National College of Ireland

Configuration Manual

MSc Research Project
Research in Computing CA2

Javed Shaikh
Student ID: 21171581

School of Computing
National College of Ireland

Supervisor: Sean Heeney

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: Javed Abidali Shaikh
……..

Student ID: 21171581
……

Programme: MSc in Cloud Computing …… **Year:** 2022-23 ……..

Module: Research in Computing
……

Supervisor Sean Heeney
……

Submission Due Date: 14th August 2023 at 2:00pm
……

Project Title: Dynamic Scheduling in Edge-Cloud Computing Environments using Metaheuristic Techniques
……

Word Count: ……

Page Count: ……..

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: Javed Abidali Shaikh
……
…

Date: 14th August 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

Forename Surname
Student ID:

1 Introduction

This manual provides a comprehensive overview of the orderly procedure employed in this project, encompassing the establishment of the environment, execution, and evaluation. The handbook provides comprehensive information to the installation of the IDE, the configuration of the system, and CloudSim simulation environment.

2 System Specifications

Hardware Configuration for the local run:

- Processor: Intel 12th Gen Core i5-1235 @4.4 GHz
- Nvidia RTX3070 GPU
- RAM: 16 GB DDR4 RAM 3200MHz
- Storage (SSD): 512GB
- Operating System: Windows 10, 64-bit

Software Packages for the local run:

- Java SDK 1.8 or above
- IntelliJ IDEA Community Edition
- CloudSim 4.0

3 Environment Setup

3.1 IntelliJ IDEA installation

JetBrains created the integrated development environment (IDE) IntelliJ IDEA. It is predominantly used for Java development, but it also supports a variety of other programming languages and technologies. IntelliJ IDEA provides a robust set of features to facilitate software development and boost productivity.

Source Link: https://www.jetbrains.com/idea/download/?section=windows

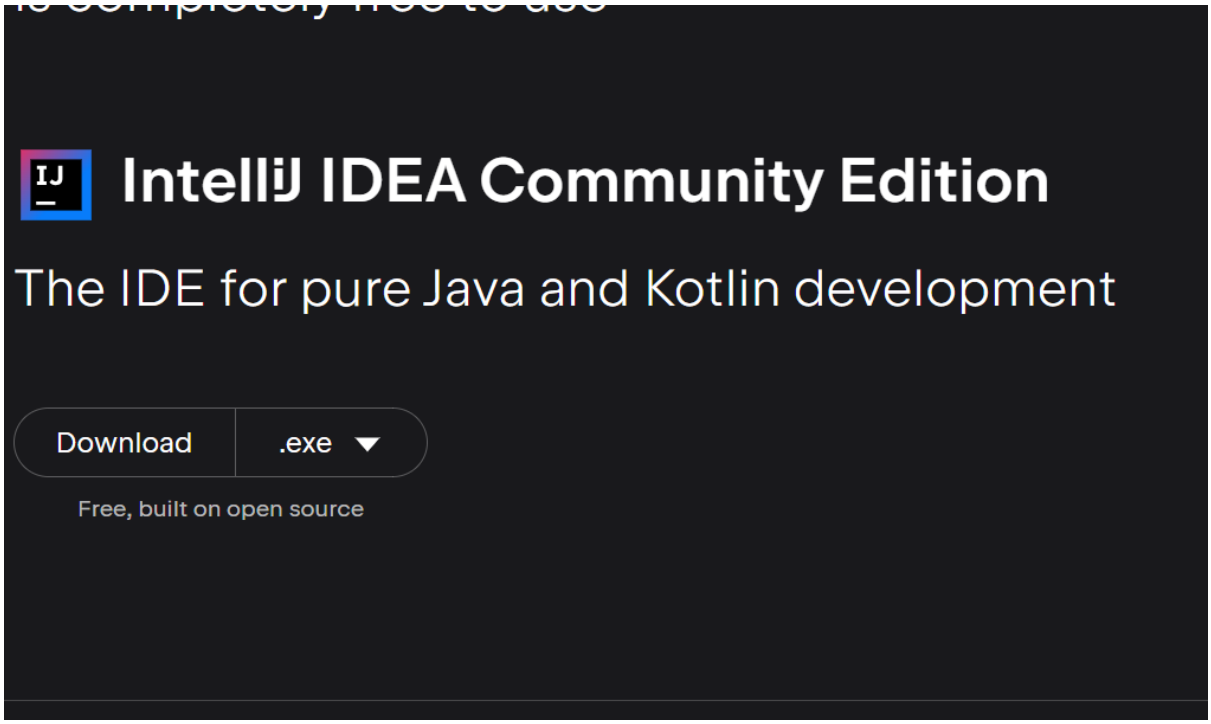

Figure 1: IntelliJ IDEA Community Edition Download

After installing, the IDE can be configured to run new projects or open existing projects. Figure 2 and 3 shows the settings for creating a new project and assigning the correct Java SDK version for your project.

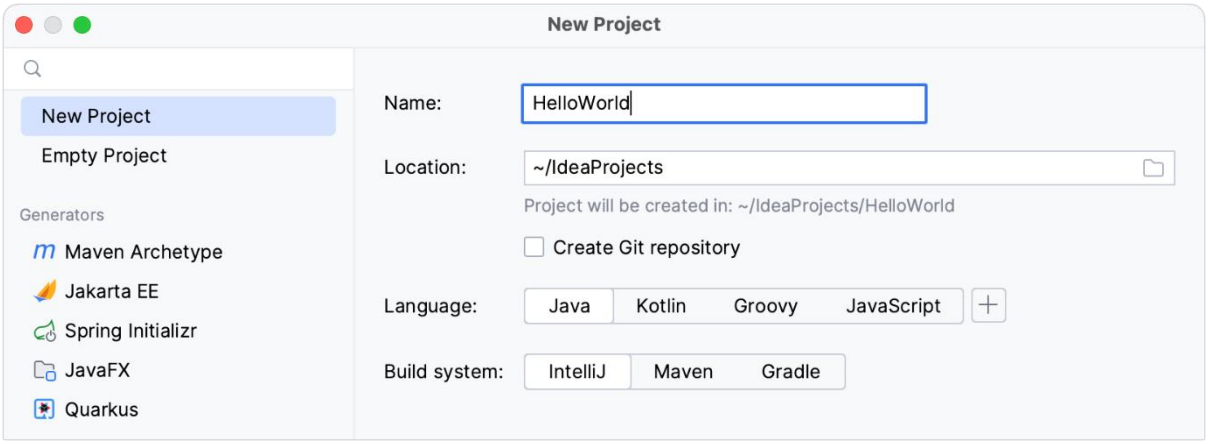

Figure 2: Creating a new project on IntelliJ

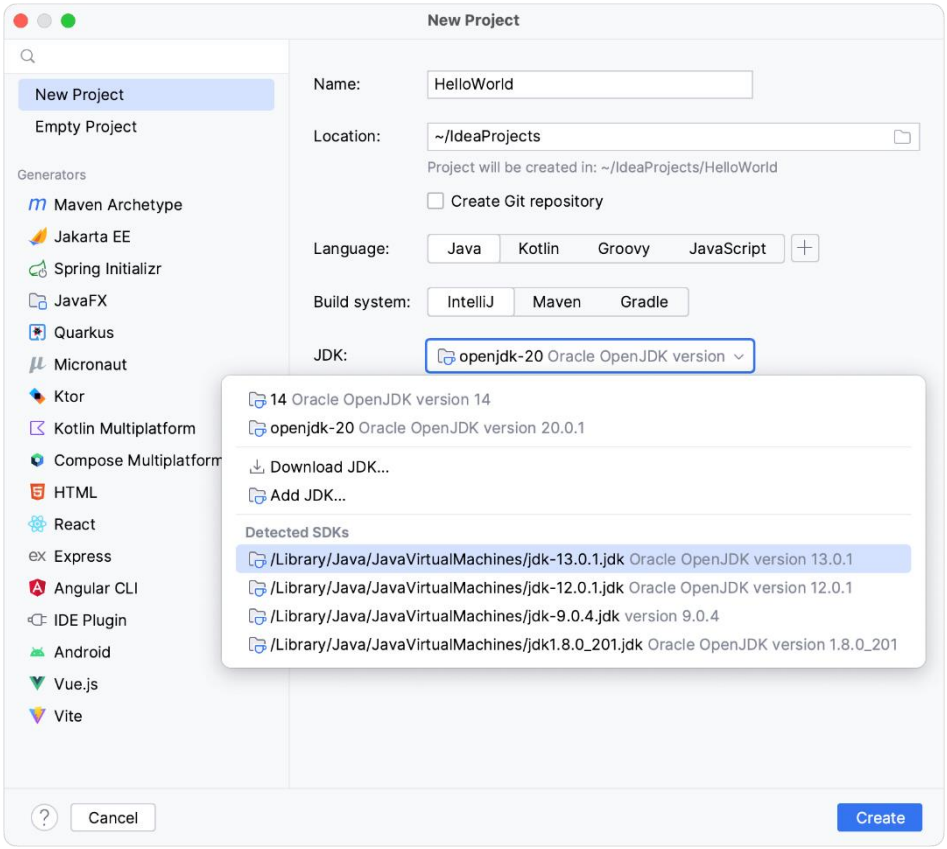

Figure 3: Java SDK version selection

3.2 CloudSim Package

The CloudSim package version 4.0 used in this research can be downloaded from as a zip file.

Source Link: https://github.com/Cloudslab/cloudsim/releases/tag/cloudsim-4.0

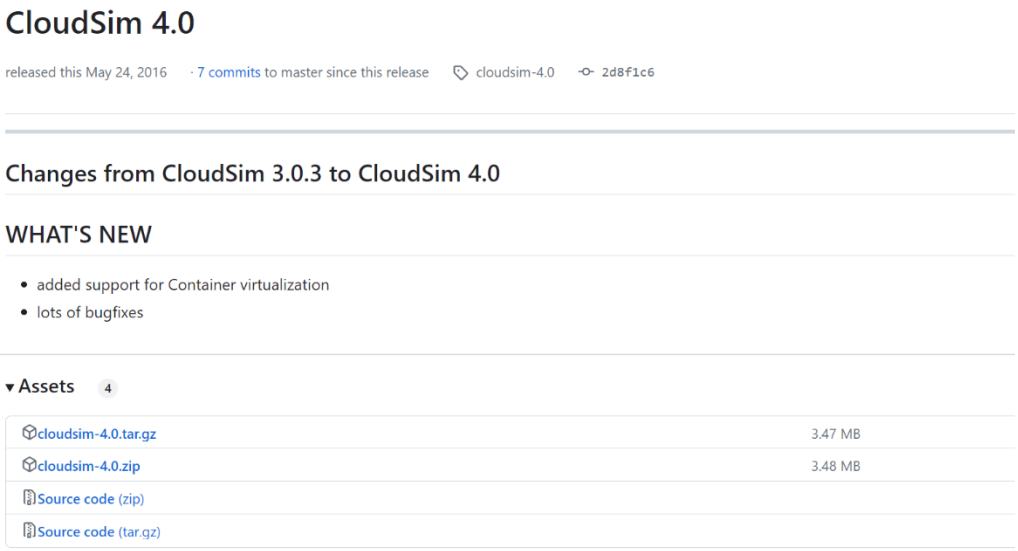

Figure 4: CloudSim 4.0 simulator download

Once it was downloaded, it can be extracted to a folder in the local drive. This can later be exported to the IntelliJ IDEA IDE to create the new project files, compile and run the environment. We have renamed the folder to EdgeScheduler, to represent our proposed work.

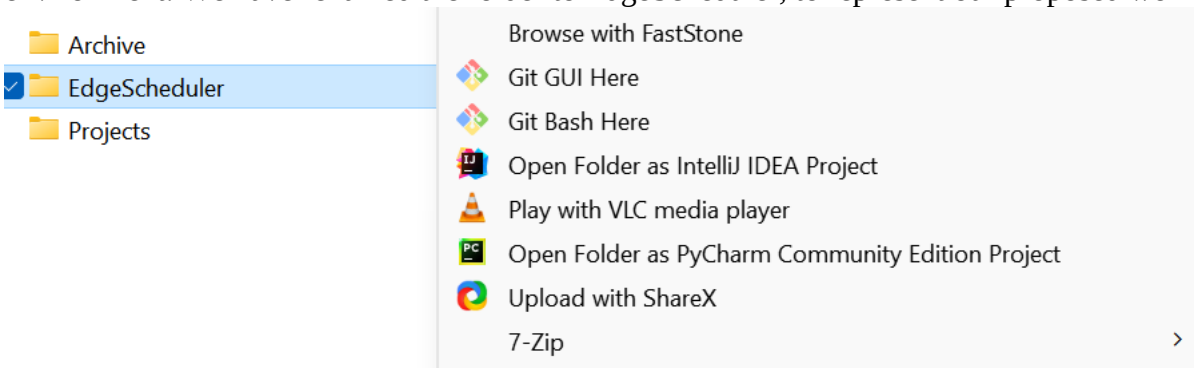

Figure 5: Open the project folder as IntelliJ Project

4 Project Development

The directory structure for is presented in Figure 6.

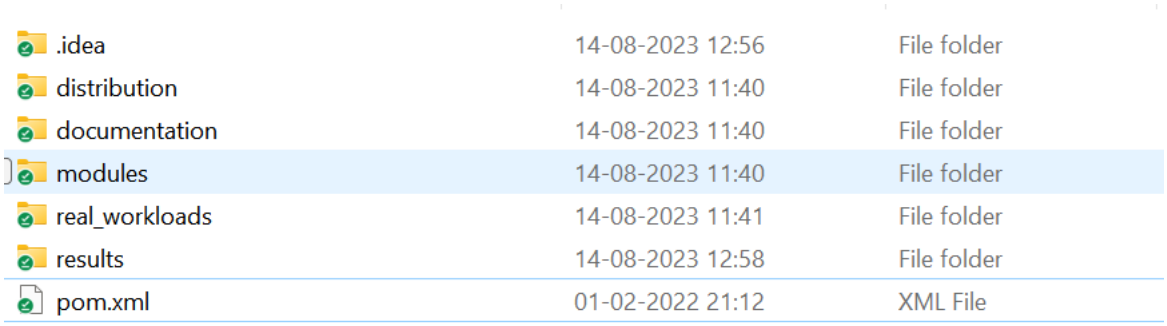

Figure 6: Project Directory Structure

The directories, modules, real workloads and results contain our simulation model, workload files and results, respectively.

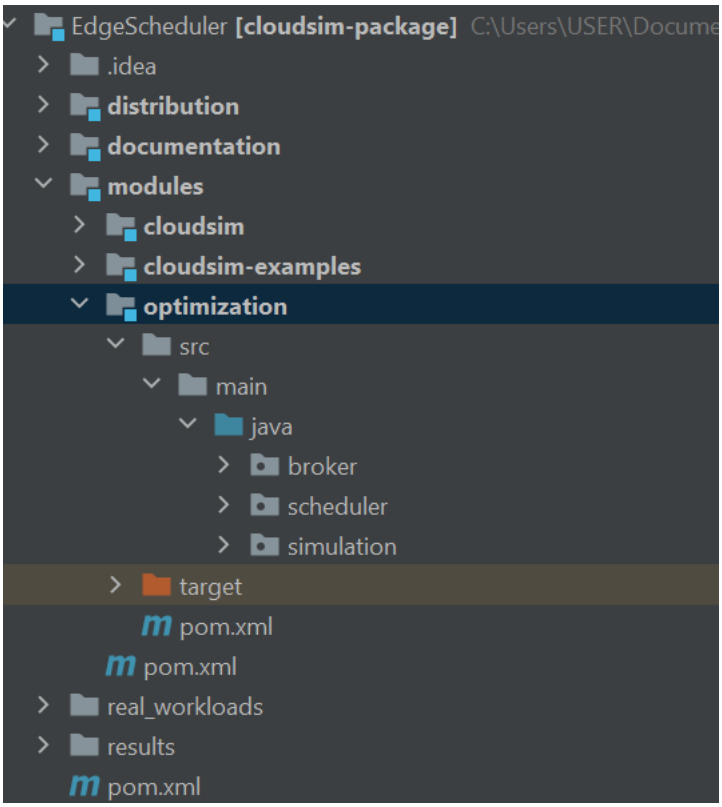

Figure 7: Code Directory→modules→optimization→src→main→java

The code directory is shown in Figure 7. It is classified into three folders, broker, scheduler and simulation. The files in these folders are shown in Figure 8.

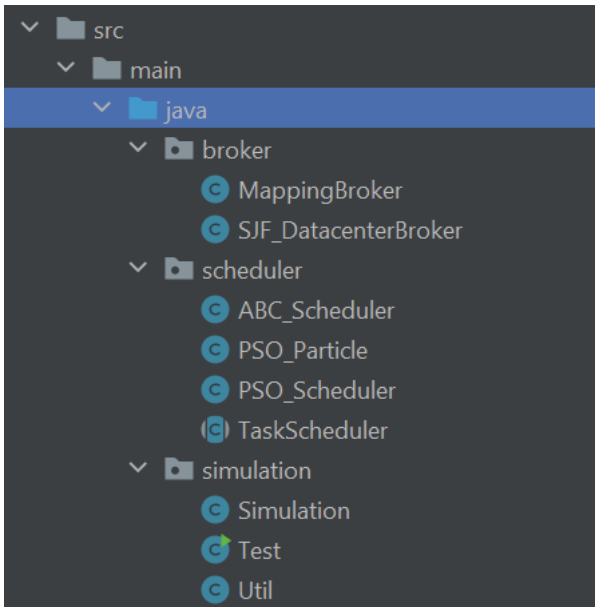

Figure 8: Code Files for broker, scheduler and simulation runs shown

Simulation.java contains all the modules and function calls to create the cloud environment for this work.

```java
import java.util.Random;
import org.cloudbus.cloudsim.Cloudlet;
import org.cloudbus.cloudsim.CloudletScheduler;
import org.cloudbus.cloudsim.CloudletSchedulerSpaceShared;
import org.cloudbus.cloudsim.CloudletSchedulerTimeShared;
import org.cloudbus.cloudsim.Datacenter;
import org.cloudbus.cloudsim.DatacenterBroker;
import org.cloudbus.cloudsim.DatacenterCharacteristics;
import org.cloudbus.cloudsim.Host;
import org.cloudbus.cloudsim.Log;
import org.cloudbus.cloudsim.Pe;
import org.cloudbus.cloudsim.Storage;
import org.cloudbus.cloudsim.UtilizationModel;
import org.cloudbus.cloudsim.UtilizationModelFull;
import org.cloudbus.cloudsim.Vm;
import org.cloudbus.cloudsim.VmAllocationPolicySimple;
import org.cloudbus.cloudsim.VmSchedulerTimeSharedOverSubscription;
import org.cloudbus.cloudsim.core.CloudSim;
import org.cloudbus.cloudsim.provisioners.BwProvisionerSimple;
import org.cloudbus.cloudsim.provisioners.PeProvisionerSimple;
```

Figure 9: Import CloudSim modules for the simulation

```java
4 usages
public Simulation(int cloudletSchedulerType, int numOfCloudlets, int numOfVMs, int brokerType,
    this.cloudletSchedulerType = cloudletSchedulerType;
    this.numOfCloudlets = numOfCloudlets;
    this.numOfVMs = numOfVMs;
    this.brokerType = brokerType;
    this.fitnessType = fitnessType;
    this.rng = rng;
    this.silent = silent;
```

Figure 10: Simulation Class

```java
public double runSimulation(int[] mapping) {
    if (silent) {
        //Log.disable();
    }
    Log.enable();
    Log.printLine("Simulation Starts...");

    double fitness = -1;

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

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

Figure 10: Run Simulation

```java
    /unused/
    Datacenter datacenter0 = createDatacenter( name: "Datacenter_0");
    /unused/
    //Datacenter datacenter1 = createDatacenter("Datacenter_1");

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

    //Fourth step: Create VMs and Cloudlets and send them to broker
    vmlist = createVM(brokerId, numOfVMs); //creating 10 vms
    cloudletList = createCloudlet(brokerId, numOfCloudlets); // creat

    broker.submitVmList(vmlist);
    broker.submitCloudletList(cloudletList);
```

Figure 11: Creation of Datacenter, Datacenter Broker, VM and Cloudlets

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

Figure 12: StartSimulation() will start the simulation of the selected scheduler code.

To run the experiments, four scenarios were considered in the *test.java* that must be run to execute the results.
Scenario 0: low number of cloudlets, high heterogeneity
Scenario 1: low number of cloudlets, low heterogeneity
Scenario 2: medium number of cloudlets, high heterogeneity
Scenario 3: medium number of cloudlets, low heterogeneity
The dataset used in these simulations is NASA-iPSC-1993-3. The results are evaluated in terms of statistical metrics, like average, min, max and standard deviation.

```
//scenario 0
System.out.println("\n********************** SCENARIO 0 **************************");
numOfCloudlets = 100;
highHeterogeneity = 1;
MAX_FES = numOfCloudlets * 1000;
doAllExperiments();

//scenario 1
System.out.println("\n********************** SCENARIO 1 **************************");
numOfCloudlets = 100;
highHeterogeneity = 0;
MAX_FES = numOfCloudlets * 1000;
doAllExperiments();

//scenario 2
System.out.println("\n********************** SCENARIO 2 **************************");
numOfCloudlets = 1000;
highHeterogeneity = 1;
MAX_FES = numOfCloudlets * 1000;
doAllExperiments();
```

Figure 13: Configurations for the considered scenarios

```
public static void ABCExp() {
    double[] results = new double[NUM_TRY];
    for (int i = 0; i < NUM_TRY; i++) {
        Random rng = new Random( seed: SEED + i);
        brokerType = 0;
        Simulation sim = new Simulation(cloudletSchedulerType, numOfCloudlets, numOfVMs, brokerType, fitnessType, r
        ABC_Scheduler abc_scheduler = new ABC_Scheduler(sim);
        int[] mapping = abc_scheduler.schedule(MAX_FES);
        double makespan = sim.runSimulation(mapping);
        results[i] = makespan;
    }
    calculateStatistics( algName: "ABC", results);
}
```

Figure 14: Running an experiment- Shown here is ABC Scheduler that is run under the simulation settings provided by *simulation.java* package. The results from the scheduler are saved to the results list and was printed as stdout values on the terminal.

```java
private static void calculateStatistics(String algName, double[] results) {
    System.out.println("\n\n------ " + algName + " -----");
    DescriptiveStatistics ds = new DescriptiveStatistics();
    for (double data : results) {
        ds.addValue(data);
    }
    System.out.println("\tAvg: " + ds.getMean());
    System.out.println("\tMin: " + ds.getMin());
    System.out.println("\tMax: " + ds.getMax());
    System.out.println("\tStd: " + ds.getStandardDeviation());
}
```

Figure 15: CalculateStatistics() computes the results in terms of Average, Min, Max and Standard Deviation values

```
********************* SCENARIO 1 *************************


------ FCFS -----
    Avg: 30.752494929677418
    Min: 29.50883882816658
    Max: 32.785413092231785
    Std: 1.124918417307495


------ SJF -----
    Avg: 29.94044823067099
    Min: 28.29690329845752
    Max: 31.213611591555235
    Std: 0.8109144908371719


------ ABC -----
    Avg: 28.248523598022413
    Min: 26.690237650708816
    Max: 29.372275877867875
    Std: 0.9255522082558976
```

Figure 16: Results in terms of statistical values for Scenario 1 presented

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID        Time    Start Time    Finish Time
     5         SUCCESS         2             5          1.44        0.1           1.54
     7         SUCCESS         2             7          1.73        0.1           1.83
     1         SUCCESS         2             1          2.12        0.1           2.22
     15         SUCCESS         2             5          1.5        1.54           3.03
     9         SUCCESS         2             9          2.93        0.1           3.03
     3         SUCCESS         2             3          3.06        0.1           3.16
     2         SUCCESS         2             2          3.43        0.1           3.53
     0         SUCCESS         2             0          3.57        0.1           3.67
     17         SUCCESS         2             7          2.44        1.83           4.27
     11         SUCCESS         2             1          2.34        2.22           4.56
     25         SUCCESS         2             5          1.64        3.03           4.67
     4         SUCCESS         2             4          4.8         0.1           4.9
     19         SUCCESS         2             9          2.35        3.03           5.38
     35         SUCCESS         2             5          1.3         4.67           5.97
     27         SUCCESS         2             7          2.08        4.27           6.35
     21         SUCCESS         2             1          2.33        4.56           6.89
     45         SUCCESS         2             5          1.22        5.97           7.19
     13         SUCCESS         2             3          4.24        3.16           7.4
     10         SUCCESS         2             0          3.84        3.67           7.51
     29         SUCCESS         2             9          2.13        5.38           7.51
     6         SUCCESS         2             6          7.61        0.1           7.71
     37         SUCCESS         2             7          1.81        6.35           8.16
     55         SUCCESS         2             5          1.72        7.19           8.9
     14         SUCCESS         2             4          4.29        4.9           9.19
     47         SUCCESS         2             7          1.42        8.16           9.58
     8         SUCCESS         2             8          9.59        0.1           9.69
     31         SUCCESS         2             1          2.91        6.89           9.8
     12         SUCCESS         2             2          6.27        3.53           9.8
     65         SUCCESS         2             5          1.18        8.9           10.08
```

Figure 17: Output file that shows the successful allocation of cloudlets to VMs and datacenter assigned to with their runtimes.

References

Sundas, A. and Panda, S.N., 2020, March. An introduction of CloudSim simulation tool for modelling and scheduling. In *2020 international conference on emerging smart computing and informatics (ESCI)* (pp. 263-268). IEEE.

Hicham, G.T. and Chaker, E.A., 2016. Cloud Computing CPU Allocation and Scheduling Algorithms Using CloudSim Simulator. *International Journal of Electrical & Computer Engineering (2088-8708)*, *6*(4).

The NASA Ames iPSC/860 log.
Available at: https://www.cs.huji.ac.il/labs/parallel/workload/l_nasa_ipsc/ (Accessed: 21 July 2023)