

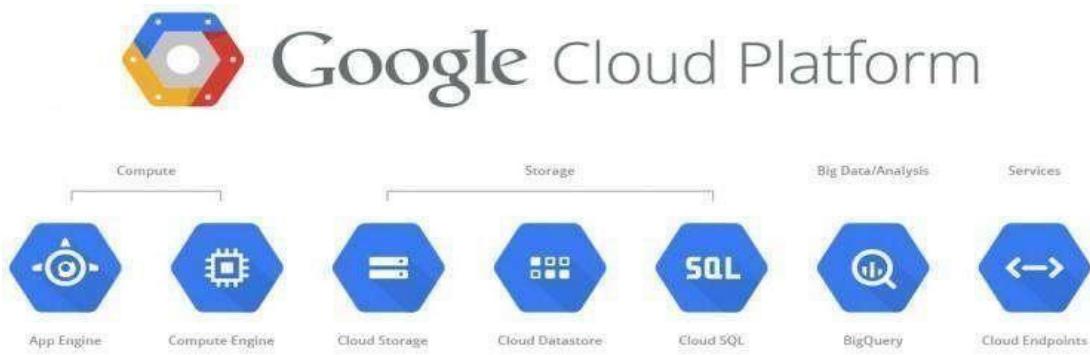
**Ex No.
1 & 2**

Install Google App Engine. Create hello world app and other simple web applications using python/java. Use GAE launcher to launch the web applications

Introduction

○ Google Cloud Platform (GCP)

- **Google Cloud Platform (GCP)**, offered by Google, is a suite of cloud computing services that runs on the same infrastructure that Google uses internally for its end-user products, such as Google Search, Gmail, file storage, and YouTube.
- Alongside a set of management tools, it provides a series of modular cloud services including computing, data storage, data analytics and machine learning.
- Google Cloud Platform provides infrastructure as a service, platform as a service, and serverless computing environments.



- **Platform as a Service (PaaS)** ○ Cloud computing service which provides a computing platform and a solution stack as a service.
 - Consumer creates the software using tools and/or libraries from the provider.
 - Provider provides the networks, servers, storage, etc.

-  **Google App Engine:**

- Google App Engine was first released as a beta version in April 2008.
- It is a Platform as a Service (PaaS) cloud computing platform for developing and hosting web applications in Google-managed data centers.

- Google's App Engine opens Google's production to any person in the world at no charge.
- Google App Engine is software that facilitates the user to run his web applications on Google infrastructure.
- It is more reliable because failure of any server will not affect either the performance of the end user or the service of the Google.
- It virtualizes applications across multiple servers and data centers.

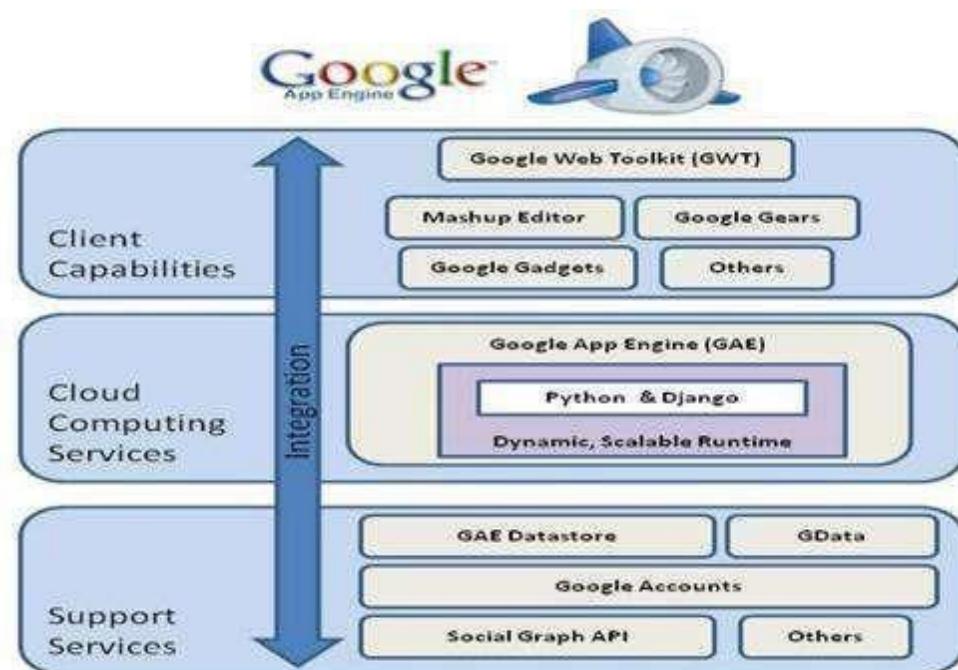
★ Other cloud-based platforms include offerings such as Amazon Web

Services and Microsoft's Azure Services Platform. ○ **Introduction of Google App Engine**

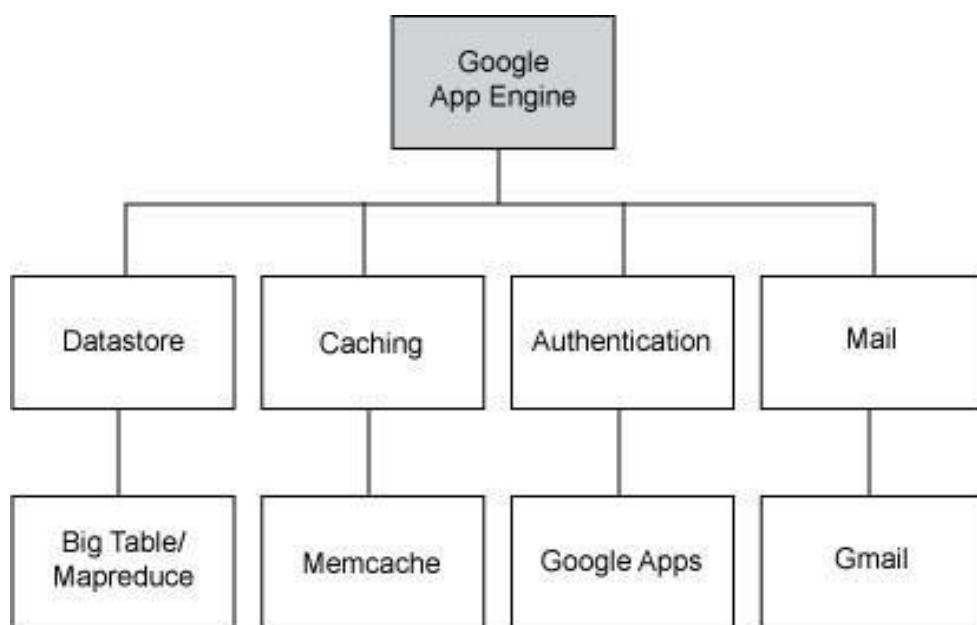
- Google App Engine lets you run your web applications on Google's infrastructure. App Engine applications are easy to build, easy to maintain, and easy to scale as your traffic and data storage needs grow. With App Engine, there are no servers to maintain: You just upload your application, and it's ready to serve your users.
- You can serve your app from your own domain name (such as <https://www.example.com/>) using Google Apps. Or, you can serve your app using a free name on the appspot.com domain. You can share your application with the world, or limit access to members of your organization.
- Google App Engine supports apps written in several programming languages. With App Engine's Java runtime environment, you can build your app using standard Java technologies, including the JVM, Java servlets, and the Java programming language—or any other language using a JVM-based interpreter or compiler, such as JavaScript or Ruby. App Engine also features a dedicated Python runtime environment, which includes a fast Python interpreter and the Python standard library. The Java and Python runtime environments are built to ensure that your application runs quickly, securely, and without interference from other apps on the system.
- With App Engine, you only pay for what you use. There are no set-up costs and no recurring fees. The resources your application uses, such as storage and bandwidth, are measured by the gigabyte, and billed at competitive rates. You control the maximum amounts of resources your app can consume, so it always stays within your budget. App Engine costs nothing to get started. All applications can use up to 500 MB of storage and enough CPU and bandwidth to support an efficient app serving around 5 million page views a month,

absolutely free. When you enable billing for your application, your free limits are raised, and you only pay for resources you use above the free levels. ◉

Architecture of Google App Engine



◉ Features of Google App Engine



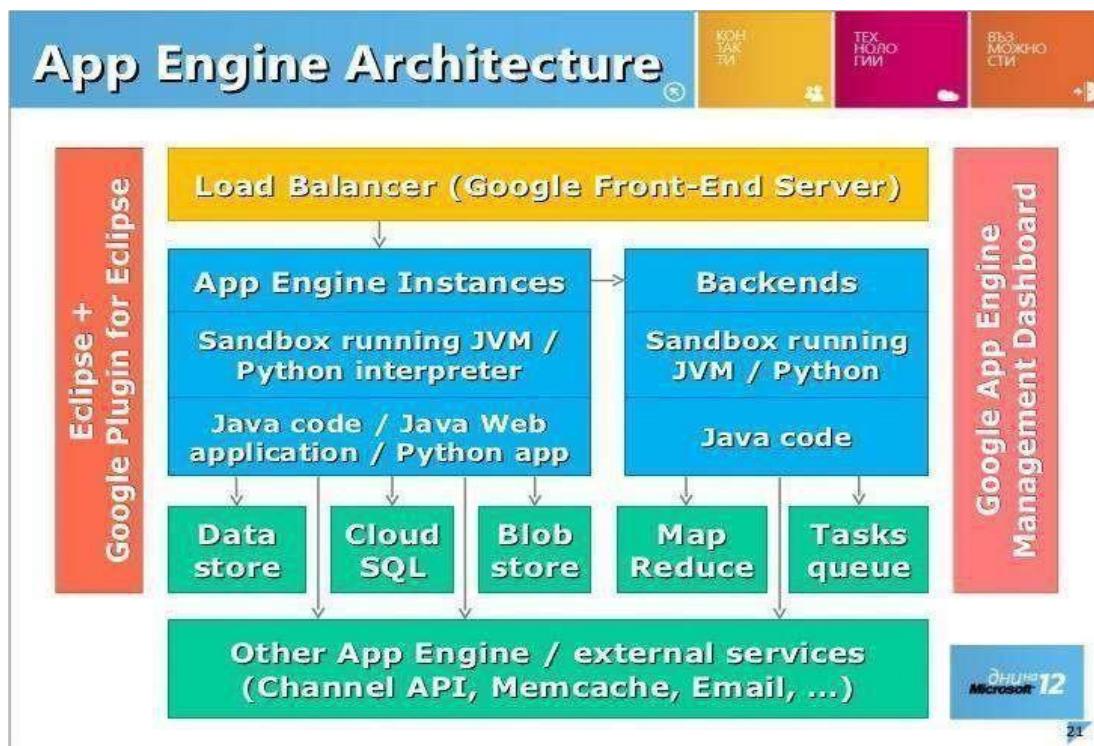
○ GAE Application Environment:

- Google App Engine makes it easy to build an application that runs reliably, even under heavy load and with large amounts of data. App Engine includes the following features:
 - Persistent storage with queries, sorting and transactions
 - Automatic scaling and load balancing
 - APIs for authenticating users and sending email using Google Accounts
 - Task queues for performing work outside of the scope of a web request
 - Scheduled tasks for triggering events at specified times and regular intervals
 - Dynamic web serving, with full support for common web technologies

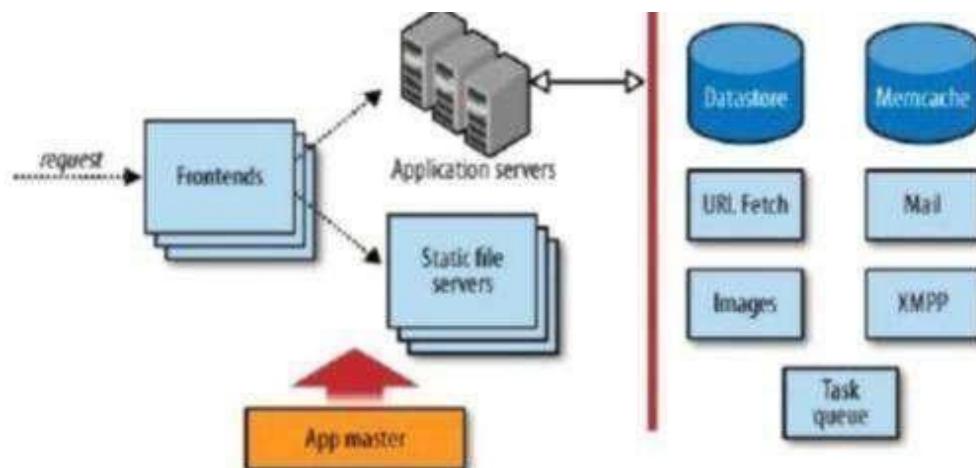
○ Java Runtime Environment

- You can develop your application for the Java runtime environment using common Java web development tools and API standards. Your app interacts with the environment using the Java Servlets standard, and can use common web application technologies such as Java Server Pages
- The Java runtime environment uses Java 6. The App Engine Java SDK supports developing apps using either Java 5 or 6. The environment includes the Java SE Runtime Environment (JRE) 6 platform and libraries. The restrictions of the sandbox environment are implemented in the JVM. An app can use any JVM byte code or library feature, as long as it does not exceed the sandbox restrictions. For instance, byte code that attempts to open a socket or write to a file will throw a runtime exception.
- Your app accesses most App Engine services using Java standard APIs. For the App Engine data store, the Java SDK includes implementations of the Java Data Objects (JDO) and Java Persistence API (JPA) interfaces. Your app can use the JavaMail API to send email messages with the App Engine Mail service. The java.net HTTP APIs access the App Engine URL fetch service.
- App Engine also includes low-level APIs for its services to implement additional adapters, or to use directly from the application. See the documentation for the data store, memcache, URL fetch, mail, images and Google Accounts APIs. Typically, Java developers use the Java programming language and APIs to implement web applications for the JVM. With the use

of JVM-compatible compilers or interpreters, you can also use other languages to develop web applications, such as JavaScript, Ruby.



○ Workflow of Google App Engine



Step1 : Login to www.cloud.google.com

Google search results for "google cloud platform". The top result is an Ad for Google Cloud Platform (GCP) from Google Cloud. The card for Google Cloud Platform includes the following information:

- Google Cloud Platform (GCP) - Google Cloud**
- Modernise your apps and services on **Google Cloud's** scalable infrastructure. Reliable and high performance **cloud** services. Start your free trial today. Get the support you need. Use our worldwide network. Focus on your product. No-ops apps. Mix & match services.
- Create free account**
- Learn & build with our Free Tier & \$300 free credit!
- Compute Engine**
- Run high-performance scalable VMs on Google's infrastructure
- Google Cloud Storage**
- Store & manage objects across four storage classes with a unified API.
- Pricing calculator**
- Enter what you need to run in the Cloud & we'll calculate the cost.

The card also includes a link to <https://cloud.google.com/gcp> and a note that it was Launched: April 7, 2008; 14 years ago.

Step2 : Goto Console

Google Cloud homepage. Key elements include:

- Build what's next. Better software. Faster.**
- A bulleted list of benefits:
 - ✓ Use Google's core infrastructure, data analytics, and machine learning
 - ✓ Secure and fully featured for all enterprises
 - ✓ Committed to open source and industry-leading price-performance
- Call-to-action buttons: [Get started for free](#) and [Contact sales](#)
- A floating chat window asking, "Hi there! What brings you to Google Cloud today?"

Step 3 : Google Cloud Platform is shown

The screenshot shows the Google Cloud Platform 'Welcome' page. At the top, there's a navigation bar with links like 'Internet', 'Django', 'Dashboard : Func...', 'GitHub', 'Tata Sky – Live TV', 'YouTube', 'HackerRank', 'InfyTQ', 'Learner Dashboard...', 'OQE449802346', and 'Our Story | ZS'. Below the navigation bar is a search bar and a user profile icon. The main area features a 'Welcome' section with a colorful cloud icon and the text 'Welcome'. It says 'You're working in Project01' and provides 'Project number: 448952074324' and 'Project ID: project01-343808'. There are buttons for 'Create a VM', 'Run a query in BigQuery', 'Create a GKE cluster', and 'Create a storage bucket'. Below these buttons is a 'Quick access' section with a search bar and a toolbar with icons for various Google services. The status bar at the bottom shows 'Waiting for cloudbusersettings-pa.clients6.google.com...' and system information like 'Privacy Policy · Terms of Service', '12:39 09-05-2022', and a battery level of 12%.

Step 4 : Click Dashboard in the Google Cloud Platform

The screenshot shows the Google Cloud Platform 'Dashboard' page. The top navigation bar is identical to the previous screenshot. The main dashboard area has three tabs: 'DASHBOARD' (which is selected), 'ACTIVITY', and 'RECOMMENDATIONS'. On the left, there's a 'Project info' sidebar with details: 'Project name: Project01', 'Project number: 448952074324', and 'Project ID: project01-343808'. It also has a 'ADD PEOPLE TO THIS PROJECT' button and a link to 'Go to project settings'. In the center, there's a 'API APIs' section showing a line chart for 'Requests (requests/sec)' over time (12:00 to 12:45). The chart shows values of 1.0, 0.8, 0.6, 0.4, 0.2, and 0. A note says 'No data is available for the selected time frame.' Below the chart is a link to 'Go to APIs overview'. On the right, there are two sections: 'Google Cloud Platform status' (showing 'All services normal') and 'Monitoring' (with options to 'Create my dashboard', 'Set up alerting policies', 'Create uptime checks', and 'View all dashboards'). The status bar at the bottom shows 'DISMISS ACTIVATE' buttons, a search bar, a toolbar with icons, and system information like '12:50 09-05-2022' and a battery level of 12%.

Step 5 : Dashboard in the Google Cloud Platform

The screenshot shows the Google Cloud Platform dashboard for project 'Project01'. The left sidebar includes 'Cloud overview', 'Recent', and 'View all products' under 'PINNED' categories. The main area displays 'API APIs' with a chart showing requests per second over time, indicating no data available for the selected frame. To the right, there are sections for 'Google Cloud Platform status' (All services normal), 'Monitoring' (Create my dashboard, Set up alerting policies, Create uptime checks), and 'View all dashboards'. The bottom taskbar shows the Windows Start button, a search bar, and various pinned icons.

Step 6 : Click New Project and give unique Project Name.

Example : pravin-cloud-project

The screenshot shows the 'New Project' dialog in the Google Cloud Platform. It prompts the user to enter a project name ('Pravin-Cloud-Project') and location ('No organisation'). A note states that the project ID will be based on the project name and cannot be changed later. Buttons for 'CREATE' and 'CANCEL' are at the bottom. The background shows the same dashboard elements as the previous screenshot.

Step 7 : Google App Engine is initiated

The screenshot shows a web browser window for the Google Cloud Platform App Engine dashboard. The URL is `console.cloud.google.com/appengine/start?project=pravin-cloud-project`. The main content area displays a "Welcome to App Engine" message with the subtext "Build scalable apps in any language on Google's infrastructure". Below this is a blue "CREATE APPLICATION" button. To the left, a sidebar titled "App Engine" lists various management options: Dashboard, Services, Versions, Instances, Task queues, Cron jobs, Security scans, Firewall rules, Quotas, and Release notes. The "Dashboard" option is currently selected. On the right side, there is a "LEARN Home" section with links to "App Engine overview", "Choosing an App Engine environment", "Structuring web services in App Engine", "Installing an SDK for App Engine", "App Engine pricing", and "Quotas in App Engine". The bottom of the screen shows a Windows taskbar with icons for File Explorer, Task View, WhatsApp, YouTube, Settings, Google Chrome, Visual Studio Code, File Explorer, Task View, and a search bar.

Step 8 : Click create Application

The screenshot shows the 'Create app' page in the Google Cloud Platform App Engine interface. On the left, a sidebar menu for 'App Engine' includes options like Dashboard, Services, Versions, Instances, Task queues, Cron jobs, Security scans, Firewall rules, Quotas, Memcache, Search, Release notes, and a 'NEXT' button. The main area features a world map with regions labeled: OCEANIA, NORTH AMERICA, SOUTH AMERICA, AFRICA, ASIA, and ANTLERCTICA. Below the map is a dropdown menu titled 'Select a region *' with 'asia-south1' selected. At the bottom right of the main area is a 'NEXT' button.

Step 9 : Create app and Select Language Python

The screenshot shows the 'Get started' page in the Google Cloud Platform App Engine interface. The left sidebar is identical to the previous screen. The main area has a 'Resources' section with dropdown menus for 'Language' (set to 'Python') and 'Environment' (set to 'Standard'). Below these are links to 'Documentation' and 'Github'. To the right is a 'Deploy with Google Cloud SDK' section with a 'DOWNLOAD THE CLOUD SDK' button and two code snippets: '\$ gcloud init' and '\$ gcloud app deploy'. At the bottom of the main area is a link 'I'LL DO THIS LATER'.

Step 10 : Python app is created in Google App Engine

The screenshot shows the Google Cloud Platform App Engine dashboard for the project "Pravin-Cloud-Project". The left sidebar lists various options: Dashboard, Services, Versions, Instances, Task queues, Cron jobs, Security scans, Firewall rules, Quotas, Memcache, Search, and Release notes. The main content area displays a "Welcome to App Engine" message with the subtext "Build scalable apps in any language on Google's infrastructure". It also states "Your App Engine application has been created" and provides instructions to "Let us help you deploy to your application by pointing you towards the relevant resources based on your programming language." A prominent blue "GET STARTED" button is at the bottom of this message box. The top navigation bar includes links for Internet, Django, Dashboard : Func..., GitHub, Tata Sky – Live TV, YouTube, HackerRank, InfyTQ, Learner Dashboard..., OQE449802346, Our Story | ZS, and a user profile icon. The taskbar at the bottom shows several open applications, including Microsoft Word, Microsoft Excel, Microsoft Powerpoint, Microsoft Edge, Google Chrome, and File Explorer.

Step 12 : Click Cloud Shell in the Pravin-Cloud-Project

Welcome to App Engine

Your App Engine application has been created

GET STARTED

CLOUD SHELL Terminal (pravin-cloud-project) +

```
Welcome to Cloud Shell! Type "help" to get started.  
Your Cloud Platform project in this session is set to pravin-cloud-project.  
Use "gcloud config set project [PROJECT_ID]" to change to a different project.  
pravinbiloli123@cloudshell:~ (pravin-cloud-project)$
```

Step 13 : Create a Directory PythonProject using mkdir command

Syntax : mkdir PythonProject

Welcome to App Engine

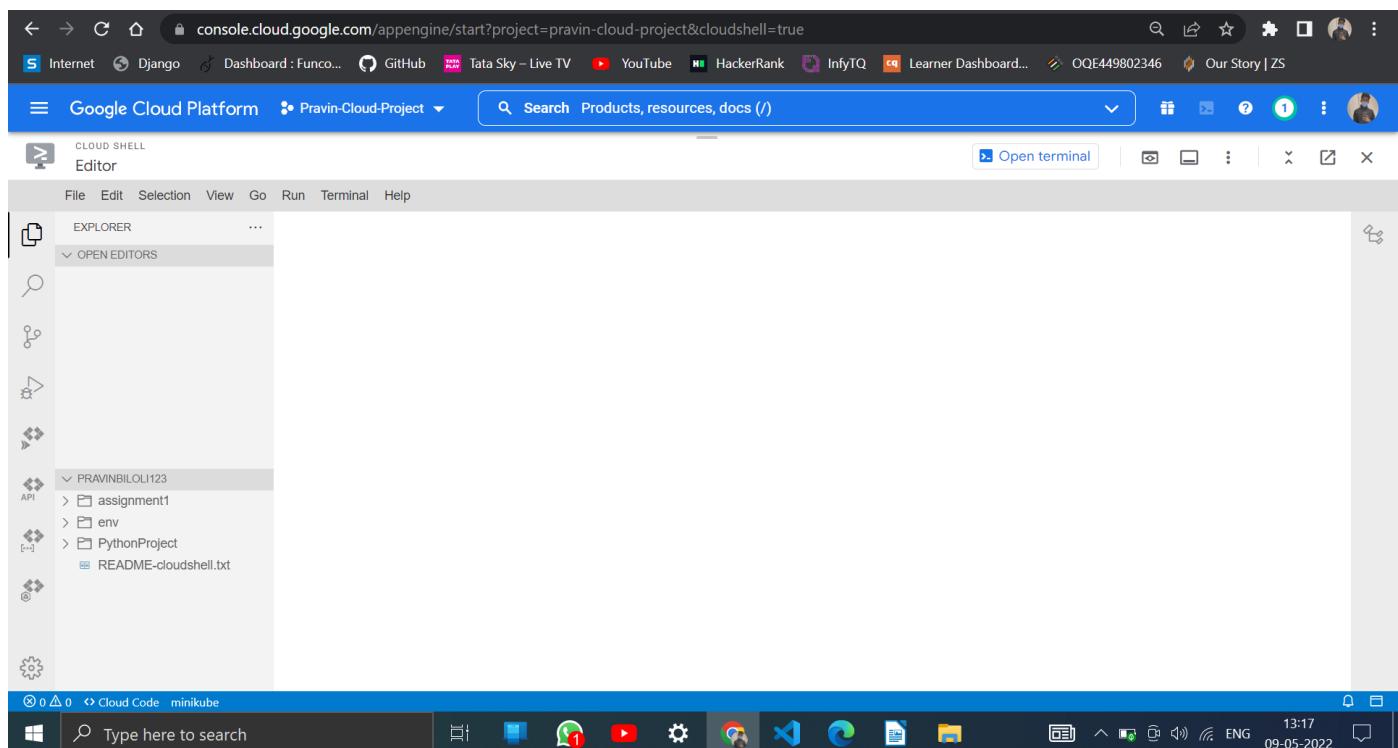
Your App Engine application has been created

GET STARTED

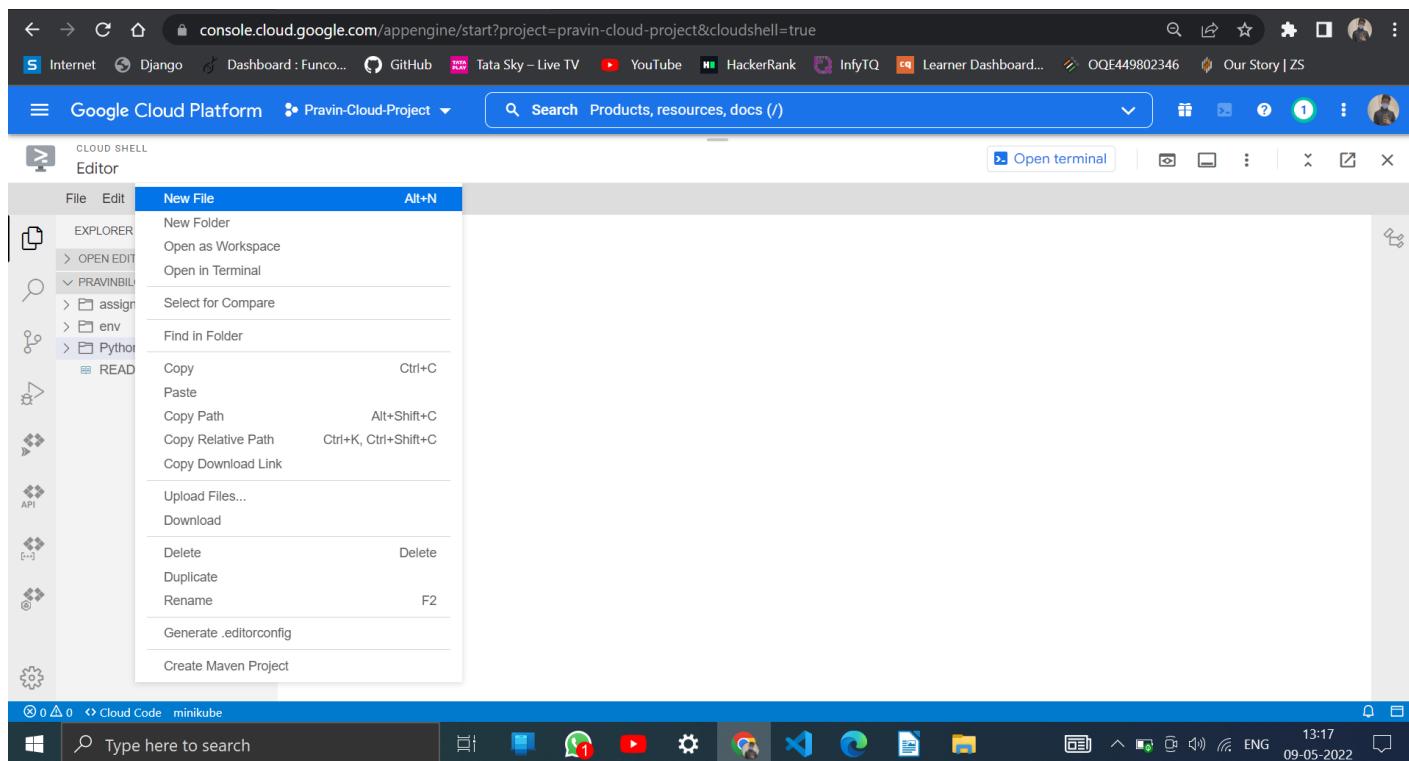
CLOUD SHELL Terminal (pravin-cloud-project) +

```
Welcome to Cloud Shell! Type "help" to get started.  
Your Cloud Platform project in this session is set to pravin-cloud-project.  
Use "gcloud config set project [PROJECT_ID]" to change to a different project.  
pravinbiloli123@cloudshell:~ (pravin-cloud-project)$ mkdir PythonProject  
pravinbiloli123@cloudshell:~ (pravin-cloud-project)$ ls  
assignment1 env PythonProject README-cloudshell.txt  
pravinbiloli123@cloudshell:~ (pravin-cloud-project)$
```

Step 14 : Click Editor to create Python application



Step 15 : Click New File in the PythonProject Folder (Python file)



Step 16 : Create main.py file

The screenshot shows the Google Cloud Platform Editor interface. The left sidebar displays a project structure with files like main.py, app.yaml, README-cloudshell.txt, and assignment1. The main editor area shows the content of main.py:

```
main.py file
from flask import Flask
app = Flask(__name__)
def hello():
    return 'Hello World!'
if __name__ == '__main__':
    app.run(host='127.0.0.1', port=8080, debug=True)
```

The code defines a Flask application with a single route '/' that returns 'Hello World!'. It also includes a conditional check for the __name__ variable to ensure the application runs directly rather than being imported.

Step 17 : Create app.yaml file app.yaml

The screenshot shows the Google Cloud Platform Cloud Shell Editor interface. The left sidebar displays a file tree with 'main.py' and 'app.yaml' selected. The main editor area shows the content of 'app.yaml':

```
runtime: python
env: flex
entrypoint: gunicorn -b :$PORT main:app

runtime_config:
  python_version: 3
```

The status bar at the bottom indicates the file is in YAML format.

```
runtime: python
env: flex
entrypoint: gunicorn -b :$PORT main:app

runtime_config:
  python_version: 3
```

Step 18 : Create requirements.txt file

The screenshot shows the Google Cloud Platform Cloud Shell interface. In the left sidebar, under 'OPEN EDITORS', there is a 'PythonProject' folder containing 'app.yaml', 'main.py', and 'requirements.txt'. The 'requirements.txt' file is open in the main editor area, showing the following content:

```
1 Flask==2.0.2
2 gunicorn==20.1.0
```

The status bar at the bottom right indicates the file has 2 lines, 17 columns, and is in UTF-8 encoding. The date and time shown are 09-05-2022 13:23.

requirements.txt

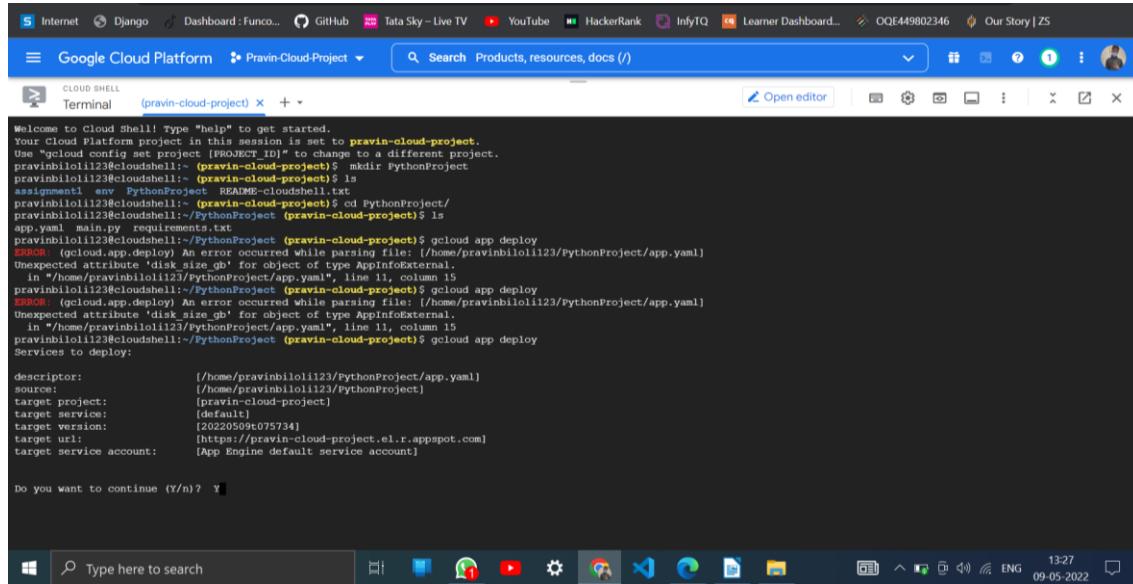
```
Flask==0.11.1
gunicorn==19.6.0
```

Step 19 : Move to Cloud Shell Environment to run the application

The screenshot shows the Google Cloud Platform Cloud Shell terminal. The terminal window title is '(pravin-cloud-project)'. The terminal output shows the user navigating into the project directory and running the command 'app.yaml main.py requirements.txt'. The status bar at the bottom right indicates the date and time as 09-05-2022 13:24.

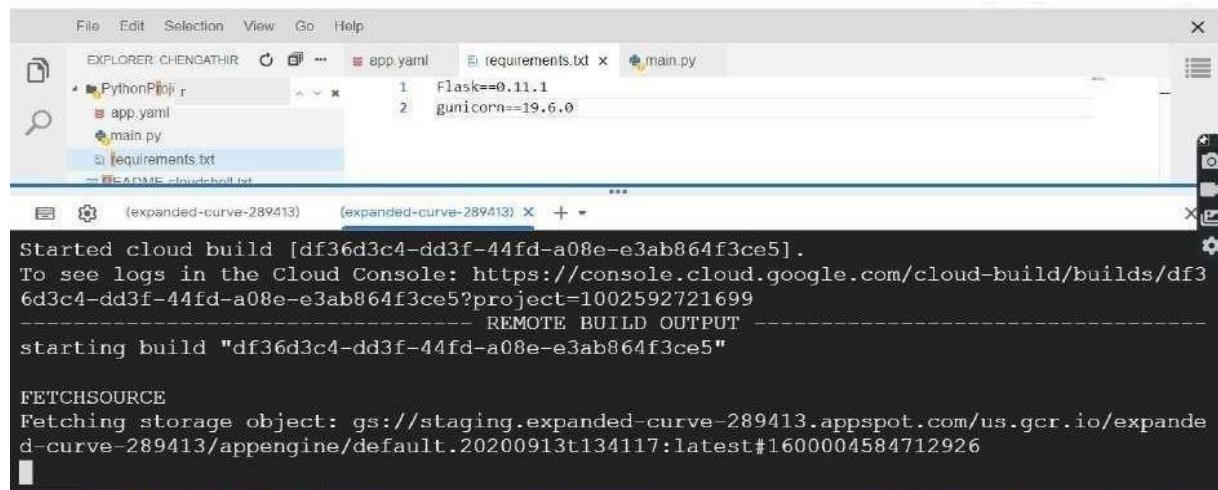
Step 20 : Move to Cloud Shell Environment to run the

applicationSyntax: gcloud app deploy



```
Welcome to Cloud Shell! Type "help" to get started.  
Your Cloud Shell term project in this session is set to pravin-cloud-project.  
Use "cd" and "cd .." to move between projects or to a different project.  
pravinbiloli1123@cloudshell:~ (pravin-cloud-project)$ cd PythonProject  
pravinbiloli1123@cloudshell:~ (pravin-cloud-project)$ ls  
app.yaml main.py requirements.txt  
pravinbiloli1123@cloudshell:~/PythonProject (pravin-cloud-project)$ gcloud app deploy  
ERROR: (gcloud.app.deploy) An error occurred while parsing file: [/home/pravinbiloli1123/PythonProject/app.yaml]  
Unexpected attribute 'disk_size_gb' for object of type AppInfoExternal.  
in "/home/pravinbiloli1123/PythonProject/app.yaml", line 11, column 15  
pravinbiloli1123@cloudshell:~/PythonProject (pravin-cloud-project)$ gcloud app deploy  
ERROR: (gcloud.app.deploy) An error occurred while parsing file: [/home/pravinbiloli1123/PythonProject/app.yaml]  
Unexpected attribute 'disk_size_gb' for object of type AppInfoExternal.  
in "/home/pravinbiloli1123/PythonProject/app.yaml", line 11, column 15  
pravinbiloli1123@cloudshell:~/PythonProject (pravin-cloud-project)$ gcloud app deploy  
Services to deploy:  
  
descriptor:      [/home/pravinbiloli1123/PythonProject/app.yaml]  
source:          [/home/pravinbiloli1123/PythonProject]  
target project:  [pravin-cloud-project]  
target service:   [default]  
target version:  [20220509t075734]  
target url:      [https://pravin-cloud-project.el.r.appspot.com]  
target service account:  [App Engine default service account]  
  
Do you want to continue (y/n)? Y
```

Continue the application. It enable service on the given project



```
Started cloud build [df36d3c4-dd3f-44fd-a08e-e3ab864f3ce5].  
To see logs in the Cloud Console: https://console.cloud.google.com/cloud-build/builds/df3  
6d3c4-dd3f-44fd-a08e-e3ab864f3ce5?project=1002592721699  
----- REMOTE BUILD OUTPUT -----  
starting build "df36d3c4-dd3f-44fd-a08e-e3ab864f3ce5"  
  
FETCHSOURCE  
Fetching storage object: gs://staging.expanded-curve-289413.appspot.com/us.gcr.io/expande  
d-curve-289413/appengine/default.20200913t134117:latest#1600004584712926
```

It started building the object and fetching the storage object for the created application

```
File Edit Selection View Go Help
EXPLORER CHENGATHIR ① ... app.yaml requirements.txt main.py
PythonProject
  app.yaml
  main.py
  requirements.txt
  README-cloudshell.txt
(expanded-curve-289413) (expanded-curve-289413) + ...
89e14614ab6b: Layer already exists
18ecc1cf8b2f: Pushed
15942b628b0d: Layer already exists
ed16353646db: Pushed
latest: digest: sha256:a56ba6724428bce0a0f89a6258cab5d985097ed4dd85027c24bcd176ca06d4b6 size: 3457
DONE

Updating service [default] (this may take several minutes)....
```

It is updating the service

```
File Edit Selection View Go Help
EXPLORER CHENGATHIR ① ... app.yaml requirements.txt main.py
PythonProject
  app.yaml
  main.py
  requirements.txt
  README-cloudshell.txt
(expanded-curve-289413) (expanded-curve-289413) + ...
DONE

Updating service [default] (this may take several minutes)...done.
Setting traffic split for service [default]...done.
Deployed service [default] to [https://expanded-curve-289413.uc.r.appspot.com]

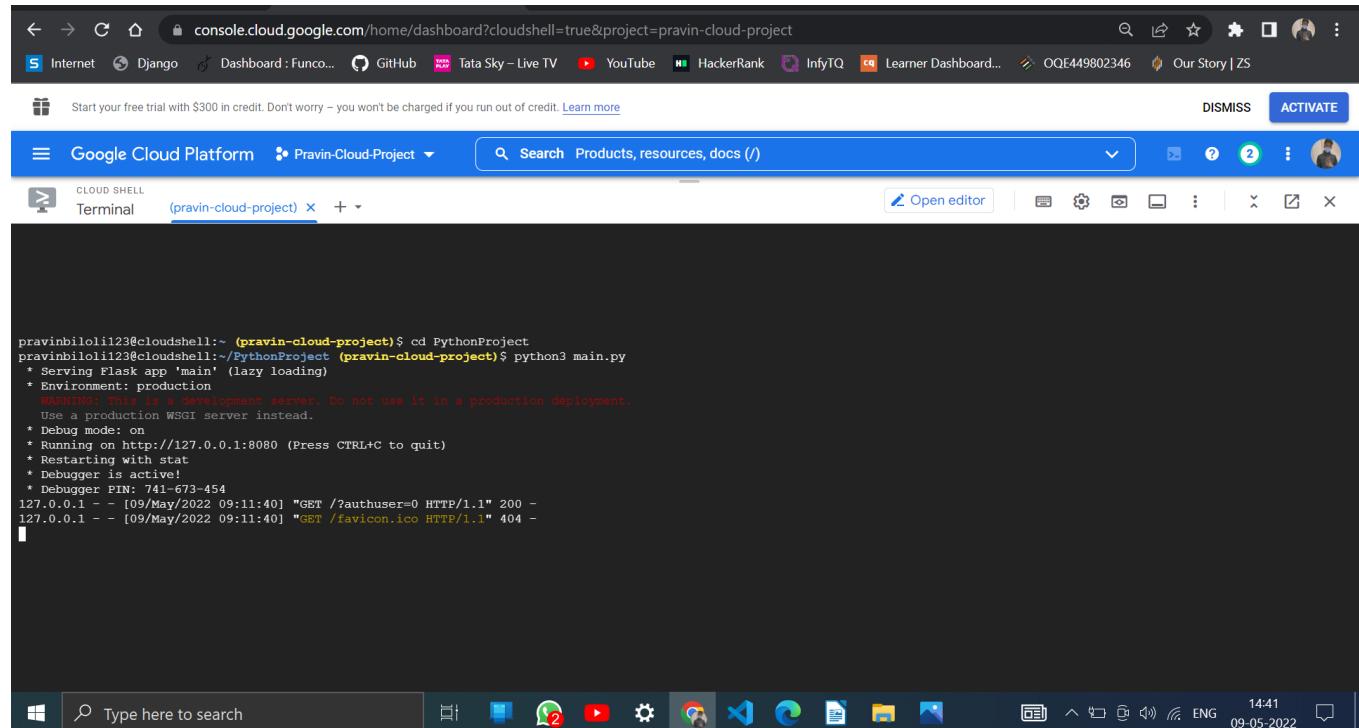
You can stream logs from the command line by running:
$ gcloud app logs tail -s default

To view your application in the web browser run:
$ gcloud app browse
chengathir@cloudshell:~/PythonProject (expanded-curve-289413)$
```

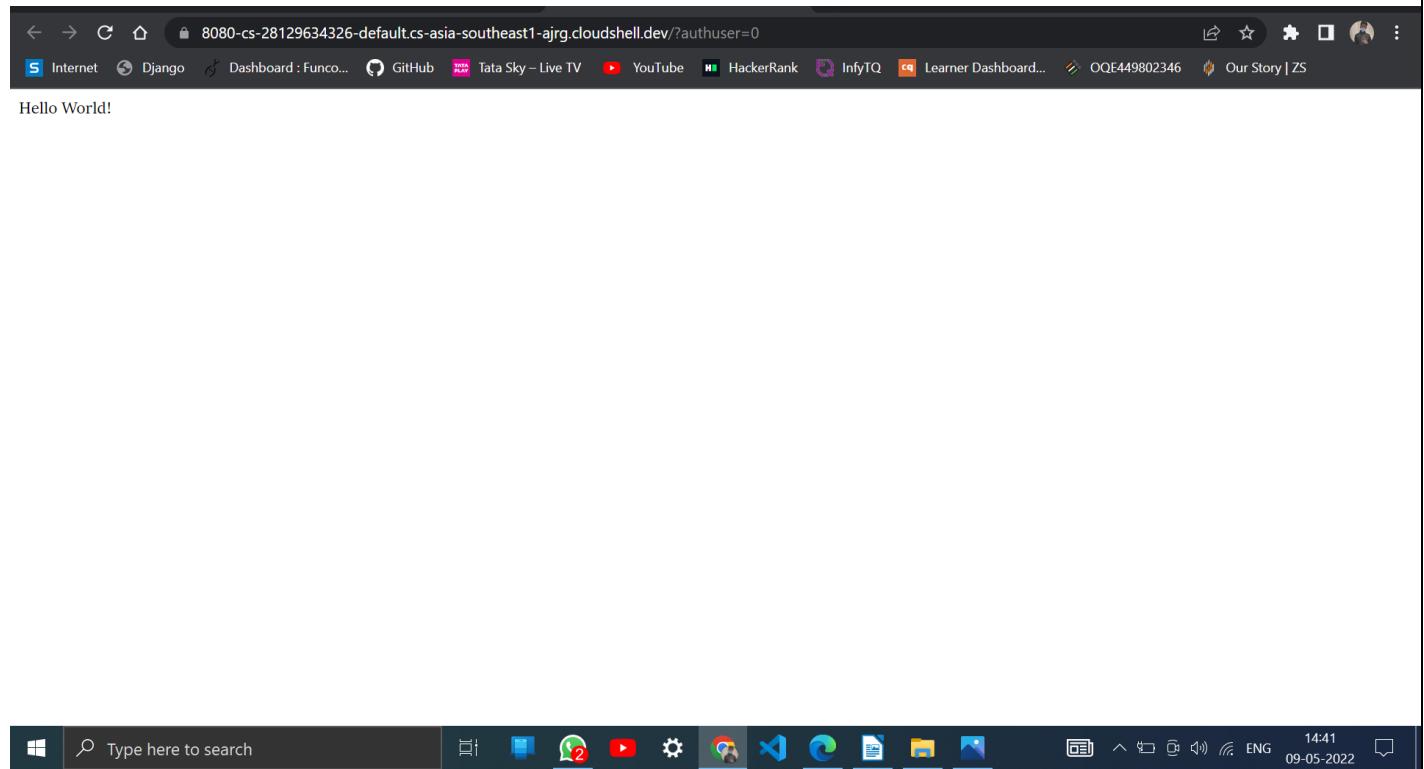
The application is successfully deployed and URL is

<https://expanded-curve-289413.uc.r.appspot.com>

Step 21 : Run your program in the browser



Step 22 : Hello World Program is sucessfully run in the browser



Result:

Thus the Google App Engine is installed successfully and a web application to display hello world using python is developed and deployed in the GAE and used GAE Launcher to launch the web applications.

Ex No. 3 a

Simulate a cloud scenario using CloudSim

Introduction:

† CloudSim

- A Framework for modeling and simulation of Cloud Computing Infrastructures and services
- Originally built at the Cloud Computing Distributed Systems (CLOUDS) Laboratory, The University of Melbourne, Australia
- It is completely written in JAVA

† Main Features of CloudSiM

◦ Modeling and simulation

- Data centre network topologies and message-passing applications
- Dynamic insertion of simulation elements
- Stop and resume of simulation
- Policies for allocation of hosts and virtual machines

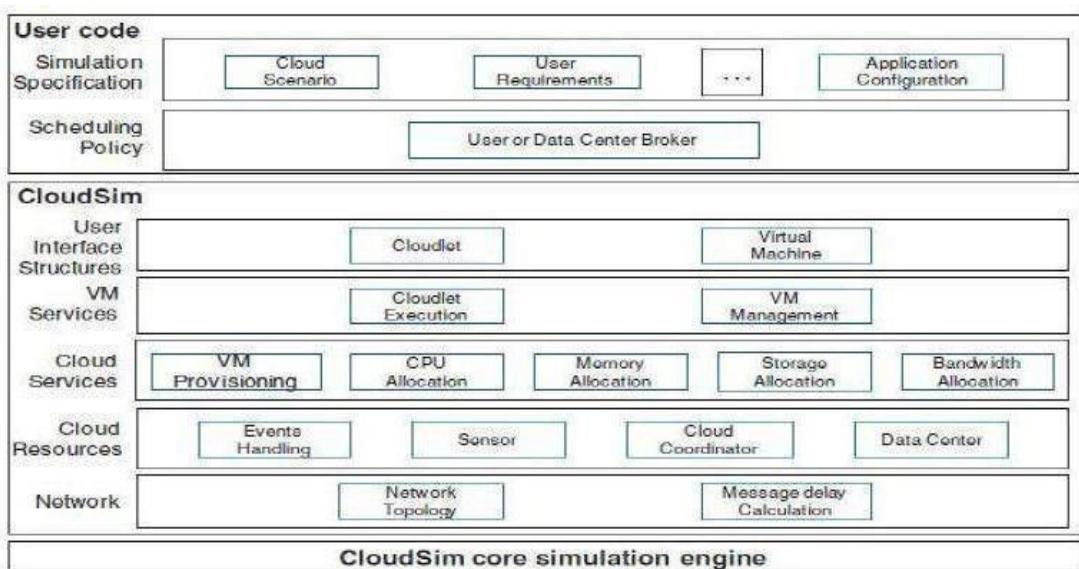
† Cloudsim – Essentials

- JDK 1.6 or above <http://tinyurl.com/JNU-JAVA>
- Eclipse 4.2 or above <http://tinyurl.com/JNU-Eclipse>
- Alternatively NetBeans <https://netbeans.org/downloads>
- Up & Running with cloudsim guide: <https://goo.gl/TPL7Zh>

† Cloudsim-Directory structure

- cloudsim/ -- top level CloudSim directory
- docs/ -- CloudSim API Documentation
- examples/ -- CloudSim examples
- jars/ -- CloudSim jar archives
- sources/ -- CloudSim source code

† Cloudsim - Layered Architecture



† **Cloudsim - Component model classes** ○

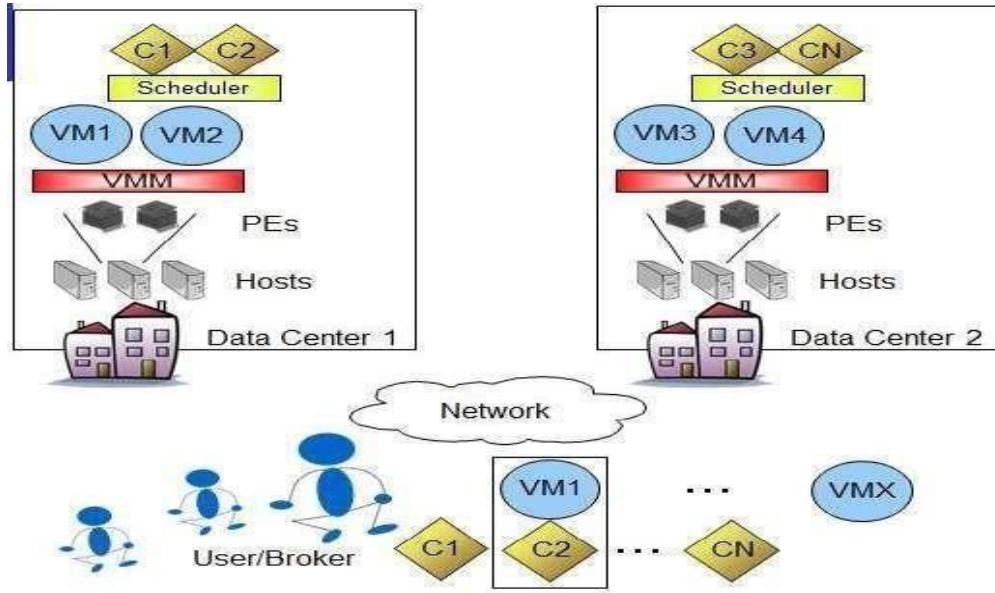
- CloudInformationService.java ○
- Datacenter.java,Host.java,Pe.java ○
- Vm.java,Cloudlet.java ○ DatacenterBroker.java
 - Storage.java,HarddriveStorage.java, SanStorage.java

† **Cloudsim - Major blocks/Modules** ○

- org.cloudbus.cloudsim ○
- org.cloudbus.cloudsim.core ○
- org.cloudbus.cloudsim.core.predicates ○
- org.cloudbus.cloudsim.distributions ○
- 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
 - org.cloudbus.cloudsim.util

† **Cloudsim - key components** ○ Datacenter ○

- DataCenterCharacteristics ○ Host ○
- DatacenterBroker ○ RamProvisioner ○
- BwProvisioner ○ Storage
 - Vm
 - VMAllocationpolicy ○ VmScheduler ○ Cloudlet ○ CloudletScheduler ○
 - CloudInformationService ○ CloudSim ○ CloudSimTags ○ SimEvent ○
 - SimEntity ○ CloudsimShutdown ○ FutureQueue ○ DefferedQueue ○
 - Predicate and associative classes.



CloudSim Elements/Components

Procedure to import Eclipse, Cloudsim in your system

Step 1: Link to download Eclipse and download Eclipse for Windows 64bit into your Local machine

<https://www.eclipse.org/downloads/packages/release/kepler/sr1/eclipse-ide-java-developers>

The screenshot shows the Eclipse Foundation website's download section for the Eclipse IDE for Java Developers Kepler SR1. The page has a dark header with the Eclipse Foundation logo and navigation links for Projects, Working Groups, Members, and More+. Below the header, there are links for Eclipse Installer, Eclipse Packages, and Eclipse Developer Builds. A red banner at the top states: "This package was released on 09/26/2013. A newer package is available [here](#)". The main content area features a large image for the HDC.Cloud conference. On the left, there's a "Package Description" section for the Eclipse IDE for Java Developers, which includes a brief overview and a "This package includes:" list. The list contains items like Code Recommenders Developer Tools, Eclipse Git Team Provider, Eclipse Java Development Tools, Maven Integration for Eclipse, Mylyn Task List, WindowBuilder Core, Eclipse XML Editors and Tools, and a link to the Detailed features list. To the right, there are "Download Links" for Windows 32-bit/x86_64, macOS 32-bit/x86_64, Linux 32-bit/x86_64, and a note that the Eclipse Installer 2022-03 R now includes a JRE for macOS, Windows and Linux. There are also links for Downloaded 1,044,337 Times, Checksums..., and Bugzilla. At the bottom, there's a cookie consent message from the Eclipse Foundation.

Some Eclipse Foundation pages use cookies to better serve you when you return to the site. You can set your browser to notify you before you receive a cookie or turn off cookies. If you do so, however, some areas of some sites may not function properly. To read Eclipse Foundation Privacy Policy [click here](#).

Step 2: Download cloudsim-3.0.3 from git hub repository in your local machine

<https://github.com/Cloudslab/cloudsim/releases/tag/cloudsim-3.0.3>

The screenshot shows the GitHub release page for the 'cloudsim' repository. The title is 'cloudsim-3.0.3'. It was released by nikolayg on March 19, 2015, with 55 commits since master. The page includes sections for 'WHAT'S NEW' and 'Changes from CloudSim 3.0.2 to CloudSim 3.0.3', which lists bug fixes and refactoring changes.

Step 3: Download commons-math3-3.6.1 from git hub repository in your local

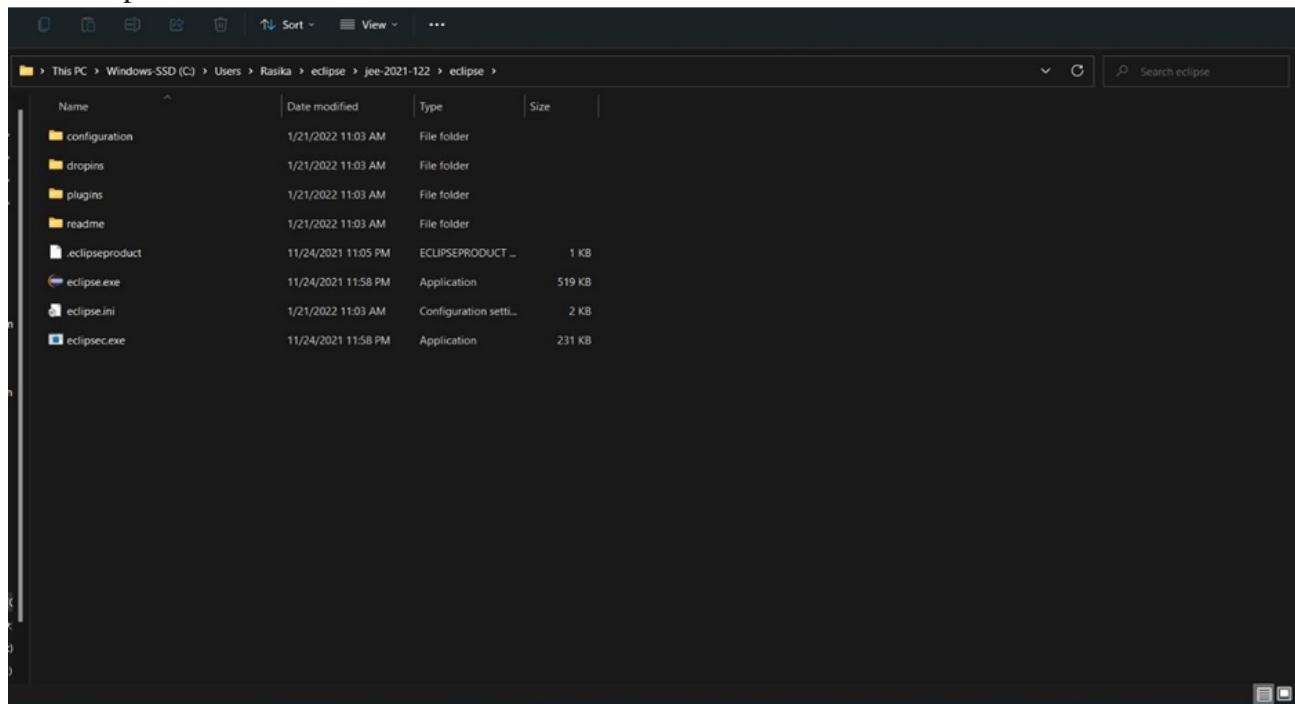
machine https://commons.apache.org/proper/commons-math/download_math.cgi

The screenshot shows the Apache Commons Math 3.6.1 download page. It features a sidebar with links for Javadoc, Issue Tracking, Source Repository, Wiki, Developers Guide, and Proposal. The main content area is titled 'Apache Commons Math 3.6.1 (requires Java 5+)'. It provides download links for 'Binaries' (tar.gz and zip files) and 'Source' (tar.gz and zip files). A 'Source' section also includes a link to the 'Archives' page, which lists older releases.

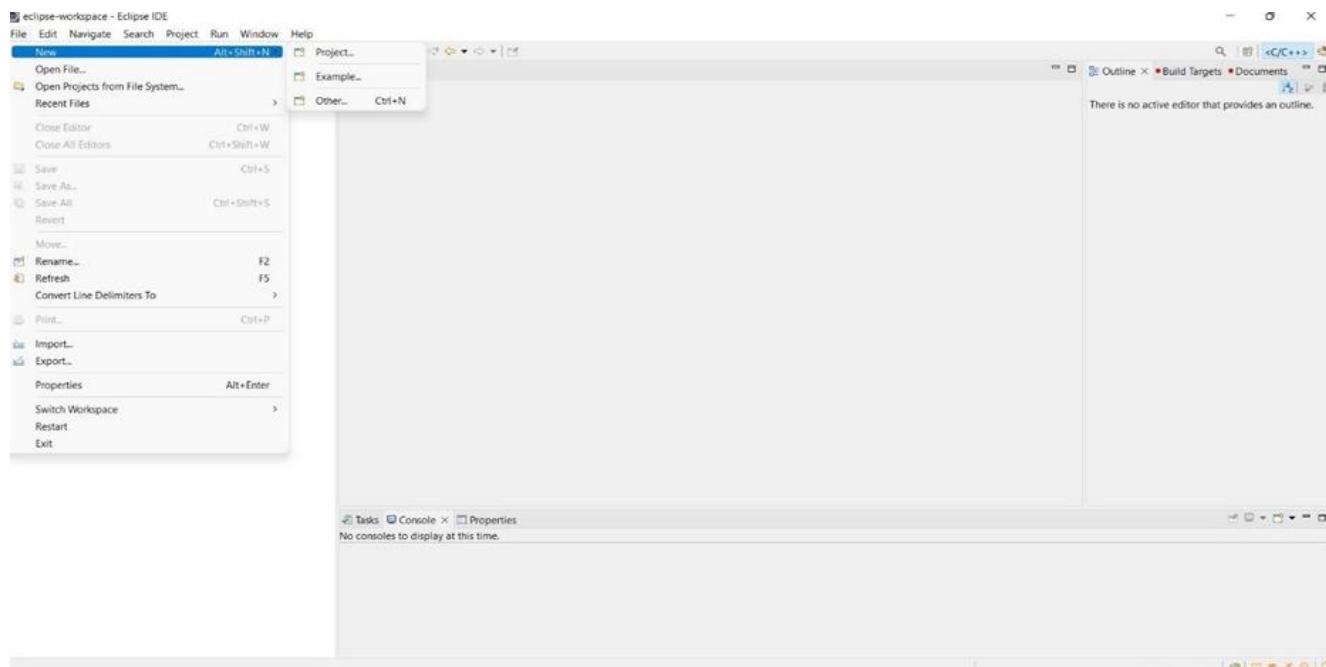
Step 4: Downloaded Eclipse, cloudsim-code-master and Apache Commons Math 3.6.1 in your local machine and extract cloudsim-3.0.3 and Apache Commons Math 3.6.1

VSCodeUserSetup-x64-143.0.exe	3/12/2020 10:31 PM	Application	57,515 KB
Windows10Upgrade9252 (1).exe	12/1/2020 10:51 PM	Application	6,072 KB
Windows10Upgrade9252.exe	12/1/2020 10:34 PM	Application	6,072 KB
commons-math3-3.6.1-bin.zip	5/5/2022 12:01 PM	Compressed (zipp...)	21,712 KB
cloudsim-3.0.3.zip	5/5/2022 12:01 PM	Compressed (zipp...)	13,367 KB

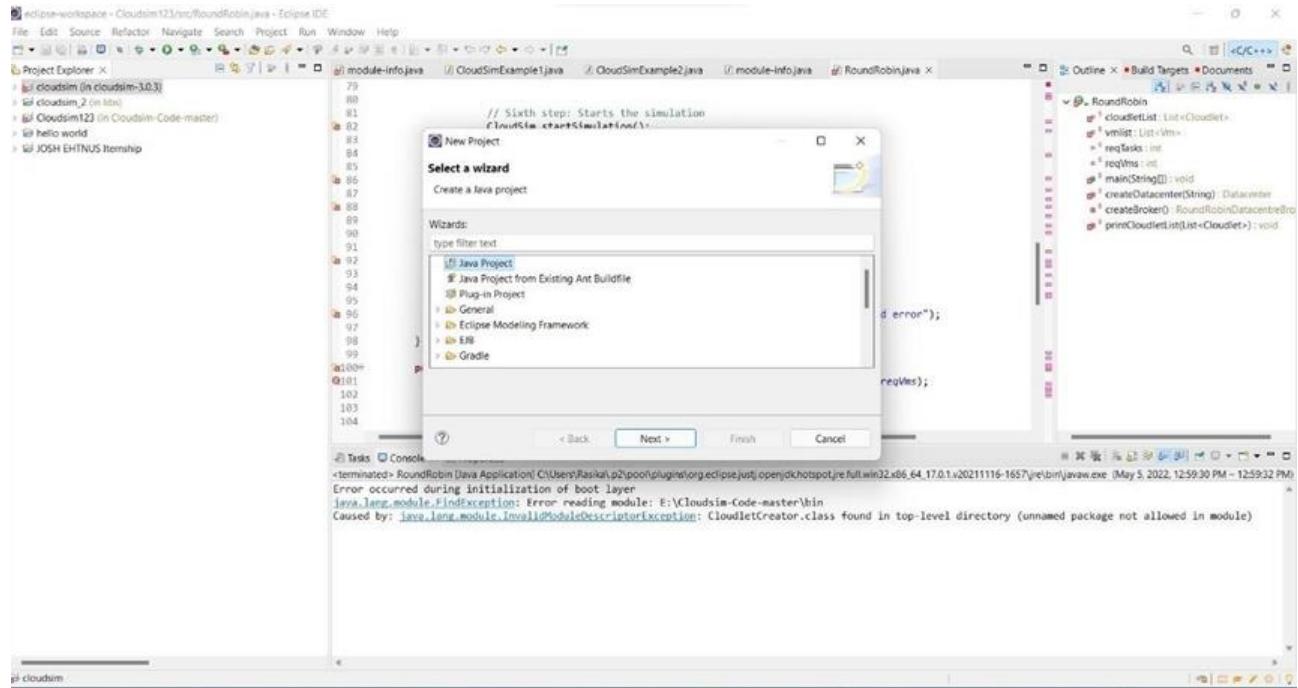
Step 5: First of all, navigate to the folder where you have unzipped the eclipse folder and open Eclipse.exe



Step 6: Now within Eclipse window navigate the menu: *File* -> *New* -> *Project*, to open the new project wizard

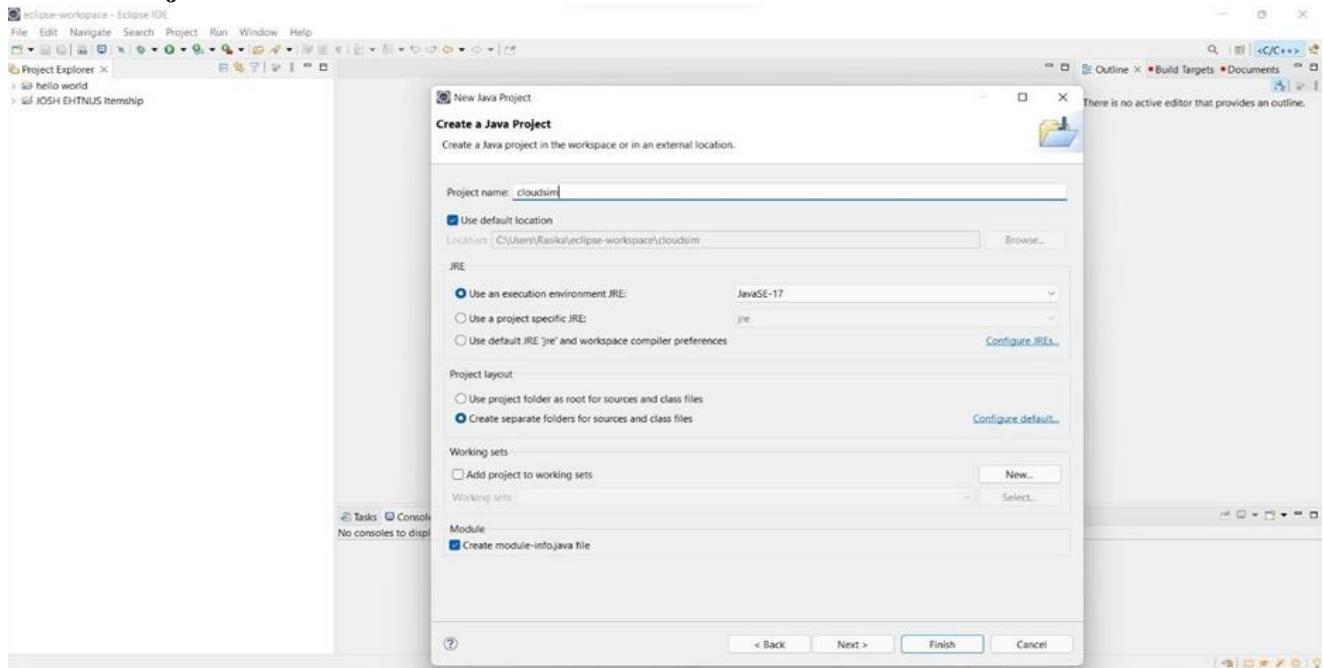


Step 7: A New Project wizard should open. There are a number of options displayed and you have to find & select the Java Project option, once done click 'Next'

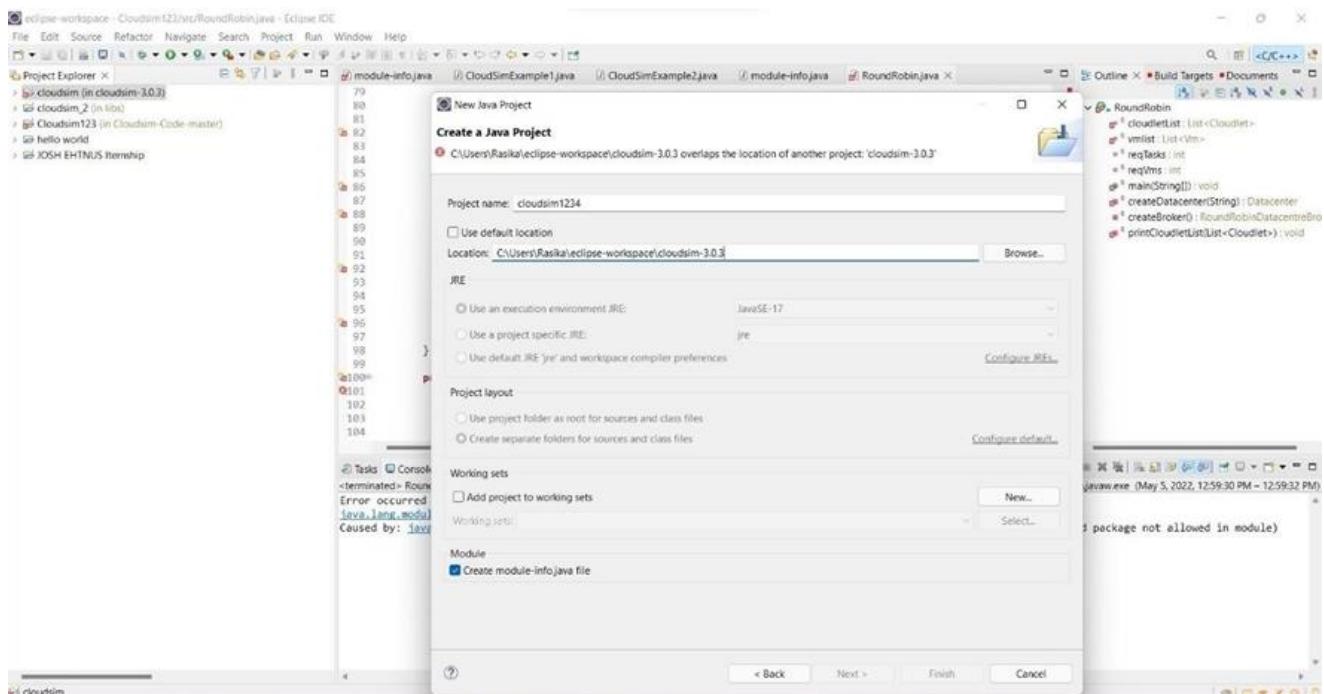


Step 8: Now a detailed new project window will open, here you will provide the project name and the path of CloudSim project source code, which will be done as follows:

Project Name: CloudSim.

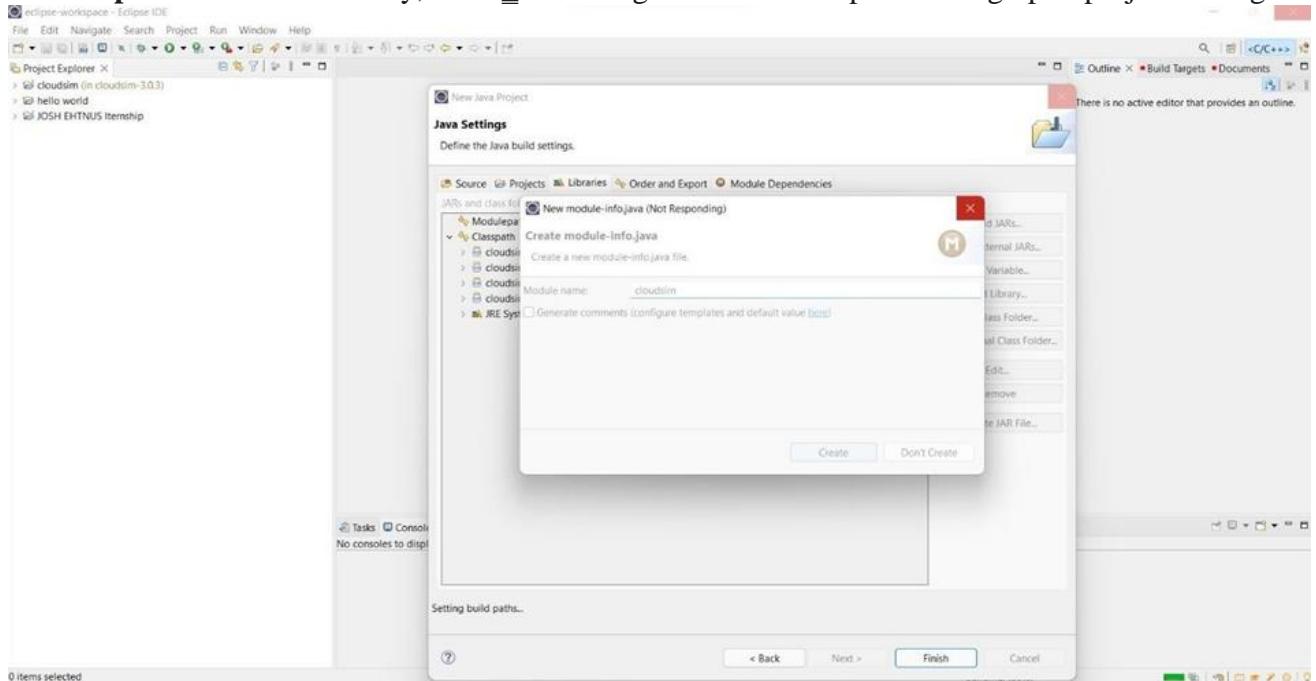


Step 9: Unselect the '*Use default location*' option and then click on '*Browse*' to open the path where you have unzipped the Cloudsim project and finally click Next to set project settings.

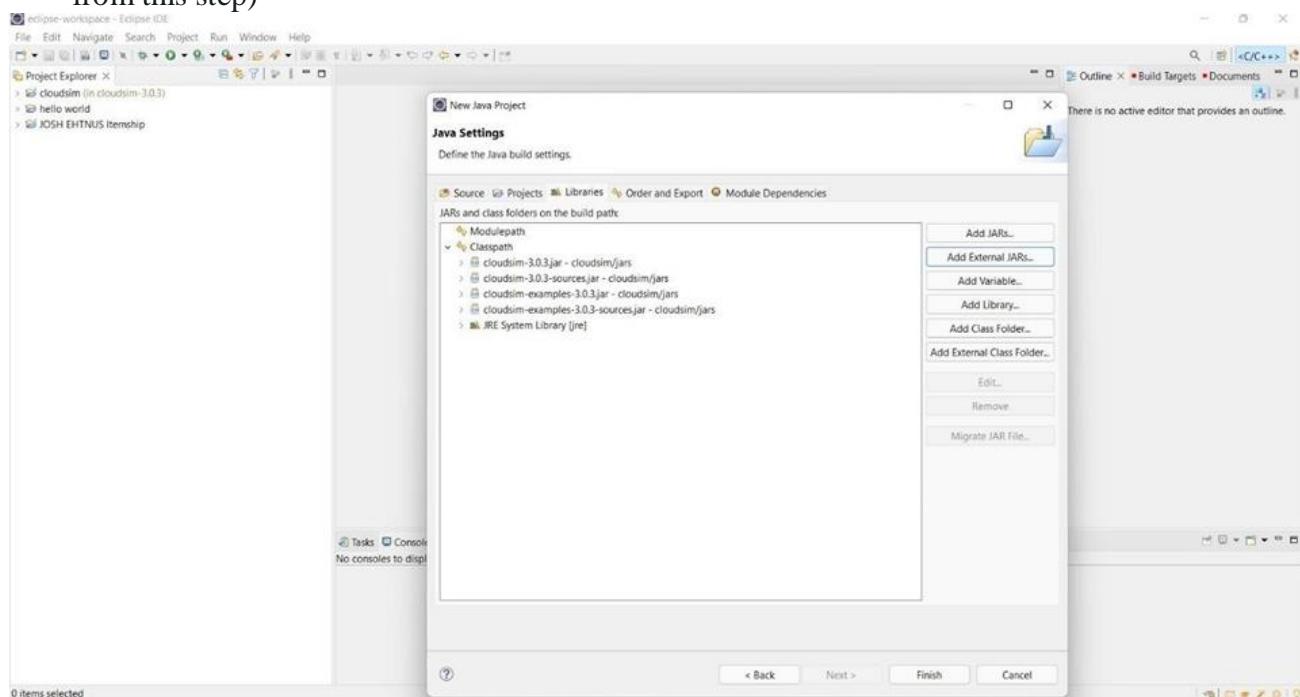


Step 10: Make sure you navigate the path till you can see the bin, docs, examplesetc folder in the navigation plane.

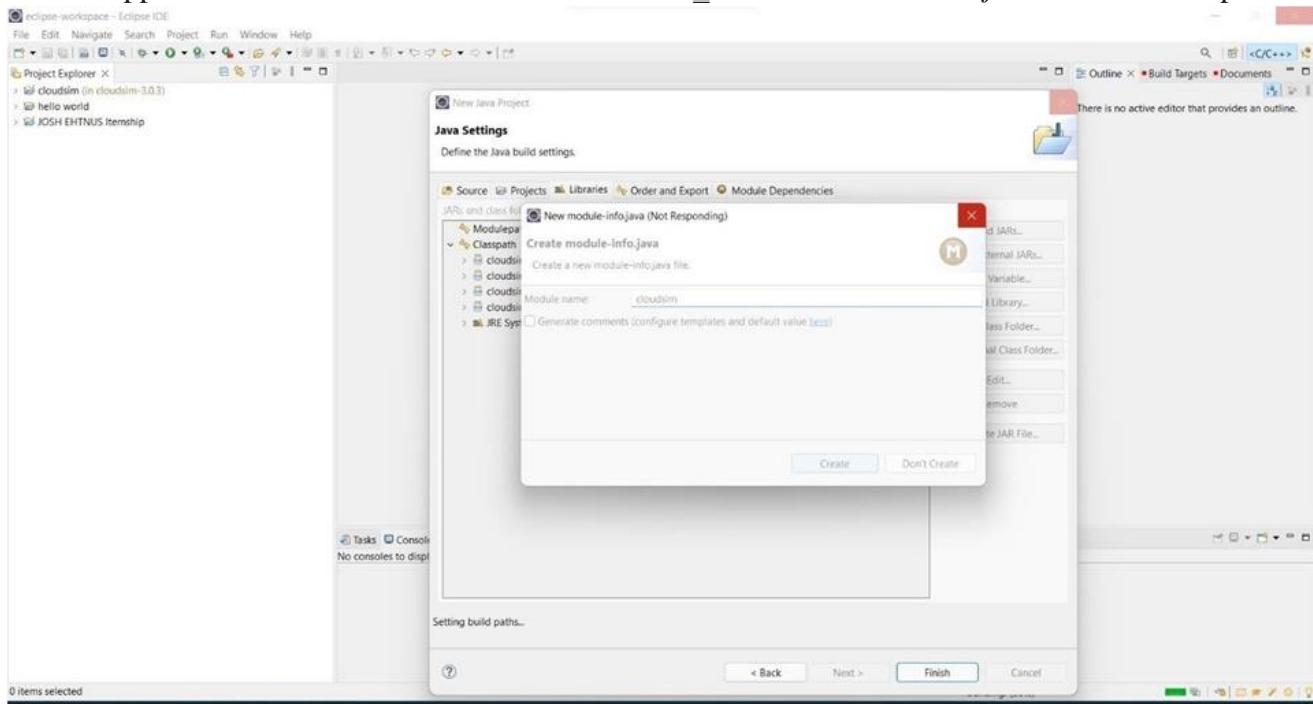
Step 11: Once done finally, click “Next” to go to the next step i.e. setting up of project settings



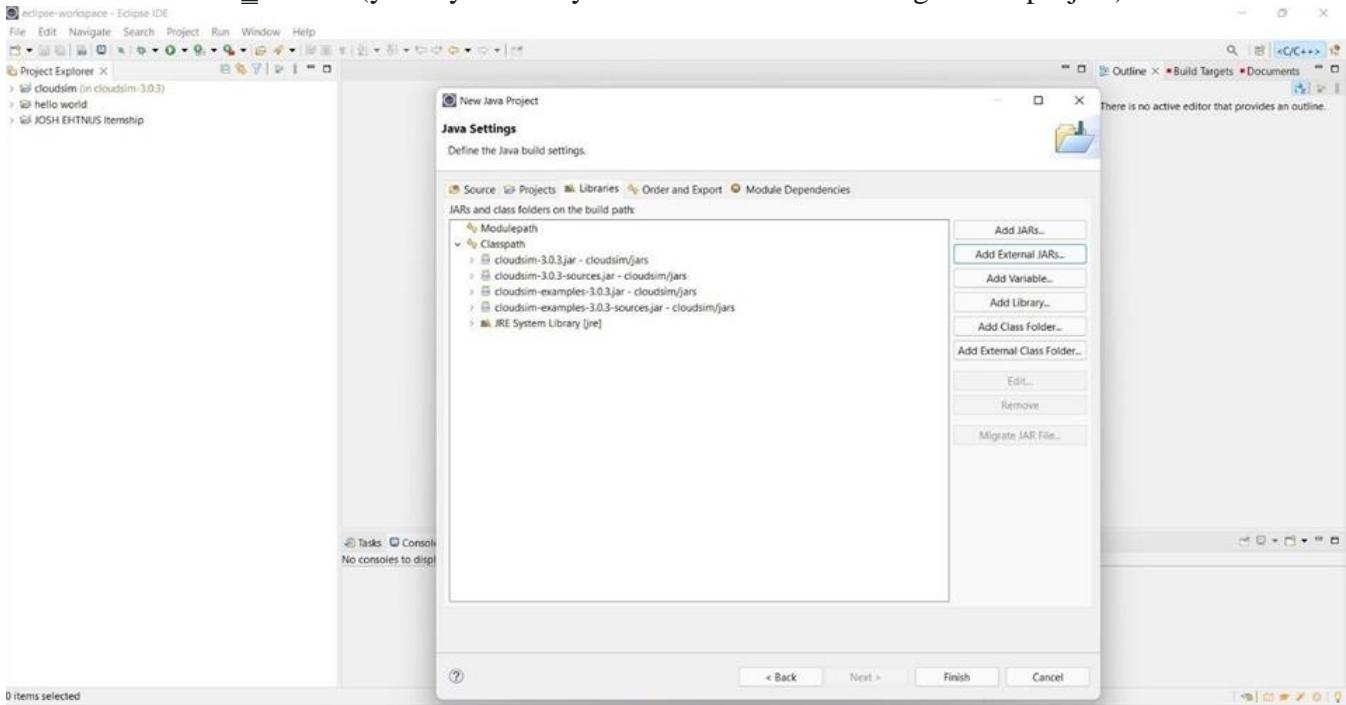
Step 12: Now open ‘Libraries’ tab and if you do not find commons-math3-3.x.jar (here ‘x’ means the minor version release of the library which could be 2 or greater) in the list then simply click on “Add External Jar” (commons-math3-3.x.jar will be included in the project from this step)



Step 13: Once you have clicked on Add External JAR's Open the path where you have unzipped the commons-math binaries and select Commons-math3-3.x.jar and click on open.



Step 14: Ensure external jar that you opened in the previous step is displayed in the list and then click on Finish (your system may take 2-3 minutes to configure the project)



Step 15: Once the project is configured you can open the Project Explorer and start exploring the Cloudsim project. Also for the first time eclipse automatically start building the workspace for newly configured Cloudsim project, which may take some time depending on the configuration of the computer system.

Following is the final screen which you will see after Cloudsim is configured.

Step 16: Now just to check you within the Project Explorer, you should navigate to the examples folder, then expand the package org.cloudbus.cloudsim.examples and double click to open the CloudsimExample1.java

The screenshot shows the Eclipse IDE interface with the title "eclipse-workspace - cloudsim/examples/module-info.java - Eclipse IDE". The Project Explorer view on the left lists several projects and files, including "cloudsim (in cloudsim-3.0.3)", "IRE System Library [ire]", and "examples". The "examples" folder contains subfolders like "org", "org.cloudbus", "org.cloudbus.cloudsim", and "org.cloudbus.cloudsim.examples". Inside "examples", there are multiple Java files such as "CloudSimExample1.java", "CloudSimExample2.java", "CloudSimExample3.java", "CloudSimExample4.java", "CloudSimExample5.java", "CloudSimExample6.java", "CloudSimExample7.java", and "CloudSimExample8.java". The "org.cloudbus.cloudsim.examples" folder also contains "network", "power", and "power.planetlab" subfolders. The "build.xml", "changelog.txt", "examples.txt", "license.txt", "pom.xml", "readme.txt", and "release_notes.txt" files are listed under the root "examples" folder. The central workspace shows the code for "module-info.java":

```
1 module cloudsim {  
2     exports org.cloudbus.cloudsim.network;  
3     exports org.cloudbus.cloudsim.examples.network;  
4     exports org.cloudbus.cloudsim.distributions;  
5     exports org.cloudbus.cloudsim.power.lists;  
6     exports org.cloudbus.cloudsim.util;  
7     exports org.cloudbus.cloudsim.examples;  
8     exports org.cloudbus.cloudsim.examples.power.random;  
9     exports org.cloudbus.cloudsim.examples.power.random;  
10    exports org.cloudbus.cloudsim.core.predicates;  
11    exports org.cloudbus.cloudsim.lists;  
12    exports org.cloudbus.cloudsim.examples.power;  
13    exports org.cloudbus.cloudsim.power.models;  
14    exports org.cloudbus.cloudsim.provisioners;  
15    exports org.cloudbus.cloudsim.network.datacenter;  
16    exports org.cloudbus.cloudsim.examples.power.planetlab;  
17    exports org.cloudbus.cloudsim;  
18    exports org.cloudbus.cloudsim.core;  
19    exports org.cloudbus.cloudsim.power;  
20  
21    requires cloudsim;  
22    requires cloudsim.examples;  
23 }
```

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer** (left): Shows the project structure with packages like org.cloudsim, org.cloudsim.examples, and org.cloudsim.examples.network.
- CloudSimExample1.java** (center): The code editor displays the Java source code for CloudSimExample1. The code initializes the CloudSim package, creates a datacenter with one host, and runs one VM on it. It includes imports for java.util.List and java.text.DecimalFormat, and annotations for SuppressWarnings and Main.
- Outline** (right): Shows the class hierarchy and member definitions for CloudSimExample1.

```
module-info.java  ||| CloudSimExample1.java
1 package org.cloudsim.cloudsim.examples;
2
3 /**
4  * Title: CloudSim Toolkit
5 */
6
7 import java.util.List;
8 import java.text.DecimalFormat;
9
10 /**
11  * A simple example showing how to create a datacenter with one host and run one
12  * cloudlet on it.
13 */
14
15 public class CloudSimExample1 {
16
17     /**
18      * The cloudlet list.
19     */
20     private static List<Cloudlet> cloudletList;
21
22     /**
23      * The vmlist.
24     */
25     private static List<Vm> vmList;
26
27     /**
28      * Creates main() to run this example.
29     */
30     @Param(args = "args")
31     /**
32      * @param args
33     */
34     @SuppressWarnings("unused")
35     public static void main(String[] args) {
36
37         log.println("Starting CloudSimExample1...");
38
39         try {
40             // First step: Initialize the CloudSim package. It should be called
41             // before creating any entities.
42
43             // Create a Datacenter
44             Datacenter dc = Datacenter.createDatacenter("Datacenter1");
45
46             // Create a Host
47             Host host = dc.getHost("Host1");
48
49             // Create a Cloudlet
50             Cloudlet cloudlet = new Cloudlet("Cloudlet1", 1, 1);
51
52             // Add Cloudlet to the Cloudlet list
53             cloudletList.add(cloudlet);
54
55             // Create a VM
56             Vm vm = host.createVm("Vm1");
57
58             // Add VM to the VM list
59             vmList.add(vm);
60
61             // Run the Cloudlet
62             host.runCloudlet(cloudlet);
63
64         } catch (Exception e) {
65             e.printStackTrace();
66         }
67     }
68 }
```

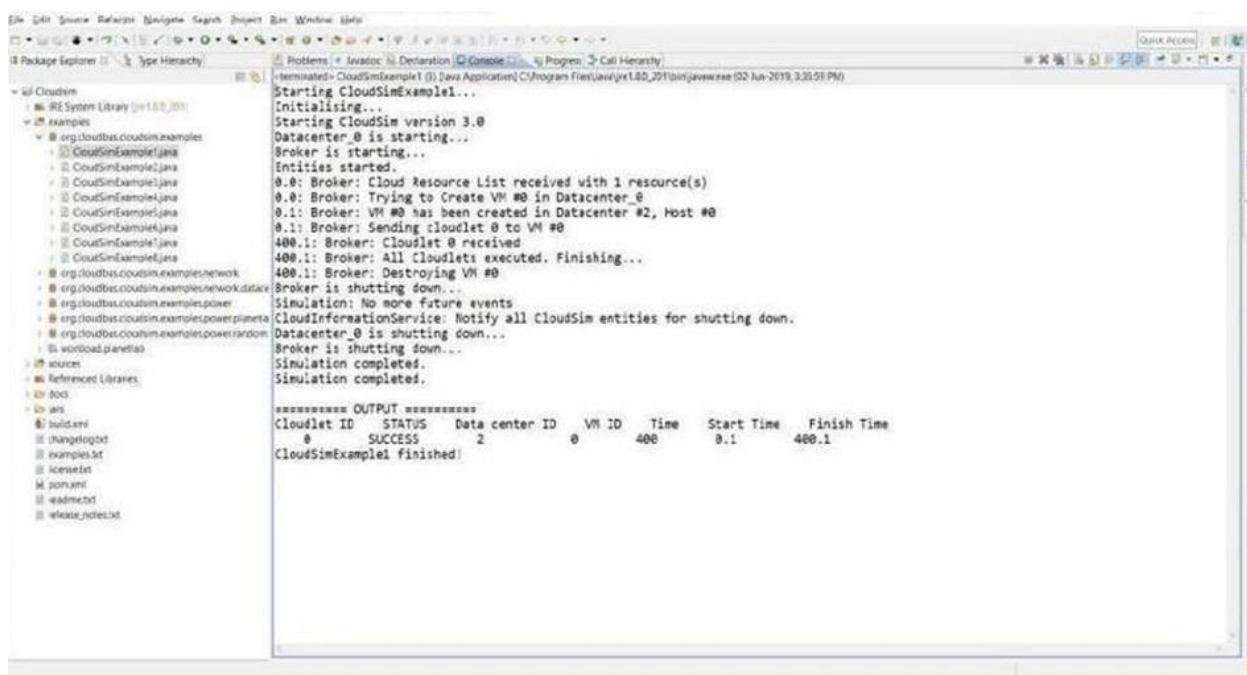
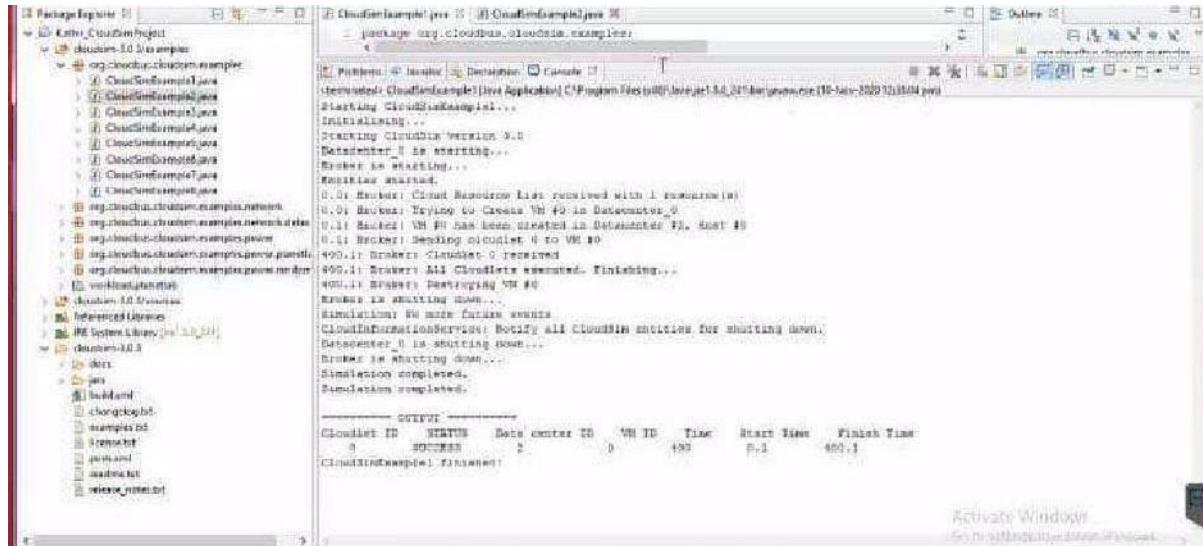
The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace - cloudsim/examples/org/cloudbus/cloudsim/examples/CloudSimExample2.java - Eclipse IDE
- File Menu:** File Edit Source Refactor Navigate Search Project Run Window Help
- Project Explorer:** Shows the project structure under "cloudsim (cloudsim-3.0.3)".
- Editor:** Displays the Java code for `CloudSimExample2.java`. The code creates a datacenter with one host and two cloudlets, runs them in VMs with the same MIPS requirements, and prints the cloudlet list.
- Outline View:** Shows the class structure and methods: `CloudSimExample2`, `cloudletList`, `vmList`, `mainStringID`, `createDatacenter`, `createBroker`, and `printCloudletList`.
- Tasks, Console, Properties:** Standard Eclipse workspace components.

```
2 * Title: CloudSim Toolkit
3 package org.cloudbus.cloudsim.examples;
4
5 import java.text.DecimalFormat;
6
7 /**
8  * A simple example showing how to create
9  * a datacenter with one host and run two
10 * cloudlets on it. The cloudlets run in
11 * VMs with the same MIPS requirements.
12 * The cloudlets will take the same time to
13 * complete the execution.
14 */
15 public class CloudSimExample2 {
16
17     /** The cloudlet list. */
18     private static List<Cloudlet> cloudletList;
19
20     /** The vm list. */
21     private static List<Vm> vmList;
22
23     /**
24      * Creates main() to run this example
25      */
26     public static void main(String[] args) {
27
28         Log.printLine("Starting CloudSimExample2...");
29
30     }
31 }
```

Step 17: Now navigate to the Eclipse menu _Run ->Run_ or directly use a keyboard shortcut 'Ctrl + F11' to execute the _CloudsimExample1.java_.

Step 18: If it is successfully executed it should be displaying the following type to output in the console window of the Eclipse IDE.



Result:

Thus the cloudsim is simulated using Eclipse Environment successfully.

Ex No. 3 b

Simulate a cloud scenario using CloudSim and running a scheduling algorithm

Procedure to import Eclipse, running scheduling algorithms in your system

Step 1: Link to download Eclipse and download Eclipse for Windows 64bit into your Local machine

<https://www.eclipse.org/downloads/packages/release/kepler/sr1/eclipse-ide-java-developers>

The screenshot shows the Eclipse Foundation website's download section. At the top, there's a navigation bar with links for 'Projects', 'Working Groups', 'Members', and 'More...'. Below the navigation, a banner indicates the package was released on 09/26/2013, with a link to a newer version. The main content area features a large image of a mountain range with the text '< HDC.Cloud >' overlaid. To the left, there's a section for 'Eclipse IDE for Java Developers' with a small icon, followed by a 'Package Description' and a list of included tools. On the right, there are 'Download Links' for various operating systems, download statistics (1,044,337 times), checksums, and a link to Bugzilla. A note at the bottom mentions the inclusion of a JRE for macOS, Windows, and Linux in the installer. A cookie consent banner at the bottom asks if the user wants to allow cookies, with 'Decline' and 'Allow cookies' buttons.

Step 2: Download scheduling source code **cloudsim-code-master** from git hub repository in your local machine

<https://github.com/shiro873/Cloudsim-Code>

The screenshot shows a GitHub repository page for 'shiro873 / Cloudsim-Code'. The repository has 1 branch and 0 tags. It contains 4 commits from 'shiro873' with the message 'Errors fixed'. The README.md file describes the implementation of scheduling algorithms in cloudsim. The repository has 14 forks and 13 stars.

Step 3: Download commons-maths3-3.6.1 from git hub repository in your local machine

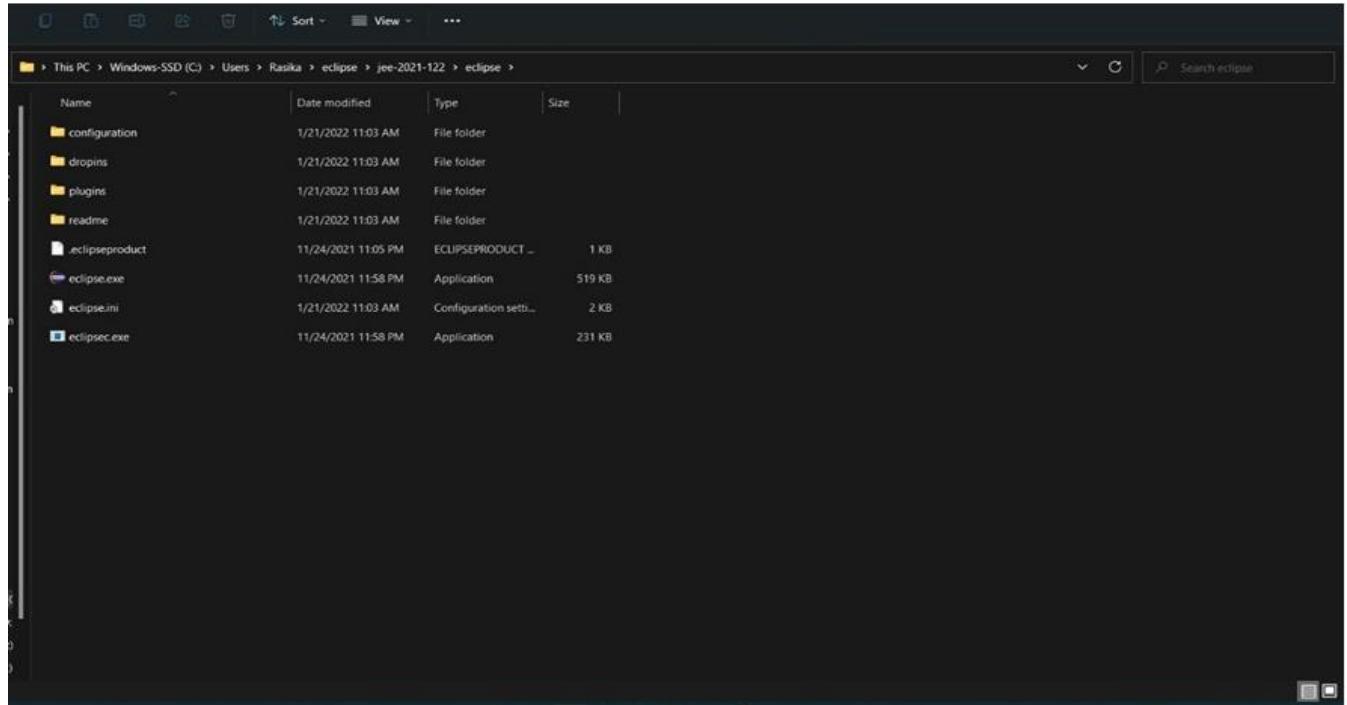
https://commons.apache.org/proper/commons-math/download_math.cgi

The screenshot shows the Apache Commons Math 3.6.1 download page. It provides links for 'Binaries' (tar.gz and zip) and 'Source' (tar.gz and zip). The 'Archives' section lists older releases. A sidebar on the left contains links to other Apache projects like Javadoc, Issue Tracking, Source Repository, and various mathematical components.

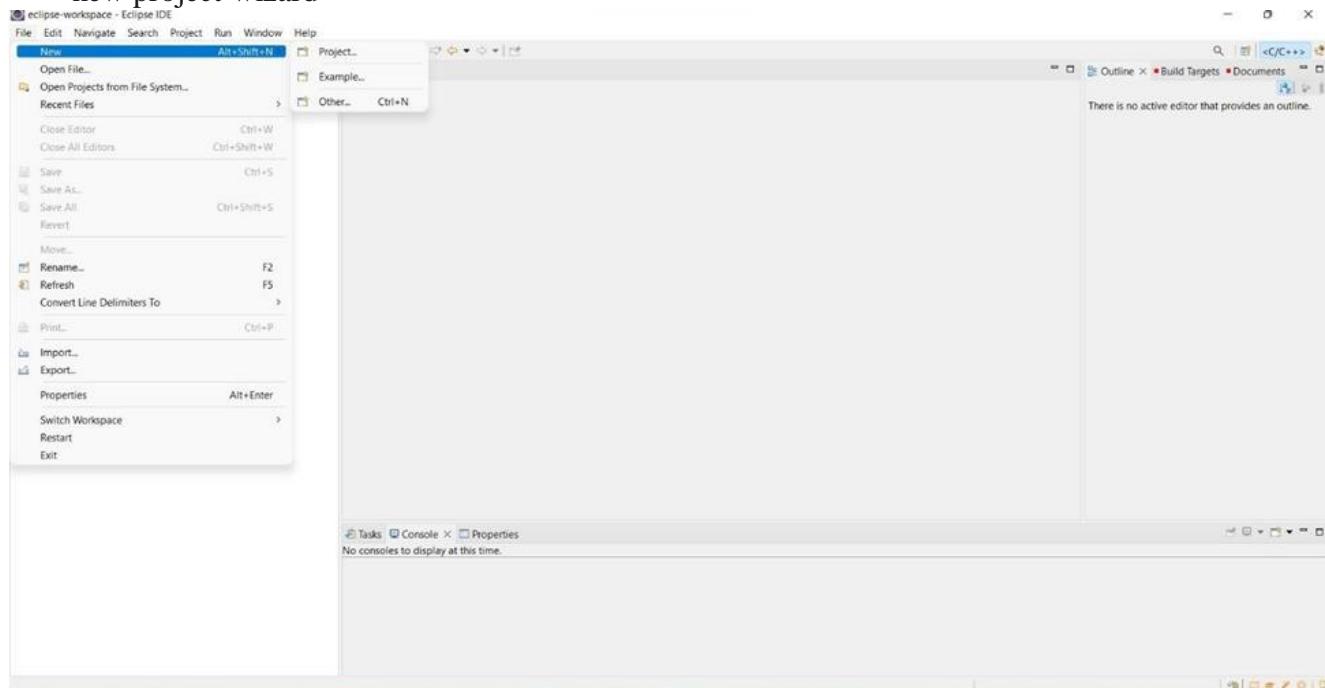
Step 4: Downloaded Eclipse, cloudsim-3.0.3 and Apache Commons Math 3.6.1 in your local machine and extract cloudsim-3.0.3 and Apache Commons Math 3.6.1

VSCodeUserSetup-x64-1.43.0.exe	3/12/2020 10:31 PM	Application	57,515 KB
Windows10Upgrade9252 (1).exe	12/1/2020 10:51 PM	Application	6,072 KB
Windows10Upgrade9252.exe	12/1/2020 10:34 PM	Application	6,072 KB
commons-math3-3.6.1-bin.zip	5/5/2022 12:01 PM	Compressed (zipp...)	21,712 KB
cloudsim-3.0.3.zip	5/5/2022 12:01 PM	Compressed (zipp...)	13,367 KB

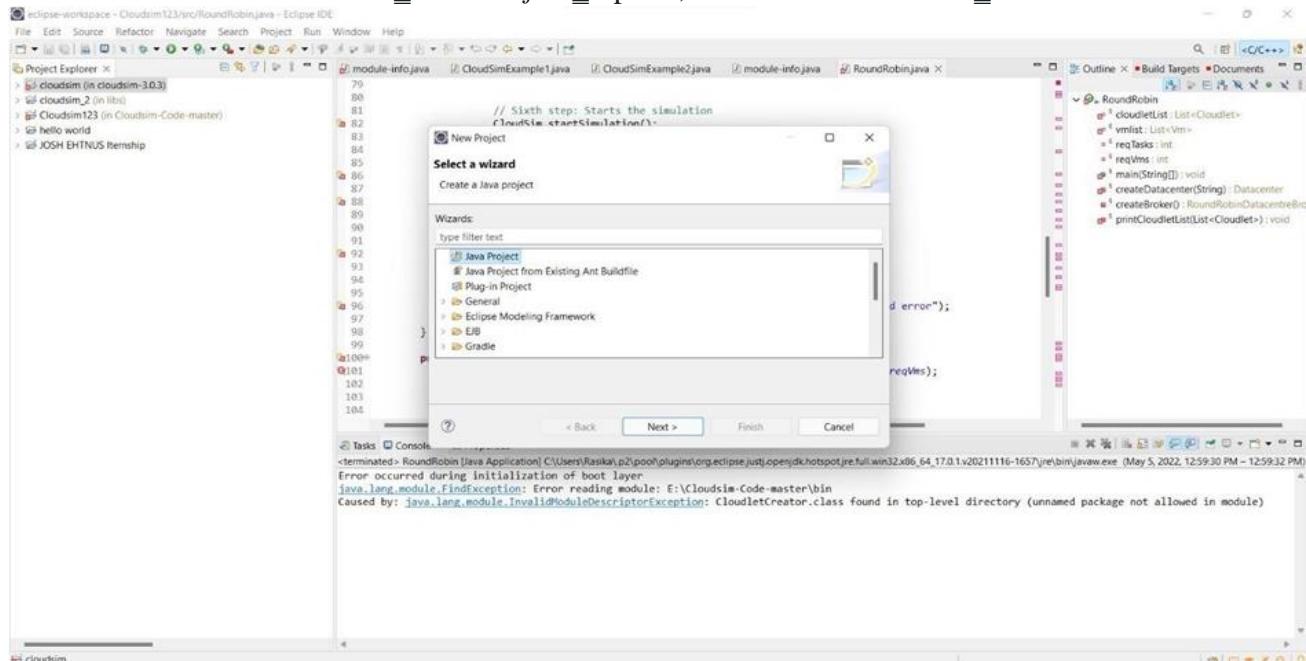
Step 5: First of all, navigate to the folder where you have unzipped the eclipse folder and open Eclipse.exe



Step 6: Now within Eclipse window navigate the menu: *File* -> *New* -> *Project*, to open the new project wizard

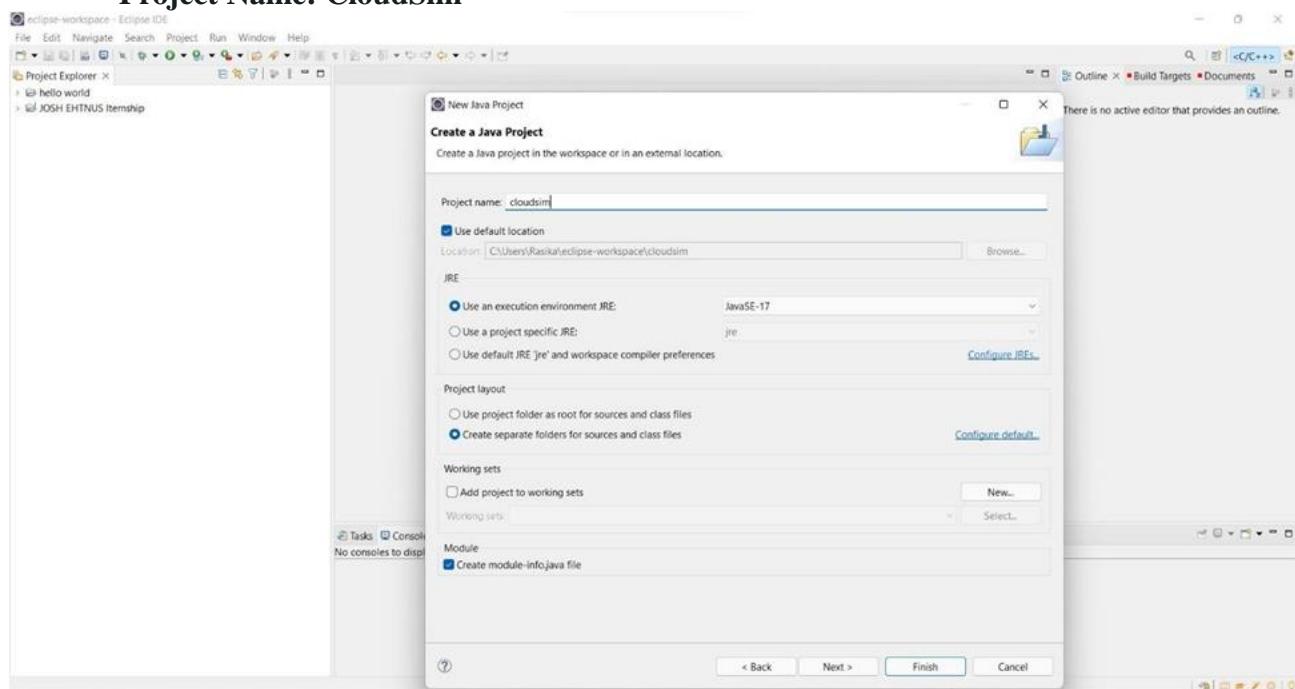


Step 7: A New Project wizard should open. There are a number of options displayed and you have to find & select the Java Project option, once done click ‘Next’

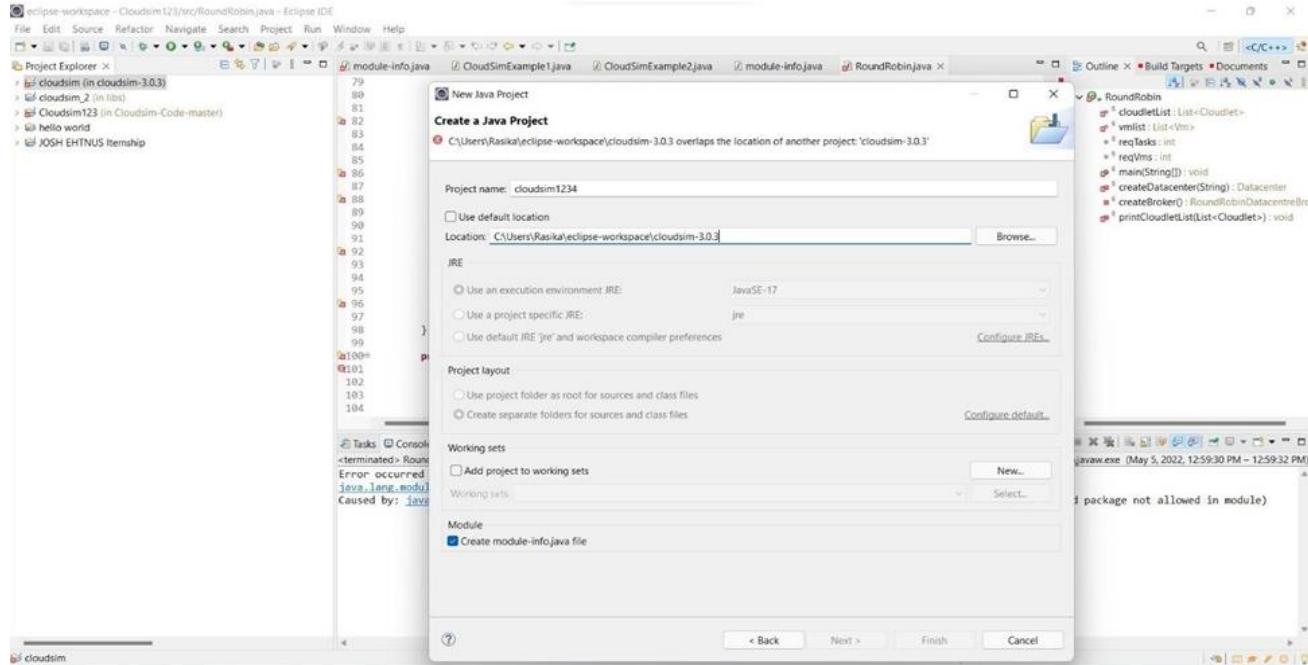


Step 8: Now a detailed new project window will open, here you will provide the project name and the path of CloudSim-master-code project source code, which will be done as follows:

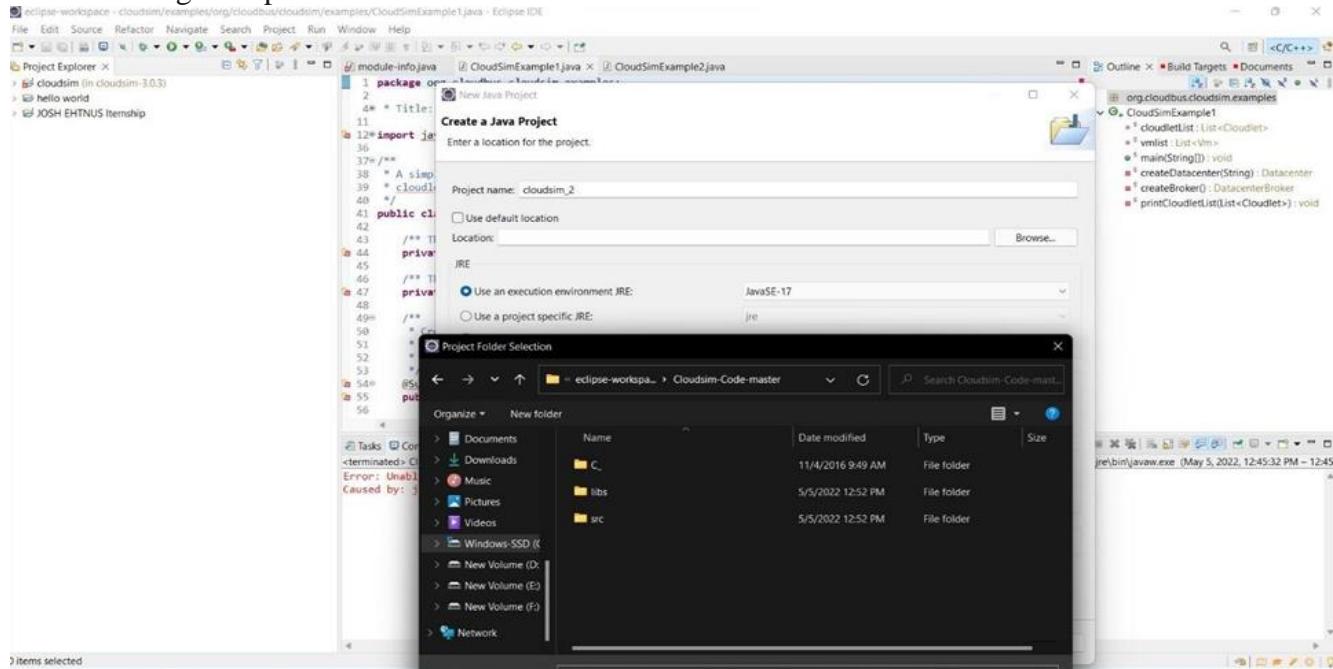
Project Name: CloudSim



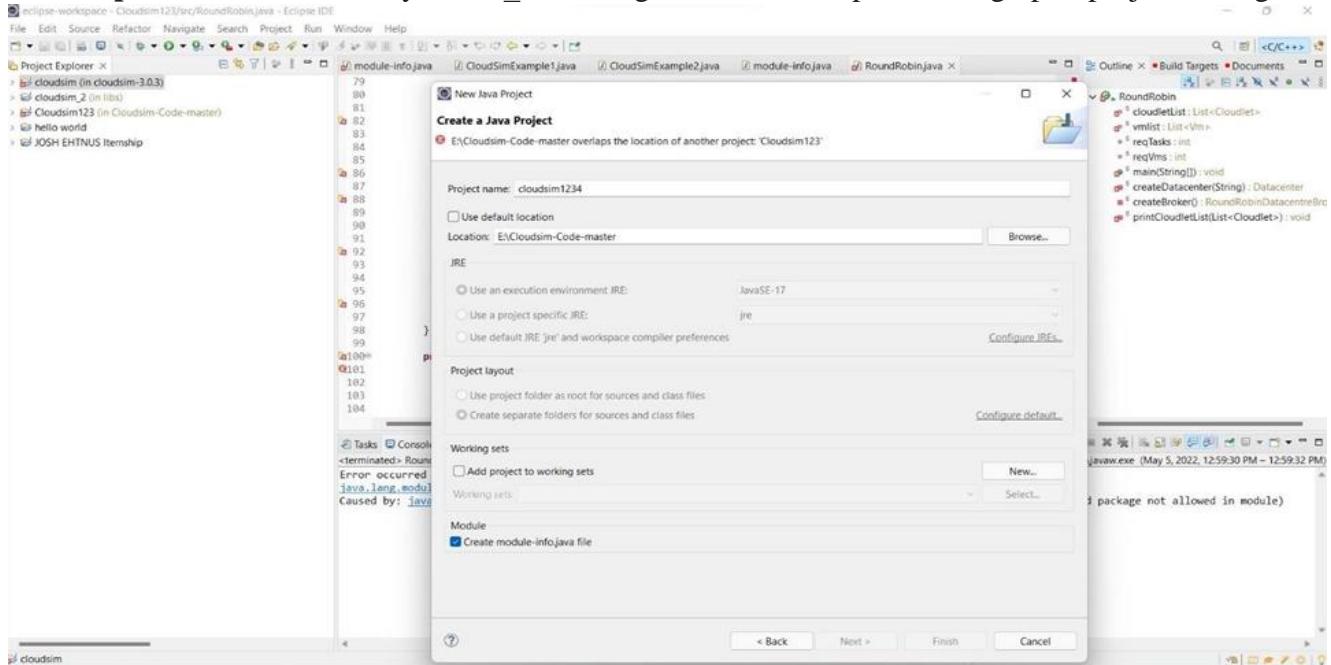
Step 9: Unselect the ‘*Use default location*’ option and then click on ‘*Browse*’ to open the path where you have unzipped the Cloudsim-code-master project and finally click Next to set project settings.



Step 10: Make sure you navigate the path till you can see the bin, docs, examplesetc folder in the navigation plane.

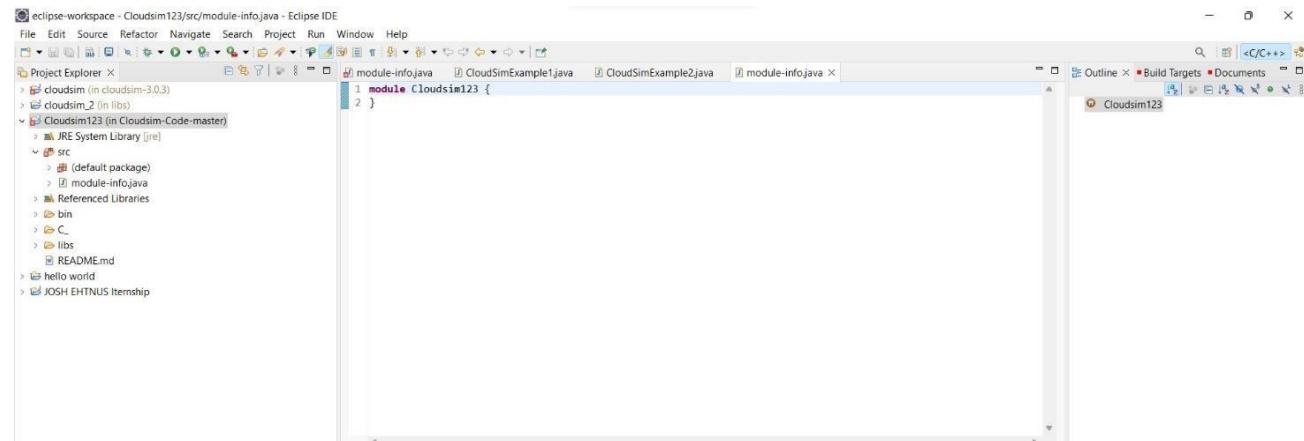


Step 11: Once done finally, click ‘Next’ to go to the next step i.e. setting up of project settings

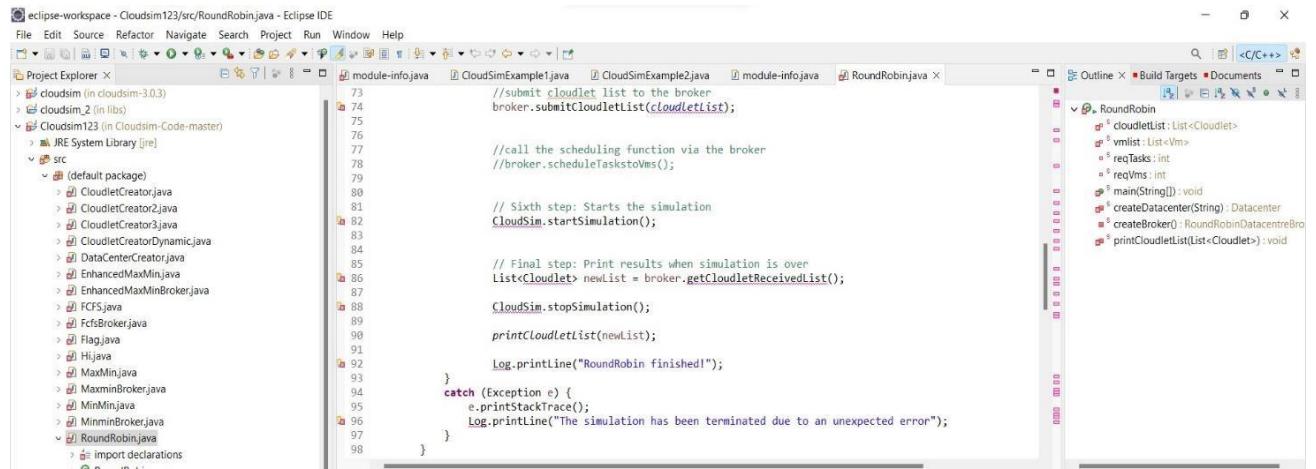


Step 12: Once the project is configured you can open the *Project Explorer* and start exploring the Cloudsim project. Also for the first time eclipse automatically start building the workspace for newly configured Cloudsim project, which may take some time depending on the configuration of the computer system.

Following is the final screen which you will see after Cloudsim is configured.



Step 13: Now just to check you within the Project Explorer, you should navigate to the _src_ folder, then expand the package _default package_ and double click to open the _RoundRobin.java_.

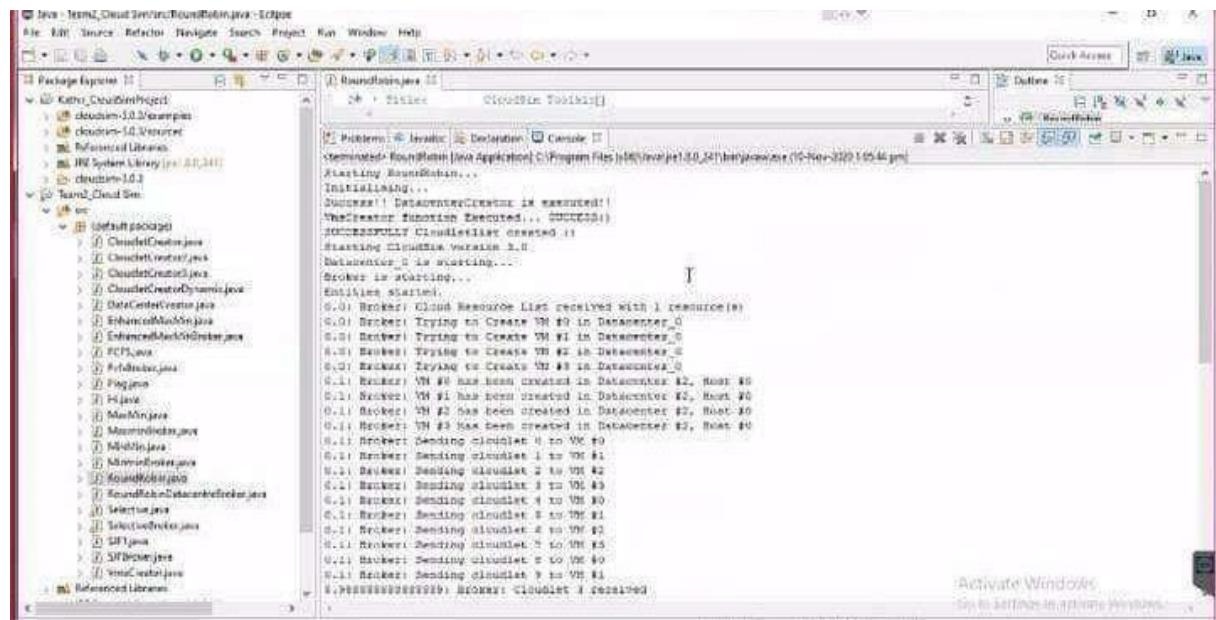


```

eclipse-workspace - Cloudsim123/src/RoundRobin.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer × CloudSimExample1.java CloudSimExample2.java module-info.java RoundRobin.java ×
CloudSim123 (in Cloudsim-Code-master)
JRE System Library [jre]
src
CloudletCreator.java
CloudletCreator2.java
CloudletCreator3.java
CloudletCreatorDynamic.java
DataCenterCreator.java
EnhancedMaxMinJava
EnhancedMaxMinBroker.java
FCFS.java
FcfsBroker.java
Flag.java
Hijava
MaxMinJava
MaxMinBroker.java
MinMinJava
MinMinBroker.java
RoundRobin.java
Import declarations
73     //submit cloudlet list to the broker
74     broker.submitCloudletList(cloudletList);
75
76     //call the scheduling function via the broker
77     //broker.scheduleTaskstoVms();
78
79     // Sixth step: Starts the simulation
80     CloudSim.startSimulation();
81
82     // Fifth step: Print results when simulation is over
83     List<Cloudlet> newList = broker.getCloudletReceivedList();
84
85     CloudSim.stopSimulation();
86
87     printCloudletList(newList);
88
89     Log.println("RoundRobin finished!");
90
91     }
92     catch (Exception e) {
93         e.printStackTrace();
94         Log.println("The simulation has been terminated due to an unexpected error");
95     }
96 }
97
98 }
}

```

Step 14: Now navigate to the Eclipse menu *Run ->Run* or directly use a keyboard shortcut '*Ctrl + F11*' to execute the '*RoundRobin.java*'. If it is successfully executed it should be displaying the following type to output in the console window of the Eclipse IDE.



```

Java - Test2_Cloud_Sim123/RoundRobin.java - Eclipse IDE
File Edit Sources Refactor Navigate Search Project Run Window Help
Package Explorer × RoundRobin.java × Problems Java Declaration Console
Starting Environment...
INITIALIZING...
Success! DatacenterCreation is executed!
TheCreate function Executed... SUCCESS!
Successfully CloudScheduler created.
Starting CloudletExecution...
Datacenter_0 is starting...
Broker is starting...
Broker started.
0.0 Broker: Client Resource List received with 1 resource(s)
0.0 Broker: Trying to Create VM #0 in Datacenter_0
0.0 Broker: Trying to Create VM #1 in Datacenter_0
0.0 Broker: Trying to Create VM #2 in Datacenter_0
0.0 Broker: Trying to Create VM #3 in Datacenter_0
0.0 Broker: VM #0 has been created in Datacenter_0, Host #0
0.0 Broker: VM #1 has been created in Datacenter_0, Host #0
0.0 Broker: VM #2 has been created in Datacenter_0, Host #0
0.0 Broker: VM #3 has been created in Datacenter_0, Host #0
0.0 Broker: Sending cloudlet 0 to VM #0
0.0 Broker: Sending cloudlet 1 to VM #1
0.0 Broker: Sending cloudlet 2 to VM #2
0.0 Broker: Sending cloudlet 3 to VM #3
0.0 Broker: Sending cloudlet 4 to VM #0
0.0 Broker: Sending cloudlet 5 to VM #1
0.0 Broker: Sending cloudlet 6 to VM #2
0.0 Broker: Sending cloudlet 7 to VM #3
0.0 Broker: Sending cloudlet 8 to VM #0
0.0 Broker: Sending cloudlet 9 to VM #1
0.0 Broker: Cloudlet 0 received
0.0 Broker: Cloudlet 1 received
0.0 Broker: Cloudlet 2 received
0.0 Broker: Cloudlet 3 received
0.0 Broker: Cloudlet 4 received
0.0 Broker: Cloudlet 5 received
0.0 Broker: Cloudlet 6 received
0.0 Broker: Cloudlet 7 received
0.0 Broker: Cloudlet 8 received
0.0 Broker: Cloudlet 9 received

```

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the project structure for "Kater_CloudSimProject". It includes subfolders like "cloudsim-3.0-examples", "cloudsim-3.0-Vmware", and "cloudsim-3.0".
- Terminal:** The terminal window displays the execution of RoundRobin jobs. It shows the creation of Cloudlets, their execution on VMs, and the shutdown of the simulation.
- Output:** The output window shows a table of completed Cloudlet executions with columns: Cloudlet ID, Status, Datacenter ID, VM ID, Time, Start Time, Finish Time, and waiting time.

Cloudlet ID	Status	Datacenter ID	VM ID	Time	Start Time	Finish Time	Waiting time
3	SUCCESS	2	3	8.95	0.1	8.95	0
7	SUCCESS	2	3	12.22	0.1	12.22	0
5	SUCCESS	2	1	13.6	0.1	13.6	0
6	SUCCESS	2	2	14.66	0.1	14.76	0
8	SUCCESS	2	1	20.5	0.1	20.6	0
2	SUCCESS	2	3	31.33	0.1	34.83	0
9	SUCCESS	2	0	35.15	0.1	35.27	0
4	SUCCESS	2	0	35.55	0.1	40.35	0
1	SUCCESS	2	1	41.45	0.1	41.55	0
0	SUCCESS	2	0	47.27	0.1	47.37	0

Result:

Thus the scheduling algorithm is executed in cloudsim is simulated using Eclipse Environment successfully.

Aim:

To procedure File Transfer in Client & Server using virtual machine

Steps: Steps to perform File Transfer in Client & Server using virtual machine.

Step 1: Open a virtual machine to do file transfer.

Step 2: Write the java program for FTP Client and FTP Server.

Step 3: Run the program.

Source Code:**FTPClient.java**

```
import java.io.*; import java.net.*; import java.util.*; public class  
FTPClient{ public static void main(String args[])throws  
IOException { try { int number;  
        Socket s=new Socket("127.0.0.1",10087);  
        Scanner sc=new Scanner(System.in);  
        System.out.println("Enter the file name:");  
        String fn=sc.next();  
        DataOutputStream dos=new DataOutputStream(s.getOutputStream());  
        dos.writeUTF(fn);  
        DataInputStream dis=new DataInputStream(s.getInputStream());  
        String input=(String)dis.readUTF();  
        FileInputStream fis=new FileInputStream(input);  
        System.out.println("Even Numbers in the" +fn+" are");  
        int i=0;  
        while((i=fis.read())!=-1){  
            System.out.println((char)i);  
        }  
        s.close(); }  
    catch(Exception e){  
        System.out.println("Port not available "+e); }  
}
```

FTPServer.java

```
import java.io.*; import java.net.*; import java.util.*; public class  
FTPServer{ public static void main(String args[])throws  
IOException{ try{ int num;  
        Scanner sc=new Scanner(System.in);  
        ServerSocket ss=new ServerSocket(10087);  
        Socket s=ss.accept();  
        System.out.println("Waiting. ....");  
        DataInputStream dis=new DataInputStream(s.getInputStream());  
        String input=(String)dis.readUTF();  
        DataOutputStream dos=new DataOutputStream(s.getOutputStream());  
        FileInputStream fis = new FileInputStream("out.txt");  
        FileOutputStream fos = new  
        FileOutputStream(input); while((num=fis.read())!= -  
1) { if(num%2==0) { fos.write(num);  
        }  
        }  
        dos.writeUTF(input);  
        System.out.println("File is sent to  
client"); ss.close(); s.close(); }  
catch(Exception e) {  
        System.out.println("Port not available"+e); }  
    }  
}
```

Out.txt

1 2
3
4 5
6
7 8
9

Output:

A screenshot of a Linux desktop environment. A terminal window titled "Terminal" is open, showing the command line interface. The terminal window has a dark purple background and contains the following text:

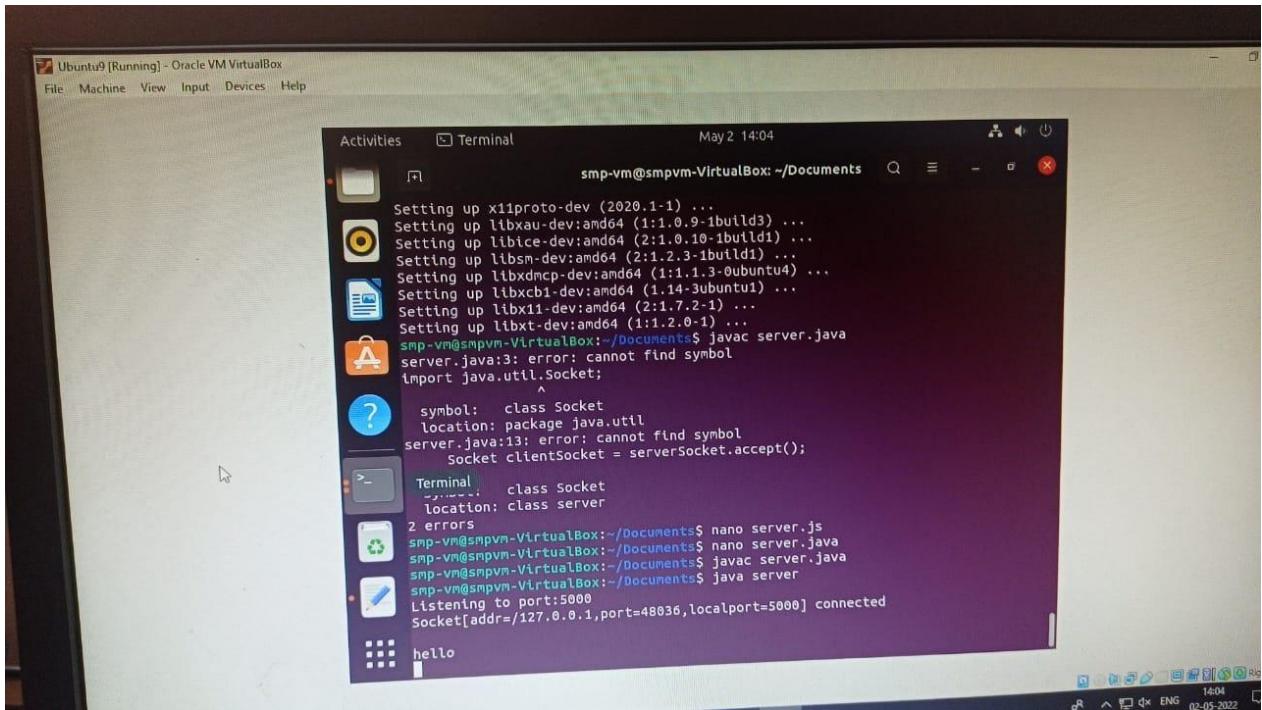
```
smp-vm@smppvm-VirtualBox: ~/Documents
done.
done.
Processing triggers for sgml-base (1.30) ...
Setting up x11proto-dev (2020.1-1) ...
Setting up libxau-dev:amd64 (1:1.0.9-1build3) ...
Setting up libice-dev:amd64 (2:1.0.10-1build1) ...
Setting up libsm-dev:amd64 (2:1.2.3-1build1) ...
Setting up libxdmcp-dev:amd64 (1:1.1.3-0ubuntu4) ...
Setting up libxcb1-dev:amd64 (1.14-3ubuntu1) ...
Setting up libx11-dev:amd64 (2:1.7.2-1) ...
Setting up libxt-dev:amd64 (1:1.2.0-1) ...
smp-vm@smppvm-VirtualBox:~/Documents$ javac server.java
server.java:3: error: cannot find symbol
import java.util.Socket;
                           ^
symbol:   class Socket
location: package java.util
server.java:13: error: cannot find symbol
    Socket clientSocket = serverSocket.accept();
                           ^
symbol:   class Socket
location: class server
2 errors
smp-vm@smppvm-VirtualBox:~/Documents$ nano server.js
smp-vm@smppvm-VirtualBox:~/Documents$ nano server.java
smp-vm@smppvm-VirtualBox:~/Documents$ javac server.java
smp-vm@smppvm-VirtualBox:~/Documents$ java server
smp-vm@smppvm-VirtualBox:~/Documents$ Listening to port:5000
```

The terminal window is titled "Terminal" and shows the date and time as May 2 14:02. The desktop interface includes a dock with various icons at the bottom.

A screenshot of a Linux desktop environment. A terminal window titled "Terminal" is open, showing the command line interface. The terminal window has a dark purple background and contains the following text:

```
smp-vm@smppvm-VirtualBox: ~/Documents
smp-vm@smppvm-VirtualBox: $ cd Documents
smp-vm@smppvm-VirtualBox: ~/Documents$ ls
client.java
smp-vm@smppvm-VirtualBox: ~/Documents$ cd
smp-vm@smppvm-VirtualBox: $ cd Documnets
bash: cd: Documnets: No such file or directory
smp-vm@smppvm-VirtualBox: $ cd Documents
smp-vm@smppvm-VirtualBox: ~/Documents$ nano client.java
smp-vm@smppvm-VirtualBox: ~/Documents$ javac client.java
smp-vm@smppvm-VirtualBox: ~/Documents$ java client
input> hello
input>
```

The terminal window is titled "Terminal" and shows the date and time as May 2 14:04. The desktop interface includes a dock with various icons at the bottom.



Result:

Thus the program to the File transfer operation using virtual machine was successfully executed and verified.

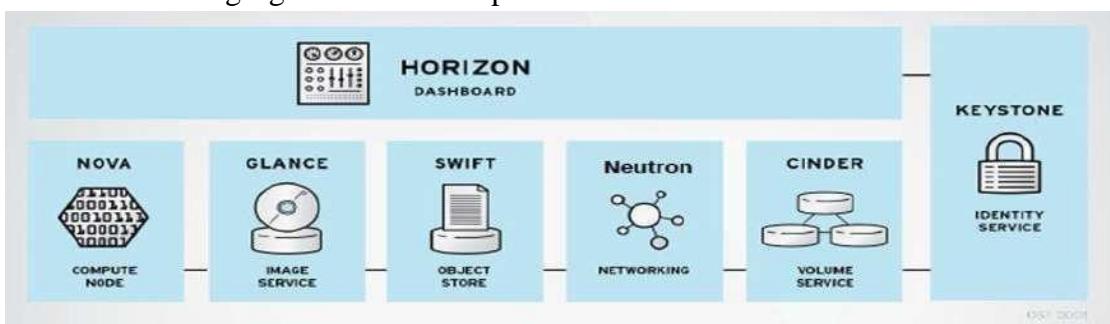
Ex No. 5

Find a procedure to launch virtual machine using Openstack

Introduction:

- † OpenStack was introduced by Rackspace and NASA in July 2010.
- † OpenStack is an Infrastructure as a Service known as Cloud Operating System, that take resources such as Compute, Storage, Network and Virtualization Technologies and control those resources at a data center level
- † The project is building an open source community - to share resources and technologies with the goal of creating a massively scalable and secure cloud infrastructure.
- † The software is open source and limited to just open source APIs such as Amazon.

The following figure shows the OpenStack architecture



OpenStack architecture

- It is modular architecture
- Designed to easily scale out
- Based on (growing) set of core services **The major components are**

1. **Keystone**
2. **Nova**
3. **Glance**
4. **Swift**
5. **Quantum**
6. **Cinder**

- **KEYSTONE :**
 - Identity service ○ Common authorization framework
 - Manage users, tenants and roles
 - Pluggable backends (SQL,PAM,LDAP, IDM etc)
- **NOVA**
 - Core compute service comprised of
 - ◆ Compute Nodes – hypervisors that run virtual machines
 - Supports multiple hypervisors KVM,Xen,LXC,Hyper-V and ESX
 - ◆ Distributed controllers that handle scheduling, API calls, etc
 - Native OpenStack API and Amazon EC2 compatible API
- **GLANCE**
 - Image service ○ Stores and retrieves disk images (Virtual machine templates) ○ Supports RAW,QCOW,VHD,ISO,OVF & AMI/AKI
 - Backend Storage : File System, Swift, Gluster, Amazon S3
- **SWIFT**
 - Object Storage service ○ Modeled after Amazon's Service
 - Provides simple service for storing and retrieving arbitrary data ○ Native API and S3 compatible API
- **NEUTRON**
 - Network service
 - Provides framework for Software Defined Network ○ Plugin architecture
 - ◆ Allows integration of hardware and software based network solutions
 - Open vSwitch, Cisco UCS,Standard Linux Bridge,NiCira NVP



CINDER

- Block Storage (Volume) service ◦ Provides block storage for Virtual machines(persistent disks) ◦ Similar to Amazon EBS service ◦ Plugin architecture for vendor extensions
 - ◆ NetApp driver for cinder

HORIZON

- Dashboard
- Provides simple self service UI for end-users ◦ Basic cloud administrator functions
 - ◆ Define users, tenants and quotas
 - ◆ No infrastructure management

HEAT OpenStack Orchestration

- Provides template driven cloud application orchestration ◦ Modeled after AWS Cloud Formation ◦ Targeted to provide advanced functionality such as high availability and auto scaling
- Introduced by Redhat
- **CEILOMETER** – OpenStack Monitoring and Metering ◦ Goal: To Provide a single infrastructure to collect measurements from an entire OpenStack Infrastructure; Eliminate need for multiple agents attaching to multiple OpenStack Projects ◦ Primary targets metering and monitoring: Provided extensibility

† Steps in Installing Openstack Step 1:

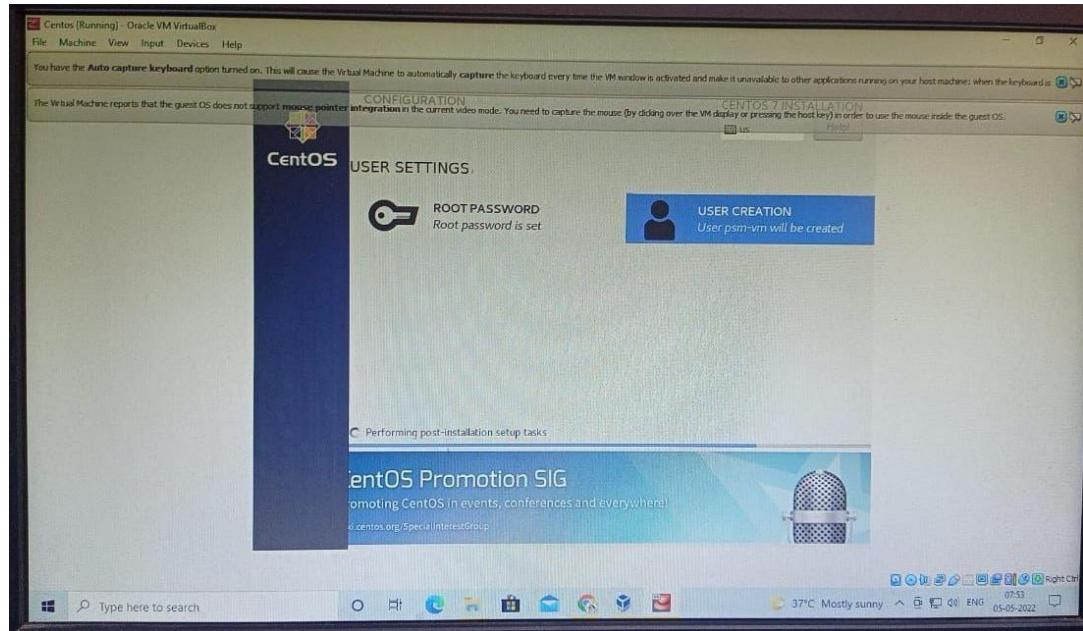
- Download and Install Oracle Virtual Box latest version & Extension package
 - <https://virtualbox.org/wiki/downloads>

Step 2:

- Download CentOS 7 OVA(Open Virtual Appliance) from ◦ Link : <https://linuxvmimages.com/images/centos-7>
- Import CentOS 7 OVA(Open Virtual Appliance) into Oracle Virtual Box



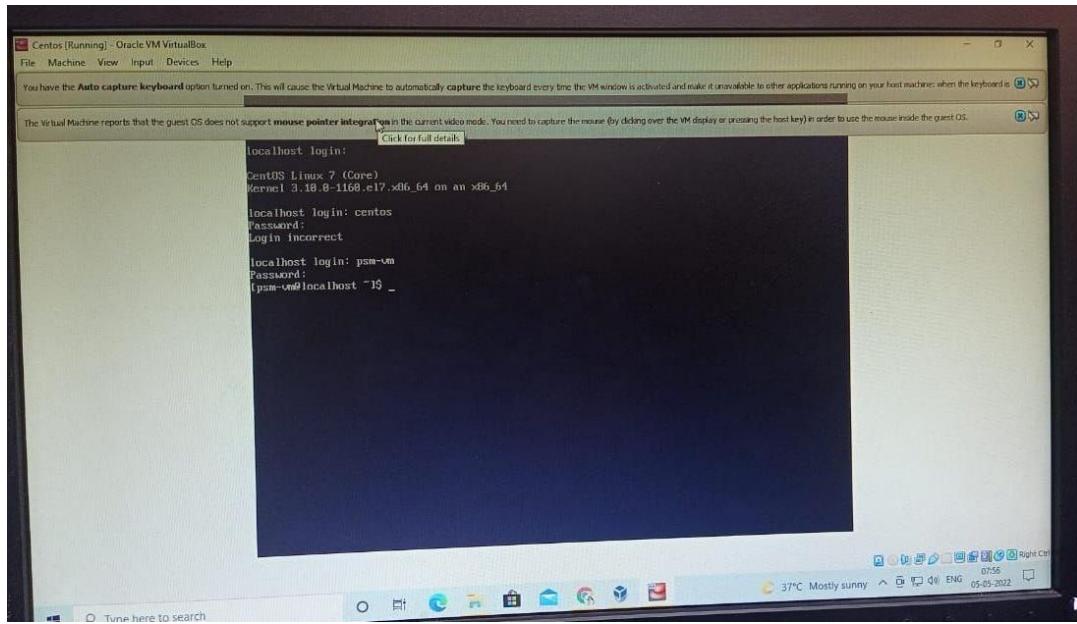
1. Create a Virtual Machine on your VM Ware or Oracle Virtual Box.



Step 3: Login into CenOS 7

- Login Details ○ **User name : centos ○ Password : centos**
- To change into root user in Terminal

#sudosu-



Step 4: Installation Steps for OpenStack

Step5: Command to disable and stop firewall

systemctl disable firewalld

#systemctl stop firewalld

```
You have the Auto capture keyboard option turned on. This will cause the Virtual Machine to automatically capture the keyboard every time the VM window is activated and make it unavailable to other applications running on your host machine; when the keyboard is captured, you will not be able to type in other windows.
```

```
The Virtual Machine reports that the guest OS does not support mouse pointer integration in the current video mode. You need to capture the mouse (by clicking over the VM display or pressing the host key) in order to use the mouse inside the guest OS.
```

```
localhost login: centos
CentOS Linux 7 (Core)
Kernel 3.10.0-1160.el7.x86_64 on an x86_64
localhost login: centos
Password:
Login incorrect
localhost login: psm-vm
Password:
[psm-vm@localhost ~]$ systemctl stop firewalld
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ===
Authentication is required to manage system services or units.
Authenticating as: root
Password:
==== AUTHENTICATION COMPLETE ===
[psm-vm@localhost ~]$
```

Step 6: Command to disable and stop Network Manager

systemctl disable NetworkManager

systemctl stop NetworkManager

```
You have the Auto capture keyboard option turned on. This will cause the Virtual Machine to automatically capture the keyboard every time the VM window is activated and make it unavailable to other applications running on your host machine; when the keyboard is captured, you will not be able to type in other windows.
```

```
The Virtual Machine reports that the guest OS does not support mouse pointer integration in the current video mode. You need to capture the mouse (by clicking over the VM display or pressing the host key) in order to use the mouse inside the guest OS.
```

```
localhost login: centos
CentOS Linux 7 (Core)
Kernel 3.10.0-1160.el7.x86_64 on an x86_64
localhost login: centos
Password:
Login incorrect
localhost login: psm-vm
Password:
[psm-vm@localhost ~]$ systemctl stop firewalld
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ===
Authentication is required to manage system services or units.
Authenticating as: root
Password:
==== AUTHENTICATION COMPLETE ===
[psm-vm@localhost ~]$ systemctl stop NetworkManager
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ===
Authentication is required to manage system services or units.
Authenticating as: root
Password:
==== AUTHENTICATION COMPLETE ===
[psm-vm@localhost ~]$
```

Step 7: Enable and start Network

```
#systemctl enable network
```

```
#systemctl start network
```

```
[root@localhost ~]# systemctl enable network
network.service is not a native service, redirecting to /sbin/chkconfig.
Executing /sbin/chkconfig network on
[root@localhost ~]# systemctl start network
[root@localhost ~]#
```

Step 8: OpenStack will be deployed on your Node with the help of **PackStack** package provided by **rdo** repository (**RPM Distribution of OpenStack**).In order to enable **rdo** repositories on Centos **7** run the below command.

```
#yum install -y https://rdoproject.org/repos/rdo-release.rpm
```

```
[root@localhost ~]# yum install -y centos-release-openstack-newton
```

Step 9: Update Current packages

```
#yum update -y
```

```
[root@localhost ~]# yum update -y
Loaded plugins: fastestmirror, langpacks
centos-ceph-jewel
centos-openstack-newton
centos-qemu-ev
(1/3): centos-ceph-jewel/7/x86_64/primary_db | 2.9 kB 00:00:00
(2/3): centos-qemu-ev/7/x86_64/primary_db | 2.9 kB 00:00:00
(3/3): centos-openstack-newton/x86_64/primary_db | 2.9 kB 00:00:00
63 kB 00:00:01
52 kB 00:00:00
853 kB 00:00:02
Loading mirror speeds from cached hostfile
 * base: centos.excellmedia.net
 * extras: centos.excellmedia.net
 * updates: mirrors.viethosting.com
```

Step 10:Install OpenStack Release for CentOS

#yum install -y openstack-packstack

```
[root@localhost ~]# yum install -y openstack-packstack
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: centos.excellmedia.net
 * extras: centos.excellmedia.net
 * updates: mirrors.viethosting.com
```

Step 11:Start packstack to install OpenStack Newton

#packstak --allinone

```
[root@localhost ~]# packstack --allinone
Welcome to the Packstack setup utility

The installation log file is available at: /var/tmp/packstack/20170314-065810-b8cxch/openstack-setup.log
Packstack changed given value to required value /root/.ssh/id_rsa.pub

Installing:
Clean Up [ DONE ]
Discovering ip protocol version [ DONE ]
Setting up ssh keys [ DONE ]
Preparing servers [ DONE ]
Pre installing Puppet and discovering hosts' details [ DONE ]
Preparing pre-install entries [ DONE ]
Setting up CACERT [ DONE ]
Preparing AMQP entries [ DONE ]
Preparing MariaDB entries [ DONE ]
Fixing Keystone LDAP config parameters to be undef if empty[ DONE ]
Preparing Keystone entries [ DONE ]
Preparing Glance entries [ DONE ]
Checking if the Cinder server has a cinder-volumes vg[ DONE ]
Preparing Cinder entries [ DONE ]
Preparing Nova API entries [ DONE ]
[ DONE ]
```

SUBSCRIBE

Step 12: Note the user name and password from kestonerc_admin

#cat kestonerc_admin

```
[root@localhost ~]# ls
anaconda-ks.cfg      kestonerc_admin  packstack-answers-20170314-065812.txt
initial-setup-ks.cfg  kestonerc_demo
[root@localhost ~]# cat kestonerc_admin
unset OS_SERVICE_TOKEN
export OS_USERNAME=admin
export OS_PASSWORD=cdc897f8cb7f4dda
export OS_AUTH_URL=http://10.0.2.15:5000/v2.0
export PS1='[\u@\h \W(keystone_admin)]\$ '

export OS_TENANT_NAME=admin
export OS_REGION_NAME=RegionOne
[root@localhost ~]#
```

Step 13: Click the URL and enter the user name and password to start OpenStack



OpenStack is successfully launched in your machine





Result:

Thus the OpenStack Installation is executed successfully.

Title: Design an Assignment to retrieve, verify, and store user credentials using Firebase Authentication, the Google App Engine standard environment, and Google Cloud Data store

Before you begin

Step-1: Install Git,

[Add](#) [apt](#) [git](#)

Step-II: Install Python 2.7

Step-III: Sign in to your Google Account

If you don't already have one, sign up for a new account.

Step-IV: Select or create a GCP project

Note: If you don't plan to keep the resources you create in this tutorial, create a new project instead of selecting an existing project. After you finish, you can delete the project, removing all resources associated with the project and tutorial.

GO TO THE MANAGE RESOURCES PAGE

Step-V: Install and initialize the Cloud SDK.

If you have already installed and initialized the SDK to a different project, set the gcloud project to the App Engine project ID you're using for Firenotes. See Managing Cloud SDK Configurations for specific commands to update a project with the gcloud tool.

Step-VI: Cloning the sample app

To download the sample to your local machine:

Clone the sample application repository to your local machine:

```
git clone https://github.com/GoogleCloudPlatform/python-docs-samples.git
```

Alternatively, you can download the sample as a zip file and extract it.

Navigate to the directory that contains the sample code:

```
cd python-docs-samples/appengine/standard.firebaseio/firenotes
```

Step-VII: Adding the Firebase Authentication user interface

To configure FirebaseUI and enable identity providers:

1. Add Firebase to your app by following these steps:
 - a. Create a Firebase project in the [Firebase console](#).

If you don't have an existing Firebase project, click **Add project** and enter either an existing Google Cloud Platform project name or a new project name.

If you have an existing Firebase project that you'd like to use, select that project from the console.

- b. From the project overview page, click **Add Firebase to your web app**. If your project already has an app, select **Add App** from the project overview page.
Use the Initialize Firebase section of your project's customized code snippet to fill out the following section of the frontend/main.js file:
- c.

GO to given Path [appengine/standard.firebaseio/firenotes/frontend/main.js](#)

[appengine/standard.firebaseio.firenotes.frontend.main.js](#)

[VIEW ON GITHUB](#)

```
// Obtain the following from the "Add Firebase to your web app" dialogue
// Initialize Firebase
  fig =
apiKey: <API KEY>
authDomain: "<ID>.firebaseapp.com",
databaseURL: "https://<DATABASE NAME>.firebaseio.com",
projectId: <ID>
storageBucket: "<BUCKET>.appspot.com",
messagingSenderId: "<MESSAGING_SENDER_ID>"
```

2. Edit the **backend/app.yaml** file and enter your Firebase project ID in the environment

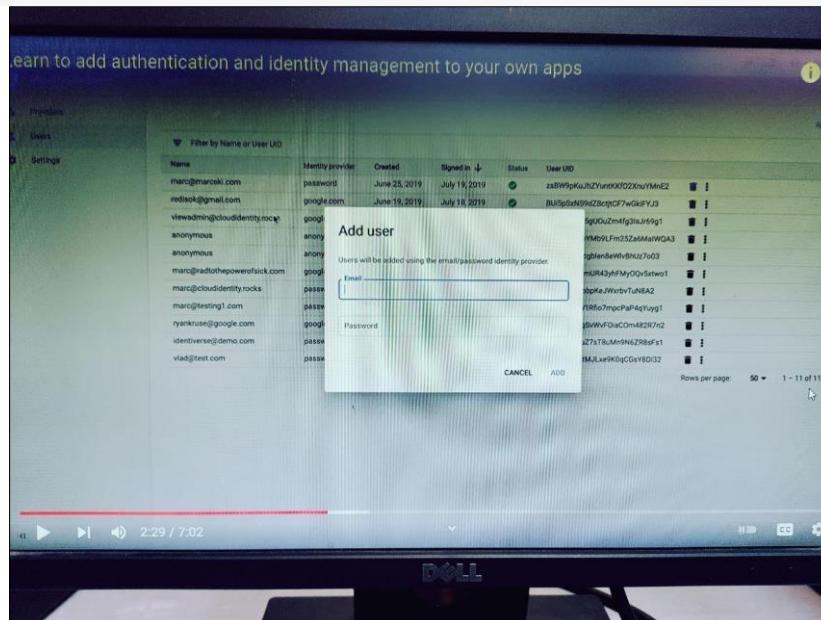
Firebase project ID in the environment

[VIEW ON GITHUB](#)

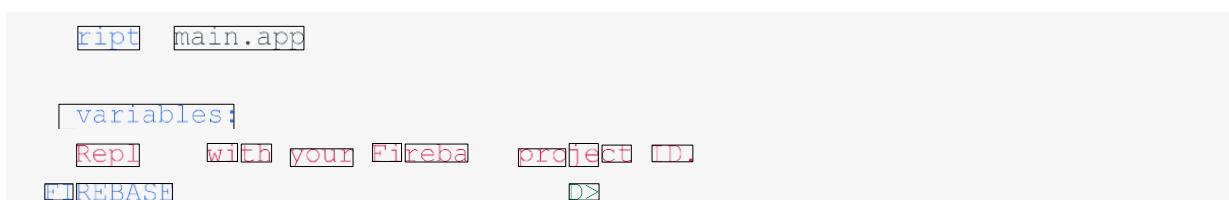
```
runtime
  threadsafe
    env
      backend

handlers:
- url: /.*

```

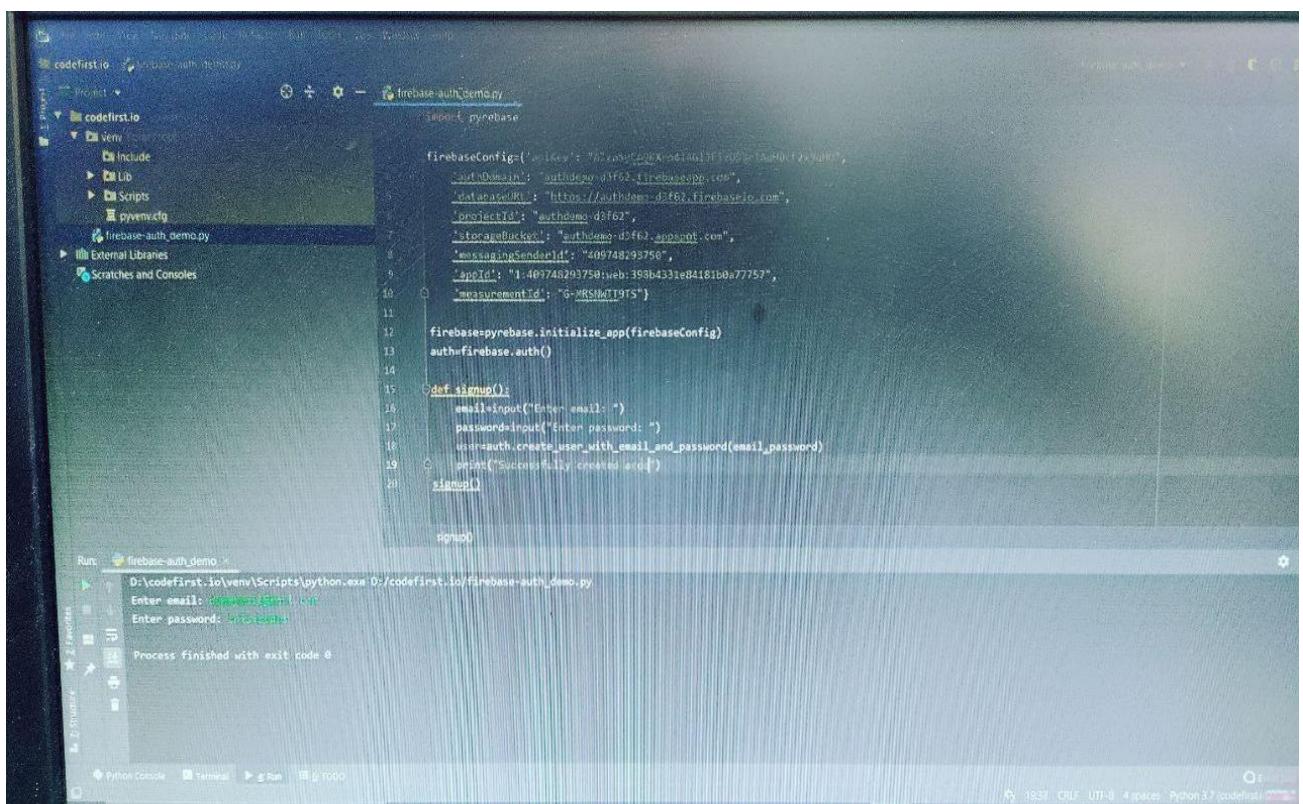


This screenshot shows the 'Edit Edition / Password provider' settings page. It includes sections for 'Enabled' (which is checked), 'Configure email/password' (with options for 'Passwords' and 'Allow passwordless login'), 'Templates' (for bulk import), and 'Import users' (with a 'VIEW PARAMETERS' link). On the right, there are 'Project Settings' for 'Authorized Domains' (listing gcp-mktg-env.appspot.com, gcp-mktg-env.firebaseio.com, gcp-mktg-env.web.app, and localhost) and 'Custom SMTP settings' (with an 'Enable' checkbox).



In the **frontend/main.js** file, configure the

3. providers you want to offer your users. [FirebaseUI login widget](#) by selecting which



4. Enable the providers you have chosen to keep in the [Firebase console](#) by clicking **Authentication > Sign-in method**. Then, under Sign-in providers, hover the cursor over a provider and click the pencil icon.

The screenshot shows the Firebase console's Authentication section. On the left sidebar, 'Authentication' is selected under the 'Develop' heading. The main page displays a table of users. There is one entry for a user with the identifier 'dummy@mail.com' and provider 'Email'. The user was created and signed in on June 3, 2020. The User UID is listed as 'JcC7dLBjBfjTNNGy1feoJuQ5C2'. At the top right, there are buttons for 'Add user' and other account management options. Below the table, pagination controls show '1 item per page' and '1-1 of 1'.

Identifier	Provider	Created	Signed In	User UID
dummy@mail.com	Email	Jun 3, 2020	Jun 3, 2020	JcC7dLBjBfjTNNGy1feoJuQ5C2

The screenshot shows the 'Providers' section of the Firebase Authentication console. It lists various sign-in methods: Anonymous, Email / Password, Facebook, GitHub, Google, LinkedIn, Microsoft, OIDC demo, Phone, Play Games, SAML demo, Twitter, and Yahoo. Each method has an 'Enabled' toggle switch, which is turned on for most providers except LinkedIn, Microsoft, and Phone.

- Toggle the **Enable** button and, for third-party identity providers, enter the provider ID and secret from the provider's developer site. The Firebase docs give specific instructions in the "Before you begin" sections of the [Facebook](#), [Twitter](#), and [GitHub](#) guides. After enabling a provider, click **Save**.

This screenshot shows the configuration dialog for the Facebook provider. It includes fields for 'App ID' and 'App secret', both of which are currently empty. An 'Enable' toggle switch is turned on. Below the fields, there is a note about adding an OAuth redirect URI to the Facebook app configuration, followed by a text input field containing the URL `https://test-84e93.firebaseio.com/_/auth/handler`. At the bottom right are 'CANCEL' and 'SAVE' buttons.

- In the Firebase console, under **Authorized Domains**, click **Add Domain** and enter the domain of your app on App Engine in the following format:

[PROJECT_ID].appspot.com

Do not include http:// before the domain name.

Reference:

<https://cloud.google.com/appengine/docs/standard/python/authenticating-users-firebase-appengine>

