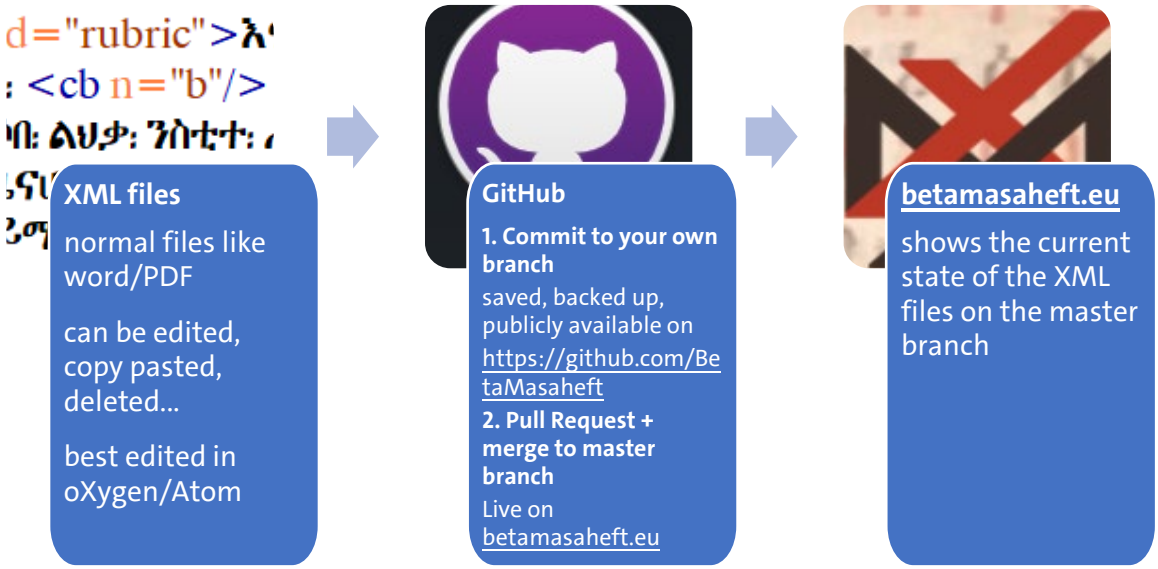
GitHub Workflow



GitHub Branches

- Every repository on GitHub has a master branch. From this branch, other branches can be created.
- Everything on the master branch is live on the app. **We don't work directly on the master branch**, we create a new branch for every unit of work.
- When we finish a unit of work on our individual branch, we make a pull request to merge our branch into the master branch. When it is merged, we delete our branch.
- If we continue to work, we create a new branch from the master branch and start the entire process again.
- We can also go to the master branch and create a new branch from it for our work while waiting for the changes on another branch to be approved and merged in a pull request.

Common reasons for branch confusion:

- GitHub is not synchronised.
- The branch was not created from the master branch.
- The branch was created from a deleted branch.
- The branch was created correctly, but not published.
- The changes were either not committed or committed, but not published.
- Remember that the branches for all repositories are completely separate. If you create a new branch to edit a manuscript, and then notice that you also need to edit a person record, you need to go to the persons repository and create a new branch there for your work in the person record.

If everything has been committed, all changes can retrieved and problems corrected, even if work was carried out on the wrong branch.

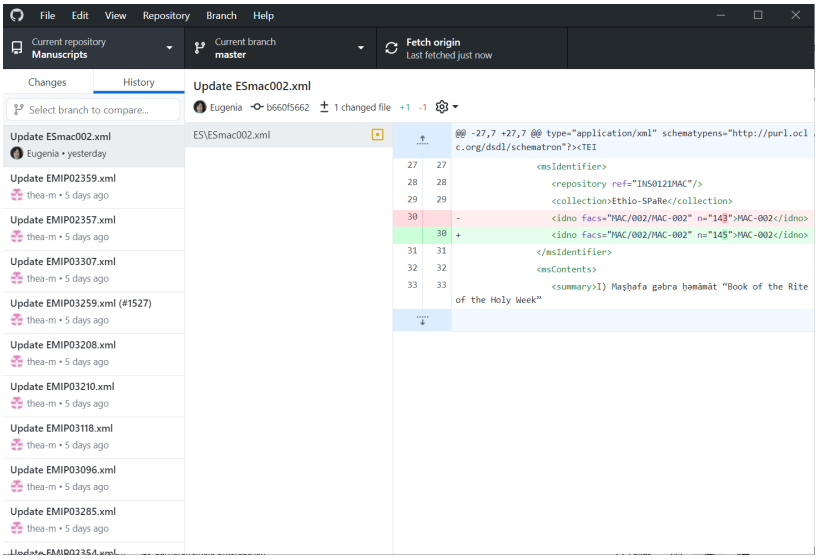
If a branch conflict arises, stay calm and try to analyse where exactly the problem occurred. You can always ask for help to figure it out and solve the conflict.

Remember that XML files are normal files: Many problems can be solved by copying a file to another location, deleting and pasting in the conflicted branch. As long as all changes are committed, it is also always possible to delete and reclone repositories without much trouble.

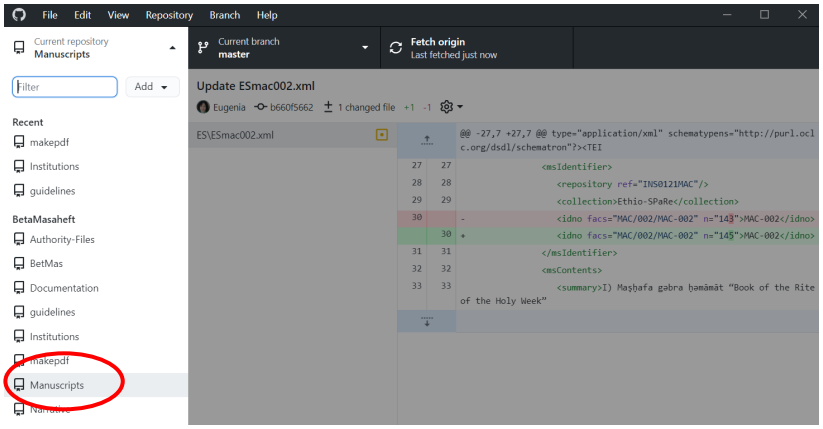
Editing XML files in the Beta maṣāḥəft digital research environment

This describes the normal workflow for working in the Beta maṣāḥəft digital research environment **after** the initial setup (https://betamasaheft.eu/Guidelines/?id=setup).

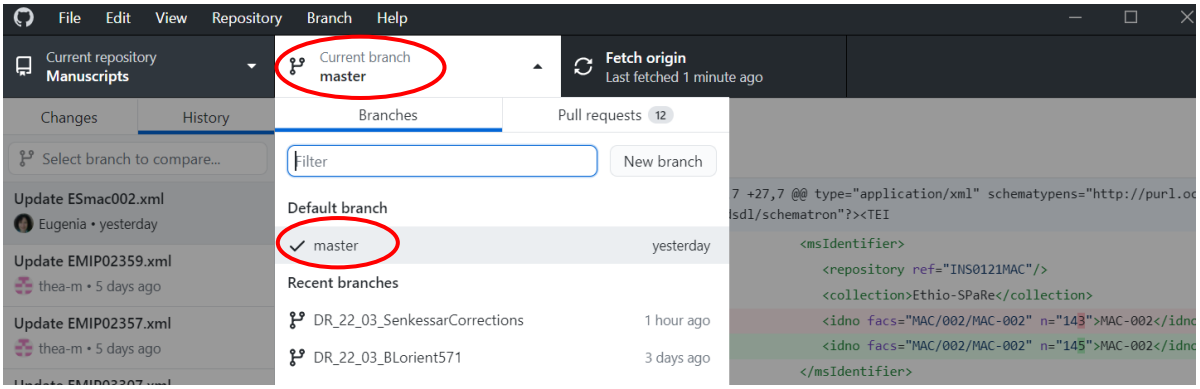
1. **Open GitHub Desktop.**



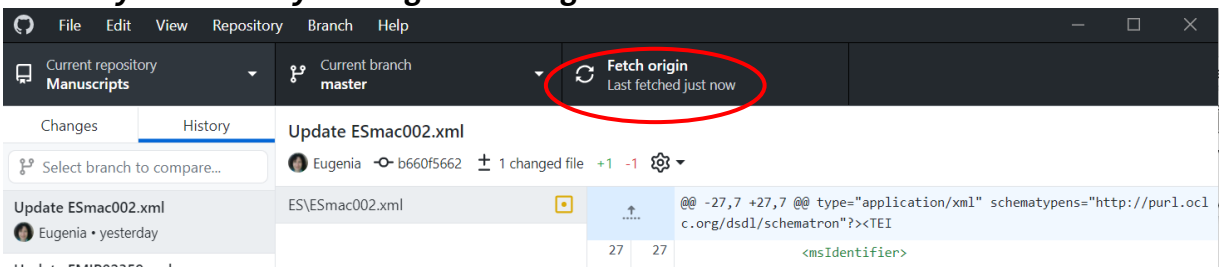
2. **Go to the repository you want to work in.**



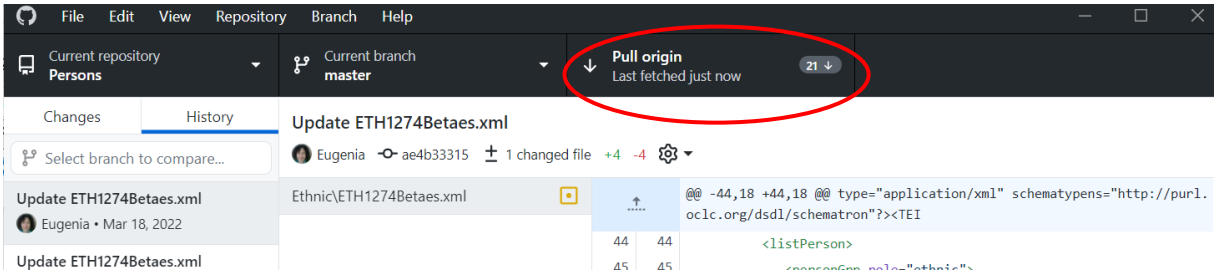
3. **Make sure you are on the master branch. If not, switch to the master branch.**



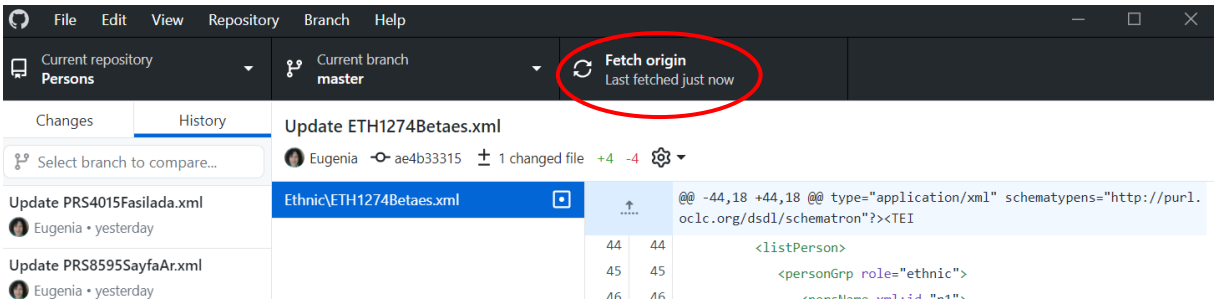
4. Synchronise by clicking “fetch origin”.



If there are new changes to this branch since you last synchronised, the button will change to “pull origin”, and you have to click it again:

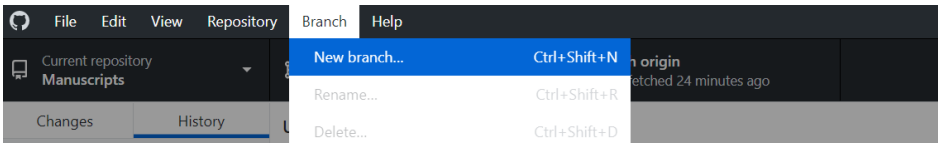


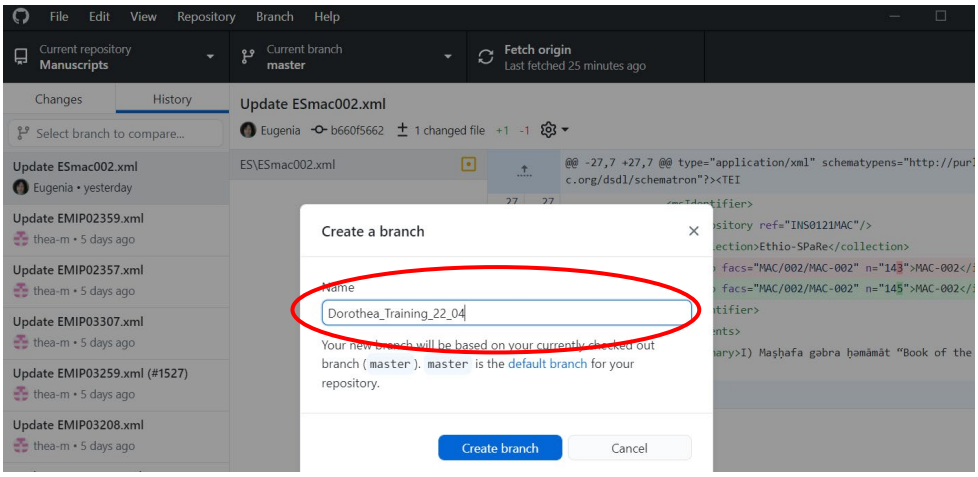
After it has been pulled, it should show “fetch origin” again and you are synchronised.



5. Create a branch.

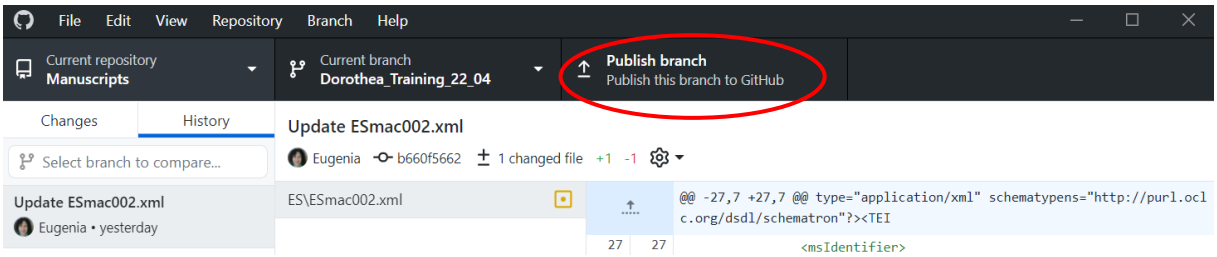
Give your branch a readable, clear and searchable name.





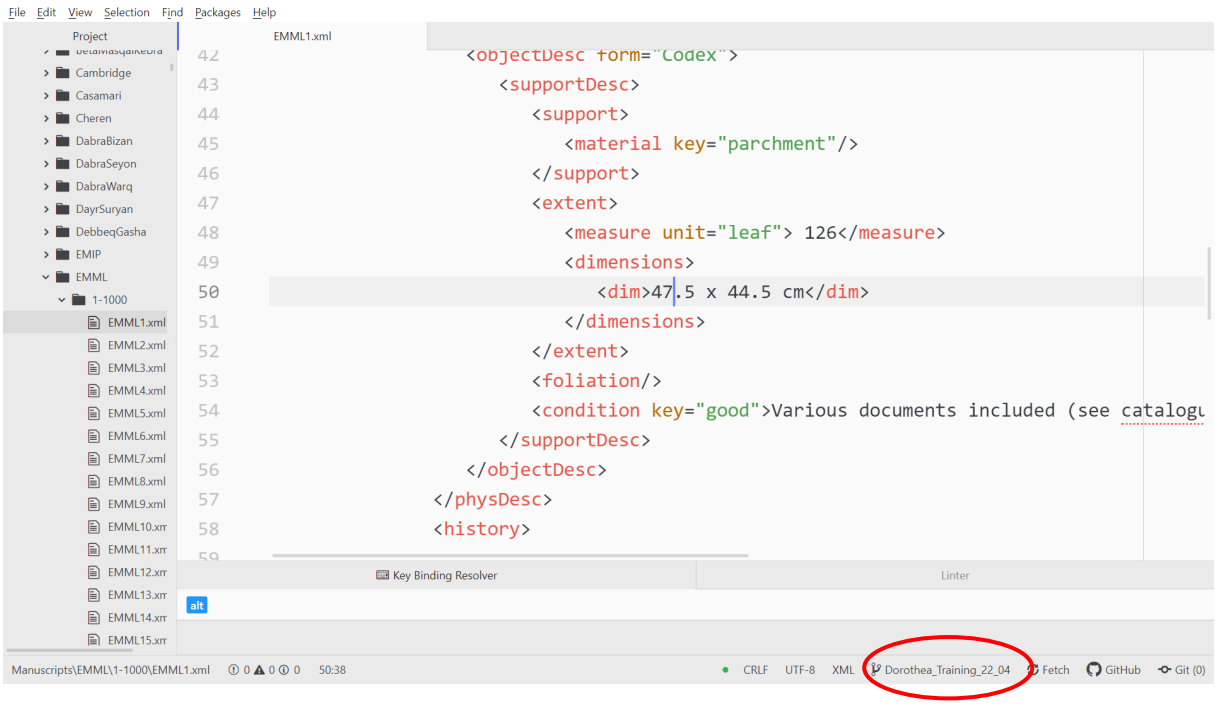
6. Publish the branch.

"Publish" means that this branch is available for everyone to see on the Beta maṣāḥəft GitHub organisation and that everything committed to this branch is saved and backed up – it doesn't mean that the changes on this branch are live on betamasaheft.eu.

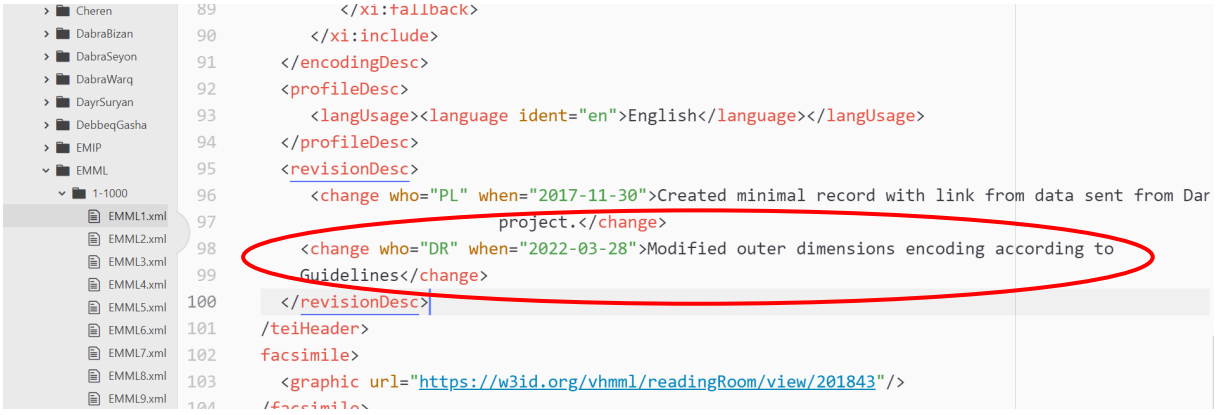


7. Edit the XML record(s).

Open your editor and open the file you want to work with. You will be automatically on the branch on which you're in GitHub desktop. If you work with Atom, check again that you are on the right branch.

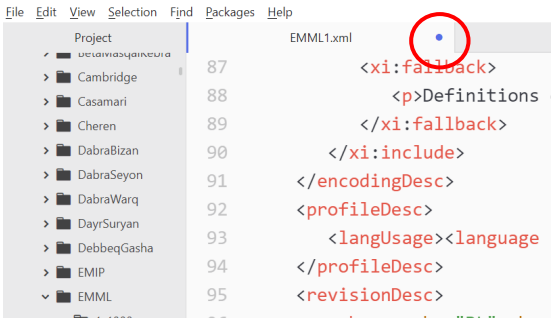


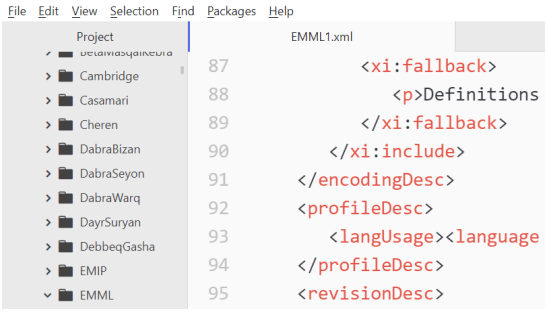
8. Add a change element after each edit.



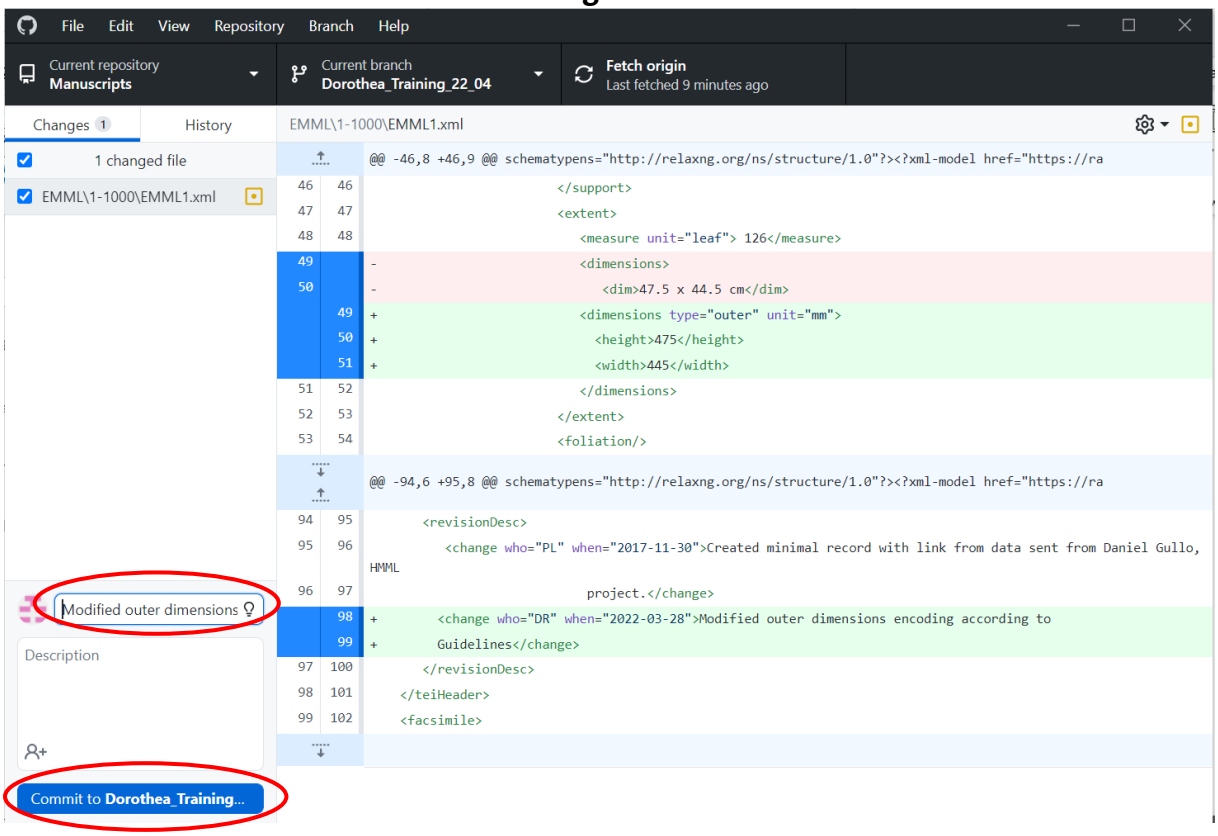
9. Save.

Unsaved files have a blue dot in the upper corner, saved files don't:



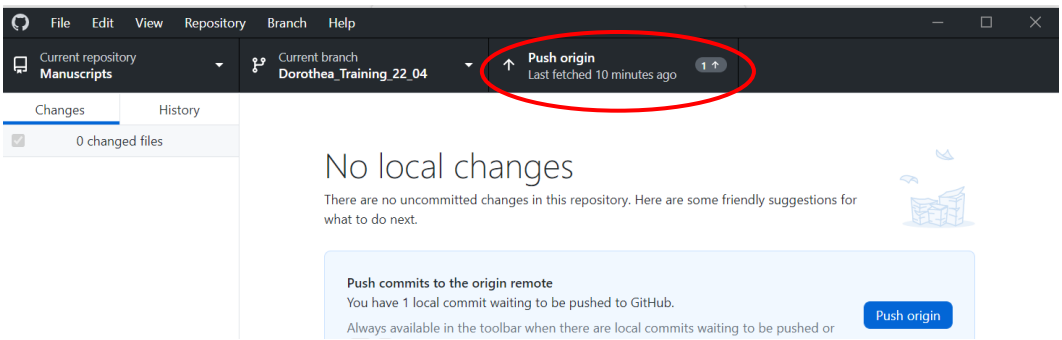


10. Go to GitHub desktop, and check in the view there that all your changes are saved. Write a clear commit message.

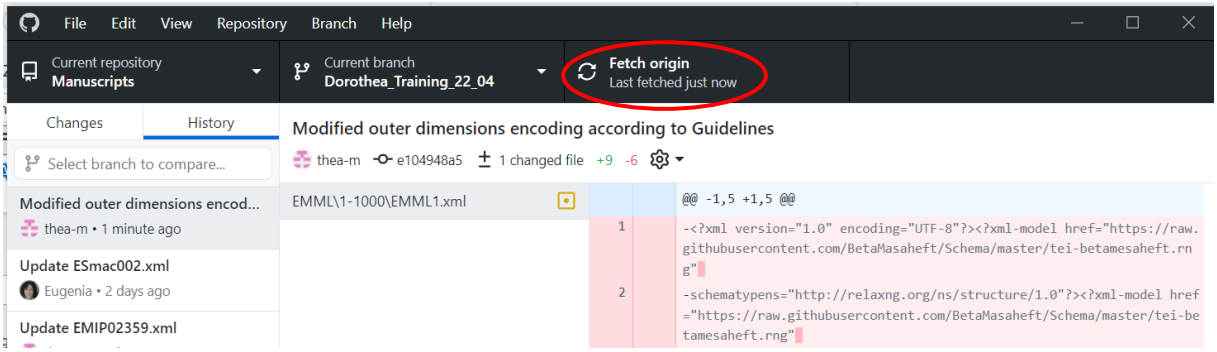


11. Click on "Commit"", then on "Push origin".

If you only click "Commit" and not "Push origin", the change will only be saved on your computer, and this can lead to branch confusion.



After having clicked “Push origin”, you should see “fetch origin” again. This means that your change has been synchronised. It should now be listed in the “History” tab.

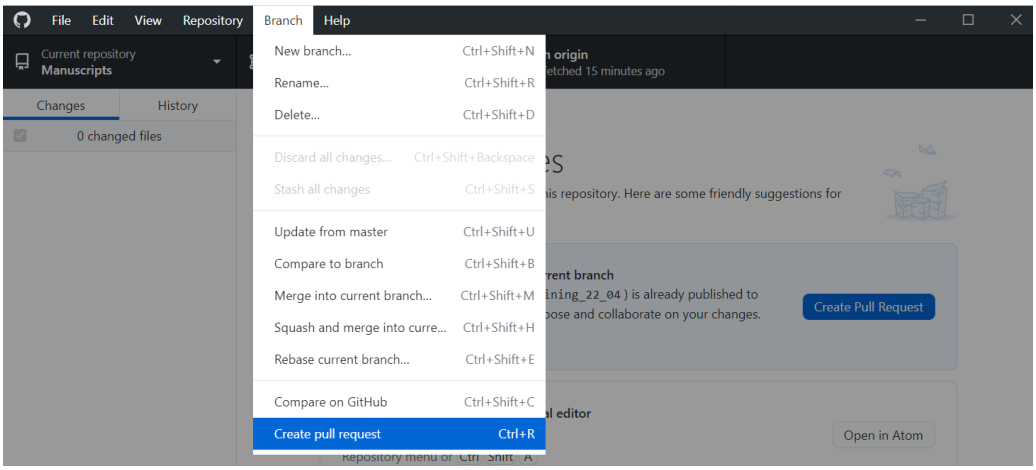


Now you have two possibilities:

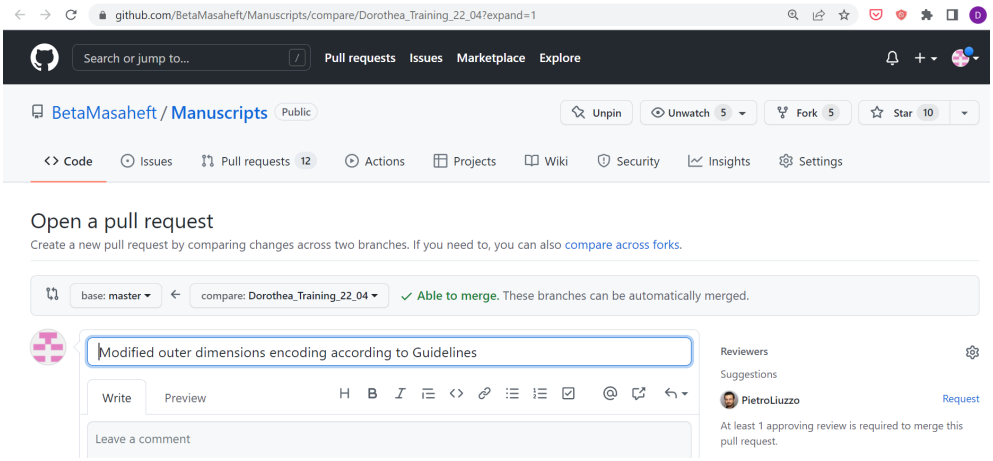
- make a pull request (**step 12**) to merge this change into the master branch,
- or continue to work on the same branch (go back to **step 7** – always making sure that you are on the correct branch and “fetch origin” before you start working again), repeating the same steps until this unit of work is finished and you make a pull request.

At the beginning of your work with GitHub, we advise to make rather many small pull requests than to work for a long period of time on your own branch without making a pull request.

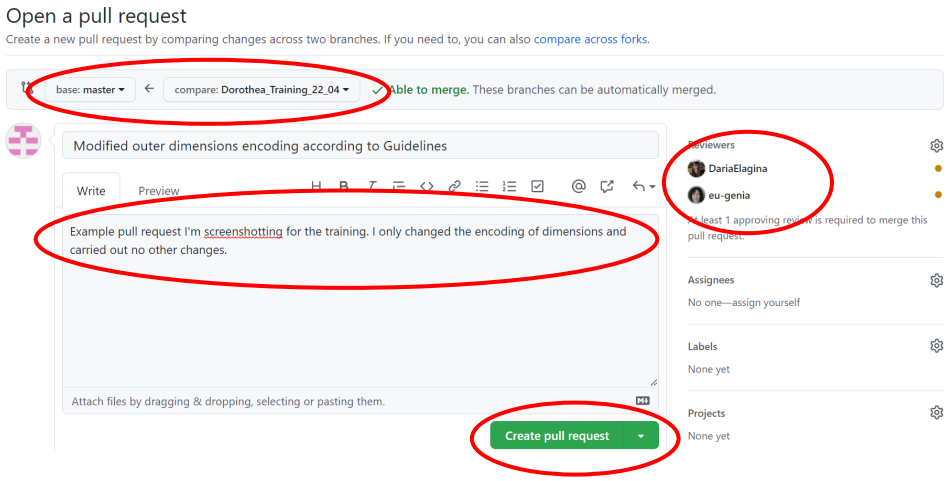
12. If you want to merge your changes into the master branch and see them on betamasaheft.eu: make a Pull Request.



13. **After clicking on “create pull request”, a new tab in your web browser will open:**



Check that the base branch is the master branch, and that the branch you’re comparing with it is the branch you were working on. You can now add further messages to clarify what work you carried out and what you want to be reviewed. Then, assign reviewers.



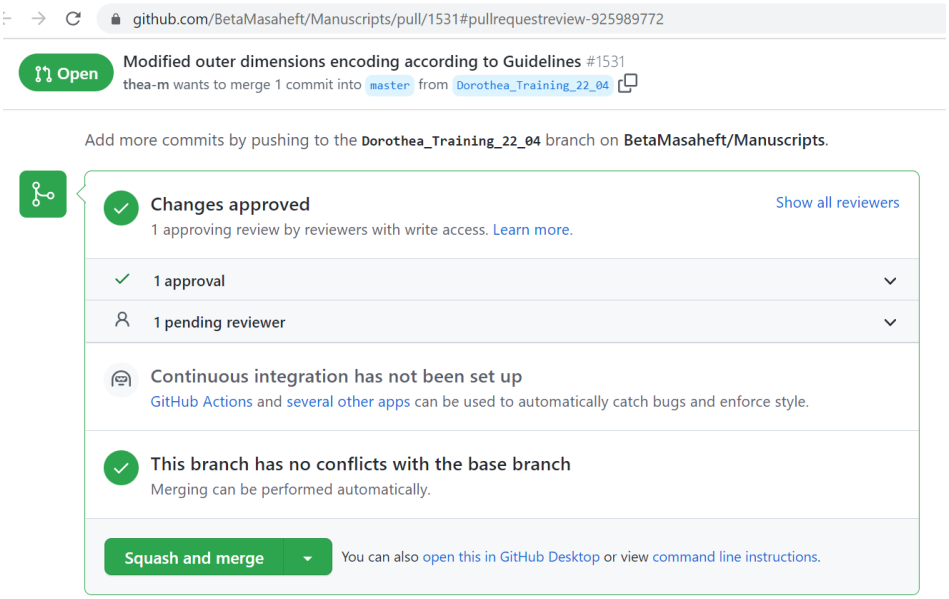
Click on “Create pull request”. The pull request is publicly visible to everyone. Your reviewers will receive an email.

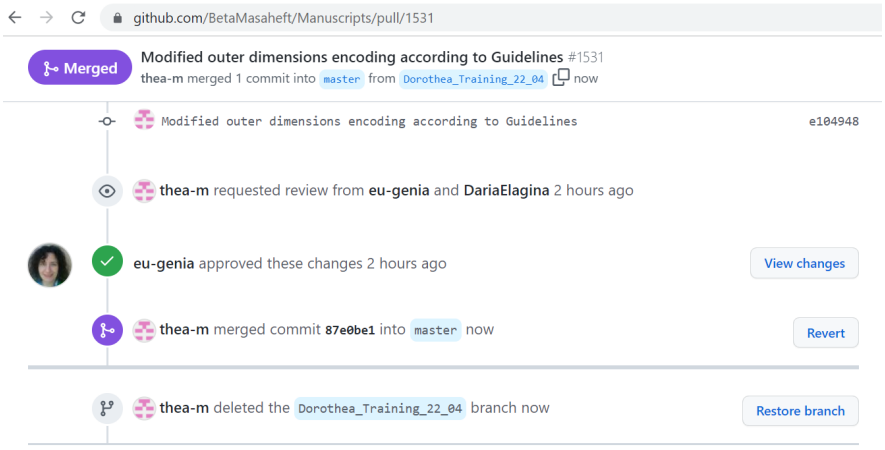
14. **Review process: Your reviewers read and evaluate the changes.**

They can:

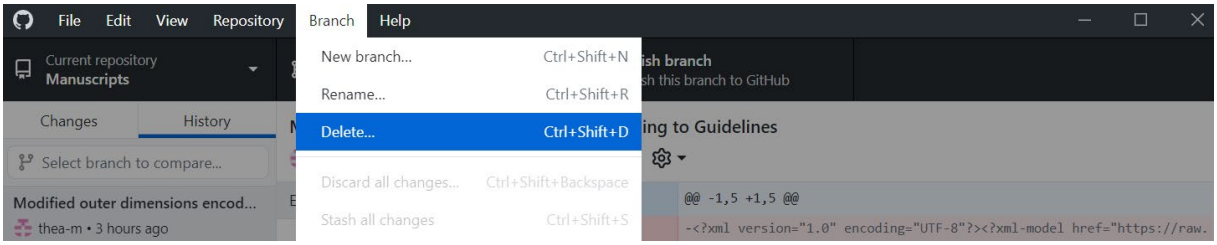
- “approve” the changes: they will merge the branch with its changes into the master branch, delete the branch created for the pull request. You don’t have to do anything.
- “request changes”: This normally happens. If you agree with the suggested changes, go back to step 7 and carry out the changes on your branch in the files, save and commit them to the branch. The reviewers will receive an email for all commits to this branch. If you don’t agree with the suggested changes, you can discuss this with the reviewers in the pull request until you reach a joint decision. In the end, the reviewers will approve the changes, merge the branch, close the pull request and delete the branch.

Sometimes, there are conflicts with the base branch. These can be solved in the web editor and are usually no big problem. You can contact us if this happens and is confusing.





15. **Delete the branch after it has been merged into the master branch both in the web browser and on GitHub desktop.**



16. Go back to the master branch and “fetch origin”, then “pull origin” so that the master branch on your computer is updated with the changes from your pull request: