



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)

НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.03.01 Информатика и вычислительная техника

ОТЧЕТ

по лабораторной работе № 6

Название: Система контроля версий GIT

Дисциплина: Языки интернет программирования

Студент

ИУ6-64Б
(Группа)

(Подпись, дата)

Д. О. Романов
(И.О. Фамилия)

Преподаватель

(Подпись, дата)

В. И. Кузнецов
(И.О. Фамилия)

Москва, 2026

1. ЦЕЛЬ РАБОТЫ

Цель работы - изучение основ работы с распределенной системой контроля версий Git, получение практических навыков инициализации репозитория, фиксации изменений, работы с ветвлением и взаимодействия с удаленными репозиториями на платформе GitHub, включая создание запросов на слияние.

2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

2.1. Назначение Git

Git - распределенная система контроля версий, позволяющая отслеживать историю изменений в файлах и координировать работу нескольких разработчиков над одним проектом. В отличие от централизованных систем, в Git каждый разработчик имеет полную копию репозитория на своем локальном компьютере.

2.2. Основные понятия

- Репозиторий: хранилище файлов проекта и истории их изменений.
- Коммит (Commit): фиксация состояния файлов в определенный момент времени. Каждый коммит имеет уникальный хеш.
- Ветка (Branch): независимая линия разработки. Основная ветка обычно называется main или master.
- Удаленный репозиторий (Remote): версия проекта, хранящаяся на сервере (например, github/gitlab) для обмена изменениями.

2.3. Основные команды

- git init: создание нового репозитория.
- git add: добавление файлов в индекс для подготовки к коммиту.
- git commit: создание снимка изменений.
- git push: отправка локальных изменений в удаленный репозиторий.
- git pull: получение и слияние изменений из удаленного репозитория.
- git checkout: переключение между ветками или восстановление файлов.

3. ХОД ВЫПОЛНЕНИЯ РАБОТЫ

3.1. Создание репозитория и первый коммит

Был создан новый пустой репозиторий на GitHub с названием web-bmstu. В локальной папке проекта выполнена инициализация Git. Файлы добавлены в

индекс и зафиксированы первым коммитом. Выполнена привязка к удаленному репозиторию origin и отправка ветки main на сервер.

```
(base) dmitryromanov@Mac web % git init
Initialized empty Git repository in /Users/dmitryromanov/Desktop/web/.git/
(base) dmitryromanov@Mac web % git add .
(base) dmitryromanov@Mac web % git commit -m "commit 1: load directories"
[main (root-commit) 558c0f2] commit 1: load directories
 20 files changed, 509 insertions(+)
 create mode 100644 .DS_Store
 create mode 100644 web lb1/.DS_Store
 create mode 100644 web lb1/LW1 Romanov IU6-64.pdf
 create mode 100644 web lb1/base.py
 create mode 100644 "web lb1/\320\273\320\261 1.pdf"
 create mode 100644 web lb2/index.html
 create mode 100644 web lb2/test.html
 create mode 100644 "web lb2/\320\273\320\2612.pdf"
 create mode 100644 "web lb3/\320\273\320\2613.pdf"
 create mode 100644 "web lb4/\320\273\320\2614.pdf"
 create mode 100644 "web lb5/\320\273\320\2615.pdf"
 create mode 100644 web lb7/lab.py
 create mode 100644 web lb7/sopranos.txt
 create mode 100644 "web lb7/\320\273\320\2616.pdf"
 create mode 100644 "web lb7/\320\273\320\2617.pdf"
 create mode 100644 "\320\273\320\261 11 \321\207\320\265\321\200\320\275\320\276\320\262\320\270\320\272.pdf"
 create mode 100644 "\320\273\320\2618.pdf"
 create mode 100644 "\320\273\320\2619.pdf"
 create mode 100644 "\321\202\320\270\321\202\321\203\320\273.docx"
(base) dmitryromanov@Mac web % git remote add origin https://github.com/dmtrx/web-bmstu.git
(base) dmitryromanov@Mac web % git push -u origin main
Enumerating objects: 28, done.
Counting objects: 100% (28/28), done.
Delta compression using up to 10 threads
Compressing objects: 100% (25/25), done.
Writing objects: 100% (28/28), 9.35 MiB | 21.95 MiB/s, done.
Total 28 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), done.
To https://github.com/dmtrx/web-bmstu.git
 * [new branch]    main -> main
branch 'main' set up to track 'origin/main'.
```

Рисунок 1 - Инициализация репозитория и первый коммит

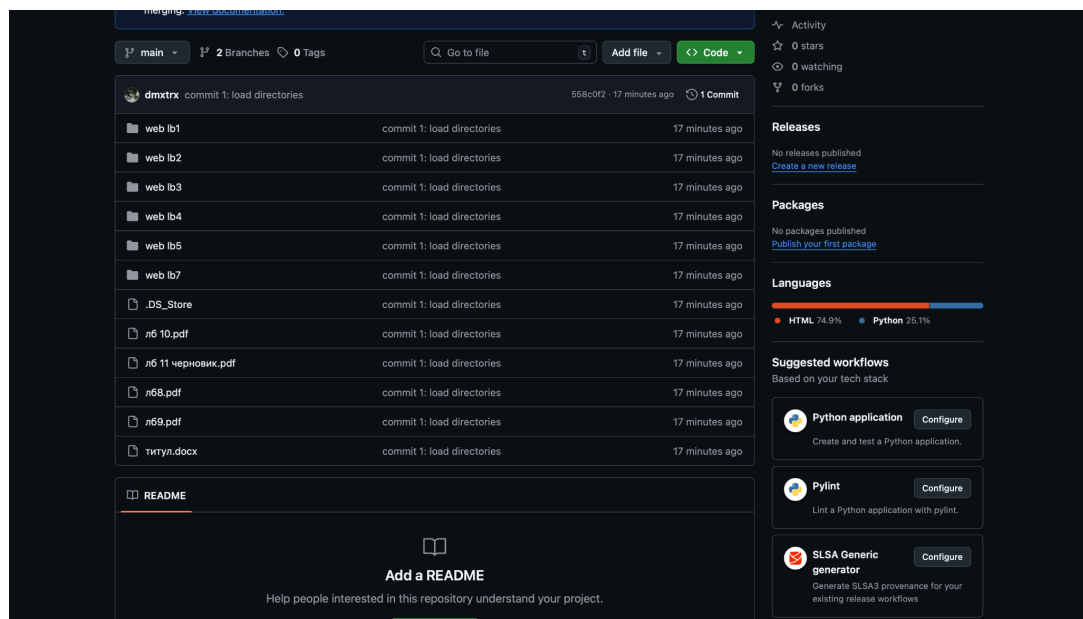


Рисунок 2 - Отображение файлов в репозитории GitHub

3.2. Работа с ветвлением

Для выполнения задания по добавлению нового функционала была создана новая ветка feature. Сделано два коммита: первый с добавлением файла, второй - с внесением изменений. Изменения отправлены в удаленный репозиторий в новую ветку.

```
(base) dmitryromanov@Mac web % git checkout -b feature
Switched to a new branch 'feature'
[(base) dmitryromanov@Mac web % git commit -am "update Sopranos"
[feature 1f1c517] update Sopranos
 2 files changed, 1 insertion(+), 1 deletion(-)
(base) dmitryromanov@Mac web % git add .
(base) dmitryromanov@Mac web % git commit -m "add empty gitignore"
[[feature de678e2] add empty gitignore
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 .gitignore
(base) dmitryromanov@Mac web % git push origin feature
Enumerating objects: 12, done.
Counting objects: 100% (12/12), done.
Delta compression using up to 10 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (8/8), 1.43 KiB | 1.43 MiB/s, done.
Total 8 (delta 5), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (5/5), completed with 4 local objects.
remote:
remote: Create a pull request for 'feature' on GitHub by visiting:
remote:   https://github.com/dmxtxr/web-bmstu/pull/new/feature
remote:
To https://github.com/dmxtxr/web-bmstu.git
 * [new branch]      feature -> feature
(base) dmitryromanov@Mac web %
```

Рисунок 3 - Работа в ветке feature

3.3. Настройка коллабораторов

В настройках репозитория на GitHub в разделе Collaborators было отправлено приглашение пользователю Emetless для предоставления доступа к репозиторию.

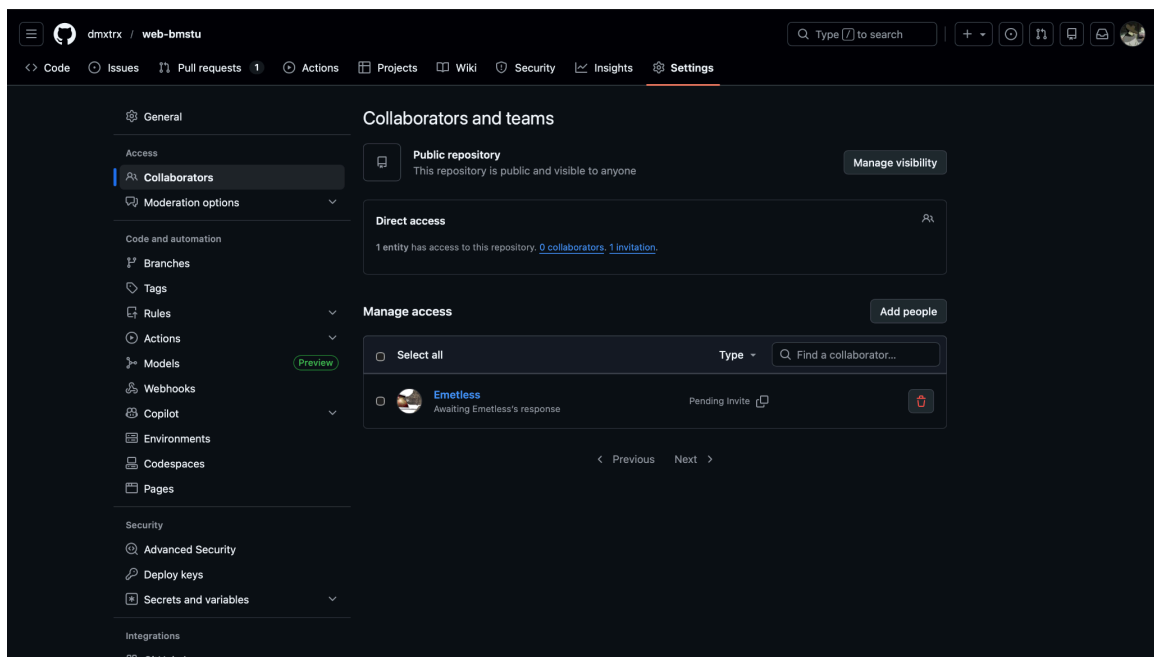


Рисунок 4 - Добавление коллаборатора

3.4. Создание Pull Request

На платформе GitHub был оформлен запрос на слияние изменений из ветки feature/ в ветку main. В заголовке указана группа и ФИО студента. В качестве ревьюера назначен пользователь Emetless.

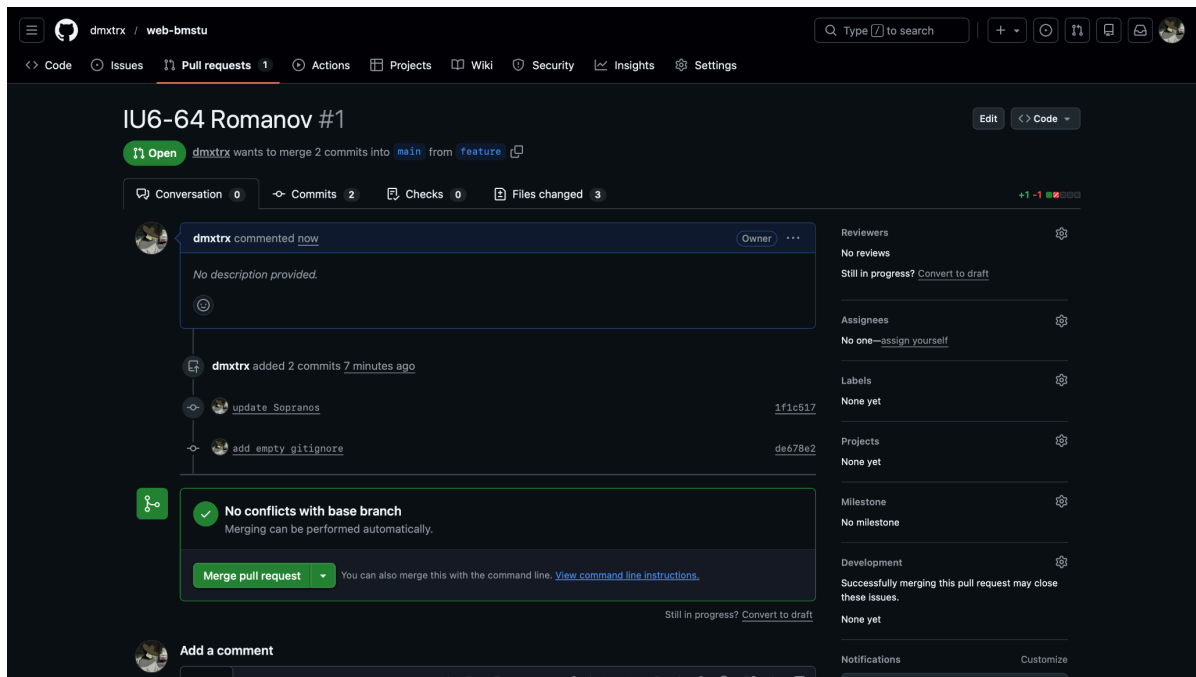


Рисунок 5 - Созданный Pull Request

4. ВЫВОДЫ

В ходе лабораторной работы были освоены базовые принципы работы с системой контроля версий Git. Научились инициализировать локальные репозитории, фиксировать изменения, создавать ветки для разработки новых функций и синхронизировать локальную работу с удаленным сервисом GitHub. Также изучен процесс командной разработки через механизм Pull Request и Code Review.

5. ОТВЕТЫ НА КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое Git-репозиторий и чем локальный репозиторий отличается от удалённого?

Git-репозиторий - это хранилище, в котором Git отслеживает и управляет файлами проекта, включая полную историю всех изменений. Локальный репозиторий находится на компьютере разработчика и содержит рабочую директорию, индекс и историю коммитов. Удалённый репозиторий находится на сервере и используется для синхронизации работы команды. Он обычно является "чистым" и не содержит рабочей директории для редактирования файлов.

2. Объясните разницу между рабочей директорией, индексом и коммитом. Каков типичный цикл команд для фиксации изменений?

- Рабочая директория: папка с файлами проекта, где вы непосредственно редактируете код.
- Индекс: промежуточная зона, куда добавляются файлы, готовые к включению в следующий коммит.
- Коммит: зафиксированный снимок состояния проекта, сохраненный в истории Git.

Типичный цикл команд:

1. Изменение файлов в рабочей директории.
2. ``git add <файл>`` - добавление изменений в индекс.
3. ``git commit -m "Сообщение"`` - создание коммита из содержимого индекса.

3. Какие команды используют, чтобы: а) увидеть текущие изменения, б) посмотреть историю коммитов? В чём разница между `git status` и `git log`?

а) Увидеть текущие изменения: ``git status`` - показывает, какие файлы изменены, добавлены в индекс или не отслеживаются, ``git diff`` -показывает конкретные изменения в содержимом файлов.

б) Посмотреть историю: ``git log`` - выводит список коммитов с хешами, авторами и датами.

Разница:

- ``git status`` показывает *текущее состояние* рабочей директории и индекса относительно последнего коммита.
- ``git log`` показывает *историю* уже сделанных коммитов.

4. Как создать новую ветку и сразу переключиться на неё? Написать одну или две возможные команды. Самый быстрый способ:

``git checkout -b <branch>``

В новых версиях Git также доступна команда:

``git switch -c <branch>``

Альтернативный вариант в две команды:

1. ``git branch <branch>`` (создать)
2. ``git checkout <branch>`` (переключиться)

5. Чем отличаются `git fetch`, `git pull` и `git push`? В каких ситуациях применяется каждая команда?

- ``git fetch``: скачивает изменения из удаленного репозитория, но **не сливает** их с текущей работой. Используется, чтобы посмотреть, что нового появилось на сервере, не меняя свои файлы.
- ``git pull``: делает ``git fetch``, а затем сразу пытается слить полученные изменения в текущую ветку. Используется для обновления своего локального кода до актуальной версии с сервера.
- ``git push``: отправляет ваши локальные коммиты на удаленный сервер. Используется, чтобы поделиться своей работой с командой.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Chacon S., Straub B. Pro Git. – Apress, 2014. – URL: <https://git-scm.com/book/ru/v2>
2. Кузнецов В.И. Система контроля версий GIT: методические указания к выполнению лабораторной работы №6. – М.: МГТУ им. Н.Э. Баумана, 2025.
3. Официальная документация GitHub. – URL: <https://docs.github.com/ru>
4. Loeliger J., McCullough M. Version Control with Git. – O'Reilly Media, 2012.