Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Тихоокеанский государственный университет»

Высшая школа кибернетики и цифровых технологий

ОТЧЕТ ПО ИНДИВИДУАЛЬНОМУ ЗАДАНИЮ

ПО КУРСУ «ПРОГРАММНАЯ ИНЖЕНЕРИЯ»

«РАБОТА С git и github»

Выполнил: студент 2 курса группы ПО(б) – 31

Магонов Александр Александрович

Проверила: ассистент ВШ КЦТ

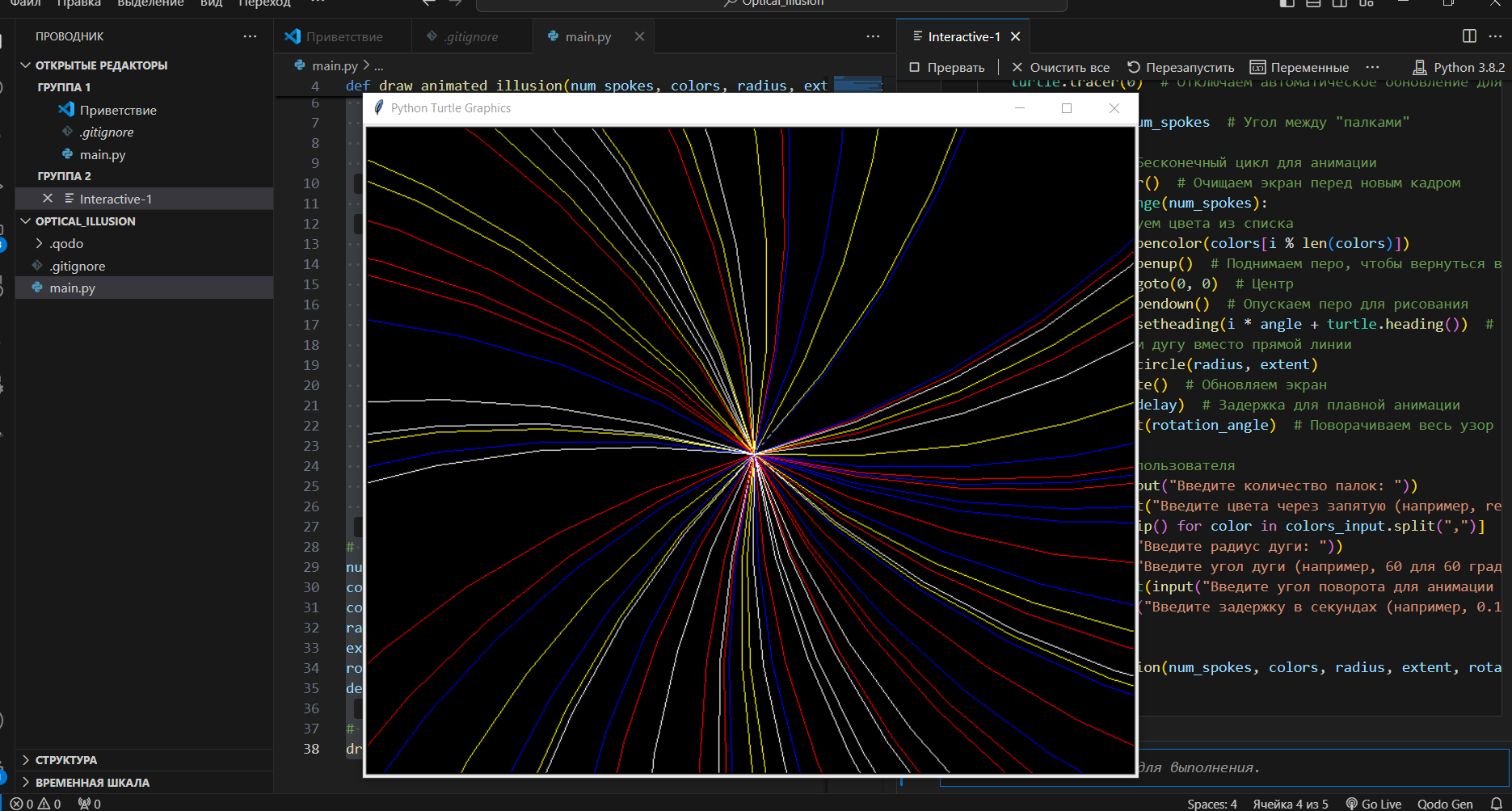
Балабасова Алиса Дмитриевна

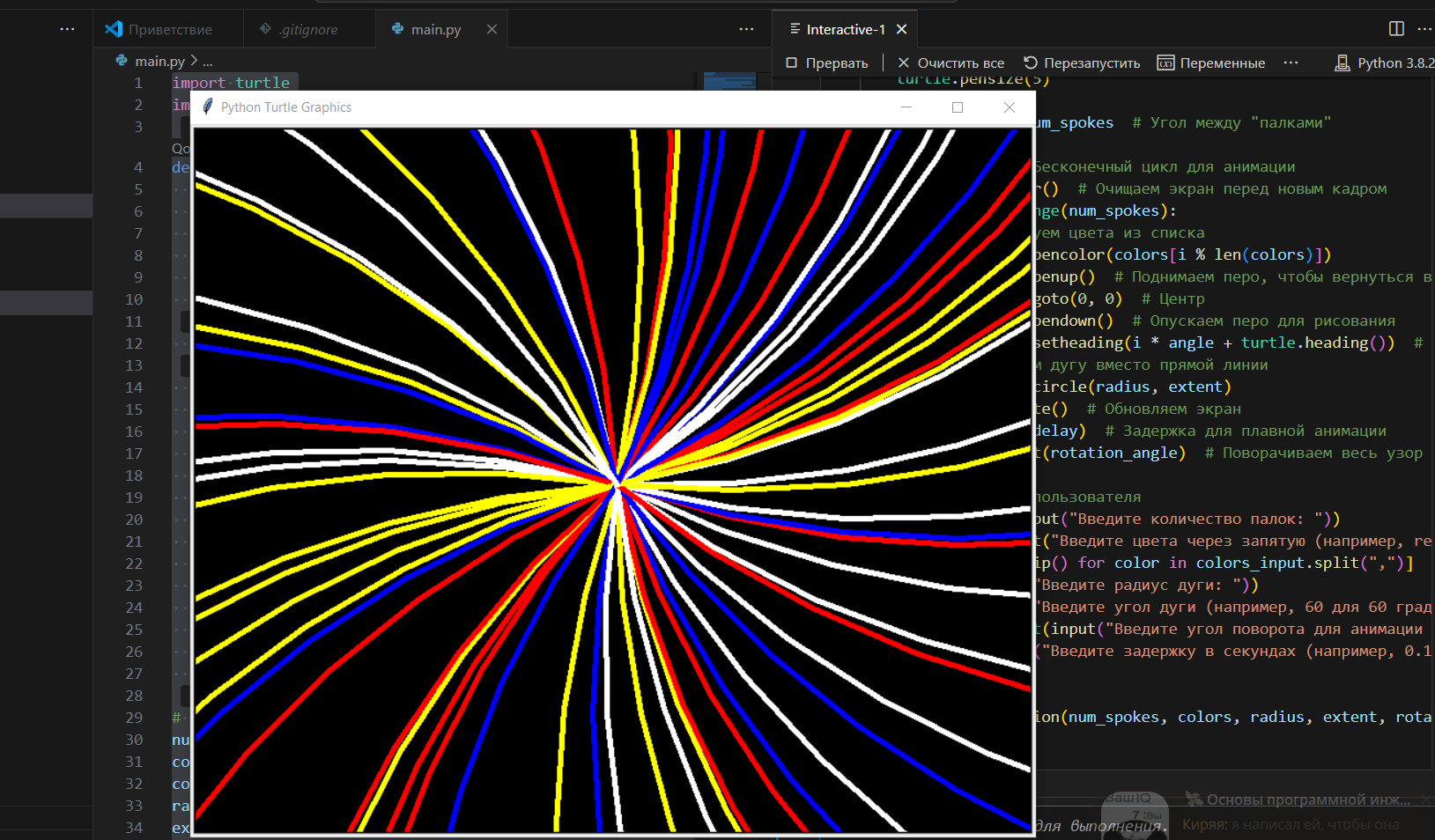
Хабаровск 2025 г.

Цель работы: освоить создание репозитория, добавление файлов, работу с ветками, форками и pull request.

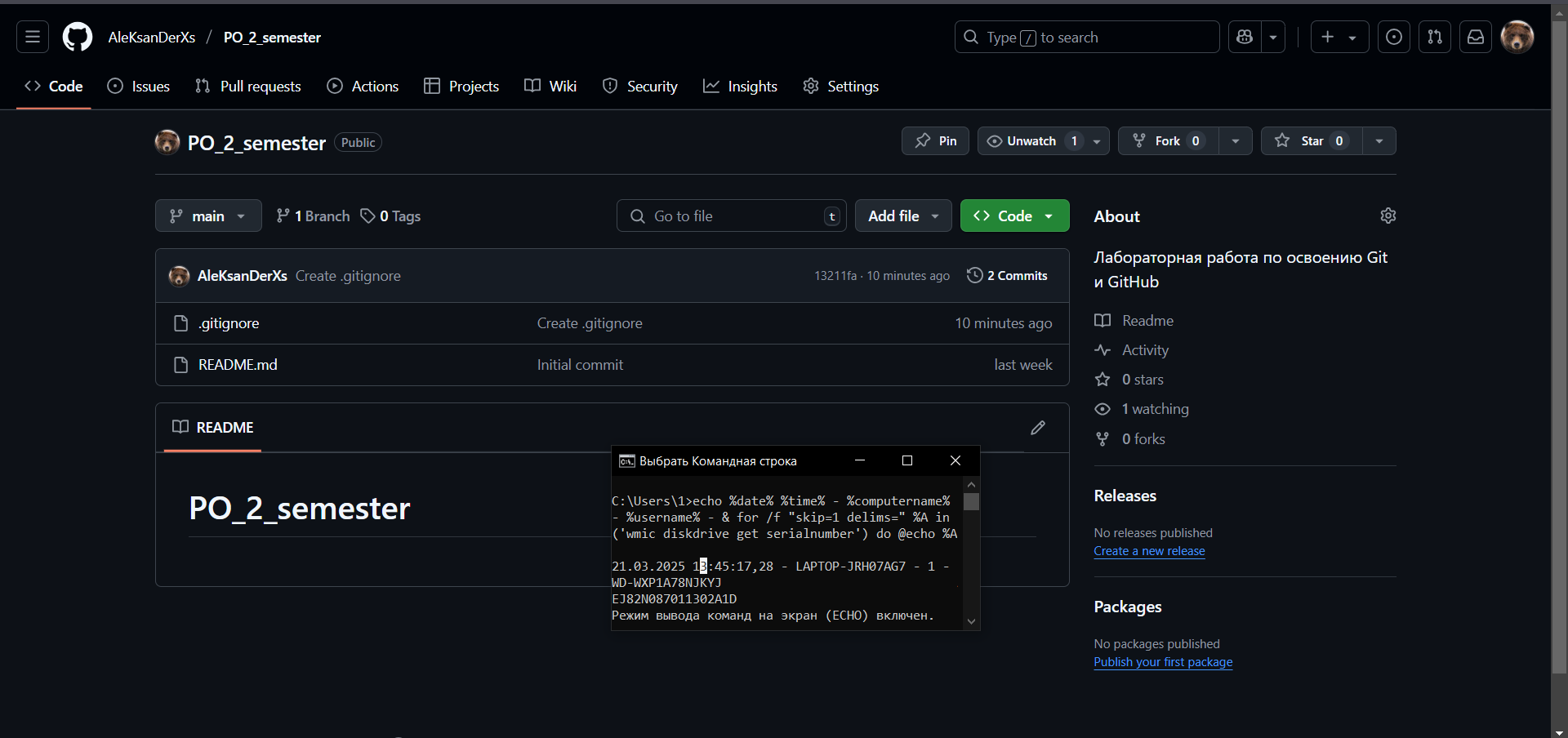
Ход работы:

Сам проект:

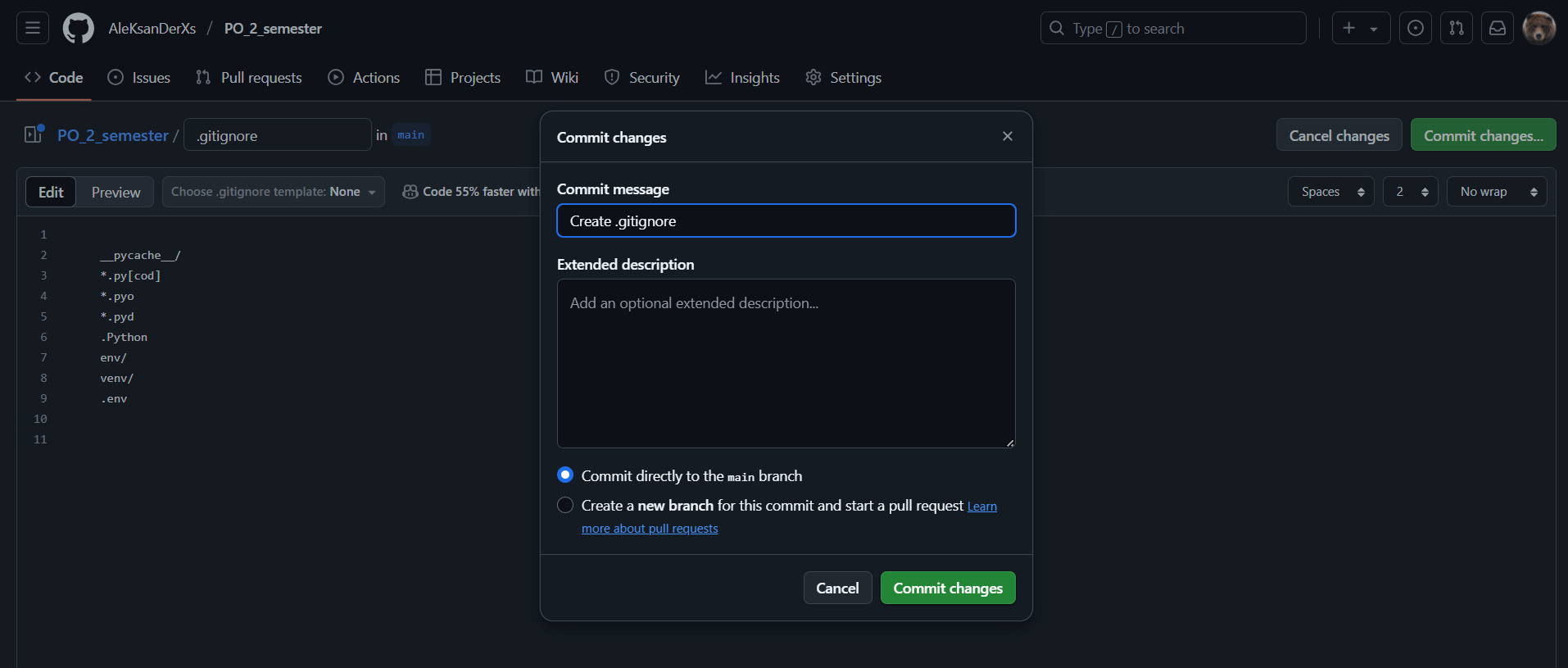




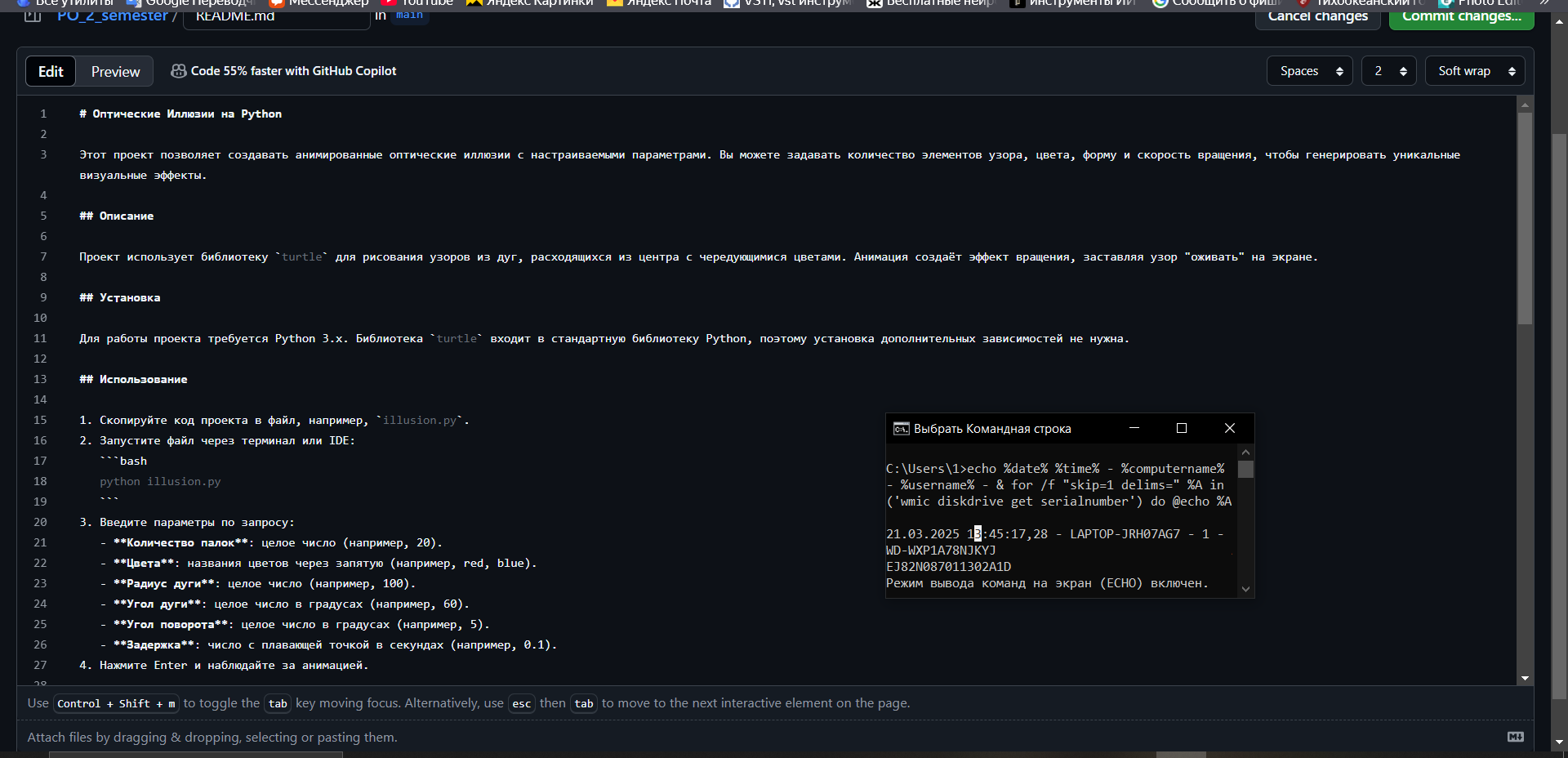
1. Создание репозитория и настройка описания



1. Создание файла .gitignore



1. Написание и добавление файла README.md в формате MarkDown:



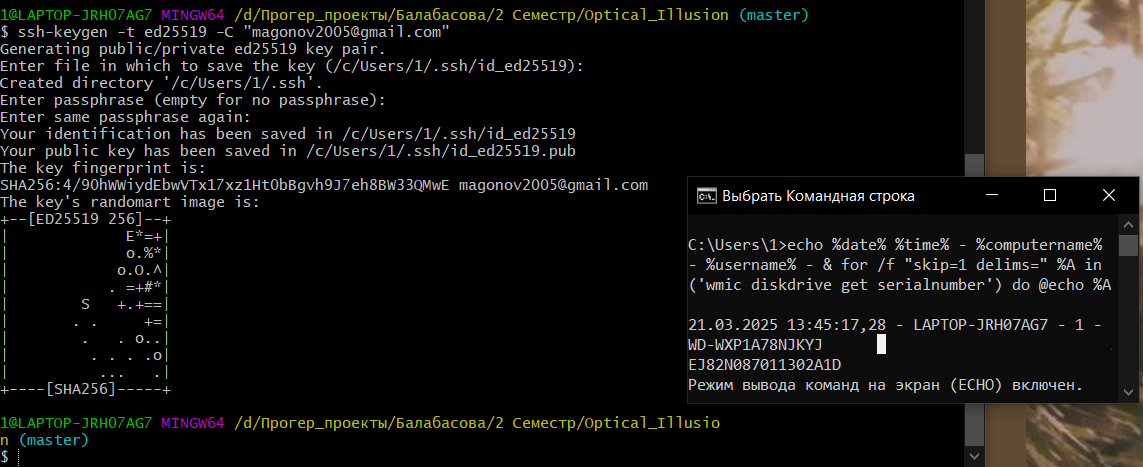
1. «Коммитим» и отправляем изменения в репозиторий (то есть наш проект), предварительно связывая его с git:



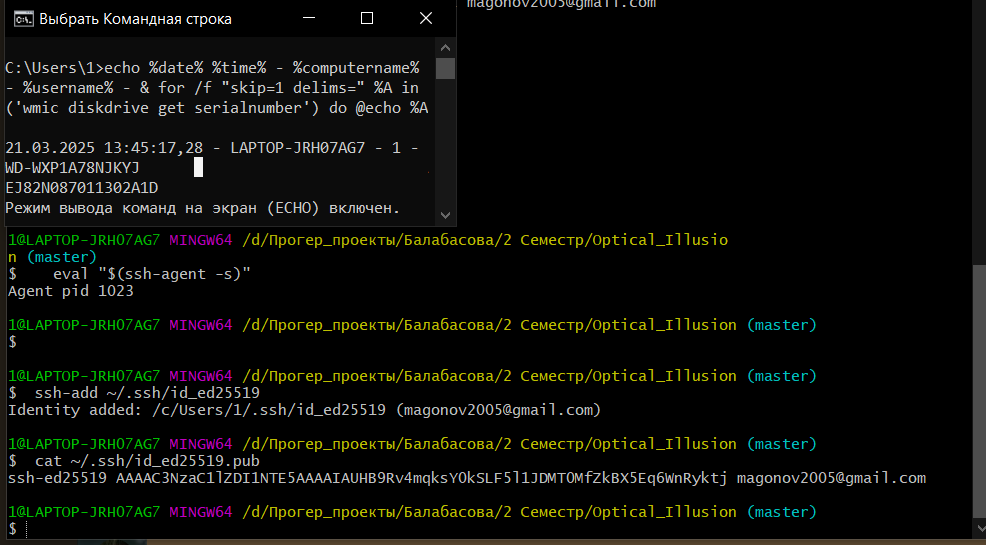
1. Настроить SSH-ключ для аутентификации с GitHub
2. Проверка на наличие ключей



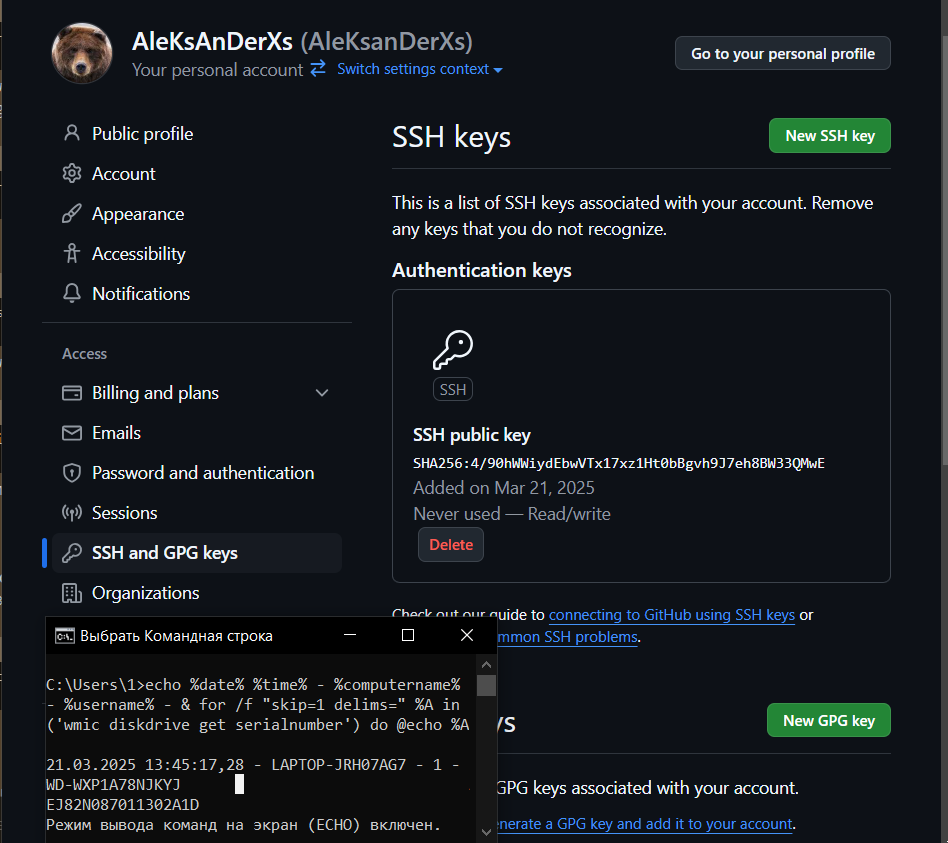
1. Ключей нет – генерируем новый.



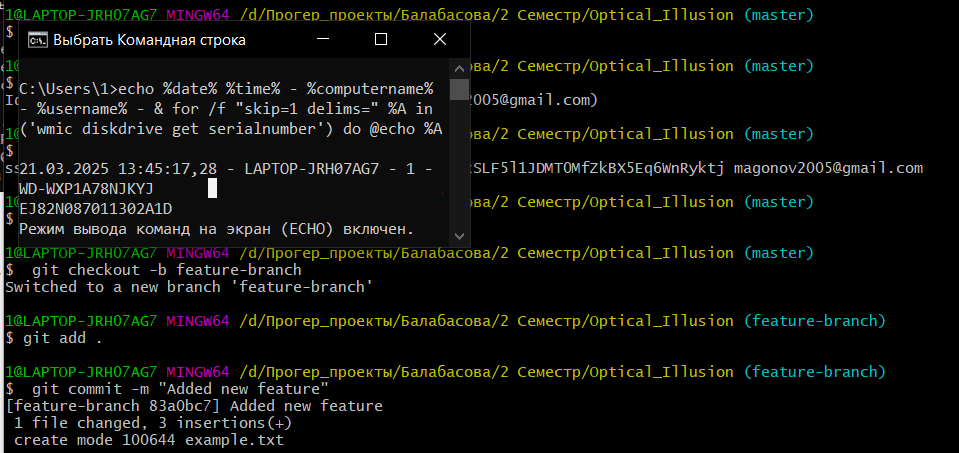
1. Добавляем и копируем публичный ключ.



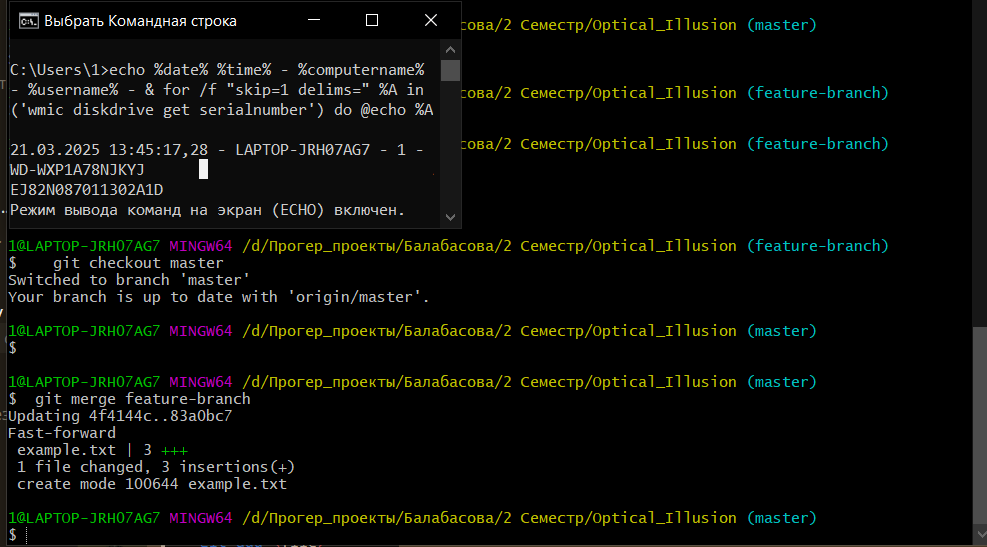
1. Копируем ключ и вставляем его в Settings -> SSH and GPG keys -> New SSH key и сохраняем.



6. Создать новую ветку "feature-branch"

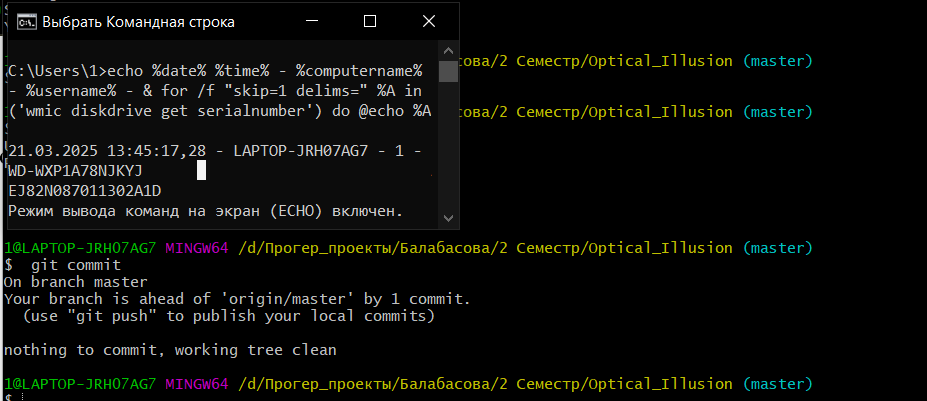


7. Переключиться на основную ветку, объединить изменения

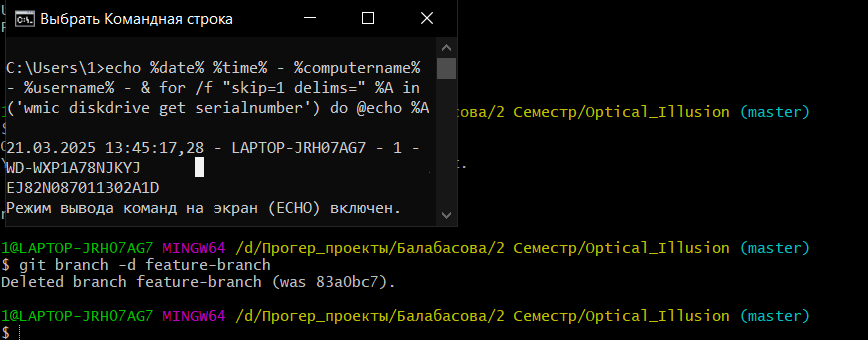


8. Разрешить возможные конфликты при слиянии

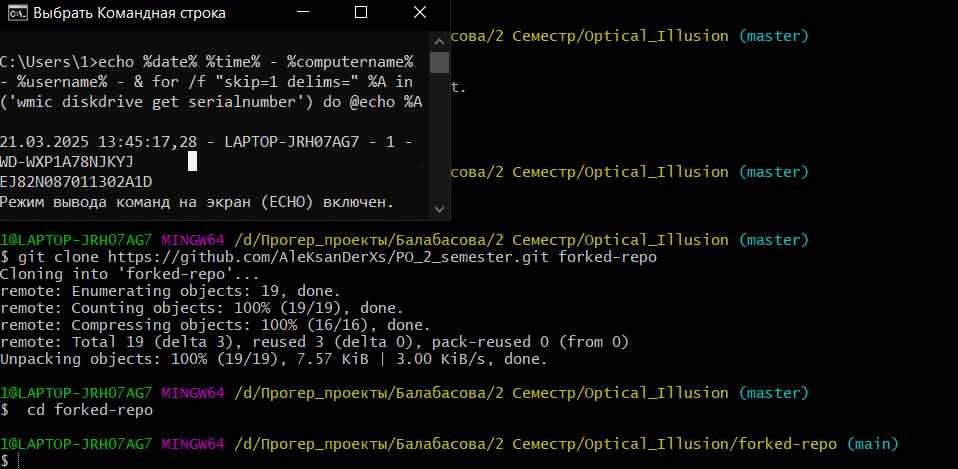
Конфликтов не возникло, разрешать нечего.



9. Удаляем ветку "feature-branch" после успешного объединения

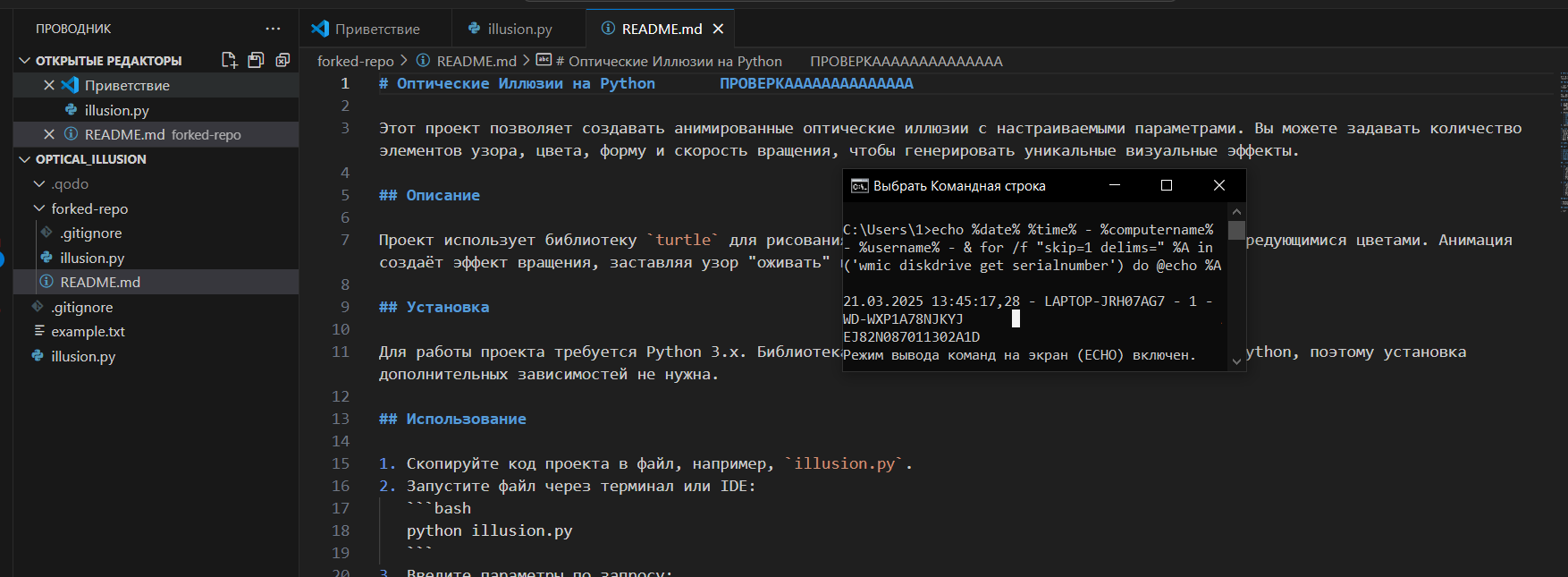


10. Сделать форк существующего учебного репозитория или своего проекта.

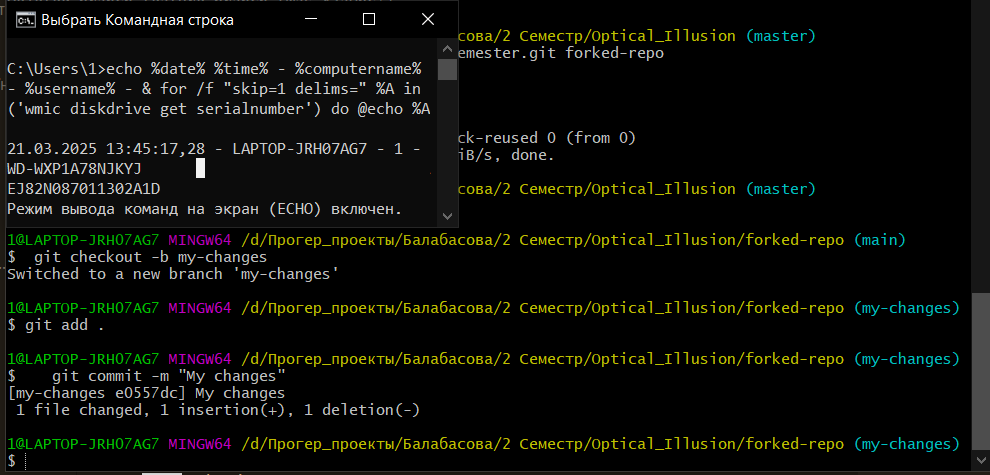


11. Внести изменения в код, закоммитить и запушить в форк

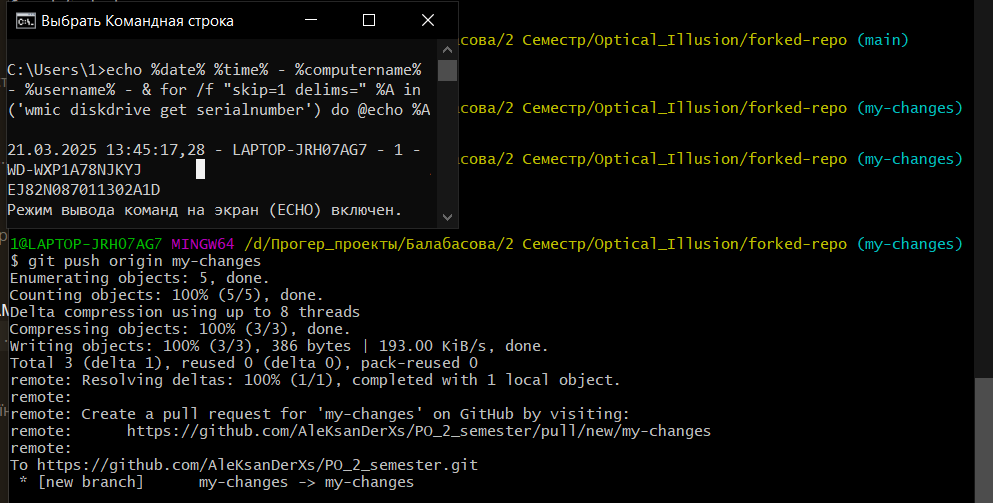
Изменяем код



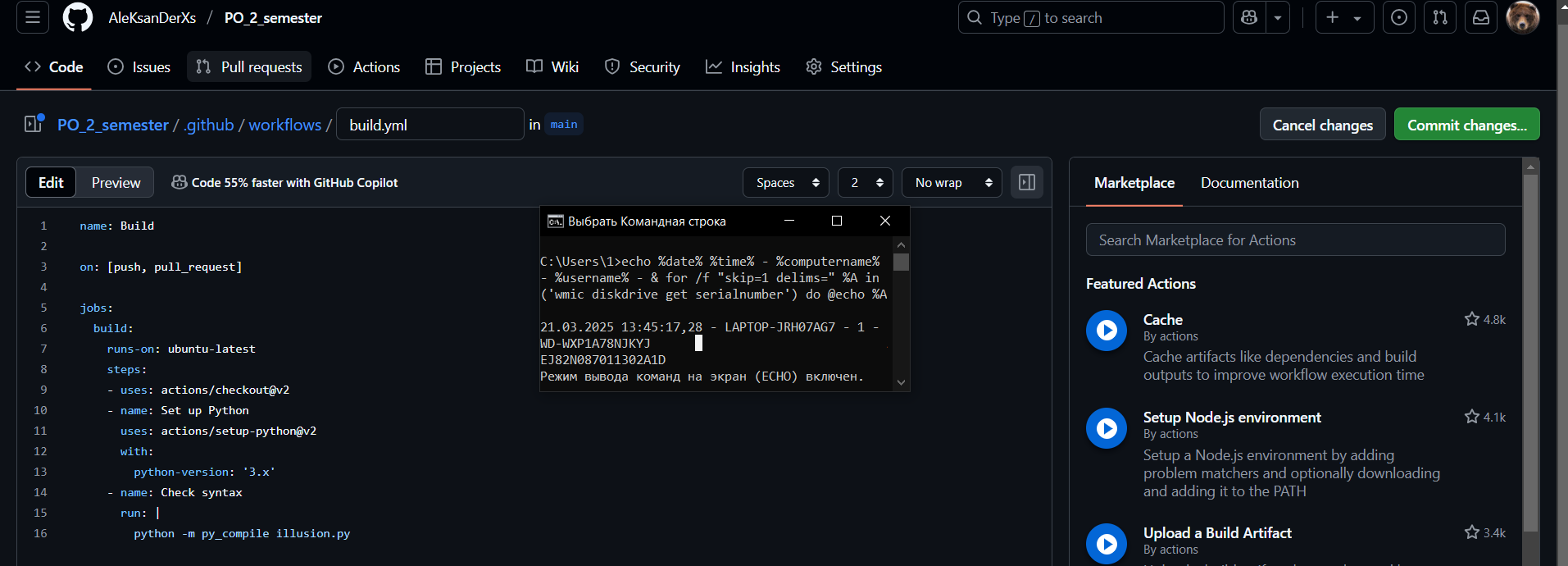
Коммитим новую ветку

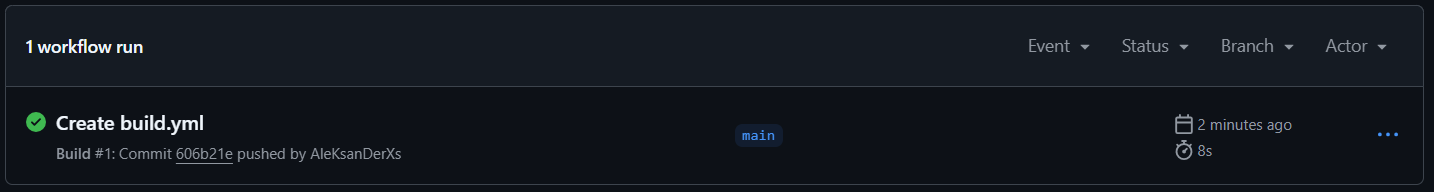


Пушим!



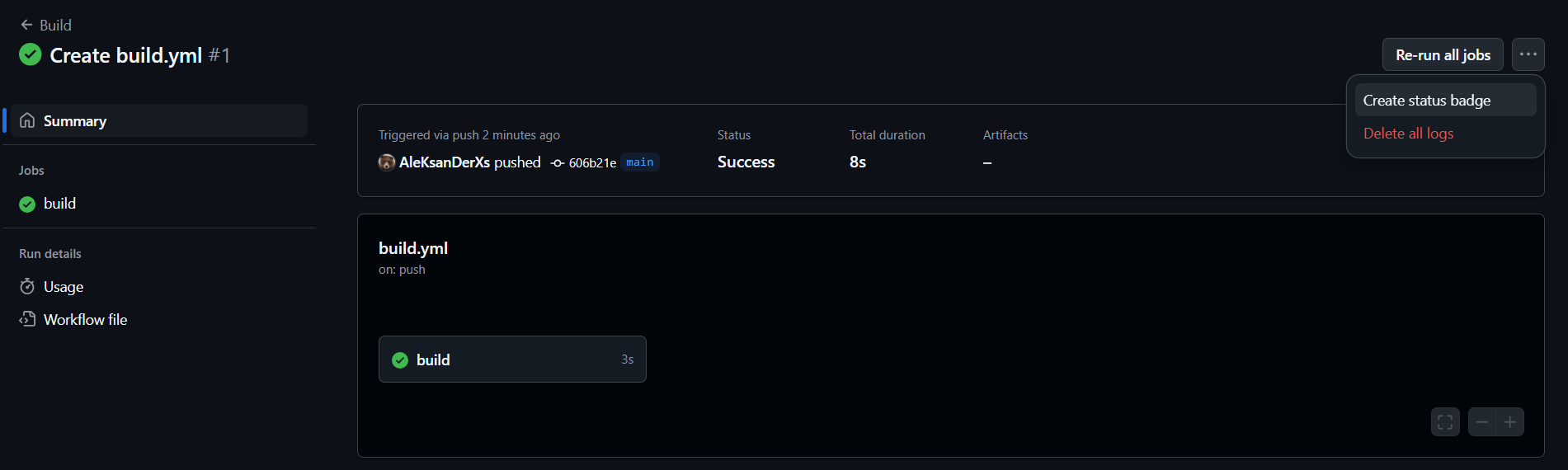
12. Настроим GitHub Actions для автоматической сборки

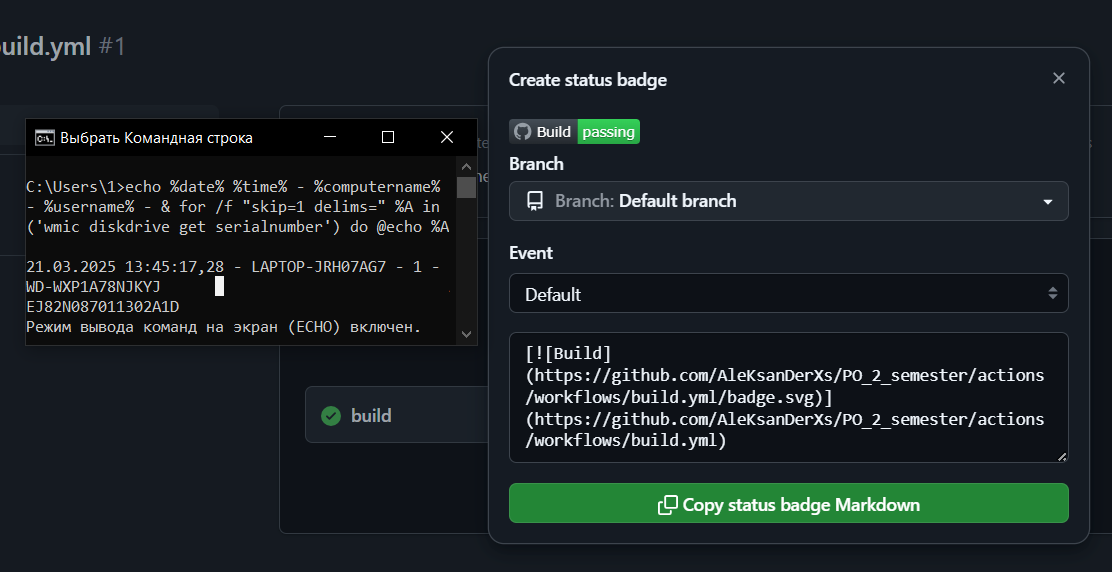




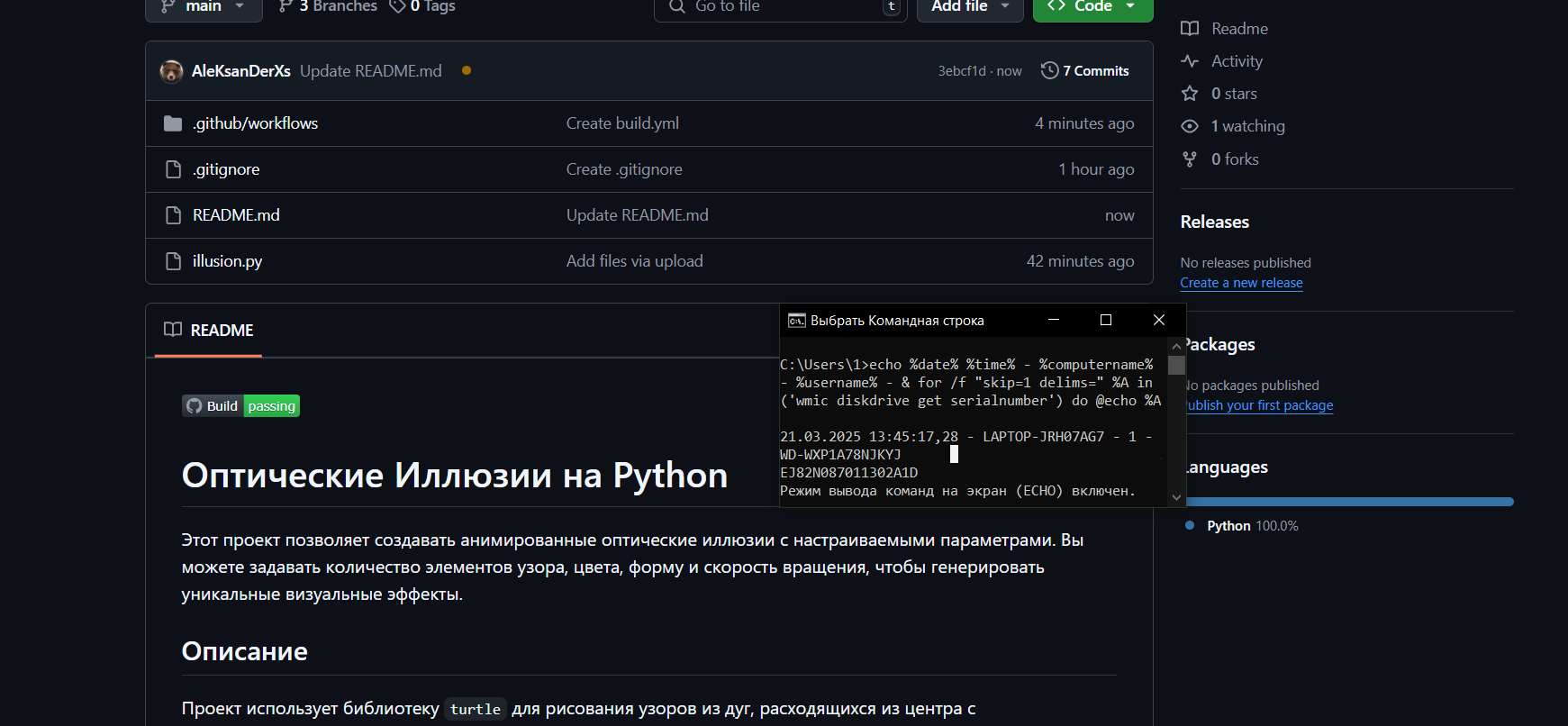
13. Добавить бейдж статуса сборки в README.md

Находим наш workflow и в нём находим кнопочку с «Create status badge».

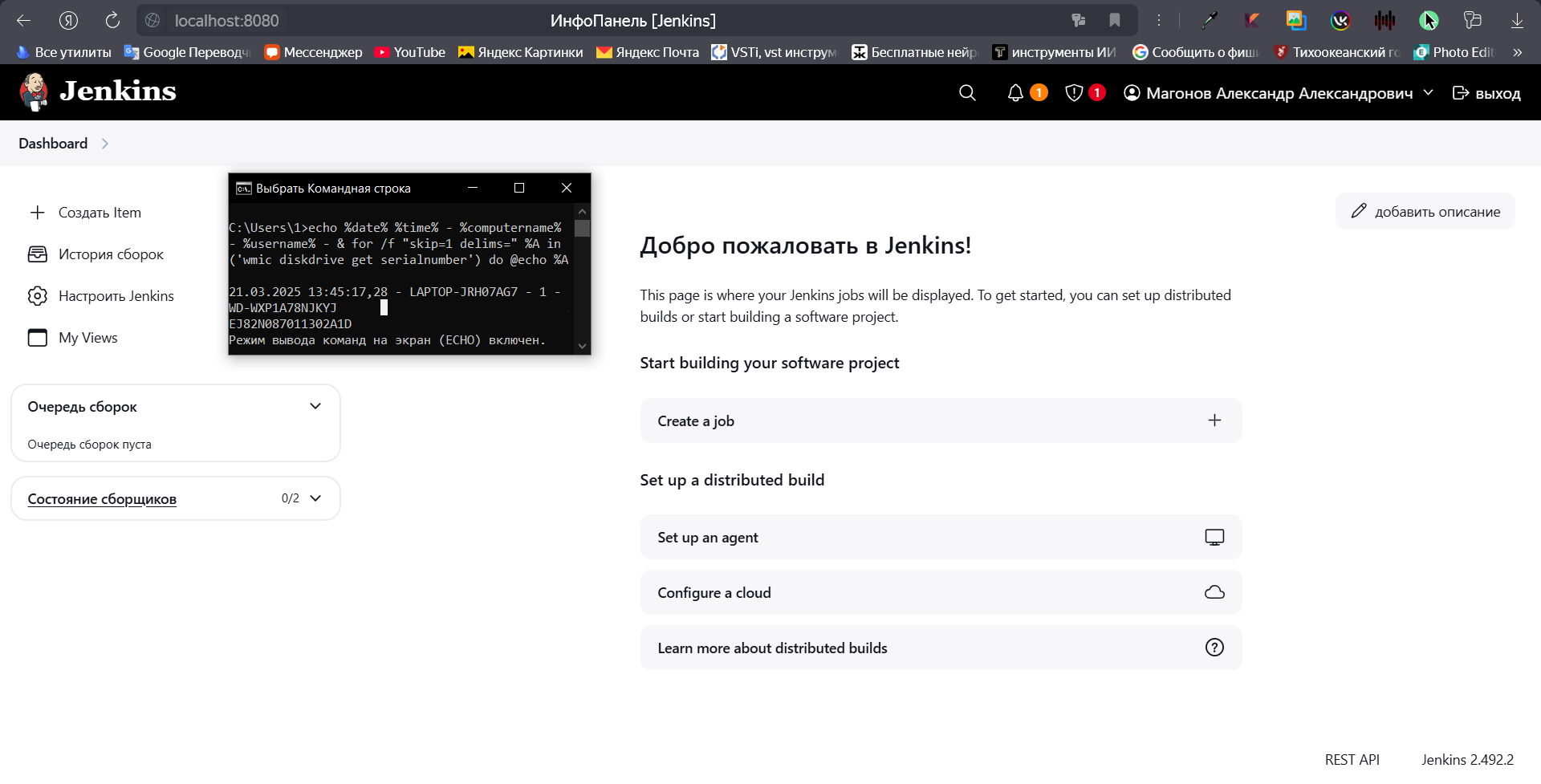




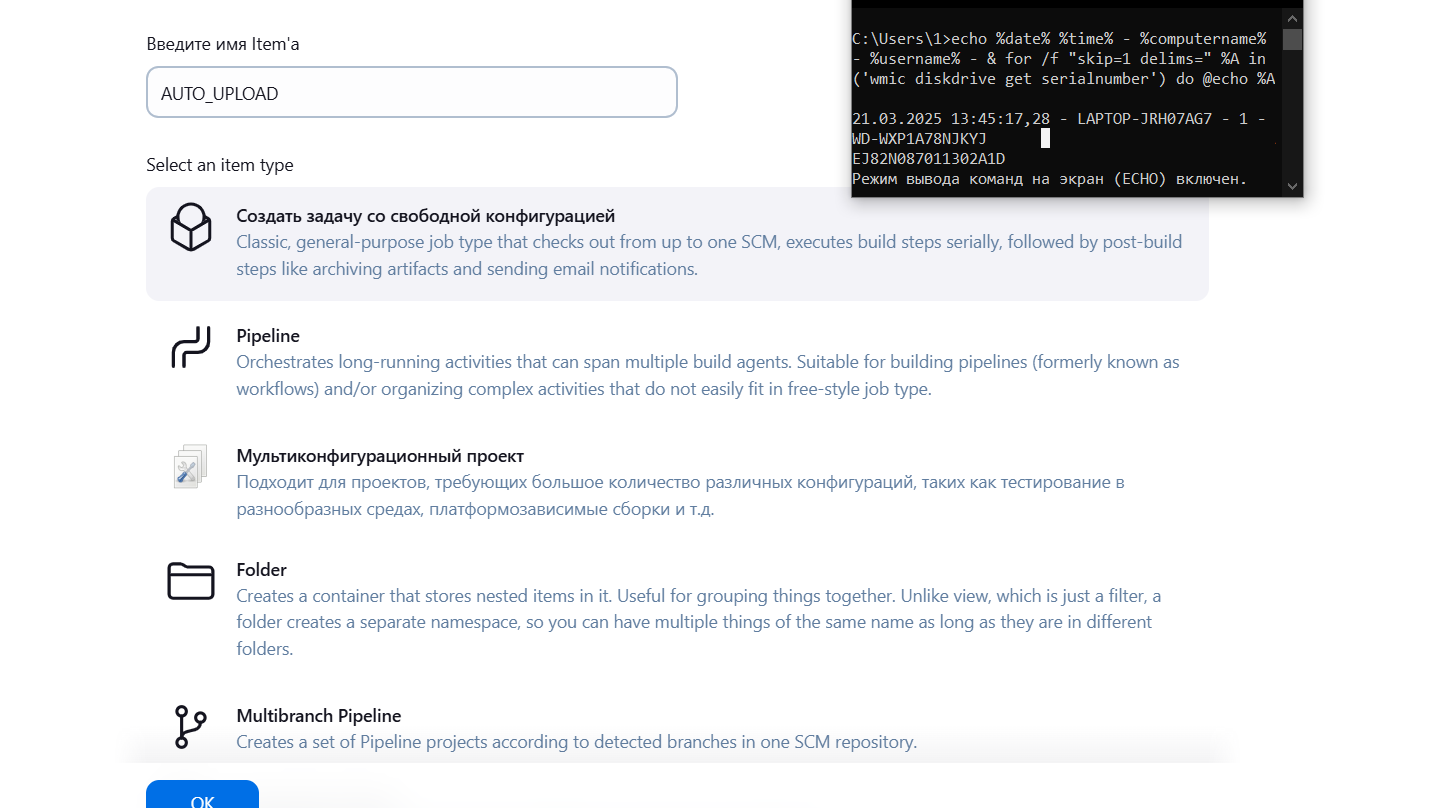
Копируем markdown код и вставляем в свой README.md файл (например, я вставлю в начало):



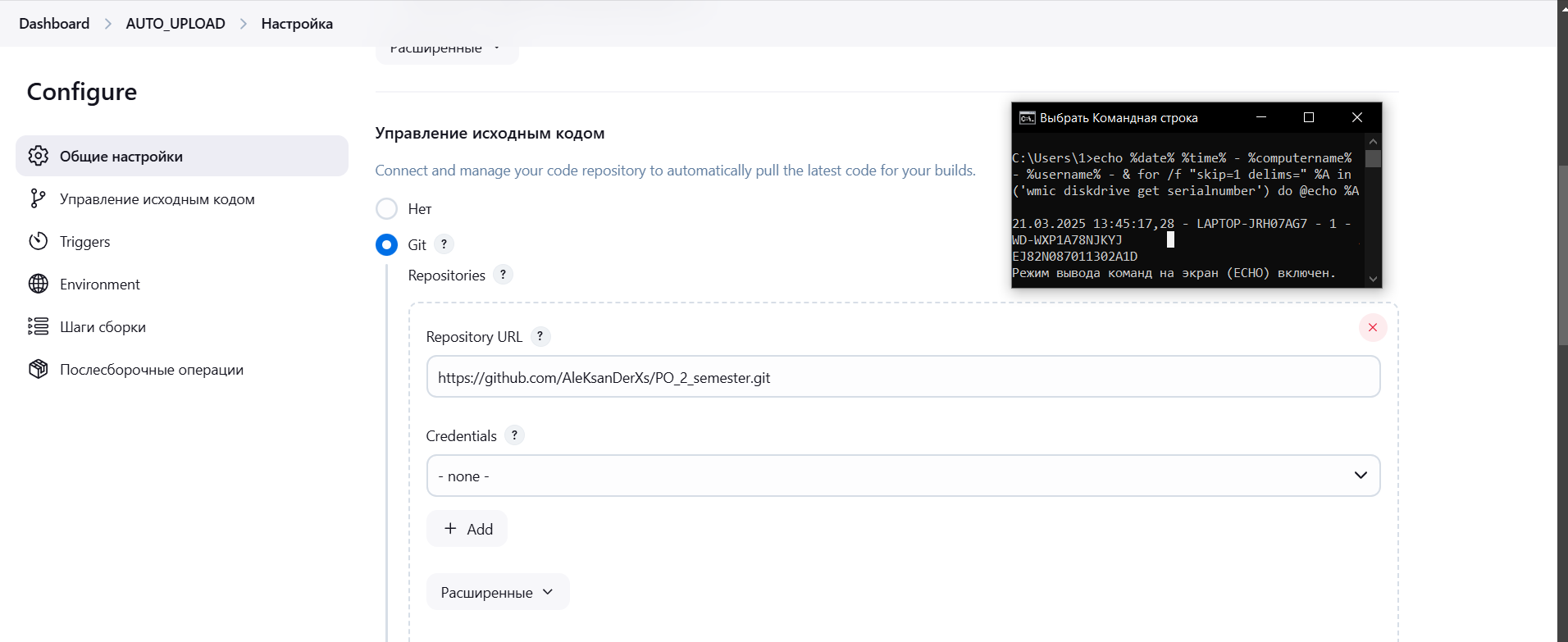
14. Устанавливаем и настраиваем Jenkins



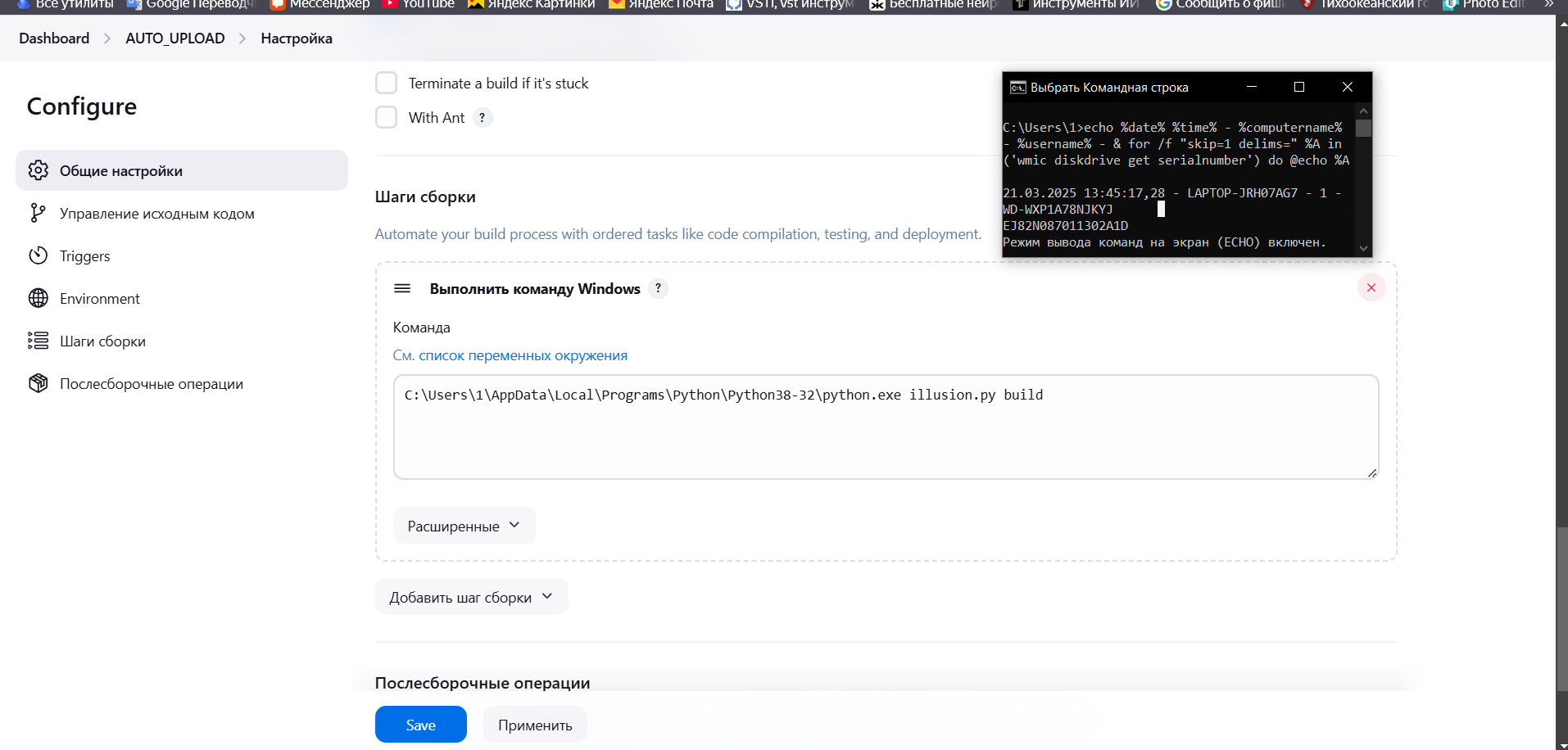
Создаём новый Job со свободной конфигурацией:



В общих настройках добавим мой репозиторий:

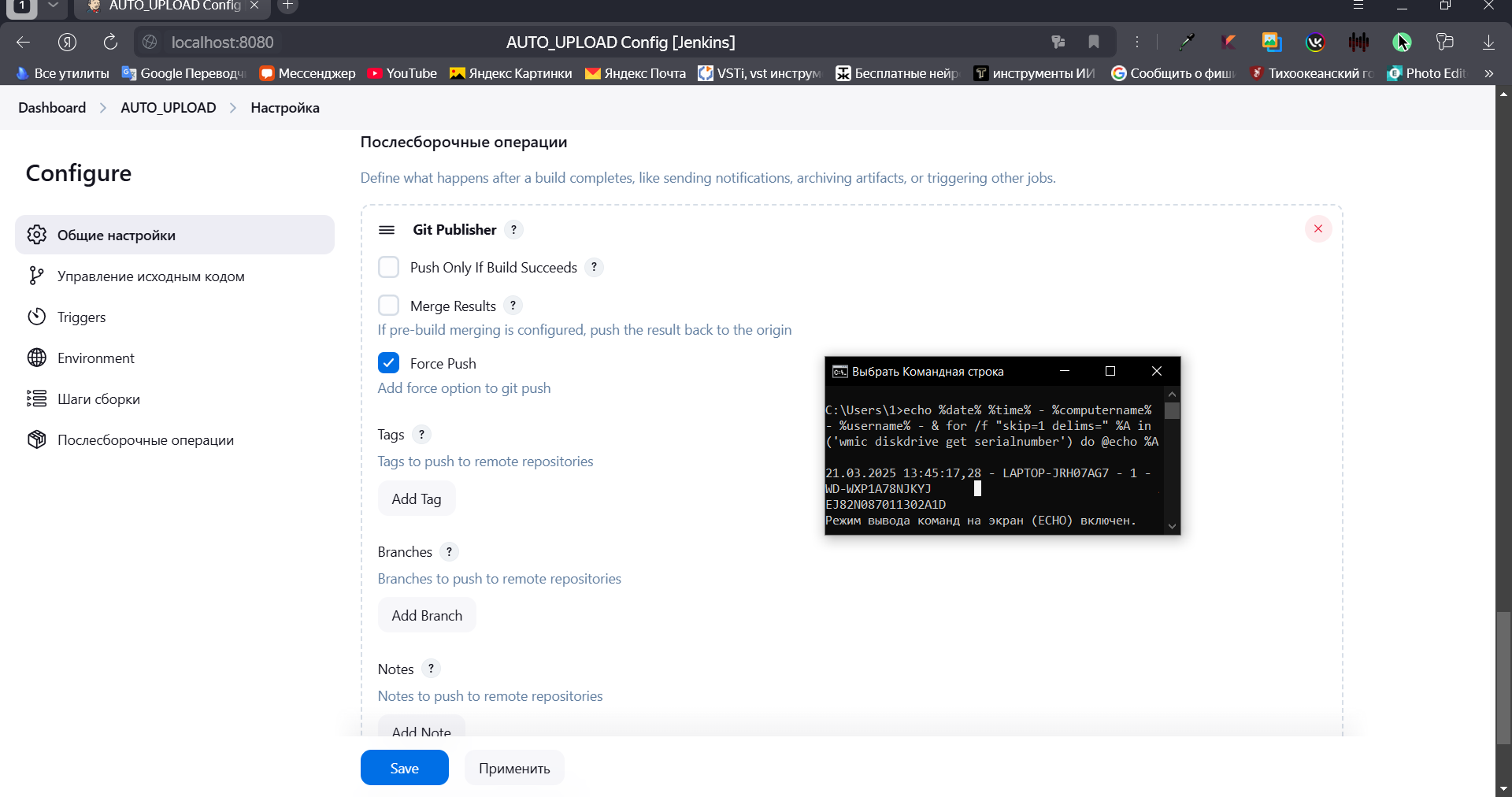


Настроим команду выполнения python кода



15. Автоматический деплой

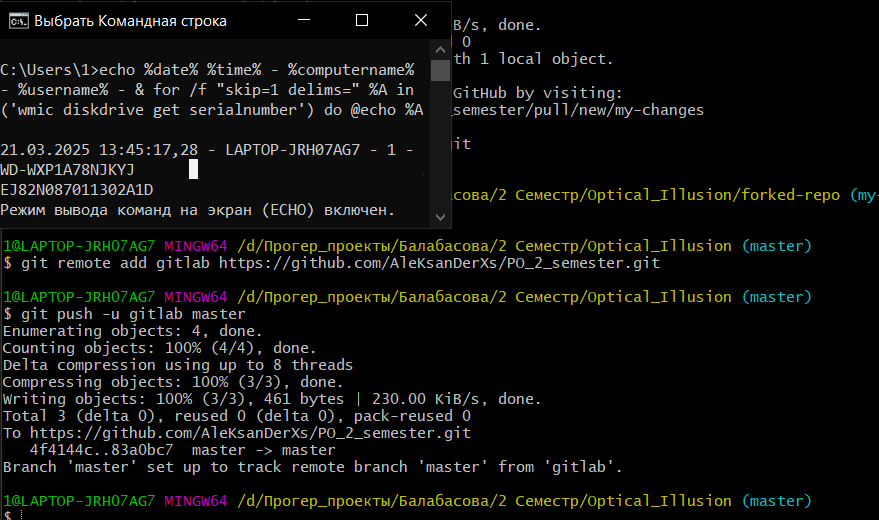
Добавляем послесборочную операцию загрузки проекта после сборки в нашу ветку на GitHub.

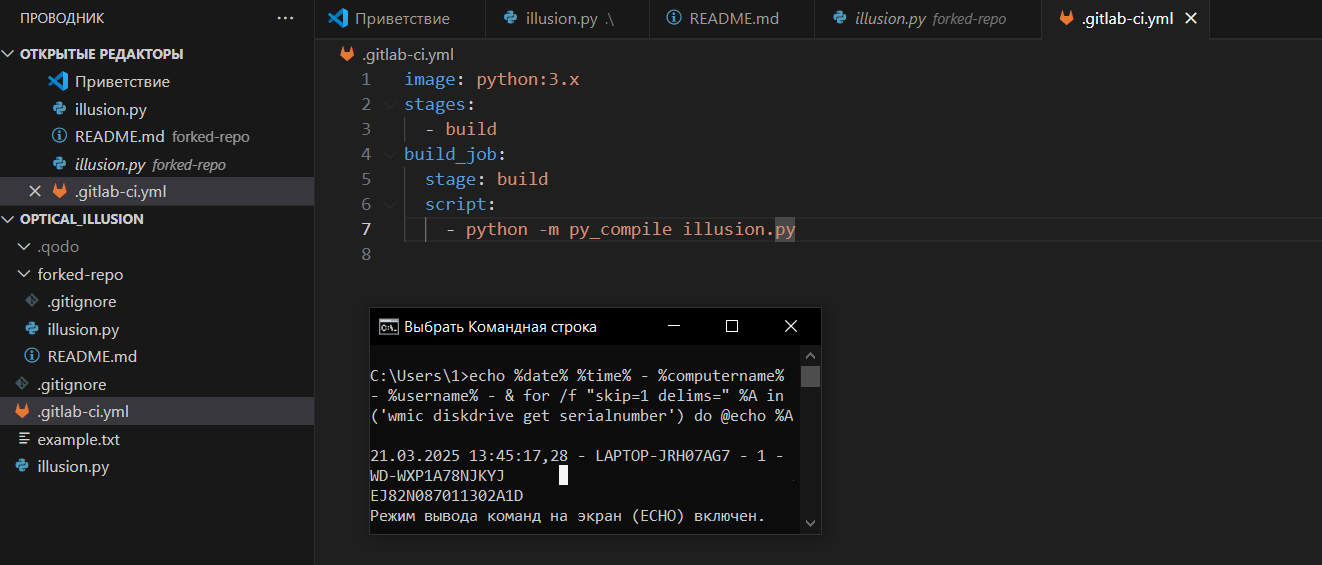


16. Настроить альтернативный CI/CD инструмент

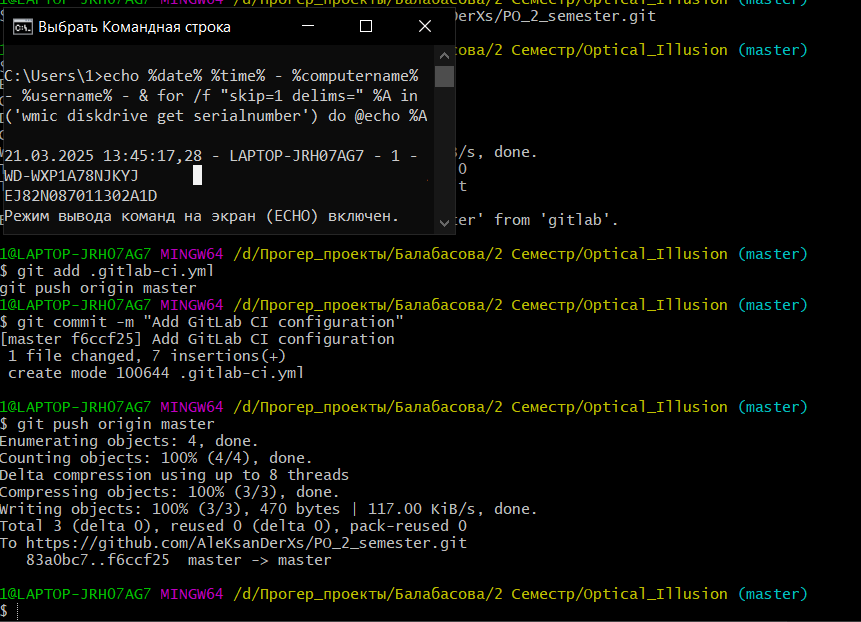
Будем использовать GitLab CI.

Для начала переносим репозиторий в гитлаб репозиторий.

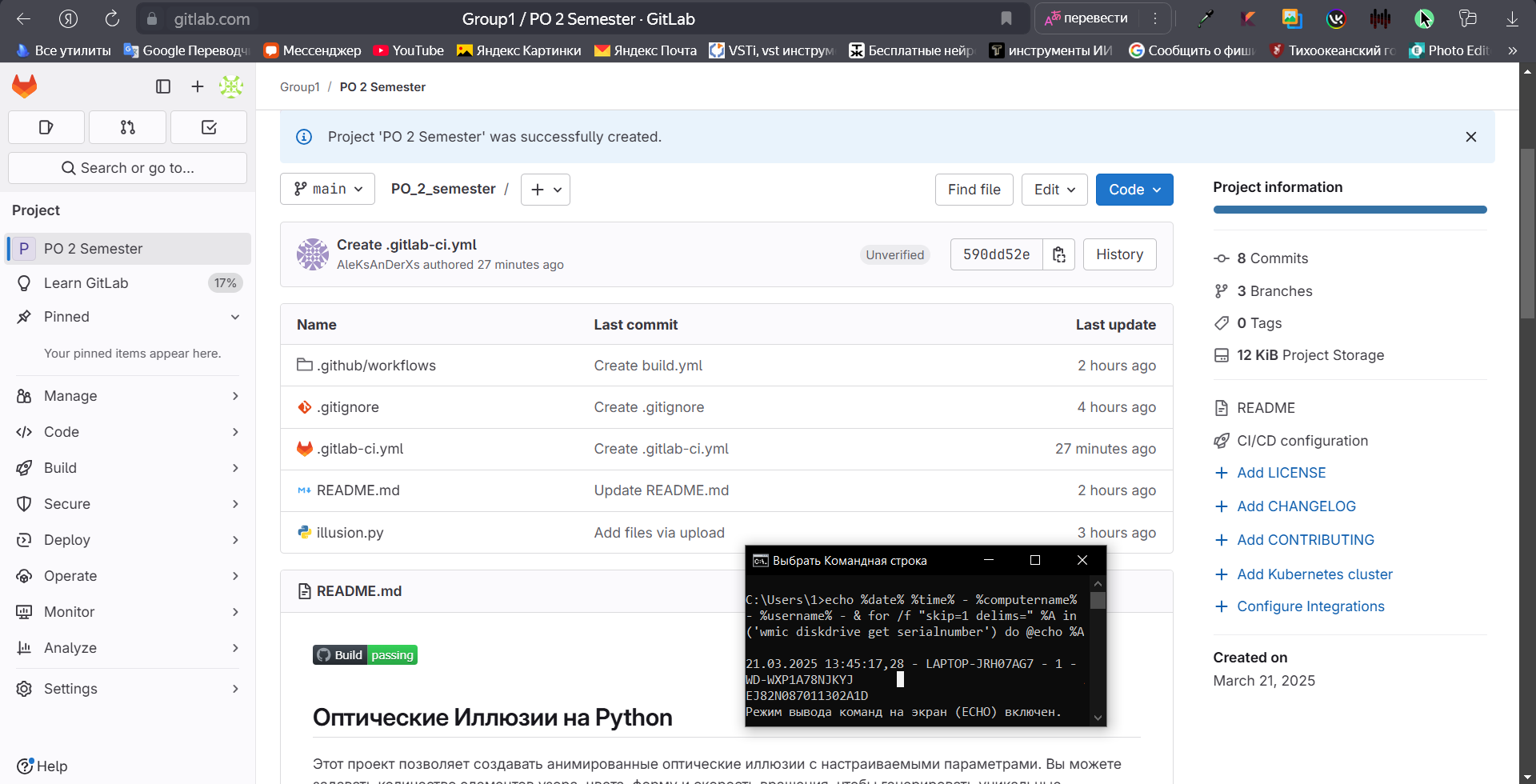


Напишем конфигурационный файл для GitLab: 

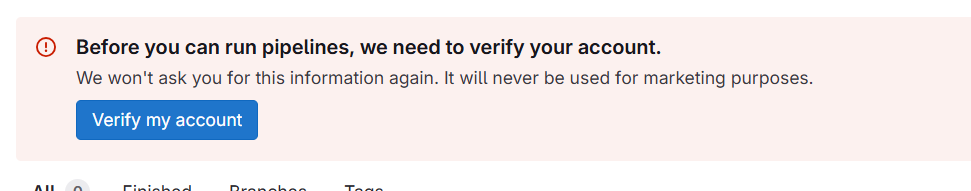
И пушим в ветку:



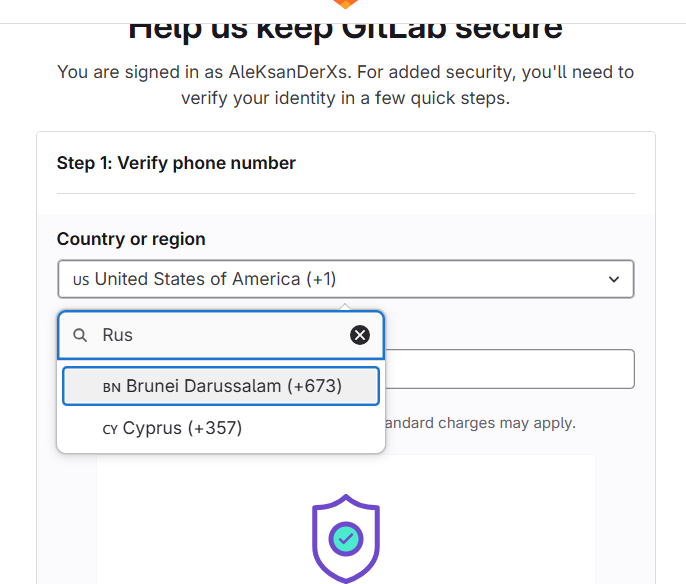
Предварительно регистрируемся в GitLab через GitHub. И проверяем.



По задумке, должна происходить автоматическая сборка, но я столкнулся с одной проблемой:

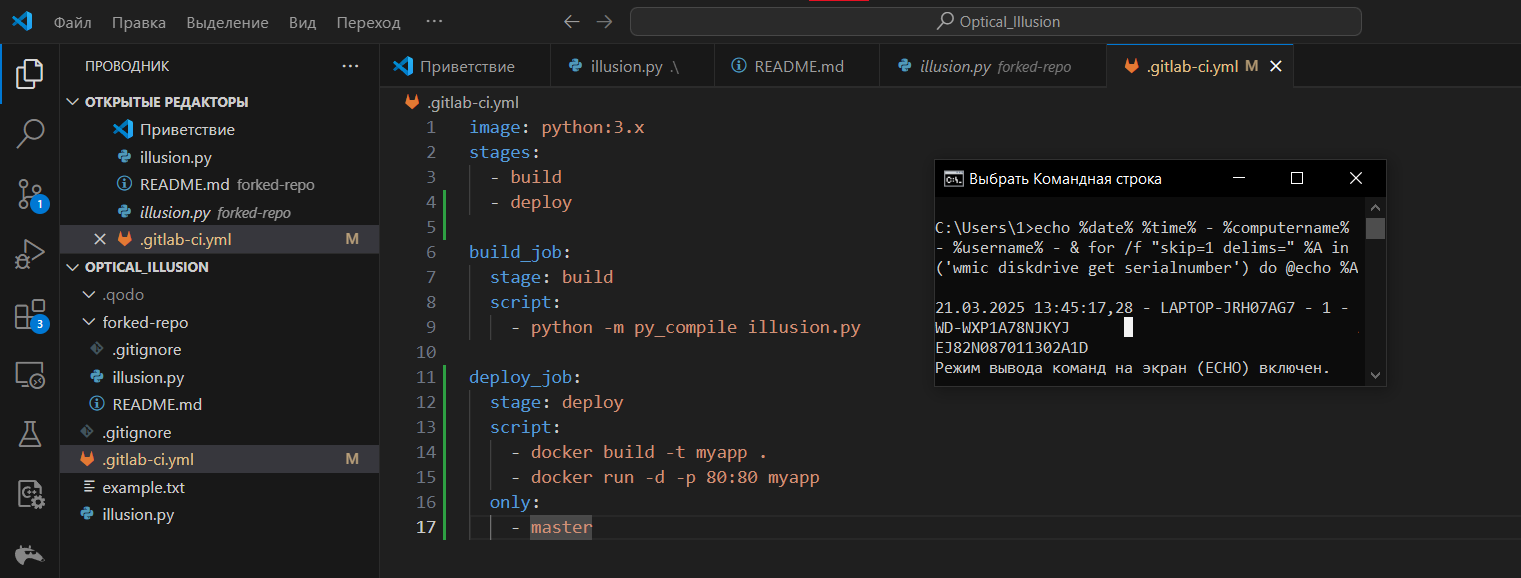


Похоже, что русским этой возможности не видать…



17. Автоматический Деплой через Gitlab CI.

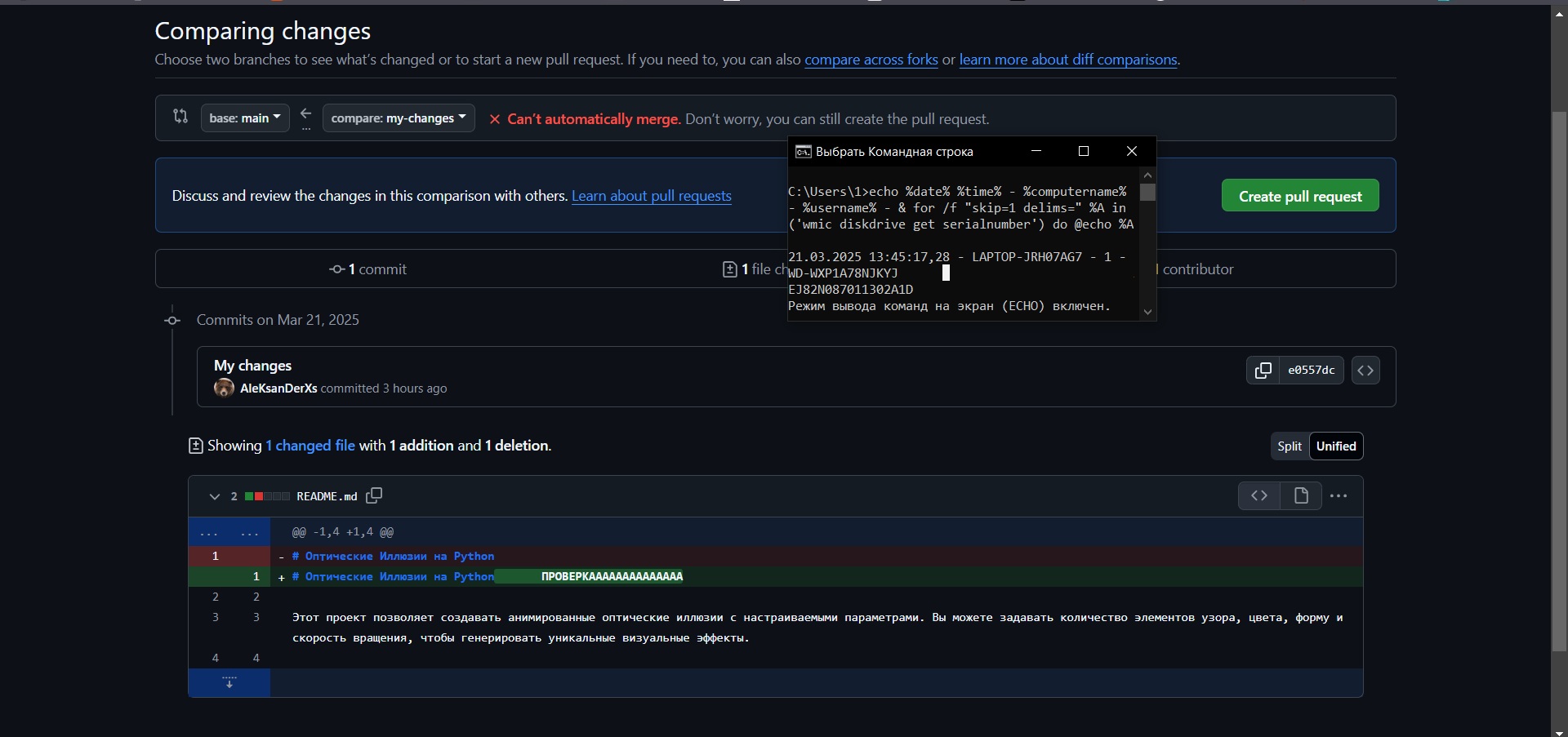
Для этого нам достаточно обновить существующий файл конфигурации для GitLab, добавив скрипт для деплоя:



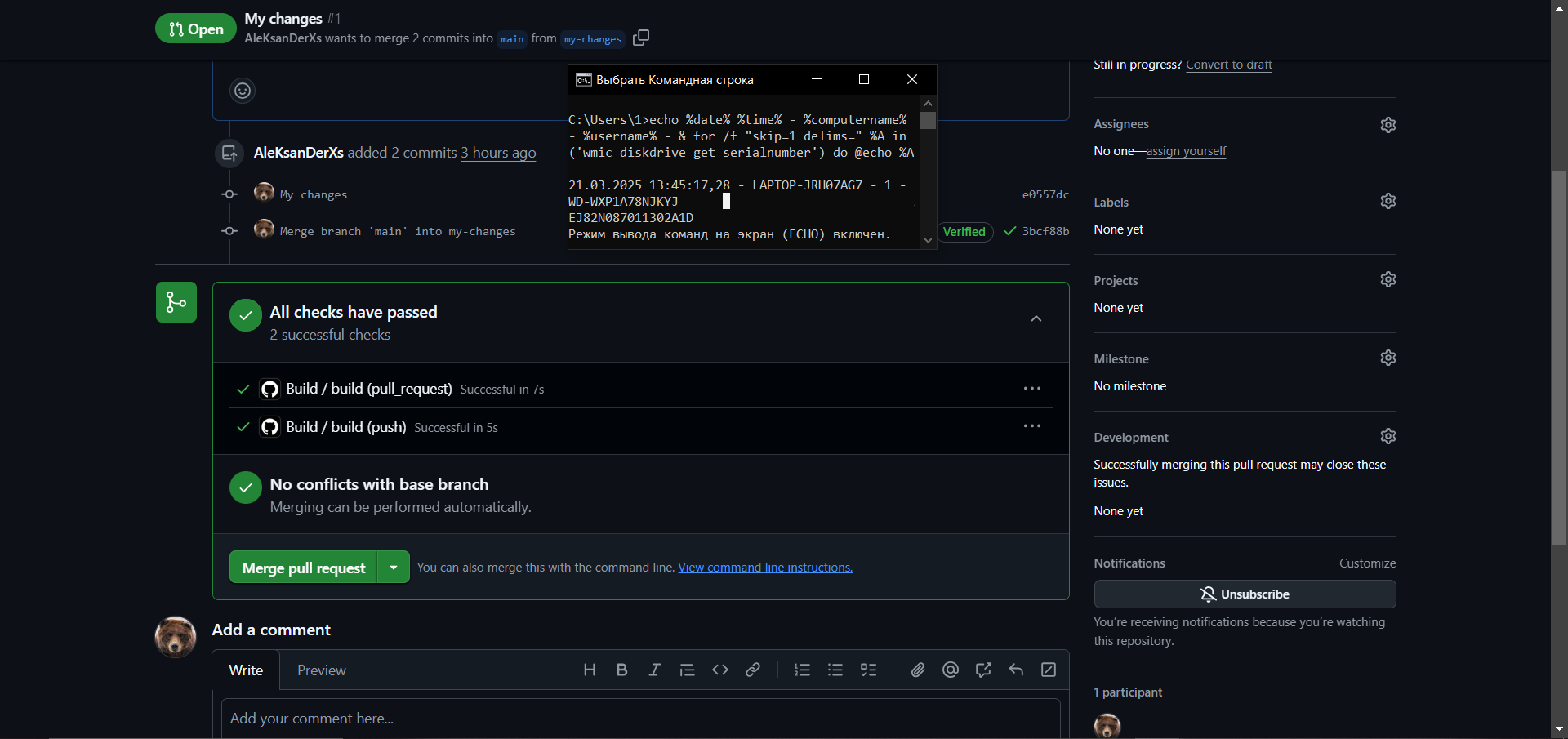
Теперь он будет выгружать проект после сборки мне в Docker.

18. Создать Pull Request

Создаём Pull Request между ветками main и my-changes (созданными раннее)

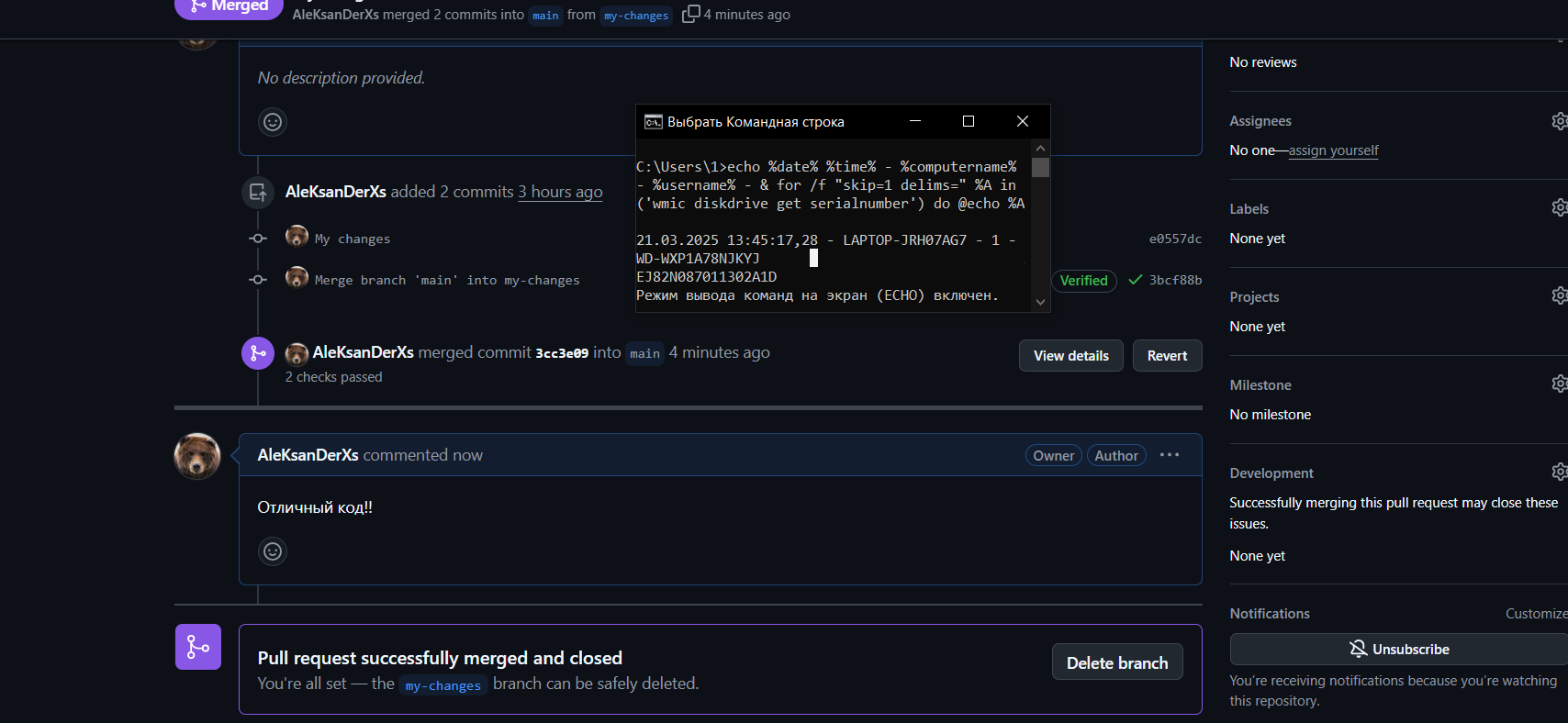


Решаем возникший конфликтик и подтверждаем:



19. Проводим code review

Просматриваем все изменения и правильность синтаксиса, сборки проекта, наличие ошибок. Пишем review (комментарий) о том, что всё прекрасно!



Вывод: лабораторная работа по освоению Git и GitHub позволила глубже понять основы работы с системами контроля версий и процессами непрерывной интеграции/развертывания (CI/CD). В ходе выполнения заданий были успешно созданы репозитории, настроены ветки и форки, а также реализованы процессы автоматической сборки и деплоя с использованием инструментов GitHub Actions, Jenkins и GitLab CI. Особое внимание было уделено практическим аспектам, таким как настройка .gitignore, управление конфликтами при слиянии веток и проведение code review через Pull Requests, что способствовало развитию навыков командной работы и анализа кода.

Полученные знания и опыт имеют высокую практическую ценность, так как они применимы в реальных проектах разработки программного обеспечения. Автоматизация процессов сборки и деплоя позволила сократить время выполнения рутинных задач и повысить надежность проектов, а освоение инструментов code review улучшило способность выявлять ошибки и предлагать оптимизации. Эта работа стала важным шагом в освоении современных методологий разработки, подготавливая к эффективной работе в профессиональной среде, где требуется интеграция и совместное управление кодом.