**Домашнее задание к занятию "09.05 Gitlab"**

**Подготовка к выполнению**

1. Необходимо [зарегистрироваться](https://about.gitlab.com/free-trial/)
2. Создайте свой новый проект
3. Создайте новый репозиторий в gitlab, наполните его [файлами](https://github.com/netology-code/mnt-homeworks/blob/master/09-ci-05-gitlab/repository)
4. Проект должен быть публичным, остальные настройки по желанию

**Основная часть**

**DevOps**

В репозитории содержится код проекта на python. Проект - RESTful API сервис. Ваша задача автоматизировать сборку образа с выполнением python-скрипта:

1. Образ собирается на основе [centos:7](https://hub.docker.com/_/centos?tab=tags&page=1&ordering=last_updated)
2. Python версии не ниже 3.7
3. Установлены зависимости: flask flask-jsonpify flask-restful
4. Создана директория /python\_api
5. Скрипт из репозитория размещён в /python\_api
6. Точка вызова: запуск скрипта
7. Если сборка происходит на ветке master: Образ должен пушится в docker registry вашего gitlab python-api:latest, иначе этот шаг нужно пропустить

**Product Owner**

Вашему проекту нужна бизнесовая доработка: необходимо поменять JSON ответа на вызов метода GET /rest/api/get\_info, необходимо создать Issue в котором указать:

1. Какой метод необходимо исправить
2. Текст с { "message": "Already started" } на { "message": "Running"}
3. Issue поставить label: feature

**Developer**

Вам пришел новый Issue на доработку, вам необходимо:

1. Создать отдельную ветку, связанную с этим issue
2. Внести изменения по тексту из задания
3. Подготовить Merge Requst, влить необходимые изменения в master, проверить, что сборка прошла успешно

**Tester**

Разработчики выполнили новый Issue, необходимо проверить валидность изменений:

1. Поднять докер-контейнер с образом python-api:latest и проверить возврат метода на корректность
2. Закрыть Issue с комментарием об успешности прохождения, указав желаемый результат и фактически достигнутый

**Итог**

После успешного прохождения всех ролей - отправьте ссылку на ваш проект в гитлаб, как решение домашнего задания

**Необязательная часть**

Автомазируйте работу тестировщика, пусть у вас будет отдельный конвейер, который автоматически поднимает контейнер и выполняет проверку, например, при помощи curl. На основе вывода - будет приниматься решение об успешности прохождения тестирования

**Как оформить ДЗ?**

Выполненное домашнее задание пришлите ссылкой на .md-файл в вашем репозитории.

1. Установили все зависимости для теста файла

sudo pip install Flask #устанавливаем зависимости для питона скрипта

sudo pip install Resource

sudo pip install Api

sudo pip install jsonify

pip install dump

скопировали файл из каталога нетологии # копируем питон файл по ДЗ

git clone [git@gitlab.com:netology8/homework-for-lesson-09.05-gitlab.git](mailto:git@gitlab.com:netology8/homework-for-lesson-09.05-gitlab.git) # склонировал

git switch -c main # переходим в ветку main

git add \* # в stage

git config --global user.name Dmitry Krasilnikov #логиним

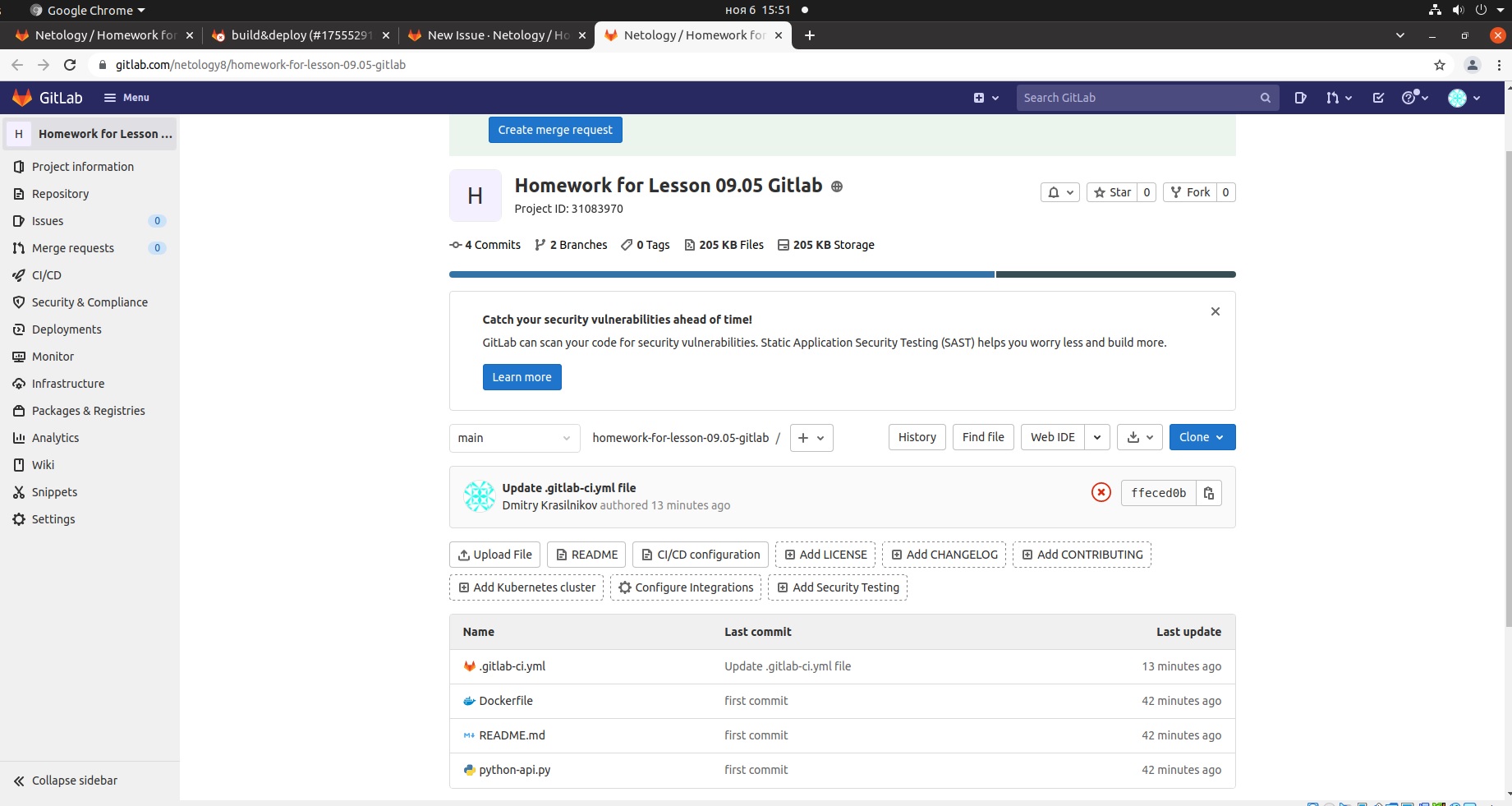
git config --global user.email [kras5.59@yandex.ru](mailto:kras5.59@yandex.ru)

git commit -m "python file" # комитим

git remote add origin <https://gitlab.com/netology8/homework-for-lesson-09.05-gitlab> #переходим ? В инструкции указана команда

git push origin master # пушим

см. скрин “надобавляли файлы.jpg”



Файлик

from flask import Flask, request

from flask\_restful import Resource, Api

from json import dumps

from flask\_jsonpify import jsonify

app = Flask(\_\_name\_\_)

api = Api(app)

class Info(Resource):

def get(self):

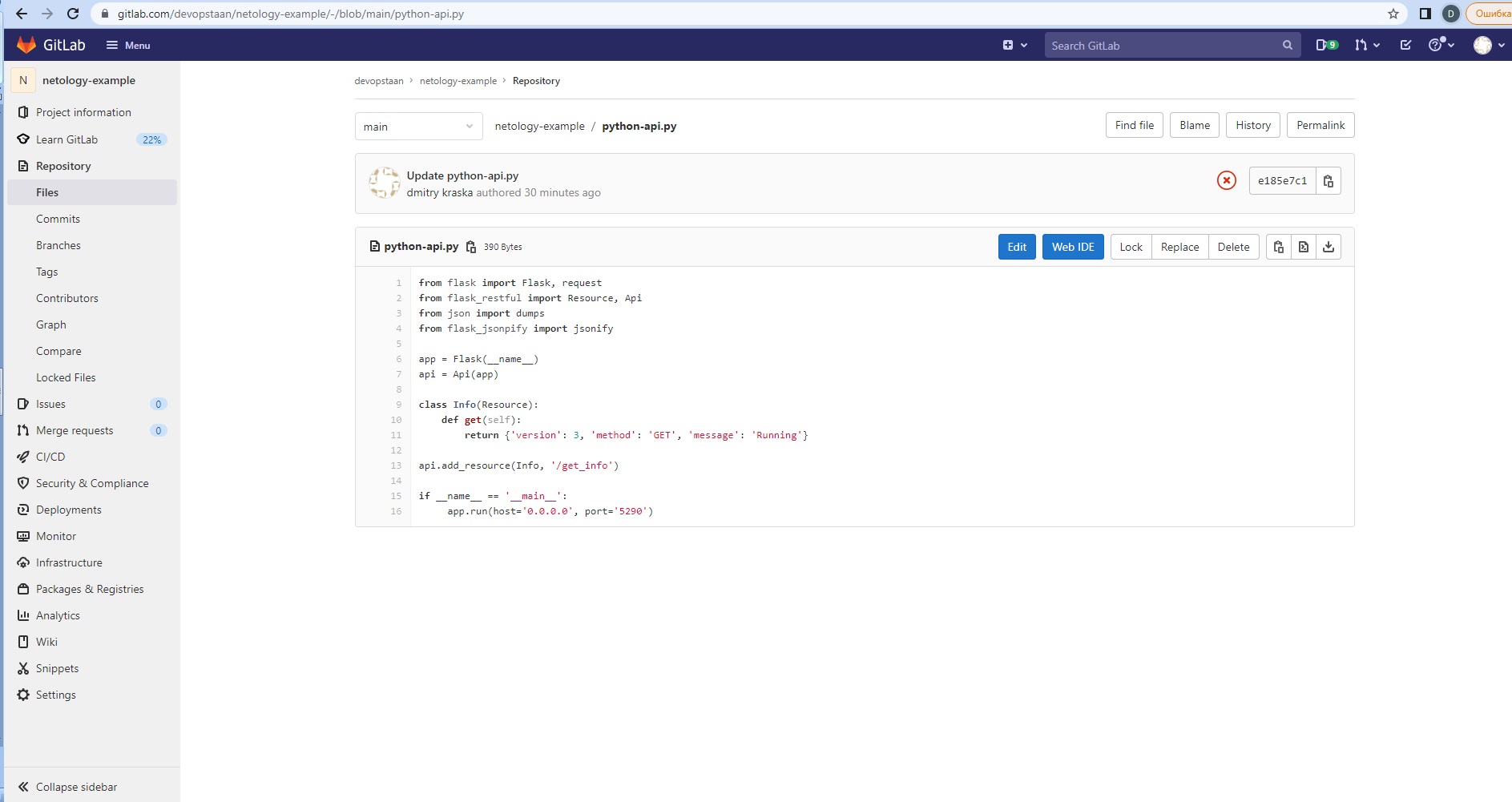
return {'version': 3, 'method': 'GET', 'message': 'Running'}

api.add\_resource(Info, '/get\_info')

if \_\_name\_\_ == '\_\_main\_\_':

app.run(host='0.0.0.0', port='5290')

см. скрин “питон.jpg”



FROM centos:7

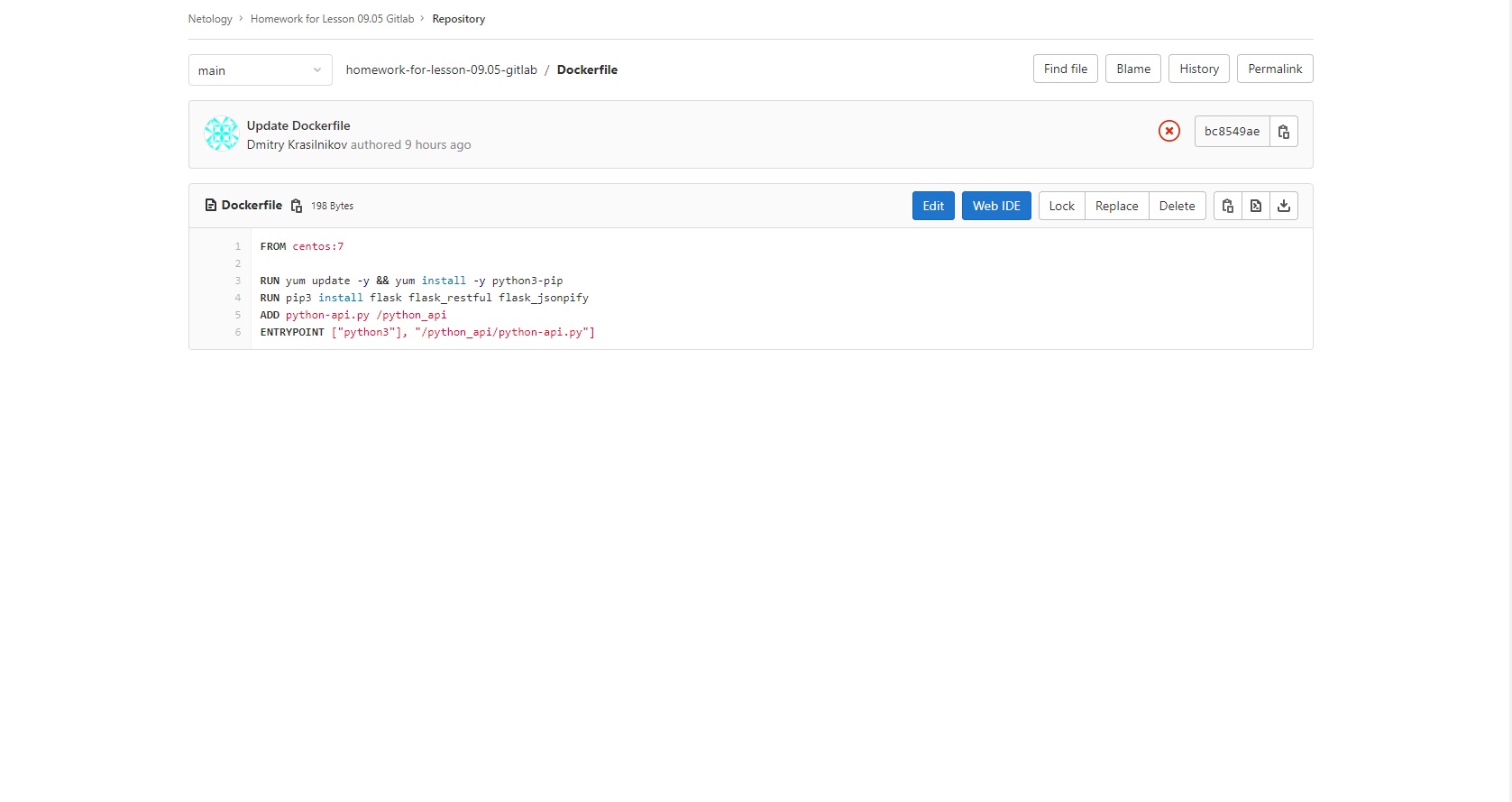
RUN yum update -y && yum install -y python3-pip

RUN pip3 install flask flask\_restful flask\_jsonpify

ADD python-api.py /python\_api

ENTRYPOINT ["python3"], "/python\_api/python-api.py"]

См. скрин “dockerfile.jpg”



Пишем

image: docker:20.10.5

services:

- docker:20.10.5-dind

stages:

- build

- deploy

build:

stage: build

script:

- docker login -u $CI\_REGISTRY\_USER -p $CI\_REGISTRY\_PASSWORD $CI\_REGISTRY

- docker build -t $CI\_REGISTRY/gitlab.com/devopstaan/netology-example/image:latest .

except:

- main

build&deploy:

stage: deploy

script:

- docker login -u $CI\_REGISTRY\_USER -p $CI\_REGISTRY\_PASSWORD $CI\_REGISTRY

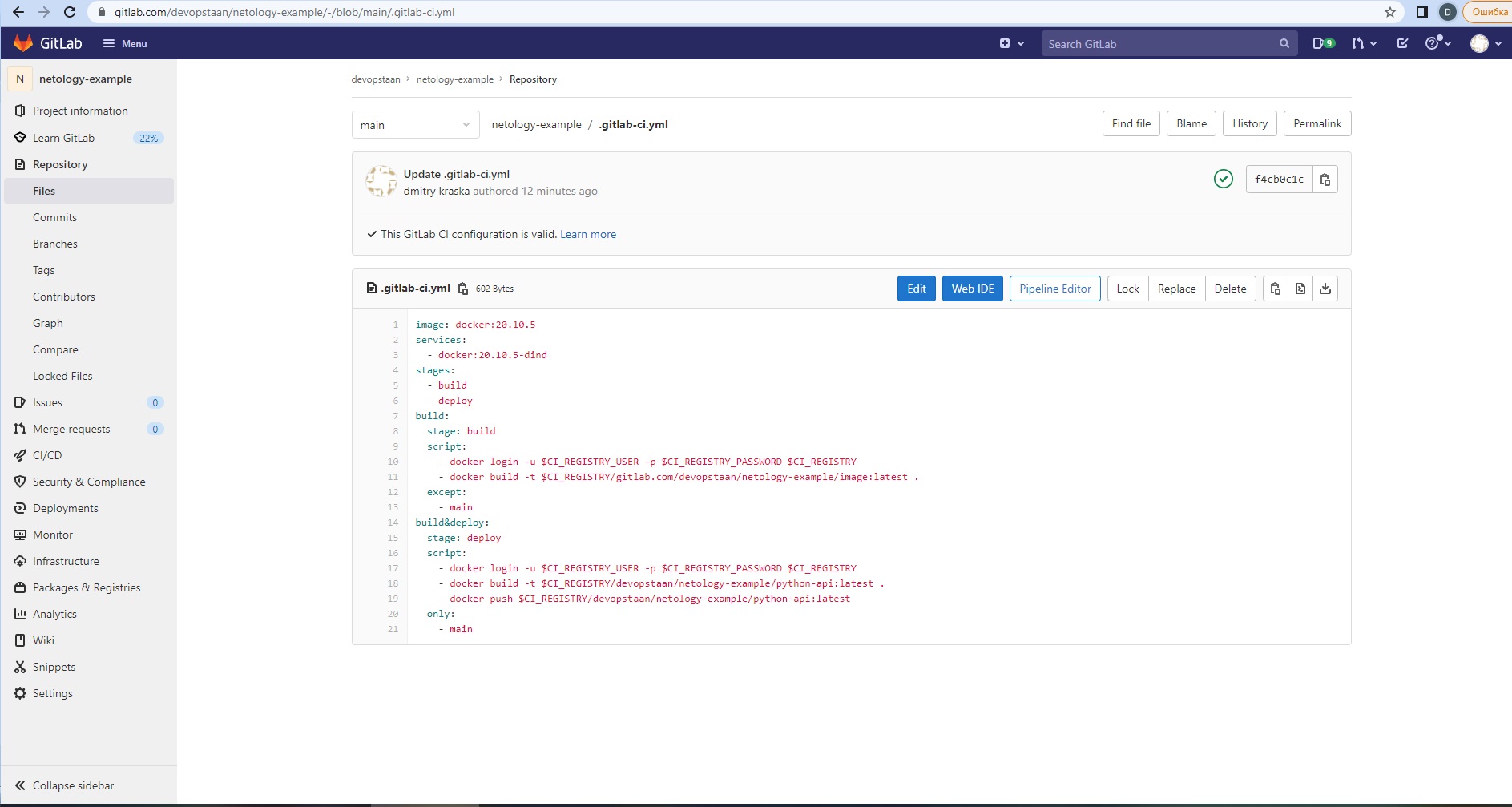
- docker build -t $CI\_REGISTRY/devopstaan/netology-example/python-api:latest .

- docker push $CI\_REGISTRY/devopstaan/netology-example/python-api:latest

only:

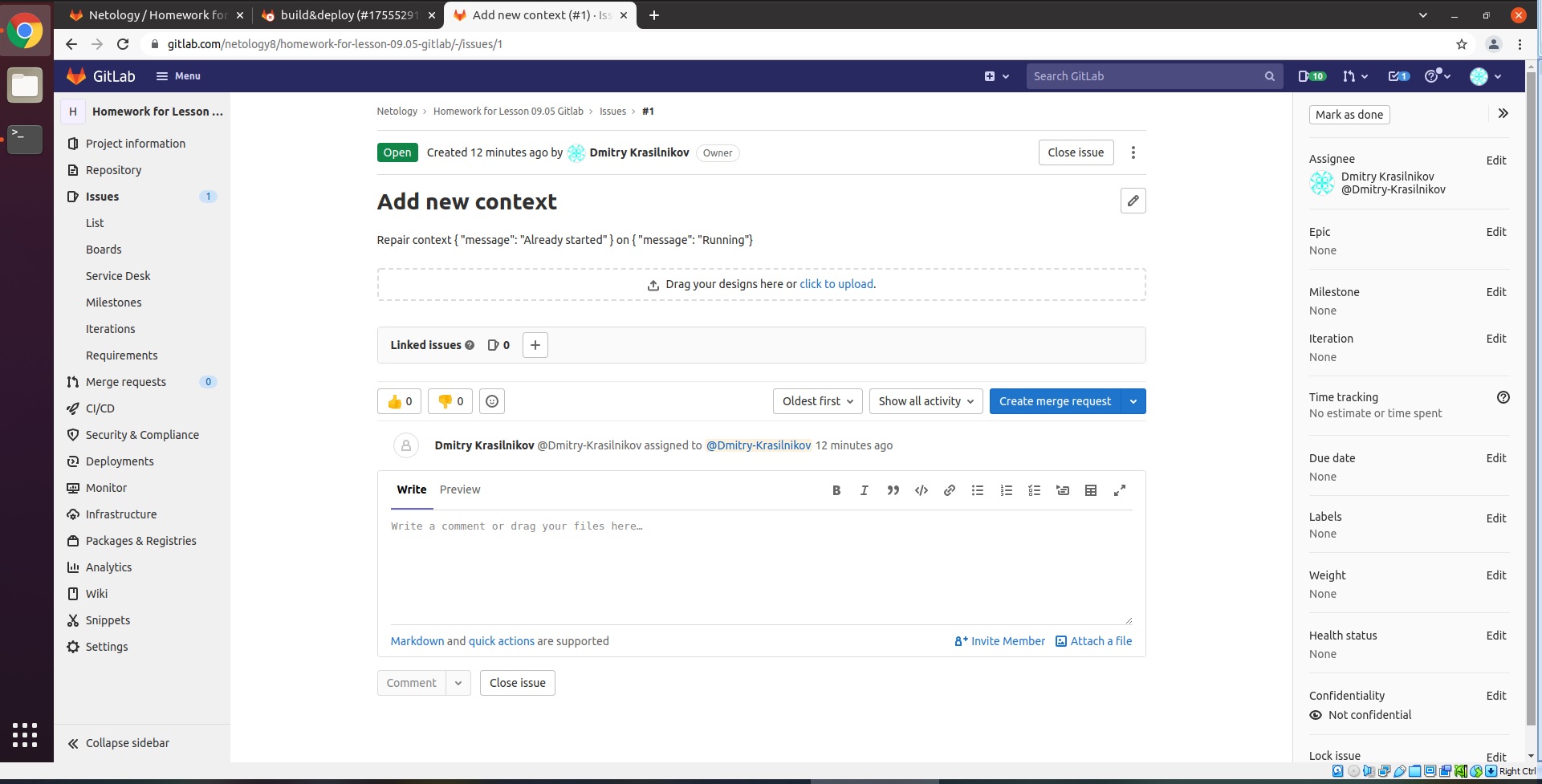
- main

См. скрин “CI.jpg”

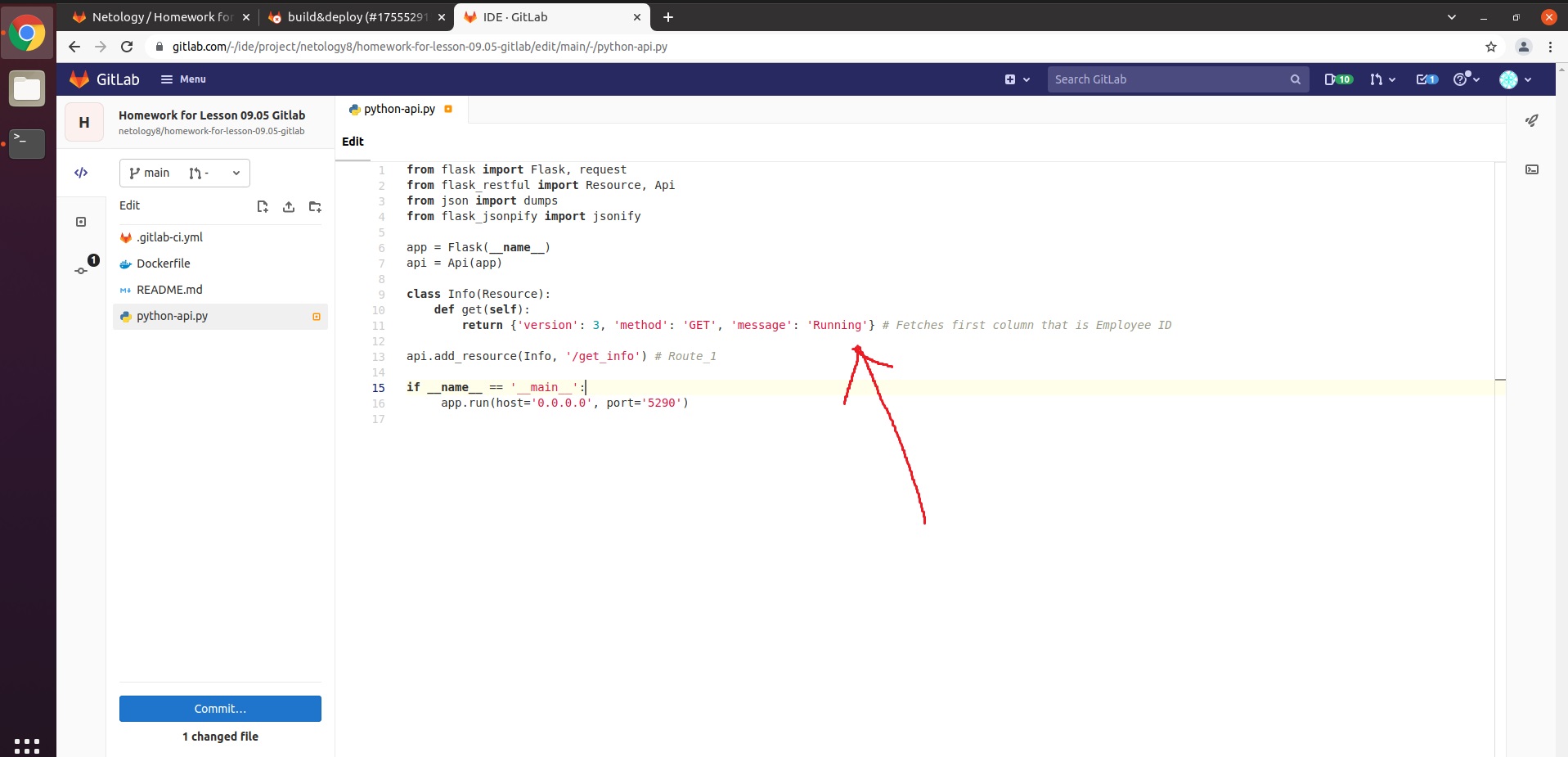


На нас повесили задачу issues

См. скрин “create issues.jpg”



Залетаем – меняем статус на running



Залетаем в merge request – мержим ветку

