EveryFin - Módulo de Entradas Financeiras (Documento Técnico e Negocial)
Versão Detalhada e Aprimorada

Sumário

---

1. Visão Geral do Módulo de Entradas Financeiras

O módulo de Entradas Financeiras do EveryFin é projetado para oferecer um controle detalhado e eficiente sobre todas as receitas do usuário, desde o cadastro até a análise.

1.1. Objetivo

O objetivo principal é permitir que o usuário controle suas receitas com um cadastro completo e a aplicação de filtros detalhados. Além disso, busca associar clientes às entradas de forma automatizada, agilizando o processo e aumentando a rastreabilidade dos recebíveis.

1.2. Escopo e Valor de Negócio

- **Módulo Essencial**: Este módulo é essencial para o controle de entradas e recebíveis de qualquer sistema financeiro.

- **Rastreabilidade e Gestão**: A associação de clientes ao lançamento aumenta a rastreabilidade e a capacidade de gestão das receitas.
- **Análise Financeira**: Os filtros detalhados ajudam significativamente na análise financeira, permitindo que o usuário compreenda a origem e a natureza de suas receitas.
- **Percepção do Sistema**: A facilidade de uso, acessibilidade e a integração visual elevam a percepção de valor do sistema EveryFin.

2. Funcionalidades do Módulo de Entradas

O módulo de Entradas abrangerá as operações CRUD (Create, Read, Update, Delete) completas e funcionalidades de filtragem.

2.1. Cadastro de Entradas (Formulário)

O formulário de cadastro de entradas (`FormEntrada.tsx`) incluirá os seguintes campos obrigatórios:

- **Descrição**: Um campo de texto para descrever a receita.
- **Valor**: Um campo numérico para o valor da entrada, que deve ser positivo.
- **Data**: Um seletor de data para registrar quando a entrada ocorreu.
- **Categoria**: Um campo para classificar a entrada (ex: "fixa", "variável", "outros", "salário", "freelance"). Esta categoria pode ser selecionada de uma lista predefinida ou permitir a criação de novas, conforme o documento de Categorias.
- **Cliente**: Um campo de autocomplete vinculado ao cadastro de clientes, que permitirá associar a entrada a um cliente existente.

2.2. Listagem de Entradas

A lista de entradas (`ListaEntradas.tsx`) exibirá todas as receitas cadastradas pelo usuário, com as seguintes características:

- **Tabela com Paginação**: Os dados serão apresentados em uma tabela que suporta paginação para lidar com grandes volumes de registros.
- **Ordenação**: As colunas da tabela serão ordenáveis (ex: por data, valor, descrição) para facilitar a análise.
- **Visualização Detalhada**: Cada linha da tabela poderá ser clicável para exibir detalhes da entrada ou iniciar o fluxo de edição.

2.3. Filtros de Entradas

A funcionalidade de filtro (`FiltroEntradas.tsx`) permitirá ao usuário refinar a visualização da lista de entradas:

- **Filtro por Data**: Seleção de um período inicial e final.
- **Filtro por Categoria**: Seleção de uma ou mais categorias para visualizar entradas específicas.

- **Filtro por Cliente**: Seleção de um cliente específico para ver todas as entradas associadas a ele.

2.4. Edição e Exclusão de Entradas

- **Edição (`PUT /entradas/:id`)**: Os usuários poderão editar os detalhes de uma entrada existente através do formulário de cadastro.
- **Exclusão (`DELETE /entradas/:id`)**: Será possível excluir uma entrada, com uma confirmação para evitar exclusões acidentais.

3. Componentes Frontend Dedicados

O frontend será construído com componentes React específicos para o módulo de Entradas, garantindo modularidade e reutilização.

3.1. `FormEntrada.tsx`

- Componente principal do formulário de cadastro/edição de entradas.
- Utilizará `React Hook Form` para gerenciamento de estado do formulário e `Zod` para validação em tempo real dos campos.

3.2. `ListaEntradas.tsx`

- Componente responsável pela exibição da tabela de entradas.
- Incluirá a lógica de paginação, ordenação e interação para edição/exclusão de itens.

3.3. `FiltroEntradas.tsx`

- Componente dedicado aos controles de filtro da lista de entradas.
- Gerenciará o estado dos filtros e enviará os parâmetros para a `ListaEntradas.tsx` acionar a busca na API.

3.4. `TotalizadorEntradas.tsx`

- Um componente específico que exibirá o valor total das entradas no topo da tela, atualizando dinamicamente conforme os filtros são aplicados. Isso fornece um resumo rápido ao usuário.

4. Integração com Backend (API RESTful)

O backend fornecerá os endpoints necessários para as operações CRUD e filtragem do módulo de Entradas.

4.1. Endpoints da API

- **`POST /entradas`**: Para a criação de novas entradas financeiras.

- **`GET /entradas?data=&categoria=&cliente=:id`**: Para a listagem de entradas, com suporte a filtros por data, categoria e ID do cliente.
- **`PUT /entradas/:id`**: Para a edição de uma entrada específica pelo seu ID.
- **`DELETE /entradas/:id`**: Para a exclusão de uma entrada específica pelo seu ID.

4.2. Modelagem de Dados

- O modelo `Entry` no Prisma (conforme detalhado no documento do Backend) será a base para a persistência das entradas, incluindo campos para descrição, valor, data, categoria e `userId` (para vincular ao usuário logado) e `clientId` (para vincular ao cliente/fonte da receita).

4.3. Validação e Persistência

- O backend realizará validações adicionais (utilizando Zod) para garantir a integridade dos dados antes da persistência no PostgreSQL.
- Todas as rotas serão protegidas por autenticação JWT, garantindo que apenas usuários autorizados possam realizar operações.

5. Integração com o Módulo de Clientes

A integração com o módulo de clientes é um diferencial para a agilidade e precisão no cadastro de entradas.

5.1. Campo de Autocomplete

- No formulário `FormEntrada.tsx`, o campo de seleção de cliente será um componente de autocomplete (`AutocompleteInput.tsx`).

5.2. Busca em Tempo Real e Preenchimento Automático

- O autocomplete realizará buscas em tempo real na base de clientes do usuário/empresa via um endpoint `GET /clientes?search=` no backend.
- A funcionalidade de `debounce` otimizará as requisições ao backend, e o preenchimento automático do `clientId` garantirá a correta associação da entrada ao cliente selecionado.

6. Experiência do Usuário (UX) e Acessibilidade (A11y)

O módulo de Entradas será projetado para ser intuitivo, eficiente e acessível a todos os usuários.

6.1. Feedback Visual

- **Alertas e Confirmações**: O sistema fornecerá feedback visual claro para operações (ex: "Entrada cadastrada com sucesso!", "Entrada excluída com sucesso!"). Alertas serão exibidos para erros ou confirmações de exclusão.

- **Estados de Carregamento**: Indicadores de carregamento (spinners, skeletons) serão exibidos durante as requisições à API.
- **Mensagens de Erro/Vazio**: Mensagens informativas serão mostradas em caso de falha no carregamento de dados ou quando a lista de entradas estiver vazia.

6.2. Totalizador de Entradas

- Um totalizador visível no topo da lista exibirá o valor total das entradas, adaptando-se aos filtros aplicados.

6.3. Formulário e Filtros Navegáveis por Teclado

- Todas as interações nos formulários e filtros (campos, botões, selects) serão totalmente navegáveis e operáveis via teclado, com foco visual claro nos elementos.

6.4. Design Responsivo

- O layout da tela de Entradas será totalmente responsivo, adaptando-se a diferentes tamanhos de tela (desktop, tablet, mobile), garantindo a usabilidade dos formulários, filtros e tabela.

7. Testes Automatizados

A qualidade e a confiabilidade do módulo de Entradas serão asseguradas por uma estratégia de testes abrangente.

7.1. Estratégia de Testes

- **Unitários**: Testes focados nos componentes isolados (campos de formulário, lógica de validação, funções de filtro) e nos serviços do backend.
- **E2E com Cypress**: Testes End-to-End para simular fluxos completos do usuário, como:
    - **Criação de Entrada**: Testar o preenchimento do formulário, seleção de cliente via autocomplete, envio e verificação da persistência.
    - **Edição de Entrada**: Seleção de uma entrada na lista, preenchimento do formulário de edição, envio e verificação da atualização.
    - **Exclusão de Entrada**: Seleção de uma entrada, confirmação de exclusão e verificação da remoção.
    - **Filtros**: Aplicação de diferentes filtros (data, categoria, cliente) e verificação da lista de resultados.

7.2. Ferramentas de Teste

- **Jest + React Testing Library**: Para testes unitários e de integração de componentes frontend.
- **Supertest**: Para testes de integração da API do backend.
- **Cypress**: Para testes End-to-End.

7.3. Cobertura de Testes

- Será mantida uma alta cobertura de testes (ex: 90%) para as funcionalidades críticas do módulo de Entradas, garantindo a robustez do sistema.

8. CI/CD

O módulo de Entradas será integrado ao pipeline de CI/CD para automação de builds e deploys.

8.1. Pipeline de Integração Contínua

- O pipeline no GitHub Actions garantirá que o código do módulo de Entradas passe por linting, testes unitários, de integração e E2E a cada push ou Pull Request.

8.2. Deploy Contínuo

- Somente builds validados por todos os testes serão implantados automaticamente nos ambientes de staging e produção via Vercel (frontend) e Render.com (backend).

9. Proposta Negocial Aprimorada

A implementação do módulo de Entradas Financeiras é um pilar fundamental para o valor do EveryFin.

9.1. Valor Estratégico do Módulo

- **Controle Centralizado**: Oferece ao usuário um controle centralizado e detalhado sobre todas as suas fontes de receita, fundamental para uma gestão financeira eficaz.
- **Rastreabilidade Aprimorada**: A capacidade de associar clientes aos lançamentos de entrada aumenta a rastreabilidade e a capacidade de análise das origens dos recebíveis.

9.2. Benefícios Chave

- **Análise Financeira Otimizada**: Os filtros detalhados facilitam a análise financeira, permitindo ao usuário identificar tendências, as maiores fontes de receita e tomar decisões mais informadas.
- **Facilidade de Uso**: O design intuitivo, o campo de autocomplete e o feedback visual positivo elevam a facilidade de uso do sistema.
- **Acessibilidade e Integração Visual**: A atenção à acessibilidade e a integração visual consistente tornam o sistema mais atraente e funcional para um público mais amplo, aumentando a percepção de valor e a fidelização do usuário.

10. Prompt Reutilizável

Este prompt pode ser utilizado para solicitar a implementação de módulos de entradas financeiras semelhantes em outros sistemas, mantendo o padrão de excelência e detalhamento.

"Olá, [Nome do Modelo]!

Implemente um módulo de entradas financeiras completo em um sistema de gestão, com os seguintes requisitos:

1. **CRUD Completo**: Permitir Cadastro, Listagem, Edição e Exclusão de entradas via formulário validado.
2. **Campo de Cliente com Autocomplete**: Integração com o módulo de clientes, utilizando um campo de autocomplete para seleção de clientes na entrada.
3. **Filtros Detalhados**: Implementar filtros por data, categoria e cliente para a listagem de entradas.
4. **Totalizador de Entradas**: Exibir um totalizador no topo da lista que se atualiza com os filtros.
5. **Integração com API e Token**: Backend com rotas RESTful seguras (JWT) e integração frontend-backend via Axios.
6. **Testes Automatizados**: Inclusão de testes unitários (campos, filtros, botões) e E2E (criação, edição, exclusão, filtro).
7. **CI/CD Habilitado**: Garanta que o pipeline de CI/CD no GitHub Actions e o deploy contínuo (Vercel) sejam habilitados.

Finalize com um documento técnico-negocial detalhado em PDF, incluindo visão geral, funcionalidades, componentes, integração, UX/A11y, testes, CI/CD e a proposta de valor.

Documento expandido para excelência máxima e alinhamento com padrões UX e de negócios."

Fontes

Criar Resumo em Áudio

Deep Research
Canvas
Vídeo

O Gemini pode comete