Ministério da Educação
Universidade Tecnológica Federal do Paraná
Departamento Acadêmico de Computação - DACOM

RELATÓRIO FINAL DE PROJETO PARA DISCIPLINA OFICINA DE INTEGRAÇÃO

MB Stock Monitor

Vinicius de Carvalho Baggio
Vítor Ângelo Misciato Teixeira

Engenharia de Computação

CAMPUS CORNÉLIO PROCÓPIO, 2021

SUMÁRIO

INTRODUÇÃO

O mercado de ações, por muitos temido e, por outros, visto como uma oportunidade de crescer financeiramente, vem crescendo de maneira assustadora nos últimos anos no Brasil. A bolsa de valores brasileira, representada pela Brasil, Bolsa, Balcão (B3), conquistou, em 2020, 1,5 milhões de novos investidores, um aumento de 92% em relação a 2019, saltando de cerca de 1,7 milhões de contas de pessoas físicas cadastradas para 3,2 milhões [1]. Em fevereiro deste ano, esse número subiu para 3,5 milhões [2], o que indica que esse crescimento não demonstra tendência de desacelerar tão cedo.

Um dos principais fatores que motivou os brasileiros a apostar no mercado de risco foi a queda da taxa básica de juros nos últimos anos, a SELIC, tornando investimentos conservadores de renda fixa, como a poupança, pouco rentáveis e atrativos [3]. Além disso, com a pandemia e a paralisação mundial, muitas empresas tiveram que interromper seus serviços, causando grande desvalorização de suas ações, ocasionando uma série de *circuit breakers*, em março de 2020, atraindo pessoas que acreditavam em uma recuperação posterior.

Relacionada a isso, vem entrelaçada a busca por informação, acarretando, também, no uso de *sites* e ferramentas que disponibilizam dados para treinamento para começar a investir e se aprimorar no mercado financeiro, bem como para monitorar o comportamento das ações. Alguns dos principais *sites* nacionais sobre o mercado financeiro são o *InfoMoney* [4], o *MoneyTimes* [5] e a *Suno Research* [6]. Além desses, também existem outros internacionais, mas que cobrem também o mercado financeiro brasileiro, tais como o *Investing.com* [7], o *Yahoo Finance* [8] e a *Bloomberg* [9].

Nesses *sites*, são disponibilizados uma série de serviços, trazendo desde as últimas notícias, cotações e indicadores envolvendo os principais ativos do mercado (ações, índices, criptomoedas, títulos, etc.), até mesmo fornecendo calculadoras específicas, análises técnicas e disponibilização de minicursos.

OBJETIVOS

O objetivo geral do projeto é o desenvolvimento de uma interface *web* com gráficos para monitorar o comportamento diário e fornecer dados históricos das empresas listadas na B3, bem como fazer o mesmo para os índices. Com isso, a ideia é que a interface possua:

- Um gráfico interativo dos valores históricos e *intraday* de uma determinada ação;
- Um gráfico interativo dos valores históricos e *intraday* de um determinado índice;
- Valores das principais ações em tempo real;
- Campos para que o usuário escolha a ação e o índice que deseja ver as informações;
- Lista com os principais indicadores das ações, como, por exemplo, último valor pago por dividendos e variação do preço/pontos;
- Valor de previsão de fechamento de ação para o pregão do dia, com uso de algum modelo de *machine learning*.

JUSTIFICATIVA

Como existem diversas informações disponíveis e variáveis para serem consideradas em uma análise de mercado, muitas vezes se torna confuso e trabalhoso abrir diversas ferramentas ao mesmo tempo e organizá-las para encontrar o que se quer. Assim, esse projeto é uma iniciativa para produzir uma ferramenta de análise de dados sobre o mercado de ações que apresente a síntese de informações, priorizando trazer dados mais importantes e valiosos para os usuários.

Como base, foi escolhido um projeto desenvolvido pela *Alpha Vantage* [10], chamado *Stock Visualization Website*, disponível nesse link. Porém, como citado nos objetivos, pretende-se trazer mais funcionalidades ao projeto, mas mantendo a simplicidade da informação, com valores e dados que pessoas mais leigas possam analisar e compreender, buscando atingir a grande massa de novos investidores que ingressaram no mercado de risco neste e no último ano.

Como principais diferenciais da ideia podem ser citadas a utilização intuitiva e a simples compreensão das informações presentes na interface, garantindo que possa ser utilizada por um público maior, não requerer que o usuário se cadastre para acessar e possuir, ainda que simples, uma funcionalidade voltada ao uso do aprendizado de máquina, que ainda vem sendo explorado no mercado de ações e não está tão presente na maior parte dos *sites*.

MATERIAL E MÉTODOS

Para organização, o projeto foi dividido em duas partes, sendo uma relacionada com a construção propriamente dita do *site* e a outra relacionada à manipulação e armazenamento dos dados. Como base, optou-se por utilizar o Django [11], um *framework open source* de desenvolvimento *web Python* de alto nível para desenvolvimento rápido, que utiliza o padrão *Model-View-Template* (MVT).

Na parte *front-end* do projeto, foram utilizadas as linguagens HTML [12], CSS [12] e JavaScript [13], com uso da *Application Programming Interface* (API) jQuery [14] para manipulação de elementos HTML e da API ApexCharts.js [15] para criação e manipulação dos gráficos.

No *back-end*, para a manipulação e armazenamento dos dados, como banco de dados optou-se por utilizar o MySQL [16], um banco relacional. Para coleta dos dados sobre as ações e índices, foi utilizada a biblioteca *yfinance* [17], que foi desenvolvida buscando substituir a API do *Yahoo!Finance* para baixar os dados históricos de mercado, que foi descontinuada há alguns anos. Já para criação do modelo de *machine learning* para previsão do valor de fechamento da ação foi utilizada a biblioteca *scikit-learn*.

RESULTADOS

Banco de dados e população das tabelas. Inicialmente, a primeira coisa a ser estruturada no projeto foi o banco de dados, para definir como as informações que utilizamos são armazenadas. Na Figura 1, é possível ver o Diagrama Entidade-Relacionamento (DER) do banco de dados do projeto, que mostra as tabelas que foram criadas, seus atributos, chaves e como estão relacionadas entre si.

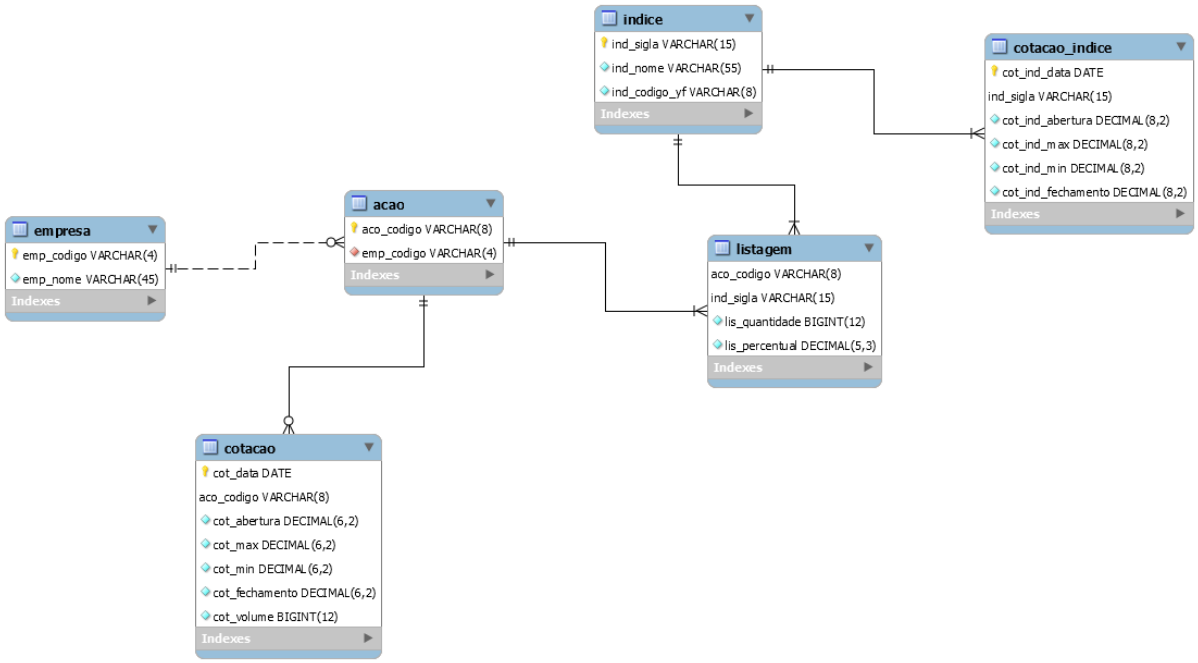

Figura 1. DER do banco de dados do projeto.

Para popular as tabelas, foram utilizados dados de duas fontes: do *site* da B3 e da biblioteca *yfinance*. Do *site* da B3, foram utilizados os dados responsáveis por popular as tabelas *empresa, acao, indice* e *listagem*, que são fornecidos publicamente através de planilhas e documentos no formato CSV, sendo necessário um trabalho inicial de manipulação e limpeza para se adequarem à estrutura do banco, para onde foram importados posteriormente utilizando o MySQL *Workbench*, que é a ferramenta visual de manipulação de banco de dados do MySQL. Na Figura 2, é possível ver como os dados para estas tabelas estão inseridos no banco.

emp_codigo	emp_nome
RRRP	3R PETROLEUM
QVQP	524 PARTICIP
ABCB	ABC BRASIL
EALT	ACO ALTONA
AERI	AERIS
AESB	AES BRASIL
AESL	AES SUL
AFLT	AFLUENTE T
GRAO	AGRIBRASIL
BRGE	ALFA CONSORC
CRIV	ALFA FINANC
RPAD	ALFA HOLDING

aco_codigo	ind_sigla	list_quantidade	list_percentual
AALR3	IBrA	46716473	0.019
AALR3	ICON	46716473	0.073
AALR3	IGC	93432946	0.023
AALR3	IGC-NM	46716473	0.033
AALR3	IGCT	46716473	0.020
AALR3	ITAG	46716473	0.020
AALR3	SMLL	46716473	0.150
ABCB4	IBrA	70580422	0.038
ABCB4	IDIV	143406221	0.664
ABCB4	IFNC	70580422	0.200
ABCB4	IGC	105870633	0.034

ind_sigla	ind_nome
IFNC	Índice BM&FBOVESPA Financeiro
Ibovespa	Índice Bovespa
IBrX 100	Índice Brasil 100
IBrX 50	Índice Brasil 50
IBrA	Índice Brasil Amplo BM&FBOVESPA
ICO2	Índice Carbono Eficiente
IGC	Índice de Ações com Governança Corporativa D...
ITAG	Índice de Ações com Tag Along Diferenciado
ICON	Índice de Consumo
IEE	Índice de Energia Elétrica
IGC-NM	Índice de Governança Corporativa – Novo Merc...

aco_codigo	emp_codigo
AALR3	AALR
ABCB4	ABCB
ABEV3	ABEV
AERI3	AERI
AESB3	AESB
AGRO3	AGRO
ALLD3	ALLD
ALPA3	ALPA
ALPA4	ALPA
ALPK3	ALPK
ALSO3	ALSO

Figura 2. Organização dos dados retirados da B3 no banco.

Da biblioteca *yfinance*, foram utilizados os dados para popular as tabelas *cotacao* e *cotacao_indice*, que são as tabelas de maior importância do banco, já que vão sempre receber novas informações. Para populá-las, foram utilizados apenas dados históricos, ou seja, apenas um registro para cada dia em que o pregão já tenha sido encerrado. Portanto, não serão armazenados os dados *intraday* no banco. Na Figura 3, podemos ver como os dados são retornados pela biblioteca, sendo a estrutura similar tanto para cotação das ações, quanto para cotação dos índices. Como pode ser notado, o *DataFrame* retornado já possui a estrutura praticamente adequada para os dados serem salvos no banco, bastando apenas um trabalho simples de manipulação no *script* em *Python* para remover os campos que não serão utilizados, adicionar o código da ação/índice e pegar apenas a data do campo *Date*.

Date	Open	High	Low	Close	Adj Close	Volume
2021-08-18 00:00:00	105.74	105.74	103.75	103.77	103.77	12797600
2021-08-17 00:00:00	108.11	109.32	105.66	107	107	23402600
2021-08-16 00:00:00	107.24	108.9	105.92	108.8	108.8	21856700
2021-08-13 00:00:00	109.7	109.9	107.96	108.3	108.3	11855200
2021-08-12 00:00:00	109.2	109.98	108.83	109.2	109.2	14849500
2021-08-11 00:00:00	110.19	110.5	109.26	109.27	109.27	14680600
2021-08-10 00:00:00	108.95	110.74	108.84	110.06	110.06	17440500
2021-08-09 00:00:00	108.2	109.45	107.68	109.01	109.01	21502300
2021-08-06 00:00:00	109.85	110.89	109.5	109.7	109.7	19273100
2021-08-05 00:00:00	109.64	109.99	107.9	109.08	109.08	32215200
2021-08-04 00:00:00	112	113.25	111.62	112.52	112.52	18828200
2021-08-03 00:00:00	109.78	113.03	109.44	112.64	112.64	36641400

Figura 3. Estrutura de retorno das cotações pela *yfinance*.

***Back-end* do projeto.** Com a estrutura do banco de dados definida, a próxima etapa foi a criação do projeto no Django, que, através de um comando inicial, gera uma coleção de configurações para uma instância do Django, envolvendo criação de diretórios e arquivos onde são armazenados os códigos, e, dentro desse projeto, foi criada uma aplicação, que é propriamente o *MB Stock Monitor*. É importante destacar a diferença entre projeto e aplicação: uma aplicação é um conjunto de elementos *web* que faz alguma coisa, e, dentro de um projeto de um *website*, podem existir várias aplicações.

Em seguida, foi estabelecida a conexão entre o banco de dados criado e o projeto. No Django, o conceito de *Models* é a representação dos dados armazenados, onde cada modelo é uma classe *Python* (Entidade) que mapeia uma tabela única. Na criação dos modelos existem duas opções: ou as classes são criadas no código em *Python* e as mudanças são migradas para o banco, criando a estrutura, ou através de um comando, a partir de um banco de dados já existente, o Django cria as classes e importa as suas configurações. No caso do nosso projeto, a segunda opção é o que de fato aconteceu. Na Figura 4 é possível ver um trecho de código que mostra algumas das classes importadas do banco. Como pode-se ver e é de se esperar, as configurações

e atributos refletem o que está no DER apresentado na Figura 1.

```
class Acao(models.Model):
    aco_codigo = models.CharField(primary_key=True, max_length=8)
    emp_codigo = models.ForeignKey('Empresa', models.DO_NOTHING, db_column='emp_codigo')

    class Meta:
        managed = False
        db_table = 'acao'


class Cotacao(models.Model):
    cot_data = models.DateField(primary_key=True)
    aco_codigo = models.ForeignKey(Acao, models.DO_NOTHING, db_column='aco_codigo')
    cot_abertura = models.DecimalField(max_digits=6, decimal_places=2)
    cot_max = models.DecimalField(max_digits=6, decimal_places=2)
    cot_min = models.DecimalField(max_digits=6, decimal_places=2)
    cot_fechamento = models.DecimalField(max_digits=6, decimal_places=2)
    cot_volume = models.BigIntegerField()

    class Meta:
        managed = False
        db_table = 'cotacao'
        unique_together = (('cot_data', 'aco_codigo'),)


class CotacaoIndice(models.Model):
    cot_ind_data = models.DateField(primary_key=True)
    ind_sigla = models.ForeignKey('Indice', models.DO_NOTHING, db_column='ind_sigla')
    cot_ind_abertura = models.DecimalField(max_digits=8, decimal_places=2)
    cot_ind_max = models.DecimalField(max_digits=8, decimal_places=2)
    cot_ind_min = models.DecimalField(max_digits=8, decimal_places=2)
    cot_ind_fechamento = models.DecimalField(max_digits=8, decimal_places=2)

    class Meta:
        managed = False
        db_table = 'cotacao_indice'
```

Figura 4. *Models* do projeto importadas do banco de dados já criado.

O próximo passo envolveu a criação das *views*. As *views* são as funções que recebem uma solicitação *Web* (*request*) e retornam uma resposta (*response*). Portanto, elas recebem como primeiro parâmetro um objeto *HttpRequest* e retornam qualquer tipo de coisa, podendo ser o conteúdo HTML de uma página, um redirecionamento, um erro, um documento, etc. Além disso, cada *view* é responsável por mapear uma URL, ou seja, é responsável pelo comportamento dela.

Na Figura 5, podem ser vistas as *views* criadas no projeto. No nosso caso, a *view home* é a responsável por retornar e renderizar o *template* da aplicação, ficando a cargo das demais o processamento, a atualização e o retorno dos dados utilizados para popular os gráficos e campos de informações, cada uma delas retornando um objeto *HttpResponse* com um documento no formato JSON. Os dados retornados por elas são:

- Cotações históricas e *intraday* das ações;
- Cotações históricas e *intraday* dos índices;
- Principais indicadores de mercado das ações;
- Dados para auxiliar no comportamento dos objetos contidos na *interface* gráfica.

```
@csrf_exempt
def home(request):
    return render(request,'index.html',{})

@csrf_exempt
def get_stock_data(request): ...

@csrf_exempt
def get_index_data(request): ...

def get_index_stock(request): ...

@csrf_exempt
def get_stock_diff_view(request): ...
```

Figura 5. *Views* do projeto.

Juntamente com esses dados, é retornado um valor de previsão para fechamento da ação que está sendo visualizada, obtido através da aplicação de um modelo de *machine learning*. Como modelo, optou-se por utilizar o de Regressão Linear, disponibilizado através da biblioteca *scikit-learn*. Nesse caso, foram selecionados como previsores os valores de cotação de abertura, máximo e mínimo, treinando o modelo com dados datados até 15 dias anteriores ao do dia atual e pegando os últimos valores disponíveis *intraday* para realizar a previsão.

Como o comportamento das *views* é parecido e as etapas do processamento dos dados podem ser extensas, foi criado um arquivo contendo as funções, que funciona como se fosse uma biblioteca, sendo essas funções que manipulam de fato os dados, utilizadas dentro das *views*. A Figura 6 mostra as funções criadas.

```
def dict_return(cursor): ...

def last_day_bd_acao(ticker): ...

def atualiza_acao_banco(start, ticker): ...

def get_dados_historicos_acao(ticker): ...

def get_intraday_acao(ticker): ...

def last_day_bd_index(cod_index): ...

def atualiza_index_banco(start, cod_index): ...

def get_dados_historicos_index(cod_index): ...

def get_intraday_index(cod_index): ...

def get_top_index_stock(cod_index): ...

def get_stock_by_index(): ...

def get_stock_diff(ticker): ...

def get_indicadores_stock(ticker): ...

def previsao_acao(ticker): ...
```

Figura 6. Funções auxiliares das *views* do projeto.

***Front-end* do projeto.** Na Figura 7, é possível ver o *layout* do *site*, contendo apenas o que foi criado com HTML e CSS. Nele, temos as três áreas de informações que serão preenchidas com dados advindos do *back-end* através eventos que fazem requisições do tipo *Asynchronous JavaScript and XML* (AJAX), que buscam os dados das URLs pelas quais cada *view* apresentada anteriormente é responsável.

A linha logo abaixo do nome do *site* é um espaço com efeito rolável, chamado *marquee*, contendo as dez ações com maior participação na listagem do índice que estiver ativo no gráfico da direita e onde cada elemento dentro deste espaço corresponde a uma ação. Este gráfico da direita, como citado, é o espaço para conter todas as informações sobre o índice da bolsa que o usuário escolher. Por último, temos o gráfico da esquerda, sendo a principal funcionalidade, que mostra todas as informações referentes à ação selecionada também pelo usuário.

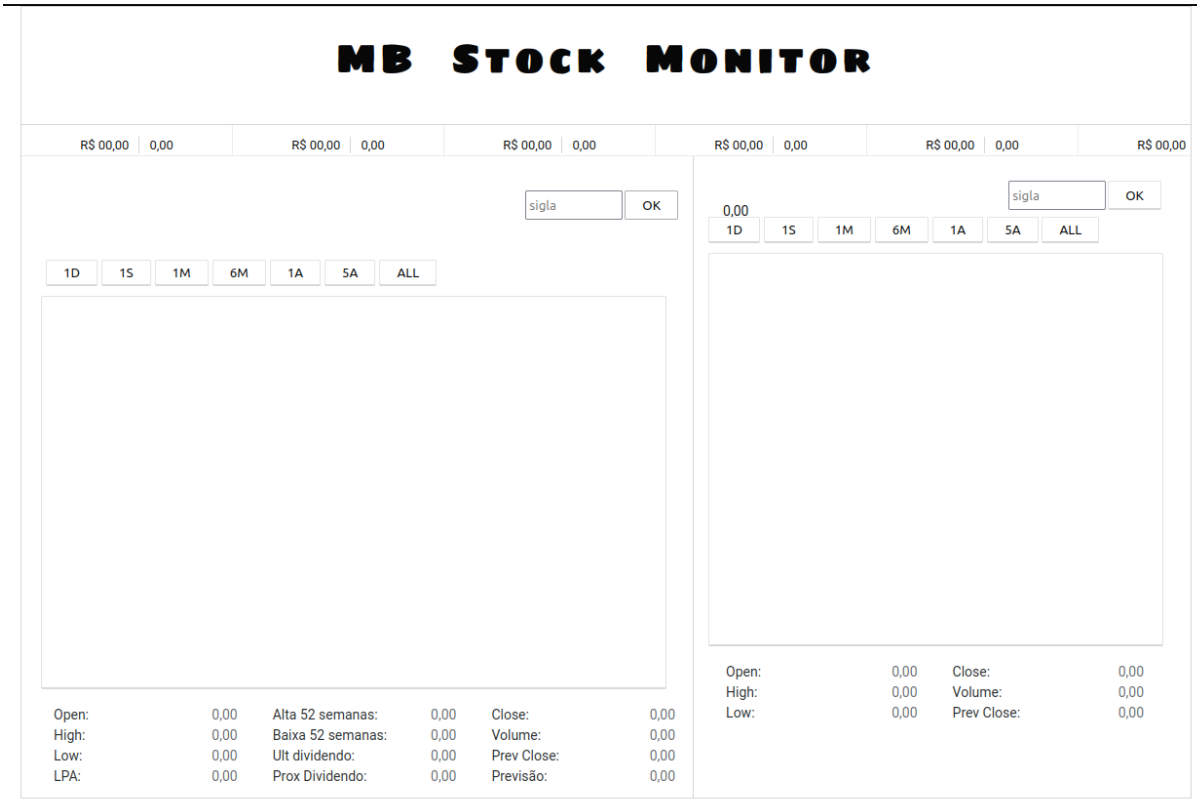

Figura 7. *Layout* do *site*.

Ao carregar o *site* pela primeira vez, uma requisição AJAX do tipo *GET* é feita para retornar todos os nomes de índices e ações correspondentes listadas em cada um disponíveis no banco, sendo esses valores responsáveis por controlar o comportmento das demais requisições feitas ao *back-end*, e, além disso, também serve para popular as caixas de texto, criando um *autocomplete*, a fim de mostrar as opções disponíveis ao usuário.

Após o retorno desses dados, é acionado (via código, visto que é o carregamento do *site* e o usuário ainda não interagiu com a interface) o botão do gráfico da direita, que responde ao evento de pegar o valor da caixa de texto correspondente e fazer uma requisição AJAX do tipo *POST*, passando o valor do *"ticker"* ao *back-end*, que retorna todos os valores referentes ao índice selecionado. Após o retorno desses valores, são processados no JavaScript os dados para popular todos os campos referentes ao gráfico, o que pode ser visto na Figura 8.

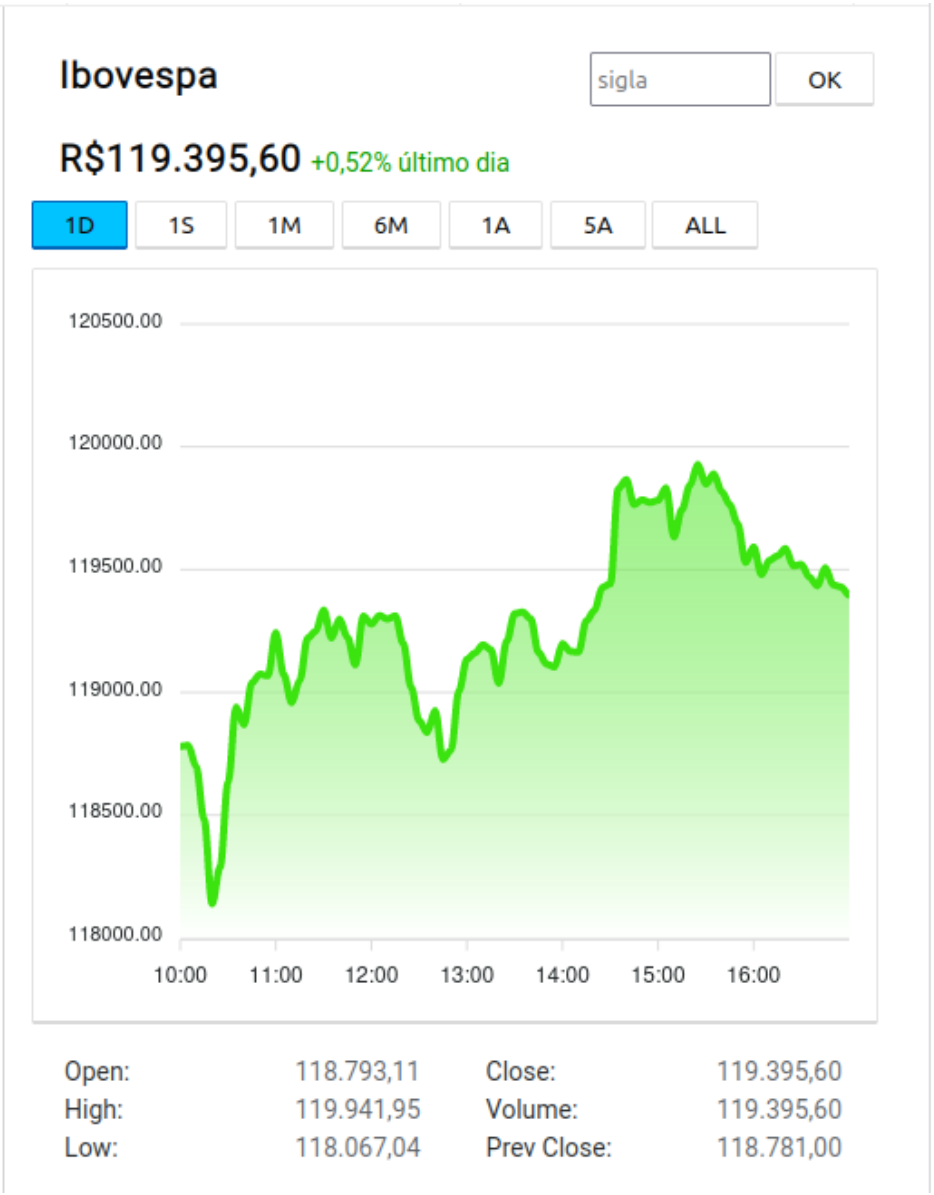

Figura 8. Gráfico do índice após ser populado.

Ainda nos eventos disparados por esse botão, junto com os dados do índice também vêm as dez ações com maior participação nesse índice selecionado. Assim, como última ação desencadeada, é chamada a rotina para preenchimento do *marquee* passando essas ações como argumentos.

Na rotina de preenchimento do *marquee*, para cada ação é feita um requisição AJAX do tipo *POST* passando como dado o *"ticker"* da ação, que retorna seu valor atual e a porcentagem de variação em relação ao seu último valor de fechamento, que são usados para preencher cada elemento dentro do *marquee* correspondente à ação que fez a requisição, resultando no que é mostrado na Figura 9.

Figura 9. *Marquee*.

Após essa rotina processar todas as ações, é acionado como comportamento final deste primeiro carregamento a ativação do botão do gráfico das ações (da esquerda). Esse botão, por sua vez, seleciona o valor na caixa de texto referente ao gráfico, sendo que nessa primeira vez será usado como padrão a primeira ação que aparece listada no índice ativo no gráfico ao lado. Com esse valor, é feita uma requisição AJAX do tipo *POST* passando como dado o *"ticker"* da ação, retornando todos os valores correspondentes das cotações, além de informações extras, como: nome da empresa, dividendo, previsão de fechamento, etc. Com esses dados, após o processamento deles, todos os campos referentes ao espaço das ações são preenchidos, resultando na Figura 10.

Figura 10. Gráfico da ação após ser populado.

Vale mencionar que, a rotina de preenchimento do *marquee*, e, consequentemente, o acionamento do botão do gráfico das ações (esquerda), foi programada para ser ativada a cada minuto, pois assim o *marquee* e o gráfico da ação ativa estarão sempre se modificando e mantendo seus valores atualizados.

Com isso temos o *site* inteiramente carregado e pronto para uso, como mostra a Figura 11.

Figura 11. *Site* totalmente carregado.

Com a página carregada, o usuário pode interagir com os gráficos através dos botões acima dos mesmos, que demonstram os valores dos índices ou ações em determinados períodos, como mostra a Figura 12. Para melhor utilização, para alguns períodos os dados históricos são filtrados, reduzindo assim o número de registros que aparecem nos gráficos com maior volume.

Figura 12. Interação com os gráficos.

Através das caixas de texto, pode-se trocar os valores de índice e ação, digitando ou selecionando através do *autocomplete* os valores desejados e pressionando os botões correspondentes à cada gráfico, cujos comportamentos seguem o que foi descrito no carregamento inicial da página. É importante destacar que as ações que podem ser selecionadas dependem diretamente do índice que está em exibição, ou seja, o usuário pode selecionar apenas aquelas que estiverem listadas nele. Portanto, caso algum nome inválido ou ação que não esteja listada no índice seja utilizado para realizar a busca, o gráfico não será atualizado, informando valor inválido.

Figura 13. Alteração dos gráficos.

Não só os gráficos, mas também o elemento *marquee* é interativo, sendo possível clicar em cada ação em amostra nesse elemento para selecioná-la e assim ativar o botão do gráfico das ações, carregando seu valor no gráfico.

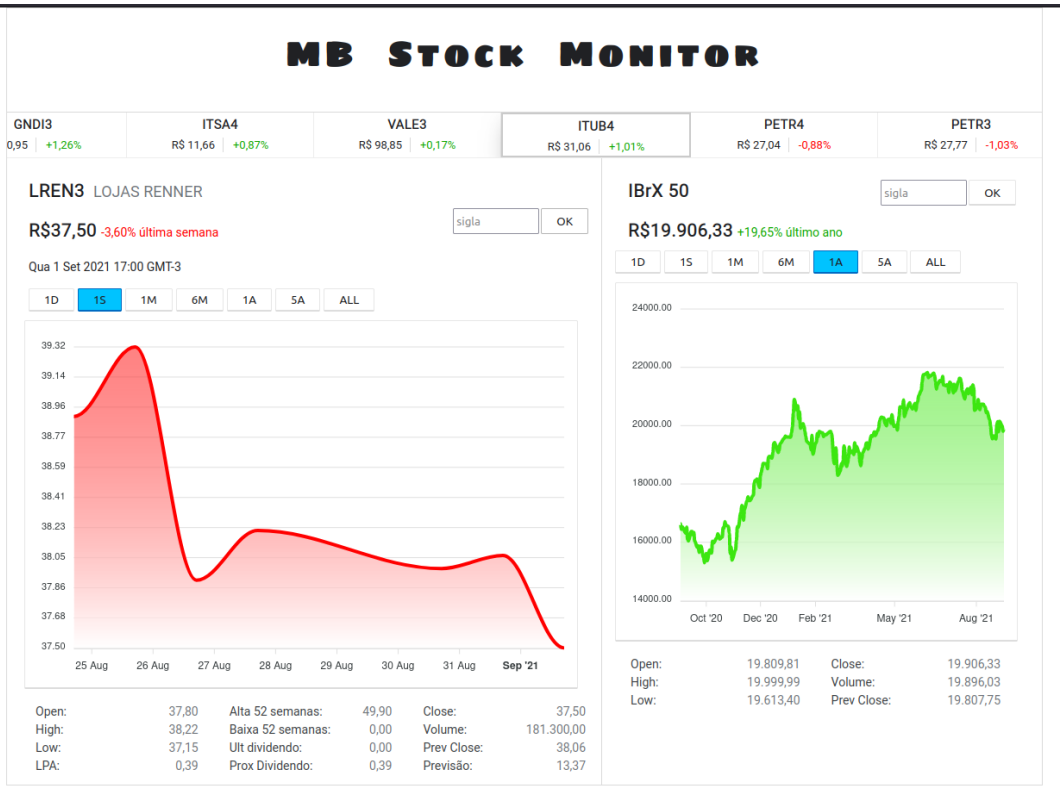

Figura 14. Selecionando uma ação do *marquee*.

Figura 15. Resultado da seleção da ação do *marquee*.

Outro ponto relevante sobre o carregamento de informações é que, ao fazer uma requisição para o retorno dos dados, estes são atualizados no banco caso o último valor de cotação histórica relacionada à ação não seja da data anterior ao momento da execução. Além disso, como o pregão não funciona todos os dias (apenas de segunda a sexta), caso não existam registros *intraday* para o dia serão utilizados os registros *intraday* do último dia disponível.

CRONOGRAMA

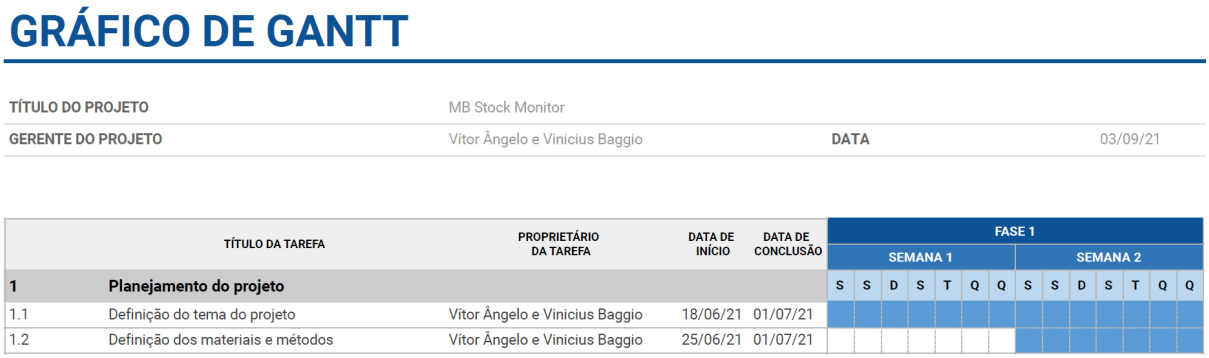

GRÁFICO DE GANTT

TÍTULO DO PROJETO	MB Stock Monitor		
GERENTE DO PROJETO	Vítor Ângelo e Vinicius Baggio	DATA	03/09/21

	TÍTULO DA TAREFA	PROPRIETÁRIO DA TAREFA	DATA DE INÍCIO	DATA DE CONCLUSÃO	FASE 1												
					SEMANA 1							SEMANA 2					
1	**Planejamento do projeto**				S	S	D	S	T	Q	Q	S	S	D	S	T	Q	Q
1.1	Definição do tema do projeto	Vítor Ângelo e Vinicius Baggio	18/06/21	01/07/21													
1.2	Definição dos materiais e métodos	Vítor Ângelo e Vinicius Baggio	25/06/21	01/07/21													

Figura 16. Cronograma fase 1.

	TÍTULO DA TAREFA	PROPIETÁRIO DA TAREFA	DATA DE INÍCIO	DATA DE CONCLUSÃO
2	**Fase inicial do projeto**			
2.1	Desenvolvimento do layout do site	Vinicius Baggio	02/07/21	15/07/21
2.2	Definição das funcionalidades dos site	Vinicius Baggio	16/07/21	29/07/21
2.3	Fazer configurações iniciais do projeto no Django	Vítor Ângelo	06/07/21	07/07/21
2.4	Criar a estrutura do banco	Vítor Ângelo	02/07/21	03/07/21
2.5	Alimentar com dados iniciais o banco	Vítor Ângelo	04/07/21	05/08/21
2.6	Conectar o banco ao Django	Vítor Ângelo	08/07/21	09/08/21
2.7	Implementar gráficos ao site	Vinicius Baggio	23/07/21	29/07/21

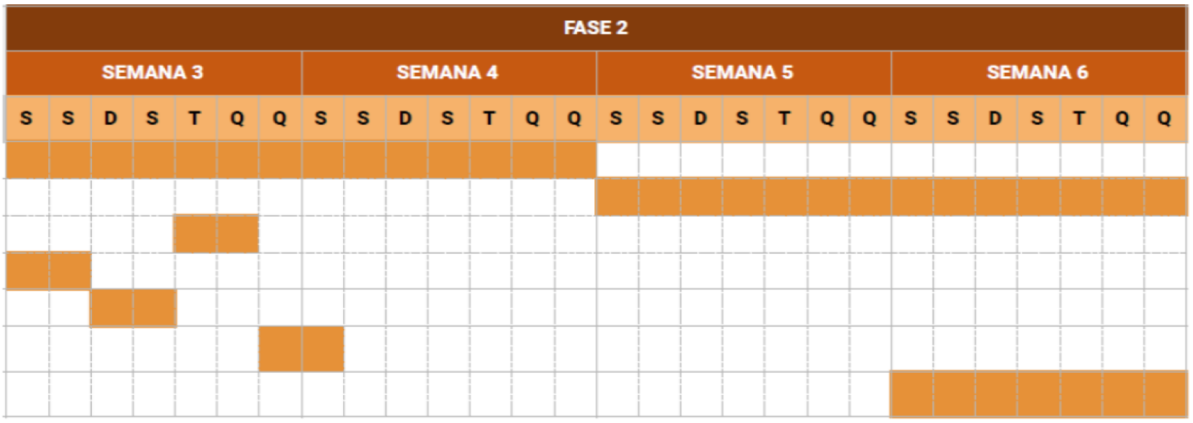

Figura 17. Cronograma fase 2.

	TÍTULO DA TAREFA	PROPIETÁRIO DA TAREFA	DATA DE INÍCIO	DATA DE CONCLUSÃO
3	**Fase final do projeto**			
3.1	Organizar o retorno de dados do banco	Vítor Ângelo	30/07/21	29/08/21
3.2	Finalizar os comportamentos das *views*	Vítor Ângelo	30/07/21	29/08/21
3.3	Desenvolver modelo de previsão de fechamento	Vítor Ângelo	19/08/21	22/08/21
3.4	Integrar *front-end com o back-end*	Vítor Ângelo e Vinicius Baggio	30/07/21	29/08/21
3.5	Finalizar o layout do site	Vinicius Baggio	30/07/21	21/08/21
3.6	Finalizar as funcionalidades do site	Vinicius Baggio	30/07/21	21/09/21
3.7	Configurar o comportamento do site	Vinicius Baggio	22/08/21	28/09/21
3.8	Testar e resolver os últimos erros	Vinicius Baggio	22/08/21	28/08/21
3.9	Finalização do projeto	Vinicius Baggio	29/08/21	02/09/21

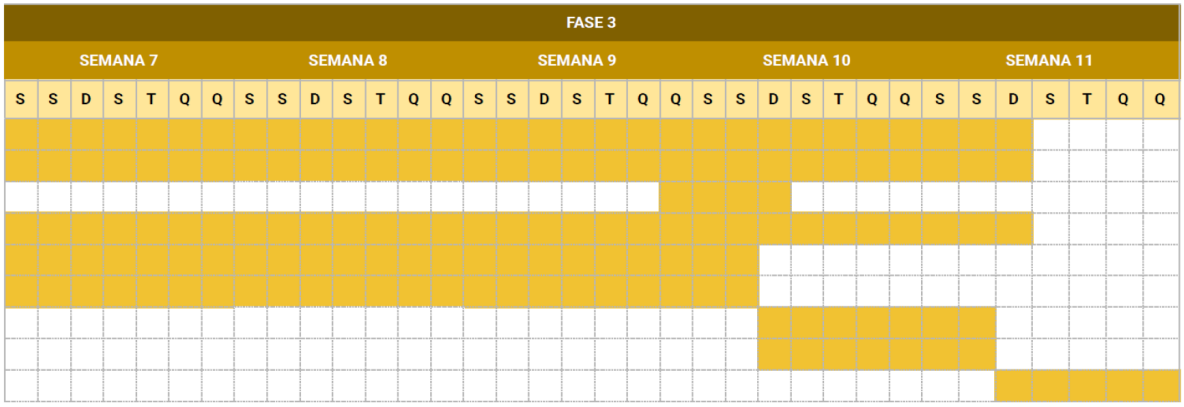

Figura 18. Cronograma fase 3.

CONCLUSÕES

Este projeto teve como principal objetivo desenvolver uma interface *web* para monitoramento das ações e índices listados no mercado de ações brasileiro, representado pela B3, trazendo gráficos dos dados referentes a cotações e informações mais relevantes de uma maneira simples e intuitiva, para que possa ser utilizado até por pessoas ingressantes nesse domínio.

Ao longo do desenvolvimento, foram identificadas as ferramentas e os dados necessários para a construção, sendo escolhido como base de produção o *framework* de desenvolvimento *web* Django, baseado na linguagem *Python*. Como dados, foram utilizados registros disponibilizados pela própria B3 e as cotações e previsores retornados pela biblioteca *yfinance*.

Dentro do que foi produzido, os aspectos que se destacam como de maior relevância são:

1. A interface possui gráficos interativos e de fácil manipulação, bem como presença de outros indicadores importantes sobre os ativos;
2. Permite que o usuário facilmente mude o índice e/ou a ação apresentada, tendo seu comportamento de seleção pré-configurado para evitar situações adversas, como ausência de *ticker* e visualização de ação não pertencente ao índice;
3. Retorno de um valor de previsão do valor de fechamento para ação, baseado em um modelo de *machine learning*.

Ao longo do projeto foram encontradas algumas dificuldades e desafios, sendo eles:

1. Manipular os dados vindos da biblioteca *yfinance*, principalmente pelo fato do retorno das informações dela não ser consistente, faltando por vezes dados ou vindo errados, além de produzir desempenho que varia conforme o horário do dia;

2. Definir a estrutura e arranjo dos elementos presentes na interface, bem como configurar os comportamentos deles, priorizando a facilidade de compreensão pelo usuário, mas sem perda de informações importantes;
3. Desenvolver a arte do *site*.

Analisando o que foi apresentado nos resultados obtidos, identificamos que atingimos os objetivos inicialmente propostos, trazendo as principais funcionalidades desejadas, de uma maneira que pessoas com menos conhecimento sobre o mercado de ações possam compreender os dados e informações expostas.

Esse domínio do mercado de ações demonstra que tende a crescer muito ainda, permitindo também a exploração de vários aspectos. Assim, como extensões naturais e sugestões de trabalhos futuros podem ser citados:

1. Aprimoramento da previsão para o valor de fechamento da ação, buscando aplicar mais técnicas e identificar mais *features* para treinar o modelo;
2. Trazer notícias e análises sobre o mercado de ações, bem como apresentar o comportamento dos investidores nas redes sociais, utilizando, por exemplo, Análise de Sentimento;
3. Aprimoramento e otimização o funcionamento da ferramenta, além de melhorar a arte da plataforma.

A seguir, são apresentadas as referências bibliográficas utilizadas na elaboração deste projeto.

REFERÊNCIAS

[1] D'ÁVILA, M. Z. *Bolsa conquista 1,5 milhão de novos investidores em 2020, um aumento de 92% no ano*: Forte volatilidade do mercado financeiro não impediu o maior apetite dos investidores ao risco. [S.l.: s.n.], jan. 2021. https://www.infomoney.com.br/onde-investir/bolsa-conquista-15-milhao-de-novos-investidores-em-2020-um-aumento-de-92-no-ano/. Acesso em 30-06-2021.

[2] CAMPOS, Á. *Número de investidores na bolsa brasileira alcança quase 3,5 milhões em fevereiro*: Em números absolutos, a B3 conquistou 120.542 investidores no mês. [S.l.: s.n.], mar. 2021. https://valorinveste.globo.com/mercados/renda-variavel/noticia/2021/03/11/numero-de-investidores-na-bolsa-brasileira-alcanca-quase-35-milhoes-em-fevereiro.ghtml. Acesso em 30-06-2021.

[3] ENTENDA o aumento do número de investidores na bolsa. [S.l.: s.n.]. https://www.investificar.com.br/entenda-o-aumento-do-numero-de-investidores-na-bolsa/. Acesso em 30-06-2021.

[4] INFOMONEY. [S.l.: s.n.]. https://www.infomoney.com.br/. Acesso em 09-08-2021.

[5] MONEYTIMES. [S.l.: s.n.]. https://www.moneytimes.com.br/. Acesso em 09-08-2021.

[6] SUNO Research. [S.l.: s.n.]. https://www.suno.com.br/. Acesso em 09-08-2021.

[7] INVESTING.COM. [S.l.: s.n.]. https://www.investing.com/. Acesso em 15-07-2021.

[8] YAHOO Finance. [S.l.: s.n.]. https://finance.yahoo.com/. Acesso em 15-07-2021.

[9] BLOOMBERG. [S.l.: s.n.]. https://www.bloomberg.com.br/. Acesso em 15-07-2021.

[10] ALPHA Vantage. [S.l.: s.n.]. https://www.alphavantage.co/. Acesso em 10-07-2021.

[11] DJANGO. [S.l.: s.n.]. https://www.djangoproject.com/. Acesso em 10-07-2021.

[12] DUCKETT, J. *HTML & CSS*: desing and build websites. [S.l.]: John Wiley Sons, 2011.

[13] DUCKETT, J. *JAVASCRIPT & JQUERY*: interactive front-end web development. [S.l.]: Wiley, 2014.

[14] JQUERY write less, do more. [S.l.: s.n.]. https://api.jquery.com/. Acesso em 01-09-2021.

[15] APEXCHART.JS. [S.l.: s.n.]. https://apexcharts.com/. Acesso em 30-06-2021.

[16] MYSQL. [S.l.: s.n.]. https://www.mysql.com/. Acesso em 30-06-2021.

[17] YFINANCE 0.1.59. [S.l.: s.n.]. https://pypi.org/project/yfinance/. Acesso em 30-06-2021.