

# NetMafia. Делаем CI CD.

Проект в рамках НИСа "Введение в облачные технологии"

[GitHub](#)

Это веб-приложение на go. Автоматизация, которую я провел:

- Написал Dockerfile для сборки бинарного файла из кода. Сразу прописал запуск бинарного файла. Образ слушает порт 8080.

```
FROM golang:1.23
WORKDIR /app
COPY . .
RUN go mod download
RUN go build -o main
EXPOSE 8080
CMD ["/app/main"]
```

- Написал docker-compose.yml. В нем просто запускаю образ, который буду получать из Dockerfile через запуск пайплайна. Прокинул порт с контейнера на хост, на котором будет запущен контейнер.

```
services:
  app:
    image: mafia:latest
    container_name: mafia-container
    ports:
      - 80:8080
```

- Написал gitlab-ci.yml пайплайн. Решил сделать автоматизацию с помощью инструментов GitLab CI/CD, т.к. большой функционал и я уже был знаком с этой системой. Docker внутри Docker (DinD) нужен для того, чтобы вы могли запускать Docker внутри контейнера GitLab Runner'a, чтобы работать с Docker-образами внутри пайплайна. За счет этого внутри пайплайна, например, в скрипте build, я могу использовать Docker, как будто я работаю на обычной машине. Строим образ mafia из Dockerfile и поднимаем контейнер через docker compose.

```
stages:
  - build
  - deploy
```

```

image: docker
services:
- "docker:dind"

build-job:
stage: build
script: docker build -t mafia .

deploy-job:
stage: deploy
script: docker compose up -d

```

- Создаем [GitLab репозиторий](#). Получаем токен для runner'a (хоста, на котором будет выполняться пайплайн)

### Runner #42124141 project

Property Name	Value
Description	
Paused	No
Protected	No
Can run untagged jobs	Yes
Locked to this project	No
Tags	
Maximum job timeout	
Last contact	49 minutes ago
Version	17.5.2
IP Address	109.252.222.187
Revision	c6eae8d7
Platform	linux
Architecture	amd64

- Не забываем установить и настроить gitlab-runner. В моем случае runner - виртуальная машина на моем ПК.

```

vboxuser@ubuntu7:/srv/gitlab-runner/config$ gitlab-runner status
Runtime platform arch=amd64 os=linux pid=318867 revision=affd9e7d version=17.5.1
gitlab-runner: Service is running
vboxuser@ubuntu7:/srv/gitlab-runner/config$


```

- Пушим все созданные файлы в GitLab репозиторий. Пайплайн должен успешно завершаться, и мы можем посмотреть приложение на <http://localhost:80> на виртуальной машине.

Но команда будет пушить коммиты в GitHub репозиторий, а не в GitLab. Я не захотел разбираться, как делать runner для GitHub Actions. Попробовал сделать зеркалирование (mirroring) с помощью встроенных инструментов GitLab в настройках проекта. Однако почему-то доступно только push mirroring (отправка с GitLab на GitHub репозиторий), а pull нельзя выбрать.

▼ **Mirroring repositories**

Set up your project to automatically push and/or pull changes to/from another repository. Branches, tags, and commits will be synced automatically. [How do I mirror repositories?](#)

**Mirrored repositories**  0

**Add new mirror repository**

Git repository URL

- The repository must be accessible over `http://`, `https://`, `ssh://` or `git://`.
- When using the `http://` or `https://` protocols, please provide the exact URL to the repository. HTTP redirects will not be followed.
- Do not include the username in the URL, use the username field below if required: `https://gitlab.company.com/group/project.git`.
- When using the `ssh://` protocol, please use the following format: `ssh://username@example.com/group/project.git`.
- The update action will time out after 180 minutes. For big repositories, use a clone/push combination.
- Pull mirrors will only create LFS objects if LFS is [enabled for the project](#).
- Push mirrors will only sync LFS objects if LFS is [enabled for the project](#).
- Push mirrors will **not** sync LFS objects over SSH.
- In case of pull mirroring, your user will be the author of all events in the activity feed that are the result of an update, like new branches being created or new commits being pushed to existing branches.

Mirror direction

Authentication method

Username

Как я понял, эта функция доступна только с подпиской. Тогда я решил сделать пайплайн для GitHub Actions, который будет делать пуш в GitLab репозиторий. Благо, уже есть [готовый модуль](#) для этой цели.

- Написал такой пайплайн в `/workflows/main.yml` в GitHub репозитории `name: Mirror and run GitLab CI`

```
on: [push]

jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v1
      - name: Mirror + trigger CI
        uses: SvanBoxel/gitlab-mirror-and-ci-action@master
        with:
          args: "https://gitlab.com/sausage2/NetMafia.git"
          env:
            FOLLOW_TAGS: "false"
            FORCE_PUSH: "true"
            GITLAB_HOSTNAME: "gitlab.com"
            GITLAB_USERNAME: "c0balt6"
```


```
GITLAB_PASSWORD: ${ secrets.GITLAB_PASSWORD }
```


```
GITLAB_PROJECT_ID: "62918100"
```





```
GITHUB_TOKEN: ${ secrets.HUB_TOKEN }
```

- Не забываем добавить в настройках репозитория в secrets токен доступа GitLab в переменную GITLAB\_PASSWORD и токен доступа GitHub в переменную HUB\_TOKEN. Меняю под себя GITLAB\_USERNAME и GITLAB\_PROJECT\_ID, который можно скопировать в настройках проекта в GitLab.
- После запуска этого workflow в консолике вывода Actions вышла ошибка о невозможности делать force push в protected branch. Поставил такую галочку в настройках репозитория в GitLab:

☒ **Protected branches**  
Keep stable branches secure and force developers to use merge requests. [What are protected branches?](#)

 Giving merge rights to a protected branch also gives elevated permissions for certain CI/CD features. [What are the security implications?](#)

**Protected branches**  1  
By default, protected branches restrict who can modify the branch. [Learn more.](#) Add protected branch

Branch	Allowed to merge	Allowed to push and merge	Allowed to force push 
main 	Maintainers 	Maintainers 	<input checked="" type="checkbox"/> <span>Unprotect</span>

**Готово! Теперь после пуша коммита в GitHub репозиторий запускается workflow, выполняется пайплайн main.yml в ходе которого делается пуш в GitLab репозиторий, запускается пайплайн gitlab-ci.yml. Runner установлен на локальной виртуальной машине. Собирается docker образ, и из этого образа запускается контейнер на 80 порту хоста.**