**PÓS-GRADUAÇÃO EM DESENVOLVIMENTO FULL STACK**

**DESENVOLVIMENTO DE UMA APLICAÇÃO WEB COM IOT PARA MONETIZAÇÃO DO MARKETPLACE INFORMAL DE CAFÉ E OUTROS CONSUMÍVEIS EM UM AMBIENTE EMPRESARIAL**

LUIZ GUSTAVO PFITSCHER E FELDMANN

Orientador: Eduardo Campos Pellanda

2025



**SUMÁRIO**

[1. Contextualização da proposta 3](#_Toc188220909)

[2. Objetivos da construção da solução 4](#_Toc188220910)

[3. Elaboração da jornada do usuário 5](#_Toc188220911)

[4. Apelo mercadológico da solução 7](#_Toc188220912)

[5. Ciclo de desenvolvimento da solução 8](#_Toc188220913)

[5.1. Análise de Viabilidade e MVP 8](#_Toc188220914)

[5.2. Elaboração da Arquitetura de Software 9](#_Toc188220915)

[5.3. Preparação dos Ambientes de Teste e Produção e Pipelines de CI/CD 9](#_Toc188220916)

[5.4. Desenvolvimento Incremental 9](#_Toc188220917)

[6. Mockup da proposta de solução 10](#_Toc188220918)

[7. Arquitetura de Software 13](#_Toc188220919)

[8. Validação da solução 14](#_Toc188220920)

[8.1. Uso do *TypeScript* 14](#_Toc188220921)

[8.2. Uso do *Next.js* 14](#_Toc188220922)

[8.3. Testes Automatizados 14](#_Toc188220923)

[8.4. Ambiente de Desenvolvimento 15](#_Toc188220924)

[8.5. Ambiente de Testes e Pipeline de CI/CD 15](#_Toc188220925)

[8.6. Ambiente de Produção 15](#_Toc188220926)

[9. Registros das evidências do projeto 16](#_Toc188220927)

[10. Considerações finais e expectativas 17](#_Toc188220928)

[Referências 18](#_Toc188220929)

# Contextualização da proposta

# Objetivos da construção da solução

.

# Elaboração da jornada do usuário

Ao acessar a página inicial do website, um novo usuário deverá primeiramente realizar o cadastro de uma nova conta, pelo preenchimento de um formulário com campos de nome, sobrenome, endereço eletrônico e criação de uma senha. Como recurso de proteção contra a inscrição de contas falsas, é obrigatório ao novo usuário acessar um link enviado para o e-mail de cadastro para verificar sua identidade antes de poder autenticar-se.

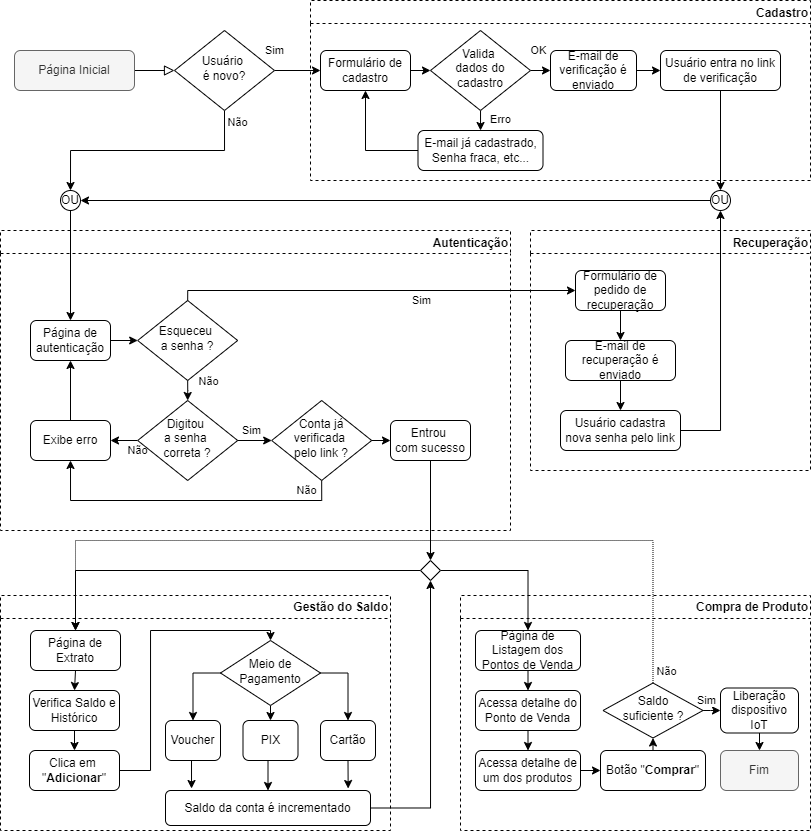
Um usuário possuidor de uma conta já verificada pode navegar da página inicial até a página de autenticação, onde deverá informar o endereço de e-mail associado à sua conta e a senha correspondente. Em caso de esquecimento da senha, poderá seguir o link apropriado para redefinição dela, o qual redireciona para um formulário solicitando informar o endereço de e-mail para o qual será enviado um link com instruções. Esse link levará o usuário a uma página onde uma nova senha poderá ser cadastrada.

Uma vez autenticado o usuário, terá acesso a um menu principal oferecendo as ações disponíveis. Uma jornada de uso típica pode começar pela consulta ao seu saldo financeiro pré-pago, e ao histórico de movimentações, como depósitos e compras realizadas. Tais informações estão disponíveis em uma página dedicada, onde também é oferecido um link que leva à área de adição de saldo. Nessa área, o usuário encontra alguns métodos de pagamentos possíveis para efetuar o depósito de valores que poderão ser utilizados na compra de produtos.

A partir do menu lateral também é possível acessar uma listagem de pontos de venda, cada qual descrito por título, endereço, fotos e média de notas das avaliações deixadas por outros usuários. Cada item da lista de pontos de venda redireciona a uma página do seu respectivo detalhamento onde, além dos dados básicos supracitados, também constam horários de funcionamento, e lista de produtos oferecidos no local. Ao usuário é facultado deixar sua própria avaliação nesta página, clicando no botão apropriado, que o leva a uma caixa de diálogo que recebe uma nota em estrelas e um comentário em texto.

Cada item da lista de produtos leva a uma página de detalhamento, onde constam título, descrição, fotos, preço e o botão “Comprar”. A jornada do usuário culmina na compra de um produto por meio deste botão. Após a confirmação explícita dessa intenção em uma caixa de diálogo, o preço do item é debitado da conta e um comando é enviado para o dispositivo *IoT* correspondente no espaço físico do ponto de venda, liberando o produto para o consumo do usuário.

**Figura 3.1** – Fluxograma detalhando a jornada do usuário.



# Apelo mercadológico da solução

# Ciclo de desenvolvimento da solução

Antes do início do desenvolvimento da solução foi definido um plano de projeto, que foi fielmente executado conforme relatado a seguir:

## Análise de Viabilidade e MVP

A partir da ideia inicial da aplicação foi realizado um *survey* na empresa, identificando os possíveis pontos de venda e seus respectivos produtos, máquinas e dispositivos utilizados no preparo, e a sua possibilidade de acionamento por dispositivos *IoT*. Adicionalmente foram conduzidas entrevistas com potenciais usuários da solução proposta para mapear o valor percebido e suas expectativas.

Essa análise culminou na seleção de um local mais adequado para a implantação do “piloto” do projeto. Foi escolhida uma cozinha com grande circulação de pessoas e frequentada pelos times mais exigentes em relação a qualidade do café que consomem. Esse público, por consequência, despende maiores investimentos em insumos e equipamentos. Decidiu-se por monetizar o preparo do café *espresso* de uma cafeteira italiana, que é de fácil conexão a um interruptor inteligente conectado à rede sem fio.

Também foi realizada a delimitação de escopo do produto mínimo viável. Para não incorrer em custos de operadora de meios de pagamento, foi decidido por inicialmente implementar apenas o pagamento por *voucher*, gerado por intermédio do Autor, atuando como intermediário dos pagamentos.

Visando acelerar o *time to market*, foi inicialmente postergado o desenvolvimento de um painel de administrador do sistema. Durante esse período as informações devem ser inseridas e monitoradas diretamente pelo banco de dados. Esses e outros aspectos complementares ficam pendentes como trabalhos futuros, conforme descrito no Capítulo 10.

## Elaboração da Arquitetura de Software

Uma vez definidos os requisitos do projeto, foi realizada uma pesquisa sobre as tecnologias apresentadas no Curso, selecionando as mais adequadas para a implementação da solução. O projeto de arquitetura buscou equilibrar o menor tempo de desenvolvimento e menores custos de hospedagem dos serviços (nesse primeiro momento), mas sem comprometer o potencial de escalabilidade futura. Estes aspectos estão mais bem descritos no Capítulo 7.

## Preparação dos Ambientes de Teste e Produção e Pipelines de CI/CD

A preocupação com a qualidade esteve presente desde o início do ciclo de desenvolvimento, o que levou à adoção de práticas de *DevOps*. A configuração inicial dos ambientes de teste e produção e da *pipeline* de *CI/CD* foi realizada antes do início da codificação das funcionalidades. Esses preparativos asseguraram que o desenvolvimento subsequente ocorresse em um fluxo controlado e confiável, como descrito no Capítulo 8.

## Desenvolvimento Incremental

O desenvolvimento da solução seguiu uma abordagem incremental, com a implementação das funcionalidades de forma sequencial, ordenada de maneira lógica por domínios, e priorizando os componentes essenciais para o MVP. O processo começou com a conexão ao banco de dados, seguida pela implementação de cadastro e autenticação de usuários, listagem de pontos de venda e seus respectivos produtos, fluxos de pagamento e extratos financeiros, e culminando com a funcionalidade de compra.

A configuração prévia dos ambientes de testes e *pipelines* de *CI/CD* possibilitou a execução de testes automatizados constantes, garantindo a integração contínua de novas funcionalidades. Além disso, foi possível realizar demonstrações frequentes para os usuários e *stakeholders* por meio do ambiente hospedado, o que facilitou a coleta de *feedback* em tempo real e a validação constante das funcionalidades implementadas.

# Mockup da proposta de solução

A ferramenta [*Figma*](https://www.figma.com/design/bAJduV9UPCEe9Aokj37ooP/TCC-Fullstack-PURCS) foi utilizada para a prototipação da interface gráfica da solução desenvolvida, cobrindo os fluxos de navegação tratados no Capítulo 3. Procurou-se oferecer uma interface minimalista e objetiva, respeitando aos padrões tradicionais de navegação e usabilidade familiares à maioria dos usuários.

Foi adotada a *MUI* como *design system* devido à sua rica oferta de componentes maduros e de fácil de uso, com vasta documentação e exemplos, integração no *Figma*, e por já ser familiar aos usuários, em vista de sua ampla difusão.

Optou-se por priorizar o design da versão *mobile*, uma vez que se espera que a solução seja utilizada por usuários em trânsito pelos pontos de venda e suas vizinhanças, sugerindo o menor uso da versão para *desktop*. Ainda assim, o design responsivo permite a utilização em dispositivos dos mais variados fatores de forma.

|  |  |  |
| --- | --- | --- |
| **Figura 6.1** - Sequência de páginas no fluxo de cadastro:  (a) formulário de dados cadastrais; (b) aviso sobre envio do e-mail de confirmação; (c) página acessada pelo link enviado no e-mail de confirmação. | | |
|  |  |  |
| **(a)** | **(b)** | **(c)** |
| **Figura 6.2** - Sequência de páginas do fluxo de entrada: (a) página inicial do *site*; (b) formulário de credenciais; (c) menu para navegação na área autenticada. | | |
|  |  |  |
| **(a)** | **(b)** | **(c)** |

|  |  |  |
| --- | --- | --- |
| **Figura 6.3** - Fluxo alternativo para recuperação de senha: (a) solicitação da recuperação; (b) confirmação do envio do link; (c) formulário para nova senha. | | |
|  |  |  |
| **(a)** | **(b)** | **(c)** |

|  |  |
| --- | --- |
| **Figura 6.4** -Páginas de finanças do usuário: (a) listagem de transações realizadas; (b) métodos de pagamento para adição de saldo. | |
|  |  |
| **(a)** | **(b)** |

|  |  |  |
| --- | --- | --- |
| **Figura 6.5.** Jornada de compra: (a) listagem dos pontos de venda; (b) detalhamento do ponto de venda; (c) detalhamento do produto. | | |
|  |  |  |
| **(a)** | **(b)** | **(c)** |

# Arquitetura de Software

# Validação da solução

Para garantir a qualidade e confiabilidade da solução, foi adotada uma série de boas práticas e tecnologias que proporcionaram um desenvolvimento robusto e consistente:

## Uso do *TypeScript*

O *TypeScript* foi utilizado em substituição ao *JavaScript*, garantindo maior confiabilidade no código, com tipagem estática que facilita a identificação de erros durante o desenvolvimento. Além disso, o *ESLint* foi configurado como *linter* para garantir a consistência do código, fazendo constantemente verificações de análise estática e dinâmica.

## Uso do *Next.js*

A escolha estratégica do *Next.js*, por utilizar o *Server-Side Rendering*, reduz o volume de dados que devem ser enviados ao cliente, o que minimiza o tamanho da *API*. Além de poupar linhas de código dedicadas aos objetos *DTO* e aos controladores, isso simplifica o gerenciamento de estados de *query* no cliente. Tudo isso colabora para reduzir potenciais pontos de falha.

## Testes Automatizados

Para a validação de funcionalidades ponta-a-ponta, foram implementados testes automatizados utilizando o *Playwright*. Essa ferramenta simula as interações do usuário através da emulação dos principais navegadores do mercado, e a cada etapa verifica o atendimento das expectativas e pós-condições para validar o comportamento esperado do sistema. Além de poupar tempo do desenvolvedor por meio da automação, ajuda a trazer maior robustez contra problemas de regressão.

## Ambiente de Desenvolvimento

Foi criado um projeto *Docker Compose* que levanta um ambiente local, incluindo um banco de dados configurado com dados pré-*seeded*, permitindo realizar os testes das funcionalidades em desenvolvimento com dados reprodutíveis e em um ambiente isolado. Essa configuração replica as condições de produção e subsidia o funcionamento do *Playwright* supracitado.

## Ambiente de Testes e Pipeline de CI/CD

A *pipeline* de *CI/CD* foi configurada usando *GitHub Actions*, proporcionando uma automação eficiente no processo de *build* e teste, e que executa a cada *push* ou *pull request*. A aprovação da *pipeline* é uma condição obrigatória para o *merge* de código na *branch* principal, garantindo que apenas código validado seja integrado.

O primeiro passo na *pipeline* é gerar um container para o *build* da aplicação *Next.js*. Em seguida, um segundo container hospeda o website enquanto o *Playwright* executa os testes automatizados. Esse processo utiliza o mesmo *Docker Compose* do ambiente de desenvolvimento, garantindo que o ambiente de testes esteja sempre alinhado com ele. Além disso, a natureza efêmera desses contêineres também assegura que os testes sejam independentes entre si e consistentes.

## Ambiente de Produção

Para o ambiente de produção, foi configurado um banco de dados na plataforma *MongoDB Atlas* e a hospedagem da aplicação *Next.js* na nuvem *Vercel*. A infraestrutura foi configurada para realizar *deploy* automático da *branch master* do repositório *GitHub* após cada *merge*, garantindo uma entrega contínua e sem interrupções apenas do código validado.

# Registros das evidências do projeto

Este trabalho de conclusão acompanha um portifólio de evidências do projeto desenvolvido, acessível pelos *links* abaixo relacionados:

**Quadro 9.1.** Listagem do material que acompanha este documento.

|  |  |
| --- | --- |
| **Artefatos** | **Link** |
| Website hospedado | <https://tcc-fullstack-pucrs-luizfeldmann.vercel.app> |
| Repositório do código-fonte | <https://github.com/luizfeldmann/tcc-fullstack-pucrs> |
| Design de UI em *Figma* | <https://tinyurl.com/luizfeldmann-figma-fullstack> |

# Considerações finais e expectativas

# Referências

Citar todas as referências utilizadas no trabalho