AGENTIA PLUGIN

Repo Doctor

A cleanup pass for a Copado Salesforce Git clone. It reads the repository on your computer, puts the worst findings first, and makes only the changes that leave every commit, tag, and the reflog alone.

No login

No Agentia credentials, no org, no API calls. The clone on disk is the whole input.

Safe by default

Recommended changes repack storage and refresh names. History rewrites are printed, and you run them yourself.

Twice a year

Built for the repository that has collected user stories, promotions, and metadata for a few seasons.

START HERE

Install it, then point it at the clone

Repo Doctor is an Agentia plugin. It adds one command, `agentia repo doctor`. Run it from the Git clone you want to look at, or pass `--path`.

```
agentia plugins install @copado/agentia-plugin-repo-doctor

# from the clone
agentia repo doctor

# environment branches that should never appear in the report
agentia repo doctor --keep dev --keep 'uat*'

# a pipeline or a script: recommended changes only, no questions
agentia repo doctor --apply --yes
```

Node.js 22.12 or newer, Git, and the Agentia CLI are the requirements. The plugin needs no Copado API key.

What a session does

1. If `git-sizer` is missing, it asks once, then downloads version 1.5.0 and checks the file against a pinned checksum. Decline, and the rest of the report still runs.
2. One findings table opens the report, worst first. Each row says what was found and what to do. Later sections add detail only for what this clone actually has.
3. In a terminal, a checklist follows. Recommended items start ticked. Optional items do not. Space toggles a line, then a “Will run” preview, then one confirmation that defaults to no.
4. After the changes, it compares `HEAD`, the refs, and the reflog with the values from before. A difference stops the run. Remote-tracking names may change when the fetch runs. Branches you chose to delete are expected to be gone.

Ctrl-C at any prompt ends with “Nothing was changed.” and exit code 130.

THE REPORT

Four things, and only what this clone has

Thresholds below are heuristics. Salesforce and Git do not enforce them. A quiet repository prints a short report.

Repository health

Size, loose objects, pack files, whether a commit graph exists, and how many files are tracked. These numbers decide the maintenance checklist. They are also the first place to look when `git status` or blame feels slow.

Branches

Local data, as fresh as your last fetch. “Merged” means the work is already on the default branch or on an environment branch. Environment branches are inferred from promotion merges, including Copado commits whose message is `Merging promotion/P… into uat`. Pass `--keep` for any environment it misses. The default branch, the branch you have checked out, and every `--keep` match stay out of the report.

GROUP	WHAT IT MEANS	WHAT TO DO
Merged features	`feature/` branches already contained in an environment or the default branch.	Safe to delete locally, after you look.
Merged promotions	Copado creates one `promotion/` branch per promotion. The work is already on the environment.	The names can go. The remote copy is left for you.
Unmerged promotions	Usually a failed or validation-only promotion.	Check that nothing is waiting on it, then delete.
Stale features	Unmerged `feature/` branches with no commit for 180 days. Change the window with `--stale-days`.	Ask the owner whether to finish the story or drop the branch.
Other names	Branches outside `feature/` and `promotion/`.	Environment names such as `dev` and `uat` belong here until you `--keep` them. Ask about the rest.
Remote copy deleted	The local branch is still here, and the remote branch is gone.	If it was merged, this copy can go too. Otherwise ask first.

Salesforce content

Read from the files at `HEAD`. Other Git repositories get no Salesforce section. A project counts as Salesforce when it has `sfdx-project.json` or a metadata tree.

FINDING	SHOWN FROM	WHY IT MATTERS
Reports and dashboards	25 files or more	Edited in the org all the time, so they flood history. Keep what code needs. Add the rest to `.forceignore`.
Large profiles	One file over 500 KiB, or 5 MiB in total	A large file overwritten on every deploy. Prefer a few small profiles and permission sets.
Object translations	1,000 files or more	One tiny file per translated field. Ignore the folder only if Git does not need it.
Content assets	10 MiB or more	Pictures and files. Each version stays in history.
Static resources	10 MiB in total, or one file over 2 MiB	Large binaries. Salesforce caps one static resource at 5 MB.
Flow versions	5 files named `Name-N.flow-meta.xml`	Every saved version is its own file. Delete old versions in the org and stop retrieving them.
Old `.object` format	Any legacy object file	One file per object is slow to merge. Convert to source format in small batches.
Local files in Git	Any tracked `.sfdx/`, `.sf/`, `node_modules/`, `.DS_Store`, or `*.log`	Tool leftovers. Add them to `.gitignore`. Older commits keep them either way.
No `.forceignore`	The file is not tracked	Deploys and retrieves include everything.

The same section draws size by metadata folder, and lists the largest tracked files that no finding already covers.

History

git-sizer scores the shape of history. A score of 10 or more is severe, and scores are allowed to go past 10. The report names the widest folder and the deepest path behind those scores, and lists the largest files in history once per file, with versions grouped. Maintenance does not remove a blob that is still in an old commit. That fix is printed, and it rewrites history.

WHAT IT CAN CHANGE

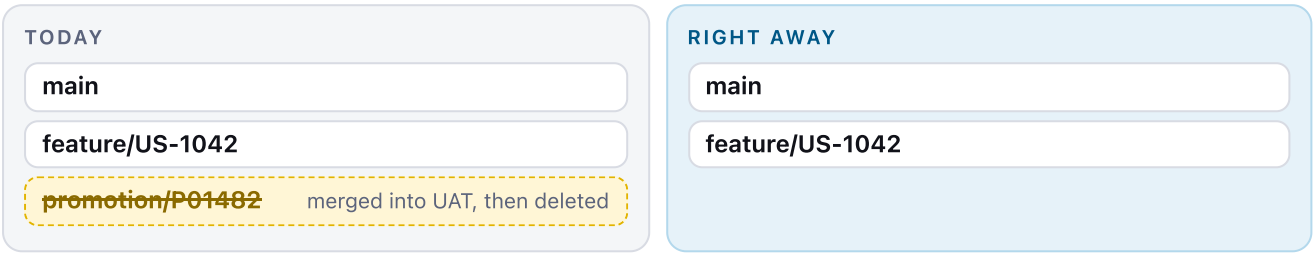
Eight changes. Commits stay. The remote stays.

These are the changes it can make on your computer. They help a large Salesforce project, or one with many user-story and promotion branches. Recommended items are the ones `--apply --yes` will run. Branch deletion is never in that set.

Stop listing promotion branches the remote already deleted

RECOMMENDED

Your computer still lists feature and promotion branches that were deleted on the remote after they merged.



EXAMPLE `promotion/P01482` was merged into UAT, then deleted on the remote. Your computer still lists it.

YOU GAIN One fetch asks the remote which names are gone and drops those names here.

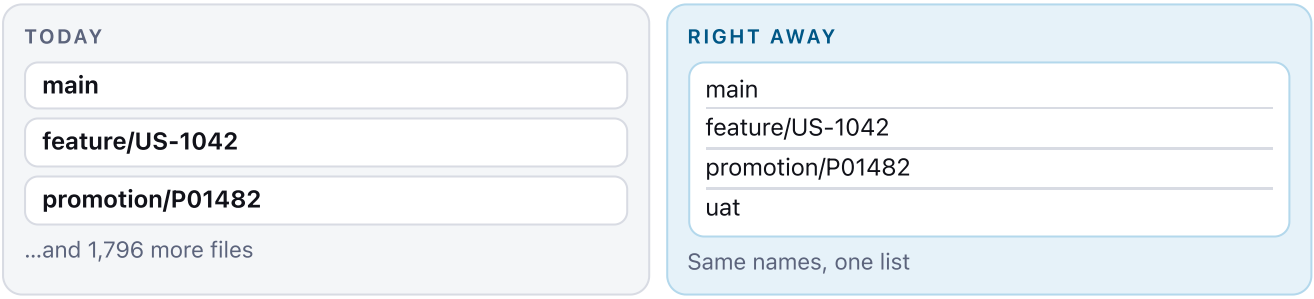
YOU KEEP Branches you are still working on. The remote. Nothing is deleted on the shared server.

`git fetch --all --prune --no-tags`

Open a long branch list faster

RECOMMENDED

Hundreds of feature and promotion branches, each name stored as its own tiny file.



EXAMPLE 1,800 branches, mostly `feature/US-…` and `promotion/P…`. Showing the list means opening 1,800 little files.

YOU GAIN The same names, gathered into one packed list, so the list opens faster.

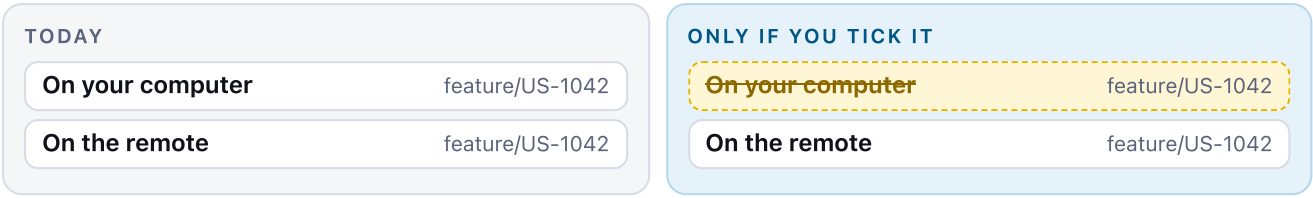
YOU KEEP Every branch name, still pointing at the same commit. You stay on the branch you are on.

`git pack-refs --all`

Remove a merged user-story branch from this computer

ONLY IF YOU TICK IT

A `feature/` or `promotion/` branch on your computer is fully merged, and you no longer need the name.



EXAMPLE `feature/US-1042` is already in the UAT environment branch. The name is clutter on this machine.

YOU GAIN A shorter branch list here. The output prints the commit id, so `git branch <name> <id>` puts the name back.

YOU KEEP The commits. The branch on the remote. A branch that is not fully merged is refused.

git branch -d -- <branch> · needs --delete-merged-branches with --apply --yes

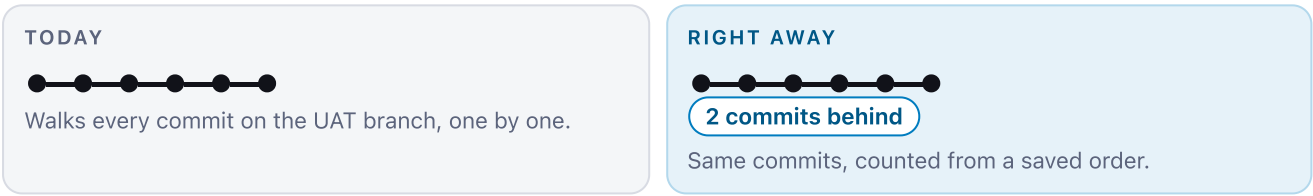
SPEED

Answers that used to walk every commit

See how far a sandbox is behind UAT, today

RECOMMENDED

The project is large, so “how far is my sandbox behind UAT?” has to walk every commit.



YOU GAIN History questions get faster the same day. Offered when the repository has 5,000 objects or more and no commit graph yet.

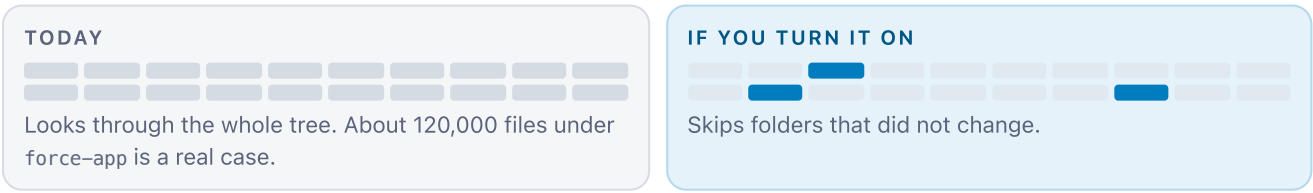
YOU KEEP The commits, in the same order, with the same content. This writes an index. It does not write a new commit.

git commit-graph write --reachable --changed-paths

Check what changed in a huge Salesforce project

STARTS OFF

The clone holds a full metadata tree: classes, profiles, reports, translations. Offered from 10,000 tracked files, and only when the setting is unset.



YOU GAIN A quicker answer to “what did I change?”. Git can skip unchanged folders.

YOU KEEP Your commits and your metadata. Some older tools cannot read the newer file list, which is why this stays off until you choose it. The setting is written only in this clone’s own config.

git config --local feature.manyFiles true

FILES ON DISK

The `.git` folder, tidied in place

Stop tracking a second checkout you already threw away

RECOMMENDED

You checked the repository out in a second folder, deleted that folder, and Git still mentions it.



EXAMPLE A hotfix folder next to the project, used to patch a sandbox, then deleted.

YOU GAIN The warnings about a missing folder stop.

YOU KEEP Any folder that still exists. No commits and no branches.

git worktree prune

Read history without opening thousands of loose files

RECOMMENDED

Years of metadata sit inside `.git` as thousands of separate files, so a simple question opens all of them. Offered from 500 loose objects, or when loose objects are already packed.



YOU GAIN Fewer files to open for the same history. On Git 2.34 or newer this also merges small packs when there are 10 or more, and writes the multi-pack index. Older Git only packs the loose objects.

YOU KEEP Commits you can check out. Unreachable objects stay. This repacks storage. It does not run `git gc`, and it does not delete history.

git repack -d --geometric=2 --write-midx

Find one file without opening every pack file

RECOMMENDED

History is already in several pack files, and Git has no index across them. Offered from four packs, when a repack is not already going to write that index.



EXAMPLE Reading one Apex class from history used to open pack 1, then 2, then 3.

YOU GAIN Git opens the right archive first.

YOU KEEP The pack files. Nothing is deleted and no commit is rewritten.

git multi-pack-index write

One fetch, even if several lines need it. Actions that talk to the remote share `git fetch --all --prune --no-tags`. Tags are left alone. The plugin does not push, and it does not run `git gc`.

PRINTED, NEVER RUN

The fixes that need a person

Read this before you copy a command. When git-sizer or the Salesforce scan finds something the checklist cannot fix — a huge file still in history, a folder that is too wide, reports that should leave the repository — Repo Doctor prints a suggestion and stops there. Steps marked as rewriting history replace old commits. Everyone who already has a copy has to clone again. Other steps only affect new files, or delete a branch on the remote. None of them run unless you run them. Commands keep `<placeholders>` instead of names from your repository.

STEP	COMMAND IT PRINTS	WHY
Delete a remote branch	`git push origin --delete <branch>`	Deletes it on the shared remote. Check with the owner first. `--list merged-feature` prints names, one per line, for that review.
Keep new big files out	`git lfs track "<pattern>"`	Git LFS stores the file beside the repo and keeps a small pointer in history.
Strip old large files	`git filter-repo --strip-blobs-bigger-than 10M`	Rewrites history. Anyone with a copy clones again.
Move existing big files	`git lfs migrate import --include="<pattern>"`	Rewrites the commits that contain them.
Erase reports from history	`git filter-repo --invert-paths --path <reports-dir>`	Rewrites history. Adding `.forceignore` only stops future retrieves.
Check out less	`git sparse-checkout set <dir>`	Limits the working tree to the folders you list. The remote is unchanged.
Clone without every version	`git clone --filter=blob:none <url>`	Fetches file contents on demand.
Split permission sets	`sf project convert source-behavior --behavior decomposePermissionSetBeta2 --dry-run`	Smaller files merge better. Profiles cannot be split. Commit the conversion on its own.
Flatten or split folders	`git mv …`	Deep paths hit Windows limits. A very wide folder checks out and diffs faster once it is split. One Salesforce metadata folder is normal; splitting the project into packages is what shrinks that.

REFERENCE

Flags, scripts, and the safety check

FLAG	WHAT IT DOES
`--path <dir>`	Clone to inspect. Defaults to the current directory.
`--keep <name>`	Branch name or `*` / `?` pattern left out of every branch finding. Repeat it. Use it for environment branches.
`--stale-days <n>`	A feature branch with no commits for this many days counts as stale. Default 180. Minimum 1.
`--apply`	Make changes. In a terminal you still choose. Without a terminal, pair it with `--yes`.
`--yes`	Skip the questions, and download git-sizer if it is missing. Required with `--apply` in scripts and with `--json`.
`--delete-merged-branches`	With `--apply --yes`, also delete local `feature/` and `promotion/` branches that are fully merged. The remote stays.
`--list <group>`	Print branch names only, one per line, without the remote prefix, then exit. Groups: `merged-feature`, `merged-promotion`, `promotion-unmerged`, `stale-feature`, `other-names`, `gone-local`. Repeat it. Cannot be combined with `--apply`.
`--json`	Machine-readable result on stdout. No checklist.
`--skip-git-sizer`	Skip history scores even when git-sizer is installed.

After it finishes

- Each deleted local branch is printed with its commit id. Restore it with `git branch <name> <id>`. A branch Git refuses to delete is skipped, and the run continues.
- `HEAD`, tags, other refs, and the reflog are compared with the snapshot from before. A surprise difference fails the command. Deleted branches, and remote-tracking names updated by the fetch, are the expected differences.
- Exit 0 means the run completed. Exit 1 means a step failed or the safety check found an unexpected change. Exit 2 means a bad flag, or a refusal to change the repository without `--yes`. Ctrl-C is exit 130.

What the process is allowed to do

It runs `git` and `git-sizer` with fixed argument lists. It builds no shell command, and it takes no command from the repository. A branch is deleted only when the name is an ordinary Git branch name. The only download is git-sizer 1.5.0, after you confirm or pass `--yes`, checked against a pinned checksum, and stored under `~/.agentia/plugins/repo-doctor/`. That download talks only to GitHub's release hosts.

Settings go into this repository's own config, and only when they are unset. The plugin leaves tracked reports and tooling files in place, and names the `.forceignore` or `.gitignore` line to add. Removing those files from history stays a manual step.

Repo Doctor 0.1.0 · @copado/agentia-plugin-repo-doctor · The thresholds in this guide match that version. A later release can move a number; the command's own report is the one to trust for the clone in front of you.