Restore Point Safety Guide

This guide explains how to safely use restore points in the Rudder\_Code project without accidentally overwriting them due to open editors or untracked changes.

# 1. Overview

The restore point system in Rudder\_Code allows you to return the repository to a known good state by resetting to a tagged commit. This is useful when testing, refactoring, or recovering from accidental changes.

# 2. Risks to Avoid

Even with restore points, you can unintentionally undo their effect if:

• Files are still open in your editor and auto-save is enabled — your editor may rewrite them immediately after a restore.

• You have untracked files (new files not yet staged) — these are not removed unless explicitly cleaned.

• You forget to push restore tags to GitHub — meaning they won’t be available if you clone elsewhere.

# 3. Best Practices

Follow these practices to ensure safe and reliable restores:

1. Close open files in your editor before restoring, or ensure your editor auto-reloads files when changed on disk.

2. Use the restore script to create restore points before making risky changes:

./scripts/make\_restore\_point.sh -m "Description of state" -p

3. When restoring, run:

git reset --hard <restore-tag>

4. Remove untracked files (if needed) with:

git clean -fd

⚠ WARNING: This will delete untracked files permanently.

5. Always push restore tags with `-p` to back them up remotely.

# 4. Verification After Restore

After a restore, verify the repository state:

• `git status` should show a clean working tree.

• `git show HEAD` should match the expected restore commit message and hash.

• Optionally, run your build/tests to confirm functionality.