# Integração contínua e entrega contínua

Introdução à integração contínua e à entrega contínua

Aula 3

Professor Gustavo Dias

# Relação entre integração contínua e entrega contínua de software

A integração contínua (CI) e a entrega contínua (CD) são, frequentemente, usadas em conjunto, mas representam estágios distintos no ciclo de vida do desenvolvimento de software.

# Exemplo

Imagine que você está construindo um aplicativo de entrega de alimentos e dois desenvolvedores estão trabalhando (CI em ação):

- Maria desenvolve uma nova funcionalidade que permite aos usuários **rastrear seus pedidos** em tempo real.
- João faz **alterações no sistema de pagamento** para suportar uma nova forma de pagamento.
- Ambos fazem **commit de suas mudanças** no mesmo dia.

# Exemplo

**Integração contínua:**

1. O sistema de CI **detecta os novos commits** e inicia, automaticamente, o processo de compilação e teste.

2. Os testes mostram que a nova **funcionalidade de rastreamento e o sistema de pagamento atualizado** estão funcionando perfeitamente juntos sem conflitos ou erros.

3. O código agora está integrado com sucesso.

# Exemplo

**Integração contínua:**

4. O sistema de CD entra em ação e **garante que esse código integrado** esteja pronto para ser lançado.

5. Uma vez que o código é validado em todos os ambientes necessários, ele **está pronto para ser lançado em produção**.

Vídeo

# PIPELINE


Fonte: https://www.youtube.com/watch?v=AZtTd3pFVTY

# Etapas do deploy de um pipeline automatizado

**1**

- Controle de versão.
- Integração contínua (CI).
- Construção (*Build*).

**2**

- Testes automatizados.
- Teste de segurança.
- Teste de performance.

**3**

- Ambiente de staging.
- Revisão e aprovação.

**4**

- Implantação em produção.
- Monitoramento e feedback.

# Desafios processuais

**Complexidade da configuração:**

Configurar um pipeline automatizado pode ser complexo, especialmente em sistemas maiores ou em ambientes com muitas dependências.

**Falhas inesperadas:**

Mesmo com testes abrangentes, problemas inesperados podem surgir, especialmente, quando o **software é implantado em ambientes de produção com variações sutis.**

**Gerenciamento de dependências:**

Assegurar que todas as dependências de software (bibliotecas, serviços etc.) estejam **corretamente versionadas e compatíveis** pode ser um desafio, especialmente, em ambientes complexos.

Vamos fazer um ***quiz***

# Teste seus conhecimento

No contexto de *deploy*, qual é o papel de um pipeline no processo de desenvolvimento e entrega de software, especialmente em relação aos testes de performance?

Vamos fazer um ***quiz***

Um pipeline é usado exclusivamente para o *deploy* automático de aplicações em ambientes de produção, sem realizar nenhum teste.

O pipeline coordena a execução sequencial de scripts de banco de dados, sem incluir testes de aplicação.

Um pipeline serve para automatizar os testes de performance, garantindo que o software atenda aos padrões de desempenho antes de ser lançado.

Pipelines são usados apenas para integração contínua, verificando a integridade do código, mas não incluem testes de desempenho.

Vamos fazer um ***quiz***

# Feedback da atividade

**RESPOSTA ERRADA!** Os testes unitários objetivam verificar unidades individuais de código e não as interações entre elas.

**RESPOSTA ERRADA!** Enquanto pipelines podem coordenar scripts de banco de dados, também incluem outras tarefas como testes de aplicação, e não se limitam a apenas uma atividade.

**RESPOSTA CORRETA!** Em um pipeline de CI/CD, uma das etapas críticas é a automação de testes de performance, que assegura que o software não apenas funciona corretamente, mas também atende aos critérios de desempenho estabelecidos antes de ser promovido para o próximo estágio, como a entrega ou o *deploy*.

**RESPOSTA ERRADA!** Pipelines não são limitados à integração contínua; eles também podem e devem incluir automação de testes de performance como parte da garantia da qualidade do software.

Vamos fazer um ***quiz***

# Teste seus conhecimentos

A equipe de desenvolvimento de uma loja on-line está implementando a entrega contínua. Eles querem ter certeza de que, quando novas funcionalidades forem adicionadas, elas não quebrarão funcionalidades existentes. Qual tipo de teste eles devem enfatizar?

- **Teste de aceitação.**
- **Teste unitário.**
- **Teste de integração.**
- **Teste manual.**

Registro

Vamos fazer um ***quiz***

# Feedback da atividade

✗ **Teste de aceitação.**

**RESPOSTA ERRADA!** Embora útil para validar se o sistema atende aos critérios de aceitação, não se concentra especificamente nas interações entre diferentes partes do sistema.

✗ **Teste unitário.**

**RESPOSTA ERRADA!** Os testes unitários objetivam verificar unidades individuais de código e não as interações entre elas.

✓ **Teste de integração.**

**RESPOSTA CORRETA!** Os testes de integração garantem que diferentes partes (ou unidades) do software trabalhem juntas corretamente, o que é essencial quando novas funcionalidades são adicionadas.

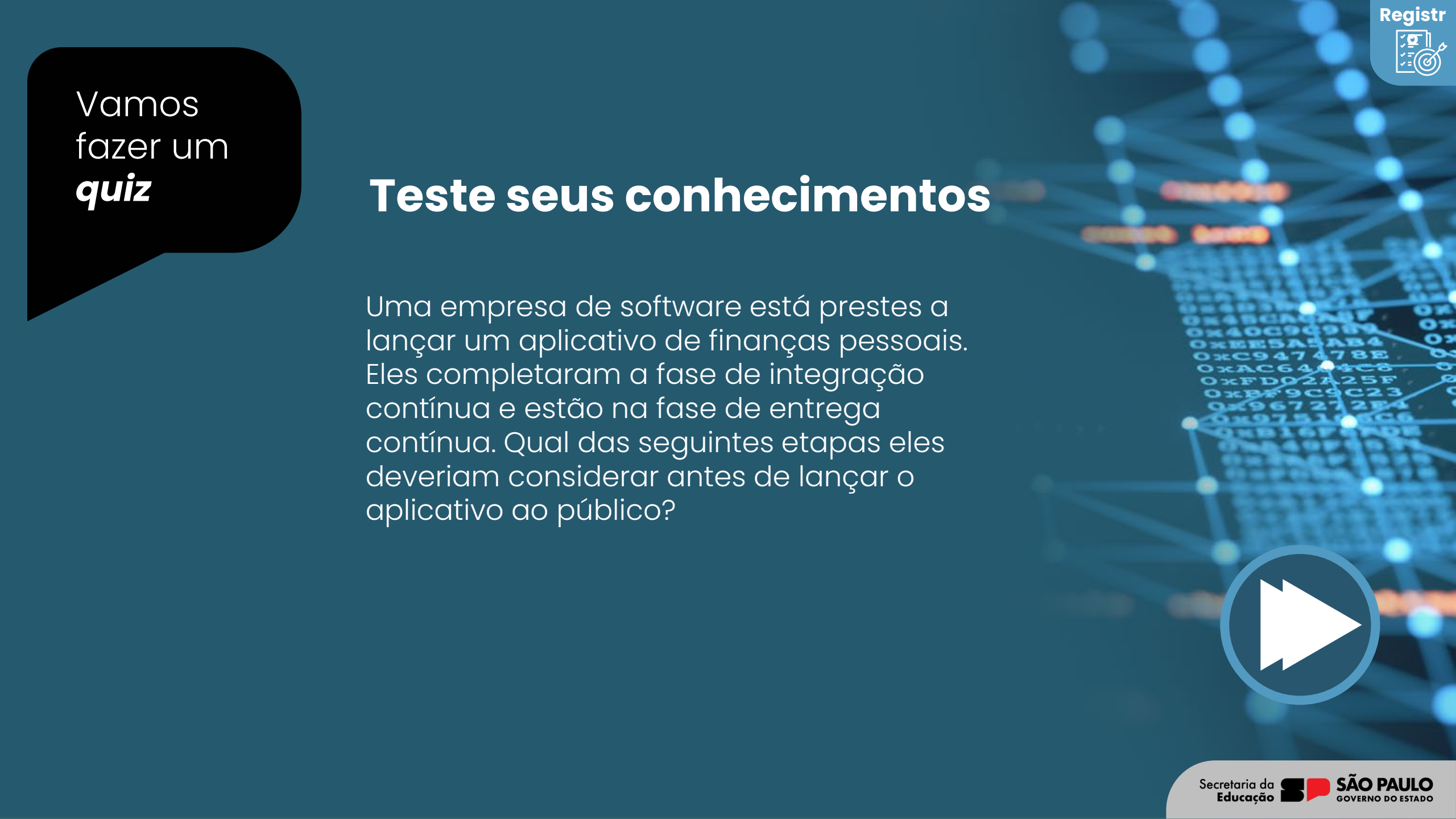
✗ **Teste manual.**

**RESPOSTA ERRADA!** Embora os testes manuais possam identificar problemas, eles não são tão eficientes ou repetíveis quanto os testes automatizados de integração.

Vamos fazer um ***quiz***

# Teste seus conhecimentos

Uma empresa de software está prestes a lançar um aplicativo de finanças pessoais. Eles completaram a fase de integração contínua e estão na fase de entrega contínua. Qual das seguintes etapas eles deveriam considerar antes de lançar o aplicativo ao público?



Registr

Secretaria da Educação SÃO PAULO GOVERNO DO ESTADO

Vamos fazer um ***quiz***

Realizar uma revisão de código.

Compilar o código.

Testar o aplicativo em um ambiente de produção simulado.

Escrever a documentação do código.

Vamos fazer um ***quiz***

# Feedback da atividade

✗ **Alternativa 1**

**RESPOSTA ERRADA!** Revisões de código são importantes e geralmente ocorrem durante a fase de integração contínua.

✗ **Alternativa 2**

**RESPOSTA ERRADA!** A compilação é uma das primeiras etapas e ocorre durante a fase de integração contínua.

✓ **Alternativa 3**

**RESPOSTA CORRETA!** Testar em um ambiente que simula a produção é crucial durante a entrega contínua para garantir que o software funcionará corretamente quando lançado ao público.

✗ **Alternativa 4**

**RESPOSTA ERRADA!** Embora a documentação seja importante, ela não é a etapa final crítica da entrega contínua antes do lançamento.