Pull Request Guidelines

The purpose of this document is to provide a unified criteria on how to create and submit PRs.

Before creating a PR: our branching model

Our internal GitFlow is organized in branch folders. We have three different folders: feature, bugfix, and hotfix.

Branches created to work on new features (i.e. stories and tasks) should go to the feature sub-folder as follows: `git checkout -b feature/JIRA-TICKET-NUMBER-[minimal-description]`. For example, `git checkout -b feature/M3TA-1234-title-styling`. Bugs and hotfixes follow the same naming convention but use “bugfix” and “hotfix” as folder names.

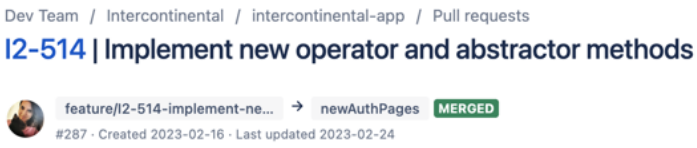
General guidelines

1. **Create small PRs**
   Avoid creating huge PRs with lots of modified files. Big PRs are hard to code review and increase the chance of small issues being missed by the reviewer.

   PR descriptions should be short and concise. The automatic list of commit messages that Bitbucket displays in the description field can be used as long as the list of commits is short and gives a proper description of what will be seen in the code review.

   ⚠️ Commit messages should never be longer than a line. However, vague messages such as “updated code” should be avoided.

2. **Use nice PR names**
   Kindly rename the name Bitbucket automatically assigns to the PR so that the corresponding Jira ticket is easily linked and quickly identified when opening the PR. For example:

3. **Avoid PR dependencies**
   A PR should almost never depend on another PR to be merged. The scope of each story, task, subtask, bug, etc. should be clearly delimited in order to avoid these situations. If you find

yourself in this situation, check with your Tech Lead how to slice the issues in order to prevent PR dependency.

4. **Avoid conversations on PRs**
   Your reviewer might leave comments on your PRs for you to take action. If these comments are unclear for you, avoid replying to them on the PR. This type of conversations need to be documented on the corresponding Jira ticket so that the entire team can easily access that information at any time.

   Your reviewer might also give oral feedback and request specific changes for each file of your PR. Remember to leave notes in comments, documenting these requests above each respective file of the PR to indicate the changes that were requested and therefore, streamline the process and make it easier for your reviewer to remember what was originally asked.

5. **Be mindful of the ticket's scope**
   When implementing a feature, you might realize that code unrelated to your tickets scope is buggy. This should definitely be addressed and fixed, but not in the branch you are working on.

   ⚠️ Do not include code that modifies or fixes anything else except for the ticket's requirements. Modifying files that are not in scope could lead to regressions in other dependent code, and QA needs to verify that this does not happen. This is true even when you think you're just fixing a bug:
   a. If you detected a bug somewhere in existing code, ask QA to open a bug issue in Jira describing it and have the code fix implemented only in the PR for that new bug issue. In those rare occasions where the bug is too hard for QA to detect, or if's not really a bug but a technical debt (i.e. code that works but does not conform to our usual standards) then you should discuss the need for a new ticket with the project lead.
   b. ⚠️ **If you need to change the API of some existing component in order to implement the current ticket you should discuss the proposed API change with the project lead / architect before implementing it.**

6. **Documentation is part of the implementation process**
   Remember to check that the documentation of your code is up to date. If it isn't, kindly update it. Create new documentation when new files are created respecting the project's

documentation criteria (refer to coding guidelines).

7. **Do not confuse implementation with local testing tweaks**
   Do not push code that was merely modified for local testing. Do whatever you need to do to test your code locally, but avoid pushing any of these changes. Note, for example, that sometimes it is convenient when testing one ticket to use code that is part of another ticket that is being worked on. Do not include that code when you push the PR - every PR should only include its own code and not release code of other PRs in progress.

8. **TODO comments**
   Check all TODO comments in all of the files of your PR. Should these comments still be there? If the answer is yes, make sure that corresponding tickets in Jira exist. Otherwise, remove them.