UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

Daniel José di Navaronne Gaudêncio Leite

EcoCalc:
Sistema de gerenciamento de despesas residenciais e pessoais

CAMPINA GRANDE - PB

2022

Daniel José di Navaronne Gaudêncio Leite

EcoCalc:
Sistema de gerenciamento de despesas residenciais e pessoais

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Orientador : Professor Dr. José Antão Beltrão Moura

CAMPINA GRANDE - PB

2022

Daniel José di Navaronne Gaudêncio Leite

EcoCalc:
Sistema de gerenciamento de despesas residenciais e pessoais

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

BANCA EXAMINADORA:

Professor Dr. José Antão Beltrão Moura

Orientador – UASC/CEEI/UFCG

Professor Dr. Carlos Wilson Dantas De Almeida

Examinador – UASC/CEEI/UFCG

Professor Dr. Francisco Vilar Brasileiro

Professor da Disciplina TCC – UASC/CEEI/UFCG

Trabalho aprovado em: 02 de Setembro de 2022.

CAMPINA GRANDE - PB

RESUMO

Muitas pessoas acabam tendo dificuldade com despesas todos os meses e não conseguem controlar e gerir com o que pode tá gastando, pois acaba não anotando ou se preocupando com os consumos seus e dos seus dependentes. Com isso, esse trabalho tem como objetivo desenvolver uma aplicação Android, com o intuito de fornecer uma plataforma na qual o usuário possa gerir e ter controle das suas despesas pessoais e residenciais mensalmente.

EcoCalc: Sistema de gerenciamento de despesas residenciais e pessoais

Trabalho de Conclusão de Curso

Daniel José Di Navaronne G. Leite
Universidade Federal de Campina Grande
Campina Grande, Paraíba, Brasil
daniel.jose.leite@ccc.ufcg.edu.br

José Antão Beltrão Moura
Universidade Federal de Campina Grande
Campina Grande, Paraíba, Brasil
antao@computacao.ufcg.edu.br

RESUMO

Muitas pessoas acabam tendo dificuldade com despesas todos os meses e não conseguem controlar e gerir com o que pode tá gastando, pois acaba não anotando ou se preocupando com os consumos seus e dos seus dependentes. Com isso, esse trabalho tem como objetivo desenvolver uma aplicação Android, com o intuito de fornecer uma plataforma na qual o usuário possa gerir e ter controle das suas despesas pessoais e residenciais mensalmente.

PALAVRAS-CHAVE

Gerenciador financeiro, organização econômica, aplicativo mobile, React Native, gerenciar despesas.

1. INTRODUÇÃO

O planejamento desempenha um papel muito importante no controle financeiro, fazendo com que a organização dos gastos realizados possam ser avaliados e distinguidos do que realmente é importante e o que não é. Essa distinção, pode auxiliar uma pessoa ou uma empresa a ter uma melhor gestão e economia do seu dinheiro de forma gradativa.

Para que isso seja possível - no caso de uma pessoa - é necessário detectar todas as receitas mensais, que podem vir a partir do salário, trabalhos extras, investimentos ou de inúmeras outras formas [1]. Com isso, o planejamento financeiro exige que a pessoa relacione todas as suas despesas a serem pagas. Assim, será possível identificar gastos extras que podem prejudicar seu orçamento.

São muitos os perigos do descontrole financeiro, entre eles está a inadimplência que causa ainda mais custos para a pessoa afetada. Além disso, o acúmulo de dívidas impacta a saúde mental das pessoas. Não são raros quadros de insônia, ansiedade, e até depressão, em quem não consegue controlar suas contas [2]. De acordo com o mais recente levantamento do Mapa de Inadimplência da Serasa [3], em maio de 2022 haviam mais de 66 milhões de endividados no Brasil, número que cresceu 0,68% em relação ao mês de abril. Isso indica que muitas pessoas podem estar nessa situação, por não terem o gerenciamento de forma correta das suas despesas.

Diante disso, o Ecocalc se propõe a criar e fornecer uma aplicação para dispositivos Android, na qual o usuário poderá gerenciar suas despesas, pessoais e residenciais, indicando em quais podem estar apresentando gastos elevados em relação ao mês anterior e fazendo cálculos para um melhor entendimento e organização de suas contas mensais.

2. SOLUÇÃO

Em uma pesquisa realizada pelo IBGE do final de 2018 ao final de 2019, a proporção de domicílios onde há pelo menos um telefone celular é de 93,2% [4]. Isso mostra que nos tempos de hoje, a população tem acesso a possíveis tecnologias que possam ajudar no dia-a-dia pessoal e de sua família. Com isso, o EcoCalc é um aplicativo com o intuito de auxiliar as pessoas que não possuem um conhecimento prévio de como organizar suas despesas e gastos. A ideia é trazer uma aplicação com layout simples mas que possa ser útil na organização financeira do usuário, fazendo com que as pessoas que não possuem educação financeira começam a se organizar em relação a suas despesas pessoais e residenciais mensalmente [Figura 1].

Figura 1 - Tela inicial do usuário

2.1. DESCRIÇÃO

O aplicativo se propõe a gerenciar e calcular as despesas que o usuário venha a ter, e organizar tais despesas de forma mensal sendo distinguidas em despesas que são pessoais e despesas residenciais. Nele, o usuário consegue ter uma melhor organização da sua vida financeira a partir de um sistema simples e que busca a efetividade no controle financeiro.

Algumas opções de gerenciamento financeiro já existem, como o MS Money, *Organizze* e o *Money Pro*, mas o sistema desenvolvido neste trabalho traz alguns pontos que esses dois concorrentes não possuem: a possibilidade de gerenciar tanto as despesas pessoais do usuário quanto às despesas de seus dependentes e de residências que o usuário gerencie as economias mensais e que não tenha um custo mensal para o usuário utilizar das funcionalidades disponíveis.

Além disso, o trabalho também busca criar uma relação entre as pessoas nas quais são dependentes do usuário, podendo trazer uma relação na qual o mesmo pode vir a ter despesas em diversas outras casas que não necessariamente seja a que ele mora.

Figura 2 - Listagem de residências

Figura 3 - Listagem de despesas de uma residência

2.2. FUNCIONALIDADES

- **Cadastro de pessoas e residências:** Com uma conta criada no sistema, o usuário poderá fazer o cadastro de pessoas que tiverem despesas a serem gerenciadas pelo usuário. Além do cadastro de residências que também devem entrar no gerenciamento mensal do usuário.

- **Cadastro de despesas:** O usuário poderá cadastrar despesas no seu gerenciamento que podem estar relacionadas a uma pessoa, uma residência ou ambas.

- **Visualização de variação mensal de gastos:** Na tela principal do aplicativo, o usuário consegue visualizar cards dos últimos 12 meses contendo informações importantes, como o total gasto naquele mês, a quantidade de despesas e a porcentagem de variação do valor total gasto em relação ao mês anterior. Além disso, o usuário terá um campo para filtrar qual pessoa ou residência deverá listar e fornecer essa visualização.

- **Listagem de pessoas e residências:** O aplicativo também tem a funcionalidade de listagem e visualização das pessoas e residências já cadastradas pelo usuário. Nessa listagem, assim como a da tela principal, o usuário poderá escolher se quer ver se todas as pessoas e residências ou se quer ver de uma específica, podendo também ser filtrado pelo mês [Figura 2].

- **Listagem detalhada de despesas de cada pessoa ou residência:** Na tela de listagem de pessoas ou residências, o usuário tem a visualização de cada

pessoa e residência, e ao clicar em um elemento dessa listagem, será aberta a visualização detalhada de cada despesa desse elemento, seja pessoa ou residência [Figura 3].

2.3. ARQUITETURA

Para o desenvolvimento do projeto, o sistema foi dividido em duas partes, *front-end* e *back-end*, sendo o *front-end* a parte que o usuário final utiliza (*mobile*) e que se comunica com o *back-end* através de requisições GraphQL (*queries* e *mutations*). Já o *back-end*, é a parte que persiste todos os dados da aplicação e responsável pela autenticação de usuários, na qual recebe as requisições GraphQL vinda do *front-end*.

Por motivos dos dados serem totalmente relacionados, escolhemos usar o banco de dados relacional PostgreSQL.

2.3.1. TECNOLOGIA DO BACK-END

Para este trabalho, foi escolhido usar o *framework Spring Boot*, pois se trata de uma *framework* de Java, na qual foi amplamente ensinada no decorrer da graduação. Também foi escolhido usar a linguagem de consulta GraphQL, por facilitar as requisições feitas na aplicação, fazendo com que apenas dados solicitados sejam fornecidos.

Além disso, utilizamos o JWT como token de autenticação nas requisições ao *back-end* e PostgreSQL como banco de dados.

2.3.2. ESTRUTURA DO BACK-END

O *back-end* da aplicação é baseado em GraphQL como a linguagem de consulta para fazer as requisições, na qual podemos requisitar dados a partir de *schema* criados de forma estaticamente tipada. É a partir desses *schemas* que as *queries* são executadas e consequentemente validadas, tanto entrada quanto saída. Seguindo padrões já utilizados para projetos em *Spring Boot* com GraphQL, a estrutura foi dividida em seis partes, são elas: models, *repositories*, *services*, *resolvers* e *resources* [Figura 4].

O GraphQL Server é a ligação por onde o *front-end* se comunica com o *back-end* a partir de um *endpoint* GraphQL. No *front-end*, são chamadas funções definidas nos *schemas* (*queries* e *mutations*), que são retornadas de funções Java criadas nos *resolvers*.

Esses *resolvers*, não possuem regra de negócio, mas invocam as funções lógicas presentes nos *services* e retorna dados dos repositories, que possuem os dados das *models* que estão sendo buscada e/ou alterada na atual função chamada.

Models são responsáveis por gerenciar e controlar a forma como os dados se comportam por meio das funções, lógica e regras de negócios estabelecidas. Recebem os dados vindos dos *repositories* e validam se ela está correta ou não e envia a resposta mais adequada.

Os repositories são responsáveis por retornar os dados do banco a partir do JPA do *Spring Boot* a partir da lógica passada no *service* e das regras dos *models*.

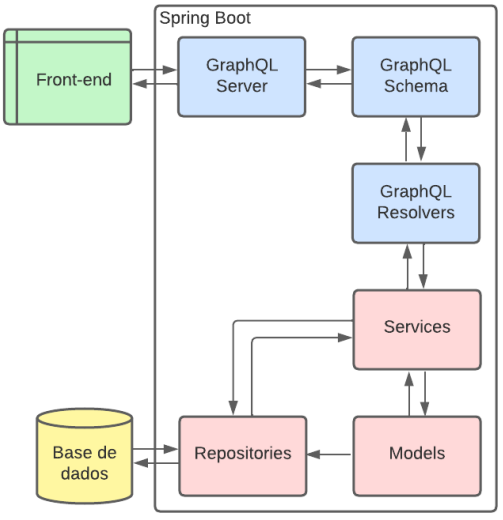

Figura 4 - Arquitetura do *back-end*

2.3.3. TECNOLOGIA DO FRONT-END

No nosso sistema, utilizamos a tecnologia *React Native* para o *front-end* pelo motivo de ser uma linguagem nativa, podendo criar a aplicação tanto para dispositivos Android como para iOS, com o reaproveitamento do mesmo código. A aplicação do sistema de gestão foi desenvolvida apenas para Android, porém pode ser estendida futuramente para IOS com mais facilidade.

Além disso, o *React Native* é amplamente usado e possui uma comunidade bastante forte no meio tecnológico, e por causa disso, o *React Native* possui uma vasta gama de bibliotecas que podem ser usadas, sejam elas gráficas, gerenciamento de estados, etc.

2.3.4. ESTRUTURA DO FRONT-END

A parte *mobile* tem uma estrutura já indicada pela documentação e tutoriais sobre *React Native* na web. Como mostrado na Figura 5, podemos ver que a aplicação é dividida em quatro partes. Que são: *Components*, *Pages*, *Routes* e *Services*.

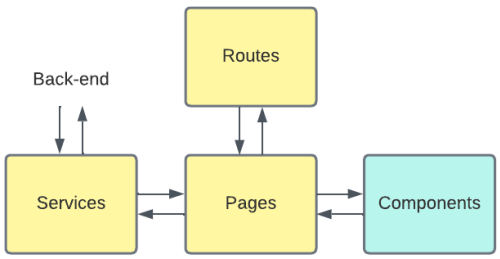

Figura 5 - Arquitetura do *front-end*

Service é onde é feita a conexão com o back-end e por onde são feitas as requisições ao mesmo. Como usamos GraphQL no nosso projeto, utilizamos da biblioteca Apollo Client para o gerenciamento de estado que permite gerenciar dados locais e remotos com o GraphQL. Com o Apollo Client, conseguimos fazer nossas requisições para o back-end solicitando apenas os dados necessários para o uso em determinada tela.

Pages são a principal parte do *front-end*, onde ficam localizadas todas as telas do aplicativo e as configurações de navegação entre elas. Em nossa solução utilizamos o StackNavigator e o DrawerNavigator.

Os *Components*, como o próprio nome diz, são onde se localizam todos os componentes da aplicação. O *React Native* permite essa componentização [7], facilitando assim o reuso destes nas páginas.

E por último, as *Routes* é onde diferenciamos as telas na qual um usuário autenticado pode acessar ou não. Como usamos JWT (*JSON Web Token*) na nossa aplicação, um usuário ao logar no sistema, gera um token que é gravado no *local storage* do aplicativo e faz a autenticação se o usuário está logado ou não. Com esse *token* salvo, conseguimos ter a informação se o usuário está logado, e direcionamos para o mesmo ter acesso apenas a determinadas telas. O mesmo acontece para o usuário não logado, podendo acessar apenas telas que não precisam de autenticação.

Figura 6 - Tela de login para usuário não autenticado

3. METODOLOGIA

3.1 PROCESSO DE DESENVOLVIMENTO

Inicialmente foi feito uma avaliação de quais tecnologias seriam utilizadas para o desenvolvimento do projeto, escolhendo aquelas que pudessem auxiliar no tempo de desenvolvimento e na qualidade do produto final. Com isso, as tecnologias *React Native* [5] *(front-end)*, *Spring Boot* [6] *(back-end)* e PostgreSQL [7] (banco de dados) foram as decididas por trazerem o necessário para a aplicação ser robusta e atender a demanda da ideia.

Em seguida, é feita a elicitação dos requisitos que seriam necessários para atender a funcionalidades idealizadas para o aplicativo com o intuito de deixar o sistema com a usabilidade mais simples para entendimento do usuário.

Com os requisitos levantados, seguimos para as prototipagens das telas do aplicativo. Essa etapa foi feita no Figma, e foi estudada e feita para seguir à risca os designs em relação às funcionalidades que teriam na aplicação.

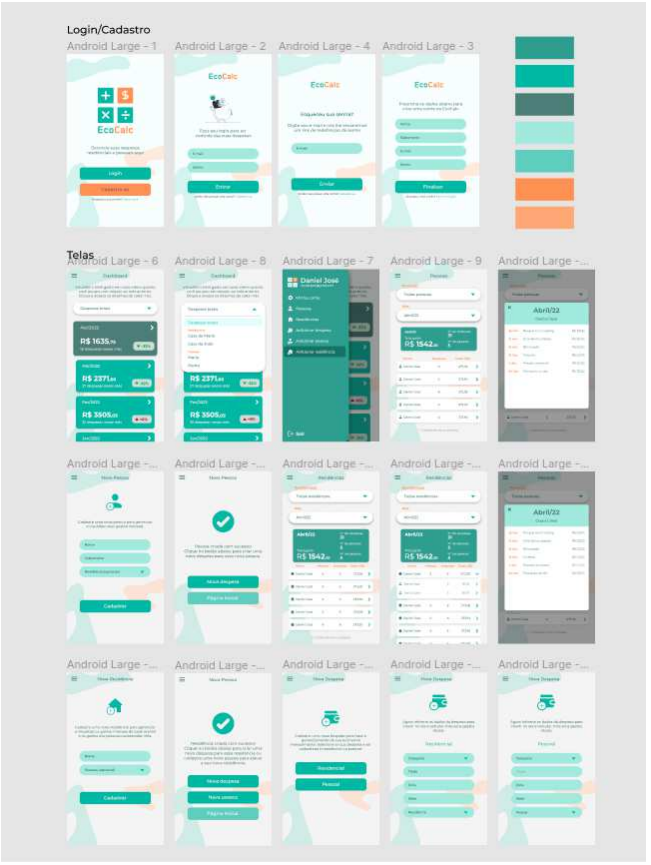

Figura 8 - Prototipagens feitas no Figma

Com o design do EcoCalc concluído, o sistema foi sendo criado, até que, por fim, foi feito o *deploy* do *back-end* em uma instância do Heroku. Assim como, foi gerado o apk do aplicativo para que pudéssemos colocar em produção.

3.2 PRINCIPAIS DESAFIOS

O principal desafio deste trabalho foi levantar requisitos para o sistema que se tornassem úteis e simples para a usabilidade do usuário, com o intuito de trazer uma boa experiência para quem o utiliza, mas focando na organização financeira mensal do mesmo.

Os componentes usados para a implementação das telas do aplicativo

também foi um desafio, pois no *React Native* não foi possível encontrar componentes que seguissem o *design* antes produzido e que fossem úteis para as funcionalidades em si.

4. RESULTADOS

Após a finalização do aplicativo, o apk gerado foi disponibilizado para 14 estudantes do ensino superior de diversas áreas usarem o EcoCalc. Após o uso, foi disponibilizado um formulário com algumas afirmações acerca de como foi a usabilidade do aplicativo. Foram elas:

1. O aplicativo é útil para gerenciamento de despesas.
2. O aplicativo tem uma fácil usabilidade.
3. O aplicativo possui todas funcionalidades que preciso.
4. O aplicativo apresenta inconsistências que inviabilizam o uso.
5. A minha experiência com o EcoCalc foi positiva.

Essas afirmações tinham como resposta uma escala linear de 1 a 5, na qual 1 representa que o respondente concorda plenamente com a afirmação e 5 que discorda plenamente com a afirmação.

Com base nos resultados da pesquisa, vimos que os usuários que utilizaram do aplicativo acharam que o sistema é útil para o gerenciamento de despesas e possui uma fácil usabilidade. Apesar disso, constatamos a partir da pesquisa que o EcoCalc precisa de mais funcionalidades para atender todo tipo de usuário e suas necessidades. Em geral, obteve-se um resultado positivo em relação a quantidade de pessoas participantes da pesquisa.

5. TRABALHOS FUTUROS

O trabalho foi pensado para auxiliar as pessoas com descontrole econômico, tratar a desorganização financeira e fazer o gerenciamento das despesas mensais tanto pessoais como residenciais. Apesar disso, há um potencial do produto em se ampliar para o gerenciamento dos ganhos financeiros do usuário, aumentando assim o leque de opções de funcionalidades e também de complementação para uma plataforma mais completa no ramo financeiro.

Além disso, o sistema poderá ser ampliado para o uso em médias e pequenas empresas. Em pequenas empresas, o gestor poderá gerenciar seus lucros e gastos mensais com produtos relacionados a seu ramo. Para médias empresas, além de gerir o financeiro dos produtos, teria a opção de gerir suas franquias de forma individual.

6. REFERÊNCIAS

[1] dos Santos Rocha, Gizele. “Finanças Pessoais”. Monografias Brasil Escola, 2010. Disponível em: https://monografias.brasilescola.uol.com.br/administracao-financas/financas-pessoais.htm. Acesso em: 20 de julho de 2022.

[2] Ferreira, Vanessa. “Como sair do descontrole financeiro e manter as finanças em dia”. Serasa, 2021. Disponível em: https://www.serasa.com.br/score/blog/como-sair-do-descontrole-financeiro-e-manter-as-financas-em-dia. Acesso em: 9 de agosto de 2022.

[3] SERASA. Serasa, 2022. Mapa da inadimplência e renegociação de dívidas no Brasil. Disponível em: https://www.serasa.com.br/limpa-nome-online/blog/mapa-da-inadimplencia-e-renogociacao-de-dividas-no-brasil. Acesso em: 9 de agosto de 2022.

[4] IBGE Educa. IBGE, 2019. Uso de Internet, televisão e celular no Brasil. Disponível em: https://educa.ibge.gov.br/jovens/materias-especiais/20787-uso-de-internet-televisao-e-celula

r-no-brasil.html#:~:text=Em%204%2C7%25%20das%20resid%C3%AAncias,m%C3%B3vel%20celular%20para%20uso%20pessoal. Acesso em: 12 de agosto de 2022.

[5] React Native, 2022. Introduction. Disponível em: https://reactnative.dev/docs/getting-started. Acesso em: 17 de agosto de 2022.

[6] Spring Boot, 2022. Spring Boot Reference Documentation. Disponível em: https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/. Acesso em: 17 de agosto de 2022.

[7] PostgreSQL, 2022. Documentation. Disponível em: https://www.postgresql.org/docs/. Acesso em: 17 de agosto 2022.