

Centro Universitário Estácio Juiz de Fora - campus Rio Branco  
Análise e Desenvolvimento de Sistemas – Turma 3001

## **Projeto de Desenvolvimento Rápido de Aplicações em Python**

Trabalho apresentado ao professor Anderson Barboza da Cruz, na disciplina Desenvolvimento Rápido de Aplicações em Python, pelas(os) alunos(as), Davi Nascimento, Gustavo Rodrigues, Jamil Salomão e Tarcísio Alves.

Juiz de Fora, 09 de Outubro de 2024.

## **Sumário**

1. Introdução
2. Objetivo
3. Justificativas para Criação do Projeto
  - 3.1. Urgência
  - 3.2. Benefícios
  - 3.3. ROI Tecnológico e Comercial
  - 3.4. Riscos de não agir
4. Requisitos Funcionais
5. Requisitos Não Funcionais
6. Padrão de Projeto
  - 6.1. Orientação a Objetos
7. Padrão de Código
8. Integrantes do Projeto e suas Funções

## **1. Introdução**

O desenvolvimento de uma aplicação web para controle financeiro pessoal exige uma abordagem cuidadosa e estratégica, integrando práticas de Engenharia de Software com princípios de Orientação a Objetos. Este projeto descreve a metodologia e as etapas envolvidas na criação de um sistema que permite aos usuários gerenciar suas finanças de maneira eficiente. A aplicação oferece funcionalidades para cadastro de receitas e despesas, resumo diário e mensal de suas finanças, gráfico simples dos seus gastos, exibição do saldo restante de suas receitas após abatimento das despesas e metas de gastos separadas por categoria proporcionando uma visão clara e simplificada das finanças pessoais.

## **2. Objetivo**

O objetivo deste projeto é desenvolver uma aplicação web que simplifique o gerenciamento financeiro dos usuários. O sistema será criado desde o design inicial da interface até a estruturação dos dados e a implementação das funcionalidades principais. A abordagem incluirá a modelagem eficaz de dados e a implementação de funcionalidades intuitivas e seguras, garantindo a entrega de um software robusto e eficiente para o controle financeiro pessoal.

### **3. Justificativas para a Criação do Projeto**

A crescente dificuldade de pessoas e empresas em gerenciar suas finanças tem se tornado uma preocupação significativa. Muitas enfrentam desafios ao tentar controlar sua renda e despesas usando planilhas, que frequentemente complicam a visualização e o acompanhamento dos gastos. Além disso, a manutenção de dados financeiros em ambientes pouco seguros expõe os usuários a riscos consideráveis, como perda de dados e erros de inserção ou atualização.

Observamos que, apesar das opções mais seguras e eficazes disponíveis no mercado, muitos indivíduos e empresas ainda dependem de planilhas no Excel para o gerenciamento financeiro. Esse método, além de ser propenso a erros, não oferece a flexibilidade e a segurança necessárias para um controle eficiente das finanças.

Portanto, a criação de uma aplicação web dedicada ao gerenciamento financeiro visa fornecer uma solução mais confiável, intuitiva e segura, facilitando o acompanhamento e a administração das finanças de maneira mais eficaz.

#### **3.1 Urgência:**

A urgência de desenvolvimento do projeto é considerada alta, pois ao analisarmos o estado atual em que vivemos, percebe-se que muitas pessoas não sabem manusear e controlar de forma eficaz sua renda e a quantidade de dívidas.

#### **3.2 Benefícios:**

Os benefícios oferecidos pela nossa aplicação são a melhor organização e controle dos ganhos, gastos e investimentos dos usuários, evitando a criação de dívidas, atraso de contas e possuindo mais confiabilidade e integridade dos dados de sua situação financeira.

### **3.3 ROI Tecnológico e Comercial:**

A implementação da nossa aplicação terá um impacto significativo na saúde financeira dos usuários, oferecendo uma série de benefícios que aprimoram a organização e a segurança dos dados financeiros.

Primeiramente, ao adotar padrões rigorosos de Confiabilidade, Integridade e Disponibilidade (CID) na segurança da informação e seguir as diretrizes da Lei Geral de Proteção de Dados (LGPD), garantimos que as informações financeiras dos usuários estarão protegidas contra riscos e acessos não autorizados. Isso proporciona um ambiente seguro e confiável para o gerenciamento das finanças pessoais.

Além disso, a aplicação utiliza uma arquitetura em camadas baseada no padrão MVC (Model-View-Controller), que não só melhora a robustez e a segurança do sistema, mas também assegura que a estrutura da aplicação seja escalável e fácil de manter. Essa organização contribui para uma experiência de usuário mais fluida e sem interrupções.

Por fim, a interface da aplicação é projetada para ser intuitiva e amigável, especialmente para aqueles que não estão familiarizados com o uso de planilhas. Com uma abordagem simplificada e acessível, a aplicação facilita a organização dos gastos e promove um controle financeiro mais eficiente, ajudando os usuários a manter suas finanças em ordem com menos esforço e maior precisão.

### **3.4 Riscos de Não Agir:**

Os riscos de não agirmos são usuários continuarem sem o controle adequado de suas finanças, correndo riscos de perder dinheiro por pagamentos de contas em atraso gerando multas e juros, perda de controle dos ganhos e gastos do mês e falta de segurança em seus dados pessoais e financeiros por estarem sendo armazenados de formas incorreta.

## 4. Requisitos Funcionais

A partir dos requisitos funcionais, o sistema oferecerá ao usuário uma experiência clara e organizada, integrando funcionalidades essenciais para o gerenciamento financeiro. Isso tornará mais fácil entender e utilizar o sistema finalizado. As principais funcionalidades do sistema incluirão:

### Autenticação e Registro:

- **Cadastro de Novos Usuários:** Permite que novos usuários se registrem, criando um perfil com suas credenciais;
- **Login e Logout:** Acesso seguro através de login com usuário e senha, e opção para logout;
- **Recuperação de Senha:** Função para redefinir senha caso o usuário a esqueça.

### Gerenciamento das Finanças:

- **Cadastro de Receitas e Despesas:** Permitir que o usuário registre manualmente suas receitas e despesas, categorizando-as como essenciais, não essenciais ou lazer;
- **Resumo Diário/Mensal:** Exibir um resumo básico das finanças, mostrando quanto foi ganho, quanto foi gasto e o saldo restante;
- **Gráfico Simples de Gastos:** Implementar um gráfico de barras ou pizza que mostra a distribuição das despesas nas diferentes categorias, sem entrar em muitas complexidades;
- **Exibição de Saldo Restante:** Após cadastrar as receitas e despesas, exibir o quanto vai sobrar no final do mês, destacando se o usuário está dentro ou fora do orçamento;
- **Metas de Gastos por Categoria:** Permitir que o usuário defina limites simples de gasto para cada categoria (essenciais, não essenciais, lazer) e exiba um alerta visual se esses limites forem ultrapassados.

### Dashboard:

- **Visão Geral das Finanças:** Interface centralizada mostrando uma visualização macro das finanças do usuário.

## 5. Requisitos Não Funcionais

No que engloba os requisitos não funcionais teremos nossas principais ferramentas de trabalho, softwares organizacionais de comunicação e planejamento, juntamente com linguagens de programação a serem utilizadas, banco de dados, a ferramenta para modelar classes e modelo lógico dele, finalizando com o servidor local de alocação para o projeto, também versionando o projeto. Abaixo detalhamos os elementos principais:

### Ferramentas de Desenvolvimento:

- **Visual Studio Code:** Utilizado como ambiente de desenvolvimento integrado (IDE) para escrever e depurar o código;
- **Figma:** Ferramenta para design de interfaces e prototipagem rápida;
- **Firebase (Google):** Utilizado para administração e design de banco de dados NoSQL.

### Softwares Organizacionais:

- **Discord:** Plataforma para comunicação em tempo real, facilitando reuniões e discussões sobre o projeto. Além disso, será utilizado para planejamento e acompanhamento do progresso do projeto, permitindo a visualização das tarefas e etapas concluídas;
- **GitHub:** Serviço para controle de versão e gerenciamento do código, proporcionando acesso colaborativo e facilitando o versionamento.

### Linguagens de Programação e Frameworks:

- **HTML, CSS e JavaScript:** Para a construção da interface do usuário e lógica de front-end;
- **Python / Flask:** Para o desenvolvimento do back-end, gerenciamento de dados, e lógica de negócios;
- **Bootstrap:** Utilizado para o design responsivo e estilização das interfaces.



### **Servidores e Banco de Dados:**

- **Servidor Flask:** Para servir o back-end e gerir as requisições do sistema;
- **Extensão Live Server:** Utilizado para o desenvolvimento rápido e eficiente do front-end;
- **Firebase:** Utilizado como banco de dados NoSQL na nuvem durante o desenvolvimento e produção.

### **Modelagem:**

- **App.diagrams:** Ferramenta utilizada para modelar diagramas de classes, bem como os modelos conceitual e lógico do banco de dados.

### **Links das Ferramentas Utilizadas:**

#### **Tecnologias Principais:**

- **Flask:** <https://flask.palletsprojects.com/en/3.0.x/>
- **App.diagrams:** <https://www.drawio.com/>
- **Postman:** <https://www.postman.com/downloads/>
- **Bootstrap:** <https://getbootstrap.com/>

#### **Tecnologia Extra:**

- **Biblioteca de Gráfico:** <https://getbootstrap.com/>

## 6. Padrão do Projeto

O padrão de projeto Model-View-Controller (MVC) organiza a aplicação em três componentes principais, separando a lógica de negócios da interface do usuário e facilitando a manutenção e a escalabilidade do sistema. Essa estrutura será aplicada tanto no front-end, utilizando HTML, CSS e JavaScript/React, quanto no back-end, com Python/Flask. A orientação a objetos será um princípio central em todo o desenvolvimento, garantindo modularidade e reutilização de código.

### Model:

Model é responsável pela lógica de dados e regras de negócio. Ele define como os dados são estruturados, manipulados e armazenados. No back-end, Python/Flask será usado para gerenciar os modelos e interagir com o banco de dados.

### View:

View é responsável pela apresentação dos dados e pela interface com o usuário. No front-end, React gerenciará as views, criando componentes reutilizáveis para a interface gráfica.

### Controller:

Controller atua como intermediário entre o Model e a View, processando entradas, atualizando modelos, e retornando respostas. Ele coordena as interações entre o usuário e o sistema.

### 6.1. Orientação a Objetos:

Todo o projeto será desenvolvido utilizando **orientação a objetos (OO)**. Isso significa que tanto no front-end quanto no back-end, as funcionalidades serão implementadas em classes e objetos, promovendo encapsulamento, modularidade e reutilização de código.

- **No Back-end (Python/Flask):** As entidades do banco de dados serão representadas como classes, com métodos para operações de CRUD/API REST e lógica de negócios.
- **No Front-end (HTML, CSS e JavaScript/React):** Componentes de interface e lógica serão estruturados como classes ou componentes funcionais com hooks, permitindo a criação de componentes reutilizáveis e bem-organizados.

## 7. Padrão de Código

Para garantir a consistência, legibilidade e manutenibilidade do código, adotaremos um conjunto de diretrizes de codificação. Este padrão de código facilitará a colaboração e a revisão do código, além de garantir que todos os desenvolvedores sigam práticas uniformes. A seguir, são apresentadas as diretrizes para a implementação tanto no front-end quanto no back-end.

- **Indentação:** Use 1 tab para indentação;
- **Aspas:** Use aspas duplas " para strings e JSONs, exceto onde aspas simples ' são necessárias;
- **Nomeação de Variáveis e Funções:** Use snake\_case para variáveis e funções, PascalCase para nomes de componentes e classes.
- **Comentários:** Use # para comentários de linha única e para blocos de comentários (Utilizar um # em cada linha). Comente o código para descrever a lógica complexa ou funções importantes.

## 8. Integrantes do Projeto e suas Funções

- **Davi Nascimento:** Desenvolvedor Backend e criação/controle do Banco de Dados;
- **Gustavo Rodrigues:** Realização de Testes e criação/controle da Documentação do Projeto;
- **Jamil Salomão:** Desenvolvedor Frontend e UI/UX;
- **Tarcísio Alves:** Desenvolver Frontend e Prototipação do Projeto.