

Documento de Requisitos de Software

Sistema de Controle de Gastos Pessoais – SCGI

Desenvolvedores

Anthony Guilherme Vieira Moraes

João Pedro de Souza Fraga

Paulo Sócrates de Souza Pinheiro

Marcos Antônio da Silva Almeida

Vinícius Santos Leite

Histórico de Alterações

Data	Versão	Descrição	Autor

1. Análise do Problema

Muitos usuários enfrentam dificuldades para manter o controle adequado de suas finanças pessoais ou organizacionais, especialmente quando não utilizam ferramentas apropriadas para acompanhar receitas, despesas e o cumprimento de metas financeiras. Essa falta de organização pode levar à perda de informações importantes, gastos desnecessários, endividamento e dificuldade na tomada de decisões. Além disso, soluções disponíveis no mercado nem sempre são acessíveis, intuitivas ou adaptadas às necessidades específicas de cada usuário. Diante desse cenário, surgiu a necessidade de um sistema simples, eficiente e acessível que ofereça uma visão clara e organizada da vida financeira do usuário.

2. Necessidades Básicas do Cliente

O cliente necessita de um sistema que permita o controle completo de suas finanças, com funcionalidades para registrar receitas e despesas, categorizar transações, definir metas financeiras e acompanhar sua evolução ao longo do tempo. É essencial que o sistema ofereça relatórios visuais e gráficos para facilitar a análise do comportamento financeiro. Além disso, espera-se uma interface web responsiva, de fácil navegação, acessível em diferentes dispositivos e com suporte a múltiplos usuários, garantindo personalização e segurança das informações.

3. Estudo de Viabilidade

Por se tratar de um trabalho institucional, supervisionado pelo professor, e considerando que as ferramentas utilizadas são softwares gratuitos, o desenvolvimento do sistema mostra-se totalmente viável dentro dos parâmetros estabelecidos. Além disso, o ambiente acadêmico oferece o suporte necessário para a realização das etapas do projeto, garantindo que os objetivos sejam alcançados sem a necessidade de recursos financeiros adicionais. Dessa forma, o sistema pode ser desenvolvido de maneira acessível, eficiente e de acordo com as orientações pedagógicas.

3.1 Viabilidade Técnica

O sistema é viável tecnicamente, pois um computador pessoal é todo o equipamento necessário para o seu desenvolvimento e execução. Além disso, o desenvolvedor possui os conhecimentos e habilidades necessários para utilizar as ferramentas de programação, bancos de dados e demais recursos tecnológicos que serão empregados no projeto. As tecnologias escolhidas são amplamente compatíveis entre si, de fácil acesso e possuem boa documentação, o que garante manutenção e futuras atualizações do sistema. Dessa forma, não há barreiras técnicas significativas que impeçam a implementação do projeto.

3.2 Viabilidade Econômica

Por se tratar de um projeto de estágio supervisionado, não há orçamento específico destinado à sua execução. No entanto, o sistema é economicamente viável, uma vez que não exige investimentos financeiros adicionais além dos recursos já disponíveis para o desenvolvimento. As ferramentas e tecnologias utilizadas são de acesso gratuito ou possuem versões open source, o que elimina custos com licenças ou infraestrutura. Assim, o projeto pode ser realizado de forma eficiente e sem comprometer recursos financeiros.

4. Missão do Software

O software tem como objetivo facilitar o controle de gastos para um cliente que deseja ter mais praticidade e organização no gerenciamento de suas finanças pessoais. Por meio do sistema, será possível registrar entradas e saídas de dinheiro, categorizar despesas, gerar relatórios e acompanhar o saldo disponível de forma simples e intuitiva. Além disso, o software visa auxiliar o usuário a identificar padrões de consumo, planejar melhor seu orçamento e alcançar maior controle financeiro no dia a dia.

5. Limites do Sistema

O sistema é voltado para o controle pessoal de gastos, não sendo indicado para uso empresarial ou em grandes organizações. A base de dados utiliza SQLite, sendo adequada para aplicações locais ou de pequeno porte, mas podendo apresentar limitações em cenários com grande volume de transações. A interface é web responsiva, embora dependa de um navegador atualizado para funcionar corretamente. Vale destacar que o sistema não realiza integração bancária automática, sendo necessário que o usuário registre receitas e despesas manualmente. A segurança do SCGI é básica, pensada para ambientes acadêmicos ou demonstrativos, não sendo recomendada para uso financeiro real em larga escala sem adaptações. Além disso, o sistema não oferece suporte a múltiplas moedas, câmbio ou cálculos complexos de investimentos.

6. Benefícios Gerais

Entre os benefícios gerais, o SCGI facilita o controle e a organização financeira pessoal, permitindo que o usuário visualize suas receitas, despesas e saldo de maneira clara. A categorização das transações ajuda na análise de como o dinheiro está sendo gasto, enquanto o acompanhamento de metas financeiras incentiva disciplina nos gastos. O sistema ainda disponibiliza relatórios e gráficos dinâmicos, tornando a visualização das informações mais intuitiva. Por ser simples, leve e acessível, ele pode ser executado em qualquer computador com Python e navegador moderno, ao mesmo tempo que favorece o aprendizado acadêmico, integrando conceitos de programação backend, frontend, banco de dados e usabilidade.

7. Requisitos Funcionais

ID	Funcionalidade	Nessecidades	Prioridade
RF1	Cadastrar Usuário	O usuário poderá se cadastrar no sistema para poder ter acessor aos seus dados.	Essencial
RF2	Listar Usuários	O sistema deverá poder listar os sários cadastrados junto a suas informações.	Essencial
RF3	Consultar Dados de Usuário	O usuário poderá visualizar seus dados.	Essencial
RF4	Editar Dados de Usuário	O usuário poderá modificar seus dados.	Essencial
RF5	Apagar Usuário	O usuário poderá apagar seu cadastro.	Essencial
RF6	Autenticar de Usuário	O usuário poderá autenticar-se no sistema, assim podendo acessar seus dados e manipulá-los	Essencial
RF7	Cadastrar Transação	O usuário poderá registrar transações no sistema.	Essencial
RF8	Listar Transações	O usuário poderá listar as transações que o mesmo cadastrou	Essencial

RF9	Consultar Dados de Transação	O usuário poderá visualizar os dados de uma transação específica cadastrada pelo mesmo.	Essencial
RF10	Editar Dados de Transação	O usuário poderá editar os dados de uma transação específica cadastrada pelo mesmo.	Essencial
RF11	Apagar Transação	O usuário poderá apagar uma transação criada pelo mesmo.	Essencial
RF12	Cadastrar Meta	O usuário poderá cadastrar metas estabelecidas pelo mesmo no sistema.	Essencial
RF13	Listar Metas	O usuário poderá listar as metas que o mesmo cadastrou	Essencial
RF14	Consultar Dados da Meta	O usuário poderá visualizar os dados de uma meta específica cadastrada pelo mesmo.	Essencial
RF15	Editar Dados da Meta	O usuário poderá editar os dados de uma meta específica cadastrada pelo mesmo.	Essencial
RF16	Apagar Meta	O usuário poderá apagar uma meta que foi criada pelo mesmo.	Essencial
RF17	Atingir Meta	O usuário poderá atingir a meta estabelecida pelo mesmo, assim a conluindo.	Essencial
RF18	Emitir Relatórios em Formato	O sistema poderá emitir relatórios com dados escolhidos pelo usuário em formatos CSV, JSON e XML.	Essencial

RF21	Gerar Relatórios com Base em Gráficos	O sistema poderá emitir relatórios em formato de gráficos para facilitar a visualização dos dados.	Essencial
------	---------------------------------------	--	-----------

8. Analise de Requisitos Não Funcionais

Id	Requisitos	Categoria
NRF1	O sistema deve possuir uma interface simples e intuitiva, de fácil compreensão para o usuários	Usabilidade
NRF2	O sistema deve ser compatível com dispositivos móveis (Android e iOS), além de funcionar em computadores.	Portabilidade
NRF3	O sistema deve proteger os dados do usuário, garantindo confidencialidade das informações financeiras cadastradas.	Segurança
NRF4	O sistema deve registrar e exibir receitas, despesas e relatórios de forma rápida, sem atrasos perceptíveis ao usuário.	Desempenho
NRF5	O sistema deve armazenar os dados de forma que não sejam perdidos em caso de falhas inesperadas, utilizando backup local ou em nuvem.	Confiabilidade

NRF6	O sistema deve ser desenvolvido de forma modular, facilitando futuras atualizações e melhorias.	Manutenibilidade
NRF7	O sistema deve estar disponível para uso pelo menos 99% do tempo, garantindo acesso contínuo ao usuário.	Disponibilidade
NRF8	O sistema deve suportar aumento no número de usuários e no volume de dados sem perda significativa de desempenho.	Escalabilidade
NRF9	O sistema deve ser compatível com diferentes navegadores (Chrome, Firefox, Edge) quando acessado via web.	Compatibilidade
NRF10	O sistema deve utilizar ícones e cores padronizadas para facilitar a navegação.	Consistência

9. Ferramentas de Desenvolvimento e Licença de Uso

- a) Python: Uma linguagem de programação de alto nível, versátil e amplamente utilizada em diversas áreas, como desenvolvimento web, automação, ciência de dados e inteligência artificial.
- b) Unicorn: Um servidor ASGI (Asynchronous Server Gateway Interface) leve e rápido, utilizado para executar aplicações web assíncronas em Python, como as desenvolvidas com FastAPI.
- c) FastAPI: Um framework moderno e de alto desempenho para criação de APIs em Python, com suporte nativo a tipagem, validação automática e documentação interativa.
- d) SQLAlchemy: Uma biblioteca Python que combina recursos do SQLAlchemy e do Pydantic, facilitando a criação de modelos de dados e a interação com bancos de dados relacionais.
- e) HTML: Linguagem de marcação utilizada para estruturar o conteúdo de páginas web.

- f) CSS: Linguagem de estilo usada para definir a aparência e o layout de páginas web.
- g) JavaScript: Linguagem de programação voltada para o desenvolvimento de interatividade e dinamicidade em páginas web.
- h) Chart.js: Uma biblioteca JavaScript que permite criar gráficos interativos e personalizáveis em páginas web utilizando o elemento <canvas>.
- i) Bootstrap: Um framework front-end que facilita o desenvolvimento de interfaces web responsivas e modernas, com uso de HTML, CSS e JavaScript.
- j) Mermaid: Uma ferramenta que permite criar diagramas e gráficos a partir de texto simples, integrada a diversas plataformas de documentação e desenvolvimento.
- k) Git: Um sistema de controle de versão distribuído, usado para gerenciar e acompanhar alterações em projetos de software.
- l) l) GitHub: Uma plataforma online que hospeda repositórios Git, facilitando o versionamento, colaboração e publicação de projetos de software.
- m) Draw.io: Uma ferramenta online gratuita para criação de diagramas, fluxogramas e modelos de arquitetura de sistemas.
- n) BrModelo: Um software utilizado para modelagem de bancos de dados, permitindo criar diagramas entidade-relacionamento (DER) e gerar scripts SQL.
- o) Visual Studio Code (VS Code): Um editor de código-fonte leve e poderoso desenvolvido pela Microsoft, com suporte para diversas linguagens de programação, extensões, depuração, integração com Git e personalização do ambiente de desenvolvimento.

10. Documentação Adicional

10.1. Protótipo do Sistema



10.2. Diagrama de Casos de Uso

