

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ  
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

**Дисциплина: Бек-энд разработка**

Отчет

Домашнее задание №6: Автодеплой приложения на удалённый  
сервер с использованием Github Actions

Выполнила:  
Пронина Александра  
К33392  
Проверил:  
Добряков Д. И.

Санкт-Петербург  
2024 г.

## Задание:

Необходимо настроить автодеплой (с триггером на обновление кода в вашем репозитории, на определённой ветке) для вашего приложения на удалённый сервер с использованием Github Actions или Gitlab CI (любая другая CI-система также может быть использована).

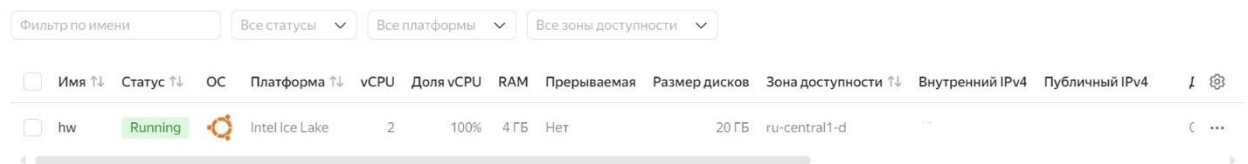
Шаги для настройки автодеплойа:

### 1. Настройка репозитория

Убедимся, что репозиторий содержит необходимый код и конфигурационные файлы для развертывания. Например, Dockerfile, docker-compose.yml, или скрипты для деплоя.

### 2. Подготовка удаленного сервера:

Я арендовала на квоту Yandex Cloud(IP скрыт).



Установим необходимое ПО, такое как Docker, ssh, и другие зависимости для приложения.

Настроим доступ по SSH для безопасного соединения CI/CD системы с сервером.

Создание SSH-ключей:

После создания директории (если её не было) повторим команду для создания ключей:

```
Администратор: Командная строка
Enter file in which to save the key (C:\Users\Елена\.ssh\id_rsa): ~/.ssh/id_rsa
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Saving key "~/.ssh/id_rsa" failed: No such file or directory

C:\Windows\system32>mkdir C:\Users\Елена\.ssh
A subdirectory or file C:\Users\Елена\.ssh already exists.

C:\Windows\system32>ssh-keygen -t rsa -b 4096 -C "alex20030708@gmail.com" -f C:\Users\Елена\.ssh\id_rsa
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:\Users\Елена\.ssh\id_rsa.
Your public key has been saved in C:\Users\Елена\.ssh\id_rsa.pub.
The key fingerprint is:
SHA256:VRUtJhnbghUF4YIe+IAocrSelPU2DRIahTyYefnNzR4 alex20030708@gmail.com
The key's randomart image is:
+---[RSA 4096]-----+
|oBoo=..  B0+o |
|. +B + + . =ooo .
|o.B+.o* + + +o.. |
|..o.o.E= o . . |
| o . .S |
| . |
+-----[SHA256]-----+
C:\Windows\system32>
```

```
C:\Windows\system32>type C:\Users\Елена\.ssh\id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQKD2fcd45TKem9jHFLpFPL8QuCkLI9MF2gg2ZL+XatrWx31F2ZCGaswBEi0Dd2wZ26qallsyYSicGIrWw
fLhs0NSDgPjjoyckgMwc+Z1lk9W441Xu3CLUQMr8sGfEVM6BD9YazB7qE3SDfFBwtZ5aN0jfSh4QTG4Wtw05Y3wGpTdLsc7cnVD0anUj4Iuk1jLa5ZoTQ0GG
IIiHxXd5pVzQs3vPLtj1l9ozR3Ls+BBTnGMIrHiser1NSXho8GJM8winLQu3gnPx7vJRvYuxAOU6tXIVHvYyECI/Y37QUdoDaZ/wPhGuGn1Tt8YwYJuryt
fgGr9cwhIJe/1SfrAPnpV0vsrjAgB0RiUiJQe1+4uhNBUu82umLLuP+bNpnIVfuXnKwqbTR9PTS2kQOp6AxsZVDp+oXY70HRTgLmaBk/j0G1UUp12P7OGLDy
TtKNG+4V5/61/PqDrvJWqQwhuB0scAqWQQZvr6A5+AmPMWEL9TLXBXQD61v4LxV91ELVqqn0/5IL8AvAHgwWf5n9naokgLtJOek0SN6984008sLpt5QeNLvs
9V1V/4wPB27a93+QyK8yvuwR0QPbM2E6eKWumt3PYV5BuydBCHJc7cdDTjwHE69G0E41bVJHORN1Hs1mUNsArNsRs1DQEpzBraF9MMjPxYQM7BwP8IYB6E/
a0Q== alexp20030708@gmail.com

C:\Windows\system32>
```

Затем добавим этот ключ в файл ~/.ssh/authorized\_keys на сервере. Для этого выполним через PowerShell следующие шаги:

Подключаемся к серверу по SSH (если уже есть доступ):

```
ssh <тут юзер>@<тут api сервера>
```

Создадим директорию .ssh, если она не существует:

```
mkdir -p ~/.ssh
```

Для дальнейшего выполнения у меня было 2 варианта (но я выбрала первый из-за удобства в частном случае):

Вариант 1: добавим публичный ключ в файл authorized\_keys:

```
echo " " >> ~/.ssh/authorized_keys
```

Вариант 2: Или с использованием SCP:

```
scp C:\Users\Елена\.ssh\id_rsa.pub <тут юзер>@<тут api сервера>:~/.ssh/temp_id_rsa.pub
```

```
ssh <тут юзер>@<тут api сервера>: ~/.'cat ~/.ssh/temp_id_rsa.pub >> ~/.ssh/authorized_keys
&& rm ~/.ssh/temp_id_rsa.pub'
```

```
PS C:\Windows\system32> dism /Online /Add-Capability /CapabilityName:OpenSSH.Server~~~~0.0.1.0

Система DISM
Версия: 10.0.19041.844

Версия образа: 10.0.19042.1288

[=====100.0%=====]

Администратор: Windows PowerShell
PS C:\Windows\system32>
PS C:\Windows\system32>

Operation
Running
[ooooooooooooooooooooo]
```

Добавим приватный ключ в GitHub Secrets:

Откроем репозиторий на GitHub и выполним следующие шаги:

1. Перейдём в раздел Settings.
2. Выберите Secrets and variables > Actions.
3. Нажмём на кнопку New repository secret.
4. Добавим следующие секреты:

SSH\_PRIVATE\_KEY: Содержимое файла C:\Users\Елена\.ssh\id\_rsa. Для этого откроем файл id\_rsa и скопируем его содержимое:

type C:\Users\Елена\.ssh\id\_rsa – в cmd в роли администратора

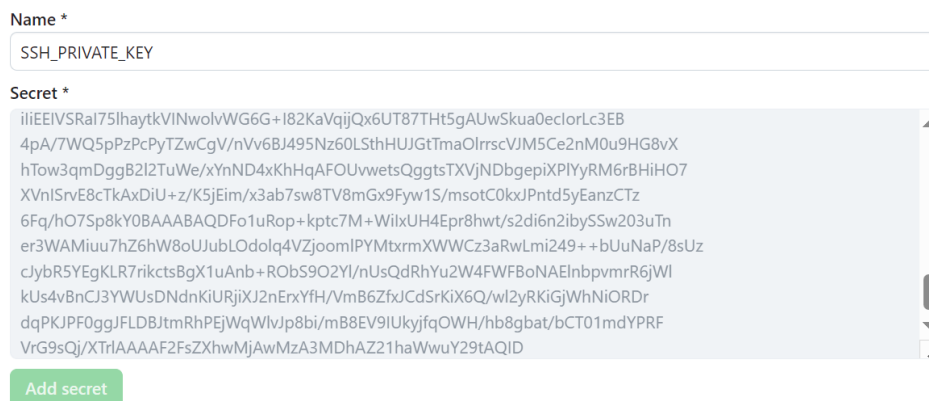
SSH\_HOST: IP-адрес сервера.

SSH\_USER: Имя пользователя для подключения через SSH.

3. Настроим workflow в GitHub Actions, добавим команды для перезапуска контейнеров
- Создадим файл workflow в репозитории (например, .github/workflows/deploy.yml):

SSH\_PRIVATE\_KEY: Содержимое файла C:\Users\Елена\.ssh\id\_rsa.

#### Actions secrets / New secret



Name \*

SSH\_PRIVATE\_KEY

Secret \*

iliEEIVSRal75lhaytkVINwolvWG6G+I82KaVqjQx6UT87THt5gAUwSkua0eclorLc3EB  
4pA/7WQ5pPzPcPyTZwCgV/nVv6BJ495Nz60LStHhUJGtTmaOIrrscVJM5Ce2nM0u9HG8vX  
hTow3qmDggB2l2TuWe/xYnND4xKhHqAFOUvwetsQggsTXVjNDbgepiXPIYyRM6rBHiHO7  
XVnlSrvE8cTkAxDiU+z/K5jEim/x3ab7sw8TV8mGx9Fyw1S/msotC0kdPntd5yEanzCTz  
6Fq/hO7Sp8kY0BAABAQDFo1uRop+kptc7M+WilxUH4Epr8hwt/s2di6n2ibySSw203uTn  
er3WAMiuu7hZ6hW8oUJubLOdolq4VZjoomlPYMtxrmXWWCz3aRwLmi249++bUuNaP/8sUz  
cJybR5YEgKLR7rikctsBgX1uAnb+RObS9O2Yl/nUsQdRhYu2W4FWFBoNAElbpmrR6jWl  
kUs4vBnCJ3YWUsDNdnKiURjiXJ2nErXyfh/VmB6ZfxCdSrKiX6Q/wl2yRKiGjWhNiORDr  
dqPKJPF0ggJFLDBJtmRhPEjWqWlvJp8bi/mB8EV9lUkyjfqOWH/hb8gbat/bCT01mdYPRF  
VrG9sQj/XtRlAAAAF2FsZXhwMjAwMzA3MDhAZ21haWwuy29tAQID

Add secret

SSH\_HOST: IP-адрес вашего сервера. (скрыто)

SSH\_USER: Имя пользователя для подключения через SSH. (скрыто)

4. Настройка workflow в GitHub Actions:

Далее создали файл workflow в репозитории (.github/workflows/deploy.yml):

SSH\_PRIVATE\_KEY, SSH\_HOST, secrets.SSH\_USER – секретники для деплоя на сервер.

Содержимое: `name: Deploy`

```
on:
  push:
    branches:
      - main # укажем нужную ветку

jobs:
  deploy:
    runs-on: ubuntu-latest

    steps:
```

```

- name: Checkout repository
  uses: actions/checkout@v2

- name: Set up SSH
  uses: webfactory/ssh-agent@v0.5.3
  with:
    ssh-private-key: ${ secrets.SSH_PRIVATE_KEY }

- name: Deploy to server
  env:
    SSH_HOST: ${ secrets.SSH_HOST }
    SSH_USER: ${ secrets.SSH_USER }
  run: |
    ssh -o StrictHostKeyChecking=no $SSH_USER@$SSH_HOST << 'EOF'
    cd /path/to/app
    git pull
    docker-compose down
    docker-compose up -d
  EOF

```

Теперь, при каждом коммите в указанную ветку (в данном случае main), GitHub Actions будет автоматически запускать деплой приложения на удалённый сервер.

**Вывод:** В процессе выполнения лабораторной работы была настроена автоматическая развертка приложения на удаленный сервер с использованием GitHub Actions. Эта развертка срабатывает при обновлении кода в репозитории, определенной на определенной ветке.