

**Projeto e Implementação de Sistemas**

**Aluno: Marcos Vinicius Melo da Silva**

**Projeto – Sistema para gestão de imagens para instalação de Sistema Operacional e softwares nas máquinas dos laboratórios do DCC**

* **Delimitar o modelo de desenvolvimento de software a ser adotado.**

**Modelo Incremental:** Desenvolvimento em etapas, entregando funcionalidades em incrementos sucessivos.

**Identificação das Necessidades**: Entender os requisitos dos usuários finais e os sistemas operacionais que precisam ser suportados.

**Escopo do Projeto**: Determinar quais aplicativos e sistemas operacionais serão abordados inicialmente.

**Prototipagem**: Desenvolver protótipos iniciais dos scripts de instalação de aplicativos e sistemas operacionais, bem como dos scripts de verificação básica.

**Desenvolvimento Iterativo**: Desenvolver os scripts em incrementos, adicionando funcionalidades e refinando conforme necessário.

**Testes Iterativos**: Realizar testes em cada incremento para garantir que os scripts estejam funcionando conforme o esperado.

**Implantação Inicial**: Implementar os scripts em um ambiente de teste para verificar seu desempenho em um ambiente real.

**Avaliação e Refinamento**: Avaliar regularmente os scripts para identificar áreas de melhoria e realizar ajustes conforme necessário.

**Atualizações e Manutenção**: Manter os scripts atualizados para acompanhar mudanças nos sistemas operacionais e requisitos dos aplicativos.

* **Delimitar padrões de qualidade e de teste de software**

Para garantir a qualidade e o funcionamento adequado dos scripts de instalação de aplicativos e sistemas operacionais, bem como dos scripts de verificação básica em computadores da faculdade, é importante realizar uma variedade de testes em diferentes estágios do desenvolvimento.

Testes a serem Realizados:

**1. Testes Unitários:**

Quando realizar: Após o desenvolvimento de cada função ou componente individual dos scripts.

Por que são importantes: Os testes unitários garantem que cada parte do código funcione conforme o esperado, isoladamente. Isso ajuda a identificar e corrigir problemas específicos de cada função ou componente antes de integrá-los ao sistema como um todo.

Como realizar:

Escrever casos de teste para cada função ou componente, cobrindo diferentes cenários de entrada e saída.

Utilizar estruturas de teste automatizadas, como frameworks de teste unitário (por exemplo, JUnit para Java, pytest para Python), para executar os testes de forma rápida.

Executar os testes regularmente, idealmente após cada alteração no código, para garantir que nenhuma nova alteração introduza regressões.

**2. Testes de Integração:**

Quando realizar: Após a conclusão da primeira versão funcional dos scripts.

Por que são importantes: Os testes de integração garantem que os diferentes componentes dos scripts funcionem corretamente juntos. Isso ajuda a identificar problemas de comunicação ou compatibilidade entre os diferentes módulos.

Como realizar:

Identificar os pontos de integração entre os diferentes componentes dos scripts.

Escrever casos de teste que cubram fluxos de trabalho completos, envolvendo múltiplos componentes.

Executar os testes de integração em um ambiente de teste separado, simulando o ambiente de produção o mais próximo possível.

**3. Testes de Aceitação do Usuário:**

Quando realizar: Antes da implantação dos scripts nos computadores da faculdade.

Por que são importantes: Os testes de aceitação do usuário garantem que os scripts atendam às expectativas e necessidades dos usuários finais. Isso ajuda a identificar problemas de usabilidade ou funcionalidade antes da implantação em larga escala.

Como realizar:

Envolver usuários finais representativos para testar os scripts em um ambiente de teste simulado.

Fornecer instruções claras sobre como usar os scripts e colete feedback dos usuários durante o processo de teste.

Realizar testes em diferentes cenários de uso para garantir que os scripts sejam robustos o suficiente para lidar com uma variedade de situações.

**4. Testes de Desempenho:**

Quando realizar: Após a implementação inicial dos scripts.

Por que são importantes: Os testes de desempenho garantem que os scripts possam lidar com uma carga de trabalho esperada sem degradar significativamente o desempenho do sistema. Isso é crucial para garantir uma experiência de usuário satisfatória e evitar problemas de tempo de inatividade.

Como realizar:

Identificar cenários de uso representativos e estresse que simulem a carga esperada nos scripts.

Utilizar ferramentas de teste de carga, como JMeter ou Gatling, para simular múltiplos usuários acessando os scripts simultaneamente.

Analisar métricas como tempo de resposta, uso de recursos do sistema e taxa de erros para identificar possíveis gargalos de desempenho e otimizar os scripts, se necessário.

**5. Testes de Segurança:**

Quando realizar: Durante todo o ciclo de vida do projeto.

Por que são importantes: Os testes de segurança garantem que os scripts não contenham vulnerabilidades que possam ser exploradas por invasores para comprometer a segurança dos sistemas. Isso é essencial para proteger os dados e a integridade do sistema.

Como realizar:

Realizar análises de vulnerabilidade estática e dinâmica nos scripts para identificar possíveis pontos fracos de segurança.

Realizar testes de penetração para avaliar a resistência dos scripts a ataques comuns, como injeção de código, cross-site scripting (XSS) e falsificação de solicitação entre sites (CSRF).

Corrigir quaisquer vulnerabilidades identificadas e aplicar práticas de segurança recomendadas, como autenticação e autorização adequadas, criptografia de dados e validação de entrada, para mitigar futuros riscos de segurança.

**6. Testes de Manutenção:**

Quando realizar: Após cada atualização ou modificação nos scripts.

Por que são importantes: Os testes de manutenção garantem que as alterações nos scripts não introduzam novos problemas ou regressem a funcionalidade existente. Isso é essencial para garantir que os scripts permaneçam estáveis e confiáveis ao longo do tempo.

Como realizar:

Sempre executar os testes unitários e de integração após cada alteração nos scripts para garantir que todas as funcionalidades permaneçam intactas.

Utilizar testes de regressão para verificar se as alterações não afetaram negativamente o comportamento existente dos scripts.

**Boas Práticas no Projeto:**

**Utilizar Controle de Versão:** Utilizar sistemas de controle de versão como Git para gerenciar o código-fonte dos scripts e facilitar a colaboração entre os membros da equipe.

**Documentação Adequada:** Manter uma documentação clara e atualizada dos scripts, incluindo instruções de uso, requisitos de sistema e notas de versão.

**Revisão de Código:** Realizar revisões de código regularmente para identificar e corrigir problemas de qualidade e garantir a conformidade com os padrões estabelecidos.

**Automação de Testes:** Automatizar o máximo possível de testes para garantir uma cobertura abrangente e rápida identificação de problemas.

**Implantação Controlada:** Implementar os scripts em um ambiente de produção controlado e realizar testes adicionais antes da implantação em larga escala.

**Monitoramento Contínuo:** Implementar mecanismos de monitoramento para os scripts em produção, a fim de detectar e resolver problemas rapidamente.

* **Planejamento e estratégia de levantamento de requisitos & pessoas que serão entrevistadas / coletar dados**

**Clientes: Professor Marcelo, Monitor (do lab, irei descobrir quem é)**

* Realizar entrevistas com as partes interessadas para entender suas necessidades e expectativas em relação ao sistema.
* Quais aplicativos e sistemas operacionais devem ser suportados pelos scripts?

R: Linux e Windows.

* Quais verificações básicas devem ser realizadas nos computadores da faculdade?

R: Verificação de atualizações, verificação de malware, verificações básicas de uso (temperatura, memória usada…).

* Quais são os principais problemas ou desafios enfrentados atualmente na instalação de aplicativos e sistemas operacionais?

R: A Internet e as configurações do pc.

* Quais são os requisitos de segurança que devem ser considerados nos scripts?

R: Autenticação

* Como os scripts poderão ser distribuídos e implantados nos computadores da faculdade?

R: Basicamente teria que ir de computador em computador implementar o projeto.

* **Organizar informações sobre os requisitos de software de modo a aumentar a probabilidade de que o trabalho de desenvolvimento atinja os objetivos necessários.**

| **Tipo de Requisito** | **Descrição** | **Exemplo** |
| --- | --- | --- |
| Requisitos Funcionais |  |  |
| Instalação automatizada de aplicativos | Os scripts devem ser capazes de instalar automaticamente uma lista de aplicativos específicos em cada computador, sem intervenção manual. | Os aplicativos de produtividade, como Microsoft Office e Adobe Acrobat, devem ser instalados em todas as máquinas da faculdade automaticamente. |
| Verificação de integridade do sistema | Os scripts devem verificar a integridade do sistema, incluindo a presença de arquivos críticos e a integridade do registro do sistema. | O script deve verificar se todos os arquivos do sistema operacional estão presentes e não foram modificados. |
| Suporte a personalização | Os scripts devem permitir personalização para atender às necessidades específicas de diferentes departamentos ou usuários. | Os administradores devem poder personalizar a lista de aplicativos a serem instalados em diferentes departamentos da faculdade. |
| Requisitos Não Funcionais |  |  |
| Suporte a múltiplos sistemas operacionais | Os scripts devem ser capazes de serem executados em diferentes sistemas operacionais, incluindo Windows, macOS e Linux. | Os scripts devem funcionar corretamente em todas as versões do Windows a partir do Windows 7, macOS 10.12 (Sierra) e Ubuntu 18.04 LTS. |
| Segurança robusta | Os scripts devem seguir as melhores práticas de segurança, como autenticação e autorização adequadas, para proteger contra acesso não autorizado. | Os scripts devem exigir autenticação de administrador para executar operações que afetam o sistema operacional. |
| Eficiência de desempenho | Os scripts devem ser otimizados para garantir um desempenho eficiente, minimizando o tempo de instalação e verificação. | O tempo total para instalação de todos os aplicativos não deve exceder 3 horas. |
| Regras de Negócio |  |  |
| Priorização de aplicativos | Os aplicativos devem ser instalados de acordo com a priorização estabelecida pela diretoria da faculdade. | Os aplicativos essenciais para o funcionamento acadêmico devem ser instalados antes dos aplicativos de uso geral. |
| Restrições de licença | Os scripts devem respeitar as licenças de software e garantir que apenas os aplicativos licenciados sejam instalados. | Os aplicativos comerciais só podem ser instalados em máquinas autorizadas e com licenças válidas. |