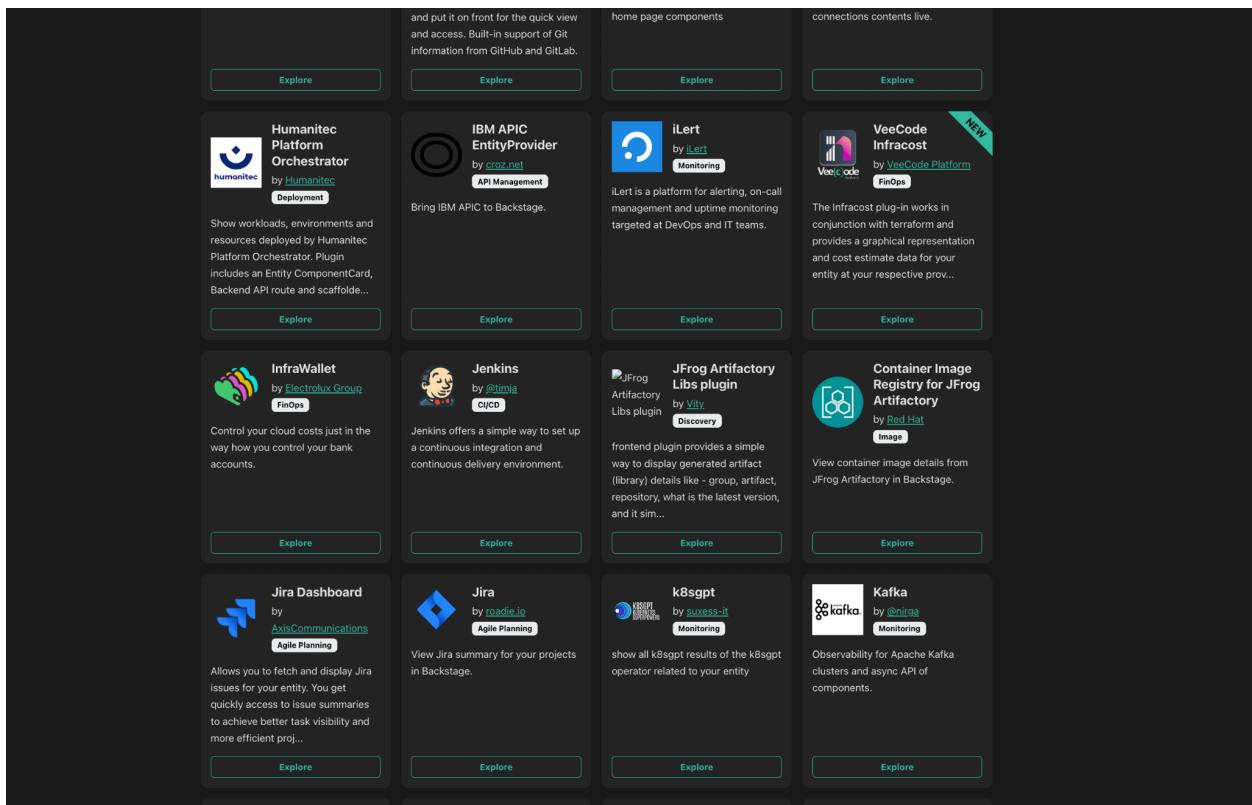


Lab F.

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 for Jenkins plugin and click on explore



You will land on a github page

The screenshot shows the Backstage application interface for the `community-plugins` workspace. The left sidebar contains a tree view of project files, including `.github`, `.husky`, `.yarn`, `docs`, `scripts`, and a large `workspaces` folder containing numerous subfolders like `3scale`, `acr`, `adr`, etc. The main content area shows a list of files under the `jenkins` directory. At the top of this list is a commit from `backstage-service and github-actions[bot]` titled "Version Packages (#1559)". Below this are other commits from various authors, such as "feat: migrate workspaces" and "fixed prettier issues". A section for `README.md` is present, followed by a prominent heading "Jenkins Plugin (Alpha)". Below the heading is a link to the [Website](https://jenkins.io/). A dark button labeled "Last master build" is visible.

Name	Last commit message
..	feat: migrate workspaces
dev	jenkins - version:bump to v1.32.0 (#1549)
src	fixed prettier issues
.eslintrc.js	Version Packages (#1559)
CHANGELOG.md	edits readme files
catalog-info.yaml	feat: migrate workspaces
knip-report.md	fixed prettier issues
package.json	Version Packages (#1559)
report-alpha.api.md	jenkins - version:bump to v1.32.0 (#1549)
report.api.md	jenkins - version:bump to v1.32.0 (#1549)

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/npm-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" has 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" has 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" has 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 > @backstage/test-utils > @backstage/core-app-api@1.15.1" has unmet peer dependency "react-dom@^16.13.1 || ^17.0.0 || ^18.0.0".
warning "@backstage-community/plugin-jenkins > @backstage/plugin-catalog-react > @backstage/frontend-test-utils > @backstage/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 jenkins 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 `overview` wherever you'd prefer):

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

So if you click on

<https://github.com/backstage/community-plugins/tree/main/workspaces/jenkins/plugins/jenkins-backend> you will land on

The screenshot shows a GitHub repository page for the Jenkins Backend Plugin. The repository path is `community-plugins / workspaces / jenkins / plugins / jenkins-backend`. The main interface includes a sidebar with navigation links like Code, Issues (70), Pull requests (30), Actions, Projects, Security, and Insights. The main content area displays a list of files and their commit history. Key files shown include `main`, `..`, `dev`, `src`, `.eslintrc.js`, `CHANGELOG.md`, `README.md`, `catalog-info.yaml`, `config.d.ts`, `knip-report.md`, `package.json`, and `report.api.md`. Pull requests listed include #1559 and #1549. At the bottom of the page, there is a section titled "Jenkins Plugin (Alpha)".

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  EntityConsumedApiCard
```

Then around line 183 you should see the below code:

```
101 },
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. You can do this by adding an annotation to your component's configuration file or by using the 'Annotations' tab in the component's details page."/>
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

```
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={isJenkinsAvailable}>
81 <EntityJenkinsContent />
82 </EntitySwitch.Case>
83 <EntitySwitch.Case if={isGithubActionsAvailable}>
84 <EntityGithubActionsContent />
85 </EntitySwitch.Case>
86
87 <EntitySwitch.Case>
88 <EmptyState />
```

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: "1156227f4deeeec3f18bfa4b48ddb7e6ac0" # Jenkins API token or password
```

As per screenshot below:

```

12 user: postgres
13 password: example
14
15 ## Uncomment the below github integrations config to add a PAT to try out
16 ## new component creation using one of the available templates.
17 integrations:
18 github:
19 - host: github.com
20 # # This is a Personal Access Token or PAT from GitHub. You can find out how to generate this token, and more info
21 # # about setting up the GitHub integration here: https://backstage.io/docs/getting-started/configuration#setting
22 token: ghp_BwzXhxY2NhHTr034rB0FJZZdg5p7Lm2vn5Ax #Enter your Github token here
23
24 # Jenkins integration added here
25 jenkins:
26 baseUrl: "http://localhost:9090" # Replace with your Jenkins server URL
27 username: "admin" # Jenkins username
28 apiKey: "1156227f4deec3f18bfa4b48ddb7e6ac0" # Jenkins API token or password
29 # Uncomment the below catalog config to add the default(example) entities to your software catalog while running developer
30 # The example entities had been placed inside the 'tibco-examples' folder under the project root folder
31 catalog:
32 locations:
33 - type: url
34 target: https://github.com/TIBCOSoftware/tibco-developer-hub/tree/main/tibco-examples/tibco-examples.yaml
35 rules:
36 - allow:
37 [
38 Component,
39 API,
40 Location,
41 Template,
42 User,
43 Group,
44 Domain,
45 System,
46 Resource,
47]
48 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

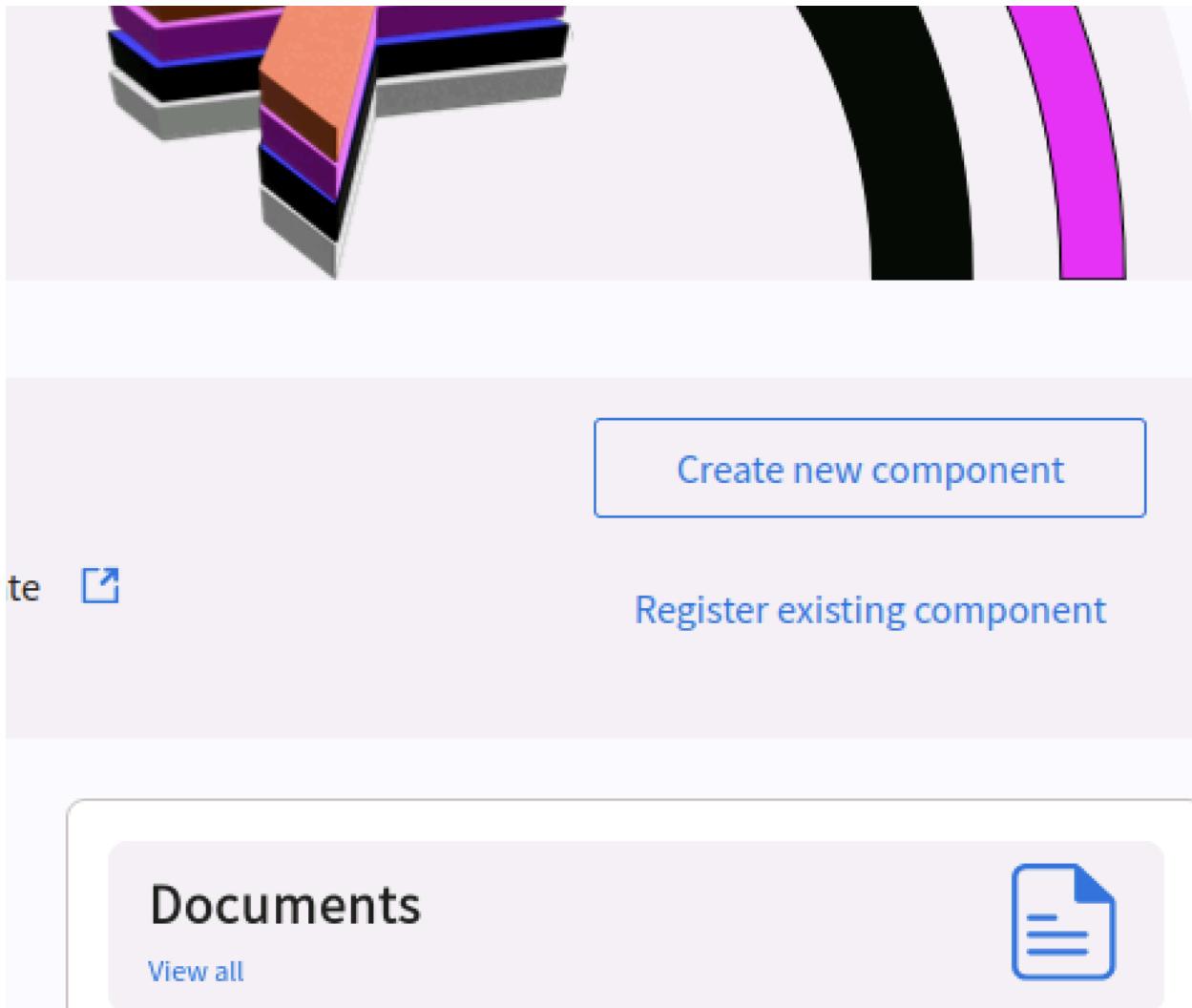
```

It will start developer hub with Jenkins plugin.

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

The screenshot shows the TIBCO Developer Hub homepage. The left sidebar has a dark blue background with white icons and text. It includes links for Home (which is highlighted in blue), Control Plane, Catalog, APIs, Docs, Develop..., Import..., Settings, and Sign out. The main content area has a light gray background. At the top, it says "Welcome to the TIBCO® Developer Hub, Guest". Below that, a sub-header reads "What is the TIBCO® Developer Hub ?". A paragraph explains the hub's purpose: "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." Two buttons are present: "Get started" and "See how it works". To the right of the main content, there are three decorative icons: a checkmark, a close X, and a circular progress bar. Below the main section, there are two cards: "Systems" and "Components". The "Systems" card features a blue icon of interconnected boxes and says "Start with a system of applications" followed by a link to "document-generation-system". The "Components" card features a blue icon of a Jenkins logo and says "Browse application components" followed by a link to "Jenkins\_Component". Both cards have a "View all" link at the bottom.

Click on Create a new component



Then click on Register Existing Component

The screenshot shows a catalog interface with two template cards:

- Documentation Template**:
  - Project type: standalone documentation project
  - Tags: techdocs, mkdocs
  - Author: Tibco
  - Action: Choose
- BWCE - Bookstore**:
  - Project type: Bookstore Sample Template
  - Tags: tibco, bwce, recommended
  - Author: Tibco templates
  - Action: Choose

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\*

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

## Start tracking your component in TIBCO® Developer Hub

1 Select URL

2 Select Locations

Discovered Locations: 1

3 Review

The following entities will be added to the catalog:



<https://github.com/tibcoeducation/JenkinsRepo/blob/main/catalog-info.yaml>

Entities: 2



Jenkins\_Component



[generated-e1030d52bb91f966c6b034afb3fb2227532304f2](#)

Back

Import

4 Finish

Click on Import

## Start tracking your component in TIBCO® Developer Hub

1 Select URL

2 Select Locations

Discovered Locations: 1

3 Review

4 Finish

The following entities have been added to the catalog:



<https://github.com/tibcoeducation/JenkinsRepo/blob/main/catalog-info.yaml>

Entities: 2

jenkins\_component

generated-e1030d52bb91f966c6b034afb3fb2227532304f2

[View Component](#)

[Register another](#)

Then click on View Component

COMPONENT — SERVICE

# Jenkins\_Component

OVERVIEW

CI/CD

API

DEPENDENCIES

DOCS

Latest main, master build

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

| Projects                 |                                    |       |                                                                                               |                   |                                                                                                                                                                                                                                                                   | Owner  Jenkins            | Lifecycle experimental | ⋮ |
|--------------------------|------------------------------------|-------|-----------------------------------------------------------------------------------------------|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------|------------------------|---|
| Source                   | Build                              | TESTS | Status                                                                                        | Last Run Duration | ACTIONS                                                                                                                                                                                                                                                           |                                                                                                                |                        |   |
| refs/remotes/origin/main | jenkinsFolder » JenkinsProjectName | n/a   |  Completed | 21.4 s            |    |                                                                                                                |                        |   |
| 5a7d3f1f                 |                                    |       |                                                                                               |                   |                                                                                                                                                                                                                                                                   | 5 rows   < < 1-1 of 1 > > |                        |   |

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 .
```

```
tibco@ip-172-31-12-237:~/tibco-developer-hub-main$ docker build --no-cache -t fleggio/jenkinsimageedu:tibcoedu .
[+] Building 239.1s (22/33)
[+] Building 239.5s (22/33)
[+] Building 239.8s (22/33)
[+] Building 433.0s (28/33)
[+] Building 498.9s (34/34) FINISHED
=> [internal] load build definition from Dockerfile
=> transferring Dockerfile: 3.50kB
=> WARN: FromPlatformFlagConstIsNotAllowed: FROM --platform flag should not use constant value "linux/amd64" (line 43)
=> [internal] load metadata for docker.io/chainguard/wolfi-base:latest
=> [internal] load metadata for docker.io/library/node:18.20-alpine3.19
=> [internal] load dockerignore
=> transferring context: 251B
=> [internal] load build context
=> transferring context: 24.81kB
=> CACHED [internal] setting cache mount permissions
=> CACHED [stage-2 1/13] FROM docker.io/chainguard/wolfi-base:latest@sha256:3221f96f52fc0a020fa6f40b0370d132403be6b3736d8dd92275ccd72129c1f
=> CACHED [packages 1/6] FROM docker.io/library/node:18.20-alpine3.19
=> CACHED [packages 2/6] WORKDIR /app
=> [build 2/9] RUN --mount=type=cache,target=/var/cache/apk,sharing=locked apk update && apk add python3 g++ make &&
=> [stage-2 2/13] RUN apk update && apk add nodejs-18-20.yarn
=> [packages 3/6] COPY package.json yarn.lock ./
=> [packages 4/6] COPY packages packages
=> [packages 5/6] COPY plugins plugins
=> [packages 6/6] RUN find packages ! -name "package.json" -mindepth 2 -maxdepth 2 -exec rm -rf {} +
=> [stage-2 3/13] RUN --mount=type=cache,target=/var/cache/apk,sharing=locked apk update && apk add python-3.12=-3
=> [build 3/9] WORKDIR /
=> [build 4/9] COPY --from=packages --chown=node:node /app
=> [build 5/9] RUN --mount=type=cache,target=/home/node/.cache/yarn,sharing=locked,uid=1000,gid=1000 yarn install --frozen-lockfile --network-timeout 600000
=> [stage-2 4/13] RUN python3 -m venv /opt/yenv
=> [stage-2 5/13] RUN pip3 install setuptools
=> [stage-2 6/13] RUN pip3 install mkdocs-techdocs-core==1.3.3
=> [stage-2 7/13] WORKDIR /app
=> [build 6/9] COPY --chown=node:node .
=> [build 7/9] RUN yarn tsc
=> [build 8/9] RUN yarn - cwd packages/backend build
=> [build 9/9] RUN mkdir packages/backend/dist/skeleton packages/backend/dist/bundle && tar xzf packages/backend/dist/skeleton.tar.gz -C packages/backend/dist/skeleton && tar
=> [stage-2 8/13] COPY --from=build /app/yarn.lock /app/package.json /app/packages/backend/dist/skeleton/ .
=> [stage-2 9/13] RUN --mount=type=cache,target=/home/node/.cache/yarn,sharing=locked,uid=1000,gid=1000 yarn install --frozen-lockfile --production --network-timeout 600000
=> [stage-2 10/13] COPY --from=build /app/packages/backend/dist/bundle/ .
=> [stage-2 11/13] COPY app-config.yaml app-config.production.yaml ./
=> [stage-2 12/13] COPY LICENSE.TXT /opt/tibco/license/
=> [stage-2 13/13] RUN chmod -R 777 /app/node_modules/@backstage/plugin-techdocs-backend
=> exporting to image
=> exporting layers
=> writing image sha256:542a96b9a0636783defb1dca4e53912344f471ccf63620ef1b39bf5fc79b5c61
=> naming to docker.io/fleggio/jenkinsimageedu:tibcoedu
tibco@ip-172-31-12-237:~/tibco-developer-hub-main$
```

From the same terminal execute

docker images

And verify that image size is around 3.2 GB

```
tibco@ip-172-31-12-237:~/tibco-developer-hub-main$ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
fleggio/jenkinsimageedu tibcoedu 542a96b9a063 2 minutes ago 3.22GB
gcr.io/k8s-minikube/kicbase v0.0.45 aeed0e1d4642 7 weeks ago 1.28GB
postgres latest b781f3a53e61 2 months ago 432MB
adminer latest 2f7580903a1d 3 months ago 250MB
```

Then type:

docker logout

docker login -u <your-docker-username>

```
tibco@ip-172-31-12-237:~/tibco-developer-hub-main$ docker login
Log in with your Docker ID or email address to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com/ to create one.
You can log in with your password or a Personal Access Token (PAT). Using a limited-scope PAT grants better security and is required for organizations using SSO. Learn more at https://docs.docker.com/go/access-tokens/
```

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
```

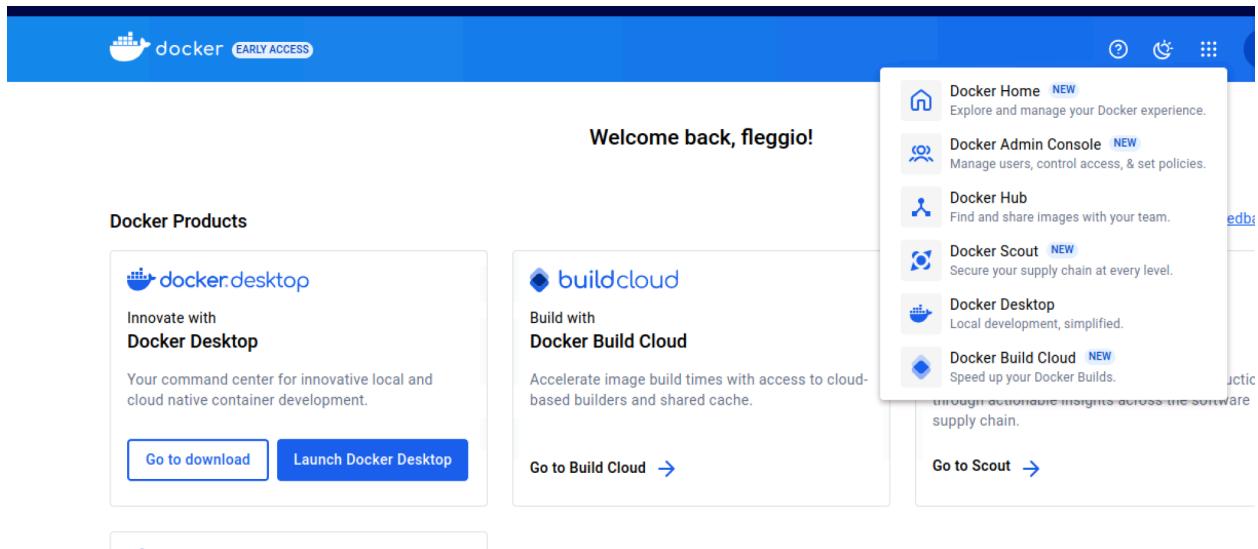
```
Login Succeeded
```

Then push the image to your docker hub

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

```
tibco@ip-172-31-12-237:~/tibco-developer-hub
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:b6c87f89b9fd88ff58.
```

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



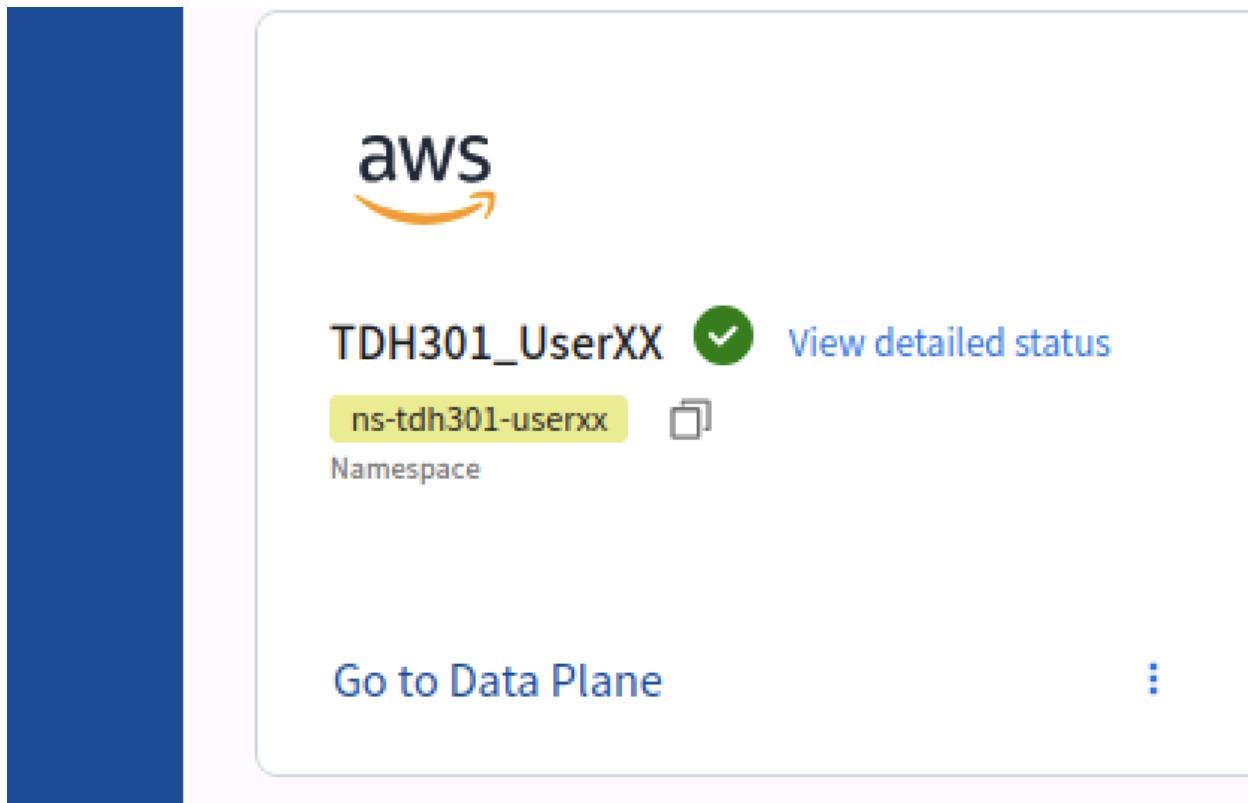
It should display the image you have just uploaded

The screenshot shows the Docker Hub interface. At the top, there are navigation tabs: 'kerhub' (highlighted in blue), 'Explore', 'Repositories' (underlined in white), 'Organizations', and 'Usage'. Below the tabs is a search bar with a magnifying glass icon and the placeholder 'Search by repository name'. To the right of the search bar is a button labeled 'All Con'. The main content area displays a repository card for 'jenkinsimageedu':

/ jenkinsimageedu  
: Image • Last pushed: 4 minutes ago

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

The screenshot shows the SD-Cloud Control Plane interface. On the left, there is a sidebar with various navigation options: Data Planes, Capability, Operator Hub, Management, Services, and Assets. The 'Data Planes' option is highlighted with a blue bar. The main content area is titled 'DP TDH301\_UserXX'. It includes a 'Running successfully' status indicator with a green checkmark and a 'View details' link. Below this, there are buttons for 'Add Description' and '+Tags'. A section titled 'Capabilities (1)' displays a single capability entry: 'userxx\_tdh301 v 1.3.0'. This entry features a circular icon with a checkmark, a trash can icon for deletion, and the capability name and version.

And click on Update Configuration

[Update Configuration](#)

Then add the below lines

```
jenkins:
 baseUrl: http://<your-private-ip>:9090
 username: admin
 apiKey: 1156227f4deec3f18bfa4b48ddb7e6ac0
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:

```
tibco@ip-172-31-43-209:~$
```

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

```
1 app:
2 baseUrl: https://devhub.local/tibco/hub
3 title: userff99_tdh301
4 backend:
5 baseUrl: https://devhub.local/tibco/hub
6 auth:
7 providers:
8 oauth2Proxy:
9 | development: []
10 enableAuthProviders:
11 - oauth2Proxy
12 integrations:
13 github:
14 | host: github.com
15 | token: ${GITHUB_TOKEN}
16 jenkins:
17 baseUrl: http://172.31.33.218:9090
18 username: admin
19 apiKey: 1156227f4deeeec3f18bfa4b48ddb7e6ac0
20 catalog:
21 locations: []
22 rules:
23 allow:
24 - Component
25 - API
26 - Location
27 - Template
28 - User
29 - Group
30 - Domain
31 - System
32 - Resource
33 |
```

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.

```
6 auth:
7 providers:
8 oauth2Proxy:
9 development: {}
10 enableAuthProviders:
11 - oauth2Proxy
12 integrations:
13 github:
14 - host: github.com
15 | token: ${GITHUB_TOKEN}
16 jenkins:
17 baseUrl: http://172.31.33.218:9090
18 username: admin
19 apiKey: 1156227f4deec3f18bfa4b48ddb7e6ac0
20 catalog:
21 locations: []
22 rules:
23 - allow:
24 - Component
25 - API
26 - Location
27 - Template
28 - User
29 - Group
30 - Domain
31 - System
32 - Resource
33
```

Kubernetes Secret Object

After that go back to DevHub capability and

The screenshot shows the Tibco Data Plane interface. On the left, there's a sidebar with various navigation items like 'Data Planes', 'Capability', 'Data Hub', 'Management', 'Logs', 'Metrics', and 'Logs'. The main area is titled 'DP TDH301\_UserXX' with a pencil icon. It shows a status of 'Running successfully' with a green checkmark and a link to 'View details'. Below this, there's a section for 'Add Description' with a pencil icon and a 'Tags' button. A large section titled 'Capabilities (1)' is shown, stating 'Capabilities provisioned on this data plane.' It lists one item: 'userxx\_tdh301 v 1.3.0'. This item has a green checkmark icon and a trash bin icon.

And then click on install a custom version

The screenshot shows the configuration page for 'userxx\_tdh301'. At the top, it says 'X > DH userxx\_tdh301' with a green checkmark and 'Running successfully'. Below this are 'Data Plane Configuration' and 'ID: cs8br588t66kv5v3agtg'. There's also a note about the 'Capability Public Base URL: https://devhub.local/tibco/hub'. At the bottom, there are two buttons: 'Update Configuration' and 'Install a Custom Version'. The 'Install a Custom Version' button is highlighted in blue. Below these buttons, there's some text: 'userxx\_tdh301' and '1.3.0' followed by a link 'https://devhub.local/tibco/hub'.

Then fill out the form as follows

Version name → tibcoedu

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

① **Install a custom version**

② Installing your custom TIBCO® Developer Hub

Provide details of the custom Docker Image for your TIBCO® Developer Hub

Version name \*

tibcoedu

Docker image URI \*

docker.io/tibco/jenkinsimageedu:tibcoedu

Docker image pull secrets name:

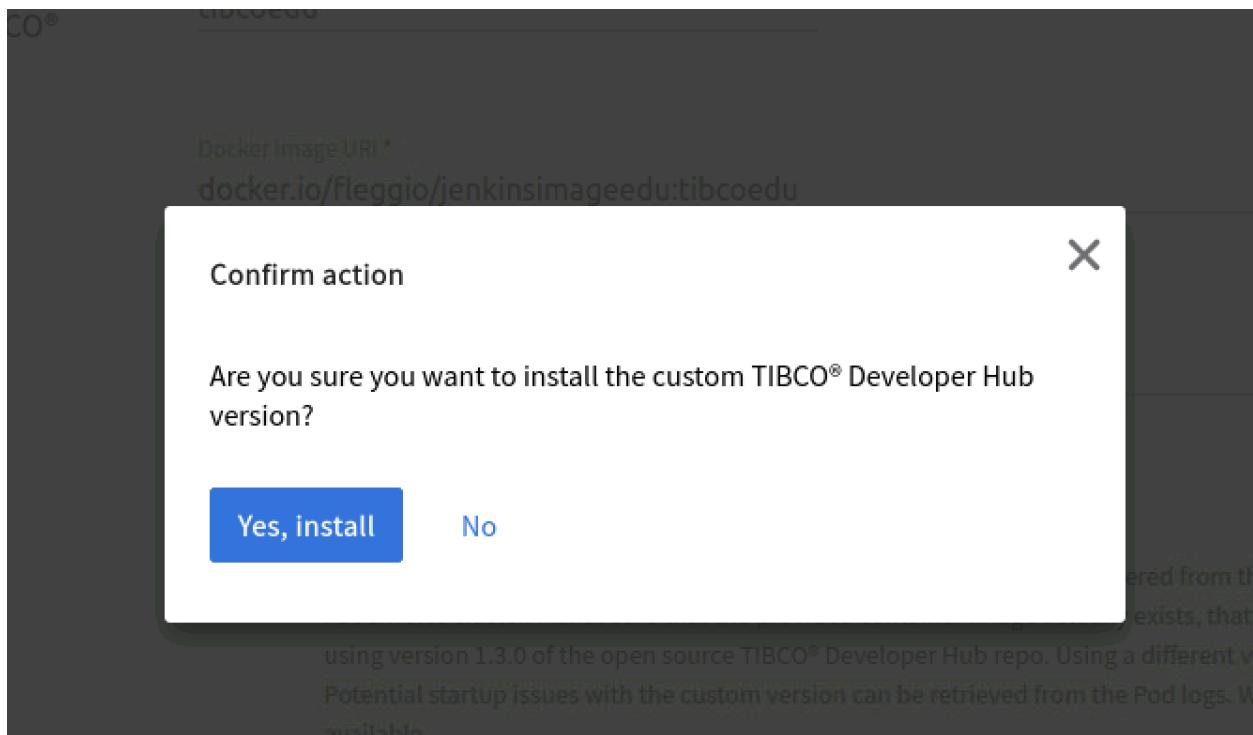


The installation of a custom version of TIBCO® Developer Hub will be triggered from this wizard. Potential issues may arise if the provided container image does not exist or if there are authentication issues with the specified Kubernetes cluster. Makes sure that the provided container image actually exists, that the authentication information is correct, and that the Kubernetes cluster supports the required features. Using a different version of the open source TIBCO® Developer Hub repo. Using a different version of a custom image may result in compatibility issues. Potential startup issues with the custom version can be retrieved from the Pod logs. When there is an issue, refer to the documentation for troubleshooting steps.

**Install custom version**

[Cancel](#)

Click Yes, Install



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



Check while we provision your capability

This should take up to 2 minutes

[Go Developer Hub](#)

[Go back to Capability Instance page](#)

Then refresh the browser and select your developer hub instance

The screenshot shows the Observability Developer Hub interface. On the left, there's a sidebar with icons for Developer Hub, User Management, Settings, Downloads, and Sign Out. The main area displays a list of environments:

- edu2\_1\_1\_devhub (with a copy icon)
- sh99\_devhub (with a copy icon)
  - devhub\_userxx.local (highlighted with a dark blue background)
- devhub\_userxx.local (with a copy icon)
  - TDH301\_UserXX (with a copy icon)

Accept the security risk



## Warning: Potential Security Risk Ahead

Firefox detected a potential security threat and did not continue to **devhub.local**. If you visit this site, attackers could try to steal information like your passwords, emails, or credit card details.

[Learn more...](#)

[Go Back \(Recommended\)](#)

[Advanced...](#)

devhub.local uses an invalid security certificate.

The certificate is not trusted because it is self-signed.

Error code: [MOZILLA\\_PKIX\\_ERROR\\_SELF\\_SIGNED\\_CERT](#)

[View Certificate](#)

[Go Back \(Recommended\)](#)

[Accept the Risk and Continue](#)

Click on the catalog, then on Create

The screenshot shows a component catalog titled "All components (5)". The table has columns for Name, System, Owner, Type, Lifecycle, Description, Tags, and Actions. The components listed are:

| Name                     | System           | Owner                      | Type    | Lifecycle  | Description       | Tags                                    | Actions |
|--------------------------|------------------|----------------------------|---------|------------|-------------------|-----------------------------------------|---------|
| car-analyzer             | car-order-system | The Finance Department     | service | production | Car Analytical... | spotfire, data-layer, frontend, backend |         |
| car-information-provider | car-order-system | The Operational Department | service | production | Car Informati...  | data-layer, flogo                       |         |
| car-order-ui             | car-order-system | The Operational Department | service | production | Car Order-UI      | angular, frontend, backend              |         |
| db-adapter               | car-order-system | The Operational Department | service | production | Database...       | data-layer, bwce, adapter, backend      |         |
| discount-calculator      | car-order-system | The Finance Department     | service | production | Car Discount...   | data-layer, flogo                       |         |

Then on register existing component

The screenshot shows a registration interface with three available templates:

- Documentation Template**: Create a new standalone documentation project. Tags: recommended, techdocs, mkdocs. Tibco templates: Choose.
- Flogo Call Service Template**: Create a new Flogo project that Call a Service. Tags: flogo, tibco, recommended. Tibco templates: Choose.
- BWCE - Bookstore**: Bookstore Sample Template. Tags: tibco, bwce, recommended. Tibco templates: Choose.

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

## Start tracking your component in devhub\_userxx.local

1 Select URL

URL\*

`https://github.com/tibcoeducation/JenkinsRepo/blob/main/catalog-info.yaml`

`https://github.com/tibcoeducation/JenkinsRepo/blob/main/catalog-info.yaml`

Analyze

2 Import Actions  
Optional

3 Review

4 Finish

Then click on Import

3 Review

The following entities will be added to the catalog:



<https://github.com/tibcoeducation/JenkinsRepo/blob/main/catalog-info.yaml>

Entities: 2

jenkins\_component

generated-e1030d52bb91f966c6b034afb3fb2227532304f2

[Back](#)

[Import](#)

4 Finish

Then click on view component

The screenshot shows a software interface with a vertical blue sidebar on the left. The main area has a light gray background. At the top, there is a purple circular icon with a checkmark and the word "Review". Below it, a thin vertical line leads down to another purple circular icon with the number "4" and the word "Finish". Underneath the "Finish" icon, the text "The following entities have been added to the catalog:" is displayed. Below this, there is a list of entities:

- 📍 <https://github.com/tibcoeducation/JenkinsRepo/blob/main/catalog-info.yaml>  
Entities: 2
  - jenkins\_component
  - generated-e1030d52bb91f966c6b034afb3fb2227532304f2

At the bottom of the catalog list, there are two buttons: a purple button labeled "View Component" and a white button labeled "Register another".

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

```
8
9 export const tibcoThemeLight = createUnifiedTheme({
0 ...createBaseThemeOptions({
```

To

```
8
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

The screenshot shows the TIBCO Developer Hub interface. On the left, there's a sidebar with links like Home, Control Plane, Catalog (which is selected and highlighted in blue), APIs, Docs, Develop..., Import..., Settings, and Sign out. The main area is titled "All components (6)". It has a table with columns: Name, System, Owner, Type, and Lifecycle. The data in the table is as follows:

| Name                     | System           | Owner                      | Type    | Lifecycle    |
|--------------------------|------------------|----------------------------|---------|--------------|
| car-analyzer             | car-order-system | The Finance Department     | service | production   |
| car-information-provider | car-order-system | The Operational Department | service | production   |
| car-order-ui             | car-order-system | The Operational Department | service | production   |
| db-adapter               | car-order-system | The Operational Department | service | production   |
| discount-calculator      | car-order-system | The Finance Department     | service | production   |
| Jenkins_Component        |                  | jenkins                    | service | experimental |

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
| 125 </Link>
126 </SidebarItem>
```

To

```
119 <LINK to="/">
120 data:image/s3,anthropic-data-us-east-2/u/marker_images/0111/0100/0010/11100001/sfishman-chandramapper-0319211047/08f325c371c148690cb877a83197d3b2.jpg</antml:image>

The screenshot shows a web browser window displaying the TIBCO Developer Hub. The URL in the address bar is `localhost:3000/tibco/hub/`. The page has a dark blue header with the TIBCO Cloud logo and a search bar. On the left, there's a sidebar with navigation links: Home (highlighted in blue), Control Plane, Catalog, APIs, Docs, and Develop... Below the sidebar, the main content area features a heading "Welcome to the TIBCO® Developer Hub, Guest" and a sub-headline "TIBCO® Developer Hub is the center for building the apps for empowering your organisation". A section titled "What is the TIBCO® Developer Hub ?" provides a brief description of the platform's purpose. At the bottom, there are two buttons: "Get started" and "See how it works". To the right of the content, the large TIBCO logo is displayed.