PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

PUC Minas Virtual

Pós-graduação *Lato Sensu* em Arquitetura de *Software* Distribuído

Projeto Integrado

Relatório Técnico

Sistema Gerenciador de Investimento de Renda Variável

Eduardo Sampaio Viana

Belo Horizonte
Março 2023

Link apresentação: Apresentação Vídeo

Projeto Integrado – Arquitetura de Software Distribuído

Sumário

1. Introdução

No Brasil os ativos de renda variável são negociados pela B3 que é uma bolsa de valores fundada em São Paulo em 1890. Atualmente o número de investidores na bolsa de valores tem aumentado cada vez mais. As pessoas estão migrando da poupança para investimento de renda variável e renda fixa. Em um cenário atual com uma taxa Selic 13,75% ao ano, a renda fixa tornou-se uma excelente oportunidade de ganhos para o curto prazo. A taxa Selic é a taxa básica de juros da economia brasileira e influência as demais taxas de juros do Brasil. Em contrapartida, os investimentos de renda variável apresentam também grandes oportunidades devido à taxa Selic elevada, fazendo com que as ações e fundos imobiliários fiquem bastantes descontados, pois, se encontram em um ciclo de baixa da bolsa de valores, porém mesmo descontados apresentam um maior risco e devem ser analisados e gerenciados com detalhes.

O mercado financeiro possui uma variedade grande de ativos e a escolha de um ativo de renda variável necessita de muitas informações, tornando uma tarefa difícil mesmo para profissionais experientes no mercado financeiro. Um bom investimento depende de boas decisões, errar na escolha de um determinado ativo pode levar o investidor a perder dinheiro. Temos hoje aproximadamente 400 empresas no qual podemos ser sócios por meio de ações com segmentos variados e 150 fundos imobiliários listados na bolsa de valores, tornando a escolha um pouco difícil. O investidor está buscando comprar ações e fundos imobiliários por um bom preço que paguem bons dividendos. Dividendos são proventos pagos aos acionistas por possuir um determinado ativo. Cada empresa paga seus dividendos em datas diferentes e com frequências diferentes, podendo ser de forma anual, semestral, quadrimestral, trimestral ou mensal, essas informações são importantes para o investidor ter uma previsibilidade de seus ganhos. Outro problema é a visualização de ativos escolhidos que muitos investidores recorrem a ferramentas como Excel ou outros sites extremamente complexos dificultando gerenciamento desses ativos, através dessas ferramentas o investidor tenta visualizar a taxa de rentabilidade dos investimentos e sua evolução patrimonial.

O objetivo deste trabalho é permitir que os investidores de renda variável possam ter informações sobre ações e fundos imobiliários que ajudem na sua escolha para montagem e gerenciamento de sua carteira de investimentos e a visualização de pagamentos de proventos, rentabilidade da carteira de investimento e evolução patrimonial. Ações são títulos de renda variável que representam uma fração do capital social de uma empresa, quando compramos uma ação nos tornamos sócios desta empresa e também dos seus resultados. Os fundos imobiliários são títulos de renda variável que reúnem seus recursos para serem aplicados em conjunto no mercado imobiliário. Uma carteira de investimentos agrupa todos os ativos de um investidor.

A principal motivação deste trabalho é uma tentativa de facilitar e ajudar de maneira social na educação financeira dos brasileiros, pois ao contrário dos Estados Unidos ainda é uma parcela muito pequena da população que conseguem ter conhecimento e dinheiro para realizar investimentos.

O objetivo deste trabalho é apresentar uma descrição do projeto arquitetural sistema gerenciador de investimentos de renda variável.

Os objetivos específicos propostos são:

- Visualizar detalhes ações e fundos imobiliários
- Gerenciar carteira de investimentos
- Visualizar a rentabilidade da carteira e de seus ativos
- Visualizar próximos proventos
- Visualizar evolução patrimonial

2. Especificação Arquitetural da solução

Esta seção apresenta a especificação básica da arquitetura da solução a ser desenvolvida, incluindo diagramas, restrições e requisitos definidos pelo autor, tal que permitam visualizar a macro arquitetura da solução.

2.1 Restrições Arquiteturais

A abaixo as restrições arquiteturais relativas à aplicação a ser desenvolvida:

R1: Deve utilizar a linguagem Python.
R2: Deve ser utilizado o framework FastApi para fornecer uma API REST
R3: Deve ser uma aplicação web utilizando framework NextJs
R4: Deve permitir usuário se autenticar através do google ou por um novo cadastro
R5: Deve permitir autorização através token JWT

2.2 Requisitos Funcionais

Os requisitos funcionais descrevem o como sistema deve ser comportar para atender a expectativas e necessidades do usuário.

Listagem de requisitos funcionais do sistema:

ID	Descrição Resumida	Dificuldade (B/M/A) *	Prioridade (B/M/A) *
RF01	Buscar ativos pelo código	B	A
RF02	Visualizar detalhes de ações	A	A
RF03	Visualizar detalhes de fundos imobiliários	A	A
RF04	Visualizar próximos pagamentos de proventos	A	M
RF05	Visualizar patrimônio	M	A
RF06	Cadastrar uma nova transação de compra de ativos	M	A
RF07	Cadastrar uma nova transação de venda de ativos	M	A
RF08	Atualizar uma transação de compra ou venda de ativos	B	A
RF09	Listar transações realizadas	B	A
RF10	Cadastrar uma nova data de proventos para um ativo	B	B
RF11	Atualizar data de proventos para um ativo	B	B
RF12	Deletar ativos da carteira de investimento	B	B

*B=Baixa, M=Média, A=Alta.

2.3 Requisitos Não-funcionais

Os requisitos funcionais são requisitos que estão relacionados ao uso da aplicação que como restrições e aspectos de qualidade.

Listagem de requisitos não funcionais do sistema:

ID	Descrição	Prioridade B/M/A
RNF01	Validar código do ativo existente	A
RNF02	Validar quantidade mínima de caracteres	B
RNF03	Validar confirmação de senha de usuário	B
RNF04	Resposta API REST em JSON	B
RNF05	Validar e-mail de cadastro do usuário	A

2.4 Mecanismos Arquiteturais

Esta seção deve apresentar uma visão geral dos mecanismos que compõem a arquitetura do software, baseando-se em três estados: (1) análise, (2) design e (3) implementação. Em termos de Análise devem ser listados os aspectos gerais que compõem a arquitetura do software, como: persistência, integração com sistemas legados, geração de logs do sistema, ambiente de front end, tratamento de exceções, formato dos testes, formato de distribuição/implantação (deploy), dentre outros. Em Design deve-se identificar o padrão tecnológico a seguir para cada mecanismo identificado na análise. Em Implementação deve-se identificar o produto a ser utilizado na solução, caso ela fosse implementada.

Análise	*Design*	Implementação
Persistência	ORM	SqlAlchemy
Persistência	Banco de dados Relacional	PostgreSQL
Cache	Banco de dados chave e valor	Redis
Front end	Interface de comunicação com usuário do sistema	Next JS
Back end	Linguagem	Python
API	REST	FastApi
Log do sistema	Implementação dos recursos de log	Logoru
Autenticação e Autorização	Protocolo de autenticação e	OpenID

	autorização	
Segurança Criptografia	Hash	Passlib bcrypt
Versionamento	Versionamento de código	Git
Deploy	Utilização de container	Docker

3. Modelagem Arquitetural

Esta seção apresenta a proposta da modelagem arquitetural do sistema de gerenciamento de renda variável, a solução proposta permite ter uma visão geral do projeto. Para esta modelagem arquitetural optou-se por utilizar o modelo C4 para documentação de arquitetura de software. Dos quatro níveis que compõem o modelo C4 três serão apresentados aqui que são diagrama de contexto, diagrama de container e o diagrama de componente. Para desenvolvimento dos diagramas foi utilizado a ferramenta PlantUML que permite criar diagramas por meio de uma linguagem de texto simples.

3.1 Diagrama de Contexto

O nível 1, o diagrama de contexto, apresenta uma visão geral da macro arquitetura do projeto proposto com nível de abstração bem elevado, mostrando o sistema de software e suas responsabilidades e seus principais usuários e suas dependências externas. O usuário e o administrador acessam o sistema através sua conta pessoal. O sistema possui um mecanismo de autorização de acesso através do Google e utiliza a API do Yahoo Finance para obter cotações dos ativos. A aplicação permite o cadastro de transações de ativos para construir uma carteira de investimentos e também permite análises de ações e fundos imobiliários e visualização de patrimônio cadastrados na plataforma. O papel do administrador e ter um acesso para poder cadastrar novos ativos na plataforma e incluir informações sobre novos pagamentos de proventos. Abaixo a figura do diagrama de contexto:

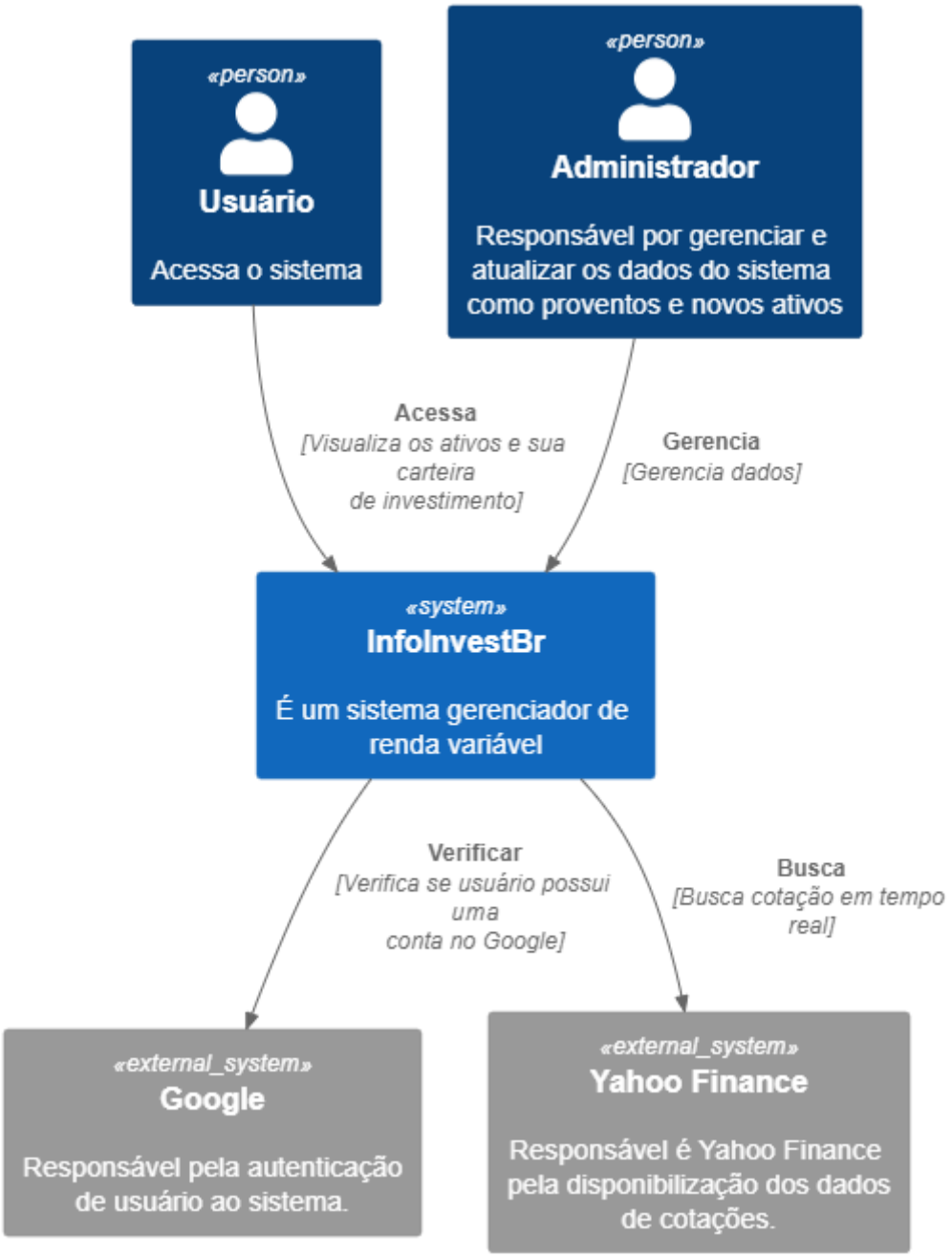

Figura 1 – Diagrama de Contexto da aplicação

Link: [Link para imagem]

3.2 Diagrama de Container

O nível 2, o diagrama de Container amplia o sistema de software e mostra os containers da aplicação como os componentes (aplicativos, armazenamentos de dados, microservices, etc.) que compõem esse sistema de software estão distribuídos e organizados. As decisões de tecnologia que você tomou devem ser contempladas nesse diagrama. O diagrama abaixo apresenta as soluções tecnológicas utilizadas. A proposta do projeto é uma aplicação web, uma Single Page Application construída com o framework NextJs. NextJs é um framework de código aberto criado em 2016 pela Vercel que permite funcionalidades como renderização do lado do servidor e geração de sites estáticos para aplicativos da web baseados em React. O backend possui uma API REST construída com framework FastApi. Framework FastApi é focado no desenvolvimento de API com Python e tem como principais características ser rápido, simples e moderno. Para persistência dos dados foi utilizado SGBD PostgreSQL devido sua alta performance e escalabilidade e também por ser de código aberto gerando um custo bem menor em relação a outras soluções de banco de dados.

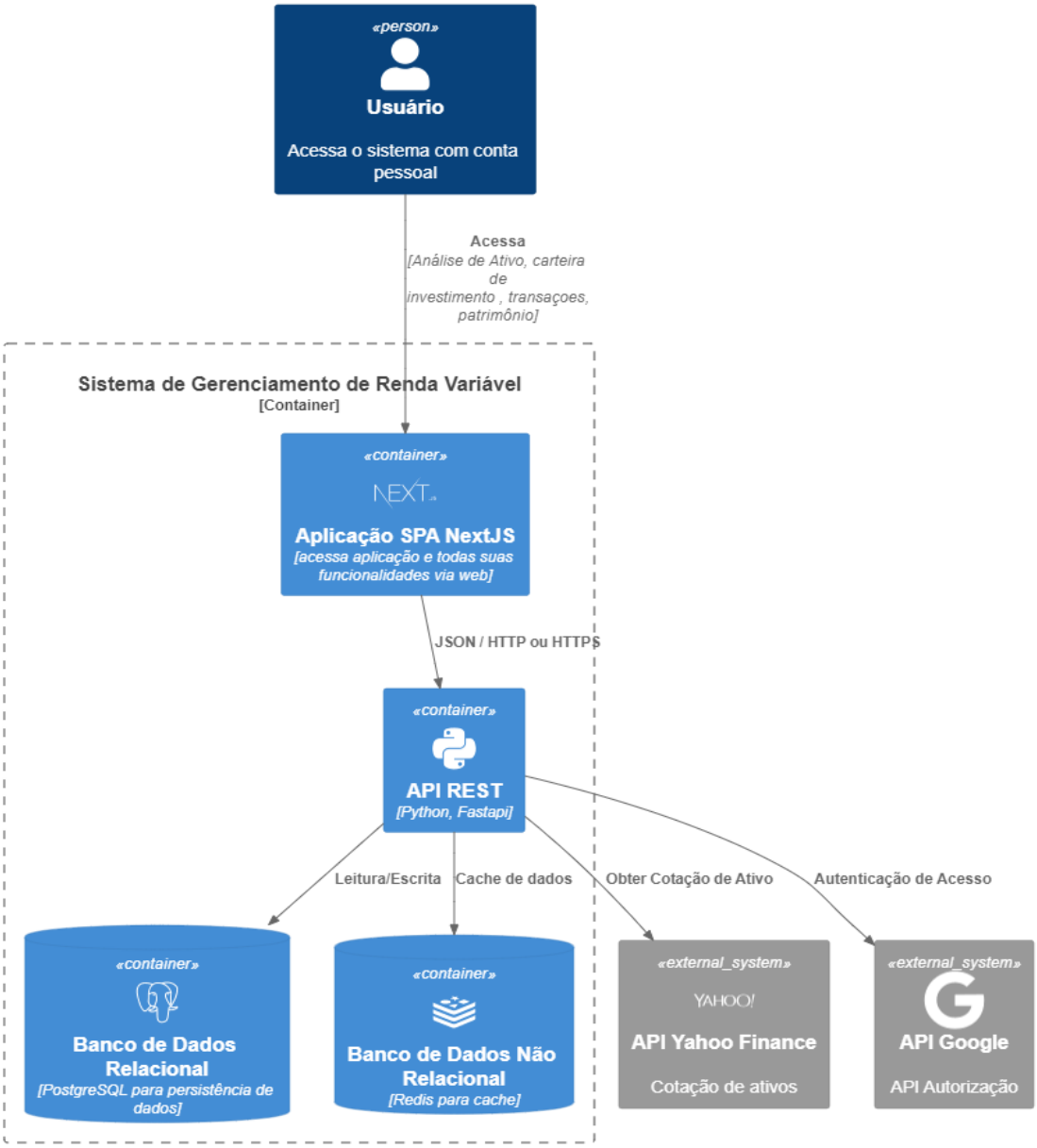

Figura 2 – Diagrama de container.

Link para imagem

3.3 Diagrama de Componentes

O diagrama componente mostra o nível 3 de detalhes para mostrar de maneira individual os detalhes da aplicação. O diagrama abaixo exibe cada funcionalidade da aplicação fornecidas por meio de uma API REST e seus endpoints.

- O componente usuário controller fornece endpoints para manipulação de usuário como a criação de um novo usuário e verificação de autenticação para usuários já cadastrados e esse único endpoint no qual não necessário passar pela camada de segurança da aplicação.
- A Camada de segurança fornece mecanismos para controle de acesso aos endpoints da aplicação através token JWT. Para acesso o usuário envia suas credenciais para aplicação, ao validar essas informações é retornado um token de acesso que autoriza o usuário a poder utilizar a aplicação.
- O controller de análise visa fornecer endpoints com informação em detalhes sobre ação e fundos imobiliários.
- O controller de cotação visa fornece cotações extraídas em tempo real da API do Yahoo Finance.
- O controller de transação visa cadastrar operações de compra e venda formando uma espécie de histórico do usuário, podendo ser consultado a qualquer momento.
- O controller de patrimônio visa consolidar as transações em uma carteira de investimentos, mostrando todos os ativos possuídos de uma maneira agrupada e realizando cálculos que irão ajudar o investidor em suas decisões.
- O controller de importação somente pode ser acesso pelo papel de administrador do sistema, onde ele pode importar planilhas com os de renda variável em lote.
- Componente serviço visa concentrar as regras de negócio da aplicação.
- Componente repositório visa acessar o banco de dados e realizar operações.
- Componente camada de modelo visa através do ORM SqlAlchemy gerar as tabelas no banco de dados através mapeamento feito no modelo.

Abaixo o diagrama de componente:

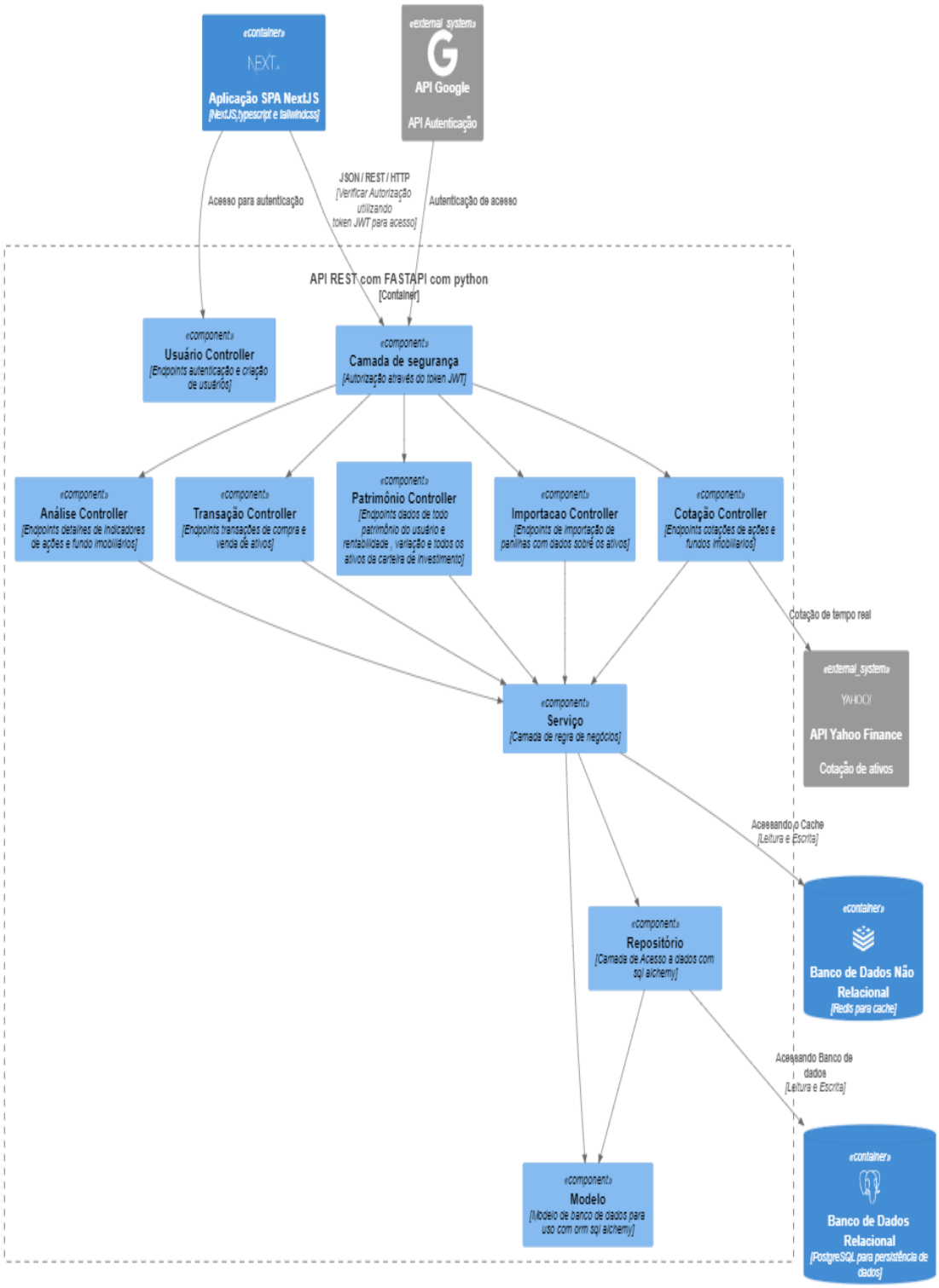

Figura 3 – Diagrama de Componentes.

Link para imagem