Versionamento com GIT

Nobody / github.com/z1nl0x

FUTURA

Overview

- Repositories / Repositórios
- Branching / Ramificações
- Committing / Comitando
- Merging / Mergeando

Repositories / Repositórios

- São diretórios onde residem o projeto que está sendo versionado pela equipe que o desenvolve.
- Podem ser clonados se os mesmos forem públicos.
- Normalmente é mantido por um grupo de desenvolvedores responsáveis pelo projeto.

Branching / Ramificações

- Como o próprio nome já diz, é uma ramificação da raiz do projeto.
- Desenvolvedores iniciam uma nova ramificação quando vão desenvolver uma nova estrutura dentro do sistema.
- É imprescindível que exista para organização e limpeza do código do projeto.

Branching / Ramificações

No nosso projeto utilizamos uma padronização de nomenclatura para a criação dessas branches/ramificações.

feature/FP-XXXX
bugfix/FP-XXXX
hotfix/FP-XXXX
release/FP-XXXX

Committing / Comitando

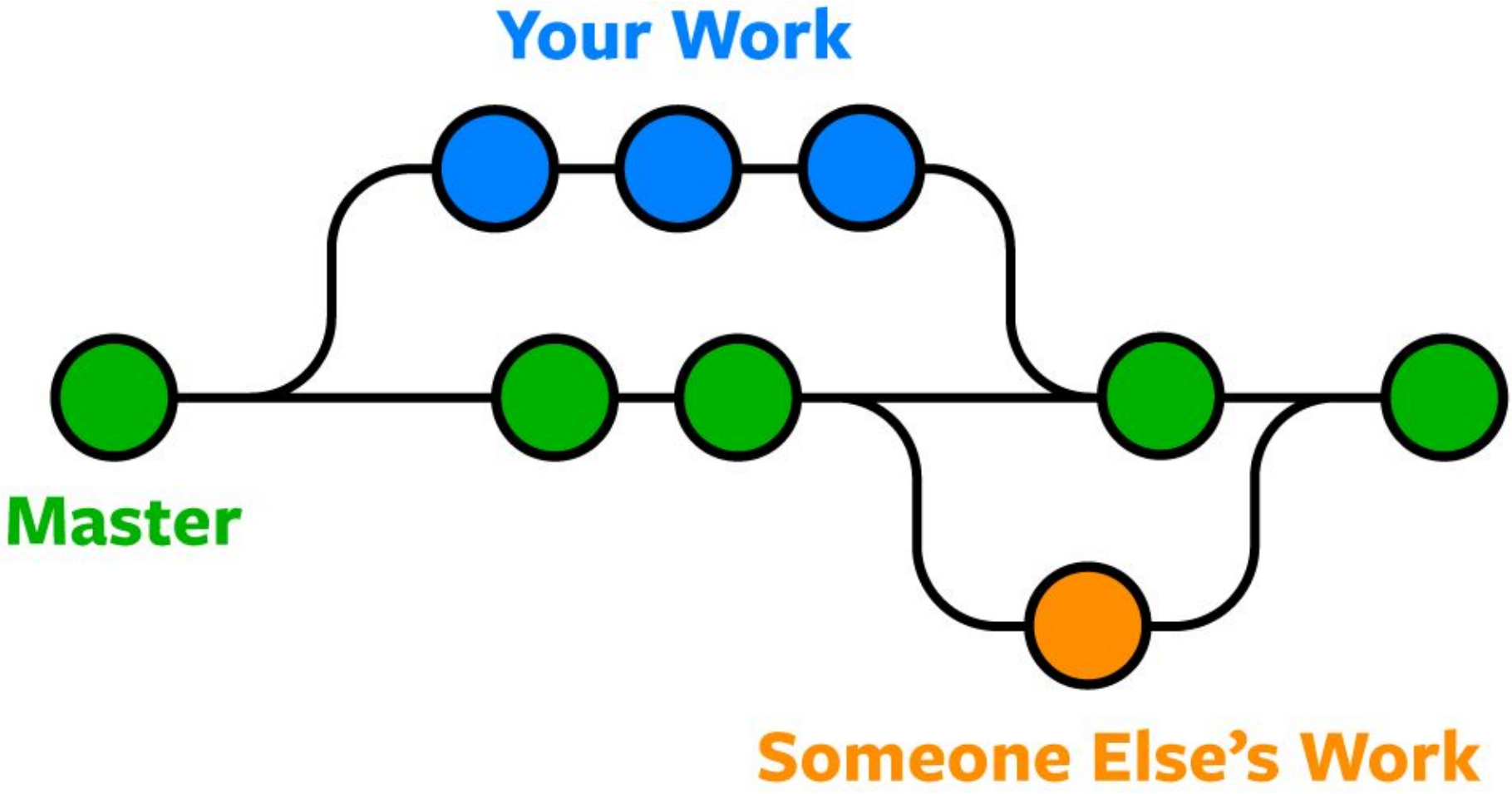
- Fazendo uma analogia, comitar é como tirar uma foto de como estava o seu código e registrar isso em uma lista de outros registros.
- É importante sempre comitar e deixar uma mensagem clara e objetiva do que foi realizado nesse commit. Para que desenvolvedores que não fizeram parte do mesmo, possam vê-lo, analisá-lo e entender o que ocorre neste registro.

Merging / Mergeando

- Fazendo outra analogia, mergear é como se você pegasse todas essas fotos (commits) e montasse um álbum de fotos com todos essas novas fotos e as fotos já existentes.
- O merge ele junta os commits vindo da ramificação que está sendo utilizada para desenvolver uma estrutura do sistema, e assim que finalizado o mesmo é agrupado na ramificação principal do projeto.

Merging / Mergeando

- Fazendo outra analogia, mergear é como se você pegasse todas essas fotos (commits) e montasse um álbum de fotos com todos essas novas fotos e as fotos já existentes.
- O merge ele junta os commits vindo da ramificação que está sendo utilizada para desenvolver uma estrutura do sistema, e assim que finalizado o mesmo é agrupado na ramificação principal do projeto.


Your Work
Master
Someone Else's Work

Workflow do Versionamento Futura

Raiz do projeto para os desenvolvedores da equipe.

Código passa pela revisão de um dev com mais experiência no projeto.

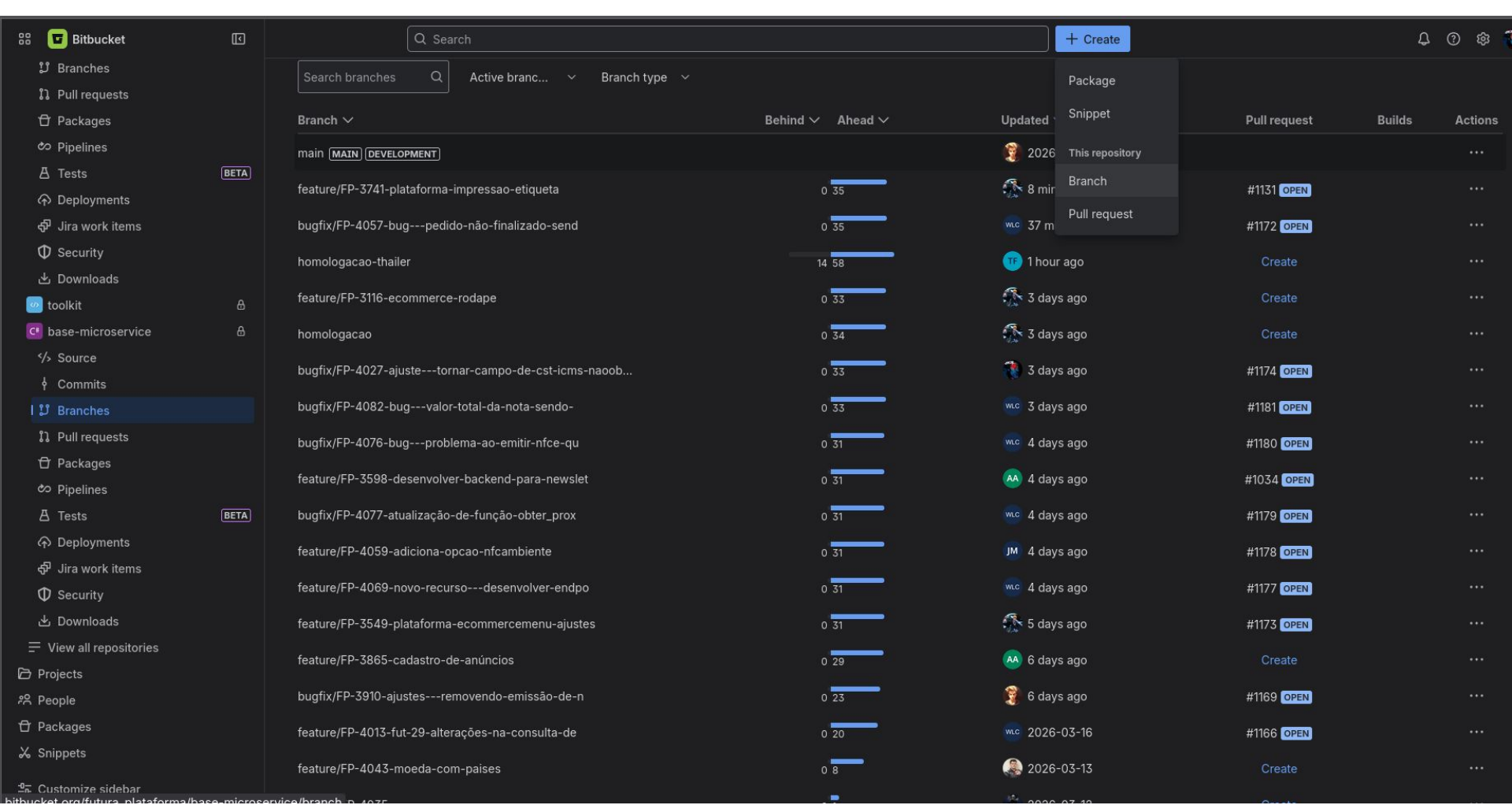
Se os testes passaram e tudo está OK. O Código é mergeado a raiz do projeto: homologacao

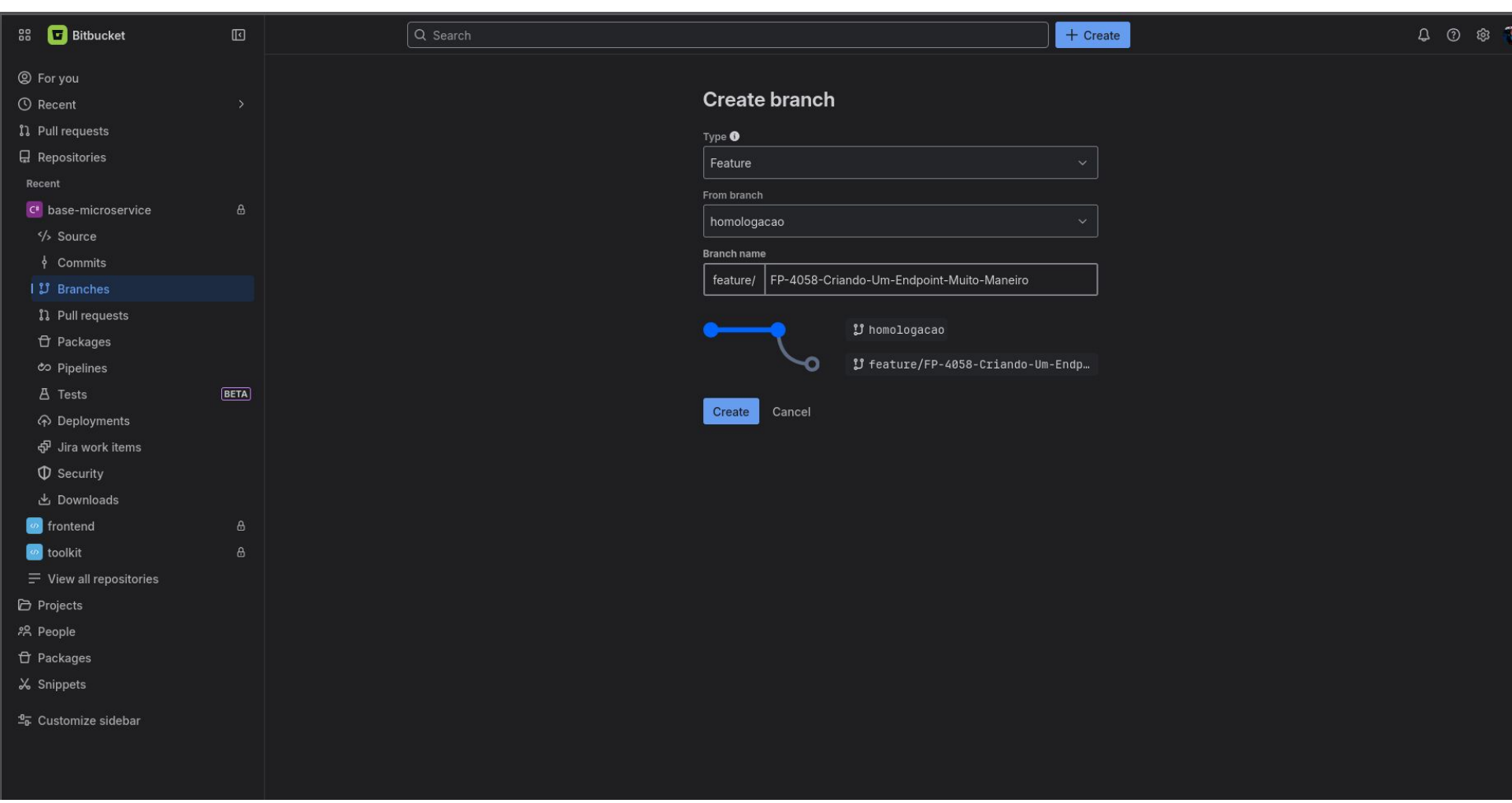
Nova branch para desenvolver uma nova tela/endpoint.

O código é testado pela equipe de testes.

Criando a branch no Bitbucket

- Criando a branch direto no Bitbucket evita alguns bugs, então segue os prints para criá-la sem maiores problemas.


Bitbucket
Search
Create
Branches
Pull requests
Packages
Pipelines
Tests
BETA
Deployments
Jira work items
Security
Downloads
toolkit
base-microservice
Source
Commits
Branches
Pull requests
Packages
Pipelines
Tests
BETA
Deployments
Jira work items
Security
Downloads
View all repositories
Projects
People
Packages
Snippets
Customize sidebar
Search branches
Active branc...
Branch type
Package
Snippet
This repository
Branch
Pull request
Branch
Behind
Ahead
Updated
Pull request
Builds
Actions
main MAIN DEVELOPMENT
2026
feature/FP-3741-plataforma-impressao-etiqueta
0 35
8 mi
#1131 OPEN
bugfix/FP-4057-bug---pedido-não-finalizado-send
0 35
37 m
#1172 OPEN
homologacao-thailer
14 58
1 hour ago
Create
feature/FP-3116-ecommerce-rodape
0 33
3 days ago
Create
homologacao
0 34
3 days ago
Create
bugfix/FP-4027-ajuste---tornar-campo-de-cst-icms-naoob...
0 33
3 days ago
#1174 OPEN
bugfix/FP-4082-bug---valor-total-da-nota-sendo-
0 33
3 days ago
#1181 OPEN
bugfix/FP-4076-bug---problema-ao-emitir-nfce-qu
0 31
4 days ago
#1180 OPEN
feature/FP-3598-desenvolver-backend-para-newslet
0 31
4 days ago
#1034 OPEN
bugfix/FP-4077-atualização-de-função-obter_prox
0 31
4 days ago
#1179 OPEN
feature/FP-4059-adiciona-opcao-nfcambiente
0 31
4 days ago
#1178 OPEN
feature/FP-4069-novo-recurso---desenvolver-endpo
0 31
4 days ago
#1177 OPEN
feature/FP-3549-plataforma-ecommercemenu-ajustes
0 31
5 days ago
#1173 OPEN
feature/FP-3865-cadastro-de-anúncios
0 29
6 days ago
Create
bugfix/FP-3910-ajustes---removendo-emissão-de-n
0 23
6 days ago
#1169 OPEN
feature/FP-4013-fut-29-alterações-na-consulta-de
0 20
2026-03-16
#1166 OPEN
feature/FP-4043-moeda-com-paises
0 8
2026-03-13
Create
bitbucket.org/futura_plataforma/base-microservice/branch


Bitbucket
Search
Create
For you
Recent
Pull requests
Repositories
Recent
base-microservice
Source
Commits
Branches
Pull requests
Packages
Pipelines
Tests
BETA
Deployments
Jira work items
Security
Downloads
frontend
toolkit
View all repositories
Projects
People
Packages
Snippets
Customize sidebar
Create branch
Type
Feature
From branch
homologacao
Branch name
feature/
FP-4058-Criando-Um-Endpoint-Muito-Maneiro
homologacao
feature/FP-4058-Criando-Um-Endp…
Create
Cancel

Comandos mais utilizados!

git fetch
Puxa todas as novas branchs criadas para seu repositório local.

git checkout feature/FP-4058-Criando-Um-Endpoint-Muito-Maneiro
Entra na branch para iniciar o desenvolvimento da feature.

git status
Mostra o status da sua branch, se ela está atualizada até a ultima versão comparada com a branch remota

Comandos mais utilizados!

git stash

Guarda temporariamente as modificações realizadas na branch atual para poder navegar pra outra branch ou realizar qualquer outra operação.

git stash pop

Devolve as alterações realizadas na branch novamente quando necessário.

git rebase origin/homologacao

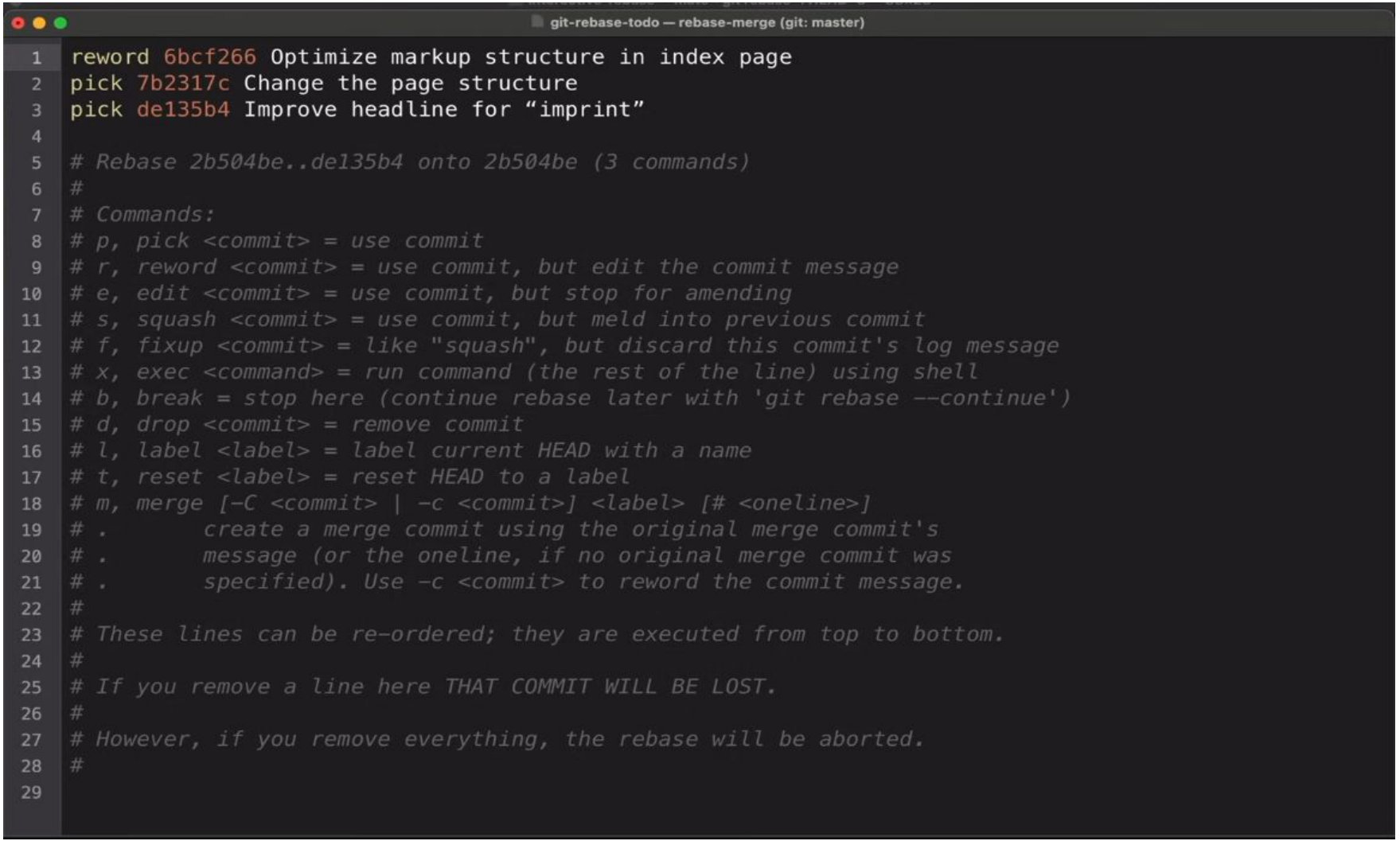
Reescreve o histórico movendo seus commits para o topo dos commits de outra branch. Em vez de criar um merge commit, ele "replanta" os seus commits como se tivessem sido criados a partir do estado mais recente da outra branch.

Comandos mais utilizados!

git branch bkp-feature/FP-XXXX
Cria uma branch de backup, se eventualmente der um problema é possível utilizá-la para outra cópia e iniciar o processo que estava realizando anteriormente.

git rebase -i $(git merge-base HEAD origin/homologacao)
Realiza o rebase interativo unificando os commits antes de abrir a PR.

git commit –amend –date “$(date)” -m “FP-XXX mensagem”
Unifica todos os commits em apenas um só e define a mensagem que este commit terá.


git-rebase-todo — rebase-merge (git: master)
reword 6bcf266 Optimize markup structure in index page
pick 7b2317c Change the page structure
pick de135b4 Improve headline for “imprint”
Rebase 2b504be..de135b4 onto 2b504be (3 commands)
Commands:
p, pick <commit> = use commit
r, reword <commit> = use commit, but edit the commit message
e, edit <commit> = use commit, but stop for amending
s, squash <commit> = use commit, but meld into previous commit
f, fixup <commit> = like "squash", but discard this commit's log message
x, exec <command> = run command (the rest of the line) using shell
b, break = stop here (continue rebase later with 'git rebase --continue')
d, drop <commit> = remove commit
l, label <label> = label current HEAD with a name
t, reset <label> = reset HEAD to a label
m, merge [-C <commit> | -c <commit>] <label> [# <oneline>]
. create a merge commit using the original merge commit's
. message (or the oneline, if no original merge commit was
. specified). Use -c <commit> to reword the commit message.
These lines can be re-ordered; they are executed from top to bottom.
If you remove a line here THAT COMMIT WILL BE LOST.
However, if you remove everything, the rebase will be aborted.

Comandos mais utilizados!

git merge homologacao

Na branch onde você está, para mergea-lá a branch de homologacao, só executar o comando acima.

OK, mas agora como isso se aplica no nosso Workflow?

- Todos os dias antes de iniciar suas atividades, vá até os repositórios frontend e base-microservices do nosso projeto e executem, git pull. Lembrando que deve ser sempre na branch de homologacao a se fazer isso.

OK, mas agora como isso se aplica no nosso Workflow?

- Todos os repositórios foram atualizados anteriormente? Ótimo, agora no repositório de base-microservices execute o script ./ExecuteMigrations.sh para executar todas as migrations disponíveis no projeto, mantendo seu backend atualizado.

OK, mas agora como isso se aplica no nosso Workflow?

- Crie a branch da sua tarefa como demonstrado nos slides anteriores no Bitbucket. Feito isso, abra seu terminal no repositório do frontend ou do base-microservices, execute um git fetch novamente, e após isso um git checkout feature/FP-XXX (Nome da sua branch) para iniciar suas tarefas.

OK, mas agora como isso se aplica no nosso Workflow?

- Sempre realize commits nas suas tarefas para você mesmo se organizar durante o desenvolvimento, eu pessoalmente só faço muitos commits quando a tarefa é muito extensa. Para commitar, utilize primeiro:

  git add .

  git commit -m “FP-XXX Adicionar input de Datepicker”.

OK, mas agora como isso se aplica no nosso Workflow?

- Finalizei o desenvolvimento da feature? Ótimo, agora existem alguns procedimentos a serem feitos.

OK, mas agora como isso se aplica no nosso Workflow?

PORÉM, DÊ PREFERÊNCIA AO SCRIPT QUE AUTOMATIZA TODOS ESSES COMANDOS PARA QUE NÃO OCORRA NENHUM ERRO.

`pre-pr.sh`

Padroniza o processo de *rebase interativo* antes da abertura de Pull Requests.

Fluxo:

1. Garante que está em um projeto válido (`frontend` ou `base-microservice`);
2. Impede execução em `homologacao` , `main` ou `master` ;
3. Gera um log de commits (`rebases/<data>_<branch>.log`);
4. Executa `git rebase -i` a partir do merge-base de `origin/homologacao` ;
5. Permite alterar a mensagem final do commit (`git commit --amend`);
6. Finaliza o rebase contra `origin/homologacao` .

Execução:

```
../toolkit/pre-pr.sh
```

> ⚠️ Execute **sempre** dentro da pasta do projeto (Ex: `frontend` ou `base-microservice`).

Acesse o toolkit para compreender melhor as ferramentas:
https://bitbucket.org/futura_plataforma/toolkit/src/main/README.md

That’s It.

Dúvidas?
Contribuições?