National College of Ireland

Configuration Manual

MSc Research Project
Cloud Computing

AbdulJalil Lotfi
Student ID: 22241388

School of Computing
National College of Ireland

Supervisor: Dr. Ahmed Makki

National College of Ireland
Project Submission Sheet
School of Computing

National College of Ireland

Student Name:	AbdulJalil Lotfi
Student ID:	22241388
Programme:	Cloud Computing
Year:	2024
Module:	MSc Research Project
Supervisor:	Dr. Ahmed Makki
Submission Due Date:	03/01/2025
Project Title:	Configuration Manual
Word Count:	730
Page Count:	7

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:	AbdulJalil Lotfi
Date:	3rd of January 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

AbdulJalil Lotfi
22241388

1 Introduction

In our initial setup in the development of this Thesis Project an Amazon SageMaker Notebook has been created to assist us with data preprocessing and preparation stages. In this notebook, various libraries were imported to support data analysis and manipulation. After that, Lambda function was developed to integrate it with the model endpoint that has been created. This Lambda function plays an important role in serving the model inference requests. Finally to allow external access to the model, an API Gateway was introduced, acting as an interface from the virtual network to trigger the lambda function.

2 Amazon SageMaker AI

2.1 NoteBook Creation

Below figure presents the successful creation of SageMaker Notebook with a sufficient computing resources.Figure 1.

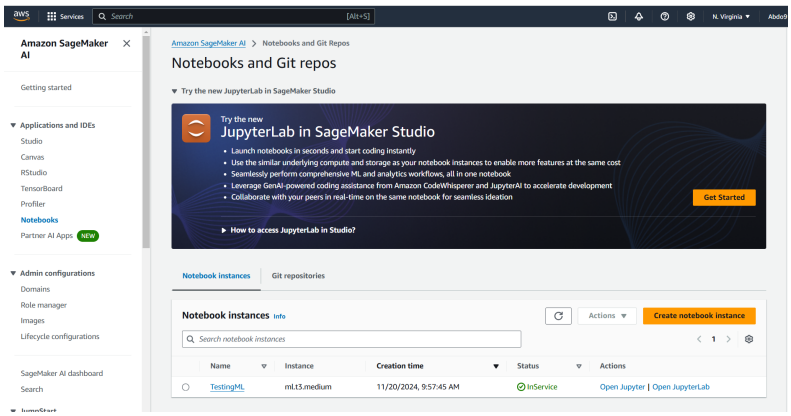


Figure 1: SageMaker NoteBook.

2.2 Data Collection

The dataset was collected from our Virtual Network and can be accessed through below GitHub link.

https://github.com/Abdu-AJ/ORAN.git

2.3 Data Exploration

All the Python libraries required to implement the entire project are listed in below figure Figure 2 and s3 bucket name.

Figure 2.

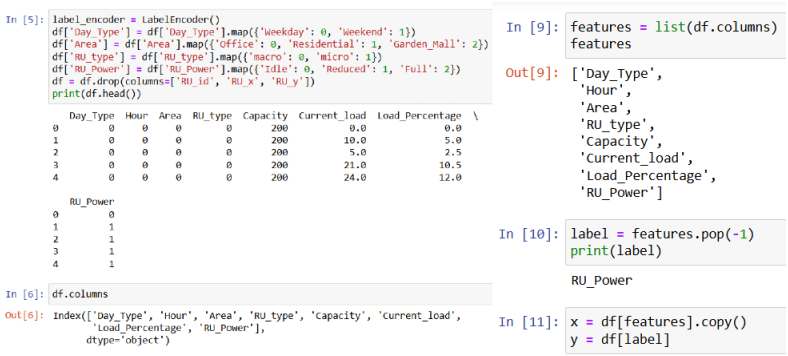
```
In [1]: import numpy as np
        from sagemaker import get_execution_role
        import sagemaker
        from sklearn.model_selection import train_test_split
        from sklearn.preprocessing import LabelEncoder, StandardScaler, OneHotEncoder
        import datetime
        import time
        import tarfile
        import boto3
        import pandas as pd
        from imblearn.over_sampling import SMOTE
        from sklearn.ensemble import RandomForestClassifier
        from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, precision_score, recall_score, f1_score,
        roc_curve, auc
        import sklearn
        import joblib
        import pathlib
        from io import StringIO
        import argparse
        import os
        from sagemaker.sklearn.estimator import SKLearn
        from sagemaker.sklearn.model import SKLearnModel
        from time import gmtime, strftime


        sm_boto3 = boto3.client("sagemaker")
        sess = sagemaker.Session()
        region = sess.boto_session.region_name
        bucket = 'bucket-sagemaker-ml'
```

Figure 2: Required Libraries.

2.4 Data Preprocessing and Preparation for SageMaker

Below photos represent data preprocessing workflow. Where the dataset was splited into features and target labels, then divided into training and testing sets using an 80-20 split. After that to address class imbalance, SMOTE (Synthetic Minority Oversampling Technique) is applied to the training data. Finally resampled training and testing datasets are saved as CSV files and uploaded to Amazon S3, making them accessible for SageMaker trainingFigure 3Figure 4.



```
In [5]: label_encoder = LabelEncoder()
        df['Day_Type'] = df['Day_Type'].map({'Weekday': 0, 'Weekend': 1})
        df['Area'] = df['Area'].map({'Office': 0, 'Residential': 1, 'Garden_Mall': 2})
        df['RU_type'] = df['RU_type'].map({'macro': 0, 'micro': 1})
        df['RU_Power'] = df['RU_Power'].map({'Idle': 0, 'Reduced': 1, 'Full': 2})
        df = df.drop(columns=['RU_id', 'RU_x', 'RU_y'])
        print(df.head())
```

	Day_Type	Hour	Area	RU_type	Capacity	Current_load	Load_Percentage	\
0	0	0	0	0	200	0.0	0.0	
1	0	0	0	0	200	10.0	5.0	
2	0	0	0	0	200	5.0	2.5	
3	0	0	0	0	200	21.0	10.5	
4	0	0	0	0	200	24.0	12.0	

	RU_Power
0	0
1	1
2	1
3	1
4	1

```
In [6]: df.columns
Out[6]: Index(['Day_Type', 'Hour', 'Area', 'RU_type', 'capacity', 'Current_load',
               'Load_Percentage', 'RU_Power'],
              dtype='object')
```

```
In [9]: features = list(df.columns)
        features
Out[9]: ['Day_Type',
         'Hour',
         'Area',
         'RU_type',
         'Capacity',
         'Current_load',
         'Load_Percentage',
         'RU_Power']
```

```
In [10]: label = features.pop(-1)
         print(label)
         RU_Power
```

```
In [11]: x = df[features].copy()
         y = df[label]
```

Figure 3: Data Preprocessing and Preparation.

2.5 Random Forest Model Training and Deployment Script

The script uses the provided data set to train a Random Forest Classifier using hyper-parameters and save the trained model for deployment. The steps followed in this script were to preprocesses the data, train the model, evaluate its performance on test data,

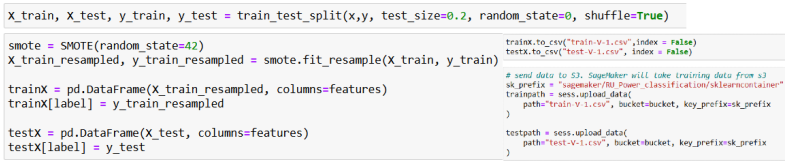


Figure 4: Data Preprocessing and Preparation.

and provide key metrics such as accuracy and classification report. These steps were implemented by the example provided in Spidy20 (2023)

Our Project Version can be found on—https://github.com/Abdu-AJ/ORAN.git

2.6 Configuring and Launching a SageMaker Training Job

Configuring and launching a SageMaker training job using the Scikit-learn framework by specifying the training script. Additionally, the job trains the model in the cloud using the provided training and testing data from S3 Figure 5.

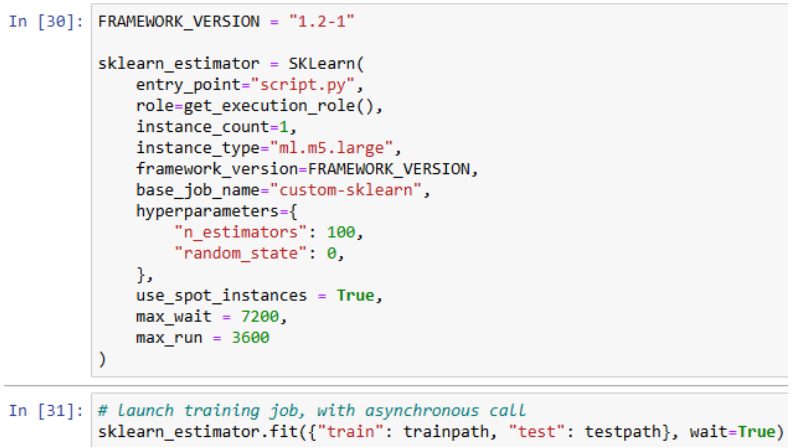


Figure 5: SageMaker Training Job.

2.7 Model Retrieval, Deployment, and Endpoint Creation in SageMaker

Below final steps are used to handle the model being trained in SageMaker. Firstly, it retrieves the model, then creates a SageMaker model using this artifact. Finally, the model is deployed to a SageMaker endpoint to serve the model real-time predictions Figure 6.

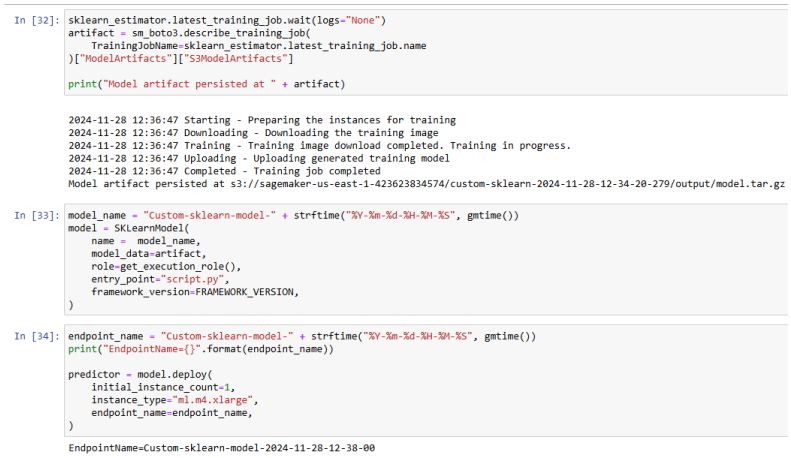


Figure 6: Model retrieval, creation, and deployment.

3 Lambda

3.1 Lambda Function Creation

First chose the suitable Lambda Function name after that chose python 3.12 version as show in below Figure 7.

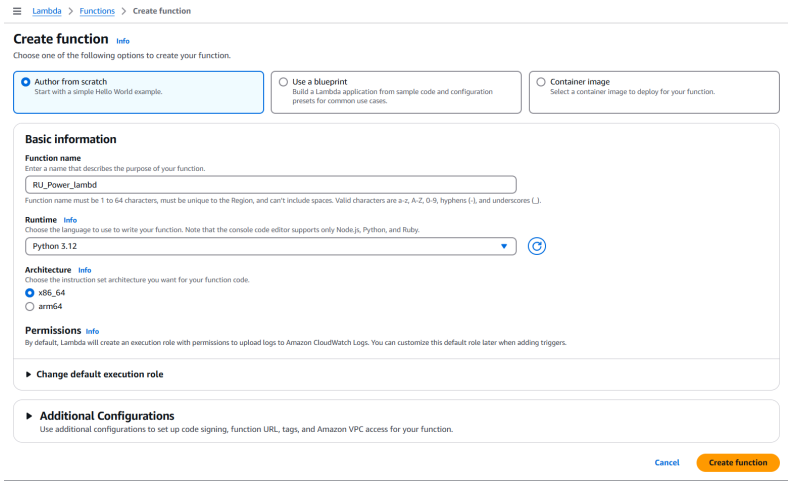


Figure 7: Lambda Function Creation

3.2 Updating Lambda Function Role

First we have to find the Role that is being used by our lambda function as shown in Figure 8. Then under the IAM role section select the role and press the policy as shown below Figure 9. Finally, edit it Figure 10 as advised by Amazon's official website Amazon Web Services (2024) Figure 11.

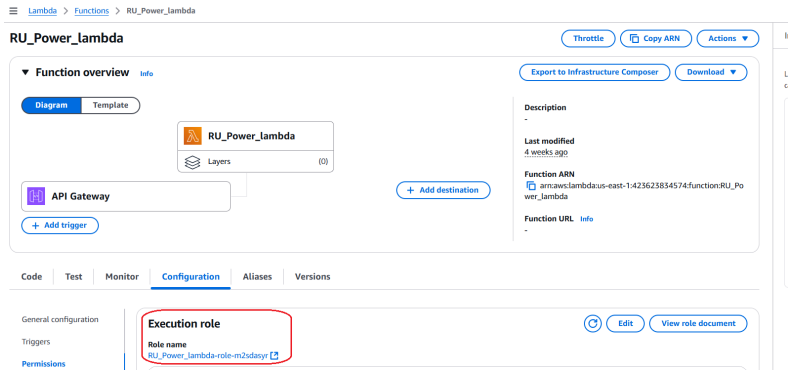


Figure 8: Lambda Role Update

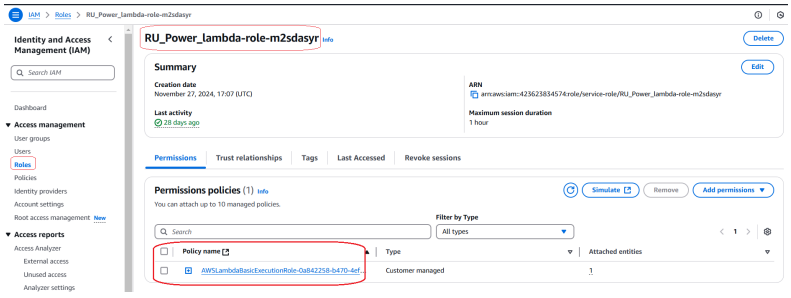


Figure 9: IAM Role Update

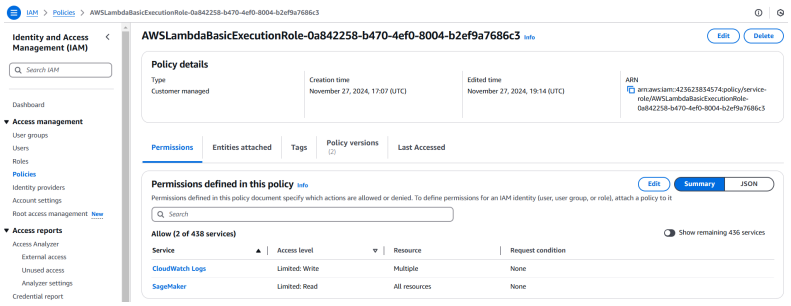


Figure 10: IAM Role Update

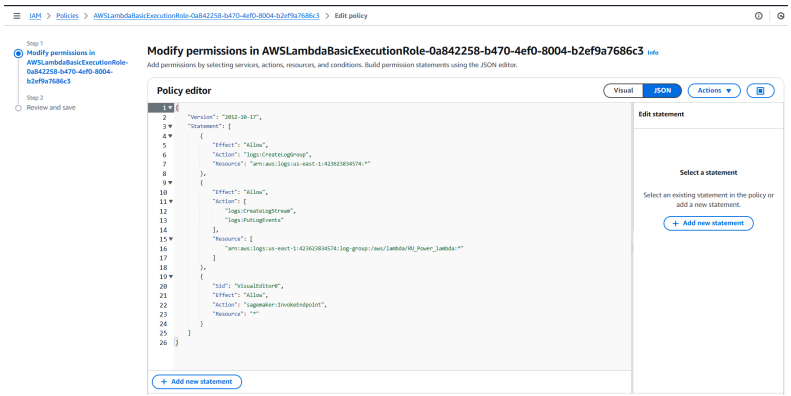


Figure 11: IAM Role Update

4 API Gateway

4.1 API Creation

REST API Creation Figure 12.

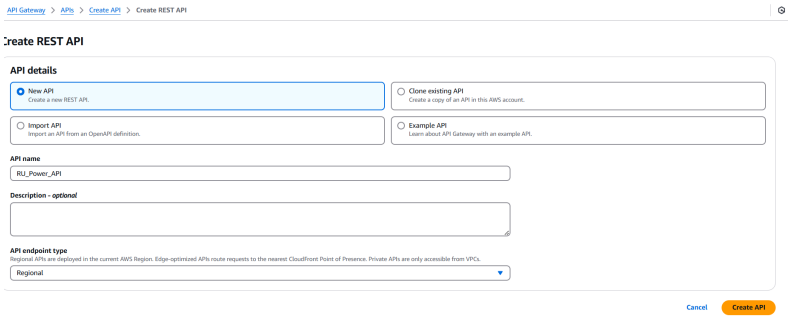


Figure 12: API Creation

4.2 Resource, Method and Deploying Creation

First we have to create a resource Figure 13 then a POST method under this resource. We dont have to forget to connect it to the lambda function as shown in Figure 14. Finally we have to deploy it so we would be able to trigger it Figure 15.

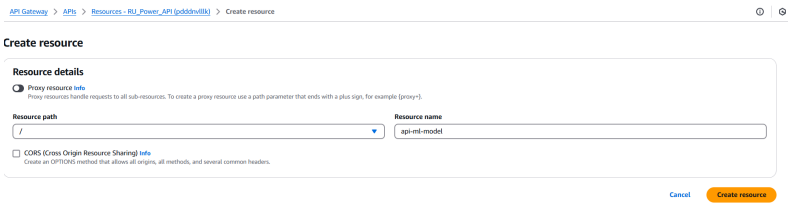


Figure 13: Resource Creation

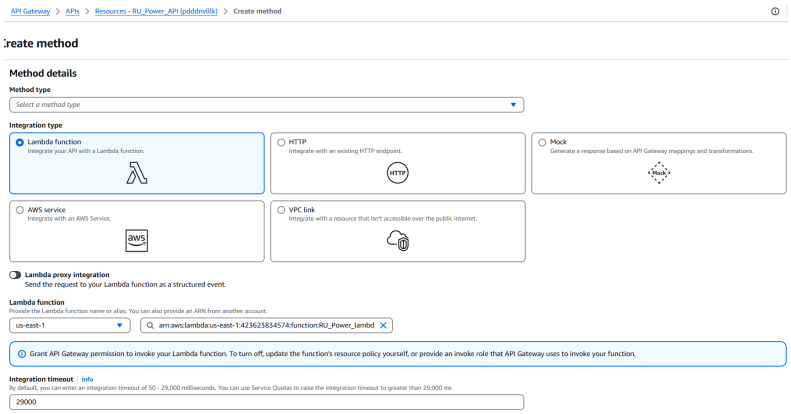


Figure 14: Method Creation

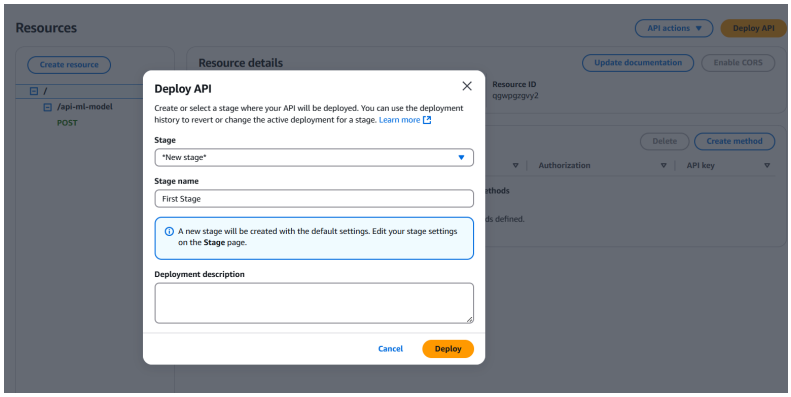


Figure 15: Deployment

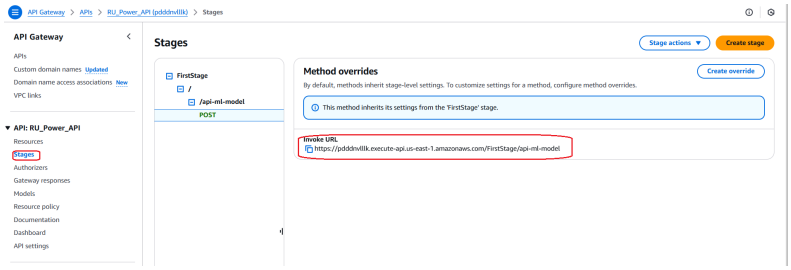


Figure 16: API GateWay URL

5 Virtual ORAN Network

The Virtual Open RAN network simulation code can be found at this link—https://github.com/Abdu-AJ/ORAN.git

References

Amazon Web Services (2024). *Amazon SageMaker Documentation: SageMaker Roles.* **URL:** *https://docs.aws.amazon.com/sagemaker/latest/dg/sagemaker-roles.html*

Spidy20 (2023). Sagemaker tutorials - tutorial 1: Sagemaker sklearn custom script mode, `https://github.com/Spidy20/Sagemaker-Tutorials/tree/master/Tutorial%20-%201%20Sagemaker%20SKLearn%20Custom%20Script%20Mode`.