Associação Propagadora Esdeva
Centro Universitário Academia - UniAcademia
Artigo de Grupo de Estudo

GRUPO DE ESTUDO EM ARQUITETURA DE MICROSERVIÇOS E IMPLANTAÇÃO NA NUVEM

Laura Castro, Franciane Aprigio, Gustavo Miranda, Camilo Clemente, Júlia Lopes e Wallace Santos[1]
Centro Universitário Academia - UniAcademia, MG
Daves Martins[2]
Centro Universitário Academia - UniAcademia, MG

RESUMO

Este artigo descreve as atividades e aprendizados obtidos no grupo de estudo sobre arquitetura de microserviços e implantação em nuvem. Foram explorados conceitos práticos e ferramentas como Spring Boot, Docker, e CI/CD, aplicados em serviços desenvolvidos colaborativamente.

Apesar de dificuldades na instalação do Kubernetes, o grupo implementou soluções funcionais e adquiriu uma compreensão sólida das tecnologias envolvidas.

Este projeto não apenas promoveu o domínio técnico, mas também destacou a importância do trabalho em equipe e do aprendizado contínuo.

Palavras-chave: JavaScript; NodeJS; webSites;

1 INTRODUÇÃO

A arquitetura de microserviços tem se destacado como uma solução moderna e eficiente para o desenvolvimento de sistemas complexos, especialmente em ambientes que exigem alta escalabilidade, modularidade e rapidez na entrega de funcionalidades. Ao contrário das arquiteturas monolíticas tradicionais, onde toda a aplicação está interligada em um único código, os microserviços dividem o sistema em componentes menores, cada um responsável por uma funcionalidade específica e capaz de operar de forma independente. Essa abordagem tem sido amplamente adotada por empresas de tecnologia que buscam inovação e maior agilidade no mercado competitivo.

Neste contexto, o Grupo de Estudo em Arquitetura de Microserviços e Implantação na Nuvem foi formado com o objetivo de capacitar seus integrantes no

1 Discente do Curso de Bacharelado em Sistemas de Informação e Engenharia de software do Centro Universitário Academia - UniAcademia. Endereço: Rua Halfeld 1.179 – 36.016-000 – Juiz de Fora – MG - Brasil.
2 Docente do Curso de Bacharelado em Sistemas de Informação e Engenharia de software do Centro Universitário Academia - UniAcademia. Orientador.

desenvolvimento e gerenciamento de sistemas distribuídos. Durante os encontros, os participantes exploraram tecnologias modernas, como o framework Spring Boot, ferramentas de containerização como Docker e práticas de automação com pipelines CI/CD. O grupo também teve como meta compreender e aplicar conceitos relacionados à orquestração de containers com Kubernetes, abordando os desafios e benefícios dessa abordagem.

Além de promover o aprendizado técnico, o grupo de estudo também buscou fortalecer a colaboração entre os participantes. Reuniões semanais foram realizadas para alinhar conhecimentos, discutir avanços nos projetos e compartilhar boas práticas. Seminários conduzidos tanto por alunos quanto por orientadores enriqueceram a experiência, permitindo uma troca contínua de ideias e soluções.

A relevância desse estudo está em sua aplicabilidade direta ao mercado de tecnologia, onde a demanda por profissionais familiarizados com microserviços e implantação em nuvem cresce exponencialmente. Empresas que adotam esses conceitos obtêm vantagens competitivas, como maior resiliência dos sistemas e redução de custos operacionais. Por isso, o aprendizado prático proporcionado pelo grupo é um diferencial significativo para a formação dos integrantes.

Este artigo apresenta as atividades realizadas pelo grupo, destacando as tecnologias utilizadas, os desafios enfrentados e as soluções desenvolvidas. Além disso, os resultados obtidos demonstram como a aplicação dos conceitos estudados contribuiu para a consolidação do conhecimento técnico e para a preparação dos integrantes para demandas reais do mercado.

2 REFERENCIAL TEÓRICO

A arquitetura de microserviços, como abordada neste estudo, representa uma abordagem modular para o desenvolvimento de software, na qual uma aplicação é decomposta em pequenos serviços independentes, cada um responsável por uma funcionalidade específica. Essa abordagem, em contraste com a arquitetura monolítica tradicional, oferece uma série de vantagens, como maior escalabilidade, flexibilidade e facilidade de manutenção.

Spring Boot: Uma das principais ferramentas utilizadas neste estudo, o Spring Boot, é um framework Java que simplifica significativamente o desenvolvimento de aplicações baseadas em microserviços. Ele oferece uma configuração automática e convenções que permitem aos desenvolvedores se concentrarem na lógica de negócio, reduzindo a quantidade de código boilerplate.

Docker: A containerização, através do Docker, é uma tecnologia fundamental na implementação de microserviços. Um container Docker encapsula um aplicativo e suas dependências em um ambiente isolado e portátil. Isso garante que o aplicativo funcione da mesma forma em qualquer ambiente, facilitando o deployment e a orquestração.

Kubernetes: O Kubernetes é uma plataforma de orquestração de containers que automatiza o deployment, escalonamento e gerenciamento de aplicações

containerizadas. Ele oferece recursos avançados como autoescalonamento, balanceamento de carga e autocura, tornando-o ideal para gerenciar grandes clusters de containers.

CI/CD: Os pipelines de integração e entrega contínuas (CI/CD) são práticas que automatizam o processo de desenvolvimento de software, desde a integração de código até o deployment em produção. Ferramentas como o GitHub Actions, utilizadas neste estudo, permitem configurar pipelines personalizados para construir, testar e implantar aplicações de forma rápida e confiável.

Outras Tecnologias:

- Java 17: A linguagem de programação Java, na sua versão 17, oferece recursos modernos e aprimoramentos de desempenho que a tornam uma excelente escolha para o desenvolvimento de microserviços.
- PostgreSQL: Um sistema gerenciador de banco de dados relacional (SGBDR) robusto e popular, utilizado para armazenar os dados dos serviços desenvolvidos.
- JUnit e Mockito: Frameworks de testes unitários utilizados para garantir a qualidade do código desenvolvido.
- Swagger: Uma ferramenta para documentar APIs RESTful, facilitando a integração entre os diferentes serviços.
- Prometheus e Grafana: Ferramentas de monitoramento e visualização de métricas, utilizadas para acompanhar o desempenho dos sistemas em produção.

Benefícios da Abordagem:

- Escalabilidade: Cada microserviço pode ser escalado independentemente, permitindo que a aplicação se adapte a diferentes cargas de trabalho.
- Flexibilidade: É possível utilizar diferentes tecnologias para cada microserviço, permitindo escolher a ferramenta mais adequada para cada caso.
- Resiliência: Falhas em um microserviço não comprometem toda a aplicação, graças ao isolamento entre os serviços.
- Desenvolvimento Ágil: Equipes menores podem trabalhar em diferentes microserviços de forma independente, acelerando o desenvolvimento.
- Facilidade de Manutenção: A modularidade dos microserviços facilita a compreensão, a depuração e a atualização do código.

Desafios:

- Complexidade: A arquitetura de microserviços introduz uma complexidade adicional, devido à necessidade de gerenciar a comunicação e a orquestração entre os serviços.
- Teste: Testar sistemas distribuídos pode ser mais desafiador do que testar aplicações monolíticas.
- Deployment: O deployment de microserviços requer uma infraestrutura mais complexa e ferramentas de orquestração como o Kubernetes.

3 METODOLOGIA

O presente grupo adotou uma abordagem metodológica que combinou elementos da pesquisa exploratória e aplicada, com o objetivo de aprofundar o conhecimento sobre arquitetura de microserviços e sua implementação em nuvem. O estudo foi conduzido em um grupo de pesquisa, proporcionando um ambiente colaborativo para a troca de conhecimentos e experiências.

Inicialmente, foi realizada uma imersão teórica sobre o tema, por meio de uma revisão extensiva da literatura científica e técnica. Artigos, livros e materiais online foram consultados para mapear os conceitos fundamentais da arquitetura de microserviços, as tecnologias envolvidas e as melhores práticas do mercado. Essa etapa foi crucial para construir uma base sólida de conhecimento e orientar as atividades práticas do grupo.

Em paralelo à revisão teórica, o grupo iniciou o desenvolvimento prático de um sistema de microserviços. Foram escolhidas tecnologias como Java, Spring Boot, Docker, Kubernetes e GitHub Actions, que se mostraram adequadas para a construção de sistemas distribuídos e escaláveis. A divisão das tarefas entre os membros do grupo foi realizada de forma colaborativa, considerando as habilidades e interesses de cada um.

O desenvolvimento dos microserviços seguiu os princípios da arquitetura de microserviços, com a criação de serviços independentes e de granularidade fina, cada um responsável por uma funcionalidade específica. A comunicação entre os serviços foi realizada por meio de APIs RESTful.

Para garantir a qualidade do software e agilizar o processo de desenvolvimento, foi implementado um pipeline de CI/CD utilizando GitHub Actions. Esse pipeline automatizou as tarefas de construção, teste e deployment dos microserviços, reduzindo o tempo de lançamento e aumentando a confiabilidade do sistema.

Ao longo do desenvolvimento do projeto, o grupo realizou reuniões semanais para discutir o progresso, compartilhar conhecimentos e solucionar problemas. Essa dinâmica colaborativa foi fundamental para o sucesso do projeto, pois permitiu que os membros do grupo aprendessem uns com os outros e encontrassem soluções criativas para os desafios enfrentados.

4. Resultados e Discussão

A implementação da arquitetura de microserviços, utilizando as tecnologias escolhidas, resultou em um sistema modular, escalável e altamente flexível. Cada microserviço desenvolvido demonstrou ser capaz de executar suas funções de forma independente, contribuindo para a robustez e a manutenibilidade do sistema como um todo.

O uso de containers Docker proporcionou um alto grau de portabilidade e isolamento entre os serviços, facilitando o deployment e a gestão do ambiente. A

orquestração com Kubernetes permitiu escalar os serviços de forma dinâmica, adaptando-se às variações de carga.

O pipeline de CI/CD automatizado, baseado no GitHub Actions, agilizou significativamente o processo de desenvolvimento e deployment, reduzindo o tempo de lançamento de novas funcionalidades e garantindo a qualidade do software.

Os resultados obtidos com a implementação da arquitetura de microserviços corroboram com os benefícios esperados para este tipo de abordagem. A modularidade dos microserviços facilitou a compreensão e a manutenção do sistema, permitindo que diferentes equipes trabalhassem em partes distintas do sistema de forma independente. A escalabilidade proporcionada pela utilização de containers e orquestração permitiu que o sistema se adaptasse a diferentes cenários de carga, garantindo um alto nível de desempenho.

No entanto, a adoção da arquitetura de microserviços também trouxe alguns desafios. A complexidade da comunicação entre os microserviços, a necessidade de gerenciar múltiplas dependências e a dificuldade em depurar problemas em um sistema distribuído são alguns dos desafios mais comuns. Além disso, a configuração e a gestão de um ambiente Kubernetes podem ser complexas, exigindo um conhecimento especializado.

5 CONSIDERAÇÕES FINAIS

A experiência adquirida com a implementação da arquitetura de microserviços demonstrou que esta é uma abordagem promissora para o desenvolvimento de sistemas modernos e escaláveis.

No entanto, é importante ressaltar que a adoção desta arquitetura exige um planejamento cuidadoso e um conhecimento profundo das tecnologias envolvidas.

REFERÊNCIAS

- Dragoni, N., et al. "Microservices: Yesterday, Today, and Tomorrow." Springer, 2017.
- Newman, S. *Building Microservices: Designing Fine-Grained Systems.* O'Reilly Media, 2019.
- Schwert, G. W. "Why Does Stock Market Volatility Change Over Time?" *The Journal of Finance,* 1989.
- Prometheus e Grafana. "Documentação Oficial." [https://prometheus.io/docs/]