Especificação do processo de gerenciamento de versões

Diego, Emerson, Jonata e Vinícius
Universidade Tecnológica Federal do Paraná - Câmpus Cornélio Procópio

A adoção de soluções modernas e com qualidade depende de um processo estabelecido que possa assegurar que o software a ser desenvolvido atenda às necessidades do cliente. Para que isso ocorra é necessário a definição de todos os processos envolvidos e como eles serão interligados. Este documento exemplifica o processo de desenvolvimento de software a ser aplicado na empresa Software Supimpa Tecnologia (2ST), que deseja manter um processo para o gerenciamento de versões do software Klassic, que já se encontra no mercado. Para isso, o processo foi definido seguindo a Norma ISO/IEC 12207 e a prática DevOps.

6 de novembro de 2017

UTFPR

Lista de figuras

Lista de tabelas

Sumário

1 Escopo

Este documento tem por objetivo definir a utilização da Baseline da aplicação Klassic da empresa Software Supimpa Tecnologia (2ST) Através deste é possível o desenvolvedor se iterar sobre a aplicação e todas as tecnologias, ferramentas, códigos e demais necessário para utilizar e continuar o desenvolvimento da aplicação e suas novas versões. Neste documento também é abordado o processo de instalação do ambiente de desenvolvimento, assim caso necessite substituir e/ou adicionar outro equipamento sem ter problemas com este quesito.

2 Tecnologias e ferramentas utilizadas

1. Linguagem Java
2. NetBeans 8
3. PostgreSQL
4. Java Persistence API (JPA)
5. Astah

3 Instalação

O projeto utiliza algumas tecnologias especificas (citadas anteriormente) para que possa funcionar. Além disso, faz necessários o download da Baseline direto do repositório e algumas configurações para que o mesmo inicialize. Abaixo será descrito o passo a passo para que o projeto seja iniciado em um novo computador, este com sistema operacional Linux, Windows ou MacOS.

Acesse o link do repositório do projeto, este hospedado no GitHub atualmente, onde nele você tem acesso a toda a documentação do sistema.

https://github.com/viniciusaugutis/ProdutoGerencia

Após o acesso, faça o download o projeto para sua máquina. Em seguida instale o Java (linguagem de programação utilizada). Segue o link para download: $https : //www.java.com/pt_BR/download/$

Também é preciso que se instale o PostgreSQL que é o banco de dados relacional utilizado na aplicação. Segue o link para download: https://www.postgresql.org/download/

Também é preciso instalar a IDE utilizada para desenvolver o sistema, que no caso é o NetBeans 8. Segue o link para download: https://netbeans.org/downloads/

4 Versionamento

O versionamento da aplicação é essencial para um projeto que tende a crescer a curto ou longo prazo. Ele registra alterações em um arquivo ou conjunto de arquivos ao longo do tempo para que você possa lembrar versões específicas mais tarde. Para o controle de versionamento da aplicação é utilizada a ferramenta GIT. Cada diretório de trabalho do Git é um repositório com um histórico completo e habilidade total de acompanhamento das revisões, não dependente de acesso a uma rede ou a um servidor

central. Como plataforma de hospedagem de código para controle de versão e colaboração é utilizado o GitHub, que é um dos mais populares na atualidade. Ele inovou ao criar uma aplicação web dinâmica, com uma interface agradável e que conseguiu tornar a contribuição de software através um processo fácil e sociável. O git funciona em vários sistemas operacionais, como Mac, Windows e Linux. A primeira tarefa do desenvolvedor consiste em instalar o git na sua máquina de trabalho. A seguir segue o processo de instalação: https://git-scm.com/book/en/v2/Getting-Started-Installing-Git Abaixo serão explicados alguns comandos básicos e iniciais do GIT para começar a trabalhar com o controle de versionamento.

1. git init: iniciar um repositório git
2. git add nome-do-arquivo-incluindo-extensão: usado para adicionar um arquivo para controle
3. git status: mostra o estado do repositório, sendo quais arquivos estão fora do controle, quais foram modificados e estão esperando por uma descrição de modificação
4. git commit -m "Mensagem do commit": comita arquivos do repositório, onde comitar é salvar uma alteração de código-fonte em um sistema de versionamento.
5. git reset HEAD nome-do-arquivo: volta ao estágio anterior ao commit

Abaixo serão explicados alguns comandos do GIT que o desenvolver deve saber para começar a trabalhar com a nossa aplicação.

1. git clone url-do-projeto: faz um clone do nosso repositório remoto da aplicação para sua máquina.
2. git pull: faz uma sincronização com os arquivos mais atualizados do repositório remoto.
3. git push origin nomeBranch: envia modificações da sua máquina local para o repositório remoto
4. git checkout -b nome-do-branch: cria um branch no projeto. Os branchs são ramificações da árvore central do seu projeto, onde eu posso criar vários branchs e após a realização da tarefa eu posso juntar ele novamente ao branch principal.

No link a seguir está a documentação completa do git e que se faz necessário o desenvolvedor aprender: https://git-scm.com/docs/git Para desenvolvimento da nossa aplicação o processo consiste através de pulls-requests. Um pullrequest permite que você informe as mudanças que você enviou para um repositório no GitHub. Uma vez que um pedido é aberto, você pode discutir e rever as possíveis mudanças com os colaboradores, para depois disso o revisor aceitar ou não o seu pedido. Para isso você pode usar o comando git request-pull 1.0.0 url-projeto master. O número do release varia de acordo com o que foi feito pelo desenvolvedor. Se foi consertado algum bug o último número deve ser incrementado (1.0.1). Se foi adicionada uma nova funcionalidade o segudo número deve ser incrementado (1.1.0). Se foi feita uma mudança que alterou a compatibilidade com a versão anterior da baseline, o primeiro número deve ser incrementado (2.0.0)

5 Padrões

Existem dois padrões de escrita:

Snake Case: Define que as palavras são separadas com um caractere de sublinhado e sem espaços, com a letra inicial de cada elemento geralmente em minúsculas dentro do composto. Exemplo: foo_bar.

Camel Case: Define que as palavras compostas ou frase devem ser iniciadas com maiúsculas e unidas sem espaços. Exemplo: FooBar

6 Classes e métodos

Os usuários do processo são todos os envolvidos com o desenvolvimento de software. Dentre eles: desenvolvedor, gerente do projeto, operador de infraestrutura, analistas de sistemas e designers.