Instituto Federal de Educação, Ciência e Tecnologia de Goiás
Pró-Reitoria de Ensino
Câmpus Goiânia
Bacharelado em Sistemas de Informação

Davi Mendes Vilas Boa
Victor Hugo Costa e Silva

A aplicação de OutSystems no Desenvolvimento de um Software de Gestão de Barbearias

Goiânia, 2025.

TERMO DE AUTORIZAÇÃO PARA DISPONIBILIZAÇÃO NO REPOSITÓRIO DIGITAL DO IFG - ReDi IFG

Com base no disposto na Lei Federal nº 9.610/98, AUTORIZO o Instituto Federal de Educação, Ciência e Tecnologia de Goiás, a disponibilizar gratuitamente o documento no Repositório Digital (ReDi IFG), sem ressarcimento de direitos autorais, conforme permissão assinada abaixo, em formato digital para fins de leitura, download e impressão, a título de divulgação da produção técnico-científica no IFG.

Identificação da Produção Técnico-Científica

[ ]	Tese	[ ]	Artigo Científico
[ ]	Dissertação	[ ]	Capítulo de Livro
[ ]	Monografia – Especialização	[ ]	Livro
[X]	TCC - Graduação	[ ]	Trabalho Apresentado em Evento
[ ]	Produto Técnico e Educacional - Tipo: _____		

Nome Completo do Autor: Davi Mendes Vilas Boa
Matrícula: 20211011090046
Título do Trabalho: A Aplicação de OutSystems no Desenvolvimento de um Softwre de Gestão de Barbearias

Autorização - Marque uma das opções

1. (X) Autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso aberto);
2. ( ) Autorizo disponibilizar meu trabalho no Repositório Digital do IFG somente após a data ___/___/_____ (Embargo);
3. ( ) Não autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso restrito).

Ao indicar a opção **2 ou 3**, marque a justificativa:

( ) O documento está sujeito a registro de patente.
( ) O documento pode vir a ser publicado como livro, capítulo de livro ou artigo.
( ) Outra justificativa: _____

DECLARAÇÃO DE DISTRIBUIÇÃO NÃO-EXCLUSIVA

O/A referido/a autor/a declara que:

i. o documento é seu trabalho original, detém os direitos autorais da produção técnico-científica e não infringe os direitos de qualquer outra pessoa ou entidade;
ii. obteve autorização de quaisquer materiais inclusos no documento do qual não detém os direitos de autor/a, para conceder ao Instituto Federal de Educação, Ciência e Tecnologia de Goiás os direitos requeridos e que este material cujos direitos autorais são de terceiros, estão claramente identificados e reconhecidos no texto ou conteúdo do documento entregue;
iii. cumpriu quaisquer obrigações exigidas por contrato ou acordo, caso o documento entregue seja baseado em trabalho financiado ou apoiado por outra instituição que não o Instituto Federal de Educação, Ciência e Tecnologia de Goiás.

_____Goiânia_____, __23__/__04__/_2025_.
Local Data

Documento assinado digitalmente
gov.br
DAVI MENDES VILAS BOA
Data: 23/04/2025 12:22:18-0300
Verifique em https://validar.iti.gov.br

_____
Assinatura do Autor e/ou Detentor dos Direitos Autorais

TERMO DE AUTORIZAÇÃO PARA DISPONIBILIZAÇÃO NO REPOSITÓRIO DIGITAL DO IFG - ReDi IFG

Com base no disposto na Lei Federal nº 9.610/98, AUTORIZO o Instituto Federal de Educação, Ciência e Tecnologia de Goiás, a disponibilizar gratuitamente o documento no Repositório Digital (ReDi IFG), sem ressarcimento de direitos autorais, conforme permissão assinada abaixo, em formato digital para fins de leitura, download e impressão, a título de divulgação da produção técnico-científica no IFG.

Identificação da Produção Técnico-Científica

[ ] Tese	[ ] Artigo Científico
[ ] Dissertação	[ ] Capítulo de Livro
[ ] Monografia – Especialização	[ ] Livro
[X] TCC - Graduação	[ ] Trabalho Apresentado em Evento
[ ] Produto Técnico e Educacional - Tipo: _____	

Nome Completo do Autor: Victor Hugo Costa e Silva
Matrícula: 20211011090208
Título do Trabalho: A Aplicação de OutSystems no Desenvolvimento de um Softwre de Gestão de Barbearias

Autorização - Marque uma das opções

1. (X) Autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso aberto);
2. ( ) Autorizo disponibilizar meu trabalho no Repositório Digital do IFG somente após a data ___/___/_____ (Embargo);
3. ( ) Não autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso restrito).

Ao indicar a opção **2 ou 3**, marque a justificativa:

( ) O documento está sujeito a registro de patente.
( ) O documento pode vir a ser publicado como livro, capítulo de livro ou artigo.
( ) Outra justificativa: _____

DECLARAÇÃO DE DISTRIBUIÇÃO NÃO-EXCLUSIVA

O/A referido/a autor/a declara que:

i. o documento é seu trabalho original, detém os direitos autorais da produção técnico-científica e não infringe os direitos de qualquer outra pessoa ou entidade;
ii. obteve autorização de quaisquer materiais inclusos no documento do qual não detém os direitos de autor/a, para conceder ao Instituto Federal de Educação, Ciência e Tecnologia de Goiás os direitos requeridos e que este material cujos direitos autorais são de terceiros, estão claramente identificados e reconhecidos no texto ou conteúdo do documento entregue;
iii. cumpriu quaisquer obrigações exigidas por contrato ou acordo, caso o documento entregue seja baseado em trabalho financiado ou apoiado por outra instituição que não o Instituto Federal de Educação, Ciência e Tecnologia de Goiás.

_____Goiânia_____, __23__/__04__/_2025_.
Local Data

gov.br Documento assinado digitalmente
VICTOR HUGO COSTA E SILVA
Data: 23/04/2025 11:43:09-0300
Verifique em https://validar.iti.gov.br

_____
Assinatura do Autor e/ou Detentor dos Direitos Autorais

Instituto Federal de Educação, Ciência e Tecnologia de Goiás
Pró-Reitoria de Ensino
Câmpus Goiânia
Bacharelado em Sistemas de Informação

Davi Mendes Vilas Boa
Victor Hugo Costa e Silva

A aplicação de OutSystems no Desenvolvimento de um Software de Gestão de Barbearias

Trabalho de Conclusão de Curso apresentado à Coordenação do Curso de Bacharelado em Sistemas de Informação como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Orientadora: Profa. Me. Carina Calixto Ribeiro de Araújo

Goiânia, 2025.

V697a Vilas Boa, Davi Mendes.
A aplicação de OutSystems no desenvolvimento de um software de gestão de barbearias / Davi Mendes Vilas Boa, Victor Hugo Costa e Silva. – Goiânia: Instituto Federal de Educação, Ciência e Tecnologia de Goiás, Câmpus Goiânia, 2025.
56 f. : il.

Orientadora: Profa. Me. Carina Calixto Ribeiro de Araújo.

TCC (Trabalho de Conclusão de Curso) – Bacharelado em Sistemas de Informação, Instituto Federal de Educação, Ciência e Tecnologia de Goiás, Câmpus Goiânia.

1. Engenharia de software. 2. Plataforma OutSystems. 3. Barbearias - gestão. I. Silva, Victor Hugo Costa e. II. Araújo, Carina Calixto Ribeiro de (orientadora). III. Instituto Federal de Educação, Ciência e Tecnologia de Goiás, Câmpus Goiânia. IV. Título.

CDD 005.3

Ficha catalográfica elaborada pelo Bibliotecário Alisson de Sousa Belthodo Santos CRB1/ 2.266
Biblioteca Professor Jorge Félix de Souza,
Instituto Federal de Educação, Ciência e Tecnologia de Goiás, Câmpus Goiânia.

Instituto Federal de Educação, Ciência e Tecnologia de Goiás
Pró-Reitoria de Ensino
Câmpus Goiânia
Bacharelado em Sistemas de Informação

Davi Mendes Vilas Boa
Victor Hugo Costa e Silva

A aplicação de OutSystems no Desenvolvimento de um Software de Gestão de Barbearias

Trabalho de Conclusão de Curso defendido no Curso de Bacharelado em Sistemas de Informação como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação, aprovada em 20 de Março de 2025, pela Banca Examinadora constituída pelos professores:

Documento assinado digitalmente
gov.br
CARINA CALIXTO RIBEIRO DE ARAUJO
Data: 22/04/2025 13:02:55-0300
Verifique em https://validar.iti.gov.br

Profa. Me. Carina Calixto Ribeiro de Araújo
Departamento IV - IFG / Câmpus Goiânia
Presidente da Banca

Documento assinado digitalmente
gov.br
CARLOS AUGUSTO DA SILVA CABRAL
Data: 22/04/2025 14:36:37-0300
Verifique em https://validar.iti.gov.br

Prof. Me. Carlos Augusto da Silva Cabral
Departamento IV - IFG

Documento assinado digitalmente
gov.br
FREDERICO NOGUEIRA LEITE
Data: 22/04/2025 15:59:17-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Frederico Nogueira Leite
Departamento IV - IFG

Dedicatória

Agradeço ao meu avô Astério, por sempre acreditar na educação como caminho de crescimento e por ter investido em mim com tanto amor e confiança. Sua presença e incentivo foram fundamentais para que eu pudesse chegar até aqui.

À minha querida avó Ana Luiza, que esteve comigo na conquista da aprovação na faculdade, deixo uma lembrança carinhosa. Sua força, apoio e fé me acompanharam desde o início, e carrego sua memória com muita gratidão e saudade. –**Davi**

Dedico esse trabalho à minha avó Guiomar, "in Memoriam", que tanto me apoiou em todas as etapas de minha educação, e fez com que eu pudesse chegar até aqui. –**Victor**

Agradecimentos

À Prof[a] Me. Carina Calixto, pelos textos, referências e orientação valiosos.

À nossas famílias, amigos e companheiras, que tanto nos apoiaram nessa jornada de 4 anos. Obrigado pelo apoio e pela paciência.

Aos colegas de curso, que compartilharam nossos desafios.

Agradecemos ao corpo docente do curso de Sistemas de Informação pelo ensino fornecido.

Resumo

Título: A aplicação de OutSystems no Desenvolvimento de um Software de Gestão de Barbearias
Autores: Davi Mendes Vilas Boa e Victor Hugo Costa e Silva
Orientadora: Me. Carina Calixto Ribeiro de Araújo

A utilização da tecnologia da informação em diversas áreas tem possibilitado a oferta de serviços mais ágeis e precisos. Na área de prestação de serviços, como é o caso das barbearias, essa realidade não é diferente, uma vez que o mercado apresenta uma competitividade acirrada, demandando estratégias eficazes para fidelizar os clientes. Diante desse cenário, surge a necessidade de intervenção da tecnologia da informação, através do desenvolvimento de uma aplicação que não apenas auxilie as barbearias na busca pela fidelização dos clientes, mas também contribua para a gestão eficiente do negócio. O desafio desse trabalho foi criar uma aplicação customizável e flexível, capaz de atender a uma variedade de clientes sem a necessidade de grandes intervenções no código. O objetivo é oferecer um produto completo para o setor, fornecendo funcionalidades como agendamento de serviços, programas de fidelidade e gestão de promoções. Além disso, foram implementadas funcionalidades de back-office para auxiliar no controle do negócio e gestão dos recursos. Optou-se por utilizar a plataforma OutSystems em todo o processo de desenvolvimento devido à sua natureza de low-code, permitindo à equipe desenvolver uma aplicação complexa e customizável com poucos recursos de tempo e mão de obra. Esse projeto abordou etapas da engenharia de software como análise de requisitos, projeto de arquitetura de software e testagem. Após a conclusão do projeto, foi possível constatar que a aplicação foi bem-sucedida em automatizar a gestão do negócio, em conjunto a plataforma OutSystems

Palavras–chave

Fidelização de clientes; barbearias; OutSystems; low-code; gestão de negócios.

Abstract

Title: The Application of OutSystems in the Development of a Barbershop Management Software
Authors: Davi Mendes Vilas Boa and Victor Hugo Costa e Silva
Advisor: Me. Carina Calixto Ribeiro de Araújo

The use of information technology in various areas has enabled the provision of more agile and accurate services. In the area of service provision, such as barbershops, this reality is no different, since the market is fiercely competitive, demanding effective strategies to build customer loyalty. Given this scenario, the need for information technology intervention arises, through the development of an application that not only helps barbershops in their quest to build customer loyalty, but also contributes to efficient business management. The challenge of this work was to create a customizable and flexible application, capable of serving a variety of customers without the need for major interventions in the code. The goal is to offer a complete product for the sector, providing features such as service scheduling, loyalty programs and promotion management. In addition, back-office features were implemented to assist in business control and resource management. The OutSystems platform was chosen to be used throughout the development process due to its low-code nature, allowing the team to develop a complex and customizable application with limited time and labor resources. This project covered software engineering stages such as requirements analysis, software architecture design, and testing. After the project was completed, it was possible to confirm that the application was successful in automating business management, together with the OutSystems platform.

Keywords

Customer retention; barber shops; OutSystems; low-code; business management.

Lista de Figuras

Figura 1 – Tela do *Service Studio* 20
Figura 2 – Exemplo de uma arquitetura *On-Premises* 22
Figura 3 – Ambiente OutSystems 23
Figura 4 – Exemplo de um ciclo Scrum 25

Figura 5 – Ambientes do Sistema 32
Figura 6 – Aplicação CS 32
Figura 7 – Agregado 33
Figura 8 – Aplicação BL 33
Figura 9 – Aplicações de FE 34
Figura 10 – Tabela de base de dados com o tenant gerenciado manualmente 35
Figura 11 – Tabela com o tenant autogerenciável pela OutSystems 35
Figura 12 – Ação de criar empresa no módulo BackOffice_BL 36
Figura 13 – Tela de listagem de empresas do BackOffice 36
Figura 14 – Tela de listagem do clube de fidelidade 37
Figura 15 – Imagem da tela de login da aplicação de Administração da Barbearia 37
Figura 16 – Imagem da tela de cadastro de novos colaboradores 38
Figura 17 – Imagem da tela de login da aplicação do cliente final 40
Figura 18 – Variáveis de definição de cores no HTML 40
Figura 19 – função que faz o GET das cores configuradas no BackOffice .. 41
Figura 20 – Ação que realiza a troca das cores dinamicamente 41
Figura 21 – Alteração dinâmica do ícone da empresa e do nome 42
Figura 22 – Tela Inicial no desktop 43
Figura 23 – Tela inicial de agendamento a partir de um desktop 43
Figura 24 – Tela inicial a partir de um smartphone (Iphone XR) 44
Figura 25 – Tela inicial de agendamento a partir de um smartphone (Iphone XR) 45
Figura 26 – Tela de agendamento: escolha do horário 46
Figura 27 – Imagem da aplicação Cliente Final 47
Figura 28 – Imagem de demonstração de uma *role* aplicacional 48

Lista de Abreviaturas e Siglas

API	Application Programming Interface
APP	aplicação
AWS	Amazon Web Services
BL	Business Logic
CRUD	Create, Read, Update e Delete
CS	Core Services
CSS	Folhas de Estilo em Cascata
DOM	Modelo de Objeto de Documento
DSDM	Dynamic Systems Development Method
FE	Front End
HTML	Liguagem de Marcação de Hipertexto
UI	User Interface
XP	Extreme Programming

Sumário

1 Introdução 13
1.1 Objetivos 14
1.1.1 Objetivos Gerais 14
1.1.2 Objetivos Específicos 14
1.2 Justificativa 14
1.3 Resumo dos Próximos Capítulos 15

2 Referencial Teórico 17
2.1 A Plataforma OutSystems e suas Características 17
2.1.1 Arquitetura Aplicacional em OutSystems 17
2.1.2 Base de Dados 18
2.1.3 Desenvolvimento *Frontend* 18
2.1.4 Desenvolvimento *Backend* 19
2.1.5 Linguagem de programação 19
2.1.6 Infraestrutura 21
2.2 Engenharia de Requisitos 24
2.3 Medologias Ágeis de Desenvolvimento e Scrum 24
2.4 Aplicação Web Responsível 26
2.5 Métodos e Técnicas de Desenvolvimento 26
2.6 Ferramentas Utilizadas 27
2.7 Análise de Requisitos 27
2.8 Testes e Feedback 28

3 Resultados e Análise 29
3.1 Requisitos do Sistema GoBarber 29
3.1.1 Requisitos Funcionais 29
3.1.1.1 Ambiente BackOffice (Plataforma) 29
3.1.1.2 Ambiente Administração da Barbearia 30
3.1.1.3 Ambiente Interface do Cliente 30
3.1.2 Requisitos Não-Funcionais 30
3.2 Estruturação do Desenvolvimento 31
3.3 Implementação Incremental 34
3.4 Desenvolvimento das aplicações 35
3.5 Rotinas de Teste 48
3.6 Resultados Obtidos 49

4 Considerações Finais 52

Referências **54**

CAPÍTULO 1

Introdução

Como resultado do ganho cada vez maior de espaço do autocuidado masculino, o número de barbearias tem crescido consideravelmente, intensificando a concorrência nesse mercado (Vieira *et al.*, 2024). Diante desse cenário, a criação de ambientes agradáveis e promoções já não são suficientes para garantir a fidelização dos clientes. A modernização do setor trouxe uma gama de serviços além do corte de cabelo e barba, como limpeza de pele e depilação. O leque de serviços expandiu-se muito além da definição tradicional de "barbearia", com o objetivo de atrair e atender diferentes tipos de públicos. Essa modernização e a crescente interação das barbearias com a tecnologia destacam a necessidade de estratégias inovadoras para se destacar em um mercado tão competitivo.

A partir desse cenário, surge a oportunidade de implementar soluções tecnológicas para potencializar a fidelização de clientes e melhorar a gestão dos estabelecimentos. Nesse contexto, o uso da tecnologia não só atende às demandas de um mercado em constante evolução, como também proporciona uma gestão mais eficiente das operações diárias (Freitas, 2023). O desenvolvimento de uma aplicação dedicada pode aprimorar as estratégias de fidelização e otimizar a eficiência operacional, oferecendo mais comodidade aos clientes em um mundo cada vez mais conectado.

O presente trabalho busca desenvolver uma aplicação voltada para o aprimoramento dos serviços das barbearias, considerando sua relevância para o setor e as oportunidades que oferece tanto para as empresas quanto para os clientes. Contribuindo para um entendimento aprofundado do papel da tecnologia na modernização das barbearias e na construção de relacionamentos mais sólidos com os clientes, abrindo caminho para inovações e melhorias contínuas nesse mercado competitivo.

Para o desenvolvimento da aplicação proposta, foi utilizada a plataforma OutSystems, devio à agilidade que proporciona no desenvolvimento de soluções web. A escolha se deu pela capacidade da ferramenta em integrar, em um único ambiente, funcionalidades essenciais do processo de desenvolvimento de software.

Para projetar esse sistema, foi necessário seguir as várias etapas da engenha-

ria de software. De acordo com Sommerville (2019), foram selecionadas as seguintes etapas para construir esse projeto, desde a fase de planejamento ao produto final: análise e definição de requisitos, elaboração do projeto de software, desenvolvimento do sistema, fase de testes, fase de ajustes e correções, implantação do sistema e documentação.

1.1 Objetivos

1.1.1 Objetivos Gerais

- Desenvolver um software para barbearias que otimize a gestão de agendamentos, clientes, serviços e produtos, proporcionando maior eficiência operacional e melhor experiência para os usuários.

1.1.2 Objetivos Específicos

- Conceituar as necessidades processuais de uma barbearia, ou seja, os requisitos que serão utilizados no projeto de software;
- Elaborar um projeto de software adequado às necessidades da aplicação, onde sejam especificados os limites de orçamento, tempo, assim como os aspectos mais técnicos como design da interface de usuário, modelagem do banco de dados e modelagem das classes e objetos do código;
- Desenvolver o sistema: criar o código conforme o projeto, e implementar as principais funcionalidades como: cadastro de empresas, cadastro de funcionários, gestão de agendamentos, gestão de planos de benefícios e gestão de produtos;

1.2 Justificativa

O crescente interesse pelo autocuidado masculino e a intensificação da concorrência no setor de barbearias evidenciam a necessidade de soluções tecnológicas que promovam a fidelização de clientes e a melhoria na gestão dos serviços. No cenário atual, muitas barbearias ainda utilizam métodos manuais para controlar seus atendimentos — como agendamento por WhatsApp, cadernos físicos ou aplicativos genéricos — e realizam o controle financeiro e de estoque de forma informal, frequentemente com planilhas simples ou anotações.

Esses processos impactam diretamente a eficiência e a qualidade do atendimento. O gerenciamento de agendamentos, por exemplo, é frequentemente feito de forma descentralizada e sem controle de histórico, o que pode gerar conflitos de

horário, esquecimentos e desorganização. Da mesma forma, o controle financeiro e de comissões dos barbeiros costuma ser feito manualmente, aumentando o risco de erros e dificultando a análise de desempenho do negócio. Além disso, o estoque de produtos essenciais — como lâminas, cremes e loções — raramente é monitorado de forma precisa, o que pode causar falhas na operação e afetar a experiência do cliente.

Diante desse cenário, o desenvolvimento de um sistema específico para barbearias se justifica pela necessidade de automatizar esses processos. A proposta deste trabalho é oferecer uma aplicação capaz de gerenciar atendimentos, cadastrar clientes com preferências salvas, controlar produtos, gerenciar comissões e implementar programas de fidelidade. Além disso, espera-se oferecer integração com meios de pagamento e relatórios que apoiem a tomada de decisão dos gestores.

A plataforma OutSystems foi escolhida por permitir o desenvolvimento ágil de soluções customizadas com foco em usabilidade e escalabilidade. Sua abordagem *low-code* possibilita a criação de sistemas complexos com rapidez e menor necessidade de recursos técnicos, o que torna a ferramenta especialmente interessante para micro e pequenas empresas do setor de serviços.

Por fim, o projeto também representa uma oportunidade de aplicar conceitos acadêmicos em um contexto real, conectando teoria e prática. A criação deste sistema contribui para a formação técnica dos autores, ao mesmo tempo em que propõe uma solução funcional para um problema concreto, alinhado às necessidades de um mercado em transformação.

1.3 Resumo dos Próximos Capítulos

O capítulo 2 detalhará os métodos utilizados para a realização dos objetivos desse projeto. Serão apresentadas as ferramentas e tecnologias utilizadas para o desenvolvimento do sistema, como por exemplo:

- **Análise de Requisitos:** captação e listagem dos requisitos formais e não-formais do sistema, de acordo com as necessidades do cliente.
- **Projeto de Software:** decisão da arquitetura do sistema, hierarquia de classes, modelagem de dados, tecnologias utilizadas, entre outros.
- **Modelo de Desenvolvimento Ágil e Versionamento:** detalha-se como foram organizados os ciclos de entrega das funcionalidades do sistema, assim como a lógica de versionamento do sistema.
- **Testes e Feedback:** Detalha-se as rotinas de testes e captação de feedbacks para garantir a funcionalidade do sistema.

Já no capítulo 3 será detalhado como foi feito o desenvolvimento do sistema utili- zando a plataforma low-code da OutSystems, como a divisão das pastas, arquitetura dos serviços do sistema, exemplos de código, arquitetura, assim como descrições detalhadas das diferentes telas e funcionalidades do sistema.

Por fim, o capítulo 4 detalhará as conclusões finais do trabalho, assim como possíveis melhorias para projetos futuros.

CAPÍTULO 2

Referencial Teórico

No presente capítulo são abordados temas comuns no desenvolvimento de *software*, a fim de estabelecer a fundamentação teórica do projeto. Serão discutidos temas como: Arquitetura de Software, Engenharia de Requisitos, Metodologias ágeis, OutSystems, entre outros.

2.1 A Plataforma OutSystems e suas Características

2.1.1 Arquitetura Aplicacional em OutSystems

No desenvolvimento de um software, a arquitetura aplicacional é um fator fundamental no projeto. Ela define a estrutura e organização do sistema, determinando como as diferentes partes do software irão interagir entre si. Uma boa arquitetura garante não só um software mais robusto e escalável, mas também melhora a manutenibilidade e a reutilização de componentes.

Neste projeto foi adotado a arquitetura em camadas, uma abordagem recomendada pela OutSystems, que é um modelo que divide a aplicação em camadas distintas, cada uma com responsabilidades bem definidas, encapsulando as lógicas em módulos separados. Isso permite reutilizar esses módulos em diferentes partes do sistema e garantir uma manutenção com baixo impacto. A arquitetura é dividida em etapas e camadas claras, que facilitam o desenvolvimento, a manutenção e a escalabilidade do sistema que consistem nos seguintes processos:

- **Identificação de Conceitos:** Nesta etapa, são levantadas as necessidades funcionais (o que o sistema precisa fazer), não funcionais (requisitos como desempenho, segurança e escalabilidade) e de integração como APIs (Application Programming Interface (API), serviços externos ou outros sistemas.
- **Definição de Módulos:** Com base nos conceitos identificados, os módulos são definidos para implementar a lógica de negócio, integração e interface

do usuário. Esses módulos serão organizados nas camadas da arquitetura, respeitando o princípio de separação de responsabilidades.

- **Camadas da Arquitetura Outsystems:** No desenvolvimento com Outsystems, utiliza-se de uma arquitetura modular geralmente separada em três camadas principais: ***Core*****,** responsável pela lógica de dados e entidades fundamentais; ***Services*****,** onde se concentram as APIs e regras de negócio reutilizáveis; **(*User Interface (UI)*)** destinada à construção das interfaces gráficas para os usuários finais (Outsystems, 2024).

2.1.2 Base de Dados

A base de dados é um componente central de qualquer aplicação, pois é a partir dela que obtemos os dados necessários para manipular e processar em nosso software. De acordo com Silberschatz, Korth e Sudarshan (2019), os sistemas de banco de dados são essenciais para o gerenciamento eficiente de grandes volumes de dados, permitindo que as informações sejam organizadas e acessadas de maneira estruturada. Em OutSystems, o gerenciamento da base de dados é feito de forma interna e integrada à própria plataforma, permitindo que o desenvolvedor crie e gerencie entidades (tabelas) de forma visual e intuitiva dentro do ambiente de desenvolvimento.

A plataforma utiliza um modelo de banco de dados relacional, onde cada entidade representa um conjunto de dados estruturados, com atributos que correspondem às colunas da tabela. Esse modelo permite criar relações entre as tabelas de forma simples, promovendo a integridade e eficiência na manipulação dos dados.

O OutSystems oferece flexibilidade em relação ao local onde a base de dados é hospedada. As bases de dados podem ser hospedadas tanto em uma infraestrutura *on-premises* (local) quanto em ambientes *cloud* (hospedado em nuvem). Por padrão, quando o ambiente OutSystems *Cloud* é utilizado, as bases de dados são hospedadas em serviços de nuvem, que podem ser alteradas conforme a preferência do cliente, como Microsoft Azure ou Amazon Web Services (AWS) (OutSystems, 2024).

Para o esse projeto, que tem fins acadêmicos, utilizou-se a base de dados padrão oferecida pela OutSystems, hospedada em serviços de nuvem.

2.1.3 Desenvolvimento *Frontend*

No âmbito do desenvolvimento de software, o *frontend* contempla a interface gráfica do usuário, assim como todos os elementos de design que atribuem a experiência do usuário, incluindo elementos como menus de navegação, botões, imagens, gráficos, entre outros (AWS, 2023).

As tecnologias mais utilizadas no desenvolvimento *frontend* são:

- **Liguagem de Marcação de Hipertexto (HTML):** é linguagem de marcação de texto, utilizada como a base de uma página web, nomeando todos os elementos do Modelo de Objeto de Documento (DOM), para que outras tecnologias possam interagir com tais elementos.
- **Folhas de Estilo em Cascata (CSS):** linguagem de estilo que interage com o DOM para inserir elementos de visuais de design, ou estilos.
- **Javascript:** linguagem de programação padrão interpretada pelos navegadores de internet. Permite ao desenvolvedor *frontend* criar elementos interativos na interface, assim como permite interagir com a base de dados da aplicação de forma dinâmica.

2.1.4 Desenvolvimento *Backend*

No âmbito do desenvolvimento de software, o *backend* contempla a lógica programacional da aplicação, as regras de negócio, a gestão de banco de dados, entre outros fatores usados no lado do servidor da aplicação (AWS, 2023). As principais tecnologias utilizadas no backend são: linguagem de programação (OutSystems com C#, linguagem de banco de dados (MySQL), API's de serviços consumidos, protocolos de transferência de dados (FTP, FTPS), protocolos de autentificação, entre outros. Também é no *backend* onde toda a parte de modelagem do software é feita por meio da linguagem de programação.

O *backend* contempla também as verificações e autentificações dos dados dos usuários, e por isso é necessário utilizar práticas de segurança para impedir vulnerabilidades e invasões. É no *backend* onde são implementadas as rotinas de testes, como testes unitários, testes de componentes e testes de integração (AWS, 2023).

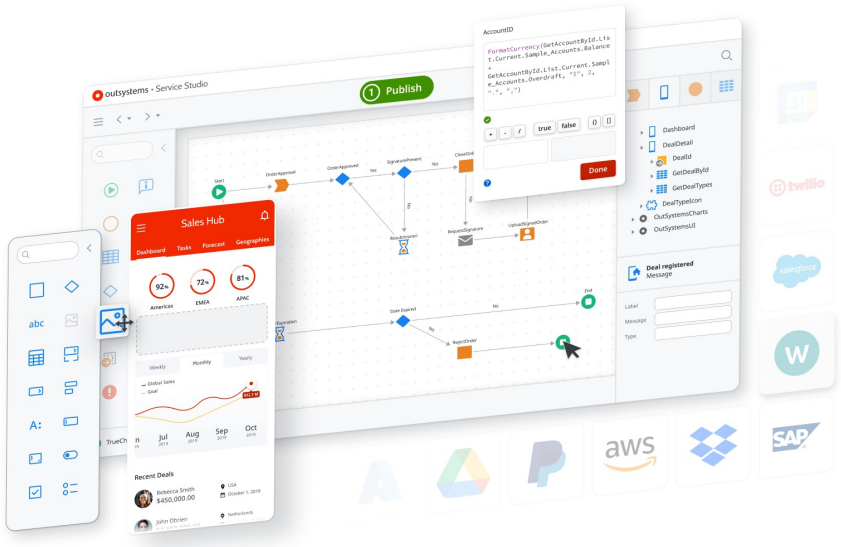
2.1.5 Linguagem de programação

As linguagens de programação são conjuntos de comandos com diferentes sintaxes que, ao serem interpretados ou compilados, são transformados em instruções que a máquina pode entender. Praticamente todas as aplicações possuem uma ou mais linguagens de programação em seu desenvolvimento, e isso não é diferente no OutSystems.

No OutSystems, utiliza-se a abordagem low-code, que, em tradução literal, significa "baixo código". Isso quer dizer que, ao desenvolver uma aplicação usando a plataforma OutSystems, o desenvolvedor terá pouco contato direto com linguagens de programação. A plataforma abstrai grande parte da complexidade técnica e

oferece o *Service Studio*, um ambiente visual com conectores lógicos e componentes reutilizáveis. Conforme a Figura 1, o ambiente do Service Studio traz um fluxo de desenvolvimento visual. A partir do conhecimento da regra de negócio, são construídos fluxos lógicos de forma visual, sem a necessidade de escrever código extensivo manualmente.

Figura 1 – *Tela do Service Studio*



Fonte: Outsystems, 2024.

No entanto, essa abstração não isenta completamente de utilizar linguagens de programação. Para a criação de componentes personalizados ou na integração com sistemas externos, pode ser necessário utilizar linguagens de programação tradicionais (como Python, JavaScript, etc) para criar soluções específicas que ainda não existem na plataforma.

Embora a plataforma abstraia as linguagens de programações, por de trás de toda a aplicação, a plataforma converte as lógicas e fluxos criados em código de várias linguagens (Outsystems, 2024):

- O backend é gerado em C# /.NET.
- O frontend utiliza JavaScript, HTML e CSS.
- Para interagir com a base de dados, a plataforma utiliza SQL.

Portanto, mesmo que não haja contato direto com esse código durante o desenvolvimento da aplicação, é importante ter ciência de que ele está sendo gerado e executado como parte do processo. O OutSystems não reinventa a roda, mas oferece uma forma mais clara e objetiva de desenvolver aplicações, tornando o processo mais ágil e acessível.

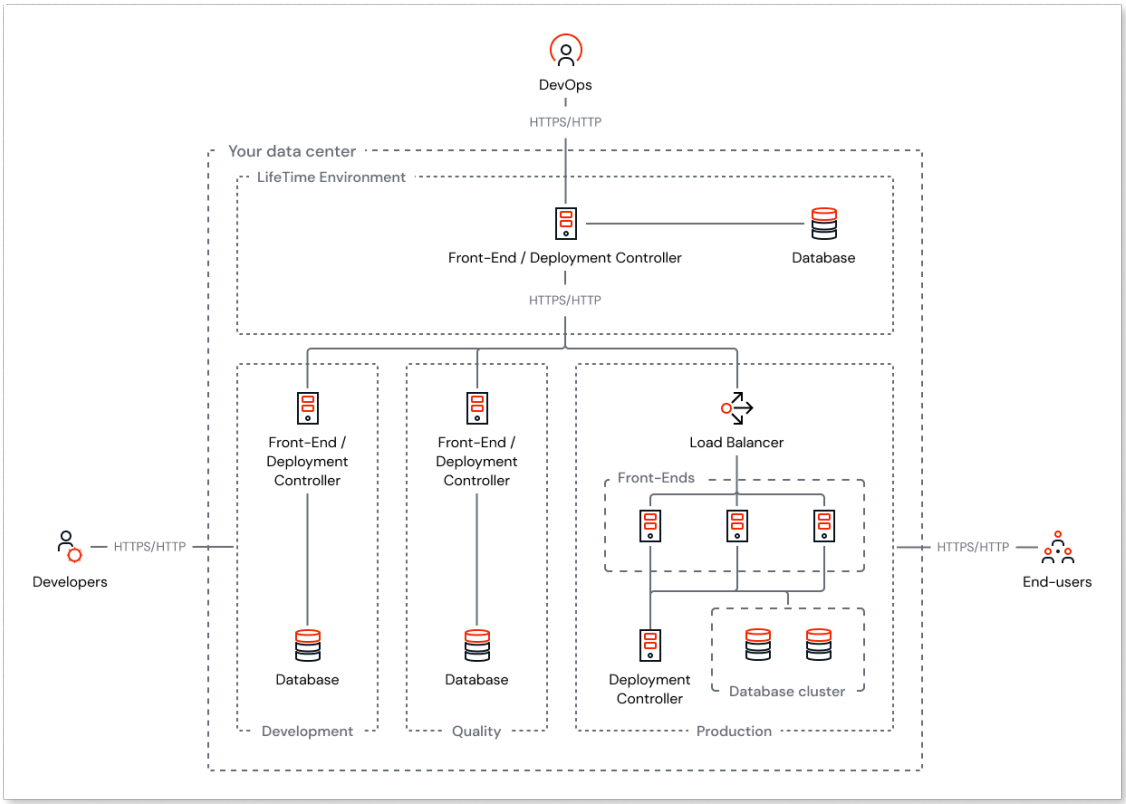
2.1.6 Infraestrutura

Em geral, a infraestrutura de um software é composta por servidores, que são máquinas com grande capacidade de processamento, capazes de executar diversas atividades em paralelo. Toda solicitação feita por um usuário precisa ser processada por um servidor. Além disso, a infraestrutura inclui serviços de segurança, bancos de dados, rede, entre outros componentes essenciais.

Existem três tipos principais de infraestrutura que podemos utilizar:

- **On-Premises (Local):** Nessa infraestrutura, todo o sistema de servidores e redes é montado e gerenciado localmente, seja na sede da empresa ou em um local específico, conhecido como datacenter (conforme demonstrado na Figura 2). Embora essa infraestrutura possa ser exposta na internet, todo o gerenciamento, manutenção e segurança são feitos internamente pela própria empresa.
- **Cloud (Nuvem):** A infraestrutura em nuvem consiste em contratar serviços de provedores de computação em nuvem (como AWS, Azure, Google Cloud, entre outros). Esses provedores disponibilizam parte de seus servidores para uso, permitindo que você pague apenas pelo que utilizar, sem os custos de aquisição e manutenção de hardware. Esses servidores podem ser dedicados a uma única empresa ou compartilhados com outras, mantendo sempre a segurança e sigilo dos dados.
- **Híbrida:** A infraestrutura híbrida combina o melhor dos dois mundos: parte dos serviços é mantida localmente (on-premises) e outra parte é contratada de provedores de nuvem. Isso permite que a empresa tenha maior controle sobre alguns aspectos críticos da sua infraestrutura, enquanto aproveita a escalabilidade e flexibilidade oferecidas pela nuvem para outros serviços.

Figura 2 – *Exemplo de uma arquitetura On-Premises*



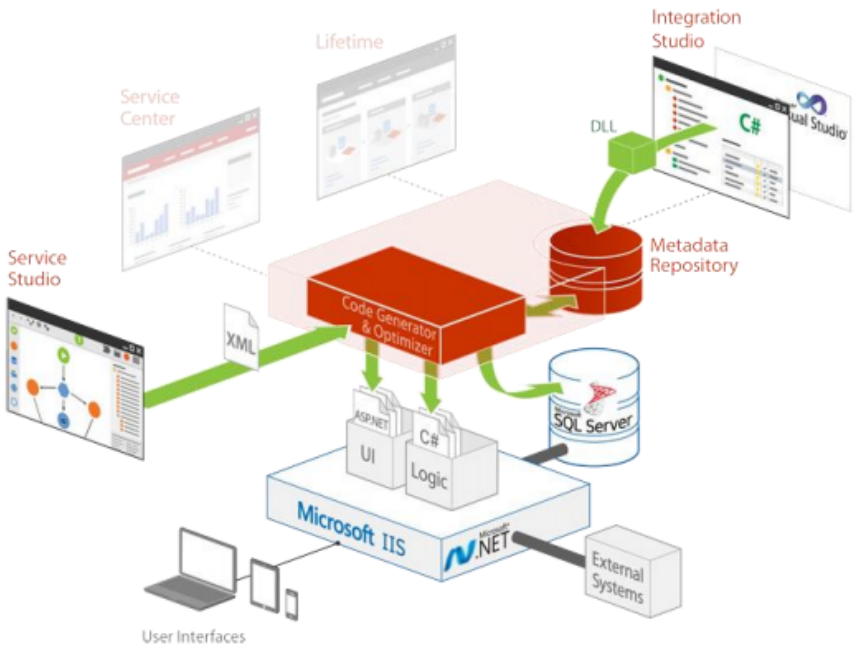
Fonte: (OutSytems, 2024).

A plataforma oferece suporte a qualquer uma das três infraestruturas mencionadas acima. Porém, por padrão, ao contratar uma solução completa da OutSystems, a infraestrutura utilizada é em nuvem. Nesse caso, não é necessário se preocupar em montar ou configurar servidores, pois toda essa parte já é gerenciada pela própria OutSystems, entregando todo o ecossistema necessário para desenvolver e manter uma aplicação (Figura 3):

- *LifeTime:* é a plataforma de gerenciamento centralizado da OutSystems. Ele permite a administração de aplicações ao longo dos diferentes ambientes (Desenvolvimento, Qualidade, Produção, etc.), possibilitando o controle de versões, permissões de acesso e monitoramento da infraestrutura. Além disso, é utilizado para orquestrar deploys entre os ambientes de forma segura e controlada.
- *Service Studio:* é a principal ferramenta de desenvolvimento da OutSystems. É um ambiente de desenvolvimento visual (IDE) onde os programadores criam aplicações *web* e *mobile*, definem a lógica de negócio, interfaces de usuário e modelam o banco de dados. Ele permite o desenvolvimento de forma low-code, acelerando a criação de soluções empresariais.

- *Service Center:* é a plataforma de administração das aplicações e infraestrutura na OutSystems. Ele permite: gerenciar logs e erros das aplicações, realizar parametrizações e configurações, administrar módulos e dependências das aplicações, monitorar o histórico de alterações e implantações,
- *Integration Studio:* é a ferramenta utilizada para criar extensões e integrações personalizadas, principalmente quando há necessidade de lógica avançada que não pode ser implementada de forma nativa na OutSystems . Com ele, é possível: desenvolver módulos de integração em .NET e Java, criar ações e componentes que não podem ser implementados diretamente no Service Studio, expor e consumir APIs de terceiros e trabalhar com conectores e bases de dados externas.
- *Base de Dados:* é responsável por armazenar não apenas os dados das aplicações, mas também informações críticas do próprio ambiente OutSystems, como: estruturas de dados modeladas no Service Studio, logs e auditoria do sistema, configurações e metadados do ambiente, informações sobre permissões e versões dos aplicativos.

Figura 3 – *Ambiente OutSystems*



Fonte: (OutSytems, 2024).

Ainda assim, os usuários têm a opção de adicionar camadas extras de segurança ou realizar outras personalizações, conforme as necessidades específicas do negócio.

Neste projeto foram adotados os serviços de infraestrutura padrão já fornecidos pela plataforma. Dessa forma, nossa preocupação será focada apenas nas regras de negócio e no desenvolvimento da aplicação, sem a necessidade de gerenciar infraestrutura.

2.2 Engenharia de Requisitos

Na Engenharia de Software, os requisitos são a descrição dos serviços e funcionalidades que serão desempenhados no sistema de informação (Sommerville, 2019). Os requisitos são uma abstração das necessidades expressas pelas partes interessadas do software, e dita como serão estruturados a arquitetura e a modelagem desse sistema e até fornece uma estimativa inicial da viabilidade do mesmo. O campo dedicado a identificar, gerenciar e documentar os requisitos de um sistema é chamado de Engenharia de Requisitos.

A Engenharia de Requisitos é um processo que envolve a definição dos serviços que o sistema deve fornecer e as restrições sob as quais ele deve operar. É uma fase crucial no desenvolvimento de software, pois estabelece as bases para o design e a implementação do sistema.

De acordo com Sommerville (2019), um processo de engenharia de requisitos abrange as seguintes etapas: identificar as partes interessadas do sistema (*stakeholders*), captação e análise de requisitos (normalmente obtidos em uma conversa inicial com o patrocinador do projeto), descrição dos requisitos, validação dos requisitos (validados com o patrocinador e a equipe de desenvolvimento) e, por fim, o gerenciamento desses requisitos (onde é fiscalizado e documentado se os requisitos estão sendo atendidos ou alterados).

2.3 Medologias Ágeis de Desenvolvimento e Scrum

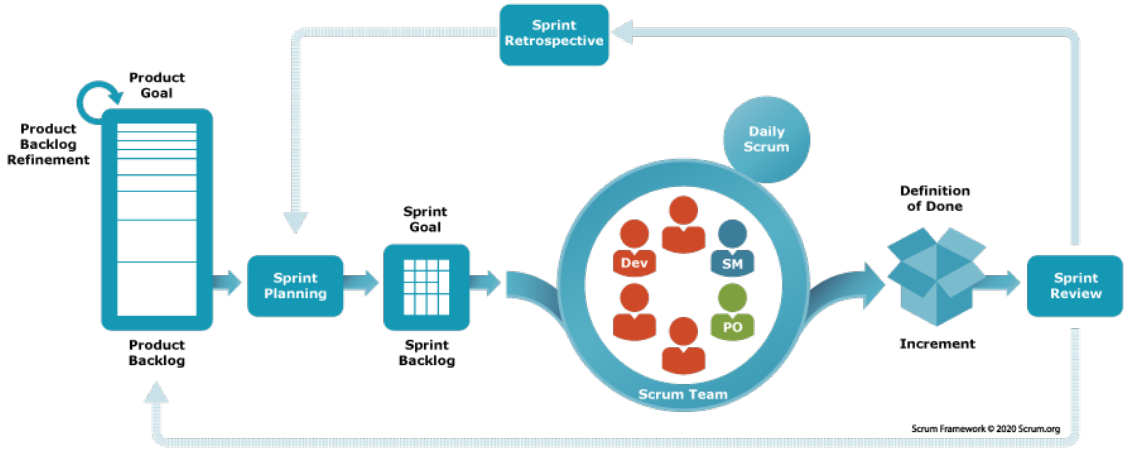
O rápido crescimento do uso de sistemas de informação no ambiente corporativo desde o final dos anos 1990 trouxe a necessidade de se reformular os antigos modelos de desenvolvimento de software, como cascata (*waterfall*), espiral e incremental (Sommerville, 2019). Esses modelos são caracterizados pelo foco em seguir rigidamente um plano pré-definido, uma estrutura rígida com fluxo unidirecional de trabalho e a pouca participação do cliente durante o processo de desenvolvimento. Essa rigidez se mostrou ineficaz em um ambiente cada vez mais

informatizado e globalizado, onde sistemas necessitam serem entregues antes que seus requisitos e serviços se tornem obsoletos (Sommerville, 2019).

A partir desse contexto nasceram novos modelos de desenvolvimento de software, implementados utilizando o modelo incremental, onde o projeto de software é separado em ciclos de desenvolvimento. Esses novos modelos, denominados "Modelos Ágeis", são caracterizados pela prioridade em entregar um produto funcional (sem tanta documentação) o quanto antes, a participação mais ativa dos clientes e a velocidade para responder às mudanças no mercado e nos requisitos do sistema (Highsmith, 2001). São exemplos dessas metodologias: Scrum, Extreme Programming (XP), Kanban e Dynamic Systems Development Method (DSDM). O presente trabalho utilizou especificamente a metodologia SCRUM.

O SCRUM é um *framework* de desenvolvimento de produtos complexos, onde as etapas do projeto são divididas em pequenos ciclos chamados sprints. Esse modelo tem uma abordagem cíclica e incremental, onde o feedback das partes interessadas é levado em conta em todas as etapas do desenvolvimento (Santos, 2022). A Figura 4 ilustra um ciclo Scrum genérico.

Figura 4 – *Exemplo de um ciclo Scrum*



Fonte: (Scrum.org, 2020).

Os *sprints* são ciclos de trabalho de duração variável, geralmente entre 1 e 4 semanas (Santos, 2022). Esses ciclos são contínuos, iniciando um novo logo após o término do anterior. O Scrum define três principais papéis na equipe: *Product Owner*, responsável por garantir que cada sprint entregue o máximo de valor ao produto, além de organizar e priorizar o backlog; *Scrum Master*, que auxilia a equipe na aplicação correta do Scrum, garantindo que todos compreendam e sigam suas regras e valores, visando maximizar o valor do projeto; e Time de Desenvolvimento

(*developers*), grupo auto-organizado que, a cada sprint, entrega um incremento do produto, baseado nas tarefas definidas no backlog.

Santos define que cada sprint possui uma série de reuniões onde são discutidos o andamento do projeto, sendo elas: *Dailies* (reuniões diárias de até 15 minutos), *Sprint Planning* (feita no início da sprint para planejar o que será realizado na mesma) e o *Sprint Review* (feita ao fim do sprint para discutir os resultados obtidos, o progresso feito e ouvir feedback de possíveis melhorias no processo). Ao final dessas reuniões são atualizados dois artefatos de gerenciamento: o *Product Backlog*, que lista todas as tarefas necessárias e restantes para o cumprimento do objetivo do projeto; e o *Sprint Backlog*, que lista todas as tarefas priorizadas pelo *Product Owner* para serem realizadas no sprint.

2.4 Aplicação Web Responsível

Hoje em dia, é comum que empresas ofereçam serviços ou produtos por meio de soluções tecnológicas. Websites são geralmente usados para esse fim, mas em sistemas que exigem praticidade ou o uso de recursos do celular, como câmera e Bluetooth, as aplicações móveis, ou *aplicação (APP)*, são preferidas (Santos, 2022).

De acordo com Santos (2022), no âmbito do desenvolvimento de apps móveis, é crucial considerar limitações como tamanho de tela, bateria, memória e conectividade. Além disso, Android e iOS, desenvolvidos pela Google e Apple respectivamente, dominam o mercado com cerca de 99% dos dispositivos globais, o que direciona o foco do desenvolvimento para esses sistemas.

Nesse enfoque, o design responsivo (ou design escalável) para web apps se trata de uma solução para adequar websites às limitações das telas de dispositivos móveis, caracterizado pela flexibilidade das páginas web e das interfaces dos apps pensado de forma que se atenda o maior número possível de tamanhos de tela (Krug, 2014). De acordo com Krug (2014), quem opta por utilizar esse método de desenvolvimento deve ter em mente algumas boas práticas, tais como: manter a consistência entre versões mobile e web, permitir o uso de zoom nas páginas e o mais importante: otimizar o desempenho do site para diminuir processo e evitar lentidão ao utilizar o *web app*.

2.5 Métodos e Técnicas de Desenvolvimento

Para o desenvolvimento do sistema, foi adotada a metodologia ágil Scrum, caracterizada por ciclos de desenvolvimento curtos e entregas incrementais, garantindo a adaptabilidade às mudanças nas necessidades dos usuários (Santos, 2022).

Segundo Sutherland e Schwaber (2017), o Scrum permite uma comunicação eficiente entre a equipe de desenvolvimento e os *stakeholders*, assegurando que os requisitos sejam ajustados conforme os feedbacks recebidos.

O uso do Scrum neste projeto incluiu os seguintes elementos:

- ***Sprints*:** ciclos de desenvolvimento com duração de duas semanas.
- **Reuniões diárias (*Dailies*):** com duração de até 15 minutos para discutir o progresso das tarefas.
- ***Sprint Planning*:** planejamento do que será desenvolvido em cada ciclo.
- ***Sprint Review*:** revisão dos resultados obtidos para obter feedback dos stakeholders.
- ***Sprint Retrospective*:** análise do que funcionou bem e do que pode ser melhorado para os próximos ciclos.

2.6 Ferramentas Utilizadas

A plataforma Service Studio, da Outsystems, foi escolhida como principal editor de código para desenvolver o projeto. Ela permite desenvolver as camadas de controle, de serviços e de interface de usuário de forma centralizada, além de permitir o uso de links de teste para a testar a funcionalidade da aplicação.

O Service Studio também foi utilizado para a modelagem do banco de dados da aplicação, assim como para gerir serviços como o envio de e-mails e notificações.

O app de mensagens Whatsapp também foi utilizado para a comunicação da equipe de desenvolvimento, principalmente para fins de atribuir tarefas e de marcar reuniões remotas.

O app de videoconferências Google Meet também foi a principal plataforma de reunião utilizada nas evoluções do projeto.

2.7 Análise de Requisitos

A Engenharia de Requisitos desempenhou um papel crucial na definição das funcionalidades do sistema. Conforme Sommerville (2019), a análise de requisitos envolve a captação, documentação, validação e gerenciamento das necessidades dos usuários. Nesse sentido, foram identificados requisitos funcionais, como agendamento de serviços, gestão de clientes e controle financeiro, além de requisitos não funcionais, como segurança, desempenho e escalabilidade.

2.8 Testes e Feedback

Para garantir a qualidade do sistema, foram realizados testes unitários e funcionais conforme descrito por (Sommerville, 2019). A ferramenta *OutSystems Test Framework* foi utilizada para automatizar testes unitários, verificando a correta execução de cada módulo de forma isolada. Além disso, foram conduzidos testes de usabilidade pela equipe de desenvolvimento para validar a experiência de uso (Krug, 2014).

Os feedbacks recebidos durante as sessões de teste foram registrados e utilizados para aprimorar a interface e corrigir inconsistências funcionais.

CAPÍTULO 3

Resultados e Análise

3.1 Requisitos do Sistema GoBarber

Uma barbearia enfrenta diversas necessidades operacionais que podem ser solucionadas por um sistema de software eficiente. Um dos principais desafios é o gerenciamento de agendamentos, pois a organização dos horários de atendimento influencia diretamente a experiência do cliente e a produtividade dos barbeiros. Muitos estabelecimentos ainda dependem de anotações manuais ou aplicativos genéricos, o que pode levar a conflitos de horário, atrasos e esquecimentos.

Um sistema de software pode oferecer um sistema de agendamento online, permitindo que os clientes escolham horários disponíveis, recebam lembretes automáticos e até façam cancelamentos ou reagendamentos de forma prática. Isso melhora a organização da barbearia e reduz o tempo ocioso dos profissionais.

Abaixo seguem os requisitos do sistema desse trabalho, formulados com base em Sommerville (2019), após uma conversa com o proprietário da barbearia:

3.1.1 Requisitos Funcionais

- **F01** - A Aplicação deve habilitar gerenciamento de várias barbearias (*multi-tenant*)
- **F02** - A aplicação deverá ter três ambientes, sendo elas:

3.1.1.1 Ambiente BackOffice (Plataforma)

- **F03** - Gestão de Novos Clientes (barbearias)
- **F04** - Gerenciamento de Barbearias
- **F05** - Gerenciamento de Serviços disponíveis (utiliza planos) (do barbeiro com nossa plataforma)
- **F06** - Gestão de Usuários Admin
- **F07** - Cadastro da empresa é feito pelo “user” admin
- **F08** - Contratação por WhatsApp, de forma pessoal

3.1.1.2 Ambiente Administração da Barbearia

- **F09** - Gestão de Funcionários
- **F10** - Gestão dos Serviços
- **F11** - Gestão dos Produtos
- **F12** - Gerenciamento dos atendimentos (também marcar se for pago)
- **F13** - Gerenciamento de programas de benefícios (benefícios, cartão fidelidade ou planos)
- **F14** – O sistema deverá permitir gerenciar o programa de benefícios, assim como gerenciar os prêmios e os pontos.
- **F15** – O sistema deverá permitir ao dono da barbearia cadastrar um plano de benefícios para sua barbearia. Esse plano (um clube, por exemplo) poderá variar entre serviços ilimitados ou uma quantidade limitada de serviços mensais. Os clientes assinantes poderão ganhar mais pontos, caso o dono da barbearia deseje. O gerenciamento do pagamento dos planos deverá ser feito por fora do sistema. O proprietário poderá também escolher produtos com descontos para assinantes.

3.1.1.3 Ambiente Interface do Cliente

- **F16** - Poder agendar/editar/cancelar serviços
- **F17** – Poder consultar seu plano
- **F18** – Poder solicitar a adesão de um plano
- **F19** - Poder Visualizar produtos disponibilizados pela barbearia
- **F20** - Poder acompanhar seus agendamentos
- **F21** – Poder acompanhar seu progresso como membro da barbearia
- **F22** - Só poderá agendar 1 serviço do mesmo tipo por vez.
- **F23** – O sistema deverá ter funcionalidade de feedback do cliente. Ele poderá avaliar o serviço, o ambiente e o atendimento. O cliente poderá ganhar pontos de benefícios com o feedback.
- **F24** – O cliente poderá acompanhar seus benefícios e seus pontos.

3.1.2 Requisitos Não-Funcionais

- **NF1** – Desempenho: O sistema deverá ser capaz de suportar um grande número de usuários simultâneos, de 400 a 600 simultâneos
- **NF2** - Desempenho: O sistema deverá ser capaz de realizar até 1000 agendamentos por dia sem perda significativa de desempenho
- **NF3** – Usabilidade: A interface deverá ser intuitiva, permitindo ao cliente realizar um agendamento em menos de 5 cliques.

- **NF4** – Segurança: O sistema deverá garantir a proteção de dados pessoais dos clientes e funcionários, cumprindo com leis de proteção de dados como a LGPD (Lei Geral de Proteção de Dados).
- **NF5** – Segurança: Todos os dados sensíveis (como senhas e informações de pagamento) deverão ser criptografados.
- **NF6** – Disponibilidade: O sistema deverá estar disponível 99,9% do tempo, com manutenção agendada em horários de baixo movimento.
- **NF7** – Portabilidade: O software deverá ser compatível com diferentes plataformas e dispositivos, rodando de forma eficaz em navegadores web e sistemas operacionais móveis como Android e iOS.
- **NF8** – Escalabilidade: O sistema deverá ser escalável, permitindo a adição de novos recursos (como integração com sistemas de marketing ou e-commerce) sem a necessidade de grandes alterações na infraestrutura.

3.2 Estruturação do Desenvolvimento

Com base nos requisitos levantados, ficou clara a necessidade utilizar uma abordagem de múltiplos ambientes para atender as necessidades do cliente.

Conforme ilustrado na Figura 5, o sistema foi dividido em três partes principais:

- **BackOffice:** responsável pelo cadastro de novas empresas, controle das mesmas e definição de configurações gerais.
- **Administração da Barbearia:** possibilita a gestão dos serviços, produtos, funcionários e horários.
- **Aplicação para o Cliente Final:** permite agendamentos, compras de produtos/serviços e, se habilitado, acompanhamento da evolução como membro da barbearia.

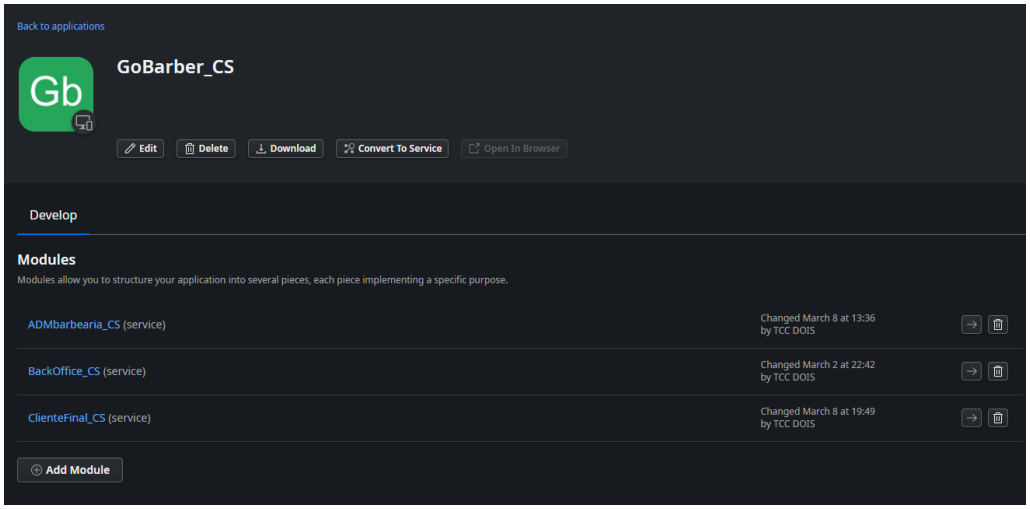
Figura 5 – *Ambientes do Sistema*

Fonte: Autor.

Na plataforma OutSystems, cada aplicação é um agregador de módulos, e poderá servir um nível arquitetural diferente, nossa aplicação adotou as seguintes camadas aplicacionais:

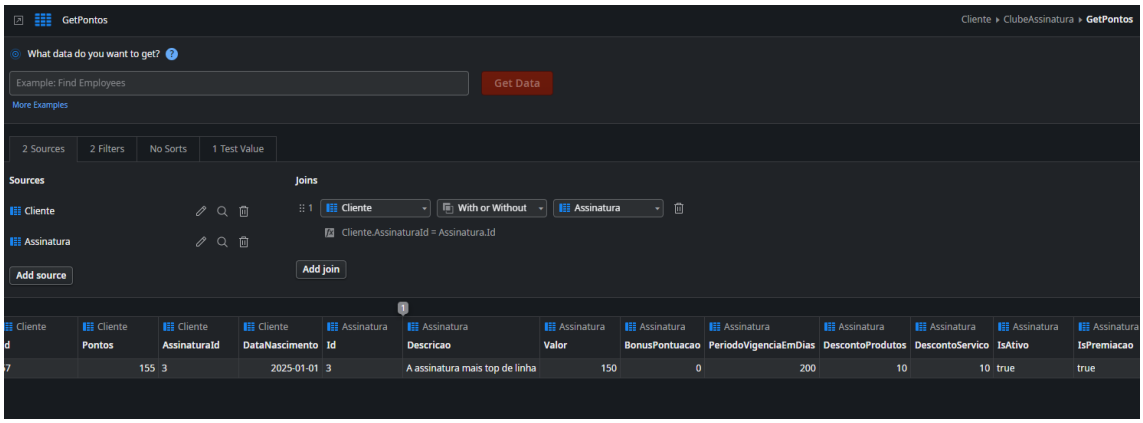
- **Core Services (CS):** Responsável pela base de dados e ações Create, Read, Update e Delete (CRUD). A requisição GET (requisição padrão de busca de dados de uma fonte) não foi implementada diretamente, pois a plataforma oferece a funcionalidade de agregados, que permite buscas simplificadas e integração com a interface. As figuras 6 e 7 demonstram os detalhes do ambiente CS.

Figura 6 – *Aplicação CS*



Fonte: Autor.

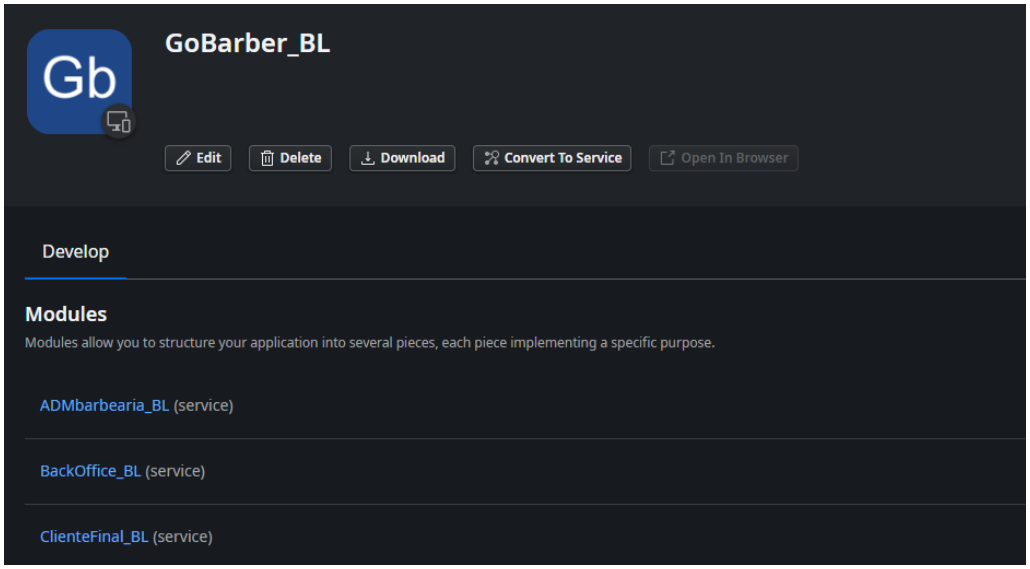
Figura 7 – *Agregado*



Fonte: Autor.

- **Business Logic (BL):** Responsável pela implementação da lógica de negócio, incluindo ações complexas, chamadas de integrações e APIs de serviços reutilizáveis. A figura 8 demonstra o ambiente BL.

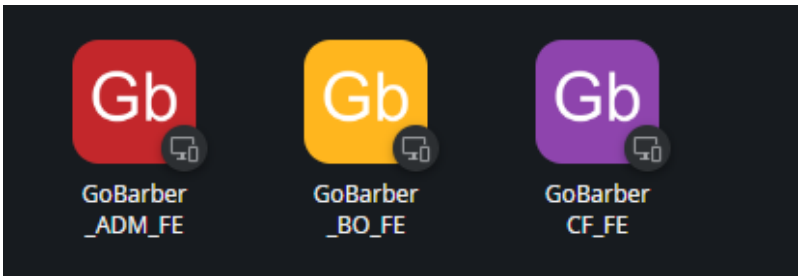
Figura 8 – *Aplicação BL*



Fonte: Autor.

- **Front End (FE):** Camada responsável pela interface visual. Contém aplicações separadas para BackOffice, Administração da Barbearia e Cliente Final. Essa separação reduz a complexidade e facilita a reutilização de componentes sem adaptações excessivas, conforme a Figura 9.

Figura 9 – *Aplicações de FE*



Fonte: Autor.

3.3 Implementação Incremental

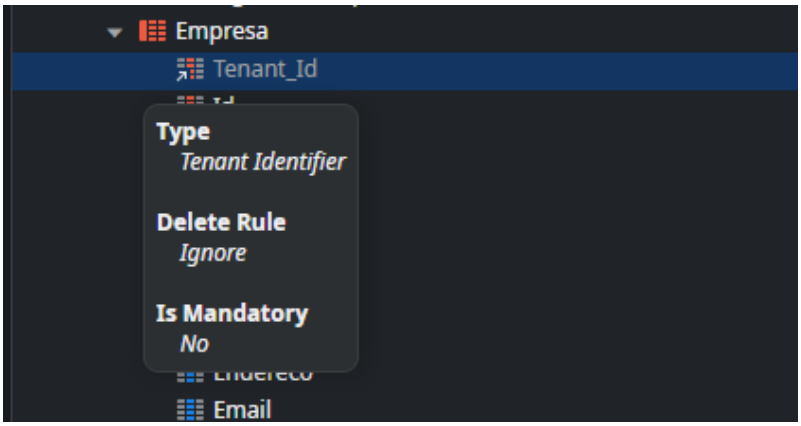
O desenvolvimento seguiu uma abordagem incremental, onde cada aplicação dependia da conclusão da aplicação anterior. A ordem de implementação foi:

1. **BackOffice:** necessário para criação das barbearias (tenants do sistema).
2. **Administração da Barbearia:** que depende do BackOffice para a criação e gerenciamento de barbearias.
3. **Cliente Final:** que depende da administração para consultar serviços, horários e realizar agendamentos.

A aplicação de BackOffice seguiu um *layout* amarelo e o Administração da Barbearia preto, o intuito aqui foi diferenciar as aplicações e não levar o utilizador ao engano, mesmo que a BackOffice seja acessível apenas internamente, é importante manter essa distinção. Já a aplicação Cliente Final seguirá com o seu *layout* personalizável conforme definido no BackOffice.

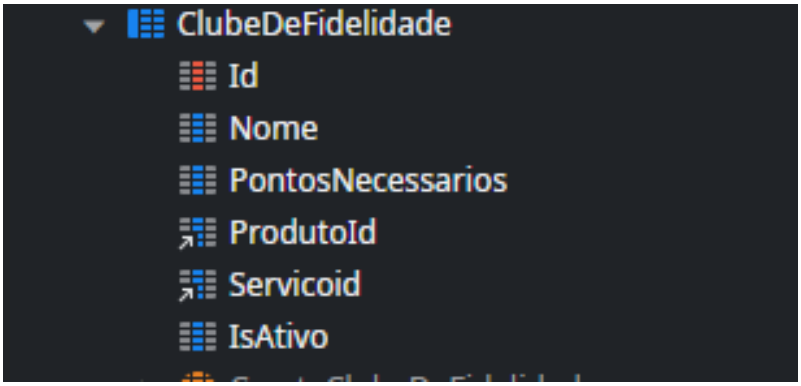
Foi adotada uma arquitetura *multitenancy*, que vai um pouco além da arquitetura aplicacional, ela permite que uma única instância do software operasse para múltiplos clientes (*tenants*). No sistema, cada *tenant* representa uma empresa, e a Outsystems oferece um recurso automático para gerenciar esse conceito. Esse encapsulamento dos dados foi essencial para garantir que cada empresa tivesse acesso apenas aos seus próprios dados, assim como demonstrado nas figuras 10 e 11.

Figura 10 – *Tabela de base de dados com o tenant gerenciado manualmente*



Fonte: Autor.

Figura 11 – *Tabela com o tenant autogerenciável pela OutSystems*

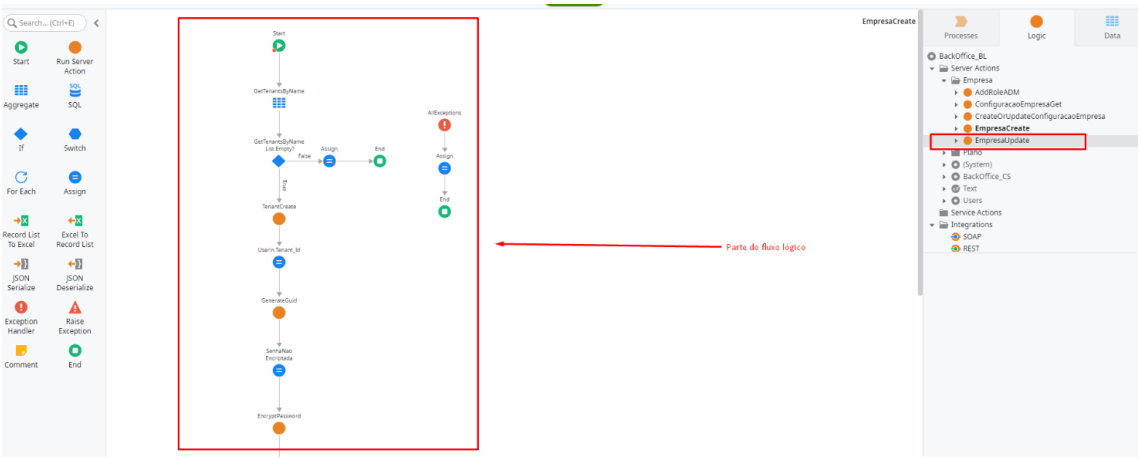


Fonte: Autor.

3.4 Desenvolvimento das aplicações

A primeira aplicação desenvolvida foi o BackOffice, que concentrou toda a lógica de criação de empresas e atributos relacionados, como configurações empresariais e planos de serviço oferecidos pela GoBarber para seus clientes (barbearias). A figura 12 exemplifica a criação de uma dessas empresas.

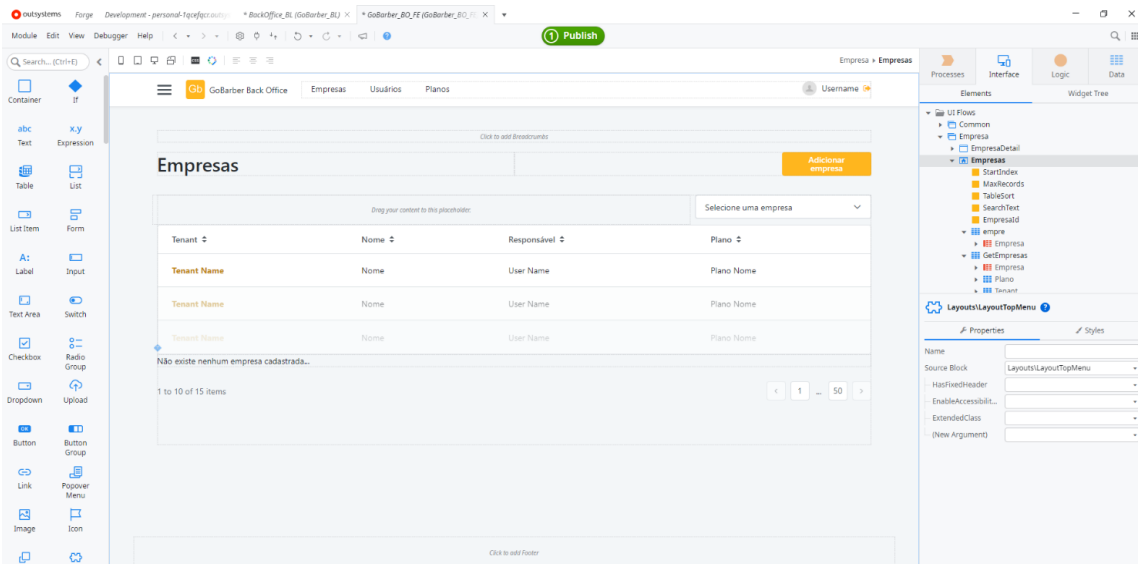
Figura 12 – *Ação de criar empresa no módulo BackOffice_BL*



Fonte: Autor.

O maior desafio encontrado foi a implementação correta do multitenancy, uma vez que a documentação da OutSystems, apesar de detalhada, ainda exigiu diversas tentativas e ajustes para garantir o correto funcionamento. Foi optado utilizar aqui os widgets de tela “crus”, sem customizações adicionais, aproveitando os aceleradores visuais da plataforma, que possibilitam a criação automática de telas para listagem e edição de registros. A figura 13 demonstra a lista de empresas na interface do Service Studio.

Figura 13 – *Tela de listagem de empresas do BackOffice*

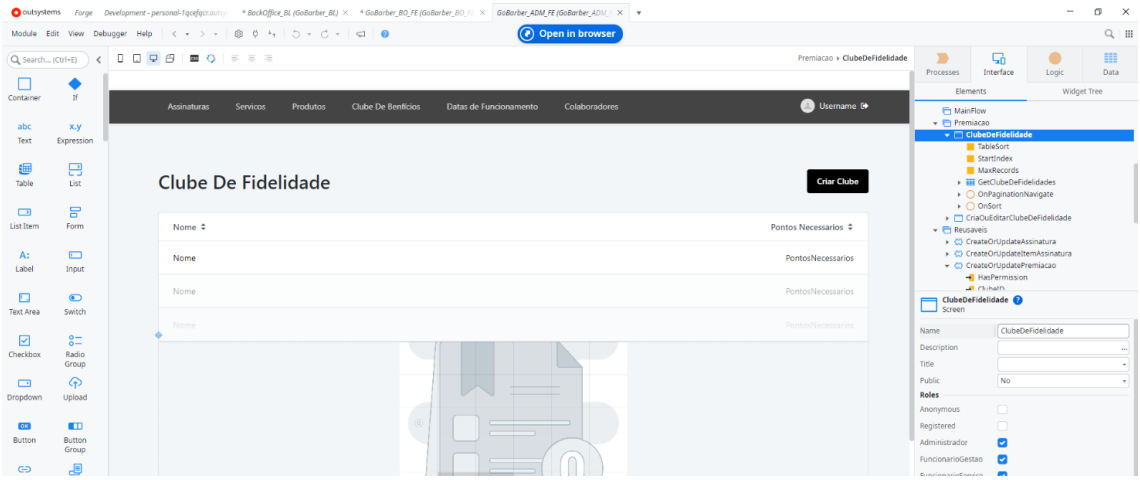


Fonte: Autor.

Com o BackOffice finalizado e testado, segue o desenvolvimento do módulo de Administração da Barbearia, que centralizou funcionalidades essenciais para a gestão do negócio, como cadastro de funcionários, produtos, serviços e assinaturas,

além da possibilidade de configurar níveis de clube de fidelidade, caso habilitado, conforme figura 14.

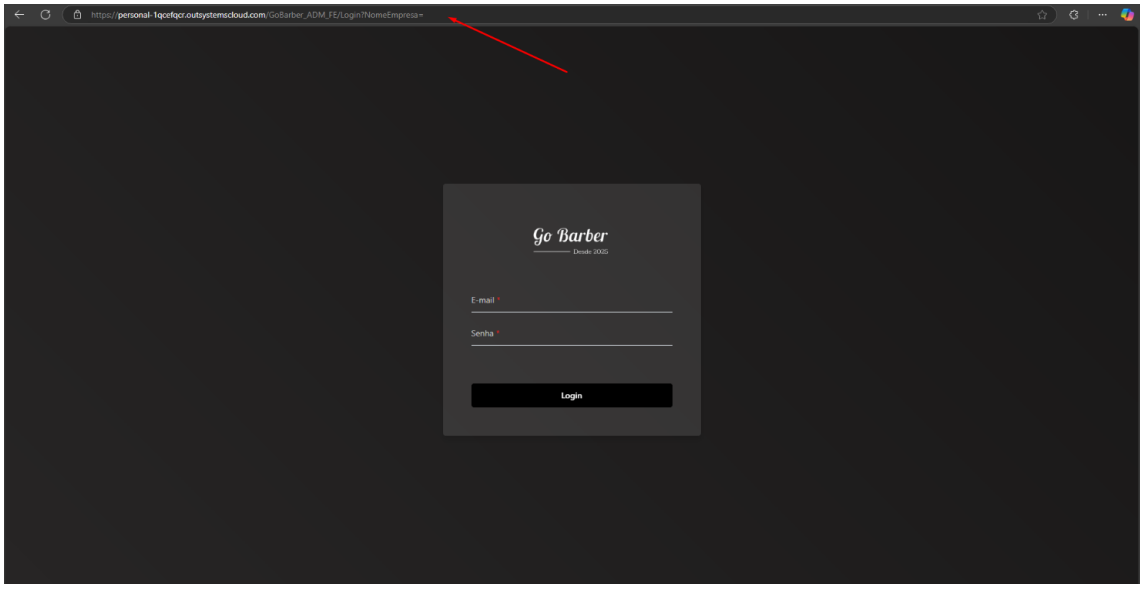
Figura 14 – *Tela de listagem do clube de fidelidade*



Fonte: Autor.

Esse módulo foi desenvolvido como uma aplicação totalmente multitenancy, garantindo que cada barbearia tenha uma instância isolada com encapsulamento de dados e recursos únicos. Essa abordagem facilita a escalabilidade do sistema sem necessidade de grandes modificações para atender novos clientes.

Figura 15 – *Imagem da tela de login da aplicação de Administração da Barbearia*



Fonte: Autor.

Agora, com o multitenancy ativado, é necessário que a aplicação se identifique, ou seja, passe o nome da empresa como parâmetro na URL para definir com

qual empresa deseja integrar e interagir.

No exemplo da figura 15 acima, pode-se ver que o campo está vazio, o que impede a identificação da empresa à qual o utilizador pertence. Como consequência, não será possível realizar o login. A URL atual está da seguinte forma:

- https://personal-1qcefqcr.outsystemscloud.com/GoBarber_ADM_FE/Login?NomeEmpresa=

Nota-se que não há nenhum valor após o "=". No entanto, se substituir o texto do link por:

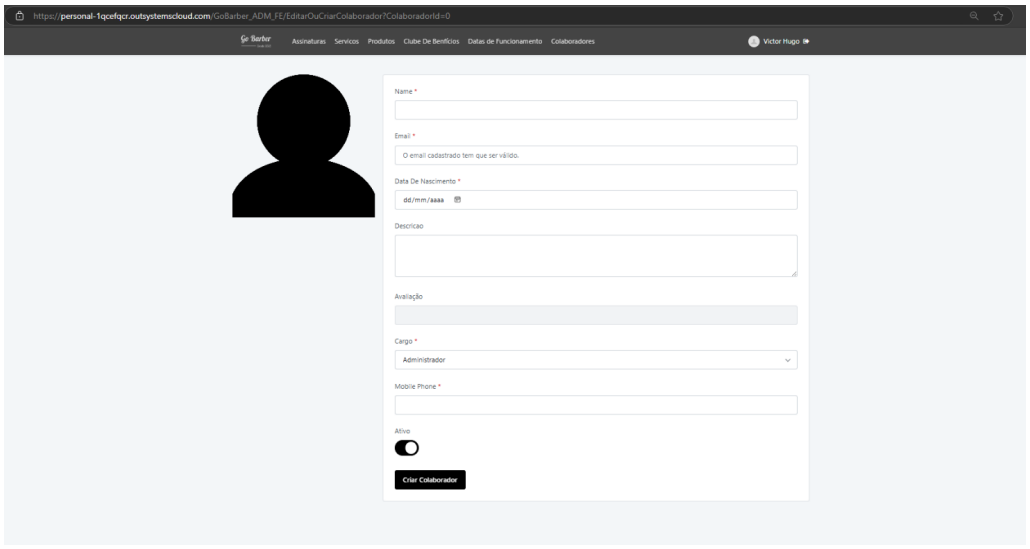
- https://personal-1qcefqcr.outsystemscloud.com/GoBarber_ADM_FE/Login?NomeEmpresa=EmpresaVictorHugo

O utilizador conseguirá se autenticar, desde que tenha um cadastro no tenant (empresa) informado.

A URL será sempre fornecida à barbearia assim que o seu cadastro for concluído no sistema. Cada empresa terá um campo "NomeEmpresa"único, utilizando as iniciais do CNPJ, garantindo que não haja duplicações. No entanto, dentro da aplicação, o nome exibido será o nome fantasia da barbearia.

Na aplicação de ADM, os utilizadores não podem se cadastrar diretamente. Apenas funcionários terão acesso à plataforma, e os acessos serão criados e fornecidos pelo administrador da empresa, que pode realizar essa gestão diretamente dentro da aplicação, assim como ilustrado na figura 16 abaixo.

Figura 16 – *Imagem da tela de cadastro de novos colaboradores*



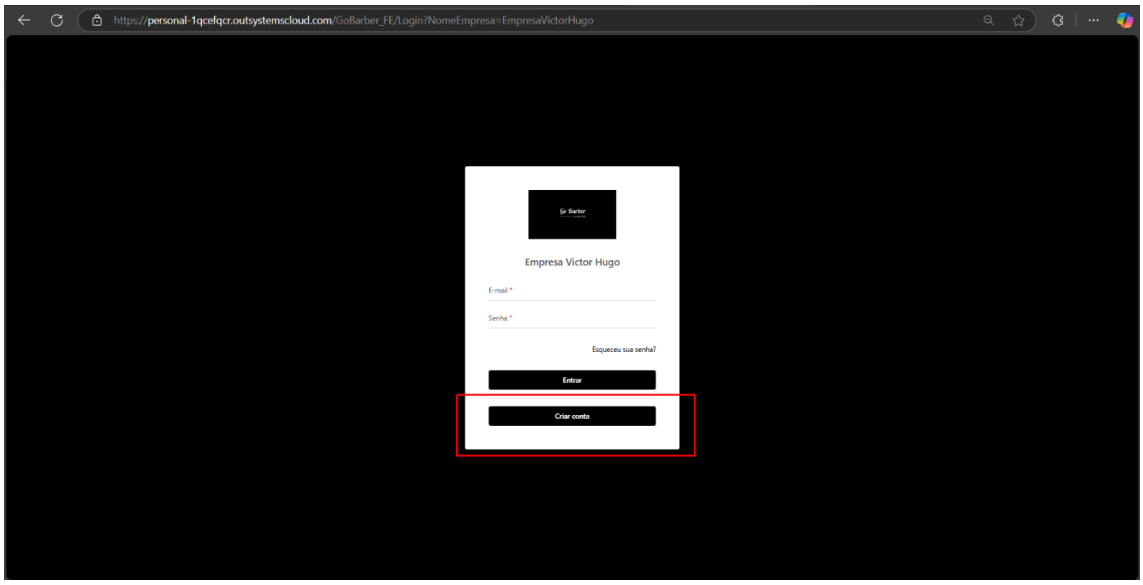
Fonte: Autor.

Durante a implementação, foi possível validar todas as funcionalidades do BackOffice, especialmente no que se refere ao gerenciamento de múltiplos tenants. A equipe não enfrentou grandes desafios no desenvolvimento estrutural da aplicação, sendo que a maior complexidade esteve na implementação das regras de negócio, que exigem um alto nível de precisão para garantir a boa funcionalidade do sistema. Isso se tornou ainda mais crítico ao considerar os níveis de personalização oferecidos aos clientes através de propostas diferenciadas. As rotinas de testes foram focadas principalmente na criação e gerenciamento de produtos e serviços, sendo que o impacto real desses recursos será mais evidente quando a aplicação do cliente final for lançada em produção.

Por fim, para a interface, foram utilizados componentes OutSystems pré-definidos, que são fundamentais na responsividade do sistema. Todos oferecem um estilo padrão, e o time de desenvolvimento optou por não os personalizar. A lógica central da aplicação foi construída com base nos módulos voltados para administração, garantindo um fluxo coeso.

Como etapa final dessa solução, foi desenvolvida a aplicação voltada para o Cliente Final, uma solução B2C disponibilizada pelas barbearias aos seus clientes. Essa aplicação integra praticamente todos os módulos anteriores, pois precisa consumir informações do BackOffice e do módulo ADM Barbearia para acessar dados essenciais, como disponibilidade de barbeiros e os privilégios de cada estabelecimento. Além disso, a aplicação foi projetada para ser totalmente *multitenancy*, garantindo que cada barbearia tenha uma instância única e personalizada. Aqui o usuário já consegue criar sua conta(ele será guiado pela URL fornecida onde terá os dados da empresa), conforme ilustrado na figura 17.

Figura 17 – *Imagem da tela de login da aplicação do cliente final*

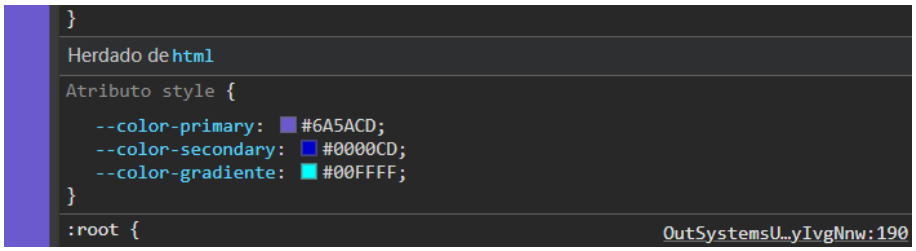


Fonte: Autor.

Um dos principais diferenciais dessa solução é a possibilidade de personalização da identidade visual. Cada barbearia pode configurar cores e imagens dinâmicas diretamente pelo BackOffice, permitindo que a aplicação reflita sua identidade visual. A equipe desenvolvedora considera esse recurso fundamental, pois empresas investem significativamente na construção de suas marcas, e oferecer um aplicativo alinhado com essa identidade fortalece a experiência do cliente e a percepção do negócio. Conforme destacado pela PRINTI (2024), a identidade visual de uma empresa é essencial para mostrar quem ela é e como atua no mercado, influenciando diretamente a opinião dos consumidores.

Abaixo, na figura 18, detalha-se o código HTML da aplicação que permite haver variáveis globais que definem o tema da aplicação:

Figura 18 – *Variáveis de definição de cores no HTML*

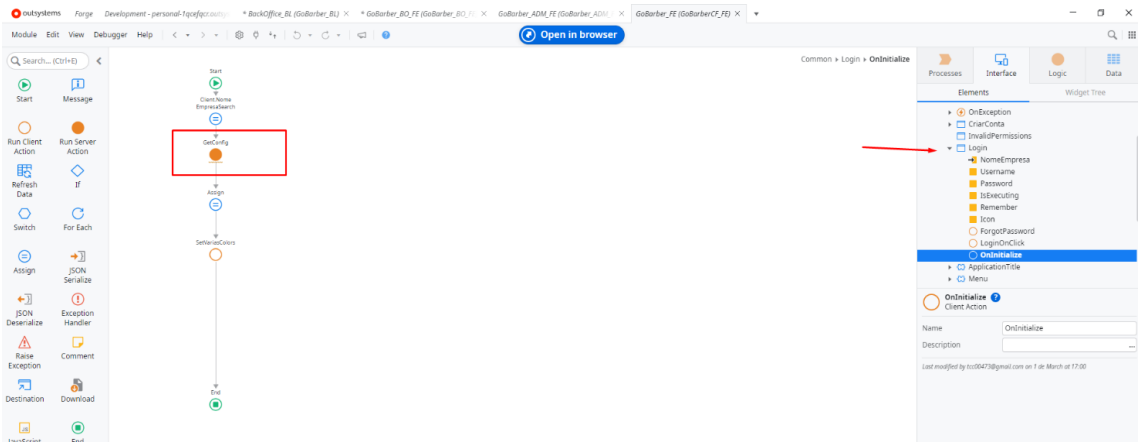


Fonte: Autor.

Primeiramente, identifica-se a aplicação à partir do parâmetro que é recebido na URL, no evento de inicialização da página, onde é feita a chamada de uma

função que traz todas as cores configuradas no BackOffice para a empresa, conforme figura 19:

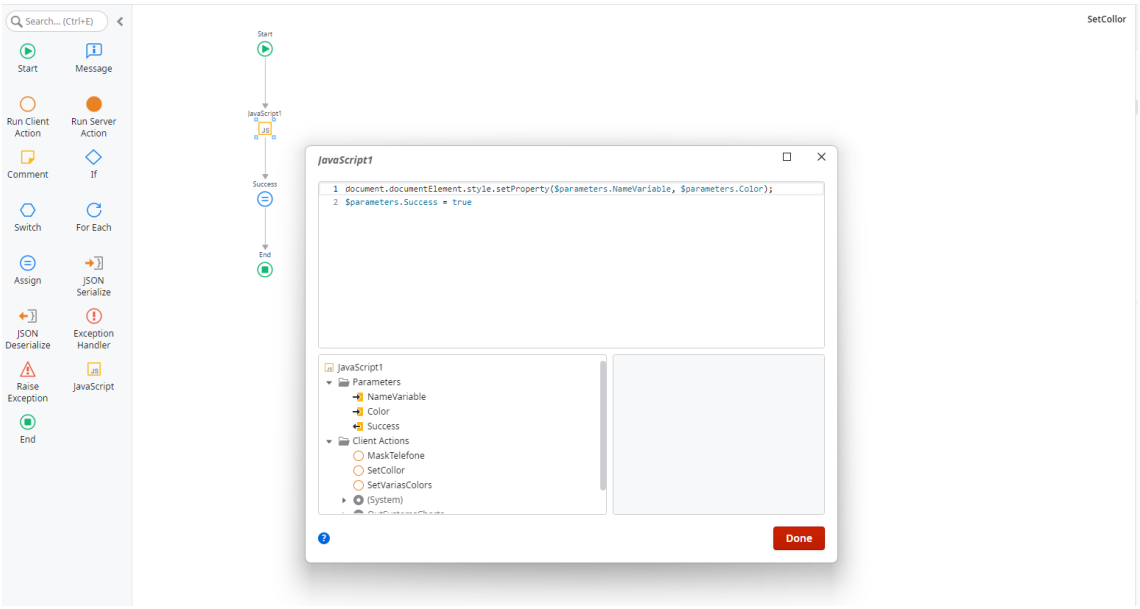
Figura 19 – *função que faz o GET das cores configuradas no BackOffice*



Fonte: Autor.

Após buscar as cores, o app chama uma função JavaScript que interage com o HTML da página. A partir dessa ação, ocorre a troca das cores dinamicamente, conforme demonstra a figura 20:

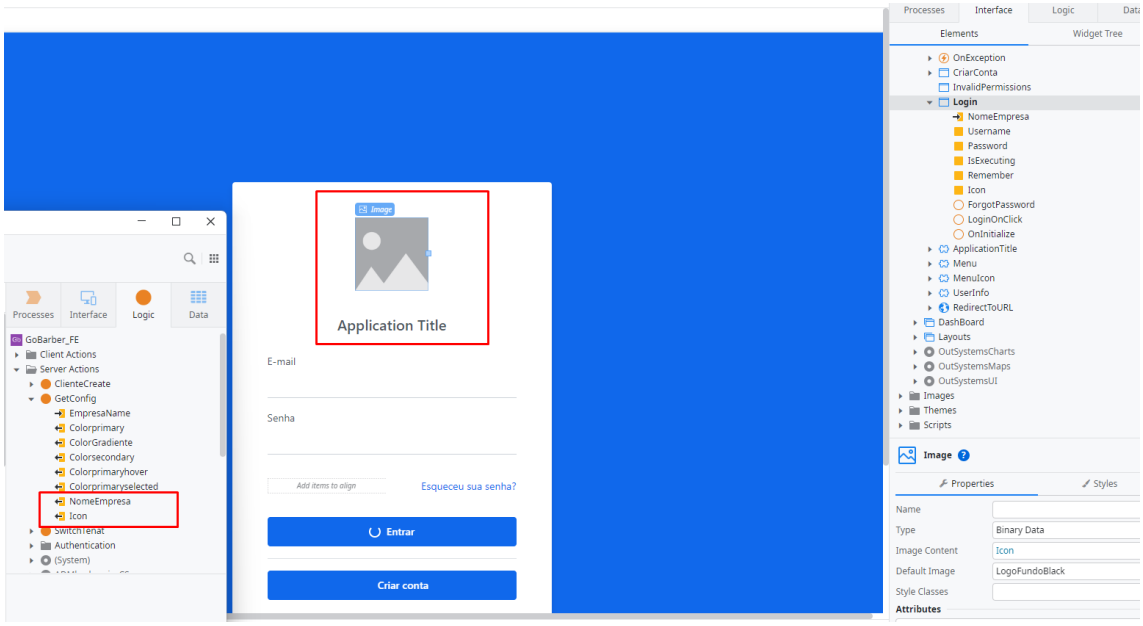
Figura 20 – *Ação que realiza a troca das cores dinamicamente*



Fonte: Autor.

Além de regatar as cores configuradas em backoffice, a ação “GetConfig” retorna o ícone que personaliza a tela de login e o nome da empresa, e os altera, conforme ilustrado na figura 21:

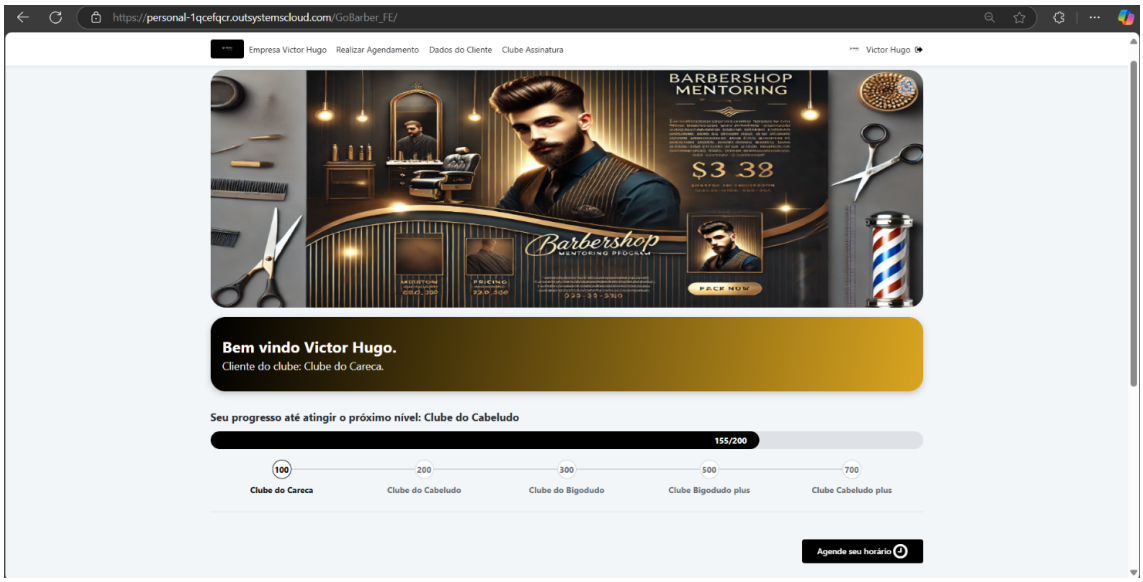
Figura 21 – *Alteração dinâmica do ícone da empresa e do nome*



Fonte: Autor.

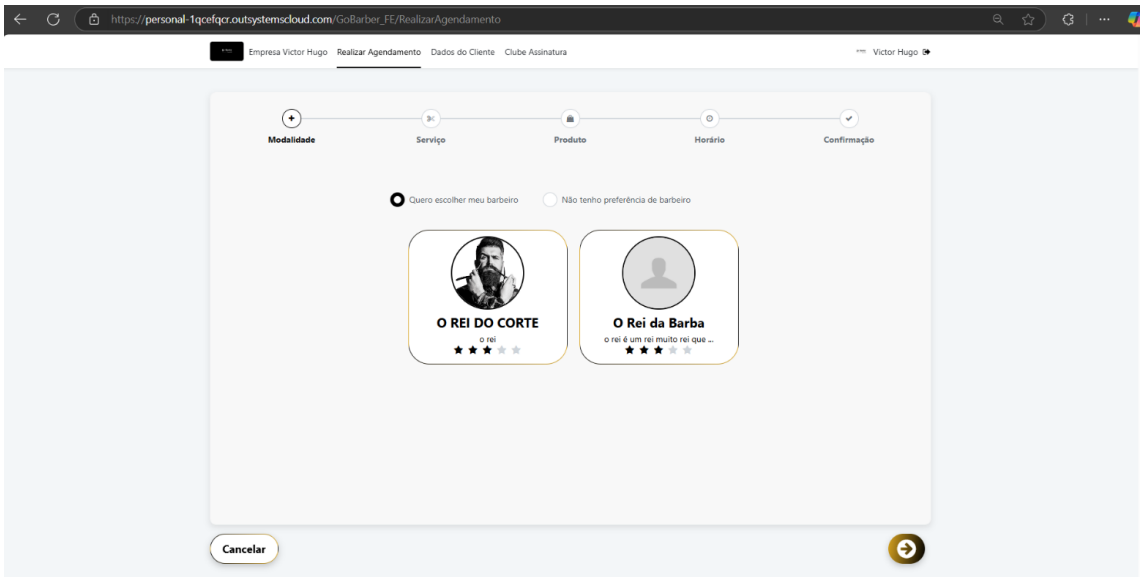
O foco principal nesta fase foi a usabilidade e experiência do usuário. Como essa aplicação representa a principal interface entre os clientes e as barbearias, buscou-se um design simples, moderno e intuitivo, que facilite a navegação e torne o processo de agendamento o mais fluido possível, conforme Krug (2014). Para isso, evitou-se o uso direto dos aceleradores padrão da OutSystems, e aplicou-se um nível extra de customização inspirado em templates modernos, garantindo um design mais sofisticado e responsivo. Essa abordagem está alinhada com os princípios de design centrado no usuário, conforme discutido por Garrett (2011).

Figura 22 – *Tela Inicial no desktop*



Fonte: Autor.

Figura 23 – *Tela inicial de agendamento a partir de um desktop*

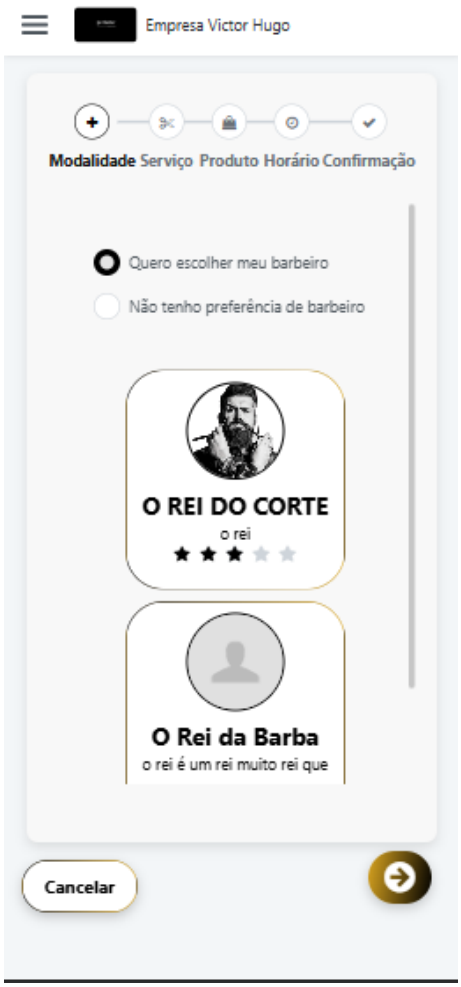


Fonte: Autor.

Figura 24 – *Tela inicial a partir de um smartphone (Iphone XR)*

Fonte: Autor.

Figura 25 – *Tela inicial de agendamento a partir de um smartphone (Iphone XR)*



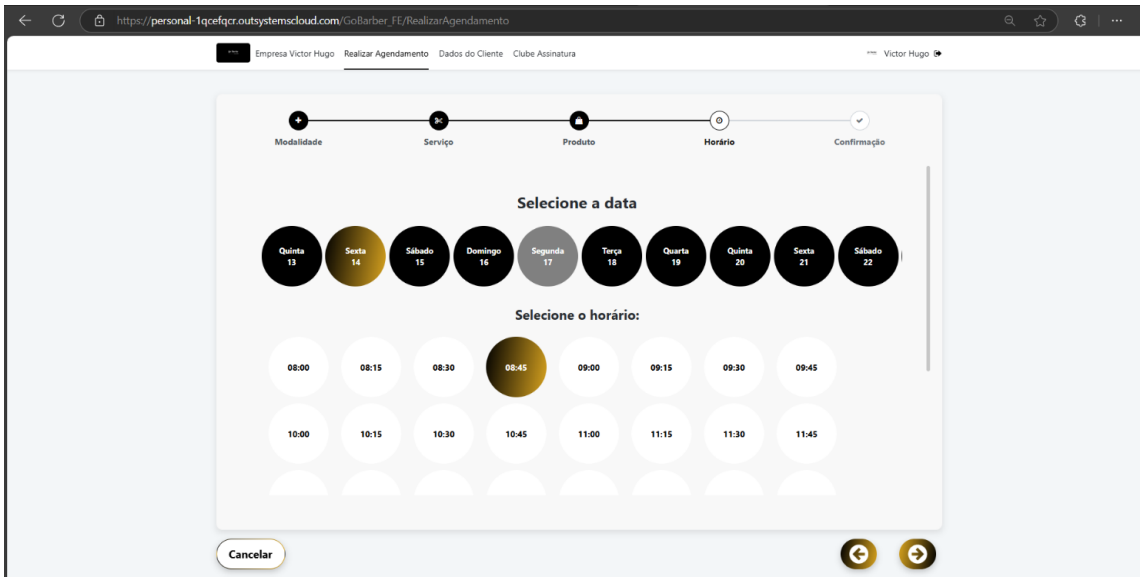
Fonte: Autor.

Conforme demonstrado nas figuras 22, 23, 24 e 25, a responsividade foi um ponto crítico no desenvolvimento, uma vez que a maioria dos acessos à aplicação ocorrerá via dispositivos móveis. A equipe dedicou cerca de metade do tempo de desenvolvimento para garantir que todos os componentes ficassem adaptados tanto para web/desktop quanto para web/mobile, proporcionando uma experiência fluida independentemente do dispositivo utilizado.

O uso de cores em gradiente em elementos estratégicos da aplicação tem o objetivo de aprimorar a experiência do usuário, tornando a navegação mais intuitiva e visualmente atraente. Conforme descrito por Elementor (2025), os gradientes de cor oferecem uma maneira de adicionar mais textura, profundidade e dinamismo a uma página web que, de outra forma, seria plana e sem vida. Dessa forma, ao implementar gradientes em botões de ação e áreas de destaque, é possível guiar a atenção do cliente para funcionalidades essenciais, incentivando a interação e enriquecendo a estética da aplicação.

Além do design, a lógica de negócio exigiu atenção especial, principalmente no matching de agendas (Figura 26). Garantir que os clientes pudessem visualizar e selecionar horários disponíveis corretamente foi um desafio que demandou diversos testes e refinamentos. A construção e validação das regras de agendamento exigiram um processo contínuo de tentativa e erro, garantindo que todas as interações fossem precisas e eficientes.

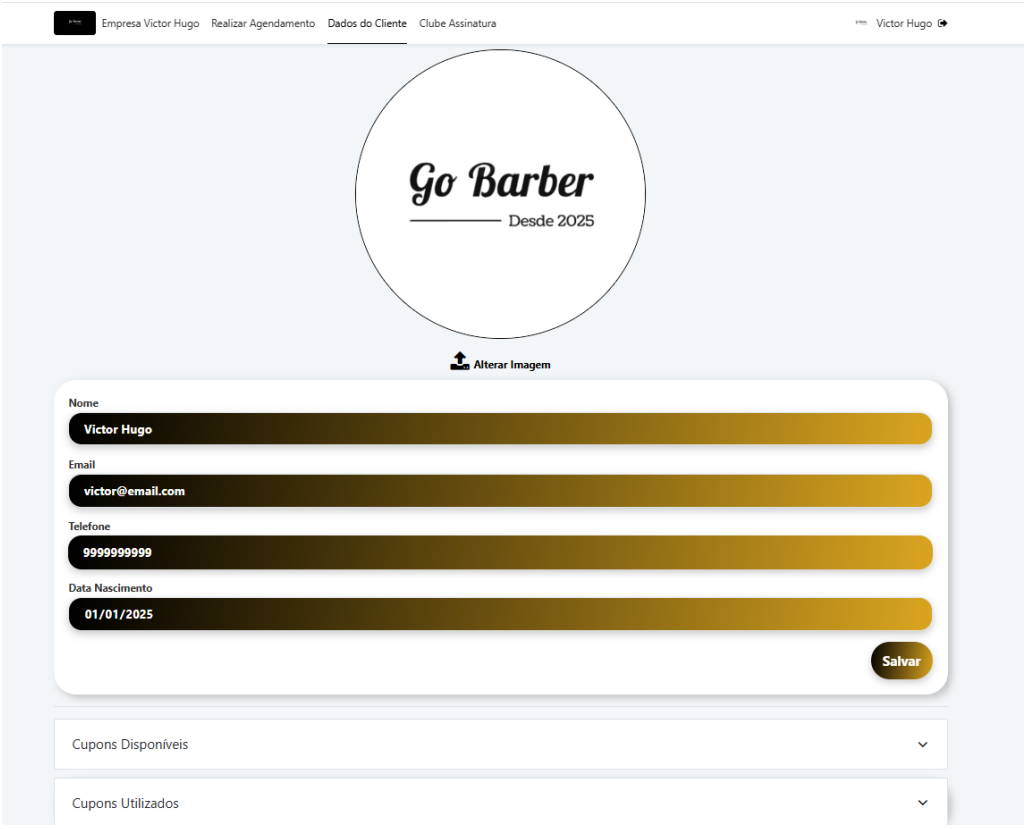
Figura 26 – *Tela de agendamento: escolha do horário*



Fonte: Autor.

No final, a aplicação se tornou um diferencial estratégico, fortalecendo a presença digital das barbearias e otimizando a comunicação com os clientes. O sucesso dessa solução reflete diretamente no crescimento da plataforma GoBarber, pois quanto melhor for a experiência do usuário final, maior será a retenção e a satisfação dos clientes da barbearia (Freitas, 2023). Na figura 27 a seguir, a tela principal do app Cliente Final.

Figura 27 – *Imagem da aplicação Cliente Final*

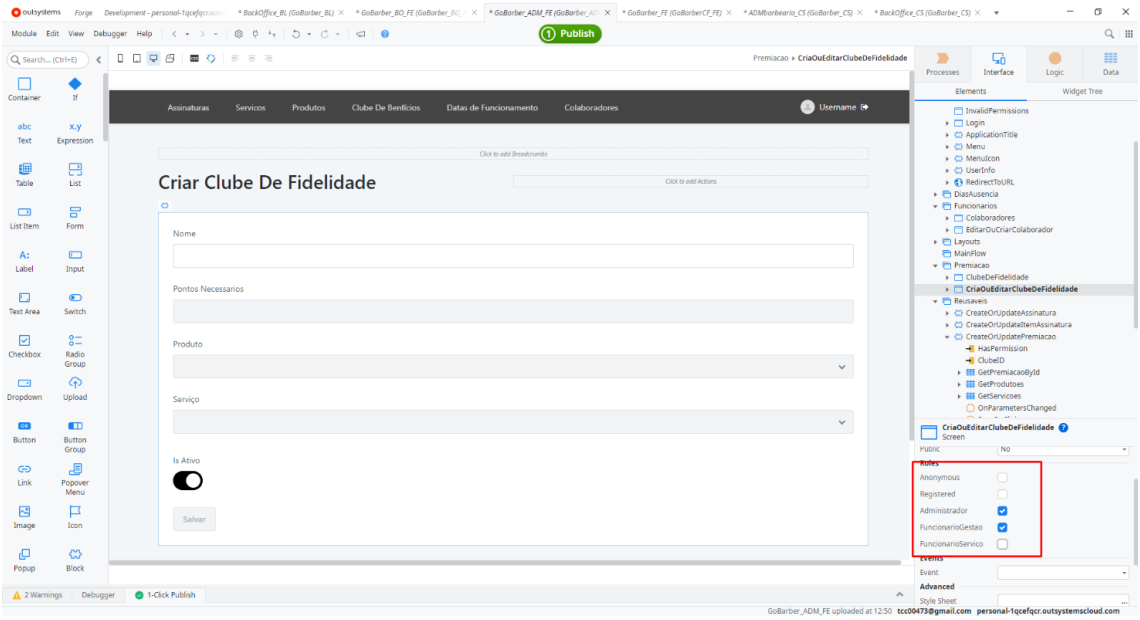


Fonte: Autor.

Uma informação importante é que, quando um utilizador cria uma conta, por exemplo, na empresa "EmpresaVictorHugo"através da aplicação destinada ao cliente final, ele se torna um utilizador válido para a Empresa Victor Hugo. Em teoria, ele poderia acessar qualquer instância de uma aplicação que tenha um tenant criado e vinculado a essa empresa.

No entanto, o que impede esse utilizador de acessar, por exemplo, a aplicação de administração da barbearia é o controle de privilégios por meio de *roles* (papéis) aplicacionais. Esse controle pode ser aplicado durante o desenvolvimento em funções específicas ou, de forma mais simples, diretamente nas permissões das telas. Quando é necessário que qualquer um possa acessar aquela tela (como a de login), basta marcar como “Anonymous” (figura 28); caso contrário, basta selecionar com qual role o utilizador poderá acessar aquela tela:

Figura 28 – *Imagem de demonstração de uma role aplicacional*



Fonte: Autor.

Caso o usuário não possua a *role* exigida e por algum motivo tente acessar a tela, ele entrará em uma tela *default*, e será redirecionado para o login. As *roles* são atribuídas automaticamente na criação de um novo usuário; enquanto no caso da aplicação de administração elas serão selecionadas de acordo com o cargo do funcionário; por fim, na aplicação do cliente final, será uma única *role* de "cliente".

Para manter a aplicação dentro do escopo desse projeto, a equipe não se empenhou em integrar serviços de pagamento e envio de mensagens SMS. Esses recursos serão importantes para o negócio em uma futura evolução e implantação em produção, porém, no momento, por se tratar de um ambiente de desenvolvimento não são relevantes para a entrega, especialmente por envolverem custos com a aquisição de serviços de terceiros. A implementação desses recursos ficará como uma possível melhoria em projetos futuros.

Mesmo com esses serviços indisponíveis, foi possível manter a funcionalidade do fluxo de serviço adotado nas barbearias.

3.5 Rotinas de Teste

Os testes desempenham um papel fundamental na garantia da qualidade de um sistema. Os testes asseguram que todas as funcionalidades operem corretamente e que a experiência do usuário seja a mais otimizada possível. Como mencionado anteriormente, foram conduzidos testes unitários e testes funcionais de forma *ad hoc*

— isto é, testes exploratórios não estruturados, sem documentação formal, realizados com o intuito de identificar falhas inesperadas no sistema — sempre que novas funcionalidades eram incorporadas ao sistema. Essa abordagem permitiu identificar e corrigir falhas de forma ágil, garantindo um ciclo de desenvolvimento eficiente e iterativo.

Além dos testes técnicos, a equipe também priorizou a realização de testes de usabilidade, com base na teoria de Krug (2014), que enfatiza a importância de interfaces intuitivas e fáceis de usar. Esses testes foram conduzidos para validar a fluidez da navegação e identificar pontos de atrito na experiência do usuário. Durante as sessões de teste, feedbacks foram coletados, sendo registrados e analisados para identificar melhorias necessárias. Esses feedbacks foram essenciais para ajustes na interface, tornando-a mais intuitiva, e para a correção de inconsistências funcionais que poderiam comprometer a experiência do usuário.

Por fim, a implementação de testes contínuos ao longo do desenvolvimento contribuiu para a redução de futuros erros em produção, assegurando que a aplicação seja entregue com um alto nível de estabilidade e conformidade com os requisitos do projeto. Adotar a estratégia de testes iterativos nos permitiu um refinamento constante do sistema, promovendo melhorias progressivas na usabilidade e desempenho da solução.

3.6 Resultados Obtidos

A implementação do sistema de gerenciamento para barbearias apresentou resultados expressivos, evidenciando a eficácia das estratégias adotadas ao longo do projeto. Desde a fase inicial, a aplicação da metodologia ágil Scrum permitiu entregas incrementais e adaptações rápidas. A abordagem iterativa possibilitou ajustes contínuos e garantiu que as funcionalidades fossem priorizadas conforme as necessidades do negócio, assegurando uma evolução estruturada do sistema.

Para validar a eficiência da solução, foram realizados testes funcionais, de usabilidade e a coleta de feedbacks dos usuários(desenvolvedores). Os testes funcionais garantiram que funcionalidades essenciais, como agendamento de serviços, gerenciamento de produtos e gerenciamento de benefícios operassem conforme os requisitos especificados. Durante os testes de usabilidade, conduzidos inicialmente com a equipe de desenvolvimento, foram avaliados aspectos como facilidade de navegação, tempo de resposta e clareza das interfaces gráficas. Os feedbacks obtidos foram fundamentais para otimizar a experiência do usuário, simplificando fluxos de agendamento e aprimorando a disposição das informações. Conforme Krug (2014), a simplicidade e a facilidade de uso são fatores determinantes para a aceitação

de sistemas por usuários não técnicos, e essa diretriz foi considerada ao longo do processo de refinamento da interface.

A estruturação modular do sistema mostrou-se essencial ao projeto. A divisão em três partes principais — BackOffice, Administração da Barbearia e Cliente Final — proporcionou uma organização eficiente das funcionalidades e atendeu aos diferentes perfis de usuários dentro da plataforma. Além disso, a arquitetura *multitenancy* permitiu que cada barbearia operasse de forma isolada, com acesso exclusivo aos seus próprios dados. Esse modelo garantiu escalabilidade e segurança para a aplicação, pois isola os dados de cada empresa e previne que usuários indevidos acessem ambientes sem permissão. A infraestrutura em nuvem oferecida pela OutSystems contribuiu para esse resultado, garantindo flexibilidade e alta disponibilidade da aplicação, conforme recomendado por Silberschatz, Korth e Sudarshan (2019).

Um dos resultados mais impactantes foi a automação dos processos de agendamento, eliminando a necessidade de registros manuais e tornando o fluxo mais ágil e preciso. Os clientes puderam agendar serviços online, selecionar profissionais e receber notificações automáticas em seus dispositivos móveis, reduzindo falhas operacionais e aumentando a conveniência para os usuários. Além disso, a gestão de estoque foi aprimorada por meio da implementação de alertas automáticos para reposição de materiais essenciais, como lâminas e cosméticos. Essa funcionalidade minimizou casos de falta de insumos, melhorando a eficiência operacional das barbearias e garantindo a continuidade dos serviços prestados.

Outro fator relevante foi a possibilidade de personalização da identidade visual dentro do sistema. O desenvolvimento do BackOffice permitiu ajustes de cores e imagens dinâmicas, possibilitando que cada barbearia refletisse sua identidade própria na aplicação. Esse recurso agregou valor à solução, uma vez que as empresas investem significativamente na construção de suas marcas e buscam ferramentas que reforcem essa identidade no ambiente digital. O alinhamento da experiência do usuário com a identidade visual do negócio fortaleceu a percepção da marca e melhorou a interação com o cliente final.

A responsividade da aplicação também foi um aspecto crítico no desenvolvimento. Considerando que a maior parte dos acessos ocorre via dispositivos móveis, foi dedicado um tempo significativo ao design para garantir que todos os componentes estivessem bem adaptados tanto para web quanto para mobile. A interface foi projetada para ser intuitiva, moderna e eficiente, proporcionando uma navegação fluida e uma experiência de uso coesa, independentemente do dispositivo utilizado. Para aprimorar essa experiência, evitou-se o uso direto dos aceleradores padrão da OutSystems, aplicando um nível extra de customização inspirado em templates mo-

dernos.

Apesar dos desafios enfrentados, principalmente na implementação do *multitenancy*, a equipe conseguiu validar todas as funcionalidades e garantir um modelo robusto e confiável. O gerenciamento eficiente de múltiplos tenants possibilitou que novas barbearias fossem adicionadas à plataforma sem comprometer a segurança e a integridade dos dados. Além disso, a lógica de negócio exigiu um alto nível de precisão, principalmente no que diz respeito ao gerenciamento de agendas, um dos pontos mais críticos do projeto. Diversos testes e refinamentos foram necessários para assegurar que os clientes pudessem visualizar e selecionar horários disponíveis corretamente, garantindo um fluxo de agendamento eficiente.

Ao final do projeto, a aplicação se consolidou como uma solução promissora e funcional, fortalecendo a presença digital das barbearias e otimizando a comunicação com os clientes. No entanto, devido às limitações de licença da plataforma, não foi possível realizar testes em barbearias reais, o que restringiu a validação do sistema em um ambiente de produção.

A combinação de uma arquitetura escalável, uma experiência de usuário aprimorada e uma identidade visual personalizável garantiu que o sistema atendesse plenamente às necessidades do mercado. Os resultados confirmaram a eficácia da plataforma OutSystems para o desenvolvimento ágil de soluções empresariais, fornecendo ferramentas e componentes suficientemente abrangentes para criações rápidas, mesmo com equipes pequenas. Além disso, a plataforma possibilitou um alto nível de personalização na interface, permitindo entregas mais adaptadas às necessidades do cliente final.

Embora a OutSystems simplifique significativamente o desenvolvimento de aplicações ao oferecer diversas ferramentas que auxiliam na construção de softwares robustos e fluidos, é necessário um certo nível de conhecimento da plataforma. Um estudo aprofundado sobre seu funcionamento e suas limitações é essencial para que a ferramenta seja utilizada de maneira eficiente.

Mesmo com essas considerações, a utilização da OutSystems permitiu melhorias significativas na gestão das barbearias, contribuindo para seu crescimento e aumentando sua competitividade no cenário digital.

CAPÍTULO 4

Considerações Finais

Este projeto utilizou diversos conhecimentos em engenharia de software, modelagem de dados e design web para desenvolver uma aplicação capaz de automatizar os serviços e requisitos de uma barbearia. Além do domínio técnico necessário para a implementação, o desenvolvimento do software exigiu conhecimento de negócios e escuta ativa, pois criar um sistema não se trata apenas de desenvolver uma aplicação, mas sim de oferecer uma solução real para um problema concreto.

Ao longo do projeto, a interação com um gerente de barbearia foi um dos momentos mais enriquecedores, pois permitiu compreender as dores e desafios enfrentados no dia a dia da gestão desse tipo de negócio. Isso levou a equipe a refletir sobre como a tecnologia pode auxiliar tarefas tradicionalmente manuais, promovendo eficiência e inovação em um setor que, há algumas décadas, dificilmente seria associado a soluções tecnológicas avançadas.

A plataforma OutSystems teve um papel essencial nesse processo, pois, ao abstrair questões técnicas mais complexas, permitiu que a equipe focasse no negócio e na melhor forma de atender as necessidades dos clientes. Essa abordagem tornou possível a implementação de requisitos de maneira ágil e com uma equipe reduzida, tornando a OutSystems uma opção viável e eficiente para soluções comerciais.

O projeto cumpriu o objetivo de desenvolver uma aplicação direcionada para barbearias, oferecendo uma ferramenta que pode agregar valor aos empresários e otimizar a gestão do negócio. Entretanto, há possibilidades de evolução, e espera-se que futuras implementações tragam ainda mais personalização e funcionalidades estratégicas. Algumas melhorias previstas incluem:

- Geração de relatórios para análise de métricas como produtividade, presença de clientes e crescimento do mercado.
- Integrações com sistemas de pagamento para facilitar transações financeiras.
- Envio automatizado de mensagens via WhatsApp ou SMS para comunicação com clientes.
- Dashboards interativos, oferecendo uma visão geral e detalhada do desempenho do negócio.

Neste primeiro momento, o foco foi entregar as funcionalidades essenciais com eficiência. Com todas as implementações concluídas, o próximo passo será validar a aplicação em um cliente real, garantindo que atenda plenamente às necessidades do mercado.

Referências

AWS. **Qual é a diferença entre front-end e back-end no desenvolvimento de aplicações?** 2023. Online. Disponível em: https://aws.amazon.com/pt/compare/the-difference-between-frontend-and-backend/. Acesso em: 22/09/2024. 18, 19

Elementor. **Como Utilizar Gradientes no Design Web: Tendências e Exemplos**. 2025. Online. Disponível em: https://elementor.com/blog/pt-br/como-utilizar-gradientes-no-design-web-tendencias-e-exemplos. Acesso em: 09/03/2025. 45

FREITAS, Matheus V. L. **Sistemas de Agendamento para Barbearias: O Papel dos Sistemas de Agendamento na Melhoria da Experiência do Cliente em Barbearias**. Jales: [s.n.], 2023. FATEC Jales. Disponível em: https://ric.cps.sp.gov.br/bitstream/123456789/16700/1/analise_e_desenvolvimento_de_sistemas_2023_2_matheus_vinicius_leal_freitas_sistemas_de_agendamento_para_barbearias_o_papel_dos_sistemas.pdf. Acesso em: 17/08/2024. 13, 46

GARRETT, Jesse James. **The Elements of User Experience: User-Centered Design for the Web and Beyond**. 2. ed. Berkeley: New Riders, 2011. 42

HIGHSMITH, Jim. **The Agile Manifesto**. 2001. Online. Disponível em: https://agilemanifesto.org. Acesso em: 13/09/2024. 25

KRUG, Steve. **Não me faça pensar, Revisitado: uma abordagem de bom senso à usabilidade na web**. 1. ed. Rio de Janeiro: Alta Books, 2014. 26, 28, 42, 49

OUTSYSTEMS. **ambiente OS**. 2024. Disponível em: https://mundooutsystems.com/planejando-a-infraestrutura-outsystems/. 23

Outsystems. **The Architecture Canvas**. 2024. Online. Disponível em: https://success.outsystems.com/documentation/best_practices/architecture/

designing_the_architecture_of_your_outsystems_applications/the_architecture_canvas/. Acesso em: 22 set. 2024. 18, 20

OUTSYSTEMS. **Exemplo de uma arquitetura On-Premises**. 2024. Disponível em: https://success.outsystems.com/documentation/11/setup_outsystems_infrastructure_and_platform/setting_up_outsystems/possible_setups_for_an_outsystems_infrastructure/. 22

PRINTI. **Identidade visual: por que toda empresa deve investir**. 2024. Online. Disponível em: https://www.printi.com.br/blog/identidade-visual-por-que-toda-empresa-deve-investir. Acesso em: 09/03/2025. 40

SANTOS, Vitor Leal dos. **UnB Estágio: desenvolvimento de um aplicativo mobile para divulgação de vagas de estágios**. Brasília: [s.n.], 2022. UnB. Disponível em: https://bdm.unb.br/bitstream/10483/34014/1/2022_VitorLealDosSantos_tcc.pdf. Acesso em: 03/09/2024. 25, 26

SCRUM.ORG. **exemplo**. 2020. Disponível em: https://www.scrum.org/resources/what-scrum-module. Acesso em: 26/09/2024. 25

SILBERSCHATZ, Abraham; KORTH, Henry F.; SUDARSHAN, S. **Sistemas de Banco de Dados**. 6. ed. São Paulo: Pearson, 2019. 18, 50

SOMMERVILLE, I. **Engenharia de software**. 10. ed. São Paulo: Pearson Education do Brasil, 2019. 14, 24, 25, 27, 28, 29

SUTHERLAND, Jeff; SCHWABER, Ken. **Um guia definitivo para o Scrum: As regras do Jogo**. 2017. Online. Disponível em: https://scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-Portuguese-Brazilian.pdf. Acesso em: 03/03/2025. 27

VIEIRA, Gustavo Henrique Freitas; SOUSA, Alvaro Recoba de; RODRIGUES, Gabriel Marcacini Vargas; VALE, Otávio Fonseca. **Uso do Design na Elaboração de uma Campanha de Rebranding para a Barbearia Hitallo Cortes**. Goiânia: [s.n.], 2024. UFG. 13