<epam> Excellence in Software Engineering

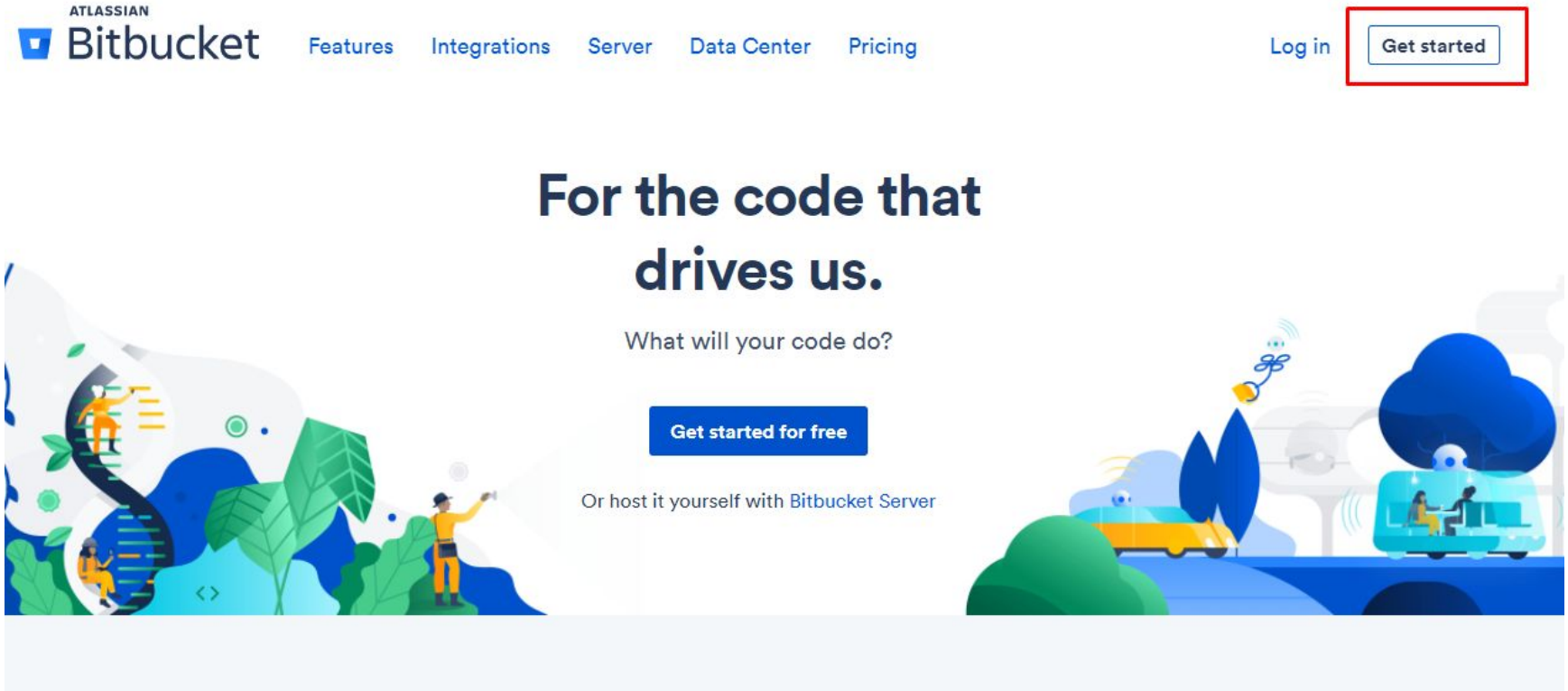
Bitbucket repository

EPAM Java Training

Mogilev 2017

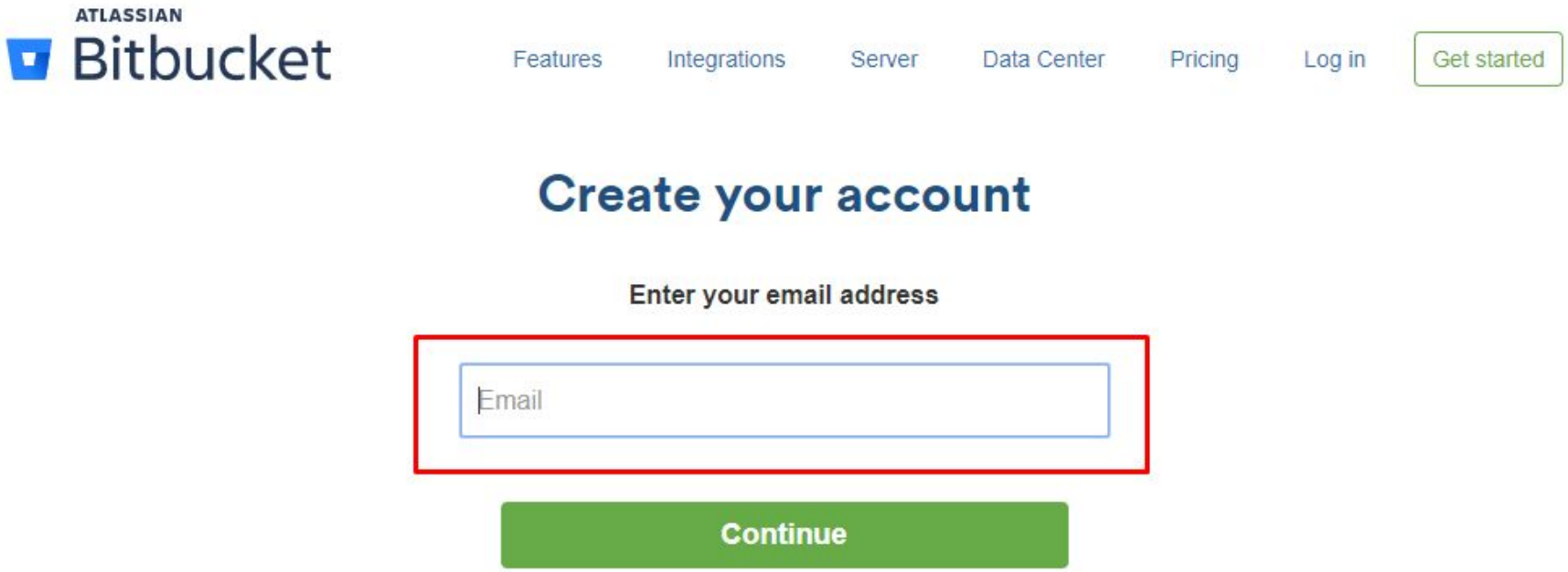
1. Create new account

- Go to https://bitbucket.org/ and click “Get started”



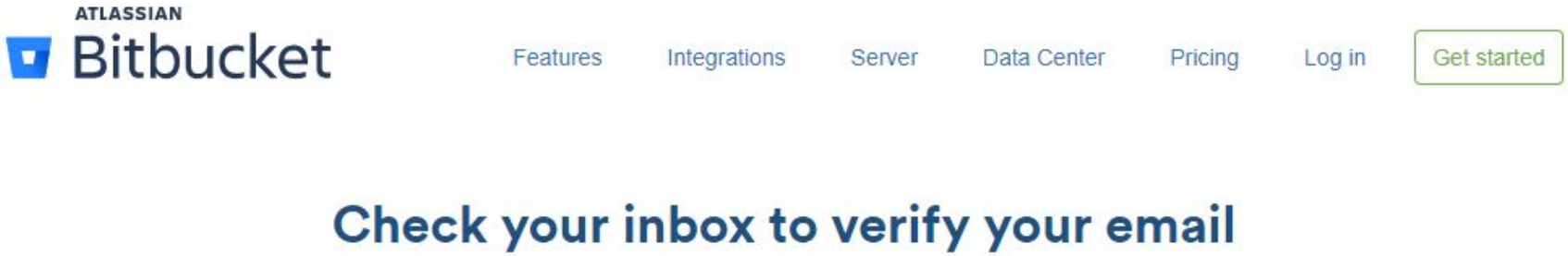
1. Create new account

- Enter your email address



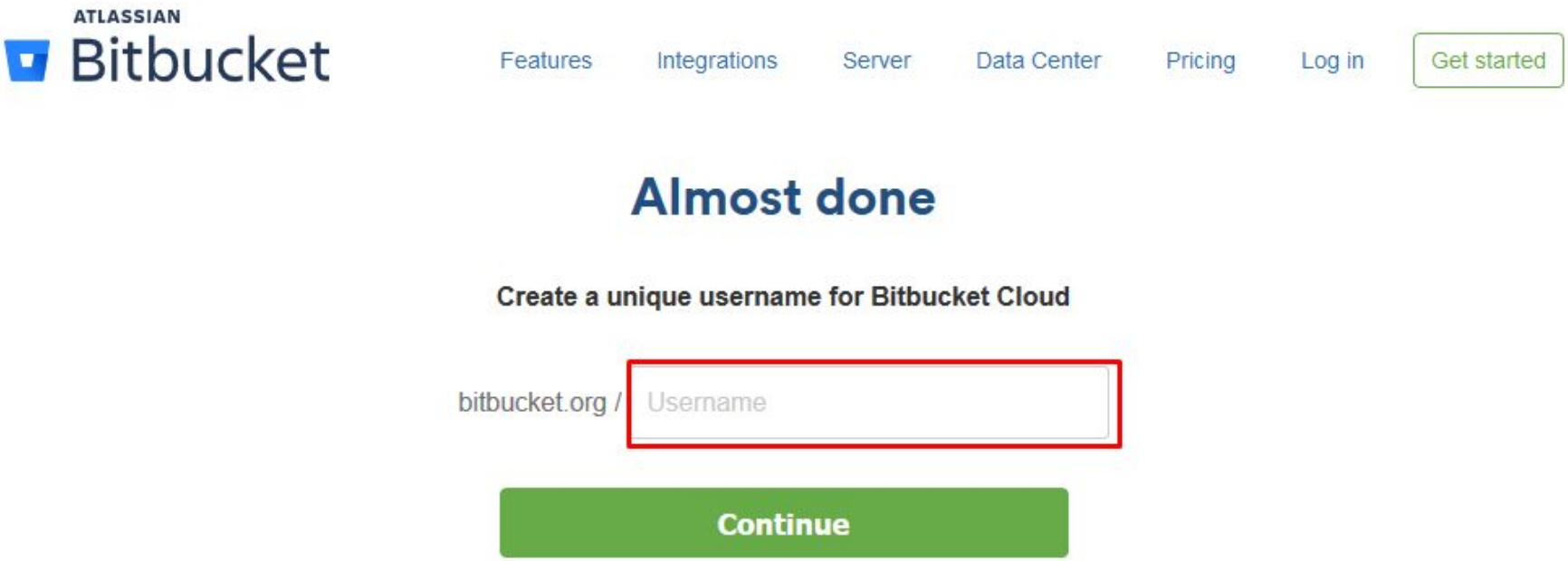
1. Create new account

- Complete your account details: Full Name and Password.
- Check “I’m not robot” and click “Continue”



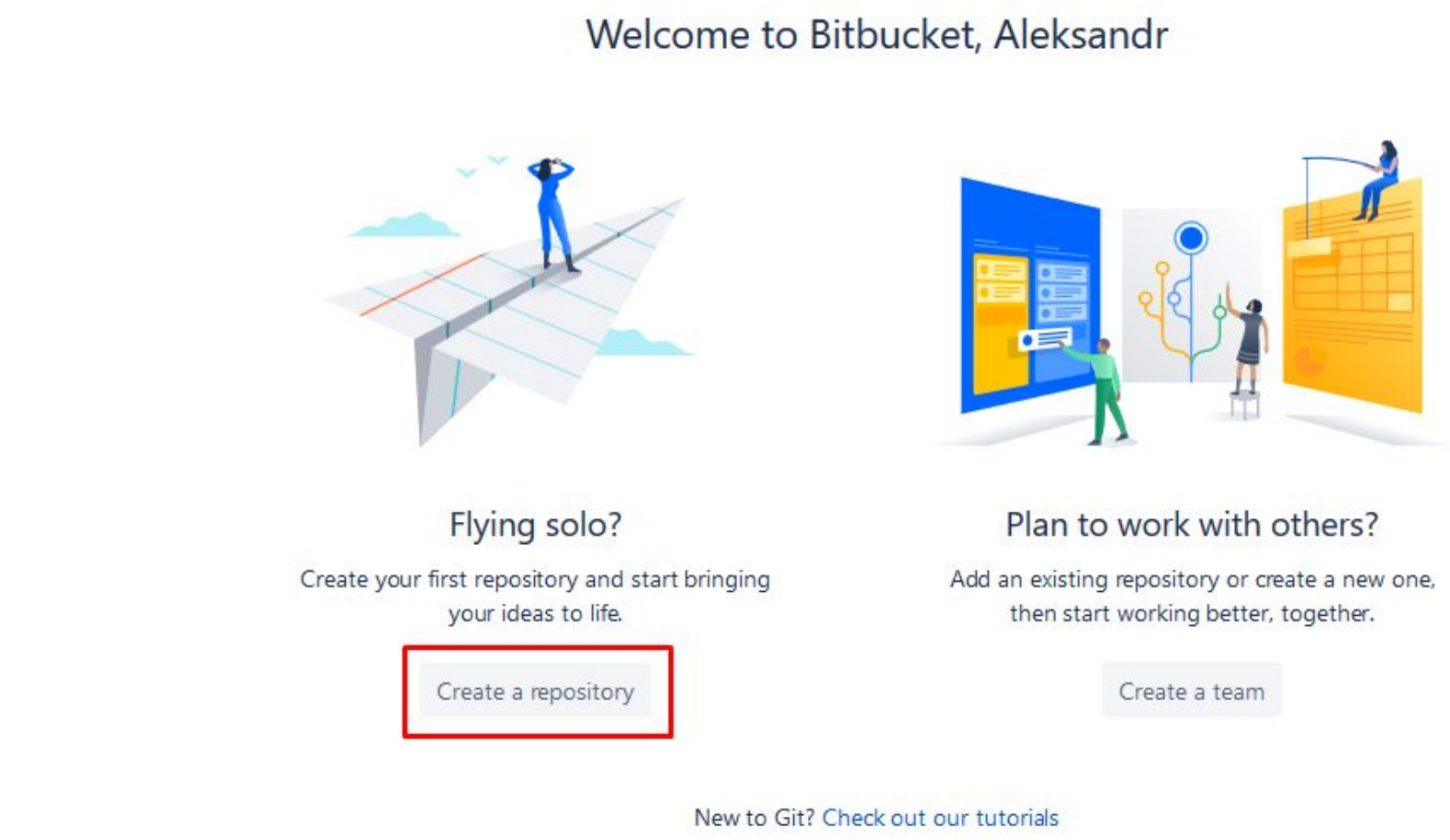
1. Create new account

- Create unique username for Bitbucket Cloud



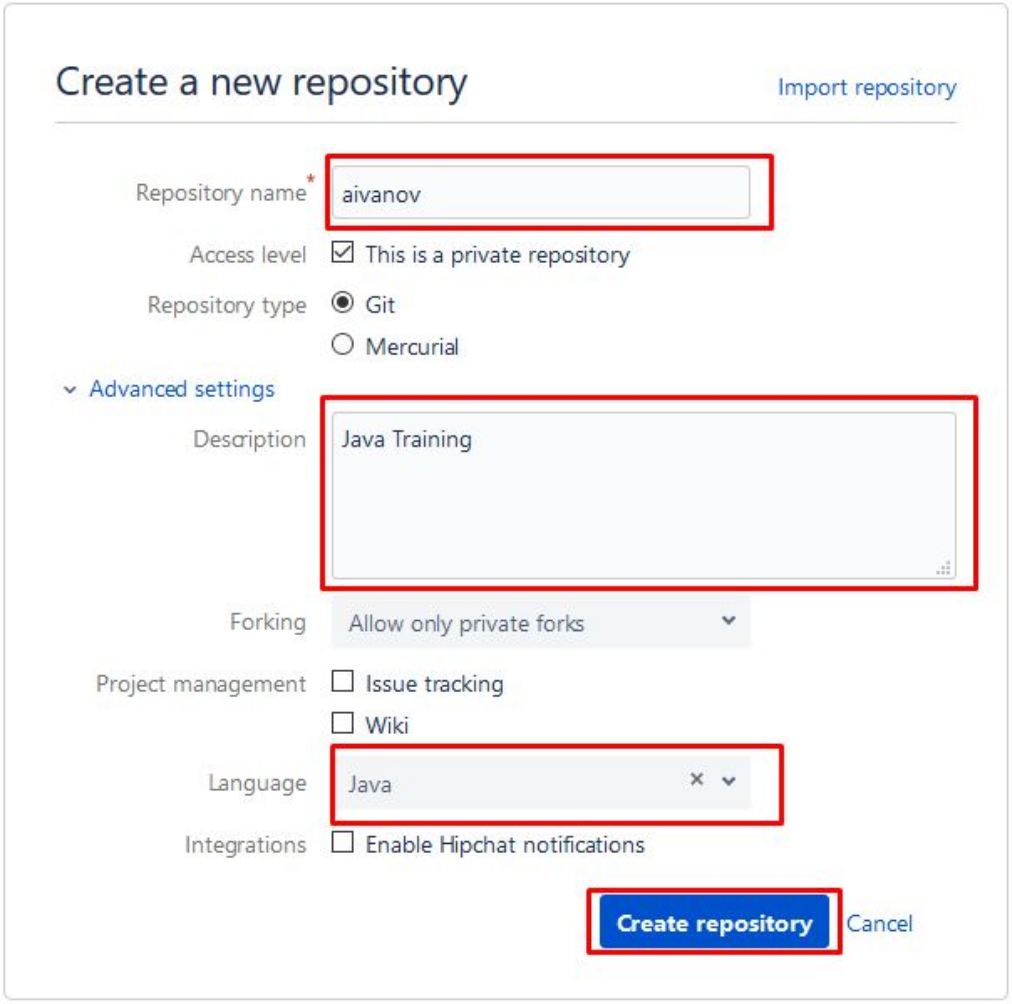
2. Create new repository

- Create a repository



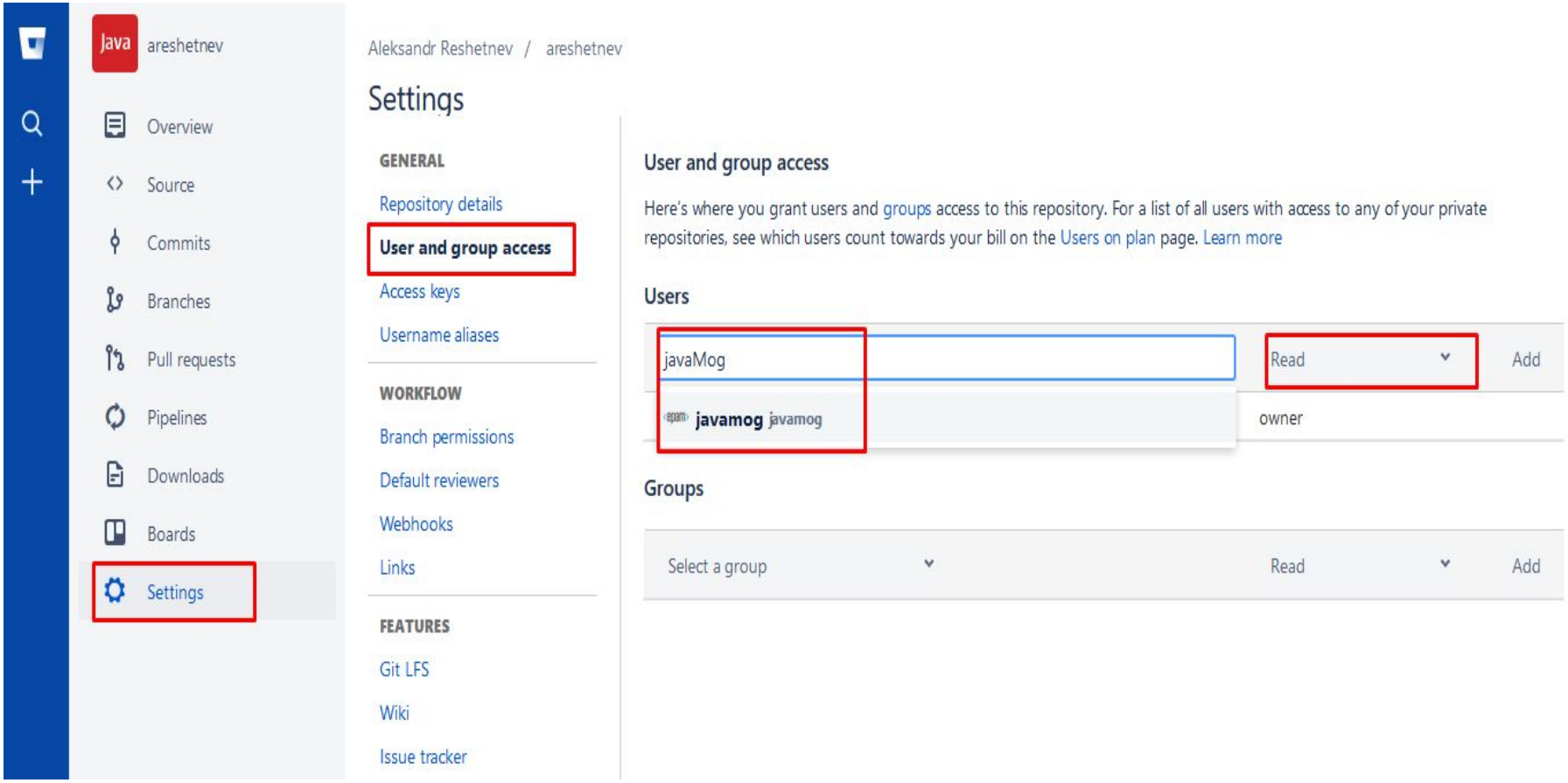
2. Create new repository

- Repository name. Use your First Name and Surname. Like “aivanov”



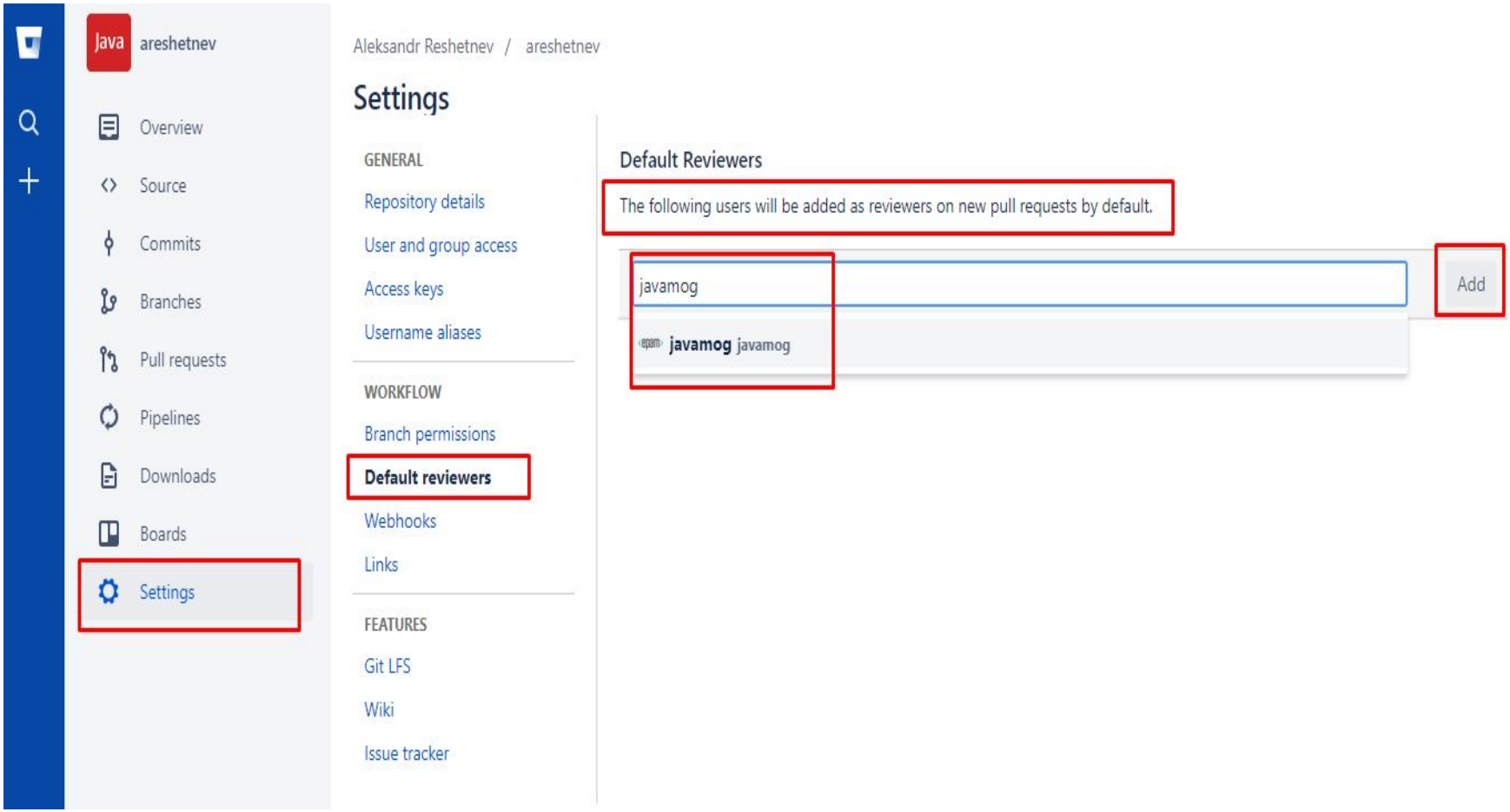
2. Create new repository

- Bitbucket repository successfully was created.



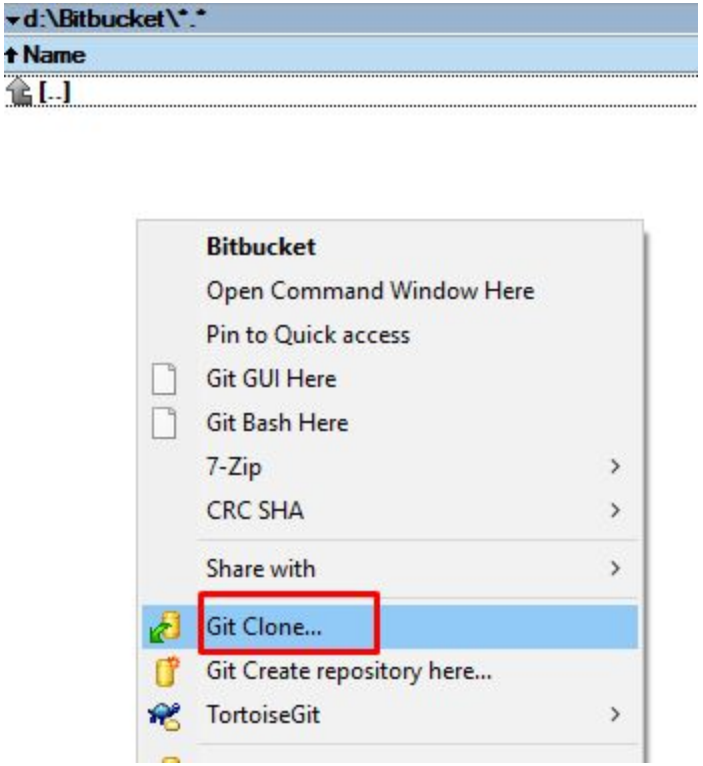
2. Create new repository

- Add default reviewers for new pull requests.



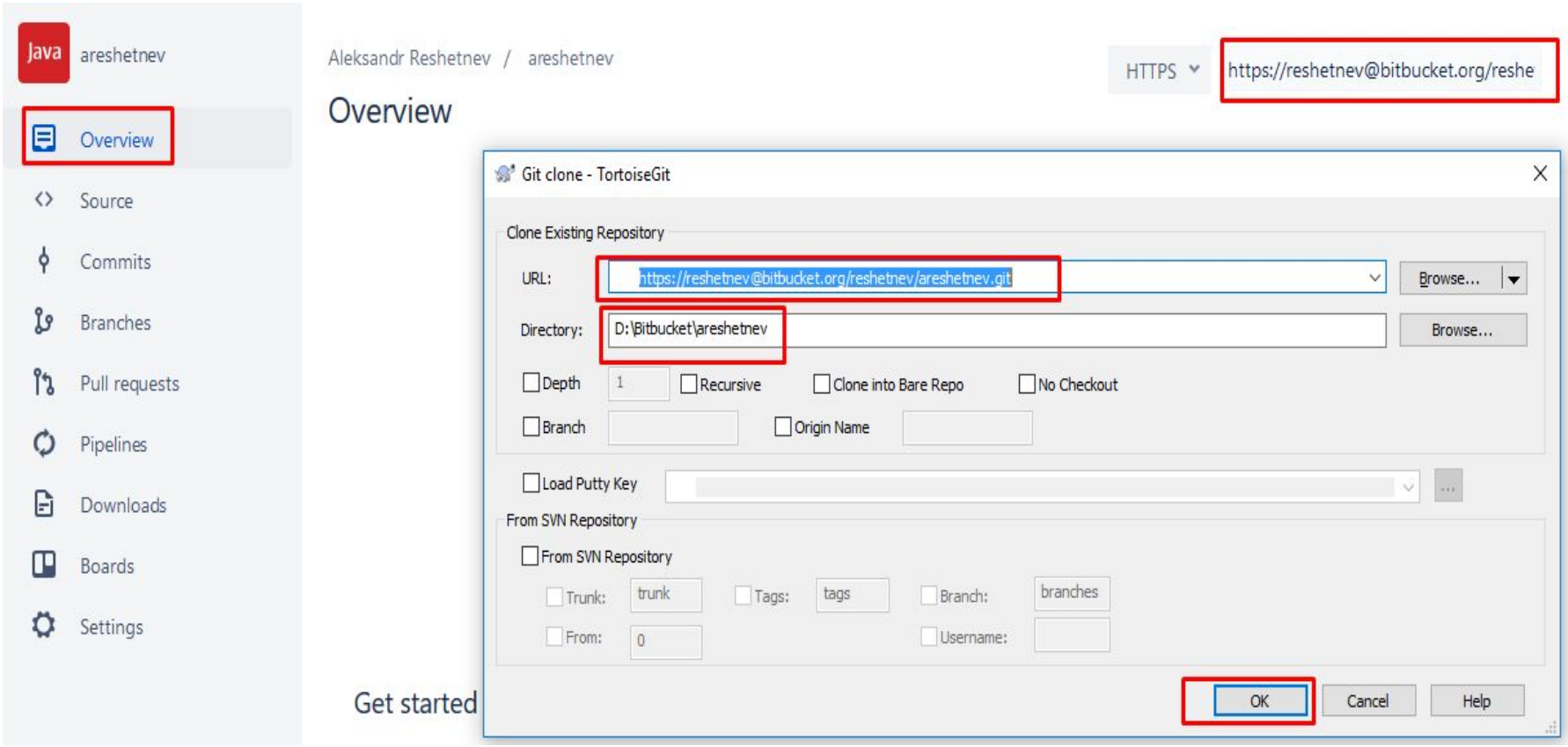
3. Checkout repository

- Install TortoiseGit
- Right click and choose “Git Clone…”



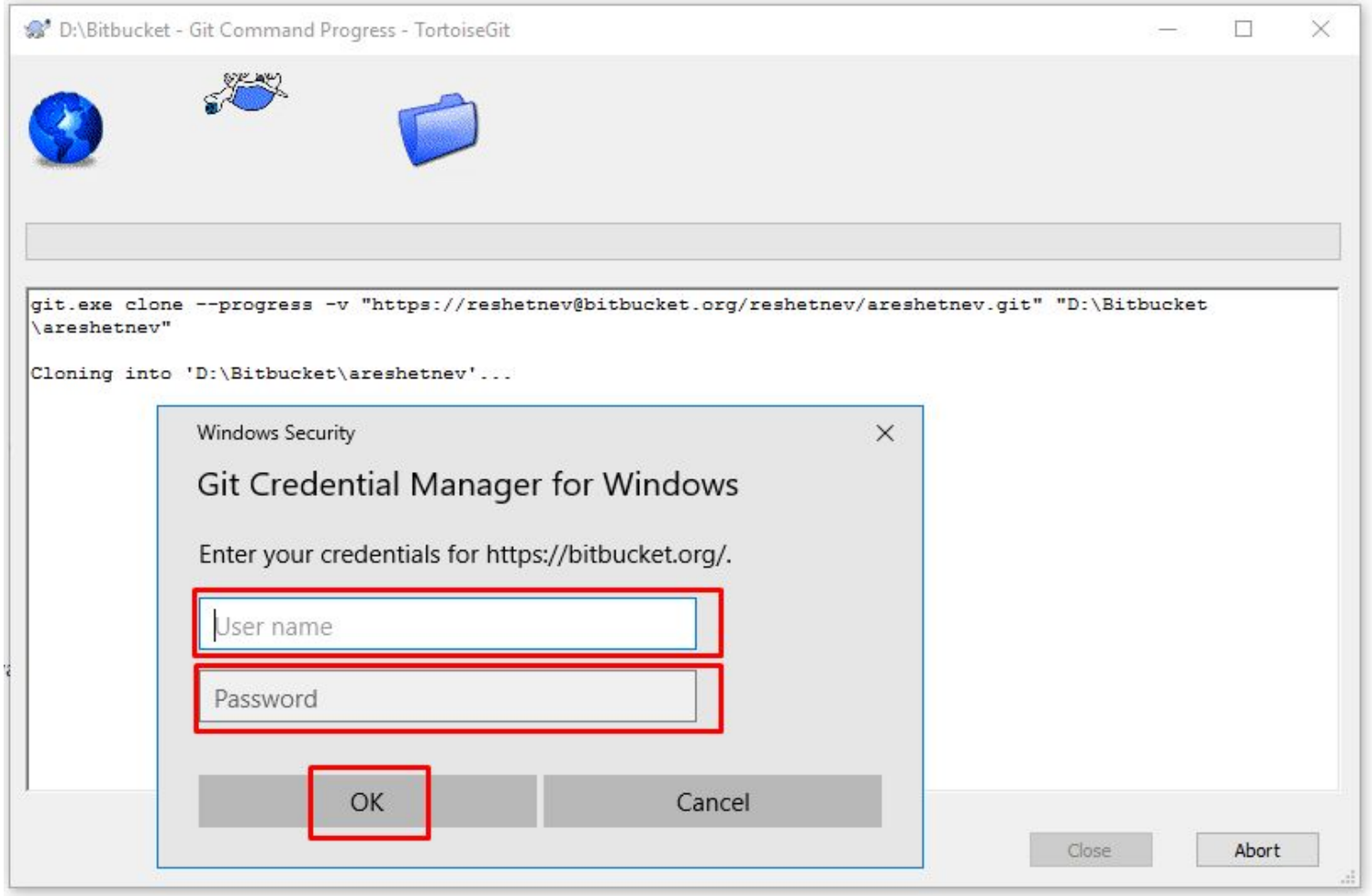
3. Checkout repository

- Choose correct checkout directory if needed



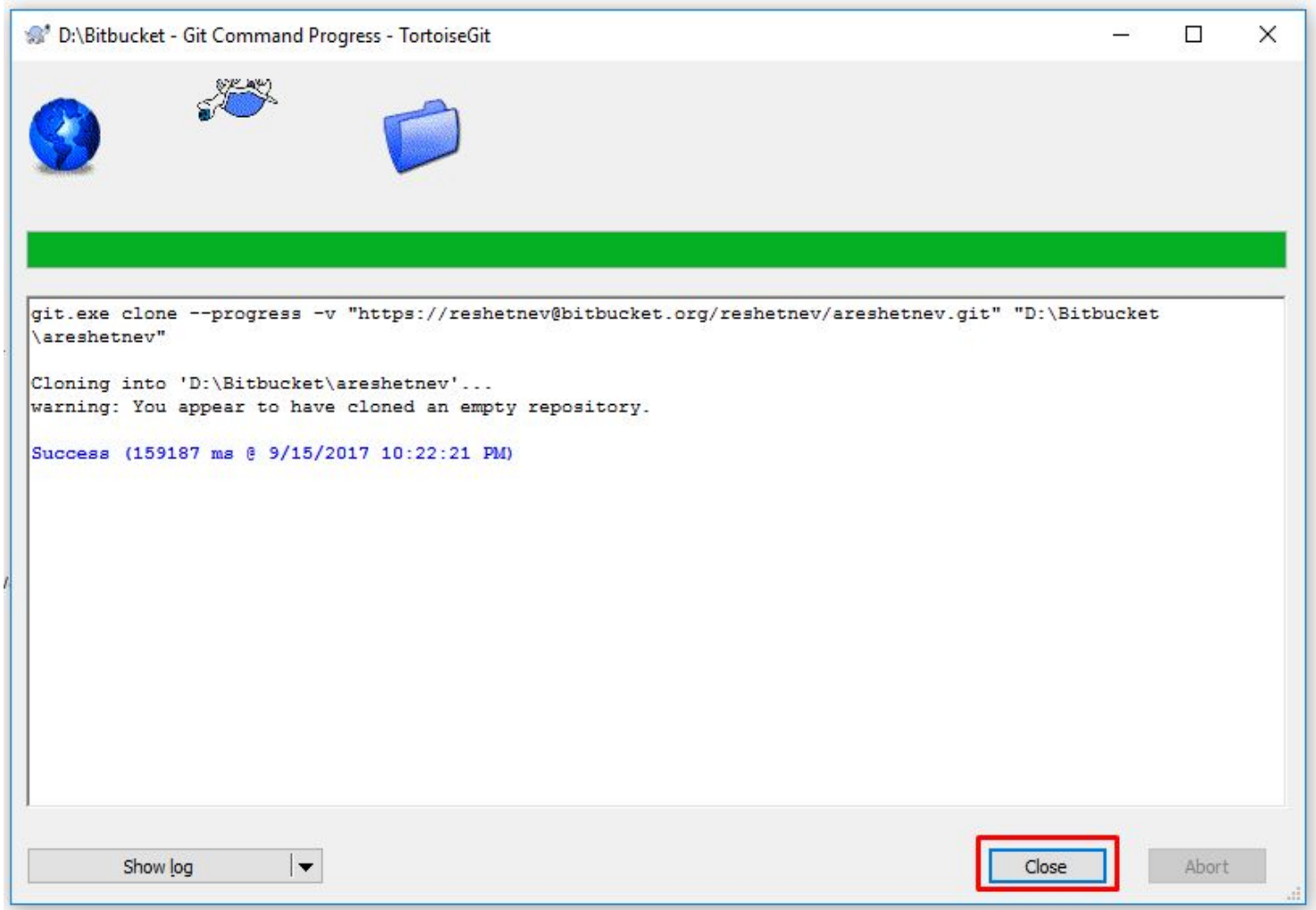
3. Checkout repository

- Enter credentials and click on save authentication checkbox



3. Checkout repository

- Checkout finished

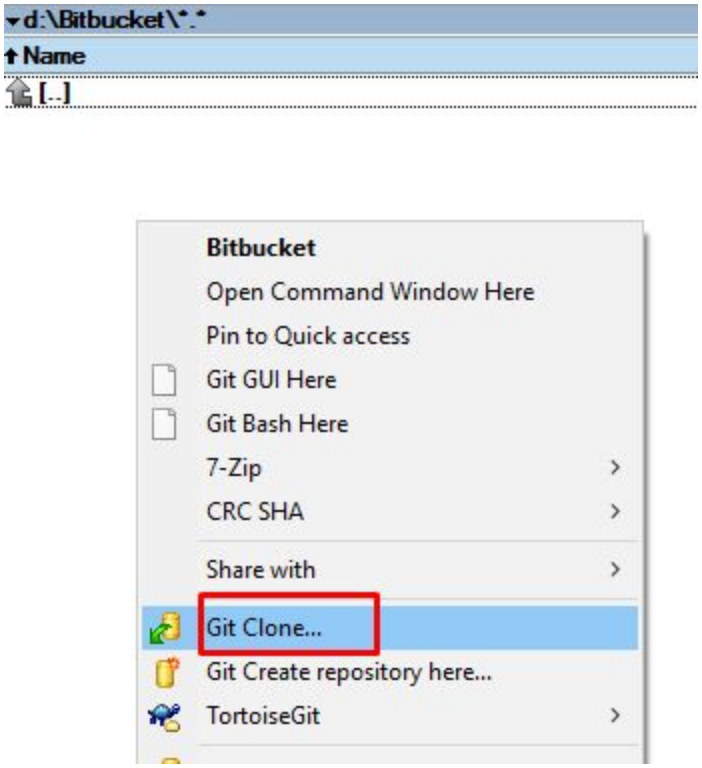


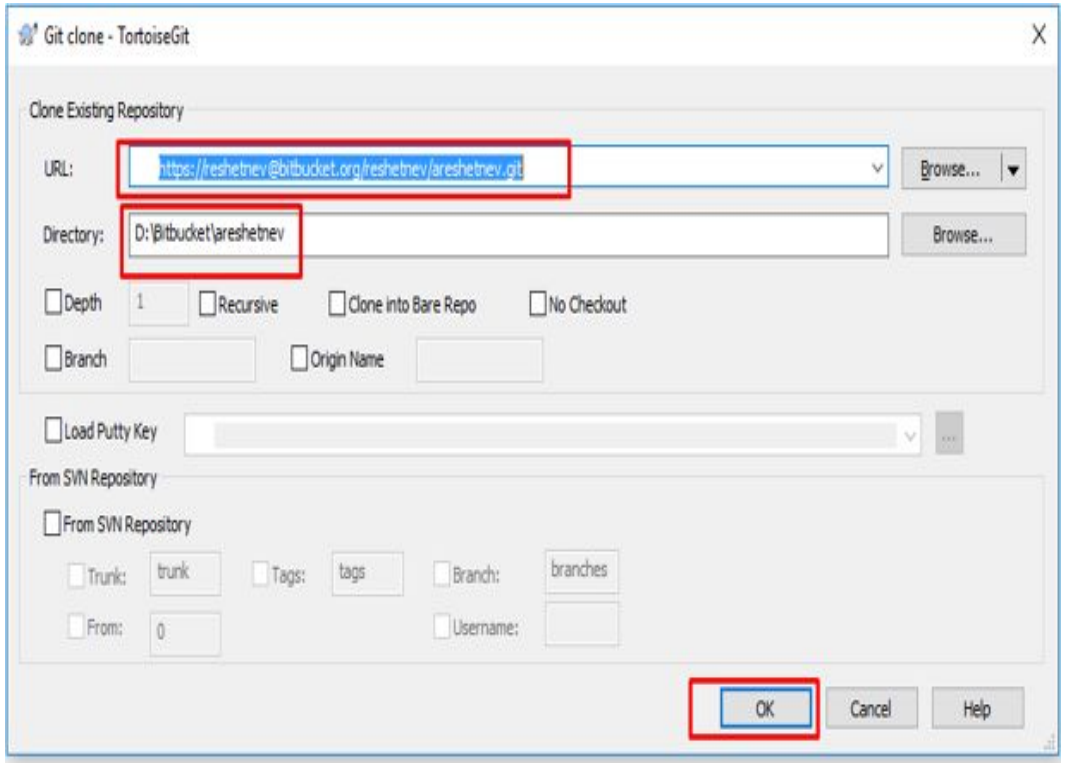
4. Pro Git

- Book “Pro Git”, Scott Chacon
- 1. https://git-scm.com/book/ru/v1
- 2. https://git-scm.com/book/en/v2
- 3. https://github.com/progit/progit2

4. Pro Git

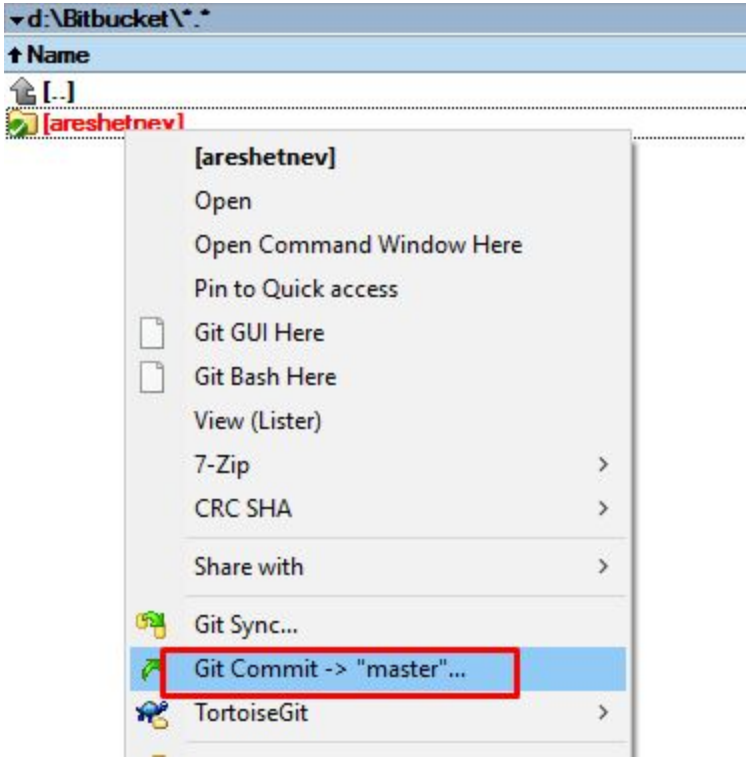
- Cloning an Existing Repository

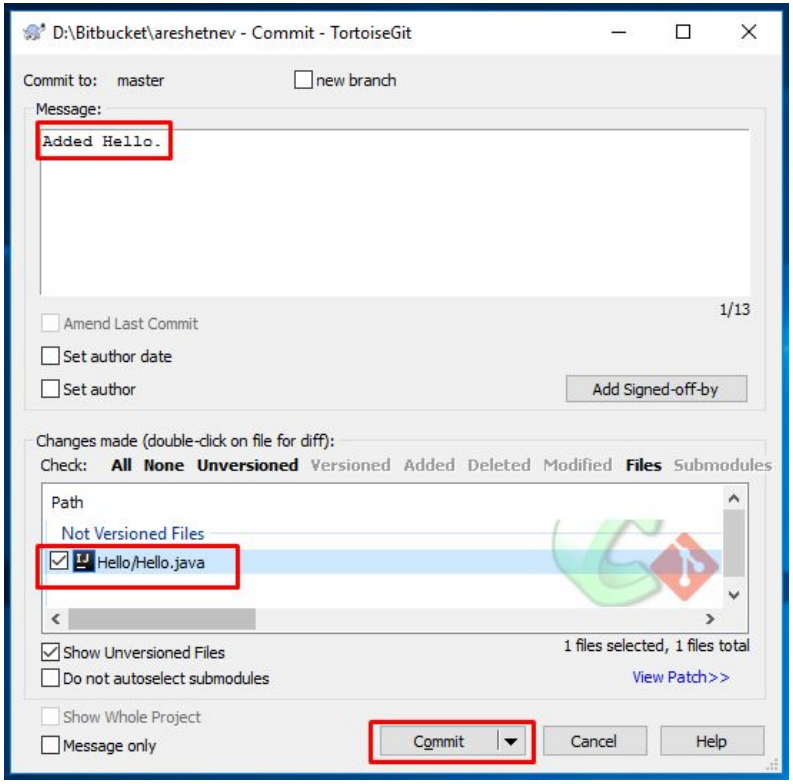




4. Pro Git

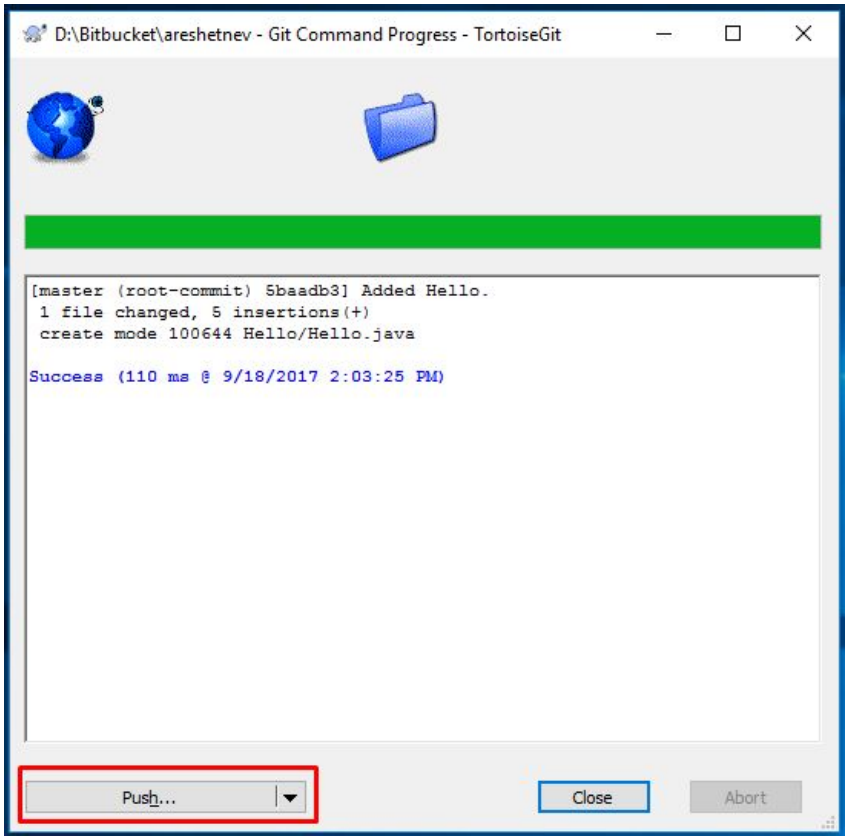
- Recording Changes to the ***local repository***

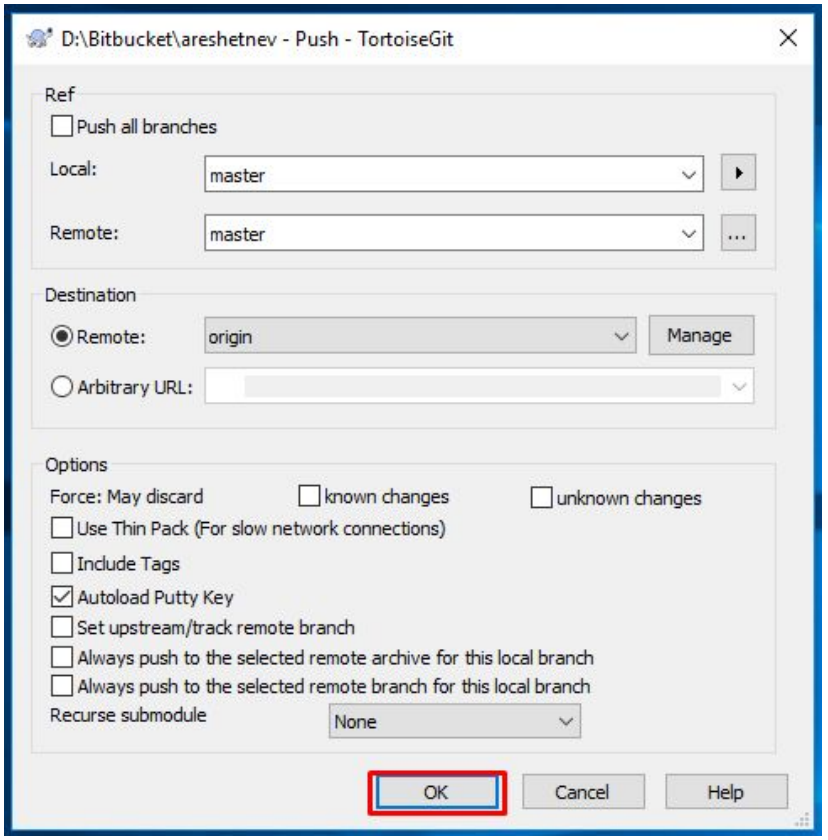




4. Pro Git

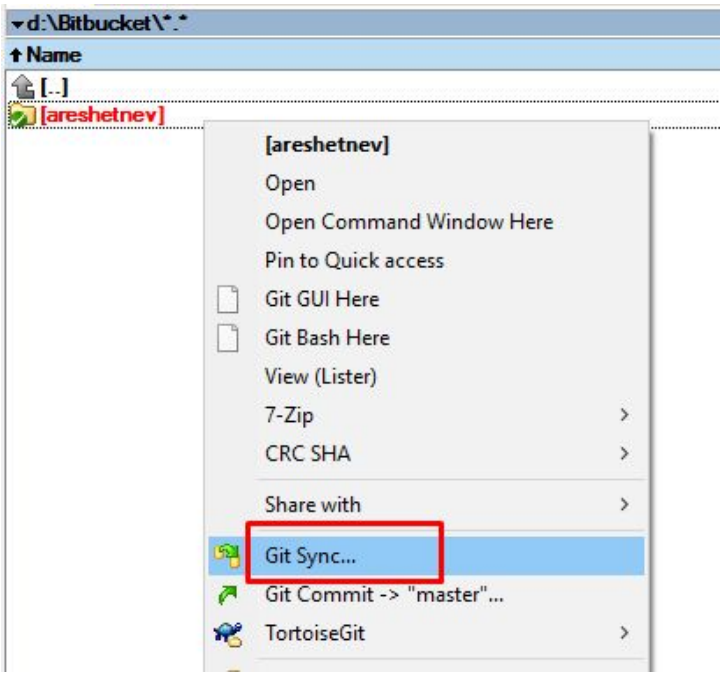
- Recording Changes to the ***remote repository***

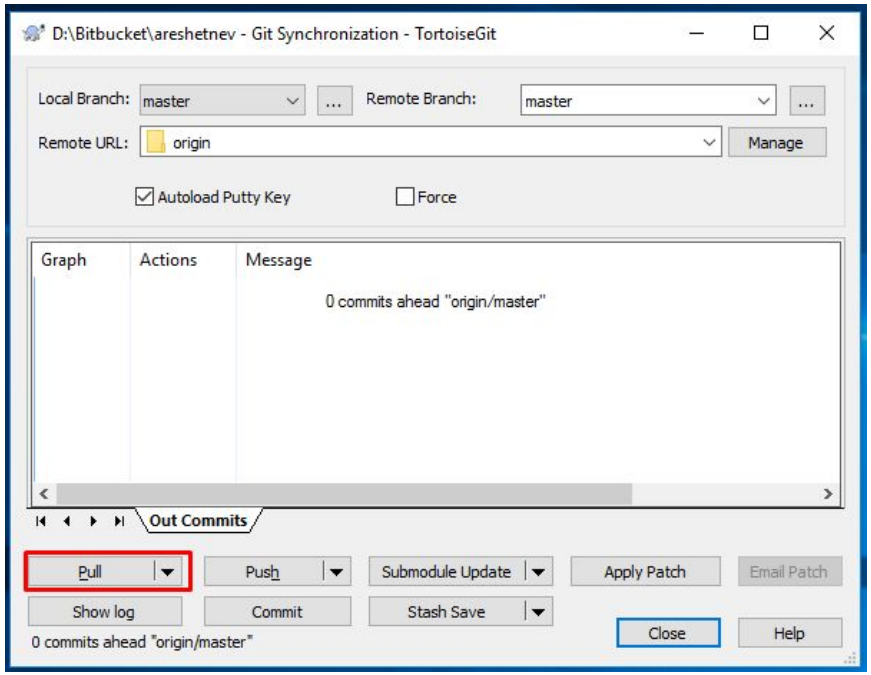




4. Pro Git

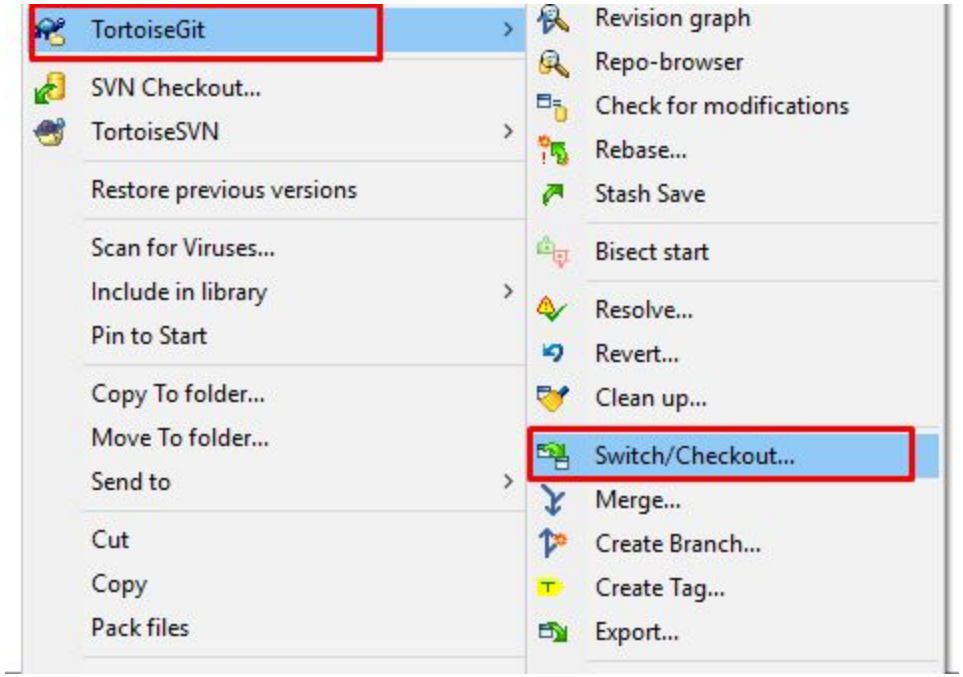
- Getting updates from remote repository

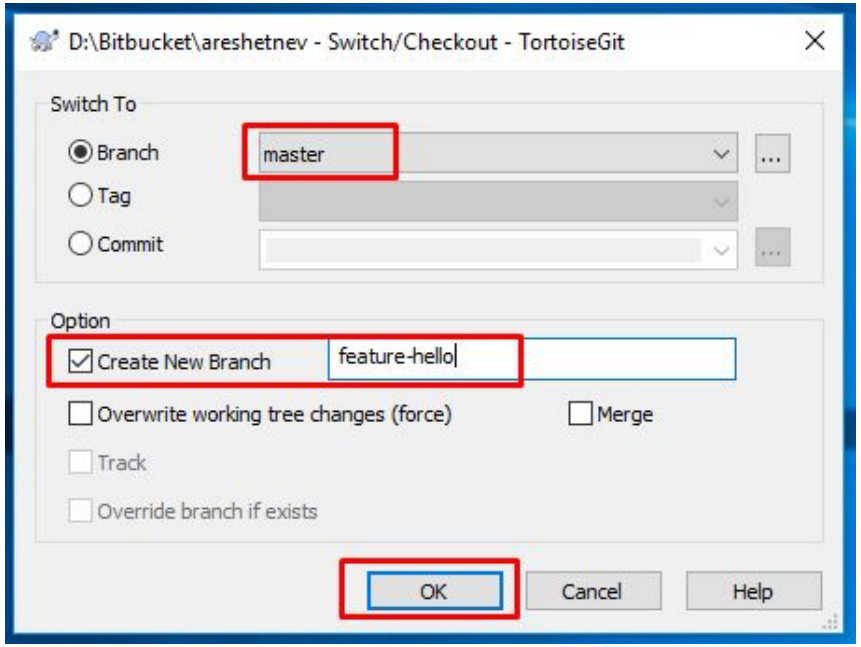




4. Pro Git

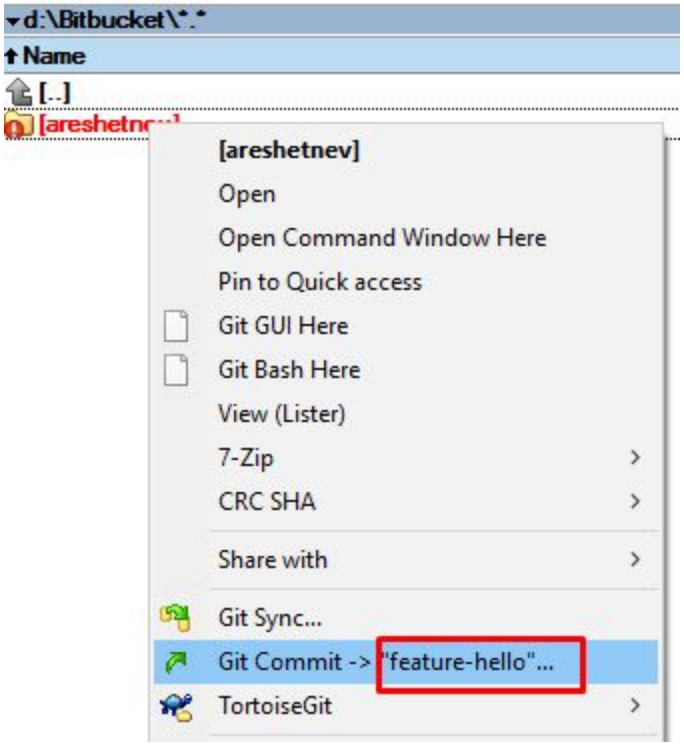
- Creating branch

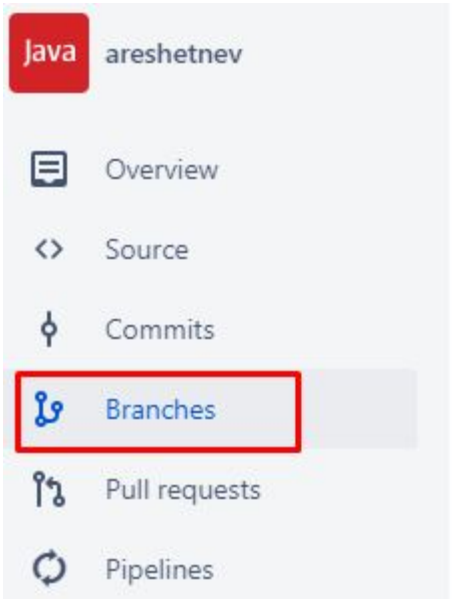


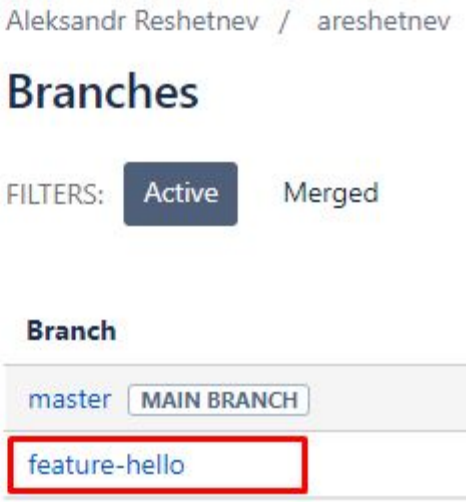


4. Pro Git

- Make sure that you commit in the created branch.
- After commit & push (check-in), Create pull request.

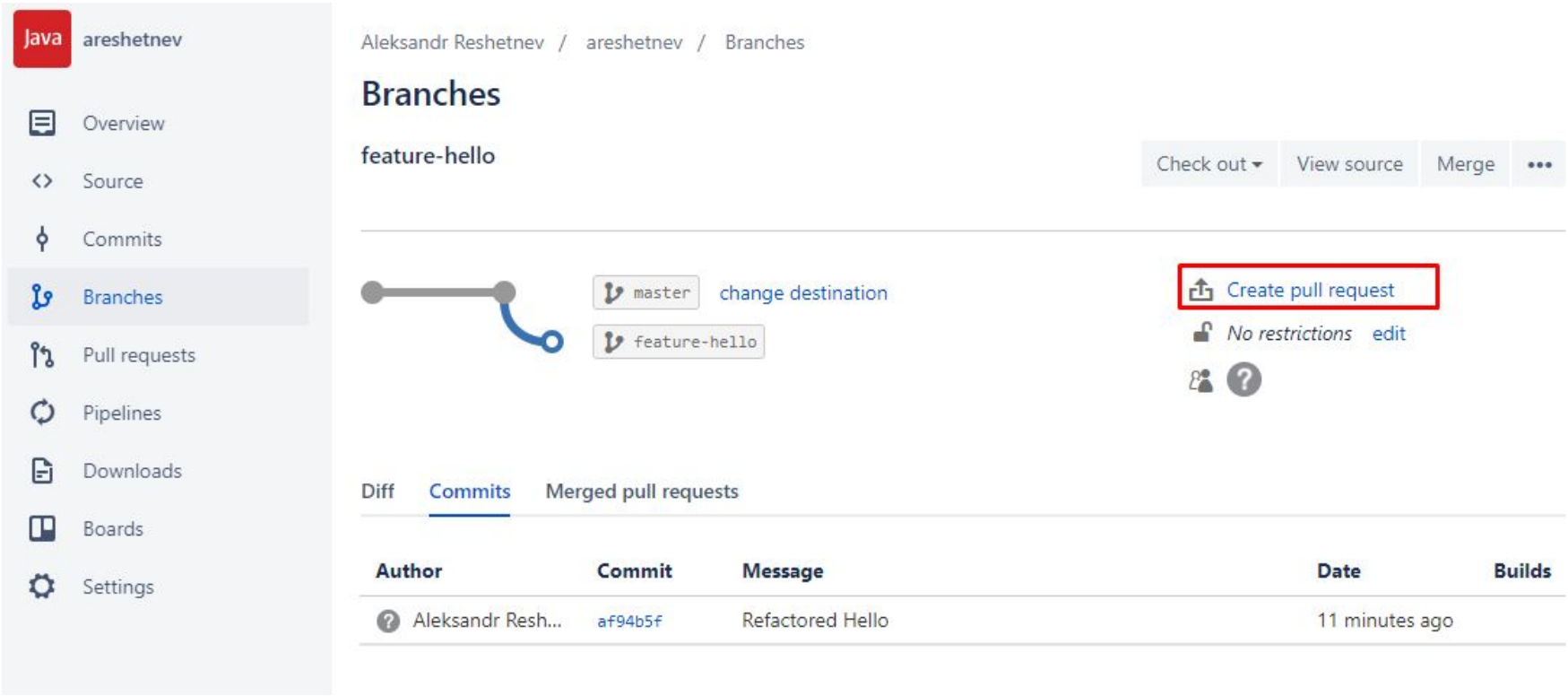




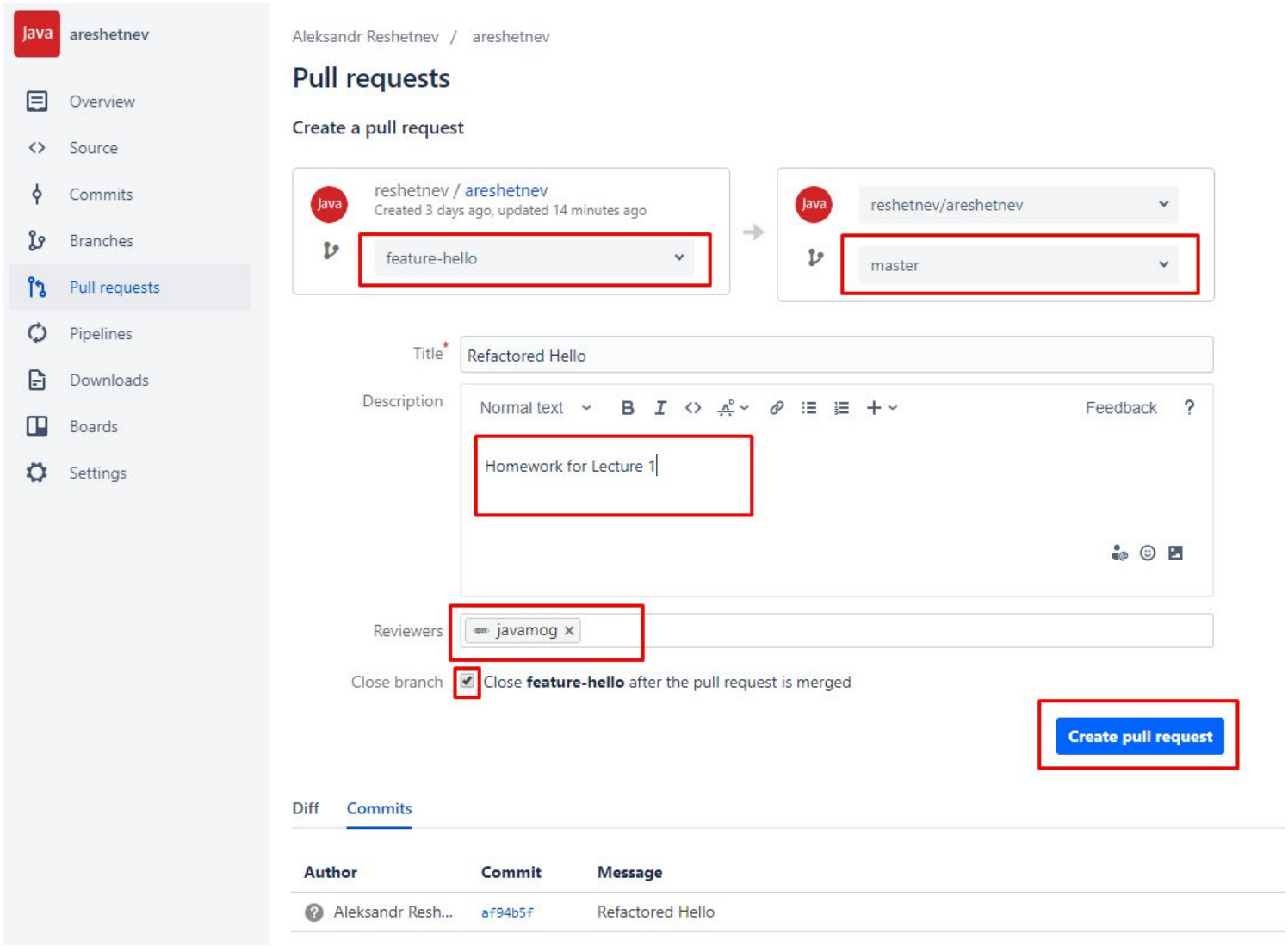


4. Pro Git

- Creating pull request

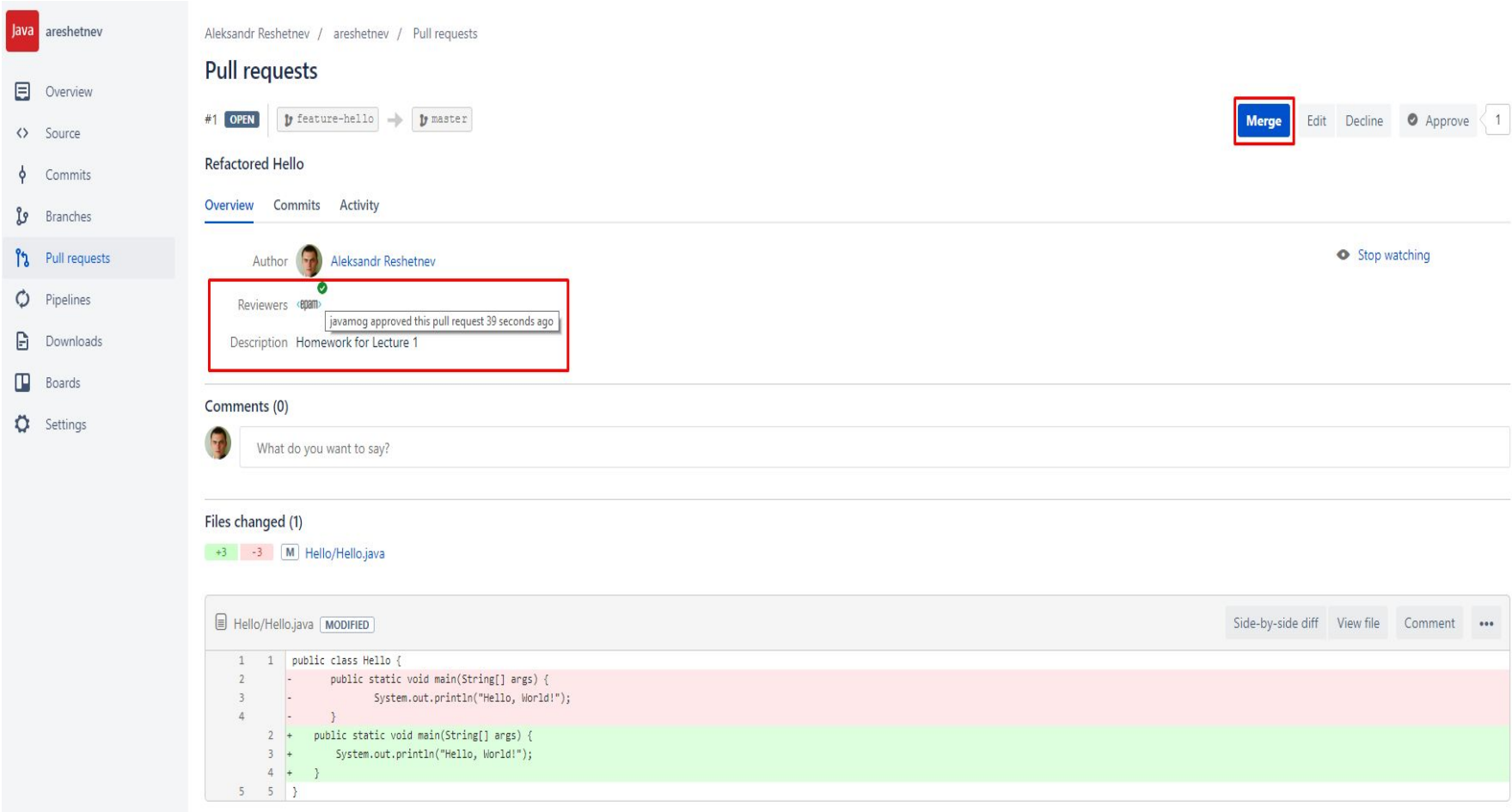


4. Pro Git


areshetnev
Overview
Source
Commits
Branches
Pull requests
Pipelines
Downloads
Boards
Settings
Aleksandr Reshetnev / areshetnev
Pull requests
Create a pull request
reshetnev / areshetnev
Created 3 days ago, updated 14 minutes ago
feature-hello
reshetnev/areshetnev
master
Title*
Refactored Hello
Description
Normal text
Feedback
Homework for Lecture 1
Reviewers
javamog
Close branch
Close feature-hello after the pull request is merged
Create pull request
Diff
Commits
Author
Commit
Message
Aleksandr Resh...
af94b5f
Refactored Hello

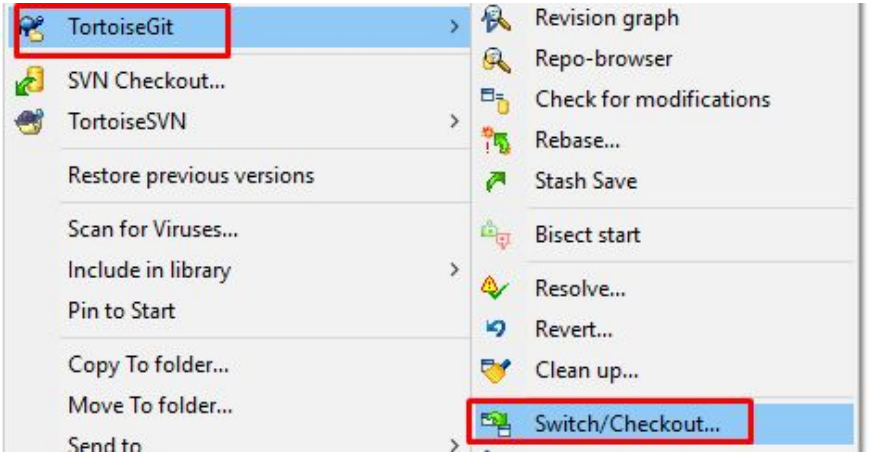
4. Pro Git

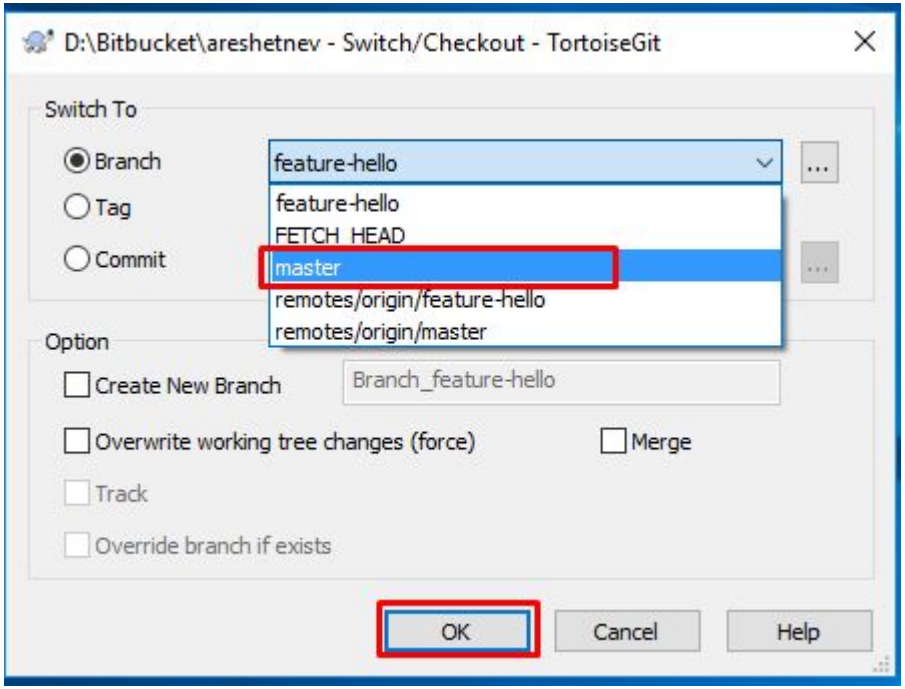
- Merge ***only approved*** pull request



4. Pro Git

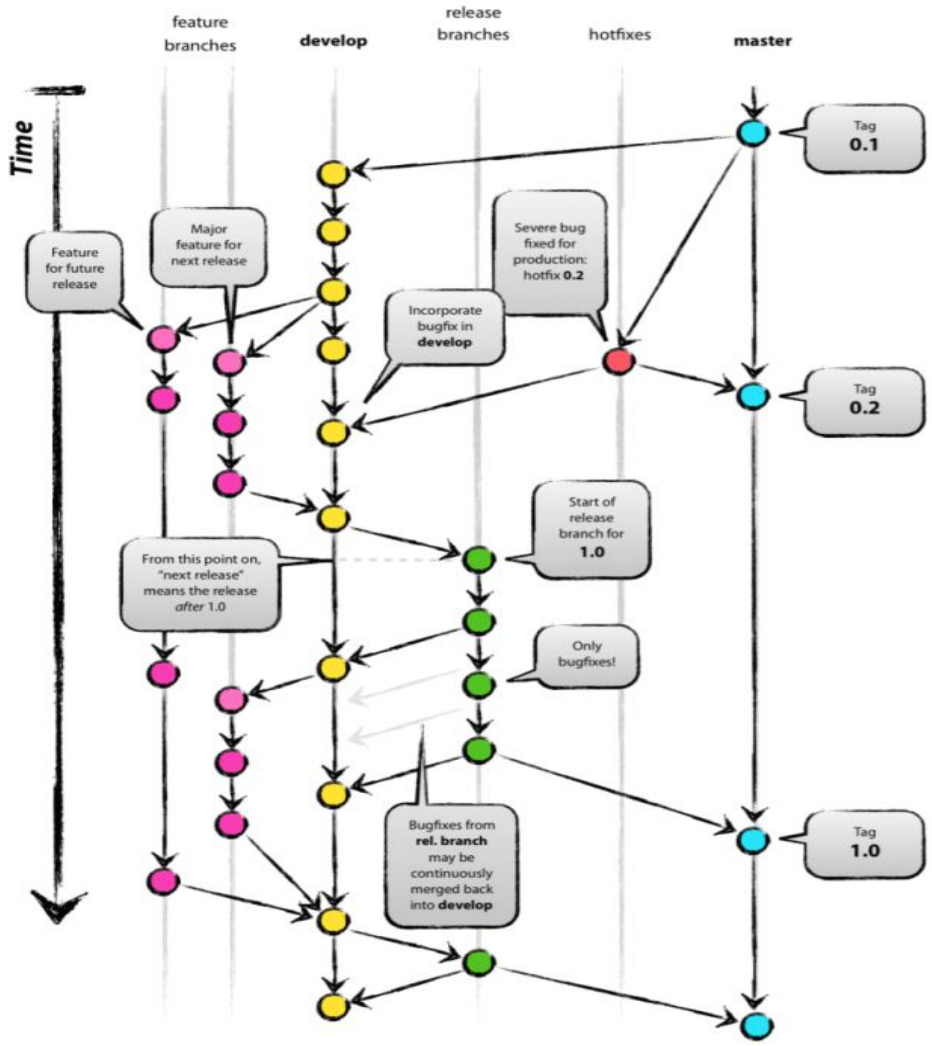
- Switch to *master* branch and update ***local repository***



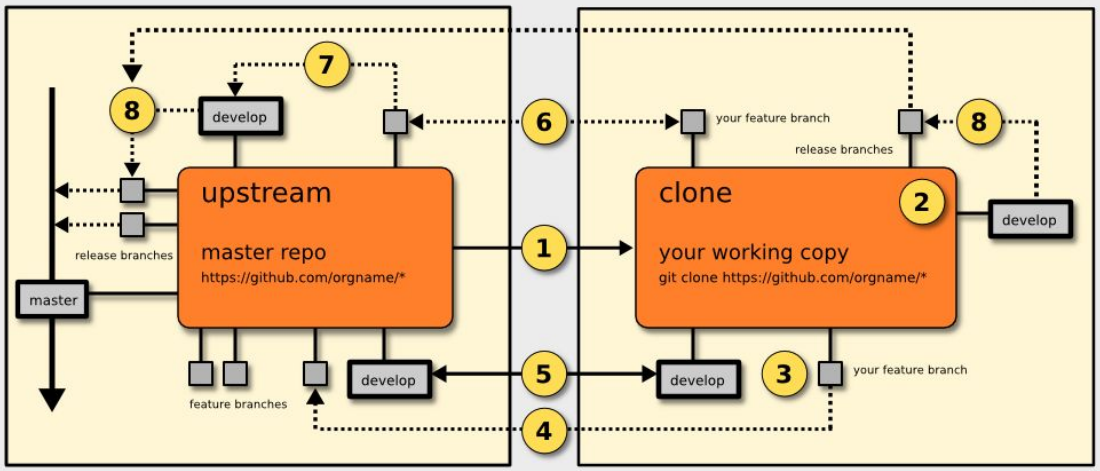


5. GitFlow

- http://nvie.com/posts/a-successful-git-branching-model/



6. HubFlow: GitFlow For GitHub



1 Clone the repo

```
$ git clone git@github.com:orgname/##repo##
```

2 Initialise the HubFlow tools

```
$ git hf init
```

3 Create a feature branch

```
$ git hf feature start ##feature-name##
```

- or checkout an existing feature branch

```
$ git hf feature checkout ##feature-name##
```

4 Push your feature branch back to GitHub as you make progress on your changes

```
$ git hf push
```

5 Keep up to date with completed features on GitHub on a regular basis

```
$ git hf update
```

6 Push your changes back to GitHub when you are ready to share them with collaborators.

```
$ git hf push
```

If you are collaborating with others, remember to fetch their changes back too!

```
$ git hf pull
```

7 When your feature is ready for review, it's time to merge your changes into origin/develop. First, push your feature:

```
$ git hf push
```

Use the GitHub website to issue a pull request to origin/develop from origin/feature/##feature-name##.

Ask a colleague to review your pull-request; don't accept it yourself unless you have to. Once accepted, close your feature:

```
$ git hf feature finish [##feature-name##]
```

8 When you have enough completed features, create a release branch, test and fix it, and then merge it into origin/master

```
$ git hf release start ##release-version##
<test + fix cycle>
$ git hf release finish ##release-version##
```

7. Useful Links

- https://tortoisegit.org/
- https://datasift.github.io/gitflow/index.html
- https://datasift.github.io/gitflow/IntroducingGitFlow.html
- https://datasift.github.io/gitflow/GitFlowForGitHub.html