**КИЇВСЬКИЙ ФАХОВИЙ КОЛЕДЖ ЗВ’ЯЗКУ**

**WORK-CASE №1**

з дисципліни «Операційні системи»

Виконали: студенти **3** курсу, групи **КСМ-13А**

**Засенко Олександр**

(прізвище та ініціали)

**Дзюбенко Дмитро**

(прізвище та ініціали)

**Сторожук Костянтин**

(прізвище та ініціали)

Київ  2023

**ЗМІСТ**

1 What is git used for, what are the main actions and commands performed in it…...3

2. What is a "commit" and how does it allow you to track changes to files………….4

3. Photo material on work…………………………………………………………….5

Conclusions…………………………………………………………………………...6

Links to original sources ……………………………………………………………..7

**What is git used for, what are the main actions and commands performed in it**

Git is a distributed version control system used for tracking changes in source code during software development. It is a crucial tool for developers and teams working collaboratively on projects. Here are some of the key actions and commands commonly used in Git:

1. **Initialization**: To start using Git in a project, you can initialize a new Git repository in a directory using the git init command.
2. **Cloning**: You can clone an existing Git repository to create a local copy of the project using the git clone command, followed by the repository's URL.
3. **Adding Files**: To stage files for commit, you use the git add command. For example, git add filename will stage a specific file, or git add . will stage all changes in the current directory.
4. **Committing Changes**: After staging changes, you commit them using git commit. It requires a commit message that describes the changes made in this commit. For example, git commit -m "Added new feature".
5. **Checking Status**: You can check the status of your repository using git status. It provides information about which files are modified, staged, or untracked.
6. **Branching**: Git allows you to work on different features or versions simultaneously using branches. You can create a new branch with git branch branchname and switch to it using git checkout branchname.
7. **Merging**: To combine changes from one branch into another, you use the git merge command. For example, git merge feature-branch will merge changes from "feature-branch" into the current branch.
8. **Pulling**: To update your local repository with changes from a remote repository, you use git pull. It fetches changes and automatically merges them into your current branch.
9. **Pushing**: After making local changes and committing them, you can push those changes to a remote repository using git push.
10. **Fetching**: To get the latest changes from a remote repository without merging them, you can use git fetch.
11. **Reverting**: If you need to undo a commit, you can use git revert to create a new commit that undoes the changes made in a previous commit.
12. **Resetting**: The git reset command allows you to move the current branch pointer to a specific commit, discarding any commits made after that point.
13. **Stashing**: When you need to save your local changes temporarily and switch to another branch, you can use git stash to stash your changes, and git stash apply to reapply them later.
14. **Logging**: To view a log of commits, you can use git log. It shows information about commit authors, dates, and commit messages.

These are just some of the fundamental actions and commands in Git. Git provides a powerful set of tools for managing source code, collaborating with others, and tracking changes over time in a software project.

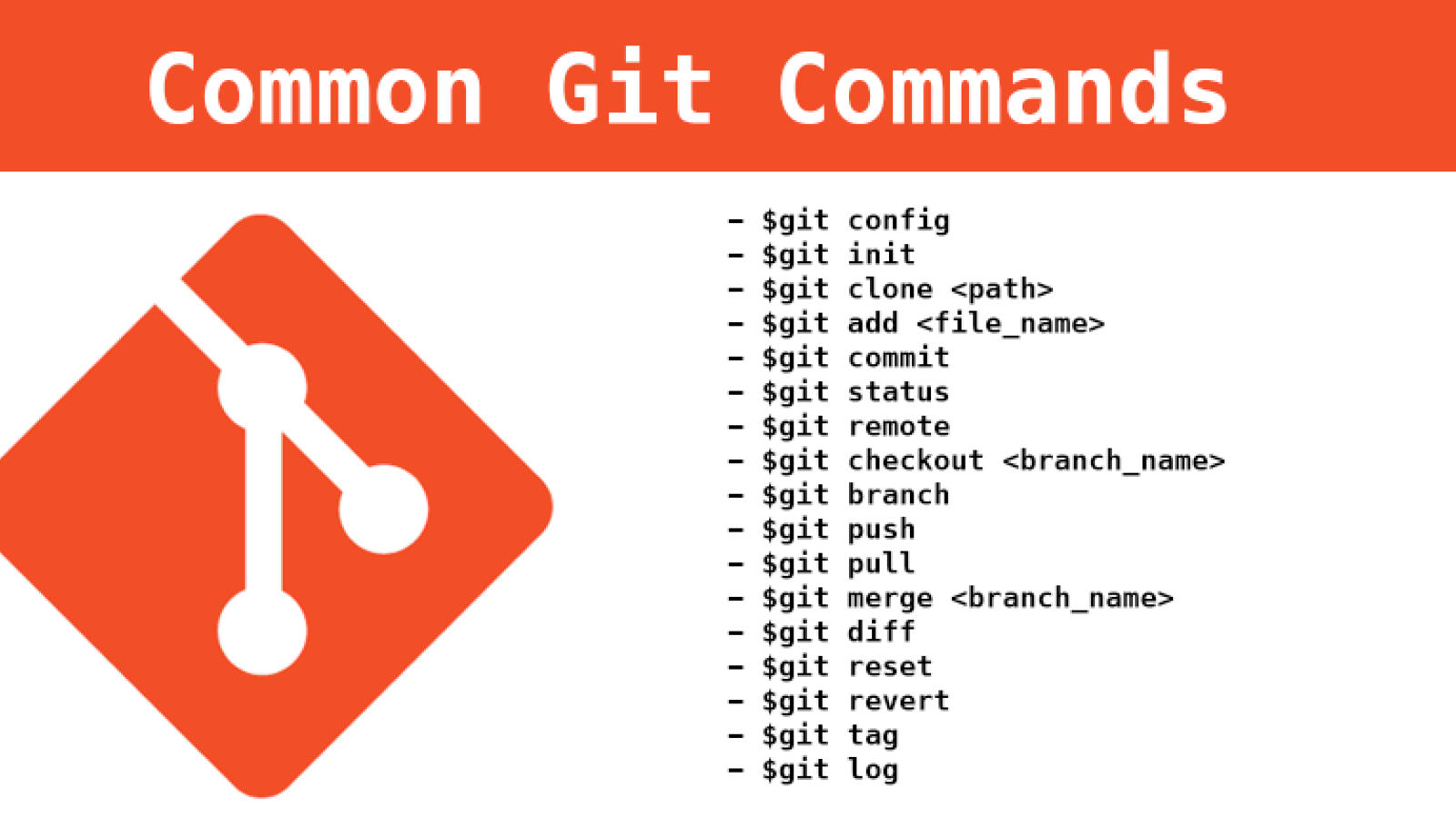
**What is a "commit" and how does it allow you to track changes to files**

A "commit" in Git is an action that allows you to save changes to your files in a repository. Each commit represents the state of your project at a specific point in time and has a unique identifier. Committing allows you to track the history of changes in your files and revert to specific points in the past when needed.

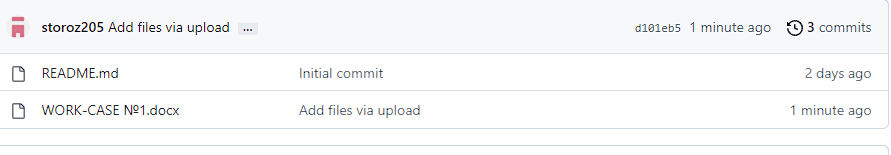
A commit includes the following components:

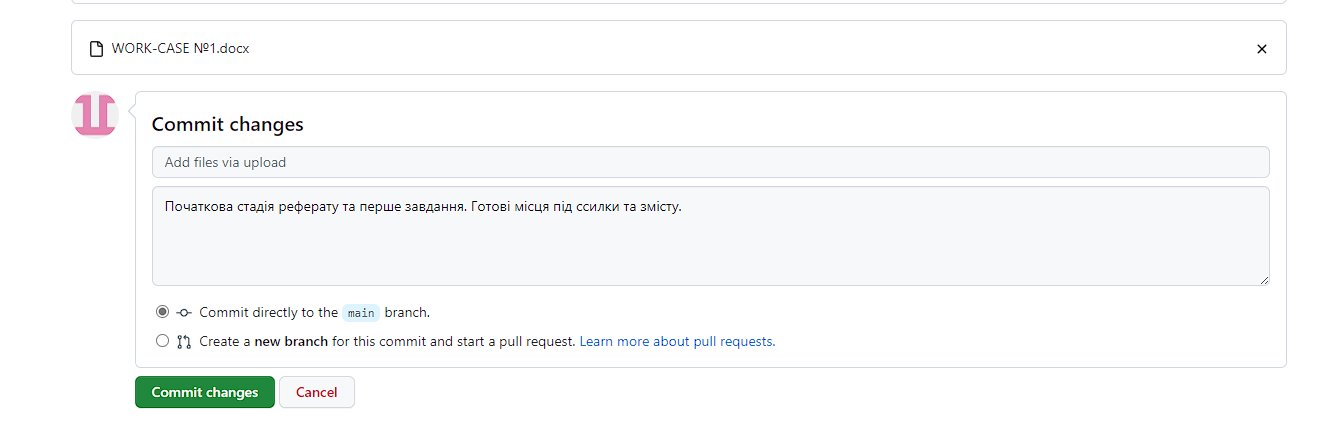
* Changes in files: All the changes you've made to your files are included in the commit, whether it's additions, deletions, or modifications to file content.
* Comment: For each commit, you can add a short comment that explains what was done in that commit. This comment is crucial for understanding the project's history.
* Unique identifier: Each commit has a unique hash code that identifies it in the Git system.

With commits, you can track when and what changes were made in your project. You can also use commits to create branches, merge branches, undo changes, and work with the project's history. This makes Git a powerful version control and collaboration tool for software projects.



Pic.1 Common git commands

Pic.2 Creating a commit

****

Pic.3 Added commit

**Conclusions**

For this work, we learned to work with such a resource as GitHub. We created our folder and commit and took turns doing the task of uploading files to the folder.

1. We described what GitHub is used for and what basic commands and actions are performed there.
2. We explained what a commit is and showed how we committed files.

As a result, we learned to work with GitHub and work in a team.

**Links to original sources**

1. aishwaryaital [Electronic resource] – Resource access mode:

https://aishwaryaital.hashnode.dev/day-8-basic-git-github-for-devops-engineers

1. medium [Electronic resource] – Resource access mode:

https://medium.com/@ujjainee.office/interview-questions-on-git-145a2b2c008e

1. w3schools [Electronic resource] – Resource access mode:

<https://www.w3schools.com/git/git_commit.asp>

1. GitHub [Electronic resource] – Resource access mode:

<https://github.com/>  
<https://github.com/FlvyDima/bashkidsclub>