**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

# **Дисциплина:** Бэк-энд разработка

Отчет

Практическая работа №6

Выполнил:

Крохин Владимир

БР1.1

Проверил:

Добряков Д. И.

Санкт-Петербург

2025 г.

**Задача**

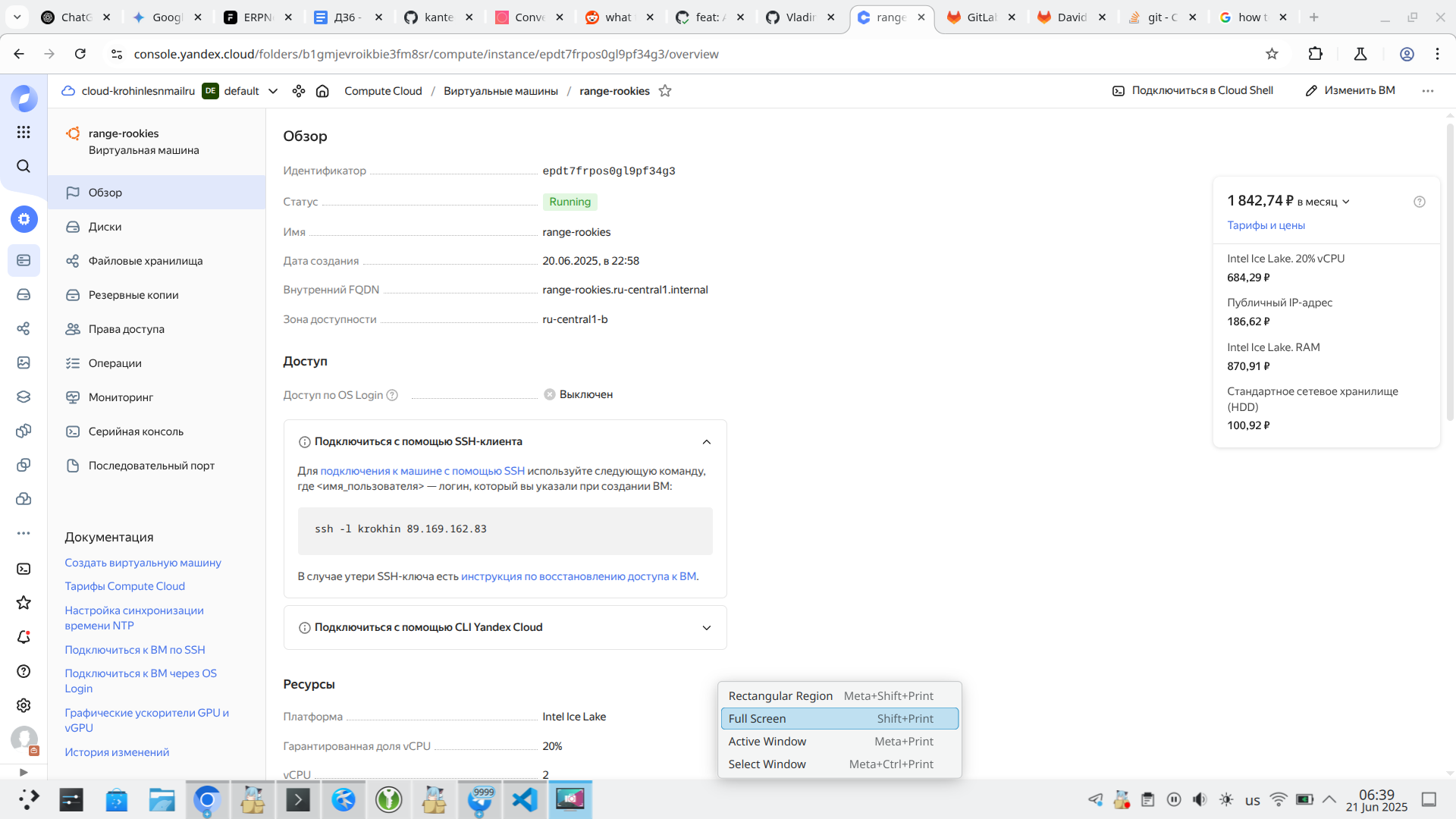
Необходимо настроить автодеплой (с триггером на обновление кода в вашем репозитории, на определённой ветке) для вашего приложения на удалённый сервер с использованием Github Actions или Gitlab CI (любая другая CI-система также может быть использована).

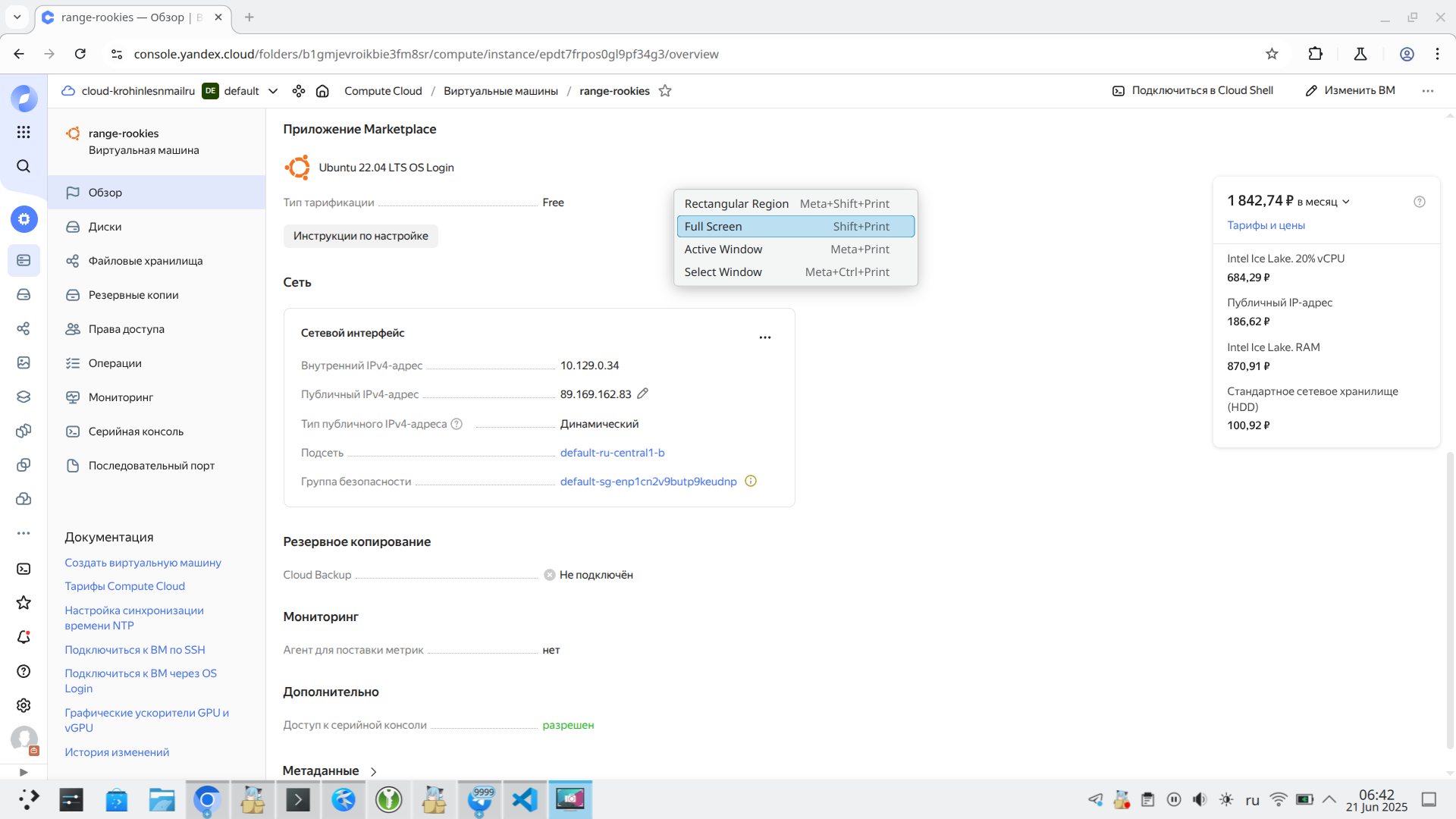
**Ход работы**

В ходе выполнения домашнего задания был создан платежный аккаунт на Yandex.Cloud и создана виртуальная машина на основе образа Ubuntu 22.04 со следующей конфигурацией:

* Intel Ice Lake. 20% vCPU
* Публичный IP-адрес (динамический)
* Intel Ice Lake. RAM (4 ГБ)
* Стандартное сетевое хранилище (HDD) (32 ГБ)

Средства для выполнения домашнего задания использовались из выделенной квоты.





При настройке ВМ был указан публичный SSH ключ. С его помощью происходило подключение.

После первого подключения к серверу, были установлены все необходимые зависимости с помощью команд:

sudo apt-get update

sudo apt-get install -y ca-certificates curl gnupg nginx

sudo install -m 0755 -d /etc/apt/keyrings

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg

sudo chmod a+r /etc/apt/keyrings/docker.gpg

sudo apt-get update

sudo apt-get install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin docker-compose

sudo usermod -aG docker $USER

а также была создана директория по пути /var/www/apps/, создана группа web, к текущему пользователю была добавлена группа web, права (владение) на созданную директорию были переданы текущему пользователю и группе web.

Был совершен перезаход в сессию, чтобы установить новую группу пользователю.

При новом перезаходе в ssh были проброшены ключи с помощью специальных ключей к команде, на хосте был склонирован репозиторий при помощи команды:

git clone [git@github.com](mailto:git@github.com):VladimirKrokhin/rangerookies.git

В проекте заранее в корневой папке была создана директория .github с поддиректорией .workflows внутри которой был создан файл auto-deploy.yml со следующим содержимым:

name: Автодеплой

on:

push:

branches: [ main ]

jobs:

deploy:

name: Деплой

runs-on: self-hosted

steps:

- name: Копирование кода

uses: actions/checkout@v4

with:

fetch-depth: 0

- name: Подготовка директории

run: |

sudo mkdir -p /var/www/apps/rangerookies

sudo chown -R $USER:$USER /var/www/apps/rangerookies

rsync -av --delete ./ /var/www/apps/rangerookies/

- name: Создание переменных окружения

run: |

cd /var/www/apps/rangerookies

echo "${{ secrets.AUTH\_ENV }}" > auth-service/.env

echo "${{ secrets.TRAINING\_ENV }}" > training-service/.env

echo "${{ secrets.REFERENCE\_ENV }}" > reference-service/.env

echo "${{ secrets.NOTES\_ENV }}" > notes-service/.env

- name: Остановка старых контейнеров

run: |

cd /var/www/apps/rangerookies

docker-compose down --remove-orphans

- name: Сборка и запуск

run: |

cd /var/www/apps/rangerookies

echo "Сборка образов..."

docker-compose build

echo "Запуск всех сервисов..."

docker-compose up -d

echo "Ожидание готовности всех сервисов..."

timeout 600 bash -c '

while docker-compose ps | grep -q "unhealthy\|starting"; do

echo "Некоторые сервисы еще не готовы, ждем..."

sleep 30

done

echo "Все сервисы готовы!"

docker-compose ps

'

- name: Проверка деплоя

run: |

cd /var/www/apps/rangerookies

echo "Проверка деплоя..."

docker-compose ps

unhealthy\_count=$(docker-compose ps | grep -c "unhealthy" || true)

if [ "$unhealthy\_count" -gt 0 ]; then

echo "Найдено $unhealthy\_count неготовых сервисов"

docker-compose logs --tail=50

exit 1

fi

echo "Все сервисы готовы и работают!"

Подсле этого была произведена установка и настройка GitHub Actions Runner:

Установка происходила согласно указаниями по ссылке <https://github.com/VladimirKrokhin/rangerookies/settings/actions/runners/new>:

## **Download**

# Create a folder

$ mkdir actions-runner && cd actions-runner

# Download the latest runner package

$ curl -o actions-runner-linux-x64-2.325.0.tar.gz -L https://github.com/actions/runner/releases/download/v2.325.0/actions-runner-linux-x64-2.325.0.tar.gz

# Optional: Validate the hash

$ echo "5020da7139d85c776059f351e0de8fdec753affc9c558e892472d43ebeb518f4 actions-runner-linux-x64-2.325.0.tar.gz" | shasum -a 256 -c

# Extract the installer

$ tar xzf ./actions-runner-linux-x64-2.325.0.tar.gz

## **Configure**

# Create the runner and start the configuration experience

$ ./config.sh --url https://github.com/VladimirKrokhin/rangerookies --token XXX

Copied!

# Last step, run it!

$ ./run.sh

В качестве операционной системы был выбран Linux, системной архитектуры - x64. Настройка в скрипте .config происходила по умолчанию за исключением рабочей директории - была выбрана директория workflows проекта. Был запущен при помощи сервиса:

sudo ./svc.sh install

sudo ./svc.sh start

sudo ./svc.sh status

По ссылке <https://github.com/VladimirKrokhin/rangerookies/settings/secrets/actions> были указаны переменные окружения проекта (креды доступа к БД, адреса API авторизации, адрес RabbitMQ и пр.).

После настройки и запуска раннера, была произведена настройка и запуск Nginx:

был создан файл по пути /etc/nginx/sites-available/default со следующим содержимым:

server {

listen 80 default\_server;

listen [::]:80 default\_server;

# Путь для логов

**access\_log** /var/log/nginx/rangerookies.access.log;

**error\_log** /var/log/nginx/rangerookies.error.log;

# GZIP Конфигурация для производительности

**gzip** on;

**gzip\_disable** "msie6";

**gzip\_vary** on;

**gzip\_proxied** any;

**gzip\_comp\_level** 6;

**gzip\_types** text/plain text/css **application**/json **application**/javascript text/xml **application**/xm

l **application**/xml+rss text/javascript;

# Тайм-ауты

**proxy\_connect\_timeout** 30s;

**proxy\_send\_timeout** 30s;

**proxy\_read\_timeout** 30s;

**send\_timeout** 30s;

# Основной прокси к API Gateway

location / {

# Адрес, на который проксируем запросы

# (docker-compose.yml пробрасывает порт 8080 контейнера на 80 хоста)

proxy\_pass http://127.0.0.1:8080;

# Стандартные заголовки для reverse-proxy

**proxy\_set\_header** Host **$host**;

**proxy\_set\_header** X-Real-IP **$remote\_addr**;

**proxy\_set\_header** X-Forwarded-For **$proxy\_add\_x\_forwarded\_for**;

**proxy\_set\_header** X-Forwarded-Proto **$scheme**;

# Настройки для поддержки WebSocket (Grafana)

**proxy\_http\_version** 1.1;

**proxy\_set\_header** Upgrade **$http\_upgrade**;

**proxy\_set\_header** Connection "upgrade";

# Дополнительные заголовки для безопасности

**proxy\_set\_header** X-Forwarded-Host **$host**;

**proxy\_set\_header** X-Forwarded-Server **$host**;

# Обработка ошибок подключения

**proxy\_intercept\_errors** on;

error\_page 502 503 504 = @fallback;

}

командой

sudo nginx -t

была проверена корректность файла конфигурации. Сообщение:

nginx: the configuration file /etc/nginx/nginx.conf syntax is ok

nginx: configuration file /etc/nginx/nginx.conf test is successful

гласит, что конфигурация настроена верно.

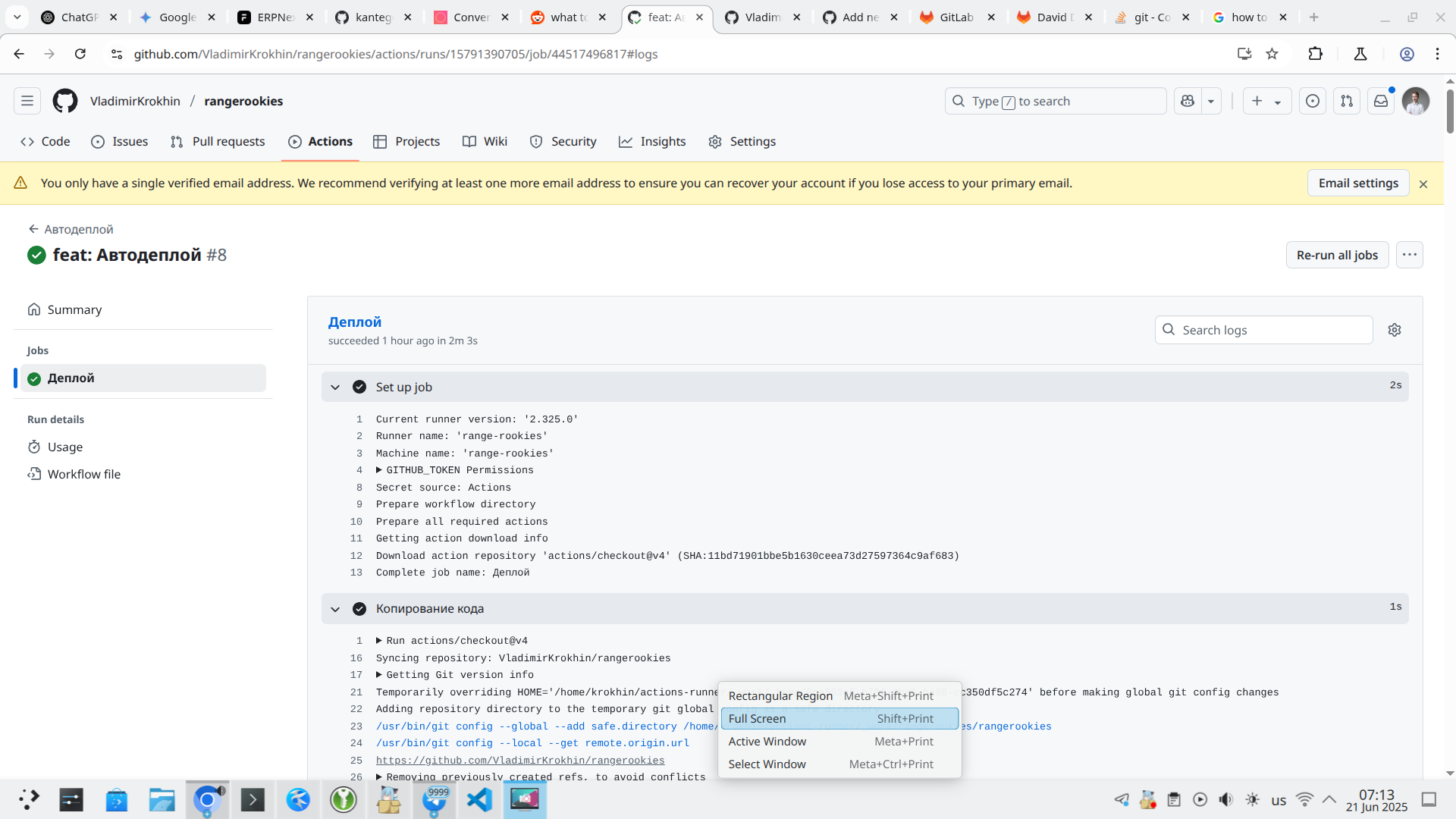
Был перезапущен сервис nginx командой

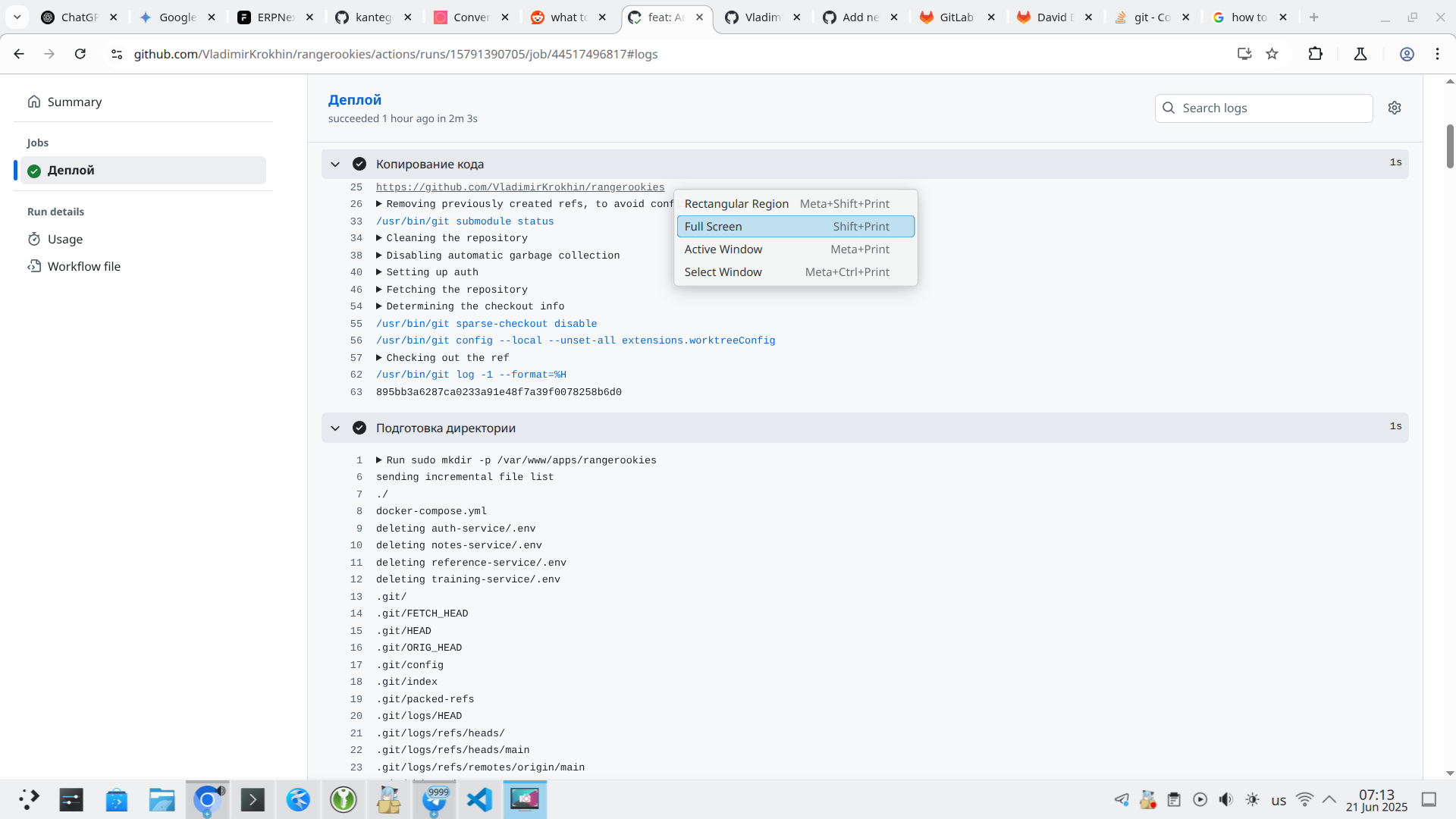
sudo systemctl reload nginx.service

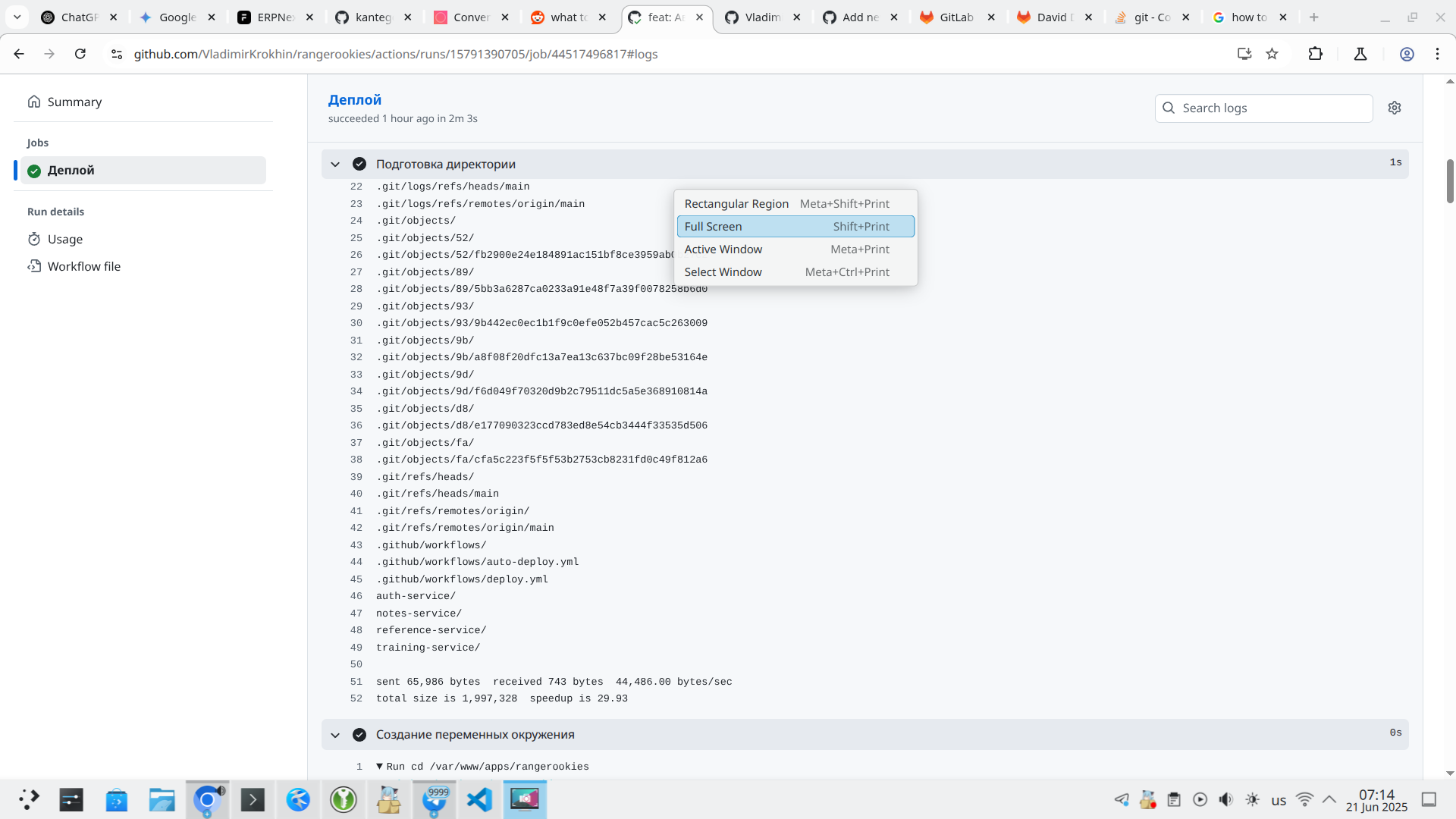
и проверен статус Ngninx

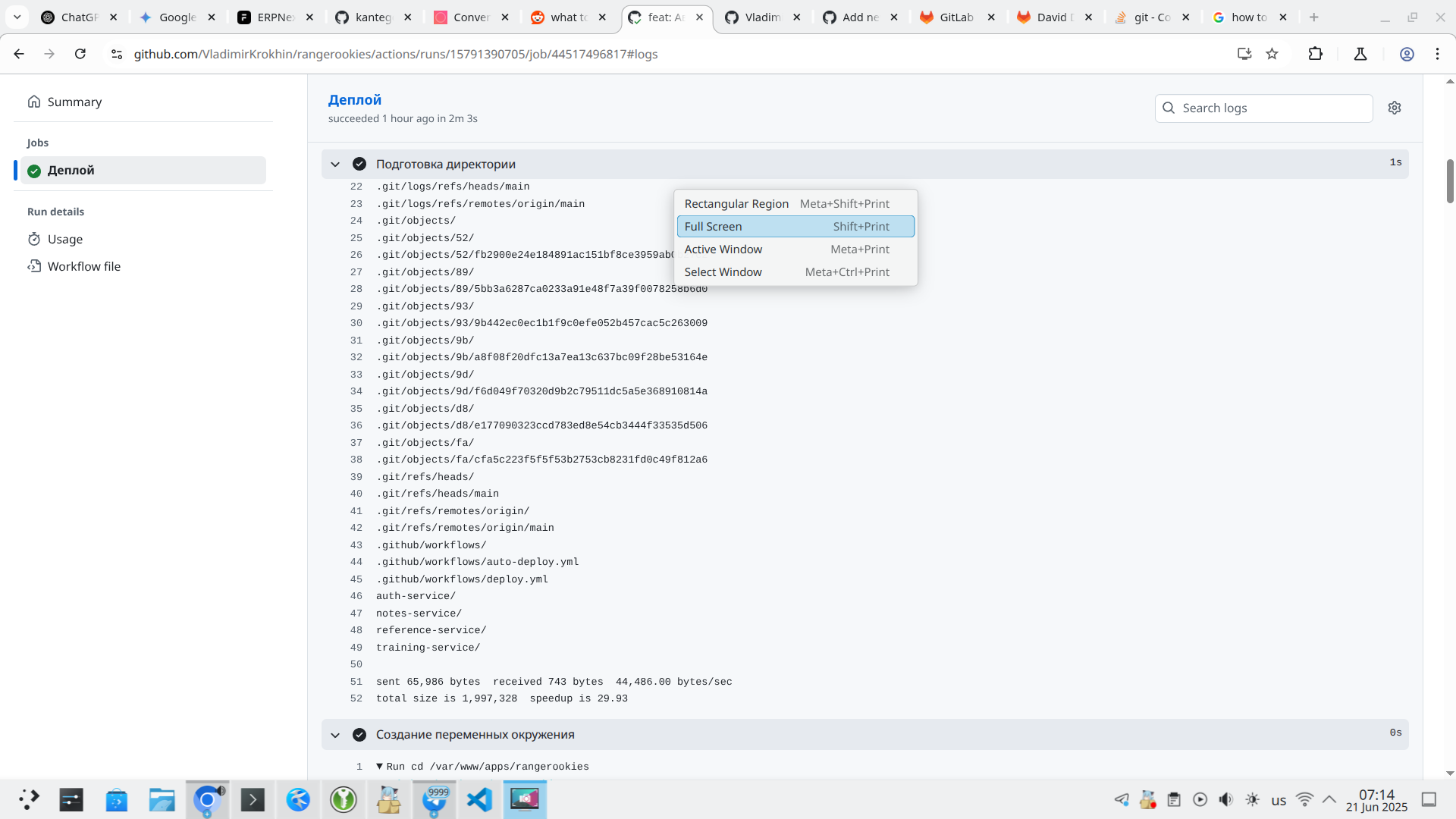
systemctl status nginx.service

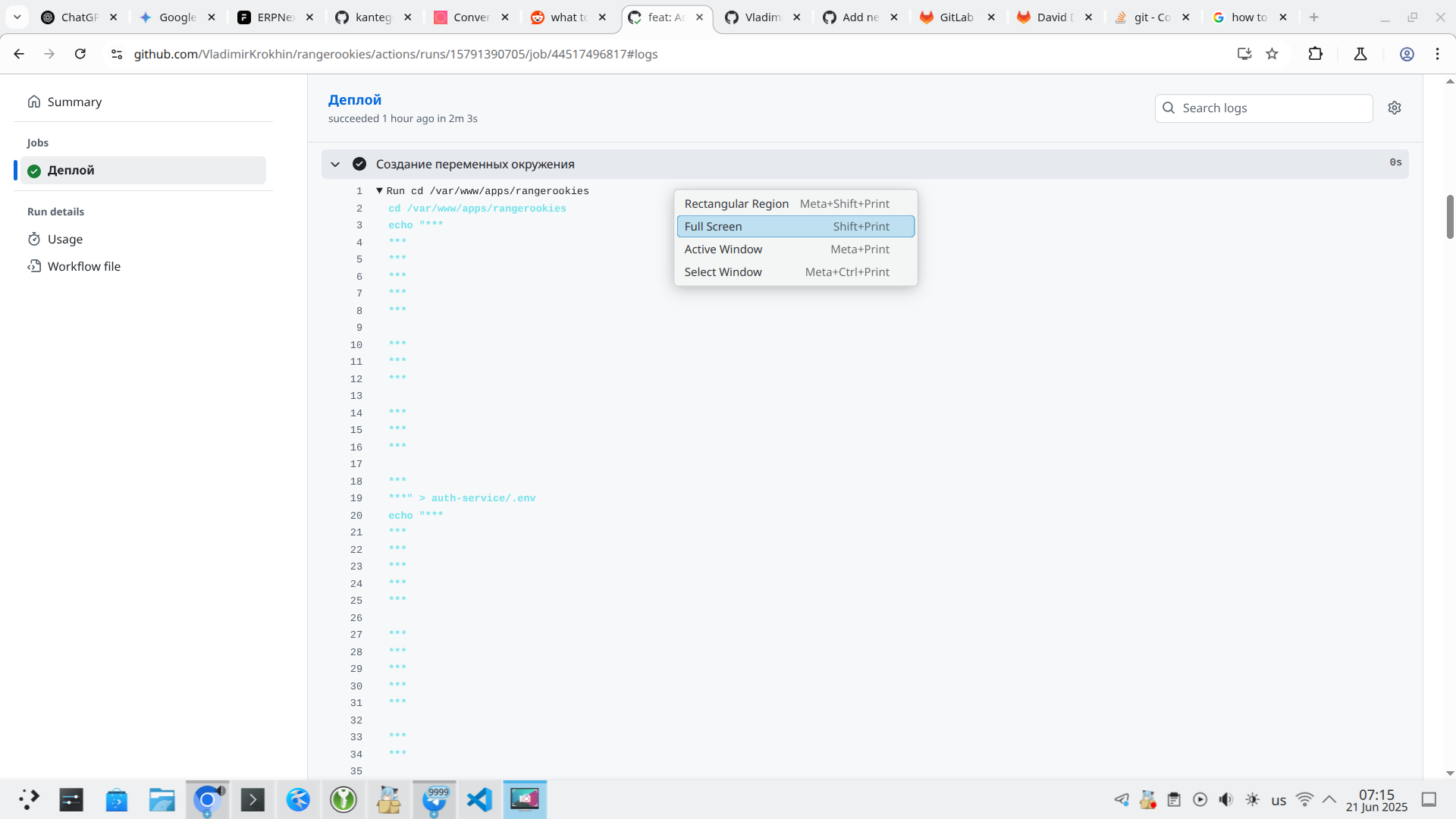
С целью проверить работоспособность автодеплоя по коммиту в ветке main, был запушен коммит. Коммит вызвал соответствующее действие по ссылке <https://github.com/VladimirKrokhin/rangerookies/actions/runs/15791390705/job/44517496817>, полностью отработал успешно.

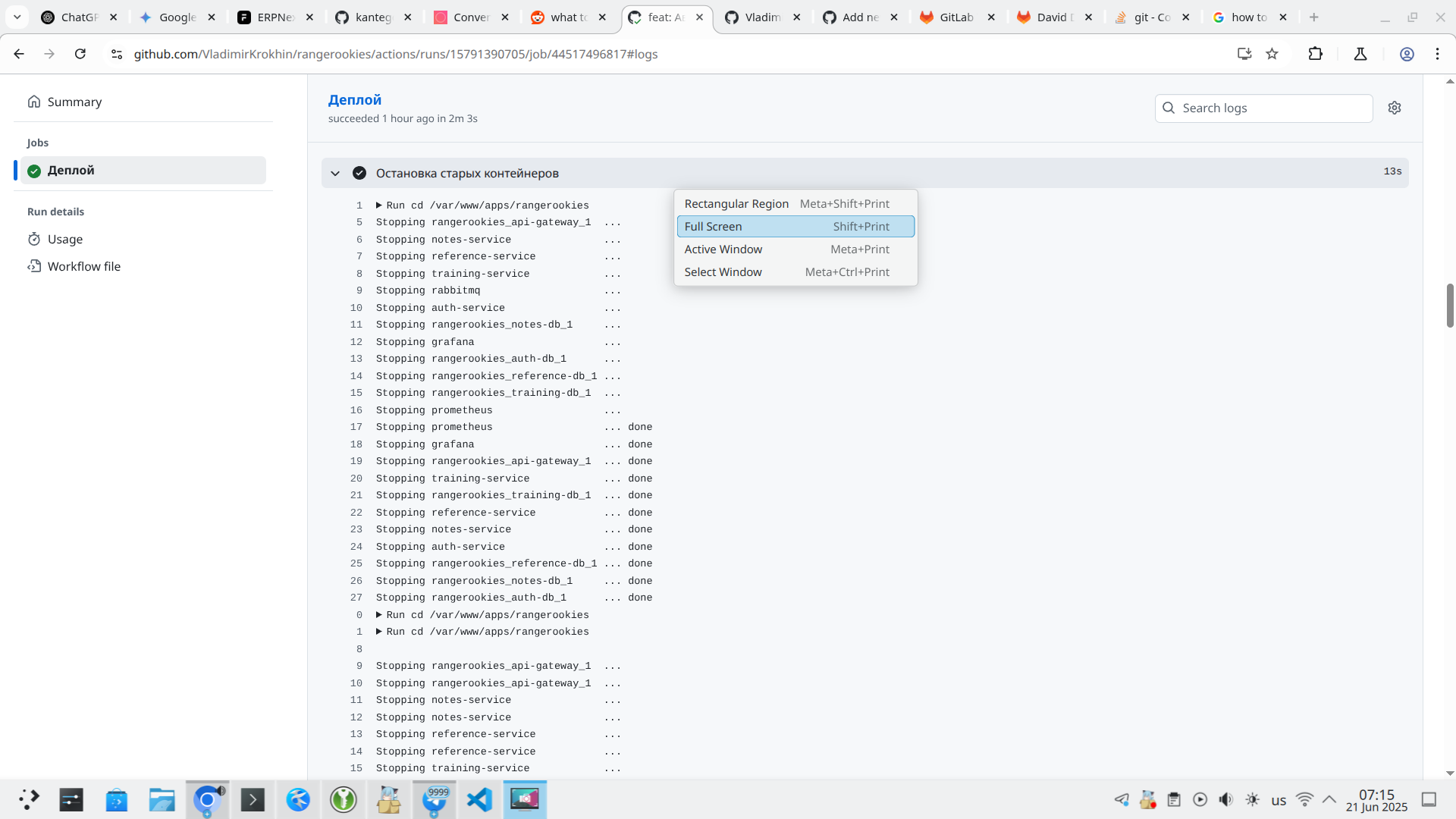


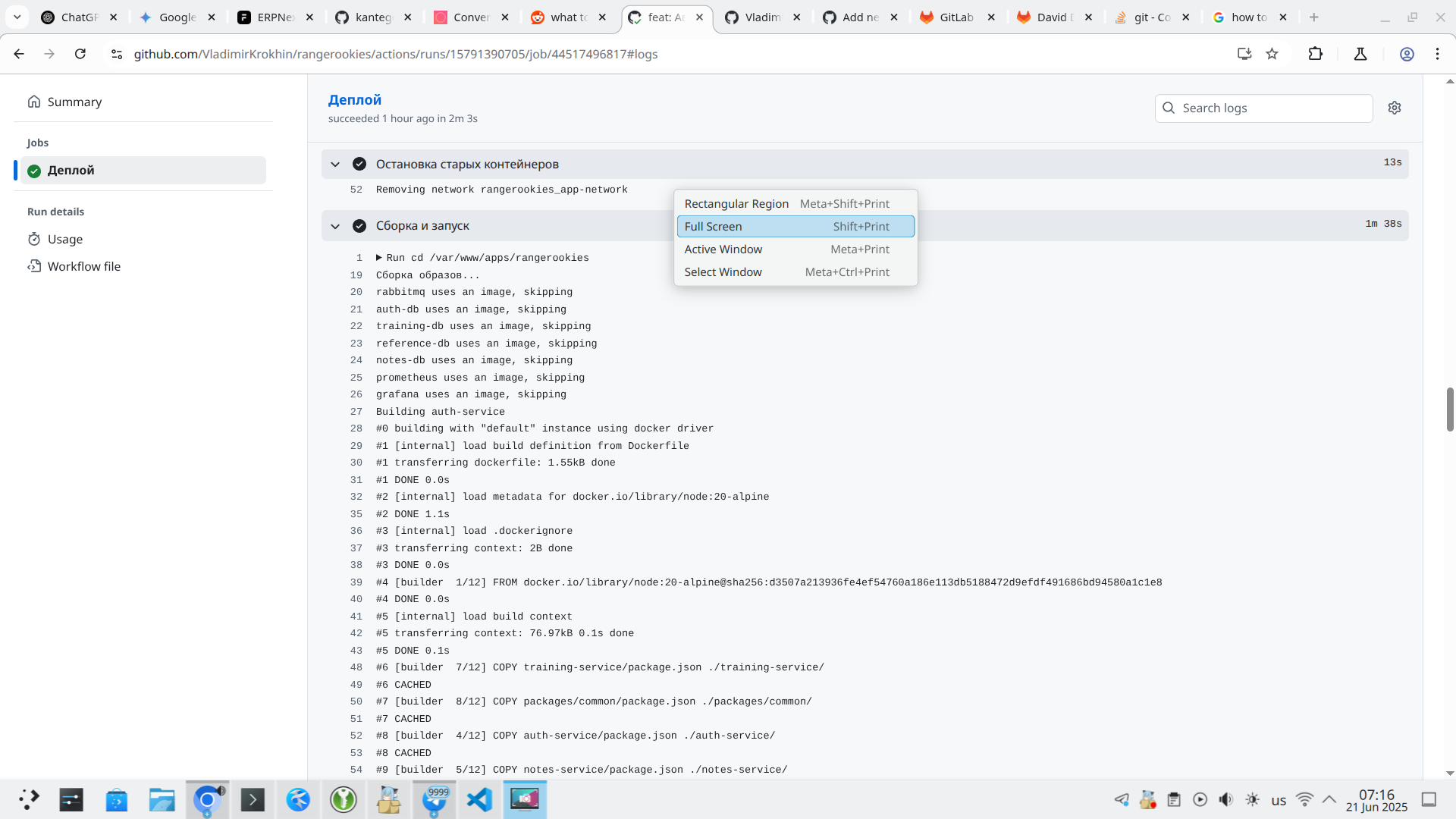


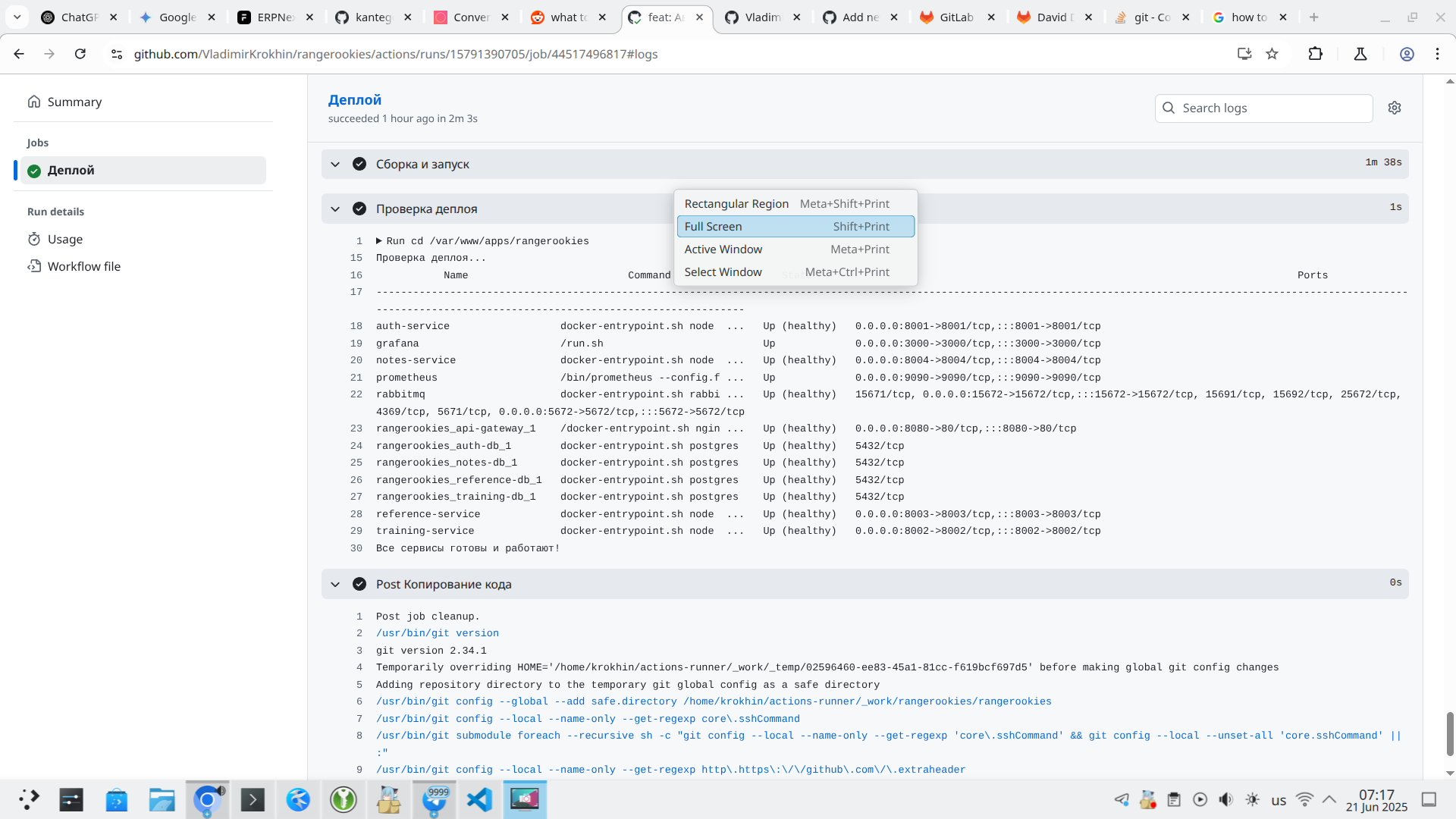








****

****

С персонального компьютера был протестирован API микросервисной системы:

paad@archlinux **~** % curl -X POST http://89.169.162.83/api/auth/register -H "Content-Type: applicati

on/json" -d '{"username":"user1","email":"usera1@example.com","password":"StrongPass123","firstNam

e":"Иван","lastName":"Иванов"}'

{"token":"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VySWQiOjEsImlhdCI6MTc1MDQ3NDQwMSwiZXhwIjoxNzU

wNDc4MDAxfQ.IbpLK8\_GYZaoLrMX42hanQnOiih2lQljYipInmHECS8","user":{"id":1,"email":"usera1@example.co

m","firstName":"Иван","lastName":"Иванов"}}**%**

**Вывод**

В ходе выполнения данной практической работы была успешно решена задача по настройке автоматического развертывания (CI/CD) веб-приложения. Была подготовлена инфраструктура на базе виртуальной машины в Yandex.Cloud, установлен и настроен GitHub Actions Runner для связи с репозиторием. Созданный CI/CD-пайплайн автоматически срабатывает при обновлении main ветки, выполняя сборку и запуск Docker-контейнеров проекта на удаленном сервере. Настроенный веб-сервер Nginx корректно проксирует запросы к развернутому приложению.