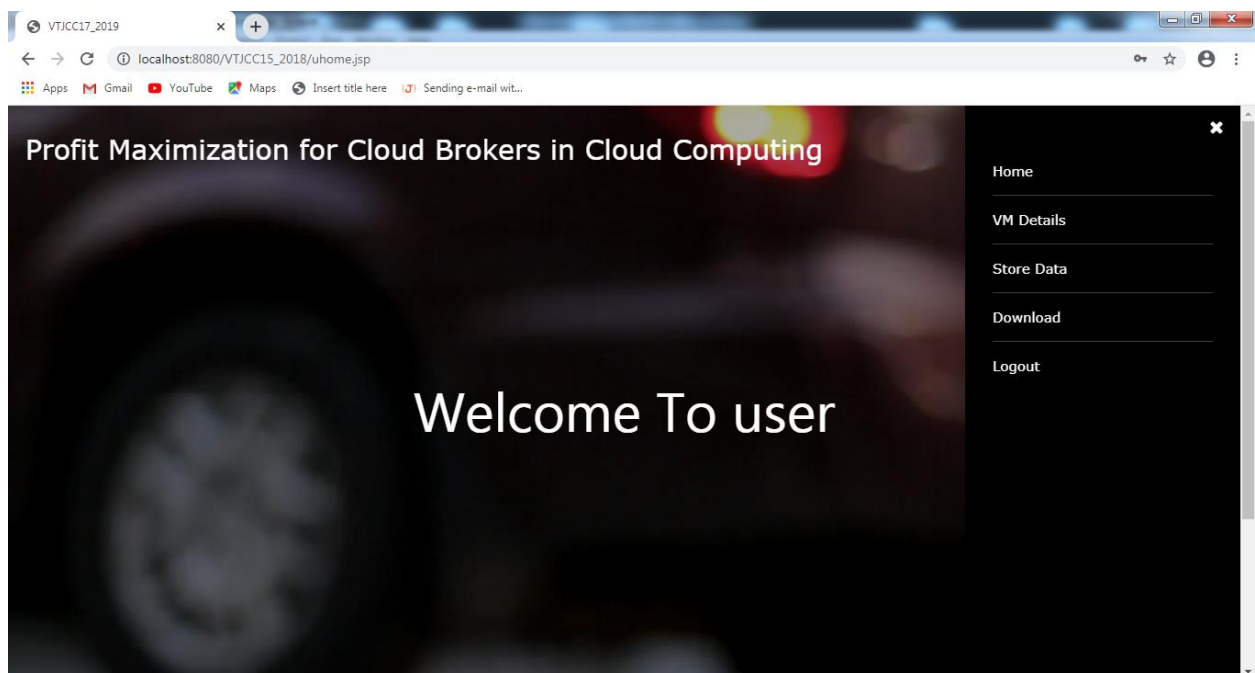
Password

Mobile

VM

REGISTER

UserLogin.jsp



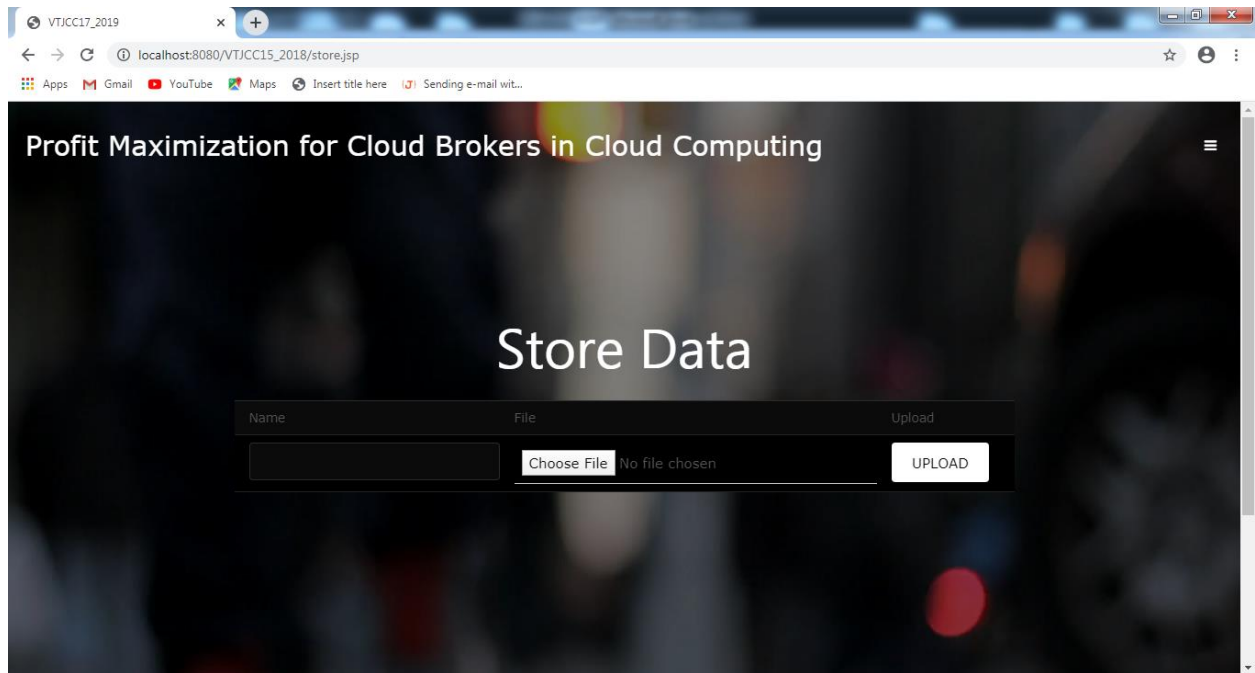
The screenshot shows a web browser window with the address bar displaying 'localhost:8080/VTJCC15_2018/uhome.jsp'. The page has a dark background with a blurred image of a car wheel. At the top, the text 'Profit Maximization for Cloud Brokers in Cloud Computing' is visible. Below this, the text 'Welcome To user' is centered in a large white font. On the right side of the page, there is a vertical navigation menu with a close button (X) at the top. The menu items are: 'Home', 'VM Details', 'Store Data', 'Download', and 'Logout'.

Profit Maximization for Cloud Brokers in Cloud Computing

Welcome To user

- Home
- VM Details
- Store Data
- Download
- Logout

StoreData.jsp



The screenshot shows a web browser window with the address bar displaying 'localhost:8080/VTJCC15_2018/store.jsp'. The page has a dark background with a blurred image of people. At the top, the text 'Profit Maximization for Cloud Brokers in Cloud Computing' is visible. Below this, the title 'Store Data' is centered. A form is displayed with three columns: 'Name', 'File', and 'Upload'. The 'Name' column has a text input field. The 'File' column contains a 'Choose File' button and the text 'No file chosen'. The 'Upload' column has an 'UPLOAD' button.

Name	File	Upload
<input type="text"/>	<input type="button" value="Choose File"/> No file chosen	<input type="button" value="UPLOAD"/>

CHAPTER 8

SOFTWARE TESTING

8.1 GENERAL

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

8.2 DEVELOPING METHODOLOGIES

The test process is initiated by developing a comprehensive plan to test the general functionality and special features on a variety of platform combinations. Strict quality control procedures are used. The process verifies that the application meets the requirements specified in the system requirements document and is bug free. The following are the considerations used to develop the framework from developing the testing methodologies.

8.3 TYPES OF TESTS

8.3.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program input produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. 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.

8.3.2 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.

8.3.3 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.

8.3.4 PERFORMANCE TEST

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

8.3.5 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.

8.3.6 ACCEPTANCE TESTING

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

Acceptance testing for Data Synchronization:

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

8.2.7 BUILD THE TEST PLAN

Any project can be divided into units that can be further performed for detailed processing. Then a testing strategy for each of this unit is carried out. Unit testing helps to identify the possible bugs in the individual component, so the component that has bugs can be identified and can be rectified from errors.

CHAPTER 9

CONCLUSION

We focus 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 fine-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.

CHAPTER 10

REFERENCE

- [1] Buyya Rajkumar, Broberg James, and Andrzej M Goscinski. Cloud computing: Principles and paradigms. 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 Comput. 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. IEEE Computational Intelligence Magazine, 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 Roderio-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] Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal, James Broberg, and Ivona Brandic. 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.
- [10] Wei Wang, Di Niu, Baochun Li, and Ben Liang. Dynamic cloud resource reservation via cloud brokerage. In IEEE International Conference on Distributed Computing Systems, pages 400–409, 2013.
- [11] Jing Mei, Kenli Li, Aijia Ouyang, and Keqin Li. A profit maximization scheme with guaranteed quality of service in cloud computing. IEEE Transactions on Computers, 64(11):3064–3078, 2015.
- [12] Sivadon Chaisiri, Bu-Sung Lee, and Dusit Niyato. Optimization of resource provisioning cost in cloud computing. IEEE Transactions on Services Computing, 5(2):164–177, 2012.
- [13] Microsoft Azure. <http://www.microsoft.com/windowsazure>, 2017.
- [14] Owen Rogers and Dave Cliff. A financial brokerage model for cloud computing. Journal of Cloud Computing, 1(1):1–12, 2012.
- [15] Daniele D’Agostino, Antonella Galizia, Andrea Clematis, Matteo Mangini, Ivan Porro, and Alfonso Quarati. A qosaware broker for hybrid clouds. Computing, 95(1):89–109, 2013.
- [16] Dhaval Limbani and Bhavesh Oza. A proposed service broker strategy in cloudanalyst for cost-effective data center selection. International Journal of Engineering Research and Applications, 2(1):793–797, 2014.