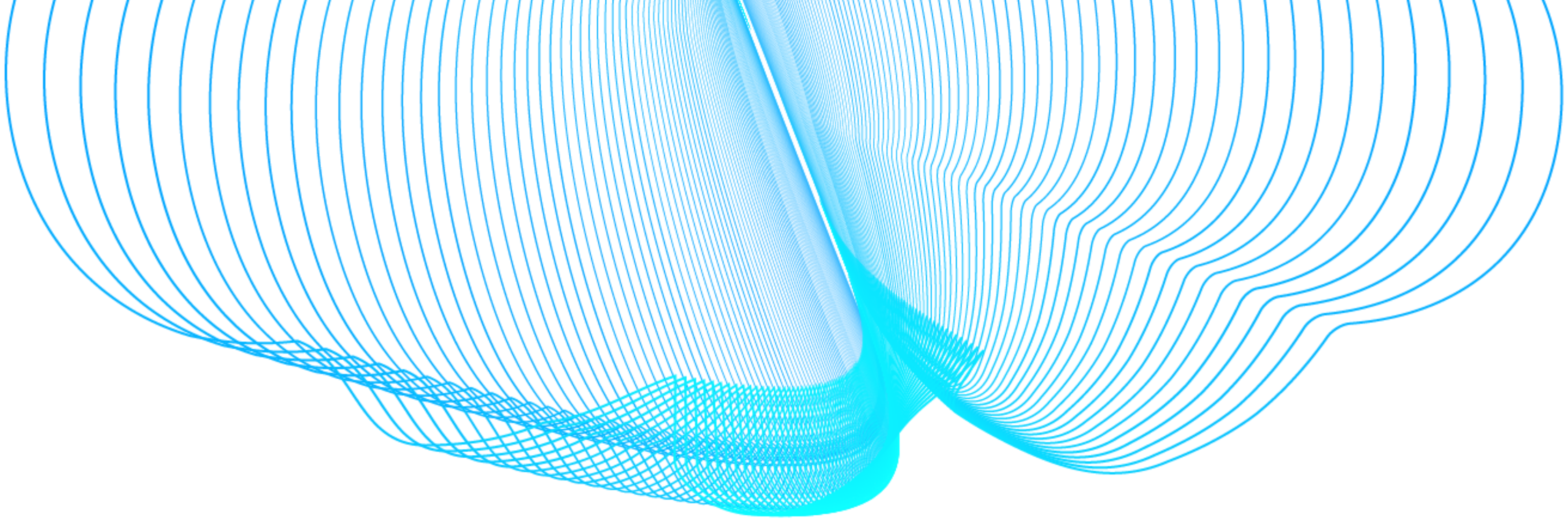


ADA-PIPE adaptation and scheduling

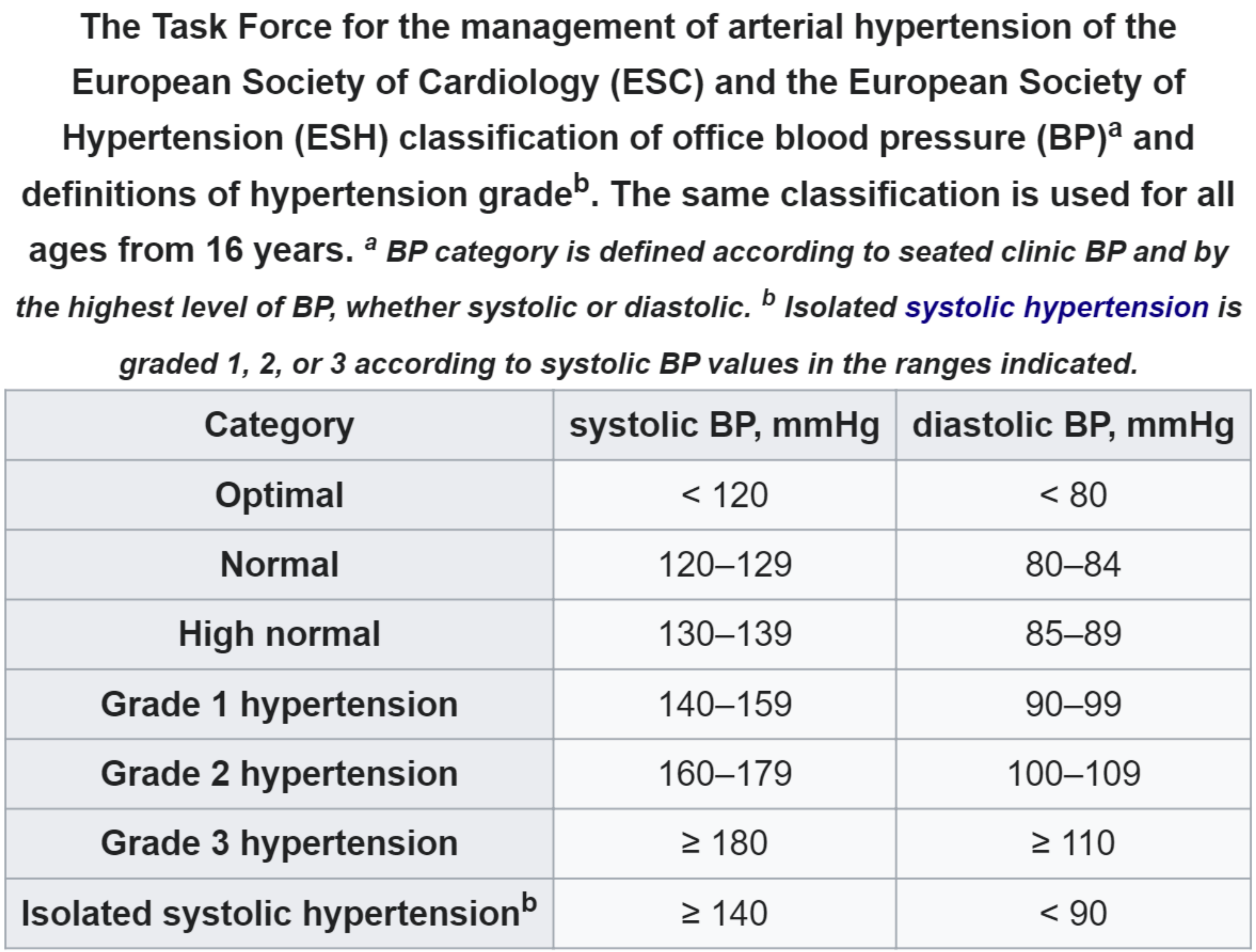
Toolkit tutorial

DCI - WS23

Narges Mehran (doctoral student and project assistant)

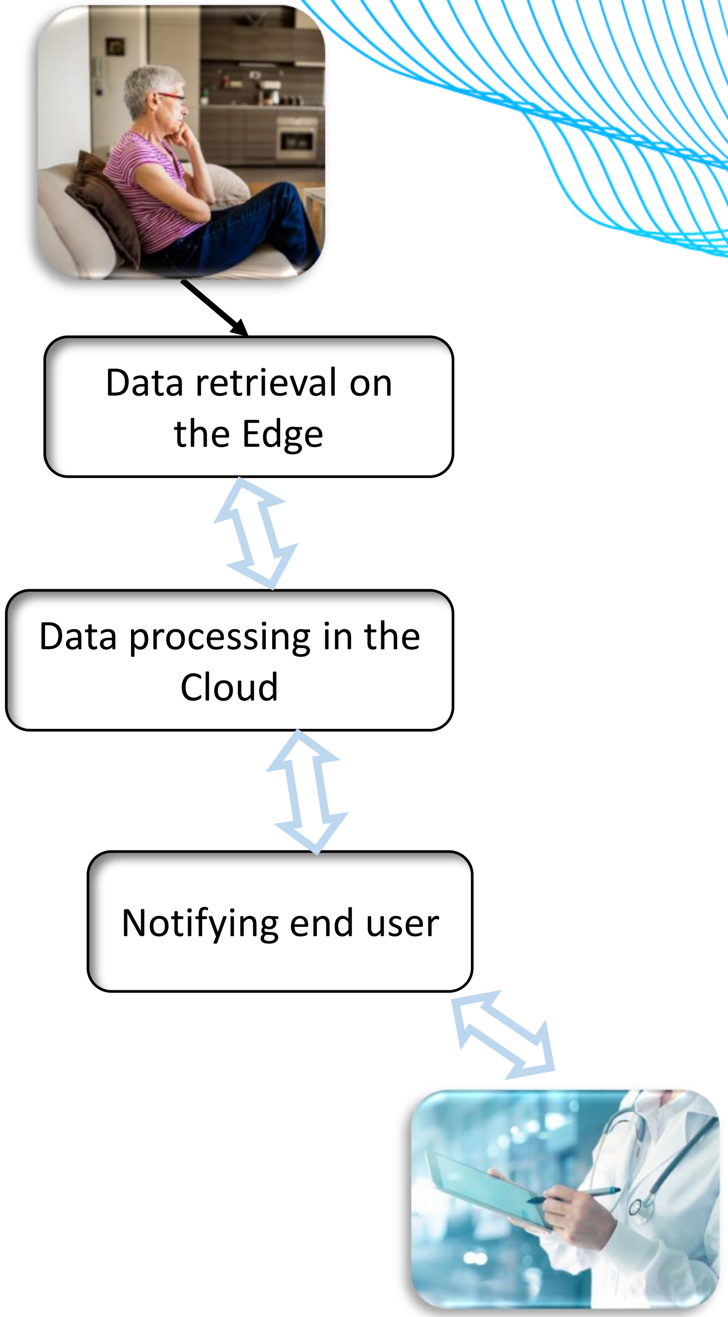
Alpen-Adria-University (AAU)
Austria

04.12.2023



The Task Force for the management of arterial hypertension of the European Society of Cardiology (ESC) and the European Society of Hypertension (ESH) classification of office blood pressure (BP)[a] and definitions of hypertension grade[b]. The same classification is used for all ages from 16 years. *[a] BP category is defined according to seated clinic BP and by the highest level of BP, whether systolic or diastolic. [b] Isolated systolic hypertension is graded 1, 2, or 3 according to systolic BP values in the ranges indicated.*

Category	systolic BP, mmHg	diastolic BP, mmHg
Optimal	< 120	< 80
Normal	120–129	80–84
High normal	130–139	85–89
Grade 1 hypertension	140–159	90–99
Grade 2 hypertension	160–179	100–109
Grade 3 hypertension	≥ 180	≥ 110
Isolated systolic hypertension[b]	≥ 140	< 90

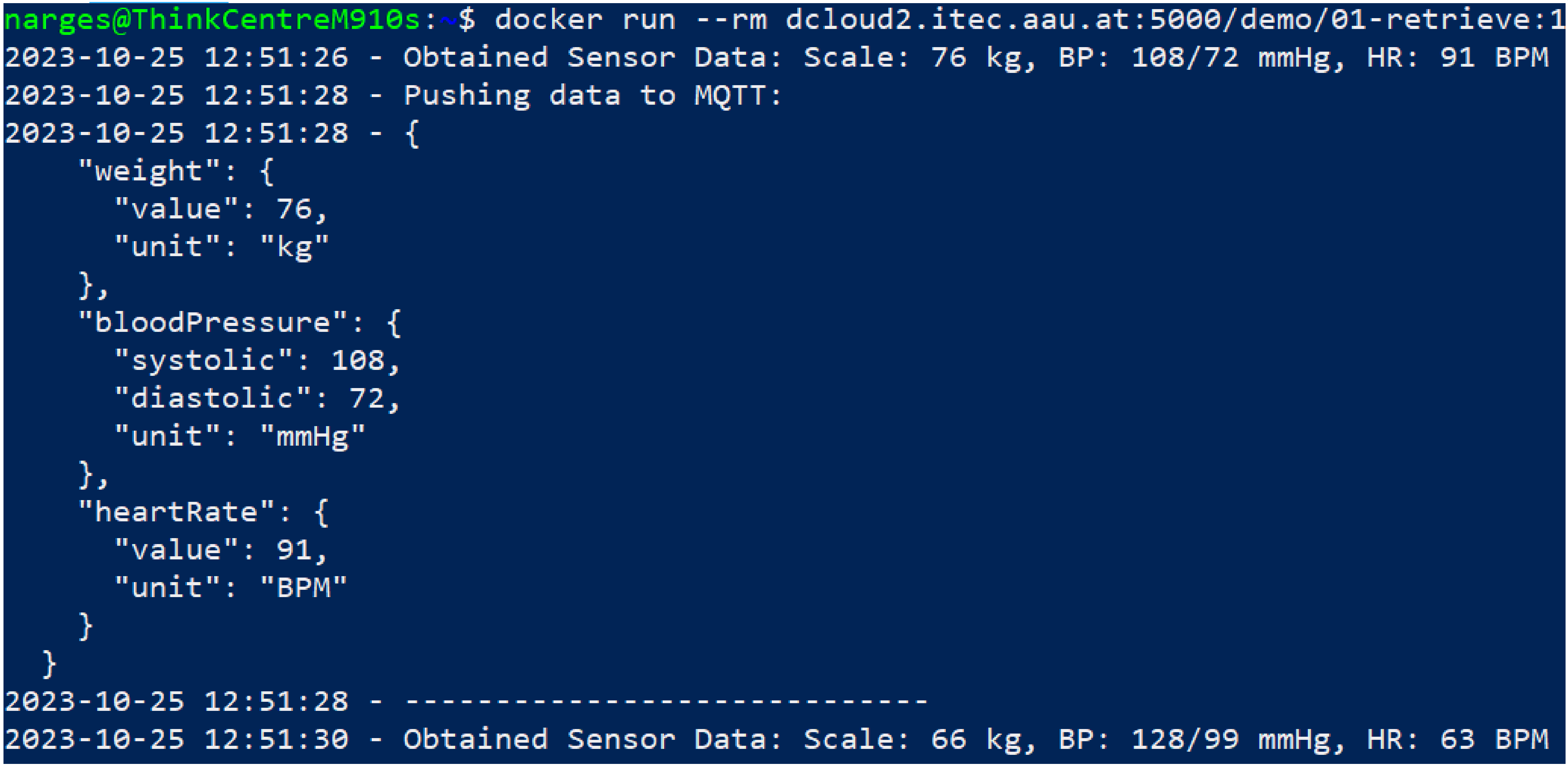


Data retrieval on the Edge

Data processing in the Cloud

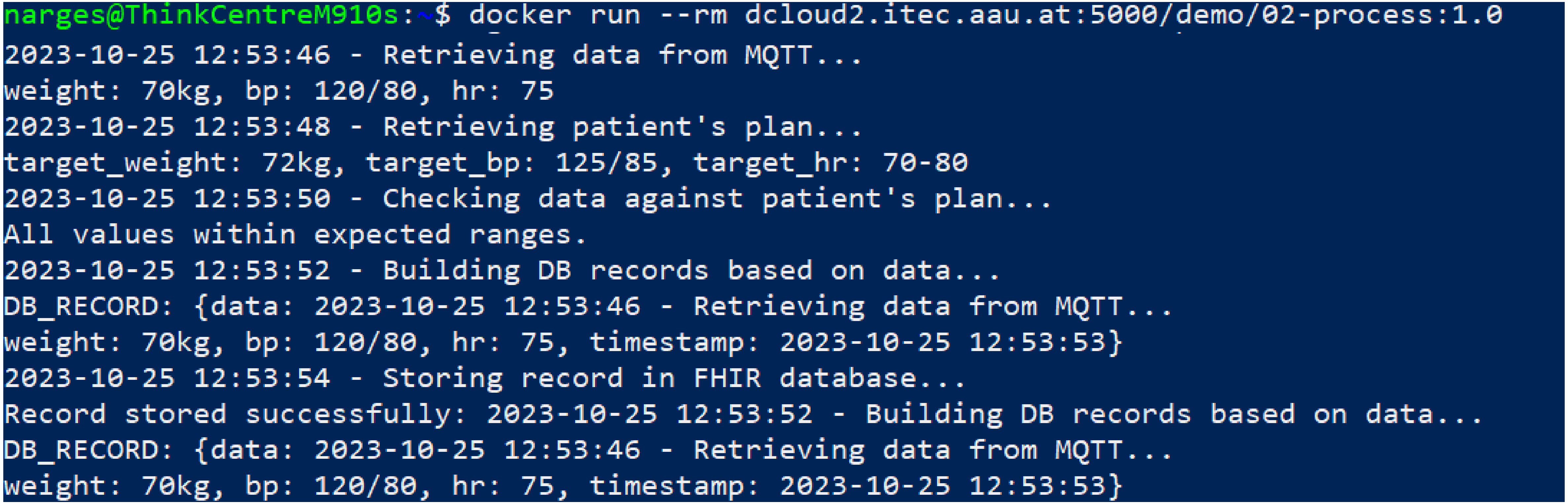
Notifying end user

Step 1: retrieve



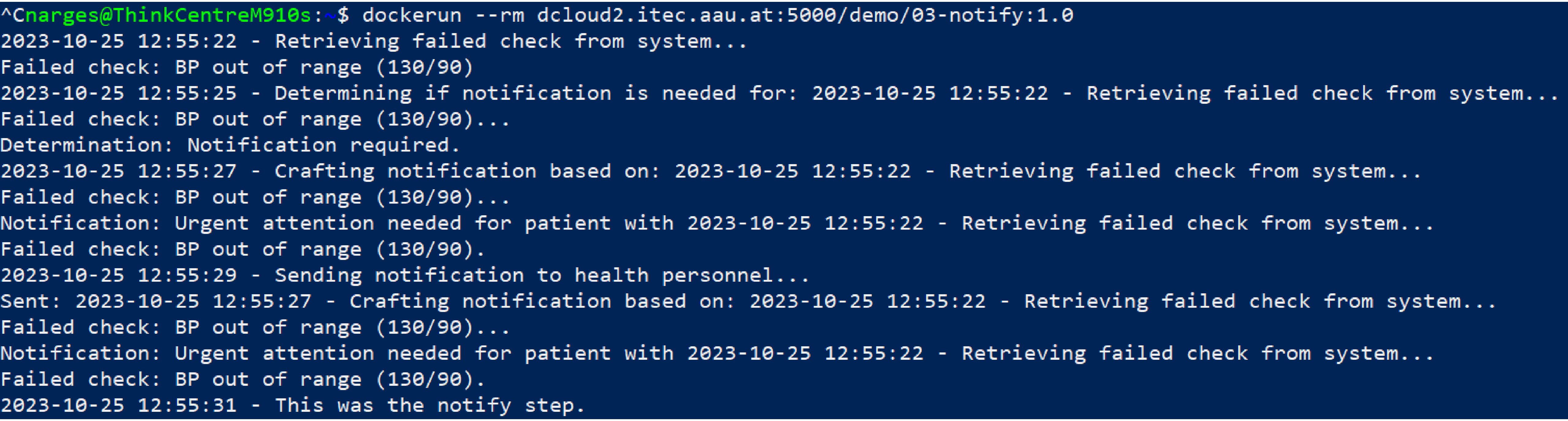
```
narges@ThinkCentreM910s:~$ docker run --rm dcloud2.itec.aau.at:5000/demo/01-retrieve:1
2023-10-25 12:51:26 - Obtained Sensor Data: Scale: 76 kg, BP: 108/72 mmHg, HR: 91 BPM
2023-10-25 12:51:28 - Pushing data to MQTT:
2023-10-25 12:51:28 - {
    "weight": {
      "value": 76,
      "unit": "kg"
    },
    "bloodPressure": {
      "systolic": 108,
      "diastolic": 72,
      "unit": "mmHg"
    },
    "heartRate": {
      "value": 91,
      "unit": "BPM"
    }
  }
2023-10-25 12:51:28 - -----------------------------
2023-10-25 12:51:30 - Obtained Sensor Data: Scale: 66 kg, BP: 128/99 mmHg, HR: 63 BPM
```

Step 2: process

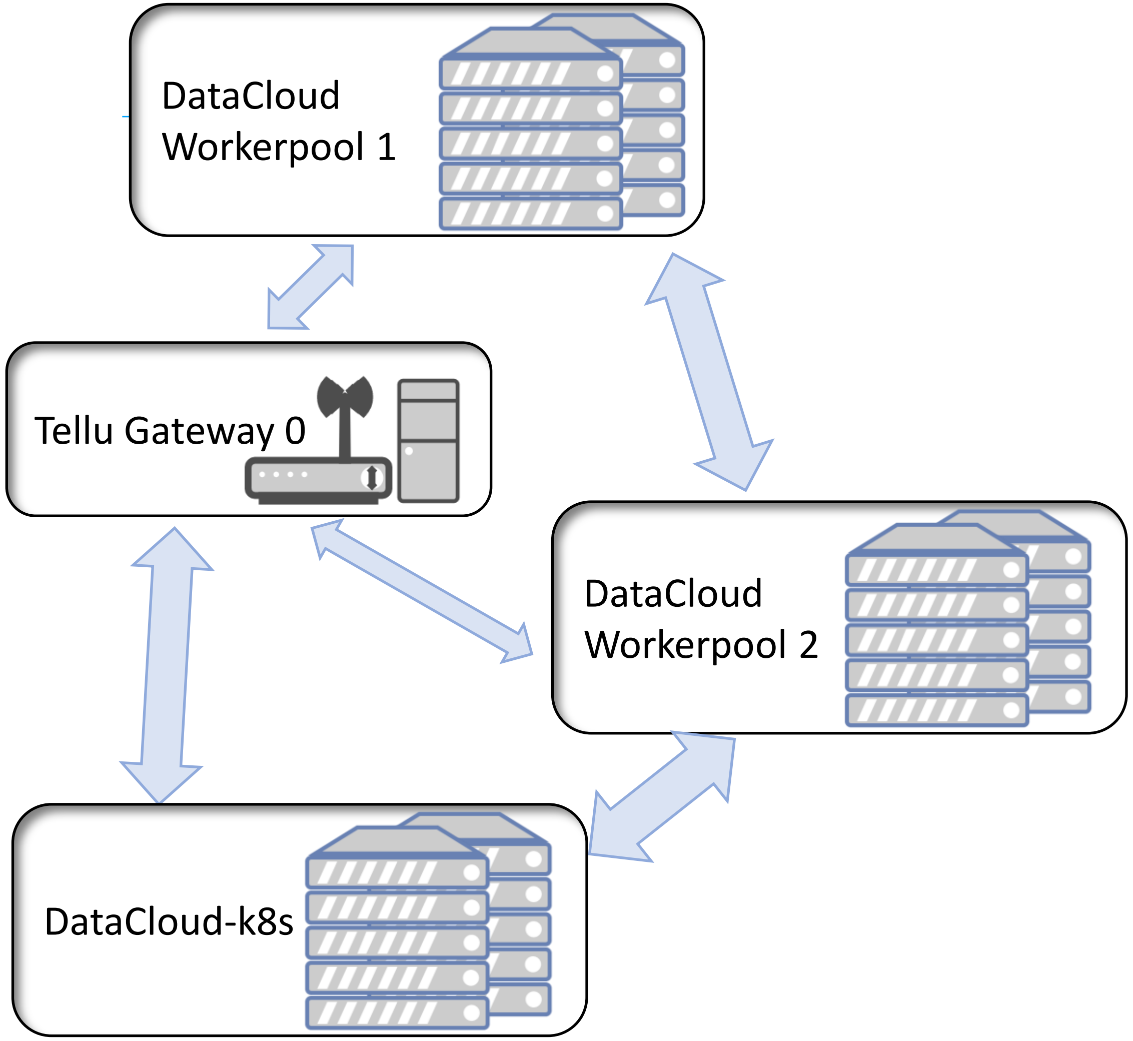


```
narges@ThinkCentreM910s:~$ docker run --rm dcloud2.itec.aau.at:5000/demo/02-process:1.0
2023-10-25 12:53:46 - Retrieving data from MQTT...
weight: 70kg, bp: 120/80, hr: 75
2023-10-25 12:53:48 - Retrieving patient's plan...
target_weight: 72kg, target_bp: 125/85, target_hr: 70-80
2023-10-25 12:53:50 - Checking data against patient's plan...
All values within expected ranges.
2023-10-25 12:53:52 - Building DB records based on data...
DB_RECORD: {data: 2023-10-25 12:53:46 - Retrieving data from MQTT...
weight: 70kg, bp: 120/80, hr: 75, timestamp: 2023-10-25 12:53:53}
2023-10-25 12:53:54 - Storing record in FHIR database...
Record stored successfully: 2023-10-25 12:53:52 - Building DB records based on data...
DB_RECORD: {data: 2023-10-25 12:53:46 - Retrieving data from MQTT...
weight: 70kg, bp: 120/80, hr: 75, timestamp: 2023-10-25 12:53:53}
```

Step 3: notify



```
^Cnarges@ThinkCentreM910s:~$ dockerun --rm dcloud2.itec.aau.at:5000/demo/03-notify:1.0
2023-10-25 12:55:22 - Retrieving failed check from system...
Failed check: BP out of range (130/90)
2023-10-25 12:55:25 - Determining if notification is needed for: 2023-10-25 12:55:22 - Retrieving failed check from system...
Failed check: BP out of range (130/90)...
Determination: Notification required.
2023-10-25 12:55:27 - Crafting notification based on: 2023-10-25 12:55:22 - Retrieving failed check from system...
Failed check: BP out of range (130/90)...
Notification: Urgent attention needed for patient with 2023-10-25 12:55:22 - Retrieving failed check from system...
Failed check: BP out of range (130/90).
2023-10-25 12:55:29 - Sending notification to health personnel...
Sent: 2023-10-25 12:55:27 - Crafting notification based on: 2023-10-25 12:55:22 - Retrieving failed check from system...
Failed check: BP out of range (130/90)...
Notification: Urgent attention needed for patient with 2023-10-25 12:55:22 - Retrieving failed check from system...
Failed check: BP out of range (130/90).
2023-10-25 12:55:31 - This was the notify step.
```

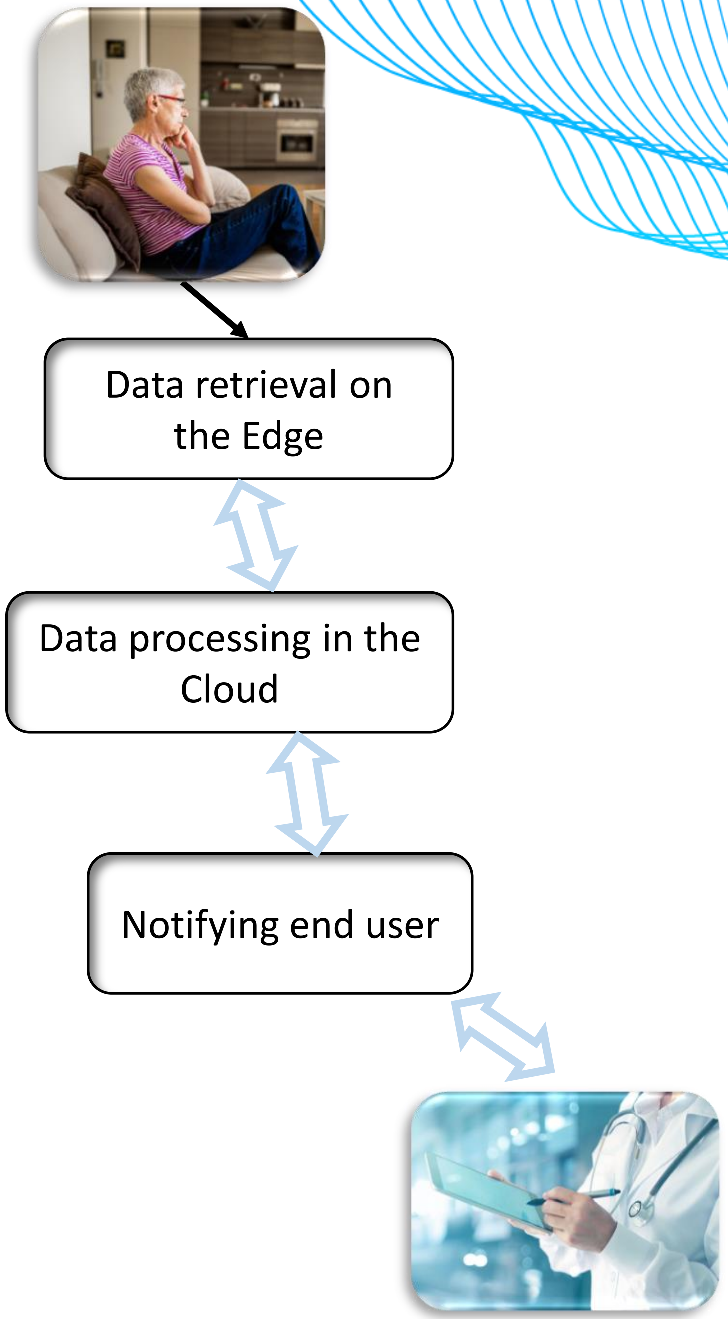


DataCloud Workerpool 1

Tellu Gateway 0

DataCloud Workerpool 2

DataCloud-k8s

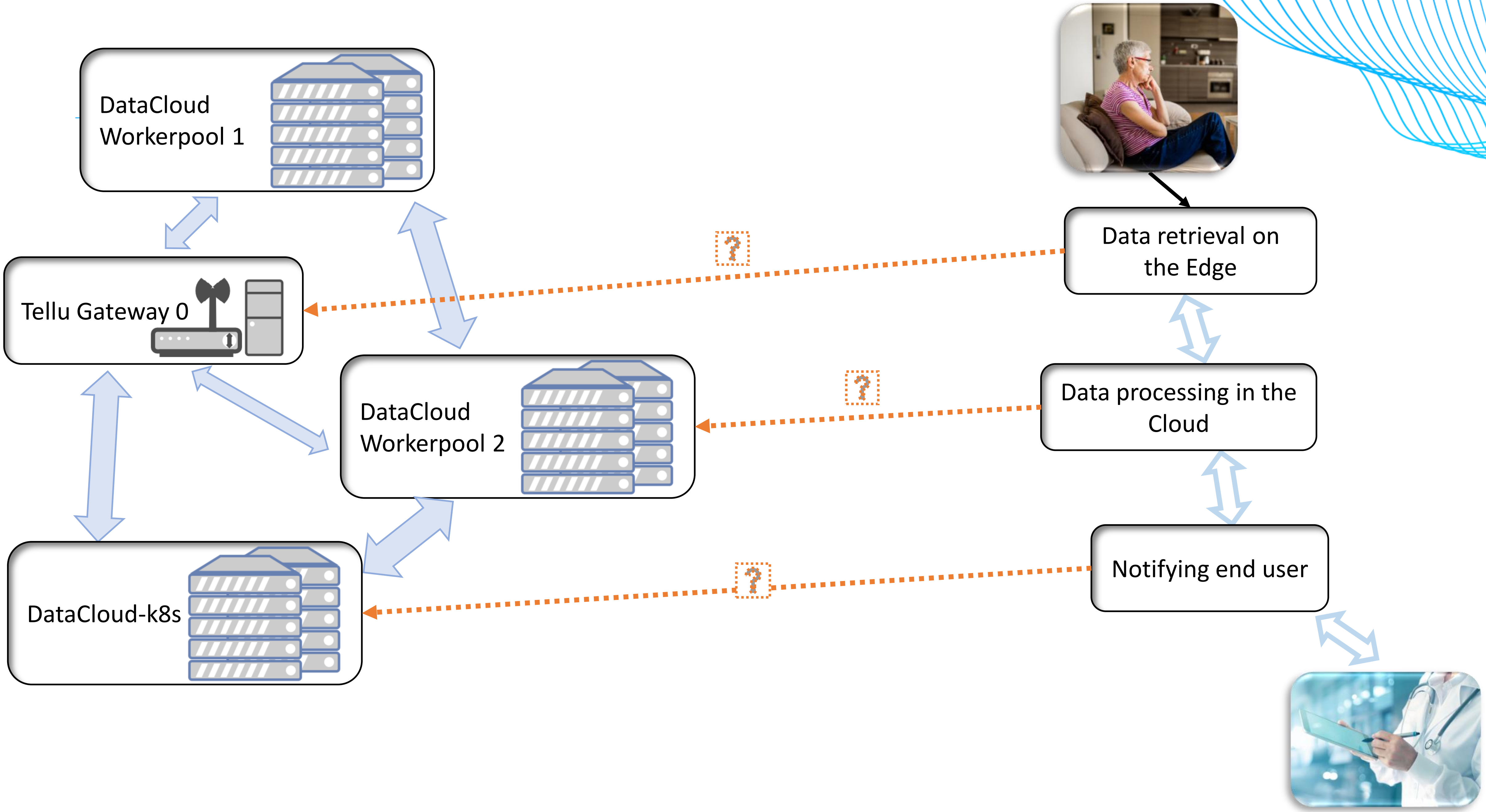


Data retrieval on the Edge

Data processing in the Cloud

Notifying end user

ADA-PIPE orchestration and scheduler



DataCloud Workerpool 1

Tellu Gateway 0

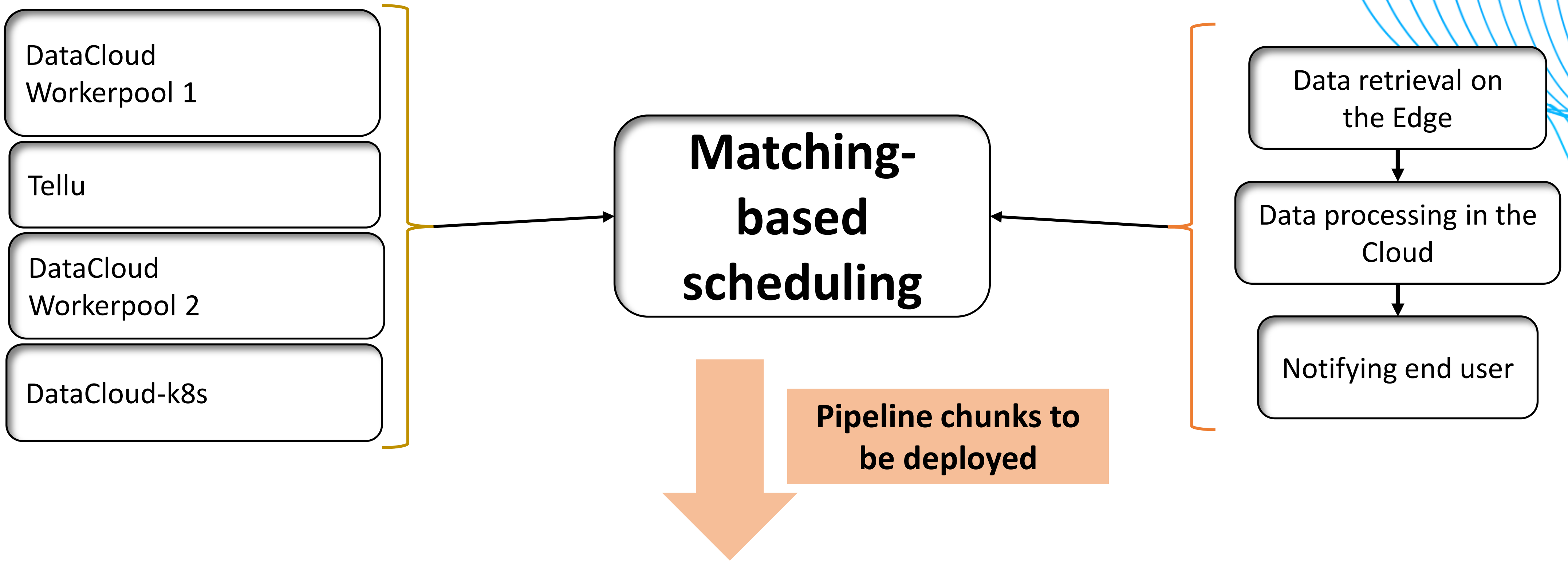
DataCloud Workerpool 2

DataCloud-k8s

Data retrieval on the Edge

Data processing in the Cloud

Notifying end user



DataCloud Workerpool 1

Tellu

DataCloud Workerpool 2

DataCloud-k8s

Matching-based scheduling

Data retrieval on the Edge

Data processing in the Cloud

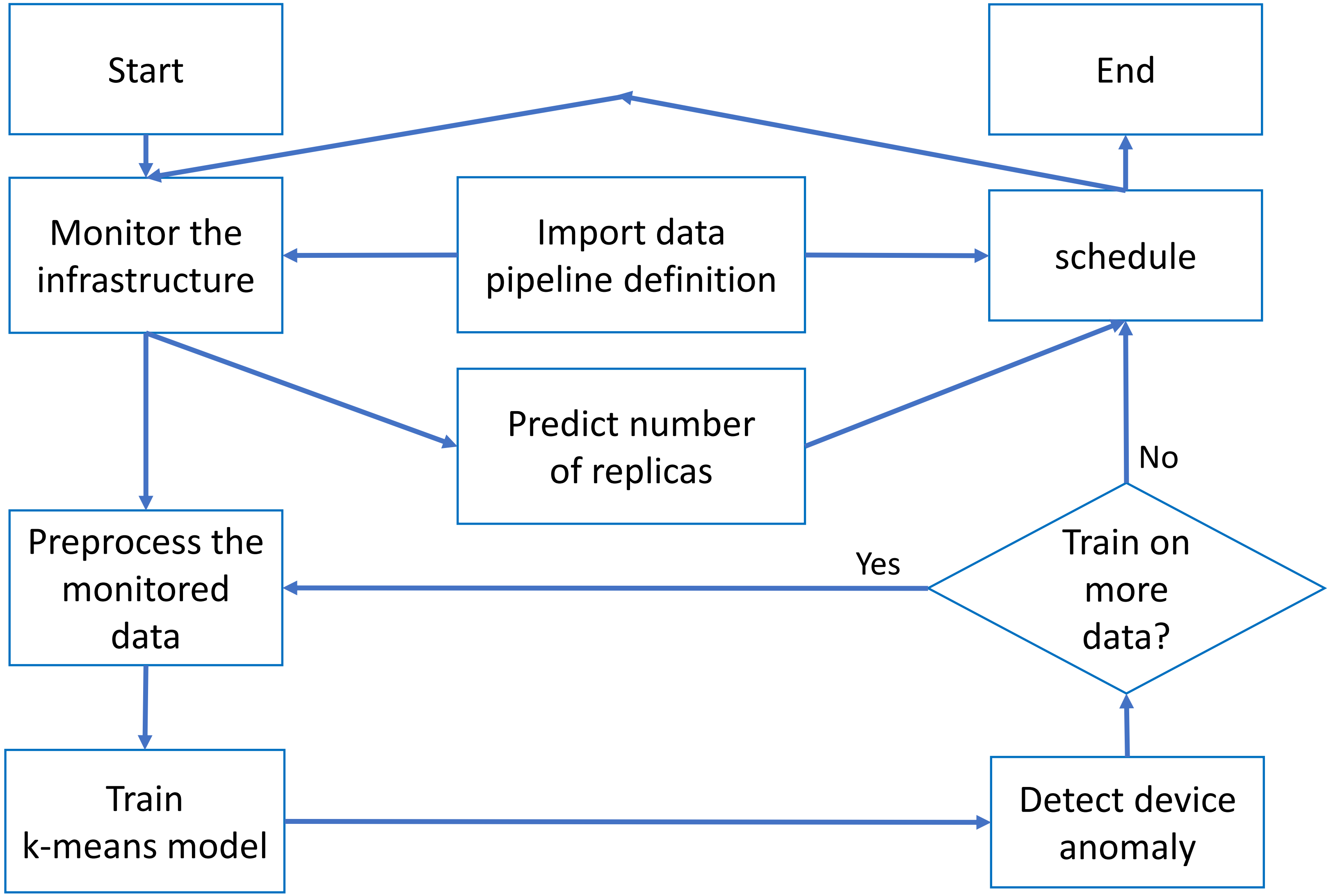
Notifying end user

Pipeline chunks to be deployed

Step	Provider	Node/Device
01 - retrieve	DataCloud	DataCloud-Edge-0
02 - process	DataCloud	Datacloud-wp1-test1
03 - notify	DataCloud	Datacloud-wp1-test2

DEP-PIPE

ADA-PIPE adaptation and scheduling architecture


Start

Monitor the infrastructure

Import data pipeline definition

schedule

End

Predict number of replicas

Preprocess the monitored data

Train on more data?

Yes

No

Train k-means model

Detect device anomaly

Scaling the steps (µ-services) of a data pipeline

- Horizontal scaling (horizontal pod autoscaler - HPA),
  - ✓ deploying more steps as a response to the increased load;
- Vertical scaling (vertical pod autoscaler - VPA),
  - ✓ assigning more resources to the pipelines that are already running for the workload.

Discussion on other methods

- *Autopilot:*
  - ➢ number of replicas from each, and
    - averaging window for the CPU usage (the default is 5 minutes);
    - target average utilization $r*$;
    - length T (the default length is 72 hours);
    - statistics S: max or P_{95} (95%ile).

$$r_S[t] = S_{\tau\in[t-T,t]}\{\textstyle\sum_i r_i[\tau]\} \longrightarrow n_r[t] = r_S[t]/r^*$$

Discussion on other methods (cont.)

- *Autopilot:*
  - number of replicas from each microservice, and resource limits for each microservice (CPU/memory limits for individual microservice - vertical scaling).

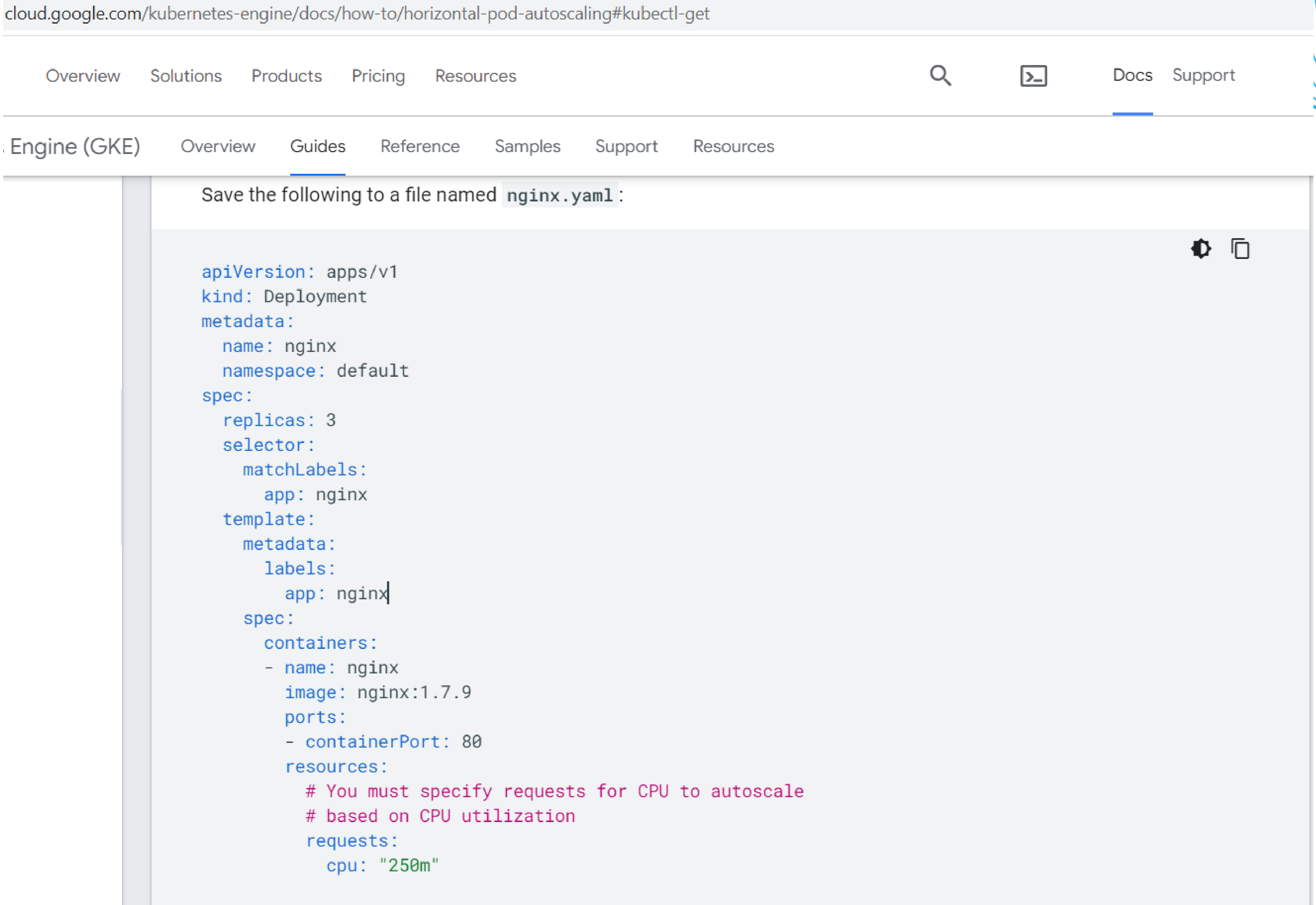
- *Kubernetes:*
  - $\text{desiredReplicas} = \left\lceil \text{currentReplicas} \cdot \frac{\text{currentMetricValue}}{\text{desiredMetricValue}} \right\rceil$
  - vertical pod `autoscaler` sets containers' limits using statistics over a moving window (e.g., for RAM, the 99th percentile over 24h).

Configuring horizontal pod autoscaling

- Deploy a step under the name of `nginx`

- Specify the following values:
  - ✓ Minimum number of replicas: 1
  - ✓ Maximum number of replicas: 10
  - ✓ Autoscaling metric: CPU
  - ✓ Target: 50
  - ✓ Unit: %

- Policy > Autoscale.

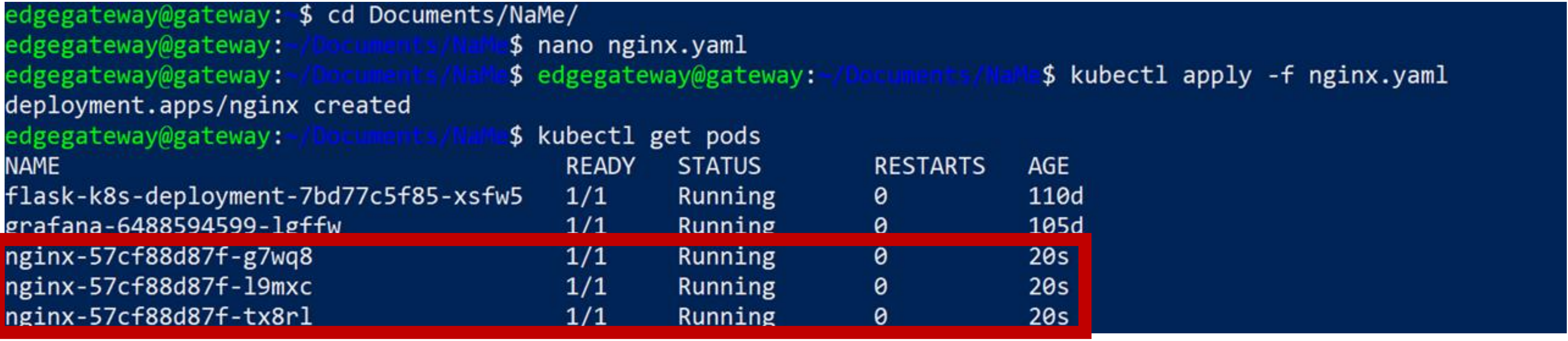


Save the following to a file named `nginx.yaml`:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
  namespace: default
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.7.9
        ports:
        - containerPort: 80
        resources:
          # You must specify requests for CPU to autoscale
          # based on CPU utilization
          requests:
            cpu: "250m"
```

https://cloud.google.com/kubernetes-engine/docs/how-to/horizontal-pod-autoscaling#kubectl-apply

Creating the workload deployment

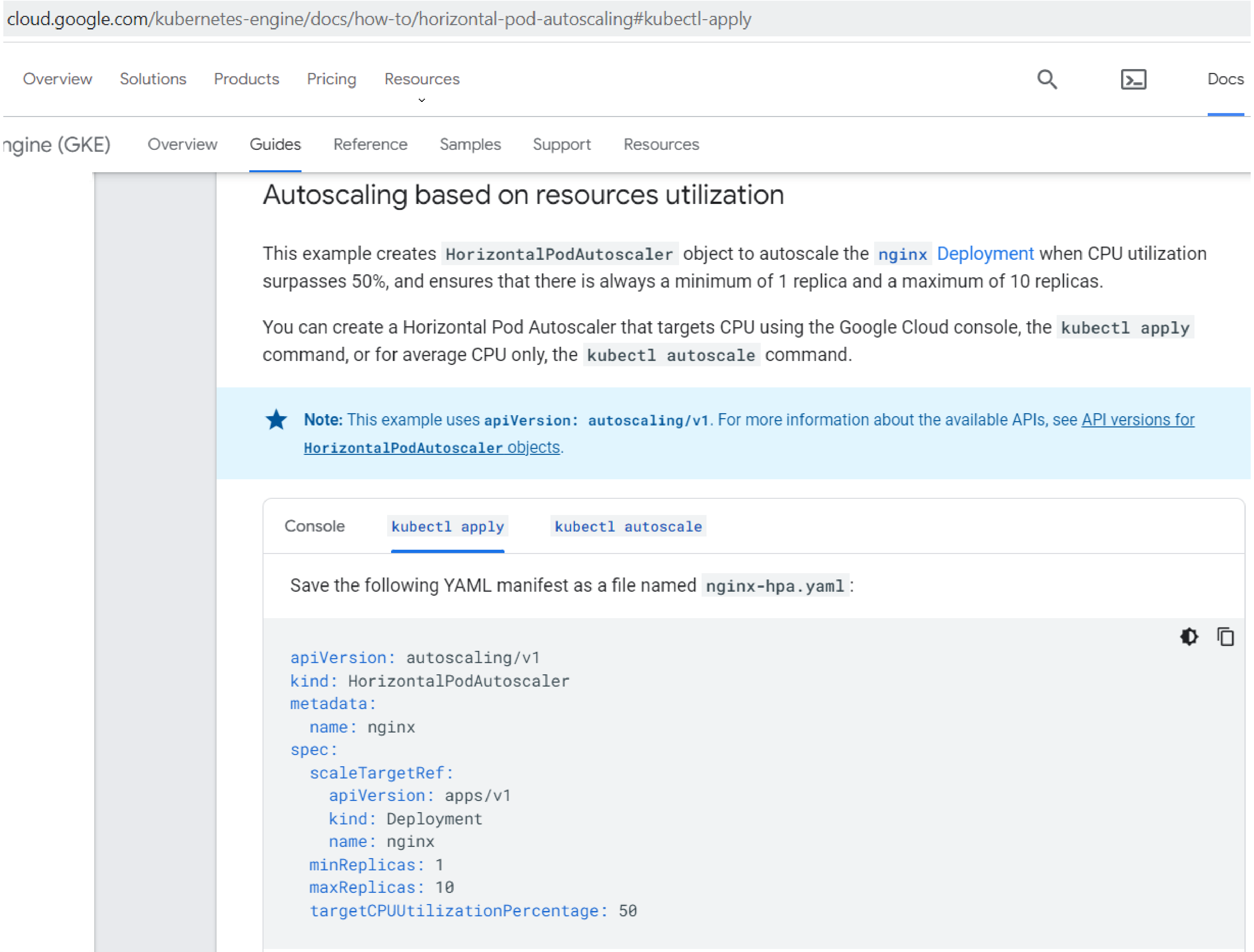


```
edgegateway@gateway:~$ cd Documents/NaMe/
edgegateway@gateway:~/Documents/NaMe$ nano nginx.yaml
edgegateway@gateway:~/Documents/NaMe$ edgegateway@gateway:~/Documents/NaMe$ kubectl apply -f nginx.yaml
deployment.apps/nginx created
edgegateway@gateway:~/Documents/NaMe$ kubectl get pods
NAME                                READY   STATUS      RESTARTS   AGE
flask-k8s-deployment-7bd77c5f85-xsfw5   1/1     Running     0          110d
grafana-6488594599-lgffw            1/1     Running     0          105d
nginx-57cf88d87f-g7wq8              1/1     Running     0          20s
nginx-57cf88d87f-l9mxc              1/1     Running     0          20s
nginx-57cf88d87f-tx8rl              1/1     Running     0          20s
```

https://cloud.google.com/kubernetes-engine/docs/how-to/horizontal-pod-autoscaling

Autoscaling based on resources utilization

- This example creates `HorizontalPodAutoscaler` object
  - ✓ to autoscale the `nginx` *deployment*
    - ➢ when CPU utilization surpasses 50%, and
    - ➢ ensures that there is always
      - ❖ a minimum of 1 replica and
      - ❖ a maximum of 10 replicas.



Autoscaling based on resources utilization

This example creates `HorizontalPodAutoscaler` object to autoscale the `nginx` Deployment when CPU utilization surpasses 50%, and ensures that there is always a minimum of 1 replica and a maximum of 10 replicas.

You can create a Horizontal Pod Autoscaler that targets CPU using the Google Cloud console, the `kubectl apply` command, or for average CPU only, the `kubectl autoscale` command.

★ **Note:** This example uses **`apiVersion: autoscaling/v1`**. For more information about the available APIs, see API versions for `HorizontalPodAutoscaler` objects.

Save the following YAML manifest as a file named `nginx-hpa.yaml`:

```
apiVersion: autoscaling/v1
kind: HorizontalPodAutoscaler
metadata:
  name: nginx
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: nginx
  minReplicas: 1
  maxReplicas: 10
  targetCPUUtilizationPercentage: 50
```

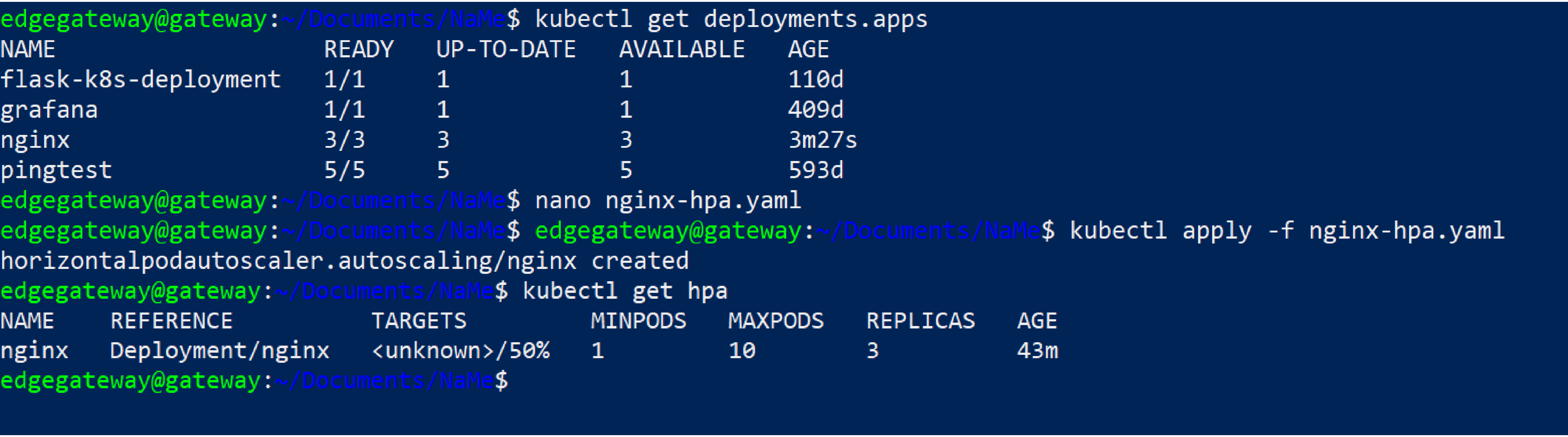
https://cloud.google.com/kubernetes-engine/docs/how-to/horizontal-pod-autoscaling#kubectl-apply

HPA YAML manifest as `nginx-hpa.yaml`

```
apiVersion: autoscaling/v1
kind: HorizontalPodAutoscaler
metadata:
  name: nginx
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: nginx
  minReplicas: 1
  maxReplicas: 10
  targetCPUUtilizationPercentage: 50
```

} If the mean of the apps CPU utilization is higher than this target, the replicas will be run.

Autoscaling based on resources utilization (cont.)

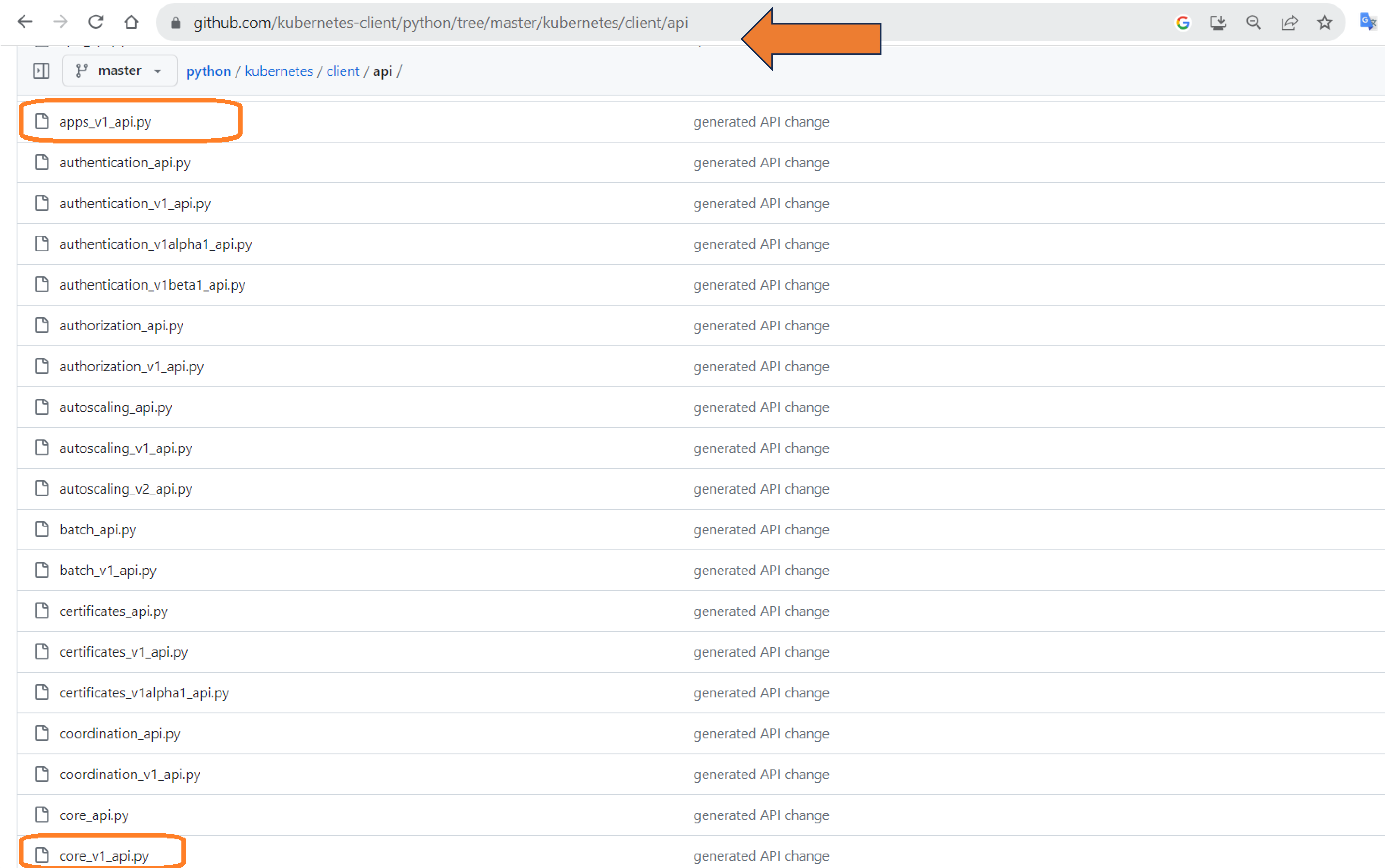


```
edgegateway@gateway:~/Documents/NaMe$ kubectl get deployments.apps
NAME                    READY   UP-TO-DATE   AVAILABLE   AGE
flask-k8s-deployment    1/1     1            1           110d
grafana                 1/1     1            1           409d
nginx                   3/3     3            3           3m27s
pingtest                5/5     5            5           593d
edgegateway@gateway:~/Documents/NaMe$ nano nginx-hpa.yaml
edgegateway@gateway:~/Documents/NaMe$ edgegateway@gateway:~/Documents/NaMe$ kubectl apply -f nginx-hpa.yaml
horizontalpodautoscaler.autoscaling/nginx created
edgegateway@gateway:~/Documents/NaMe$ kubectl get hpa
NAME    REFERENCE          TARGETS         MINPODS   MAXPODS   REPLICAS   AGE
nginx   Deployment/nginx   <unknown>/50%   1         10        3          43m
edgegateway@gateway:~/Documents/NaMe$
```

https://github.com/SiNa88/HPA

Update deployments

- Utilizing the following API:
  - https://github.com/kubernetes-client/python/tree/master/kubernetes/client
  - https://www.youtube.com/watch?v=XJOaaGSLS3U
- The source code https://github.com/SiNa88/HPA/blob/main/updateDeployment.py
  - reads all the current deployments,
  - considers default values for resource requests and limits of each step,
  - assigns the new numbers of replicas and new resources to the microservices.
  - updates the deployment through `kube-scheduler`,



github.com/kubernetes-client/python/tree/master/kubernetes/client/api

master

python / kubernetes / client / api /

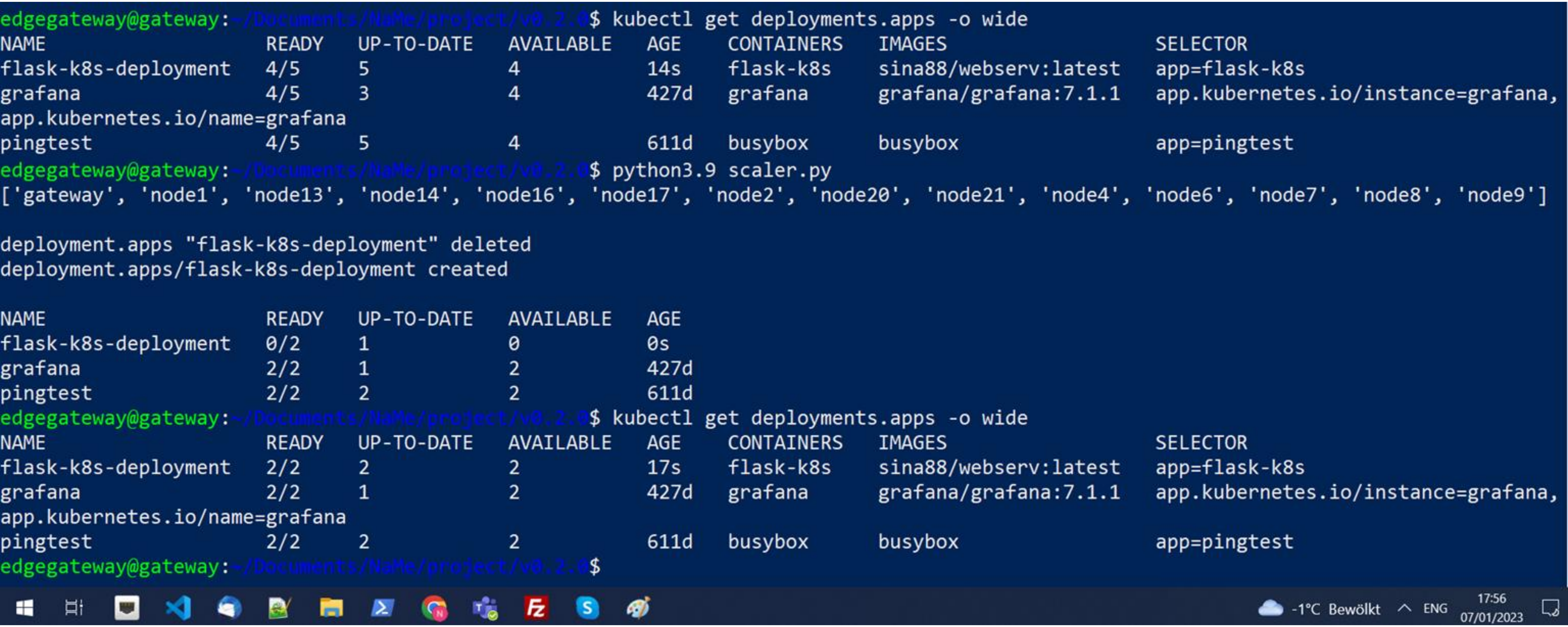
apps_v1_api.py	generated API change
authentication_api.py	generated API change
authentication_v1_api.py	generated API change
authentication_v1alpha1_api.py	generated API change
authentication_v1beta1_api.py	generated API change
authorization_api.py	generated API change
authorization_v1_api.py	generated API change
autoscaling_api.py	generated API change
autoscaling_v1_api.py	generated API change
autoscaling_v2_api.py	generated API change
batch_api.py	generated API change
batch_v1_api.py	generated API change
certificates_api.py	generated API change
certificates_v1_api.py	generated API change
certificates_v1alpha1_api.py	generated API change
coordination_api.py	generated API change
coordination_v1_api.py	generated API change
core_api.py	generated API change
core_v1_api.py	generated API change

1) Watch an API resource
2) Stream the result back via a generator

```
def main():
    print(nodes_available())
    call(["minikube", "kubectl", "--", "get", "-o", "wide", "deployments.apps"])

    w = watch.Watch()
    for event in w.stream(v1.list_namespaced_pod, "default"):
        if event['object'].status.phase == "Pending": and event['object'].spec.scheduler_name == scheduler_name:
            try:
                print("------------------------------------------")
                print("scheduling pod ", event['object'].metadata.name)
                res = scheduler(event['object'], (node_available()))
                break
            except client.rest.ApiException as e:
                print (json.loads(e.body)['message'])
    print()
    scale()
    print()
```

- The following code updates the application’s deployment through the kube-scheduler (running the code on an real cluster)https://github.com/SiNa88/HPA/blob/main/updateDeployment.py

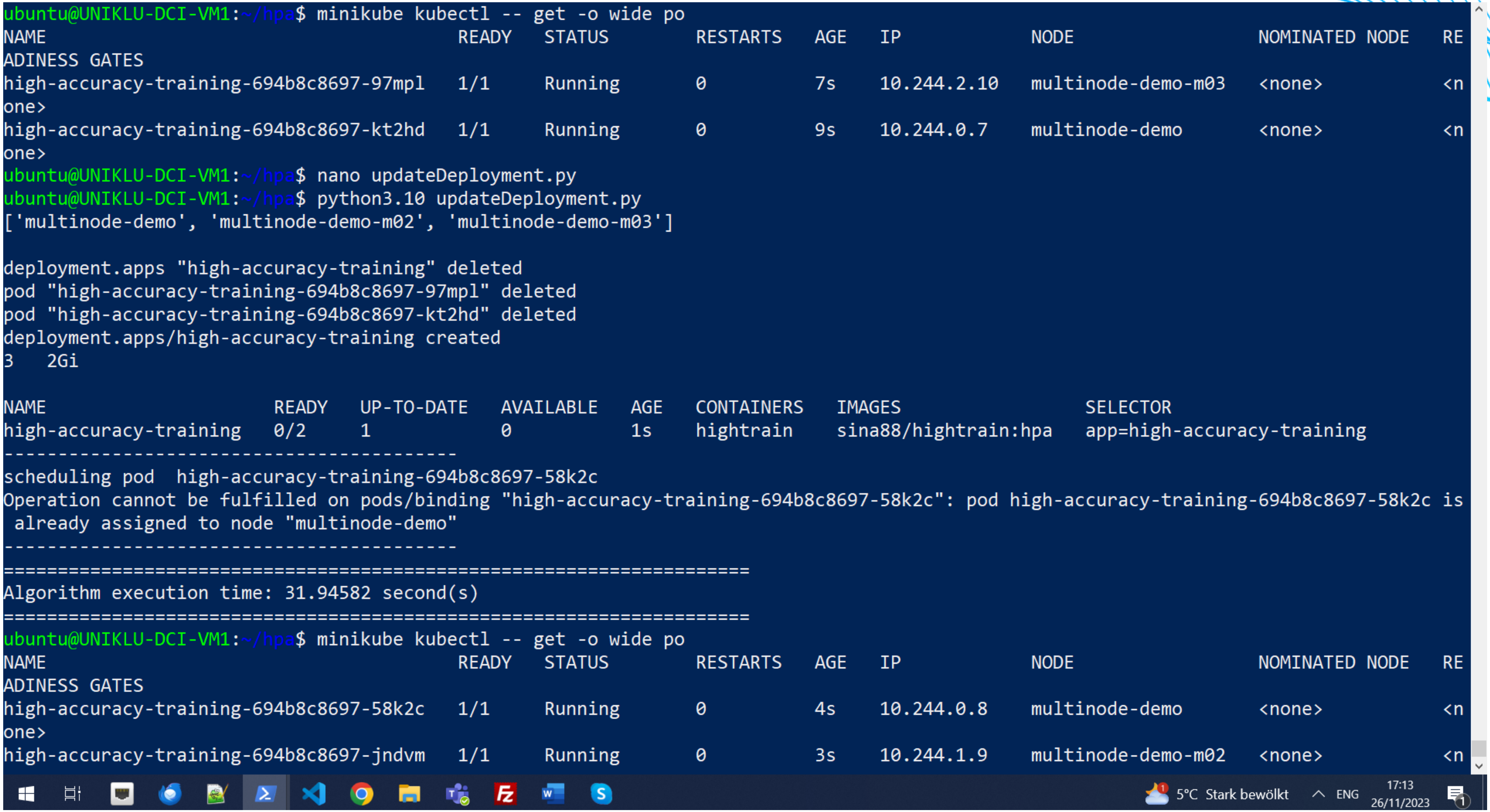


```
edgegateway@gateway:~/Documents/NaMe/project/v0.2.0$ kubectl get deployments.apps -o wide
NAME                  READY   UP-TO-DATE   AVAILABLE   AGE    CONTAINERS   IMAGES                  SELECTOR
flask-k8s-deployment   4/5     5            4           14s    flask-k8s    sina88/webserv:latest   app=flask-k8s
grafana               4/5     3            4           427d   grafana      grafana/grafana:7.1.1   app.kubernetes.io/instance=grafana,
app.kubernetes.io/name=grafana
pingtest              4/5     5            4           611d   busybox      busybox                 app=pingtest
edgegateway@gateway:~/Documents/NaMe/project/v0.2.0$ python3.9 scaler.py
['gateway', 'node1', 'node13', 'node14', 'node16', 'node17', 'node2', 'node20', 'node21', 'node4', 'node6', 'node7', 'node8', 'node9']

deployment.apps "flask-k8s-deployment" deleted
deployment.apps/flask-k8s-deployment created

NAME                  READY   UP-TO-DATE   AVAILABLE   AGE
flask-k8s-deployment   0/2     1            0           0s
grafana               2/2     1            2           427d
pingtest              2/2     2            2           611d
edgegateway@gateway:~/Documents/NaMe/project/v0.2.0$ kubectl get deployments.apps -o wide
NAME                  READY   UP-TO-DATE   AVAILABLE   AGE    CONTAINERS   IMAGES                  SELECTOR
flask-k8s-deployment   2/2     2            2           17s    flask-k8s    sina88/webserv:latest   app=flask-k8s
grafana               2/2     1            2           427d   grafana      grafana/grafana:7.1.1   app.kubernetes.io/instance=grafana,
app.kubernetes.io/name=grafana
pingtest              2/2     2            2           611d   busybox      busybox                 app=pingtest
edgegateway@gateway:~/Documents/NaMe/project/v0.2.0$
```

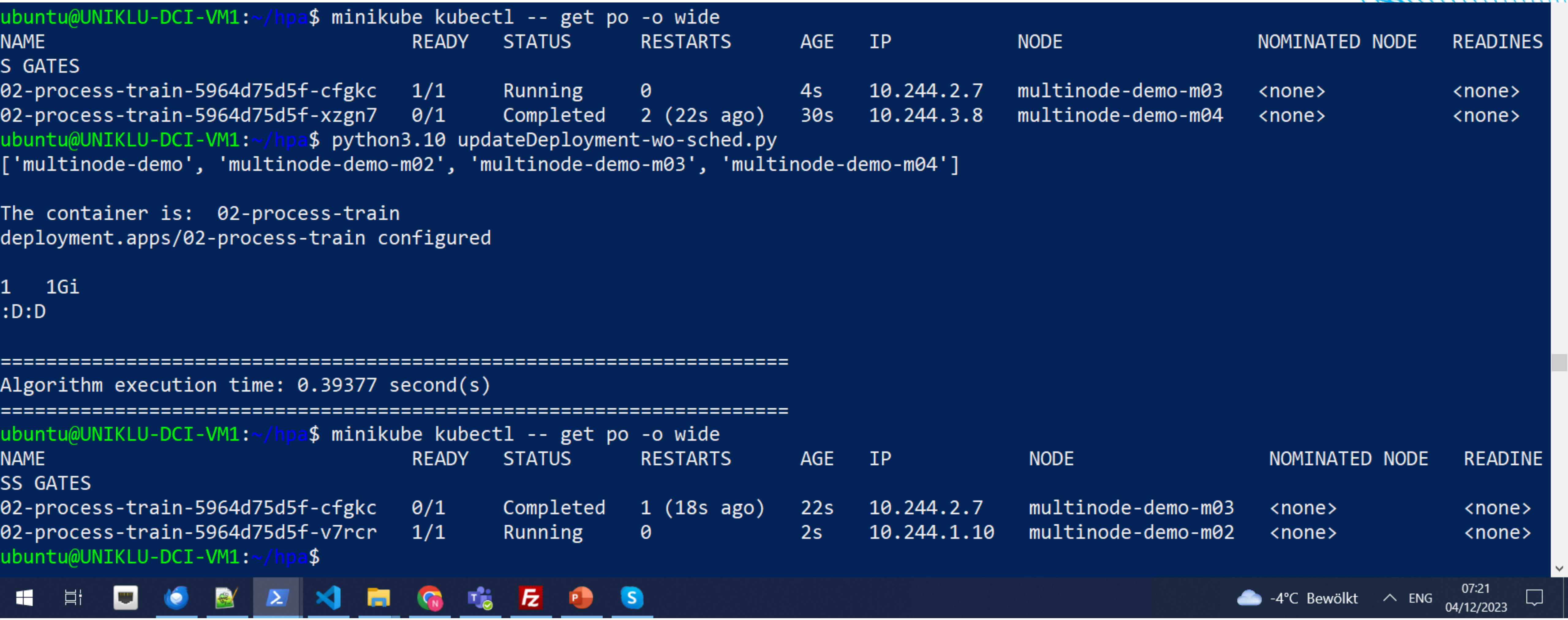
https://github.com/DataCloud-project/ADA-PIPE/blob/main/update-deployment/updateDeployment.py



```
ubuntu@UNIKLU-DCI-VM1:~/hpa$ minikube kubectl -- get -o wide po
NAME                                  READY   STATUS      RESTARTS   AGE   IP            NODE                 NOMINATED NODE    RE
ADINESS GATES
high-accuracy-training-694b8c8697-97mpl   1/1     Running     0          7s    10.244.2.10   multinode-demo-m03   <none>            <n
one>
high-accuracy-training-694b8c8697-kt2hd   1/1     Running     0          9s    10.244.0.7    multinode-demo       <none>            <n
one>
ubuntu@UNIKLU-DCI-VM1:~/hpa$ nano updateDeployment.py
ubuntu@UNIKLU-DCI-VM1:~/hpa$ python3.10 updateDeployment.py
['multinode-demo', 'multinode-demo-m02', 'multinode-demo-m03']

deployment.apps "high-accuracy-training" deleted
pod "high-accuracy-training-694b8c8697-97mpl" deleted
pod "high-accuracy-training-694b8c8697-kt2hd" deleted
deployment.apps/high-accuracy-training created
3   2Gi

NAME                     READY   UP-TO-DATE   AVAILABLE   AGE   CONTAINERS   IMAGES                 SELECTOR
high-accuracy-training   0/2     1            0           1s    hightrain    sina88/hightrain:hpa   app=high-accuracy-training
------------------------------------------
scheduling pod  high-accuracy-training-694b8c8697-58k2c
Operation cannot be fulfilled on pods/binding "high-accuracy-training-694b8c8697-58k2c": pod high-accuracy-training-694b8c8697-58k2c is
 already assigned to node "multinode-demo"
------------------------------------------
=====================================================================
Algorithm execution time: 31.94582 second(s)
=====================================================================
ubuntu@UNIKLU-DCI-VM1:~/hpa$ minikube kubectl -- get -o wide po
NAME                                  READY   STATUS      RESTARTS   AGE   IP            NODE                 NOMINATED NODE    RE
ADINESS GATES
high-accuracy-training-694b8c8697-58k2c   1/1     Running     0          4s    10.244.0.8    multinode-demo       <none>            <n
one>
high-accuracy-training-694b8c8697-jndvm   1/1     Running     0          3s    10.244.1.9    multinode-demo-m02   <none>            <n
```

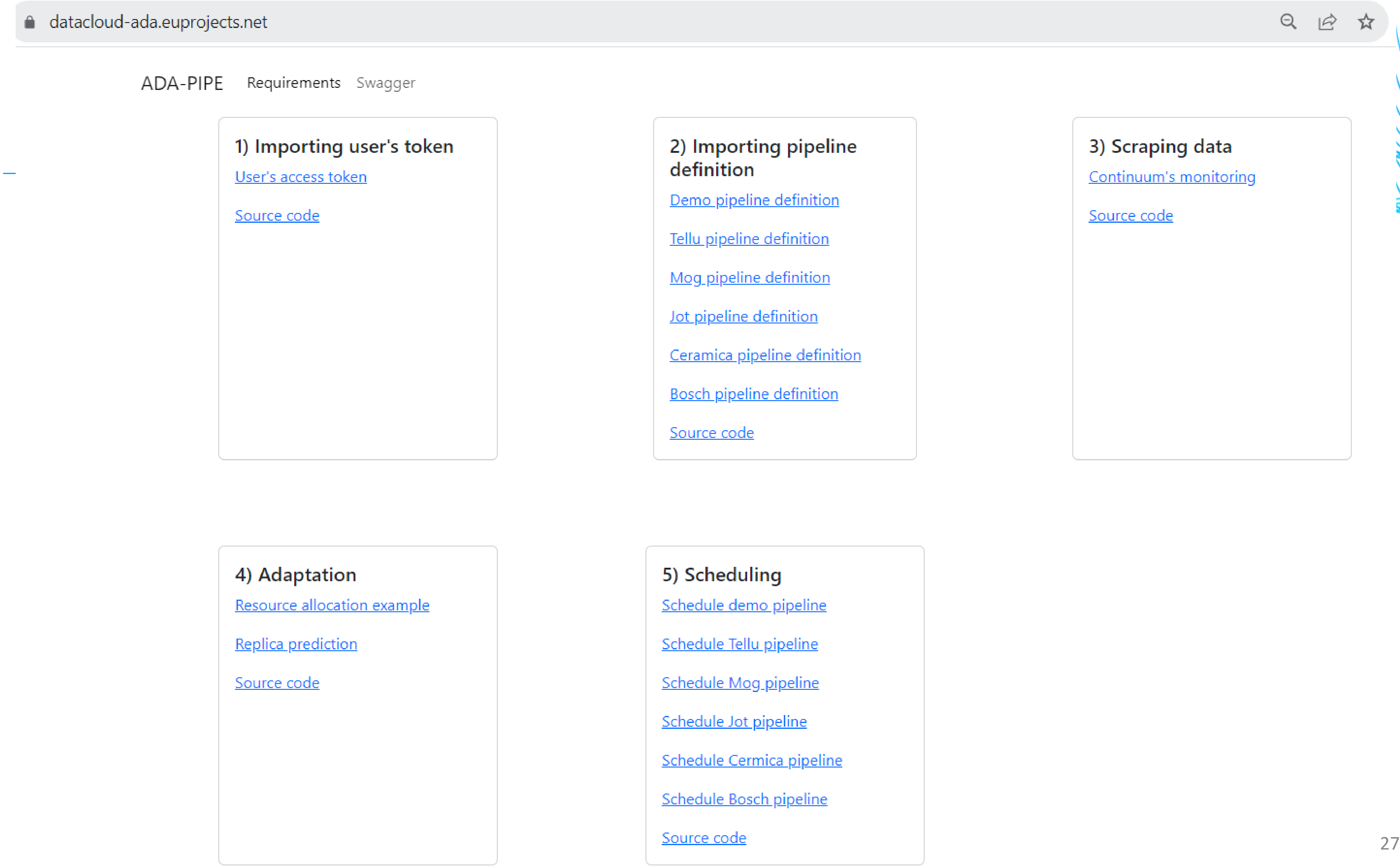


```
ubuntu@UNIKLU-DCI-VM1:~/hpa$ minikube kubectl -- get po -o wide
NAME                                 READY   STATUS      RESTARTS       AGE   IP           NODE                NOMINATED NODE   READINES
S GATES
02-process-train-5964d75d5f-cfgkc   1/1     Running     0              4s    10.244.2.7   multinode-demo-m03   <none>           <none>
02-process-train-5964d75d5f-xzgn7   0/1     Completed   2 (22s ago)    30s   10.244.3.8   multinode-demo-m04   <none>           <none>
ubuntu@UNIKLU-DCI-VM1:~/hpa$ python3.10 updateDeployment-wo-sched.py
['multinode-demo', 'multinode-demo-m02', 'multinode-demo-m03', 'multinode-demo-m04']

The container is:  02-process-train
deployment.apps/02-process-train configured

1   1Gi
:D:D

=====================================================================
Algorithm execution time: 0.39377 second(s)
=====================================================================
ubuntu@UNIKLU-DCI-VM1:~/hpa$ minikube kubectl -- get po -o wide
NAME                                 READY   STATUS      RESTARTS       AGE   IP            NODE                 NOMINATED NODE   READINE
SS GATES
02-process-train-5964d75d5f-cfgkc   0/1     Completed   1 (18s ago)    22s   10.244.2.7    multinode-demo-m03   <none>           <none>
02-process-train-5964d75d5f-v7rcr   1/1     Running     0              2s    10.244.1.10   multinode-demo-m02   <none>           <none>
ubuntu@UNIKLU-DCI-VM1:~/hpa$
```



datacloud-ada.euprojects.net

ADA-PIPE Requirements Swagger

1) Importing user's token

User's access token

Source code

2) Importing pipeline definition

Demo pipeline definition

Tellu pipeline definition

Mog pipeline definition

Jot pipeline definition

Ceramica pipeline definition

Bosch pipeline definition

Source code

3) Scraping data

Continuum's monitoring

Source code

4) Adaptation

Resource allocation example

Replica prediction

Source code

5) Scheduling

Schedule demo pipeline

Schedule Tellu pipeline

Schedule Mog pipeline

Schedule Jot pipeline

Schedule Cermica pipeline

Schedule Bosch pipeline

Source code

Installing docker engine and minikube

- https://docs.docker.com/engine/install/ubuntu/
- https://minikube.sigs.k8s.io/docs/start/
- https://minikube.sigs.k8s.io/docs/tutorials/multi_node/
- https://github.com/kubernetes-client/python/tree/master/kubernetes/client/api

References

- https://www.youtube.com/watch?v=DhojZ10Ue6w
- https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale-walkthrough/
- https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands#autoscale
- https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale/
- https://www.kubecost.com/kubernetes-autoscaling
- https://github.com/draios/kubernetes-scheduler

THANK YOU!

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101016835, the DataCloud Project.

https://datacloudproject.eu/