EPOCH Quick Start

Version 0.1.0

Exastro developer

Table of contents

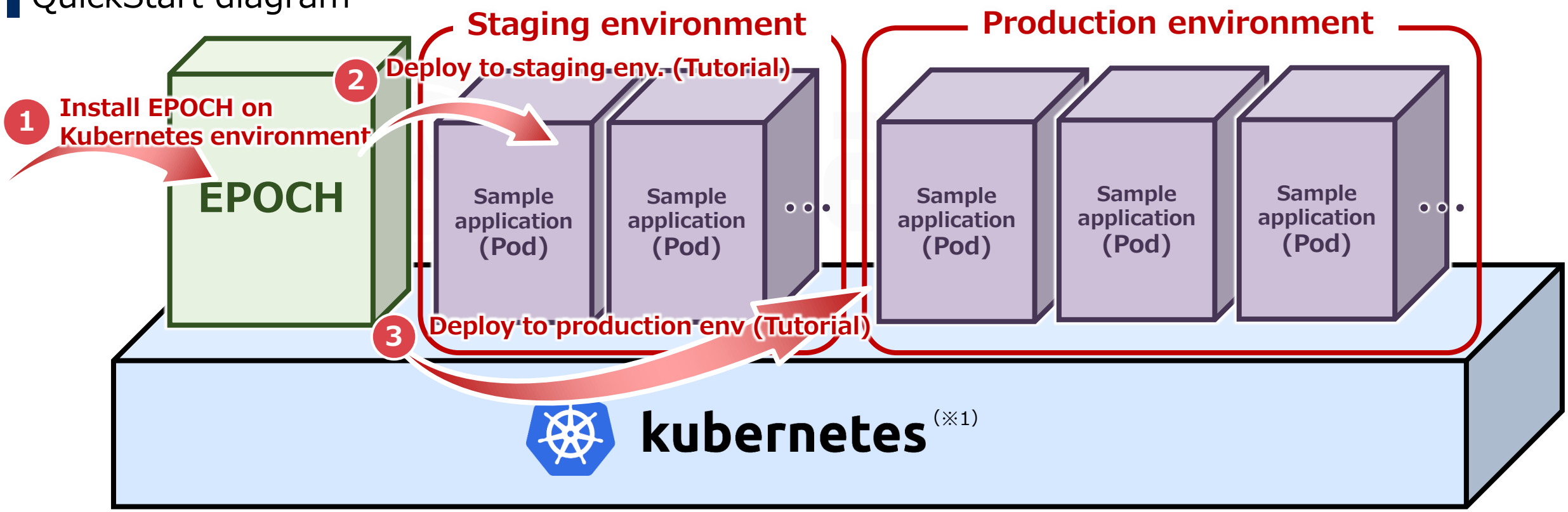
1. Introduction

1.1 Quickstart guide

About this document

This document aims to introduce the reader to Exastro EPOCH(Hereafter written as “EPOCH”), how to install it while also guiding the reader through a simple tutorial.

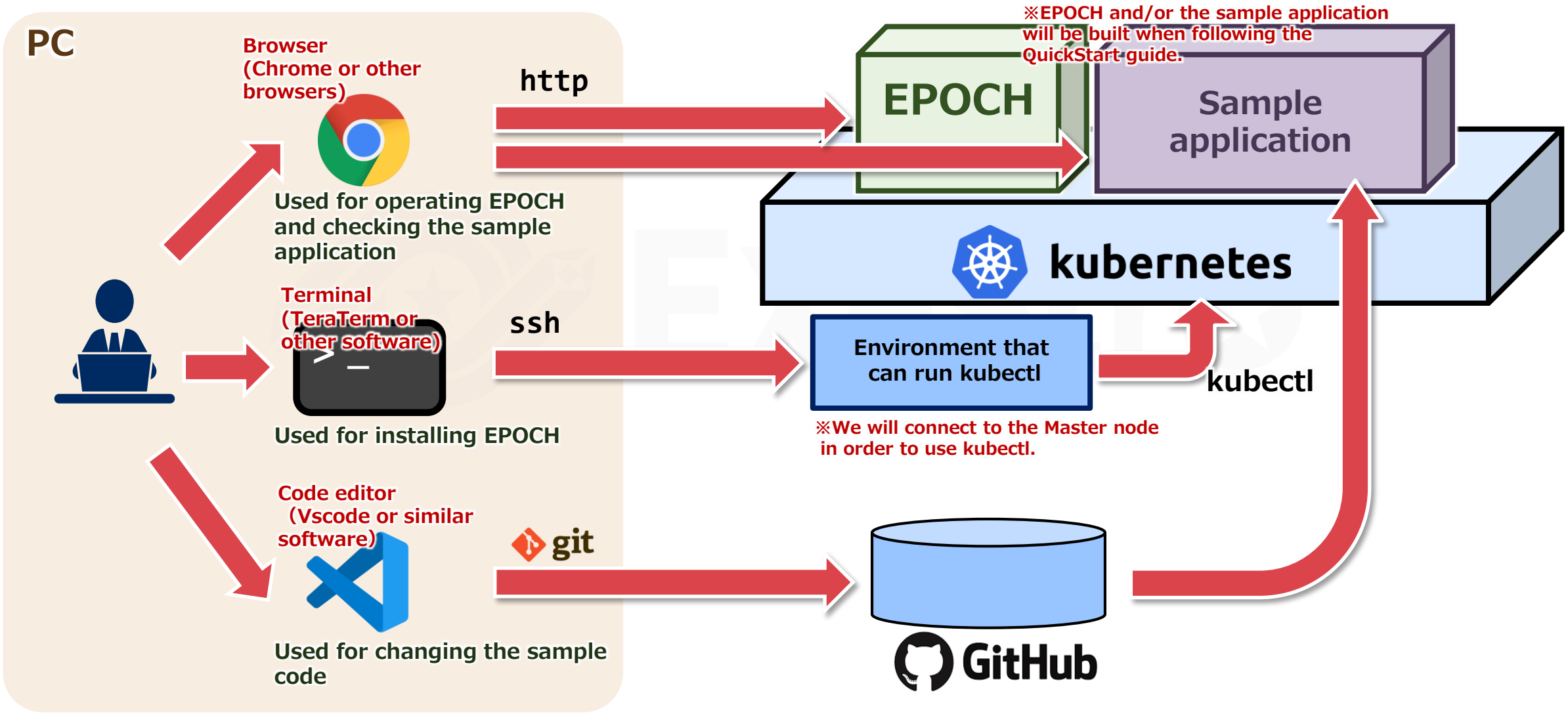
QuickStart diagram



(※1)In order to keep the tutorial simple, everything will be constructed on a single Kubernetes cluster.

1.2 Quickstart PC environment

The following figure illustrates the PC software necessary for following this guide.



2. Install

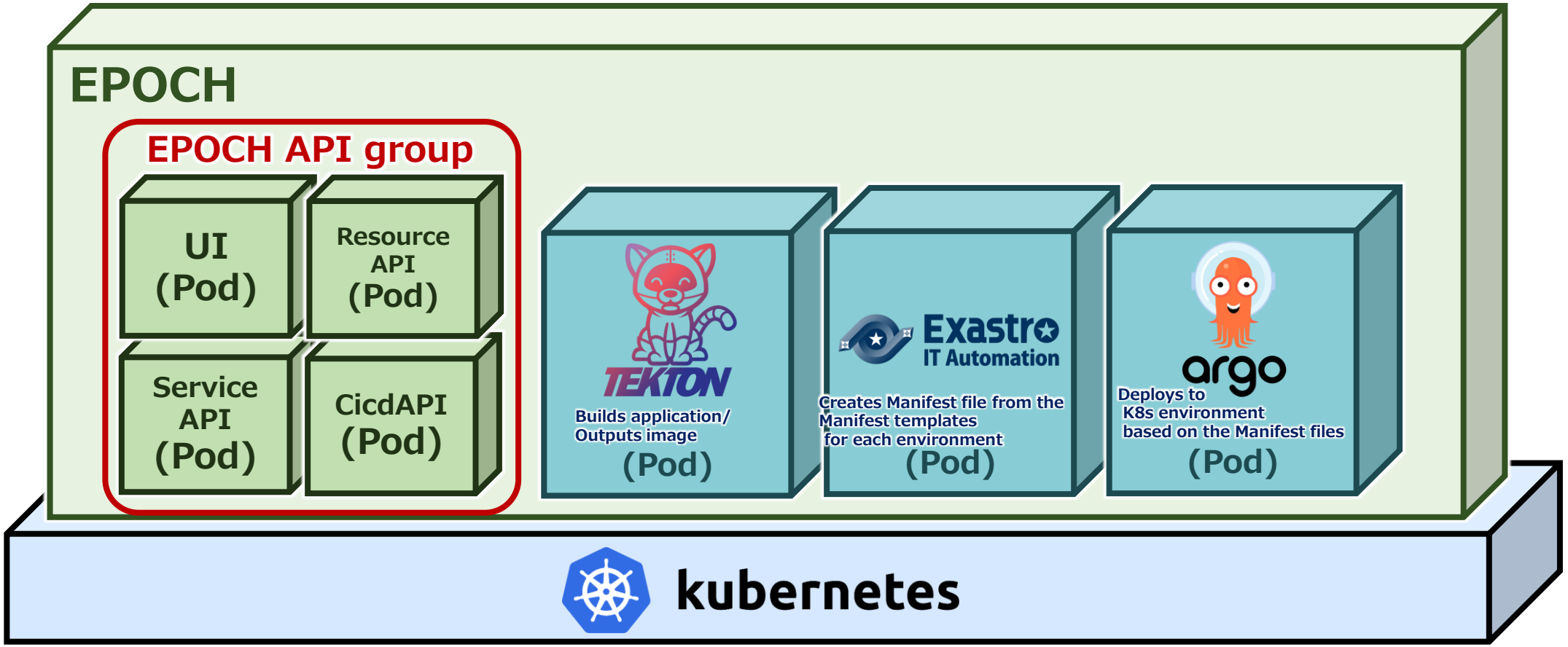
Install EPOCH and prepare the CI/CD environment

Exastro

2.1 Install EPOCH (1/5)

EPOCH diagram

The following diagram illustrates EPOCH and the Workspace configuration after it has been created



2.1 Install EPOCH (2/5)

Prerequisites

- Environment
  - A Kubernetes environment must be created
  - The ServiceAccount must have the cluster-admin role
  - The user must be able to connect to external internet from the Kubernetes environment
  - “Git for Windows” must be installed in the PC environment
  - The following port numbers must be usable(30080, 30081, 30443, 30801 , 30804, 30805, 30901～30907)
    (The port numbers are written in the epoch-install.yaml file. Run the installer in order to have the changes take effect)

- Account
  - Must have a GitHub account that can register application code
  - Must have a GitHub account that can register Kubernetes Manifests
  - Must have a GitHub account that can register Container images

2.1 Install EPOCH (3/5)

Install EPOCH

- Use a terminal to SSH login to an environment that can run kubectl. Run the following command and install

```
$  kubectl apply -f https://github.com/exastro-suite/epoch/releases/download/v0.1.0/epoch-install.yaml
```

The following command checks the installation progress.

```
$  kubectl get pod -n epoch-system
```

Check that the command result "STATUS" column displays "Running".

【Command result】

```
NAME                                          READY   STATUS    RESTARTS   AGE
epoch-cicd-api-*********-*****                1/1     Running   0          **s
epoch-rs-organization-api-*********-*****     1/1     Running   0          **s
epoch-rs-workspace-api- *********-*****       1/1     Running   0          **s
～
```

2.1 Install EPOCH (4/5)

Configure persistent volume

Configure persistent volume. This will be used for configuring the pipelines.

- Run the following command and acquire the manifest from GitHub.

```
$  curl -OL https://github.com/exastro-suite/epoch/releases/download/v0.1.0/epoch-pv.yaml
```

- The following command checks the Worker nodes and host names.

```
$  kubectl get node
```

【Command result】

```
NAME                       STATUS   ROLES                  AGE   VERSION
epoch-kubernetes-master1   Ready    control-plane,master   **d   v1.**.*
epoch-kubernetes-worker1   Ready    worker                 **d   v1.**.*
```

- Modify the epoch-pv.yaml file.（The last line of the epoch-pv.yaml file）

Change the “# Please specify the host name of the worker node #” text with the worker node and the host name acquired from the last command and save the file.

【Modified values】

```
values:
  - # Please specify the host name of the worker node #
```

```
values:
  - epoch-kubernetes-worker1
```

- Use the following command to reflect it to the kubernetes environment.

```
$  kubectl apply -f epoch-pv.yaml
```

2.1 Install EPOCH (5/5)

Install ArgoRollout

- Run the following command to install ArgoRollout.

```
$  kubectl create namespace argo-rollouts
$  kubectl apply -n argo-rollouts -f https://github.com/argoproj/argo-rollouts/releases/latest/download/install.yaml
```

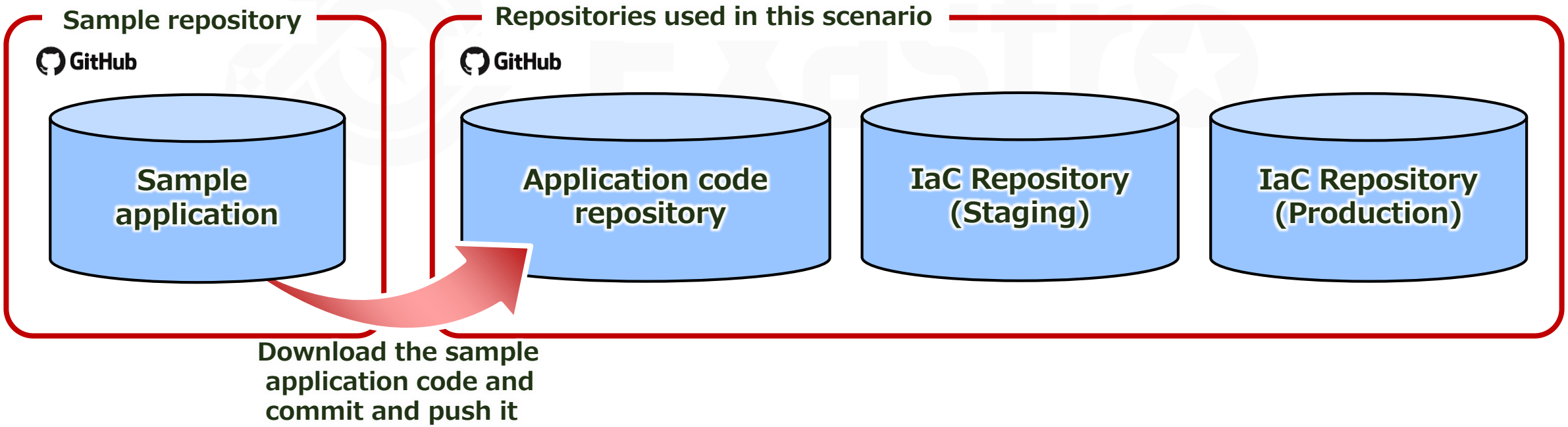
This concludes the steps to installing EPOCH.
Let us prepare for the tutorial!

2.2 Prepare repositories (1/4)

About the repositories

- The repositories used in this document are as following
  - Repository for application codes
  - Repository for IaC (Staging environment)
  - Repository for IaC (Production environment)

- Repository diagram

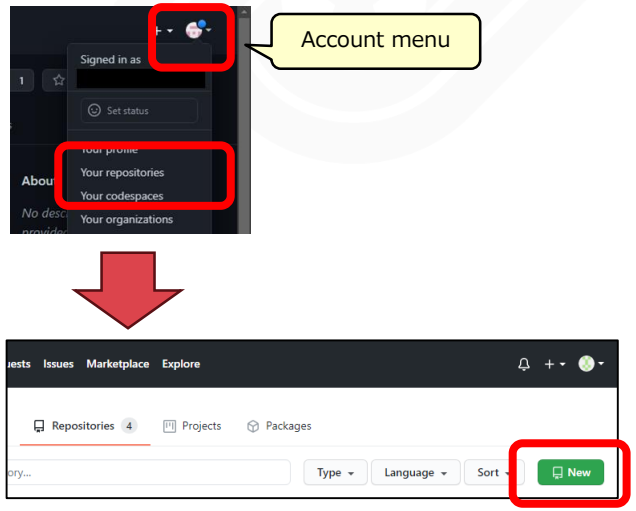


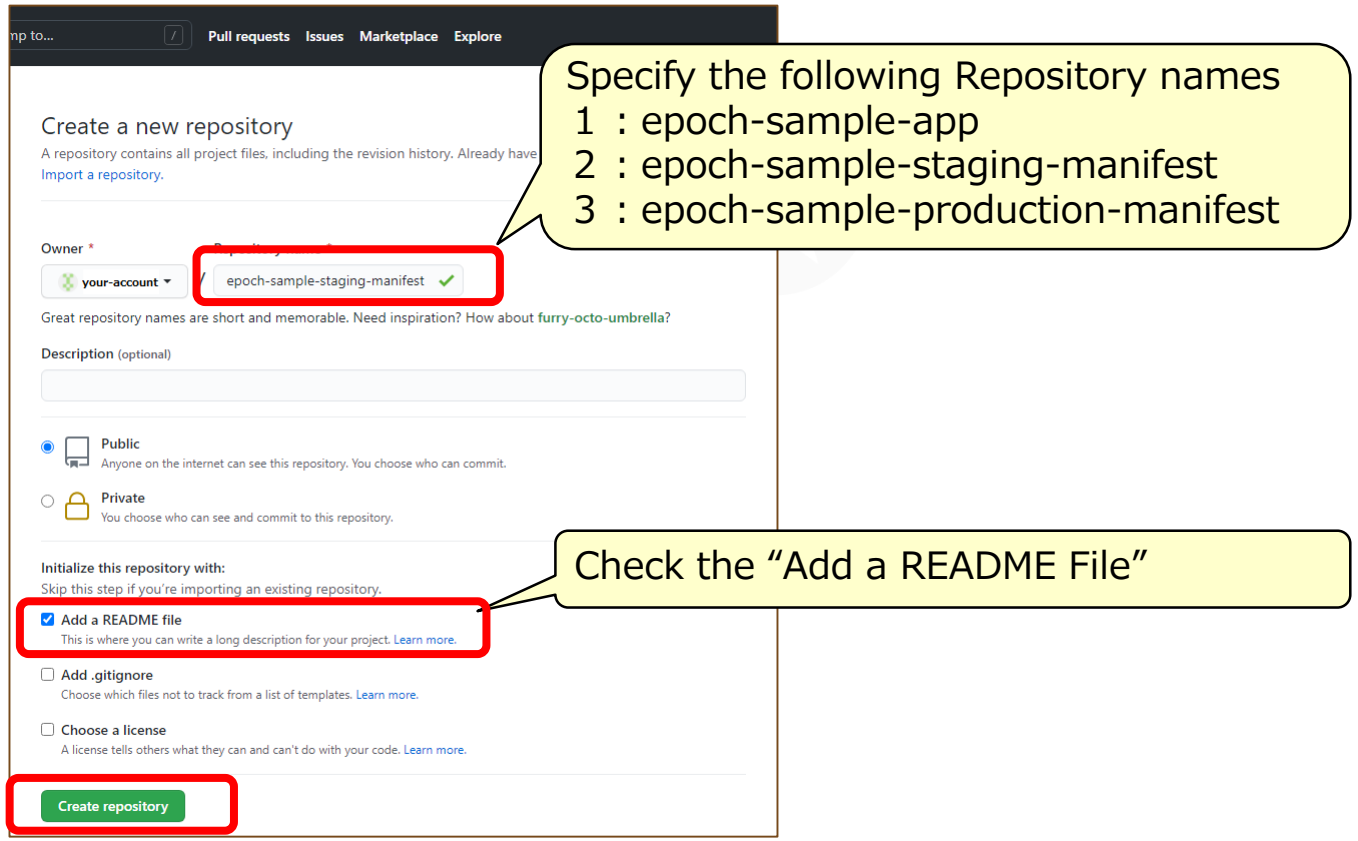
2.2 Prepare repositories (2/4)

Prepare the repositories

Prepare 3 Git repositories.

- Use your browser to sign into GitHub with your own GitHub account.
- Go to the account menu and press “Your repositories”.
- Press the “New” button in the upper right corner and follow the diagram below to fill in the values and create the repository.
- Lastly, press the “Create repository” button.





2.2 Prepare repositories (3/4)

Prepare repository for application code.

- Clone the Application code repository

  Clone the Application code repository to the PC environment.

  The commando prompt should look something like this:

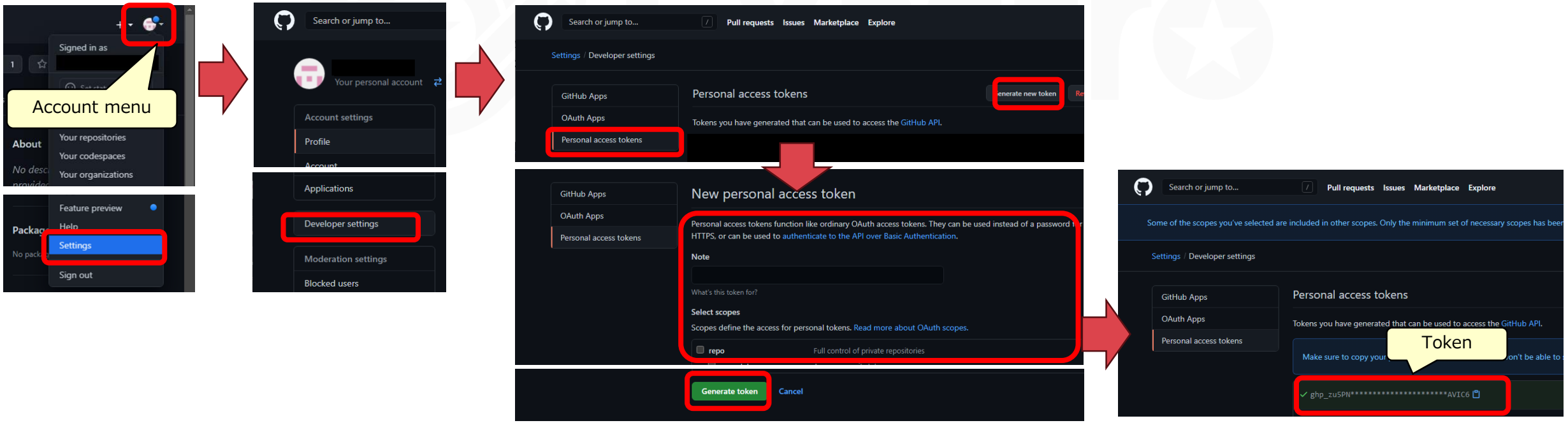
```
>   cd “[clone folder]"
>   git clone https://github.com/[Github account name]/epoch-sample-app.git
>   cd epoch-sample-app
>   git config user.name “[GitHub user name]"
>   git config user.email “[GitHub email address]"
```

  The next part of the tutorial will use the cloned local repository.

2.2 Prepare repositories (4/4)

Pay out Git token

- Use your browser to sign into GitHub with your own GitHub account.
- Go to the account menu and press “Settings”.
- Go to the Account settings and then to the “Developer settings” menu.
- Go to the “Personal access tokens” menu and press “Generate new token”.
- Input a name om the “Note” field and select all the scoped before pressing the “Generate token”
- The token (ghp_***) will be used later , so make sure to save it somewhere.



2.3 Manifest template files

Download Manifest template files

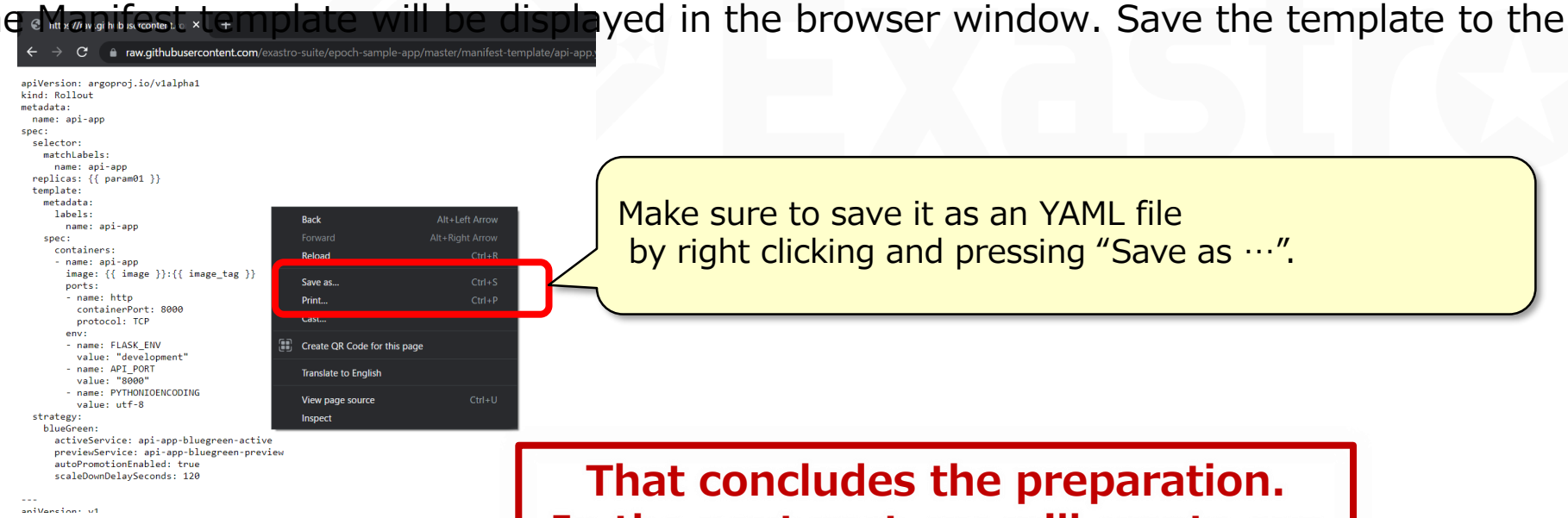
Download the 2 Manifest template files we will upload to EPOCH.

- Access the files from the following URLs.

  File 1 ： https://raw.githubusercontent.com/exastro-suite/epoch-sample-app/master/manifest-template/api-app.yaml

  File 2 ： https://raw.githubusercontent.com/exastro-suite/epoch-sample-app/master/manifest-template/ui-app.yaml

- The Manifest template will be displayed in the browser window. Save the template to the PC.



Make sure to save it as an YAML file
by right clicking and pressing “Save as …”.

That concludes the preparation.
In the next part, we will create our workspace.

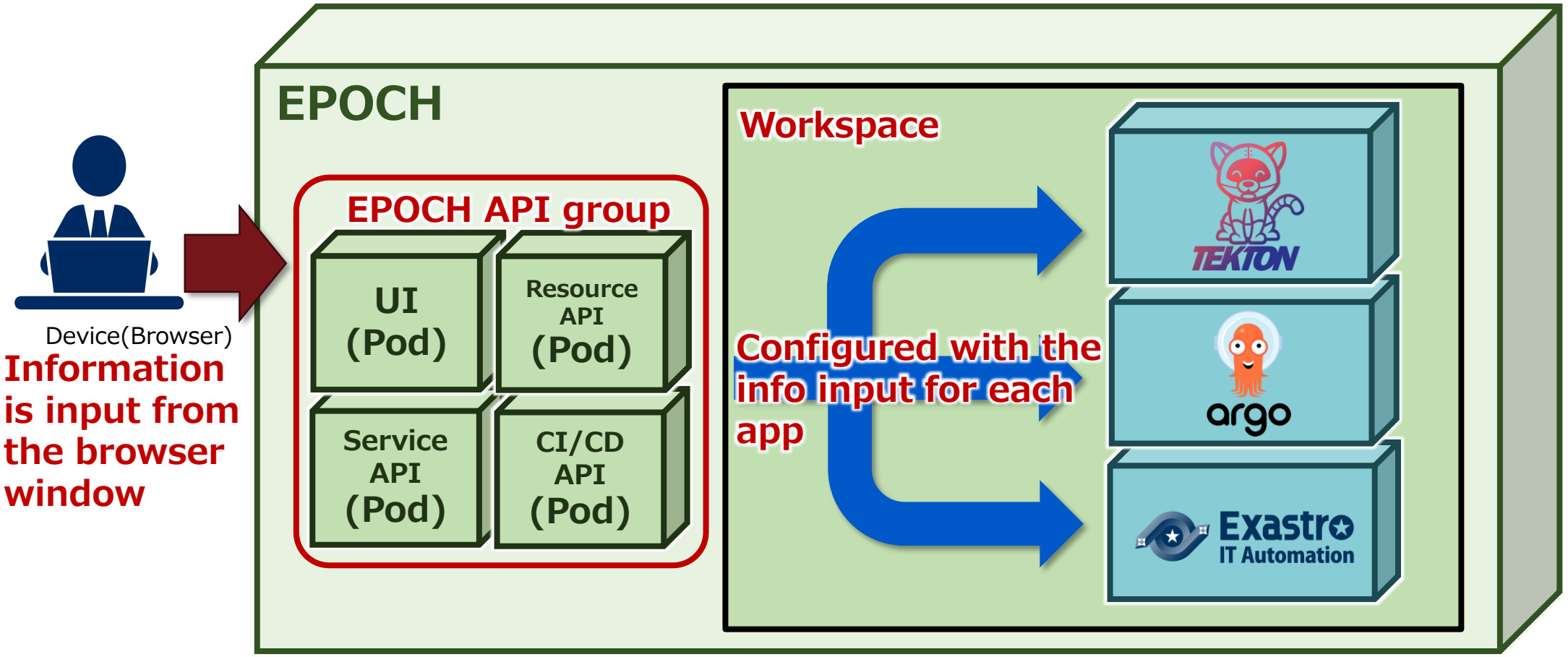
3. Create Workspace

Create Workspace and prepare CI/CD.

3.1 Workspace

Workspace

In EPOCH, the word “Workspace” specifies a workspace for 1 development environment.
A workspace can be created by inputting the necessary information on the EPOCH screen.



3.2 CI/CD

What is CI/CD?

CI/CD is a method for developers to increase the frequency of app delivery by automating the tasks from app development to release.

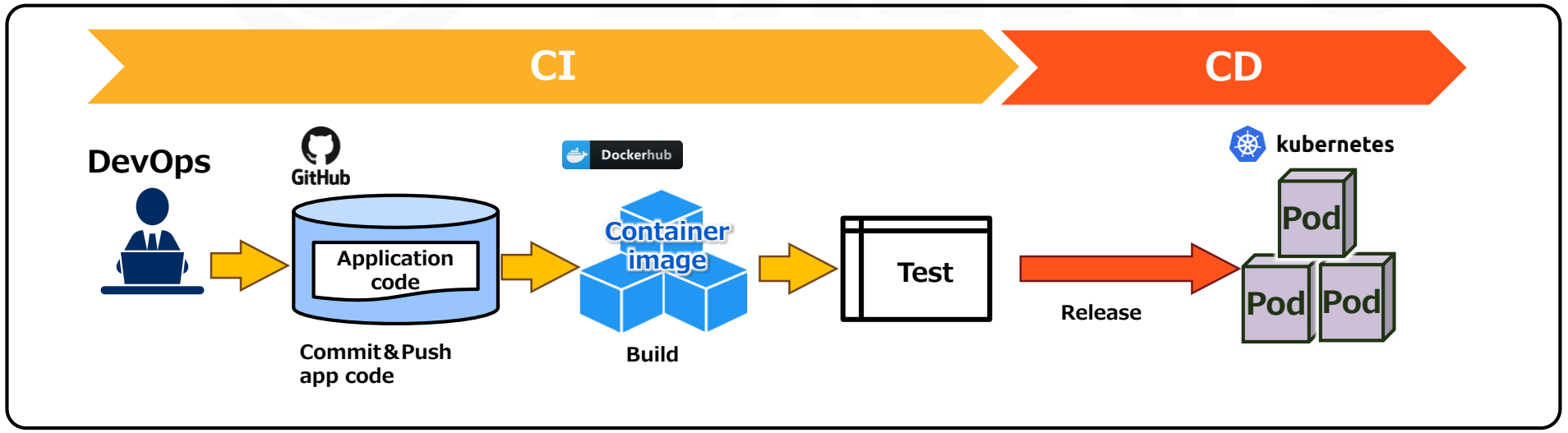
- CI（Continuous Integration）

  CI is the name of the process of automated building and testing application code whenever there has been a change to it.

- CD（Continuous Delivery）

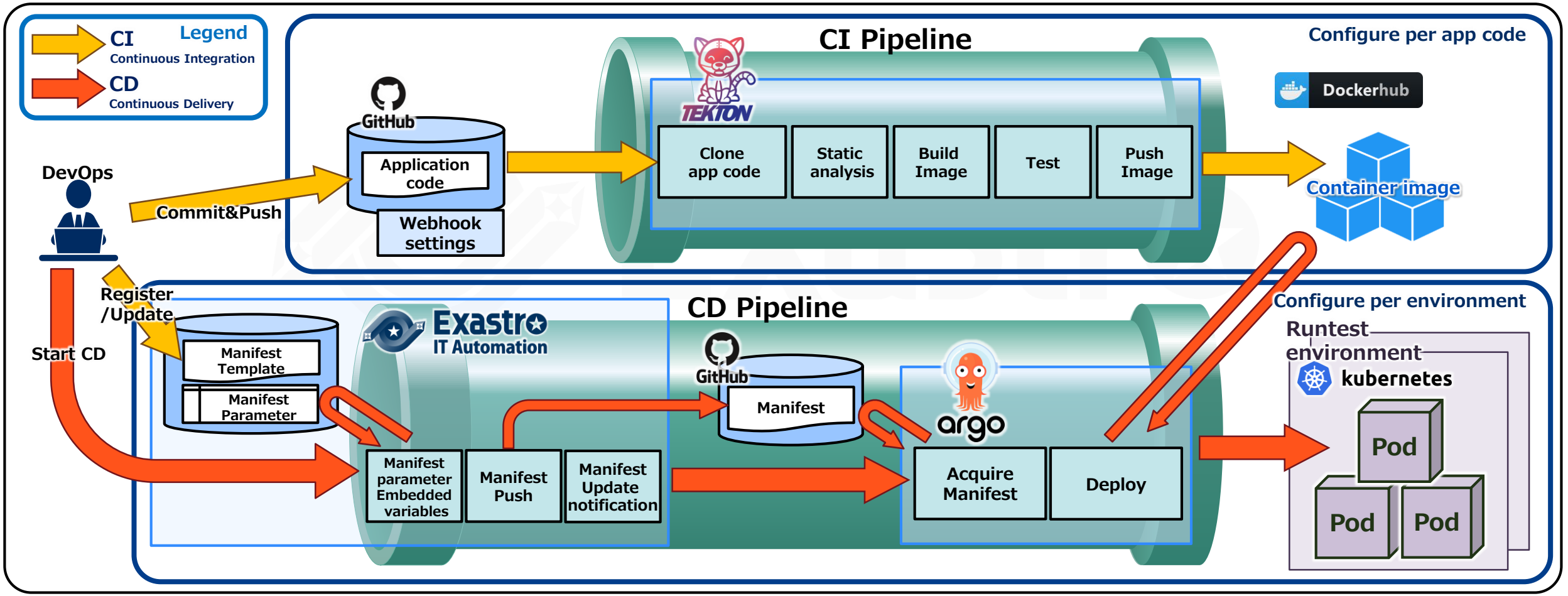
  CD is the name of the process of automating the tasks of releasing the code to the runtime environment.

- CI/CD image



CI/CD in EPOCH

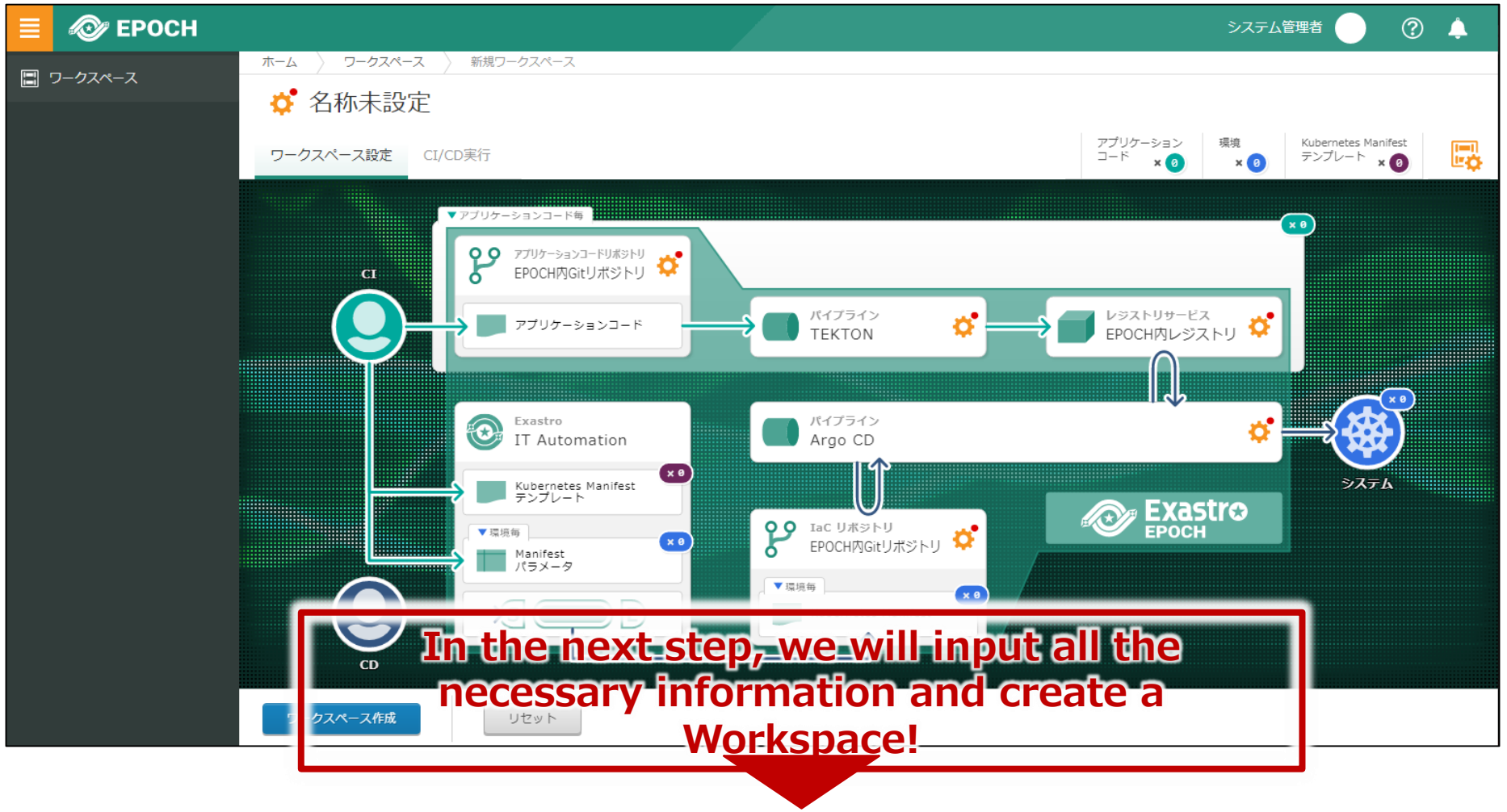
The following diagram illustrates the CI/CD used in EPOCH.



3.4 Start EPOCH

Use the following URL to access the EPOCH environment.

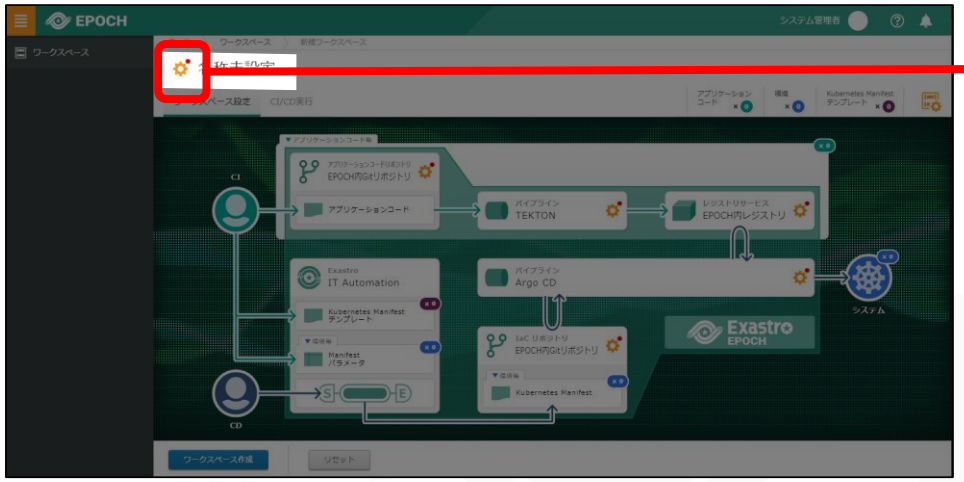
https://[IP Address or Host name]:30443/workspace.html

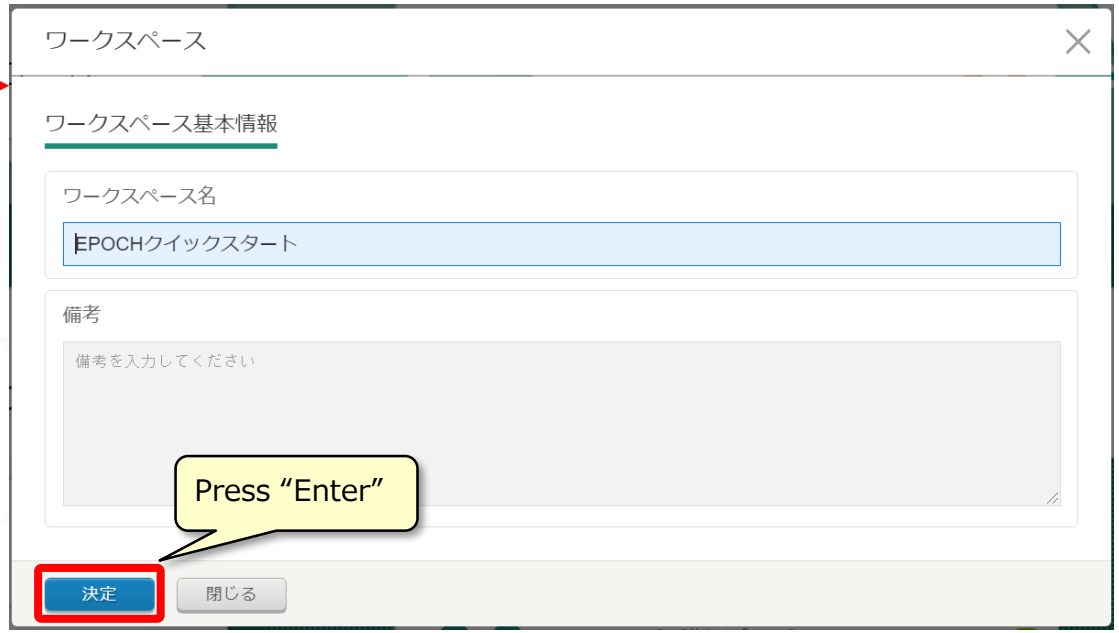


3.5 Create Workspace (1/7)

Workspace information

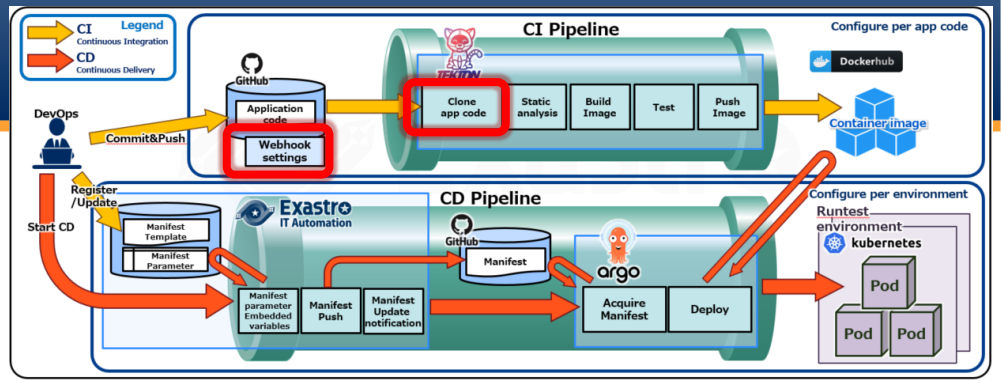
Input the Workspace name.





Item name	Input・Select values	Description
Workspace name	EPOCH Quickstart	Name of the workspace
Remarks	BLANK	Free space for describing the workspace

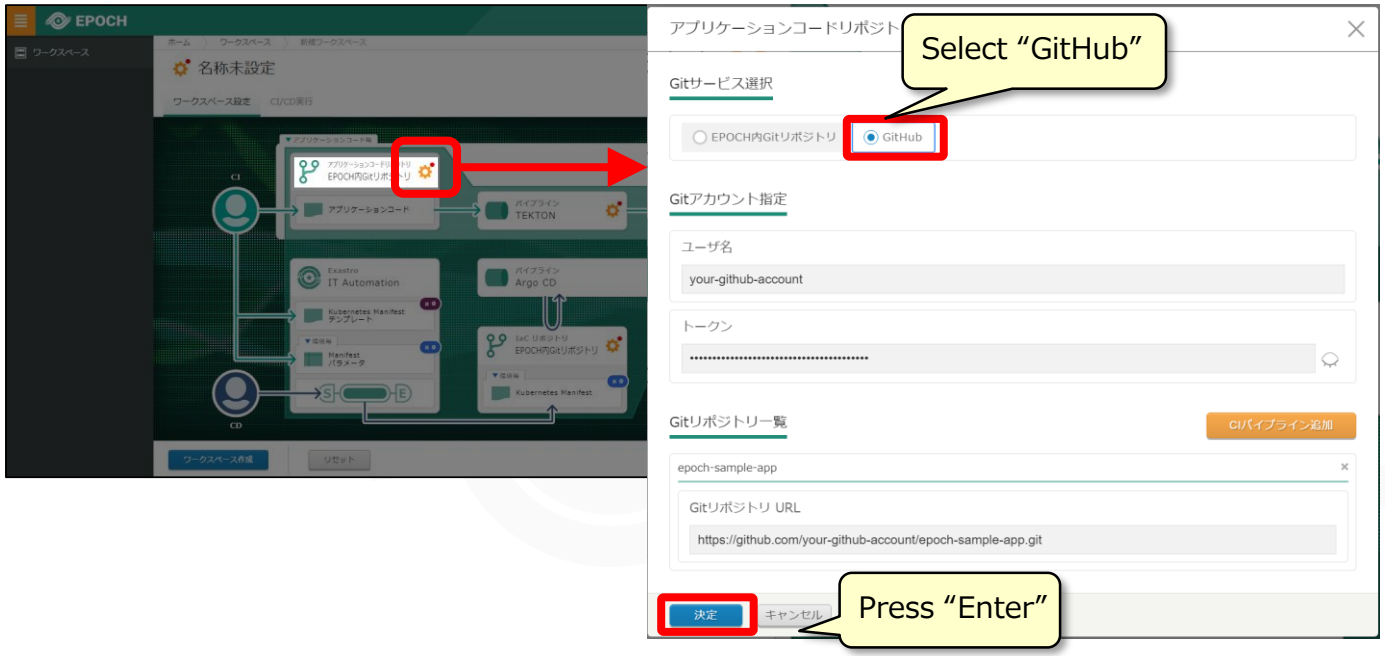
3.5 Create Workspace (2/7)



※This section specifies the areas marked in red

App code repository

Input information for the application code repository.

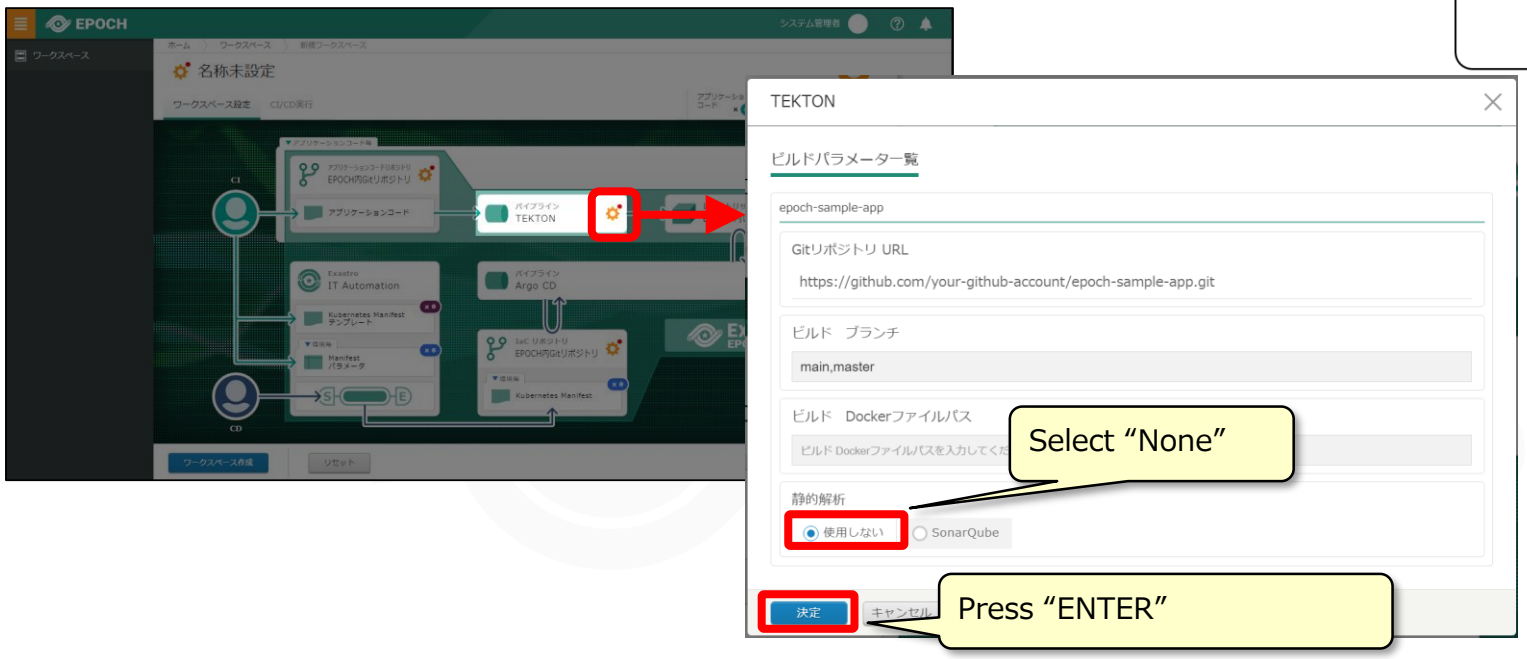


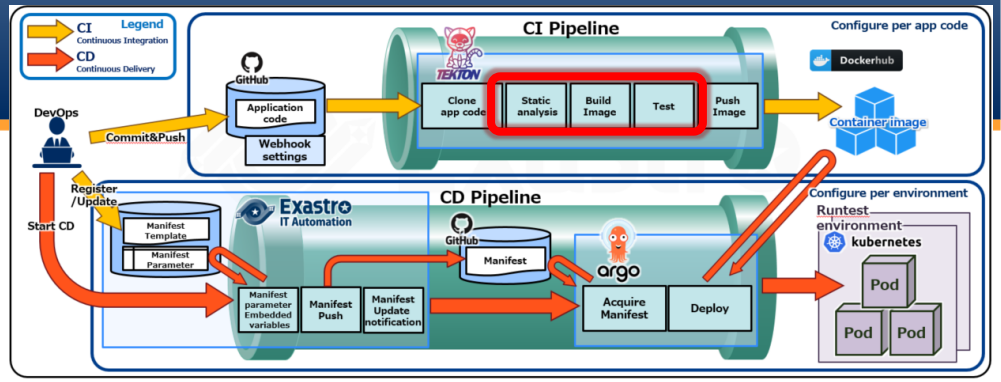
Item	Input/Select	Description
User name	(GitHub account name)	Your personal GitHub account name
Token	(GitHub token)	GitHub token (See the Git token you created earlier)
Git Repository URL	https://github.com/[GitHub account name]/epoch-sample-app.git	Input the URL for the application code repository you created earlier.

3.5 Create Workspace (3/7)

TEKTON Pipeline

Input information for the pipeline that will be set to TEKTON.





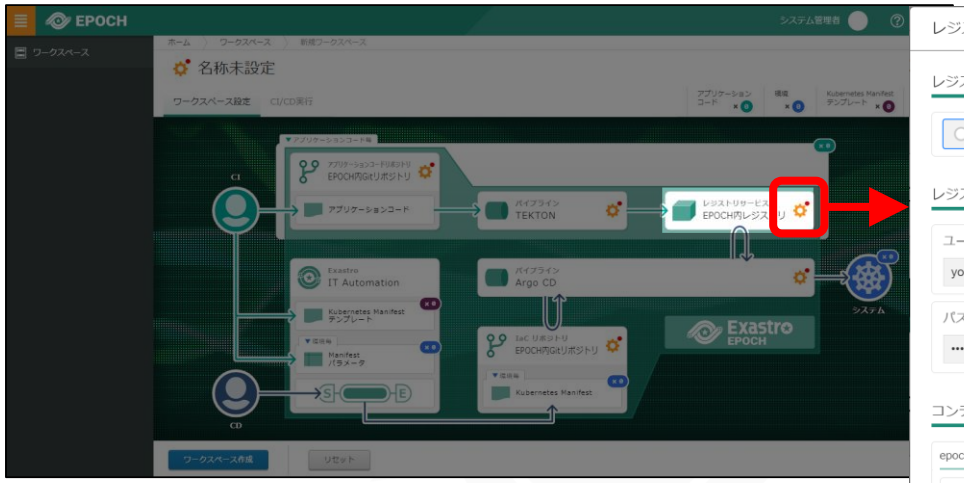
※This section specifies the areas marked in red
※Static response tests and other tests are planned to be added in the future.

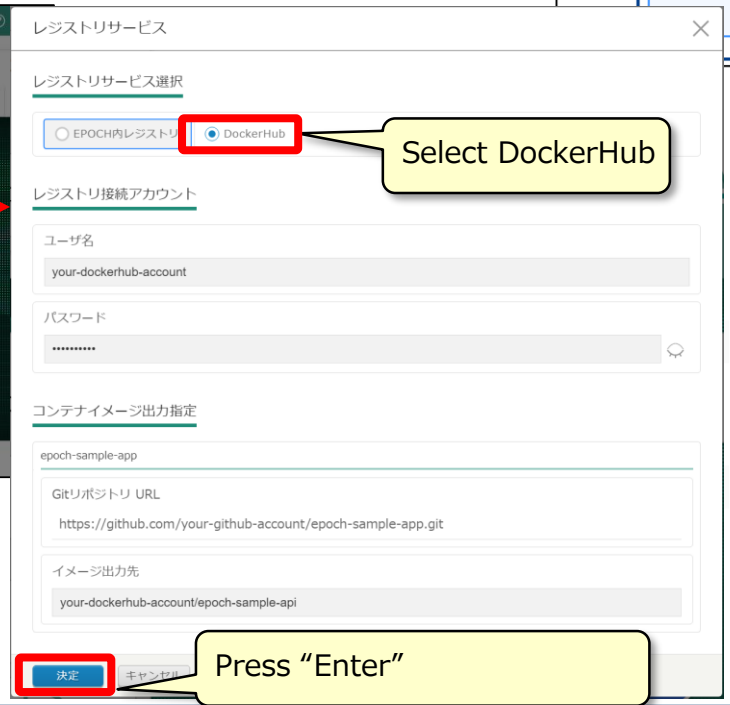
Item	Input/Select	説明
Build branch	main,master	The build's target GitHub application branch
Docker build file path	./api-app/Dockerfile	Application docker file path

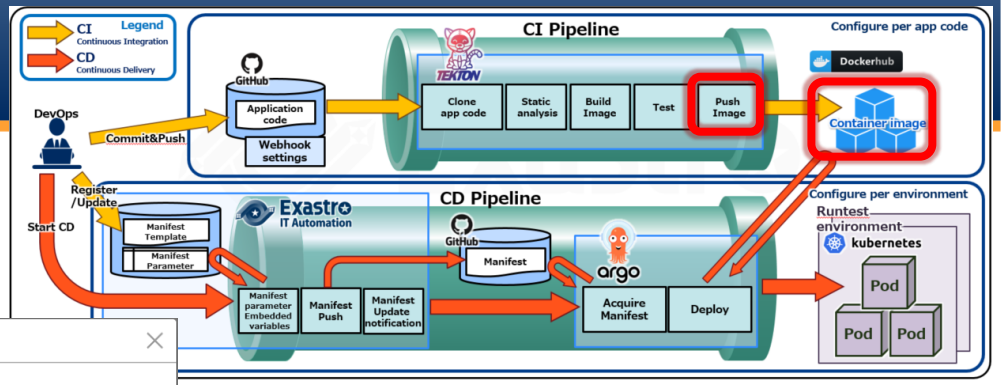
3.5 Create Workspace (4/7)

Registry service

Input the image registration destination (Registry).







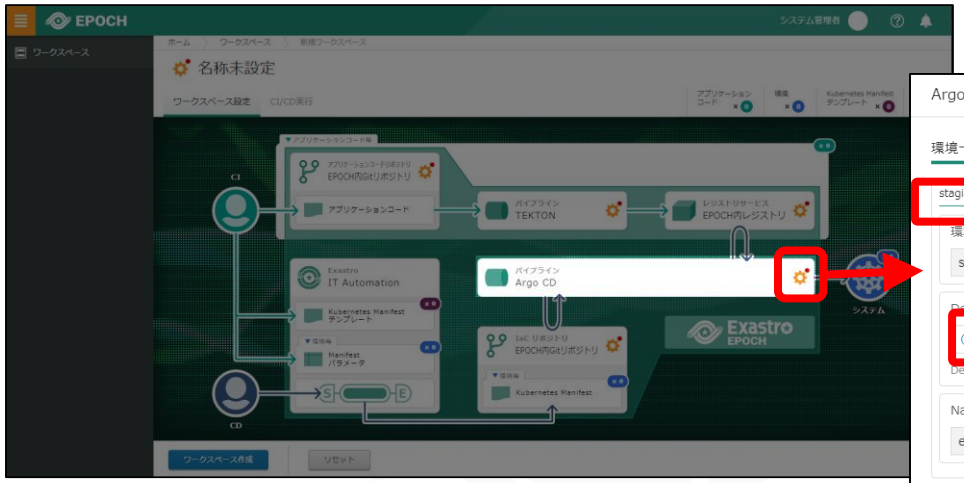
※This section specifies the areas marked in red

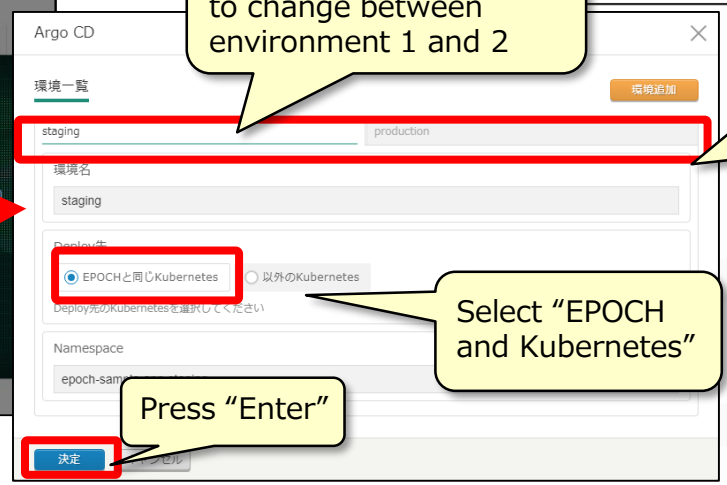
Item	Input/Select	Desciption
User name	(Your personal DockerHub account name)	DockerHub account name
Password	(Your personal DockerHub password)	DockerHub password
Image output destination	[DockerHub account name]/epoch-sample-**api** ※Modify the text that displays after inputing the user name.	DockerHub image output destination

3.5 Create Workspace (5/7)

Argo CD Pipeline

Input the Deploy information that will be set to ArgoCD.



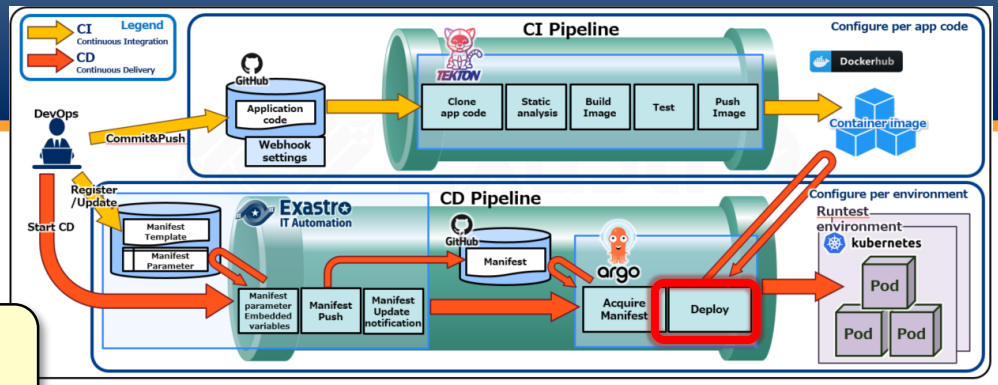


Use the tabs on the top to change between environment 1 and 2

Select “EPOCH and Kubernetes”

Press “Enter”

The tutorial in this guide will deploy to 2 environments. Press the "Add environment" button to add a new environment.



※This section specifies the areas marked in red

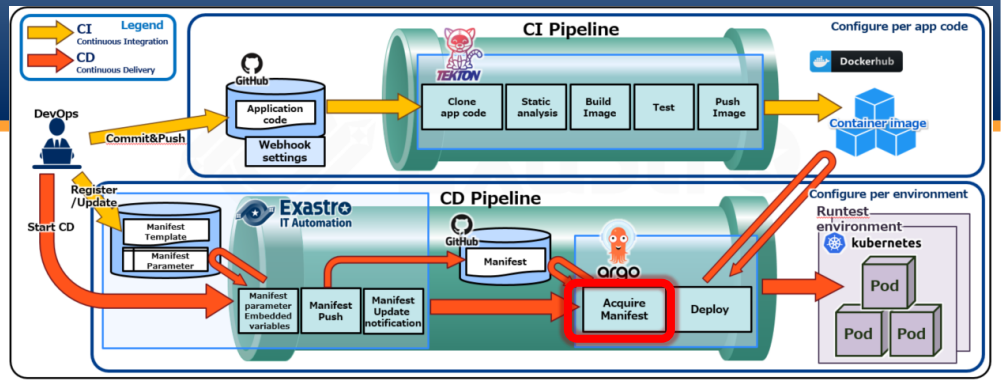
Environment 1 : Staging environment

Item	Input/Select	Description
Environment name	staging	Name of the environment
Namespace	epoch-sample-app-staging	Deploy destination Namespace

Evnrionment2 : Production environment

Item	Input/Select	Description
Environment name	production	Name of the environment
Namespace	epoch-sample-app-production	Deploy destination Namespace

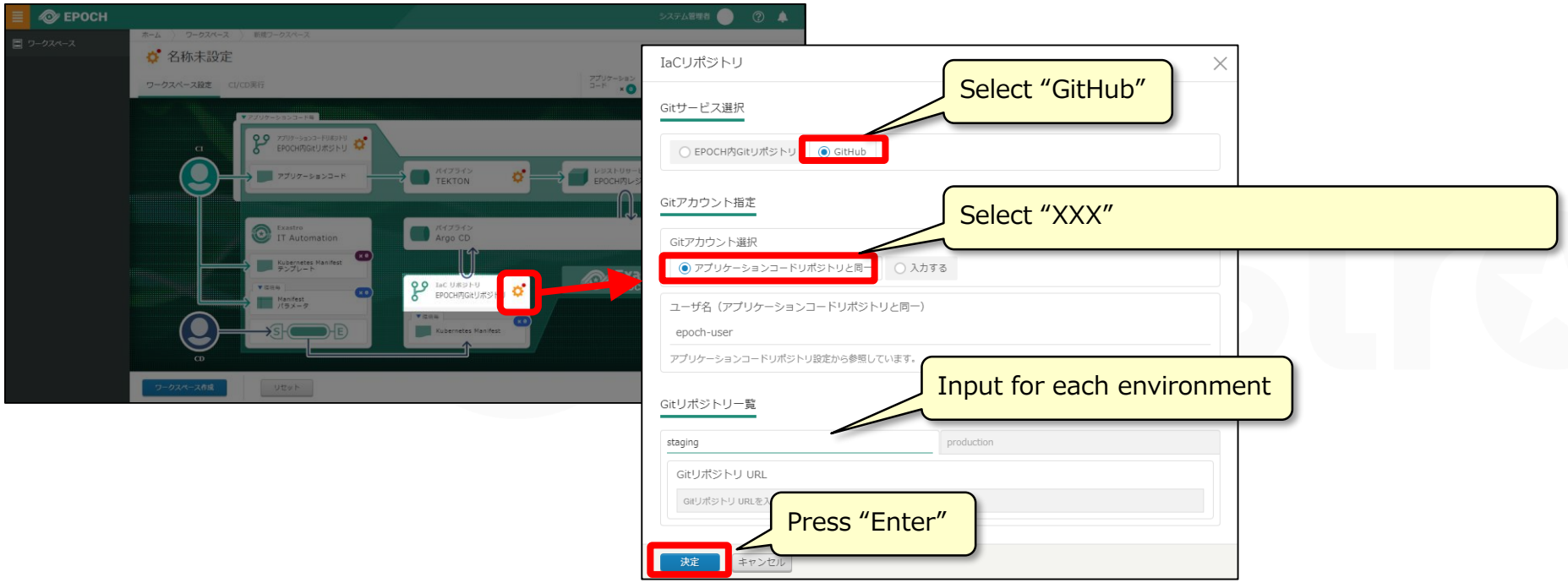
3.5 Create Workspace (6/7)



※This section specifies the areas marked in red

IaC repository

Input the repository information.
This will determine where the manifest will be registered to.

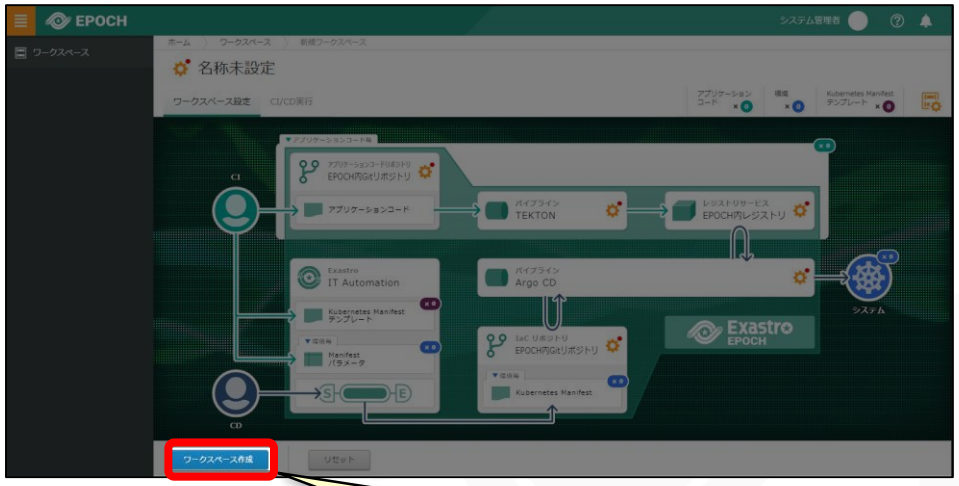


Item	Input/Select	Description
Git repository URL	https://github.com/[GitHub account name]/[Environment repository].git	The Environment’s Manifest repository. (See the IaC repository we prepared earlier)

3.5 Create Workspace (7/7)

Create workspace

After you have input all the necessary information, press “Create Workspace”.



Press “Create Workspace”

You have now created the CI/CD pipeline!
The next tutorial will guide you through a scenario of using the CI/CD pipeline.!

4. Tutorial

Let’s use the CD/CD workflow

4.1 Tutorial overview (1/2)

CI/CD development scenario

This tutorial uses the following scenario as a base to guide the reader through a CI/CD workflow.
In this tutorial, the reader will create an EC site by using a prepared sample application.

The client wants you to develop an EC site app

Can you make us an EC site for selling T-shirts?

We also want it to be cloudnative and use CI/CD.

Of course!

I will try…

I need to configure a container orchestration in Kubernetes and prepare a pipeline for both CI/CD.
I also need to configure a trigger, where I will need to do this and this...
I want the CD to be a fault-tolerant system, so I will need to think about the different patterns...

I will need a pod for the UI and the BackEnd (API), as well as two different environments for staging and production

There sure is a lot to do.
There must be an easier way to do this…
Wait, didn't EPOCH support CI/CD?

Let's try and use EPOCH!

Covered in page 34 - 58

CI/CD Workflow No.1

Create an EC site that supports both Japanese Yen (YEN) and American dollar (USD)

Done !

4.1 Tutorial overview (2/2)

The client asks for a new

Client

The EC site is doing great!

That is good to hear

DevOps

Client

We want to appeal to the European market as well. Can we add Euro as well?

Of course!

DevOps

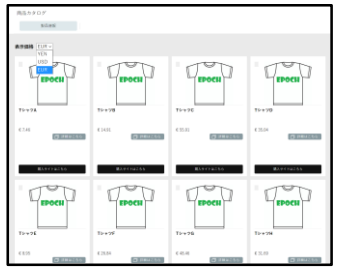
The CI/CD is already completed, so it's easy to deploy once I've fixed the code!

DevOps

Covered in page 59 - 80.

CI/CD Workflow No.1

Update the EC site so that it supports Euro(EUR) in addition to JPY and USD



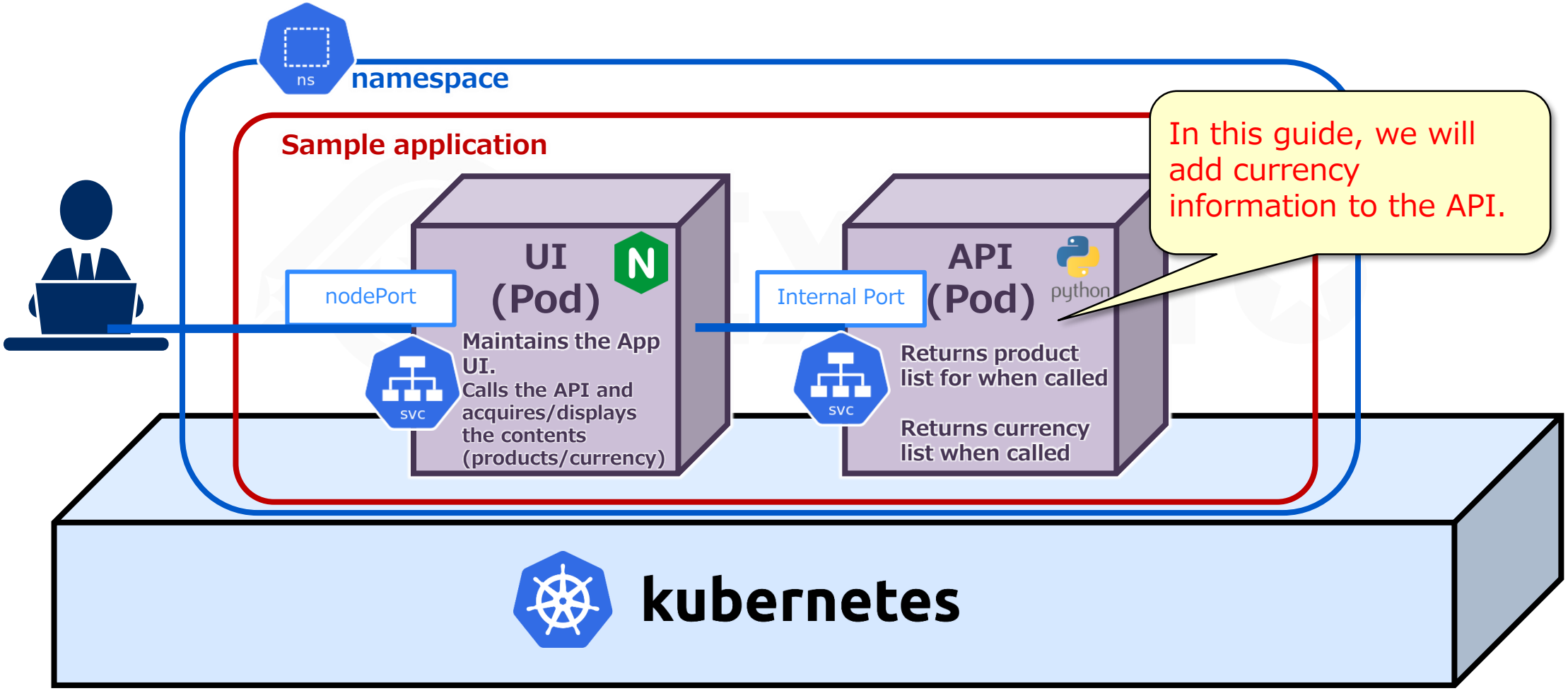
Done !

DevOps

4.2 Sample application configuration

Sample application configuration

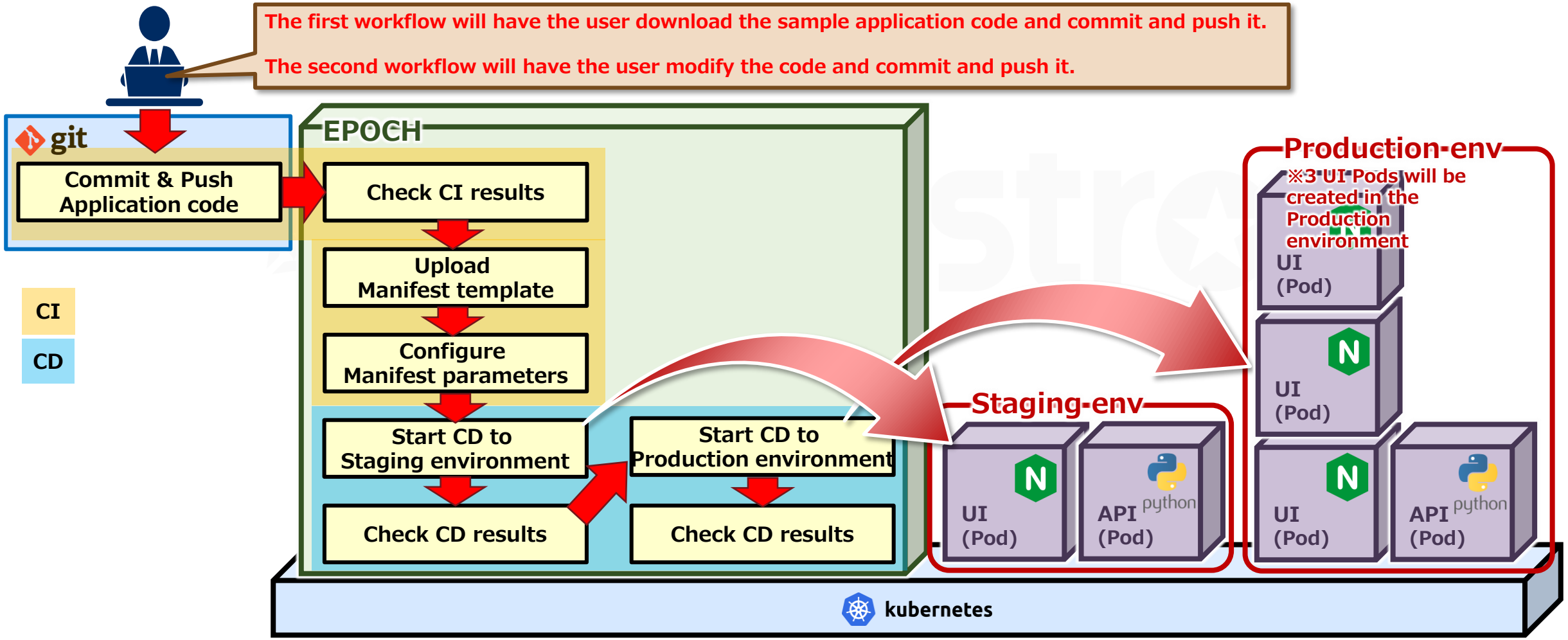
The sample application is created by an UI and an API.



4.3 Tutorial (CI/CD workflow)

CI/CD workflow

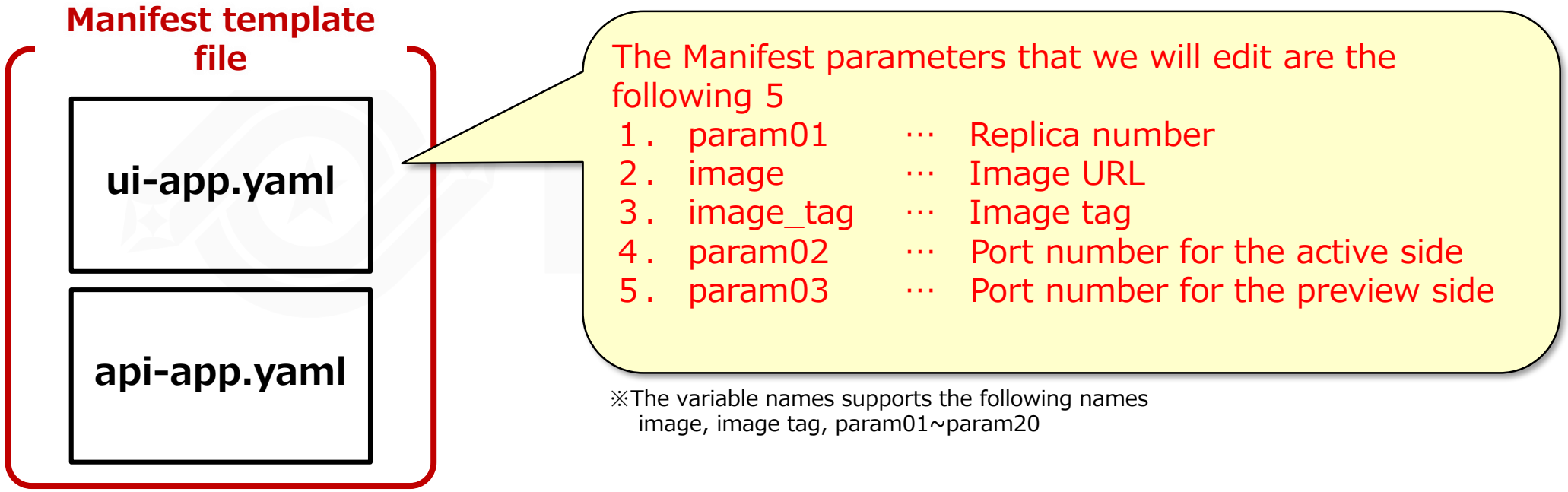
In this scenario, the user will deploy the sample code to both the Staging and the Production environment.
After that, they will modify the sample code and proceed to deploy it to both the Staging and the Production environment.



4.4 Manifest template file

Manifest template file

There are two Manifest template files for the sample application. One for UI and one for API.
They are template files that have modifiable variables for matching environments.



※The variable names supports the following names
image, image tag, param01~param20

In the next section, we will start the first CI/CD workflow!

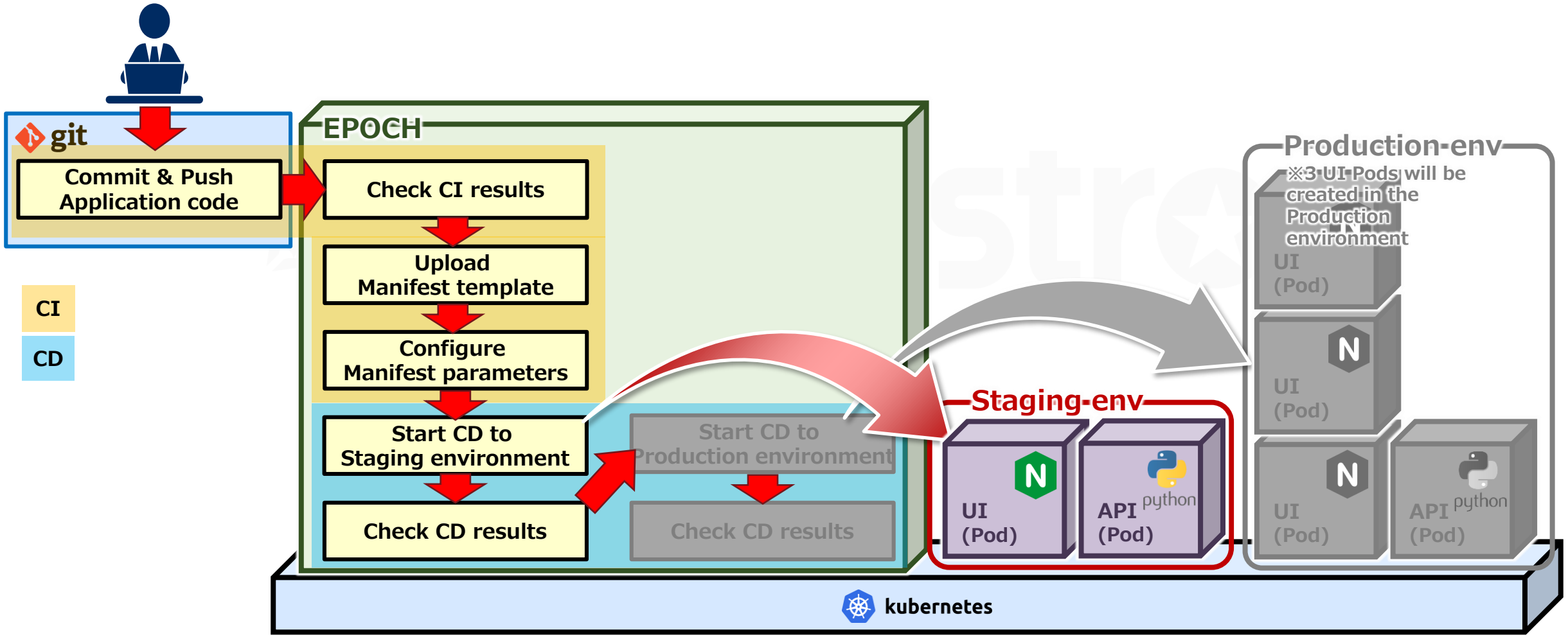
CI/CD Workflow No.1

Exastro

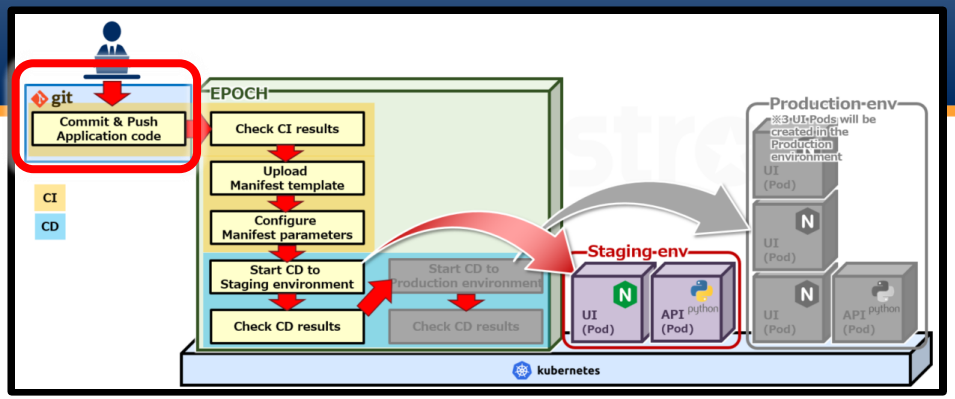
4.5 CI/CD Workflow No.1 (1/25)

Deploy to Staging environment

The first CI/CD workflow will have the user commit and push the application code, deploy it to the Staging environment, as well as checking the results of the CD.



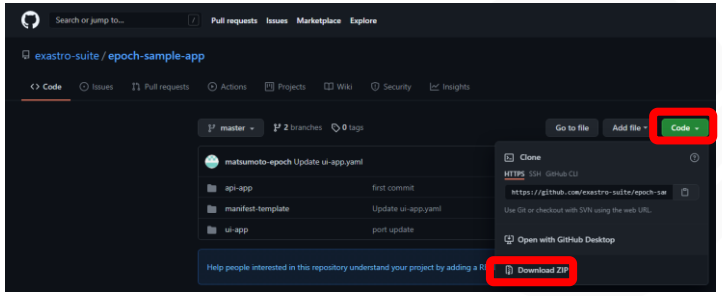
4.5 CI/CD Workflow No.1 (2/25)



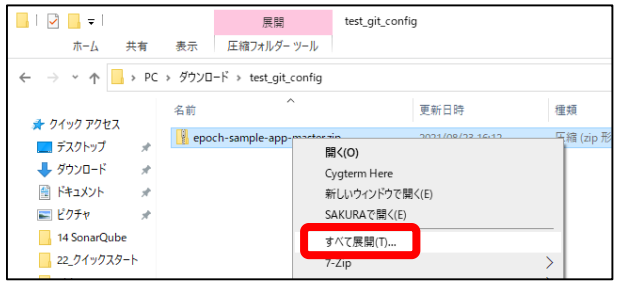
Commit & Push Application code

Run the CI Pipeline in order to create
the container image we will deploy.

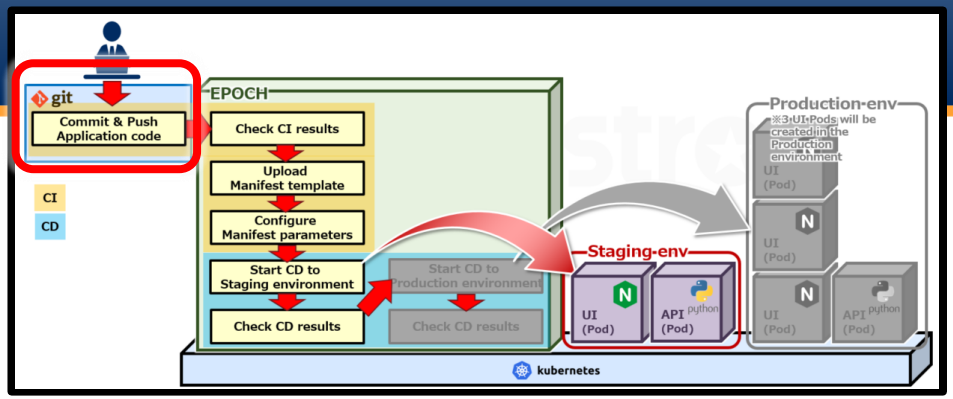
- Access the following URL to download the sample app.
  https://github.com/exastro-suite/epoch-sample-app
- From the GitHub page, press the "Code" button and "Download ZIP" in order to download the code.



- Extract the zip file and copy it to the cloned application code repositories.



4.5 CI/CD Workflow No.1 (3/25)



- Commit and Push to the Cloned application code repository.

  The commando prompt should look something like this:

```
>  git add .
>  git commit -m "first build"
>  git push origin main
```

※ If you are asked to authenticate yourself when you are running the git push command, input the necessary GitHub account information.

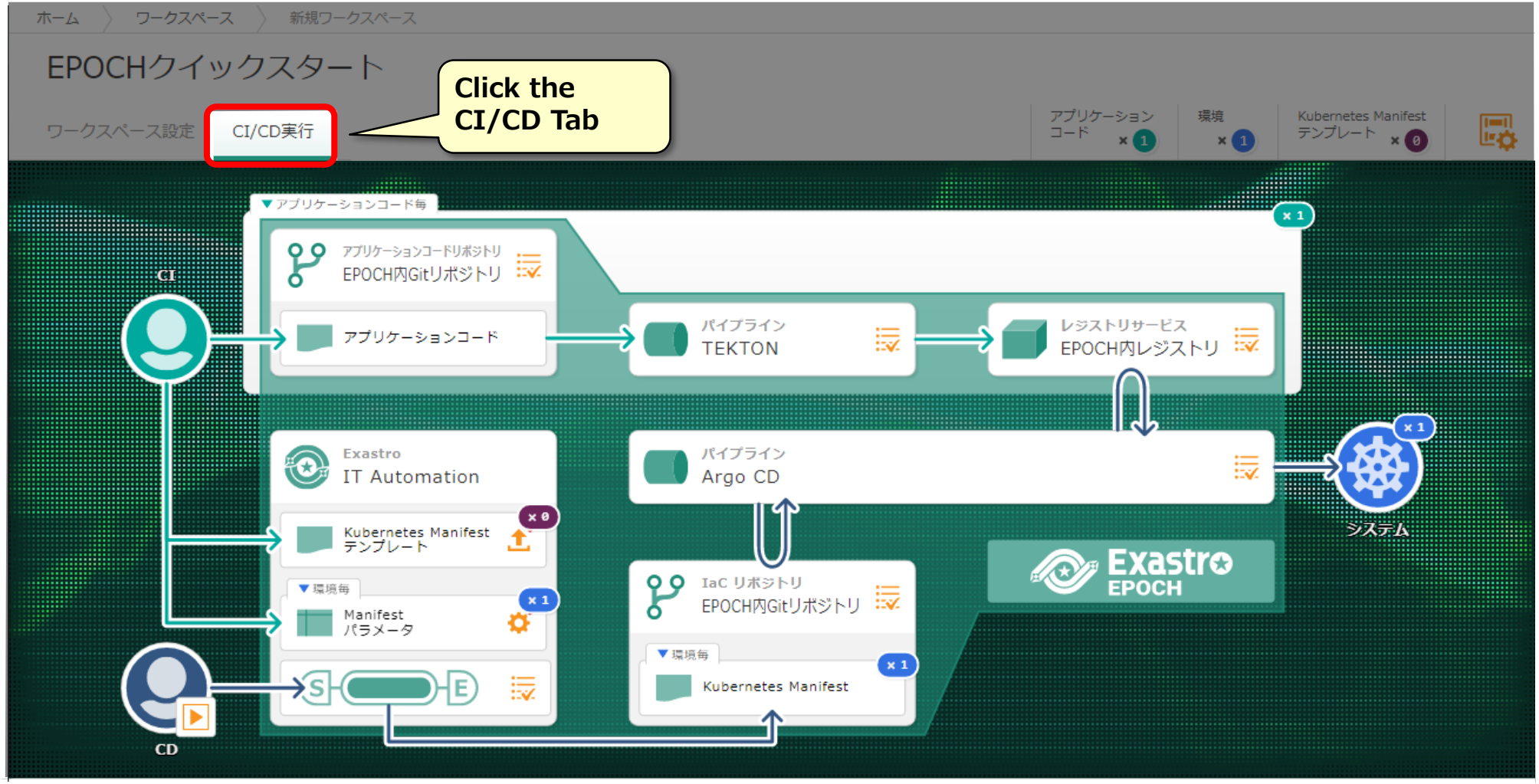
When you have successfully run the Push command, The CI Pipeline set by the TEKTON Pipeline should automatically start.

Now we need to check the CI Pipeline results.

4.5 CI/CD Workflow No.1 (4/25)

CI/CD Execution screen

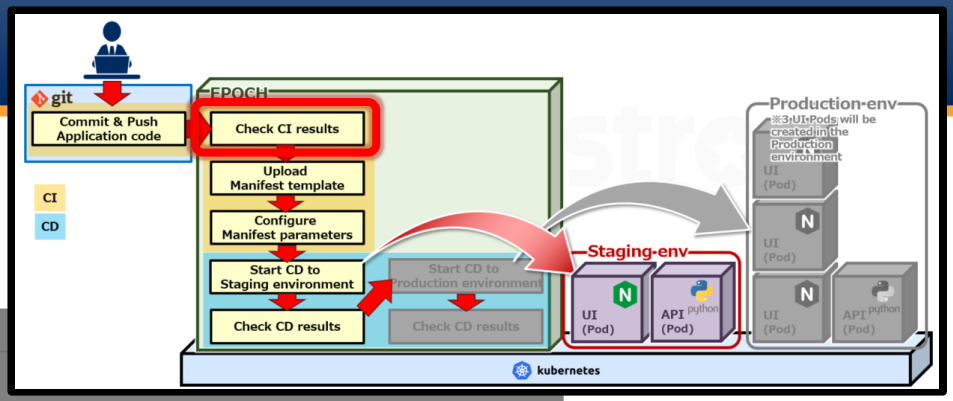
Press the CI/CD Tab in the Workspace screen to go to the CI/CD Execution screen.

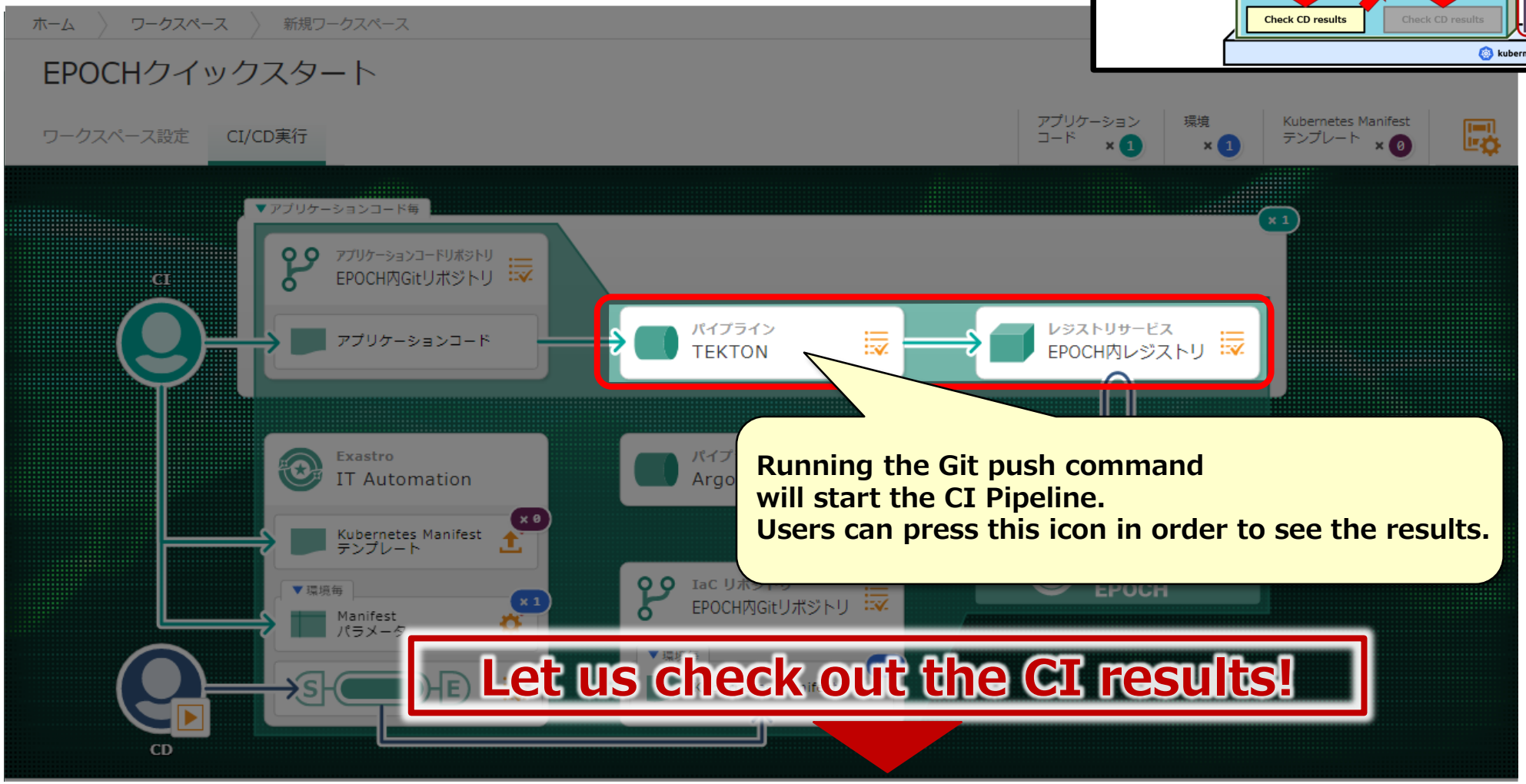


4.5 CI/CD Workflow No.1 (5/25)

Check CI results

- Check the build results of the Application code.

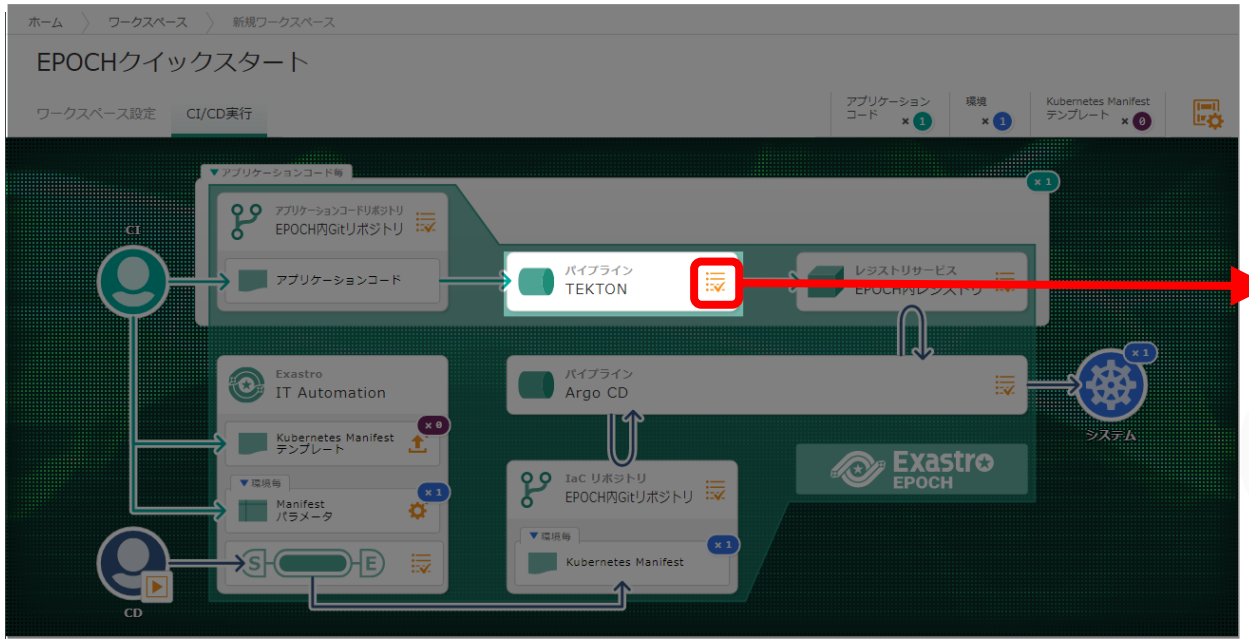


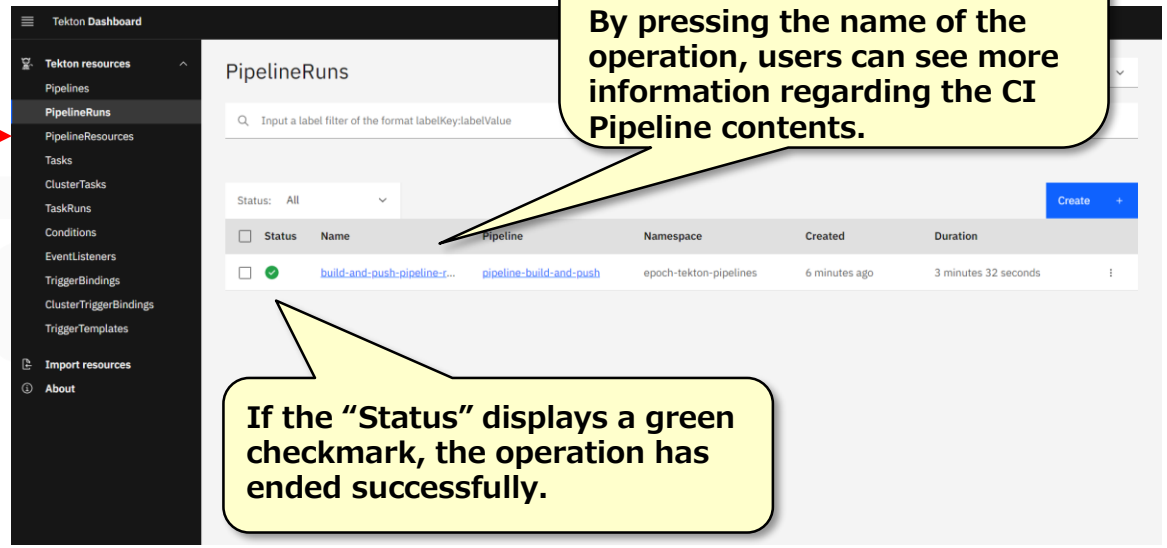


4.5 CI/CD Workflow No.1 (6/25)

Check the TEKTON Pipeline

- Check the TEKTON pipeline that the build has ended successfully.





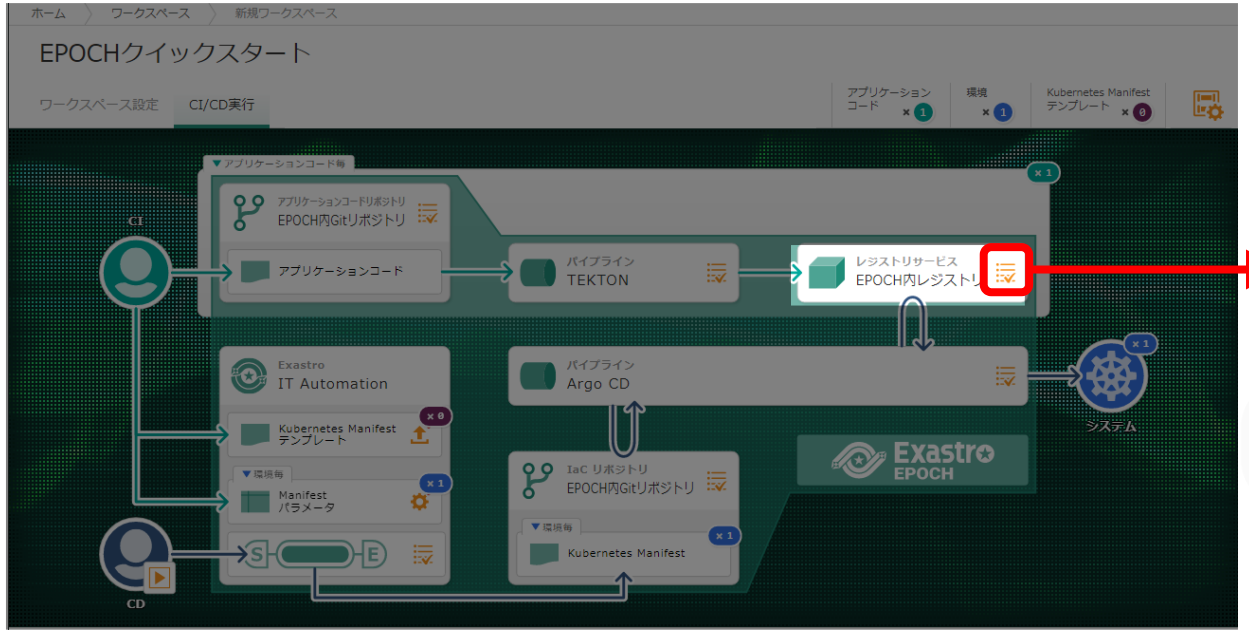
Users can check the CI Pipeline operations from the TEKTON dashboard.
By pressing the name of the operation, users can see more information regarding the CI Pipeline contents.

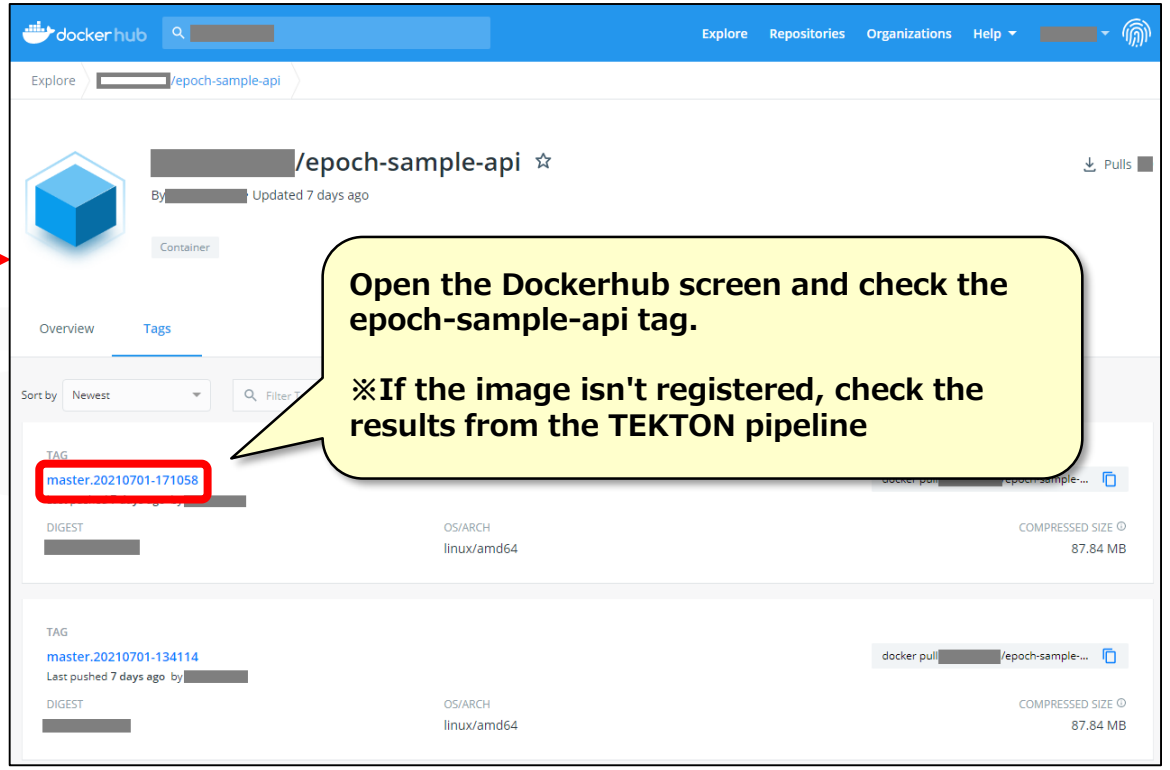
If the “Status” displays a green checkmark, the operation has ended successfully.

4.5 CI/CD Workflow No.1 (7/25)

Check the Container image tag name

- Open the registry service screen and check the tag of the built container image.





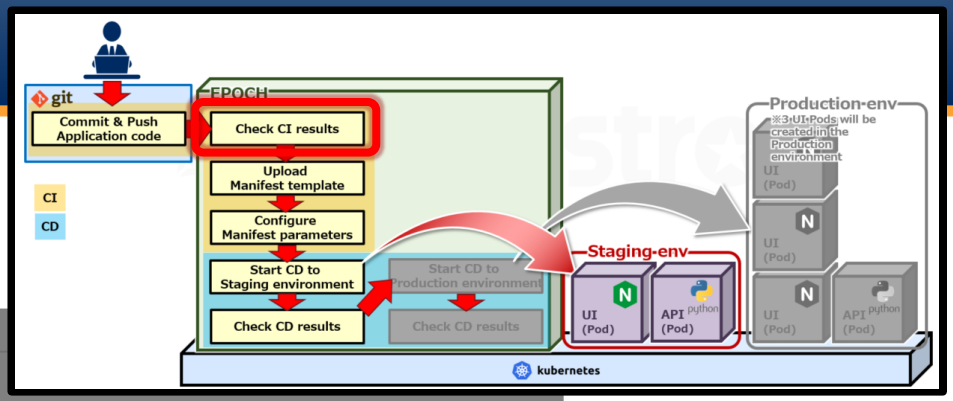
Open the Dockerhub screen and check the epoch-sample-api tag.

※If the image isn't registered, check the results from the TEKTON pipeline

The Image tag will be used in the next step where we input it into the Manifest parameter (image_tag), so make sure to save it.

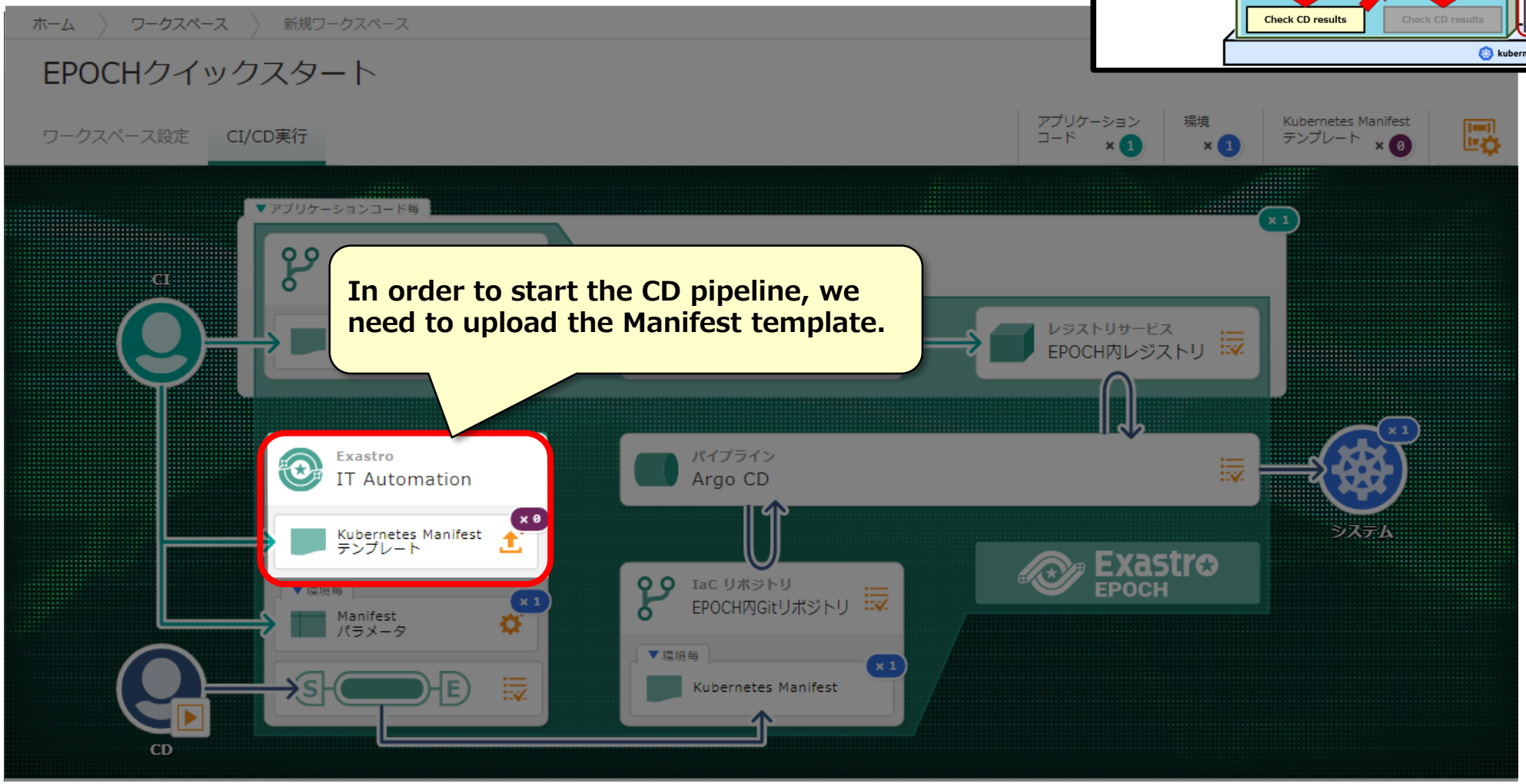
※We plan to make it possible for users to select the image_tag created in the Pipeline.

4.5 CI/CD Workflow No.1 (8/25)



Upload Manifest template

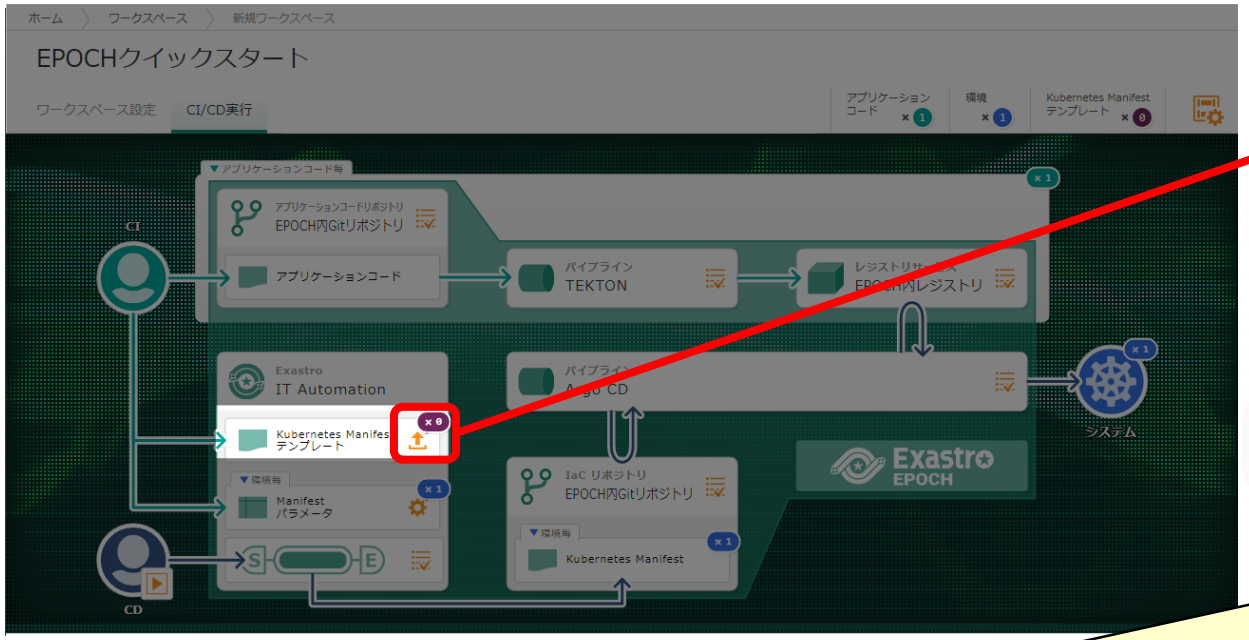
- Upload the Manifest template we downloaded earlier

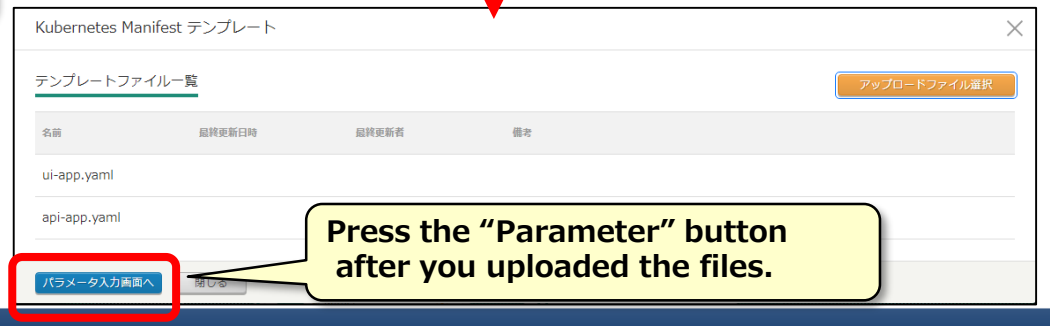


4.5 CI/CD Workflow No.1 (9/25)

Upload Manifest template

- Upload the Manifest template file we prepared earlier.





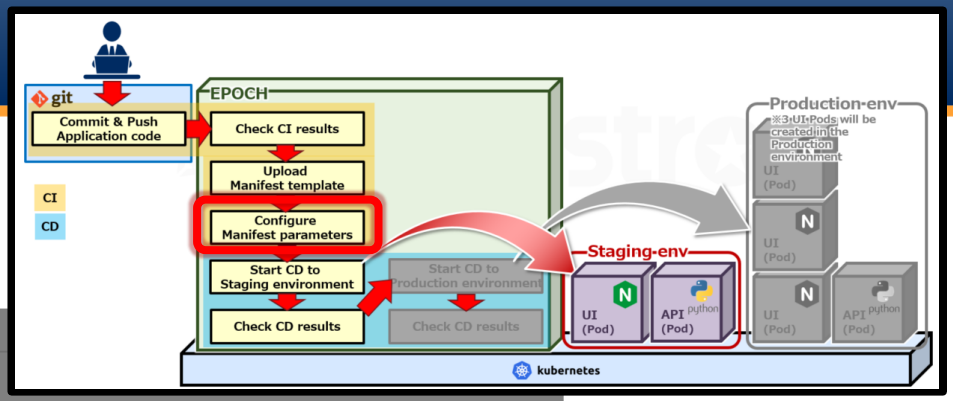
Drop the api-app.yaml and ui-app.yaml files to the area on the right

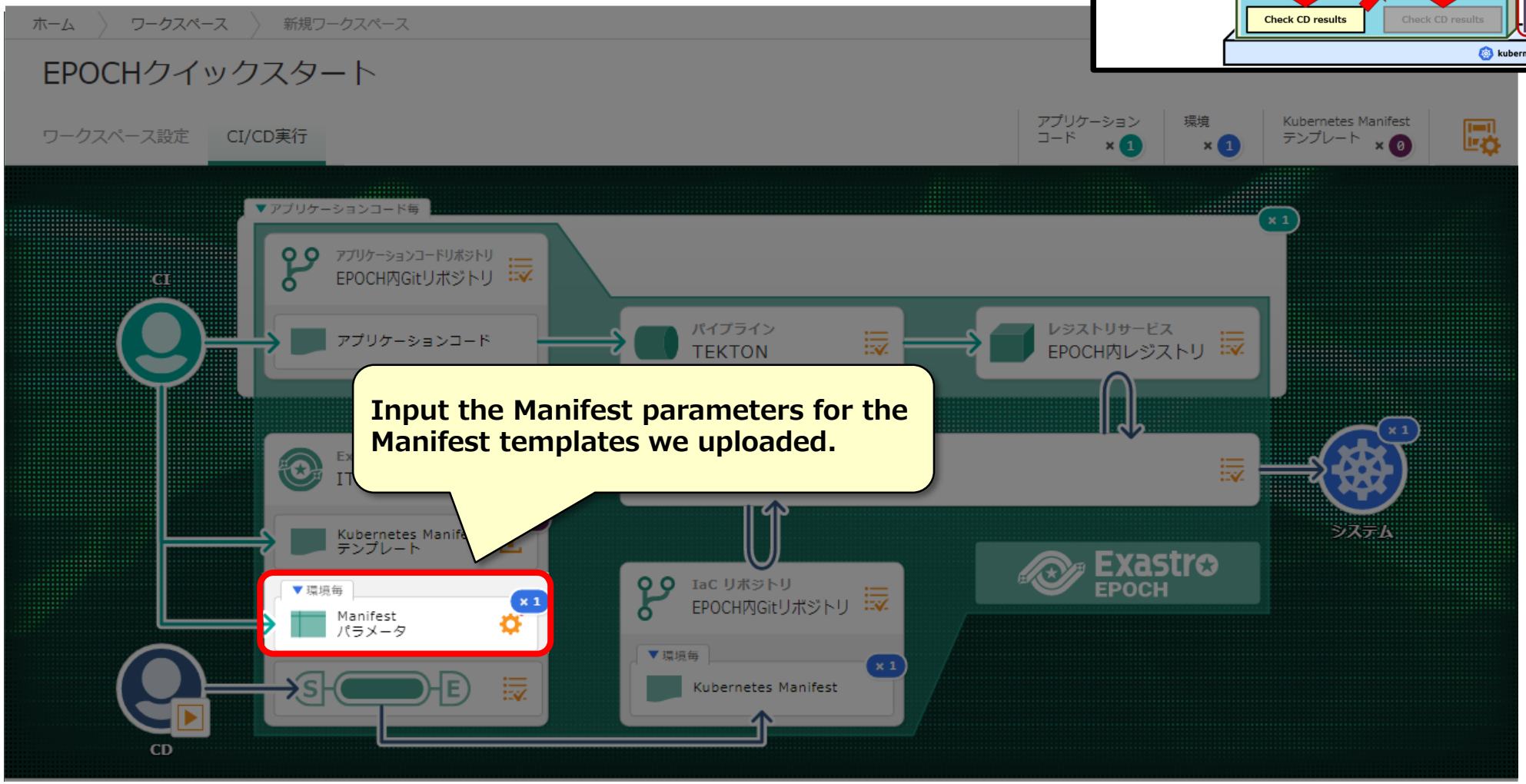
Press the “Parameter” button after you uploaded the files.

4.5 CI/CD Workflow No.1 (10/25)

Manifest parameters

- Input the Manifest parameters needed for deploying.

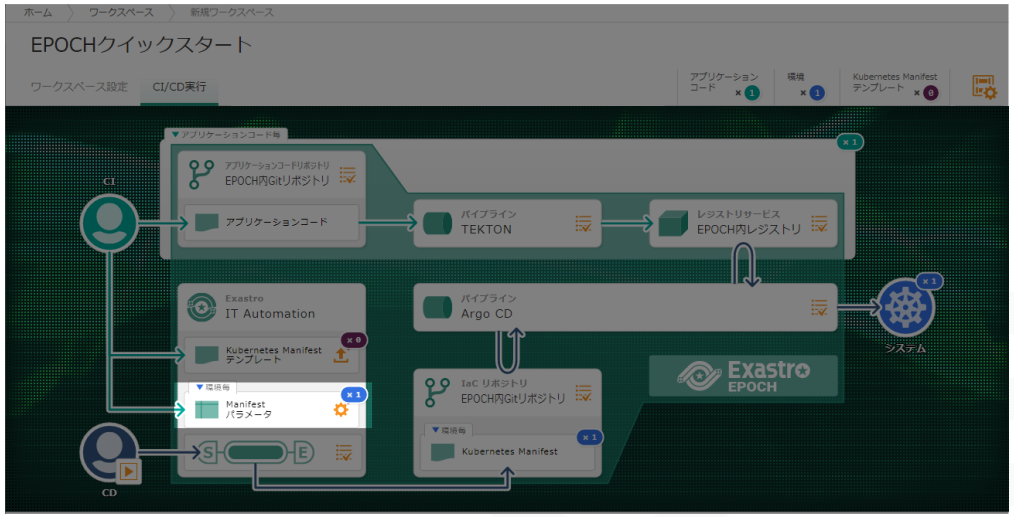


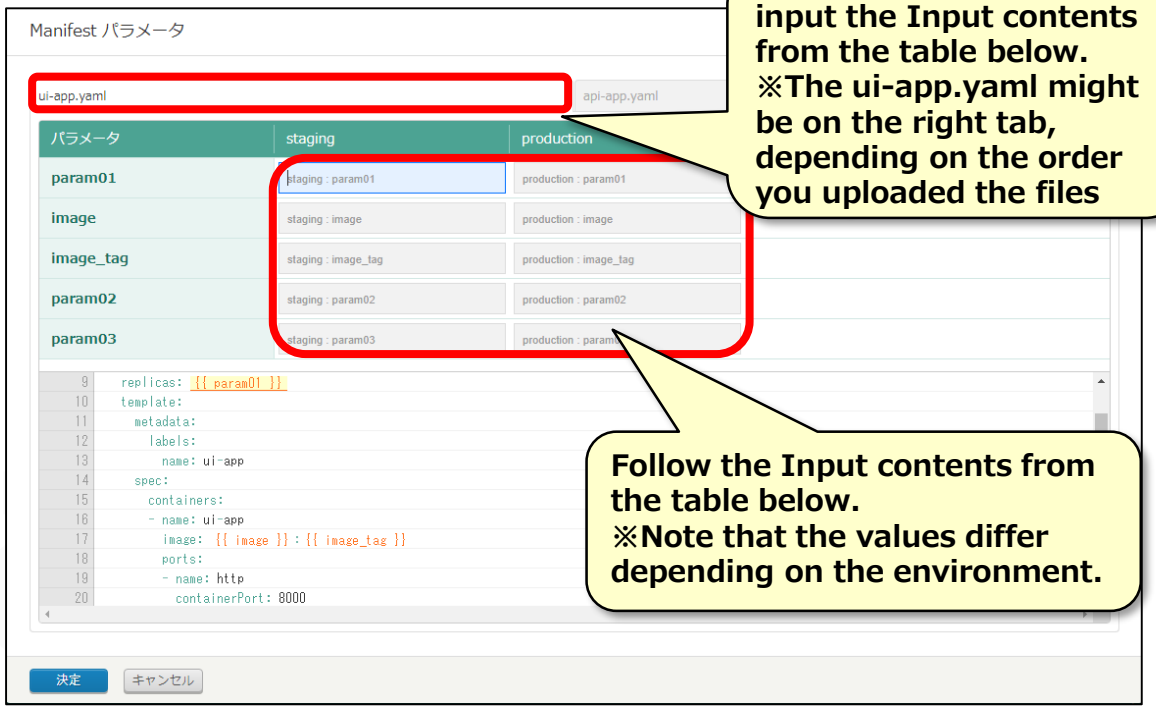


4.5 CI/CD Workflow No.1 (11/25)

Input Manifest parameters

- Follow the table below and input the Manifest parameters





Select ui-app.yaml and input the Input contents from the table below.
※The ui-app.yaml might be on the right tab, depending on the order you uploaded the files

Follow the Input contents from the table below.
※Note that the values differ depending on the environment.

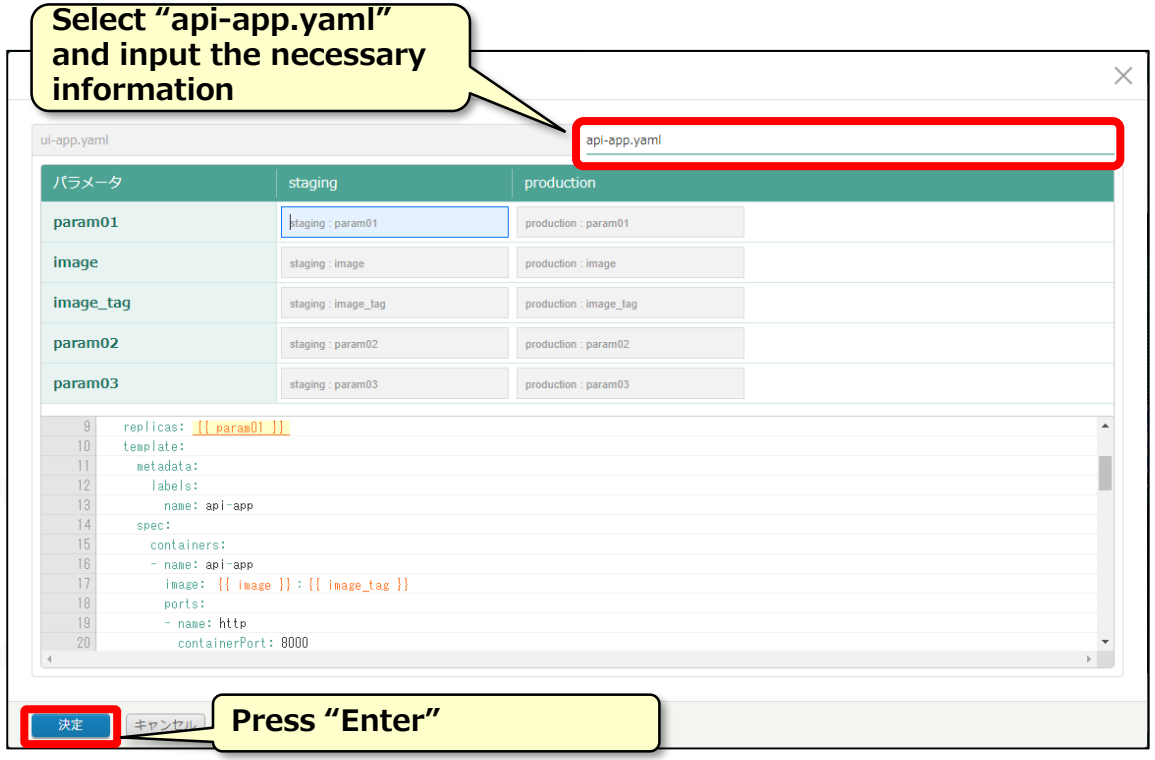
ui-app.yaml

Item	Input contents(staging)	Input contents(production)	Description
{{ param01 }}	1	3	Replica numbers
{{ image }}	exastro/epochsampleappui	exastro/epochsampleappui	Container image
{{ image_tag }}	master.20210708183910	master.20210708183910	Container image tag
{{ param02 }}	31001	31003	Port number for the blue side (Blue-Green deployment)
{{ param03 }}	32001	32003	Port number for the green side (Blue-Green deployment)

※The image_tag for ui-app.yaml is the same one from the container image we built earlier.
※We plan to make "image" and "image_tag" selection items in the future.

4.5 CI/CD Workflow No.1 (12/25)

- Change to the api-app.yaml tab and repeat.



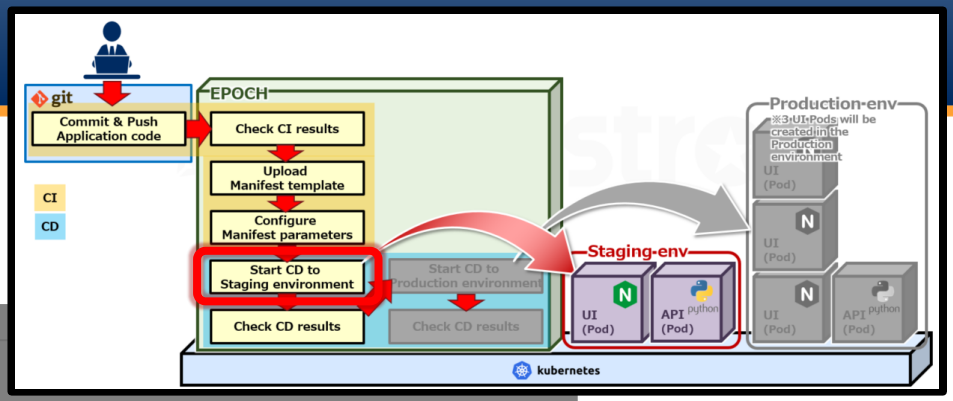
api-app.yaml

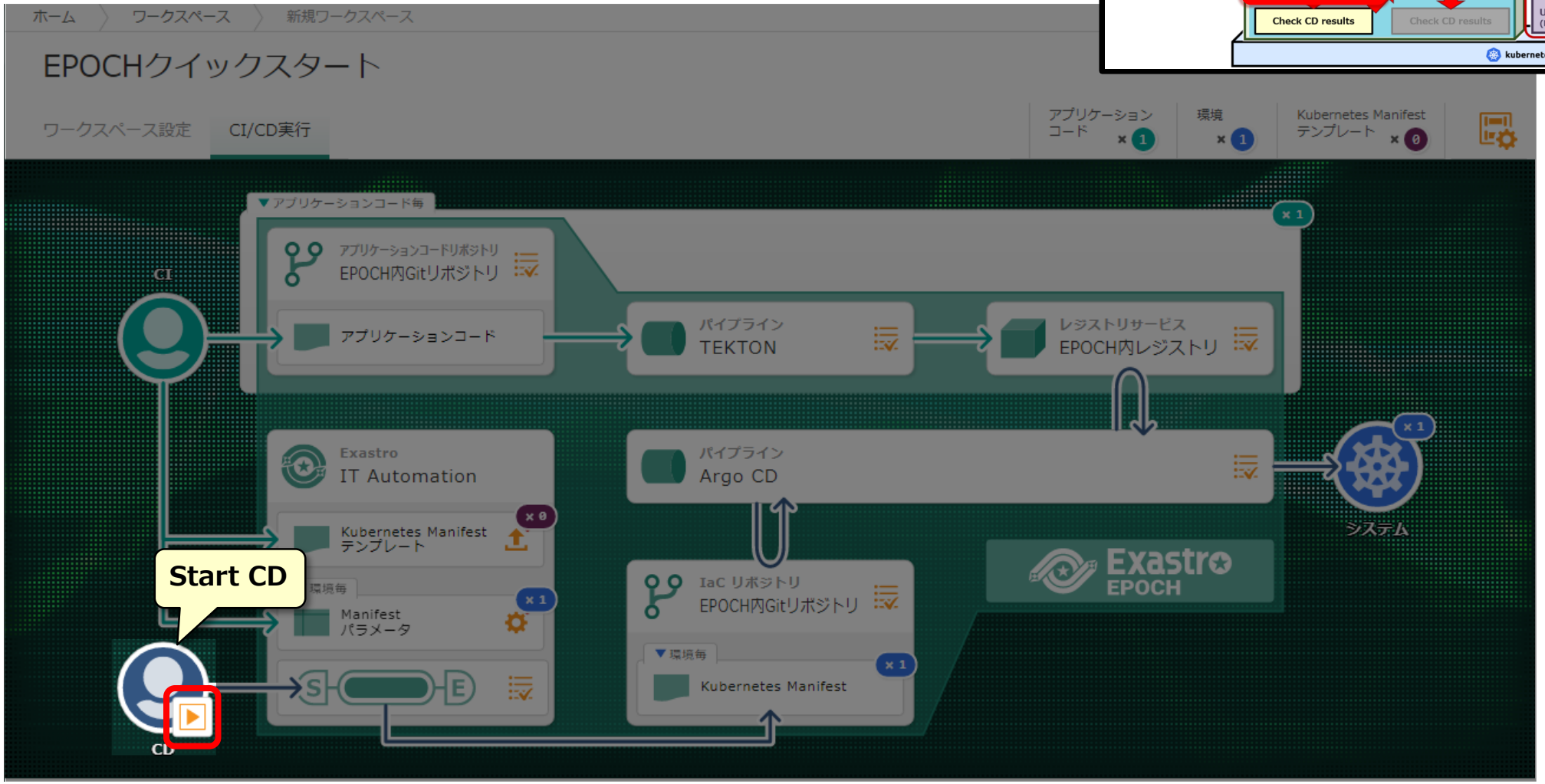
Item	Input contents(staging)	Input contents(production)	Description
{{ param01 }}	1	1	Replica number
{{ image }}	[Dockerhub account name]/epoch-sample-api	[Dockerhub account name]/epoch-sample-api	Container image
{{ image_tag }}	[Image tag name]	[Image tag name]	Container image tag acquired from the Registry service.
{{ param02 }}	31002	31004	Port number for the blue side (Blue-Green deployment)
{{ param03 }}	32002	32004	Port number for the green side (Blue-Green deployment)

4.5 CI/CD Workflow No.1 (13/25)

Deploy to Staging environment

- Start CD and deploy to the Staging environment.

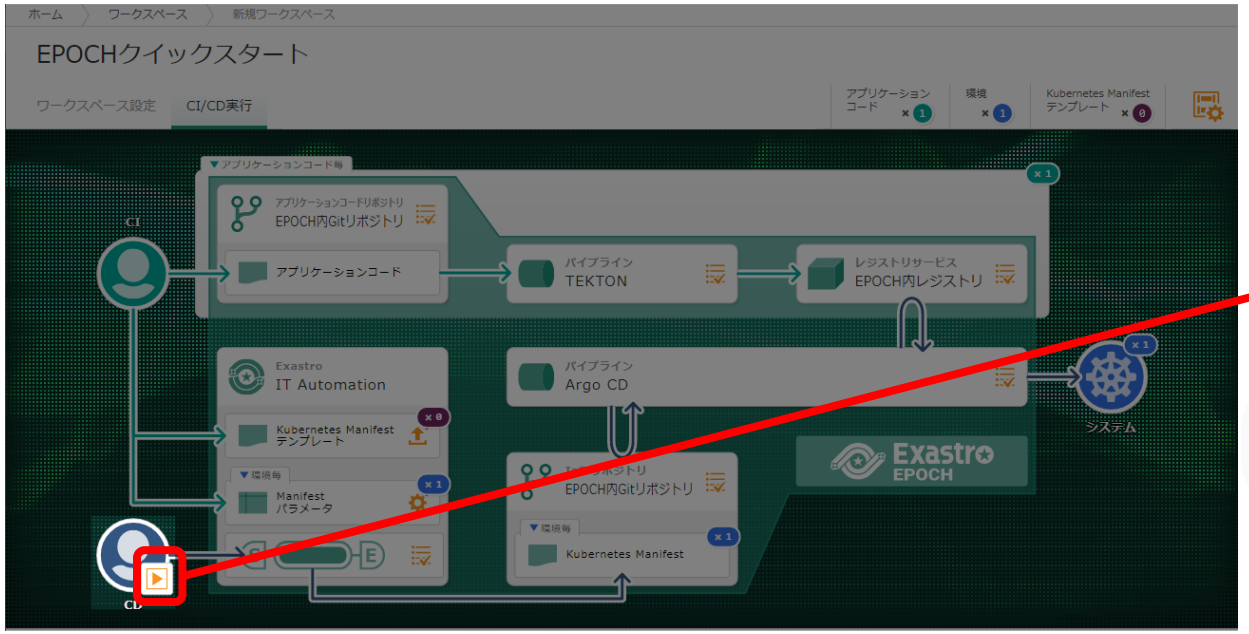




4.5 CI/CD Workflow No.1 (14/25)

Start CD for the Staging environment

- Select the environment you want to deploy to.

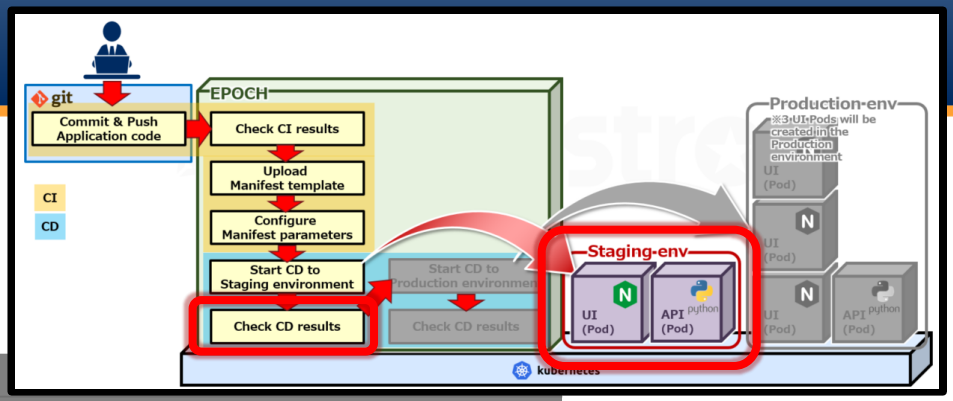


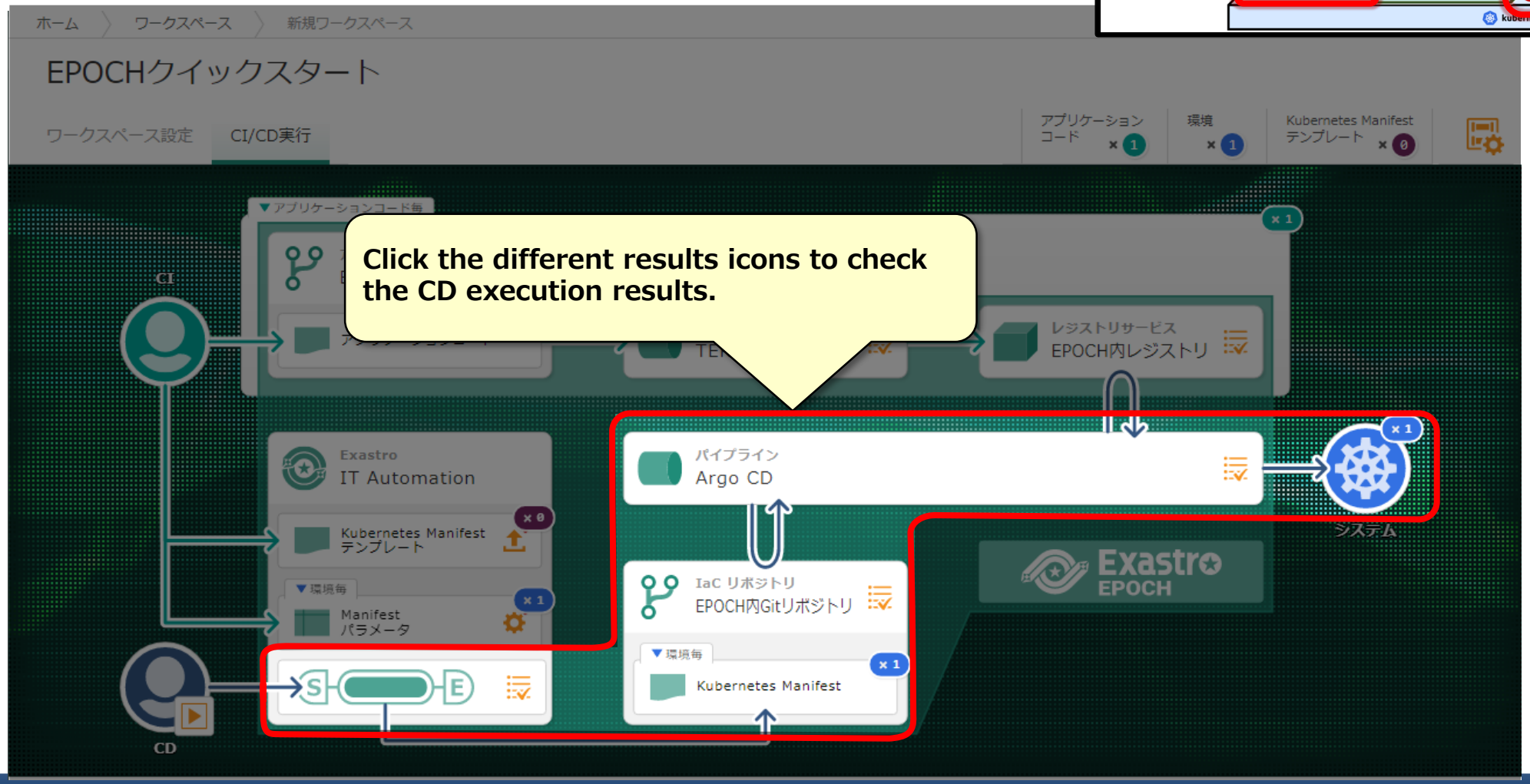
The CD process for the Staging environment has started.
Let us check that it is working as it should.

4.5 CI/CD Workflow No.1 (15/25)

Check the CD results

- We can check the CD execution results from Exastro IT-Automation and ArgoCD.

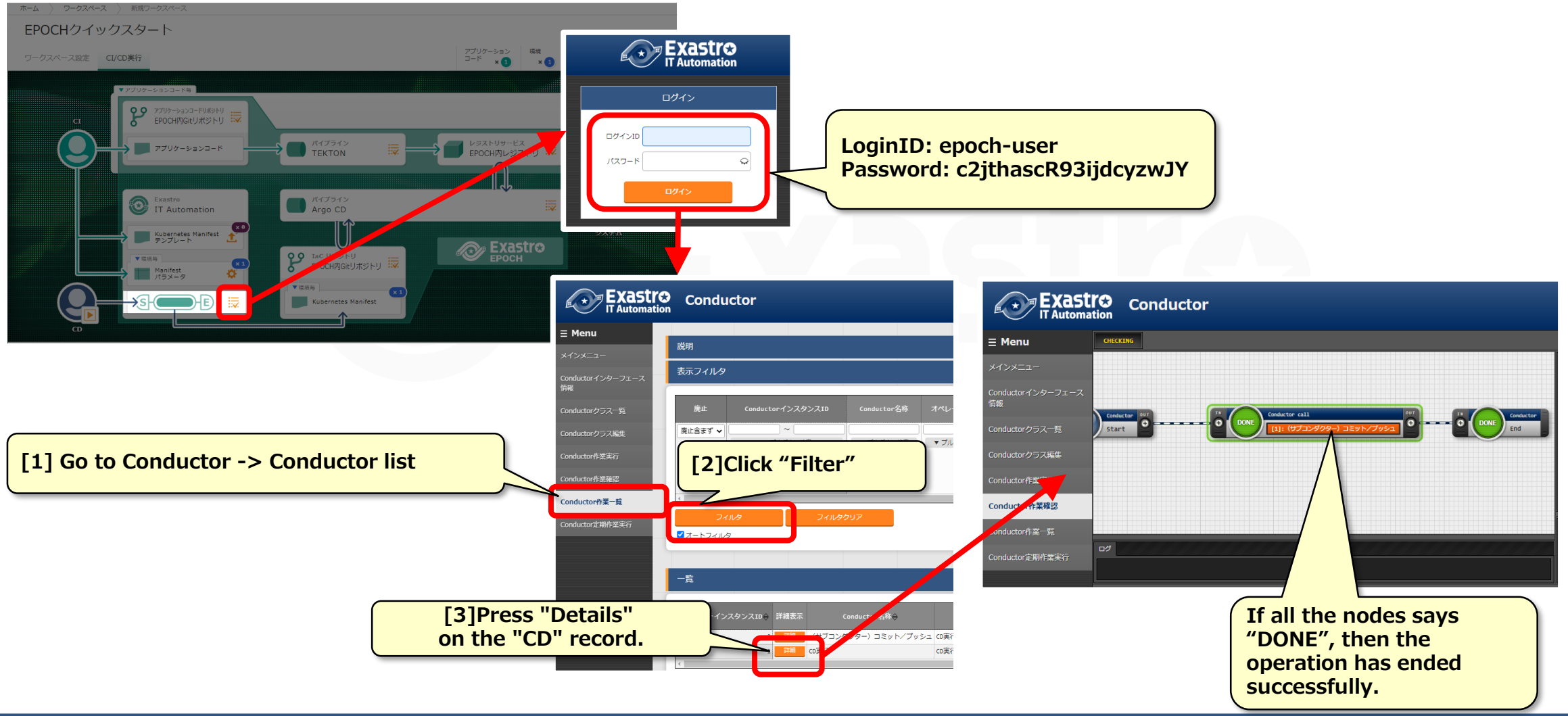




4.5 CI/CD Workflow No.1 (16/25)

Check that the Manifest file has been created (Staging environment)

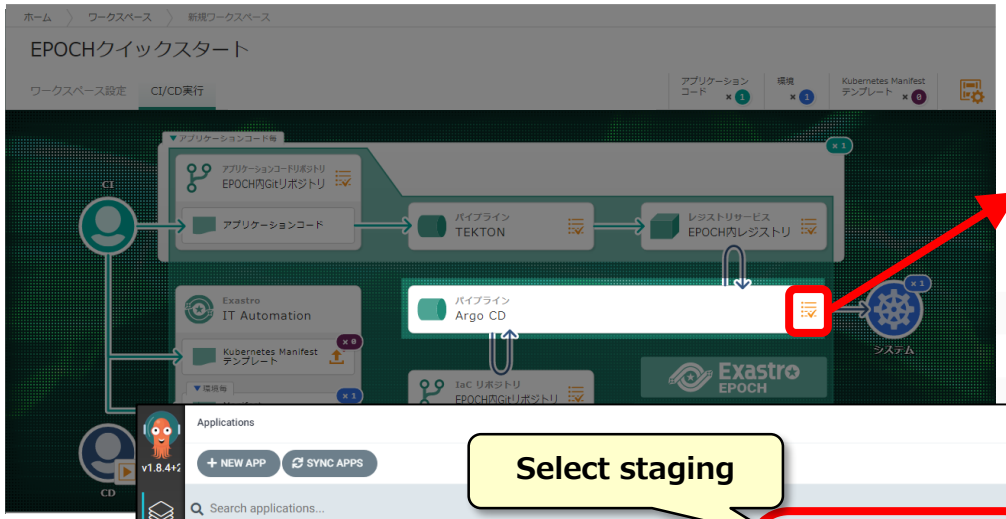
- Open Exastro IT-Automation and check that the Manifest file has been registered to the IaC repository.

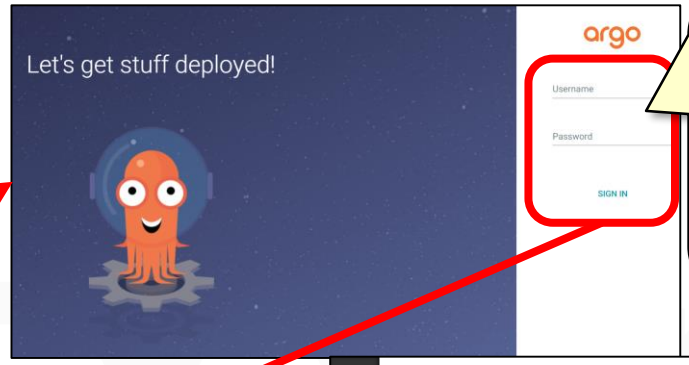


4.5 CI/CD Workflow No.1 (17/25)

Check the results of the ArgoCD Pipeline (Staging environment)

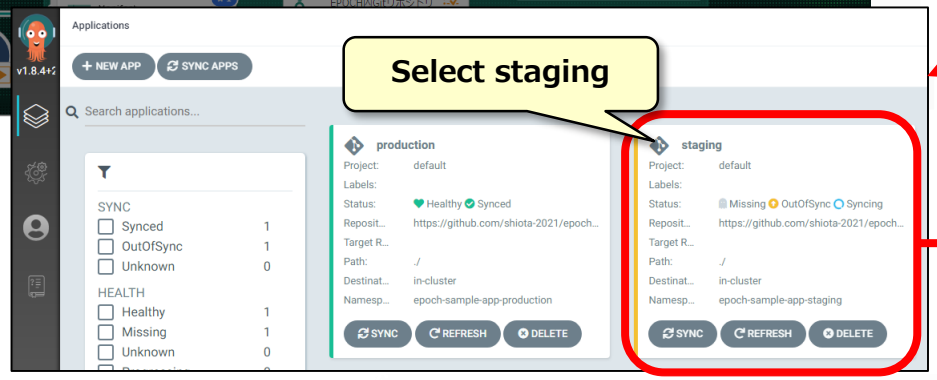
- Check that the Manifest shows up on Kubernetes



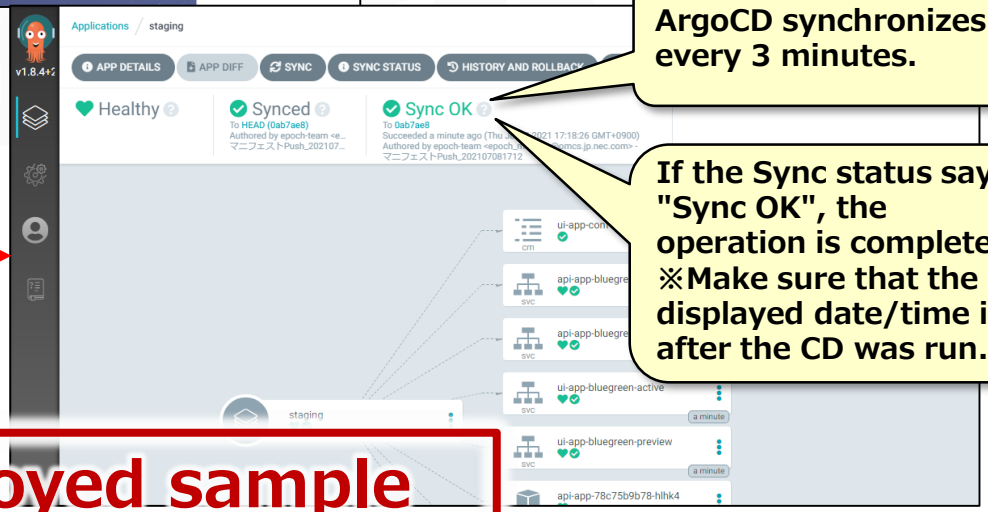


Username: admin
Password: iYSCKzx2wvxJnn4dCNwN

Your browser might say that the connection is not secure. If that is the case, open up the “more information” tab and press “Access”.



Select staging



ArgoCD synchronizes every 3 minutes.

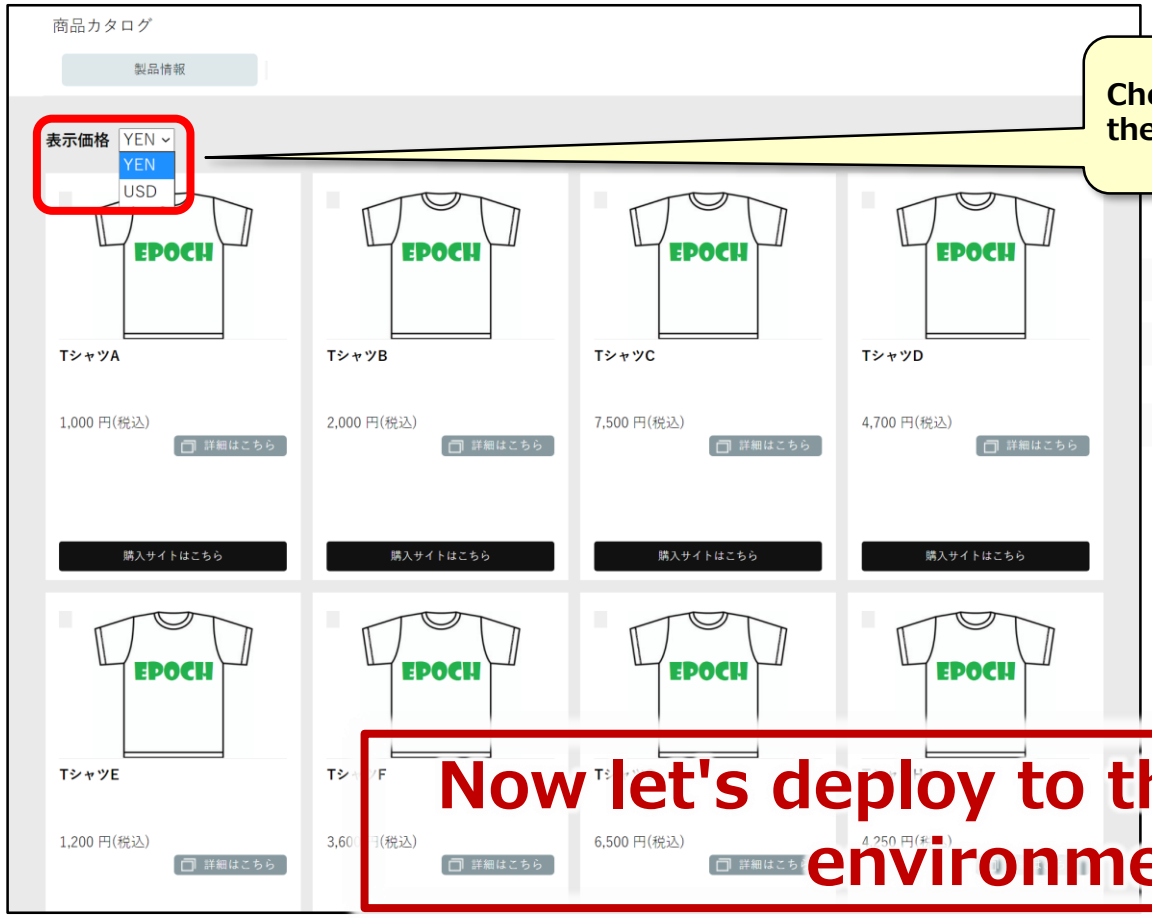
If the Sync status says "Sync OK", the operation is complete.
※Make sure that the displayed date/time is after the CD was run.

Let us check the deployed sample application

4.5 CI/CD Workflow No.1 (18/25)

Check the Staging environment sample code.

- Access the sample application from the URL below
  http://[Kubernetes master node IP address/Host name]:31001/front-end.html



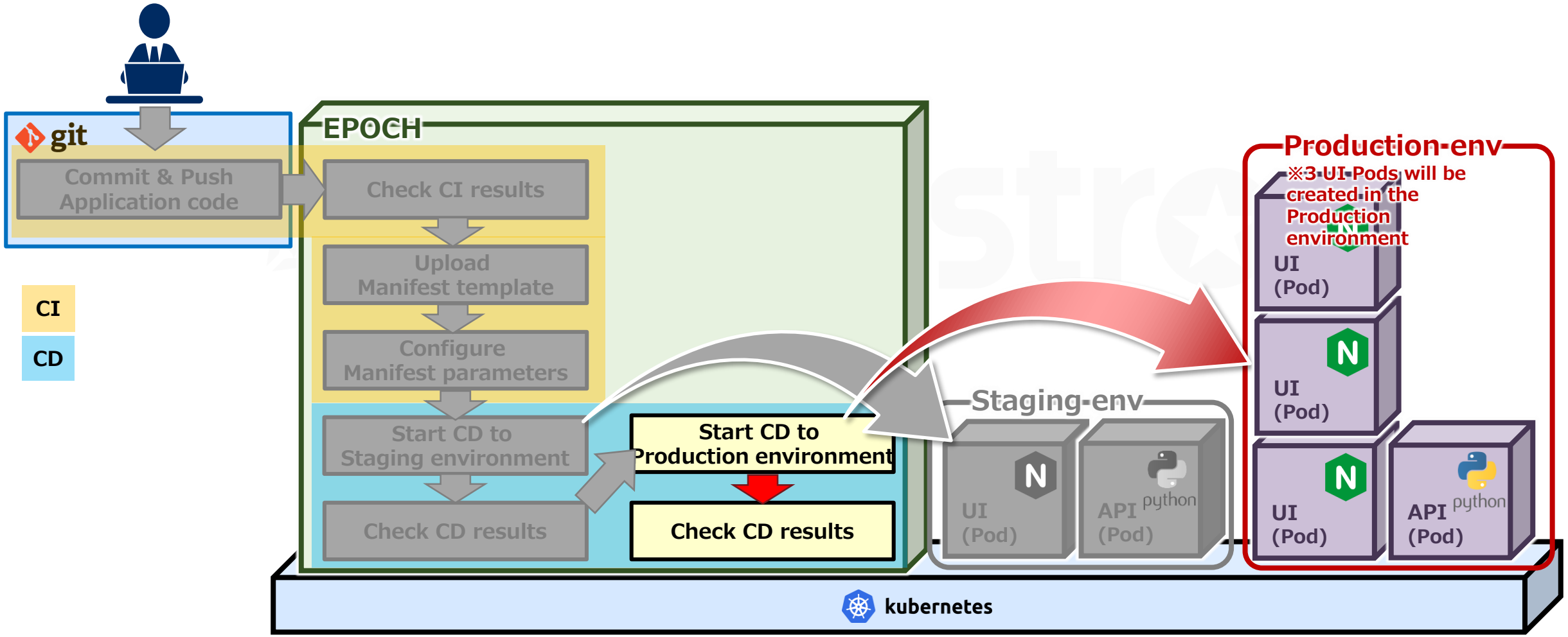
Check that both "YEN" and "USD" is displayed and selectable from the Currency pulldown selection.

Now let's deploy to the Production environment

4.5 CI/CD Workflow No.1 (19/25)

Deploy to Production environment

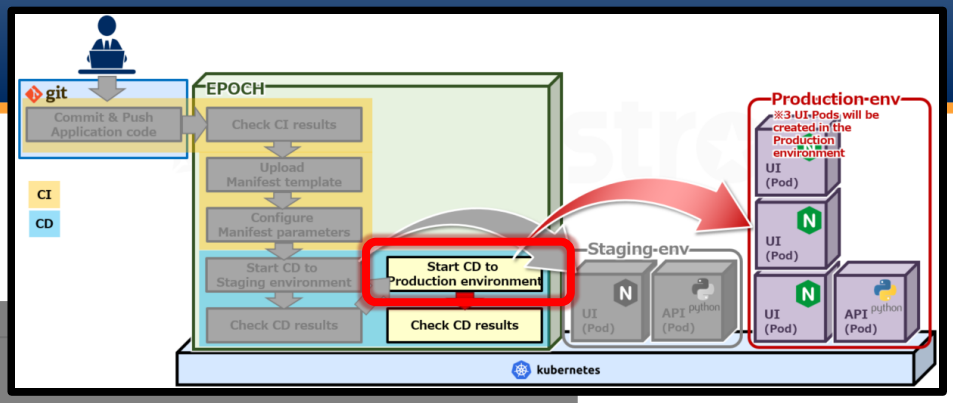
Now that we have deployed to the Staging environment and checked that everything functions as it should, we can now deploy to the Production environment.

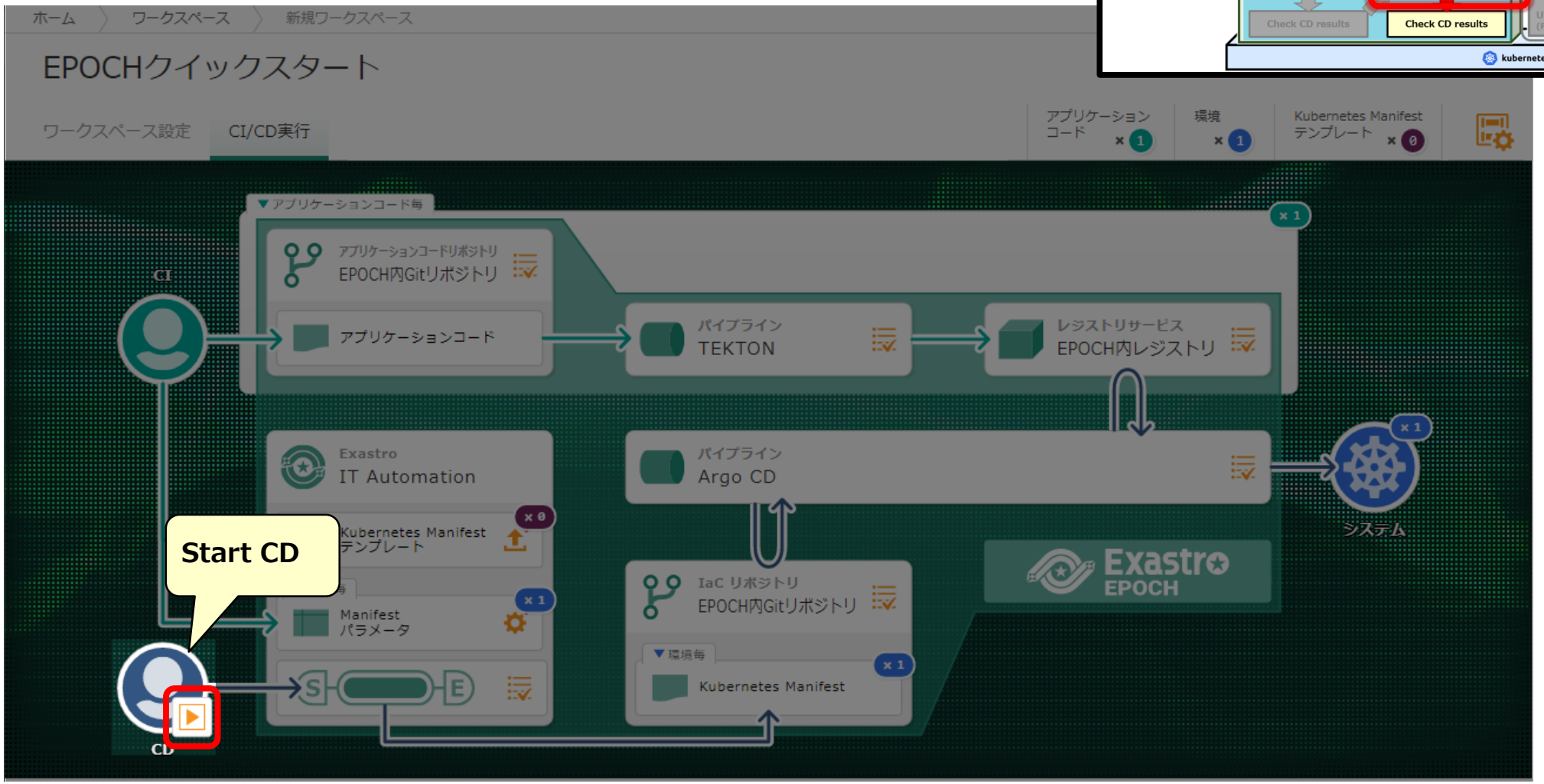


4.5 CI/CD Workflow No.1 (20/25)

Deploy to the Production environment

- Start the CD and deploy to the Production environment.

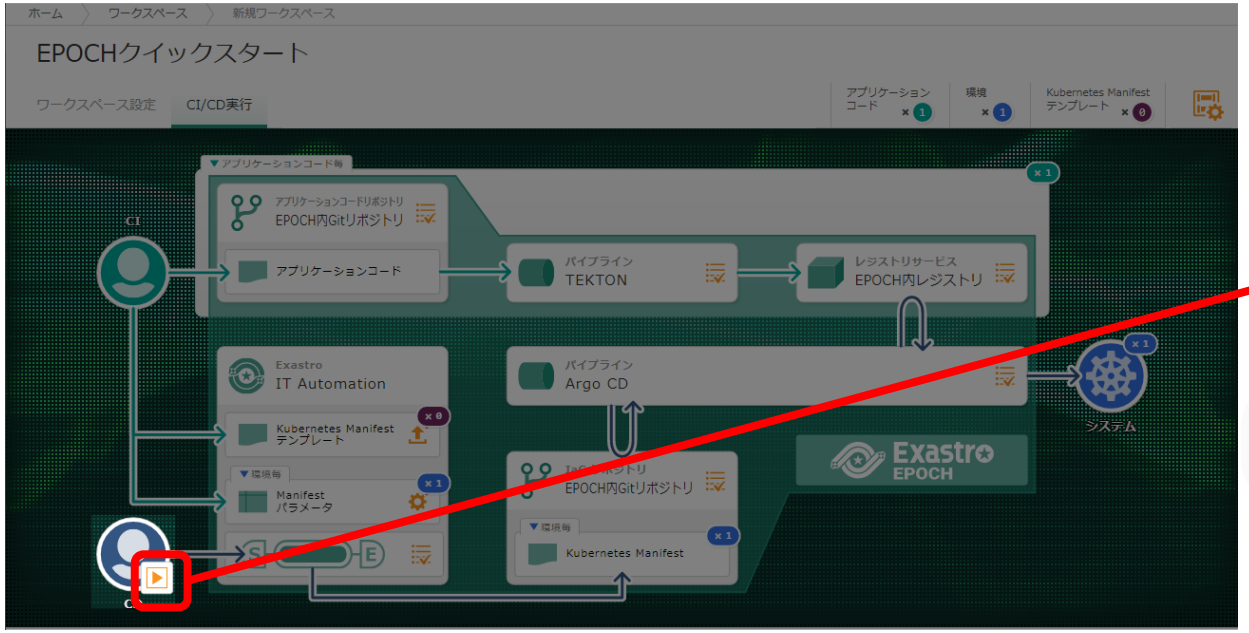




4.5 CI/CD Workflow No.1 (21/25)

Start CD for the Production environment

- Select the environment you want to deploy to.

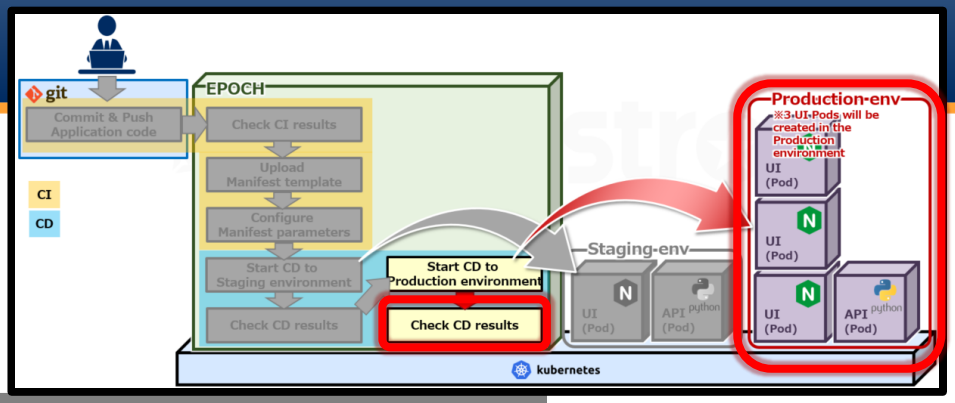


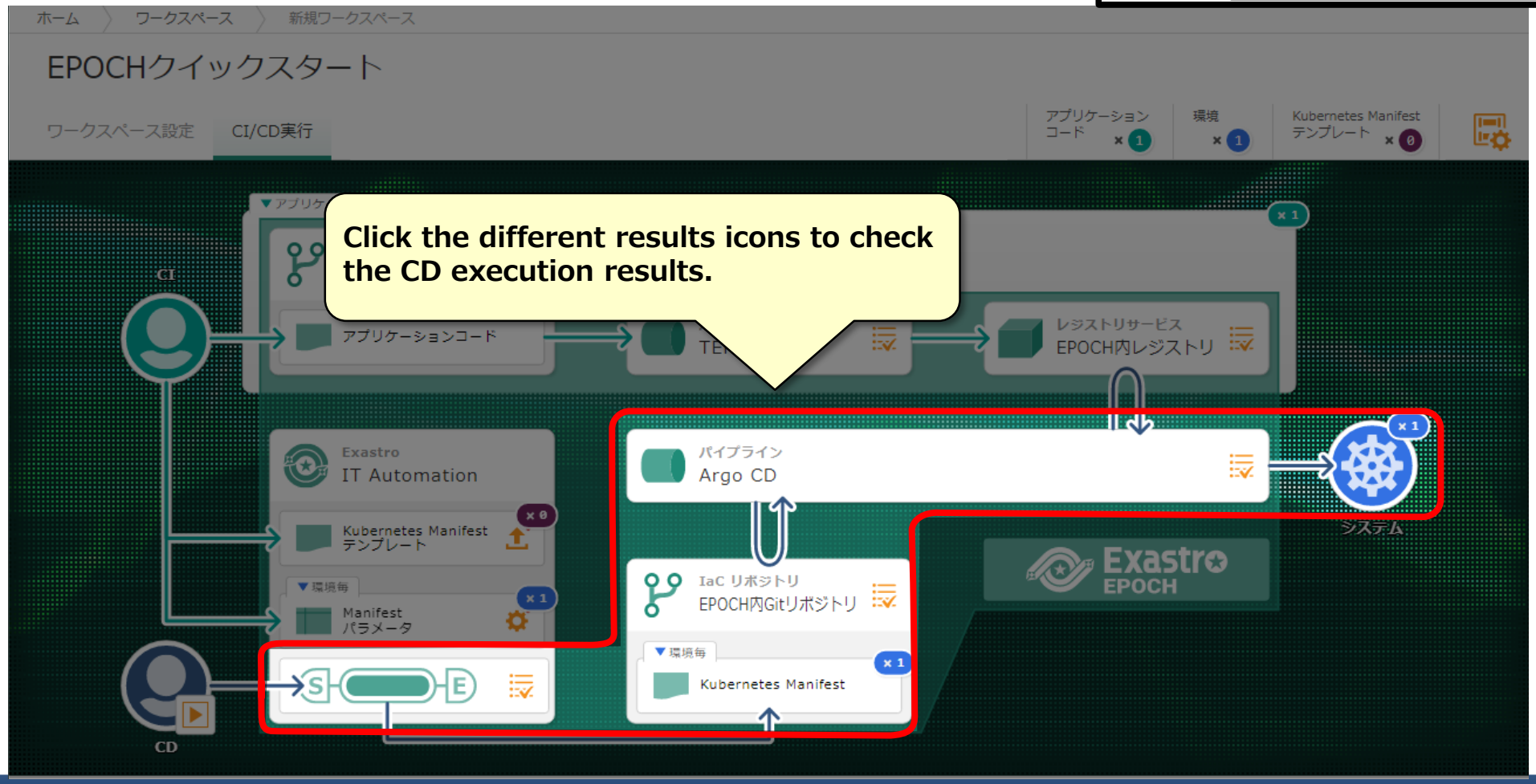
The CD process for the Production environment has started.
Let us check that it is working as it should.

4.5 CI/CD Workflow No.1 (22/25)

Check the CD results (Production)

- We can check the CD execution results from Exastro IT-Automation and ArgoCD.

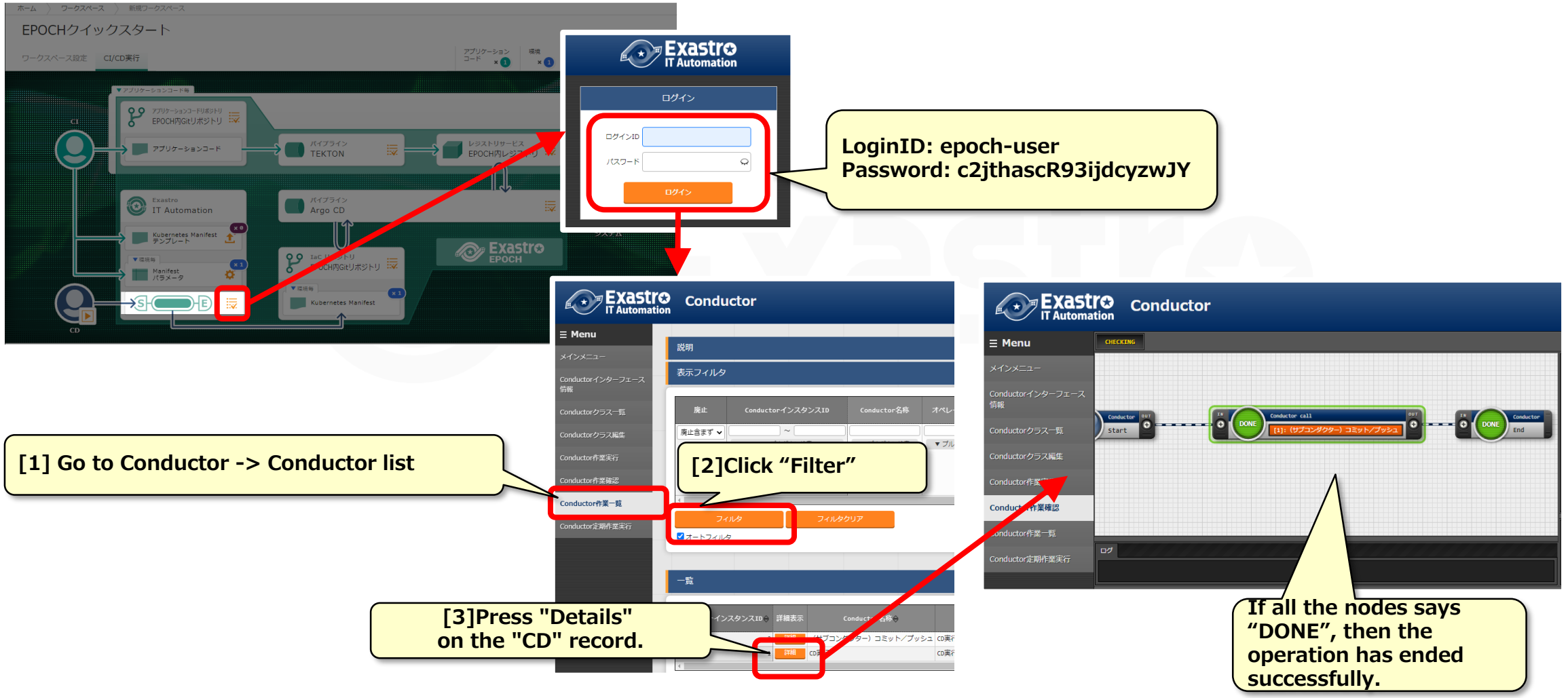




4.5 CI/CD Workflow No.1 (23/25)

Check that the Manifest file has been created (Production environment)

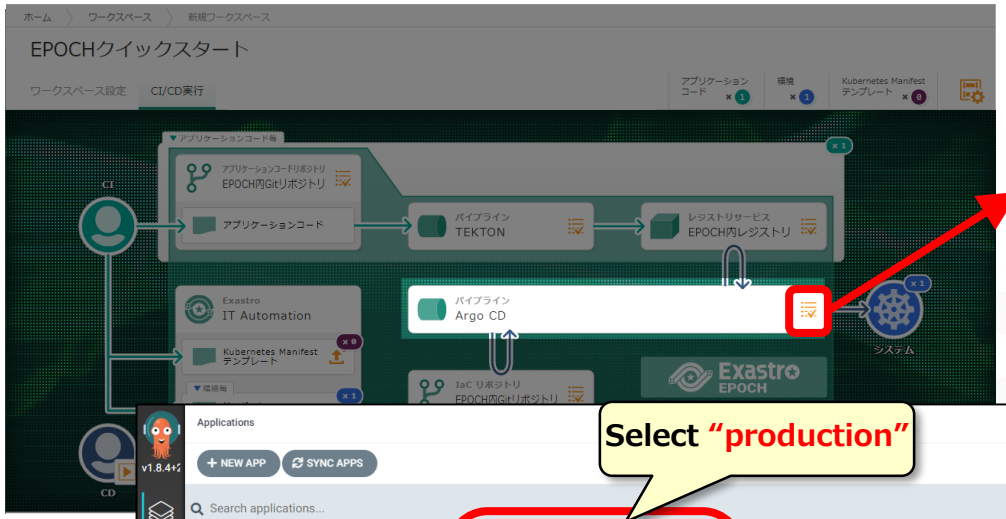
- Open Exastro IT-Automation and check that the Manifest file has been registered to the IaC repository.

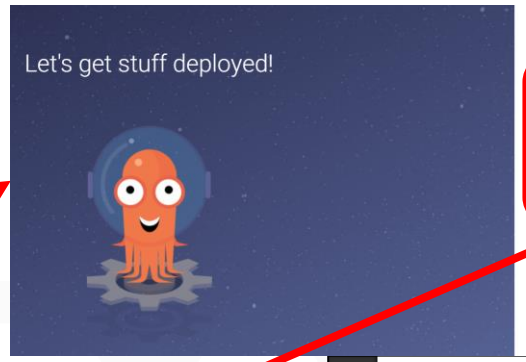


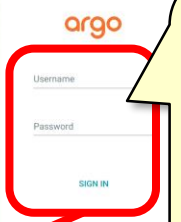
4.5 CI/CD Workflow No.1 (24/25)

Check the results of the ArgoCD Pipeline (Production environment)

- Check that the Manifest shows up on Kubernetes

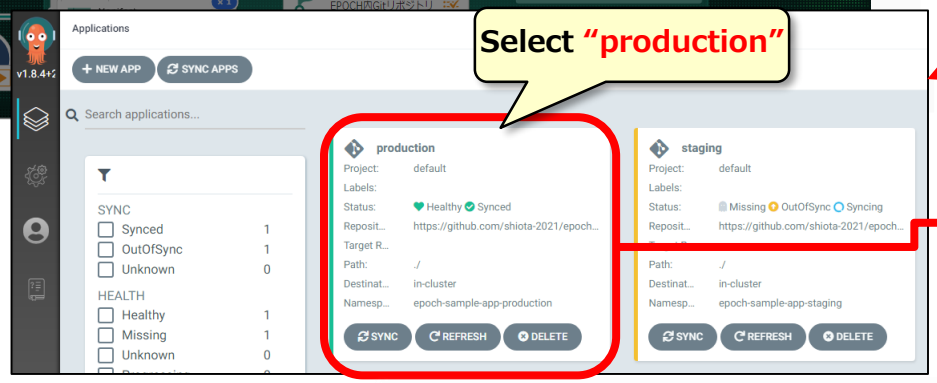




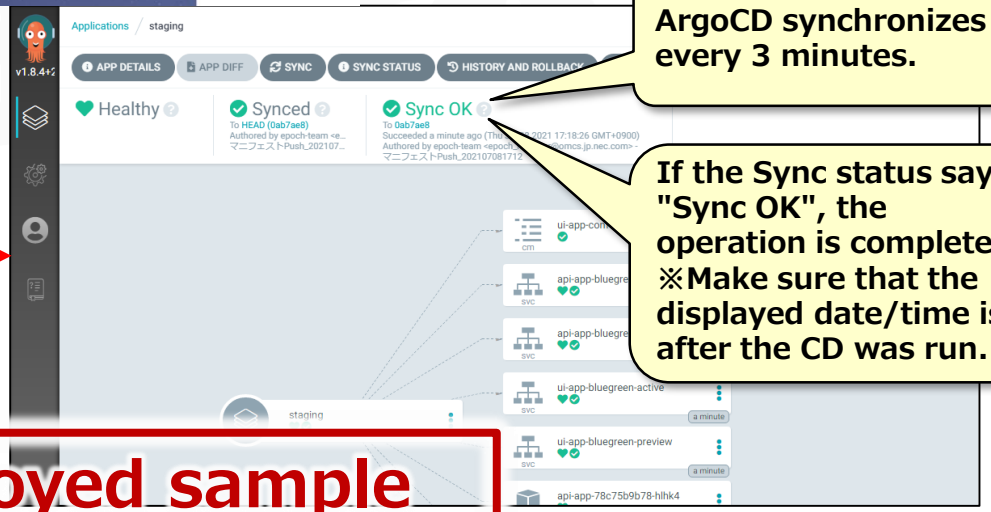


Username: admin
Password: iYSCKzx2wvxJnn4dCNwN

Your browser might say that the connection is not secure. If that is the case, open up the “more information” tab and press “Access”.



Select “production”



ArgoCD synchronizes every 3 minutes.

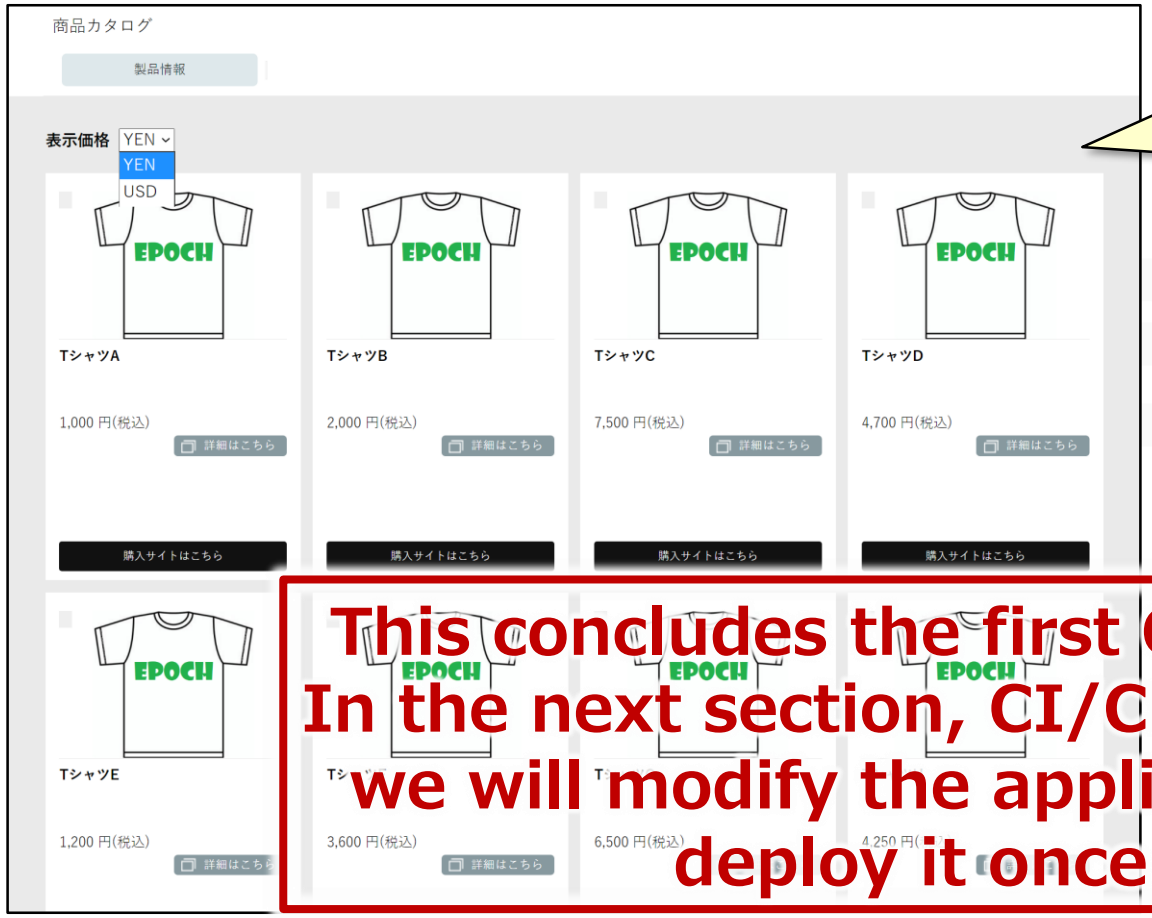
If the Sync status says "Sync OK", the operation is complete.
※Make sure that the displayed date/time is after the CD was run.

Let us check the deployed sample application

4.5 CI/CD Workflow No.1 (25/25)

Check the Production environment sample code.

- Access the sample application from the URL below
  http://[Kubernetes master node IP address/Host name]:31003/front-end.html



Check that the screen displayed is the same as the one in the Staging environment.

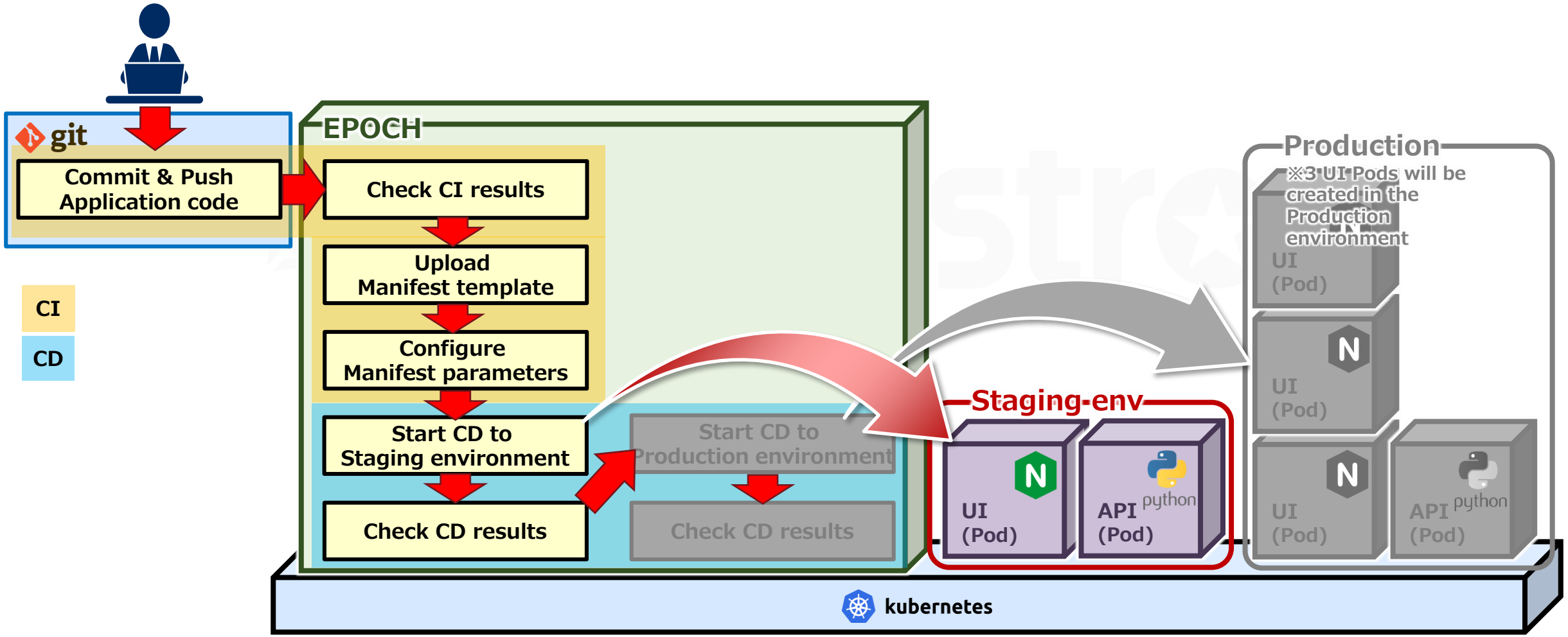
**This concludes the first CI/CD Workflow.
In the next section, CI/CD Workflow No.2, we will modify the application code and deploy it once more.**

CI/CD Workflow No.2

4.6 CI/CD Workflow No.2 (1/21)

Deploy to the Staging environment

In the second CI/CD Workflow, the user will modify the application code, commit and push it to the Staging environment and finally deploy it.



4.6 CI/CD Workflow No.2 (2/21)

Modify application code

Modify the application code and start the second CI/CD process.
In this part of the tutorial, we will make it so Euro can be selected as an currency.

Use your code editor and open the following files from the cloned application code repository and modify them.

- File ① : api-app/data/currency.json

```
 6      "USD": {
 7          "symbol"    :   "$",
 8          "formatter" :   "{symbol} {price:,.2f} (Tax Included)"
 9      },
10      "EUR": {
11          "symbol"    :   "€",
12          "formatter" :   "{symbol} {price:,.2f}"
13      }
14  }
```

Add line 9~12

- File 2② : api-app/data/rate.json

```
1  {
2      "USD": 110.56,
3      "EUR": 134.15
4  }
```

Add a comma to the end of line 2
Add line 3

4.6 CI/CD Workflow No.2 (3/21)

Commit and push the application code

Take the modified application code and commit and push it.

Run the following Git commands

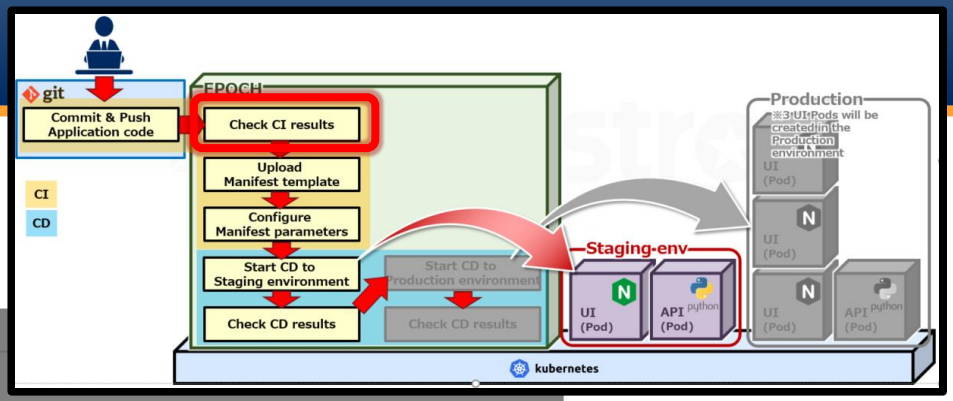
```
>  cd “[clone destination folder]"
>  cd epoch-sample-app
>  git add .
>  git commit -m “addcurrency(EUR)"
>  git push origin main
```

※ If you are asked to authenticate yourself when you are running the git push command, input the necessary GitHub account information.

4.6 CI/CD Workflow No.2 (4/21)

Check CI results

- Check the build results of the Application code.

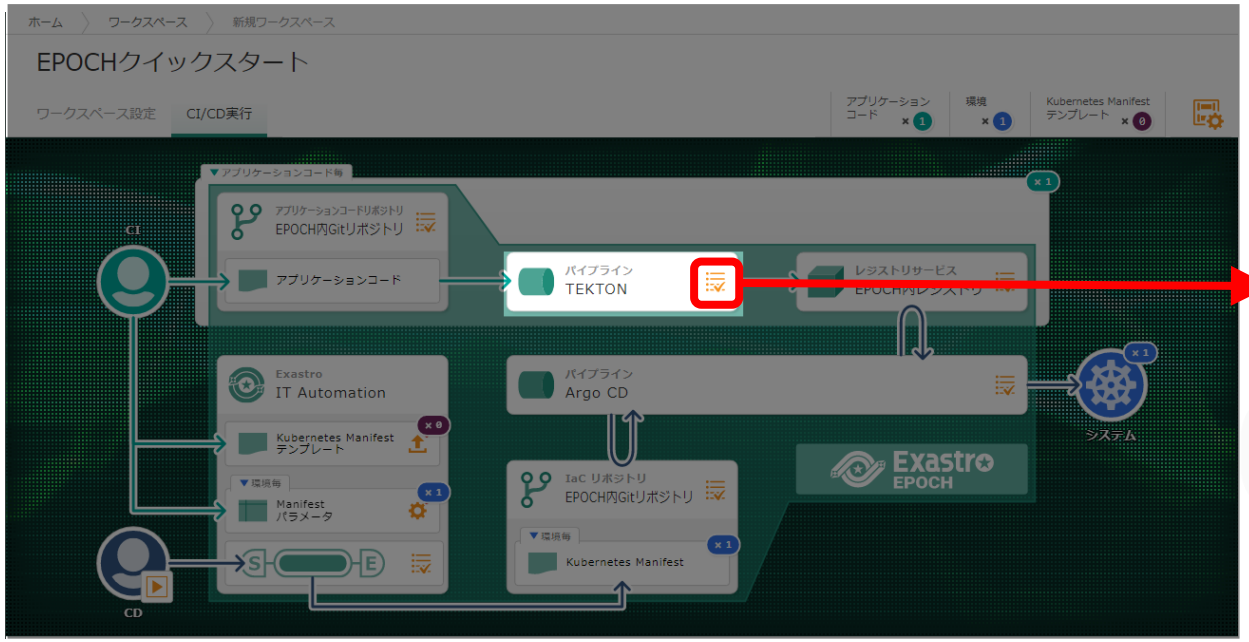


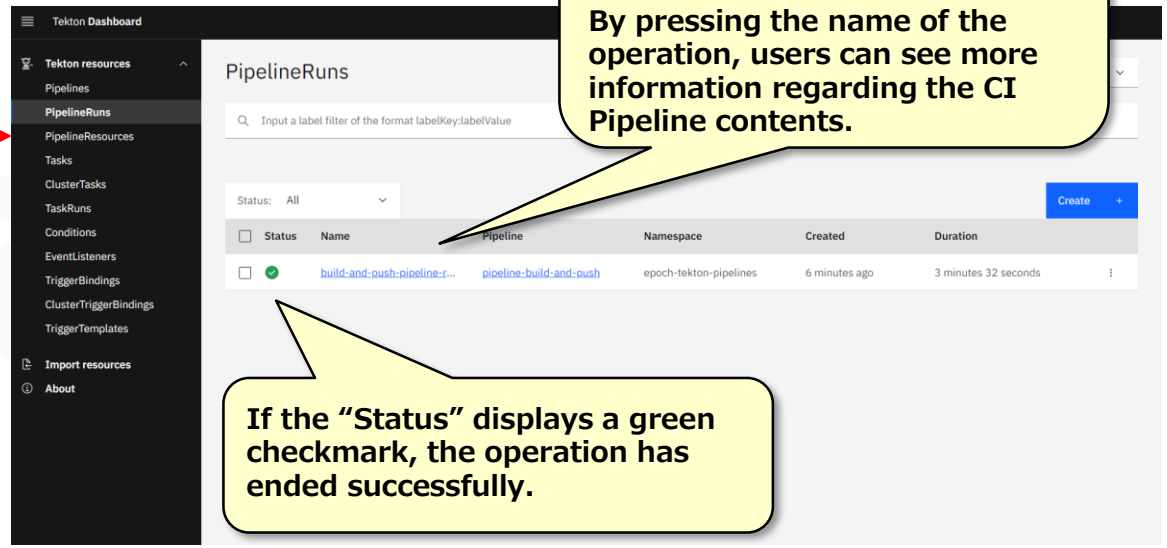
Running the Git push command will start the CI Pipeline.
Users can press this icon in order to see the results.

4.6 CI/CD Workflow No.2 (5/21)

Check the TEKTON Pipeline

- Check the TEKTON pipeline that the build has ended successfully.





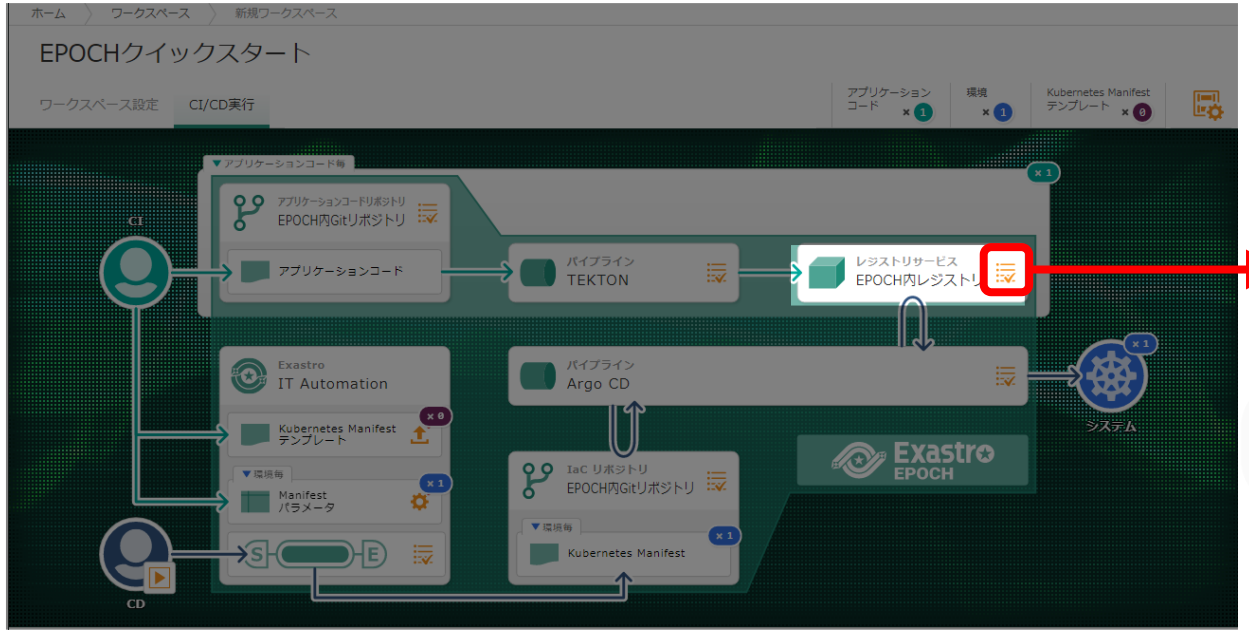
Users can check the CI Pipeline operations from the TEKTON dashboard.
By pressing the name of the operation, users can see more information regarding the CI Pipeline contents.

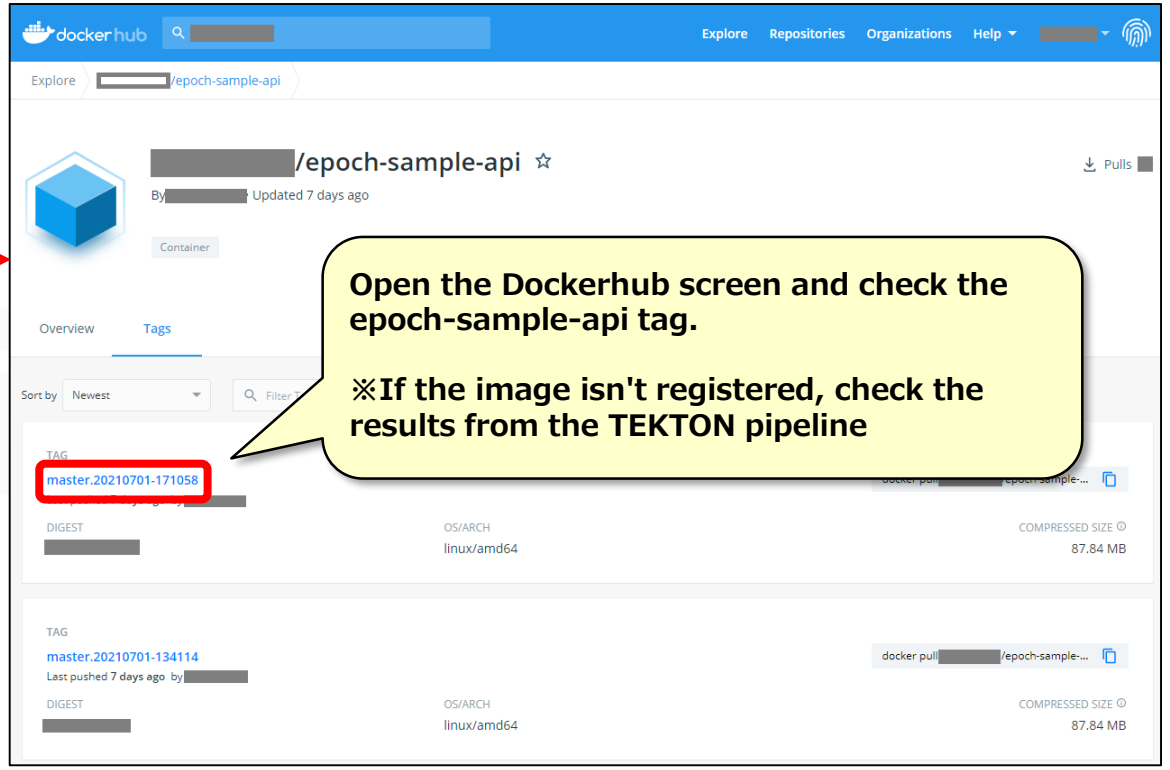
If the “Status” displays a green checkmark, the operation has ended successfully.

4.6 CI/CD Workflow No.2 (6/21)

Check the Container image tag name

- Open the registry service screen and check the tag of the built container image.





Open the Dockerhub screen and check the epoch-sample-api tag.

※If the image isn't registered, check the results from the TEKTON pipeline

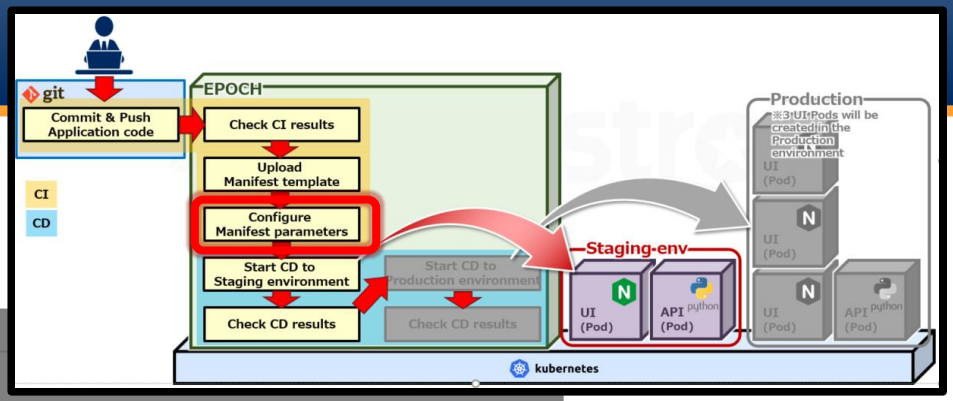
The Image tag will be used in the next step where we input it into the Manifest parameter (image_tag), so make sure to save it.

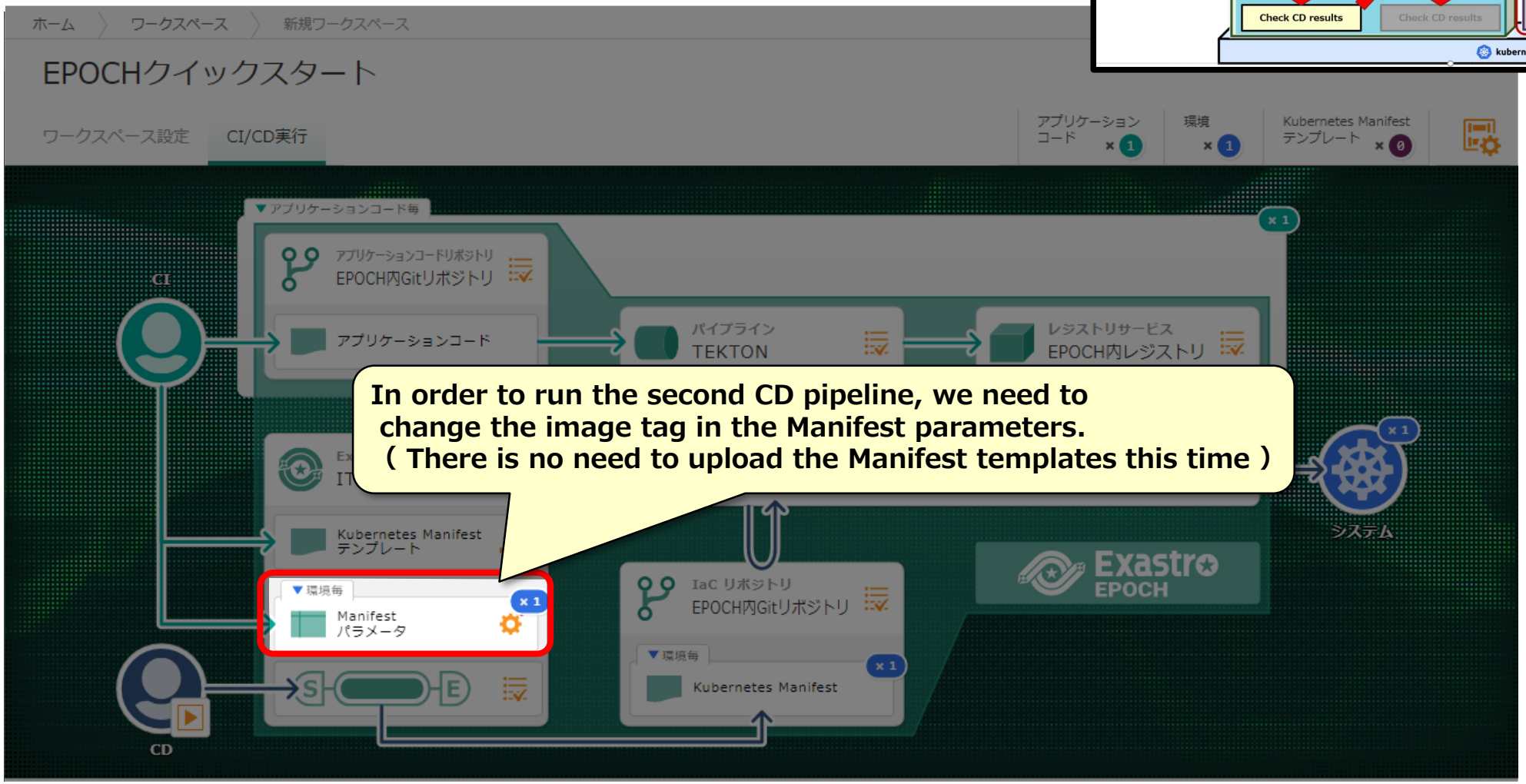
※We plan to make it possible for users to select the image_tag created in the Pipeline.

4.6 CI/CD Workflow No.2 (7/21)

Upload Manifest template

- Upload the Manifest template we downloaded earlier

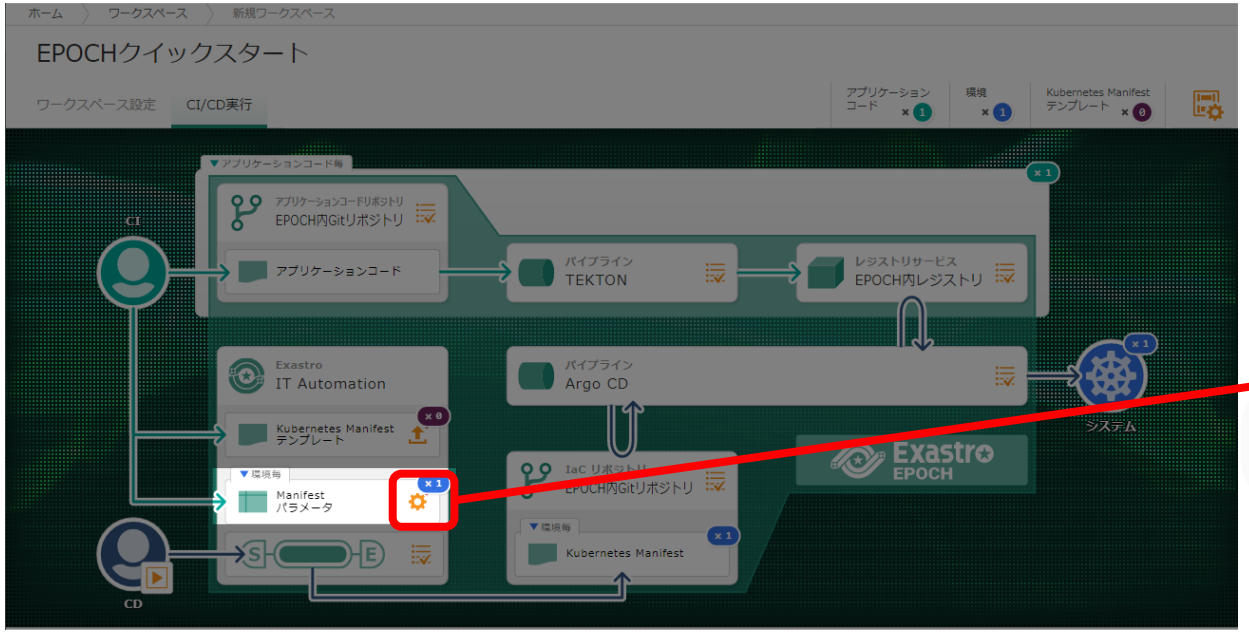


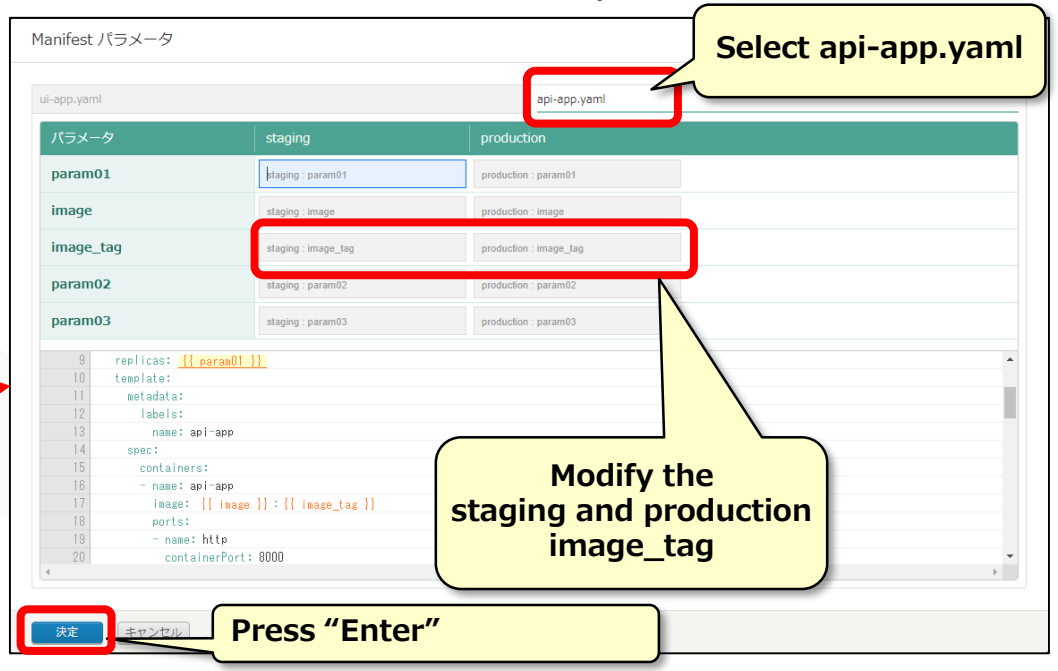


4.6 CI/CD Workflow No.2 (8/21)

Modify Manifest parameter(image_tag)

- Change the image tag name for both the staging and the production environment Manifest parameters.

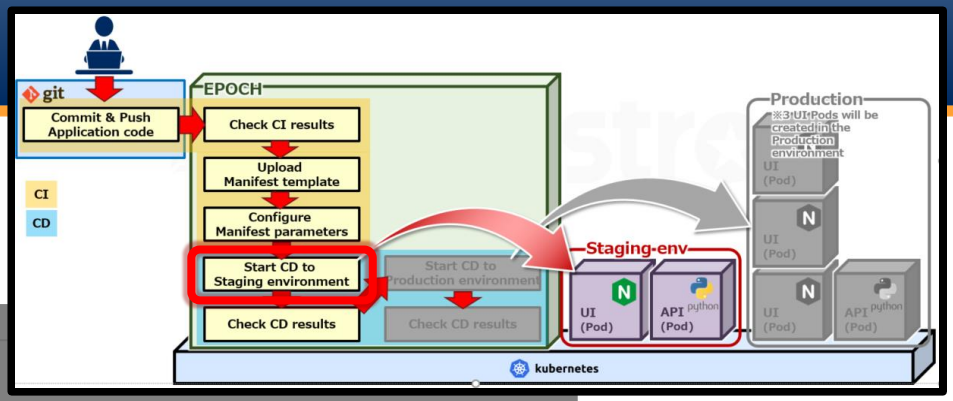




api-app.yaml

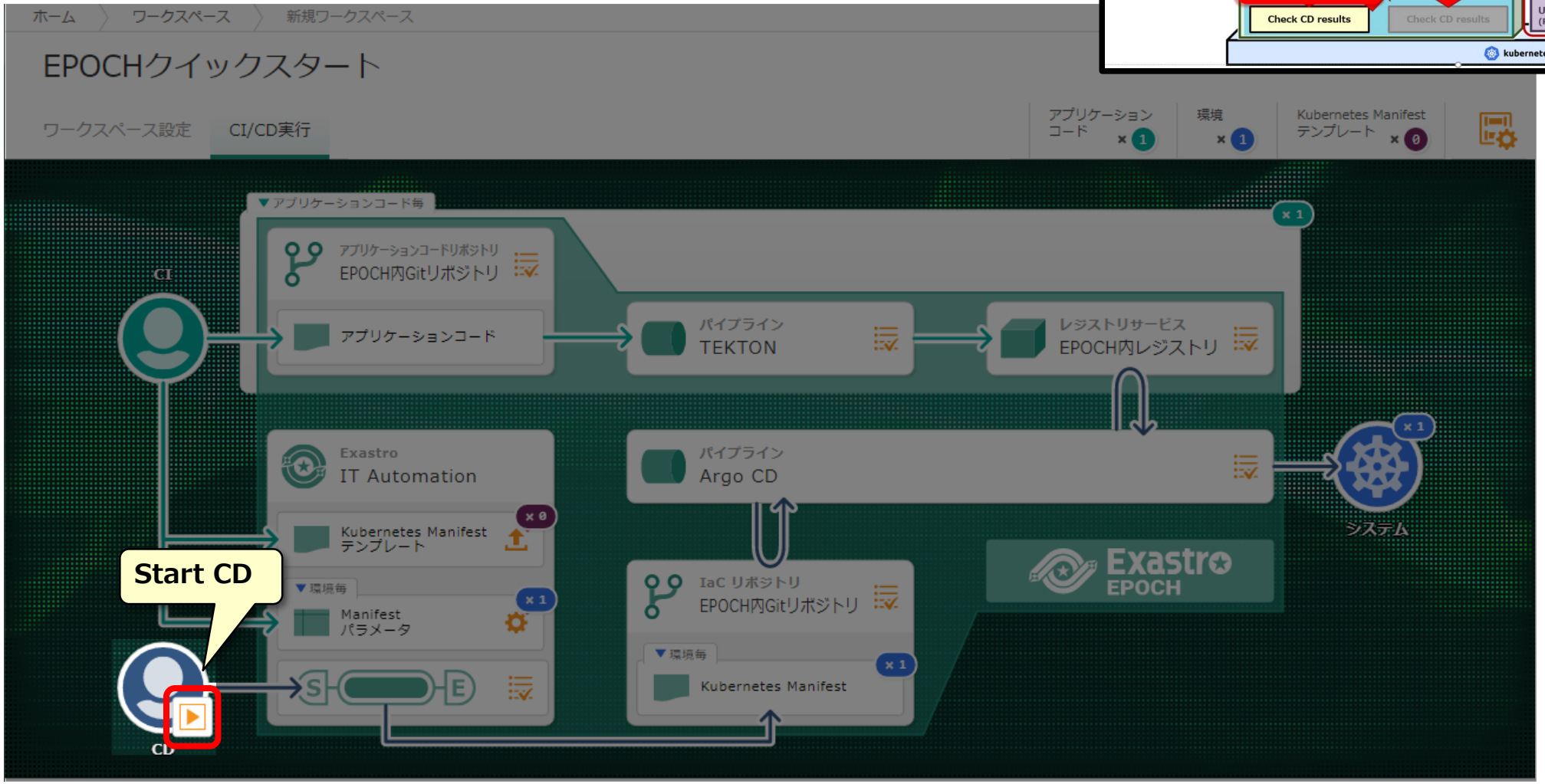
Item	Input contents(staging)	Input contents(production)	Description
{{ image_tag }}	[The most recent image tag checked in the registry service]	[The most recent image tag checked in the registry service]	Container image tag

4.6 CI/CD Workflow No.2 (9/21)



Deploy to Staging environment

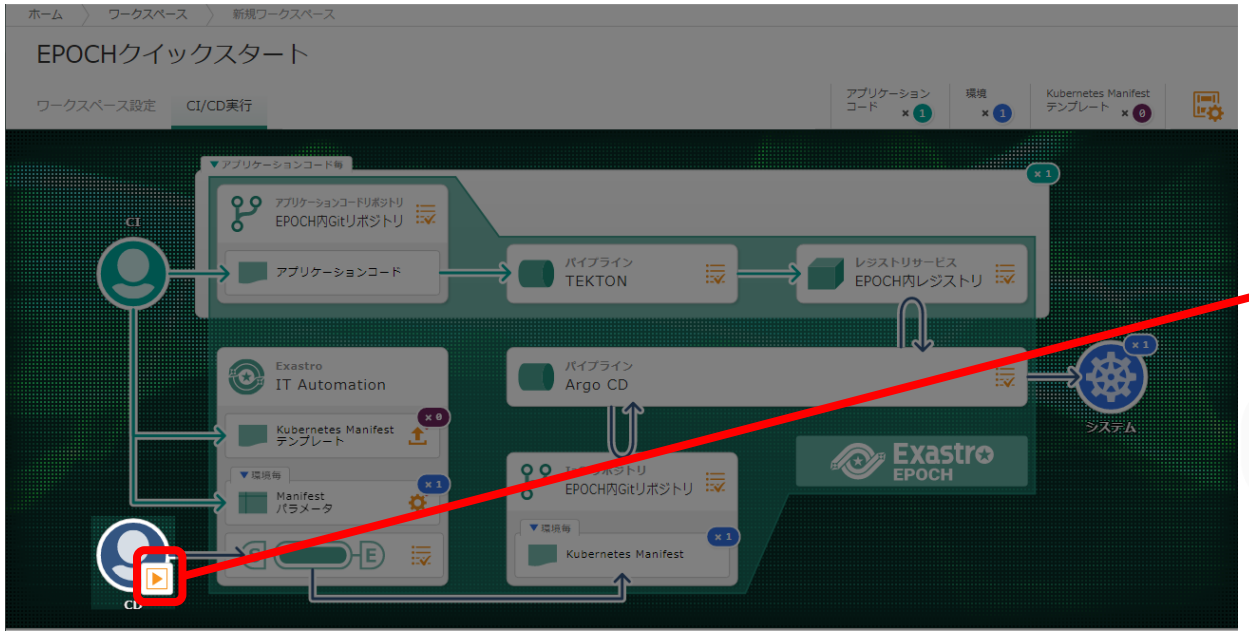
- Start CD and deploy to the Staging environment.



4.6 CI/CD Workflow No.2 (10/21)

Start CD for the Staging environment

- Select your desired environment and start deploying.

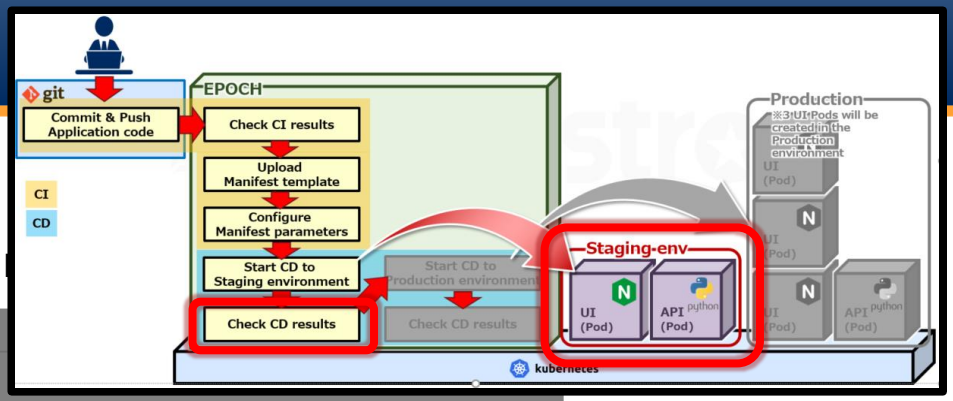


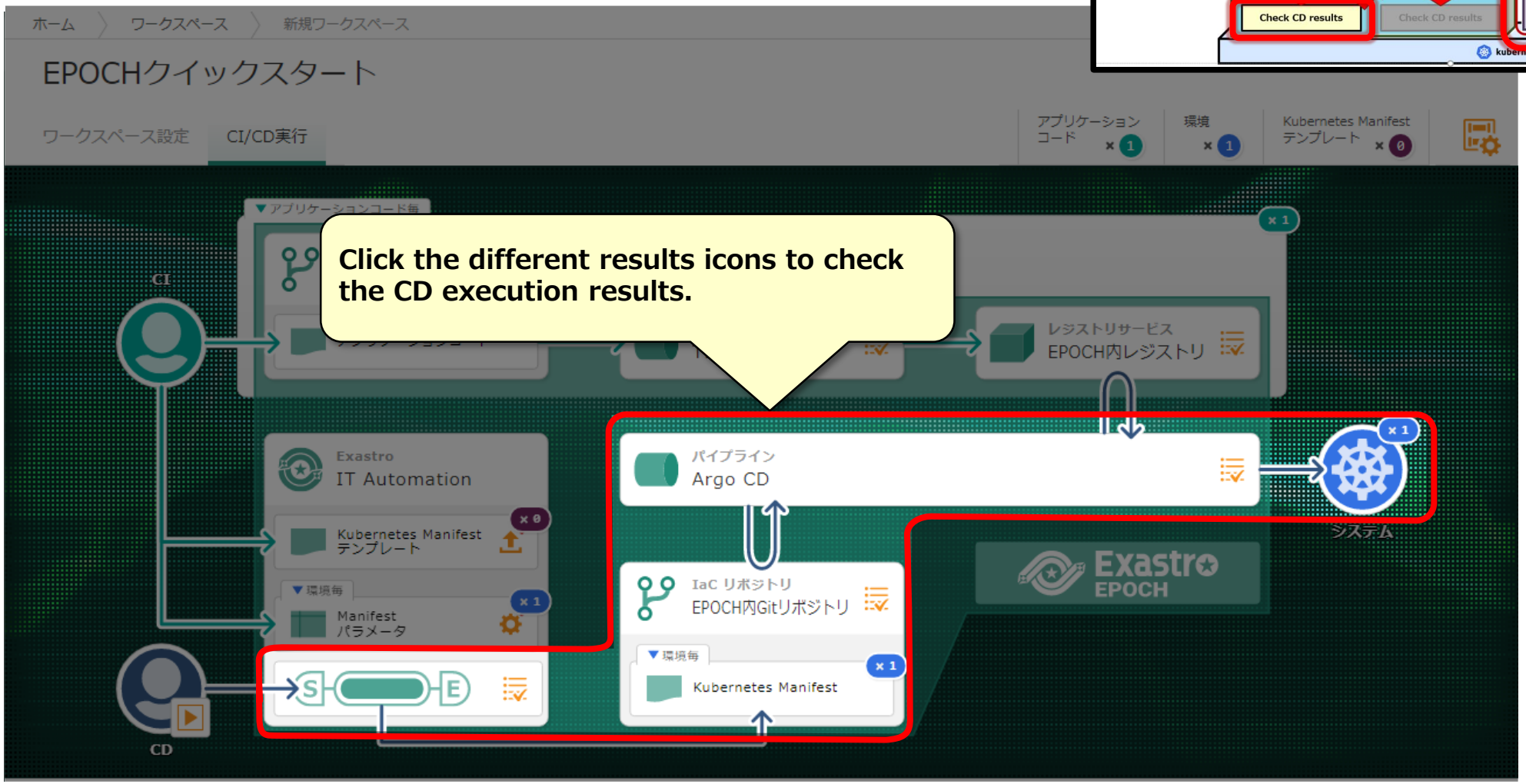
The CD process for the Staging environment has started.
Let us check that it is working as it should.

4.6 CI/CD Workflow No.2 (11/21)

Check the CD results

- We can check the CD execution results from Exastro IT-Automation a

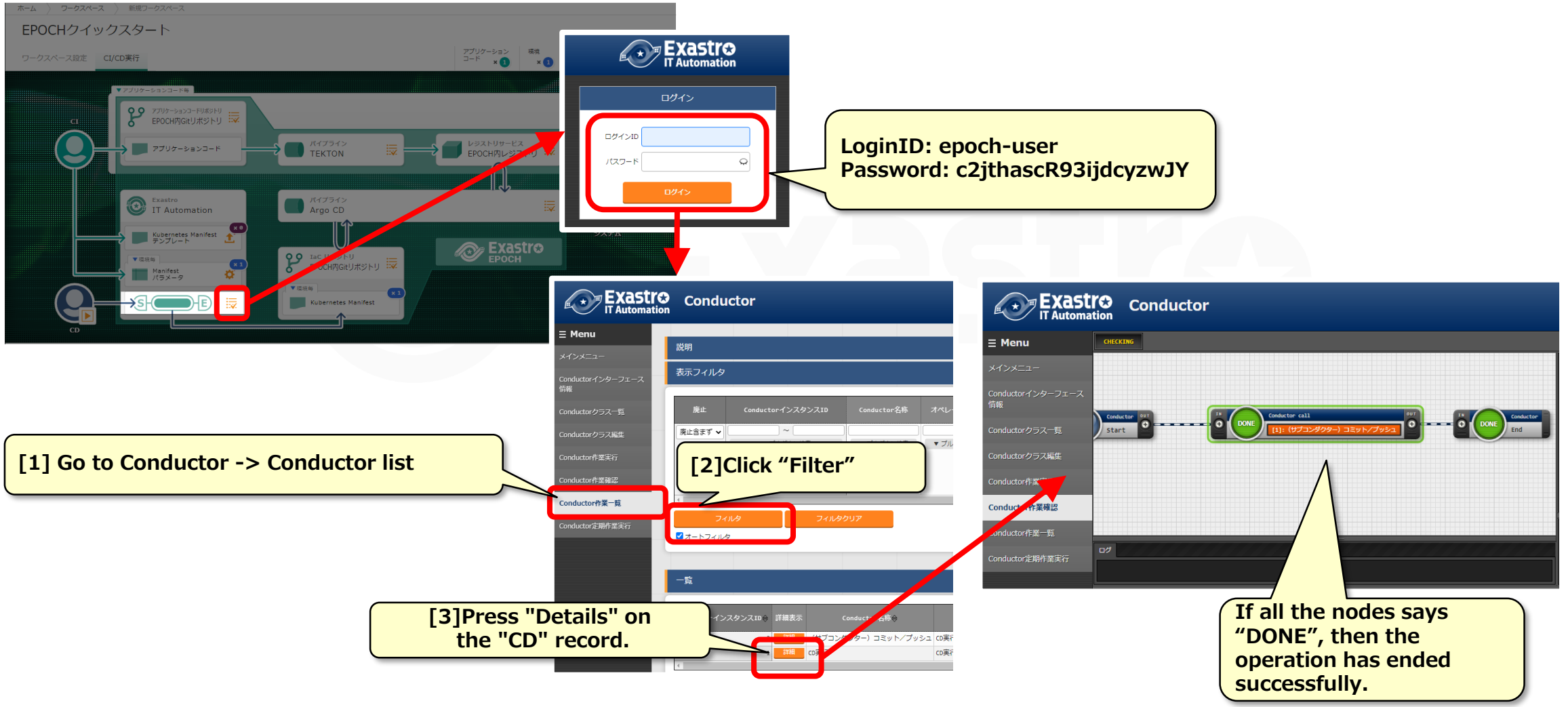




4.6 CI/CD Workflow No.2 (12/21)

Check that the Manifest file has been created (Staging environment)

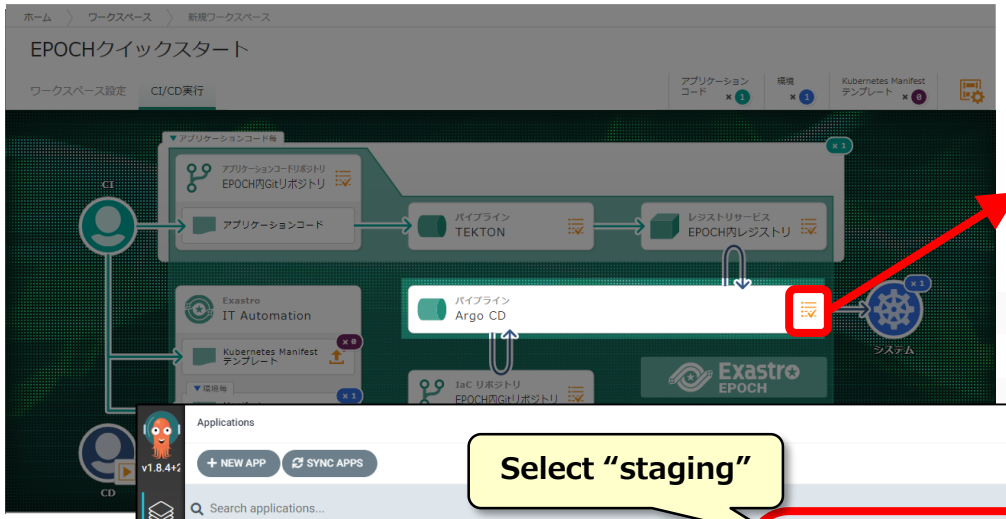
- Open Exastro IT-Automation and check that the Manifest file has been registered to the IaC repository.

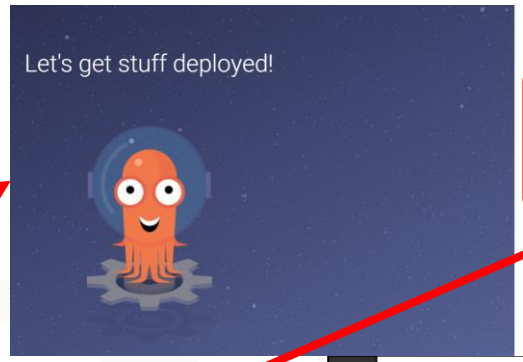


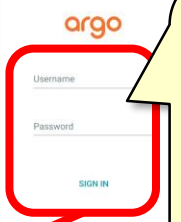
4.6 CI/CD Workflow No.2 (13/21)

Check the results of the ArgoCD Pipeline (Staging environment)

- Check that the Manifest shows up on Kubernetes

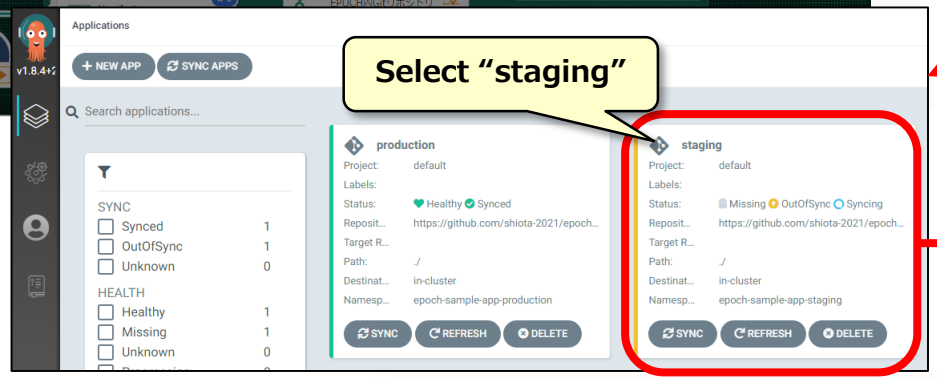




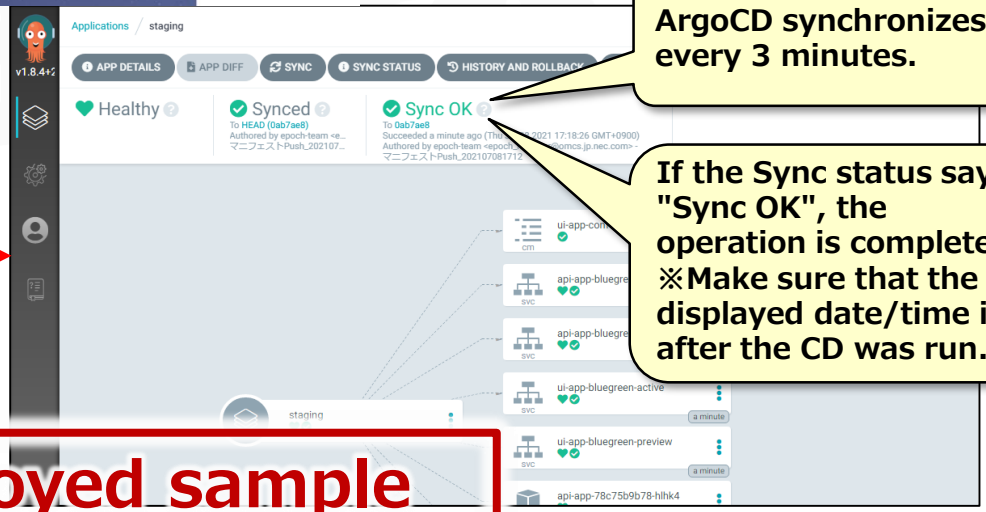


Username: admin
Password: iYSCKzx2wvxJnn4dCNwN

Your browser might say that the connection is not secure. If that is the case, open up the “more information” tab and press “Access”.



Select “staging”



ArgoCD synchronizes every 3 minutes.

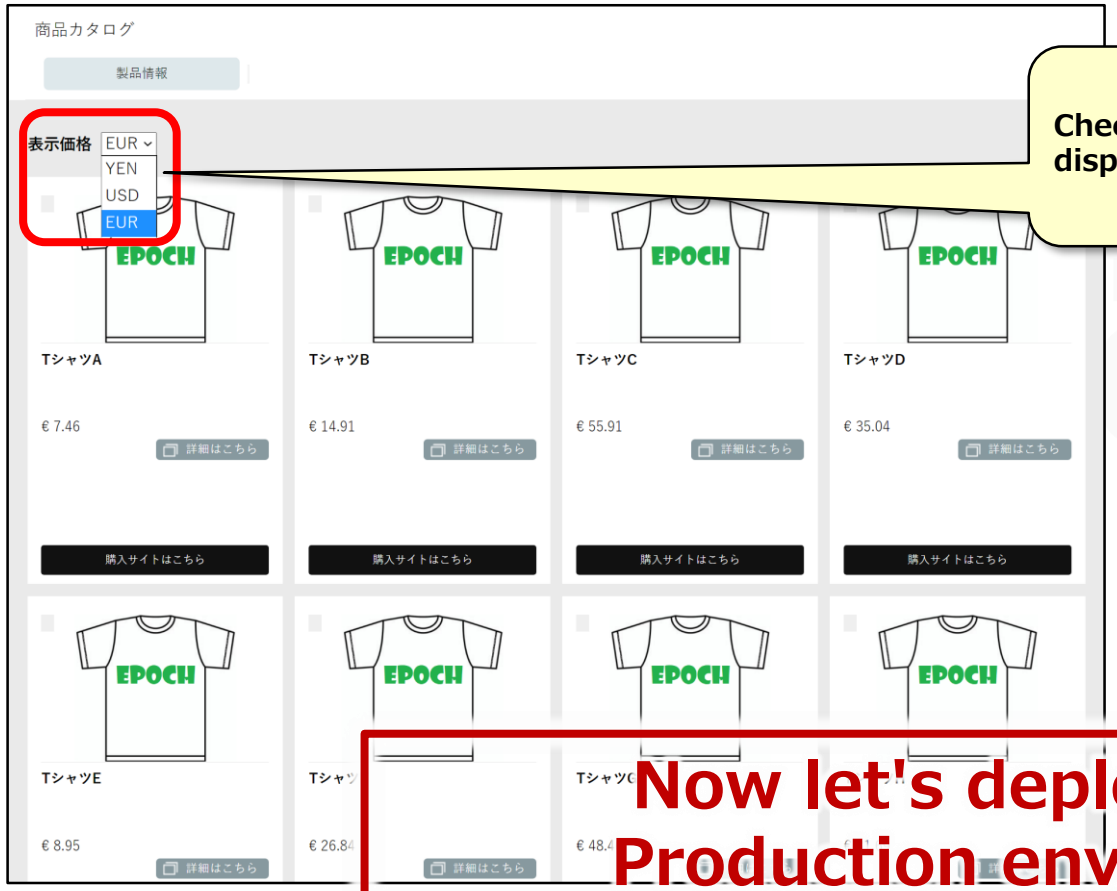
If the Sync status says "Sync OK", the operation is complete.
※Make sure that the displayed date/time is after the CD was run.

Let us check the deployed sample application

4.6 CI/CD Workflow No.2 (14/21)

Check the Staging environment sample code.

- Access the sample application from the URL below
  http://[Kubernetes master node IP address/Host name]:31001/front-end.html



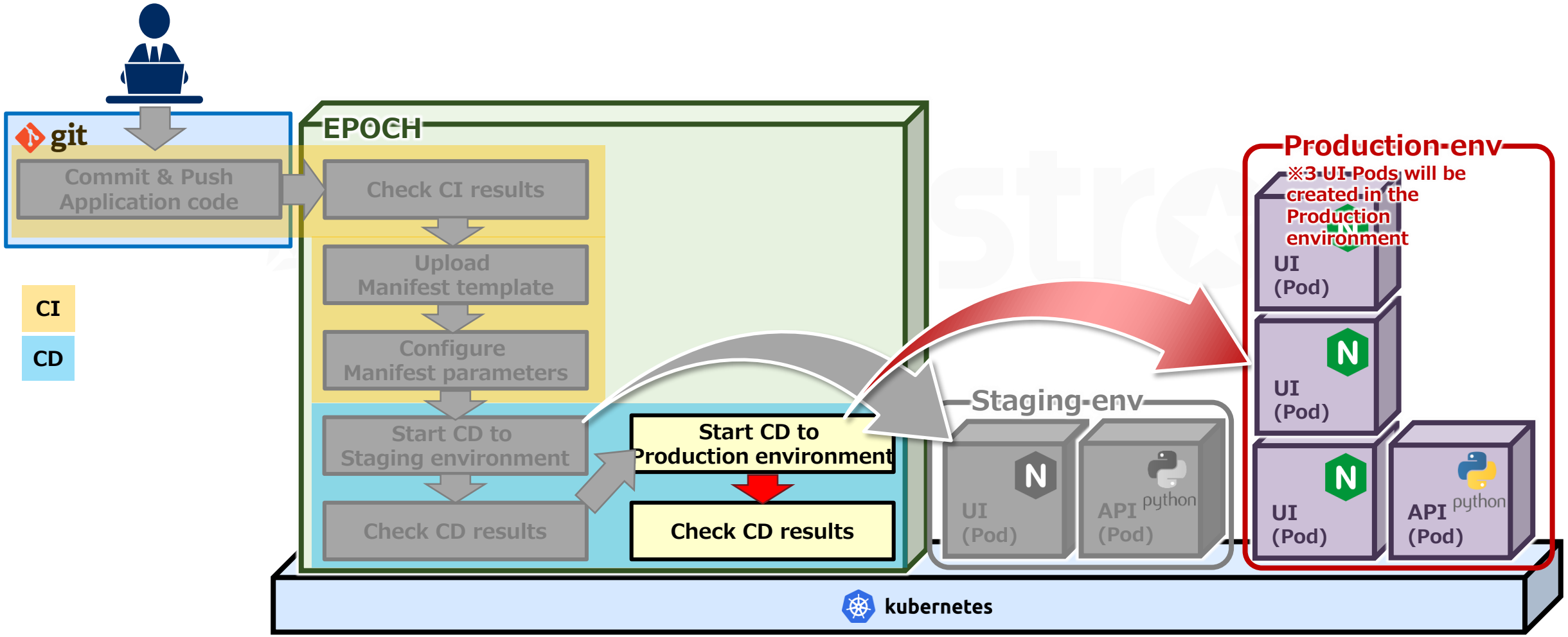
Check that "YEN", "USD", and the newly added "EUR" is displayed and selectable from the Currency pulldown selection.

Now let's deploy to the Production environment

4.6 CI/CD Workflow No.2 (15/21)

Deploy to Production environment

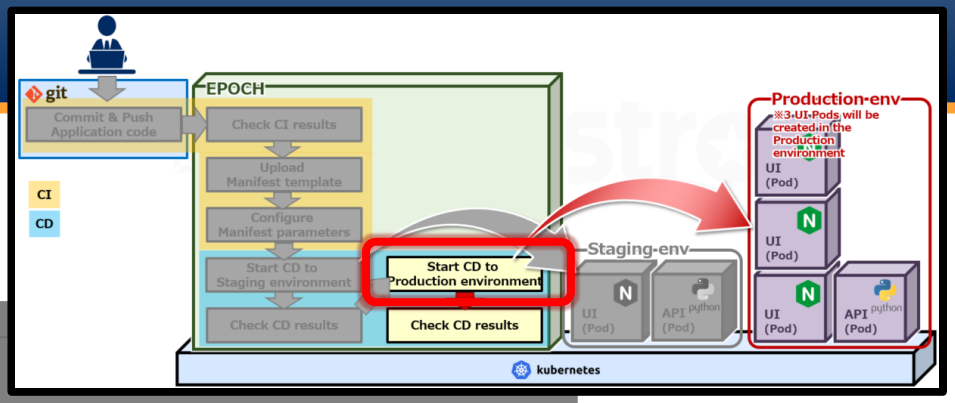
Now that we have deployed to the Staging environment and checked that everything functions as it should, we can now deploy to the Production environment.

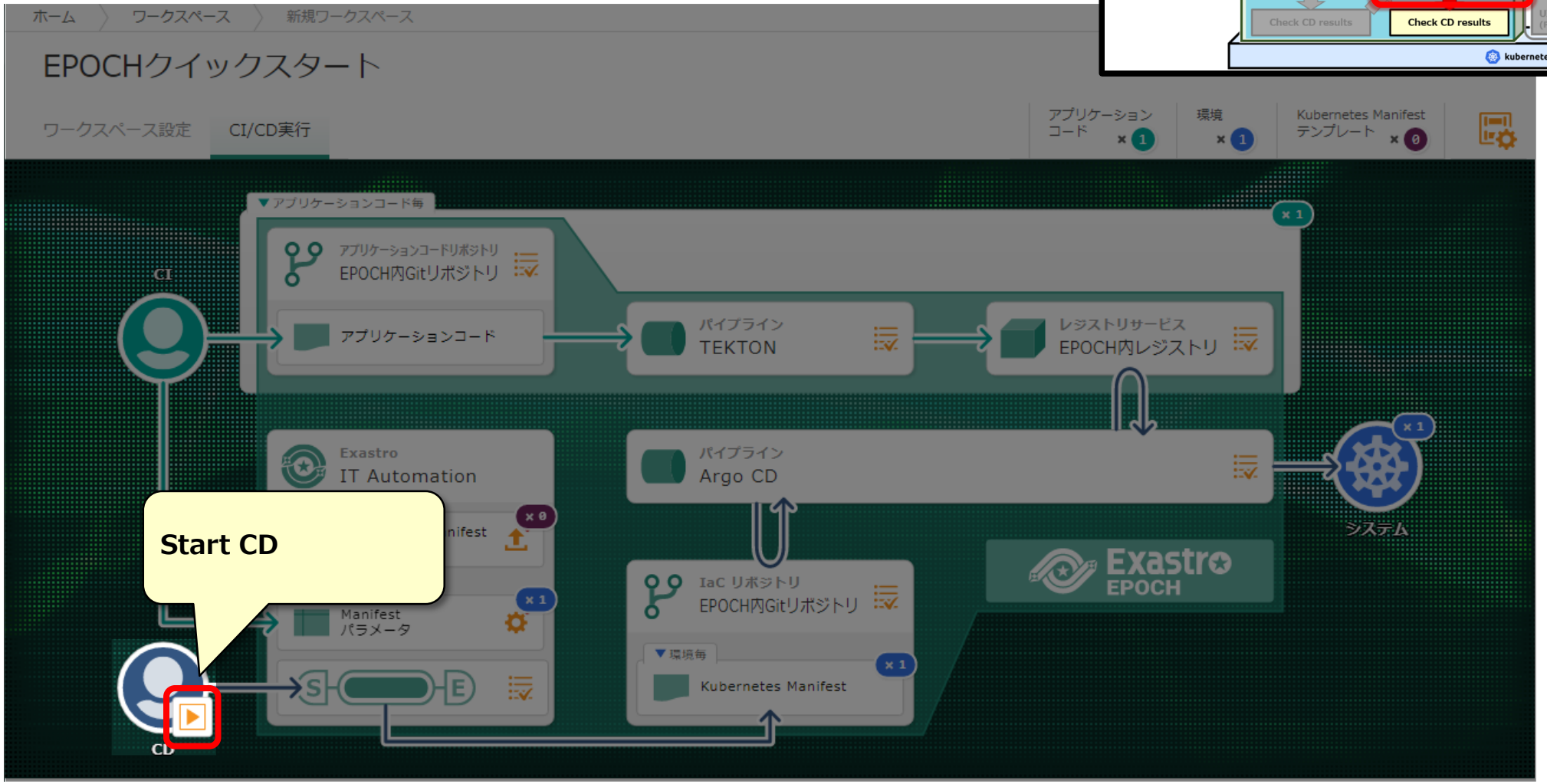


4.6 CI/CD Workflow No.2 (16/21)

Deploy to the Production environment

- Start the CD and deploy to the Production environment.

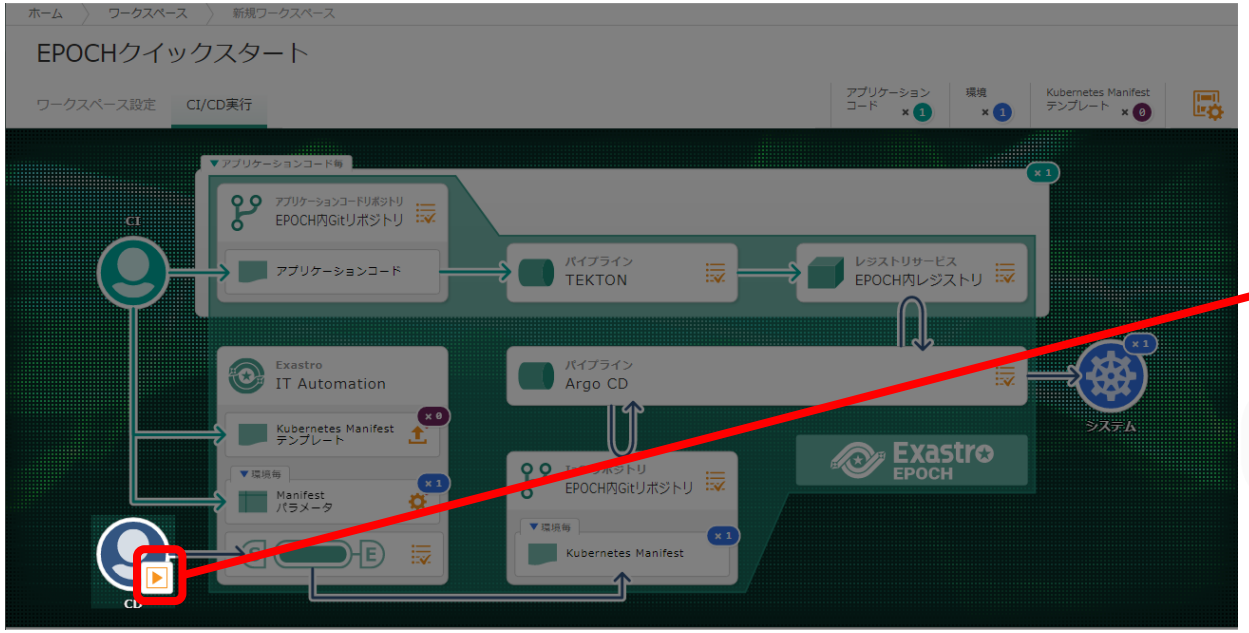




4.6 CI/CD Workflow No.2 (17/21)

Start CD for the Production environment

- Select the environment you want to deploy to.

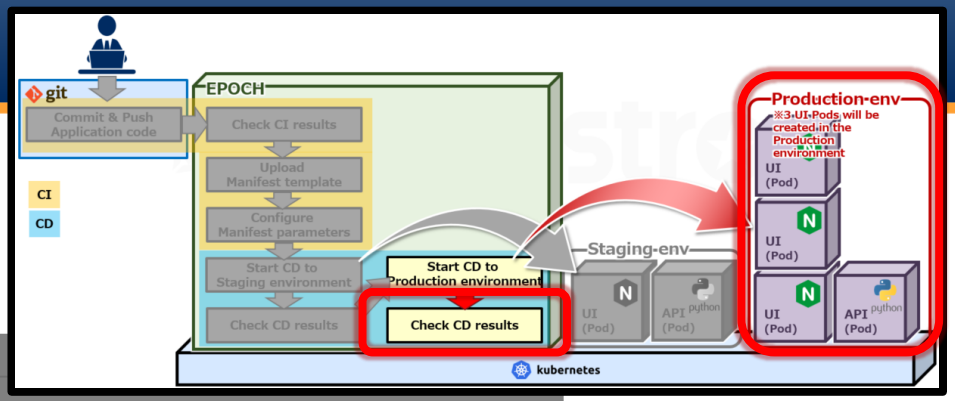


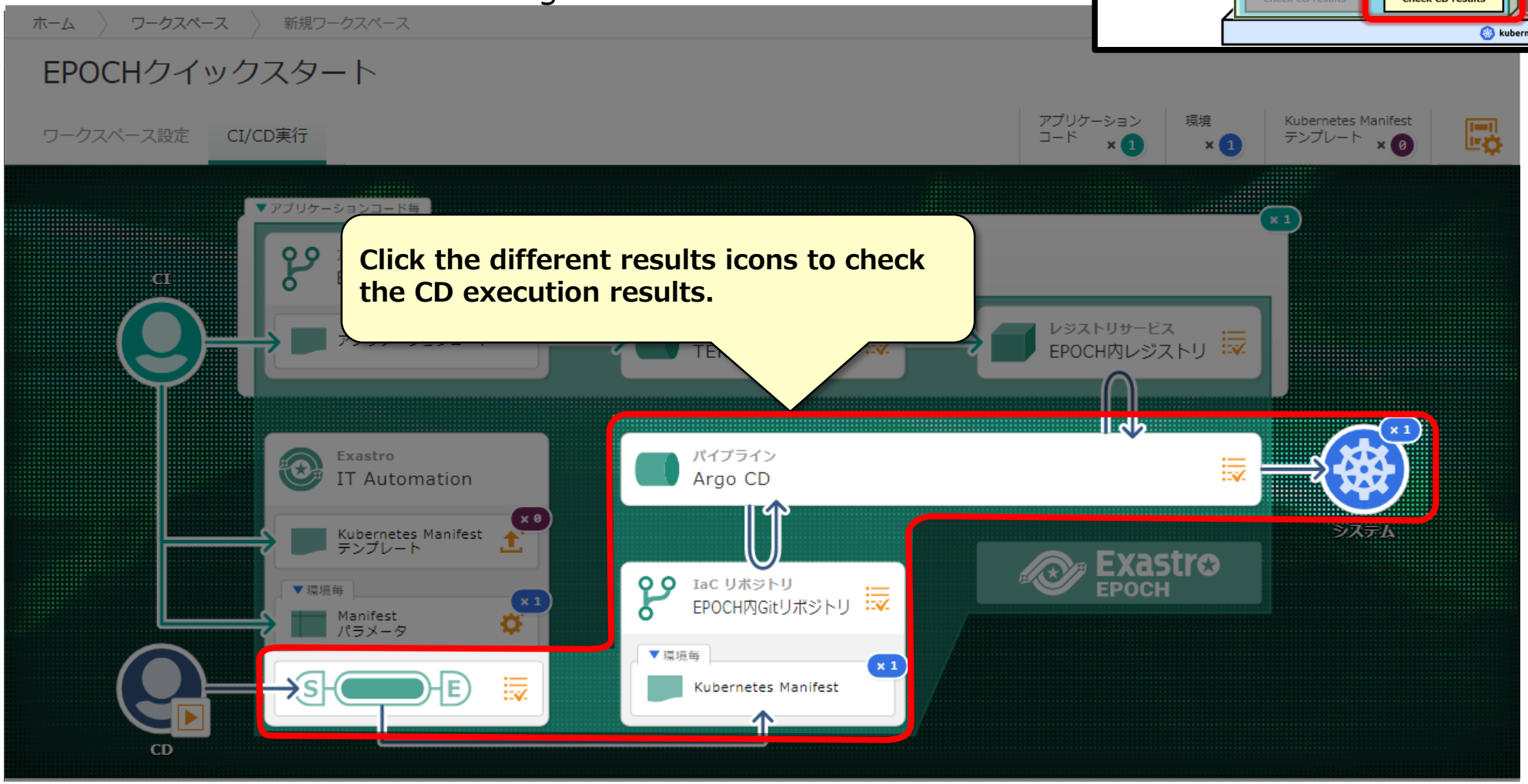
The CD process for the Production environment has started.
Let us check that it is working as it should.

4.6 CI/CD Workflow No.2 (18/21)

Check the CD results (Production)

- We can check the CD execution results from Exastro IT-Automation and ArgoCD.

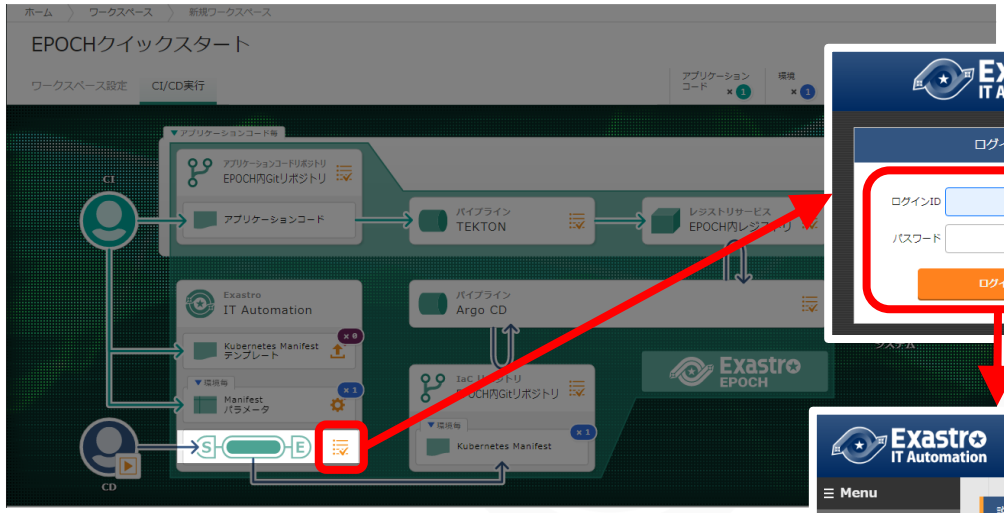




4.6 CI/CD Workflow No.2 (19/21)

Check that the Manifest file has been created (Production environment)

- Open Exastro IT-Automation and check that the Manifest file has been registered to the IaC repository.

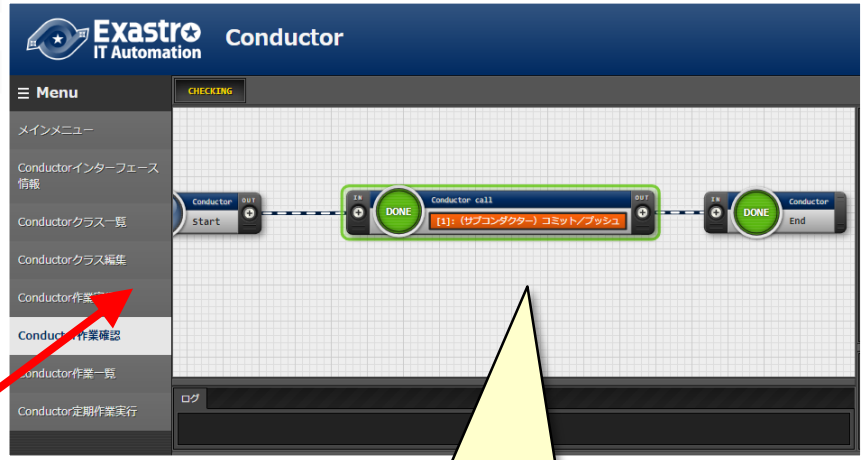


LoginID: epoch-user
Password: c2jthascR93ijdcyzwJY

[1] Go to Conductor -> Conductor list

[2]Click “Filter”

[3]Press "Details" on the "CD" record.

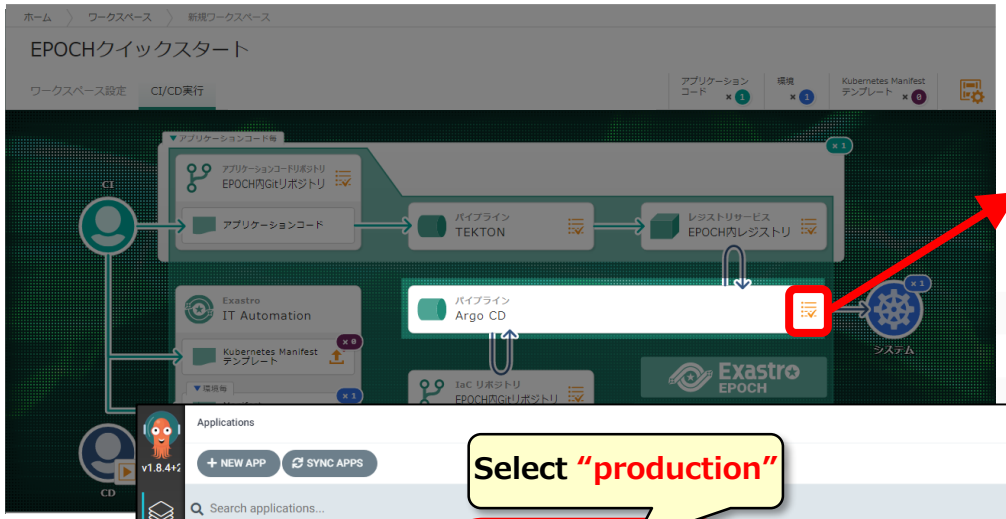


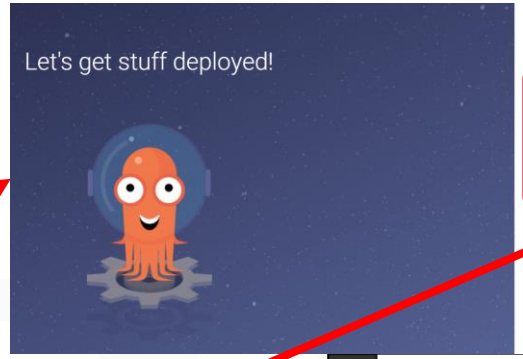
If all the nodes says “DONE”, then the operation has ended successfully.

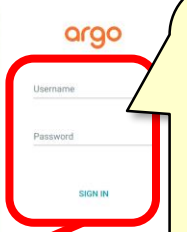
4.6 CI/CD Workflow No.2 (20/21)

Check the results of the ArgoCD Pipeline (Production environment)

- Check that the Manifest shows up on Kubernetes

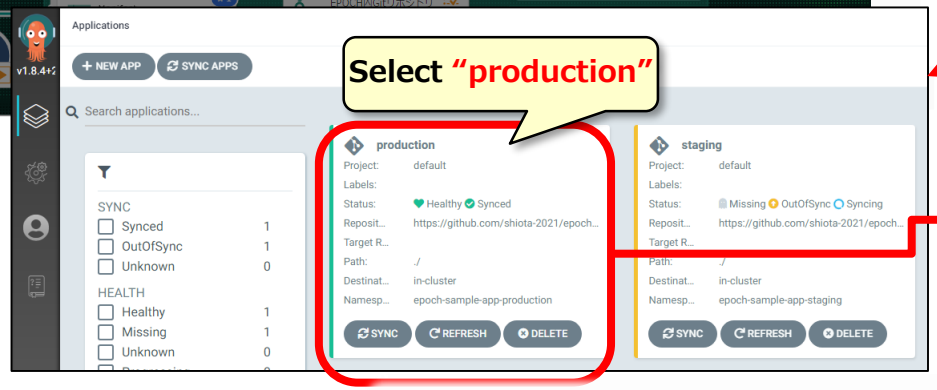




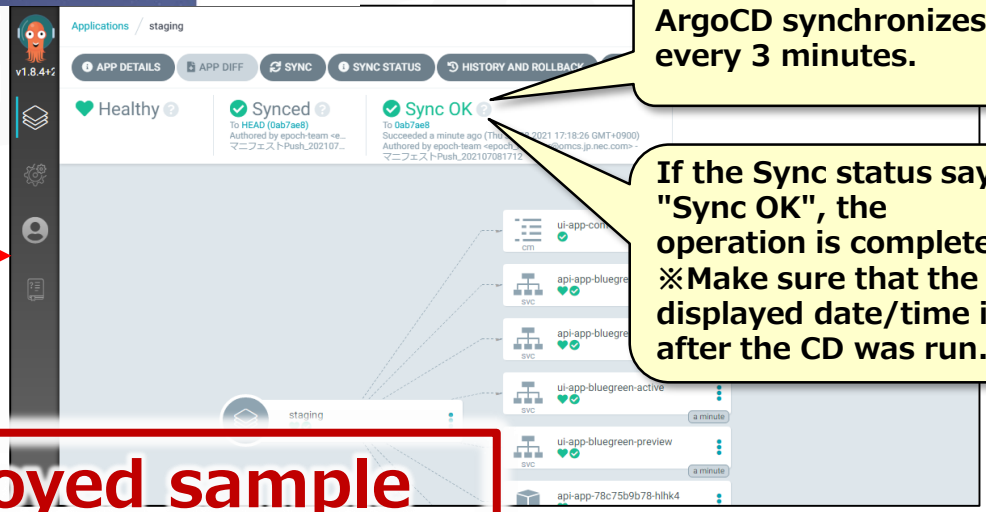


Username: admin
Password: iYSCKzx2wvxJnn4dCNwN

Your browser might say that the connection is not secure. If that is the case, open up the “more information” tab and press “Access”.



Select “production”



ArgoCD synchronizes every 3 minutes.

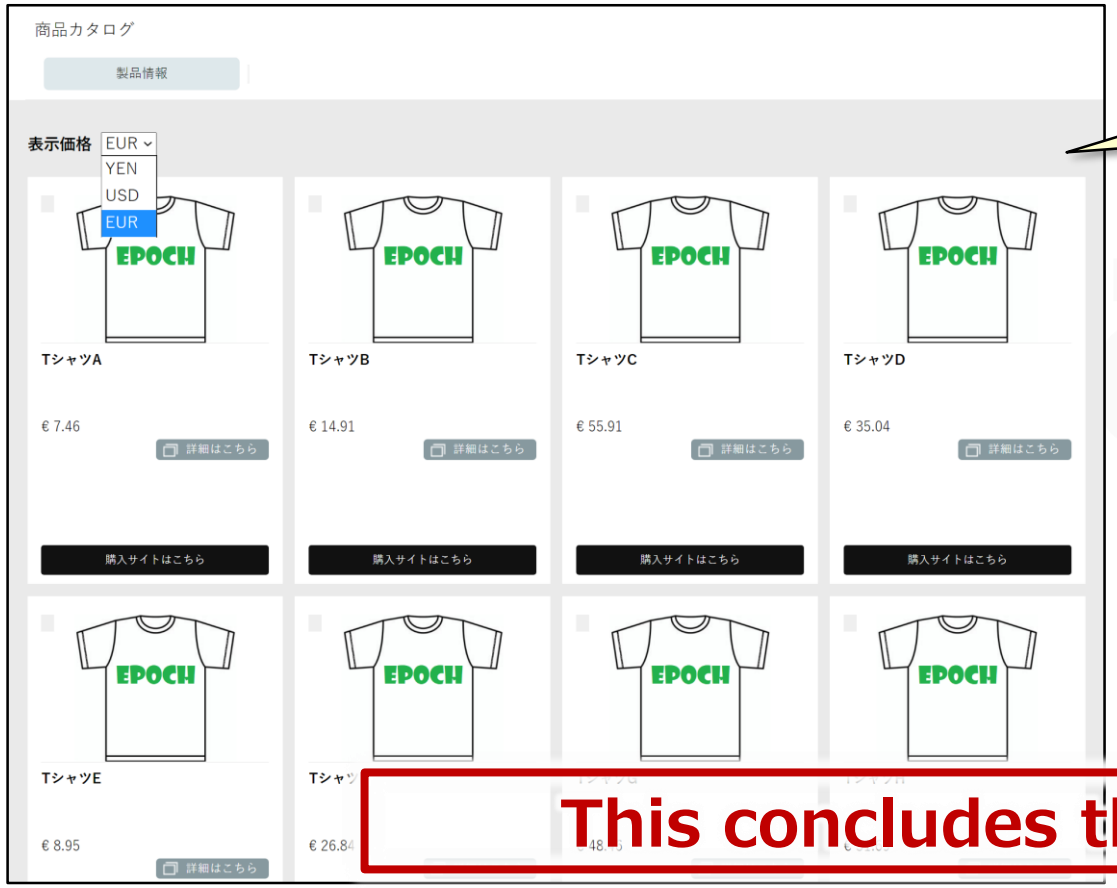
If the Sync status says "Sync OK", the operation is complete.
※Make sure that the displayed date/time is after the CD was run.

Let us check the deployed sample application

4.6 CI/CD Workflow No.2 (21/21)

Check the Production environment sample code.

- Access the sample application from the URL below
  http://[Kubernetes master node IP address/Host name]:31003/front-end.html



Check that the screen displayed is the same as the one in the Staging environment.

This concludes the tutorial.

5. Appendix

The next section contains supplementary information for the tutorial.

5.1 Restrictions

The following are the restrictions that are present in the current Exastro EPOCH version and may be subject to change.

- Restrictions（Planned fix in the future）
  - Only 1 application code repository is supported at the moment.
  - You can only have one collective Git account for the application codes.
  - An option for selecting Git service is planned to be added in future releases. Only Git repository URLs are supported in this version.
  - Support for build branches is planned to be added in future releases. Only pushed contents is supported in this version.
  - Support for Static response analysis is planned to be added in future releases. 現在はSonarQubeを選択した場合に動作しません。
  - The only registry service supported is the internal registry service.
  - Items other than the image output destination is planned to be added in future releases.
  - Support for Authentication tokens and Base64 encoded certificates is planned to be added in future releases.
  - The variables that can be specified in the template are fixed. Please see "Column" for more information.

- Points to note
  - Installing EPOCH will also install TEKTON.
  - The ”{{ Variable name }}” variables are specified contents.

Column : Manifest templates and their variable names

When a Manifest template is uploaded to EPOCH, the system will analyze pre-defined text and make them into modifiable parameters.

```
9   replicas: {{ param01 }}
10  template:
11    metadata:
12      labels:
13        name: api-app
14    spec:
15      containers:
16      - name: api-app
17        image: {{ image }} : {{ image_tag }}
18        ports:
19        - name: http
20          containerPort: 8000
```

Strings that are described in this format :
{{ Variable name }}
will be recognized as user-inputable variables.
The variables that can be used in the current version of EPOCH are the following:

Variable name	Description
{{ image }}	Container image
{{ image_tag }}	Container image tag
{{ param01 }}	Fixed variables that users can use freely.
〜	
{{ param20 }}	Fixed variables that users can use freely. (Can use max 20)

※Support for user-specified variable names is planned to be added in future releases.


Exastro