BOAS PRÁTICAS COM *GIT* – orientadas ao *GITFLOW*

Seguem orientações de boas práticas para atuarmos o mais próximo possível do fluxo de trabalho denominado *GitFlow*. Sendo assim, devemos criar *branchs* mais objetivas, designadas a somente um desenvolvedor, respeitando os seguintes pontos:

Branch* principal:** Tem como padrão o nome ***master. O projeto sempre se inicia com esta ramificação (*branch*), na qual a mesma receberá a versão consolidada funcional (sem erros) e homologada (com aprovação na qualidade de código) mais recente.

Branch* de desenvolvimento:** Tem como padrão o nome ***develop. Criada a partir da *master*, representando a junção de todas as demais ramificações de funcionalidades e de ajustes do projeto, sendo uma versão prévia da *master*. A *branch* tem como função receber a mescla de códigos de outras ramificações e realizar os devidos testes, antes de ser mesclada à ramificação principal (*master)*.

Branch* de funcionalidade:** Tem como padrão o prefixo ***feature. Criada a partir da *develop*, esta ramificação é temporária e tem o objetivo de tratar apenas de novas implementações associadas a uma ou mais funcionalidades pertencentes a regra de negócio do projeto, desde que previamente certifiquem-se na criação da branch que seu objetivo irá evitar ao máximo futuros conflitos. Ao fim de sua implementação deve-se solicitar a mescla (MR – *Merge Request*) com a *develop*. Se houverem correções necessárias no código, deverá realizar na mesma ramificação até que esta esteja homologada e aprovada pelo revisor de código principal.

Branch* de ajuste para suporte no projeto**: Tem como padrão o prefixo ***support. Ramificação temporária e destinada ao suporte, de forma que toda nova implementação geral ou específica, que não esteja ligada diretamente com alguma funcionalidade da regra de negócio, deverá ser feita em uma *branch* separada. Por exemplo: Novas implementações de leitores de arquivos ou geradores de documentos.

Branch* de ajuste para correções no projeto:** Tem como padrão o prefixo ***fix. Se surgirem erros ou alterações necessárias, após o MR, devemos criar uma nova *branch* temporária, a fim de corrigi-los.

Todo nome de branch deve ser resumido (objetivo), estar em letras minúsculas e separados por *underline* ("_"). Conforme respectivos exemplos e seus objetivos:

- **master** (branch principal)
- **develop** (branch de desenvolvimento)
- **feature_saldo** (branch da funcionalidade de saldo)
- **feature_transferencia** (branch da funcionalidade de transferência)
- **feature_conta_corrente_extrato** (branch da funcionalidade de conta corrente > extrato)
- **support_leitor_planilhas** (branch de suporte para utilitário de leitor de planilhas)
- **support_gerador_numeros** (branch de suporte para utilitário de gerador de números)
- **fix_feature_saldo** (branch para correção de erros na funcionalidade saldo)
- **fix_support_leitor_planilhas** (branch para correção de erros no utilitário de leitor de planilhas)

Ao revisor de código, após aprovada a solicitação de mescla e a mesma estando presente na ramificação *develop,* o mesmo poderá deletar a *branch* temporária de origem (a solicitante).

***Commit*:** Toda mensagem de *commit* deverá refletir a intenção e não apenas o conteúdo do que foi realizado. Por exemplo:

- Commit – Ruim ☹
    - git commit –m 'add UserModel.class'
- Commit – Bom ☺
    - git commit –m 'criação da classe modelo para dados de usuario'

*Fonte(s): https://docs.gitlab.com/ee/topics/gitlab_flow.html e https://www.atlassian.com/br/git/tutorials/comparing-workflows/gitflow-workflow