Cultura DevOps

Atividade: transformar uma feature em entrega segura

- Escolha uma feature simples (ex: cadastro de usuário / pagamento / webhook).
- Defina critérios de aceitação + 1 métrica de sucesso (SLI).
- Liste riscos técnicos (deploy, dados, segurança, performance).
- Proponha 1 estratégia de rollout (canary, blue/green, feature flag).

Feature escolhida: Cadastro de Usuário

Critérios de aceitação

1. Campos obrigatórios (Nome, e-mail e senha)
    - Nenhum desses campos pode estar vazio
    - Mensagem clara se algum campo estiver vazio
2. Validação de e-mail
    - O e-mail deve estar em um formato válido (usuario@email.com)
    - Não deve permitir e-mails duplicados
    - Mensagem clara se o e-mail já estiver cadastrado
3. Validação de senha
    - Deve ter no mínimo 8 caracteres
    - Deve conter ao menos 3 letras e 3 números
    - Mensagem clara se inválida
4. Persistência
    - Salvar os dados do usuário no banco de dados
    - Armazenar a senha de forma criptografada
5. Resposta do sistema
    - Em caso de sucesso, exibir mensagem "Usuário cadastrado com sucesso" ou redirecionar para login
    - Em caso de erro, exibir mensagem específica e não perder os dados preenchidos
6. Segurança
    - Endpoint apenas aceitar POST
    - Deve ser resistente a ataques do tipo SQL Injection, XSS, CSRF e SSRF
    - Nunca retornar a senha na resposta
7. Performance
    - O cadastro deve responder em até 2 segundos

Métrica de sucesso

1. Disponibilidade (Percentual de requisições bem-sucedidas)

- SLI = Requisições com HTTP 2xx / Total de requisições

Riscos técnicos

1. Deploy
    - Falha no pipeline (Ambiente de CI diferente do de produção)
    - Configuração incorreta (Variáveis de ambiente ou URLs erradas)
    - Incompatibilidade de versão (API nova quebra clientes antigos)
    - Downtime inesperado (Não é zero-downtime ou derruba conexões ativas)
2. Dados
    - Perda de dados
    - Corrupção de dados
    - Inconsistência
    - Duplicação
    - Backup inadequado
3. Segurança
    - Exposição de dados sensíveis (senha em plain text ou logs contendo tokens ou dados pessoais)
    - Autorização falha (Usuário acessa dados de outro usuário ou endpoint sem validação de permissão)
    - Autenticação fraca (Token previsível ou sessão sem expiração)
    - Ataques comuns (SQL Injection, XSS, CSRF e SSRF)
    - Dependência vulnerável (CVE conhecido ou falta de atualização)
    - Falta de rate limiting (Resistência a brute force ou DDoS)
4. Performance
    - Consulta lenta (Falta de índices, N + 1 queries ou full table scan)
    - Crescimento não planejado (Sistema escala para 1k usuários mas não com 100k)
    - Vazamento de memória (Objetos não liberados ou cache sem limite)
    - Falta de escalabilidade (Serviços stateful)
    - Timeout inadequado (Sem timeout ou timeout curto demais)

Estratégia de rollout

A estratégia de rollout é a Canary Release, pois permite validar para um pequeno percentual dos usuários o novo release. Caso haja algum problema, permite um rollback rápido.

Plano de release

1. Fazer o deploy em 5% dos servidores da funcionalidade a ser implantada.
2. Observar os logs em busca de erros e acompanhar a métrica de sucesso.
3. Em caso de algum problema, fazer o rollback dos servidores.
4. Se não houver erros, pode-se aumentar gradativamente os servidores com a funcionalidade implantada.