# 🚀 Streamlined Git Workflow for Solo Development

## 📌 Step 1: One-Time Setup (Run Only Once)

In Git Bash, set up the `savepoint` alias:

git config --global alias.savepoint '!git add . && git commit -m "Auto-save" && git tag -a $(date +%Y-%m-%d\_%H-%M-%S) -m "Archived version" && git push && git push --tags'

✅ This creates a shortcut for saving your work.

## 📌 Step 2: Start a New Working Branch for Development

1. When you start working on a new version, create a working branch:

git checkout -b working-v0.4

2. Push it to GitHub (only needed for new branches):

git push -u origin working-v0.4

✅ Now, you're ready to work!

## 📌 Step 3: Save Your Work Anytime

Whenever you want to archive your work, simply run:

git savepoint

✅ This will:

- Stage all files (git add .)

- Commit with 'Auto-save'

- Tag the commit with the timestamp (e.g., 2024-02-22\_17-30-00)

- Push changes and tags to GitHub

## 📌 Step 4: List Your Archives (Checkpoints)

To see all your savepoints:

git tag

✅ This lists all previous savepoints.

## 📌 Step 5: Restore Any Previous Savepoint

If you ever want to go back to an earlier version:

git checkout tags/2024-02-22\_17-30-00

(Replace with the actual tag timestamp.)

🚀 Now your project is exactly as it was at that moment.

## 📌 Step 6: Promote a Version When You're Ready

When you're happy with the state of `working-v0.4` and want to release it as `v0.4`:

1. Tag it as an official version:

git tag -a v0.4 -m "Version 0.4 release"

git push origin v0.4

2. Create a new working branch for the next version:

git checkout -b working-v0.5

git push -u origin working-v0.5

✅ Now, `v0.4` is locked as a milestone, and `working-v0.5` is ready for new development.

## 📌 Summary of Your New Git Workflow

|  |  |
| --- | --- |
| Action | Command |
| One-time setup | git config --global alias.savepoint '...' |
| Start a new working branch | git checkout -b working-vX.X |
| Push new branch (only once) | git push -u origin working-vX.X |
| Save work anytime | git savepoint |
| List all savepoints | git tag |
| Restore a previous savepoint | git checkout tags/TIMESTAMP |
| Finalize a version (vX.X) | git tag -a vX.X -m "Version X.X release" && git push origin vX.X |
| Start the next working version | git checkout -b working-vX.X+1 && git push -u origin working-vX.X+1 |

🚀 Now, Git Works Like a Simple "Save" Button!

- No unnecessary Git commands.

- Always able to restore any past version.

- Automatic incremental versioning with `savepoint`.

- Clear separation between "working versions" and "release versions".