1. Grundlagen: Was ist Git?

- **Repository (Repo):** Projektverzeichnis, das von Git verwaltet wird.
- **Commit:** Ein Snapshot des Codes (inkl. Commit-Nachricht)
- **Branch:** Eine unabhängige Entwicklungslinie.
- **Remote:** Externes Repository (z. B. auf GitHub).
- **Merge:** Zusammenführen von Branches.
- **Pull Request (PR):** Anfrage zum Merge (GitHub UI).

2. Projekt starten

```
# Repository klonen
git clone https://github.com/benutzername/projekt.git
```

3. Feature-Branch starten

```
# Neuen Branch erstellen und wechseln
git checkout -b feature/kurze-beschreibung
```

Best Practice – Namenskonventionen:

- `feature/login-form`
- `bugfix/falscher-pfad`
- `refactor/form-validation`

4. Änderungen tracken & committen

```
# Status prüfen (Welche Dateien wurden geändert?)
git status

# Änderungen zur Staging-Area hinzufügen
git add DateiA.cs DateiB.cs
# oder alle Änderungen
git add .

# Commit mit Nachricht
git commit -m "Neue Login-Validierung implementiert"
```

5. Branch pushen & PR erstellen

```
# Lokalen Branch auf Remote hochladen
git push -u origin feature/kurze-beschreibung
```

Danach GitHub öffnen und einen **Pull Request** gegen den Ziel-Branch erstellen, z. B. `develop` .

6. Remote-Änderungen holen (z. B. `develop` aktualisieren)

```
# Ziel-Branch holen und lokal updaten
git checkout develop
git pull origin develop
```

7. Feature-Branch aktuell halten (optional, z. B. vor Merge)

```
git checkout feature/kurze-beschreibung
git merge develop
# Bei Konflikten: manuell lösen, dann
git add .
git commit
```

8. Merge abgeschlossen (über GitHub)

Nach dem Merge (PR) kannst du den Branch lokal und remote löschen:

```
# Lokal löschen
git branch -d feature/kurze-beschreibung
# Remote löschen
git push origin --delete feature/kurze-beschreibung
```

Optional: Nützliche Befehle im Alltag

```
# Zeige Historie
git log --oneline --graph --all

# Dateien vergleichen
git diff            # unstaged Änderungen
git diff --staged   # staged Änderungen

# Änderungen rückgängig machen
git restore datei       # unstaged Änderung verwerfen
git reset HEAD datei    # aus staging entfernen
```

Workflow-Übersicht

1. `git pull` (aktualisieren)
2. `git checkout -b feature/neues-thema`
3. Code schreiben, `git add`, `git commit`
4. `git push -u origin feature/neues-thema`
5. PR in GitHub → Review → Merge → Branch löschen