ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ  
"ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ"

Кафедра программной инженерии

им. Л.П. Фельдмана

# ОТЧЁТ

к лабораторной работе по дисциплине  
«Профессиональная практика программной инженерии»  
на тему: «Работа с ветками»

Вариант 12: 3D-редактор

Выполнил:

ст. гр. ПИ-19б  
Носаченко А. А.

Проверил:

асс. каф. ПИ  
Дмитрюк Т. Г.

Донецк 2023

Цель: познакомиться с основами использования веток в системе контроля версий Git.

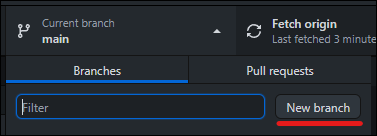
Задание:

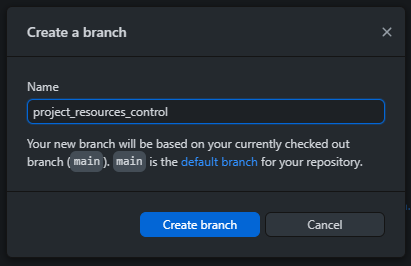
Разработать архитектуру в репозитории для разработки выбранного по заданию из 1 лабораторной работы проекта. Реализовать каждый из модулей в отдельных ветках, сделать не менее пяти фиксаций касательно разработки программного кода и после влить в ветку master. Зафиксировать лог ревизии для каждой из веток.

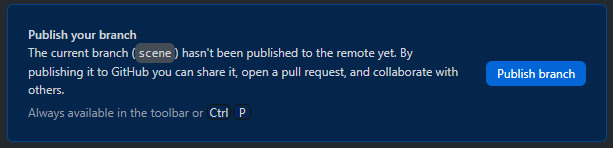
Удалить все созданные ветки и предоставить вывод команды git log для trunk после вливания всех веток.

Ход работы

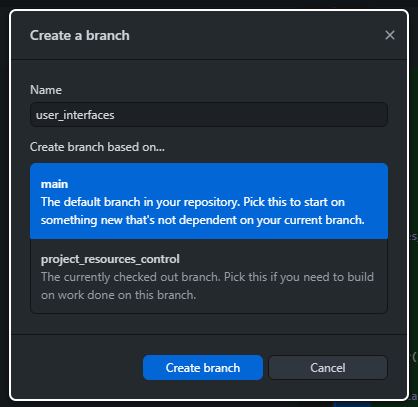
1) Создание новой ветки, когда нет других побочных веток:



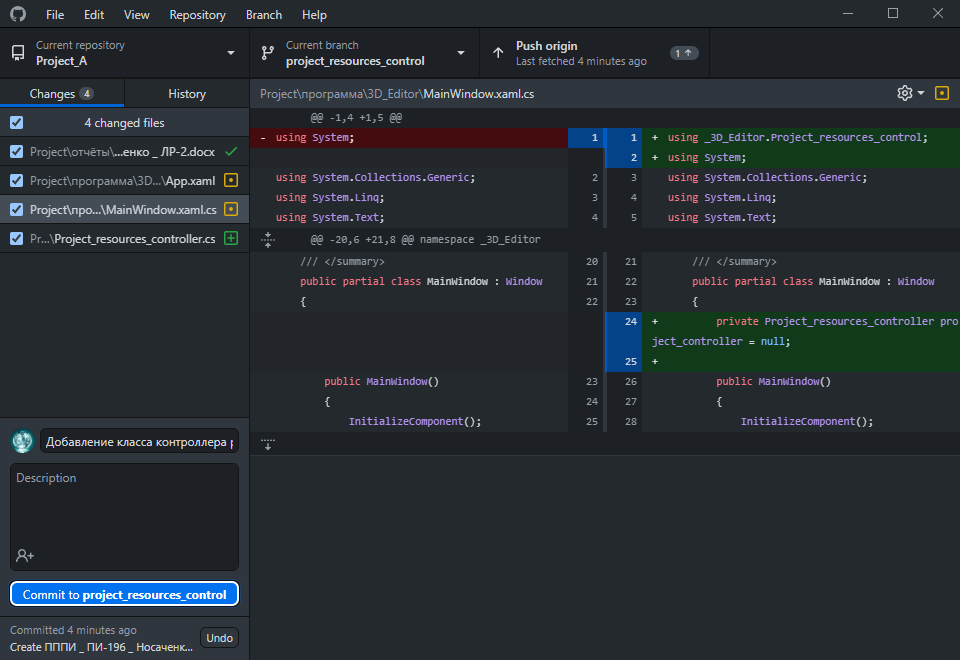




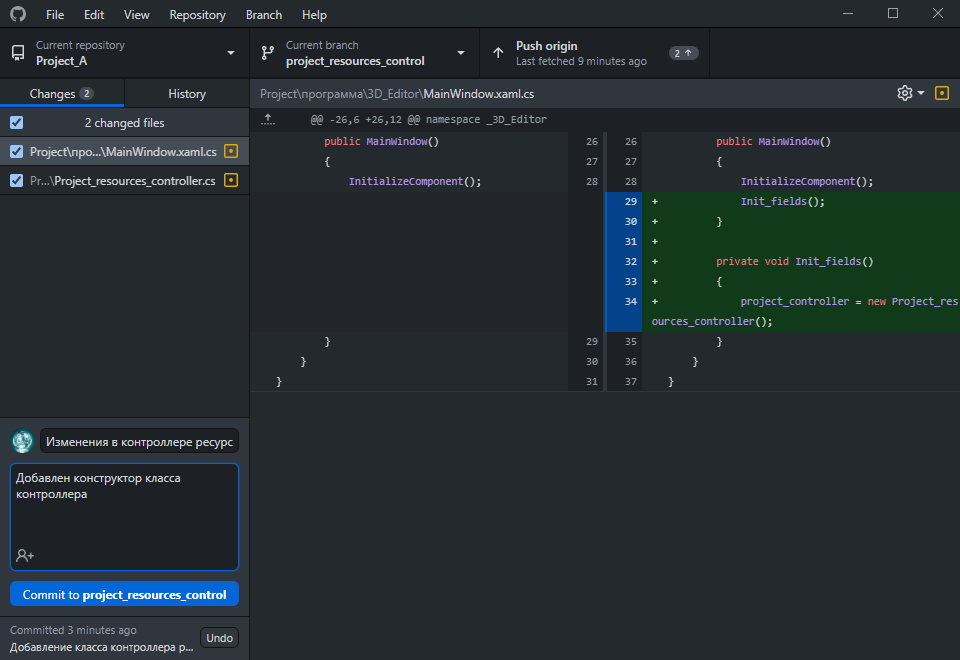
2) Создание ветки с выбором базовой ветки для создаваемой из имеющихся (базовая ветка определяет, какие коммиты будут сразу загружены в созданной ветке):



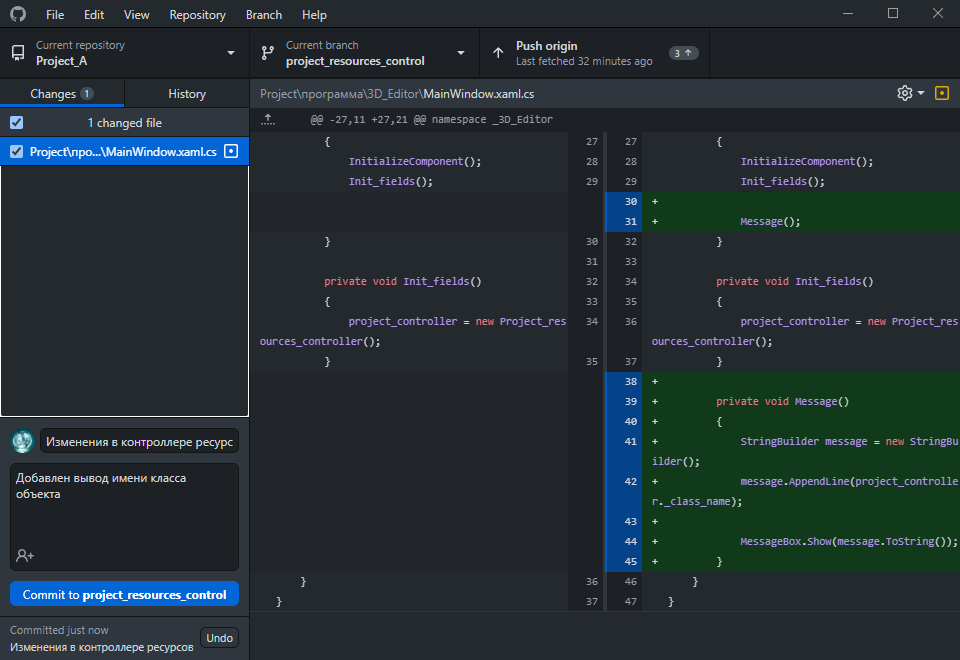
3) Первая фиксация изменений в ветке контроллера ресурсов:



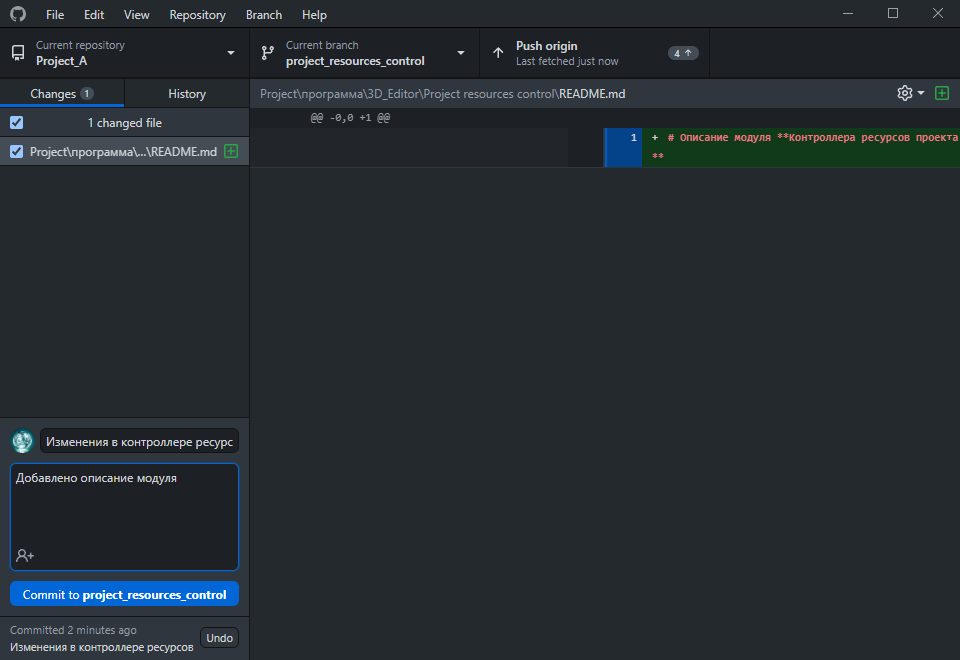
4) Вторая фиксация изменений в ветке контроллера ресурсов:



5) Третья фиксация изменений в ветке контроллера ресурсов:

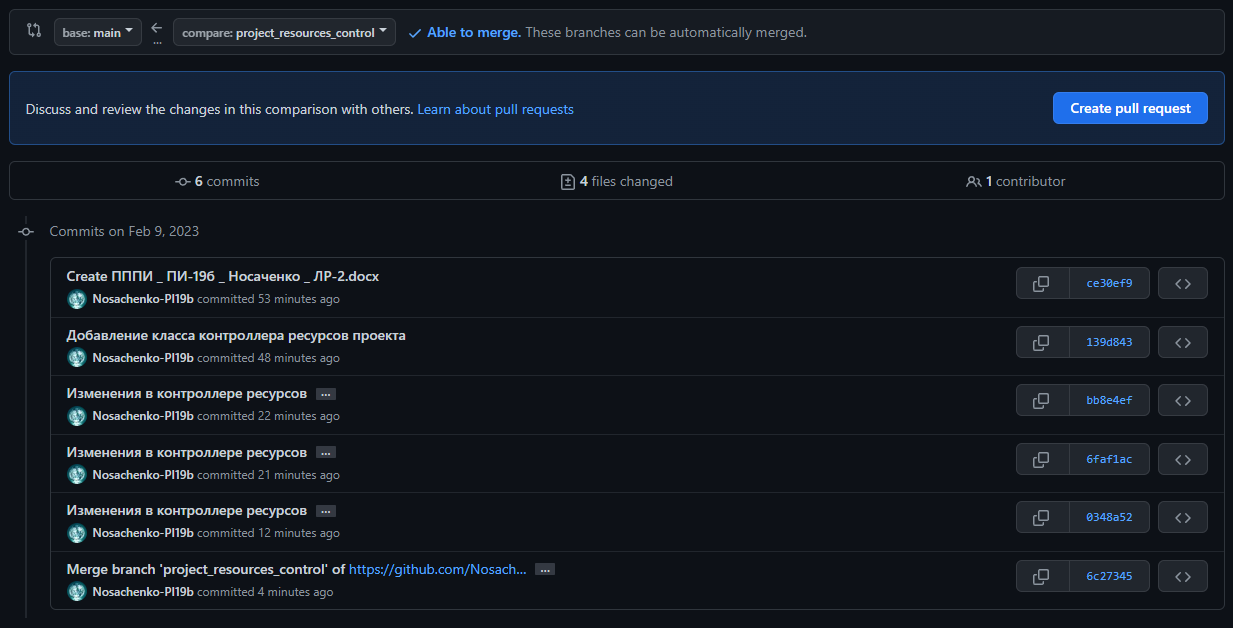


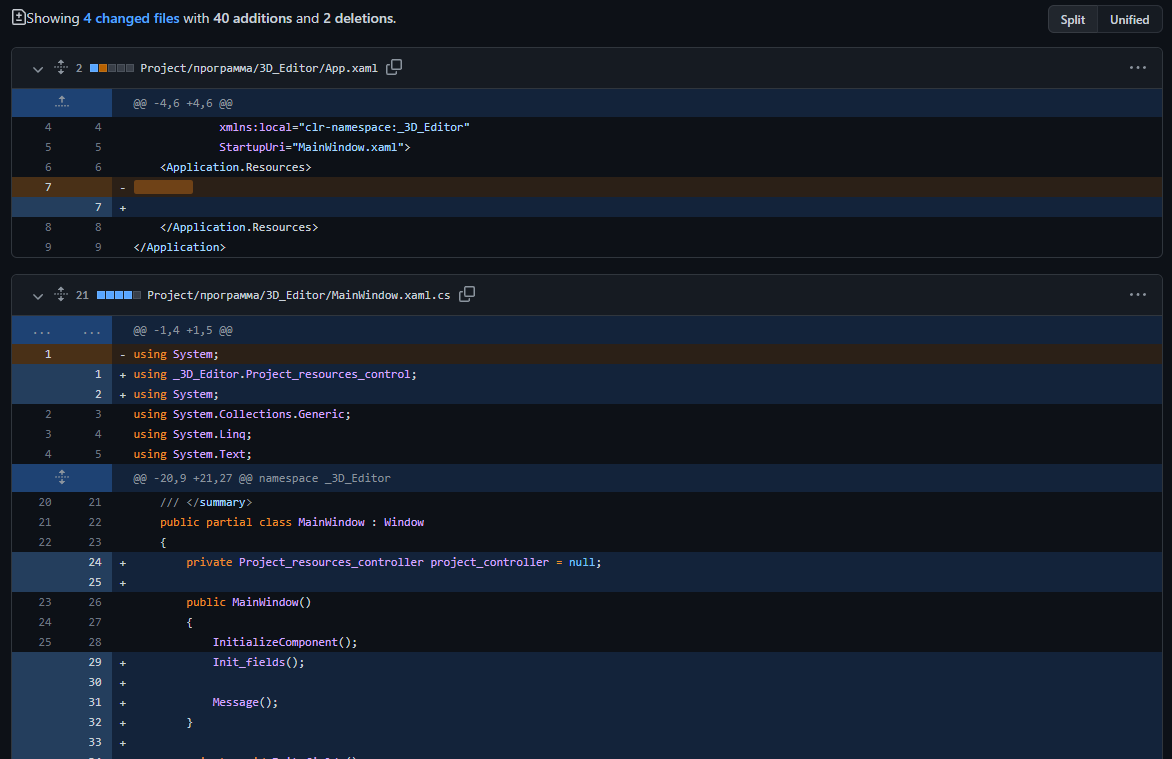
6) Четвёртая фиксация изменений в ветке контроллера ресурсов:

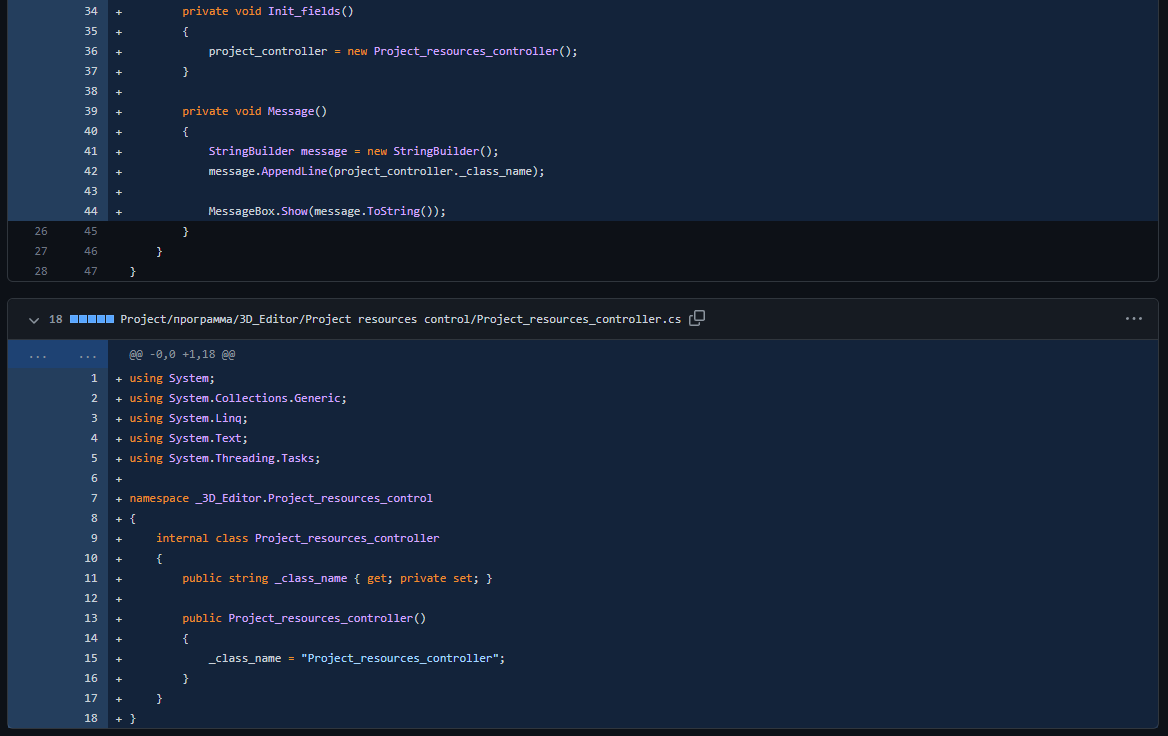


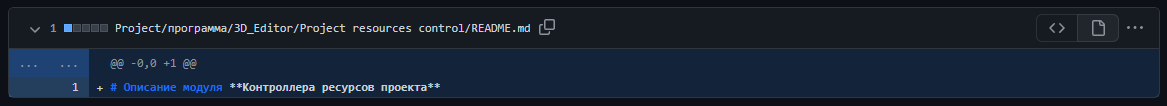
7) Список коммитов, доступных к вливанию в main-ветку после нажатия на

 и их сравнение с базовой веткой:





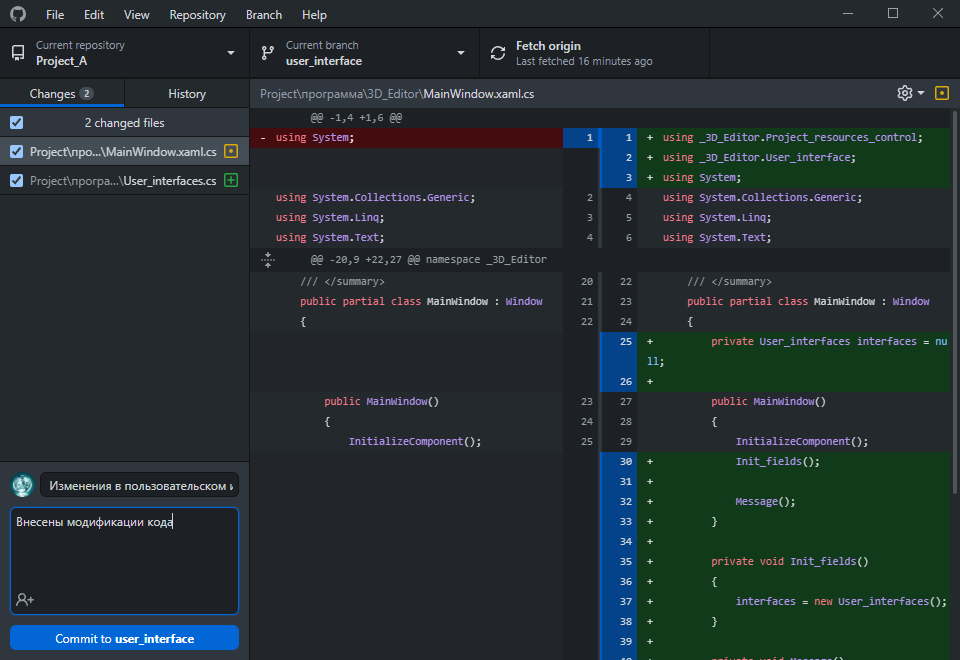




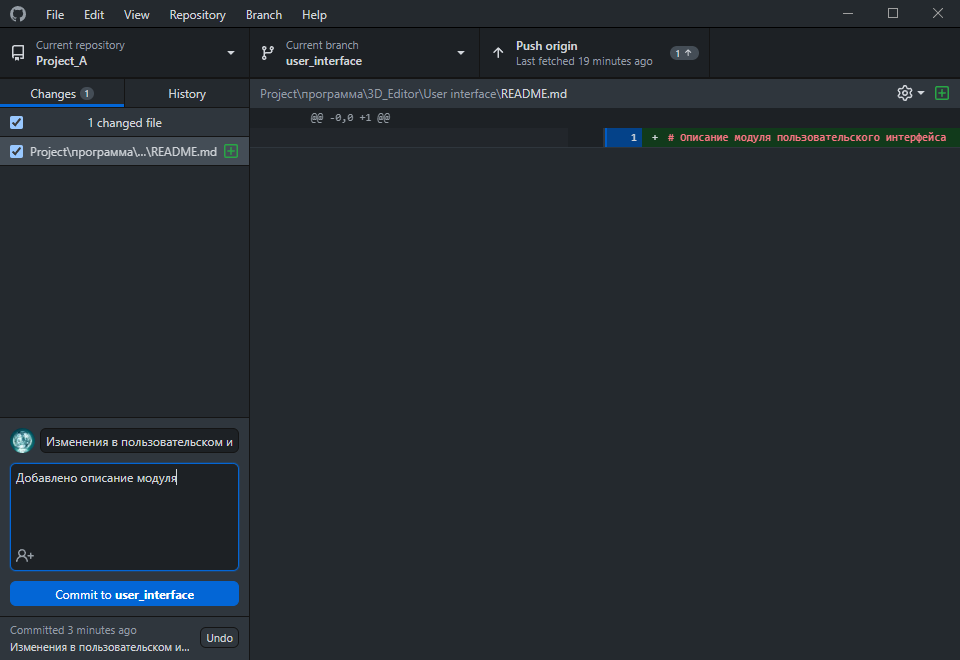
Консольная команда вывода списка всех коммитов:

git log --pretty=format:\"%h %ad | %s%d [%an]\" --graph --date=short

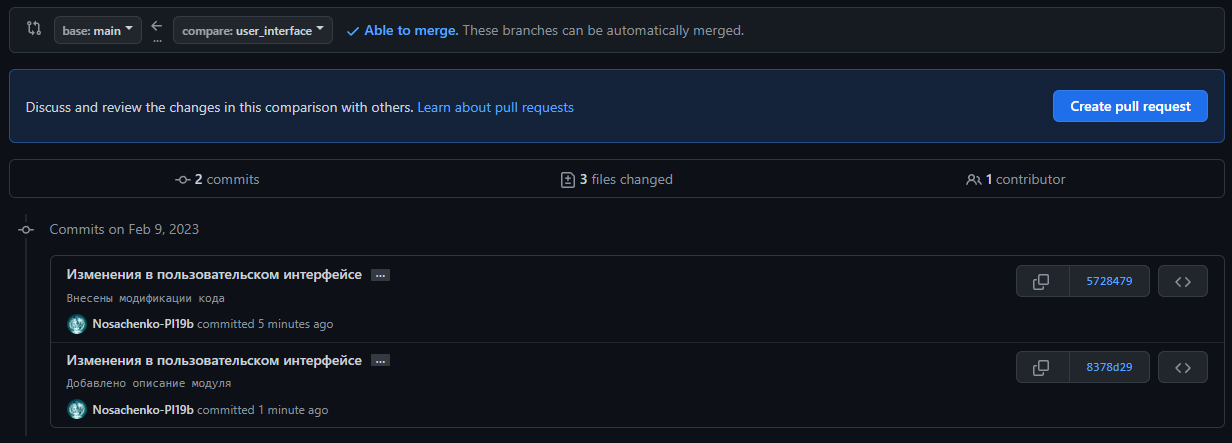
8) Первая фиксация изменений в ветке пользовательского интерфейса:

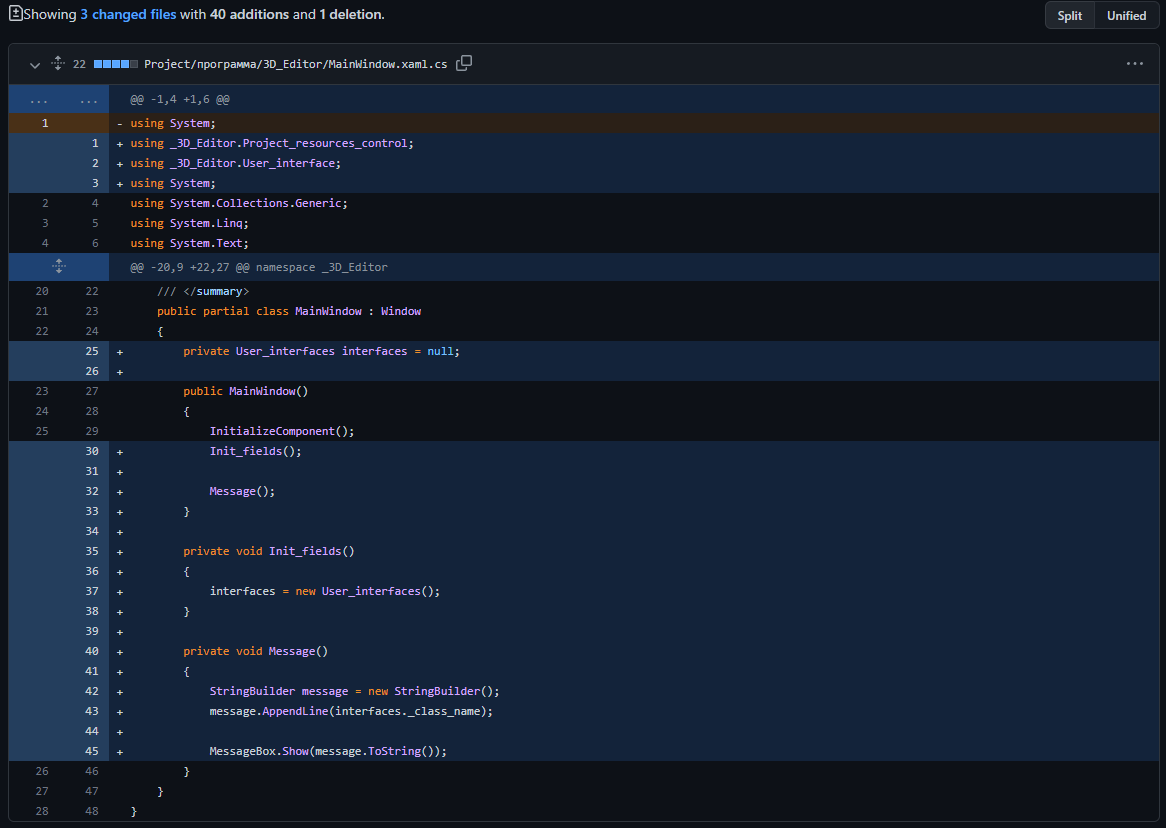


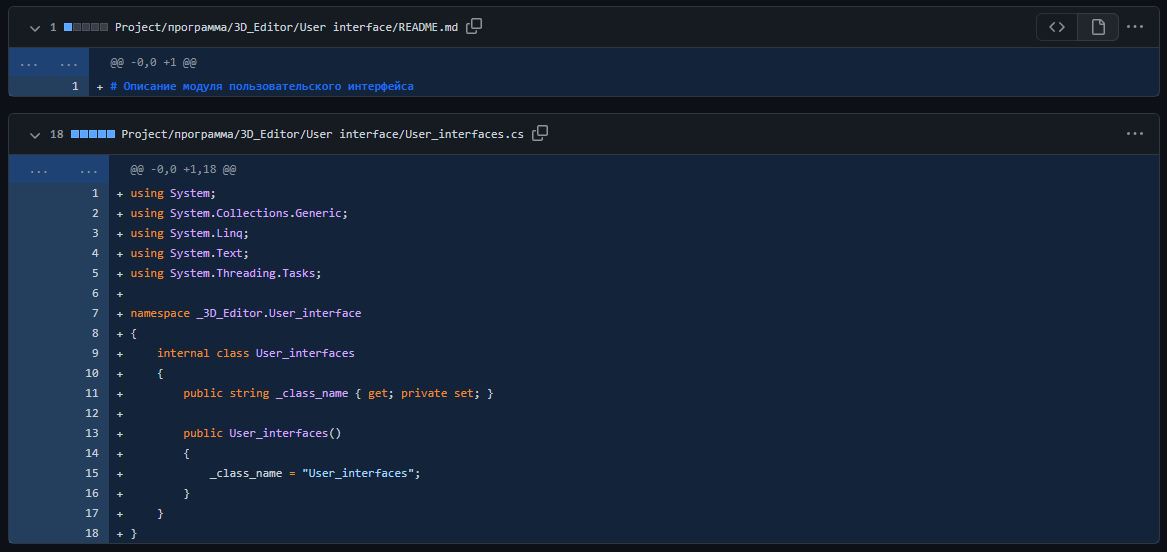
9) Вторая фиксация изменений в ветке пользовательского интерфейса:



10) Список коммитов, доступных к вливанию в main-ветку после Push’а и их сравнение с базовой веткой:







Консольная команда для перехода между ветками при их контроле при помощи консоли:

git checkout main

11) Повторение создания ветки, добавления программного класса для модуля, внесения изменений в код класса модуля и код главного класса «MainWindow» и добавления файла описания модуля для всех остальных модулей:

Имена для веток и классов модулей:

М. контроля сцены – **scene** : Scene\_controller

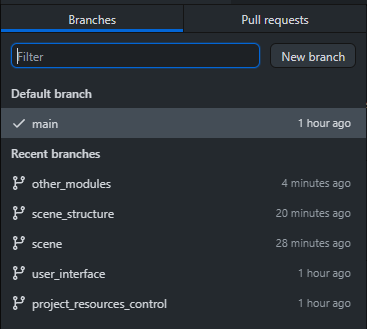
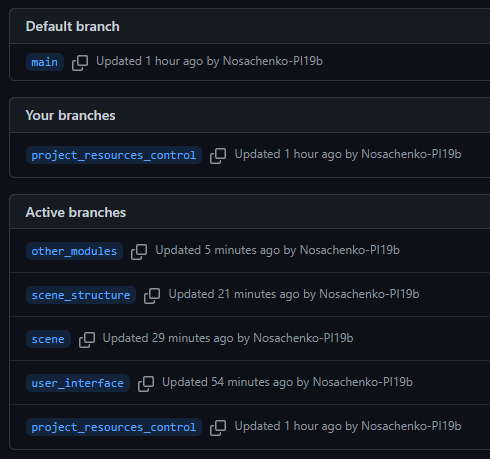
М. контроля структуры сцены – **scene\_structure** : Scene\_structure\_controller

М. создания и модификации примитивов – **other\_modules** : Primitives

М. "лепки" объектов – **other\_modules** : Modelling

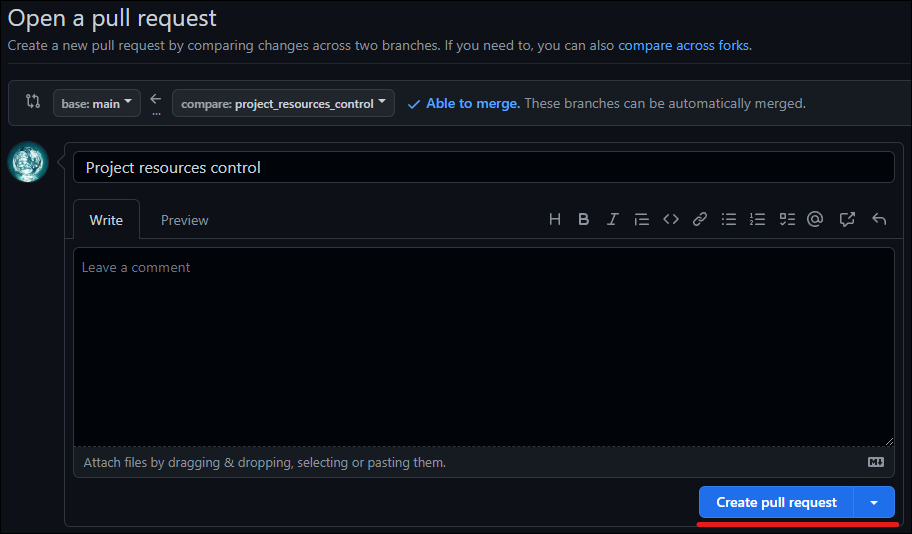
М. создания и поддержки текстур и материалов – **other\_modules** : Painting

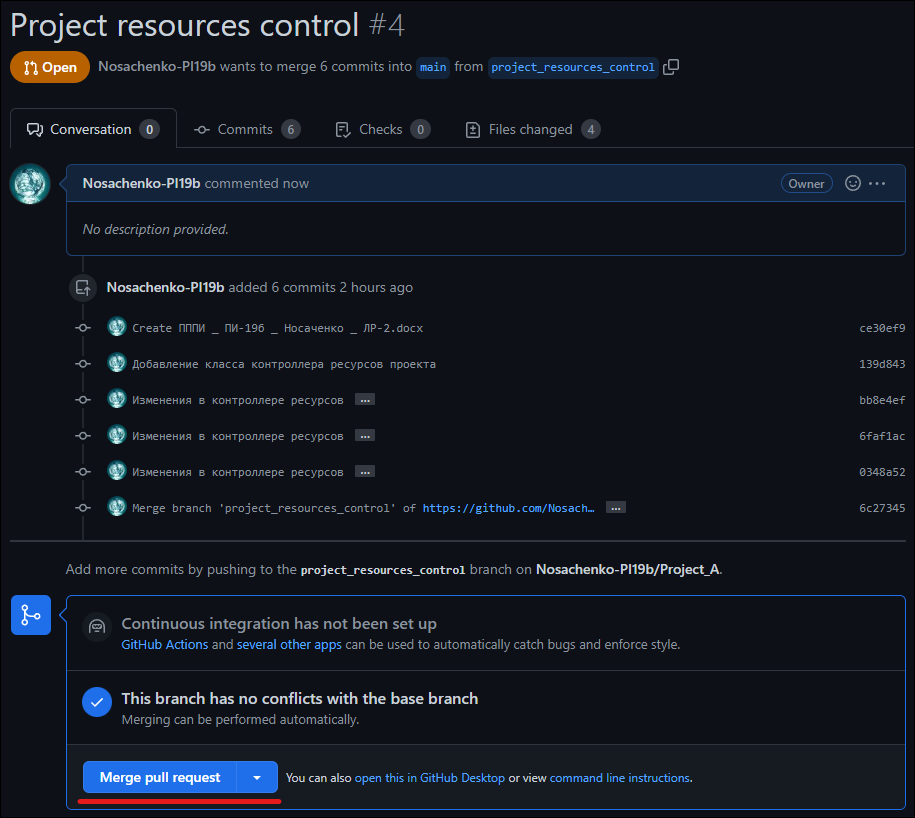
М. анимации объектов – **other\_modules** : Animation

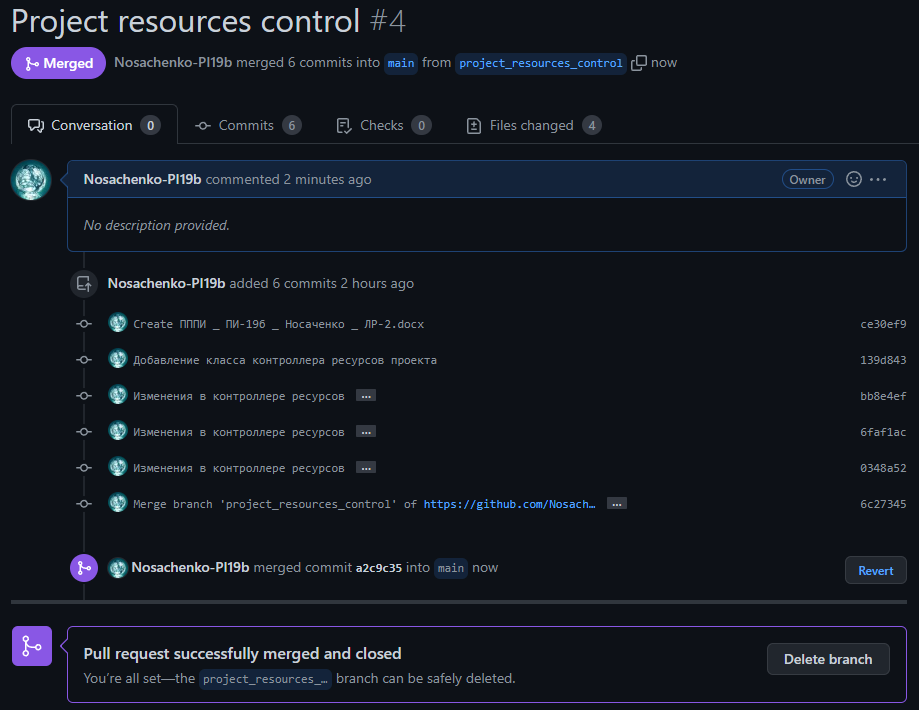
12) Запрос слияния main-ветки и ветки контроля ресурсов проекта:

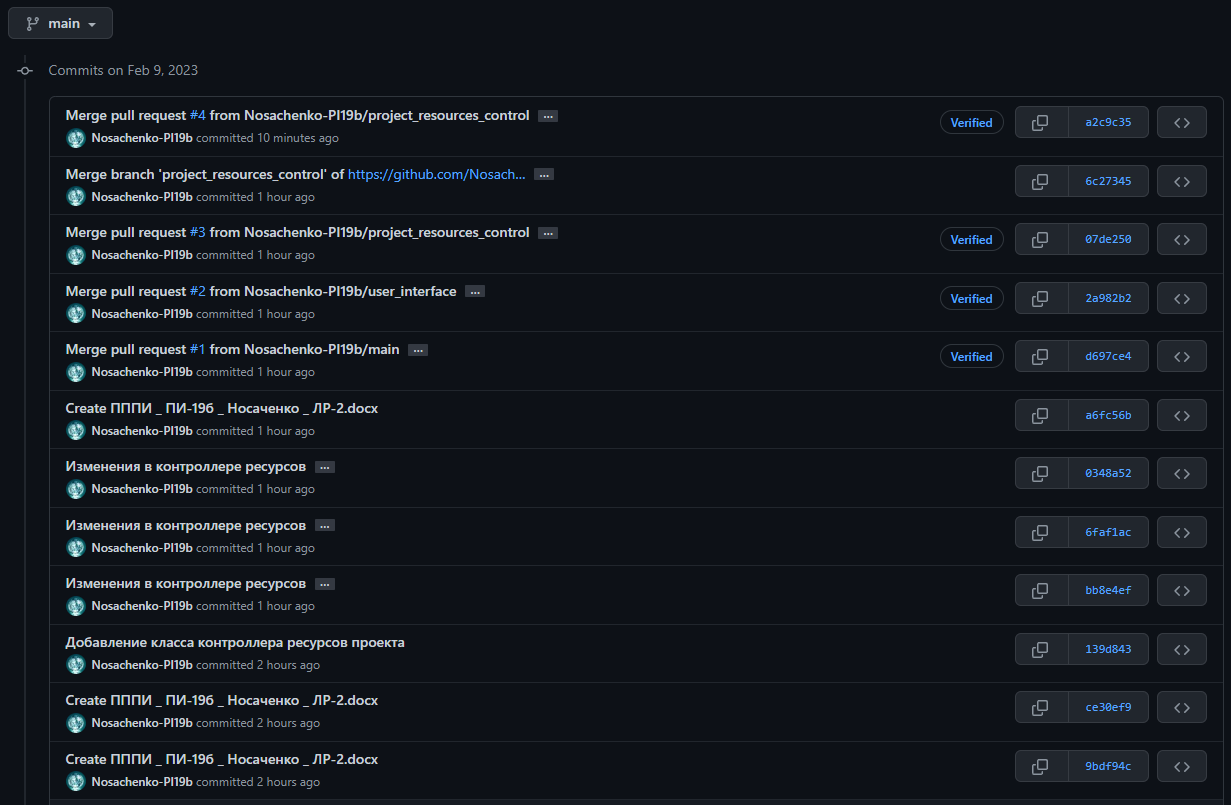










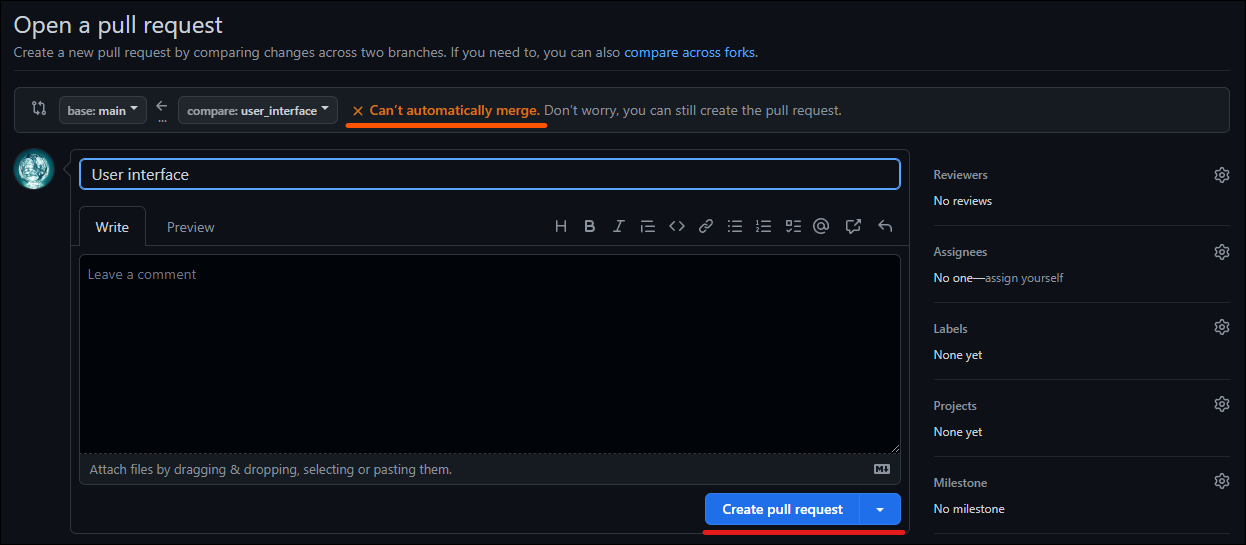


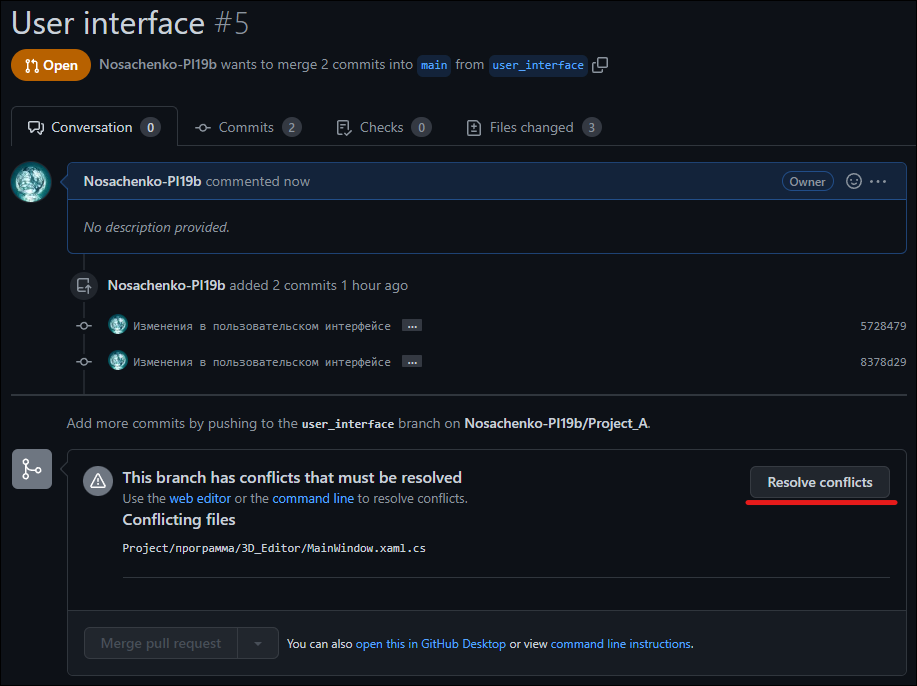
Консольная команда слияния, которая отправляет все коммиты указанной ветки в её базовую ветку:

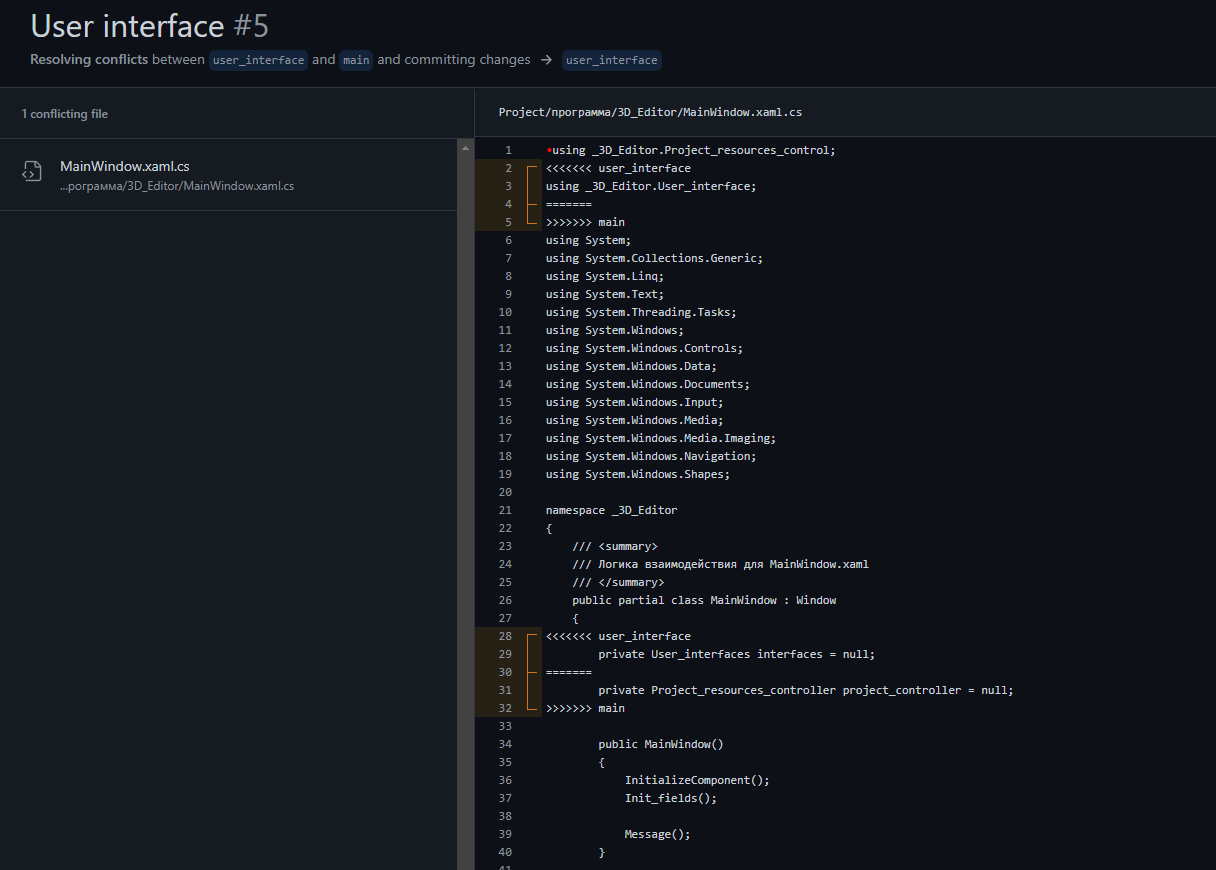
git merge project\_resources\_control

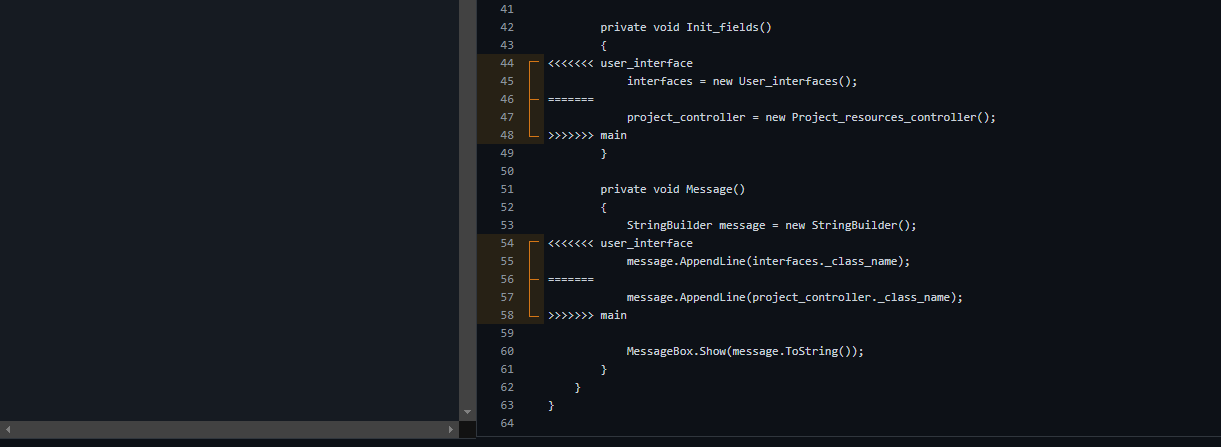
13) Запрос слияния main-ветки и ветки пользовательского интерфейса с решением конфликтов:

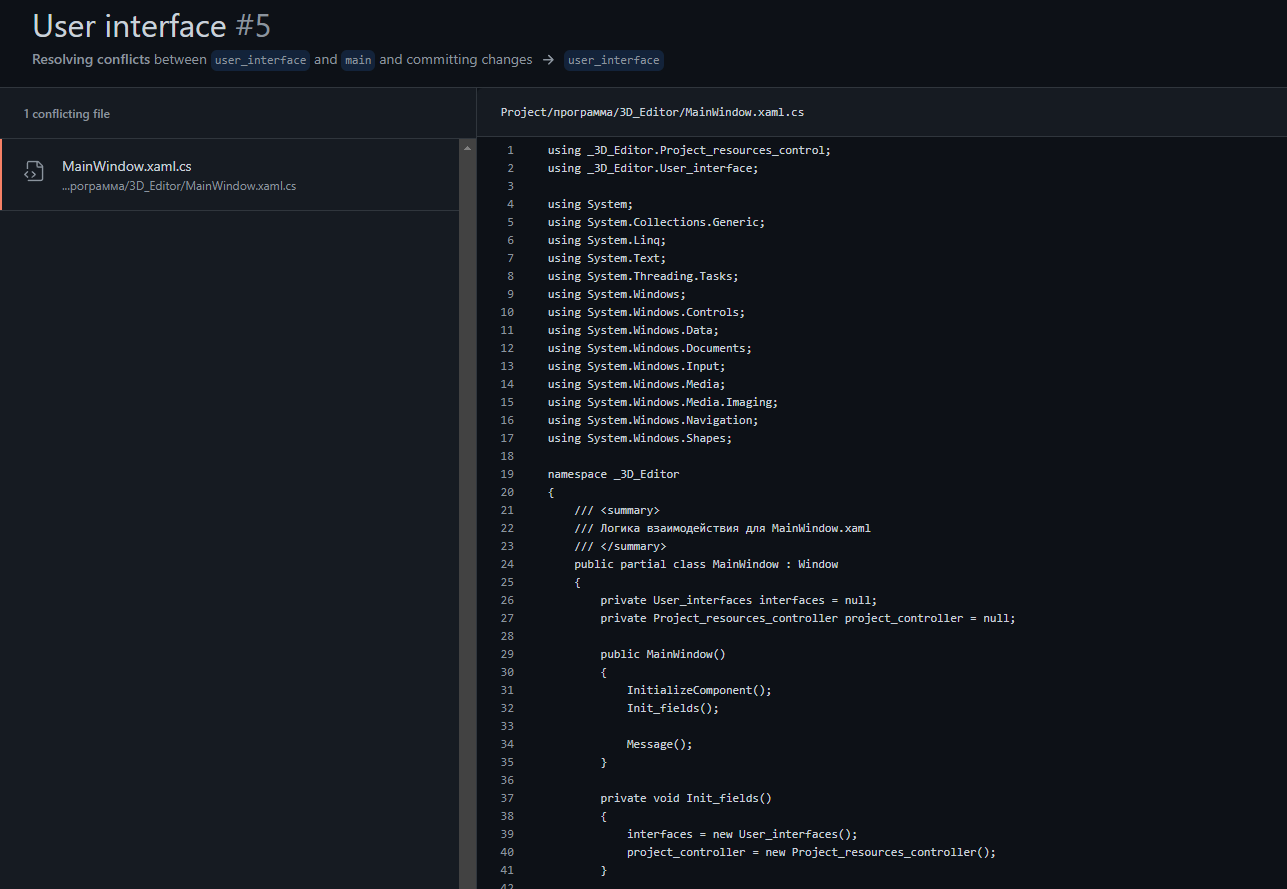


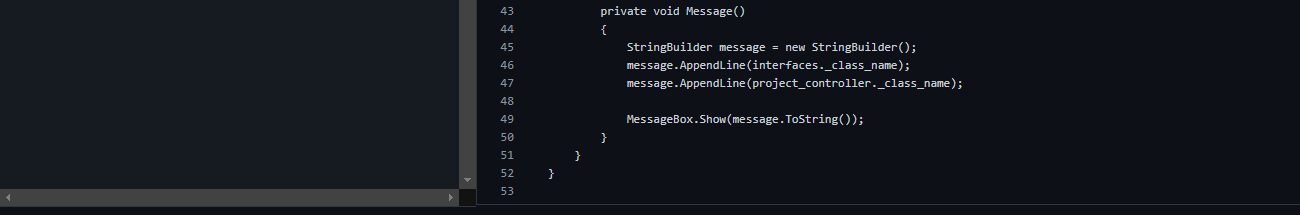


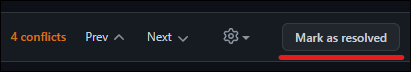
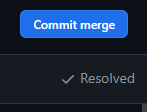


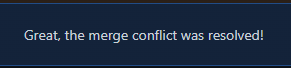


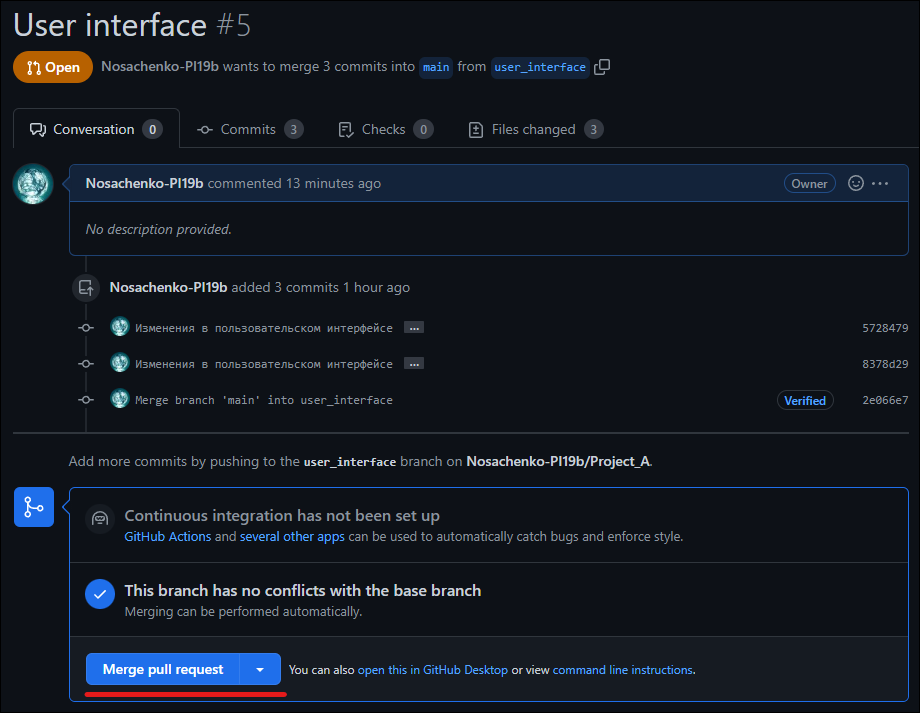


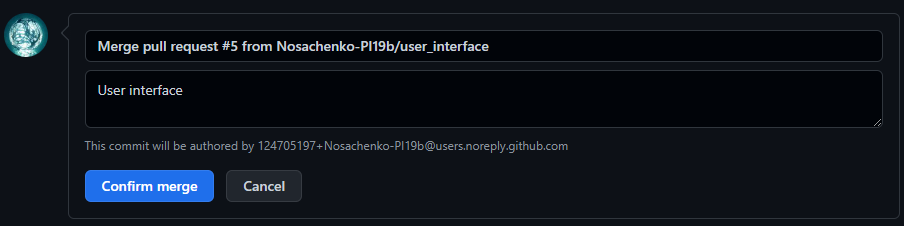


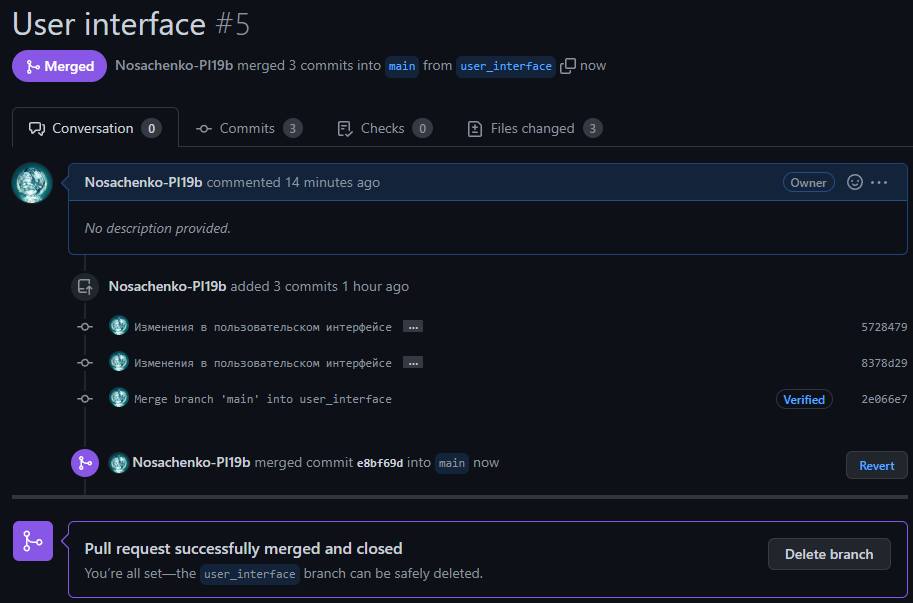


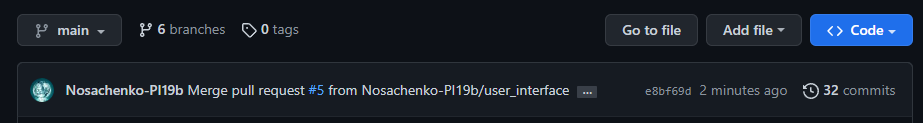
 

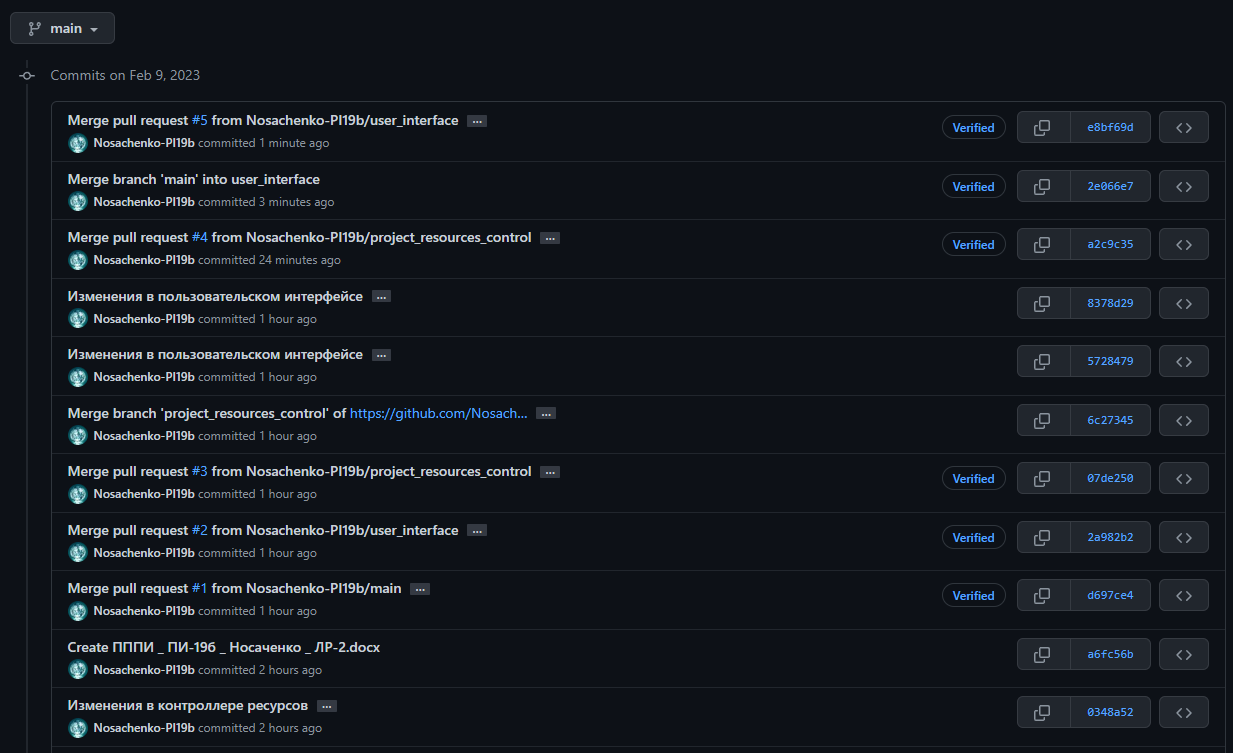




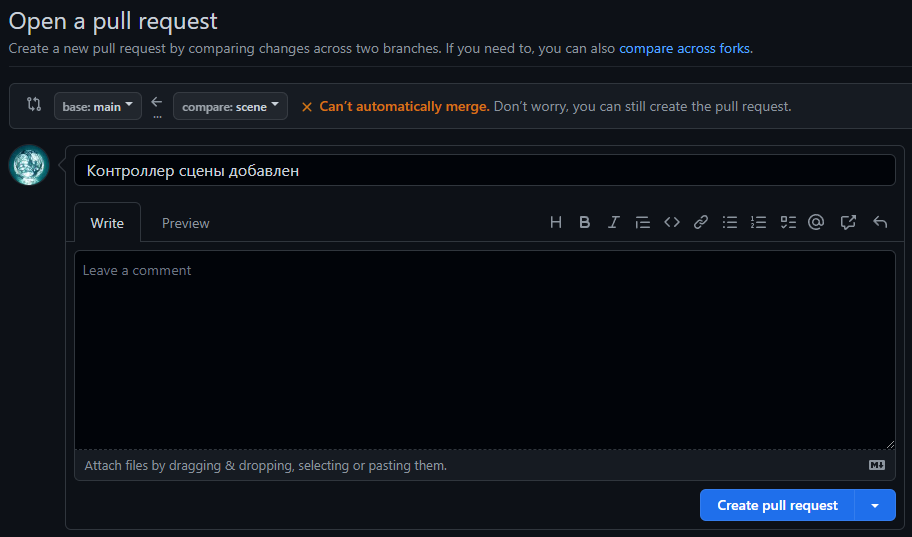




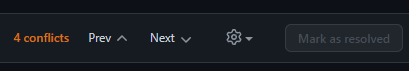


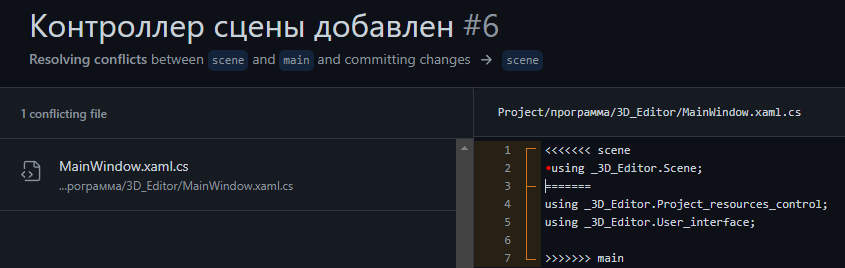


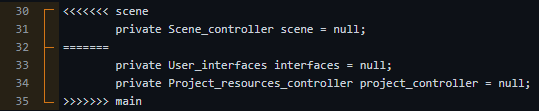
14) Запрос слияния main-ветки и ветки управления сценой:

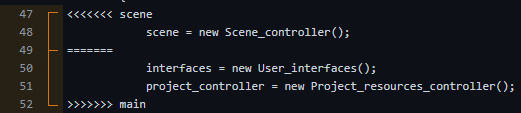


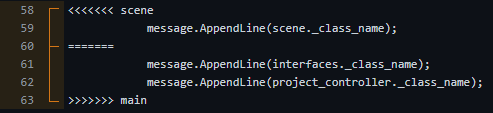




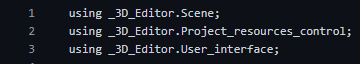








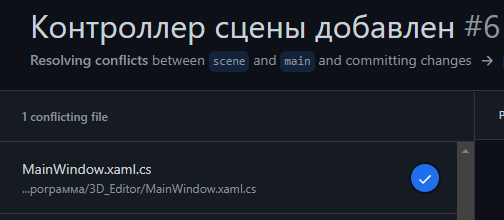
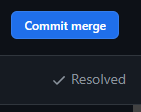
>>>

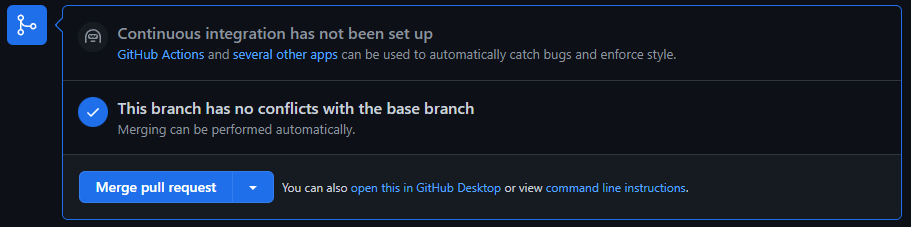


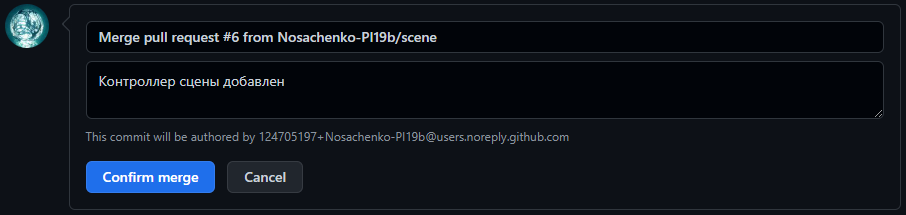




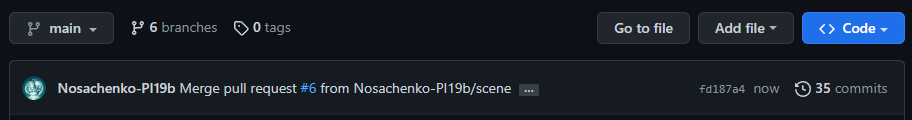


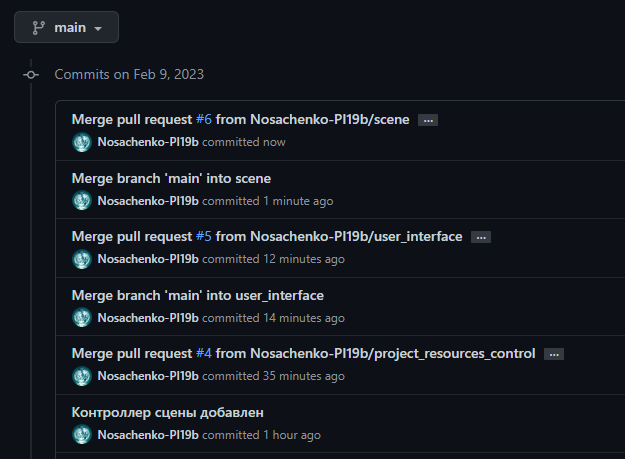
 





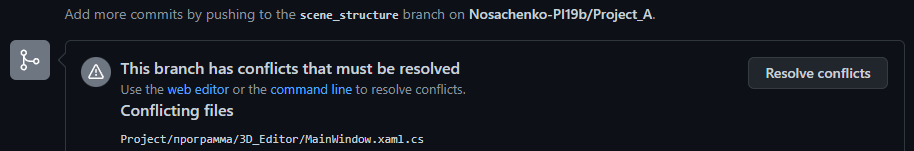




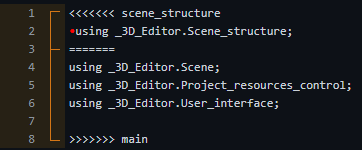


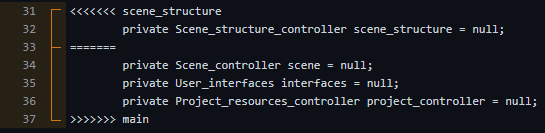
15) Запрос слияния main-ветки и ветки управления структурой сцены:

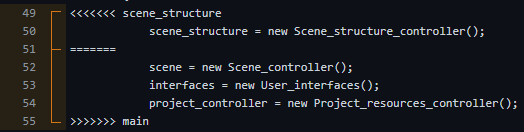
   

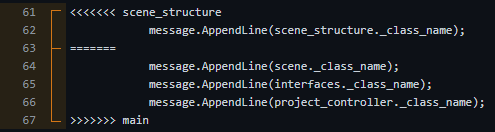




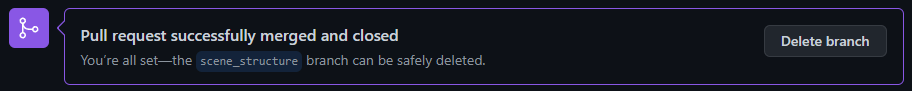


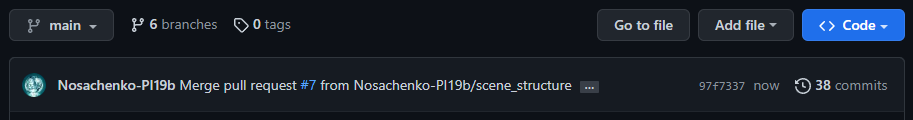


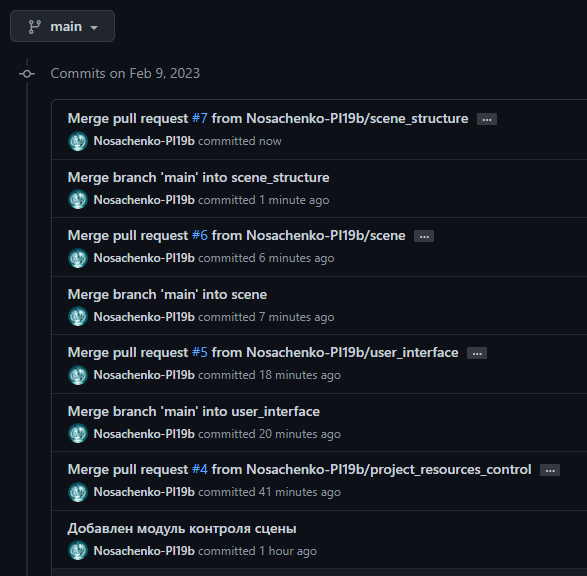






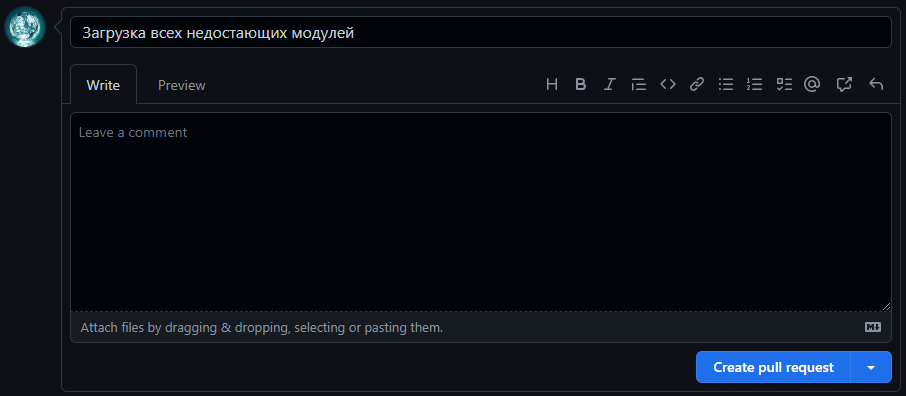


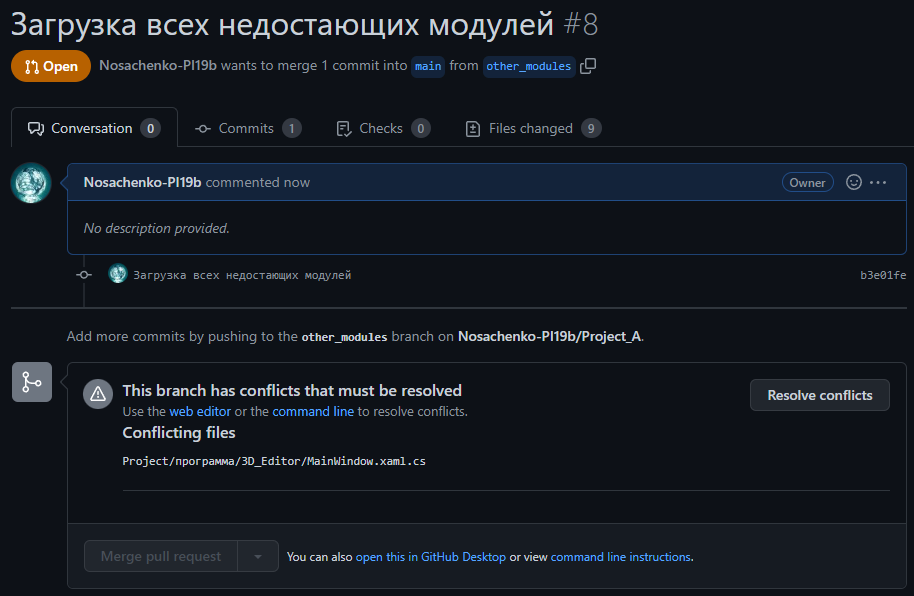


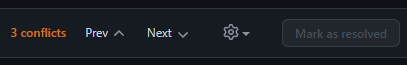
16) Запрос слияния main-ветки и ветки остальных модулей:

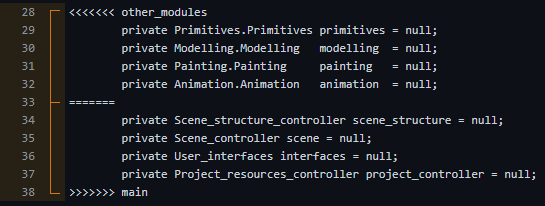


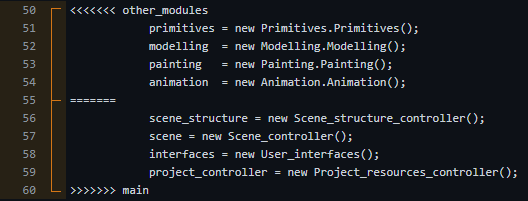


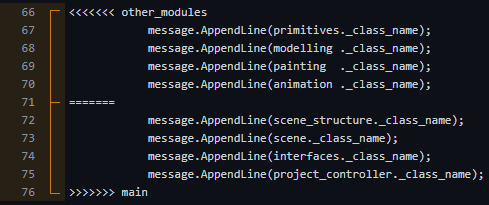




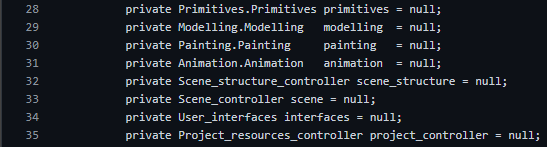


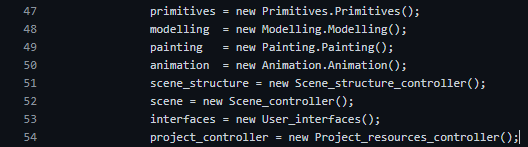


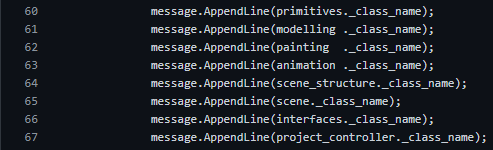


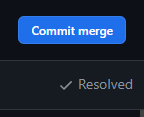


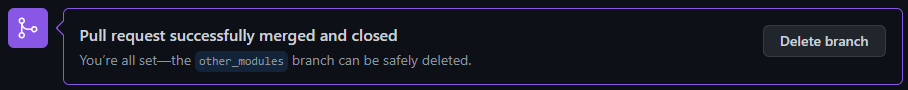
>>>

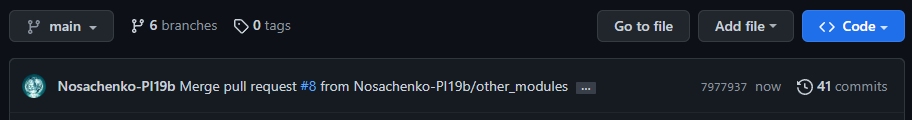






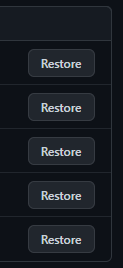






18) Удаление побочных веток:

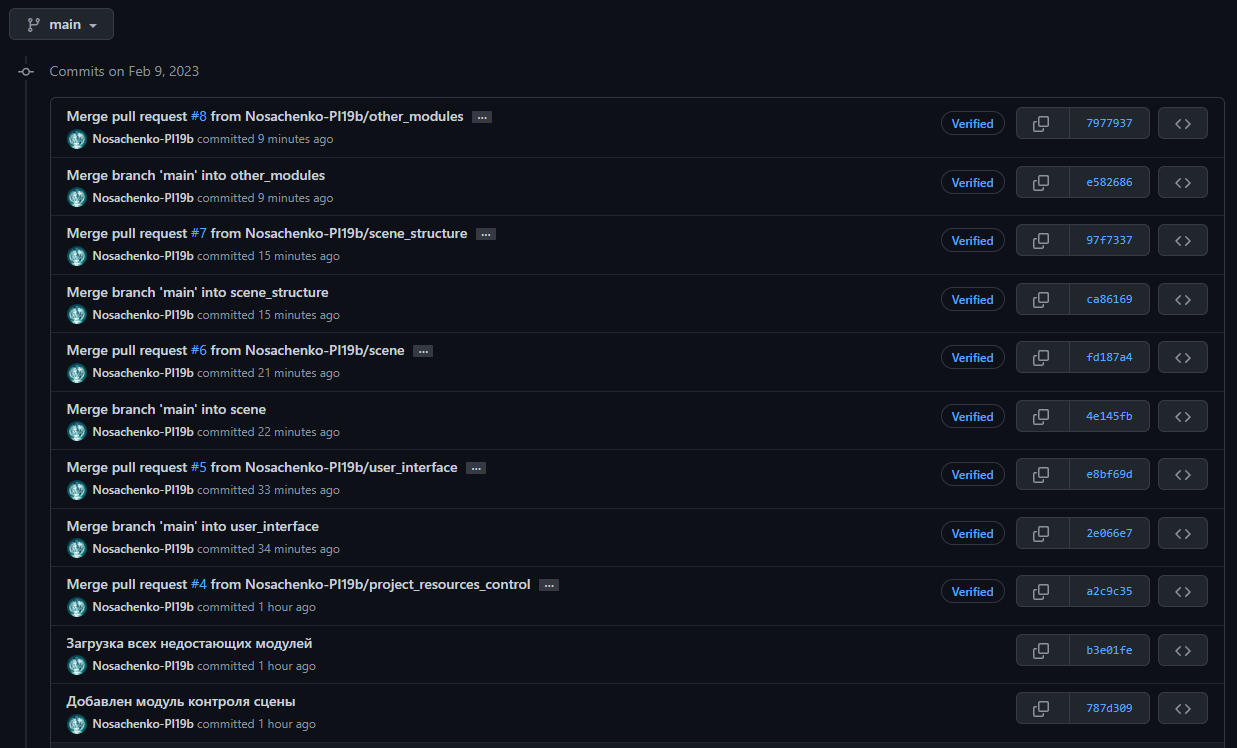


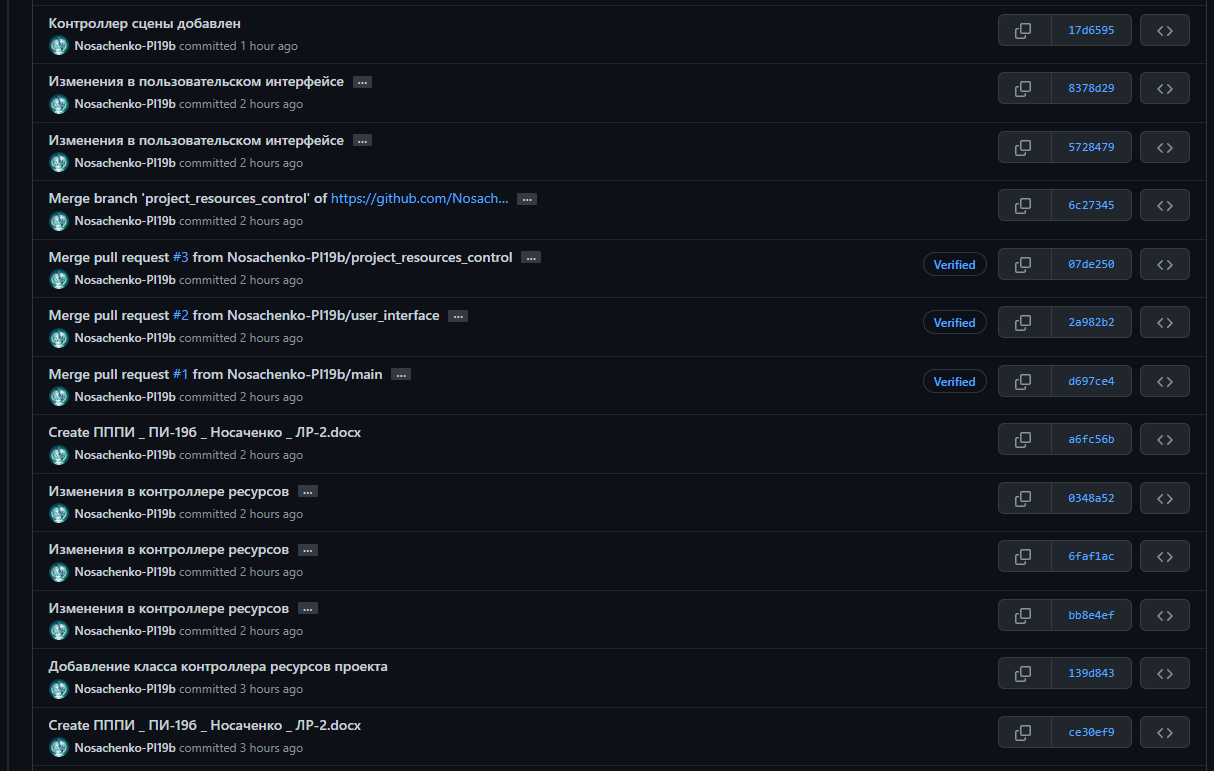
 

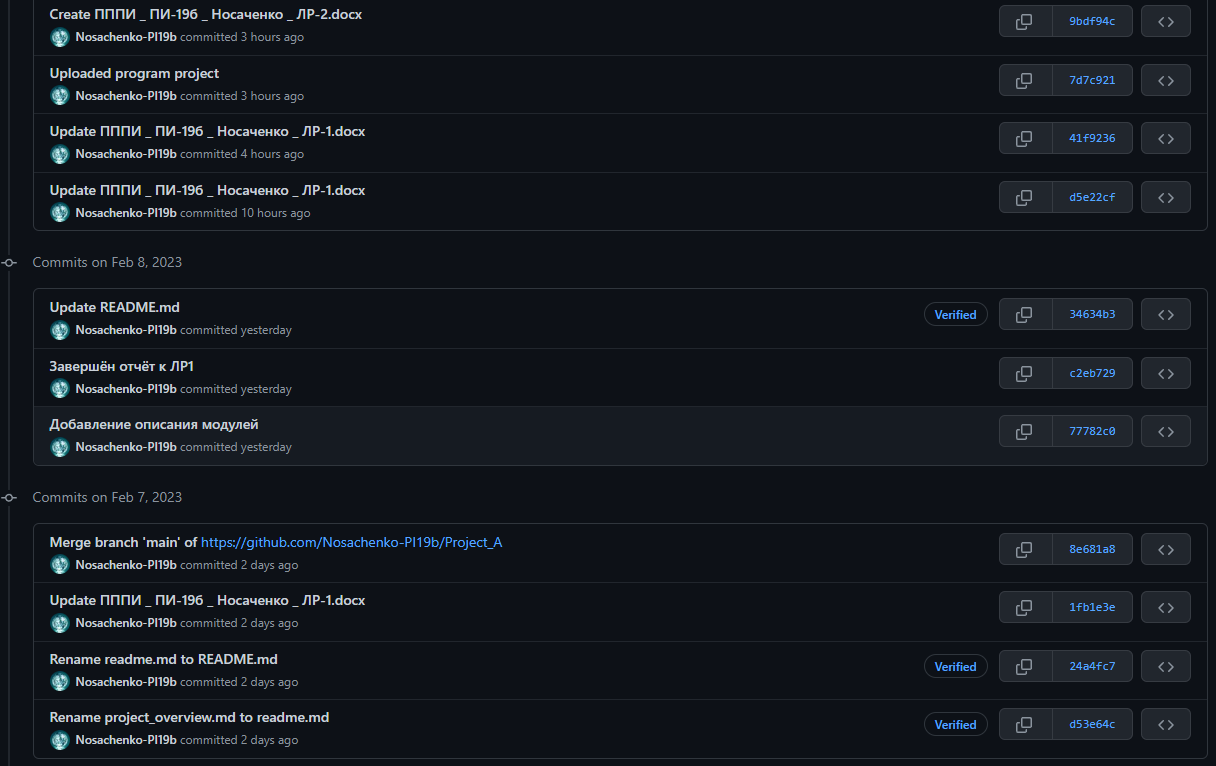
Консольная команда удаления ветки:

git branch -D user\_interface

19) Результирующий список коммитов в main-ветке:







Вывод: в ходе лабораторной работы мы познакомились с основами использования веток в системе контроля версий Git, конфликт слияния веток возникает, если несколько веток, берущих начало из одной родительской, изменяют один и тот же файл по-разному.