Git Workflow Guide for Sven

Repository Setup

You have 3 key repositories:

Remote	URL	Purpose
`prod`	chickenloop3845-commits/chickenloop	**Production** - the main source of truth
`sven` (your fork)	SvenCLDev/chickenloop	Your development fork
`joco`	jhegedus42/chickenloop	Joco's fork

Daily Workflow

1. Keep Your Fork Synced with Prod

Before starting any new work, sync your `main` branch with production:

```
# Fetch latest from prod
git fetch prod

# Update your local main
git checkout main
git merge prod/main

# Push to your fork
git push origin main
```

---

2. Create Feature Branches

Always work on feature branches, never directly on `main`:

```
# Create a new branch from main
git checkout main
git checkout -b pr/my-new-feature

# Do your work, commit often
git add .
git commit -m "Add feature X"

# Push to your fork
git push origin pr/my-new-feature
```

Branch naming convention: - `pr/feature-name` - for features destined for a PR - `fix/bug-description` - for bug fixes

---

3. Creating Pull Requests

When your feature is ready:

1. **Sync with prod first** (to avoid conflicts):

```
git fetch prod
git rebase prod/main
```

2. **Push your branch**:

   ```
   git push origin pr/my-new-feature --force-with-lease
   ```

3. **Create PR on GitHub**:

   - Go to `chickenloop3845-commits/chickenloop`
   - GitHub will show "Compare & pull request" for your branch
   - Or go to your fork and click "Contribute" → "Open pull request"

Warning: What to Avoid

X Don't	Check Do Instead
Commit directly to `main`	Use feature branches
Create merge commits when syncing	Use `git rebase prod/main`
Push without syncing first	Always `git fetch prod` before pushing
Keep unfinished work on `main`	Use branches for experiments

Common Commands Cheat Sheet

```
# Check current status
git status

# See which branch you're on
git branch

# See all branches (including remotes)
git branch -a

# Switch to a branch
git checkout branch-name

# Create and switch to a new branch
git checkout -b new-branch-name

# See commit history
git log --oneline -10

# Compare branches
git log --oneline main..pr/my-feature

# Discard local changes to a file
git checkout -- filename

# Undo last commit (keep changes)
git reset --soft HEAD~1
```

Example: Complete Feature Workflow

```
# 1. Start fresh
git checkout main
git fetch prod
git merge prod/main
git push origin main

# 2. Create feature branch
git checkout -b pr/add-email-notifications

# 3. Work and commit
# ... make changes ...
git add .
git commit -m "Add email notification service"

# ... more changes ...
git add .
git commit -m "Add email templates"

# 4. Before creating PR, sync with prod
git fetch prod
git rebase prod/main

# 5. Push to your fork
git push origin pr/add-email-notifications

# 6. Create PR on GitHub (web UI)

# 7. After PR is merged, clean up
git checkout main
git pull prod main
git push origin main
git branch -D pr/add-email-notifications
```

If You Get Stuck

- **Merge conflicts**: Ask for help before force-pushing
- **Detached HEAD**: Run `git checkout main` to get back to a branch
- **Wrong branch**: Stash changes with `git stash`, switch branches, then `git stash pop`

Happy coding!