Cluster autoscaling

Ref - https://github.com/kubernetes/autoscaler

Cluster Autoscaler - a component that automatically adjusts the size of a Kubernetes Cluster so that all pods have a place to run and there are no unneeded nodes. Supports several public cloud providers. Version 1.0 (GA) was released with kubernetes 1.8.

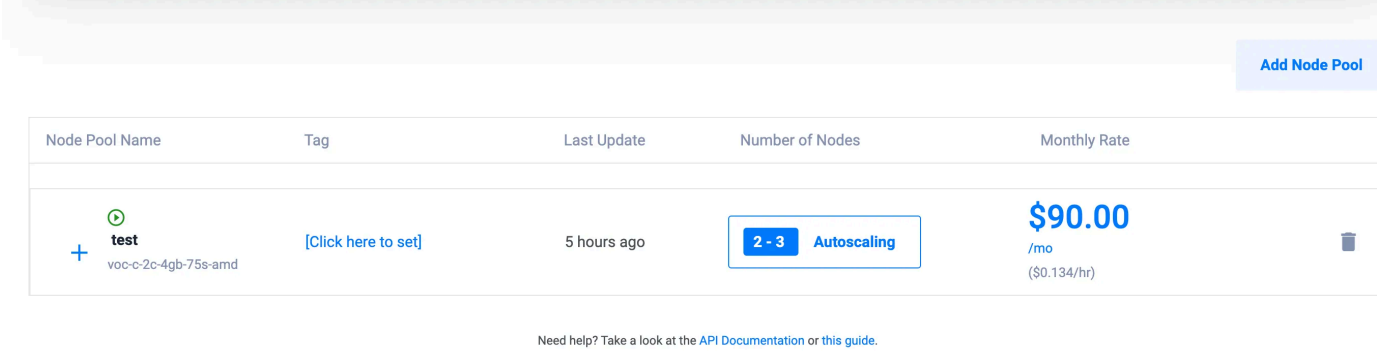
Underprovisioned resources

In the last slide, we saw that we didn't have enough resources to schedule a pod on.

```
    Type:                    Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:  3607
    ConfigMapName:           kube-root-ca.crt
    ConfigMapOptional:       <nil>
    DownwardAPI:             true
QoS Class:                   Burstable
Node-Selectors:              <none>
Tolerations:                 node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                             node.kubernetes.io/unreachable:NoExecute op=Exists for 300s

Events:
  Type     Reason            Age   From               Message
  ----     ------            ----  ----               -------
  Warning  FailedScheduling  17s   default-scheduler  0/2 nodes are available: 2 Insufficient cpu. preemption: 0/2 nodes are available: 2 No pre
emption victims found for incoming pod.
  Normal   NotTriggerScaleUp  3s    cluster-autoscaler  pod didn't trigger scale-up: 1 max node group size reached
➜  week-28 git:(main) ✗
```

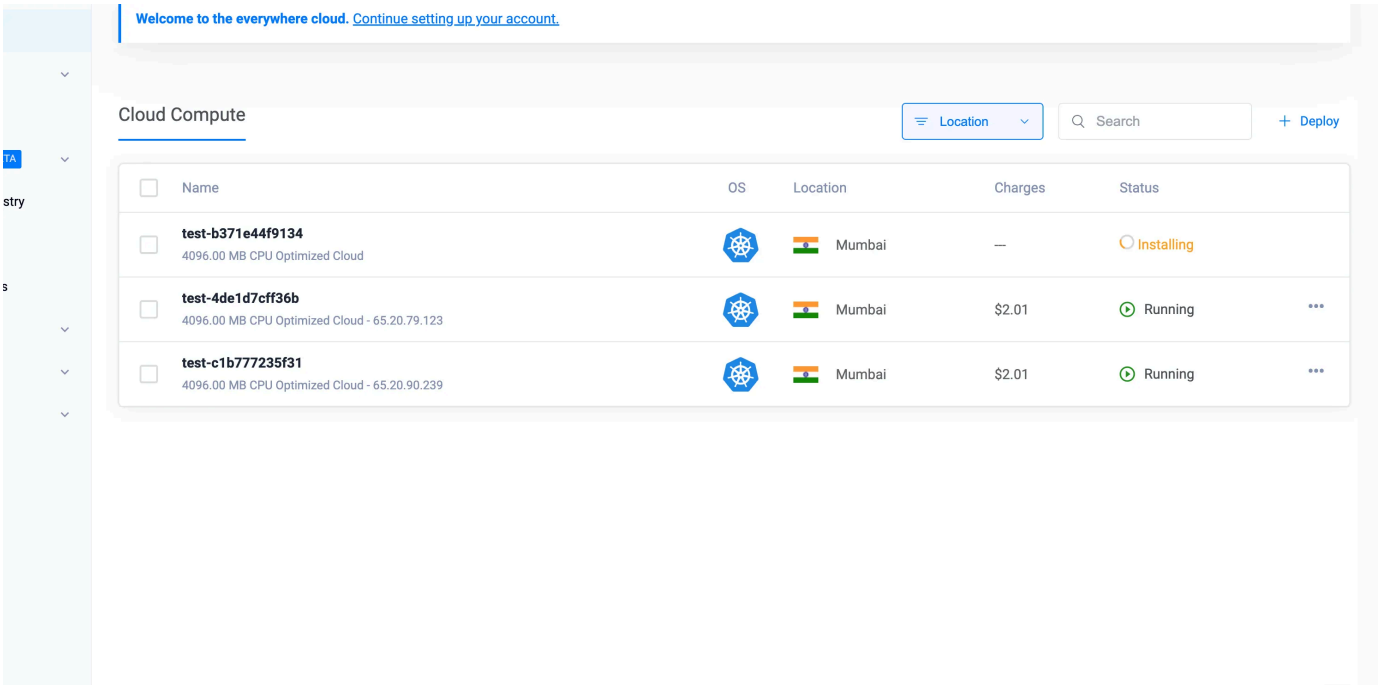
Let's make our node pool dynamic and add a min and max nodes.



Restart the deployment

```
kubectl delete deployment cpu-deployment
kubectl apply -f deployment.yml
```

Notice a new node gets deployed



Logs of the cluster autoscaler

kubectl get pods -n kube-system | grep cluster-autoscaler

```
I0615 11:32:54.972871       1 klogx.go:87] Pod default/cpu-deployment-f8d5fd76d-4xz2f is unschedulable
I0615 11:32:54.972873       1 klogx.go:87] Pod default/cpu-deployment-f8d5fd76d-6s5rn is unschedulable
I0615 11:32:54.972875       1 klogx.go:87] Pod default/cpu-deployment-f8d5fd76d-6jhhw is unschedulable
I0615 11:32:54.975029       1 orchestrator.go:184] Best option to resize: e9ff0452-1b32-45ca-b138-3378fd5c861f
I0615 11:32:54.975047       1 orchestrator.go:188] Estimated 1 nodes needed in e9ff0452-1b32-45ca-b138-3378fd5c861f
I0615 11:32:54.975062       1 orchestrator.go:257] Final scale-up plan: [{e9ff0452-1b32-45ca-b138-3378fd5c861f 2->3 (max: 3)}]
I0615 11:32:54.976329       1 executor.go:147] Scale-up: setting group e9ff0452-1b32-45ca-b138-3378fd5c861f size to 3
I0615 11:33:07.498451       1 static_autoscaler.go:432] 1 unregistered nodes present
I0615 11:33:07.499071       1 filter_out_schedulable.go:75] Schedulable pods present
I0615 11:33:07.499099       1 klogx.go:87] Pod default/cpu-deployment-f8d5fd76d-4xz2f is unschedulable
I0615 11:33:07.499103       1 klogx.go:87] Pod default/cpu-deployment-f8d5fd76d-6s5rn is unschedulable
I0615 11:33:07.499106       1 klogx.go:87] Pod default/cpu-deployment-f8d5fd76d-6jhhw is unschedulable
I0615 11:33:07.499108       1 klogx.go:87] Pod default/cpu-deployment-f8d5fd76d-8xvkg is unschedulable
I0615 11:33:07.500109       1 orchestrator.go:167] No expansion options
```

Try downscaling

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: cpu-deployment
  ...
  selector:
```

```
  matchLabels:
    app: cpu-app
 template:
  metadata:
    labels:
      app: cpu-app
  spec:
    containers:
    - name: cpu-app
      image: 100xdevs/week-28:latest
      ports:
      - containerPort: 3000
      resources:
        limits:
          cpu: "1000m"
        requests:
          cpu: "1000m"
```

Notice the number of server goes down to 2 again

Good things to learn after this -

1. Gitops (ArgoCD)
2. Custom metrics based scaling, event based autoscaling - https://www.giffgaff.io/tech/event-driven-autoscaling
3. Deploying prometheus in a k8s cluster, scaling based on custom metrics from prometheus