Caio Larroza

Leonardo Costa

Giovanni Armane

Matheus Santos

**Projeto Jarbas:**

**Controle de Finanças Pessoais**

São Caetano do Sul / SP

2017

Caio Larroza

Leonardo Costa

Giovanni Armane

Matheus Santos

**Projeto Jarbas:**

**Controle de Finanças Pessoais**

Trabalho de Conclusão de Curso apresentado à Faculdade de Tecnologia de São Caetano do Sul, sob a orientação do Prof. Humberto Luiz de Toledo, como requisito parcial para a obtenção do diploma de Graduação no Curso de Análise e Desenvolvimento de Sistemas.

São Caetano do Sul / SP

2017

**Agradecimentos**

**Resumo**

LARROZA, Caio; COSTA, Leonardo; ARMANE, Giovanni; SANTOS, Matheus. **Projeto Jarbas: Controle de Finanças Pessoais**. Trabalho de Conclusão de Curso – Faculdade de Tecnologia de São Caetano do Sul – Antônio Russo, São Caetano do Sul, 2017. Projeto de sistema mobile híbrido que auxilia o usuário a controlar suas finanças pessoais e a projetar investimentos para o futuro. O projeto tem a proposta de desenvolver uma aplicação responsiva, intuitiva e moderna, capaz de auxiliar qualquer usuário na tarefa de controlar suas contas contábeis e projetar investimentos para o futuro. É uma aplicação baseada em tecnologias *web*, tornando possível o acesso as informações de qualquer dispositivo que esteja conectado na *internet*.

**Palavras-chave:** finanças; mobile; web; responsivo; moderno; intuitivo.

**Abstract**

**Lista de Abreviaturas e Siglas**

**CSS:** Cascading Style Sheets.

**HTML:** HyperText Markup Language.

**MER:** Modelo Entidade Relacionamento.

**PDF:** Portable Document Format.

**TCC:** Trabalho de Conclusão de Curso.

**W3C:** World Wide Web Consortium.

Sumário

[Introdução 9](#_Toc497649183)

[Objetivos Gerais 9](#_Toc497649184)

[Objetivos Específicos 9](#_Toc497649185)

[Problema 9](#_Toc497649186)

[Justificativa 9](#_Toc497649187)

[Metodologia 9](#_Toc497649188)

[1. Tecnologias Usadas 10](#_Toc497649189)

[1.1. HTML 10](#_Toc497649190)

[1.2. CSS 10](#_Toc497649191)

[1.3. JavaScript 10](#_Toc497649192)

[1.4. AngularJS 11](#_Toc497649193)

[1.5. Ionic 11](#_Toc497649194)

[1.6. C# (C Sharp) 11](#_Toc497649195)

[1.7. .NET Core e ASP.NET Core Framework 12](#_Toc497649196)

[1.8. Entity Framework Core 12](#_Toc497649197)

[1.9. Identity Core 13](#_Toc497649198)

[1.10. Python 13](#_Toc497649199)

[1.11. PostgreSQL 13](#_Toc497649200)

[2. Diagramas de Casos de Uso 14](#_Toc497649201)

[3. Documentação dos Atores 14](#_Toc497649202)

[3.1. Usuário 14](#_Toc497649203)

[3.1.1. Responsabilidades 15](#_Toc497649204)

[3.1.2. Frequência na utilização do sistema 15](#_Toc497649205)

[4. Documento de Casos de Uso 15](#_Toc497649206)

[5. Especificação e levantamento de requisitos 15](#_Toc497649207)

[5.1. Requisitos Funcionais 15](#_Toc497649208)

[5.2. Requisitos Não Funcionais 16](#_Toc497649209)

[6. Diagramas de Classes 17](#_Toc497649210)

[7. Diagramas de Objetos 17](#_Toc497649211)

[8. Diagramas de Pacotes 18](#_Toc497649212)

[9. Diagramas de Componentes e Implantação 18](#_Toc497649213)

[10. Diagramas de Sequência 19](#_Toc497649214)

[11. Diagramas de Entidade Relacionamento 19](#_Toc497649215)

[12. Dicionário de Dados 20](#_Toc497649216)

[13. Layout do Sistema 26](#_Toc497649217)

[14. Considerações Finais 26](#_Toc497649218)

[15. Referências 26](#_Toc497649219)

# Introdução

# Objetivos Gerais

Desenvolver uma ferramenta para controle financeiro do usuário:

* + Controle de gastos e movimentações
  + Poupar (para eventualidades e compras de bens)
  + Acompanhar investimentos
  + Projetar gastos de acordo com o padrão
  + Fluxo de caixa (receita e despesa)

# Objetivos Específicos

Tornar prático, acessível e confiável o controle de finanças. Superar as deficiências encontradas em aplicativos semelhantes. Apresentar uma ferramenta que estaria disponível em qualquer plataforma com suporte a tecnologias web.

# Problema

Como auxiliar o usuário a administrar/gerenciar as suas contas pessoais?

# Justificativa

O sistema se justifica porque um grande problema atualmente é o controle financeiro pessoal. As pessoas têm muita dificuldade em controlar seus gastos e, sendo assim, um sistema que será capaz não só de controlar gastos já feitos, mas também estimar projeções futuras pode ser de grande utilidade para aqueles que se preocupam em manter as suas finanças organizadas.

O desenvolvimento deste tipo de aplicativo também se justifica por ser uma oportunidade para o grupo desenvolvedor de utilizar conhecimentos que foram sendo apresentados durante o curso e que agora poderão ser colocados em prática, tais como HTML, CSS, JavaScript, Engenharia de Software, Banco de Dados, Contabilidade, Economia, Estatística e Estrutura de Dados.

# Metodologia

O sistema será desenvolvido em modelo cliente-servidor com banco de dados PostgreSQL. No lado cliente, serão utilizadas tecnologias *web*, sendo estas HTML, CSS, JavaScript e frameworks especializadas. Do lado servidor, será utilizado de tecnologias do ambiente .NET, da Microsoft, com banco de dados relacional PostgreSQL e scripts Python para tarefas especializadas.

Os requisitos serão levantados juntamente com a coorientadora do projeto Prof. Me. Sônia Cristina Menoce com objetivo de melhor entender as melhores técnicas de controle financeiro, projeções e fontes de pesquisa. Em seguida, serão realizadas algumas pesquisas de campo com o objetivo de melhor entender as deficiências de aplicativos similares já no mercado e como melhor solucionar os problemas apresentados.

# Tecnologias Usadas

## HTML

Criada por Tim Berners Lee na década de 90, HTML é a mais popular linguagem de marcação utilizada na *web*. Se baseia em *tags* escritas de forma estruturada e hierárquica, interpretada pelos browsers modernos. É de fácil aprendizado, não requer programas específicos para seu desenvolvimento, podendo ser escrito com apenas um editor de texto.

O HTML é a linguagem responsável por dar estrutura a uma página *web*. Está atualmente em sua versão HTML5, contando com suporte a *tags* multimídia e melhor especificação de separações de seções na página. As especificações da linguagem são controladas pela W3C (World Wide Web Consortium), um consórcio internacional responsável por estabelecer os padrões de conteúdo para a internet.

No projeto Jarbas, o HTML é responsável pela estruturação do conteúdo na aplicação front-end. É utilizado juntamente com a tecnologia Ionic (vide abaixo).

## CSS

Cascading Style Sheets, ou CSS, é uma linguagem *web* com finalidade de adicionar estilos em páginas HTML. É um conjunto de regras composta por seletores e blocos de seleção, por qual são declarados uma variedade de estilos definidos. Com CSS é possível controlar a posição de elementos HTML na página, definir *backgrounds*, fontes, alinhamentos, etc.

Suas especificações são definidas pela W3C (World Wide Web Consortium) e está atualmente em sua terceira versão, CSS3.

No projeto Jarbas, o CSS é responsável pela estilização do conteúdo na aplicação front-end. É utilizado juntamente com a tecnologia Ionic (vide abaixo).

## JavaScript

JavaScript é uma linguagem interpretada, de tipagem fraca e dinâmica, com suporte a diversos paradigmas de programação - orientação a objetos, protótipo funcional, entre outras. Implementada por todos navegadores *web* modernos, considerado a *língua franca* do desenvolvimento *cliente-side*. É baseada na especificação ECMAScript, padronizada pela Ecma International.

Desenvolvida inicialmente na Sun Microsystems por Brendan Eich, é hoje uma das linguagens de programação mais utilizadas no mundo, com suporte de corporações como Google (desenvolvedora da engine V8 para JavaScript) e Fundação Mozilla.

É responsável por dar funcionalidade e dinamismo às páginas *web*. No projeto Jarbas, o JavaScript é utilizado em sua forma TypeScript (um *subset* JavaScript desenvolvido pela Microsoft) em conjunto com a framework AngularJS (vide abaixo) para realizar requisições HTTP ao servidor, trabalhar com dados e controlar aspectos da interface gráfica.

## AngularJS

O AngularJS é um framework open-source, mantido pela Google, que auxilia no desenvolvimento de *single-page applications* e *web apps*. É construído seguindo o padrão MVVC (model-view-view-model). É utilizado com a linguagem HTML e JavaScript para dar dinamismo as páginas.

Um dos mais conhecidos e utilizados *frameworks front-end*, o Angular encontra-se atualmente na versão Angular 4, entretanto ainda contando com suporte ás versões legadas Angular 1.

No projeto Jarbas, a versão utilizada é o AngularJS 1 – por razões de familiaridade da equipe de desenvolvimento e estabilidade do framework. É responsável pela camada de exibição e controle das *views* da aplicação cliente.

## Ionic

O Ionic é um *framework* destinado ao desenvolvimento de aplicações mobile híbridas. Construído em cima das plataformas AngularJS e Apache Cordova, a framework prove diversos componentes e serviços para a construção de aplicações mobile que utilizam tecnologias *web*, como HTML5 e CSS. Criada pela Drfty Co. em 2013.

No projeto Jarbas, Ionic é responsável pelo auxílio e construção das interfaces da aplicação cliente em conjunto com AngularJS.

## C# (C Sharp)

C# (lê-se CSharp) é uma linguagem de programação de alto-nível, compilada, de tipagem estática e/ou dinâmica forte, parcialmente inferida e com suporte multi-paradigmas de programação – orientação a objetos, funcional, baseada em eventos, estruturada e concorrente.

É objetivamente a linguagem principal do .NET Framework, criado pela Microsoft para desenvolvimento Windows e *web*. Encontra-se atualmente em sua sétima versão - C#7. Uma das linguagens mais maduras e utilizadas no mundo, provê capacidades genéricas para desenvolvimento de qualquer aplicação.

É baseada em um tempo de execução em máquina virtual. Um código C# é compilado e otimizado para um código de máquina especificado, chamado *Common Language Infrastructure*, ou CLI abreviado, que é então executado em uma implementação da máquina virtual *Common Language Runtime*, ou CLR abreviado.

A linguagem é utilizada no projeto Jarbas como base linguagem base das tecnologias do servidor – back-end.

## .NET Core e ASP.NET Core Framework

.NET Core é a mais nova framework de código aberto livre da Microsoft e comunidade .NET. É uma completa reescrita e reestruturação da antiga .NET Framework. Modular e multiplataforma, atualizada para as novas tendências do mercado de desenvolvimento *web*, *mobile*, *IoT*, etc.

O ASP.NET Core é o framework da plataforma .NET Core para desenvolvimento *web*. Originalmente chamado de ASP.NET MVC 5, a Microsoft decidiu alterar seu nome para não confundir o novo framework com uma simples atualização do ASP.NET MVC 4.

No projeto Jarbas o ASP.NET Core é responsável pelo funcionamento do servidor back-end da aplicação. A escolha pela plataforma se deve ao fato de permitir o desenvolvimento e deploy da aplicação em múltiplas plataformas, performance excepcional e suporte extenso por parte da comunidade e Microsoft.

## Entity Framework Core

Componente do ASP.NET Core, o Entity Framework permite ao desenvolvedor conectar sua aplicação com banco de dados de forma simples e fácil. O EF, sigla do framework, trabalha como ORM (*object-relational mapping*), possibilitando que um desenvolvedor programe sua aplicação pensando apenas nas relações entre classes da programação, deixando todo fardo de tratamento, mapeamento e transcrição de dados de objetos para tabelas relacionais de bancos de dados.

O Entity Framework foi escolhido para o projeto não apenas por ser considerado o framework padrão da plataforma .NET – de fato, existem diversos outros frameworks como o Entity, mas também pela agilidade que conseguiu trazer para os desenvolvedores do projeto acoplado com sua leve curva de aprendizado.

## Identity Core

Identity Core é um componente do ASP.NET Core responsável por tratar da autenticação e segurança das aplicações na plataforma.

O Identity Core foi escolhido por ser o mais simples e eficiente framework para o desenvolvimento da camada de autenticação para o projeto Jarbas.

## Python

Python é uma linguagem de programação de alto nível, interpretada, de tipagem dinâmica e forte, e com suporte multi-paradigmas de programação – orientação a objetos, funcional, imperativa, estruturada e concorrente.

Criada por Guido van Rossum, possui a filosofia de prezar por legibilidade de código e simplicidade em alta e baixa escala. Sua principal implementação, o interpretador CPython, é mantido pela comunidade Python e pela Python Software Foundation, uma organização sem fins lucrativos. É adotada por diversas empresas de software pelo mundo, notavelmente Dropbox, Google, Instagram e Spotify.

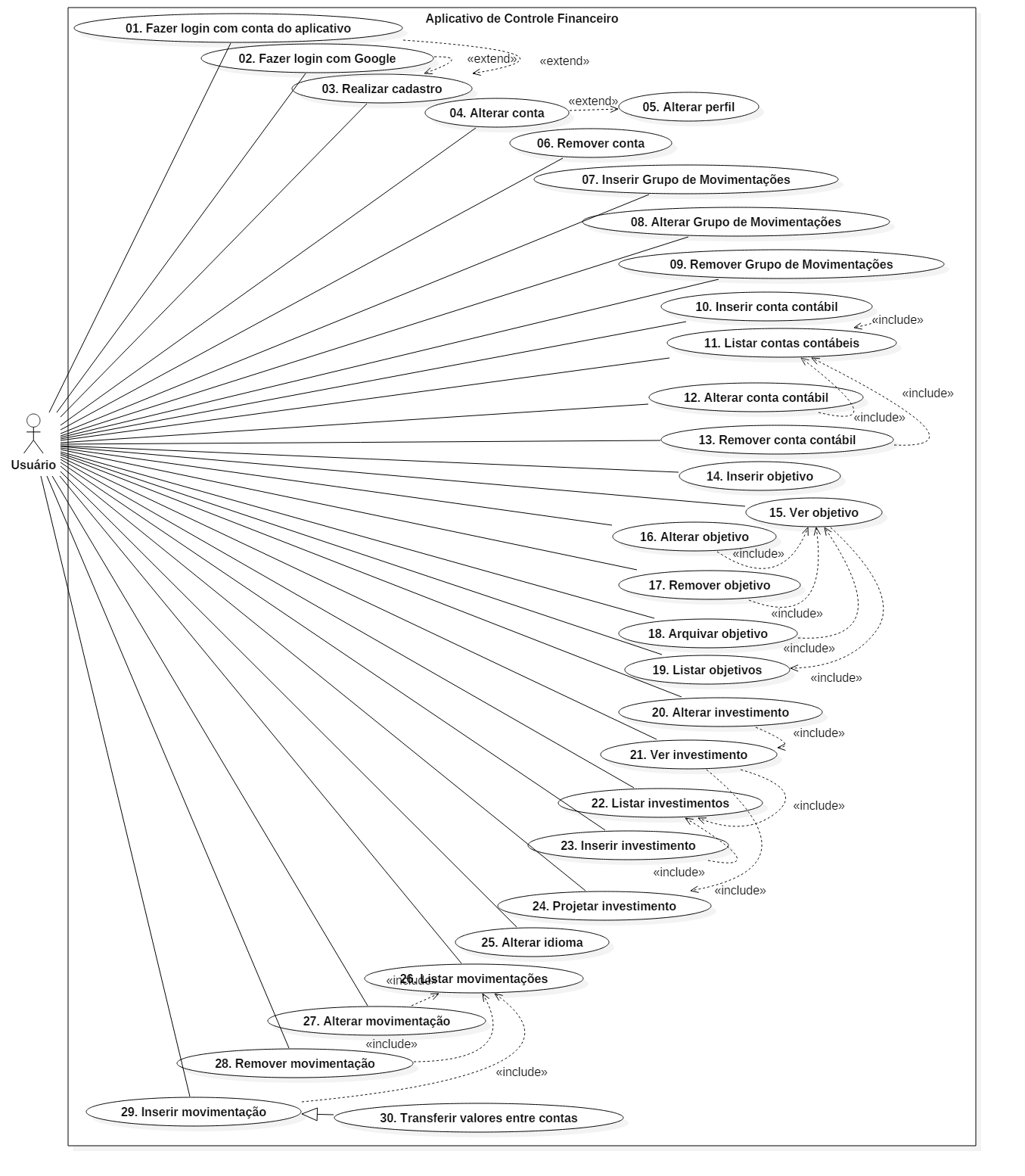
Python foi escolhida para o projeto Jarbas como complementar a tarefas simples onde não fazia sentido executar toda uma aplicação .NET para sua realização ou como suplementar a casos de uso fora do escopo da plataforma principal do servidor.

## PostgreSQL

PostgreSQL, ou simplesmente Postgres, é um sistema de gerenciamento de banco de dados relacional de código aberto livre, com um grande foco em extensibilidade e acordo com padrões de mercado. É desenvolvido pelo PostgreSQL Global Development Group, um órgão internacional composto por diversas empresas da indústria de software.

O Postgres foi escolhido como gerenciador de banco de dados relacional do projeto Jarbas pela sua facilidade de uso, por ser software livre e multiplataforma.

# Diagramas de Casos de Uso



# Documentação dos Atores

## Usuário

Usuário do aplicativo cliente. Utiliza o sistema para cadastrar e controlar suas próprias contas contábeis, investimentos e objetivos. Tem poder total sobre suas contas criadas no sistema.

### Responsabilidades

Cadastrar, alterar e excluir suas contas contábeis, movimentações sobre contas, investimentos e objetivos definidos.

### Frequência na utilização do sistema

Variável, utilizará o sistema para registrar qualquer movimentação financeira que deseja controlar.

# 4. Documento de Casos de Uso

# 5. Especificação e levantamento de requisitos

## 5.1. Requisitos Funcionais

RF001 O sistema deverá manter o cadastro de usuários, com informações de contato.

RF002 O sistema deverá efetuar a autenticação de usuários através de contas do próprio sistema.

RF003 O sistema deverá efetuar a autenticação de usuários através de contas do Google.

RF004 O sistema deverá manter funções CRUD para movimentações financeiras referentes aos usuários cadastrados.

RF005 O sistema deverá manter funções CRUD para contas contábeis referentes a movimentações financeiras.

RF006 O sistema deverá manter funções CRUD para investimentos financeiros referentes aos usuários cadastrados.

RF007 O sistema deverá manter funções CRUD para grupos de movimentações referentes aos usuários cadastrados.

RF008 O sistema deverá manter funções CRUD para objetivos referentes aos usuários cadastrados.

RF009 O sistema deverá ser capaz de realizar transferências entre contas contábeis.

RF010 O sistema deverá ser capaz de efetuar conversão monetária em caso de transferências entre contas com diferentes moedas.

RF011 O sistema deverá, com base nas informações contábeis mantidas, ser capaz de realizar projeções de valores para investimentos.

RF012 O sistema deverá mostrar informações de receitas e despesas através de gráficos.

RF013 O sistema deverá mostrar informações de investimentos através de gráficos.

RF014 O sistema deverá mostrar informações de objetivos através de gráficos.

## 5.2. Requisitos Não Funcionais

RN001 Somente o usuário da conta pode realizar ações na mesma.

RN002 Cada funcionalidade deve ser acessada pelo usuário em no máximo 5 passos.

RN003 A interface de operação será baseada em ícones e textos apresentados em uma tela “touchscreen”.

RN004 O sistema deverá ser portável para outros ambientes sem a necessidade de reprogramação do mesmo.

RN005 O sistema permitirá que o usuário realize cadastro utilizando contas do Google.

RN006 O sistema não fará uso de senhas de contas e cartões bancários.

RN007 O sistema deverá ser desenvolvido utilizando o padrão MVC.

RN008 O sistema deverá ser desenvolvido utilizando as linguagens C#, HTML, CSS e JavaScript, sobre os frameworks Ionic v1, ASP.NET Core e .NET Core.

RN009 O sistema deverá armazenar suas informações localmente em formato JSON e no servidor utilizando-se o sistema gerenciador de banco de dados PostgreSQL.

RN010 Um dispositivo deverá possuir apenas 1 usuário conectado por vez.

RN011 O sistema deverá usar a linguagem Python para obter informações externas, como cotações.

RN012 O sistema deverá ter sincronização de dados offline no aparelho do usuário.

RN013 A comunicação entre o servidor e o aparelho do usuário deve ser feita usando o a notação JSON com métodos HTTP.

# 6. Diagramas de Classes

# 7. Diagramas de Objetos

# 8. Diagramas de Pacotes

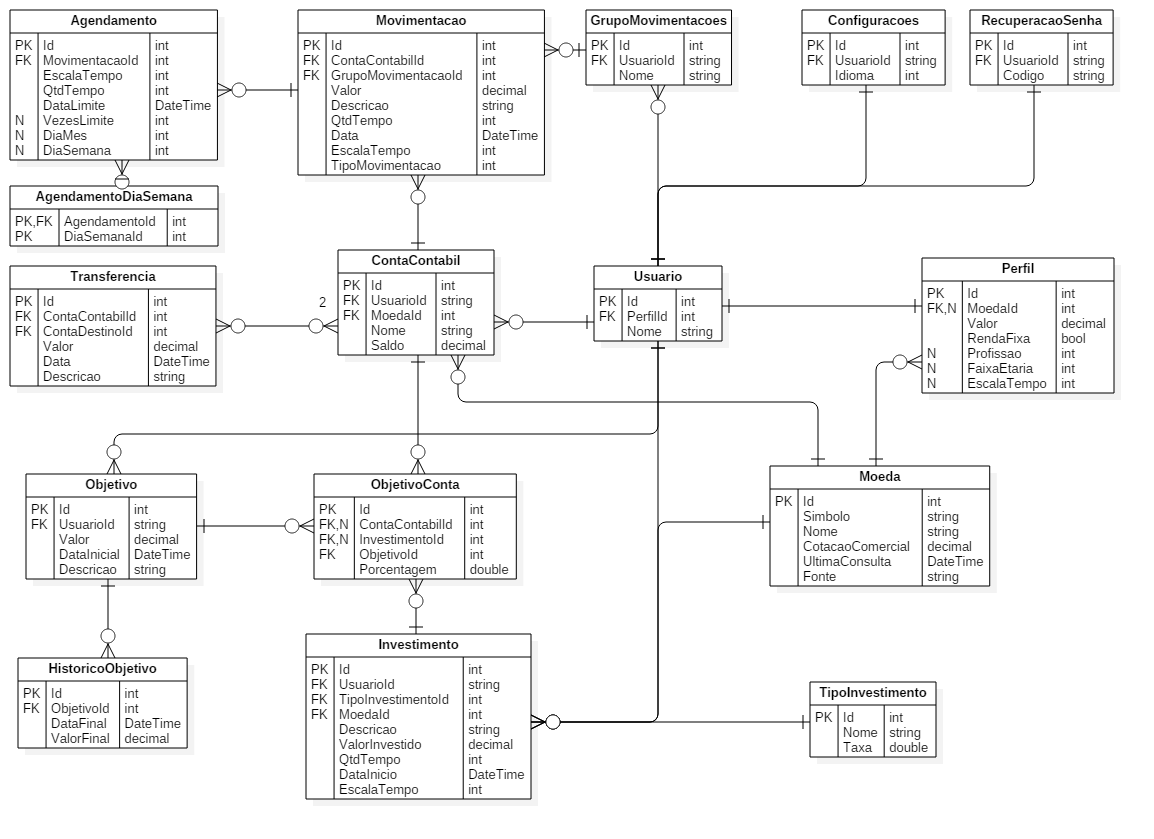


# 9. Diagramas de Componentes e Implantação



# 10. Diagramas de Sequência

# 11. Diagramas de Entidade Relacionamento



# 12. Dicionário de Dados

**Tabela 2. Dicionário de Dados – Entidades do Banco de Dados**

|  |  |
| --- | --- |
| **Nome da Tabela** | **Descrição** |
| Agendamento | Entidade que armazena dados referentes a um agendamento. |
| AgendamentoDiaSemana | Entidade que armazena o dia da semana reservado para um agendamento. |
| Movimentação | Entidade que armazena os dados referentes a uma movimentação contábil. |
| GrupoMovimentacoes | Entidade que agrupa as movimentações de um usuário. |
| ContaContabil | Entidade que armazena as informações de uma conta contábil de um usuário. |
| Transferencia | Entidade que armazena dados referentes a uma transferência contábil. |
| Usuario | Entidade que armazena os dados cadastrais de um usuário. |
| Configuracoes | Armazena informações de configurações de usuários do aplicativo. |
| RecuperacaoSenha | Modelo que armazena informações necessárias para o processo de recuperação de senha. |
| Perfil | Armazena informações de perfil de um usuário. |
| Moeda | Armazena informações de cotações de determinadas moedas. |
| ObjetivoConta | Tabela associativa entre objetivo e conta contábil. |
| Objetivo | Armazena informações referentes a um objetivo a ser alcançado pelo usuário. |
| HistoricoObjetivo | Armazena informações de histórico de um objetivo. |
| Investimento | Armazena informações de um investimento feito pelo usuário. |
| TipoInvestimento | Tabela que armazena categorização de um investimento. |

**Tabela 3. Dicionário de Dados – Entidade Agendamento**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Entidade Agendamento** | | | | | |
| **Atributo** | **Tipo** | **PK** | **FK** | **Nulo** | **Descrição** |
| Id | Integer | X |  |  | Chave primária. |
| MovimentaçãoId | Integer |  | X |  | Chave estrangeira para tabela Movimentação. |
| EscalaTempo | Integer |  |  |  | Escala de tempo utilizada para cálculo. |
| QtdTempo | Integer |  |  |  | Quantidade de tempo. |
| DataLimite | DateTime |  |  |  | Data limite para agendamento. |
| VezesLimite | Integer |  |  | X | Número de vezes de repetição. |
| DiaMes | Integer |  |  | X | Dia do mês para acontecer agendamento. |
| DiaSemana | Integer |  |  | X | Dia da semana para acontecer agendamento. |

**Tabela 4. Dicionário de Dados – Entidade AgendamentoDiaSemana**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Entidade AgendamentoDiaSemana** | | | | | |
| **Atributo** | **Tipo** | **PK** | **FK** | **Nulo** | **Descrição** |
| AgendamentoId | Integer | X | X |  | Chave primária estrangeira sobre tabela Agendamento. |
| DiaSemanaId | Integer | X |  |  | Identifica o dia da semana a ser usado. |

**Tabela 5. Dicionário de Dados – Entidade Movimentacao**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Entidade Movimentacao** | | | | | |
| **Atributo** | **Tipo** | **PK** | **FK** | **Nulo** | **Descrição** |
| Id | Integer | X |  |  | Chave primária. |
| ContaContabilId | Integer |  | X |  | Chave estrangeira referenciando uma entidade ContaContabil. |
| GrupoMovimentacaoId | Integer |  | X |  | Chave estrangeira referenciando uma entidade GrupoMovimentacoes. |
| Valor | Decimal |  |  |  | Valor transitado na movimentação. |
| Descricao | String |  |  | X | Descrição da movimentação. |
| QtdTempo | Int |  |  |  | Quantidade de tempo. |
| Data | DateTime |  |  |  | Data de realização. |
| EscalaTempo | Integer |  |  |  | Escala de tempo usada. |
| TipoMovimentacao | Integer |  |  |  | Identificador de tipo de movimentação. |

**Tabela 6. Dicionário de Dados – Entidade GrupoMovimentacoes**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Entidade GrupoMovimentacoes** | | | | | |
| **Atributo** | **Tipo** | **PK** | **FK** | **Nulo** | **Descrição** |
| Id | Integer | X |  |  | Chave primária. |
| UsuárioId | String |  | X |  | Chave estrangeira referenciando uma conta de usuário a qual o grupo pertence. |
| Nome | String |  |  |  | Nome do grupo. |

**Tabela 7. Dicionário de Dados – Entidade ContaContabil**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Entidade ContaContabil** | | | | | |
| **Atributo** | **Tipo** | **PK** | **FK** | **Nulo** | **Descrição** |
| Id | Integer | X |  |  | Chave primária. |
| UsuarioId | String |  | X |  | Chave estrangeira referenciando um usuário. |
| MoedaId | Integer |  | X |  | Chave estrangeira que referencia uma moeda para cotações. |
| Nome | String |  |  |  | Nome para entidade. |
| Saldo | Decimal |  |  |  | Saldo da conta. |

**Tabela 8. Dicionário de Dados – Entidade Transferencia**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Entidade Tranferencia** | | | | | |
| **Atributo** | **Tipo** | **PK** | **FK** | **Nulo** | **Descrição** |
| Id |  | X |  |  | Chave primária. |
| ContaContabilId |  |  | X |  | Chave estrangeira referencia a conta contábil de origem. |
| ContaDestinoId |  |  | X |  | Chave estrangeira que referencia a conta contábil de destino. |
| Valor |  |  |  |  | Valor transferido. |
| Data |  |  |  |  | Data de transferência. |
| Descricao |  |  |  | X | Descriçao. |

**Tabela 9. Dicionário de Dados – Entidade Usuario**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Entidade Usuario** | | | | | |
| **Atributo** | **Tipo** | **PK** | **FK** | **Nulo** | **Descrição** |
| Id | Integer | X |  |  | Chave primária. |
| PerfilId | Integer |  | X |  | Chave estrangeira que referencia o perfil de usuário. |
| Nome | String |  |  |  | Nome de usuário. |

**Tabela 10. Dicionário de Dados – Entidade Configuracoes**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Entidade Configuracoes** | | | | | |
| **Atributo** | **Tipo** | **PK** | **FK** | **Nulo** | **Descrição** |
| Id | Integer | X |  |  | Chave primária. |
| UsuarioId | String |  | X |  | Chave estrangeira que referencia um usuário. |
| Idioma | Integer |  |  |  | Idioma utilizado pelo usuário. |

**Tabela 11. Dicionário de Dados – Entidade RecuperacaoSenha**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Entidade RecuperacaoSenha** | | | | | |
| **Atributo** | **Tipo** | **PK** | **FK** | **Nulo** | **Descrição** |
| Id | Integer | X |  |  | Chave primária. |
| UsuarioId | String |  | X |  | Chave estrangeira que referencia um usuário. |
| Codigo | String |  |  |  | Código de recuperação. |

**Tabela 12. Dicionário de Dados – Entidade Perfil**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Entidade Perfil** | | | | | |
| **Atributo** | **Tipo** | **PK** | **FK** | **Nulo** | **Descrição** |
| Id | Integer | X |  |  | Chave primária. |
| MoedaId | Integer |  | X | X | Chave estrangeira referenciando moeda para cotações. |
| Valor | Decimal |  |  |  | Valor. |
| RendaFixa | Boolean |  |  |  | Se usuário tem renda fixa. |
| Profissao | Integer |  |  | X | Profissão do usuário. |
| FaixaEtaria | Integer |  |  | X | Faixa etária do usuário. |
| EscalaTempo | Integer |  |  | X | Escala de tempo utilizada. |

**Tabela 13. Dicionário de Dados – Entidade Moeda**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Entidade Moeda** | | | | | |
| **Atributo** | **Tipo** | **PK** | **FK** | **Nulo** | **Descrição** |
| Id | Integer | X |  |  | Chave primária. |
| Simbolo | String |  |  |  | Símbolo da moeda. |
| Nome | String |  |  |  | Nome coloquial da moeda. |
| CotacaoComercial | Decimal |  |  |  | Cotação comercial. |
| UltimaConsulta | DateTime |  |  |  | Data de última atualização de cotação. |
| Fonte | String |  |  |  | Fonte utilizada para atualizações. |

**Tabela 14. Dicionário de Dados – Entidade ObjetivoConta**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Entidade ObjetivoConta** | | | | | |
| **Atributo** | **Tipo** | **PK** | **FK** | **Nulo** | **Descrição** |
| Id | Integer | X |  |  | Chave primária. |
| ContaContabilId | Integer |  | X | X | Chave estrangeira que referencia uma conta contabil |
| InvestimentoId | Integer |  | X | X | Chave estrangeira que referencia um investimento. |
| ObjetivoId | Integer |  | X |  | Chave estrangeira que referencia um objetivo. |
| Porcentagem | Double |  |  |  | Porcentagem completa do objetivo. |

**Tabela 15. Dicionário de Dados – Entidade Objetivo**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Entidade Objetivo** | | | | | |
| **Atributo** | **Tipo** | **PK** | **FK** | **Nulo** | **Descrição** |
| Id | Integer | X |  |  | Chave primária. |
| UsuarioId | String |  | X |  | Chave estrangeira que referencia um usuário. |
| Valor | Decimal |  |  |  | Valor de objetivo. |
| DataInicial | DateTime |  |  |  | Data inicial do objetivo. |
| Descricao | String |  |  |  | Descrição do objetivo. |

**Tabela 16. Dicionário de Dados – Entidade HistoricoObjetivo**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Entidade HistoricoObjetivo** | | | | | |
| **Atributo** | **Tipo** | **PK** | **FK** | **Nulo** | **Descrição** |
| Id | Integer | X |  |  | Chave primária. |
| ObjetivoId | Integer |  | X |  | Chave estrangeira que referencia um objetivo. |
| DataFinal | DateTime |  |  |  | Data final de um histórico. |
| ValorFinal | Decimal |  |  |  | Valor final de um histórico. |

**Tabela 17. Dicionário de Dados – Entidade Investimento**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Entidade Investimento** | | | | | |
| **Atributo** | **Tipo** | **PK** | **FK** | **Nulo** | **Descrição** |
| Id | Integer | X |  |  | Chave primária. |
| UsuarioId | String |  | X |  | Chave estrangeira que referencia um usuario. |
| TipoInvestimentoId | Integer |  | X |  | Chave estrangeira que referencia um tipo de intestimento. |
| MoedaId | Integer |  | X |  | Chave estrangeira que referencia uma moeda para cotações. |
| Descricao | String |  |  |  | Descrição do investimento. |
| ValorInvestido | Decimal |  |  |  | Valor inicialmente investido. |
| QtdTempo | Integer |  |  |  | Quantidade de tempo em que o investimento transcorre. |
| DataInicio | DateTime |  |  |  | Data de início do investimento. |
| EscalaTempo | Integer |  |  |  | Escala de tempo utilizada. |

**Tabela 18. Dicionário de Dados – Entidade TipoInvestimento**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Entidade TipoInvestimento** | | | | | |
| **Atributo** | **Tipo** | **PK** | **FK** | **Nulo** | **Descrição** |
| Id | Integer | X |  |  | Chave primária. |
| Nome | String |  |  |  | Nome de identificação |
| Taxa | Double |  |  |  | Taxa do investimento. |

# 13. Layout do Sistema

# 14. Considerações Finais

# 15. Referências