**Lab 2   
  
LABELS & SELECTORS**

1. **EXAMPLE OF LABELS**

kind: Pod

apiVersion: v1

metadata:

name: labelpod

labels:

env: development

class: pods

spec:

containers:

- name: c00

image: ubuntu

command: ["/bin/bash", "-c", "while true; do echo Hello-Mr-Qasim; sleep 5 ; done"]

* kubectl apply -f labels.yml /to create pods

A screenshot of a computer

Description automatically generated

* kubectl get pods / to get info how many pods are there or pod is created

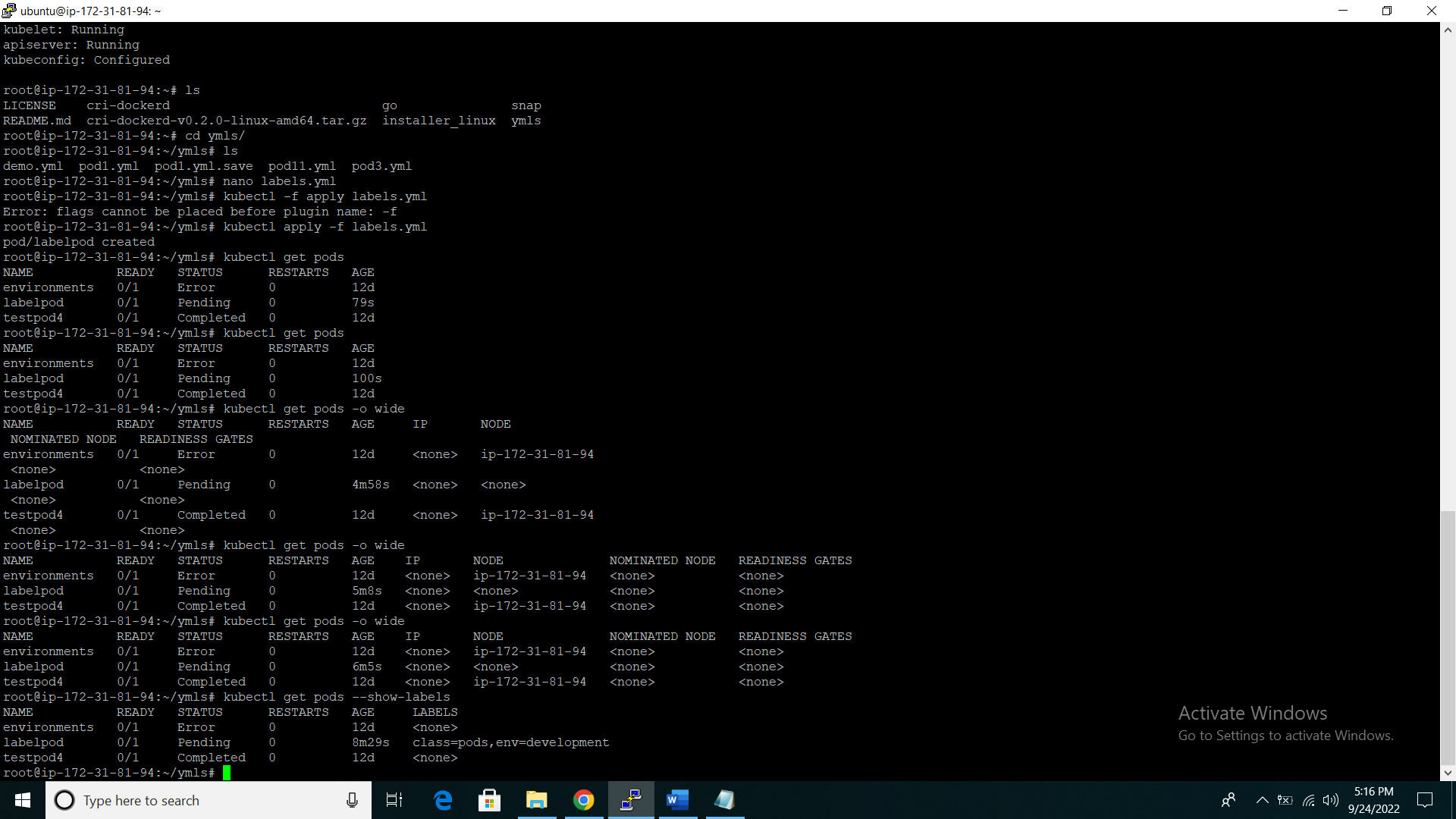
A screenshot of a computer

Description automatically generated

* kubectl get pods -o wide / details about pods like pod ip and server ip

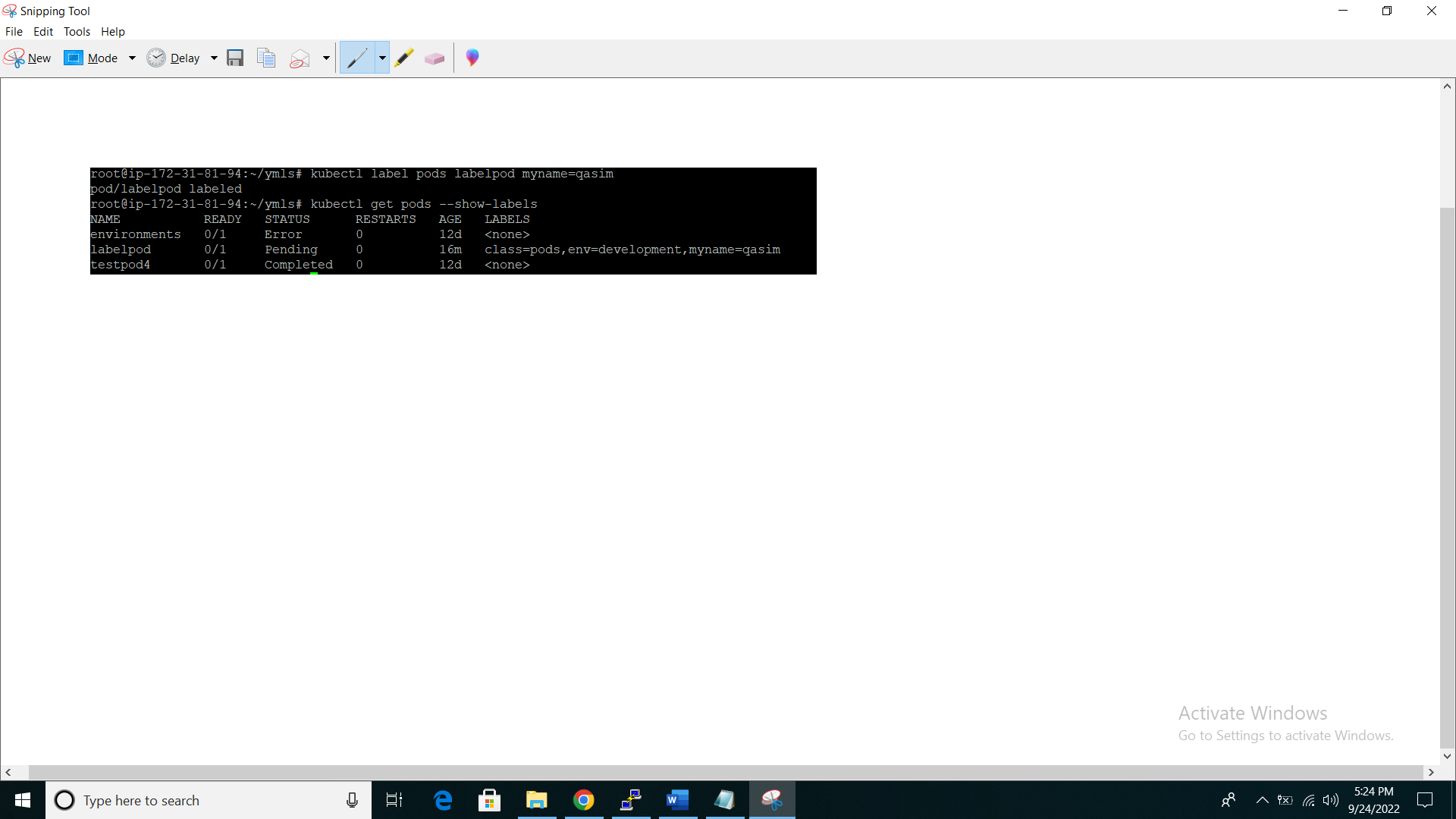
//Now check labels in pods

* Kubectl get pods –show-labels



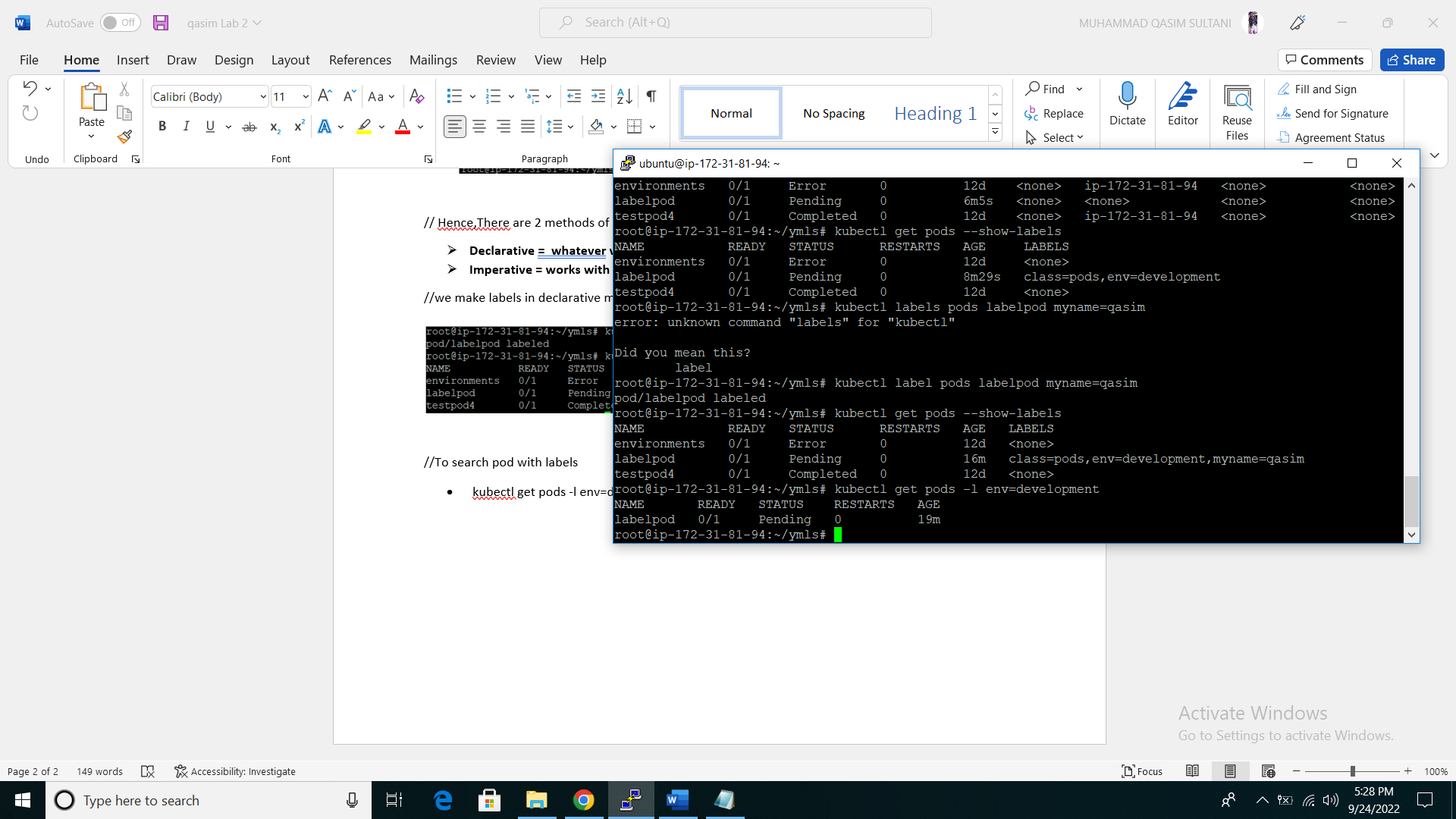
// Hence,There are 2 methods of doing anything

* **Declarative = whatever work you want, you write in yml file and apply it one time**
* **Imperative = works with command individually**

//we make labels in declarative method now we check with imperative   
  


//To search pod with labels

* kubectl get pods -l env=development

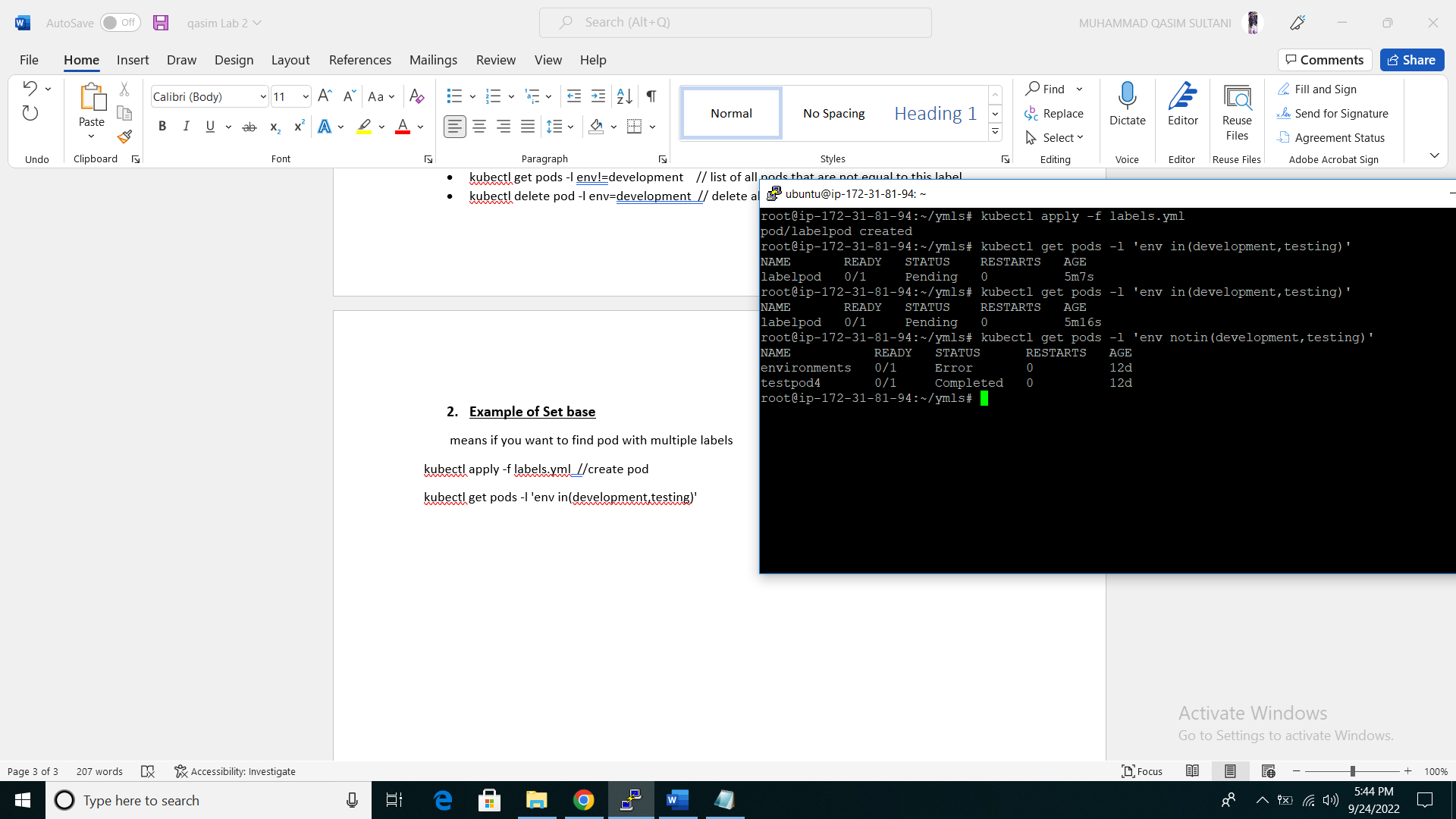


* kubectl get pods -l env!=development // list of all pods that are not equal to this label
* kubectl delete pod -l env=development // delete all pod with the help of label

1. **Example of Set base**

means if you want to find pod with multiple labels

* kubectl apply -f labels.yml //create pod
* kubectl get pods -l 'env in(development,testing)'
* kubectl get pods -l 'env notin(development,testing)'



//now check specific pod with spefic label

* kubectl get pods -l skill=devops
* kubectl get pods --show-labels
* kubectl get pods -l skill=devops

//you can delete the pods with the help o lables like this

* kubectl delete pods -l 'env in(development,testing)'

**NODE SELECTOR**

**NODE SELECTOR EXAMPLE**

kind: Pod

apiVersion: v1

metadata:

name: nodelabels

labels:

env: development

spec:

containers:

- name: c00

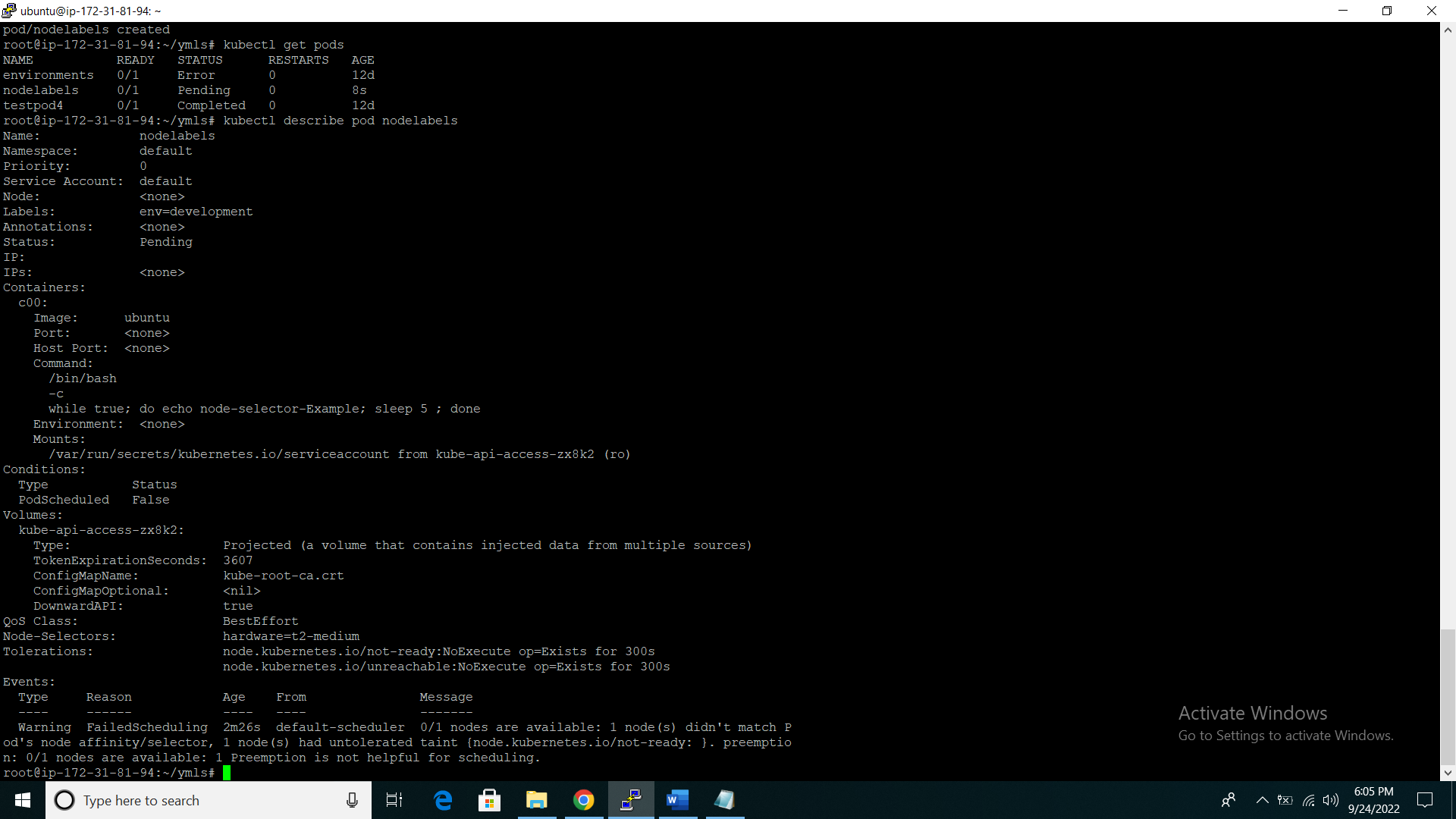
image: ubuntu

command: ["/bin/bash", "-c", "while true; do echo node-selector-Example; sleep 5 ; done"]

nodeSelector:

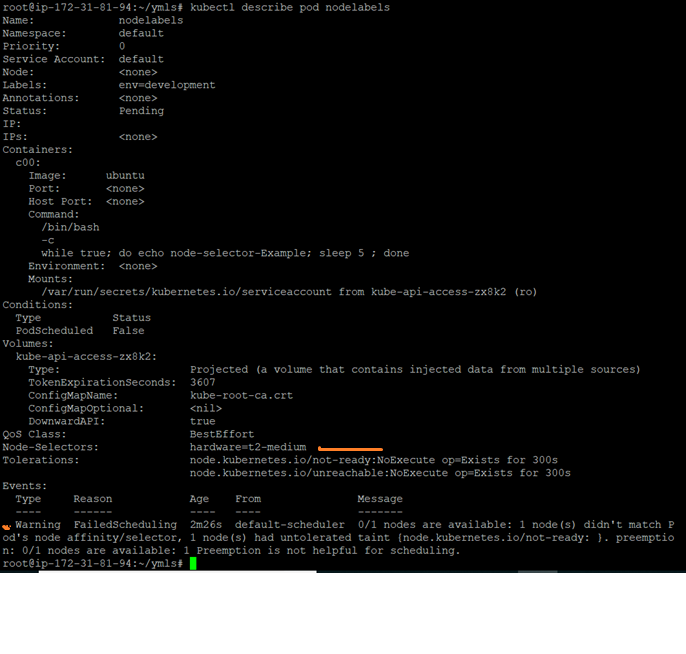
hardware: t2-medium

* kubectl apply -f node-sc.yml
* kubectl get pods



// Now you can see you pod is create “nodelabels” but not in ready state why?

* kubectl describe pod nodelabels //to more information of pod

// you can see your nodeselector is hardware=t2=medium label and warning show that they did not get any node which have hardware=t2=medium label so that why it’s not in ready state   


//So same as pods we can also label on nodes

**Label on nodes**

* kubectl get nodes
* kubectl label nodes ip-172-31-81-94 hardware=t2-medium
* kubectl describe pod nodelabels
* kubectl get pods //ready status 1/1
* kubectl delete -f node-sc.yml //delete pod

1. **EXAMPLE OF REPLICATION CONTROLLER**

kind: ReplicationController

apiVersion: v1

metadata:

name: my-replica

spec:

replicas: 3

selector:

myname: sultani

template:

metadata:

name: RC-Pod

labels:

myname: sultani

spec:

containers:

- name: c00

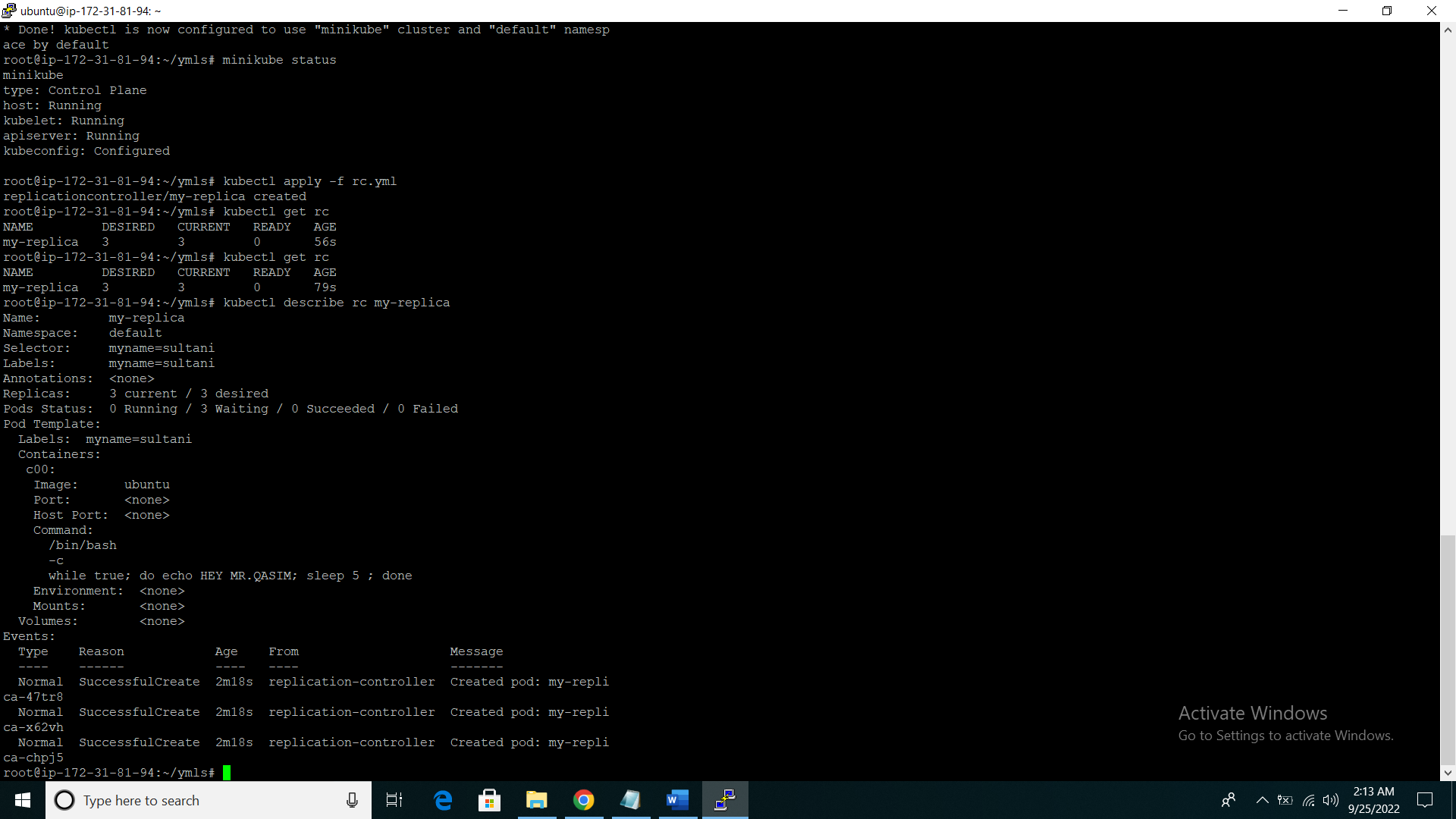
image: ubuntu

command: ["/bin/bash", "-c", "while true; do echo HEY MR.QASIM; sleep 5 ; done"]

* kubectl apply -f rc.yml
* kubectl get rc //to check rc pods available
* kubectl describe rc my-replica //to check details about replica-controller “my-replica” is replica-controller name

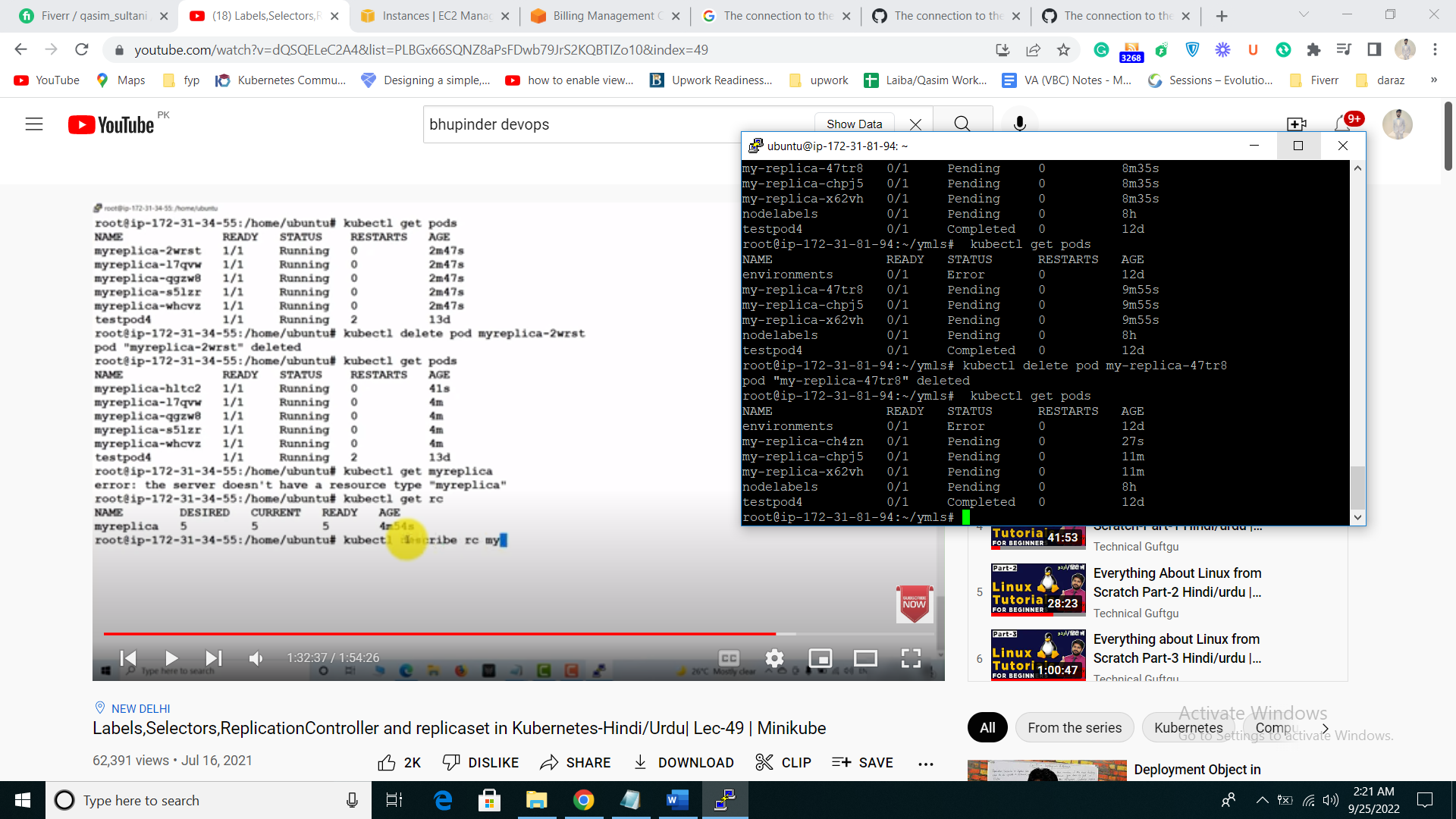
//to get to know more about replica-controller pods you can check the description on below screen shot

* selector or labels are create at same name
* Current and desired state are always same
* At last 3 replica set are create which was we mention on yml



//now delete any new replica pod and check the state there is always 3 replica controller pods created because we give the number of 3 desired number state into replica-controller yml file

* kubectl get pods
* kubectl delete pod my-replica-47tr8



**SCALE-UP REPLICASET**

// Now we can Scale-up your replica-controller pods assume you have 3 pods and you want to scale up into 6

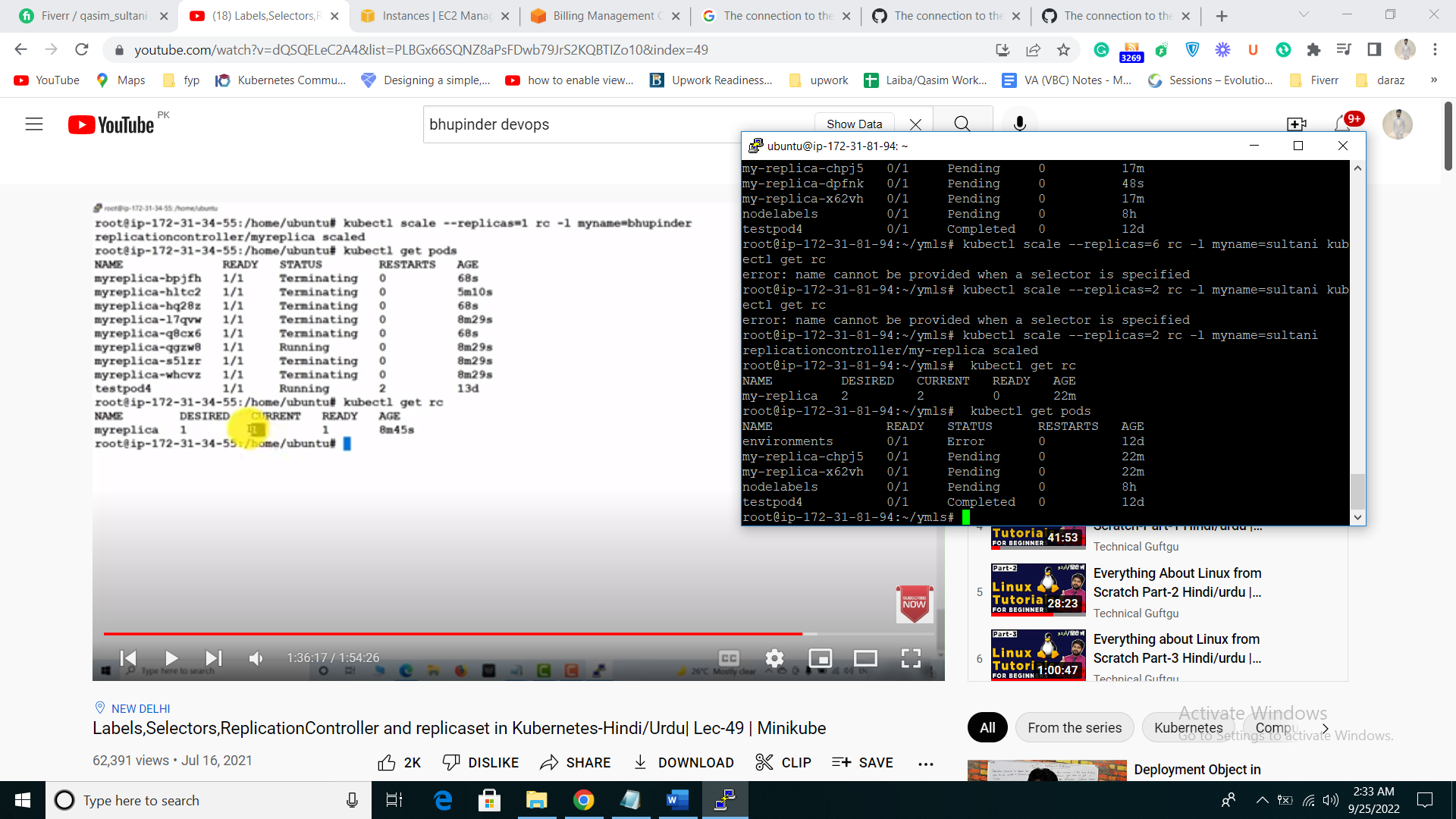
* kubectl scale --replicas=6 rc -l myname=sultani
* kubectl get rc
* kubectl get pods



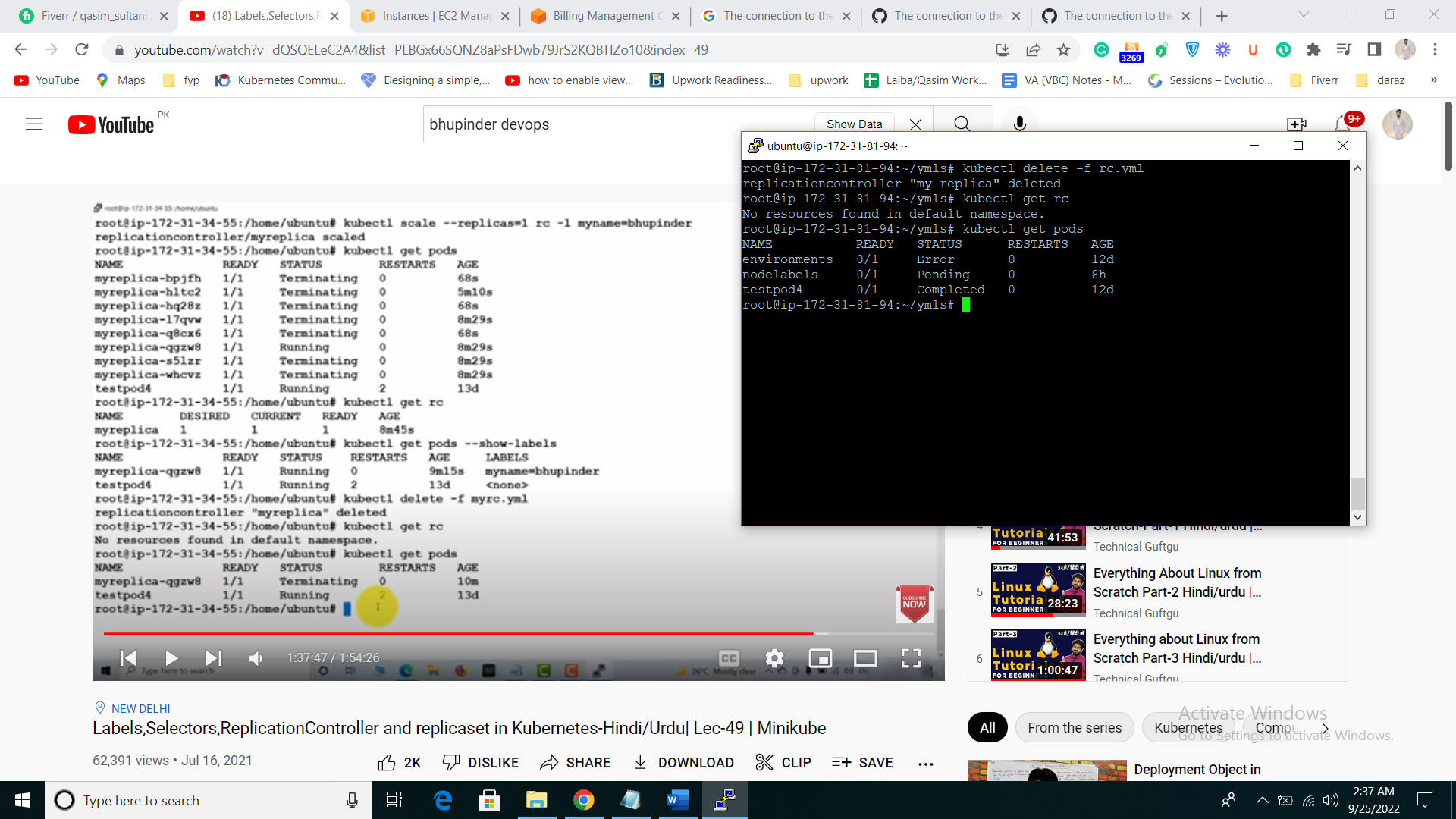
**SCALE-DOWN REPLICASET**

// Now you want to Scale-down your replica-controller pods assume you have 6 pods and you want to scale up into 2

* kubectl scale --replicas=2 rc -l myname=sultani
* kubectl get rc
* kubectl get pods



//to delete repliccontroller you have to delete repliccontroller yml file not pod if you delete pods it will create again pod because we define them into file so delete file



**REPLICA SET**

* Advance version of replica controller
* it is equality base as well as set base

**EXAMPLE OF REPLICA SET**

kind: ReplicaSet

apiVersion: apps/v1

metadata:

name: myrs

spec:

replicas: 4

selector:

matchExpressions: # these must match the labels

- {key: myname, operator: In, values: [muhammadqasimsultani, qasimsultani, muhammadsultani]}

- {key: env, operator: NotIn, values: [production]}

template:

metadata:

name: testpod7

labels:

myname: qasimsultani

spec:

containers:

- name: c00

image: ubuntu

command: ["/bin/bash", "-c", "while true; do echo Hello Summer; sleep 5 ; done"]

* kubectl apply -f replicaset.yml
* kubectl get rs
* kubectl get pods
* kubectl scale --replicas=8 rs/replicaset //replicaset is your yml file name in rs