DSKC GitHub Guidelines

V01 - SEPTEMBER 2024

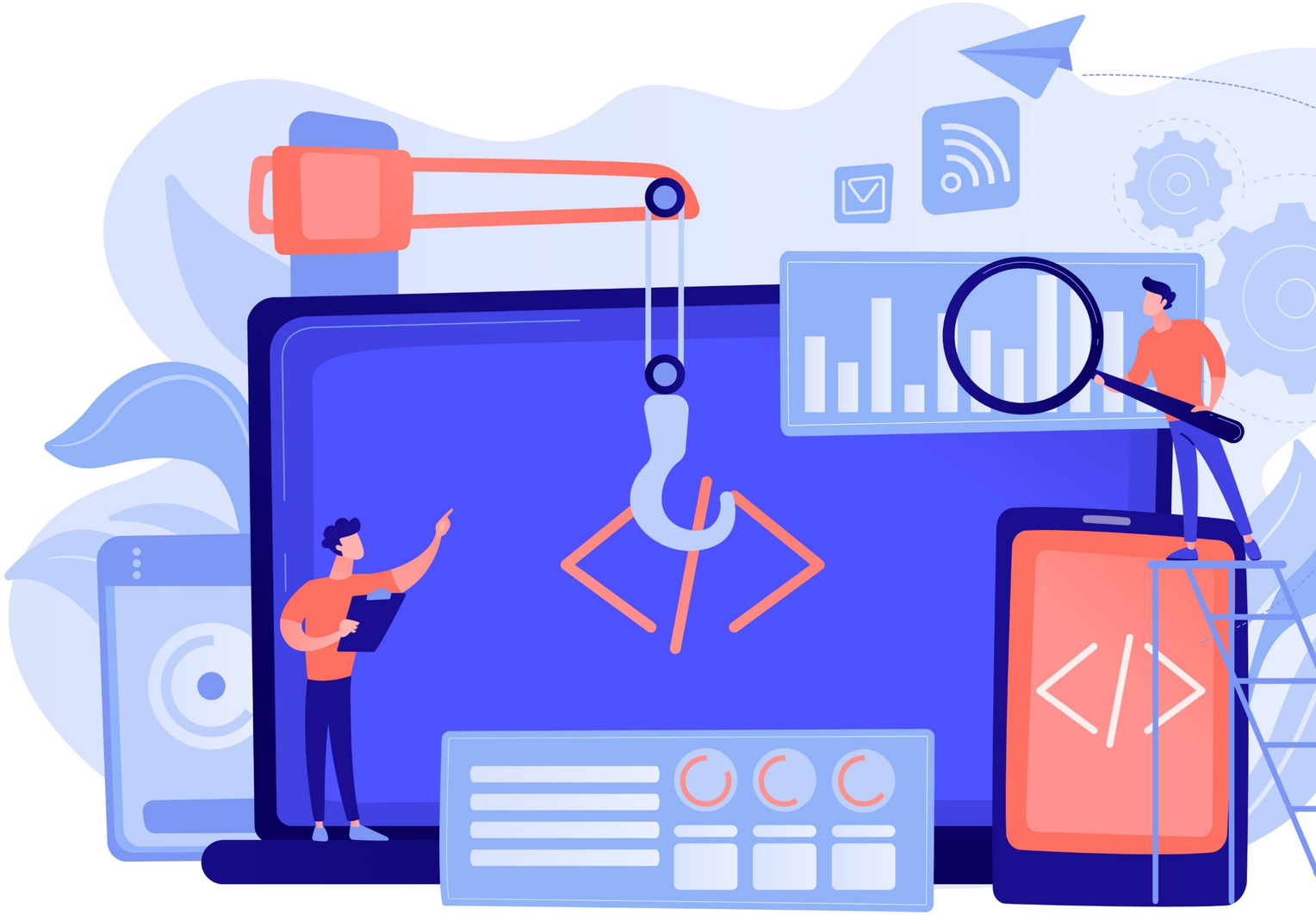


Image: freepik.com

Content

Introduction & Purpose

1. Why?

Thank you so much for taking the time to read this document, which is meant to be a live document presenting and defining general rules and best practices in managing Data Science Knowledge Center (DSKC)'s Git repositories.

2. For what?

To build a centralized, organized place to store code associated with different DSKC initiatives in a way that fosters reproducibility and collaboration, while keeping the code safe and preserving sensitive information.

3. To whom?

This document is meant primarily for DSKC's staff and affiliated faculty, as well as students directly working with DSKC staff and faculty.

Git & GitHub in a nutshell

GIT

Git is a system that stores programming code and has several key advantages:

- **Collaborative:** Multiple people can work on the code simultaneously. Git provides tools to merge these contributions and resolve any potential conflicts efficiently.
- **Reversible:** Every version of the code is saved, so if something goes wrong, you can easily go back to an earlier version.
- **Safe:** Git ensures the integrity of the code with built-in mechanisms. Additionally, because it's distributed (each collaborator has a full copy of the project), it’s highly unlikely that all the code will be accidentally lost.

GITHUB

GitHub is a popular online platform that hosts and manages Git repositories, allowing developers to collaborate on projects more easily. It provides a user-friendly interface for working with Git, enabling teams to store, share, and track their code in a centralized location.

Beyond just storing code, GitHub offers features like issue tracking, pull requests, and project management tools, making it easier for teams to organize their work and contribute to projects. It also allows for organization accounts, where multiple members may collaborate.

If you are new to Git and Github or need a refreshment, we strongly encourage this tutorial from freeCodeCamp, which is also a good resource to suggest to students starting to use Git.

(very quick) Glossary

1. **Repository:** A storage location for the project, containing all your files, history, and metadata. It’s the core of any Git project.

2. **Remote repository:** The version of the repository stored in GitHub.

3. **Local repository:** The version of the repository stored in a local computer.

4. **Commit:** A snapshot of the project at a particular moment in time.

5. **Committing:** Submitting a particular version of the repository to Git.

6. **Branch:** A "version" of the repository where you can work on changes without affecting the main project. It lets you try out new ideas or features safely, and once you're happy with the changes, you can combine them back into the main project.

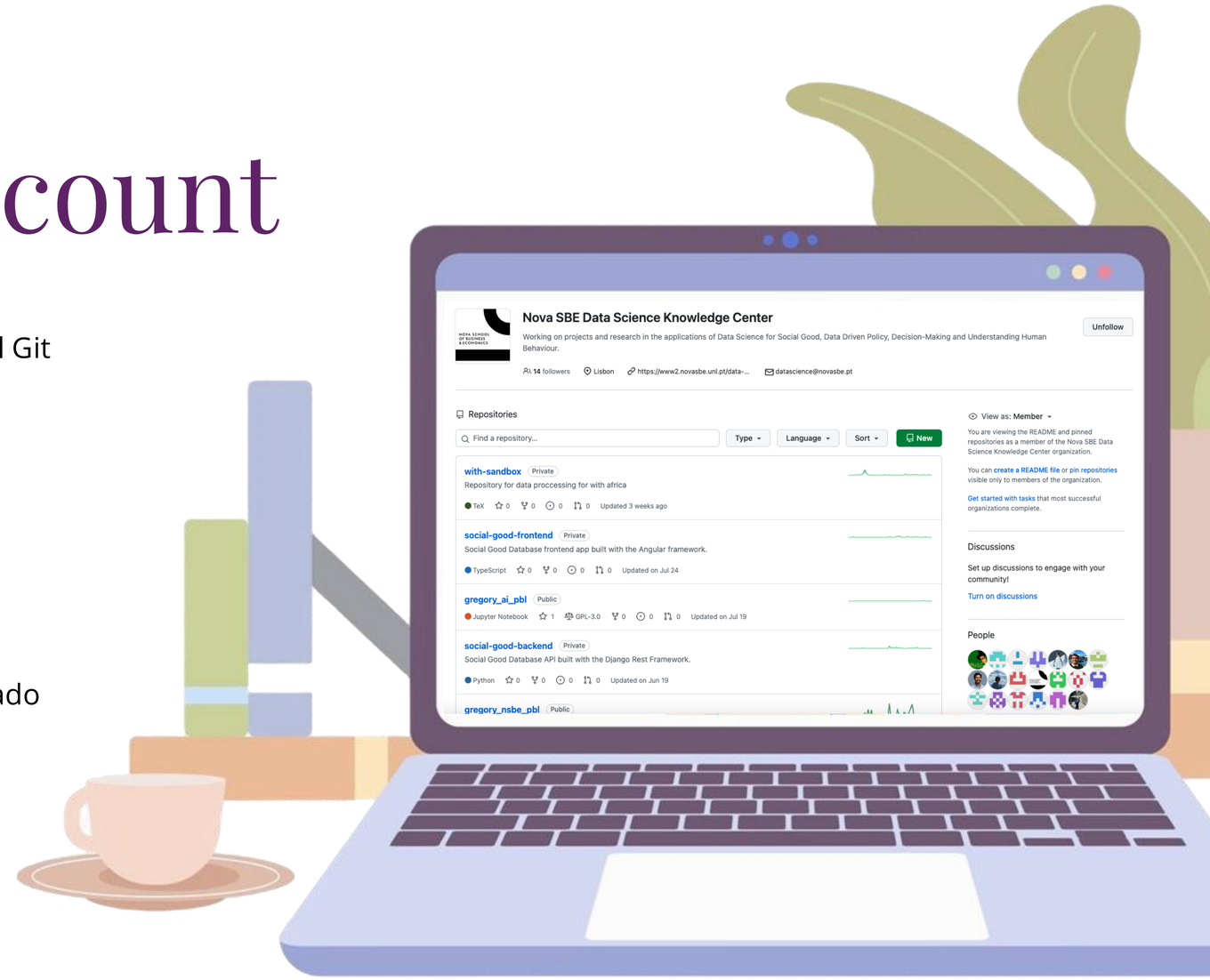
7. **Pull request:** A request to merge the changes you've made in one branch to be added to another branch, usually the main project. Typically, a member of the team reviews the work before merging the branch into the main project.

8. **Fork:** A copy of someone else's repository. It lets you make changes to a project independently without affecting the original version. Then, you can request that the original project owner review and potentially merge your changes. Forks are often used when contributing to open-source projects.

The DSKC GitHub account

DSKC has a paid organization account that is meant to be a place to store all Git repositories in a centralized fashion (learn more here).

You may find the DSKC organization account here.



Access to the account and privileges

Any person affiliated with DSKC (staff, faculty, and students) may be invited to join the DSKC GitHub account and become **a member**.

As of September 2024, Lénia Mestrinho, Bruna Riboldi and Susana Lavado own the DSKC account. They can invite members and have full access (Admin) to all repositories on the page, even if they are created as private (access protected by an NDA).

Any member of the DSKC account may **create repositories** within the page and invite other people to that repository (DSKC members or not).

Image: freepik.com

DSKC account procedures overview

1. The PM or the PI asks account owners (Susana Lavado or Bruna Riboldi) to add members to the DSKC account and place them in a team.
2. The PM or the PI creates the repository according to the specific privacy level and the name specifications. We encourage adding a readme and a *.gitignore* file.
3. The PM or the PI adds repository topics.
4. The project team updates a repository readme.
5. The PM or the PI archives the repository after the project is finished

>> Please see the next slides for detailed instructions on each of these steps.

DSKC detailed account procedures (1/2)

Managing members

If you need to add members to the GitHub page, please reach out (e-mail or Teams) to Susana Lavado or Bruna Riboldi. **In the request, please indicate:**

- The username or email of the person to be added.
- The affiliation of the person to the DSKC.
- The initiative where the person will collaborate. Common initiatives include but are not limited to, master thesis, PBL, and DSKC projects.

The new member should be added to a new or pre-existing team (see next box).

Teams

To facilitate permission management, **members should be added to teams.** The member should be directly invited to the team by the Owner who sends the invite. When applicable, **teams should be nested within other teams** (e.g., all PBL teams should be nested within the PBL team)

The team should also include the DSKC staff or faculty who requested the members to be added, which will be assigned the role of team maintainer.

The team description should include the initiative where the members are collaborating and the academic year (if applicable).

Members not assigned to teams will be removed.

Repository access

All code created within the context of DSKC projects should be **committed to a specific repository** on the DSKC account.

Every repository should be created by the PM or PI* as either **public or private**, depending on the nature of the project and any legal agreements. The privacy level must be validated by Lénia Mestrinho.

Public repositories should also indicate the type of License governing its usage. Learn here how to do so.

All members of a team will have the same level of permissions to a given repository.

Non-members of the DSKC account (e.g., clients, partners) may be added to a specific repository as external collaborators.

* Project Manager, Principal Investigator

DSKC detailed account procedures (2/2)

Creating a repository

Repositories, and especially those built under specific initiatives (e.g., PBL, thesis*) should comply with the following rules:

1. Creation of the team collaborating in the project **under a parent team for the initiative**. Team name should also include a reference to the initiative and the year it is being developed.
2. Attribution of a **team maintainer role** to the staff or faculty member responsible for the students.
3. Including the **initiative name and year** in the repository name (e.g., PBL_2024_reponame).

*Hosting master thesis and CollabDSKC projects code in DSKC GitHub account is encouraged but optional and should be discussed with the responsible faculty or staff.

Repositories topics

Although this is not Github topics intended use, we strongly **encourage that each repository is tagged according to the DSKC initiative it belongs to.** This facilitates the process of knowing the provenience of each repository and finding repositories. See how to add topics here.

Please **use preexisting topics**. As of September 2024, the available tags are:

- theses
- consultancy-project
- research-project
- support-repository
- pbl
- collab-dskc
- social-database
- Blockchain-prr
- with
- research-project
- Iefp
- data4change

Every month, the repository tags will be checked and maintained.

Repositories best practices

Repositories should have a **clear and complete readme**, indicating the purpose, context and team of the project. See readme best practices here.

When the repository includes the public release of artificial intelligence (AI) models, those models should **follow principles of transparency and accountability**, which includes documenting their weights, the model architecture, and the expected model usage, as defined in the European Union AI act.

Repositories of projects that have ended and for which no further developments are anticipated **should be archived.**

Security & Privacy (1/2)

AVOIDING COMMITTING SENSITIVE INFORMATION

Credentials

The most typical security breech happens when people submit credentials (e.g., database username and password) to GitHub. To avoid this, make sure that you do not type your credentials directly to the code. Instead, create a separate file with your credentials that is only stored in your computer (and never committed to GitHub).

.gitignore

To avoid submitting files containing sensitive information to Git, we recommend you use a .gitignore file. This file indicates to Git which files you do not want to submit to Git. Read more here.

Data

Even when the code might be publicly shared, you need to take extra to make sure you do not submit private data to a repository.

There are two typical ways people accidentally submit data to a repository. One is to submit files containing data. The other is to submit Jupyter Notebooks outputs (plots, tables) to the repository.

Also, make sure you do not accidentally submit sensitive information (e.g., partner names or details) by including it in comments or descriptions.

Clearing Jupyter Notebooks outputs before committing them

You can manually clean the Jupyter notebooks outputs (in the cell menu). However, experience shows us this is easy to forget. We recommend you make sure that the outputs are cleaned before submitting by following this tutorial.

Security & Privacy (2/2)

DECIDING ON THE LEVEL OF REPOSITORY PRIVACY

GitHub repositories may be private or public.

- Public repositories are accessible to everyone on the internet.
- Private repositories are only accessible to people/teams with explicit access and the DSKC admins.

Usually, the level of privacy of a repository is pre-agreed between the project stakeholders (e.g., clients, partners, Professors and the project team). Such agreement may be defined in legal agreements of the project.

The **PM or PI is responsible for creating the repository** and defining its level of privacy, which should be validated with Lénia Mestrinho.

Repository members can change the level of privacy of a repository at any time. Before making a repository public, the PM or the PI should **confirm with any relevant stakeholders it does not contain any sensitive information and can be published.**

TWO-FACTOR AUTHENTICATION

All account members should use a 2-factor authentication as an extra layer of security when accessing GitHub. You can learn how to set up 2-factor authentication on this GitHub page. You can use Microsoft Authenticator (which you should already have installed associated with your Nova account) as your time-based one-time password (TOTP) app.

Account safeguards – What to if...? (1/2)

1 SENSITIVE INFORMATION WAS ACCIDENTALLY COMMITTED

Once information is committed to Git, it will stay forever in its history, even if it is later changed or deleted. Simply deleting sensitive information in subsequent commits will not remove access to the repository version where the sensitive information is.

The best way to deal with a commit of sensitive information is to go back in the repository history and delete the critical commit. Another option, which is not ideal and should only be used if no element of the team wants to keep its history, is to delete and recreate the entire repository.

If you committed your credentials, please make sure you change them immediately.

If you need help, please reach out to Susana Lavado or Bruna Riboldi.

2 A REPOSITORY WAS ACCIDENTALLY DELETED

DSKC Git account may restore accidentally deleted repositories up to 90 days after their deletion (with a few exceptions).

Please contact Susana Lavado or Bruna Riboldi to restore the repository.

Account safeguards – What to if...? (2/2)

3 WRONG USER WAS ADDED AS A DSKC MEMBER / TO A TEAM

Please be extra careful when giving usernames to be added as DSKC members and to teams, to avoid leaking private information or having non-authorized accounts with power to change repositories.

If the wrong user was added to DSKC, please contact Susana Lavado or Bruna Riboldi as soon as possible.

If the wrong user was added to a team, please contact the team maintainer as soon as possible. If the team maintainer is unavailable, please contact the account owners immediately. You can learn how to remove users from teams here.

Please note that removing a user from the account or the team will not delete any local copies of the repository the person may have done.

Other Frequently Asked Questions

MAY I SUBMIT MY PRIVATE PROJECT REPOSITORY TO DSKC ACCOUNT?

No. Only repositories built within a DSKC initiative should be hosted in the DSKC account.

MAY I MIGRATE A REPOSITORY FROM MY PERSONAL ACCOUNT TO DSKC?

Yes. If you have a repository in your private account that belongs to the DSKC account, you may transfer it.

SHOULD I ADD PUBLIC DATA TO MY REPOSITORY?

No. You should not store large files in GitHub. Even for small datasets, we strongly recommend to keep all data files in other NSBE repository: figshare, or in the DSKC databases.

HOW TO ADD CODE INTO AN EXISTING PARTNER REPOSITORY?

The best option might be to fork the original repository into DSKC account, and then submit a pull request into the partner repository. The forked repository should stay in DSKC account,

STILL NEED HELP?

If your question was not answered in this document or you need help with Git, please reach out to Susana Lavado or Bruna Riboldi, via e-mail or teams.

General Git repository best practices

We have selected a set of public, external resources to help you make the best out of a Git repository:

- https://gist.github.com/luismts/495d982e8c5b1a0ced4a57cf3d93cf60
- https://acompiler.com/git-best-practices/
- https://nulab.com/learn/software-development/version-control-best-practices/

Have further suggestions to improve the DSKC GitHub account and these guidelines?

Please get in touch!

susana.lavado@novasbe.pt
data.science@novasbe.pt