**Aluno: Matheus Henrique de Souza Oliveira R.A:1744975-2**

**Disciplina: Gerenciamento de Configuração Turma: ESOFT6S**

**Plano de Gerenciamento de Configuração**

Redmine, será utilizado para gerenciar o produto de software, e terá dois projetos, um para gerenciar as mudanças/melhorias no software, e outro projeto para gerenciar as solicitações de correções de falhas no software. Ambas as solicitações podem ser solicitadas pelo(s) cliente(s), ou pode ser uma mudança solicitada internamente por algum colaborador da empresa, que devem criar uma issue que, no caso da solicitação de melhoria, será analisada pelo P.O (Product Owner) do projeto, que irá analisar se essa melhoria faz sentido para o produto, e caso seja uma melhoria válida, prosseguirá para o fluxo de desenvolvimento, passando pela análise de requisitos, análise técnica, até chegar na etapa de testes e por fim ser entregue para o cliente final. Já no caso de ser uma solicitação de correção de erro, a issue deverá passar por uma triagem, que buscará verificar se o erro ainda ocorre ou se já foi corrigido, e caso seja necessário desenvolver a correção, a issue também passará por todo o fluxo de desenvolvimento. Em ambos os projetos, as issues terão um campo onde será informado a qual funcionalidade essa melhoria ou correção de erro está relacionada, dessa forma é possível rastrear quais funcionalidades estão gerando mais falhas e quantas falhas cada funcionalidade tem até o momento, é possível também saber quais funcionalidades sofreram mais ou menos alterações, e todas essas informações possibilitam a equipe saber quais partes do produto estão mais estáveis e quais partes do produto não estão, ou seja, precisam de mais atenção, como uma maior cobertura de testes por exemplo, e através de uma análise das métricas recolhidas através da abertura de issues por funcionalidades, é possível também, saber qual o custo para manter determinada funcionalidade, através das horas gastas para corrigir os erros ou no desenvolvimento de melhorias.

Já o github será utilizado como o repositório de código-fonte, onde será configurado para que todo o commit tenha, obrigatoriamente, o número da issue a que se refere a alteração realizada no software. O repositório irá conter um branch master, que conterá todo o código com as últimas alterações do produto, e mais uma branch que irá conter a última versão ativa do produto, ou seja, a empresa irá dar suporte a uma versão e a cada 1 ano irá atualizar essa versão parando de dar suporte para a versão mais antiga, garantindo através de contratos que os clientes atualizem o sistema mantendo-se na versão mais atualizada possível. Então se no dia 01/01 de 2020 a empresa dá suporte para a versão 1.0.0, no dia 01/01 de 2021 a empresa irá lançar a versão 2.0.0 com as melhorias definidas e detalhadas no roadmap do Redmine, e irá parar de dar suporte para a versão 1.0.0, porém essa versão continuará no repositorio do github para que seja mantido todo o histórico que pode ser necessário no futuro para uma análise de causa raiz por exemplo. As versões do produto serão separados entre versão cheia, que irá conter as melhorias levantadas e definidas no Roadmap e será representado pelo número 1 tendo como exemplo a versão 1.0.0, terá também a versão de melhoria, que irá conter todas as alterações relacionadas às melhorias/mudanças que serão representados pelo número 4 tendo como exemplo a versão 1.4.0, podendo ter nessa versão uma ou mais melhorias, e por fim irá conter também os hotfixes, que serão as versões que irão conter todas as correções de erros e serão representados pelo número 10 tendo como exemplo a versão 1.X.10, e nesse caso os hotfixes serão publicados em todas as versões de melhorias da versão cheia 1, ou seja, se houver a versão de melhoria 10, o hotfix lançado sairá em todas as 10 versões de melhoria, pois um cliente pode estar em produção na versão 1.5.X e outro na 1.8.X por exemplo, e a correção de erro deve estar disponível para todos os clientes independente da versão que o mesmo esteja. Para realizar uma alteração no código, o desenvolvedor irá precisar enviar um PullRequest (PR) e nessa etapa, dois ou mais desenvolvedores irão auditar o código, ou seja, analisar o código em relação a parte técnica e de negócio, e sugerir alterações caso necessário. Caso o PR seja aprovado o desenvolvedor irá realizar o Merge da branch local (criada pelo desenvolvedor) para a branch da versão relacionada ao tipo de alteração realizada (melhoria ou correção de erro).

No Google Docs serão separados os produtos mantidos e comercializados pela empresa por pastas, e dentro de cada pasta irão ter as funcionalidades do produto, e em cada pasta de funcionalidade irá ter toda a documentação relacionada a essa funcionalidade.