





**Table 46-1.** *(continued)*

Component	Description
 <p><b>PyTorch</b></p>	<p><b>PyTorch</b> is a Python deep learning library developed by Facebook based on the Torch library for Lua, a programming language.</p>
 <p><b>NVIDIA TensorRT</b></p>	<p><b>TensorRT</b> is a platform for high-performance and scalable deployment of deep learning models for inference.</p>
 <p><b>Seldon</b></p>	<p><b>Seldon</b> is an open source platform for deploying machine learning models on Kubernetes.</p>
 <p><b>TensorFlow</b></p>	<p><b>TensorFlow</b> provides an ecosystem for the large-scale productionalization of deep learning models. This includes distributed training using TFJob, serving with TF Serving, and other Tensorflow Extended components such as TensorFlow Model Analysis (TFMA) and TensorFlow Transform (TFT).</p>

## Working with Kubeflow

1. **Set up a Kubernetes cluster on GKE.**

```
# create a GKE cluster
gcloud container clusters create ekaba-gke-cluster

# view the nodes of the kubernetes cluster on GKE
kubectl get nodes
```

2. **Create OAuth client ID to identify Cloud IAP:** Kubeflow uses Cloud Identity-Aware Proxy (Cloud IAP) to connect to Jupyter and other running web apps securely. Kubeflow uses email addresses for authentication. In this section, we'll create an OAuth client ID which will be used to identify Cloud IAP when requesting access to a user's email account:

- Go to the [APIs & Services](#) ➤ [Credentials](#) page in GCP Console.
- Go to the OAuth consent screen (see Figure 46-2).
- Assign an Application name, for example, My-Kubeflow-App.
- For authorized domains, use [YOUR\_PROJECT\_ID].cloud.goog.

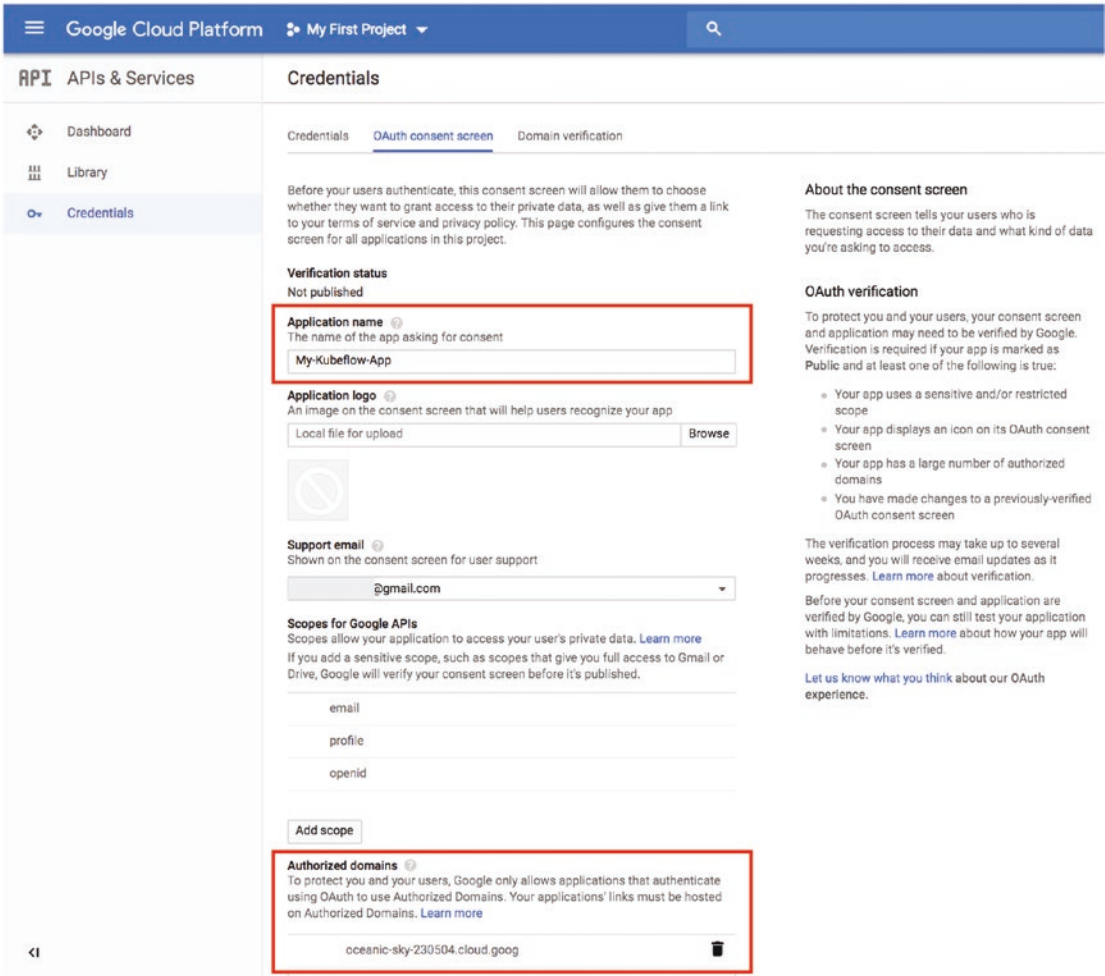
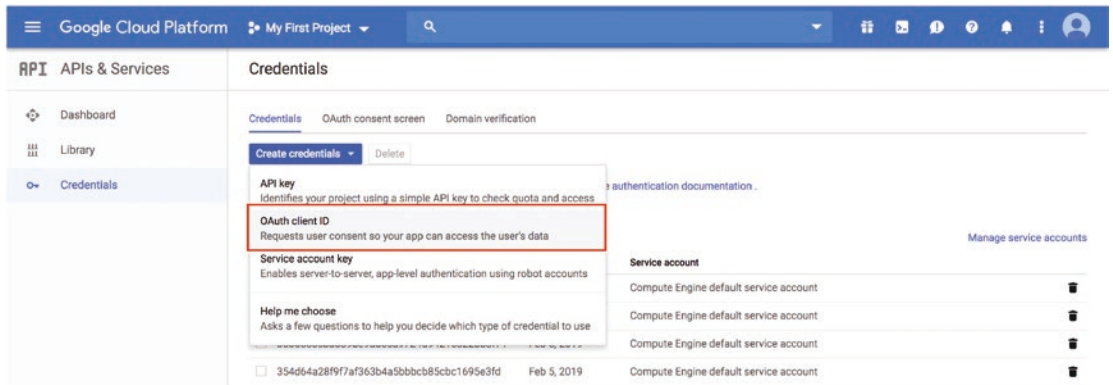


Figure 46-2. OAuth consent screen

- Go to the Credentials tab (see Figure 46-3).
- Click Create credentials, and then click OAuth client ID.
- Under Application type, select Web application.



**Figure 46-3.** GCP Credentials tab

- Choose a **Name** to identify the OAuth client ID (see Figure 46-4).

Google Cloud PlatformMy First Project

← Create OAuth client ID

For applications that use the OAuth 2.0 protocol to call Google APIs, you can use an OAuth 2.0 client ID to generate an access token. The token contains a unique identifier. See [Setting up OAuth 2.0](#) for more information.

**Application type**

☒ Web application

☐ Android [Learn more](#)

☐ Chrome App [Learn more](#)

☐ iOS [Learn more](#)

☐ Other

Nameekaba-kubeflow-oauth

**Restrictions**

Enter JavaScript origins, redirect URIs, or both [Learn More](#)

Origins and redirect domains must be added to the list of Authorized Domains in the [OAuth consent settings](#).

**Authorized JavaScript origins**

For use with requests from a browser. This is the origin URI of the client application. It can't contain a wildcard (https://\*.example.com) or a path (https://example.com/subdir). If you're using a nonstandard port, you must include it in the origin URI.

https://www.example.com

Type in the domain and press Enter to add it

**Authorized redirect URIs**

For use with requests from a web server. This is the path in your application that users are redirected to after they have authenticated with Google. The path will be appended with the authorization code for access. Must have a protocol. Cannot contain URL fragments or relative paths. Cannot be a public IP address.

https://ekaba-kubeflow-app.endpoints.oceanic-sky-230504.cloud.goog/\_gcp\_gatekeeper/authenticate

https://www.example.com

Type in the domain and press Enter to add it

Create

Cancel

Figure 46-4. Create OAuth client ID

- In the Authorized redirect URIs box, enter the following:  
`https://<deployment_name>.endpoints.<project>.cloud.goog/_gcp_gatekeeper/authenticate`
- <deployment\_name> must be the name of the Kubeflow deployment.
- <project> is the GCP project ID.

- In this case, it will be

[https://ekaba-kubeflow-app.endpoints.oceanic-sky-230504.cloud.google.com/\\_gcp\\_gatekeeper/authenticate](https://ekaba-kubeflow-app.endpoints.oceanic-sky-230504.cloud.google.com/_gcp_gatekeeper/authenticate)

- Take note of the client ID and client secret that appear in the OAuth client window. This is needed to enable Cloud IAP.

```
# Create environment variables from the OAuth client ID and
secret earlier obtained.
export CLIENT_ID=506126439013-drbrj036hihvdolgki6lflowm4bjb6c1.
apps.googleusercontent.com
export CLIENT_SECRET=bACWJuojIVm7PIMphzTOYz9D
export PROJECT=oceanic-sky-230504
```

## Download kfctl.sh

The file `kfctl.sh` is the Kubeflow installation shell script. As at this time of writing, the latest Kubeflow tag is 0.5.0.

```
# create a folder on the local machine
```

```
mkdir kubeflow
```

```
# move to created folder
```

```
cd kubeflow
```

```
# save folder path as a variable
```

```
export KUBEFLOW_SRC=$(pwd)
```

```
# download kubeflow `kfctl.sh`
```

```
export KUBEFLOW_TAG=v0.5.0
```

```
curl https://raw.githubusercontent.com/kubeflow/kubeflow/${KUBEFLOW_TAG}/
scripts/download.sh | bash
```

```
# list directory elements
```

```
ls -la
```

```
drwxr-xr-x  6 ekababisong  staff   204 17 Mar 04:15 .
drwxr-xr-x 25 ekababisong  staff   850 17 Mar 04:09 ..
drwxr-xr-x  4 ekababisong  staff   136 17 Mar 04:18 deployment
```

```
drwxr-xr-x 36 ekababisiong staff 1224 17 Mar 04:14 kubeflow
drwxr-xr-x 16 ekababisiong staff 544 17 Mar 04:14 scripts
```

## Deploy Kubeflow

Run the following code block to deploy Kubeflow.

```
# assign the name for the Kubeflow deployment
# The ksonnet app is created in the directory ${KFAPP}/ks_app
export KFAPP=ekaba-kubeflow-app

# run setup script
${KUBEFLOW_SRC}/scripts/kfctl.sh init ${KFAPP} --platform gcp --project
${PROJECT}

# navigate to the deployment directory
cd ${KFAPP}

# creates config files defining the various resources for gcp
${KUBEFLOW_SRC}/scripts/kfctl.sh generate platform

# creates or updates gcp resources
${KUBEFLOW_SRC}/scripts/kfctl.sh apply platform

# creates config files defining the various resources for gke
${KUBEFLOW_SRC}/scripts/kfctl.sh generate k8s

# creates or updates gke resources
${KUBEFLOW_SRC}/scripts/kfctl.sh apply k8s

# view resources deployed in namespace kubeflow
kubectl -n kubeflow get all
```

Kubeflow is available at a URL that will be unique for your deployment. In this case, Kubeflow is available to me at <https://ekaba-kubeflow-app.endpoints.oceanic-sky-230504.cloud.goog/> (see Figure 46-5). Again, this URL is unique for your deployment.



**Figure 46-5.** *The Kubeflow homescreen*

---

**Note** It can take 10–15 minutes for the URI to become available. Kubeflow needs to provision a signed SSL certificate and register a DNS name.

---

## Kubeflow Pipelines – Kubeflow for Poets

Kubeflow Pipelines is a simple platform for building and deploying containerized machine learning workflows on Kubernetes. Kubeflow pipelines make it easy to implement production-grade machine learning pipelines without bothering on the low-level details of managing a Kubernetes cluster.

Kubeflow Pipelines is a core component of Kubeflow and is also deployed when Kubeflow is deployed. The Pipelines dashboard is shown in Figure 46-6.