

# Sistemas Multiagentes em Mercados Financeiros

**Abstract** — Este trabalho apresenta o desenvolvimento de um sistema multiagente para análise de criptomoedas, com uma arquitetura hierárquica composta por orquestradores e agentes especializados. O sistema recolhe dados de redes sociais, fóruns e mercados financeiros, processa-os com técnicas de análise de sentimento e aprendizagem automática, e fornece previsões e notificações personalizadas. A arquitetura modular e escalável permite a integração de novos agentes e funcionalidades, garantindo eficiência e adaptabilidade. Os resultados demonstram a eficácia do sistema na análise de grandes volumes de dados e na geração de previsões úteis para os utilizadores.

**keywords:** Agentes, Criptomoedas, *Prediction Models*, Mercados Financeiros

---

## 1 Introdução

O mercado de criptomoedas tem vindo a crescer exponencialmente nos últimos anos, tornando-se um dos setores mais dinâmicos e voláteis da economia digital. Este crescimento trouxe consigo desafios significativos para investidores e analistas, que necessitam de ferramentas avançadas para monitorizar, analisar e prever os movimentos do mercado. A complexidade deste domínio reside na sua natureza descentralizada, na grande quantidade de dados gerados em tempo real e na influência de fatores externos, como notícias, redes sociais e eventos globais.

Neste contexto, o presente trabalho propõe o desenvolvimento de um sistema multiagente para análise de criptomoedas, com o objetivo de recolher, processar e analisar dados provenientes de diversas fontes, como redes sociais (Reddit), fóruns (CryptoPanic) e plataformas de notícias. Este sistema visa fornecer informações úteis e previsões baseadas em técnicas de inteligência artificial e análise de sentimento, permitindo aos utilizadores tomar decisões mais informadas.

O sistema multiagente desenvolvido é composto por uma arquitetura hierárquica, onde diferentes agentes desempenham funções especializadas, como a recolha de dados, análise de sentimento e previsão de preços. Além disso, inclui orquestradores que coordenam e gerem as atividades dos agentes, garantindo a execução eficiente das tarefas. A modularidade e escalabilidade da arquitetura permitem a integração de novas funcionalidades e fontes de dados, tornando o sistema adaptável às necessidades futuras.

Os objetivos principais deste projeto incluem:

1. **Recolha de Dados:** Implementar agentes para recolher dados de preços, volumes de transação, notícias e publicações em redes sociais.
2. **Análise de Sentimento:** Processar dados textuais para identificar o sentimento predominante em relação às criptomoedas.
3. **Previsão de Preços:** Desenvolver modelos de aprendizagem automática para prever movimentos de preços.
4. **Notificações e Alertas:** Fornecer notificações personalizadas sobre mudanças significativas no mercado.

Este relatório está estruturado para apresentar o domínio e os objetivos do sistema, descrever a sua arquitetura e funcionamento, analisar criticamente os resultados obtidos e propor melhorias para futuras iterações.

## 2 Arquitetura

A arquitetura do sistema multiagente desenvolvido é composta por uma estrutura hierárquica que combina orquestradores, agentes especializados e pipelines de comunicação. Esta abordagem modular permite a divisão de responsabilidades, garantindo

que cada componente desempenhe funções específicas de forma eficiente e coordenada. Para além disso, é altamente escalável com uma grande facilidade em acrescentar mais processos ou novas metodologias. A seguir, detalhamos os principais elementos da arquitetura.

## 2.1 Orquestradores

Os orquestradores são agentes que desempenham um papel central na arquitetura, sendo responsáveis pela coordenação e gestão das atividades dos agentes especializados. Eles garantem que as tarefas sejam executadas de forma ordenada. São eles que gerem a comunicação entre outros “módulos”, de modo a que cada agente especialista apenas comunique com um orquestrador.

### 2.1.1 Funções dos Orquestradores

- **Iniciação de Agentes especializados:** Os orquestradores enviam comandos para iniciar os agentes especializados, garantindo que cada agente execute as suas tarefas no momento apropriado.
- **Gestão de Fluxo de Dados:** Coordenam a recolha, processamento e armazenamento de dados, assegurando que os dados fluam corretamente entre os agentes.
- **Monitorização:** Supervisionam o estado dos agentes e registam logs de execução para análise posterior.

### 2.1.2 Orquestradores Implementados

- **Orquestrador Global:** Coordena todos os outros orquestradores e garante a integração de todas as funcionalidades do sistema.
- **Orquestrador de Notícias:** Coordena os agentes responsáveis pela recolha de dados de notícias, redes sociais e fóruns.
- **Orquestrador de Análise de Dados:** Coordena os agentes que realizam a análise de sentimento e a identificação de moedas mencionadas nos vários textos.
- **Orquestrador de Criptomoedas:** Gere os agentes responsáveis pela recolha de dados financeiros das criptomoedas, como preços em tempo real, dados diários detalhados (volume transacionado, preço de fecho e de abertura diário, etc.) e o índice de medo e ganância (*Fear-Greed Index*).

## 2.2 Agentes Especializados

Os agentes especializados são responsáveis por executar tarefas específicas, como a recolha de dados, análise de sentimento e previsão de preços. Cada agente é proje-

tado para operar de forma independente, mas comunica com os orquestradores para garantir a coordenação.

Os agentes especializados desempenham funções específicas dentro do sistema multiagente, sendo responsáveis por tarefas como recolha de dados, análise de informações e interação com os utilizadores. Abaixo, detalhamos os diferentes tipos de agentes implementados no sistema.

### 2.2.1 Agentes de Recolha de Dados

Os agentes de recolha de dados são responsáveis por obter informações de diversas fontes externas, garantindo que o sistema tenha acesso a dados atualizados e relevantes. Estes agentes incluem:

- **Agente de Redes Sociais:** Este agente recolhe publicações do Reddit relacionadas com criptomoedas, analisando subreddits específicos para identificar tendências e opiniões.
- **Agente de Fóruns:** Utiliza a API do *CryptoPanic* para recolher dados de fóruns, incluindo discussões e notícias partilhadas por utilizadores.
- **Agente de Notícias:** Recolhe artigos de fontes noticiosas através de *feeds* RSS, permitindo ao sistema monitorizar eventos e notícias relevantes.
- **Agente de Mercados:** Obtém preços e volumes de transação de criptomoedas em tempo real, bem como dados históricos detalhados, preço de abertura, preço de fecho e volume diário.

### 2.2.2 Agentes de Análise de Dados

Os agentes de análise de dados processam as informações recolhidas para extrair conhecimento útil. Estes agentes incluem:

- **Agente de Análise de Sentimento:** Processa textos recolhidos de redes sociais, fóruns e notícias para determinar o sentimento predominante (positivo, negativo ou neutro) em relação ao conteúdo do artigo.
- **Agente de Identificação de Moedas:** Identifica as criptomoedas mencionadas nos textos analisados, associando-as a códigos e nomes específicos. Este agente simplifica as informações para facilitar a análise posterior.

### 2.2.3 Agentes de Previsão

Os agentes de previsão utilizam modelos de aprendizagem automática para prever movimentos de preços de criptomoedas. Estes agentes analisam dados históricos e indicadores derivados, como médias móveis e índices de volatilidade, para gerar previsões que ajudam os utilizadores a tomar decisões informadas.

#### 2.2.4 Agente de Notificação

O agente de notificação é responsável por interagir diretamente com os utilizadores, enviando alertas personalizados com base em condições predefinidas. Este agente monitoriza os dados recolhidos e processados para identificar eventos significativos relacionados com criptomoedas. As notificações podem ser enviadas em dois cenários principais:

- **Notificações em Tempo Real:** O agente verifica continuamente os preços das criptomoedas em tempo real e notifica os utilizadores quando o preço atinge um limite (*threshold*) definido por eles.
- **Notificações Baseadas em Previsões:** O agente utiliza os resultados dos modelos de previsão para alertar os utilizadores sobre possíveis movimentos futuros de preços, permitindo-lhes tomar decisões informadas.

As notificações são enviadas através de email garantindo que os utilizadores recebam informações relevantes de forma rápida e eficiente.

Cada um destes agentes desempenha um papel essencial no funcionamento do sistema, garantindo que os dados são recolhidos, analisados e apresentados de forma eficiente e útil para os utilizadores. Nos próximos capítulos, serão detalhados os resultados obtidos com a colaboração destes agentes.

### 2.3 Comunicação entre Agentes

Os agentes comunicam entre si e com os orquestradores utilizando mensagens estruturadas, através de objetos para uma maior modularidade e encapsulamento. Estas mensagens incluem informações como o tipo de tarefa, os dados a processar e os resultados obtidos.

A comunicação entre os diferentes componentes do sistema é realizada através de pipelines bem definidos, que garantem o fluxo eficiente de dados e mensagens.

#### 2.3.1 Estrutura do Pipeline

- **Entrada de Dados:** Os dados recolhidos pelos agentes de recolha são enviados para os orquestradores, que os encaminham para os agentes de análise.
- **Processamento:** Os agentes de análise processam os dados e enviam os resultados para os orquestradores, que os armazenam na base de dados.
- **Saída de Dados:** Os agentes de previsão utilizam os dados armazenados para gerar previsões, que são enviadas para os utilizadores através de notificações.

### 2.3.2 Filas de Mensagens (*Queues*)

Para garantir a escalabilidade e a fiabilidade do sistema, são utilizadas *queues* para gerir a comunicação entre os componentes. Estas permitem:

- **Armazenamento Temporário:** Os dados são armazenados temporariamente até serem processados.
- **Gestão de Prioridades:** As mensagens podem ser processadas com base na sua prioridade.
- **Resiliência:** Em caso de falha de um componente, as mensagens permanecem na fila até que o componente esteja disponível novamente.

## 2.4 Servidor

O servidor é responsável por gerir a comunicação entre os diferentes componentes do sistema e fornecer uma interface para interação com os utilizadores. Foi implementado utilizando o framework Flask, que permite a criação de APIs *RESTful* para expor as funcionalidades do sistema. O servidor desempenha as seguintes funções principais:

- **Gestão de Dados:** Permite que os utilizadores acessem aos preços dos criptoativos e das previsões feitas por modelos sobre várias criptomoedas.
- **Notificações:** Envia alertas personalizados aos utilizadores com base nos critérios definidos, como *thresholds* de preços ou previsões de modelos.
- **Autenticação:** Implementa mecanismos de autenticação para garantir que apenas utilizadores autorizados podem aceder a certas funcionalidades do sistema, como o envio de emails com notificações de preços.
- **Integração com o Frontend:** Serve como intermediário entre o frontend e os agentes do sistema, processando pedidos e retornando respostas formatadas.

O servidor foi desenvolvido para ser escalável e modular, permitindo a adição de novas rotas e funcionalidades conforme necessário.

## 2.5 Frontend

O *frontend* é a interface gráfica que permite aos utilizadores interagir com o sistema de forma intuitiva. Foi desenvolvido utilizando tecnologias modernas como React.js, garantindo uma experiência de utilizador fluida e responsiva. As principais funcionalidades do frontend incluem:

- **Visualização de Dados:** Apresenta gráficos e tabelas com informações recolhidas, como preços de criptomoedas, notícias e análises de sentimento.
- **Gestão de Notificações:** Permite que os utilizadores configurem alertas personalizados para preços ou previsões de criptomoedas.

- **Autenticação e Gestão de Contas:** Inclui funcionalidades para registo, login e gestão de perfis de utilizadores.
- **Interação em Tempo Real:** Utiliza *WebSockets* para atualizar os dados apresentados em tempo real, garantindo que os utilizadores têm acesso às informações mais recentes.

O *frontend* foi projetado para ser modular e extensível, permitindo a integração de novas funcionalidades e a personalização da interface de acordo com as necessidades dos utilizadores.

## 2.6 Base de Dados e Armazenamento

O sistema utiliza uma base de dados MongoDB para armazenar os dados recolhidos e processados. A estrutura da base de dados é projetada para suportar consultas rápidas e eficientes.

### 2.6.1 Estrutura da Base de Dados

O sistema utiliza duas bases de dados principais, cada uma com várias coleções que armazenam diferentes tipos de informações. Abaixo, detalhamos as bases de dados e as respetivas coleções:

A base de dados ASM é utilizada para armazenar os dados recolhidos e processados pelo sistema. Esta inclui as seguintes coleções:

- **Coleção forum:** Armazena dados recolhidos de fóruns, como o *CryptoPanic*. Após a recolha inicial, os documentos nesta coleção são atualizados para incluir os resultados da análise de sentimento e a identificação de moedas mencionadas.
- **Coleção reddit:** Contém publicações extraídas do *Reddit*. Tal como na coleção *forum*, os documentos são enriquecidos com os resultados da análise de sentimento e identificação de moedas.
- **Coleção articles:** Armazena artigos de notícias recolhidos de várias fontes. Estes artigos também são processados para incluir os resultados da análise de sentimento e identificação de moedas.
- **Coleção crypto-price:** Regista preços das criptomoedas em tempo real.
- **Coleção detailed-crypto-data:** Contém dados diários detalhados, como preço de abertura, preço de fecho, volume de transações e variações diárias.
- **Coleção crypto-fear-greed:** Armazena o índice de medo e ganância, que mede o sentimento geral do mercado.
- **Coleção predictions:** Contém os resultados das previsões geradas pelos modelos de aprendizagem automática.

Já a base de dados ASM-Users é utilizada para gerir informações relacionadas com os utilizadores e as suas interações com o sistema. As coleções incluem:

- **Coleção users:** Armazena informações sobre os utilizadores registados no sistema, como credenciais e preferências.
- **Coleção notifications:** Contém notificações configuradas pelos utilizadores, incluindo alertas personalizados para preços de criptomoedas ou previsões de modelos.

### 2.6.2 Operações de Armazenamento

- **Upserts:** Utilizados para evitar duplicação de dados, garantindo que apenas novos dados sejam inseridos ou que os dados existentes sejam atualizados com as informações mais recentes.
- **Updates:** As operações de atualização são realizadas para modificar documentos existentes na base de dados, assegurando que os dados armazenados estão sempre atualizados.
- **Consultas:** As consultas são realizadas diretamente sobre as coleções garantindo que os dados necessários são recuperados de forma precisa e eficiente.

## 2.7 Escalabilidade e Modularidade

A arquitetura foi projetada para ser escalável e modular, permitindo a integração de novos agentes, orquestradores e fontes de dados. Esta flexibilidade garante que o sistema possa evoluir para atender a novas necessidades e desafios.

### 2.7.1 Escalabilidade

- **Horizontal:** Novos agentes podem ser adicionados para lidar com volumes maiores de dados.
- **Vertical:** Os agentes existentes podem ser otimizados para melhorar o desempenho e podem ser implementados em containers separados ou máquinas virtuais distintas, especialmente em ambientes de produção, para garantir isolamento e escalabilidade.

### 2.7.2 Modularidade

Cada componente é independente, permitindo a substituição ou atualização de agentes sem afetar o restante do sistema.

Esta arquitetura hierárquica, combinando orquestradores, agentes especializados e pipelines de comunicação, proporciona uma solução robusta e eficiente para lidar com a complexidade do domínio das criptomoedas. Nos próximos capítulos, serão detalhados o funcionamento em concreto de cada um dos agentes, os resultados obtidos e as análises realizadas com base nesta infraestrutura.



## 2.8 Diagramas

Abaixo, apresentamos vários diagramas que ilustra a arquitetura do sistema desenvolvido, destacando os principais componentes e as interações entre eles.

### 2.8.1 Diagrama de Fluxo do Sistema

O diagrama a seguir apresenta a arquitetura geral do sistema multiagente, destacando os principais componentes e as interações entre eles. Ele ilustra como os orquestradores, agentes especializados e outros módulos se conectam para formar uma solução integrada e escalável.

- **Orquestradores:** Representados como os principais coordenadores do sistema, responsáveis por gerenciar os agentes especializados.
- **Agentes Especializados:** Cada agente é responsável por uma tarefa específica, como coleta de dados, análise de sentimento ou previsão de preços.
- **Fluxo de Comunicação:** As setas indicam o fluxo de dados e mensagens entre os componentes, destacando a modularidade e a separação de responsabilidades.

Este diagrama fornece uma visão geral do sistema, permitindo entender como os diferentes módulos interagem para alcançar os objetivos do sistema.

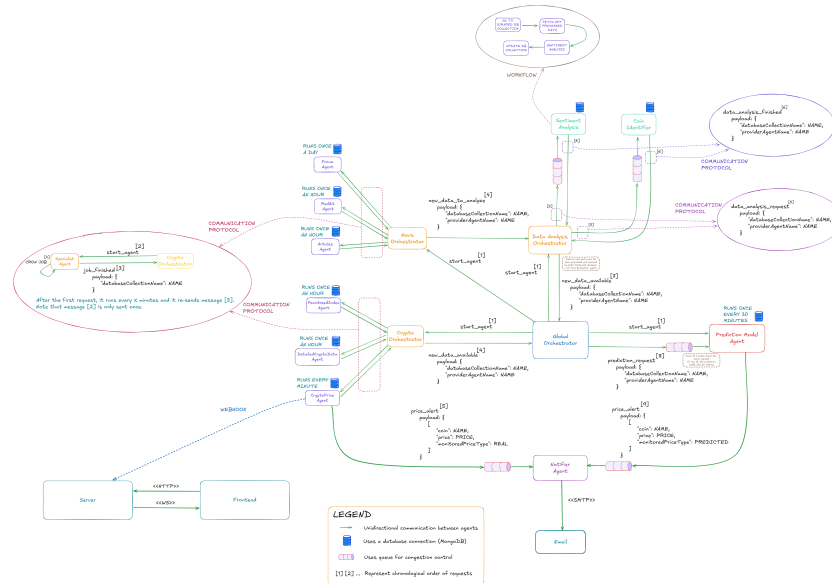


Fig. 1: Arquitetura do Sistema

### 2.8.2 Diagrama de Atividade

O diagrama de atividade ilustra o fluxo das operações dentro do sistema, destacando as principais etapas, decisões e interações entre os componentes. Ele representa uma única passagem pelos agentes, ignorando a recorrência de processos para simplificar a visualização.

- **Notificação de Orquestradores:** O processo começa com a notificação dos orquestradores, que iniciam os agentes especializados.
- **Coleta de Dados:** Os agentes de coleta obtêm informações de diferentes fontes, como fóruns, redes sociais e notícias.
- **Análise de Dados:** Os dados coletados são processados por agentes de análise, que realizam tarefas como análise de sentimento e extração de informações específicas.
- **Previsão e Notificação:** Os resultados das análises são usados para gerar previsões, que são enviadas aos usuários por meio de notificações.

Este diagrama ajuda a visualizar o fluxo de trabalho do sistema, desde a coleta de dados até a entrega de informações úteis aos usuários.

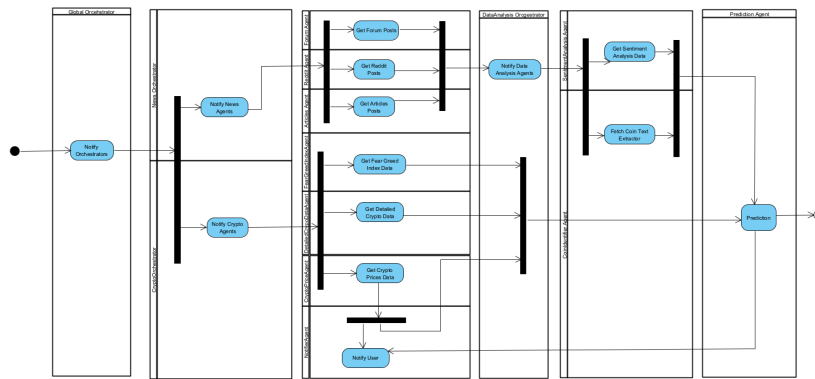


Fig. 2: Diagrama de Atividade

## 3 Funcionamento

A seção a seguir explora o funcionamento interno do sistema multiagente. Para facilitar a compreensão, apresentamos dois diagramas que ilustram diferentes aspectos do sistema: um diagrama de classes e um diagrama de colaboração. Ambos são fundamentais para entender como os componentes interagem e contribuem para o comportamento global do sistema.

O diagrama de classes representa a arquitetura lógica do sistema, destacando as principais classes, seus atributos, métodos e os relacionamentos entre elas. Ele fornece uma visão estática do sistema, mostrando a forma como os diferentes componentes

estão estruturados e como se relacionam, tudo isto será referido de seguida, quando entrarmos em detalhe acerca de cada classe.

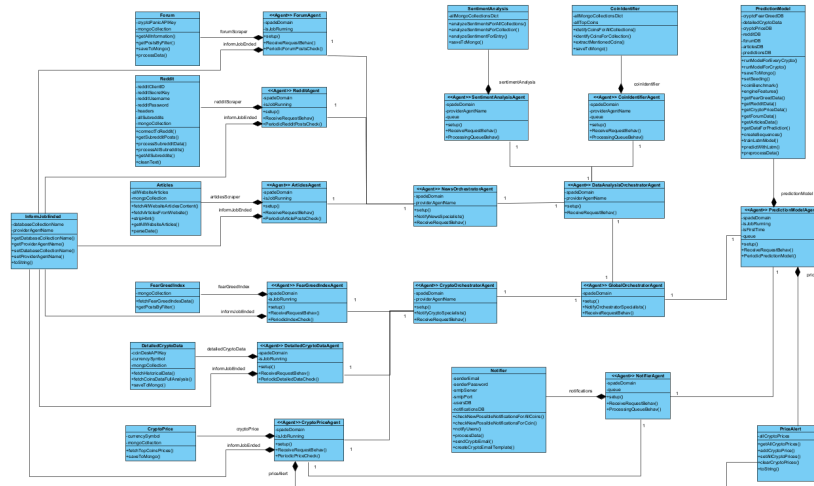


Fig. 3: Diagrama de Classes

O diagrama de colaboração detalha as interações dinâmicas entre os agentes e orquestradores do sistema. Ele destaca o fluxo de mensagens e a sequência de operações realizadas, permitindo uma visão clara de como os componentes trabalham juntos para atingir os objetivos do sistema. Neste diagrama, são claramente identificadas as mensagens enviadas entre todos os agentes, assim como a sua ordem de ocorrência. Todas as performatives aqui representadas serão referidas em cada agente abaixo.

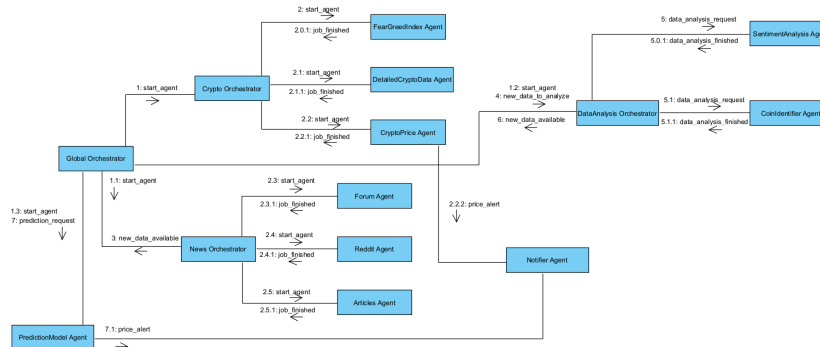


Fig. 4: Diagrama de Colaboração

Na sequência, e tendo em mente estes os diagramas acima expostos, vamos descrever cada agente com mais detalhe, explicando suas responsabilidades e como contribuem para o funcionamento geral do sistema.

### 3.1 Orquestradores

Nesta seção, apresentamos o funcionamento de cada orquestrador, as suas responsabilidades e o seu papel no fluxo geral do sistema. Todos os orquestradores são agentes que apenas têm como função coordenar o fluxo de dados e garantir o bom funcionamento do programa.

#### 3.1.1 Orquestrador Global (*Global Orchestrator*)

Coordena os orquestradores especializados e integra as funcionalidades do sistema. Ele inicia os agentes *Crypto Orchestrator*, *News Orchestrator*, *Data Analysis Orchestrator* e *Prediction Model Agent*. Após iniciar os agentes, aguarda mensagens dos orquestradores especializados, valida os dados recebidos e verifica se contêm as informações necessárias, como o nome da coleção na base de dados e o agente que forneceu os dados. Caso os dados sejam válidos, encaminha-os ao *Prediction Model Agent* para análise e previsão.

- **Performatives:**

- ▶ `start_agent`: Enviado para iniciar todos os orquestradores especializados.
- ▶ `new_data_available`: Recebido de orquestradores especializados para notificar que novos dados estão disponíveis para previsão. É um objeto que contém a coleção da base de dados referente às novas atualizações e ao agente orquestrador que encaminhou os dados.
- ▶ `prediction_request`: Enviado ao agente de previsão para iniciar a análise contendo um objeto com a coleção da base de dados e nome do agente.

- **Behaviors:**

- ▶ `NotifyOrchestratorSpecialists (OneShotBehaviour)`: Envia mensagens para iniciar os orquestradores especializados.
- ▶ `ReceiveRequestBehav (CyclicBehaviour)`: Processa mensagens recebidas, como notificações de novos dados ou pedidos de previsão.

A seguir, apresentamos um diagrama de sequência que ilustra como o Orquestrador Global interage com o sistema. Este diagrama detalha as mensagens trocadas e a ordem das operações realizadas. Para simplificar, o diagrama foca em um fluxo específico: a coordenação do receber preços de criptomoedas em tempo real, a notificação dos utilizadores e a realização de previsões com os novos dados.

O diagrama de sequência representa a interação entre os agentes no fluxo de trabalho descrito:

1. **Início do Processo:**

- O *Global Orchestrator* envia a mensagem `start_agent` ao *Crypto Orchestrator* para iniciar o processo de recolha de preços.
- O *Crypto Orchestrator*, por sua vez, inicia o *Crypto Price Agent*.

## 2. Recolha de Dados:

- O *Crypto Price Agent* executa a tarefa de recolher os preços das criptomoedas em tempo real, utilizando a API do CoinGecko.

## 3. Notificação de Dados Disponíveis:

- Após concluir a recolha, o *Crypto Price Agent* envia a mensagem `job_finished` ao *Crypto Orchestrator*, que notifica o *Global Orchestrator* com a mensagem `new_data_available`.

## 4. Previsão e Notificação:

- O *Global Orchestrator* envia a mensagem `prediction_request` ao *Prediction Model Agent* para realizar uma nova previsão com os dados atualizados.
- Simultaneamente, o *Crypto Price Agent* envia a mensagem `price_alert` ao *Notifier Agent*, que notifica os utilizadores sobre os preços atualizados.

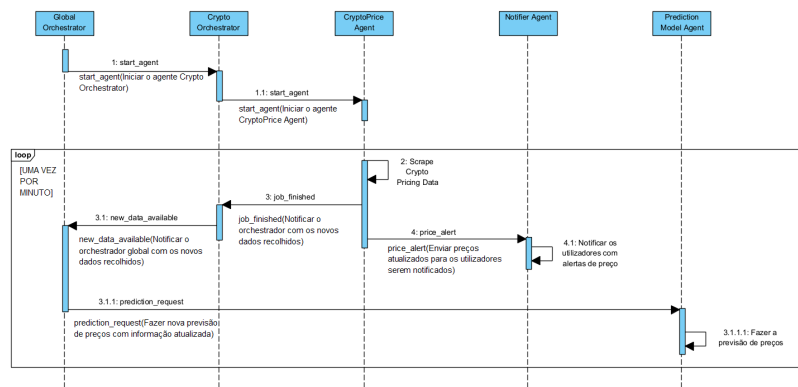


Fig. 5: Diagrama de Sequência

Note que apresentamos apenas o diagrama para este fluxo, pois, para os demais, a lógica é exatamente a mesma: *scrape* de dados, processamento e notificação ao *Prediction Model Agent*.

### 3.1.2 Orquestrador de Criptomoedas (*Crypto Orchestrator*)

Responsável por coordenar os agentes especializados na recolha de dados financeiros relacionados com criptomoedas. Ele inicia os agentes *Crypto Price Agent*, *Detailed Crypto Data Agent* e *Fear Greed Index Agent*, garantindo que cada um execute suas tarefas de forma eficiente. Após receber notificações de conclusão de tarefas dos agentes especializados, o *Crypto Orchestrator* valida os dados recolhidos e encaminha-os ao *Global Orchestrator* para serem futuramente encaminhados para novas previsões de preços.

#### • Performatives:

- `start_agent`: Enviado para iniciar agentes de recolha de dados financeiros.

- ▶ `job_finished`: Recebido dos agentes de recolha para indicar conclusão de tarefas. A comunicação é feita através de um objeto contendo a coleção da base de dados atualizada.
  - ▶ `new_data_available`: Enviado ao *Global Orchestrator* com os dados recolhidos, contendo a coleção da base de dados e o nome do agente orquestrador que fez o reencaminhamento.
- **Behaviors:**
- ▶ `NotifyCryptoSpecialists` (*OneShotBehaviour*): Notifica os agentes especializados em criptomoedas para iniciar tarefas.
  - ▶ `ReceiveRequestBehav` (*CyclicBehaviour*): Processa mensagens de ativação inicial assim como de conclusão de tarefas e redireciona os dados ao *Global Orchestrator*.

### 3.1.3 Orquestrador de Notícias (*News Orchestrator*)

Coordena os agentes especializados na recolha de dados de notícias e redes sociais relacionadas com criptomoedas. Ele inicia os agentes *Articles Agent*, *Forum Agent* e *Reddit Agent*, garantindo que cada um execute suas tarefas de forma eficiente. Após receber notificações de conclusão de tarefas dos agentes especializados, o *News Orchestrator* valida os dados recolhidos e encaminha-os ao *Data Analysis Orchestrator* para ser efetuada a análise de sentimento e identificação de moedas mencionadas nas notícias.

- **Performatives:**
- ▶ `start_agent`: Enviado para iniciar agentes de recolha de dados de notícias e redes sociais.
  - ▶ `job_finished`: Recebido dos agentes de recolha para indicar conclusão de tarefas. A comunicação é feita através de um objeto contendo a coleção da base de dados atualizada.
  - ▶ `new_data_to_analyze`: Enviado ao *Data Analysis Orchestrator* com os dados recolhidos, contendo a coleção da base de dados e o nome do agente orquestrador que fez o reencaminhamento.
- **Behaviors:**
- ▶ `NotifyNewsSpecialists` (*OneShotBehaviour*): Notifica os agentes especializados em notícias e redes sociais para iniciar tarefas.
  - ▶ `ReceiveRequestBehav` (*CyclicBehaviour*): Processa mensagens de ativação inicial assim como de conclusão de tarefas e redireciona os dados ao *Data Analysis Orchestrator*.

### 3.1.4 Orquestrador de Análise de Dados (*Data Analysis Orchestrator*)

Responsável por coordenar os agentes especializados na análise de dados recolhidos, como a análise de sentimento e a identificação de moedas mencionadas em textos.

Assim que recebe um pedido, reencaminha-o para os agentes *Sentiment Analysis Agent* e *Coin Identifier Agent*, garantindo que cada um execute suas tarefas de forma eficiente. Após receber notificações de conclusão de tarefas dos agentes especializados, o *Data Analysis Orchestrator* valida os resultados e apenas encaminha os dados ao *Global Orchestrator* quando ambos os agentes especializados tiverem concluído o mesmo processo para a mesma coleção de dados. Este mecanismo assegura que os dados enviados ao *Global Orchestrator* estejam completos e prontos para serem utilizados em previsões e futuras notificações.

- **Performatives:**

- ▶ *start\_agent*: Enviado para iniciar o orquestrador e prepará-lo para lidar com dados de análise.
- ▶ *new\_data\_to\_analyze*: Recebido do *News Orchestrator* para iniciar a análise de dados com novas notícias.
- ▶ *data\_analysis\_request*: Enviado aos agentes especializados (*Sentiment Analysis Agent* e *Coin Identifier Agent*) para realizarem as suas análises.
- ▶ *data\_analysis\_finished*: Recebido dos agentes especializados para indicar que a análise foi concluída.
- ▶ *new\_data\_available*: Enviado ao *Global Orchestrator* com os resultados da análise, contendo a coleção da base de dados e o nome do agente orquestrador que fez o reencaminhamento.

- **Behaviors:**

- ▶ *ReceiveRequestBehav* (*CyclicBehaviour*): Processa mensagens de ativação inicial, solicita análises aos agentes especializados, recebe a resposta e redireciona os resultados ao *Global Orchestrator*.

## 3.2 Agentes Especializados

Nesta seção, descrevemos os agentes especializados, destacando suas funções específicas e como complementam as atividades dos orquestradores para garantir o funcionamento eficiente do sistema.

### 3.2.1 Agente de Preços de Criptomoedas (*Crypto Price Agent*)

Tem como função recolher informações sobre os preços das criptomoedas em intervalos regulares e comunicar os resultados aos agentes relevantes, como o *Crypto Orchestrator* e o *Notifier Agent*. Quando iniciado, o agente executa um comportamento periódico que recolhe informações a cada minuto, utilizando a API do site CoinGecko. Este processo permite obter dados detalhados sobre as criptomoedas, incluindo o nome da moeda, o preço em dólares (USD) e a data exata do momento de extração. Os dados recolhidos são armazenados na coleção *crypto-price* da base de dados MongoDB, garantindo que as informações estejam sempre atualizadas e disponíveis para análise

e notificações. Além disso, o agente envia notificações aos agentes relevantes para indicar a conclusão das tarefas e disponibilizar os dados recolhidos.

- **Performatives:**

- `start_agent`: Recebido para iniciar o processo de recolha de preços.
- `job_finished`: Enviado ao *Crypto Orchestrator* para indicar que a tarefa foi concluída. Inclui informações sobre a coleção da base de dados atualizada.
- `price_alert`: Enviado ao *Notifier Agent* com as informações de novos preços recolhidos em tempo real.

- **Behaviors:**

- `ReceiveRequestBehav` (*CyclicBehaviour*): Processa mensagens recebidas e inicia o comportamento periódico de recolha de preços.
- `PeriodicPriceCheck` (*PeriodicBehaviour*): Executa a recolha de preços das criptomoedas em intervalos regulares (uma vez por minuto) e envia os resultados aos agentes relevantes.

### 3.2.2 Agente de Informações de Criptomoedas (*Detailed Crypto Data Agent*)

Recolhe dados históricos detalhados das criptomoedas, incluindo valores diários de abertura, máximo, mínimo, fecho e volume transacionado. Este agente utiliza a API do CoinDesk para obter informações sobre as principais criptomoedas, com um limite de até 365 dias de histórico. Os dados recolhidos são armazenados na coleção `detailed-crypto-data` da base de dados MongoDB, garantindo que as informações estejam disponíveis para análises futuras. Quando iniciado, o agente executa um comportamento periódico que recolhe os dados a cada hora, processa-os e notifica o *CryptoOrchestrator* sobre a conclusão da tarefa.

- **Performatives:**

- `start_agent`: Recebido para iniciar o processo de recolha de dados históricos.
- `job_finished`: Enviado ao *Crypto Orchestrator* para indicar que a tarefa foi concluída. Inclui informações sobre a coleção da base de dados atualizada.

- **Behaviors:**

- `ReceiveRequestBehav` (*CyclicBehaviour*): Processa mensagens recebidas e inicia o comportamento periódico de recolha de dados históricos.
- `PeriodicDetailedDataCheck` (*PeriodicBehaviour*): Executa a recolha de dados históricos das criptomoedas em intervalos regulares (uma vez por hora) e envia os resultados ao *Crypto Orchestrator*.

### 3.2.3 Agente do Índice de Medo e Ganância (*Fear Greed Index Agent*)

Tem como principal objetivo recolher o índice de medo e ganância do mercado de criptomoedas, um indicador que mede o sentimento geral do mercado com base em fatores como volatilidade, volume de transações e tendências sociais. Este agente



utiliza uma API externa do Alternative.me para obter o índice atualizado e armazena os dados recolhidos na coleção `crypto-fear-greed` da base de dados MongoDB. Quando iniciado, o agente executa um comportamento periódico que recolhe o índice a cada hora, processa os dados e notifica o *Crypto Orchestrator* sobre a conclusão da tarefa.

- **Performatives:**

- ▶ `start_agent`: Recebido para iniciar o processo de recolha do índice de medo e ganância.
- ▶ `job_finished`: Enviado ao *Crypto Orchestrator* para indicar que a tarefa foi concluída. Inclui informações sobre a coleção da base de dados atualizada.

- **Behaviors:**

- ▶ `ReceiveRequestBehav` (`CyclicBehaviour`): Processa mensagens recebidas e inicia o comportamento periódico de recolha do índice.
- ▶ `PeriodicFearGreedCheck` (`PeriodicBehaviour`): Executa a recolha do índice de medo e ganância em intervalos regulares (uma vez por hora) e envia os resultados ao *Crypto Orchestrator*.

### 3.2.4 Agente de Artigos (*Articles Agent*)

Responsável por recolher artigos de notícias relacionados com criptomoedas de várias fontes confiáveis. Este agente utiliza um *scraper* para obter o conteúdo completo dos artigos, incluindo o título, a data de publicação, o autor e o texto principal. Os dados recolhidos são armazenados na coleção `articles` da base de dados MongoDB, garantindo que as informações estejam disponíveis para análises futuras, como análise de sentimento e identificação de moedas mencionadas. Quando iniciado, o agente executa um comportamento periódico que recolhe os artigos a cada hora, processa os dados e notifica o *News Orchestrator* sobre a conclusão da sua tarefa.

- **Performatives:**

- ▶ `start_agent`: Recebido para iniciar o processo de recolha de artigos.
- ▶ `job_finished`: Enviado ao *News Orchestrator* para indicar que a tarefa foi concluída. Inclui informações sobre a coleção da base de dados atualizada.

- **Behaviors:**

- ▶ `ReceiveRequestBehav` (`CyclicBehaviour`): Processa mensagens recebidas e inicia o comportamento periódico de recolha de artigos.
- ▶ `PeriodicArticlePostsCheck` (`PeriodicBehaviour`): Executa a recolha de artigos de notícias em intervalos regulares (uma vez por hora) e envia os resultados ao *News Orchestrator*.

### 3.2.5 Agente do Reddit (*Reddit Agent*)

O *Reddit Agent* é responsável por recolher publicações do Reddit relacionadas com criptomoedas, analisando subreddits específicos para identificar tendências e opiniões.

Este agente utiliza a API do Reddit para obter informações detalhadas, como o título da publicação, o autor, a data de publicação e o conteúdo. São recolhidos os *posts* mais recentes e os *posts* que se encontram nas tendências. Os dados recolhidos são armazenados na coleção *reddit* da base de dados MongoDB, garantindo que estejam disponíveis para análises futuras, como análise de sentimento e identificação de moedas mencionadas. Quando iniciado, o agente executa um comportamento periódico que recolhe as publicações a cada hora, processa os dados e notifica o *News Orchestrator* sobre a conclusão da tarefa.

- **Performatives:**

- ▶ *start\_agent*: Recebido para iniciar o processo de recolha de publicações do Reddit.
- ▶ *job\_finished*: Enviado ao *News Orchestrator* para indicar que a tarefa foi concluída. Inclui informações sobre a coleção da base de dados atualizada.

- **Behaviors:**

- ▶ *ReceiveRequestBehav* (*CyclicBehaviour*): Processa mensagens recebidas e inicia o comportamento periódico de recolha de publicações.
- ▶ *PeriodicRedditPostsCheck* (*PeriodicBehaviour*): Executa a recolha de publicações do *Reddit* em intervalos regulares (uma vez por hora) e envia os resultados ao *News Orchestrator*.

### 3.2.6 Agente do *Forum* (*Forum Agent*)

Recolhe dados de fóruns relacionados com criptomoedas, como discussões e notícias partilhadas por utilizadores. Este agente utiliza a API do *CryptoPanic* para obter informações detalhadas, incluindo o título da publicação, o autor, a data de publicação e o conteúdo. Os dados recolhidos são armazenados na coleção *forum* da base de dados MongoDB, garantindo que estejam disponíveis para análises futuras, como análise de sentimento e identificação de moedas mencionadas. Quando iniciado, o agente executa um comportamento periódico que recolhe as publicações a cada hora, processa os dados e notifica o *News Orchestrator* sobre a conclusão da tarefa.

- **Performatives:**

- ▶ *start\_agent*: Recebido para iniciar o processo de recolha de publicações de fóruns.
- ▶ *job\_finished*: Enviado ao *News Orchestrator* para indicar que a tarefa foi concluída. Inclui informações sobre a coleção da base de dados atualizada.

- **Behaviors:**

- ▶ *ReceiveRequestBehav* (*CyclicBehaviour*): Processa mensagens recebidas e inicia o comportamento periódico de recolha de publicações.
- ▶ *PeriodicForumPostsCheck* (*PeriodicBehaviour*): Executa a recolha de publicações de fóruns em intervalos regulares (uma vez por hora) e envia os resultados ao *News Orchestrator*.

### 3.2.7 Agente de Análise de Sentimento (*Sentiment Analysis Agent*)

O *Sentiment Analysis Agent* é responsável por realizar a análise de sentimento em dados textuais recolhidos de várias fontes. Este agente utiliza a biblioteca NLTK, especificamente o modelo VADER (*Valence Aware Dictionary and sEntiment Reasoner*), para atribuir uma pontuação de sentimento a cada entrada de texto. Os resultados incluem uma pontuação composta (de -1 a 1) e uma classificação de sentimento (Positivo, Neutro ou Negativo) atribuídos com base no espectro definido acima. Os dados analisados são armazenados na mesma coleção da base de dados de onde foram extraídos, enriquecendo os documentos originais com os resultados da análise de sentimento. Para evitar congestionamento, o agente utiliza uma *queue* para controlar o fluxo de mensagens e processar os dados de forma eficiente. Isto pois, caso existam muitos pedidos em simultâneo o agente não fique condicionado, deste modo, apenas faz um pedido de cada vez num ambiente controlado.

- **Performatives:**

- `data_analysis_request`: Recebido do *Data Analysis Orchestrator* para iniciar a análise de sentimento em uma coleção de dados.
- `data_analysis_finished`: Enviado ao *Data Analysis Orchestrator* para indicar que a análise foi concluída. Inclui informações sobre a coleção da base de dados atualizada.

- **Behaviors:**

- `ReceiveRequestBehav` (*CyclicBehaviour*): Recebe mensagens de solicitação de análise e adiciona os dados à *queue* para processamento.
- `ProcessingQueueBehav` (*CyclicBehaviour*): Processa os dados na *queue*, realiza a análise de sentimento e envia os resultados ao *Data Analysis Orchestrator*.

### 3.2.8 Agente de Identificador de Criptomoedas (*Coin Identifier Agent*)

Identifica as criptomoedas mencionadas em textos recolhidos de várias fontes. Este agente utiliza um mapeamento de palavras-chave para associar nomes ou símbolos de criptomoedas aos seus identificadores únicos. No final será criada uma lista contendo todas as criptomoedas que estão presentes no determinado texto. Os resultados da identificação são armazenados na mesma coleção da base de dados de onde os textos foram extraídos, enriquecendo também, os documentos originais com as informações das criptomoedas mencionadas. Para evitar congestionamento, o agente utiliza também uma *queue* para controlar o fluxo de mensagens e processar os dados de forma eficiente.

- **Performatives:**

- `data_analysis_request`: Recebido do *Data Analysis Orchestrator* para iniciar a identificação de criptomoedas em uma coleção de dados.
- `data_analysis_finished`: Enviado ao *Data Analysis Orchestrator* para indicar que a identificação foi concluída. Inclui informações sobre a coleção da base de dados atualizada.

- **Behaviors:**

- `ReceiveRequestBehav` (`CyclicBehaviour`): Recebe mensagens de solicitação de identificação e adiciona os dados à *queue* para processamento.
- `ProcessingQueueBehav` (`CyclicBehaviour`): Processa os dados na *queue*, realiza a identificação de criptomoedas e envia os resultados ao *Data Analysis Orchestrator*.

### 3.2.9 Agente do Modelo Preditivo (*Prediction Model Agent*)

Este agente é responsável por prever os preços futuros das criptomoedas com base em dados históricos de preços e notícias, assim como outros indicadores derivados. Este agente utiliza um modelo de redes neurais recorrentes (*Recurrent Neural Networks*, RNN), mais especificamente uma arquitetura LSTM (*Long Short-Term Memory*), que é amplamente reconhecida por sua capacidade de capturar dependências temporais em séries temporais. O modelo LSTM é composto por várias camadas, incluindo camadas de entrada, camadas LSTM ocultas e uma camada densa final para gerar as previsões. A escolha do LSTM é motivada pela sua eficácia em lidar com problemas de séries temporais, como a previsão de preços de criptomoedas, onde os dados apresentam padrões temporais complexos e não lineares.

O modelo é treinado utilizando dados históricos recolhidos de várias fontes, como preços diários, volumes de transação e indicadores técnicos. Além disso, o modelo incorpora dados de sentimento provenientes da análise de sentimento realizada por outros agentes, como o *Sentiment Analysis Agent*. Durante o treino, o modelo utiliza técnicas como *Backpropagation Through Time* (BPTT) para ajustar os pesos das camadas LSTM e minimizar o erro de previsão. Para avaliar o desempenho do modelo, são utilizados vários indicadores de *benchmarking*, como o erro médio absoluto (MAE), o erro quadrático médio (MSE) e o coeficiente de determinação ( $R^2$ ). Estes indicadores permitem avaliar a performance do modelo, de modo a saber, se são necessários ajustes futuros ou não.

Os resultados das previsões (próximos 7 dias) são armazenados na coleção *predictions* da base de dados e incluem informações como preços históricos, preços previstos e métricas de sentimento positivo. Além disso, o agente realiza uma análise de benchmarking para avaliar a precisão das previsões em relação aos dados reais. Este processo permite identificar tendências de mercado e fornecer informações úteis para os utilizadores. Ou seja, não só temos os resultados teóricos do treino do modelo, mas também os práticos de previsões de dias anteriores. O *Prediction Model Agent* também utiliza uma *queue* para gerenciar o fluxo de dados e garantir que as previsões sejam realizadas de forma eficiente, mesmo em cenários de alta carga de dados. Esta *queue* funciona de uma forma um pouco diferente, quando o intervalo de 30 minutos passar, o agente verifica se a queue tem algo, se não tiver, não faz nada e espera mais 30 minutos, caso contrário roda a previsão e esvazia a queue. Optamos por esta estratégia para não estar a sobrecarregar o modelo a fazer uma previsão por cada nova fonte de dados, devido aos recursos necessários para fazer as previsões.

- **Performatives:**

- `prediction_request`: Recebido do *Global Orchestrator* para iniciar o processo de previsão de preços.
- `prediction_finished`: Enviado ao *Global Orchestrator* para indicar que a previsão foi concluída. Inclui os resultados das previsões e métricas de avaliação.
- `price_alert`: Enviado ao *Notifier Agent* com as informações de novos preços recolhidos em tempo real.

- **Behaviors:**

- `ReceiveRequestBehav` (*CyclicBehaviour*): Recebe mensagens de solicitação de previsão e adiciona os dados à fila para processamento.
- `ProcessingQueueBehav` (*CyclicBehaviour*): Processa os dados na fila, realiza as previsões e notifica o *Notifier Agent* acerca das novas previsões.

### 3.2.10 Agente de Notificações (*Notifier Agent*)

O *Notifier Agent* é responsável por enviar notificações personalizadas aos utilizadores com base em condições predefinidas, como alterações significativas nos preços das criptomoedas. Este agente utiliza o protocolo SMTP (*Simple Mail Transfer Protocol*) para enviar emails, garantindo uma comunicação eficiente e segura com os utilizadores. As notificações incluem informações detalhadas, como o nome da criptomoeda, o preço atual, a condição de alerta (acima ou abaixo de um valor-alvo) e o seu tipo (preço real ou previsão). Para evitar congestionamento, o agente utiliza uma *queue* para gerenciar as notificações pendentes, processando-as de forma assíncrona e garantindo que cada utilizador receba as informações relevantes.

Quando uma notificação é gerada, o agente verifica as condições de alerta configuradas pelos utilizadores na base de dados MongoDB. Caso a condição seja atendida (por exemplo, o preço atual excede o valor-alvo), a notificação é adicionada à fila para envio. O comportamento `ProcessingQueueBehav` processa as notificações na fila, enviando emails aos utilizadores através do protocolo SMTP. Este fluxo garante que o agente opere de forma eficiente, mesmo em cenários de alta carga de notificações.

- **Performatives:**

- `price_alert`: Recebido para adicionar notificações à *queue* de envio.

- **Behaviors:**

- `ReceiveRequestBehav` (*CyclicBehaviour*): Recebe mensagens de solicitação de envio de notificações e adiciona-as à *queue*.
- `ProcessingQueueBehav` (*CyclicBehaviour*): Processa as notificações na *queue*, enviando emails aos utilizadores e atualizando o estado das notificações na base de dados.

### 3.3 Servidor

O servidor do sistema foi implementado utilizando o framework Flask, que fornece uma API *RESTful* para permitir a interação entre os utilizadores e os agentes do sistema. Ele expõe várias rotas para consulta de dados, configuração de notificações e visualização de previsões, além de utilizar *WebSockets* para fornecer atualizações em tempo real.

As rotas implementadas são:

- **GET**
  - `api/crypto/{coin}`: Retorna os dados mais recentes sobre a criptomoeda especificada.
  - `api/notification/{coin}`: Retorna as notificações configuradas para a criptomoeda especificada por um determinado utilizador.
  - `api/notification`: Retorna os detalhes de uma notificação específica.
- **POST**
  - `api/crypto/update`: Realiza um *broadcast* com *WebSockets* em tempo real do preço das moedas para o *frontend*.
  - `api/auth/login`: Permite que os utilizadores façam *login* no sistema.
  - `api/auth/register`: Permite que novos utilizadores se registem no sistema.
  - `api/notification/add`: Adiciona uma nova notificação para o utilizador especificado.
- **DELETE**
  - `api/notification`: Remove uma notificação específica.
- **PATCH**
  - `api/notification/edit`: Edita uma notificação existente.

Além das rotas *RESTful*, o servidor utiliza *WebSockets* para fornecer atualizações em tempo real. Por exemplo, quando o preço de uma criptomoeda varia, atualizamos o preço das moedas no *dashboard* do utilizador. Este mecanismo é particularmente útil para cenários onde a latência mínima é essencial, como no acompanhamento de preços em tempo real.

### 3.4 Frontend

O *frontend* inclui mecanismos avançados para lidar com grandes volumes de dados e garantir uma experiência de utilizador consistente. Ele utiliza técnicas de *lazy loading* para carregar apenas os dados necessários à medida que o utilizador navega pela aplicação, reduzindo o tempo de carregamento inicial e otimizando o desempenho.

Outro destaque do *frontend* é a integração com notificações em tempo real. Quando o *backend* envia atualizações via *WebSocket*, o *frontend* processa essas mensagens e atualiza automaticamente os componentes relevantes, como gráficos ou alertas de

preços. Isso elimina a necessidade de recarregar a página e garante que os utilizadores tenham acesso imediato às informações mais recentes. Além disso, o *frontend* foi projetado para ser acessível e simplista.

## 4 Resultados obtidos

Os resultados obtidos no desenvolvimento do sistema incluem a criação de uma interface gráfica funcional e intuitiva, composta por quatro páginas principais: Login, Registo, Gestão de Notificações e Painel de Análise. Cada uma dessas páginas foi projetada para atender a diferentes necessidades dos utilizadores, garantindo uma experiência fluida e eficiente.

### 4.1 Login

A página de *Login* é a porta de entrada para os utilizadores existentes no sistema, permitindo que acessem a algumas funcionalidades adicionais. A autenticação é realizada através de credenciais (email e senha), com validação no servidor de *backend* para garantir a segurança dos dados. Após o *login* bem-sucedido, o utilizador recebe um token de sessão que é utilizado para autenticar futuras interações com o sistema, como a configuração de notificações ou a visualização de previsões. De seguida, o utilizador é redirecionado automaticamente para a página de Painel de Análise.

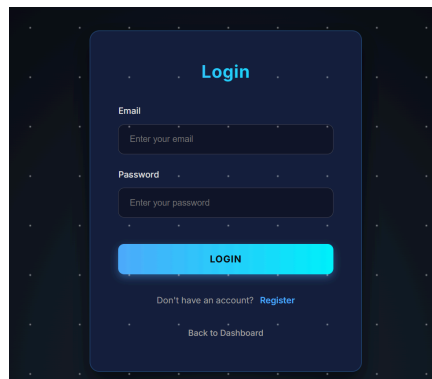


Fig. 6: Página de *Login* do Sistema

### 4.2 Registo

A página Registo permite que novos utilizadores se registem no sistema, criando uma conta para acessar todas as funcionalidades disponíveis. O processo de registo inclui

a introdução de informações básicas, como nome, email e senha, que são validadas no *frontend* antes de serem enviadas ao servidor de *backend*. Este realiza verificações adicionais, como garantir que o email não esteja já registrado, e armazena os dados do utilizador de forma segura na base de dados. Após o registo bem-sucedido, o utilizador é redirecionado automaticamente para a página de Painel de Análise.

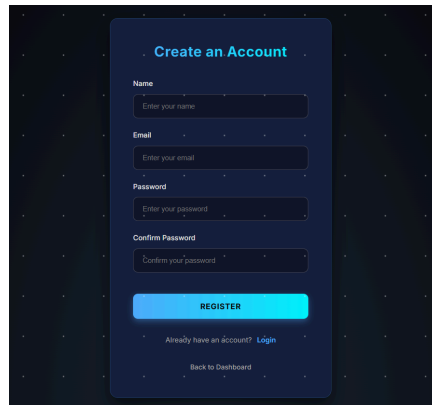
A screenshot of a web application's registration page. The page has a dark blue background with a subtle grid pattern. In the center, there is a white rectangular card titled 'Create an Account' in a bold, dark blue font. Below the title, there are four input fields: 'Name' with the placeholder 'Enter your name', 'Email' with 'Enter your email', 'Password' with 'Enter your password', and 'Confirm Password' with 'Confirm your password'. Each input field has a small blue eye icon on the right side. Below these fields is a prominent blue button with the text 'REGISTER' in white. At the bottom of the card, there is a link that says 'Already have an account? Login' and a link that says 'Back to Dashboard'.

Fig. 7: Página de Registo do Sistema

## 4.3 Gestão de Notificações

A página Gestão de Notificações foi desenvolvida para oferecer aos utilizadores um controle completo sobre os alertas configurados para preços de criptomoedas. Esta funcionalidade permite visualizar, adicionar, editar e remover notificações de forma simples e eficiente. A interface foi projetada para ser intuitiva, garantindo que os utilizadores possam configurar alertas personalizados e acompanhar alterações no mercado de forma proativa.

### 4.3.1 Visualizar Alertas

A funcionalidade de visualização de alertas apresenta uma lista de notificações configuradas, categorizadas como ativas ou inativas. A interface inclui filtros que permitem aos utilizadores organizar os alertas por estado (ativo ou inativo), tipo (tempo real ou previsão) e condição (acima ou abaixo de um valor). Cada alerta exibe informações detalhadas, como o tipo de notificação, a condição configurada e o preço-alvo. Esta funcionalidade garante que os utilizadores tenham uma visão clara e organizada de todos os alertas configurados.



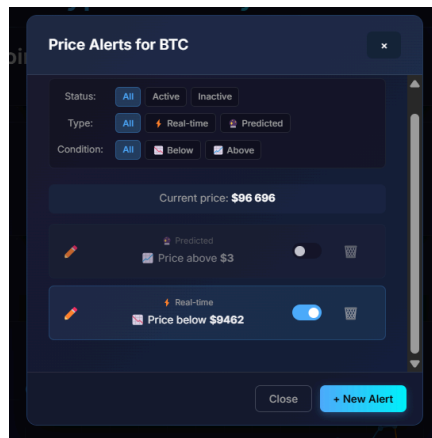


Fig. 8: Interface de Visualização de Alertas

#### 4.3.2 Adicionar Alertas

A funcionalidade de adicionar alertas permite que os utilizadores configurem notificações personalizadas para preços de criptomoedas. A interface apresenta opções para seleccionar o tipo de notificação (preço em tempo real ou previsão), a condição (acima ou abaixo de um valor) e o preço-alvo desejado. O preço actual da criptomoeda é exibido para ajudar os utilizadores a tomar decisões informadas. Após configurar os parâmetros, o utilizador pode guardar o alerta, que será monitorizado pelo sistema e exibido na lista de notificações.

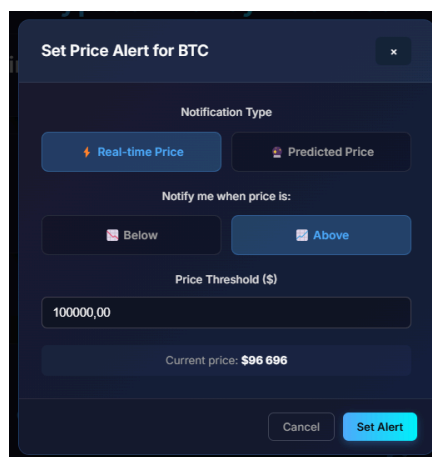
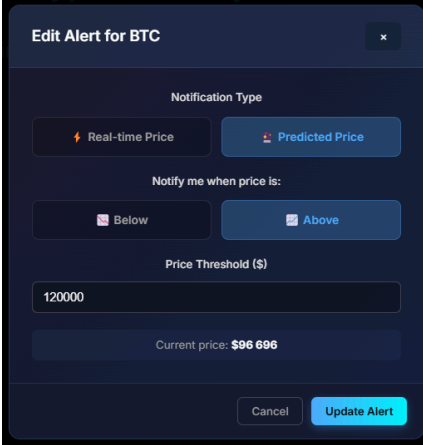


Fig. 9: Interface para Adicionar um Novo Alerta

#### 4.3.3 Editar Alertas

A funcionalidade de edição de alertas permite que os utilizadores modifiquem notificações existentes de forma rápida e eficiente. Os utilizadores podem alterar parâmetros como o tipo de notificação, a condição e o preço-alvo diretamente na interface. Após a edição, o alerta atualizado é guardado e monitorizado pelo sistema, garantindo que as alterações sejam refletidas imediatamente. Esta funcionalidade é essencial para que os utilizadores possam ajustar os seus alertas de acordo com as mudanças no mercado.



The image shows a dark-themed modal window titled "Edit Alert for BTC". Inside, there are three main sections: "Notification Type" with buttons for "Real-time Price" (lightning bolt icon) and "Predicted Price" (calendar icon); "Notify me when price is:" with buttons for "Below" (down arrow icon) and "Above" (up arrow icon); and a "Price Threshold (\$)" input field containing "120000". Below the input field, it says "Current price: \$96 696". At the bottom right, there are two buttons: "Cancel" and "Update Alert".

Fig. 10: Interface para Editar um Alerta Existente

#### 4.3.4 Receber um Alerta

Quando uma condição configurada em um alerta é atendida, o utilizador recebe uma notificação diretamente no email utilizado para criar a conta. O alerta contém informações detalhadas, como o nome da criptomoeda, o preço atual, a condição que foi atingida (acima ou abaixo do valor configurado) e o tipo de notificação (tempo real ou previsão). Este mecanismo garante que os utilizadores sejam informados de forma imediata e eficiente sobre alterações significativas no mercado, permitindo que tomem decisões rápidas e informadas.

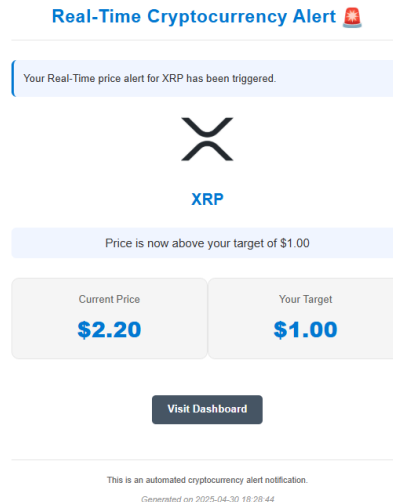


Fig. 11: Alerta Recebido por Email

## 4.4 Painel de Análise

A página Painel de Análise é o centro de visualização de dados do sistema, permitindo que os utilizadores acompanhem informações detalhadas sobre as criptomoedas em tempo real e as previsões geradas pelo modelo de aprendizagem automática. Esta página foi projetada para ser interativa e informativa, oferecendo uma visão clara e organizada dos dados recolhidos e processados pelo sistema.

Para facilitar a explicação, dividiremos a página em várias secções. Cada secção será descrita em detalhe, destacando as suas funcionalidades e como elas contribuem positivamente para a experiência do utilizador.

### 4.4.1 Seleção de Moeda e Preço em Tempo Real

O painel permite que os utilizadores selecionem a criptomoeda que desejam analisar através de um menu interativo localizado no canto superior esquerdo. Após a seleção, o preço atual da moeda é exibido em tempo real, utilizando dados recolhidos pelo sistema de agentes e atualizados por *WebSockets*. Esta funcionalidade garante que os utilizadores tenham acesso imediato às informações mais recentes do mercado, permitindo uma análise rápida e precisa.

Além do preço atual, o painel também exibe a variação percentual e a diferença absoluta no preço da criptomoeda, dependendo do intervalo de tempo selecionado.

Isso permite que os utilizadores compreendam rapidamente as tendências de curto e longo prazo, ajudando na tomada de decisões informadas.

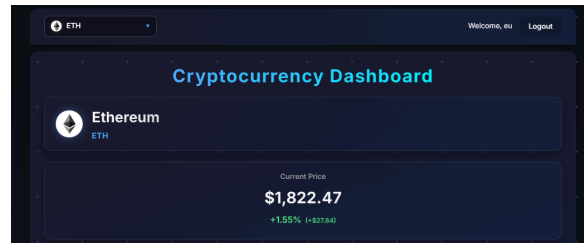


Fig. 12: Interface de Seleção de Moeda e Preço em Tempo Real

#### 4.4.2 Previsão de Preços

A funcionalidade de previsão de preços apresenta um gráfico interativo que combina dados históricos, preços previstos e análises de sentimento. O gráfico exibe várias métricas, incluindo o preço histórico da criptomoeda, o preço inicial, o preço previsto pelo modelo LSTM, a razão de sentimento e o sentimento neutro. Esta visualização permite que os utilizadores compreendam as tendências passadas e futuras da criptomoeda, correlacionando os preços com os sentimentos do mercado.

Além do gráfico, a interface apresenta previsões detalhadas para o próximo dia e para os próximos sete dias. Estas previsões incluem o preço esperado, a variação percentual e a diferença absoluta em relação ao preço atual. Esta combinação de informações ajuda os utilizadores a tomar decisões informadas, fornecendo uma visão clara das tendências futuras do mercado.

A interface também oferece elementos interativos que enriquecem a experiência do utilizador. No canto superior direito, encontra-se o botão de alertas, que permite configurar notificações personalizadas diretamente a partir do painel de análise. Adicionalmente, os utilizadores podem alternar entre diferentes intervalos de tempo, como 7 dias, 1 mês ou 1 ano, ajustando a visualização do gráfico para explorar tendências em diferentes escalas temporais.

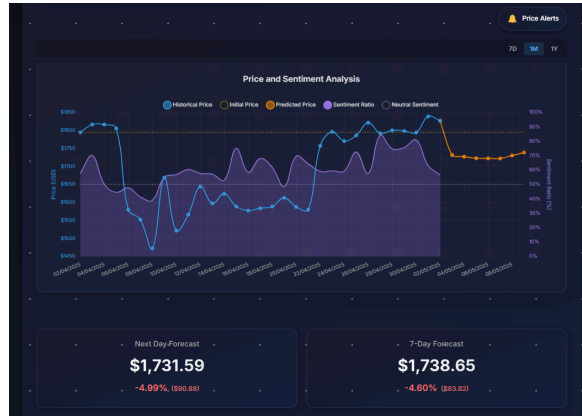


Fig. 13: Gráfico de Previsão de Preços e Análise de Sentimento

#### 4.4.3 Avaliação do Modelo

A secção de avaliação do modelo apresenta uma análise detalhada do desempenho do modelo de previsão, incluindo previsões passadas e métricas de avaliação. Esta secção foi projetada para fornecer transparência sobre a precisão do modelo e ajudar os utilizadores a compreenderem a confiabilidade das previsões.

A tabela de previsões passadas exhibe os preços previstos pelo modelo em comparação com os preços reais, juntamente com a diferença absoluta e percentual entre eles. Esta informação permite que os utilizadores avaliem a precisão do modelo em diferentes datas e identifiquem padrões ou desvios significativos.

Além disso, a secção inclui métricas de desempenho do modelo, como o erro absoluto médio (MAE), o erro percentual absoluto médio (MAPE), o erro quadrático médio (MSE), a raiz do erro quadrático médio (RMSE) e o coeficiente de determinação ( $R^2$ ). Estas métricas fornecem uma visão quantitativa da eficácia do modelo em prever os preços das criptomoedas.

Por fim, a interface também exhibe a data e hora em que a última avaliação do modelo foi realizada, garantindo que os utilizadores saibam quando os dados foram atualizados pela última vez.



Fig. 14: Avaliação do Modelo com Previsões Passadas e Métricas de Desempenho

5 Análise Crítica do Trabalho

O desenvolvimento deste sistema multiagente para análise de criptomoedas representa um avanço significativo na aplicação de tecnologias de inteligência artificial e sistemas distribuídos em mercados financeiros. No entanto, como qualquer projeto complexo, existem pontos fortes e limitações que merecem ser analisados criticamente.

5.1 Pontos Fortes

- 1. **Arquitetura Modular e Escalável:** A arquitetura hierárquica do sistema, composta por orquestradores e agentes especializados, demonstrou ser altamente modular e escalável. A separação clara de responsabilidades entre os agentes permitiu uma implementação eficiente e facilitou a integração de novas funcionalidades, como a adição de novos agentes ou fontes de dados.
- 2. **Integração de Dados em Tempo Real:** A utilização de *WebSockets* para atualizações em tempo real garantiu que os utilizadores tivessem acesso imediato às informações mais recentes do mercado. Este mecanismo foi particularmente útil para acompanhar preços de criptomoedas e notificações de alertas.
- 3. **Análise de Sentimento e Previsão de Preços:** A integração de técnicas de análise de sentimento e modelos de aprendizagem automática, como LSTM, permitiu uma análise aprofundada dos dados recolhidos. Os resultados das previsões demonstraram ser úteis para identificar tendências de mercado e fornecer informações valiosas aos utilizadores.

4. **Interface de Utilizador Intuitiva:** O *frontend*, desenvolvido em React.js, ofereceu uma experiência de utilizador fluida e responsiva. A organização das páginas foi projetada para ser intuitiva, permitindo que utilizadores de diferentes níveis de experiência interagissem facilmente com o sistema.
5. **Automatização e Eficiência:** A utilização de *queues* para controlar o fluxo de mensagens e tarefas garantiu que o sistema operasse de forma eficiente, mesmo em cenários de alta carga de dados. Este mecanismo evitou congestionamentos e assegurou que os agentes processassem as tarefas de forma ordenada.

## 5.2 Limitações e Desafios

1. **Alta Dependência de Fontes Externas:** O sistema depende de APIs externas, como CoinGecko e CryptoPanic, para recolher dados. Embora estas fontes sejam confiáveis, a sua disponibilidade e desempenho podem impactar diretamente o funcionamento do sistema. Uma interrupção ou alteração nas APIs pode comprometer a recolha de dados.
2. **Complexidade do Modelo de Previsão:** Embora o modelo LSTM tenha demonstrado bons resultados em algumas criptomoedas, nem todas as moedas apresentam previsões precisas. A qualidade das previsões está diretamente relacionada à quantidade e qualidade dos dados históricos disponíveis. Para alcançar previsões mais precisas, seria necessário recolher uma quantidade significativamente maior de dados, cobrindo períodos mais longos e incluindo mais variáveis externas. Além disso, é importante reconhecer que os preços das criptomoedas não são influenciados apenas por notícias ou *posts* nas redes sociais ou análises de sentimento. Fatores externos, como regulamentações governamentais, eventos globais e manipulação de mercado, também desempenham um papel crucial e são difíceis de modelar. Já para não falar da própria natureza do mercado, que é extremamente volátil.
3. **Escalabilidade em Cenários de Alta Carga:** Embora o sistema tenha sido projetado para ser escalável, cenários de alta carga, como um aumento exponencial no número de utilizadores ou fontes de dados, podem exigir otimizações adicionais na infraestrutura, como a utilização de *load balancers* ou a migração para arquiteturas baseadas em microsserviços.

### 5.2.1 Ética e Privacidade

A implementação de sistemas como este levanta questões importantes relacionadas com ética e privacidade, especialmente no contexto da recolha e utilização de dados. Atualmente, o sistema utiliza APIs públicas de plataformas como CoinGecko, CryptoPanic e Reddit para recolher informações. Embora estas fontes sejam amplamente utilizadas, a sua utilização em contextos comerciais pode ser eticamente questionável, uma vez que os dados recolhidos podem não ter sido explicitamente disponibilizados para fins de previsão ou análise comercial.

Para garantir conformidade ética e legal, seria necessário recorrer a APIs pagas e licenciadas, que oferecem acesso a dados de forma regulamentada e com permissões explícitas para uso comercial. Estas APIs geralmente incluem termos de serviço que garantem a privacidade dos dados e a proteção dos utilizadores das plataformas de origem.

Além disso, é importante reconhecer que plataformas como Reddit podem conter informações sensíveis ou pessoais que, mesmo sendo públicas, não foram compartilhadas com o objetivo de serem utilizadas em previsões financeiras. A utilização destes dados para fins comerciais pode violar a privacidade dos utilizadores e comprometer a confiança nas plataformas. Para mitigar este risco, seria essencial implementar filtros rigorosos para garantir que apenas dados relevantes e anonimizados sejam utilizados no sistema.

### 5.3 Sugestões e Recomendações

Com base na análise crítica do sistema desenvolvido, identificamos algumas áreas que podem ser aprimoradas para aumentar a eficiência, escalabilidade e conformidade ética do sistema. Abaixo, apresentamos algumas sugestões e recomendações para futuras iterações:

1. **Integração de APIs Pagas e Licenciadas:** Para garantir conformidade ética e legal, recomenda-se a utilização de APIs pagas que ofereçam acesso regulamentado a dados. Estas APIs geralmente incluem termos de serviço claros e suporte técnico, reduzindo os riscos associados à utilização de fontes públicas.
2. **Otimização do Modelo de Previsão:** Explorar modelos alternativos, como *Transformers* ou *Ensemble models*, para melhorar a precisão das previsões. Além disso, recolher uma quantidade maior de dados históricos e incluir variáveis externas, como regulamentações e eventos globais, pode aumentar a eficácia do modelo.
3. **Privacidade e Anonimização de Dados:** Implementar filtros rigorosos para garantir que apenas dados relevantes e anonimizados sejam utilizados no sistema. Além disso, adotar práticas de conformidade com regulamentações de privacidade, como o GDPR, para proteger os dados dos utilizadores.
4. **Monitorização e Alertas:** Desenvolver um sistema de monitorização para verificar a disponibilidade e o desempenho das APIs utilizadas. Caso uma API apresente problemas, o sistema pode alternar automaticamente para uma fonte alternativa, garantindo a continuidade das operações.
5. **Expansão de Funcionalidades:** Adicionar novas funcionalidades, como a análise de correlações entre diferentes criptomoedas ou a inclusão de indicadores técnicos avançados, pode aumentar o valor do sistema para os utilizadores.



Estas recomendações visam não apenas melhorar o desempenho técnico do sistema, mas também garantir que ele esteja alinhado com padrões éticos e regulamentares, consolidando-se como uma ferramenta confiável e eficaz no mercado de criptomoedas.

## 6 Conclusão

O sistema multiagente desenvolvido demonstrou ser uma solução robusta e eficiente para análise de criptomoedas, integrando tecnologias avançadas de recolha de dados, análise de sentimento e previsão de preços. A arquitetura modular e escalável permitiu a implementação de funcionalidades complexas, como notificações personalizadas e previsões baseadas em modelos de aprendizagem automática, garantindo uma experiência fluida e informativa para os utilizadores.

Apesar das limitações identificadas, como a dependência de fontes externas e os desafios associados à precisão das previsões, os resultados obtidos validam a eficácia da abordagem proposta. O sistema mostrou-se capaz de lidar com grandes volumes de dados e fornecer informações úteis para a tomada de decisões no mercado de criptomoedas. No entanto, é importante reconhecer que o sucesso de um sistema como este depende não apenas da tecnologia, mas também da sua conformidade ética e da privacidade dos dados utilizados.

Com as melhorias sugeridas, como a integração de APIs pagas e licenciadas, a otimização do modelo de previsão e a adoção de práticas mais robustas de segurança e privacidade, o sistema poderá alcançar um nível ainda maior de desempenho, escalabilidade e acessibilidade. Estas melhorias não apenas aumentariam a confiabilidade do sistema, mas também reforçariam o seu alinhamento com padrões éticos e regulamentares.

Em suma, o sistema desenvolvido tem o potencial de se consolidar como uma ferramenta valiosa para investidores e analistas no mercado de criptomoedas, contribuindo para decisões mais informadas e estratégicas em um setor dinâmico e em constante evolução.