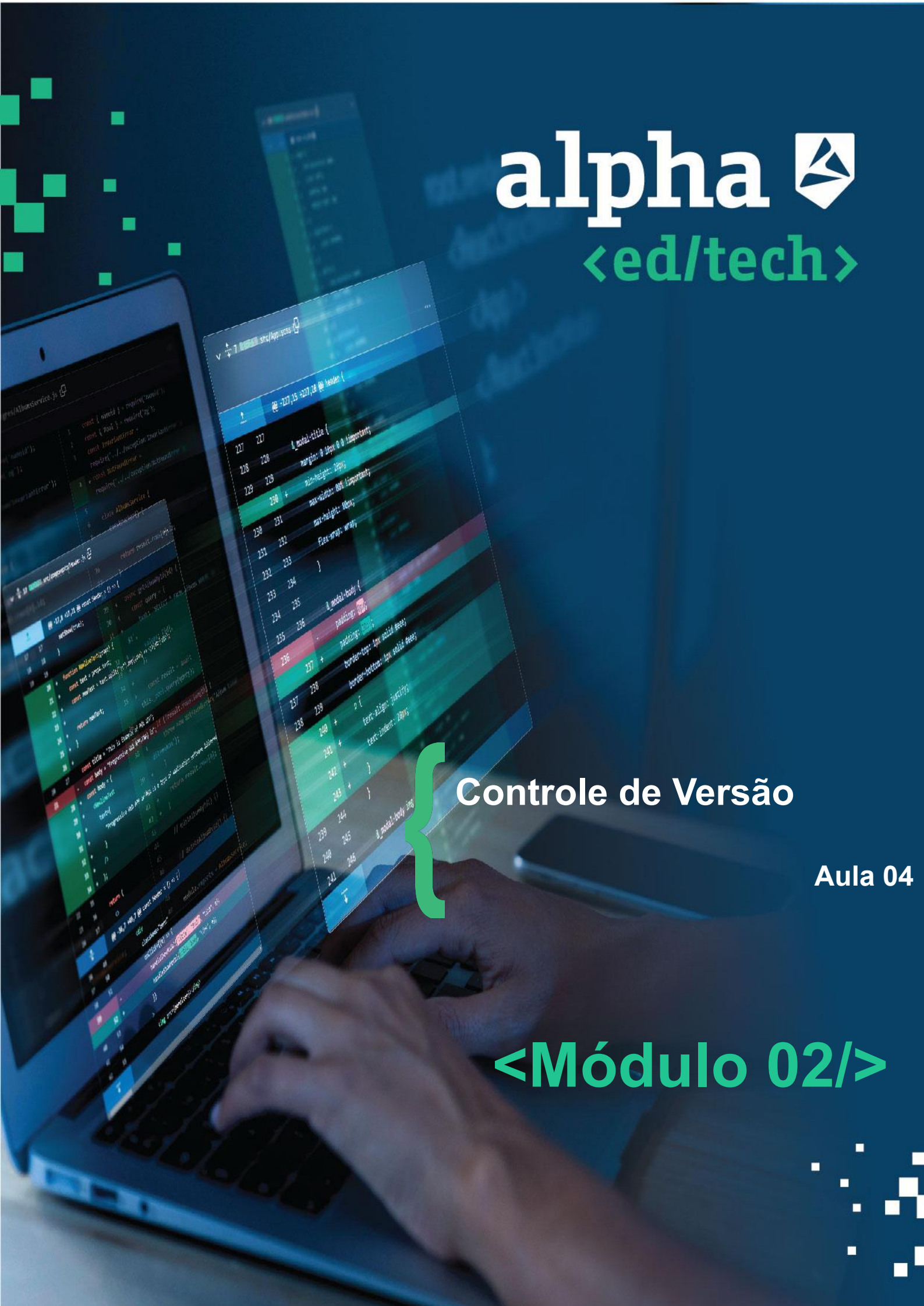
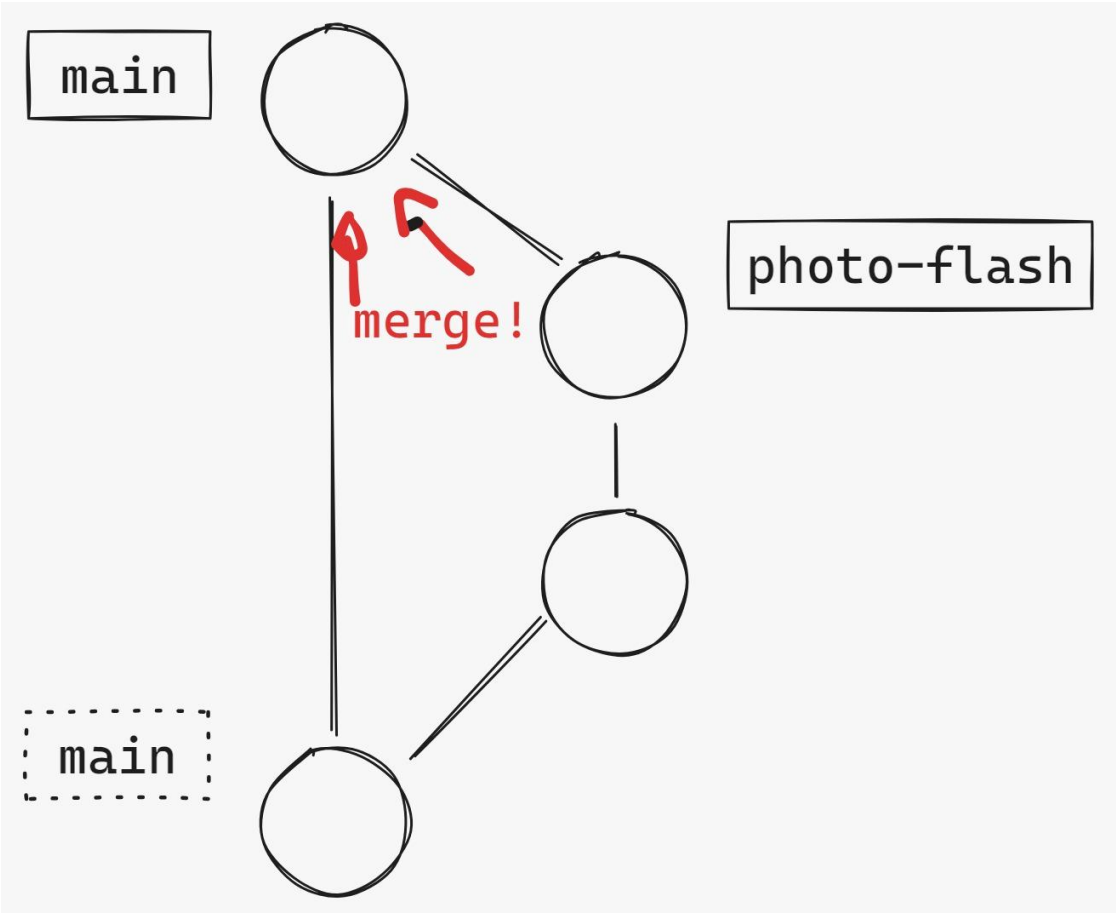

alpha
<ed/tech>
Controle de Versão
Aula 04
<Módulo 02/>

Pull Request

Introdução

Imagine o mesmo cenário de onde paramos na aula anterior, o guia completo sobre fotografia digital. Onde você, Emily, Alice e Bob e estão trabalhando no mesmo projeto.

Vamos lembrar na aula passada, onde Emily criou uma branch `photo-flash`, publicou a branch no repositório remoto e após terminar todas a mudanças mandou a você um email para que você faça um merge da branch dela na branch `main`.



Este fluxo é um tanto trabalhoso, depende de ferramentas externas de comunicação e além disso não é um método que trás muita segurança de versionamento para seu projeto.

Você resolve melhorar o fluxo de trabalho do seu projeto, então a solução encontrada foi o Pull Request.

A diferença da Pull Request (PR) para o fluxo de merge da aula passada é:

- Na aula passada, você precisou baixar a branch photo-flash da Emily, fazer o merge sobre a main, e então push da main
- Com a PR, a Emily solicita merge da branch dela pelo próprio GitHub. Você revisa as alterações dela pelo site mesmo. E então clica num botão para confirmar o merge da photo-flash sobre a main, tudo acontece no remoto (GitHub).

Pull Request

Mas por que Pull Request (PR)?

Vimos como podemos organizar melhor nosso projeto e evitar conflitos, mas porque não usar o método de você mesmo fazer um merge? A resposta é simples: Comunicação, Organização e Automação!

O Pull Request tem os seguintes benefícios:

1. **Comunicação**
   Informe outros colaboradores sobre as alterações por push feitas em uma ramificação de um repositório, você poderá discutir e revisar as possíveis alterações com colaboradores.

2. **Organização**
   Adicione confirmações, aprovações e testes antes que as alterações sejam mescladas na ramificação base.

3. **Automação**
   Adicione eventos e processos automatizados para um Pull Request em determinada branch.

Comunicação

Dentro de um Pull Request é possível que a equipe ou revisor façam comentários ou iniciem discussões sobre um determinado conteúdo, onde é possível uma colaboração diretamente pelo GitHub, sem a necessidade de ferramentas externas, que tornarão a comunicação menos assertiva e mais lenta.

Organização

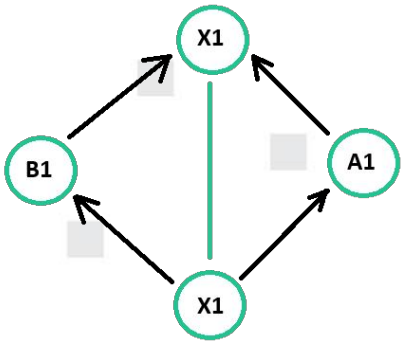
É possível restringir o merge, para que só ocorra com determinadas aprovações ou testes, que permitirá uma melhor organização para a equipe e para o gestor da equipe, centralizando as decisões e rumos do projeto.

Automação

É possível também adicionar eventos ao Pull Requests, com testes automatizados e vários outros processos de automação e integração com outras ferramentas como o GitHub Actions.

Pull Request

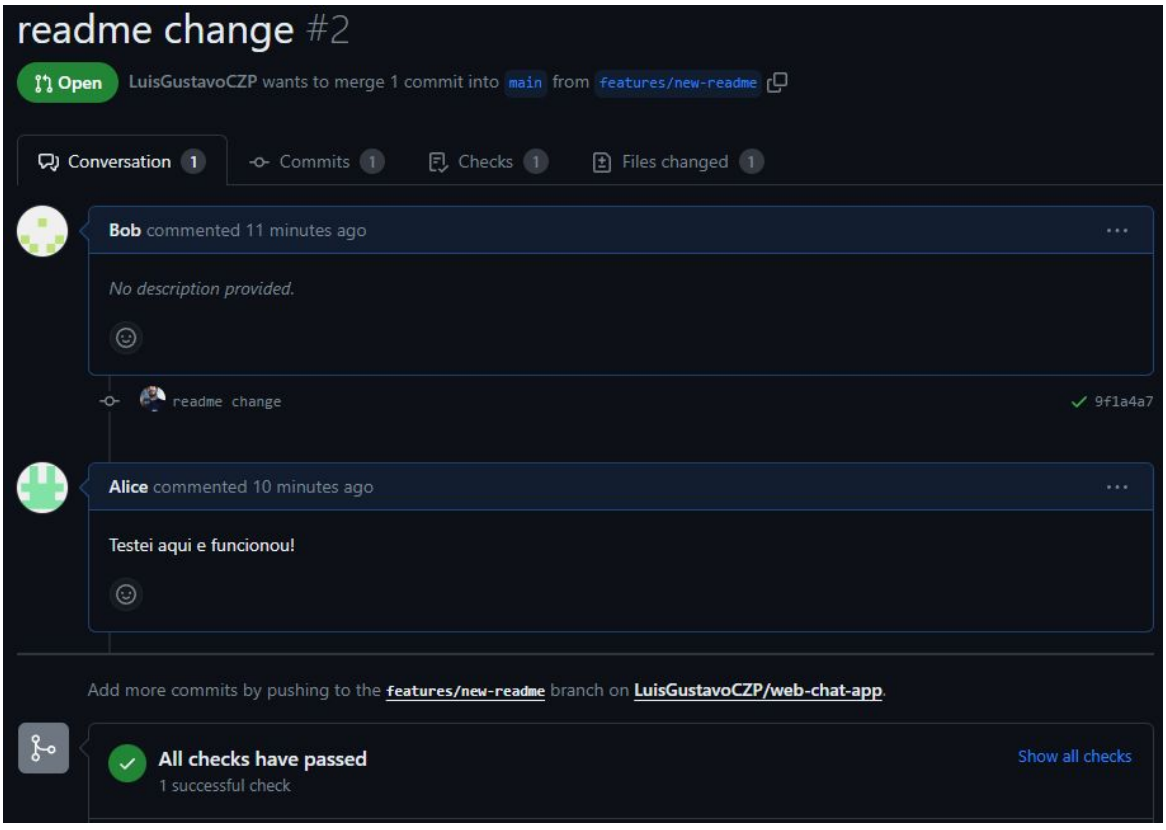
Como funciona esse novo fluxo?



1. O colaborador precisa criar uma nova branch (origin/a ou origin/b) a partir da branch base (origin/x);
2. Completar a tarefa;
3. Caso haja conflitos, deve resolve-los (veremos a seguir);
4. Publicar a branch no repositório remoto;
5. O colaborador solicita um PR na branch base (origin/x);

Agora vamos trazer este cenário para este projeto, onde estão trabalhando Bob e Alice.
E neste caso, você quer ter um melhor controle, para que Bob e Alice possam ter feedbacks e uma comunicação mais assertiva com você.

Então é combinado entre vocês que só você pode aceitar ou rejeitar um Pull Request, enquanto eles podem solicita-lo. E todos podem revisa-lo e dar feedbacks.

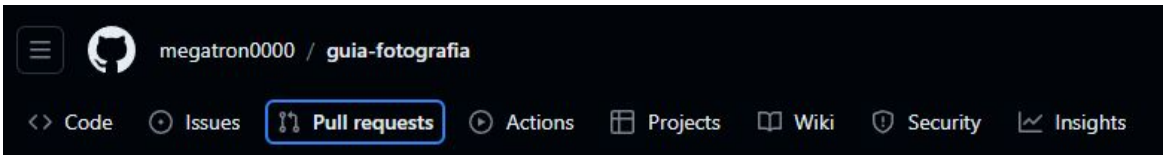


Pull Request

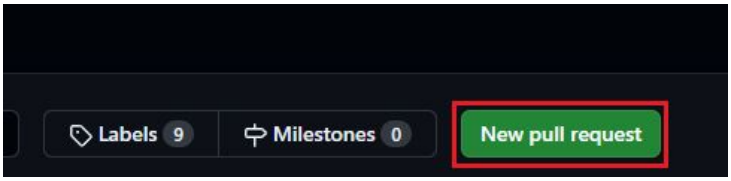
Criando um Pull Request

O primeiro passo é ter uma branch já criada e publicada remotamente.
A partir deste pontos seguiremos os seguintes passos:

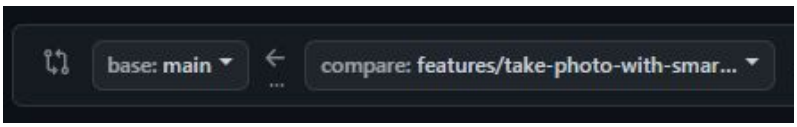
1. No repositório do GitHub, abaixo do nome do repositório, clique em **Pull Request**.



2. Após isso, clique em "New Pull Request"



3. Na próxima tela será possível escolher a branch onde será feito o merge e a branch a qual queremos mergear através do PR, respectivamente.



4. Ao escolher ambas as branches, o próximo passo é clicar em "Create Pull Request"

5. Digite um título e uma descrição para a pull request.

6. Para criar uma solicitação de pull pronta para revisão, clique em Criar Solicitação de Pull. Para criar uma solicitação de pull de rascunho, use o manu suspenso, selecione Criar Solicitação de Pull de Rascunho e clique em Solicitação de Pull de Rascunho.

Se durante o processo for possível ver essa frase:

✓ Able to merge.

É porque não há conflitos e o merge poderá ser feito sem quaisquer problemas

Se durante o processo for possível ver essa frase:

× Can't automatically merge.

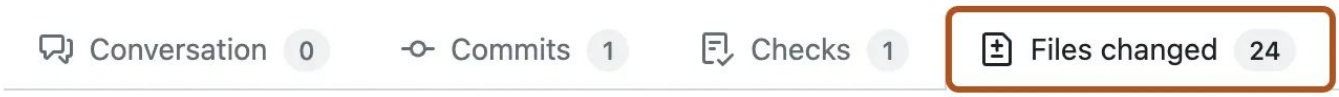
É porque há conflitos entre as branches e será necessário a resolução destes conflitos antes do merge.

Diff

Comparando Branches

Como mantenedor e revisor do projeto, será necessário revisar as mudanças feitas pelos colaboradores, para tanto, será de muita utilidade a ferramenta de comparação oferecida pelo GitHub.

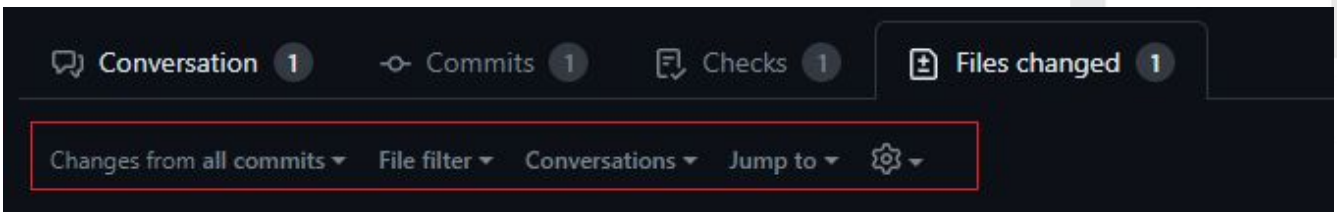
Ao entrar no PR que foi solicitada por um colaborador, é possível ver a seguinte opção: Files Changed.

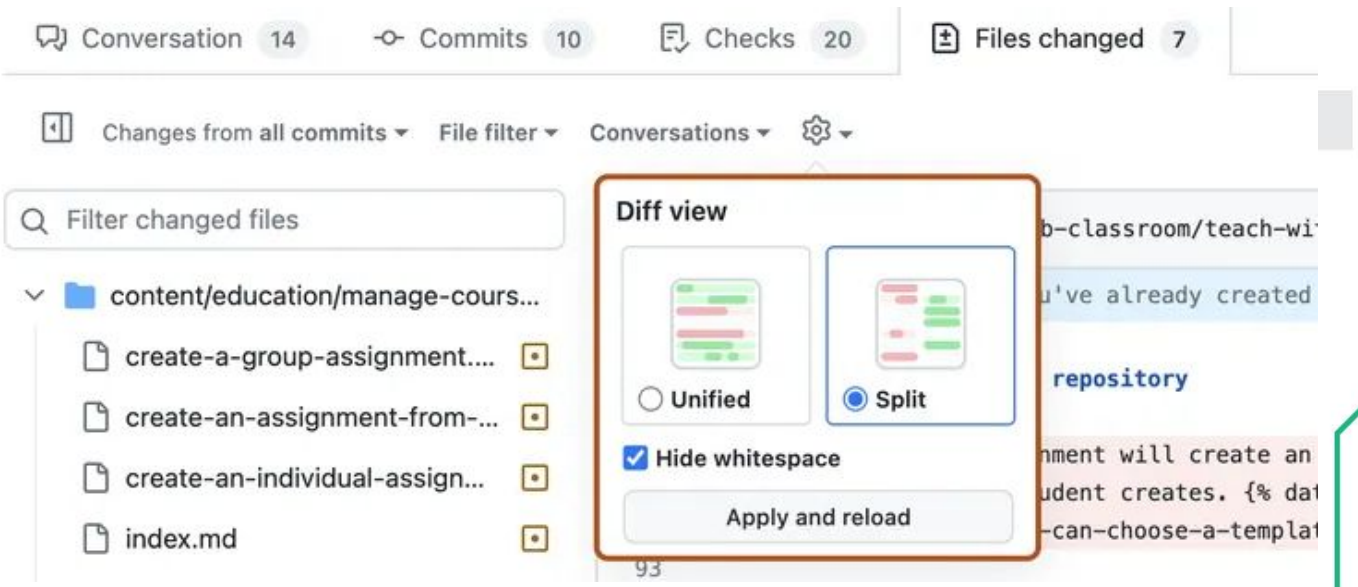


Em vez de exibir os commits em si, você pode ver as alterações propostas como elas aparecerão nos arquivos assim que a pull request passar pelo merge. Os arquivos são exibidos em ordem alfabética na guia Arquivos alterados.

As adições aos arquivos são exibidas em verde e são precedidas por um sinal "+", enquanto o conteúdo que foi removido é exibido em vermelho e é precedido por um sinal "-".

Algumas opções adicionais podem ser encontradas nas opções de exibição, como filtros, conversas, e configurações de visualização.





Diff

Comparando Branches pelos comandos Git

Também será possível a comparação entre branch através do comando `git diff`. Há dois métodos de comparação para o comando `git diff`:

- Dois pontos (`git diff A..B`);
- Três pontos (`git diff A...B`);

Por padrão, as solicitações de pull no GitHub mostram uma comparação de três pontos.

Comparação Git de dois pontos

A comparação de dois pontos mostra a diferença entre o estado mais recente do branch base (por exemplo, main) e a versão mais recente do branch do tópico.

Um diff de dois pontos compara duas referências de *committish* do Git, como SHAs ou IDs de objeto (OIDs, Object IDs), diretamente entre si. No GitHub, as referências de *committish* do Git em uma comparação de diff de dois pontos devem ser enviadas por push ao mesmo repositório ou para suas bifurcações.

Se desejar simular um diff de dois pontos em uma pull request e ver uma comparação entre as versões mais recentes de cada branch, você poderá fazer merge do branch base no branch de tópico, o que atualiza o último ancestral comum entre seus branches.

Comparação Git de três pontos

A comparação de três pontos mostra a diferença entre a confirmação comum mais recente de ambos os branchs (base de mesclagem) e a versão mais recente do branch do tópico.

Como a comparação de três pontos se compara com a base de mesclagem, ela se concentra "no que uma solicitação de pull apresenta".

Quando você usa uma comparação de dois pontos, ela muda quando o branch base é atualizado, mesmo que você não tenha feito nenhuma alteração no branch do tópico. Além disso, uma comparação de dois pontos se concentra no branch base. Isso significa que tudo o que você adicionar é exibido como ausente do branch base, como se fosse uma exclusão e vice-versa. Como resultado, as alterações que o branch de tópico introduz tornam-se ambíguas.

Por outro lado, comparando os branches usando a comparação de três pontos, as alterações no branch de tópico estarão sempre na comparação se o branch base for atualizado, pois ela mostra todas as alterações desde que os branches divergiram.

Review

Introdução

As revisões permitem que colaboradores comentem sobre as alterações propostas em pull requests, aprovem as alterações ou solicitem outras alterações antes do merge da pull request. Os administradores do repositório podem exigir que todas as pull requests sejam aprovadas antes de sofrerem o merge.

Revisão de Pull Request

Depois que uma solicitação de pull for aberta, qualquer pessoa com acesso de leitura poderá revisar as alterações propostas e adicionar comentários a elas. Você também pode sugerir alterações específicas às linhas de código, que o autor pode aplicar diretamente a partir da pull request.

Por padrão, em repositórios públicos, qualquer usuário pode enviar análises que aprovem ou solicitem alterações em um pull request. Os proprietários da organização e os administradores de repositório podem limitar quem pode conceder revisões de solicitação de pull ou alterações de solicitação de aprovação.

Os proprietários de repositório e colaboradores podem solicitar uma revisão de pull request de uma pessoa específica. Os integrantes da organização também podem solicitar uma revisão de pull request de uma equipe com acesso de leitura ao repositório. Você pode especificar um subconjunto de membros da equipe a ser atribuído automaticamente no lugar de toda a equipe.

As revisões permitem discussão das alterações propostas e ajudam a garantir que as alterações atendam às diretrizes de contribuição do repositório e outros padrões de qualidade. Você pode definir quais indivíduos ou equipes possuem determinados tipos de área de código em um arquivo *CODEOWNERS*. Quando uma pull request modifica código que tem um proprietário definido, esse indivíduo ou equipe será automaticamente solicitado como um revisor.

Você pode agendar lembretes para solicitações de pull que precisam ser revisadas.

Status de Revisão

Uma revisão tem três status possíveis:

- **Comentário**
  Envie comentários gerais sem aprovar explicitamente as alterações nem solicitar alterações adicionais.

- **Aprovação**
  Envie comentários e aprove a mesclagem das alterações propostas na solicitação de pull.

- **Solicitação de alterações**
  Envie comentários que precisam ser resolvidos antes que a solicitação de pull possa ser mesclada.

Review

Solicitação de Revisão

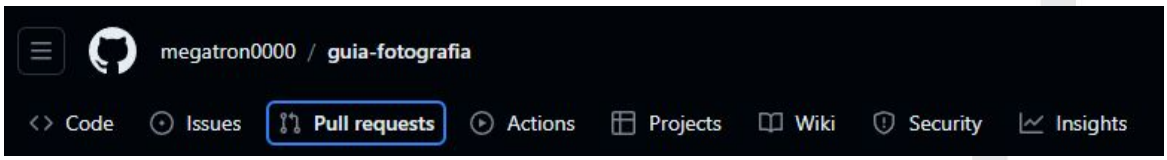
Você pode solicitar uma revisão para uma pessoa específica ou sugerida.

Se você solicitar uma revisão, outras pessoas com acesso de leitura no repositório poderão revisar sua pull request. Depois que alguém revisar sua PR e você fizer as alterações necessárias, você poderá solicitar novamente a revisão do mesmo revisor.

Se o revisor solicitado não enviar uma revisão e a PR atender aos requisitos de capacidade de mesclagem do repositório, você ainda poderá mesclar a PR.

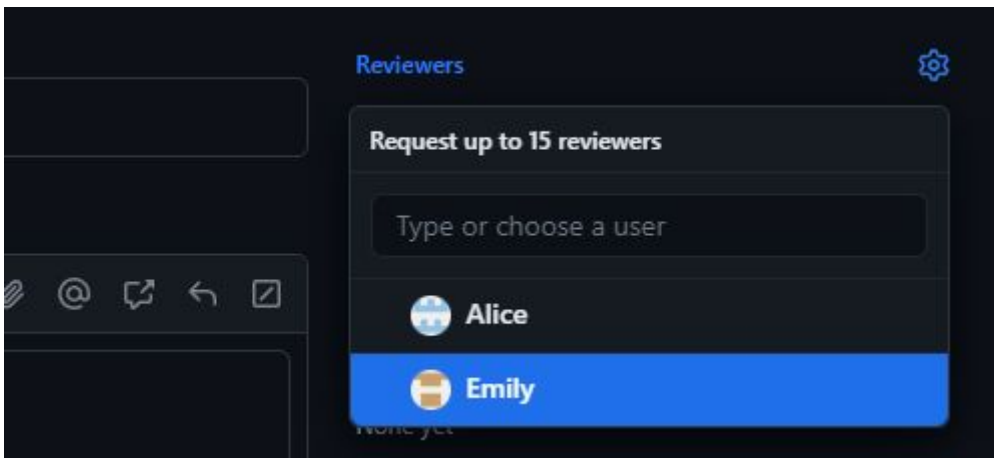
Solicitando uma Revisão

1. No repositório do GitHub, abaixo do nome do repositório, clique em **Pull Request**.



2. Na lista de pull requests, clique na pull request que deve ser revisada por uma pessoa ou equipe específica.

3. Para solicitar uma revisão de uma pessoa sugerida em Revisores, ao lado do nome de usuário dela, clique em Solicitar.



4. Opcionalmente, para solicitar uma revisão de alguém que não seja uma pessoa sugerida, clique em Revisores.
   Se você souber o nome da pessoa ou da equipe da qual deseja obter uma revisão, digite o nome de usuário da pessoa ou o nome da equipe da qual solicitando a revisão das alterações. Clique no nome da equipe ou no nome de usuário para solicitar a revisão.

5. Depois que a pull request for revisada e você fizer as alterações necessárias, você poderá solicitar que ela seja revisada novamente por um revisor. Procure Revisores na barra lateral direita e clique em ao la do nome do revisor cuja revisão deseja obter.

Review

Revisando alterações

Em uma solicitação de pull, você pode examinar e discutir commits, arquivos alterados e as diferenças (ou "comparação") entre os arquivos nos branches base e de comparação.

Você pode revisar as alterações em um arquivo de pull request por vez. Ao revisar os arquivos em um pull request, você pode deixar comentários individuais em alterações específicas.

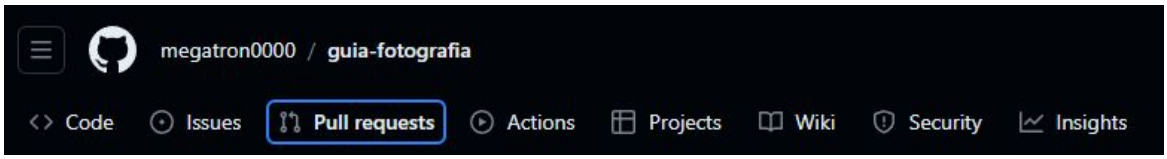
Após terminar de revisar cada arquivo, você pode marcar o arquivo como visualizado. Isso aninha o arquivo e ajuda a identificar os arquivos que ainda precisam ser revisadas.

Uma barra de progresso no cabeçalho do pull request mostra o número de arquivos que você visualizou.

Depois de revisar todos os arquivos você desejar, você pode aprovar a solicitação de pull ou solicitar alterações adicionais enviando a sua revisão com um comentário resumido.

Revisando alterações

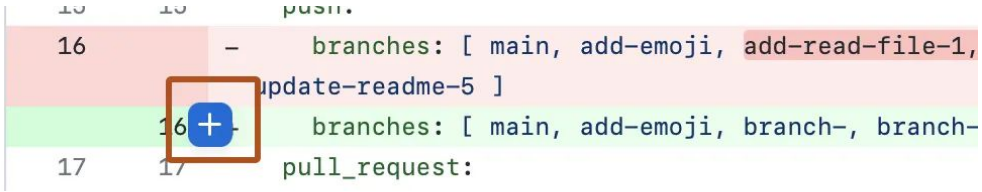
1. No repositório do GitHub, abaixo do nome do repositório, clique em **Pull Request**.



2. Selecione a Pull Request que você quer revisar.

3. Na solicitação de pull, clique em Files changed .

4. Passe o mouse sobre a linha de código em que você gostaria de adicionar um comentário e clique no ícone de comentário azul. Para adicionar um comentário em várias linhas, clique e arraste para selecionar o intervalo de linhas e clique no ícone de comentário azul.



5. No campo comentário, digite o seu comentário.

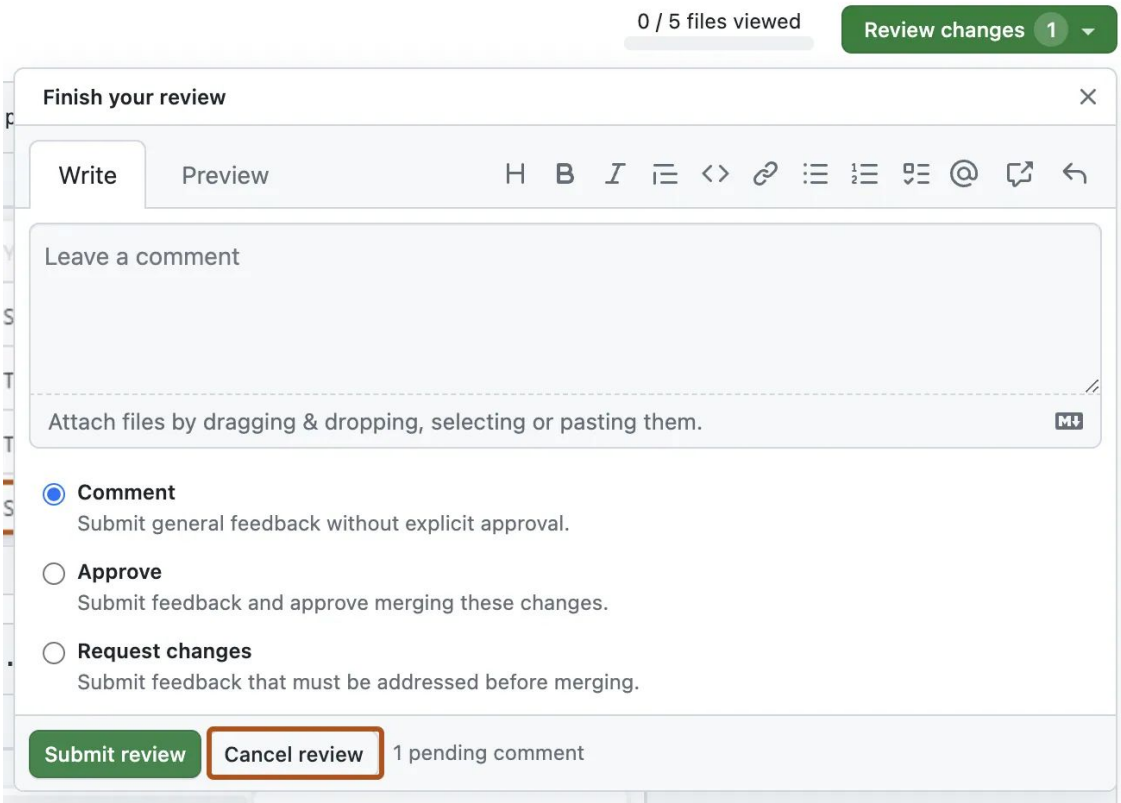
6. Opcionalmente, para sugerir uma alteração específica nas linhas, clique em e editar o texto no bloco de sugestão.

7. Para comentar diretamente em um arquivo, à direita do arquivo, clique em e digite seu comentário.

Review

8. Quando terminar, clique em **Start a review**.
   Se você já tiver iniciado uma revisão, clique em **Add review comment** sobre a revisão.

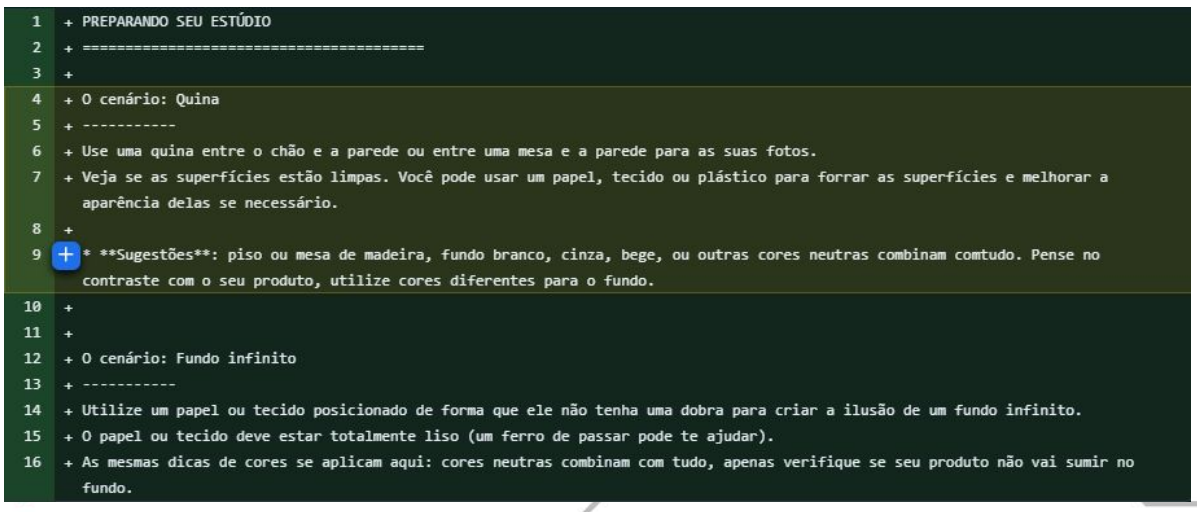
Antes de enviar a revisão, os comentários em linha ficam com o status pendente e somente você pode visualizá-los. Você pode editar os comentários pendentes a qualquer momento antes de enviar a revisão. Para cancelar uma revisão pendente, incluindo todos os comentários pendentes, clique em **Review Changes** acima do código alterado e em **Cancel Review**. Para concluir, clique em **Submit Review**.



Referenciando commits e issues

É possível referenciar os arquivos, commits e issues nas revisões, assim como um trecho específico:

- para isso, clique no número da linha na barra lateral, aperte shift e clique na última linha desejada, depois clique no botão ⊕ . Pronto, seu review irá especificar o trecho desejado.



Review

Conversation

Você pode exibir todas as revisões que uma pull request recebeu na linha do tempo Conversation (Conversa), assim como pode ver revisões por proprietários e colaboradores do repositório na caixa de merge da pull request.

Resolvendo Conversa

É possível resolver uma conversa em um pull request se você abriu o pull request ou se você tem acesso de gravação ao repositório em que o pull request foi aberto.

Para indicar que uma conversa na guia Arquivos alterados foi concluída, clique em Resolver conversa.

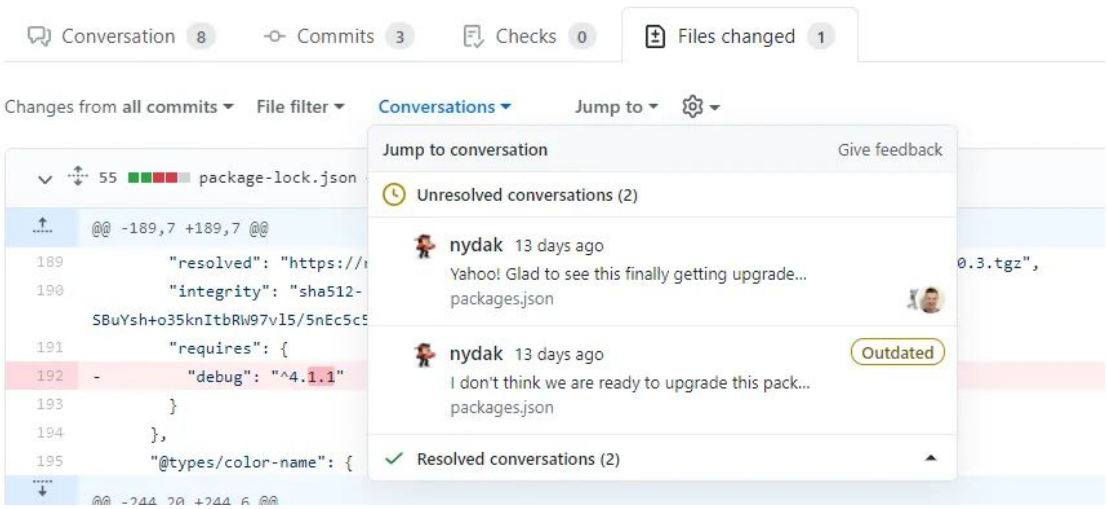
Toda a conversa será colapsada e marcada como resolvida, tornando mais fácil encontrar conversas que ainda precisam ser consideradas.

Se a sugestão em um comentário estiver fora do escopo do seu pull request, você pode abrir um novo problema que rastreia os comentários e relaciona o comentário original.

Descobrindo e navegando por conversas

Você pode descobrir e procurar todas as conversas na sua solicitação de pull usando o menu Conversas, que é mostrado na parte superior da guia Arquivos Alterados.

Nesta visualização, você pode ver quais conversas não foram resolvidas, quais foram resolvidas e desatualizadas. Isso facilita a descoberta e a resolução de conversas.



Conflitos

Introdução

Você se lembrou de quando falamos sobre resolução de conflitos?

Vamos a um caso hipotético, de forma análoga ao caso da aula passada, onde Alice e Bob fizeram uma bifurcação da branch. Neste caso em especial, Alice e Bob não criaram uma bifurcação, mas um conflito, onde em ambas as branchs o mesmo arquivo foi alterado separadamente.

Isso causa um problema para o Git entender qual é de fato a última atualização que queremos do arquivo, sendo assim não possui uma resolução automática, cabendo ao usuário a resolução dos conflitos de forma manual.

Resolvendo Conflitos

1. Primeiro iremos migrar para a branch da qual estamos fazendo a PR
   `git checkout photo-with-phone;`

2. A seguir iremos mergear o conteúdo da branch para qual estamos fazendo a PR (rebase)
   `git merge main;`

3. Resolvemos os conflitos

4. Para finalizar, subiremos as alterações e resolução para nossa branch atual
   `git push;`

Evitando Conflitos

Vimos que a resolução de conflitos é relativamente "simples", porém pode ser desastrosa de acordo com cada caso, por isso, a melhor estratégia é evitar conflitos. Mas como? Aqui vão algumas regras práticas:

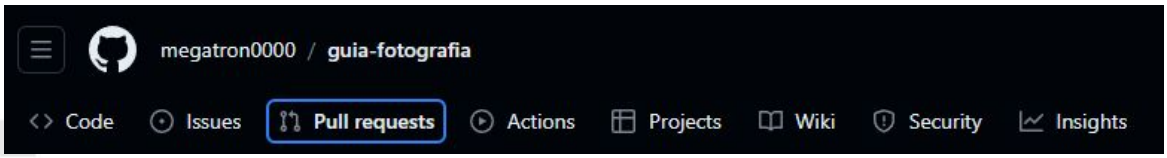
- Cada colaborador deve trabalhar em seu próprio arquivo.
- Cada feature (funcionalidade ou tarefa) deve estar contida em uma branch específica para ela.
- Se mais de um colaborador estiver trabalhando na mesma feature, é necessário ter uma ramificação(branch) principal para a feature e cada colaborador ter sua própria branch. Ao final cada um deve fazer seu PR na ramificação principal.

Conclusão

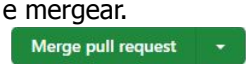
Finalizando um Pull Request

Uma vez resolvidos os conflitos (se houver) e as conversações (se houver) é possível iniciarmos o Merge do Pull Request. Para tanto, vamos aos seguintes passos:

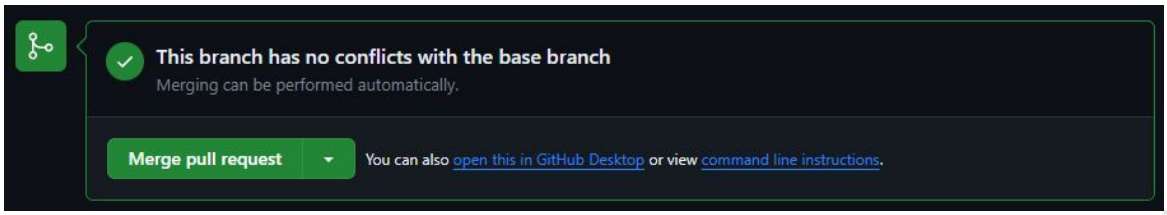
1. No repositório do GitHub, abaixo do nome do repositório, clique em **Pull Request**.



2. Selecione a Pull Request que você quer finalizar e mergear.



3. Desça a página até a seguinte parte e clique em .



4. Depois clique em  para finalizar.

Pronto, agora seu Pull Request foi finalizado, a partir de agora temos 2 opções:

- Apagar a branch usada para Pull Request: Geralmente usado quando não há a necessidade de manter a branch, visto que ela já está mergeada em outra branch. A vantagem é que ficamos com a lista de branchs mais limpa, onde só temos as necessárias e que estão sendo usadas.

- Manter a branch usada para Pull Request: Esta prática geralmente é usada por alguns gestores como backup de segunça, até que seja lançada a release da qual a feature faz parte. A vantagem é que caso algo dê errado teremos um backup, ao menos até a release mostrar-se completa e funcional, então todas a branchs que contém as features desta release serão apagadas.

O próximo passo a seguir é irmos para a branch na qual fizemos o pull request e atualizarmos ela no nosso repositório local. Para isso vamos usar:

1. `git checkout main`

2. `git pull`

Pronto, nossa branch `main` está atualizada no nosso repositório local.

Com isso você aprendeu as várias etapas de um pull request, e também como revisar, solicitar revisão e ver a revisão de outros colaboradores!