✅ Solution la plus simple : réinitialiser l'historique complet

Étape par étape :

🔥1️⃣Sauvegarde ton travail (toujours)

Avant de tout casser :

bash
CopierModifier

```
# Crée une branche de secours
git checkout -b sauvegarde-avant-reset
```

🔥2️⃣Crée une nouvelle branche vide

bash
CopierModifier

```
# Va sur la branche principale
git checkout main

# Crée une branche "nouvelle-base"
git checkout --orphan nouvelle-base
```

👉 `--orphan` = branche sans historique.

🔥3️⃣Ajoute tous les fichiers actuels

bash
CopierModifier

```
# Supprime l'index des anciens commits
git rm -rf .

# Ajoute les fichiers actuels
git add .

# Nouveau commit tout propre
git commit -m "Réinitialisation de l'historique"
```

🔥4️⃣ Remplace la branche principale par la nouvelle

bash
CopierModifier

```
git branch -M main  # S'assurer qu'on est sur "main"
```

🔥5️⃣ Écrase le dépôt distant

⚠️ **Attention**, ça force tout le monde à resynchroniser !

bash
CopierModifier

```
git push --force origin main
```

🎉 Résultat

✅ **Un seul commit**, tout l'historique passé est supprimé localement **et** à distance.

⚡ Alternative : supprimer uniquement les N premiers commits

Si tu veux **conserver seulement les 100 derniers commits** par exemple :

bash
CopierModifier

```
git checkout main

# Trouver le hash du dernier commit que tu veux SUPPRIMER
git log  # Trouve le hash à partir duquel tu veux repartir

# Supposons que ce soit abcdef123456
# Crée une nouvelle branche à partir de ce commit
git checkout --detach abcdef123456

# Crée une branche propre à partir de là
git checkout -b nouvelle-base
```

```
# Force le push
git push --force origin nouvelle-base:main
```

🗝️ Conseils

✅ **Préviens toujours ton équipe** avant un `--force`.

✅ Sauvegarde la branche initiale en local ou sur un autre dépôt.

✅ Si c'est trop risqué ➜ crée un **nouveau repo** et pousse-y ton projet tout neuf.