Name: Ivan Pryymak

Date: 8/31/2024

Assignment 3.2 – Version Control Guidelines

For this research paper, I read and analyzed these 3 articles:

1) <https://www.quora.com/What-are-the-best-practices-for-version-control-in-software-development-and-how-do-they-contribute-to-a-more-efficient-development-process>

2) <https://homes.cs.washington.edu/~mernst/advice/version-control.html>

3) <https://mudacodes.medium.com/a-beginners-guide-to-using-version-control-for-efficient-software-development-8d94f3cf4679>

Version control is crucial for collaboration on projects in software development. It allows the teams to work efficiently and maintain an organized codebase.

1. Best practices from Quora article:

According to the website, the most important practices include:

* Frequent commits – helps to keep a granular history of changes.
* Meaningful commit messages – provides context to commits, allowing easier understanding of commits.
* Use of branching strategies – using separate branches for features allows to keep the main branch isolated and stable.

1. Best practices from University of Washington website:

This article is highly focused on maintaining a clean and organized repository. The main points are:

* Avoiding large commits
* Keeping the main branch stable and always ready for deployment.
* Utilizing tags for each release.
* Each commit represents a single change.

These tips allow for a simpler code review process and easier roll back in case of a bad commit.

1. Advice from Medium website:

This article is aimed at beginners by providing general information of picking a version control system, setting it up, and understanding the basic concepts.

The main points of the article are:

* Setting up a repository
* Committing changes
* Branching and merging
* Using remote repositories
* Resolving conflicts
* History and time travel

Best practices include:

* Commit frequently, and make sure the commits are meaningful and self-contained.
* Meaningful commit messages.
* Keeping .gitignore file up to date to exclude unnecessary files form VCS.
* Pull updates from remote repository regularly to stay up to date.

None of the articles provide a practice that I could consider irrelevant today. But they do mention that having infrequent and large commits an outdated practice. Such an approach can significantly complicate the development and debugging process and makes it harder to see the progression of the project.

My guidelines

To sum up the best practices from these three sources, I consider the following guidelines essential for effective version control of any software development project.

* Frequent and small commits – ensures that changes are atomic, with each commit representing a single change. This also makes it easier to track changes and debug the code.
* Descriptive commit messages – informative messages provide context for the changes, making it easier to understand the reason for commit.
* Consistent branching strategies – helps manage the development of features separately from the main development branch. It also ensures that the main branch always remains in a deployable state.
* Using tags for releases – marking stable releases allows for easier roll backs.
* Continuous Integration – merging branches into a main branch regularly prevents accumulation of conflicts and lowers the overall complexity of code integration.

By following these guidelines for software development version control, the management and collaboration on a project can be significantly improved.