Шаблон отчёта по лабораторной работе

Продвинутое исполльзование git

Талебу тенке франк устно

Содержание

# 1 Цель работы

Рабочий процесс Gitflow Workflow. Будем описывать его с использованием пакета git-flow.

Общая информация

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.  
Работа по модели Gitflow включает создание отдельной ветки для исправлений ошибок в рабочей среде.  
Последовательность действий при работе по модели Gitflow:  
 Из ветки master создаётся ветка develop.  
 Из ветки develop создаётся ветка release.  
 Из ветки develop создаются ветки feature.  
 Когда работа над веткой feature завершена, она сливается с веткой develop.  
 Когда работа над веткой релиза release завершена, она сливается в ветки develop и master.  
 Если в master обнаружена проблема, из master создаётся ветка hotfix.  
 Когда работа над веткой исправления hotfix завершена, она сливается в ветки develop и master.

# 2 Задание

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

# 3 Теоретическое введение

Рабочий процесс Gitflow Рабочий процесс Gitflow Workflow. Будем описывать его с использованием пакета git-flow.

Общая информация 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. Работа по модели Gitflow включает создание отдельной ветки для исправлений ошибок в рабочей среде. Последовательность действий при работе по модели Gitflow:

# 4 Выполнение лабораторной работы

Установка программного обеспечения Установка git-flow (рис. **¿fig:001?**). Название рисунка

Установка Node.js (рис. **¿fig:002?**). Название рисунка

Настройка Node.js Для работы с Node.js добавим каталог с исполняемыми файлами, устанавливаемыми yarn, в переменную PATH. (рис. **¿fig:003?**). Название рисунка

(рис. **¿fig:009?**). Название рисунка

Перелогиньтесь, или выполните:  
source ~/.bashrc

Общепринятые коммиты commitizen (рис. **¿fig:010?**). Название рисунка

Данная программа используется для помощи в форматировании коммитов. (рис. **¿fig:011?**). Название рисунка

(рис. **¿fig:004?**).

Название рисунка Данная программа используется для помощи в форматировании коммитов. При этом устанавливается скрипт git-cz, который мы и будем использовать для коммитов. standard-changelog (рис. **¿fig:005?**). Название рисунка

Подключение репозитория к github Создайте репозиторий на GitHub. Для примера назовём его git-extended. Делаем первый коммит и выкладываем на github: (рис. **¿fig:006?**).

Название рисунка Конфигурация общепринятых коммитов Конфигурация для пакетов Node.js pnpm init Необходимо заполнить несколько параметров пакета. (рис. **¿fig:012?**). Название рисунка (рис. **¿fig:013?**). Название рисунка Добавим новые файлы: git add . Выполним коммит: git cz Отправим на githu (рис. **¿fig:014?**). Название рисунка

(рис. **¿fig:015?**). Название рисунка

рис. **¿fig:016?**). Название рисунка Конфигурация git-flow

Инициализируем git-flow

рис. **¿fig:017?**). Название рисунка

Проверьте, что Вы на ветке develop:

рис. **¿fig:018?**). [Название рисунка](image/18.jpg) Добавим журнал изменений в индекс рис. **¿fig:019?**). [Название рисунка](image/19.jpg) Зальём релизную ветку в основную ветку Отправим данные на github рис. **¿fig:020?**). Название рисунка # Выводы

Здесь кратко описываются итоги проделанной работы.