МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Пермский государственный национальный исследовательский университет

Кафедра информационных технологий

Лабораторная работа по дисциплине

«Методология автоматизации технологических процессов сборки, настройки и развертывания программного обеспечения»

Проверил:

кандидат технических наук, доцент Моисеев В.И.

Выполнил:

студент группы ИТХ-1,2-21

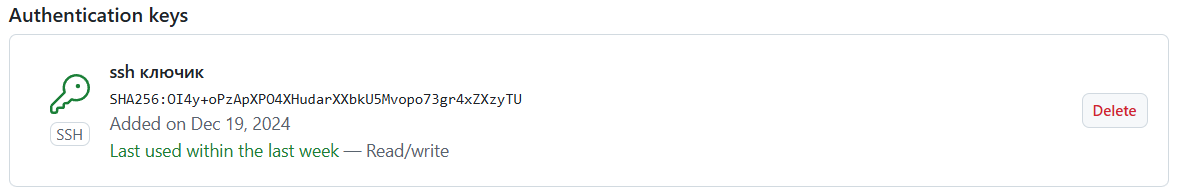
Парамонов А.С.

Пермь 2024

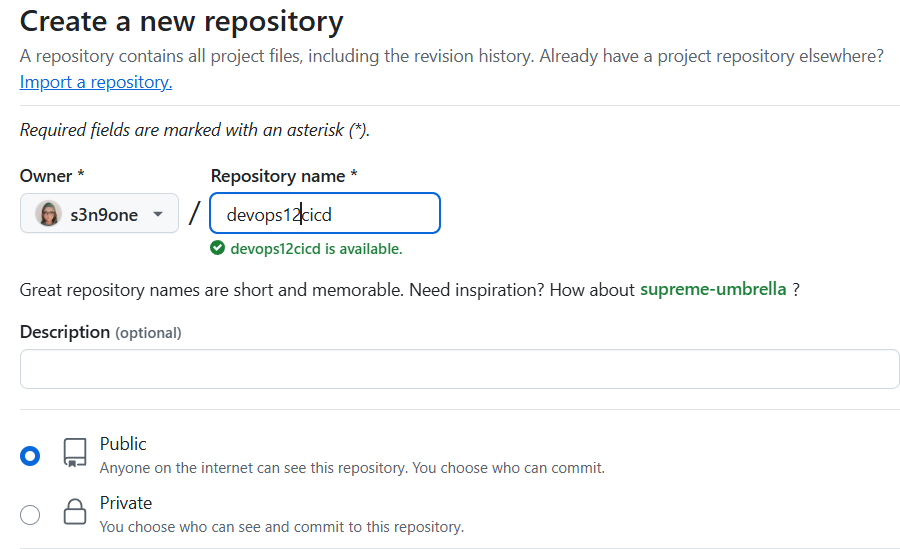
Непрерывная интеграция и развертывание (CI/CD)

Порядок выполнения работы:

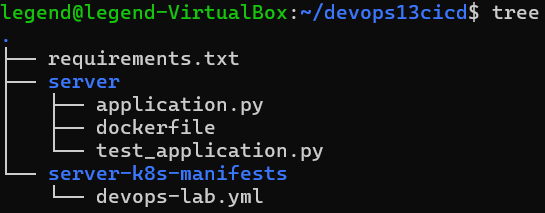
Добавляем публичный ssh ключ в профиль GitHub



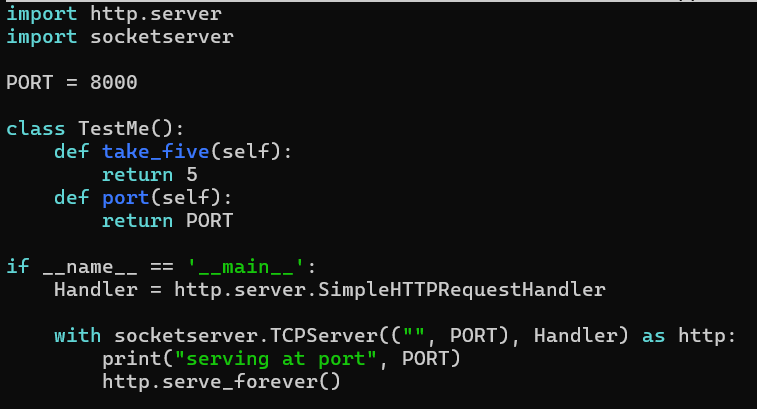
Создаем репозиторий для приложения



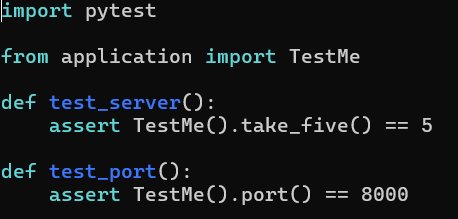
Далее копируем репозиторий с гитхаба. Ниже представлена структура каталогов (директория .github почему-то не видна, но она есть)



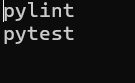
Приложение (уже без ошибок) server/application.py



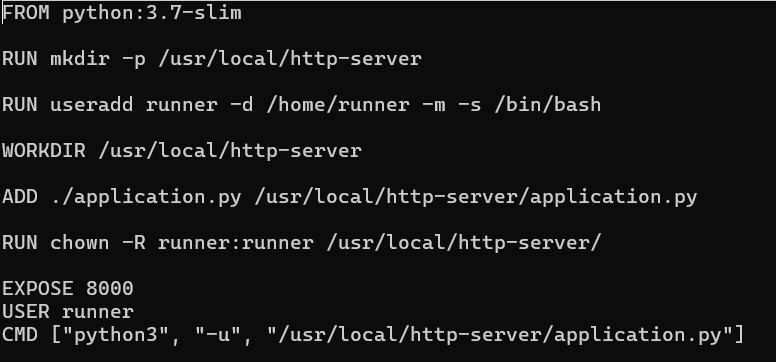
Юнит-тесты server/test\_application.py



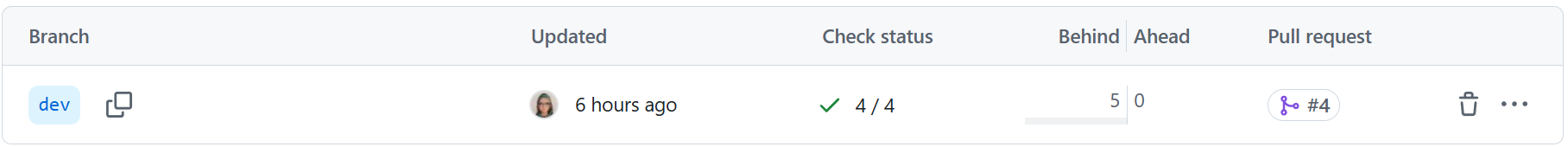
Файл с зависимостями requirements.txt



Файл для сборки образа dockerfile

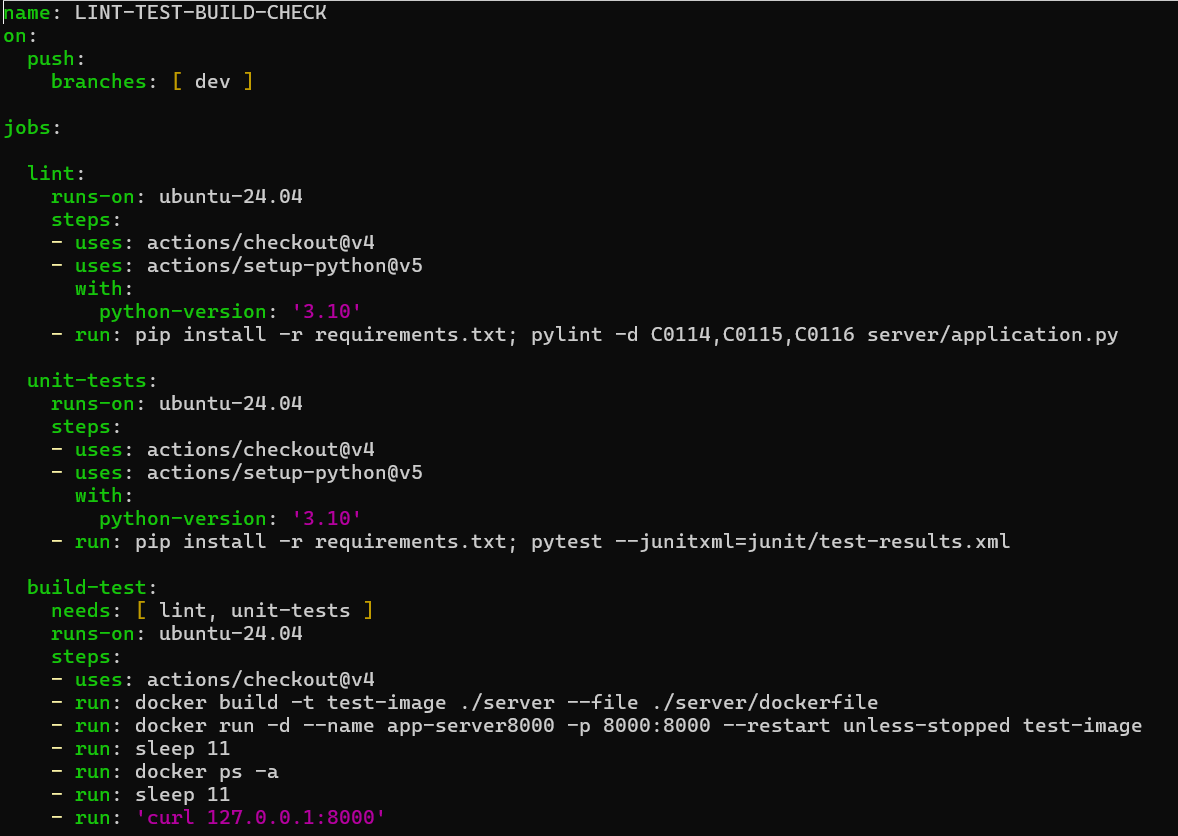


Далее добавляем все созданные файлы в гит, делаем коммит и пушим в репозиторий. Затем переходим на ветку dev, а ветку main защищаем от прямых изменений.



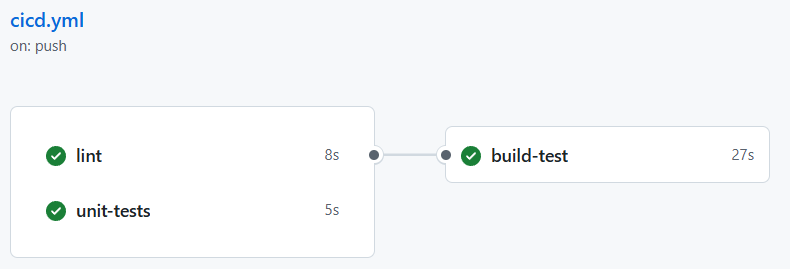
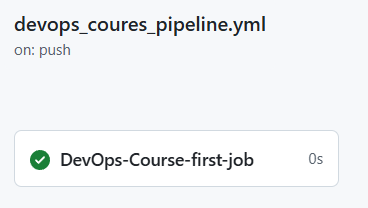
Continuous Integration

Добавляем CI-пайплайны (сценарии) – Github Actions. Файл .github/workflows/cicd.yml



Добавляем сценарий в гит и проверяем, как он отработает.

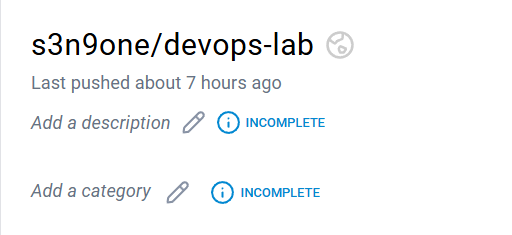




Затем куберизируем приложение. Создаем для приложения k8s-манифесты. Создаем отдельный каталог для манифестов.

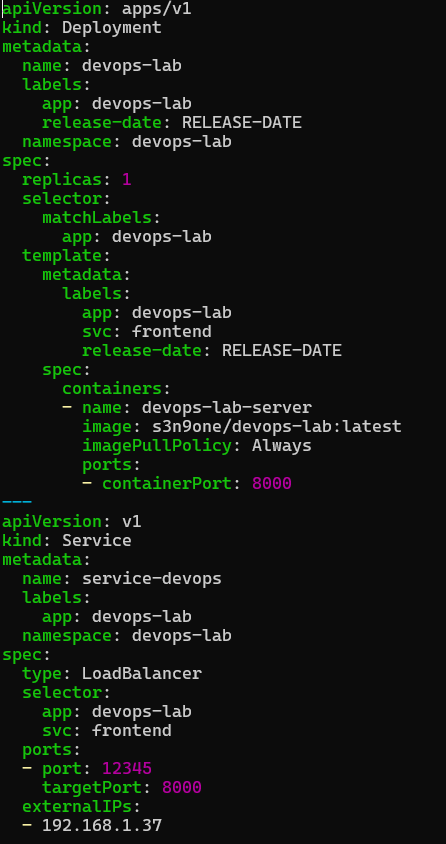


Далее переходим на Docker Hub и создаем публичный репозиторий с именем devops-lab.



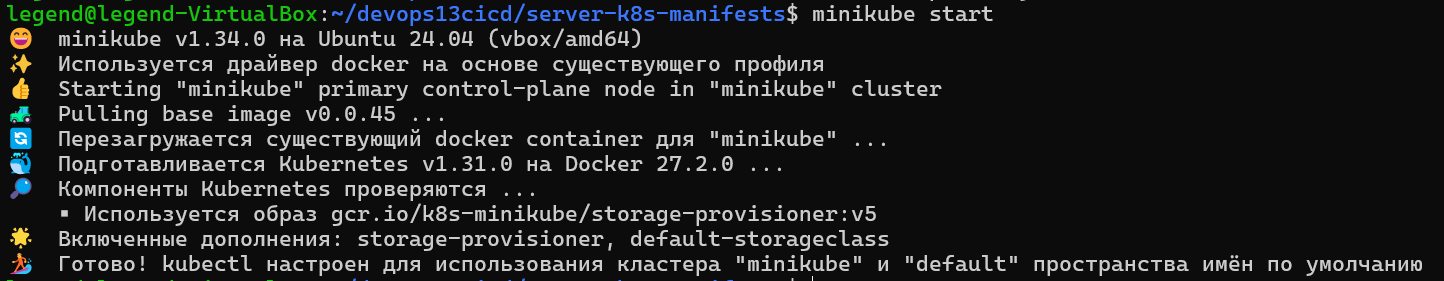
Куберизируем приложение: создаем K8s манифест

./server-k8s-manifests/devops-lab.yml

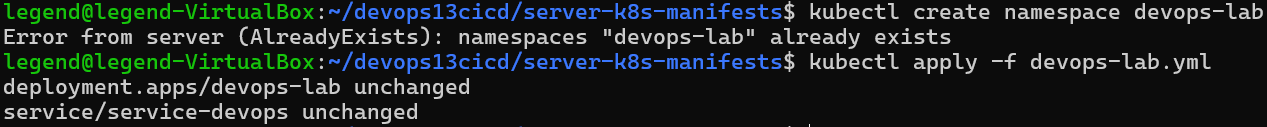


У вм другой ip, потому что использовался сетевой мост. При использовании NAT на вм отсутствовало подключение к интернету и хостовая машина не видела вм в сети.

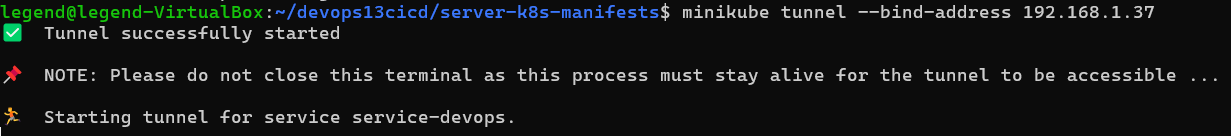
Проверяем/запускаем minikube



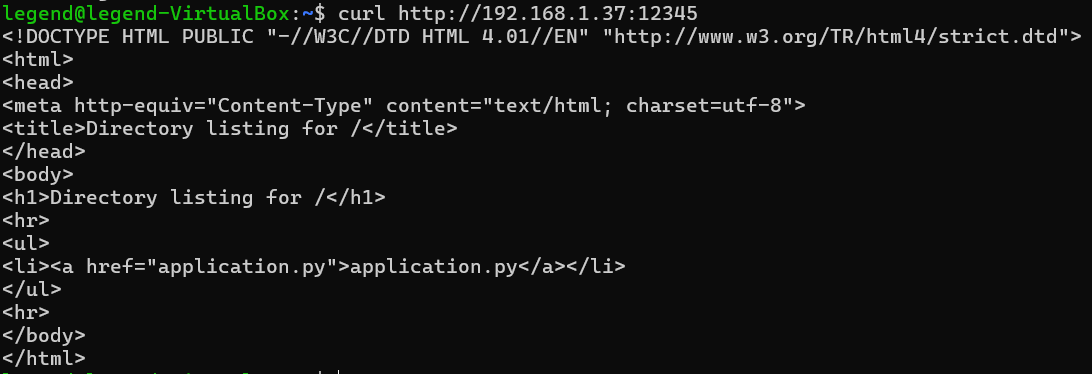
Далее применяем k8s-манифесты



Запускаем проброс портов для сервисов LoadBalancer



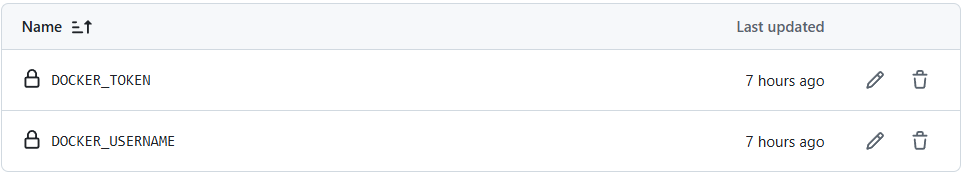
Проверка



Далее сохраняем манифесты в github.

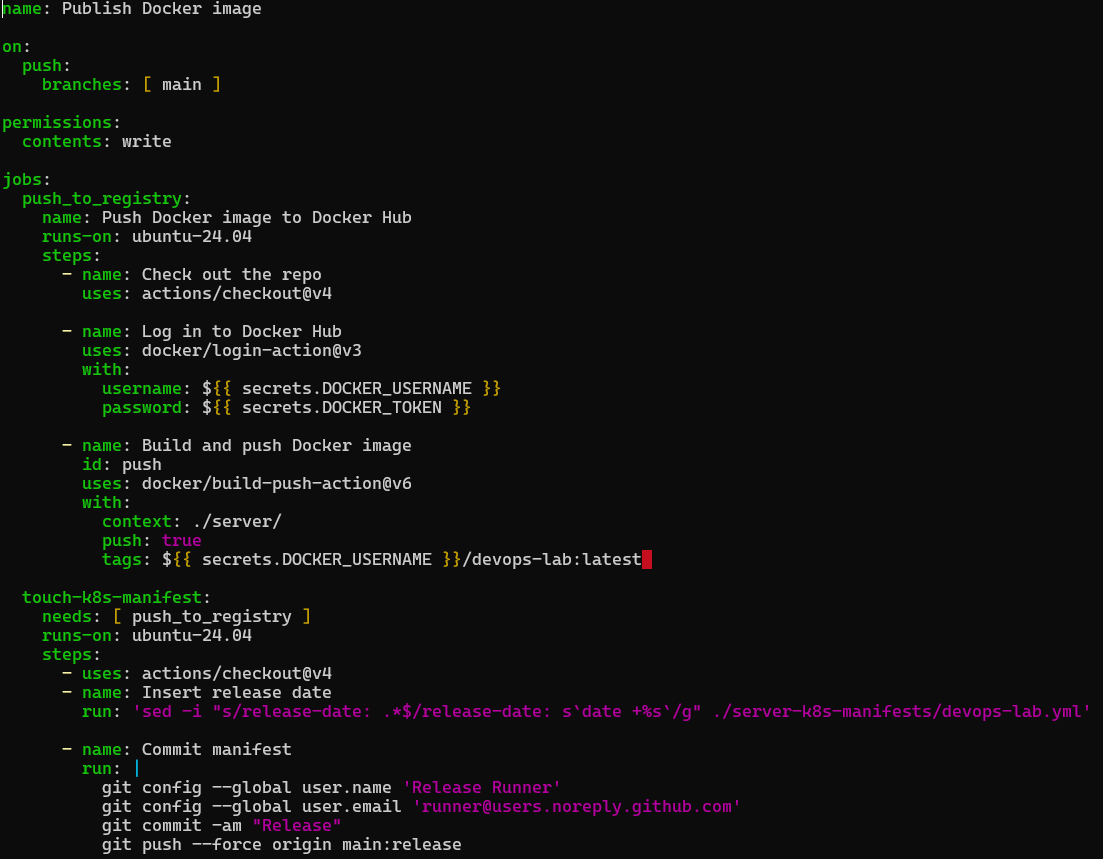
Continuous Delivery

Создаем 2 секрета в репозитории на github с DOCKER\_USERNAME и DOCKER\_TOKEN

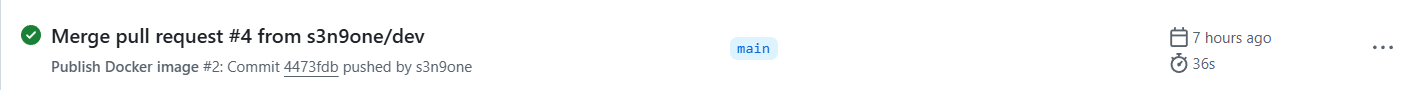


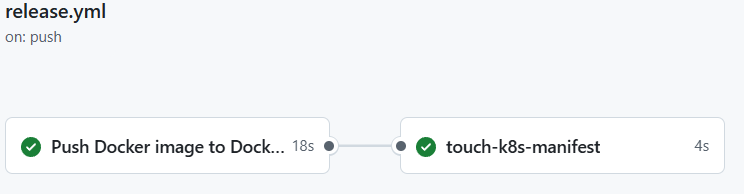
Далее создаем пайплайн для сборки и публикации образа в Docker Hub

.github/workflows/release.yml



Делаем коммит и пушим новые файлы в репозиторий, а затем создаем pull-request и выполняем сливание в main ветку. При этом будет выполнен пайплайн release.yml. Проверяем обработку пайплайна.





Видим, что пайплайн успешно положил новый образ в docker hub

