
PDS TECH

Tech Challenge

O Tech Challenge é o projeto da fase que englobará os conhecimentos obtidos em todas as disciplinas da fase. Esta é uma atividade que, em princípio, deve ser desenvolvida em grupo. Importante atentar-se ao prazo de entrega, pois trata-se de uma atividade obrigatória, uma vez que vale 60% da nota de todas as disciplinas da fase.

Desafio

Com a expansão da oficina para múltiplas unidades e o aumento constante na base de clientes, tornou-se necessário garantir segurança, escalabilidade e alta disponibilidade, além de obter visibilidade total sobre o funcionamento do sistema.

Agora, a direção da oficina quer:

- Controlar acessos e autenticações com segurança;
- Monitorar o ambiente e detectar gargalos em tempo real;
- Adotar soluções **serverless** para autenticação e notificações;
- Segregar a aplicação em repositórios organizados com CI/CD completo;
- Melhorar e documentar a modelagem do banco de dados, garantindo consistência e performance.

Objetivo

Elevar a aplicação a um nível de operação corporativa, utilizando práticas de cloud, infraestrutura como código, segurança e observabilidade.

Requisitos obrigatórios

Autenticação e API Gateway

- Implementar um **API Gateway** (exemplo: AWS API Gateway, Kong, Traefik ou outro).
- Proteger rotas sensíveis da aplicação com autenticação via CPF.
- Criar uma **Function Serverless** para:
    - Validar o CPF do cliente;
    - Consultar a existência e o status do cliente na base de dados;

    - Gerar e devolver um token (JWT) válido para consumo das APIs protegidas.

Estrutura de Repositórios e CI/CD

Organizar o projeto em quatro repositórios separados, cada um com CI/CD implementado (GitHub Actions, GitLab CI, etc.), com deploy automático para a nuvem:

1. **Lambda (Function Serverless)**
2. **Infraestrutura Kubernetes (Terraform)**
3. **Infraestrutura do Banco de Dados Gerenciado (Terraform)**
4. **Aplicação principal executando em Kubernetes**

Regras de proteção:

- Branch **main/master** protegida (sem commits diretos).
- Uso obrigatório de Pull Requests para merge.
- Deploy automático das branches de homologação e produção.

Infraestrutura obrigatória (livre escolha de nuvem):

- **API Gateway** para controle e roteamento.
- **Function Serverless** para autenticação.
- **Banco de Dados Gerenciado** (PostgreSQL, MySQL, SQL Server, etc.).
- **Cluster Kubernetes** com escalabilidade.
- **Terraform** para provisionamento.

Monitoramento e Observabilidade

- Implementar integração com ferramentas como **Datadog** ou **New Relic** (escolha livre).
- Monitorar:
    - Latência das APIs.
    - Consumo de recursos do Kubernetes (CPU, memória).
    - Healthchecks e uptime.
    - Alertas para falhas no processamento de ordens de serviço.

    - Logs estruturados (JSON), incluindo correlação entre requisições.
- Expor dashboards com:
    - Volume diário de ordens de serviço.
    - Tempo médio de execução por status (Diagnóstico, Execução, Finalização).
    - Erros e falhas nas integrações.

Documentação da Arquitetura

Entregar a documentação arquitetural completa, contendo:

- **Diagrama de Componentes** (com a visão de nuvem, APIs, banco e monitoramento).
- **Diagrama de Sequência** para o fluxo de autenticação e abertura de ordens de serviço.
- **RFCs (Request for Comments)** para decisões técnicas relevantes (exemplo: escolha da nuvem, do banco e da estratégia de autenticação).
- **ADRs (Architecture Decision Records)** para decisões arquiteturais permanentes (exemplo: escolha do padrão de comunicação, uso de HPA).
- Justificativa formal para a escolha do banco de dados e ajustes no modelo relacional, com diagramas ER e explicação dos relacionamentos.

Entregável:

Repositórios Git:

- 4 repositórios separados com código, CI/CD e instruções claras no README.md.
- Todos os repositórios devem incluir:
    - Dockerfiles (quando aplicável).
    - Pipelines de CI/CD funcionais.
    - Links para os deploys ativos (se aplicável).

README.md (em cada repositório):

- Descrição clara do propósito.
- Tecnologias utilizadas.
- Passos para execução e deploy.
- Diagrama da arquitetura específica daquele repositório.
- Link para o Swagger/Postman das APIs.

Vídeo de demonstração:

- Upload no YouTube ou Vimeo (público ou não listado) com até 15 minutos de duração.
- Demonstrar:
    - Autenticação com CPF.
    - Execução da pipeline CI/CD.
    - Deploy automatizado.
    - Consumo das APIs protegidas.
    - Dashboard de monitoramento com análise ao vivo.
    - Logs e traces em execução.

Entrega no Portal do Aluno:

- Documento **PDF** único com:
    - Links dos 4 repositórios.
    - Links do vídeo com até 15 minutos.
    - Links das documentações.
    - Confirmação do usuário soat-architecture adicionado a todos os repositórios.