

Git Handbook

Remote to Local Setup & Rebasing Workflow

Setting Up Your Environment

```
# Creating a workspace to pull the changes
mkdir project-workspace
cd project-workspace

# Clone frontend repository
git clone <repository-url>
cd frontend
git checkout -b <your-branch-name>
##### Start working in your branch ###

cd ..

# Clone backend repository
git clone <repository-url>
cd backend
git checkout -b <your-branch-name>
##### Start working in your branch ###
```

Daily Development Workflow

Pulling Updates from Remote

```
# For frontend repository
cd frontend
git fetch origin # Update remote references without merging
git pull origin main # Pull and merge changes from main branch

cd ..
```

```
# For backend repository
cd backend
git fetch origin # Update remote references without merging
git pull origin main # Pull and merge changes from main branch
```

Using Rebase while pushing to remote

Before Committing Changes

```
# Stage your changes
git add . # Add all changes or specify files
git status # Review what will be committed (not required but good practice)

# Create a local commit
git commit -m "Descriptive message about your changes"
```

Rebasing Process

```
# Update your local main with remote changes
git checkout main
git pull origin main

# Switch back to your feature branch
git checkout <your-branch-name>

# Rebase your branch on top of main
git rebase main

# If conflicts occur:
# 1. Fix conflicts in your editor
# 2. Add resolved files
git add <resolved-files>
# 3. Continue rebase
```

```
git rebase --continue  
# If you need to abort: git rebase --abort
```

Pushing Changes After Rebasing

```
# Since rebasing rewrites history, force push is required  
git push origin feature-branch --force-with-lease
```

References:

- <https://www.freecodecamp.org/news/how-to-use-git-rebase/>