1. Requisitos Funcionais (RF)

Módulo: Cadastro de Contas (Bancos e Cartões)

RF001 - Gerenciar Contas: O sistema deve permitir ao usuário Cadastrar, Visualizar, Editar e Excluir (CRUD) contas bancárias e cartões de crédito.

Campos mínimos: Nome da conta, Tipo (Banco/Cartão), Saldo Inicial (para bancos), Limite Total (para cartões), Instituição.

RF002 - Gerenciar Categorias: O sistema deve permitir ao usuário Cadastrar, Visualizar, Editar e Excluir (CRUD) categorias de receitas e despesas (Ex: Salário, Alimentação, Lazer).

Módulo: Lançamentos (Receitas e Despesas)

RF003 - Lançamento Rápido: O sistema deve permitir o registro rápido de uma transação (Receita ou Despesa).

Campos: Tipo (Receita/Despesa), Valor, Data, Categoria, Conta (Origem/Destino), Descrição (opcional).

RF004 - Lançamento Detalhado: O sistema deve permitir o registro de transações parceladas (ex: compra no cartão em 12x) e agendadas (ex: conta de luz todo dia 10).

Módulo: Orçamentos e Metas

RF005 - Controle de Orçamento: O sistema deve permitir que o usuário defina um limite de gastos mensais por categoria.

RF006 - Acompanhamento de Orçamento: O sistema deve alertar (visualmente) o usuário quando o gasto de uma categoria estiver próximo ou ultrapassar o limite definido.

Módulo: Dashboard e Relatórios

RF007 - Balanço Mensal: O dashboard deve exibir o saldo atual (Receitas - Despesas) do mês corrente.

RF008 - Gráfico de Evolução: O sistema deve gerar um gráfico comparativo de receitas vs. despesas nos últimos 6 ou 12 meses.

RF009 - Gráfico de Categorias: O sistema deve exibir um gráfico (pizza ou barras) com os maiores gastos por categoria no mês selecionado.

2. Requisitos Não Funcionais (RNF)

RNF001 - Usabilidade (Interface)

UX Mobile First: A interface do dashboard e da tela de lançamento rápido deve ser responsiva e intuitiva, preferencialmente pensada primeiro para mobile (celular), já que controle financeiro é muito feito "on-the-go".

Feedback Visual: O sistema deve fornecer feedback imediato para ações do usuário (toast de sucesso/erro, loading states).

RNF002 - Performance

Carregamento de Relatórios: O dashboard com gráficos (Balanço mensal) deve carregar em menos de 2 segundos, mesmo com grande volume de dados, utilizando cache ou consultas otimizadas no banco de dados.

RNF003 - Segurança

Autenticação: O sistema deve exigir login (autenticação) para acesso, garantindo que um usuário veja apenas os seus próprios dados financeiros.

Dados Sensíveis: Informações sensíveis não devem ser trafegadas ou armazenadas sem criptografia.

RNF004 - Tecnologia e Banco de Dados

Estrutura de Dados: O banco de dados deve ser relacional (SQL) para garantir consistência nas transações financeiras (ex: PostgreSQL, MySQL ou SQLite).

API: As comunicações entre front-end (telas) e back-end (servidor) devem ser feitas via API RESTful ou GraphQL.

3. Divisão de Tarefas (Sugestão de Sprint)

Sprint 1: Fundação e Cadastros

Modelagem e Banco de dados (Equipe 1):

Criar o modelo relacional: Tabelas users, accounts, categories, transactions, budgets.

Configurar o ambiente e conexão com banco de dados.

CRUD de Contas e Categorias (Equipe 2):

Desenvolver o backend (API) para criar/editar/deletar contas e categorias.

Desenvolver as telas (frontend) de listagem e formulários de Contas e Categorias.

Conectar o frontend à API.

Sprint 2: Funcionalidade Principal

Tela de Lançamento Rápido (Equipe 3):

Desenvolver a interface principal de lançamentos (botões ou input rápido).

Conectar com as contas e categorias já cadastradas.

Implementar a lógica de salvar a transação no banco.

Lógica de Orçamentos e Metas (Equipe 4):

Desenvolver a tela para criar metas de orçamento por categoria.

Criar a "regra de negócio" que calcula, no backend, o quanto já foi gasto em cada categoria no mês.

Comparar o gasto real com a meta e gerar um status (ok, alerta, estourado).

Sprint 3: Visualização e Relatórios

Dashboard com gráficos (Equipe Mista):

(Back) Criar endpoints (rotas na API) que consultem o banco e retornem os dados agregados: total de receitas do mês, despesas, e dados agrupados por categoria.

(Front) Consumir esses endpoints e desenhar os gráficos utilizando bibliotecas como Chart.js, Recharts ou ApexCharts.

Integrar o gráfico de Balanço Mensal e o balanço por categoria.