CENTRO UNIVERSITÁRIO SENAC

Bacharelado em Sistemas de Informação

PROJETO INTEGRADOR: ARQUITETURA ORIENTADA A SERVIÇOS

4° Semestre

Lucas Fregoneze Debastiani, Lucas Patrick Farias Figueiredo, Guilherme Cunha da Silva, Patrick Peixoto Violin, Vinicius Souza Monteiro

São Paulo

2024

Lucas Fregoneze Debastiani, Lucas Patrick Farias, Guilherme Cunha da Silva, Patrick Peixoto Violin, Vinicius Souza Monteiro

PROJETO INTEGRADOR: ARQUITETURA ORIENTADA A SERVIÇO

4° Semestre

Projeto Integrador apresentado ao curso Bacharelado em Sistemas de Informação, como requisito para obtenção de nota.

Professor Orientador: Clayton Mendonça Feliciano

São Paulo

2024

Resumo

O "Fit Finance" é um software inovador destinado a ajudar os usuários a gerenciar suas finanças e investimentos com precisão, facilitando o controle financeiro e a identificação de erros e acertos. Com uma interface intuitiva e funcionalidades robustas, o site permite a categorização de despesas, criação de orçamentos personalizados e geração de relatórios detalhados, promovendo uma vida financeira saudável e a prevenção de endividamentos. O projeto tem grande potencial de impacto social, contribuindo para a educação financeira, transparência e inclusão financeira, além de apoiar os Objetivos de Desenvolvimento Sustentável (ODS) da ONU, como a educação de qualidade e o crescimento econômico sustentável. No Brasil, onde a falta de educação financeira é um problema significativo, o "Fit Finance" se destaca como uma ferramenta essencial para melhorar a gestão financeira pessoal, reduzir o estresse financeiro e promover a independência econômica. O desenvolvimento do site focará na criação de uma interface clara e funcional, incluindo recursos como categorização de gastos baseada na Pirâmide de Maslow, conversão de moeda e visualização financeira de longo prazo, assegurando uma experiência de usuário excepcional e uma gestão financeira proativa.

Palavras-chave: Fit Finance, software, finanças pessoais, controle financeiro, investimentos, interface intuitiva, categorização de despesas, orçamentos personalizados, relatórios detalhados, impacto social, educação financeira, inclusão financeira, Objetivos de Desenvolvimento Sustentável, Brasil, gestão financeira pessoal, independência econômica, interface clara, categorização de gastos, Pirâmide de Maslow, conversão de moeda, visualização financeira.

Abstract

"Fit Finance" is an innovative software designed to help users manage their finances and investments accurately, facilitating financial control and identifying errors and successes. With an intuitive interface and robust features, the website allows for expense categorization, creation of personalized budgets, and generation of detailed reports, promoting a healthy financial life and preventing indebtedness. The project has great potential for social impact, contributing to financial education, transparency, and financial inclusion, as well as supporting the United Nations' Sustainable Development Goals (SDGs), such as quality education and sustainable economic growth. In Brazil, where lack of financial education is a significant problem, "Fit Finance" stands out as an essential tool for improving personal financial management, reducing financial stress, and promoting economic independence. The website development will focus on creating a clear and functional interface, including features such as expense categorization based on Maslow's Hierarchy of Needs, currency conversion, and long-term financial visualization, ensuring an exceptional user experience and proactive financial management.

Key-words: Fit Finance, software, personal finance, financial control, investments, intuitive interface, expense categorization, personalized budgets, detailed reports, social impact, financial education, financial inclusion, Sustainable Development Goals, Brazil, personal financial management, economic independence, clear interface, expense categorization, Maslow's Hierarchy of Needs, currency conversion, financial visualization.

Lista de Ilustrações

Sumário

1. Introdução

Fit Finance é um software que ajuda os usuários a entenderem com maior precisão suas finanças e sua carteira de investimentos, de forma que eles possam entender sua movimentação financeira, facilitando a visualização dos erros e acertos, trazendo maior controle, tudo isso por simplesmente adicionar suas finanças no sistema.

2. Visão Geral

O Web Site "Fit Finance" é uma ferramenta inovadora projetada para transformar a maneira como as pessoas lidam com suas finanças pessoais. Com uma interface intuitiva e funcionalidades poderosas, o site oferece um ambiente seguro e eficiente para o controle completo das finanças. Os usuários poderão categorizar despesas e criar orçamentos personalizados. Além disso, o "Fit Finance" fornecerá insights valiosos por meio de gráficos e relatórios detalhados, facilitando o entendimento das tendências financeiras individuais. Com o objetivo de promover uma vida financeira saudável, este site é uma ferramenta essencial para aqueles que desejam evitar endividamentos e alcançar seus objetivos financeiros com confiança e tranquilidade.

2.1 O Projeto e a Contribuição à Comunidade

Fit Finance tem um grande potencial para contribuir de forma significativa para a sociedade, melhorando o controle financeiro pessoal, promovendo a educação financeira, incentivando a transparência e responsabilidade, ampliando a inclusão financeira e reduzindo o desperdício financeiro. Sua extensibilidade o torna uma ferramenta versátil que pode ser adaptada para atender às necessidades específicas de diversas comunidades, tornando-o uma contribuição valiosa para a melhoria das condições financeiras e econômicas das pessoas.

2.2 ESG e as ODS contempladas no projeto

ODS 4 - Educação de Qualidade: Este objetivo está relacionado à promoção de educação inclusiva, equitativa e de qualidade para todos. O Fit Finance não apenas

ajuda os usuários a entender suas finanças, mas também funciona como uma ferramenta de educação financeira. Ao ensinar as pessoas a gerenciar suas finanças de maneira mais eficaz, ele contribui para elevar o nível de educação financeira da sociedade, capacitando as pessoas a tomar decisões financeiras mais informadas.

ODS 8 - Trabalho Decente e Crescimento Econômico: A ODS 8 visa promover o crescimento econômico sustentável, inclusivo e sustentável, com foco no emprego pleno e produtivo e no trabalho decente para todos. O projeto Fit Finance, ao ajudar as pessoas a gerenciar suas finanças pessoais de forma mais eficaz, contribui indiretamente para o crescimento econômico sustentável. Quando as pessoas têm um controle financeiro sólido, são menos propensas a enfrentar dificuldades financeiras, o que, por sua vez, pode levar a uma maior estabilidade no mercado de trabalho. Além disso, ao promover a educação financeira, o projeto pode ajudar a melhorar a empregabilidade das pessoas, capacitando-as para tomar decisões financeiras mais informadas, como investir em sua educação ou desenvolver habilidades relevantes.

2.3 Levantamento do problema

No Brasil, enfrentamos uma série de desafios relacionados à cultura financeira da população, refletindo-se em baixos índices de poupança e investimento. As pesquisas apontam para uma preocupante falta de educação financeira e planejamento entre os brasileiros, contribuindo para a ausência de reservas financeiras e investimentos. O desconhecimento sobre o controle de despesas pessoais é predominante, com a maioria das pessoas demonstrando pouca ou nenhuma compreensão sobre como gerenciar efetivamente suas finanças. Esse cenário é agravado pelo padrão de consumo descontrolado, onde o brasileiro tende a gastar mais do que ganha, não deixando espaço para a formação de uma reserva financeira e planejamento adequado para o futuro.

2.4 Justificativa

Relevância: A gestão financeira pessoal é uma preocupação universal. Pessoas de todas as idades, origens e situações econômicas podem se beneficiar

de um controle financeiro mais eficaz. Isso torna o projeto relevante para uma ampla gama de comunidades.

Impacto Social: Ao promover a gestão financeira responsável, o projeto ajuda a melhorar a qualidade de vida das pessoas, reduzindo o estresse financeiro e promovendo a independência econômica. Isso tem um impacto direto na sociedade, melhorando as condições de vida dos indivíduos e famílias.

Sustentabilidade Financeira: A capacidade de gerenciar o próprio dinheiro de forma eficaz é fundamental para a sustentabilidade financeira a longo prazo. Isso se alinha com o conceito de desenvolvimento sustentável e a erradicação da pobreza, conforme definido pelos ODS da ONU.

Empoderamento: O projeto empodera as pessoas, permitindo que elas tomem o controle de suas finanças. Isso é essencial para a promoção da autonomia e da dignidade das pessoas, conceitos que também estão ligados aos ODS.

Educação Financeira: A promoção da educação financeira é uma abordagem proativa para lidar com problemas financeiros. Isso ajuda as pessoas a evitar a armadilha da dívida e a fazer escolhas financeiras mais conscientes, o que está alinhado com o ODS da Educação de Qualidade.

2.5 Design do projeto

Página principal:

Figura 1 - Prototipação de média e alta fidelidade da Página Inicial

Fonte: Elaborado pelo autor (2024)

Pagina Home Logada:

Figura 1.1 - Prototipação de média fidelidade da Home Logada

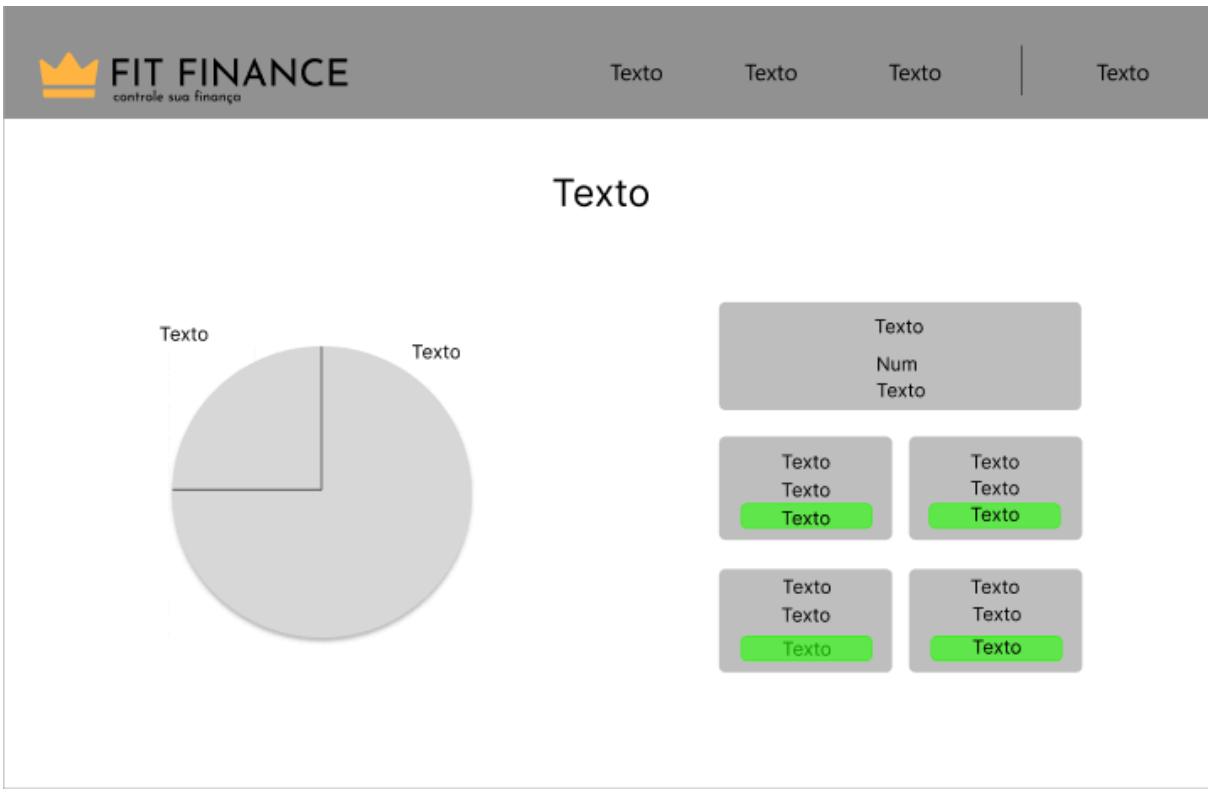
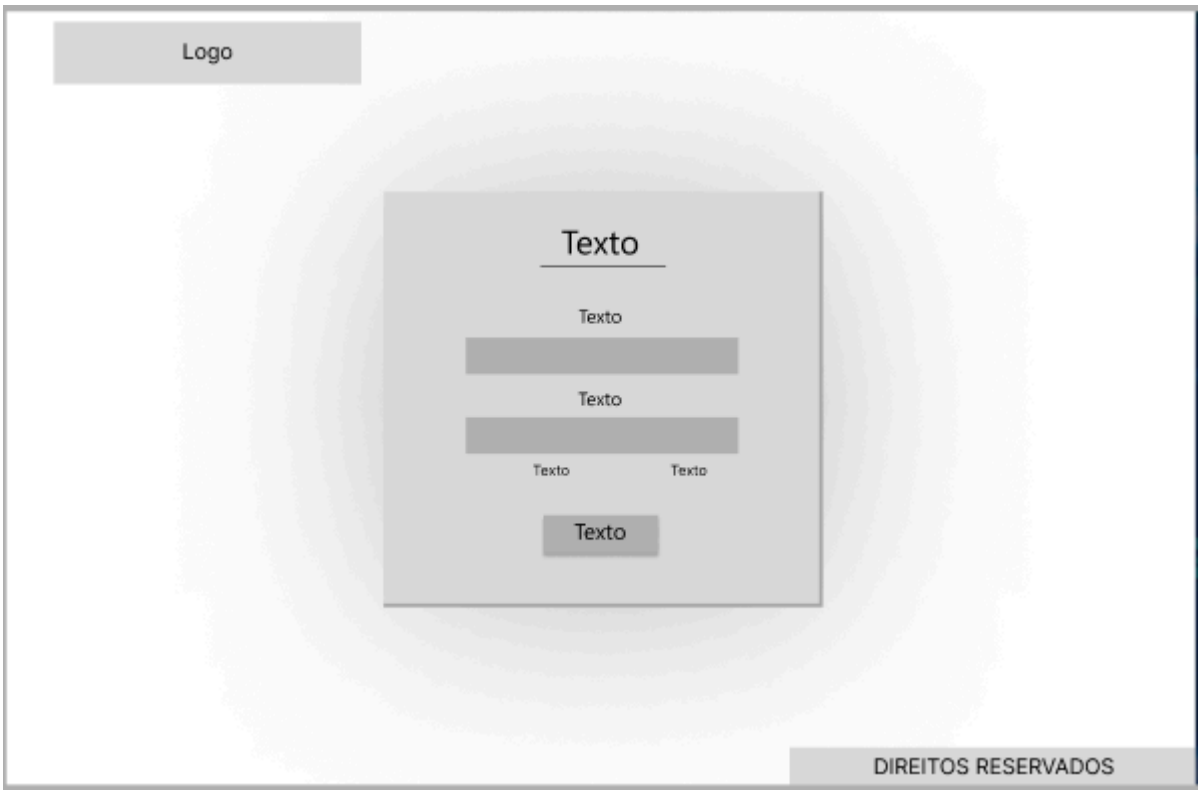


Figura 1.1 Prototipação de alta fidelidade da Página de Home Logada

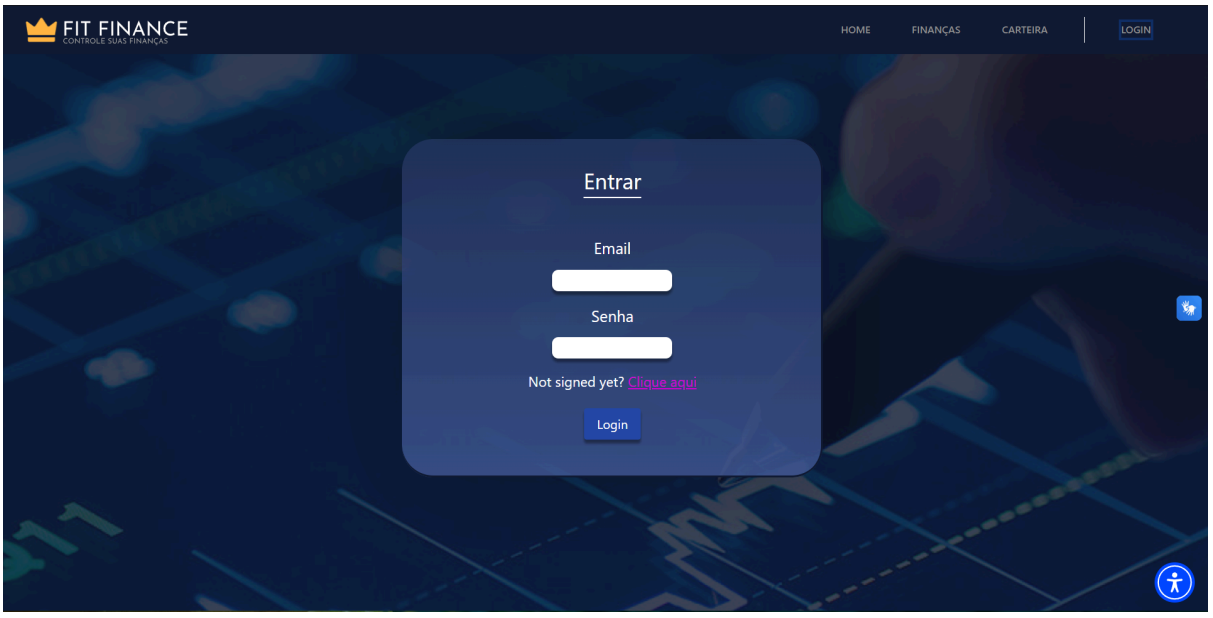
Página Login:

Figura 2 - Prototipação de média fidelidade da Página de Login



Fonte: Elaborado pelo autor (2024)

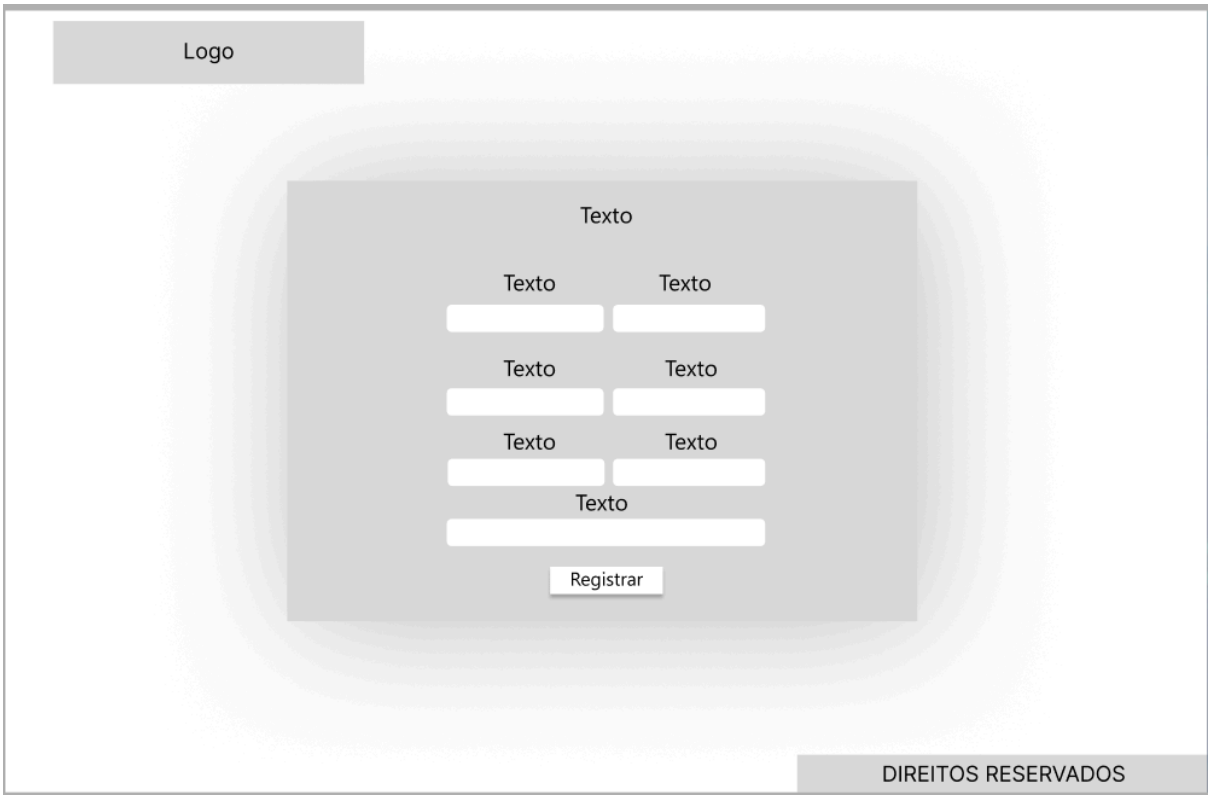
Figura 3 - Prototipação de alta fidelidade da Página de Login



Fonte: Elaborado pelo autor (2024)

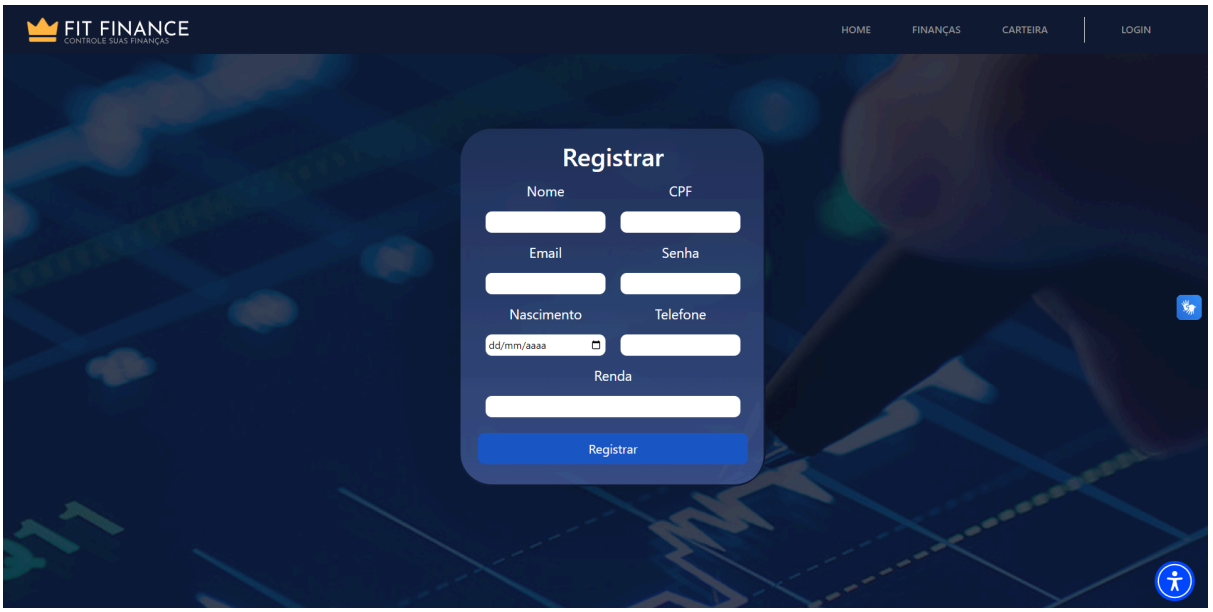
Página Registrar:

Figura 4 -Prototipação de média fidelidade da Página de Registro



Fonte: Elaborado pelo autor (2024)

Figura 5 - Prototipação de alta fidelidade da Página de Registro



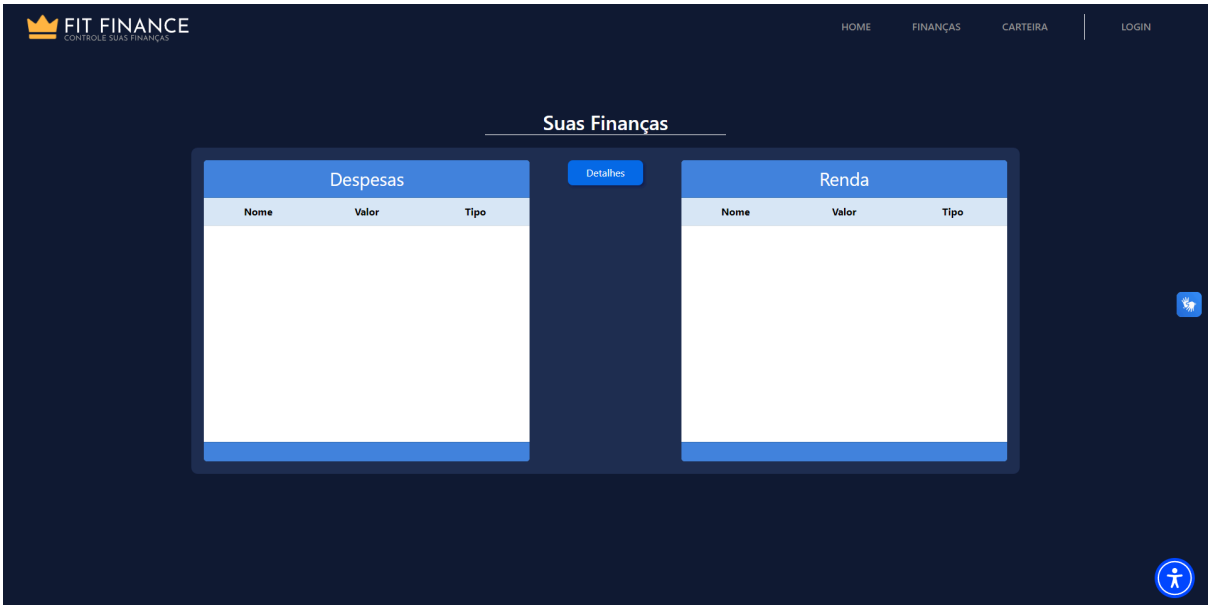
Fonte: Elaborado pelo autor (2024)

Página Finanças:

Figura 6 - Prototipação de média fidelidade da Página de Finanças

Fonte: Elaborado pelo autor (2024)

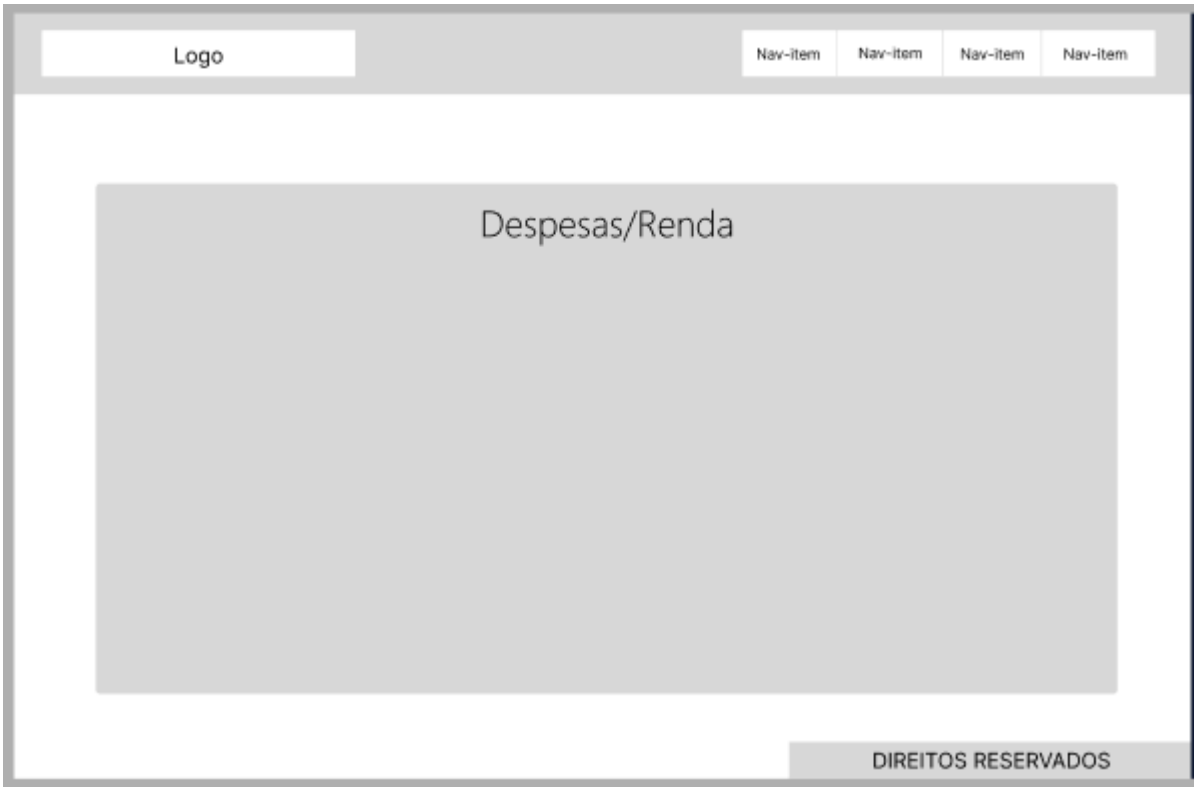
Figura 7 - Prototipação de alta fidelidade da Página de Finanças



Fonte: Elaborado pelo autor (2024)

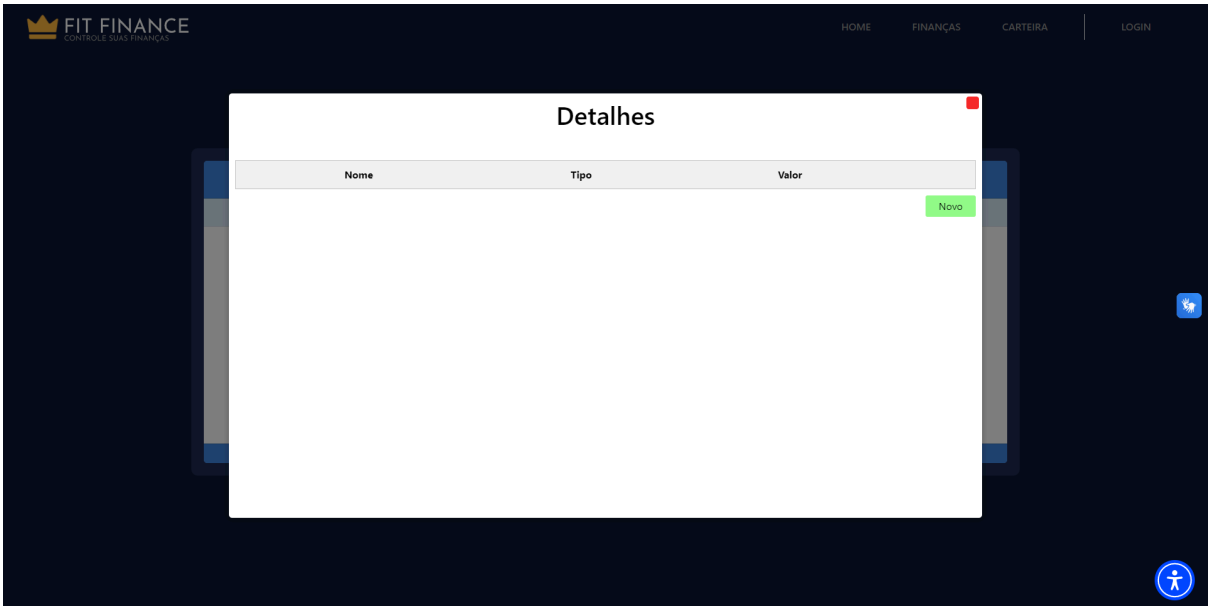
Página Finanças/Detalhes:

Figura 8 - Prototipação de média fidelidade da Página de Finanças/Detalhes



Fonte: Elaborado pelo autor (2024)

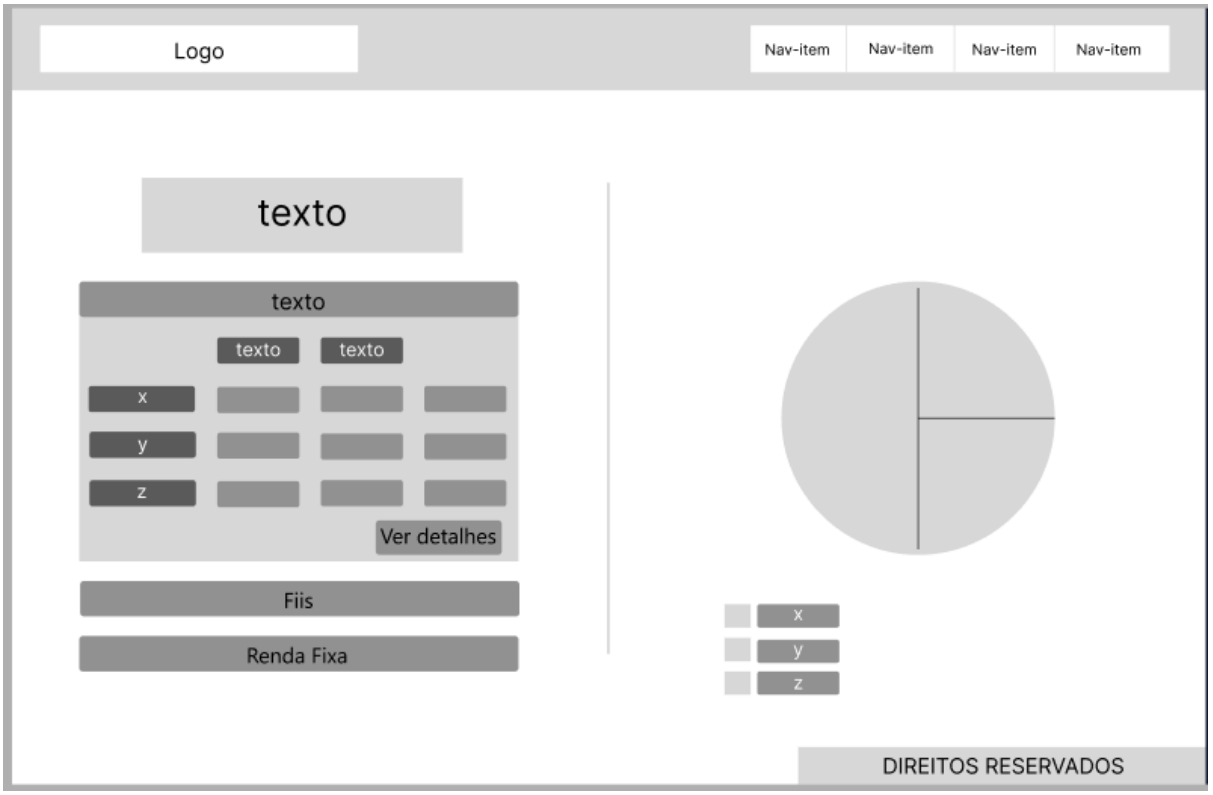
Figura 9 - Prototipação de alta fidelidade da Página de Finanças/Detalhes



Fonte: Elaborado pelo autor (2024)

Página Carteira:

Figura 10 - Prototipação de média fidelidade de Página da Carteira



Fonte: Elaborado pelo autor (2024)

Figura 11 - Prototipação de alta fidelidade de Página da Carteira

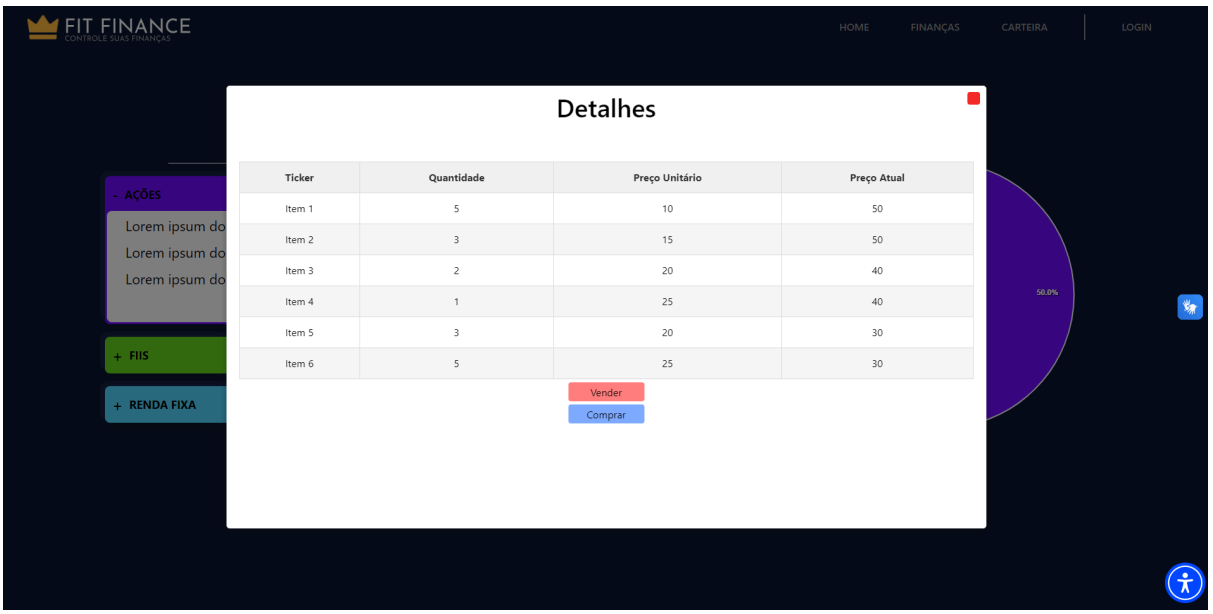
Fonte: Elaborado pelo autor (2024)

Página Carteira/Detalhes:

Figura 12 - Prototipação de média fidelidade da Página “Minha Carteira”

Fonte: Elaborado pelo autor (2024)

Figura 13 - Prototipação de alta fidelidade da Página “Minha Carteira”



Fonte: Elaborado pelo autor (2024)

3. Escopo do Projeto

O desenvolvimento do website "Fit Finance" tem como foco principal criar uma interface intuitiva e de fácil navegação, garantindo uma experiência excepcional para o usuário. Será crucial oferecer uma visão geral das finanças de forma clara e organizada, com informações concisas e relevantes. Recursos como controle de saldo por percentual e categorização baseada na Pirâmide de Maslow serão implementados para proporcionar uma abordagem única para entender e gerenciar gastos. A inclusão de funcionalidades como conversão de moeda e cálculo de impostos aumentará a utilidade do website em um contexto global. Além disso, um gráfico que exibe o mês atual e os próximos 6 meses será disponibilizado para oferecer uma visão de longo prazo e promover uma gestão financeira mais proativa. A ênfase na clareza, simplicidade e funcionalidade da interface e da experiência do usuário será fundamental para o sucesso e eficácia do website "Fit Finance".

3.1 Tecnologias Utilizadas

Frontend: HTML 5, CSS 3, Bootstrap 5 e JavaScript (com React JS)
Backend: Java e Spring
Banco de dados: MySQL

3.2 Matriz de papeis de responsabilidades

Guilherme Cunha da Silva: Desenvolvimento do Software
Lucas Fregoneze Debastiani: Documentação do Projeto
Lucas Patrick Farias Figueiredo: Parte criativa/Design
Patrick Peixoto Violin: Desenvolvimento do Software
Vinicius Souza Monteiro: Parte criativa/Design

4. Cliente

Nome: Bem gasto org.

Ramo de Atividade: A Bem Gasto é um projeto sem fins lucrativos cujo objetivo é transformar a vida das pessoas através da Educação Financeira. Sem educação financeira, fica mais difícil conquistar sonhos e até mesmo suprir as necessidades do dia a dia. A Bem Gasto é uma instituição reconhecida pela Estratégia Nacional de Educação Financeira (ENEF), do Banco Central.

O website “Fit Finance" é ideal para uma ampla gama de usuários que buscam aprimorar sua gestão financeira de maneira eficiente e personalizada. Destina-se a indivíduos de todas as idades e perfis socioeconômicos que desejam ter um controle mais preciso e consciente de suas finanças pessoais. Desde jovens profissionais em início de carreira até funcionários experientes, o website atende a uma audiência diversificada. Além disso, é especialmente valioso para viajantes frequentes e profissionais que lidam com moedas estrangeiras, oferecendo uma solução prática para a conversão de moeda. Com sua abordagem inclusiva e recursos abrangentes, o “Fit Finance" se destaca como uma ferramenta versátil e essencial para qualquer pessoa comprometida em alcançar estabilidade financeira e evitar endividamentos.

5. Descrição dos Requisitos

5.1 Requisitos Funcionais

RF1: O sistema deve permitir o registro de transações, permitindo aos usuários adicionar e categorizar suas despesas e receitas, fornecendo opções detalhadas de classificação.
RF2: O sistema deve oferecer uma tela inicial que apresente um resumo claro e conciso do estado financeiro atual, incluindo saldo total, receitas e despesas.
RF3: O sistema deve possibilitar aos usuários alocar porcentagens específicas do saldo para diferentes categorias de gastos, facilitando o planejamento financeiro.
RF4: O sistema deve integrar um sistema de categorização inspirado na pirâmide de necessidades de Maslow, para uma visão holística dos gastos em áreas como

necessidades básicas, segurança, crescimento pessoal, entre outros.
RF5: O sistema deve disponibilizar uma funcionalidade que permita aos usuários converter valores entre diferentes moedas, com atualizações em tempo real.
RF6: O sistema deve integrar uma ferramenta que calcule estimativas de impostos sobre as transações, oferecendo uma visão mais precisa do impacto fiscal.
RF7: O sistema deve gerar relatórios gráficos e textuais para oferecer aos usuários uma compreensão mais profunda de seus padrões de gastos ao longo do tempo.
RF8: O sistema deve permitir a configuração de alertas para datas de vencimento de contas, metas de economia ou gastos excessivos em determinadas categorias.
RF9: O sistema deve fornecer um gráfico que exibe uma projeção financeira para o mês atual e os próximos seis meses, com base nos padrões de gastos atuais.
RF10: O sistema deve implementar medidas robustas de segurança, como autenticação de dois fatores e criptografia de dados, para proteger as informações financeiras dos usuários.

5.2 Requisitos Não Funcionais

RNF1: Acessibilidade de Tela: O website deve ser compatível com leitores de tela populares, como TalkBack, para garantir que usuários com deficiência visual possam navegar e interagir com facilidade.
RNF2: Contraste de Cores: O website deve seguir as diretrizes de acessibilidade de cores, garantindo um contraste adequado para facilitar a leitura e a identificação de elementos para usuários com deficiência visual.
RNF3: Legibilidade de Fontes: As fontes e tamanhos de texto devem ser configurados de forma a garantir uma leitura clara e fácil para todos os usuários, incluindo aqueles com dificuldades de visão.
RNF4: Navegação Intuitiva: A navegação no website deve ser intuitiva e coerente, permitindo que os usuários localizem facilmente funcionalidades e informações importantes.
RNF5: Compatibilidade com Tamanho de Tela Variado: O website deve se adaptar de forma eficaz a diferentes tamanhos de resoluções.
RNF6: Compatibilidade com diferentes navegadores: O website deve ser compatível com uma ampla variedade de versões dos sistemas operacionais, para atender a uma ampla base de usuários.

RNF7: Tempo de Resposta Rápido: O tempo de resposta do website, desde a interação do usuário até a execução da ação correspondente, deve ser mínimo para proporcionar uma experiência fluida e sem interrupções.
RNF8: Requisitos de Armazenamento Mínimo: O website deve ser otimizado para consumir uma quantidade razoável de espaço de armazenamento no dispositivo do usuário, garantindo que não haja sobrecarga desnecessária.
RNF9: Comportamento em Modo Offline: O website deve salvar as últimas alterações que o usuário fez antes da perda da conexão com a internet, garantindo que os usuários possam manter informações seguras sem conexão à internet.
RNF10: Atualizações sem interrupções: Atualizações de software devem ser implementadas de forma a não interromper ou comprometer a experiência do usuário durante o processo de atualização.
RNF11: Segurança de Dados: O website deve adotar práticas de segurança sólidas para proteger as informações dos usuários contra acessos não autorizados ou vazamentos de dados.
RNF12: Feedback Visual e Sonoro: O website deve fornecer feedback claro e perceptível ao usuário em forma de indicadores visuais e sonoros para confirmar ações ou alertar sobre erros.
RNF13: Compatibilidade com Recursos de Acessibilidade Web: O website deve aproveitar ao máximo os recursos de acessibilidade fornecidos pelo sistema operacional, para melhorar a experiência do usuário.
RNF14: Tempo de Carregamento Rápido: O website deve ser otimizado para iniciar rapidamente, minimizando o tempo que os usuários precisam esperar para começar a usá-lo.
RNF15: Compatibilidade com Dispositivos de Entrada Diversificados: O website deve suportar diferentes métodos de entrada, como teclado, mouse e comandos de voz, para garantir acessibilidade a todos os tipos de usuários.

5.3 Regras de Negócio

RN1: Autenticação Segura: O sistema deve garantir que o processo de autenticação dos usuários seja seguro, utilizando métodos de criptografia e proteção contra tentativas de acesso não autorizado.
RN2: Política de Privacidade e Proteção de Dados: O website deve seguir as

regulamentações de privacidade e proteção de dados, fornecendo aos usuários informações claras sobre como suas informações serão coletadas, armazenadas e utilizadas.

RN3: Manutenção de Backup de Dados: O sistema deve realizar regularmente backups dos dados dos usuários para garantir a segurança e a integridade das informações em caso de falhas ou incidentes.

RN4: Respeito aos Termos de Serviço: Os usuários devem concordar com os termos de serviço e políticas de uso do website antes de utilizá-lo, assegurando conformidade com as diretrizes estabelecidas pela empresa.

RN5: Suporte ao Cliente Eficiente: O sistema deve fornecer canais de suporte ao cliente eficazes, incluindo opções de contato direto e respostas rápidas para solucionar dúvidas ou problemas dos usuários.

RN6: Atualizações Regulares e Melhorias: A equipe de desenvolvimento deve realizar atualizações regulares no website, introduzindo melhorias de desempenho, novas funcionalidades e correções de bugs para manter a qualidade do serviço.

RN7: Política de Reembolsos e Cancelamentos: O website deve estabelecer uma política clara e justa para reembolsos e cancelamentos de assinaturas ou transações, fornecendo aos usuários informações detalhadas sobre os procedimentos necessários.

RN8: Comunicação Transparente: A empresa deve manter uma comunicação transparente com os usuários, informando sobre mudanças significativas no website, atualizações de políticas ou qualquer outra informação relevante.

RN9: Compatibilidade com Legislação Local: O website deve estar em conformidade com as leis e regulamentações locais, garantindo que todas as atividades e transações realizadas estejam de acordo com os requisitos legais do país de operação.

RN10: Garantia de Disponibilidade do Serviço: A empresa deve assegurar uma alta disponibilidade do serviço, minimizando períodos de inatividade e mantendo uma infraestrutura robusta para atender à demanda dos usuários.

6. Modelo de Casos de Uso

6.1 Identificação dos Atores e suas Responsabilidades

Cliente: Este ator é o principal encarregado de interagir com o software de forma a

utilizar as funcionalidades oferecidas pelo "Fit Finance". Suas responsabilidades incluem:

- Acessar e navegar pelas diferentes seções do website.
- Inserir e atualizar dados pessoais e financeiros.
- Utilizar as ferramentas de categorização de despesas.
- Estabelecer e modificar orçamentos personalizados.
- Analisar gráficos e relatórios detalhados para obter insights financeiros e receber notificações e alertas relevantes para sua situação financeira.
- MySQL: Atuando como uma parte crucial do sistema, o ator MySql é responsável por:
- Gerenciar a autenticação e autorização de usuários.
    - Facilitar a comunicação em tempo real entre o website e o servidor. Garantir a segurança e integridade dos dados do "Fit Finance".

6.2 Definição de Prioridade de Desenvolvimento dos Casos de Uso

A prioridade de desenvolvimento dos casos de uso continuará sendo determinada com base em critérios como impacto na experiência do usuário, importância para a funcionalidade central do website e relevância para os objetivos financeiros dos usuários. A adição do ator MySql será considerada para garantir uma integração eficiente e segura.

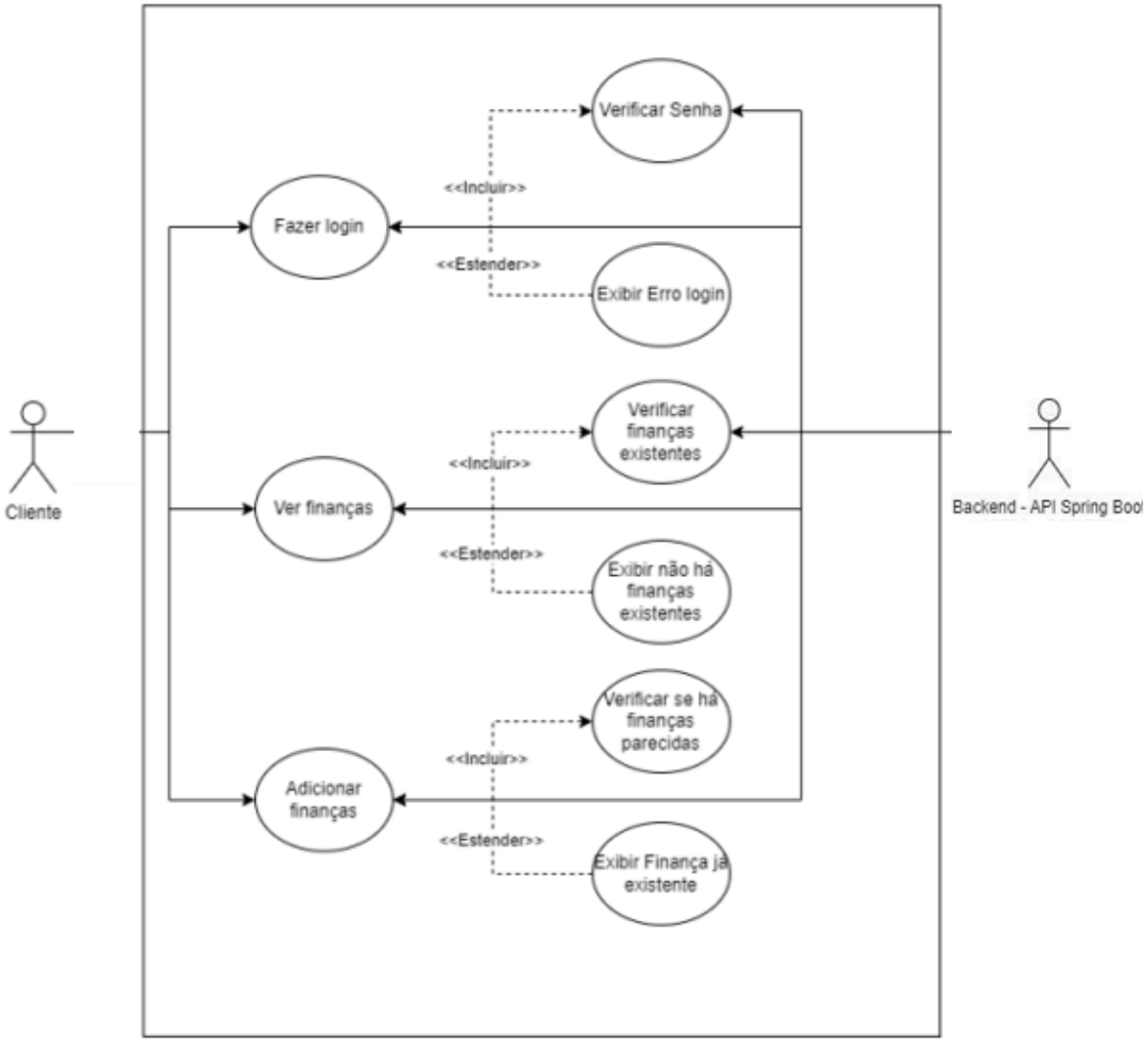
- Impacto na Segurança: Casos de uso relacionados à integração e segurança do MySql terão uma prioridade elevada, garantindo a proteção robusta dos dados e a autenticação segura dos usuários.
- Integração Fluida: Casos de uso que envolvem a interação entre o cliente e o MySql, como a sincronização eficiente de dados em tempo real, terão prioridade para oferecer uma experiência de usuário contínua.
- Funcionalidades Centrais: Casos de uso essenciais para a funcionalidade principal do website, como categorização de despesas e análise de tendências financeiras, permanecerão como prioridade para garantir a utilidade fundamental do "Fit Finance".
- Experiência do Usuário: Casos de uso que melhoram diretamente a

experiência do usuário, como notificações push relevantes, podem ser priorizados para proporcionar uma interação mais agradável e engajadora.

- Feedback dos Usuários: Consideração especial será dada aos casos de uso que recebem feedback positivo dos usuários, buscando atender às expectativas e demandas da comunidade "Fit Finance".

6.3 Diagrama de Casos de Uso

Figura 14 - Diagrama de Caso de Uso



Fonte: Elaborado pelo autor (2024)

6.4 Descrição Detalhada dos Casos de Uso

1. Autenticação do Usuário:
    - Pré-condições: O cliente inicia o website.
    - Passos Principais:
        1. O cliente insere suas credenciais.
        2. O MySql autentica as credenciais do cliente.
    - Pós-condições: O cliente obtém acesso seguro ao "Fit Finance".

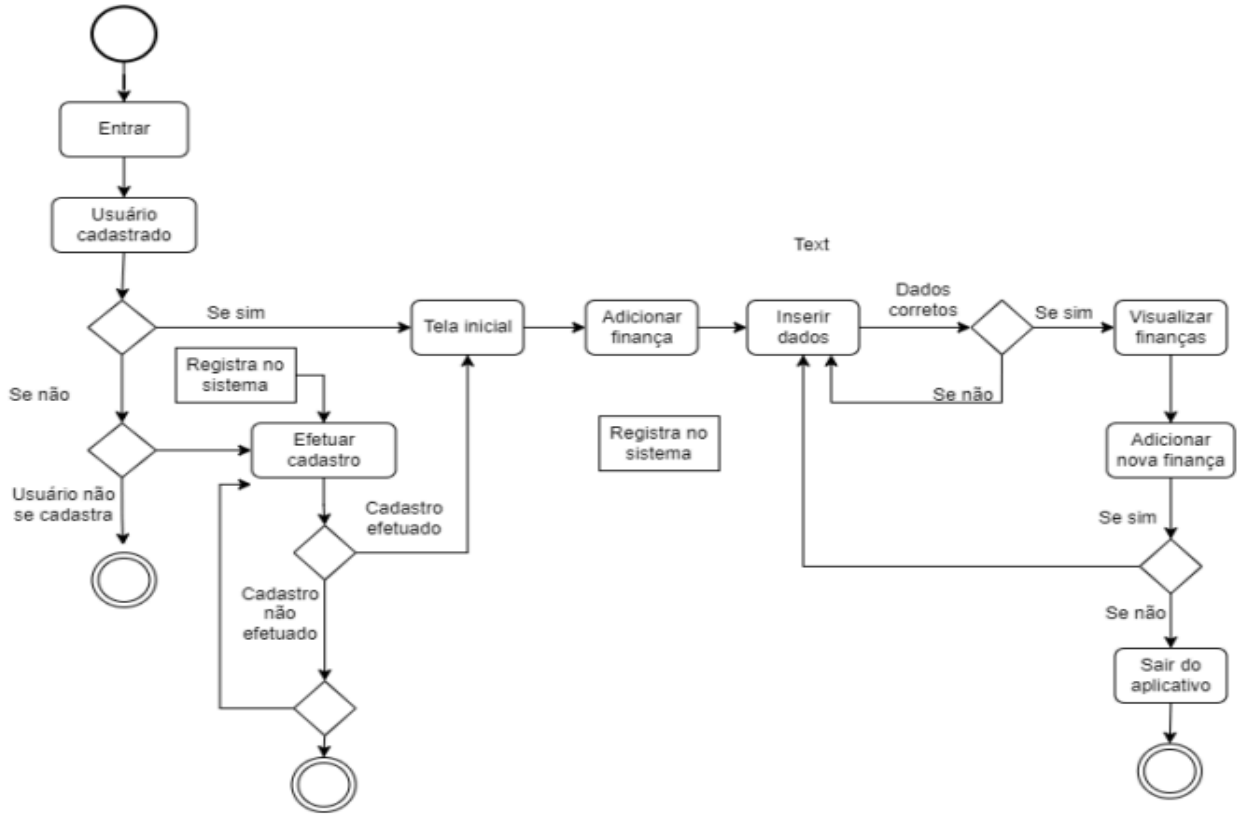
2. Armazenamento de Dados Financeiros:
    - Pré-condições: O cliente realiza uma atualização nas informações financeiras.
    - Passos Principais:
        1. O cliente insere novos dados financeiros.
        2. O MySql armazena os dados de forma segura.
    - Pós-condições: As informações financeiras são atualizadas no sistema.

6.5 Banco de Dados

No desenvolvimento do software Fit Finance, optou-se pelo MySQL como o banco de dados principal. Essa escolha foi feita visando assegurar eficiência e segurança na gestão dos dados financeiros. Principalmente pela robustez, escalabilidade e pelos recursos avançados oferecidos, como autenticação, armazenamento em nuvem e notificações em tempo real. Com isso, é possível proporcionar aos usuários uma experiência fluida e segura durante o gerenciamento de suas finanças por meio deste software.

6.5.1 Diagrama de Atividades

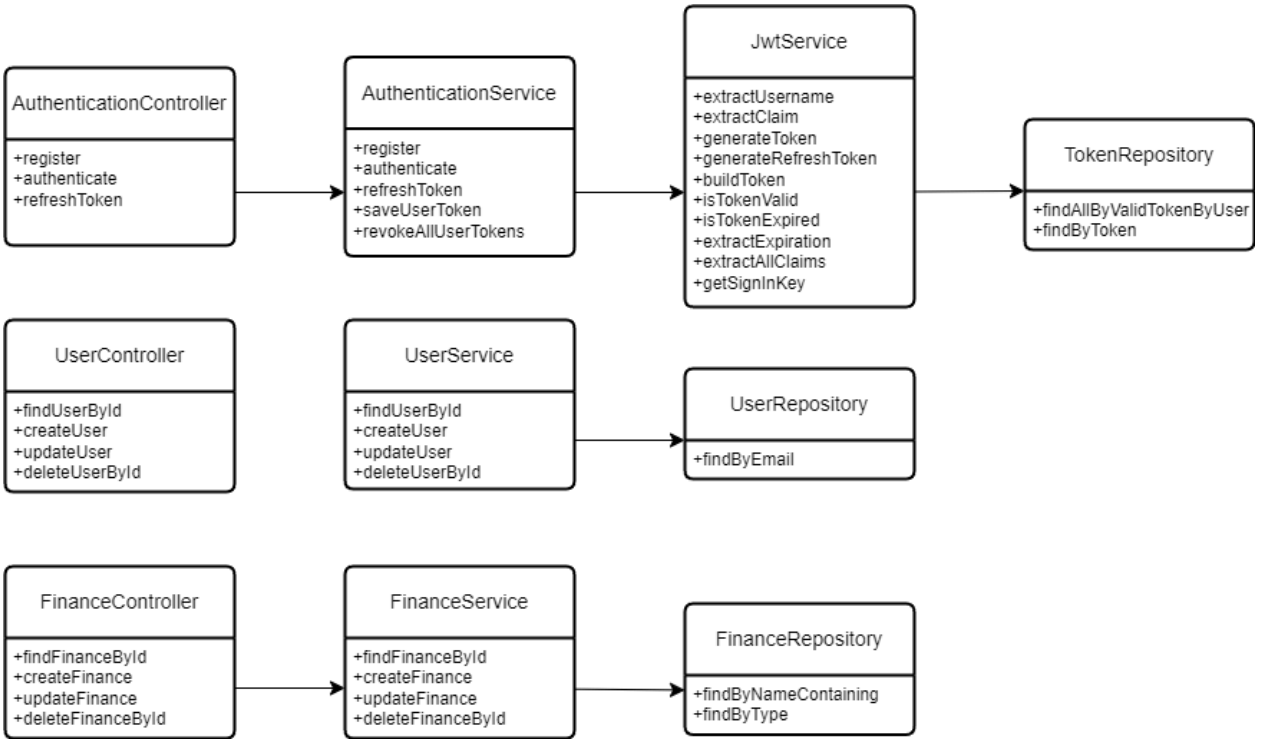
Figura 15 - Diagrama de Atividades



Fonte: Elaborado pelo autor (2024)

6.5.2 Diagrama de Classes Conceitual

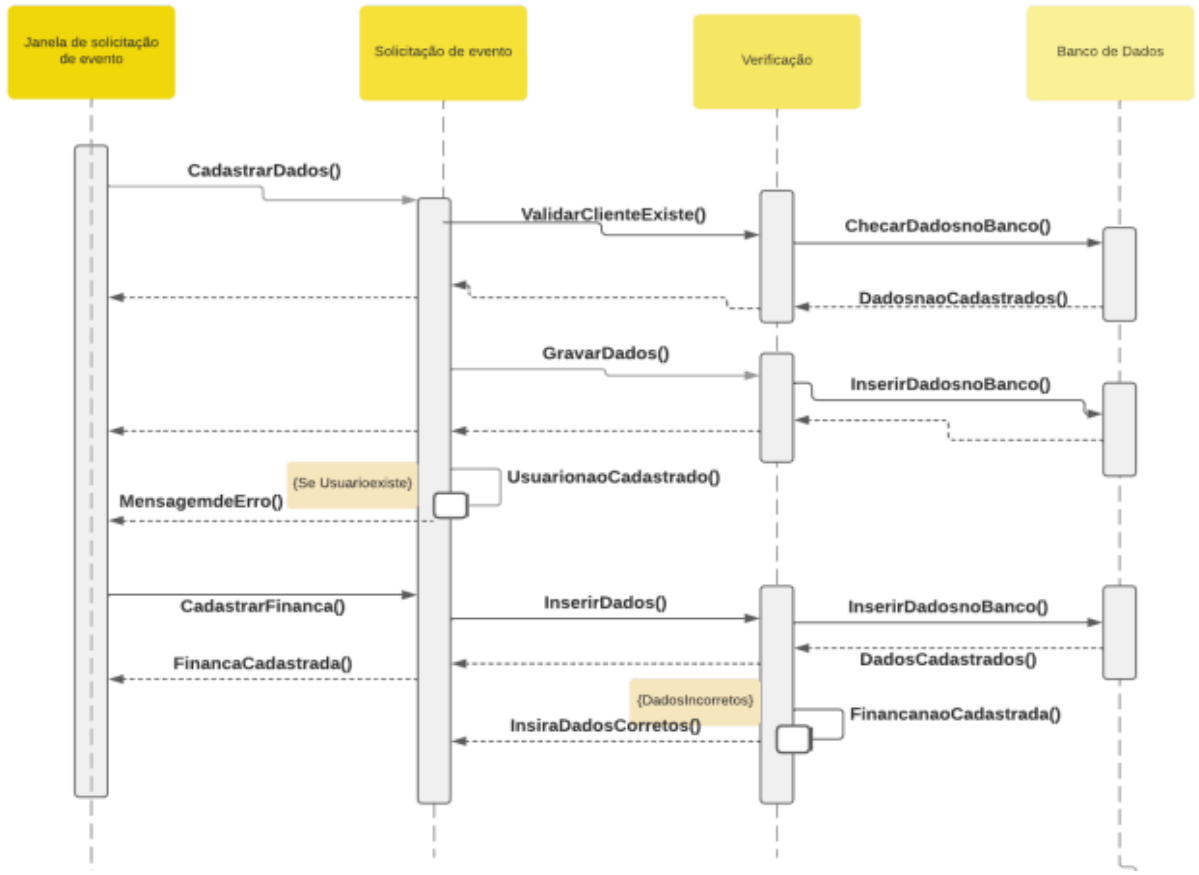
Figura 16 - Diagrama de Classes Conceituais



Fonte: Elaborado pelo autor (2024)

6.5.3 Diagrama de Sequência

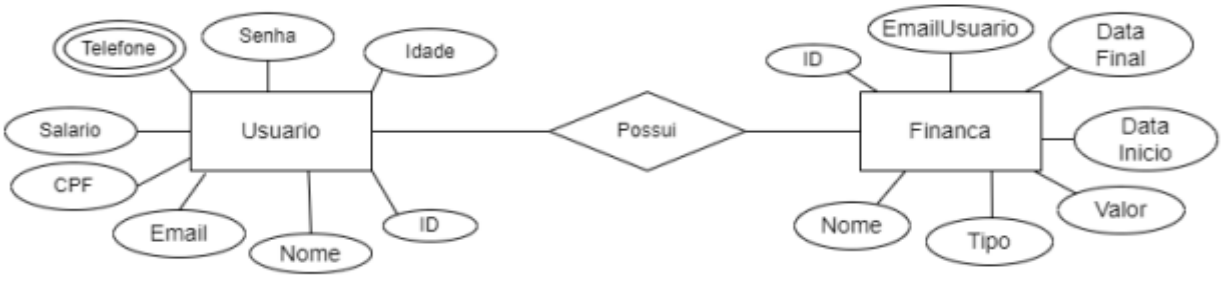
Figura 17 - Diagrama de Sequência



Fonte: Elaborado pelo autor (2024)

6.5.4 DER

Figura 18 - DER



Fonte: Elaborado pelo autor (2024)

7. Pesquisa de tecnologia

7.1 Comparação com Outras Stacks

- **Node.js**

JavaScript: Usar JavaScript no backend (Node.js) e no frontend (React) pode ser vantajoso pela consistência, mas Node.js não oferece a mesma robustez e ferramentas empresariais que o ecossistema Java/Spring.

Single-Threaded: Node.js é single-threaded e pode enfrentar dificuldades com operações CPU-intensivas, enquanto Java pode se beneficiar de multithreading para melhor performance em sistemas financeiros.

- **Python (Django/Flask)**

Django: É excelente para desenvolvimento rápido e possui uma interface administrativa out-of-the-box, mas pode não oferecer a mesma performance que Java em aplicações de grande escala.

Performance: Java geralmente supera Python em termos de performance bruta, o que pode ser crucial para sistemas financeiros que exigem alta performance.

- **Ruby on Rails**

Facilidade de Uso: Rails é conhecido pela sua facilidade de uso e desenvolvimento rápido, mas pode não oferecer a mesma escalabilidade e performance que Java/Spring em aplicações empresariais.

Comunidade: Embora Rails tenha uma comunidade ativa, o ecossistema Java/Spring é mais vasto, especialmente para aplicações empresariais.

7.2 Escolha da Tecnologia

1. Java com Spring Boot

Spring Boot é um framework baseado em Java, usado para criar aplicações robustas e escaláveis. Ele simplifica o desenvolvimento com configuração mínima e fornece diversas funcionalidades para criar APIs RESTful.

Principais vantagens:

- Inicialização rápida com configurações automáticas.
- Suporte a várias dependências através do Spring Boot Starter.
- Ferramentas integradas para segurança, acesso a dados…

2. React

React é uma biblioteca JavaScript popular para construir interfaces de usuário (UI) dinâmicas e responsivas. Ele permite criar componentes reutilizáveis e gerenciar o estado de forma eficiente.

Principais vantagens:

- Criação de componentes reutilizáveis.
- Atualizações eficientes e rápidas na UI.
- Grande ecossistema e comunidade ativa.

3. MySQL

MySQL é um sistema de gerenciamento de banco de dados relacional (RDBMS) amplamente utilizado. Ele é conhecido por ser rápido, confiável e fácil de usar.

Principais vantagens:

- Suporte a transações ACID.
- Alta performance para operações de leitura e escrita.
- Grande compatibilidade com diversas linguagens de programação.

7.3 Softwares semelhantes

Em softwares semelhantes temos os seguintes websites que podem servir como concorrentes ao nosso projeto.

Meu Dinheiro - O Meu Dinheiro é um gerenciador financeiro online, ágil e completo, em diferentes moedas, como R$, USD$, Euro (€) e outros, para um eficiente controle financeiro pessoal, familiar ou de uma pequena empresa.

Organizze - O Organizze prega ser um gerenciador financeiro de verdade com transparência e sem oferecer serviços externos. oferece funcionalidades como a categorização de despesas, configuração de orçamentos, visualização de relatórios financeiros e acompanhamento de saldos de contas e cartões de crédito. Tem como crença fornecer uma visão clara das entradas e saídas de dinheiro, de modo a ajudar a identificar oportunidades de economia e a planejar para futuros investimentos ou despesas.

Orçamento pessoal - Consiste em um sistema com o foco em controle de finanças pessoais, conta também com versão para empresas. Categoriza as finanças por tipo de conta (Poupança e Corrente), além de conter relatórios e estatísticas sobre os gastos.

7.4 Tecnologias empregadas

7.4.1 Java

Java é uma linguagem de programação de propósito geral, orientada a objetos, e desenvolvida por James Gosling e sua equipe na Sun Microsystems em 1995. Desde então, ela se tornou uma das linguagens mais populares e amplamente utilizadas no mundo, especialmente em aplicações empresariais e sistemas de grande escala. Java foi projetada para ser robusta, segura e de alto desempenho, proporcionando um ambiente de desenvolvimento e execução confiável.

1. **Orientação a Objetos:**

Java é uma linguagem puramente orientada a objetos. Isso significa que conceitos como herança, polimorfismo, encapsulamento e abstração são fundamentais. A programação orientada a objetos ajuda a criar sistemas modulares e reutilizáveis, facilitando a manutenção e a evolução do software.

2. **Portabilidade:**

Um dos maiores pontos de venda do Java é sua portabilidade. O slogan "write once, run anywhere" (escreva uma vez, execute em qualquer lugar) é possível graças à Java Virtual Machine (JVM). O código Java é compilado em bytecode, que pode ser executado em qualquer dispositivo que possua uma JVM, independentemente da arquitetura do sistema operacional.

3. **Segurança:**

Java possui diversas características de segurança embutidas, como a verificação de bytecode em tempo de execução, gerenciamento automático de memória, e a ausência de ponteiros explícitos, que ajudam a evitar uma série de vulnerabilidades comuns em linguagens como C e C++.

4. **Multithreading:**

O suporte a multi-threading nativo em Java permite a execução de múltiplas threads simultaneamente. Isso é crucial para aplicações que precisam realizar várias tarefas ao mesmo tempo, como servidores web e sistemas financeiros.

5. **Bibliotecas e Frameworks:**

Java tem uma vasta coleção de bibliotecas padrão (Java Standard Library) e frameworks de terceiros que aceleram o desenvolvimento de aplicações. Estes incluem tudo, desde bibliotecas para manipulação de dados (como JPA/Hibernate) até frameworks web (Spring Framework).

- **Vantagens de Utilizar Java em um Sistema de Gestão de Finanças Pessoais**

Desenvolver um sistema de gestão de finanças pessoais requer uma linguagem que seja confiável, segura e capaz de lidar com operações complexas de maneira eficiente. Java se destaca como uma escolha ideal para esse tipo de aplicação por várias razões.

1. Robustez e Confiabilidade

A. Histórico de Uso em Aplicações Críticas

Java é amplamente utilizado em setores onde a confiabilidade é crítica, como bancos, telecomunicações e comércio eletrônico. Isso se deve à sua robustez e capacidade de lidar com grandes volumes de transações de maneira segura e eficiente.

B. Detecção de Erros em Tempo de Compilação

A tipagem estática e o rigoroso sistema de tipos de Java ajudam a detectar muitos erros em tempo de compilação, antes mesmo de o código ser executado. Isso resulta em menos bugs e maior estabilidade, o que é vital para sistemas financeiros onde a precisão é essencial.

2. Segurança

A. Recursos de Segurança Embutidos

Java foi projetado com a segurança em mente. Ele inclui várias características de segurança, como a verificação de bytecode em tempo de execução e o gerenciamento automático de memória, que ajudam a proteger contra vulnerabilidades comuns.

B. Bibliotecas de Segurança Avançadas

Java possui um rico conjunto de bibliotecas e frameworks para segurança, como Spring Security, que facilita a implementação de autenticação e autorização. Essas ferramentas ajudam a garantir que apenas usuários autorizados possam acessar dados financeiros sensíveis, protegendo a integridade e a confidencialidade das informações.

3. Escalabilidade e Performance

A. Suporte a Multithreading

O suporte nativo a multithreading em Java permite que o sistema execute múltiplas operações simultaneamente, como processamento de transações e consultas ao banco de dados, sem comprometer a performance.

B. Ferramentas de Monitoramento e Otimização

Java oferece várias ferramentas para monitoramento e otimização de performance, como JMX (Java Management Extensions) e JMH (Java Microbenchmark Harness). Essas ferramentas ajudam a garantir que o sistema mantenha alta performance mesmo sob carga intensa.

4. Portabilidade

A. Independência de Plataforma

A capacidade de Java de ser executado em qualquer plataforma que suporte a JVM permite que o sistema de gestão de finanças pessoais seja acessível de qualquer dispositivo, seja ele um desktop, laptop, ou dispositivo móvel.

B. Integração com Dispositivos Móveis

Java é a base para o desenvolvimento de aplicativos Android, o que significa que um

sistema de gestão de finanças pessoais desenvolvido em Java pode ser facilmente adaptado para dispositivos móveis, ampliando seu alcance e acessibilidade.

5. Ecossistema Rico e Comunidade Ativa

A. Ampla Disponibilidade de Bibliotecas e Frameworks

O ecossistema Java é vasto, com uma infinidade de bibliotecas e frameworks disponíveis para praticamente qualquer necessidade, desde acesso a dados (JPA/Hibernate) até construção de interfaces gráficas (JavaFX, Swing). Isso acelera o desenvolvimento e reduz a necessidade de reinventar a roda.

B. Suporte e Recursos Educativos

A comunidade de desenvolvedores Java é uma das maiores e mais ativas do mundo. Isso significa que é fácil encontrar suporte, tutoriais, e recursos educativos, além de uma vasta quantidade de código de exemplo e bibliotecas open-source que podem ser reutilizadas.

Optar por Java para desenvolver um sistema de gestão de finanças pessoais oferece inúmeras vantagens em termos de robustez, segurança, escalabilidade, portabilidade e suporte da comunidade. A capacidade de Java de lidar com operações complexas e processar grandes volumes de dados de forma eficiente faz dela uma escolha ideal para aplicações financeiras. Além disso, a maturidade da linguagem e a riqueza de seu ecossistema garantem que os desenvolvedores tenham acesso às melhores ferramentas e práticas, resultando em um sistema confiável e duradouro.

7.4.2 Spring Boot

Spring Boot é um projeto da Spring Framework, criado para simplificar o processo de configuração e desenvolvimento de aplicações baseadas em Java. Lançado pela Pivotal em 2014, Spring Boot visa tornar o desenvolvimento de aplicações Spring mais rápido e fácil, eliminando a necessidade de configuração manual e complexa. Ele oferece uma abordagem simplificada para a criação de microservices e aplicações robustas, utilizando convenções sensatas para minimizar a configuração explícita.

Características Principais do Spring Boot

1. **Configuração Automática (Auto-Configuration)**

Spring Boot pode configurar automaticamente muitos aspectos de uma aplicação Spring com base nas dependências presentes no classpath e em outras propriedades específicas do projeto. Isso significa que muitas configurações padrão são aplicadas automaticamente, permitindo que os desenvolvedores iniciem rapidamente.

2. **Inicialização Rápida (Spring Initializr)**

Spring Boot fornece uma ferramenta chamada Spring Initializr, que permite criar rapidamente um novo projeto Spring Boot com todas as dependências necessárias. Isso elimina a complexidade de configurar manualmente um novo projeto e garante que as dependências corretas sejam incluídas desde o início.

3. **Servidor Embutido**

Spring Boot inclui servidores embutidos como Tomcat, Jetty, e Undertow, permitindo que as aplicações sejam executadas como executáveis independentes. Isso simplifica o processo de desenvolvimento e implantação, pois elimina a necessidade de configurar servidores de aplicação externamente.

4. **Arquitetura de Microservices**

Spring Boot facilita a criação de microservices, que são pequenos serviços independentes que podem ser implantados e escalados de forma independente. Isso é particularmente útil em arquiteturas modernas, onde a escalabilidade e a resiliência são cruciais.

5. **Monitoramento e Gestão (Spring Boot Actuator)**

Spring Boot Actuator oferece uma série de endpoints para monitoramento e gestão da aplicação. Isso inclui métricas, saúde, informações do sistema, e muito mais, ajudando os desenvolvedores a manter e monitorar a aplicação em produção.

- **Vantagens de Utilizar Spring Boot em um Sistema de Gestão de Finanças Pessoais**

Desenvolver um sistema de gestão de finanças pessoais requer uma plataforma que seja robusta, segura, escalável e fácil de manter. Spring Boot oferece uma série de vantagens que o tornam uma excelente escolha para esse tipo de aplicação.

1. Facilidade de Desenvolvimento

A. Configuração Simplificada

Spring Boot elimina a necessidade de configurações complexas, permitindo que os desenvolvedores foquem na lógica de negócios em vez de se preocuparem com a configuração do framework. A auto-configuração ajusta automaticamente muitos aspectos da aplicação com base nas dependências e propriedades, o que acelera significativamente o processo de desenvolvimento.

B. Inicialização Rápida

Com o Spring Initializr, os desenvolvedores podem rapidamente criar um projeto Spring Boot com todas as dependências necessárias para iniciar o desenvolvimento de imediato. Isso é especialmente útil para prototipagem rápida e iterações ágeis, permitindo que novas funcionalidades sejam adicionadas rapidamente.

2. Arquitetura Moderna

A. Suporte a Microservices

Spring Boot facilita a criação de microservices, que são ideais para aplicações financeiras que podem necessitar de alta escalabilidade e resiliência. Cada serviço pode ser desenvolvido, testado, e implantado de forma independente, permitindo uma evolução contínua e redução do tempo de inatividade.

B. Servidores Embutidos

O uso de servidores embutidos simplifica o processo de desenvolvimento e implantação, permitindo que a aplicação seja executada como um executável independente. Isso facilita o ciclo de desenvolvimento e teste, além de simplificar a implantação em ambientes de produção.

3. Robustez e Segurança

A. Segurança Integrada

Spring Boot integra-se perfeitamente com Spring Security, oferecendo um conjunto robusto de funcionalidades para autenticação e autorização. Isso é crucial para um sistema de gestão de finanças pessoais, onde a segurança dos dados é uma prioridade

máxima.

B. Gestão de Transações

Spring Boot oferece suporte robusto para transações, garantindo que todas as operações de banco de dados sejam executadas de forma consistente e segura. Isso é essencial para garantir a integridade dos dados financeiros.

4. Monitoramento e Gestão

A. Actuator

Spring Boot Actuator fornece endpoints para monitoramento e gestão da aplicação, oferecendo métricas detalhadas sobre a saúde, desempenho e uso do sistema. Isso permite que os administradores monitorem a aplicação em tempo real, detectem e solucionem problemas rapidamente.

B. Facilidade de Integração com Ferramentas de Monitoramento

A arquitetura extensível de Spring Boot permite uma fácil integração com ferramentas de monitoramento e gestão de terceiros, como Prometheus, Grafana, e ELK Stack. Isso oferece uma visão abrangente do desempenho e da saúde do sistema, facilitando a manutenção proativa.

5. Comunidade e Suporte

A. Comunidade Ativa

Spring Boot é suportado por uma comunidade vibrante e ativa, que continuamente contribui com melhorias, bibliotecas e documentação. Isso significa que os desenvolvedores têm acesso a uma vasta quantidade de recursos e suporte, facilitando a resolução de problemas e a implementação de novas funcionalidades.

B. Documentação Extensa

Spring Boot possui uma documentação abrangente e de alta qualidade, que cobre todos os aspectos do desenvolvimento e configuração da aplicação. Isso reduz a curva de aprendizado e permite que os desenvolvedores se tornem produtivos rapidamente.

Spring Boot é uma escolha excelente para desenvolver um sistema de gestão de finanças pessoais devido à sua facilidade de desenvolvimento, arquitetura moderna, robustez,

segurança e capacidades de monitoramento e gestão. Com sua configuração simplificada, suporte a microservices, e uma comunidade ativa, Spring Boot permite que os desenvolvedores criem aplicações robustas, escaláveis e seguras de forma eficiente e eficaz. Essas características tornam Spring Boot uma plataforma ideal para aplicações financeiras.

7.4.3 React

React é uma biblioteca JavaScript de código aberto, mantida pelo Facebook, que é utilizada para construir interfaces de usuário (UIs) interativas e dinâmicas. Lançada em 2013, React rapidamente ganhou popularidade devido à sua abordagem eficiente e flexível para criar componentes de UI reutilizáveis. A biblioteca é amplamente adotada por desenvolvedores de todo o mundo e é usada por grandes empresas como Facebook, Instagram, Airbnb e muitos outros.

Características Principais do React

1. **Componentização**

React é baseado em uma arquitetura de componentes. Cada componente é uma unidade independente que pode ser reutilizada em diferentes partes da aplicação. Isso promove a modularidade e a reutilização de código, facilitando a manutenção e a expansão do sistema.

2. **Virtual DOM**

React utiliza um Virtual DOM (Document Object Model) para melhorar a performance da aplicação. Quando o estado de um componente muda, o Virtual DOM calcula a diferença (diff) entre a versão anterior e a nova do DOM e aplica apenas as mudanças necessárias ao DOM real. Isso resulta em atualizações mais rápidas e eficientes.

3. **Unidirecional Data Flow**

Em React, o fluxo de dados é unidirecional, ou seja, os dados fluem de cima para baixo (dos componentes pais para os componentes filhos). Isso facilita o rastreamento de mudanças no estado e torna o código mais previsível e fácil de entender.

4. **JSX (JavaScript XML)**

JSX é uma extensão de sintaxe que permite escrever código HTML dentro do JavaScript. Isso torna o código mais legível e expressivo, facilitando a construção de componentes **de UI complexos.**

5. **Ferramentas de Desenvolvimento**

React possui um rico ecossistema de ferramentas e bibliotecas complementares, como Redux para gerenciamento de estado, React Router para navegação e Next.js para renderização do lado do servidor (SSR).

- **Vantagens de Utilizar React em um Sistema de Gestão de Finanças Pessoais**

Desenvolver um sistema de gestão de finanças pessoais requer uma interface de usuário que seja intuitiva, responsiva e eficiente. React oferece várias vantagens que o tornam uma excelente escolha para este tipo de aplicação.

1. Componentização e Reutilização de Código

A. Modularidade

React permite a criação de componentes modulares que podem ser facilmente reutilizados em diferentes partes da aplicação. Isso reduz a duplicação de código e facilita a manutenção. Por exemplo, um componente de gráfico financeiro pode ser usado em várias seções da aplicação, como visualizações de despesas, receitas e investimentos.

B. Manutenção Simplificada

Com a arquitetura baseada em componentes, é mais fácil isolar e corrigir bugs ou adicionar novas funcionalidades sem afetar outras partes do sistema. Cada componente pode ser desenvolvido e testado de forma independente, o que melhora a eficiência do desenvolvimento.

2. Performance Otimizada

A. Atualizações Eficientes com o Virtual DOM

O uso do Virtual DOM em React permite que as atualizações da interface do usuário sejam realizadas de forma rápida e eficiente. Em um sistema de gestão de finanças pessoais, onde os dados podem mudar frequentemente (por exemplo, ao adicionar transações ou atualizar saldos), o Virtual DOM garante que apenas as partes necessárias

da interface sejam atualizadas, melhorando a performance geral.

B. Renderização Seletiva

React atualiza apenas os componentes que realmente mudaram, minimizando o tempo e os recursos necessários para atualizar a interface. Isso é particularmente importante em aplicações financeiras, onde a atualização de gráficos e tabelas de dados pode ser intensiva.

3. Fluxo de Dados Previsível

A. Unidirecional Data Flow

O fluxo de dados unidirecional de React torna a aplicação mais previsível e fácil de depurar. Em um sistema de gestão de finanças pessoais, onde a consistência dos dados é crucial, essa característica ajuda a garantir que as mudanças no estado da aplicação sejam controladas e rastreáveis.

B. Fácil Integração com Ferramentas de Gerenciamento de Estado

Ferramentas como Redux ou Context API podem ser usadas com React para gerenciar o estado global da aplicação de forma eficiente. Isso é essencial para manter a consistência dos dados em um sistema financeiro, onde múltiplos componentes podem depender dos mesmos dados.

4. Experiência do Usuário e Interatividade

A. UIs Dinâmicas e Responsivas

React facilita a criação de interfaces de usuário dinâmicas e interativas. Por exemplo, gráficos de despesas podem ser atualizados em tempo real à medida que o usuário adiciona novas transações, proporcionando uma experiência mais rica e envolvente.

B. Componentes de UI Prontos para Uso

Há uma ampla gama de bibliotecas de componentes de UI disponíveis para React, como Material-UI, que fornecem componentes prontos para uso e bem projetados. Isso acelera o desenvolvimento e garante uma interface de usuário profissional e intuitiva.

5. Ferramentas de Desenvolvimento e Ecossistema

A. Ferramentas de Debugging

React Developer Tools é uma extensão do navegador que permite inspecionar a hierarquia de componentes e o estado da aplicação em tempo real. Isso facilita a depuração e o desenvolvimento iterativo.

B. Grande Comunidade e Suporte

React possui uma grande e ativa comunidade de desenvolvedores, o que significa que há uma abundância de recursos, tutoriais e suporte disponíveis. Isso facilita a resolução de problemas e a implementação de novas funcionalidades.

React é uma excelente escolha para desenvolver um sistema de gestão de finanças pessoais devido à sua modularidade, performance otimizada, fluxo de dados previsível, e capacidades de criar interfaces de usuário dinâmicas e responsivas. Com um rico ecossistema de ferramentas e uma grande comunidade de suporte, React permite que os desenvolvedores criem aplicações robustas, eficientes e de alta qualidade. Essas características tornam React uma plataforma ideal para aplicações financeiras, onde a usabilidade e a performance são cruciais para uma experiência de usuário satisfatória e eficiente.

7.4.4 MySQL

MySQL é um sistema de gerenciamento de banco de dados relacional (RDBMS) de código aberto, desenvolvido originalmente pela MySQL AB, uma empresa sueca, e posteriormente adquirida pela Sun Microsystems, que mais tarde foi adquirida pela Oracle Corporation. Desde seu lançamento em 1995, MySQL se tornou uma das soluções de banco de dados mais populares no mundo, amplamente utilizado por desenvolvedores e empresas para armazenar e gerenciar dados de forma eficiente e segura.

Características Principais do MySQL

1. **Modelo Relacional**

MySQL utiliza um modelo relacional para organizar os dados em tabelas, linhas e

colunas. Isso facilita a definição de relacionamentos entre diferentes conjuntos de dados.

2. **SQL (Structured Query Language)**

MySQL usa SQL como sua linguagem de consulta padrão. SQL é uma linguagem declarativa que permite realizar operações complexas de manipulação e recuperação de dados de maneira intuitiva e eficiente.

3. **Suporte a Múltiplos Sistemas Operacionais**

MySQL é compatível com uma ampla gama de sistemas operacionais, incluindo Windows, Linux, e macOS. Isso oferece flexibilidade para ser utilizado em diversos ambientes de desenvolvimento e produção.

4. **Desempenho e Escalabilidade**

MySQL é conhecido por seu alto desempenho e capacidade de escalabilidade. Ele pode ser otimizado para lidar com grandes volumes de dados e alto tráfego de transações, o que o torna adequado tanto para pequenas aplicações quanto para grandes sistemas empresariais.

5. **Segurança e Controle de Acesso**

MySQL oferece robustas funcionalidades de segurança, incluindo autenticação de usuários, controle de acesso baseado em permissões, e suporte a criptografia. Isso garante que os dados sejam protegidos contra acesso não autorizado e ataques maliciosos.

- **Vantagens de Utilizar MySQL em um Sistema de Gestão de Finanças Pessoais**

Desenvolver um sistema de gestão de finanças pessoais requer um banco de dados que seja robusto, seguro, eficiente e capaz de lidar com grandes volumes de transações. MySQL oferece várias vantagens que o tornam uma escolha ideal para este tipo de aplicação.

1. Eficiência no Gerenciamento de Dados

A. Estrutura Relacional

MySQL organiza os dados em tabelas relacionais, o que facilita a definição de

relacionamentos entre diferentes tipos de dados. Isso é crucial para um sistema de gestão de finanças pessoais, onde é necessário vincular transações a contas, categorias, e usuários.

B. Consultas SQL Poderosas

O suporte a SQL permite realizar consultas complexas de maneira eficiente. Por exemplo, é possível gerar relatórios financeiros detalhados, como balanços mensais e anuais, categorização de despesas e receitas, e análises de tendências de gastos, com poucas linhas de código SQL.

2. Performance e Escalabilidade

A. Alta Performance

MySQL é otimizado para alta performance em operações de leitura e escrita. Em um sistema de gestão de finanças pessoais, onde os dados são constantemente atualizados e consultados, essa característica garante uma experiência de usuário rápida e responsiva.

B. Escalabilidade Horizontal e Vertical

MySQL suporta escalabilidade tanto horizontal (sharding) quanto vertical (aumento de recursos de hardware). Isso permite que o sistema cresça conforme aumenta o volume de dados e o número de usuários, sem comprometer a performance.

3. Segurança de Dados

A. Controle de Acesso Granular

MySQL oferece um robusto sistema de controle de acesso baseado em permissões, permitindo definir quem pode acessar e manipular os dados. Isso é essencial para proteger informações financeiras sensíveis e garantir a privacidade dos usuários.

B. Suporte a Criptografia

MySQL suporta criptografia de dados em repouso e em trânsito. Isso protege os dados financeiros contra acessos não autorizados e ataques durante a transmissão, garantindo a integridade e a confidencialidade das informações.

4. Backup e Recuperação

A. Ferramentas de Backup

MySQL oferece várias ferramentas para backup e recuperação de dados, como mysqldump e MySQL Enterprise Backup. Isso permite realizar backups regulares e automáticos dos dados financeiros, minimizando o risco de perda de dados.

B. Recuperação de Desastres

Em caso de falhas ou desastres, MySQL facilita a recuperação rápida dos dados. Ferramentas de replicação e logs de transações ajudam a restaurar o banco de dados ao seu estado anterior, garantindo continuidade e resiliência da aplicação.

5. Suporte e Comunidade

A. Grande Comunidade de Usuários

MySQL possui uma das maiores comunidades de usuários de banco de dados, proporcionando acesso a uma vasta quantidade de recursos, tutoriais, e fóruns de suporte. Isso facilita a resolução de problemas e a implementação de novas funcionalidades.

B. Suporte Comercial

Além do suporte da comunidade, MySQL oferece opções de suporte comercial através da Oracle, garantindo que empresas tenham acesso a suporte técnico especializado e atualizações regulares.

7.4.5 Docker

Docker é uma plataforma de código aberto que automatiza a implantação de aplicações dentro de contêineres de software. Um contêiner é uma unidade de software que empacota o código da aplicação e todas as suas dependências, como bibliotecas e configurações, para garantir que a aplicação seja executada de forma consistente em qualquer ambiente. Desde seu lançamento em 2013, Docker revolucionou a forma como desenvolvedores e operações trabalham, promovendo práticas de DevOps e simplificando o desenvolvimento, teste e implantação de software.

Características Principais do Docker

- **Isolamento de Contêineres**

Cada container Docker é isolado do sistema host e de outros contêineres, garantindo que

as aplicações rodem em ambientes segregados. Isso permite que múltiplas aplicações e suas dependências coexistem no mesmo host sem conflitos.

- **Portabilidade**

Containers Docker podem ser executados em qualquer máquina que tenha o Docker instalado, independentemente do sistema operacional subjacente. Isso facilita a migração de aplicações entre diferentes ambientes, como desenvolvimento, teste e produção.

- **Leveza e Eficiência**

Contêineres são mais leves do que máquinas virtuais (VMs) tradicionais porque compartilham o mesmo kernel do sistema operacional host, reduzindo a sobrecarga de recursos. Isso permite executar mais contêineres em um único host em comparação com VMs.

- **Facilidade de Distribuição**

Docker Hub é um repositório de contêineres que permite compartilhar e distribuir imagens de contêineres. Desenvolvedores podem criar, enviar e baixar imagens de contêineres facilmente, promovendo a reutilização e colaboração.

- **Gerenciamento de Dependências**

Dockerfile é um script que define as etapas necessárias para construir uma imagem de contêiner. Ele especifica todas as dependências e configurações da aplicação, garantindo que o ambiente de execução seja consistente em todas as instâncias.

Vantagens de Utilizar Docker em um Sistema de Gestão de Finanças Pessoais

Desenvolver e manter um sistema de gestão de finanças pessoais requer um ambiente que seja robusto, seguro, escalável e fácil de gerenciar. Docker oferece várias vantagens que o tornam uma escolha ideal para este tipo de aplicação.

1. **Consistência e Reprodutibilidade**
    a. **Ambiente de Execução Consistente**

Com Docker, o ambiente de execução da aplicação é encapsulado dentro de um contêiner, garantindo que a aplicação funcione da mesma forma em qualquer ambiente, seja ele de desenvolvimento, teste ou produção. Isso elimina o problema de "funciona na minha máquina", reduzindo o tempo gasto em solucionar problemas de ambiente.

    b. **Construção e Desdobramento Automatizados**

Utilizando Dockerfiles, o processo de construção e desdobramento de contêineres pode ser automatizado, garantindo que cada vez que uma nova imagem de contêiner é criada, ela é idêntica às anteriores. Isso promove a reprodutibilidade e facilita o versionamento e

rollback de aplicações.

2. **Escalabilidade e Performance**
    a. **Gerenciamento Eficiente de Recursos**

Docker permite executar múltiplos contêineres em um único host, utilizando recursos de forma mais eficiente do que máquinas virtuais tradicionais. Isso é crucial para um sistema de gestão de finanças pessoais que precisa escalar horizontalmente para atender a um número crescente de usuários e transações.

    b. **Fácil Escalonamento Horizontal**

Contêineres podem ser facilmente replicados e distribuídos entre vários servidores, permitindo que a aplicação escale horizontalmente conforme a demanda. Ferramentas de orquestração como Kubernetes podem ser usadas para gerenciar a escalabilidade e distribuição de contêineres de forma automatizada.

3. **Segurança**
    a. **Isolamento de Contêineres**

Cada contêiner opera em um ambiente isolado, protegendo a aplicação e seus dados de interferências externas. Isso é vital para um sistema de gestão de finanças pessoais, onde a segurança dos dados financeiros dos usuários é uma prioridade máxima.

    b. **Atualizações e Patches Facilitados**

Com Docker, é possível atualizar ou aplicar patches a componentes individuais da aplicação sem afetar o restante do sistema. Isso garante que as vulnerabilidades de segurança possam ser corrigidas rapidamente, minimizando a janela de exposição.

4. **Desenvolvimento e Integração Contínua**
    a. **Integração com CI/CD**

Docker se integra facilmente com ferramentas de integração contínua e entrega contínua (CI/CD) como Jenkins, GitLab CI, e CircleCI. Isso permite automatizar todo o ciclo de vida do desenvolvimento de software, desde a construção até o teste e implantação, garantindo que novas funcionalidades e correções sejam entregues de forma rápida e confiável.

    b. **Ambiente de Desenvolvimento Consistente**

Desenvolvedores podem configurar e compartilhar ambientes de desenvolvimento

consistentes usando Docker, garantindo que todos os membros da equipe estejam trabalhando com as mesmas versões de software e dependências. Isso reduz o tempo gasto na configuração do ambiente e facilita a colaboração.

5. **Portabilidade e Flexibilidade**
    a. **Migração Fácil entre Ambientes**

Com Docker, containers podem ser facilmente movidos entre diferentes ambientes, como desenvolvimento, teste, e produção, sem a necessidade de reconfiguração. Isso facilita a migração de aplicações entre diferentes provedores de nuvem ou centros de dados.

    b. **Suporte a Microsserviços**

Docker é ideal para arquiteturas de microservices, onde cada serviço é empacotado em um contêiner separado. Isso permite que os serviços sejam desenvolvidos, testados e implantados de forma independente, promovendo a modularidade e a escalabilidade do sistema.

7.4.6 Prometheus

Prometheus é uma ferramenta de monitoramento e alerta de código aberto projetada originalmente pela SoundCloud em 2012. Desde então, evoluiu para se tornar um dos sistemas de monitoramento mais populares e amplamente utilizados, especialmente em arquiteturas baseadas em contêineres e microsserviços. O Prometheus se destaca por seu modelo de dados multidimensional, eficiente coleta de dados através de "pull", e sua linguagem de consulta poderosa, PromQL.

Vantagens do Prometheus

1. Modelo de Dados Multidimensional

O modelo de dados do Prometheus é baseado em séries temporais identificadas por um conjunto de rótulos chave-valor. Isso permite a coleta e agregação de métricas de maneira flexível e detalhada, possibilitando a visualização de dados granulares e a criação de alertas precisos.

2. Independência de Alvo

Prometheus coleta dados de endpoints HTTP, o que significa que cada serviço pode

expor suas próprias métricas através de um simples endpoint HTTP. Isso elimina a necessidade de um agente de monitoramento centralizado, permitindo que cada componente do sistema seja responsável por suas próprias métricas.

3. Linguagem de Consulta Poderosa (PromQL)

PromQL é uma linguagem de consulta específica do Prometheus, que permite aos usuários escrever consultas complexas e expressivas para extrair e analisar métricas. Com o PromQL, é possível criar gráficos detalhados e alertas personalizados com base em dados históricos e em tempo real.

4. Alertas Avançados

Prometheus inclui um sistema de alertas que permite a definição de condições baseadas em consultas PromQL. Quando essas condições são atendidas, alertas são gerados e podem ser enviados para diversos sistemas de notificação, como e-mail, Slack, ou PagerDuty, através do Alertmanager.

5. Integração com Grafana

Prometheus se integra facilmente com Grafana, uma popular ferramenta de visualização de dados, permitindo a criação de dashboards interativos e personalizados. Essa integração oferece uma visualização rica das métricas coletadas e facilita o monitoramento contínuo do sistema.

Vantagens do Prometheus na Computação em Nuvem

1. Escalabilidade Dinâmica

Uma das maiores vantagens do Prometheus na computação em nuvem é sua capacidade de escalar dinamicamente. Em ambientes de nuvem, onde os recursos podem aumentar ou diminuir rapidamente, Prometheus pode ajustar-se automaticamente ao dimensionamento dos recursos monitorados, sem a necessidade de reconfigurações manuais extensivas.

2. Monitoramento de Microsserviços

A arquitetura de microsserviços, comum na computação em nuvem, se beneficia enormemente do Prometheus. Cada micro serviço pode expor suas métricas, e o

Prometheus pode coletar essas métricas individualmente, proporcionando uma visão detalhada do desempenho e da saúde de cada componente.

3. Integração com Orquestradores de Contêineres

Prometheus se integra nativamente com orquestradores de contêineres como Kubernetes. Ele pode descobrir automaticamente novos contêineres e serviços à medida que são criados, ajustando-se dinamicamente ao ambiente de execução. Isso é particularmente útil em nuvens públicas e privadas, onde os serviços podem ser criados e destruídos rapidamente.

4. Economia de Custos

Com a nuvem, os recursos são cobrados com base no uso. Prometheus, sendo eficiente e leve, pode coletar e armazenar grandes volumes de dados sem consumir recursos excessivos. Isso resulta em uma solução de monitoramento econômica, especialmente em comparação com soluções proprietárias que podem ter custos de licenciamento e operação mais elevados.

5. Flexibilidade e Adaptabilidade

A computação em nuvem oferece uma ampla gama de serviços e plataformas. Prometheus pode ser facilmente adaptado para monitorar esses diferentes serviços, sejam eles máquinas virtuais, funções sem servidor (serverless), ou serviços gerenciados. Sua capacidade de ser configurado e estendido permite que ele acompanhe a evolução e a inovação contínua nos ambientes de nuvem.

Prometheus é uma ferramenta de monitoramento robusta e flexível, com vantagens significativas em termos de escalabilidade, capacidade de integração e detalhamento das métricas. Quando utilizado em conjunto com a computação em nuvem, Prometheus oferece uma solução poderosa para o monitoramento e alerta de sistemas complexos e dinâmicos. Sua capacidade de se integrar com diversas tecnologias e plataformas, junto com sua eficiência e flexibilidade, o tornam uma escolha excelente para operações de TI modernas que necessitam de visibilidade contínua e aprofundada dos sistemas.

8. Microsserviços utilizados

Nos recursos de microsserviços temos o empregos dos seguintes serviços, VLibras, uma biblioteca que facilita a comunicação entre os microserviços, garantindo uma interação

fluida e eficiente.

Docker: Uma plataforma que permite containerizar seus microserviços, facilitando a implantação e o gerenciamento em diferentes ambientes.

Spring Boot: Um framework Java que simplifica o desenvolvimento de microserviços, sua proposta consiste em desenvolver serviços que são facilmente mantidos por equipe pequenas, fornecendo recursos prontos para uso e diminuindo o tempo de desenvolvimento.

8.1 Libras

O emprego do microsserviço de Libras através da API VLibras em um website é fundamental para promover a acessibilidade e inclusão digital de pessoas surdas ou com deficiência auditiva. A Libras, ou Língua Brasileira de Sinais, é a língua oficial das pessoas surdas no Brasil e sua utilização permite uma comunicação eficaz e igualitária para esse público.

A importância desse microsserviço reside na quebra de barreiras de comunicação, garantindo que todos os usuários tenham acesso ao conteúdo do website, independentemente de sua capacidade auditiva. Ao disponibilizar a tradução do conteúdo para Libras, o website se torna mais acessível e inclusivo, permitindo que pessoas surdas possam compreender e interagir plenamente com o que está sendo oferecido.

A implementação do microsserviço de Libras pode ser realizada de diversas formas, mas geralmente envolve a integração de um player de vídeo que exibe um intérprete de Libras traduzindo o conteúdo em tempo real. Esse player pode ser ativado pelo usuário conforme sua necessidade, proporcionando uma experiência personalizada e intuitiva.

Além disso, o microsserviço de Libras também pode ser integrado a outras funcionalidades do website, como chats ao vivo ou atendimento ao cliente, garantindo que pessoas surdas tenham suporte adequado em suas interações online.

Em resumo, o emprego do microsserviço de Libras em um website não só é importante para garantir a acessibilidade e inclusão digital, mas também é uma demonstração do compromisso em promover a igualdade de oportunidades e o respeito à diversidade.

8.2 Docker

O emprego do microsserviço com Docker em um website desempenha um papel crucial na eficiência operacional, na escalabilidade e na consistência do ambiente de desenvolvimento e produção. O Docker é uma plataforma de virtualização de contêineres

que permite empacotar, distribuir e executar aplicativos de forma independente em ambientes isolados, chamados contêineres. Essa abordagem modular oferece uma série de benefícios significativos para o desenvolvimento e implantação de websites.

A importância do Docker reside na simplificação do processo de desenvolvimento, teste e implantação de software. Ao encapsular cada serviço do website em containers Docker, é possível garantir que todas as dependências e configurações necessárias estejam presentes e consistentes em todos os ambientes, desde o desenvolvimento local até a produção em escala. Isso elimina muitos dos problemas de compatibilidade e configuração que costumam surgir em ambientes distribuídos e heterogêneos.

Além disso, o Docker facilita a escalabilidade horizontal, permitindo que o website seja dimensionado de forma dinâmica e eficiente conforme a demanda do tráfego. Novos contêineres podem ser facilmente criados e distribuídos em um cluster de servidores para lidar com picos de carga, garantindo assim um desempenho consistente e uma experiência de usuário satisfatória.

A utilização do Docker também simplifica o gerenciamento de versões e atualizações do software, tornando o processo mais ágil e seguro. Os containers Docker são imutáveis por natureza, o que significa que as alterações no código ou nas configurações resultam na criação de uma nova imagem de contêiner, garantindo assim a consistência e a rastreabilidade das alterações ao longo do tempo.

A implementação do Docker em um website envolve a criação de um Dockerfile para cada serviço ou componente do sistema, especificando as dependências e as instruções necessárias para construir a imagem do contêiner. Essas imagens podem ser distribuídas e compartilhadas por meio de um registro de contêineres, como o Docker Hub, facilitando a colaboração e a integração contínua entre os membros da equipe de desenvolvimento.

Em resumo, o emprego do microsserviço Docker em um website oferece uma abordagem moderna e eficiente para o desenvolvimento, implantação e operação de aplicativos web, proporcionando benefícios tangíveis em termos de escalabilidade, consistência e agilidade no ciclo de vida do software.

8.3 UserWay

O UserWay é um microserviço de acessibilidade que visa tornar os sites mais acessíveis para pessoas com deficiência. A importância do UserWay reside na sua capacidade de

ajudar as organizações a cumprir normas de acessibilidade, como as Diretrizes de Acessibilidade para Conteúdo Web (WCAG), a Lei dos Americanos com Deficiências (ADA), entre outras regulamentações globais. A inclusão digital é um aspecto crítico na sociedade moderna, e o UserWay oferece uma solução eficiente para garantir que todos, independentemente de suas capacidades, possam acessar e navegar em websites sem dificuldades.

O UserWay ajuda as empresas a se manterem em conformidade com várias regulamentações de acessibilidade, evitando possíveis processos legais e multas.
Aumento de Alcance: Ao tornar um site acessível, as empresas podem alcançar um público maior, incluindo milhões de pessoas com deficiência.
Um site acessível melhora a experiência de todos os usuários, não apenas daqueles com deficiências. Demonstra um compromisso com a inclusão e a responsabilidade social pode melhorar a imagem e a reputação da marca.

O UserWay é integrado a websites para proporcionar várias funcionalidades de acessibilidade. Ele oferece uma barra de ferramentas de acessibilidade que permite aos usuários ajustar o conteúdo do site conforme suas necessidades.

Texto Alternativo para Imagens: Facilita a compreensão do conteúdo visual para usuários de leitores de tela. Contraste de Cores Ajustável: Permite aos usuários ajustar o contraste das cores para melhorar a legibilidade. Ampliação de Texto: Usuários podem aumentar o tamanho do texto sem perder a estrutura do site. Navegação por Teclado: Permite a navegação pelo site usando apenas o teclado, beneficiando usuários com dificuldades motoras. Leitor de Texto: Oferece a leitura do conteúdo em voz alta, útil para usuários com deficiências visuais.

A implementação do UserWay é bastante direta e pode ser feita adicionando um snippet de código JavaScript ao seu site. O UserWay permite personalizar as funcionalidades e a aparência da barra de ferramentas de acessibilidade. Você pode acessar as configurações no painel da sua conta no UserWay e ajustar conforme necessário.

Após adicionar o código ao seu site, teste a barra de ferramentas de acessibilidade para garantir que todas as funcionalidades estão funcionando corretamente e que a barra está aparecendo conforme esperado.

Em lojas online, a barra de ferramentas UserWay permite que os clientes ajustem o site para suas necessidades, aumentando a satisfação do usuário e potencialmente aumentando as vendas.

O UserWay é uma ferramenta essencial para qualquer organização que deseja melhorar a acessibilidade do seu site. Sua implementação simples e as vastas funcionalidades de personalização o tornam uma escolha ideal para tornar o conteúdo web acessível a todos. Além disso, ao adotar o UserWay, as empresas demonstram um compromisso com a inclusão e a responsabilidade social, beneficiando tanto os usuários quanto a própria organização.

8.4 Spring Boot

O uso do microsserviço Spring Boot se destaca como uma escolha estratégica para impulsionar o desenvolvimento de websites. Este framework Java oferece uma gama de vantagens que vão desde a simplificação do desenvolvimento até a melhoria da escalabilidade e da manutenção.

O Spring Boot simplifica drasticamente o desenvolvimento web, proporcionando um ambiente de configuração mínimo e prontamente disponível para criar aplicativos Java de maneira rápida e eficiente. Sua abordagem "convention over configuration" reduz a quantidade de código necessário, permitindo que os desenvolvedores se concentrem na lógica de negócios e na criação de valor para os usuários finais. Além disso, o Spring Boot integra-se perfeitamente com outras tecnologias e frameworks, facilitando a implementação de funcionalidades.

O Spring Boot é amplamente utilizado em websites para diversos propósitos, incluindo a criação de APIs RESTful, o processamento de solicitações HTTP, o gerenciamento de dependências e a configuração de segurança. Ele oferece uma variedade de recursos prontos para uso, como Spring Data JPA para acesso a banco de dados, Spring Security para autenticação e autorização, e Spring MVC para desenvolvimento de aplicativos web. Além disso, sua arquitetura modular permite que os desenvolvedores escolham e integrem os componentes necessários para atender aos requisitos específicos do projeto.

A implementação do Spring Boot em um website geralmente envolve a criação de um projeto Spring Boot com o uso de ferramentas como Maven ou Gradle para gerenciar as

dependências. Os desenvolvedores podem então criar controladores para lidar com solicitações HTTP, serviços para realizar a lógica de negócios e repositórios para interagir com o banco de dados. Além disso, o Spring Boot oferece suporte para a configuração de propriedades externas, o que facilita a implantação em diferentes ambientes, como desenvolvimento, teste e produção.

O Spring Boot desempenha um papel fundamental no desenvolvimento eficiente e escalável de websites, fornecendo uma estrutura robusta e flexível para criar aplicativos Java modernos. Sua simplicidade, juntamente com sua extensa gama de recursos e integrações, torna-o uma escolha ideal para empresas e desenvolvedores que buscam maximizar a produtividade e oferecer experiências web de alta qualidade aos usuários.

9. Metodologia

Desenvolver um sistema de controle de finanças pessoais utilizando Java, Spring Boot 3, React JS e MySQL como banco de dados requer uma abordagem estruturada para garantir eficiência, qualidade e alinhamento com as necessidades do usuário final. Para este projeto, a metodologia ágil será adotada, especificamente o Scrum, devido à sua flexibilidade e foco na entrega incremental de valor. Aqui está uma visão detalhada de como essa metodologia será aplicada:

9.1 Planejamento e Levantamento de Requisitos

A. Identificação das Partes Interessadas

Identificar todos os stakeholders do projeto, incluindo usuários finais, gerentes de projeto, desenvolvedores, e outros interessados.

B. Reuniões de Requisitos

Realizar reuniões com os stakeholders para entender as necessidades e expectativas. Documentar os requisitos funcionais e não funcionais do sistema.

C. Priorização dos Requisitos

Classificar os requisitos com base na importância e urgência. Utilizar técnicas como MoSCoW (Must have, Should have, Could have, Won't have) para ajudar na priorização.

D. Criação do Product Backlog

Compilar todos os requisitos em um backlog de produto, que servirá como a lista de funcionalidades e melhorias a serem implementadas.

9.2 Planejamento de Sprints

A. Definição do Sprint Backlog

Para cada sprint, selecionar um conjunto de itens do backlog de produto que serão implementados. Esses itens compõem o sprint backlog.

B. Estimativas de Tarefas

Utilizar técnicas como Planning Poker para estimar o esforço necessário para cada item do backlog. As estimativas ajudam a planejar o trabalho dentro do tempo disponível para o sprint.

9.3 Desenvolvimento Incremental

A. Sprints Curto e Iterativos

Cada sprint terá uma duração fixa, geralmente de 2 a 4 semanas. O objetivo é entregar uma versão funcional do software ao final de cada sprint.

B. Reuniões Diárias (Daily Stand-ups)

Realizar reuniões diárias rápidas para discutir o progresso, identificar impedimentos e ajustar o plano conforme necessário.

C. Desenvolvimento Orientado a Testes (TDD)

Adotar práticas de TDD para garantir que cada funcionalidade seja desenvolvida com testes automatizados. Isso melhora a qualidade do código e facilita a manutenção.

9.4 Integração Contínua e Entrega Contínua (CI/CD)

A. Ferramentas de CI/CD

Utilizar ferramentas como Jenkins, GitLab CI ou CircleCI para automatizar o processo de construção, teste e implantação.

B. Integração Contínua

Implementar a integração contínua para garantir que o código integrado ao repositório seja automaticamente testado e construído. Isso ajuda a detectar problemas rapidamente.

C. Entrega Contínua

Configurar entrega contínua para implantar automaticamente alterações aprovadas no código em ambiente de staging ou produção.

9.5 Design e Implementação

A. Arquitetura de Microservices

Utilizar uma arquitetura de microservices com Spring Boot 3, onde cada funcionalidade principal do sistema (como gestão de transações, relatórios financeiros, etc.) é implementada como um serviço independente.

B. Frontend com React JS

Desenvolver a interface do usuário utilizando React JS, garantindo uma experiência de usuário dinâmica e responsiva.

C. Integração com MySQL

Configurar o banco de dados MySQL para armazenar dados financeiros de forma segura e eficiente. Utilizar JPA (Java Persistence API) e Hibernate para facilitar o mapeamento objeto-relacional.

D. Segurança e Autenticação

Implementar autenticação e autorização utilizando Spring Security, garantindo que apenas usuários autorizados possam acessar ou modificar dados.

9.6 Testes e Qualidade

A. Testes de Integração

Realizar testes de integração para garantir que os diferentes componentes do sistema funcionem bem juntos.

B. Revisões de Código

Adotar práticas de revisão de código para garantir a qualidade e consistência do código. Utilizar ferramentas como GitHub ou GitLab para gerenciar pull requests e revisões.

9.7 Documentação e Treinamento

A. Documentação do Usuário

Criar documentação de usuário e manuais de treinamento para ajudar os usuários finais a utilizar o sistema de forma eficaz.

9.8 Implantação e Manutenção

A. Deploy em Ambientes de Produção

Utilizar ferramentas de orquestração de contêineres como Kubernetes para gerenciar a implantação e escalabilidade dos serviços em produção.

B. Monitoramento e Logging

Implementar soluções de monitoramento e logging para acompanhar o desempenho e detectar problemas em tempo real.

C. Feedback Contínuo

Recolher feedback contínuo dos usuários finais e stakeholders para orientar melhorias futuras. Utilizar esse feedback para ajustar e priorizar o backlog de produto.

Para este projeto, seguindo a tendência de metodologia ágil do mercado, utilizaremos a metodologia scrum. Para esta fase do projeto, utilizaremos o seguinte product backlog, podendo sofrer alterações no futuro:

1. Registro de Transações
2. Visão Geral das Finanças
3. Controle do Saldo por Percentual
4. Conversão de Moeda
5. Cálculo de Impostos
6. Relatórios Detalhados

7. Lembretes e Notificações Personalizadas

8. Projeções Financeiras a Curto Prazo

9. Segurança e Autenticação

10. Acessibilidade de Tela (Requisito Não Funcional)

11. Contraste de Cores (Requisito Não Funcional)

12. Legibilidade de Fontes (Requisito Não Funcional)

13. Navegação Intuitiva (Requisito Não Funcional)

14. Compatibilidade com Tamanho de Tela Variado (Requisito Não Funcional)

15. Tempo de Resposta Rápido (Requisito Não Funcional)

16. Requisitos de Armazenamento Mínimo (Requisito Não Funcional)

17. Comportamento em Modo Offline (Requisito Não Funcional)

18. Atualizações sem Interrupções (Requisito Não Funcional)

19. Segurança de Dados (Requisito Não Funcional)

20. Feedback Visual e Sonoro (Requisito Não Funcional)

21. Compatibilidade com Recursos de Acessibilidade (Requisito Não Funcional)

22. Tempo de Carregamento Rápido (Requisito Não Funcional)

23. Compatibilidade com Dispositivos de Entrada Diversificados (Requisito Não Funcional)

9.9 Divisão de Sprints

Sprint 1 (2 semanas)

RF1 - Registro de Transações

RF2 - Visão Geral das Finanças

RF3 - Controle do Saldo por Percentual

Sprint 2 (2 semanas)

RF4 - Categorização Baseada na Pirâmide de Maslow

RF5 - Conversão de Moeda

RNF1 - Autenticação Segura (Requisito Não Funcional)

Sprint 3 (2 semanas)

RF6 - Cálculo de Impostos

RF7 - Relatórios Detalhados

RNF2 - Política de Privacidade e Proteção de Dados (Requisito Não Funcional)

Sprint 4 (2 semanas)

RF8 - Lembretes e Notificações Personalizadas

RF9 - Projeções Financeiras a Curto Prazo

RNF3 - Manutenção de Backup de Dados (Requisito Não Funcional) Sprint 5 (2 semanas)

RF10 - Segurança e Autenticação

RNF4 - Respeito aos Termos de Serviço (Requisito Não Funcional)

RNF5 - Suporte ao Cliente Eficiente (Requisito Não Funcional)

Sprint 6 (2 semanas)

RNF6 - Atualizações Regulares e Melhorias (Requisito Não Funcional)

RNF7 - Política de Reembolsos e Cancelamentos (Requisito Não Funcional)

RNF8 - Comunicação Transparente (Requisito Não Funcional)

Sprint 7 (2 semanas)

RNF9 - Compatibilidade com Legislação Local (Requisito Não Funcional)

RNF10 - Garantia de Disponibilidade do Serviço (Requisito Não Funcional)

RNF11 - Segurança de Dados (Requisito Não Funcional)

Sprint 8 (2 semanas)

RNF12 - Feedback Visual e Sonoro (Requisito Não Funcional)

RNF13 - Compatibilidade com Recursos de Acessibilidade (Requisito Não Funcional)

RNF14 - Tempo de Carregamento Rápido (Requisito Não Funcional)

RNF15 - Compatibilidade com Dispositivos de Entrada Diversificados (Requisito Não Funcional)

10. Propostas Futuras

O sistema Fit Finance, estará em constante evolução para atender melhor às necessidades de seus usuários. Entre os projetos futuros, haverá a criação de um hub de cursos de finanças pessoais, que oferecerá uma variedade de módulos educacionais para ajudar os usuários a melhorar suas habilidades financeiras. Este hub contará com cursos sobre planejamento financeiro, investimento, economia doméstica e outras áreas essenciais para o fortalecimento da saúde financeira individual.

Além disso, está previsto o desenvolvimento de funcionalidades avançadas, como ferramentas de análise preditiva e relatórios personalizados, que permitirão aos usuários tomar decisões financeiras mais informadas.

O Fit Finance também pretende expandir suas capacidades de inteligência artificial para oferecer assistentes financeiros virtuais, que fornecerão recomendações personalizadas baseadas nos hábitos e objetivos financeiros dos usuários. Com esses avanços, o Fit Finanças reafirma seu compromisso com a educação financeira e o empoderamento dos seus usuários, visando sempre proporcionar uma experiência cada vez mais completa e eficiente na gestão de suas finanças pessoais.

11. Conclusão

As finanças das famílias brasileiras têm sido objeto de intensa reflexão e preocupação, à medida que desafios econômicos e sociais impõem uma série de obstáculos à administração eficaz dos recursos. Diante de um cenário marcado por inflação, desemprego e instabilidade econômica, muitas famílias enfrentam dificuldades significativas para equilibrar seus orçamentos e manter um padrão de vida sustentável.

Uma das principais questões que afetam as finanças familiares no Brasil é a persistência das desigualdades sociais. Enquanto alguns lares desfrutam de estabilidade financeira e acesso a oportunidades, uma parcela considerável da população lida com a escassez de recursos e a falta de acesso a serviços essenciais. Essa disparidade torna-se evidente na disparidade educacional, no acesso limitado ao mercado de trabalho e na falta de acesso a crédito, elementos fundamentais para a construção de uma base financeira sólida.

O endividamento é outra pedra angular dos desafios financeiros enfrentados pelas famílias brasileiras. Taxas de juros elevadas, somadas à falta de educação financeira, muitas vezes levam a um ciclo vicioso de dívidas crescentes. O fácil acesso ao crédito, sem o devido entendimento das implicações financeiras, contribui para que muitas famílias se vejam envolvidas em empréstimos de longo prazo, comprometendo uma parte significativa de suas rendas mensais.

12. Referências Bibliográficas

ANBIMA. Menos da metade dos brasileiros tem dinheiro aplicado em produtos financeiros. Recuperado de https://www.anbima.com.br Acesso em: 18 abr. 2024.

Até o momento. Diagrama de Sequência UML. Recuperado de https://www.ateomomento.com.br/diagrama-de-sequencia Acesso em: 18 abr. 2024.

BANKS, J. React: Up & Running: Building Web Applications. Sebastopol, CA: O'Reilly Media, 2017. Acesso em: 18 abr. 2024.

BEM GASTO. Recuperado de https://www.bemgasto.org Acesso em: 18 abr. 2024.

BORTOLUZZI, Daiane Antonini et al. Aspectos do endividamento das famílias brasileiras no período de 2011-2014. Revista Perspectiva. Rio Grande do Sul, v. 39, n. 146, p. 111-123, 2015. Acesso em: 18 abr. 2024.

DEITEL, P.; DEITEL, H. M. Java: Como Programar. 10ª edição. Porto Alegre: Pearson Education do Brasil, 2017. Acesso em: 18 abr. 2024.

DE QUEIRÓS MATTOSO, Cecília Lima. Me empresta seu nome?: um estudo sobre os consumidores pobres e seus problemas financeiros. Rio de Janeiro: Mauad Editora Ltd. Acesso em: 18 abr. 2024.

DUBOIS, P. MySQL 8 Cookbook: Over 150 Recipes for High-Performance Database Querying and Administration. Birmingham, UK: Packt Publishing, 2016. Acesso em: 18 abr. 2024.

ESTADÃO. Número de famílias pobres é o maior da história. Recuperado de https://www.estadao.com.br/economia/numero-de-familias-pobres-com-dividas-alcanca-nova-maxima-historica Acesso em: 18 abr. 2024.

GERENCIADOR Financeiro e de Investimentos. Disponível em: https://www.meudinheiroweb.com.br/ Acesso em: 18 abr. 2024.

JKOLB. Tag: Diagrama de Atividades. Recuperado de https://www.jkolb.com.br/diagrama-de-atividades Acesso em: 18 abr. 2024.

OIKOS: Revista Brasileira de Economia Doméstica. Viçosa, v. 27, n.2, p. 152-174, 2016. Empréstimos consignados e endividamento familiar: estudo junto a servidores públicos federais em Pernambuco. Acesso em: 18 abr. 2024.

USERWAY. Documentação UserWay. Disponível em: https://userway.org/docs/#userway-api. Acesso em: 20 maio 2024.

ORGANIZZE: Controle as suas finanças pessoais. Disponível em: https://www.organizze.com.br/ Acesso em: 25 mai. 2024.

ORÇAMENTO Pessoal. Disponível em: https://www.orcamentopessoal.net/ Acesso em: 20 mai. 2024.

SANTOS, Adla Carla; SILVA, Maciel. Importância do planejamento financeiro no processo de controle do endividamento familiar: um estudo de caso nas regiões metropolitanas da Bahia e Sergipe. Revista Formadores, v. 7, n. 1, p. 05-17, 2014. Acesso em: 23 mai. 2024.

SOUZA, A. C. de. Spring Boot: Acelere o desenvolvimento de aplicações Java. São Paulo: Novatec, 2020. Acesso em: 22 mai. 2024.

STACK OVERFLOW. Exemplo de diagrama de atividades. Recuperado de https://www.stackoverflow.com/diagrama-de-atividades Acesso em: 25 mai. 2024.

USERWAY. UserWay. Disponível em: https://userway.org/pt/. Acesso em: 20 maio 2024.