Шаблон отчёта по лабораторной работе 4

Продвинутое использование git

Абдуллахи Бахара

}

# Цель работы

* Получение навыков правильной работы с репозиториями git.

# Задание

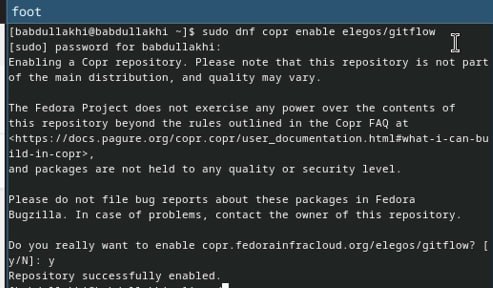
* Выполнить работу для тестового репозитория.
* Преобразовать рабочий репозиторий в репозиторий с git-flow и conventional commits.

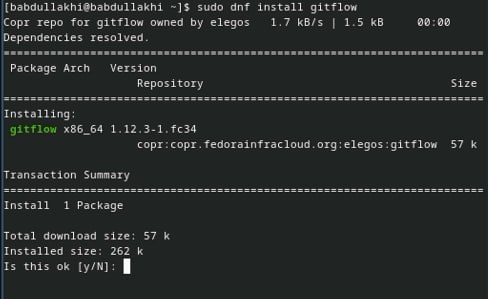
# Выполнение лабораторной работы:

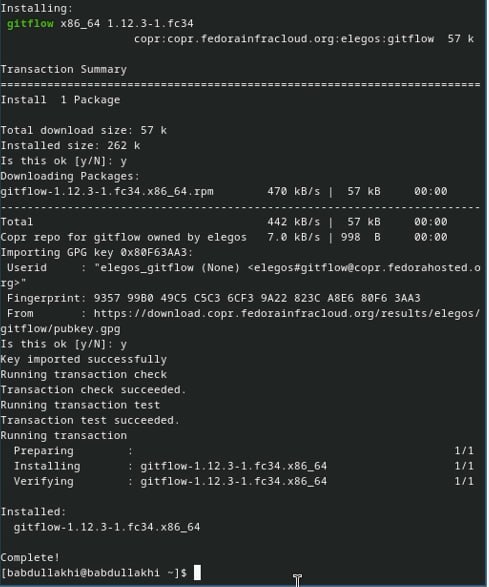
## Установка программного обеспечения

* Установка git-flow:
* Linux
* Fedora
* Установка из коллекции репозиториев Copr (https://copr.fedorainfracloud.org/coprs/elegos/gitflow/):

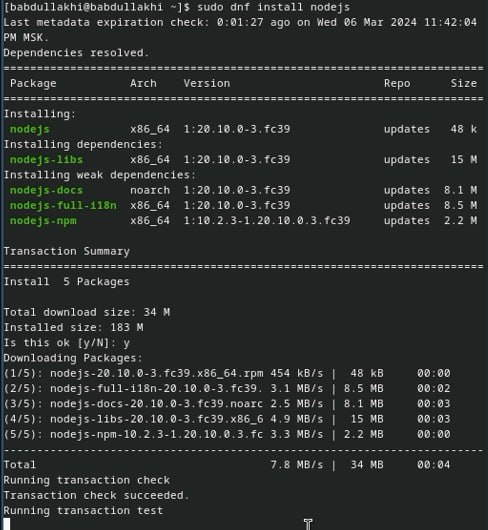
# Enable the copr repository dnf copr enable elegos/gitflow # Install gitflow dnf install gitflow



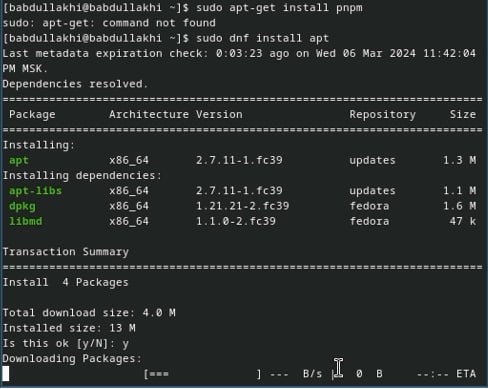


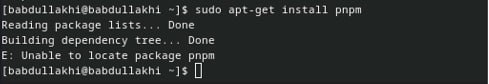


Установка Node.js :



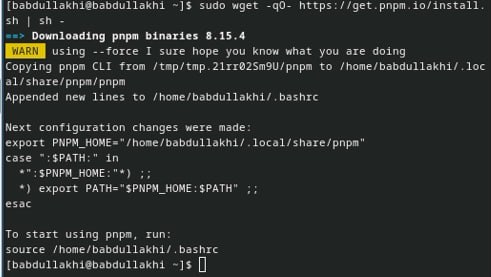
* Чтобы установить apt-get install pnpm, нужно установить apt с помощью этого камнда dnf install apt:



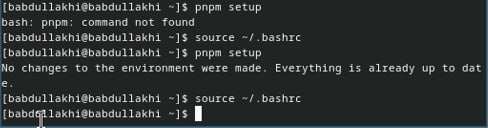


# Настройка Node.js:

* Для работы с Node.js добавим каталог с исполняемыми файлами, устанавливаемыми yarn, в переменную PATH.



* Запустите:
* Перелогиньтесь, или выполните:

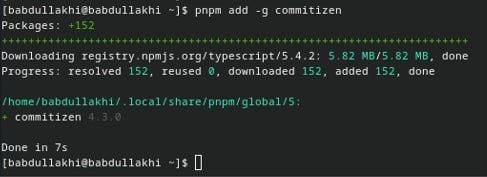


# Общепринятые коммиты:

* commitizen:
* Данная программа используется для помощи в форматировании коммитов.



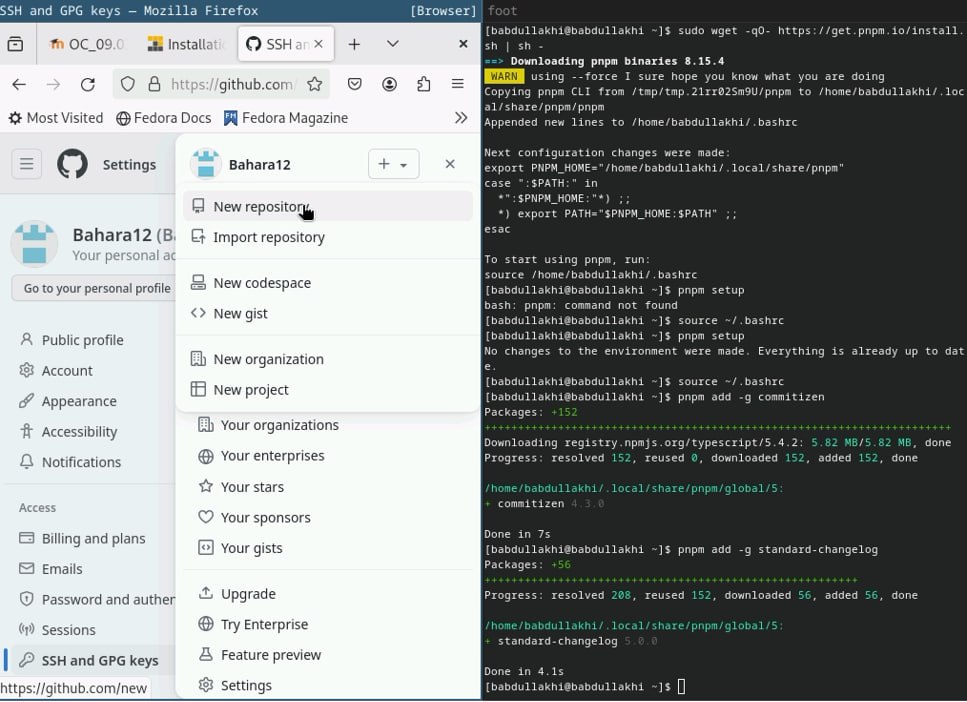
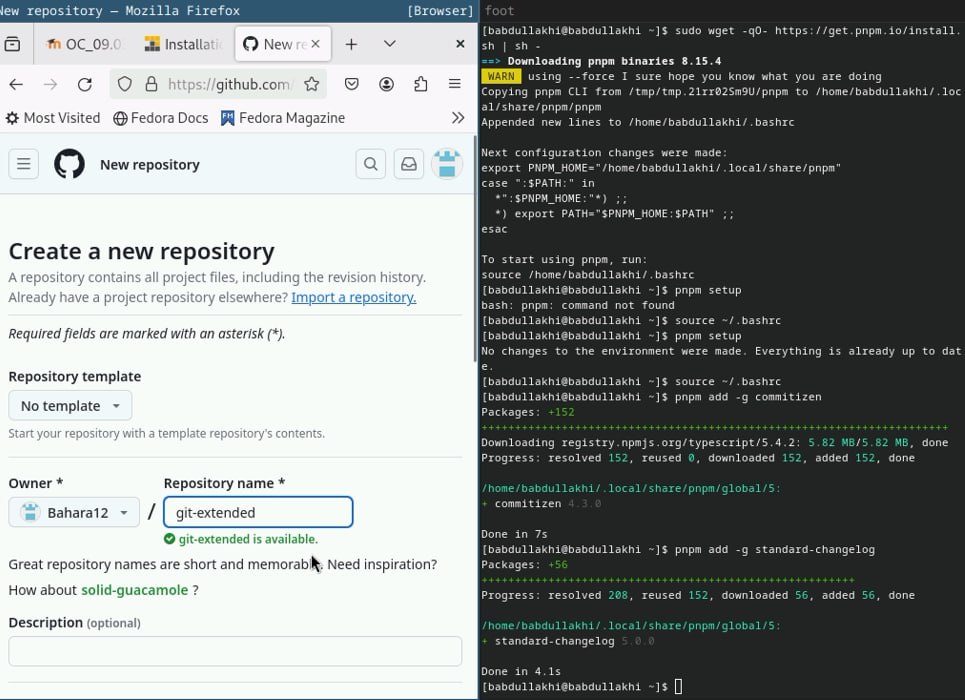
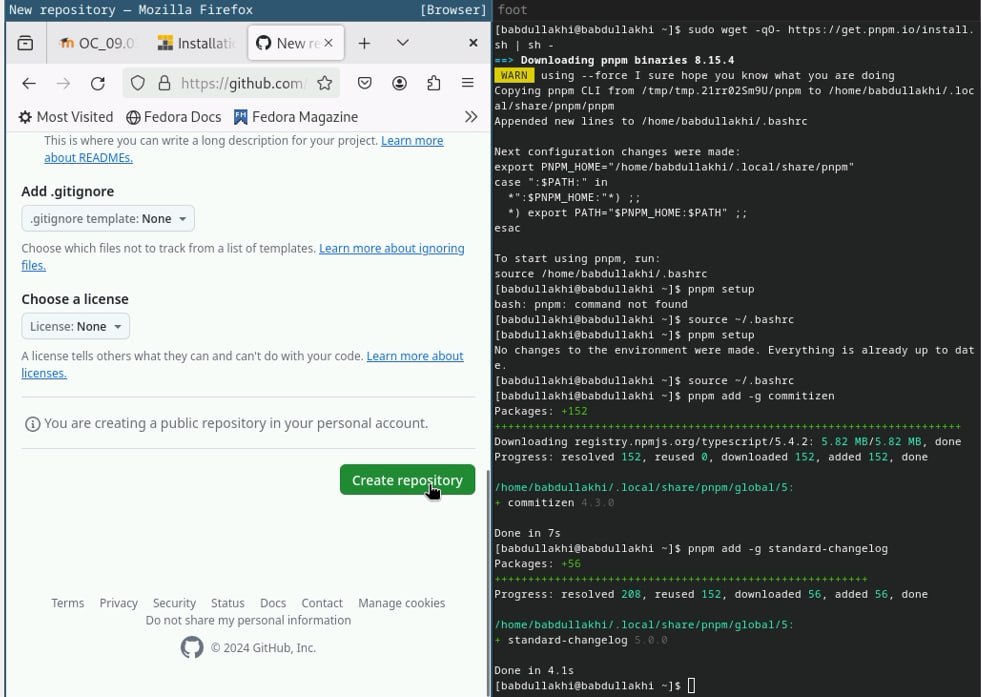
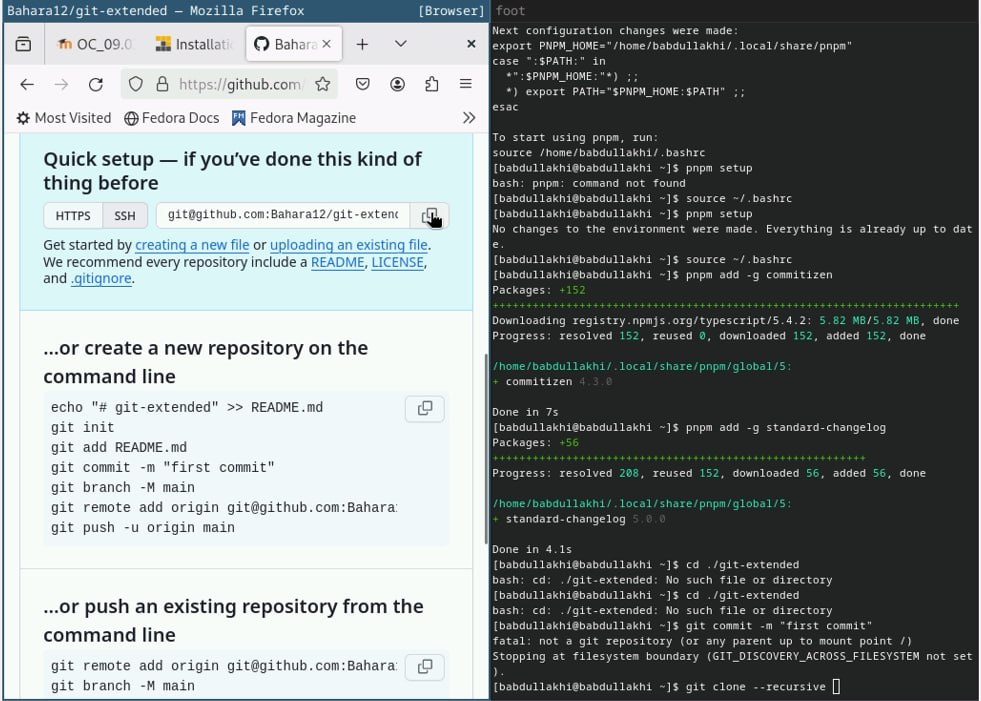
* При этом устанавливается скрипт git-cz, который мы и будем использовать для коммитов.
* standard-changelog:
* Данная программа используется для помощи в создании логов.



* Практический сценарий использования git:

1. Подключение репозитория к github

* Создайте репозиторий на GitHub. Для примера назовём его git-extended.

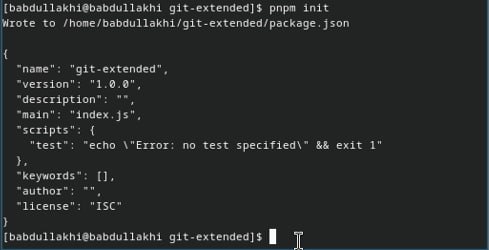
# Делаем первый коммит и выкладываем на github:

* Сначала мы входим в файл git-extended, затем создадим файл README.md, открываем там README.md, потом с помощью nano там напишим README.md после этого начинаем коммитить на github:

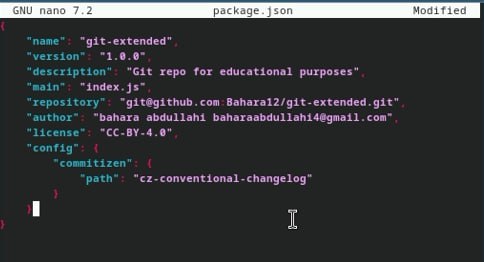


# Конфигурация общепринятых коммитов:

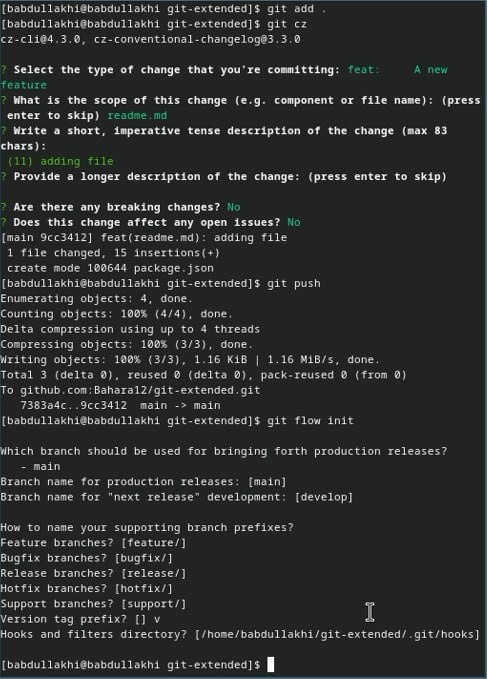
* Конфигурация для пакетов Node.js



* Сконфигурим формат коммитов. Для этого добавим в файл package.json команду для формирования коммитов:

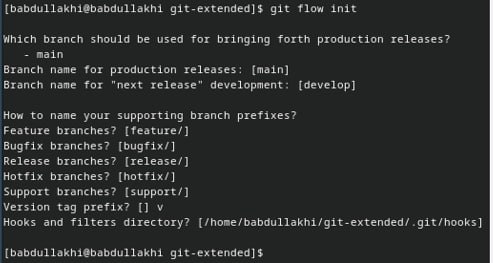


* добавим новые файлы, зафиксируем их и отправим на Gthub:

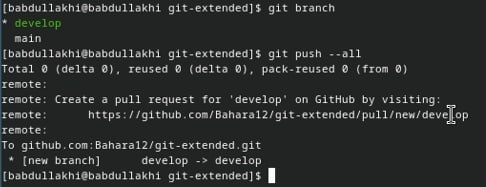


# Конфигурация git-flow:

* мы инициализируем git-flow и устанавливаем префикс для ярлыков v.



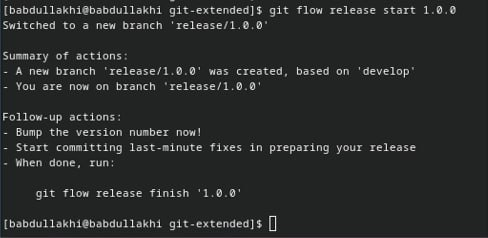
* Проверьте, что Вы на ветке develop:
* Загрузите весь репозиторий в хранилище:



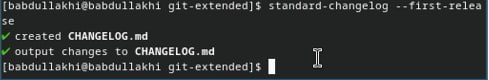
* Установите внешнюю ветку как вышестоящую для этой ветки:



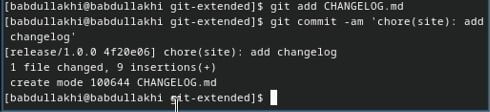
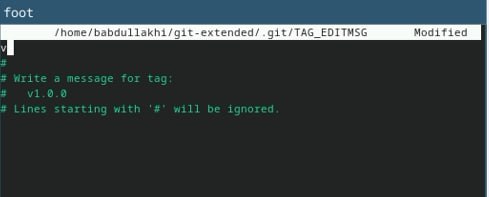
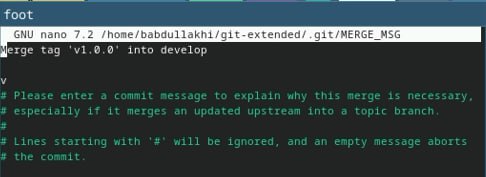
* Создадим релиз с версией 1.0.0



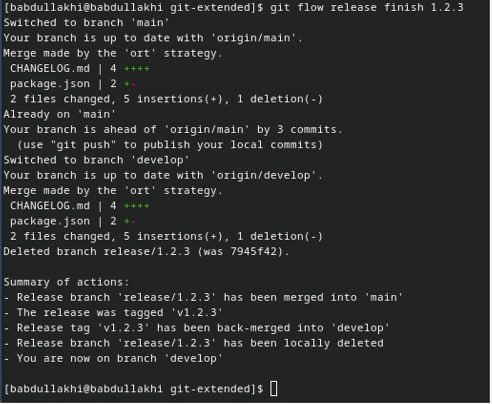
* Создадим журнал изменений



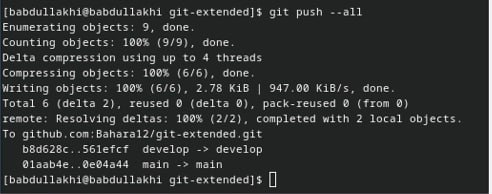
* Добавим журнал изменений в индекс:

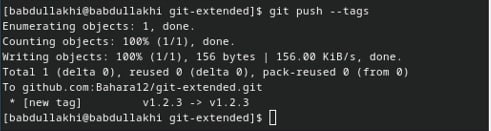
* Зальём релизную ветку в основную ветку:



* Отправим данные на github:

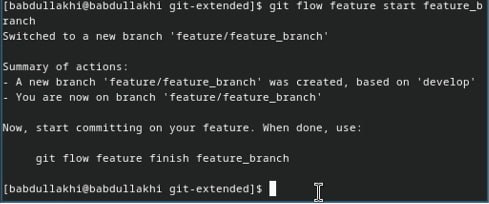


* Создадим релиз на github. Для этого будем использовать утилиты работы с github:

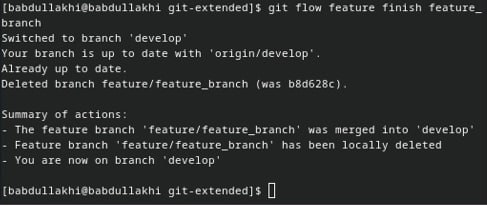


# Работа с репозиторием git:

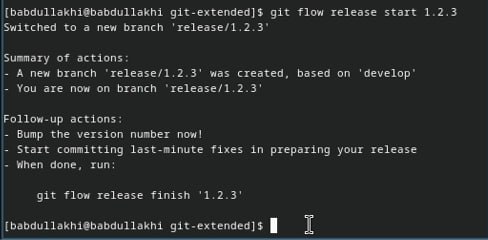
* Разработка новой функциональности:
* Создадим ветку для новой функциональности:



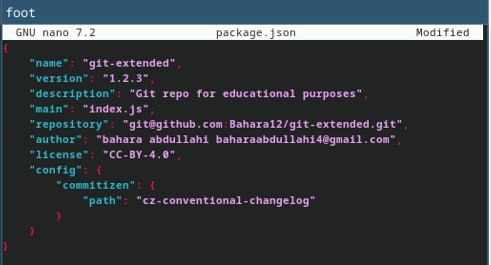
* следует объединить ветку:



* Создание релиза git-flow:
* Создадим релиз с версией 1.2.3:



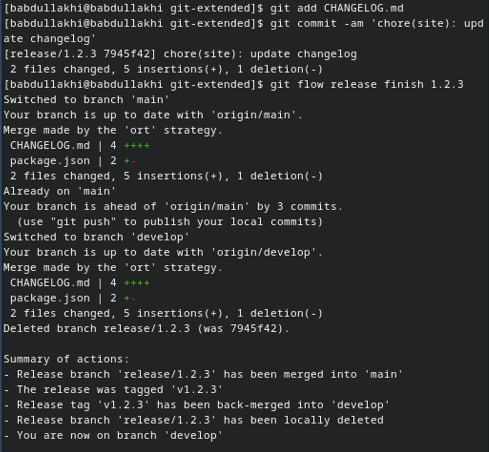
* Обновите номер версии в файле package.json.

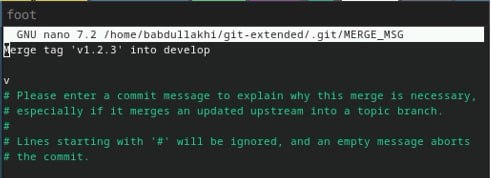


* Создадим журнал изменений

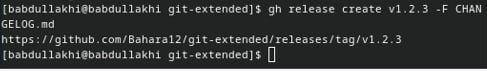


* Добавим журнал изменений в индекс
* Зальём релизную ветку в основную ветку





* Отправим данные на github
* Создадим релиз на github с комментарием из журнала изменений:



# Выводы:

* Получение навыков правильной работы с репозиториями git.