

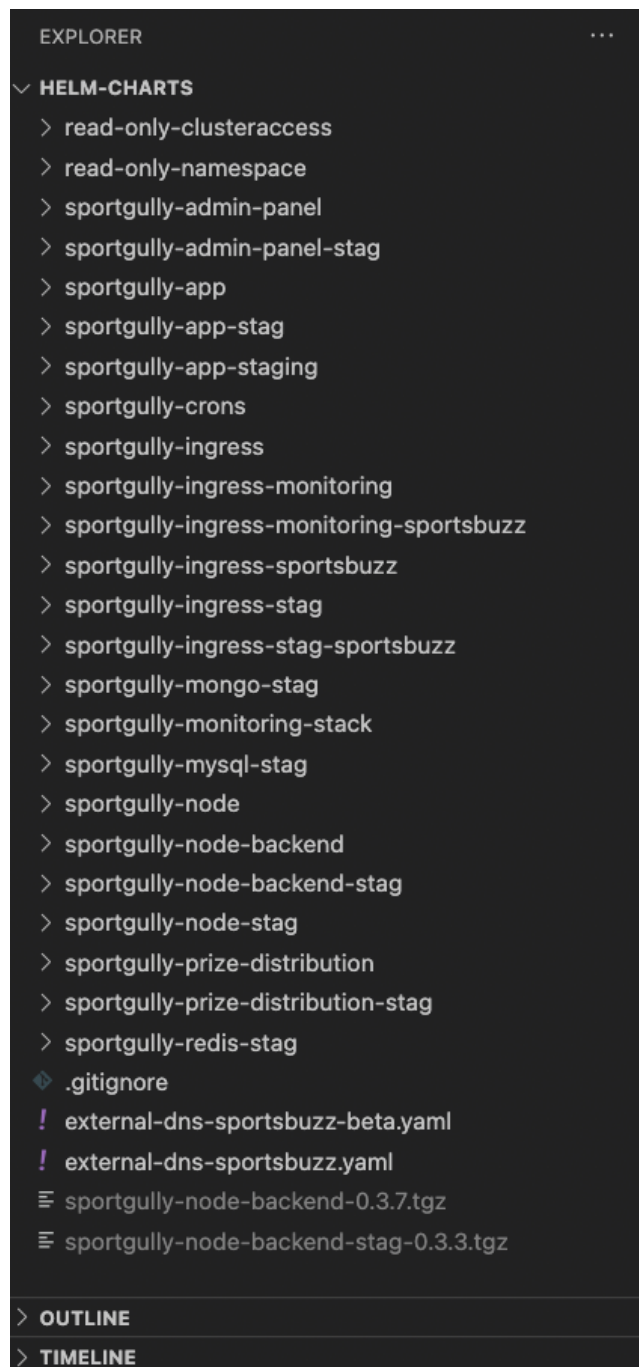
Configuring Helm Chart in the local system

First, install helm in your system. It can be downloaded from [here](#), for different OS versions, various methods are defined.

After installing helm, clone the helm repo in your system at your desired location, [here](#) you can find the repo link.

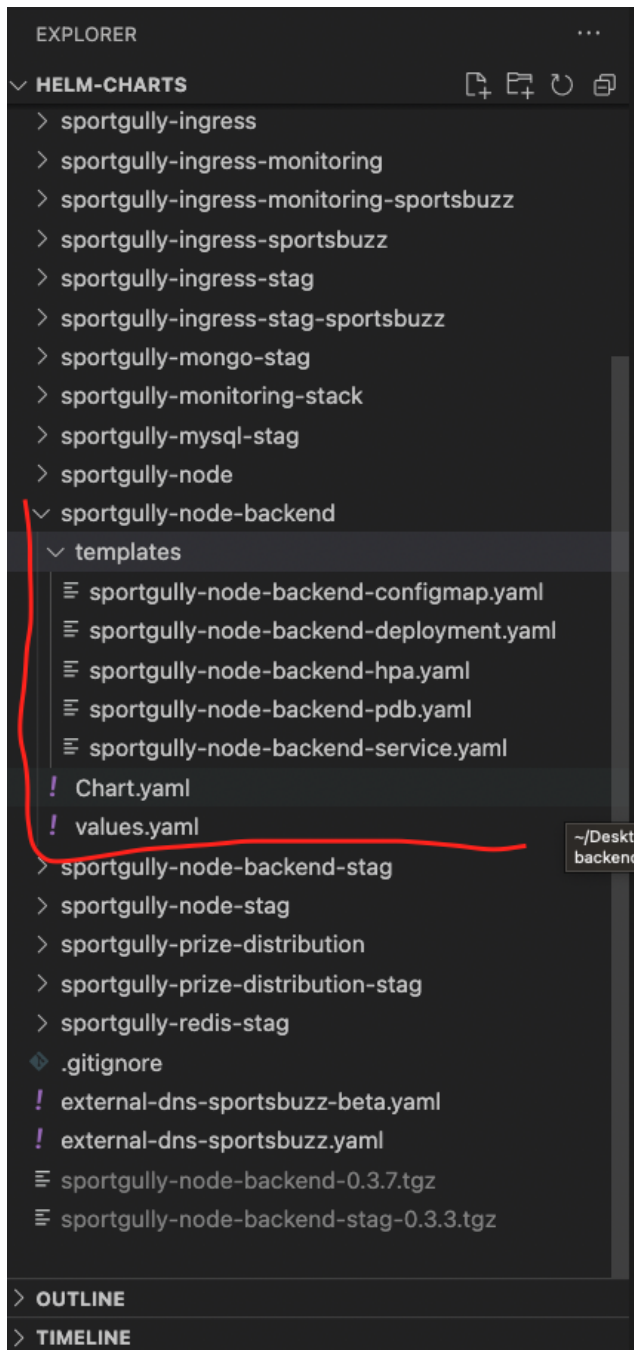
Open it in your favorite editor, it will look something like this

Note: Image used in this doc are for reference only



For every microservice, separate helm charts are created, so if you want to edit any value in the chart for a particular service, open that service's helm chart and edit it. For e.g. let's say there's a need to change the HPA value in the node backend,

1. Open the node backend chart



2. Now head over to **values.yaml** and edit the autoscaling section, whichever value needs to be changed.

```
autoscaling:
  enabled: true
  minReplicas: "15"
  maxReplicas: "25"
  targetCPU: "60"
  # targetMemory: ""
```

3. First, check the chart name using the command
helm ls -n production (instead of production change as per the environment)
4. This repo is managed using git so you can use all git commands, and do a git pull every time before making any change in the files.
5. **Before upgrading, it is very much important to change the cluster in which you want to update.**
6. Now upgrade it in the cluster using a command, which is as follows

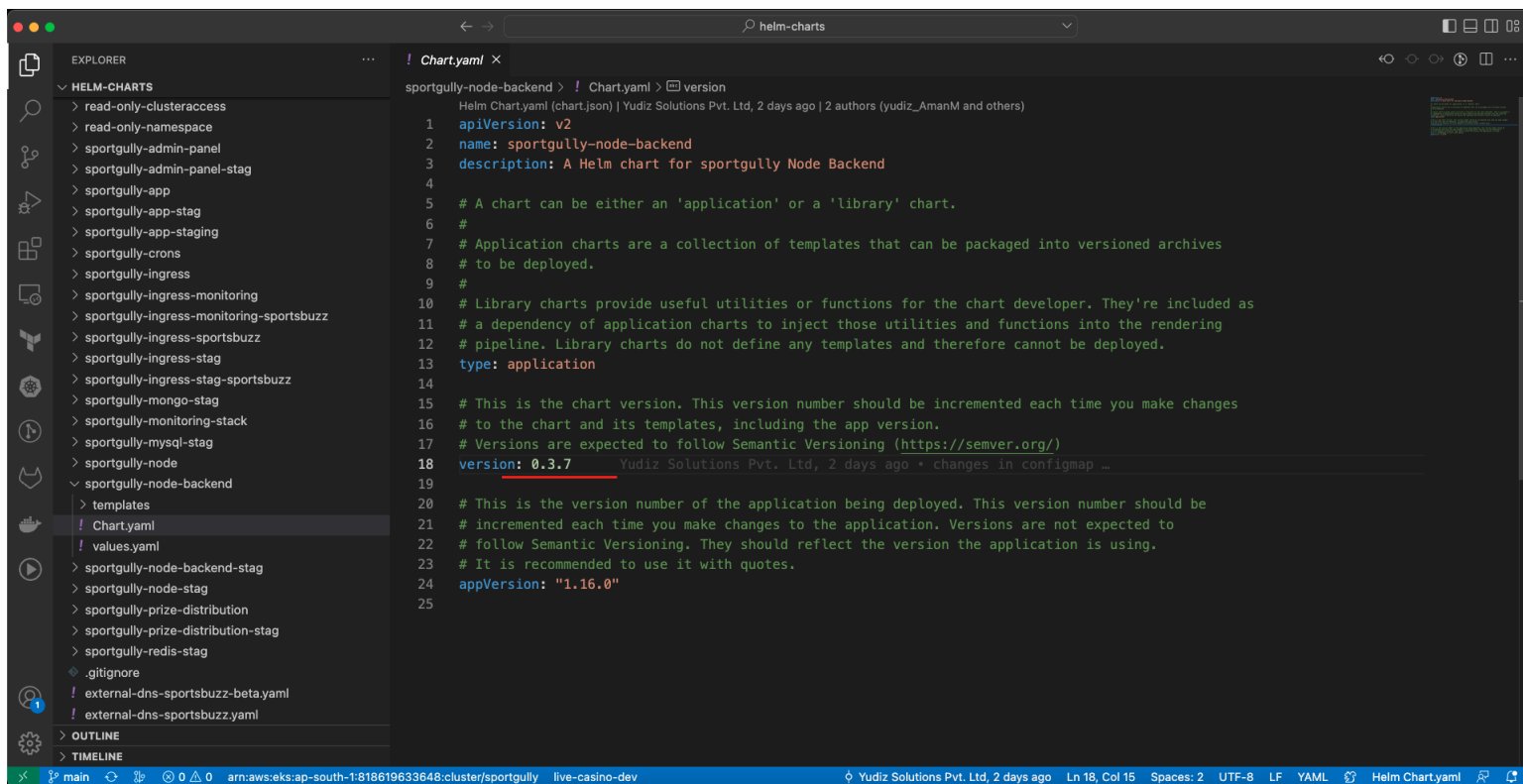
```
helm upgrade -n production sportgully-node-backend sportgully-node-backend
```

namespace

Release name

Chart name

7. Now update the chart number first by editing the chart version in the file



```
! Chart.yaml X
sportgully-node-backend > ! Chart.yaml > version
Helm Chart.yaml (chart.json) | Yudiz Solutions Pvt. Ltd, 2 days ago | 2 authors (yudiz_AmanM and others)
1  apiVersion: v2
2  name: sportgully-node-backend
3  description: A Helm chart for sportgully Node Backend
4
5  # A chart can be either an 'application' or a 'library' chart.
6  #
7  # Application charts are a collection of templates that can be packaged into versioned archives
8  # to be deployed.
9  #
10 # Library charts provide useful utilities or functions for the chart developer. They're included as
11 # a dependency of application charts to inject those utilities and functions into the rendering
12 # pipeline. Library charts do not define any templates and therefore cannot be deployed.
13 type: application
14
15 # This is the chart version. This version number should be incremented each time you make changes
16 # to the chart and its templates, including the app version.
17 # Versions are expected to follow Semantic Versioning (https://semver.org/)
18 version: 0.3.7      Yudiz Solutions Pvt. Ltd, 2 days ago • changes in configmap ...
19
20 # This is the version number of the application being deployed. This version number should be
21 # incremented each time you make changes to the application. Versions are not expected to
22 # follow Semantic Versioning. They should reflect the version the application is using.
23 # It is recommended to use it with quotes.
24 appVersion: "1.16.0"
25
```

8. Now make a package of the chart which was updated, in order to reflect the changes whenever CI/CD is triggered. To package a chart using this command (**P.S. these all are compulsory steps**)

helm package <chart-name>(here fantasy-node-backend)

9. Install the cm-push plugin from [here](#)

10. Login into the helm-repo using this command

```
helm repo add \  
--username nisarg.satani \  
--password glpat-Prpng987UuWqeW9xsQAH \  
Sportsbuzz11 \  
https://gitlab.com/api/v4/projects/40317289/packages/helm/stabl  
e
```

11. After that push the chart to the package registry using this command

```
helm cm-push fantasy-node-backend(change the name as  
required) Sportsbuzz11
```