

## BILL: DESENVOLVIMENTO E VALIDAÇÃO DE UMA APLICAÇÃO MÓVEL PARA GERENCIAMENTO FINANCEIRO PESSOAL

Guilherme Henrique Ferreira Assis

Graduando em Sistemas de Informação - Uni-FACEF

guilherme\_hrq99@outlook.com

João Vitor de Oliveira Rodrigues

Graduando em Sistemas de Informação - Uni-FACEF

joaovitoroliveira19992015@gmail.com

Orientador: Prof. Me. Geraldo Henrique Neto

Mestre em Ciências com Ênfase em Informática Médica - FMRP-USP

geraldohenrique@usp.br

### RESUMO

É cada vez mais comum, identificarmos pessoas que não possuem nenhum tipo de controle de sua vida financeira, diversas pesquisas realizadas ao longo dos últimos anos, mostram que milhões de brasileiros não realizam nenhum tipo de gerenciamento de suas finanças, e grande parte dos que realizam, fazem de forma errada ou utilizam métodos ineficazes. Com isso diversos problemas são gerados, e o pior deles, o endividamento, já afeta milhões de pessoas por todo o país. Este projeto teve como objetivo desenvolver um aplicativo para apoiar o controle financeiro, o qual realizará a tarefa de armazenar todas as receitas e despesas, transferências, agendamento de contas, controle de objetivos, além de disponibilizar diversos gráficos para se analisar e juntamente com dicas de economia. Desta forma, o usuário pode cadastrar movimentações diárias de ganhos e gastos e o aplicativo armazena estas informações e apresenta ao usuário de forma categorizada ou através de gráficos em um *dashboard*, para uma melhor interpretação desses dados, além de fornecer dicas de educação financeira. Este artigo demonstra técnicas de desenvolvimento de aplicações computacionais móveis juntamente com os processos de documentação e modelagem de software.

**Palavras-chave:** Aplicação móvel. Gestão financeira. Educação financeira.

## ABSTRACT

*It is increasingly common to identify people who have no control over their financial lives, several surveys carried out over the past few years show that millions of Brazilians do not manage their finances at all, and most of those who do, do it wrongly or use ineffective methods. As a result, several problems are created, and the worst of them, indebtedness, already affects millions of people across the country. This article aimed to develop a mobile application to assist and teach people how to better manage their finances. In this way, the user can register daily movements of earnings and expenses and the application stores this information and presents it to the user categorically or through graphics on a dashboard, for a better interpretation of this data, in addition to providing financial education tips. This article demonstrates techniques for developing mobile computing applications together with the software modeling and documentation processes.*

**Keywords:** *Mobile application. Financial management. Financial education.*

## 1 INTRODUÇÃO

Fazer um controle financeiro pessoal não é algo que podemos classificar como prazeroso. Assim como ninguém gosta de cortar seus pratos preferidos do cardápio, pouquíssimas pessoas ficam contentes em poupar dinheiro e abrir mão de comprar bens que trazem um nível considerável de satisfação, certo? Neste ponto que a ideia discutida neste artigo se encaixa, desenvolver um aplicativo que possa realizar o controle financeiro. A maioria das pessoas ou não fazem esse controle ou fazem em uma planilha, que é pouco seguro e eficiente.

Nos dias atuais, não possuir um controle de sua vida financeira é algo muito comum entre milhões de brasileiros e com isso vários problemas são gerados e se agravam ao longo dos anos, além de ser um dos principais motivos dos mais temidos, a instabilidade financeira e o endividamento.

Segundo pesquisas do SPC (Serviço de Proteção ao Crédito) Brasil, mais de 47% dos jovens entre 18 e 30 anos não faz nenhum controle de seus ganhos e gastos, o que acaba gerando o endividamento e tornando cada vez mais difícil sair dessa situação. Dentre as milhões de pessoas que não realizam um controle de sua vida financeira, os principais motivos são de não saberem como fazer e muitas vezes por preguiça (SOLDI, 2019).

Pensando nisso o objetivo do projeto é desenvolver um aplicativo para apoiar o controle financeiro, o qual realizará a tarefa de armazenar todas as receitas e despesas, transferências, agendamento de contas, controle de objetivos, além de disponibilizar diversos gráficos para se analisar e juntamente com dicas de economia.

Foi realizada uma pesquisa de mercado para validação da ideia proposta, com perguntas voltadas a entender os problemas relacionados a vida financeira de cada participante, quais os possíveis motivos que lhes renderam esses problemas e quais as medidas que tomam para tentar resolver ou minimizar as dores citadas. Em conjunto, foi apresentada a elas a ideia principal do projeto Bill, e questionado aos participantes da pesquisa se o mesmo atendia a suas necessidades e quais outras funcionalidades seriam importantes para serem oferecidas pela ferramenta.

A pesquisa foi realizada com mais de 20 participantes, e o *feedback* colhido foi significativamente positivo e motivador, a ideia foi bem aceita por todos os participantes, inclusive todos se voluntariando a experimentar o produto assim que disponibilizada sua primeira versão, e isso foi muito importante para que o projeto fosse realizado. Os resultados da pesquisa de validação da ideia estão disponíveis na íntegra através do Github (ASSIS E RODRIGUES, 2020).

Focando no desenvolvimento do projeto, foram definidas tecnologias que garantem uma implementação simples, mas com performance e resultados robustos. Para a criação do aplicativo *mobile* foi escolhido o SDK (*Software Development Kit*) Flutter, por sua fácil curva de aprendizagem, ampla gama de ferramentas que facilita e diminui o tempo de desenvolvimento, além de ser totalmente híbrido, ou seja, possibilidade de gerar um aplicativo para Android e iOS com apenas um código fonte.

Para realizar a comunicação entre o aplicativo e nosso banco de dados, servindo as informações a aplicação, será criada uma API (*Application Programming Interface*) em Node.js, um *framework* JavaScript amplamente utilizado pelo mercado por conta de sua simplicidade e poder de processamento. Para facilitar a comunicação entre a aplicação e o *backend*, a API será toda desenvolvida aplicando o padrão REST (*Representational State Transfer*) que simplifica todas as chamadas que serão realizadas.

Pensando na armazenagem dos dados dos usuários do sistema, será criado um banco de dados baseado em SQL (*Structured Query Language*) por meio do sistema gerenciador de bancos de dados, intitulado de PostgreSQL. Uma das

principais características desse SGBD é ser gratuito e ser reconhecido no mercado por sua robustez, além de ser um dos mais utilizados do mercado.

Na seção dois é apresentado o referencial teórico com os motivos de se manter um controle de sua vida financeira, os benefícios de se utilizar sistemas de informação como aliado em seu controle financeiro, além de uma introdução e motivo de cada tecnologia escolhida para desenvolvimento do projeto.

Na seção três é abordado o tema empreendedorismo, apresentando o conceito de *startup* e o modelo de negócio Canvas, que foi desenvolvido durante o projeto.

Na seção quatro são descritas as etapas de engenharia de software com os diagramas e fluxos juntamente com a modelagem do banco de dados da aplicação.

Na seção cinco é abordado o desenvolvimento do aplicativo Bill, é apresentada a prototipação das telas do projeto, com uma breve descrição do fluxo das mesmas, a implementação do aplicativo *mobile*, o desenvolvimento da *API* e o meio de comunicação entre o *back-end* e o banco de dados.

Na seção seis é abordada a conclusão, nela apresentamos uma análise sobre os objetivos alcançados no desenvolvimento do projeto.

## 2 REFERENCIAL TEÓRICO

Nesta seção serão apresentados os motivos de se manter um controle de sua vida financeira, os benefícios de se utilizar sistemas de informação como aliado em seu controle financeiro, além de uma introdução e motivo de cada tecnologia escolhida para desenvolvimento do projeto.

### 2.1 CONTROLE FINANCEIRO E SUA IMPORTÂNCIA

Ter um controle de sua vida financeira é de extrema importância, pois previne diversos problemas que a falta do mesmo pode acarretar. Ter controle financeiro pessoal é a forma mais básica de cuidar do seu dinheiro. É assim que você conhece exatamente sua renda, os seus gastos e o que pode ser melhorado para sobrar mais dinheiro no fim do mês. Dinheiro para fazer uma reserva financeira, investir e multiplicar sua renda (INVESTIMENTOS, 2018).

De acordo com a INVESTIMENTOS (2018), controle financeiro pessoal é o hábito de organizar todas as receitas e despesas no período, geralmente mensal, considerando tanto as contas fixas quanto as despesas gerais. Esse é um hábito importante porque muitas pessoas não sabem quanto ganham e quanto gastam por mês, e essa é a principal razão do descontrole financeiro.

## 2.2 SISTEMAS DE INFORMAÇÃO E O CONTROLE FINANCEIRO

A cada dia que se passa, a tecnologia vem sendo aderida por diversas áreas, e no meio financeiro não poderia ser diferente. Com inúmeros recursos tecnológicos disponíveis na atualidade, nada melhor do que inovar em sua área de atuação, fazendo com que a Tecnologia da Informação (TI) trabalhe ao seu favor, melhorando as finanças por meio da utilização dos dados como ferramentas nas tomadas de decisões — ou seja, quanto mais informação precisa, menores os erros (TAINO, 2018).

Com as inovações tecnológicas, crianças e jovens já podem ser educados financeiramente de maneira mais fácil e interessante. Como falamos, os *apps*, *sites* e outros recursos ajudam no controle do dinheiro. As tecnologias são boas alternativas para entender os gastos por oferecerem um visual mais claro, além de muitas oferecerem ferramentas de organização que facilitam o planejamento das finanças para alcançar metas, por exemplo. (S.O.S, 2019)

Realizar o controle de sua vida financeira pode não ser uma tarefa simples, a não ser que tenha em mãos as ferramentas certas. Com a ajuda dos sistemas de informação essa tarefa se torna mais simples, rápida e eficaz. Além disso tudo, “O uso de um sistema de controle financeiro pode proporcionar é o acesso às informações em qualquer lugar” (Ever Flow, 2017).

De acordo com S.O.S (2019) já que você tem conhecimento sobre as despesas da sua vida financeira, depois de organizá-las, é preciso saber o que fazer com essas informações. Com um aplicativo, também é possível aprender e compreender sobre dinheiro, pois os dados são apresentados de forma simples e visual.

## 2.3 DESENVOLVIMENTO MOBILE COM FLUTTER

O desenvolvimento de aplicativos híbridos e com performance de aplicativos nativos tem sido um grande problema do desenvolvimento *mobile*.

Criar aplicativos móveis com aparência, funcionalidade e performance de aplicativos nativos e também híbridos é uma proposta complicada, mesmo depois de todos esses anos de desenvolvedores trabalhando para alcançar esse objetivo. Você pode escrever código nativo para cada plataforma e fazer o possível para criar eles o mais semelhante possível, e essa é certamente uma boa maneira de obter desempenho nativo e recursos em seu aplicativo. Mas, efetivamente, isso significa escrever seu aplicativo várias vezes. Os clientes tendem a não gostar de pagar por esse tipo de coisa! (ZAMMETTI, 2019).

Com isso, o Google desenvolveu o Flutter, um pacote de ferramentas de interface do usuário que busca se tornar a principal ferramenta para desenvolvimento *mobile*.

Flutter é uma SDK para desenvolvimento mobile lançada em 2017 pela Google, capaz de criar *apps* para Android e iOS com um único código. O framework foi totalmente desenvolvido em Dart, uma linguagem de propósito geral criada pela Google e muito similar a C# e Java, compilando código nativo para ARM e x86 (HENRIQUE, 2018).

## 2.4 APIS E O NODEJS

As APIs (*Application Programming Interface*) são uma forma simples de comunicação entre sistemas. Em uma definição mais completa, “A API é um conjunto de definições e protocolos utilizados no desenvolvimento e na integração de software de aplicações. API é um acrônimo em inglês que significa interface de programação de aplicações.” (RED HAT, s.d).

Segundo a Red Hat (s.d), as APIs facilitam muito o trabalho de implementação de novos projetos além de garantir uma grande economia de tempo e dinheiro.

Com as APIs, sua solução ou serviço pode se comunicar com outros produtos e serviços sem precisar saber como eles foram implementados. Isso simplifica o desenvolvimento de aplicações, gerando economia de tempo e dinheiro. Ao desenvolver novas ferramentas e soluções (ou ao gerenciar aquelas já existentes), as APIs oferecem a flexibilidade necessária para simplificar o design, a administração e o uso, além de fornecer oportunidades de inovação (RED HAT, s.d).

Neste projeto a API será toda desenvolvida utilizando a tecnologia Node.js criada por Ryan Dahl, ela será a responsável por realizar a comunicação do aplicativo com o banco de dados, buscando e armazenando dados quando for necessário. De acordo com a NodeBR (2016), o Node.js, é uma plataforma construída

sobre o motor JavaScript do Google Chrome para facilmente construir aplicações de rede rápidas e escaláveis. Node.js usa um modelo de I/O direcionada a evento não bloqueante que o torna leve e eficiente, ideal para aplicações em tempo real com troca intensa de dados através de dispositivos distribuídos.

Inspirado no Event Machine do Ruby e no Twisted desenvolvido para Python, Node.js muda o paradigma de desenvolvimento em que tudo depende de Entrada-e-Saída. O Node.js se baseia em uma arquitetura formada por eventos de modo assíncrono e não bloqueável, sem que qualquer método trave sua execução enquanto aguarda alguma entrada ou saída. Dessa maneira, a performance do programa se mantém maior, tanto do ponto de vista de consumo de processamento da CPU quanto com uma vazão maior de informações, facilitando a escalabilidade de sistemas (ASEF, 2019).

Sua forma de trabalho assíncrona é exatamente o que o diferencia de várias outras linguagens que trabalham de modo síncrono, além disso, também é o que o proporciona diversas vantagens em questão de processamento.

A programação tradicional realiza I/O da mesma maneira que executa chamadas de funções locais: O processamento não pode continuar até que a operação termine. Esse modelo de programação que bloqueia enquanto executa operações de I/O deriva do início dos sistemas de *time-sharing* em que cada processo correspondia a um único usuário. O objetivo era isolar os usuários uns dos outros. Naqueles sistemas, um usuário normalmente precisaria acabar uma operação antes de decidir qual seria a próxima a ser executada. Mas com o amplo uso das redes de computadores e da Internet, esse modelo de "um usuário, um processo" não foi bem dimensionado. O gerenciamento de muitos processos sobrecarrega o sistema operacional - na memória e nos custos de troca de contexto - e o desempenho dessas tarefas começa a decair após um certo número ser atingido (TEIXEIRA, 2012, p. 16).

Ele possui esse comportamento graças a sua utilização de *multi-threads*, que possibilitam a execução de mais de um processo ao mesmo tempo o que garante uma melhor performance por ser mais rápida e mais leve. Segundo Teixeira (2012), o *Multi-threading* é uma alternativa para esse modelo de programação. Um *thread* é um tipo de processo leve que compartilha memória com todos os outros *threads* no mesmo processo. Os *threads* foram criados como uma extensão *ad hoc* do modelo anterior para acomodar vários *threads* de execução simultâneos. Quando um *thread* está aguardando uma operação de I/O, outro *thread* pode assumir o controle da CPU. Quando a operação de I/O é concluída, esse *thread* pode ser ativado, o que significa que o *thread* em execução pode ser interrompido e, eventualmente, retomado mais tarde. Além disso, alguns sistemas permitem que os threads sejam executados em paralelo em diferentes núcleos da CPU (*Central Processing Unit*).

## 2.5 POSTGRESQL

De acordo com Matthew e Stones (2005), um sistema de gerenciamento de banco de dados (SGBD) geralmente é um conjunto de bibliotecas, aplicativos e utilitários que liberam um desenvolvedor de aplicativos do fardo de se preocupar com os detalhes de armazenamento e gerenciamento de dados. Ele também fornece facilidades para pesquisar e atualizar registros. Os SGBDs vêm em vários tipos desenvolvidos ao longo dos anos para resolver tipos específicos de problemas de armazenamento de dados.

A definição proposta pela página de documentação do projeto PostgreSQL diz o seguinte, o PostgreSQL é um poderoso sistema de banco de dados objeto relacional de código aberto que usa e estende a linguagem SQL combinada com muitos recursos que armazenam e escalam com segurança as cargas de trabalho de dados mais complicadas. As origens do PostgreSQL remontam a 1986 como parte do projeto POSTGRES da Universidade da Califórnia em Berkeley e tem mais de 30 anos de desenvolvimento ativo na plataforma principal (POSTGRESQL, s.d).

Além disto o PostgreSQL (s.d) complementa que o SGBD vem com muitos recursos destinados a ajudar os desenvolvedores na criação de aplicativos, os administradores a proteger a integridade dos dados e a criar ambientes tolerantes a falhas, além de ajudá-lo a gerenciar seus dados, independentemente de quão grande ou pequeno o conjunto de dados. Além de ser gratuito e de código aberto, o PostgreSQL é altamente extensível. Por exemplo, você pode definir seus próprios tipos de dados, criar funções personalizadas e até escrever código de diferentes linguagens de programação sem recompilar seu banco de dados.

### **3 EMPREENDEDORISMO**

O conceito de Empreendedorismo vem mudando ao longo do tempo e ainda hoje possui várias definições. Segundo o Babson College, o conceito de Empreendedorismo está mais ligado a uma forma de pensar e agir, de agregar valor e não somente a criação de algo novo.

Empreendedorismo é o processo gerencial para criação e gerenciamento de inovação. Uma organização empreendedora é focada nas oportunidades, dando prioridade ao novo e inovador, buscando apostar em pessoas que encontrem formas de como a organização pode capitalizar ideias ou realizar projetos promissores (DRUCKER, 2015).

### 3.1 STARTUPS

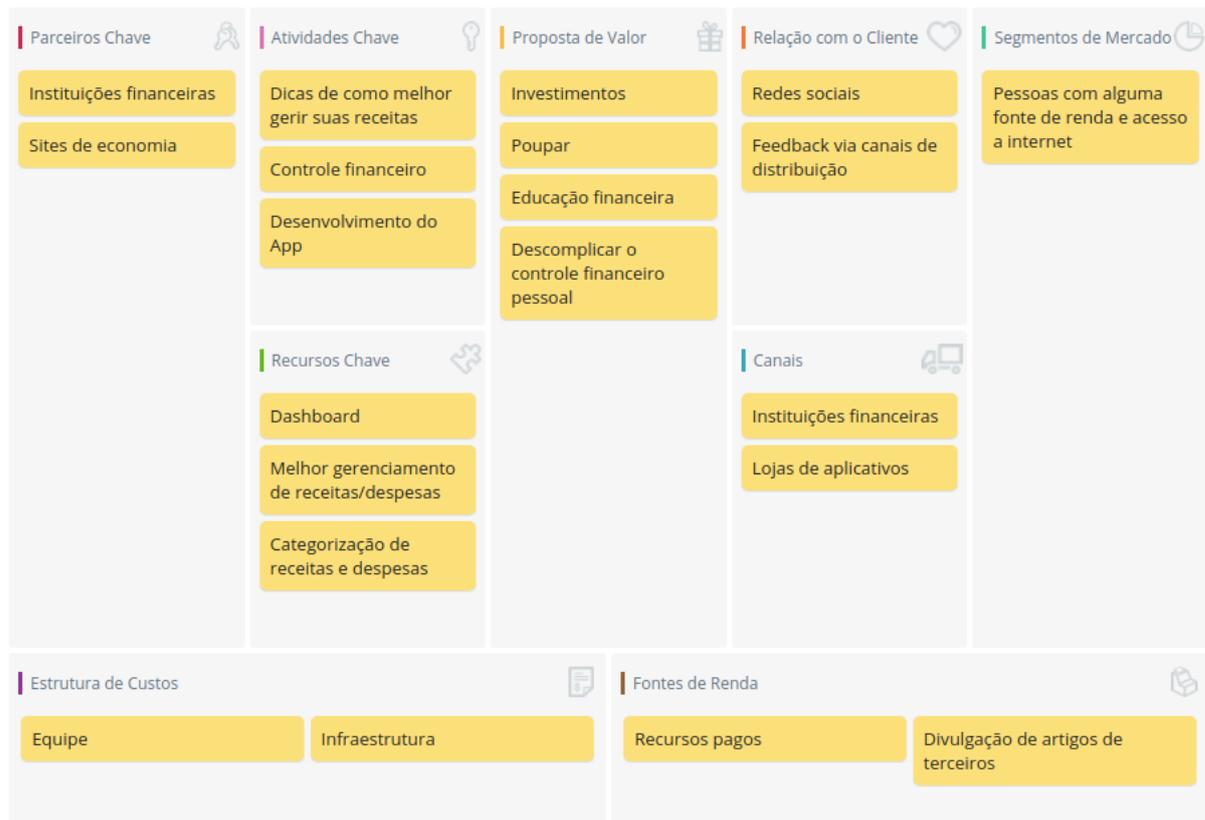
Com as incertezas do mercado crescendo em diversos setores e as empresas em constante mudança, o empreendedorismo tem sido utilizado para colaborar com o negócio por meio de novas ideias de produtos que ajudam a criar uma relação mais próxima com seu cliente final.

Muitas pessoas dizem que qualquer pequena empresa em seu período inicial pode ser considerada uma *startup*. Outros defendem que uma *startup* é uma empresa com custos de manutenção muito baixos, mas que consegue crescer rapidamente e gerar lucros cada vez maiores. Mas há uma definição mais atual, que parece satisfazer a diversos especialistas e investidores: uma *startup* é um grupo de pessoas à procura de um modelo de negócios repetível e escalável, trabalhando em condições de extrema incerteza (MOREIRA, 2016).

### 3.2 CANVAS

O *Business Model Canvas* ou apenas Canvas surgiu nos anos 1990, mas só foi padronizado quando Alex e Pigneur lançaram o livro *Business Model Generation – Inovação em Modelagem de Negócios*. O Canvas é uma ferramenta que fornece uma visão do negócio após preencher todos os nove elementos que toda organização possui, descrevendo o modelo de negócio da organização. A Figura 1 demonstra o modelo canvas desenvolvido para este projeto.

Figura 1 - Modelo Canvas do projeto



Fonte: Os autores.

### 3.2.1 Detalhamento do canvas do projeto

Abaixo serão detalhados os 9 pontos que compõem o modelo Canvas do projeto aqui apresentado, explicando o que foi definido em cada um e o motivo para isso.

- 1) Proposta de valor: É o que de fato sua empresa vai oferecer para o mercado, o que realmente vai agregar valor para seus clientes. E o que nosso projeto oferece é algo bem simples, mas que gera um imenso valor para nossos clientes, queremos descomplicar o controle financeiro pessoal, ajudar na educação financeira dos nossos usuários, mostrar a eles maneiras de poupar seu dinheiro e com isso poder investir para garantir uma renda extra, e para isso queremos dar aos nossos clientes meios tecnológicos que irão auxiliar no controle de suas vidas financeiras e livrando os mesmos de diversos problemas que uma má gestão pode causar;

- 2) Segmento de mercado: O segmento de mercado visa definir quais segmentos de clientes serão o foco da sua empresa e nossa empresa visa atingir pessoas com alguma fonte de renda e acesso à internet;
- 3) Canais: Os canais são como o cliente compra e recebe o nosso aplicativo e nossa empresa distribuirá nas lojas de aplicativos mais conceituadas do mercado e através de instituições financeiras parceiras que podem divulgar e indicar a utilização do aplicativo para seus clientes;
- 4) Relação com cliente: É como a empresa se relaciona com cada segmento de cliente e nossa empresa estabelecerá esse relacionamento via redes sociais e via *feedback* nas plataformas de distribuição do aplicativo;
- 5) Atividades chave: São as atividades essenciais para que seja possível realizar a entrega da Proposta de Valor do projeto. Neste projeto as atividades principais são o desenvolvimento da aplicação *mobile*, pois nesta plataforma será onde os clientes terão acesso a todos os recursos propostos pelo projeto. Ajudar no controle financeiro dos usuários, armazenando a renda e o gasto mensal, organizado em categorias e em dias e com base nessas informações construir gráficos para demonstrar ao usuário onde estão suas maiores fontes de rendas ou onde está gastando mais. O aplicativo disponibiliza dicas ao usuário de como gerir melhor seu dinheiro, com artigos e tutoriais de economia e sobre o mercado financeiro;
- 6) Recursos chave: São os recursos que diferenciam seu projeto de outros, aqueles que realmente vão te fazer ganhar os clientes. Os diferenciais da nossa plataforma de Gerência Financeira Pessoal serão a possibilidade da inserção de rendas e despesas além de se poder categorizar as mesmas, podendo assim diferenciar o que aquela transação representa da maneira que preferir. Um *dashboard* para exibição de gráficos e informações baseados nas categorias e datas de transação para uma melhor análise da sua vida financeira para assim garantir uma melhor gestão do seu dinheiro;

- 7) **Parceiros chave:** Parceiros-chaves, são as atividades principais que são feitas de forma terceirizada e os recursos principais adquiridos fora da empresa, ou seja, são aquelas atividades que são importantes para seu projeto, mas não são de fato realizadas pela empresa e sim são fechadas parcerias com outras pessoas e ou empresas que irão realizar aquela tarefa para você. No nosso projeto temos como premissa realizar parcerias com sites de economia e instituições financeiras, como bancos, corretoras e etc., pois um dos recursos principais que será oferecido por nosso aplicativo são os artigos, e nesse ponto que entram nossas parcerias, pois iremos indexar os artigos de economia produzidos por esses sites;
  
- 8) **Fontes de renda:** São as formas de obter receita por meio de propostas de valor e nossa empresa disponibilizará recursos pagos para melhorar a experiência do usuário e também fornecerá a divulgação de artigos sobre economia para propor o aprendizado para o nosso usuário;
  
- 9) **Estrutura de custos:** Neste ponto são englobados os custos relevantes que são de extrema necessidade para que a estrutura proposta possa funcionar. Pelo fato de o ponto principal do projeto ser a aplicação *mobile*, serão iminentes os gastos para manter essa plataforma disponível 24 horas por dia, 7 dias por semana. Sendo isso indispensável, nossos gastos com infraestrutura será o principal, além é claro, da equipe que será responsável por desenvolver e manter a plataforma interoperante.

## 4 ENGENHARIA DE SOFTWARE

O termo Engenharia de Software foi criado pelo professor Friedrich Ludwig Bauer, hoje aposentado, na década de 1960, época em que ele lecionava na Universidade de Tecnologia de Munique, Alemanha, e, utilizado oficialmente em 1968 na Conferência sobre Engenharia de Software da OTAN (NATO Conference on Software Engineering) (FREITAS, 2015).

Desde então, este termo é amplamente utilizado para identificar a área do conhecimento da informática cujo objetivo é o de especificar (descrever), desenvolver e promover a manutenção em sistemas de software (programas de computador), aplicando tecnologias e boas práticas (práticas consideradas

as mais adequadas) provenientes das disciplinas de ciência da computação e gerência de projetos, entre outras, levando, assim, a produtividade e qualidade (FREITAS, 2015).

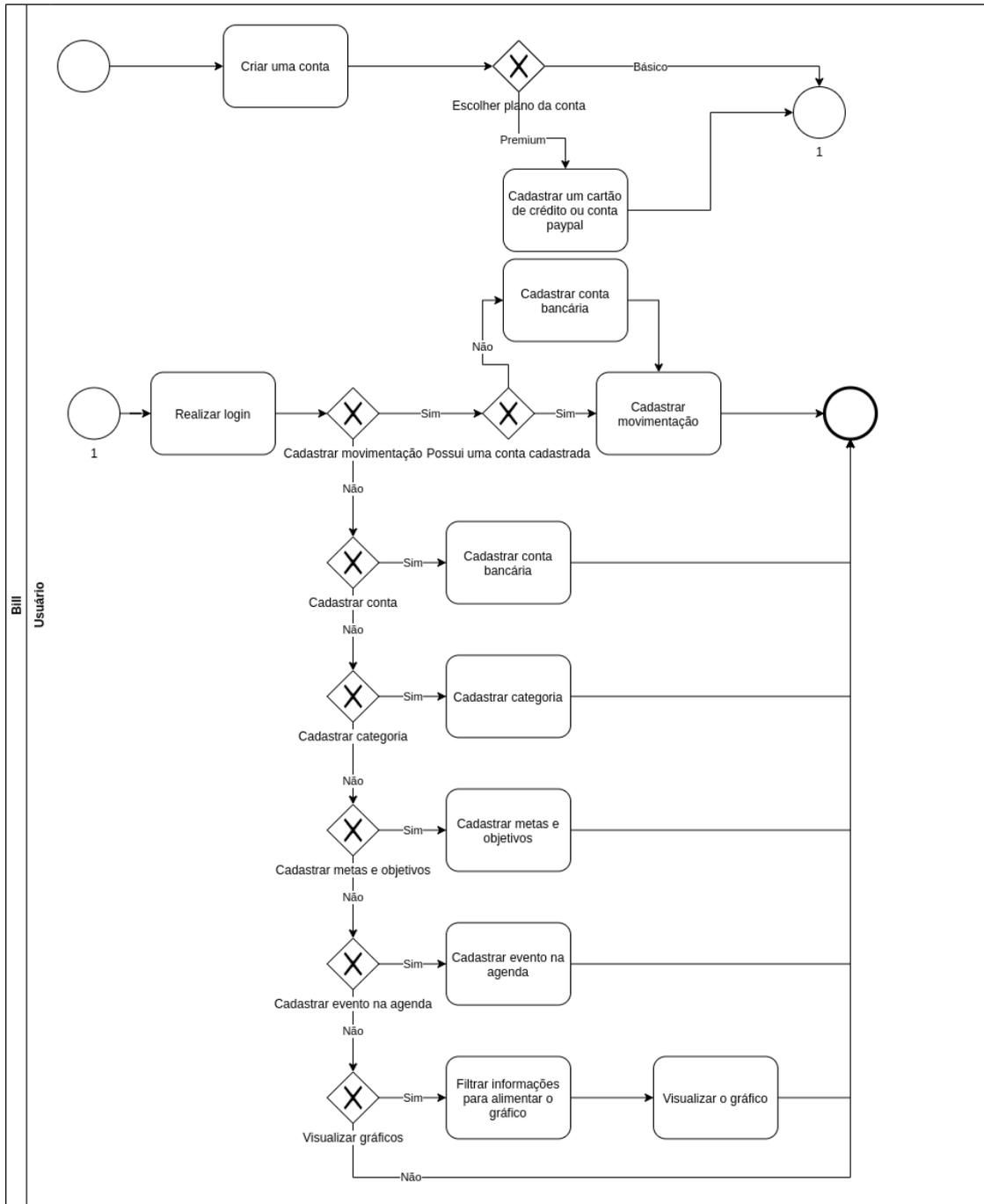
A seguir são apresentadas as documentações de processos realizadas para o projeto, bem como os diagramas BPMN (*Business Process Model and Notation*), Diagrama de Caso de Uso e o Modelo de Entidade Relacionamento, que facilitam o entendimento de todas as atividades da solução e auxiliam em possíveis alterações e correções que o produto poderá sofrer.

## 4.1 BPMN

O modelo BPMN é amplamente utilizada no mundo todo para representar fluxos de sistemas de software, composto por diversos padrões que podem ser compreendidos por qualquer pessoa conhecedora do modelo. A Notação de modelagem de processos de negócio é um método de fluxograma que modela as etapas, de ponta a ponta, de um processo de negócios planejado. Peça-chave na gestão de processos de negócios, representa de forma visual uma sequência detalhada de atividades de negócios e fluxos de informação necessários para concluir um processo (LUCIDCHART, S.D.).

O BPMN do projeto Bill, ilustrado pela Figura 2, representa todo o fluxo da aplicação, desde o processo de autenticação do usuário a criação de contas bancárias, categorias, agendas entre outras funcionalidades.

Figura 2 - BPMN



Fonte: Os autores.

Todo o fluxo do BPMN, apresentado na Figura 2, se inicia quando o usuário cria sua conta no aplicativo e pode escolher o tipo de conta a ser utilizada, sendo o tipo básico, o qual será o padrão ou o premium, um tipo pago que precisa do cadastro do número de um cartão de crédito ou uma conta PayPal para realizarmos o faturamento do valor mensal. Realizando o *login* o usuário ganhará acesso às funcionalidades dependendo do tipo de conta, já podendo realizar o controle de sua

vida financeira. O BPMN e os demais artefatos deste projeto estão disponíveis na íntegra através do Github (ASSIS E RODRIGUES, 2020).

## 4.2 DIAGRAMA DE CASO DE USO

Em sua forma mais simples, um caso de uso identifica os atores envolvidos em uma interação e dá nome ao tipo de interação. Essa é, então, suplementada por informações adicionais que descrevem a interação com o sistema (SOMMERVILLE, 2011).

O Diagrama de Caso de Uso do projeto Bill, ilustrado pela Figura 3, apresenta os fluxos propostos para os principais processos que o aplicativo oferece aos usuários da plataforma.

Observando o Diagrama de Caso de Uso, apresentado na Figura 3, após o usuário se autenticar no aplicativo ele ganha acesso a todas as funcionalidades liberadas para o seu tipo de usuário, sendo de um dos tipos, básico ou premium, podendo cadastrar várias contas bancárias, inserir as movimentações financeiras de cada conta, filtradas por uma categoria de movimentação, que pode ser cadastrada pelo usuário para melhor classificar suas despesas e receitas, e várias outras funcionalidades. O Diagrama de Caso de Uso e os demais artefatos deste projeto estão disponíveis na íntegra através do Github (ASSIS E RODRIGUES, 2020).

Figura 3 - Diagrama de Caso de Uso



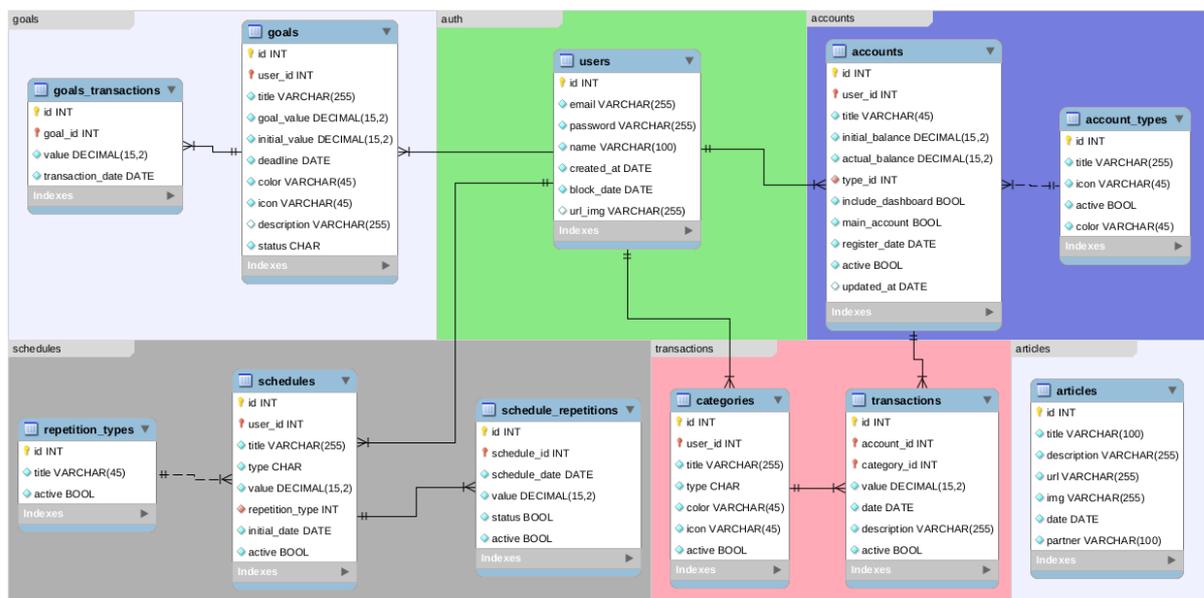
Fonte: Os autores.

O fluxo representado na Figura 3, compreende as funcionalidades e requisitos levantados para o desenvolvimento da primeira versão do projeto, para versões futuras o diagrama apresentado deverá sofrer alterações com a inclusão de novas etapas e até a alteração de etapas já existentes.

### 4.3 MODELO ENTIDADE RELACIONAMENTO

Como as demais modelagens presentes em uma documentação de software, o MER é utilizado em âmbito mundial para representar o modelo entre objetos relacionais. O Modelo Entidade Relacionamento (também chamado Modelo ER, ou simplesmente MER), como o nome sugere, é um modelo conceitual utilizado na Engenharia de Software para descrever os objetos (entidades) envolvidos em um domínio de negócios, com suas características (atributos) e como elas se relacionam entre si (relacionamentos) (RODRIGUES, 2014).

Figura 4 - MER



Fonte: Os autores.

Na Figura 4, podemos observar a estrutura relacional que representa o banco de dados do projeto. Com a figura é possível analisar facilmente as tabelas utilizadas para persistência dos dados relacionados aos usuários, que serão utilizados para autenticação e identificação em todo o sistema, além de tabelas também importantes para armazenar os dados das contas bancárias que o usuário deseja gerenciar, juntamente com categorias para classificação de suas receitas e despesas em conjunto com as transações e movimentações realizadas pelo usuário em suas contas bancárias. O MER e os demais artefatos deste projeto estão disponíveis na íntegra através do Github (ASSIS E RODRIGUES, 2020).

#### 4.4 DIAGRAMA DE CLASSES E LEVANTAMENTO DE REQUISITOS

Foram confeccionados outros artefatos para o desenvolvimento do projeto como o Diagrama de Classes que especifica como os componentes e funcionalidades do projeto comunicam entre si, também foi realizado o levantamento de requisitos para abordar quais as funcionalidades eram de suma importância estarem presentes no projeto. Todos esses artefatos estão disponíveis na íntegra através do Github (ASSIS E RODRIGUES, 2020).

## 5 DESENVOLVIMENTO DO APLICATIVO

Nesta seção será abordado o desenvolvimento do aplicativo Bill, será apresentada a prototipação das telas do projeto, com uma breve descrição do fluxo de cada, a implementação do aplicativo *mobile*, o desenvolvimento da *API* e o meio de comunicação entre o *back-end* e o banco de dados.

### 5.1 TELAS DA APLICAÇÃO E SUAS FUNCIONALIDADES

Nesta seção são apresentadas todas as telas presentes no MVP (*Minimum Viable Product*) da aplicação móvel Bill, em conjunto com as especificações de suas funcionalidades. A imagem das telas do aplicativo e os demais artefatos deste projeto estão disponíveis na íntegra através do Github (ASSIS E RODRIGUES, 2020).

#### 5.1.1 Cadastro de usuário e autenticação

Para que o usuário possa ter acesso às funcionalidades da aplicação, é necessário que o mesmo se cadastre, para isso ele pode utilizar a tela de Cadastro de Usuário (Figura 5), a tela foi construída apenas contendo os dados necessários para o cadastro, de uma forma simples e rápida. Após a realização do cadastro com sucesso, o usuário é direcionado ao *Dashboard* (Figura 7) da aplicação e a partir da mesma terá acesso as demais funcionalidades do sistema.

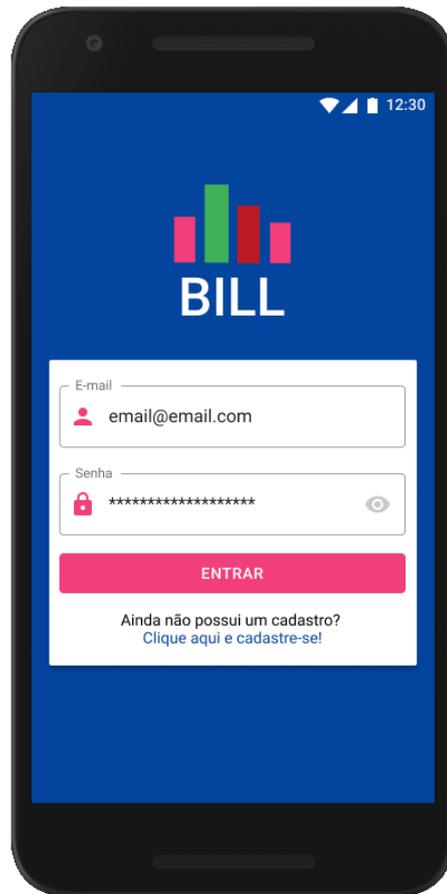
Em um segundo acesso do usuário, já cadastrado, ele poderá ser direcionado a tela de Autenticação (Figura 6), nela ele informará o e-mail e senha cadastrados anteriormente e, caso informados corretamente, será direcionado ao *dashboard* e terá acesso a todas as funcionalidades do produto.

Figura 5 - Cadastro de usuário



Fonte: Os autores.

Figura 6 - Autenticação



Fonte: Os autores.

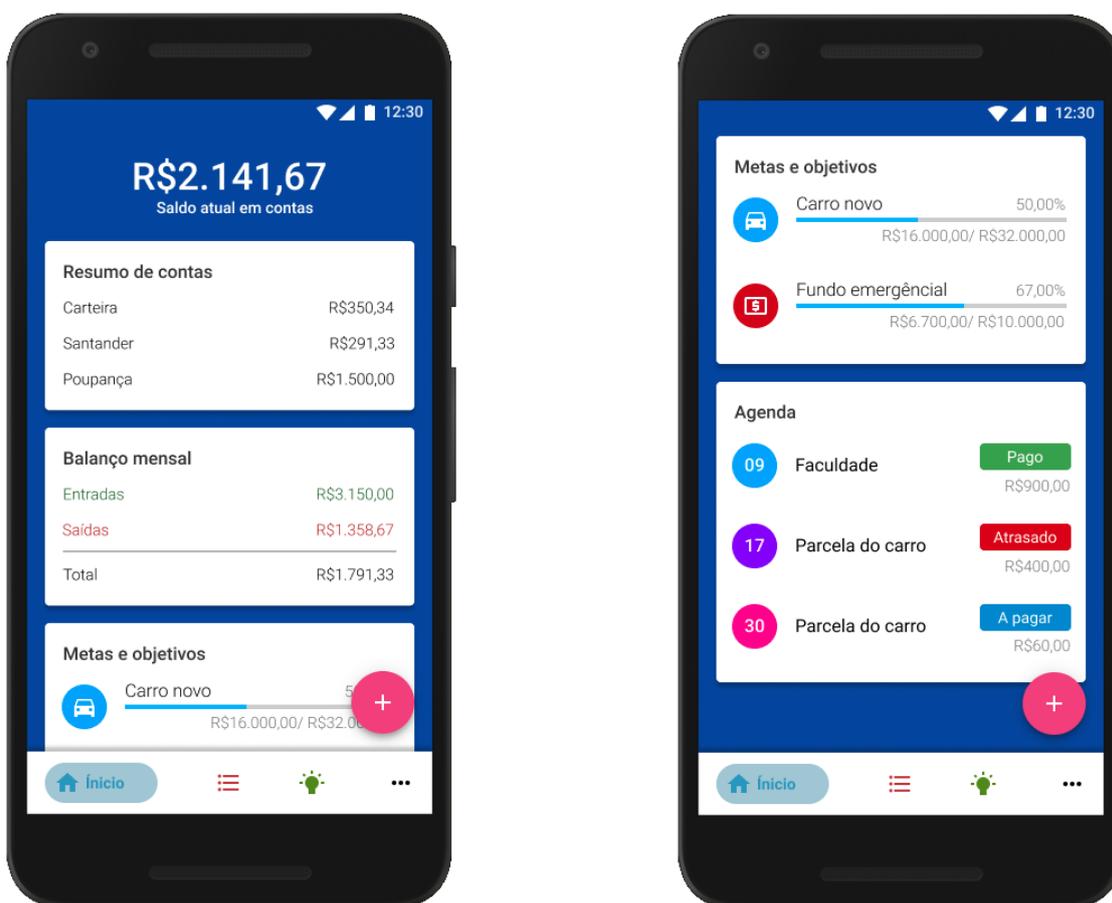
### 5.1.2 Dashboard

Após se autenticar, a primeira funcionalidade a qual o usuário terá acesso é o *Dashboard* (Figura 7). Nele, é possível visualizar informação sobre o saldo total de contas cadastradas e pré-selecionadas, o resumo de saldo em contas que compõe o saldo total exibido, uma seção que exibe o balanço do mês atual, composto por receitas e despesas e o balanço calculado.

Além das funcionalidades citadas, por meio do *dashboard* o usuário também terá acesso às metas e objetivos cadastrados, com dados sobre o progresso realizado em cada um dos itens e a agenda, onde são demonstrados os pagamentos ou recebimentos que estão chegando ao prazo cadastrado. Estas duas funcionalidades citadas não estão inclusas no *MVP* do projeto, mas serão implementadas em versões futuras.

A partir desta tela o usuário também pode acessar as demais funcionalidades do sistema através do menu presente na parte inferior da tela, nele são encontrados os atalhos para as telas de Listagem de Transações (Figura 8), Artigos (Figura 10), Opções (Figura 11), além do Cadastro de Transações (Figura 9), que pode ser acessado ao clicar no botão flutuante.

Figura 7 - Dashboard



Fonte: Os autores.

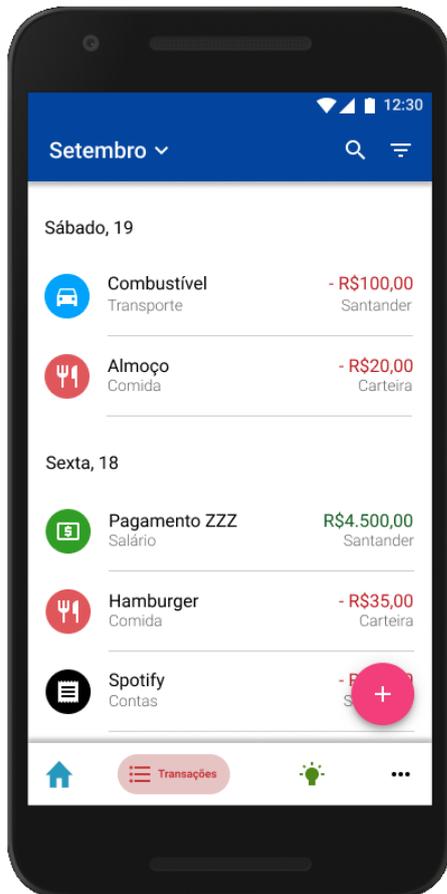
### 5.1.3 Listagem e cadastro de transações

Ao acessar a tela de Listagem de Transações (Figura 8) o usuário tem acesso a todas as transações (receitas, despesas e transferências) realizadas no mês selecionado, além de poder realizar filtragens com base nos meses. E ao clicar em uma das transações listadas o usuário é encaminhado para a tela de edição da transação, que se assemelha a de cadastro.

Para acesso a tela de Cadastro de Transações (Figura 9), o usuário deve clicar no botão flutuante e com isso aparecerá as opções para adicionar uma

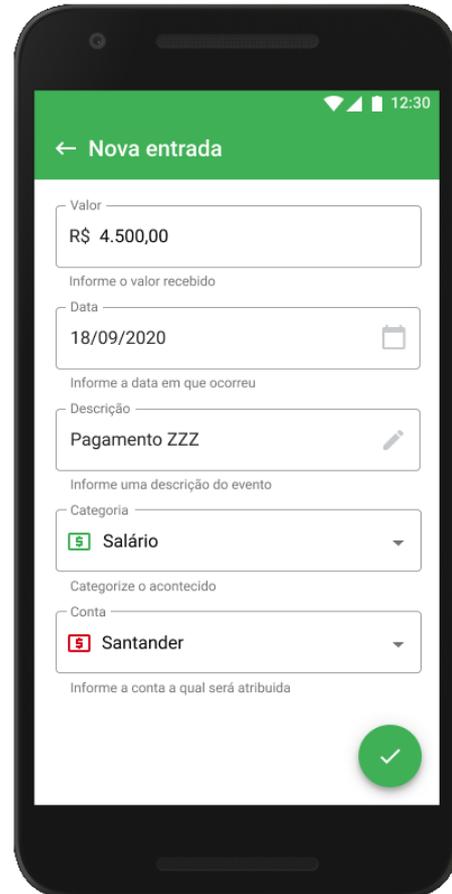
transação de saída, entrada ou uma transferência entre contas. Ao selecionar o tipo de transação desejado o usuário é encaminhado a tela de cadastro, onde será disposto a ele os campos como valor, data, descrição, categoria e a qual conta será atribuída a receita ou despesa.

Figura 8 - Listagem de Transações



Fonte: Os autores.

Figura 9 - Cadastro/Edição de Transações



Fonte: Os autores.

#### 5.1.4 Artigos

A proposta da tela de Artigos (Figura 10) é oferecer ao usuário conteúdos como forma de estudos, dicas e incentivo aos usuários para que possam aprimorar sua vida financeira de uma forma mais simples, porém eficaz, onde este conteúdo será disponibilizado por parceiros que criam conteúdos relacionados a finanças.

Figura 10 - Artigos

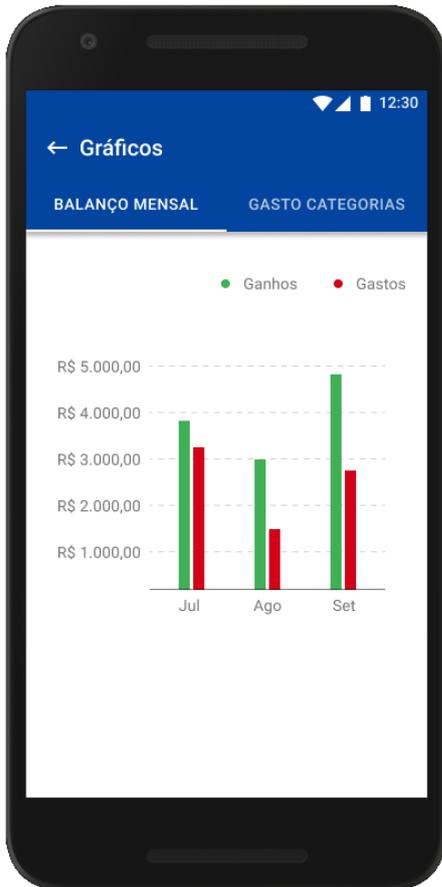


Fonte: Os autores.

### 5.1.5 Gráficos

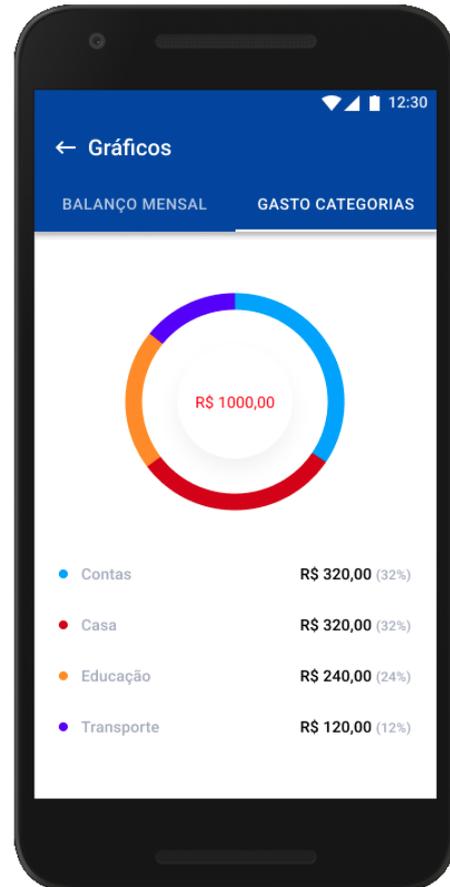
A plataforma disponibiliza ao usuário dois gráficos, mostrando diferentes visões da saúde de sua vida financeira. O primeiro deles é o gráfico de Balanço Mensal (Figura 11), que exhibe ao usuário o balanço mensal (Ganhos - Gastos) dos últimos 3 meses, para que ele possa visualizar como andam suas finanças de uma maneira sintética. Já o segundo, exhibe os gastos do mês atual de uma forma categorizada, na tela de Gasto por Categorias (Figura 12), separando e exibindo onde estão os maiores gastos, para que, assim o usuário possa se atentar em categorias específicas, a fim de reduzir os gastos e desta forma poder economizar e gerenciar melhor o seu dinheiro.

Figura 11 - Balanço Mensal



Fonte: Os autores.

Figura 12 - Gastos por Categoria



Fonte: Os autores

## 5.2 IMPLEMENTAÇÃO DO FRONT-END

Para o desenvolvimento da aplicação móvel foi utilizado o pacote de ferramentas de interface do usuário Flutter baseado na linguagem Dart.

A Figura 13 representa uma fração do código de inicialização da aplicação. Entre as linhas 76 e 85 estão as definições de estilos padrão da aplicação móvel, sendo definidas as cores primárias e secundárias, a fonte de texto padrão para botões e por fim na linha 86 é definido a tela inicial do aplicativo: a tela de Login.

O código-fonte do projeto está disponível na íntegra através do Github (ASSIS e RODRIGUES, 2020).

Figura 13 - Fragmento da implementação do aplicativo *mobile*

```
73     return runApp(  
74         MaterialApp(  
75             title: 'Bill',  
76             theme: ThemeData(  
77                 primaryColor: Colors.indigo,  
78                 accentColor: Colors.greenAccent,  
79                 cursorColor: Colors.greenAccent,  
80                 textTheme: TextTheme(  
81                     button: TextStyle(  
82                         fontFamily: 'OpenSans',  
83                     ) // TextStyle  
84                 ), // TextTheme  
85             ), // ThemeData  
86             home: Login(),  
87         ) // MaterialApp  
88     );
```

Fonte: Os autores.

### 5.3 IMPLEMENTAÇÃO DO BACK-END

Para aplicação do padrão REST (*Representational State Transfer*) foi utilizado o *framework Express.js*, para tornar o desenvolvimento mais simples e ágil, dar maior performance a aplicação e para expor as rotas que dão acesso aos dados e funcionalidades da aplicação aos usuários.

É apresentado na Figura 14 a implementação das rotas relacionadas às funcionalidades de busca, criação e atualização de contas na aplicação. Por exemplo, podemos analisar a busca de contas, na linha 23 sendo especificado o caminho *'/accounts'*, e logo em seguida, na linha 24, a definição do método *GET* e a função a ser executada *accounts.getAccounts*.

O código-fonte do projeto está disponível na íntegra através do Github (ASSIS e RODRIGUES, 2020).

Figura 14 - Fragmento código da implementação das rotas da aplicação

```
23     app.route('/accounts')  
24         .get(accounts.getAccounts)  
25         .post(accounts.createAccount);  
26     app.route('/accounts/:id')  
27         .get(accounts.getAccountByID)  
28         .put(accounts.updateAccount);
```

Fonte: Os autores.

Figura 15 - Comunicação entre o *back-end* e o banco de dados.

```
97  const res = await pool.query(  
98    `SELECT a.id,  
99          a.title,  
100         a.actual_balance AS "actualBalance",  
101         a.type_id        AS "typeID",  
102         act.title        AS "typeTitle",  
103         act.color,  
104         act.icon  
105         FROM accounts a  
106         INNER JOIN account_types act ON (act.id = a.type_id)  
107         WHERE a.user_id = $1 ${filters.join(" ")}`  
108         ORDER BY a.title`,  
109     [userID]  
110  );
```

Fonte: Os autores.

Na Figura 14 podemos observar as rotas cadastradas como a de cadastro de usuários, disponível no caminho */users* e utilizando o método *POST*, a rota para autenticação do usuário cadastrado, disponível no caminho */auth* e utilizando o método *POST* e as demais para disponibilização das funcionalidades de categorias e transações seguindo padrão de registro semelhante as rotas das funcionalidades de *accounts*, ilustrado pela Figura 14.

Para realizar a comunicação entre o *back-end* e o banco de dados para armazenamento e busca dos dados do usuário na aplicação foi utilizada a biblioteca *node-postgres* para *Node.js* de conexão com banco de dados *PostgreSQL*.

Pode ser analisado na Figura 15, um exemplo de implementação da comunicação entre a API e o banco de dados da aplicação. A figura ilustra a busca de Contas cadastradas pelo usuário, da linha 98 a 104 são os dados que serão retornados ao front-end da aplicação exibidos ao usuário, já na linha 106 é feita uma junção entre as tabelas *accounts* e *account\_types*, afim de trazer os dados referentes ao tipo da conta (Conta Corrente, Dinheiro, Poupança, Investimentos e Outros).

## 6 PESQUISA DE VALIDAÇÃO DO MVP

Após a conclusão do *MVP* (Produto Mínimo Viável), o mesmo foi disponibilizado a 34 usuários para que pudessem testar a aplicação e dar o seu

*feedback*, para isso todos os usuários tiveram acesso a um questionário com 6 afirmações e responderam a cada uma delas baseando-se no método da escala de *Likert*, com pontuação de 1 à 5 correspondendo a afirmações de “Discordo plenamente” à “Concordo plenamente”.

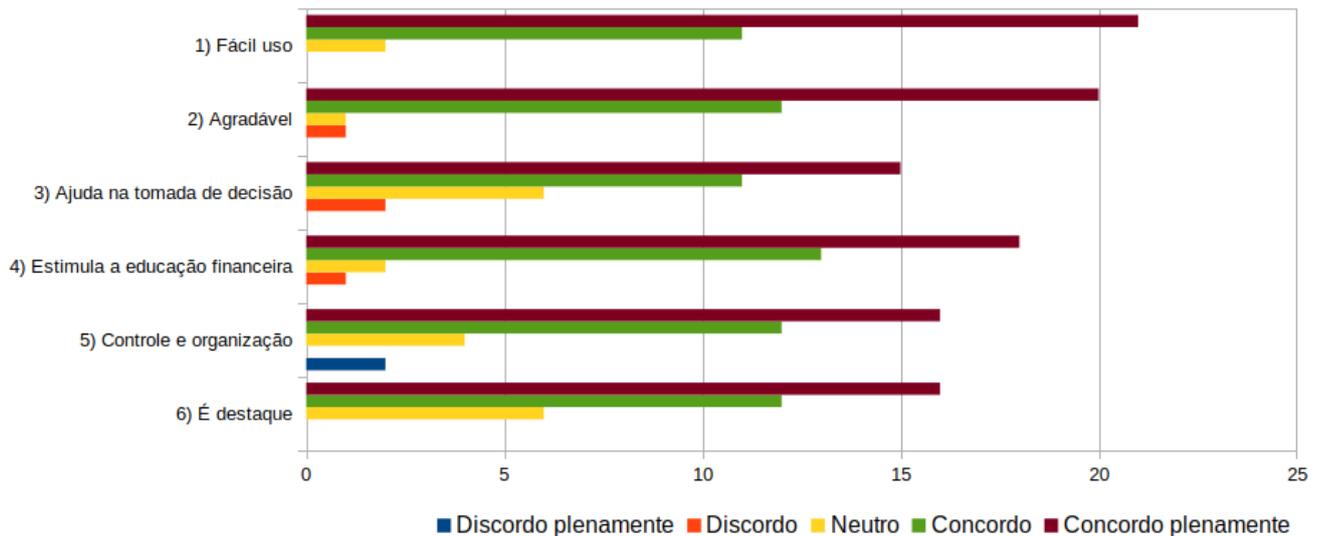
No geral o *feedback* recebido pelos usuários que tiveram a oportunidade de testar o *MVP* do projeto foi ótimo, indicando que o projeto é promissor e está no caminho certo. Podemos observar na Figura 18 um gráfico compilado a partir dos números obtidos no questionário para conseguirmos uma visão melhor das experiências e primeiras impressões que os usuários tiveram.

Analisando o gráfico é factível afirmarmos que os pontos de usabilidade, experiência do usuário e interface foram muito bem aprovados pelos usuários, ou seja, 94,2% dos usuários concordaram que o aplicativo é de fácil usabilidade e, 94,1%, reportaram que o aplicativo é agradável, de forma geral.

As ferramentas referentes ao controle, organização, tomada de decisões e estímulos a educação financeira, fornecidas aos usuários pela plataforma, tiveram um alto nível de aprovação por parte dos usuários, obtendo 76,5% de aprovação sobre a ajuda na tomada de decisões, 91,1% apontando que o aplicativo estimula a educação financeira e 82,4%, dos usuários, concordaram que o aplicativo contribui para o controle e organização das finanças pessoais.

Por fim, a última afirmação tinha como objetivo comparar o aplicativo Bill com outros aplicativos de mesmo seguimento, a fim de validar se o mesmo se destaca perante os outros com a sua simplicidade e eficiência, e o *feedback* para esse ponto foi satisfatório, obtendo 82,4% de aprovação por parte dos usuários.

Figura 16 - Respostas da validação do MVP



Fonte: Os autores.

## 7 CONCLUSÃO

Com a finalização do desenvolvimento do produto mínimo viável da aplicação móvel para as plataformas Android e iOS do projeto Bill, o propósito deste trabalho foi alcançado. No decorrer deste desenvolvimento foram analisadas melhorias e novas funcionalidades que podem agregar ao projeto final a fim de desenvolver um produto mais completo e com mais opções que irão tornar a experiência do usuário mais satisfatória e robusta. Melhorias nas funcionalidades já existentes no *MVP* como a finalização da função de Metas e Objetivos que vai possibilitar que o usuário tenha o acompanhamento de forma completa de seus objetivos financeiros, e a funcionalidade de Agenda, onde o usuário poderá realizar o controle de forma eficiente de suas contas a pagar ou recebimentos, onde a plataforma ficará responsável por notificar o usuário quando chegar a data programada.

Além das melhorias, foram levantadas novas funcionalidades a serem implementadas em versões futuras, como a implementação de mais gráficos com diferentes visões e análises do financeiro dos usuários, possibilidade de controle das movimentações de cartões de crédito, controle de carteiras de investimentos entre outras ideias que podem melhorar a usabilidade e experiência do usuário com o produto.

## REFERÊNCIAS

ASEF, Mazen. **Simples, mas complexo! O que é e como funciona o Node.js.** TecMundo, 2019. Disponível em: <<https://www.tecmundo.com.br/mercado/137805-o-node-js.htm>>. Acesso em: 11 abr. 2020.

ASSIS, Guilherme Henrique; RODRIGUES, João Vitor. **Repositório contendo documentação, código-fonte e pesquisas realizadas para o projeto.** Disponível em: <<https://github.com/guilhermehrq/bill>>. Acesso em: 15 set. 2020

DRUCKER, P. F. **Innovation and Entrepreneurship Practice and Principles.** 2ª. ed. Abingdon: London and New York, 2015.

EVER FLOW. **Controle financeiro: a importância da tecnologia para as empresas.** 2017. Disponível em: <<https://everflow.com.br/controle-financeiro-a-importancia-da-tecnologia-para-as-empresas/>>. Acesso em: 12 abr. 2020.

FREITAS, Romualdo Rubens de. **Análise e Projeto de Software.** 1. ed. Cuiabá, MT, 2015.

HENRIQUE, Paulo. Medium. **Conheça o Flutter, a aposta da Google para a criação de apps nativos multiplataforma.** 2018. Disponível em: <<https://medium.com/liferay-engineering-brazil/conhe%C3%A7a-o-flutter-a-aposta-da-google-para-a-cria%C3%A7%C3%A3o-de-apps-nativos-multiplataforma-e59c610134d8>>. Acesso em: 11 abr. 2020.

INVESTIMENTOS, Equipe Toro. Toro Blog. **Controle financeiro pessoal - 13 dicas para controlar suas finanças.** 2018. Disponível em: <<https://blog.toroinvestimentos.com.br/controle-financeiro-pessoal>>. Acesso em: 11 abr. 2020.

LUCIDCHART. Artigo: **O que é BPMN?**, S.D. Disponível em: <<https://www.lucidchart.com/pages/pt/o-que-e-bpmn>>. Acesso em: 10 mai. 2020.

MATTHEW, Neil; STONES, Richard. **Beginning Databases with PostgreSQL From Novice to Professional**. 2. ed. Apress, 2005.

MOREIRA, D. Exame. **O que é uma startup?**. 2016. Disponível em: <<https://exame.abril.com.br/pme/o-que-e-uma-startup/>>. Acesso em: 11 abr. 2020.

NODEBR. **O QUE É NODEJS?**. 2016. Disponível em: <<http://nodebr.com/o-que-e-node-js/>>. Acesso em 11 abr. 2020.

POSTGRESQL. **PostgreSQL: About**. [S.D]. Disponível em: <<https://www.postgresql.org/about/>>. Acesso em: 12 abr. 2020.

RED HAT. **O QUE É UMA API?**. [S.D]. Disponível em: <<https://www.redhat.com/pt-br/topics/api/what-are-application-programming-interfaces>>. Acesso em 11 abr. 2020.

RODRIGUES, Joel. **Modelo Entidade Relacionamento (MER)**. Disponível em: <<https://www.devmedia.com.br/modelo-entidade-relacionamento-mer-e-diagrama-entidade-relacionamento-der/14332>>. Acesso em: 10 mai. 2020.

SOLDI, Dimas. **Apenas 25% dos jovens de 18 a 30 anos fazem controle financeiro**. 2019. Disponível em: <<https://agenciabrasil.ebc.com.br/economia/noticia/2019-10/apenas-25-dos-jovens-de-18-30-anos-fazem-controle-financeiro>> Acesso em: 14 mai. 2020.

SOMMERVILLE, Ian. **Engenharia de Software**. 9ª ed. São Paulo: Pearson Prentice Hall, 2011.

S.O.S. **Como a tecnologia pode ajudar na organização financeira**. 2019. Disponível em: <<https://www.sos.com.br/noticias/tecnologia/como-a-tecnologia-pode-ajudar-na-organizacao-financeira>>. Acesso em: 12 abr. 2020.

TAINO, Denise. **Controle financeiro e tecnologia: 7 vantagens para sua empresa**. Blog da Soften, 2018. Disponível em: <<https://blog.softensistemas.com.br/controle-financeiro-e-tecnologia-7-vantagens-para-sua-empresa/>>. Acesso em: 12 abr. 2020.

TEIXEIRA, Pedro. **Professional Node. js: Building Javascript based scalable software.** 1. ed. John Wiley & Sons, 2012.

ZAMMETTI, Frank. **Practical Flutter Improve your Mobile Development with Google's Latest Open-Source SDK.** 1. ed. Pottstown, PA, USA: Apress, v. 1, 2019.