PERFORMANCE E ESCALABILIDADE DE SOFTWARE: UM ESTUDO SOBRE MODELOS DE ARQUITETURA EM SISTEMAS DE E-COMMERCE CONSTRUÍDOS EM PHP

Eduarda Cirina Rosa Santos

Giovani Fonseca Ravagnani Disperati

Instituto Federal de Educação, Ciência e Tecnologia de São Paulo

Resumo

É de suma importância, a escolha do modelo de arquitetura de um projeto, o modelo escolhido, impacta na forma como o projeto será desenvolvido e estruturado, podendo trazer pontos positivos, tanto como negativos, trazendo efeitos colaterais a longo prazo, então essa definição é essencial, para anular/reduzir a chance de erros/problemas futuros. O Objetivo deste trabalho é realizar uma revisão sistemática da literatura, analisando modelos de projetos, e seus impactos voltados para as especificações que estão na questão em pesquisa, desenvolvimento de um e-commerce web em PHP, considerando os atributos escalabilidade e performance. O resultado esperado é encontrar um modelo de projeto indicado para um e-commerce escalável, com menor impacto negativo.

Palavras-chave: Modelo de Arquitetura. E-commerce. Escalável. Performance. PHP.

1. Introdução

Souza, 2018, afirma que “O comércio eletrônico vem destacando-se conforme o avanço dos serviços digitais, possibilitando aos microempreendedores colocar seus produtos nas plataformas de venda de marketplace.”. O autor também destaca que “[...]o comércio eletrônico é uma maneira de conseguir expor de forma mais ampla produtos e serviços de pequenos lojistas e sem limite geográfico, podendo expandir suas áreas de vendas.”.

Dada essa grande importância das ferramentas para comércio eletrônico, particularmente plataformas de e-commerce, percebe-se que um cuidado com requisitos não-funcionais é crucial para o atendimento das necessidades dos usuários: sistemas de e-commerces com alta disponibilidade e performance são preferíveis ao fornecer uma melhor experiência para os usuários.

Assim, neste trabalho será apresentado uma análise feita por uma revisão sistemática da literatura, usar o conceito de Engenharia de Software Baseada em Evidências (EBSE), visando o aprofundamento em engenharia de software e levando em consideração atributos críticos para o funcionamento de uma plataforma web, o e-commerce, que seja escalável, com desempenho que suporta milhares de acessos, e que permita expansão futura conforme ocorra a evolução do comércio.

Encontrar o modelo mais indicado para o desenvolvimento de uma loja virtual com PHP, pensando e planejando para a possibilidade de novas funcionalidades, o processo de migração de projeto é dolorosa e apresenta riscos, então o ideal é que seu planejamento seja feito o quanto antes levando-se em consideração o maior número de pontos a serem considerados.

Nesse sentido, o objetivo geral deste trabalho é o desenvolvimento de uma revisão sistemática da literatura, sendo assim, o intuito da revisão é avaliar revisões sistemáticas da literatura (referidas como estudos secundários), de modo que este estudo seja categorizado como uma revisão de literatura terciária. Foi utilizado durante as buscas nas bases bibliográficas o foco em alguns modelos de arquitetura de projeto, tais como: MVC, Microsserviços e Design Patterns, com ênfase na obtenção de resultado um modelo que seja o mais ideal seguindo tais características.

Além disso, por meio da revisão dos artigos e trabalhos analisados, objetiva-se encontrar entre eles uma arquitetura que apresenta uma relação entre desempenho e performance que seja viável do ponto de vista da Engenharia de Software para o desenvolvimento de lojas virtuais em PHP, considerando esses requisitos não-funcionais como prioridades. Ao focar a busca em informações que possam auxiliar no processo da escolha da melhor arquitetura de projetos dentro do escopo que foi abordado durante todo o trabalho, espera-se criar subsídios que possam ser adotados por desenvolvedores de novos projetos e-commerce em PHP.

2. Materiais e Métodos

Para o desenvolvimento deste trabalho foi adotado o modelo de Engenharia de Software baseada em evidências. A pesquisa e a prática baseada em evidências foram desenvolvidas inicialmente na área da saúde, por médicos que queriam avaliar resultados acumulados por experiências científicas e desde então muitas áreas começaram a aderir o método (LIM et al., 2022). O objetivo da Engenharia de Software baseada em evidências é: "[...] fornecer os meios pelos quais as melhores evidências atuais de a

pesquisa podem ser integradas com a experiência prática e valores humanos no processo de tomada de decisão sobre o desenvolvimento e manutenção de software.". (LIM et al., 2022).

Com o método EBSE, tem-se o intuito de conseguir indicar o melhor modelo de projeto de um e-commerce que seja desenvolvido com a linguagem de PHP, atribuindo os critérios de performance e escalabilidade.

Durante a elaboração desta revisão sistemática da literatura, um total de 15 artigos foi cuidadosamente analisado, abrangendo uma ampla variedade de fontes acadêmicas e especializadas. É essencial destacar que esses artigos foram selecionados após um rigoroso processo de filtragem, excluindo aqueles que não atenderam aos critérios durante o fichamento de nível 1.

As pesquisas foram conduzidas no Google Acadêmico e em revistas especializadas, garantindo uma ampla cobertura de fontes confiáveis e relevantes.

Ao longo desta jornada em busca de insights valiosos, foi observado que uma parcela dos artigos analisados apresentava citações pertinentes a metodologias de desenvolvimento, incluindo as metodologias ágeis em seus processos de desenvolvimento.

Dos 15 artigos analisados, apenas 4 atenderam 100% todos os requisitos da pesquisa, tais como serem sobre o desenvolvimento de um e-commerce, serem desenvolvidos em PHP, e terem citações sobre melhoria de performance e escalabilidade. Esses 4 artigos foram apresentados no capítulo de Discussões sobre a revisão da Literatura.

A análise minuciosa dessas citações desempenhou um papel substancial na consolidação dos resultados alcançados, fornecendo uma base sólida e embasada para as conclusões desta pesquisa. Esses estudos enriqueceram o trabalho, permitindo contextualizar e ampliar a compreensão sobre o uso de metodologias dentro do processo de desenvolvimento, especialmente considerando um projeto de e-commerce em PHP.

Cada contribuição citada desempenhou um papel fundamental no enriquecimento e na qualidade dos resultados obtidos.

3. Resultados e Discussão

Este trabalho irá seguir conforme o planejado no cronograma pré-estabelecido, levando em consideração o cronograma parcial como início desta pesquisa.

Durante fase de coleta de informações e de artigos para realizar a revisão sistemática, foram encontrados alguns artigos que atingiam os requisitos que envolvem a pergunta de pesquisa deste trabalho. Os artigos que foram utilizados para a análise deste trabalho foram:

- Migração de sistemas monolíticos para microsserviços: Estudo de caso de migração de um módulo de pagamentos de e-Commerce.
- Solução Headless REST API para e-commerce.
- Loja virtual utilizando interface Web/SMS.
- Transformação de um site de e-commerce e um e-marketplace.

3.1 Migração de sistemas monolíticos para microsserviços: Estudo de caso de migração de um módulo de pagamentos de e-Commerce.

O primeiro artigo a ser analisado foi o: migração de sistemas monolíticos para microsserviços: estudo de caso de migração de um módulo de pagamentos de e-commerce. Este estudo é um sistema real de uma loja virtual no ramo de viagens. O autor deste projeto visa resolver problemas como: dificuldade de escalabilidade e manutenção, sendo que é uma arquitetura de legado (STRADOLINI, 2021).

O foco do projeto foi voltado para o módulo de pagamentos, onde havia sido desenvolvida em modelo monolítico, porém dentro do projeto já existiam pequenos módulos que foram quebrados desse monolítico e desenvolvidos como microsserviços (STRADOLINI, 2021).

O autor cita que dentro deste projeto que houve um grande exemplo na decisão de arquitetura que não evoluiu de forma efetiva ao longo dos anos, monolítico, este cenário continha mais de 5000 arquivos e mais de 2,5 milhões de linhas de código.

Stradolini cita que todo esse tamanho não é proporcional quando se refere a códigos de teste dentro da aplicação. A cobertura de testes de 10 % é muito inferior à porcentagem de cobertura recomendada por especialistas de desenvolvimento de software (STRADOLINI, 2021).

A manutenção de projetos como esse é muito complexa e desafiadora, e requer muito cuidado e atenção ao realizar qualquer modificação no código. São muitas as Funcionalidades, funcionalidades agrupadas, com falta de cultura e processos de testes, o

grande desafio está em conseguir modificar e ajustar o código, garantindo que o comportamento original se mantém preservado (STRADOLINI, 2021).

Stradolini, 2021, aponta que “Trabalhar numa base de código desta forma é bastante custoso e propenso a erros.”.

Projetos de legado geralmente tem esse efeito colateral ao ser modificado, para corrigir ou criar uma nova funcionalidade no sistema, é muito comum casos assim gerarem erros completamente desconexos com o que foi mudado, devido ao grande acoplamento existente no código e fazendo com que a longo prazo essa situação se torne insustentável, sendo preciso que haja um processo de manutenção bem definido (STRADOLINI, 2021).

O projeto sendo monólito, não há possibilidades de fazer escalabilidade por módulos, em casos de picos de acessos, para manter o site no ar, é preciso abrir uma nova instância da aplicação, servidores potentes que possam inicializar a aplicação monólito (STRADOLINI, 2021).

O autor relata, que por conta do tamanho da aplicação, e a questão do tempo para o desenvolvimento, foi escolhido apenas o módulo de pagamentos para a migração, por se tratar de um módulo de suma importância para os negócios, e o módulo servirá de base para as próximas migrações no futuro (STRADOLINI, 2021).

Levando em consideração o processo de migração do módulo de pagamento do e-commerce, foram apresentados os seguintes resultados considerando os parâmetros vistos anteriormente, escalabilidade e performance (STRADOLINI, 2021).

No artigo, é mencionado que foi realizada uma investigação e comparação considerando diversas tecnologias, abordagens e técnicas para haver o passo a passo para a efetividade da migração, as avaliações tiveram como pontos principais: resolver os problemas de acoplamento, escalabilidade e disponibilidade da aplicação. (STRADOLINI, 2021) O levantamento dos resultados foi feito por meio de uma discussão, com 4 profissionais do ramo de desenvolvimento de sistemas, com idades entre 26 a 31 anos, tendo de 4 a 8 anos de experiência com sistemas (STRADOLINI, 2021).

Os colaboradores concordaram entre si sobre problemas de coesão e acoplamento do sistema monolítico. O grande tamanho das classes e da base de dados, aliado a um grande número de dependências é a explicação utilizada por todos. O colaborador mais sênior destacou que o maior problema é o acoplamento, visto que a coesão não parece ser necessariamente um problema baseado nas informações da pesquisa (STRADOLINI, 2021).

A manutenção do sistema teve pontuação unânime pelos especialistas que avaliaram o projeto, por conta do uso de novas tecnologias e processos de qualidade de software, como a legibilidade e reuso do código. Porém, um dos especialistas com maturidade sênior apontou um ponto, a complexidade do módulo de pagamentos aumentou, pelo fato de terem adicionado na arquitetura o message broker RabbitMQ, e sendo assim a manutenção em um primeiro momento, se tornou mais complexa, mesmo acreditando que essas novas estruturas irão se pagar com o tempo (STRADOLINI, 2021).

Os especialistas concordaram que a estabilidade e performance do sistema monolítico apresentado é deficitária, todos apontam pontos como: gasto de recursos necessários para escalar horizontalmente a aplicação, visto que caso apenas um módulo esteja em pico de tráfego, uma aplicação inteira precisa ser instanciada, gerando gasto de recursos de hardware e por módulos que não precisam ser escalonados (STRADOLINI, 2021).

A performance foi um dos pontos citados como dor no sistema com a arquitetura monólito, houve melhora por conta de uma maior eficiência do uso dos recursos de hardware e software, já que é possível escalar separadamente o módulo de pagamento ou o message broker, o módulo de pagamentos estando isolado do sistema legado, evita problemas de lentidão causados pelo sistema maior (STRADOLINI, 2021).

O especialista sênior, aponta que na análise feita por ele sobre a nova arquitetura proposta, ele confirma que diversas melhorias apontadas no trabalho não se originam apenas da arquitetura de software proposto, mas sim de todo o contexto em volta, mudanças de provedora, nuvem, plataforma de Kubernetes, etc. Finaliza reforçando que seu ponto anterior de que parte dos problemas da arquitetura legada analisada não foram gerados pela arquitetura em si, mas por diversos outros fatores (STRADOLINI, 2021).

3.2 Solução Headless REST API para e-commerce

O projeto Solução Headless REST API para e-commerce foi realizado visando reestruturar e reformular os projetos mais antigos da empresa Blue-Infinity, associados à uma loja virtual. A aplicação era toda em um framework em PHP e tinha vários anos de desenvolvimento (FARINHA, 2019).

A dor que eles viram dava-se à complexidade e dimensão do antigo ecossistema, tornando cada vez mais difícil a adaptação para a atender as necessidades dos clientes (FARINHA, 2019). Para ser realizada a reformulação, o projeto foi dividido em duas

partes: a primeira parte refere-se à vista, que é apresentada ao cliente, e à parte lógica, onde é efetuado todo o tratamento da informação (FARINHA, 2019).

Como objetivo, era esperado que fosse desenvolvida uma API genérica com aplicabilidade à loja virtual, completamente independente, que possa ser consumida pelo projeto responsável, mostrada ao cliente (FARINHA, 2019).

Falando sobre a arquitetura do projeto, o autor aponta que uma das vantagens em utilizar o sistema por camadas é que uma aplicação que segue essa arquitetura poderá utilizar um servidor dedicado apenas à API, outro servidor para autenticar pedidos efetuados pelo cliente e outro para guardar as informações e/ou cache. Outro ponto que foi considerado neste projeto, falando de arquitetura REST, é a importância dada à uniformização da interface, exemplos:

- "Cada recurso é definido nos pedidos efetuados através da utilização de URIs (Uniform Resource Identifier)" (FARINHA, 2019).
- "Os pedidos efetuados sobre um recurso disponibilizam informação suficiente que representa a ação que se permite efetuar, através dos múltiplos meta dados associados, como o verbo HTTP (HyperText Transfer Protocol) do pedido ou Content-Type" (FARINHA, 2019).
- "Utilização de HATEOAS (Hypermedia As The Engine Of Application State) que define que o estado da aplicação deverá vir bem explícito nos pedidos do cliente e respostas do servidor.” (FARINHA, 2019).

Por ser uma solução Headless, projeto está associado a um CMS, o CMS contém um backend para manuseamento de conteúdo e um frontend para mostrar esse mesmo conteúdo. Um Headless CMS faria com que este passasse a não ter parte gráfica e restasse apenas a parte lógica do mesmo (FARINHA, 2019).

Ao falarmos de uma arquitetura como esta que envolve o desacoplamento total de dois componentes que envolvem um CMS, estas podem trabalhar quase independentemente uma da outra, pelo da arquitetura REST permitir. Cada uma pode ser modificada sem interferir com a outro, trazendo flexibilidade e escalabilidade ao ecossistema (FARINHA, 2019).

O projeto foi concluído com sucesso, usando a API REST como arquitetura, para servir uma loja online, com o “extra” do desenvolvimento mais genérico, que consiga ser utilizado também como base de qualquer loja online. Houve uma migração do framework Laravel para o Lumen, durante o processo de desenvolvimento (FARINHA, 2019).

A única condição imposta foi a de ser um e-commerce, todas as outras funcionalidades necessárias foram adicionadas através da experiência profissional obtida no projeto, que possibilitou definir prioridades e objetivos (FARINHA, 2019). O autor destaca que “[...] todas as outras funcionalidades foram implementadas de modo a complementar este comportamento através da adição de “utilidades” que podem ser encontradas em muitas das lojas online - promoções, áreas de administração, auditoria, entre outros." (FARINHA, 2019)

Foram utilizados dois tipos de testes, unitários e de carga, com o intuito de garantir um bom funcionamento geral da aplicação. Os testes unitários tiveram um papel fundamental na descoberta de alguns bugs/erros que foram aparecendo em cada interação de entrega (FARINHA, 2019). Segundo Farinha, 2019, isto prova "[...] a relevância da implementação destes previamente à adição de novas funcionalidades em qualquer ambiente de desenvolvimento."

Os testes de carga mostraram que a migração para uma versão do framework mais "leve" trouxe vantagens na performance da aplicação, que faz parte de uma das grandes necessidades de qualquer API (FARINHA, 2019).

Como citado anteriormente, existiu uma migração da implementação do projeto para Lumen, e por isso houve a necessidade de se habilitar alguns itens inativos por omissão, que podem afetar um pouco o desempenho desta (FARINHA, 2019). Mesmo não sendo uma situação de nível crítico, há que ter noção de que o propósito principal da criação do Lumen foi baseado em fornecer muitas das funcionalidades do Laravel, priorizando o desempenho da aplicação, através da facilidade e desativação de alguns dos seus componentes. No contexto de como o projeto seria desenvolvido, houve necessidade de ativar a abstração da base de dados. Uma melhoria poderia ser realizada através da “tradução” para raw queries, possibilitando assim desativar a sua injeção no projeto, fornecendo um maior desempenho por parte da API (FARINHA, 2019).

3.3 Loja virtual utilizando interface Web/SMS

Este trabalho foi desenvolvido para elaborar um sistema web que funcionará como uma loja virtual de vendas de celulares. Foi implementada tamém a funcionalidade de se comunicar com o cliente via SMS (BALIEIRO, 2008).

O sistema desenvolvido foi feito de forma independente, uma vez que suas atribuições permitem, além das vendas de celulares, a administração da loja e controle de estoque, não sendo necessário integrá-lo a outros sistemas (BALIEIRO, 2008).

Foram considerados os seguintes atributos no desenvolvimento:

• Amigabilidade - Pensando em usabilidade, para que os clientes se sintam bem ao navegar pelo site.

• Segurança - Dados pessoais são sensíveis e precisam de um bom sistema de segurança.

• Manutenibilidade - o sistema precisa ser projetado para ter a possibilidade de uma boa manutenção, no sentido de correções de erros e expansão.

Características do projeto incluíam seu desenvolvimento em PHP, com uso do Apache como Servidor e Banco de dados Mysql.

A Engenharia de software, foi a etapa considerada a mais importante pelo autor, a qual foram determinados alguns pontos de importância do projeto (BALIEIRO, 2008).

Foi utilizada a metodologia estruturada para retratar o fluxo e o conteúdo das informações utilizadas pelo sistema, dividir o sistema em partições funcionais e comportamentais e descrever a essência daquilo que será construído (BALIEIRO, 2008).

O modelo que o projeto seguiu foi o Modelo Linear Sequencial, que segue este ciclo de vida: análise de requisitos, projeto, codificação, Testes e Manutenção (BALIEIRO, 2008).

As conclusões que o autor teve sobre este projeto foram: a metodologia utilizada no projeto, estrutural, foi adequada e proporcionou que o desenvolvimento do projeto de forma objetiva e organizada. Com isso, foi possível analisar pontos mais críticos e planejar as melhores soluções (BALIEIRO, 2008).

Considerando as ferramentas utilizadas, como MySQL, PHP e o servidor apache, todas atenderam às expectativas para o projeto (BALIEIRO, 2008).

O autor sabe que o projeto precisa de melhorias no futuro, como imagens nos produtos que não foram implementadas, além da possibilidade de uma pesquisa na plataforma mais ampla sobre o modelo que deseja comprar de celular (BALIEIRO, 2008).

E é enfatizado pelo autor que a metodologia estruturada foi adequada porque garantiu um projeto organizado e permitiu analisar os pontos críticos (BALIEIRO, 2008).

3.4 Transformação de um site de um site de e-commerce em um e-martketplace

Este projeto consistiu na transformação de um site de e-commerce em um e-marketplace, aproveitando a oportunidade do aumento dos acessos e das vendas online durante a pandemia. Para isso, foi desenvolvido um módulo que permite aos fornecedores controlarem seus produtos e vendas, possibilitando o gerenciamento das vendas, a unificação dos valores dos itens e o repasse dos valores aos vendedores após a aprovação do pagamento (GEMELLI, 2015).

A tecnologia escolhida para o desenvolvimento do e-marketplace foi a Community do Magento, visando evitar limitações, como o número de vendedores ou produtos. Outro fator que contribuiu para a escolha dessa tecnologia foi o fato de o e-commerce atual ter sido desenvolvido pela mesma plataforma (GEMELLI, 2015).

A Community Edition (CE) do Magento é uma versão gratuita da plataforma Magento, e o framework utilizado por trás do seu desenvolvimento é o Zend Framework, baseado em PHP, com arquitetura de projeto no modelo MVC e banco de dados MySQL (GEMELLI, 2015).

O e-marketplace apresenta um catálogo da loja abastecido por vários fornecedores. Qualquer fornecedor que produza ou comercialize produtos do mesmo nicho de mercado da loja poderá oferecer seus produtos na loja virtual (GEMELLI, 2015).

O uso da plataforma Magento, desenvolvida em PHP através do framework Zend, possibilita que a aplicação seja segura e escalável, utilizando o conceito de orientação a objetos (GEMELLI, 2015).

Além disso, a estrutura do projeto segue o padrão MVC, arquitetura utilizada pelo framework Zend. Foi definida uma arquitetura de banco de dados do tipo EAV, que permite flexibilidade na modelagem de dados, possibilitando a adição de novos atributos às entidades sem a necessidade de alterar a estrutura da tabela de dados (GEMELLI, 2015).

O autor conclui que o Magento CE foi escolhido por ser um sistema robusto e líder no mercado mundial e brasileiro. No final do desenvolvimento do módulo, ele foi adicionado a uma loja como teste (GEMELLI, 2015).

4. Considerações Finais

O desenvolvimento deste projeto teve como objetivo avaliar padrões arquiteturais melhores para o desenvolvimento de lojas virtuais, considerando a performance e desempenho em PHP. Foi avaliada uma amostra de dados obtida a partir de trabalhos e artigos que têm o desenvolvimento com essas mesmas características.

Ao todo, durante a análise da amostra, foram feitas diversas avaliações em muitos projetos que não foram selecionados para a análise dentro deste trabalho, pois não contribuíram de forma total com todas as características avaliadas, mas foram selecionados os trabalhos que passaram a ser estudados e avaliados.

Todos os trabalhos avaliados estavam dentro do escopo projetado e com as características devidas selecionadas para encontrar filtros e parâmetros. Avaliando e estudando cada um dos trabalhos, foi possível ver que, independentemente da arquitetura escolhida para o projeto a ser desenvolvido, existem outras variáveis que precisam ser avaliadas para que a arquitetura escolhida tenha um maior percentual de sucesso de aderência ao projeto. Dentre essas variáveis que interferem no resultado do uso de uma arquitetura, tais como: tipo de servidor, o framework utilizado, o banco de dados escolhido, até mesmo o modelo de negócio, têm impacto na performance e desempenho de um projeto. Essas características precisam estar bem definidas e alinhadas no cenário total para que se tenha sucesso no processamento do projeto.

Nesse sentido, as diferentes arquiteturas adotadas possuem potencial para atender aos requisitos não-funcionais considerados desde que tais fatores sejam levados em consideração.

Como trabalho futuro sugere-se a continuidade da investigação por projetos e estudos de caso em busca de novos cenários e características que contribuem de forma positiva ou negativa em um projeto seguindo uma determinada arquitetura, que possa ser implementada com PHP em e-commerces.

5. Referências

a. ALMEIDA, J. Escalabilidade. Revista Trama Interdisciplinar, v. 5, n. 3, 2014.
b. ALMEIDA, V. S. Plataforma de e-commerce integrada a microserviços. 2022.
c. ALVES, J. M. Utilização do modelo MVC no desenvolvimento de aplicações E-saúde utilizando o padrão openEHR. Dissertação (B.S. thesis) — Universidade Tecnológica Federal do Paraná, 2017.
d. BALIEIRO, F. A. d. H. Loja virtual utilizando interface web/sms. Universidade Federal do Rio de Janeiro, 2008.
e. FARINHA, M. J. S. Solução Headless REST API para e-commerce. Tese (Doutorado), 2019.

f. GEMELLI, B. G. M. Transformação de um site e-commerce em um e-marketplace. 2015.
g. LIM, W. M. et al. Environmental social governance (esg) and total quality management (tqm): a multi-study meta-systematic review. Total Quality Management & Business Excellence, Taylor & Francis, p. 1–23, 2022.
h. RIBEIRO, F. E. S.; FREITAS, M. E-commerce. Seminário de Tecnologia Gestão e Educação, v. 3, n. 1, 2021.
i. SOUZA, T. Testes de desempenho de software: Teoria e prática. Sociedade Brasileira de Computação, 2018.
j. STRADOLINI, C. J. Migração de sistemas monolíticos para microsserviços: estudo de caso de migração de um módulo de pagamentos de e-commerce. 2021.