**Лабораторная работа №3**

**Выбор платформы и декомпозиция проекта.**

**Основы использования системы контроля версий Git.**

Наша команда собирается использовать язык программирования Java для создания приложения. Вот причины, по которым мы выбрали этот язык:

* поддержка концепции объектно-ориентированного программирования: Java поддерживает ООП, что способствует более структурированному и организованному коду, облегчая его поддержку и повторное использование;
* кроссплатформенность: программы, написанные на Java, могут запускаться на любой платформе, поддерживающей Java Virtual Machine (JVM);
* безопасность: Java имеет встроенные механизмы безопасности, такие как управление доступом, проверка байт-кода и механизмы для защиты от вредоносных действий;
* большая стандартная библиотека: Java предлагает обширную стандартную библиотеку, которая включает множество готовых классов и методов для работы с сетью, вводом-выводом, базами данных и многим другим;
* многопоточность: Java поддерживает многопоточность на уровне языка, что позволяет разработчикам создавать высокопроизводительные приложения, способные эффективно использовать ресурсы процессора;
* сообщество и поддержка: Java имеет одно из самых больших сообществ разработчиков, что обеспечивает доступ к множеству ресурсов, библиотек и инструментов;
* поддержка современного программирования: Java постоянно обновляется и адаптируется к современным требованиям, включая поддержку функционального программирования и новых языковых конструкций;
* инструменты разработки: существуют множество мощных IDE (например, IntelliJ IDEA, Eclipse), которые значительно упрощают процесс разработки и отладки приложений на Java;
* несложный синтаксис: ясный и структурированный синтаксис делает код более читаемым, что упрощает его поддержку и совместную работу в команде.

**Работа с Git**



Рисунок . Установленная версия

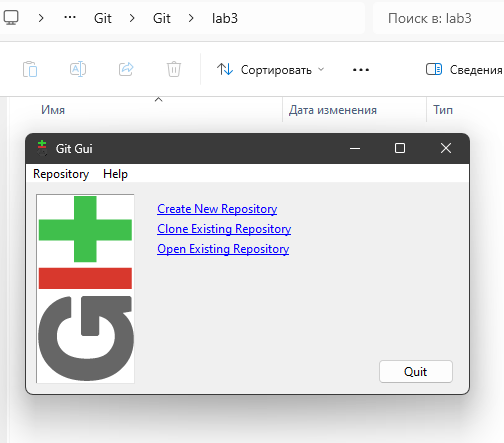


Рисунок . Создание пустой папки проекта

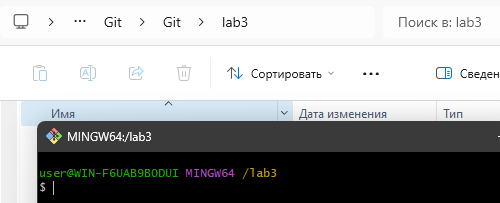


Рисунок . Создание пустой папки проекта на консоли



Рисунок . Создание пустого репозитория Git

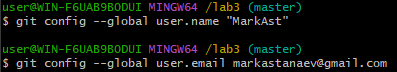
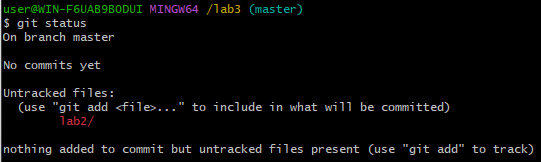


Рисунок . Создание подписей репозитория



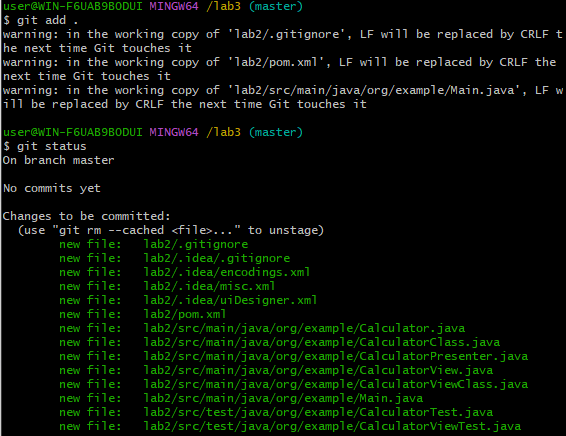


Рисунок . Добавление всех файлов и папок в список сохраняемых

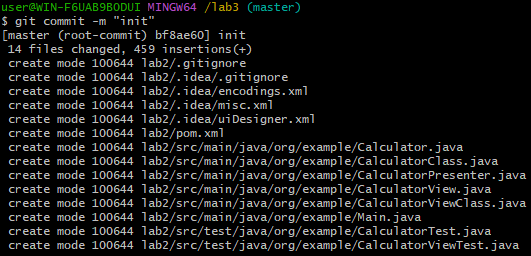


Рисунок . Коммит без изменений в файлах

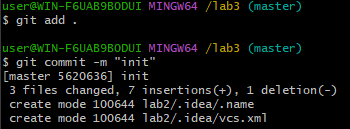


Рисунок . Коммит после внесённых изменений

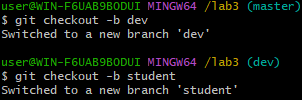


Рисунок . Создание веток dev (бета-версия проекта) и student (рабочая ветка)

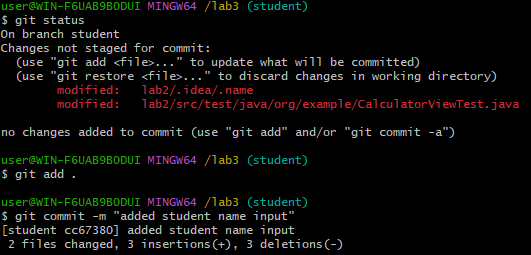


Рисунок . Внесение изменений в текст программы и выполнение коммита в рабочей ветке student

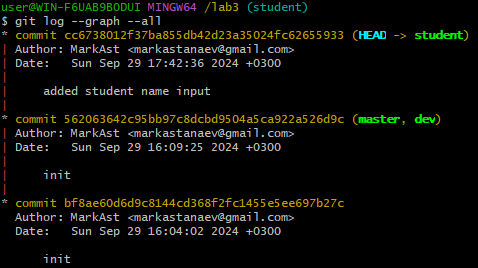


Рисунок . История коммитов

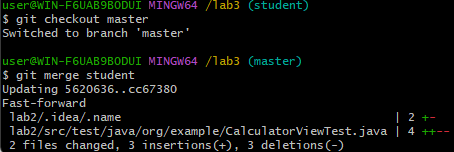
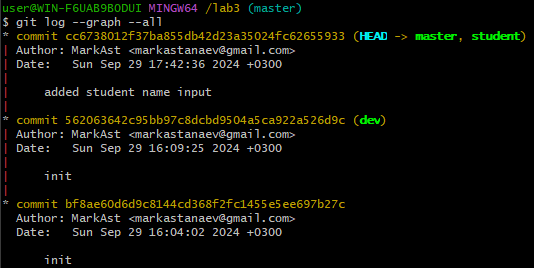


Рисунок . Переключение на ветку master и выполнение слияния веток



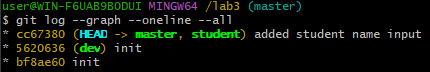


Рисунок . История после слияния веток

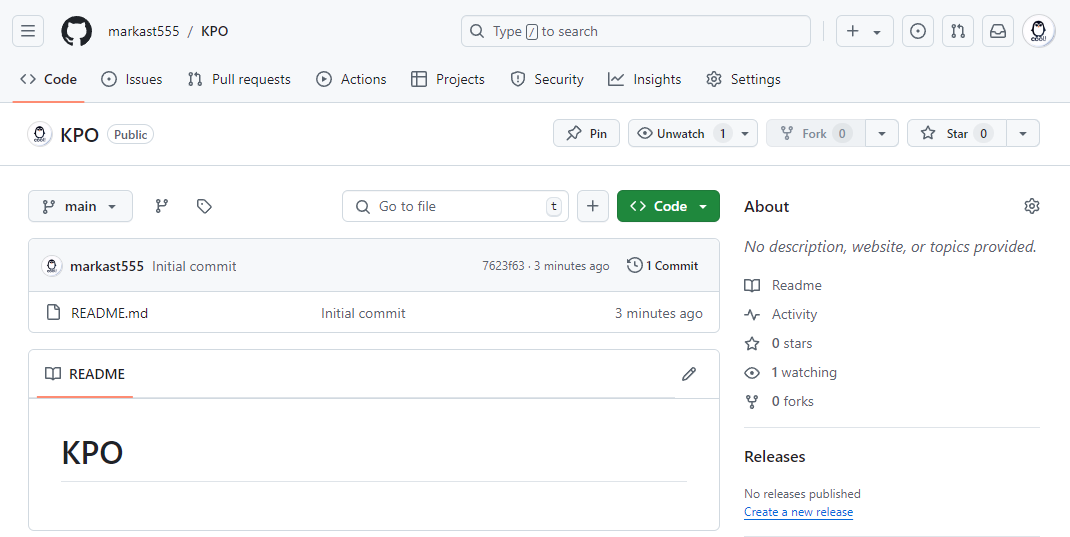


Рисунок . Создание удалённого репозитория

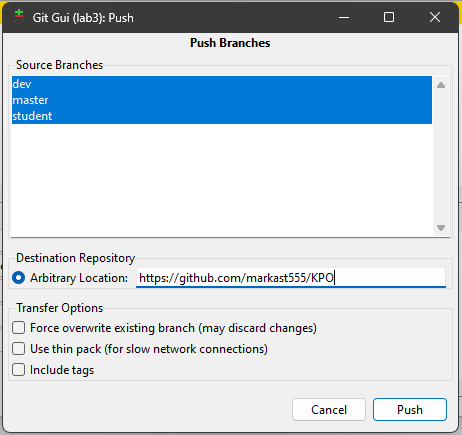
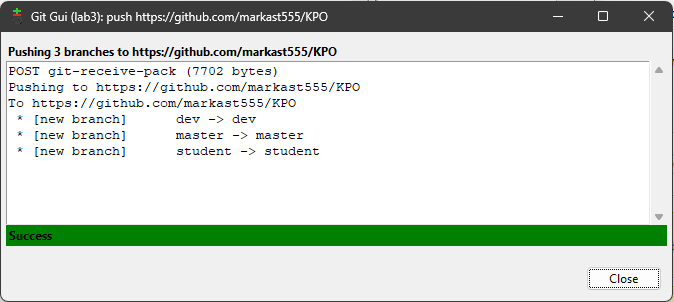


Рисунок . Вставка ссылки на удалённый репозиторий



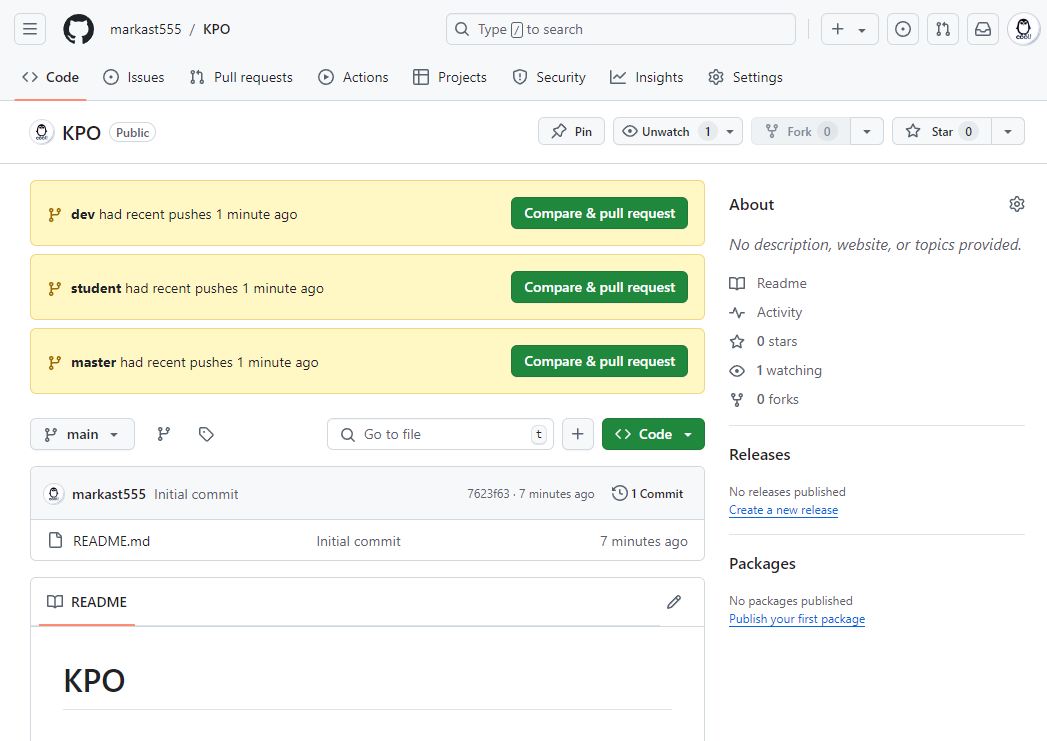


Рисунок . Успешная отправка на удалённый репозиторий

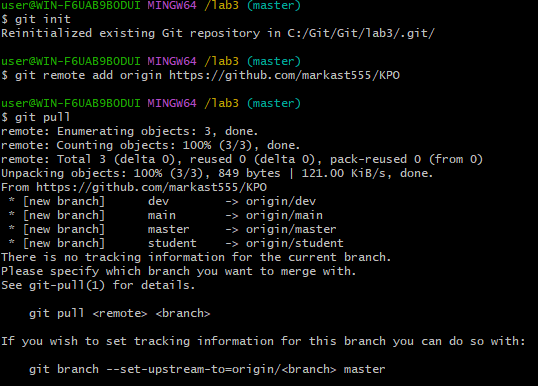


Рисунок . Размещение на удалённый репозиторий



Рисунок . Обновление ветки dev с удалённого репозитория

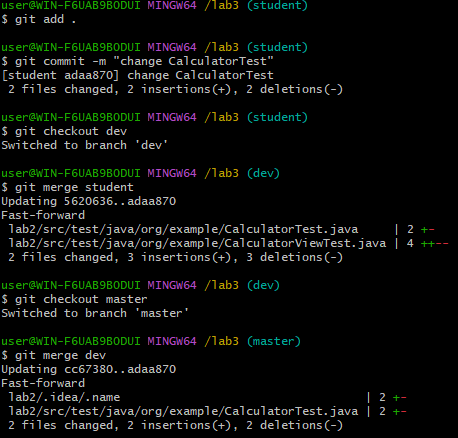


Рисунок . Добавление изменений и слияния веток

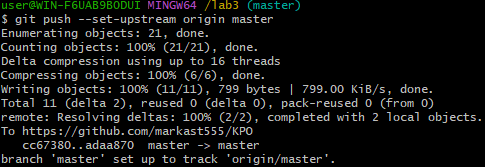


Рисунок . Синхронизация локального и удаленного репозиториев

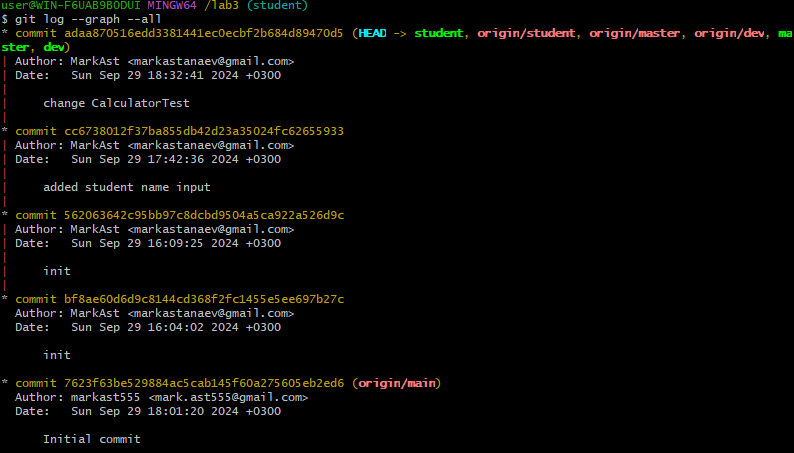


Рисунок . Итоговый вид дерева коммитов

[Удалённый репозиторий](https://github.com/markast555/KPO)

Работа с удалённым репозиторием

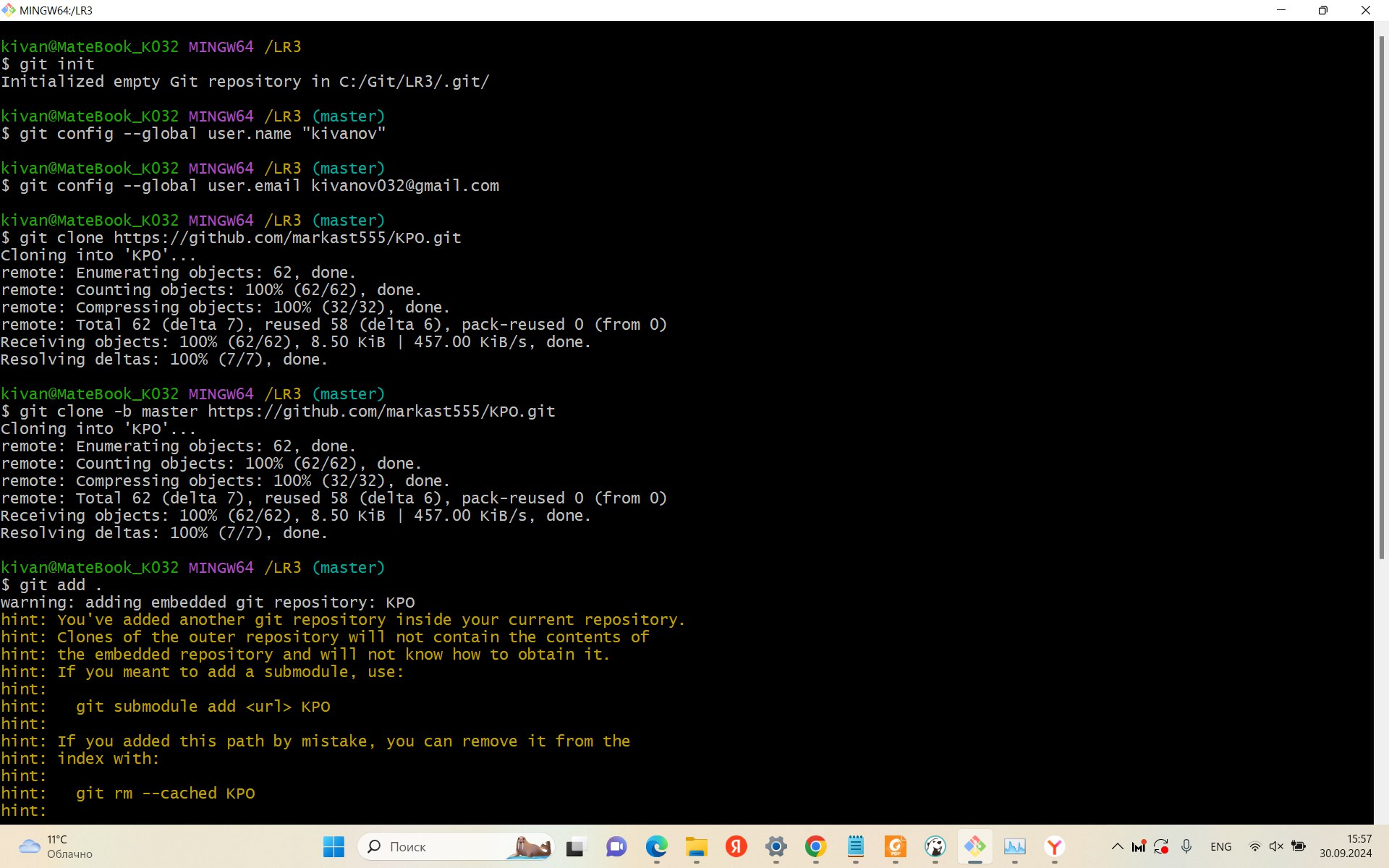


Рисунок . Создание локального репозитория и подписей к нему

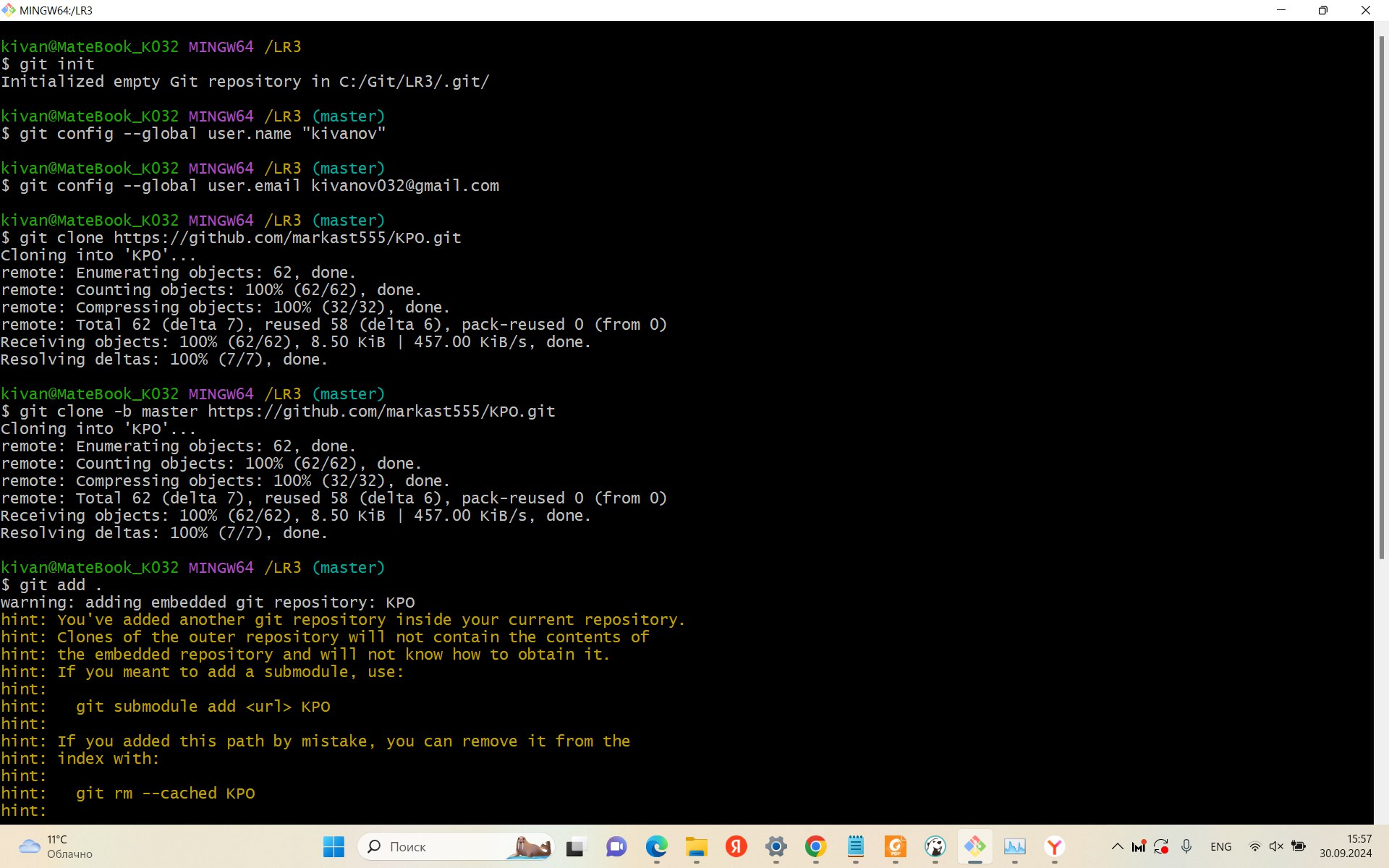


Рисунок . Клонирование ветки master с удалённого репозитория

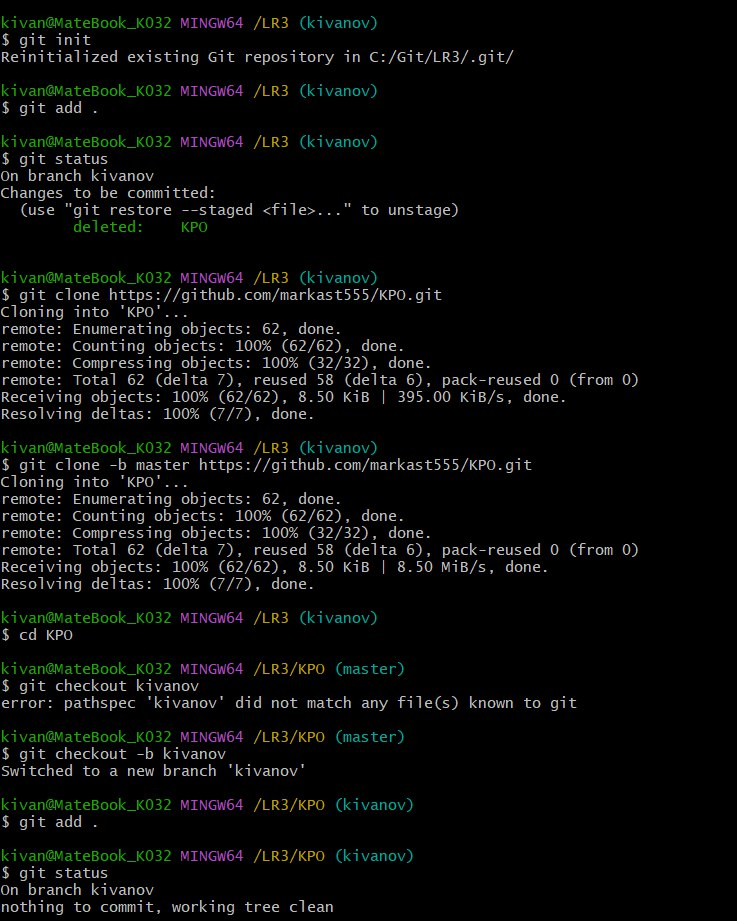


Рисунок . Добавление изменений после клонирования и создание рабочей ветки

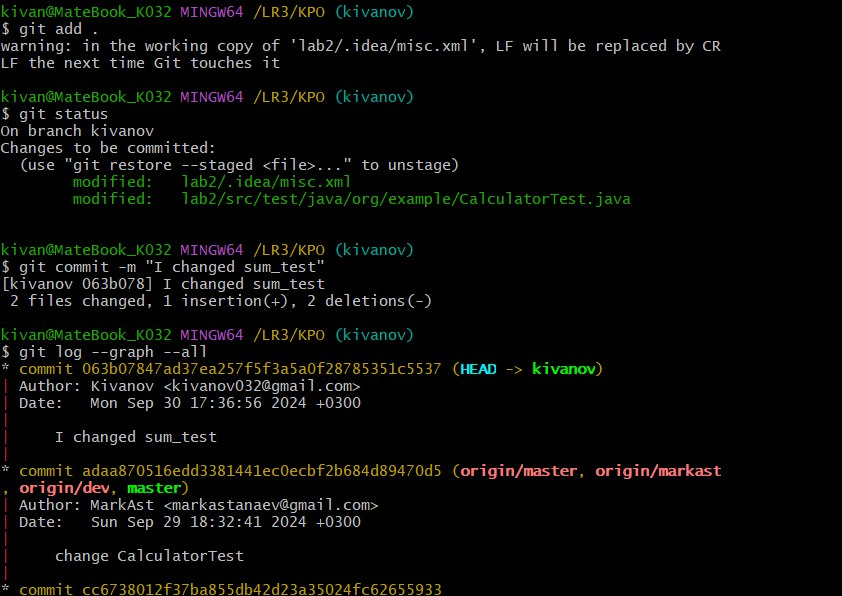


Рисунок . Внесение изменений в один из файлов и добавление коммита

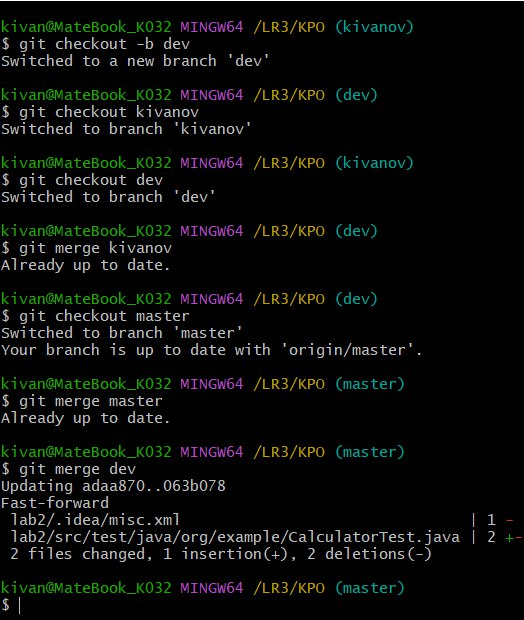
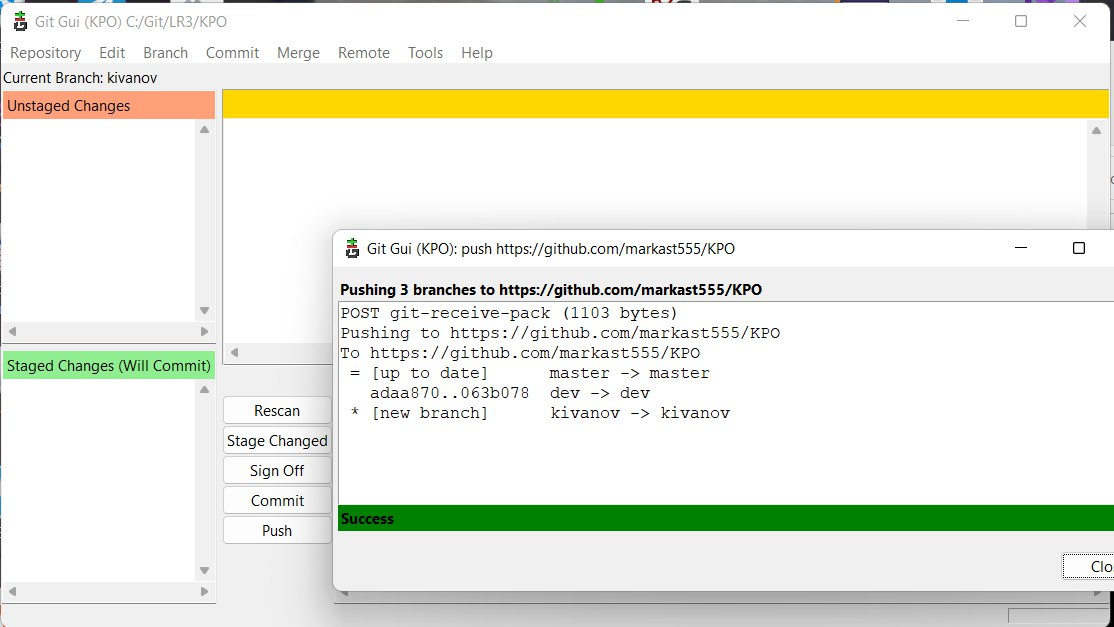


Рисунок . Слияние рабочей ветки с веткой master



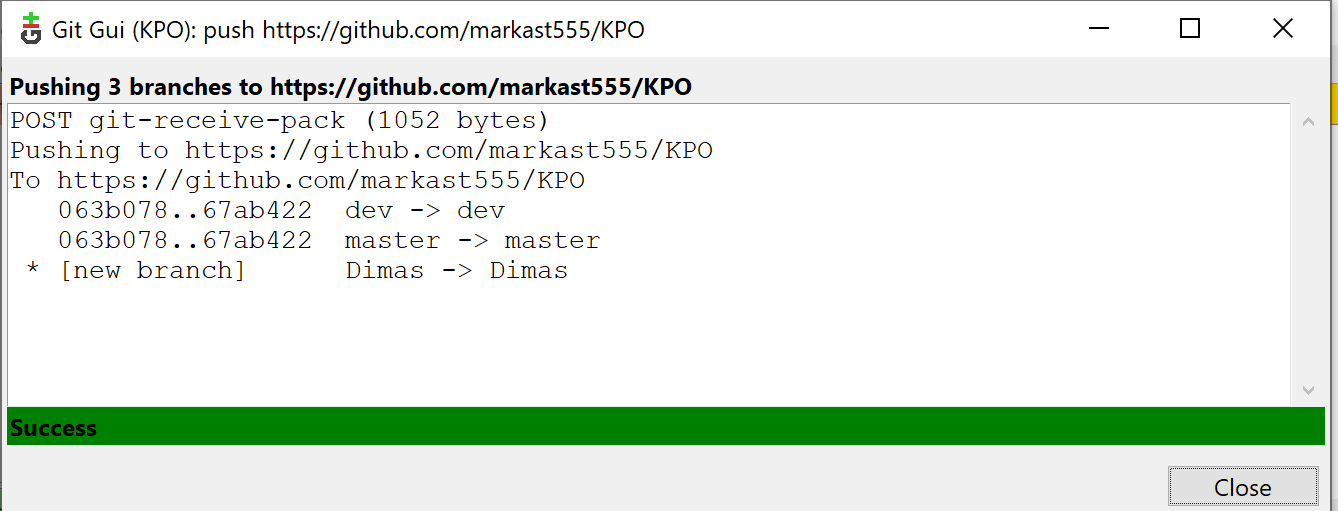
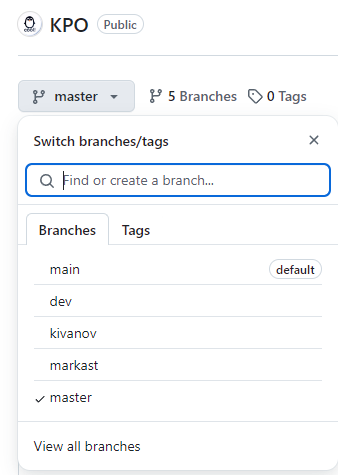
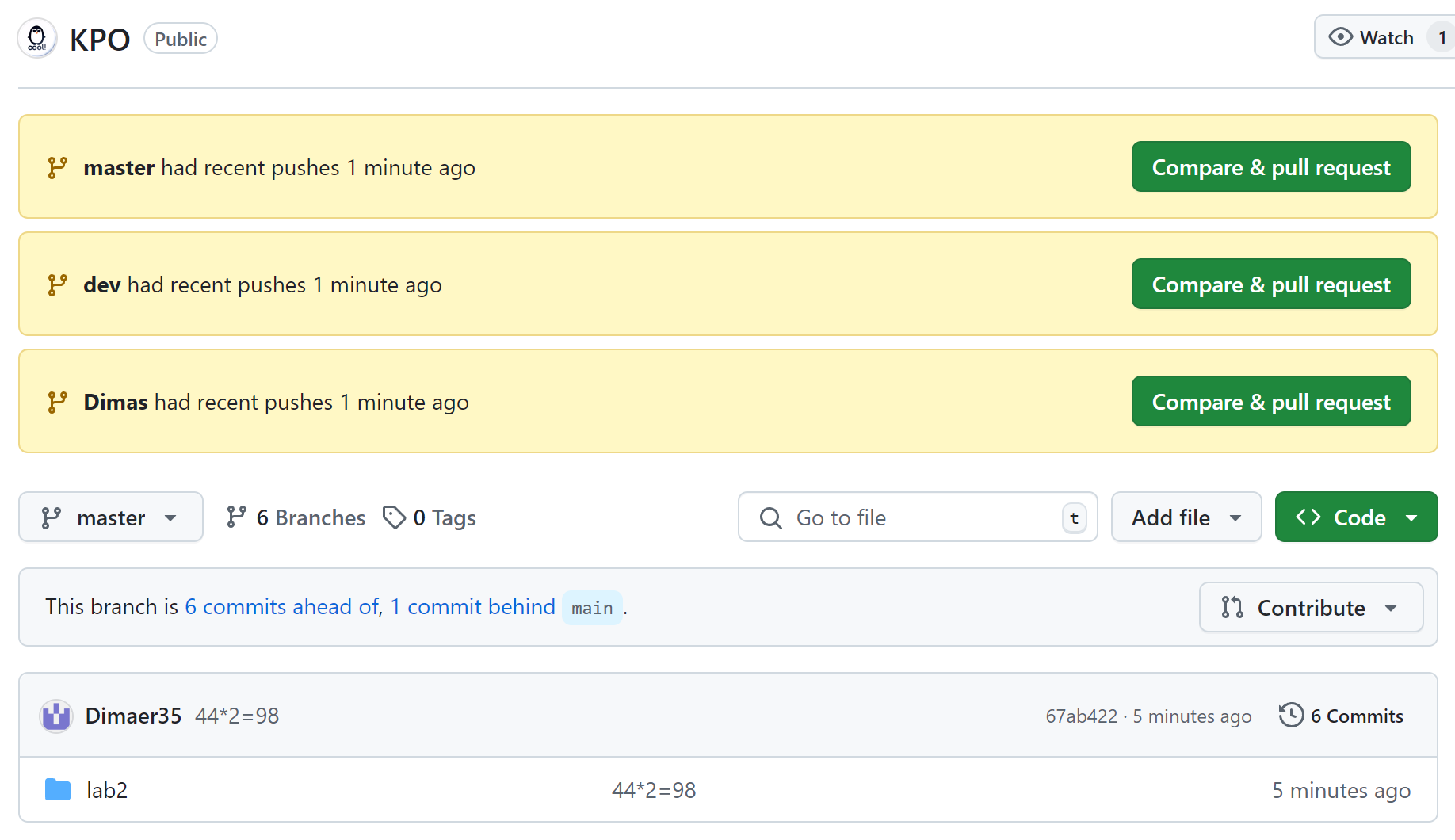


Рисунок . Успешная отправка на удалённый репозиторий





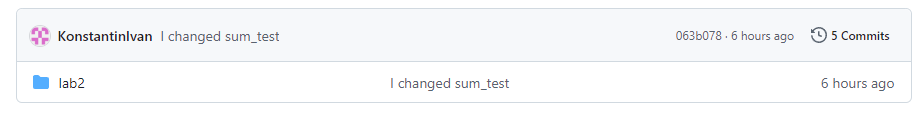


Рисунок . Успешные изменения