When researching **Version Control Guidelines**, I found three prominent sources that discuss best practices: **Nulab**, **Tower**, and **Atlassian**.

1. **Nulab** emphasizes the importance of branching strategies and naming conventions. They advocate for regular merging and pruning of branches to keep the repository organized and recommend maintaining a "green build" state to ensure that the main codebase is stable before any changes are integrated. They also stress the necessity of thorough reviews before changes are pushed to the main repository​([Nulab](https://nulab.com/learn/software-development/version-control-best-practices/" \t "_blank)).
2. **Tower** highlights a few core practices, such as committing related changes and committing often. They suggest that commits should be focused and small to ease collaboration and conflict resolution. The site also advises against committing incomplete work and encourages thorough testing before commits​([Tower Git Client](https://www.git-tower.com/blog/version-control-best-practices/)).
3. **Atlassian** provides a comprehensive overview, discussing the significance of using branches extensively for new features and bug fixes. They also advocate for adopting a common workflow among team members and emphasize the importance of writing clear commit messages ​([Nulab](https://nulab.com/learn/software-development/version-control-best-practices/" \t "_blank))​([Tower Git Client](https://www.git-tower.com/blog/version-control-best-practices/)).

**Comparison**

While all three sources emphasize branching and the importance of clear commit messages, there are some differences. For instance, Nulab focuses more on the "green build" concept, while Tower stresses the importance of committing often and not pushing half-done work. Atlassian takes a broader approach, discussing overall workflows and collaboration within teams.

**Relevant Guidelines Today**

Some guidelines, such as maintaining commit integrity and using branches, remain universally relevant. However, certain more rigid practices from the past, like strict commit message formats, may vary based on team preferences today.

**My List of Important Guidelines**

1. **Commit Related Changes**: This promotes clarity and organization. All three sources agree on the importance of making commits that encapsulate related changes. Nulab suggests that commits should be small and focused, ideally addressing one specific issue or feature. This practice makes it easier to identify bugs and roll back changes if necessary​ ([Nulab](https://nulab.com/learn/software-development/version-control-best-practices/" \t "_blank)). Tower echoes this sentiment, emphasizing that a well-defined commit structure improves clarity for all developers working on a project ​([Tower Git Client](https://www.git-tower.com/blog/version-control-best-practices/)).
2. **Test Before Committing**: This ensures stability and reliability. Testing code before it is committed to the repository is crucial. Tower advocates for this practice, stating that code should only be committed when it is thoroughly tested and free of errors​ ([Tower Git Client](https://www.git-tower.com/blog/version-control-best-practices/)). This guideline ensures that the main codebase remains stable, preventing issues that can arise from incomplete or buggy code.
3. **Use Branches**: This allows for parallel development without interference. Branching is a powerful feature in version control systems that allows developers to work on features or fixes independently from the main codebase. Nulab recommends extensive use of branches for new features, bug fixes, and experimental work. They note that branches help avoid conflicts and keep the main code clean and deployable​ ([Nulab](https://nulab.com/learn/software-development/version-control-best-practices/" \t "_blank)). Tower also emphasizes the importance of adopting a branching strategy that aligns with the team’s workflow​ ([Tower Git Client](https://www.git-tower.com/blog/version-control-best-practices/)).
4. **Write Clear Commit Messages**: This improves collaboration and understanding among team members. Atlassian stresses the significance of clear and descriptive commit messages, as these facilitate better communication among team members​ ([Nulab](https://nulab.com/learn/software-development/version-control-best-practices/" \t "_blank)). A well-structured commit message should explain what was changed and why.

I selected these guidelines because they directly impact the quality of the code and the efficiency of the development process. By adhering to these practices, teams can foster a collaborative environment and maintain a high standard of code integrity.

Sources Used: <https://nulab.com/>

<https://www.git-tower.com/>

<https://www.atlassian.com/>