



Exastro

EPOCH Quick Start

第0.1.0版

Exastro developer

目次

1. はじめに

1.1 QuickStartについて

1.2 QuickStartを実施するPC環境について

2. インストール

2.1 EPOCHのインストール

2.2 リポジトリ準備

2.3 Manifestテンプレートファイル準備

3. ワークスペース作成

3.1 ワークスペース

3.2 CI/CDについて

3.3 EPOCHのCI/CD

3.4 EPOCH起動

3.5 ワークスペース作成

4. チュートリアル

4.1 サンプルアプリの構成

4.2 チュートリアルの流れ(CI/CDワークフロー)

4.3 Manifestテンプレートファイルについて

4.4 1回目のCI/CDワークフロー手順

4.5 2回目のCI/CDワークフロー手順

5. 付録

5.1 注意事項・制限事項

1. はじめに

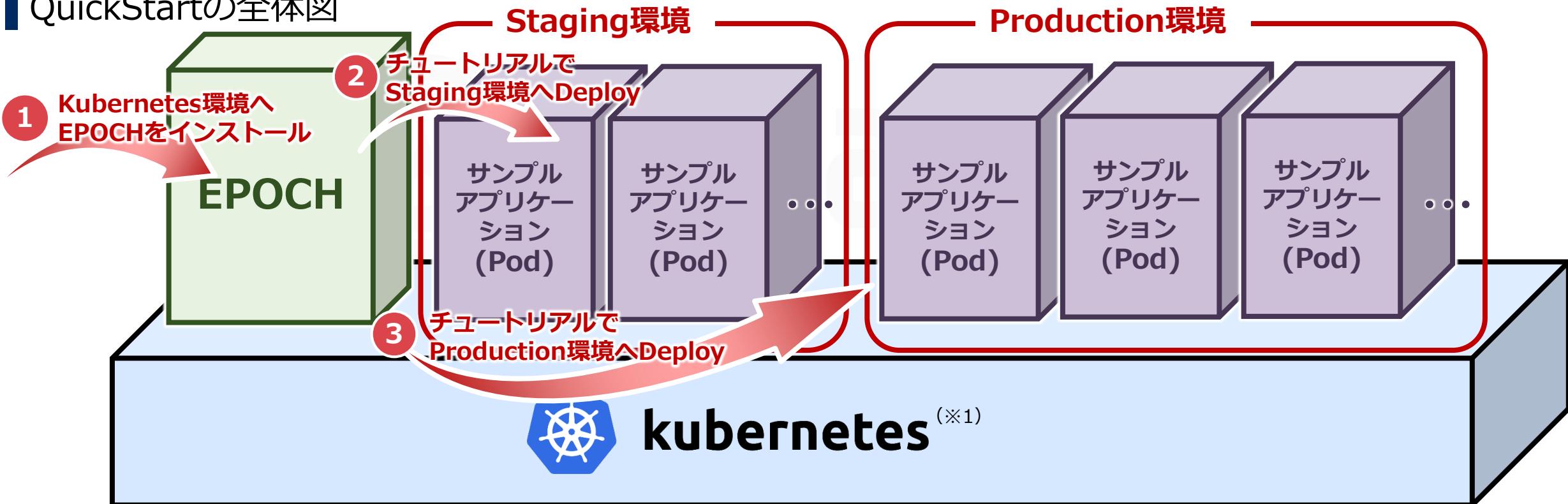


1.1 QuickStartについて

はじめに

本書は、Exastro EPOCH(以降、EPOCHと表記する)の導入方法ならびに簡単な使い方をチュートリアルを用いて説明します。

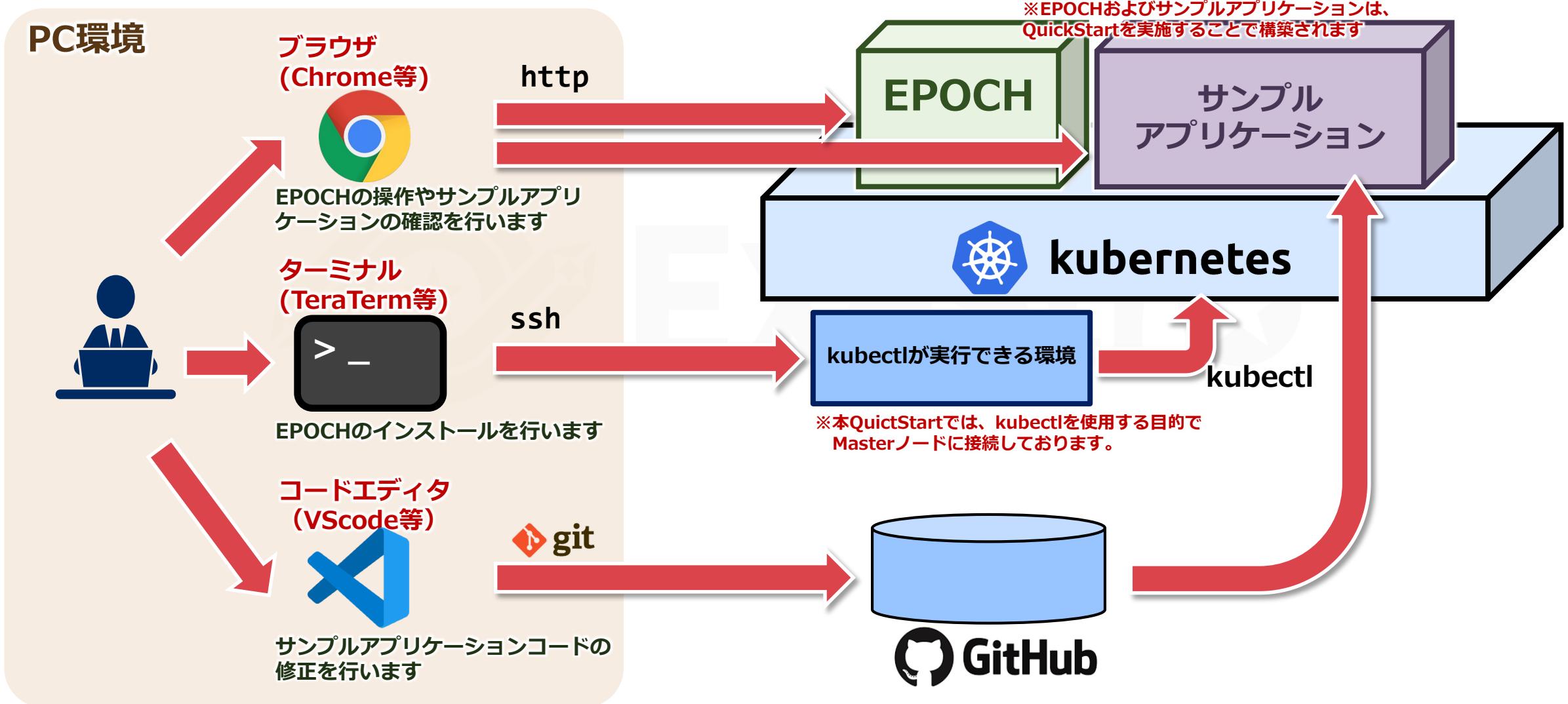
QuickStartの全体図



(※1)本クイックスタートでは手順を簡素化するため1つのKubernetesクラスタ上で構成します

1.2 QuickStartを実施するPC環境について

QuickStartの手順を実施するにあたってのPCのソフトウェアは以下の通りです。





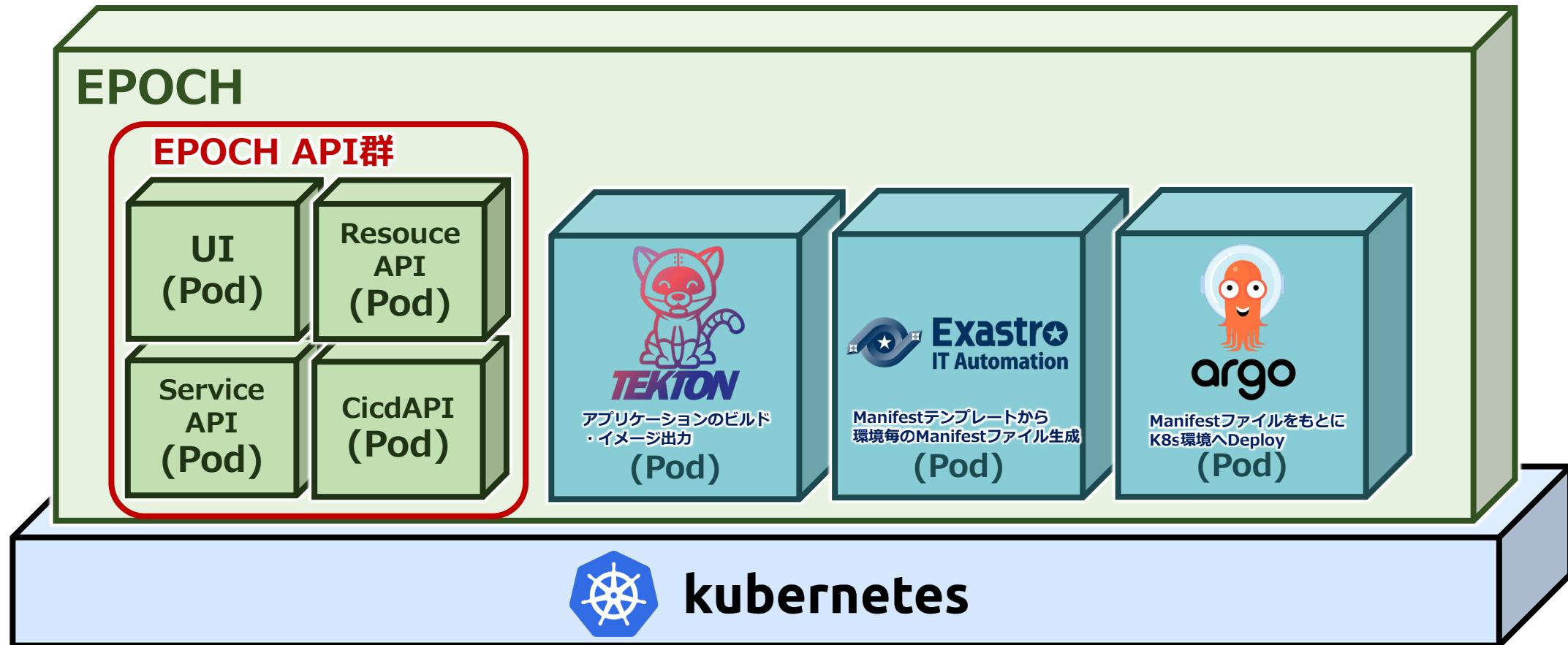
2. インストール

EPOCHをインストールして、CI/CDの環境を準備をしましょう。

2.1 EPOCHのインストール(1/5)

EPOCH全体図

EPOCHをインストールおよびワークスペースを作成した後の構成は、以下の図のようになります。



2.1 EPOCHのインストール(2/5)

■ 前提条件

- 環境
 - Kubernetes環境が構築されていること
 - 使用するServiceAccountにcluster-adminロールが付与されていること
 - Kubernetes環境から外部インターネットに接続できること
 - PC環境から外部インターネットに接続できること
 - ポート番号(30080, 30081, 30901~30907)が使用できること
(ポート番号はepoch-install.yamlに記述されており、変更する際は編集後インストールを実行する必要があります)

- アカウント
 - アプリケーションコードを登録するGitHubのアカウントが準備されていること
 - Kubernetes Manifestを登録するGitHubのアカウントが準備されていること
 - コンテナイメージを登録するDockerHubのアカウントが準備されていること

2.1 EPOCHのインストール(3/5)

EPOCHインストール

- ターミナルでkubectlが実行できる環境にSSHログインし、以下のコマンドを実行してEPOCHをインストールします。

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

以下のコマンドでインストールの進行状況を確認できます。

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

コマンド結果に表示されているすべてのコンポーネントのSTATUSが“Running”であることを確認します。

【コマンド結果 イメージ】

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 EPOCHのインストール(4/5)

永続ボリューム設定

パイプライン設定用の永続ボリュームを設定します。

- 以下のコマンドを実行し、マニフェストをGitHubから取得します。

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

- 以下のコマンドを実行し、Workerノードのホスト名を確認します。

```
$ kubectl get node
```

【コマンド結果 イメージ】

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

- epoch-pv.yamlを修正します。（修正箇所はepoch-pv.yamlの最終行）

「# Please specify the host name of the worker node #」の部分を、先ほど確認したWorkerノードのホスト名に置き換え保存します。

【変更イメージ】

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

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

- 以下のコマンドでkubernetes環境へ反映します。

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

2.1 EPOCHのインストール(5/5)

ArgoRolloutインストール

- 以下のコマンドを実行し、ArgoRolloutのインストールします。

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

以上でEPOCHのインストールは完了しました。
次にチュートリアルを実施するための事前準備を実施しましょう！

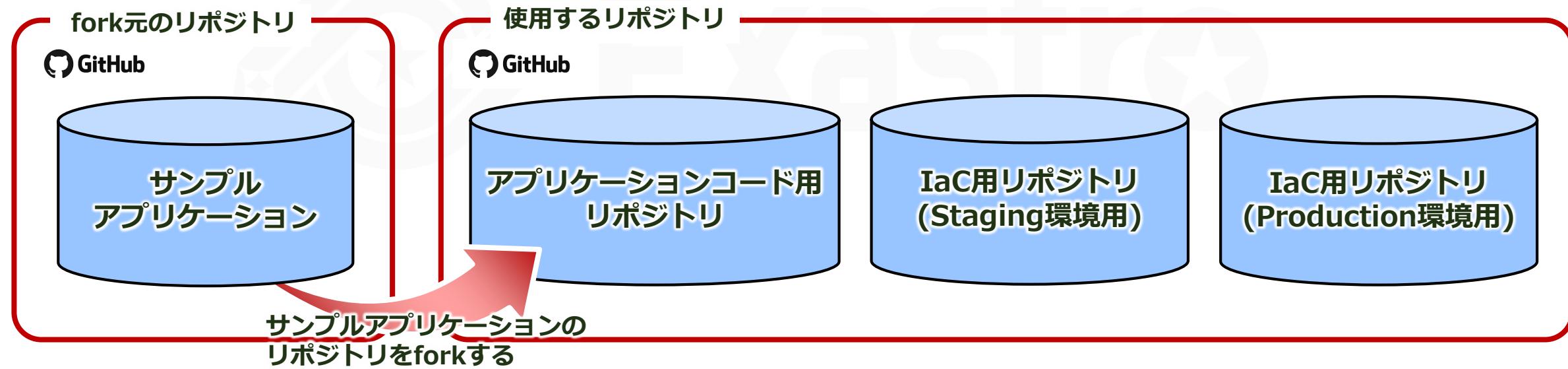
2.2 リポジトリ準備(1/4)

使用するリポジトリについて

- 本クイックスタートで使用するリポジトリは以下の通りです。

- アプリケーションコード用リポジトリ
- IaC用リポジトリ(Staging環境用)
- IaC用リポジトリ(Production環境用)

- イメージ図



2.2 リポジトリ準備(2/4)

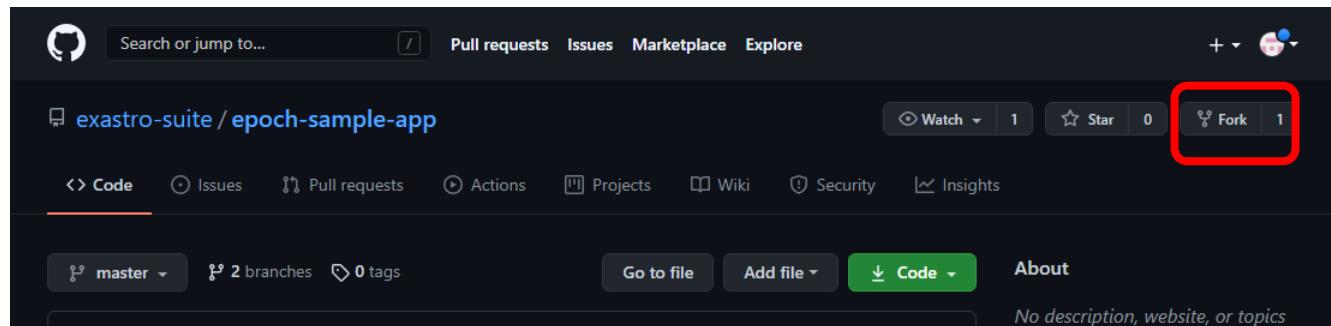
■ アプリケーションコード用リポジトリの準備

- ブラウザにてサンプルアプリケーションのURLを表示します。

<https://github.com/exastro-suite/epoch-sample-app>

- サンプルアプリケーションの画面から「Fork」を押下して、アプリケーションコード用リポジトリを作成します。

(「Fork」押下後にGitHubのサインインを求められたときはご自身のGitHubアカウントでサインインしてください)



- アプリケーションコード用リポジトリのclone

アプリケーションコード用リポジトリをPC環境にcloneします。

例としてコマンドプロンプトでは、以下の通りとなります。

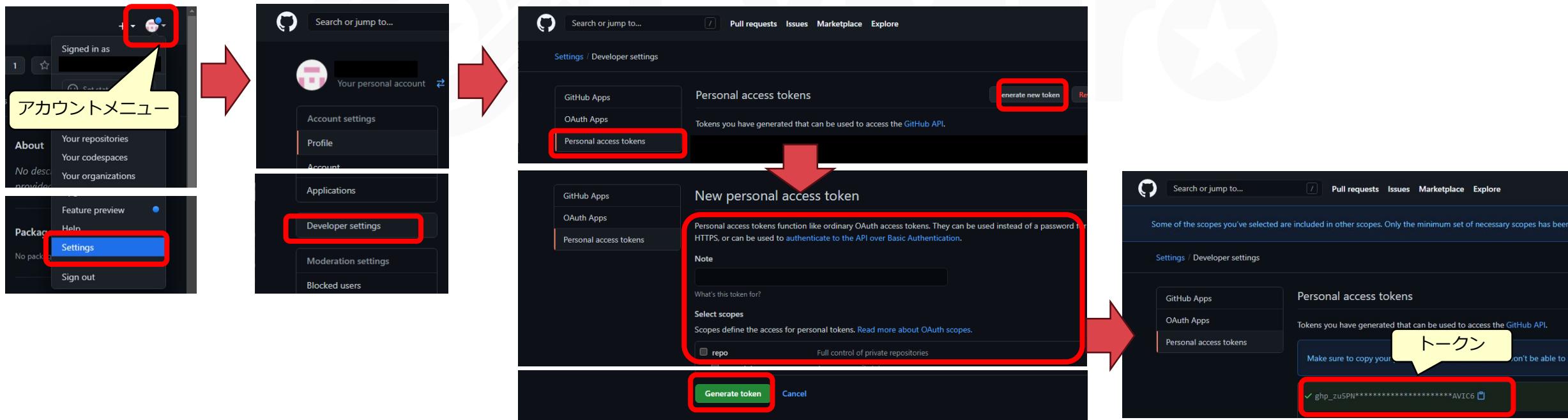
```
> cd "[clone先のフォルダ]"
> git clone https://github.com/[Githubのアカウント名]/epoch-sample-app.git
> cd epoch-sample-app
> git config user.name "[GitHubのユーザ名]"
> git config user.email "[GitHubのemailアドレス]"
```

ここでcloneしたソースコードを使って、チュートリアルを行います。

2.2 リポジトリ準備(3/4)

Git トークンの払い出し

- ブラウザにて自身のGitHubのアカウントでGitHubにサインインします。
- アカウントメニューからSettingsを選択します。
- Account settings画面からDeveloper settingsメニューを選択します。
- Developer settings画面からPersonal access tokensメニューを選択し、Generate new tokenボタンを選択します。
- New personal access token画面でNote(任意の名称)、Select scopesを選択し、Generate tokenボタンを選択します。
- 表示されたトークン(ghp_*****)を後に使用しますので控えてください。

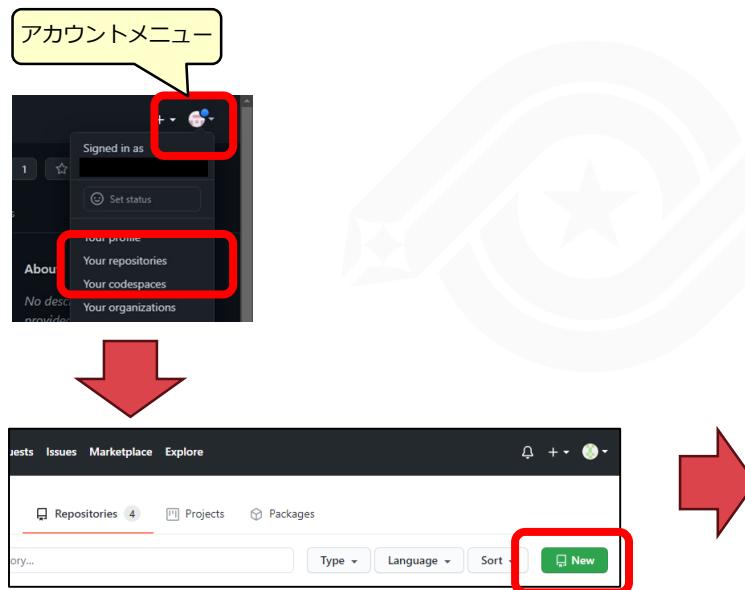


2.2 リポジトリ準備(4/4)

IaC用リポジトリの準備

Manifestを格納するGitリポジトリを2つ用意します。

- ブラウザにて自身のGitHubのアカウントでGitHubにサインインします。
- アカウントメニューからYour Repositoriesを選択します。
- Newを選択し、図で示した値を入力し、Create repositoryを選択します。



The second part of the diagram shows the 'Create a new repository' form on GitHub. A red box highlights the 'Owner' dropdown set to 'your-account' and the repository name 'epoch-sample-staging-manifest'. A yellow callout box contains the text: '次のRepository nameを指定
1 : epoch-sample-staging-manifest
2 : epoch-sample-production-manifest'. Another yellow callout box points to the 'Add a README File' checkbox, which is checked. A final red box highlights the 'Create repository' button at the bottom.

Create a new repository

A repository contains all project files, including the revision history. Already have [Import a repository](#).

Owner *

your-account / epoch-sample-staging-manifest ✓

Great repository names are short and memorable. Need inspiration? How about [furry-octo-umbrella](#)?

Description (optional)

Public Anyone on the internet can see this repository. You choose who can commit.

Private You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

Add a README file This is where you can write a long description for your project. [Learn more](#).

Add .gitignore Choose which files not to track from a list of templates. [Learn more](#).

Choose a license A license tells others what they can and can't do with your code. [Learn more](#).

Create repository

2.3 Manifestテンプレートファイルの準備

Manifestテンプレートファイルのダウンロード

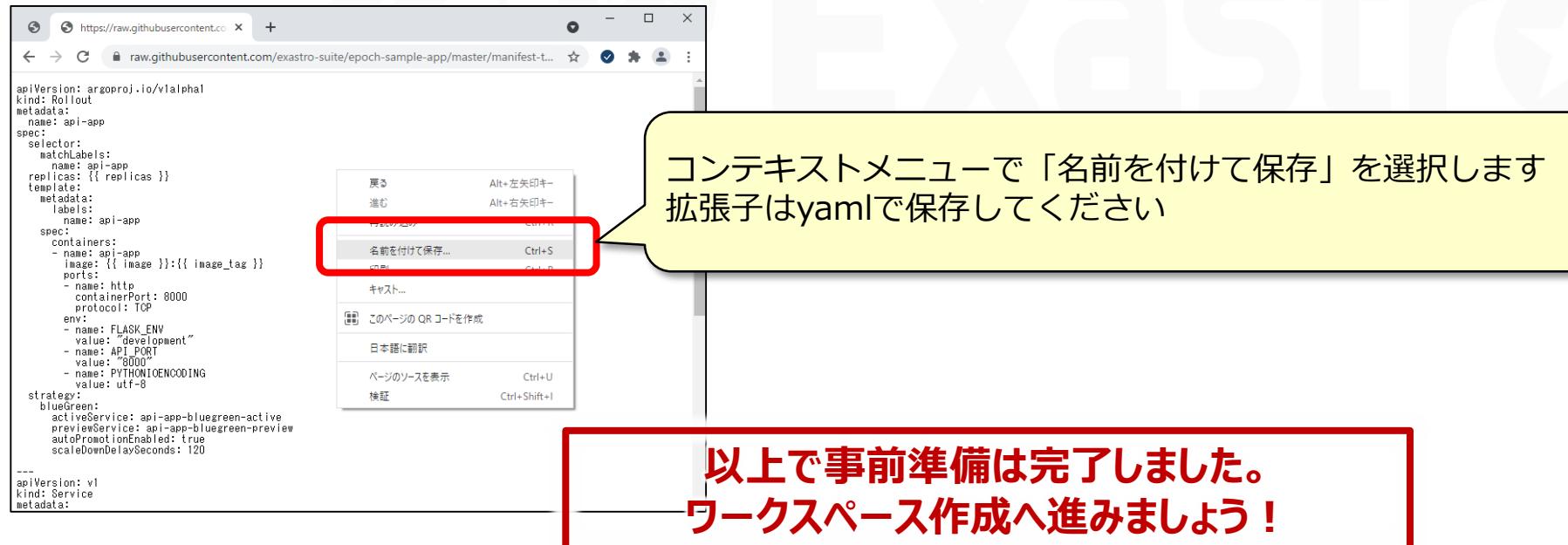
EPOCHにアップロードするManifestテンプレートファイル（2ファイル）をダウンロードします。

- ブラウザで以下のURLを表示します。

ファイル1 : <https://raw.githubusercontent.com/exastro-suite/epoch-sample-app/master/manifest-template/api-app.yaml>

ファイル2 : <https://raw.githubusercontent.com/exastro-suite/epoch-sample-app/master/manifest-template/ui-app.yaml>

- ブラウザにManifestテンプレートが表示されますので、操作しているPCに保存します。





3. ワークスペース作成

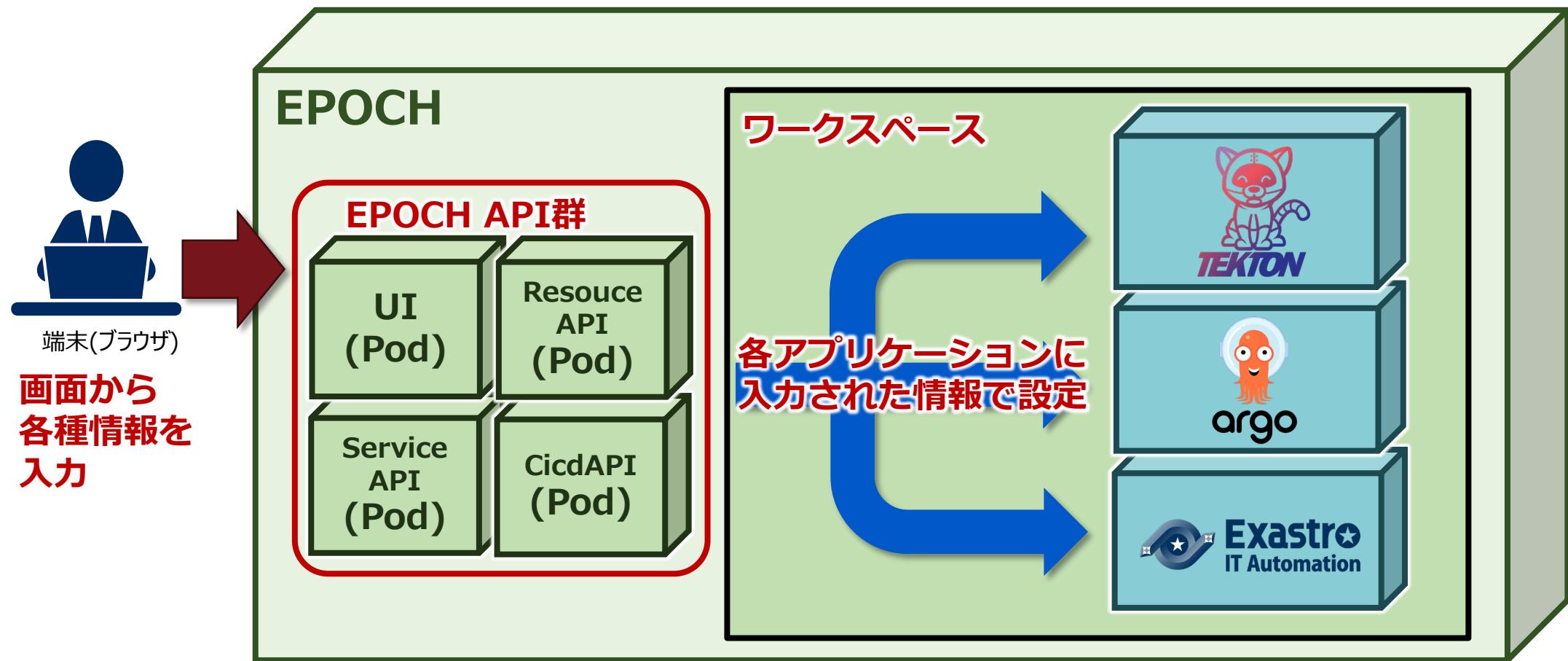
ワークスペースを作成し、CI/CDの準備をしましょう。

3.1 ワークスペース

ワークスペース

EPOCHでは、1つの開発環境をワークスペースという単位で管理します。

ワークスペース作成は、画面から入力された情報をもとに、各アプリケーションへ必要な情報を登録し、CI/CDの準備を行ないます。



3.2 CI/CDについて

CI/CDとは

アプリケーションの開発～リリースまでの一連の作業を自動化し、アプリケーション提供の頻度を高める手法です。

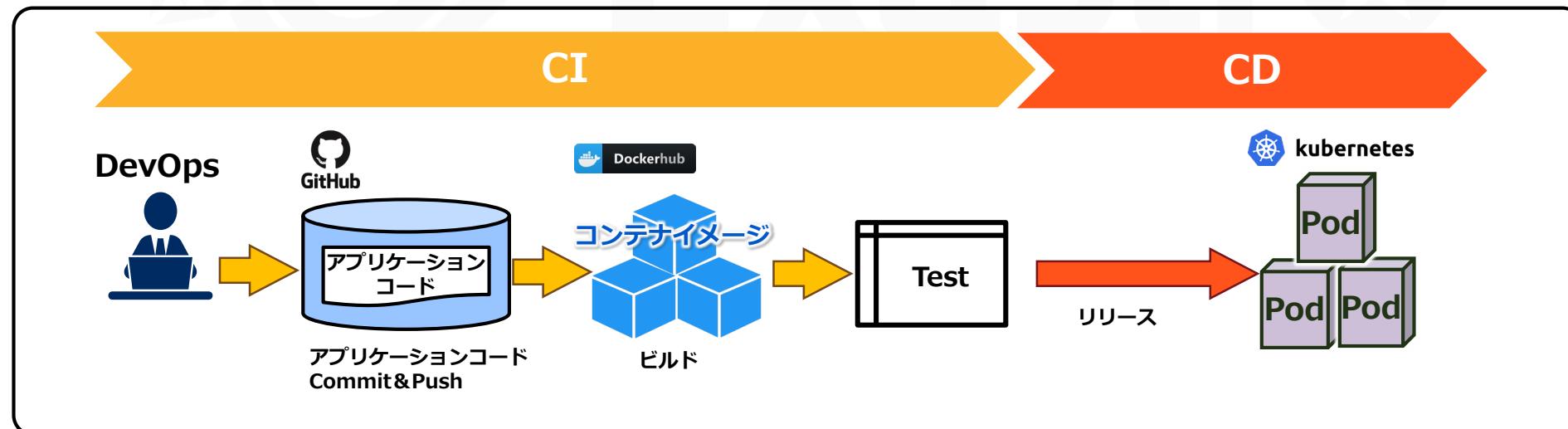
- CI（継続的インテグレーション）

アプリケーションコードの変更を起点に、ビルドやテストの実行といった開発者の作業を自動化する手法を指します。

- CD（継続的デリバリー）

実行環境へのリリースまでを自動化する手法を指します。

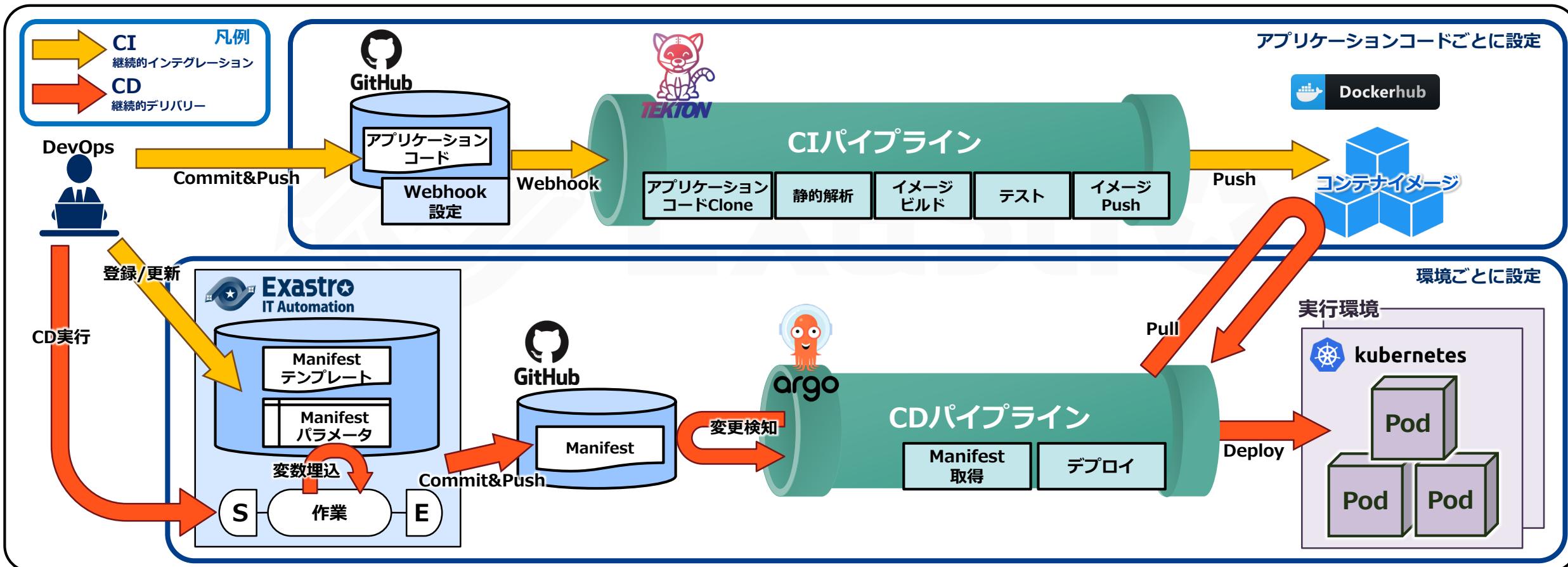
- CI/CDのイメージ



3.3 EPOCHのCI/CD

EPOCHのCI/CD

EPOCHのCI/CDの流れを、下図に示します。



3.4 EPOCH起動

ブラウザより以下のURLで接続します。

http://[インストール先のIPアドレスまたはホスト名]:30080/workspace.html

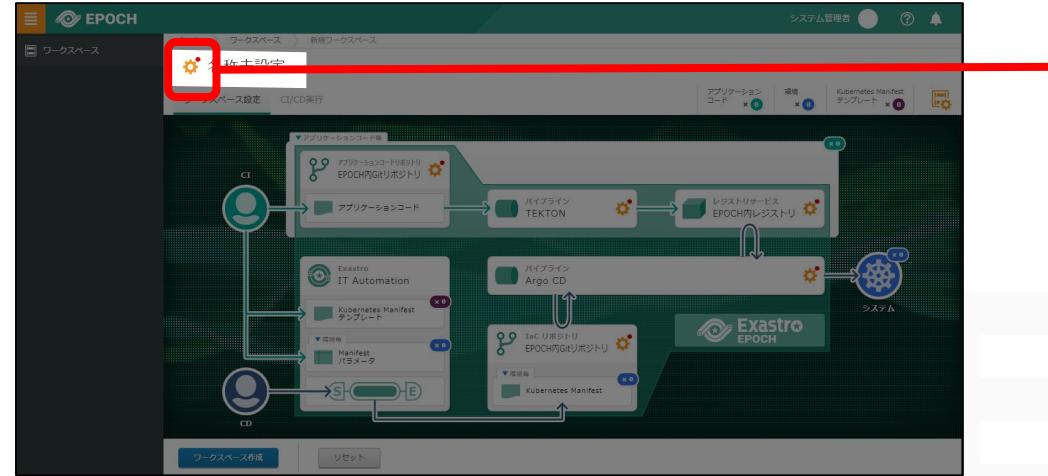


※本QuickStartでは、httpsではなくhttpを使用しております
(今後httpsベースへ変更する予定です)

3.5 ワークスペース作成(1/7)

ワークスペース基本情報

ワークスペース名を入力します。



項目	入力・選択内容	説明
ワークスペース名	EPOCHクイックスタート	作成するワークスペース名
備考	なし	作成するワークスペースの説明や備考

3.5 ワークスペース作成(2/7)

アプリケーションコードリポジトリ

アプリケーションコードリポジトリの情報を入力します。

GitHubを選択

GitHubを選択

Gitサービス選択

EPOCH内Gitリポジトリ GitHub

Gitアカウント指定

ユーザ名: your-github-account

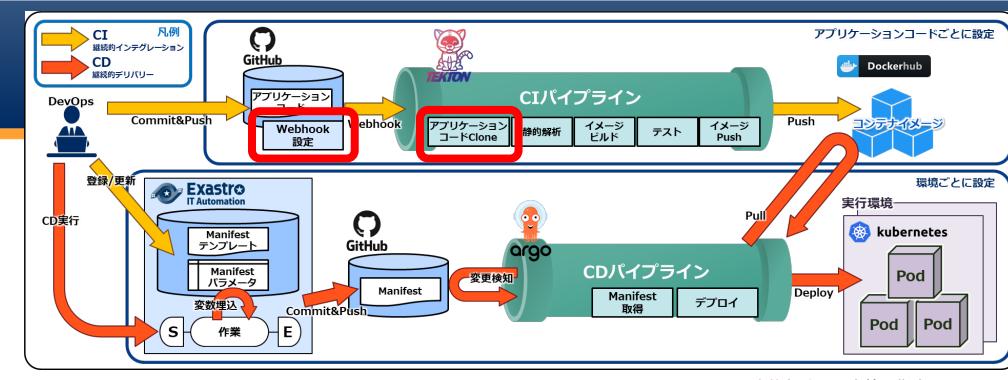
トークン: [REDACTED]

Gitリポジトリ一覧

epoch-sample-app

Gitリポジトリ URL: https://github.com/your-github-account/epoch-sample-app.git

決定 キャンセル

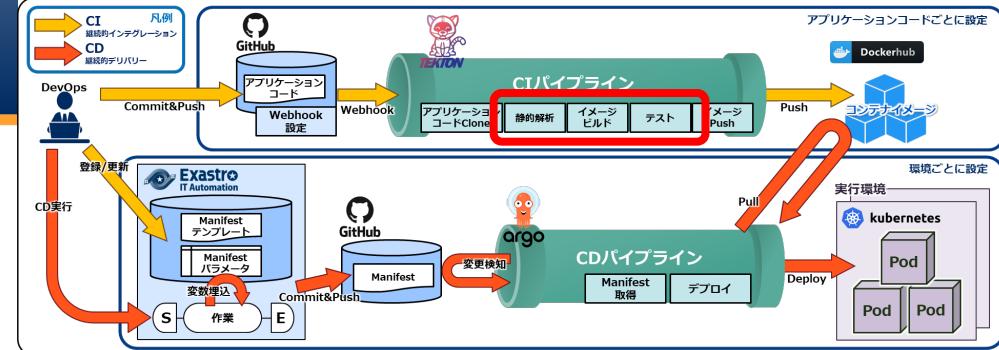
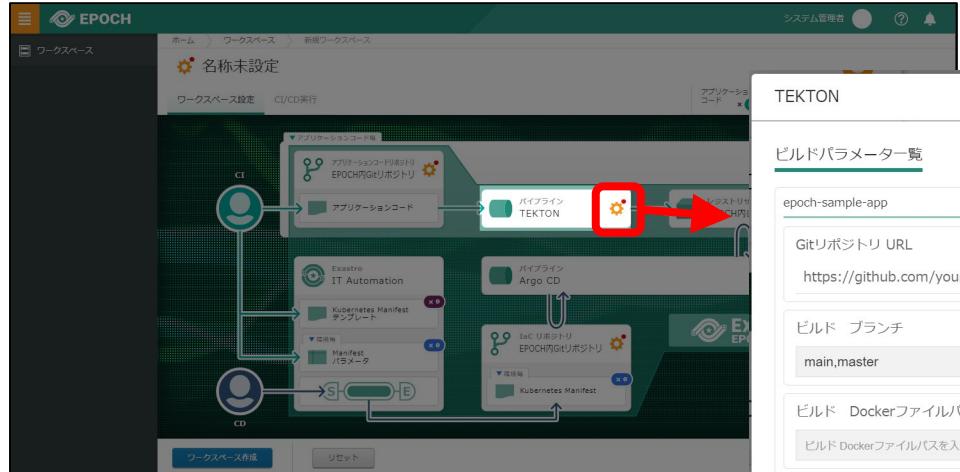


項目	入力・選択内容	説明
ユーザ名	(自身のGitHubのアカウント名)	GitHubのアカウント名
トークン	(自身のGitHubのトークン)	GitHubのトークン (事前準備 Gitトークンの払い出しを参照)
GitリポジトリURL	https://github.com/[GitHubのアカウント名]/epoch-sample-app.git	アプリケーションコードをforkしたリポジトリのURL

3.5 ワークスペース作成(3/7)

パイプラインTEKTON

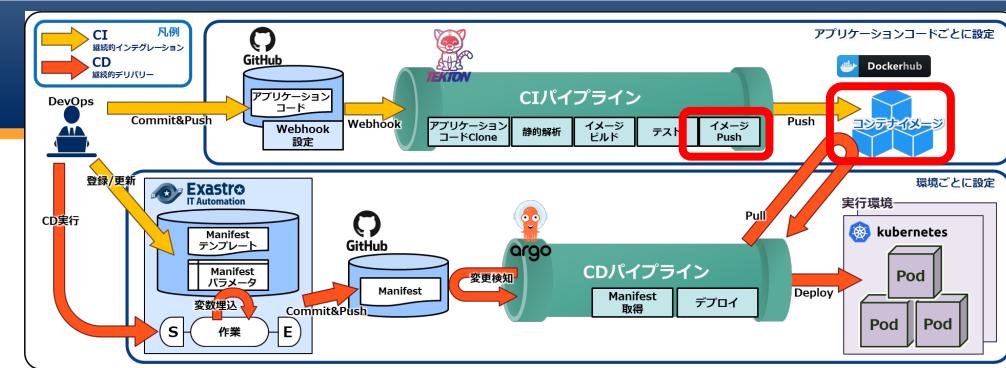
TEKTONに設定するパイプライン情報を入力します。



※赤枠部分の設定値を指定しています
※静的解析、テスト等については対応する予定です

項目	入力・選択内容	説明
ビルドブランチ	main, master	ビルド対象のアプリケーションのGitHubのブランチ
ビルドDockerファイルパス	./api-app/Dockerfile	アプリケーションのDockerfileのパス

3.5 ワークスペース作成(4/7)



レジストリサービス

ビルド後のイメージ登録先（レジストリ）情報を入力します。

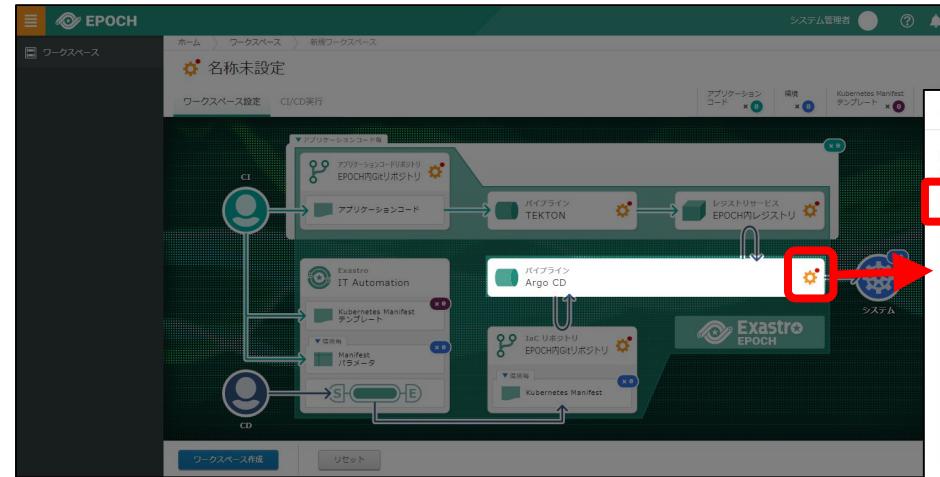
The screenshot shows the EPOCH UI for workspace creation. On the left, there's a large diagram of the CI/CD pipeline. On the right, a modal window titled 'レジストリサービス' (Registry Service) is open. It has sections for 'レジストリサービス選択' (Registry Service Selection) and 'レジストリ接続アカウント' (Registry Connection Account). In the selection section, 'DockerHub' is selected (indicated by a red box and a callout 'DockerHubを選択'). In the account section, the user name is 'your-dockerhub-account' and the password is masked. Below that, under 'コンテナイメージ出力指定' (Container Image Output Specification), the Git repository URL is 'https://github.com/your-dockerhub-account/epoch-sample-app.git' and the image output path is 'your-dockerhub-account/epoch-sample-api'.

項目	入力・選択内容	説明
ユーザ名	(自身のDockerHubのアカウント名)	DockerHubのアカウント名
パスワード	(自身のDockerHubのパスワード)	DockerHubのパスワード
イメージ出力先	[DockerHubのアカウント名]/epoch-sample-api ※ユーザ名入力後に表示される内容を修正してください。	DockerHubのイメージ出力先のパス

3.5 ワークスペース作成(5/7)

パイプラインArgo CD

ArgoCDに設定するDeploy先の情報を入力します。



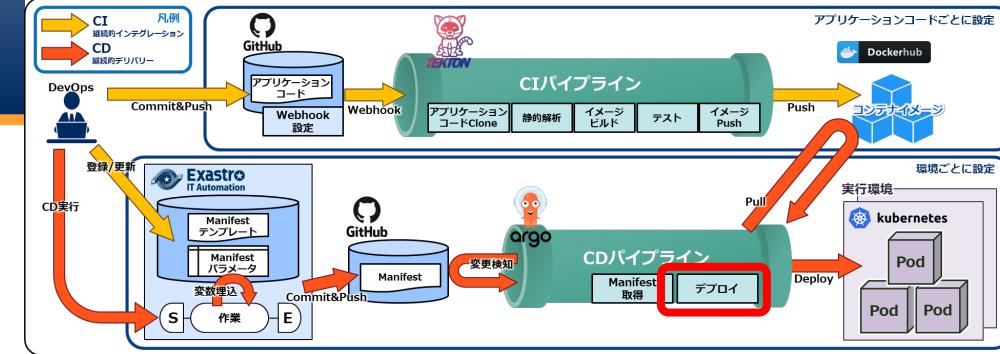
環境1 : Staging環境

項目	入力・選択内容	説明
環境名	staging	デプロイ環境の名前
Namespace	epoch-sample-app-staging	デプロイ先のNamespace

環境2 : Production環境

項目	入力・選択内容	説明
環境名	production	デプロイ環境の名前
Namespace	epoch-sample-app-production	デプロイ先のNamespace

環境1つめ2つめの切り替えはタブを選択して切り替えます



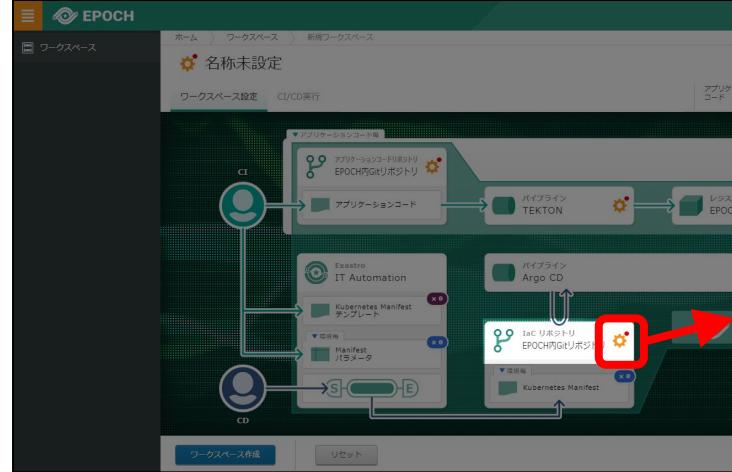
本QuickStartでは2つの環境に対してDeployしますので、【環境追加】ボタンを押下して、環境を追加します

内部クラスタを選択

3.5 ワークスペース作成(6/7)

IaCリポジトリ

マニフェストの登録先となるリポジトリ情報を入力します。



IaCリポジトリ

GitHubを選択

Gitサービス選択

GitHub

Gitアカウント指定

アプリケーションコードリポジトリと同一

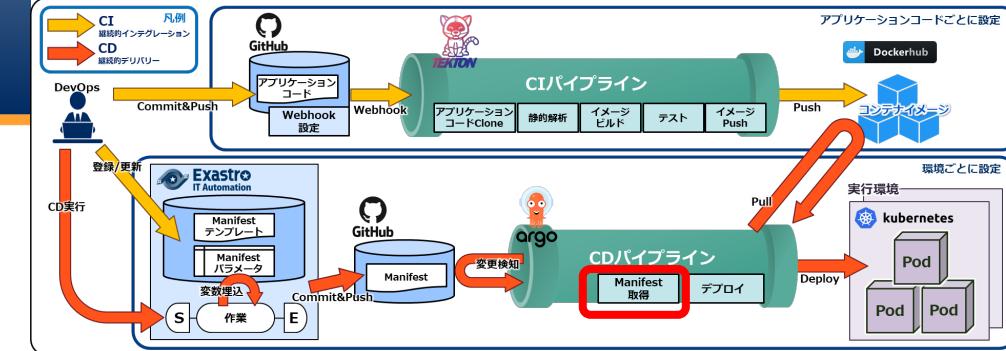
ユーザ名 (アプリケーションコードリポジトリと同一)
epoch-user

Gitリポジトリ URL

環境ごとに入力

staging production

決定 キャンセル

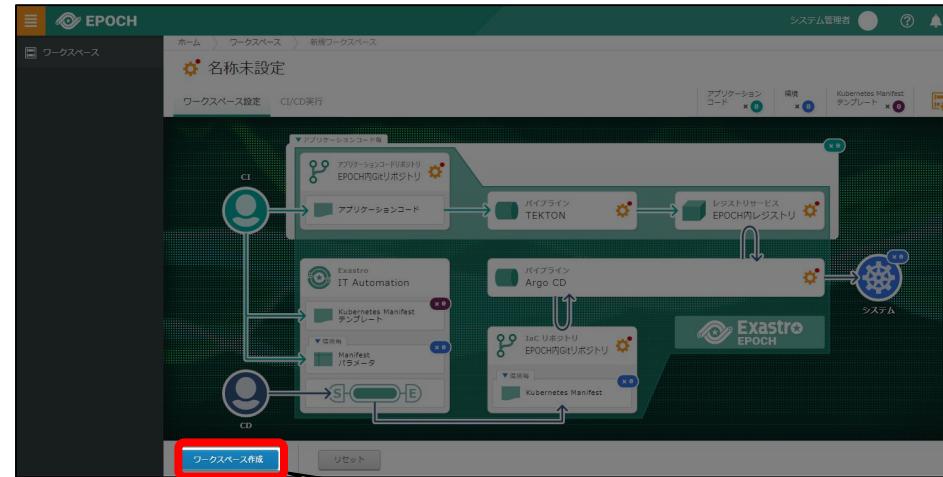


項目	入力・選択内容	説明
ユーザ名	(自身のGitHubのアカウント名)	GitHubのアカウント名
トークン	(自身のGitHubのトークン)	GitHubのトークン (事前準備 Gitトークンの払い出しを参照)
GitリポジトリURL	https://github.com/[GitHubのアカウント名]/[各環境のリポジトリ].git	各環境のmanifestリポジトリのURL (事前準備 IaC用リポジトリの準備を参照)

3.5 ワークスペース作成(7/7)

ワークスペース作成

すべての入力が完了しましたら【ワークスペース作成】ボタンを押下します。



【ワークスペース作成】ボタンを押下します

これでCI/CDパイプラインが構築されました。
チュートリアルを実践してCI/CDパイプラインを体験してみましょう！



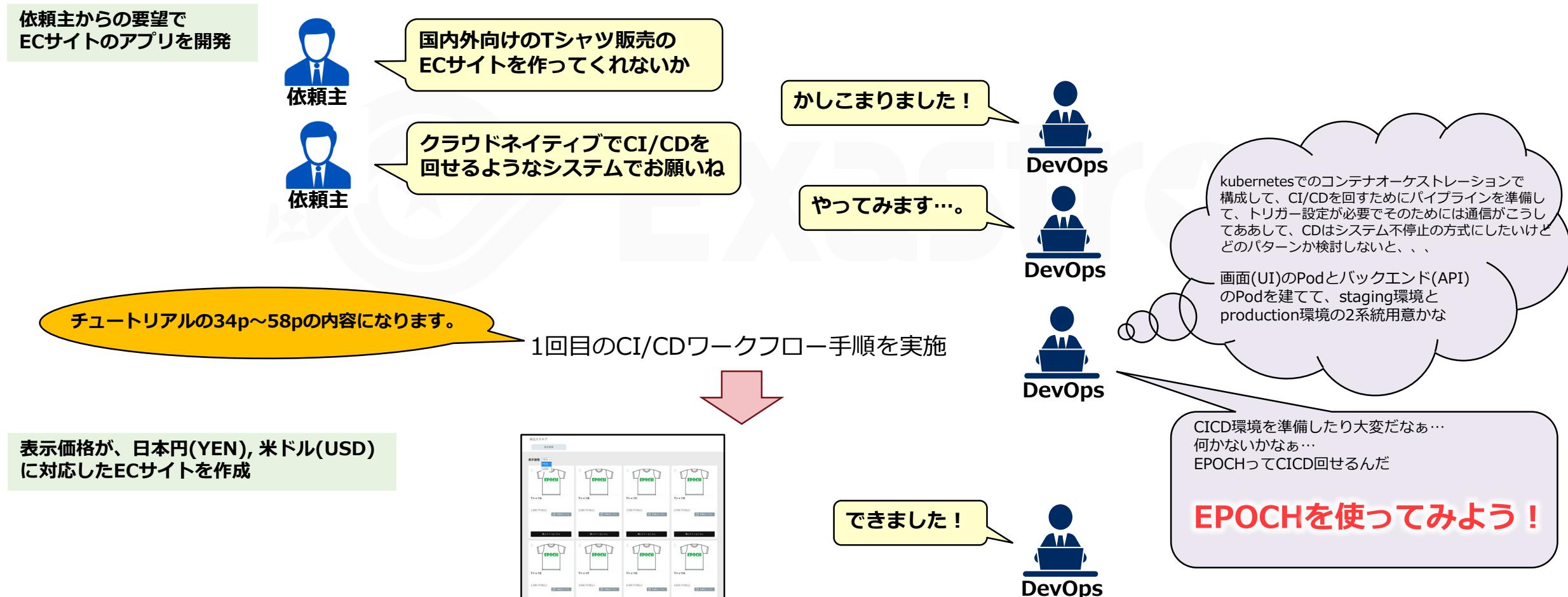
4. チュートリアル

CI/CDワークフローを体験してみましょう。

4.1 チュートリアルの概要(1/2)

CI/CD開発シナリオ

チュートリアルでは以下のシナリオに沿って、CI/CDワークフローの手順を実施していきます。本QuickStartで作成するECサイトは、サンプルアプリケーションを用いて実施していきます。



4.1 チュートリアルの概要(2/2)

少し経ってから
追加案件が発生



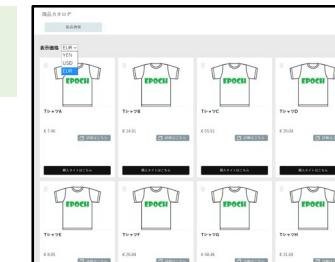
この前のECサイトはとても好評
だったよ



ヨーロッパ向けにも展開したい
通貨にEURも追加してほしい

チュートリアルの59p~80pの内容になります。

表示価格が、日本円(YEN), 米ドル(USD), **ユーロ(EUR)**
に対応したECサイトを作成



2回目のCI/CDワークフロー手順を実施



ありがとうございます



かしこまりました！



CI/CDの仕組みはできているから
コード修正してからデプロイまでは
単純作業♪

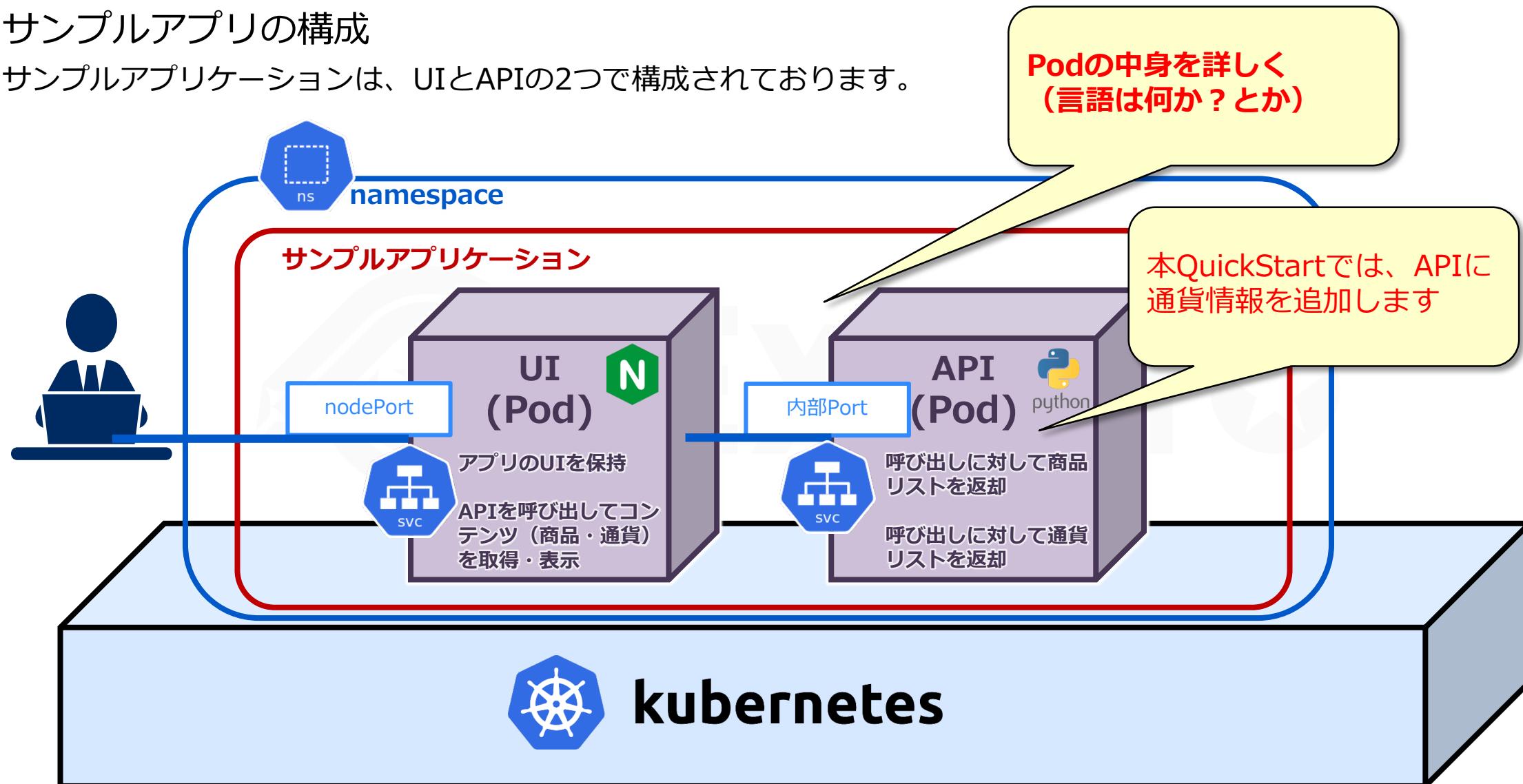
できました！



4.2 サンプルアプリの構成

サンプルアプリの構成

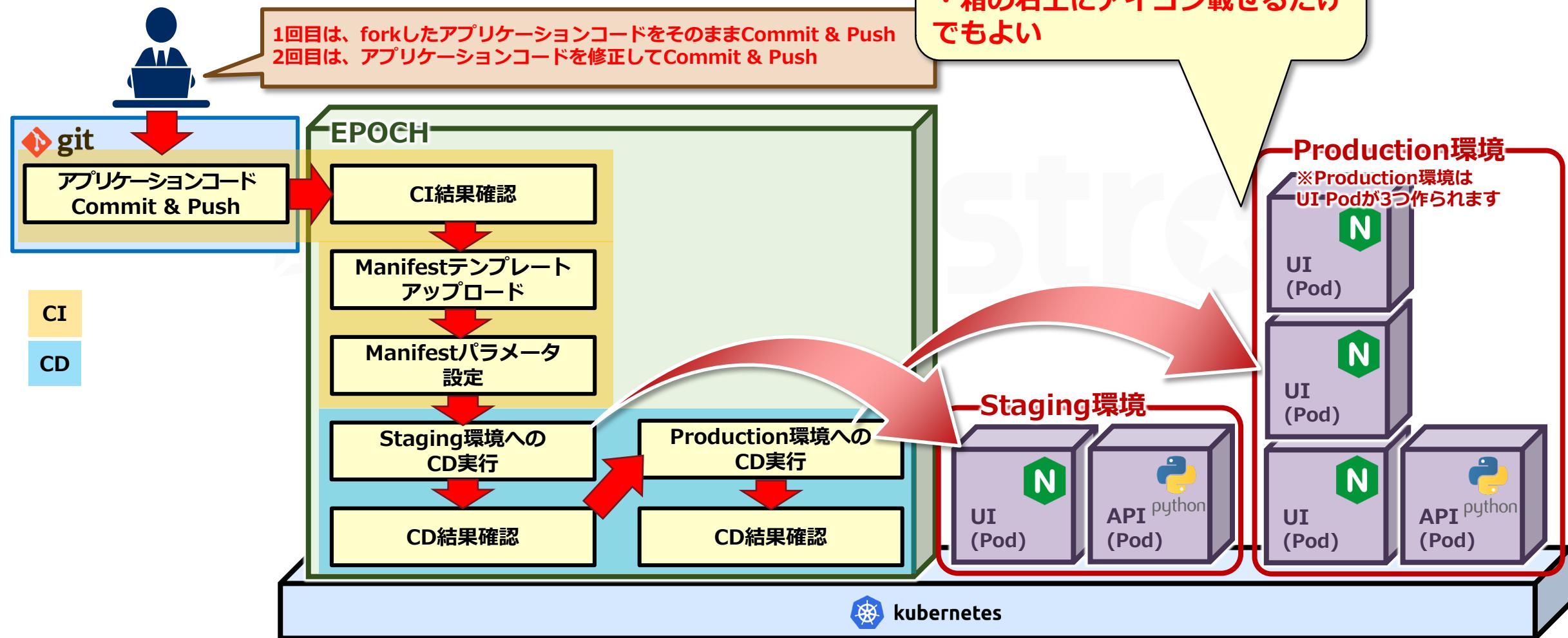
サンプルアプリケーションは、UIとAPIの2つで構成されております。



4.3 チュートリアルの流れ(CI/CDワークフロー)

CI/CDワークフロー

本説明では、サンプルアプリケーションをStaging環境、Production環境へ行い、Staging環境、Production環境へのDeployする手順を説明します。



4.4 Manifestテンプレートファイルについて

Manifestテンプレートファイル

サンプルアプリケーションのManifestテンプレートファイルは、UIとAPI用の2つが用意されています。環境一致を考慮した上で可変部分を変数化したテンプレート形式となっており、レプリカ数、イメージ(tag含む)、ポート番号をあらかじめ定義しています。



それでは1回目のCI/CDワークフロー手順を
実行してみましょう！

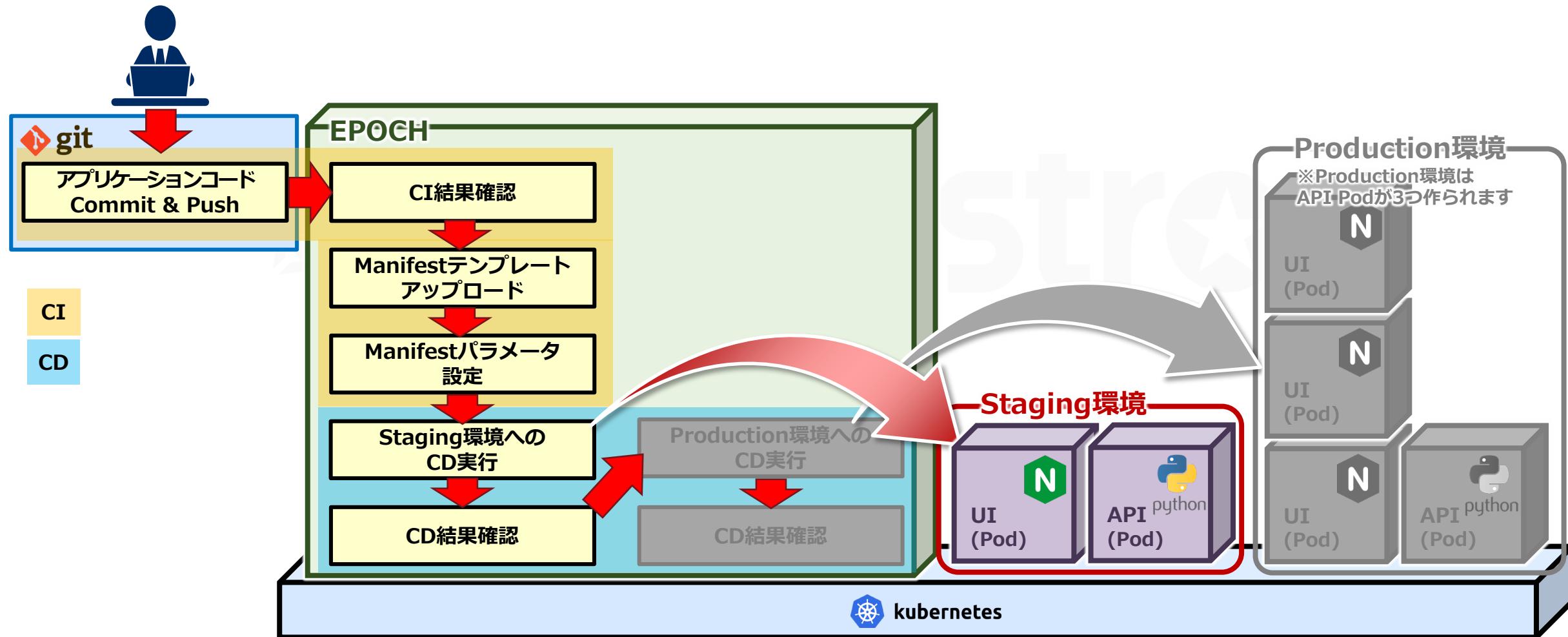
1回目のCI/CDワークフロー手順



4.5 1回目のCI/CDワークフロー手順(1/24)

Staging環境へのDeploy

1回目のCI/CDワークフロー手順として、アプリケーションコードのCommit & PushからStaging環境へのDeploy、CD結果確認までの手順は以下の通りとなります。



4.5 1回目のCI/CDワークフローハンドル(2/24)

アプリケーションコード Commit & Push

初回デプロイするコンテナイメージを作るためCIパイプラインを実行します。

- PC環境にcloneしたアプリケーションコード用リポジトリでCommit & Pushします

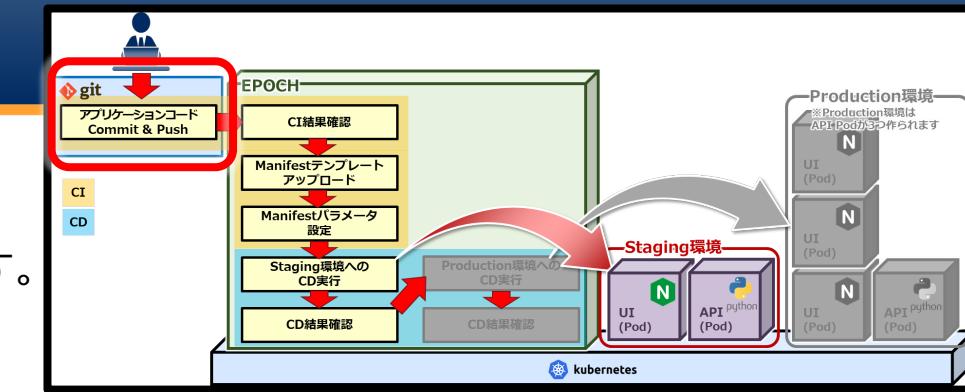
例としてコマンドプロンプトでは、以下の通りとなります。

```
> cd "[clone先のフォルダ]"
> cd epoch-sample-app
> echo "first build" > dummy.txt
> git add .
> git commit -m "first build"
> git push origin master
```

```
Username for 'https://github.com': [自身のGitHubユーザ名を入力]
Password for 'https://xxxxxx@github.com': [自身のGitHubパスワードを入力]
```

Pushが完了すると、パイプラインTEKTONで設定されたCIパイプラインが自動的に動き出します。

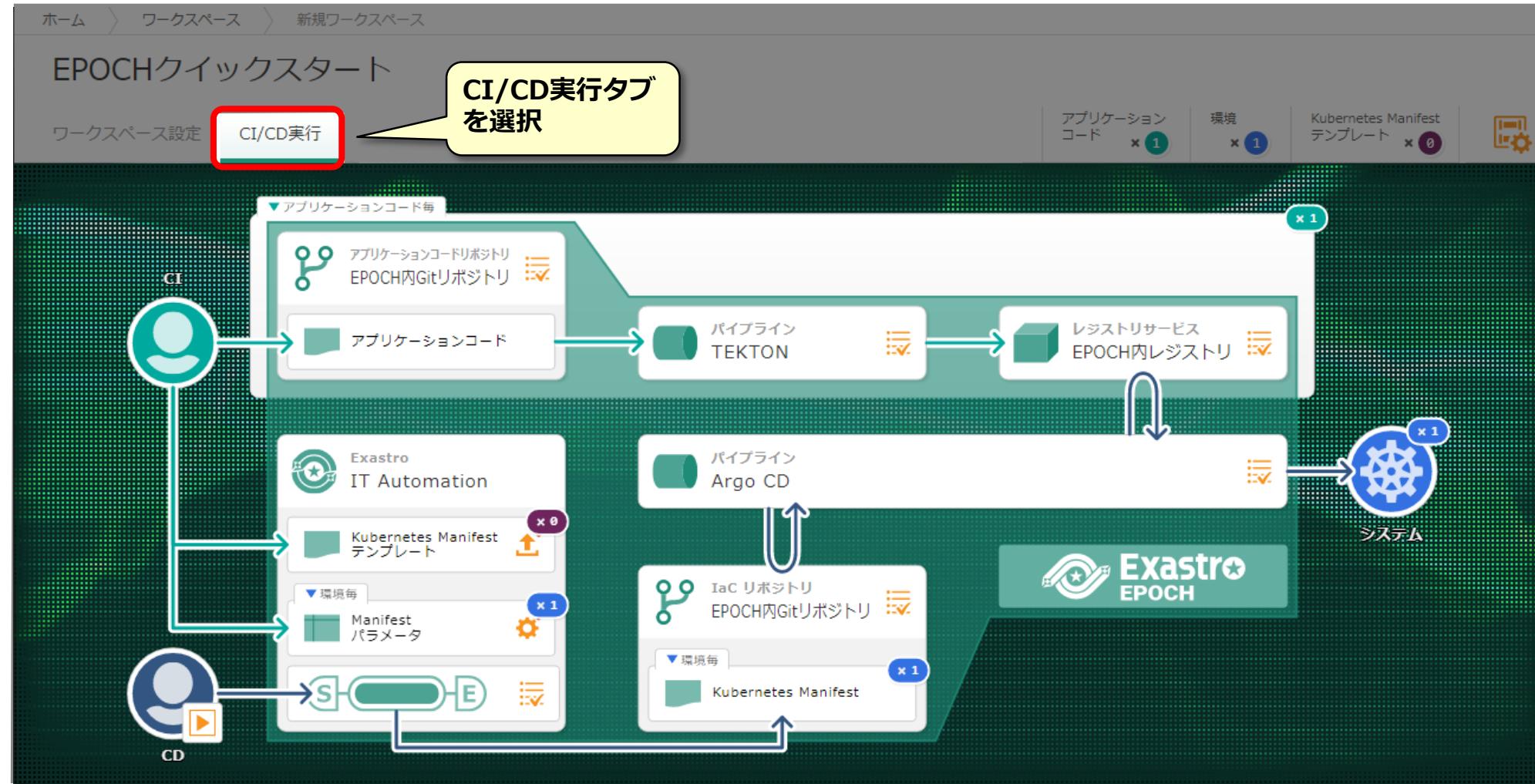
CIパイプラインの結果を確認していきましょう。



4.5 1回目のCI/CDワークフロー手順(3/24)

CI/CD実行画面の表示

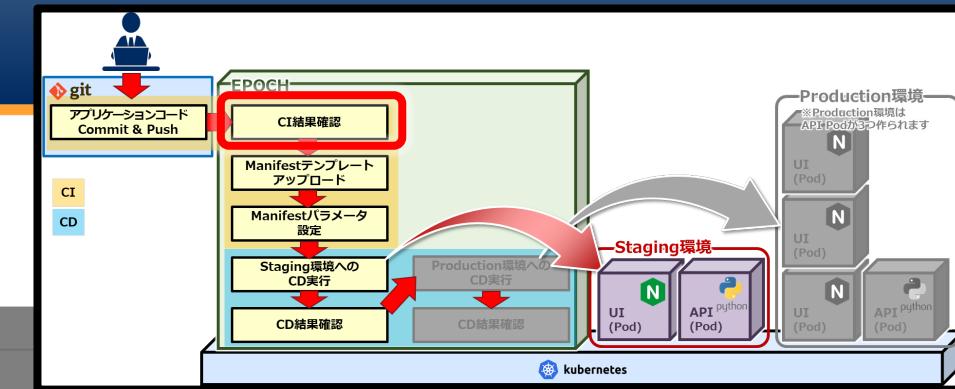
ワークスペース画面のCI/CD実行タブを選択し、CI/CD実行画面を表示します。



4.5 1回目のCI/CDワークフロー手順(4/24)

CI結果確認

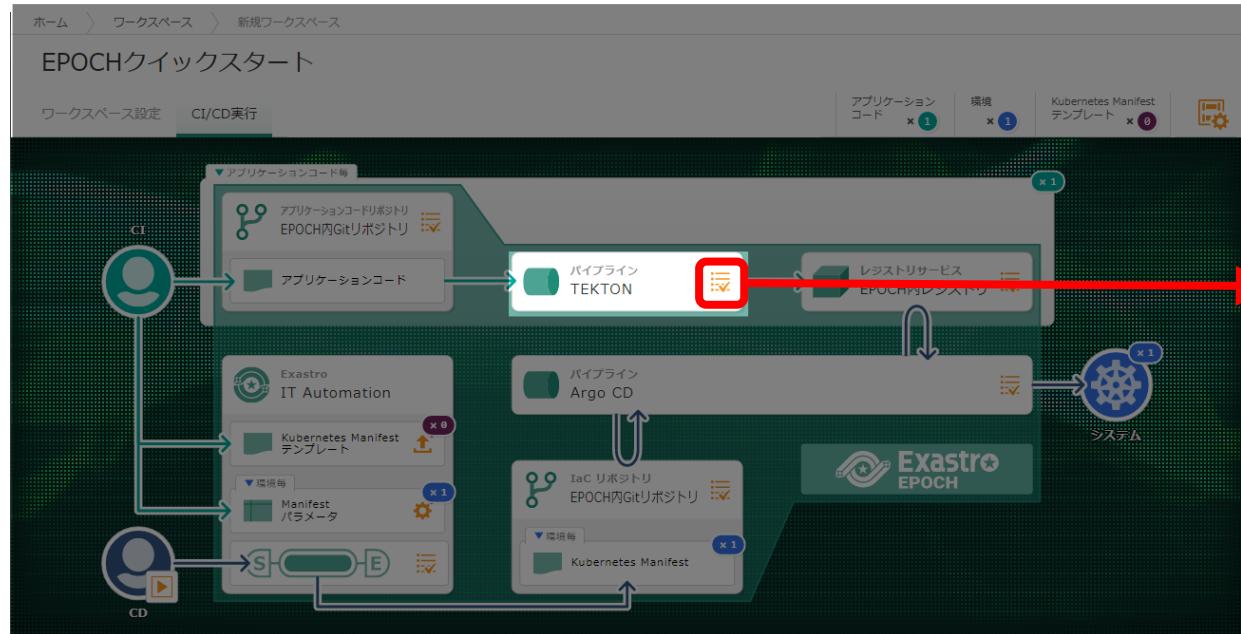
- アプリケーションコードのビルド結果を確認します。



4.5 1回目のCI/CDワークフロー手順(5/24)

■ パイプラインTEKTONの結果確認

- TEKTONのパイプラインを実際に確認し、ビルドが正常に終了したか確認します。



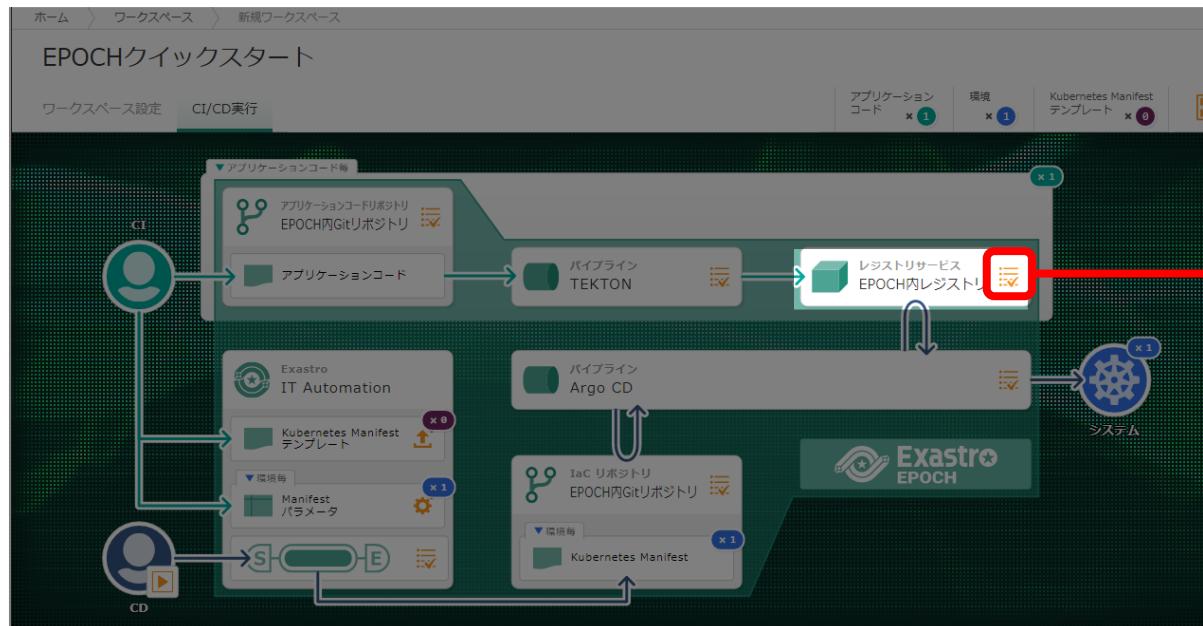
The Tekton Dashboard displays the PipelineRuns page. A table lists a single pipeline run: **build-and-push-pipeline-run** (Pipeline: pipeline-build-and-push, Namespace: epoch-tekon-pipelines, Created: 6 minutes ago, Duration: 3 minutes 32 seconds). A yellow callout box points to the **Status** column, which shows a green checkmark. Another callout box contains the text: "CIパイプラインの動作は、TEKTONダッシュボードにて確認することができます。Nameをクリックすることにより、より詳細なCIパイプラインの内容を確認することができます。" (The CI pipeline's operation can be confirmed on the Tekton Dashboard. Clicking on the Name will allow you to view more detailed CI pipeline information.)

Status	Name	Pipeline	Namespace	Created	Duration
	build-and-push-pipeline-run	pipeline-build-and-push	epoch-tekon-pipelines	6 minutes ago	3 minutes 32 seconds

4.5 1回目のCI/CDワークフロー手順(6/24)

コンテナイメージのタグ名の確認

- レジストリサービスの画面を開き、ビルトしたコンテナイメージのTagを確認します。



The screenshot shows the Docker Hub page for the repository `/epoch-sample-api`. It displays two tags: `master.20210701-171058` and `master.20210701-134114`. A yellow callout box contains the following text:

Dockerhubの画面でepoch-sample-apiのTagを確認します
※イメージが登録されていないときは、パイプラインTEKTONから結果を確認してください

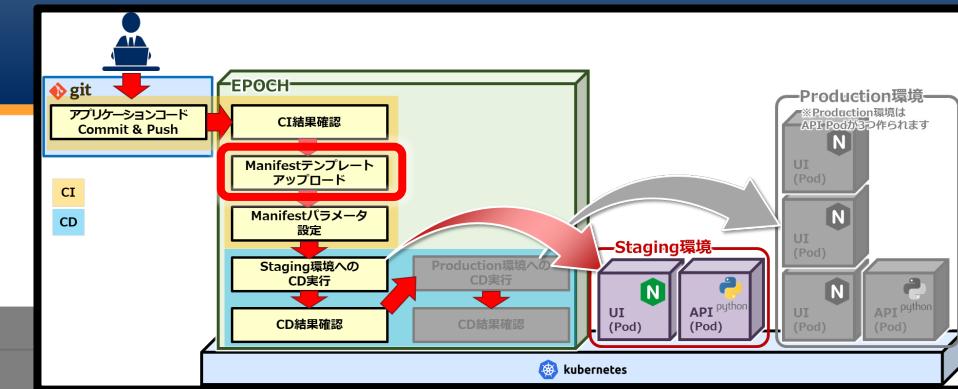
Below the callout, the Docker Hub page shows the following details for each tag:

Tag	Last pushed	OS/ARCH	Compressed Size
master.20210701-171058	7 days ago	linux/amd64	87.84 MB
master.20210701-134114	7 days ago	linux/amd64	87.84 MB

4.5 1回目のCI/CDワークフロー手順(7/24)

Manifestテンプレートアップロード

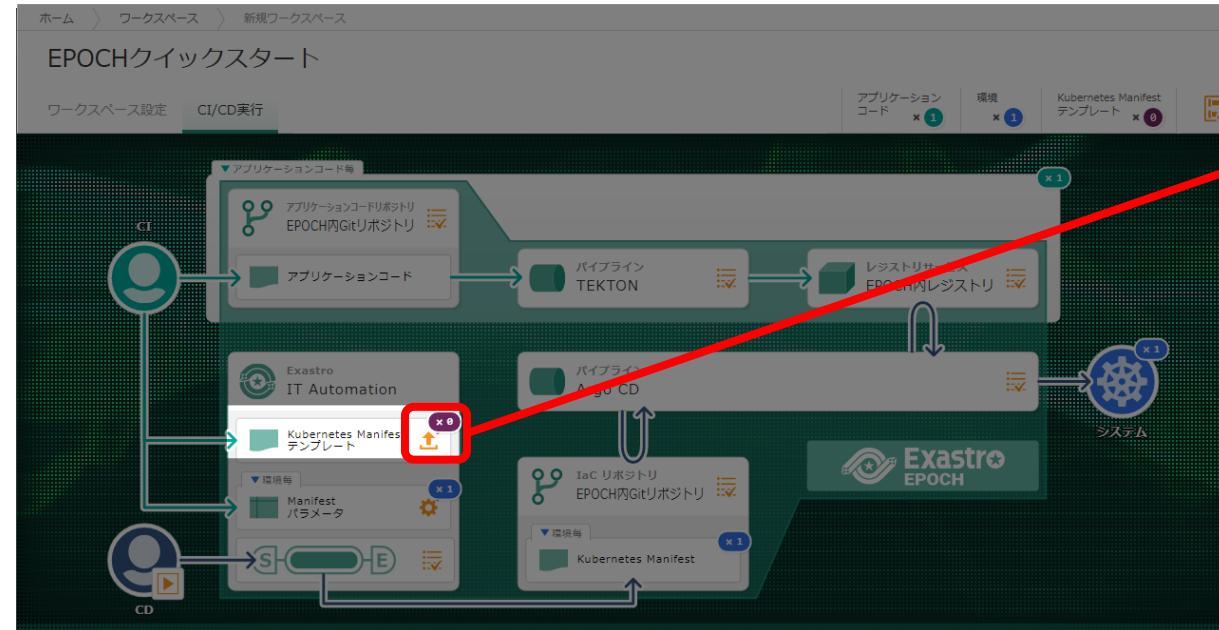
- ダウンロードしたManifestテンプレートをアップロードします。



4.5 1回目のCI/CDワークフロー手順(8/24)

Manifestテンプレートアップロード

- Manifestテンプレートファイルの準備でダウンロードしたManifestテンプレートファイルをアップロードします。



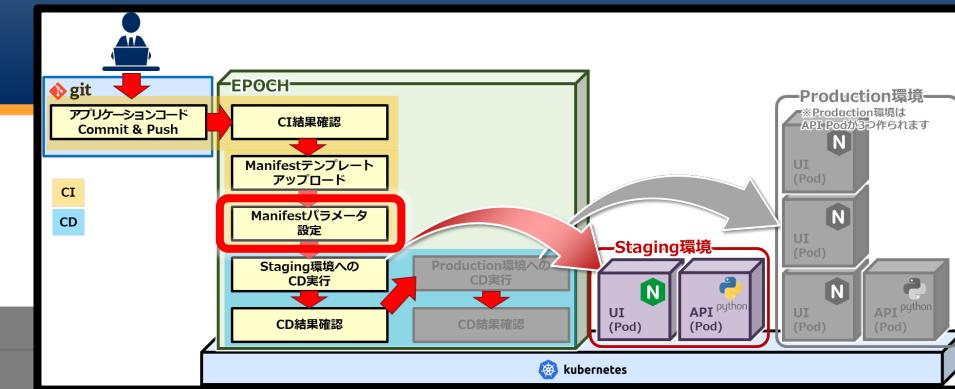
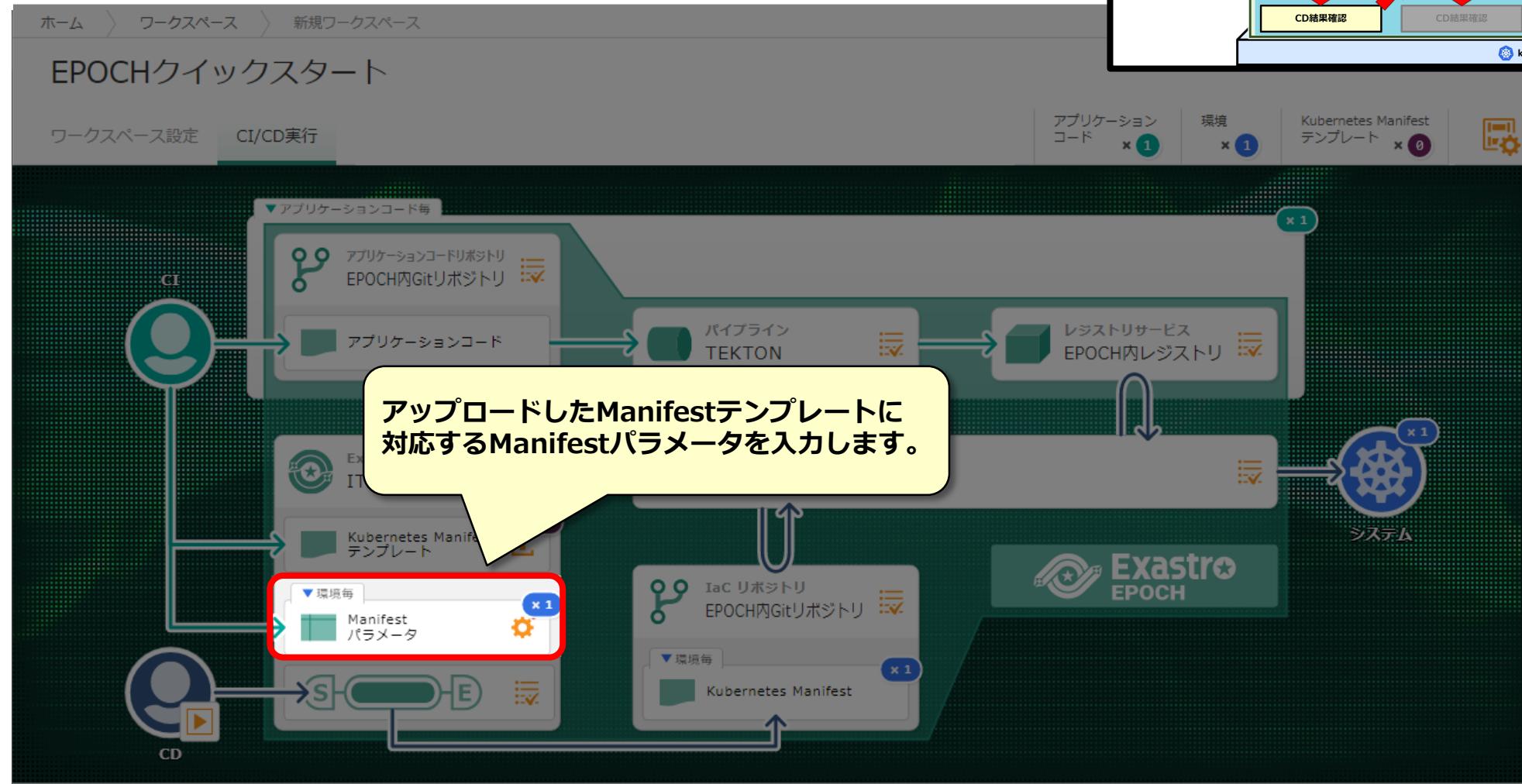
ファイル : api-app.yaml, ui-app.yamlを
右の画面領域にドロップします

The right side of the image shows two overlapping dialog boxes. The top one is titled 'Kubernetes Manifest テンプレート' and has a 'アップロードファイル選択' button highlighted with a red box and arrow. The bottom dialog is titled 'Kubernetes Manifest テンプレートアップロード' and contains a large red box around its central file upload area, which says 'ここにファイルをドロップ または クリック'. At the bottom of this dialog are 'アップロード', '再選択', and 'キャンセル' buttons.

4.5 1回目のCI/CDワークフロー手順(9/24)

Manifestパラメータ

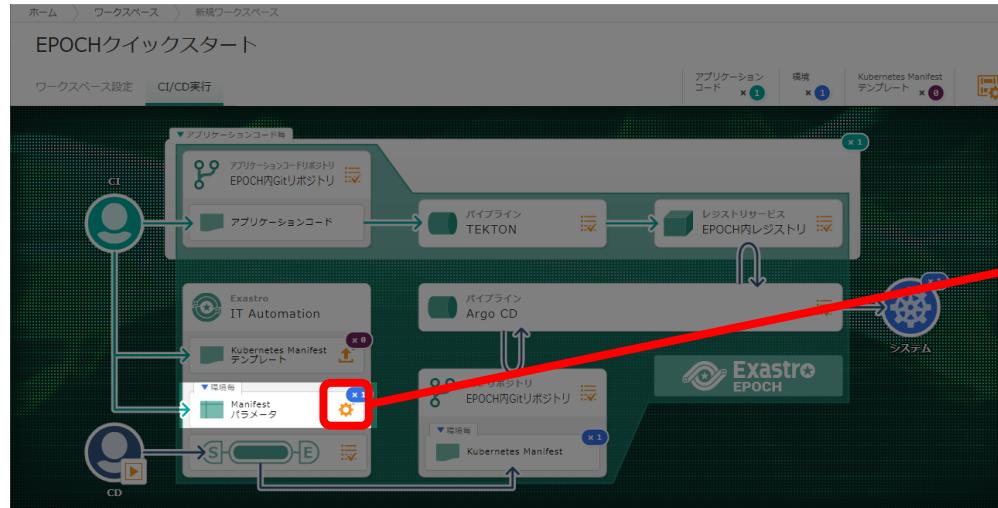
- Deployに必要なManifestパラメータを入力します。



4.5 1回目のCI/CDワークフロー手順(10/24)

Manifestパラメータ入力

- 入力内容に従って、Manifestパラメータを入力します。



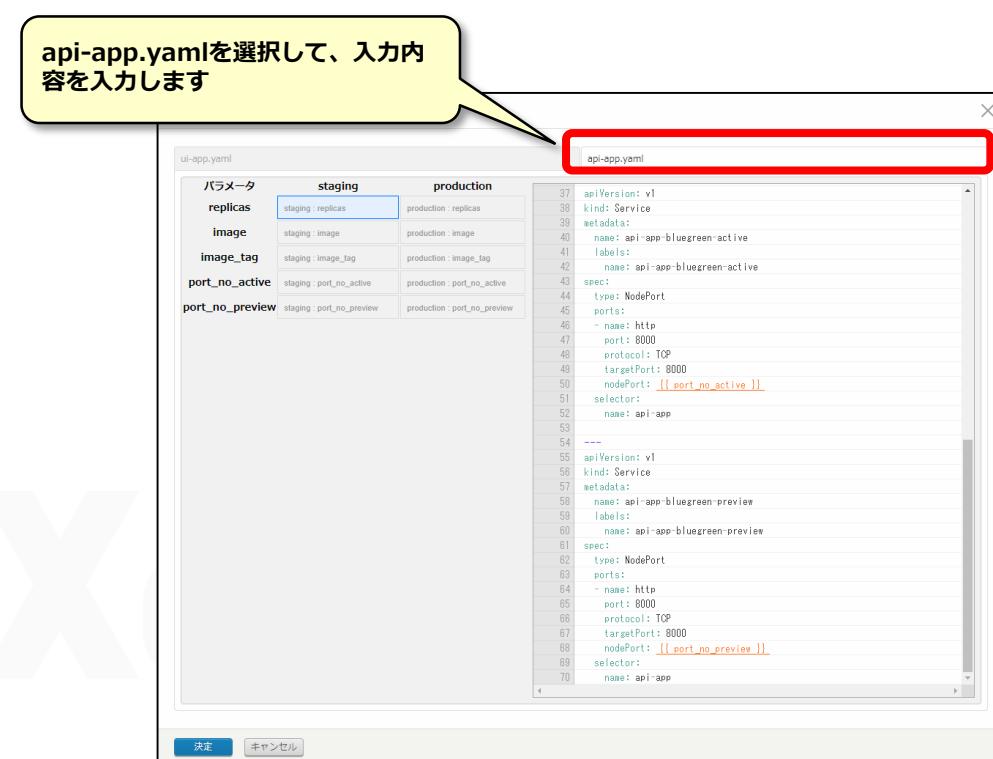
ui-app.yaml

項目	入力内容(staging)	入力内容(production)	説明
<code>replicas</code>	1	3	レプリカ数
<code>image</code>	exastro/epochsampleappui	exastro/epochsampleappui	コンテナイメージ
<code>image_tag</code>	master.20210708183910	master.20210708183910	コンテナイメージのタグ
<code>port_no_active</code>	31001	31003	ブルーグリーンデプロイ用のブルー面のポート番号
<code>port_no_preview</code>	32001	32003	ブルーグリーンデプロイ用のグリーン面のポート番号



4.5 1回目のCI/CDワークフロー手順(11/24)

- タブを切り替えてapi-app.yamlにも入力します。



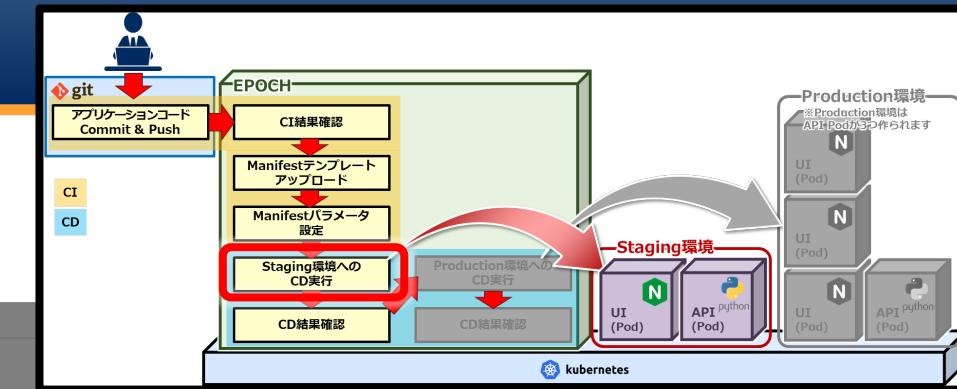
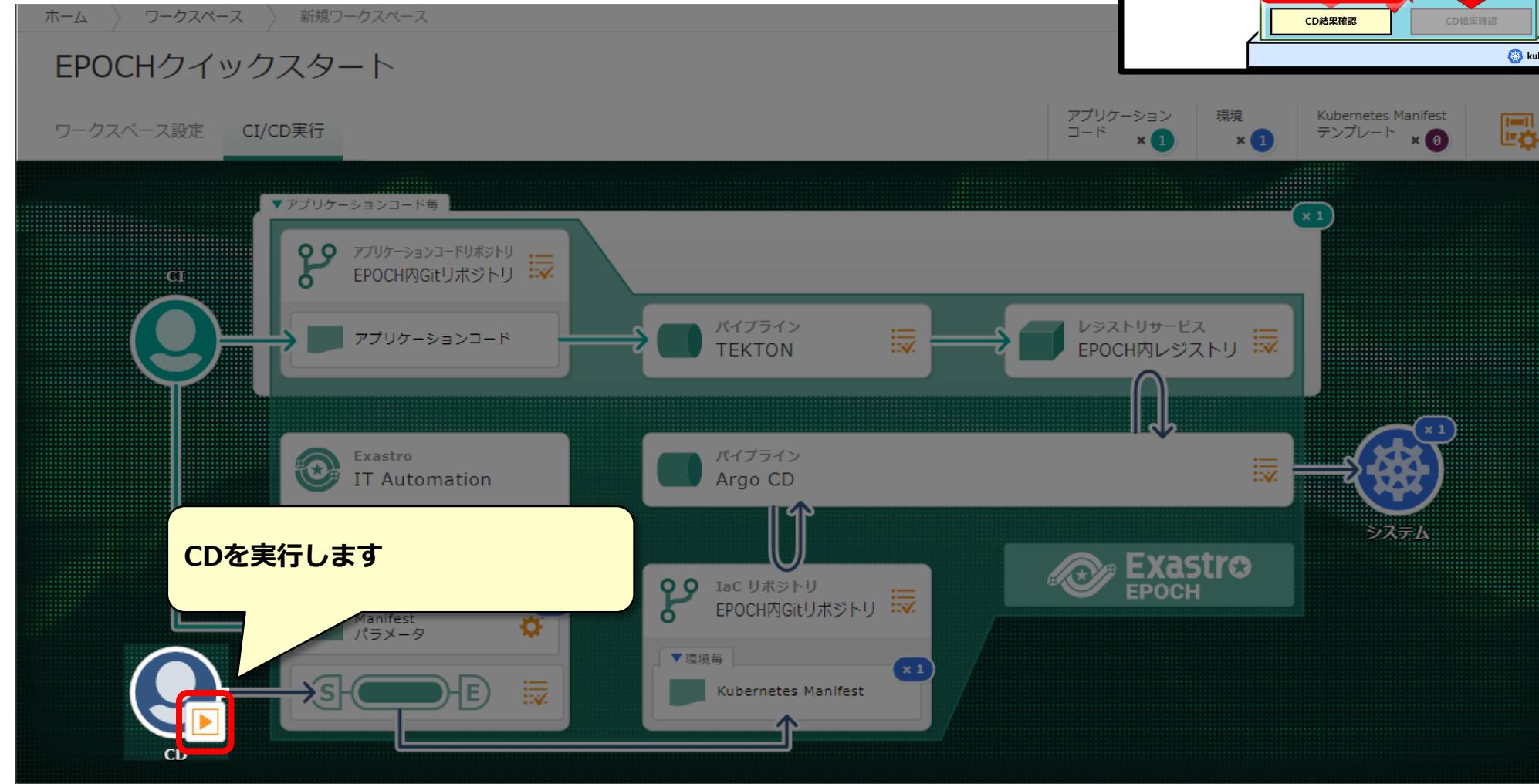
api-app.yaml

項目	入力内容(staging)	入力内容(production)	説明
{{ replicas }}	1	1	レプリカ数
{{ image }}	[Dockerhubのアカウント名]/epoch-sample-api	[Dockerhubのアカウント名]/epoch-sample-api	コンテナイメージ
{{ image_tag }}	[レジストリサービスで確認したimageのタグ名]	[レジストリサービスで確認したimageのタグ名]	コンテナイメージのタグ
{{ port_no_active }}	31002	31004	ブルーグリーンデプロイ用のブルー面のポート番号
{{ port_no_preview }}	32002	32004	ブルーグリーンデプロイ用のグリーン面のポート番号

4.5 1回目のCI/CDワークフロー手順(12/24)

Staging環境へのDeploy実行

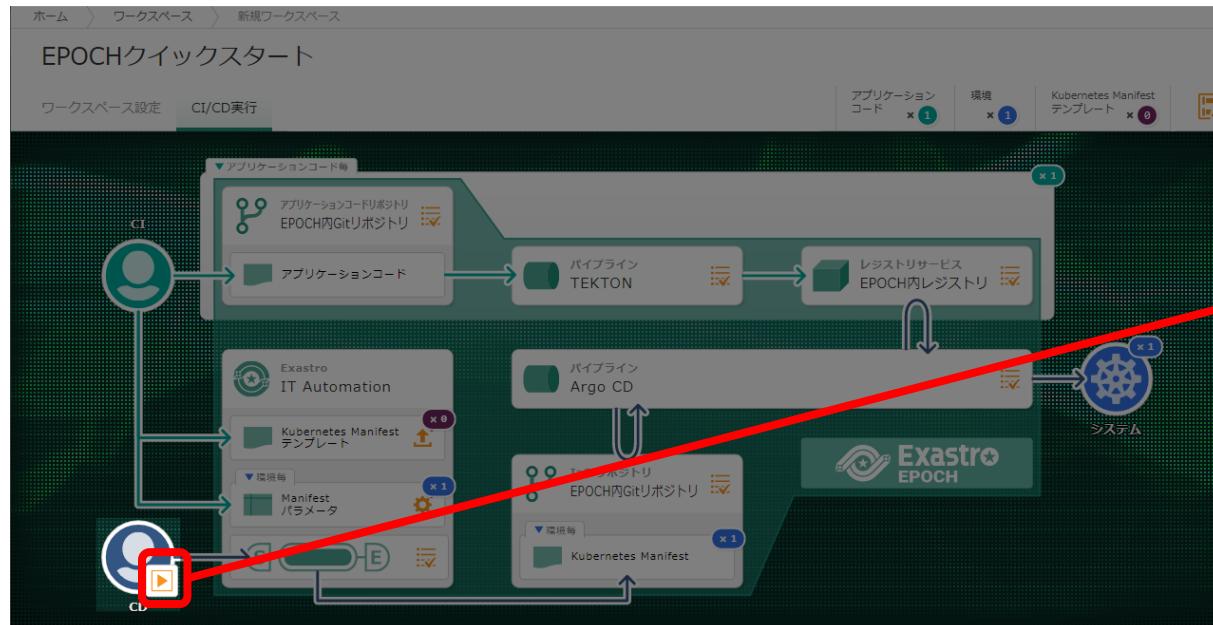
- CD実行で、Staging環境へDeployを実行します。



4.5 1回目のCI/CDワークフロー手順(13/24)

Staging環境のCD実行

- Deploy先の環境を選択して実行します。



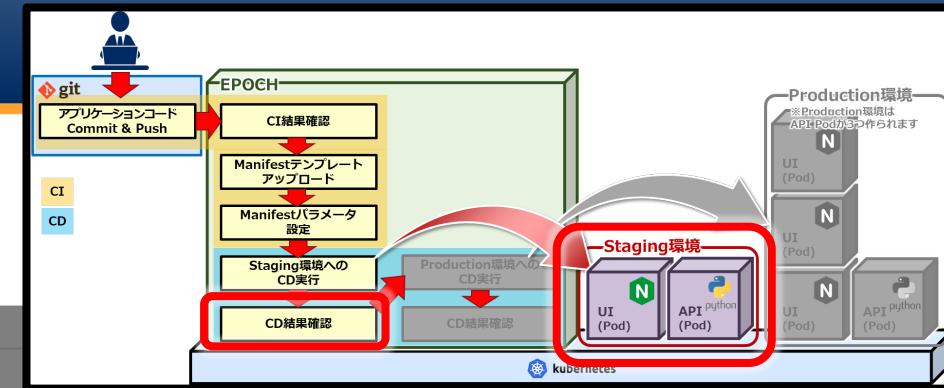
The dialog box is titled 'CD実行指定' (CD Execution Specification). It has sections for '実行条件' (Execution Conditions) and 'Manifest/パラメータ' (Manifest/Parameters). In the '実行条件' section, '即実行' (Run Now) is selected, and the '環境' (Environment) dropdown is set to 'staging'. A yellow callout box says 'stagingを選択します' (Select staging). In the 'Manifest/パラメータ' section, the environment is set to 'staging', and the manifest repository is 'https://github.com/epoch-team/argocd_manifest.git'. A red box highlights the '実行' (Execute) button at the bottom, with a yellow callout box saying '環境選択後、実行ボタンを押下します' (Press the execute button after selecting the environment).

Staging環境へのCD実行が完了しました
CD実行結果を確認してみましょう

4.5 1回目のCI/CDワークフロー手順(14/24)

Staging環境のCD結果確認

- CDの実行結果は、Exastro IT-Automation、ArgoCDより確認できます。



4.5 1回目のCI/CDワークフロー手順(15/24)

Manifestファイルの生成確認(Staging環境)

- Exastro IT-Automationから、IaCリポジトリへManifestファイルを登録するまでの状況を確認します。

The screenshot shows the Exastro IT Automation interface. On the left, the 'EPOCHクイックスタート' dashboard displays various CI/CD components like Application Code, TEKTON Pipelines, and Argo CD. A red box highlights the 'Conductor' icon in the bottom right corner of the dashboard area. A yellow callout box [1] points to this icon with the text '[1] Conductorメニュー > Conductor作業一覧を選択します'. In the center, a 'Conductor' task list window is open. A red box highlights the 'Conductor作業一覧' button in the menu. A yellow callout box [2] points to the 'フィルタ' button with the text '[2] フィルタをクリック'. At the bottom, a red box highlights the '詳細' button next to a 'CD実行' row. A yellow callout box [3] points to this button with the text '[3] [CD実行] の詳細をクリック'. On the right, a 'Conductor' task list window shows a workflow step labeled '[1]: (サブコンダクター)コミット/プッシュ' with a green 'DONE' status. A yellow callout box [4] points to this step with the text 'すべて[DONE]と表示されていれば完了です'. Above the central window, a 'Login' dialog box is shown with a red box highlighting the 'ログインID' and 'パスワード' fields. A yellow callout box [5] points to these fields with the text 'ログインID: administrator
パスワード: fmSCuPLsbvG7KCMRrtT9
でログインします'.

4.5 1回目のCI/CDワークフロー手順(16/24)

パイプラインArgoCDの結果確認(Staging環境)

- Manifestがkubernetesに反映されるまでの状況を確認します。

stagingを選択

Deployされたサンプルアプリケーションを確認してみましょう

Username: admin
Password: iYSCKzx2wvxJnn4dCNwN
でSIGN INします

※「この接続ではプライバシーが保護されません」が表示されますが、詳細表示から「アクセスする」を選択してログイン画面に遷移します

3分ごとに同期されますので同期されることを待ちます

【Sync OK】が表示されましたら完了です
※日付がCD実行後であることを確認してください

4.5 1回目のCI/CDワークフロー手順(17/24)

Staging環境のサンプルアプリケーション確認

- 次のURLでデプロイしたサンプルアプリケーションを表示します

[http://\[Kubernetes masterノードのIPアドレスまたはホスト名\]:31001/front-end.html](http://[Kubernetes masterノードのIPアドレスまたはホスト名]:31001/front-end.html)

The screenshot shows a product catalog page with a dropdown menu for currency selection. The menu is open, showing 'YEN' (selected), 'YEN', and 'USD'. A red box highlights this dropdown. To the right, a callout box contains the following text:

表示価格の選択欄に「YEN」「USD」が表示されていることを確認します。
確認後、「USD」を選択して、表示が切り替わることを確認します。

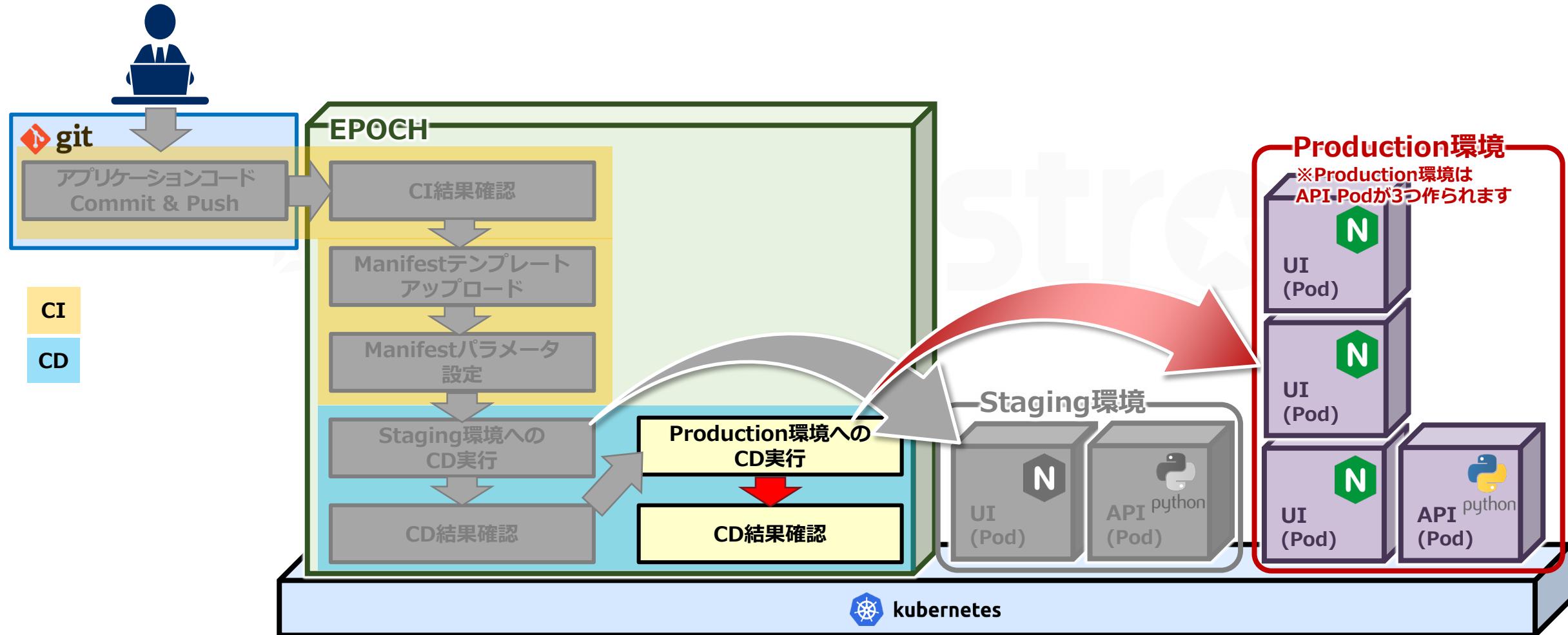
The catalog displays seven shirts (TシャツA-G) with their prices in yen. Shirts A, C, D, E, and G have a price of 1,000 yen (税込). Shirts B and F have a price of 2,000 yen (税込). Shirt G has a price of 6,500 yen (税込). Each shirt has a '詳細はこちら' (Details) button and a '購入サイトはこちら' (Purchase site) button.

**続いてProduction環境へ
Deployしてみましょう**

4.5 1回目のCI/CDワークフロー手順(18/24)

Production環境へのDeploy

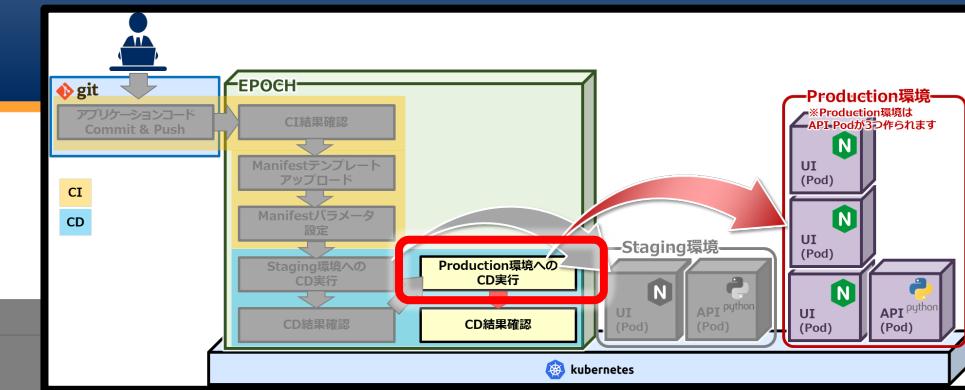
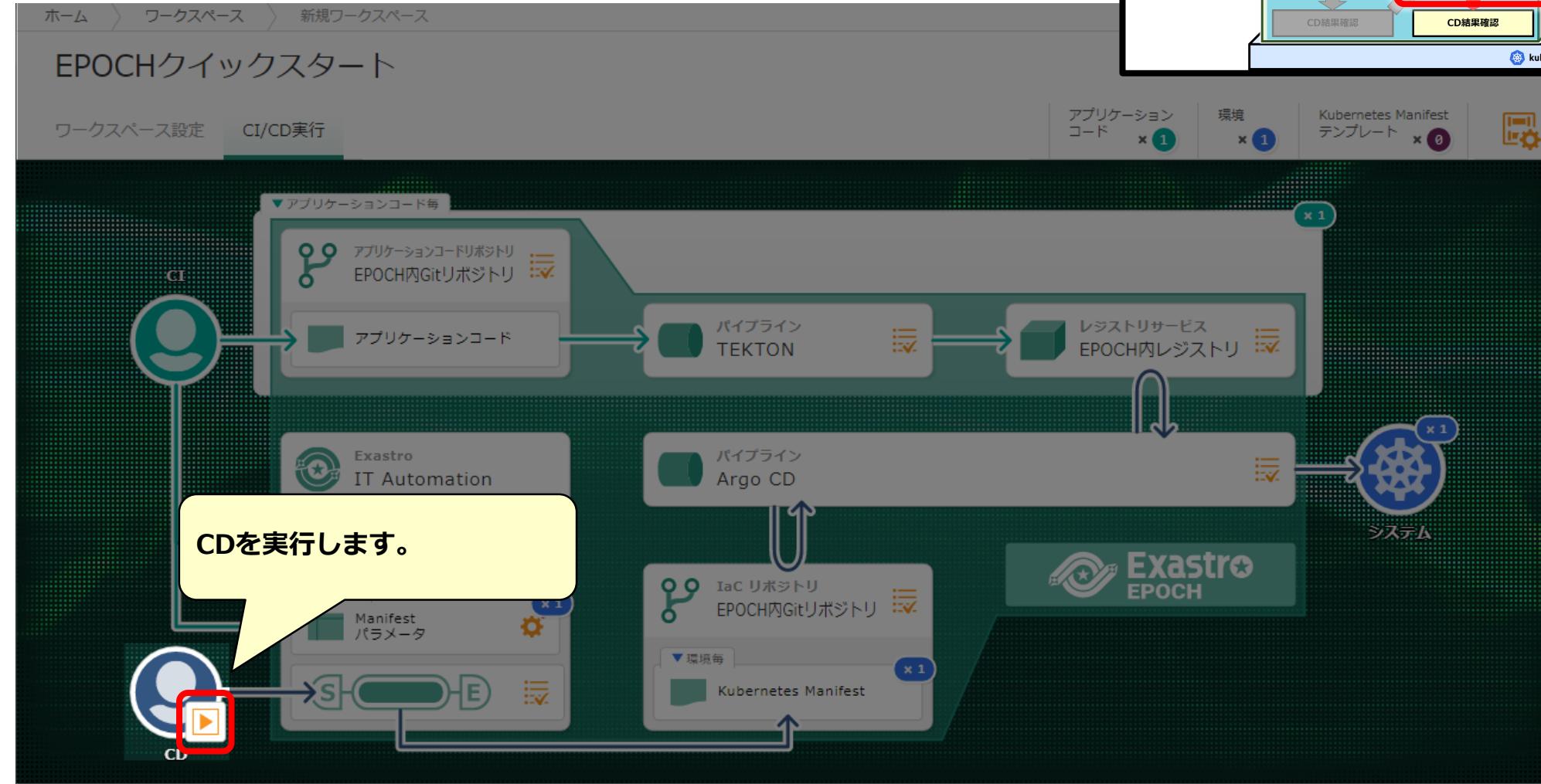
Staging環境へDeploy後、以下の内容でProduction環境へDeployします。



4.5 1回目のCI/CDワークフロー手順(19/24)

Production環境へのDeploy実行

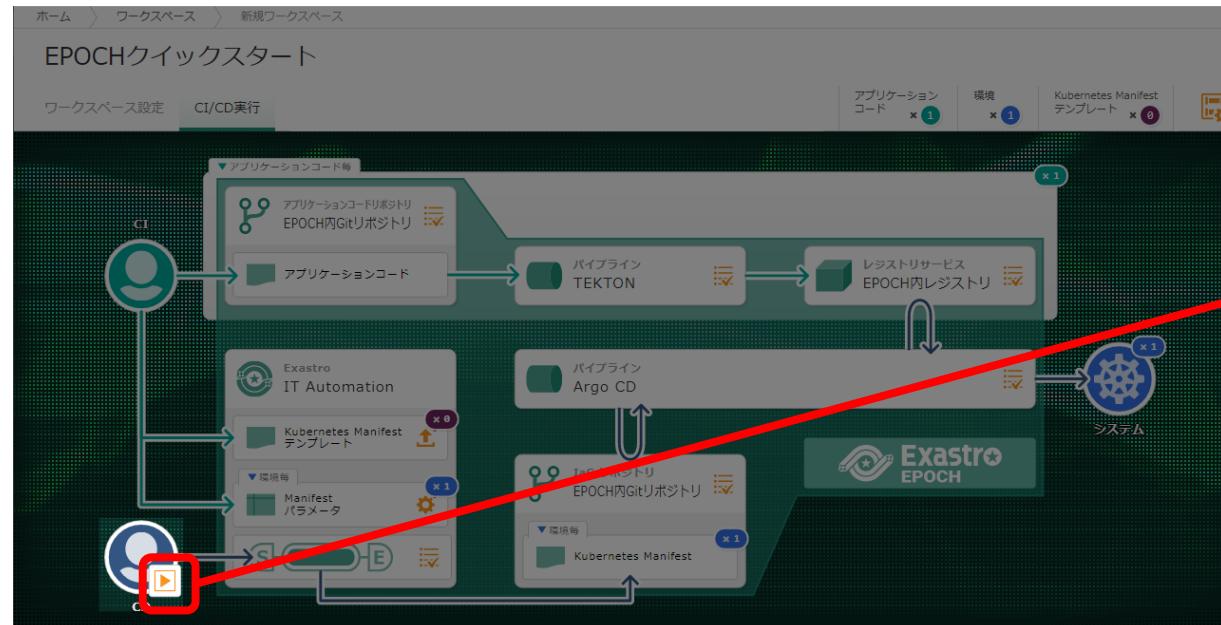
- CD実行で、Production環境へDeployを実行します。



4.5 1回目のCI/CDワークフロー手順(20/24)

Production環境のCD実行

- Deploy先の環境を選択して実行します。



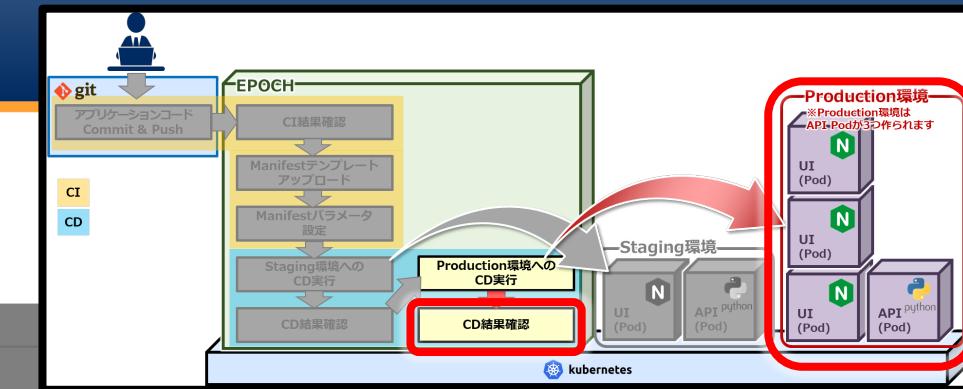
The screenshot shows the 'CD Execution Specification' dialog box. It has sections for 'Execution Conditions' (选择条件), 'Manifest Parameters' (Manifestパラメータ), and 'ArgoCD Pipeline' (ArgoCDパイプライン). A red box highlights the 'Environment' dropdown where 'production' is selected. A callout bubble says 'productionを選択します' (Select production). Another callout bubble at the bottom right says '環境選択後、実行ボタンを押下します' (After selecting the environment, press the execute button).

Production環境へのCD実行が完了しました
CD実行結果を確認してみましょう

4.5 1回目のCI/CDワークフロー手順(21/24)

Production環境のCD結果確認

- CDの実行結果は、Exastro IT-Automation、ArgoCDより確認できます。



4.5 1回目のCI/CDワークフロー手順(22/24)

Manifestファイルの生成確認(Production環境)

- Exastro IT-Automationから、IaCリポジトリへManifestファイルを登録するまでの状況を確認します。

The screenshot shows the Exastro IT Automation interface. On the left, the EPOCH Quick Start dashboard displays various CI/CD components like Application Code, Pipeline TEKTON, Pipeline Argo CD, and Kubernetes Manifest. A red box highlights the 'Conductor' icon in the bottom right corner of the dashboard. A red arrow points from this icon to the 'Conductor' task list on the right.

[1] Conductorメニュー > Conductor作業一覧を選択します

[2] フィルタをクリック

[3] 【CD実行】の詳細をクリック

ログインID: administrator
パスワード: fmSCuPLsbvG7KCMRrtT9
でログインします

すべて[DONE]と表示されていれば完了です

Detailed description: The image is a composite of several screenshots from the Exastro IT Automation platform. 1. Top-left: EPOCH Quick Start dashboard showing a CI/CD pipeline with various stages and tools like TEKTON, Argo CD, and Kubernetes Manifest. 2. Top-right: A 'Login' dialog box with fields for 'Login ID' and 'Password', and a 'Login' button. A yellow callout box says 'ログインID: administrator' and 'パスワード: fmSCuPLsbvG7KCMRrtT9 でログインします'. 3. Middle-left: 'Conductor' menu with options like 'Conductorインターフェース情報', 'Conductorクラス一覧', etc. A red box highlights the 'Conductor作業一覧' (Conductor Task List) option. 4. Middle-center: 'Conductor Task List' screen with a table of tasks. A red box highlights the 'フィルタ' (Filter) button. 5. Middle-right: 'Conductor Task List' screen showing a single task named 'Conductor-call' with status 'DONE'. 6. Bottom-right: A large yellow callout box says 'すべて[DONE]と表示されていれば完了です'.

4.5 1回目のCI/CDワークフロー手順(23/24)

パイプラインArgoCDの結果確認(Production環境)

- Manifestがkubernetesに反映されるまでの状況を確認します。

The diagram illustrates the CI/CD workflow and deployment status across three main stages:

- CI (Left):** Shows the flow from Application Code to Pipeline TEKTON, which then interacts with the Registry Service EPOCH内レジストリ and the Argo CD Pipeline.
- CD (Bottom Left):** Shows the Applications screen where the "production" environment is selected. It displays two entries: "production" (Synced, Healthy) and "staging" (Out of Sync, Missing). A red box highlights the "production" entry.
- Deployment Status (Bottom Right):** Shows the "staging" environment details, indicating it is healthy and Synced. A red box highlights the "Sync OK" status message.
- Argo Login (Top Right):** A login form for "Username: admin" and "Password: iYSCKzx2wvxJnn4dCNwN". A callout notes that users should select "Access" instead of "Protect" for privacy protection.
- Final Callout (Bottom Center):** A large red box contains the text: "Deployされたサンプルアプリケーションを確認してみましょう" (Let's check the deployed sample application).

4.5 1回目のCI/CDワークフロー手順(24/24)

Production環境のサンプルアプリケーション確認

- 次のURLでデプロイしたサンプルアプリケーションを表示します

[http://\[Kubernetes masterノードのIPアドレスまたはホスト名\]:31003/front-end.html](http://[Kubernetes masterノードのIPアドレスまたはホスト名]:31003/front-end.html)

商品カタログ

表示価格 YEN
YEN
USD

Tシャツ	価格	操作
A	1,000 円(税込)	詳細はこちら
B	2,000 円(税込)	詳細はこちら
C	7,500 円(税込)	詳細はこちら
D	4,700 円(税込)	詳細はこちら
E	1,200 円(税込)	詳細はこちら

Staging環境と同様に画面が表示されることを確認します

1回目のCI/CDワークフローはここで完了です
続いて実際にアプリケーションコードを修正し
Deployされるまでの2回目のCI/CDワークフロー手順
を実行してみましょう！

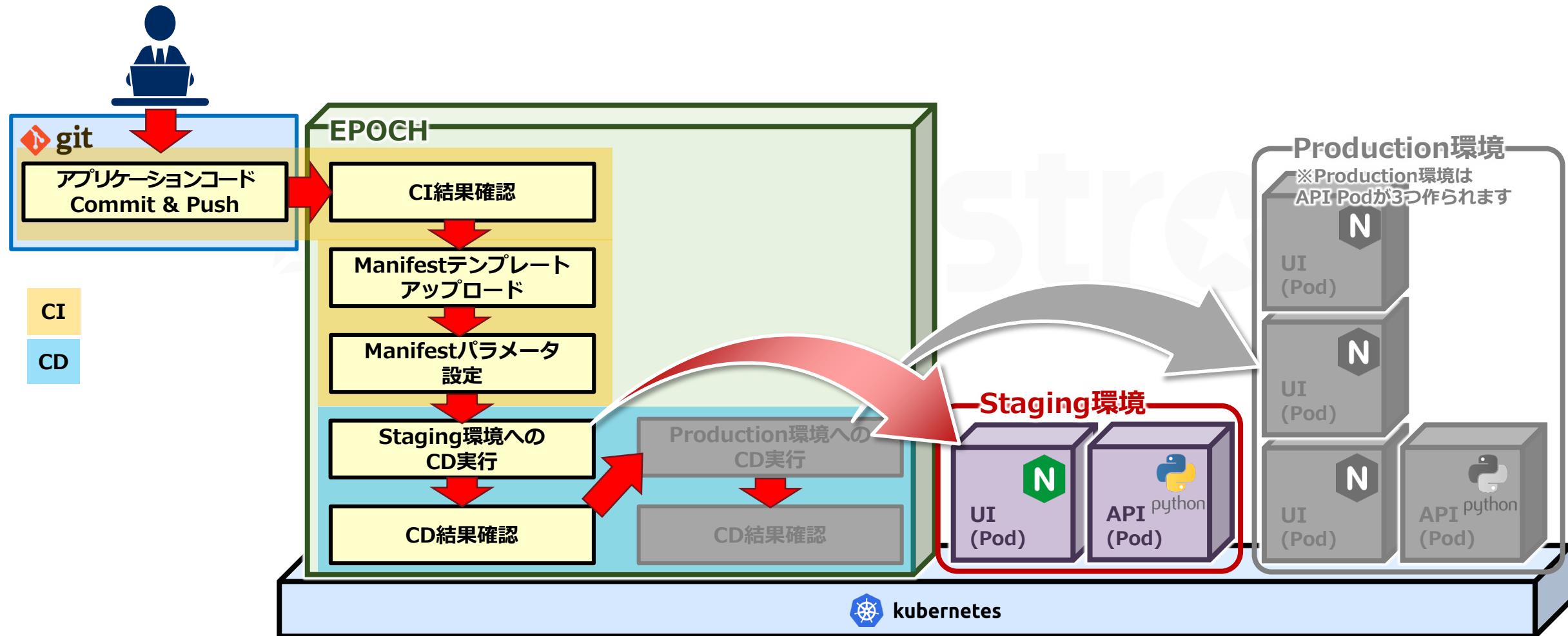
2回目のCI/CDワークフロー手順



4.6 2回目のCI/CDワークフロー手順(1/21)

Staging環境へのDeploy

2回目のCI/CDワークフロー手順として、アプリケーションコードを修正してCommit & PushからStaging環境へのDeploy、CD結果確認までの手順を説明します。



4.6 2回目のCI/CDワークフロー手順(2/21)

■ アプリケーションコードの修正

アプリケーションコードを修正して、2回目のCI/CDを実行していきます。
チュートリアルでは、画面に通貨（ユーロ）の表示追加を行います。

PC環境にcloneしたアプリケーションコード用リポジトリの、以下のファイルをコードエディタで修正します。

- 修正対象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 }
```

} 9~12行目を追加

- 修正対象② : api-app/data/rate.json

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

} 2行目の末尾にカンマを追加
} 3行目を追加

4.6 2回目のCI/CDワークフロー手順(3/21)

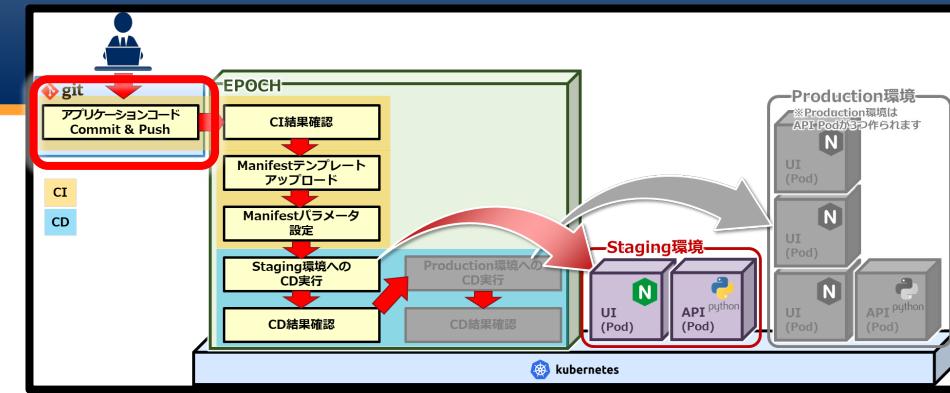
アプリケーションコード Commit & Push

修正した内容をCommit&Pushします。

コマンドプロンプトで、以下の様に実行します。

```
> cd "[clone先のフォルダ]"
> cd epoch-sample-app
> git add .
> git commit -m "通貨追加(EUR)"
> git push origin master
```

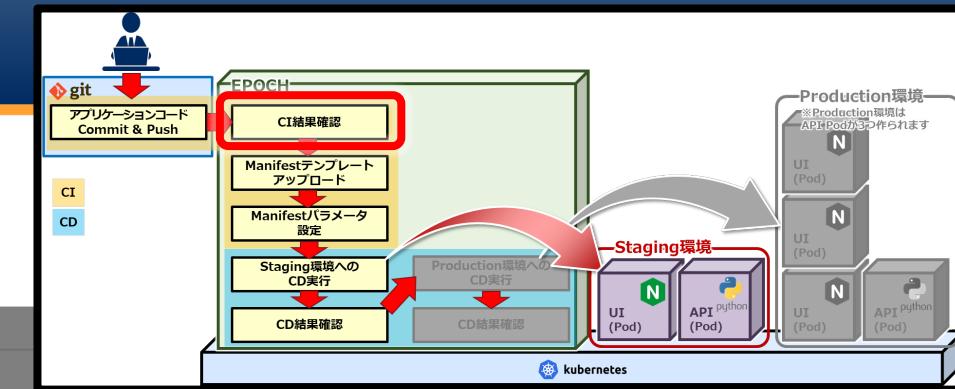
```
Username for 'https://github.com': [自身のGitHubユーザ名を入力]
Password for 'https://xxxxxx@github.com': [自身のGitHubパスワードを入力]
```



4.6 2回目のCI/CDワークフロー手順(4/21)

CI結果確認

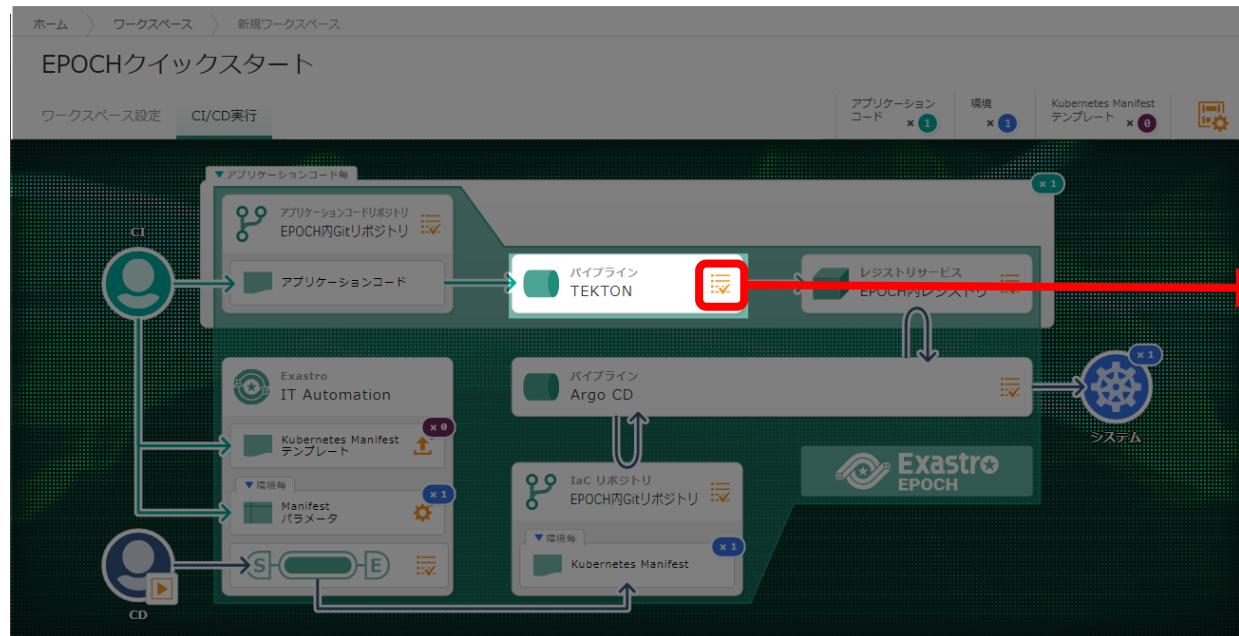
- アプリケーションコードのビルド結果を確認します。



4.6 2回目のCI/CDワークフロー手順(5/21)

■ パイプラインTEKTONの結果確認

- TEKTONのパイプラインを実際に確認し、ビルドが正常に終了したか確認します。



The Tekton Dashboard shows the 'PipelineRuns' table. It lists a single run: 'build-and-push-pipeline-run...' with status 'Succeeded'. The table includes columns for Status, Name, Pipeline, Namespace, Created, and Duration. A callout box on the right states: 'CIパイプラインの動作は、TEKTONダッシュボードにて確認することができます。Nameをクリックすることにより、より詳細なCIパイプラインの内容を確認することができます。' (The CI pipeline's operation can be confirmed on the Tekton Dashboard. Clicking on the Name will allow you to check the detailed content of the CI pipeline.)

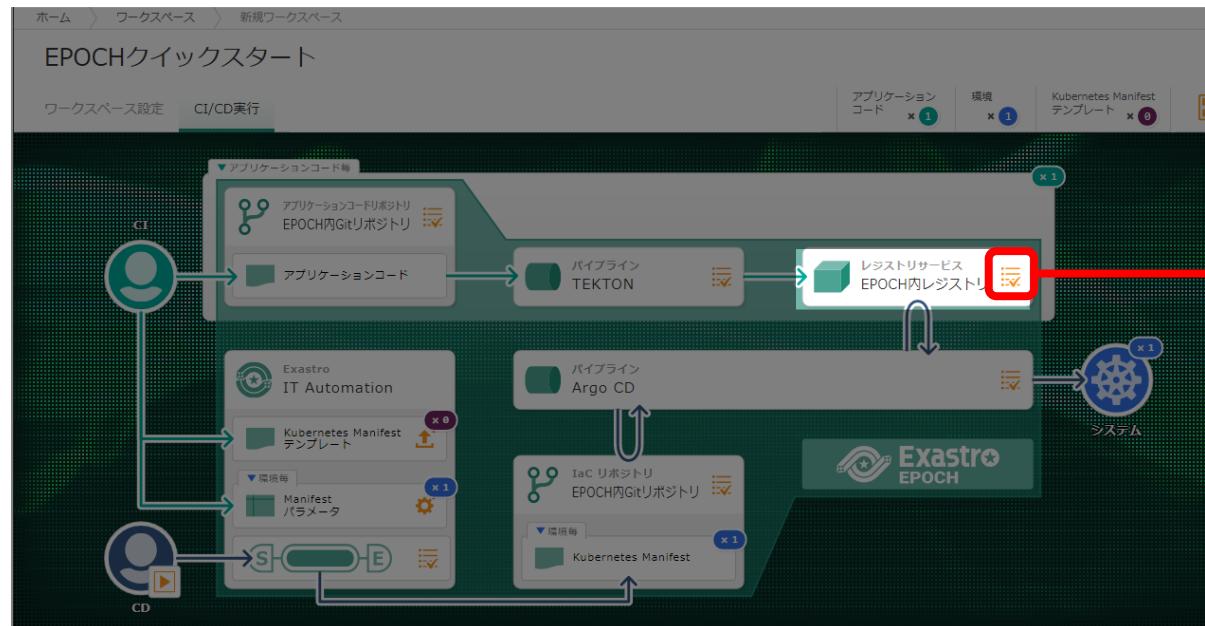
Status	Name	Pipeline	Namespace	Created	Duration
	build-and-push-pipeline-run...	pipeline-build-and-push	epoch-tekton-pipelines	6 minutes ago	3 minutes 32 seconds

Status欄にチェックマークが表示されていれば正常終了となります。

4.6 2回目のCI/CDワークフロー手順(6/21)

コンテナイメージのタグ名の確認

- レジストリサービスの画面を開き、ビルトしたコンテナイメージのTagを確認します。



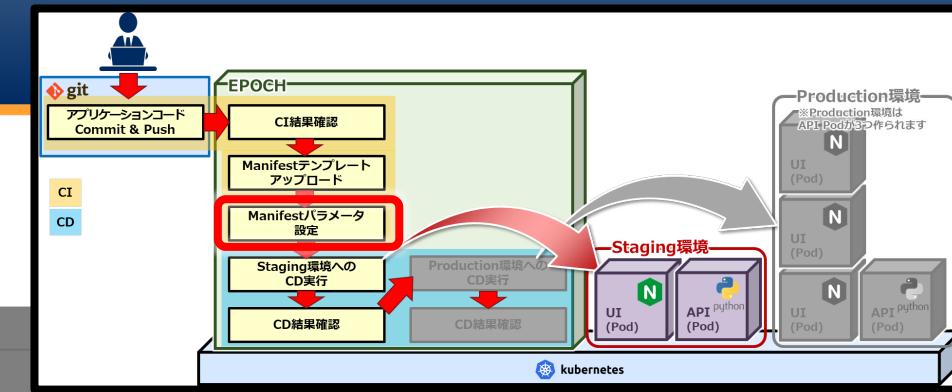
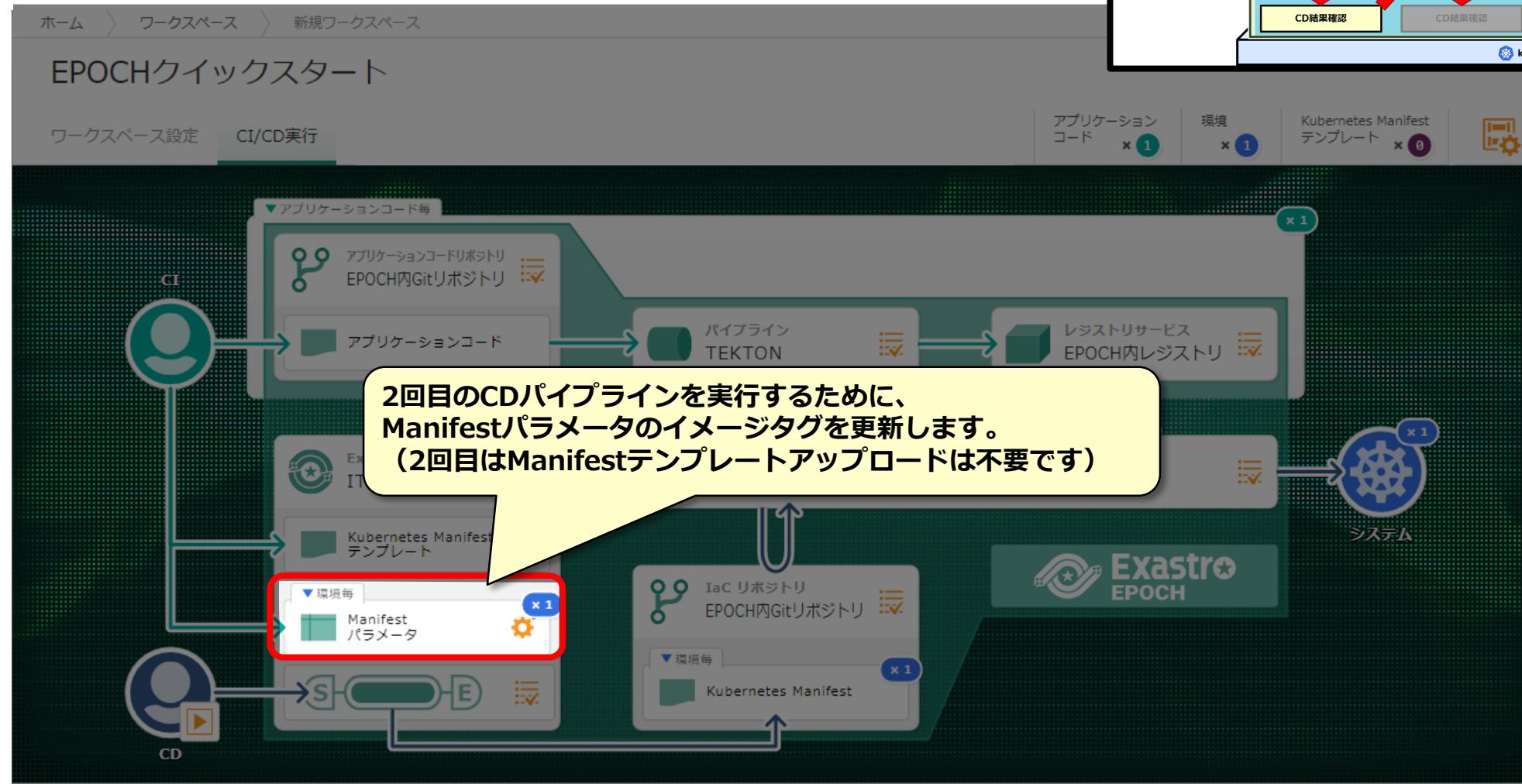
The screenshot shows the Docker Hub page for the repository `/epoch-sample-api`. It displays two tags: `master.20210701-171058` and `master.20210701-134114`. A yellow callout box points to the first tag with the text: "Dockerhubの画面でepoch-sample-apiのTagを確認します" (Check the tag for epoch-sample-api on the Dockerhub page). Another text box below it says: "※イメージが登録されていないときは、パイプラインTEKTONから結果を確認してください" (If the image is not registered, check the results from the TEKTON pipeline).

ここで確認したイメージのTagは、
次の手順で、Manifestパラメータ(image_tag)に
手入力が必要になるため控えておいてください

4.6 2回目のCI/CDワークフロー手順(7/21)

Manifestパラメータ設定

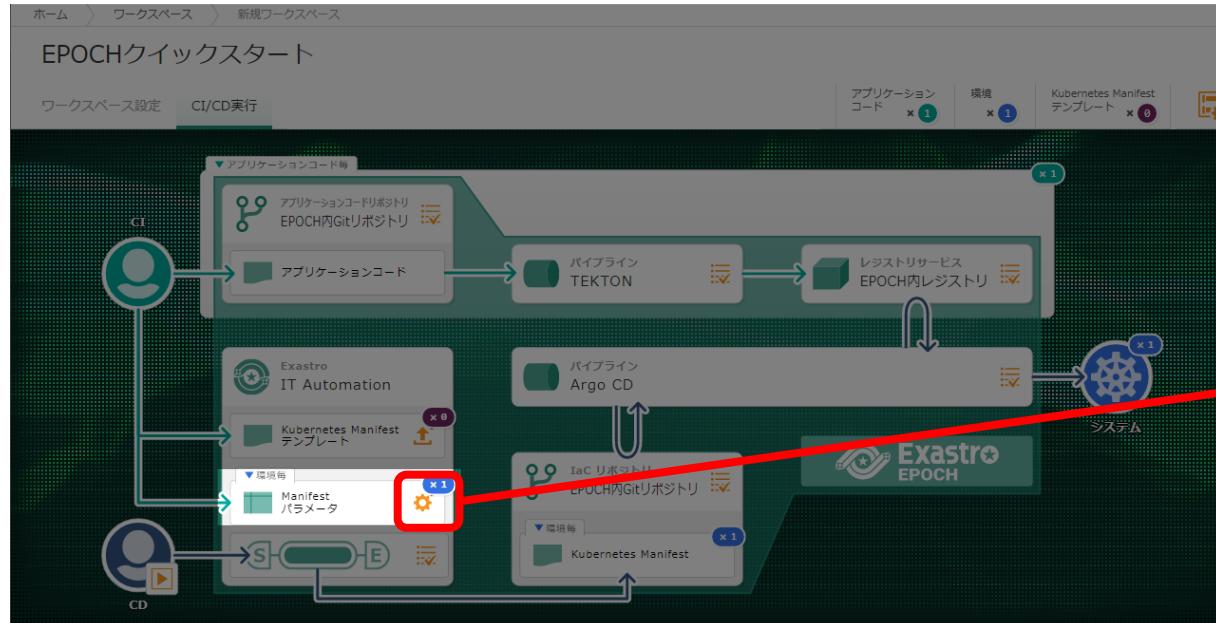
- Manifestパラメータの設定のイメージタグを更新します。



4.6 2回目のCI/CDワークフロー手順(8/21)

Manifestパラメータ(image_tag)の修正

- Manifestパラメータで、staging, production環境のイメージのタグ名を修正します。



Manifest パラメータ

api-app.yaml

パラメータ	staging	production
replicas	staging : replicas	production : replicas
image	staging : image	production : image
image_tag	staging : image_tag	production : image_tag
port_no_active	staging : port_no_active	production : port_no_active
port_no_preview	staging : port_no_preview	production : port_no_preview

staging, productionの image_tagを修正

api-app.yamlを選択

image_tagを選択

```
apiVersion: argoproj.io/v1alpha1
kind: Rollout
metadata:
  name: api-app
spec:
  selector:
    matchLabels:
      name: api-app
  replicas: {{ replicas }}
  template:
    metadata:
      labels:
        name: api-app
    spec:
      containers:
        - name: api-app
          image: {{ image }}:{{ image_tag }}
          ports:
            - name: http
              containerPort: 8000
              protocol: TCP
          env:
            - name: FLASK_ENV
              value: "development"
            - name: API_PORT
              value: "8000"
            - name: PYTHONIOENCODING
              value: utf-8
      strategy:
        blueGreen:
          activeService: api-app-bluegreen-active
          previewService: api-app-bluegreen-preview
          autoPromotionEnabled: true
          scaleDownDelaySeconds: 120
```

決定 キャンセル

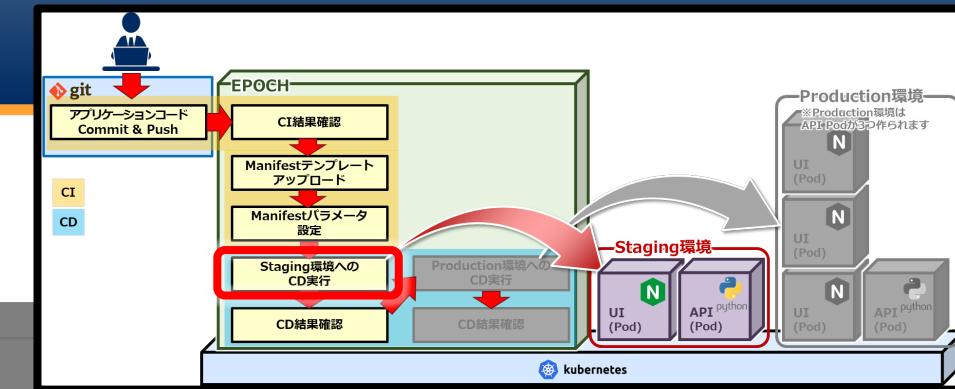
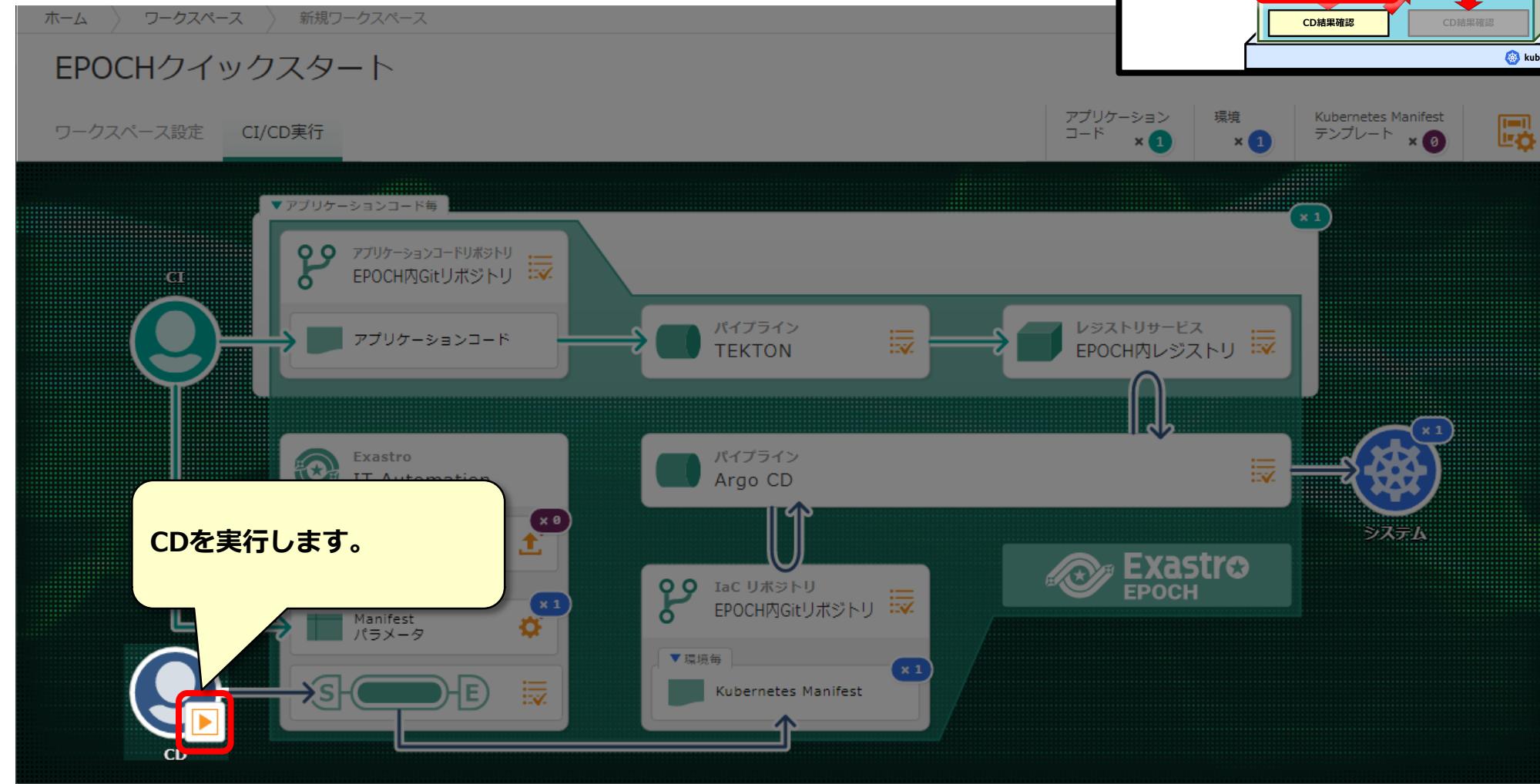
api-app.yaml

項目	入力内容(staging)	入力内容(production)	説明
{{ image_tag }}	[レジストリサービスで確認した最新のイメージのタグ名]	[レジストリサービスで確認した最新のイメージのタグ名]	コンテナイメージのタグ

4.6 2回目のCI/CDワークフロー手順(9/21)

Staging環境へのCD実行

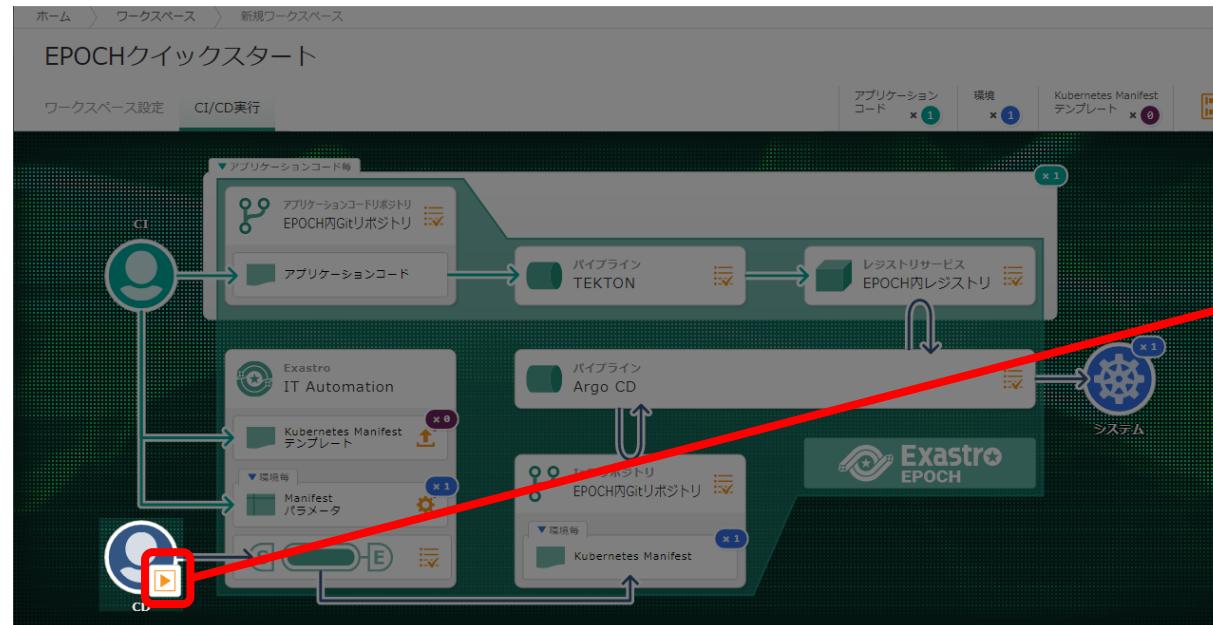
- CD実行で、実際にStaging環境へDeployします。



4.6 2回目のCI/CDワークフロー手順(10/21)

CD実行指定

- Deploy先の環境を選択してDeployを実行します。



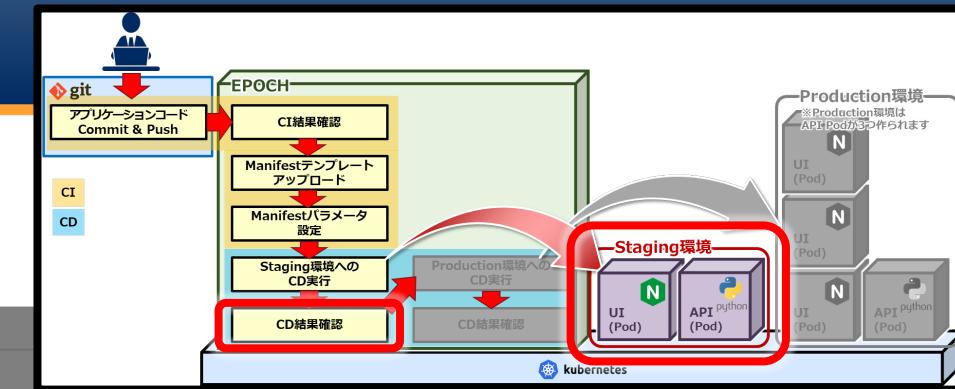
The dialog box is titled 'CD実行指定' (CD Execution Specification). Under '実行条件' (Execution Conditions), '即実行' (Run Now) is selected. The '環境' (Environment) dropdown is set to 'staging', highlighted with a red box. A yellow callout bubble says 'stagingを選択します' (Select staging). In the 'Manifest/パラメータ' (Manifest/Parameters) section, the environment is set to 'staging'. The 'ArgoCDパイプライン' (ArgoCD Pipeline) section lists the manifest repository as 'https://github.com/epoch-team/argocd_manifest.git', Kubernetes API Server URL as 'https://kubernetes.default.svc', and Namespace as 'staging-app'. A red box highlights the '実行' (Execute) button at the bottom, with a yellow callout bubble saying '環境選択後、実行ボタンを押下します' (Press the execute button after selecting the environment).

Staging環境へのCD実行が完了しました
CD実行結果を確認してみましょう

4.6 2回目のCI/CDワークフロー手順(11/21)

Staging環境のCD結果確認

- CDの実行結果は、Exastro IT-Automation、ArgoCDより確認できます。



4.6 2回目のCI/CDワークフロー手順(12/21)

Manifestファイルの生成確認(Staging環境)

- Exastro IT-Automationから、IaCリポジトリへManifestファイルを登録するまでの状況を確認します。

The screenshot illustrates the workflow for generating and confirming Manifest files in the Staging environment. It shows the Exastro IT Automation interface with various components like EPOCH Quick Start, Application Code, Environment, Pipeline TEKTON, Pipeline Argo CD, and Kubernetes Manifest. A red box highlights the 'Conductor' icon in the bottom navigation bar.

[1] Conductorメニュー > Conductor作業一覧を選択します

[2] フィルタをクリック

[3] 【CD実行】の詳細をクリック

**ログインID: administrator
パスワード: fmSCuPLsbvG7KCMRrtT9
でログインします**

すべて[DONE]と表示されていれば完了です

4.6 2回目のCI/CDワークフロー手順(13/21)

パイプラインArgoCDの結果確認(Staging環境)

- Manifestがkubernetesに反映されるまでの状況を確認します。

The screenshot shows the EPOCH UI interface. On the left, the 'EPOCHクイックスタート' (Quick Start) panel displays the CI/CD pipeline flow: Application Code → Pipeline TEKTON → Registry Service EPOCH内レジストリ. A red arrow points from this panel to the central 'Let's get stuff deployed!' screen, which features a cartoon octopus character.

In the center, there is a 'staging' application detail view. A yellow callout box says 'stagingを選択' (Select staging). Another red box highlights the 'staging' section in the list. A red arrow points from this section to the right-hand 'Sync OK' status message.

On the right, a 'Sign In' dialog box is shown with a yellow callout: 'Username: admin' and 'Password: iYSCKzx2wvxJnn4dCNwN'. Below it, a note says: '※「この接続ではプライバシーが保護されません」が表示されますが、詳細表示から「アクセスする」を選択してログイン画面に遷移します' (Note: 'This connection does not protect privacy' is displayed, but select 'Access' in the detailed view to navigate to the login screen).

Further down on the right, another yellow callout says: '3分ごとに同期されますので同期されることを待ちます。' (Sync occurs every 3 minutes, so wait for synchronization).

At the bottom, a large red box contains the text: 'Deployされたサンプルアプリケーションを確認してみましょう' (Let's check the deployed sample application).

The 'Sync OK' message details the sync status:

- From: 'To Staging' (Synced 1 minute ago)
- To: 'To Staging' (Synced 1 minute ago)
- Authored by epoch-team <epoch-team@nec.com>
- Succeeded a minute ago (17:18:26 GMT+0900)
- Authored by epoch-team <epoch-team@nec.com>
- Manifest Push_202107081712

The 'Sync OK' section also lists several Kubernetes resources:

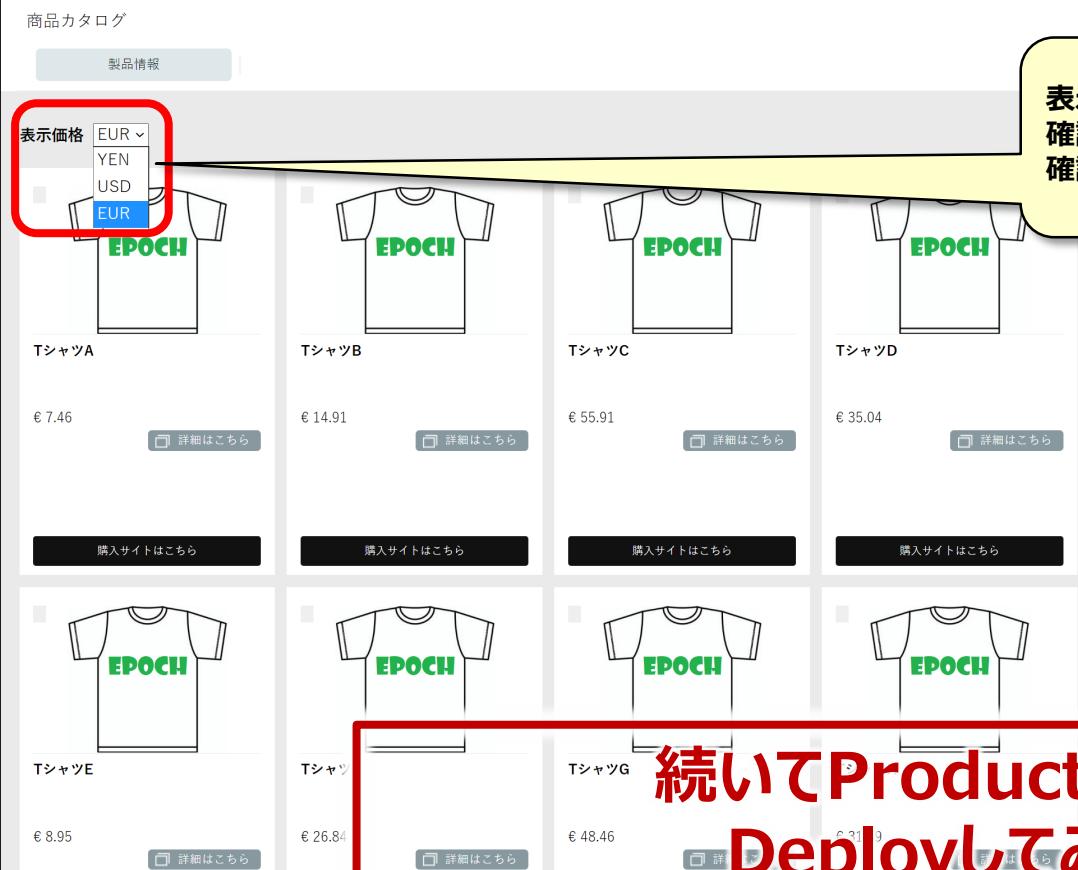
- ui-app-com (cm)
- api-app-bluegreen (svc)
- api-app-bluegreen (svc)
- ui-app-bluegreen-active (svc)
- ui-app-bluegreen-preview (svc)
- api-app-78c75b9b78-hlhk4 (pod)

4.6 2回目のCI/CDワークフロー手順(14/21)

Staging環境のアプリケーションの確認

- ブラウザで以下のURLに接続し、デプロイしたサンプルアプリケーションを表示します

[http://\[Kubernetes masterノードのIPアドレスまたはホスト名\]:31001/front-end.html](http://[Kubernetes masterノードのIPアドレスまたはホスト名]:31001/front-end.html)



表示価格の選択欄に「YEN」「USD」「EUR」が表示されていることを確認します。
確認後、「EUR」を選択して、表示が切り替わることを確認します。

商品カタログ

表示価格 EUR
YEN
USD
EUR

TシャツA TシャツB TシャツC TシャツD

€ 7.46 詳細はこちら 購入サイトはこちら

TシャツE TシャツF TシャツG TシャツH

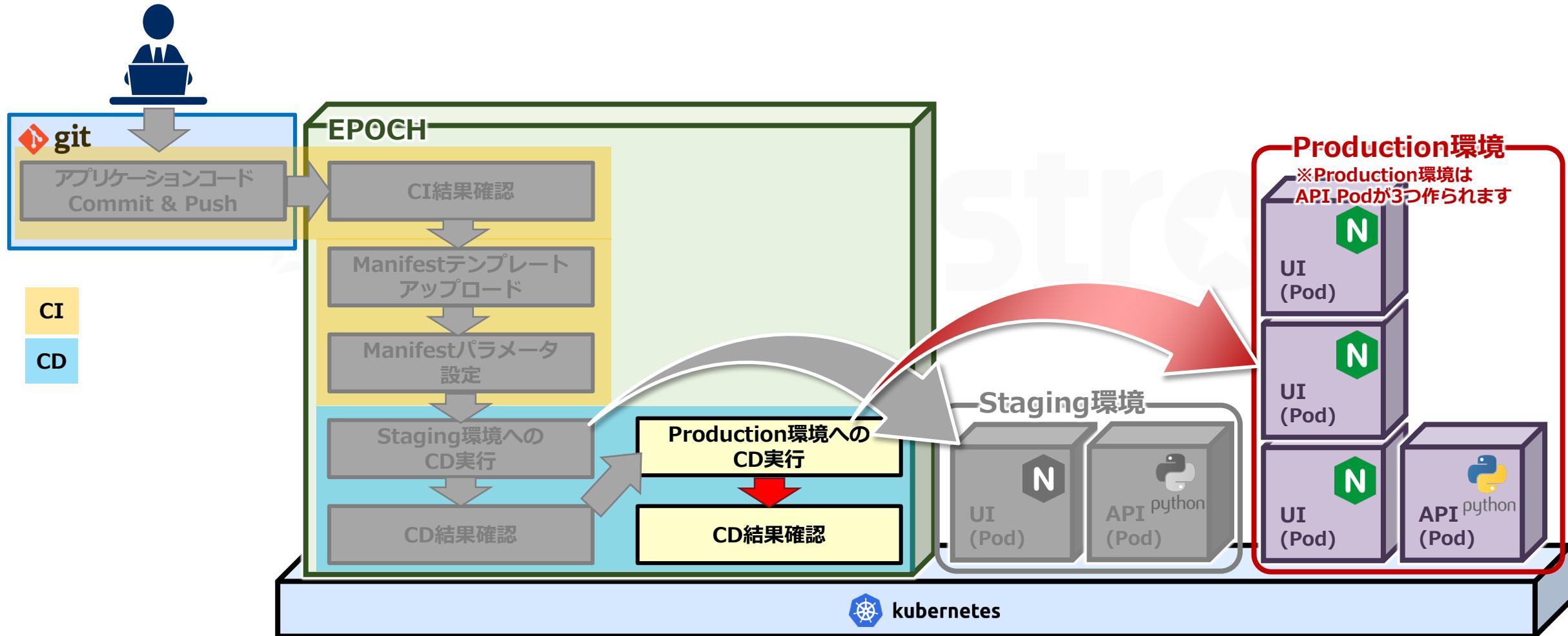
€ 8.95 € 26.84 € 48.46 € 31.99 詳細はこちら 購入サイトはこちら

続いてProduction環境へ Deployしてみましょう

4.6 2回目のCI/CDワークフローハンドル(15/21)

Production環境へのDeploy

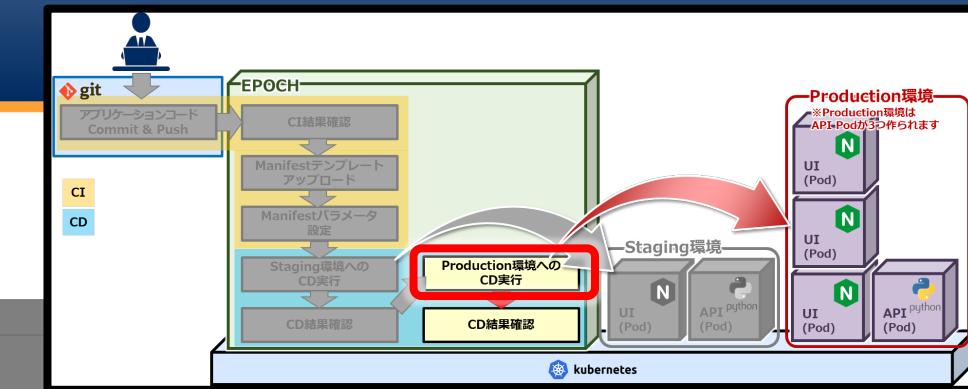
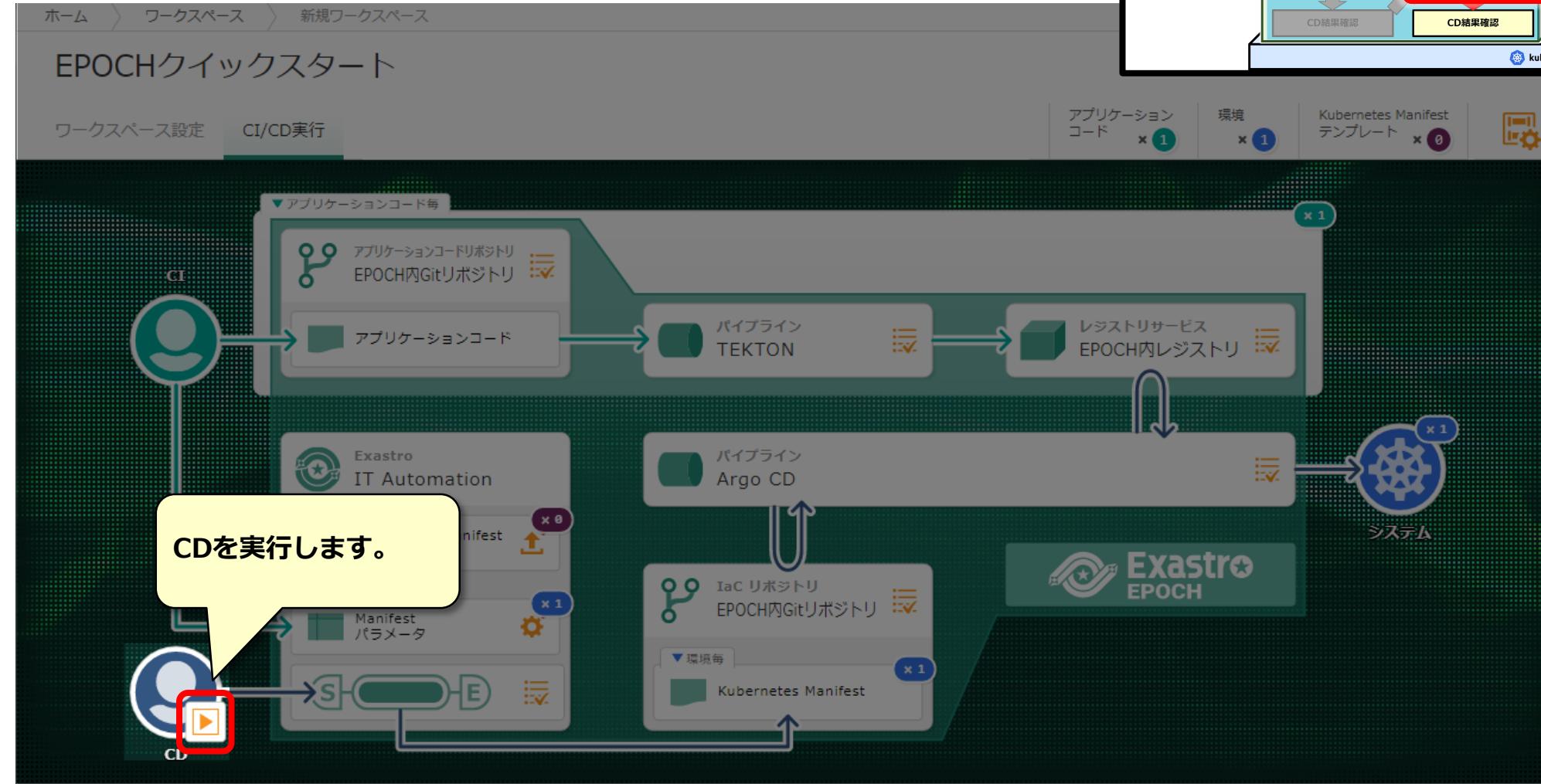
Staging環境へDeploy後、Production環境へDeployする流れを説明します。



4.6 2回目のCI/CDワークフロー手順(16/21)

Production環境へのCD実行

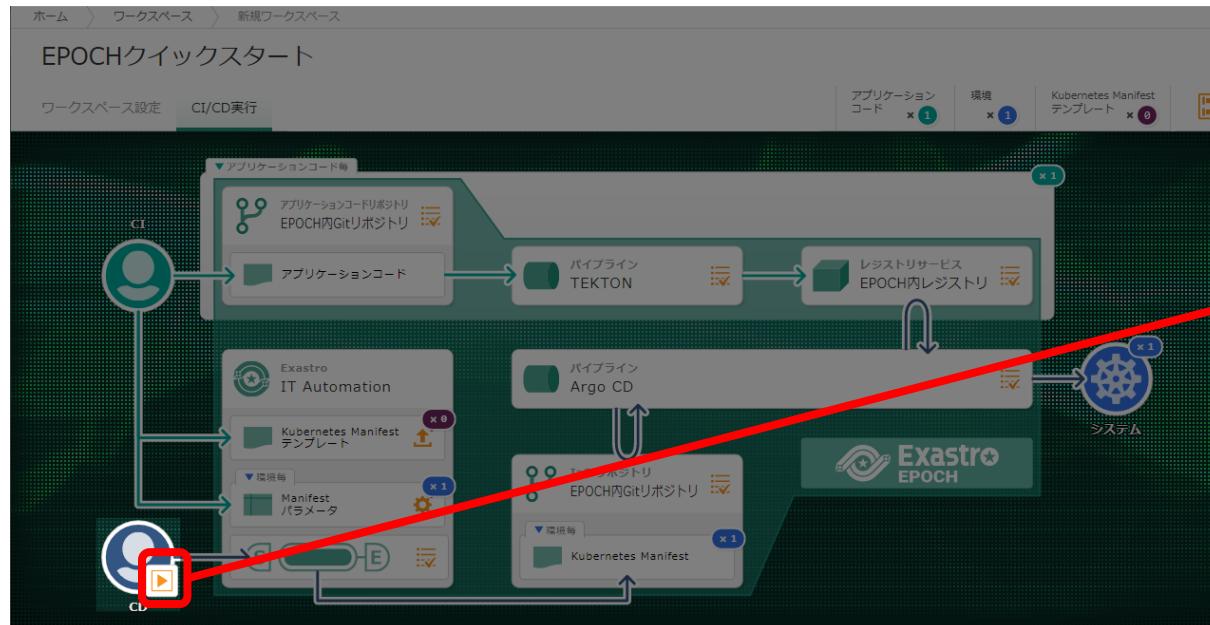
- CD実行で、実際にProduction環境へDeployを実行します。



4.6 2回目のCI/CDワークフロー手順(17/21)

CD実行指定

- Deploy先の環境を選択してDeployを実行します。



The dialog box contains the following fields:

- 実行条件:
 - CD実行日時: 即実行 (selected)
 - 予約日時指定: 2021/07/08 14:17
- 環境: staging (highlighted with a red box)
- Manifest/パラメータ
- ArgoCDパイプライン
- 以下的内容でDeployします。よろしいですか?
 - 環境名: staging
 - Manifestリポジトリ: https://github.com/epoch-team/argocd_manifest.git
 - Kubernetes API Server URL: https://kubernetes.default.svc
 - Namespace: staging-app
- 実行 (button highlighted with a red box)

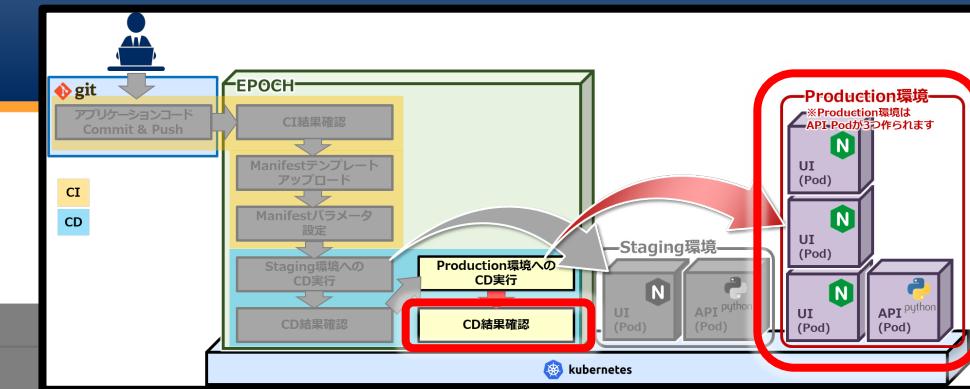
A callout box says "productionを選択します" (Select production) pointing to the environment dropdown. Another callout box says "環境選択後、実行ボタンを押下します" (Press the execute button after selecting the environment) pointing to the '実行' button.

Production環境へのCD実行が完了しました
CD実行結果を確認してみましょう

4.6 2回目のCI/CDワークフロー手順(18/21)

Production環境のCD結果確認

- CDの実行結果は、Exastro IT-Automation、ArgoCDより確認できます。



4.6 2回目のCI/CDワークフロー手順(19/21)

Manifestファイルの生成確認(Production環境)

- Exastro IT-Automationから、IaCリポジトリへManifestファイルを登録するまでの状況を確認します。

The screenshot shows the Exastro IT Automation interface. On the left, the 'EPOCHクイックスタート' dashboard displays a CI/CD pipeline with various stages like Application Code, Pipeline TEKTON, and Registry Service. A red box highlights the 'Conductor' icon in the bottom right corner of the pipeline diagram. On the right, the 'Conductor' interface is shown in three panels:

- Top Panel (Login):** Shows a login form with fields for 'ログインID' and 'パスワード'. A yellow callout notes: 'ログインID: administrator パスワード: fmSCuPLsbvG7KCMRrtT9 でログインします'.
- Middle Panel (Conductor Task List):** Shows a list of tasks under 'Conductor作業一覧'. A red box highlights the 'フィルタ' button. A yellow callout notes: '[1]Conductorメニュー > Conductor作業一覧を選択します'.
- Bottom Panel (Task Details):** Shows a detailed view of a task labeled 'Conductor-call [1]: (サブコンダクター)コミット/プッシュ'. The status is 'DONE'. A yellow callout notes: '[2]フィルタをクリック'.
- Bottom Right Callout:** Notes: 'すべて[DONE]と表示されていれば完了です'.
- Bottom Left Callout:** Notes: '[3]【CD実行】の詳細をクリック'.

4.6 2回目のCI/CDワークフロー手順(20/21)

パイプラインArgoCDの結果確認(Production環境)

- Manifestがkubernetesに反映されるまでの状況を確認します。

The diagram illustrates the CI/CD workflow for deploying a sample application. It shows the flow from Application Code to Pipeline TEKTON, then to the EPOCH Registry, and finally to the Argo CD step. A red arrow points from the Argo CD step to the Argo UI login screen. Another red arrow points from the Argo UI to the Kubernetes dashboard, which displays the deployed application's status.

productionを選択

**Username: admin
Password: iYSCKzx2wvxJnn4dCNwN
でSIGN INします**

※「この接続ではプライバシーが保護されません」が表示されますが、詳細表示から「アクセスする」を選択してログイン画面に遷移します

3分ごとに同期されますので同期されることを待ちます。

**【Sync OK】が表示されましたら完了です
※日付がCD実行後であることを確認してください**

Deployされたサンプルアプリケーションを確認してみましょう

4.6 2回目のCI/CDワークフロー手順(21/21)

Production環境のアプリケーションの確認

- ブラウザで以下のURLに接続し、デプロイしたサンプルアプリケーションを表示します

[http://\[Kubernetes masterノードのIPアドレスまたはホスト名\]:31003/front-end.html](http://[Kubernetes masterノードのIPアドレスまたはホスト名]:31003/front-end.html)





5. 付録

本資料中で行った内容の補足をします。

5.1 注意事項・制限事項

以下、現在のExastro EPOCHのバージョンでの制限事項となります。今後のバージョンで変更される可能性があります。

● 制限事項（今後対応する予定）

- アプリケーションコードのリポジトリは、現在1つのみ対応となっております。
- 現在は、アプリケーションコード毎のGitアカウントには対応しておりません。
- Gitサービス選択は次バージョン以降で対応予定です。現在は指定されたURLのGitリポジトリの動作となります。
- ビルドブランチは次バージョン以降で対応予定です。現在はPushされた内容でビルドされます。
- 静的解析は次バージョン以降で対応予定です。現在はSonarQubeを選択した場合に動作しません。
- レジストリサービスは現在内部のレジストリサービスのみとなっております。
- イメージ出力先以外の項目については次バージョン以降で対応予定です。
- Authentication token, Base64 encoded certificateは次バージョン以降で対応予定です。
- テンプレートで指定できる変数は、現在固定です。詳細は「コラム」を参照してください。

● 注意事項

- EPOCHをインストールすると、TEKTONもインストールされます。
- 変数は "{{ 変数名 }}" で指定した内容となります。

コラム：Manifestテンプレートとパラメータ

Manifestテンプレートをアップロードするとファイル内の定義文字が解析され、パラメータ入力できる状態になります。

```
6   labels:  
7     name: catalogue  
8   spec:  
9     replicas: {{ replicas }}  
10    selector:  
11      matchLabels:  
12        name: catalogue  
13    template:  
14      metadata:  
15        labels:  
16          name: catalogue  
17    spec:  
18      containers:  
19        - name: catalogue  
20          image: {{ image }} : {{ image_tag }}  
21          ports:  
22            - containerPort: 8000  
23    nodeSelector:
```

{{ パラメータ名 }} の形式で記述された文字がパラメータとして認識され、ユーザが入力できるようになります。
現在、EPOCHが使用できるパラメータ名は以下になります。

パラメータ名	説明
{{ replicas }}	レプリカ数
{{ image }}	コンテナイメージ
{{ image_tag }}	コンテナイメージのタグ
{{ port_no_active }}	ブルーグリーンデプロイ用のブルー面のポート番号
{{ port_no_preview }}	ブルーグリーンデプロイ用のグリーン面のポート番号
{{ epoch_var_01 }}	ユーザが自由に使用できる固定のパラメータ名01 (※)
{	
{{ epoch_var_20 }}	ユーザが自由に使用できる固定のパラメータ名20 (20が上限)

※ユーザ定義の任意パラメータ名が設定できるようになるのは、次バージョン以降で対応予定です



Exastro 