FACULDADE SENAC GOIAS

GESTÃO TECNOLOGIA DA INFORMAÇÃO

GERÊNCIA DE MUDANÇAS ECOMMERCE

PROF: Luciéliton

GOIÂNIA

2020

FACULDADE SENAC GOIAS

GESTÃO TECNOLOGIA DA INFORMAÇÃO

VINICIUS LOPES SILVA

LUCAS EDUARDO RODRIGUES

FABRICIO MOREIRA MACHADO

ELSON CRISTINO FARIAS

GOIÂNIA

2020

INTRODUÇÃO

A Gerência de configuração é fundamental para gerenciar alteração de uma funcionalidade ou tarefa de um software , onde engloba processos dependendo

de situações um pouco complexas para que tenha a fim uma boa funcionalidade para o cliente onde pode ter melhoria em processos.

A gerência de configuração de software pode ser entendida como uma disciplina que permita manter a evolução dos produtos de software sobre controle, e, dessa forma , contribuir para satisfazer as restrições de qualidade e prazo . esta disciplina passou a ser considerada como tal a partir da chamada “crise de software” , quando se contatou que o processo de desenvolvimento de software estava incompleto , que seria necessário analisar outros aspectos para melhorar a qualidade dos softwares produzidos. Ao longo dos anos a gerência de configuração tem evoluído , e atualmente pode ser vista com três grandes objetivos:

1. Gerenciar componentes em repositório, que envolvem gerenciamento de versões a modelagem de produtos e o gerenciamento de objetos complexos.

2. Ajudar os engenheiros em suas atividades rotineiras orientando-os ao acesso correto no repositório, procedimentos de atualização de itens de configuração e controle de seus espaços de trabalho.

3. Fornece suporte e controle ao processo de desenvolvimento ,através do controle de mudanças de software ao longo tempo.

Software: E-commerce de pedidos eletrônicos

Itens De configuração:

1. **Especificação do Sistema**

   Software de venda de produtos eletrônicos o e-mail é enviado com especificação da compra
2. **Plano de Projeto de Software**

   Vamos usar um serviço do recurso Spring onde podemos configurar o envio de e-mail para o usuário diretório de configuração (`src/main/resources/application-dev.properties`)
3. **Especificação de Requisitos do Software**

   O sistema irá coletar os dados de uma compra e após o pagamento irá receber o um e-mail com confirmando a compra em um formato em HTML , para qualquer e-mail. para comprovar a compra de um usuário.

4. Especificação do Projeto

a) Descrição do Projeto de Dados

O sistema irá gerar um objeto onde tem a especificação para coletar id , nome e valor e valor total da compra no documento.

b) Descrição do Projeto Arquitetura

- O Usuário confirma o pedido.
- O sistema comunica o servidor SMTP do golpe (O onde o e-mail do administrador -e cadastrado)
- A requisição do servidor é enviada para o cliente para qualquer e-mail;

d) Descrições do Projeto de Interface

O cliente poderá realizar o serviço em qualquer local desde que tenha acesso a internet

e) Descrições de Objetos (se forem usadas técnicas orientadas a objetos)

- No modelo orientado ao Objetos foi implementado o Polimorfismo
- foi implementado toString para ItemPedido e Pedido
- foi ajustado na operação de insert em PedidoService: o Instanciar os objetos relacionados (Cliente e Produto) a partir do banco de dados ao Instanciar a data do pedido com base na data do sistema
- Criar a interface EmailService (padrão Strategy)
- Criar a classe abstrata AbstractEmailService ao Criar método prepareSimpleMailMessageFromPedido o Sobrescrever o método sendOrderConfirmationEmail (padrão Template Method)

5. Planos, Procedimentos, Casos de Testes e Resultados Registrados

Foi implementado o MockEmailService • Em TestConfig, criar um método @Bean EmailService que retorna uma instância de MockEmailService para teste de envio do e-mail no console spring

6. Programa Executável e Módulos Interligados

Remetente e destinatário default no application.properties

10. Descrição do Banco de Dados

12. Documentos de Manutenção

a) Relatórios de problemas de software

O usuário realizava compra no portal e não recebia nenhum comprovante de compra.

b) Solicitações de manutenção

após uma reclamação o cliente solicitou para gera um envio automático por e-mail para o usuário que realizou a compra.

c) Pedidos de mudança

14. Ferramentas de produção de software (editores, compiladores, CASE, etc.)

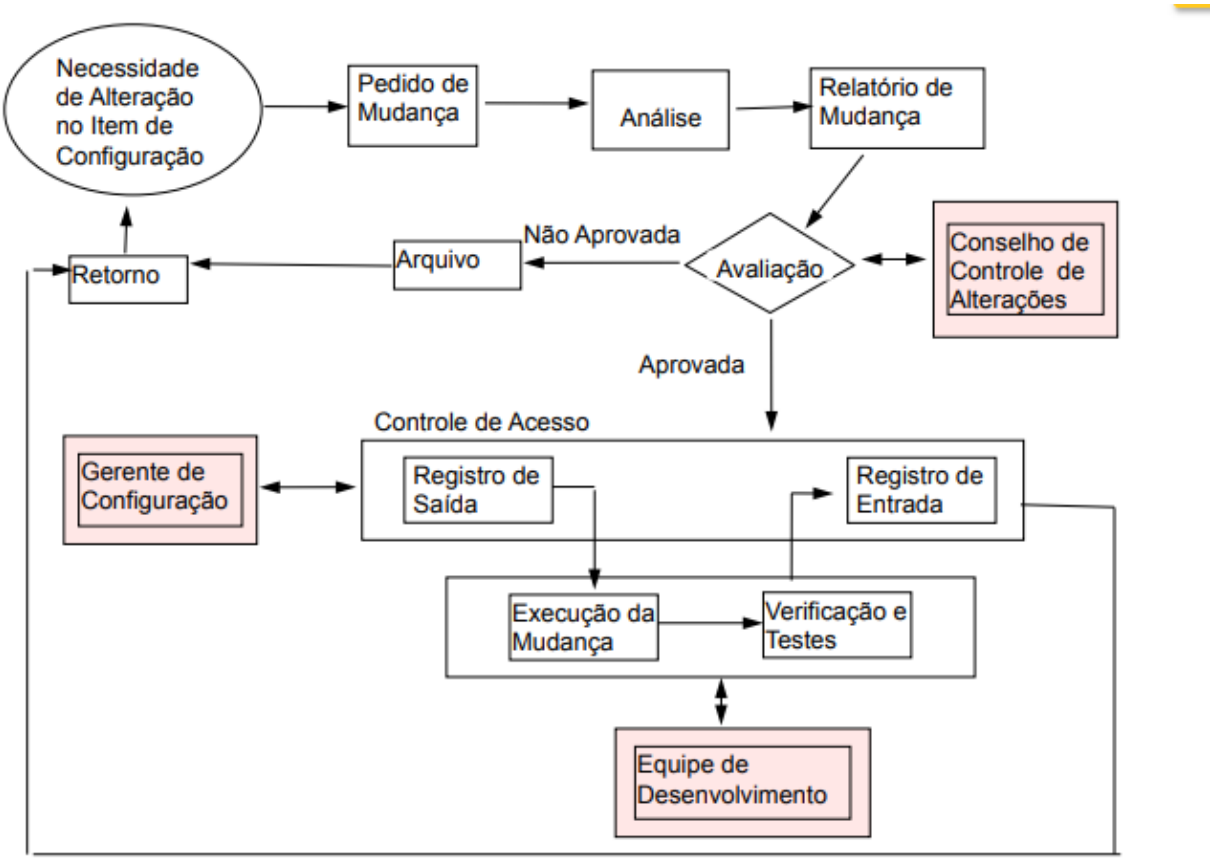
- compilador JAVA
- Spring Tool Suíte 4
- MySQL xammp

Exemplo de um esquema de identificação dos itens de configuração

TIPO	Projeto	Tipo	Nome	Versão	Nome completo
Especificação do Sistema	AA	ES		1.1	AAES v1.1
Plano de Projeto	AA	PP		1.2	AAPP v1.1
Especificação de Requisitos do Software	AA			1.1	
Especificação de Projeto		ER		1.1	AAER v1.1
Programa Fonte	AA	EP	PRIN	1.1	AAPFPrin v1.1
Programa Fonte (sub-rotinas)	AA	PF	EC	1.1	AAPFECv1.1
Plano e Casos de Testes	AA	PF		1.1	AAPF1.1
Nova versão das sub-rotinas	AA	TT	BT	1.1	AATTBT1.1

Controle de Mudanças

Durante o processo de desenvolvimento de software, mudanças descontroladas podem levar rapidamente ao caos. Assim, deve ser instituído na organização um processo que combine procedimentos humanos e ferramentas automatizadas para proporcionar um mecanismo de controle das mudanças.



Procedimentos formais de organização e de controle das mudanças no sistema permitem que:

1. os pedidos de alteração possam ser considerados em conjunto com outros pedidos.

2. os pedidos similares possam ser agrupados.

CONTROLE DE VERSÂO (GIT)

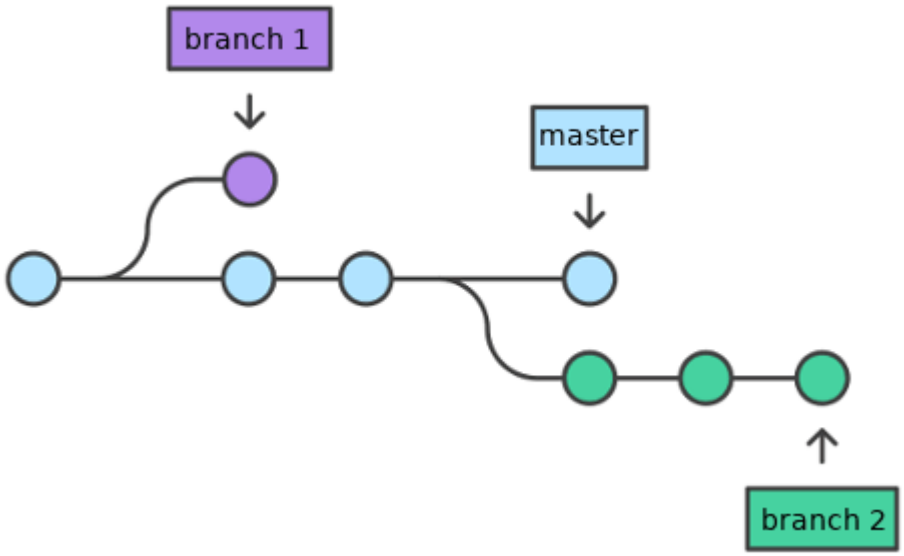
Branches

Depois de liberarmos o código de uma aplicação para o cliente, continuamos melhorando funcionalidades existentes e criando novas funcionalidades, sempre comitando essas alterações. Mas, e se houver um bug que precisa ser corrigido imediatamente? O código com a correção do bug seria liberado junto com funcionalidades pela metade, que ainda estavam em desenvolvimento.

Para resolver esse problema, a maioria dos sistemas de controle de versão tem **branches**: linhas independentes de desenvolvimento nas quais podemos trabalhar livremente, versionando quando quisermos, sem atrapalhar outras mudanças no código.

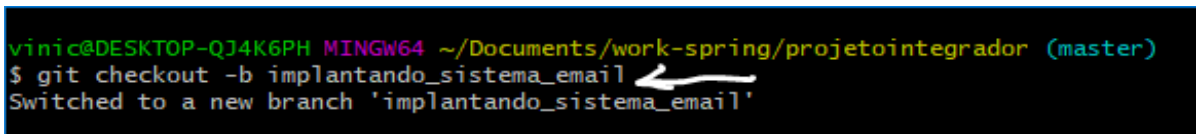
Em projetos que usam Git, podemos ter tanto branches locais, presentes apenas na máquina do programador, quanto branches remotas, que apontam para outras máquinas. Por padrão, a branch principal é chamada **master**, tanto no repositório local quanto no remoto. Idealmente, a master será uma branch estável, isto é, o código nessa branch estará testado e pronto para ser entregue.

Exemplo:

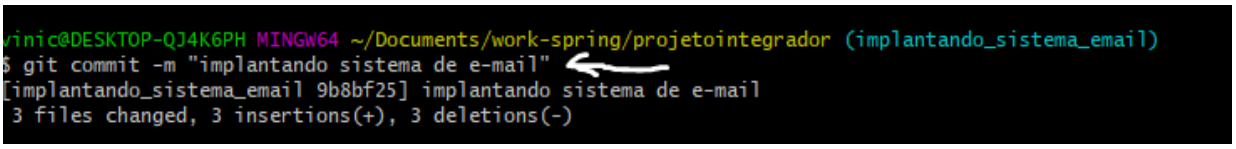


Branches de sistema de e-mail

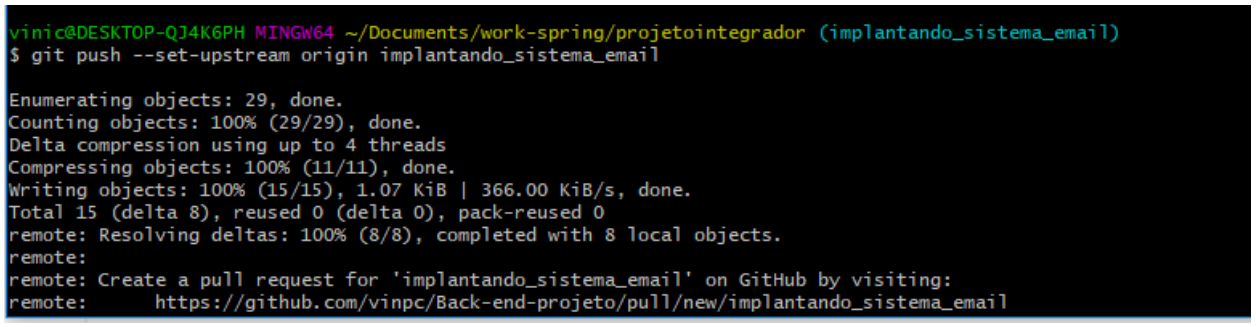
1- Para começar uma ramificação tem que digitar o git Checkout -b **nome_do branche**



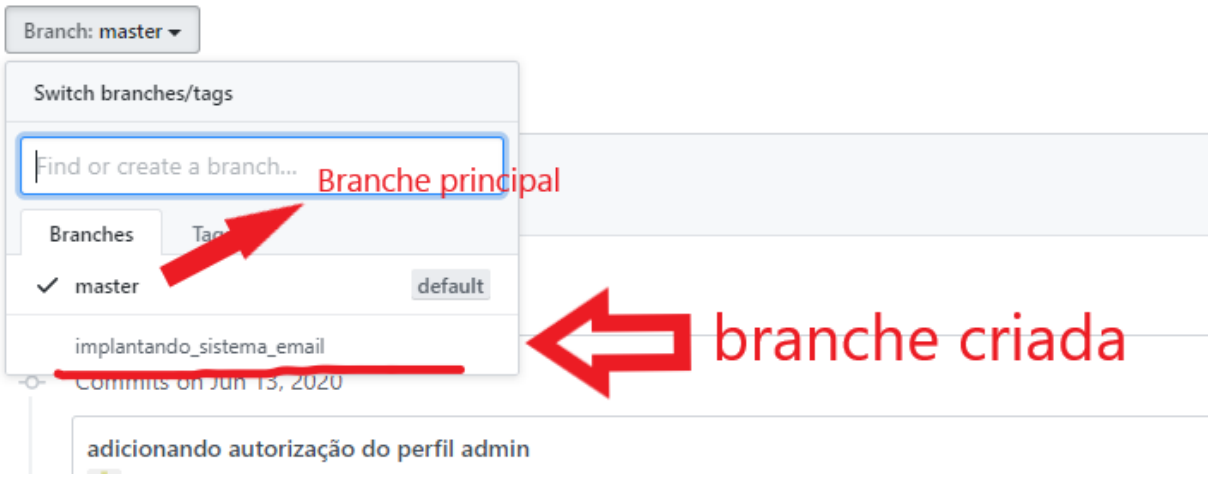
2- Depois faça alteração em seu código e dê um git commit – m e descrvendendo o que foi alterado o que foi realizado : **implantando um sistema de e-mail**:



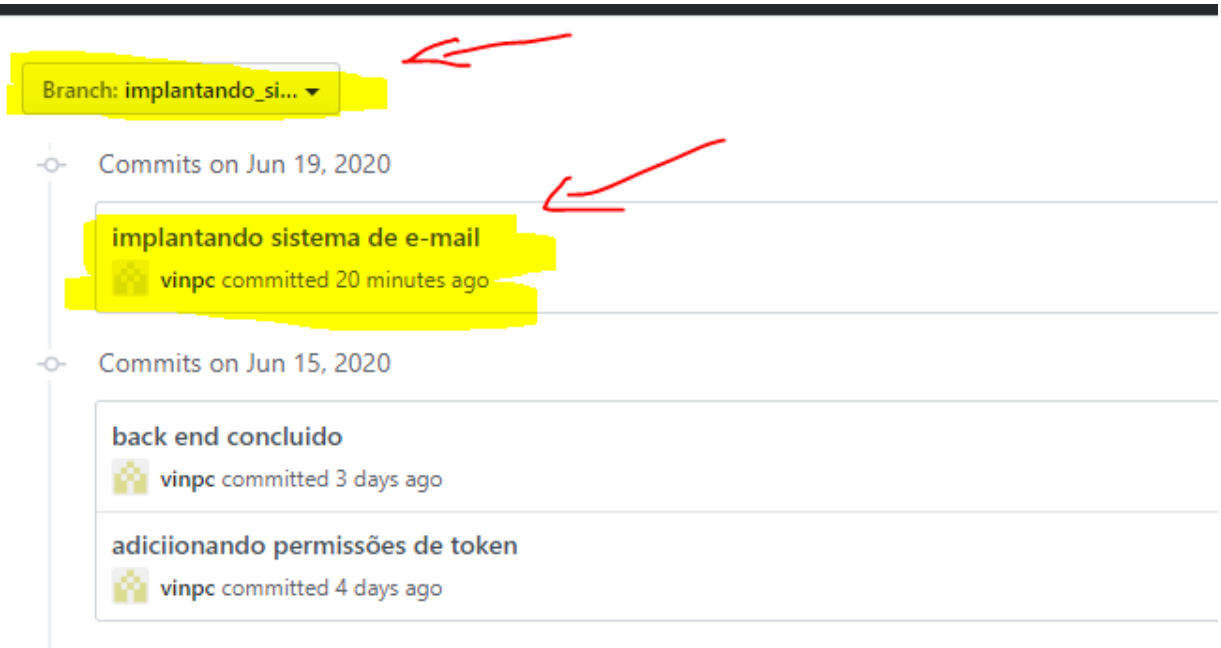
3- Após o commit tem que dar um **git push –set -upstream origin implantando sistema de e-mail**



4. Após o git push realizado poderá ver os branches realizados na imagem abaixo.



5- Na imagem abaixo o branche com o codo implementado.



6- Depois de finalizar poderá juntar junco com o código principal essa ferramenta facilita a implementação e serviços para outras áreas

Link do git realizado : https://github.com/vinpc/Back-end-projeto/commits/implantando_sistema_email

,

Auditoria de Configuração

Auditoria funcional:

A auditoria funcional preocupa-se com aspectos internos dos arquivos, compreendendo uma verificação técnica formal nos itens de configuração. l Essa verificação é uma atividade de controle de qualidade que tenta descobrir omissões ou erros na configuração, que degradam os padrões de construção do software.

A auditoria física

complementa a auditoria funcional, determinando características não consideradas durante a revisão

Relato da Situação

O objetivo do Relato da Situação é relatar a todas as pessoas envolvidas no desenvolvimento e na manutenção do software. As seguintes informações sobre as alterações na configuração de software: – O que aconteceu? – Quem o fez? – Quando aconteceu? – O que mais será afetado?,

Controle de Subcontratados e Fornecedores

Para itens subcontratados deve-se descrever:

a) Os requisitos de gerenciamento de configuração de software a serem satisfeitos pelo subcontratado

b) Como será feito o monitoramento sobre o subcontratado

c) Como o código, documentação e dados externos serão testados, aceitos e adicionados ao projeto

d) Como serão tratadas as questões de propriedade do código produzido, como direitos autorais e royalties.

Ferramentas GCS:

https://www.easyredmine.com/redmine-org/?utm_campaign=bra-local-all-group-0002&gclid=Cj0KCQjwoaz3BRDnARIsAF1RfLc_bafXwJekFrUMXgN6p2EwhGhBHZvBHGCHBgj4AYJDOEFGMxmiyHAaAsBzEALw_wcB

Referências Biliográficas:

- https://edisciplinas.usp.br/pluginfile.php/3351604/mod_resource/content/1/Aula_GCS.pdf

- https://repositorio.ufpe.br/bitstream/123456789/2295/1/arquivo2915_1.pdf