Manual Avançado de Git - EasyPCM

1. Fundamentos Essenciais

Git é um sistema de controle de versão distribuído baseado em snapshots.

Cada commit representa um estado completo do projeto.

Git armazena diferenças (deltas), não cópias completas como arquivos RAR.

2. Fluxo Mental do Git

Modified → Staged → Committed

Você altera arquivos, adiciona ao stage com 'git add' e cria um commit com 'git commit'.

3. Comandos Fundamentais

Ver status do projeto:

```
git status
```

Ver histórico resumido:

```
git log --oneline --graph --decorate
```

Ver diferenças antes de commitar:

```
git diff
```

Adicionar arquivos:

```
git add .
```

Criar commit:

```
git commit -m "Descrição clara da alteração"
```

Enviar para repositório remoto:

```
git push
```

4. Navegação no Histórico

Ver histórico de um arquivo específico:

```
git log -- app.py
```

Ver alterações de um commit específico:

```
git show <hash>
```

5. Voltar no Tempo

Checkout temporário para um commit:

```
git checkout <hash>
```

Reset completo (CUIDADO):

```
git reset --hard <hash>
```

Reverter commit com segurança:

```
git revert <hash>
```

6. Restaurar Arquivo Específico

Restaurar arquivo para versão antiga:

```
git restore --source <hash> -- app.py
```

7. Branches Profissionais

Criar nova branch:

```
git checkout -b dev
```

Mesclar branch:

```
git merge dev
```

8. Boas Práticas

- Commits pequenos e lógicos.
- Mensagens claras e objetivas.
- Commitar frequentemente.

- Nunca versionar arquivos temporários (__pycache__, .env).

- Usar branches para novas funcionalidades.

9. Comparação Git vs Backup Tradicional

RAR/Backup copia tudo a cada versão.

Git salva apenas alterações.

Git permite restaurar arquivo individual.

Git mantém histórico auditável completo.