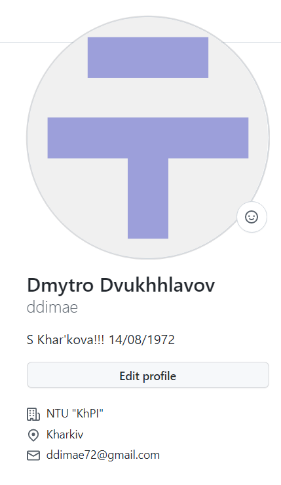
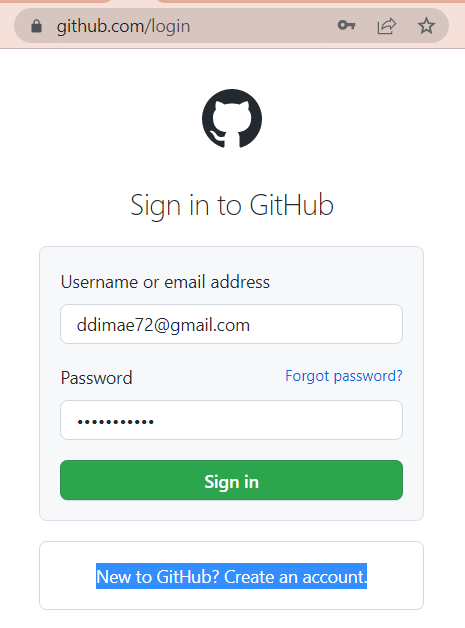
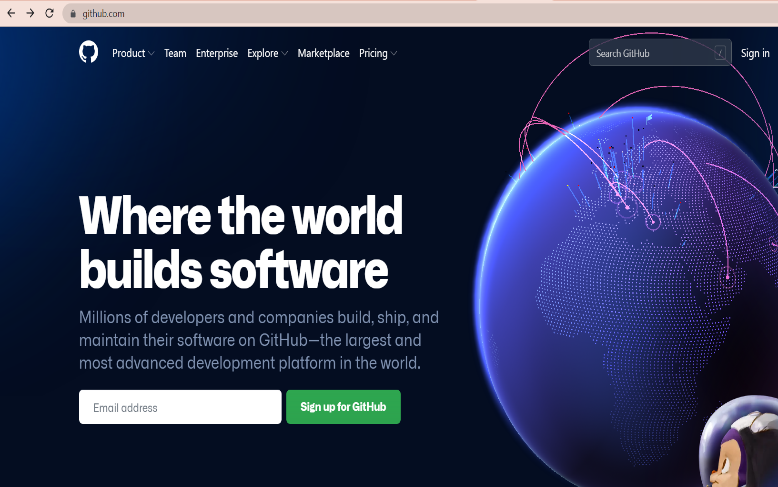
**Scenario LR-GIT5**

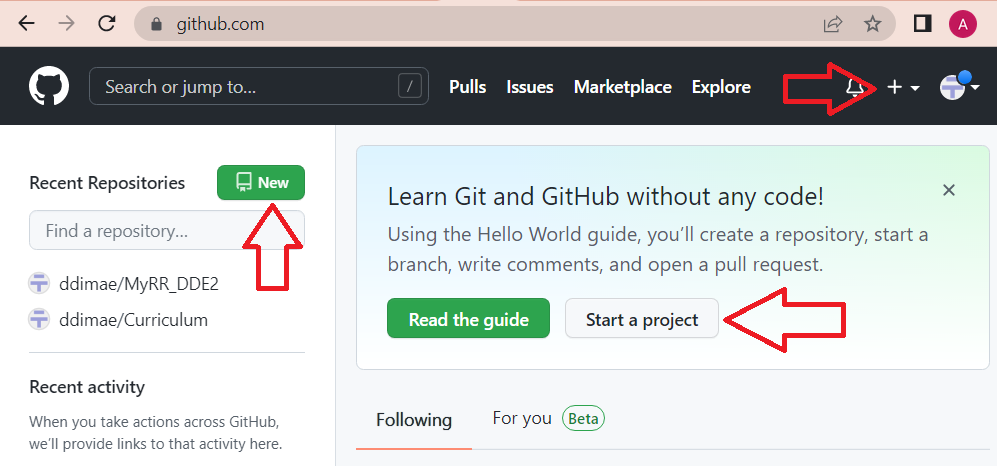
1 Create account on GitHub

Result on left screenshot.



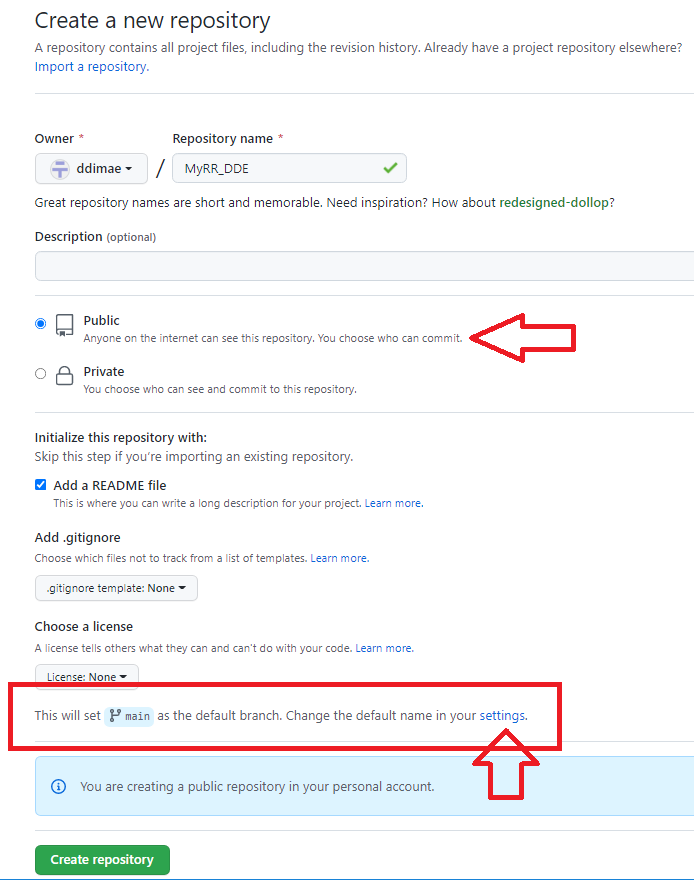
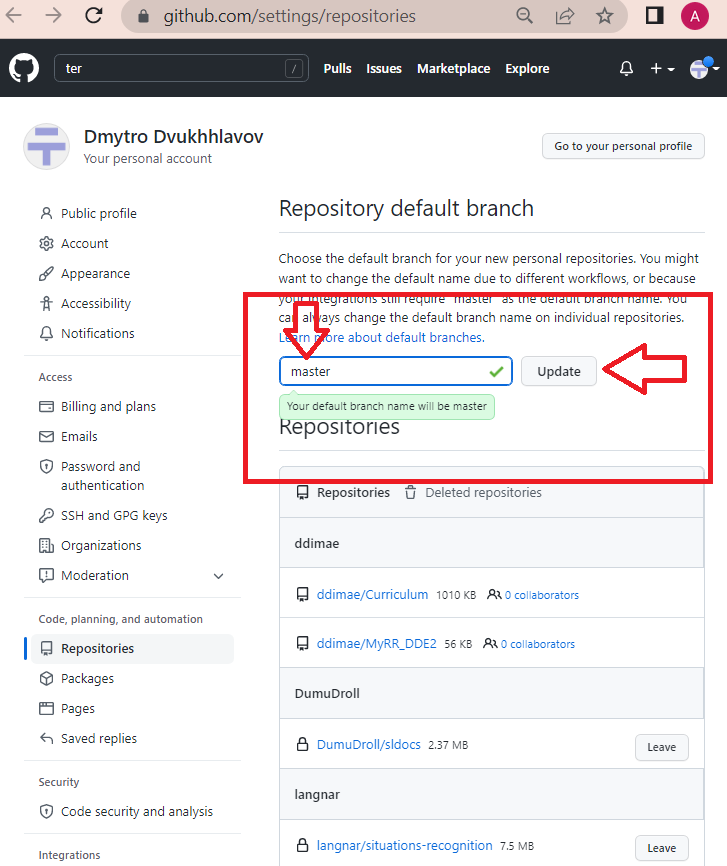
2 Create remote repository in your account with name “MyRD\_FIO”. For example, MyRR\_DDE.

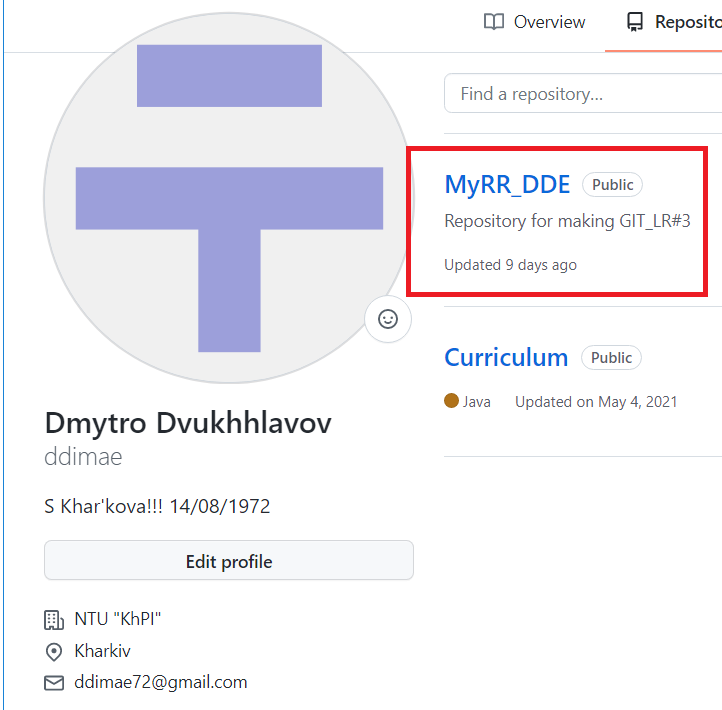
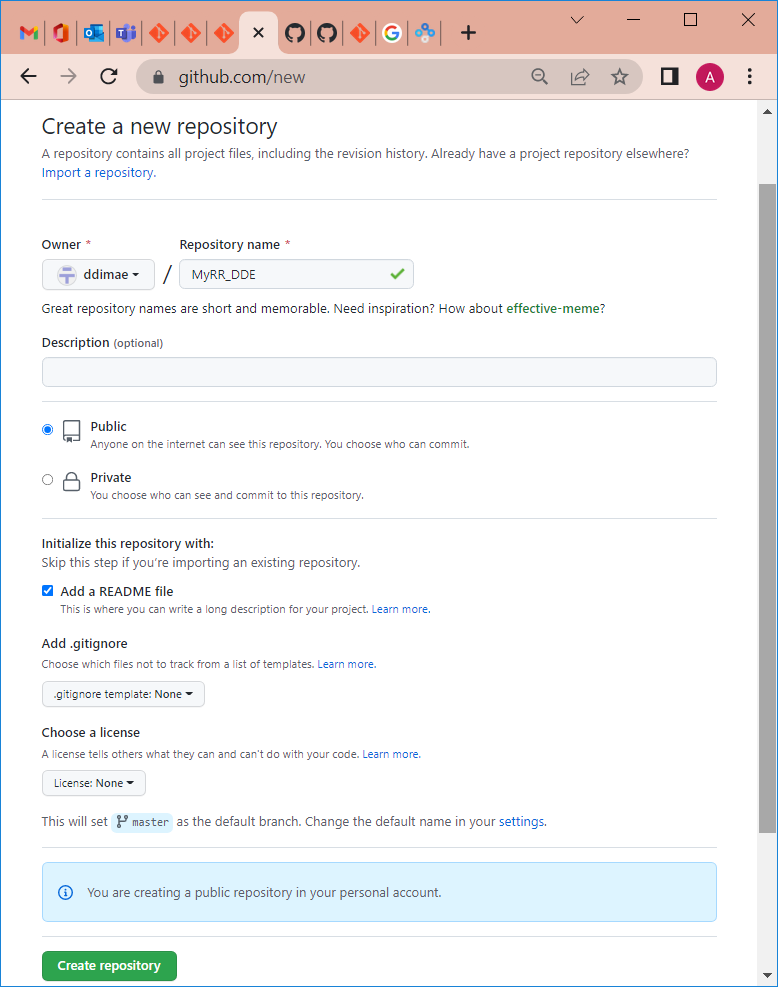
Arrows on screen – different variants of go to create new repository



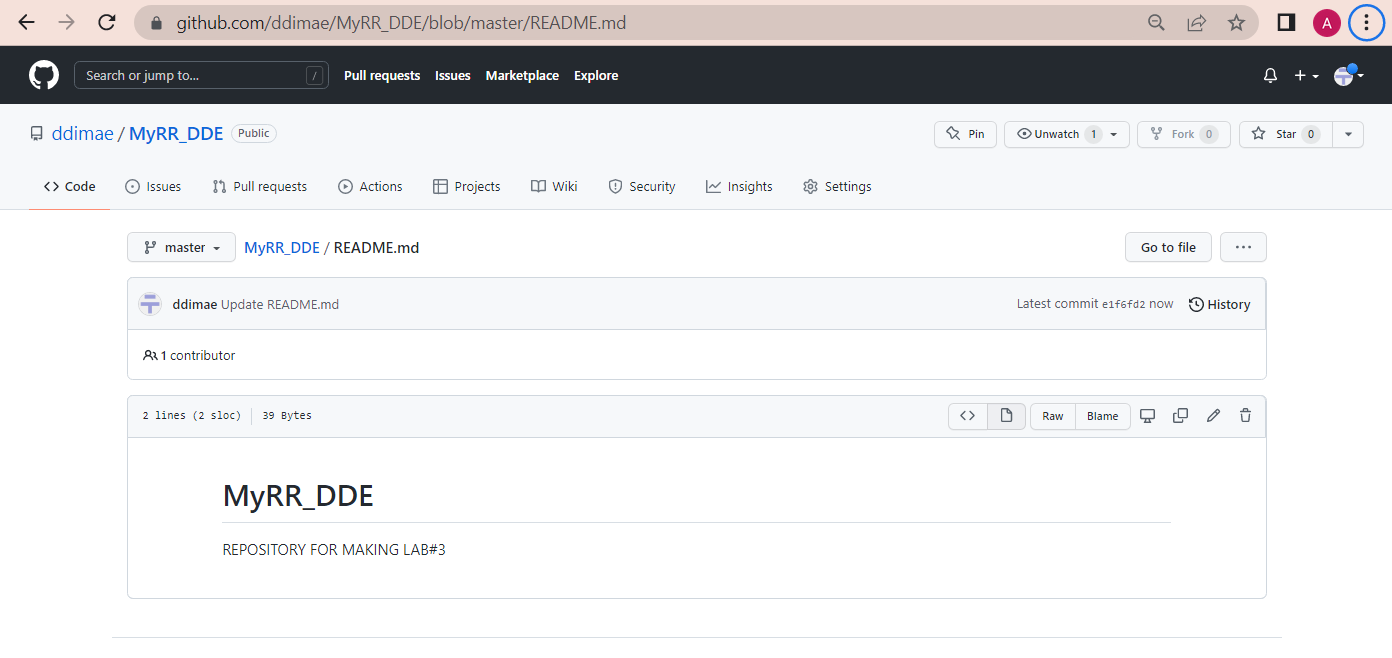
Important! Good results (or less problem) you will get if main branches in the remote and in the local repository will has same names. By default in git-Bash was set **master**, for GitHub – **main**. I was set name master for GitHub (look screenshots).

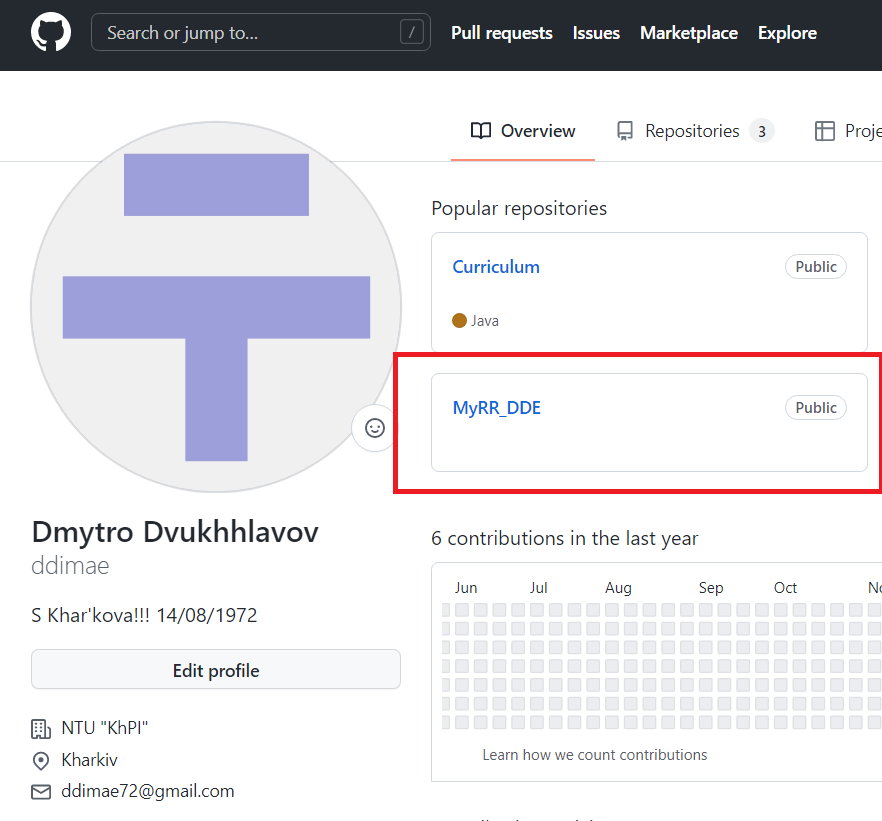
Also exist other way to solve this problem, but I not say about it – find in Google!



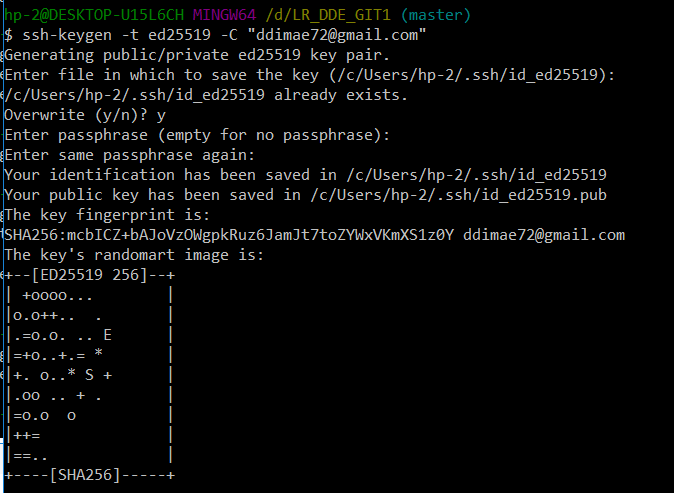
After done creating I get empty repo with Readme file.



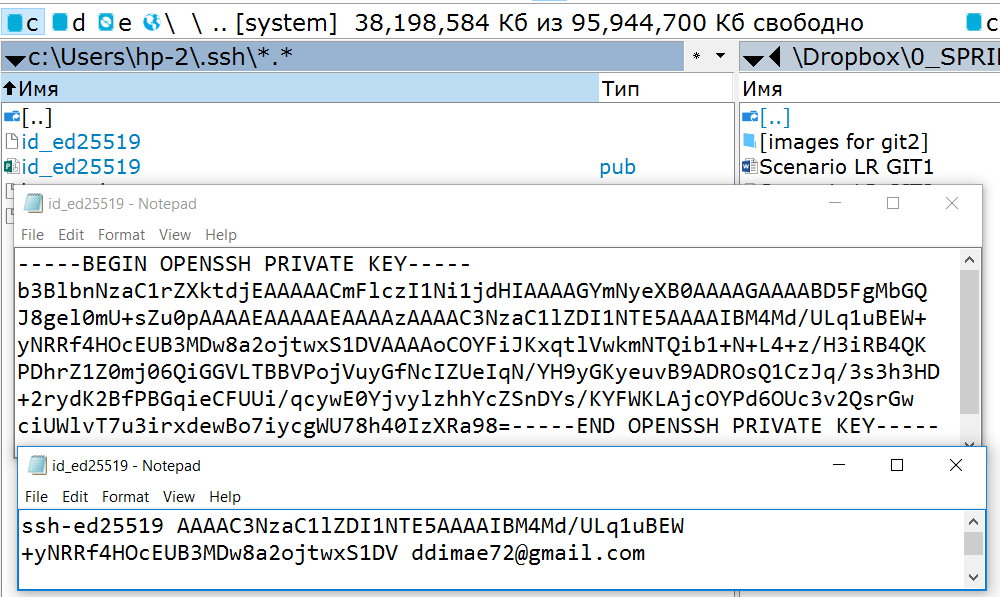


3 Generate ssh-key and integrate it with GitHub.

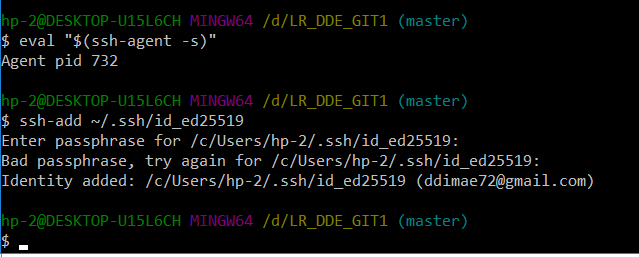
Attention! There exist many methods to create ssh-key. I used ed25519 from [Generating a new SSH key and adding it to the ssh-agent - GitHub Docs](https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent). You can find also variant of the key creating using rsa or dsa.



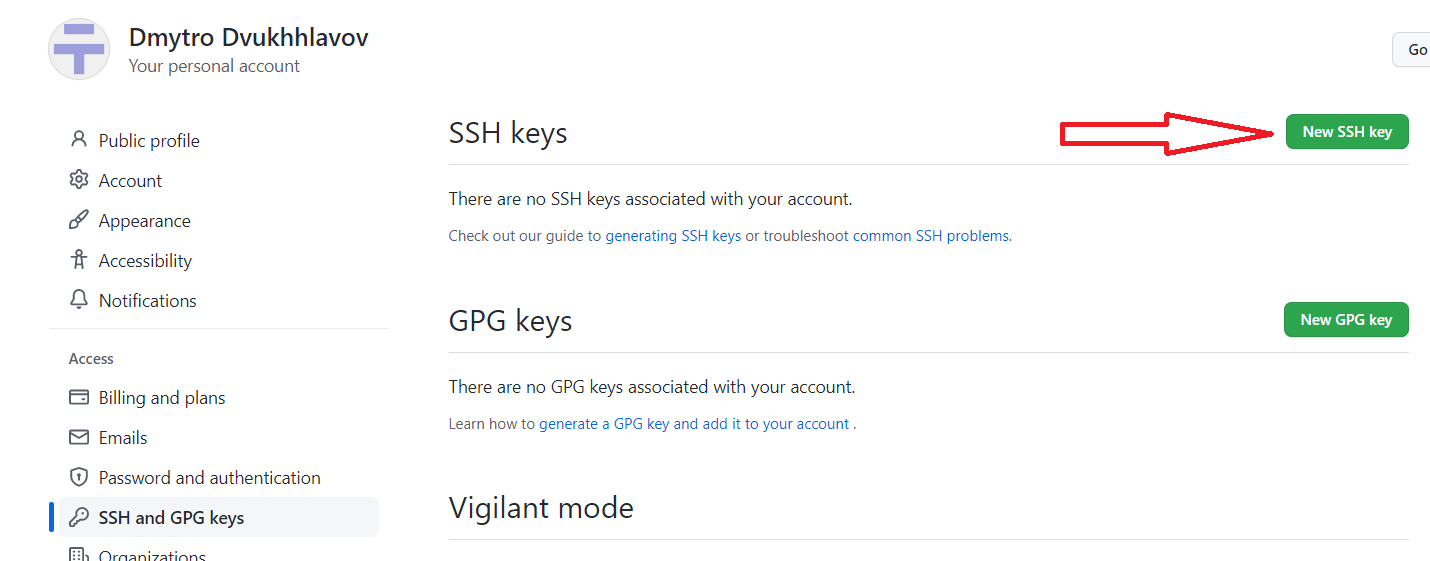
Then you must go to created directory and find generated key.

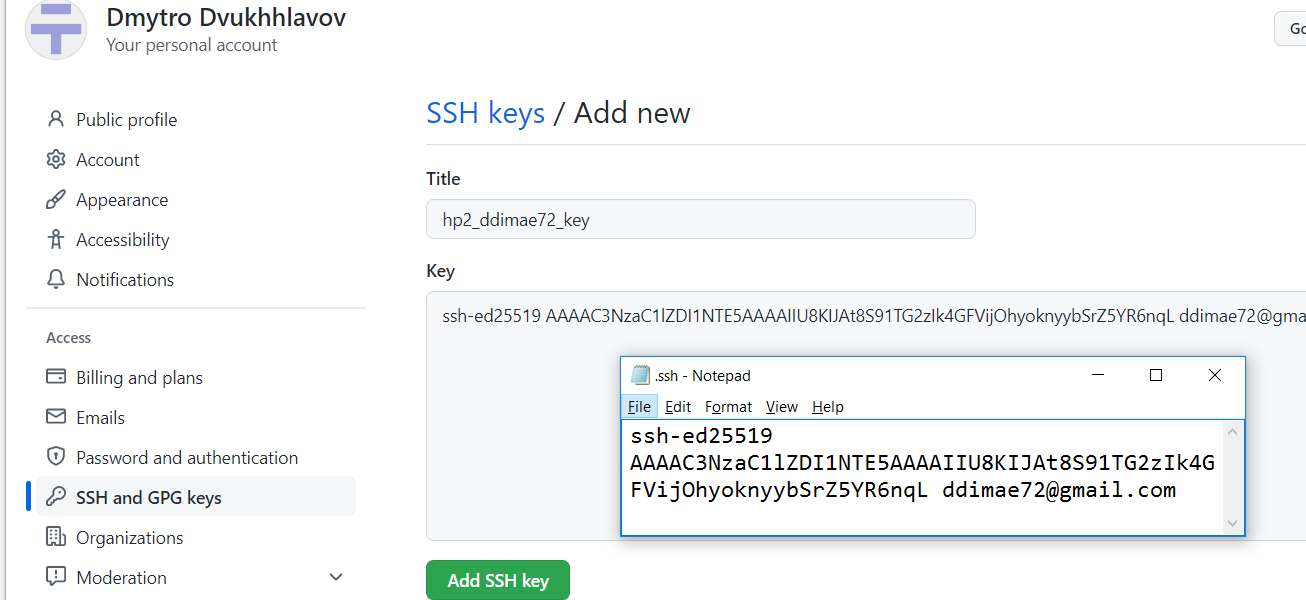


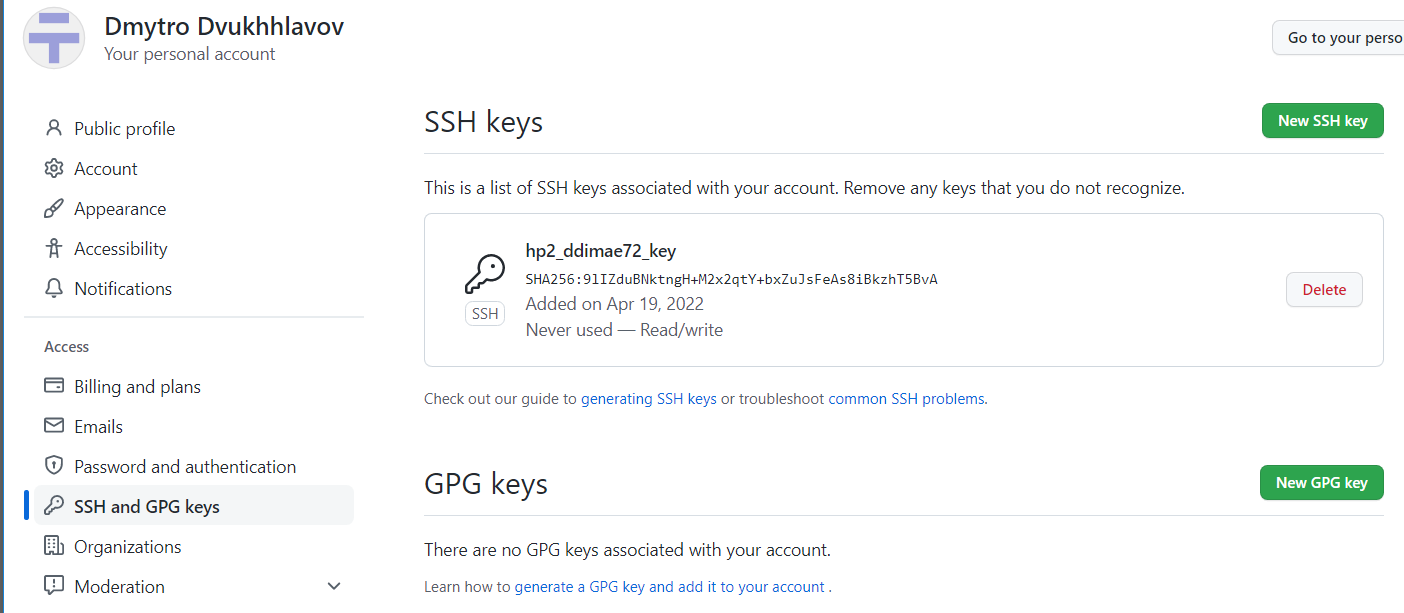
Start ssh-agent and add ssh-key to ssh-agent.



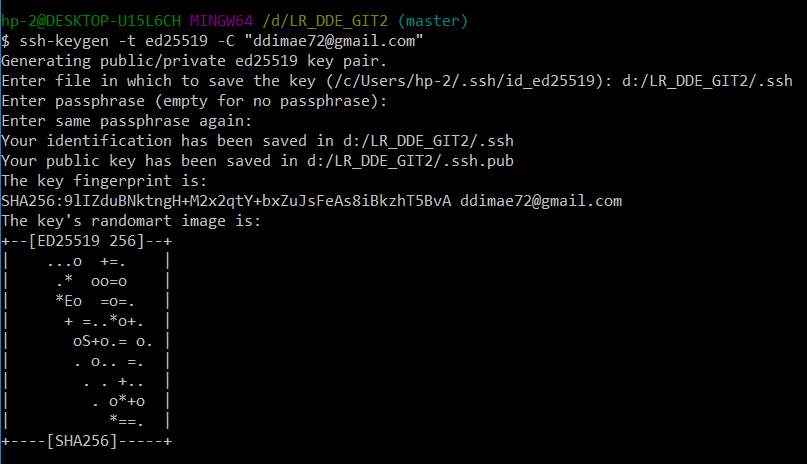
Go back to the your git-profile and add generated SSH – set name and copy text in the file **.ssh.pub**. Also GitHub send you the message about generating of the key.





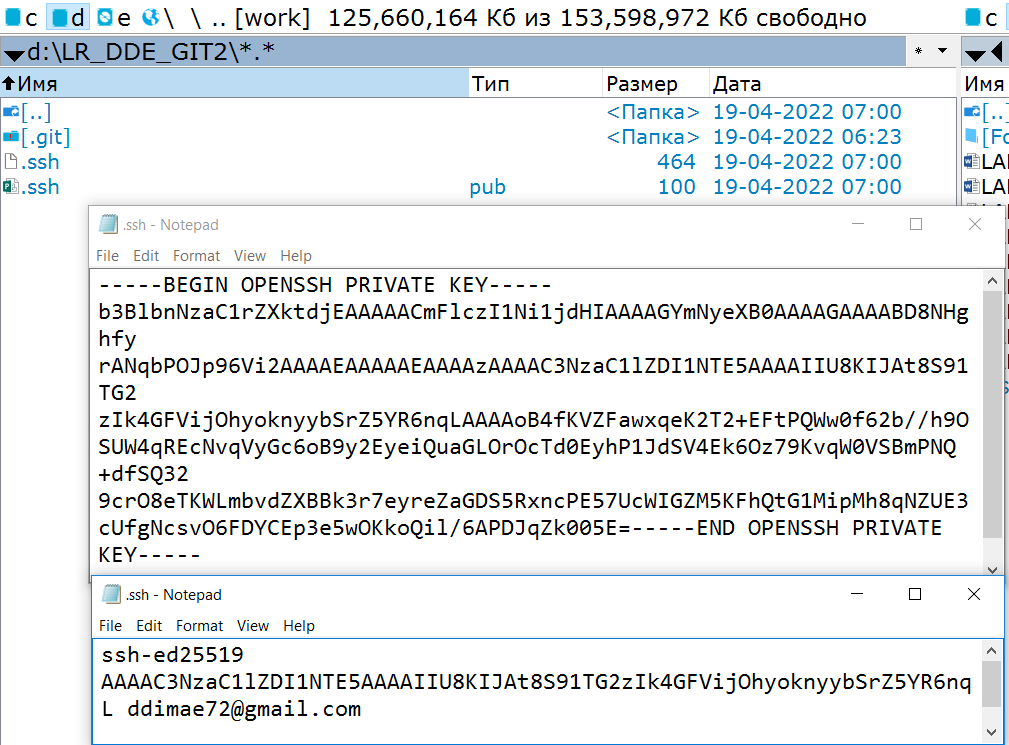


/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

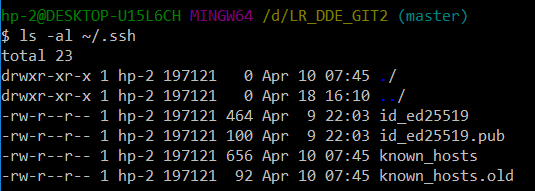


I am change the place of storing ssh-keys. By default, they store in user’s Windows folder.

Then you must go to created directory and find generated key.

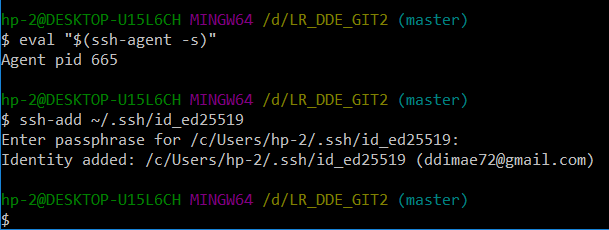


Checking of existing keys.



Yes, key pair is exist.

Start ssh-agent and ssh-key to ssh-agent.

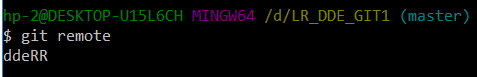


\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

4 Linking your local and remote repository.

<https://git-scm.com/book/en/v2/Git-Basics-Working-with-Remotes>

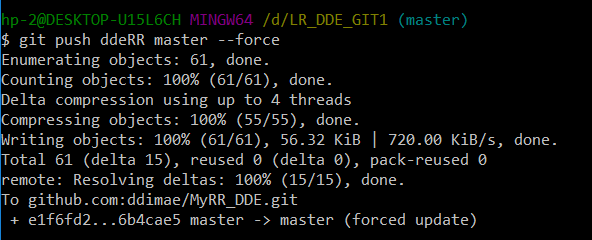




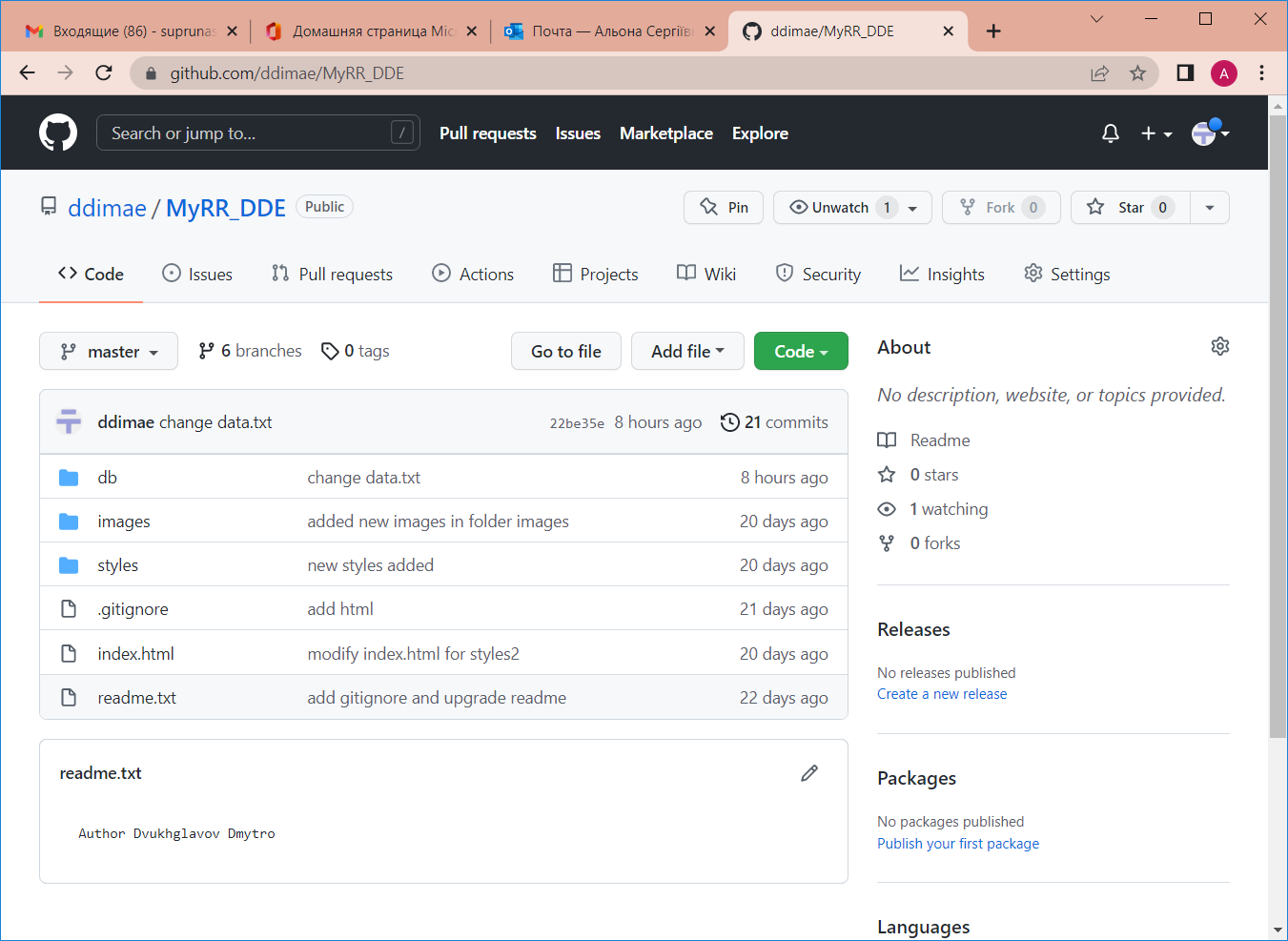
5 Push all your changes with all your branches to remote repository ($ git push <remote repository name> --all).

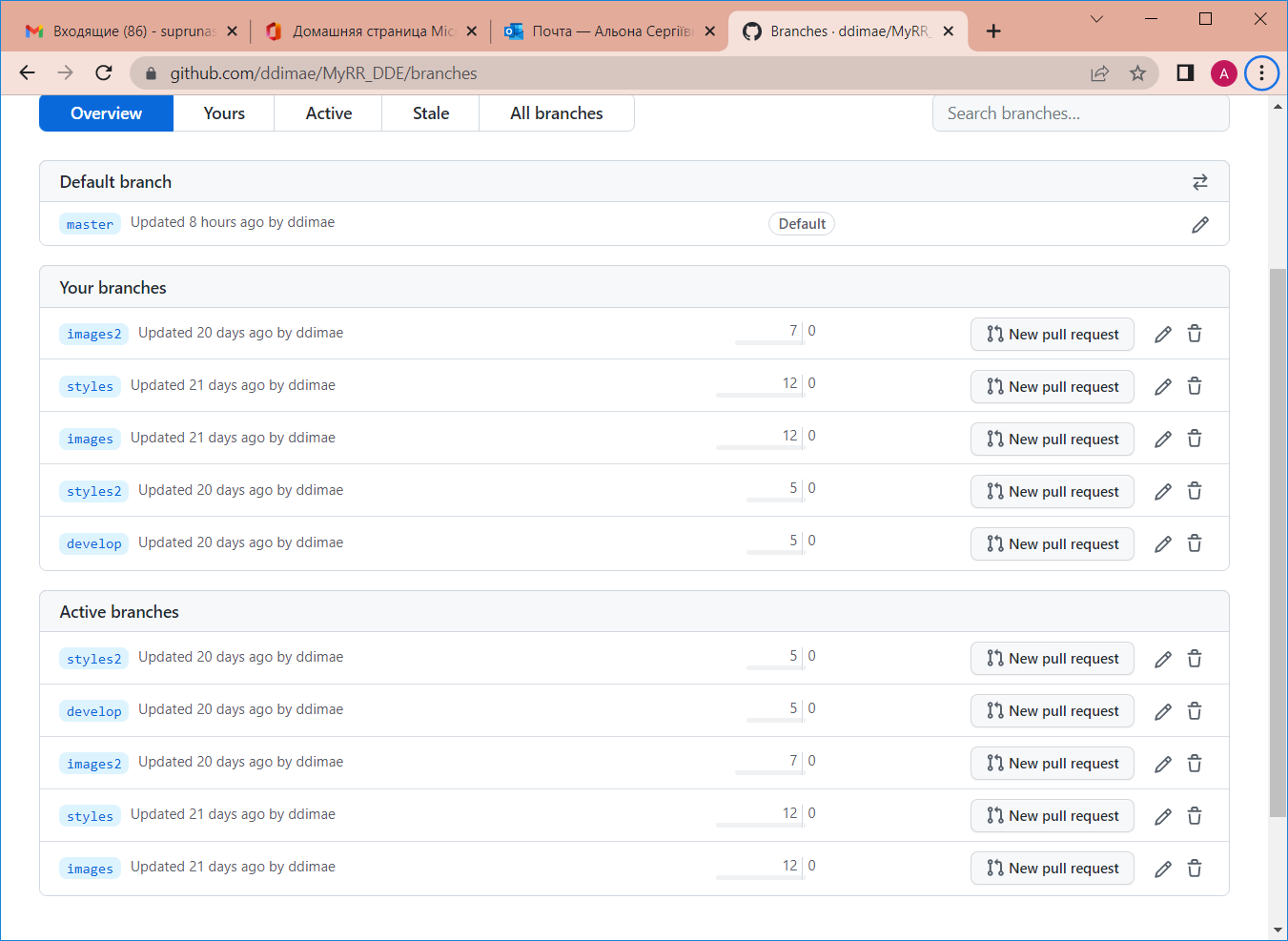


Problem… After creating repo by Github their appear some files. For push in this repo version look like in local repo must be done force pushing – with option --force.



6 Check loading your working repository to remote repository by used browser.





As result, you have copy of your local repo with all commits stored in GitHub.

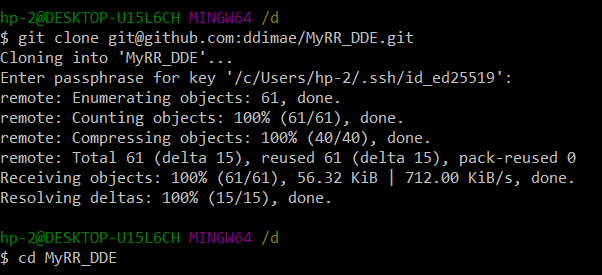
**Task Git-6. Update project and save changes in the remote repository**

This task allow training skills in saving changes in remote repository without the consent of other developers.

1. Create folder with name “LR\_FIO\_GIT2” (example - “LR\_FIO\_GIT2”) and clone project in this folder.

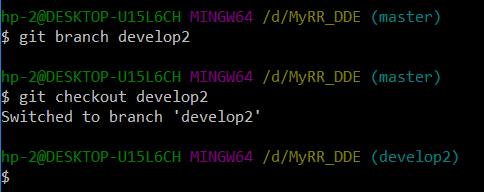
mkdir d:/LR\_DDE\_GIT2 or by use Explorer or other program.





Look at last command! After clone in your folder has been created folder with name of the remote repository. For start work with current version of project it is necessary move into this folder. If you input command ls (or you Explorer), you will see same folder and files as in the repo on GitHub.

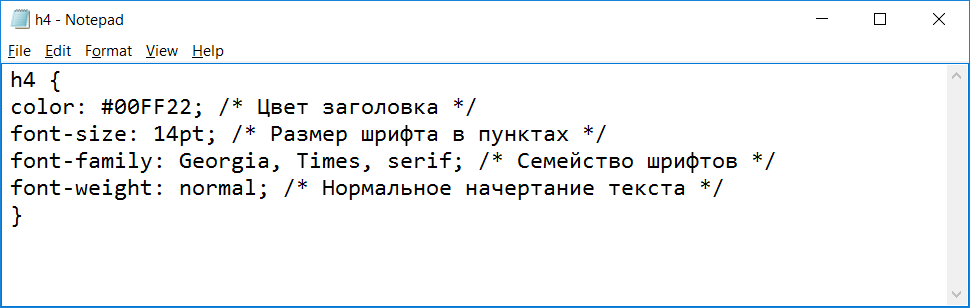
1. Open git Bash. Make upgrade of your project - implements new feature. Before this the branch *develop2* has been created. Checkout on it.

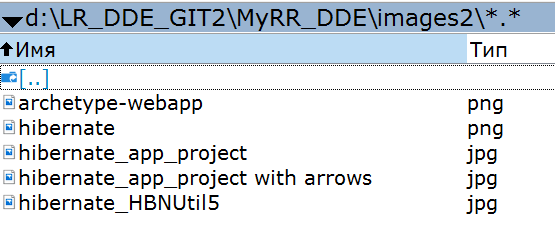


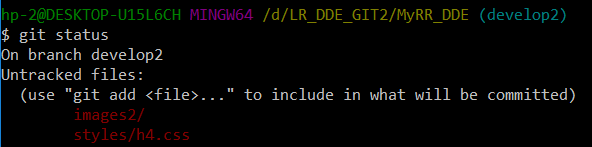
1. Make 2-3 commits during development new feature.

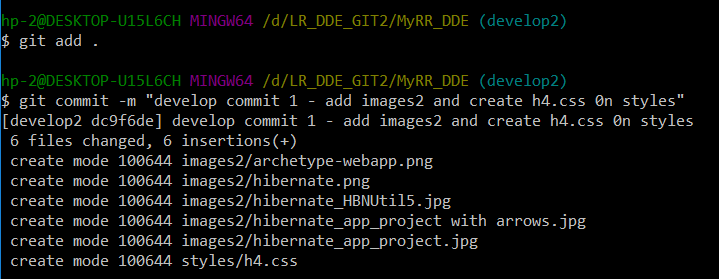
I done:

* Create h4.css and folder image2 (commit 1);

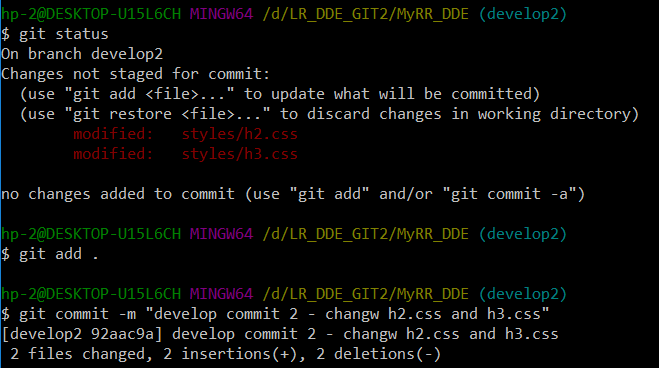




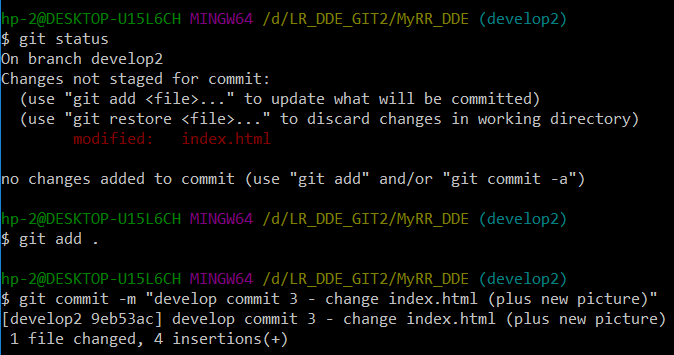




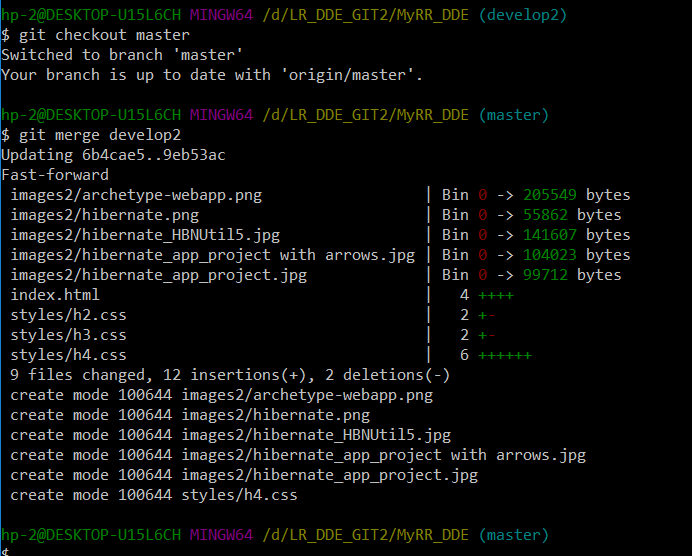
* Change h2.css and h3.css (commit 2);



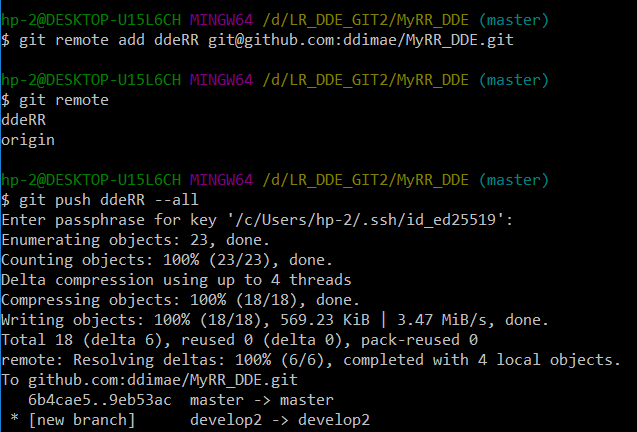
* Change index.html (commit 3).



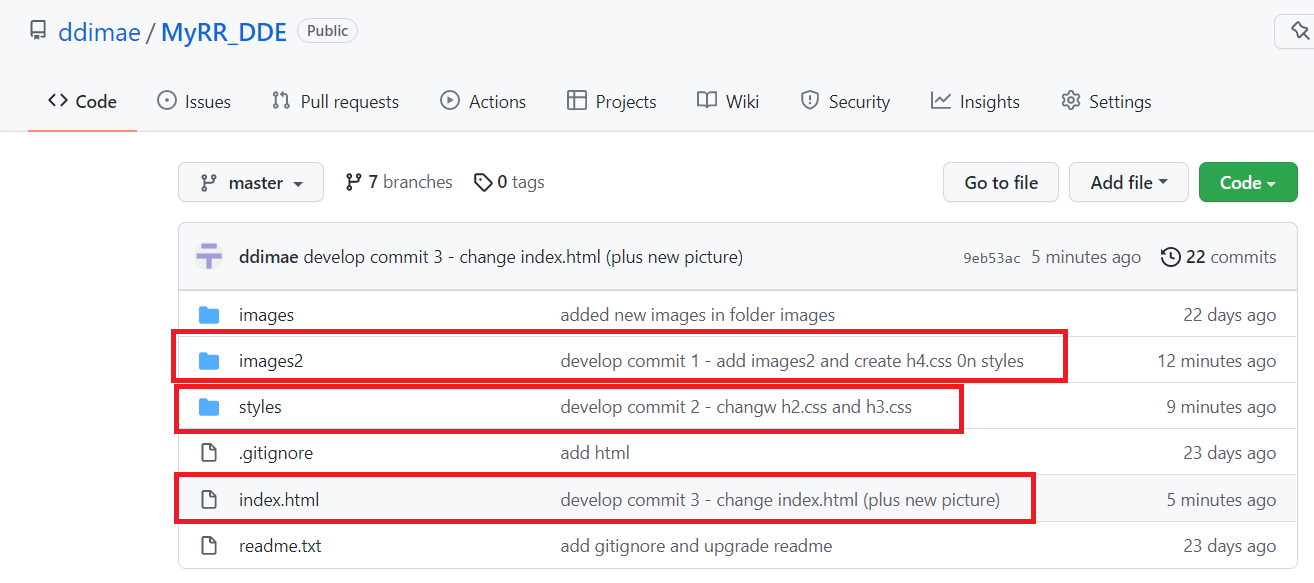
1. After complete upgrade, merge *develop2* to *master.*



1. Push made changes from your local repository to remote repository on github.



1. Check appearance new version of project in remote repository.



**Task Git-7. Make changes in project and save it by used pull requests**

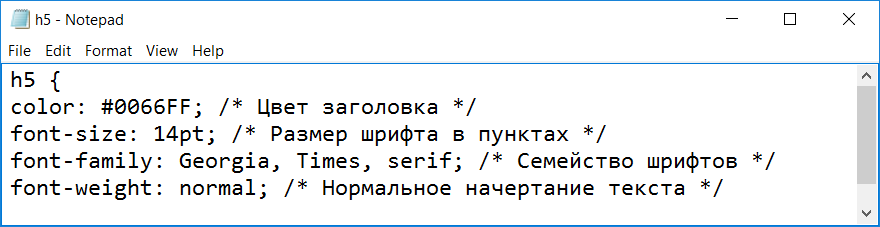
This task allow training skills in saving changes in remote repository after consent of other developers.

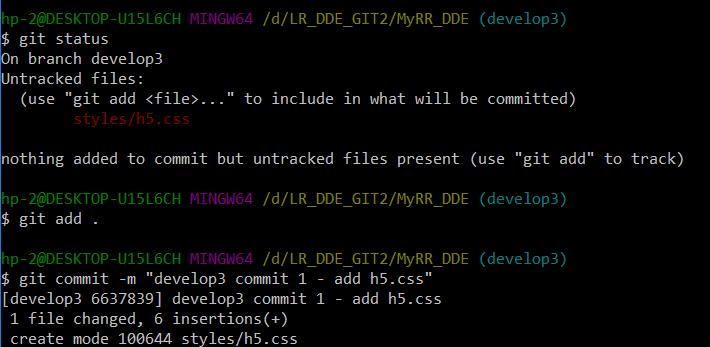
In executing this task you would continue working with clone of the project from remote repository (it placed at folder LR\_<FIO>\_GIT2/MyRR\_<FIO>). If you want you can to clone project to other folder.

1. Use **git Bash** for create branch *develop3* for implementation a new feature. Checkout on it.
2. Make 2-3 commits during development new feature. After finish your work you NOT MERGE or NOT REBASE project into master and go to next step.

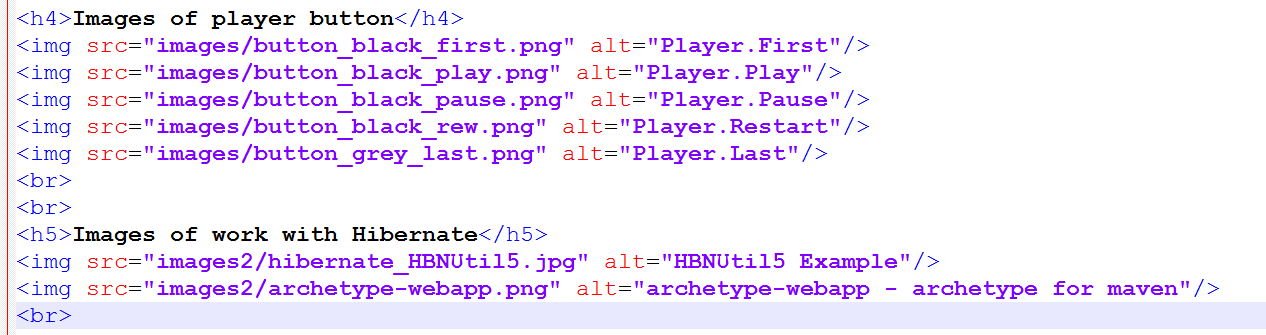
I done:

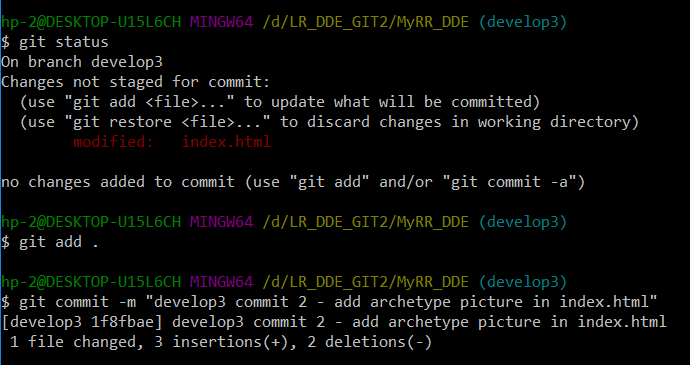
* Create h5.css (commit 1);



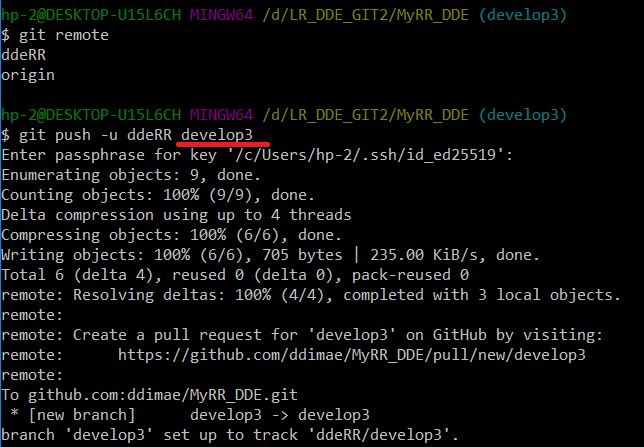


* Change index.html (commit 2).

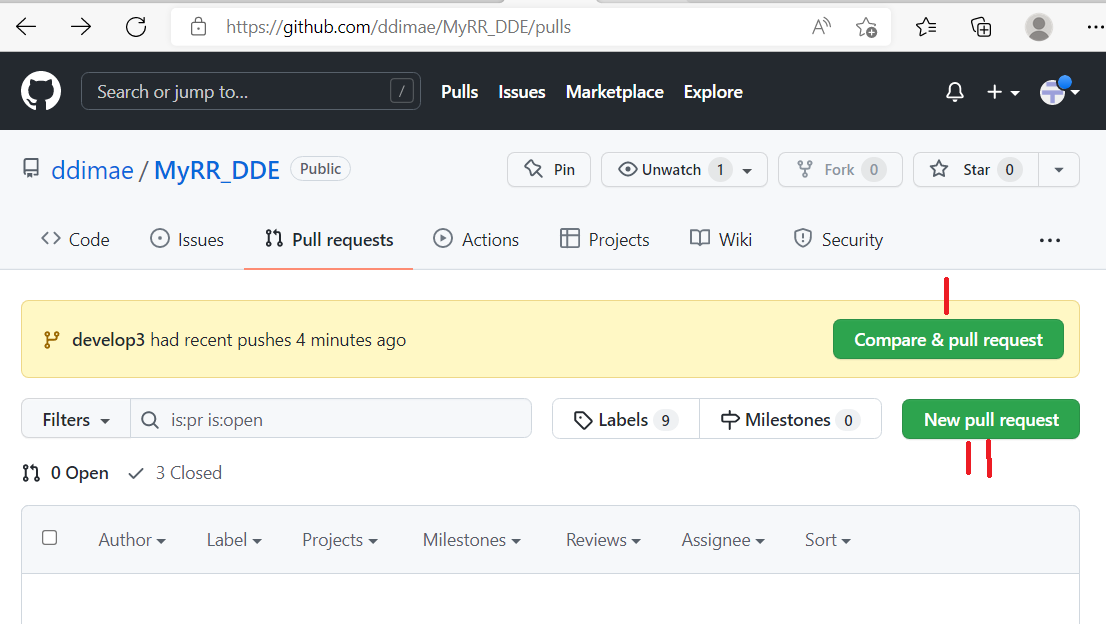




1. Push made changes made on branch *develop3* from your local repository to remote repository on github. In difference with same step in Task Git-6 you will set branch *develop3* as branch to push.

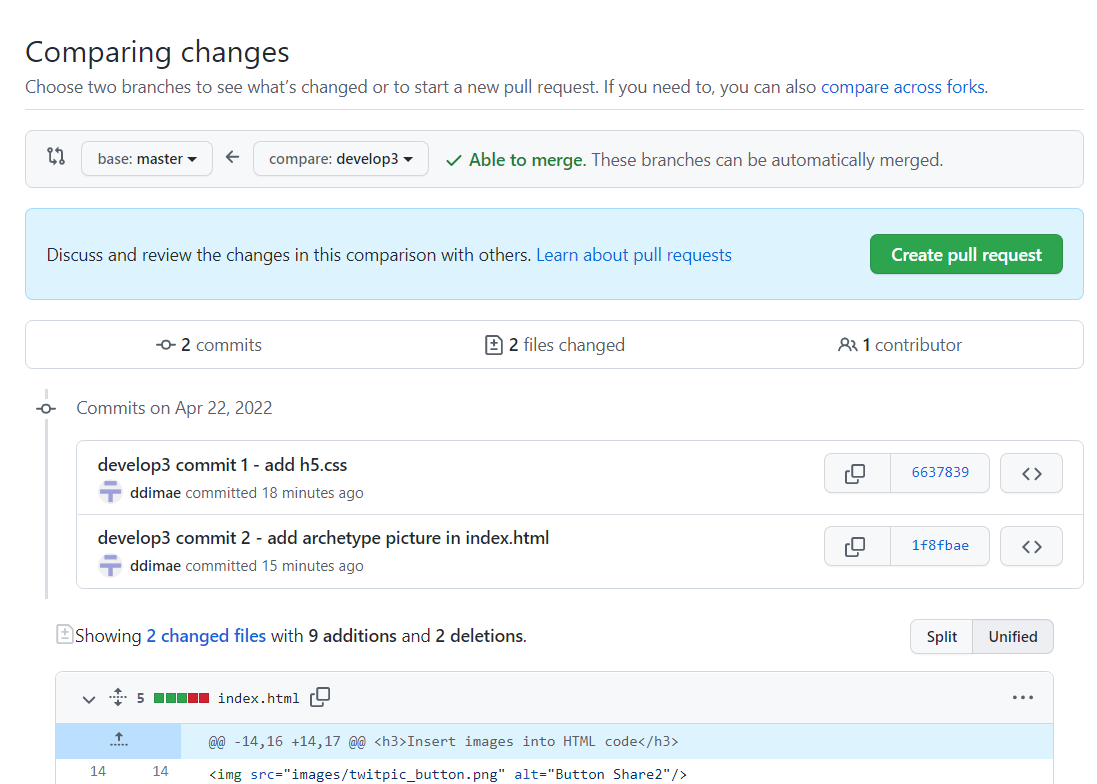


1. Open your remote repository on your **github** account. On bookmark «Pull request» create pull request for merge *develop3* to *master*. Make sure what you try to download correct version of new feature implementation. Send message about this.

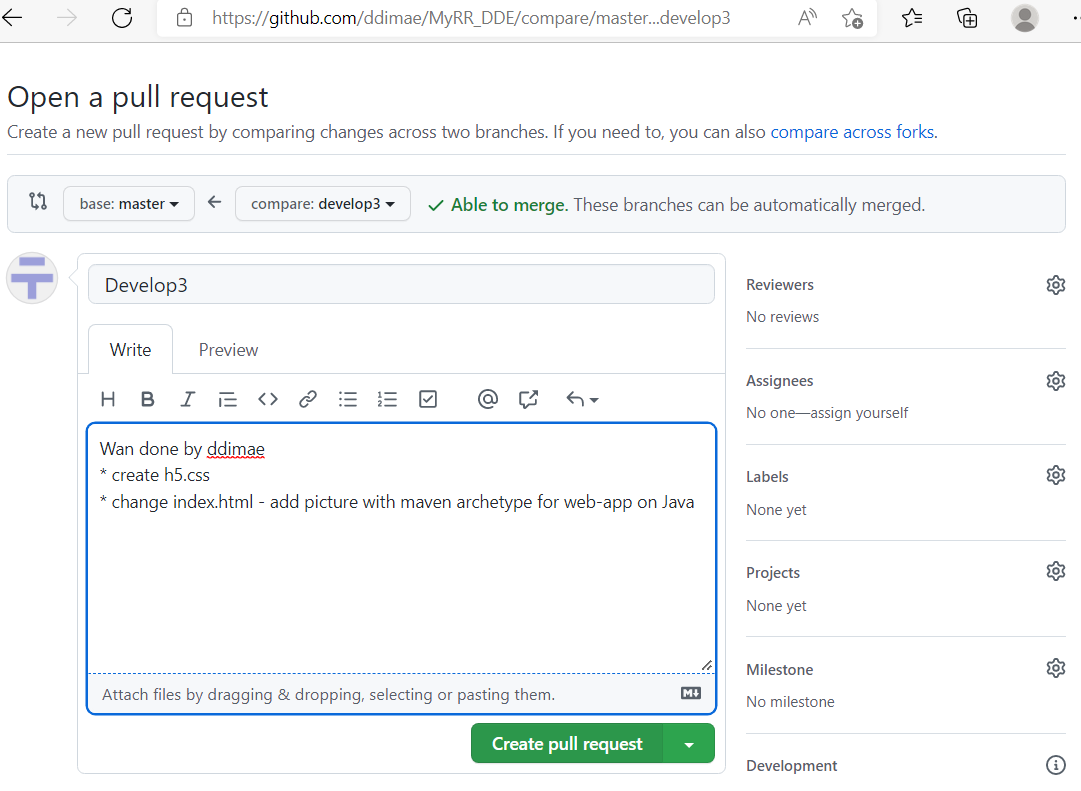


You have to way for create new pull request. I – more easy. II – you at first must define branches to merge. Show how to do by way II.

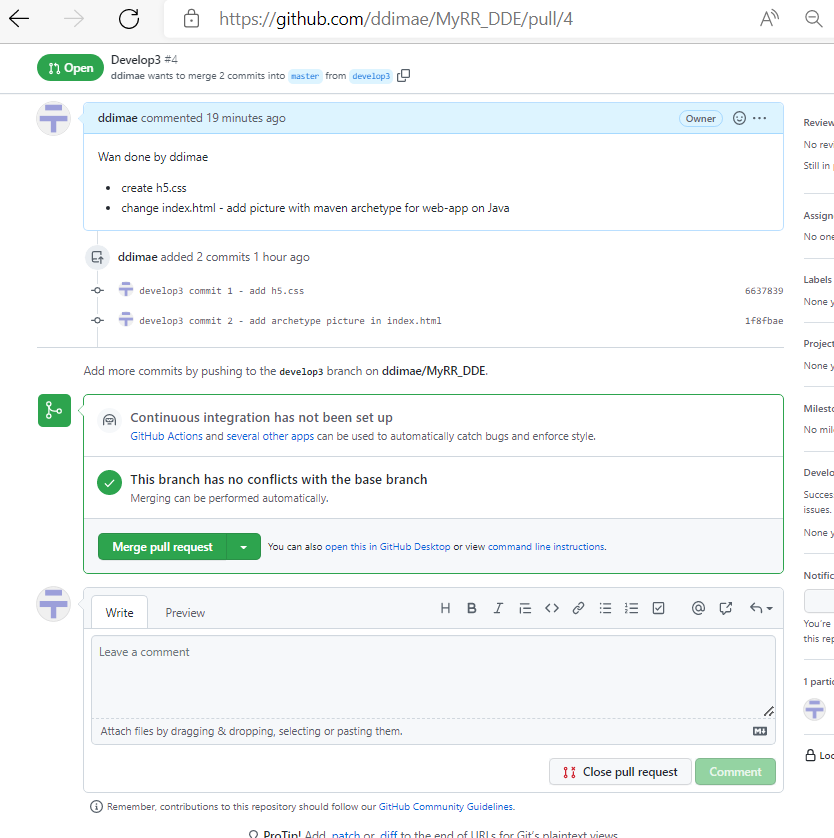
Press New pull request. In window what appear select branches.



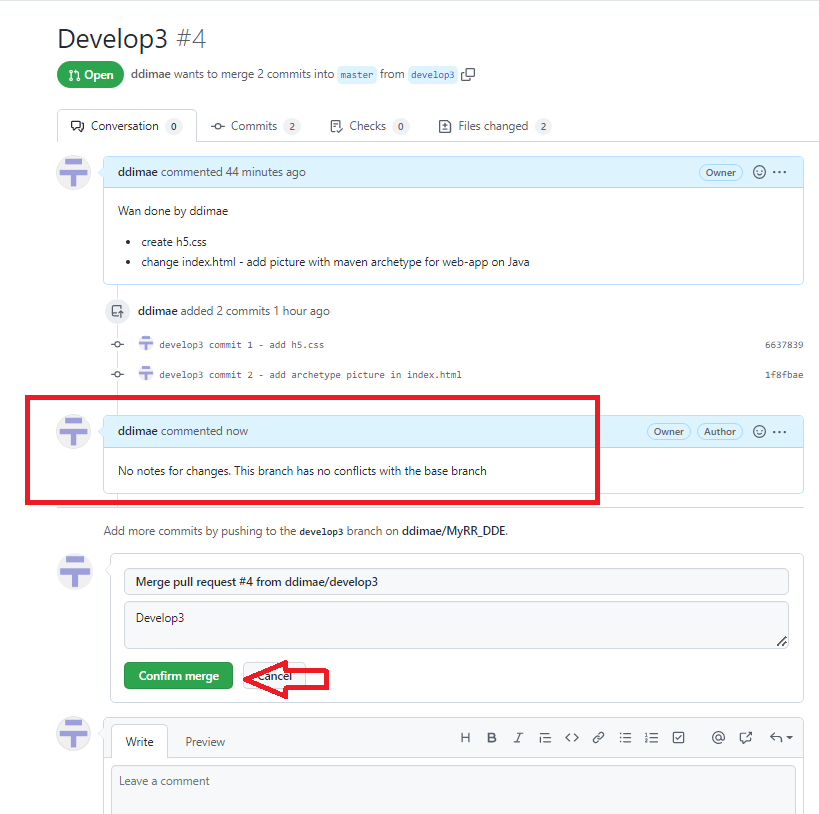
After press “Create pull request”. Next, you must set name for pull request and describe it. In addition, you can check changes in commit to be sure what you propose correct version of your code what implement new feature.



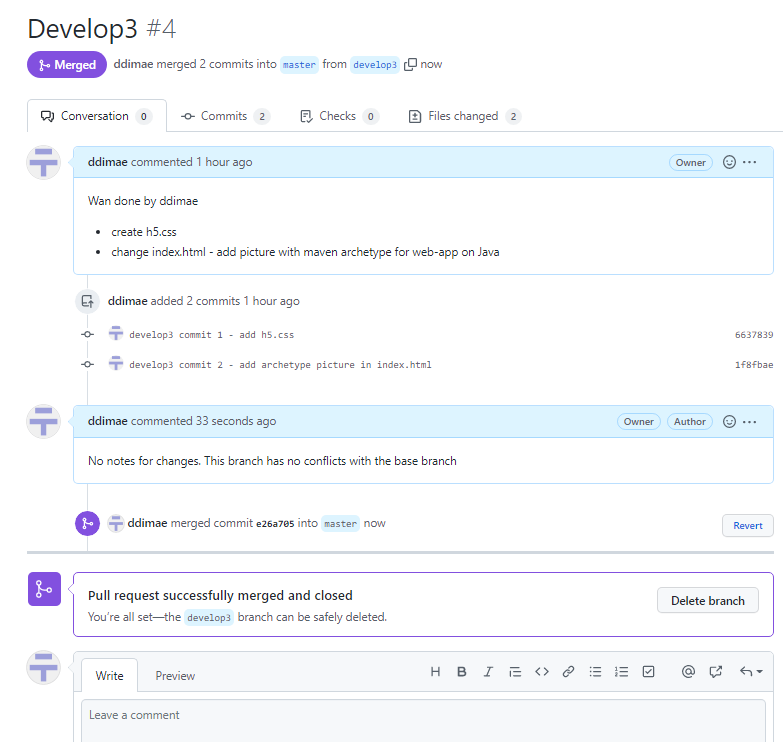
After press “Create pull request”, you get next screen.

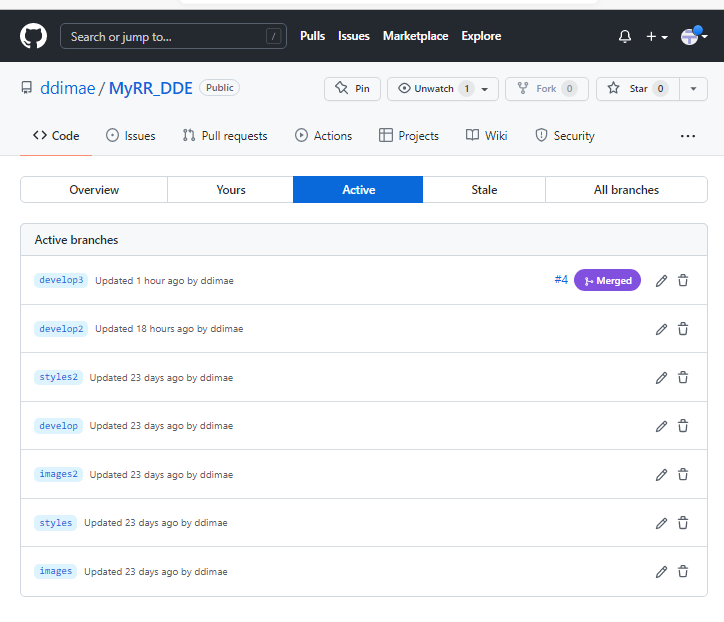


1. Push new branches into repository used option “Create merge commit”, which involves what all commit in branch develop3 will to be add to master via a merge commit.

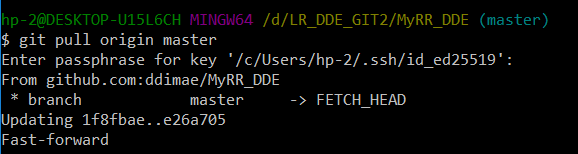


1. Check result of merging in list of branches.



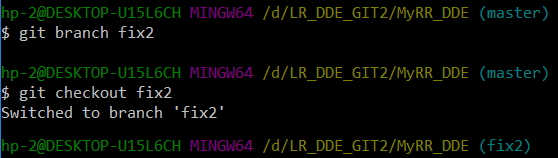


1. Return to **git Bash.** Checkout to *master* (!!!). Prepare for next upgrading of your project.



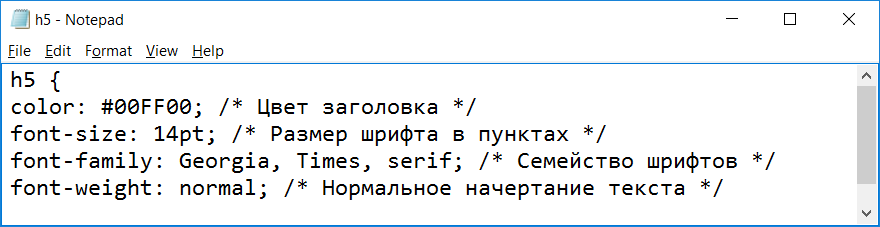
1. Create new branch to develop one more feature or fix some problem. Name for braches – choice of the student.

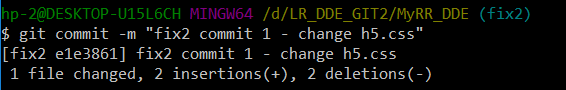
I created branch *fix2*.



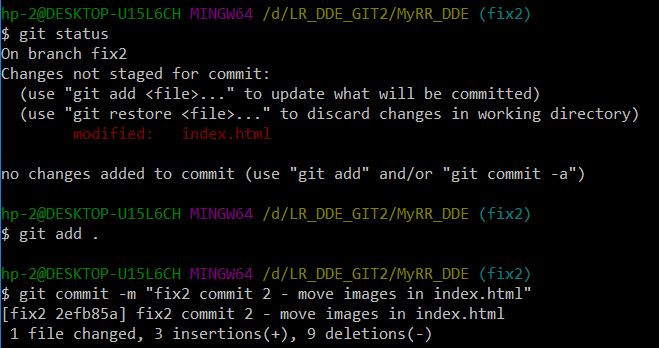
Make 2-3 commits on this branch.

Commit “fix2 commit 1”

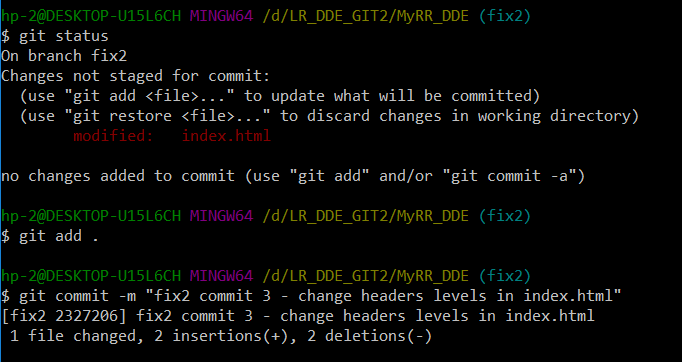




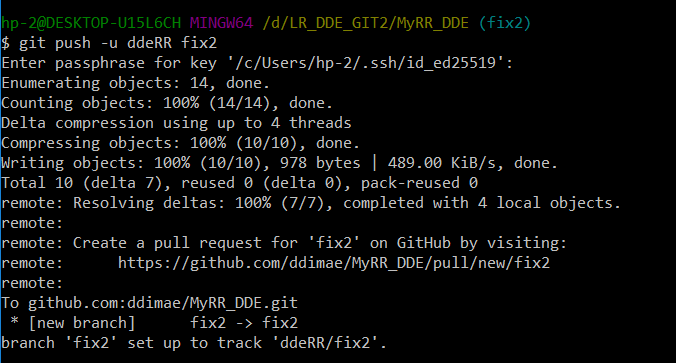
Commit “fix2 commit 2”



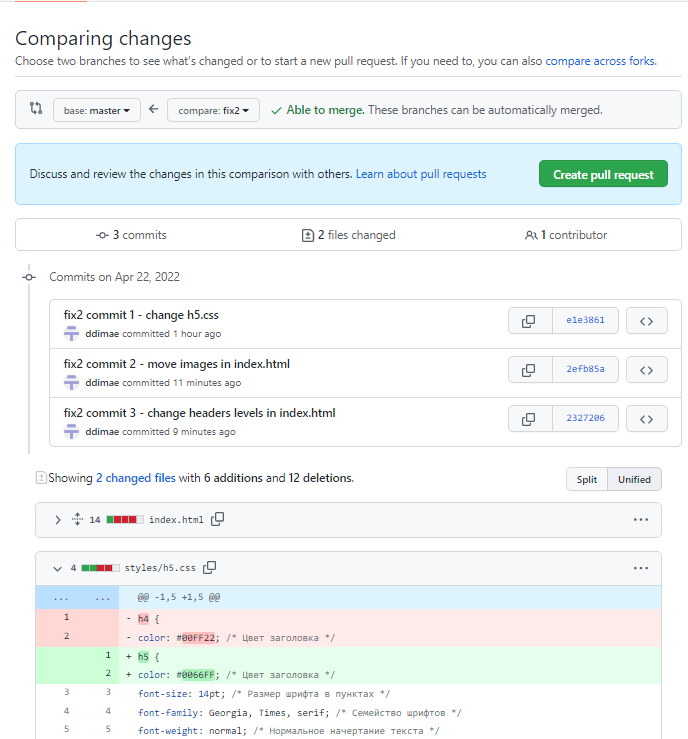
Commit “fix2 commit 3”



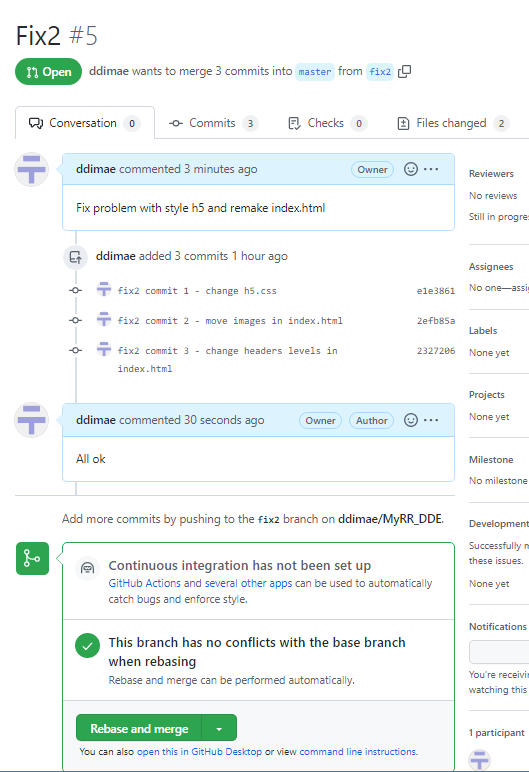
Push branch with changes in remote repository also without merging with *master*.



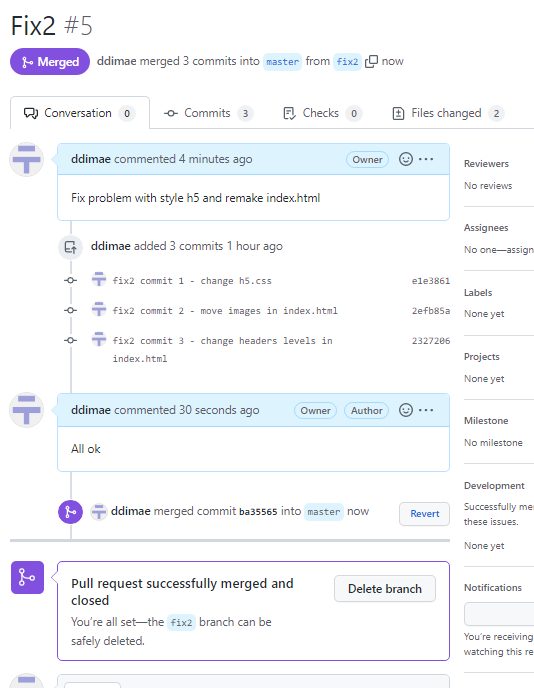
1. Open your remote repository on your **github** account. Create pull request for integrating new code.

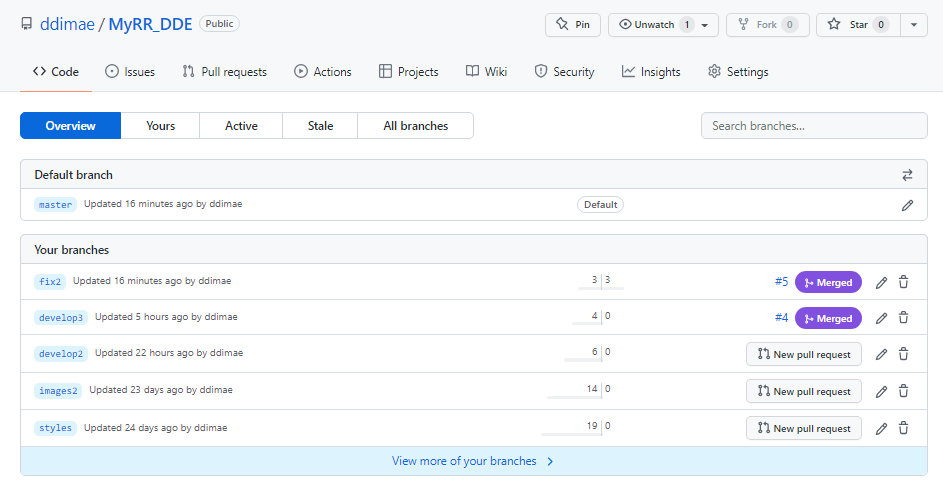


Push new branches into repository used option “*Rebase and merge*”, which involves what all commit in branch will be rebased and added to *master*.

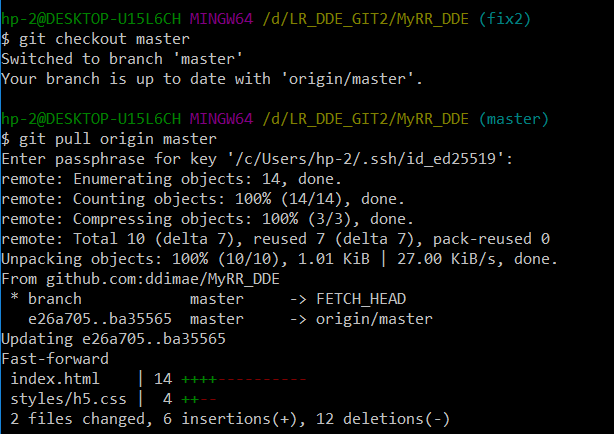


1. Check result of merging in list of branches.

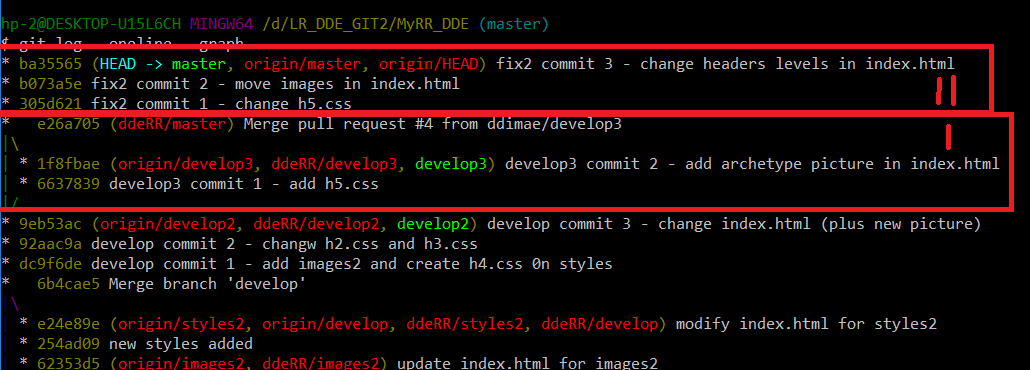




1. Return to **git Bash.** Checkout to *master* and get changes from remote repository.



1. Show commits history in tree view. Research how to github made merge your branches.



I – it is commits have been merged; II - it is commits have been rebased. Both braches have been commit by use github.

This was presented typical actions must be train in LAB#3