

Money Tracker

Relatório Técnico de Projeto de Software

(versão 0.1)

Sumário

1	Introdução	3
1.1	Stakeholders	3
1.2	Objetivo Geral do Sistema	3
1.3	Metodologia	4
2	Desenvolvimento do Sistema	5
2.1	Etapa de Imersão	5
2.2	Etapa de análise	7
2.2.1	Lista de Requisitos x Funcionalidades	7
2.2.2	Lista de Requisitos Não-Funcionais	8
2.2.3	Diagrama de Caso de Uso	8
2.2.4	Diagrama de Classe	9
2.3	Etapa de Ideação	9
2.3.1	Backend	9
2.3.2	Frontend	10
2.3.3	Backlog	11
2.4	Etapa de Prototipação	16

1 Introdução

O sistema de software a ser desenvolvido tem como principal base o cliente poder ter a dimensão dos gastos que estão sendo realizados, assim como, também, como todo o dinheiro que está entrando no financeiro pessoal ou familiar.

1.1 Stakeholders

Cliente: O sistema foi desenvolvido focando, principalmente, em pessoas que gostariam de ter uma plataforma dedicada ao controle financeiro, seja ele pessoal ou familiar, no quesito empresarial não seria o melhor uso da ferramenta.

Colaborador: Os colaboradores do sistema foram de extrema importância, visto que todas as entrevistas realizadas foram de muita valia e, com certeza, adicionaram muito conteúdo para o desenvolvimento da plataforma.

1.2 Objetivo Geral do Sistema

O objetivo do sistema consiste em apresentar aplicação web capaz de gerenciar, com auxílio do cliente, toda o financeiro mostrando todo o lançamento dos meses posteriores, como também dos anteriores. para o usuário uma visão geral do de todo gasto e renda realizados. Ainda, ao utilizar a aplicação será possível realizar:

- Definir metas e gastos
 - Em outras palavras será possível criar *caixas* onde será possível separar dinheiro para investimentos, presentes...
- Definir gastos/renda recorrente
- Definir reserva de emergência
- Aviso semanal ou diário para fazer o gerenciamento das contas
- Dicas para evitar gastos

Por fim, o usuário será capaz de realizar todas as ferramentas citadas, mas

todos os dados deverão ser adicionados pelo usuário.

1.3 Metodologia

O projeto de gerenciamento financeiro adotou uma abordagem metodológica baseada no design thinking para a coleta de dados. Este processo incluiu extensas pesquisas online, entrevistas detalhadas com possíveis usuários da plataforma e uma análise minuciosa de aplicativos similares já existentes. A análise de dados concentrou-se principalmente nas informações obtidas durante as entrevistas, as quais foram cruciais para a definição e desenvolvimento do projeto.

O desenvolvimento da plataforma escolheu a interface web como principal ponto de interação, armazenando todas as informações essenciais em um banco de dados *MariaDB*. A implementação técnica envolveu o uso de *React.js* (*React*) para o frontend, *Node.js* (*Node*) para o backend, e *Prisma.js* para a modelagem.

A validação do desenvolvimento foi conduzida pelos próprios entrevistados, que desempenharam um papel fundamental ao fornecer feedback contínuo. Modificações e ajustes foram implementados conforme necessário, sem restrições temporais rigorosas.

Todo o ciclo de desenvolvimento, desde a concepção até a implementação, foi realizado exclusivamente pelo autor do projeto. Os recursos utilizados incluíram um notebook, um gravador e ferramentas de modelagem de diagramas como o draw.io.

Em relação às limitações enfrentadas, o projeto adotou uma estratégia de mitigação cortando funcionalidades consideradas não essenciais para o funcionamento fundamental do software.

2 Desenvolvimento do Sistema

2.1 Etapa de Imersão

Na etapa de imersão deu-se início na extração de características relevantes ao projeto. Para extração do SWOT, foi realizado 3 entrevistas com pessoas diferentes perspectivas de gerenciamento de gastos. As perguntas realizadas para extração de características tiveram um cunho voltado as ferramentas utilizadas no gerenciamento, tempo gasto, frequência, dificuldades e eventuais problemas, se acontecem. Ao final das entrevistas foi ferado uma lista de requisitos.

Artefatos gerados na Etapa de Imersão

Requisito	Tipo	Regra de Negócio
CRUD de receita	F	CRUD com todas as informações pertinentes a entrada de dinheiro
CRUD de gastos	F	CRUD com todas as informações pertinentes a saída de dinheiro
Mostrar perspectiva anual	NF	O sistema deve mostrar todos os gastos do ano
Mostrar perspectiva mensal	NF	O sistema deve mostrar todos os gastos do mês
CRUD de notificação	F	CRUD com todas as informações pertinentes a notificação de gerenciamento da conta
CRUD de anotação	F	CRUD com todas as informações pertinentes a anotação da uma gasto ou receita
Definir gastos recorrente	NF	O sistema deve permitir o usuário definir gastos recorrentes ao criar ou modificar algum gasto

Definir receita recorrentes	NF	O sistema deve permitir o usuário definir receita recorrentes ao criar ou modificar alguma receita
CRUD de meta de gastos	F	CRUD com todas as informações pertinentes a meta de gastos
Emitir notificação sempre que ultrapassar limite de gasto definidor por alguma meta	F	O sistema deve notificar o usuário por email ou dentro da aplicação sempre que algum gasto ultrapassar algum limite auto imposto
Desenvolvimento do frontend	NF	O sistema deverá ser realizado utilizando <i>typescript</i> , <i>react</i> .
Desenvolvimento do backend	NF	O sistema deverá ser realizado utilizando <i>node.js</i>
Desenvolvimento do banco de dados	NF	O sistema deverá ser realizado utilizando <i>node.js</i>
O controle financeiro familiar pode ser alterado por usuários permitidos	F	O sistema deve permitir usuários com permissão a fazer lançamentos no gerenciamento financeiro familiar
O controle financeiro familiar pode ser alterado por usuários permitidos	F	O sistema deve permitir usuários com permissão a fazer lançamentos no gerenciamento financeiro familiar
O controle financeiro pode ser compartilhado com outros usuários	F	O sistema deve permitir usuários compartilhar o controle financeiro com outros usuários

Legenda do Tipo:

F – Funcionais

NF – Funcionais

2.2 Etapa de análise

2.2.1 Lista de Requisitos x Funcionalidades

id. Req	Requisito	Ator	Funcionalidade	Regra de negócio
1	CRUD de receitas	Usuário	Criar nova receita	O sistema deve permitir o usuário criar novas receitas
2	CRUD de receitas	Usuário	Ler receitas	O sistema deve permitir o usuário ler receitas criadas
3	CRUD de receitas	Usuário	Atualizar receitas criadas	O sistema deve permitir o usuário atualizar receitas criadas
4	CRUD de receitas	Usuário	Deletar receitas criadas	O sistema deve permitir o usuário deletar receitas criadas
5	CRUD de despesas	Usuário	Criar nova despesa	O sistema deve permitir o usuário criar novas despesas
6	CRUD de despesas	Usuário	Ler despesa	O sistema deve permitir o usuário ler despesas criadas
7	CRUD de despesas	Usuário	Atualizar despesas criadas	O sistema deve permitir o usuário atualizar despesas criadas

8	CRUD de despesas	Usuário	Deletar despesas criadas	O sistema deve permitir o usuário deletar despesas criadas
9	Definir despesa recorrente	Usuário	Definir a recorrência da despesa	O sistema deve permitir o usuário definir a recorrência de uma despesa
10	Definir receita recorrente	Usuário	Definir a recorrência da receita	O sistema deve permitir o usuário definir a recorrência de uma receita

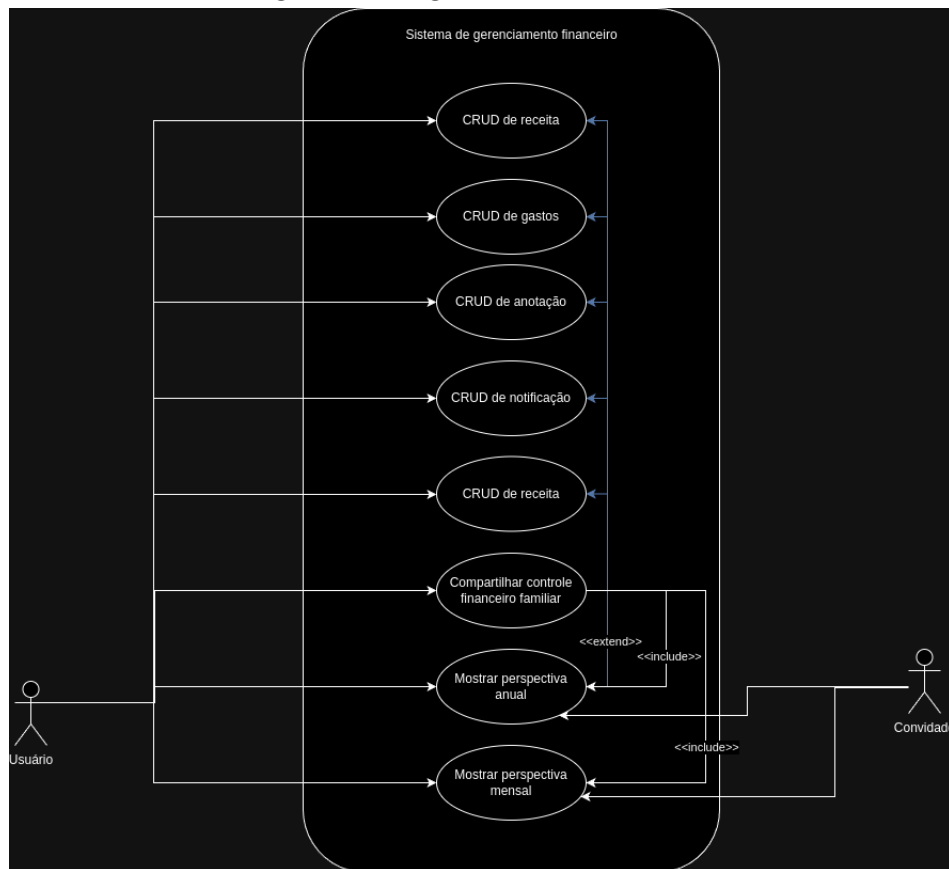
2.2.2 Lista de Requisitos Não-Funcionais

id. Req	Requisito	Categoria
1	Desenvolvimento do banco de dados deverá ser realizado em node.js	Performance
2	Desenvolvimento do backend deverá ser realizado em node.js	Performance
3	Desenvolvimento da interface de usuário deverá ser realizado utilizando react	Performance
4	O compartilhamento do gerenciamento financeiro deve ser feito apenas com usuários especificados	Segurança

2.2.3 Diagrama de Caso de Uso

O diagrama de caso de uso foi realizado utilizando a o site *draw.io*, o resultado pode ser observado na figura 1.

Figura 1: Diagrama de caso de uso



2.2.4 Diagrama de Classe

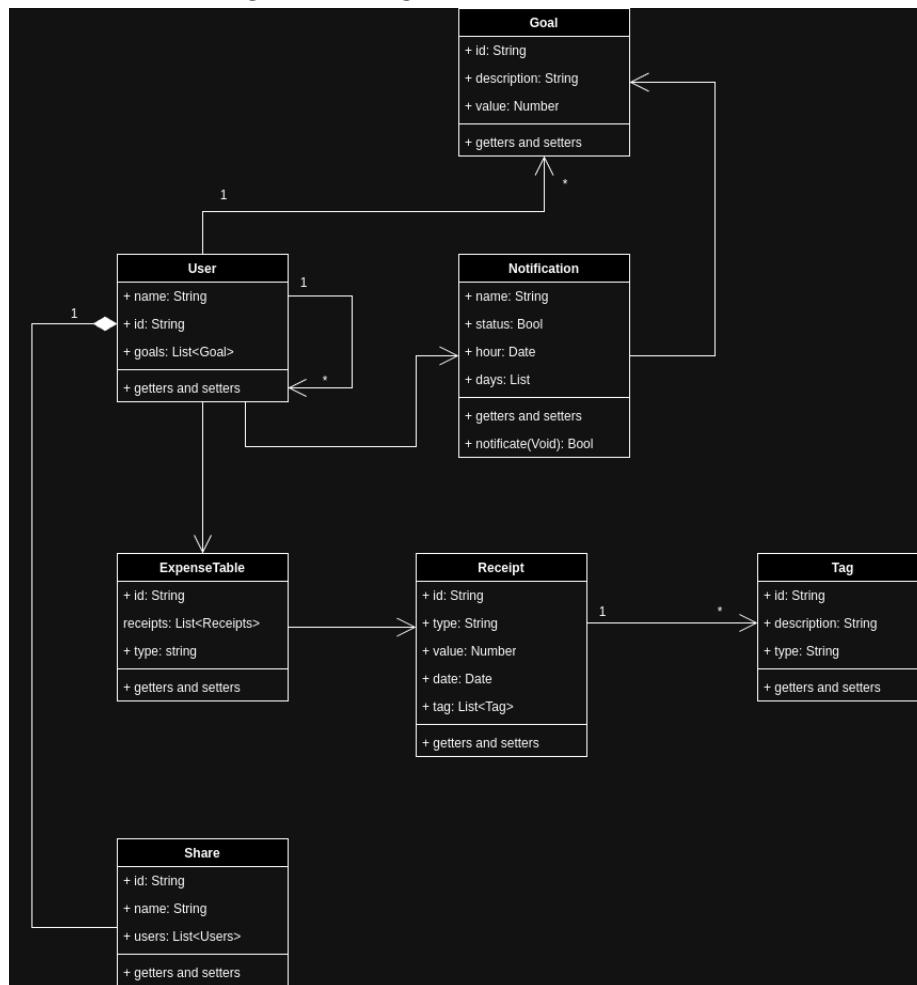
O diagrama de classes foi realizado utilizando a o site *draw.io*, o resultado pode ser observado na figura 2.

2.3 Etapa de Ideação

2.3.1 Backend

A modelagem do banco de dados sempre será atualizado conforme a necessidade de implementação e conforme forem surgindo as dificuldades de implementação.

Figura 2: Diagrama de caso de classe



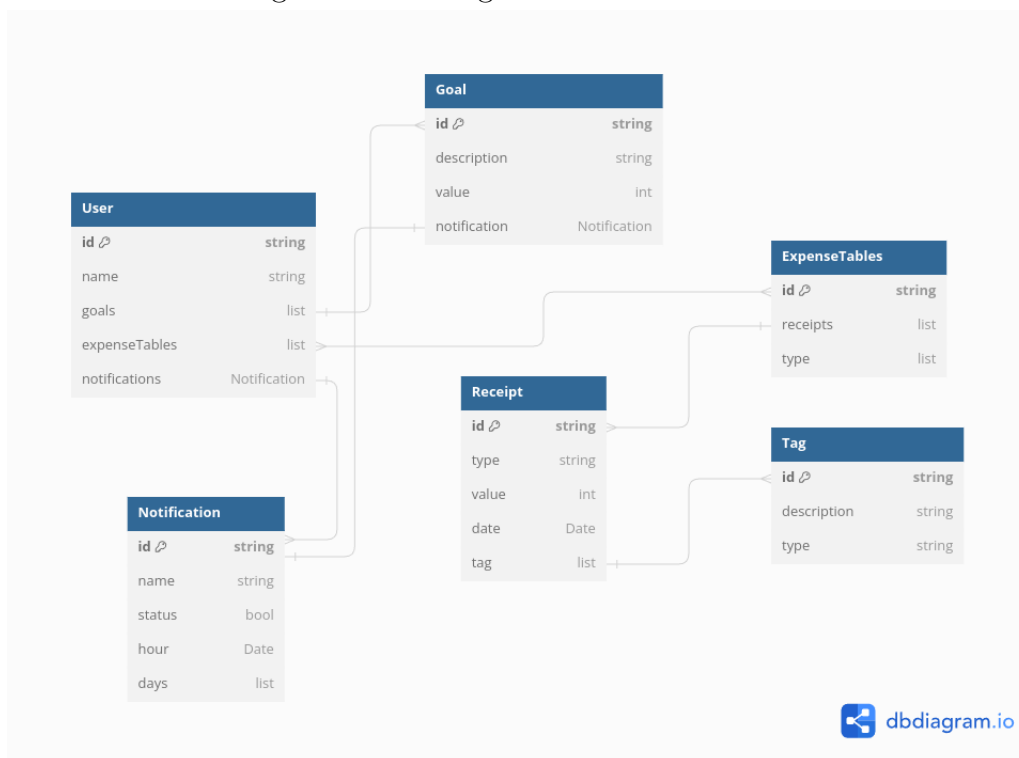
2.3.2 Frontend

O frontend utilizado foi a partir de um modelo pronto desenvolvido para mobile que será adaptado para uma interface web, visto que um dos requisitos das pessoas entrevistadas era ter uma visão completa de todos os meses, e para isso seria muito melhor ser mostrado dentro de uma tela de computador.

Portanto, todo o design do frontend foi baseado no *Montra*, uma design desenvolvido, especificamente, para mobile.

OBS: O design pode ser observado a partir da figura 3.

Figura 3: Modelagem do banco de dados



2.3.3 Backlog

Id	História de Usuário	Regras de Negócio	CrITÉrios de aceite	Prioridade
1	Eu como usuário quero adicionar uma nova receita no gerenciamento	O usuário poderá criar quantas receitas achar necessárias para seu gerenciamento	<ol style="list-style-type: none"> 1. O usuário deverá criar uma receita por vez 2. O usuário pode criar quantas receitas forem necessárias. 	Alta
2	Eu como usuário quero modificar uma receita existente no gerenciamento	O usuário poderá editar quantas receitas achar necessárias para seu gerenciamento	<ol style="list-style-type: none"> 1. O usuário deverá modificar uma receita por vez 2. O usuário pode modificar quantas receitas forem necessárias. 	Alta

3	Eu como usuário quero deletar uma receita no gerenciamento	O usuário poderá deletar quantas receitas achar necessárias para seu gerenciamento	<ol style="list-style-type: none"> 1. O usuário deverá deletar uma receita por vez 2. O usuário pode deletar quantas receitas forem necessárias. 	Alta
4	Eu como usuário quero deletar uma despesa no gerenciamento	O usuário poderá deletar quantas despesas achar necessárias para seu gerenciamento	<ol style="list-style-type: none"> 1. O usuário deverá deletar uma despesa por vez 2. O usuário pode deletar quantas despesas forem necessárias. 	Alta

5	Eu como usuário quero criar uma despesa no gerenciamento	O usuário poderá criar quantas despesas achar necessárias para seu gerenciamento	<ol style="list-style-type: none"> 1. O usuário deverá criar uma despesa por vez 2. O usuário pode criar quantas despesas forem necessárias. 	Alta
6	Eu como usuário quero modificar uma despesa no gerenciamento	O usuário poderá modificar quantas despesas achar necessárias para seu gerenciamento	<ol style="list-style-type: none"> 1. O usuário deverá modificar uma despesa por vez 2. O usuário pode modificar quantas despesas forem necessárias. 	Alta

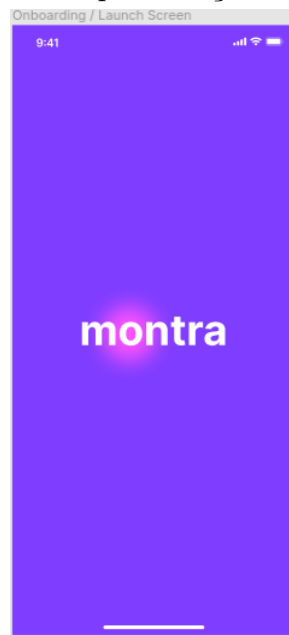
7	Eu como usuário quero visualizar o ano inteiro com todos os gastos e receitas em seus respectivos meses.	O usuário poderá ter a visão anual de gastos e receitas em cada mês.	<ol style="list-style-type: none"> 1. O usuário deverá visualizar todos os meses no ano. 2. O usuário deverá visualizar em cada mês deverá constar cada despesa ou receita respectiva a cada mês. 	Alta
8	Eu como usuário quero visualizar o mês o com todos os gastos e receitas.	O usuário poderá ter a visão mensal de gastos e receitas em cada mês.	<ol style="list-style-type: none"> 1. O usuário deverá visualizar de um mês do ano. 2. O usuário deverá visualizar no mês deverá constar cada despesa ou receita respectiva a cada mês. 	Alta

2.4 Etapa de Prototipação

O protótipo da interface do usuário será desenvolvida utilizando como base o modelo utilizando no *Montra*, um projeto com design free, ou seja, que pode ser utilizado por qualquer pessoa.

Todo o protótipo gerado pode ser conferido com mais detalhes no figma, <https://www.figma.com/file/ONcoN1mFA371DaaP44y0C1/Montra---Expense-Tracker-UI-Kit-?type=design&node-id=223-1&mode=design&t=xuHI5oFSQPix7ojR-0>.

Figura 4: Apresentação do App



As figuras mostradas foram apenas uma porção, *sneakpeak* do que foi desenvolvido, o material completo pode ser encontrado no site do figma

Figura 5: Página Inicial

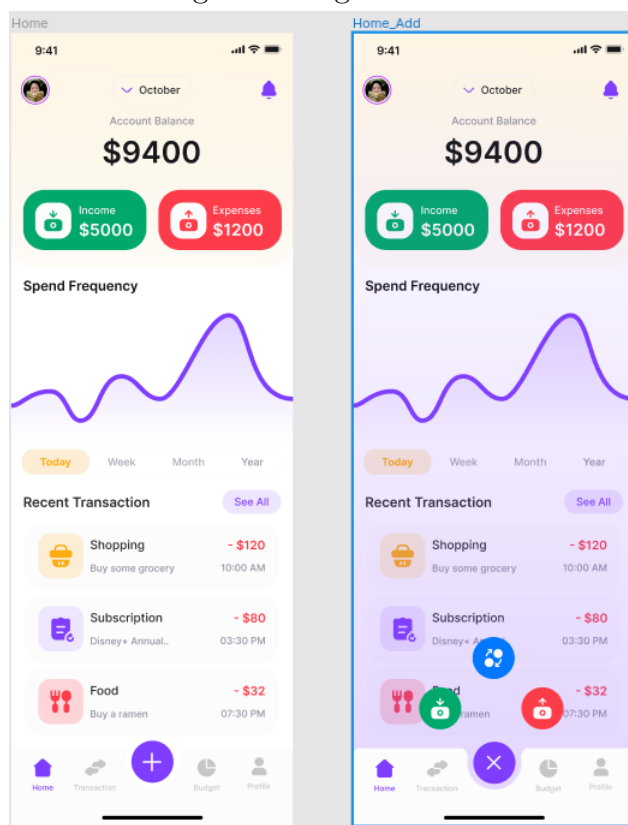


Figura 6: Adicionar despesa/receita

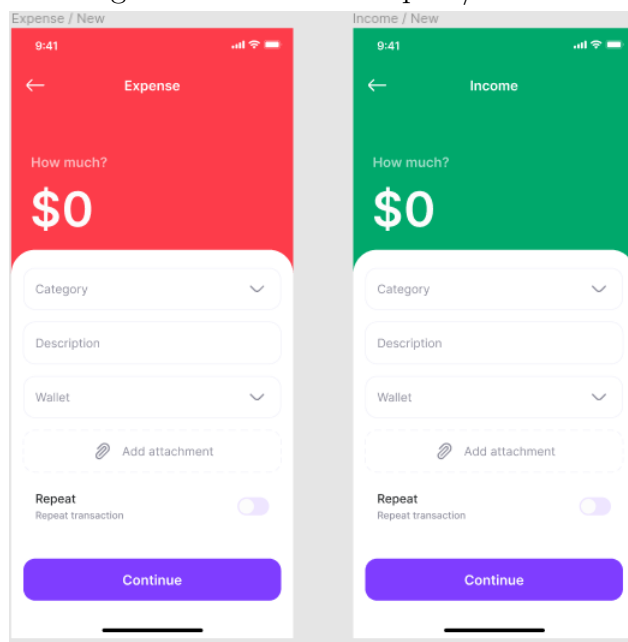


Figura 7: Notificações

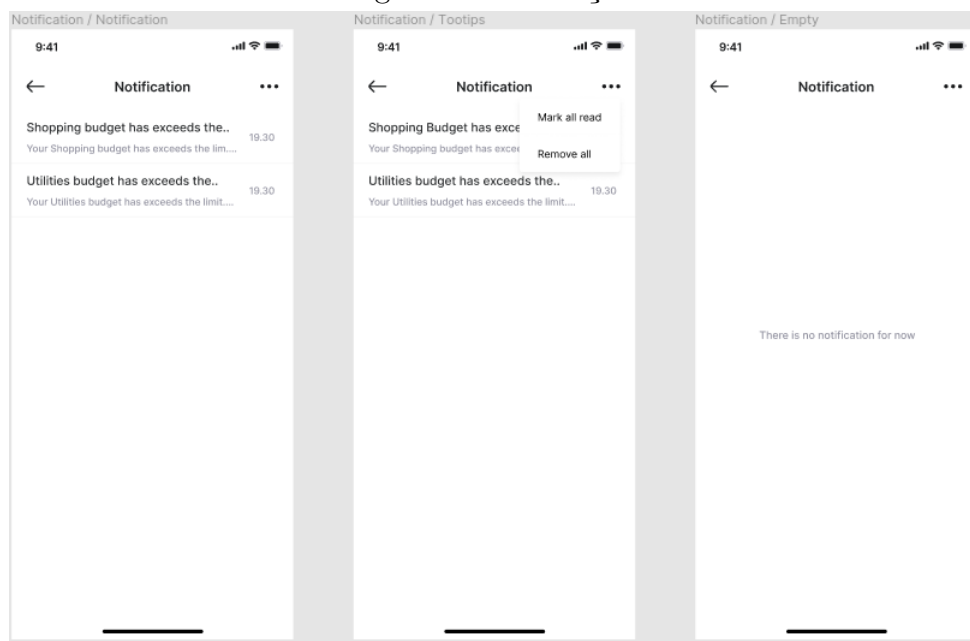


Figura 8: Apresentação da tabela mensal

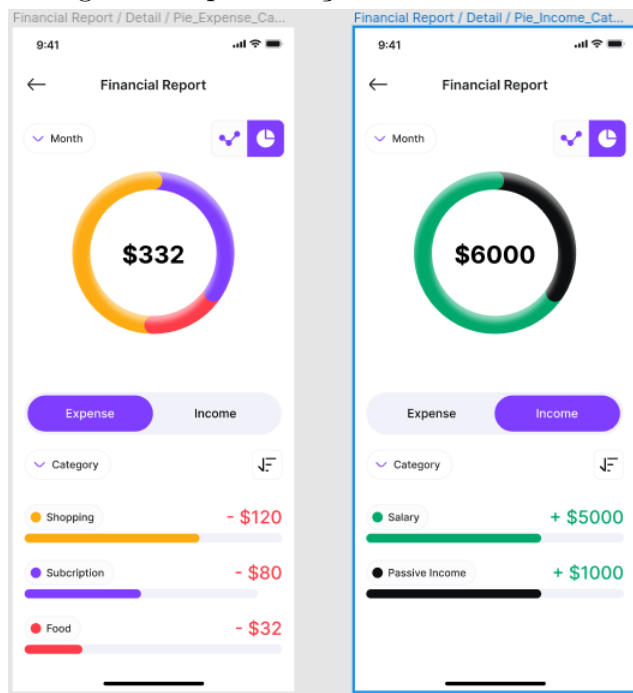


Figura 9: Gráficos

