A Rotina de uma Equipe de Desenvolvimento

Em uma pequena empresa de desenvolvimento de software, a rotina da equipe é marcada pela correria constante e pela busca incessante de cumprir prazos apertados. O produto principal da empresa é um sistema de gestão escolar, utilizado por diversas instituições de ensino para controlar matrículas, notas, presença e financeiro.

A equipe é composta por quatro desenvolvedores, dois estagiários e um gerente de projetos, que também acumula a função de analista de requisitos. As tarefas são repassadas por meio de mensagens rápidas ou reuniões improvisadas no corredor da empresa. Não há documentação formal; a maioria dos requisitos chega verbalmente ou por e-mails soltos.

As entregas são feitas diretamente em produção, sem um ambiente de testes ou homologação. O ciclo de desenvolvimento é simples: o gerente recebe uma demanda, conversa rapidamente com um dos desenvolvedores e, em poucas horas, o código já está em produção. Não há revisão de código, nem testes automatizados.

Os erros são tratados como parte natural do processo. Quando os usuários reclamam — e isso ocorre com frequência — a equipe age rapidamente para “corrigir” o problema, realizando alterações diretamente no servidor de produção, muitas vezes sem compreender completamente o impacto das mudanças.

Em uma dessas situações, um estagiário foi incumbido de alterar a regra de cálculo de frequência escolar. Por falta de orientação, ele modificou a função central responsável pelo processamento das faltas e acidentalmente gerou um erro que resultou na reprovação automática de 120 alunos de uma das escolas clientes. O problema só foi percebido uma semana depois, após várias reclamações.

Quando o erro foi identificado, a correção foi feita apressadamente, sem análise de causa raiz. O gerente, sob pressão, ordenou que se criasse uma exceção no código que, na prática, desabilitou temporariamente a validação da frequência para todos os alunos, comprometendo a integridade dos dados.

Outro problema recorrente está na interface do sistema. As telas foram criadas conforme as demandas surgiam, sem qualquer padrão ou atenção à usabilidade. Existem botões com rótulos confusos, como "Confirma?" e "Executar!", sem indicar claramente qual ação será realizada. Além disso, algumas páginas demoram mais de 10 segundos para carregar, o que gera frustração entre os usuários.

A equipe raramente discute arquitetura ou padrões de projeto. O código é altamente acoplado: mudanças simples, como adicionar um novo campo ao cadastro de alunos, afetam dezenas de arquivos. Não há um repositório central de código, e parte dos desenvolvedores mantém cópias

locais, dificultando a colaboração e aumentando o risco de sobrescrever funcionalidades importantes.

Apesar dos constantes problemas, o gerente acredita que o time está funcionando bem: “A gente sempre consegue entregar. Não adianta perder tempo com burocracias”, costuma dizer. Para ele, investir tempo em testes, documentação ou revisão de código seria um luxo que a empresa não pode se dar.

Porém, o ambiente está cada vez mais tenso. A rotatividade de profissionais é alta: muitos desenvolvedores permanecem na empresa por menos de um ano, alegando estresse e falta de perspectivas de crescimento. Os clientes, por sua vez, começam a procurar soluções alternativas no mercado, insatisfeitos com a quantidade de falhas e com o suporte deficiente.

Este cenário, infelizmente, não é raro na indústria de software, especialmente em pequenas empresas que não compreendem a importância dos processos de qualidade. A ausência de práticas como revisão de código, testes sistemáticos, documentação e foco na experiência do usuário compromete não apenas a satisfação do cliente, mas a própria sustentabilidade do negócio.

Questionário para ser respondido em dupla

1) Quais critérios da ISO/IEC 25010 foram claramente negligenciados pela equipe? Explique suas escolhas:

( ) Funcionalidade
( ) Usabilidade
( ) Eficiência de desempenho
( ) Manutenibilidade
( ) Portabilidade

2) A ausência de um ambiente de testes e a realização de alterações diretamente em produção pode comprometer principalmente qual alternativa abaixo? Qual a relação disso com qualidade de software?

a) A estética do sistema
b) A confiabilidade do sistema
c) A segurança da informação
d) O custo de desenvolvimento

3) A falta de documentação e de padronização de código impacta diretamente qual característica da qualidade de software? Quais são os possíveis impactos disso e como podemos mitigar esse problema?

a) Compatibilidade

b) Eficiência
c) Manutenibilidade
d) Segurança

4) O que a criação de exceções "temporárias" no código, sem análise de causa raiz, pode gerar? Justifique sua resposta:

( ) Maior controle sobre as funcionalidades.
( ) Redução no tempo de desenvolvimento.
( ) Aumento do risco de novos erros e comprometimento da integridade do sistema.
( ) Melhoria da segurança do software.

5) Na visão de vocês, qual o principal problema na cultura organizacional dessa equipe e como ele afeta a qualidade do software?

6) Proponham três medidas concretas que poderiam ser implementadas imediatamente pela equipe para melhorar a qualidade do software.

7) O gerente acredita que testes, documentação e revisão de código são “burocracias”. Como vocês responderiam a essa afirmação, com base nos riscos e problemas apresentados no texto?

8) Se vocês fossem chamados como consultores para ajudar essa empresa, qual seria o primeiro aspecto que priorizariam resolver? Justifiquem.

9) E se vocês fossem chamados para iniciar um departamento de QA junto a essa equipe, qe tipo de tarefas priorizariam? Pesquisar o que é QA em software e justificar.

10) Se vocês fossem chamados para substituir o gerente da equipe, como equilibrariam as mudanças necessárias com o custo de desenvolvimento?

Instruções:

- Cada dupla deve discutir e responder sem necessidade de entrega.
- Critérios: acertos, clareza, coerência e profundidade das análises.