```
import base64
import requests
from bs4 import BeautifulSoup
import pandas as pd
from urllib.parse import urlparse, parse_qs
import streamlit as st
from streamlit import session_state as state

st.markdown("<h2 style='text-align: center; background-color: #f1f1f1; color: #035689;
padding: 8px; margin-bottom: 10px; box-shadow: 0px 2px 5px rgba(0, 0, 0, 0.1); font-
family: Georgia;'>Flipkart Scraping Tool</h2>", unsafe_allow_html=True)
st.write('##')
# Initialize session state
if 'urls' not in state:
    state.urls = []

num_urls = st.number_input("**Enter the number of URLs:**", value=len(state.urls) + 1,
min_value=1, step=1)

for i in range(num_urls):
    url = st.text_input(f"**Enter URL {i+1}:**", key=f"url_{i}")

    if st.button(f"Add URL {i+1}", key=f"add_{i}"):
        if url and url not in state.urls:
            state.urls.append(url)
            st.success("URL added successfully.")
        elif url in state.urls:
            st.warning("URL already exists. Skipping...")
        else:
            st.warning("Please enter a valid URL.")

#st.write("URLs entered:")
#for url in state.urls:
#    st.write(url)
# Create a list of the entered URLs
url_list = state.urls
st.write("**URL List:**")
st.write(url_list)

def get_product_info(url):
    # Send an HTTP GET request and retrieve the webpage content
    response = requests.get(url)
    content = response.content

    # Parse the HTML content
    soup = BeautifulSoup(content, 'html.parser')

    # Extract the name of product
    prod_name = soup.find('div', {'class': 'aMaAEs'})
    name = prod_name.text.strip() if prod_name else 'N/A'

    # Extract the category of product
    a_tag = soup.find('a',{'class': '_2whKao'})
    parent_div = a_tag.find_parent('div',{'class':'_3GIHBu'})
    sibling_div = parent_div.find_next_sibling('div', {'class':'_3GIHBu'})
```

```
    category = sibling_div.find('a',{'class':'_2whKao'}).text

    # Extract the Star out of 5
    prod_star = soup.find('div', {'class': '_3LWZlK'})
    star = prod_star.text.strip() if prod_star else 'N/A'

    # Ratings and Reviews
    span_tag = soup.find('span', {'class' : '_2_R_DZ'})
    ratings = span_tag.find('span').text.strip().split()[0]

    reviews_span = span_tag.find_all('span')[-1]
    reviews = reviews_span.text.strip() if reviews_span else "NA"

    # Extract the product cost
    cost_element = soup.find('div', {'class': '_30jeq3 _16Jk6d'})
    cost = cost_element.text.strip() if cost_element else 'N/A'

    # Extract the seller information
    seller_element = soup.find('div', {'class': '_1RLviY'})
    seller = seller_element.text.strip() if seller_element else 'N/A'

    # Extract other sellers
    other_sellers = []
    element = soup.find('a', string='See other sellers')

    # Parse the href link to a variable
    href_link = None
    if element:
      href_link = element['href']
      other_sellers.append('https://www.flipkart.com'+href_link)

    # Extract Flipkart Serial Number
    parsed_url = urlparse(url)
    query_params = parse_qs(parsed_url.query)
    serial_number = query_params.get('pid', [''])[0]

    # Return the extracted information
    return {
        'Flipkart Serial Number': serial_number,
        'Product URL': url,
        'Product Name': name,
        'Product Category': category,
        'Star Rating': star,
        'Count of Ratings': ratings,
        'Reviews': reviews,
        'Cost of product': cost,
        'Seller': seller,
        'Other Sellers': ', '.join(other_sellers)
    }

# Create an empty DataFrame
data = []

# Extract information from each product URL
for url in state.urls:
```

```
    product_info = get_product_info(url)
    data.append(product_info)

if data:
    # Convert the data to a DataFrame
    df = pd.DataFrame(data)
    df['Seller'] = df['Seller'].str[:-3]
    st.dataframe(df)

    csv = df.to_csv(index=False)
    b64 = base64.b64encode(csv.encode()).decode()
    button_label = "Download Data"
    button_text = f'<a href="data:file/csv;base64,{b64}" download="data.csv"><button
style="background-color: white; color: #035689; border: 2px solid
#035689;">{button_label}</button></a>'
    st.markdown(button_text, unsafe_allow_html=True)
else:
    st.write("No data available.")
```