A quick note on editing PODs and Deployments

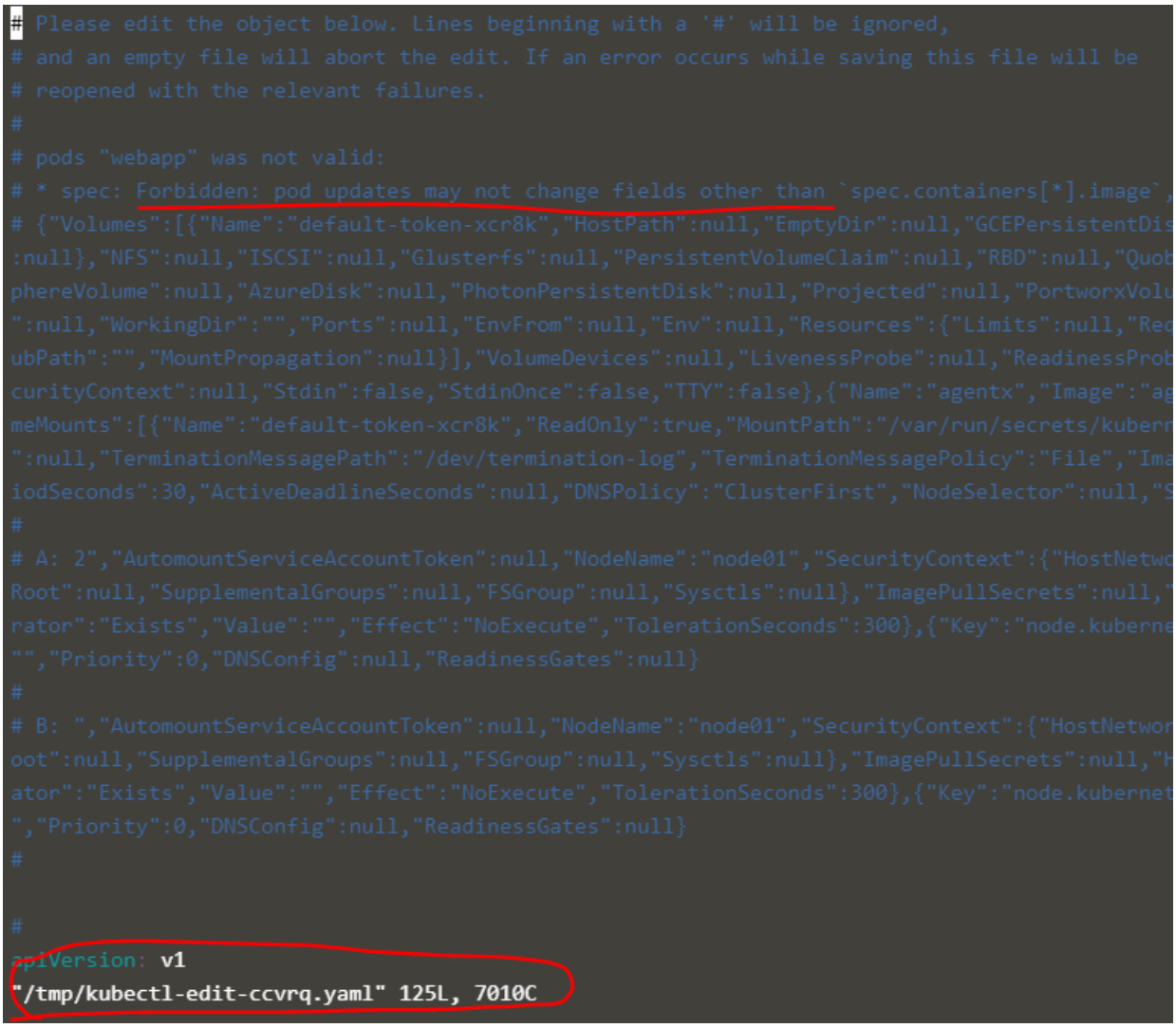
Edit a POD

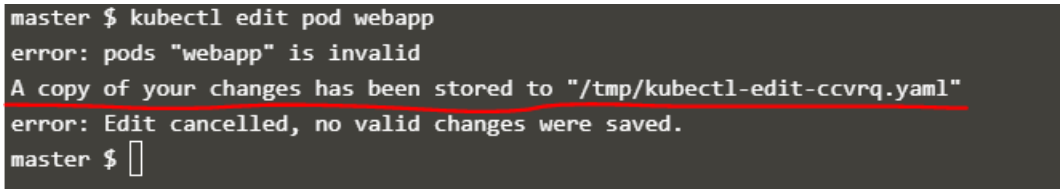
Remember, you CANNOT edit specifications of an existing POD other than the below.

- spec.containers[*].image
- spec.initContainers[*].image
- spec.activeDeadlineSeconds
- spec.tolerations

For example you cannot edit the environment variables, service accounts, resource limits (all of which we will discuss later) of a running pod. But if you really want to, you have 2 options:

1. Run the `kubectl edit pod <pod name>` command. This will open the pod specification in an editor (vi editor). Then edit the required properties. When you try to save it, you will be denied. This is because you are attempting to edit a field on the pod that is not editable.





A copy of the file with your changes is saved in a temporary location as shown above.

You can then delete the existing pod by running the command:

```
kubectl delete pod webapp
```

Then create a new pod with your changes using the temporary file

```
kubectl create -f /tmp/kubectl-edit-ccvrq.yaml
```

2. The second option is to extract the pod definition in YAML format to a file using the command

```
kubectl get pod webapp -o yaml > my-new-pod.yaml
```

Then make the changes to the exported file using an editor (vi editor). Save the changes

```
vi my-new-pod.yaml
```

Then delete the existing pod

```
kubectl delete pod webapp
```

Then create a new pod with the edited file

```
kubectl create -f my-new-pod.yaml
```

Edit Deployments

With Deployments you can easily edit any field/property of the POD template. Since the pod template is a child of the deployment specification, with every change the deployment will automatically delete and create a new pod with the new changes. So if you are asked to edit a property of a POD part of a deployment you may do that simply by running the command

```
kubectl edit deployment my-deployment
```