FATEC SHUNJI NISHIMURA – POMPEIA

CURSO TECNOLOGIA EM BIG DATA NO AGRONEGÓCIO

Relatório de Atividades Desenvolvidas

CAIO HENRIQUE PEREIRA
EVANDRO FIGUEIREDO DE ALMEIDA
LUCAS DOMINGUES PERETTI
LUÍS OTÁVIO JASSI RODRIGUES
LUIZ HENRIQUE DE OLIVEIRA FERNEDA

Relatório de Projeto apresentado à Faculdade de Tecnologia Shunji Nishimura – FATEC Pompeia, como quesito para conclusão do 1º semestre do Curso de Tecnologia em Big Data no Agronegócio na disciplina de Projeto Integrador ministrada pelo Prof. Dr. Luís Hilário Tobler Garcia

Pompeia

2024

Introdução

O presente projeto acadêmico visa desenvolver um sistema automatizado para a coleta e disseminação de preços de frutas na internet, integrando esses dados ao aplicativo de mensagens Telegram. A proposta surge da necessidade de facilitar o acesso à informação de preços para consumidores e comerciantes, economizando tempo e dinheiro.

Utilizando técnicas de *web scraping*, os dados de preços de frutas são extraídos de *websites* de notícias agrícolas, processados e organizados em uma base de dados estruturada. Um *bot* no Telegram responde às consultas dos usuários, fornecendo os preços mais recentes das frutas desejadas.

Este sistema beneficia os consumidores ao reduzir o tempo e esforço necessários para encontrar as melhores ofertas, promovendo uma compra mais consciente e econômica. Além disso, o projeto demonstra a aplicabilidade prática de tecnologias de automação e análise de dados no cotidiano.

2. Objetivo

2.2 Objetivo Geral

- Facilitar o acesso à informação de preços de hortifruti para consumidores e comerciantes.

- Desenvolver um sistema automatizado para a coleta e disseminação de preço de frutas, integrando esses dados ao aplicativo de mensagens Telegram, com o intuito de fornecer aos consumidores informações atualizadas de preços.

2.3Objetivos Específicos

- Coletar dados de preços de frutas, por meio de técnicas de *web scraping* para extrair informações de preços de frutas de *website*.
- Organizar e armazenar dados, por meio de processamento e estruturação dos dados coletados em uma base de dados, garantindo a atualização e integridade das informações.
- Desenvolver um *bot* no Telegram com a criação, desenvolvimento e configuração de um *bot* que permita aos usuários consultar os preços das frutas através de mensagens no Telegram, fornecendo respostas rápidas e precisas.
- Promover a usabilidade do sistema e garantir que a interface do *bot* seja intuitiva e fácil de usar, proporcionando uma experiência positiva aos usuários.
- Monitorar e atualizar dados e implementar mecanismos para a atualização periódica dos dados de preços, assegurando que as informações fornecidas sejam sempre atuais.

3. Metodologia

A metodologia deste projeto envolveu várias etapas, desde a coleta de dados até a implementação e avaliação do sistema integrado ao Telegram. As etapas descritas a seguir detalham o processo de desenvolvimento, utilizando as bibliotecas Selenium, Pandas, Telebot e Os.

Foi realizado em 05/03/2024, a coleta de dados da fonte principal de informações para os preços das frutas foi o site "Notícias Agrícolas". Este site foi escolhido por sua abrangência e atualização constante dos preços das frutas.

Entre os dias 06/03/2024 e 10/03/2024, a coleta de dados foi realizada utilizando a biblioteca Selenium para *web scraping*. *Scripts* em Python foram desenvolvidos para acessar a página de cotações de frutas do site "Notícias Agrícolas" (Notícias Agrícolas,2024) , navegar até as seções relevantes e extrair os dados de preços e locais.

O código utilizado para essa tarefa é apresentado abaixo na Imagem 1:

Imagem 1 – Trecho de Captura dos Dados

```python
import pandas as pd
import os
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.chrome.options import Options
from datetime import date

data_hoje = date.today()
data_formatada = data_hoje.strftime("%d/%m/%Y")

# Config para nao abrir navegador
chrome_options = Options()
chrome_options.add_argument('--headless')  # Executar em modo headless (sem abrir a janela do navegador)
chrome_options.add_argument('--no-sandbox')
chrome_options.add_argument('--disable-dev-shm-usage')

browser = webdriver.Chrome(options=chrome_options)

# browser = webdriver.Chrome()
url = 'https://www.noticiasagricolas.com.br/cotacoes/frutas'
browser.get(url)

def search_bot(tam:int,id:int):
    txt_lista_nomes = []
    txt_lista_precos = []
    for i in range(1,tam):
        pracas_nomes = browser.find_elements(By.CSS_SELECTOR, f'table#\\"anchor\\"{id}.cot-fisicas tbody tr:nth-child({i}) td:nth-child(1)')
        precos_produto = browser.find_elements(By.CSS_SELECTOR, f'table#\\"anchor\\"{id}.cot-fisicas tbody tr:nth-child({i}) td:nth-child(2)')
        txt_lista_nomes.extend([elemento.text for elemento in pracas_nomes])
        txt_lista_precos.extend([elemento.text for elemento in precos_produto])
    return txt_lista_nomes, txt_lista_precos

frutas = {
    'abacate': [search_bot(10, 189)],
    'abacaxi': [search_bot(7, 190)],
    'mamao': [search_bot(10 ,191)],
    'maracuja': [search_bot(10, 192)],
```

Fonte: Elaborado pelos Autores (2024).

A Imagem 2 abaixo demonstra o código que define um dicionário chamado `frutas`, onde cada chave é o nome de uma fruta e o valor é o resultado da função `*search_bot*`. Em seguida, um *loop* `*for*` percorre cada fruta e seu resultado no dicionário, extraindo a lista de locais e preços do resultado.

Um *DataFrame* é criado com essas listas, contendo colunas 'Local' e 'Preço'. Finalmente, o *DataFrame* é salvo em um arquivo CSV, cujo nome é baseado no nome da fruta e está localizado dentro da pasta 'data', sem incluir o índice do *DataFrame* no arquivo CSV, conforme demonstrado na imagem 2.

Imagem 2 – Trecho da conversão armazenamento dos dados.

```
    'pera': [search_bot(10, 193)],
    'tangerina': [search_bot(10, 195)],
    'uva': [search_bot(10, 194)],
    'melancia': [search_bot(13, 89)],
    'banana': [search_bot(10, 68)],
    'limao_tahiti': [search_bot(12, 72)],
    'maca_fuji': [search_bot(4, 73)],
    'maca_gala': [search_bot(4, 74)]
}

for fruta, resultado in frutas.items():
    local, preco = resultado[0]  # Extrair os resultados
    df = pd.DataFrame({'Local': local, 'Preço': preco})
    # Salvar em um arquivo CSV separado
    local_salvar = os.path.join('data', fruta + '_data.csv')
    df.to_csv(local_salvar, index=False)

```

Fonte: Elaborado pelos Autores (2024).

Nos dias 11/03/2024 e 12/03/2024, os dados processados foram armazenados em arquivos CSV, organizados por tipo de fruta. Esta estruturação permitiu o fácil acesso e atualização dos dados.

Entre os dias 13/03/2024 e 20/03/2024, o *bot* no Telegram foi desenvolvido utilizando a biblioteca Telebot. Este *bot* serve como a interface de interação entre os usuários e o sistema de preços de frutas, permitindo consultas de preços diretamente pelo aplicativo Telegram.

Entre os dias 21/03/2024 e 25/03/2024, o *bot* foi programado para acessar os arquivos CSV sempre que uma solicitação de preços fosse recebida. Utilizando a biblioteca Telebot, o *bot* processava as consultas e respondia aos usuários com informações precisas e atualizadas sobre os preços das frutas. O código utilizado para essa tarefa é apresentado abaixo na Imagem 3:

Imagem 3 – Trecho do código do *bot.*

```python
import pandas as pd
import telebot
from telebot import types
from cotacoes import data_formatada

bot = telebot.TeleBot('6995053124:AAEmaVvDWBqnJt2GFLRHmRXGm0R69rKRmDM')


@bot.message_handler(commands=['start'])
def question(message):
    markup = types.InlineKeyboardMarkup(row_width=4)

    abacate = types.InlineKeyboardButton('Abacate', callback_data='abacate')
    abacaxi = types.InlineKeyboardButton('Abacaxi', callback_data='abacaxi')
    mamao = types.InlineKeyboardButton('Mamao', callback_data='mamao')
    maracuja = types.InlineKeyboardButton('Maracuja', callback_data='maracuja')
    pera = types.InlineKeyboardButton('Pera', callback_data='pera')
    tangerina = types.InlineKeyboardButton('Tangerina', callback_data='tangerina')
    uva = types.InlineKeyboardButton('Uva', callback_data='uva')
    melancia = types.InlineKeyboardButton('Melancia', callback_data='melancia')
    banana = types.InlineKeyboardButton('Banana', callback_data='banana')
    limao_tahiti = types.InlineKeyboardButton('Limao Tahiti', callback_data='limao_tahiti')
    maca_fuji = types.InlineKeyboardButton('Maca Fuji', callback_data='maca_fuji')
    maca_gala = types.InlineKeyboardButton('Maca Gala', callback_data='maca_gala')

    markup.add(abacate, abacaxi, mamao, maracuja, pera, tangerina, uva, melancia, banana, limao_tahiti, maca_fuji, maca_gala)

    bot.send_message(message.chat.id, 'Qual cotação gostaria de ver hoje? Preço/KG', reply_markup=markup)


@bot.callback_query_handler(func=lambda call:True)
def answer(callback):
    if callback.message:
        if callback.data == 'abacate':
            abacate_dados = pd.read_csv('data/abacate_data.csv')
            bot.send_message(callback.message.chat.id, f"🥑Cotação do Abacate🥑 Data: {data_formatada}")
            for i in range(len(abacate_dados)):
```

Fonte: Elaborado pelos Autores (2024).

Na Imagem 4, na Imagem 5 e na imagem 6, o código processa *callbacks* para diferentes frutas e envia mensagens com suas informações. Para cada fruta (como 'abacaxi', 'banana', etc.), ele lê os dados de um arquivo CSV específico (por exemplo, 'data/abacaxi_data.csv'). Em seguida, envia uma mensagem inicial com a cotação da fruta e a data formatada. Itera sobre as linhas dos dados, criando e enviando mensagens com o local e o preço da fruta. Após processar todas as linhas, envia uma linha de separação ('-' * 40). Isso é feito para cada fruta, formatando e enviando as informações via *bot*.

Imagem 4 – Trecho do código do *bot.*

```
38                mensagem = f"{abacate_dados['Local'].iloc[i]} : {abacate_dados['Preço'].iloc[i]}"
39                bot.send_message(callback.message.chat.id, mensagem)
40            bot.send_message(callback.message.chat.id, '-' *40)
41
42        elif callback.data == 'abacaxi':
43            abacaxi_dados = pd.read_csv('data/abacaxi_data.csv')
44            bot.send_message(callback.message.chat.id, f'🍍Cotação do Abacaxi🍍 Data: {data_formatada}')
45            for i in range(len(abacaxi_dados)):
46                mensagem = f"{abacaxi_dados['Local'].iloc[i]} : {abacaxi_dados['Preço'].iloc[i]}"
47                bot.send_message(callback.message.chat.id, mensagem)
48            bot.send_message(callback.message.chat.id, '-' *40)
49
50
51        elif callback.data == 'banana':
52            banana_dados = pd.read_csv('data/banana_data.csv')
53            bot.send_message(callback.message.chat.id, f'🍌Cotação da Banana🍌 Data: {data_formatada}')
54            for i in range(len(banana_dados)):
55                mensagem = f"{banana_dados['Local'].iloc[i]} : {banana_dados['Preço'].iloc[i]}"
56                bot.send_message(callback.message.chat.id, mensagem)
57            bot.send_message(callback.message.chat.id, '-' *40)
58
59        elif callback.data == 'limao_tahiti':
60            limao_tahiti_dados = pd.read_csv('data/limao_tahiti_data.csv')
61            bot.send_message(callback.message.chat.id, f'🍋Cotação do Limao Tahiti🍋 Data: {data_formatada}')
62            for i in range(len(limao_tahiti_dados)):
63                mensagem = f"{limao_tahiti_dados['Local'].iloc[i]} : {limao_tahiti_dados['Preço'].iloc[i]}"
64                bot.send_message(callback.message.chat.id, mensagem)
65            bot.send_message(callback.message.chat.id, '-' *40)
66
67        elif callback.data == 'maca_fuji':
68            bot.send_message(callback.message.chat.id, f'🍎Cotação da Maça Fuji🍎 Data: {data_formatada}')
69            maca_fuji_dados = pd.read_csv('data/maca_fuji_data.csv')
70            for i in range(len(maca_fuji_dados)):
71                mensagem = f"{maca_fuji_dados['Local'].iloc[i]} : {maca_fuji_dados['Preço'].iloc[i]}"
72                bot.send_message(callback.message.chat.id, mensagem)
73            bot.send_message(callback.message.chat.id, '-' *40)
```

Fonte: Elaborado pelos Autores (2024).

Imagem 5 – Trecho do código do *bot.*

```
73          bot.send_message(callback.message.chat.id, '-' *40)
74
75      elif callback.data == 'maca_gala':
76          bot.send_message(callback.message.chat.id, f'🍎Cotação da Maça🍎 Data: {data_formatada}')
77          maca_gala_dados = pd.read_csv('data/maca_gala_data.csv')
78          for i in range(len(maca_gala_dados)):
79              mensagem = f"{maca_gala_dados['Local'].iloc[i]} : {maca_gala_dados['Preço'].iloc[i]}"
80              bot.send_message(callback.message.chat.id, mensagem)
81          bot.send_message(callback.message.chat.id, '-' *40)
82
83      elif callback.data == 'mamao':
84          mamao_dados = pd.read_csv('data/mamao_data.csv')
85          bot.send_message(callback.message.chat.id, f'🍋Cotação do Mamão🍋 Data: {data_formatada}')
86          for i in range(len(mamao_dados)):
87              mensagem = f"{mamao_dados['Local'].iloc[i]} : {mamao_dados['Preço'].iloc[i]}"
88              bot.send_message(callback.message.chat.id, mensagem)
89          bot.send_message(callback.message.chat.id, '-' *40)
90
91      elif callback.data == 'maracuja':
92          maracuja_dados = pd.read_csv('data/maracuja_data.csv')
93          bot.send_message(callback.message.chat.id, f'Cotação do Maracuja Data: {data_formatada}')
94          for i in range(len(maracuja_dados)):
95              mensagem = f"{maracuja_dados['Local'].iloc[i]} : {maracuja_dados['Preço'].iloc[i]}"
96              bot.send_message(callback.message.chat.id, mensagem)
97          bot.send_message(callback.message.chat.id, '-' *40)
98
99      elif callback.data == 'melancia':
100         melancia_dados = pd.read_csv('data/melancia_data.csv')
101         bot.send_message(callback.message.chat.id, f'🍉Cotação da melancia🍉 Data: {data_formatada}')
102         for i in range(len(melancia_dados)):
103             mensagem = f"{melancia_dados['Local'].iloc[i]} : {melancia_dados['Preço'].iloc[i]}"
104             bot.send_message(callback.message.chat.id, mensagem)
105         bot.send_message(callback.message.chat.id, '-' *40)
106
107     elif callback.data == 'pera':
108         pera dados = pd.read csv('data/pera data.csv')
```

Fonte: Elaborado pelos Autores (2024).

A imagem 6 abaixo é a última parte do código fonte, o código ‘*else’* lida com qualquer valor de callback.data que não seja uma das opções, enviando uma mensagem de erro. E no código ‘bot.polling()’ faz com que ele comece a ouvir e responder às mensagens.

Imagem 6 – Foto do código do *bot.*

```
109             bot.send_message(callback.message.chat.id, f'🍐 Cotação da pera 🍐 Data: {data_formatada}')
110             for i in range(len(pera_dados)):
111                 mensagem = f"{pera_dados['Local'].iloc[i]} : {pera_dados['Preço'].iloc[i]}"
112                 bot.send_message(callback.message.chat.id, mensagem)
113             bot.send_message(callback.message.chat.id, '-' *40)
114
115         elif callback.data == 'tangerina':
116             tangerina_dados = pd.read_csv('data/tangerina_data.csv')
117             bot.send_message(callback.message.chat.id, f'🍊Cotação da Tangerina🍊 Data: {data_formatada}')
118             for i in range(len(tangerina_dados)):
119                 mensagem = f"{tangerina_dados['Local'].iloc[i]} : {tangerina_dados['Preço'].iloc[i]}"
120                 bot.send_message(callback.message.chat.id, mensagem)
121             bot.send_message(callback.message.chat.id, '-' *40)
122
123         elif callback.data == 'uva':
124             uva_dados = pd.read_csv('data/uva_data.csv')
125             bot.send_message(callback.message.chat.id, f'🍇Cotação da uva🍇 Data: {data_formatada}')
126             for i in range(len(uva_dados)):
127                 mensagem = f"{uva_dados['Local'].iloc[i]} : {uva_dados['Preço'].iloc[i]}"
128                 bot.send_message(callback.message.chat.id, mensagem)
129             bot.send_message(callback.message.chat.id, '-' *40)
130
131         else:
132             bot.send_message(callback.message.chat.id, 'ERROR')
133
134 bot.polling()
```

Fonte: Elaborado pelos Autores (2024).

4. Considerações Finais

O desenvolvimento do *bot* para consulta de cotações de frutas no Telegram demonstrou ser uma solução eficiente e prática para disponibilizar informações atualizadas aos usuários. Durante o processo, foram seguidas etapas importantes que garantiram a integridade e a funcionalidade do sistema, desde a coleta e processamento de dados até a integração e implementação no ambiente do Telegram.

O uso da biblioteca Selenium para *web scraping* se mostrou eficaz na extração dos dados de preços de frutas do site "Notícias Agrícolas". A implementação do *script* para navegar e coletar informações de várias frutas permitiu a criação de uma base de dados consistente e atualizada. No entanto, é importante monitorar possíveis mudanças na estrutura do site que possam afetar o funcionamento do scraping.

A organização dos dados em arquivos CSV foi uma escolha acertada, pois facilitou a manipulação e a leitura dos dados pelo *bot*. O uso da biblioteca Pandas permitiu uma fácil limpeza e organização dos dados, garantindo que apenas

informações relevantes fossem armazenadas e acessadas. A estrutura modular dos arquivos CSV por tipo de fruta assegurou uma consulta rápida e específica.

REFERÊNCIAS BIBLIOGRÁFICAS

NOTÍCIAS AGRÍCOLAS. **Cotações de frutas**. 2024. Disponível em: https://www.noticiasagricolas.com.br/cotacoes/frutas. Acesso em: 05.mar.2024.