**Pramodkumar Upadhyay**

Course: Msc in DevOps,

Institute of Technology, Tallaght

Blessington Rd, Tallaght, Dublin 24

Student ID: X00159360

Email:Pramod\_ppr@yahoo.com

Table of Contents

[**1.** **CA-Part1** 2](#_Toc39504580)

[1.1 Synchronous: 2](#_Toc39504581)

[1.2 Asynchronous: 3](#_Toc39504582)

[1.3 Deployment on Google Kubernetes Engine: 3](#_Toc39504583)

[**2** **CA-Part2** 5](#_Toc39504584)

[2.1 Avg Response Time : Apache JMeter tool settings 5](#_Toc39504585)

[2.2 Average Response Time Graph 7](#_Toc39504586)

[2.3 Average Recovery Time Graph 8](#_Toc39504587)

[2.4 Function on GKE 9](#_Toc39504588)

[**3** **Image Repository** 10](#_Toc39504589)

[**4** **Manifest and Shell scripts** 12](#_Toc39504590)

# **CA-Part1**

We have created following two systems using microservices deployed on GKE. All the images are build and stored on docker hub repository.

* 1. Synchronous:

The system contains main service called the allthenews(atn) with two helper services, namely, weatherfetcher(wf) and newsfetcher(nf). All the services are deployed onto Google Kubernetes Engine. In addition to that the main service has also called a function to get the sport news.

|  |
| --- |
| Allthenews Service:  With argument; style:colourful    Fig.1:Allthenews service with style=colourful |
| Allthenews Service:  With argument; style:plain |

Fig.2:Allthenews service with style=plain

* 1. Asynchronous:

The system consists of 3 door services along with allthenews service.

|  |
| --- |
|  |

Fig.3:Allthenews and door service running together

## **Deployment on Google Kubernetes Engine:**

The services allthenews, newsfectcher, weather news, seccon, door and functions are deployed on GKE.

|  |
| --- |
|  |
| Fig.4:Service Deployment on GKE |

# **CA-Part2**

The microservices services are already hosted on GKE, refer the Fig4.

## Avg Response Time : Apache JMeter tool settings

To get the average response time for the asynchronous and synchronous microservices we have used Apache JMeter tool. The test plan uses 10 users and 20 cycle to produce enough data for the testing. In the below screen shoot “Number of threads” represent “number of users” and “Loop count” represent number times test to be executed.

|  |
| --- |
|  |

Fig.5: Avg Response time testing configuration on JMeter tool

In the following two figures we have setup the door and allthenews microservices for http request to get the response time.

|  |
| --- |
|  |

Fig.6: Http request setup for door service

|  |
| --- |
|  |

Fig.7: Http request setup for allthenews service

In the below screenshot we can see the response from the site as the http request output.

|  |
| --- |
|  |

Fig.8: Response for door service

|  |
| --- |
|  |

Fig.9: Response for allthenews service

The below figure shows the bar chart comparison for average response time for door and allthenews services.

## Average Response Time Graph

|  |
| --- |
|  |

Fig.10: Average response time for Door and Allthenews service

## Average Recovery Time Graph

We have used bash shell script to find out the average recovery time. The shell script provides the pod startup time. The average has been calculated by executing the script 4 time and the sum of each output is divided by 4. A standard script has been used to get the pod startup time which is defined as below. The script is used for the “door1-deployment”. By replacing the deployment name we can get the pod startup time.

|  |
| --- |
| #!/bin/bash  echo "Set replicas=0 to delete the pod"  startTime=$(date -u +%Y-%m-%dT%H:%M:%SZ)  kubectl scale deploy door1-deployment --replicas=0  echo "Set relicas=1 to create the pod"  kubectl scale deploy door1-deployment --replicas=1  echo "Sleep for 10 sec to run the container"  sleep 10  newPod=$(kubectl get pods | grep "door1-deployment" | awk '{print $1}')  newPodReadyTime=$(kubectl get pod $newPod -o json | jq -r '.status.containerStatuses[0].state.running.startedAt')  echo "Pod deletion time"  echo $startTime  echo "New Pod Ready time"  echo $newPodReadyTime  t=$(date -d $newPodReadyTime +%s)  t1=$(date -d $startTime +%s)  diff=$(expr $t - $t1)  echo "Pod Uptime in seconds:"$diff |

The data is then passed to google function to plot the bar chart as shown in fig.11.

Following shell script has been used to call the function.

|  |
| --- |
| #!/bin/bash  curl -i -H "Accept: application/json" -H "Content-Type:application/json" -X POST --data '{"filename": "average\_recovery\_time.png","plottype": "bar","x": ["A-sync:nf","B-sync:wf","G-sync:atn","A-async:d1","B-async:d2","G-async:scn"],"y": [2.75,2.5,3,3.5,2.5,1.5],"ylab": "Average recovery"}' https://us-central1-eades-275104.cloudfunctions.net/fngraph |

|  |
| --- |
|  |

Fig.11: Average recovery time for sync and async services

## Function on GKE

|  |
| --- |
| Fig.12: fngraph on GKE used for above graph plot |
| Fig.13: Fngraph URL |

# **Image Repository**

All the images are stored in Docker hub.

|  |
| --- |
|  |

Fig.14: Image Repository

# **Manifest and Shell scripts**

|  |  |
| --- | --- |
| File Name | Script |
| seccon.yaml | apiVersion: v1  kind: Service  metadata:  name: seccon-service  labels:  app: seccon-service  spec:  ports:  - port: 9090  protocol: TCP  targetPort: 8080  nodePort: 31080  selector:  app: seccon  type: NodePort  ---  apiVersion: apps/v1  kind: Deployment  metadata:  name: seccon-deployment  spec:  replicas: 1  selector:  matchLabels:  app: seccon  template:  metadata:  labels:  app: seccon  spec:  containers:  - name: seccon  image: pramodppr/seccon:v1  args: ["redis-service:6379", "2000", "50", "door1", "door2", "door3"]  ports:  - containerPort: 8080 |
| deployment\_d1.yaml | apiVersion: apps/v1  kind: Deployment  metadata:  name: door1-deployment  spec:  replicas: 1  selector:  matchLabels:  app: door1  template:  metadata:  labels:  app: door1  spec:  containers:  - name: door1  args: ["1", "10", "redis-service:6379"]  image: pramodppr/door:v1 |
| deployment\_d2.yaml | apiVersion: apps/v1  kind: Deployment  metadata:  name: door2-deployment  spec:  replicas: 1  selector:  matchLabels:  app: door2  template:  metadata:  labels:  app: door2  spec:  containers:  - name: door2  args: ["2", "20", "redis-service:6379"]  image: pramodppr/door2:v1 |
| deployment\_d3.yaml | apiVersion: apps/v1  kind: Deployment  metadata:  name: door3-deployment  spec:  replicas: 1  selector:  matchLabels:  app: door3  template:  metadata:  labels:  app: door3  spec:  containers:  - name: door3  args: ["3", "30", "redis-service:6379"]  image: pramodppr/door3:v1 |
| deployment\_atn2.yaml | apiVersion: apps/v1  kind: Deployment  metadata:  name: atn-deployment  spec:  replicas: 1  selector:  matchLabels:  app: atn  template:  metadata:  labels:  app: atn  spec:  containers:  - name: atn  image: pramodppr/allthenews:v2  args: ["news", "http://nf-service:8888", "weather", "http://wf-service:8888"]  ports:  - containerPort: 8080 |
| deployment\_atn3.yaml | apiVersion: apps/v1  kind: Deployment  metadata:  name: atn-deployment  spec:  replicas: 1  selector:  matchLabels:  app: atn  template:  metadata:  labels:  app: atn  spec:  containers:  - name: atn  image: pramodppr/allthenews:v3  args: ["news", "http://nf-service:8888", "weather", "http://wf-service:8888"]  ports:  - containerPort: 8080  - name: atn-redis  image: redis  ports:  - containerPort: 6379 |
| deployment\_atn4.yaml | apiVersion: apps/v1  kind: Deployment  metadata:  name: atn-deployment  spec:  replicas: 1  selector:  matchLabels:  app: atn  template:  metadata:  labels:  app: atn  spec:  containers:  - name: atn  image: pramodppr/allthenews:v4  args: ["news", "http://nf-service:8888", "weather", "http://wf-service:8888", "sport", "https://us-central1-eades-275104.cloudfunctions.net/fnsport", "door", "http://35.228.237.51:31080/"]  ports:  - containerPort: 8080  - name: atn-redis  image: redis  ports:  - containerPort: 6379 |
| deployment\_nf.yaml | apiVersion: apps/v1  kind: Deployment  metadata:  name: nf-deployment  spec:  replicas: 1  selector:  matchLabels:  app: nf  template:  metadata:  labels:  app: nf  spec:  containers:  - name: nf  image: pramodppr/newsfetcher:v2 |
| deployment\_wf.yaml | apiVersion: apps/v1  kind: Deployment  metadata:  name: wf-deployment  spec:  replicas: 1  selector:  matchLabels:  app: wf  template:  metadata:  labels:  app: wf  spec:  containers:  - name: wf  image: pramodppr/weatherfetcher:v2 |
| service\_atn.yaml | apiVersion: v1  kind: Service  metadata:  name: atn-service  labels:  app: atn-service  spec:  ports:  - port: 9090  protocol: TCP  targetPort: 8080  nodePort: 31916  selector:  app: atn  type: NodePort |
| service\_nf.yaml | apiVersion: v1  kind: Service  metadata:  name: nf-service  labels:  name: nf-service  spec:  ports:  - port: 8888  protocol: TCP  targetPort: 8888  selector:  app: nf  type: ClusterIP |
| service\_wf.yaml | apiVersion: v1  kind: Service  metadata:  name: wf-service  labels:  name: wf-service  spec:  ports:  - port: 8888  protocol: TCP  targetPort: 8888  selector:  app: wf  type: ClusterIP |
| Door1.sh | #!/bin/bash  echo "Set replicas=0 to delete the pod"  startTime=$(date -u +%Y-%m-%dT%H:%M:%SZ)  kubectl scale deploy door1-deployment --replicas=0  echo "Set relicas=1 to create the pod"  kubectl scale deploy door1-deployment --replicas=1  echo "Sleep for 10 sec to run the container"  sleep 10  newPod=$(kubectl get pods | grep "door1-deployment" | awk '{print $1}')  newPodReadyTime=$(kubectl get pod $newPod -o json | jq -r '.status.containerStatuses[0].state.running.startedAt')  echo "Pod deletion time"  echo $startTime  echo "New Pod Ready time"  echo $newPodReadyTime  t=$(date -d $newPodReadyTime +%s)  t1=$(date -d $startTime +%s)  diff=$(expr $t - $t1)  echo "Pod Uptime in seconds:"$diff |
| Door2.sh | #!/bin/bash  echo "Set replicas=0 to delete the pod"  startTime=$(date -u +%Y-%m-%dT%H:%M:%SZ)  kubectl scale deploy door2-deployment --replicas=0  echo "Set relicas=1 to create the pod"  kubectl scale deploy door2-deployment --replicas=1  echo "Sleep for 10 sec to run the container"  sleep 10  newPod=$(kubectl get pods | grep "door2-deployment" | awk '{print $1}')  newPodReadyTime=$(kubectl get pod $newPod -o json | jq -r '.status.containerStatuses[0].state.running.startedAt')  #((podUptime=newPodReadyTime-startTime))  #echo $podUptime  echo "Pod deletion time"  echo $startTime  echo "New Pod Ready time"  echo $newPodReadyTime  t=$(date -d $newPodReadyTime +%s)  t1=$(date -d $startTime +%s)  diff=$(expr $t - $t1)  echo "Pod Uptime in seconds:"$diff |
| Allthenews.sh | #!/bin/bash  echo "Set replicas=0 to delete the pod"  startTime=$(date -u +%Y-%m-%dT%H:%M:%SZ)  kubectl scale deploy atn-deployment --replicas=0  echo "Set relicas=1 to create the pod"  kubectl scale deploy atn-deployment --replicas=1  echo "Sleep for 10 sec to run the container"  sleep 10  newPod=$(kubectl get pods | grep "atn-deployment" | awk '{print $1}')  newPodReadyTime=$(kubectl get pod $newPod -o json | jq -r '.status.containerStatuses[0].state.running.startedAt')  #((podUptime=newPodReadyTime-startTime))  #echo $podUptime  echo "Pod deletion time"  echo $startTime  echo "New Pod Ready time"  echo $newPodReadyTime  t=$(date -d $newPodReadyTime +%s)  t1=$(date -d $startTime +%s)  diff=$(expr $t - $t1)  echo "Pod Uptime in seconds:"$diff |
| nf.sh | #!/bin/bash  echo "Set replicas=0 to delete the pod"  startTime=$(date -u +%Y-%m-%dT%H:%M:%SZ)  kubectl scale deploy nf-deployment --replicas=0  echo "Set relicas=1 to create the pod"  kubectl scale deploy nf-deployment --replicas=1  echo "Sleep for 10 sec to run the container"  sleep 10  newPod=$(kubectl get pods | grep "nf-deployment" | awk '{print $1}')  newPodReadyTime=$(kubectl get pod $newPod -o json | jq -r '.status.containerStatuses[0].state.running.startedAt')  #((podUptime=newPodReadyTime-startTime))  #echo $podUptime  echo "Pod deletion time"  echo $startTime  echo "New Pod Ready time"  echo $newPodReadyTime  t=$(date -d $newPodReadyTime +%s)  t1=$(date -d $startTime +%s)  diff=$(expr $t - $t1)  echo "Pod Uptime in seconds:"$diff |
| Wf.sh | #!/bin/bash  echo "Set replicas=0 to delete the pod"  startTime=$(date -u +%Y-%m-%dT%H:%M:%SZ)  kubectl scale deploy wf-deployment --replicas=0  echo "Set relicas=1 to create the pod"  kubectl scale deploy wf-deployment --replicas=1  echo "Sleep for 10 sec to run the container"  sleep 10  newPod=$(kubectl get pods | grep "wf-deployment" | awk '{print $1}')  newPodReadyTime=$(kubectl get pod $newPod -o json | jq -r '.status.containerStatuses[0].state.running.startedAt')  #((podUptime=newPodReadyTime-startTime))  #echo $podUptime  echo "Pod deletion time"  echo $startTime  echo "New Pod Ready time"  echo $newPodReadyTime  t=$(date -d $newPodReadyTime +%s)  t1=$(date -d $startTime +%s)  diff=$(expr $t - $t1)  echo "Pod Uptime in seconds:"$diff |
| Seccon.sh | #!/bin/bash  echo "Set replicas=0 to delete the pod"  startTime=$(date -u +%Y-%m-%dT%H:%M:%SZ)  kubectl scale deploy seccon-deployment --replicas=0  echo "Set relicas=1 to create the pod"  kubectl scale deploy seccon-deployment --replicas=1  echo "Sleep for 10 sec to run the container"  sleep 10  newPod=$(kubectl get pods | grep "seccon-deployment" | awk '{print $1}')  newPodReadyTime=$(kubectl get pod $newPod -o json | jq -r '.status.containerStatuses[0].state.running.startedAt')  #((podUptime=newPodReadyTime-startTime))  #echo $podUptime  echo "Pod deletion time"  echo $startTime  echo "New Pod Ready time"  echo $newPodReadyTime  t=$(date -d $newPodReadyTime +%s)  t1=$(date -d $startTime +%s)  diff=$(expr $t - $t1)  echo "Pod Uptime in seconds:"$diff |
|  |  |