ФЕДЕРАЛЬНОЕ АГЕНТСТВО ЖЕЛЕЗНОДОРОЖНОГО ТРАНСПОРТА

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«ИРКУТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ПУТЕЙ СООБЩЕНИЯ

(ФГБОУ ВО ИрГУПС)»

Факультет «Управление на транспорте и информационные технологии»

Кафедра «Информационные системы и защита информации»

Дисциплина «Инструментальные средства разработки ИС»

Лабораторная работа №2

Отчет по лабораторной работе

ЛР.420000.09.03.02.ПЗ

Выполнил: Проверила:

студент гр. ИС.1-21-1-2 Береснева Н.М.

Безгубов А.В.

Иркутск 2024

Содержание

[GitFlow 3](#_Toc183212490)

[GitHub Desktop 8](#_Toc183212491)

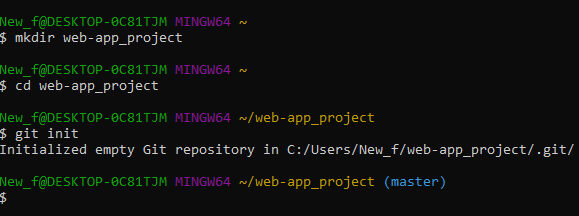
[GitHub Action 13](#_Toc183212492)

# GitFlow

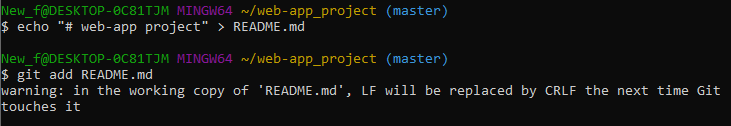
GitFlow — это популярная модель ветвления для Git, которая помогает организовать работу с репозиториями, особенно в команде. Она включает в себя несколько типов веток: master, develop, а также ветки для новых фич, исправлений и релизов. Давайте рассмотрим, как использовать GitFlow на командном уровне на примере локального репозитория.

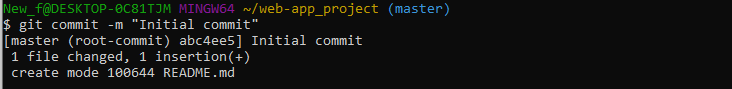
*Шаг 1: Установка Git и создание локального репозитория*

1. Для начала необходимо скачать и установить Git.



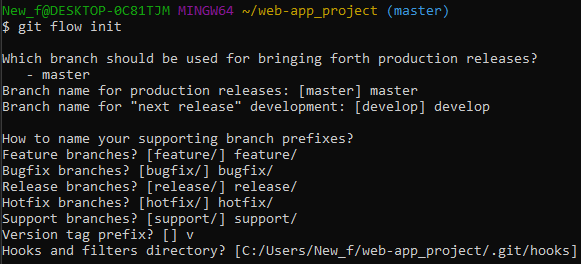
2. Для создания первого коммита необходимо создать файл README.md:





*Шаг 2: Инициализация GitFlow*

После установки GitFlow и создания репозитория, необходимо инициализировать GitFlow:



Далее необходимо указать имена веток. По умолчанию:

• Master branch name: master

• Development branch name: develop

• Feature branches prefix: feature/

• Release branches prefix: release/

• Hotfix branches prefix: hotfix/

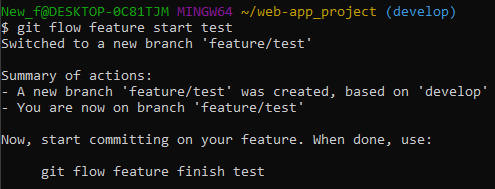
• Support branches prefix: support/

• Version tag prefix: v

Достаточно просто нажимать Enter для всех значений по умолчанию.

*Шаг 3: Создание новой фичи*

Теперь можно создать новую ветку для разработки новой функции:



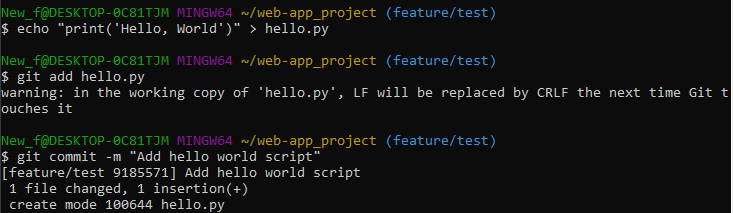


Это создаст новую ветку feature/test от ветки develop.

1. Переход на новую ветку:

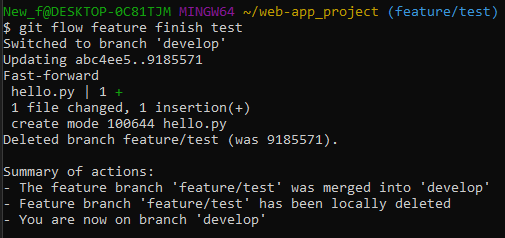


2. Создание файла и добавление кода:



*Шаг 4: Завершение работы над фичей*

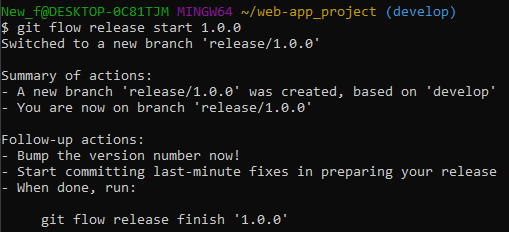
Когда работа над фичей завершена, можно завершить ее:



Это объединит ветку ветку feature/test с веткой develop и удалит ветку фичи.

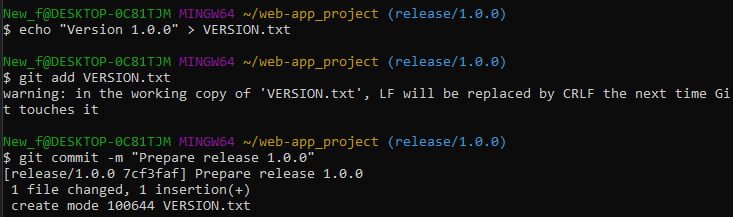
*Шаг 5: Создание релиза*

Когда проект готов к релизу, необходимо создать новую ветку релиза:

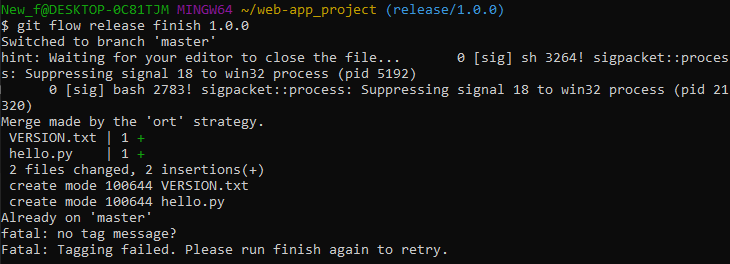


Это создаст ветку release/1.0.0 от ветки develop.

1. Добавление изменений и создание коммита:



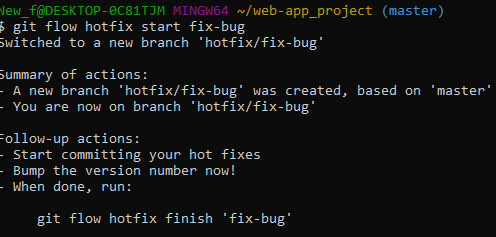
2. Завершение релиза:



Это объединит ветку релиза с master, создаст тег v1.0.0, а затем объединит изменения обратно в develop.

*Шаг 6: Работа с исправлениями (hotfix)*

Если нужно быстро исправить ошибку в продакшене, можно создать ветку hotfix:



После внесения исправлений необходимо сделать коммит и завершить hotfix:



# GitHub Desktop

GitHub — это платформа для совместной разработки и хостинга проектов. С её помощью над кодом проекта может работать неограниченное количество программистов из любых точек мира.

GitHub состоит из следующих элементов:

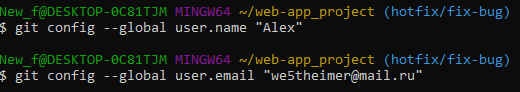
* Облачное хранилище. В нём размещаются все файлы.
* Социальная сеть для разработчиков. В ней можно найти проекты с открытым кодом от других разработчиков, практиковаться в написании кода и хранить своё портфолио.
* Система контроля версий. Фиксирует все редакции, вносимые в код тем или иным участником.

Функционал GitHub обеспечивает удобный и быстрый контроль над текущим состоянием программного продукта, а также сохраняет его последние версии и авторство внесённых изменений.

*Шаг 1: Настройка Git*

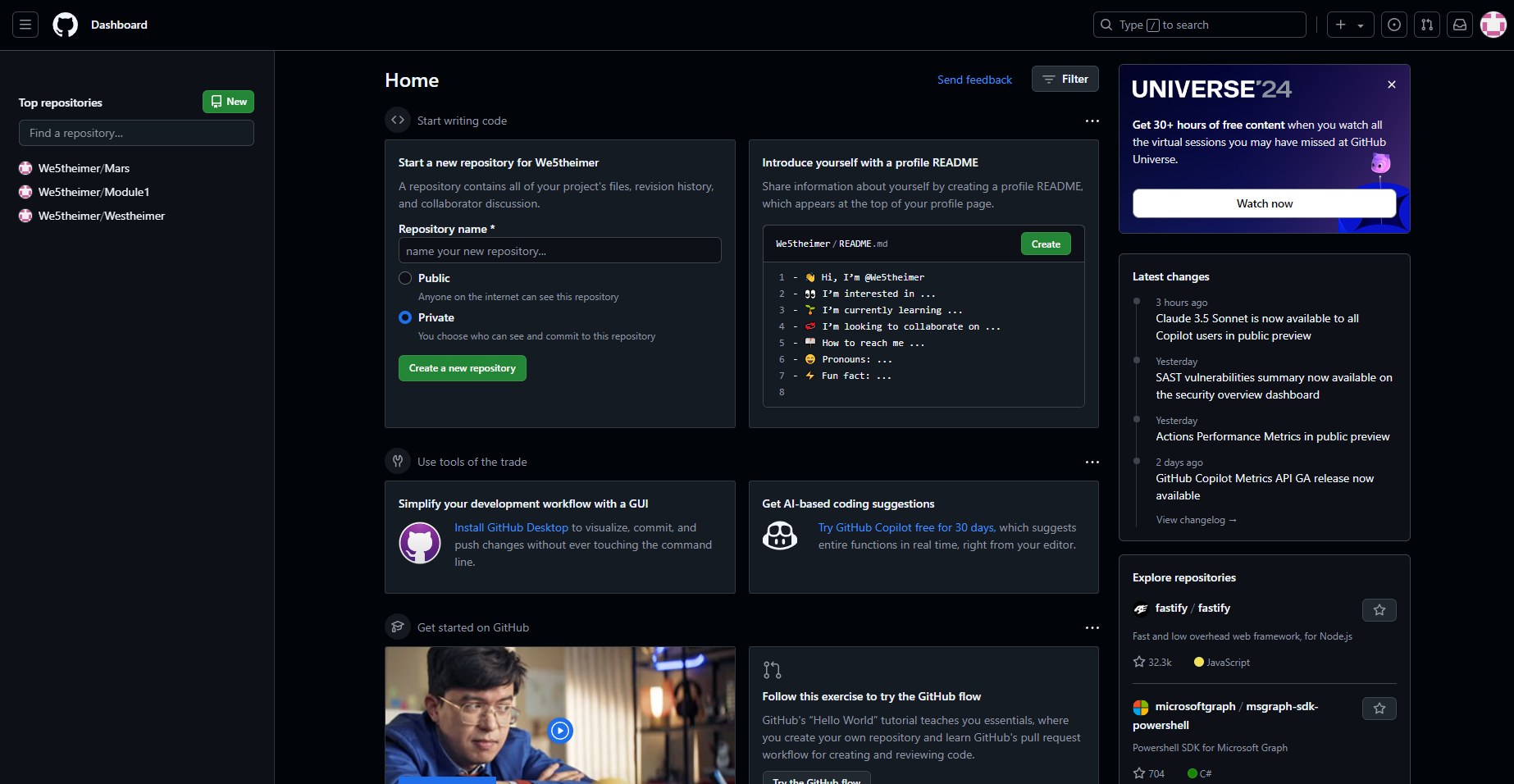
Необходимо настроить имя и email:

Эти данные будут использоваться для коммитов.

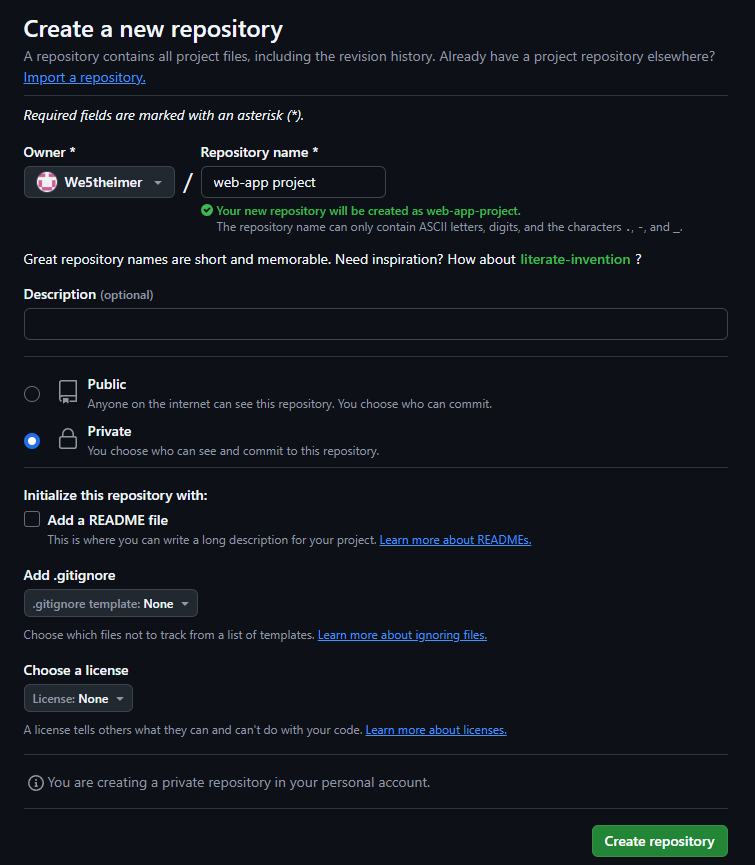


*Шаг 2: Создание удаленного репозитория на GitHub*

1. Регистрация или вход в учетную запись на GitHub.



2. Создание нового репозитория:



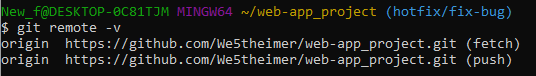
*Шаг 3: Связывание локального репозитория с удаленным*

1. Добавьте удаленный репозиторий:

После создания репозитория на GitHub будет предложено добавить удаленный репозиторий к локальному. Необходимо скопировать URL-адрес (HTTPS или SSH) и выполнить команду:



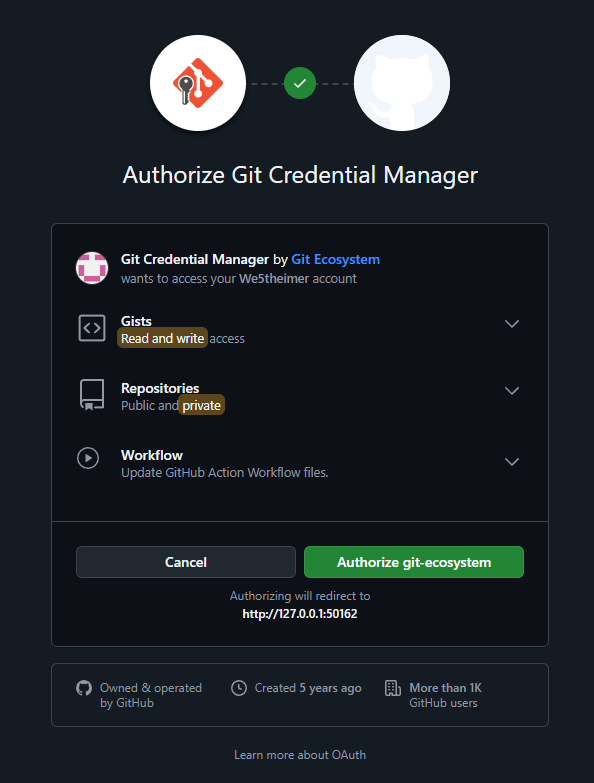
2. Далее необходимо проверить, что удаленный репозиторий добавлен:



*Шаг 4: Отправка изменений в удаленный репозиторий*

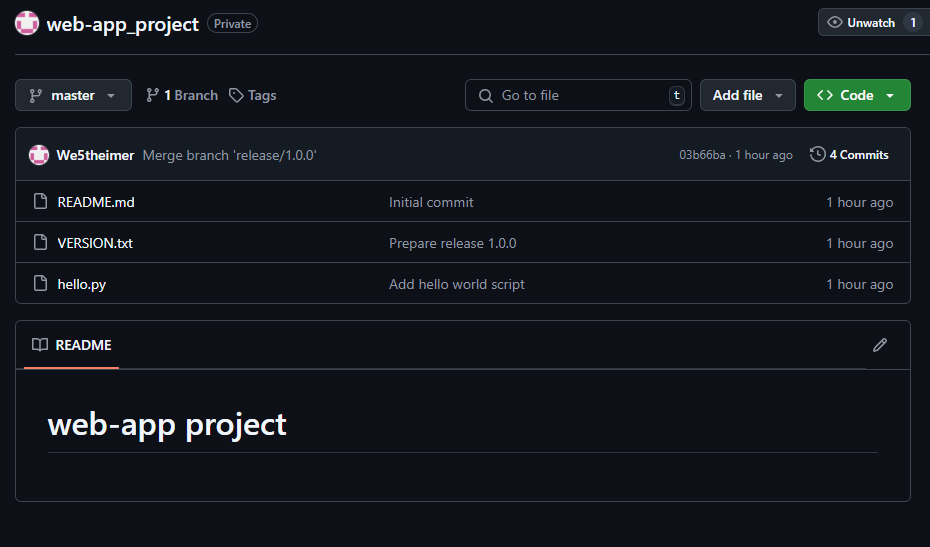
1. Отправка изменений на GitHub:





*Шаг 5: Проверка изменений на GitHub*

Далее необходимо перейти на страницу репозитория на GitHub и убедиться, что изменения были успешно загружены.

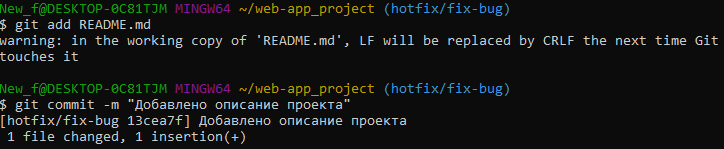


*Шаг 6: Внесение изменений и обновление на GitHub*

1. Для проверки можно внести изменения в файл (например, в README.md):



2. Необходимо сделать коммит:



3. Отправка изменений на GitHub:



Результат:



# GitHub Action

GitHub Actions — это инструмент для автоматизации рутины в области разработки программного обеспечения, автоматического тестирования, сборки и публикации приложений, который глубоко интегрирован в экосистему инструментов GitHub.

Задача: пощупать тестирование на простом случае теста (деление на ноль, чет/нечет и т.п.), отработать автоматизацию этого момента.

Структура проекта будет иметь следующий вид:

/my\_project

├── .github

│ └── workflows

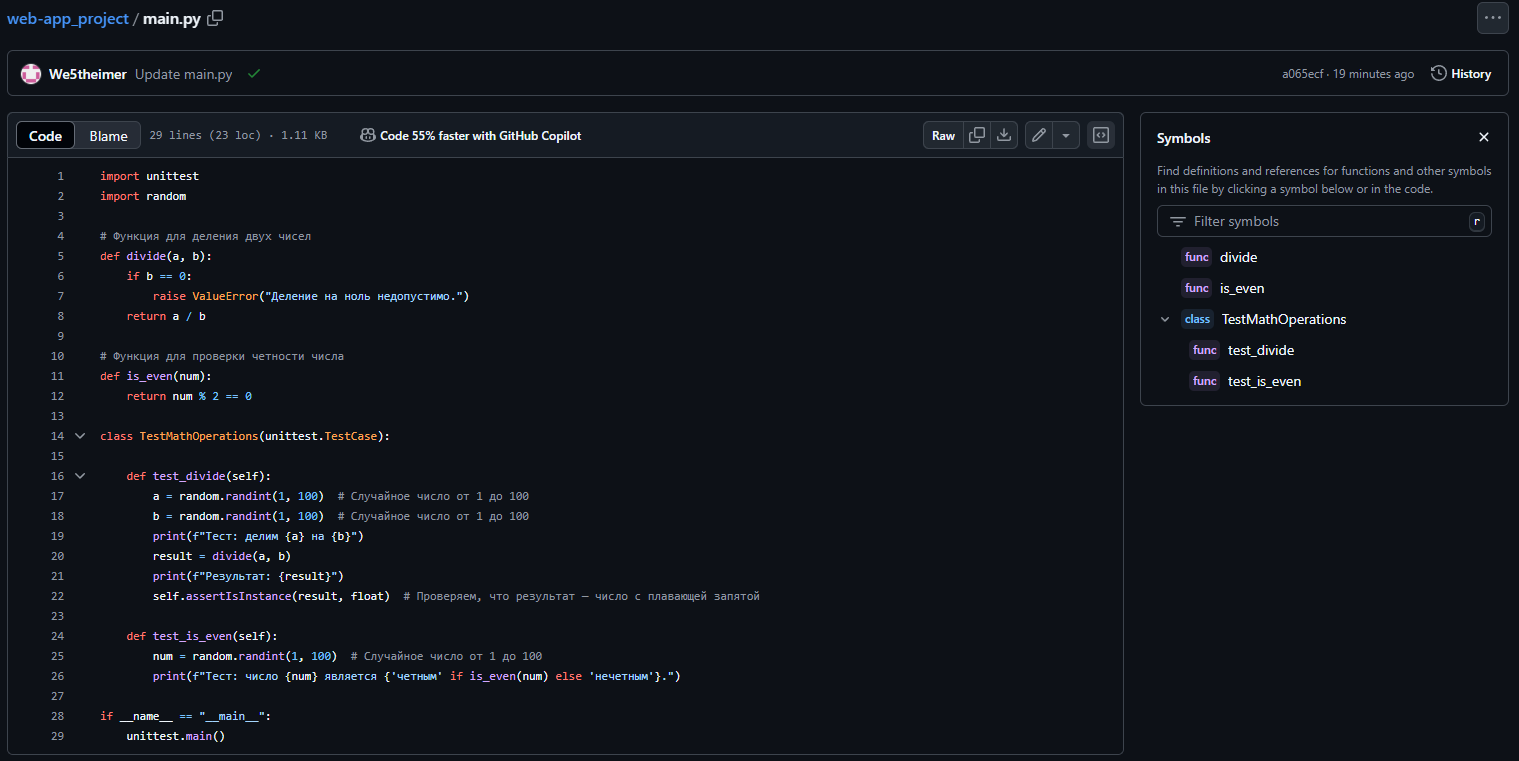
│ └── python-app.yml

├── main.py

└── test\_main.py

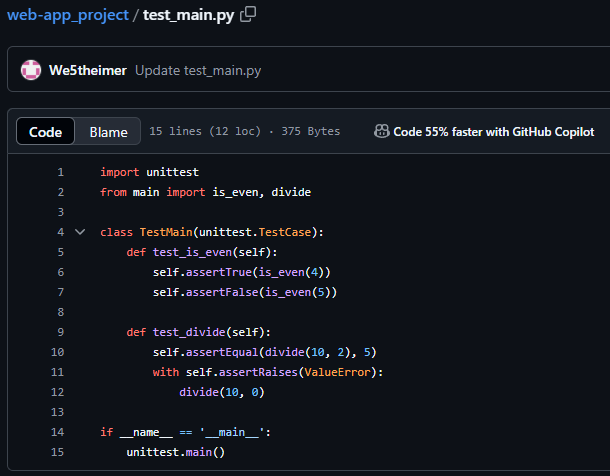
*Шаг 1: Создание функций проверки*

Создаём файл *main.py* в корневом каталоге. В этом файле будем реализовывать функции для проверки четности числа и деления.



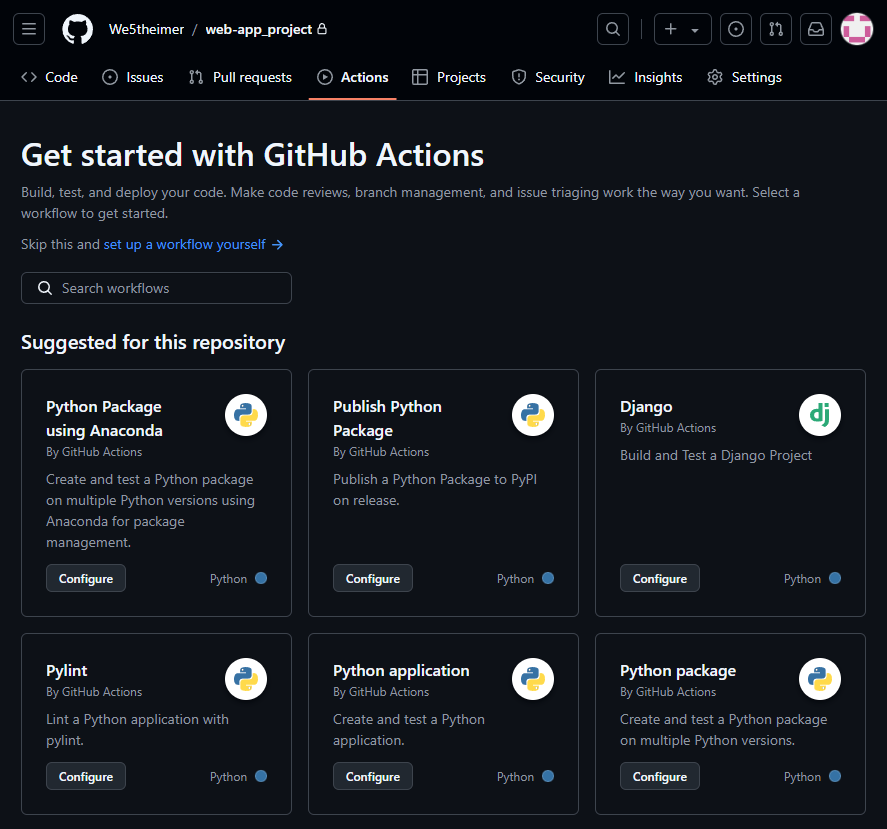
*Шаг 2: Написание тестов.*

Создаём файл *test\_main.py*. В этом файле будем писать тесты для функций *is\_even* (чет/нечет) и *divide* (деление на ноль).



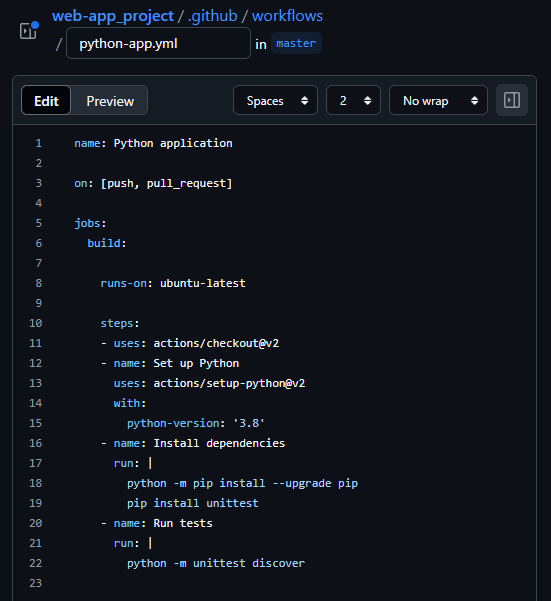
Шаг 3: GitHub Actions Workflow.

Теперь создадим файл для GitHub Actions, который будет запускать тесты. Создать файл надо по следующему пути: *.github/workflows/python-app.yml*. Можно сделать вручную или перейти во вкладку *Actions* и перейти по ссылке с названием: *set up a workflow yourself*



Даём название создаваемому файлу *python-app.yml*, пишем код, в котором указываем:

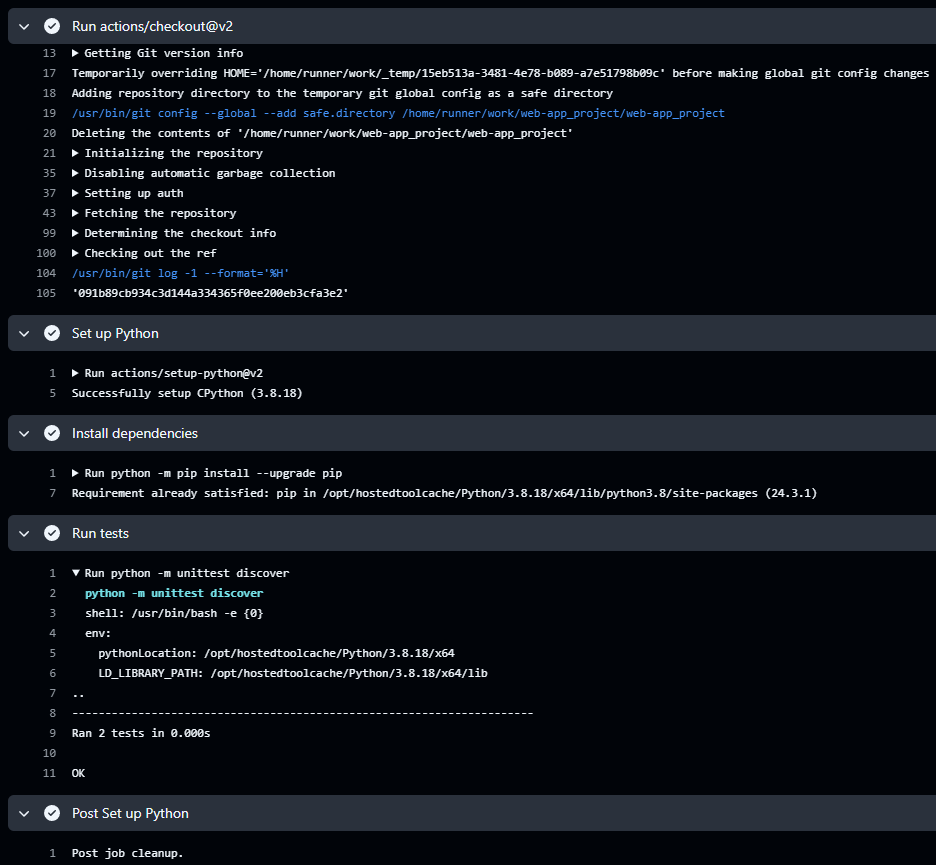
* Триггеры для запуска Action (push и pull\_request);
* Использование Ubuntu как операционной системы;
* Установка Python: установка версии Python (3.8);
* Запуск тестов: Run tests - выполняет тесты.



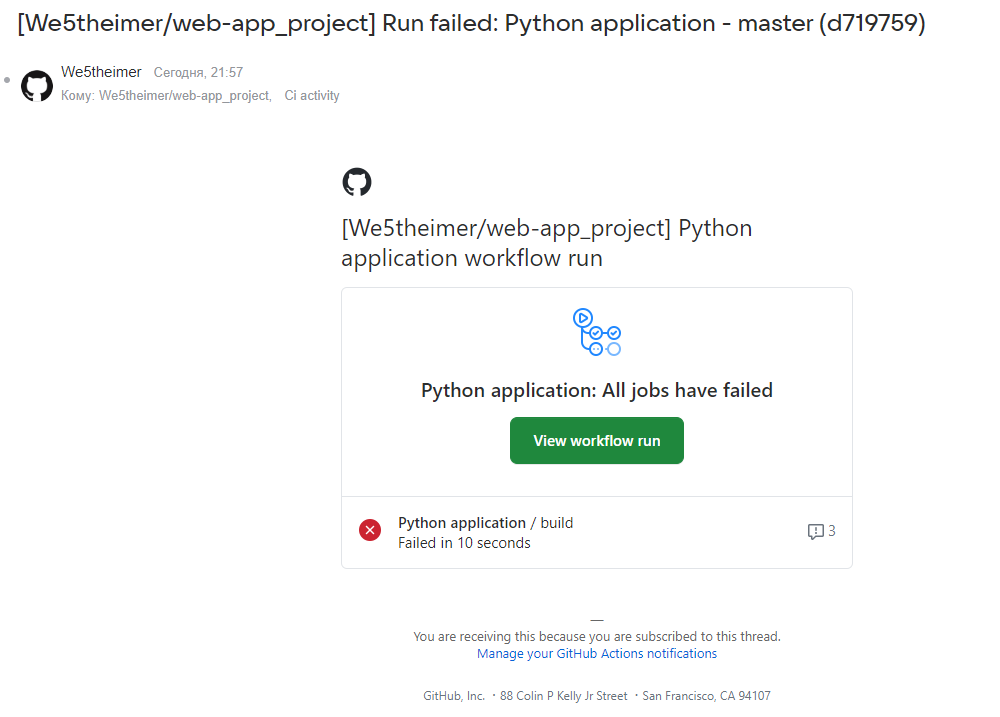
Шаг 4: Проверка тестов.

Вносим изменения в ветку main и переходим во вкладку *actions*





В случае ошибки сообщение приходит на электронную почту.



Ссылка на Github: https://github.com/We5theimer/web-app\_project