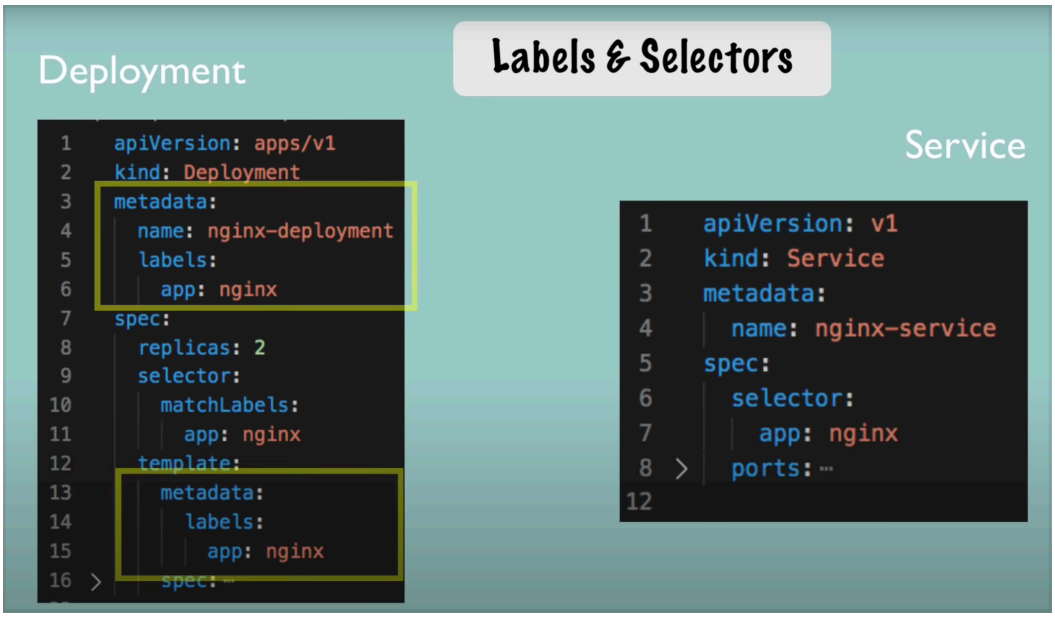
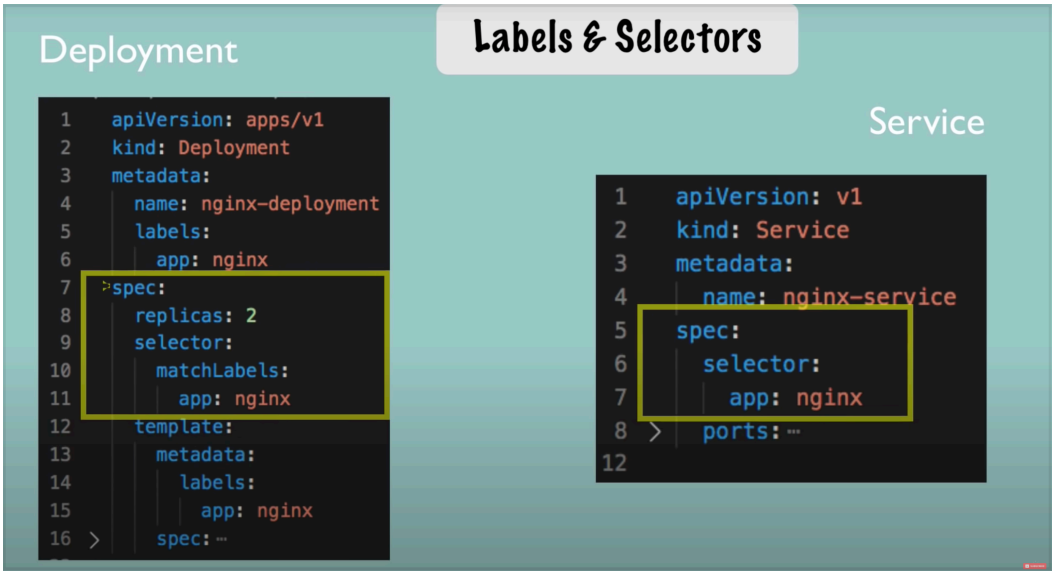
K8s YAML Configuration File

- The 3 parts of the Configuration File
  - kind, apiVersion (which will be different based on the component)
  - 1) is the metadata of the component that you are creating resides, the name of it
  - 2) is the specification , where you basically put every kind of configuration that you want to apply for that component , attributes of “spec” are specific to the “kind” that you are creating
  - 3) will be the status, it’s gonna be automatically generated and added by K8s, will compare what is the desired and actual state/ status of that component and if not equal K8s will try to fix it.
    - K8s updates state continuously,
    - If you have question like, where does K8s get this status data? It is from the “etcd“ - which holds the current status of any K8s component.

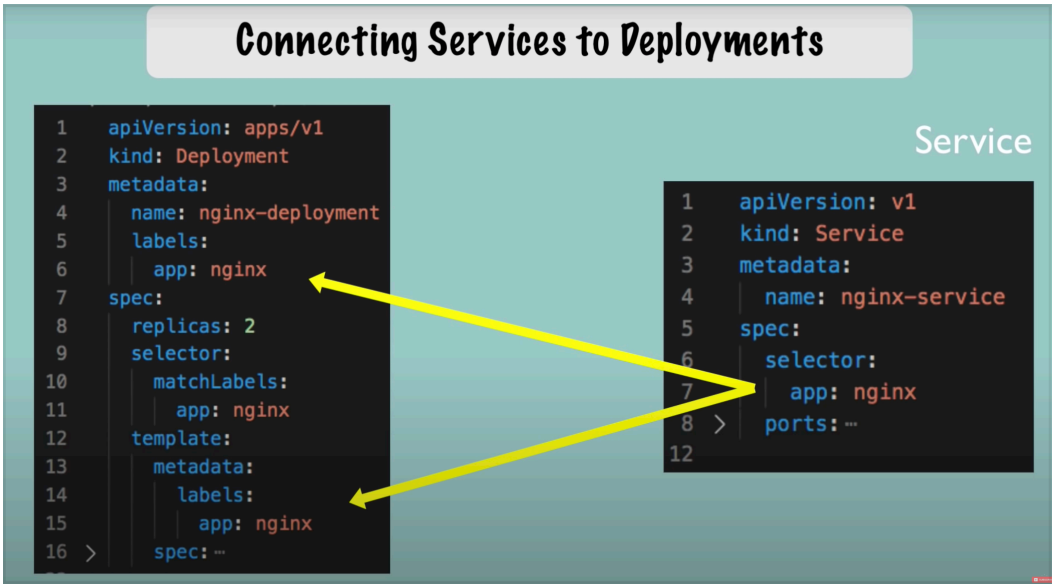
- Format of configuration File
  - YAML (human friendly data serialisation standard for all programming language, strict in indentation)
  - Usual practice is to store them with your code
  - We have been talking about layer of abstraction like, if you update deployment rest will be taken care by K8s, so Howard where does this happen, it’s in ”template”
  - template
    - Has its own “metadata” and “spec” section
    - It’s kind of config file inside config file.
    - This config applies to Pod, so Pod should have it’s own config inside of deployment config files.
    - blueprint for a Pod, like port? Name? Image

- Connecting components (Labels, Selectors & Ports)
  - https://www.youtube.com/watch?v=X48VuDVv0do&t=2462s
  -

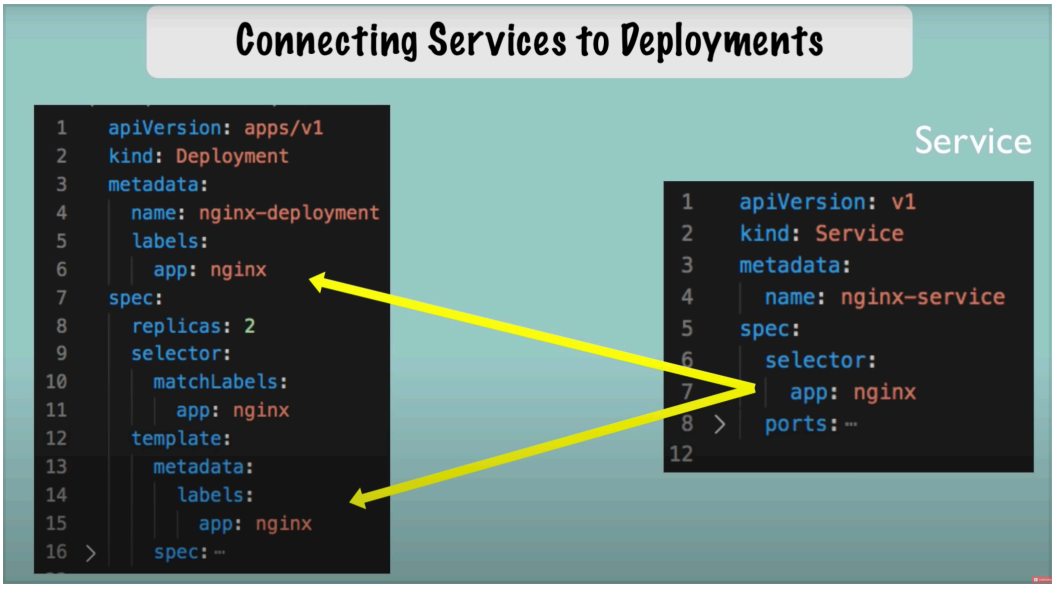




- Any key-value pair for component, like app: nginx
  - Pods get the label through the template blueprint
  - This label is matched by the selector, -> matchLabels
- deployments labels is used by service



- Service component will also have ports,



- This all are minimus confgiuration required
- Some cmds
  - kubectl get service
  - kubectl describe service _serivce_name_
  - And the endpoints will be the Ip address and port of the Pod
  - kubectl get pod -o (output) wide (more info)
  - kubectl get deployment _deployment_name_ -o yaml
    - Will get the result of updated config of the deployment from the “etcd”