Лабораторная работа №4

Дисциплина: Операционные системы

Панина Жанна Валерьевна

Содержание

# 1 Цель работы

Получение навыков правильной работы с репозиториями git.

# 2 Задание

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

# 3 Теоретическое введение

Gitflow Workflow опубликована и популяризована Винсентом Дриссеном. Gitflow Workflow предполагает выстраивание строгой модели ветвления с учётом выпуска проекта. Данная модель отлично подходит для организации рабочего процесса на основе релизов.v Работа по модели Gitflow включает создание отдельной ветки для исправлений ошибок в рабочей среде.

Семантическое версионирование описывается в манифесте семантического версионирования. Кратко его можно описать следующим образом: Версия задаётся в виде кортежа МАЖОРНАЯ\_ВЕРСИЯ.МИНОРНАЯ\_ВЕРСИЯ.ПАТЧ. Номер версии следует увеличивать: МАЖОРНУЮ версию, когда сделаны обратно несовместимые изменения API. МИНОРНУЮ версию, когда вы добавляете новую функциональность, не нарушая обратной совместимости. ПАТЧ-версию, когда вы делаете обратно совместимые исправления. Дополнительные обозначения для предрелизных и билд-метаданных возможны как дополнения к МАЖОРНАЯ.МИНОРНАЯ.ПАТЧ формату.

Спецификация Conventional Commits: Соглашение о том, как нужно писать сообщения commit’ов. Совместимо с SemVer. Даже вернее сказать, сильно связано с семантическим версионированием. Регламентирует структуру и основные типы коммитов.

# 4 Выполнение лабораторной работы

## 4.1 Установка программного обеспечения

### 4.1.1 Установка git-flow

Устанавливаю git-flow из коллекции репозиториев Copr 1. Выполняю команду dnf copr enable elegos/gitflow в режиме суперпользователя (рис. 1).

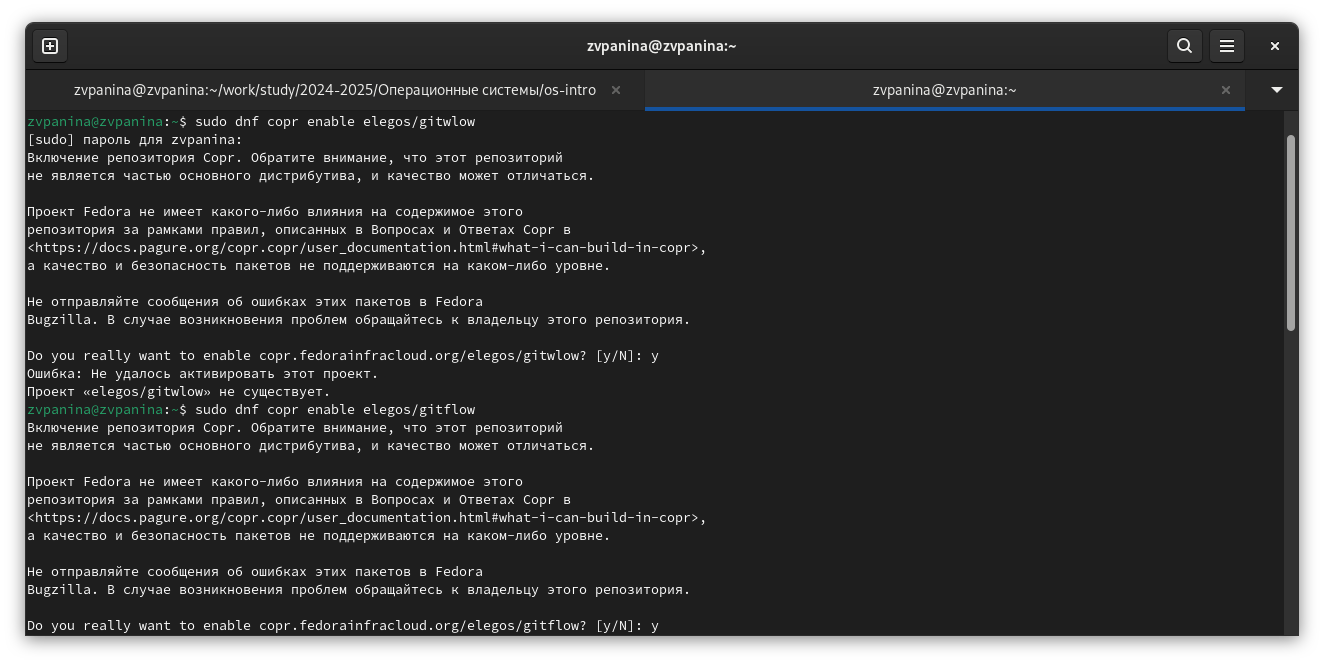


Рис. 1: Команда dnf copr enable elegos/gitflow

1. Выполняю команду dnf install gitflow (рис. 2).

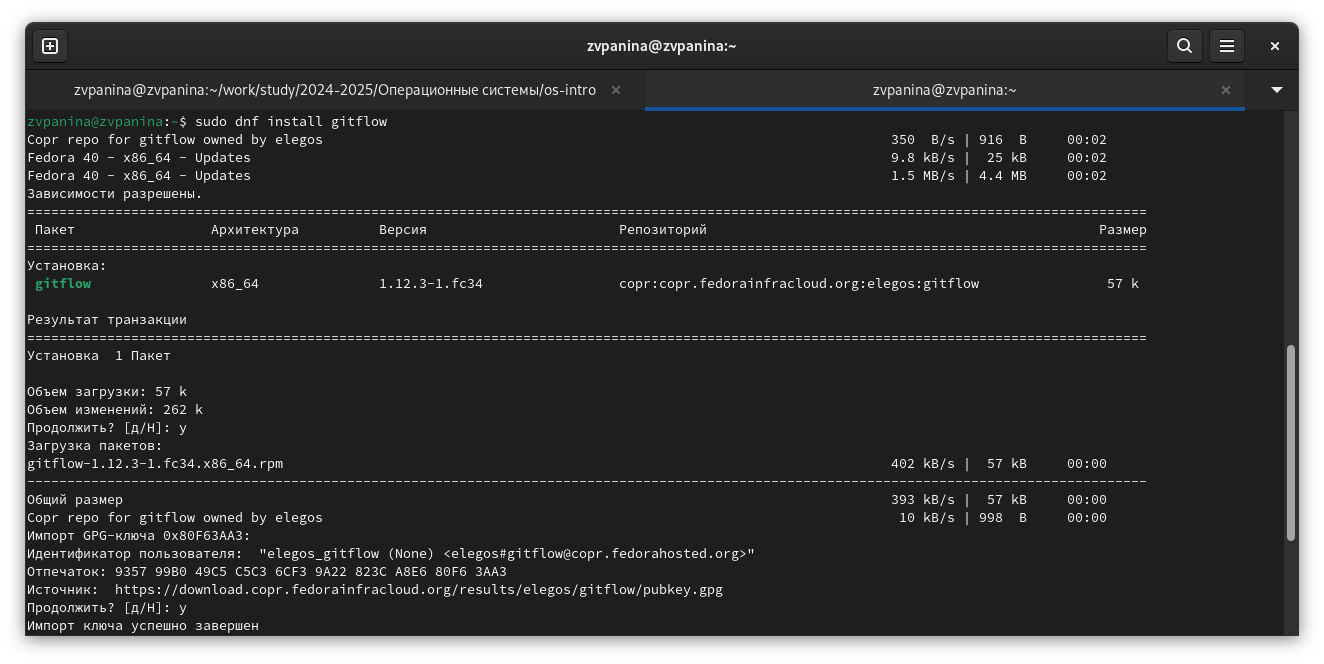


Рис. 2: Команда dnf install gitflow

### 4.1.2 Установка Node.js

1. Выполняю команду dnf install nodejs в режиме суперпользователя (рис. 3).

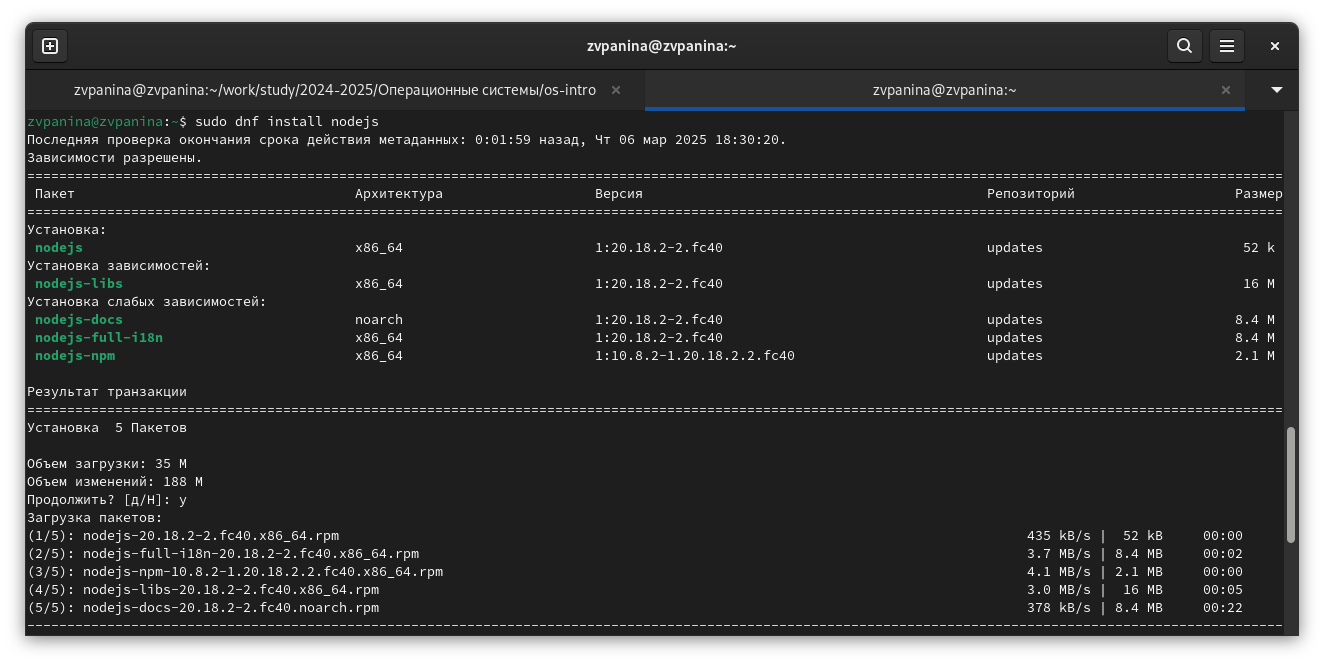


Рис. 3: Команда dnf install nodejs

1. Выполняю команду dnf install pnpm (рис. 4).

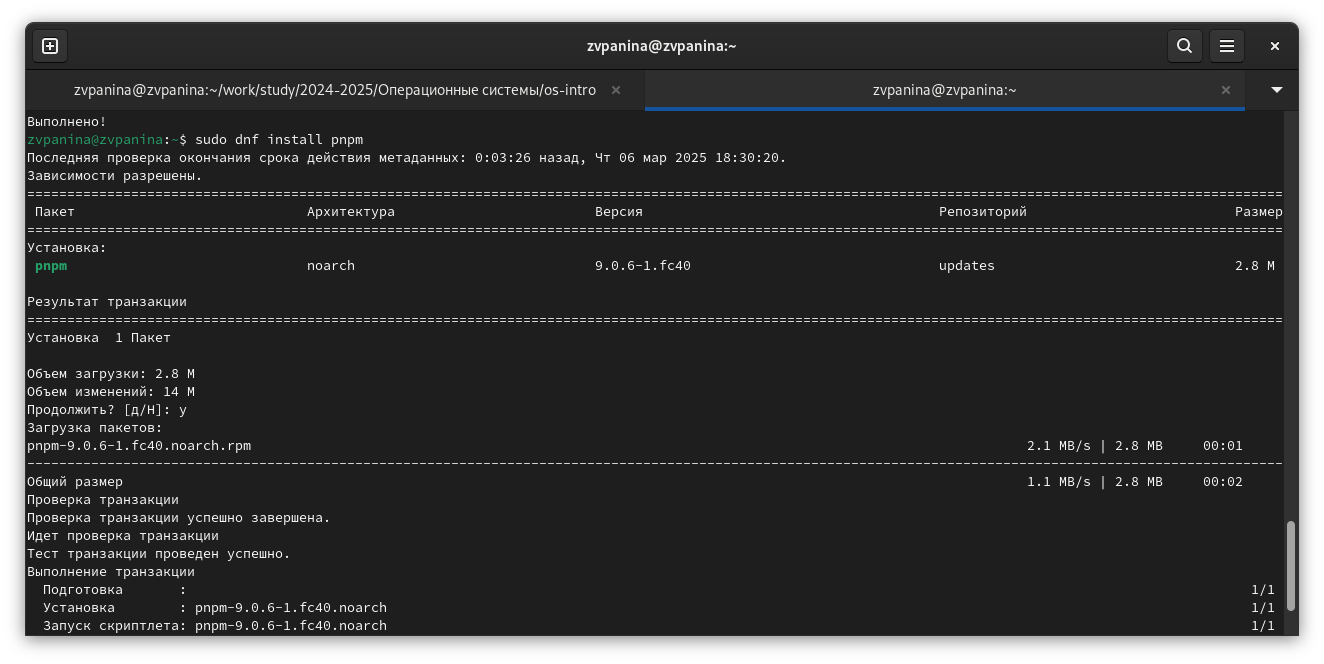


Рис. 4: Команда dnf install pnpm

### 4.1.3 Настройка Node.js

Для работы с Node.js добавляю каталог с исполняемыми файлами, устанавливаемыми в yarn, в переменную PATH (рис. 5).

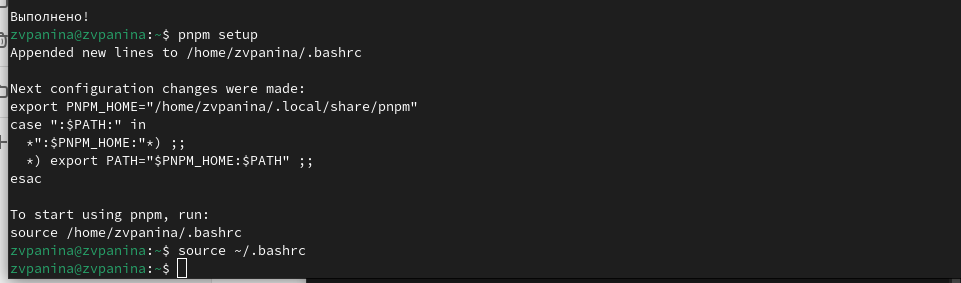


Рис. 5: Настройка Node.js

### 4.1.4 Общепринятые коммиты

1. Данная программа используется для помощи в форматировании коммитов (рис. 6).

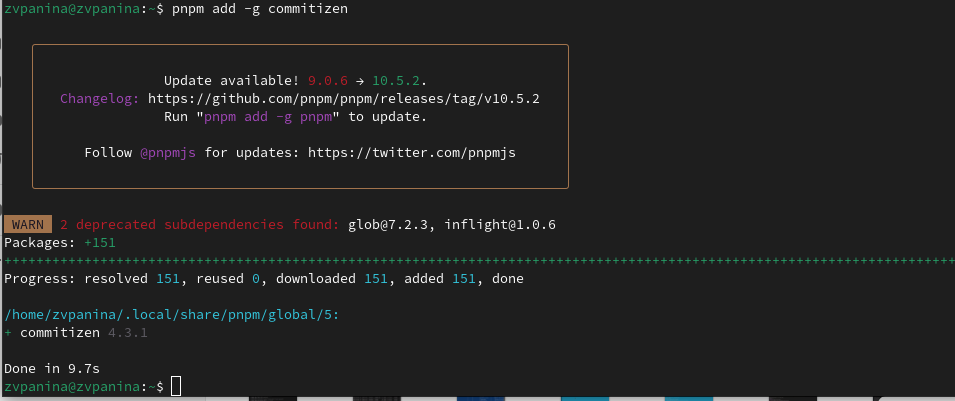


Рис. 6: Программа commitizen

1. Данная программа используется для помощи в создании логов (рис. 7).

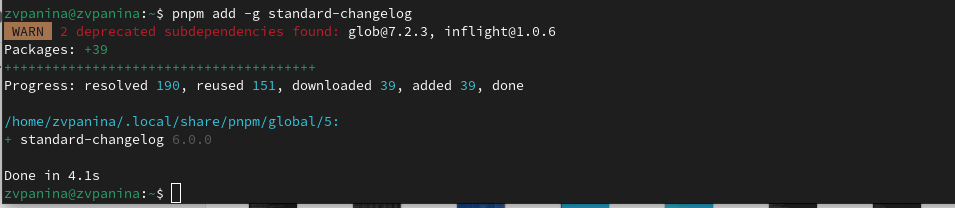


Рис. 7: Программа standard-changelog

## 4.2 Практический сценарий использования git

### 4.2.1 Создание репозитория git

1. Создаю репозиторий на GitHub и называю его git-extended (рис. 8).

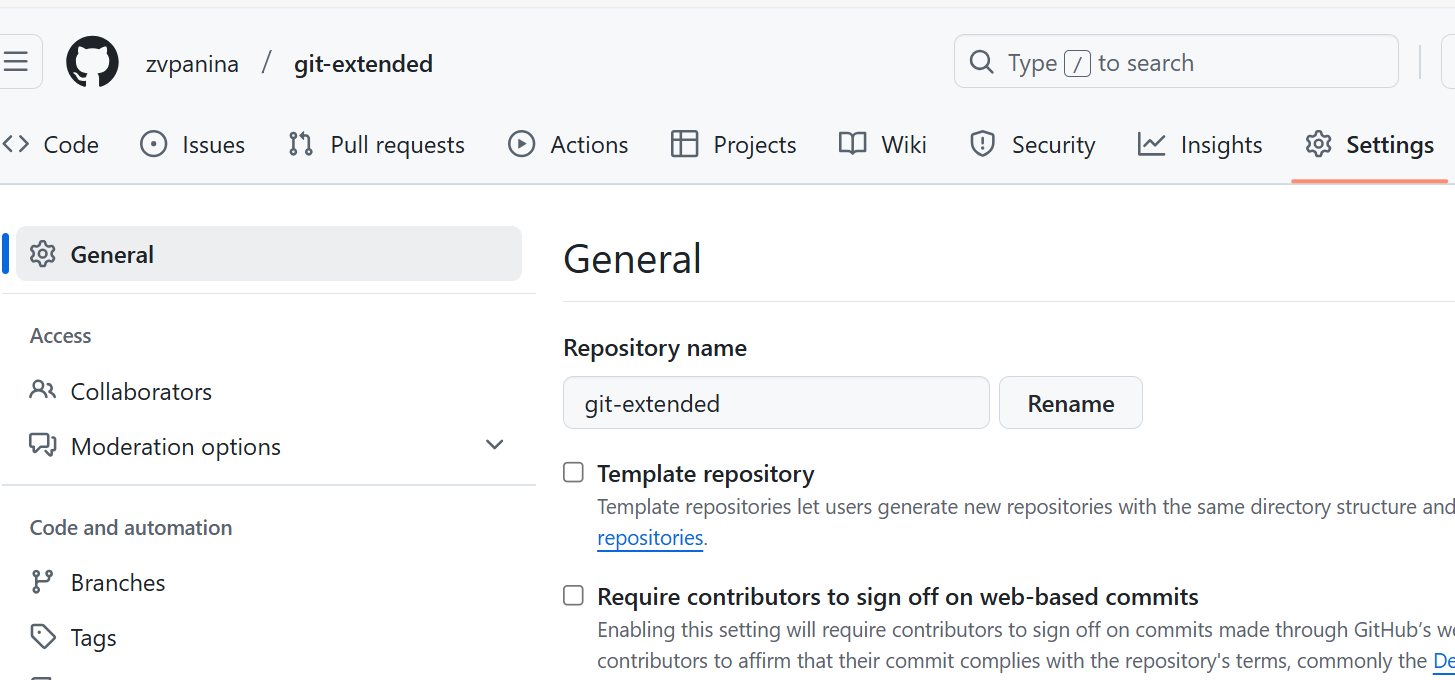


Рис. 8: Создание репозитория

1. Клонирую репозиторий, перехожу в него и создаю файл README.md, чтобы закоммитить его. С помощью команд git add., git commit, git push делаю первый коммит и выкладываю его на GitHub (рис. 9).

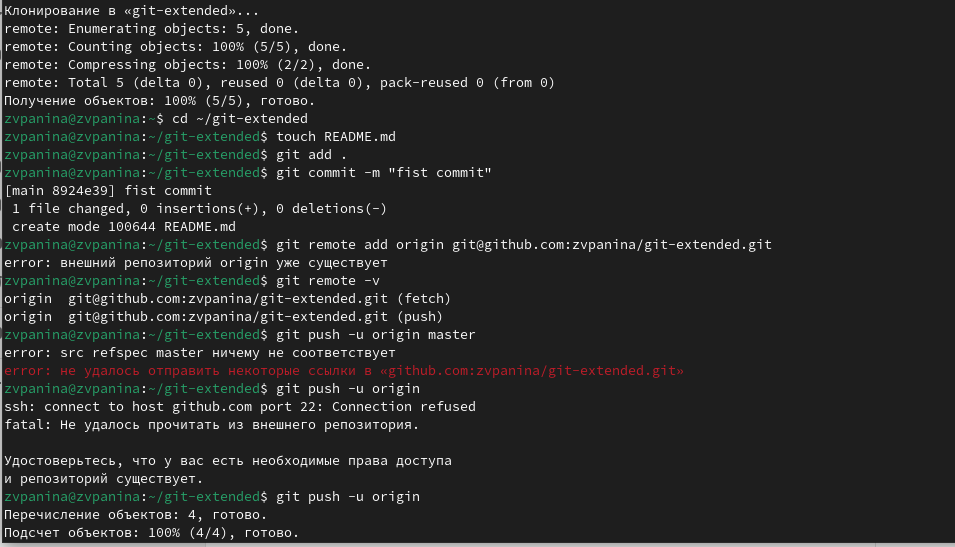


Рис. 9: Первый коммит

1. Создаю конфигурацию для пакетов Node.js и открываю файл git-extended в mc (рис. 10).

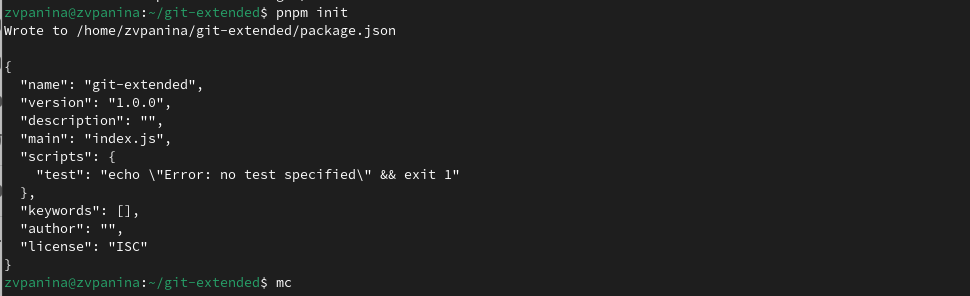


Рис. 10: Конфигурация общепринятых коммитов

1. Заполняю несколько параметров пакета (название, лицензия, конфигурация пакета коммитов), чтобы файл выглядел вот так (рис. 11).



Рис. 11: Заполнение параметров пакета

1. С помощью следующих команд добавляю новые файлы, выполняю коммит, выкладываю на GitHub (рис. 12).

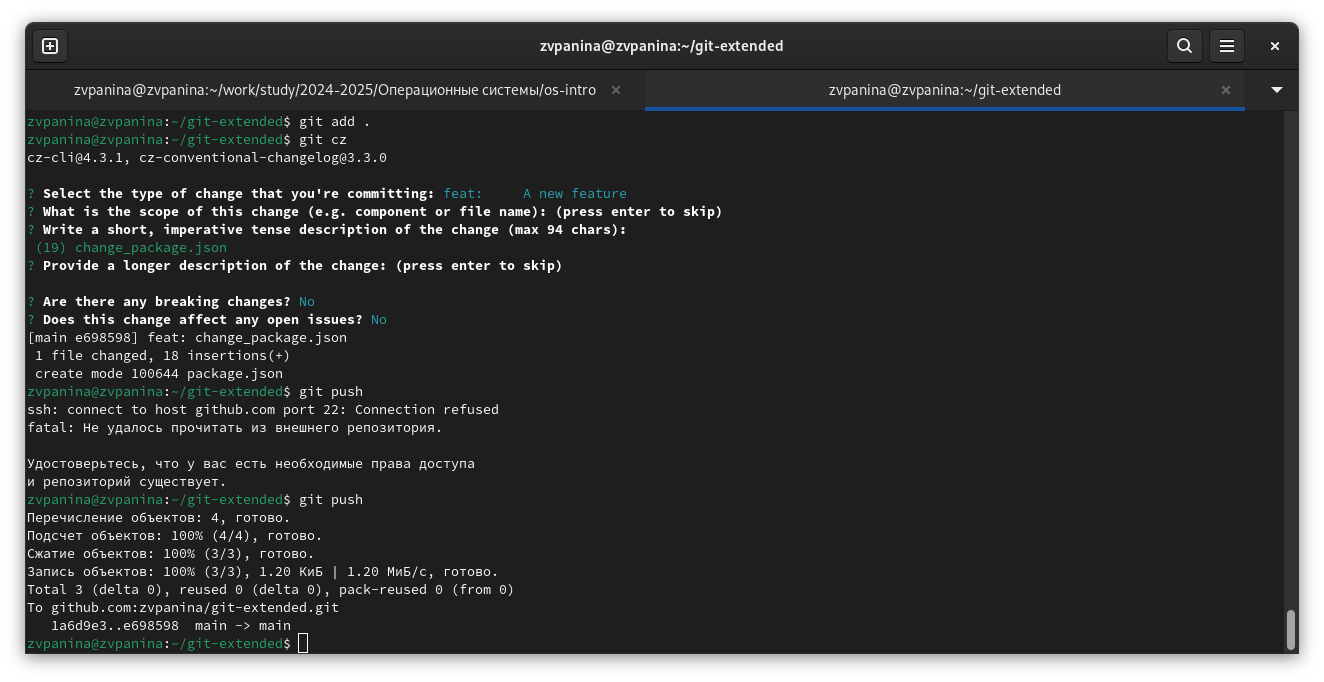


Рис. 12: Отправка файлов на GitHub

1. Инициализирую git-flow (Префикс для ярлыков устанавливаю в v); проверяю, что я на ветке develop; загружаю весь репозиторий в хранилище; устанавливаю внешнюю ветку как вышестоящую для этой ветки (рис. 13).

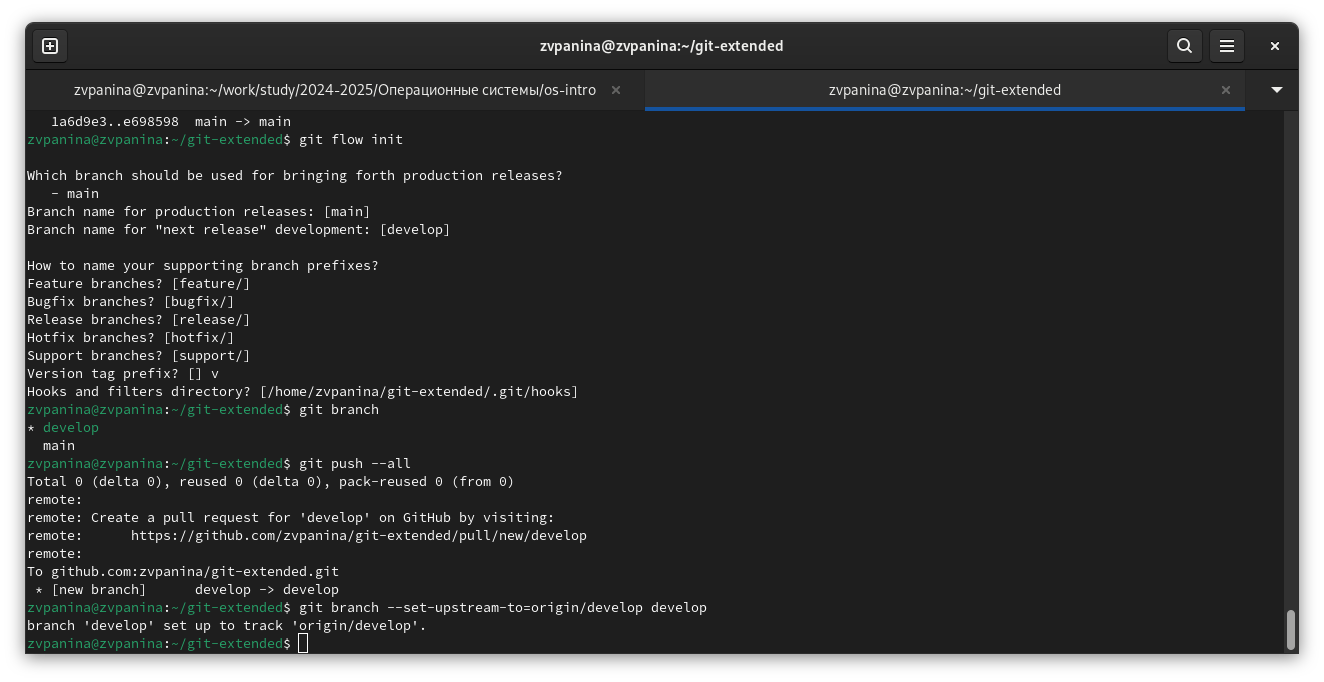


Рис. 13: Конфигурация git-flow

1. Создаю релиз с версией 1.0.0 , создаю журнал изменений (рис. 14).

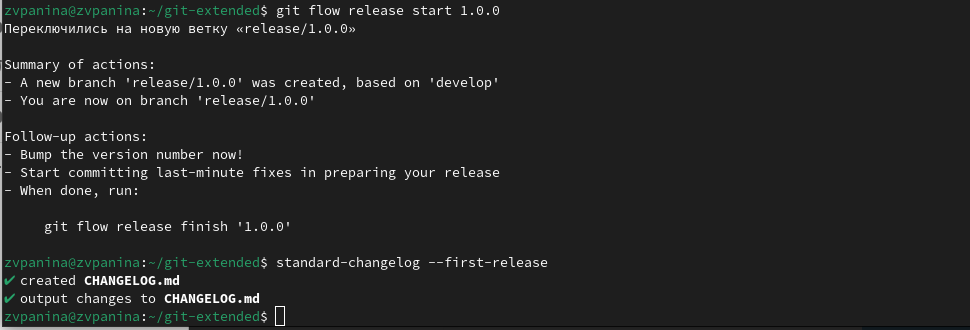


Рис. 14: Создание релиза и журнала изменений

После этой команды всплывает окно, где нужно написать сообщение для коммита (рис. 15).

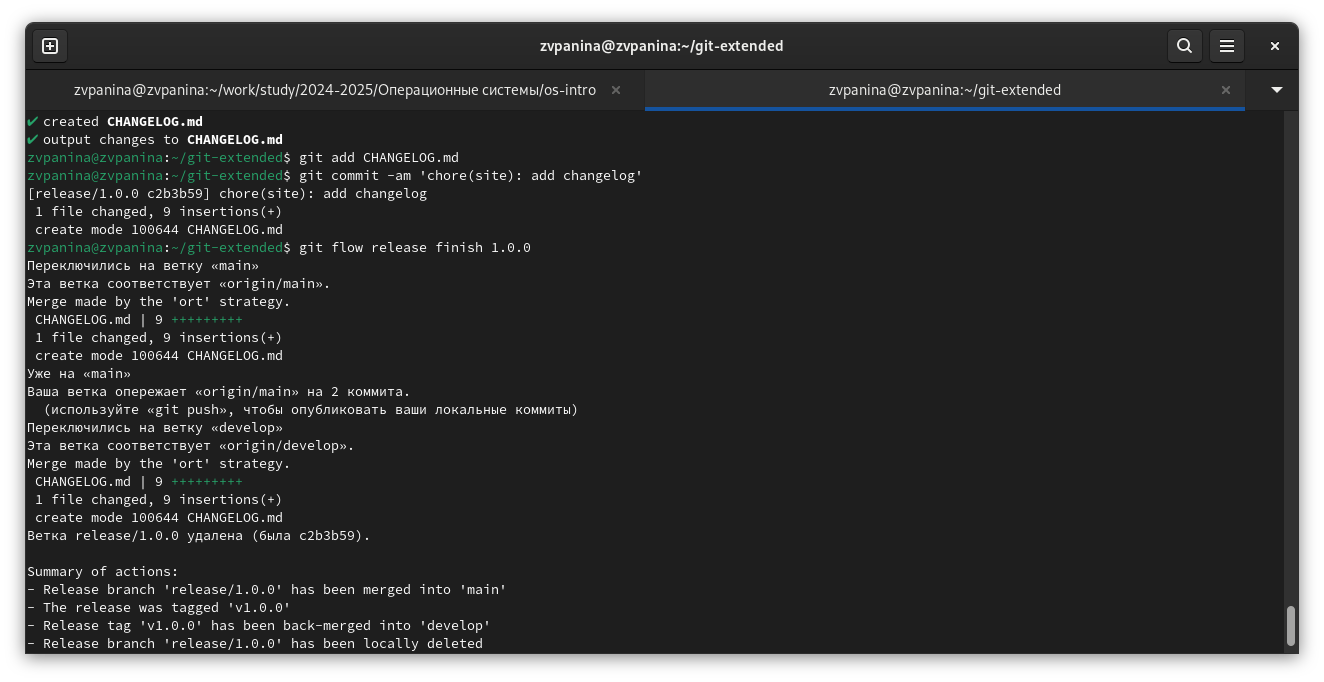


Рис. 15: Редактирование коммита

1. Заливаю релизную ветку в основную ветку (рис. 16).

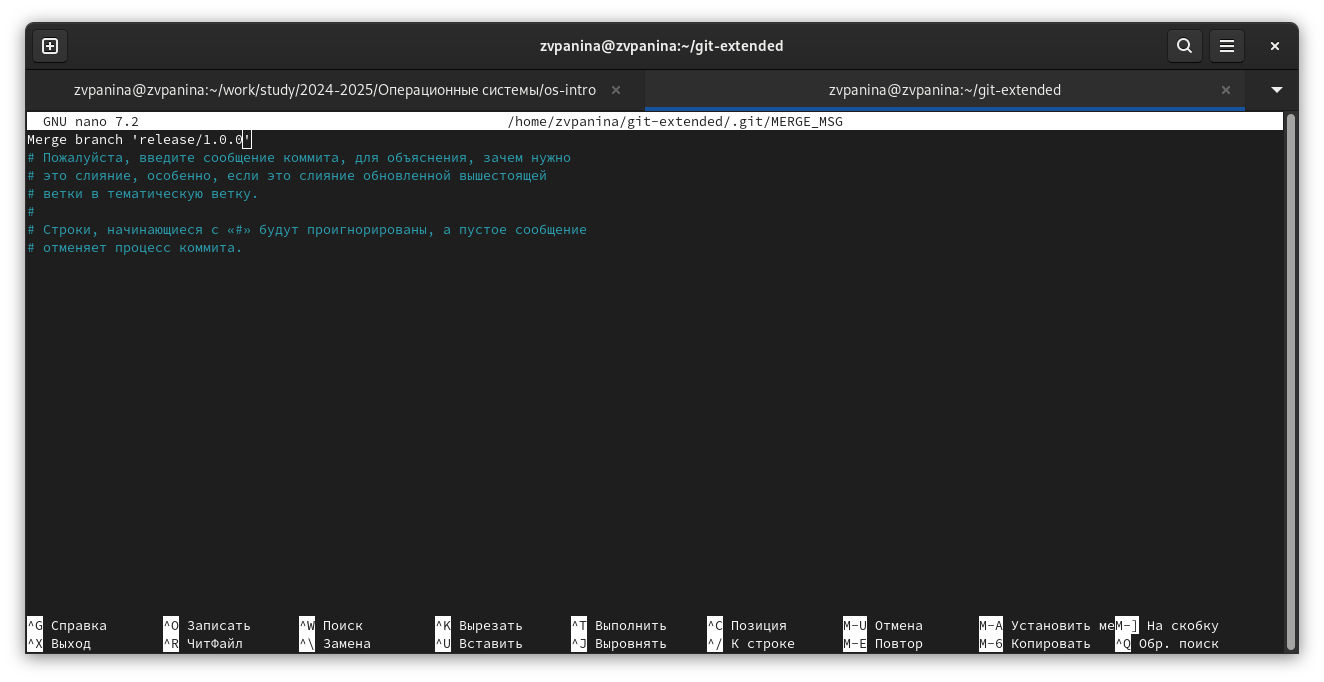


Рис. 16: Команда finish

1. Отправляю данные на github (рис. 17).

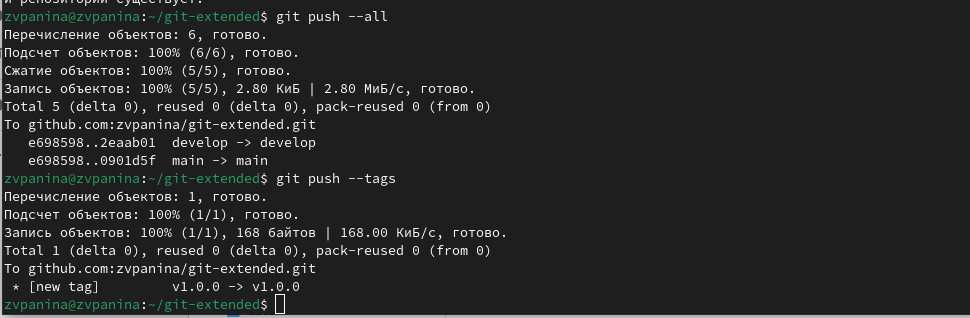


Рис. 17: Отправка данных на github

1. Создаю релиз на github. Для этого использую утилиты работы с github (Первая команда на рисунке 4.18) .

### 4.2.2 Работа с репозиторием git

1. Создаю ветку для новой функциональности. Объединяю ветку feature\_branch с develop (рис. 18).

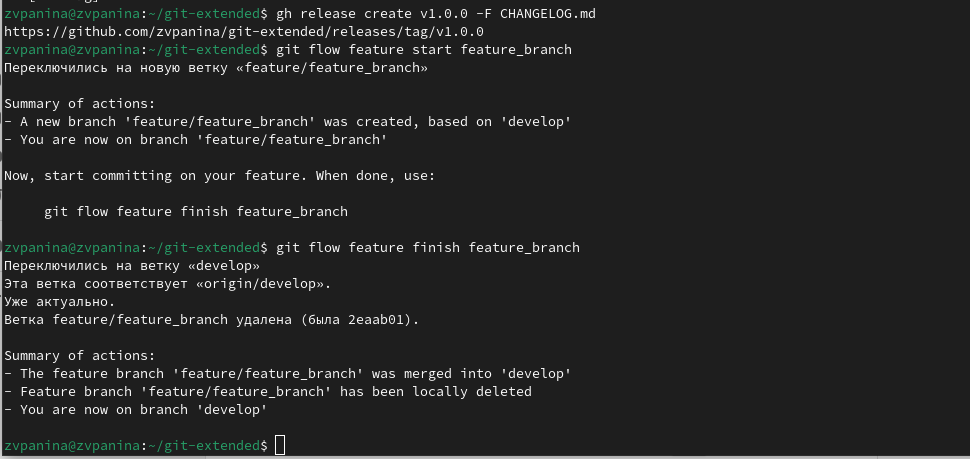


Рис. 18: Разработка новой функциональности

1. Создаю релиз с версией 1.2.3 и открываю файл package\_json в mc (рис. 19).

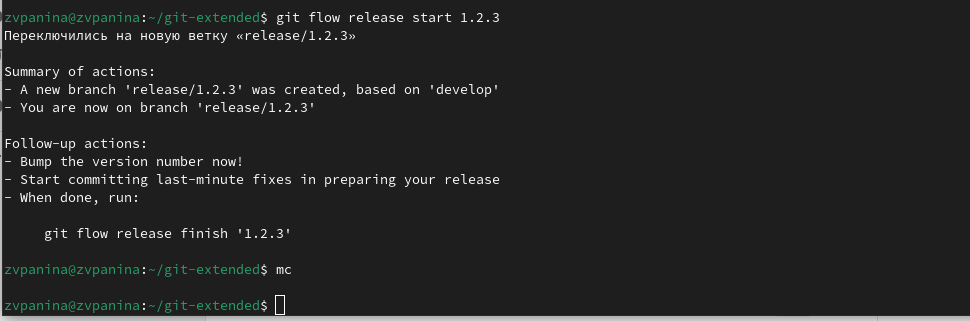


Рис. 19: Создание релиза git-flow

1. В файле устанавливаю номер версии в 1.2.3 (рис. 20).

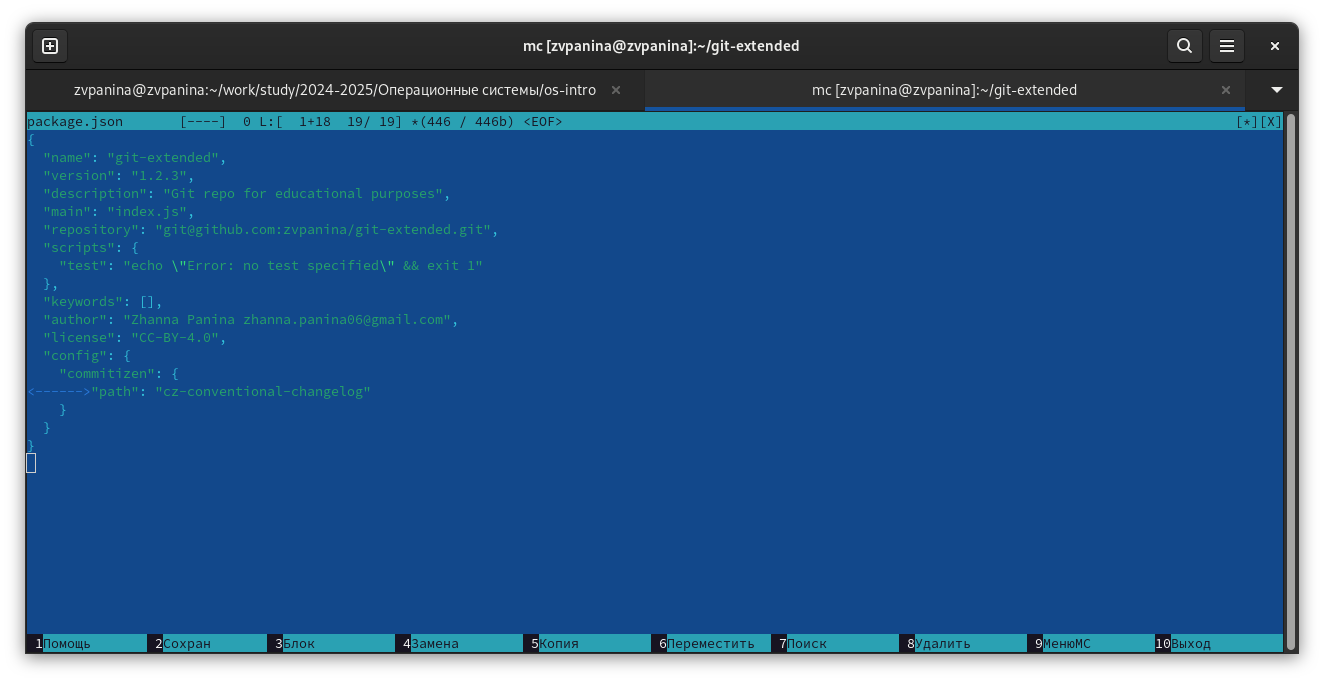


Рис. 20: Обновление номера версии в файле

1. Создаю журнал изменений, добавляю его в индекс (рис. 21).

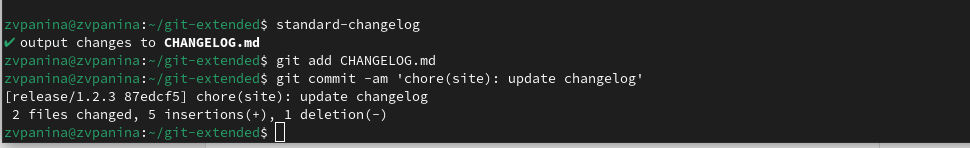


Рис. 21: Создание журнала изменений

1. В всплывающем окне пришу сообщение для коммита (рис. 22).

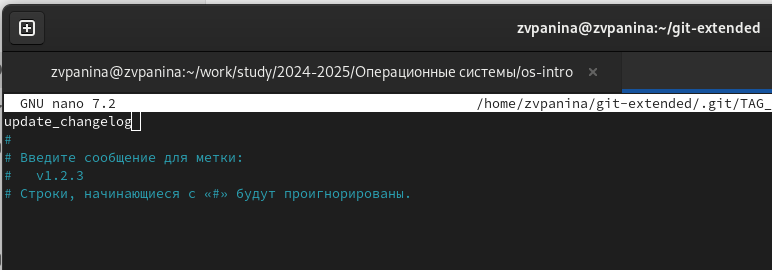


Рис. 22: Редактирование коммита

1. Заливаю релизную ветку в основную ветку (рис. 23).

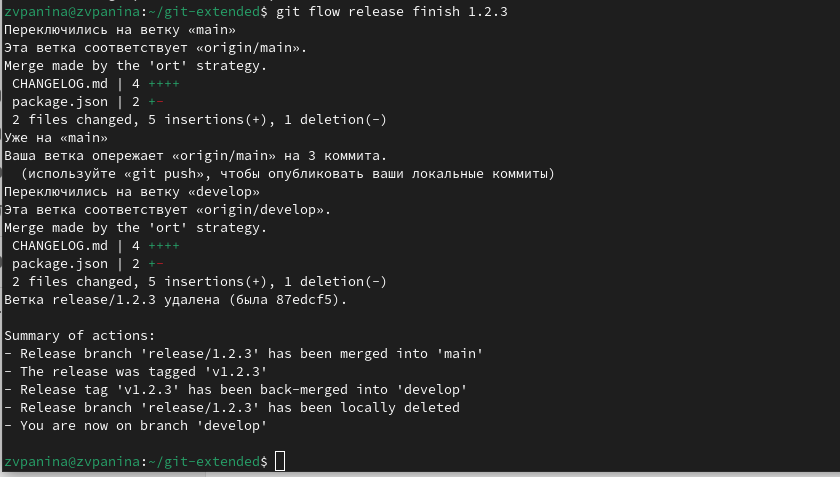


Рис. 23: Команда finish

1. Отправляю данные на github. Создаю релиз на github с комментарием из журнала изменений (рис. 24).

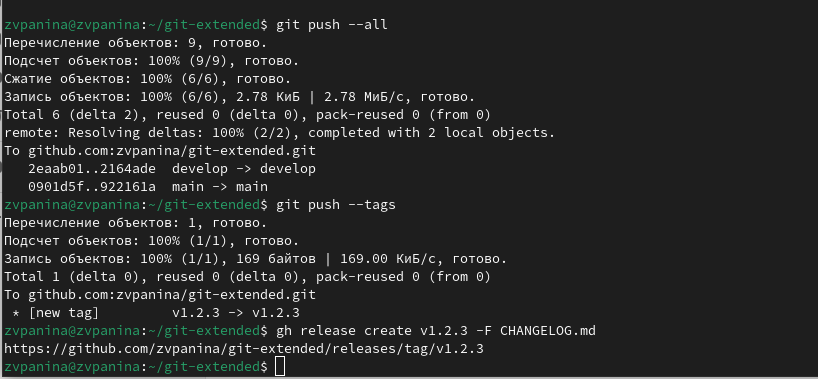


Рис. 24: Создание релиза на github

# 5 Выводы

В ходе выполнения лабораторной работы я приобрела навыки правильной работы с репозиториями git.

# Список литературы

Рабочий процесс с Gitflow(электронный ресурс) URL: <https://yamadharma.github.io/ru/post/2021/04/18/gitflow-workflow/>