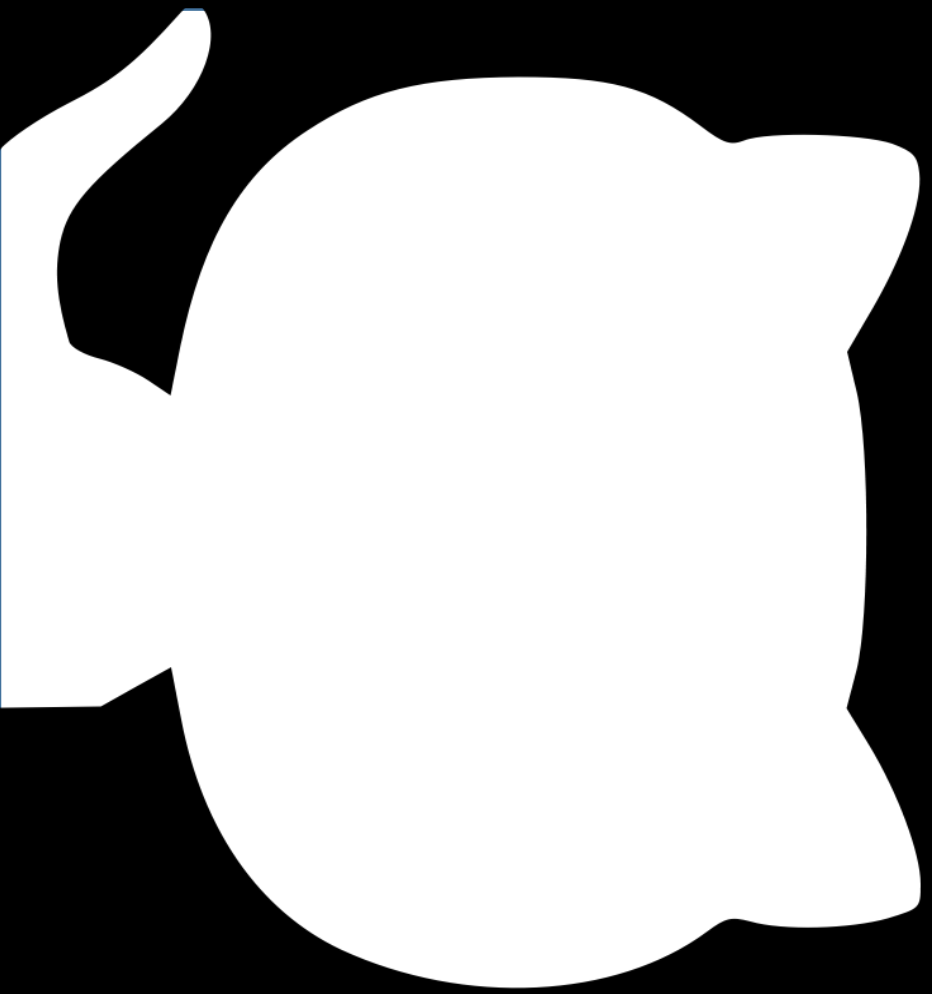
# WORK – CASE 1



Git is a distributed version control system that allows you to track software development history and collaborate on complex projects from anywhere in the world.

# 10 Git Commands You Should Know

**git init.** Allows you to create a new Git repository.

**git clone.** Creates a local working copy of source code from a remote repository.

**git branch.** Allows us to add a new branch to an existing branch, view all existing branches, and delete them.

**git checkout.** Allows you to switch to an existing branch or create and switch to a new branch.

**git add.** Adds your changes to the file to the workspace where you can compare your local version with the version in the remote repository.

**git commit.** Saves changes to your local repository. Whenever you commit code changes, you should include a brief description of the changes made. This commit message helps others understand the changes you've made.

**git push.** Sends committed file changes from a local repository to a remote repository for others to use. It will also create a named branch in the remote repository if it doesn't exist.

**git pull.** Gets the latest uploaded changes from the remote server to the local repository so you can get the latest updates from your teammates.

**git merge.** Merges your branch with the parent branch.

**git status.** Provides an overview of the current state of your storage.

# What is a git commit command?

A commit is a snapshot of your entire repository at a specific time. When you create a commit, it makes a record of exactly how all the files and directories looked at the moment of creation. Git commit has two phases. First you use git add to add the files you want to appear in your commit. Before adding those, you can run git status to check which files have changes and look through them. To add all the changes, you can run git add . (with a dot at the end).

Once you're ready to commit, run git commit -m "your descriptive commit message" explaining what was completed in that commit.

The commit message is required and important for painting a clear history of what was done in the repository over time. There are several best practices concerning writing a good commit message.

Capitalize the first letter of the commit, i.e. "Fix bug with global search input deleting user text"
Keep it short. Try to keep your message under 50 characters
Focus on what you have changed, rather than why you changed

# REGISTRATION


**How to register on GitHub?**

Just go to https://github.com, choose a username that no one else has yet, provide an email address and password, click the big green “Sign up for Github” button.

# CREATION OF REPOSITORY

1. In the upper-right corner of any page, use the + drop-down menu, and select New repository.

2. Optionally, to create a repository with the directory structure and files of an existing repository, select the Choose a template dropdown menu and click a template repository. You'll see template repositories that are owned by you and organizations you're a member of or that you've used before. For more information, see "Creating a repository from a template."

3. Optionally, if you chose to use a template, to include the directory structure and files from all branches in the template, and not just the default branch, select Include all branches.

4. Use the Owner dropdown menu to select the account you want to own the repository.

5. Type a name for your repository, and an optional description.

6. Choose a repository visibility. For more information, see "About repositories."

To perform this WORK CASE 1, it was necessary to research GitHub, its main commands, create a repository, learn what a commit is and how to work with it.