

Trabalhando com Branches no Github – Manual 2022.2

O que é uma branch?

Uma “branch” um galho, uma bifurcação de um projeto em algum momento no tempo do seu desenvolvimento.

Trabalhar com **branches** é uma maneira de trabalhar simultaneamente com vários desenvolvedores no mesmo projeto. Em vez de todos fazerem alterações diretamente na versão principal (**main**, antes chamada de master), cada alteração é feita em uma **branch**, um galho, uma versão paralela da versão que está em **main**.

Vantagens de se usar branches:

1. Se esse galho acabar “quebrando” alguma parte do projeto, é mais fácil de jogar fora e não é preciso reverter nenhum commit.
2. Se você estiver no arquivo A e seu colega no arquivo B, o Git consegue juntar as versões ao “comitarem”, sem que você precise esperar que o desenvolvedor do arquivo B faça o “commit” do que está fazendo.

Desvantagens de se usar branches:

1. É necessário ter um bom **mapeamento** de o que está sendo feito, onde e por quem, para que pedaços de código não acabem no “limbo” e precisando ser reescritos porque o desenvolvedor responsável não sabe onde deixou...
2. **Conflitos**. Conflito é o que acontece quando mais de um desenvolvedor fez alterações em uma mesma parte do projeto e, ao juntarem seus pedaços, o Git identifica que há inconsistências e não faz a junção, ou “merge”, das partes automaticamente. Quando isso acontece, um desenvolvedor deve manualmente apontar qual código manter e qual eliminar.



Como implementar branches no meu projeto?

1. Primeiro você deve criar uma branch ou selecionar uma já criada e adicionar seu código.

```
# 1. Acesse o seu repositório pelo GitBash ou pelo terminal do VSCode ANTES  
DE CODAR O QUE VOCE VAI DESENVOLVER.
```

*É possível criar uma branch depois que você já adicionou alterações,
mas criar a branch antes é mais fácil.*

```
# 2. Execute o comando abaixo para CRIAR uma branch:
```

Caso queira ir a uma branch já criada, não digite -b

```
git checkout -b nome-da-branch
```

```
----- 3. CODA CODA CODA -----
```

Adicione ao código suas alterações, melhorias, correções...

```
----- TERMINOU? -----
```

```
# 4. Execute os comandos:
```

```
git add .
```

```
git commit -m "mensagem"
```

```
git push -u origin nome-da-branch
```

```
----- FIM -----
```

```
# 5. Acesse o repositório remoto (Gitub) e siga os próximos passos, para  
criar a Pull Request (PR)
```

Ao final destes passos, o seu código desenvolvido localmente já está presente e disponível no repositório remoto, mas não está ainda adicionado ao resto do projeto. Vamos então criar uma Pull Request nos próximos passos para acrescentar ao projeto original o seu novo código, desenvolvido nesta nova branch apartada.

2. Fazendo Pull Request (PR) no Github.

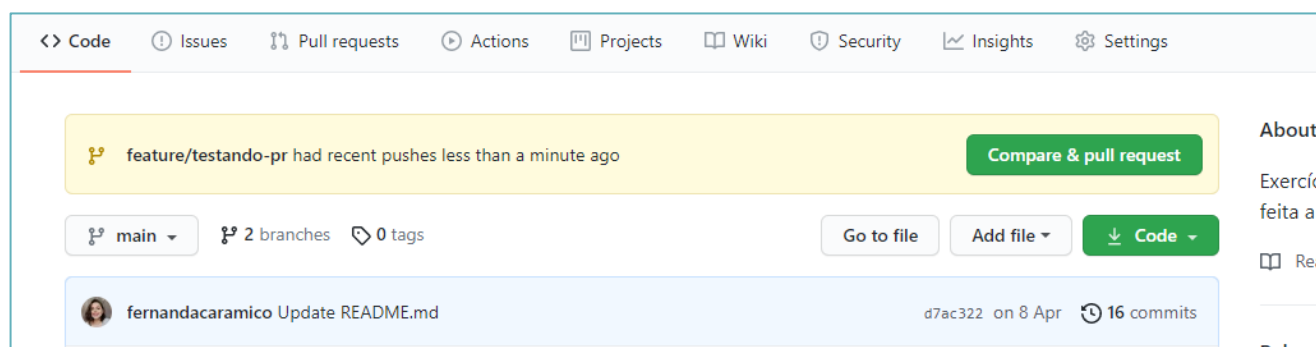
Pull Request é uma requisição para que uma alteração feita em uma branch seja adicionada, ou “mergeada” (merged), a outra branch.

É comum você ouvir nas squads de desenvolvedores a frase “vou fazer PR pra dev”. Nesta frase o desenvolvedor está querendo dizer que abrirá uma requisição para adicionar seu código ao código já presente na branch “dev”. Outra frase comum é “abre PR de dev pra main”. Aqui, o desenvolvedor está se referindo a adicionar todo o código presente em “dev” para a branch “main”.

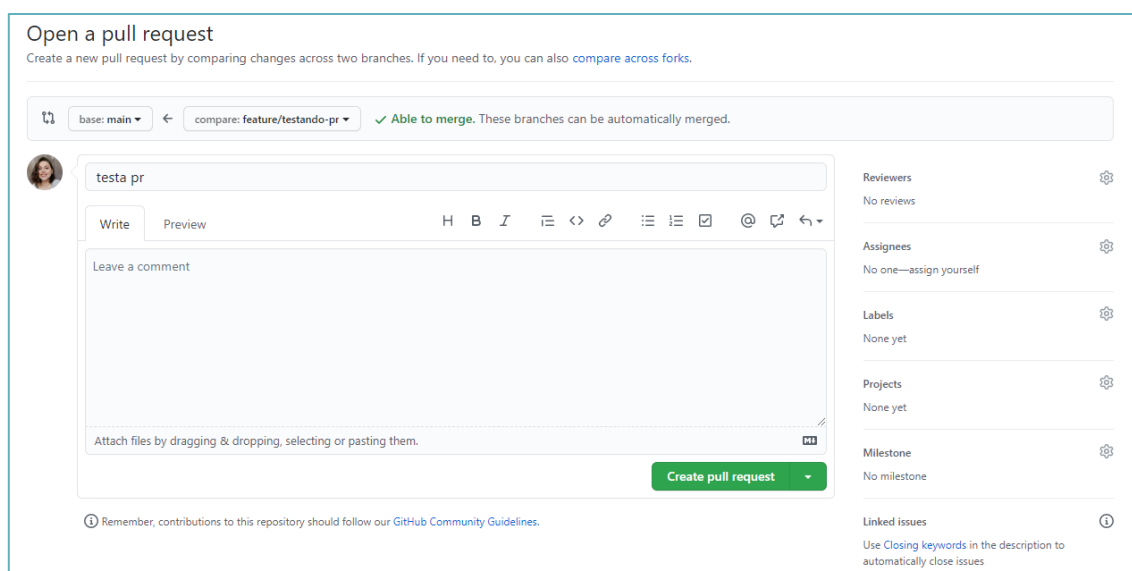
Os “merges” efetuados de uma branch para a outra sempre seguirão as políticas definidas pela sua empresa e é comum que passem pela aprovação de um Tech Lead antes das alterações serem realmente efetuadas.

Acesse o repositório do GitHub. Você verá algo como abaixo:

O nome escolhido para esta branch foi “feature/testando-pr”, porque estávamos adicionando uma nova característica, funcionalidade, ou feature.



Clique no botão **Compare & pull request** para comparar as suas alterações com as alterações na branch contendo a versão original do seu código. A tela será como abaixo:



Como fizemos o comando de criação da branch (`git checkout -b nome-da-branch`) quando estávamos na **main**, automaticamente o GitHub propõe que façamos “merge” da nova branch para a branch main. Podemos ver isso abaixo:

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).

base: main ← compare: feature/testando-pr ✓ Able to merge. These branches can be automatically merged.

testa pr

Write Preview

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

Create pull request

Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

Reviewers
No reviews

Assignees
No one—assign yourself

Labels
None yet

Projects
None yet

Milestone
No milestone

Linked issues
Use [Closing keywords](#) in the description to automatically close issues

O texto **✓ Able to merge.** mostra que as alterações que você está tentando adicionar à branch main podem ser adicionadas pois não há conflitos!

Para abrir então sua PR:

1. Adicione um título para sua alteração;
2. Deixe um comentário (opcional);
3. Adicione revisadores (opcional mas pode ser obrigatório dependendo da política de segurança da sua empresa);
4. Então crie sua PR clicando em “Create pull request”.

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).

base: main ← compare: feature/testando-pr ✓ Able to merge. These branches can be automatically merged.

testa pr

Write Preview

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

Create pull request

Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

Reviewers
No reviews

Assignees
No one—assign yourself

Labels
None yet

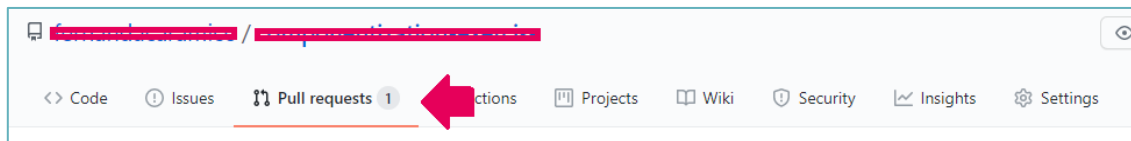
Projects
None yet

Milestone
No milestone

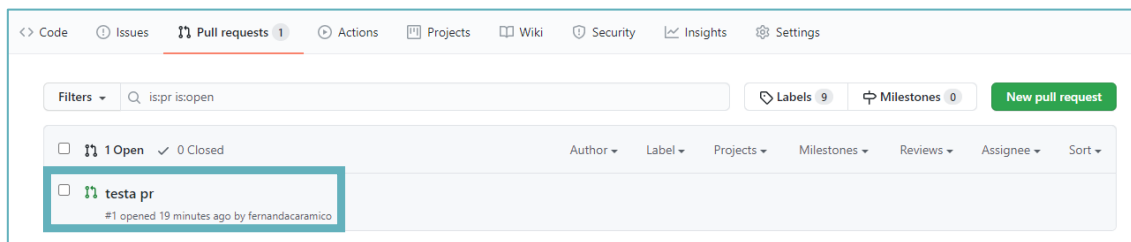
Linked issues
Use [Closing keywords](#) in the description to automatically close issues

3. Aprovando pull request

Acesse a página das Pull Requests (o redirecionamento poderá ser automático).



A página de PRs é como abaixo, contendo a PR aberta:



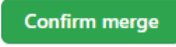
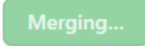


Clique na PR desejada. A tela será como a imagem seguinte, onde você poderá ver:

1. O título da PR;
 - a. Este título é o adicionado no passo anterior.
2. O status da PR e as branches envolvidas neste processo;
 - a. Podemos ver que a PR está aberta e o código será inserido em "main" a partir de "feature/testando-pr".
3. Comentários adicionados e commits envolvidos na alteração;
 - a. Aqui vemos que esta PR levará apenas 1 commit para a branch original.
4. Uma dica de como adicionar mais código nesta mesma PR;
 - a. Esta funcionalidade só é possível quando o Status da PR é "Open" (aberta).
5. Informação de conflito ou não conflito;
 - a. Explicaremos mais à frente como lidar quando é detectado um conflito.
6. Botão para aprovar a PR;
 - a. Esta funcionalidade é mostrada, ou não, dependendo das políticas de segurança da sua empresa.
7. Espaço para adicionar comentários na PR;
8. Botão para fechar/reprovar a PR.
 - a. Ao fechar uma PR você está reprovando a solicitação de adição do código de uma branch para a outra. A branch com as alterações continua existindo.

The screenshot shows a GitHub Pull Request (PR) titled "testa pr #1" (1). The PR is created by "fernandacaramico" and aims to merge 1 commit into the "main" branch from the "feature/testando-pr" branch (2). The interface includes tabs for Conversation (0), Commits (1), Checks (0), and Files changed (1) (3). A comment from "fernandacaramico" states "Testando PR sem conflitos." (4). Below the comment, a message indicates that continuous integration has not been set up but that the branch has no conflicts with the base branch, allowing for automatic merging (5). A green button labeled "Merge pull request" (6) is visible. At the bottom, there is a "Write" tab with a text area for comments (7) and a "Close pull request" button (8).

Analise as alterações feitas:

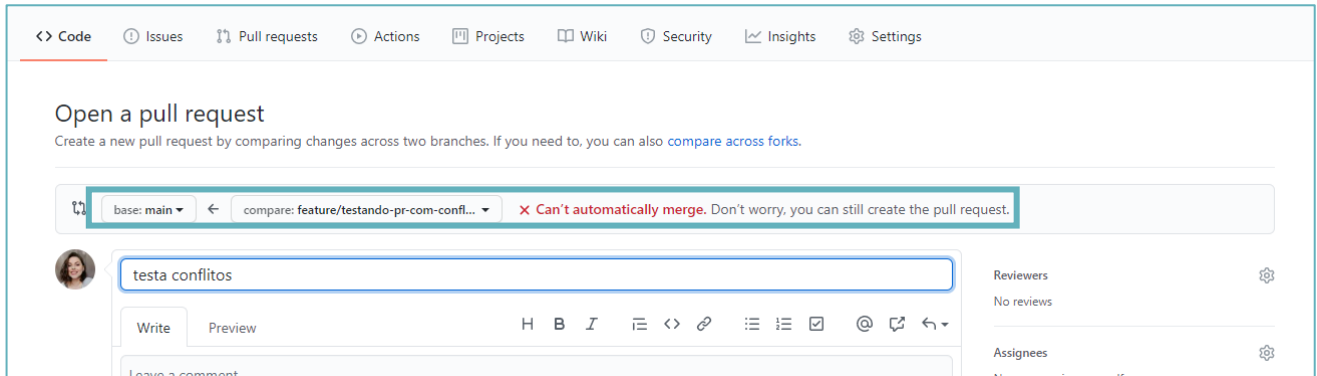
- Se quiser reprovar a PR (ou seja, não aceitar a atualização), clique em .
- Se quiser aprovar a PR, clique em  para "mergear" as alterações.
 - O botão irá alterar para . Clique nele para efetuar o "merge", a adição do código novo ao código original.
 - O botão irá alterar para . Aguarde o "merge" ser finalizado.
 - A Pull Request será então finalizada com sucesso! Você pode já deletar a branch mas tenha cuidado, tenha certeza de que ninguém da sua equipe está usando.

A notification box with a purple icon and text: "Pull request successfully merged and closed. You're all set—the 'feature/testando-pr' branch can be safely deleted." A button labeled "Delete branch" is located to the right.

Feito! Você acaba de abrir e aprovar sua primeira PR!

4. E se der algum conflito?

Se o Git identifica que há conflito entre as alterações que você desenvolvedor está tentando efetuar e o código já presente na branch original, ao criar a Pull Request será exibido um aviso de conflito como abaixo:



✗ Can't automatically merge.

Significa: "Não é possível mergear automaticamente".

Ainda assim você pode criar a Pull Request! Clique em

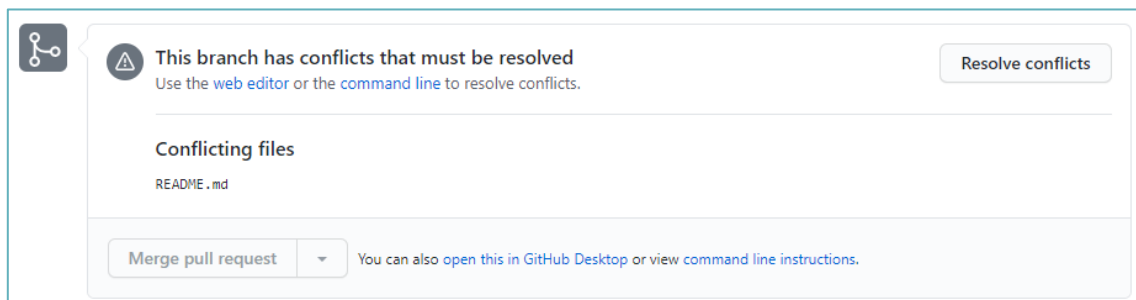
Create pull request

Ao aprovar a Pull Request (Passo 3), o botão

Merge pull request

estará desabilitado e você deverá resolver os conflitos antes de prosseguir, como é apontado na tela:

"Esta branch tem conflitos que devem ser resolvidos."



Resolve conflicts



Resolve conflicts

O botão

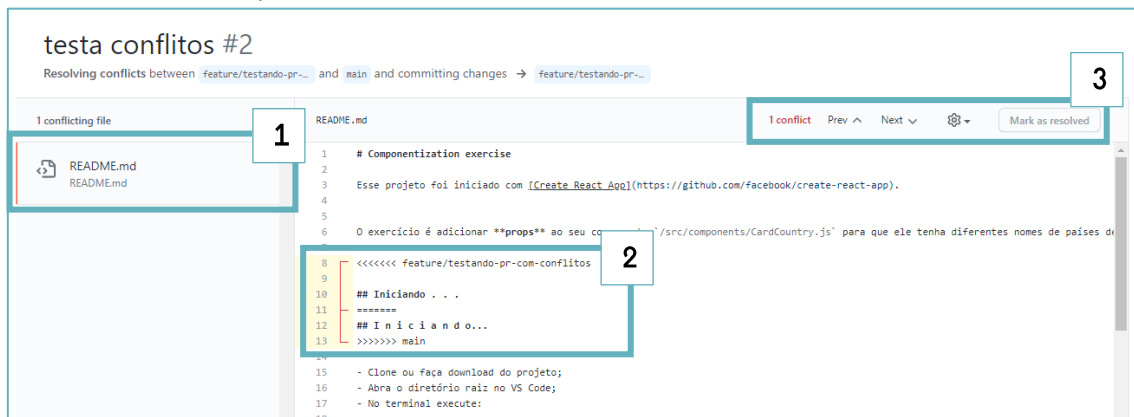
apareceu!

Precisamos resolver os conflitos antes de merger as branches, então clique nele.

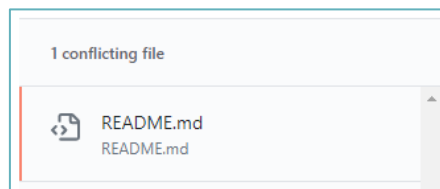
4.1 Resolvendo conflitos

Ao clicar em **Resolve conflicts**, a página será como a seguir:

1. **Onde**, em qual arquivo, está o conflito?
 - Clique nestes itens para ir resolvendo os conflitos, arquivo por arquivo.
2. **O que** está conflitante?
 - Este espaço é editável, então você pode modificar o texto para resolver os conflitos.
3. **Quantos** conflitos há?
 - Aqui você vai ver a quantidade de conflitos a serem resolvidos, poderá acessá-los e sinalizar quando forem resolvidos.



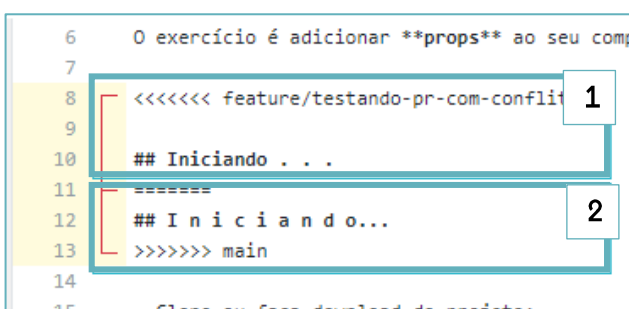
Onde? O conflito foi apenas em um arquivo com título README.md, como vemos em:



"1 arquivo conflitante – README.md"

O que? Em main está escrito **## Iniciando...** mas a branch de nome "feature/testando-pr-com-conflitos" está tentando alterar esta linha para **## Iniciando . . .**, como podemos ver no trecho destacado abaixo:

1. Conteúdo de branch "feature/testando-pr-com-conflitos";
2. Conteúdo de branch "main".



Para resolver o conflito, digite o que deseja que seja a versão final e corrigida deste trecho (linhas 8 até 13). Aqui, vamos escolher para que seja “## Como começar”. Vamos então apagar o conteúdo das linhas em destaque que estão causando o conflito e inserir o que desejamos que contenha no arquivo.

```
6    O exercício é adicionar **props** ao seu
7
8    ## Como começar:
9
10   - Clone ou faça download do projeto;
```

Quantos? Aqui, tivemos apenas um conflito como pudemos ver em **1 conflict**. Repita o processo em todos os conflitos presentes nos seus arquivos.

Assim que as alterações desejadas forem feitas, clique em **Mark as resolved**.

À medida que você for clicando em “Mark as resolved” (marcar como resolvido), os conflitos serão atualizados:

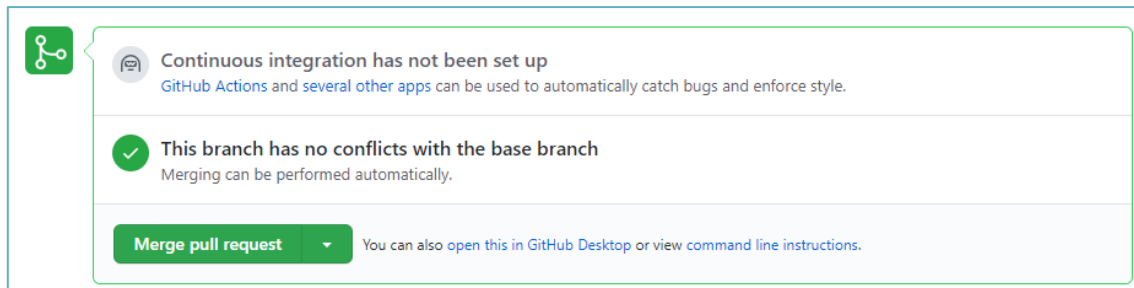


E será possível “commitar o merge” no botão **Commit merge** que apareceu!

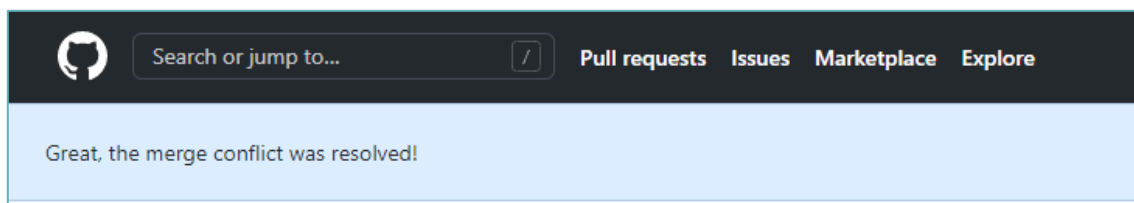


Clique no botão **Commit merge**. Agora, ao acessar esta PR (redirecionamento automático), o merge de conteúdo das branches poderá ser efetuado com sucesso, como podemos ver:

✓ This branch has no conflicts with the base branch – “Esta branch não tem conflitos com a branch de base, original”.



O Github também informará o sucesso em resolver conflitos em:



Agora seu código adicionado em uma branch separada foi adicionado a outra branch, no nosso caso, main.

Agora volte aonde você efetuou os comandos de add/commit/push e execute os comandos abaixo, para trazer uma versão da main contendo seu novo código.

```
git checkout main
git pull
```

Feito! Você acaba de criar branches, abrir PRs, resolver conflitos e trazer suas alterações para o repositório local!

Onde estudar mais sobre branches e conhecer sobre Git Flow?

- Documentação do Git
 - <https://git-scm.com/book/pt-br/v2/Branches-no-Git-Branched-em-poucas-palavras>
- Artigo sobre GitFlow no Alura
 - <https://www.alura.com.br/artigos/git-flow-o-que-e-como-quando-utilizar>

Qualquer dúvida, estamos à disposição.

Bons estudos!

Equipe SPTECH