Урок 8. Деплой тестового приложения в кластер, CI/CD

Практическое задание

Студент Виталий Зайцев, DE\_622

Contents

[Домашнее задание 3](#_Toc84689605)

[Реализация 4](#_Toc84689606)

[Создание GitLab аккаунта 4](#_Toc84689607)

[Добавление SSH-ключа в GitLab аккаунт 4](#_Toc84689608)

[Генерация SSH-ключа 4](#_Toc84689609)

[Добавление SSH ключа в аккаунт 6](#_Toc84689610)

[Создание репозитория 9](#_Toc84689611)

[Создание нового каталога и инициирование создание «пустого» репозитория 9](#_Toc84689612)

[Создание репозитория 9](#_Toc84689613)

[Добавление тестового файла, добавление к репозиторию и коммит 10](#_Toc84689614)

[Активация GitLab Runner 12](#_Toc84689615)

[Создание Access token 12](#_Toc84689616)

[Отключение Shared Runners 14](#_Toc84689617)

[Подготовка GitLub Runner манифеста: gitlab-runner.yaml 17](#_Toc84689618)

[Применение манифеста GitLub Runner манифеста: gitlab-runner.yaml 18](#_Toc84689619)

[Проверка появления Runner 19](#_Toc84689620)

[Содание и конфигурирование namespaces к деплою Runner’ом 21](#_Toc84689621)

[Создание namespaces с именами stage и prod 21](#_Toc84689622)

[Создание авторизационных объектов 21](#_Toc84689623)

[Binking ролей 21](#_Toc84689624)

[Получение токенов для деплоя в namespaces 21](#_Toc84689625)

[Создание в GitLab проекте переменных сред stage и prod 23](#_Toc84689626)

[Создание secrets для авторизации и деплоя Kubernetes в GitLab Registry 26](#_Toc84689627)

[НЕУСПЕХ! Развёртывание приложений 29](#_Toc84689628)

[Применение манифестов из kube\postgres 29](#_Toc84689629)

[Корректировка настроек ingress и маппим на stage и prod 29](#_Toc84689630)

[НЕУСПЕХ! Проверка работы приложения: нет доступа CURL’ом 30](#_Toc84689631)

[НЕУСПЕХ! Настройка CI: .gitlab-ci.yml 31](#_Toc84689632)

[НЕУСПЕХ! Перенос Go файлов и коммит на удалённый репозиторий 33](#_Toc84689633)

[Само домашнее задание НЕ НАЧАТО! 34](#_Toc84689634)

# Домашнее задание

В рамках данного задания нужно продолжить работу с CI приложения из лекции. То есть для начала выполнения этого домашнего задания необходимо проделать то, что показывалось в лекции. Все задание должно выполняться применительно к файлам в директории practice/8.ci-cd/app

Переделайте шаг деплоя в CI/CD, который демонстрировался на лекции таким образом, чтобы при каждом прогоне шага deploy в кластер применялись манифесты приложения. При этом версия докер образа в деплойменте при апплае должна подменяться на ту, что была собрана в шаге build.

Для этого самым очевидным способом было бы воспользоваться утилитой sed.

* Измените образ в деплойменте приложения (файл kube/deployment.yaml) на плейсхолдер.  
  Вот это

image: nginx:1.12 # это просто плэйсхолдер  
На это

* + image: IMAGE
* Измените шаг деплоя в .gitlab-ci.yml, чтобы изменять IMAGE на реальное имя образа и тег

Это

kubectl set image deployment/$CI\_PROJECT\_NAME \*=$CI\_REGISTRY\_IMAGE:$CI\_COMMIT\_REF\_SLUG.$CI\_PIPELINE\_ID --namespace $CI\_ENVIRONMENT\_NAME  
На это

sed -i "s,**IMAGE**,$CI\_REGISTRY\_IMAGE:$CI\_COMMIT\_REF\_SLUG.$CI\_PIPELINE\_ID,g" kube/deployment.yaml

kubectl apply -f kube/ --namespace $CI\_ENVIRONMENT\_NAME

Вторую строчку шага деплоя (которая отслеживает статус деплоя) оставьте без изменений.

Попробуйте закоммитить свои изменения, запушить их в репозиторий (тот же, который вы создавали во время лекции на Gitlab.com) и посмотреть на выполнение CI в интерфейсе Gitlab.

Так как окружений у нас два (stage и prod), то помимо образа при апплае из CI нам также было бы хорошо подменять host в ingress.yaml. Попробуйте реализовать это по аналогии, подставляя в ингресс вместо плэйсхолдера значение переменной $CI\_ENVIRONMENT\_NAME

Так же попробуйте протестировать откат на предыдущую версию, при возникновении ошибки при деплое

Для этого можно изменить значение переменной DB\_HOST в deployment.yaml на какое нибудь несуществующее. Тогда при старте приложения оно не сможет найти БД и будет постоянно рестрартовать. CI должен в течении progressDeadlineSeconds: 300 и по.сле этого запустить процедуру отката. При этом не должно возникать недоступности приложения, так как старая реплика должна продолжать работать, пока новая пытается стартануть.

Путь к файлам <https://github.com/adterskov/geekbrains-conteinerization/tree/master/practice/8.ci-cd/app>

# Реализация

## Создание GitLab аккаунта

Аккаунт: <https://gitlab.com/VitaliiZaitsev>

Проект: <https://gitlab.com/vitaliizaitsev_public/gb_cicd>

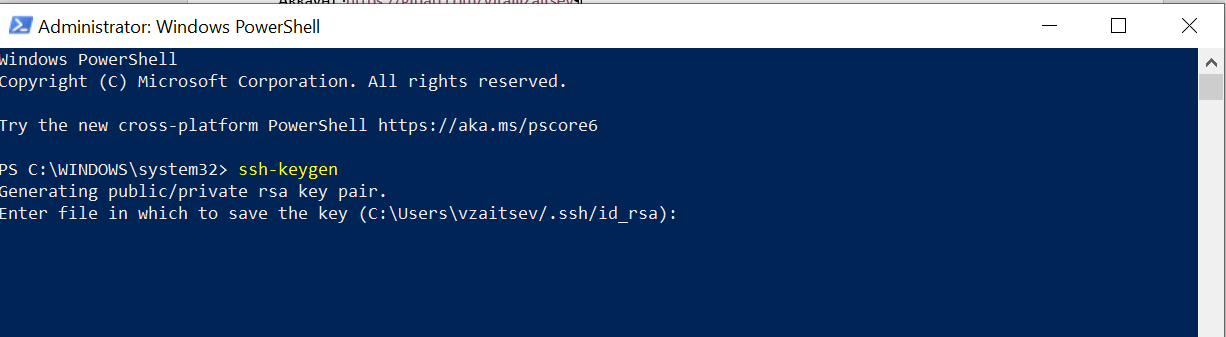
## Добавление SSH-ключа в GitLab аккаунт

### Генерация SSH-ключа

Инструкция: <https://docs.gitlab.com/ee/ssh/index.html#generate-an-ssh-key-pair>

Run Windows PowerShell as Administrator

ssh-keygen

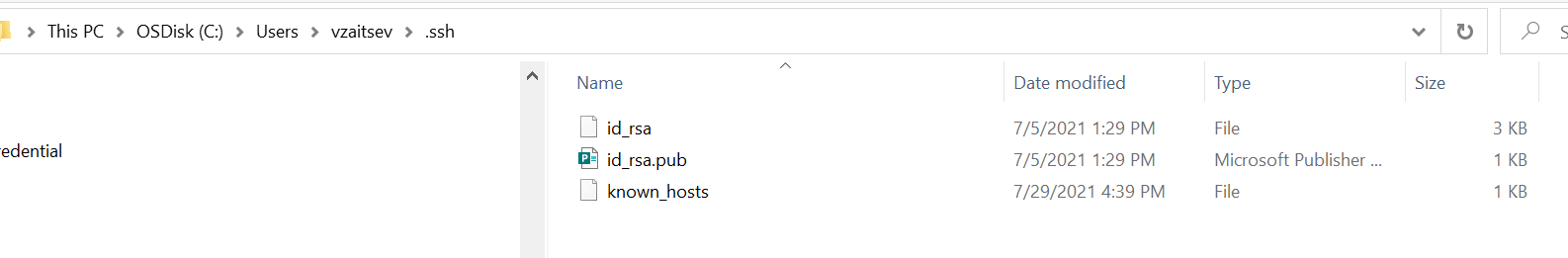


Путь размещения ключа

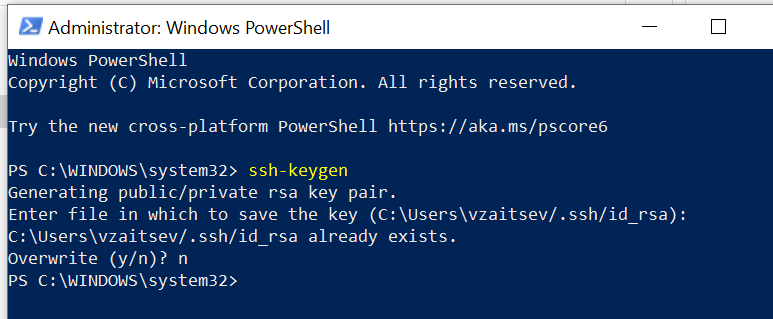
C:\Users\vzaitsev/.ssh/id\_rsa

Перечень фалов ДО генерации

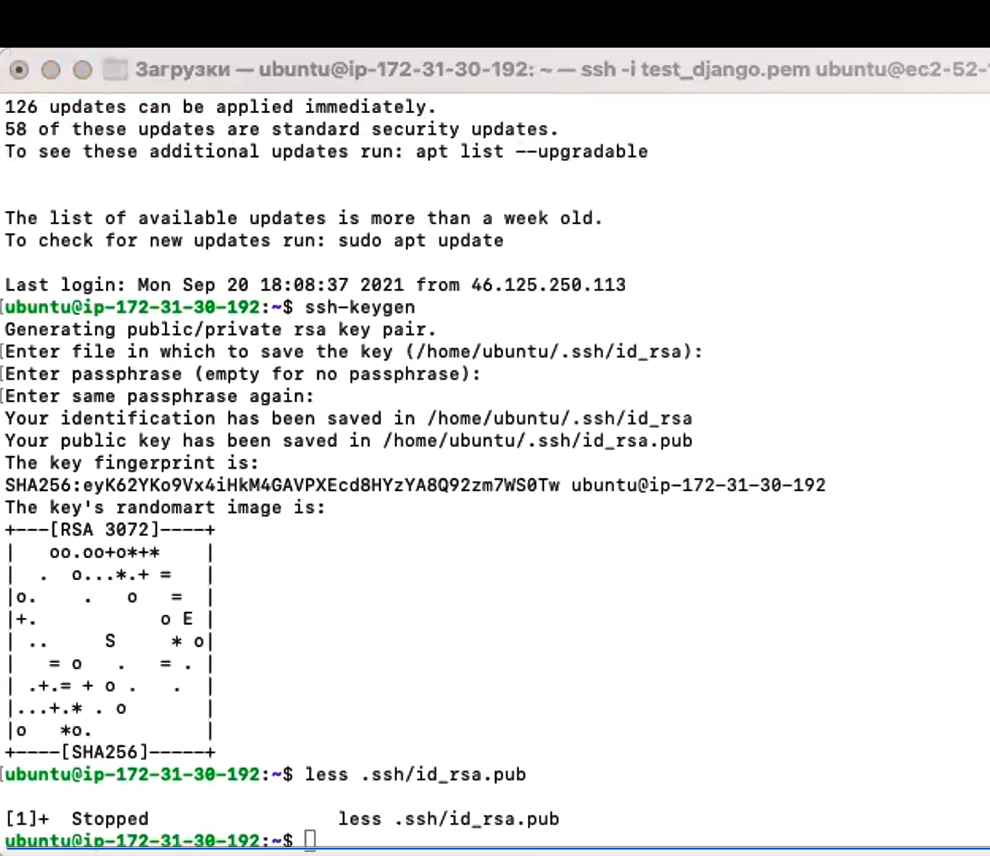
C:\Users\vzaitsev\.ssh



Решил НЕ перегенерировать ключ, использовать имеющийся



А иначе бы ключ был бы сгенерирован, см пример из лекции

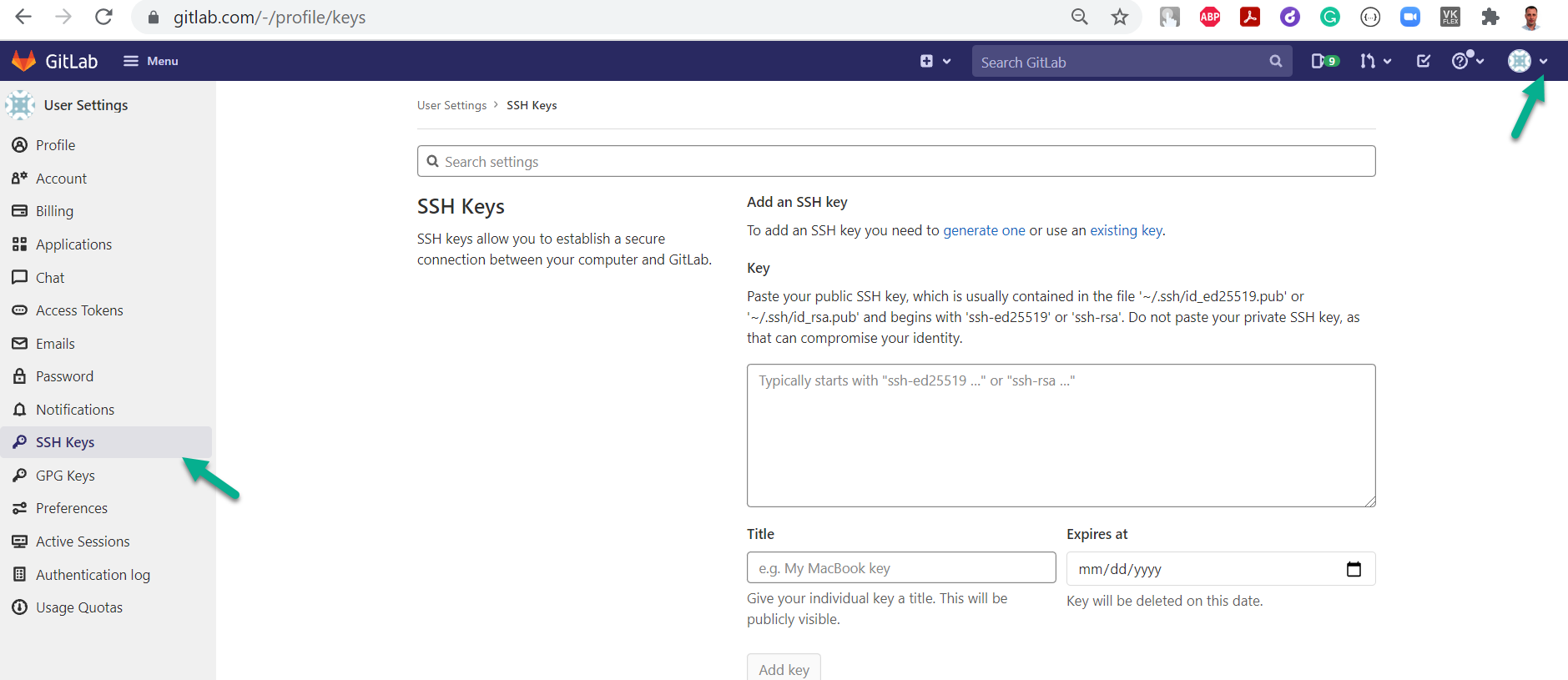


Открытие id\_rsa.pub и копирование ключа (для последующей настройки в GitLab репозитории)

ssh-rsa  vzaitsev@MOW-LAT-19-2455

### Добавление SSH ключа в аккаунт

Account ->Preferences->SSH keys

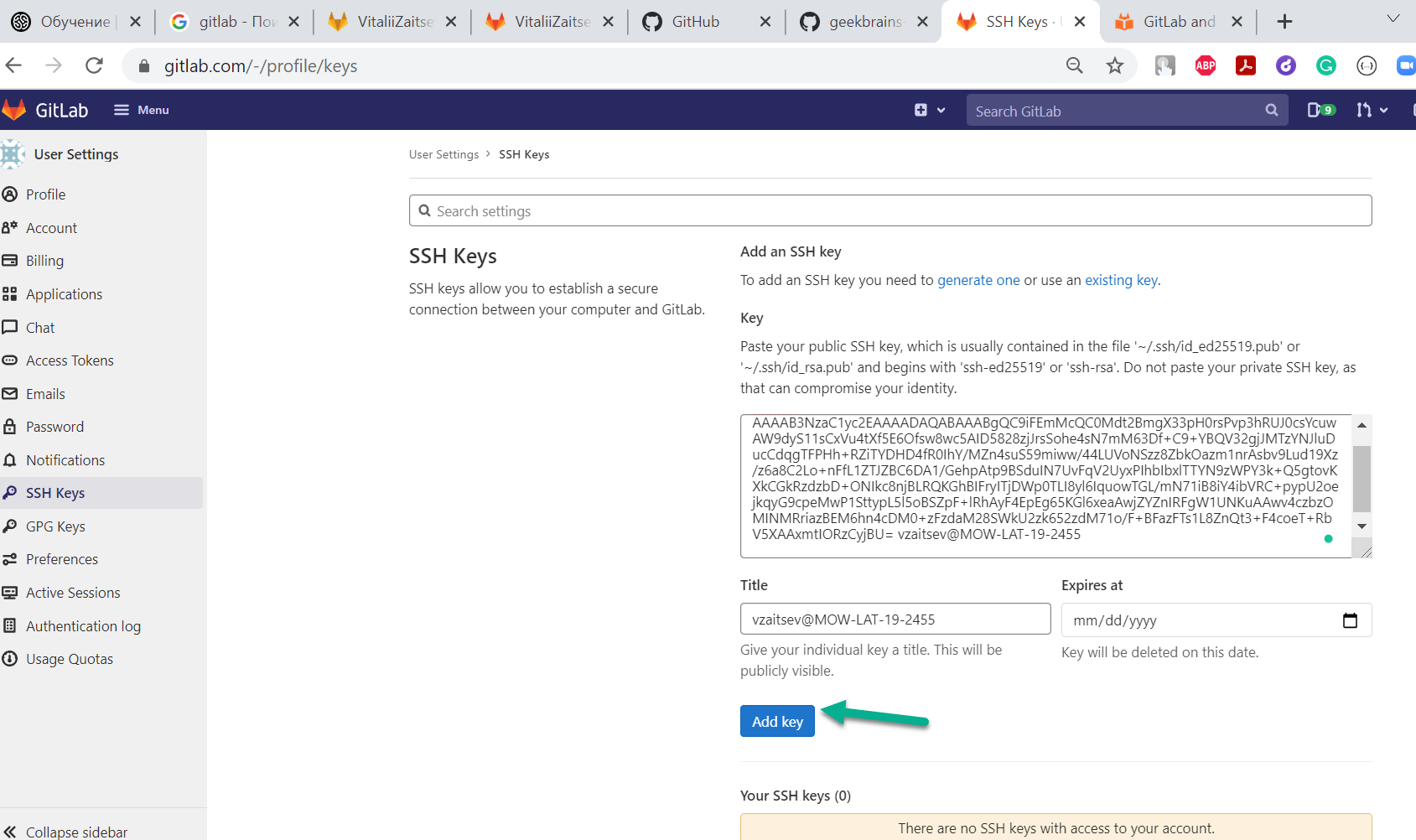


Key

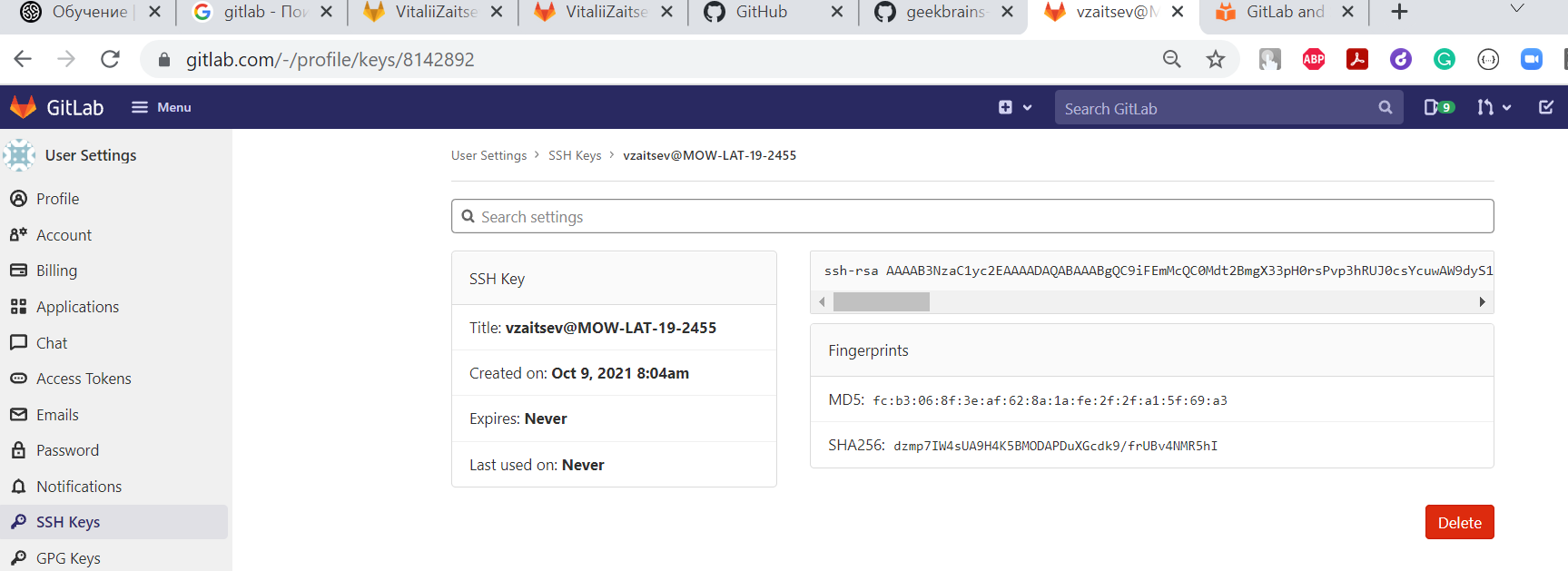
ssh-rsa  vzaitsev@MOW-LAT-19-2455

Title

vzaitsev@MOW-LAT-19-2455



<https://gitlab.com/-/profile/keys/8142892>



ssh-rsa  vzaitsev@MOW-LAT-19-2455

SSH Key

Title: vzaitsev@MOW-LAT-19-2455

Created on: Oct 9, 2021 8:04am

Expires: Never

Last used on: Never

Fingerprints

MD5: fc:b3:06:8f:3e:af:62:8a:1a:fe:2f:2f:a1:5f:69:a3

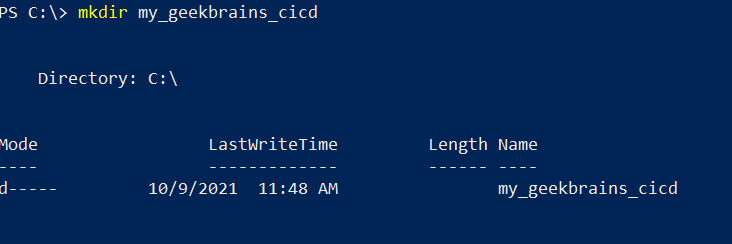
SHA256: dzmp7IW4sUA9H4K5BMODAPDuXGcdk9/frUBv4NMR5hI

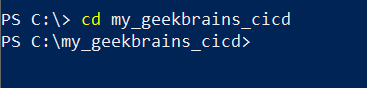
## Создание репозитория

### Создание нового каталога и инициирование создание «пустого» репозитория

mkdir my\_geekbrains\_cicd

cd my\_geekbrains\_cicd

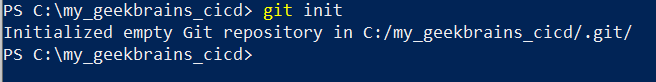




### Создание репозитория

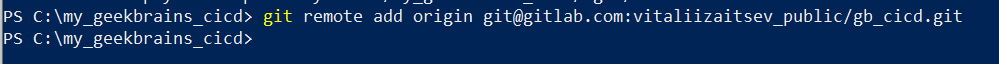
Инициирование локального, «пустого» репозитория (пока локального)

git init



Добавление удалённого репозитория с **МОЕМУ** проекту[[1]](#footnote-1)

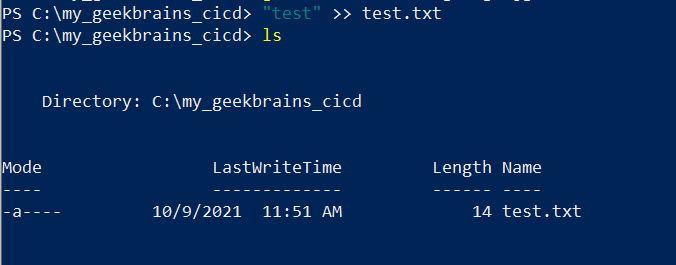
git remote add origin git@gitlab.com:vitaliizaitsev\_public/gb\_cicd.git



### Добавление тестового файла, добавление к репозиторию и коммит

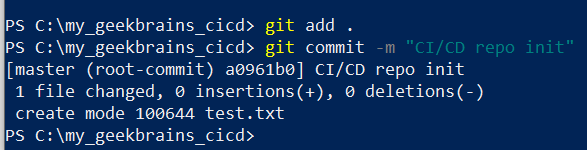
"test" >> test.txt

ls

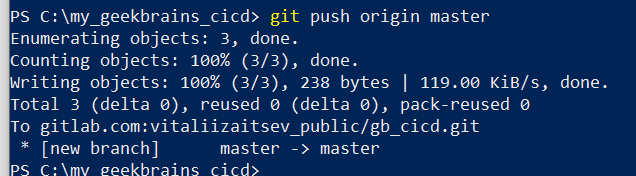


git add .

git commit -m "CI/CD repo init"

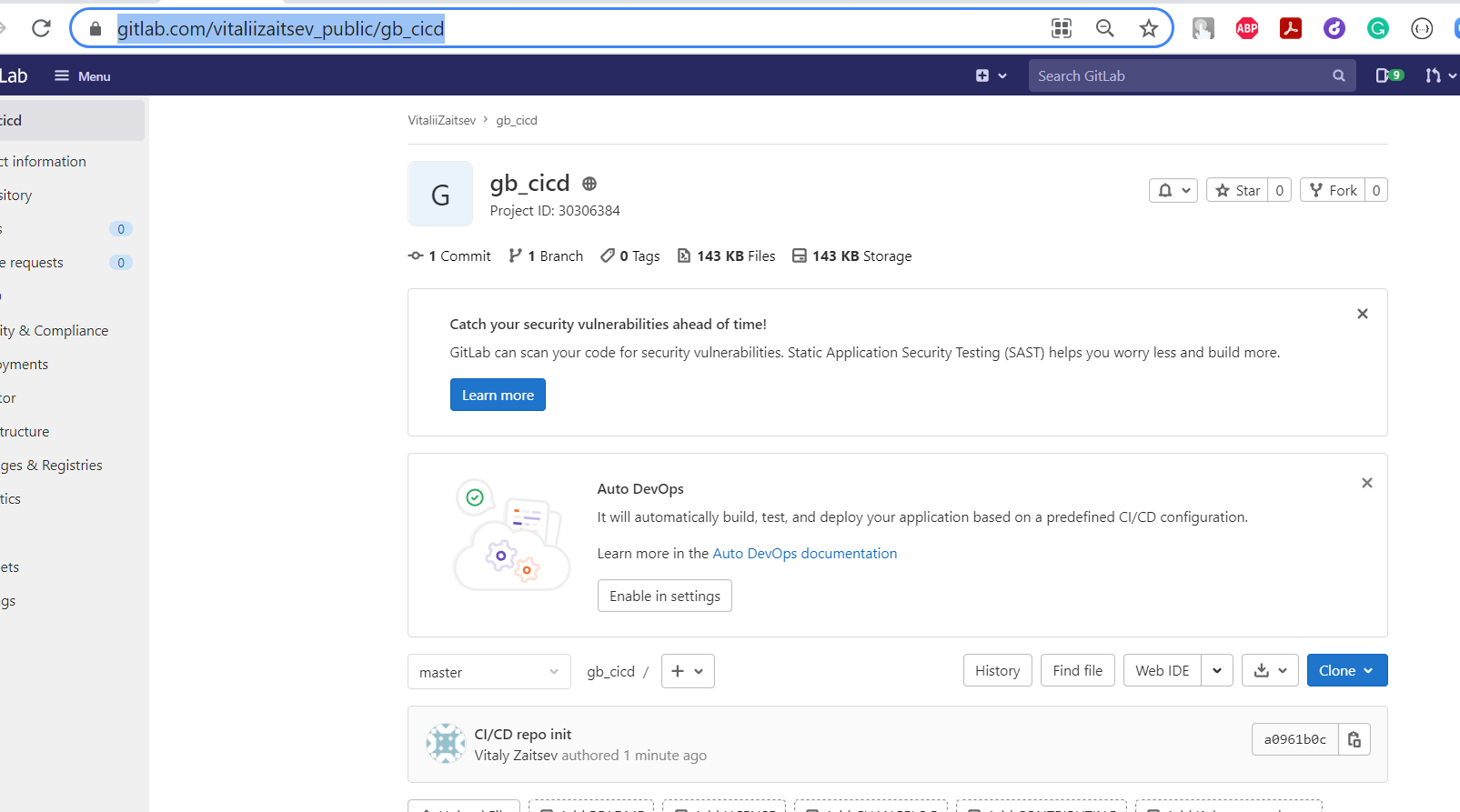


git push origin master



<https://gitlab.com/vitaliizaitsev_public/gb_cicd>

commit: a0961b0ca29519747c71dccce47b954d9e2680e3

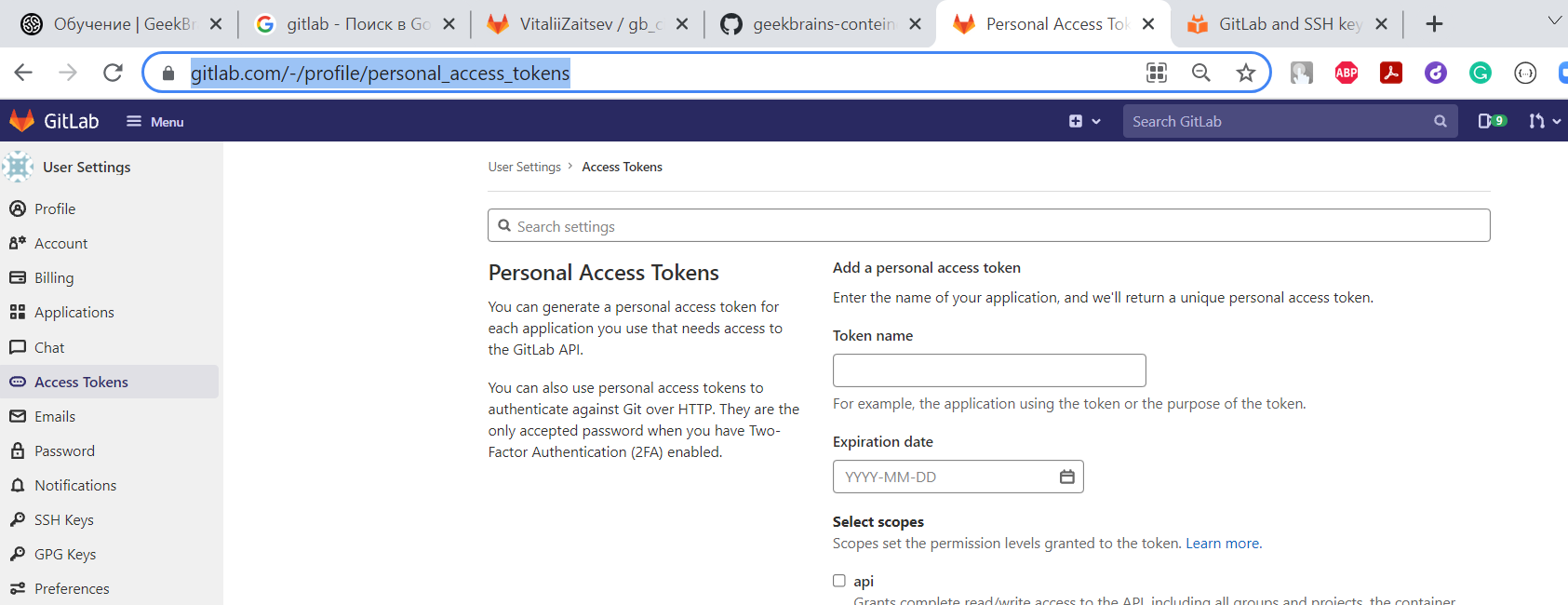


## Активация GitLab Runner

### Создание Access token

Settings -> Preferences -> Access tokens

<https://gitlab.com/-/profile/personal_access_tokens>

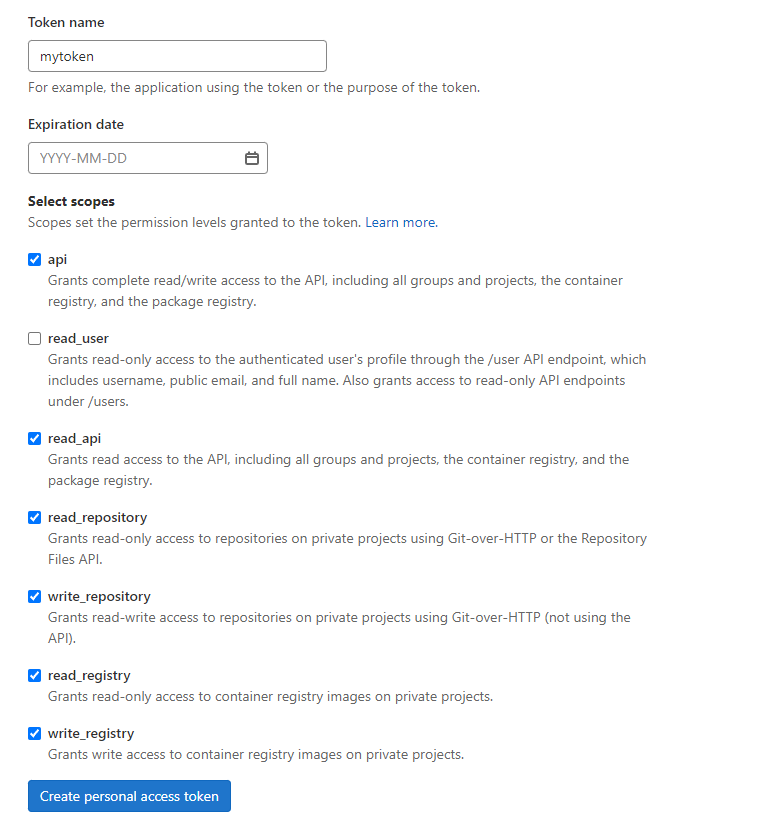


Token name

mytoken

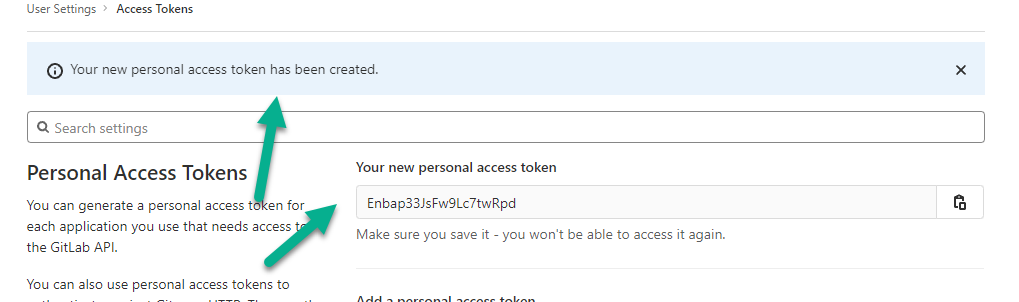
No expiration date

Permissions



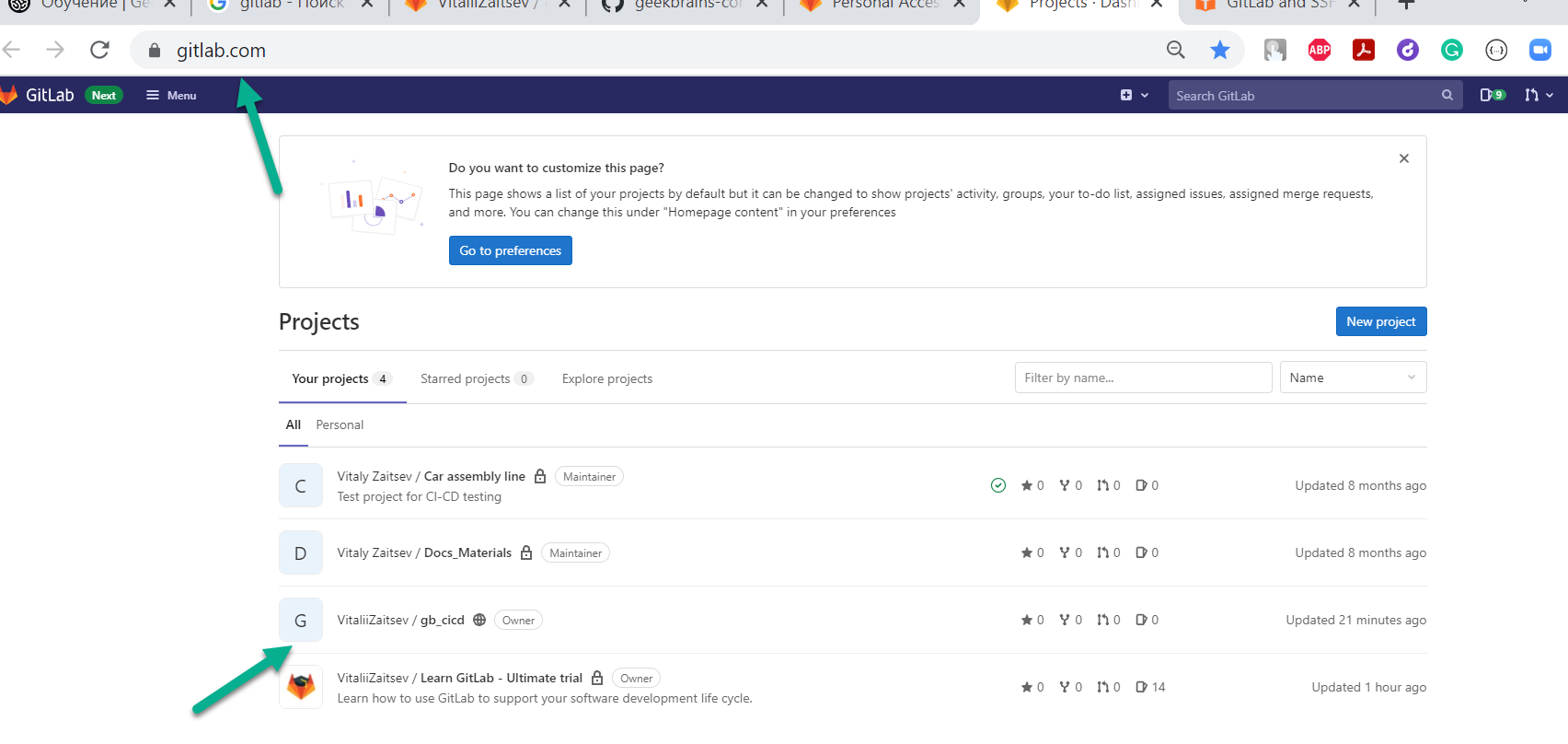
Your new personal access token

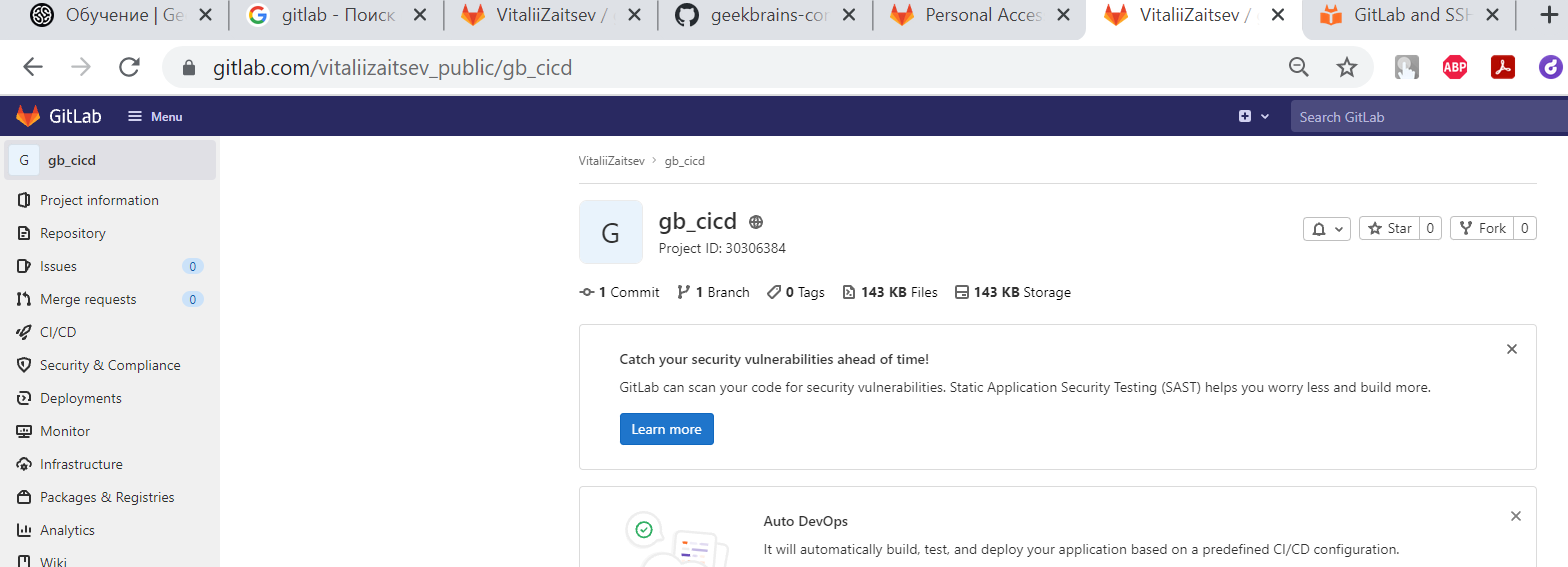
Enbap33JsFw9Lc7twRpd



### Отключение Shared Runners

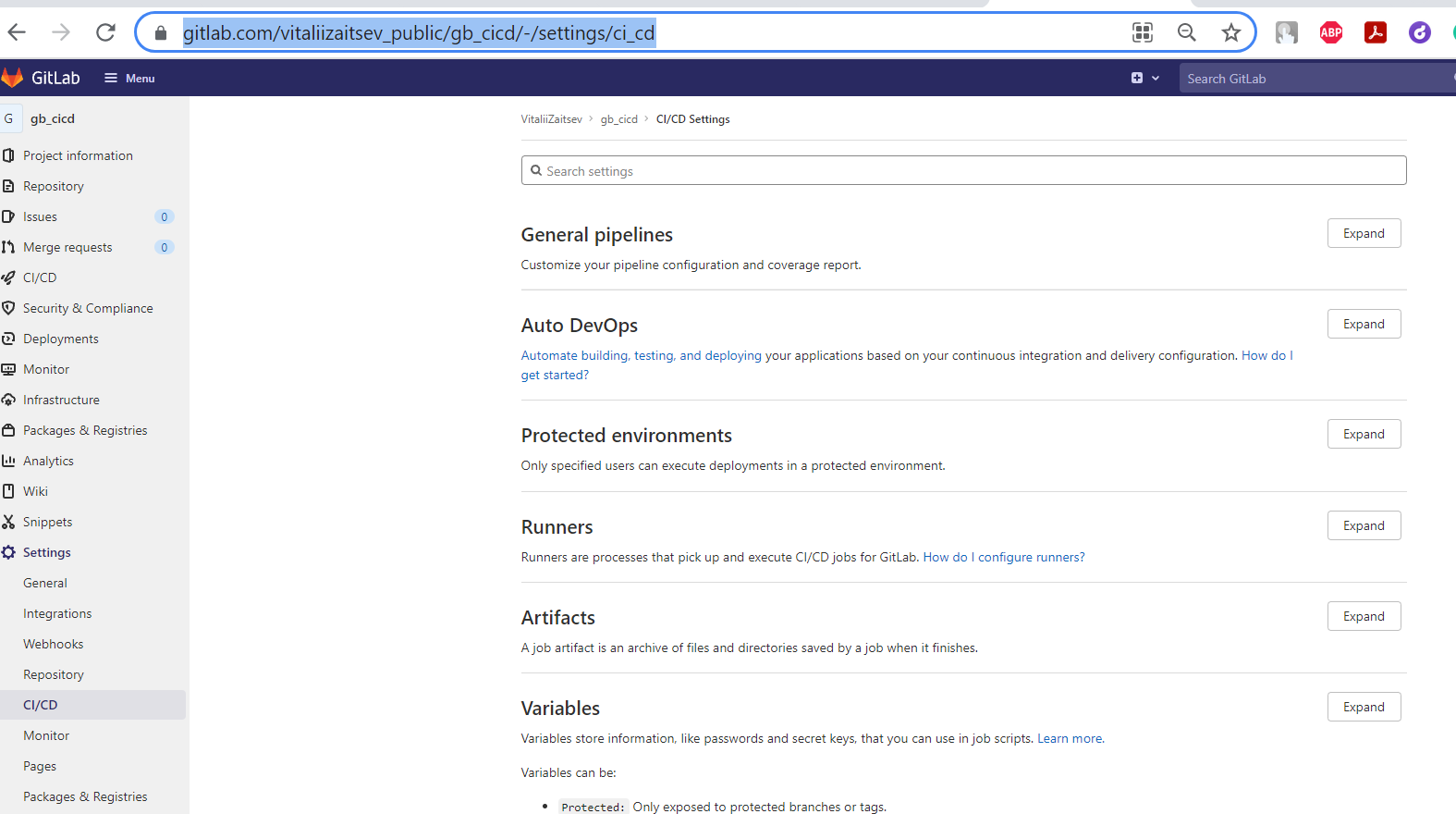
Главная страница -> Репозиторий



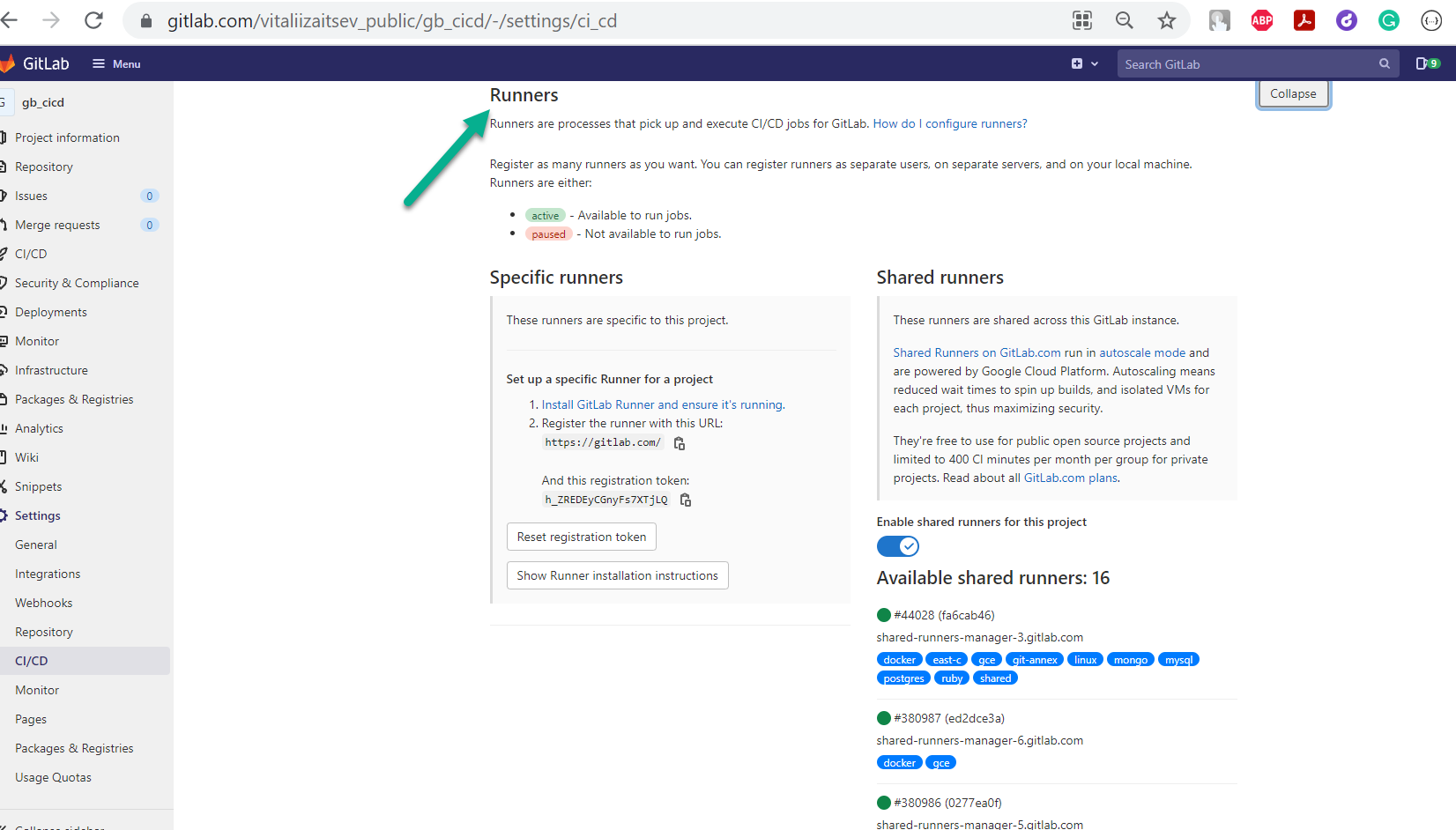


Settings -> CICD

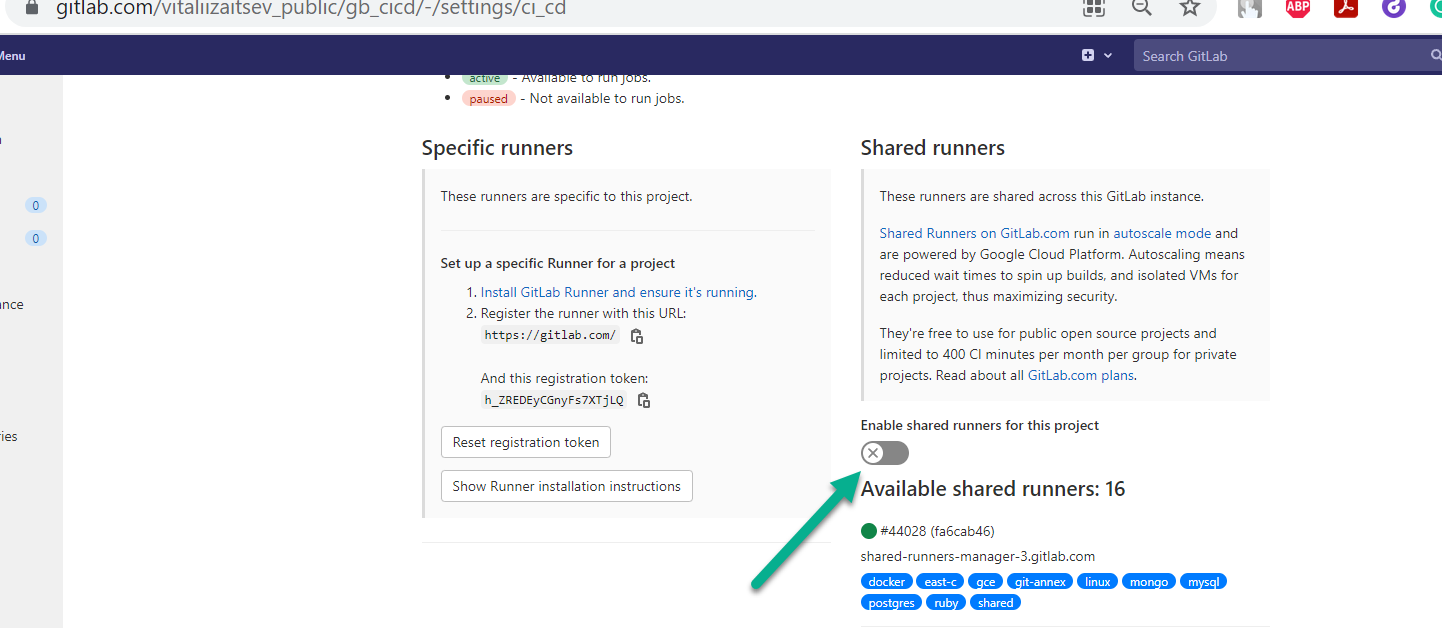
<https://gitlab.com/vitaliizaitsev_public/gb_cicd/-/settings/ci_cd>



Секция Runners



ОТКЛЮЧЕНИЕ ОПЦИИ **Enable shared runners for this project**



**Скопируем Specific runners информацию**

These runners are specific to this project.

Set up a specific Runner for a project

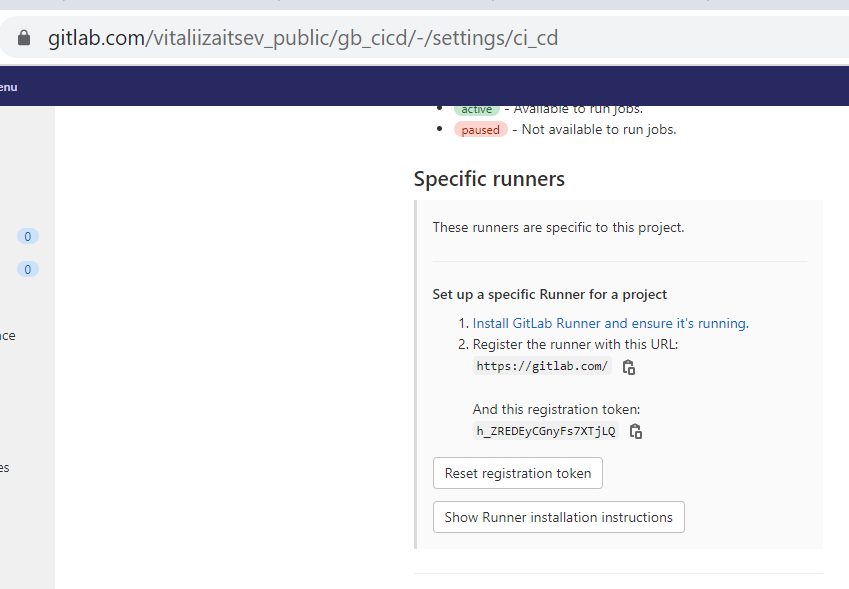
Install GitLab Runner and ensure it's running.

Register the runner with this URL:

https://gitlab.com/

And this registration token:

h\_ZREDEyCGnyFs7XTjLQ



### Подготовка GitLub Runner манифеста: gitlab-runner.yaml

Меняем

stringData:

runner-registration-token: <CHANGE ME>

runner-token: ""

на

stringData:

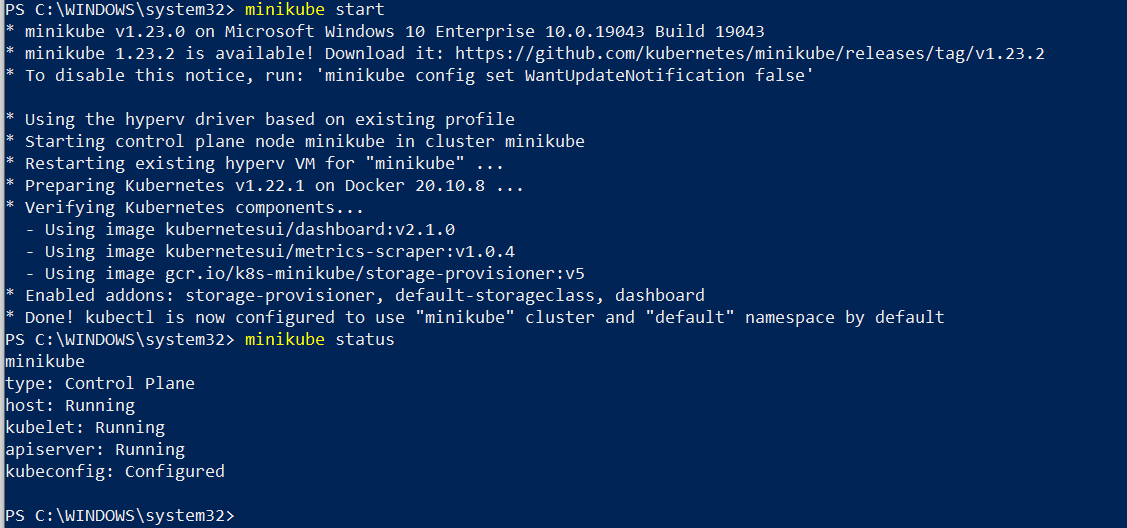
runner-registration-token: h\_ZREDEyCGnyFs7XTjLQ

runner-token: ""

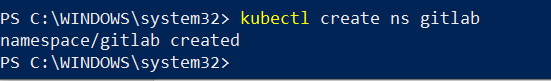
### Применение манифеста GitLub Runner манифеста: gitlab-runner.yaml

minikube start

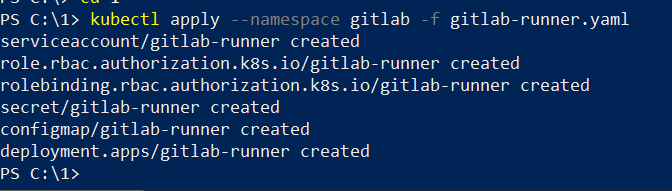
minikube status



kubectl create ns gitlab

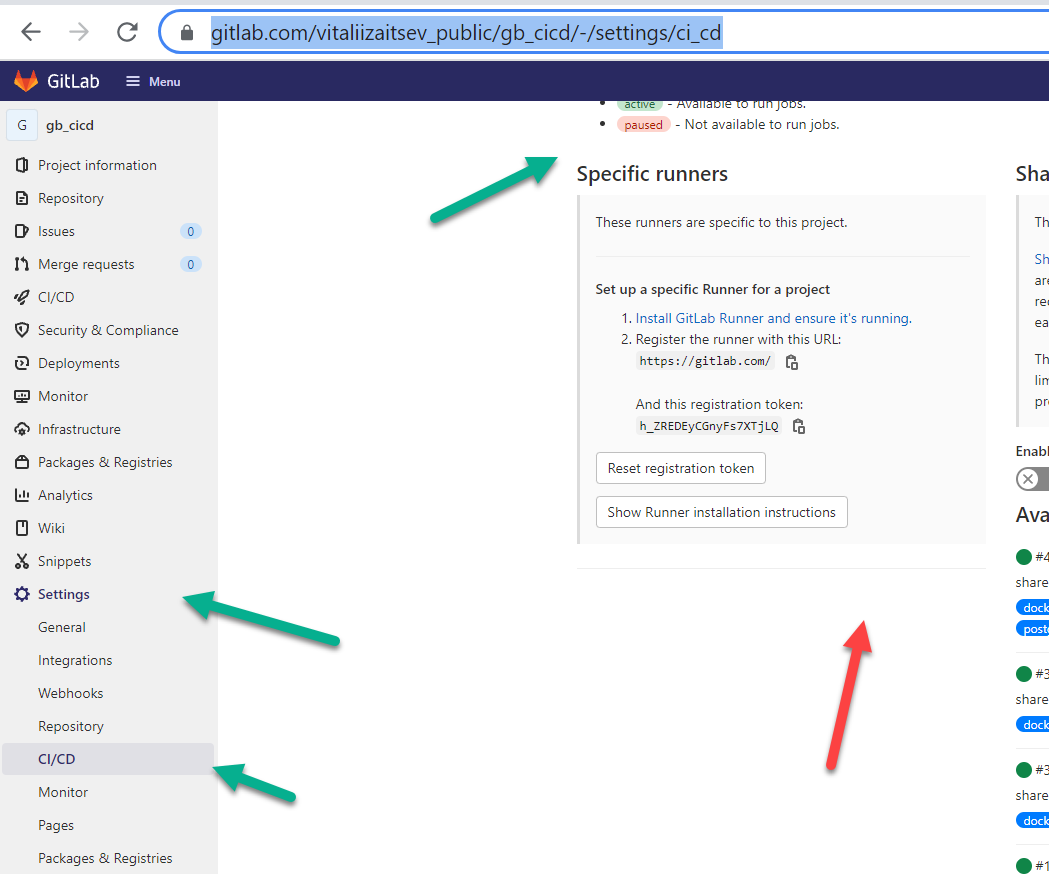


kubectl apply --namespace gitlab -f gitlab-runner.yaml

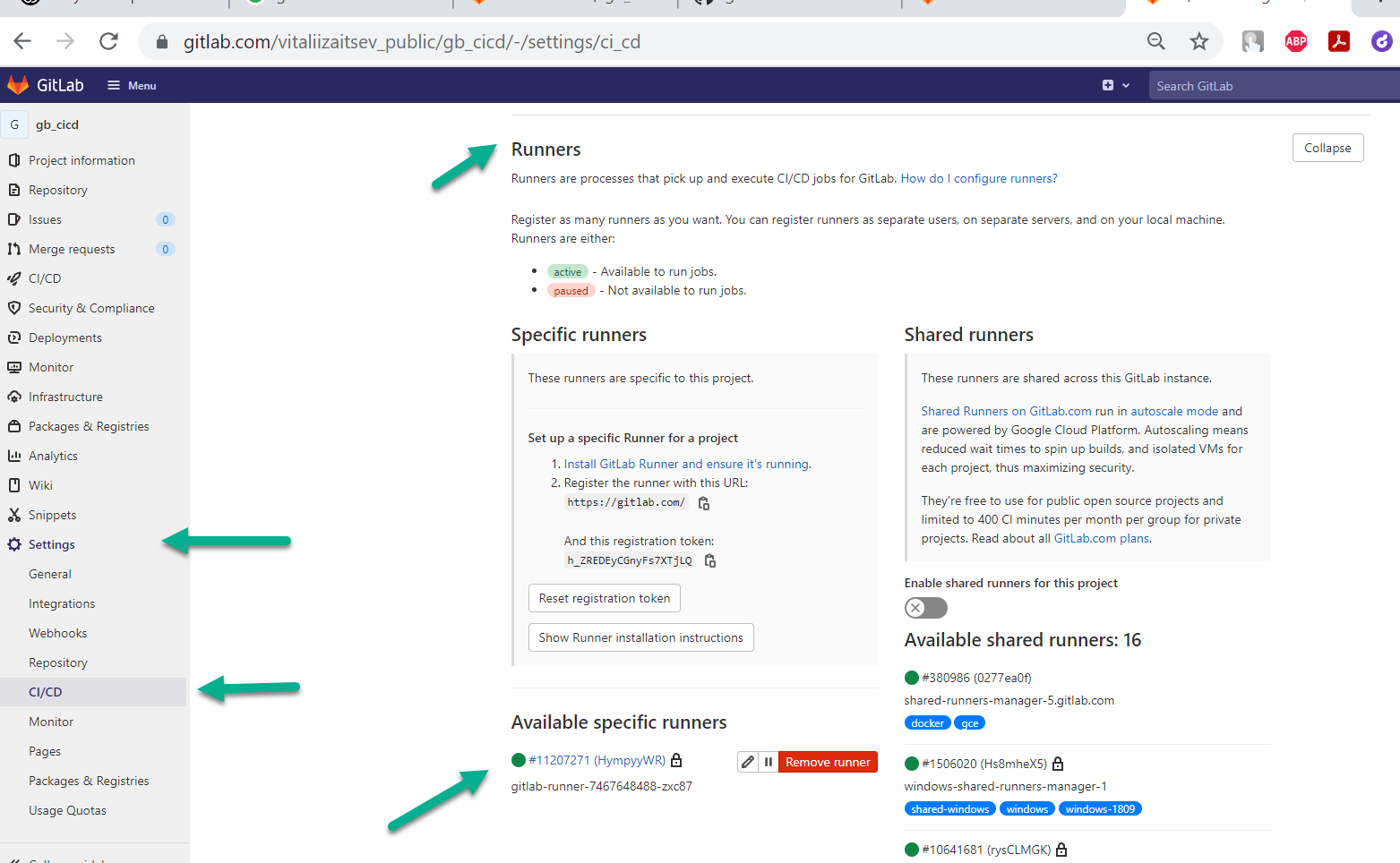


### Проверка появления Runner

<https://gitlab.com/vitaliizaitsev_public/gb_cicd/-/settings/ci_cd>



gitlab-runner-7467648488-zxc87

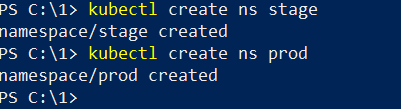


## Содание и конфигурирование namespaces к деплою Runner’ом

### Создание namespaces с именами stage и prod

kubectl create ns stage

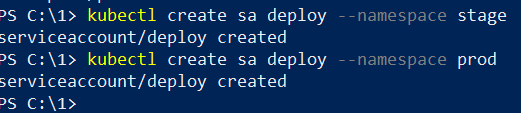
kubectl create ns prod



### Создание авторизационных объектов

kubectl create sa deploy --namespace stage

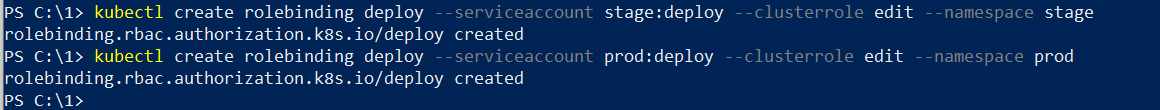
kubectl create sa deploy --namespace prod



### Binking ролей

kubectl create rolebinding deploy --serviceaccount stage:deploy --clusterrole edit --namespace stage

kubectl create rolebinding deploy --serviceaccount prod:deploy --clusterrole edit --namespace prod



### Получение токенов для деплоя в namespaces

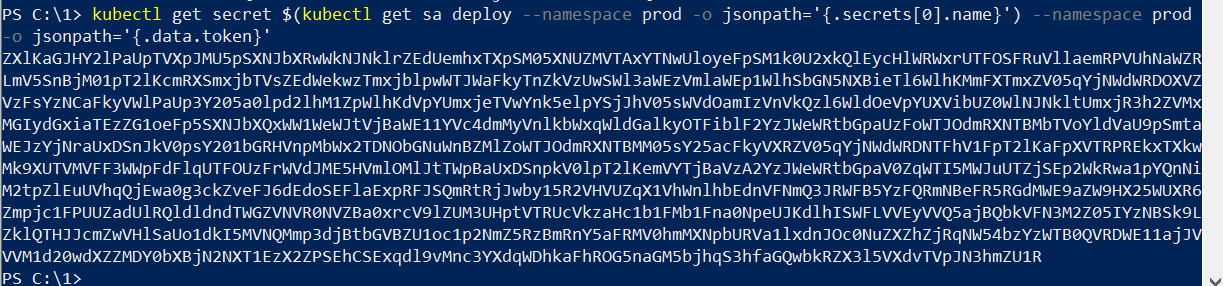
kubectl get secret $(kubectl get sa deploy --namespace stage -o jsonpath='{.secrets[0].name}') --namespace stage -o jsonpath='{.data.token}'





kubectl get secret $(kubectl get sa deploy --namespace prod -o jsonpath='{.secrets[0].name}') --namespace prod -o jsonpath='{.data.token}'

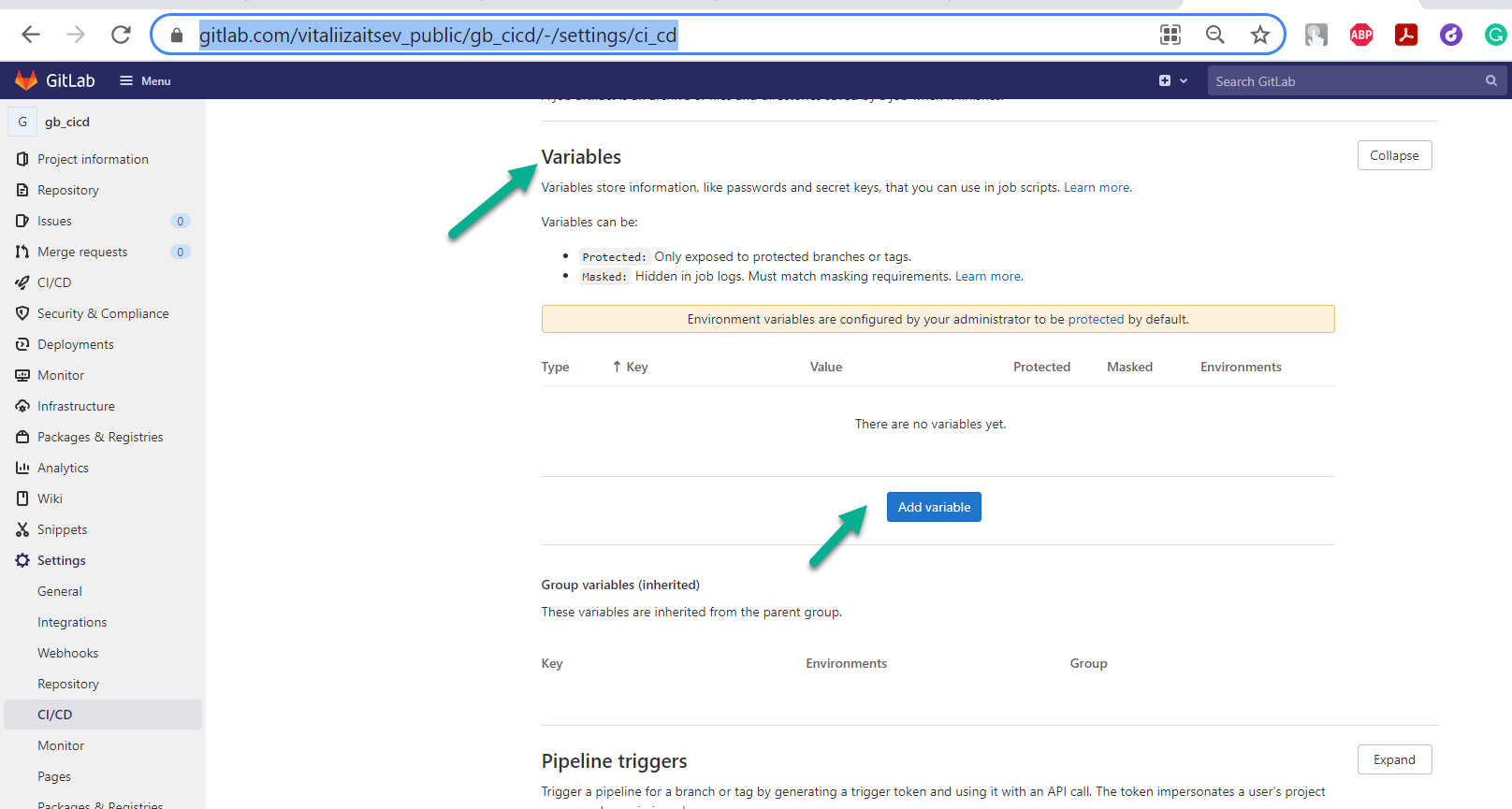




### Создание в GitLab проекте переменных сред stage и prod

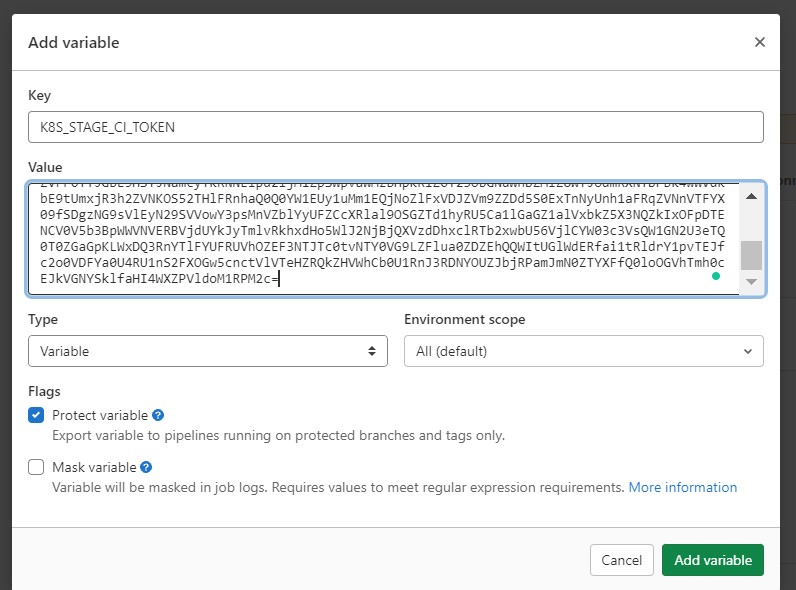
<https://gitlab.com/vitaliizaitsev_public/gb_cicd/-/settings/ci_cd>

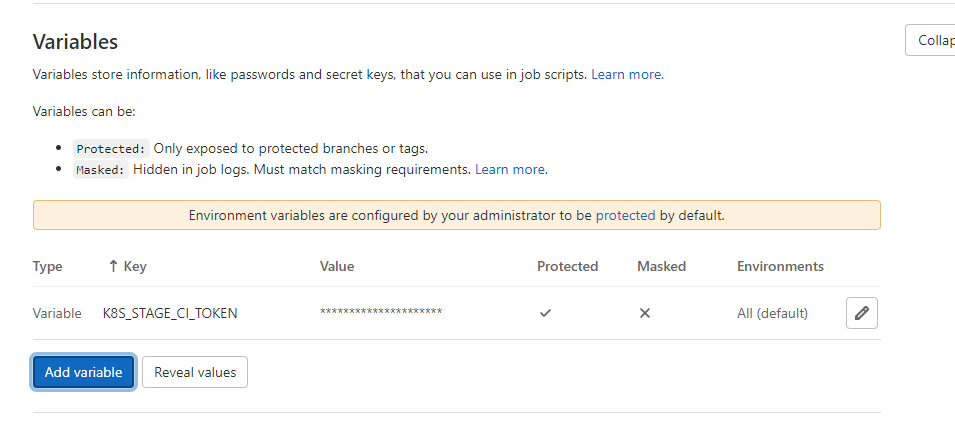
Секция Variables



**K8S\_STAGE\_CI\_TOKEN**

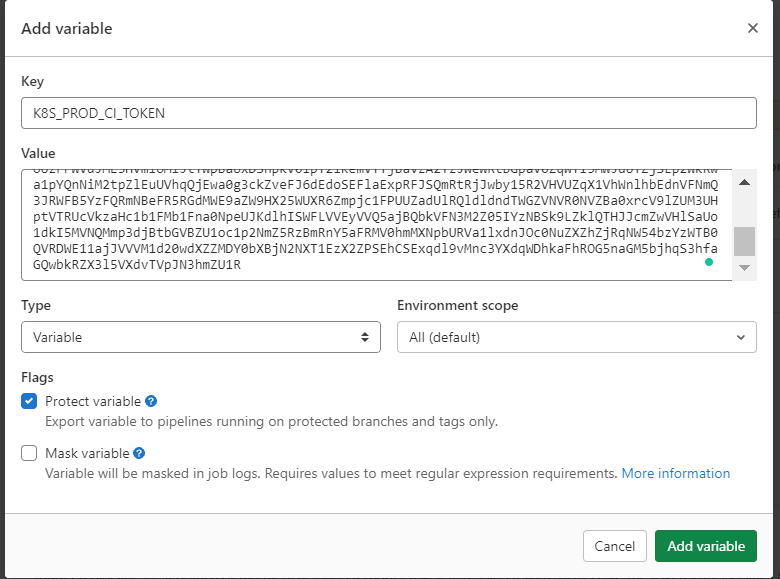


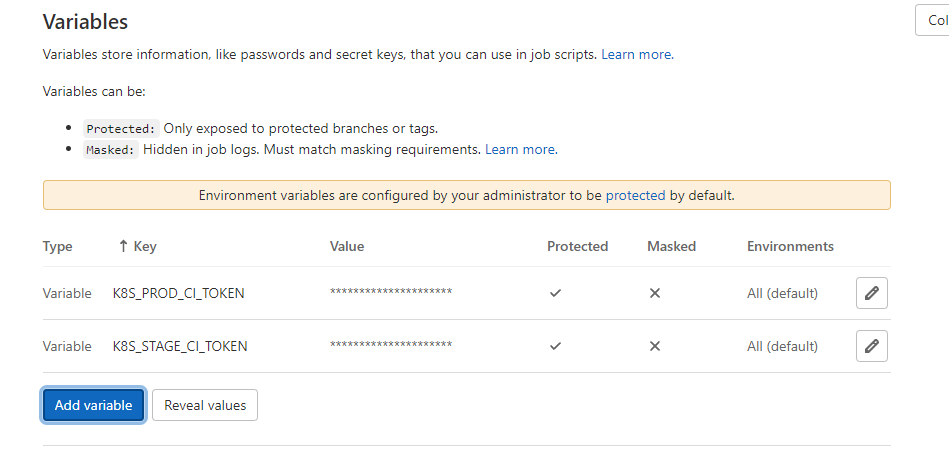




**K8S\_PROD\_CI\_TOKEN**





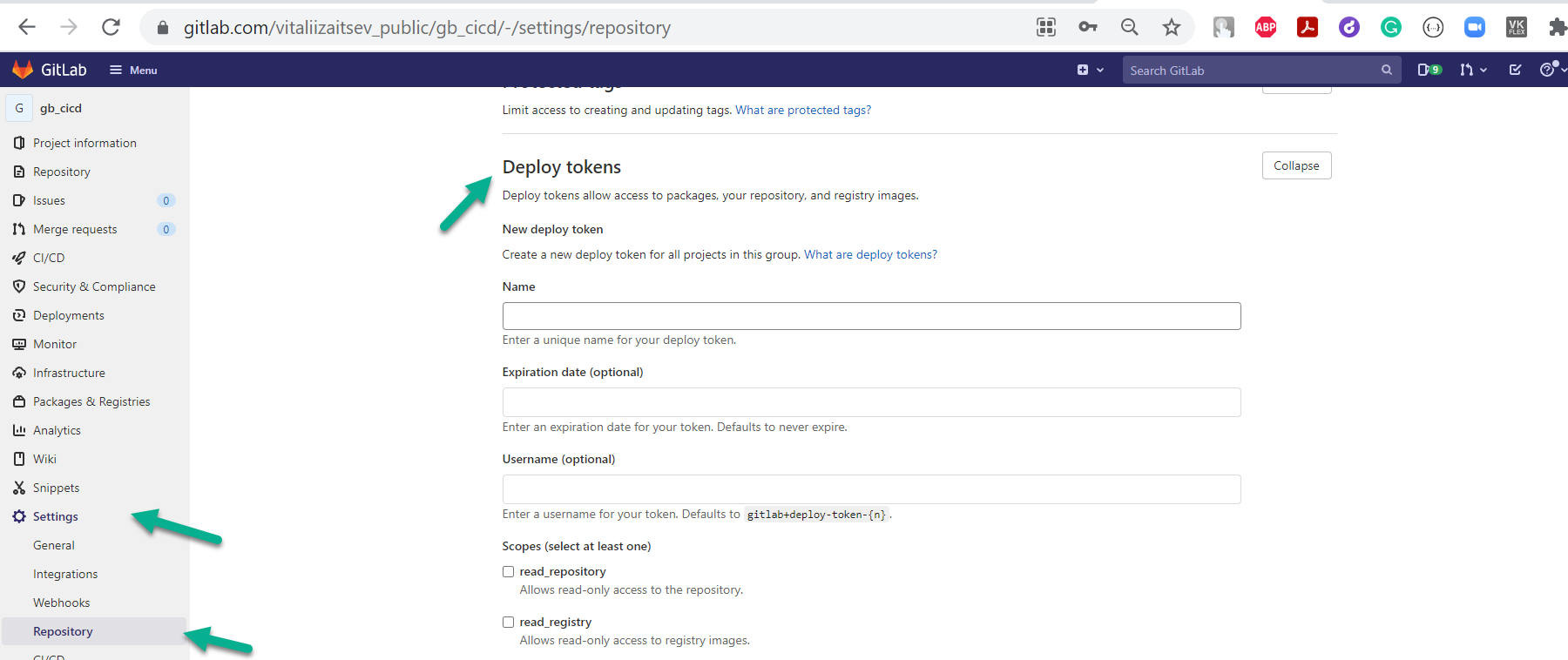


### Создание secrets для авторизации и деплоя Kubernetes в GitLab Registry

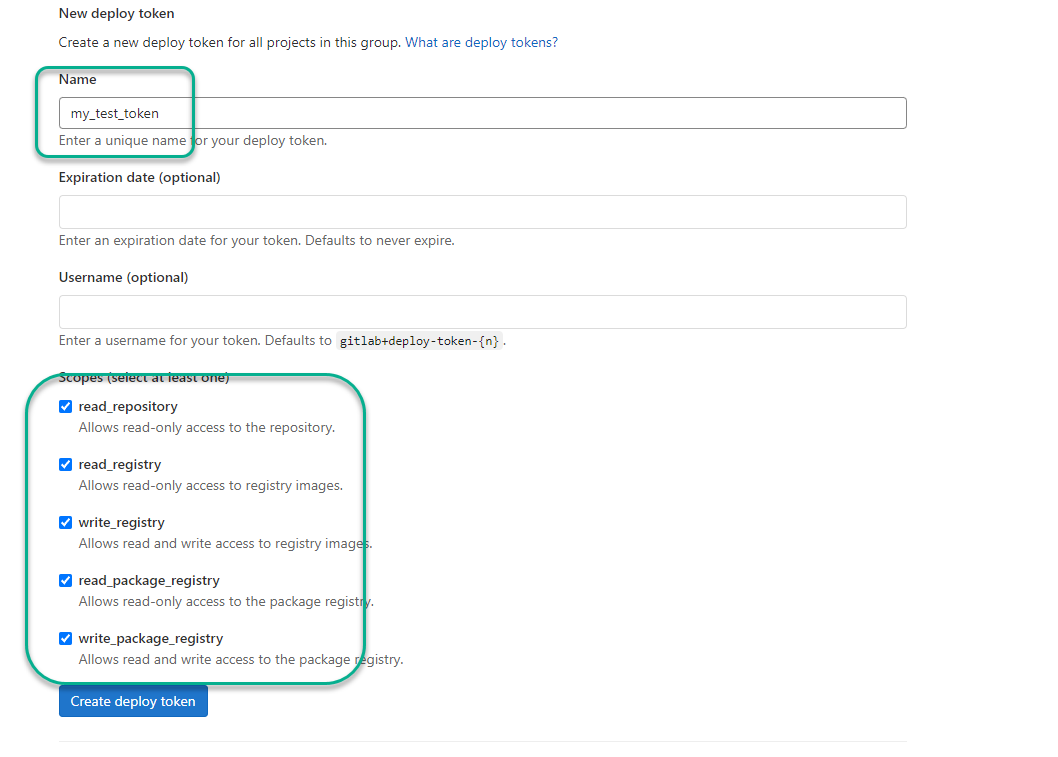
<https://gitlab.com/vitaliizaitsev_public/gb_cicd/-/settings/repository>

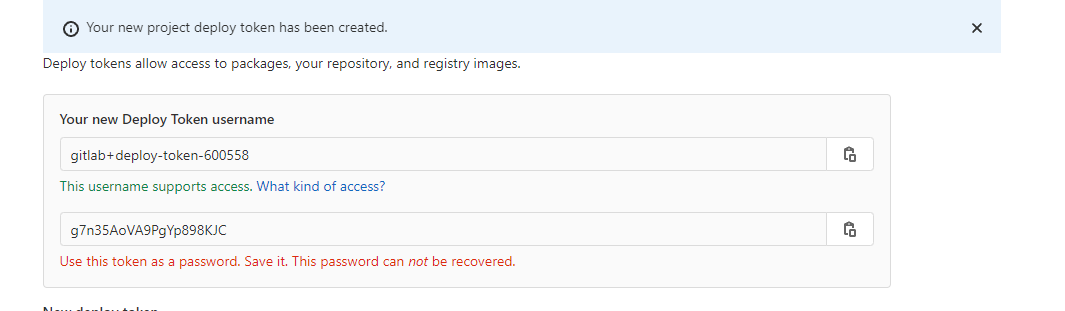
Раздел Settings-> Repository

Секция Deploy tokens



my\_test\_token





Your new Deploy Token username

gitlab+deploy-token-600558

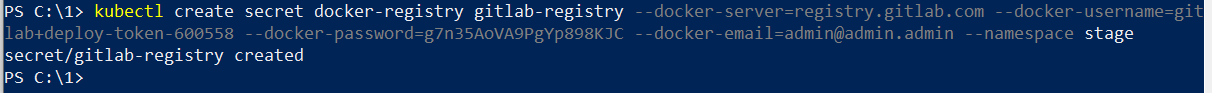
This username supports access. What kind of access?

g7n35AoVA9PgYp898KJC

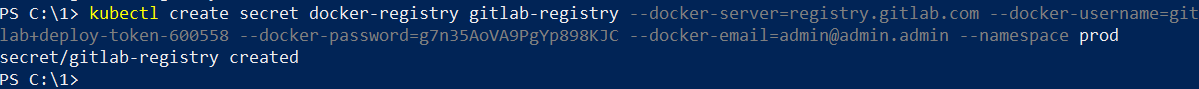
Use this token as a password. Save it. This password can not be recovered.

**Создание secrets**

kubectl create secret docker-registry gitlab-registry --docker-server=registry.gitlab.com --docker-username=gitlab+deploy-token-600558 --docker-password=g7n35AoVA9PgYp898KJC --docker-email=admin@admin.admin --namespace stage



kubectl create secret docker-registry gitlab-registry --docker-server=registry.gitlab.com --docker-username=gitlab+deploy-token-600558 --docker-password=g7n35AoVA9PgYp898KJC --docker-email=admin@admin.admin --namespace prod



Патчинг дефолтного serviceaccount нашими secrets

kubectl patch serviceaccount default -p '{"imagePullSecrets": [{"name": "gitlab-registry"}]}' -n stage

kubectl patch serviceaccount default -p '{"imagePullSecrets": [{"name": "gitlab-registry"}]}' -n prod

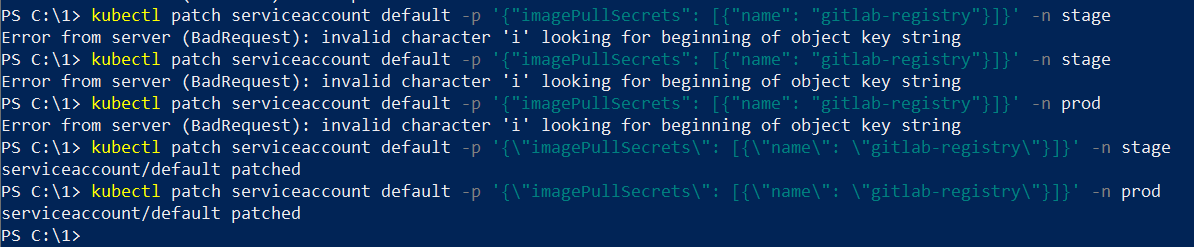
ВНИМАНИЕ!

Для случая Windows Powershell требуется экранировать двойные кавычки!!!

<https://stackoverflow.com/questions/55602559/kubectl-patch-works-on-linux-bash-but-not-in-windows-powershell-ise>

kubectl patch serviceaccount default -p '{ \"imagePullSecrets\": [{ \"name\": \"gitlab-registry\" }]}' -n stage

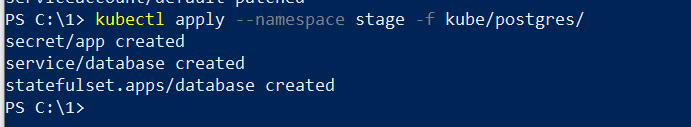
kubectl patch serviceaccount default -p '{ \"imagePullSecrets\": [{ \"name\": \"gitlab-registry\" }]}' -n prod



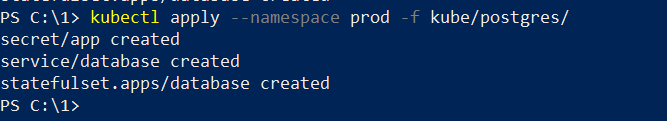
## НЕУСПЕХ! Развёртывание приложений

### Применение манифестов из kube\postgres

kubectl apply --namespace stage -f kube/postgres/



kubectl apply --namespace prod -f kube/postgres/



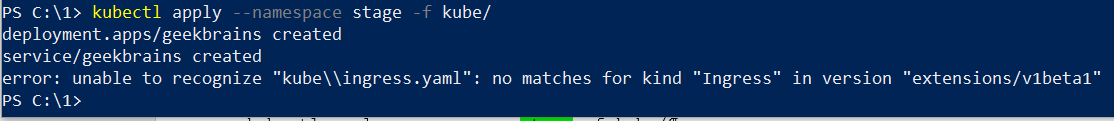
### Корректировка настроек ingress и маппим на stage и prod

Меняем на

rules:

- host: stage

kubectl apply --namespace stage -f kube/



**Корректируем ingress - networking.k8s.iov1.yaml по последним изменениям**

apiVersion: networking.k8s.io/v1

kind: Ingress

metadata:

name: geekbrains

spec:

rules:

- host: stage

http:

paths:

- path: /users

pathType: Prefix

backend:

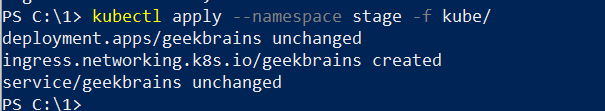
service:

name: geekbrains

port:

number: 8000

kubectl apply --namespace stage -f kube/

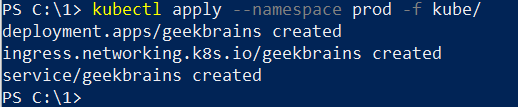


Меняем в ingress - networking.k8s.iov1.yaml на

rules:

- host: prod

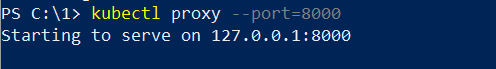
kubectl apply --namespace prod -f kube/



### НЕУСПЕХ! Проверка работы приложения: нет доступа CURL’ом

Делаем loadbalancer доступным извне

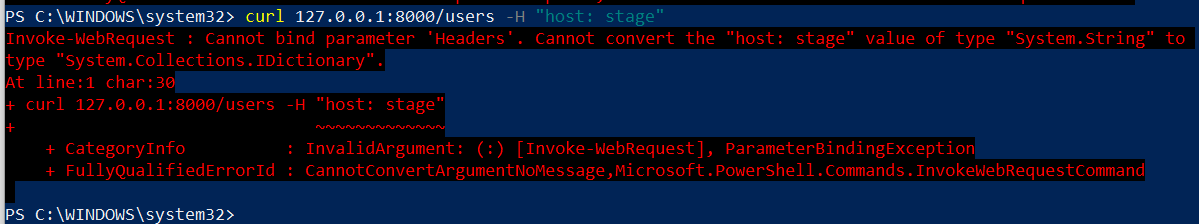
kubectl proxy --port=8000



Запуск другого терминала и подключение к прокси

*curl localhost:8000/users -H "Host: stage" -X POST -d '{"name": "Vasiya", "age": 34, "city": "Vladivostok"}'*

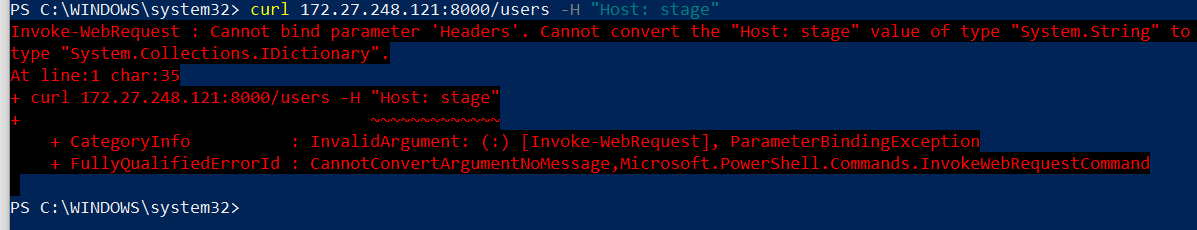
*curl localhost:8000/users -H "Host: stage"*



minikube ip

172.27.248.121

*curl* 172.27.248.121*:8000/users -H "Host: stage"*



CURL’ом НЕ удалось подключиться

## НЕУСПЕХ! Настройка CI: .gitlab-ci.yml

<https://gitlab.com/vitaliizaitsev_public/gb_cicd/-/ci/editor>

variables:

K8S\_API\_URL: https://kubernetes.default

stages:

- test

- build

- deploy

test:

stage: test

image: golang:1.14

script:

- echo OK

build:

stage: build

image: docker:19.03.12

services:

- docker:19.03.12-dind

variables:

DOCKER\_DRIVER: overlay

DOCKER\_HOST: tcp://docker:2375

DOCKER\_TLS\_CERTDIR: ""

before\_script:

- docker login -u $CI\_REGISTRY\_USER -p $CI\_REGISTRY\_PASSWORD $CI\_REGISTRY

script:

- docker build . -t $CI\_REGISTRY\_IMAGE:$CI\_COMMIT\_REF\_SLUG.$CI\_PIPELINE\_ID

- docker push $CI\_REGISTRY\_IMAGE:$CI\_COMMIT\_REF\_SLUG.$CI\_PIPELINE\_ID

.deploy: &deploy

stage: deploy

image: bitnami/kubectl:1.16

before\_script:

- export KUBECONFIG=/tmp/.kubeconfig

- kubectl config set-cluster k8s --insecure-skip-tls-verify=true --server=$K8S\_API\_URL

- kubectl config set-credentials ci --token=$(echo $K8S\_CI\_TOKEN | base64 --decode)

- kubectl config set-context ci --cluster=k8s --user=ci

- kubectl config use-context ci

script:

- sed -i "s,\_\_IMAGE\_\_,$CI\_REGISTRY\_IMAGE:$CI\_COMMIT\_REF\_SLUG.$CI\_PIPELINE\_ID,g" kube/deployment.yaml

- kubectl apply -f kube/ --namespace $CI\_ENVIRONMENT\_NAME

- kubectl rollout status deployment/$CI\_PROJECT\_NAME --namespace $CI\_ENVIRONMENT\_NAME || (kubectl rollout undo deployment/$CI\_PROJECT\_NAME --namespace $CI\_ENVIRONMENT\_NAME && exit 1)

deploy:stage:

<<: \*deploy

environment:

name: stage

variables:

K8S\_CI\_TOKEN: $K8S\_STAGE\_CI\_TOKEN

only:

- master

deploy:prod:

<<: \*deploy

environment:

name: prod

variables:

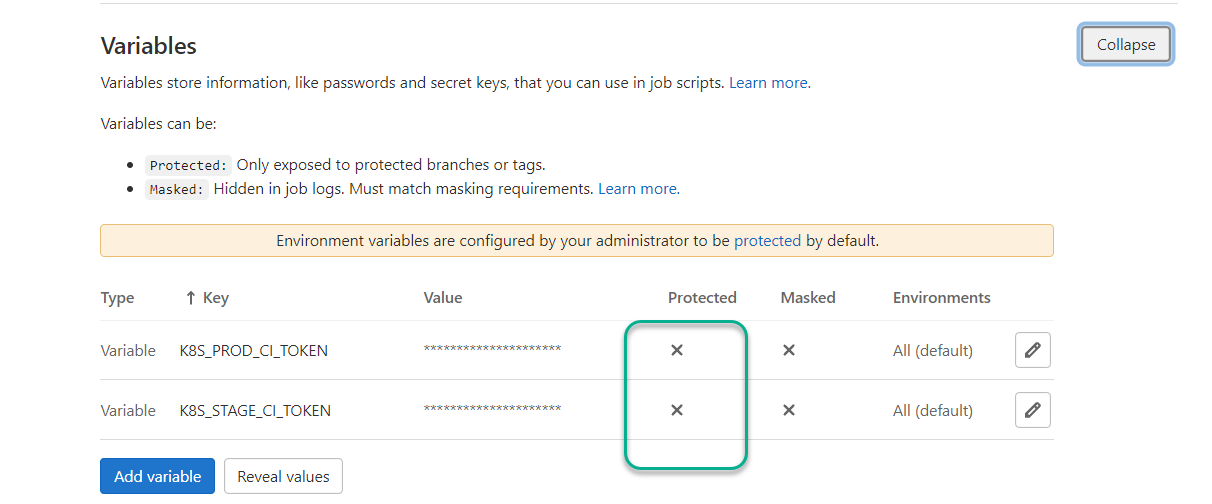
K8S\_CI\_TOKEN: $K8S\_PROD\_CI\_TOKEN

only:

- master

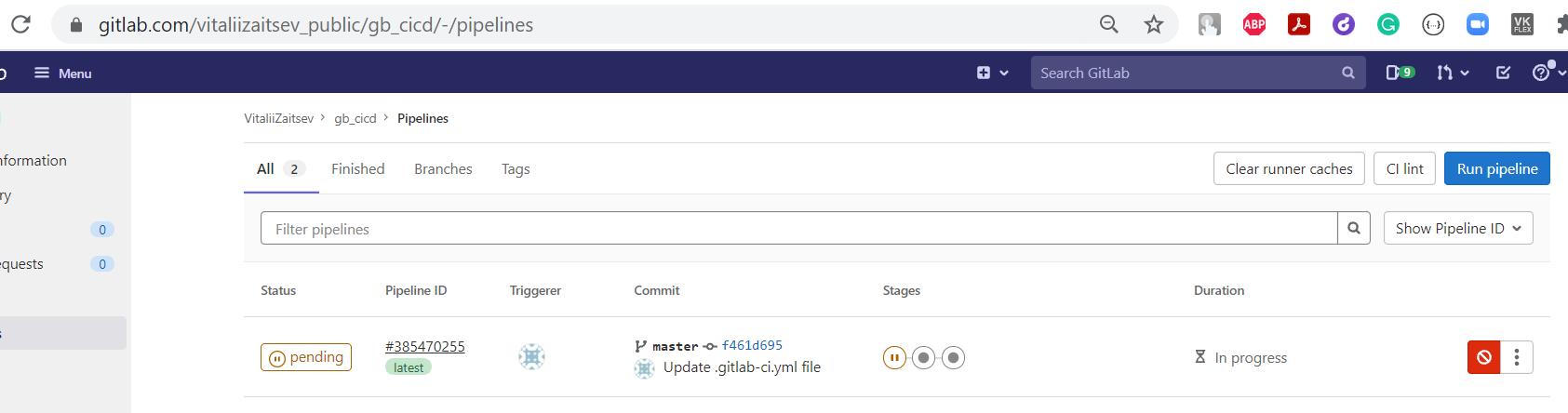
when: manual

Также подсмотрел, снял чекбокс Protected и переменных сред



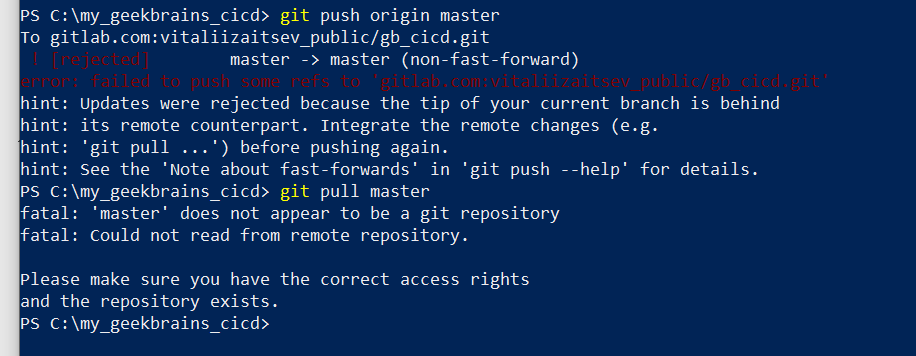
**Неуспех! Pipeline подвис в статусе Pending вручную**

<https://gitlab.com/vitaliizaitsev_public/gb_cicd/-/pipelines>



## НЕУСПЕХ! Перенос Go файлов и коммит на удалённый репозиторий

git push origin master



PS C:\my\_geekbrains\_cicd> git push origin master

To gitlab.com:vitaliizaitsev\_public/gb\_cicd.git

! [rejected] master -> master (non-fast-forward)

error: failed to push some refs to 'gitlab.com:vitaliizaitsev\_public/gb\_cicd.git'

hint: Updates were rejected because the tip of your current branch is behind

hint: its remote counterpart. Integrate the remote changes (e.g.

hint: 'git pull ...') before pushing again.

hint: See the 'Note about fast-forwards' in 'git push --help' for details.

## Само домашнее задание НЕ НАЧАТО!

В итоге до выполнения самого домашнего задания НЕ добрался

1. <https://docs.github.com/en/get-started/getting-started-with-git/managing-remote-repositories#removing-a-remote-repository> [↑](#footnote-ref-1)