


RELATÓRIO TÉCNICO: PROJETO INTEGRADOR – UNIDADE 05

Disciplina:  Computação em Nuvem e Containers

Instituição: Faculdade CDL | Programa: Residência em TIC 2.0 (Capacita iRede / MCTI Futuro)


Aluno: João Lucas Ribeiro Lima Moreira

1. Planejamento da Arquitetura (Tarefa 1)


O modelo de serviço selecionado para este projeto foi o PaaS (Platform as a Service). Esta escolha fundamenta-se na necessidade de focar no ciclo de vida da aplicação (desenvolvimento e containerização) em detrimento da gestão de infraestrutura de baixo nível. Através do PaaS, garantimos que recursos como elasticidade e alta disponibilidade sejam gerenciados de forma transparente pelo provedor de nuvem.

2. Preparação do Ambiente com Docker (Tarefa 2)

A aplicação foi construída utilizando a imagem oficial python:3.10-slim. As configurações principais incluem:


 Portas: Exposição da porta 8080 para comunicação externa.


 Rede Isolada: Criação da rede-projeto para simular segmentação de tráfego e isolamento de recursos.

 Volume Persistente: Implementação de um volume (meu_volume) mapeado para o diretório de logs da aplicação (/app/dados), assegurando a persistência de dados críticos mesmo após o encerramento ou reinício do container.

3. Simulação de Deploy e Ferramentas (Tarefa 3)

A estratégia de entrega adotada foi a Automatizada (CI/CD). Para otimizar o fluxo de trabalho, utilizamos a extensão oficial "GitHub Actions" integrada diretamente no VS Code.

 Automação: Através de um workflow em YAML, cada atualização de código no GitHub dispara automaticamente o processo de build da imagem Docker e a simulação de deploy na nuvem.

 Monitoramento: A extensão permitiu validar o status do deploy em tempo real sem a necessidade de sair do ambiente de desenvolvimento.

4. Análise Detalhada: Por que não utilizamos IaaS ou SaaS?

Exclusão do modelo IaaS (Infraestrutura como Serviço)

O modelo IaaS oferece o maior nível de controlo, mas foi descartado devido à elevada carga operacional. No IaaS, seríamos responsáveis pela gestão do Sistema Operacional, aplicação de patches de segurança e instalação/configuração da engine do Docker. Para este projeto, o IaaS desviaria o foco do desenvolvimento para a administração de sistemas, tornando o processo de escala e elasticidade manual e complexo.

Exclusão do modelo SaaS (Software como Serviço)

O modelo SaaS é inviável para este projeto integrador, pois não oferece qualquer controle sobre o código-fonte ou a infraestrutura. No SaaS, o desenvolvedor é apenas um utilizador do software pronto. Não teríamos acesso ao Dockerfile, não conseguiríamos configurar redes isoladas ou volumes de persistência, e seria impossível implementar uma esteira de CI/CD personalizada.

☁ 5. Análise de Conceitos de Nuvem (Tarefa 4)

📊 Escalabilidade e Elasticidade: A containerização permite que novas instâncias da aplicação sejam criadas instantaneamente para suprir demandas de pico, reduzindo os recursos automaticamente em períodos de baixa utilização.

🔒 Responsabilidade Compartilhada: O provedor de nuvem assegura a infraestrutura física e a camada de virtualização, enquanto a nossa equipa responde pela integridade do código Python, bibliotecas e configurações do container.

☑ 6. Conclusão e Resultados

A execução deste projeto permitiu validar a portabilidade de aplicações via Docker e a eficiência da automação de deploy. O uso de volumes garantiu a persistência necessária para logs, enquanto a escolha pelo modelo PaaS provou ser a estratégia mais ágil para o desenvolvimento moderno de software.

🔗 7. Referências e Acesso ao Projeto

O código-fonte completo e os fluxos de automação podem ser consultados no repositório público:

🔗 Link do Repositório: <https://github.com/lucasrbmoreira/projetointegrado-irede.git>