**ЛАБОРАТОРНАЯ РАБОТА №3**

**Курс «Технологии разработки программного обеспечения»**

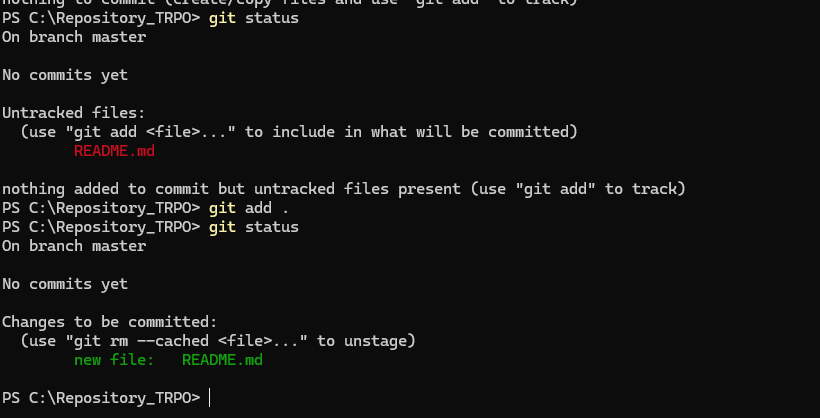
**Тема**: Системы контроля версий.

**Цель:** Приобрести навыки работы с системами контроля версий на примере программ TortoiseSVN и git.

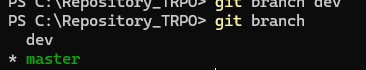
**Задание:**

**Ссылка на репозиторий: https://github.com/Amudz1/TRPO**

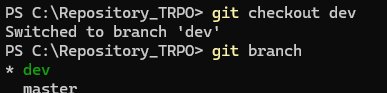
Статус репозитория:



Создаем ветку dev:



Переходим в ветку dev:



Так же можно посмотреть какие были изменения до коммита в репозитории, или же в определенном файле

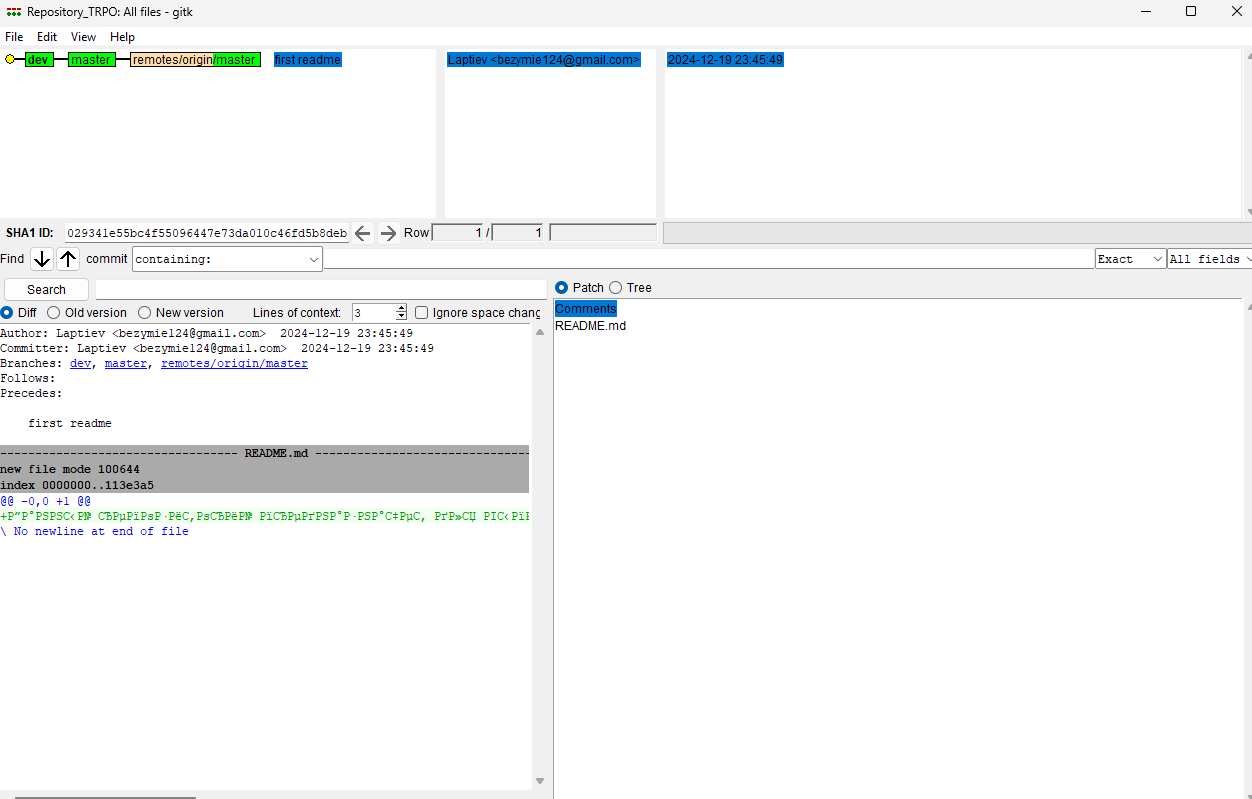


Либо же отменить изменения коммитов



Открываем  графический браузер репозитория gitk**:**

  
Открывается данное окно:



**Контрольные вопросы:**

1. **Основные отличия централизованных и распределённых систем контроля версий:**

Основные отличия централизованных и распределённых систем контроля версий заключаются в подходе к хранению данных и организации работы. В централизованных системах, таких как Subversion (SVN), все данные хранятся на одном центральном сервере. Пользователи подключаются к серверу для выполнения любых операций, таких как просмотр истории изменений или получение обновлений. Если сервер становится недоступен, работа с репозиторием останавливается. Этот подход проще для понимания и настройки, но уязвим к сбоям сервера и требует постоянного подключения к сети. В распределённых системах, таких как Git, каждая копия репозитория на локальном компьютере содержит полную историю всех изменений. Это позволяет выполнять большинство операций, таких как создание коммитов или анализ истории, без подключения к сети. Распределённый подход обеспечивает высокую надёжность (так как данные дублируются) и большую скорость локальных операций, но может быть сложнее для понимания новичками.

**2. Модели совместного использования файлов системами контроля версий:**

1. **Lock-Modify-Unlock (Заблокировать-Изменить-Разблокировать)**:
   * Пользователь блокирует файл перед редактированием.
   * После завершения изменений файл разблокируется.
   * Применяется для бинарных файлов.
   * Пример: используется в Subversion (SVN).
2. **Copy-Modify-Merge (Копировать-Изменить-Слить)**:
   * Пользователи получают копии файлов и работают параллельно.
   * После завершения изменений выполняется слияние.
   * Подходит для текстовых файлов.
   * Пример: используется в Git и SVN.

**3. Модель TortoiseSVN по умолчанию и её преимущества:**

* **Модель**: *Copy-Modify-Merge (Копировать-Изменить-Слить)*.
* **Преимущества**:
  1. Позволяет нескольким разработчикам работать одновременно над одним файлом.
  2. Уменьшает количество простоев, связанных с блокировкой.
  3. Более гибкая для работы с текстовыми файлами и кодом.

**4. Отличие команд «Обновить» и «Фиксировать». Что делает команда «Слить»?**

* **Обновить (Update)**:
  + Получение последних изменений из центрального/удалённого репозитория в локальную копию.
* **Фиксировать (Commit)**:
  + Отправка локальных изменений в центральный/удалённый репозиторий.
* **Слить (Merge)**:
  + Объединяет изменения из двух веток или из удалённого репозитория в текущую ветку.
  + Используется для интеграции изменений, сделанных параллельно.

**5. Этапы стандартного workflow в Git и команды:**

1. **Клонирование репозитория**:
   * Команда: git clone <URL>
   * Создаёт локальную копию удалённого репозитория.
2. **Создание ветки**:
   * Команда: git branch <branch\_name>
   * Создаёт новую ветку для изолированной работы.
3. **Переключение между ветками**:
   * Команда: git checkout <branch\_name>
   * Переключает текущую ветку.
4. **Добавление изменений в индекс**:
   * Команда: git add <file\_name> или git add .
   * Подготавливает изменения для фиксации.
5. **Просмотр статуса изменений**:
   * Команда: git status
   * Показывает состояние репозитория (изменения, готовые к фиксации).
6. **Сравнение изменений**:
   * Команда: git diff
   * Показывает разницу между текущими и предыдущими версиями.
7. **Фиксация изменений**:
   * Команда: git commit -m "<message>"
   * Сохраняет изменения в локальный репозиторий.
8. **Обновление репозитория**:
   * Команда: git pull
   * Получает изменения из удалённого репозитория и сливает их с текущей веткой.
9. **Отправка изменений на сервер**:
   * Команда: git push
   * Отправляет локальные изменения в удалённый репозиторий.
10. **Сброс изменений**:
    * Команда: git reset <commit>
    * Возвращает состояние репозитория к указанному коммиту.

**Рекомендуемые источники:**

1. TortoiseSVN – Инструкция по применению [Электронный ресурс] / URL: <http://tortoisesvn.net/docs/nightly/TortoiseSVN_en/index.html>.

2. Документация по git [Электронный ресурс] / URL: <http://git-scm.com/book/ru/v1>.

3. Брауде Э. Технология разработки программного обеспечения. – СПб.: Питер, 2004. – 655с., ил.