Федеральное государственное бюджетное образовательное учреждение

высшего образования

РОССИЙСКАЯ АКАДЕМИЯ НАРОДНОГО ХОЗЯЙСТВА и ГОСУДАРСТВЕННОЙ СЛУЖБЫ

при ПРЕЗИДЕНТЕ РОССИЙСКОЙ ФЕДЕРАЦИИ

НИЖЕГОРОДСКИЙ ИНСТИТУТ УПРАВЛЕНИЯ

Факультет управления  
Кафедра информатики и информационных технологий  
Направление (профиль) подготовки: 09.03.03 Прикладная информатика

Отчёт по проделанной лабораторной работе

по предмету «Программная инженерия» № 1

Выполнил студент группы Иб-321

Бондарев Георгий Александрович

Нижний Новгород

2024

Теоретическая часть

Назначение, возможности и особенности ...............................................................................3

Сравнение Git, GitHub и GitLab...............................................................................................4

Российская система GitFlick и её возможности. ....................................................................5

Команды системы Git: названия и назначение.....................................................................6

Регистрация в GitHub………………………………………7

**Назначение, возможности и особенности**

Назначение Git. Git используется для управления версиями файлов и их

изменений в проектах разработки ПО. Он позволяет отслеживать историю

изменений, управлять ветками разработки, совместно работать над проектом,

а также интегрировать изменения из разных источников.

Особенности Git:

 Распределенная система контроля версий: Каждый участник проекта

работает с полной копией репозитория, что позволяет им работать

независимо от сети.

 Эффективное хранение данных: Git хранит данные в виде снимков

файлов, а не изменений, что делает операции быстрыми и

эффективными.

 Ветвление и слияние: Легкость создания и управления ветками, а

также слияние изменений между ними.

 Гибкая конфигурация: Пользовательские конфигурации и

возможность настройки хуков для автоматизации процессов.

 Безопасность: Использование хеш-функций для гарантирования

целостности данных и обнаружения изменений.

 Поддержка больших проектов: Git легко масштабируется и

поддерживает эффективную работу с большим объемом файлов и

коммитов.

Возможности Git:

 Отслеживание изменений. Git сохраняет историю всех изменений в

проекте, позволяя откатываться к предыдущим версиям.

 Создание веток. Удобное создание веток для разработки новых

функций или исправлений без влияния на основную ветку.

 Слияние изменений. Возможность объединения изменений из разных веток для интеграции новых функций.

 Удаленное совместное выполнение проекта. Работа нескольких

разработчиков над одним проектом через центральный удаленный

репозиторий.

 Откат изменений. Возможность откатиться к предыдущим версиям

проекта в случае необходимости.

 Разрешение конфликтов. Автоматическое и ручное разрешение

конфликтов при слиянии изменений.

**Сравнение Git, GitHub и GitLab**

1. Git:

 Git - это система управления версиями, позволяющая отслеживать

изменения в коде.

 Предоставляет основные команды для работы с репозиториями,

такие как ‘git init’, ‘git add’, ‘git commit’, ‘git push’, ‘git pull’ и т.д.

 Работает локально на компьютере разработчика.

2. GitHub:

 GitHub - это веб-платформа для хостинга репозиториев Git.

 Предоставляет возможности совместной работы над проектами,

управления задачами, отслеживания ошибок и другие

инструменты.

 Позволяет создавать открытые и закрытые репозитории, а также

делать форки и запросы на объединение изменений.

3. GitLab:

 GitLab - это аналогичная GitHub веб-платформа для хостинга

репозиториев Git.

 Предлагает возможность управления проектами, непрерывной

интеграции и доставки (CI/CD), управление правами доступа и

другие функции.

 Может быть развернут как самостоятельное приложение на

сервере.

**Российская система GitFlick и её возможности.**

GitFlic ― новый отечественный аналог GitHub. Его запустили в

декабре 2021 года, хотя зарубежный сервис заявил, что не уходит из

России. Приятно – и ладно.

Этот облачный сервис хранения репозиториев исходного кода GitFlic

от московской компании «Ресолют» можно считать альтернативой

американскому GitHub.

Его возможности содержат в себе все возможности Git: от создания и

выпуска, до тестов и контроля над проектом, однако есть особенности

проекта, которые выделяют его на фоне привычного GitHub:

 Возможность работы с образовательными учреждениями.

Для образовательных групп упрощенная выдача заданий, антиплагиат

модуль, обратная связь, автозапуск решений и автооценка работы.

Можно прямо пообщаться с командой, задать вопросы, рассказать свои

идеи и предложения в телеграм или в самом репозитории.

 Сам сервис можно развернуть на своем отдельном выделенном

сервере и пользоваться им локально. Возможность скачать и

установить сервер GitFlic также имеется

 Запросы на слияние. Нечто похожее на то, как в GitHub можно после

форка проекта предложить через пулл-реквест свои изменения.

 Двухфакторная аутентификация. Стандарт современной защиты.

 Трекер задач. Полная интеграция с кодом, что очень важно и

востребовано.

 Уведомления в телеграм. Как и было сказано выше – это отличная

возможность.

**Команды системы Git: названия и назначение.**

1. `git init`: Инициализация нового репозитория Git.

2. `git clone`: Клонирование существующего репозитория.

3. `git add`: Добавление файлов в индекс (staging area) для последующего

коммита.

4. `git commit`: Фиксация (сохранение) изменений в индексированных

файлах.

5. `git push`: Отправка изменений на удалённый репозиторий.

6. `git pull`: Получение и объединение изменений из удалённого

репозитория.

7. `git status`: Показ состояния рабочего каталога и индекса.

8. `git log`: Отображение истории коммитов.

9. `git branch`: Список, создание и удаление веток.

10. `git checkout`: Переключение между ветками или восстановление файлов

из репозитория.

11. `git merge`: Объединение изменений из одной ветки в другую.

12. `git fetch`: загрузка объектов и ссылок из другого репозитория.

13. `git reset`: Сброс индекса и рабочего каталога до определенного

состояния.

14. `git stash`: Скрытие локальных изменений.

15. `git remote`: Управление списком удалённых репозиториев.

16. `git diff`: Показать изменения, которые еще не проиндексированы.

17. `git rebase`: Применение коммитов на другую ветку базируясь на их

изменениях.

18. `git tag`: Создание, удаление и просмотр тегов.

19. `git cherry-pick`: Применение конкретного коммита к текущей ветке.

20. `git blame`: Отображение кому принадлежит каждая строка кода и какой

коммит ее внёс.

**Регистрация в GitHub**



