**Git Branching strategy used in our company**

**Name branching convention**

We use a consistent naming convention for our feature branches to identify the work done in the branch. We also include other information in the branch name, such as who created the branch…

**Using feature branches for your work**

### Develop your features and fix bugs in feature branches based off your main branch. These branches are also known as *topic branches*. Feature branches isolate work in progress from the completed work in the main branch. Git branches are inexpensive to create and maintain. Even small fixes and changes should have their own feature branch.

**Reviewing and merging code with pull requests**

The review that takes place in a pull request is critical for improving code quality. Only merge branches through pull requests that pass your review process. Avoid merging branches to the main branch without a pull request.

### Keeping high quality, up-to-date main branch

The code in your main branch should pass tests, build cleanly, and always be current. Your main branch needs these qualities so that feature branches created by your team start from a known good version of code.

**How to create a pull request and merge it upstream.**

* On GitHub.com, navigate to the main page of the repository.
* In the "Branch" menu, choose the branch that contains your commits.
* Then above the list of files click on pull request
* use the base branch dropdown menu to select the branch you'd like to merge your changes into, then use the compare branch drop-down menu to choose the topic branch you made your changes in.
* Type a title and description for your pull request
* To create a pull request that is ready for review, click Create Pull Request. To create a draft pull request, use the drop-down and select Create Draft Pull Request, then click Draft Pull Request.
* Click confirm merge.
* Delete the branch to keep the list of branches in your repository tidy.