МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт Информационных технологий

Цифровая кафедра

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 4

по дисциплине

«Непрерывная разработка и интеграция CI/CD»

Тема практической работы: Внедрение проекта в GitLab

Выполнил студент группы 14 Стока И.П.

Руководитель практической работы Волков М.Ю.

Практическая работа выполнена «__»_____202__ г.

«Зачтено» «__»_____202__ г.

Москва 2023г.

Создадим новый проект в GitLab и назовем его pract (Рисунок 1).

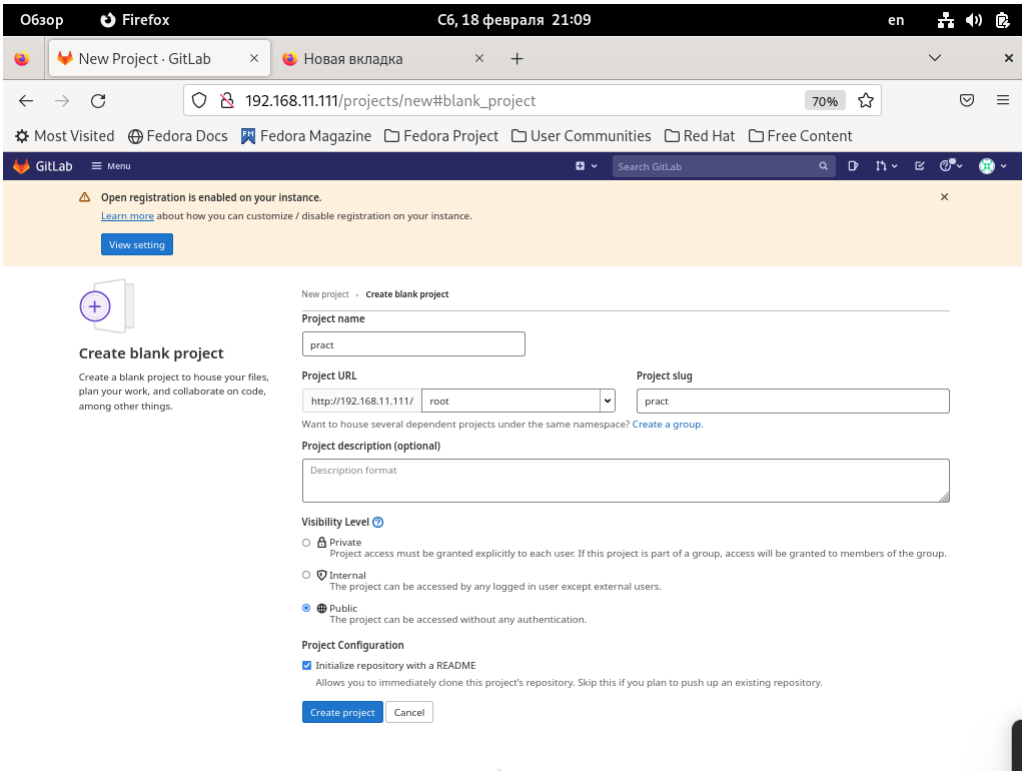

Рисунок 1 – Создание объекта в GitLab

Далее проведем выгрузку проекта на GitLab (Рисунок 2).

```
[stokaivan@fedora pract]$ git config --global user.name "Administrator"
[stokaivan@fedora pract]$ git config --global user.email admin@example.com
[stokaivan@fedora pract]$ git init --initial-branch=main
warning: re-init: ignored --initial-branch=main
Переинициализирован существующий репозиторий Git в /home/stokaivan/pyProject/pract/.git/
[stokaivan@fedora pract]$ git remote add origin http://192.168.11.111/root/prac.git
error: внешний репозиторий origin уже существует
[stokaivan@fedora pract]$ git add .
[stokaivan@fedora pract]$ git commit -m "Initial commit"
[main (корневой коммит) 49f8bc7] Initial commit
 1 file changed, 1 insertion(+)
 create mode 100644 project.py
```

Рисунок 2 – Подготовка выгрузка проекта в GitLab

Пересылаем проект в удаленный репозиторий при помощи команды push (Рисунок 3).

```
[stokaivan@fedora pract]$ git push -u origin main
Username for 'http://192.168.11.111': root
Password for 'http://root@192.168.11.111':
warning: переадресация на http://192.168.11.111/root/prac.git/
Перечисление объектов: 3, готово.
Подсчет объектов: 100% (3/3), готово.
Запись объектов: 100% (3/3), 238 байтов | 238.00 КиБ/с, готово.
Всего 3 (изменений 0), повторно использовано 0 (изменений 0), повторно использовано паке
тов 0
To http://192.168.11.111/root/prac
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
[stokaivan@fedora pract]$
```

Рисунок 3 – Выгрузка проекта в GitLab

Обновим страницу GitLab и убедимся, что проект был загружен (Рисунок 4).

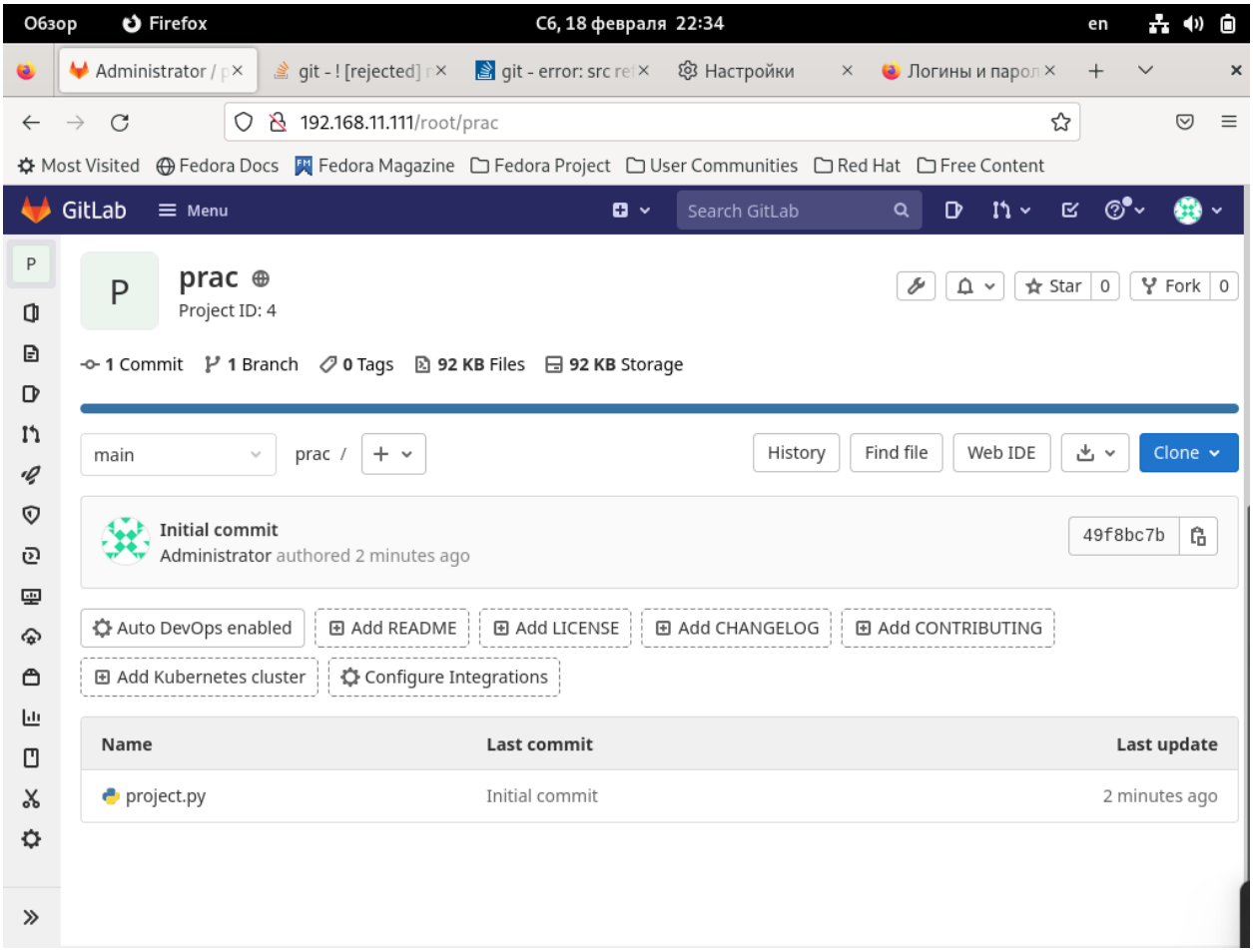

Рисунок 4 – Загруженный проект в удаленный репозиторий

Вопросы к практической работе

1. Набор ПО, позволяющего управлять процессом установки, настройки, обновления и удаления различных компонентов ПО. Так, например, компания Red Hat использует RPM и yum, также есть dpkg и apt и многие другие.

2. Место для хранения и поддержания данных, чаще всего данные хранятся в файлах и в дальнейшем распространяются по сети.

3. Это этап процесса разработки ПО, заключающееся в автоматизации широкого спектра задач: компиляции исходного кода в объектный модуль, сборка бинарного кода в исполняемый файл, тестирование, развертывание и много другое.

4. Автоматизация по запросу (по запросу пользователя), запланированная автоматизация (непрерывная интеграция), условная автоматизация (при каждом изменении кода).

5. Это процесс вызовов компилятора и компоновщика, передающихся множеству компьютеров для ускоренной сборки.

6. Преимущества: проблемы интеграции выявляются и исправляются быстро, быстрый прогон модульных тестов для свежих изменений, постоянное наличие текущей стабильной версии вместе с продуктом сборки. Недостатки: значительные затраты, необходимость в дополнительных вычислительных ресурсах под нужды непрерывной интеграции.

7. Является компонентом верхнего уровня CICD, также включают в себя задания, которые определяют, что делать, этапы, определяющие время выполнения заданий.

8. Build compile -> test test1 test2 -> staging deploy-to-stage -> production deploy-to-prod

9. Система автоматической сборки, представляющая DSL на языках Groovy и Kotlin вместо традиционной XML-образной формы представления конфигурации проекта.

10. Это фреймворк для автоматизации сборки проектов на основе описания их структуры в файлах на языке POM.

11. Gradle основан на графе зависимостей задач, в котором задачи — это то, что выполняет работу, в то время как Maven основан на фиксированной и линейной модели фаз. В Maven цели привязываются к фазам проекта, и цели выполняют ту же функцию, что и задачи в Gradle, являясь "элементами, которые выполняют работу".