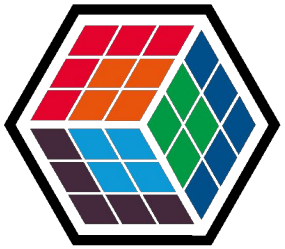


THE DEVELOPER'S CONFERENCE

Deployando Hyperledger Fabric com Kubernetes

Cláudio Ramos

Cláudio Ramos
Senior Developer, SAP

Graduado em Ciências da Computação - UNISINOS

Treinamentos sobre Blockchain as a Service em SAP Cloud Platform
Avaliação de viabilidade de projetos referentes a blockchains permissionadas
Membro da Hyperledger Community POA
Contribuidor no projeto Hyperledger Fabric da Linux Foundation

https://event.on24.com/wcc/r/1819160/1D0ED8AC50AC8CB6E0158C1BE45B25E7

Três passos para ir LIVE com HLF & K8S

1 Hyperledger Fabric

Projeto Hyperledger Fabric e como seus componentes interagem

2 Kubernetes

Framework Kubernetes, Helm Charts e building blocks básicos

3 Deployando Hyperledger Fabric com Kubernetes

Passo-a-passo para deploy de uma rede usando Helm charts para CA, Orderer e Peers

4 Próximo passos

Melhorias futuras e como contribuir

HFL direcionado a KS8

Fabric CA

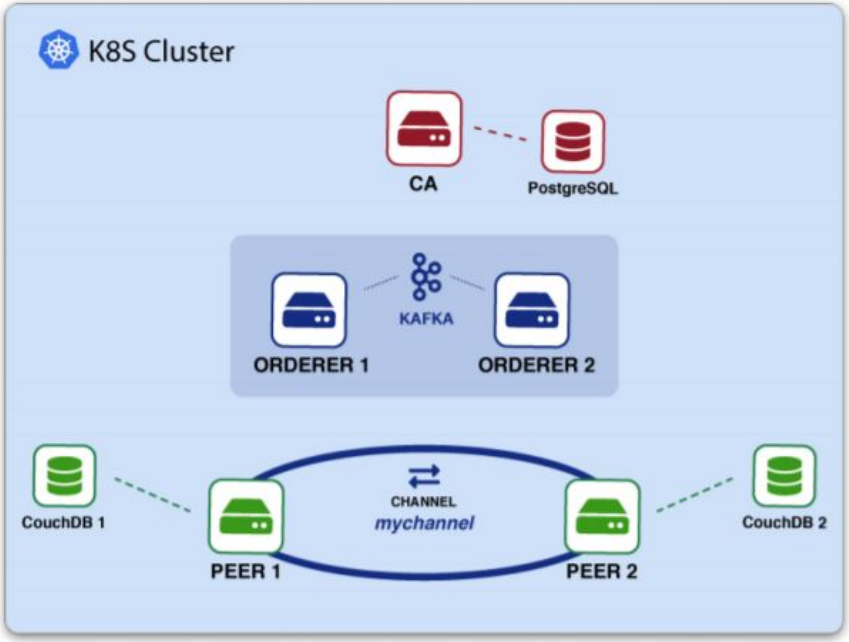
Fabric Certificate Authority
registro & definição de identidades

Fabric Orderer

Serviço Fabric Ordering provém consenso em redes para desenvolvimento (solo) e produção (Kafka).

Fabric Peer

Fabric Peer gerencia a blockchain através da comunicação com o serviço de ordenação (ordering)



THE
DEVELOPER'S
CONFERENCE

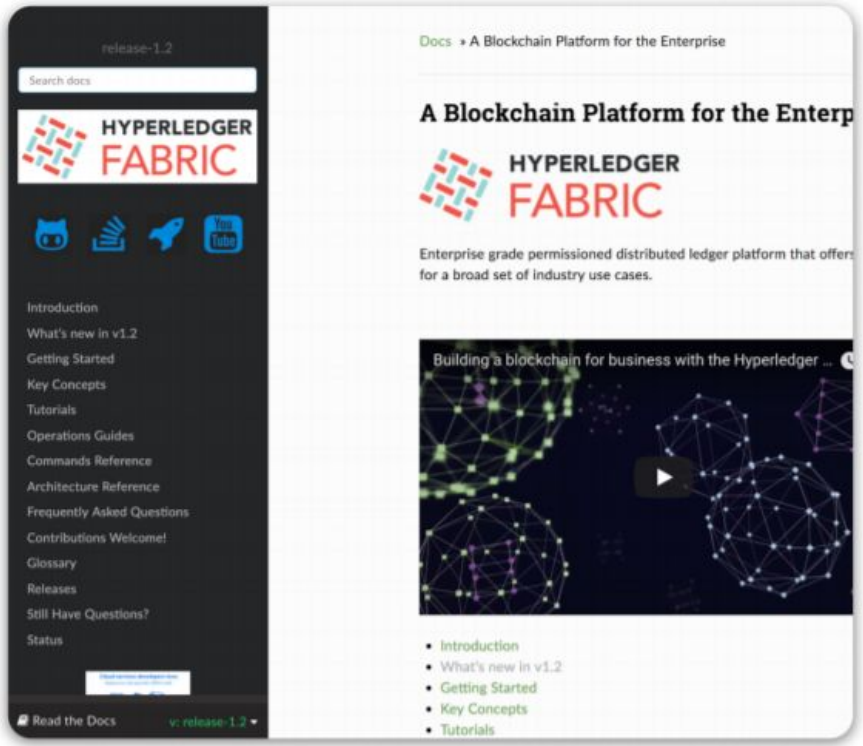
Hyperledger Fabric

Hyperledger Fabric

Fabric é um dos frameworks do projeto Hyperledger.

Permite a construção de redes de blockchain permissionadas.

https://hyperledger-fabric.readthedocs.io/



Hyperledger Fabric precisa de 3 tipos de nodo para ser deployado em produção

Fabric **Certificate Authority** para registro e definição de identidades

Fabric **Ordering Service** provendo consenso para redes solo e KAFKA

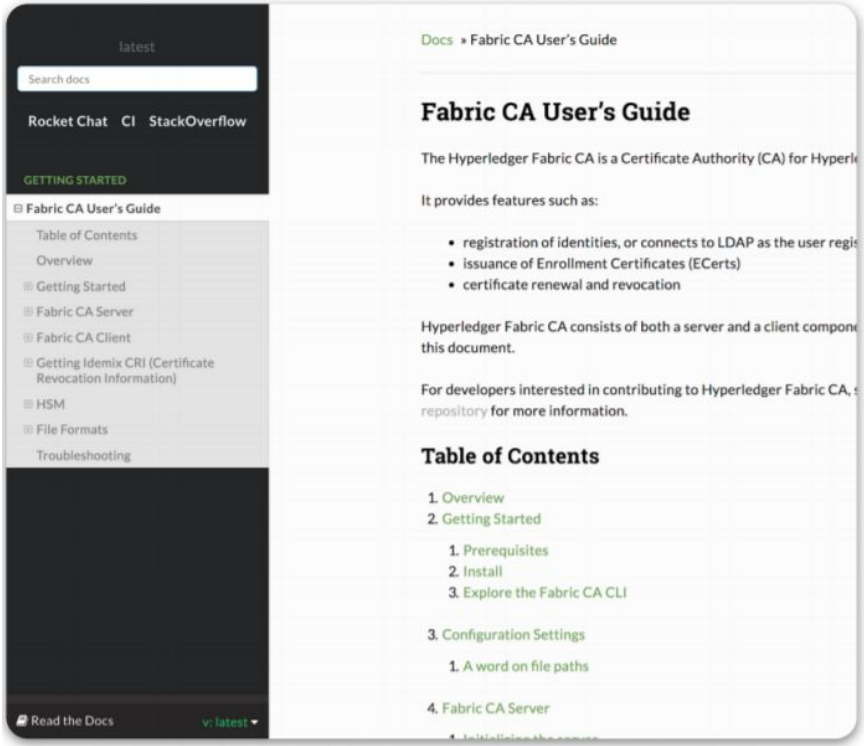
Fabric **Peer** para gerenciar a blockchain pela comunicação com Ordering Service

Certificate Authority

Fabric CA gerencia identidades, uma vez que:

- Registra identidades
- Emite ECerts
- Renova e revoga certificados

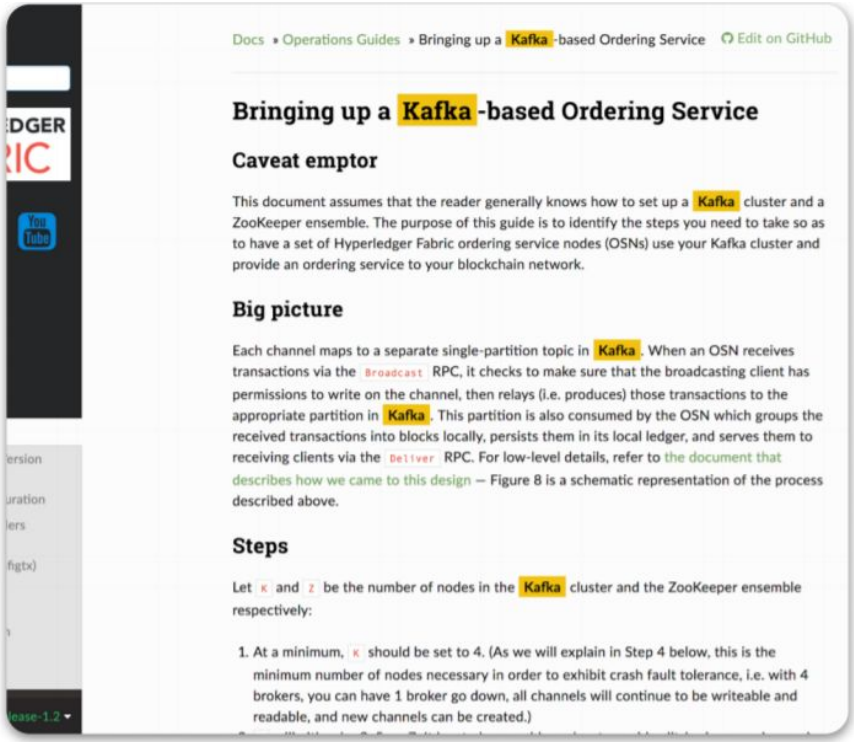
https://hyperledger-fabric-ca.readthedocs.io/en/latest/users-guide.html



Ordering Service

Ordering Service consiste em um grupo de nodos Orderer que estabelecem **consenso** através da troca de mensagens via cluster KAFKA

https://hyperledger-fabric.readthedocs.io/en/release-1.2/kafka.html

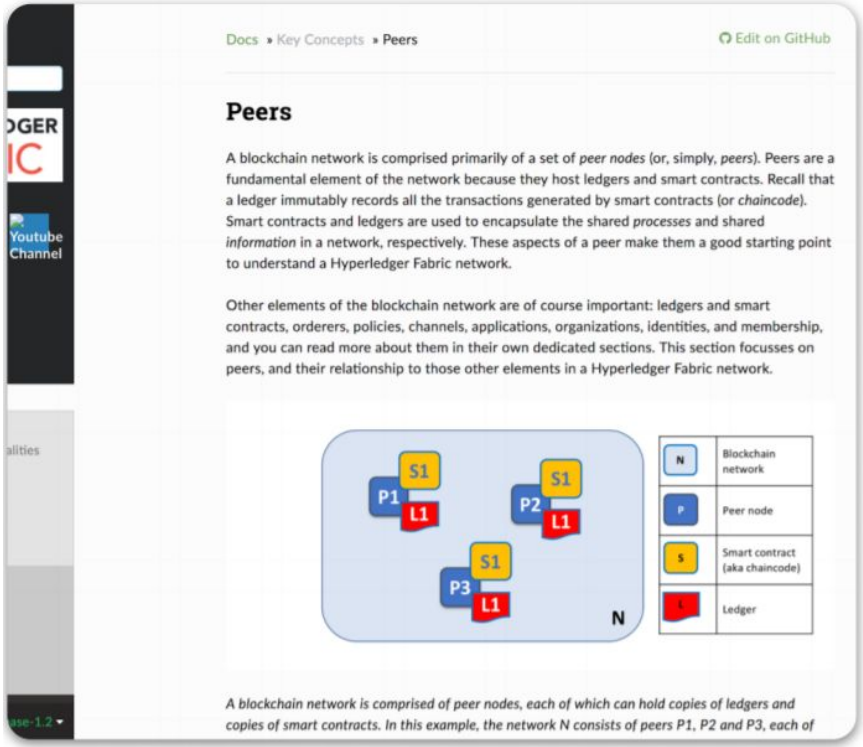


Peer

Peers são os únicos nodos que realmente gravam a blockchain ledger.

Podemos anexar CouchDB para manter e fazer pesquisas sobre o **estado geral** da blockchain

https://hyperledger-fabric.readthedocs.io/en/release-1.2/peers/peers.html



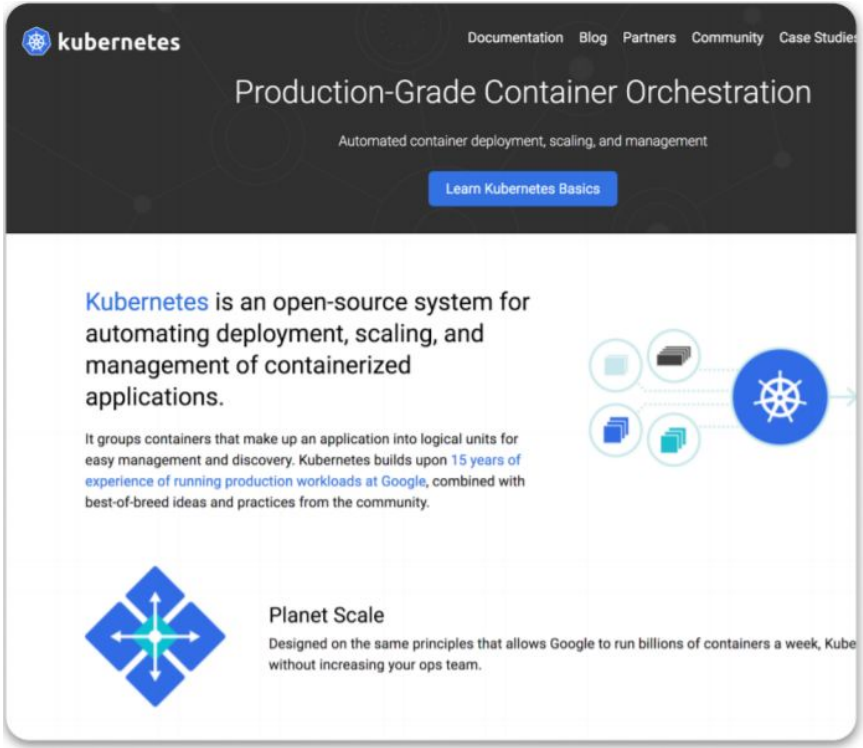
THE
DEVELOPER'S
CONFERENCE

Kubernetes

Kubernetes

Kubernetes é um sistema para orquestrar o deploy e gerenciamento de conteiners (e. g. Docker)

https://kubernetes.io/



Para trabalhar com **Kubernetes,** precisamos entender 3 conceitos principais

O Kubernetes **Cluster** gerencia conjunto de abstrações que tornam fácil orquestrar containers

Charts são pacotes junto a templates de Kubernetes, atuando como facilitadores

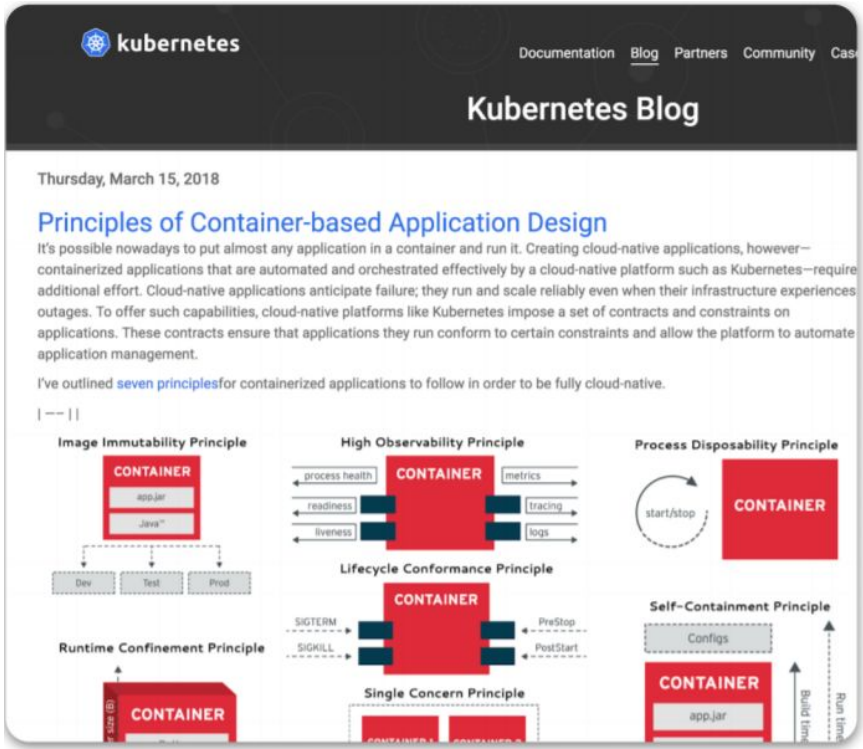
Helm (e seu modo server-side Tiller) permitem a instalação de Charts

Kubernetes cluster

Kubernetes (K8S) nos permite agrupar, fazer deploy, escalar, realizar auto-ajuste e conectar a containers

K8S orquestra containers

https://kubernetes.io/blog/2018/03/principles-of-container-app-design/

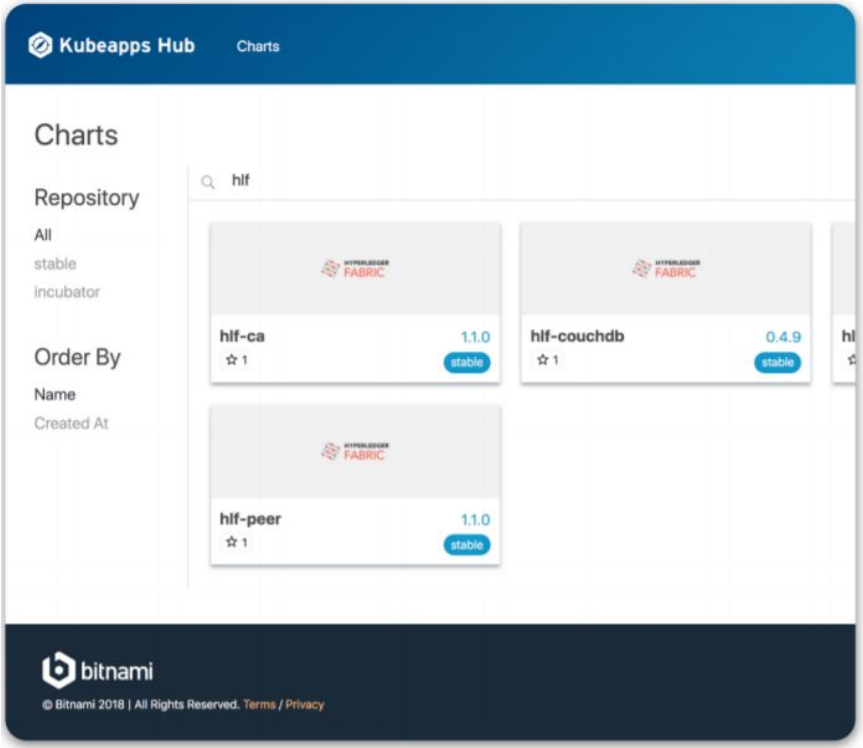


Charts

Charts são pacotes de templates que definem componentes de Kubernetes

É possível buscar por Charts no Kubeapps Hub

https://hub.kubeapps.com/charts?q=hlf

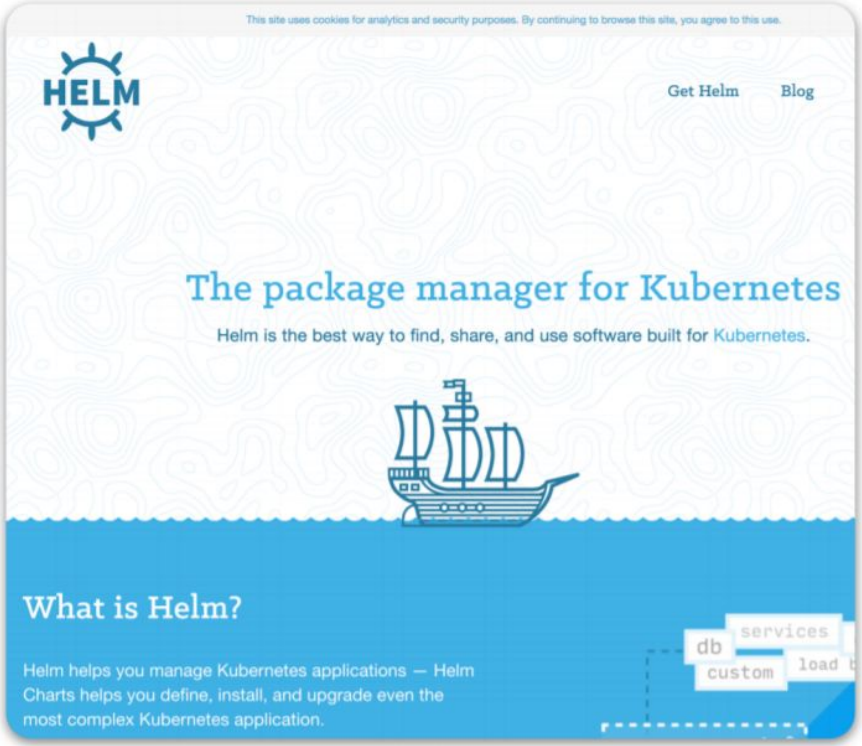


Helm

Helm permite o empacotamento, pesquisa e deploy de Charts

Seu componente server-side, Tiller, gerencia o deployment em K8S

https://www.helm.sh/



Antes de começar, são necessárias 3 coisas

Setup do **Kubernetes** cluster e instalação do Helm/Tiller nele

Obtenção de um **domain name** (gratuito / baratinho) para seu cluster

Download do **repositório** no endereço **https://github.com/aidtechnology/lf-k8s-hlf-webinar**

THE DEVELOPER'S CONFERENCE

Instale Ingress controller e cert manager

helm install stable/nginx-ingress -n nginx-ingress --namespace ingress-controller

helm install stable/cert-manager -n cert-manager --namespace cert-manager

kubectl create -f *./extra/certManagerCI_staging.yaml*

kubectl create -f *./extra/certManagerCI_production.yaml*

THE DEVELOPER'S CONFERENCE

Deployando HLF no K8S

Começando a festa, certo meow!

HFL direcionado a KS8

Fabric CA

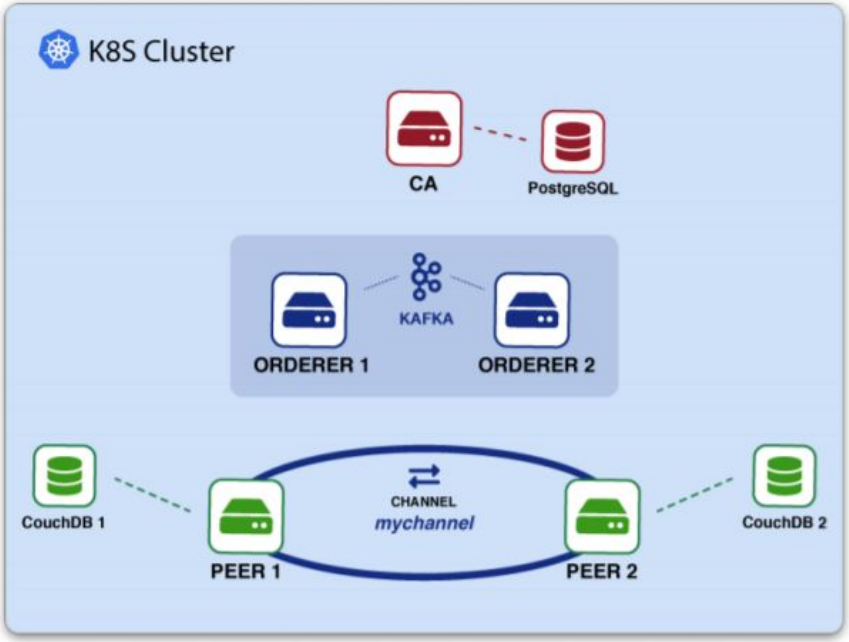
Fabric Certificate Authority
registro & definição de identidades

Fabric Orderer

Serviço Fabric Ordering provém consenso em redes para desenvolvimento (solo) e produção (Kafka).

Fabric Peer

Fabric Peer gerencia a blockchain através da comunicação com o serviço de ordenação (ordering)

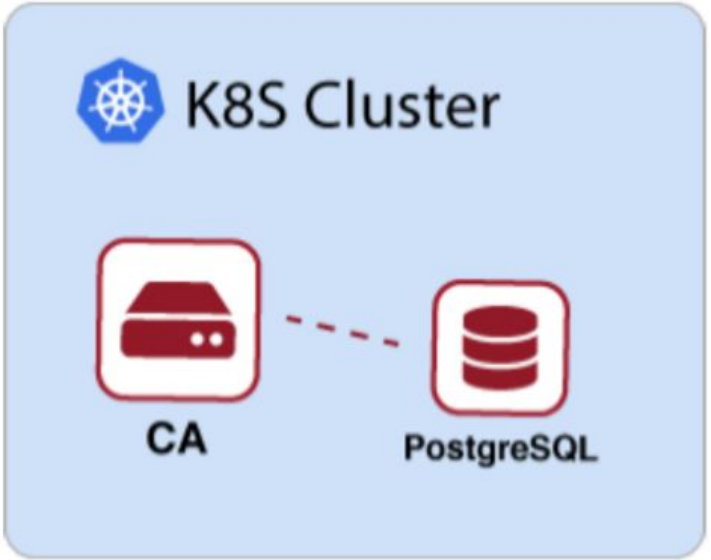


THE DEVELOPER'S CONFERENCE

Fabric CA

Fabric CA

- Instalar CA Database Helm Chart
- Instalar Fabric CA Helm Chart
- Gerar Fabric CA Identity
- Obter Crypto Material
- Salvar Crypto Material no K8S
- Gerar Genesis e Channel

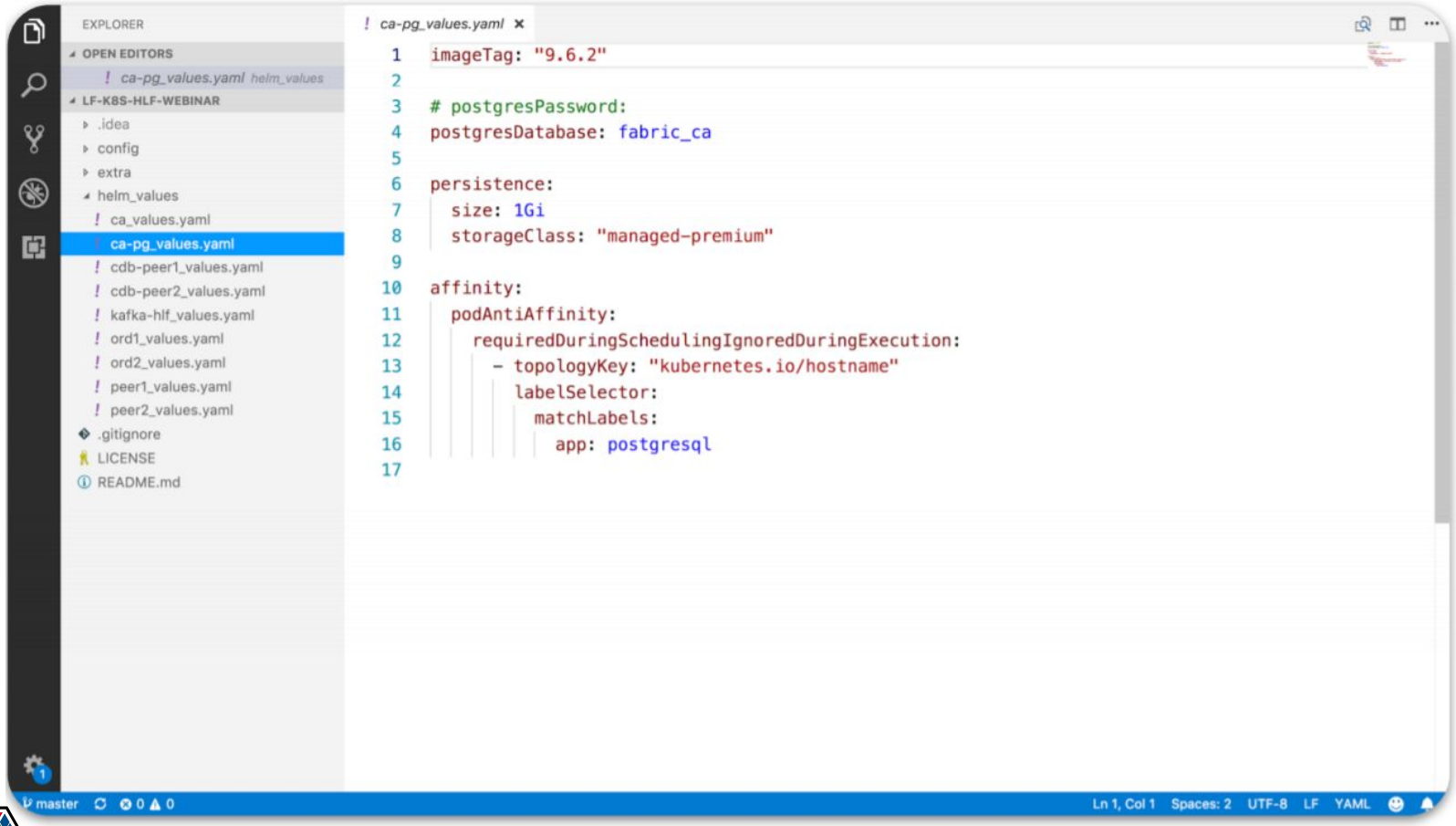


Instalar CA Database Helm Chart

Instalar CA Database Helm Chart

```
helm install stable/postgresql -n ca-pg --namespace
blockchain -f ./helm_values/ca-pg_values.yaml
```

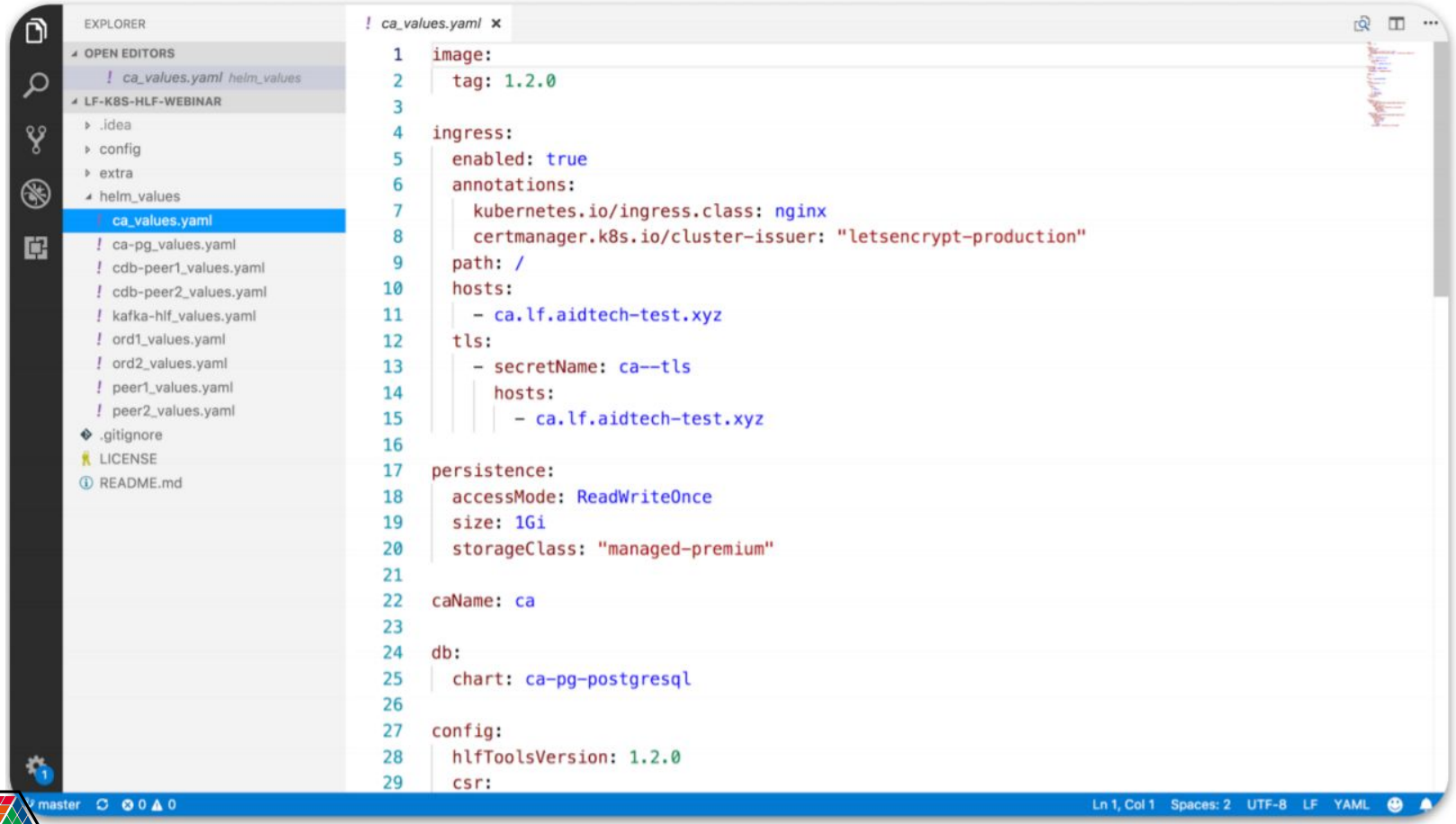
THE DEVELOPER'S CONFERENCE


EXPLORER
OPEN EDITORS
ca-pg_values.yaml helm_values
LF-K8S-HLF-WEBINAR
.idea
config
extra
helm_values
ca_values.yaml
ca-pg_values.yaml
cdb-peer1_values.yaml
cdb-peer2_values.yaml
kafka-hlf_values.yaml
ord1_values.yaml
ord2_values.yaml
peer1_values.yaml
peer2_values.yaml
.gitignore
LICENSE
README.md
ca-pg_values.yaml
1 imageTag: "9.6.2"
2
3 # postgresPassword:
4 postgresDatabase: fabric_ca
5
6 persistence:
7 size: 1Gi
8 storageClass: "managed-premium"
9
10 affinity:
11 podAntiAffinity:
12 requiredDuringSchedulingIgnoredDuringExecution:
13 - topologyKey: "kubernetes.io/hostname"
14 labelSelector:
15 matchLabels:
16 app: postgresql
17
master
0 0
Ln 1, Col 1
Spaces: 2
UTF-8
LF
YAML

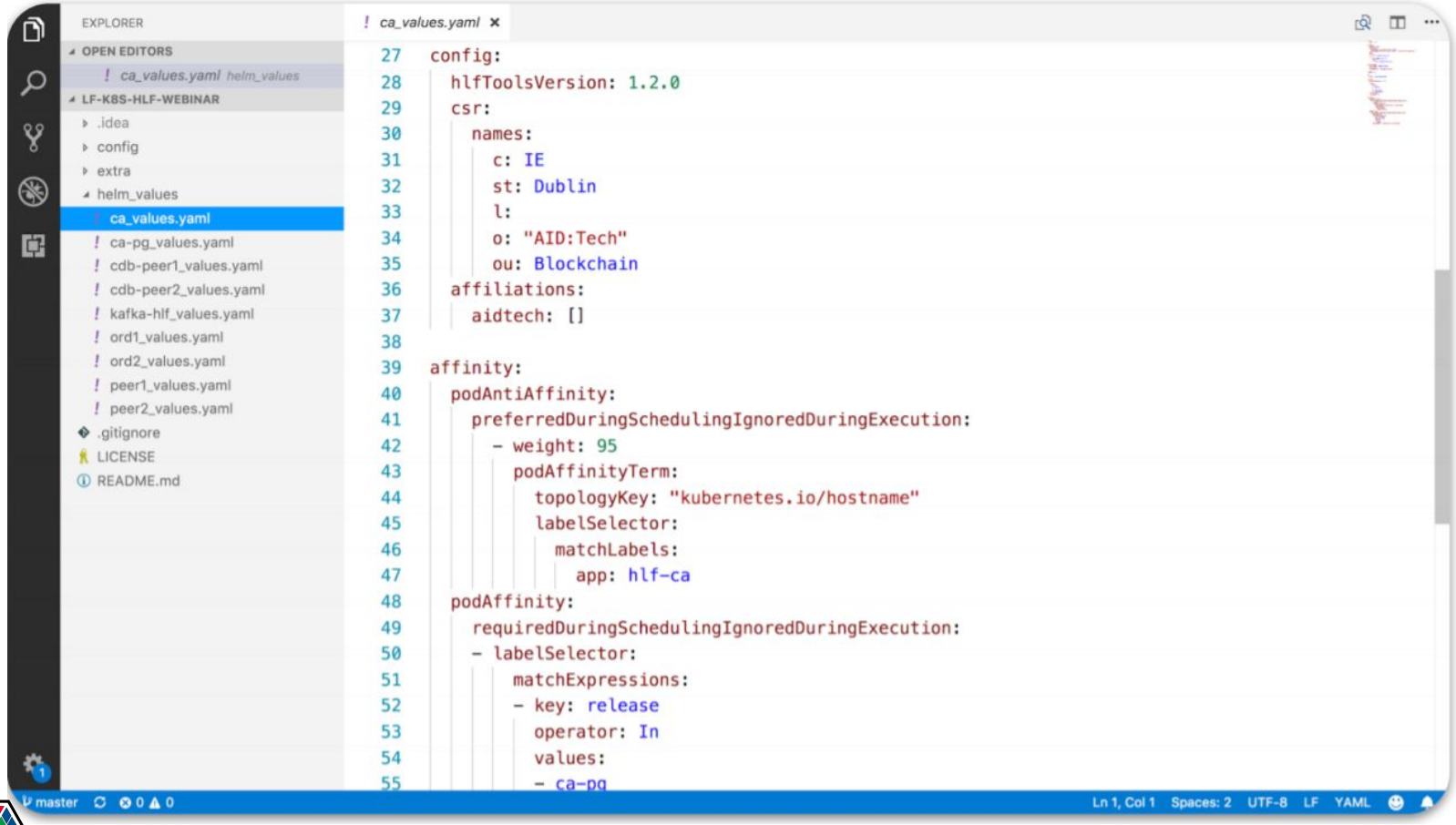
Instalar Cabric CA Helm Chart

- Instalar CA Database Helm Chart
- Instalar Fabric CA Helm Chart

```
helm install stable/hlf-ca -n ca --namespace blockchain -f
./helm_values/ca_values.yaml

CA_POD=$(kubectl get pods -n blockchain -l "app=hlf-
ca,release=ca" -o jsonpath="{.items[0].metadata.name}")

kubectl logs -n blockchain $CA_POD | grep 'Listening on'
```



THE DEVELOPER'S CONFERENCE


EXPLORER
ca_values.yaml
OPEN EDITORS
ca_values.yaml helm_values
LF-K8S-HLF-WEBINAR
.idea
config
extra
helm_values
ca_values.yaml
ca-pg_values.yaml
cdb-peer1_values.yaml
cdb-peer2_values.yaml
kafka-hlf_values.yaml
ord1_values.yaml
ord2_values.yaml
peer1_values.yaml
peer2_values.yaml
.gitignore
LICENSE
README.md
config:
hlfToolsVersion: 1.2.0
csr:
names:
c: IE
st: Dublin
l:
o: "AID:Tech"
ou: Blockchain
affiliations:
aidtech: []
affinity:
podAntiAffinity:
preferredDuringSchedulingIgnoredDuringExecution:
- weight: 95
podAffinityTerm:
topologyKey: "kubernetes.io/hostname"
labelSelector:
matchLabels:
app: hlf-ca
podAffinity:
requiredDuringSchedulingIgnoredDuringExecution:
- labelSelector:
matchExpressions:
- key: release
operator: In
values:
- ca-pg
master
Ln 1, Col 1
Spaces: 2
UTF-8
LF
YAML

Gerar Fabric CA Identity

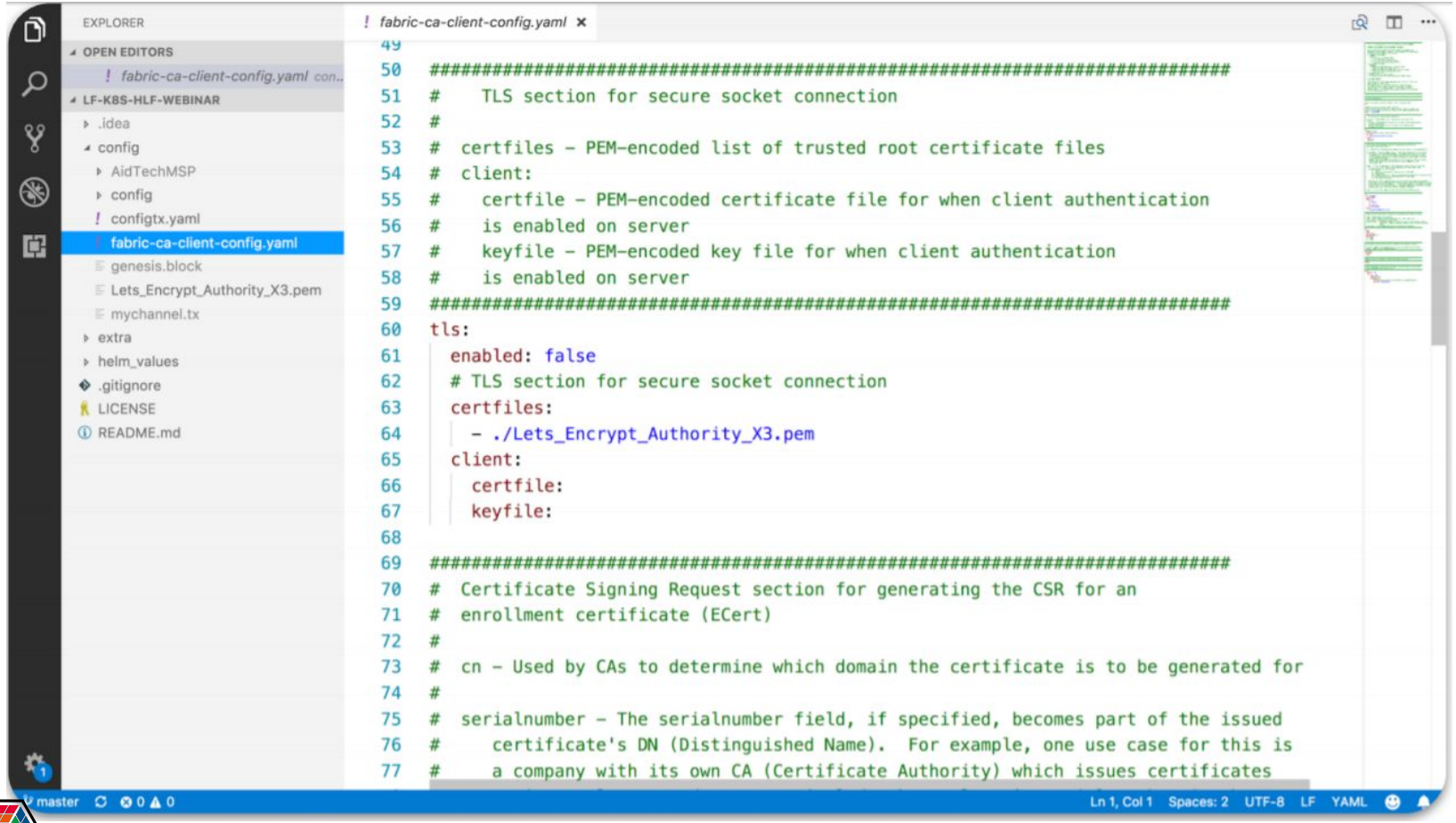
- Instalar CA Database Helm Chart
- Instalar Fabric CA Helm Chart
- Gerar Fabric CA Identity

```
kubectl exec -n blockchain $CA_POD -- cat
/var/hyperledger/fabric-ca/msp/signcerts/cert.pem

kubectl exec -n blockchain $CA_POD -- bash -c 'fabric-ca-client
enroll -d -u
http://$CA_ADMIN:$CA_PASSWORD@$SERVICE_DNS:7054'

CA_INGRESS=$(kubectl get ingress -n blockchain -l "app=hlf-
ca,release=ca" -o jsonpath="{.items[0].spec.rules[0].host}")

curl https://$CA_INGRESS/cainfo
```

Obter crypto material

- Instalar CA Database Helm Chart
- Instalar Fabric CA Helm Chart
- Gerar Fabric CA Identity
- Obter Crypto Material

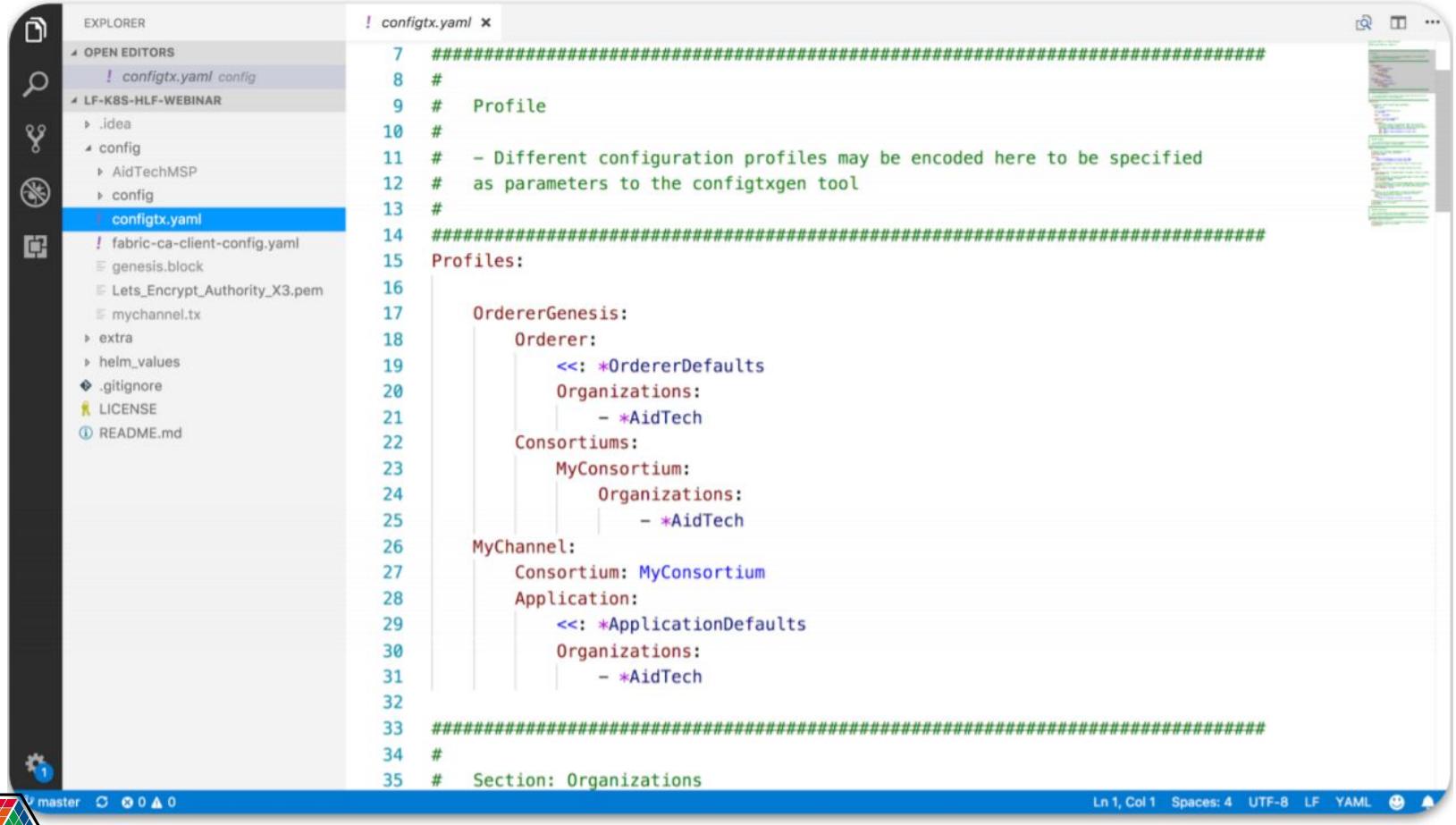
```
FABRIC_CA_CLIENT_HOME=./config fabric-ca-client getcacert -u
http://$CA_INGRESS -M ./AidTechMSP

kubectl exec -n blockchain $CA_POD -- fabric-ca-client register --
id.name org-admin --id.secret OrgAdm1nPW --id.attrs 'admin=true:ecert'

FABRIC_CA_CLIENT_HOME=./config fabric-ca-client enroll -u
http://org-admin:OrgAdm1nPW@$CA_INGRESS -M ./AidTechMSP

mkdir -p ./config/AidTechMSP/admincerts

cp ./config/AidTechMSP/signcerts/* ./config/AidTechMSP/admincerts
```

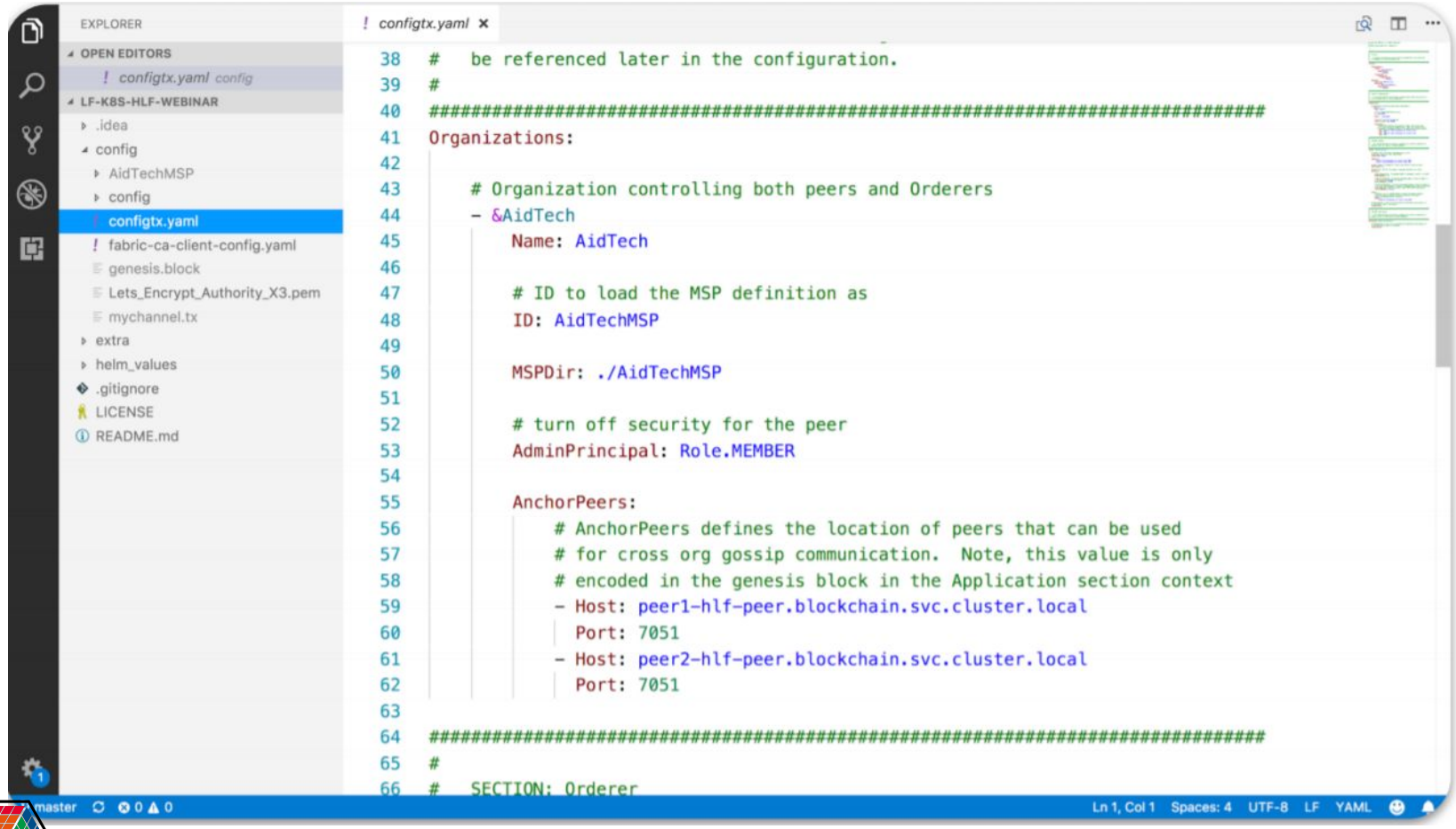

EXPLORER
fabric-ca-client-config.yaml
OPEN EDITORS
fabric-ca-client-config.yaml
LF-K8S-HLF-WEBINAR
.idea
config
AidTechMSP
config
configtx.yaml
fabric-ca-client-config.yaml
genesis.block
Lets_Encrypt_Authority_X3.pem
mychannel.tx
extra
helm_values
.gitignore
LICENSE
README.md
49
50 ##
51 # TLS section for secure socket connection
52 #
53 # certfiles - PEM-encoded list of trusted root certificate files
54 # client:
55 # certfile - PEM-encoded certificate file for when client authentication
56 # is enabled on server
57 # keyfile - PEM-encoded key file for when client authentication
58 # is enabled on server
59 ##
60 tls:
61 enabled: false
62 # TLS section for secure socket connection
63 certfiles:
64 - ./Lets_Encrypt_Authority_X3.pem
65 client:
66 certfile:
67 keyfile:
68
69 ##
70 # Certificate Signing Request section for generating the CSR for an
71 # enrollment certificate (ECert)
72 #
73 # cn - Used by CAs to determine which domain the certificate is to be generated for
74 #
75 # serialnumber - The serialnumber field, if specified, becomes part of the issued
76 # certificate's DN (Distinguished Name). For example, one use case for this is
77 # a company with its own CA (Certificate Authority) which issues certificates
master
0 0
Ln 1, Col 1
Spaces: 2
UTF-8
LF
YAML

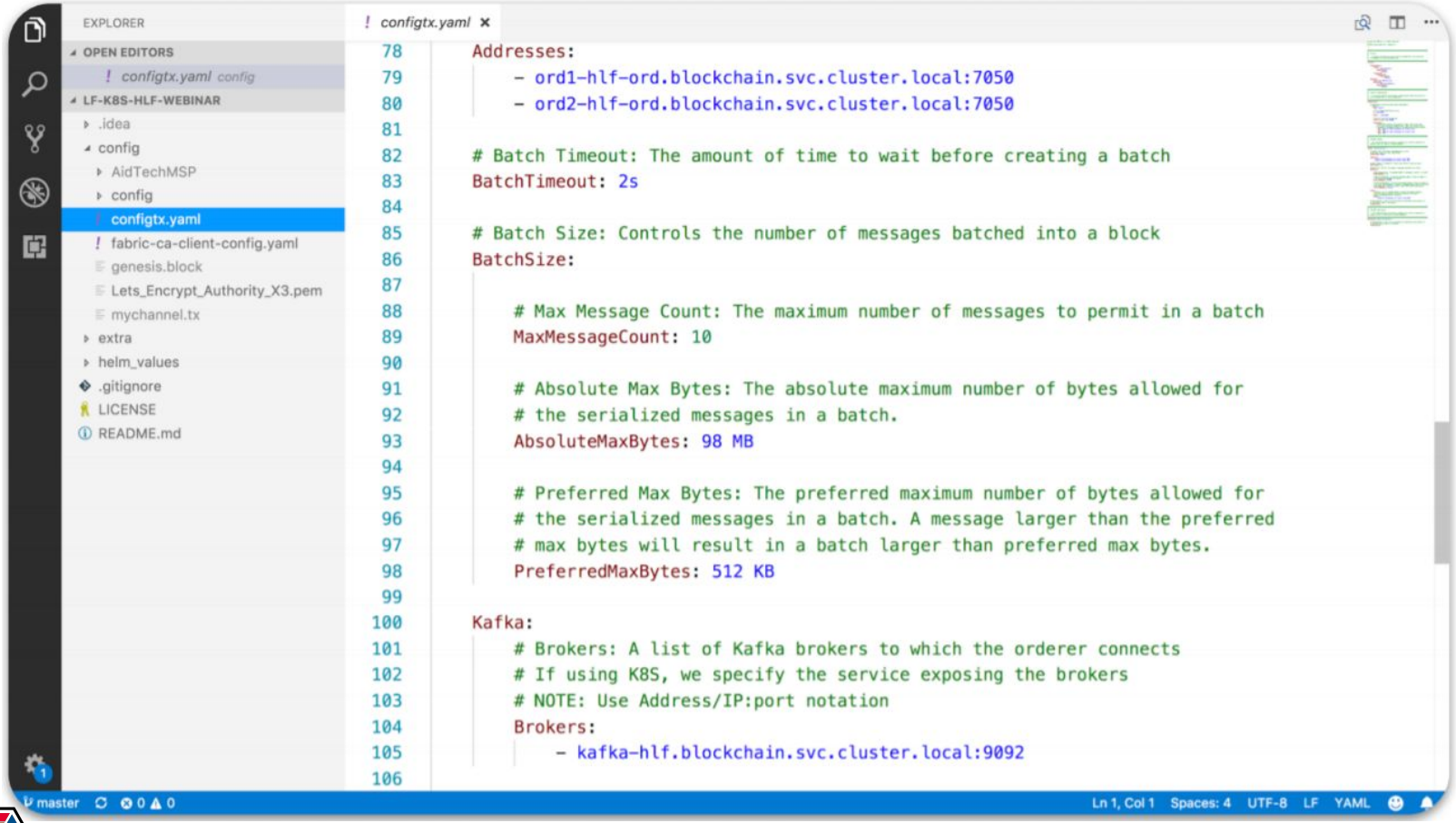
Salvar crypto material no K8S

- Instalar CA Database Helm Chart
- Instalar Fabric CA Helm Chart
- Gerar Fabric CA Identity
- Obter Crypto Material
- Salvar Crypto Material no K8S

```
ORG_CERT=$(ls ./config/AidTechMSP/admincerts/cert.pem)

kubectl create secret generic -n blockchain hlf--org-admincert
--from-file=cert.pem=$ORG_CERT

ORG_KEY=$(ls ./config/AidTechMSP/keystore/*_sk)

kubectl create secret generic -n blockchain hlf--org-adminkey
--from-file=key.pem=$ORG_KEY
```


EXPLORER
OPEN EDITORS
configtx.yaml config
LF-K8S-HLF-WEBINAR
.idea
config
AidTechMSP
config
configtx.yaml
fabric-ca-client-config.yaml
genesis.block
Lets_Encrypt_Authority_X3.pem
mychannel.tx
extra
helm_values
.gitignore
LICENSE
README.md
configtx.yaml
7 ##
8 #
9 # Profile
10 #
11 # - Different configuration profiles may be encoded here to be specified
12 # as parameters to the configtxgen tool
13 #
14 ##
15 Profiles:
16
17 OrdererGenesis:
18 Orderer:
19 <<: *OrdererDefaults
20 Organizations:
21 - *AidTech
22 Consortiums:
23 MyConsortium:
24 Organizations:
25 - *AidTech
26 MyChannel:
27 Consortium: MyConsortium
28 Application:
29 <<: *ApplicationDefaults
30 Organizations:
31 - *AidTech
32
33 ##
34 #
35 # Section: Organizations
master
Ln 1, Col 1
Spaces: 4
UTF-8
LF
YAML


EXPLORER
configtx.yaml
OPEN EDITORS
configtx.yaml config
LF-K8S-HLF-WEBINAR
.idea
config
AidTechMSP
config
configtx.yaml
fabric-ca-client-config.yaml
genesis.block
Lets_Encrypt_Authority_X3.pem
mychannel.tx
extra
helm_values
.gitignore
LICENSE
README.md
38 # be referenced later in the configuration.
39 #
40 ###
41 Organizations:
42
43 # Organization controlling both peers and Orderers
44 - &AidTech
45 Name: AidTech
46
47 # ID to load the MSP definition as
48 ID: AidTechMSP
49
50 MSPDir: ./AidTechMSP
51
52 # turn off security for the peer
53 AdminPrincipal: Role.MEMBER
54
55 AnchorPeers:
56 # AnchorPeers defines the location of peers that can be used
57 # for cross org gossip communication. Note, this value is only
58 # encoded in the genesis block in the Application section context
59 - Host: peer1-hlf-peer.blockchain.svc.cluster.local
60 Port: 7051
61 - Host: peer2-hlf-peer.blockchain.svc.cluster.local
62 Port: 7051
63
64 ###
65 #
66 # SECTION: Orderer
master
0 0
Ln 1, Col 1
Spaces: 4
UTF-8
LF
YAML


EXPLORER
configtx.yaml
OPEN EDITORS
configtx.yaml config
LF-K8S-HLF-WEBINAR
.idea
config
AidTechMSP
config
configtx.yaml
fabric-ca-client-config.yaml
genesis.block
Lets_Encrypt_Authority_X3.pem
mychannel.tx
extra
helm_values
.gitignore
LICENSE
README.md
Addresses:
- ord1-hlf-ord.blockchain.svc.cluster.local:7050
- ord2-hlf-ord.blockchain.svc.cluster.local:7050
Batch Timeout: The amount of time to wait before creating a batch
BatchTimeout: 2s
Batch Size: Controls the number of messages batched into a block
BatchSize:
Max Message Count: The maximum number of messages to permit in a batch
MaxMessageCount: 10
Absolute Max Bytes: The absolute maximum number of bytes allowed for
the serialized messages in a batch.
AbsoluteMaxBytes: 98 MB
Preferred Max Bytes: The preferred maximum number of bytes allowed for
the serialized messages in a batch. A message larger than the preferred
max bytes will result in a batch larger than preferred max bytes.
PreferredMaxBytes: 512 KB
Kafka:
Brokers: A list of Kafka brokers to which the orderer connects
If using K8S, we specify the service exposing the brokers
NOTE: Use Address/IP:port notation
Brokers:
- kafka-hlf.blockchain.svc.cluster.local:9092
master
Ln 1, Col 1
Spaces: 4
UTF-8
LF
YAML

Gerar genesis & channel

- Instalar CA Database Helm Chart
- Instalar Fabric CA Helm Chart
- Gerar Fabric CA Identity
- Obter Crypto Material
- Salvar Crypto Material no K8S
- Gerar Genesis e Channel

cd ./config

configtxgen -profile OrdererGenesis -outputBlock *./genesis.block*

configtxgen -profile MyChannel -channelID mychannel -outputCreateChannelTx *./mychannel.tx*

kubectl create secret generic -n blockchain hlf--genesis --from-file=*genesis.block*

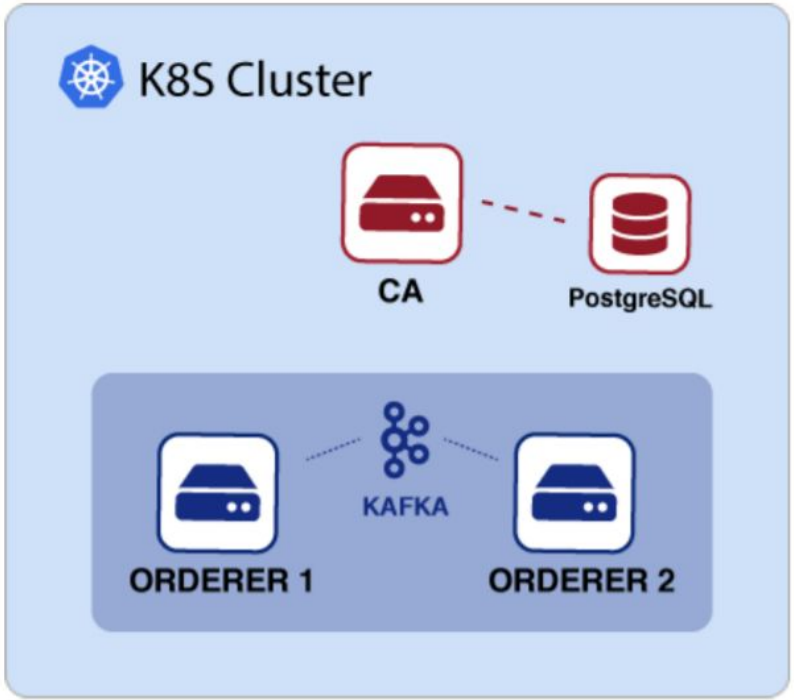
kubectl create secret generic -n blockchain hlf--channel --from-file=*mychannel.tx*

THE DEVELOPER'S CONFERENCE

Fabric Orderer

Fabric Orderer

- Instalar Kafka Helm Chart
- Instalar Fabric Orderer Helm Chart
- Definir Fabric Orderer Identity

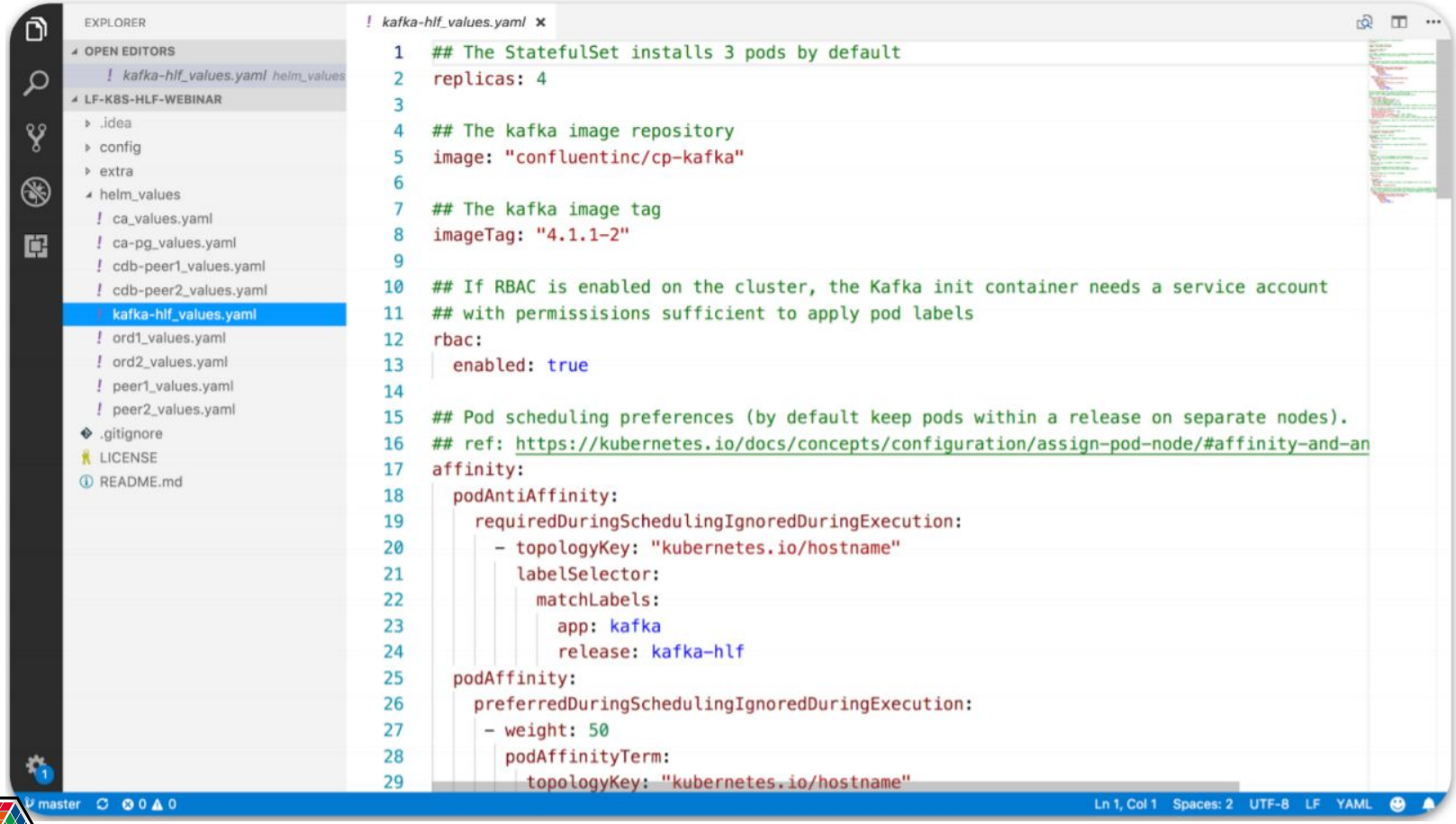


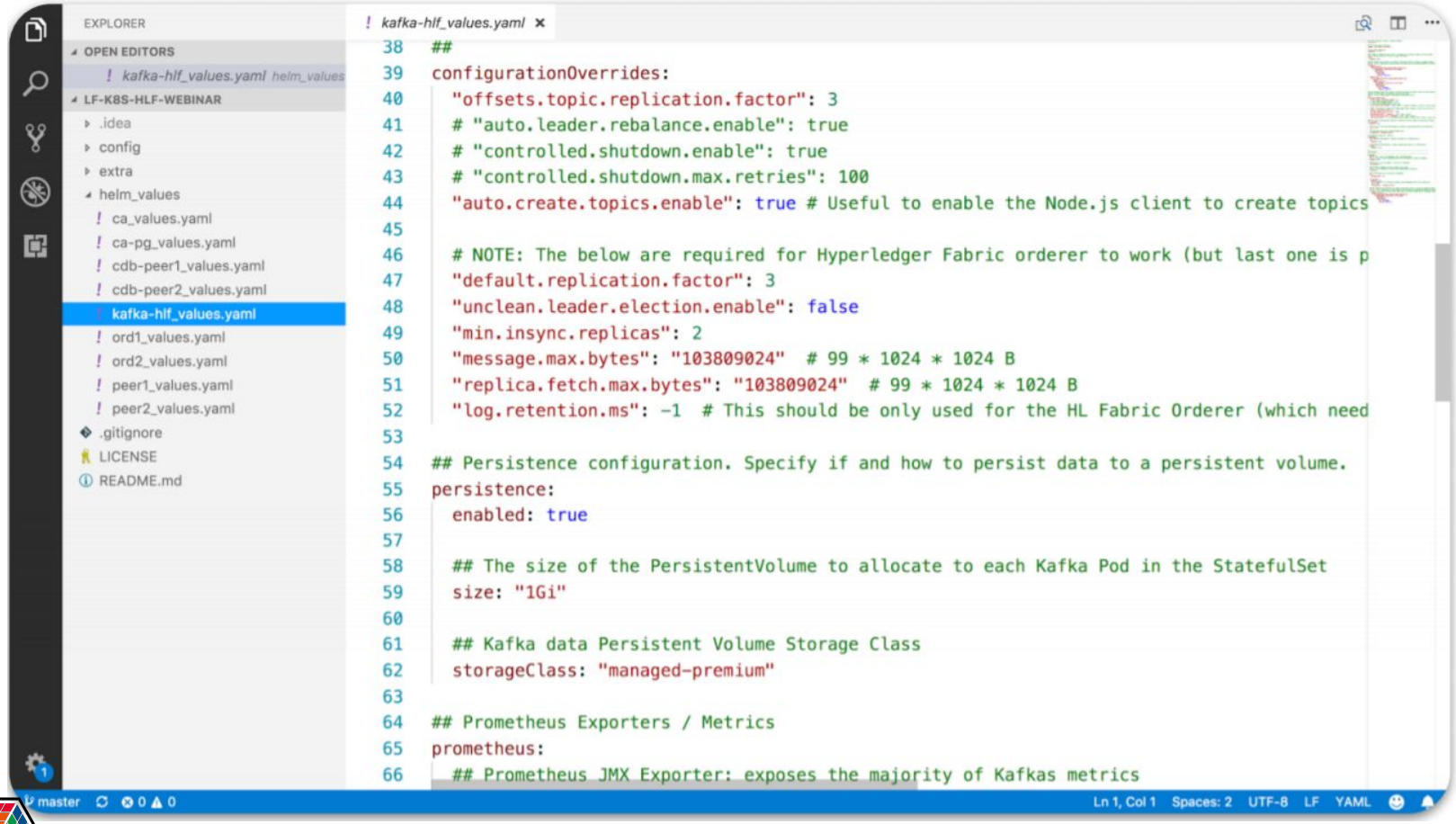
Instalar Kafka Helm Chart

Instalar Kafka Helm Chart

```
helm install incubator/kafka -n kafka-hlf --
namespace blockchain -f ./helm_values/kafka-
hlf_values.yaml
```

THE DEVELOPER'S CONFERENCE


EXPLORER
OPEN EDITORS
kafka-hlf_values.yaml helm_values
LF-K8S-HLF-WEBINAR
.idea
config
extra
helm_values
ca_values.yaml
ca-pg_values.yaml
cdb-peer1_values.yaml
cdb-peer2_values.yaml
kafka-hlf_values.yaml
ord1_values.yaml
ord2_values.yaml
peer1_values.yaml
peer2_values.yaml
.gitignore
LICENSE
README.md
kafka-hlf_values.yaml
The StatefulSet installs 3 pods by default
replicas: 4
The kafka image repository
image: "confluentinc/cp-kafka"
The kafka image tag
imageTag: "4.1.1-2"
If RBAC is enabled on the cluster, the Kafka init container needs a service account
with permissisions sufficient to apply pod labels
rbac:
enabled: true
Pod scheduling preferences (by default keep pods within a release on separate nodes).
ref: https://kubernetes.io/docs/concepts/configuration/assign-pod-node/#affinity-and-an
affinity:
podAntiAffinity:
requiredDuringSchedulingIgnoredDuringExecution:
- topologyKey: "kubernetes.io/hostname"
labelSelector:
matchLabels:
app: kafka
release: kafka-hlf
podAffinity:
preferredDuringSchedulingIgnoredDuringExecution:
- weight: 50
podAffinityTerm:
topologyKey: "kubernetes.io/hostname"
master
Ln 1, Col 1
Spaces: 2
UTF-8
LF
YAML



THE DEVELOPER'S CONFERENCE

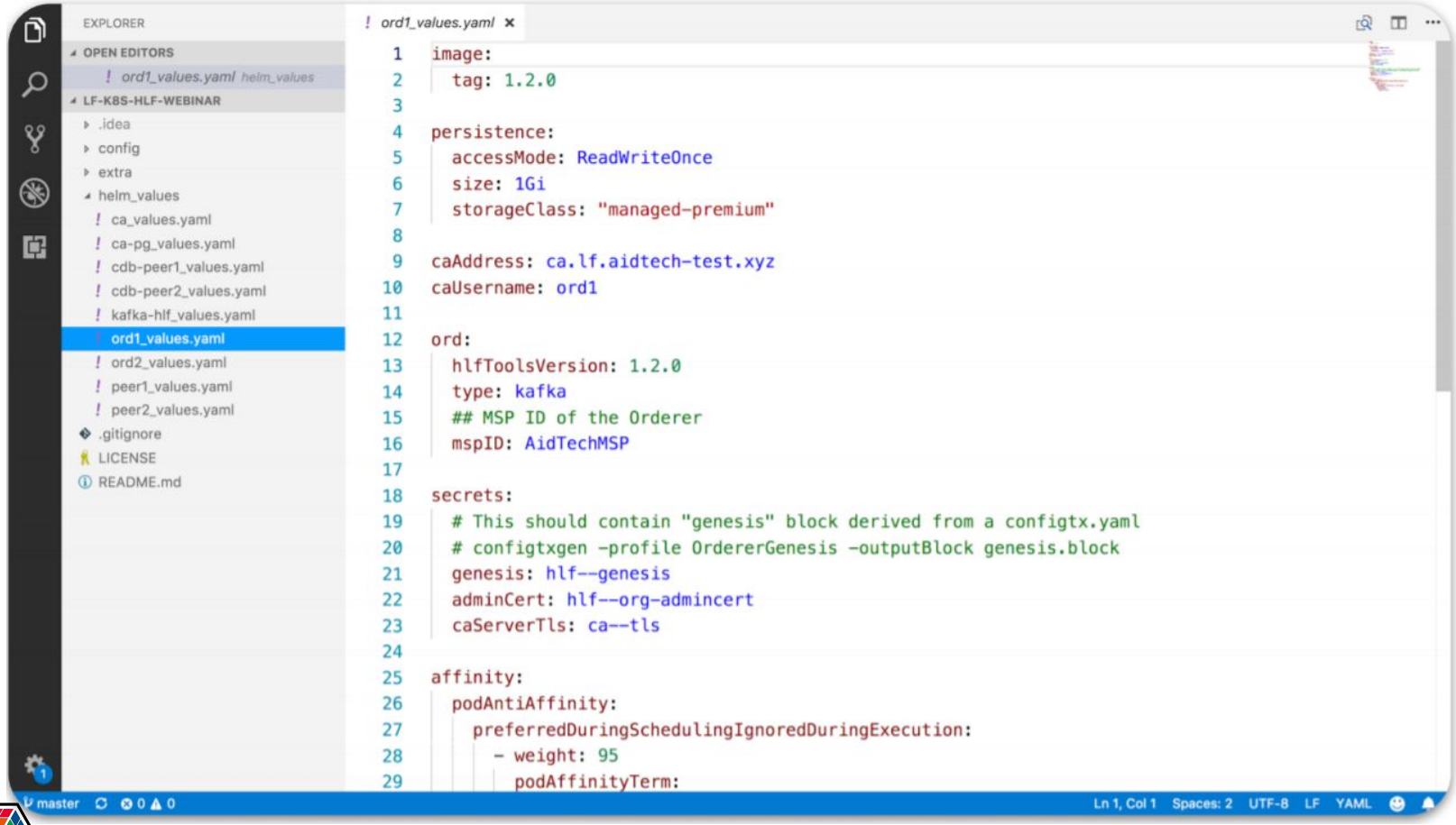
Instalar Fabric Orgerer Helm Chart

Instalar Kafka Helm Chart

Instalar Fabric Orderer Helm Chart

```
export NUM=1

helm install stable/hlf-ord -n ord${NUM} --
namespace blockchain -f
./helm_values/ord${NUM}_values.yaml
```

THE DEVELOPER'S CONFERENCE


EXPLORER
OPEN EDITORS
ord1_values.yaml helm_values
LF-K8S-HLF-WEBINAR
.idea
config
extra
helm_values
ca_values.yaml
ca-pg_values.yaml
cdb-peer1_values.yaml
cdb-peer2_values.yaml
kafka-hlf_values.yaml
ord1_values.yaml
ord2_values.yaml
peer1_values.yaml
peer2_values.yaml
.gitignore
LICENSE
README.md
ord1_values.yaml
image:
tag: 1.2.0
persistence:
accessMode: ReadWriteOnce
size: 1Gi
storageClass: "managed-premium"
caAddress: ca.lf.aidtech-test.xyz
caUsername: ord1
ord:
hlfToolsVersion: 1.2.0
type: kafka
MSP ID of the Orderer
mspID: AidTechMSP
secrets:
This should contain "genesis" block derived from a configtx.yaml
configtxgen -profile OrdererGenesis -outputBlock genesis.block
genesis: hlf--genesis
adminCert: hlf--org-admincert
caServerTls: ca--tls
affinity:
podAntiAffinity:
preferredDuringSchedulingIgnoredDuringExecution:
- weight: 95
podAffinityTerm:
master
Ln 1, Col 1
Spaces: 2
UTF-8
LF
YAML

Definir Orderer Identity

- Instalar Kafka Helm Chart
- Instalar Fabric Orderer Helm Chart
- Definir Fabric Orderer Identity

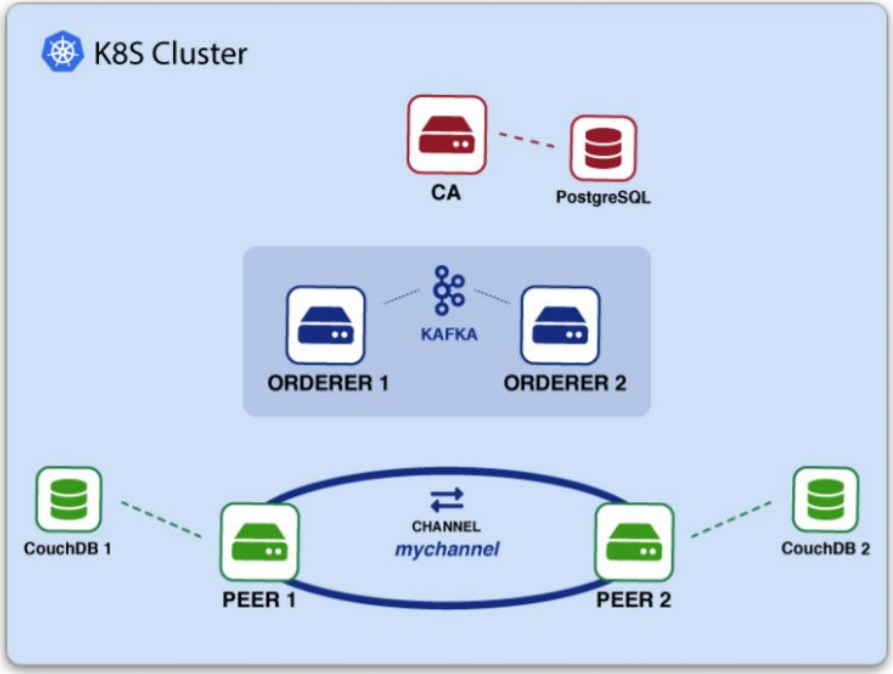
```
ORD_SECRET=$(kubectl get secret -n blockchain ord${NUM}-hlf-ord
-o jsonpath="{.data.CA_PASSWORD}" | base64 --decode)

kubectl exec $CA_POD -n blockchain  -- fabric-ca-client register --
id.name ord${NUM} --id.secret $ORD_SECRET --id.type orderer

ORD_POD=$(kubectl get pods -n blockchain -l "app=hlf-
ord,release=ord${NUM}" -o jsonpath="{.items[0].metadata.name}")

kubectl logs -n blockchain $ORD_POD | grep 'completeInitialization'
```

THE DEVELOPER'S CONFERENCE

Fabric Peer

Fabric Peer

- Instalar CouchDB Helm Chart
- Instalar Fabric Peer Helm Chart
- Definir PeerIdentity
- Criar Channel
- Ingressar no Channel

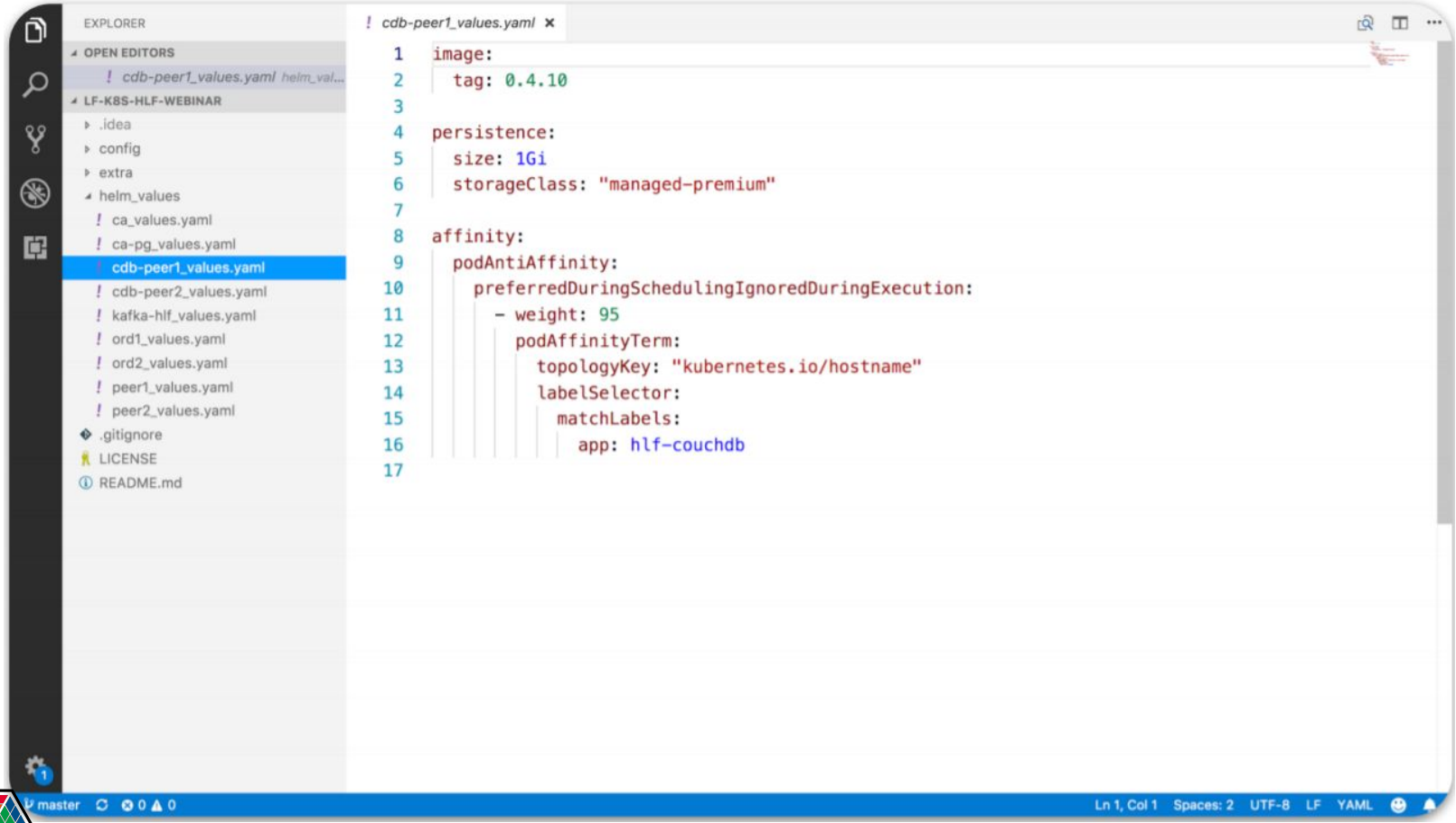


Instalar CouchDB Helm Chart

Instalar CouchDB Helm Chart

```
export NUM=1

helm install stable/hlf-couchdb -n cdb-peer${NUM} --namespace
blockchain -f ./helm_values/cdb-peer${NUM}_values.yaml

CDB_POD=$(kubectl get pods -n blockchain -l "app=hlf-
couchdb,release=cdb-peer${NUM}" -o
jsonpath="{.items[*].metadata.name}")

kubectl logs -n blockchain $CDB_POD | grep 'Apache CouchDB
has started on'
```

THE DEVELOPER'S CONFERENCE

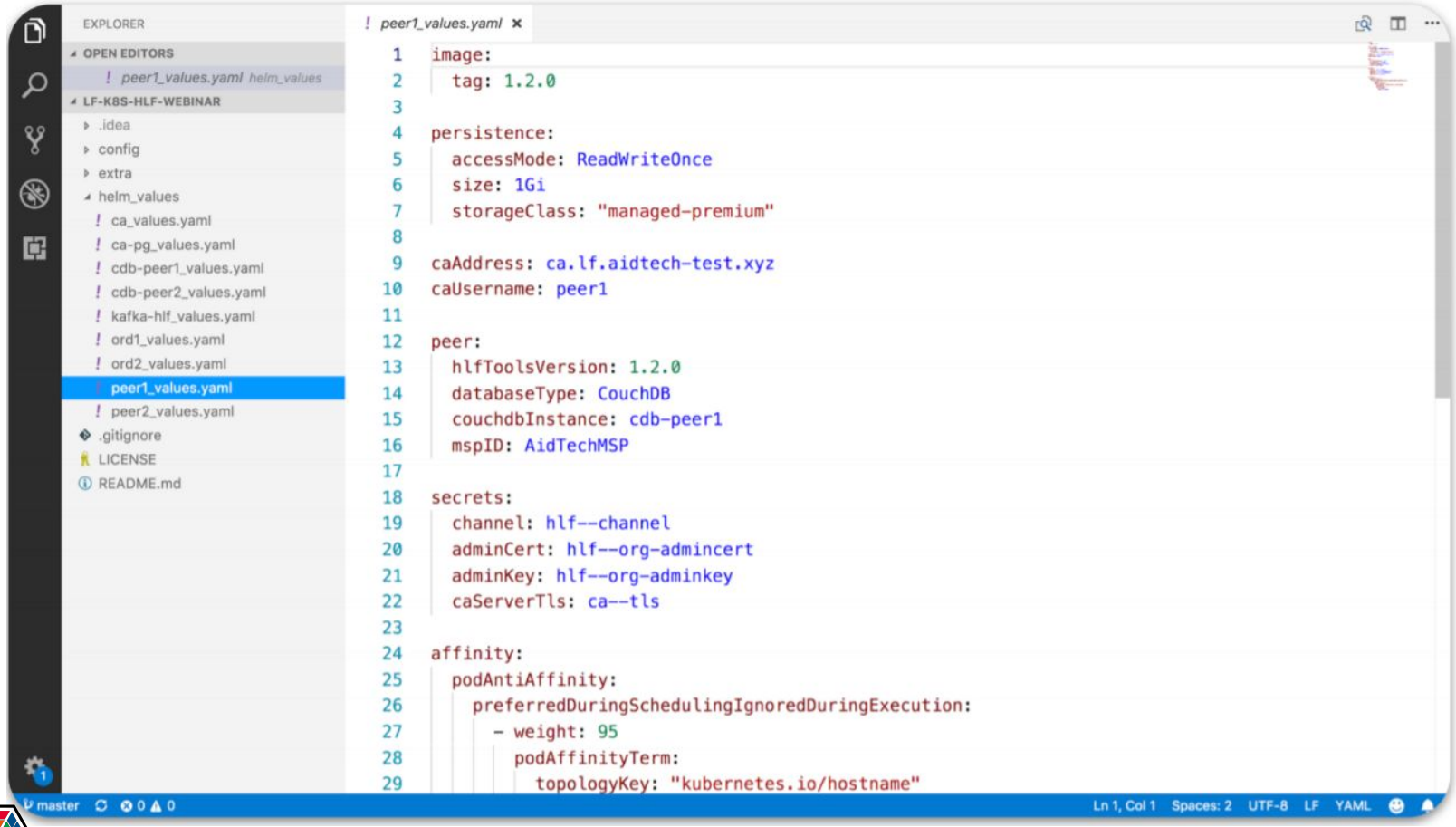

EXPLORER
OPEN EDITORS
cdb-peer1_values.yaml helm_val...
LF-K8S-HLF-WEBINAR
.idea
config
extra
helm_values
ca_values.yaml
ca-pg_values.yaml
cdb-peer1_values.yaml
cdb-peer2_values.yaml
kafka-hlf_values.yaml
ord1_values.yaml
ord2_values.yaml
peer1_values.yaml
peer2_values.yaml
.gitignore
LICENSE
README.md
cdb-peer1_values.yaml
image:
tag: 0.4.10
persistence:
size: 1Gi
storageClass: "managed-premium"
affinity:
podAntiAffinity:
preferredDuringSchedulingIgnoredDuringExecution:
- weight: 95
podAffinityTerm:
topologyKey: "kubernetes.io/hostname"
labelSelector:
matchLabels:
app: hlf-couchdb
master
Ln 1, Col 1
Spaces: 2
UTF-8
LF
YAML

Instalar Fabric Peer Helm Chart

Instalar CouchDB Helm Chart

Instalar Fabric Peer Helm Chart

```
helm install stable/hlf-peer -n peer${NUM} --
namespace blockchain -f
./helm_values/peer${NUM}_values.yaml
```


EXPLORER
OPEN EDITORS
peer1_values.yaml helm_values
LF-K8S-HLF-WEBINAR
.idea
config
extra
helm_values
ca_values.yaml
ca-pg_values.yaml
cdb-peer1_values.yaml
cdb-peer2_values.yaml
kafka-hlf_values.yaml
ord1_values.yaml
ord2_values.yaml
peer1_values.yaml
peer2_values.yaml
.gitignore
LICENSE
README.md
peer1_values.yaml
image:
tag: 1.2.0
persistence:
accessMode: ReadWriteOnce
size: 1Gi
storageClass: "managed-premium"
caAddress: ca.lf.aidtech-test.xyz
caUsername: peer1
peer:
hlfToolsVersion: 1.2.0
databaseType: CouchDB
couchdbInstance: cdb-peer1
mspID: AidTechMSP
secrets:
channel: hlf--channel
adminCert: hlf--org-admincert
adminKey: hlf--org-adminkey
caServerTls: ca--tls
affinity:
podAntiAffinity:
preferredDuringSchedulingIgnoredDuringExecution:
- weight: 95
podAffinityTerm:
topologyKey: "kubernetes.io/hostname"
master
Ln 1, Col 1
Spaces: 2
UTF-8
LF
YAML

Definir Peer Identity

Instalar CouchDB Helm Chart

Instalar Fabric Peer Helm Chart

Definir Peer Identity

```
PEER_SECRET=$(kubectl get secret -n blockchain peer${NUM}-hlf-peer -o jsonpath="{.data.CA_PASSWORD}" | base64 --decode)

kubectl exec -n blockchain $CA_POD -- fabric-ca-client register --id.name peer${NUM} --id.secret $PEER_SECRET --id.type peer

PEER_POD=$(kubectl get pods -n blockchain -l "app=hlf-peer,release=peer${NUM}" -o jsonpath="{.items[0].metadata.name}")

kubectl logs -n blockchain $PEER_POD | grep 'Starting peer'
```

THE DEVELOPER'S CONFERENCE

Criar Channel

- Instalar CouchDB Helm Chart
- Instalar Fabric Peer Helm Chart
- Definir Peer Identity
- Criar Channel

```
kubectl exec -n blockchain $PEER_POD -- peer channel create -o ord1-hlf-ord.blockchain.svc.cluster.local:7050 -c mychannel -f /hl_config/channel/mychannel.tx
```

Ingressar no Channel

- Instalar CouchDB Helm Chart
- Instalar Fabric Peer Helm Chart
- Definir Peer Identity
- Criar Channel
- Ingressar Channel

```
kubectl exec -n blockchain $PEER_POD -- peer channel fetch config /var/hyperledger/mychannel.block -c mychannel -o ord1-hlf-ord.blockchain.svc.cluster.local:7050

kubectl exec -n blockchain $PEER_POD -- bash -c 'CORE_PEER_MSPCONFIGPATH=$ADMIN_MSP_PATH peer channel join -b /var/hyperledger/mychannel.block'

kubectl exec -n blockchain $PEER_POD -- peer channel list
```

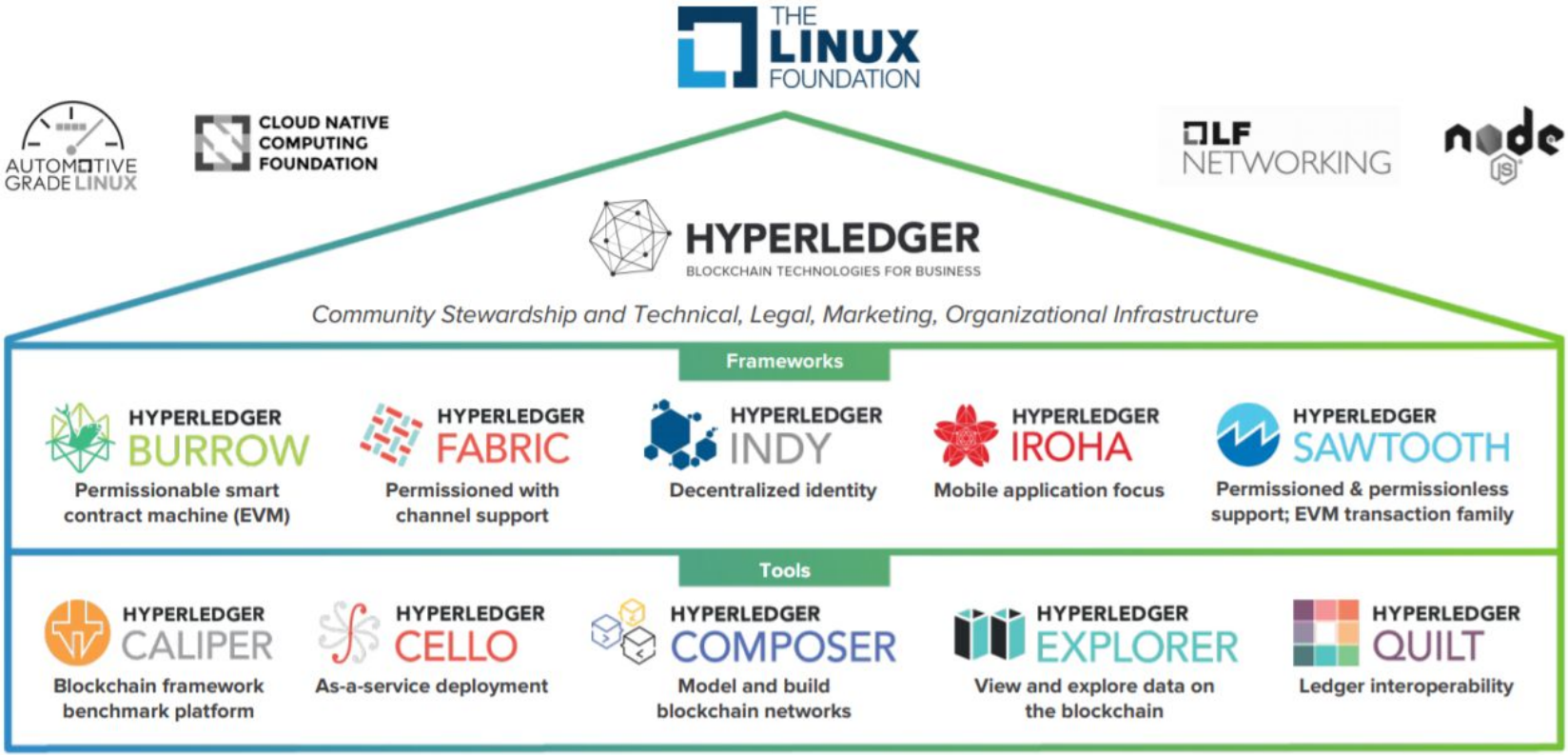
THE DEVELOPER'S CONFERENCE

Próximo Passos

2019...

Disponibilização de **Chart novo** para deploy de **Hyperledger Composer**

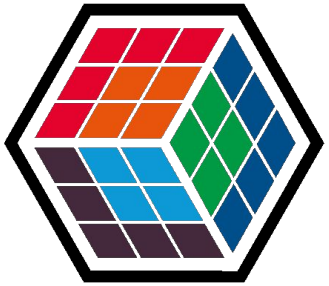
Colaboração na criação de **chart** para **Hyperledger Sawtooh**


THE LINUX FOUNDATION
AUTOMOTIVE GRADE LINUX
CLOUD NATIVE COMPUTING FOUNDATION
LF NETWORKING
node
HYPERLEDGER
BLOCKCHAIN TECHNOLOGIES FOR BUSINESS
Community Stewardship and Technical, Legal, Marketing, Organizational Infrastructure
Frameworks
HYPERLEDGER BURROW
Permissionable smart contract machine (EVM)
HYPERLEDGER FABRIC
Permissioned with channel support
HYPERLEDGER INDY
Decentralized identity
HYPERLEDGER IROHA
Mobile application focus
HYPERLEDGER SAWTOOTH
Permissioned & permissionless support; EVM transaction family
Tools
HYPERLEDGER CALIPER
Blockchain framework benchmark platform
HYPERLEDGER CELLO
As-a-service deployment
HYPERLEDGER COMPOSER
Model and build blockchain networks
HYPERLEDGER EXPLORER
View and explore data on the blockchain
HYPERLEDGER QUILT
Ledger interoperability

Perguntas?

Cláudio Ramos
claudiovtramos@gmail.com

THE DEVELOPER'S CONFERENCE



THE DEVELOPER'S CONFERENCE