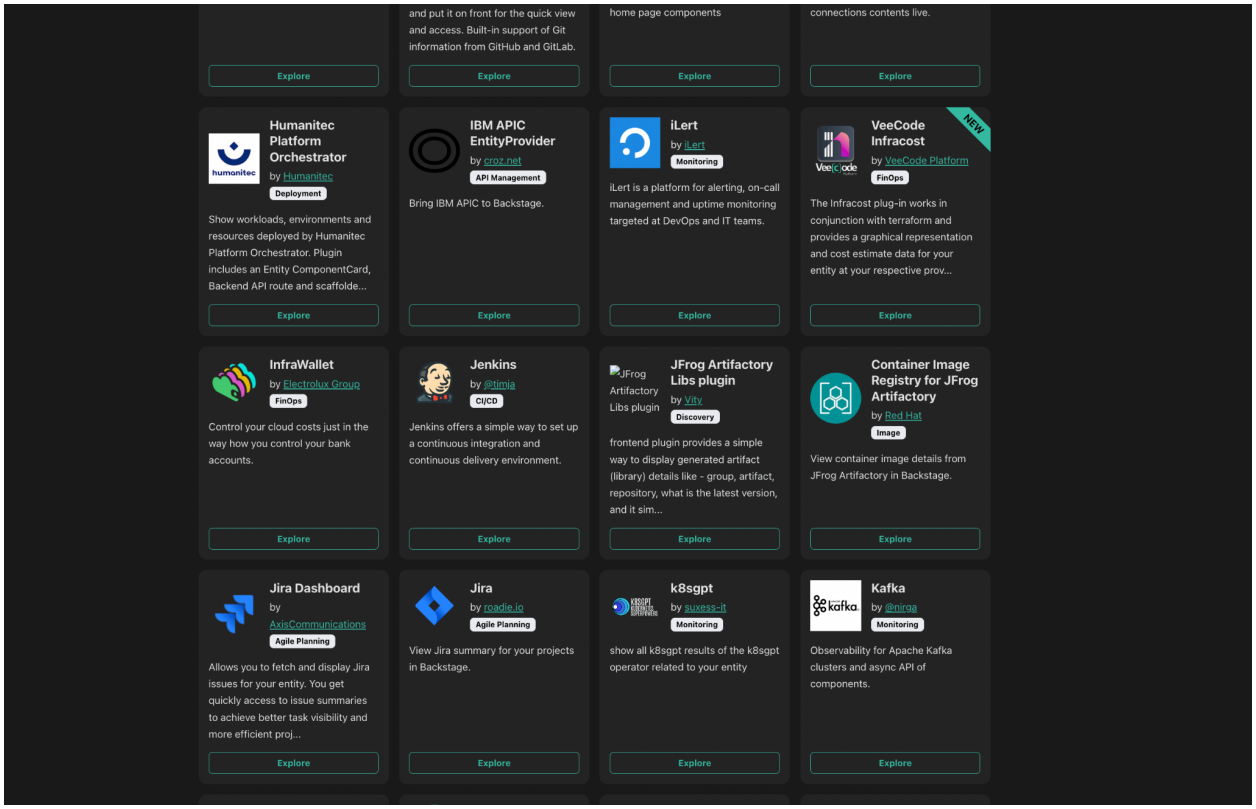
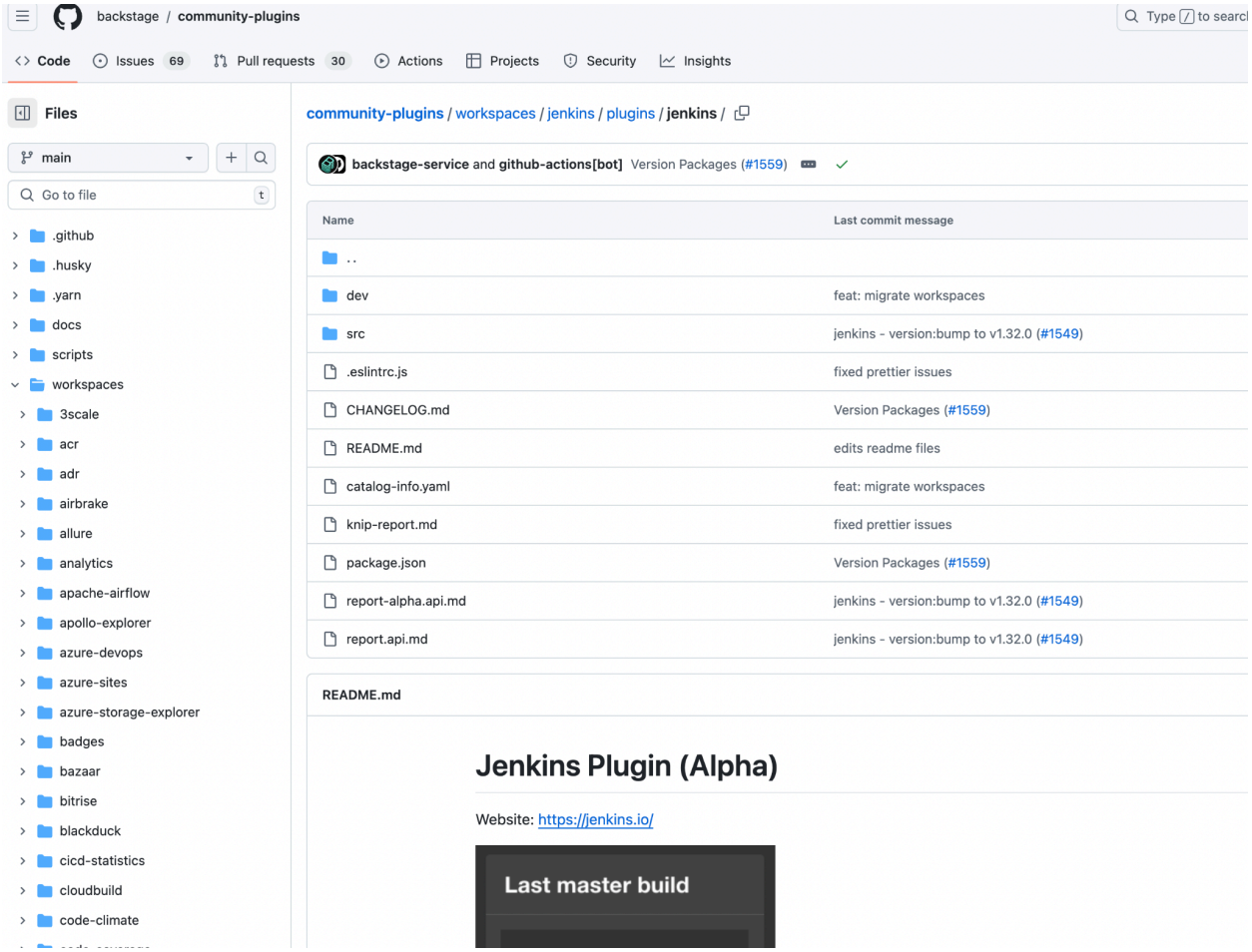
Lab F.
This laboratory provides a step-by-step guide for integrating the Jenkins plugin into Backstage. The Jenkins plugin allows you to seamlessly view and access your Jenkins pipelines and builds within the Backstage Developer Hub. This introduction will guide you through the installation process, enabling you to leverage the plugin's features and enhance your development workflow. Furthermore, it will walk through the steps necessary to customize the frontend.

Ex.1 Browse plugins, download Jenkins plugin

Open a browser window and go to https://backstage.io/plugins/
Search forJenkins plugin and click on explore



You will land on a github page



Please take 5 minutes to read all the instructions given at README.md, the landing page, then open a terminal window and type the following commands:

cd /home/tibco/tibco-developer-hub-main/

yarn --cwd packages/app add @backstage-community/plugin-jenkins

```
tibco@ip-172-31-12-237:~$ cd tibco-developer-hub-main
tibco@ip-172-31-12-237:~/tibco-developer-hub-main$ # From your Backstage root directory
yarn --cwd packages/app add @backstage-community/plugin-jenkins
yarn add v1.22.19
[1/4] Resolving packages...
warning Lockfile has incorrect entry for "@types/react@^16.13.1 || ^17.0.0". Ignoring it.
[2/4] Fetching packages...
warning Pattern ["app@link:packages/app"] is trying to unpack in the same destination "/home/tibco/.cache/yarn/v6/np
m-app-1.3.0/node_modules/app" as pattern ["app@1.3.0","app@1.3.0"]. This could result in non-deterministic behavior,
 skipping.
[3/4] Linking dependencies...
warning "workspace-aggregator-bf4ea170-0861-46c1-8233-e3a17114d4c8 > @internal/backstage-plugin-import-flow@1.3.0" h
as unmet peer dependency "react@^16.13.1 || ^17.0.0 || ^18.0.0".
warning "workspace-aggregator-bf4ea170-0861-46c1-8233-e3a17114d4c8 > @internal/backstage-plugin-import-flow@1.3.0" h
as unmet peer dependency "react-router-dom@^6.3.0".
warning "workspace-aggregator-bf4ea170-0861-46c1-8233-e3a17114d4c8 > @internal/plugin-tibco-platform-plugin@1.3.0" h
as unmet peer dependency "react@^16.13.1 || ^17.0.0 || ^18.0.0".
warning "@backstage-community/plugin-jenkins > @material-ui/core@4.12.4" has unmet peer dependency "react@^16.8.0 ||
 ^17.0.0".
warning "@backstage-community/plugin-jenkins > @material-ui/core@4.12.4" has unmet peer dependency "react-dom@^16.8.
0 || ^17.0.0".
warning "@backstage-community/plugin-jenkins > @material-ui/icons@4.11.3" has unmet peer dependency "react@^16.8.0 |
| ^17.0.0".
```

And wait till you see

```
warning "@backstage-community/plugin-jenkins > @backstage/plugin-catalog-react > @backstage/frontend-test-utils > @b
ackstage/test-utils > @backstage/core-app-api@1.15.1" has unmet peer dependency "react-dom@^16.13.1 || ^17.0.0 || ^1
8.0.0".
warning "@backstage-community/plugin-jenkins > @backstage/plugin-catalog-react > @backstage/frontend-test-utils > @b
ackstage/test-utils > @backstage/core-app-api@1.15.1" has unmet peer dependency "react-router-dom@6.0.0-beta.0 || ^6
.3.0".
warning " > @backstage-community/plugin-jenkins@0.11.1" has unmet peer dependency "react@^16.13.1 || ^17.0.0 || ^18.
0.0".
warning " > @backstage-community/plugin-jenkins@0.11.1" has unmet peer dependency "react-dom@^16.13.1 || ^17.0.0 ||
^18.0.0".
warning " > @backstage-community/plugin-jenkins@0.11.1" has unmet peer dependency "react-router-dom@6.0.0-beta.0 ||
^6.3.0".
[4/4] Building fresh packages...
success Saved lockfile.
success Saved 5 new dependencies.
info Direct dependencies
info All dependencies
├─ @backstage-community/plugin-jenkins-common@0.1.30
├─ @backstage-community/plugin-jenkins@0.11.1
├─ @backstage/frontend-defaults@0.1.1
├─ @backstage/frontend-test-utils@0.2.1
└─ @types/scheduler@0.16.8
Done in 155.90s.
tibco@ip-172-31-12-237:~/tibco-developer-hub-main$
```

The command you just executed is explained below:

- yarn is the package manager.
- --cwd packages/app changes the current working directory to the packages/app folder, where your Backstage frontend app is located.
- add @backstage-community/plugin-jenkins installs the Jenkins plugin as a dependency in your frontend app.

This ensures that the plugin is added to your packages/app folder, where it can be used in the Developer HUB frontend.

In jenkings plugin readme.md file look for Section and click on jenkins-backend setup as per instruction below step 2

Setup

1. If you have a standalone app (you didn't clone this repo), then do

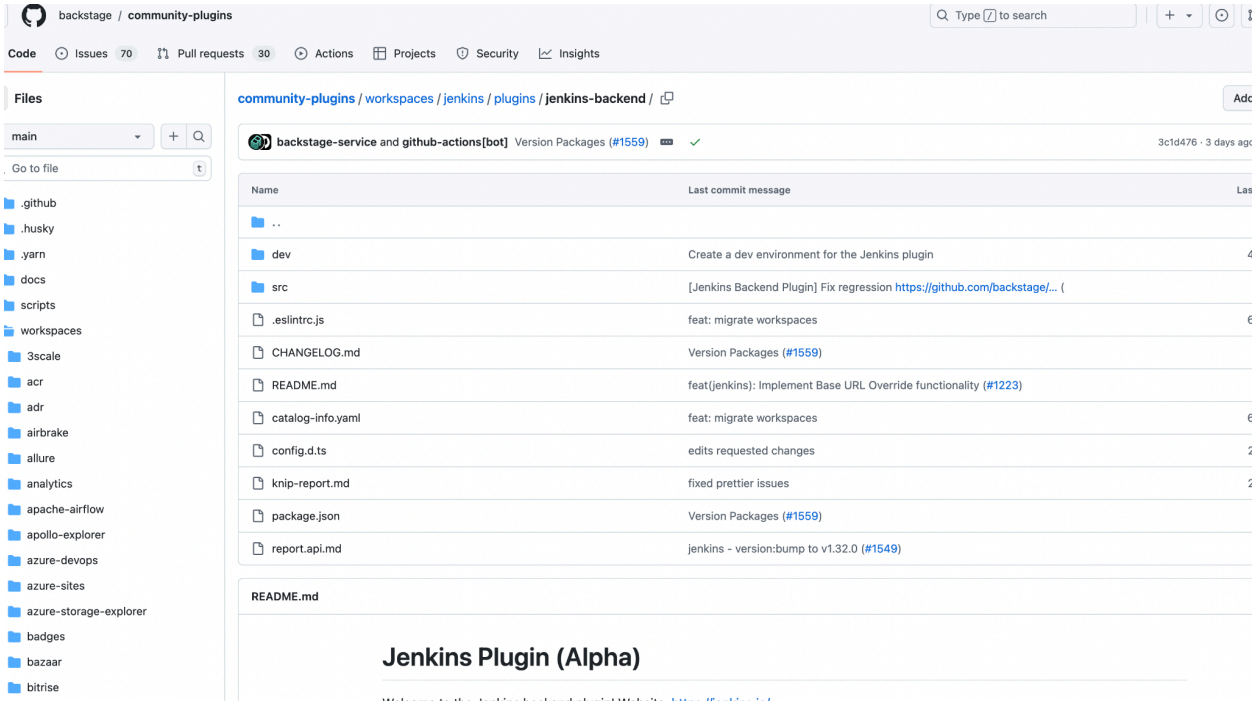
```
# From your Backstage root directory
yarn --cwd packages/app add @backstage-community/plugin-jenkins
```

2. Add and configure the jenkins-backend plugin according to it's instructions

3. Add the `EntityJenkinsContent` extension to the `CI/CD` page and `EntityLatestJenkinsRunCard` to the `overv` wherever you'd prefer):

Note that if you configured a custom JenkinsInfoProvider in step 2, you may need a custom isJenkinsAvailable. Also i a new default branch name, you can pass multiple branch names as a comma-separated list and it will check for eac

So if you click on https://github.com/backstage/community-plugins/tree/main/workspaces/jenkins/plugins/jenkins-backend you will land on



From the same terminal window type:

cd /home/tibco/tibco-developer-hub-main/packages/backend/src/
gedit index.ts &

It will open gedit :

Add to the bottom before backend.start():

//jenkins plugin
backend.add(import('@backstage-community/plugin-jenkins-backend'));

As per screenshot below

```
103 //jenkins plugin
104 backend.add(import('@backstage-community/plugin-jenkins-backend'));
105
106 backend.start();
```

Save and close

From the same terminal window add backend package
cd /home/tibco/tibco-developer-hub-main
`yarn --cwd packages/backend add @backstage-community/plugin-jenkins-backend`
It's the only step necessary in this case to install the backend, as per screenshot below, from jenkins readme.md

New Backend System

The jenkins backend plugin has support for the new backend system, here's how you can set that up:

In your `packages/backend/src/index.ts` make the following changes:

```
import { createBackend } from '@backstage/backend-defaults';
const backend = createBackend();
// ... other feature additions
backend.add(import('@backstage-community/plugin-jenkins-backend'));
backend.start();
```

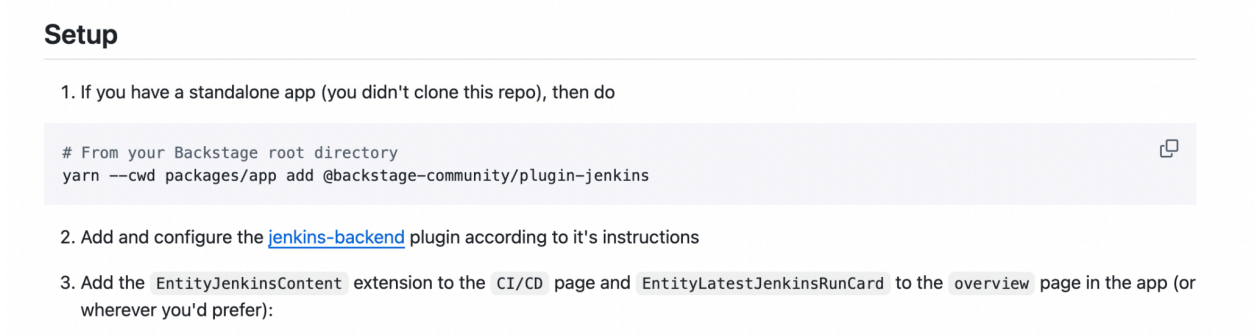
Integrating into a backstage instance

This plugin needs to be added to an existing backstage instance

"New Backend System" and "Integrating into a backstage instance" are alternative steps. DeveloperHUB 1.3 supports the new backend, anyway not all the plugins can support this.

Go back to your browser window and close the jenkins backend page and surf to https://github.com/backstage/community-plugins/blob/main/workspaces/jenkins/plugins/jenkins/README.md

In the setup section move to step 3.



Setup

1. If you have a standalone app (you didn't clone this repo), then do

```
# From your Backstage root directory
yarn --cwd packages/app add @backstage-community/plugin-jenkins
```

2. Add and configure the jenkins-backend plugin according to it's instructions
3. Add the `EntityJenkinsContent` extension to the `CI/CD` page and `EntityLatestJenkinsRunCard` to the `overview` page in the app (or wherever you'd prefer):

From the same terminal execute the below commands:

```
cd /home/tibco/tibco-developer-hub-main/packages/app/src/components/catalog/
gedit EntityPage.tsx &
```

Then add at line 2 the code

```
import {
  EntityJenkinsContent,
  EntityLatestJenkinsRunCard,
  isJenkinsAvailable,
} from '@backstage-community/plugin-jenkins';
```

```
1 import React from 'react';
2 import {
3   EntityJenkinsContent,
4   EntityLatestJenkinsRunCard,
5   isJenkinsAvailable,
6 } from '@backstage-community/plugin-jenkins';
7 import { Button, Grid } from '@material-ui/core';
8 import {
9   EntityApiDefinitionCard,
10  EntityConsumedApisCard
```

Then around line 183 you should see the below code:

```
181 };
182
183 const serviceEntityPage = (
184   <EntityLayout>
185     <EntityLayout.Route path="/" title="Overview">
186       <OverviewContent />
187     </EntityLayout.Route>
188
189     <EntityLayout.Route path="/ci-cd" title="CI/CD">
190       {cicdContent}
191     </EntityLayout.Route>
192
193     <EntityLayout.Route path="/api" title="API">
194       <Grid container spacing={3} alignItems="stretch">
195         <Grid item md={6}>
196           <EntityProvidedApisCard />
197         </Grid>
198         <Grid item md={6}>
199           <EntityConsumedApisCard />
200         </Grid>
201       </Grid>
202     </EntityLayout.Route>
203
204     <EntityLayout.Route path="/dependencies" title="Dependencies">
```

You should change OverviewContent as per below:

```
183     );
184   };
185
186   const serviceEntityPage = (
187     <EntityLayout>
188       <EntityLayout.Route path="/" title="Overview">
189       {/* ... */}
190       <EntitySwitch>
191         <EntitySwitch.Case if={isJenkinsAvailable}>
192           <Grid item sm={6}>
193             <EntityLatestJenkinsRunCard
194               branch="main,master"
195               variant="gridItem"
196             />
197           </Grid>
198         </EntitySwitch.Case>
199         {/* ... */}
200       </EntitySwitch>
201     </EntityLayout.Route>
202
203     <EntityLayout.Route path="/ci-cd" title="CI/CD">
204       {cicdContent}
205     </EntityLayout.Route>
206
```

And then search for {cicdContent} const definition, it should be around line 76

```
73   </EntityTechdocsContent>
74 );
75
76 const cicdContent = (
77   // This is an example of how you can implement your company's logic in entity page.
78   // You can for example enforce that all components of type 'service' should use GitHubActions
79   <EntitySwitch>
80     <EntitySwitch.Case if={isGithubActionsAvailable}>
81       <EntityGithubActionsContent />
82     </EntitySwitch.Case>
83
84     <EntitySwitch.Case>
85       <EmptyState
86         title="No CI/CD available for this entity"
87         missing="info"
88         description="You need to add an annotation to your component if you want to enable CI/CD for it. Y
89         action={
90           <Button
91             variant="contained"
92             color="primary"
93             href="https://backstage.io/docs/features/software-catalog/well-known-annotations"
94           >
95             Read more
96           </Button>
97         }
98       />
99     </EntitySwitch.Case>
100  </EntitySwitch>
101 );
```

And change it by adding after tag <EntitySwitch> the code

```
        <EntitySwitch.Case if={isJenkinsAvailable}>
          <EntityJenkinsContent />
```

```
        </EntitySwitch.Case>
```

As per screenshot below

Save and close the file

********** **NOTE** **********
The modified file is available at /home/tibco/TDH301/Other/copy_and_paste/EntityPage.tsx
****

From the same terminal add jenkins integration to app-config.local.yaml:

cd /home/tibco/tibco-developer-hub-main
gedit app-config.local.yaml &

Then around line 30 add the code:

Jenkins integration added here
jenkins:
  baseUrl: "http://localhost:9090" # Replace with your Jenkins server URL
  username: "admin"   # Jenkins username
  apiKey: "1156227f4deeec3f18bfa4b48ddb7e6ac0"   # Jenkins API token or password

As per screenshot below:

```
      user: postgres
      password: example

## Uncomment the below github integrations config to add a PAT to try out
## new component creation using one of the available templates.
integrations:
   github:
     - host: github.com
#       # This is a Personal Access Token or PAT from GitHub. You can find out how to generate this token, and more inf
#       # about setting up the GitHub integration here: https://backstage.io/docs/getting-started/configuration#setting
       token: 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  #Enter your Github token here

# Jenkins integration added here
jenkins:
  baseUrl: "http://localhost:9090"  # Replace with your Jenkins server URL
  username: "admin"       # Jenkins username
  apiKey: "1156227f4deeec3f18bfa4b48ddb7e6ac0"          # Jenkins API token or password
# Uncomment the below catalog config to add the default(example) entities to your software catalog while running develo
# The example entities had been placed inside the 'tibco-examples' folder under the project root folder
catalog:
  locations:
    - type: url
      target: https://github.com/TIBCOSoftware/tibco-developer-hub/tree/main/tibco-examples/tibco-examples.yaml
      rules:
        - allow:
            [
              Component,
              API,
              Location,
              Template,
              User,
              Group,
              Domain,
              System,
              Resource,
            ]
auth:
```

Code is also available at
/home/tibco/TDH301/Other/copy_and_paste/JenkinsIntegrationSnippetCode.txt

Save the file and close gedit.

From the same terminal window execute the below commands:

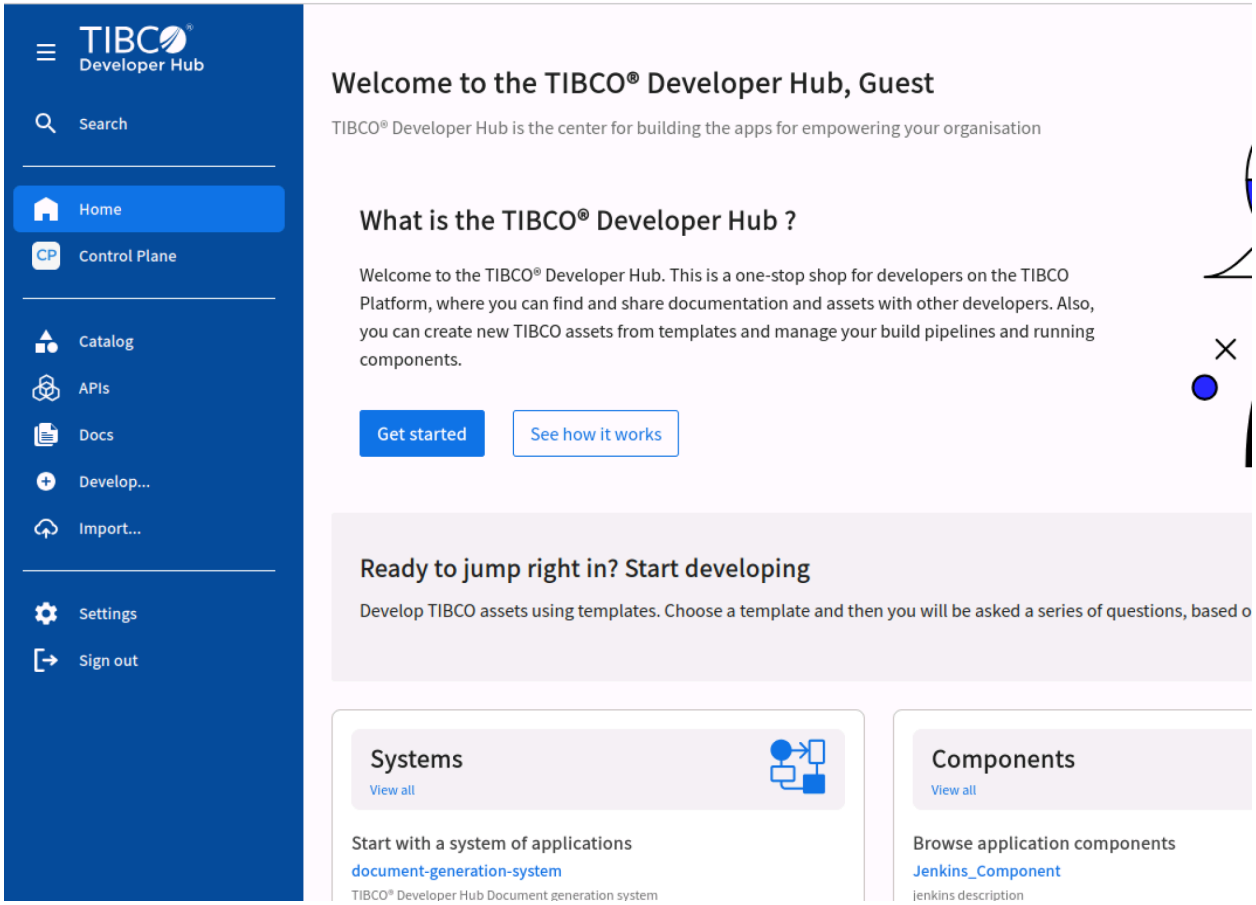
cd /home/tibco/TDH301/Other
./stop-devhub-services.sh
cd /home/tibco/tibco-developer-hub-main/docker
docker-compose up

It will start the database for local developer hub
Then open a new tab from the same terminal and execute the following commands

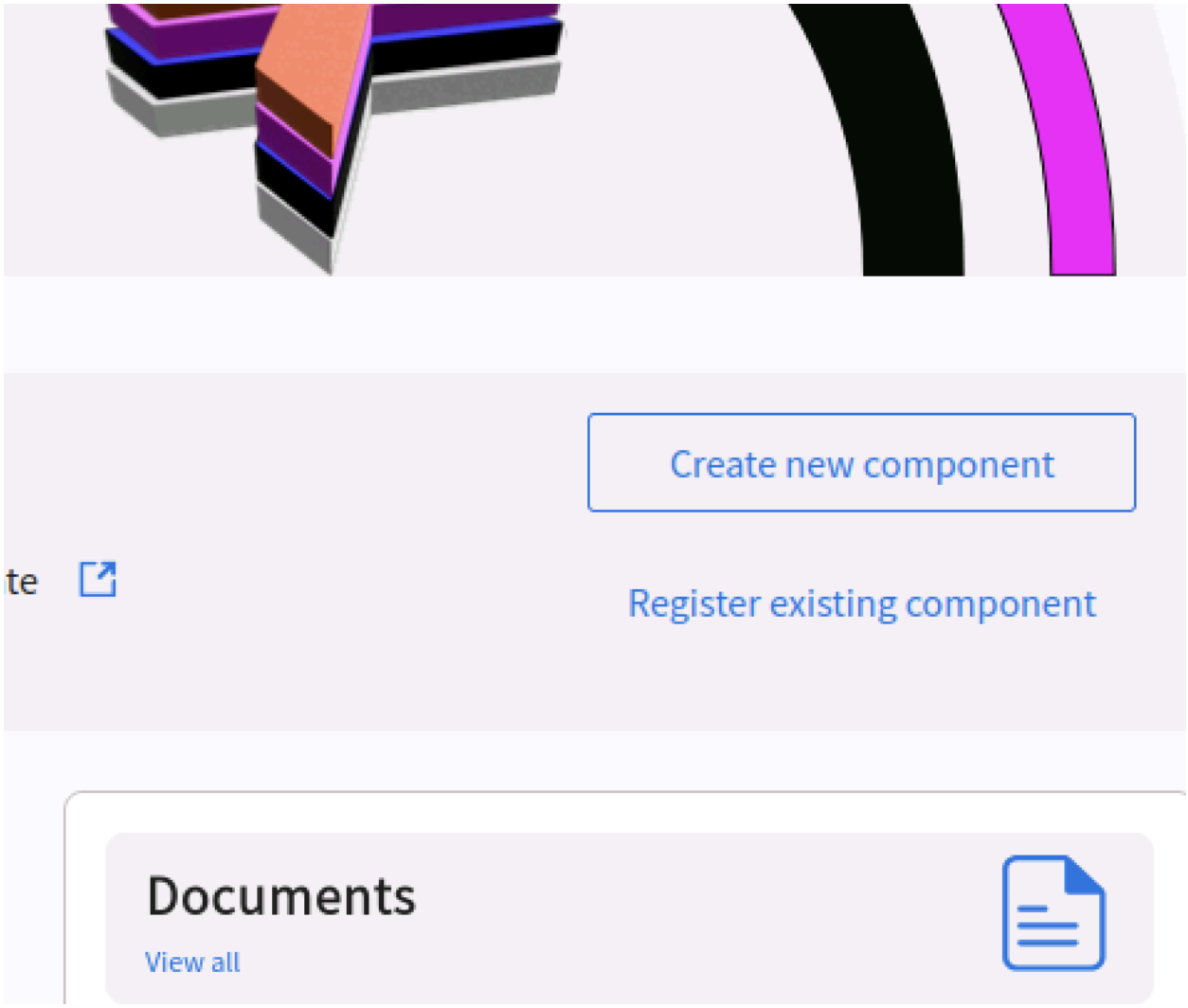
cd /home/tibco/tibco-developer-hub-main/docker
yarn dev

It will start developer hub with Jenkins plugin.

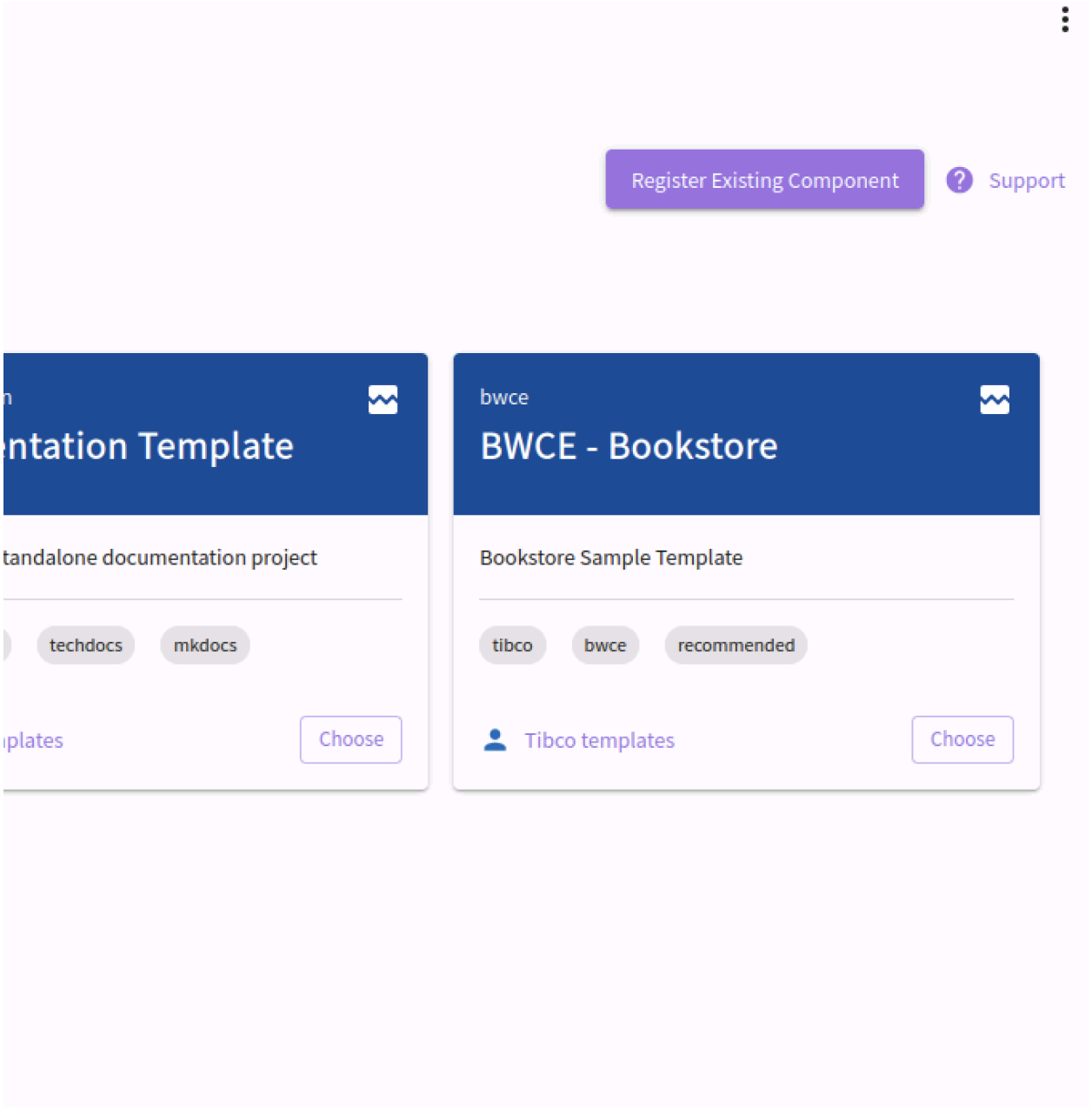
From a browser go to http://localhost:3000/tibco/hub if not already opened



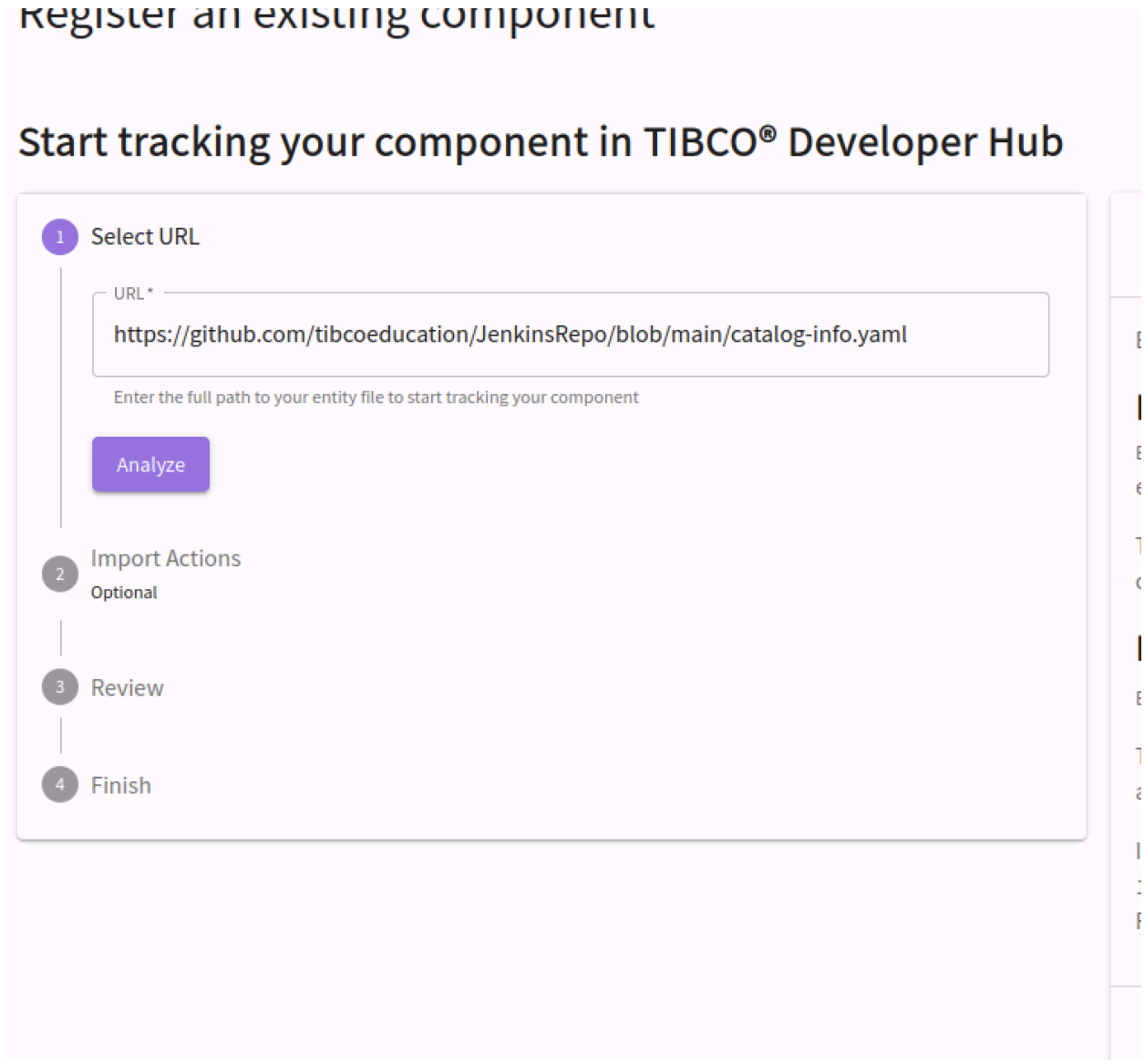
Click on Create a new component



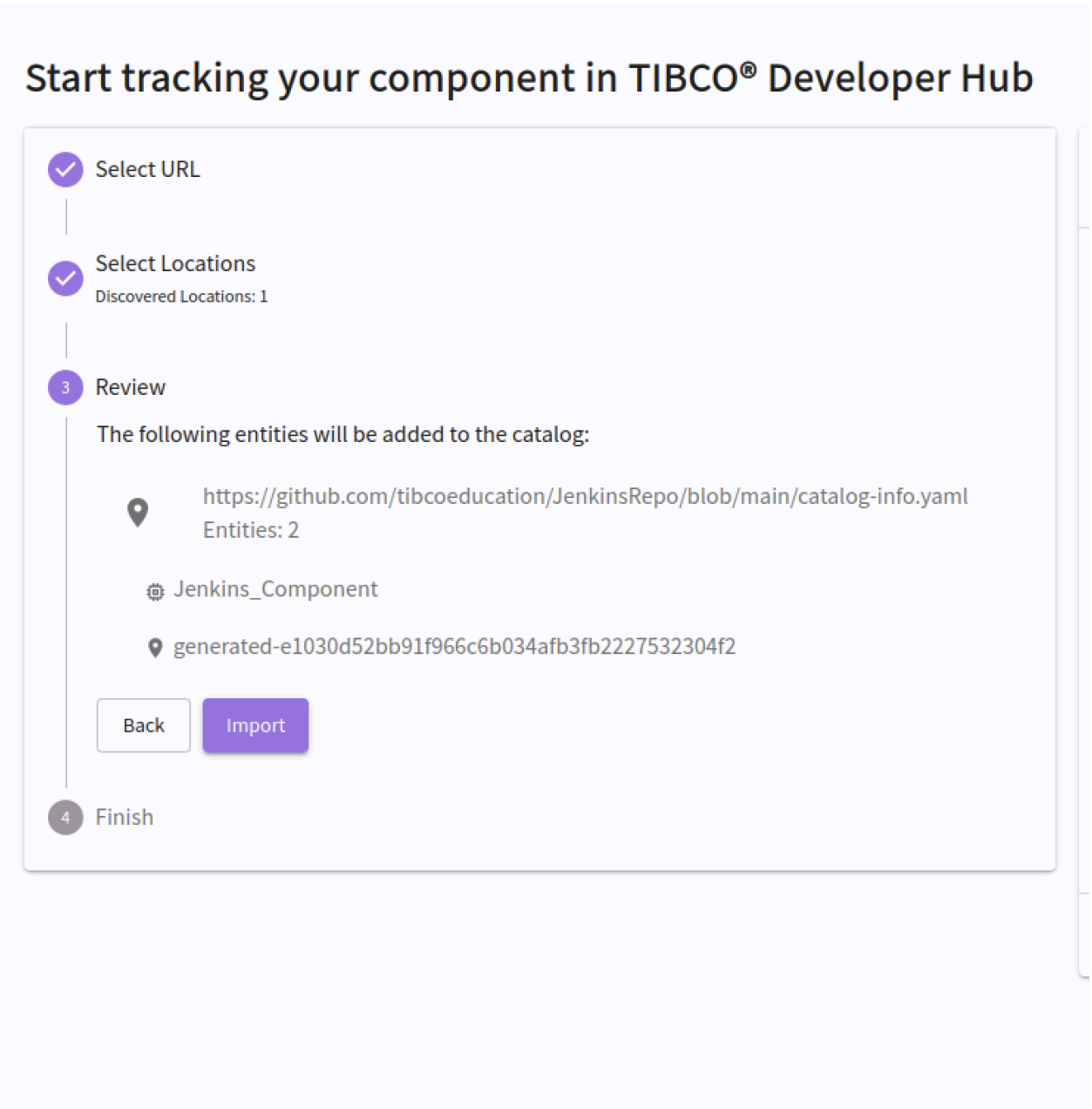
Then click on Register Existing Component



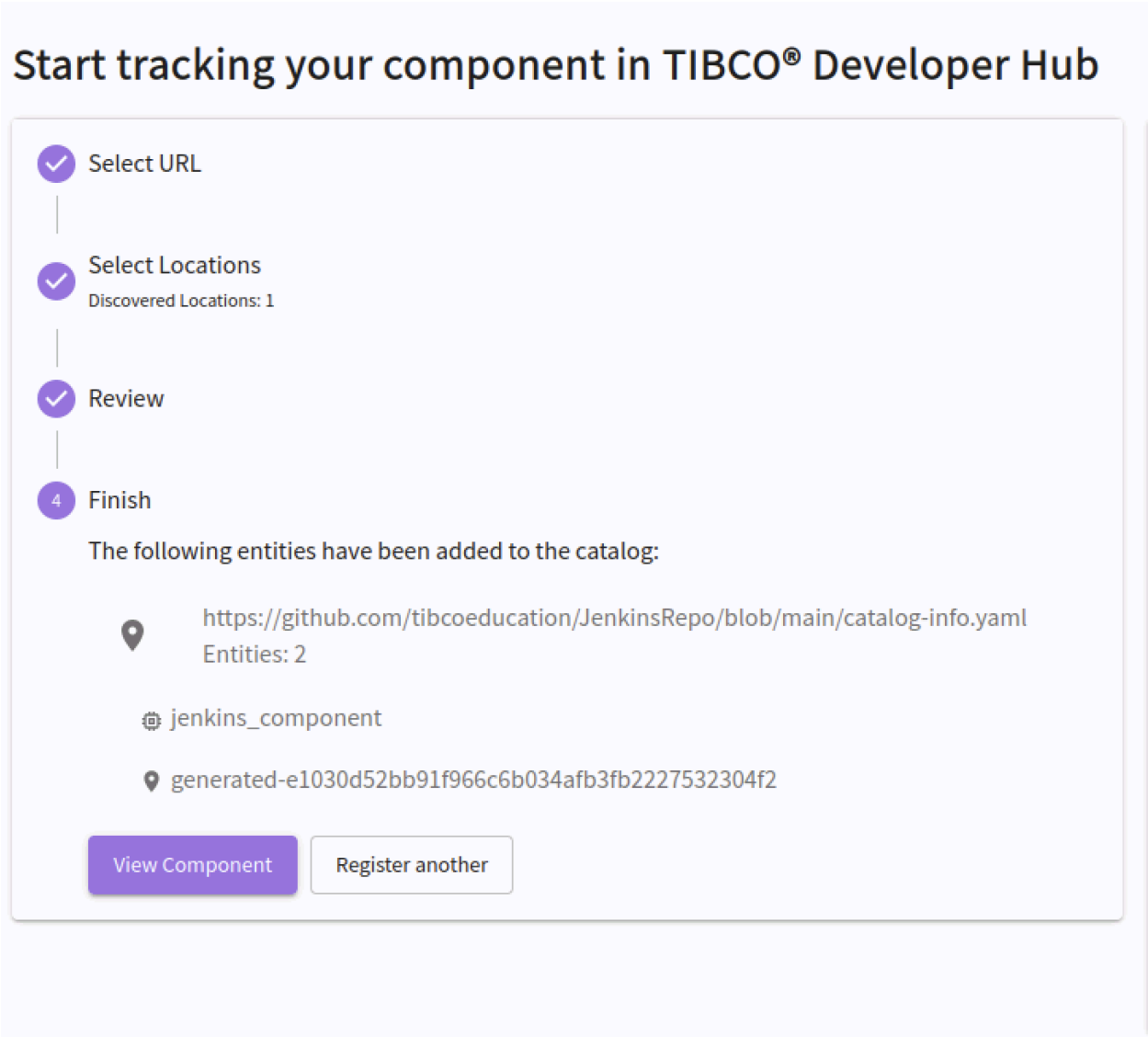
Enter URL https://github.com/tibcoeducation/JenkinsRepo/blob/main/catalog-info.yaml
And then click on Analyze


Register an existing component
Start tracking your component in TIBCO® Developer Hub
1
Select URL
URL *
https://github.com/tibcoeducation/JenkinsRepo/blob/main/catalog-info.yaml
Enter the full path to your entity file to start tracking your component
Analyze
2
Import Actions
Optional
3
Review
4
Finish

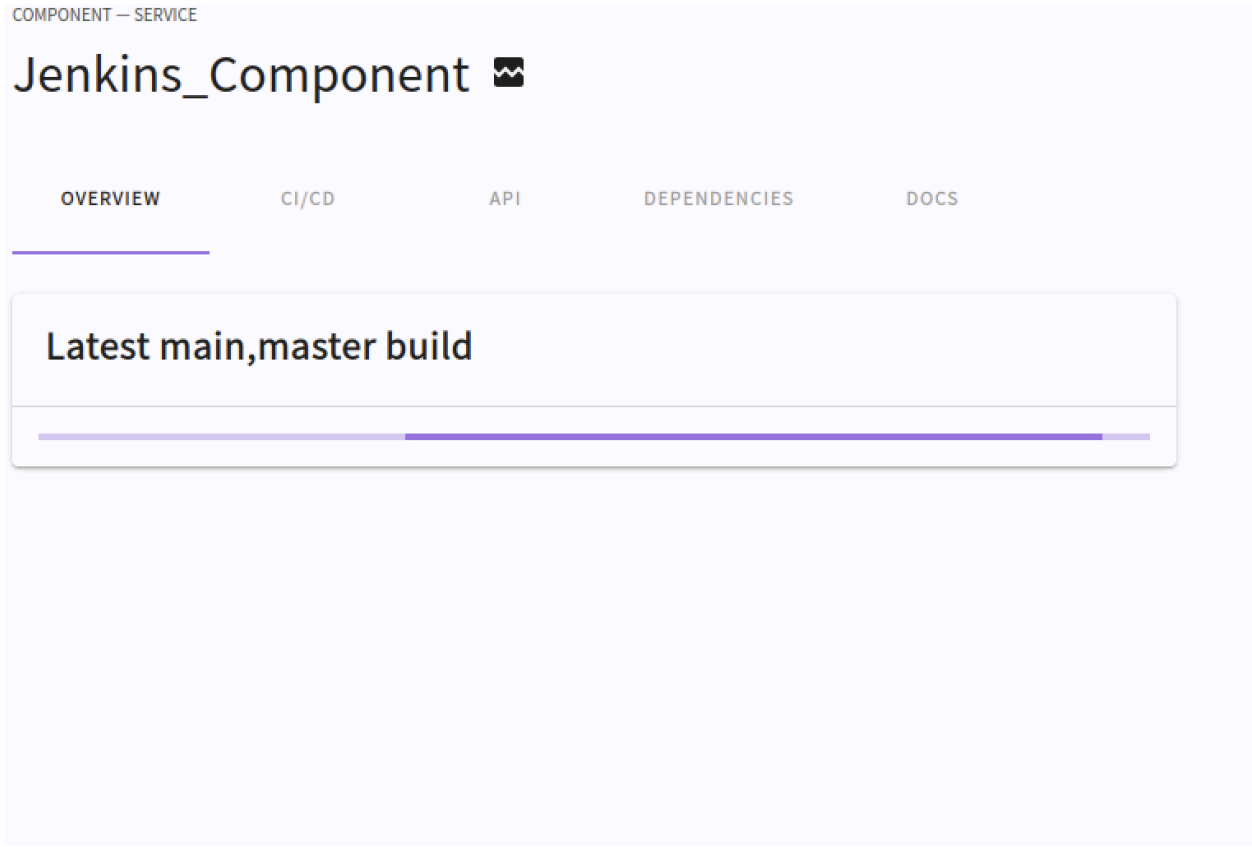
You will get



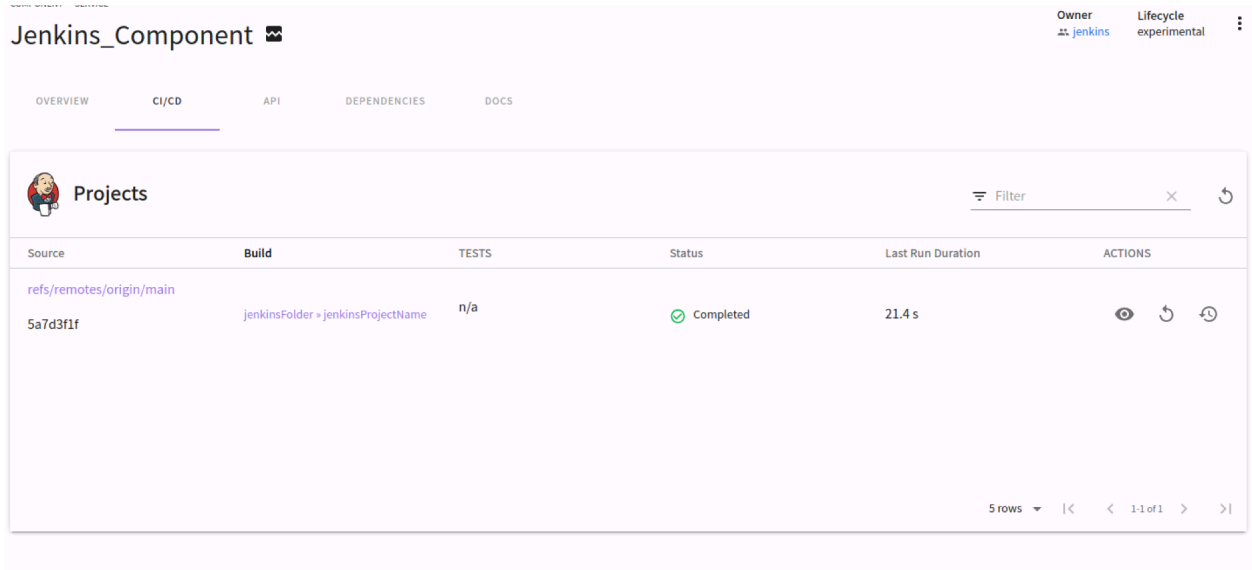
Click on Import



Then click on View Component

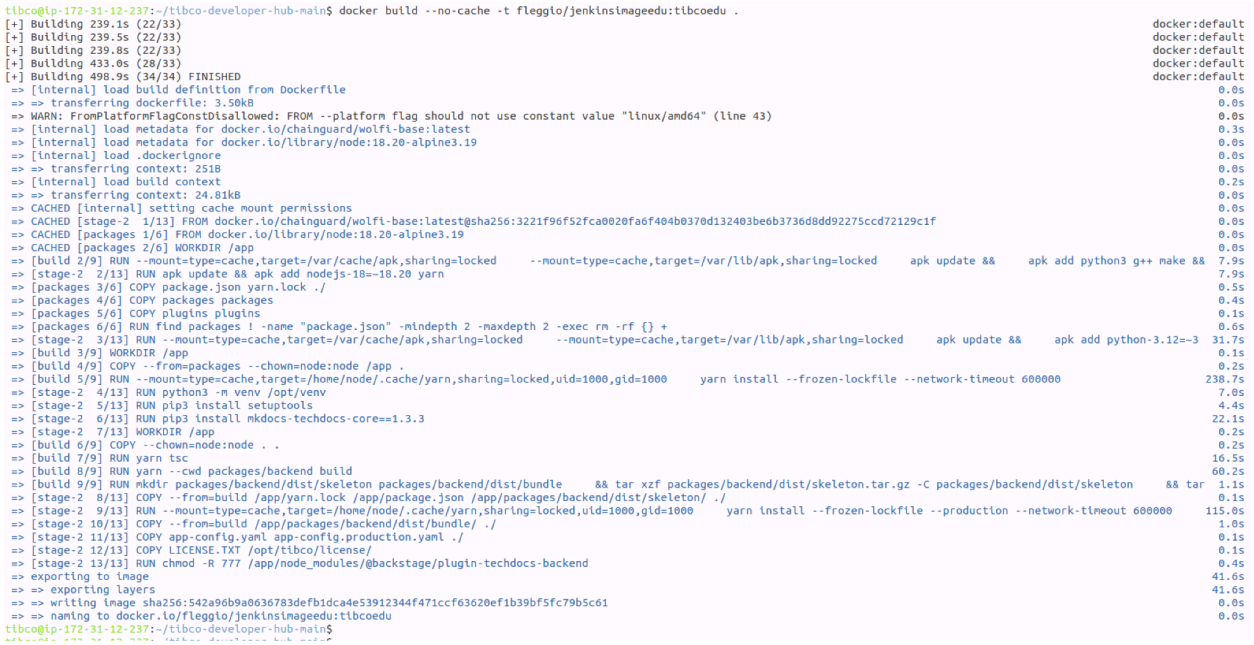


And click on CI/CD it will display the list of builds executed

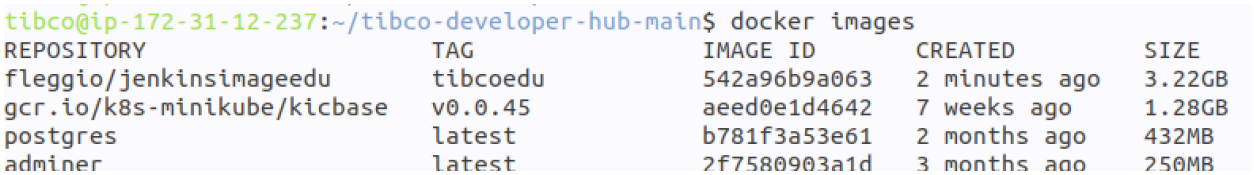


From a terminal type (it can take up to 10 minutes to complete the build):

cd /home/tibco/tibco-developer-hub-main
docker build -t <your-docker-username>/jenkinsimageedu:tibcoedu .

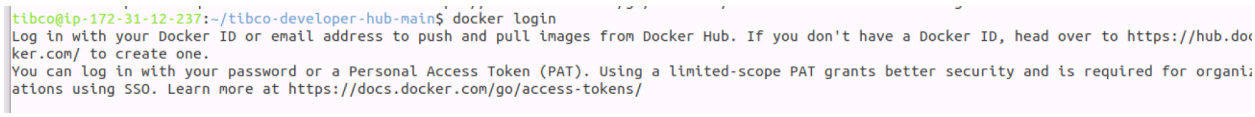


From the same terminal execute
docker images
And verify that image size is around 3,2 GB



Then type:

docker logout
docker login -u <your-docker-username>



And provide your docker username and password

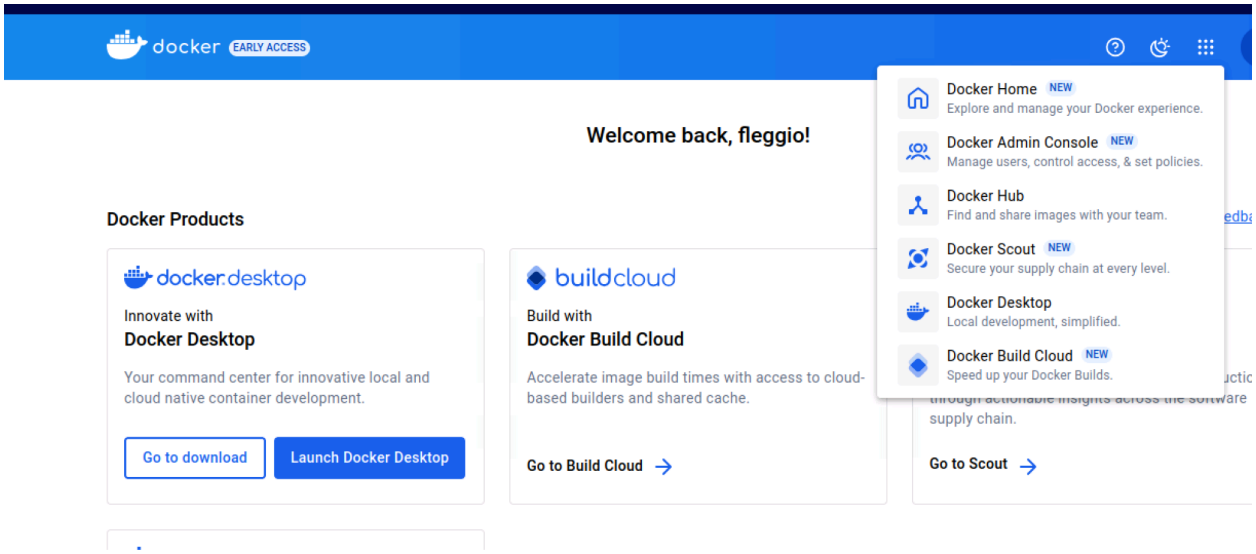
```
Password:
WARNING! Your password will be stored unencrypted in /home/tibco/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

Then push the image to your docker hub

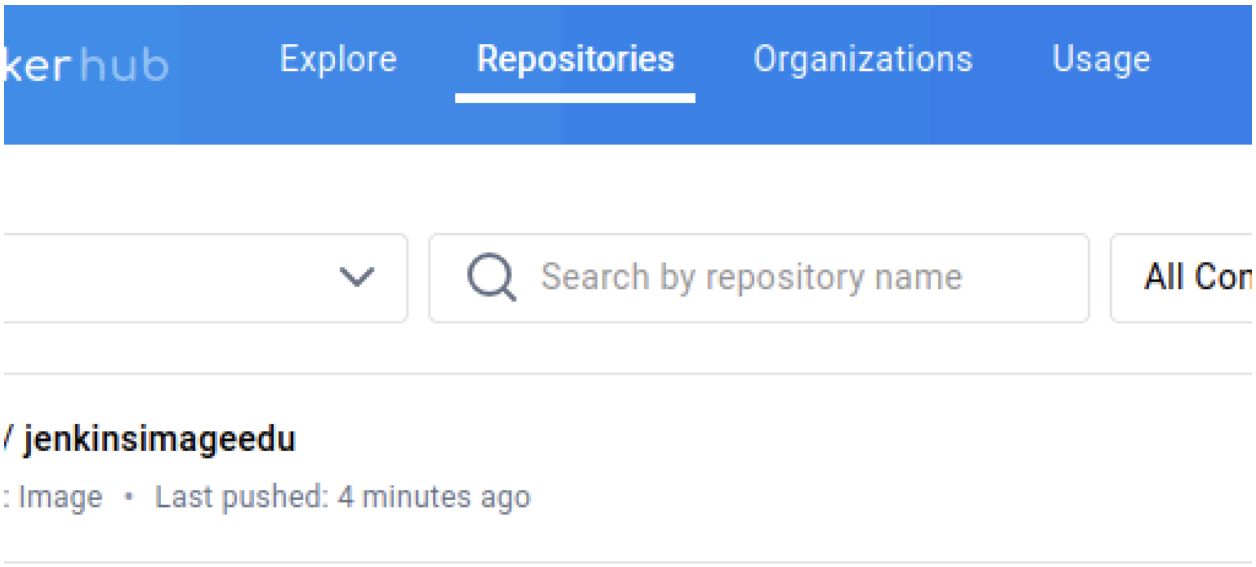
docker push <your-docker-username>/jenkinsimageedu:tibcoedu

```
tibco@ip-172-31-12-237:~/tibco-developer-hul
The push refers to repository [docker.io/fle
13506a2f9d54: Pushed
414aef402fea: Pushed
99f09d838761: Pushed
f0ae223ce8e9: Pushed
69f4b4df5a5e: Pushed
203ac3917d71: Mounted from library/node
3080a0a0ca08: Mounted from library/node
317a42804fcd: Mounted from library/node
94b0f5987cb7: Mounted from library/node
tibcoedu: digest: sha256:b6c87f89b9fd88ff584
```

Verify it has been uploaded, open a browser and navigate to https://www.docker.com/, login and check your docker hub

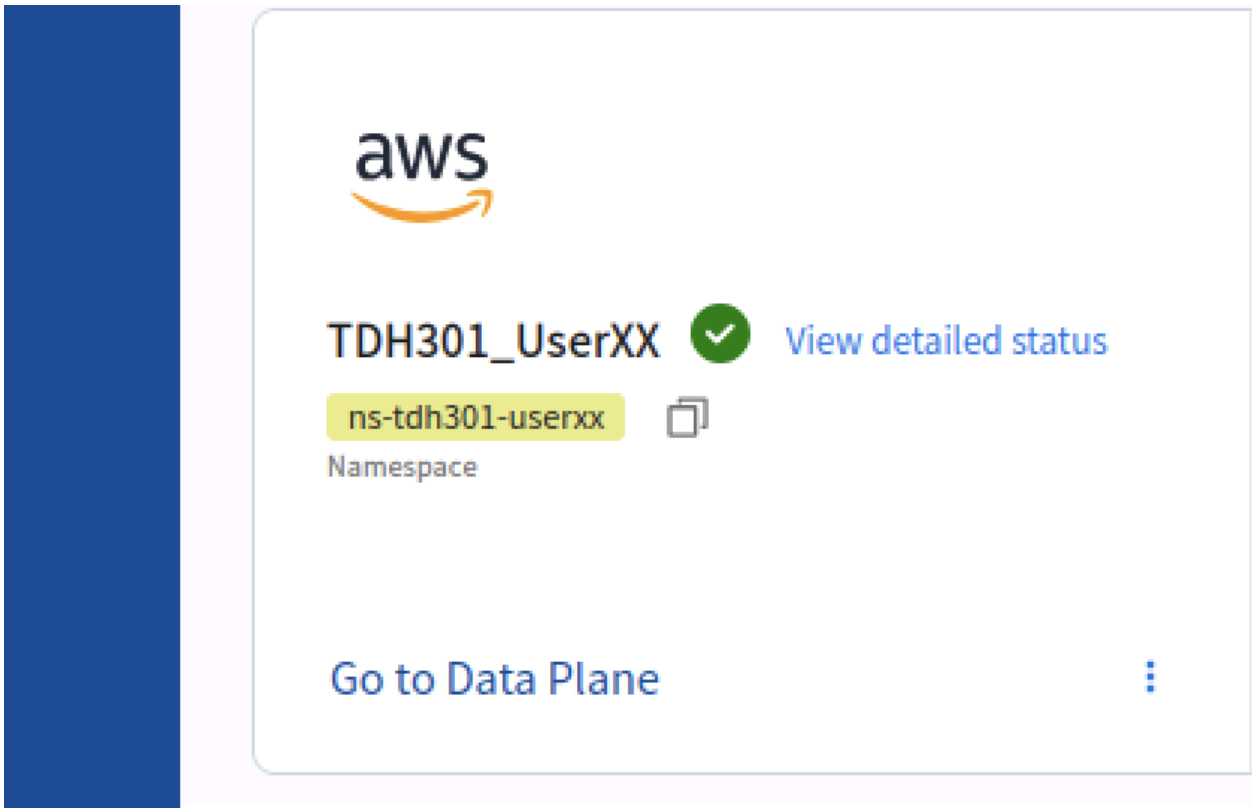


It should display the image you have just uploaded

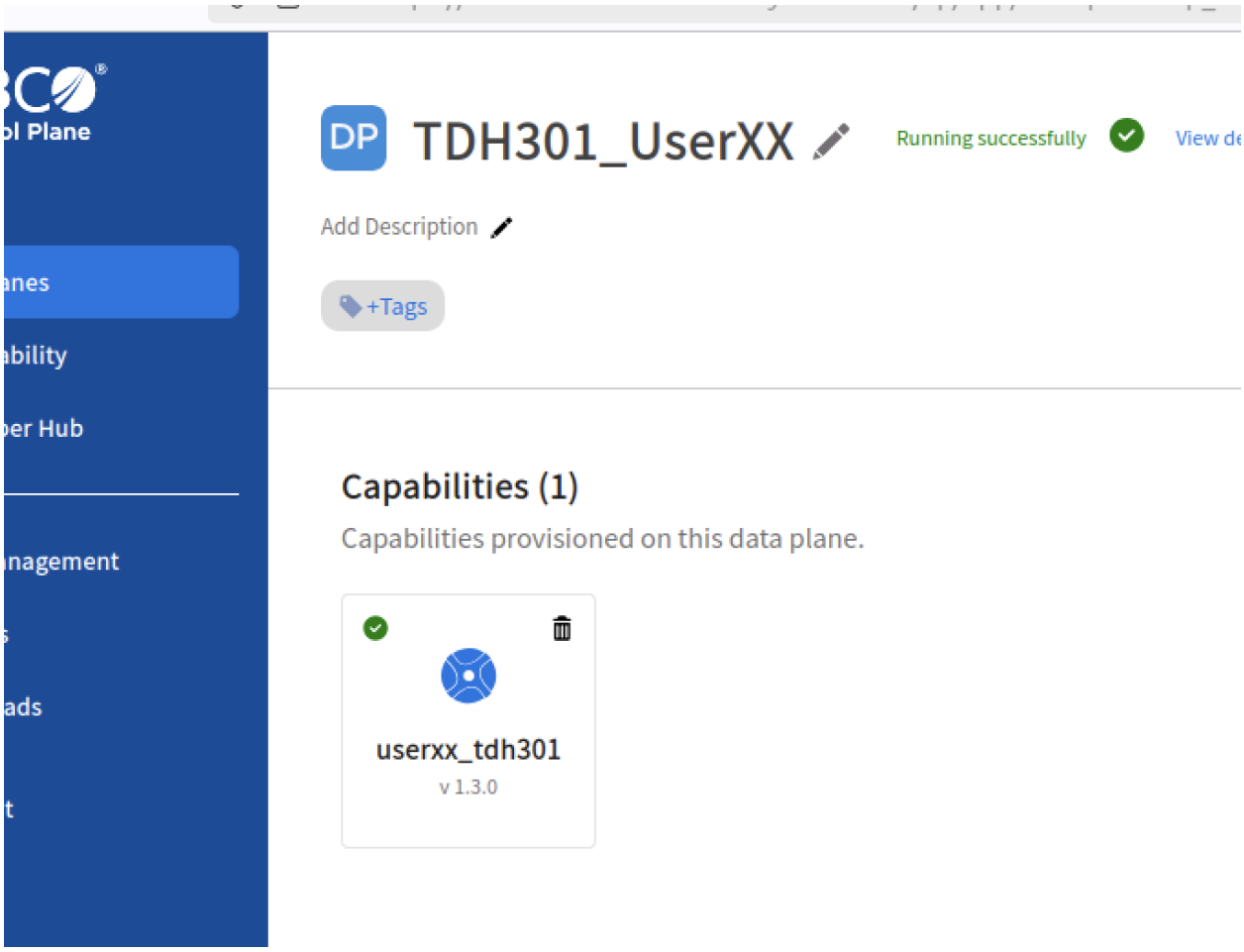


Then open a browser and navigate to https://edutraineval.eu-west.my.tibco.com/

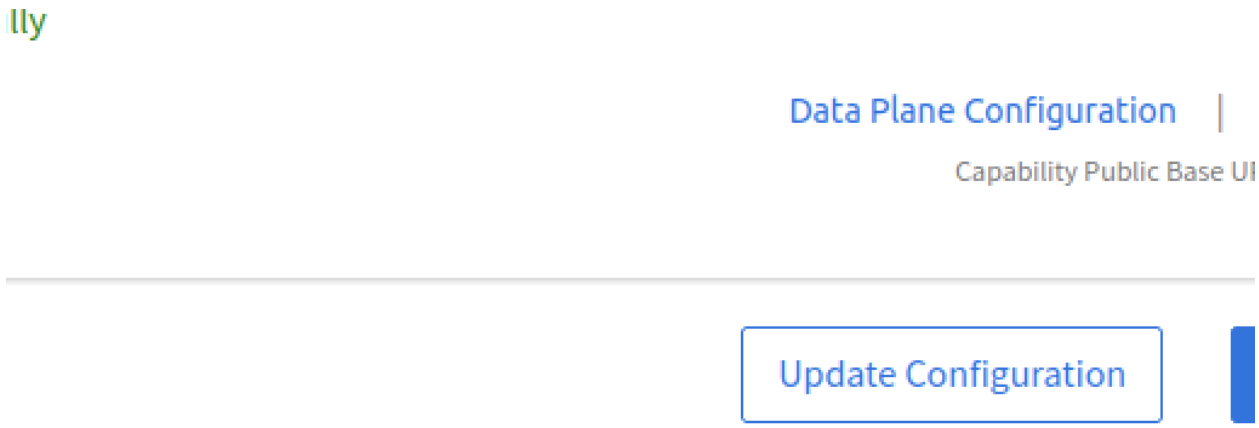
Login and then select your data plane:



Click on Developer HUB capability



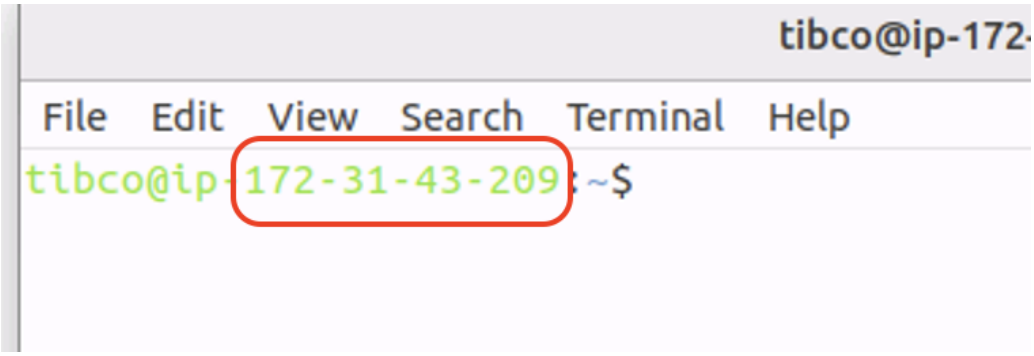
And click on Update Configuration



Then add the below lines

```
jenkins:
  baseUrl: http://<your-private-ip>:9090
  username: admin
  apiKey: 1156227f4deeec3f18bfa4b48ddb7e6ac0
catalog:
  locations: []
  rules:
    - allow:
        - Component
        - API
        - Location
        - Template
        - User
        - Group
        - Domain
        - System
        - Resource
```

In order to find your private ip and replace the placeholder <your-private-ip> above, you need to open a terminal, and take it from the prompt as highlighted below:



So in this case the private IP is 172.31.43.209

And then replace the placeholder <your-private-ip> with 172.31.43.209 and append the new configuration to the existing one as per screenshot below

```
app:
  baseUrl: https://devhub.local/tibco/hub
  title: userff99_tdh301
backend:
  baseUrl: https://devhub.local/tibco/hub
auth:
  providers:
    oauth2Proxy:
      development: {}
  enableAuthProviders:
    - oauth2Proxy
integrations:
  github:
    - host: github.com
      token: ${GITHUB_TOKEN}
jenkins:
  baseUrl: http://172.31.33.218:9090
  username: admin
  apiKey: 1156227f4deeec3f18bfa4b48ddb7e6ac0
catalog:
  locations: []
  rules:
    - allow:
        - Component
        - API
        - Location
        - Template
        - User
        - Group
        - Domain
        - System
        - Resource
```

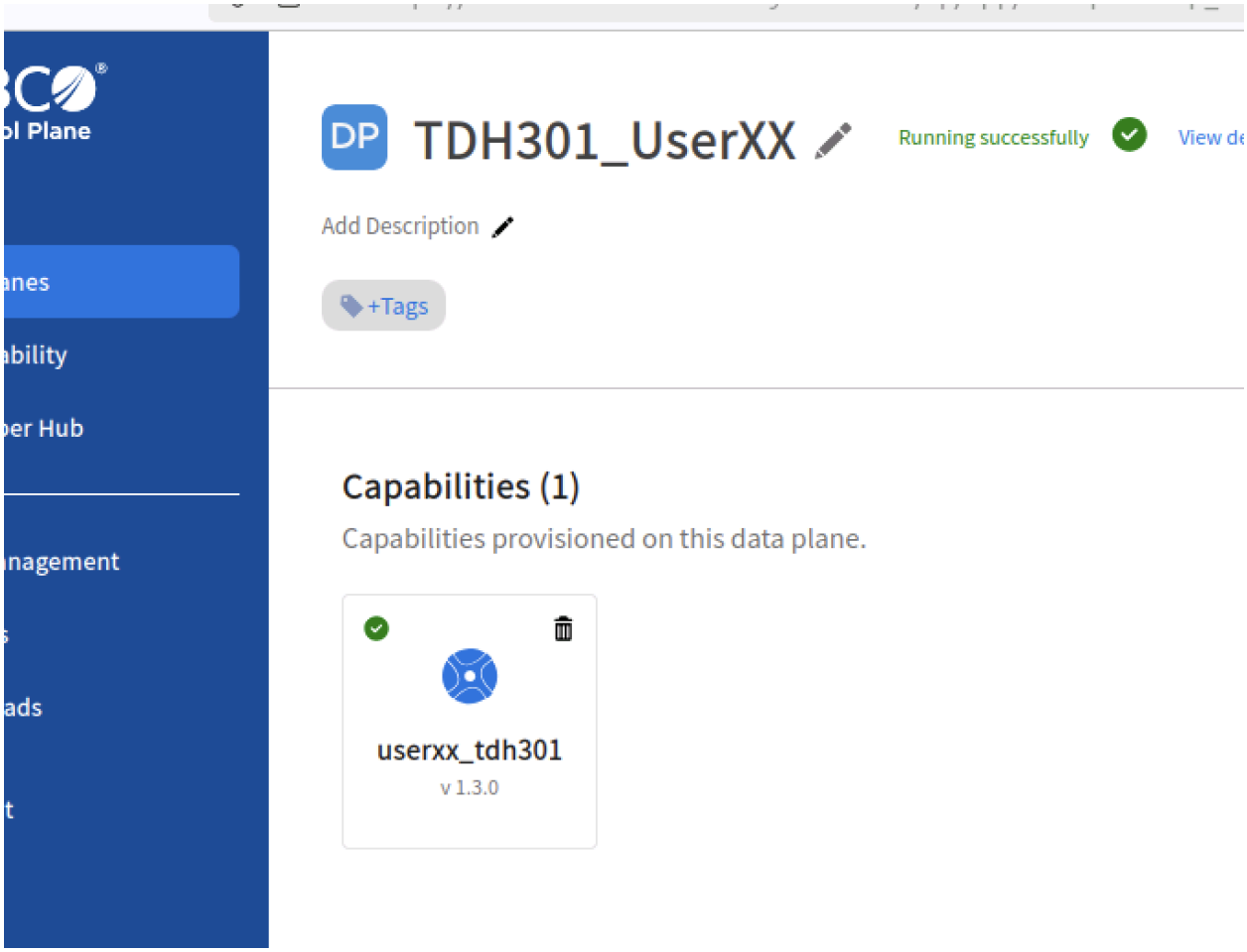
If you want to avoid any typo you can find the full yaml file to copy and paste at /home/tibco/TDH301/Other/copy_and_paste/DevHubJenkinsIntegrationSnippetCode.txt

And then click on Update Configuration.

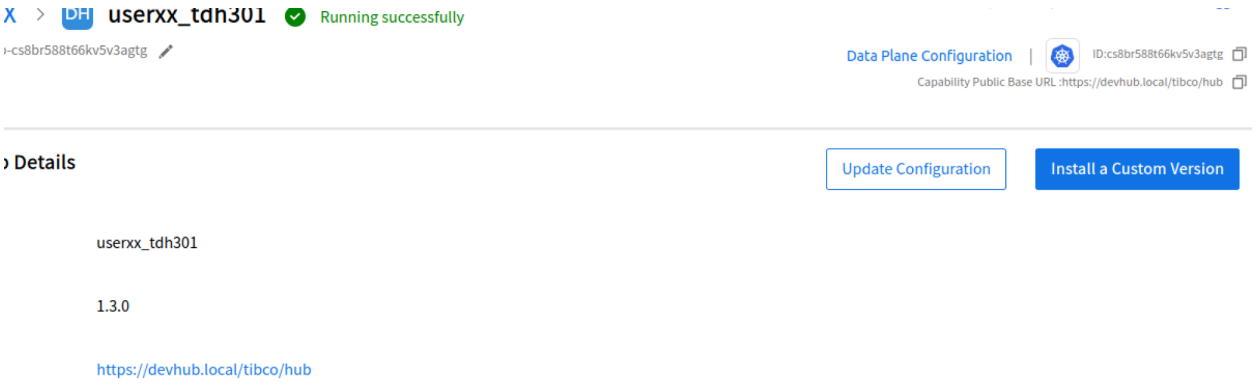
```
auth:
  providers:
    oauth2Proxy:
      development: {}
  enableAuthProviders:
    - oauth2Proxy
integrations:
  github:
    - host: github.com
      token: ${GITHUB_TOKEN}
jenkins:
  baseUrl: http://172.31.33.218:9090
  username: admin
  apiKey: 1156227f4deeec3f18bfa4b48ddb7e6ac0
catalog:
  locations: []
  rules:
    - allow:
        - Component
        - API
        - Location
        - Template
        - User
        - Group
        - Domain
        - System
        - Resource
```

Kubernetes Secret Object

After that go back to DevHub capability and



And then click on install a custom version



Then fill out the form as follows

Version name → tibcoedu

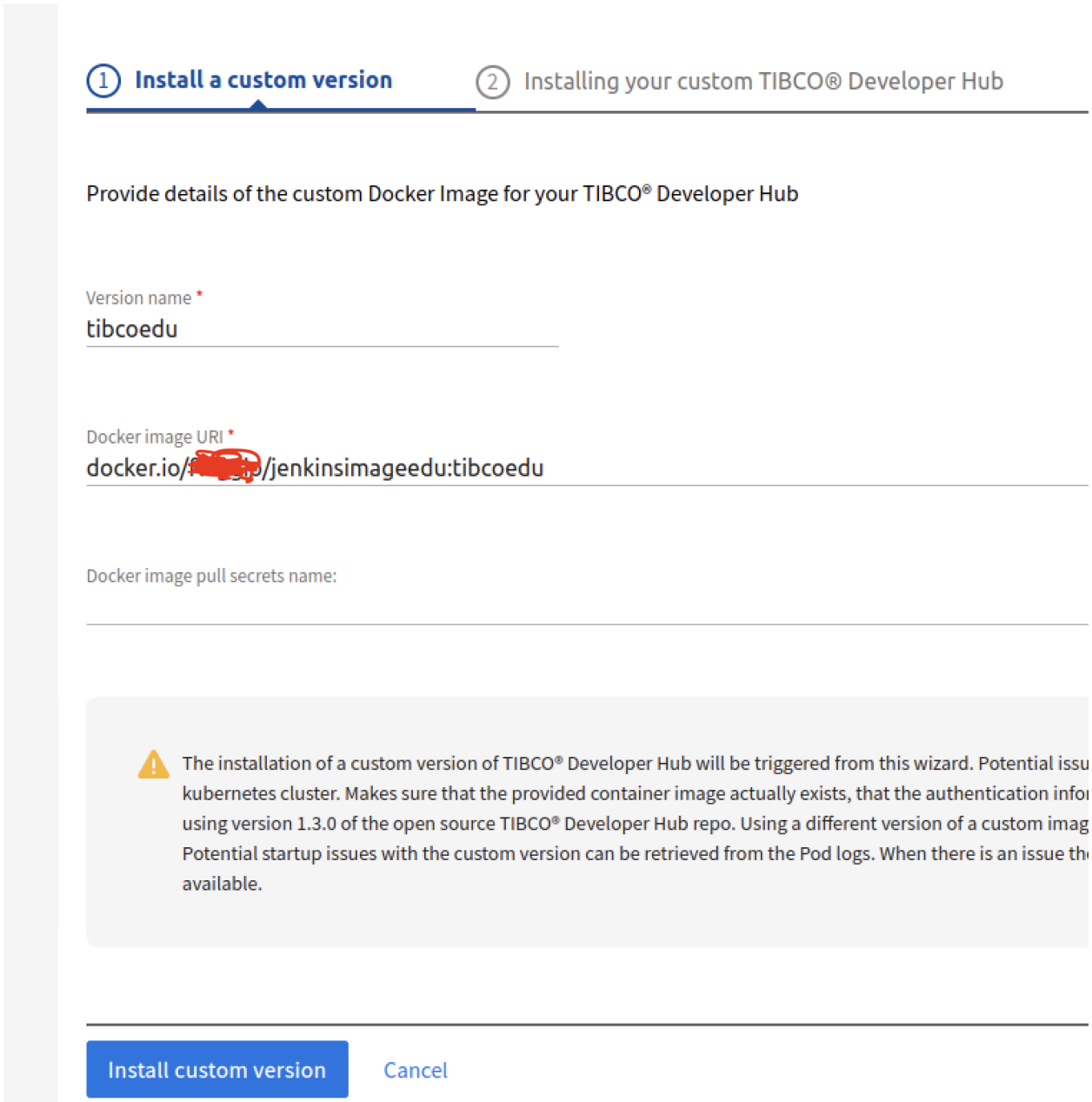
Docker image uri → docker.io/<your-docker-username>/jenkinsimageedu:tibcoedu

In case your repository is private then you need to provide also a kubernetes pull secret to authenticate with docker hub, so from a terminal type:

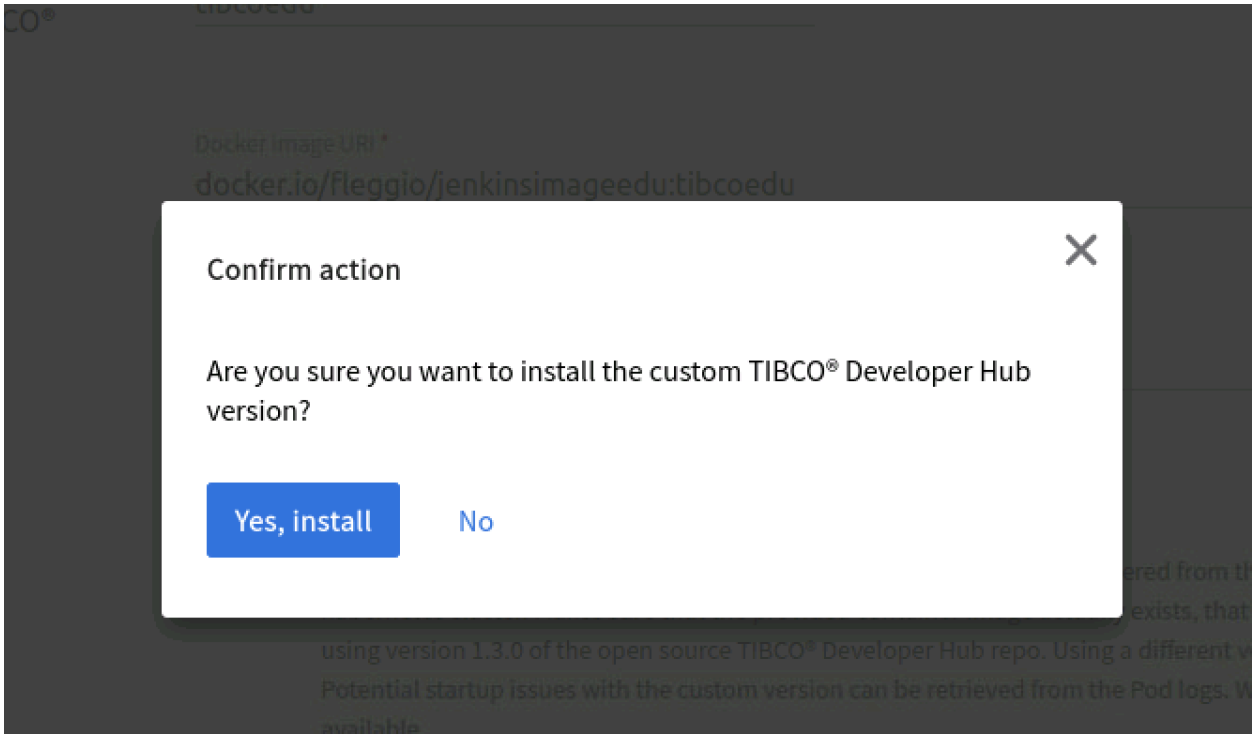
kubectl create secret docker-registry myregistrykey
--docker-username=**<your-docker-username>**
--docker-password=**<your-docker-password>** --docker-email=**<your-docker-email>** -n ns-tdh301-userxx

Once the secret is created, you can provide myregistrykey in the field for **Docker image pull secrets name**

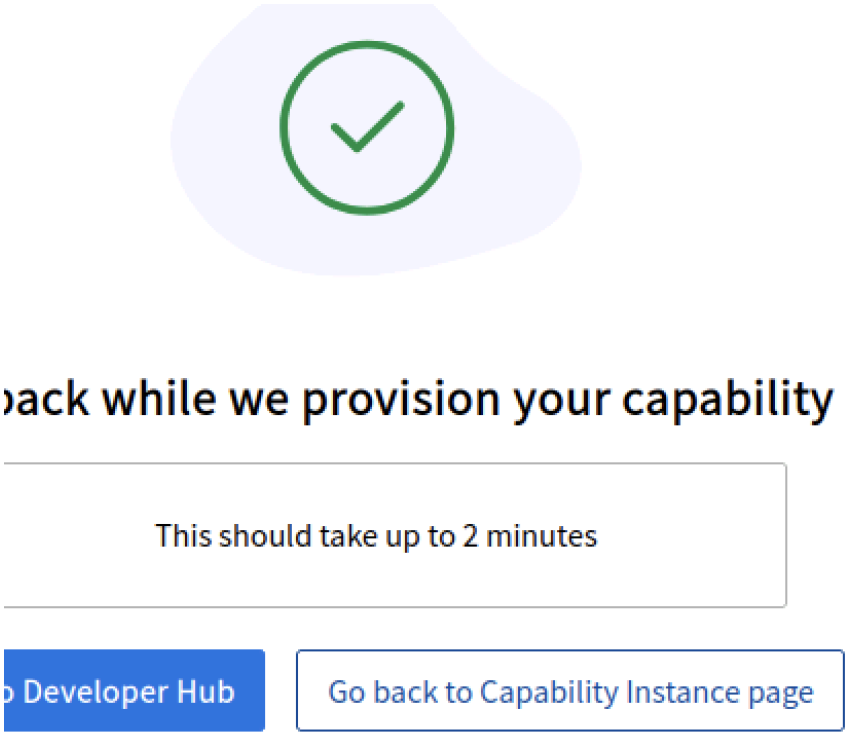
And click on Install Custom Version



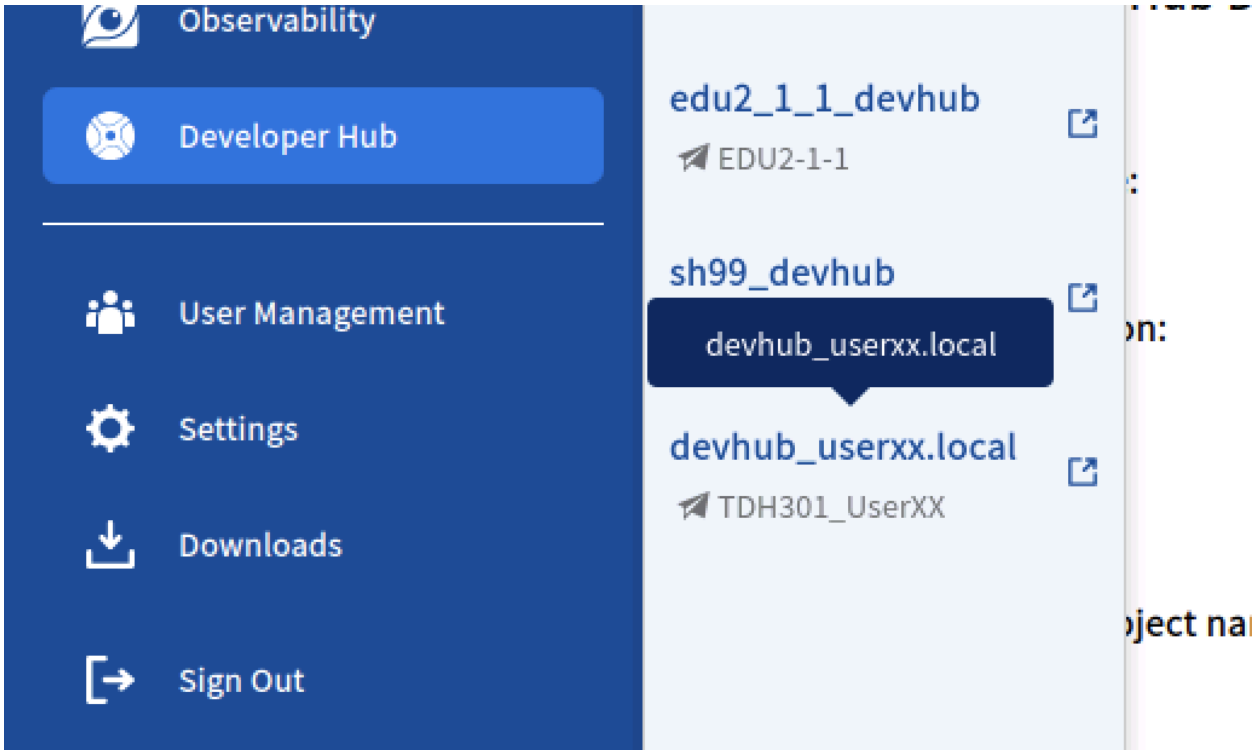
Click Yes, Install



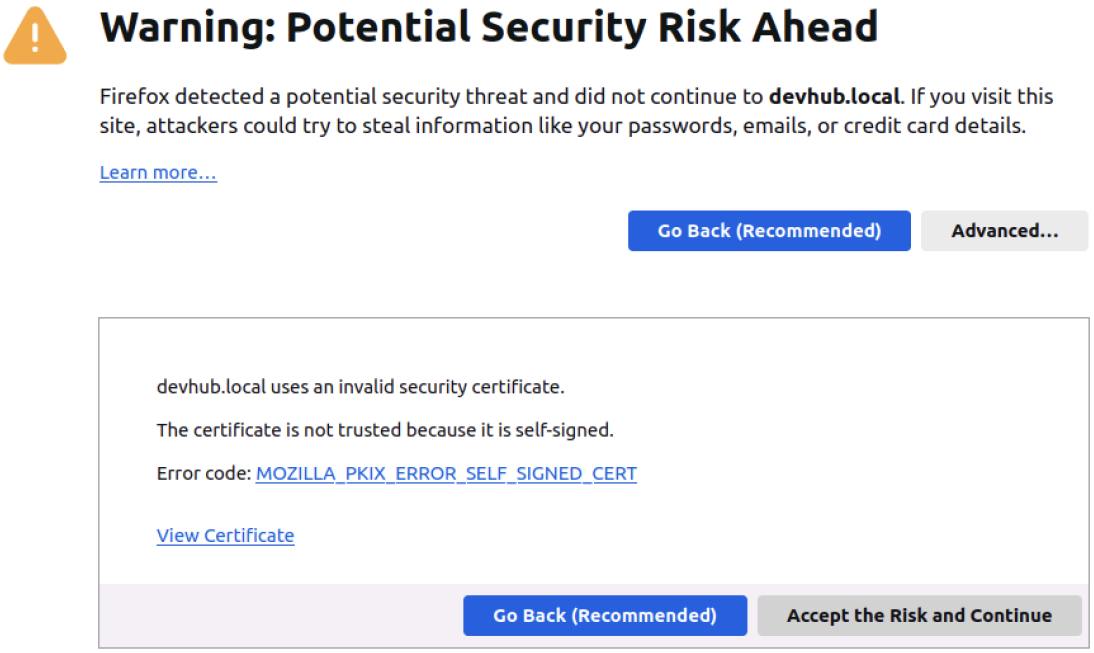
Then go back to Capability Instance page and wait til image is being installed



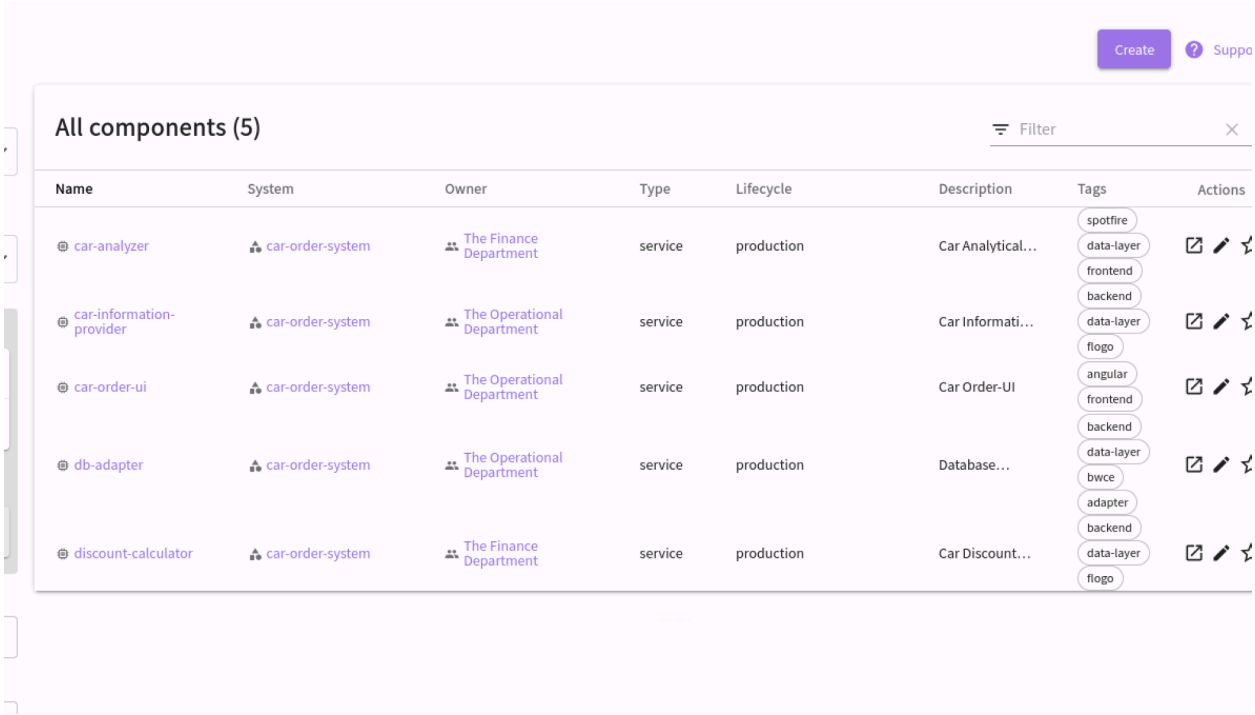
Then refresh the browser and select your developer hub instance



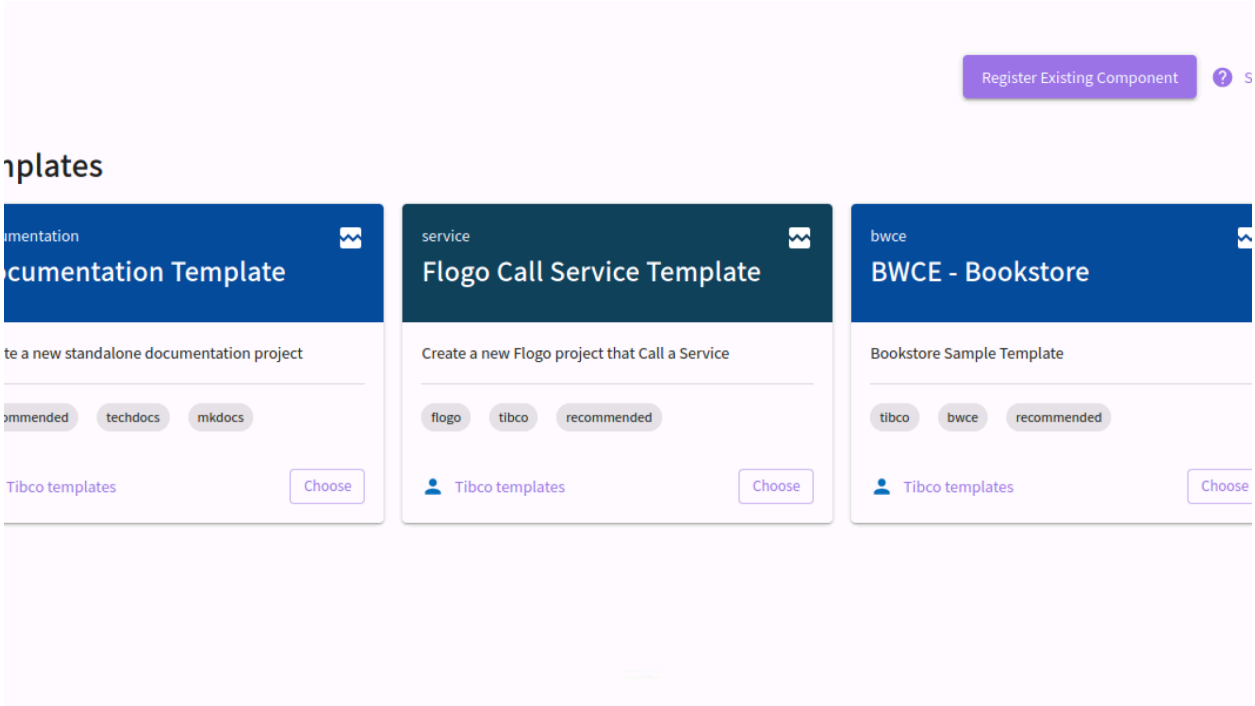
Accept the security risk



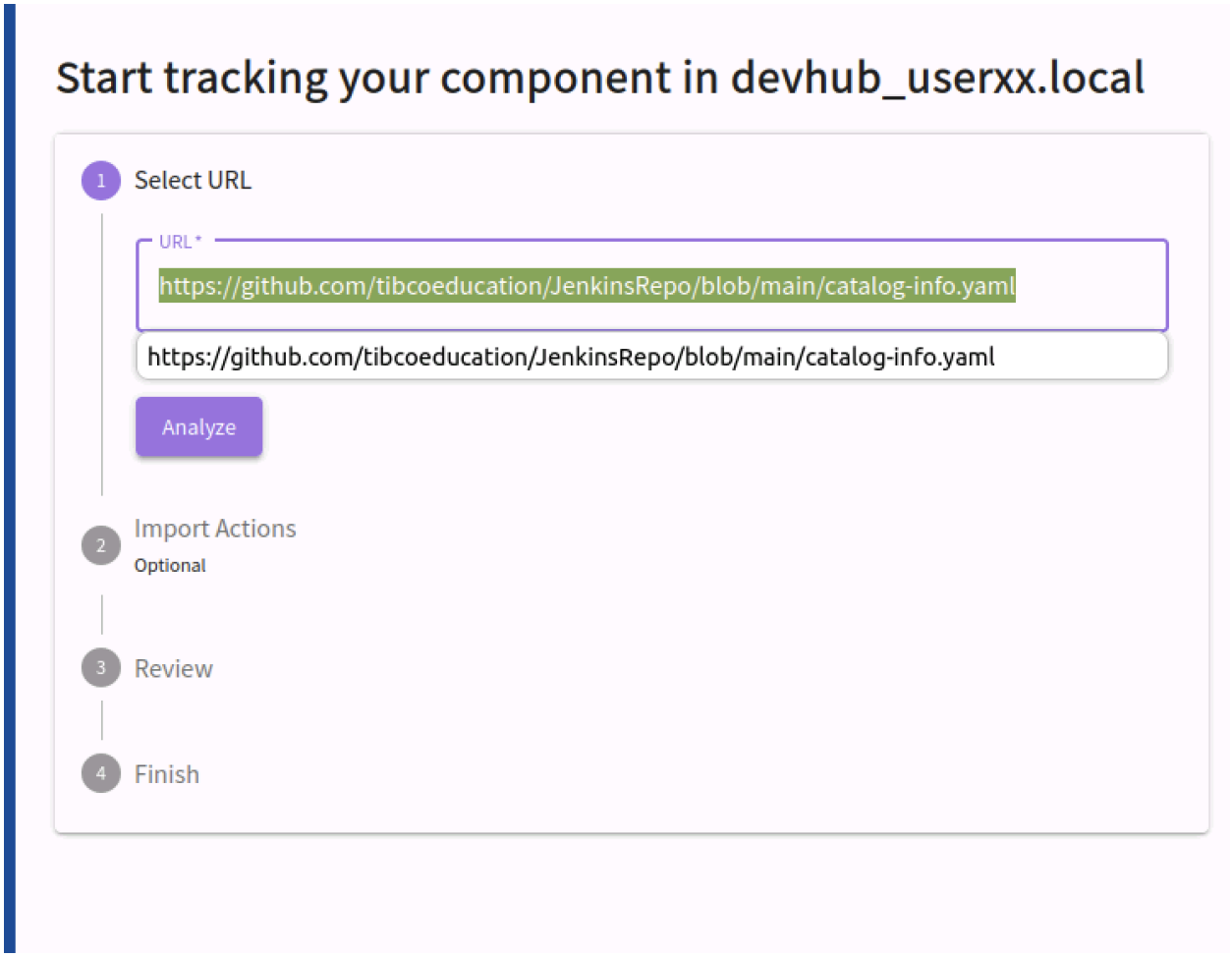
Click on the catalog, then on Create



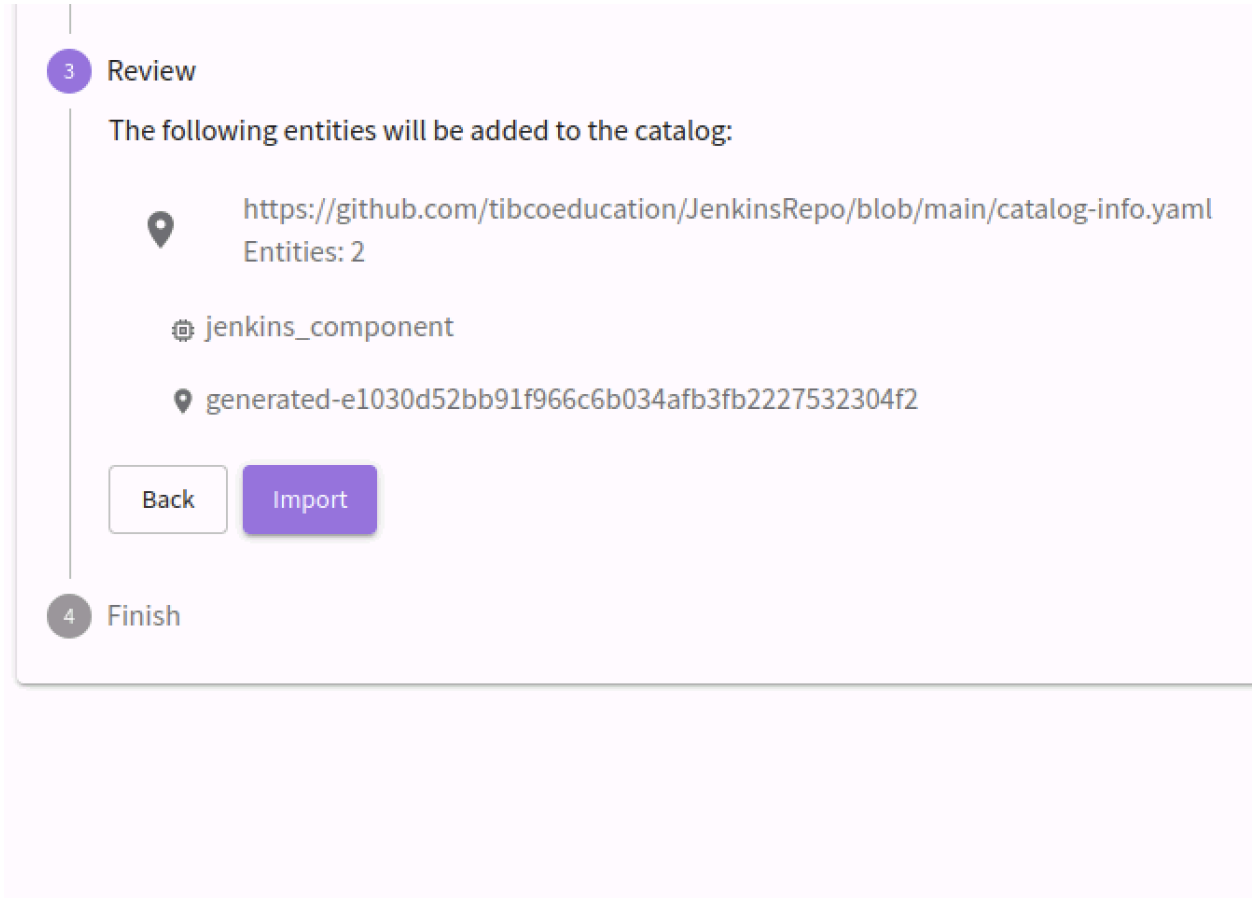
Then on register existing component



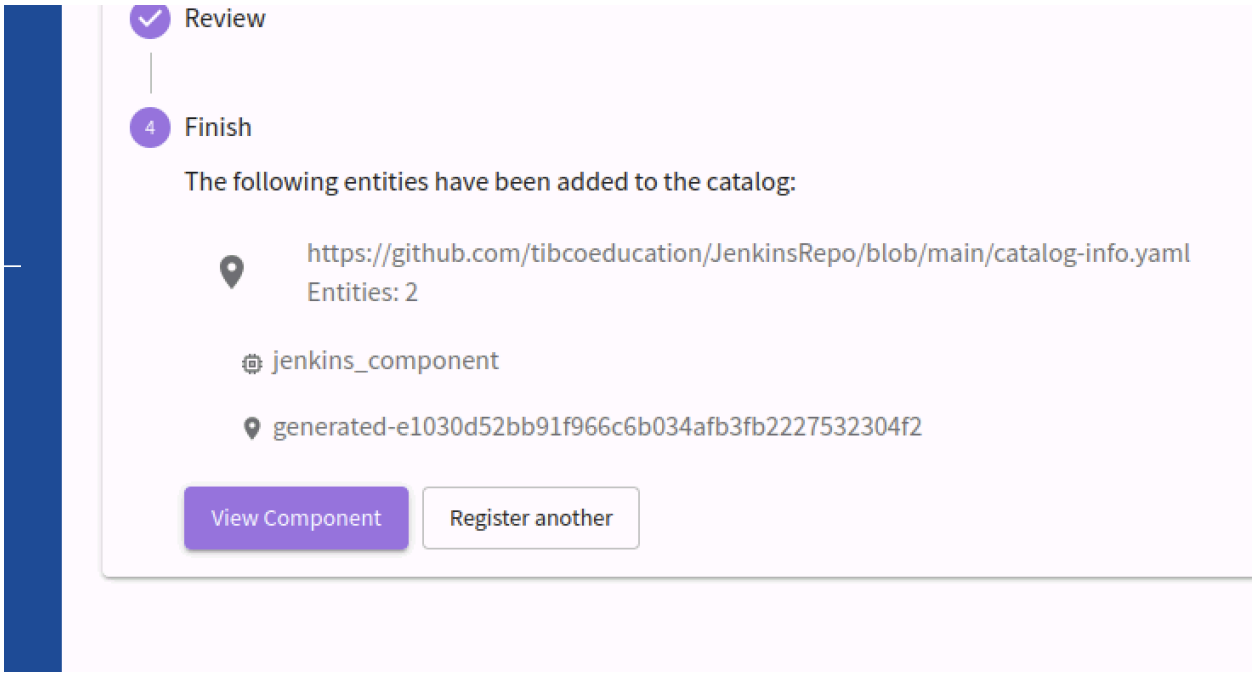
And provide https://github.com/tibcoeducation/JenkinsRepo/blob/main/catalog-info.yaml as url and click on Analyze



Then click on Import



Then click on view component



Then click on View component and all the Jenkins tab will be displayed in your custom plugin

Ex. 2 Replace Theme

Open a terminal and type the following commands

cd /home/tibco/tibco-developer-hub-main/packages/app/src/themes
cp tibcoThemeLight.ts myCustomTheme.ts
gedit myCustomTheme.ts &

It will open the file in gedit and you need to change the following code

Line 9

From

```
 9 export const tibcoThemeLight = createUnifiedTheme({
10   ...createBaseThemeOptions({
```

To

```
 9 export const myCustomTheme = createUnifiedTheme({
10   ...createBaseThemeOptions({
11     palette: {
```

Line 14

From

```
12       ...palettes.light,
13       primary: {
14         main: '#1774e5', // Changes inactive, clickable links, buttons, icons
15       },
16       secondary: {
17         main: '#565a6e',
```

To

```
12       ...palettes.light,
13       primary: {
14         main: '#9874e5', // Changes inactive, clickable links, buttons, icons
15       },
16       secondary: {
```

Save and close gedit.

To add a custom theme to your DeveloperHUB app, you pass it as configuration to createApp, in App.tsx. From the same terminal type

gedit ../App.tsx &

Then search for “tibcoThemeLight”, it should be around line 57
And change it from

```
56
57 import { tibcoThemeLight } from './themes/tibcoThemeLight';
```

To

```
56
57 import { myCustomTheme } from './themes/myCustomTheme';
```

Then around line 157
from

```
152       id: 'tibco-theme',
153       title: 'TIBCO Theme',
154       variant: 'light',
155       icon: <LightIcon />,
156       Provider: ({ children }) => (
157         <UnifiedThemeProvider theme={tibcoThemeLight} children={children} />
158       ),
```

To

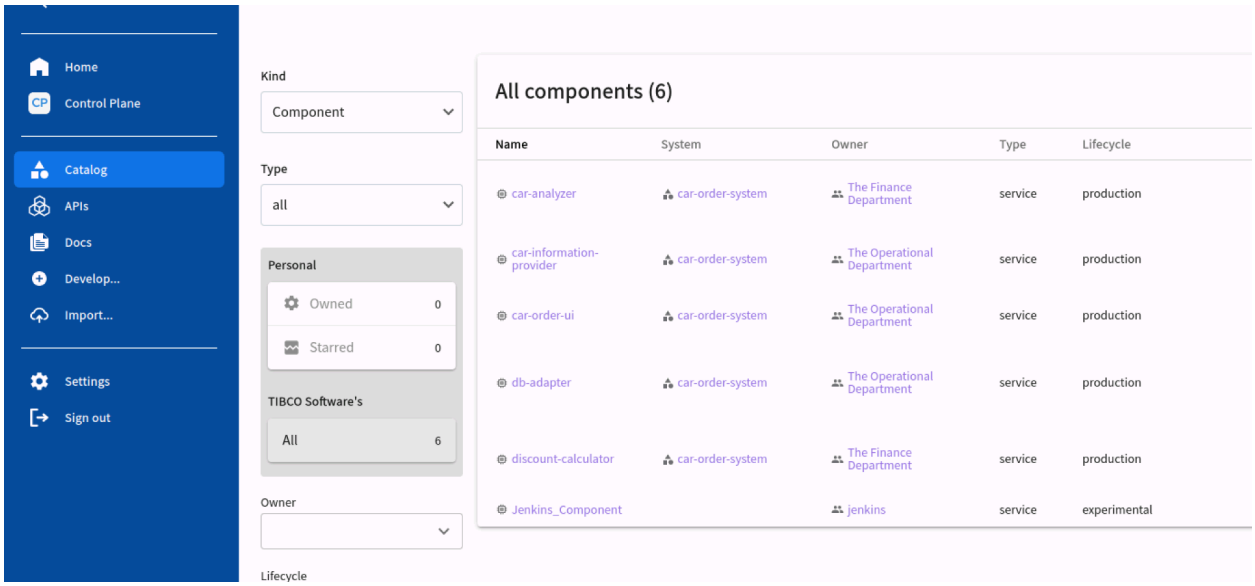
```
152       id: 'tibco-theme',
153       title: 'My Custom Theme',
154       variant: 'light',
155       icon: <LightIcon />,
156       Provider: ({ children }) => (
157         <UnifiedThemeProvider theme={myCustomTheme} children={children} />
158       ),
```

Then save the file and close gedit.

From the same terminal window type:

cd /home/tibco/TDH301/Other
./stop-devhub-services.sh
cd /home/tibco/tibco-developer-hub-main/docker
yarn dev

As it opens the browser go to the catalog and you will notice that the font color changed to purple



Shut down Developer HUB, from a terminal type:
cd /home/tibco/TDH301/Other
./stop-devhub-services.sh

Ex. 3 Change Icon and Logo

Open a terminal and type:

cd /home/tibco/tibco-developer-hub-main/packages/app
cp /home/tibco/TDH301/Images/newLogo.svg public/
gedit src/components/Root/Root.tsx &

Then around line 121 replace devhub-logo.svg with newLogo.svg

from

```
115              />
116            </div>
117          )}
118        >
119          <Link to="/">
120            <img
121              src="/tibco/hub/devhub-logo.svg"
122              className={classes.img}
123              alt="logo"
124            />
125          </Link>
126        </SidebarItem>
```

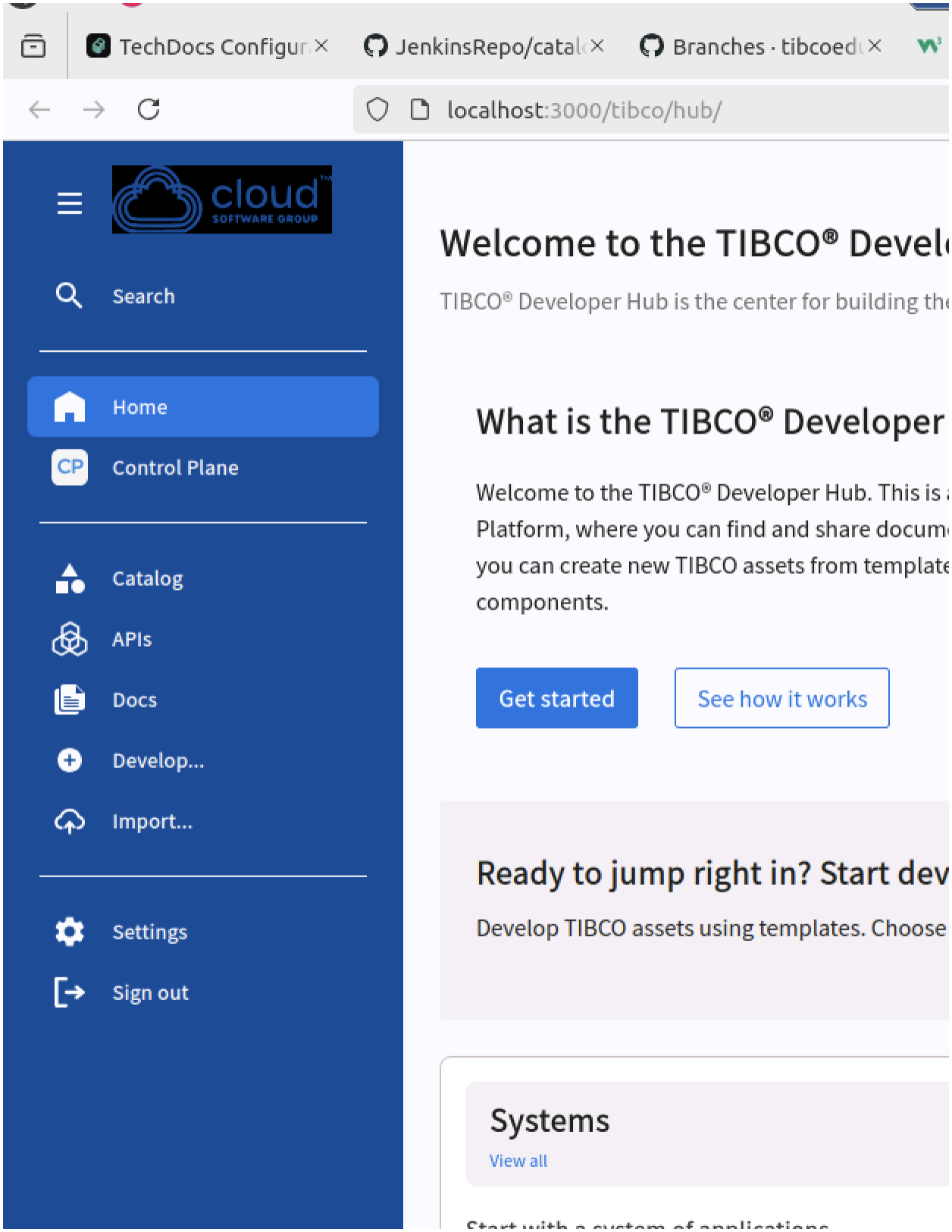
To

```
119          <Link to="/">
120            <img
121              src="/tibco/hub/newLogo.svg"
122              className={classes.img}
123              alt="logo"
```

Save and close the file, then from the same terminal open a new tab and type:
cd /home/tibco/tibco-developer-hub-main/docker
docker-compose up

Then open a new Tab from the same terminal and type:
yarn dev

It will open devhub portal on a browser and wait till it completes loading, it will display a different logo on the top left of the screen.


TechDocs Configur
JenkinsRepo/catal
Branches · tibcoed
localhost:3000/tibco/hub/
cloud
SOFTWARE GROUP
Search
Home
CP
Control Plane
Catalog
APIs
Docs
Develop...
Import...
Settings
Sign out
Welcome to the TIBCO® Devel
TIBCO® Developer Hub is the center for building th
What is the TIBCO® Developer
Welcome to the TIBCO® Developer Hub. This is
Platform, where you can find and share docum
you can create new TIBCO assets from templat
components.
Get started
See how it works
Ready to jump right in? Start dev
Develop TIBCO assets using templates. Choose
Systems
View all

From the same terminal open a new tab and shutdown developer hub by typing
cd /home/tibco/TDH301/Other
./stop-devhub-services.sh

Then close the terminal and open a new one and execute the following commands:

cd /home/tibco/tibco-developer-hub-main/packages/app
cp /home/tibco/TDH301/Images/newHome-header-img.svg src/components/home/images/
gedit src/components/home/HomePage.tsx &

Around line 20, change

```
19 import { Introduction } from './components/Introduction/Introduction'
20 import HomeHeaderImg from './images/home-image.svg';
21 import useAsync from 'react-use/lib/useAsync';
22 import { Content, Page } from '@backstage/core-components';
```

To

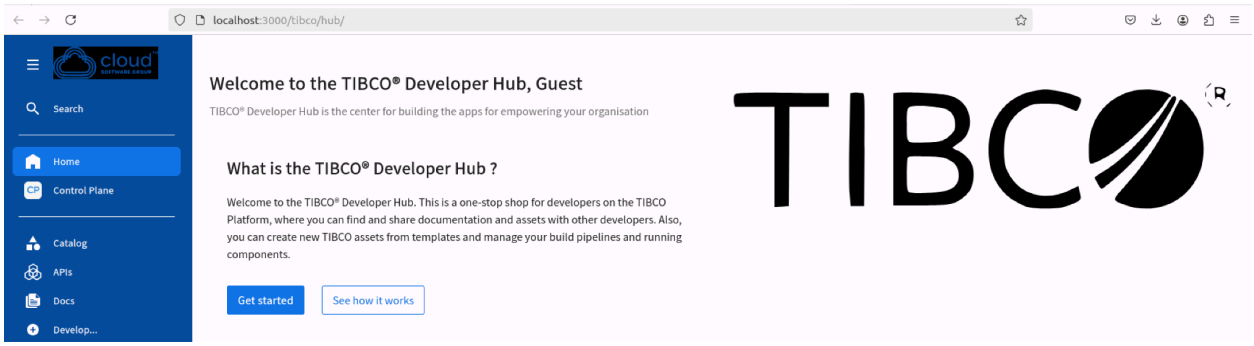
```
18 import { HomeCard } from './components/HomeCard/HomeCard';
19 import { Introduction } from './components/Introduction/Introduction';
20 import HomeHeaderImg from './images/newHome-header-img.svg';
21 import useAsync from 'react-use/lib/useAsync';
22 import { Content, Page } from '@backstage/core-components';
23
```

Then save and close gedit.

From the same terminal, execute the following commands:
cd /home/tibco/tibco-developer-hub-main/docker
docker-compose up

Then open a new tab and execute:
cd /home/tibco/tibco-developer-hub-main/docker
yarn dev

It will open the devhub portal on a browser and wait till it completes loading, it will display a different header on the top right of the screen.


localhost:3000/tibco/hub/
cloud
Search
Home
CP
Control Plane
Catalog
APIs
Docs
Develop...
Welcome to the TIBCO® Developer Hub, Guest
TIBCO® Developer Hub is the center for building the apps for empowering your organisation
What is the TIBCO® Developer Hub ?
Welcome to the TIBCO® Developer Hub. This is a one-stop shop for developers on the TIBCO Platform, where you can find and share documentation and assets with other developers. Also, you can create new TIBCO assets from templates and manage your build pipelines and running components.
Get started
See how it works
TIBCO