Configuration Manual

MSc Research Project

Cloud Computing

Sachin Reddy Chidurala

Student ID: X22140476

School of Computing

National College of Ireland

Supervisor: Dr. Punit Gupta

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Sachin Reddy Chidurala		
Student ID:	X22140476		
Programme:	Cloud Computing	**Year:**	2023
Module:	Msc Research Project		
Lecturer:	Dr. Punit Gupta		
Submission Due Date:	14-12-2023		
Project Title:	Strategic Management of Multi Cloud Adoption: A Framework for Decision-making and Governance		
Word Count:	836 **Page Count:** 12		

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Sachin Reddy Chidurala

Date: 14-12-2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Sachin Reddy Chidurala
Student ID: X22140476

1 Introduction

This article is a detailed guide on setting up the Cloud sim simulator for container-based simulations. It explains the tools and technology needed for the setup. The article also covers important pre-requirements and steps for using the simulator effectively. It focuses on both the technical aspects and how to use the simulator.

2 Tools/Technologies Pre-requirements

Here are some of the essential conditions/pre-requirements for the setup.

- JavaSE - 1.8
- Eclipse IDE
- Cloudsim - cloudsim-4.0.

3 Deployment Configuration

Before running simulations with the Cloudsim tool on a Mac, there are several preparatory steps to follow, particularly concerning Java installation, as Cloudsim is a Java-based application:

3.1 Installation of Java

Java is essential for running Cloudsim since it's a Java application. To create and manage your Cloudsim projects effectively, you can use Eclipse, a popular Java IDE. Eclipse is not just an IDE but also a platform for developing IDE plugins and rich client applications.

3.1.1 Using Eclipse as an IDE for Java

Eclipse is well-suited for Java development and is recommended for running simulations in Cloudsim. It provides various tools and features that make coding, debugging, and testing Java applications more manageable.

3.1.2 Downloading Java

To get started, you need to download and install the Java Development Kit (JDK). You can download JDK from the following URL:

- Java Download Link: https://www.oracle.com/java/technologies/downloads/

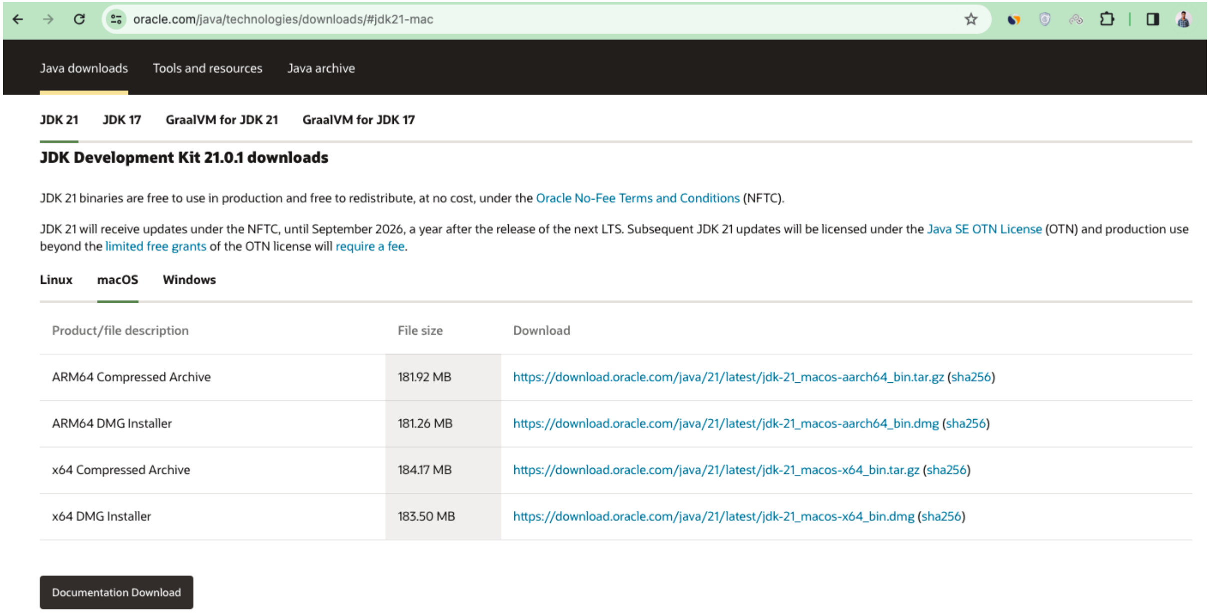

Figure 1: Java JDK download

After downloading the JDK file, Install the JDK file.

Figure 2: Java JDK Installation

After Installing Java, Open the terminal and check for the Java version by typing Java -version in the terminal.

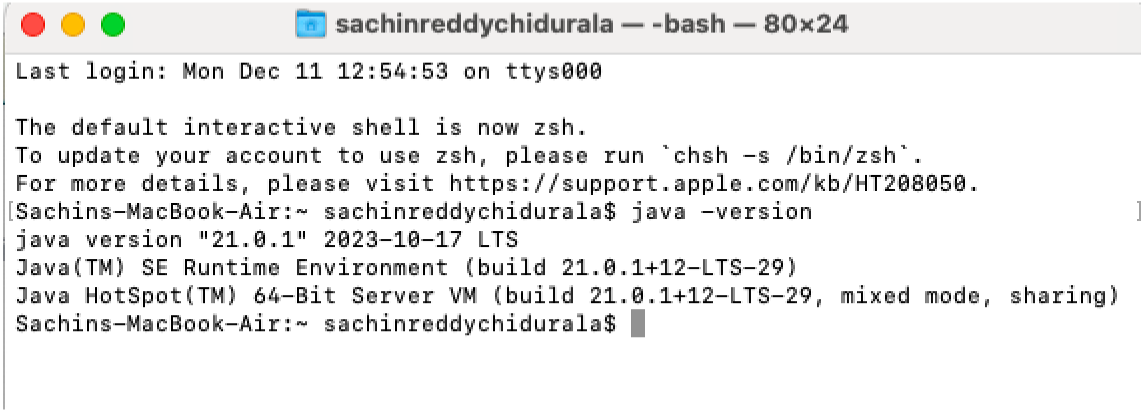

Figure 3: Java version check in the terminal

Once Java is installed on the Mac, proceed to set up Eclipse and configure it for the Cloudsim projects. Ensure that the Java version downloaded is compatible with Cloudsim and the Eclipse version.

3.2 Installation of Eclipse IDE

To use Eclipse on the Mac OS, the system must meet certain criteria.

Memory	Processor	File Size:	OS
4GB	Intel Core, Apple Silicon	30 GB	MacOS 10.9 or later

Table 1: System Requirements for Eclipse IDE

Downloading Eclipse:

- Visit the Eclipse download page (https://www.eclipse.org/downloads/.
- Choose the Eclipse IDE suitable for your development needs (e.g., Eclipse IDE for Java Developers).
- Download the macOS version.

Figure 4: Eclipse IDE for Java Developers download

After downloading the eclipse, unzip the file and install the file. When trying to Launch it for the first time the Mac will ask "are you sure you want to open eclipse" since it is downloaded from the internet. Press Open.

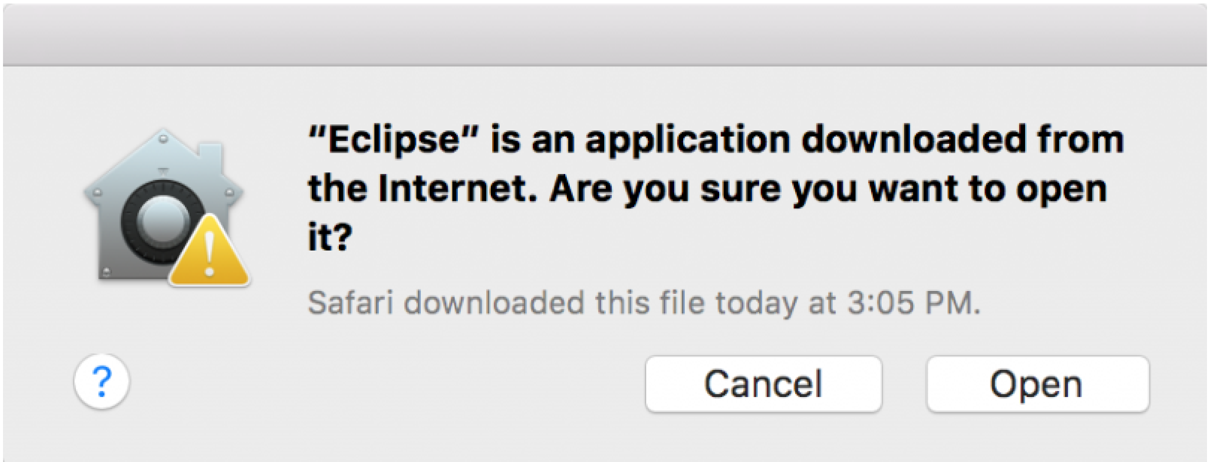

Figure 5: Eclipse IDE popup Window

After Opening the eclipse installer, select "Eclipse IDE for Java Developers".

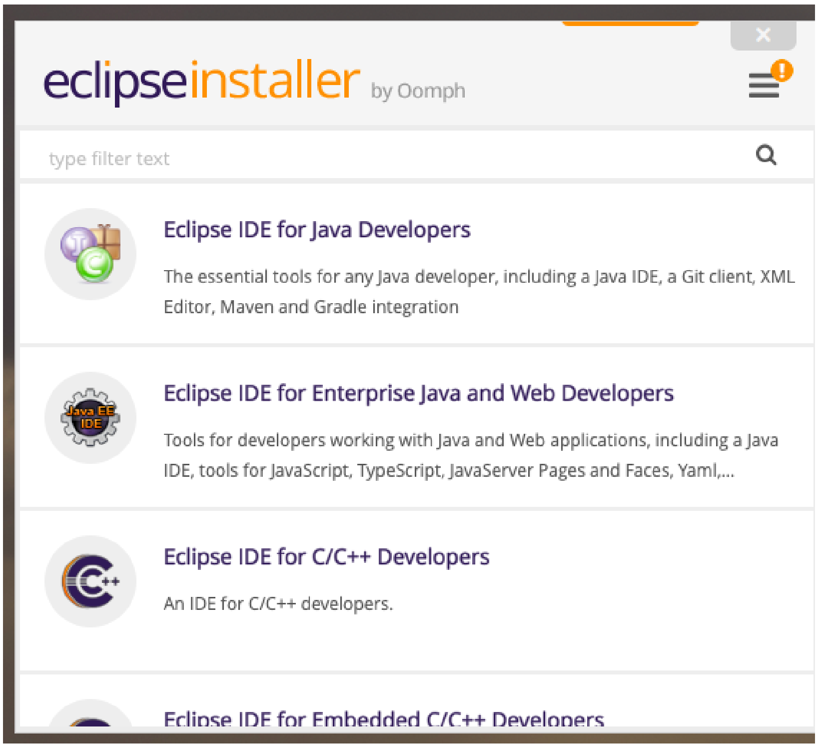

Figure 6: Eclipse installer

In the next window, from the drop down select the Java JDK and give the installation folder and enter “Install”.

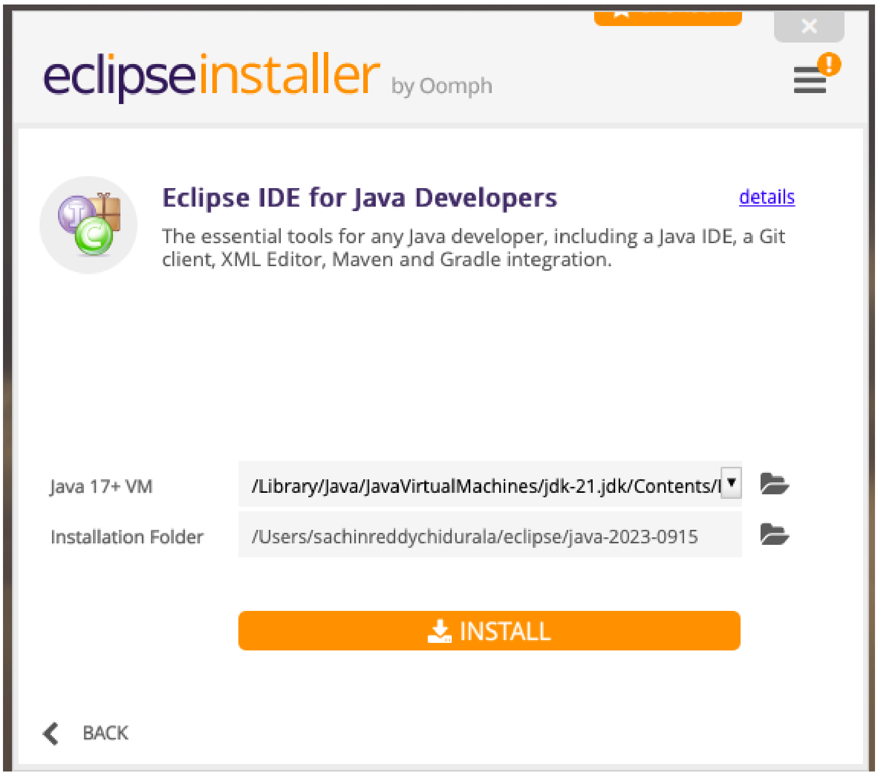

Figure 7: Eclipse Installer Installation

Now Eclipse Installation is done and a welcome pop window will display.

Figure 8: Eclipse Welcome Popup window

3.3 CloudSim Installation

3.3.1 Downloading Artifact (Cloudsim-4.0):

Download the cloudsim-4.0 zip file from Github using the link and Unzip the file. (https://github.com/Cloudslab/cloudsim/releases).

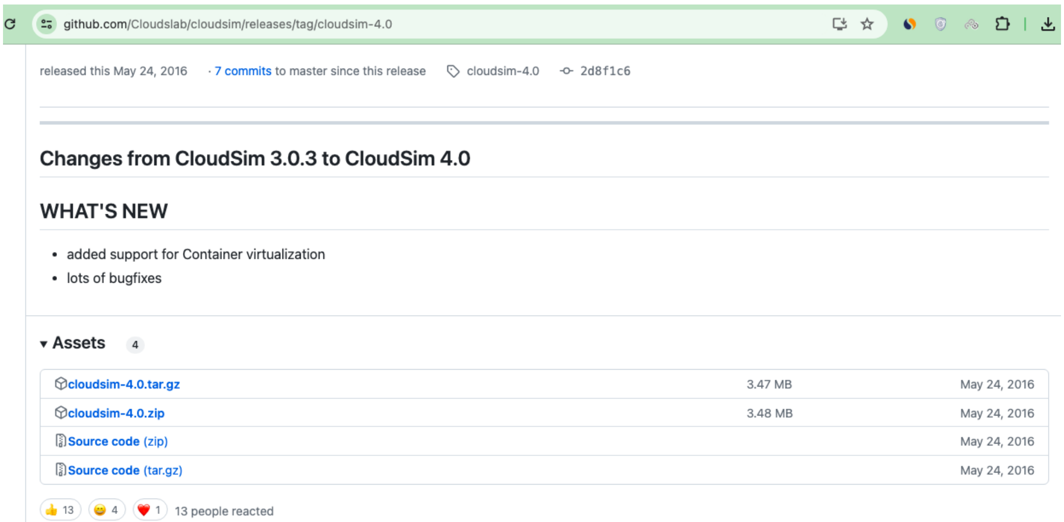

Figure 9: Downloading Cloudsim from Github

Now Open the Eclipse and just go to file and select import, then select Maven and from the dropdown choose the existing Maven project.

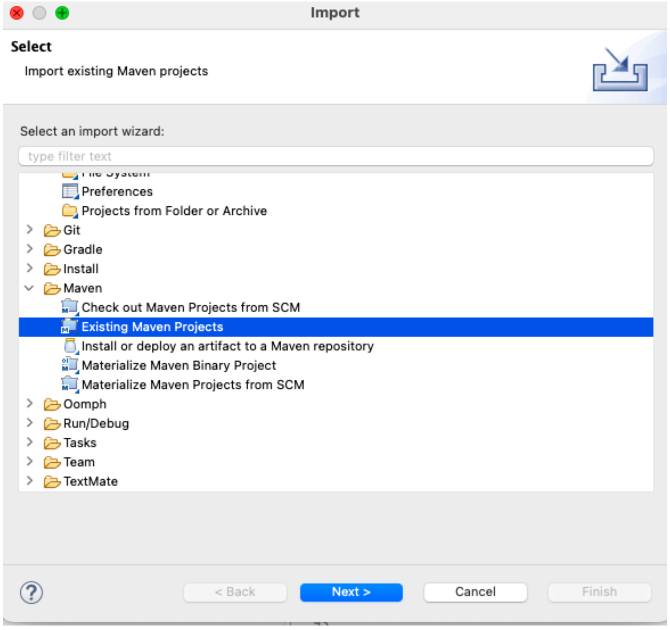

Figure 10: Importing Cloudsim from Existing Maven Projects

Next in the Root directory browse the Cloudsim Zip file and click finish. Make sure to mark all the boxes.

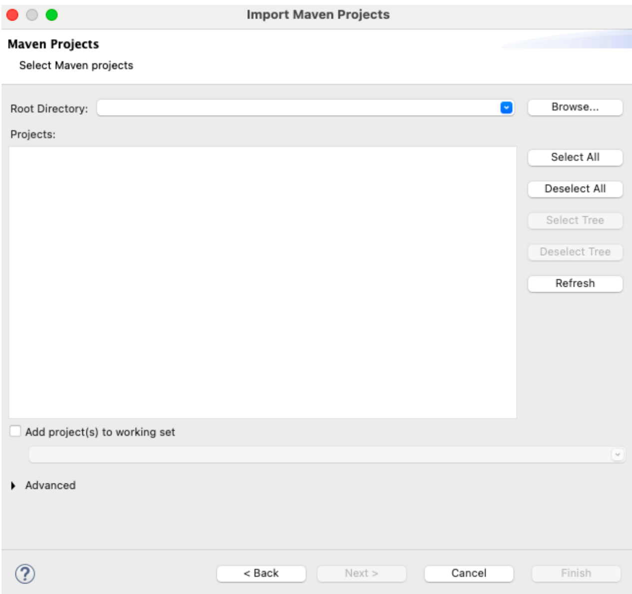

Figure 11: Browsing Cloudsim from local

Just wait for the build process to finish and now Cloudsim is imported into Eclipse. After the build process is completed the Eclipse IDE Looks like the below image.

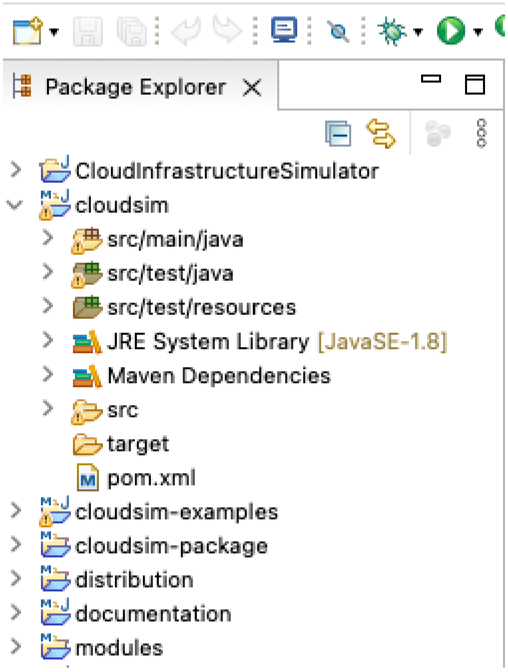

Figure 12: Cloudsim Package Explorer

Now right-click on the cloudsim- example folder and select Build Path and choose Java Build Path and add External Jars using Add external archive.

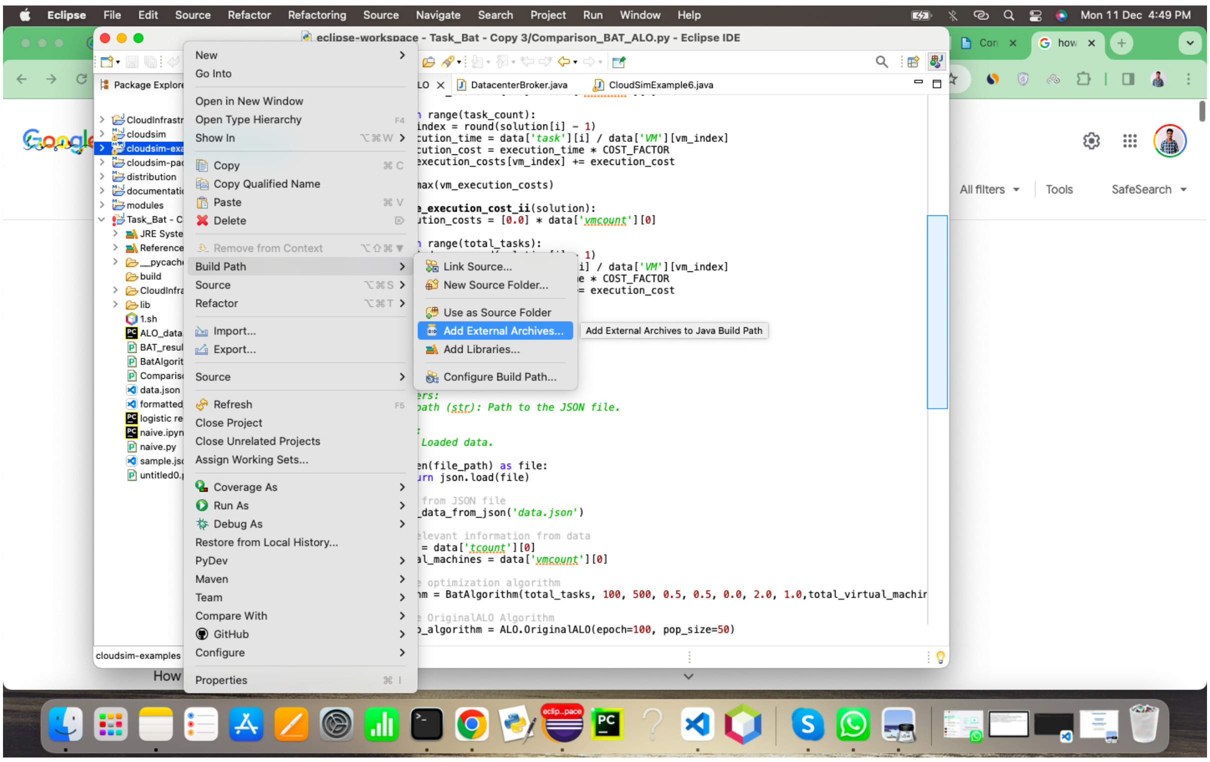

Figure 13: Adding Jar files in the java Build path

Now choose the Jars from the cloudsim zip file.

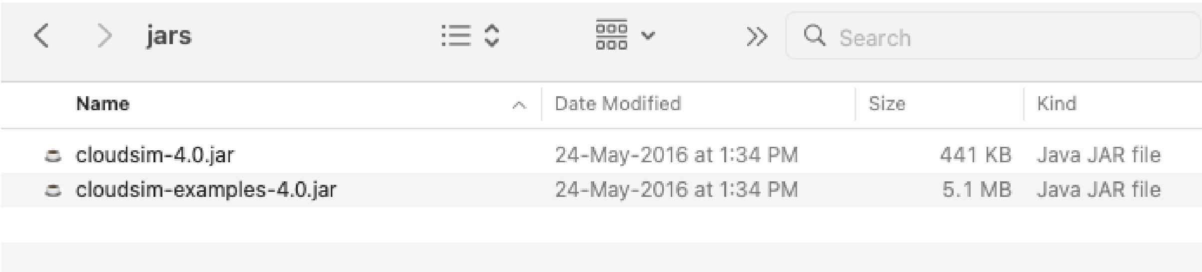

Figure 14: Cloudsim Jar Files in the system

4. Validations

Now will run the cloudsim example 6.

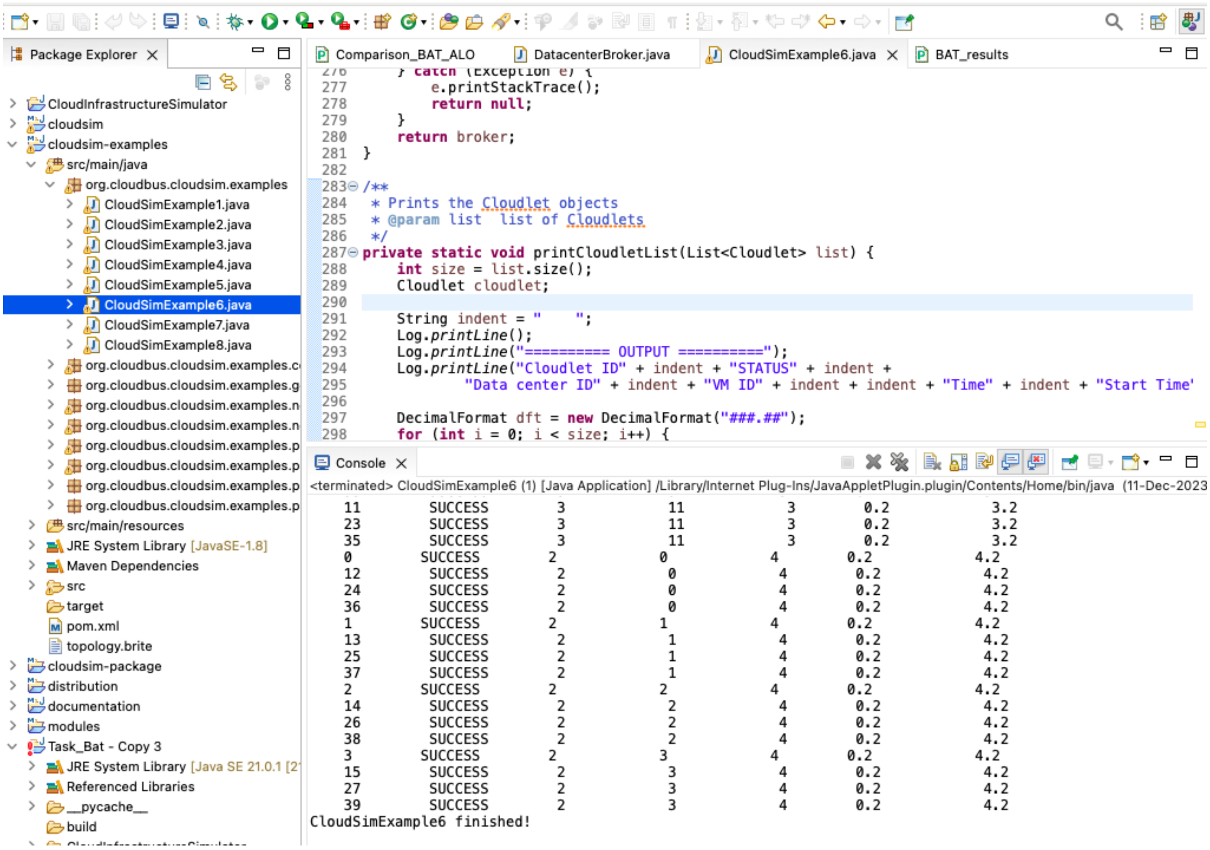

Figure 15: Cloudsim Example 6

The validations have been performed using Cloudsim and algorithms like BAT and ALO.

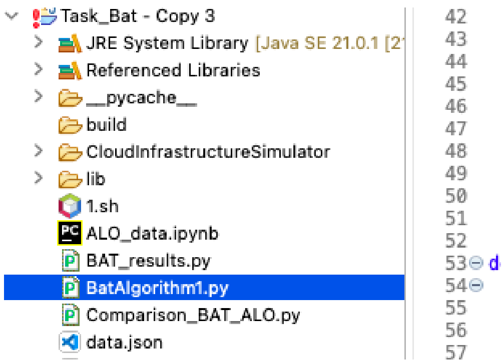

Figure 16: Imported BAT algorithm

From the data.json file the code will take the input with number of VMS, VMS capacity and Task count.

Figure 17: Data.json code where it shows VM count, VM capacity, Task Count

Also have implemented the BAT algorithm to find the execution cost and Run time for different tasks 100, 200, and 500. Below is the screenshot of the BAT optimization code.

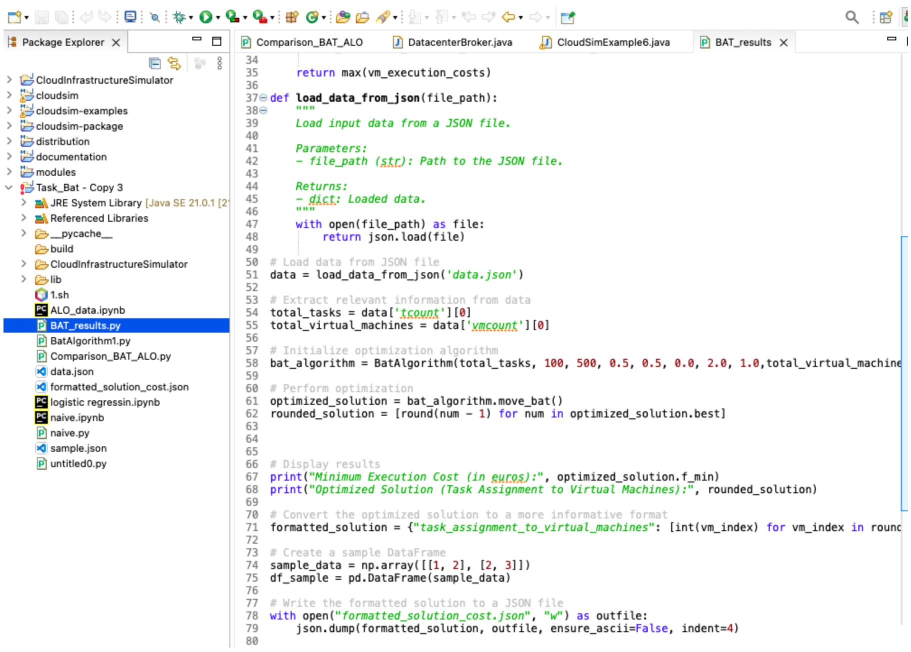

Figure 18: BAT algorithm Optimization code snippet.

Below are the execution cost and runtime results of the BAT optimization technique.

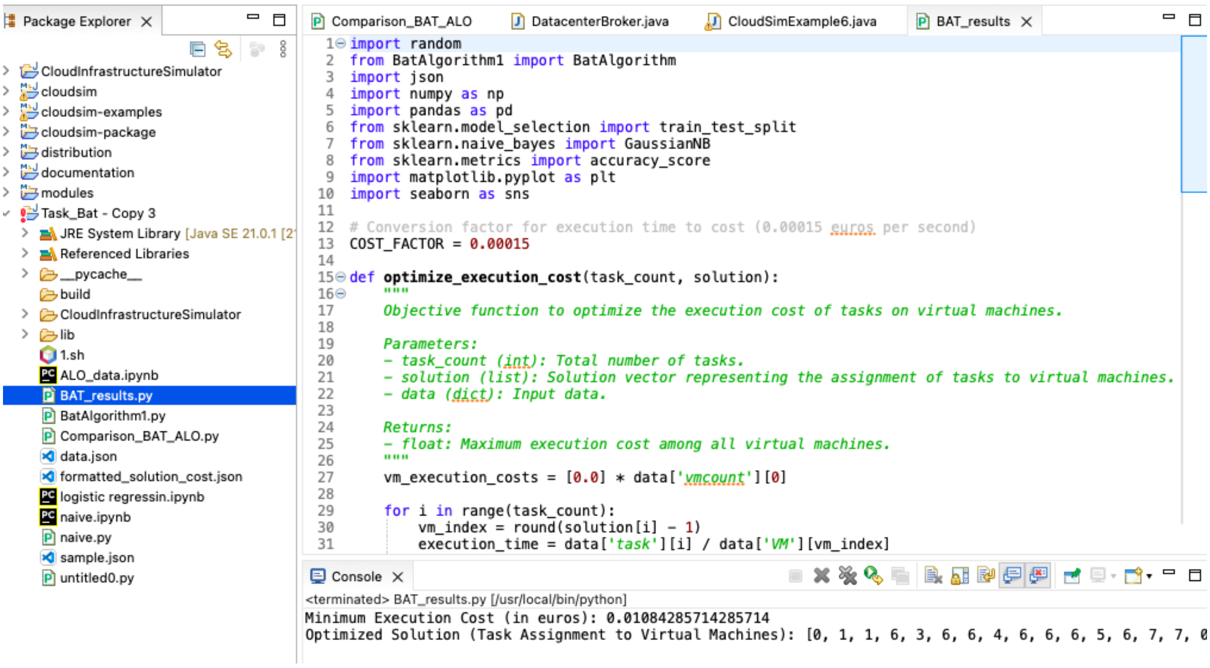

Figure 19: Simulation result for BAT algorithm with Execution cost and Task Assignment

Also I have done comparison of BAT results with the ALO algorithms results by importing the ALO algorithm using Mealpy library.

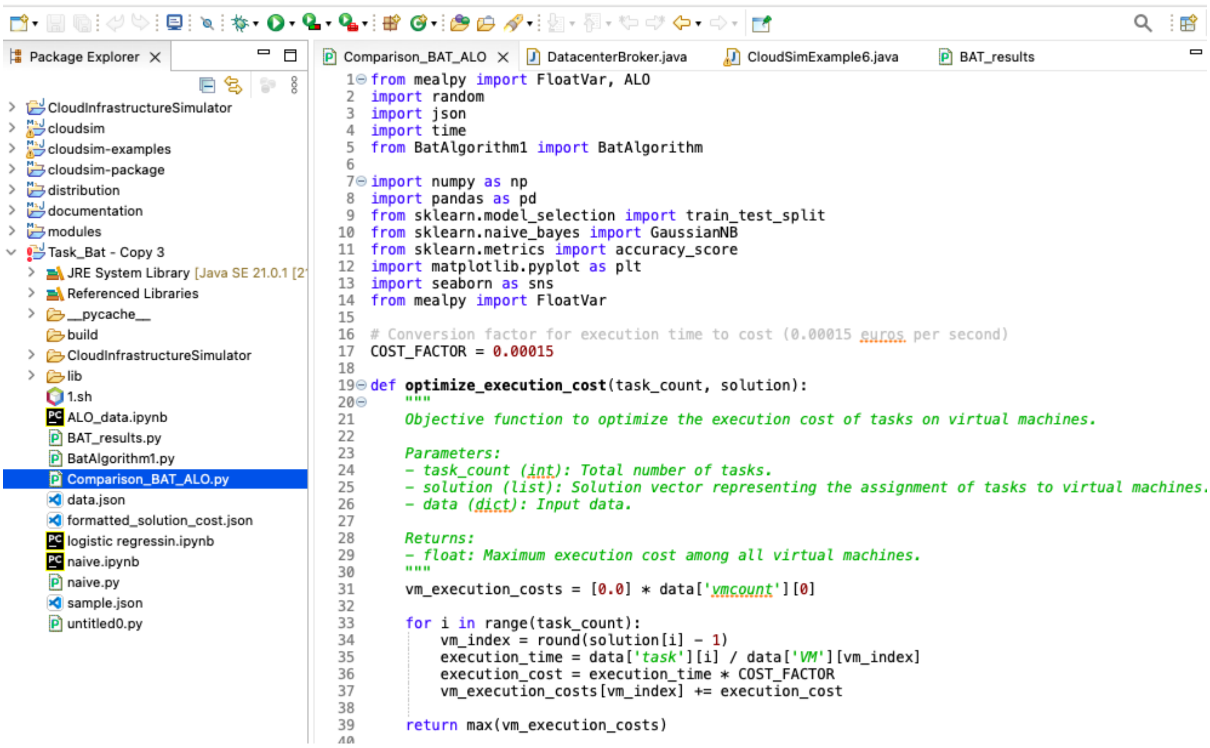

Figure 20: Comparison code snippet of both BAT and ALO

Here are the execution cost and runtime comparison results of both the BAT and ALO algorithm.

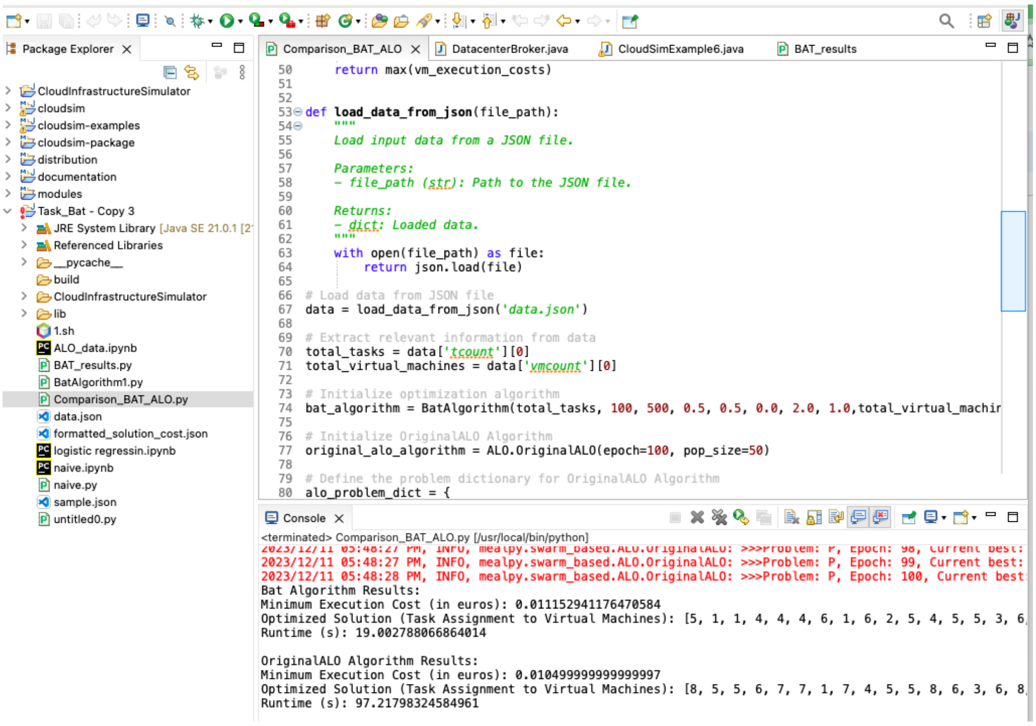

Figure 21: Execution cost and runtime comparison of both BAT and ALO algorithm