Gitlab CI Guide.

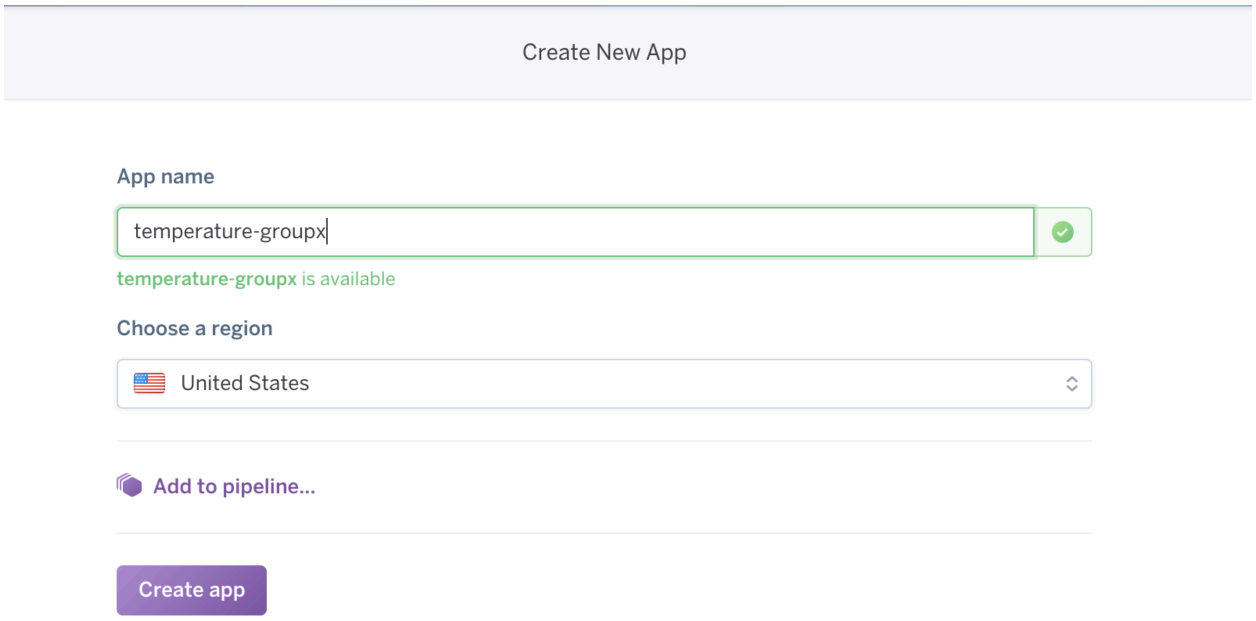
This guide will help you set up CI with an Uber JAR instead of a WAR. If you already have CI working, by no means do you have to change it.

Overall Steps:

1. Set up Heroku
2. Make secret variables
3. Add 'stage' task to build.gradle
4. Add .gitlab-ci.yml
5. Add Procfile

Set up Heroku

1. Create a new App – choose any appropriate, unique name



Create New App

App name

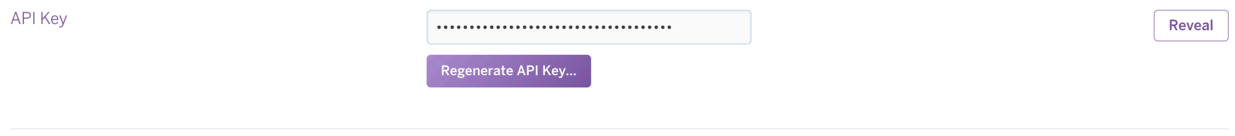
temperature-groupx

temperature-groupx is available

Choose a region

United States

2. Find your API Key under *Account Settings*

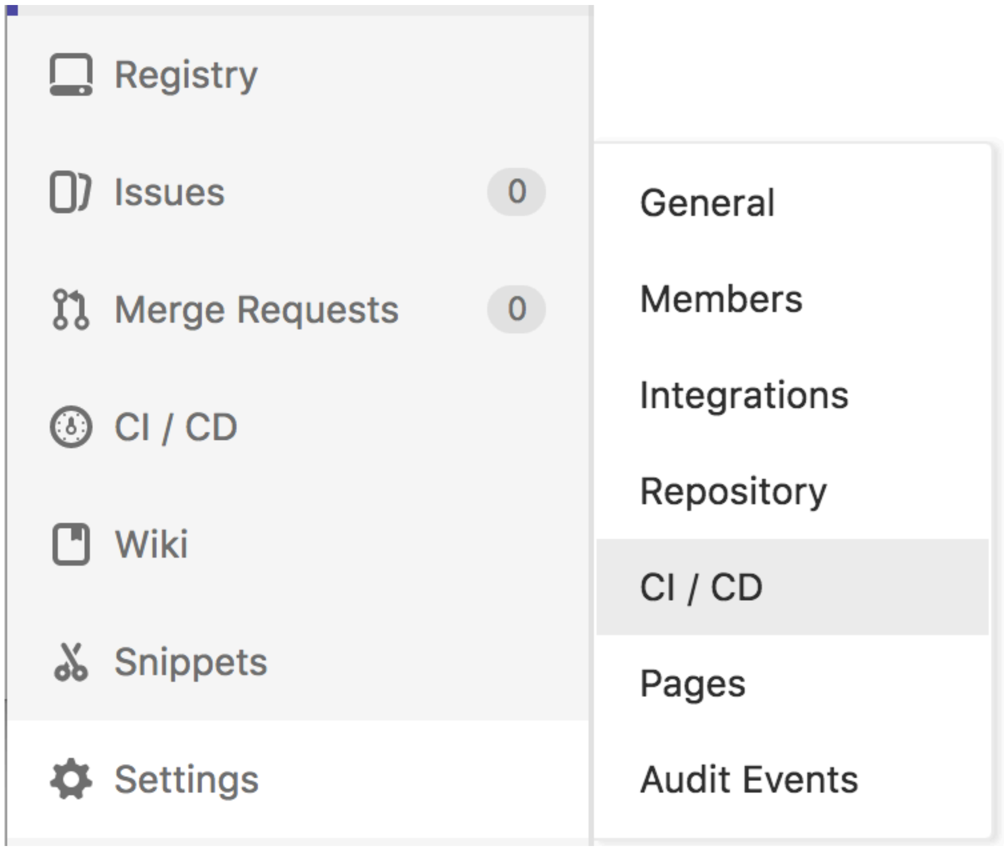


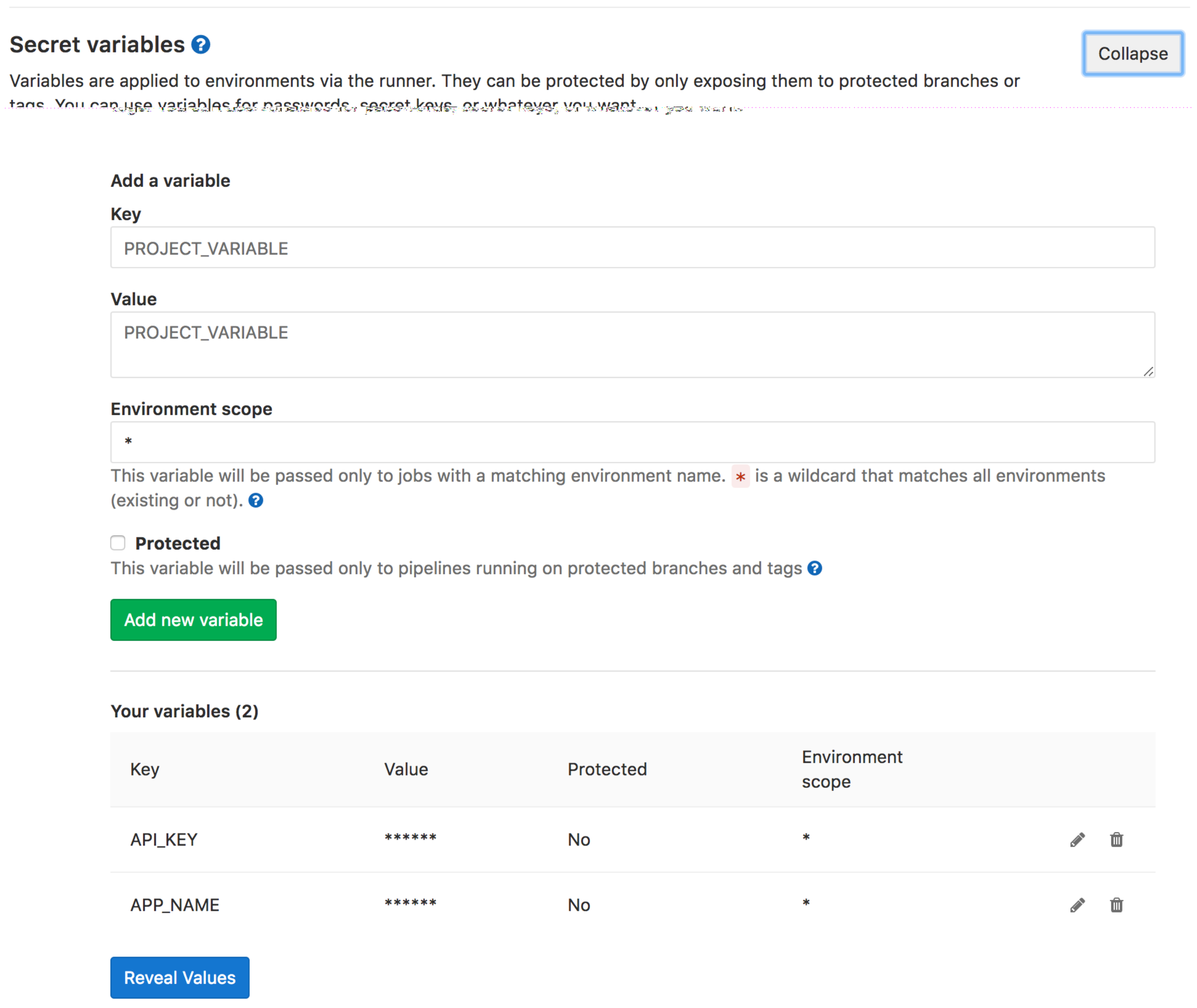
API Key

••

Make Secret Variables

1. Within your Project on Gitlab go to *Settings -> CI/CD -> Secret Variables*





Secret variables

Collapse

Variables are applied to environments via the runner. They can be protected by only exposing them to protected branches or tags. You can use variables for passwords, secret keys, or whatever you want.

Add a variable

Key

PROJECT_VARIABLE

Value

PROJECT_VARIABLE

Environment scope

*

This variable will be passed only to jobs with a matching environment name. * is a wildcard that matches all environments (existing or not).

Protected

This variable will be passed only to pipelines running on protected branches and tags

Add new variable

Your variables (2)

Key	Value	Protected	Environment scope
API_KEY	******	No	*
APP_NAME	******	No	*

Reveal Values

2. Create 2 variables API_KEY and APP_NAME and the value should be the API Key you found in the Set up Heroku step and the the name you chose for your app (in our case *temperature-groupx*)

Add ‘stage’ task to build.gradle

Heroku by default looks for a stage task to run, so we’ll need to add that task that cleans then builds our JAR.

1. At the bottom of the build.gradle add a new task:

```
task stage() {
   dependsOn clean, build
}
build.mustRunAfter clean
```

Add .gitlab-ci.yml

1. Create a new file called .gitlab-ci.yml
2. At the top, specify the image:

```
image: java:10
```

3. Add a deploy stage and use dpl to deploy your app to heroku:

```
deploy:
   stage: deploy
   image: ruby:2.3
   script:
      - apt-get update -qy
      - apt-get install -y ruby-dev
      - gem install dpl
      - dpl --provider=heroku --app=$APP_NAME --api-key=$API_KEY
```

4. Finally, add only master to ensure the deploy pipeline only runs when a change has been made to the master branch.

```
only:
- master
```

Add Procfile

Add a new file to the root of your project called *Procfile*. This file tells Heroku how to run your jar on its servers.

```
web: java -Dspring.profiles.active=qa -Dserver.port=$PORT -jar build/libs/*.jar
```