# **Задание 1**

1. Разверните GitLab локально, используя Vagrantfile и инструкцию, описанные в [этом репозитории](https://github.com/netology-code/sdvps-materials/tree/main/gitlab).
2. Создайте новый проект и пустой репозиторий в нём.
3. Зарегистрируйте gitlab-runner для этого проекта и запустите его в режиме Docker. Раннер можно регистрировать и запускать на той же виртуальной машине, на которой запущен GitLab.

В качестве ответа в репозиторий шаблона с решением добавьте скриншоты с настройками раннера в проекте.

# **Решение 1**

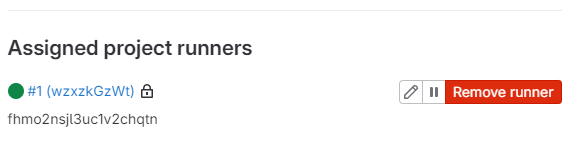
<http://158.160.59.133/netology/netology-gitlab-hw.git>

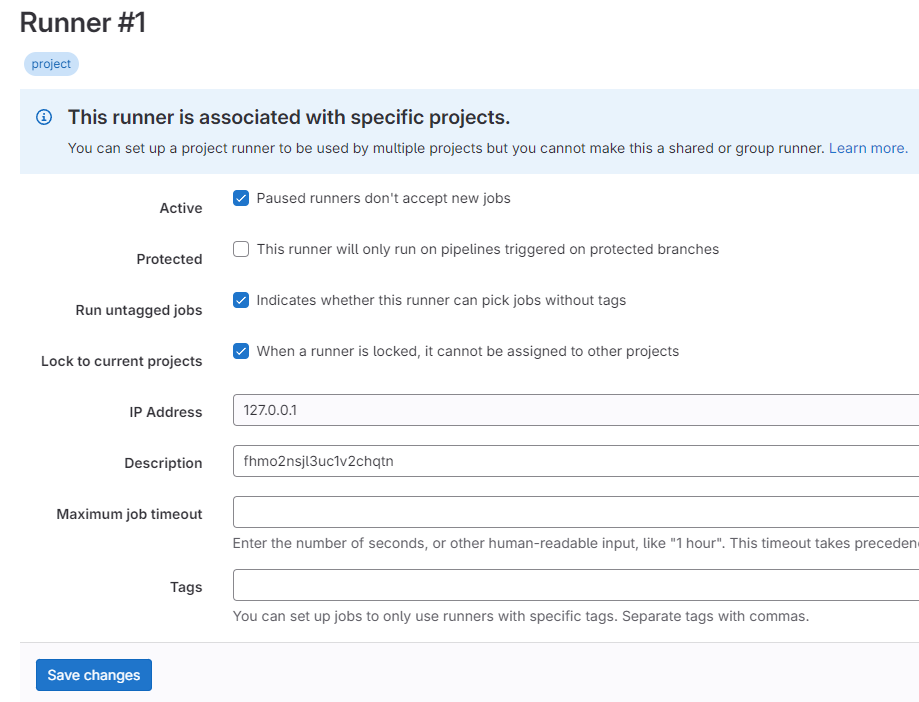
root

nce5WgCLiznmogy+MuAFqIgI6xBY0f0u0bGIg8oe97c=

Gitlab запустил в YC через терраформ, вагрант не хочу.

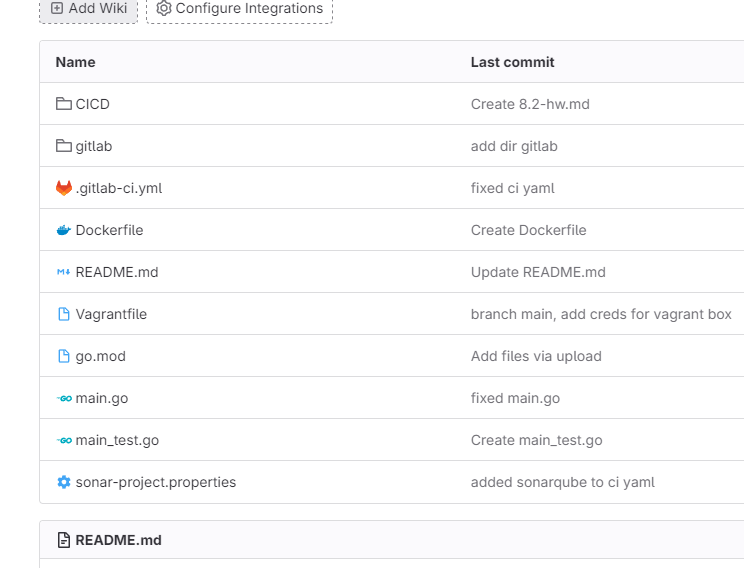
Раннер:





Там же где и гитлаб.

Сам проект:



# **Задание 2**

1. Запушьте [репозиторий](https://github.com/netology-code/sdvps-materials/tree/main/gitlab) на GitLab, изменив origin. Это изучалось на занятии по Git.
2. Создайте .gitlab-ci.yml, описав в нём все необходимые, на ваш взгляд, этапы.

В качестве ответа в шаблон с решением добавьте:

* файл gitlab-ci.yml для своего проекта или вставьте код в соответствующее поле в шаблоне;
* скриншоты с успешно собранными сборками.

# **Решение 2**

Здесь две сборки (1 и 3, вторая ошибочная), где в первой без сонаркуба просто тест и сборка, вторая уже с сонаркубом. Предварительно создал (в соответствии с мануалом, предложенным сонаркубом) переменные в гитлабе. Сонар крутится там же где и гитлаб. Итого:



stages:

- test

- static-analysis

- build

test:

stage: test

image: golang:1.17

script:

- go test .

static-analysis:

stage: test

image:

name: sonarsource/sonar-scanner-cli

entrypoint: [""]

variables:

script:

- sonar-scanner

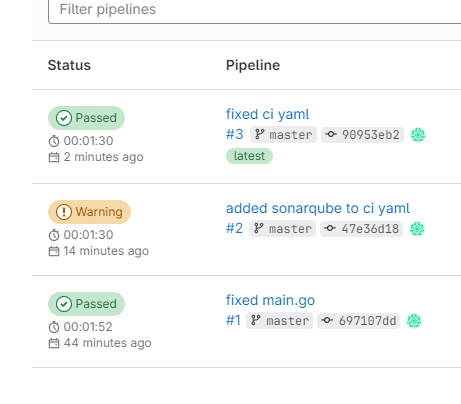
build:

stage: build

image: docker:latest

script:

- docker build .



# **Задание 3**

Измените CI так, чтобы:

* этап сборки запускался сразу, не дожидаясь результатов тестов;
* тесты запускались только при изменении файлов с расширением \*.go.

В качестве ответа добавьте в шаблон с решением файл gitlab-ci.yml своего проекта или вставьте код в соответствующее поле в шаблоне.

# **Решение 3**

stages:

- build

- test

- static-analysis

test:

stage: test

image: golang:1.17

script:

- go test .

only:

changes:

- '\*\*/\*.go'

static-analysis:

stage: test

image:

name: sonarsource/sonar-scanner-cli

entrypoint: [""]

variables:

script:

- sonar-scanner

only:

changes:

- '\*\*/\*.go'

build:

stage: build

image: docker:latest

script:

- docker build .

То есть здесь просто ставим build вперед. Он будет собираться в любом случае. Проверки же будут только в случае если изменились файлы go.