**Федеральное агентство связи**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра Математической кибернетики и информационных технологий

**Отчет по лабораторной работе**

по дисциплине «Технологии разработки программного обеспечения»

на тему: «Основы работы с системой контроля версий»

Выполнил:

студент группы М091801(71)

Масанов Артём Сергеевич

Москва 2018

**Цель работы:** изучение базовых функций системы контроля версий на примере Git

**Задание:** Определить проект (состоящий из нескольких файлов), который будет добавлен под версионный контроль, и произвести ряд операций в СКВ, включающих:

1. Создание репозитория или клонирование существующего
2. Внесение ряда изменений в файл(ы) и выполнение коммитов (кол-во не меньше 2)
3. Создание веток и выполнение ряда изменений в проекте в созданных ветках
4. Слияние ветки с веткой мастер без конфликтов
5. Слияние ветки с веткой мастер с решением возникающих конфликтов
6. Откат изменений до одного из предыдущих коммитов

**Ход выполнения работы**

1. **Создание репозитория**

Создание репозитория происходит на веб-сайте https:// github.com/.

Так же можно создавать репозиторий через Git Bash следующими командами:

* $ git init - Назначение директории репозиторием
* $ git add filename - Добавление отслеживаемых файлов
* $ git commit –m “initial version” - Стартовый коммит
* $ git clone - C сервера выгружается каждая версия каждого файла из истории проекта

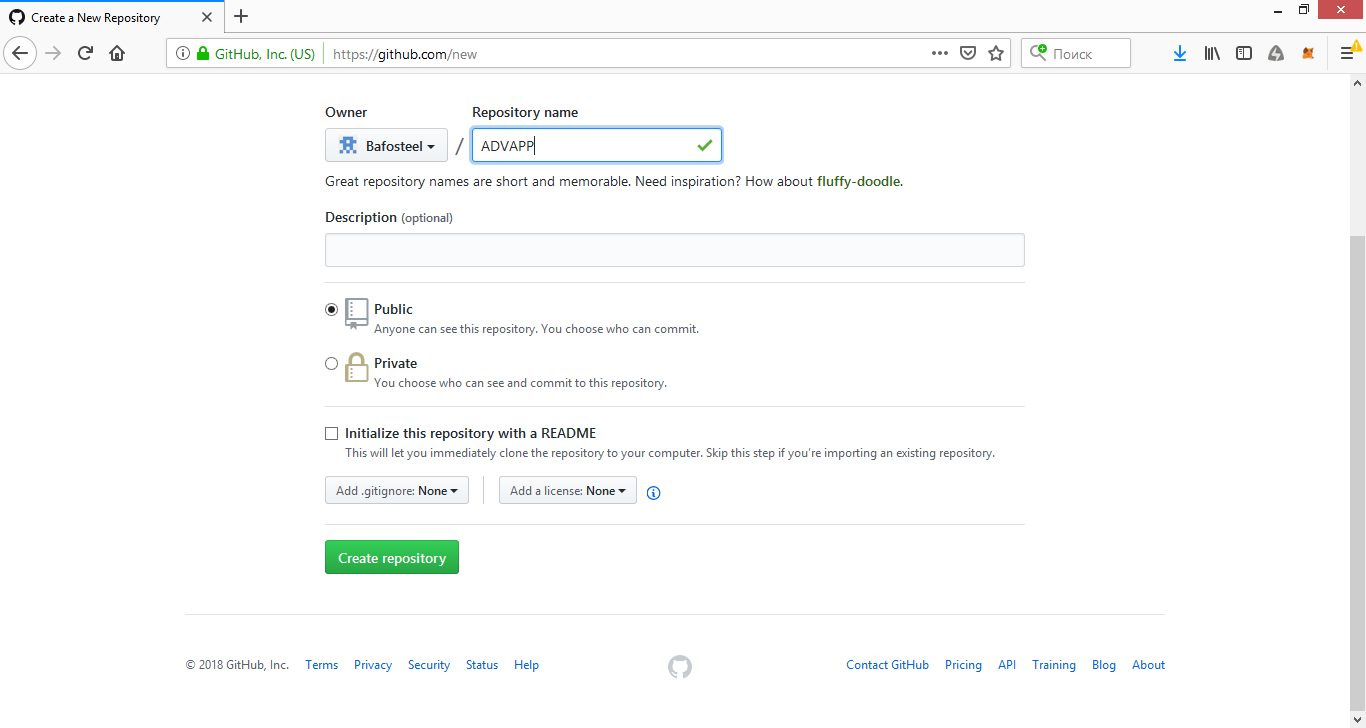
****

Рис.1 Процесс создания репозитория

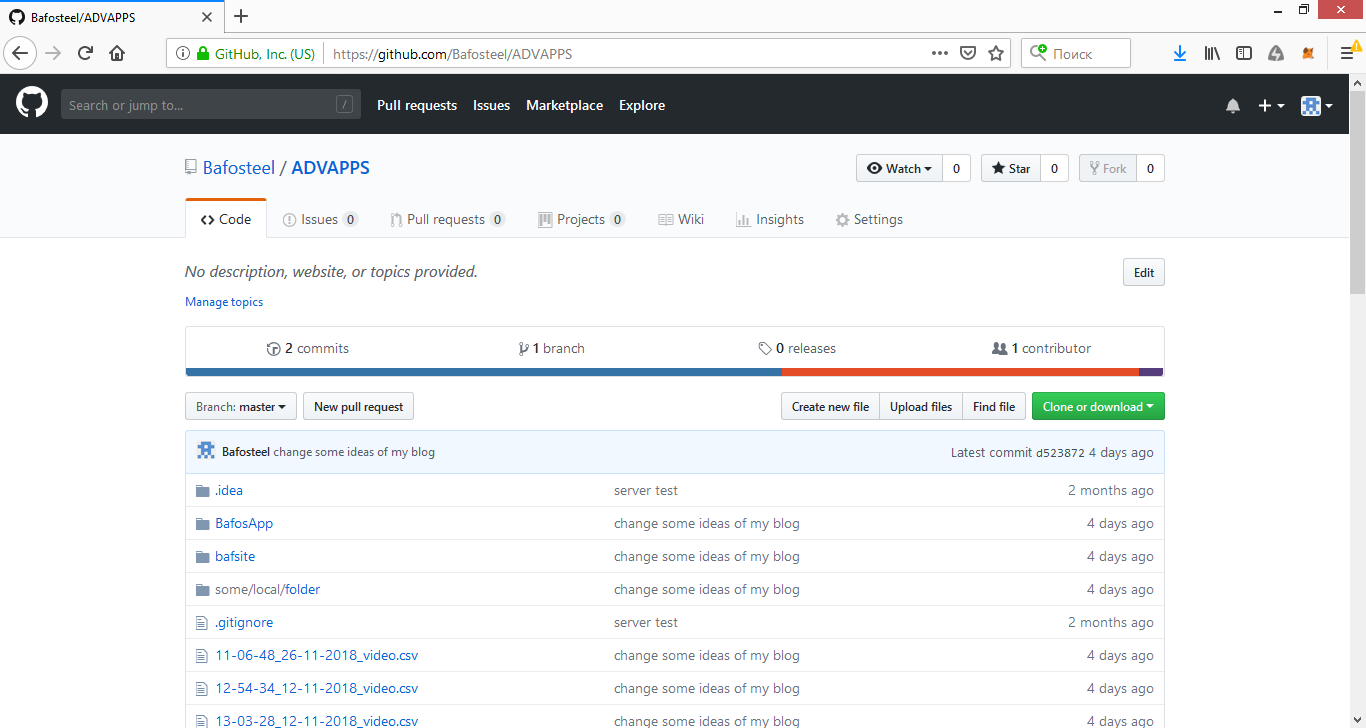
****

Рис.2 Созданный репозиторий

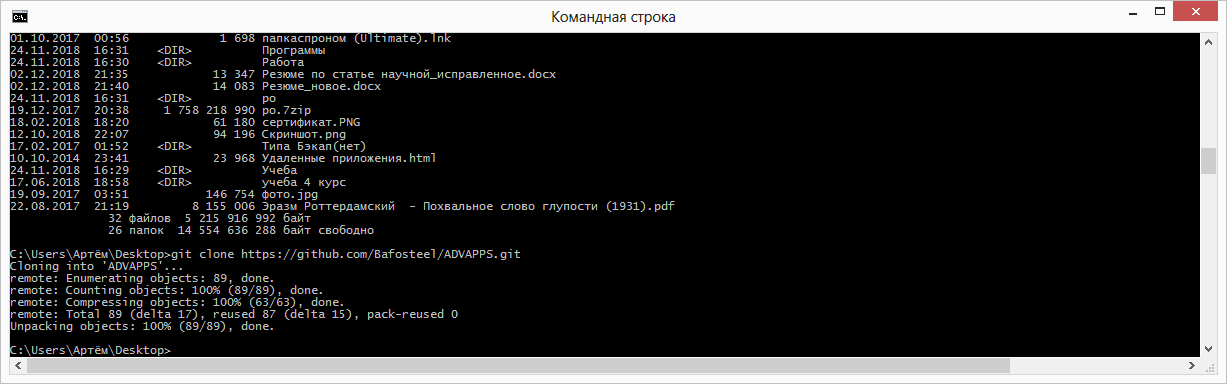
****

Рис.3 Процесс клонирования репозитория на рабочий стол.

Ссылка на репозиторий: https://github.com/Bafosteel/ADVAPPS

1. **Выполнение коммитов**

Для выполнения коммитов можно производить требуемые действия помощью репозитория на сайте, добавляя или удаляя файлы с требуемым комментарием.

Так же можно выполнять коммиты через Git Bash следующими командами:

* $ git status - Состояние репозитория
* $ git add - Добавление файла под версионный контроль или индексация изменений
* $ git commit –m “message” - коммит с текстом
* $ git log - Просмотр истории коммитов

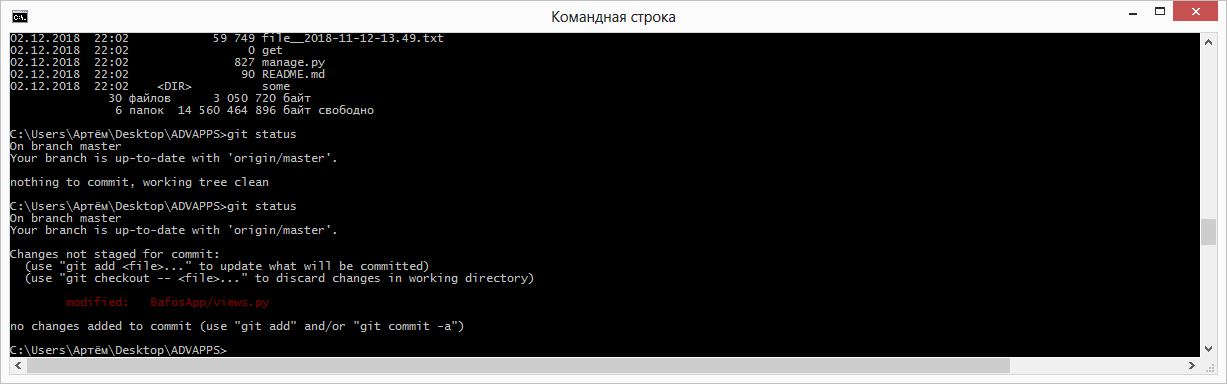


Рис.4 Состояние репозитория.

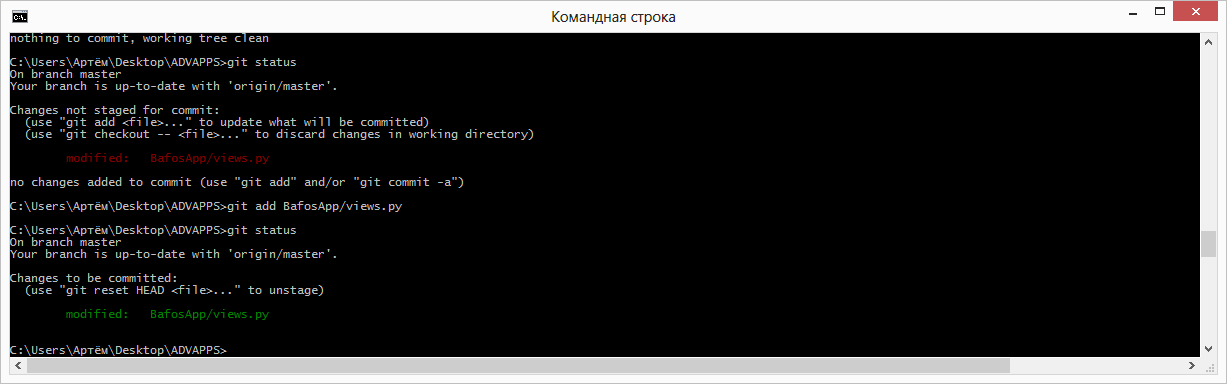


Рис.5 Процесс выполнения коммитов.

(Добавление комментария)

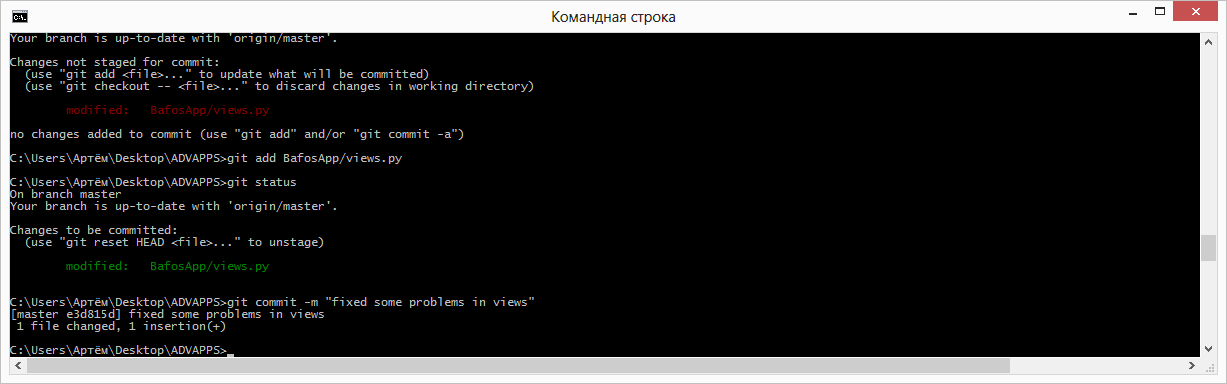


Рис.6 Процесс выполнения коммитов.

(Индексация файлов)

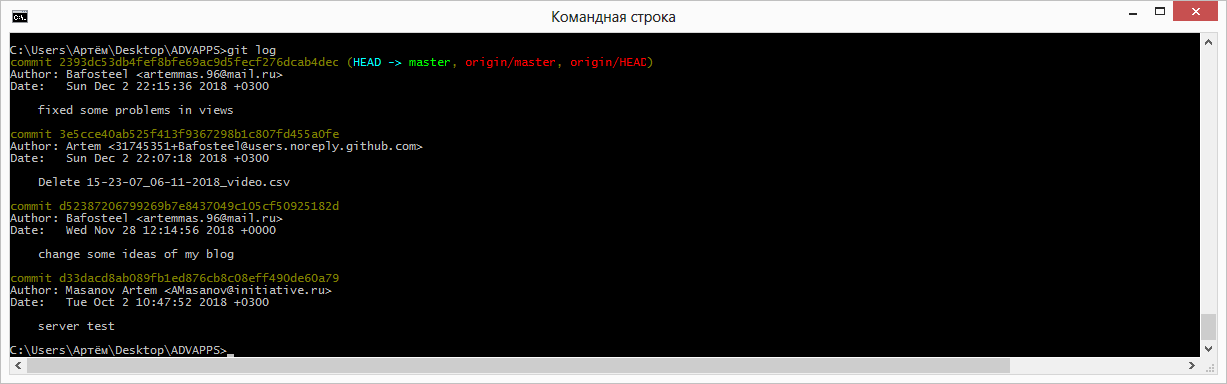


Рис.7 Просмотр созданных локальных и глобальных коммитов

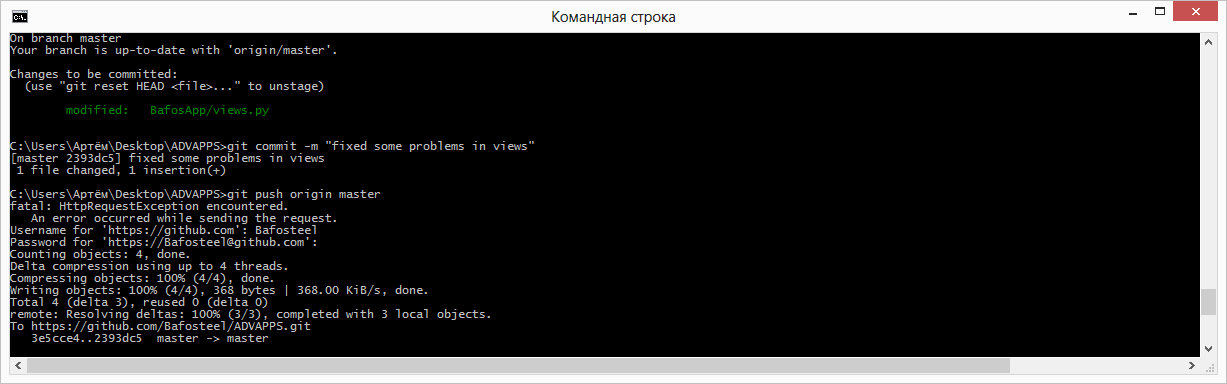


Рис.8 Процесс отправки в репозиторий ранее созданных локальных коммитов

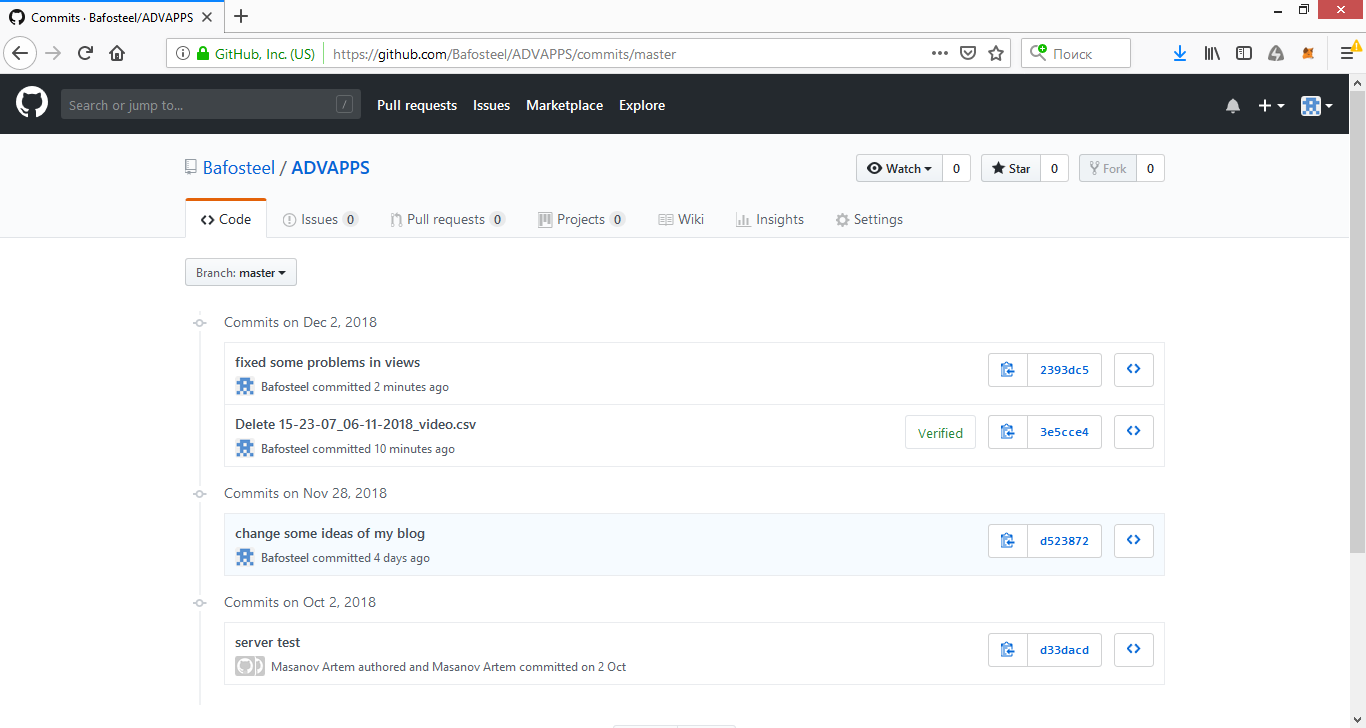


Рис.9 Просмотр созданных коммитов в репозитории.

1. **Создание веток и выполнение ряда изменений в проекте в созданных ветках**

Для создания ветки в репозитории можно воспользоваться через Git Bash следующими командами:

* $ git branch – Создание ветки
* $ git checkout - Переход на коммит

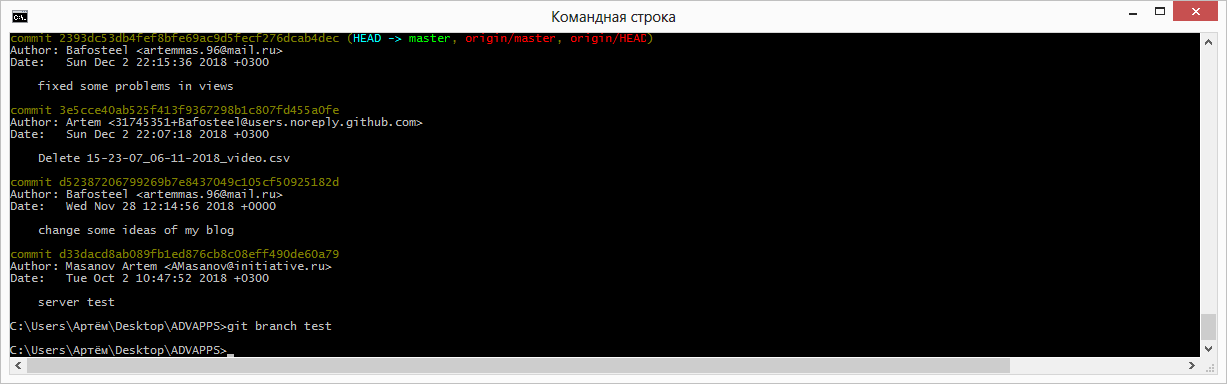
****

Рис.10 Процесс создания ветки.

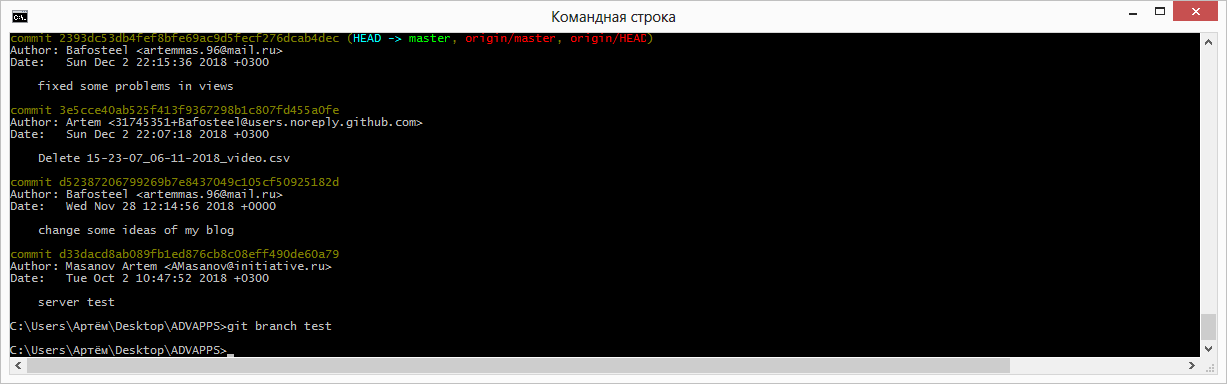
****

Рис.11 Процесс перехода на созданную ветку.

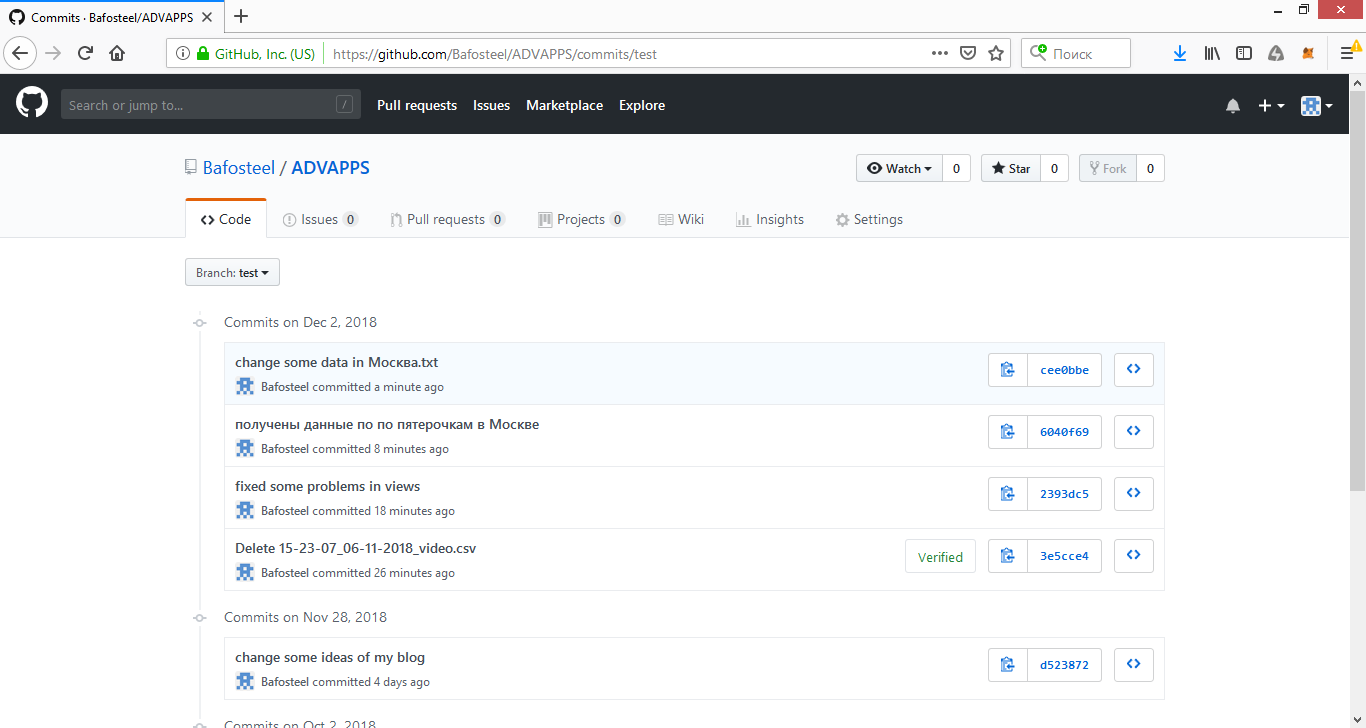


Рис.12 Просмотр коммитов в ветке test

1. **Слияние ветки с веткой мастер без конфликтов**

Для создания ветки можно воспользоваться через Git Bash следующими командами:

* $ git merge - Слияние текущей ветки с выбранной

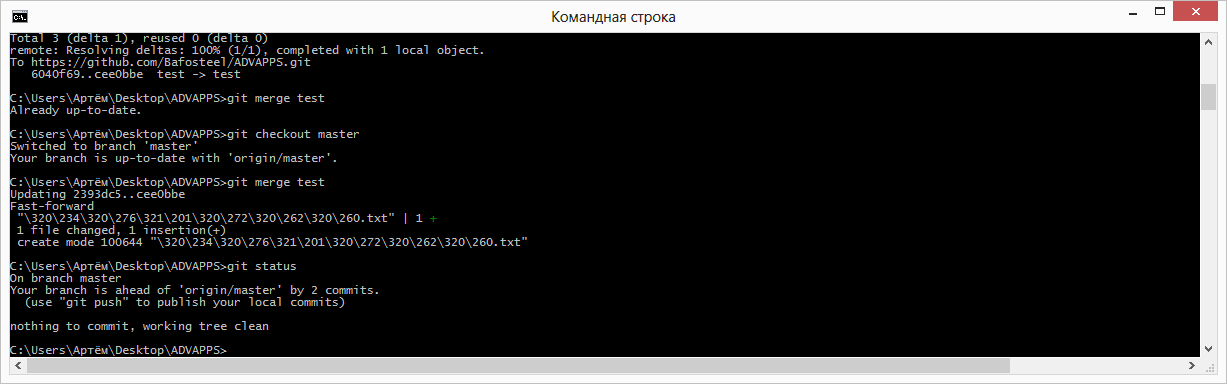
****

Рис.13 Процесс слияния веток без конфликтов.

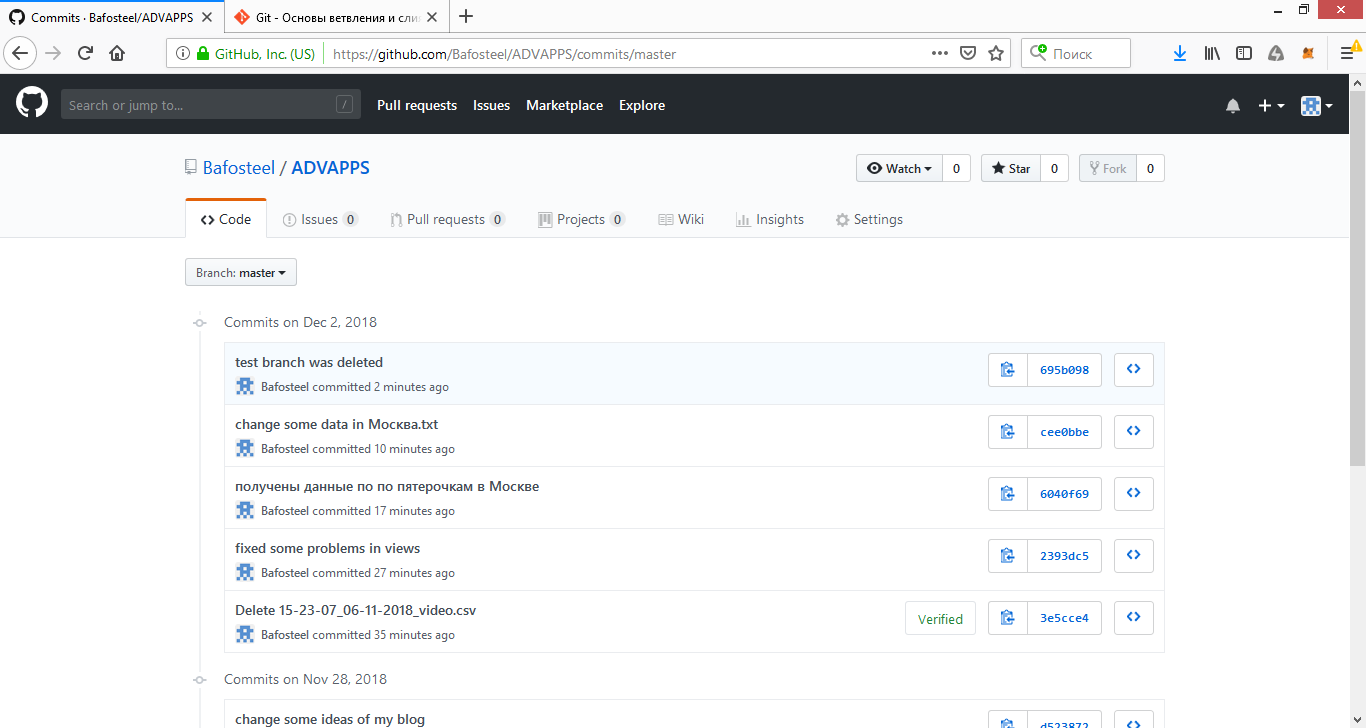


Рис.14 Просмотр коммитов во всех созданных ветках.

1. **Слияние ветки с веткой мастер с решением возникающих конфликтов**

Решение конфликтов слияния можно через Git Bash следующими командами:

* $ git mergetool - запустит соответствующий графический инструмент и покажет конфликтные ситуации
* $ git add - Индексация изменений файла

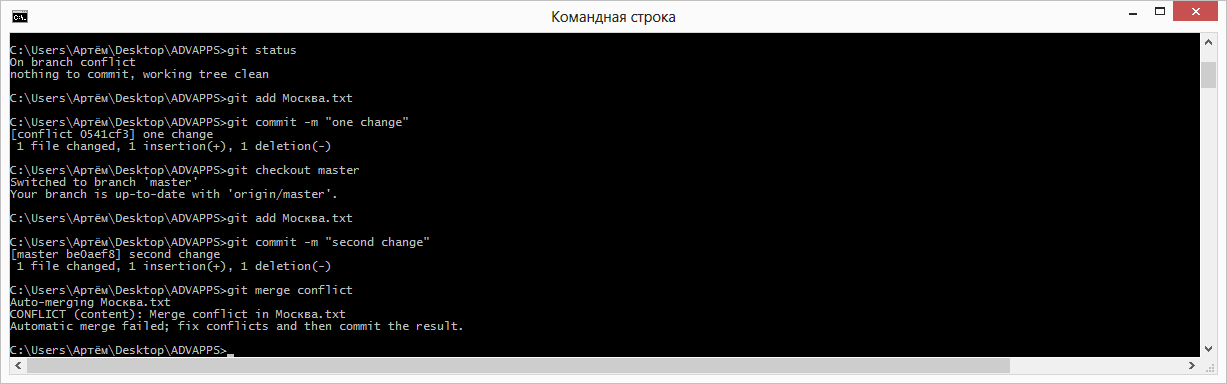


Рис.15 Сообщение о конфликте слияния веток.

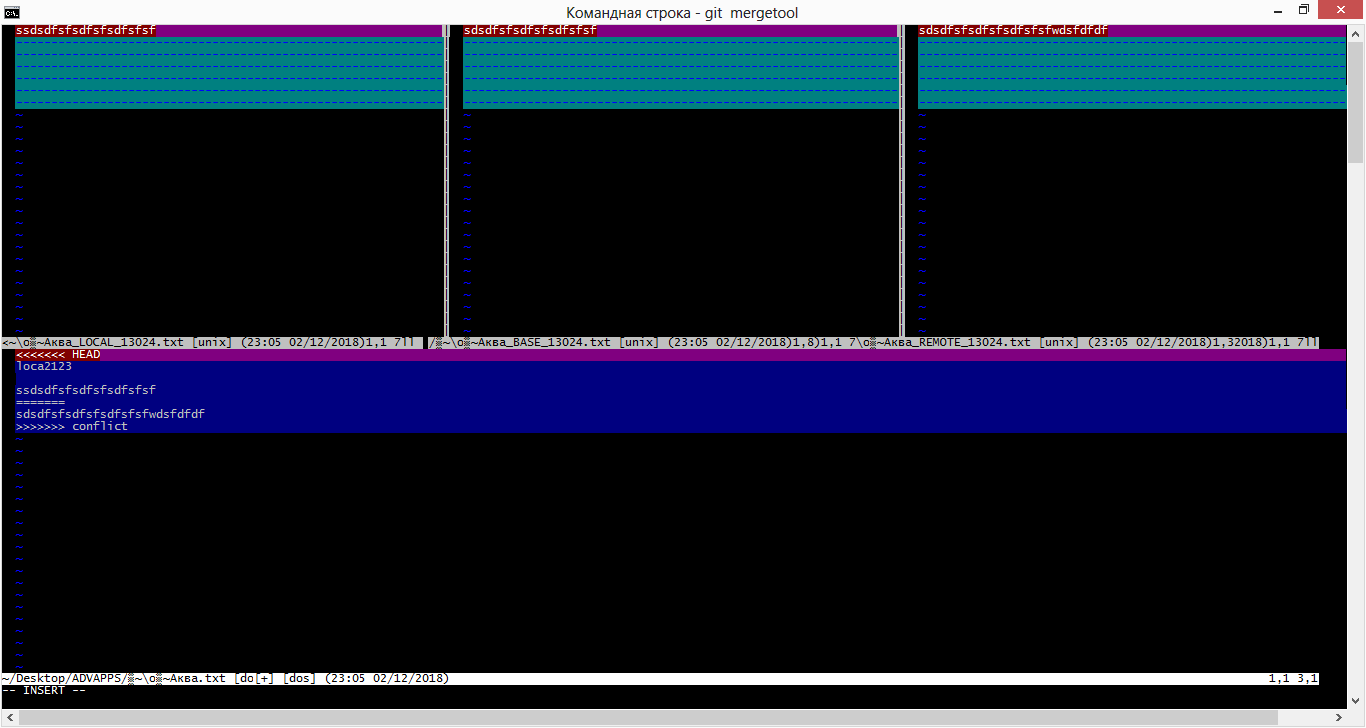


Рис.16 Процесс разрешения конфликта слияния веток

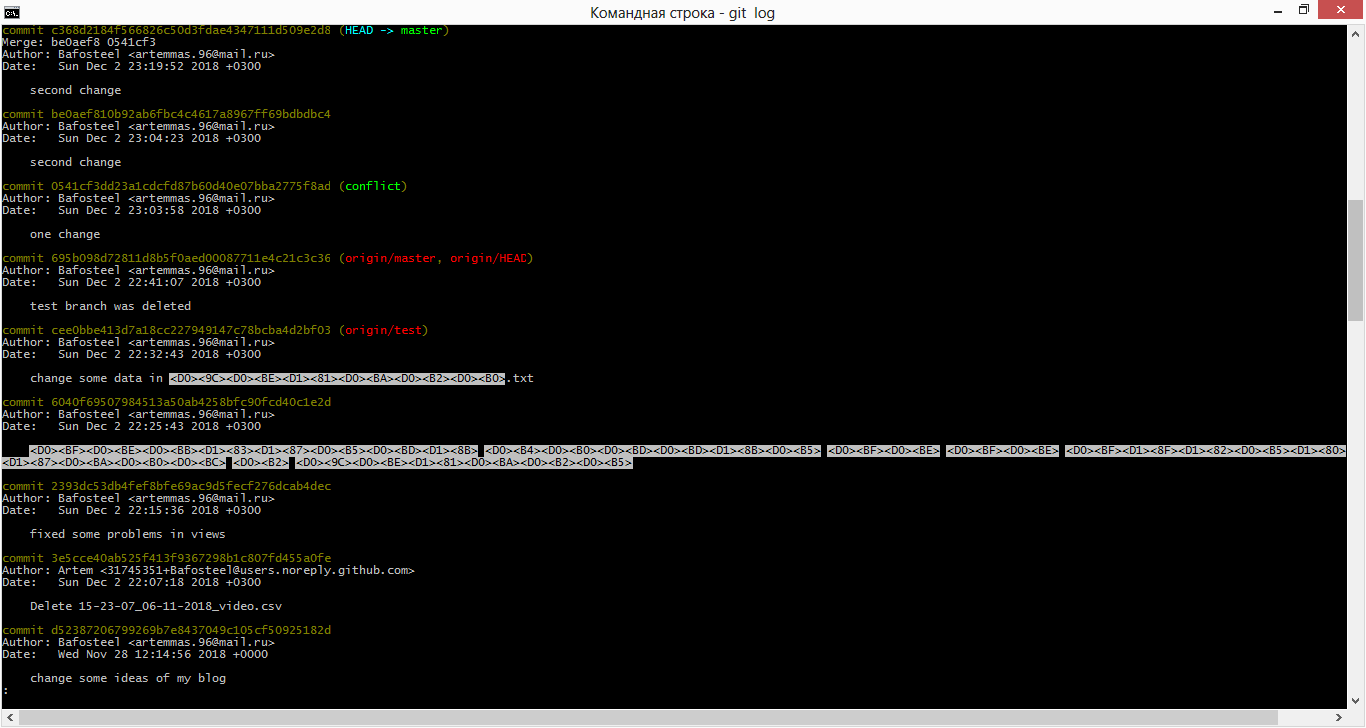


Рис.17 Просмотр коммитов во всех созданных ветках

1. **Откат изменений до одного из предыдущих коммитов**

Для отката изменений до одного из предыдущих коммитов в репозитории достаточно кликнуть на нужном коммите.

Так же можно решать конфликты слияния через Git Bash следующими командами:

* $ git checkout - Переход на коммит

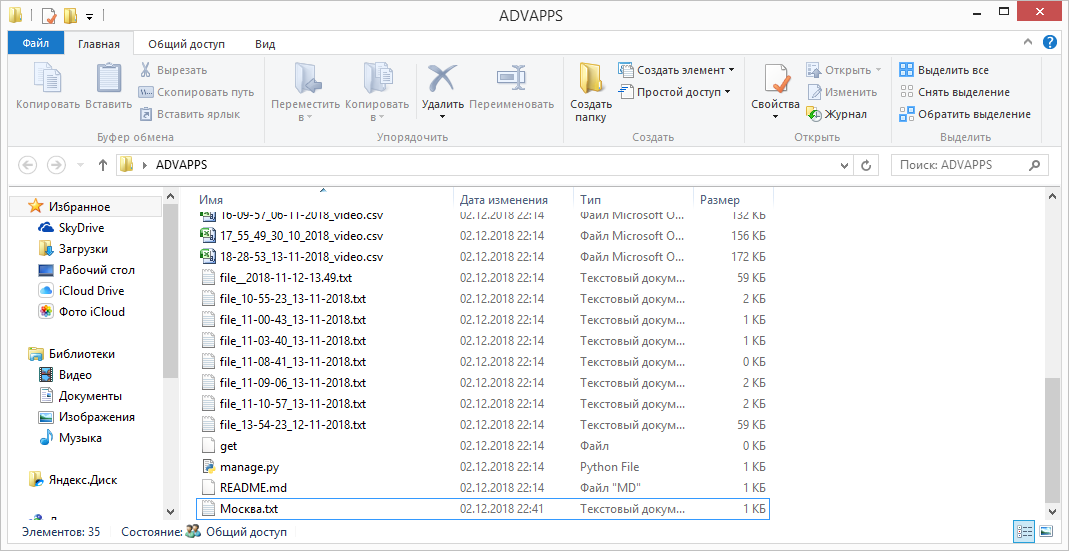
****

Рис.18 Содержание папки проекта до отката на более ранний коммит

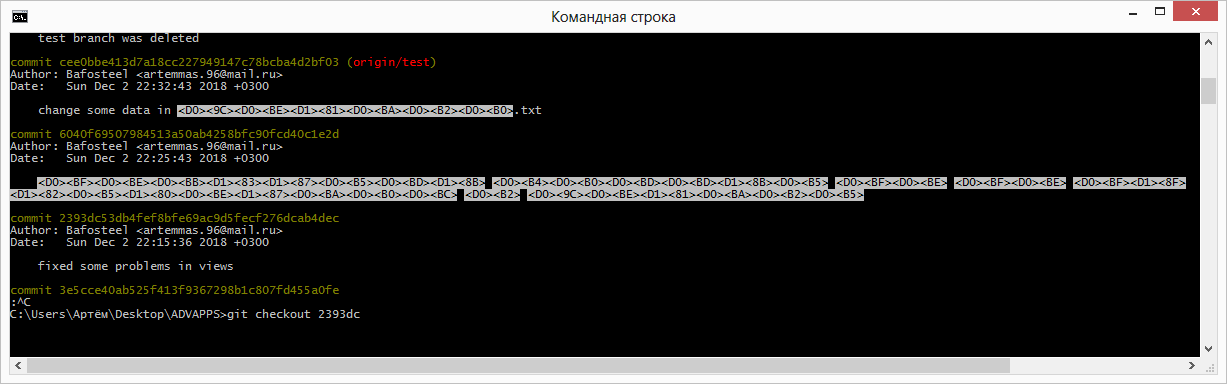
****

Рис.19 Процесс отката изменений до одного из предыдущих коммитов

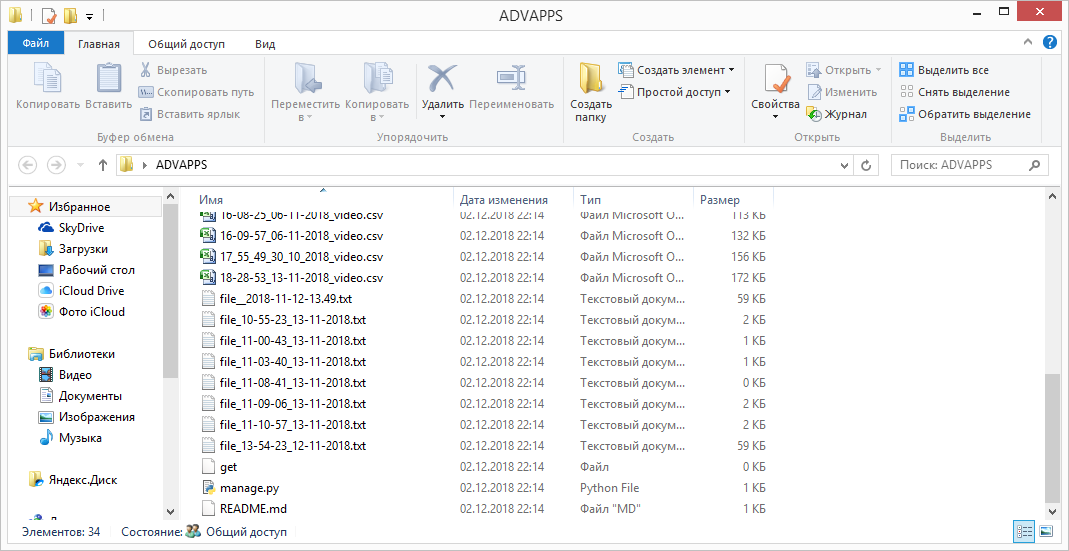
****

Рис.20 Содержание папки проекта после отката на более ранний коммит

**Выводы**

Система контроля версий полезный и важный инструмент для разработки программного обеспечения, в особенности, когда разработка ведётся в команде. Система контроля версий так же помогает отобразить всю историю проекта, а также спасает от случайно потерянных фрагментов кода и файлов.

**Литература**

Рекомендуемая литература по курсу ТРПО:

1. Камаев В.А., Костерин В.В. Технологии программирования. М.: Высшая школа, 2006.
2. Жоголев Е.А.Технология программирования. – М.: Научный мир, 2004.

Git:

1. Scott Chacon, Ben Straub «Pro Git»
2. [git-scm.com](https://git-scm.com/)