National College of Ireland

Configuration Manual

MSc Research Project
Cloud Computing

Kotagiri Venkata Durga Abhinav
Student ID: 23344997

School of Computing
National College of Ireland

Supervisor: Shaguna Gupta

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Kotagiri Venkata Durga Abhinav
Student ID:	23344997
Programme:	Cloud Computing
Year:	2024-2025
Module:	MSc Research Project
Supervisor:	Shaguna Gupta
Submission Due Date:	15/09/2025
Project Title:	Configuration Manual
Word Count:	297
Page Count:	5

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:	Kotagiri Venkata Durga Abhinav
Date:	15th September 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

Kotagiri Venkata Durga Abhinav
23344997

1 Introduction

This Configuration Manual will help to set up this project, from installing the required operating system on the machines, which are Raspberry Pi, it also guides to installation of K3s, deploying a custom scheduler, and deployment of the cluster.
GitHub URL: `https://github.com/aabhinnav1999/k8s_research`

2 Raspberry Pi Setup

This section outlines the Raspberry Pi setup, including which Operating system to install. Connect the Raspberry Pi memory card to the personal computer and go to the official Raspberry Pi Software website Raspberry Pi Foundation (2025), download the software, and install Ubuntu 24.04 to the memory card. (Be careful while installing the OS into the memory card; it will format the memory card before installing the OS). After installation, select the default settings on the Raspberry Pi and complete the setup by allowing port 22 on all the machines.

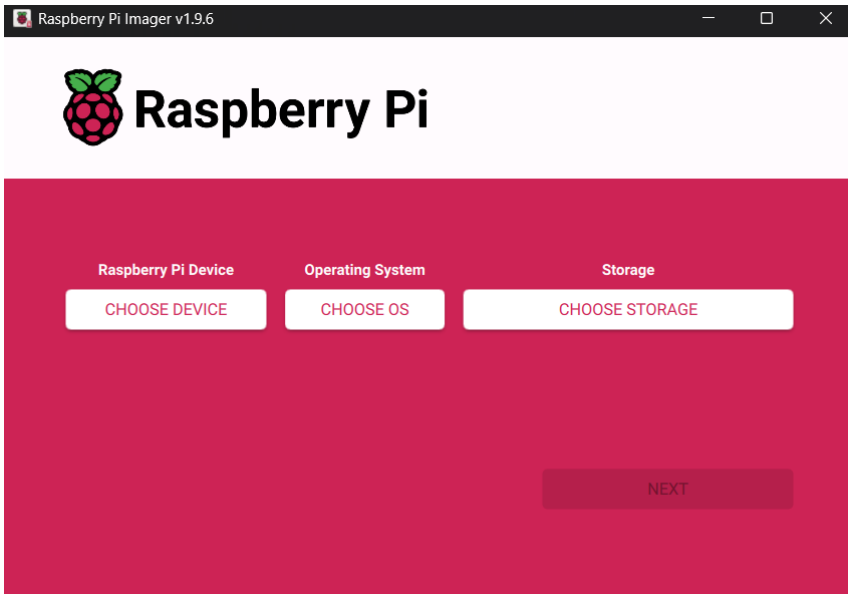

Figure 1: Raspberry Pi Software

3 Install K3s

Install the K3s from K3s Project (2019) on the master node. In worker nodes, install K3s from this command:

`curl -sfL https://get.k3s.io | K3S_URL=https://master-node-ip-address:6443 K3S_TOKEN=master-node-token sh -`

Replace master-node-ip-address and master-node-token. We can get the token by using:

`sudo cat /var/lib/rancher/k3s/server/node-token`

Now check if all the nodes are ready or not in the cluster:

`kubectl get nodes`

```
abhinav@abhinav-master:~$ kubectl get nodes
NAME               STATUS   ROLES                  AGE   VERSION
abhinav-master     Ready    control-plane,master   11d   v1.33.3+k3s1
abhinav-worker-1   Ready    worker-1               11d   v1.33.3+k3s1
abhinav-worker-2   Ready    worker-2               11d   v1.33.3+k3s1
abhinav@abhinav-master:~$
```

Figure 2: kubectl get nodes

Install K3s in large clusters using:

`curl -sfL https://get.k3s.io | K3S_URL=https://master-node-ip-address:6443 K3S_TOKEN=master-node-token sh -`

4 Install Prometheus and Grafana

First, install Helm on the master node, and then install Prometheus and Grafana Grafana Labs (2013) using Helm charts.

commands are:

```
Sudo snap install helm --classic
helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
helm repo update
helm install prometheus prometheus-community/prometheus
helm repo add grafana https://grafana.github.io/helm-charts
helm repo update
helm install grafana grafana/grafana
```

5 Deploy Custom Scheduler

5.1 Custom Scheduler

Deploy the custom scheduler using the YAML file. The Docker image is stored in the aabhinnavdocker/custom-scheduler repository on Docker Hub Docker, Inc. (2013).

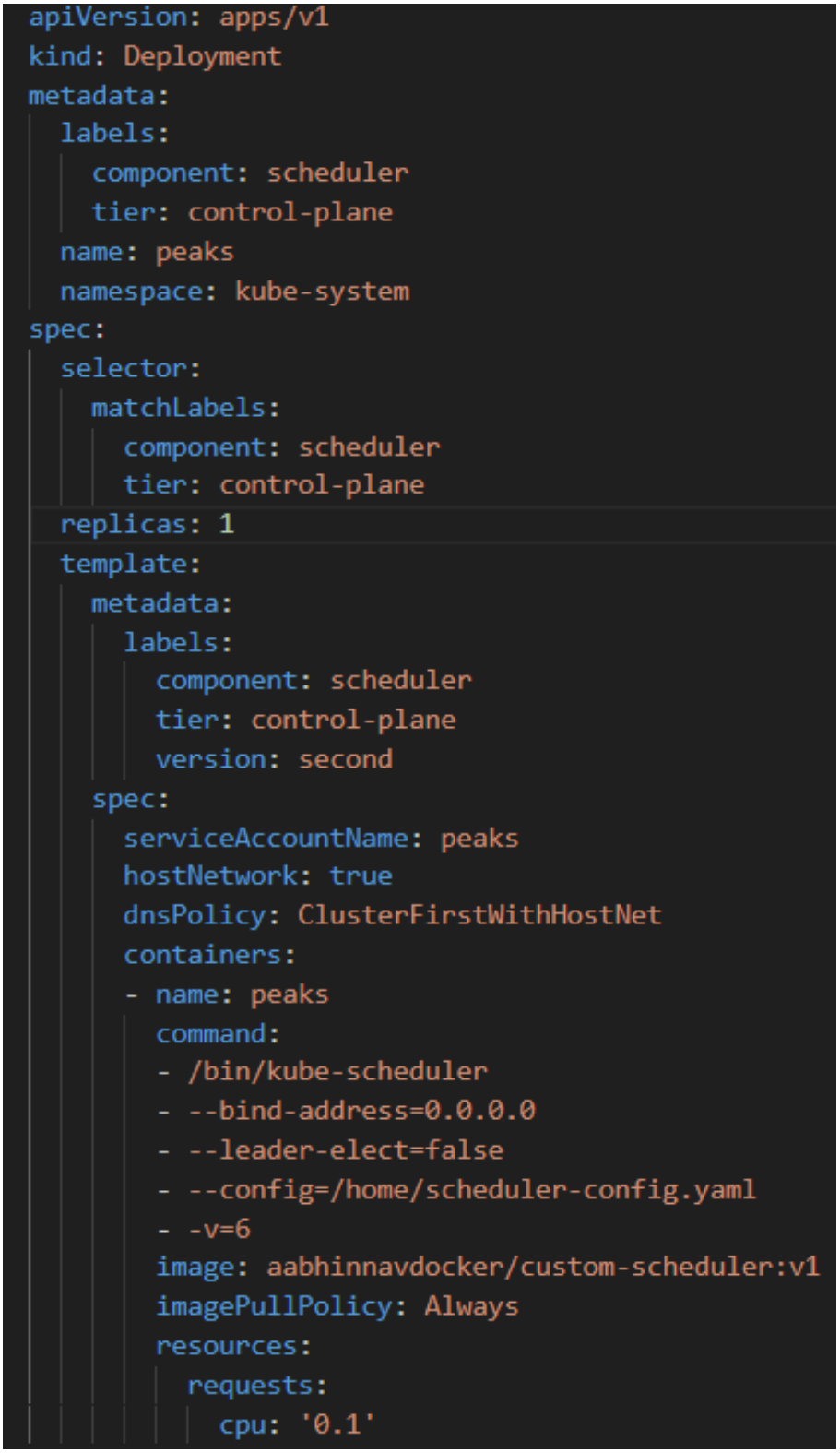

Figure 3: Yaml Deployment

5.2 Install Kepler

Install Kepler Sustainable Computing (2025), which is a Prometheus exporter that measures energy consumption metrics at the container, pod, and node levels in Kubernetes

clusters.

```
NAMESPACE     NAME                 READY   STATUS    RESTARTS       AGE
kepler        kepler-bj6bz         1/1     Running   9 (19h ago)    10d
kepler        kepler-qsfx8         1/1     Running   9 (36m ago)    10d
```

Figure 4: Kepler

5.3 Install Loadwatcher

Deploy the custom scheduler using the Kubernetes scheduler plugin framework. Load watcher (2025)

```
NAME                                      READY   STATUS    RESTARTS       AGE
load-watcher-deployment-786657c755-rblct  1/1     Running   4 (37m ago)    6d16h
```

Figure 5: Loadwatcher

6 Test Custom Scheduler

Deploy a stress-ng deployment manifest using the kubectl command; the image should be ARM64 compatible. Here colinianking/stress-ng Colin Ian King (2015) image is used, which is an open source image and supported on ARM architecture. Because of this deployment, one of the worker nodes will consume more power, and therefore, the custom scheduler will now deploy the workloads to the other worker node.

Figure 6: Nodes Power Usage

```
abhinav@abhinav-master:~/k8s$ kubectl get po -o wide
NAME                                          READY   STATUS    RESTARTS   AGE     IP            NODE              NOMINATED NODE   READINESS GATES
cpu-stress-default-phase2-v5rhv               1/1     Running   0          7m49s   10.42.2.20    abhinav-worker-1  <none>           <none>
cpu-stress-default-phase3-fdxmn               1/1     Running   0          6m35s   10.42.2.21    abhinav-worker-1  <none>           <none>
cpu-stress-default-phase4-5j2dl               1/1     Running   0          6m32s   10.42.2.22    abhinav-worker-1  <none>           <none>
nginx-deployment-default-7989c6489c-426sf     1/1     Running   0          2m55s   10.42.2.23    abhinav-worker-1  <none>           <none>
nginx-deployment-default-7989c6489c-4xmwg     1/1     Running   0          2m55s   10.42.2.27    abhinav-worker-1  <none>           <none>
nginx-deployment-default-7989c6489c-785qc     1/1     Running   0          2m55s   10.42.4.217   abhinav-worker-2  <none>           <none>
nginx-deployment-default-7989c6489c-78f5x     1/1     Running   0          2m55s   10.42.4.214   abhinav-worker-2  <none>           <none>
nginx-deployment-default-7989c6489c-ffzw8     1/1     Running   0          2m55s   10.42.2.24    abhinav-worker-1  <none>           <none>
nginx-deployment-default-7989c6489c-jzdhx     1/1     Running   0          2m55s   10.42.4.216   abhinav-worker-2  <none>           <none>
nginx-deployment-default-7989c6489c-tblqq     1/1     Running   0          2m55s   10.42.4.215   abhinav-worker-2  <none>           <none>
nginx-deployment-default-7989c6489c-thz67     1/1     Running   0          2m55s   10.42.2.26    abhinav-worker-1  <none>           <none>
nginx-deployment-default-7989c6489c-z89vx     1/1     Running   0          2m55s   10.42.2.25    abhinav-worker-1  <none>           <none>
```

Figure 7: Default Scheduler Deployment

```
nginx-deployment-peaks-557f4cbc4f-8jkpv   1/1     Running   0          23s   10.42.2.32    abhinav-worker-1  <none>           <none>
nginx-deployment-peaks-557f4cbc4f-bndck   1/1     Running   0          23s   10.42.2.33    abhinav-worker-1  <none>           <none>
nginx-deployment-peaks-557f4cbc4f-g6lns   1/1     Running   0          23s   10.42.4.224   abhinav-worker-2  <none>           <none>
nginx-deployment-peaks-557f4cbc4f-k8ps7   1/1     Running   0          23s   10.42.4.228   abhinav-worker-2  <none>           <none>
nginx-deployment-peaks-557f4cbc4f-kfxnh   1/1     Running   0          23s   10.42.4.226   abhinav-worker-2  <none>           <none>
nginx-deployment-peaks-557f4cbc4f-l4mrt   1/1     Running   0          23s   10.42.4.225   abhinav-worker-2  <none>           <none>
nginx-deployment-peaks-557f4cbc4f-m7h2x   1/1     Running   0          23s   10.42.4.227   abhinav-worker-2  <none>           <none>
nginx-deployment-peaks-557f4cbc4f-xgwxs   1/1     Running   0          23s   10.42.4.229   abhinav-worker-2  <none>           <none>
nginx-deployment-peaks-557f4cbc4f-zb4b7   1/1     Running   0          23s   10.42.4.223   abhinav-worker-2  <none>           <none>
```

Figure 8: Custom Scheduler Deployment

References

Colin Ian King (2015). stress-ng: This is the stress-ng upstream project git repository, https://github.com/ColinIanKing/stress-ng. Accessed: 25 August 2025.

Docker, Inc. (2013). Docker hub, https://hub.docker.com/. Accessed: 25 August 2025.

Grafana Labs (2013). Grafana: The open and composable observability platform, https://grafana.com/. Accessed: 25 August 2025.

K3s Project (2019). Installation, https://docs.k3s.io/installation. Last updated: 18 August 2025; Accessed: 25 August 2025.

Load watcher (2025). Load watcher: cluster-wide aggregator of resource usage metrics, https://github.com/paypal/load-watcher. Accessed: 25 August 2025.

Raspberry Pi Foundation (2025). Raspberry pi software, https://www.raspberrypi.com/software/. Accessed: 25 August 2025.

Sustainable Computing (2025). Kepler (kubernetes-based efficient power level exporter), https://github.com/sustainable-computing-io/kepler. Accessed: 25 August 2025.