Resposta GCM

1) Na gestão de mudanças, quais são as características dos tipos de mudança

abaixo?

1. Gestão de mudanças:
   1. Mudança padrão
      1. é uma mudança que é pré-autorizada pelo Gerenciamento de Mudanças e que se tornou rotineira, já tendo um script de procedimento para execução. Por este motivo, geralmente o fluxo para execução desta alteração é mais ágil.
   2. Mudança normal
      1. é uma mudança para a qual não existe um script já pronto, e precisa passar pelo fluxo mais extenso para ser autorizada e planejada antes de sua execução.
   3. Muda emergencial
      1. é aquela que precisa ser implementada rapidamente para resolver falhas (incidentes). Neste caso, nem sempre será possível realizar todos os testes. Este tipo de mudança é tratado pelo Comitê Consultivo de Mudanças

Emergenciais.

2) Quem são os participantes de um comitê consultivo de mudanças?

1. Os participantes do comitê consultivo de mudanças são:
   1. Fornecedores, Gerente de Problemas Gerente de Nível de Serviço, Equipe de relações com Cliente, etc..

3) Dê 3 exemplos de regras usuais para que os desenvolvedores realizem mudanças nas funcionalidades de um software

1. Criar um branch separado, criar um pull request para o dono do projeto, estabelecer datas para as modificações

4) Cite 4 exemplos de ferramentas de controle de versão de código-fonte.

1. Os exemplos são:
   1. GitHub, SVN Subversion, Visual Source Safe, ClearCase.

5) O que é integração contínua?

1. Integração continua é uma prática de desenvolvimento de software onde membros de um time integram seu trabalho frequentemente. Cada integração é verificada por um build automatizado (incluindo a execução de testes) para detectar erros o mais cedo possível. Muitas equipes acreditam que esta abordagem reduz problemas de integração e permite que o software seja desenvolvido de forma rápida e coesa.

6) Quais são os benefícios da integração contínua?

1. As vantagens são que cada integração é verificada por um build automatizado (incluindo a execução de testes) para detectar erros o mais cedo possível. Muitas equipes acreditam que esta abordagem reduz problemas de integração e permite que o software seja desenvolvido de forma rápida e coesa e mais interativa.

7) Cite 2 exemplos de ferramentas de software que podem dar apoio à gestão de mudanças.

1. Os exemplos são :
   1. Travis CI, bamboo, go, integrity, Jenkins, etc.

8) Cite 2 exemplos de ferramentas de software para a realização de integração contínua.

1. Os exemplos são:
   1. Travis ci, bamboo, go.

9) No que consiste fazer o gerenciamento de releases?

1. Consiste no empacotamento de um sistema para promovê-lo de desenvolvimento para QA (Quality Assurance) e, posteriormente, para produção.

10) Cite 3 princípios do gerenciamento de releases.

1. Os 3 princípios são:
   1. Releases devem ser identificados por um identificador (ID) de versão imutável.
   2. Releases devem ser empacotados com todas as suas dependências.
   3. O gerenciamento de releases deve ser rápido e confiável para facilitar o desenvolvimento iterativo.
   4. O empacotamento de releases deve ser automatizado e desenhado para evitar erros humanos.
   5. O gerenciamento de releases deve ser uma fonte de informação no status de todos os releases, idealmente através de um dashboard.

11) Explique como é feita a identificação de um release utilizando o

versionamento semântico, e qual é o significado de cada parte desta

identificação.

1. Uma versão é identificada por uma sequência de 3 números MAJOR, MINOR, PATCH.
   1. MAJOR deve ser alterado se somente quando as mudanças tornam a API incompatível com as versão anteriores.
   2. MINOR deve ser alterado quando é adicionado funções novas mas mantendo a compatibilidade com versão anteriores
   3. PATCH deve ser alterado quando as mudanças são para arrumar bugs mantendo a compatibilidade com versões anteriores.

12) Dê um exemplo de identificação de um release utilizando o versionamento semântico.

1. Software 1.0.1

13) Cite um benefício obtido com a utilização de uma ferramenta como o Codenvy.

1. Um benefício é a portabilidade uma vez que você pode programar em qualquer lugar e ter a mesma interface e API em todos os lugares.

14) Cite um benefício obtido com a utilização de uma ferramenta como o Heroku.

1. Um benefício do heroku e que você pode testar seus projetos de maneira rápida e ágil.