Configuration Manual

MSc Research Project
Cloud Computing

Pravin Ravindran Shanmugam
Student ID: x23186607

School of Computing
National College of Ireland

Supervisor: Shivani Jaswal

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Pravin Ravindran Shanmugam
Student ID:	X23186607
Programme:	MSc in Cloud Computing..... **Year:** 2024
Module:	Research Project.....
Lecturer:	Shivani Jaswal
Submission Due Date:	28.01.2025.....
Project Title:	Improvement of Intelligent Task Prediction and Computation Offloading towards mobile-edge cloud computing.
Word Count:	907..... **Page Count:** 10.....

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: Pravin Ravindran Shanmugam

Date: 28.01.2025

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

Pravin Ravindran Shanmugam
Student ID: x23186607

AWS SageMaker Setup with S3

First, make sure your AWS account – be it your private account or college's one – is functional. If you do not have an account on AWS, then first, register for one. Configure the necessary IAM roles with the necessary permissions, such as:

AWS Generate Policy should be used when creating new ones. You can add new policies to build policies with an explicit ARN for any service and principle. Be certain to do this for all resources; if not, utilize the edit option in policy to change each certain policy as you deem fit.

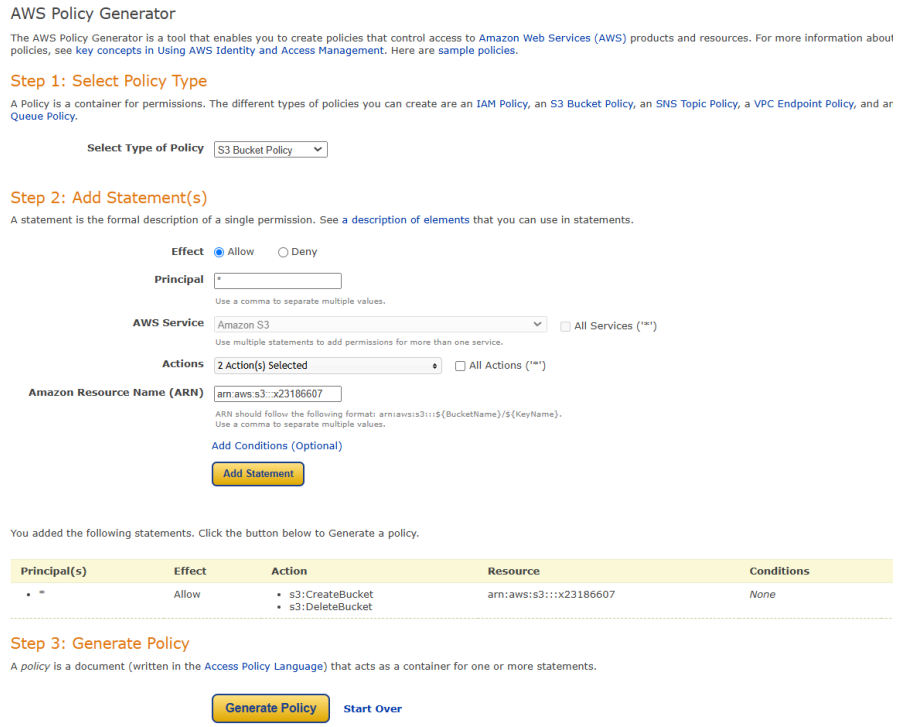

Fig. 1: Policy generation

This way you can paste it to include the specific services once the policy has been generated.

1. SageMaker Execution Role:

Wherever SageMaker will be utilised, this position should be able to communicate with the S3 buckets. Make careful to define execution roles for every S3 bucket if you are utilising more than one. The prototype for the authorization of the current S3 bucket included with a particular S3 bucket from the available permission option is shown in the image below.

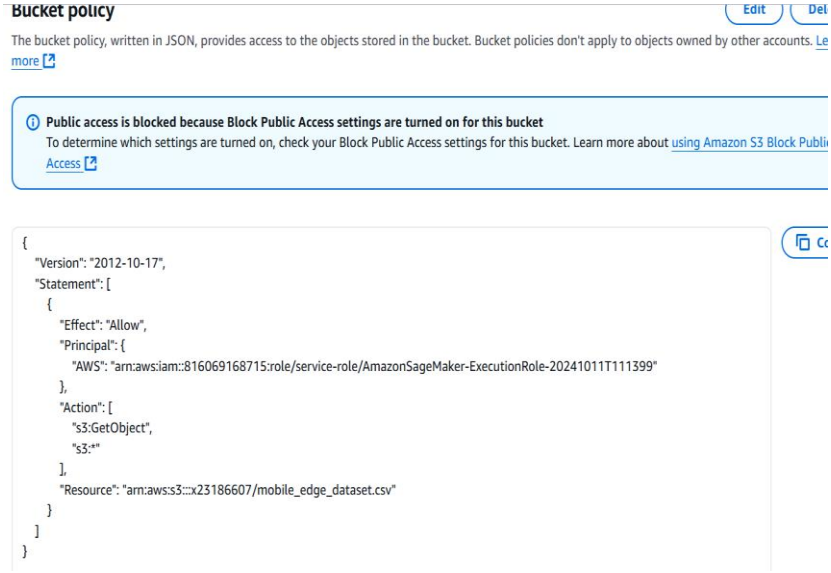

Fig. 2: S3 Bucket Policy Access

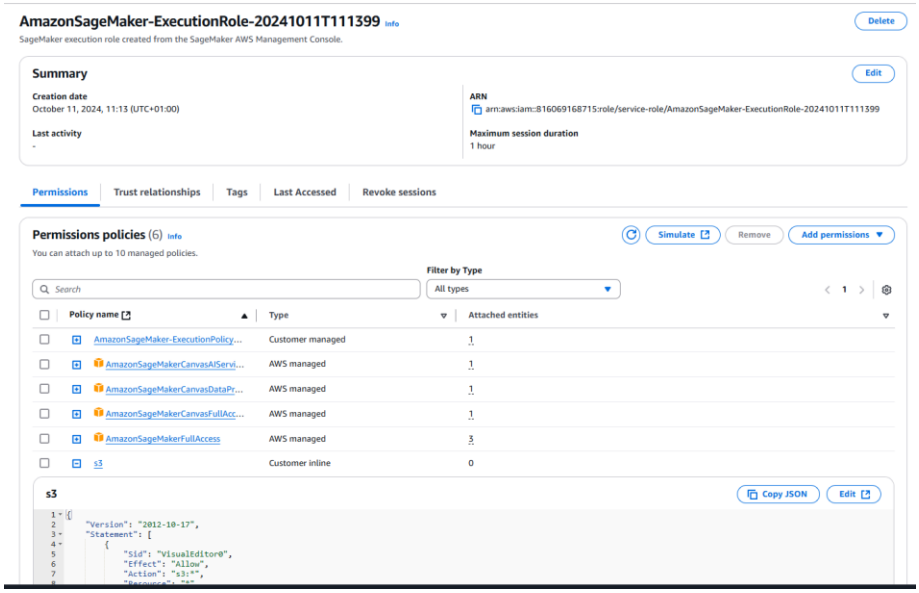

Fig. 3: Execution Role Permission for SageMaker

Environmental Setup

Python: Make sure that the python has been installed in your computer. Depending upon whether Python is not installed in the computer, able to install and download Python available from the official web site. In the case of observation, follow the instructions based on your requirement. If you have already done that, please use the code below to check if the suitable version have been installed:

Fig. 6: Check Python and Scikit Version

Next check the scikit learn is install with 1.2.2 version since verify the deployed model's scikit learn version and machine's installed version are same.

Scikit Learn 1.2.2

Libraries: Please ensure to install all the necessary libraries for python as described in the fig. 7.

```
# Import necessary libraries
import sagemaker
import boto3
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import xgboost as xgb
import joblib
import os
```

Fig. 7: Import Python libraries

SageMaker:

1. **Notebook Instance:** The Jupyter instances Notebook configuration is basic to design, training and utilize model in SageMaker, as has been illustrated in fig. 8 first steps leading to the creation of a new notebook have to be undertaken.

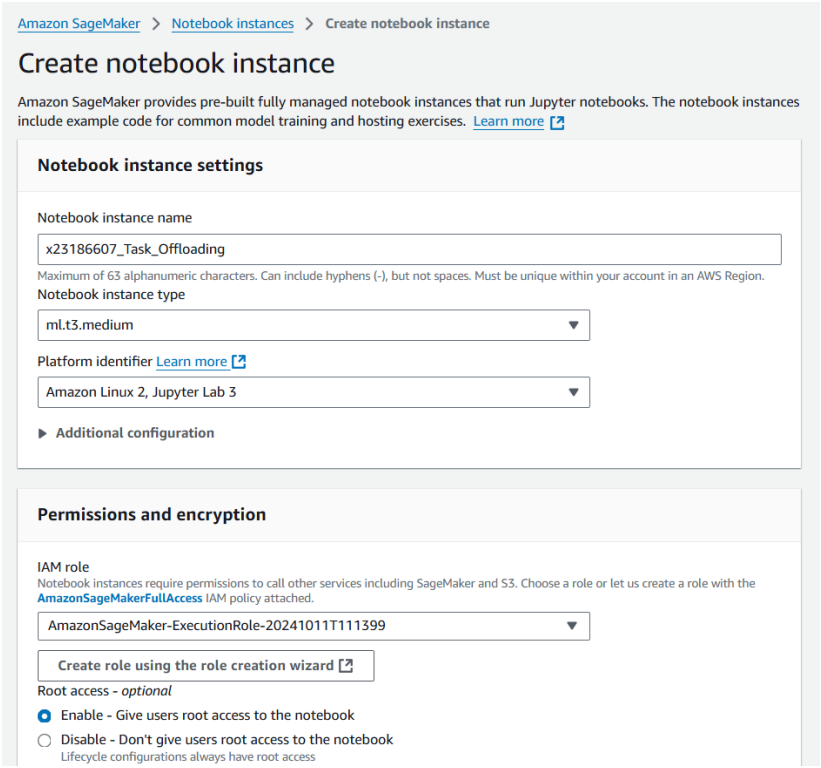

Fig. 8: Notebook Instance using SageMaker

For extended user interface, We can utilize the SageMaker Home console. In fact, once an instance has been created one can be able to see the status as Running. The specific reminder I have is always ensure you have stopped all the running services before turning it off.

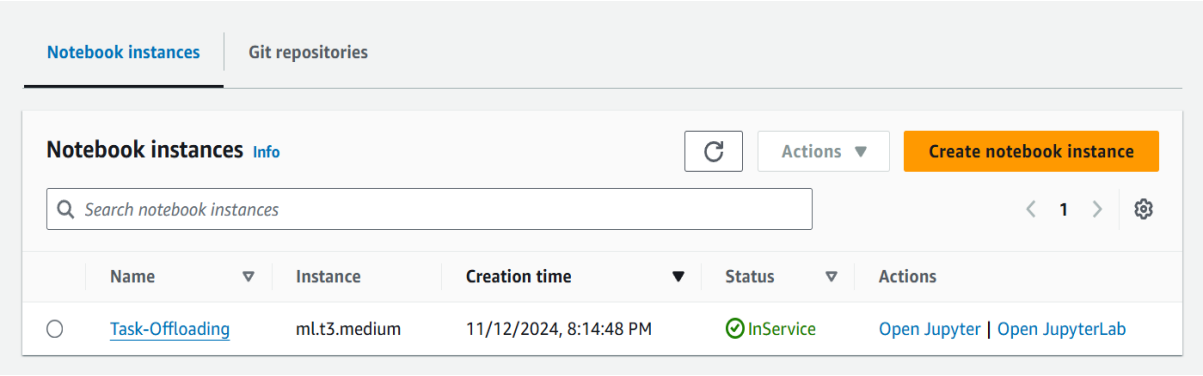

Fig. 9: InService Notebook Instance

2. **Loading of Dataset from S3:** With generally setting region and the name of the bucket you can access to store the data and to read dataset from that storage cloud.

```
[5]: region = boto3.Session().region_name
     sagemaker_session = sagemaker.Session()
     role = sagemaker.get_execution_role()

[6]: print(f"AWS Region: {region}")
     print(f"RoleArn: {role}")

AWS Region: us-east-1
RoleArn: arn:aws:iam::816069168715:role/AWS-Sagemaker-ExecutionRole-816069168715
```

2. Let's Look at the dataset Stored in S3 utilized by its name

- mobile_edge_dataset.csv is the dataset file taken from kaggle and used in this model, its a various task category details dataset
- The dataset management is managed by AWS S3 Bucket on cloud.

```
[7]: # Stored data Bucket name & file name
     bucket_name = 'x23186607'
     file_key = 'mobile_edge_dataset.csv'

[8]: # Read the CSV file from S3
     s3 = boto3.client('s3')
     prefix = 'task-offloading'  # Folder in S3
     csv_obj = s3.get_object(Bucket=bucket_name, Key=file_key)
     body = csv_obj['Body']
     csv_string = body.read().decode('utf-8')

[9]: # Load the dataset into a pandas dataframe
     data = pd.read_csv(StringIO(csv_string))

[10]: data
```

[10]:

	Device_ID	CPU_Usage (%)	Memory_Usage (%)	Battery_Level (%)	Server_ID	CPU_Capacity (%)	Available_Memory (%)	Network_Latency (ms)	Task_Type
0	1	61	83	49	2	64	85	20	Sensor
1	2	85	83	5	2	99	59	35	Image

Fig. 10: Loading of Dataset from S3 to be used entirely

3. **Model Training and Deployment of Endpoint:** The following code snippet is responsible for the model training job as well as creation of the endpoint configuration. Use the code below.

```
s3_train_path = sagemaker_session.upload_data(path='train.csv', bucket=bucket_name, key_prefix='train')

# Define SKLearn estimator for RandomForest model
sklearn_estimator = SKLearn(
    entry_point='train_rf.py',  # Specify the correct file name here
    instance_type='ml.m5.large',
    framework_version='1.2-1',  # Use the closest supported version
    role=role,
    sagemaker_session=sagemaker_session,
    hyperparameters={
        'n_estimators': 100,
        'max_depth': 10,
        'random_state': 42
    }
)

# Fit RandomForest model using SKLearn estimator on SageMaker
sklearn_estimator.fit({'train': s3_train_path})

# Local model training for comparison: Logistic Regression and XGBoost
# Logistic Regression
log_reg_model = LogisticRegression(random_state=42, max_iter=2000)
log_reg_model.fit(X_train, y_train)
y_pred_log_reg = log_reg_model.predict(X_test)
accuracy_log_reg = accuracy_score(y_test, y_pred_log_reg)

# XGBoost
xgb_model = XGBClassifier(eval_metric='logloss', random_state=42)
xgb_model.fit(X_train, y_train)
y_pred_xgb = xgb_model.predict(X_test)
accuracy_xgb = accuracy_score(y_test, y_pred_xgb)

# RandomForest (for local testing only)
rf_model = RandomForestClassifier(n_estimators=100, max_depth=10, random_state=42)
rf_model.fit(X_train, y_train)
```

```
INFO:sagemaker:Creating training-job with name: sagemaker-scikit-learn-2024-11-30-13-47-04-030
2024-11-30 13:47:05 Starting - Starting the training job...
2024-11-30 13:47:19 Starting - Preparing the instances for training...
2024-11-30 13:47:53 Downloading - Downloading input data...
2024-11-30 13:48:18 Downloading - Downloading the training image.....2024-11-30 13:49:15,060 sagemaker-containers INFO
ainer.training
```

Fig. 11: Job of Model Training

After the training of the task have been finished, the model is ready for deployment.

```
[20]: # Deploy the trained RandomForest model
predictor = sklearn_estimator.deploy(
    initial_instance_count=1,
    instance_type='ml.m5.large'
)

# Define helper function to preprocess and make predictions with the deployed model
def predict_with_sagemaker(predictor, data):
    response = predictor.predict(data)
    return response

# Prepare test data for prediction (using the same scaling and feature encoding)
y_pred_rf = predict_with_sagemaker(predictor, X_test)
accuracy_rf = accuracy_score(y_test, y_pred_rf)

# Display accuracy comparison
print(f"Logistic Regression Accuracy: {accuracy_log_reg}")
print(f"XGBoost Accuracy: {accuracy_xgb}")
print(f"RandomForest Accuracy (SageMaker): {accuracy_rf}")
```

```
INFO:sagemaker:Creating model with name: sagemaker-scikit-learn-2024-11-30-13-49-52-663
INFO:sagemaker:Creating endpoint-config with name sagemaker-scikit-learn-2024-11-30-13-49-52-663
INFO:sagemaker:Creating endpoint with name sagemaker-scikit-learn-2024-11-30-13-49-52-663
------!Logistic Regression Accuracy: 0.49166666666666664
XGBoost Accuracy: 0.525
RandomForest Accuracy (SageMaker): 0.5333333333333333
```

Fig. 12: Model deployed as endpoint for further usage.

Eclipse & iFogSim Setup:

1. **Download Eclipse IDE in your System using specified link:**

Link: https://www.eclipse.org/downloads/packages/

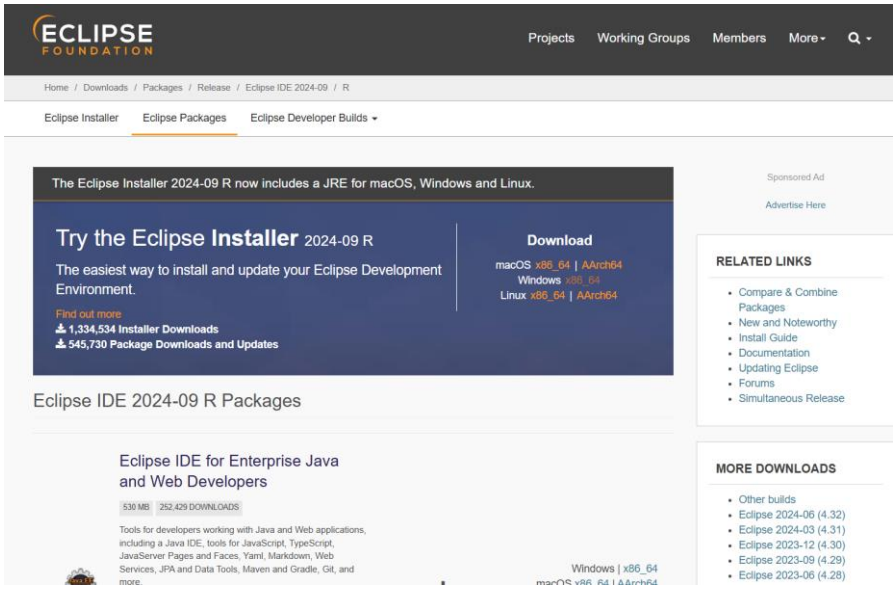

Fig. 13: Download Eclipse

For the purposes of this simulation, **Eclipse IDE for Java Developers** is the recommended option; select the appropriate version of Eclipse, Windows, MacOS or Linux compatible to your system.

1. With this download button, people can download the one that required.

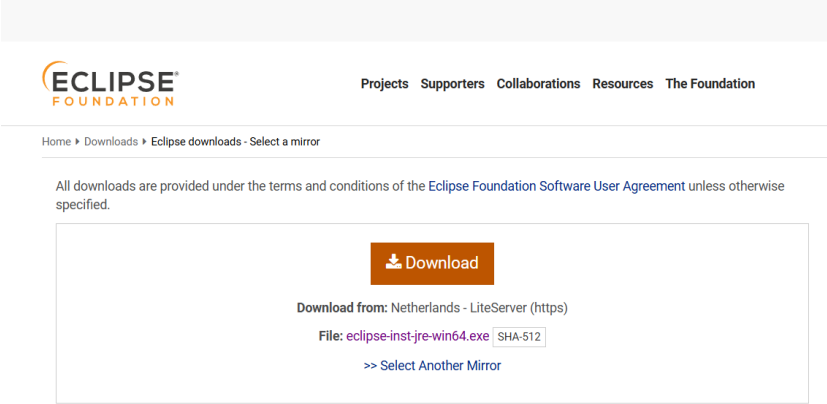

Fig. 14: Install Eclipse

2. Install Eclipse:
- After running the installer select Eclipse IDE for Java Developers.
- Once that is done, launch the Eclipse tools.
3. Select and obtain a directory for your work that will act as the workspace.

2. Configure the environment for Java development

1. Verify that the Java Development Kit (JDK) have been installed on your computer: Available here : Download JDK.
2. Verify Java in Eclipse:

- Go to Window > Preferences > Java > Installed JREs.
- Add and select your installed JDK if it is not listed.

3. Download iFogSim from GitHub

1. Go to the iFogSim GitHub repository: iFogSim GitHub.
2. Click **Download ZIP** to download the repository.
3. Extract the ZIP file to a folder on your local machine.

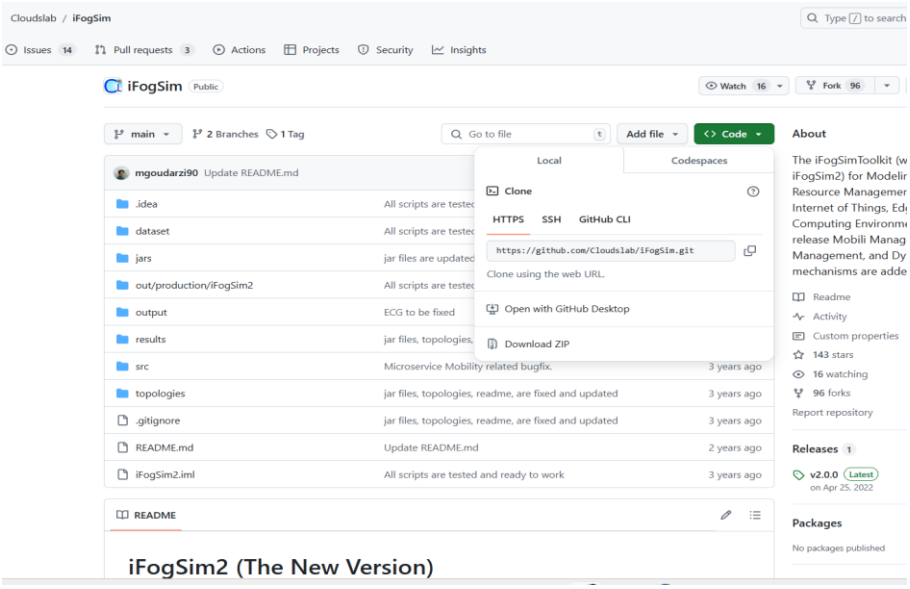

Fig. 15: iFogSim Git Repo

4. Import iFogSim into Eclipse

1. Open Eclipse IDE.
2. Select **File - Import - Existing Projects into Workspace**.
3. On the import wizard:
    - Select the folder where you extracted the iFogSim repository.
    - Ensure the projects are listed and selected.
4. Click **Finish** to import iFogSim into your workspace.

5. Build and run iFogSim

1. Import the source project and the source plugins into Eclipse; Eclipse should build these two automatically.
2. Verify the build:
3. This is important to check that there are no mistakes on the Problems view.
4. Run an example simulation:
5. Visit a sample Java file for instance org.fog.test.perfeval.x23186607.java.
6. Form a right click on the file select run as > java application.

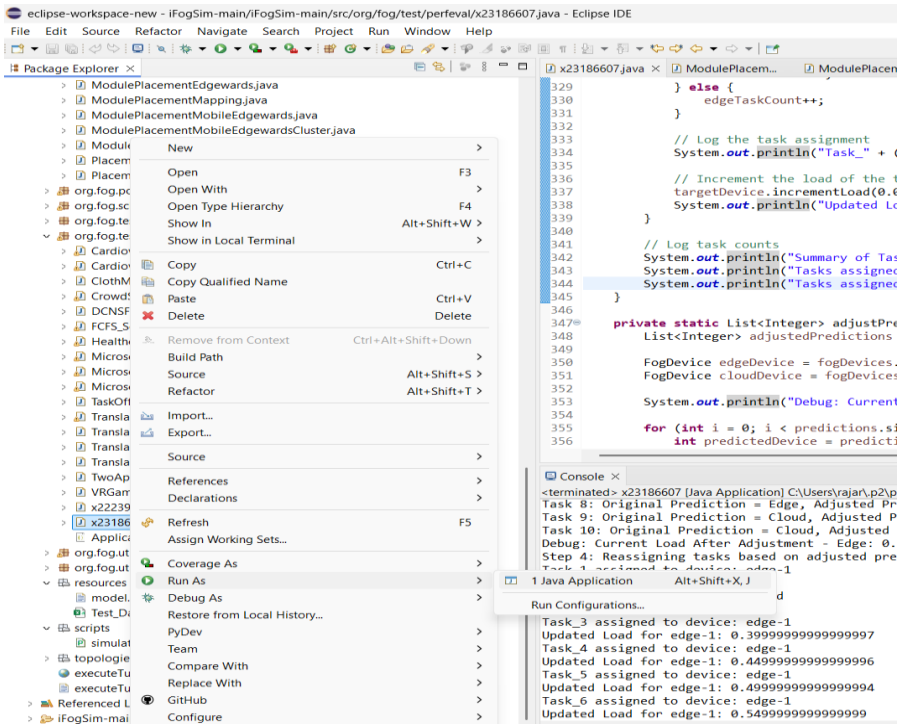

Figure 16: iFogSim Results

References

Gupta, H., Vahid Dastjerdi, A., Ghosh, S.K. and Buyya, R., 2017. iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments. *Software: Practice and Experience*, *47*(9), pp.1275-1296.

D. M. Budden, P. Wang, O. Obst and M. Prokopenko, "RoboCup Simulation Leagues: Enabling Replicable and Robust Investigation of Complex Robotic Systems," in IEEE Robotics & Automation Magazine, vol. 22, no. 3, pp. 140-146, Sept. 2015, doi: 10.1109/MRA.2015.2446911.

R. Daniels, "The Impacts on Management of Electromagnetic Compatibility," 1964 6th National Symposium Electromagnetic Compatibility, Los Angeles, CA, USA, 1964, pp. 1-2, doi: 10.1109/ISEMC.1964.7565199.