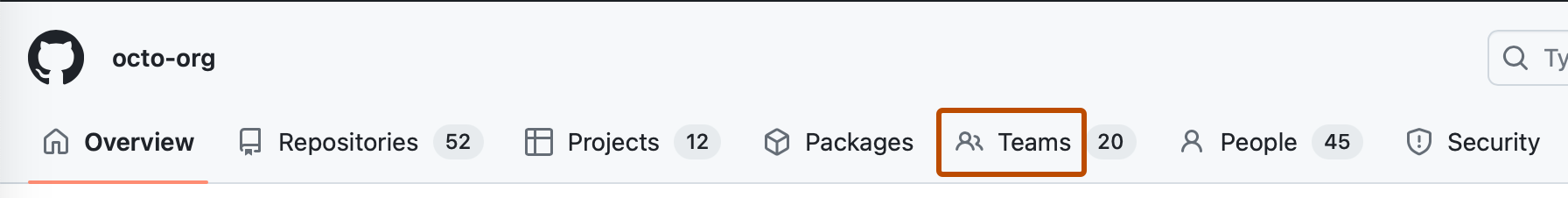
# Creating a Working Repository

* Or by creating and organization and a team inside
* Or by just creating a working repository by one of the team members

Choose one of the ways: you can try both of them and then choose.

# Option1: Creating a working repository (by one of the team members) by creating an organization and a team inside

1. In the upper-right corner of GitHub, select your profile photo, then click  **Your organizations**.
2. Create an organization – it could be “your own” or of some organization, like  
   “fullstack-ashquelon-carmiel-team”
3. When the organization is ready, click the name of your organization.
4. Under your organization name, click  **Teams**.



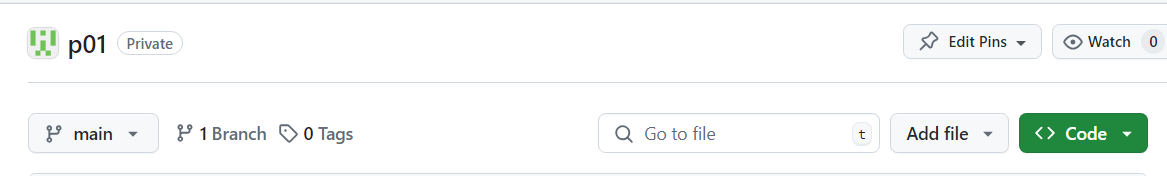
1. At the top of the page, click **New team**.
2. Under "Create new team", type the name for your new team.
3. Optionally, in the "Description" field, type a description of the team.
4. Click **Create team**.
5. Invite members to the team
6. Optionally, give the team access to organization repositories. For more information, see "[Managing team access to an organization repository](https://docs.github.com/en/organizations/managing-user-access-to-your-organizations-repositories/managing-repository-roles/managing-team-access-to-an-organization-repository)."
7. Now, one of the team creates new repository (better not empty, with readme.md file) in the organization and invites the other members of the team
8. They got the invitations and add themselves to the repository

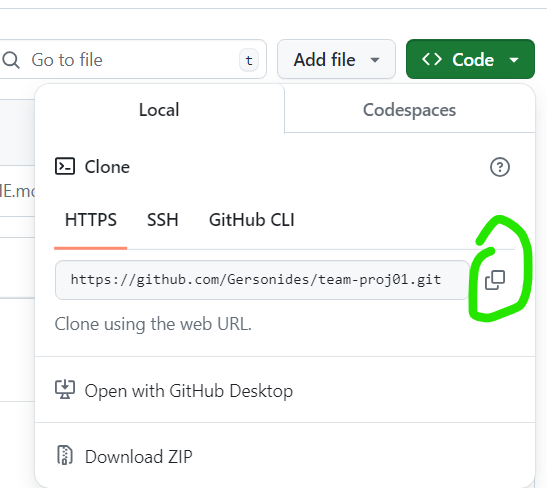
# Option 2: Just creating a working repository by one of the team members

1. In your account create new repository (better not empty, with readme.md file)
2. You can choose it to be private or public
3. Now you go to Settings -> Collaborators
4. Add collaborators
5. They got the invitations and add themselves to the repository

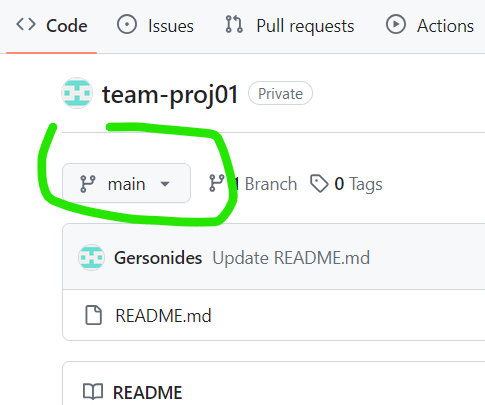
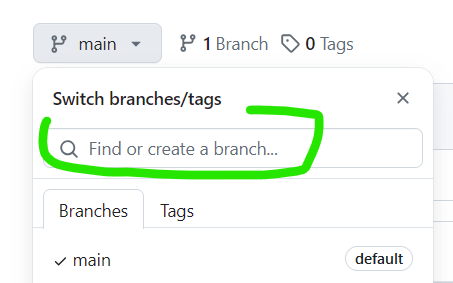
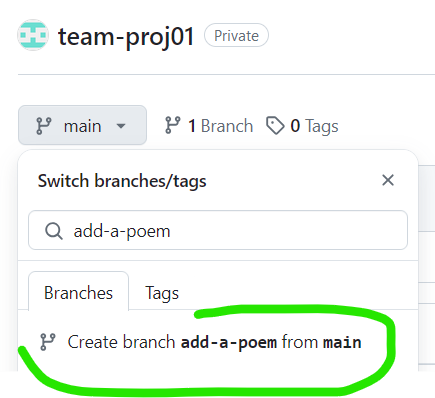
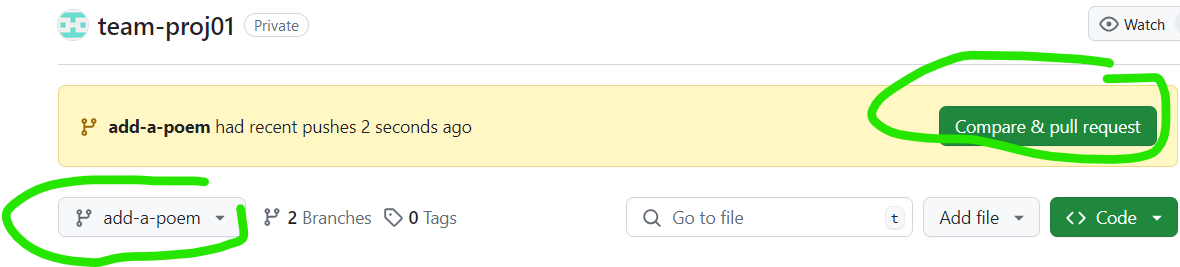
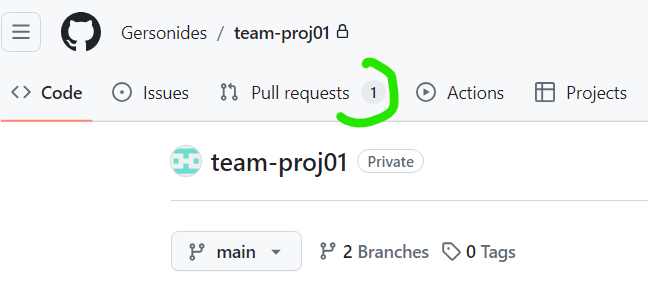
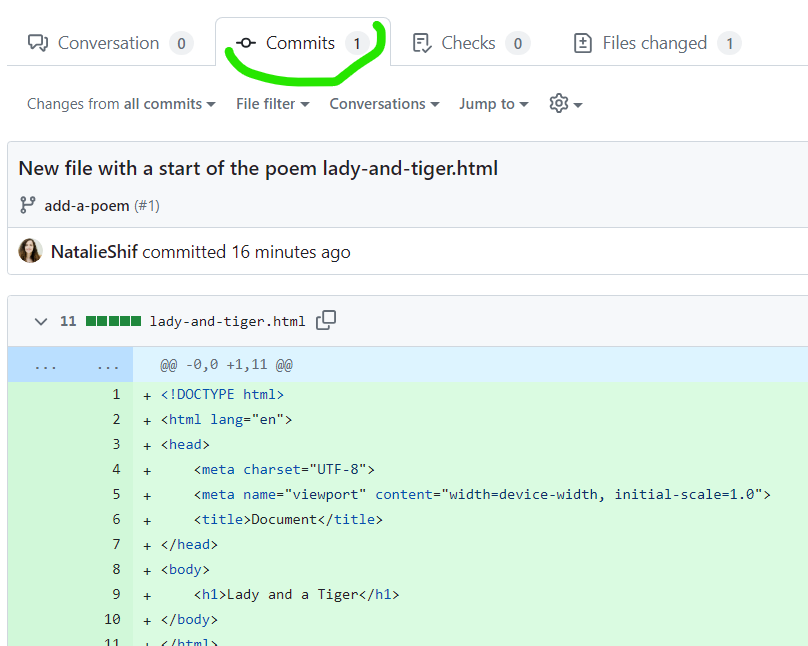
# Cloning the repository

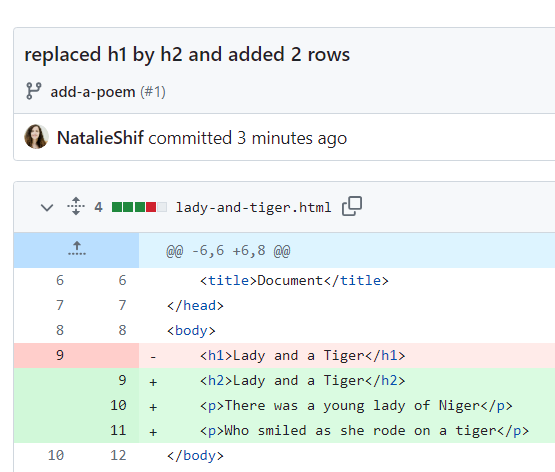
1. Be sure that you’ve got smth in your GitHub repository, for example readme.md file.
2. If there is none, just create it and write a couple of words about your project. If you’ve already has one, anyway perform there some change – just to make sure when you will clone it – no click the “Commit”, write the commit message and “Commit directly to the main branch”
3. Each collaborator creates on her/his machine a general dir for the project (for all it’s branches)
4. Opens Git Bash terminal and goes to this dir
5. Now, in the browser the collaborator goes to the “Code” button of the repository:



1. Copies the link to the repository  
   
2. And in the git bush terminal:  
     
   $ git clone copied-link.git
3. It is possible, that you’ll got an error, that it is not a “git” directory, so you initialize it first with:  
   $ git init  
   and then  
   $ git clone copied-link.git  
   $ git remote add origin copied-link.git
4. $ ls
5. Now you see that you’ve got the dir of the project on your machine  
   or it’s contents
6. $ cd cloned-proj-dir (if you see the dir, but if you see the contents – you’re already there)
7. Now you see that you’re inside “main” branch
8. If you are not, change the branch to “main”:  
     
   $ git branch -m “main”
9. $ ls
10. You see the readme.md file
11. Try the useful command   
    $ git status  
      
    It shows now:  
    “ON branch main  
    your branch is up to date with branch ‘origin/main’  
    Nothing to commit, working tree clean”
12. Try other useful command  
    $ git list  
      
    It shows the commits in the order from the last to the first.  
      
    You could see now the commit of README.md file  
    and that before it - there was an initial commit

# Using branches

1. We prefer not to work on the “main” repository, but use the next workflow:  
     
   a) create a branch  
   b) work on it  
   c) pull request  
   d) work on the reviews  
   e) merge all pull requests  
   f) the branch now is obsolete – we can delete it
2. So, now we create a branch, and we could do it locally or remotedly, but we’ll work on it locally
3. Option1. Creating the branch remotedly and working on it locally  
     
   On the page of the repository go to the list of the branches:  
   
4. In “find or create a branch” you write the name of the new branch:  
   
5. Something meaningful, like add-a-poem, and now you’ve got a button “create branch add-a-poem from main”  
     
   
6. Push the button. Mazal tov! You’ve got a new branch!
7. Now you go to your machine Git Bash terminal and perform:  
     
   $ git pull  
     
   … new branch  
   … up to date
8. You switch to this new branch:  
     
   $ git checkout add-a-poem  
     
   Switched to a new branch “add-a-poem”  
   branch “add-a-poem” set up to track origin/add-a-poem
9. Also you see that you’re working in this branch by the turquoise brackets that contain (add-a-poem)
10. One more ways to ensure on which branch you’re working now and which branches are there at all:  
      
    $ git branch  
      
    \* add-a-poem  
    main  
      
    This one with asterisk (\*) – is your current branch
11. Now you start working in this dir. Just open it with VS CODE, for example:  
      
    $ code .
12. Create my-new-poem.html with some rows inside and in the terminal check status:  
      
    $ git status
13. Now we see “untracked” red that says that some changes – like the new file – are unknown to git. So we add them to the “stage”:  
      
    $ git add \*  
      
    $ git status
14. Now it sees the changes, but they’re uncommitted, so we commit them with some meaningful message, like:  
      
    $ git commit -m”New file with a poem my-new-poem.html”  
      
    $ git status
15. You should get a message that your local branch of add-a-poem is 1 commit ahead of the remote branch add-a-poem, which is right.  
      
    By the way  
      
    $ git log  
      
    Now is showing your last commit too
16. Let’s update the remote branch:  
      
    $ git push
17. Go to GitHub  
      
      
    Be sure you’re on the right branch – no you click compare and pull request
18. You fill the title and the description and click the button of “create pull request”
19. Now each member of the team will see, that there some pull requests:  
      
    
20. When you click on the “Pull requests” tab you see the list of pull requests.  
      
    Click on the pull request you’d like to review
21. Click on the “Commits” tab and go to the last commit, now you see the lines of code:  
      
    
22. Click there on the + of the line that you’d like to comment/review, and now you could comment or start review:  
      
    A screenshot of a computer

    Description automatically generated
23. Explore the options, you can continue the review, you can finish it and submit, you can request that your review will be addressed before the merge, etc.
24. So, let’s say we submit the review.
25. Now everybody on the team can see the conversation – so please add your comments too, and the owner of the branch adds the comments also – so we’ve got a conversation
26. There are plenty of options: for example, we could request changes, resolve conversation, etc.
27. Let’s say the person who’s changes we’ve reviewed performs one more change, git add \*, git commit -m”meaningful comment” and git push
28. Now everybody who reviews click on the pull request and sees there a new commit.   
      
    There in pink we see the old code and in the green – the new one
29. Again, the reviewers and the coder could manage a conversation, approve the changes, etc. When it’s decided, the person who wrote the code, goes to “Pull Request” clicks on MERGE
30. Now we could delete the merged branch “add-a-poem”. For example, from the local machine (first switch to the main branch):  
      
    $ git checkout main  
    $ git pull  
    $ git branch -d add-a-poem  
    $ git push origin --delete add-a-poem
31. Every team member can ensure now that there is only one branch left and that there are the changes that each team member has performed

# Issues – additional way to collaborate

1. You’ve got an idea or see a problem. Create an issue
2. Now click on ‘Issues’ – assign yourself to an issue that smb else has created
3. We can manage our project through the list of Issues – what should be done

<https://www.atlassian.com/git/tutorials/undoing-changes/git-reset>

great tutorial

<https://medium.com/@jake.page91/the-guide-to-git-i-never-had-a89048d4703a>

the guide to git I never had