Getting Started in the Cloud with AWS/Google

Objective: To learn about Compute Cloud which allows users to rent virtual machines on which they can to run their own computer applications.

Task: Create AWS instances, connecting to Your Linux Instance Using SSH and Run a Linux OS in a Virtual Machine

1.1 Procedure

1. We have used Qwiklabs.
2. From Qwiklabs console we have gone to Amazon console
3. Create and launch an instance from the Amazon EC2 console.

   Select the distribution of Linux you want to configure in your instance.

   Select the Key Pair you want to use to log into the instance.

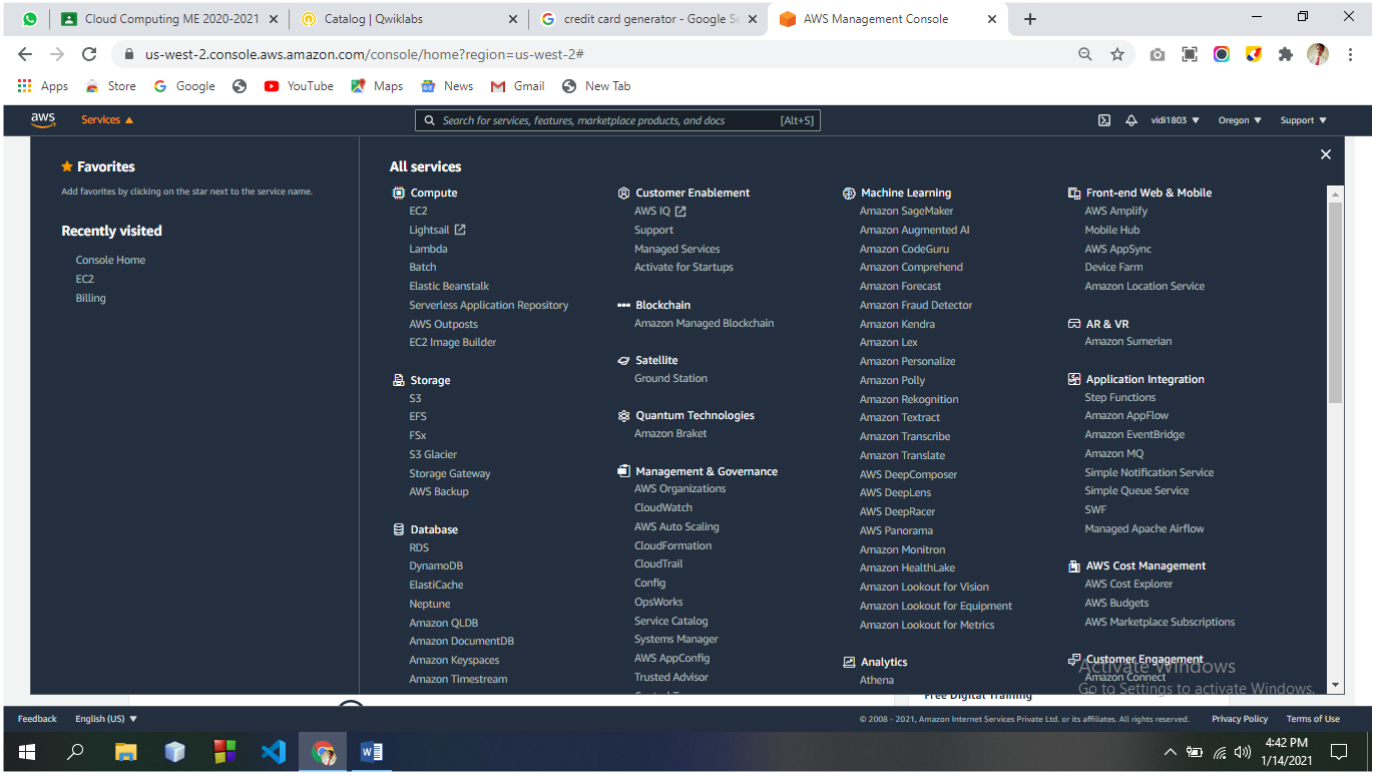
4. Download .pem key and change it's access permissions (You cannot access your instance until file permissions for your key is r–).

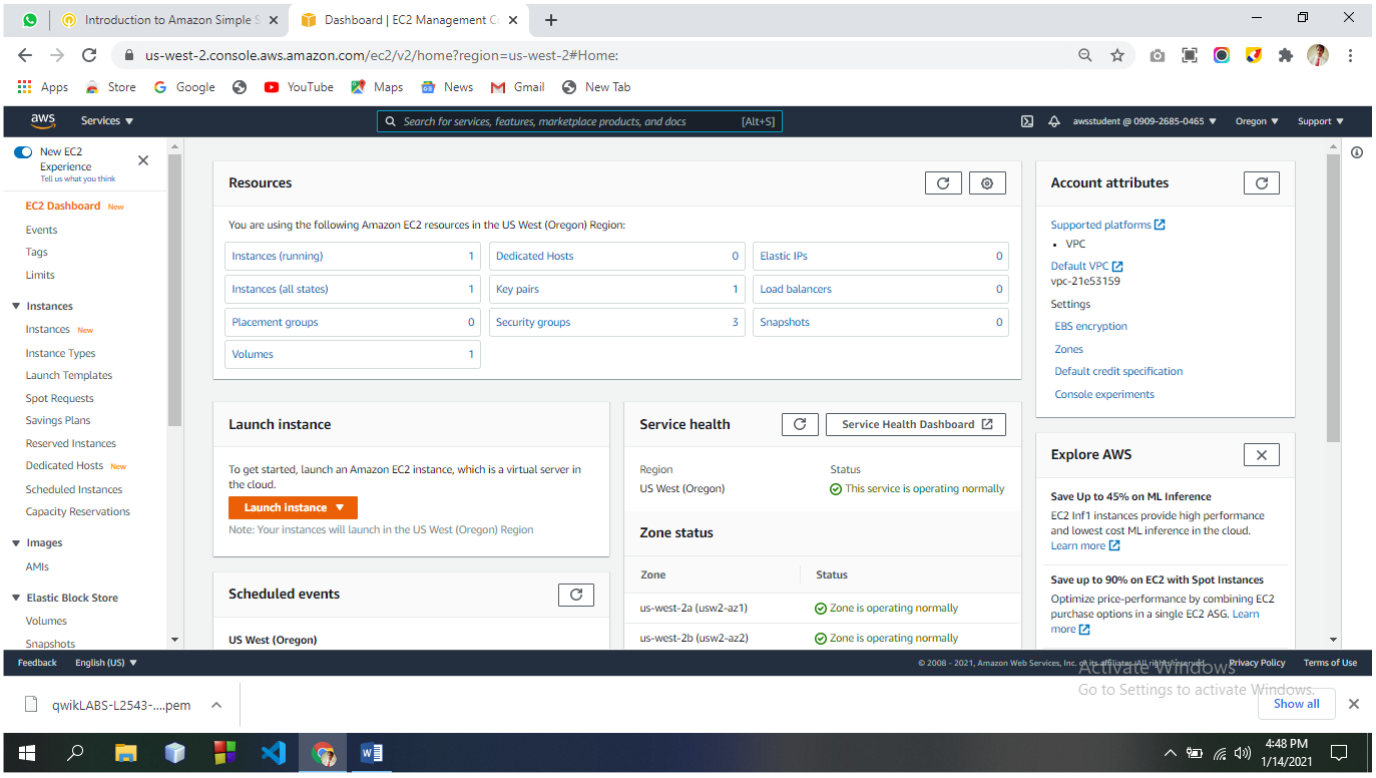
   chmod 400 KeyPairName.pem

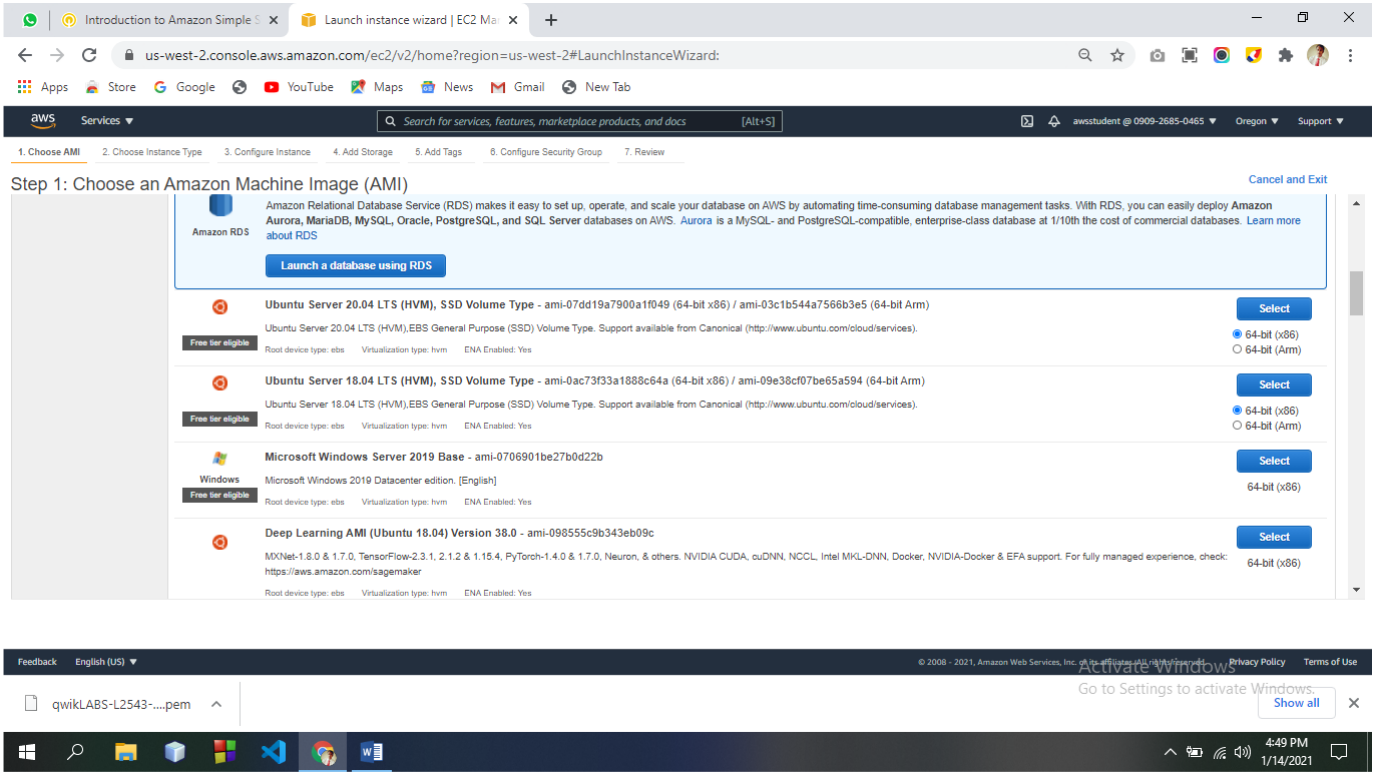
5. Open your terminal and navigate to the directory containing the key.
6. Remote login to your instance using ssh.
7. Check specifications of your instance using **uname**, **lshw** etc

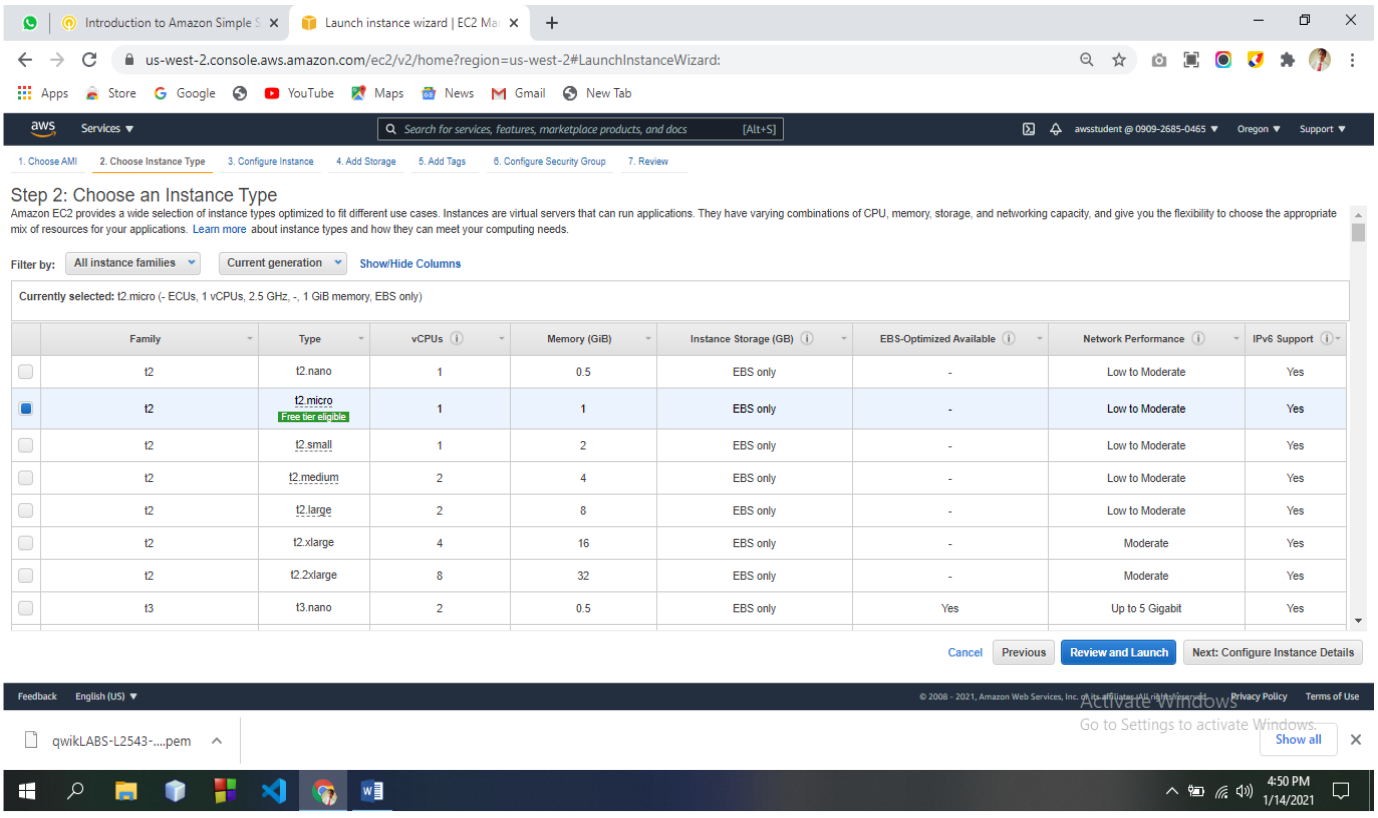
   ssh -i KeyName.pem LinuxDistribution@IPAddressOfYourInstance

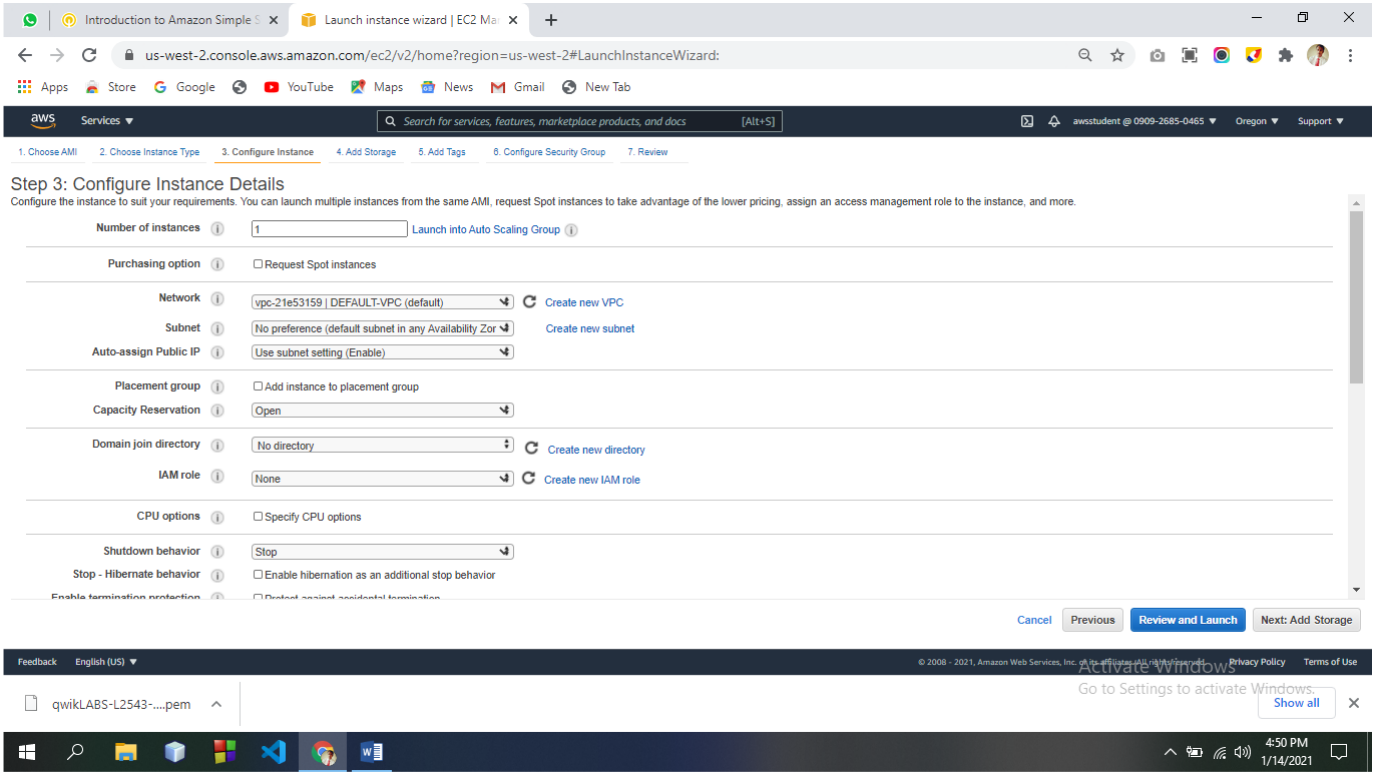
1.2 Demonstration

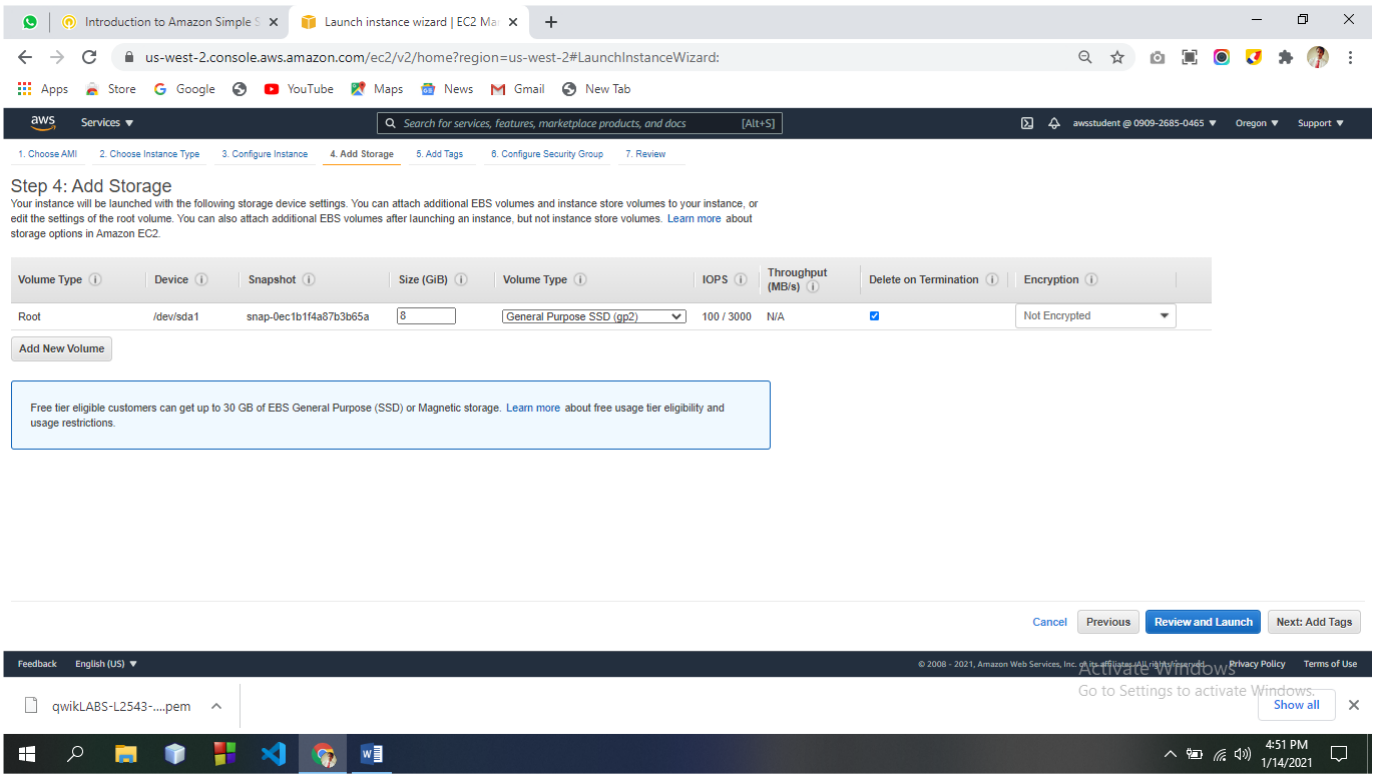

Cloud Computing ME 2020-2021
Catalog | Qwiklabs
credit card generator - Google S
AWS Management Console
us-west-2.console.aws.amazon.com/console/home?region=us-west-2#
Apps Store Google YouTube Maps News Gmail New Tab
aws Services
Search for services, features, marketplace products, and docs [Alt+S]
vidi1803 Oregon Support
Favorites
Add favorites by clicking on the star next to the service name.
Recently visited
Console Home
EC2
Billing
All services
Compute
EC2
Lightsail
Lambda
Batch
Elastic Beanstalk
Serverless Application Repository
AWS Outposts
EC2 Image Builder
Storage
S3
EFS
FSx
S3 Glacier
Storage Gateway
AWS Backup
Database
RDS
DynamoDB
ElastiCache
Neptune
Amazon QLDB
Amazon DocumentDB
Amazon Keyspaces
Amazon Timestream
Customer Enablement
AWS IQ
Support
Managed Services
Activate for Startups
Blockchain
Amazon Managed Blockchain
Satellite
Ground Station
Quantum Technologies
Amazon Braket
Management & Governance
AWS Organizations
CloudWatch
AWS Auto Scaling
CloudFormation
CloudTrail
Config
OpsWorks
Service Catalog
Systems Manager
AWS AppConfig
Trusted Advisor
Machine Learning
Amazon SageMaker
Amazon Augmented AI
Amazon CodeGuru
Amazon Comprehend
Amazon Forecast
Amazon Fraud Detector
Amazon Kendra
Amazon Lex
Amazon Personalize
Amazon Polly
Amazon Rekognition
Amazon Textract
Amazon Transcribe
Amazon Translate
AWS DeepComposer
AWS DeepLens
AWS DeepRacer
AWS Panorama
Amazon Monitron
Amazon HealthLake
Amazon Lookout for Vision
Amazon Lookout for Equipment
Amazon Lookout for Metrics
Analytics
Athena
Front-end Web & Mobile
AWS Amplify
Mobile Hub
AWS AppSync
Device Farm
Amazon Location Service
AR & VR
Amazon Sumerian
Application Integration
Step Functions
Amazon AppFlow
Amazon EventBridge
Amazon MQ
Simple Notification Service
Simple Queue Service
SWF
Managed Apache Airflow
AWS Cost Management
AWS Cost Explorer
AWS Budgets
AWS Marketplace Subscriptions
Customer Engagement
Amazon Connect
Activate Windows
Go to Settings to activate Windows.
Feedback English (US)
© 2008 - 2021, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.
Privacy Policy Terms of Use
4:42 PM
1/14/2021

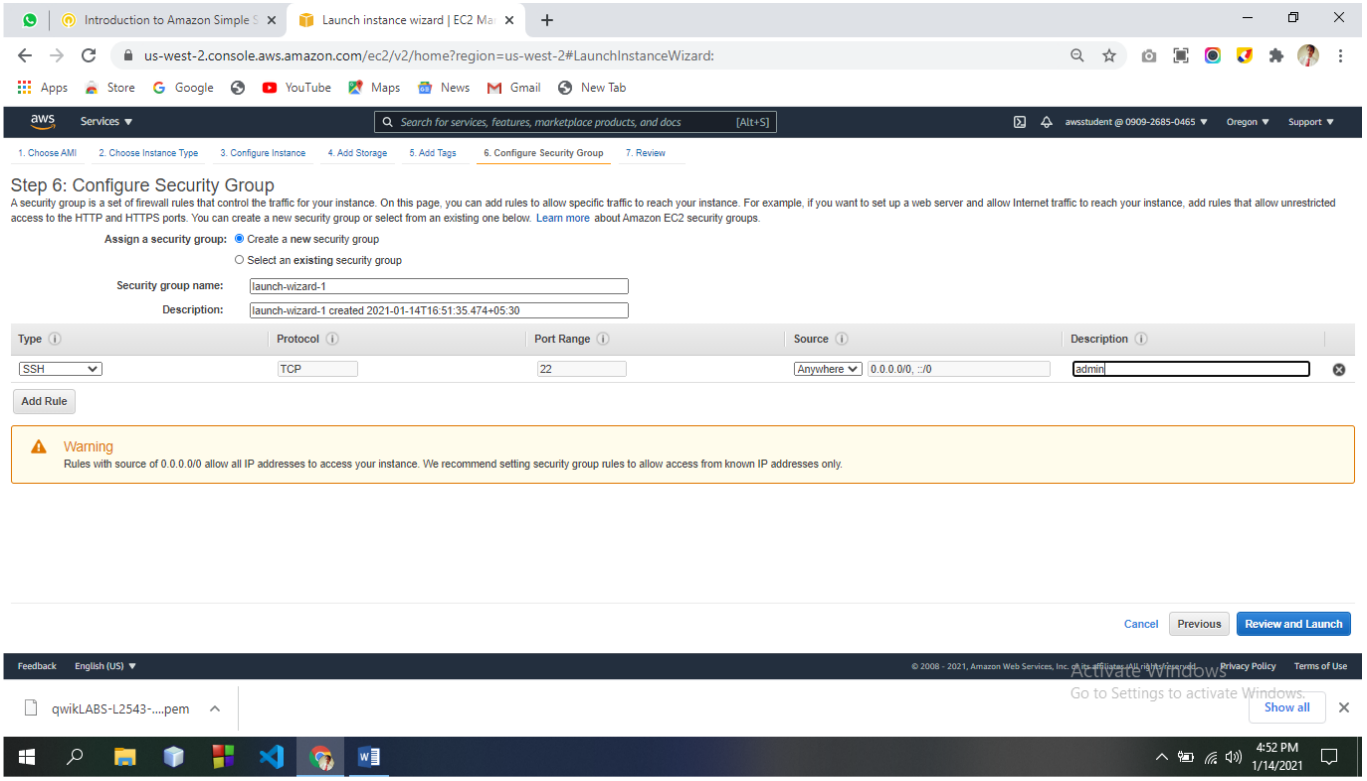

Introduction to Amazon Simple S
Dashboard | EC2 Management C
us-west-2.console.aws.amazon.com/ec2/v2/home?region=us-west-2#Home:
Apps Store Google YouTube Maps News Gmail New Tab
aws Services
Search for services, features, marketplace products, and docs [Alt+S]
awsstudent @ 0909-2685-0465 Oregon Support
New EC2 Experience
Tell us what you think
EC2 Dashboard New
Events
Tags
Limits
Instances
Instances New
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts New
Scheduled Instances
Capacity Reservations
Images
AMIs
Elastic Block Store
Volumes
Snapshots
Resources
You are using the following Amazon EC2 resources in the US West (Oregon) Region:
Instances (running) 1
Dedicated Hosts 0
Elastic IPs 0
Instances (all states) 1
Key pairs 1
Load balancers 0
Placement groups 0
Security groups 3
Snapshots 0
Volumes 1
Account attributes
Supported platforms
VPC
Default VPC
vpc-21e53159
Settings
EBS encryption
Zones
Default credit specification
Console experiments
Launch instance
To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.
Launch instance
Note: Your instances will launch in the US West (Oregon) Region
Service health
Service Health Dashboard
Region
US West (Oregon)
Status
This service is operating normally
Zone status
Zone
Status
us-west-2a (usw2-az1)
Zone is operating normally
us-west-2b (usw2-az2)
Zone is operating normally
Explore AWS
Save Up to 45% on ML Inference
EC2 Inf1 instances provide high performance and lowest cost ML inference in the cloud.
Learn more
Save up to 90% on EC2 with Spot Instances
Optimize price-performance by combining EC2 purchase options in a single EC2 ASG.
Learn more
Scheduled events
US West (Oregon)
Feedback English (US)
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy Terms of Use
Activate Windows
Go to Settings to activate Windows.
qwikLABS-L2543-....pem
Show all
4:48 PM
1/14/2021

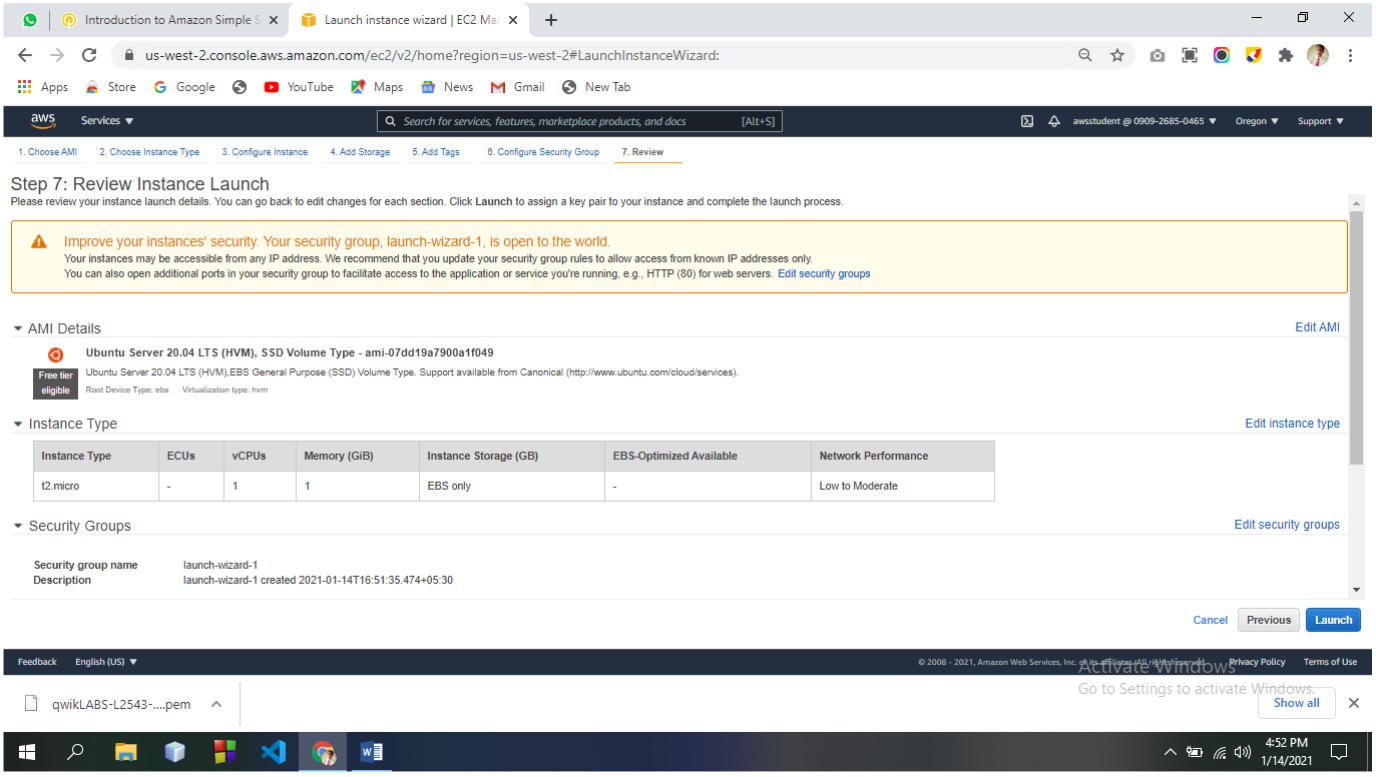

Introduction to Amazon Simple S
Launch instance wizard | EC2 Ma
us-west-2.console.aws.amazon.com/ec2/v2/home?region=us-west-2#LaunchInstanceWizard:
Apps Store Google YouTube Maps News Gmail New Tab
aws Services Search for services, features, marketplace products, and docs [Alt+S]
awsstudent @ 0909-2685-0465 Oregon Support
1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review
Step 1: Choose an Amazon Machine Image (AMI)
Cancel and Exit
Amazon RDS
Amazon Relational Database Service (RDS) makes it easy to set up, operate, and scale your database on AWS by automating time-consuming database management tasks. With RDS, you can easily deploy Amazon Aurora, MariaDB, MySQL, Oracle, PostgreSQL, and SQL Server databases on AWS. Aurora is a MySQL- and PostgreSQL-compatible, enterprise-class database at 1/10th the cost of commercial databases. Learn more about RDS
Launch a database using RDS
Ubuntu Server 20.04 LTS (HVM), SSD Volume Type - ami-07dd19a7900a1f049 (64-bit x86) / ami-03c1b544a7566b3e5 (64-bit Arm)
Ubuntu Server 20.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).
Free tier eligible
Root device type: ebs Virtualization type: hvm ENA Enabled: Yes
Select
64-bit (x86)
64-bit (Arm)
Ubuntu Server 18.04 LTS (HVM), SSD Volume Type - ami-0ac73f33a1888c64a (64-bit x86) / ami-09e38cf07be65a594 (64-bit Arm)
Ubuntu Server 18.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).
Free tier eligible
Root device type: ebs Virtualization type: hvm ENA Enabled: Yes
Select
64-bit (x86)
64-bit (Arm)
Microsoft Windows Server 2019 Base - ami-0706901be27b0d22b
Windows
Free tier eligible
Microsoft Windows 2019 Datacenter edition. [English]
Root device type: ebs Virtualization type: hvm ENA Enabled: Yes
Select
64-bit (x86)
Deep Learning AMI (Ubuntu 18.04) Version 38.0 - ami-098555c9b343eb09c
MXNet-1.8.0 & 1.7.0, TensorFlow-2.3.1, 2.1.2 & 1.15.4, PyTorch-1.4.0 & 1.7.0, Neuron, & others. NVIDIA CUDA, cuDNN, NCCL, Intel MKL-DNN, Docker, NVIDIA-Docker & EFA support. For fully managed experience, check: https://aws.amazon.com/sagemaker
Root device type: ebs Virtualization type: hvm ENA Enabled: Yes
Select
64-bit (x86)
Feedback English (US)
Privacy Policy Terms of Use
qwikLABS-L2543-....pem
Show all
Activate Windows
Go to Settings to activate Windows.
4:49 PM
1/14/2021


Introduction to Amazon Simple S
Launch instance wizard | EC2 Ma
us-west-2.console.aws.amazon.com/ec2/v2/home?region=us-west-2#LaunchInstanceWizard:
Apps Store Google YouTube Maps News Gmail New Tab
aws Services Search for services, features, marketplace products, and docs [Alt+S]
awsstudent @ 0909-2685-0465 Oregon Support
1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review
Step 2: Choose an Instance Type
Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. Learn more about instance types and how they can meet your computing needs.
Filter by: All instance families Current generation Show/Hide Columns
Currently selected: t2.micro (- ECUs, 1 vCPUs, 2.5 GHz, -, 1 GiB memory, EBS only)
Family | Type | vCPUs | Memory (GiB) | Instance Storage (GB) | EBS-Optimized Available | Network Performance | IPv6 Support
t2 | t2.nano | 1 | 0.5 | EBS only | - | Low to Moderate | Yes
t2 | t2.micro Free tier eligible | 1 | 1 | EBS only | - | Low to Moderate | Yes
t2 | t2.small | 1 | 2 | EBS only | - | Low to Moderate | Yes
t2 | t2.medium | 2 | 4 | EBS only | - | Low to Moderate | Yes
t2 | t2.large | 2 | 8 | EBS only | - | Low to Moderate | Yes
t2 | t2.xlarge | 4 | 16 | EBS only | - | Moderate | Yes
t2 | t2.2xlarge | 8 | 32 | EBS only | - | Moderate | Yes
t3 | t3.nano | 2 | 0.5 | EBS only | Yes | Up to 5 Gigabit | Yes
Cancel Previous Review and Launch Next: Configure Instance Details
Feedback English (US)
Privacy Policy Terms of Use
qwikLABS-L2543-....pem
Show all
Activate Windows
Go to Settings to activate Windows.
4:50 PM
1/14/2021


Introduction to Amazon Simple S
Launch instance wizard | EC2 Ma
us-west-2.console.aws.amazon.com/ec2/v2/home?region=us-west-2#LaunchInstanceWizard:
Apps Store Google YouTube Maps News Gmail New Tab
aws Services Search for services, features, marketplace products, and docs [Alt+S]
awsstudent @ 0909-2685-0465 Oregon Support
1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review
Step 3: Configure Instance Details
Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.
Number of instances 1 Launch into Auto Scaling Group
Purchasing option Request Spot instances
Network vpc-21e53159 | DEFAULT-VPC (default) Create new VPC
Subnet No preference (default subnet in any Availability Zor Create new subnet
Auto-assign Public IP Use subnet setting (Enable)
Placement group Add instance to placement group
Capacity Reservation Open
Domain join directory No directory Create new directory
IAM role None Create new IAM role
CPU options Specify CPU options
Shutdown behavior Stop
Stop - Hibernate behavior Enable hibernation as an additional stop behavior
Cancel Previous Review and Launch Next: Add Storage
Feedback English (US)
Privacy Policy Terms of Use
Activate Windows
Go to Settings to activate Windows.
qwikLABS-L2543-....pem
Show all
4:50 PM
1/14/2021


Introduction to Amazon Simple S
Launch instance wizard | EC2 Ma
us-west-2.console.aws.amazon.com/ec2/v2/home?region=us-west-2#LaunchInstanceWizard:
Apps Store Google YouTube Maps News Gmail New Tab
aws Services Search for services, features, marketplace products, and docs [Alt+S]
awsstudent @ 0909-2685-0465 Oregon Support
1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review
Step 4: Add Storage
Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. Learn more about storage options in Amazon EC2.
Volume Type Device Snapshot Size (GiB) Volume Type IOPS Throughput (MB/s) Delete on Termination Encryption
Root /dev/sda1 snap-0ec1b1f4a87b3b65a 8 General Purpose SSD (gp2) 100 / 3000 N/A Not Encrypted
Add New Volume
Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. Learn more about free usage tier eligibility and usage restrictions.
Cancel Previous Review and Launch Next: Add Tags
Feedback English (US)
Privacy Policy Terms of Use
Activate Windows
Go to Settings to activate Windows.
qwikLABS-L2543-....pem
Show all
4:51 PM
1/14/2021


Introduction to Amazon Simple S
Launch instance wizard | EC2 Ma
us-west-2.console.aws.amazon.com/ec2/v2/home?region=us-west-2#LaunchInstanceWizard:
Apps Store Google YouTube Maps News Gmail New Tab
aws Services Search for services, features, marketplace products, and docs [Alt+S]
awsstudent @ 0909-2685-0465 Oregon Support
1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review
Step 6: Configure Security Group
A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. Learn more about Amazon EC2 security groups.
Assign a security group: Create a new security group
Select an existing security group
Security group name: launch-wizard-1
Description: launch-wizard-1 created 2021-01-14T16:51:35.474+05:30
Type Protocol Port Range Source Description
SSH TCP 22 Anywhere 0.0.0.0/0, ::/0 admin
Add Rule
Warning
Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.
Cancel Previous Review and Launch
Feedback English (US)
Privacy Policy Terms of Use
qwikLABS-L2543-....pem
Show all
4:52 PM
1/14/2021


Introduction to Amazon Simple S
Launch instance wizard | EC2 Ma
us-west-2.console.aws.amazon.com/ec2/v2/home?region=us-west-2#LaunchInstanceWizard:
Apps Store Google YouTube Maps News Gmail New Tab
aws Services Search for services, features, marketplace products, and docs [Alt+S]
awsstudent @ 0909-2685-0465 Oregon Support
1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review
Step 7: Review Instance Launch
Please review your instance launch details. You can go back to edit changes for each section. Click Launch to assign a key pair to your instance and complete the launch process.
Improve your instances' security. Your security group, launch-wizard-1, is open to the world.
Your instances may be accessible from any IP address. We recommend that you update your security group rules to allow access from known IP addresses only.
You can also open additional ports in your security group to facilitate access to the application or service you're running, e.g., HTTP (80) for web servers. Edit security groups
AMI Details
Edit AMI
Ubuntu Server 20.04 LTS (HVM), SSD Volume Type - ami-07dd19a7900a1f049
Free tier eligible
Ubuntu Server 20.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).
Root Device Type: ebs Virtualization type: hvm
Instance Type
Edit instance type
Instance Type ECUs vCPUs Memory (GiB) Instance Storage (GB) EBS-Optimized Available Network Performance
t2.micro - 1 1 EBS only - Low to Moderate
Security Groups
Edit security groups
Security group name launch-wizard-1
Description launch-wizard-1 created 2021-01-14T16:51:35.474+05:30
Cancel Previous Launch
Feedback English (US)
Privacy Policy Terms of Use
qwikLABS-L2543-....pem
Show all
4:52 PM
1/14/2021

Launch Status

Your instances are now launching

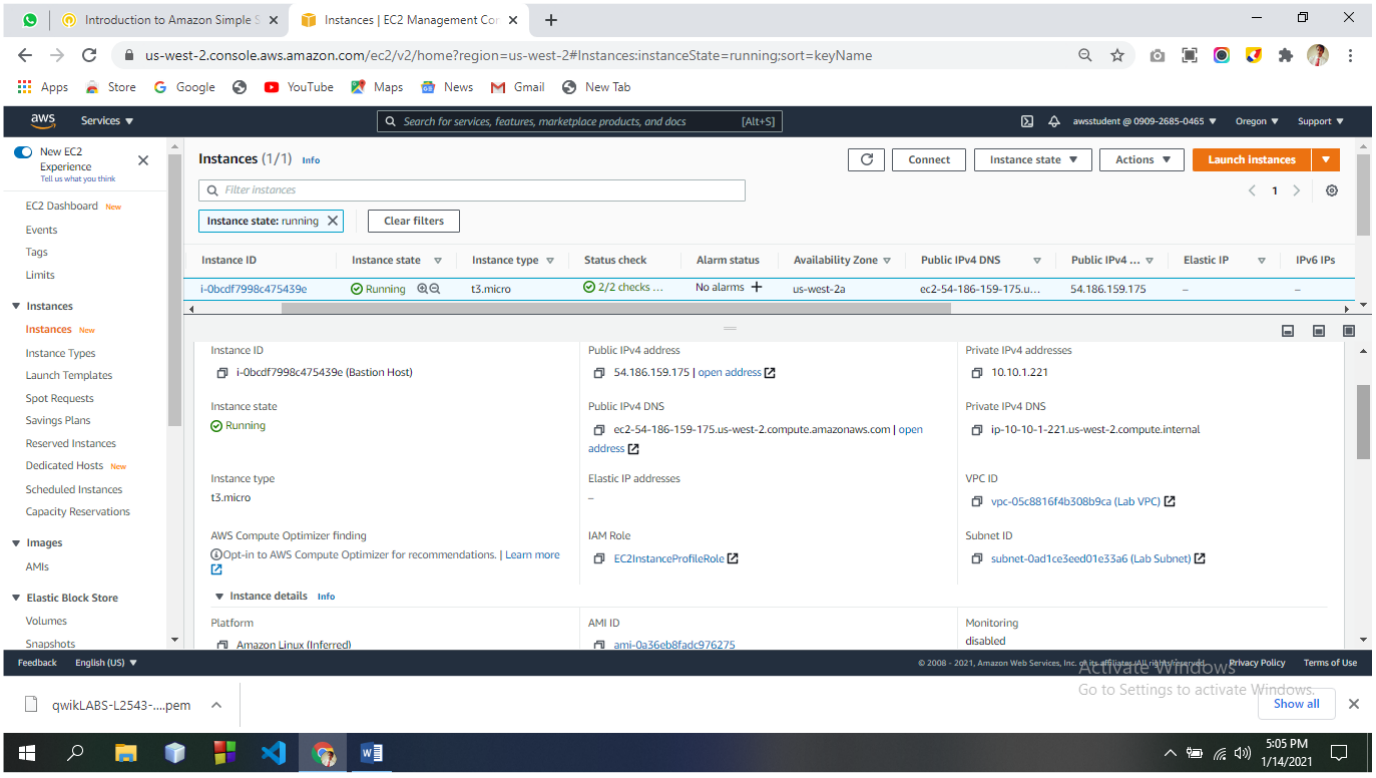
The following instance launches have been initiated: i-4c2c3cff Hide launch log

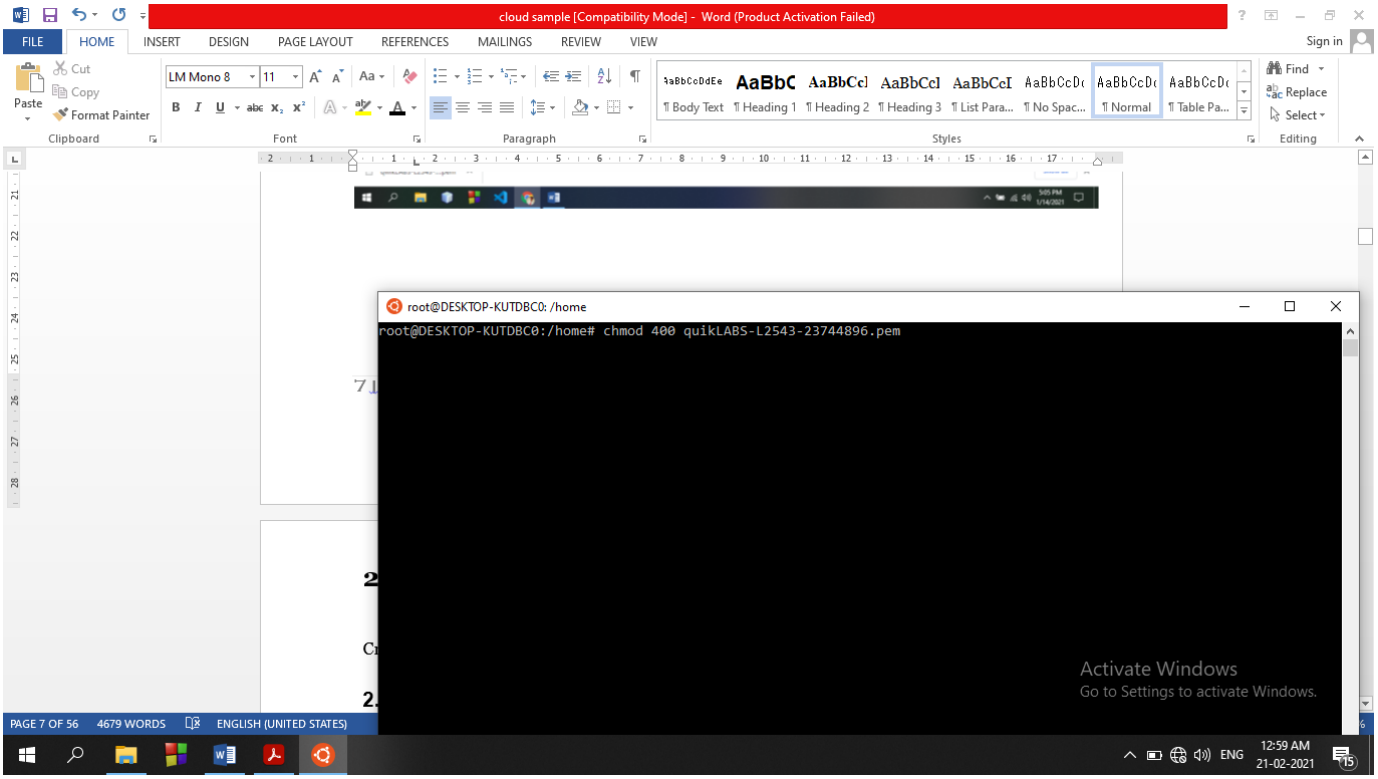
Creating security groups	Successful (sg-62d7d21b)
Authorizing inbound rules	Successful
Initiating launches	Successful
Applying tags	Successful

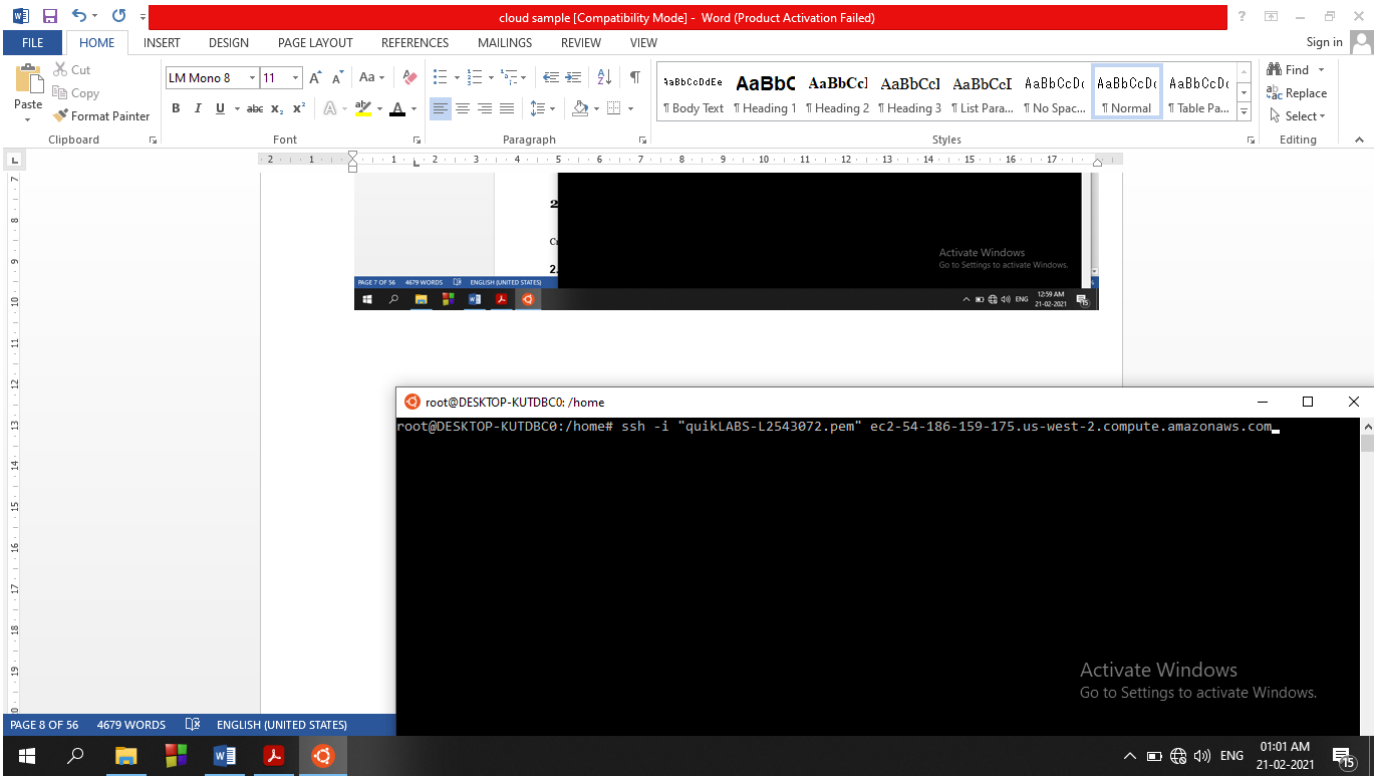
Launch initiation complete

Get notified of estimated charges

Create billing alerts to get an email notification when estimated charges on your AWS bill exceed an am




cloud sample [Compatibility Mode] - Word (Product Activation Failed)
root@DESKTOP-KUTDBC0: /home
root@DESKTOP-KUTDBC0:/home# chmod 400 quikLABS-L2543-23744896.pem
Activate Windows
Go to Settings to activate Windows.
PAGE 7 OF 56
4679 WORDS
ENGLISH (UNITED STATES)


cloud sample [Compatibility Mode] - Word (Product Activation Failed)
root@DESKTOP-KUTDBC0: /home
root@DESKTOP-KUTDBC0:/home# ssh -i "quikLABS-L2543072.pem" ec2-54-186-159-175.us-west-2.compute.amazonaws.com
Activate Windows
Go to Settings to activate Windows.
PAGE 8 OF 56
4679 WORDS
ENGLISH (UNITED STATES)

2 Creating and running virtual machines

Objective: To learn, Virtualization Basics and benefits of Virtualization in cloud.

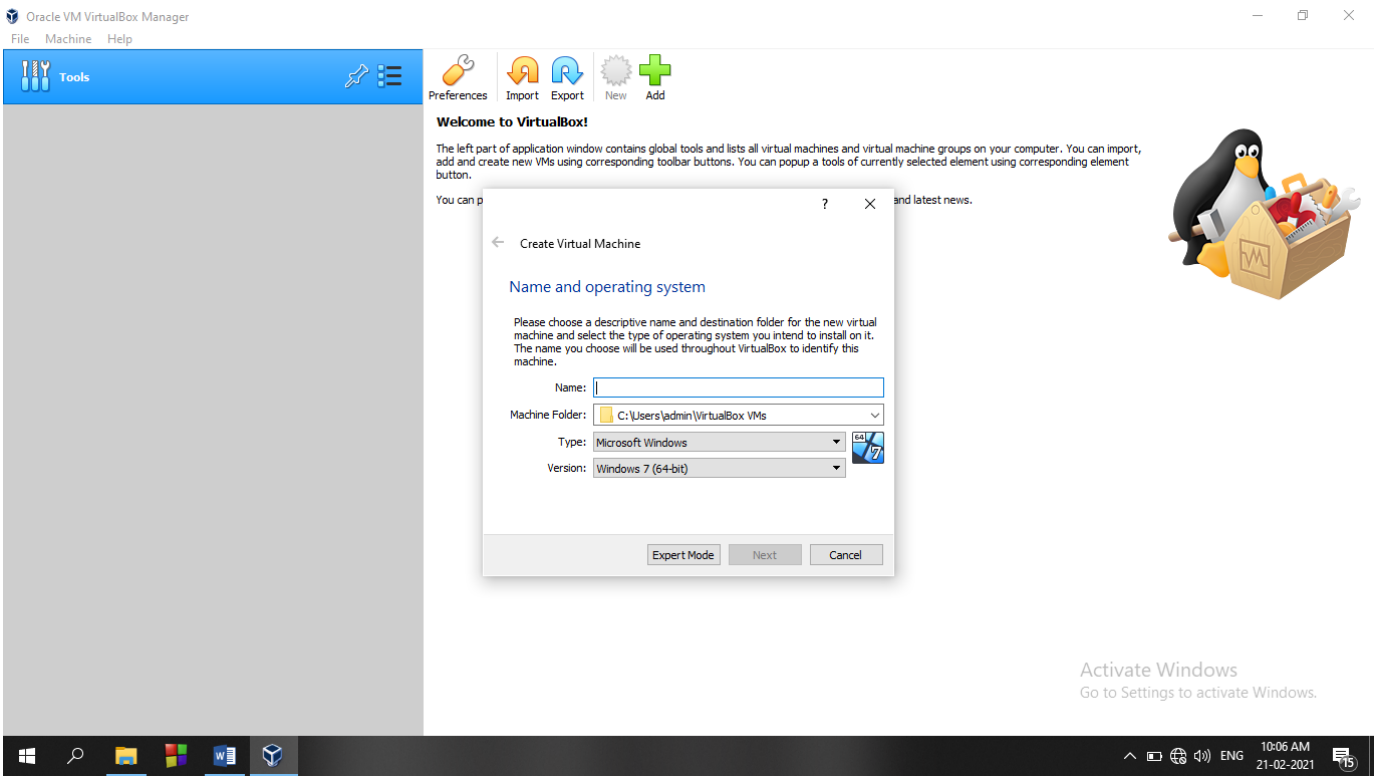
Task: Create VM on and perform migration of VM.

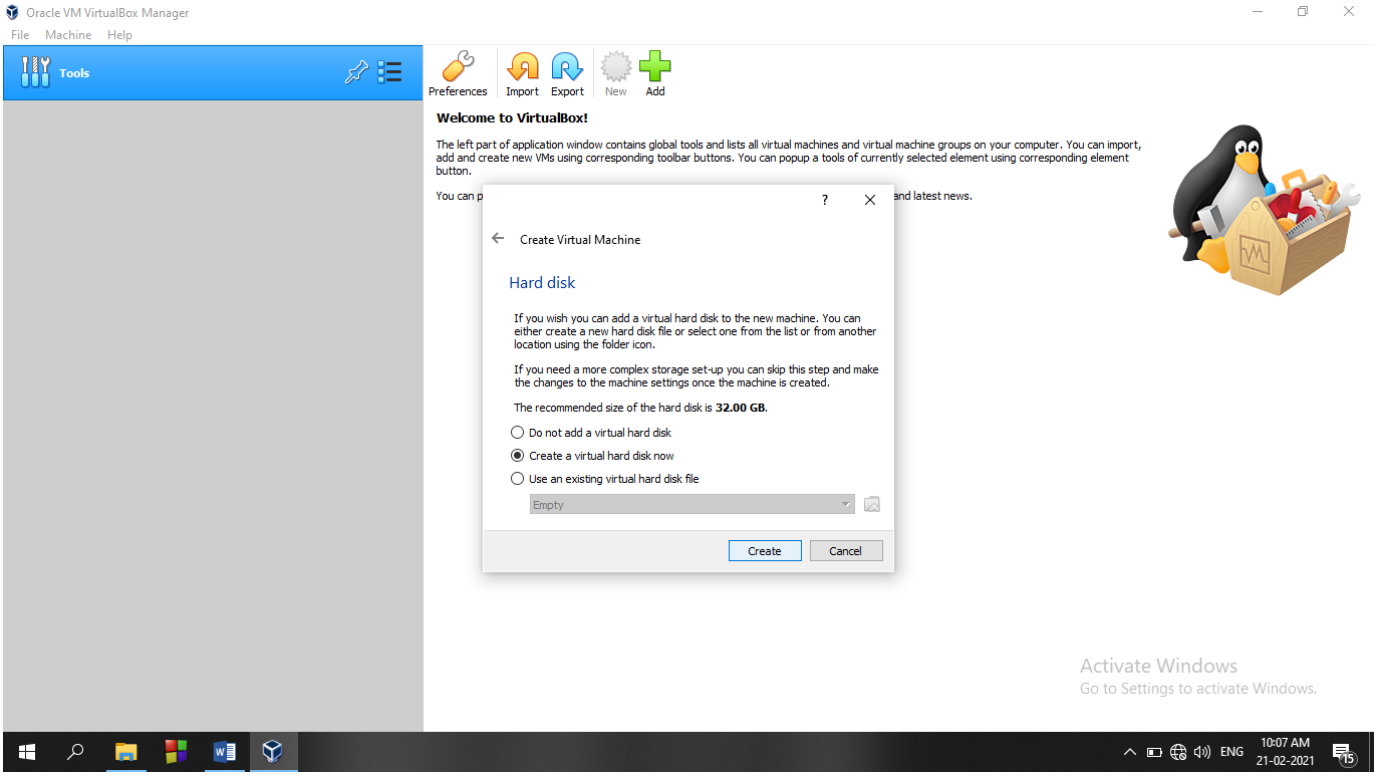
Procedure

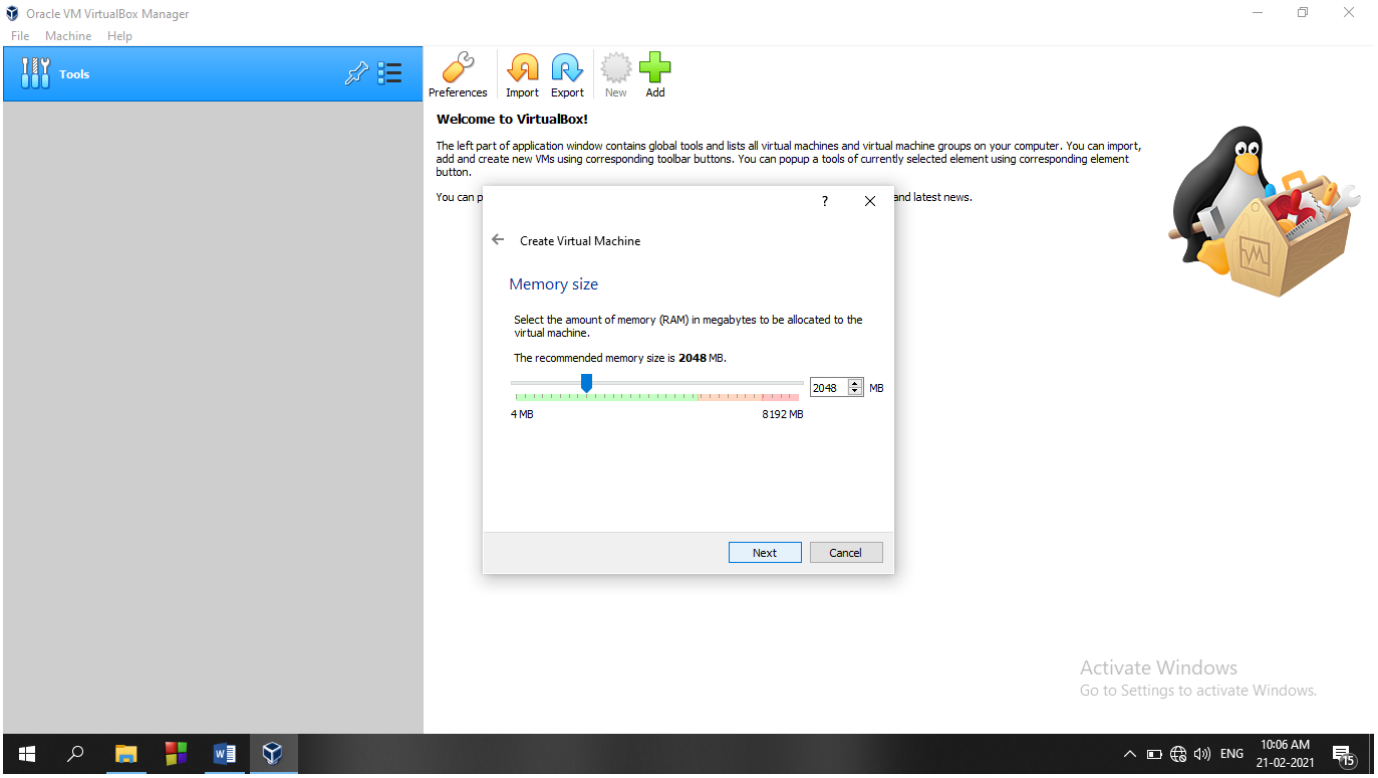
1. For creating VM we need to download . iso file of required OS and install Virtualbox in our system .

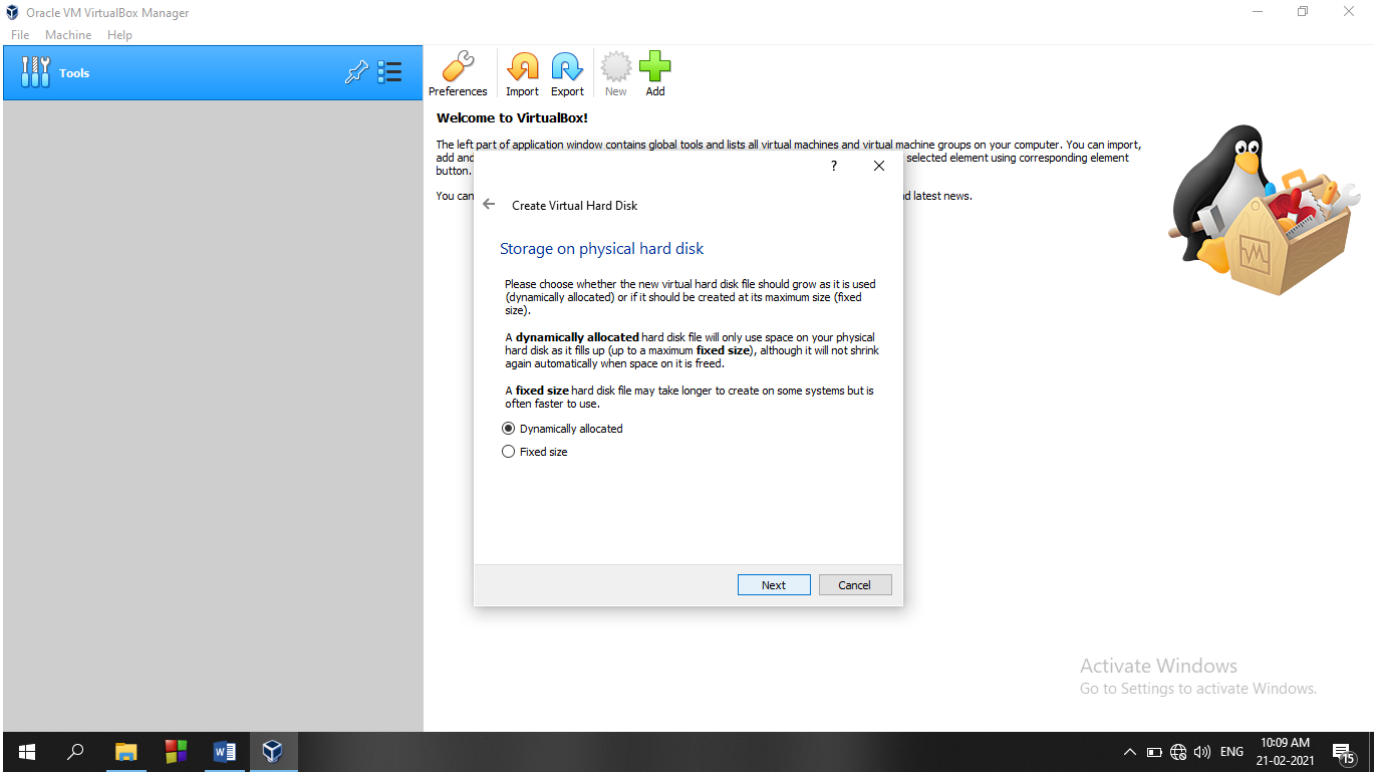
2. Then in virtualbox we need to follow the multiple operation that are mentioned in the demonstration section below step by step .

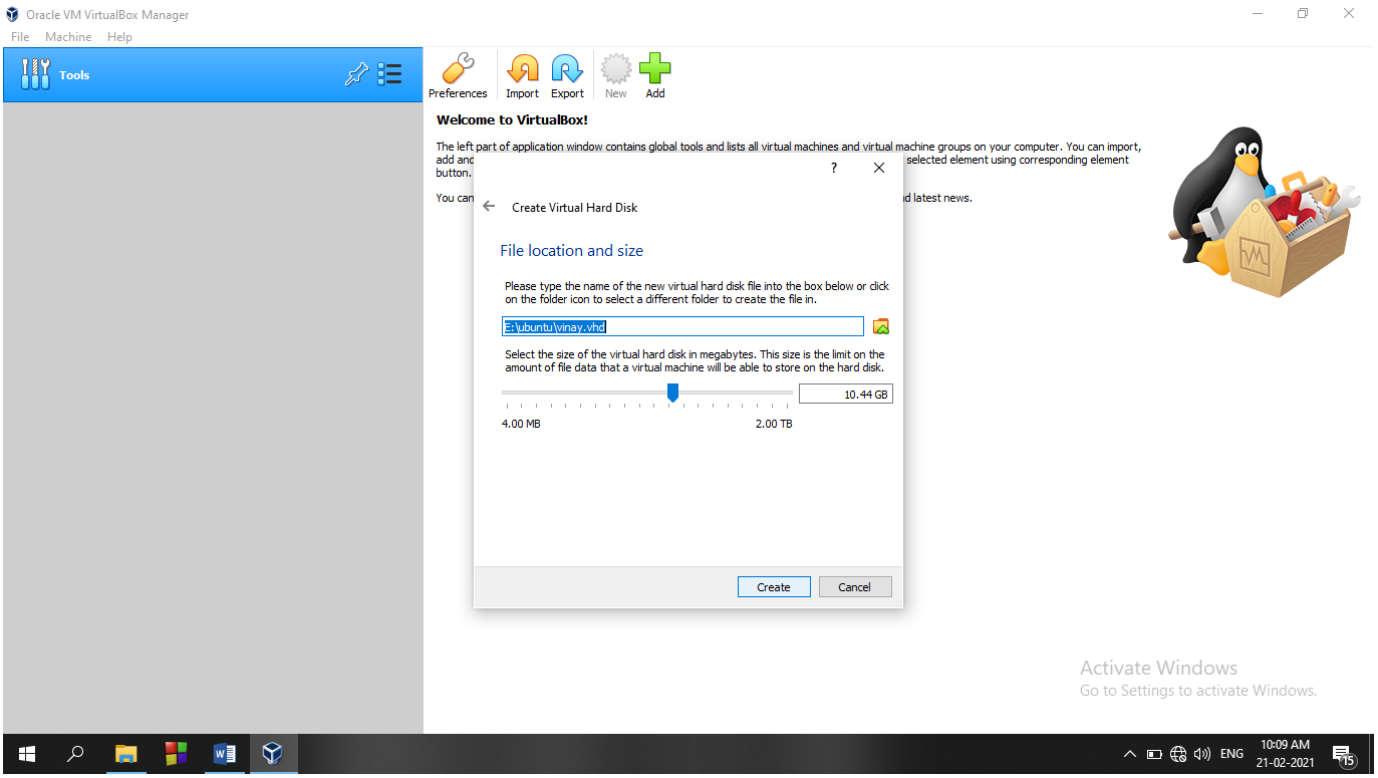
Demonstration

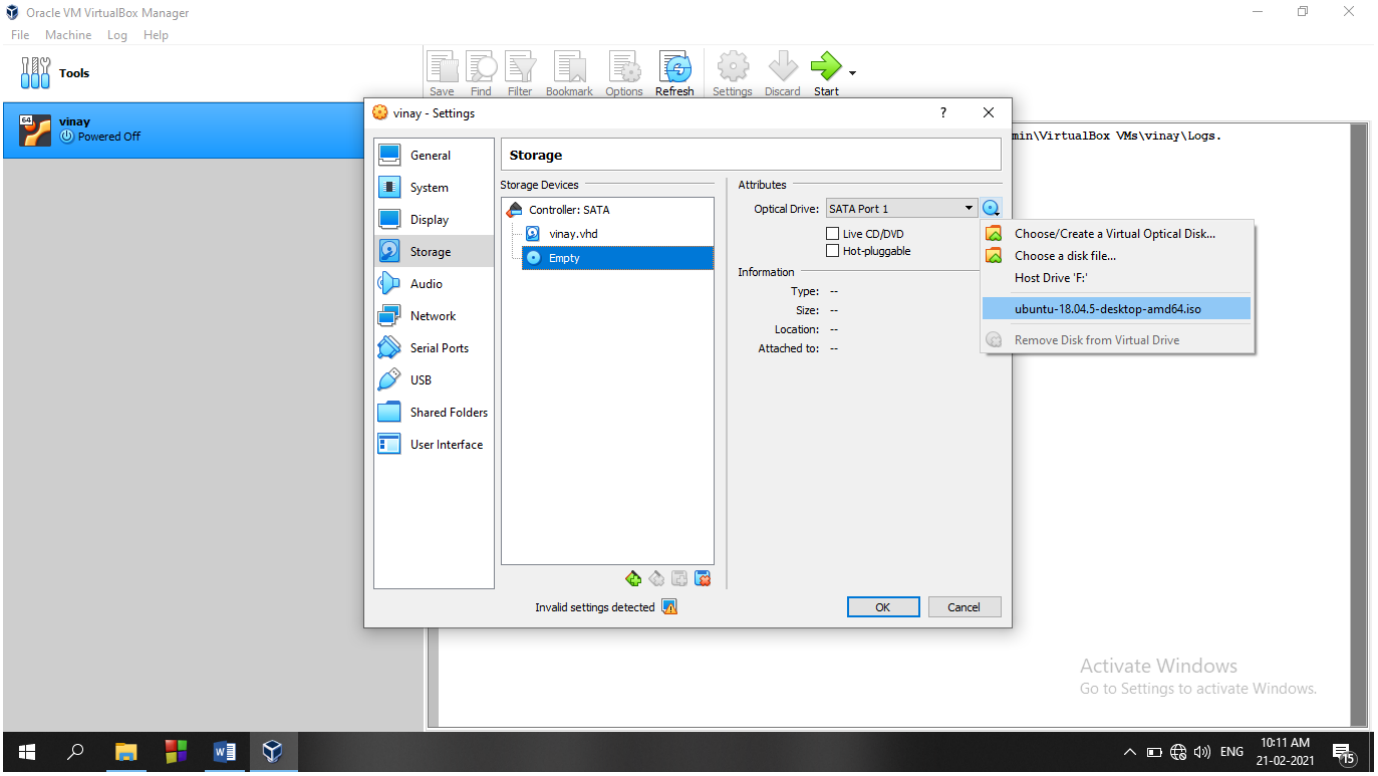


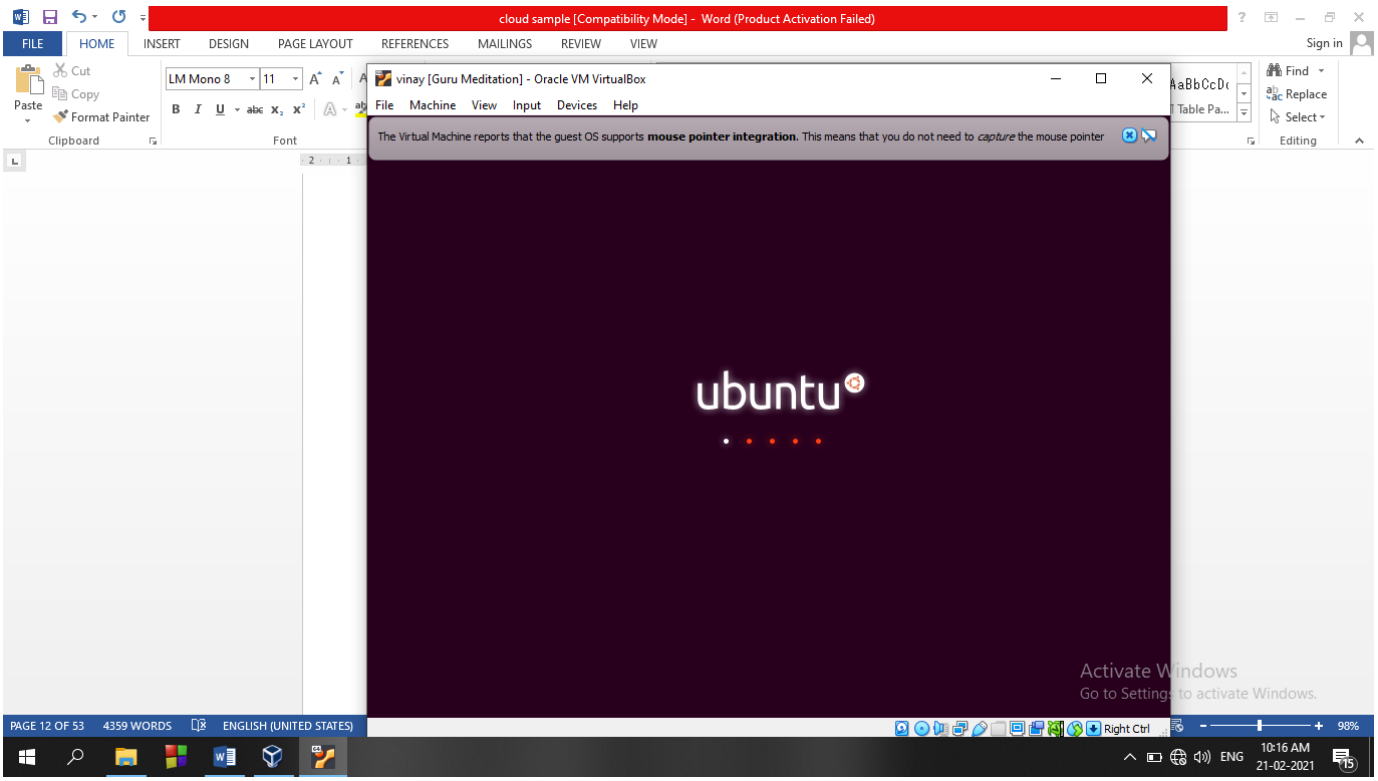

Oracle VM VirtualBox Manager
File Machine Help
Tools
Preferences Import Export New Add
Welcome to VirtualBox!
The left part of application window contains global tools and lists all virtual machines and virtual machine groups on your computer. You can import, add and create new VMs using corresponding toolbar buttons. You can popup a tools of currently selected element using corresponding element button.
Create Virtual Machine
Hard disk
If you wish you can add a virtual hard disk to the new machine. You can either create a new hard disk file or select one from the list or from another location using the folder icon.
If you need a more complex storage set-up you can skip this step and make the changes to the machine settings once the machine is created.
The recommended size of the hard disk is 32.00 GB.
Do not add a virtual hard disk
Create a virtual hard disk now
Use an existing virtual hard disk file
Empty
Create
Cancel
Activate Windows
Go to Settings to activate Windows.
ENG
10:07 AM
21-02-2021


Oracle VM VirtualBox Manager
File Machine Help
Tools
Preferences Import Export New Add
Welcome to VirtualBox!
The left part of application window contains global tools and lists all virtual machines and virtual machine groups on your computer. You can import, add and create new VMs using corresponding toolbar buttons. You can popup a tools of currently selected element using corresponding element button.
Create Virtual Machine
Memory size
Select the amount of memory (RAM) in megabytes to be allocated to the virtual machine.
The recommended memory size is 2048 MB.
2048 MB
4 MB
8192 MB
Next
Cancel
Activate Windows
Go to Settings to activate Windows.
ENG
10:06 AM
21-02-2021


Oracle VM VirtualBox Manager
File Machine Help
Tools
Preferences Import Export New Add
Welcome to VirtualBox!
Create Virtual Hard Disk
Storage on physical hard disk
Please choose whether the new virtual hard disk file should grow as it is used (dynamically allocated) or if it should be created at its maximum size (fixed size).
A dynamically allocated hard disk file will only use space on your physical hard disk as it fills up (up to a maximum fixed size), although it will not shrink again automatically when space on it is freed.
A fixed size hard disk file may take longer to create on some systems but is often faster to use.
Dynamically allocated
Fixed size
Next
Cancel
Activate Windows
Go to Settings to activate Windows.
ENG
10:09 AM
21-02-2021


Oracle VM VirtualBox Manager
File Machine Help
Tools
Preferences Import Export New Add
Welcome to VirtualBox!
Create Virtual Hard Disk
File location and size
Please type the name of the new virtual hard disk file into the box below or click on the folder icon to select a different folder to create the file in.
E:\ubuntu\vinay.vhd
Select the size of the virtual hard disk in megabytes. This size is the limit on the amount of file data that a virtual machine will be able to store on the hard disk.
10.44 GB
4.00 MB
2.00 TB
Create
Cancel
Activate Windows
Go to Settings to activate Windows.
ENG
10:09 AM
21-02-2021


Oracle VM VirtualBox Manager
File Machine Log Help
Tools
Save Find Filter Bookmark Options Refresh Settings Discard Start
vinay
Powered Off
vinay - Settings
General
System
Display
Storage
Audio
Network
Serial Ports
USB
Shared Folders
User Interface
Storage
Storage Devices
Controller: SATA
vinay.vhd
Empty
Attributes
Optical Drive: SATA Port 1
Live CD/DVD
Hot-pluggable
Information
Type: --
Size: --
Location: --
Attached to: --
Choose/Create a Virtual Optical Disk...
Choose a disk file...
Host Drive 'F:'
ubuntu-18.04.5-desktop-amd64.iso
Remove Disk from Virtual Drive
Invalid settings detected
OK
Cancel
Activate Windows
Go to Settings to activate Windows.
ENG
10:11 AM
21-02-2021


cloud sample [Compatibility Mode] - Word (Product Activation Failed)
FILE HOME INSERT DESIGN PAGE LAYOUT REFERENCES MAILINGS REVIEW VIEW
Sign in
Paste Cut Copy Format Painter
Clipboard
LM Mono 8 11
Font
Find Replace Select
Editing
vinay [Guru Meditation] - Oracle VM VirtualBox
File Machine View Input Devices Help
The Virtual Machine reports that the guest OS supports mouse pointer integration. This means that you do not need to capture the mouse pointer
ubuntu
Activate Windows
Go to Settings to activate Windows.
PAGE 12 OF 53
4359 WORDS
ENGLISH (UNITED STATES)
Right Ctrl
98%
ENG
10:16 AM
21-02-2021

Study and implementation of cloud Task Scheduling using cloud simulator

Install cloudsim and implement basic task scheduling algorithm.

1. RounRobin
2. Priority Scheduling

Introduction

CloudSim is a simulation toolkit that supports the modeling and simulation of the core functionality of cloud, like job/task queue, processing of events, creation of cloud entities(datacenter, datacenter brokers, etc), communication between different entities, implementation of broker policies, etc. This toolkit allows to:

- Test application services in a repeatable and controllable environment.
- Tune the system bottlenecks before deploying apps in an actual cloud.
- Experiment with different workload mix and resource performance scenarios on simulated infrastructure for developing and testing adaptive application provisioning techniques

2.2 Installation of cloudsim

Cloudsim simulation toolkit setup is easy. Before you start to setup CloudSim, following resources must be Installed/downloaded on the local system

- Java Development Kit(JDK)
- Eclipse IDE for Java developers
- CloudSim source code
- One external requirement of Cloudsim i.e. common jar package of math-related functions is to be downloaded from the Apache website
- Unzip Eclipse, Cloudsim and Common Math libraries to some common folder.

1. First of all, navigate to the folder where we have unzipped the eclipse folder and open Eclipse.exe
2. Now within Eclipse window navigate the menu: File -> New -> Project, to open the new project wizard
3. 'New Project' wizard should open. There are a number of options displayed and we have to find & select the 'Java Project' option, once done click 'Next'
4. Now a detailed new project window will open, here we will provide the project name and

the path of CloudSim project source code, which will be done as follows:

Project Name: CloudSim.

5. Unselect the 'Use default location' option and then click on 'Browse' to open the path where we have unzipped the Cloudsim project and finally click Next to set project settings.

6. Once done finally, click 'Next' to go to the next step i.e. setting up of project settings

7. Now open 'Libraries' tab and in the list then simply click on 'Add External Jar'

8. Once we have clicked on 'Add External JAR's' Open the path where we have un-zipped the commons-math binaries and select 'Commons-math3-3.x.jar' and click on open.

9. Ensure external jar that we opened in the previous step is displayed in the list and then click on 'Finish' (our system may take 2-3 minutes to configure the project)

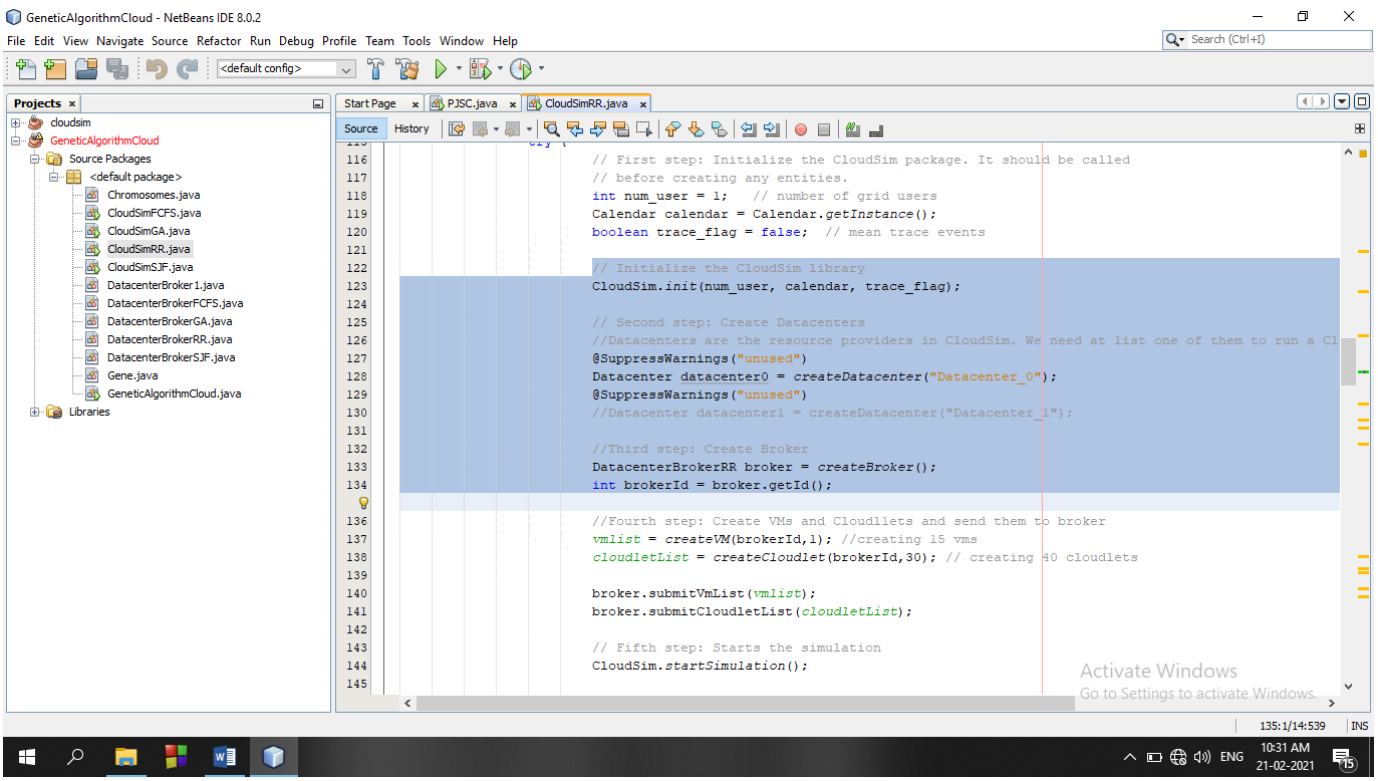
Demonstration

Once we have completed our installation/setup and understand the basic working of the cloudsim, the next step is to implement our own custom scenario. Any simulation will go through the following steps:

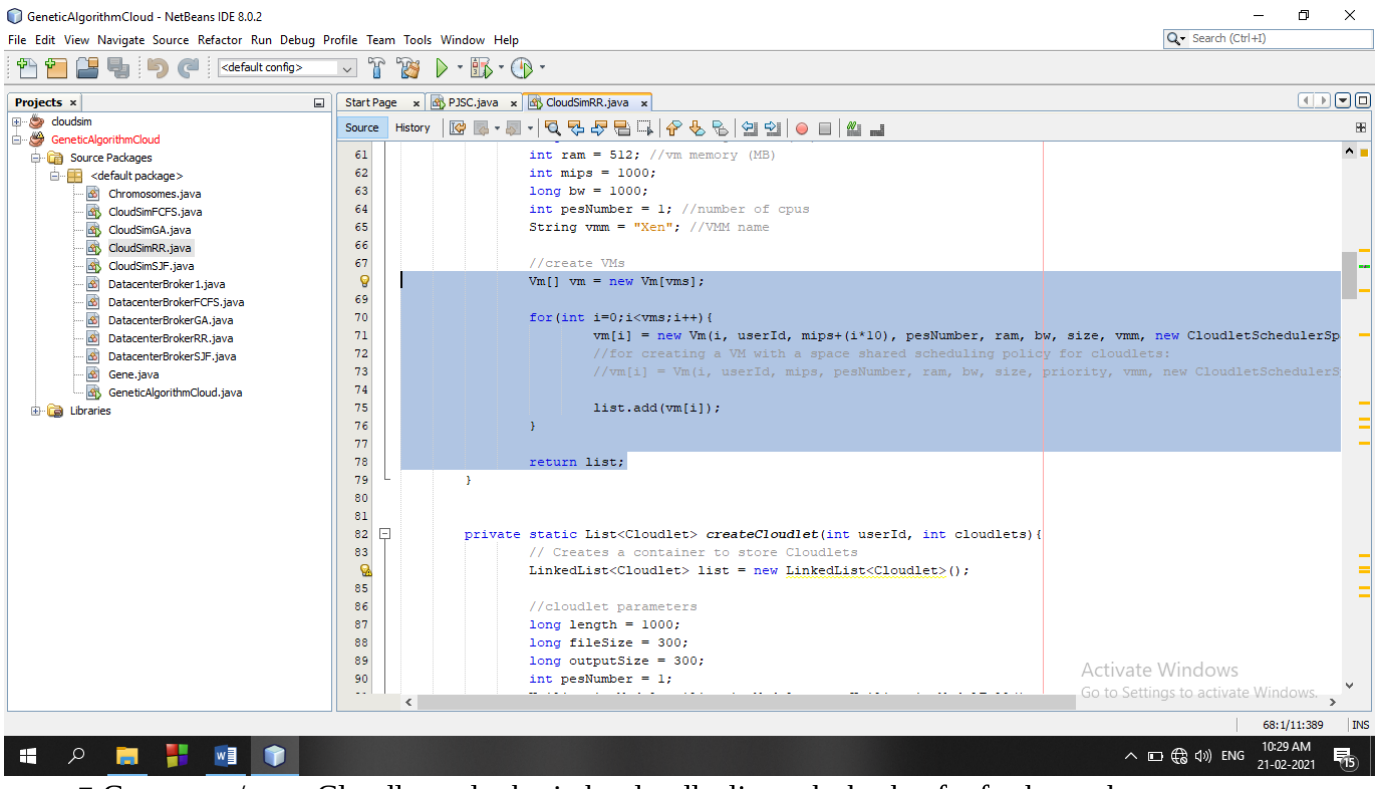
3. Initialize the CloudSim with the current clock time and this will also initialize the core Cloud Information Service entity.

4. Create Datacenter(s) as Datacenters are the resource providers in CloudSim. We need at list one of them to run a CloudSim simulation.

5. Create Broker to simulate the user workload scheduling as well as virtual machine allocation and placements(Differ according to scheduling algorithm).



6. Create one/more virtual machine and submit to the broker for further submitting it to the respective DataCenters for its placement and execution management during the simulation run.



7. Create one/more Cloudlet and submit the cloudlet list to the broker for further task scheduling on the active virtual machines for its processing during the simulation run.

8. Starts the simulation, this will initiate all the entities and components created above and put them into execution for supporting various simulation operations. Stop the simulation,

concludes simulation and flush all the entities components before the exit of a simulation run.

9. Print results when the simulation is over, where we will be able to display which cloudlet executed on which virtual machine along with how much time it spent in execution, its start time as well as its finish time.

ROUNDROBIN SCHEDULING

A round robin is an arrangement of choosing all elements in a group equally in some rational order, usually from the top to the bottom of a list and then starting again at the top of the list and so on. A simple way to think of round robin is that it is about "taking turns." Used as an adjective, round robin becomes "round-robin."

Code:

```
import java.text.DecimalFormat;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.LinkedList;
import java.util.List;

//import java.util.Random;

import org.cloudbus.cloudsim.Cloudlet;
import org.cloudbus.cloudsim.CloudletSchedulerSpaceShared;

import org.cloudbus.cloudsim.Datacenter;
//import org.cloudbus.cloudsim.DatacenterBroker;
import org.cloudbus.cloudsim.DatacenterCharacteristics;
import org.cloudbus.cloudsim.Host;
import org.cloudbus.cloudsim.Log;
import org.cloudbus.cloudsim.Pe;
import org.cloudbus.cloudsim.Storage;
import org.cloudbus.cloudsim.UtilizationModel;
import org.cloudbus.cloudsim.UtilizationModelFull;
import org.cloudbus.cloudsim.Vm;
import org.cloudbus.cloudsim.VmAllocationPolicySimple;
//import org.cloudbus.cloudsim.VmSchedulerSpaceShared;
import org.cloudbus.cloudsim.VmSchedulerTimeShared;
import org.cloudbus.cloudsim.core.CloudSim;
import org.cloudbus.cloudsim.provisioners.BwProvisionerSimple;
import org.cloudbus.cloudsim.provisioners.PeProvisionerSimple;
import org.cloudbus.cloudsim.provisioners.RamProvisionerSimple;

/**
 * An example showing how to create
 * scalable simulations.
 */
public class CloudSimRR {
```

```
    /** The cloudlet list. */
    private static List<Cloudlet> cloudletList;

    /** The vmlist. */
    private static List<Vm> vmlist;
        private static Datacenter[] datacenter;
   private static double[][] commMatrix;
   private static double[][] execMatrix;

    private static List<Vm> createVM(int userId, int vms) {

      //Creates a container to store VMs. This list is passed to the broker later
      LinkedList<Vm> list = new LinkedList<Vm>();

      //VM Parameters
      long size = 10000; //image size (MB)
      int ram = 512; //vm memory (MB)
      int mips = 1000;
      long bw = 1000;
      int pesNumber = 1; //number of cpus
      String vmm = "Xen"; //VMM name

      //create VMs
      Vm[] vm = new Vm[vms];

      for(int i=0;i<vms;i++){
              vm[i] = new Vm(i, userId, mips+(i*10), pesNumber, ram, bw, size, vmm, new
CloudletSchedulerSpaceShared());
              //for creating a VM with a space shared scheduling policy for cloudlets:
              //vm[i] = Vm(i, userId, mips, pesNumber, ram, bw, size, priority, vmm, new
CloudletSchedulerSpaceShared());

              list.add(vm[i]);
      }

      return list;
    }


    private static List<Cloudlet> createCloudlet(int userId, int cloudlets){
      // Creates a container to store Cloudlets
      LinkedList<Cloudlet> list = new LinkedList<Cloudlet>();

      //cloudlet parameters
      long length = 1000;
      long fileSize = 300;
      long outputSize = 300;
      int pesNumber = 1;
```

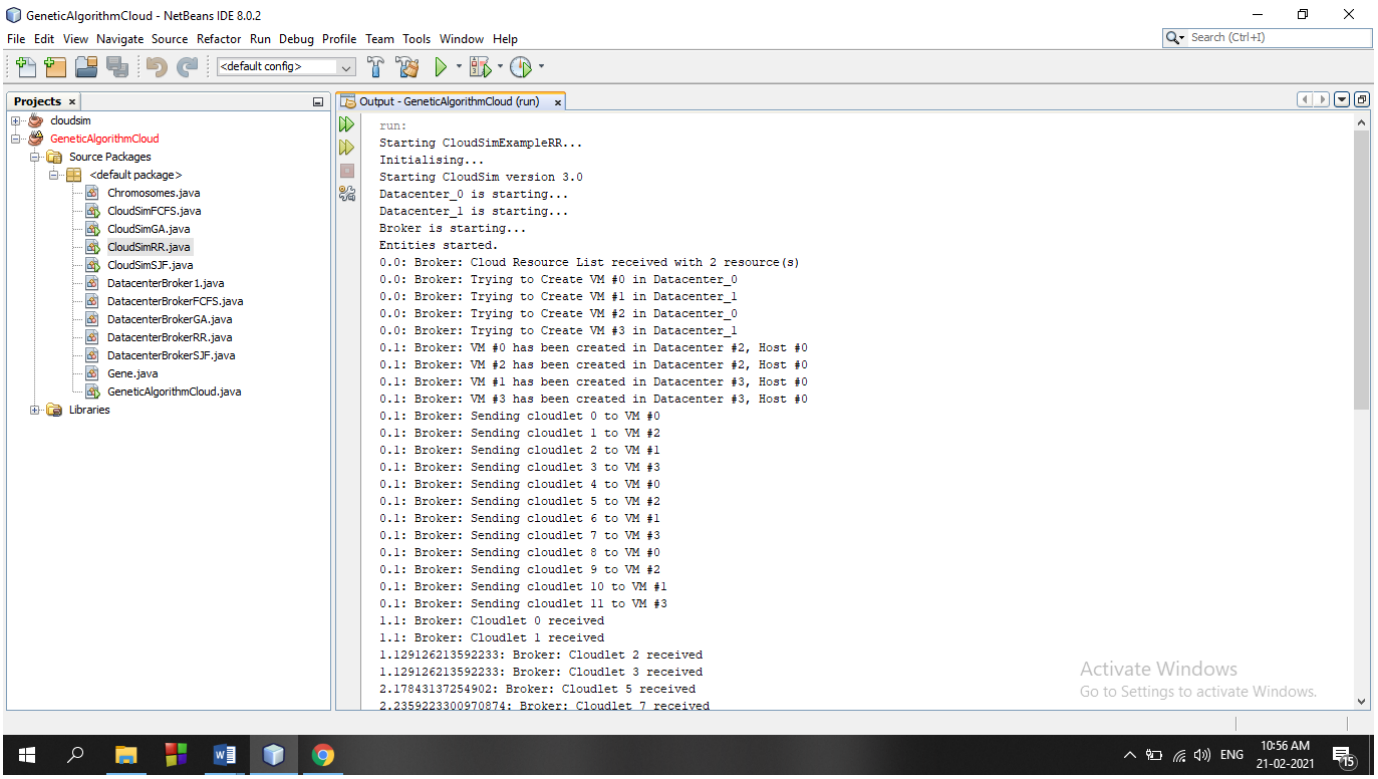
```
        UtilizationModel utilizationModel = new UtilizationModelFull();

        Cloudlet[] cloudlet = new Cloudlet[cloudlets];

        for(int i=0;i<cloudlets;i++){
        //          Random r=new Random();
                    cloudlet[i] = new Cloudlet(i, (length + 2*i*10), pesNumber, fileSize,
outputSize, utilizationModel, utilizationModel, utilizationModel);
                    // setting the owner of these Cloudlets
                    cloudlet[i].setUserId(userId);
                    list.add(cloudlet[i]);
        }
        return list;
    }
    public static void main(String[] args) {
        Log.printLine("Starting CloudSimExampleRR...");

        try {
                    // First step: Initialize the CloudSim package. It should be called
                    // before creating any entities.
                    int num_user = 1;   // number of grid users
                    Calendar calendar = Calendar.getInstance();
                    boolean trace_flag = false;  // mean trace events
                    // Initialize the CloudSim library
                    CloudSim.init(num_user, calendar, trace_flag);
                    // Second step: Create Datacenters
        //Datacenters are the resource providers in CloudSim. We need at list one of them to
run a CloudSim simulation
                    @SuppressWarnings("unused")
                    Datacenter datacenter0 = createDatacenter("Datacenter_0");
                    @SuppressWarnings("unused")
                    //Datacenter datacenter1 = createDatacenter("Datacenter_1");

                    //Third step: Create Broker
                    DatacenterBrokerRR broker = createBroker();
                    int brokerId = broker.getId();
                    //Fourth step: Create VMs and Cloudl1ets and send them to broker
                    vmlist = createVM(brokerId,1); //creating 15 vms
                    cloudletList = createCloudlet(brokerId,30); // creating 40 cloudlets
                    broker.submitVmList(vmlist);
                    broker.submitCloudletList(cloudletList);
                    // Fifth step: Starts the simulation
                    CloudSim.startSimulation();
                    // Final step: Print results when simulation is over
                    List<Cloudlet> newList = broker.getCloudletReceivedList();
                    List<Vm> newList1 = broker.getVmsCreatedList();
                    CloudSim.stopSimulation();
                    printCloudletList(newList,newList1);
```

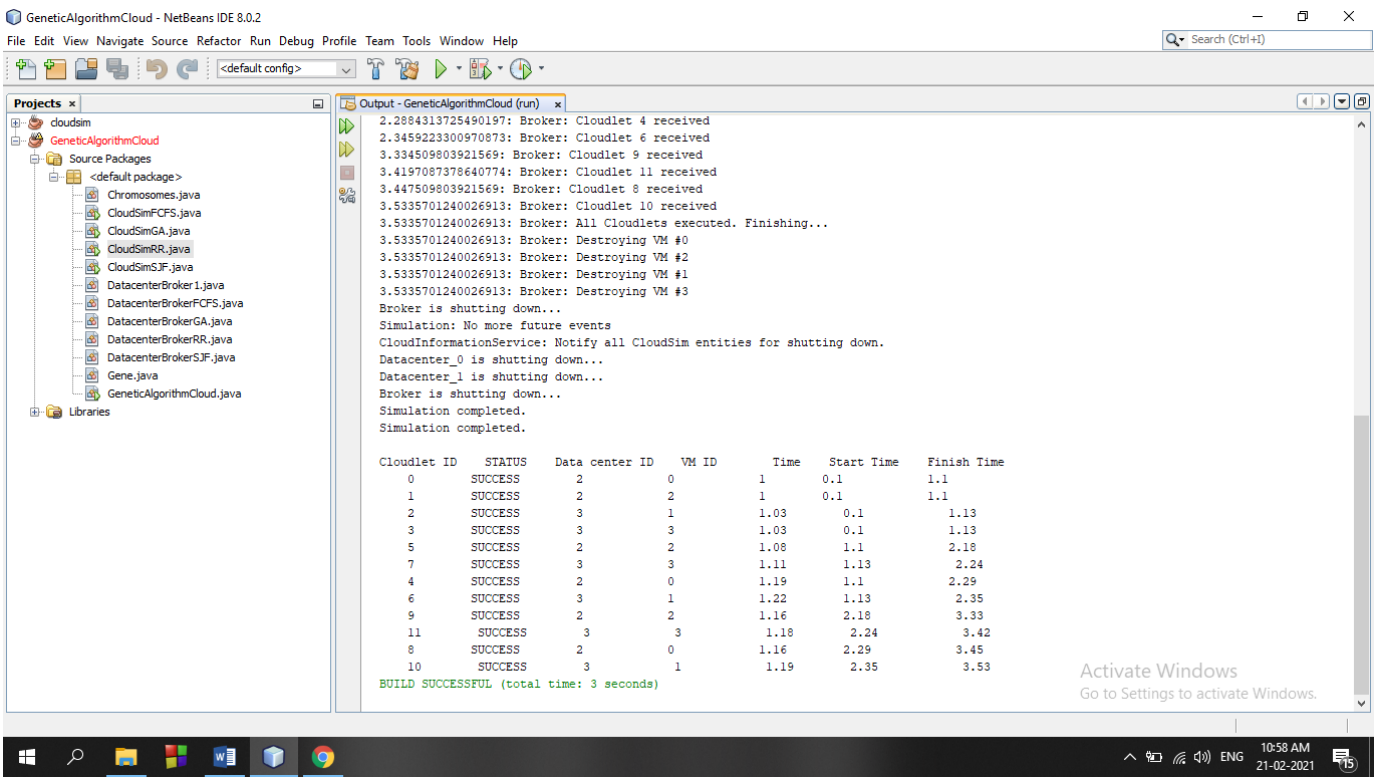
```
        }
        catch (Exception e)
        {
                e.printStackTrace();
                Log.printLine("The simulation has been terminated due to an unexpected
error");
        }
    }
    private static Datacenter createDatacenter(String name){
        // Here are the steps needed to create a PowerDatacenter:
        // 1. We need to create a list to store one or more
        //    Machines
        List<Host> hostList = new ArrayList<Host>();
        // 2. A Machine contains one or more PEs or CPUs/Cores. Therefore, should
        //    create a list to store these PEs before creating
        //    a Machine.
        List<Pe> peList1 = new ArrayList<Pe>();
        int mips = 10000;
        // 3. Create PEs and add these into the list.
        //for a quad-core machine, a list of 4 PEs is required:
        peList1.add(new Pe(0, new PeProvisionerSimple(mips + 500))); // need to store Pe id
and MIPS Rating
        peList1.add(new Pe(1, new PeProvisionerSimple(mips + 1000)));
        peList1.add(new Pe(2, new PeProvisionerSimple(mips + 1500)));
        peList1.add(new Pe(3, new PeProvisionerSimple(mips + 700)));
        //Another list, for a dual-core machine
        List<Pe> peList2 = new ArrayList<Pe>();
        peList2.add(new Pe(0, new PeProvisionerSimple(mips + 700)));
        peList2.add(new Pe(1, new PeProvisionerSimple(mips + 900)));
        //4. Create Hosts with its id and list of PEs and add them to the list of machines
        int hostId=0;
        int ram = 1002048; //host memory (MB)
        long storage = 1000000; //host storage
        int bw = 10000;
        hostList.add(
                new Host(
                        hostId,
                        new RamProvisionerSimple(ram),
                        new BwProvisionerSimple(bw),
                        storage,
                        peList1,
                        new VmSchedulerTimeShared(peList1)
                )
        ); // This is our first machine
        hostId++;
        hostList.add(
                new Host(
                        hostId,
                        new RamProvisionerSimple(ram),
```

```
                        new BwProvisionerSimple(bw),
                        storage,
                        peList2,
                        new VmSchedulerTimeShared(peList2)
                    )
        ); // Second machine
        String arch = "x86";      // system architecture
        String os = "Linux";          // operating system
        String vmm = "Xen";
        double time_zone = 10.0;         // time zone this resource located
        double cost = 3.0;              // the cost of using processing in this resource
        double costPerMem = 0.05;               // the cost of using memory in this resource
        double costPerStorage = 0.1;  // the cost of using storage in this resource
        double costPerBw = 0.1;                         // the cost of using bw in this resource
        LinkedList<Storage> storageList = new LinkedList<Storage>();     //we are not adding
SAN devices by now

        DatacenterCharacteristics characteristics = new DatacenterCharacteristics(
                arch, os, vmm, hostList, time_zone, cost, costPerMem, costPerStorage,
costPerBw);


        // 6. Finally, we need to create a PowerDatacenter object.
        Datacenter datacenter = null;
        try {
                datacenter = new Datacenter(name, characteristics, new
VmAllocationPolicySimple(hostList), storageList, 0);
        } catch (Exception e) {
                e.printStackTrace();
        }
        return datacenter;
    }
    private static DatacenterBrokerRR createBroker(){
        DatacenterBrokerRR broker = null;
        try {
                broker = new DatacenterBrokerRR("Broker");
        } catch (Exception e) {
                e.printStackTrace();
                return null;
        }
        return broker;
    }
    private static void printCloudletList(List<Cloudlet> list,List<Vm> list1) {
        int size = list.size();
        Cloudlet cloudlet;

        String indent = "    ";
        Log.printLine();
        Log.printLine("Cloudlet ID" + indent + "STATUS" + indent +
```

```
                "Data center ID" + indent + "VM ID" + indent + indent + "Time" +
indent + "Start Time" + indent + "Finish Time");

        DecimalFormat dft = new DecimalFormat("###.##");
        for (int i = 0; i < size; i++) {
                cloudlet = list.get(i);
                Log.print(indent + cloudlet.getCloudletId() + indent + indent);

                if (cloudlet.getCloudletStatus() == Cloudlet.SUCCESS){
                        Log.print("SUCCESS");

                        Log.printLine( indent + indent + cloudlet.getResourceId() +
                                        indent + indent + indent + cloudlet.getVmId() +
                                        indent + indent + indent +
dft.format(cloudlet.getActualCPUTime()) +
                                        indent + indent +
dft.format(cloudlet.getExecStartTime())+
                                        indent + indent + indent +
dft.format(cloudlet.getFinishTime()));
                }
        }
    }
}
```

Output Simulation





PRIORITY SCHEDULING

Priority Scheduling is a method of scheduling processes that is based on priority. In this algorithm, the scheduler selects the tasks to work as per the priority. The processes with higher priority should be carried out first, whereas jobs with equal priorities are carried out on a round-robin or FCFS basis

Code:

```
package org.cloudbus.cloudsim.examples;
 import java.text.DecimalFormat;
 import java.util.ArrayList;
 import java.util.Calendar;
 import java.util.LinkedList;
 import java.util.List;
 import java.util.Scanner;
 import org.cloudbus.cloudsim.Cloudlet;
 import org.cloudbus.cloudsim.CloudletSchedulerTimeShared;
 import org.cloudbus.cloudsim.Datacenter;
 import org.cloudbus.cloudsim.DatacenterBroker;
 import org.cloudbus.cloudsim.DatacenterCharacteristics;
 import org.cloudbus.cloudsim.Host;
 import org.cloudbus.cloudsim.Log;
 import org.cloudbus.cloudsim.Pe;
 import org.cloudbus.cloudsim.Storage;
 import org.cloudbus.cloudsim.UtilizationModel;
 import org.cloudbus.cloudsim.UtilizationModelFull;
 import org.cloudbus.cloudsim.Vm;
 import org.cloudbus.cloudsim.VmAllocationPolicySimple;
 import org.cloudbus.cloudsim.VmSchedulerTimeShared;
 import org.cloudbus.cloudsim.core.CloudSim;
```

```
import org.cloudbus.cloudsim.provisioners.BwProvisionerSimple;
import org.cloudbus.cloudsim.provisioners.PeProvisionerSimple;
import org.cloudbus.cloudsim.provisioners.RamProvisionerSimple;

/**
 * A simple example showing how to create a datacenter with one host and run one
 * cloudlet on it.
 */
public class PJSC {

 private static List<Cloudlet> cloudletList;
 private static List<Vm> vmlist;
 static int[] ids;

 @SuppressWarnings("unused")
 public static void main(String[] args) {

  Log.printLine("Starting CloudSimExample1...");

  try {
   // First step: Initialize the CloudSim package. It should be called
   // before creating any entities.
   int num_user = 1; // number of cloud users
   Calendar calendar = Calendar.getInstance();
   boolean trace_flag = false; // mean trace events

   // Initialize the CloudSim library
   CloudSim.init(num_user, calendar, trace_flag);

   // Second step: Create Datacenters
   // Datacenters are the resource providers in CloudSim. We need at
   // list one of them to run a CloudSim simulation
   Datacenter datacenter0 = createDatacenter("Datacenter_0");
   Datacenter datacenter1 = createDatacenter("Datacenter_1");

   // Third step: Create Broker
   DatacenterBroker broker = createBroker();
   int brokerId = broker.getId();

   // Fourth step: Create one virtual machine
   vmlist = new ArrayList<Vm>();

   // VM description
   int vmid = 0;
   int mips = 1000;
   long size = 10000; // image size (MB)
   int ram = 512; // vm memory (MB)
   long bw = 1000;
   int pesNumber = 1; // number of cpus
   String vmm = "Xen"; // VMM name

   // create VM
   Vm vm = new Vm(vmid, brokerId, mips, pesNumber, ram, bw, size, vmm, new
CloudletSchedulerTimeShared());

   // add the VM to the vmList
   vmlist.add(vm);
```

```
// submit vm list to the broker
broker.submitVmList(vmlist);

// Fifth step: Create one Cloudlet
cloudletList = new ArrayList<Cloudlet>();


// Creates a container to store Cloudlets
LinkedList<Cloudlet> list = new LinkedList<Cloudlet>();

// Cloudlet properties
int id = 0;
long length = 400000;
long fileSize = 300;
long outputSize = 300;
UtilizationModel utilizationModel = new UtilizationModelFull();


Cloudlet[] cloudlet = new Cloudlet[5];
int userId=0,cloudlets=0,idShift=0;

for(int i=0;i<4;i++)
{
 cloudlet[i] = new Cloudlet(  idShift + i, length, pesNumber, fileSize, outputSize, utilizationModel,
utilizationModel, utilizationModel);
 cloudlet[i].setUserId(brokerId);
 cloudlet[i].setVmId(vmid);
}

    A a=new A();
    int i,j;
    float[][] resource=new float[3][3];
    float[] eigen1=new float[3];
    float[] pv=new float[3];
    float[] f=new float[4];

 //resource matrix
System.out.println("enter the resource matrix");
 Scanner sc =new Scanner(System.in);
 for(i=0;i<3;i++)
 {
  for(j=0;j<3;j++)
  {
   resource[i][j]=sc.nextFloat();
  }
 }
 for(j=0;j<3;j++)
 {
  for(i=0;i<3;i++)
  {
   eigen1[j]+=resource[i][j];
  }
 }
 for(j=0;j<3;j++)
  {
   System.out.println(eigen1[j]);
```

```
  }
 for(i=0;i<3;i++)
 {
  for(j=0;j<3;j++)
  {
   resource[j][i]=resource[j][i]/eigen1[i];
  }
 }
 for(i=0;i<3;i++)
 {
  for(j=0;j<3;j++)
  {
   System.out.println(resource[i][j]);

  }
 }
 for(i=0;i<3;i++)
 {
  for(j=0;j<3;j++)
  {
   pv[i]+=resource[i][j];

  }
  pv[i]=pv[i]/3;
 }


 for(j=0;j<3;j++)
  {
   System.out.println(pv[j]);
  }
  for(j=0;j<3;j++)

  {
   a.jungle();
  }
 System.out.println("Priority Order");

 for(j=0;j<4;j++)
  {
   f[j]=0;
  }
 for(j=0;j<4;j++)
 {
     for(int k=0;k<3;k++)
     {
         f[j]+=pv[k] * a.pv1[j][k];
     }
}
for(j=0;j<4;j++)
  {
   System.out.println(f[j]);
  }
int[] ids={0,1,2,3};

for(j=0;j<4;j++)
{
```

```
 for(i=j+1;i<4;i++)
 {
  if (f[j]<f[i])
  {
   float temp;
   temp=f[j];
   f[j]=f[i];
   f[i]=temp;
   int t;
   t=ids[j];
   ids[j]=ids[i];
   ids[i]=t;
  }

 }
}
System.out.println("sorted priorities");
for(i=0;i<4;i++)
{

 System.out.println(f[i]+"\t"+ids[i]);
}


 for(i=0;i<4;i++)
 {
  cloudletList.add(cloudlet[ids[i]]);
 }



 broker.submitCloudletList(cloudletList);


 for(i=0;i<4;i++)
 {
  broker.bindCloudletToVm(cloudlet[ids[i]].getCloudletId(),vm.getId());
 }


 CloudSim.startSimulation();

 CloudSim.stopSimulation();

 List<Cloudlet> newList = broker.getCloudletReceivedList();
 printCloudletList(newList);

 Log.printLine("PJSC Successfully Executed!!!");
}
  catch (Exception e)
    {
     e.printStackTrace();
     Log.printLine("Unwanted errors happen");
    }
}

private static Datacenter createDatacenter(String name) {
```

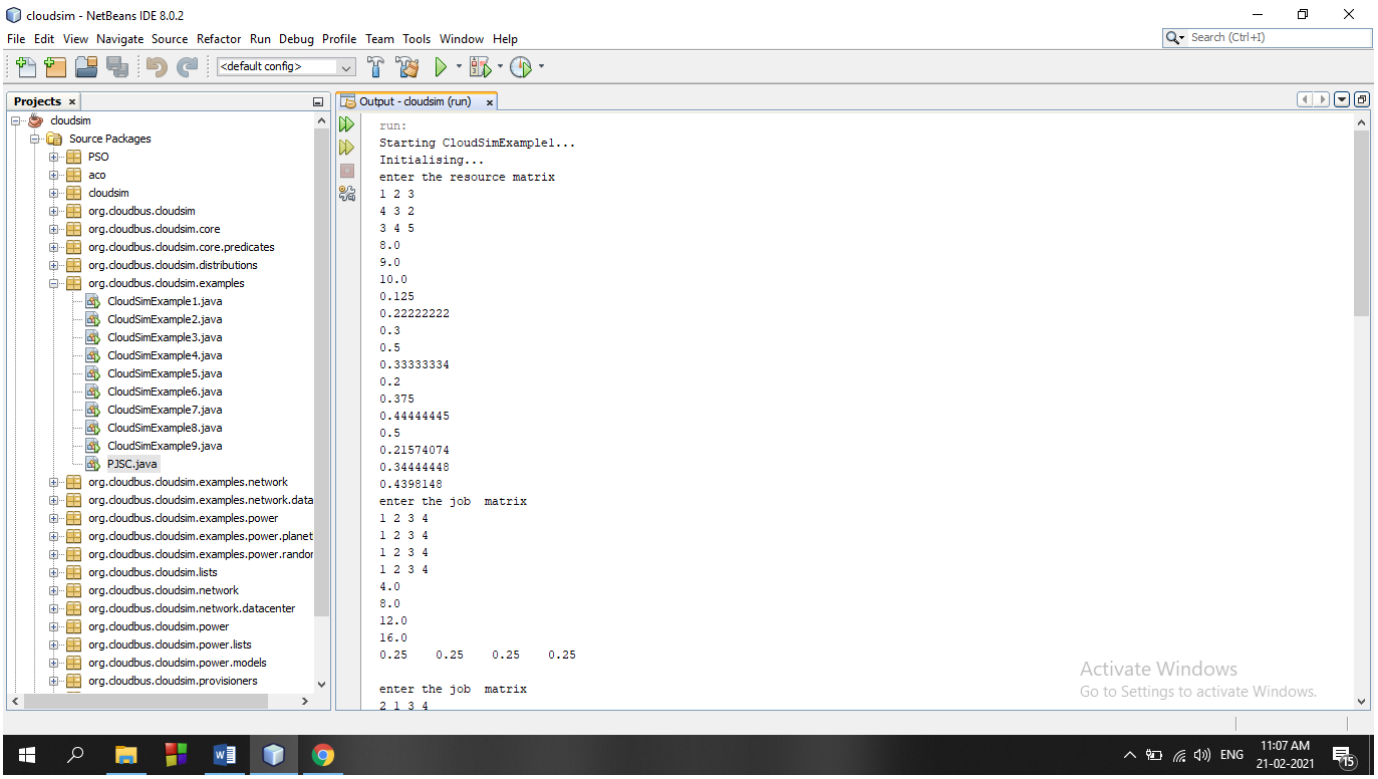
```
List<Host> hostList = new ArrayList<Host>();

List<Pe> peList = new ArrayList<Pe>();

int mips = 1000;

// 3. Create PEs and add these into a list.
peList.add(new Pe(0, new PeProvisionerSimple(mips))); // need to store Pe id and MIPS Rating

// 4. Create Host with its id and list of PEs and add them to the list
// of machines
int hostId = 0;
int ram = 2048; // host memory (MB)
long storage = 1000000; // host storage
int bw = 10000;

hostList.add(
 new Host(
  hostId,
  new RamProvisionerSimple(ram),
  new BwProvisionerSimple(bw),
  storage,
  peList,
  new VmSchedulerTimeShared(peList)
 )
); // This is our machine

// 5. Create a DatacenterCharacteristics object that stores the
// properties of a data center: architecture, OS, list of
// Machines, allocation policy: time- or space-shared, time zone
// and its price (G$/Pe time unit).
String arch = "x86"; // system architecture
String os = "Linux"; // operating system
String vmm = "Xen";
double time_zone = 10.0; // time zone this resource located
double cost = 3.0; // the cost of using processing in this resource
double costPerMem = 0.05; // the cost of using memory in this resource
double costPerStorage = 0.001; // the cost of using storage in this
       // resource
double costPerBw = 0.0; // the cost of using bw in this resource
LinkedList<Storage> storageList = new LinkedList<Storage>(); // we are not adding SAN
          // devices by now

DatacenterCharacteristics characteristics = new DatacenterCharacteristics(
  arch, os, vmm, hostList, time_zone, cost, costPerMem,
  costPerStorage, costPerBw);

// 6. Finally, we need to create a PowerDatacenter object.
Datacenter datacenter = null;
try {
 datacenter = new Datacenter(name, characteristics, new VmAllocationPolicySimple(hostList), storageList, 0);
} catch (Exception e) {
 e.printStackTrace();
}

return datacenter;
```

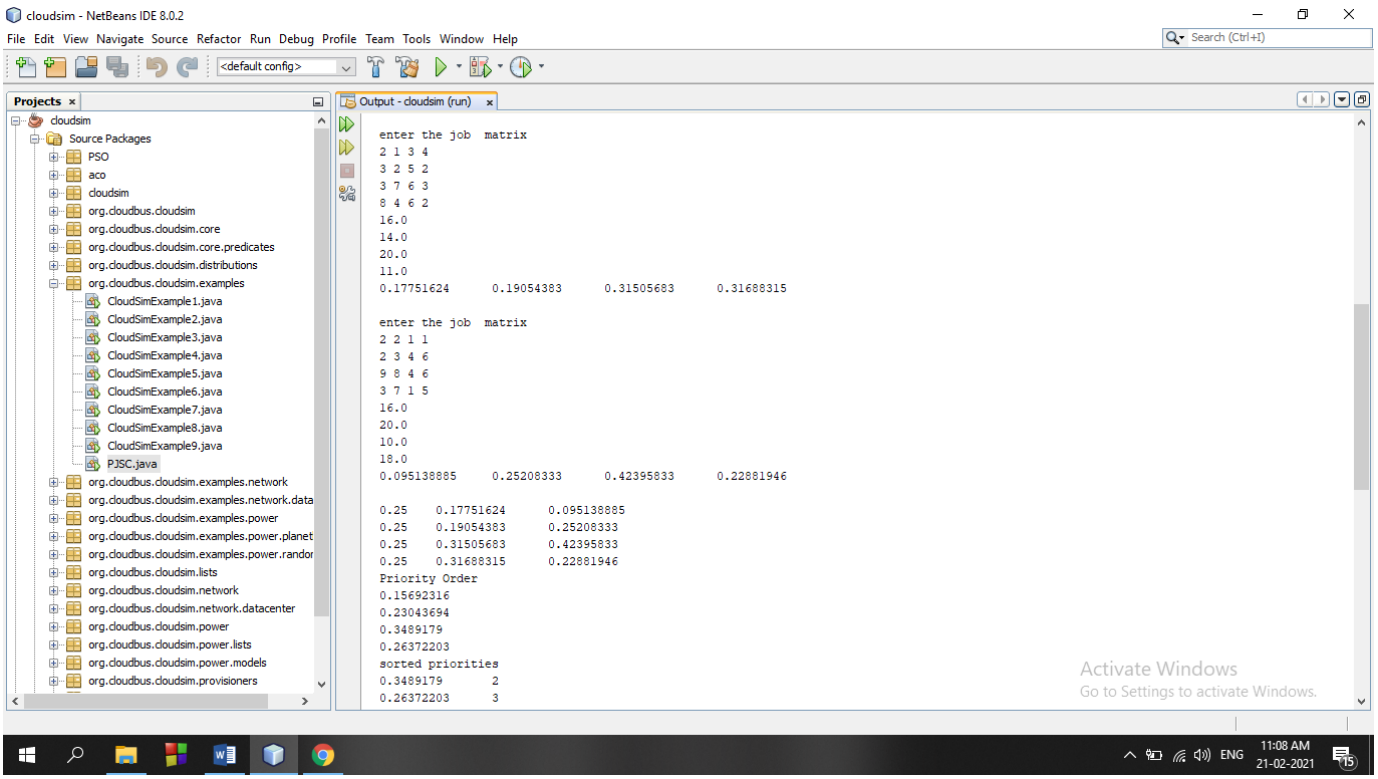
```
}

// We strongly encourage users to develop their own broker policies, to
// submit vms and cloudlets according
// to the specific rules of the simulated scenario

private static DatacenterBroker createBroker() {
 DatacenterBroker broker = null;
 try {
  broker = new DatacenterBroker("Broker");
 } catch (Exception e) {
  e.printStackTrace();
  return null;
 }
 return broker;
}

/**
 * Prints the Cloudlet objects.
 *
 * @param list list of Cloudlets
 */
private static void printCloudletList(List<Cloudlet> list) {
 int size = list.size();
 Cloudlet cloudlet;

 String indent = "    ";
 Log.printLine();
 Log.printLine("========== OUTPUT ==========");
 Log.printLine("Cloudlet ID" + indent + "STATUS" + indent
   + "Data center ID" + indent + "VM ID" + indent + "Time" + indent
   + "Start Time" + indent + "Finish Time");

 DecimalFormat dft = new DecimalFormat("###.##");
 for (int i = 0; i < size; i++) {
  cloudlet = list.get(i);
  Log.print(indent + cloudlet.getCloudletId() + indent + indent);

  if (cloudlet.getCloudletStatus() == Cloudlet.SUCCESS) {
   Log.print("SUCCESS");

   Log.printLine(indent + indent + cloudlet.getResourceId()
     + indent + indent + indent + cloudlet.getVmId()
     + indent + indent
     + dft.format(cloudlet.getActualCPUTime()) + indent
     + indent + dft.format(cloudlet.getExecStartTime())
     + indent + indent
     + dft.format(cloudlet.getFinishTime()));
  }
 }
}

}

class A
 {
 static int k=0;
```

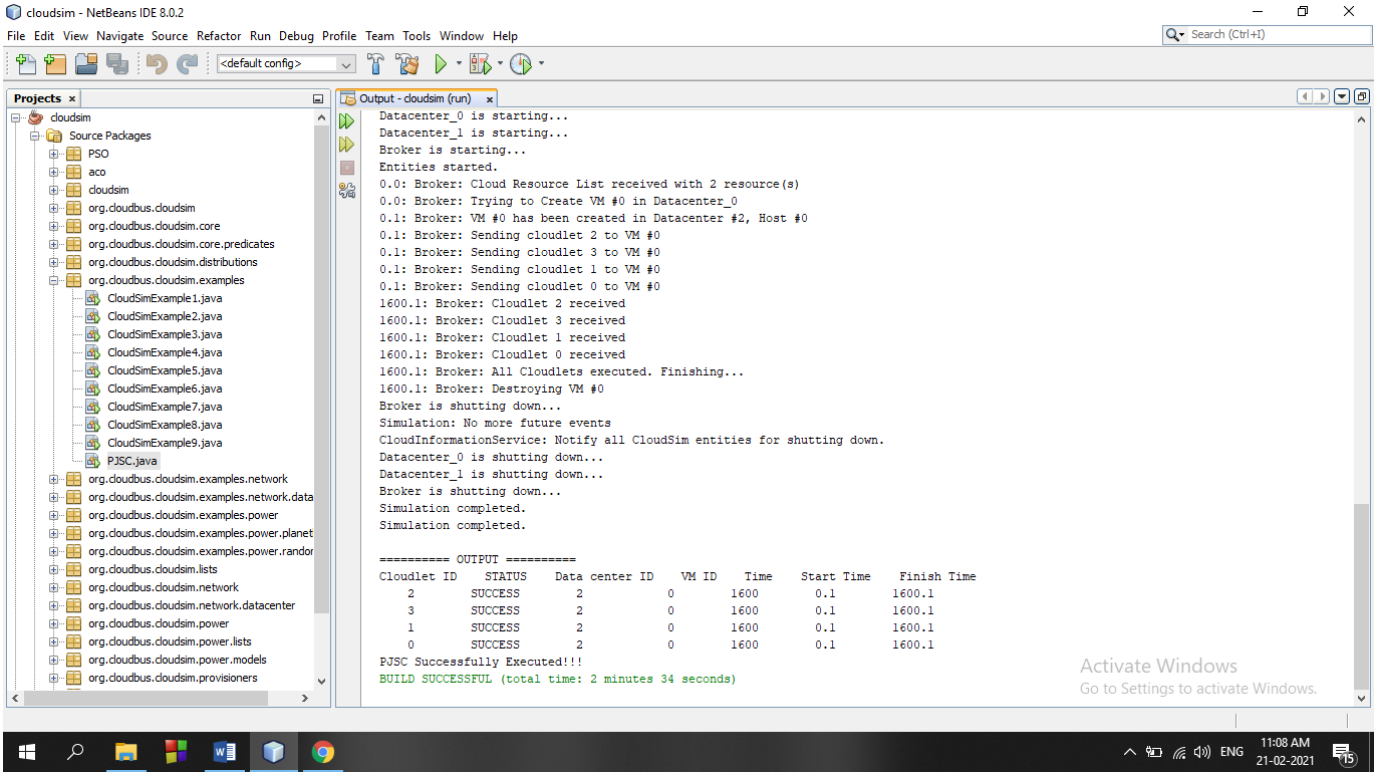
```
static float[][] pv1=new float[4][3];

void jungle()
{

 //float[][] resource=new float[3][3];
 float[] eigen1=new float[3];
 float[] eigen2=new float[4];
 float[][] job1=new float[4][4];
 //float[][] job2=new float[4][4];
 //float[][] job3=new float[4][4];
 //float[] pv=new float[3];

 int i,j;

 System.out.println("enter the job  matrix");
   Scanner sc1=new Scanner(System.in);
 for(i=0;i<4;i++)
 {
  for(j=0;j<4;j++)
  {
   job1[i][j]=sc1.nextFloat();
  }
 }

 for(j=0;j<4;j++)
 {
  for(i=0;i<4;i++)
  {
   eigen2[j]+=job1[i][j];
  }
 }
 for(j=0;j<4;j++)
  {
   System.out.println(eigen2[j]);
  }
 for(i=0;i<4;i++)
 {
  for(j=0;j<4;j++)
  {
   job1[j][i]=job1[j][i]/eigen2[i];
  }
 }
 // this is for printing eigen matrix.

 for(i=0;i<4;i++)
 {
  for(j=0;j<4;j++)
  {
   pv1[i][k]+=job1[i][j];

  }
  pv1[i][k]=pv1[i][k]/4;
 }

 for(j=0;j<4;j++)
  {
```

```
  System.out.print(pv1[j][k]+"\t");
 }
 System.out.print("\n\n");
k++;


if(k==3)
{

 for(j=0;j<4;j++)
 {
  for(i=0;i<3;i++)
  {
  System.out.print(pv1[j][i]+"\t");
  }
  System.out.print("\n");
 }
}

}
}
```

Output simulation:




cloudsim - NetBeans IDE 8.0.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
Projects
cloudsim
Source Packages
PSO
aco
cloudsim
org.cloudbus.cloudsim
org.cloudbus.cloudsim.core
org.cloudbus.cloudsim.core.predicates
org.cloudbus.cloudsim.distributions
org.cloudbus.cloudsim.examples
CloudSimExample1.java
CloudSimExample2.java
CloudSimExample3.java
CloudSimExample4.java
CloudSimExample5.java
CloudSimExample6.java
CloudSimExample7.java
CloudSimExample8.java
CloudSimExample9.java
PJSC.java
org.cloudbus.cloudsim.examples.network
org.cloudbus.cloudsim.examples.network.data
org.cloudbus.cloudsim.examples.power
org.cloudbus.cloudsim.examples.power.planet
org.cloudbus.cloudsim.examples.power.randor
org.cloudbus.cloudsim.lists
org.cloudbus.cloudsim.network
org.cloudbus.cloudsim.network.datacenter
org.cloudbus.cloudsim.power
org.cloudbus.cloudsim.power.lists
org.cloudbus.cloudsim.power.models
org.cloudbus.cloudsim.provisioners
Output - cloudsim (run)
enter the job matrix
2 1 3 4
3 2 5 2
3 7 6 3
8 4 6 2
16.0
14.0
20.0
11.0
0.17751624 0.19054383 0.31505683 0.31688315
enter the job matrix
2 2 1 1
2 3 4 6
9 8 4 6
3 7 1 5
16.0
20.0
10.0
18.0
0.095138885 0.25208333 0.42395833 0.22881946
0.25 0.17751624 0.095138885
0.25 0.19054383 0.25208333
0.25 0.31505683 0.42395833
0.25 0.31688315 0.22881946
Priority Order
0.15692316
0.23043694
0.3489179
0.26372203
sorted priorities
0.3489179 2
0.26372203 3
Activate Windows
Go to Settings to activate Windows.
ENG
11:08 AM
21-02-2021


cloudsim - NetBeans IDE 8.0.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
Projects
cloudsim
Source Packages
PSO
aco
cloudsim
org.cloudbus.cloudsim
org.cloudbus.cloudsim.core
org.cloudbus.cloudsim.core.predicates
org.cloudbus.cloudsim.distributions
org.cloudbus.cloudsim.examples
CloudSimExample1.java
CloudSimExample2.java
CloudSimExample3.java
CloudSimExample4.java
CloudSimExample5.java
CloudSimExample6.java
CloudSimExample7.java
CloudSimExample8.java
CloudSimExample9.java
PJSC.java
org.cloudbus.cloudsim.examples.network
org.cloudbus.cloudsim.examples.network.data
org.cloudbus.cloudsim.examples.power
org.cloudbus.cloudsim.examples.power.planet
org.cloudbus.cloudsim.examples.power.randor
org.cloudbus.cloudsim.lists
org.cloudbus.cloudsim.network
org.cloudbus.cloudsim.network.datacenter
org.cloudbus.cloudsim.power
org.cloudbus.cloudsim.power.lists
org.cloudbus.cloudsim.power.models
org.cloudbus.cloudsim.provisioners
Output - cloudsim (run)
Datacenter_0 is starting...
Datacenter_1 is starting...
Broker is starting...
Entities started.
0.0: Broker: Cloud Resource List received with 2 resource(s)
0.0: Broker: Trying to Create VM #0 in Datacenter_0
0.1: Broker: VM #0 has been created in Datacenter #2, Host #0
0.1: Broker: Sending cloudlet 2 to VM #0
0.1: Broker: Sending cloudlet 3 to VM #0
0.1: Broker: Sending cloudlet 1 to VM #0
0.1: Broker: Sending cloudlet 0 to VM #0
1600.1: Broker: Cloudlet 2 received
1600.1: Broker: Cloudlet 3 received
1600.1: Broker: Cloudlet 1 received
1600.1: Broker: Cloudlet 0 received
1600.1: Broker: All Cloudlets executed. Finishing...
1600.1: Broker: Destroying VM #0
Broker is shutting down...
Simulation: No more future events
CloudInformationService: Notify all CloudSim entities for shutting down.
Datacenter_0 is shutting down...
Datacenter_1 is shutting down...
Broker is shutting down...
Simulation completed.
Simulation completed.
========== OUTPUT ==========
Cloudlet ID STATUS Data center ID VM ID Time Start Time Finish Time
2 SUCCESS 2 0 1600 0.1 1600.1
3 SUCCESS 2 0 1600 0.1 1600.1
1 SUCCESS 2 0 1600 0.1 1600.1
0 SUCCESS 2 0 1600 0.1 1600.1
PJSC Successfully Executed!!!
BUILD SUCCESSFUL (total time: 2 minutes 34 seconds)
Activate Windows
Go to Settings to activate Windows.
ENG
11:08 AM
21-02-2021

Lab Assignment 2

S.No	Detailed Content	Date of Submission	Mapping
1	**Objective:** To learn HDFS commands and Design Map Reduce Algorithm. **Title:** Explore HDFS and design Map Reduce algorithm. **Task:** Design Map reduce algorithm for some basic problems **like word count problem.**	**20/02/2021**	**CO 3**
2	**Objective :To learn how to use/devlope REST API** **Title:** Designing an web application that uses REST API's . **Task**: Design any application that uses the REST interface in json format and xml format.	**20/02/2021**	**CO 2**
3	**Objective:** To learn QOS techniques in cloud like–Load Management **Title:** Study and implementation of cloud load management techniques **Task:** Install cloudsim and simulate load balancing algorithm .	**20/02/2021**	**CO 4**
4	**Objective:** To learn simulating Evolutionary algorithm in cloud environment. **Title:** Explore workflow simulator and run workflow application redesign the code with some optimization algorithm like ACO,PSO. **Task:** Install simulator and perform ACO and PSO algorithm in cloud environment	**20/02/2021**	**CO 4**

Explore HDFS and design Map Reduce algorithm

Objective: To learn HDFS commands and Design Map Reduce Algorithm.

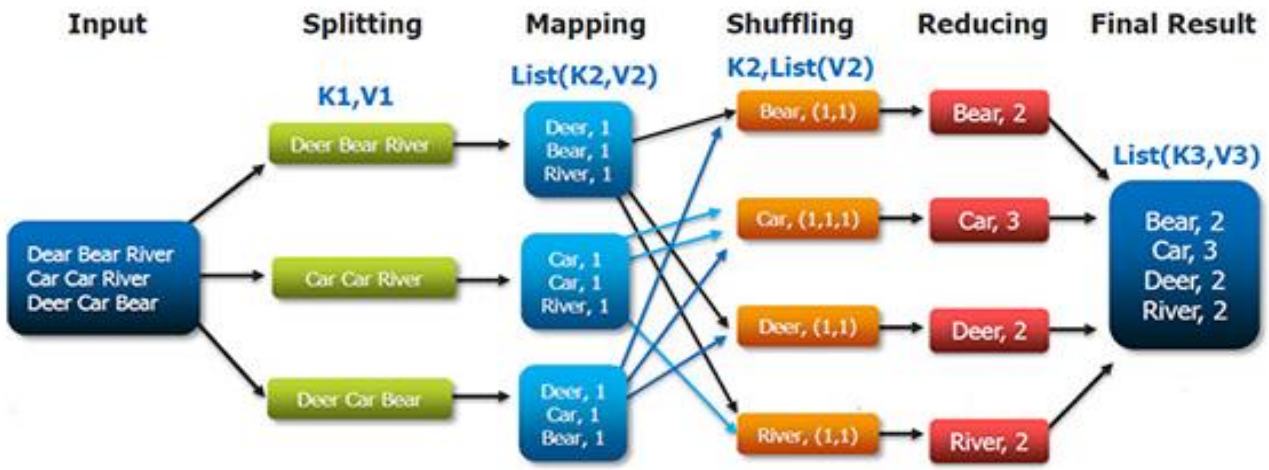
Task: Design Map reduce algorithm for some basic problems like word count problem.

Introduction

Hadoop is an open-source software framework for storing data and running applications on clusters of commodity hardware. It provides massive storage for any kind of data, enormous processing power and the ability to handle virtually limitless concurrent tasks or jobs

MapReduce is a programming model and an associated implementation for processing and generating big data sets with a parallel, distributed algorithm on a cluster. A MapReduce program is composed of a map procedure, which performs filtering and sorting, and a reduce method, which performs a summary operation.

MapReduce Algorithm Steps
It basically consists of two parts:
Map : It takes a set of data and converts it into intermediate set of data which must be of <key,value> pair form.
Reduce: It takes input tuples from map and combine those tuples into smaller set of tuples.



Code1:

WordCount.java

```
importjava.io.IOException;
importjava.util.StringTokenizer;
importorg.apache.hadoop.conf.Configuration;
importorg.apache.hadoop.fs.Path;
importorg.apache.hadoop.io.IntWritable;
importorg.apache.hadoop.io.Text;
importorg.apache.hadoop.mapreduce.Job;
importorg.apache.hadoop.mapreduce.Mapper;
importorg.apache.hadoop.mapreduce.Reducer;
importorg.apache.hadoop.mapreduce.lib.input.FileInputFormat;
importorg.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

publicclassWordCount {

  publicstaticclassTokenizerMapper
       extendsMapper<Object, Text, Text, IntWritable>{

    privatefinalstaticIntWritable one =newIntWritable(1);
    privateText word =newText();

    publicvoidmap(Object key, Text value, Context context
                  )throwsIOException, InterruptedException {
      StringTokenizer itr =newStringTokenizer(value.toString());
      while(itr.hasMoreTokens()) {
        word.set(itr.nextToken());
        context.write(word, one);
      }
    }
  }

  publicstaticclassIntSumReducer extendsReducer<Text,IntWritable,Text,IntWritable>
       {
    privateIntWritable result =newIntWritable();

    publicvoidreduce(Text key, Iterable<IntWritable> values,
                     Context context
                     )throwsIOException, InterruptedException {
      intsum = 0;
      for(IntWritable val : values) { sum
        += val.get();
      }
      result.set(sum); context.write(key,
      result);
    }
  }
```

```
  publicstaticvoidmain(String[] args)throwsException {
    Configuration conf =newConfiguration();
    Job job = Job.getInstance(conf,"wordcount");
    job.setJarByClass(WordCount.class);
    job.setMapperClass(TokenizerMapper.class);
    job.setCombinerClass(IntSumReducer.class);
    job.setReducerClass(IntSumReducer.class); job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    FileInputFormat.addInputPath(job,newPath(args[0]));
    FileOutputFormat.setOutputPath(job,newPath(args[1]));
    System.exit(job.waitForCompletion(true) ? 0 : 1);
  }
}
```

Procedure

1. Install and configure docker then check version by the command

docker --version

2. Start any server to host hadoop (we have used nginx)

docker run -d -p 80:80 --name myserver nginx

3. Start any server to host hadoop (we have used nginx)

wget "https://github.com/big-data-europe/docker-hadoop/archive/master.zip" && unzip master.zip && cd docker-hadoop-master/

4. To compose and install all required dependencies run command

docker-compose up –d

5. To print list of all docker running and then run namenode docker

docker ps

docker exec -it namenode bash

6. Create input file from which we have to perform wordcount

mkdir input
echo "Hello World is my first program" >input/f1.txt
echo "Hello Docker is first docker input program" >input/f2.txt

7. Move our input to hadoop file system

hadoop fs -mkdir -p input
hdfs dfs -put ./input/* input

8. download map reduce program and copy that to namenode

wget "https://repo1.maven.org/maven2/org/apache/hadoop/hadoop-mapreduce-examples/2.7.1/hadoop-mapreduce-examples-2.7.1-sources.jar"

docker cp hadoop-mapreduce-examples-2.7.1-sources.jar namenode:hadoop-mapreduce-examples-2.7.1-sources.jar

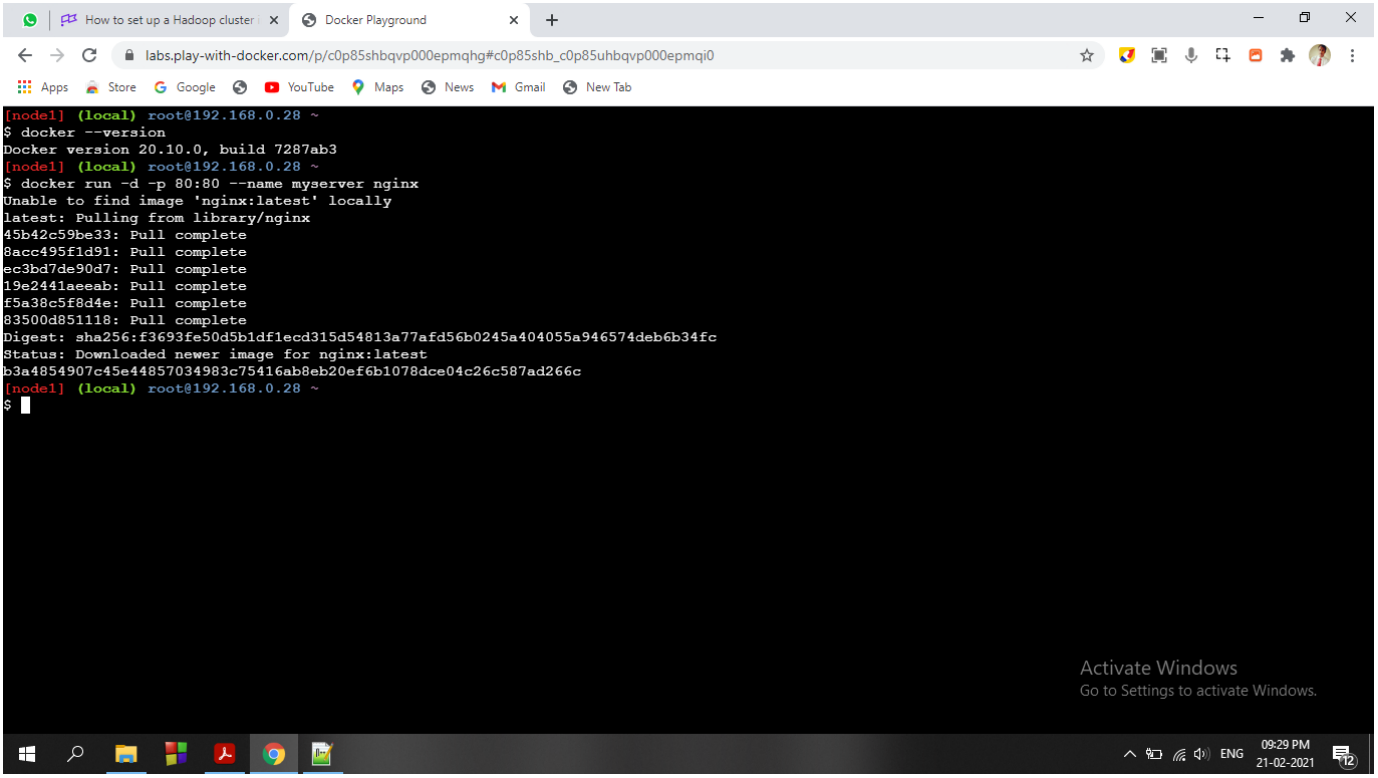
9. go to namenode and execute hadoop program

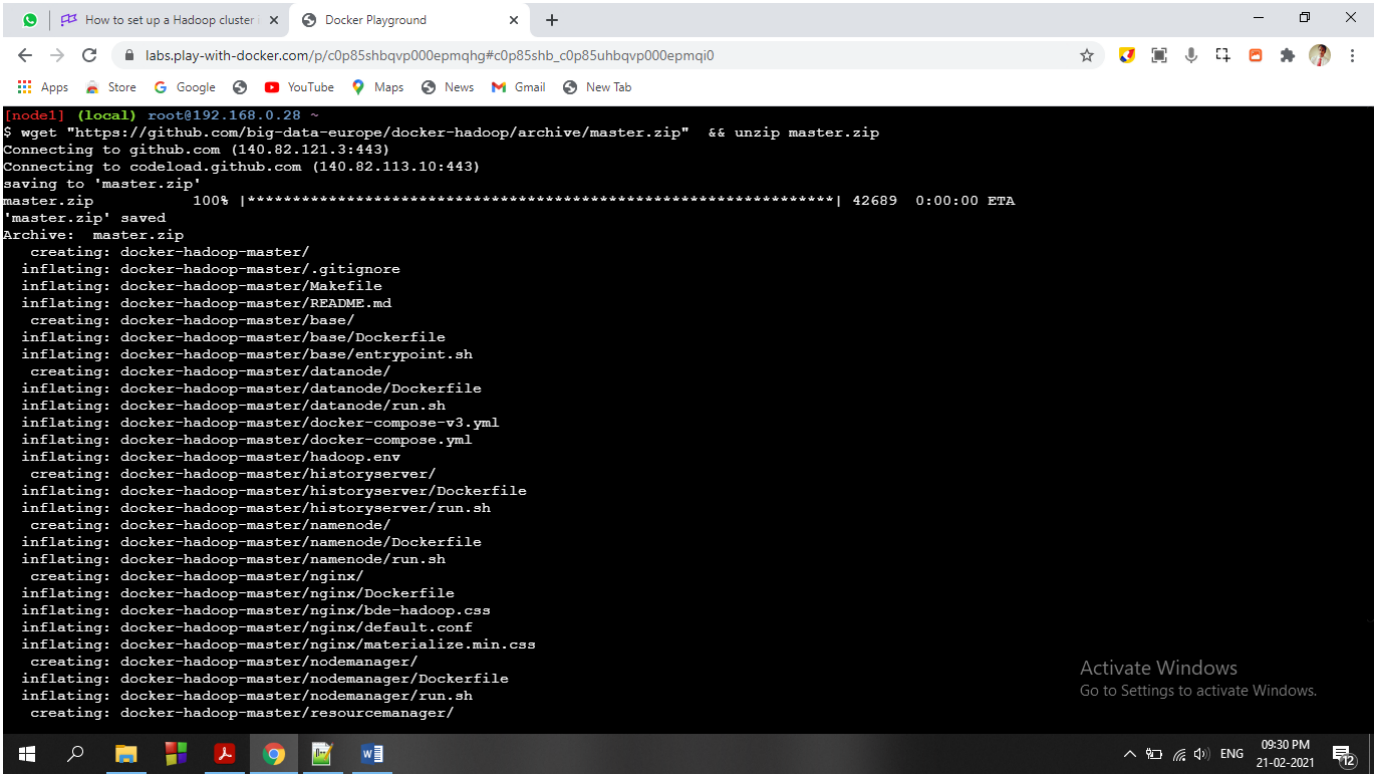
hadoop jar hadoop-mapreduce-examples-2.7.1-sources.jar org.apache.hadoop.examples.WordCount input output

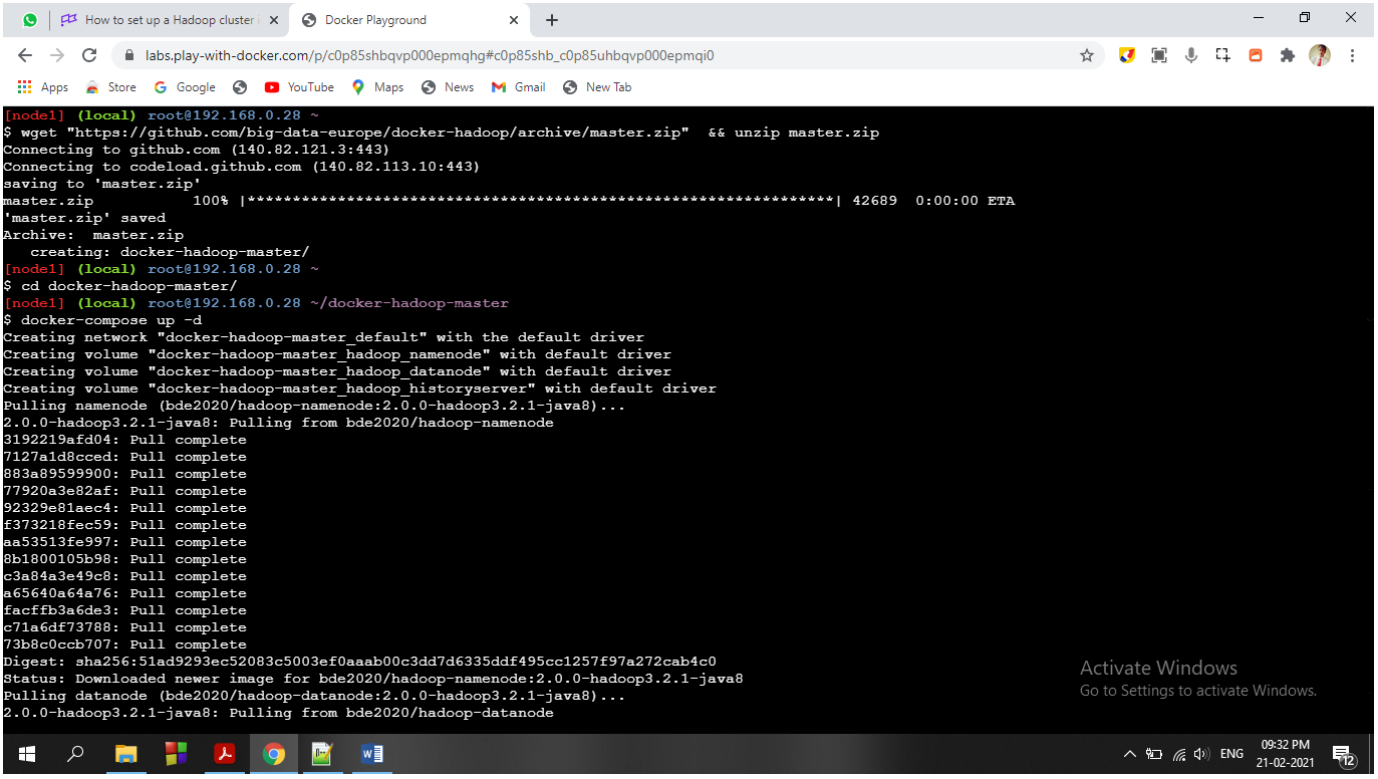
10. To get ouptput use the following command -

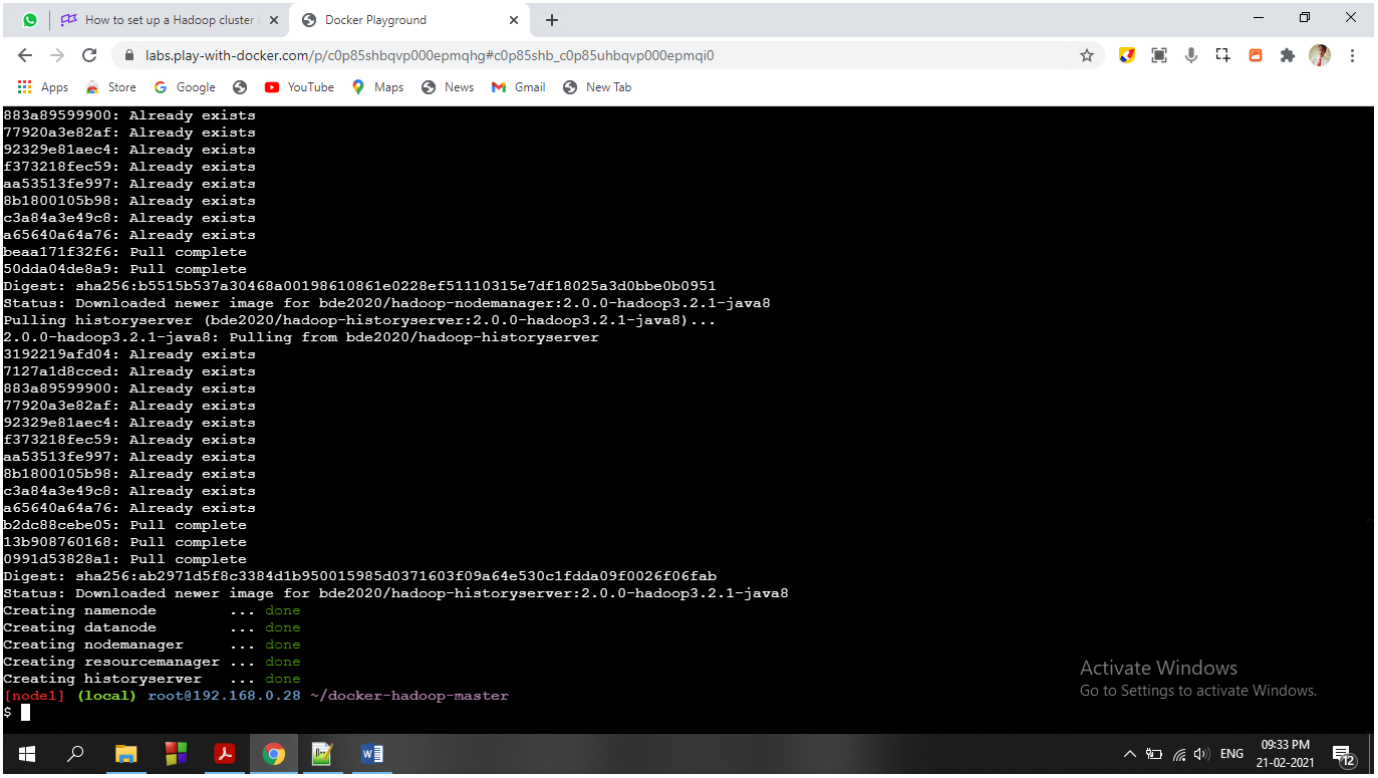
hdfs dfs -cat output/*

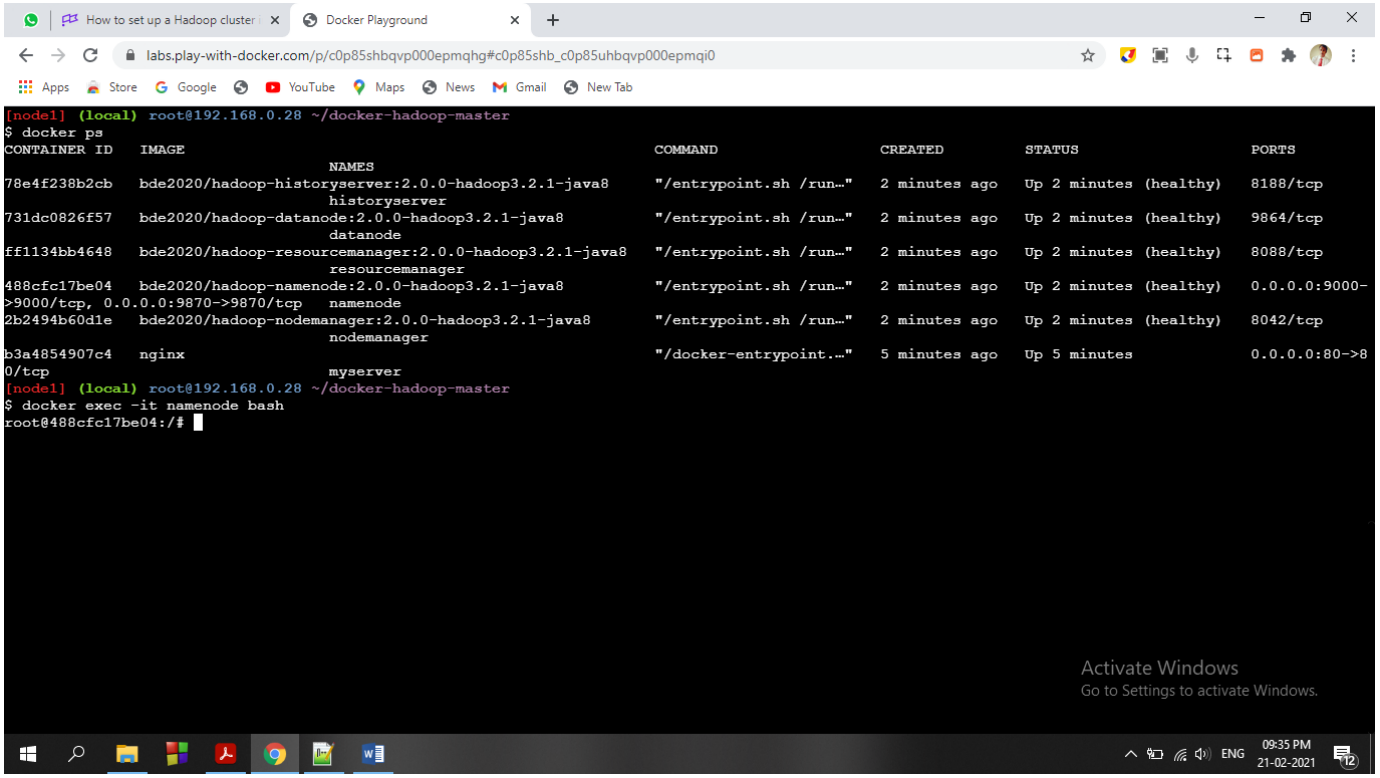
Demonstration

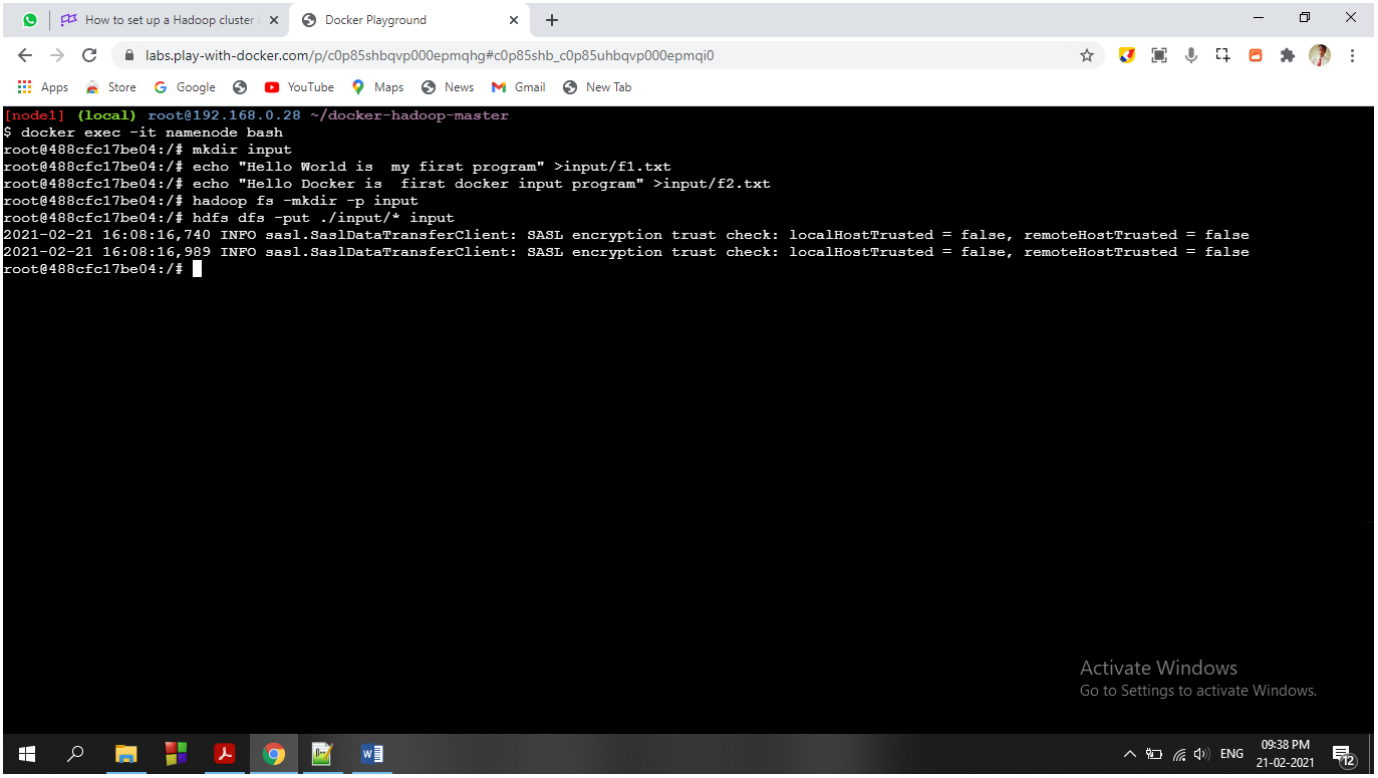


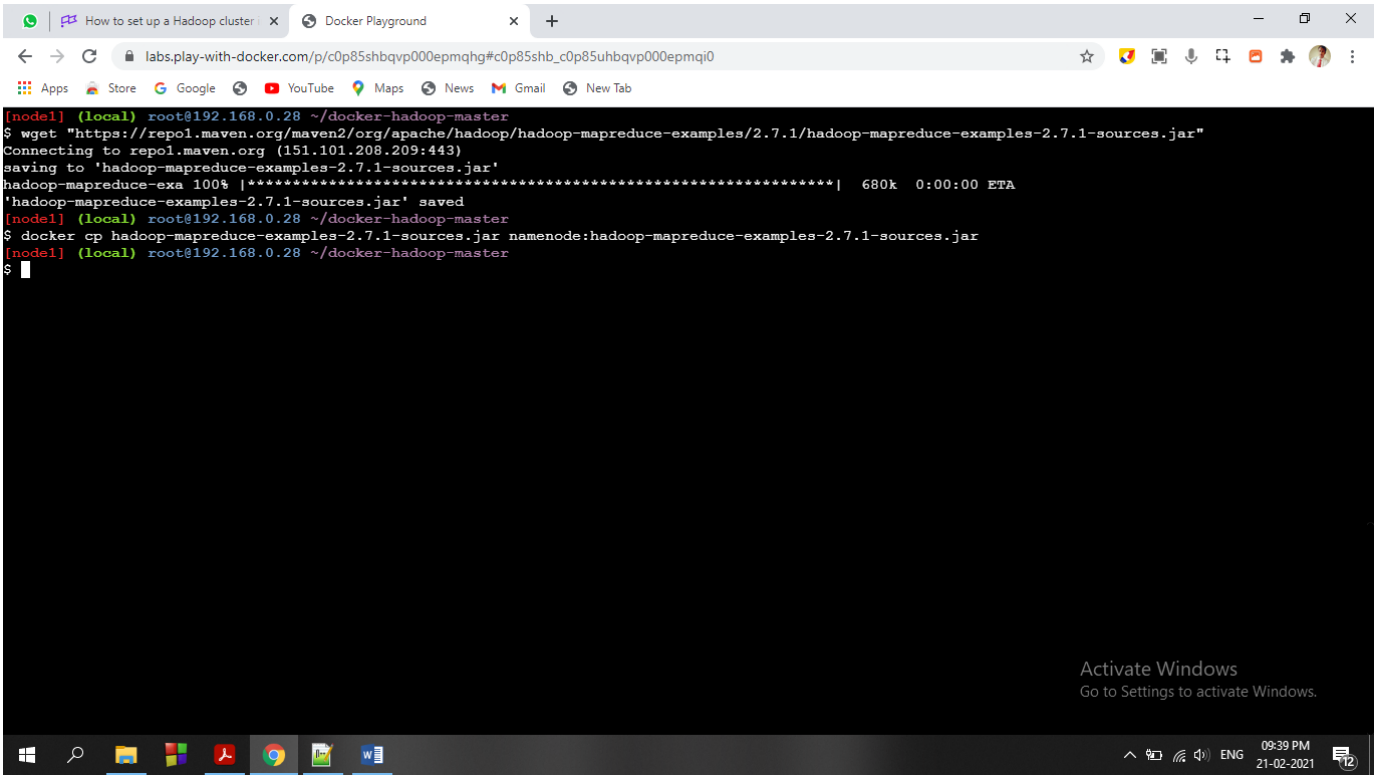

How to set up a Hadoop cluster
Docker Playground
labs.play-with-docker.com/p/c0p85shbqvp000epmqhg#c0p85shb_c0p85uhbqvp000epmqi0
Apps Store Google YouTube Maps News Gmail New Tab
[node1] (local) root@192.168.0.28 ~
$ wget "https://github.com/big-data-europe/docker-hadoop/archive/master.zip" && unzip master.zip
Connecting to github.com (140.82.121.3:443)
Connecting to codeload.github.com (140.82.113.10:443)
saving to 'master.zip'
master.zip 100% |*****| 42689 0:00:00 ETA
'master.zip' saved
Archive: master.zip
creating: docker-hadoop-master/
inflating: docker-hadoop-master/.gitignore
inflating: docker-hadoop-master/Makefile
inflating: docker-hadoop-master/README.md
creating: docker-hadoop-master/base/
inflating: docker-hadoop-master/base/Dockerfile
inflating: docker-hadoop-master/base/entrypoint.sh
creating: docker-hadoop-master/datanode/
inflating: docker-hadoop-master/datanode/Dockerfile
inflating: docker-hadoop-master/datanode/run.sh
inflating: docker-hadoop-master/docker-compose-v3.yml
inflating: docker-hadoop-master/docker-compose.yml
inflating: docker-hadoop-master/hadoop.env
creating: docker-hadoop-master/historyserver/
inflating: docker-hadoop-master/historyserver/Dockerfile
inflating: docker-hadoop-master/historyserver/run.sh
creating: docker-hadoop-master/namenode/
inflating: docker-hadoop-master/namenode/Dockerfile
inflating: docker-hadoop-master/namenode/run.sh
creating: docker-hadoop-master/nginx/
inflating: docker-hadoop-master/nginx/Dockerfile
inflating: docker-hadoop-master/nginx/bde-hadoop.css
inflating: docker-hadoop-master/nginx/default.conf
inflating: docker-hadoop-master/nginx/materialize.min.css
creating: docker-hadoop-master/nodemanager/
inflating: docker-hadoop-master/nodemanager/Dockerfile
inflating: docker-hadoop-master/nodemanager/run.sh
creating: docker-hadoop-master/resourcemanager/
Activate Windows
Go to Settings to activate Windows.
ENG 09:30 PM 21-02-2021

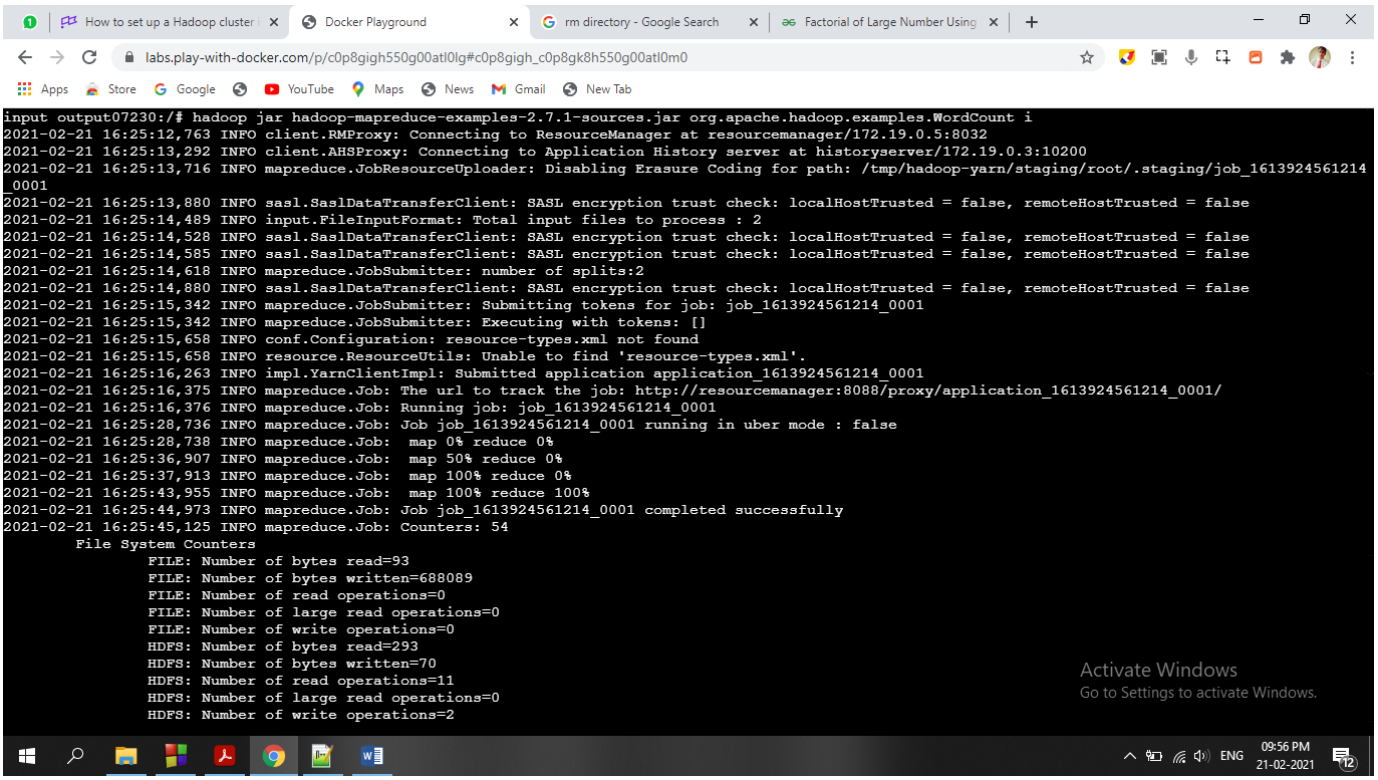

How to set up a Hadoop cluster
Docker Playground
labs.play-with-docker.com/p/c0p85shbqvp000epmqhg#c0p85shb_c0p85uhbqvp000epmqi0
Apps Store Google YouTube Maps News Gmail New Tab
[node1] (local) root@192.168.0.28 ~
$ wget "https://github.com/big-data-europe/docker-hadoop/archive/master.zip" && unzip master.zip
Connecting to github.com (140.82.121.3:443)
Connecting to codeload.github.com (140.82.113.10:443)
saving to 'master.zip'
master.zip 100% |*****| 42689 0:00:00 ETA
'master.zip' saved
Archive: master.zip
creating: docker-hadoop-master/
[node1] (local) root@192.168.0.28 ~
$ cd docker-hadoop-master/
[node1] (local) root@192.168.0.28 ~/docker-hadoop-master
$ docker-compose up -d
Creating network "docker-hadoop-master_default" with the default driver
Creating volume "docker-hadoop-master_hadoop_namenode" with default driver
Creating volume "docker-hadoop-master_hadoop_datanode" with default driver
Creating volume "docker-hadoop-master_hadoop_historyserver" with default driver
Pulling namenode (bde2020/hadoop-namenode:2.0.0-hadoop3.2.1-java8)...
2.0.0-hadoop3.2.1-java8: Pulling from bde2020/hadoop-namenode
3192219afd04: Pull complete
7127a1d8cced: Pull complete
883a89599900: Pull complete
77920a3e82af: Pull complete
92329e81aec4: Pull complete
f373218fec59: Pull complete
aa53513fe997: Pull complete
8b1800105b98: Pull complete
c3a84a3e49c8: Pull complete
a65640a64a76: Pull complete
facffb3a6de3: Pull complete
c71a6df73788: Pull complete
73b8c0ccb707: Pull complete
Digest: sha256:51ad9293ec52083c5003ef0aaab00c3dd7d6335ddf495cc1257f97a272cab4c0
Status: Downloaded newer image for bde2020/hadoop-namenode:2.0.0-hadoop3.2.1-java8
Pulling datanode (bde2020/hadoop-datanode:2.0.0-hadoop3.2.1-java8)...
2.0.0-hadoop3.2.1-java8: Pulling from bde2020/hadoop-datanode
Activate Windows
Go to Settings to activate Windows.
ENG 09:32 PM 21-02-2021

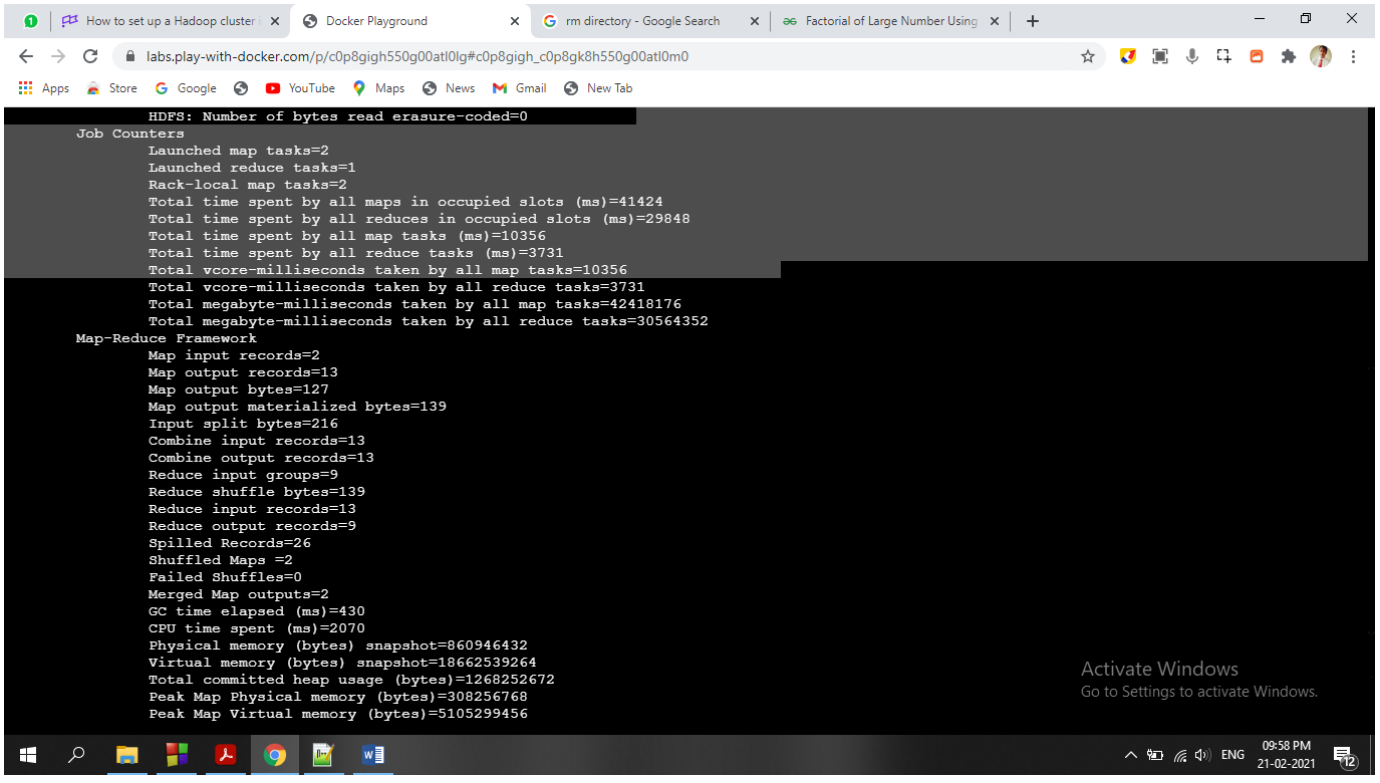

How to set up a Hadoop cluster
Docker Playground
labs.play-with-docker.com/p/c0p85shbqvp000epmqhg#c0p85shb_c0p85uhbqvp000epmqi0
Apps Store Google YouTube Maps News Gmail New Tab
883a89599900: Already exists
77920a3e82af: Already exists
92329e81aec4: Already exists
f373218fec59: Already exists
aa53513fe997: Already exists
8b1800105b98: Already exists
c3a84a3e49c8: Already exists
a65640a64a76: Already exists
beaa171f32f6: Pull complete
50dda04de8a9: Pull complete
Digest: sha256:b5515b537a30468a00198610861e0228ef51110315e7df18025a3d0bbe0b0951
Status: Downloaded newer image for bde2020/hadoop-nodemanager:2.0.0-hadoop3.2.1-java8
Pulling historyserver (bde2020/hadoop-historyserver:2.0.0-hadoop3.2.1-java8)...
2.0.0-hadoop3.2.1-java8: Pulling from bde2020/hadoop-historyserver
3192219afd04: Already exists
7127a1d8cced: Already exists
883a89599900: Already exists
77920a3e82af: Already exists
92329e81aec4: Already exists
f373218fec59: Already exists
aa53513fe997: Already exists
8b1800105b98: Already exists
c3a84a3e49c8: Already exists
a65640a64a76: Already exists
b2dc88cebe05: Pull complete
13b908760168: Pull complete
0991d53828a1: Pull complete
Digest: sha256:ab2971d5f8c3384d1b950015985d0371603f09a64e530c1fdda09f0026f06fab
Status: Downloaded newer image for bde2020/hadoop-historyserver:2.0.0-hadoop3.2.1-java8
Creating namenode ... done
Creating datanode ... done
Creating nodemanager ... done
Creating resourcemanager ... done
Creating historyserver ... done
[node1] (local) root@192.168.0.28 ~/docker-hadoop-master
$
Activate Windows
Go to Settings to activate Windows.
ENG 09:33 PM 21-02-2021

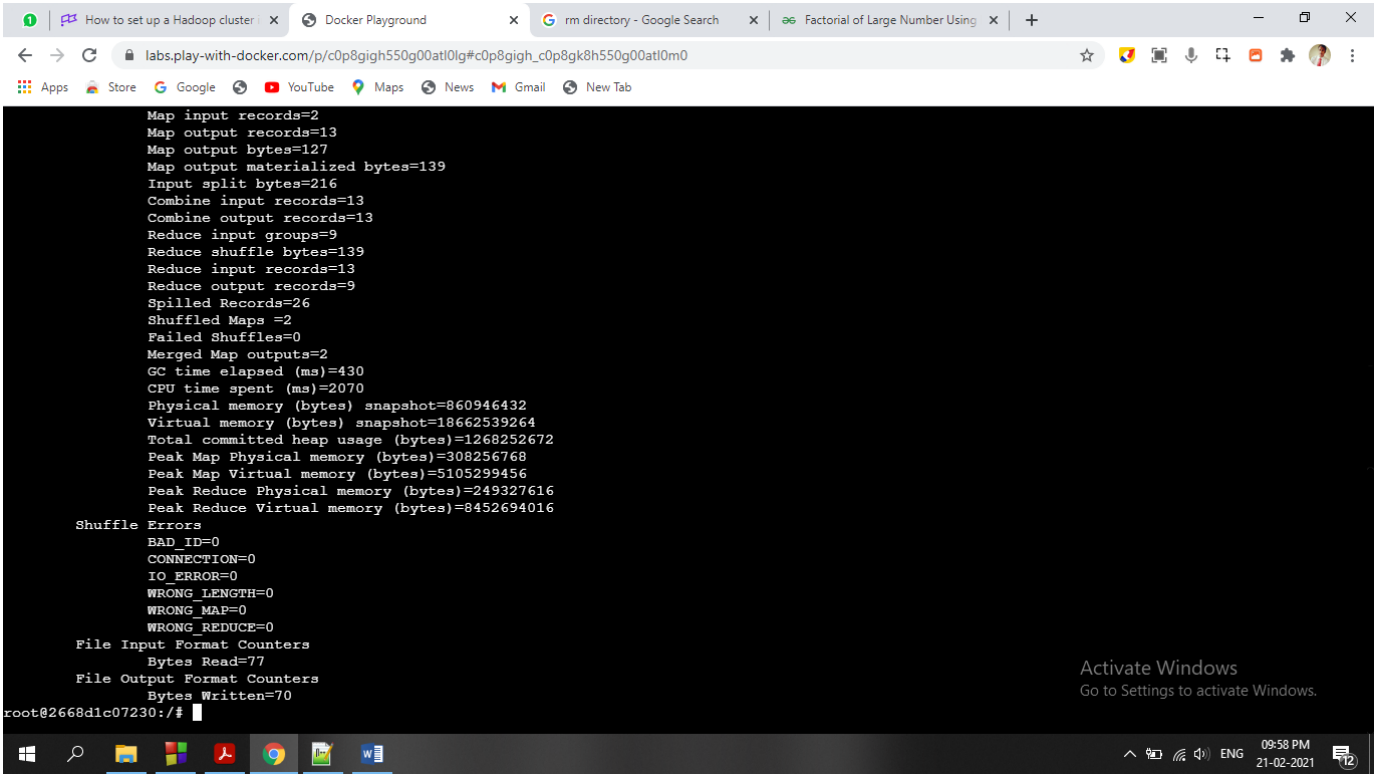

How to set up a Hadoop cluster
Docker Playground
labs.play-with-docker.com/p/c0p85shbqvp000epmqhg#c0p85shb_c0p85uhbqvp000epmqi0
Apps Store Google YouTube Maps News Gmail New Tab
[node1] (local) root@192.168.0.28 ~/docker-hadoop-master
$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
78e4f238b2cb bde2020/hadoop-historyserver:2.0.0-hadoop3.2.1-java8 "/entrypoint.sh /run…" 2 minutes ago Up 2 minutes (healthy) 8188/tcp historyserver
731dc0826f57 bde2020/hadoop-datanode:2.0.0-hadoop3.2.1-java8 "/entrypoint.sh /run…" 2 minutes ago Up 2 minutes (healthy) 9864/tcp datanode
ff1134bb4648 bde2020/hadoop-resourcemanager:2.0.0-hadoop3.2.1-java8 "/entrypoint.sh /run…" 2 minutes ago Up 2 minutes (healthy) 8088/tcp resourcemanager
488cfc17be04 bde2020/hadoop-namenode:2.0.0-hadoop3.2.1-java8 "/entrypoint.sh /run…" 2 minutes ago Up 2 minutes (healthy) 0.0.0.0:9000->9000/tcp, 0.0.0.0:9870->9870/tcp namenode
2b2494b60d1e bde2020/hadoop-nodemanager:2.0.0-hadoop3.2.1-java8 "/entrypoint.sh /run…" 2 minutes ago Up 2 minutes (healthy) 8042/tcp nodemanager
b3a4854907c4 nginx "/docker-entrypoint.…" 5 minutes ago Up 5 minutes 0.0.0.0:80->80/tcp myserver
[node1] (local) root@192.168.0.28 ~/docker-hadoop-master
$ docker exec -it namenode bash
root@488cfc17be04:/#
Activate Windows
Go to Settings to activate Windows.
ENG 09:35 PM 21-02-2021

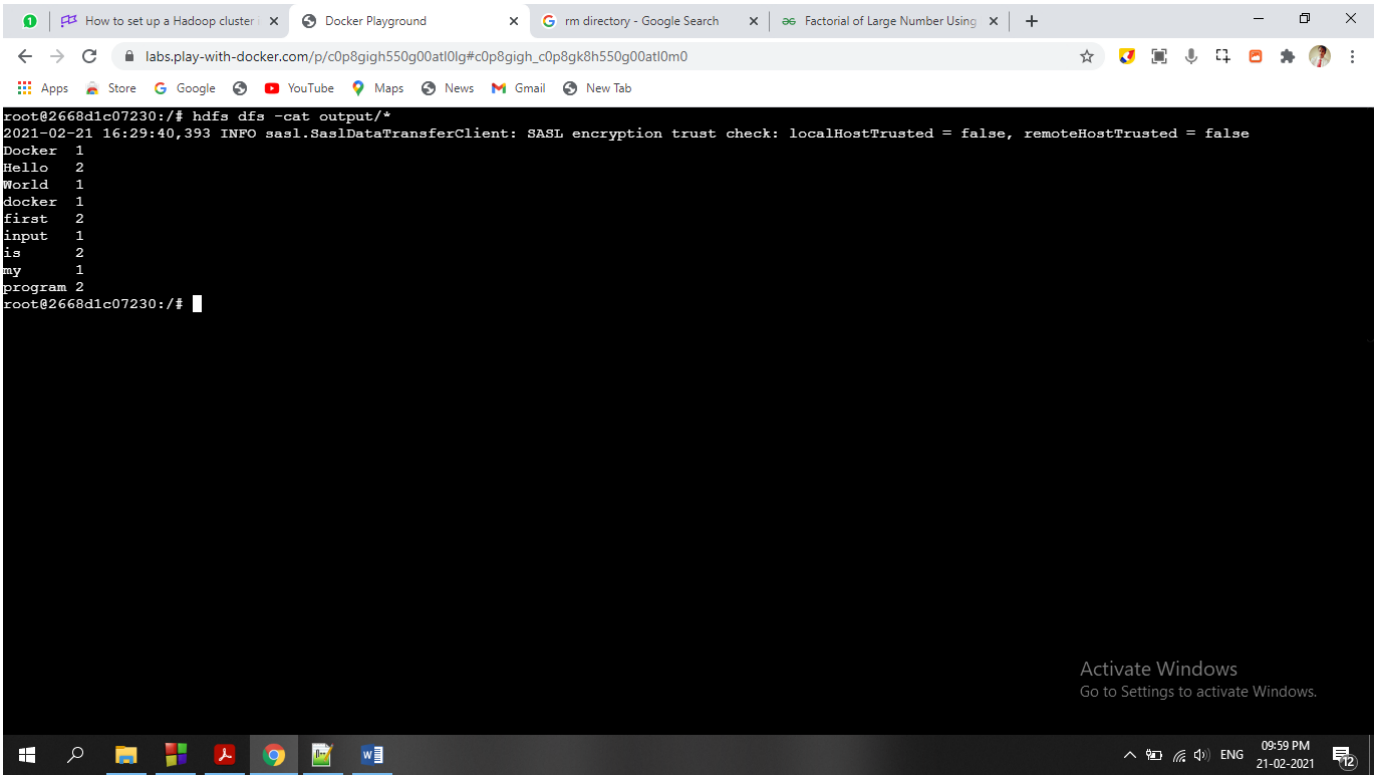

How to set up a Hadoop cluster
Docker Playground
labs.play-with-docker.com/p/c0p85shbqvp000epmqhg#c0p85shb_c0p85uhbqvp000epmqi0
Apps Store Google YouTube Maps News Gmail New Tab
[node1] (local) root@192.168.0.28 ~/docker-hadoop-master
$ docker exec -it namenode bash
root@488cfc17be04:/# mkdir input
root@488cfc17be04:/# echo "Hello World is my first program" >input/f1.txt
root@488cfc17be04:/# echo "Hello Docker is first docker input program" >input/f2.txt
root@488cfc17be04:/# hadoop fs -mkdir -p input
root@488cfc17be04:/# hdfs dfs -put ./input/* input
2021-02-21 16:08:16,740 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2021-02-21 16:08:16,989 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
root@488cfc17be04:/#
Activate Windows
Go to Settings to activate Windows.
ENG
09:38 PM
21-02-2021


How to set up a Hadoop cluster
Docker Playground
labs.play-with-docker.com/p/c0p85shbqvp000epmqhg#c0p85shb_c0p85uhbqvp000epmqi0
Apps Store Google YouTube Maps News Gmail New Tab
[node1] (local) root@192.168.0.28 ~/docker-hadoop-master
$ wget "https://repo1.maven.org/maven2/org/apache/hadoop/hadoop-mapreduce-examples/2.7.1/hadoop-mapreduce-examples-2.7.1-sources.jar"
Connecting to repo1.maven.org (151.101.208.209:443)
saving to 'hadoop-mapreduce-examples-2.7.1-sources.jar'
hadoop-mapreduce-exa 100% |***| 680k 0:00:00 ETA
'hadoop-mapreduce-examples-2.7.1-sources.jar' saved
[node1] (local) root@192.168.0.28 ~/docker-hadoop-master
$ docker cp hadoop-mapreduce-examples-2.7.1-sources.jar namenode:hadoop-mapreduce-examples-2.7.1-sources.jar
[node1] (local) root@192.168.0.28 ~/docker-hadoop-master
$
Activate Windows
Go to Settings to activate Windows.
ENG
09:39 PM
21-02-2021


How to set up a Hadoop cluster
Docker Playground
rm directory - Google Search
Factorial of Large Number Using
labs.play-with-docker.com/p/c0p8gigh550g00atl0lg#c0p8gigh_c0p8gk8h550g00atl0m0
Apps Store Google YouTube Maps News Gmail New Tab
input output07230:/# hadoop jar hadoop-mapreduce-examples-2.7.1-sources.jar org.apache.hadoop.examples.WordCount i
2021-02-21 16:25:12,763 INFO client.RMProxy: Connecting to ResourceManager at resourcemanager/172.19.0.5:8032
2021-02-21 16:25:13,292 INFO client.AHSProxy: Connecting to Application History server at historyserver/172.19.0.3:10200
2021-02-21 16:25:13,716 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/root/.staging/job_1613924561214_0001
2021-02-21 16:25:13,880 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2021-02-21 16:25:14,489 INFO input.FileInputFormat: Total input files to process : 2
2021-02-21 16:25:14,528 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2021-02-21 16:25:14,585 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2021-02-21 16:25:14,618 INFO mapreduce.JobSubmitter: number of splits:2
2021-02-21 16:25:14,880 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2021-02-21 16:25:15,342 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1613924561214_0001
2021-02-21 16:25:15,342 INFO mapreduce.JobSubmitter: Executing with tokens: []
2021-02-21 16:25:15,658 INFO conf.Configuration: resource-types.xml not found
2021-02-21 16:25:15,658 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2021-02-21 16:25:16,263 INFO impl.YarnClientImpl: Submitted application application_1613924561214_0001
2021-02-21 16:25:16,375 INFO mapreduce.Job: The url to track the job: http://resourcemanager:8088/proxy/application_1613924561214_0001/
2021-02-21 16:25:16,376 INFO mapreduce.Job: Running job: job_1613924561214_0001
2021-02-21 16:25:28,736 INFO mapreduce.Job: Job job_1613924561214_0001 running in uber mode : false
2021-02-21 16:25:28,738 INFO mapreduce.Job: map 0% reduce 0%
2021-02-21 16:25:36,907 INFO mapreduce.Job: map 50% reduce 0%
2021-02-21 16:25:37,913 INFO mapreduce.Job: map 100% reduce 0%
2021-02-21 16:25:43,955 INFO mapreduce.Job: map 100% reduce 100%
2021-02-21 16:25:44,973 INFO mapreduce.Job: Job job_1613924561214_0001 completed successfully
2021-02-21 16:25:45,125 INFO mapreduce.Job: Counters: 54
File System Counters
FILE: Number of bytes read=93
FILE: Number of bytes written=688089
FILE: Number of read operations=0
FILE: Number of large read operations=0
FILE: Number of write operations=0
HDFS: Number of bytes read=293
HDFS: Number of bytes written=70
HDFS: Number of read operations=11
HDFS: Number of large read operations=0
HDFS: Number of write operations=2
Activate Windows
Go to Settings to activate Windows.
ENG 09:56 PM 21-02-2021


How to set up a Hadoop cluster
Docker Playground
rm directory - Google Search
Factorial of Large Number Using
labs.play-with-docker.com/p/c0p8gigh550g00atl0lg#c0p8gigh_c0p8gk8h550g00atl0m0
Apps Store Google YouTube Maps News Gmail New Tab
HDFS: Number of bytes read erasure-coded=0
Job Counters
Launched map tasks=2
Launched reduce tasks=1
Rack-local map tasks=2
Total time spent by all maps in occupied slots (ms)=41424
Total time spent by all reduces in occupied slots (ms)=29848
Total time spent by all map tasks (ms)=10356
Total time spent by all reduce tasks (ms)=3731
Total vcore-milliseconds taken by all map tasks=10356
Total vcore-milliseconds taken by all reduce tasks=3731
Total megabyte-milliseconds taken by all map tasks=42418176
Total megabyte-milliseconds taken by all reduce tasks=30564352
Map-Reduce Framework
Map input records=2
Map output records=13
Map output bytes=127
Map output materialized bytes=139
Input split bytes=216
Combine input records=13
Combine output records=13
Reduce input groups=9
Reduce shuffle bytes=139
Reduce input records=13
Reduce output records=9
Spilled Records=26
Shuffled Maps =2
Failed Shuffles=0
Merged Map outputs=2
GC time elapsed (ms)=430
CPU time spent (ms)=2070
Physical memory (bytes) snapshot=860946432
Virtual memory (bytes) snapshot=18662539264
Total committed heap usage (bytes)=1268252672
Peak Map Physical memory (bytes)=308256768
Peak Map Virtual memory (bytes)=5105299456
Activate Windows
Go to Settings to activate Windows.
ENG 09:58 PM 21-02-2021


How to set up a Hadoop cluster
Docker Playground
rm directory - Google Search
Factorial of Large Number Using
labs.play-with-docker.com/p/c0p8gigh550g00atl0lg#c0p8gigh_c0p8gk8h550g00atl0m0
Apps Store Google YouTube Maps News Gmail New Tab
Map input records=2
Map output records=13
Map output bytes=127
Map output materialized bytes=139
Input split bytes=216
Combine input records=13
Combine output records=13
Reduce input groups=9
Reduce shuffle bytes=139
Reduce input records=13
Reduce output records=9
Spilled Records=26
Shuffled Maps =2
Failed Shuffles=0
Merged Map outputs=2
GC time elapsed (ms)=430
CPU time spent (ms)=2070
Physical memory (bytes) snapshot=860946432
Virtual memory (bytes) snapshot=18662539264
Total committed heap usage (bytes)=1268252672
Peak Map Physical memory (bytes)=308256768
Peak Map Virtual memory (bytes)=5105299456
Peak Reduce Physical memory (bytes)=249327616
Peak Reduce Virtual memory (bytes)=8452694016
Shuffle Errors
BAD_ID=0
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0
File Input Format Counters
Bytes Read=77
File Output Format Counters
Bytes Written=70
root@2668d1c07230:/#
Activate Windows
Go to Settings to activate Windows.
ENG
09:58 PM
21-02-2021


How to set up a Hadoop cluster
Docker Playground
rm directory - Google Search
Factorial of Large Number Using
labs.play-with-docker.com/p/c0p8gigh550g00atl0lg#c0p8gigh_c0p8gk8h550g00atl0m0
Apps Store Google YouTube Maps News Gmail New Tab
root@2668d1c07230:/# hdfs dfs -cat output/*
2021-02-21 16:29:40,393 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
Docker 1
Hello 2
World 1
docker 1
first 2
input 1
is 2
my 1
program 2
root@2668d1c07230:/#
Activate Windows
Go to Settings to activate Windows.
ENG
09:59 PM
21-02-2021

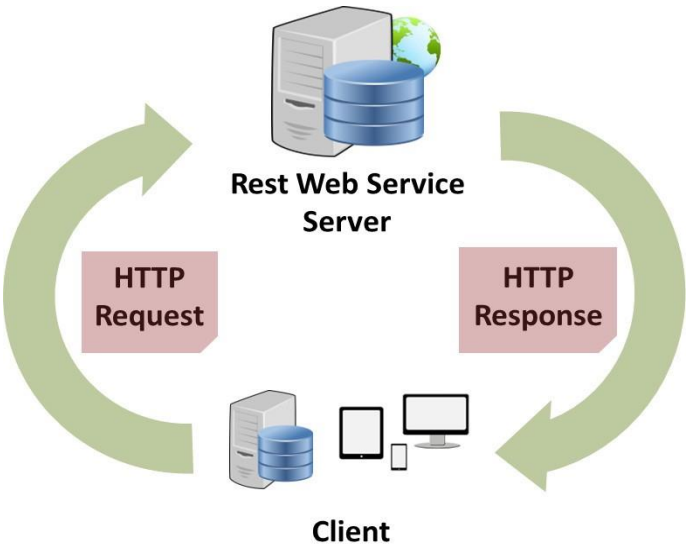
Designing an web application that uses REST API's

Objective: To learn how to use/develop REST API

Task: Design any application that uses the REST interface in json format and xml format.

Introduction

A REST API (also known as RESTful API) is an application programming interface (API or web API) that conforms to the constraints of REST architectural style and allows for interaction with RESTful web services. REST stands for representational state transfer and was created by computer scientist Roy Fielding.
REST is a set of architectural constraints, not a protocol or a standard. API developers can implement REST in a variety of ways.When a client request is made via a RESTful API, it transfers a representation of the state of the resource to the requester or endpoint. This information, or representation, is delivered in one of several formats via HTTP: JSON (Javascript Object Notation), HTML, XLT, Python, PHP, or plain text. JSON is the most generally popular programming language to use because, despite its name, it's language-agnostic, as well as readable by both humans and machines.



Procedure

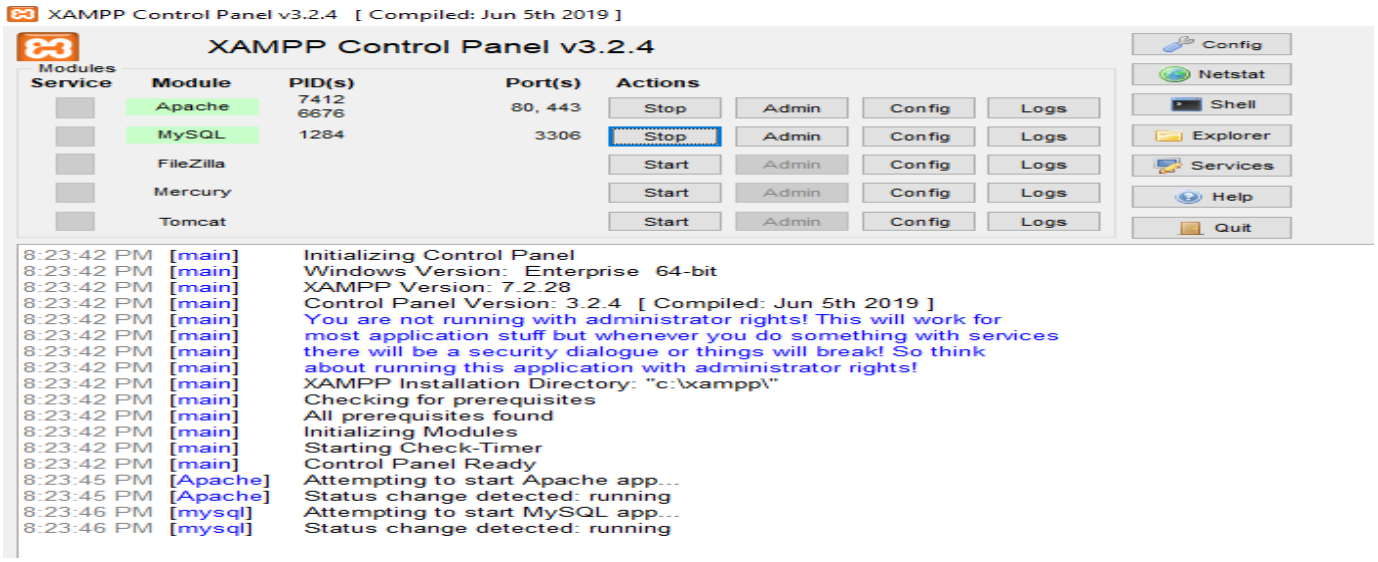
1. For developing I have used PHP language and Localhost (apache) as my server
2. For testing api's we have used Postman
3. So for this we need to install either XAMPP or WAMP for apache server and php development. And install Postman for testing
4. Now design the code for handling various requests for CRUD operations such as GET,POST,PUT,DELETE,UPDATE, etc

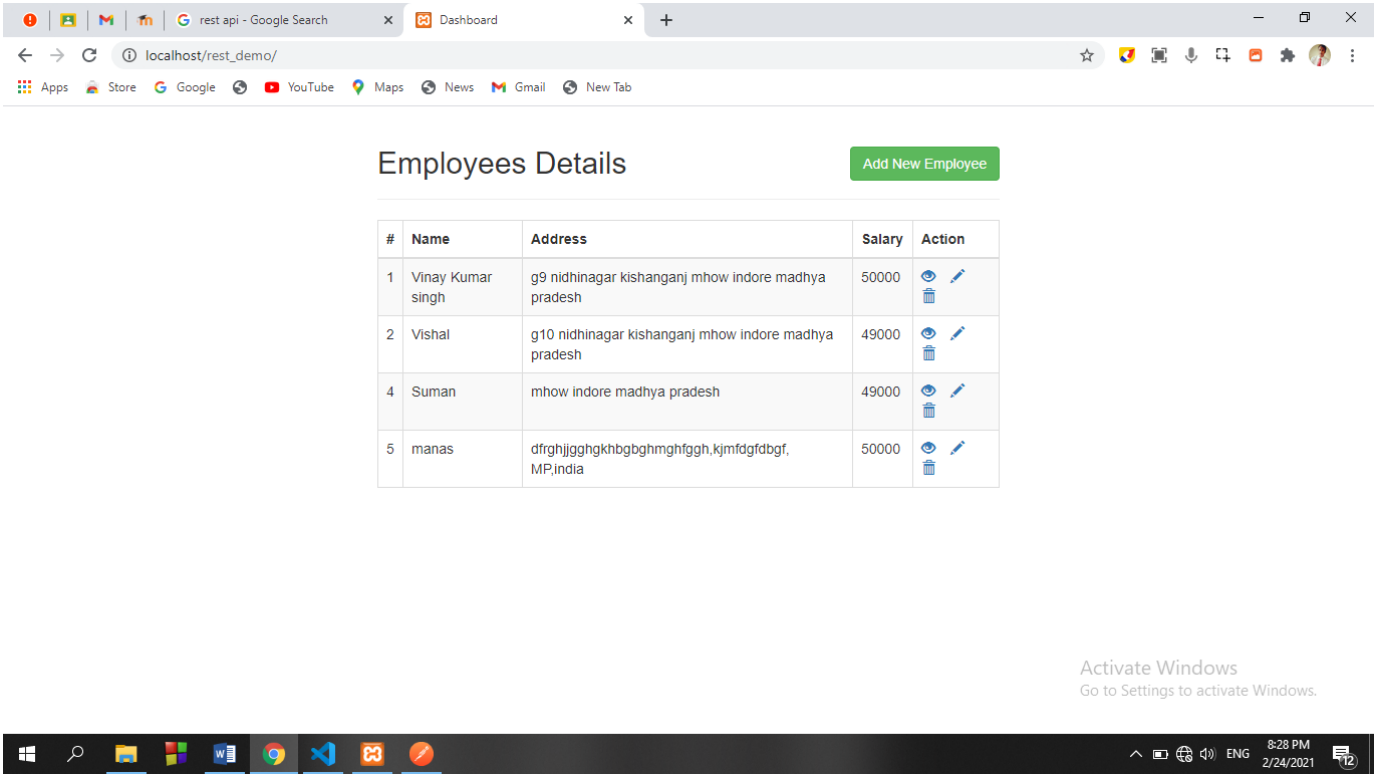
```php
<?php
require_once "config.php";
$method = $_SERVER['REQUEST_METHOD'];
if ($method == 'POST'){
    $input_name = $_POST["name"];
    $input_address = trim($_POST["address"]);
    $input_salary = trim($_POST["salary"]);
    $sql = "INSERT INTO employees (name, address, salary) VALUES (?, ?, ?)";
    $stmt = mysqli_prepare($link, $sql);
    mysqli_stmt_bind_param($stmt, "sss", $input_name, $input_address, $input_salary);
    if(mysqli_stmt_execute($stmt)){
        //  Redirect to landing page
        echo json_encode("Records Added successfully.");
    } else{
        echo "Oops! Something went wrong. Please try again .";
    }
    mysqli_stmt_close($stmt);
    mysqli_close($link);
}
elseif ($method == 'GET'){
    $sql = "SELECT * FROM employees";
    $result = mysqli_query($link, $sql);
    $r=mysqli_fetch_all($result);
    //$row = mysqli_fetch_array($result);
    mysqli_free_result($result);
    mysqli_close($link);
    echo json_encode($r);
}


elseif ($method == 'DELETE'){
    $sql = "DELETE FROM employees WHERE id = ?";
    $stmt = mysqli_prepare($link, $sql);
    mysqli_stmt_bind_param($stmt, "i", $param_id);
    $param_id = trim($_POST["id"]);
    if(mysqli_stmt_execute($stmt)){
        //  Redirect to landing page
        echo json_encode("Records deleted successfully.");
    } else{
        echo "Oops! Something went wrong. Please try again .";
    }
}
else {
    echo json_encode("Invalid method  try again with another method")
}
?>
```

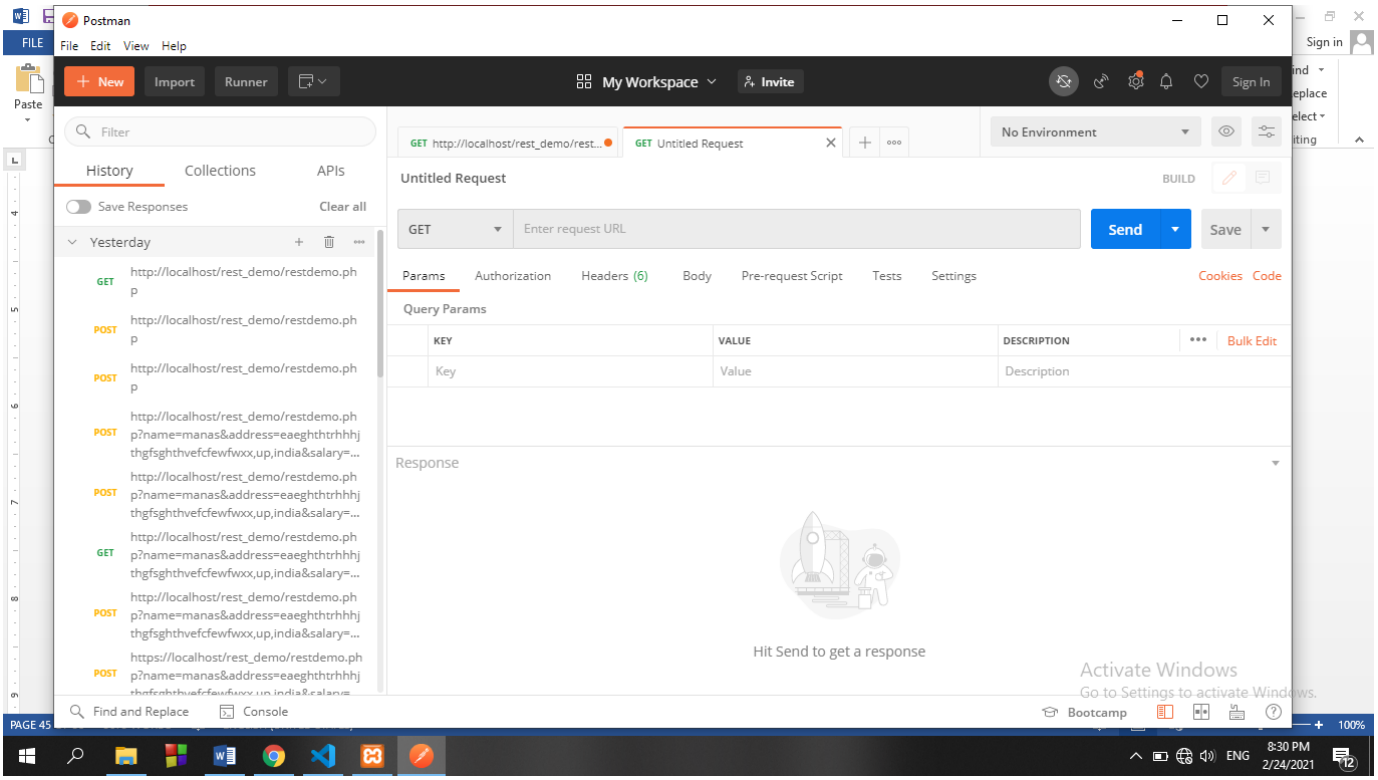
5. Launch code on your localhost server and then through postman test your Api's

Here are some screenshots mentioned in demonstration section

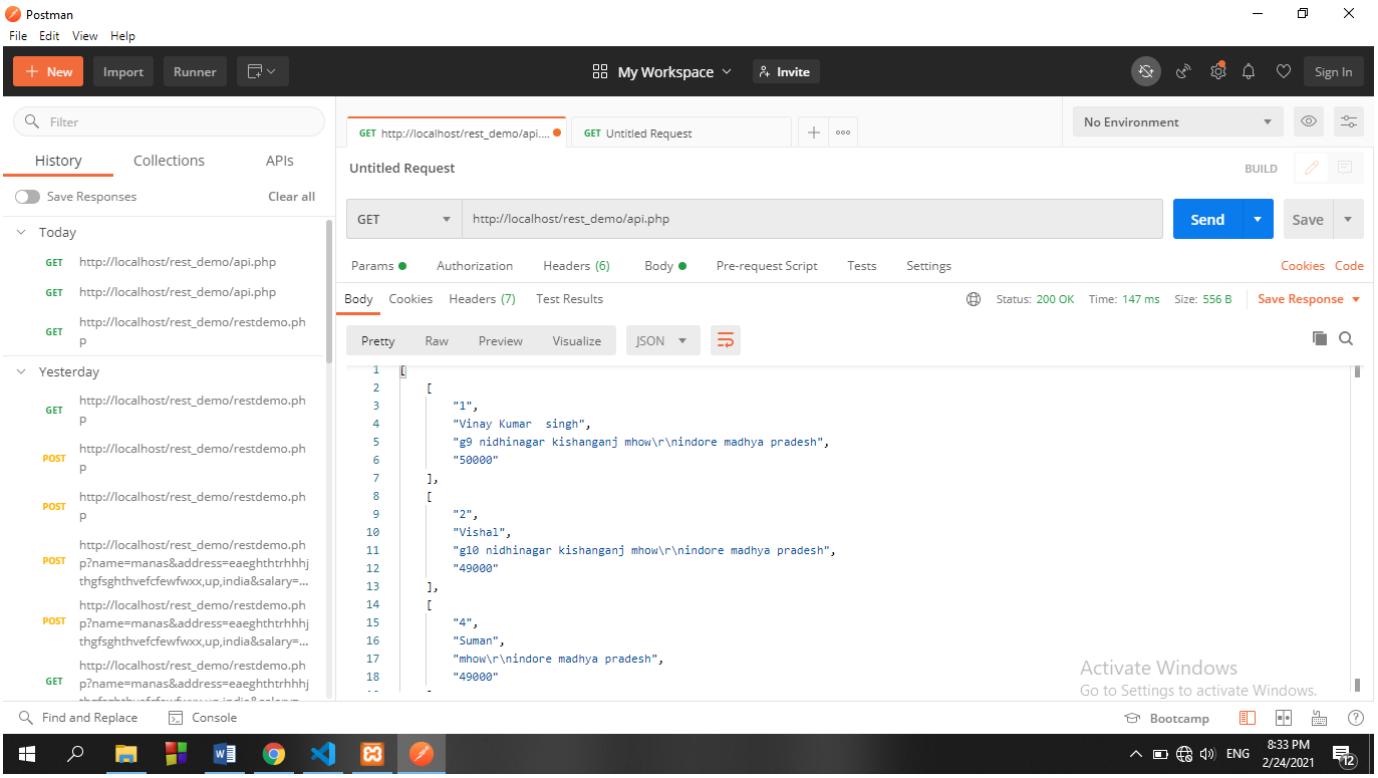
Demonstration



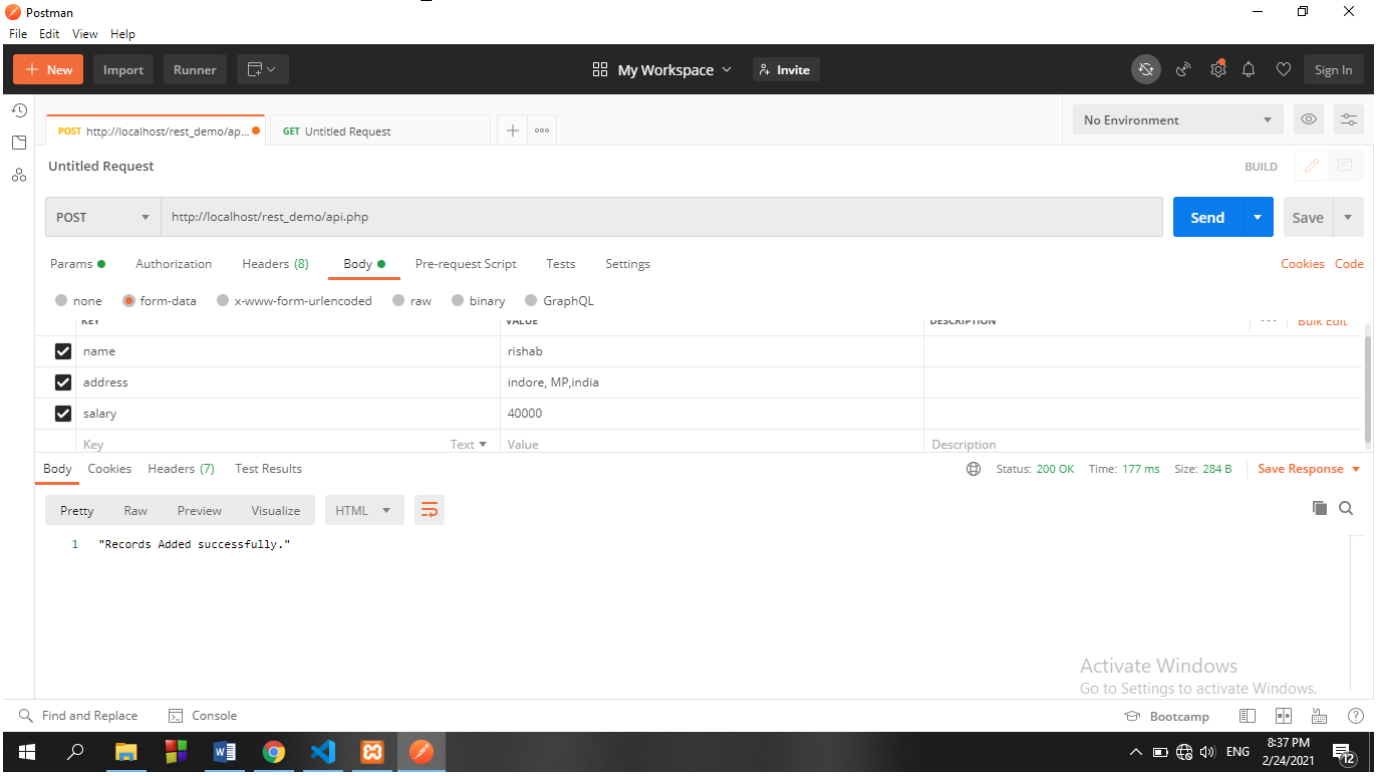




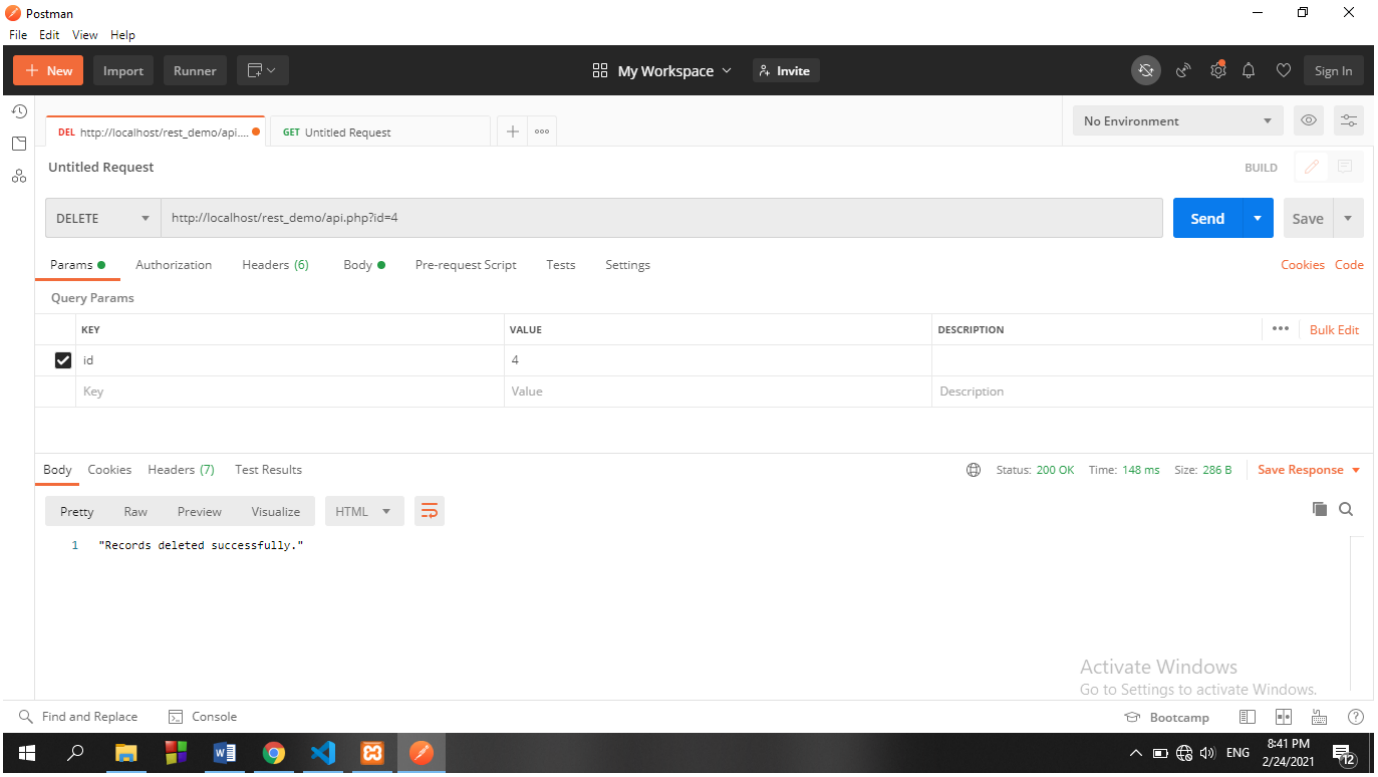
GET REQUEST OUTPUT-



For POST method output-



For DELETE method output-



Study and implementation of cloud load management techniques

Objective: To learn QOS techniques in cloud like–Load Management

Task: Install cloudsim and simulate load balancing algorithm.

Introduction:

Cloud Computing is set of resources available on pay per use model. The User requests on cloud for the services which is increasing day by day and for the better performance it should be balanced such that all the requests should be answered in minimum time. Therefore, Load Balancing is required and it is one of the major Issues in Cloud Computing. Moreover, it is termed as Load Balancing is NP-Complete Problem because as the number of request increases, balancing the load becomes tougher. This paper gives a genetic algorithm (GA) based approach for load balancing in cloud. For population initialization, priority of request is considered based on their time. The idea behind the considering the priority is to get real world visualization. In Real World Scenario requests have some priorities which we can used for our Algorithm. Simulation of the proposed method is done using Cloud Analyst. Simulation Results shows that proposed method performs well then existing once with some real world picture.

Code

```
import java.text.DecimalFormat;
import java.util.ArrayList;
import java.util.Random;
import java.util.Calendar;
import java.util.LinkedList;
import java.util.List;
import org.cloudbus.cloudsim.Cloudlet;
import org.cloudbus.cloudsim.CloudletSchedulerSpaceShared;
import org.cloudbus.cloudsim.Datacenter;
import org.cloudbus.cloudsim.DatacenterBroker;
import org.cloudbus.cloudsim.DatacenterCharacteristics;
import org.cloudbus.cloudsim.Host;
import org.cloudbus.cloudsim.Log;
import org.cloudbus.cloudsim.Pe;
import org.cloudbus.cloudsim.Storage;
import org.cloudbus.cloudsim.UtilizationModel;
import org.cloudbus.cloudsim.UtilizationModelFull;
import org.cloudbus.cloudsim.Vm;
```

```
import org.cloudbus.cloudsim.VmAllocationPolicySimple;
import org.cloudbus.cloudsim.VmSchedulerTimeShared;
import org.cloudbus.cloudsim.core.CloudSim;
import org.cloudbus.cloudsim.core.CloudSimTags;
import org.cloudbus.cloudsim.lists.VmList;
import org.cloudbus.cloudsim.provisioners.BwProvisionerSimple;
import org.cloudbus.cloudsim.provisioners.PeProvisionerSimple;
import org.cloudbus.cloudsim.provisioners.RamProvisionerSimple;

/**
 * An example showing how to create scalable simulations.
 */
public class GeneticAlgorithmCloud {

	/** The cloudlet list. */
	private static List<Cloudlet> cloudletList;

	/** The vmlist. */
	private static List<Vm> vmlist;
	private static List<Cloudlet> finalcloudletList;

	/** The vmlist. */
	private static List<Vm> finalvmlist;

	private static List<Vm> createVM(int userId, int vms) {

		// Creates a container to store VMs. This list is passed to the broker
		// later
		LinkedList<Vm> list = new LinkedList<Vm>();

		// VM Parameters
		long size = 10000; // image size (MB)
		int ram = 512; // vm memory (MB)
		int mips = 500;
		long bw = 10;
		int pesNumber = 4;// number of cpus
		String vmm = "Xen"; // VMM name
		Random rOb = new Random();
		// create VMs
		Vm[] vm = new Vm[vms];
		for (int i = 0; i < vms; i++) {
			vm[i] = new Vm(i, userId, mips + rOb.nextInt(500), pesNumber, ram,
					bw , size, vmm,
					new CloudletSchedulerSpaceShared());
			// for creating a VM with a space shared scheduling policy for
			// cloudlets:
			// vm[i] = Vm(i, userId, mips, pesNumber, ram, bw, size, priority,
			// vmm, new CloudletSchedulerSpaceShared());

			list.add(vm[i]);
		}

		return list;
	}

	private static List<Cloudlet> createCloudlet(int userId, int cloudlets) {
```

```
		// Creates a container to store Cloudlets
		LinkedList<Cloudlet> list = new LinkedList<Cloudlet>();

		// cloudlet parameters
		long length = 1000;
		long fileSize = 300;
		long outputSize = 300;
		int pesNumber = 1;
		UtilizationModel utilizationModel = new UtilizationModelFull();

		Cloudlet[] cloudlet = new Cloudlet[cloudlets];
		Random randomObj = new Random();

		for (int i = 0; i < cloudlets; i++) {
			int x = (int) (Math.random() * ((2000 - 1) + 1)) + 1;
			cloudlet[i] = new Cloudlet(i, (length ), pesNumber, fileSize,
					outputSize, utilizationModel, utilizationModel,
					utilizationModel);
			// setting the owner of these Cloudlets
			cloudlet[i].setUserId(userId);
			list.add(cloudlet[i]);
		}

		return list;
	}

	// /////////////////////// STATIC METHODS //////////////////////

	/**
	 * Creates main() to run this example
* @param args
	 */
	public static void main(String[] args) {
		Log.printLine("Starting GeneticAlgorithmCloudSimulation...");

		try {
			// First step: Initialize the CloudSim package. It should be called
			// before creating any entities.
			int num_user = 1; // number of grid users
			Calendar calendar = Calendar.getInstance();
			boolean trace_flag = false; // mean trace events

			// Initialize the CloudSim library
			CloudSim.init(num_user, calendar, trace_flag);

			// Second step: Create Datacenters
			// Datacenters are the resource providers in CloudSim. We need at
			// list one of them to run a CloudSim simulation
			@SuppressWarnings("unused")
			Datacenter datacenter0 = createDatacenter("Datacenter_0");
			@SuppressWarnings("unused")
			Datacenter datacenter1 = createDatacenter("Datacenter_1");

			// Third step: Create Broker
			DatacenterBroker1 broker = createBroker();
			int brokerId = broker.getId();
```

```
		// Fourth step: Create VMs and Cloudlets and send them to broker
		vmlist = createVM(brokerId, 25); // creating 20 vms
		cloudletList = createCloudlet(brokerId, 50); // creating 20 cloudlets

		List<Cloudlet> sortedList = new ArrayList<Cloudlet>();
		for(Cloudlet cloudlet:cloudletList){
			sortedList.add(cloudlet);
		}
		int numCloudlets=sortedList.size();
		for(int i=0;i<numCloudlets;i++){
			Cloudlet tmp=sortedList.get(i);
			int idx=i;
			for(int j=i+1;j<numCloudlets;j++)
			{
				if(sortedList.get(j).getCloudletLength()<tmp.getCloudletLength())
				{
					idx=j;
					tmp=sortedList.get(j);
				}
			}
			Cloudlet tmp2 = sortedList.get(i);
			sortedList.set(i, tmp);
			sortedList.set(idx,tmp2);
		}

		ArrayList<Vm> sortedListVm = new ArrayList<Vm>();
	ArrayList<Vm> toBeUsedVm = new ArrayList<Vm>();
	ArrayList<Vm> leftOutVm = new ArrayList<Vm>();
	for(Vm vm:vmlist){
		sortedListVm.add(vm);
	}
	int numVms=sortedListVm.size();

	for(int i=0;i<numVms;i++){
		Vm tmp=sortedListVm.get(i);
		int idx=i;
		if(i<numCloudlets)
			toBeUsedVm.add(tmp);
		else
			leftOutVm.add(tmp);
		for(int j=i+1;j<numVms;j++)
		{
			if(sortedListVm.get(j).getMips()>tmp.getMips())
			{
				idx=j;
				tmp=sortedListVm.get(j);
			}
		}
		Vm tmp2 = sortedListVm.get(i);
		sortedListVm.set(i, tmp);
		sortedListVm.set(idx,tmp2);
	}
	ArrayList<Chromosomes> initialPopulation = new ArrayList<Chromosomes>();
	for(int j=0;j<numCloudlets;j++)
	{
```

```
            ArrayList<Gene> firstChromosome = new ArrayList<Gene>();

            for(int i=0;i<numCloudlets;i++)
            {
                    int k=(i+j)%numVms;
                    k=(k+numCloudlets)%numCloudlets;
                    Gene geneObj = new Gene(sortedList.get(i),sortedListVm.get(k));
                    firstChromosome.add(geneObj);
            }
            Chromosomes chromosome = new Chromosomes(firstChromosome);
            initialPopulation.add(chromosome);
}

int populationSize=initialPopulation.size();
Random random = new Random();
for(int itr=0;itr<20;itr++)
{
            int index1,index2;
            index1=random.nextInt(populationSize) % populationSize;
            index2=random.nextInt(populationSize) % populationSize;
            ArrayList<Gene> l1= new ArrayList<Gene>();
            l1=initialPopulation.get(index1).getGeneList();
            Chromosomes chromosome1 = new Chromosomes(l1);
            ArrayList<Gene> l2= new ArrayList<Gene>();
            l2=initialPopulation.get(index2).getGeneList();
            Chromosomes chromosome2 = new Chromosomes(l2);
            double rangeMin = 0.0f;
   double rangeMax = 1.0f;
   Random r = new Random();
   double crossProb = rangeMin + (rangeMax - rangeMin) * r.nextDouble();
            if(crossProb<0.5)
            {
                    int i,j;
                    i=random.nextInt(numCloudlets) % numCloudlets;
                    j=random.nextInt(numCloudlets) % numCloudlets;
                    Vm vm1 = l1.get(i).getVmFromGene();
                    Vm vm2 = l2.get(j).getVmFromGene();
                    chromosome1.updateGene(i, vm2);
                    chromosome2.updateGene(j, vm1);
                    initialPopulation.set(index1, chromosome1);
                    initialPopulation.set(index2, chromosome2);
            }
            double mutProb = rangeMin + (rangeMax - rangeMin) * r.nextDouble();
            if(mutProb<0.5)
            {
                    int i;
                    i=random.nextInt(populationSize) % populationSize;
                    ArrayList<Gene> l= new ArrayList<Gene>();
                    l=initialPopulation.get(i).getGeneList();
                    Chromosomes mutchromosome = new Chromosomes(l);
                    int j;
                    j=random.nextInt(numCloudlets) % numCloudlets;
                    Vm vm1 = sortedListVm.get(0);
                    mutchromosome.updateGene(j,vm1);
            }
}
```

```
int fittestIndex=0;
double time=1000000;

for(int i=0;i<populationSize;i++)
{
        ArrayList<Gene> l= new ArrayList<Gene>();
        l=initialPopulation.get(i).getGeneList();
        double sum=0;
        for(int j=0;j<numCloudlets;j++)
        {
                Gene g = l.get(j);
                Cloudlet c = g.getCloudletFromGene();
                Vm v = g.getVmFromGene();
                double temp = c.getCloudletLength()/v.getMips();
                sum+=temp;
        }
        if(sum<time)
        {
                time=sum;
                fittestIndex=i;
        }
}

ArrayList<Gene> result = new ArrayList<Gene>();
result = initialPopulation.get(fittestIndex).getGeneList();

        List<Cloudlet> finalcloudletList = new ArrayList<Cloudlet>();
        List<Vm> finalvmlist = new ArrayList<Vm>();




        for(int i=0;i<result.size();i++)
        {
                finalcloudletList.add(result.get(i).getCloudletFromGene());
                finalvmlist.add(result.get(i).getVmFromGene());
                Vm vm=result.get(i).getVmFromGene();
                //Log.printLine("################ VM FROM GENE  "+vm.getId());
        }

        broker.submitVmList(finalvmlist);
        broker.submitCloudletList(finalcloudletList);

        // Fifth step: Starts the simulation
        CloudSim.startSimulation();

        // Final step: Print results when simulation is over
        List<Cloudlet> newList = broker.getCloudletReceivedList();

        CloudSim.stopSimulation();

        printCloudletList(newList);

        Log.printLine("CloudSimExample6 finished!");
} catch (Exception e) {
        e.printStackTrace();
```

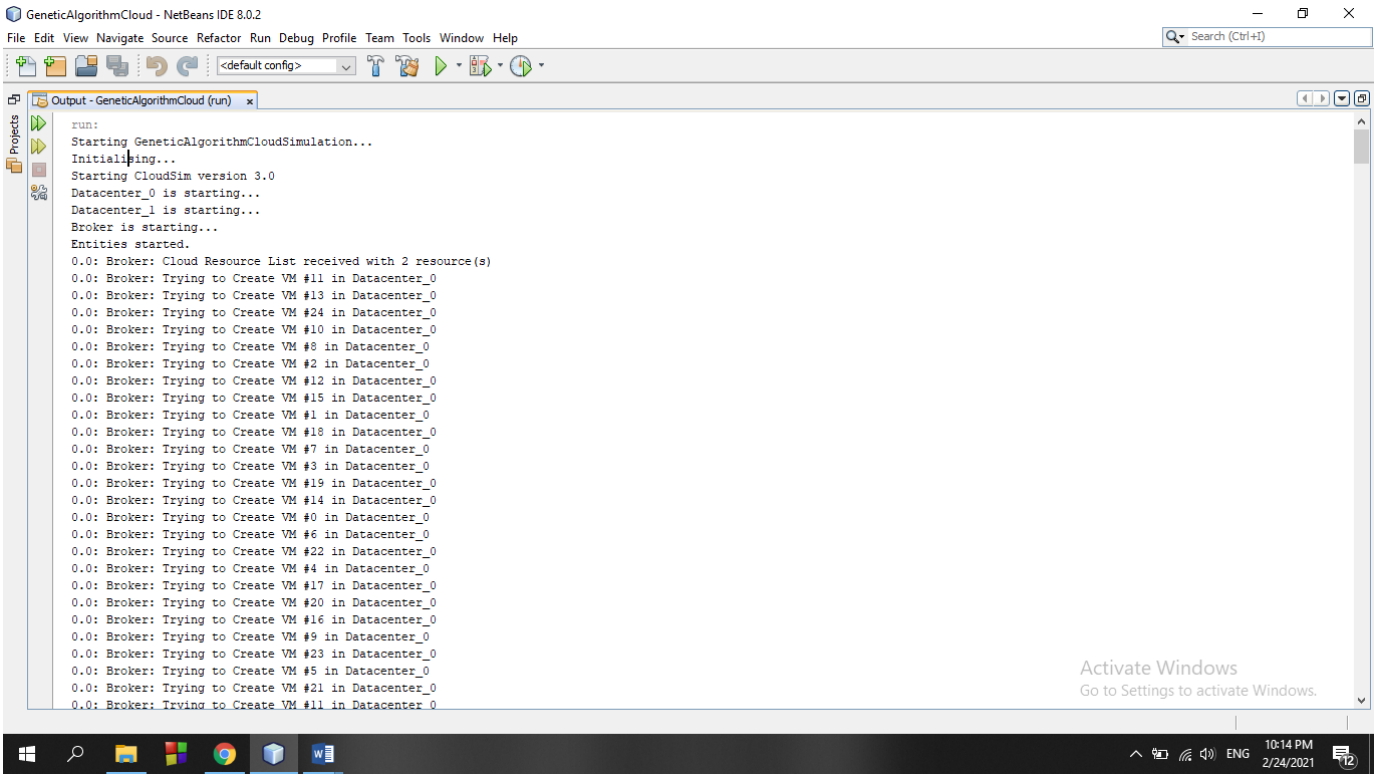
```
                Log.printLine("The simulation has been terminated due to an unexpected error");
            }
        }

        private static Datacenter createDatacenter(String name) {

            // Here are the steps needed to create a PowerDatacenter:
            // 1. We need to create a list to store one or more
            // Machines
            List<Host> hostList = new ArrayList<Host>();

            // 2. A Machine contains one or more PEs or CPUs/Cores. Therefore,
            // should
            // create a list to store these PEs before creating
            // a Machine.
            List<Pe> peList1 = new ArrayList<Pe>();

            int mips = 1000;

            // 3. Create PEs and add these into the list.
            // for a quad-core machine, a list of 4 PEs is required:
            peList1.add(new Pe(0, new PeProvisionerSimple(mips))); // need to store

            // Pe id and

            // MIPS Rating
            peList1.add(new Pe(1, new PeProvisionerSimple(mips)));
            peList1.add(new Pe(2, new PeProvisionerSimple(mips)));
            peList1.add(new Pe(3, new PeProvisionerSimple(mips)));

            // Another list, for a dual-core machine
            List<Pe> peList2 = new ArrayList<Pe>();

            peList2.add(new Pe(0, new PeProvisionerSimple(mips)));
            peList2.add(new Pe(1, new PeProvisionerSimple(mips)));

            // 4. Create Hosts with its id and list of PEs and add them to the list
            // of machines
            int hostId = 0;
            int ram = 204800; // host memory (MB)
            long storage = 10000000; // host storage
            int bw = 100000;

            hostList.add(new Host(hostId, new RamProvisionerSimple(ram),
                                  new BwProvisionerSimple(bw), storage, peList1,
                                  new VmSchedulerTimeShared(peList1))); // This is our first
                                                                                                  //
machine

            hostId++;

            hostList.add(new Host(hostId, new RamProvisionerSimple(ram),
                                  new BwProvisionerSimple(bw), storage, peList2,
                                  new VmSchedulerTimeShared(peList2))); // Second machine

            // To create a host with a space-shared allocation policy for PEs to
```

```
// VMs:
// hostList.add(
// new Host(
// hostId,
// new CpuProvisionerSimple(peList1),
// new RamProvisionerSimple(ram),
// new BwProvisionerSimple(bw),
// storage,
// new VmSchedulerSpaceShared(peList1)
// )
// );

// To create a host with a oportunistic space-shared allocation policy
// for PEs to VMs:
// hostList.add(
// new Host(
// hostId,
// new CpuProvisionerSimple(peList1),
// new RamProvisionerSimple(ram),
// new BwProvisionerSimple(bw),
// storage,
// new VmSchedulerOportunisticSpaceShared(peList1)
// )
// );

// 5. Create a DatacenterCharacteristics object that stores the
// properties of a data center: architecture, OS, list of
// Machines, allocation policy: time- or space-shared, time zone
// and its price (G$/Pe time unit).
String arch = "x86"; // system architecture
String os = "Linux"; // operating system
String vmm = "Xen";
double time_zone = 10.0; // time zone this resource located
double cost = 3.0; // the cost of using processing in this resource
double costPerMem = 0.05; // the cost of using memory in this resource
double costPerStorage = 0.1; // the cost of using storage in this
                                                        // resource
double costPerBw = 0.1; // the cost of using bw in this resource
LinkedList<Storage> storageList = new LinkedList<Storage>(); // we are

                    // not

                    // adding

                    // SAN

                    // devices

                    // by

                    // now

DatacenterCharacteristics characteristics = new DatacenterCharacteristics(
                    arch, os, vmm, hostList, time_zone, cost, costPerMem,
                    costPerStorage, costPerBw);
```

```
        // 6. Finally, we need to create a PowerDatacenter object.
        Datacenter datacenter = null;
        try {
                datacenter = new Datacenter(name, characteristics,
                                new VmAllocationPolicySimple(hostList), storageList, 0);
        } catch (Exception e) {
                e.printStackTrace();
        }

        return datacenter;
}

// We strongly encourage users to develop their own broker policies, to
// submit vms and cloudlets according
// to the specific rules of the simulated scenario
private static DatacenterBroker1 createBroker() {

        DatacenterBroker1 broker = null;
        try {
                broker = new DatacenterBroker1("Broker");
        } catch (Exception e) {
                e.printStackTrace();
                return null;
        }
        return broker;
}

/**
 * Prints the Cloudlet objects
 *
 * @param list
 *            list of Cloudlets
 */
private static void printCloudletList(List<Cloudlet> list) {
        int size = list.size();
        Cloudlet cloudlet;

        String indent = "    ";
        Log.printLine();
        Log.printLine("========== OUTPUT ==========");
        Log.printLine("Cloudlet ID" + indent + "STATUS" + indent
                        + "Data center ID" + indent + "VM ID" + indent + indent
                        + "Time" + indent + "Start Time" + indent + "Finish Time");

        DecimalFormat dft = new DecimalFormat("###.##");
        for (int i = 0; i < size; i++) {
                cloudlet = list.get(i);



                Log.print(indent + cloudlet.getCloudletId() + indent + indent);

                if (cloudlet.getCloudletStatus() == Cloudlet.SUCCESS) {
                        Log.print("SUCCESS");

                        Log.printLine(indent + indent + cloudlet.getResourceId()
```

```
                                        + indent + indent + indent + cloudlet.getVmId()
                                        + indent + indent + indent
                                        + dft.format(cloudlet.getActualCPUTime()) + indent
                                        + indent + dft.format(cloudlet.getExecStartTime())
                                        + indent + indent + indent
                                        + dft.format(cloudlet.getFinishTime()));
                        }
                    }

            }
}
```

OUTPUTS :



========== OUTPUT of Improved Round Robin Scheduling ==========

Cloudlet ID	STATUS	Data center ID	VM ID	Time	Start Time	Finish Time
0	SUCCESS	2	10	0.91	0.1	1.01
2	SUCCESS	2	12	1.02	0.1	1.12
4	SUCCESS	2	14	1.02	0.1	1.12
1	SUCCESS	2	11	1.02	0.1	1.12
3	SUCCESS	2	13	1.02	0.1	1.12
5	SUCCESS	2	0	1.13	0.1	1.23
6	SUCCESS	2	1	1.13	0.1	1.23
7	SUCCESS	2	2	1.13	0.1	1.23
8	SUCCESS	2	3	1.13	0.1	1.23
9	SUCCESS	2	4	1.24	0.1	1.34
11	SUCCESS	2	6	1.24	0.1	1.34
13	SUCCESS	2	8	1.24	0.1	1.34
10	SUCCESS	2	5	1.24	0.1	1.34
12	SUCCESS	2	7	1.24	0.1	1.34
14	SUCCESS	2	9	1.24	0.1	1.34
15	SUCCESS	2	10	1.18	1.01	2.19
16	SUCCESS	2	11	1.19	1.12	2.31
17	SUCCESS	2	12	1.3	1.12	2.42
19	SUCCESS	2	14	1.3	1.12	2.42
18	SUCCESS	2	13	1.3	1.12	2.42
20	SUCCESS	2	0	1.4	1.23	2.63
21	SUCCESS	2	1	1.51	1.23	2.74
22	SUCCESS	2	2	1.51	1.23	2.74
23	SUCCESS	2	3	1.51	1.23	2.74
24	SUCCESS	2	4	1.51	1.34	2.85
26	SUCCESS	2	6	1.51	1.34	2.85
28	SUCCESS	2	8	1.51	1.34	2.85
25	SUCCESS	2	5	1.51	1.34	2.85
27	SUCCESS	2	7	1.51	1.34	2.85
29	SUCCESS	2	9	1.51	1.34	2.85
30	SUCCESS	2	10	1.45	2.19	3.64
31	SUCCESS	2	11	1.46	2.31	3.77
32	SUCCESS	2	12	1.46	2.42	3.88
34	SUCCESS	2	14	1.57	2.42	3.99
33	SUCCESS	2	13	1.57	2.42	3.99
35	SUCCESS	2	0	1.7	2.63	4.33
36	SUCCESS	2	1	1.7	2.74	4.44
37	SUCCESS	2	2	1.81	2.74	4.55
38	SUCCESS	2	3	1.81	2.74	4.55
39	SUCCESS	2	4	1.81	2.85	4.66
41	SUCCESS	2	6	1.81	2.85	4.66
43	SUCCESS	2	8	1.81	2.85	4.66
40	SUCCESS	2	5	1.81	2.85	4.66
42	SUCCESS	2	7	1.81	2.85	4.66
44	SUCCESS	2	9	1.81	2.85	4.66
45	SUCCESS	2	10	1.73	3.64	5.37
46	SUCCESS	2	11	1.73	3.77	5.5
47	SUCCESS	2	12	1.73	3.88	5.61
48	SUCCESS	2	13	1.73	3.99	5.73
49	SUCCESS	2	14	1.84	3.99	5.84

GA with Improved Round Robin ends....

BUILD SUCCESSFUL (total time: 15 seconds)

========== OUTPUT of Round Robin Scheduling ==========

Cloudlet ID	STATUS	Data center ID	VM ID	Time	Start Time	Finish Time
0	SUCCESS	2	0	1	0.1	1.1
1	SUCCESS	2	2	1	0.1	1.1
2	SUCCESS	2	4	1	0.1	1.1
4	SUCCESS	2	8	1	0.1	1.1
6	SUCCESS	2	12	1	0.1	1.1
3	SUCCESS	2	6	1	0.1	1.1
5	SUCCESS	2	10	1	0.1	1.1
7	SUCCESS	2	14	1	0.1	1.1
14	SUCCESS	3	13	1.13	0.1	1.23
8	SUCCESS	3	1	1.24	0.1	1.34
9	SUCCESS	3	3	1.24	0.1	1.34
10	SUCCESS	3	5	1.24	0.1	1.34
12	SUCCESS	3	9	1.24	0.1	1.34
11	SUCCESS	3	7	1.24	0.1	1.34
13	SUCCESS	3	11	1.24	0.1	1.34
22	SUCCESS	2	14	1.26	1.1	2.36
15	SUCCESS	2	0	1.37	1.1	2.47
16	SUCCESS	2	2	1.37	1.1	2.47
17	SUCCESS	2	4	1.37	1.1	2.47
19	SUCCESS	2	8	1.37	1.1	2.47
21	SUCCESS	2	12	1.37	1.1	2.47
18	SUCCESS	2	6	1.37	1.1	2.47
20	SUCCESS	2	10	1.37	1.1	2.47
29	SUCCESS	3	13	1.4	1.23	2.63
28	SUCCESS	3	11	1.4	1.34	2.75
23	SUCCESS	3	1	1.51	1.34	2.86
24	SUCCESS	3	3	1.51	1.34	2.86
25	SUCCESS	3	5	1.51	1.34	2.86
27	SUCCESS	3	9	1.51	1.34	2.86
26	SUCCESS	3	7	1.51	1.34	2.86
37	SUCCESS	2	14	1.53	2.36	3.89
36	SUCCESS	2	12	1.54	2.47	4.01
30	SUCCESS	2	0	1.65	2.47	4.12
31	SUCCESS	2	2	1.65	2.47	4.12
32	SUCCESS	2	4	1.65	2.47	4.12
34	SUCCESS	2	8	1.65	2.47	4.12
33	SUCCESS	2	6	1.65	2.47	4.12
35	SUCCESS	2	10	1.65	2.47	4.12
44	SUCCESS	3	13	1.66	2.63	4.29
43	SUCCESS	3	11	1.68	2.75	4.42
42	SUCCESS	3	9	1.69	2.86	4.55
38	SUCCESS	3	1	1.8	2.86	4.66
39	SUCCESS	3	3	1.8	2.86	4.66
40	SUCCESS	3	5	1.8	2.86	4.66
41	SUCCESS	3	7	1.8	2.86	4.66
49	SUCCESS	2	8	1.83	4.12	5.95
45	SUCCESS	2	0	1.94	4.12	6.06
46	SUCCESS	2	2	1.94	4.12	6.06
47	SUCCESS	2	4	1.94	4.12	6.06
48	SUCCESS	2	6	1.94	4.12	6.06

Round Robin ends....

BUILD SUCCESSFUL (total time: 12 seconds)

Explore workflow simulator and run workflow application redesign the code with some optimization algorithm like ACO, PSO.

Objective: To learn simulating Evolutionary algorithm in cloud environment.

Task: Install simulator and perform ACO and PSO algorithm in cloud environment

Introduction

Workflow management systems (WFMS) enable automated and seamless execution of workflows. It allows users to define and model workflows, set their deadline and bud- get limitations, and the environments in which they wish to execute. The WFMS then evaluates these inputs and executes them within the defined constraints. The prominent components of a typical cloud WFMS are :

10. The workflow portal is used to model and define abstract workflows, i.e., tasks and their dependencies.

11. The workflow enactment engine takes the abstract workflows and parses them using a language parser. Then, the task dispatcher analyzes the dependencies and dispatches the ready tasks to the scheduler. The scheduler, based on the defined scheduling algorithms, schedules the workflow task onto a resource.

12. The resource broker interfaces with the infrastructure layer and provides a unified view to the enactment engine. The resource broker communicates with compute services to provide the desired resource.

13. The directory and catalogue services house information about data objects, the application and the compute resources. This information is used by the enactment engine, and the resource broker to make critical decisions.

Procedure

1. Install wokflow simulator from github.

2. Modify the workflow scheduling by applying ACO algorithm.

3. Similarly, Modify the workflow scheduling by applying PSO algorithm.

ACO (Ant Colony Optimization)

In the ant colony optimization algorithms, an artificial ant is a simple computational agent that searches for good solutions to a given optimization problem. To apply an ant colony algorithm, the optimization problem needs to be converted into the problem of finding

the shortest path on a weighted graph. In the first step of each iteration, each ant stochastically constructs a solution, i.e. the order in which the edges in the graph should be followed. In the second step, the paths found by the different ants are compared. The last step consists of updating the pheromone levels on each edge.

```
procedure ACO_MetaHeuristic is
   while not_termination do
      generateSolutions()
      daemonActions()
      pheromoneUpdate()
   repeat
end procedure
```

Code

```
package aco;

/**
 *
 * @author Admin
 */
import java.text.DecimalFormat;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.LinkedList;
import java.util.List;

import org.cloudbus.cloudsim.Cloudlet;
import org.cloudbus.cloudsim.CloudletSchedulerSpaceShared;
import org.cloudbus.cloudsim.Datacenter;
import org.cloudbus.cloudsim.DatacenterBroker;
import org.cloudbus.cloudsim.DatacenterCharacteristics;
import org.cloudbus.cloudsim.Host;
import org.cloudbus.cloudsim.Log;
import org.cloudbus.cloudsim.Pe;
import org.cloudbus.cloudsim.Storage;
import org.cloudbus.cloudsim.UtilizationModel;
import org.cloudbus.cloudsim.UtilizationModelFull;
import org.cloudbus.cloudsim.Vm;
import org.cloudbus.cloudsim.VmAllocationPolicySimple;
import org.cloudbus.cloudsim.VmSchedulerTimeShared;
import org.cloudbus.cloudsim.core.CloudSim;
import org.cloudbus.cloudsim.provisioners.BwProvisionerSimple;
import org.cloudbus.cloudsim.provisioners.PeProvisionerSimple;
import org.cloudbus.cloudsim.provisioners.RamProvisionerSimple;

public class MyAllocationTest {
        private static List<Cloudlet> cloudletList;
        private static int cloudletNum=20;
```

```
	private static List<Vm> vmList;
	private static int vmNum=2;

	public static void main(String args[]){
		Log.printLine("Starting ACO Example...");
		int num_user=1;
		Calendar calendar=Calendar.getInstance();
		boolean trace_flag=false;

		CloudSim.init(num_user, calendar, trace_flag);

		Datacenter datacenter0=createDatacenter("Datacenter_0");

		DatacenterBroker broker=createBroker();
		int brokerId=broker.getId();

		int vmid=0;
		int []mipss=new int[]{278,289,132,209,286};
		long size=10000;
		int ram=2048;
		long bw=10000;
		int pesNumber=1;
		String vmm="xen";

		vmList=new ArrayList<Vm>();
		for(int i=0;i<vmNum;i++){
			vmList.add(new Vm(vmid, brokerId, mipss[i], pesNumber, ram, bw, size, vmm,
new CloudletSchedulerSpaceShared()));
			vmid++;
		}
		broker.submitVmList(vmList);

		int id=0;
		long[] lengths=new
long[]{19365,49809,30218,44157,16754,18336,20045,31493,30727,2341,31017,32100,2145,12343,1456
7,23452,23121,1345,7643,1234,2345,4321,2113,5674,5432,2123,4321,1231,2131,2455};
		long fileSize=300;
		long outputSize=300;
		UtilizationModel model=new UtilizationModelFull();

		cloudletList=new ArrayList<Cloudlet>();
		for (int i = 0; i < cloudletNum; i++) {
			Cloudlet cloudlet=new Cloudlet(id, lengths[i], pesNumber, fileSize, outputSize,
model, model, model);
			cloudlet.setUserId(brokerId);
			cloudletList.add(cloudlet);
			id++;
		}
		broker.submitCloudletList(cloudletList);
		//broker.bind(5,50);//bind(ÂìÒÏ¸öÊý£¬µü´ú´ÎÊý)

		//broker.bindCloudletsToVmsSimple();
```

```
		CloudSim.startSimulation();

		List<Cloudlet> newList=broker.getCloudletReceivedList();
		CloudSim.stopSimulation();
		printCloudletList(newList);
		Log.printLine("ACO finished!");
	}

	public static Datacenter createDatacenter(String name){
		List<Host> hostList=new ArrayList<Host>();

		int mips=1000;
		int hostId=0;
		int ram=2048;
		long storage=1000000;
		int bw=10000;
		for(int i=0;i<vmNum;i++){
			List<Pe> peList=new ArrayList<Pe>();
			peList.add(new Pe(0,new PeProvisionerSimple(mips)));
			hostList.add(
					new Host(
							hostId,
							new RamProvisionerSimple(ram),
							new BwProvisionerSimple(bw),
							storage,
							peList,
							new VmSchedulerTimeShared(peList)));
			hostId++;
		}

		String arch="x86";
		String os="Linux";
		String vmm="Xen";
		double time_zone=10.0;
		double cost=3.0;
		double costPerMcm=0.05;
		double costPerStorage=0.001;
		double costPerBw=0.0;
		LinkedList<Storage> storageList=new LinkedList<Storage>();

		DatacenterCharacteristics characteristics=new DatacenterCharacteristics(arch, os, vmm,
hostList, time_zone, cost, costPerMcm, costPerStorage, costPerBw);
		Datacenter datacenter=null;
		try {
			datacenter=new Datacenter(name, characteristics, new
VmAllocationPolicySimple(hostList), storageList, 0);
		} catch (Exception e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
		return datacenter;
	}
```

```
	private static DatacenterBroker createBroker(){
		DatacenterBroker broker=null;
		try {
			broker=new DatacenterBroker("Broker");
		} catch (Exception e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
		return broker;
	}

	private static void printCloudletList(List<Cloudlet> list){
		int size=list.size();
		Cloudlet cloudlet;
		String indent="    ";
		Log.printLine();
		Log.printLine("========OUTPUT========");
		Log.printLine("Cloudlet ID"+indent+"STATUS"+indent+"Datacenter ID"+indent+"VM
ID"+indent+"Time"+indent+"Start Time"+indent+"Finish Time");
		DecimalFormat dft=new DecimalFormat("###.##");
		for(int i=0;i<size;i++){
			cloudlet=list.get(i);
			Log.print(indent+cloudlet.getCloudletId()+indent+indent);
			if (cloudlet.getCloudletStatus()==Cloudlet.SUCCESS) {
				Log.print("SUCESSS");

	Log.printLine(indent+indent+cloudlet.getResourceId()+indent+indent+indent

	+cloudlet.getVmId()+indent+indent+dft.format(cloudlet.getActualCPUTime())+
					indent+indent+dft.format(cloudlet.getExecStartTime())+
					indent+indent+dft.format(cloudlet.getFinishTime()));

			}
		}
	}
}
```

Particle Swarm Optimization (PSO)

PSO is a computational method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. It solves a problem by having a population of candidate solutions, here dubbed particles, and moving these particles around in the search-space according to simple mathematical formulae over the particle's position and velocity. Each particle's movement is influenced by its local best known position, but is also guided toward the best known positions in the search-space, which are updated as better positions are found by other particles. This is expected to move the swarm toward the best solutions.

Code

```
package PSO;
import org.cloudbus.cloudsim.*;
import org.cloudbus.cloudsim.core.CloudSim;
import utils.Constants;
import utils.DatacenterCreator;
import utils.GenerateMatrices;
import java.text.DecimalFormat;
import java.util.*;

public class PSO_Scheduler {

    private static List<Cloudlet> cloudletList;
    private static List<Vm> vmList;
    private static Datacenter[] datacenter;
    private static PSO PSOSchedularInstance; //call pso.java
    private static double mapping[]; //function
    private static double[][] commMatrix; //calculation
    private static double[][] execMatrix;

    private static List<Vm> createVM(int userId, int vms) {
        //Creates a container to store VMs. This list is passed to the broker later
        LinkedList<Vm> list = new LinkedList<Vm>();

        //VM Parameters
        long size = 10000; //image size (MB)
        int ram = 512; //vm memory (MB)
        int mips = 250;
        long bw = 1000;
        int pesNumber = 1; //number of cpus
        String vmm = "Xen"; //VMM name

        //create VMs
        Vm[] vm = new Vm[vms];

        for (int i = 0; i < vms; i++) {
            vm[i] = new Vm(datacenter[i].getId(), userId, mips, pesNumber, ram, bw, size, vmm, new
CloudletSchedulerSpaceShared());
            list.add(vm[i]);
        }

        return list;
    }

    private static List<Cloudlet> createCloudlet(int userId, int cloudlets, int idShift) {
        LinkedList<Cloudlet> list = new LinkedList<Cloudlet>();

        //cloudlet parameters
        long fileSize = 300;
        long outputSize = 300;
        int pesNumber = 1;
        UtilizationModel utilizationModel = new UtilizationModelFull();
```

```
        Cloudlet[] cloudlet = new Cloudlet[cloudlets];

        for (int i = 0; i < cloudlets; i++) {
            int dcId = (int) (mapping[i]);
            long length = (long) (1e3 * (commMatrix[i][dcId] + execMatrix[i][dcId]));
            cloudlet[i] = new Cloudlet(idShift + i, length, pesNumber, fileSize, outputSize,
utilizationModel, utilizationModel, utilizationModel);
            cloudlet[i].setUserId(userId);
            list.add(cloudlet[i]);
        }

        return list;
    }

    public static void main(String[] args) {
        Log.printLine("Starting PSO Scheduler...");

        new GenerateMatrices();
        commMatrix = GenerateMatrices.getCommMatrix(); // call the generate matrices class
        execMatrix = GenerateMatrices.getExecMatrix();
        PSOSchedularInstance = new PSO();
        mapping = PSOSchedularInstance.run();

        try {
            int num_user = 1;   // number of grid users
            Calendar calendar = Calendar.getInstance();
            boolean trace_flag = false;  // mean trace events

            CloudSim.init(num_user, calendar, trace_flag);

            // Second step: Create Datacenters
            datacenter = new Datacenter[Constants.NO_OF_DATA_CENTERS];
            for (int i = 0; i < Constants.NO_OF_DATA_CENTERS; i++) {
                datacenter[i] = DatacenterCreator.createDatacenter("Datacenter_" + i);
            }

            //Third step: Create Broker
            PSODatacenterBroker broker = createBroker("Broker_0");
            int brokerId = broker.getId();

            //Fourth step: Create VMs and Cloudlets and send them to broker
            vmList = createVM(brokerId, Constants.NO_OF_DATA_CENTERS);
            cloudletList = createCloudlet(brokerId, Constants.NO_OF_TASKS, 0);

            // mapping our dcIds to cloudsim dcIds
            HashSet<Integer> dcIds = new HashSet<>();
            HashMap<Integer, Integer> hm = new HashMap<>();
            for (Datacenter dc : datacenter) {
                if (!dcIds.contains(dc.getId()))
                    dcIds.add(dc.getId());
            }
            Iterator<Integer> it = dcIds.iterator();
```

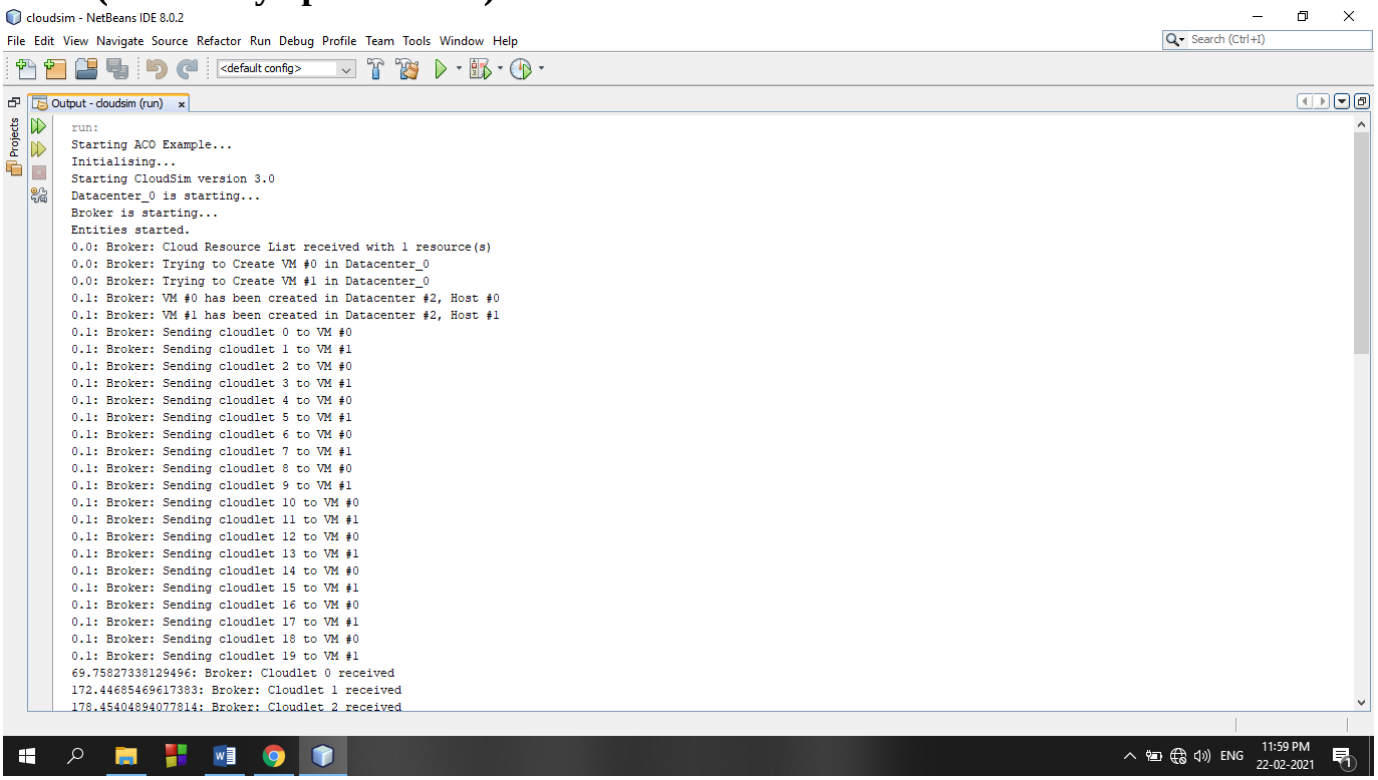
```
            for (int i = 0; i < mapping.length; i++) {
                if (hm.containsKey((int) mapping[i])) continue;
                hm.put((int) mapping[i], it.next());
            }
            for (int i = 0; i < mapping.length; i++)
                mapping[i] = hm.containsKey((int) mapping[i]) ? hm.get((int) mapping[i]) : mapping[i];

            broker.submitVmList(vmList);
            broker.setMapping(mapping);
            broker.submitCloudletList(cloudletList);


            // Fifth step: Starts the simulation
            CloudSim.startSimulation();

            List<Cloudlet> newList = broker.getCloudletReceivedList();

            CloudSim.stopSimulation();

            printCloudletList(newList);

            Log.printLine(PSO_Scheduler.class.getName() + " finished!");
        } catch (Exception e) {
            e.printStackTrace();
            Log.printLine("The simulation has been terminated due to an unexpected error");
        }
    }

    private static PSODatacenterBroker createBroker(String name) throws Exception {
        return new PSODatacenterBroker(name);
    }

    /**
     * Prints the Cloudlet objects
     *
     * @param list list of Cloudlets
     */
    private static void printCloudletList(List<Cloudlet> list) {
        int size = list.size();
        Cloudlet cloudlet;

        String indent = "    ";
        Log.printLine();
        Log.printLine("========== OUTPUT ==========");
        Log.printLine("Cloudlet ID" + indent + "STATUS" +
                indent + "Data center ID" +
                indent + "VM ID" +
                indent + indent + "Time" +
                indent + "Start Time" +
                indent + "Finish Time");

        double mxFinishTime = 0;
```

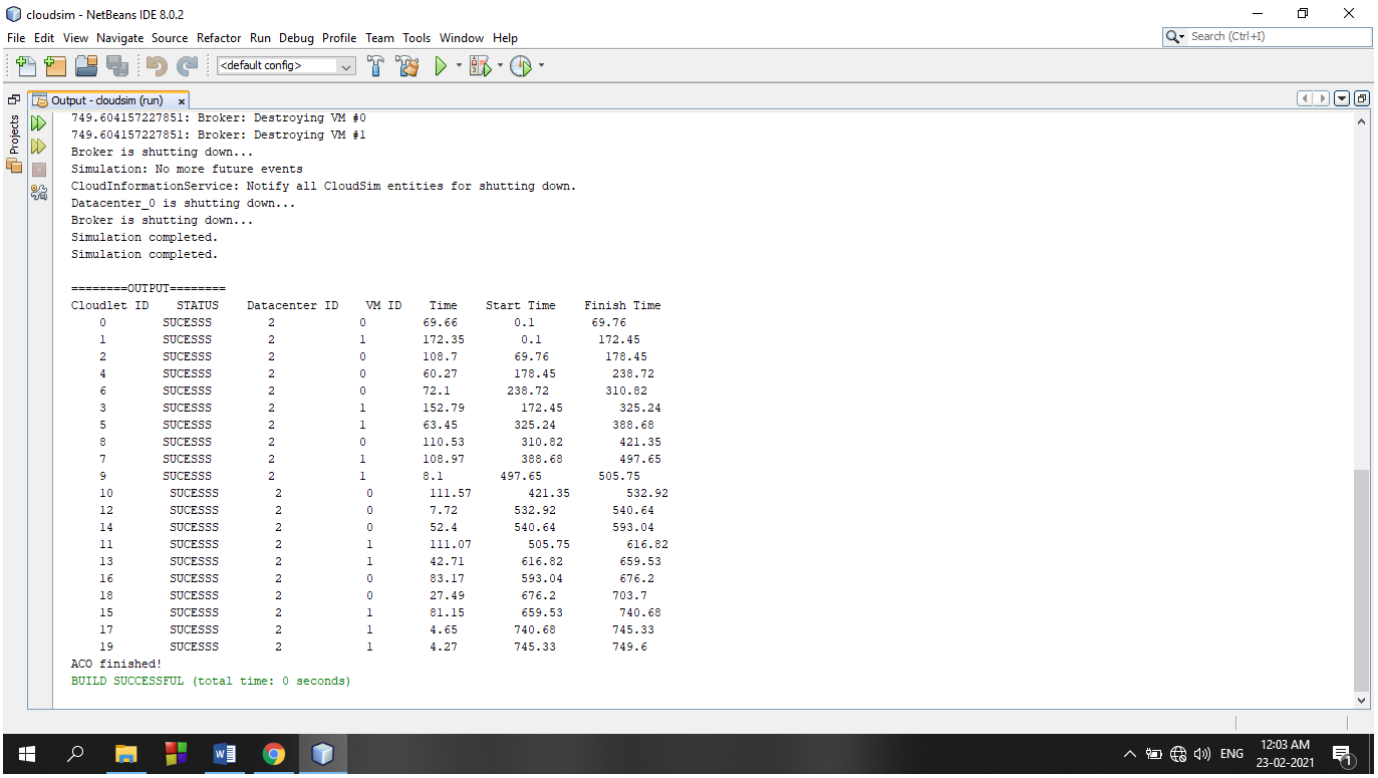
```
        DecimalFormat dft = new DecimalFormat("###.##");
        dft.setMinimumIntegerDigits(2);
        for (int i = 0; i < size; i++) {
            cloudlet = list.get(i);
            Log.print(indent + dft.format(cloudlet.getCloudletId()) + indent + indent);

            if (cloudlet.getCloudletStatus() == Cloudlet.SUCCESS) {
                Log.print("SUCCESS");
                Log.printLine(indent + indent + dft.format(cloudlet.getResourceId()) +
                        indent + indent + indent + dft.format(cloudlet.getVmId()) +
                        indent + indent + dft.format(cloudlet.getActualCPUTime()) +
                        indent + indent + dft.format(cloudlet.getExecStartTime()) +
                        indent + indent + indent + dft.format(cloudlet.getFinishTime()));
            }
           // mxFinishTime = Math.max(mxFinishTime, cloudlet.getFinishTime());
        }
        //Log.printLine(mxFinishTime);
        PSOSchedularInstance.printBestFitness();
    }

}
```

OUTPUT

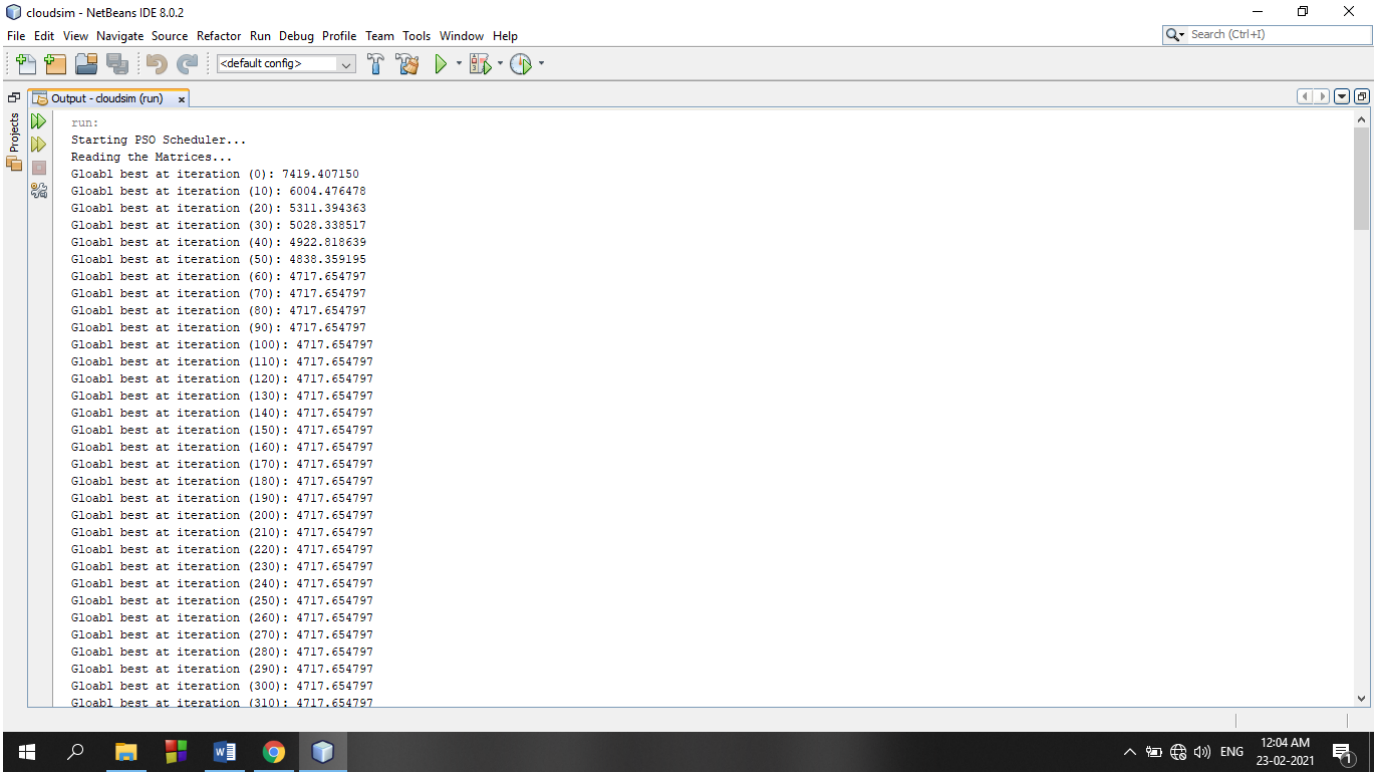
Here are some of the screenshots attached

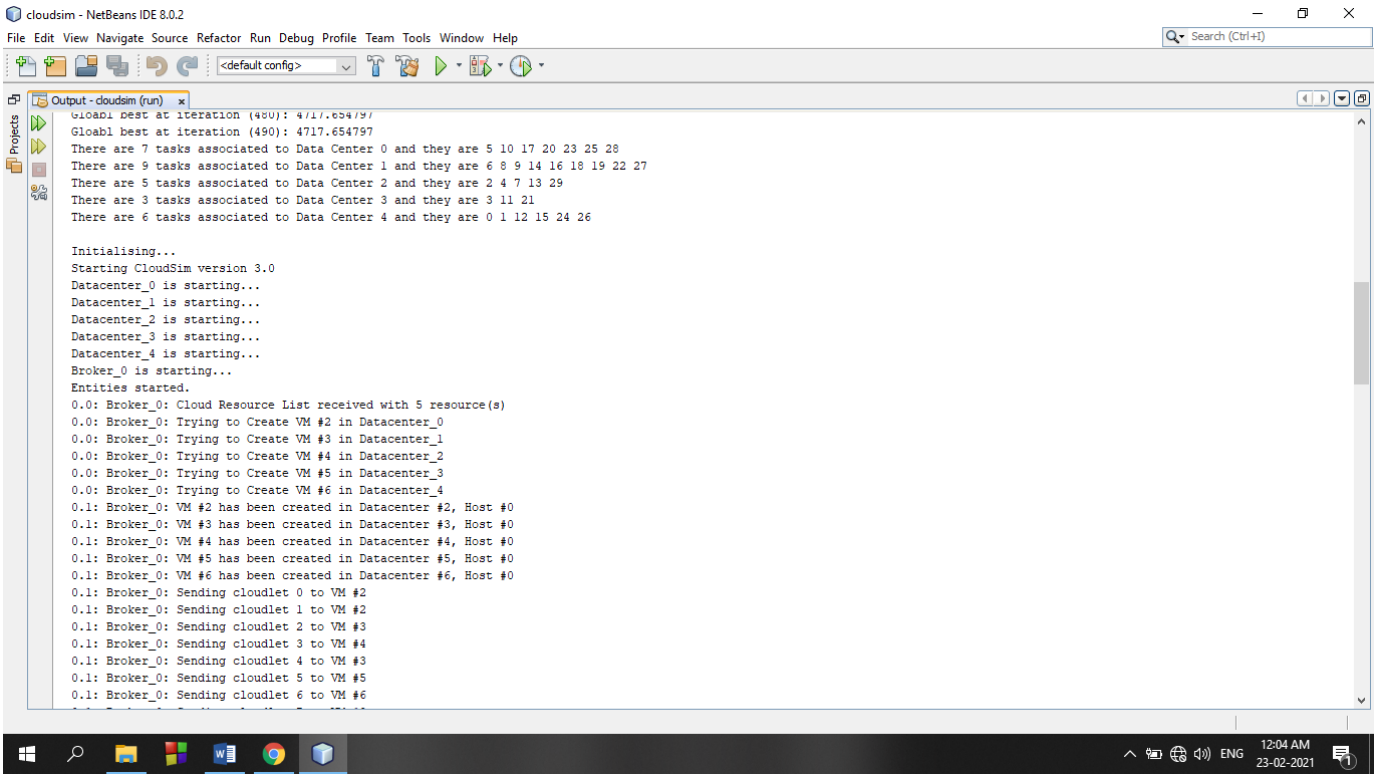
ACO (ant colony optimization)

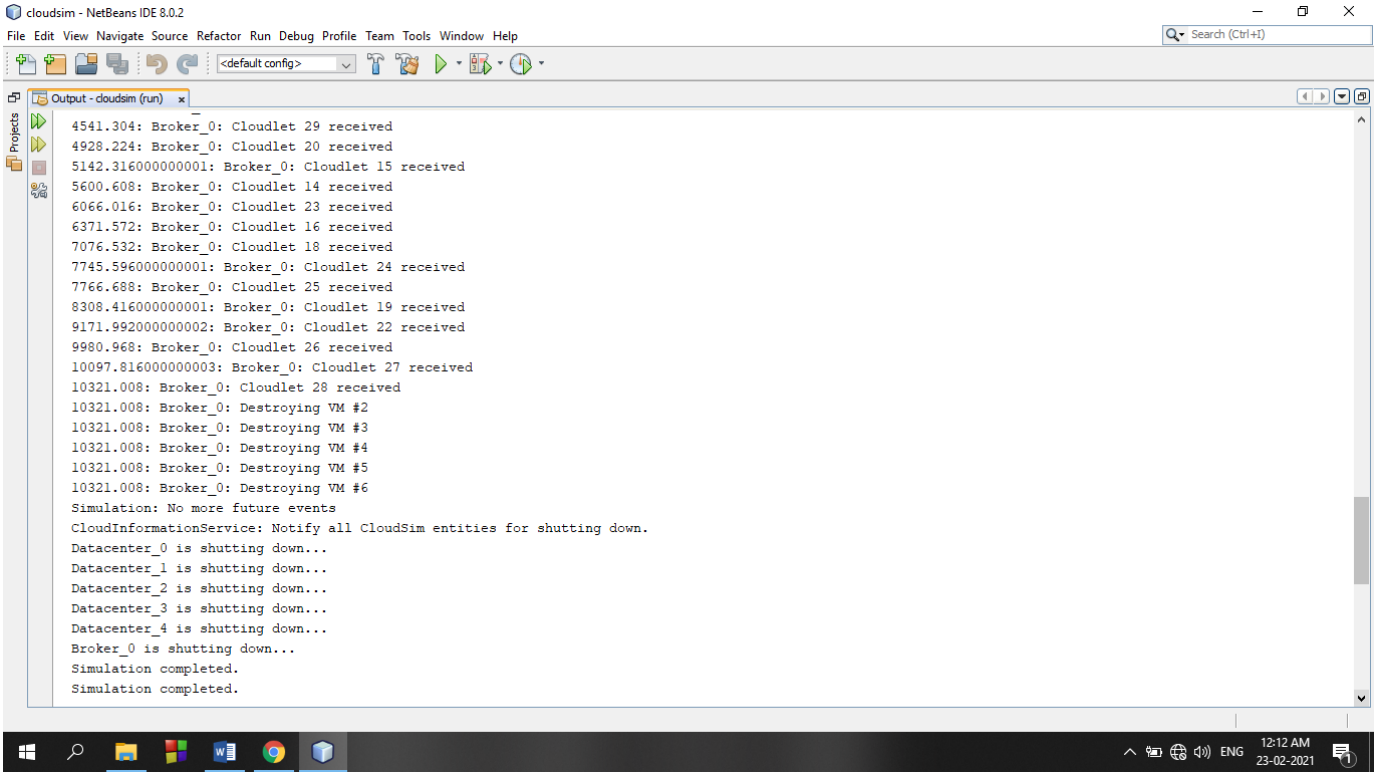


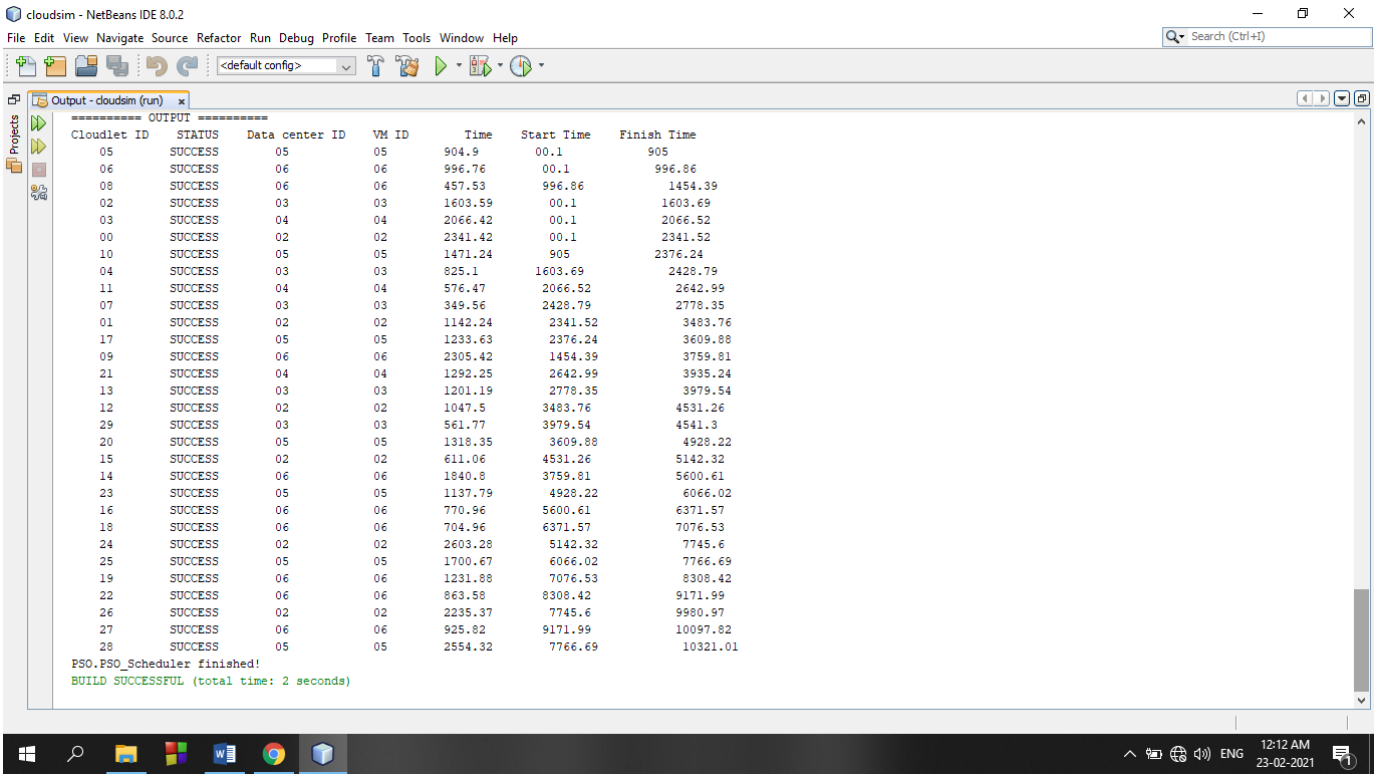


Particle Swarm Optimization (PSO)




cloudsim - NetBeans IDE 8.0.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Output - cloudsim (run)
Global best at iteration (480): 4717.654797
Global best at iteration (490): 4717.654797
There are 7 tasks associated to Data Center 0 and they are 5 10 17 20 23 25 28
There are 9 tasks associated to Data Center 1 and they are 6 8 9 14 16 18 19 22 27
There are 5 tasks associated to Data Center 2 and they are 2 4 7 13 29
There are 3 tasks associated to Data Center 3 and they are 3 11 21
There are 6 tasks associated to Data Center 4 and they are 0 1 12 15 24 26
Initialising...
Starting CloudSim version 3.0
Datacenter_0 is starting...
Datacenter_1 is starting...
Datacenter_2 is starting...
Datacenter_3 is starting...
Datacenter_4 is starting...
Broker_0 is starting...
Entities started.
0.0: Broker_0: Cloud Resource List received with 5 resource(s)
0.0: Broker_0: Trying to Create VM #2 in Datacenter_0
0.0: Broker_0: Trying to Create VM #3 in Datacenter_1
0.0: Broker_0: Trying to Create VM #4 in Datacenter_2
0.0: Broker_0: Trying to Create VM #5 in Datacenter_3
0.0: Broker_0: Trying to Create VM #6 in Datacenter_4
0.1: Broker_0: VM #2 has been created in Datacenter #2, Host #0
0.1: Broker_0: VM #3 has been created in Datacenter #3, Host #0
0.1: Broker_0: VM #4 has been created in Datacenter #4, Host #0
0.1: Broker_0: VM #5 has been created in Datacenter #5, Host #0
0.1: Broker_0: VM #6 has been created in Datacenter #6, Host #0
0.1: Broker_0: Sending cloudlet 0 to VM #2
0.1: Broker_0: Sending cloudlet 1 to VM #2
0.1: Broker_0: Sending cloudlet 2 to VM #3
0.1: Broker_0: Sending cloudlet 3 to VM #4
0.1: Broker_0: Sending cloudlet 4 to VM #3
0.1: Broker_0: Sending cloudlet 5 to VM #5
0.1: Broker_0: Sending cloudlet 6 to VM #6
ENG
12:04 AM
23-02-2021


cloudsim - NetBeans IDE 8.0.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Output - cloudsim (run)
4541.304: Broker_0: Cloudlet 29 received
4928.224: Broker_0: Cloudlet 20 received
5142.316000000001: Broker_0: Cloudlet 15 received
5600.608: Broker_0: Cloudlet 14 received
6066.016: Broker_0: Cloudlet 23 received
6371.572: Broker_0: Cloudlet 16 received
7076.532: Broker_0: Cloudlet 18 received
7745.596000000001: Broker_0: Cloudlet 24 received
7766.688: Broker_0: Cloudlet 25 received
8308.416000000001: Broker_0: Cloudlet 19 received
9171.992000000002: Broker_0: Cloudlet 22 received
9980.968: Broker_0: Cloudlet 26 received
10097.816000000003: Broker_0: Cloudlet 27 received
10321.008: Broker_0: Cloudlet 28 received
10321.008: Broker_0: Destroying VM #2
10321.008: Broker_0: Destroying VM #3
10321.008: Broker_0: Destroying VM #4
10321.008: Broker_0: Destroying VM #5
10321.008: Broker_0: Destroying VM #6
Simulation: No more future events
CloudInformationService: Notify all CloudSim entities for shutting down.
Datacenter_0 is shutting down...
Datacenter_1 is shutting down...
Datacenter_2 is shutting down...
Datacenter_3 is shutting down...
Datacenter_4 is shutting down...
Broker_0 is shutting down...
Simulation completed.
Simulation completed.
ENG
12:12 AM
23-02-2021


cloudsim - NetBeans IDE 8.0.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
Output - cloudsim (run)
Projects
========== OUTPUT ==========
Cloudlet ID STATUS Data center ID VM ID Time Start Time Finish Time
05 SUCCESS 05 05 904.9 00.1 905
06 SUCCESS 06 06 996.76 00.1 996.86
08 SUCCESS 06 06 457.53 996.86 1454.39
02 SUCCESS 03 03 1603.59 00.1 1603.69
03 SUCCESS 04 04 2066.42 00.1 2066.52
00 SUCCESS 02 02 2341.42 00.1 2341.52
10 SUCCESS 05 05 1471.24 905 2376.24
04 SUCCESS 03 03 825.1 1603.69 2428.79
11 SUCCESS 04 04 576.47 2066.52 2642.99
07 SUCCESS 03 03 349.56 2428.79 2778.35
01 SUCCESS 02 02 1142.24 2341.52 3483.76
17 SUCCESS 05 05 1233.63 2376.24 3609.88
09 SUCCESS 06 06 2305.42 1454.39 3759.81
21 SUCCESS 04 04 1292.25 2642.99 3935.24
13 SUCCESS 03 03 1201.19 2778.35 3979.54
12 SUCCESS 02 02 1047.5 3483.76 4531.26
29 SUCCESS 03 03 561.77 3979.54 4541.3
20 SUCCESS 05 05 1318.35 3609.88 4928.22
15 SUCCESS 02 02 611.06 4531.26 5142.32
14 SUCCESS 06 06 1840.8 3759.81 5600.61
23 SUCCESS 05 05 1137.79 4928.22 6066.02
16 SUCCESS 06 06 770.96 5600.61 6371.57
18 SUCCESS 06 06 704.96 6371.57 7076.53
24 SUCCESS 02 02 2603.28 5142.32 7745.6
25 SUCCESS 05 05 1700.67 6066.02 7766.69
19 SUCCESS 06 06 1231.88 7076.53 8308.42
22 SUCCESS 06 06 863.58 8308.42 9171.99
26 SUCCESS 02 02 2235.37 7745.6 9980.97
27 SUCCESS 06 06 925.82 9171.99 10097.82
28 SUCCESS 05 05 2554.32 7766.69 10321.01
PSO.PSO_Scheduler finished!
BUILD SUCCESSFUL (total time: 2 seconds)
ENG
12:12 AM
23-02-2021