



Web



Deployment



WEB > GUIDES > PSPDFKIT SERVER > DEPLOYMENT

Deploy PSPDFKit Server on Google Cloud Platform



PSPDFKit Server has been deprecated and replaced by [Document Engine](#). To migrate to Document Engine and unlock advanced document processing capabilities, refer to our [migration guide](#). Learn more about these enhancements on our [blog](#).

This guide will walk you through the steps for deploying PSPDFKit Server to the [Google Cloud Platform](#) with [Kubernetes](#).

Creating a Google Cloud project

To create a Google Cloud project, please refer to the steps outlined in the Google Cloud Platform Documentation.

Setting up Google Cloud SDK

To be able to deploy PSPDFKit Server to the [Google Cloud Platform](#) with [Kubernetes](#), you have to set up the `gcloud` utility in order to manage your [Kubernetes](#) cluster in the command line.

To install `gcloud`, follow the installation instructions for your platform:

[Windows](#)

ASK AI

❖ [macOS](#)

❖ [Linux](#)

`kubectl` is a utility for running commands against Kubernetes clusters. After installing `gcloud`, add the `kubectl` components by running the following command:

```
gcloud components install kubectl
```



Make sure to set the [compute zone](#) to the one you want to deploy to. In this example, we are using `europe-west2-c`, but you can also pick another one if you are not located in Europe. Then insert it into the example below:

```
gcloud config set compute/zone europe-west2-c
```



Now run `gcloud compute zones list` to get a list of all available compute zones.

Creating a Kubernetes cluster

To create a [Kubernetes](#) cluster with the name `pspdfkit-example-cluster`, run the following:

```
gcloud container clusters create pspdfkit-example-cluster
```



This will create a cluster using the default settings. To get an overview of your clusters, visit the [Container Engine dashboard](#).

	Kubernetes clusters						
	+ CREATE CLUSTER REFRESH DELETE						
	Filter by label or name						
	Kubernetes clusters						
	<input type="checkbox"/> Name ^	Location	Cluster size	Total cores	Total memory	Notifications	Labels
	<input checked="" type="checkbox"/> pspdfkit-example-cluster	europe-west2-c	3	3 vCPUs	11.25 GB		Connect

Run the following command to make sure `kubectl` is connected to your cluster:

```
kubectl get pods
```



This command should print `No resources found.` When you get an error, try to authenticate to the Google Cloud Platform with `gcloud auth application-default login`.

Creating a Google Cloud SQL PostgreSQL database instance

To run PSPDFKit Server, you will need to set up a Postgres database for it. To do this, go to the Google Cloud SQL Instances page and click on Create Instance.

Cloud SQL

Cloud SQL Instances

Cloud SQL instances are fully managed, relational MySQL, PostgreSQL and SQL Server databases. Google handles replication, patch management and database management to ensure availability and performance. [Learn more](#)

To get started with Cloud SQL, you can create a new instance or use Cloud SQL to migrate your SQL database to Google Cloud.

[CREATE INSTANCE](#)

[MIGRATE DATA](#)

Then you will be able to select a database engine for Google Cloud SQL to run on. Here you need to select PostgreSQL.



PostgreSQL

Versions: 9.6, 10, 11, 12



Choose PostgreSQL

Finally, you have to set an instance ID, the default user password, and the region of the database.

Instance info

Instance ID

Choice is permanent. Use lowercase letters, numbers and hyphens. Start with a letter.

Default user password

Set a password for the 'postgres' user. A password is required for the user to log in.

[Learn more](#)

Location

For better performance, keep your data close to the services that need it.

Region

Choice is permanent

Zone


Can be changed at any time

Database version

PostgreSQL 9.6

PostgreSQL 10

PostgreSQL 11

PostgreSQL 12 

Make sure to choose PostgreSQL 11 for the database version.

Connecting the Kubernetes engine to the Cloud SQL instance

To enable the PSPDFKit Server deployment to connect to the Cloud SQL Instance, we will use the [Cloud SQL Proxy Docker image](#). First you need to [enable the Cloud SQL Administration API](#). Then you have to set up a service account with access privileges for your Cloud SQL instance.

To do this, go to the [service accounts page](#), click on Select, select your Google Cloud project, and then click on Create Service Account. Set a service account name and an ID, and make sure to select Cloud SQL Admin as the Project role and Furnish a new private key with the key type JSON.

Click on Save to save this service account and to download the service account private key file. Be sure to save this file, because you will need it later.

Create service account

A service account represents a Google Cloud service identity, such as code running on Compute Engine VMS, App Engine apps, or systems running outside Google.

Service account name

pspdfkitexampleserviceaccount

Describe what this service account will do

Service account ID

pspdfkitexampleserviceaccount @ [redacted] X ↺

Project role ?

Role

Cloud SQL Admin ▼



Full control of Cloud SQL resources.

+ ADD ANOTHER ROLE

☒ Furnish a new private key

Downloads a file that contains the private key. Store the file securely because this key can't be recovered if lost.

Key type

☒ JSON

Recommended

☐ P12

For backward compatibility with code using the P12 format

☐ Enable G Suite Domain-wide Delegation

Allows this service account to be authorized to access all users' data on a G Suite domain without manual authorization on their parts. [Learn more](#)

SAVE

CANCEL

Run the following command to create a user named `proxyuser` for the Google Cloud SQL database. This user will be used to access the database from the PSPDFKit Server deployment:

```
gcloud sql users create proxyuser host --instance=[INSTANCE_NAME] --password=
```



Make sure to replace `[INSTANCE_NAME]` with the name of your Google Cloud SQL instance and `[PASSWORD]` with a password for the user.

You also need to create two secrets for the Kubernetes Engine application to be able to access the Cloud SQL instance.

To create the secret for the service account, replace `[PROXY_KEY_FILE_PATH]` in the following command with the path where you saved the service accounts' private key, and then run it:

```
1 kubectl create secret generic cloudsql-instance-credentials \  
2   --from-file=credentials.json=[PROXY_KEY_FILE_PATH]
```



The second secret you need to create provides the proxy user account and its password. Here you need to replace `[PASSWORD]` with the password you set for the proxy user you just created:

```
1 kubectl create secret generic cloudsql-db-credentials \  
2   --from-literal=username=proxyuser --from-literal=password=[PASSWORD]
```



Creating a ConfigMap

[ConfigMaps](#) allow you to decouple configuration artifacts from image content. To create the `pspdfkit-config` [ConfigMap](#), run the following command:

```
kubectl create configmap pspdfkit-config
```



After the [ConfigMap](#) is created, you can edit it with the following:

This should open the created [ConfigMap](#) in your editor. Edit the file to match the following file and replace `activation_key` with your activation key:

```
1 # Please edit the object below. Lines beginning with a '#' will be ignored,
2 # and an empty file will abort the edit. If an error occurs while saving,
3 # this file will be reopened with the relevant failures.
4 #
5 apiVersion: v1
6 data:
7   activation_key: YOUR_ACTIVATION_KEY_GOES_HERE
8   api_auth_token: secret
9   dashboard_password: secret
10  dashboard_username: dashboard
11  jwt_algorithm: RS256
12  jwt_public_key: |
13    -----BEGIN PUBLIC KEY-----
14    MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA2gzhmJ9TDanEzWdP1WG+
15    0Ecwbe7f3bv6e5UUpvcT5q68IQJKP47AQdBANs1FVi4X9SaurbWoXdS6jpmPpk24
16    QvitzLNFphHdwjFBelTA0a6taZrSusoFvrtK9x5xsW4zzt/bkpUraNx82Z8MwLwr
17    t6HlY7dgO9+xBAabj4t1d2t+0HS8O/ed3CB6T21j6S8AbLDSEFc9ScO6Uc1XJlSo
18    rgyJJSPCpNhSq3AubEZ1wMS1iEtgAzTPRDSQv50qWIbn634HLWxTP/UH6YNJBwzt
19    3O6q29kTtjXlMGXCvin37PyX4Jy1IiPFwJm45aWJGKSfVGMDojTJbuUtM+8P9Rrn
20    AwIDAQAB
21    -----END PUBLIC KEY-----
22  pgdatabase: pspdfkit
23  secret_key_base: secret-key-base
24 kind: ConfigMap
```

Don't change anything that comes after the `kind: ConfigMap` line, because that part is autogenerated.

Creating the services and deployments

For the configuration of the proxy container you will use to connect to the [Cloud SQL database](#), you will need the instance connection name of the instance. To get this, run the following command:

```
gcloud sql instances describe [INSTANCE_NAME]
```

The output of this command will contain a line like this, where `pspdfkit-example-project:europa-west1:pspdfkitexampledb` is the name of the instance:



Kubernetes [services](#) and [deployments](#) can be configured in a file. To run PSPDFKit Server, you have to define a service and a deployment for PSPDFKit Server. To do this, create the `pspdfkit.yml` file in the current directory, ensure that the `pspdfkit/pspdfkit` image tag corresponds to the latest PSPDFKit Server version, and replace `pspdfkit-example-project:europa-west1:pspdfkitexampledb` with the name of the Cloud SQL database instance from the previous command:

```
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: pspdfkit
5  spec:
6    ports:
7      - protocol: TCP
8        port: 5000
9        targetPort: 5000
10   selector:
11     app: pspdfkit
12   type: LoadBalancer
13 ---
14 apiVersion: apps/v1
15 kind: Deployment
16 metadata:
17   name: pspdfkit
18 spec:
19   template:
20     metadata:
21       labels:
22         app: pspdfkit
23     spec:
24       containers:
25         - name: cloudsql-proxy
26           image: gcr.io/cloudsql-docker/gce-proxy:1.11
27           command: ["/cloud_sql_proxy",
28                     "-instances=pspdfkit-example-project:europa-west1:pspdfkit",
29                     "-credential_file=/secrets/cloudsql/credentials.json"]
30       volumeMounts:
31         - name: cloudsql-instance-credentials
32           mountPath: /secrets/cloudsql
33           readOnly: true
34       - image: pspdfkit/pspdfkit:2024.1.2
35         name: pspdfkit
36         env:
37           - name: PGUSER
38             valueFrom:
39               secretKeyRef:
40                 name: cloudsql-db-credentials
41                 key: username
42           - name: PGPASSWORD
```




```
43         valueFrom:
44             secretKeyRef:
45                 name: cloudsql-db-credentials
46                 key: password
47     - name: PGDATABASE
48       valueFrom:
49         configMapKeyRef:
50             name: pspdfkit-config
51             key: pgdatabase
52     - name: PGHOST
53       value: "127.0.0.1"
54     - name: PGPORT
55       value: "5432"
56     - name: ACTIVATION_KEY
57       valueFrom:
58         configMapKeyRef:
59             name: pspdfkit-config
60             key: activation_key
61     - name: API_AUTH_TOKEN
62       valueFrom:
63         configMapKeyRef:
64             name: pspdfkit-config
65             key: api_auth_token
66     - name: SECRET_KEY_BASE
67       valueFrom:
68         configMapKeyRef:
69             name: pspdfkit-config
70             key: secret_key_base
71     - name: JWT_ALGORITHM
72       valueFrom:
73         configMapKeyRef:
74             name: pspdfkit-config
75             key: jwt_algorithm
76     - name: JWT_PUBLIC_KEY
77       valueFrom:
78         configMapKeyRef:
79             name: pspdfkit-config
80             key: jwt_public_key
81     - name: DASHBOARD_USERNAME
82       valueFrom:
83         configMapKeyRef:
84             name: pspdfkit-config
85             key: dashboard_username
86     - name: DASHBOARD_PASSWORD
87       valueFrom:
88         configMapKeyRef:
89             name: pspdfkit-config
90             key: dashboard_password
91   ports:
92     - containerPort: 5000
93       name: pspdfkit
94   volumes:
95     - name: cloudsql-instance-credentials
```



secret:

To create the services and deployments needed to run PSPDFKit Server, execute the following:

```
kubectl create -f ./pspdfkit.yml
```



Viewing the dashboard

To be able to access the server, you have to get the external IP address that was assigned to the server. Run the following command to view all the services in your cluster, along with their assigned external IP addresses:

```
kubectl get services
```



This will show something like the following:

	NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)
1	kubernetes	ClusterIP	10.15.240.1	<none>	443/TCP
2	pspdfkit	LoadBalancer	10.15.247.197	12.345.678.910	5000:32393/TCP



Copy the `EXTERNAL-IP` address from the `pspdfkit` column and access the [dashboard](#) with the port `5000` and the `/dashboard` path in your web browser. In this example, you would access the dashboard with `http://12.345.678.910:5000/dashboard`.

Limitations

Be aware that this is just an example setup, and we recommend looking deeper into the Google Cloud Platform for a production-ready setup.

Was this helpful?

✓ YES

✗ NO

Questions? [Contact us](#)

