Implementação de um micro-serviço para gestão de blogs

Gabriel Lucena [1], Marco A. Fernandes[1]

[1] Instituto Metrópole Digital - Universidade Federal do Rio Grande do Norte.
Natal, RN - Brasil.

gabriel.lucena.106@ufrn.edu.br, marco.fernandes.070@ufrn.edu.br

Abstract. *This paper describes a brief review on architectures for corporate systems, doing the implementation of a blog management system using the Model-View-Controller approach.*

Resumo. *Este artigo descreve uma breve revisão de arquiteturas para sistemas corporativos, realizando a implementação de um sistema de gerenciamento de blogs utilizando a abordagem Model-View-Controller.*

1. Introdução

Como parte do processo de crescimento de uma organização e, por consequência, dos seus sistemas, é natural o surgimento de um maior número de complexidades, conforme a necessidade de regras, infraestruturas e sistemas cada vez mais distribuídos estabelece-se. Para isso, em níveis diversos dos sistemas, faz-se necessário a determinação de padrões que sirvam para sanar as principais dores desses sistemas, promovendo maior facilidade em refatorações e a separação adequada das responsabilidades em camadas específicas.

Para demonstrar esses processos e sua organização ao longo do tempo, o projeto Blogs Manager foi proposto e desenvolvido, usando a arquitetura Model-View-Controller (MVC) como base.

Quando definida em 1979 por Reenskaug, o padrão MVC tinha como objetivo servir como um padrão que pudesse ser reutilizado em qualquer sistema que lidasse com um conjunto grande e complexo de dados, permitindo que o software exibisse uma distinção clara entre o que cada parte do sistema poderia fazer e definindo um fluxo de orquestração mais direto.

Inicialmente, esse padrão de arquitetura possuía quatro partes, modelo, responsável por servir como a representação dos dados e do conhecimento, a visualização, setor responsável por exibir ou entregar os dados ao usuário e também por servir como porta de entrada, onde são recebidas e processadas quaisquer interações do usuário, os controladores, responsáveis por realizar a orquestração entre regras de negócio, representação de modelos e permitir que ambos possam comunicar-se sem conhecer seus detalhes e o editor, uma espécie de super-controlador que permitiria a edição da informação apresentada pela view. Posteriormente, esse último foi descartado, e tomou forma o que veio a ser conhecido como MVC.

Desde então, por ter sido desenvolvido diretamente pelo grupo de desenvolvedores responsáveis pelo smalltalk, o MVC toma uma posição preponderante entre as primeiras formas de arquiteturação por vir embarcada de forma nativa em diversas linguagens e frameworks ao longo dos anos, eventualmente evoluindo em outras igualmente populares arquiteturas, como o MVVM, padrão para desenvolvimento web e mobile considerando reatividade, e o MVP, que possuem elementos baseados no

primeiro, mas com sua própria visão e opinião em como cada camada deve lidar com as mudanças.

2. Blog Manager

Como objeto de trabalho nesse estudo, a arquitetura estudada foi usada como base para a implementação de um sistema próprio, um micro serviço baseado em Java que possibilite ao usuário gerir e acompanhar um sistema de blogs, com suas postagens, usuários e comentários, sendo este construído ainda utilizando o framework Spring como base para a implementação dos endpoints, o estilo arquitetural REST para guiar o desenvolvimento e ainda o banco de dados MySQL para armazenamento e permanência dos dados trafegados.

Igualmente, convém estabelecer a forma como o padrão base foi modificado para adequar-se às necessidades do projeto, haja visto que, por ser apenas a implementação do micro serviço, sem interface gráfica associada e a necessidade de que o sistema possa ser organizado de forma mais isolada, não haver da forma convencional a implementação de uma interface de visualização.

Dessa forma, substitui-se a parte tradicional da View pela coleção de estruturas que permitem a comunicação com os meios externos, ou seja, os endpoints que compõem a aplicação, e o tratamento dos dados que são enviados ou recebidos por eles, sendo essa a única interface de interação possível com o usuário. Pela proposta da aplicação, pressupõe-se a disponibilização da documentação e dos endpoints, permitindo ao usuário que monte posteriormente uma camada adicional que torne o sistema em um web-app completo, com as interfaces que forem necessárias.

Para isso, a aplicação tem como objetivo permitir que um dado usuário tenha como criar seus posts, que possam por sua vez ser interagidos por terceiros, mantendo uma discussão sequencial em torno de um tema inicial. Assim, a aplicação organiza-se em três partes:

2.1. Users

Como base para o funcionamento do sistema, o fluxo de usuários passa por permitir a criação e leitura de um usuário. Para isso, uma estrutura de controladores exporta os endpoints “/getuser” e “/createUser”, que permitem o acesso a ambos os pontos. Para uso na implementação de webapp proposta, faz-se necessário que seja recuperado o usuário e reutilizado nos demais fluxos.

2.2. Posts

Em seguida, o fluxo de posts permite por sua vez a criação, deleção, atualização e leitura dos posts, permitindo o acesso ao fluxo de listagem dos posts já existentes, sua modificação e mesmo criação de novos posts, considerando que exista um usuário para ser associado.

2.3. Comments

Por fim, os comentários surgem como forma de dar seguimento aos posts, permitindo que outros usuários registrem suas opiniões sobre um tema, e estando associados ao post original para leitura posterior.

3. Conclusões e próximos passos

Com base nesses pontos, podemos vislumbrar a importância de uma arquitetura compatível com a aplicação para permitir ao mesmo tempo uma maior manutenibilidade e escalabilidade, ao custo de um overhead pequeno de desenvolvimento como base para a construção de um sistema mais robusto e completo, que permita ao time e desenvolvedores futuros mais facilmente encontrar eventuais problemas e incluir novos fluxos ao sistema.

Em passos posteriores, a implementação de um sistema web que utilize os conceitos e endpoints previsto nesse trabalho pode ser uma direção adequada de seguimento para os estudos nesse campo, permitindo o uso de estruturas compatíveis com a camada de View, como o MVVM, como base para construir um sistema mais completo e que entregue ao usuário de forma mais direta a operação do sistema como um todo.

4. Referências

Meinema, J. L. "Corporate architecture: A conceptual approach." University of Twente (1999), https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=7307a59f703f1324baadcdf84bce64454ae41d99.

REENSKAUG, T. “Models - Views- Controllers”. Universitet I Oslo (1979), https://folk.universitetetioslo.no/trygver/1979/mvc-2/1979-12-MVC.pdf.

MERINO, J. “MVC but for non-UI apps”. Blog, 2023, https://jmmv.dev/2023/06/mvc-non-ui-apps.html