Nome: Paulo Roberto Xavier da Silva

Turma: 23-1N

Trabalho de pesquisa sobre conceitos no Teste de Software

1. Conceitos de teste

1.1 Caso de Teste

Os casos de teste são como roteiros que definem as etapas de interação com o software, eles exibem o passo a passo detalhado de como interagir com o software, abordando de uma maneira detalhada o que se espera como resultado e como realizar os testes. Nele, podem ser registradas as seguintes informações: Objetivo do teste, condições prévias para conseguir executar o teste, as etapas do teste, os dados de entrada, o resultado esperado, observações, etc. Cada caso de teste irá testar determinada funcionalidade específica na nossa aplicação.

1.2 Plano de Teste

São documentos mais abrangentes do que os casos de teste, pois eles descrevem de uma maneira mais ampla as estratégias para a execução desses testes de software. Eles descrevem os recursos que vamos precisar, os cronogramas de execução dos testes, os possíveis riscos ou formas de mitigar tais riscos, os cenários de tais testes, os ambientes de teste, entre outros. Funcionam como um planejamento mais amplo, não se preocupando em testar alguma funcionalidade específica da nossa aplicação.
Os casos de teste irão testar determinada funcionalidade da nossa aplicação, o plano de teste irá planejar todo o contexto e o ambiente para execução de tal teste.

1.3 Defeito

Um defeito ou bug, como também é chamado, refere-se a uma falha no código que leva a um comportamento indesejado. Nesse caso, o código pode conter erros de sintaxe, e por conta disso, pode ocasionar comportamentos indesejados durante a execução do nosso programa. Ou ainda, por eventuais problemas de comunicação ou má compreensão dos requisitos.

1.4 Falha

Uma falha refere-se a uma situação na qual o software não opera conforme o esperado ou não atende aos requisitos do projeto. Nesse caso, não temos necessariamente um erro de código, mas sim requisitos que não são cumpridos no momento da entrega do software, resultando em comportamentos indesejados, ou seja, o software não executa uma determinada funcionalidade de maneira correta e esperada.

1.5 Teste Estático

Nesse tipo de teste não teremos a execução do programa ou sistema que está sendo trabalhado. Esse tipo de teste é realizado antes do código ser compilado e executado. Ele é muito usado como levantamento de requisitos, verificando se os mesmos estão sendo seguidos, além de permitir verificar erros de lógica, sintaxe, etc.

1.6 Teste Dinâmico

Diferentemente do teste estático, nessa abordagem de teste temos a execução do código-fonte do programa, e a observação do seu comportamento durante sua execução. São testes fundamentais para avaliar a funcionalidade do programa, seu desempenho e outras eventuais falhas que os testes estáticos não conseguem atingir efetivamente. Esse tipo de teste é muito próximo da realidade, pois permite simular situações do dia a dia dos usuários desse software.

1.7 Critério de Aceitação

São métricas que devem ser atingidas para que os testes sejam considerados bem-sucedidos. Esses critérios podem variar entre qualidade, desempenho e performance, usabilidade, segurança, entre outros.

2. Níveis de teste

2.1 Teste Unitário

Esse tipo de teste caracteriza-se por ser feito por partes, em vez de olharmos para o código como um todo, testamos cada parte separadamente, em unidades. Cada função, método, classes, entre outros, são testados individualmente, para garantir que cada unidade atenda as expectativas e objetivos. Esse tipo de teste nos permite fazer a correção de problemas logo nas primeiras fases de desenvolvimento, permitindo assim fazer a correção e somente depois, fazer o teste como um todo. Pensando em um sistema bancário, cada operação seria testada individualmente, a fim de garantir que esteja operando dentro das necessidades e do esperado para tal software.

2.2 Teste de Integração

Nesse teste, as etapas que antes eram testadas separadamente, agora são combinadas e testadas em conjuntos. O objetivo principal é testar como essas unidades interagem entre si e verificar possíveis problemas que podem surgir como resultado dessa interação entre unidades. Um exemplo seria fazer um sistema bancário, onde iríamos ter testado previamente as operações de saldo e saque, e agora iriamos juntar ambas para verificar se após o saque o saldo também é atualizado.

2.3 Teste de Sistema

Esse tipo de teste buscará fazer uma validação no sistema como um todo. O objetivo principal é garantir que todo o sistema esteja atendendo aos requisitos funcionais e não

funcionais do projeto. Nessa etapa os conjuntos são unidos e testados coletivamente, buscando fazer uma validação prévia antes de mostrar para o cliente. Pensando num sistema bancário, o sistema seria testado de forma total.

2.4 Teste de Aceitação

Essa é a fase final, o cliente e/ou as partes interessadas farão os testes do projeto final e ver se está dentro do que foi pedido. O objetivo final continua ser suprir as demandas solicitadas pelo cliente e ver o parecer final dele sobre o projeto, além de verificar se o sistema está pronto para ser implantado. É a parte mais importante, já que caso ainda não esteja dentro do que o cliente solicitou será necessário fazer novas alterações. Mantendo o exemplo do sistema bancário, essa seria a etapa final aonde os próprios usuários e públicos-alvo desse sistema iriam fazer os testes.

3. Tipos de teste

3.1 Teste Funcional

É um tipo de teste que visa verificar se o software atende aos requisitos funcionais, ou seja, se ele realiza as ações que foram solicitadas durante a etapa de planejamento de software. Esse tipo de teste garante que todas as interações entre o usuário e o programa estejam acontecendo e de maneira correta e dentro do que foi pedido pelo usuário.

3.2 Teste de Regressão

Esse tipo de teste visa testar se alterações feitas no código não afetaram funcionalidades testadas previamente e que estava em funcionamento. Ele é realizado para garantir que novas mudanças não causem regressões no código, fazendo com que funcionalidades testadas e aprovadas previamente continuem a funcionar conforme o esperado.

3.3 Teste de Desempenho

O teste de desempenho visa avaliar o desempenho do software submetendo o mesmo a diferentes ocasiões, como situações de alto acesso simultâneo, alto volume de carga e estresse, volume elevado de dados etc. O objetivo é identificar possíveis gargalos e assim, otimizar ao máximo o software para que atenda as expectativas em termos de velocidade, desempenho e eficiência.

3.4 Smoke Test

Também conhecido como “Teste de Fumaça”, visa verificar se as funcionalidades principais do sistema continuam operando de maneira funcional, mesmo após alguma mudança significativa no código ou alguma nova versão do software. O termo “fumaça”

transmite a ideia de que caso houver algum problema será possível verificar os primeiros sinais da fumaça indicando um incêndio. Ele é um teste de curta duração, visando verificar se a aplicação está minimamente funcional antes de prosseguir com testes mais abrangentes.

3.5 Teste Exploratório

É um tipo de teste que dá mais liberdade para o testador, onde ele tem liberdade para testar o sistema sem a necessidade de seguir scripts de teste ou casos de testes predefinidos. O testador tem total liberdade para conhecer o sistema e ir adaptando suas estratégias à medida que for conhecendo melhor o sistema e suas funcionalidades principais. Essa flexibilidade garante que o testador se baseie na sua intuição e experiência, permitindo assim uma exploração mais profunda no software.

3.6 Teste de Confirmação

É um teste que visa confirmar se a correção ou algum ajuste específico de um defeito foi eficaz, confirmando assim que o defeito foi solucionado com sucesso. Esse tipo de teste concentra-se especificamente na área afetada pela correção.

3.7 Teste de Estresse

É um tipo de teste que visa avaliar a estabilidade e desempenho de um sistema quando submetido a condições extremas ou além dos limites normais de operação. O sistema é submetido a cargas extremas, e com isso permite determinar a capacidade máxima que o sistema pode suportar, e observar seu comportamento durante longos períodos de atividade. Além disso, os recursos do sistema como CPU, memória, largura de bande de rede, e outros recursos são testados.

3.8 Teste de Carga

Esse tipo de teste submete o sistema a uma carga específica e esperada, visando avaliar o desempenho e como o sistema se comporta em condições normais de operação. Com isso, torna-se possível identificar eventuais gargalos, problemas de desempenho ou falhas que podem ocorrer quando a demande é aumentada. Ao contrário do Teste de Estresse que visa submeter o sistema além dos seus limites, o Teste de Carga visa avaliar o sistema sob uma carga planejada e controlada. Um dos exemplos seria testar um número de usuários acessando o software ao mesmo tempo para avaliar como o sistema lida com essa demanda de usuários fazendo requisições simultâneas.

3.9 Teste de Volume

O teste de volume se caracteriza por avaliar como o sistema se comporta quando submetido a grandes volumes de dados. O objetivo principal é verificar como o sistema lida com o aumento significativo na quantidade de informações, transações ou usuários. Diferentemente do Teste de Carga que se concentra nas cargas simultâneas de usuários ou transações, o Teste de Volume visa monitorar o desempenho do sistema em relação a sua capacidade de processar grandes quantidades de dados.

3.10 Teste de Recuperação

Esse tipo de teste visa avaliar a capacidade de um sistema de lidar com falhas, erros ou eventos inesperados, como problemas de hardware, falta de energia elétrica, ou problemas no software. Essa prática visa garantir que o sistema se mantenha operando mesmo após situações adversas. Também são testados procedimentos de backup e restauração de dados para garantir que os dados possam ser recuperados em casos de perda. É um teste focando na resiliência do sistema, de como o sistema lida com problemas inesperados e de sua capacidade de recuperar-se e continuar operando de maneira correta.

3.11 Teste de Segurança

Também é conhecido como “Teste de Ethical Hacking”, esse tipo de teste visa verificar como o sistema reage contra ataques maliciosos, identificando vulnerabilidades e verificando sua resistência contra eventuais ataques de hackers. O objetivo é verificar se o sistema é seguro, protegido contra ameaças externas e se atende aos requisitos de segurança. Os testadores de segurança, frequentemente chamados de “Ethical Hackers” ou “Hackers Éticos” simulam ataques contra esses sistemas para identificar eventuais lacunas e breches no sistema.