Configuration Manual

MSc Research Project
MSc in Cloud Computing

Vyom Dubey
Student ID: 23185732

School of Computing
National College of Ireland

Supervisor: Sudarshan Deshmukh

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Vyom Dubey
Student ID:	23185732
Programme:	MSc in Cloud Computing
Year:	2024
Module:	MSc Research Project
Supervisor:	Sudarshan Deshmukh
Submission Due Date:	12/12/2024
Project Title:	Configuration Manual
Word Count:	753
Page Count:	4

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:	
Date:	12th December 2024

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

Vyom Dubey
23185732

1 Introduction

Instructions on how to setup the workspace and perform the experiments are proposed. This document presents the pre-requisites and the requirements needed to perform the proposed research solution. This document also demonstrates the steps to be followed to achieve the proposed results.

2 System Specification

2.1 Hardware Requirements

System with minimum 4GB of RAM and 100GB of storage is required to run Cloudsim software toolkit.

2.2 Software Requirements

2.2.1 Java

Java programming language should be installed. Java Development Kit above version 11.0 is required to run the simulation program.

2.2.2 Eclipse IDE

Eclipse is an interactive development environment used for code workspace. We have used Eclipse for implementing our Java simulation code.

2.2.3 CloudSim

CloudSim is a cloud simulation tool used to simulate dynamic cloud environments consisting of cloud data centers, virtual machines and cloudlets.

3 Setup of Simulation Workspace

3.1 Eclipse IDE

- Download Eclipse IDE from the below link.
  https://eclipseide.org/
- Open the downloaded installation file of Eclipse ide.

- Make sure to select Java Development Kit during installation for debugging, developing and running simulation code in Java programming language.

- Complete the installation process of Eclipse ide.

3.2 CloudSim

- Download the CloudSim toolkit from the below link. https://github.com/Cloudslab/cloudsim

- Unzip the downloaded CloudSim file in the desired location.

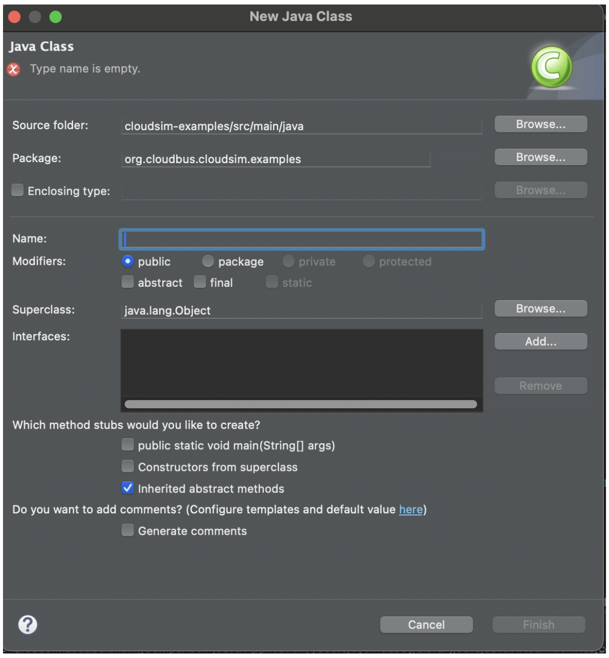
- Open the Eclipse ide and select the **import** option under the File toolbar.

- Select the unzipped Cloudsim folder to import the files.

- In the file explorer of Eclipse ide, open cloudsim-examples/src/main/java/org.cloudbus.cloudsim.ex folder.

4 Implementation

- Create a new class in the selected folder and name it "CloudSimLoadBalancingComparison" as seen in figure 1.



Java Class

Type name is empty.

Source folder: cloudsim-examples/src/main/java

Package: org.cloudbus.cloudsim.examples

□ Enclosing type:

Name:

Modifiers: ● public ○ package ○ private ○ protected

□ abstract □ final □ static

Superclass: java.lang.Object

Interfaces:

Which method stubs would you like to create?

□ public static void main(String[] args)

□ Constructors from superclass

☑ Inherited abstract methods

Do you want to add comments? (Configure templates and default value here)

□ Generate comments

Figure 1: New Class Dialog-Box

- Add the simulation code as added in the source artefact.

- Create a new class with name "FractionalKnapsackVmAllocationPolicy" and add the proposed scheduling algorithm as submitted in the source artefact.

- Create a new class with name ”AntColonyVmAllocationPolicy” with source code as provided in the source artefact for comparing the proposed algorithm and Ant Colony Optimization Algorithm.

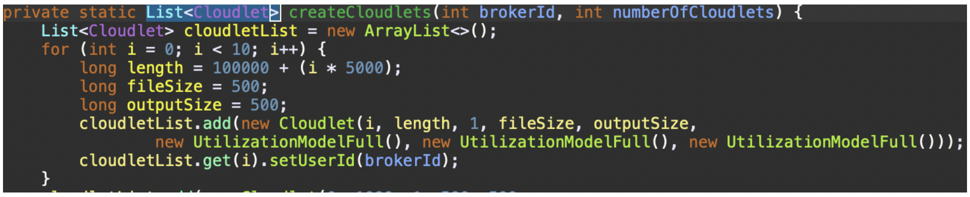
- Create a new class with name ”RoundRobinVmAllocationPolicy” with source code as provided in the source artefact for comparing the proposed algorithm and Round Robin Algorithm.

- Create a new class with name ”BinPackingVmAllocationPolicy” with source code as provided in the source artefact for comparing the proposed algorithm and Bin Packing Multitenancy Algorithm.

5 Experiments

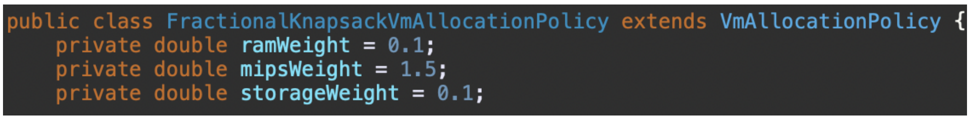
5.1 Varying the Cloudlets

For varying the Cloudlets based on different scenarios and workloads, make changes in the createCloudlets function as illustrated in the figure 2.



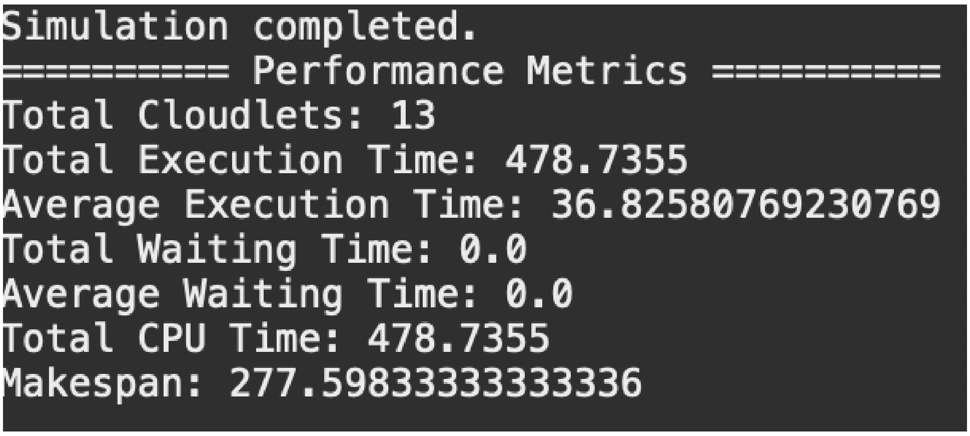
```
private static List<Cloudlet> createCloudlets(int brokerId, int numberOfCloudlets) {
    List<Cloudlet> cloudletList = new ArrayList<>();
    for (int i = 0; i < 10; i++) {
        long length = 100000 + (i * 5000);
        long fileSize = 500;
        long outputSize = 500;
        cloudletList.add(new Cloudlet(i, length, 1, fileSize, outputSize,
                new UtilizationModelFull(), new UtilizationModelFull(), new UtilizationModelFull()));
        cloudletList.get(i).setUserId(brokerId);
    }
```

Figure 2: Create Cloudlets Function



```
public class FractionalKnapsackVmAllocationPolicy extends VmAllocationPolicy {
    private double ramWeight = 0.1;
    private double mipsWeight = 1.5;
    private double storageWeight = 0.1;
```

Figure 3: Metrics Weight Variables



```
Simulation completed.
========== Performance Metrics ==========
Total Cloudlets: 13
Total Execution Time: 478.7355
Average Execution Time: 36.82580769230769
Total Waiting Time: 0.0
Average Waiting Time: 0.0
Total CPU Time: 478.7355
Makespan: 277.59833333333336
```

Figure 4: Output

5.2 Varying resources metrics weight

For varying the weight of metrics (ram, bandwidth and mips) of cloud resources for configuring the proposed algorithm for different use-cases, we need to change the values of weight of metrics in the proposed ”FractionalKnapsackVmAllocationPolicy” algorithm class as presented by the figure 3.

6 Results

After following the above steps, we need to run the simulation code by pressing run button in the eclipse ide toolbar. The results obtained from the simulation code are presented in the console as depicted by the figure 4