МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение высшего образования

«Российский экономический университет имени Г.В. Плеханова»

**Московский приборостроительный техникум**

**Практическая работа №2**

**На тему**: Знакомство с системой контроля версий Git.

**по специальности**: 09.02.06 Сетевое и системное администрирование.

**По дисциплине**: Сетевые языки.

**Выполнил**:

Марус Э.В.

**Студент группы**: CA50-4-22

**Проверил**:

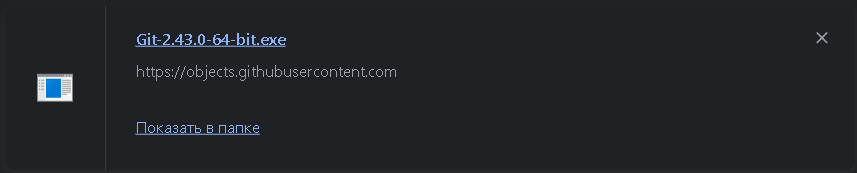
Кочарян Э.Р.

преподаватель ФГБОУ ВО РЭУ им. Г.В. Плеханова;

Москва, 2024г

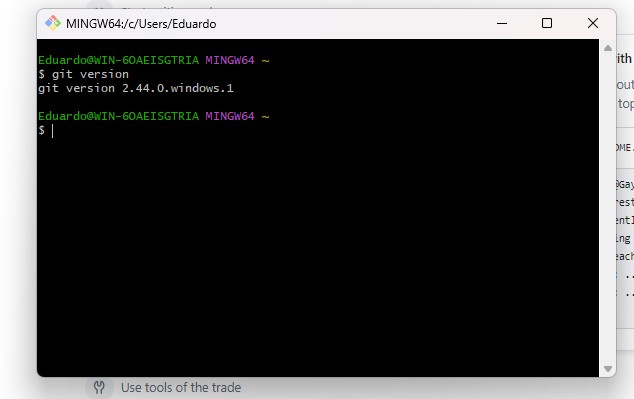
**Ход работы.**

1) Установка командной строки git bash.



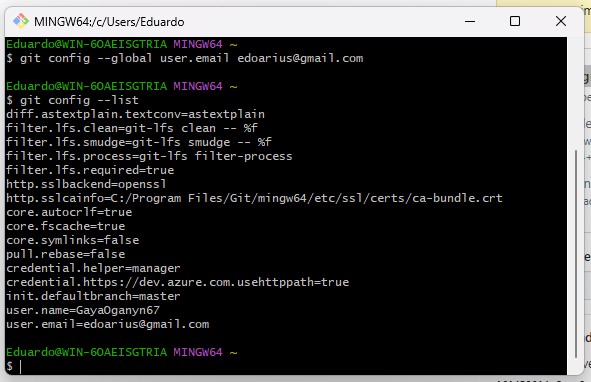
*Рис.1 Установка git*

2) Запуск терминала git.

**

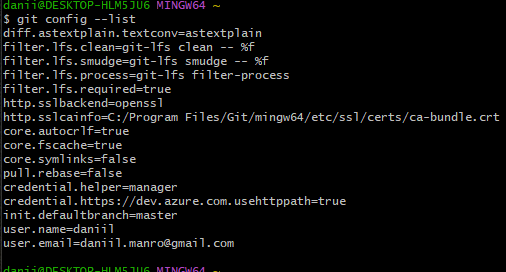
*Рис.2 Информация о версии программы git*

3) Произведена настройка git командами.

git config --list

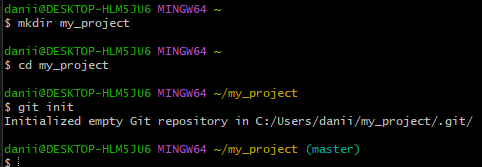
*Рис.3 Настройка git командами*

4) Проверка сохранения данных.

mkdir

*Рис.4 Проверка*

5) Создайте новую директорию для проекта и инициализируйте в ней Git- репозиторий.



*Рис.5 Создание новой директории и ее инициализация*

6) Создан файл README.md. Добавлен в индекс Git с помощью git add. Сделан коммит с комментарием об изменениях.



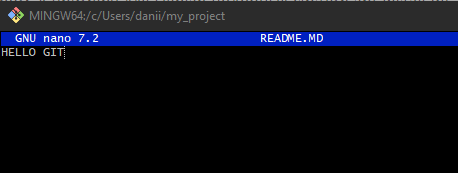
*Рис.6 Работа с файлом README.md*

7) Залейте локальный репозиторий в удаленный на Github.



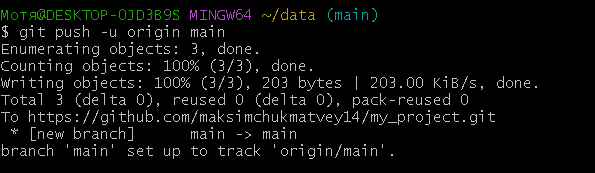
*Рис.7 Изменение репозитория*

8) Добавлен текст в файл README.md.

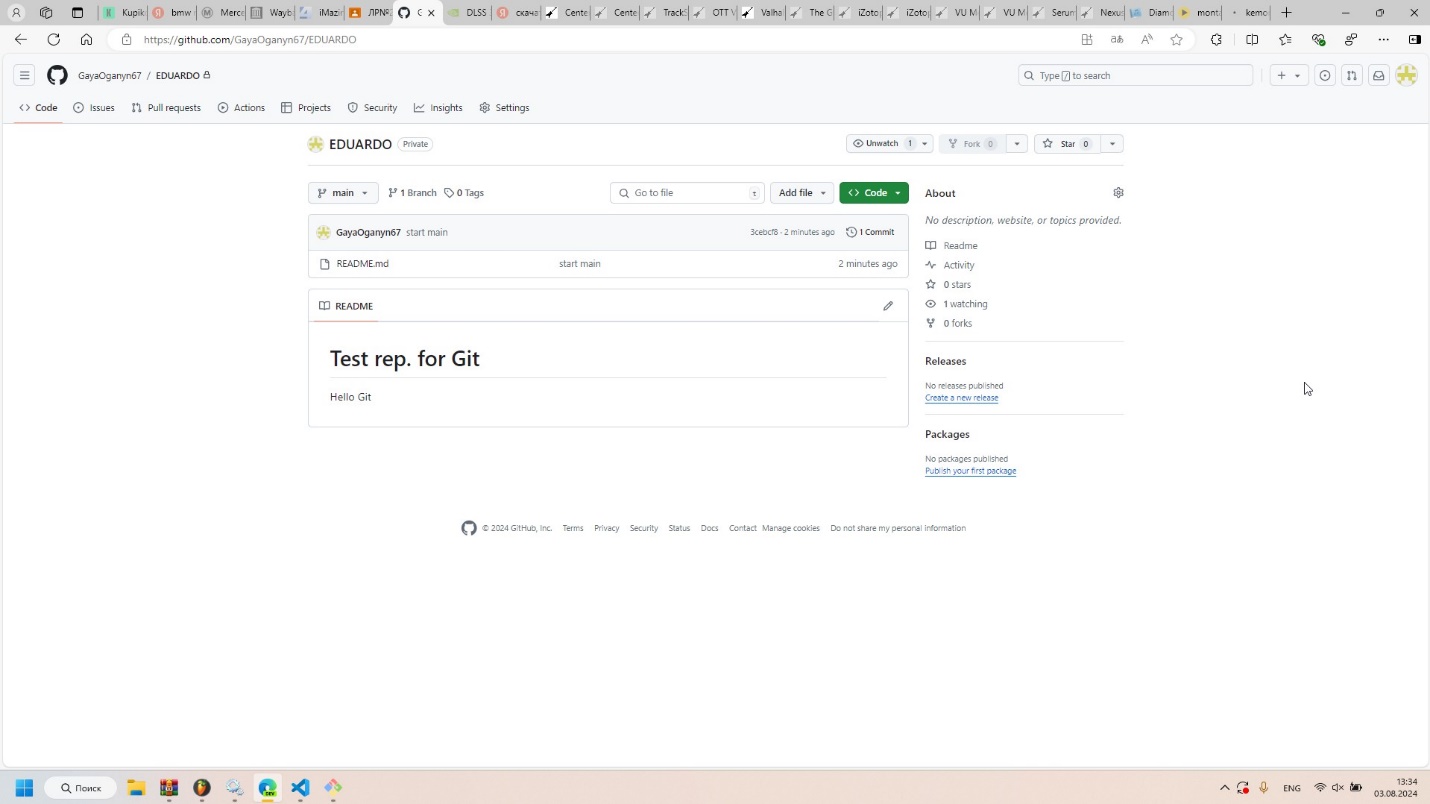


*Рис.8 Добавление текста*

9) Изменения залиты в удаленный репозиторий.



*Рис.9 Изменения залиты в удаленный репозиторий*

**

*Рис.10 Проверка*

**Контрольные вопросы.**

1) Система контроля версий (СКВ) позволяет отслеживать изменения в коде, документах и других файлах, сохраняя их историю и обеспечивая возможность возвращения к предыдущим версиям. Преимущества использования СКВ включают:

1. Организацию совместной работы: разработчики могут одновременно работать над одним проектом, не боясь потерять изменения других участников.

2. Возможность отката к предыдущим версиям: при возникновении ошибок можно легко вернуться к стабильной версии проекта.

3. Отслеживание изменений: система фиксирует, кто, когда и что изменил в проекте, что полезно для анализа и устранения проблем.

4. Резервное копирование: СКВ помогает сохранить целостность данных и избежать потери информации.

5. Ветвление и слияние: разработчики могут создавать отдельные ветки для экспериментов, а затем объединять их с основной веткой.

6. Улучшение качества кода: возможность проводить ревью кода и отслеживать изменения способствует повышению качества программного продукта.

Эти преимущества делают системы контроля версий необходимыми для эффективной работы над проектами любого масштаба.

2) Git - это распределенная система контроля версий, разработанная Линусом Торвальдсом. Она позволяет отслеживать изменения в исходном коде и других файлах, сохраняя историю изменений и обеспечивая возможность совместной работы над проектами.

Основные особенности Git:

1. Распределенная структура: каждый разработчик имеет полную копию репозитория, что позволяет работать независимо от других участников и даже без подключения к сети.

2. Ветвление и слияние: Git обеспечивает удобные инструменты для создания отдельных веток разработки, экспериментов и функциональных изменений, а также для их последующего объединения.

3. Высокая производительность: Git был разработан с учетом скорости работы, что делает его эффективным при работе с большими проектами.

4. Гибкий набор команд: Git предоставляет широкий набор команд для работы с репозиторием, позволяя выполнять различные операции с файлами и историей изменений.

5. Поддержка нелинейной разработки: разработчики могут одновременно работать над различными частями проекта, создавая отдельные ветки для каждой задачи.

Git является одним из наиболее популярных инструментов для управления версиями кода и широко используется в индустрии разработки программного обеспечения.

3) 1. Создание репозитория:

- Локальный репозиторий: для создания локального репозитория нужно перейти в папку проекта и выполнить команду git init. Это создаст новый репозиторий Git в текущей директории.

- Удаленный репозиторий: для создания удаленного репозитория, например на GitHub или GitLab, нужно зайти на соответствующий сайт, зарегистрироваться, создать новый репозиторий и склонировать его на локальную машину с помощью команды git clone.

2. Добавление файлов:

- После создания репозитория нужно добавить файлы, которые вы хотите отслеживать. Для этого используется команда git add, например git add file.txt. Это подготовит файлы к сохранению изменений.

3. Сохранение изменений (commit):

- Когда вы внесли изменения в файлы и хотите сохранить их, используйте команду git commit. Например: git commit -m "Добавил новую функциональность". Это создаст новый коммит с вашими изменениями.

4. Работа с ветками:

- Для создания новой ветки используйте команду git branch, например git branch new-feature. Чтобы переключиться на другую ветку, используйте команду git checkout, например git checkout new-feature.

5. Получение изменений из удаленного репозитория:

- Чтобы получить изменения из удаленного репозитория, используйте команду git pull. Например: git pull origin master.

6. Отправка изменений в удаленный репозиторий:

- Чтобы отправить свои изменения в удаленный репозиторий, используйте команду git push. Например: git push origin master.

4) Для загрузки (заливки) локального репозитория в удаленный репозиторий, вы можете использовать команду git push. Вот как это делается:

1. Сначала убедитесь, что вы находитесь в локальной ветке, которую вы хотите отправить в удаленный репозиторий. Вы можете проверить текущую ветку с помощью команды git branch.

2. Затем используйте команду git remote -v, чтобы убедиться, что у вас есть ссылка на удаленный репозиторий, куда вы хотите загрузить свои изменения.

3. Если ссылка на удаленный репозиторий отсутствует, добавьте ее с помощью команды git remote add <название\_удаленного\_репозитория> <URL\_удаленного\_репозитория>.

4. Теперь, когда у вас есть ссылка на удаленный репозиторий, используйте команду git push <название\_удаленного\_репозитория> <название\_локальной\_ветки> для загрузки локальной ветки в удаленный репозиторий. Например: git push origin master.