

## 1. Saber o que Pedir é a Chave! Otimize seus Prompts!

Prompt desenvolvido para o Copilot aprimorar o PRD, neste também foi anexado o PRP proposto:

“Gostaria que melhorasse o PRD (Product Requirements Document) abaixo deixando mais simples, claro e que tornasse o aplicativo financeiro acessível ao maior número de pessoas (inclusive idosos e pessoas com necessidades especiais), seja funcional em todos os equipamentos desde celulares até os notebooks e tenha um design universal, ou seja, permita boa experiência pelo máximo de usuários possíveis. Este PRD será usado no lovable para aplicação de vibe coding, assim existe a necessidade do formato snippet txt, sem a necessidade de emojis e com a formatação markdown simples na resposta.”

Esse foi o o PRD (Product Requirements Document) criado pelo copilot:

### # Contexto

Criar um aplicativo de Organização de Finanças Pessoais baseado em conversas simples e naturais.

O objetivo é permitir que qualquer pessoa, incluindo idosos e pessoas com necessidades especiais, consiga controlar suas finanças sem depender de formulários complexos ou planilhas.

O app deve funcionar em diferentes dispositivos (celulares, tablets e notebooks) e seguir princípios de design universal para garantir acessibilidade e boa experiência para o maior número de usuários.

### # Problema

Muitos usuários abandonam aplicativos financeiros porque exigem entradas manuais extensas e oferecem pouca personalização.

A proposta é resolver isso com uma experiência conversacional, acessível e inclusiva, que ofereça recomendações automáticas de economia e adaptação às necessidades individuais.

### # Público-Alvo

- Pessoas que desejam iniciar o controle financeiro de forma prática e sem complicação.
- Usuários com pouca familiaridade com tecnologia.
- Idosos e pessoas com necessidades especiais que precisam de acessibilidade e clareza na interface.

#### # Funcionalidades-Chave

1. Registrar gastos por meio de chat em linguagem natural.
2. Classificação automática das transações.
3. Definição e acompanhamento de metas financeiras.
4. Recomendações de economia fornecidas pelo “Agente Financeiro”.
5. Relatórios simples, claros e personalizados.
6. Interface acessível com suporte a leitores de tela, contraste adequado e navegação intuitiva.
7. Compatibilidade com diferentes dispositivos e tamanhos de tela.

#### # Entregável da IA

Gerar um plano de MVP que inclua:

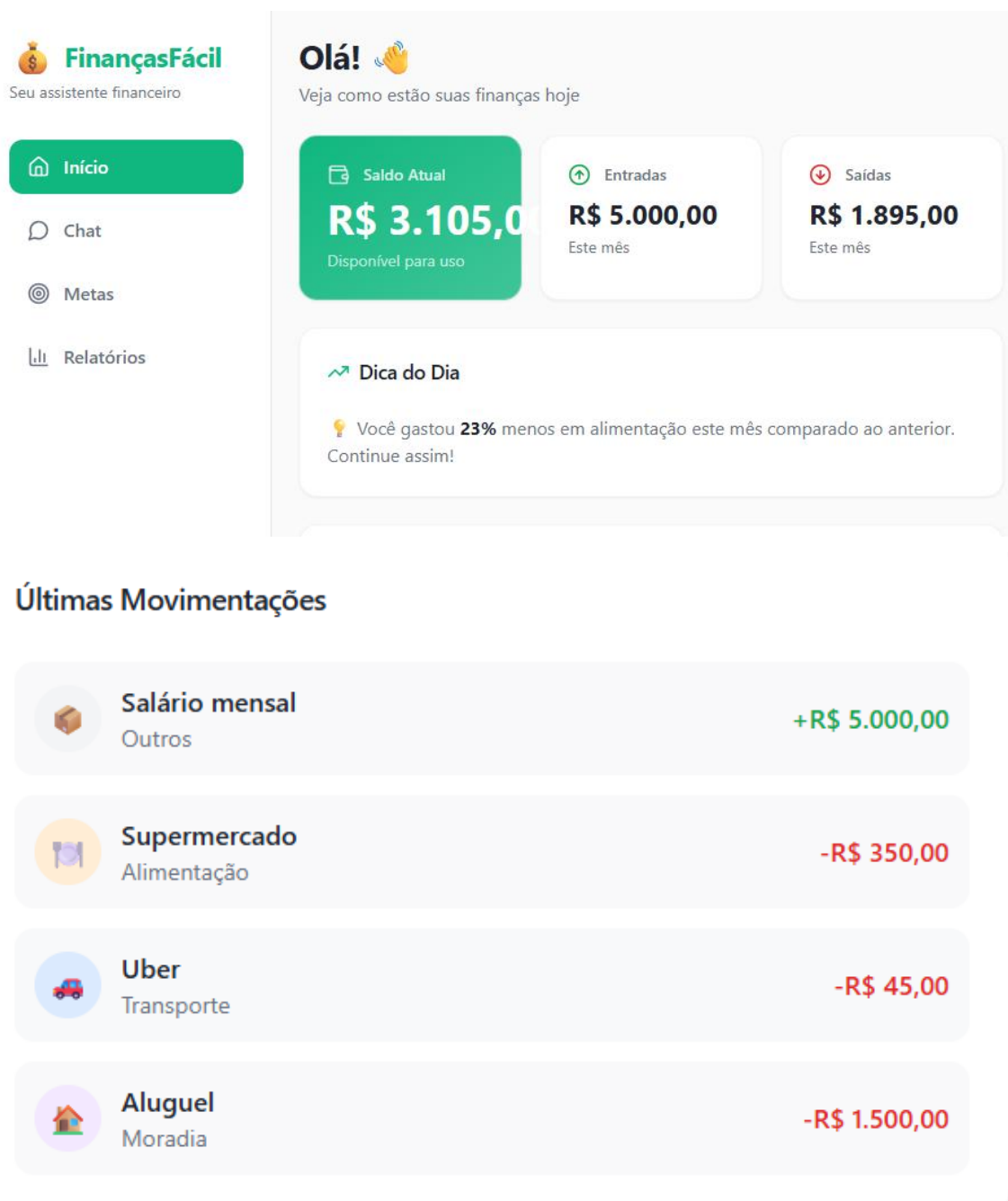
- Principais telas (chat, metas, relatórios).
- Recursos necessários para acessibilidade e design universal.
- Esboço de validação inicial com usuários diversos.

O conteúdo deve ser apresentado em tom educativo e linguagem acessível, em português.

## 2. Explorando o Lovable na Prática

Utilizando o PRD (Product Requirements Document) desenvolvido anteriormente e com o seguinte prompt: “crie app de finanças pessoais utilizando o seguinte PRD:”

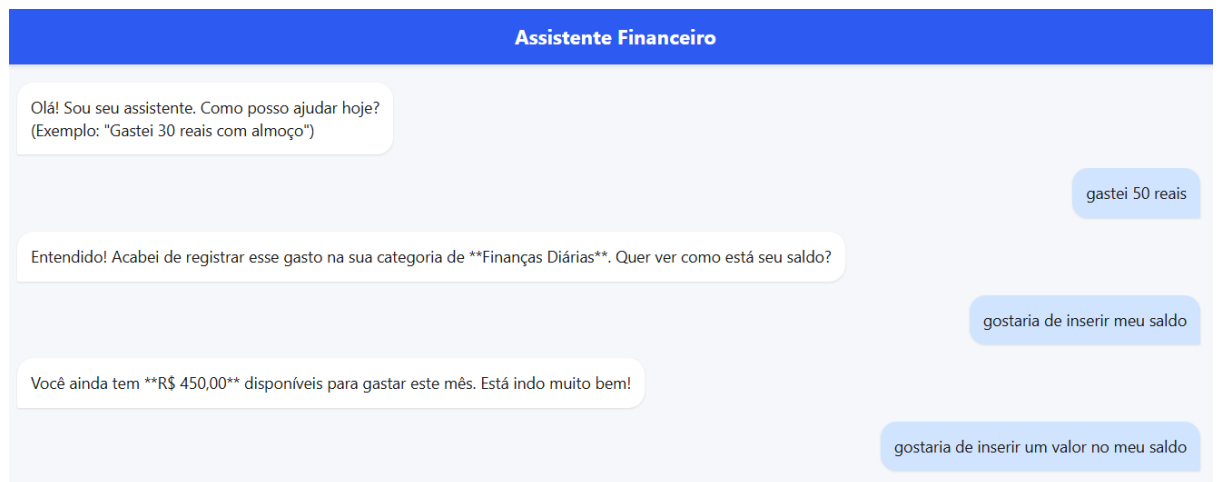
Assim, foi criado o aplicativo, cuja as imagens estão abaixo:



Observa-se que o aplicativo não permite adicionar a renda e também adicionar o usuário, por isso foi descartado e optou-se por utilizar a inteligência artificial e o aplicativo do vscode para testar as funcionalidades.

Com o mesmo PRD (Product Requirements Document) e com o seguinte prompts: “gostaria que criasse um aplicativo de finanças utilizando o seguinte PRD (Product Requirements Document) “ e outro “Com base na conversa anterior, gostaria que fornecesse o arquivo html, css e javascript deste, formando

um aplicativo estático apenas para prototipagem”, foi gerado um arquivo index para ser testado no vscode e obtive o seguinte aplicativo:



Como demonstrado na imagem acima, o problema continua o mesmo. Não é possível inserir valores. E também não é possível o aplicativo ser pessoal.

Sendo assim, enviou-se outro prompt : “torne a prototipagem um pouco mais pessoal para cada usuário permitindo que este insira seu própria renda e seus próprios gastos”. E foi gerado o seguinte aplicativo, como mostra a imagem abaixo:



A imagem acima, mostra como o aplicativo evoluiu, tornando-se possível inserir o saldo e os gastos durante o dia. Porém, esses dados são temporários e podem ser apagados de um dia para o outro.

Por isso foi gerado outro prompt: “esse aplicativo gerado pode ser usado por mais de uma pessoa ao mesmo tempo”

Desta maneira, foi obtida a seguinte resposta:

### 1. O estado atual é local e temporário

No código que forneci, as informações (nome, renda e gastos) ficam salvas apenas na **memória volátil** da aba do navegador que está aberta.

- **Se você atualizar a página (F5):** Todos os dados desaparecem e o chat recomeça do zero.
- **Acesso em dispositivos diferentes:** Se você abrir o arquivo no seu celular e outra pessoa abrir no notebook dela, cada um verá sua própria versão isolada. Eles não "conversam" entre si.

### 2. Como permitir o uso por mais pessoas (Caminho para o Produto Real)

Para que várias pessoas usem o mesmo aplicativo e os dados sejam salvos, precisaríamos evoluir o protótipo para uma arquitetura de **Cliente-Servidor**.

Para transformar este protótipo em algo que várias pessoas usem simultaneamente, os próximos passos seriam:

- **Banco de Dados (Ex: Firebase ou Supabase):** Em vez de salvar o nome e a renda em uma variável `userData`, o app enviaria esses dados para um banco de dados na nuvem.
- **Sistema de Login:** Cada usuário teria seu e-mail e senha. Assim, ao entrar em qualquer aparelho, o app buscaria os dados específicos daquela pessoa.
- **API (Backend):** Um servidor processaria as mensagens de todos os usuários ao mesmo tempo, garantindo que o gasto que eu anotei no meu celular não apareça no saldo da sua conta.

### 3. Uma solução "Meio Termo" (LocalStorage)

Se você quiser que o app **lembre** dos seus dados mesmo depois de fechar o navegador (mas ainda apenas no seu computador), eu posso ajustar o código para usar o localStorage. Isso faria com que o app parecesse mais "real" para testes individuais de longa duração.

Optou-se de gerar outro código com a terceira opção. E assim obteve-se o seguinte aplicativo:



O visual é muito parecido com o anterior, mas a funcionalidade ficou um pouco melhor.

Segue o link do arquivo gerado das interações com a inteligência artificial:

<<https://github.com/danielegpbpuppin-boop/trabalhosiniciais/blob/main/app%20finan.txt>>

### 3. Entregando o Desafio na DIO

#### 1- O que funcionou bem?

As aplicabilidades simples do aplicativo, como inserir a renda, os gastos funcionaram muito bem. E principalmente com design simples, o que favoreceu o uso por um número de usuários grandes.

2- O que não funcionou como o esperado?

Opções mais complexas do aplicativo, como por exemplo a geração de metas, geração de gráficos não foram possíveis de serem criadas.

A dificuldade maior é no uso do “Lovable”, pois não permitiu nenhuma interação.

O aplicativo foi criado, porém não foi possível

3- O que aprendeu sobre conversar com IAs?

Na prática com a Inteligência Artificial foi possível criar algo novo em pouco tempo. Porém foi necessário uma quantidade mínima de entendimento para que a atividade proposta fosse finalizada. Concluindo, a inteligência artificial é uma facilitadora de processos e não a detentora.