

**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

**ОТЧЕТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ №2**

дисциплина: **Архитектура компьютера**

**Студент: Газдиев Ахмад**

Группа: НКАбд-02-25

**МОСКВА**

**2025 г.**

### **Цель работы**

Целью работы является изучение идеологии и применения средств контроля версий, приобретение практических навыков по работе с системой контроля версий git.

# Теоретическое введение

## 2.1 Системы контроля версий: основные представления

Система контроля версий (VCS) — это инструмент, позволяющий отслеживать изменения в файлах и при необходимости возвращаться к любому состоянию проекта. Проще говоря, она хранит историю редактирования и даёт возможность работать с файлами как с «черновиками», где можно всегда откатиться к нужному этапу.

Применяться такие системы могут не только в программировании, но и в любых сферах, где важен учёт изменений. Например, дизайнеры и разработчики сайтов используют их для сохранения разных версий макетов или изображений. VCS позволяет понять, кто и когда внёс правки, восстановить исходный вариант проекта, выявить ошибку и устранить её без риска потери данных. Таким образом, контроль версий даёт уверенность в сохранности информации и экономит время при совместной работе.

## 2.2 Git как система контроля версий

Git — это распределённая система управления версиями, которая хранит не одну, а множество независимых копий проекта. Каждая копия может развиваться в своём направлении, а затем легко объединяться с другими. Такой подход упрощает командную работу: разработчики могут параллельно экспериментировать, не мешая друг другу, и при необходимости возвращать изменения в основную ветку проекта.

Git ценят за высокую скорость, гибкость и простоту интеграции в рабочие процессы. Кроме того, это полностью открытая технология, что сделало её стандартом в мире разработки программного обеспечения.

## 2.2 Основные команды git

В данной таблице будут предоставлены все основные команды Git.

**git init** — инициализация нового репозитория.

**git config** — настройка параметров конфигурации Git.

**git status** — отображение состояния рабочего каталога и индекса.

**git add** — добавление изменений в индекс.

**git reset** — отмена изменений в репозитории.

**git commit** — сохранение изменений в локальном репозитории.

**git log** — просмотр истории коммитов.

**git push** — отправка изменений в удалённый репозиторий.

**git branch** — создание и управление ветками.

**git switch** — переключение между ветками.

**git clone** — создание копии удалённого репозитория.

**git stash** — временное сохранение изменений (откладывание «на потом»).

**git config alias** — создание псевдонимов для команд.

**git checkout** — работа с ветками, восстановление файлов и переключение на конкретные коммиты.

**git merge** — слияние изменений из разных веток.

**git fetch** — загрузка обновлений из удалённого репозитория.

**git pull** — извлечение и одновременное слияние изменений.

**git rebase** — перестройка истории коммитов (сильный инструмент, требующий осторожности).

**git diff** — просмотр различий между файлами.

**git difftool** — просмотр различий и их редактирование с помощью внешних инструментов.

**git remote** — управление удалёнными репозиториями.

**git tag** — создание и работа с тегами.

**git restore** — восстановление файлов из индекса или коммитов.

**git cherry-pick** — применение выбранных коммитов из одной ветки в другую.

**git revert** — откат изменений с созданием нового коммита.

## Выполнения лабораторной работы

### 3.1 Настройка github

Первым шагом было создание учётной записи на сайте GitHub, который будет использоваться как удаленный сервер для хранения репозитория.

Инструкция по выполнению:

1. Перейдите на сайт <https://github.com/>.

2. Нажмите на кнопку "Sign up".

3. Следуйте инструкциям на экране: введите вашу электронную почту, создайте пароль и выберите имя пользователя.

4. Подтвердите свою учетную запись через письмо, которое придет на указанную почту.

Customize your pins

Pinned

berkat1 Public

беркат копия сайта

HTML

0 contributions in the last year

Contribution settings

2025

2024

Sherst01

Edit profile

Contribution activity

September 2025

Sherst01 has no activity yet for this period.

Show more activity

У меня был аккаунт , но регистрация простая

### 3.2 Базовая настройка git

После установки **Git** на локальной машине необходимо было выполнить базовую конфигурацию: указать имя пользователя и адрес электронной почты, которые будут отображаться в истории коммитов, а также настроить другие параметры для корректной работы.

```
liveuser@GazdievAhmad:~$  
liveuser@GazdievAhmad:~$ git config --global user.name "Gazdiev Ahmad"  
liveuser@GazdievAhmad:~$ git config --global user.email "ahmadgazdievgmail.com"  
git: «config» не является командой git. Смотрите «git --help».
```

Самые похожие команды:

```
config  
liveuser@GazdievAhmad:~$ git config --global user.email "ahmadgazdievgmail.com"  
liveuser@GazdievAhmad:~$ git config --global init.defaultBranch master  
liveuser@GazdievAhmad:~$ git cinfig --global core.autocrlf inut  
git: «cinfig» не является командой git. Смотрите «git --help».
```

Самые похожие команды:

```
config  
liveuser@GazdievAhmad:~$ git config --global core.autocrlf inPut  
liveuser@GazdievAhmad:~$ git config --global core.safecrlf warn  
liveuser@GazdievAhmad:~$ git config --global core.quotePath false  
liveuser@GazdievAhmad:~$
```

Я выполнил базовую настройку **Git**. Эти команды нужны, чтобы каждый мой коммит (сохранение изменений) был подписан моим именем и почтой. Это очень важно для отслеживания истории изменений, особенно при работе в команде.

### 3.3 Создание SSH-ключа

Для безопасного подключения к **GitHub** без необходимости каждый раз вводить пароль, я сгенерировал пару SSH-ключей (приватный и публичный) и добавил публичный ключ в свой профиль на GitHub.

```

liveuser@GazdievAhmad:~$ ssh-keygen -C "Gazdiev Akhmad ahmadgazdievgmail.com"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/liveuser/.ssh/id_ed25519):
Created directory '/home/liveuser/.ssh'.
Enter passphrase for "/home/liveuser/.ssh/id_ed25519" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/liveuser/.ssh/id_ed25519
Your public key has been saved in /home/liveuser/.ssh/id_ed25519.pub
The key's fingerprint is:
SHA256:AY8eODDnV0l39LgDMVG+sXsTWWZRpG1KVRGNou1Fnqw Gazdiev Akhmad ahmadgazdievgmail
com
The key's randomart image is:
+--[ED25519 256]--+
| o o.o..o. . *X|
| * ..+o.. . o=.|
| +.o o+o= =*.o|
| o..o..++o*=o|
| ..Sooooo. |
| o.E. |
| ..o |
| . . |
+-----[SHA256]-----+
liveuser@GazdievAhmad:~$

```

Теперь давайте добавим ключ на **GitHub**:

1. Зайдем в свой профиль на **GitHub**.
2. Перейдем в **Settings** -> **SSH and GPG keys**.
3. Нажмем **New SSH key**.
4. В поле Title введем название ключа (например, "My Work Laptop"), а в поле Key вставим скопированный ключ.
5. Нажмите **Add SSH key**.

### 3.4 Создание рабочего пространства и репозитория курса

На этом этапе я создал репозиторий на GitHub на основе предоставленного шаблона, затем создал локальную структуру каталогов для учебных проектов и клонировал удаленный репозиторий на свой компьютер.

1. Я перешел на страницу репозитория с шаблоном курса:  
<https://github.com/yamadharma/course-directory-student-template>.
2. Нажал на кнопку **Use this template**.
3. В открывшемся окне задал имя репозитория (**Repository name**) **study\_2025–2026\_arch-pc** и создал репозиторий, нажав кнопку **Create repository from template**.

Теперь создадим каталог и перейдем в него:

## Выполнение самостоятельной работы

**Задание №1.** Создайте отчет по выполнению лабораторной работы в соответствующем каталоге рабочего пространства (labs/lab02/report).

```

~$ cd /home/sgritsko/work/study/2025-2026/Архитектура\ компьютера/arch-pc/labs/lab01/report
~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab01/report$ git status

```

Я перешел в каталог для сдачи лабораторной работы

**Задание №2.** Скопируйте отчеты по выполнению предыдущих лабораторных работ в соответствующие каталоги созданного рабочего пространства.

```
:~$ cp ~/Загрузки/lab01_report.pdf labs/lab01/report/report.md  
:~$
```

В данном задании, я перенес лабораторную работу №1 в каталог, который меня просят

**Задание №3.** Загрузите файлы на github.

Я выполнил самостоятельное задание. Создал файл для отчета по этой лабораторной работе и скопировал отчет по предыдущей. Все новые файлы я так же добавил, закоммитил и отправил на **GitHub**. Это закрепило мои практические навыки работы с основными командами **Git**.

## Выводы

В процессе выполнения лабораторной работы я познакомился с основами системы контроля версий Git и закрепил практические навыки её использования. Были освоены базовые настройки Git, создание и конфигурация репозитория на GitHub, а также работа с SSH-ключами для безопасного взаимодействия. Я изучил основной рабочий процесс: добавление изменений (**add**), их фиксацию (**commit**) и отправку на удалённый репозиторий (**push**). Поставленные задачи выполнены, цель работы достигнута.