# Введение

Для выполнения этой и последующих лабораторных работ нам потребуется подключение к удалённому серверу gitlab и другим. Для этого необходимо произвести несколько настроек.

## Настройка своего компьютера

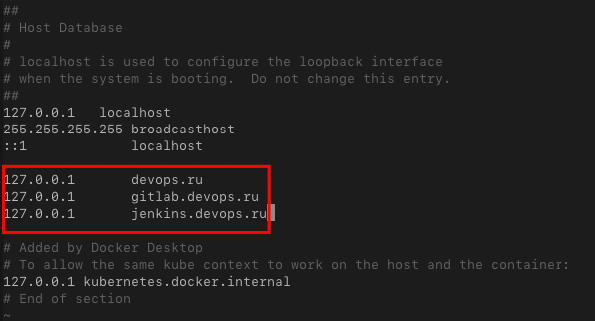
### Настройка предустановленных доменных имен

#### Для подключения к серверам необходимо указать ряд доменных имён, для которых не существует DNS записи. Адрес, который требуется указать, будет передан преподавателем.

#### Linux и MacOS:

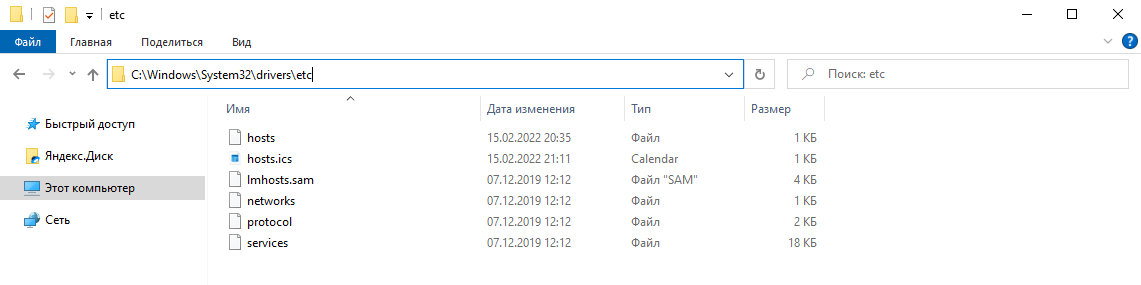
Открываем с помощью терминала файл по пути */etc/hosts* и добавляем все выделенные записи.



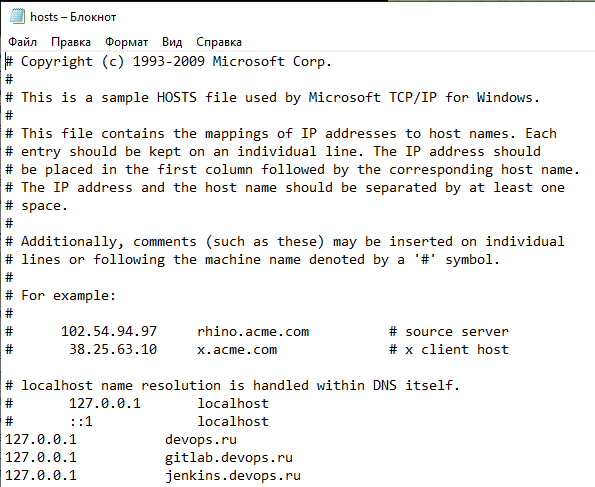


#### Windows:

Переходим по пути “*C:\Windows\System32\drivers\etc*”.



Открываем в блокноте файл hosts в режиме администратора и добавляем запись по аналогии с Linux.

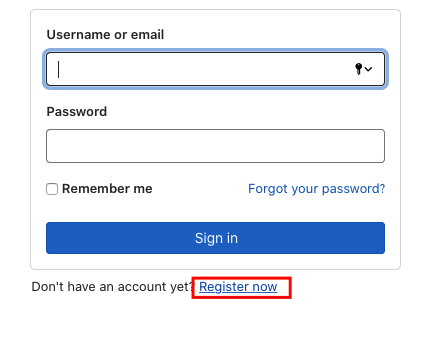


Теперь все готово!

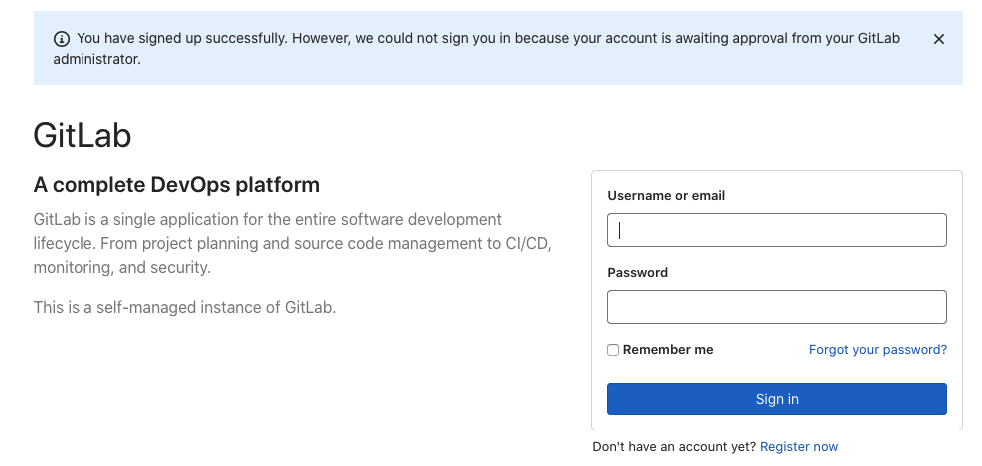
## Регистрация в удаленном GitLab.

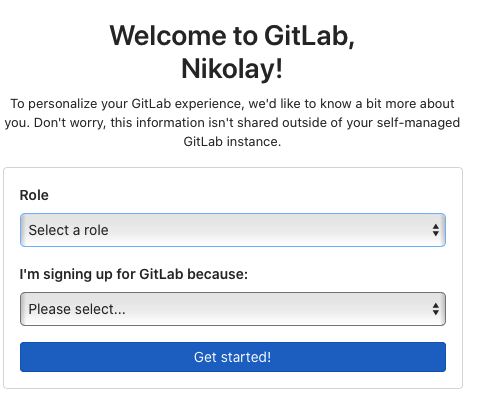
Переходим по ссылке: [*gitlab.devops.ru*](http://gitlab.devops.ru) и мы попадаем в веб интерфейс развернутого на удаленном компьютере GitLab.

Далее, как в первой работе, регистрируемся.

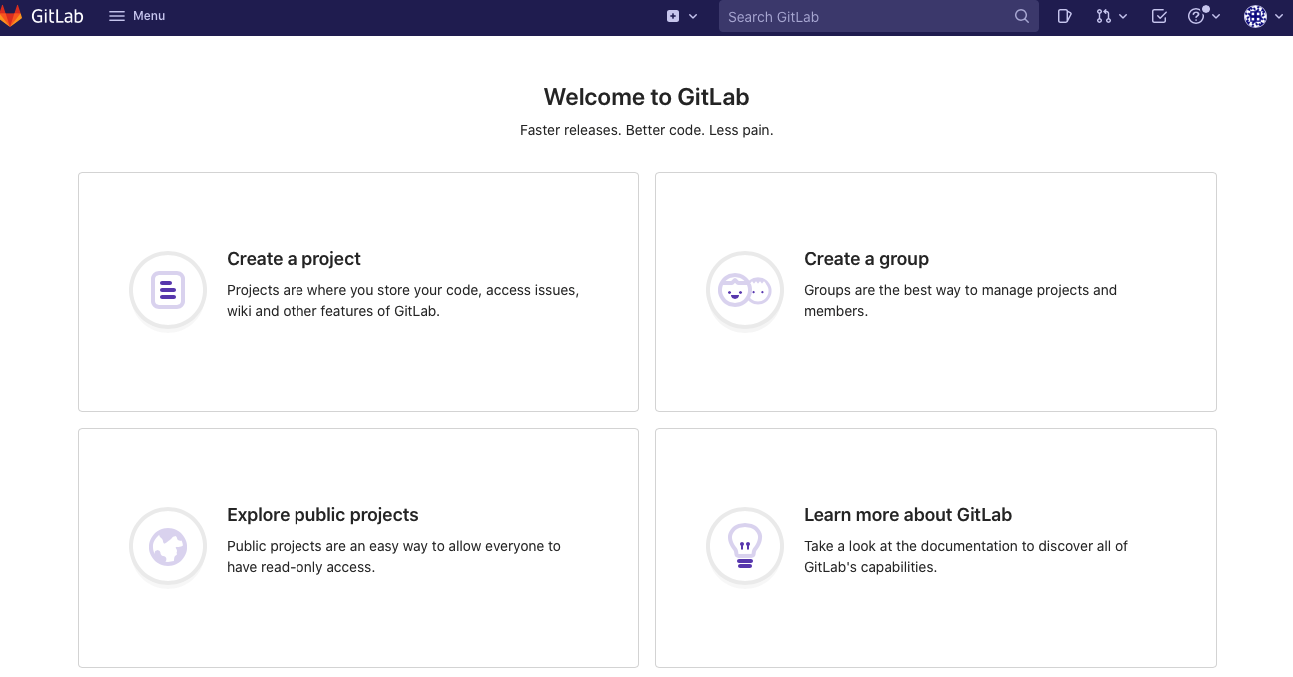


После регистрации нас опять перебросит на страницу авторизации. Внимательно читаем появившееся предупреждение. Сначала нам будет необходимо подтвердить вашу учетную запись, после этого вы сможете авторизоваться.



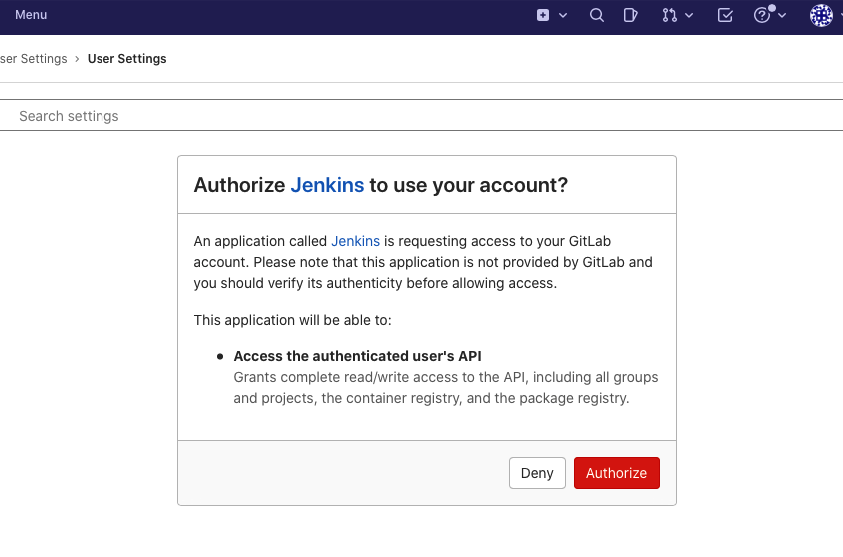


Теперь мы попадаем на стартовую страницу GitLab.



## Jenkins

Переходим по ссылке: [jenkins.devops.ru](http://jenkins.devops.ru). Однако вместо привычного интерфейса у нас снова открывается GitLab, но с предложением привязать свою учетную запись GitLab к Jenkins



Нажимаем на “*Authorize*”

Jenkins готов к работе.

## 

## 

## 

## Задание

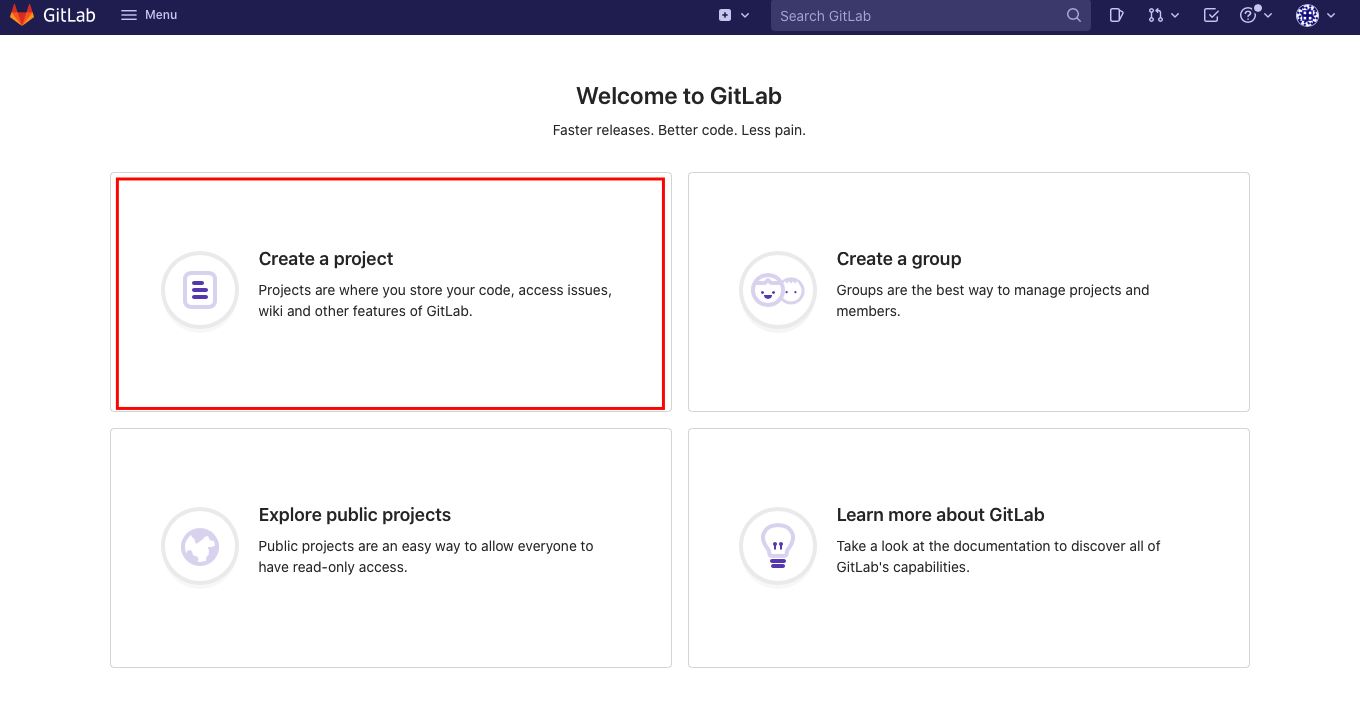
Необходимо будет создать простой REST сервис (без GUI) и написать к нему тесты (желательно воспользоваться встроенным в фреймворк инструментами, либо pytest + requests, что не очень правильно).

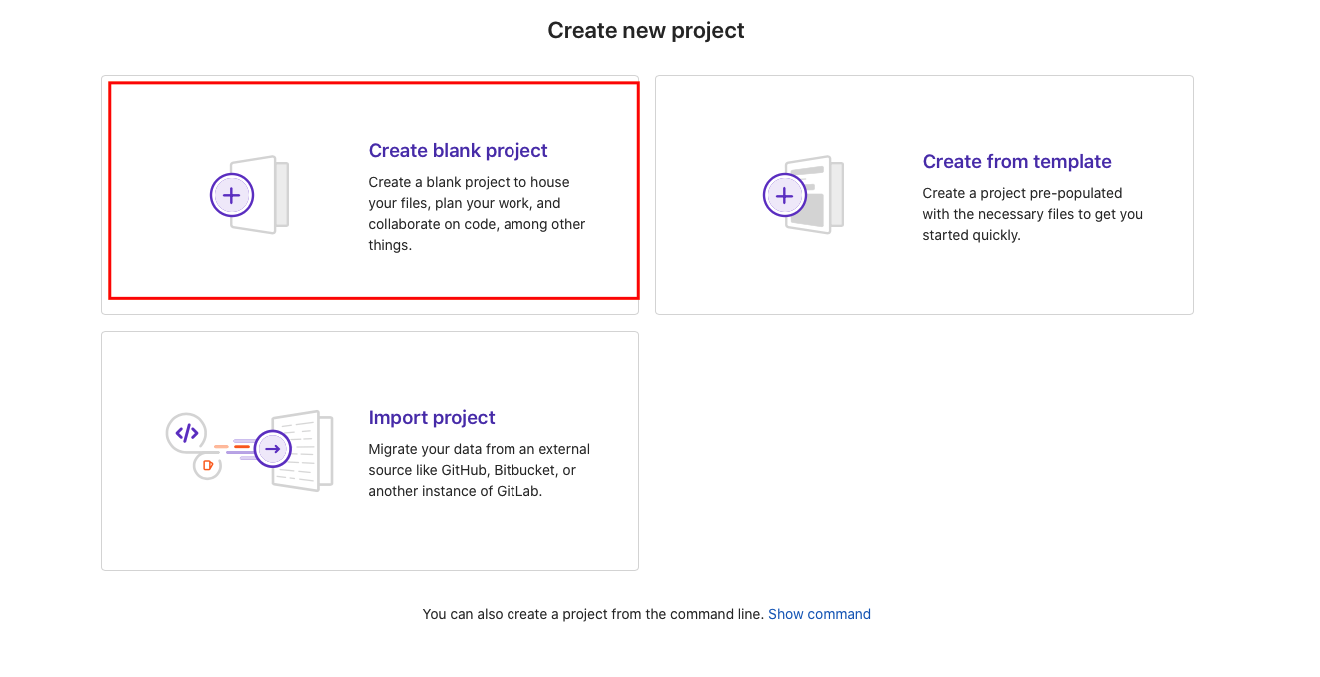
Выбор серверной библиотеки и задания осуществляется по номеру в журнале в следующей таблице (10 в списке снова берет 1 и т.д.):

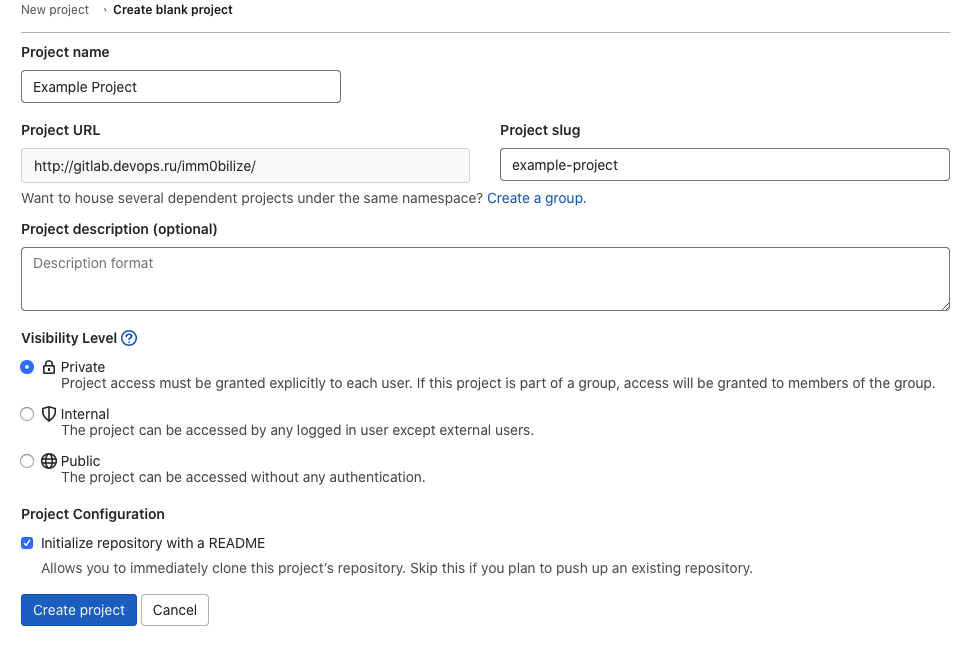
|  |  |  |
| --- | --- | --- |
| № | Фреймворк | Задание |
| 1 | FastAPI | Калькулятор |
| 2 | FastAPI | Форматирование текста |
| 3 | FastAPI | N-ое число ряда простых чисел или ряда Фибоначчи |
| 4 | Flask | Калькулятор |
| 5 | Flask | Форматирование текста |
| 6 | Flask | N-ое число ряда простых чисел или ряда Фибоначчи |
| 7 | Tornado | Калькулятор |
| 8 | Tornado | Форматирование текста |
| 9 | Tornado | N-ое число ряда простых чисел или ряда Фибоначчи |

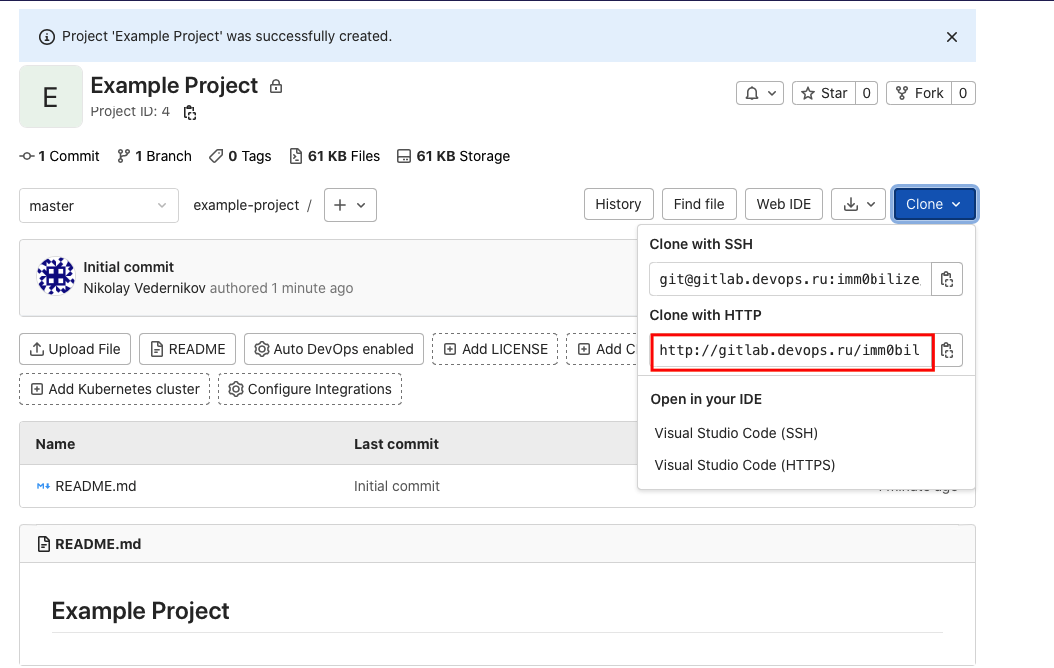
Далее необходимо будет сделать запушить полученный код в репозиторий в гитлаб.

Пример:

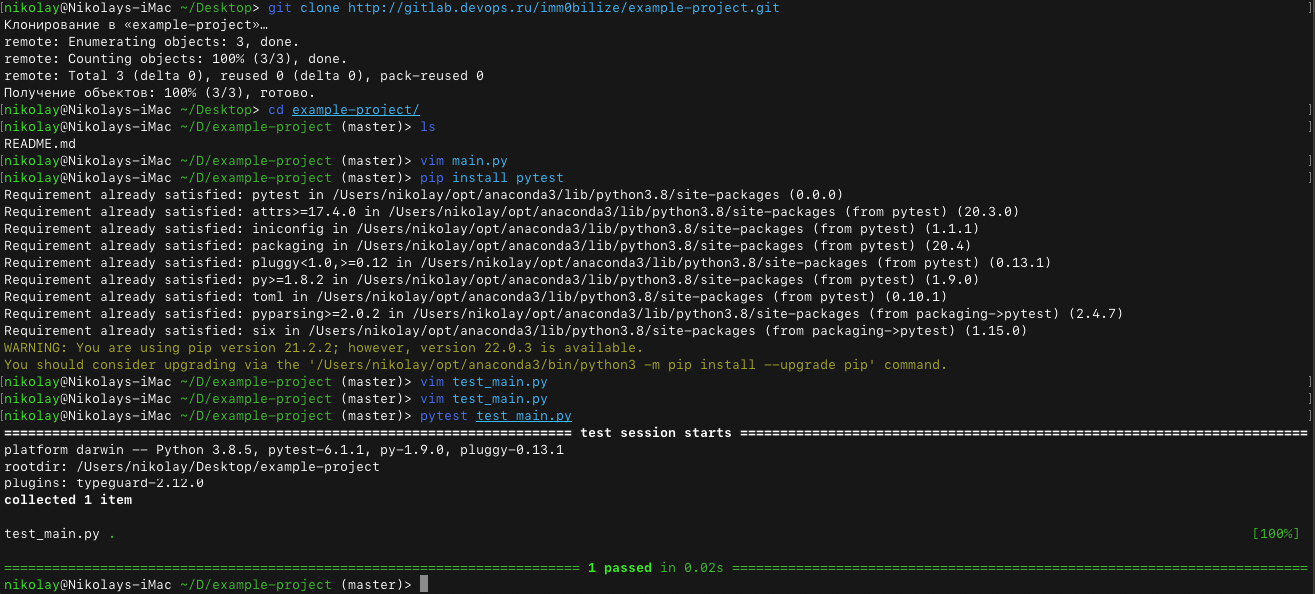








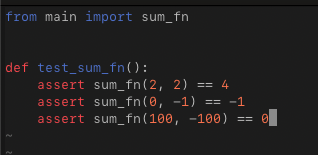
Пример подготовки кода. Обратите внимание, в файлах main.py и test\_main.py представлен код не rest сервиса, а демонстрирующий принцип разделения кода на рабочий и тестирующий. Для понимания, как тестировать rest сервис, обратитесь к документации на фреймворк



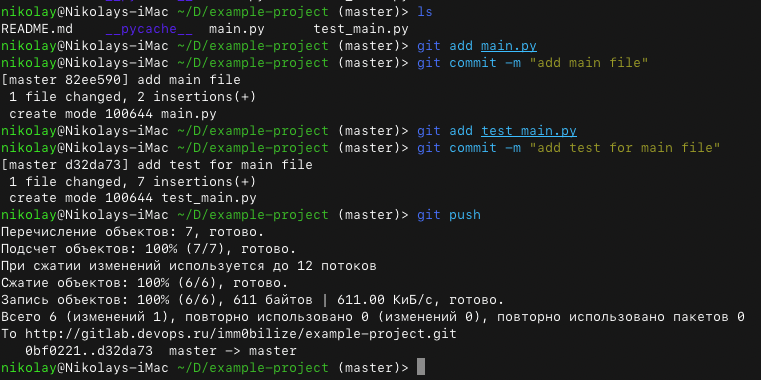
Файл main.py:



Файл test\_main.py:



Пушим получившийся результат в созданный репозиторий.



## Ссылки:

# [What is a REST API?](https://www.redhat.com/en/topics/api/what-is-a-rest-api" \l ":~:text=A%20REST%20API%20(also%20known,by%20computer%20scientist%20Roy%20Fielding.)

# [Введение в REST API — RESTful веб-сервисы](https://habr.com/ru/post/483202/)

# [Общение микросервисов: REST, JSON, GraphQL или gRPC (Их история появления и зачем они нужны) видео](https://youtu.be/o7ogFTMJW1A)

1. [Документация по FastAPI](https://fastapi.tiangolo.com/tutorial/first-steps/)
2. [Документация по Flask](https://flask.palletsprojects.com/en/2.0.x/tutorial/layout/)
3. [Документация по Tornado](https://www.tornadoweb.org/en/stable/)