APLICAÇÃO WEB PARA AGENDAMENTO DE SERVIÇOS E PAGAMENTOS ONLINE COM INFRAESTRUTURA EM NUVEM E ARQUITETURA ESCALÁVEL UTILIZANDO SERVIÇOS DA AWS

WEB APPLICATION FOR SERVICE SCHEDULING AND ONLINE PAYMENTS WITH CLOUD INFRASTRUCTURE AND SCALABLE ARCHITECTURE USING AWS SERVICES

Enzo Rocco dos Santos
Tecnólogo em Análise e Desenvolvimento de Sistemas
Centro Universitário Einstein de Limeira- UniEinstein
enzo.rocco@gmail.com

Gustavo Manfredi
Centro Universitário Einstein de Limeira- UniEinstein
Tecnólogo em Análise e Desenvolvimento de Sistemas
gugamanfredi@gmail.com

Vitor Gabriel de Almeida
Tecnólogo em Análise e Desenvolvimento de Sistemas
Centro Universitário Einstein de Limeira- UniEinstein
vgalmeida10@gmail.com

Thiago Salhab Alves
Mestre em Ciência da Computação
Centro Universitário Einstein de Limeira- UniEinstein
thiagosalhab@gmail.com

RESUMO

A transformação digital dos pequenos negócios tem impulsionado a adoção de soluções baseadas em computação em nuvem para automatizar processos operacionais, aumentar a disponibilidade dos sistemas e melhorar a experiência dos usuários. Nesse contexto, este estudo analisa a implementação de uma arquitetura escalável em nuvem aplicada a uma plataforma

web de agendamento de serviços e pagamentos online, avaliando como a integração de serviços da Amazon Web Services (AWS) contribui para a escalabilidade, disponibilidade, segurança e automação dos processos de negócio. A pesquisa caracteriza-se como aplicada, de natureza qualitativa, utilizando o método de estudo de caso, no qual foi desenvolvida uma aplicação web utilizando React.js no front-end e FastAPI no back-end, integrada aos serviços Amazon EC2, Amazon S3, CloudFront, Amazon RDS, DynamoDB e Amazon Simple Email Service (SES). A avaliação concentrou-se na análise da arquitetura implementada, dos testes funcionais realizados durante o desenvolvimento e da capacidade da solução em atender aos requisitos de modularidade, integração e escalabilidade. Os resultados evidenciam que a utilização integrada dos serviços AWS permitiu a construção de uma infraestrutura flexível, com baixo acoplamento entre componentes, facilidade de expansão e suporte à automação de notificações e pagamentos. Conclui-se que a arquitetura proposta apresenta potencial para atender pequenos e médios prestadores de serviços, constituindo uma alternativa tecnicamente viável para aplicações cloud-native, além de fornecer uma base consistente para futuras avaliações quantitativas de desempenho e usabilidade.

Palavras-chave: Computação em nuvem. Arquitetura escalável. Amazon Web Services. Agendamento de serviços. FastAPI. React.js.

ABSTRACT

Digital transformation has driven small businesses to adopt cloud computing solutions to automate operational processes, increase system availability, and improve user experience. In this context, this study analyzes the implementation of a scalable cloud architecture applied to a web platform for service scheduling and online payments, evaluating how the integration of Amazon Web Services (AWS) contributes to scalability, availability, security, and business process automation. The research is characterized as applied qualitative research using the case study method. A web application was developed using React.js for the front-end and FastAPI for the back-end, integrated with Amazon EC2, Amazon S3, CloudFront, Amazon RDS, DynamoDB, and Amazon Simple Email Service (SES). The evaluation focused on the implemented architecture, functional testing performed during development, and the solution's ability to meet requirements related to modularity, integration, and scalability. The results indicate that the integrated use of AWS services enabled the construction of a flexible infrastructure with low coupling between components, ease of expansion, and support for automated notifications and online payments. It is concluded that the proposed architecture represents a technically feasible solution for cloud-native applications aimed at small and medium-sized service providers while providing a consistent basis for future quantitative evaluations of performance and usability.

Keywords: Cloud Computing. Scalable Architecture. Amazon Web Services. Service Scheduling. FastAPI. React.js

INTRODUÇÃO

A crescente digitalização dos processos organizacionais tem impulsionado empresas de diferentes portes a migrarem suas aplicações para ambientes de computação em nuvem, buscando maior disponibilidade, escalabilidade, segurança e redução dos custos operacionais. Entre os setores que mais têm se beneficiado dessa transformação encontram-se os pequenos prestadores de serviços, que dependem de ferramentas eficientes para organizar agendas, gerenciar clientes e automatizar processos administrativos.

Apesar da ampla disseminação das tecnologias digitais, muitos profissionais autônomos e pequenos estabelecimentos ainda utilizam métodos tradicionais de agendamento, como planilhas eletrônicas, agendas físicas ou aplicativos de mensagens instantâneas. Essas práticas dificultam o gerenciamento simultâneo de horários, aumentam a ocorrência de conflitos de agenda, comprometem a comunicação com os clientes e favorecem a ocorrência de faltas sem aviso prévio (no-shows), impactando diretamente a produtividade e a rentabilidade do negócio.

Segundo Mell e Grance (2011), a computação em nuvem fornece acesso sob demanda a recursos computacionais compartilhados, permitindo que aplicações sejam rapidamente provisionadas, escaladas e administradas com menor esforço operacional. Complementarmente, Armbrust *et al.* (2010) destacam que a elasticidade constitui uma das principais características desse paradigma, possibilitando que sistemas adaptem automaticamente sua capacidade computacional conforme a demanda.

Nesse contexto, plataformas de agendamento online tornam-se uma alternativa para automatizar processos administrativos, reduzir erros operacionais e melhorar a experiência dos usuários. Além da organização das agendas, essas soluções podem integrar pagamentos digitais, envio automático de notificações, programas de fidelidade e acompanhamento dos serviços realizados.

Diversas plataformas comerciais oferecem funcionalidades semelhantes, como GetNinjas e Singu. Entretanto, essas soluções apresentam limitações relacionadas à personalização da plataforma, aos custos de utilização e à flexibilidade para adaptação a diferentes segmentos de prestação de serviços. Dessa forma, torna-se relevante investigar arquiteturas capazes de oferecer elevada escalabilidade, baixo custo de manutenção e facilidade de evolução tecnológica.

A utilização da Amazon Web Services (AWS) destaca-se nesse cenário devido ao amplo conjunto de serviços gerenciados disponibilizados para desenvolvimento de aplicações cloud-native. Recursos como Amazon EC2, Amazon RDS, Amazon S3, CloudFront, DynamoDB e Amazon Simple Email Service (SES) permitem construir arquiteturas distribuídas capazes de atender aplicações com diferentes níveis de demanda, mantendo elevados padrões de disponibilidade, desempenho e segurança.

Diante desse cenário, este estudo tem como objetivo analisar a implementação de uma arquitetura escalável baseada em serviços da Amazon Web Services aplicada a uma plataforma web de agendamento de serviços e pagamentos online, avaliando sua contribuição para a disponibilidade, modularidade, integração dos componentes e automação dos processos operacionais.

Como contribuição científica, este trabalho demonstra como a integração de diferentes serviços gerenciados da AWS pode ser empregada para construir uma arquitetura escalável destinada a aplicações de pequeno porte, discutindo os benefícios obtidos em relação à organização da infraestrutura, facilidade de manutenção e potencial de expansão da solução.

Este artigo está organizado da seguinte forma: a Seção 2 apresenta a fundamentação teórica sobre computação em nuvem, arquitetura escalável, desenvolvimento web e segurança em aplicações distribuídas; a Seção 3 descreve a metodologia adotada; a Seção 4 apresenta os resultados obtidos e sua discussão; por fim, a Seção 5 reúne as considerações finais e as perspectivas para trabalhos futuros.

DESENVOLVIMENTO

Computação em Nuvem

A computação em nuvem (Cloud Computing) consolidou-se como um dos principais paradigmas da Tecnologia da Informação ao permitir o fornecimento de recursos computacionais sob demanda, eliminando a necessidade de investimentos elevados em infraestrutura física. Segundo Mell e Grance (2011), do National Institute of Standards and Technology (NIST), computação em nuvem consiste em um modelo que possibilita acesso conveniente e sob demanda a um conjunto compartilhado de recursos computacionais configuráveis, como servidores, armazenamento, redes e aplicações, que podem ser rapidamente provisionados e liberados com mínimo esforço administrativo.

Entre as principais características desse paradigma destacam-se o autoatendimento sob demanda, amplo acesso pela rede, compartilhamento de recursos, rápida elasticidade e serviços mensuráveis (Mell; Grance, 2011). Essas propriedades permitem que aplicações aumentem ou reduzam automaticamente sua capacidade computacional conforme a necessidade, tornando a infraestrutura mais eficiente e econômica.

Armbrust *et al.* (2010) destacam que a elasticidade representa um dos maiores diferenciais da computação em nuvem, permitindo que aplicações suportem variações significativas de carga sem necessidade de superdimensionamento permanente da infraestrutura. Essa característica torna-se especialmente relevante para sistemas de agendamento online, que podem apresentar picos de utilização em determinados horários do dia ou períodos específicos do mês.

Além da elasticidade, a computação em nuvem favorece a alta disponibilidade dos sistemas por meio da distribuição dos serviços em múltiplas zonas de disponibilidade e regiões geográficas, reduzindo riscos de indisponibilidade decorrentes de falhas de hardware ou infraestrutura (AWS, 2023).

Arquiteturas Escaláveis para Aplicações Web

O crescimento contínuo das aplicações web exige arquiteturas capazes de suportar expansão do número de usuários sem comprometer desempenho, disponibilidade e facilidade de manutenção.

Segundo Bass, Clements e Kazman (2022), uma arquitetura de software representa o conjunto de estruturas necessárias para compreender um sistema, incluindo seus componentes, suas propriedades e os relacionamentos existentes entre eles. Uma arquitetura bem projetada influencia diretamente atributos de qualidade como desempenho, escalabilidade, segurança e confiabilidade.

A escalabilidade pode ser classificada em vertical e horizontal. A escalabilidade vertical consiste no aumento dos recursos computacionais de um único servidor, enquanto a escalabilidade horizontal distribui a carga entre múltiplas instâncias da aplicação, permitindo maior capacidade de processamento e maior tolerância a falhas (Bass; Clements; Kazman, 2022).

Fowler (2019) destaca que arquiteturas modernas privilegiam componentes fracamente acoplados, facilitando manutenção, evolução e implantação contínua. Essa abordagem reduz

impactos decorrentes de alterações em módulos específicos e favorece a reutilização dos componentes.

No contexto deste trabalho, a arquitetura proposta utiliza diferentes serviços especializados da AWS, distribuindo responsabilidades entre armazenamento, processamento, banco de dados e serviços de comunicação, característica compatível com princípios arquiteturais modernos voltados à modularidade e escalabilidade.

Amazon Web Services (AWS)

A Amazon Web Services (AWS) constitui atualmente uma das maiores plataformas de computação em nuvem do mundo, disponibilizando centenas de serviços gerenciados destinados ao desenvolvimento de aplicações escaláveis.

Segundo o AWS Well-Architected Framework (2023), aplicações desenvolvidas na plataforma devem seguir cinco pilares fundamentais:

- excelência operacional;
- segurança;
- confiabilidade;
- eficiência de desempenho;
- otimização de custos.

No presente trabalho foram utilizados diversos serviços gerenciados da AWS, cada um desempenhando funções específicas na arquitetura proposta.

O Amazon EC2 foi empregado para hospedar o back-end desenvolvido em FastAPI, oferecendo flexibilidade na configuração do ambiente computacional e possibilidade de expansão futura da capacidade de processamento.

O Amazon S3 foi utilizado para armazenamento estático da aplicação React.js, proporcionando elevada durabilidade dos arquivos e integração nativa com o serviço Amazon CloudFront, responsável pela distribuição do conteúdo por meio de uma rede global de entrega (Content Delivery Network – CDN), reduzindo a latência de acesso pelos usuários.

Para persistência dos dados relacionais foi utilizado o Amazon RDS com PostgreSQL, serviço gerenciado que automatiza tarefas administrativas como backups, replicação e monitoramento.

Complementarmente, empregou-se o Amazon DynamoDB para armazenamento de dados não relacionais relacionados ao programa de fidelidade e informações de acesso rápido, reduzindo a carga sobre o banco relacional.

O envio automatizado de mensagens eletrônicas foi realizado utilizando o Amazon Simple Email Service (SES), permitindo recuperação de senha, notificações de agendamento e comunicação automática entre clientes e prestadores de serviços.

A utilização integrada desses serviços contribui para uma arquitetura modular, desacoplada e preparada para futuras expansões da aplicação.

Desenvolvimento Web Moderno

O desenvolvimento de aplicações web modernas caracteriza-se pela separação entre as camadas de apresentação, regras de negócio e persistência dos dados, favorecendo manutenção, reutilização de código e escalabilidade.

O React.js é uma biblioteca JavaScript desenvolvida pelo Meta destinada à construção de interfaces de usuário baseadas em componentes reutilizáveis. Segundo Banks e Porcello (2020), essa arquitetura facilita o desenvolvimento de aplicações altamente interativas, reduzindo a complexidade de manutenção e promovendo melhor organização do código.

Para implementação do back-end foi utilizado o framework FastAPI, desenvolvido em Python e projetado para criação de APIs REST de alto desempenho. Conforme Ramírez (2023), o FastAPI apresenta excelente desempenho devido ao uso assíncrono do protocolo ASGI, além de fornecer documentação automática das APIs e validação de dados baseada em tipagem.

A comunicação entre front-end e back-end foi realizada utilizando APIs REST, padrão arquitetural proposto por Fielding (2000), fundamentado na troca de mensagens utilizando o protocolo HTTP e recursos identificados por Uniform Resource Identifiers (URI). Esse modelo favorece interoperabilidade entre diferentes tecnologias e amplia a flexibilidade para futuras integrações.

Segurança em Aplicações de Pagamentos Online

A segurança constitui um dos aspectos mais relevantes em aplicações que realizam autenticação de usuários e processamento de pagamentos digitais.

Segundo o PCI Security Standards Council (2022), aplicações que manipulam informações relacionadas a cartões de pagamento devem seguir requisitos mínimos de proteção de dados, incluindo criptografia, controle de acesso, autenticação forte e monitoramento contínuo.

Além disso, Pressman e Maxim (2020) destacam que sistemas web modernos devem incorporar mecanismos de autenticação, autorização e comunicação segura utilizando

protocolos criptográficos como Transport Layer Security (TLS), protegendo os dados transmitidos entre cliente e servidor.

Na arquitetura proposta, os mecanismos de autenticação foram implementados utilizando tokens de acesso e comunicação protegida por HTTPS. O armazenamento das informações sensíveis ocorre em serviços gerenciados da AWS, que oferecem recursos nativos de monitoramento, controle de acesso e alta disponibilidade.

Embora o presente estudo tenha como foco principal a arquitetura da aplicação, a adoção dessas práticas contribui significativamente para aumentar a confiabilidade do sistema e reduzir riscos associados ao tratamento de informações dos usuários.

Esta pesquisa caracteriza-se como **aplicada**, uma vez que objetiva desenvolver e analisar uma solução computacional destinada à resolução de um problema prático relacionado ao gerenciamento de agendamentos de serviços.

Quanto à abordagem, trata-se de uma pesquisa **qualitativa**, pois a análise concentrou-se nas características da arquitetura implementada, na integração entre os serviços de computação em nuvem e na avaliação funcional da aplicação, sem utilização de procedimentos estatísticos.

Do ponto de vista dos objetivos, a pesquisa é classificada como **exploratória e descritiva**. O caráter exploratório decorre da investigação sobre a aplicação integrada de diferentes serviços da Amazon Web Services (AWS) em uma plataforma de agendamento online. O caráter descritivo está relacionado à apresentação da arquitetura proposta, das tecnologias empregadas e dos resultados observados durante o desenvolvimento da solução.

Como procedimento técnico, adotou-se o **método de estudo de caso**, concentrando-se na implementação de uma aplicação web destinada ao gerenciamento de agendamentos, pagamentos e comunicação entre clientes e prestadores de serviços.

O levantamento inicial dos requisitos subsidiaram a definição das funcionalidades essenciais da plataforma, incluindo gerenciamento de agenda, notificações automáticas, autenticação de usuários, programa de fidelidade e integração com pagamentos digitais.

Após o desenvolvimento da aplicação foram realizados **testes funcionais internos**, utilizando as ferramentas **Postman**, para validação das APIs REST, e **Jest** e **React Testing Library**, para verificação do comportamento da interface desenvolvida em React.js. Esses testes tiveram como finalidade verificar o correto funcionamento das funcionalidades implementadas, identificar inconsistências e validar a integração entre os diferentes componentes da arquitetura.

A análise concentrou-se na avaliação qualitativa da arquitetura implementada, considerando aspectos relacionados à modularidade, escalabilidade, integração entre serviços, facilidade de manutenção e potencial de expansão da infraestrutura baseada em computação em nuvem. Por não envolver experimentação com seres humanos, coleta de dados sensíveis ou procedimentos que oferecessem riscos aos participantes, a pesquisa não demandou submissão ao Comitê de Ética em Pesquisa (CEP), conforme as orientações institucionais vigentes.

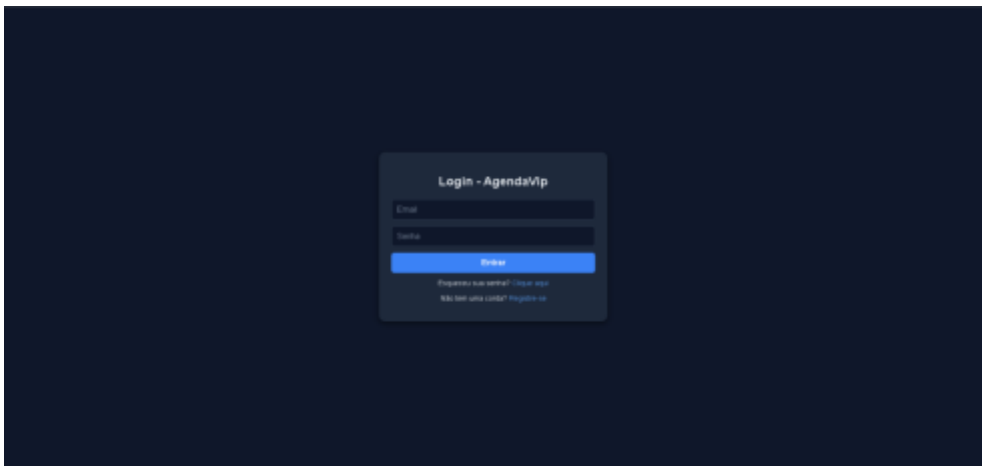
A arquitetura desenvolvida demonstrou que a utilização integrada dos serviços da Amazon Web Services (AWS) possibilitou a construção de uma aplicação web modular, escalável e preparada para futuras expansões. Diferentemente de soluções monolíticas tradicionais, a separação entre as camadas de apresentação, lógica de negócio, persistência de dados e serviços auxiliares permitiu maior independência entre os componentes, favorecendo a manutenção e a evolução do sistema. A Figura 1 apresenta a landing page da aplicação, responsável por fornecer informações iniciais aos usuários e direcioná-los para o processo de autenticação ou cadastro.

Figura 1 – Landing Page da Plataforma

Fonte: Os autores

A interface foi desenvolvida utilizando React.js e hospedada no Amazon S3, sendo distribuída pelo Amazon CloudFront. Essa configuração reduz a latência de acesso e aumenta a disponibilidade da aplicação, conforme recomendado pelo AWS Well-Architected Framework (AWS, 2023). Além disso, a separação entre o front-end e o back-end favorece a independência entre as equipes de desenvolvimento e facilita futuras atualizações da interface sem necessidade de alterações na infraestrutura do servidor. A Figura 2 apresenta a tela de autenticação da aplicação.

Figura 2 – Tela de Login

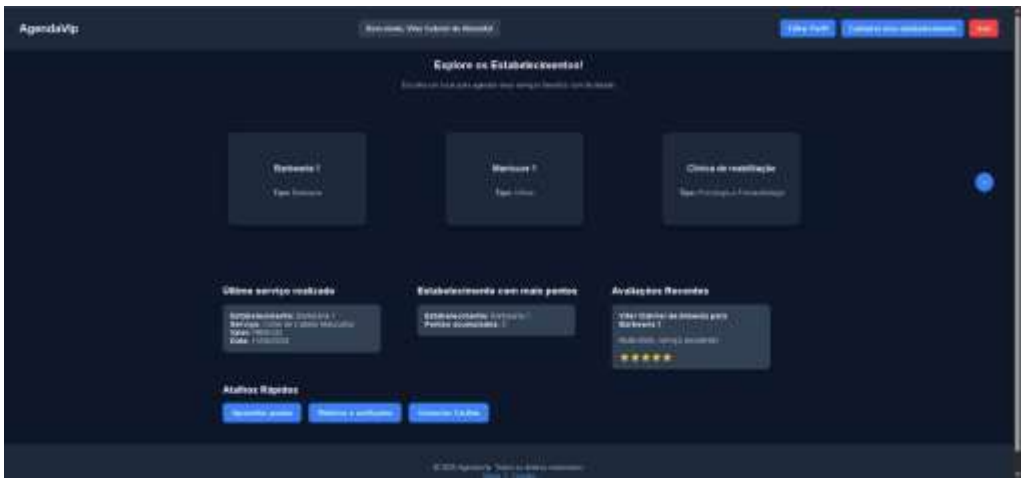


Fonte: Os autores

O processo de autenticação foi integrado ao Amazon SES para envio automático de mensagens relacionadas à recuperação de senha. Essa funcionalidade automatiza um processo que frequentemente exige intervenção manual em pequenos estabelecimentos, reduzindo o tempo de atendimento e melhorando a experiência do usuário.

Do ponto de vista arquitetural, a autenticação também representa um componente importante para a segurança da aplicação, restringindo o acesso às funcionalidades conforme o perfil de cada usuário. A Figura 3 apresenta o painel destinado aos clientes.

Figura 3 – Painel do Cliente



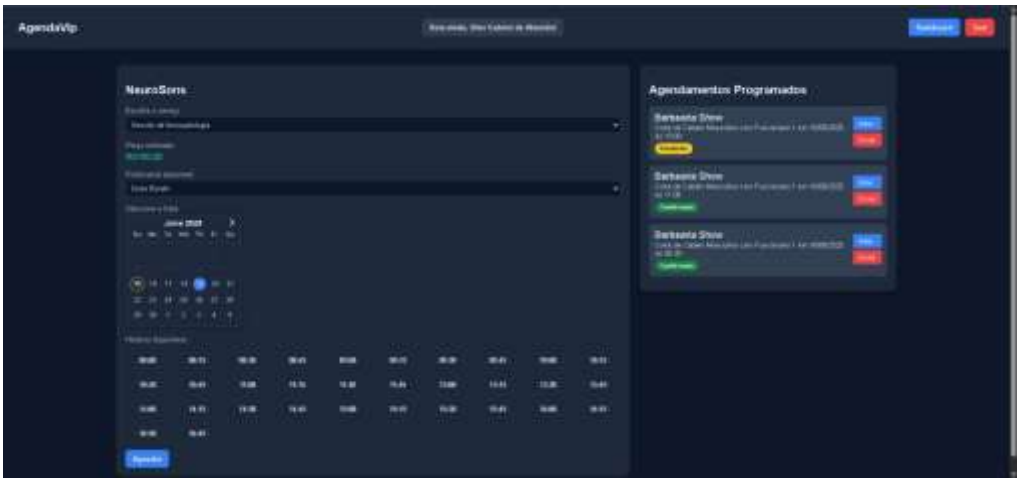
Fonte: Os autores

Esse módulo centraliza todas as operações realizadas pelos clientes, incluindo gerenciamento de perfil, consulta aos estabelecimentos cadastrados, histórico de serviços, programa de fidelidade e gerenciamento dos cartões utilizados para pagamento.

A organização dessas funcionalidades em uma única interface reduz a quantidade de etapas necessárias para realização dos serviços, contribuindo para melhoria da usabilidade.

Embora não tenham sido conduzidos testes formais de usabilidade, os testes funcionais internos demonstraram consistência no fluxo de navegação e ausência de falhas críticas durante a execução das operações. A Figura 4 apresenta o módulo responsável pelo agendamento.

Figura 4 – Tela de Agendamento



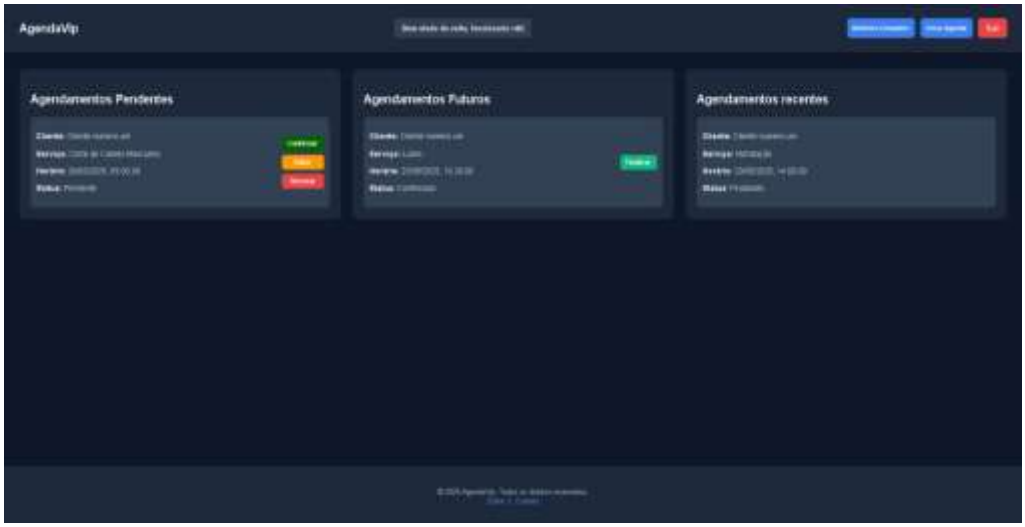
Fonte: Os autores

O processo de agendamento representa uma das funcionalidades mais importantes da plataforma. Durante os testes realizados, verificou-se que o sistema foi capaz de impedir conflitos de horários por meio da verificação automática da disponibilidade dos profissionais cadastrados.

Além disso, alterações ou cancelamentos realizados pelos usuários geram automaticamente notificações eletrônicas enviadas por meio do Amazon SES, reduzindo a necessidade de contato manual entre clientes e estabelecimentos.

Segundo Cucco (2018), sistemas que utilizam confirmações automáticas apresentam maior eficiência operacional e contribuem para redução da ocorrência de faltas sem aviso prévio (no-shows). A Figura 5 apresenta o painel utilizado pelos profissionais.

Figura 5 – Painel do Profissional



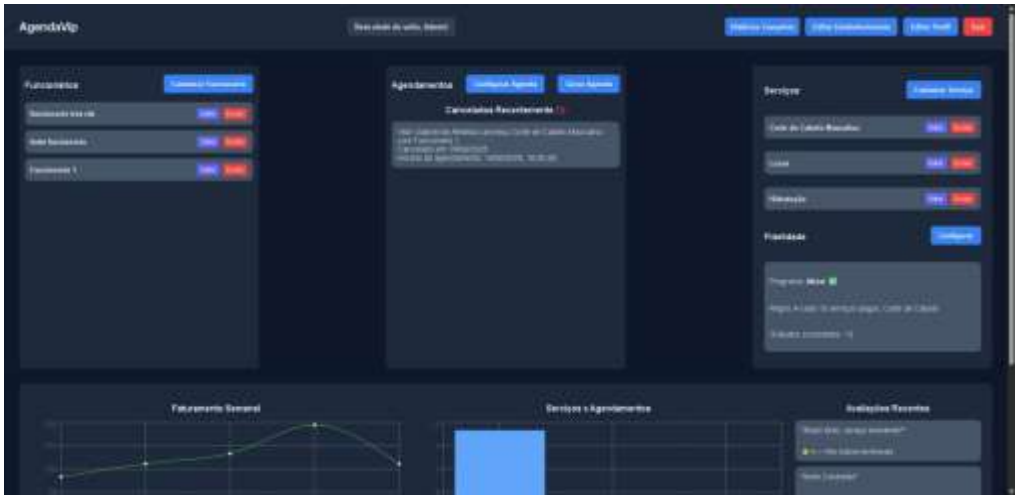
Fonte: Os autores

Esse módulo concentra funcionalidades relacionadas ao gerenciamento da agenda semanal, confirmação dos agendamentos, alterações de horários e conclusão dos atendimentos realizados.

Observou-se que a separação entre os perfis de cliente, profissional e administrador contribuiu para reduzir a complexidade operacional da aplicação, permitindo que cada usuário visualize apenas os recursos compatíveis com suas atribuições.

Outro aspecto relevante foi a automatização do processo de cobrança. Após a finalização do atendimento, o sistema registra automaticamente o pagamento correspondente, reduzindo procedimentos administrativos normalmente executados manualmente. A Figura 6 apresenta o painel administrativo da plataforma.

Figura 6 – Painel Administrativo



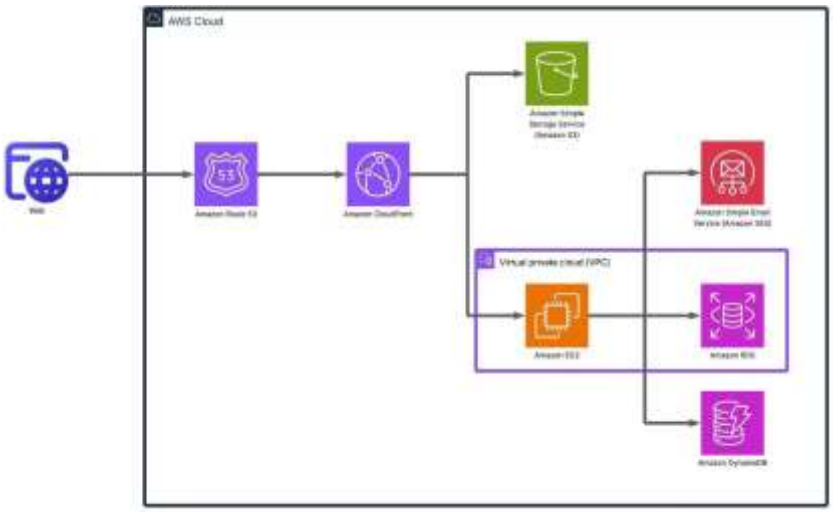
Fonte: Os autores

O ambiente administrativo concentra as funcionalidades de gerenciamento do estabelecimento, funcionários, serviços, programa de fidelidade, indicadores financeiros e acompanhamento dos agendamentos.

A existência desse módulo amplia o potencial de utilização da plataforma em pequenos negócios, permitindo maior controle das operações e fornecendo informações gerenciais relevantes para tomada de decisão.

Sob a perspectiva arquitetural, a centralização das funções administrativas em um único painel reduz redundâncias e facilita futuras ampliações da aplicação. A Figura 7 apresenta o diagrama da arquitetura implementada.

Figura 7 – Arquitetura da Aplicação



Fonte: Os autores

A infraestrutura foi organizada em ambiente de nuvem utilizando uma Virtual Private Cloud (VPC), contendo sub-redes públicas e privadas distribuídas entre diferentes zonas de disponibilidade.

O front-end foi hospedado no Amazon S3 e distribuído pelo CloudFront, enquanto o back-end foi implantado em instância Amazon EC2. O banco de dados relacional PostgreSQL permaneceu isolado em sub-rede privada por meio do Amazon RDS, aumentando a segurança da aplicação.

O armazenamento complementar realizado pelo DynamoDB permitiu reduzir consultas frequentes ao banco relacional, melhorando o desempenho das funcionalidades relacionadas ao programa de fidelidade e ao histórico de agendamentos.

Essa organização segue princípios descritos pelo AWS Well-Architected Framework (2023), especialmente nos pilares de confiabilidade, desempenho e segurança. A Figura 8 apresenta o Diagrama Entidade-Relacionamento da base de dados.

Figura 8 – Modelo Entidade-Relacionamento

Fonte: Os autores

A modelagem adotada contempla as entidades responsáveis pelo gerenciamento dos usuários, estabelecimentos, profissionais, serviços, pagamentos, avaliações, notificações e programas de fidelidade.

O modelo relacional foi normalizado visando reduzir redundâncias e preservar a integridade dos dados, permitindo expansão futura sem necessidade de alterações estruturais significativas.

A integração entre PostgreSQL e DynamoDB mostrou-se adequada para aplicações que exigem simultaneamente consistência transacional e consultas rápidas sobre informações específicas.

CONSIDERAÇÕES FINAIS

A crescente transformação digital dos pequenos negócios evidencia a necessidade de soluções tecnológicas capazes de automatizar processos operacionais, reduzir custos administrativos e melhorar a experiência dos usuários. Nesse contexto, este estudo analisou a

implementação de uma arquitetura escalável em computação em nuvem aplicada a uma plataforma web de agendamento de serviços e pagamentos online, utilizando recursos disponibilizados pela Amazon Web Services (AWS).

Diferentemente de uma abordagem voltada apenas ao desenvolvimento de software, o presente trabalho concentrou-se na análise da arquitetura implementada, demonstrando como a integração entre serviços gerenciados de computação em nuvem pode contribuir para o desenvolvimento de aplicações cloud-native com elevada modularidade, disponibilidade e facilidade de manutenção.

A utilização conjunta do Amazon EC2, Amazon S3, Amazon CloudFront, Amazon RDS, Amazon DynamoDB e Amazon Simple Email Service (SES) possibilitou a construção de uma infraestrutura organizada em componentes independentes, favorecendo a escalabilidade horizontal, a separação das responsabilidades entre os serviços e a automação de funcionalidades essenciais, como autenticação, notificações por e-mail, gerenciamento de agendas e integração com pagamentos digitais.

Os testes funcionais realizados durante o desenvolvimento demonstraram estabilidade da aplicação e funcionamento adequado das funcionalidades implementadas. A arquitetura proposta apresentou flexibilidade suficiente para permitir futuras ampliações sem necessidade de alterações estruturais significativas, característica considerada essencial em aplicações destinadas a pequenos e médios prestadores de serviços.

Sob a perspectiva científica, este trabalho contribui ao demonstrar a aplicabilidade prática dos princípios de arquitetura escalável em computação em nuvem na construção de sistemas de agendamento online, evidenciando como a utilização integrada dos serviços da AWS pode reduzir a complexidade de infraestrutura e facilitar a evolução tecnológica da aplicação.

Entretanto, algumas limitações devem ser consideradas. A pesquisa não contemplou testes quantitativos de desempenho envolvendo elevado número de usuários simultâneos, impossibilitando mensurar indicadores como tempo médio de resposta, utilização de recursos computacionais e comportamento da infraestrutura sob cargas intensivas. Da mesma forma, não foram conduzidos estudos formais de usabilidade envolvendo usuários finais, restringindo a avaliação da experiência de utilização da plataforma.

Como trabalhos futuros, recomenda-se a realização de testes de carga utilizando ferramentas específicas, como Apache JMeter e Locust, visando avaliar a capacidade de escalabilidade da infraestrutura em diferentes cenários de utilização. Também se propõe a

realização de pesquisas de usabilidade com usuários reais, utilizando métodos consolidados, como o System Usability Scale (SUS), além da ampliação da arquitetura para ambientes baseados em containers, Kubernetes e computação serverless, explorando serviços como Amazon ECS, Amazon EKS e AWS Lambda.

Conclui-se, portanto, que a arquitetura proposta constitui uma alternativa tecnicamente viável para aplicações de agendamento destinadas a pequenos negócios, oferecendo elevada capacidade de evolução, manutenção simplificada e potencial para adoção em diferentes segmentos de prestação de serviços.

REFERÊNCIAS

AMAZON WEB SERVICES. AWS Well-Architected Framework. Seattle: Amazon Web Services, 2023.

ARMBRUST, M. *et al*. A view of cloud computing. *Communications of the ACM*, New York, v. 53, n. 4, p. 50-58, 2010.

BANKS, A.; PORCELLO, E. *Learning React*: Modern Patterns for Developing React Apps. 2. ed. Sebastopol: O'Reilly Media, 2020.

BASS, L.; CLEMENTS, P.; KAZMAN, R. *Software Architecture in Practice*. 4. ed. Boston: Addison-Wesley, 2022.

CUCCO, R. S. Tecnologias aplicadas ao agendamento online de serviços: desafios e oportunidades. *Revista Brasileira de Gestão e Inovação,* v. 7, n. 2, p. 145-160, 2018.

ELMASRI, R.; NAVATHE, S. *Fundamentals of Database Systems*. 7. ed. Boston: Pearson, 2017.

FASTAPI. *FastAPI Documentation.* Disponível em: https://fastapi.tiangolo.com. Acesso em: 20 maio 2026.

FIELDING, R. T. *Architectural Styles and the Design of Network-based Software Architectures.* 2000. Tese (Doutorado em Ciência da Computação) – University of California, Irvine, 2000.

FOWLER, M. *Refactoring*: Improving the Design of Existing Code. 2. ed. Boston: Addison-Wesley, 2019.

MELL, P.; GRANCE, T. *The NIST Definition of Cloud Computing.* Gaithersburg: National Institute of Standards and Technology, 2011. (Special Publication 800-145).

META. *React Documentation*. Disponível em: https://react.dev. Acesso em: 20 maio 2026.

POSTGRESQL GLOBAL DEVELOPMENT GROUP. *PostgreSQL Documentation*. Disponível em: https://www.postgresql.org/docs. Acesso em: 20 maio 2026.

MICROSOFT. *REST API Guidelines*. Redmond, 2023.

NEWMAN, S. *Building Microservices*. 2. ed. Sebastopol: O'Reilly Media, 2021.

PCI SECURITY STANDARDS COUNCIL. *PCI DSS*: Payment Card Industry Data Security Standard Version 4.0. Wakefield, 2022.

PRESSMAN, R. S.; MAXIM, B. R. *Engenharia de Software:* uma abordagem profissional. 9. ed. Porto Alegre: AMGH, 2021.

SOMMERVILLE, I. *Engenharia de Software.* 10. ed. São Paulo: Pearson, 2019.