**Отчёт по лабораторной работе №3. Знакомство с системой GIT**

Работу выполнял Сергеев Дмитрий из группы БИС23-01

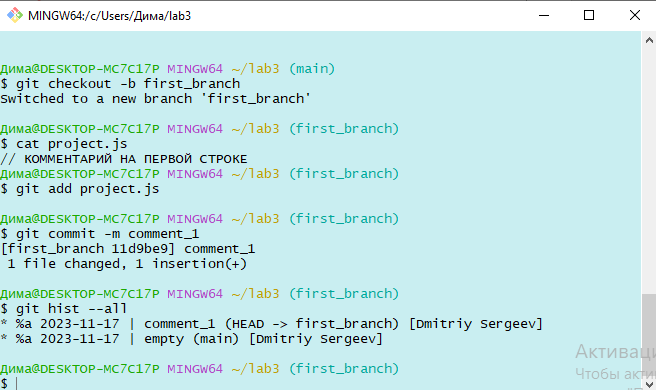
10 Вариант: использовать 3 ветки, создать 2 конфликта и решить их.

**Ход работы**

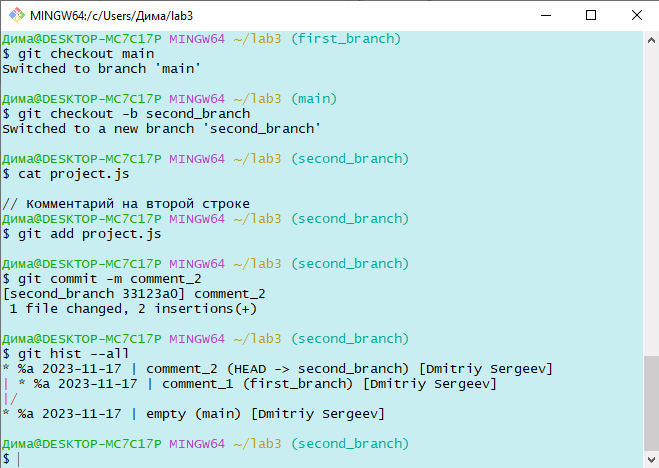
Создадим в каталоге пользователя папку **lab3**, перейдём в неё и уже в папке создадим файл **project.js**. Создадим в ней же репозиторий, добавим наш пустой файл в репозиторий и закоммитим это.



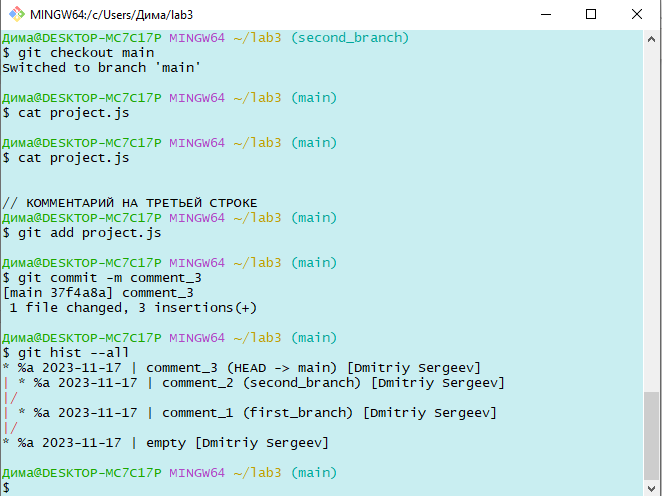
Затем создадим ветку **first\_branch**, внесём изменения в файл и сохраним их, посмотрим итог.



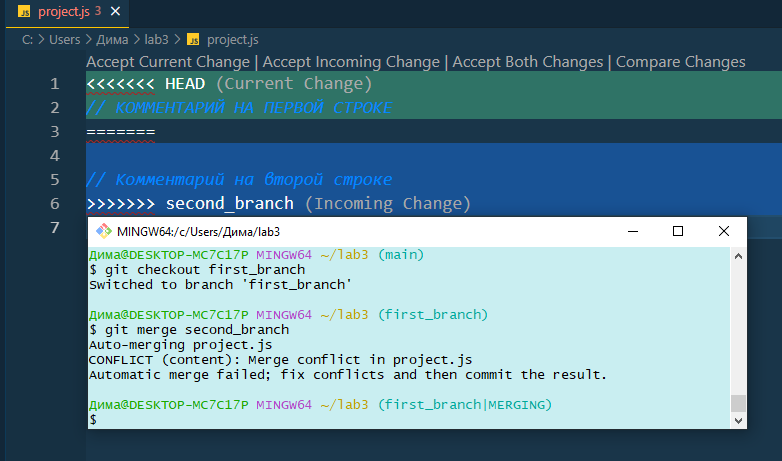
Вернёмся в main, создадим ещё одну ветку и назовём её **second\_branch**, изменим файл, сделаем коммит и посмотрим результат.



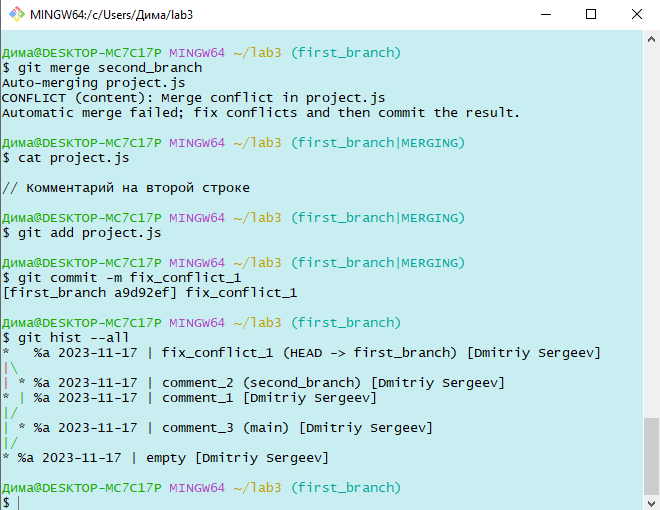
Снова вернёмся в **main**, изменим файл пустой файл, сделаем коммит и посмотрим на результат.



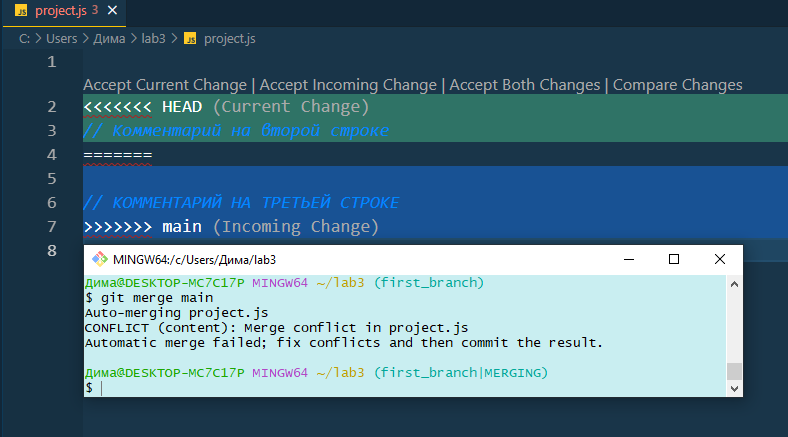
Итого, теперь у нас есть 3 ветки с разным содержимым.   
Выполним их слияние.

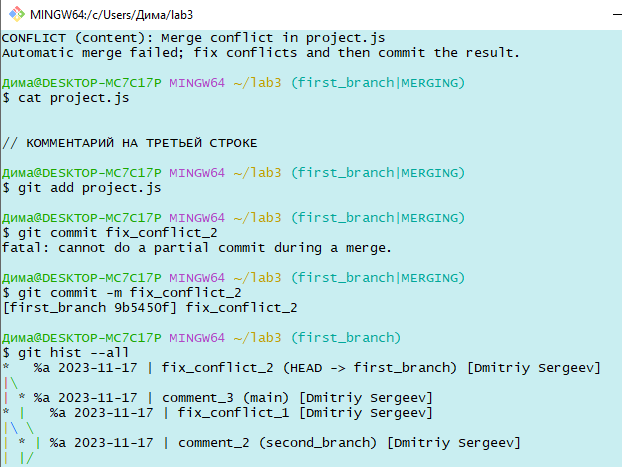
Сделаем слияние **first\_branch** с **second\_branch**, и получим конфликт из-за разного содержимого файлов, для устранения конфликта воспользуемся подсказкой от VS CODE, выбрав версию второй ветки **Accept Incomming Change**. 

Затем добавим изменения и закоммитим их. Посмотрим на результат, и увидим удачное слияние веток.



Теперь сделаем слияние ветки **main** и ветки **first\_branc**.   
Мы снова получим конфликт, выберем версию ветки **main**, добавим изменения, закоммитим их и посмотрим на результат.





**Выводы**

Проделав эту работу, я лучше понял принципы работы систем контроля версий, вспомнил основные команды для работы с GIT, узнал что такое конфликты и как я могу их решить.

**Ответы на контрольные вопросы**

1. Что представляет собой система GIT?

Git – ПО, являющееся системой контроля версий.

2. Перечислите возможности систем управления версиями.

С помощью системы контроля версий можно отслеживать какие изменения были произведены в файлых, кем они были сделаны. Если потребуется изменение можно откатить назад. Важной функцией является создание веток, с помощью которых создаются разные направления развития проекта, со своими целями, измениями и т.п.

3. Что такое коммит и когда он выполняется?

Коммит это отправка изменении, которые были совершены в проекте. Совершается после индексирования изменений.

4. Как посмотреть историю коммитов?

С помощью команды - **git log**

5. Как создать пустой GIT репозиторий?

- команда **git init**

6. Какую архитектуру имеет система контроля версий GIT?

Есть рабочая копия(working copy) – набор файлов, с которыми мы работаем.

Есть рипозиторий(repository) – хранилище нашего проекта, обслуживаемое системой контроля версий.

Есть коммит(commit) – операция отправки изменений рабочей копии в репозиторий.

Есть кэш(stage) – место, в котором хранятся изменённые файлы, до того момента, когда будет произведён коммит, и эти изменения отправятся в рипозиторий.

Т.е. – Мы производим изменения в рабочей копии, изменённые файлы попадают в кэш, файлы добавляются в кэш и удаляются из него до того как будет произведён коммит и эти изменения попадут в рипозиторий.

7. Как создать новую ветку в GIT и соединить ветки?

Создание ветки - команда **git checkout -b “Название ветки”**

Для соединения веток нужно перейти в одну одну из них  
- команда **git checkout “Название одной ветки”**

Затем выполнить слудующию команду

- команда **git merge “Название другой ветки”**

8. Поясните понятие «staging area».

Место, в которую попадают изменённые файлы, которые потом будут отправлены в репозиторий с помощью коммита.

9. Как сравнить версии файла в GIT?

Можно использовать следующее

- команда **cat “Имя файла”** для прочтения файла в командной строке.

Затем нужно перейти в нужный коммит и снова прочесть файл, и искать разницу.

Ну или использовать

– команду **git diff** и её вариации для более быстрого и точного нахождения изменений(или их отсутствия).