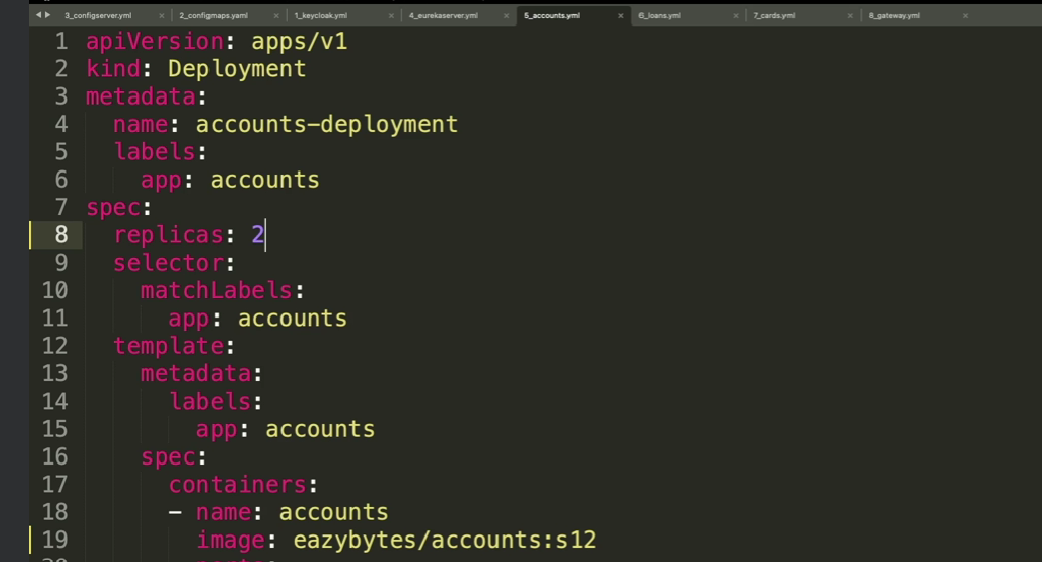
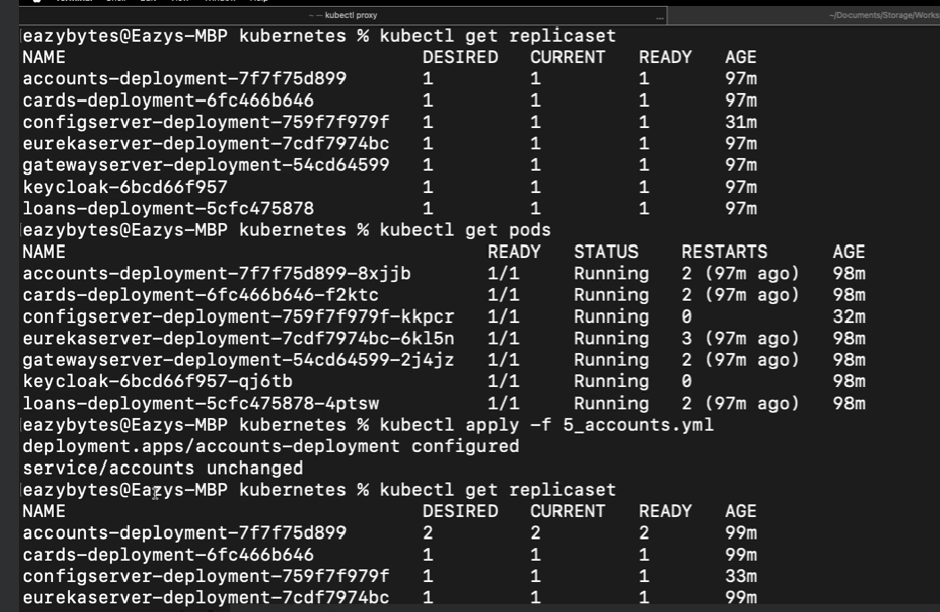
Real Use of Kubernetes

Self-healing inside Kubernetes cluster, That is Kubernetes is responsible for keeping track of the health of the containers. If any container is down or not able to work properly then Kubernetes will try to self heal the container by recreating it. Lets check by doing a small demo

Lets increase the replicas for Accounts microservice by changing replicas properties in manifest file.

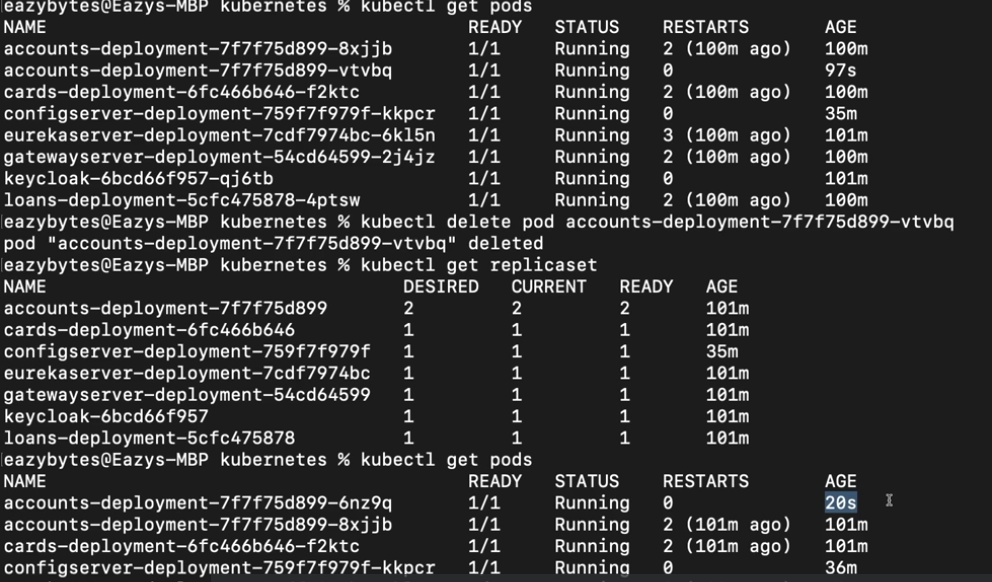




As you can see now Account service has 2 instances running also you can run

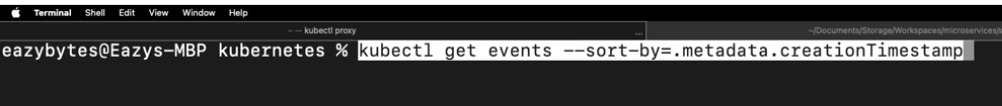
Kubectl get pods

Now lets delete one of the pods using delete command



But after some time when you execute again get pods you should see new pods created for accounts service so this is the beauty of Kubernetes which is not accomplished by Docker or Docker compose.

You can also check get events command to get all the events happened

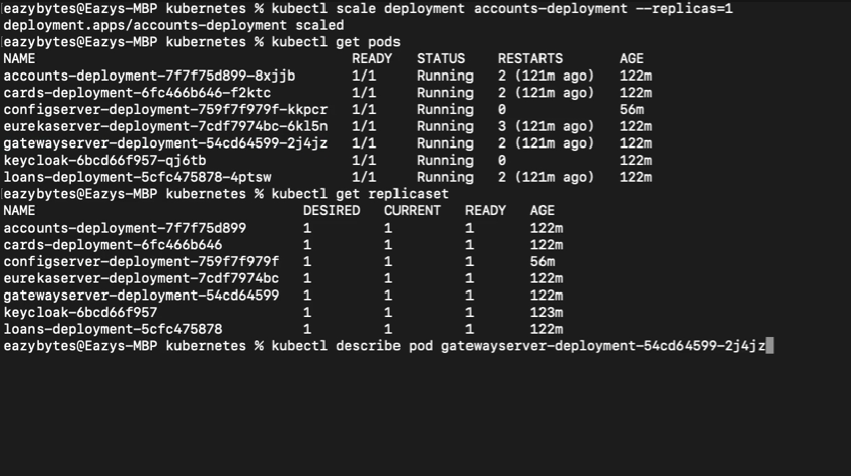


**Automatic rollout and rollbacks**

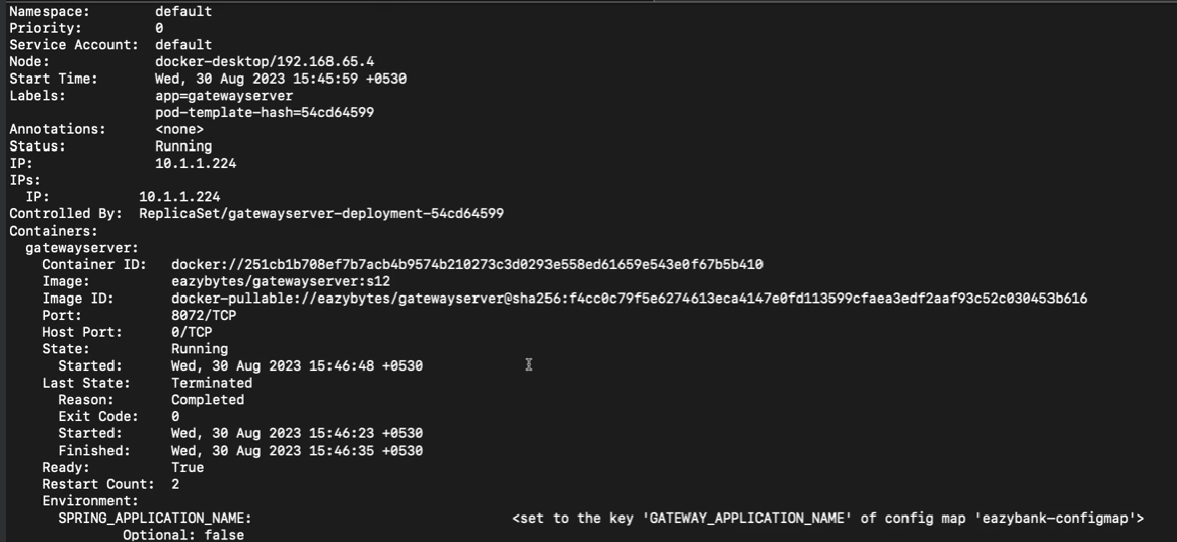
Rollout feature is another important feature of Kubernetes where we can deploy new changes without the downtime lets check this by doing demo

Lets change gateway server where it is modified to ignore security token to access API’s

Lets run describe command for gateway server



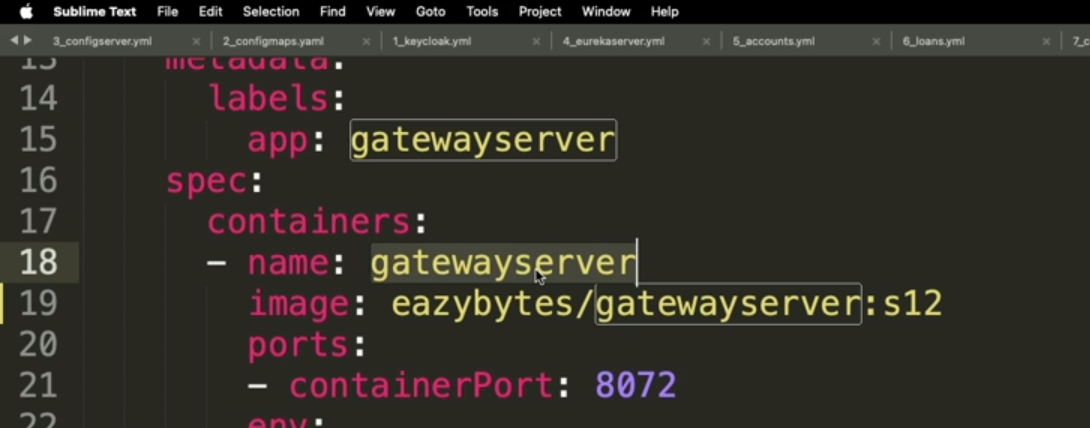
You will get detailed information of pod



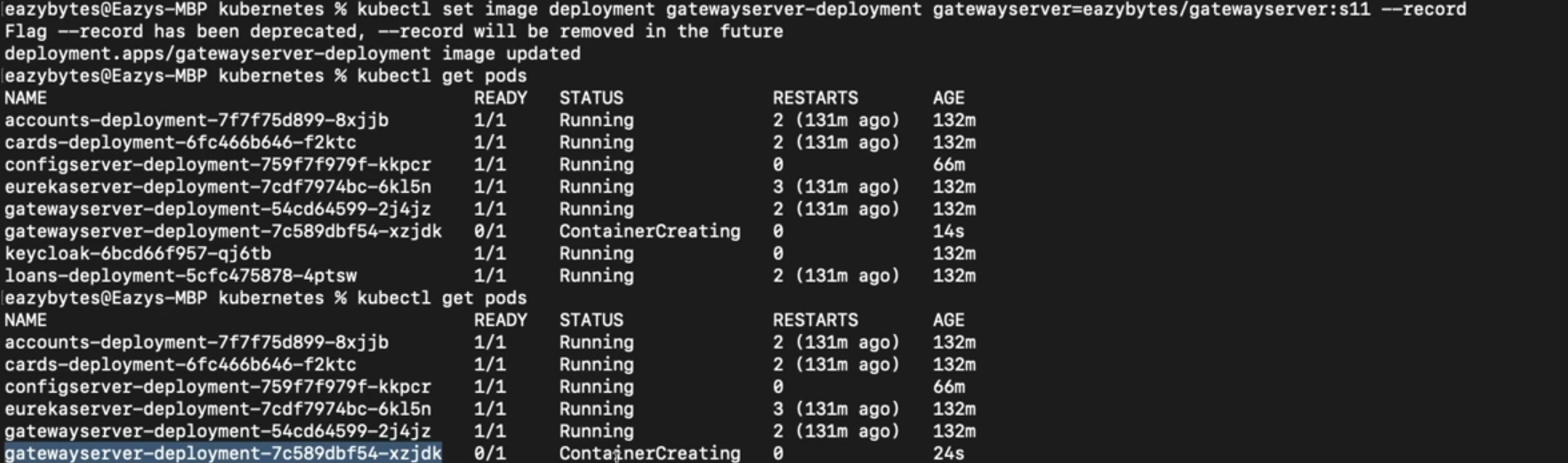
We can deploy older version of gateway server to do that we can simply run a below command where it will install an version 11 code



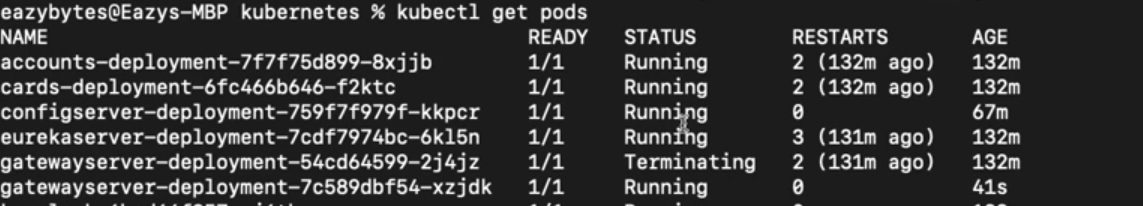
Here gatewayserver= is actually an container name that we have used in manifest file



When you execute the above command new gateway server will be getting creating and older version will be still running once the new version is created then automatically older version will get disabled.

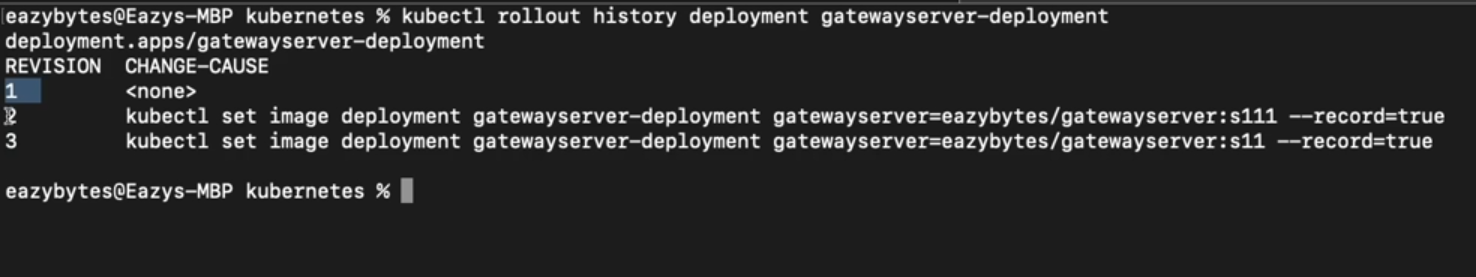


After some time gateway server will be terminated and new gatewayserver will be running



If we have multiple instances then same will be done in incremental fashion.

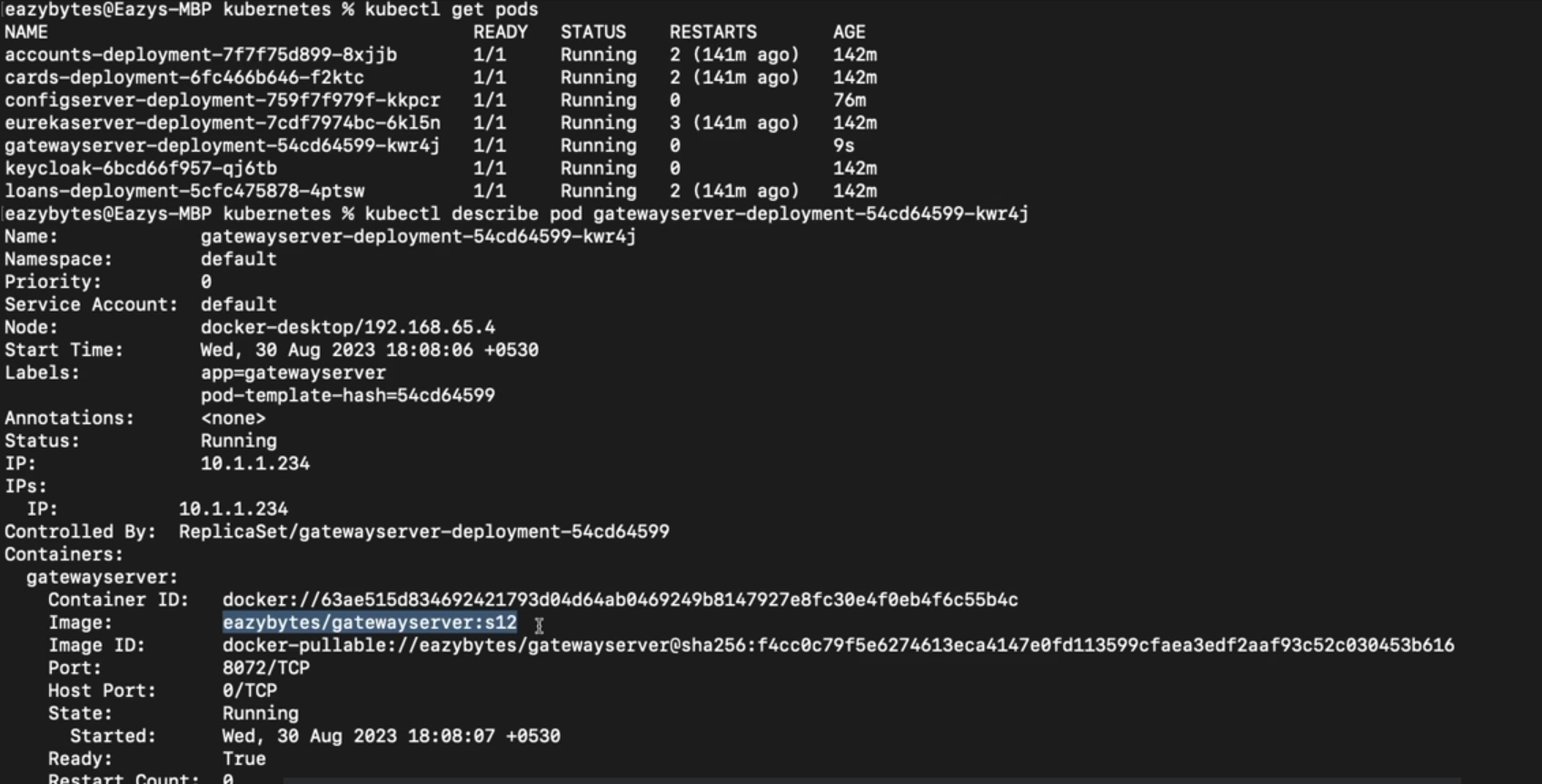
You can also check rollout history



Now if we want to rollback to the previous version then we can execute below command

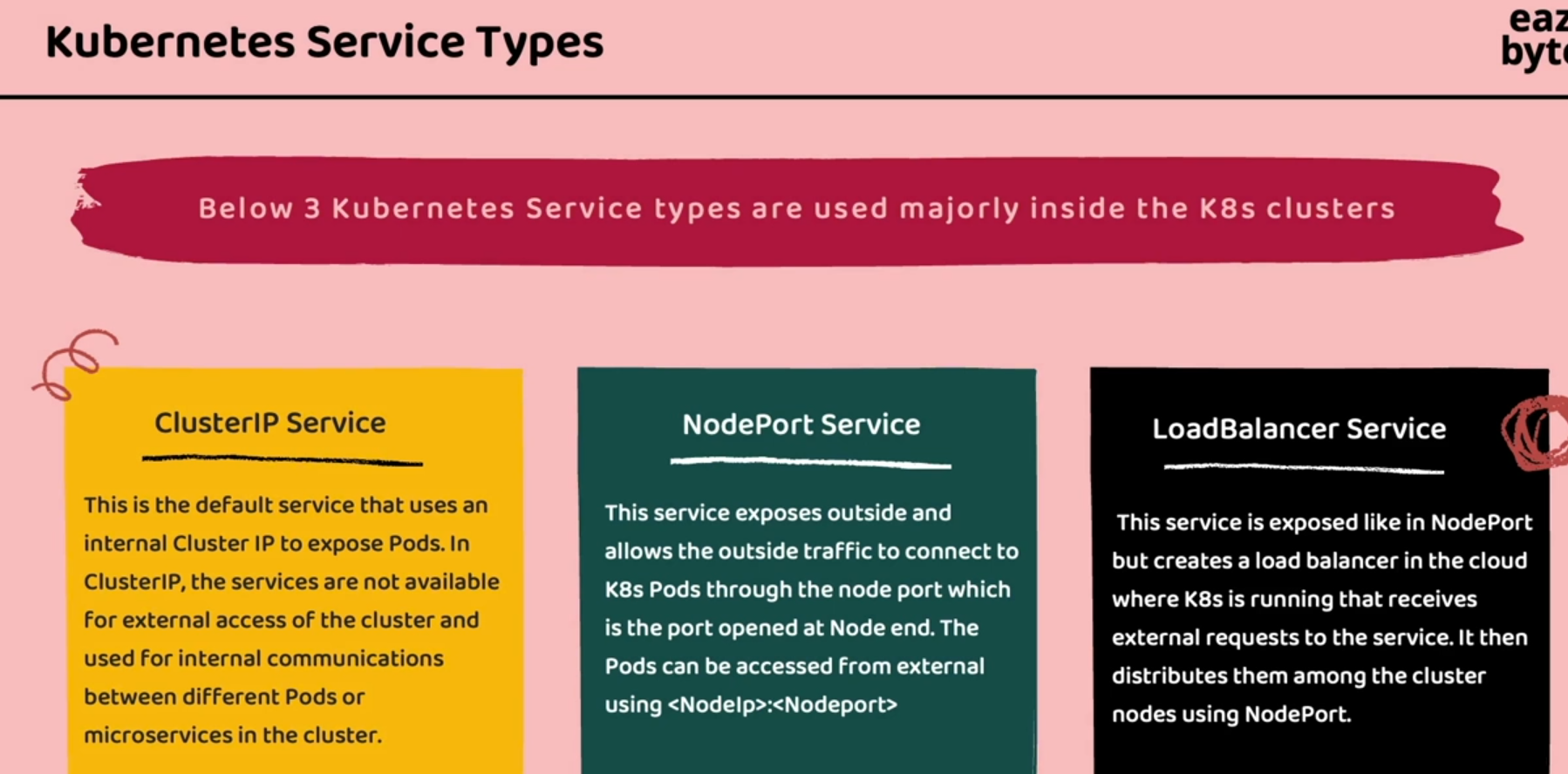


Kubernetes will treat the previous version as revision=1 hence we can rollback to older version with above command

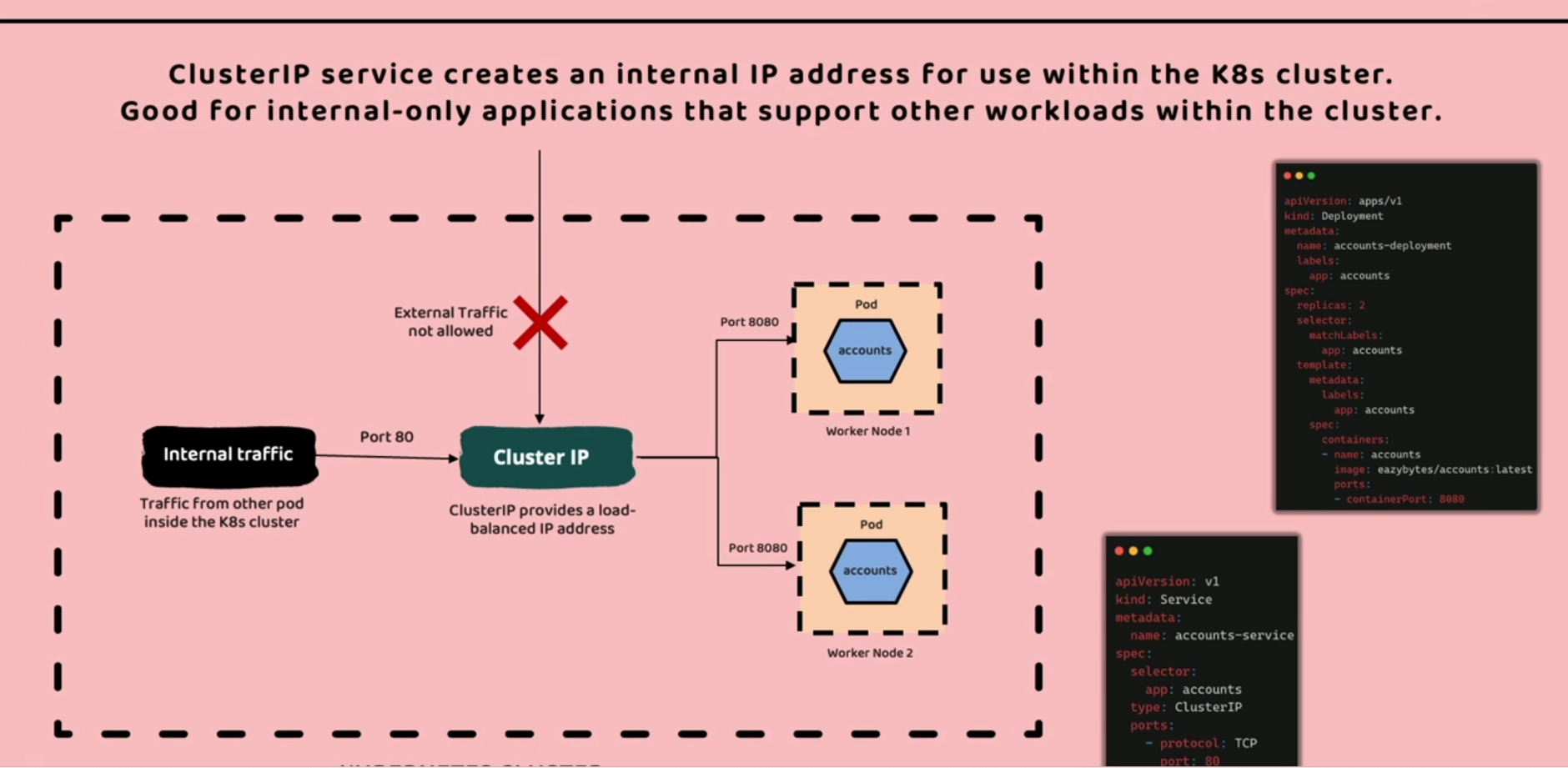


As of now services are created using LoadBalancer service type which is used to expose the services to the outside world.

Kubernetes supports 3 types of service as below

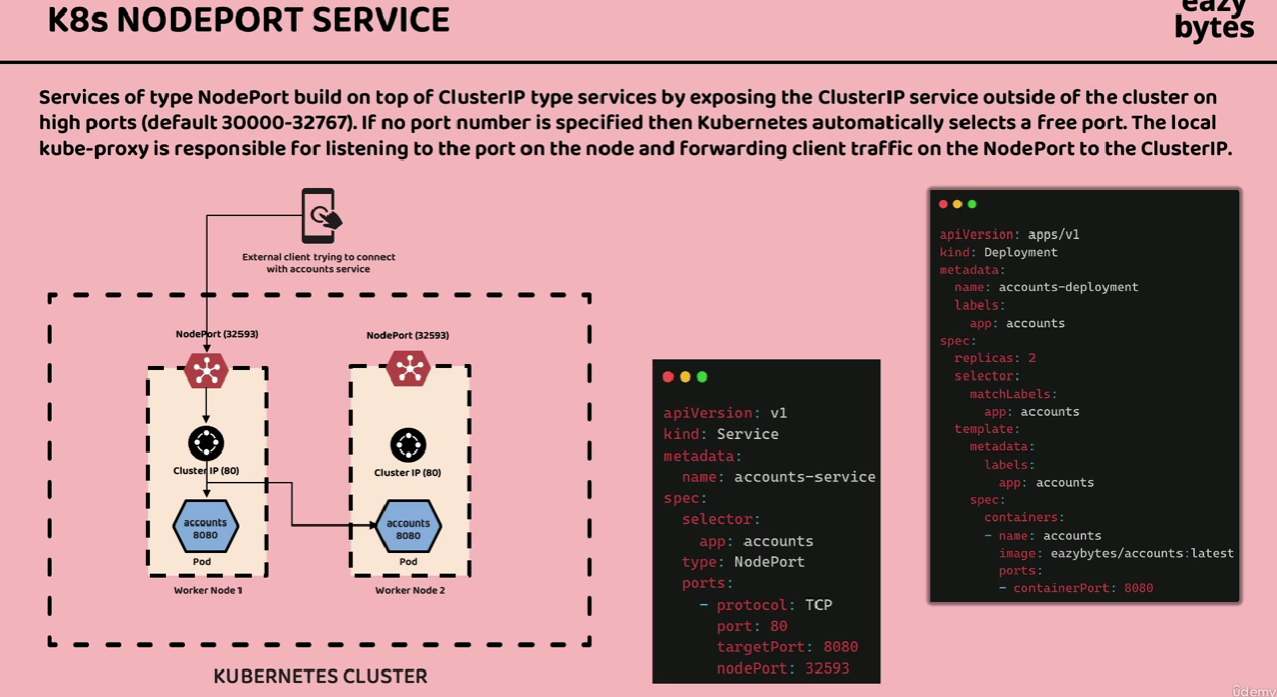


ClusterIP Service

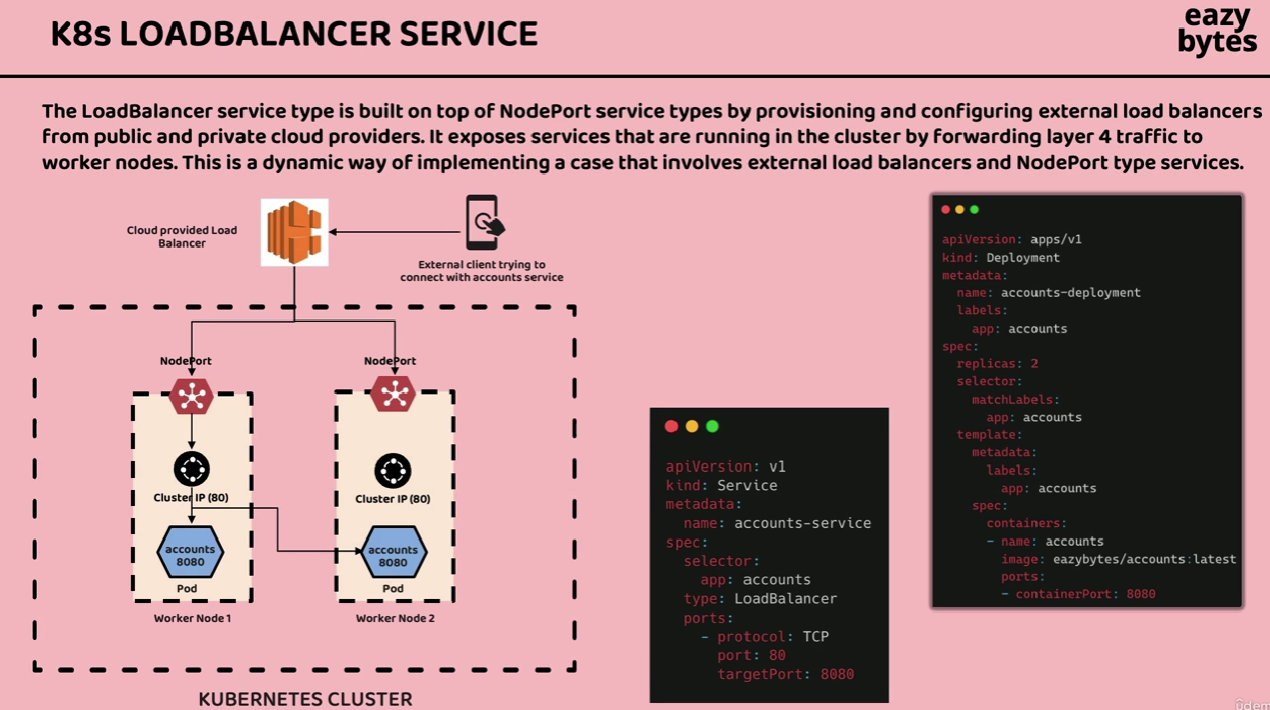


In clusterIP service type the pods are not exposed to outside world but to the internal services. Whenever any service is sending the request at port 80 using clustedIP. In this Kubernetes is going to do load balancing it will decide to which pod request has to be sent. So here we don’t have to know about IP address or the worker node address simply by using port 80 we can access that particular service via clusterIP. And also this is the one of the way of securing our microservices from the external traffic.

NodePort Service



Here external client can invoke the service via NodePort if we don’t mention nodePort in manifest file then Kubernetes automatically assign any port. But there is a drawback here client should know the port and ip address. If any pod is deleted and restarted then port and ip address are assigned different hence we at client also we need to update such details hence it is not recommended approach.



In loadbalancer service we can configure cloud provider to have fixed IP address to the cluster hence outside world can access through this IP only loadbalancer is responsible to identify worker node based on the port here 80 is for accounts service and it sends the request to either worker node1 or node2 internally this request is then passed to the nodeport and then forwarded to cluster ip .

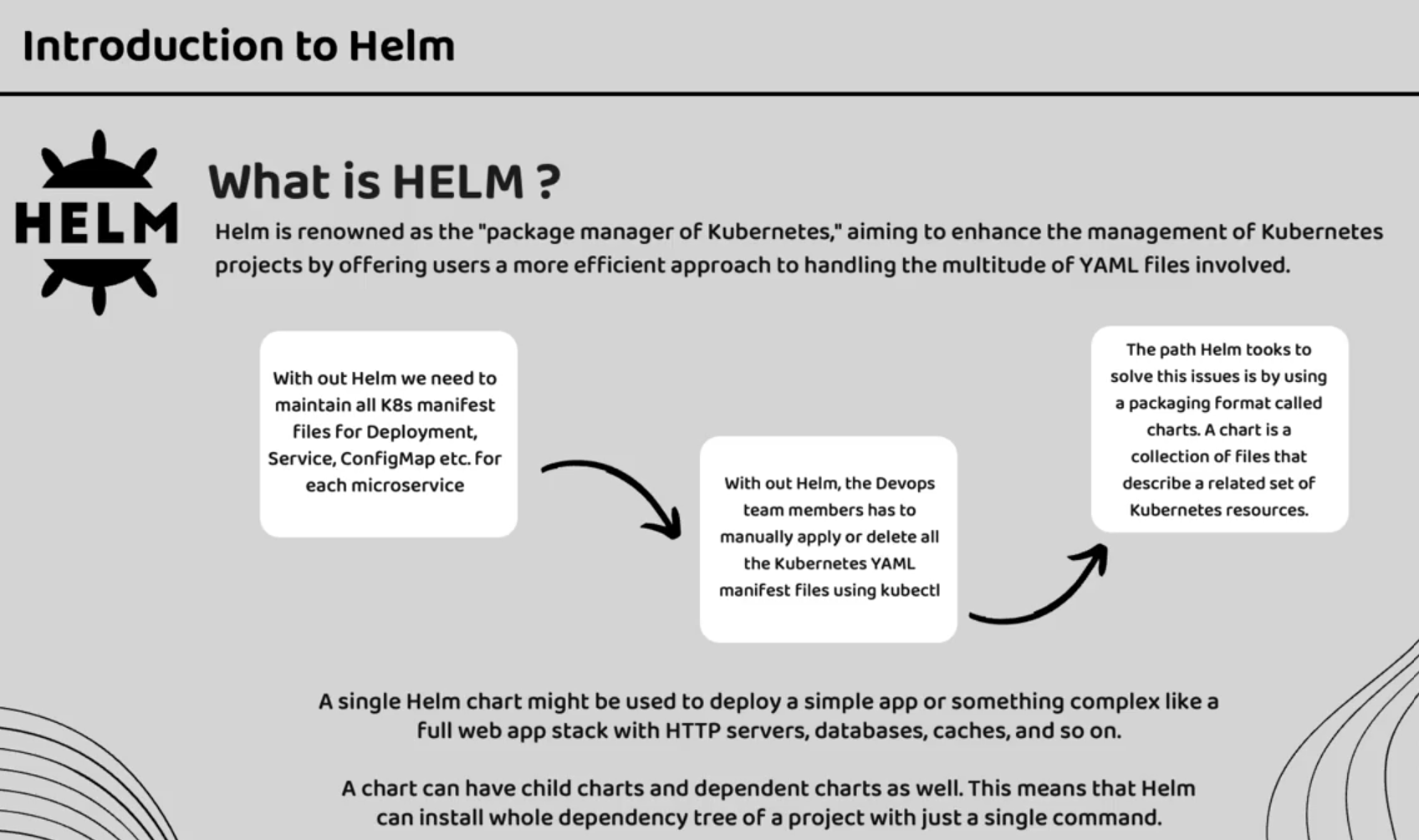
To check Demo check video 230

Disadvantages of using manifest file

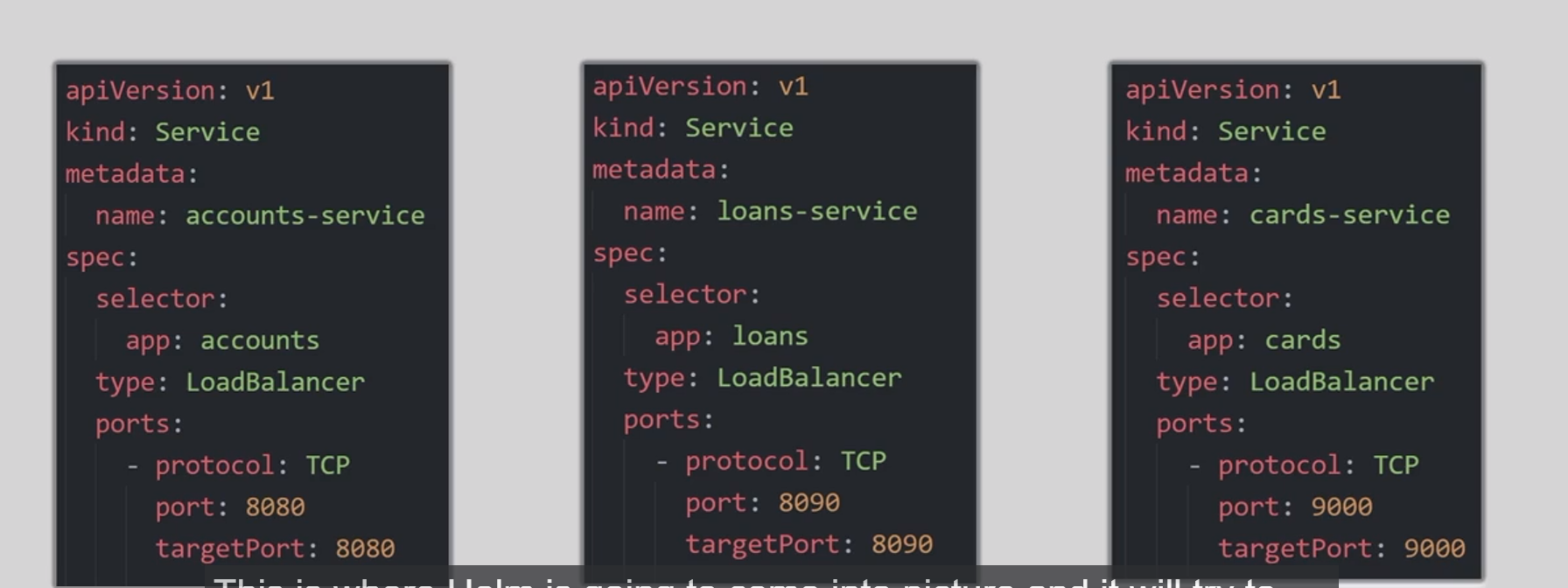
Suppose there are larger number of microservices then creating manifest file for all the services is really a cumbersome process.

Applying the changes is also a tedious task as we have to execute apply command for individual microservices and also if we want to empty the cluster then we have to execute delete command.

So, this can be avoided using HELM



Special type of package is created called charts to overcome the disadvantages of the manifest file. It actually contains the set of files related to Kubernetes resources



If you look at the Service object file there are some fields which are static and some are dynamic. While the template remains the same for all the service object. In such scenario we can use helm to overcome this multiple service object. We can create a single template file for all the above Service object file and inject values dynamically

