Bloco 4 - Proatividade em Processos

Objetivo

> Entender como eu criaria e melhoraria fluxos de QA em um cenário onde não há processo formal de qualidade, e as entregas vão diretamente do ambiente de desenvolvimento para produção.

Estrutura Proposta de QA Inicial

Etapas do Fluxo de QA

1. **Planejamento de Testes**
    - Análise de requisitos e definição de critérios de aceitação.
    - Criação de casos de teste manuais com rastreabilidade para as histórias de usuário.
2. **Ambiente de Homologação**
    - Separação clara entre DEV, QA e PRODUÇÃO.
    - Implantação de um ambiente intermediário para testes e validações.
3. **Execução de Testes**
    - Testes funcionais, regressivos, exploratórios e de integração.
    - Registro de bugs no Jira (ou ferramenta equivalente).
4. **Validação e Go/No-Go**
    - Criação de checklist de liberação (validação de build, endpoints críticos, smoke test).
    - Aprovação final por QA antes do deploy em produção.

Tipos de Testes Implementados

- **Funcionais:** validação dos critérios de aceitação e fluxos principais.

- **Regressão:** revalidação dos módulos impactados após cada entrega.
- **Exploratórios:** busca de comportamentos inesperados fora do fluxo feliz.
- **Integração:** validação entre frontend (Vue.js) e backend (Laravel).
- **Performance e Carga:** medição de estabilidade e tempo de resposta.
- **Smoke Tests:** automações rápidas para validar builds antes da liberação.

Ferramentas Sugeridas

Categoria	Ferramenta	Finalidade
Gestão de Testes	Jira + Confluence	Organização de casos, bugs e documentações
Automação E2E	Cypress / Playwright	Testes de interface Vue.js
Automação Backend	PHPUnit / PestPHP	Testes em API Laravel
Versionamento	Git + GitHub Actions	Integração contínua e histórico de execução
Banco de Dados	PostgreSQL	Validação de dados e consistência
Comunicação	Slack / Notion	Notificações e centralização de informações

Métricas Acompanhadas

- **Taxa de Sucesso de Build:** % de builds aprovadas após execução dos testes.
- **Defects per Release:** número médio de bugs encontrados por versão.
- **Tempo Médio de Correção (MTTR):** velocidade na resolução de falhas críticas.
- **Cobertura de Testes:** proporção de código coberto por testes automatizados.
- **Rastreabilidade de Requisitos:** relação entre requisitos, casos e resultados.

Futuras Automações

- Integração de pipelines CI/CD para rodar testes automatizados em cada pull request.
- Envio automático de resultados de testes e logs para o **Notion** (seguindo o modelo de integração já utilizado pela Systock).
- Execução de testes agendados para monitoramento contínuo de performance e disponibilidade.
- Geração automática de relatórios de regressão e dashboards de qualidade.
