**#Project Generation Orders**

**#Генератор заказов 0.3.5**

**1. Описание решения**

Приложение "Генератор заказов" реализует функционал по заказу товаров и услуг, например, для интернет-магазина. В нём реализована модульная система с возможностью репликации (Master-Replica) и партиционирования данных по городам в базе данных PostgreSQL.

Исходные файлы приложения доступны в общем репозитории на GitHub:  
[GitHub Repository](https://github.com/nickfrolov2/orders-generator)

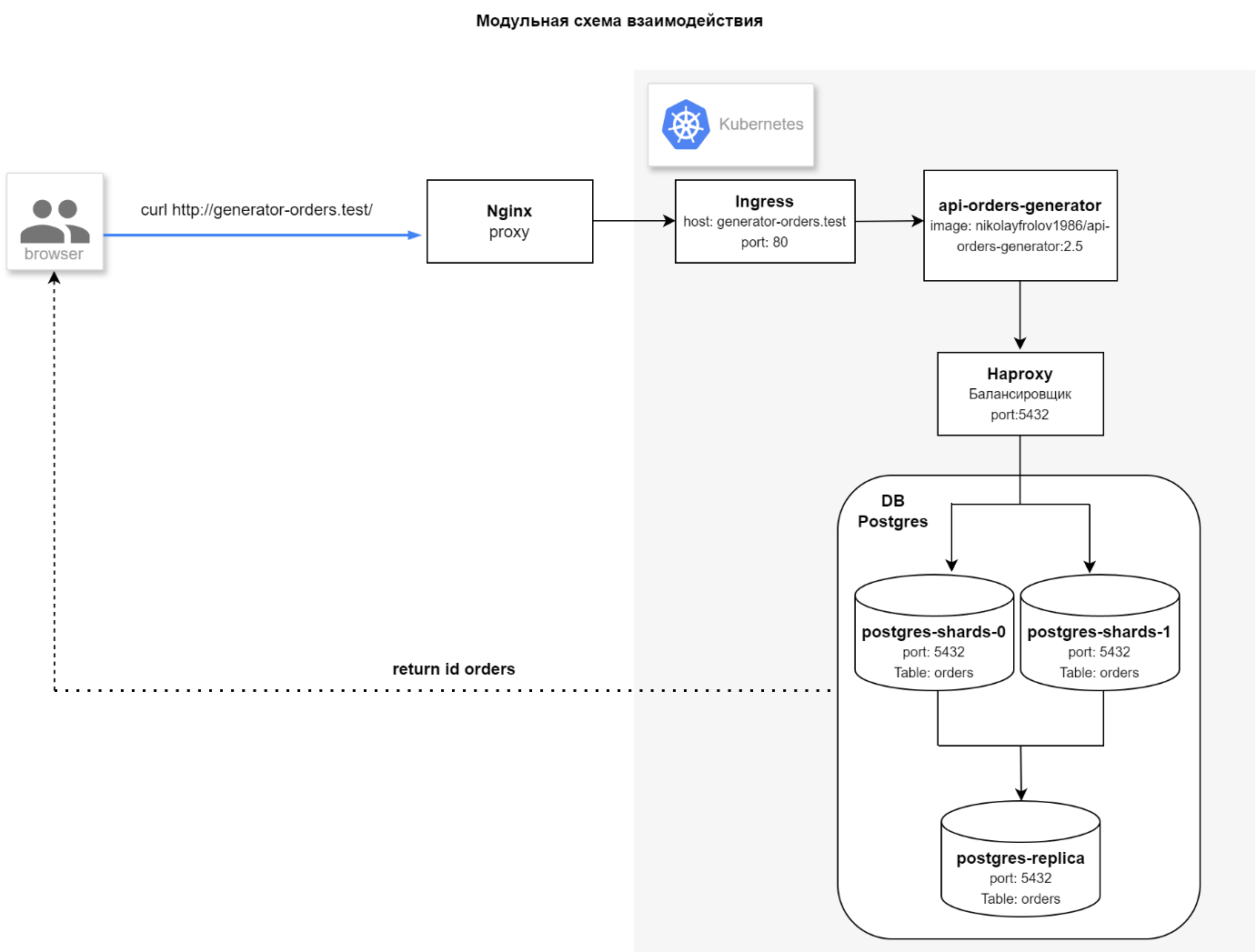
Локальный репозиторий для клонирования (необходима авторизация):

* SSH: [git@gitlaboff.devopsf.ru](mailto:git@gitlaboff.devopsf.ru):mygroup1/test\_project\_frolovna.git
* HTTPS: <https://gitlaboff.devopsf.ru/mygroup1/test_project_frolovna.git>

Подключение репозитория через Helm:  
helm repo add orders\_generator <https://raw.githubusercontent.com/nickfrolov2/orders-generator/refs/heads/main/>

Запуск/установка производится с помощью команды:  
helm install orders orders\_generator/project\_orders

**2. Общая схема взаимодействия модулей**



**3. Автоматический Deploy**

Для реализации автоматического деплоя был разработан механизм, использующий GitLab. В процессе используются следующие инструменты:

* Helm — для упаковки приложений.
* Docker — для создания образов контейнеров.
* Container Registry — для хранения созданных образов.
* Kubernetes — для оркестрации контейнеров.

Pipelines описаны в примере файла .gitlab-ci.yml, который расположен в папке artifacts. В целях безопасности из файла были удалены пароли доступа к платформам, поэтому необходимо добавить свои учетные данные перед использованием.

**Этапы:**

1. Устанавливаем gitlab offline на виртуальную машину или в облако.

На примере: <https://gitlaboff.devopsf.ru/> (Проект развернут на виртуальной машине в облаке Базис.DynamiX)

1. Копируем репозиторий с файлами Helm Charts в корневую папку проекта.
2. Создаем файл .gitlab-ci.yml и описываем stages. Пример файла находится в папке Artifacts/
3. stages:
4. - setup\_helm
5. - package\_helm
6. - push\_container\_registry
7. - deploy\_helm\_kubernetes

Setup\_helm – установка Helm

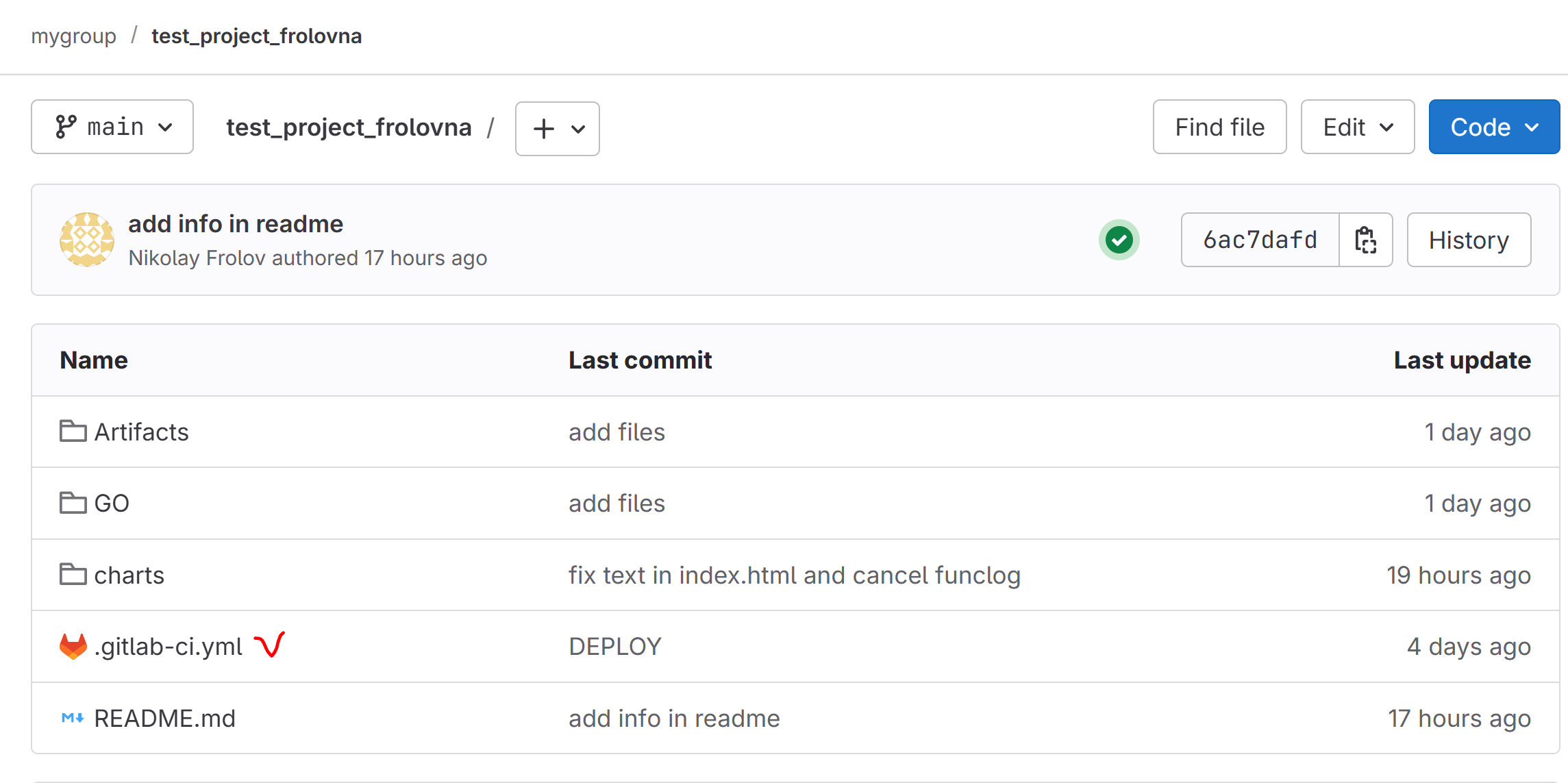
Package\_helm – упаковка проекта Helm

Push\_container\_registry – создание образа и пушинг его в gitlab container\_registry

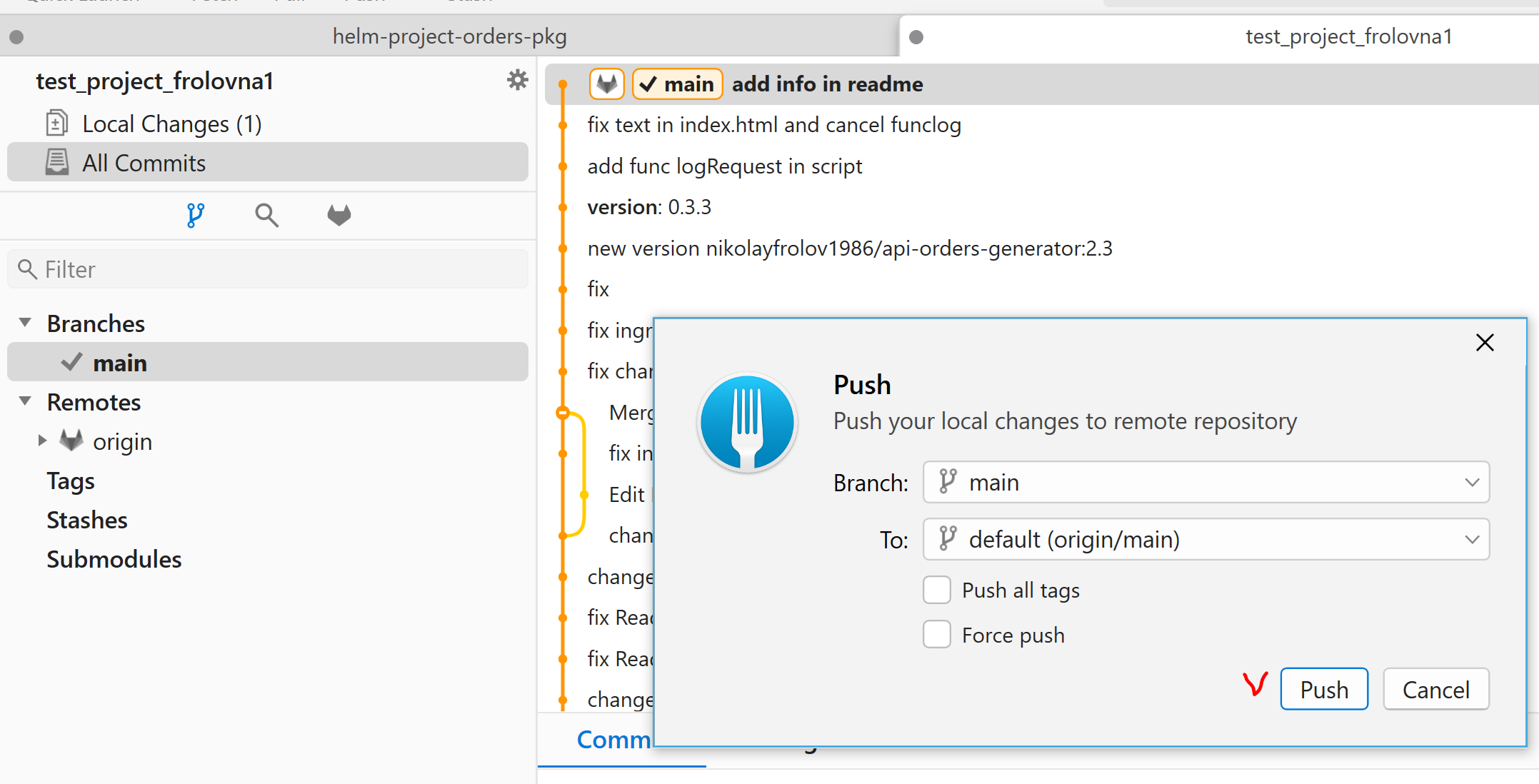
Deploy\_helm\_kubernetes – деплой образа в Kubernetes

4. Перед запуском pipelines необходимо настроить Runners (процесс описан в документации на официальном сайте: <https://docs.gitlab.com/runner/>)

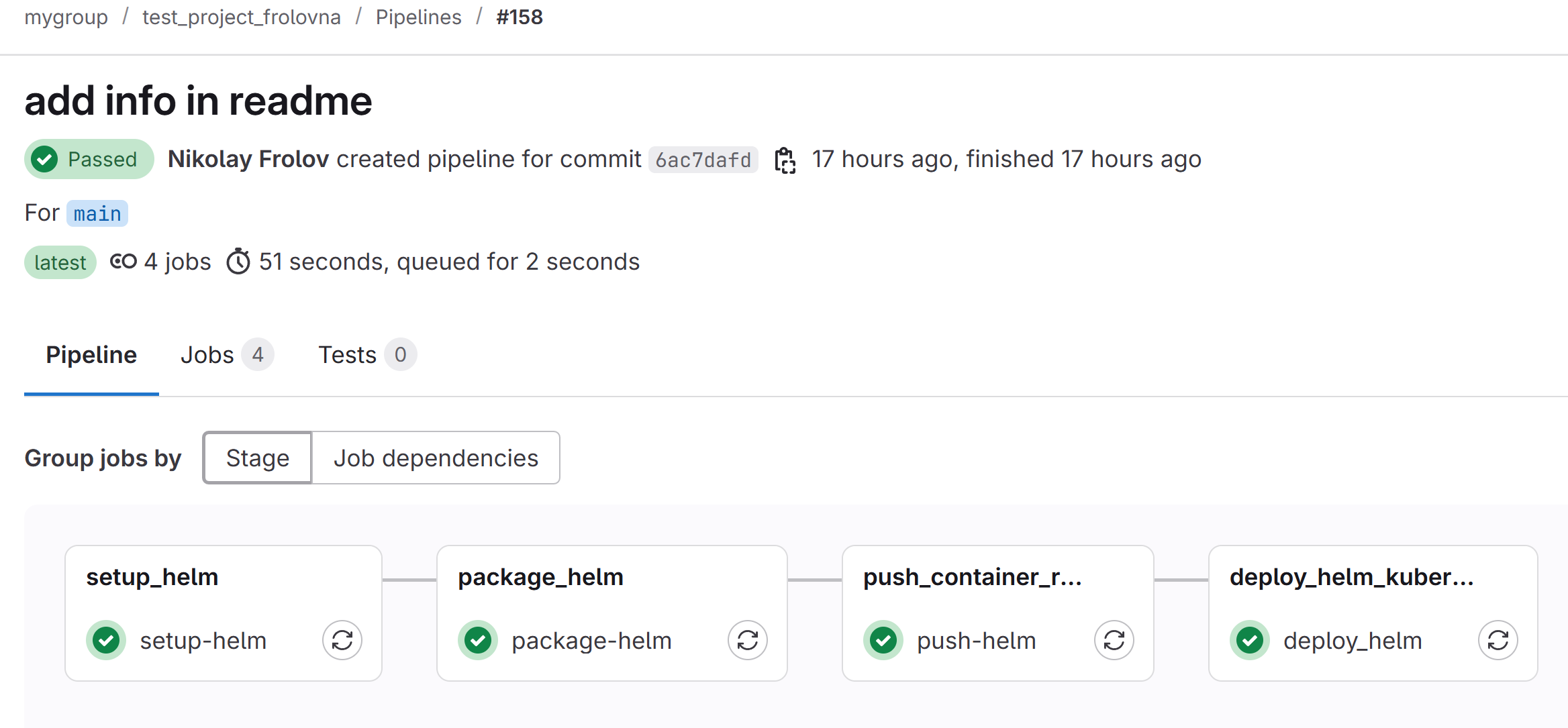
5. После копирования файлов проекта в репозиторий и подключения runner к проекту, делаем push в ветку (необходимо убедиться, что файл **.gitlab-ci.yml** находится в корне проекта и настроен)



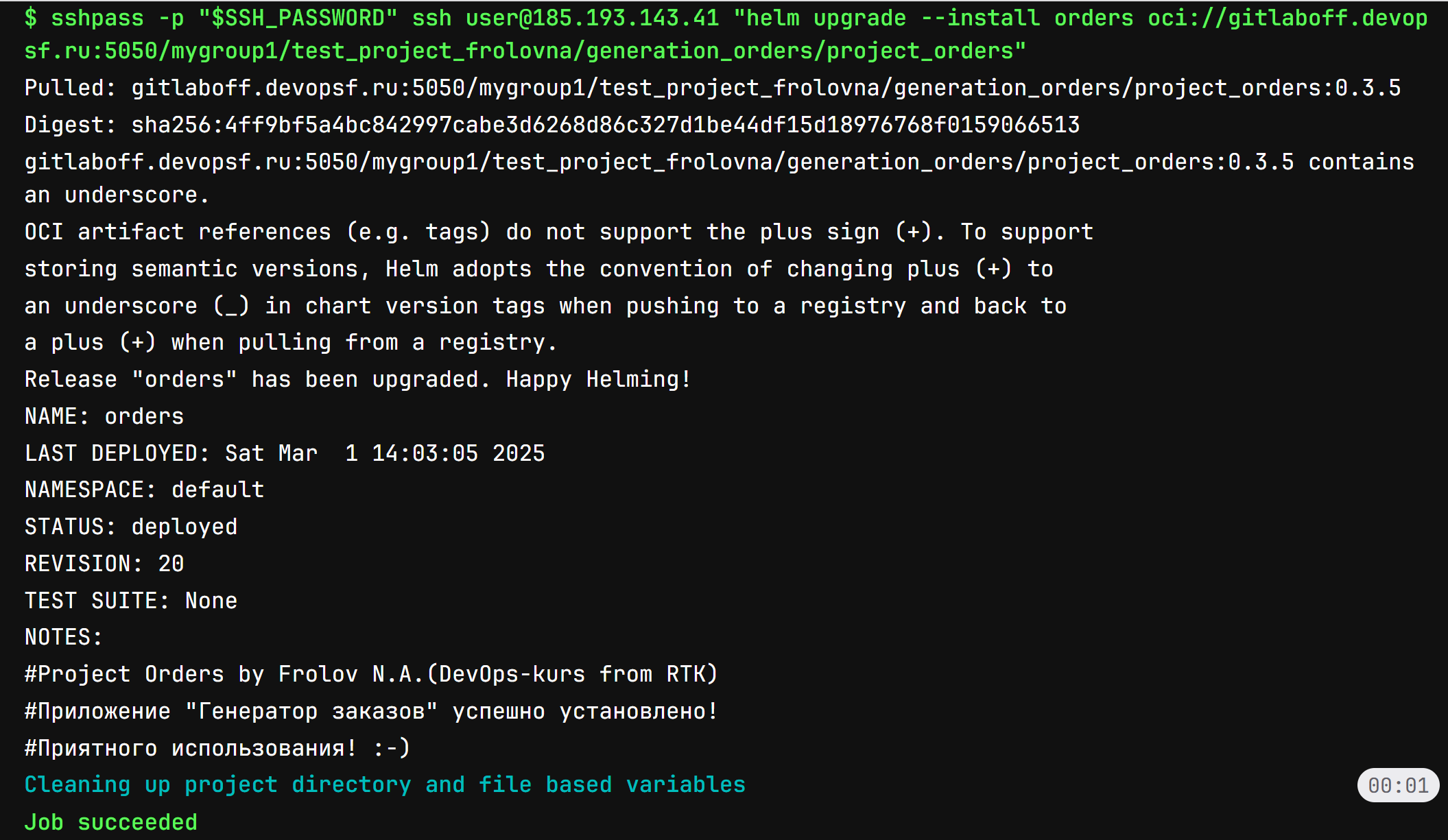
**Делаем Push в ветку Main:**



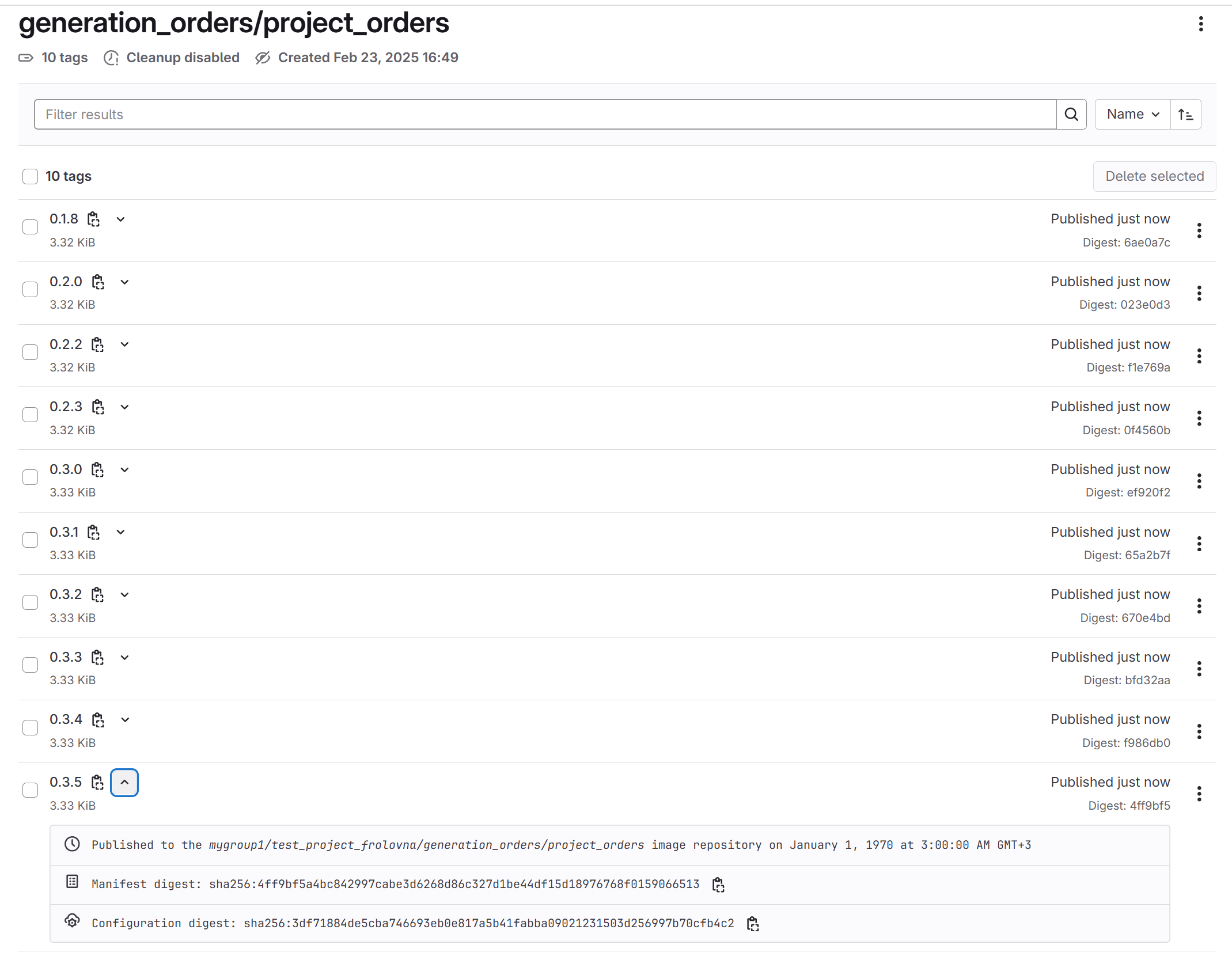
6. Автоматически стартует процесс Pipeline. Наблюдаем за процессом в разделе Build/Pipelines



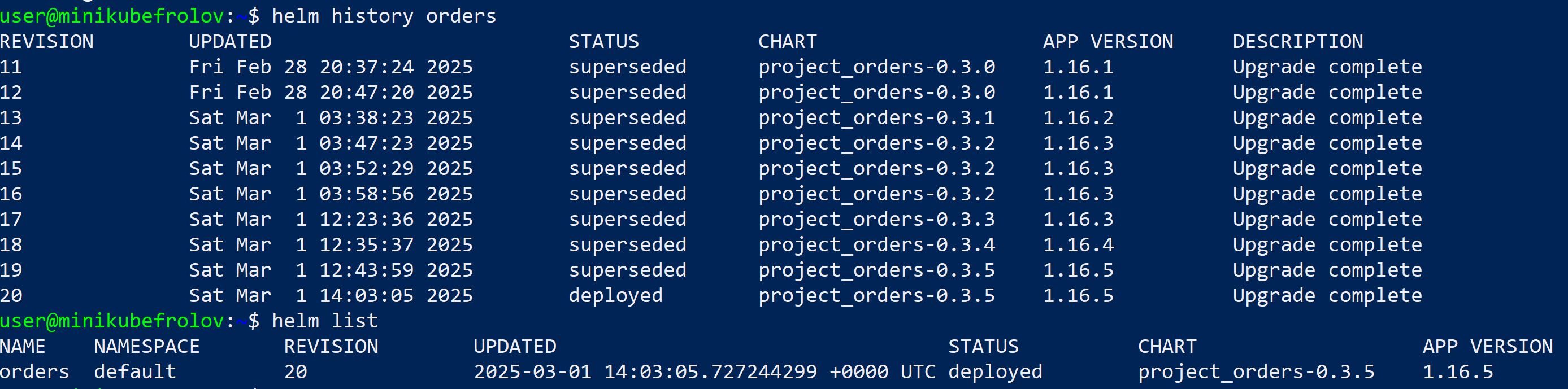
7. При успешном deploy увидим результат:



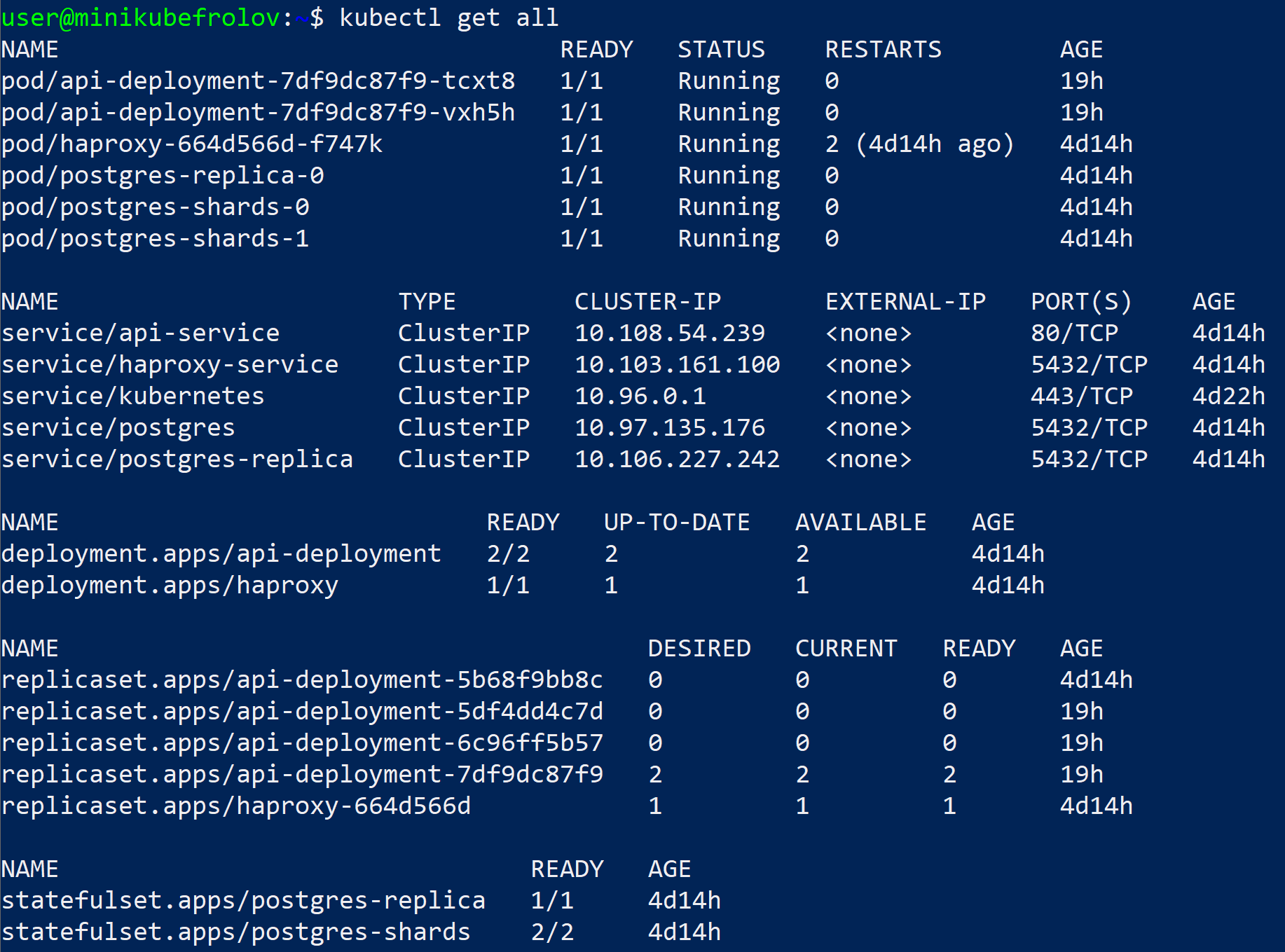
1. В разделе Deploy/Container Registry должен добавиться созданный образ с нужным тэгом приложения:



1. На стороне Kubernetes убеждаемся, что приложение развернулось из образа и успешно установилось:

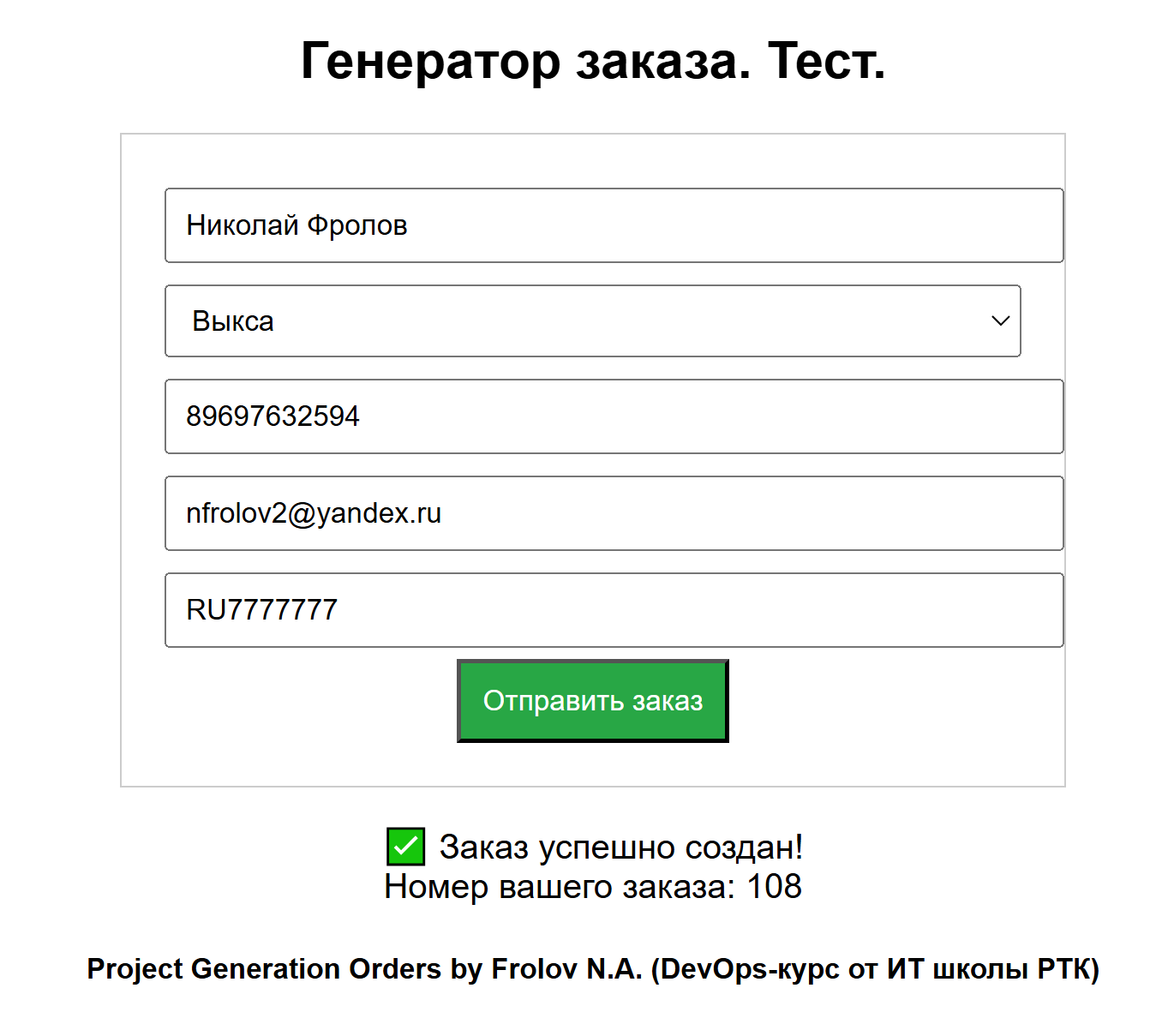


Все деплойменты и скрипты отработали успешно:



1. В браузере проверяем работоспособность приложения:

На примере, <https://generator-orders.devopsf.ru/>



1. **Конфигурация Nginx.**

# Default server configuration

server {

root /var/www/html;

# Add index.php to the list if you are using PHP

index index.html index.htm index.nginx-debian.html;

server\_name generator-orders.devopsf.ru; # managed by Certbot

location / {

proxy\_pass http://generator-orders.test;

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;

}

listen [::]:443 ssl ipv6only=on; # managed by Certbot

listen 443 ssl; # managed by Certbot

ssl\_certificate /etc/letsencrypt/live/generator-orders.devopsf.ru/fullchain.pem; # managed by Certbot

ssl\_certificate\_key /etc/letsencrypt/live/generator-orders.devopsf.ru/privkey.pem; # managed by Certbot

include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot

ssl\_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot

}

server {

if ($host = generator-orders.devopsf.ru) {

return 301 https://$host$request\_uri;

} # managed by Certbot

listen 80 ;

listen [::]:80 ;

server\_name generator-orders.devopsf.ru;

return 404; # managed by Certbot

}