Konecta Internship Task 2 (Web Scraping)

Name: Ahmed Ayman Ahmed Alhofy

Track: Artificial Intelligence & Machine Learning

Repository Link: https://github.com/AhmedAyman4/konecta-internship/tree/main/Task-2

In [2]:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd
import time

# Lists to store data
titles = []
prices = []
ratings = []
product_links = []
image_links = []


for page in range(1, 6):
    url = f"https://www.noon.com/egypt-en/eg-gaming-laptops/?page={page}"
    headers = {
    "accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed
    "accept-language": "en,ar-AE;q=0.9,ar;q=0.8,en-US;q=0.7",
    "cache-control": "max-age=0",
    "priority": "u=0, i",
    "sec-ch-ua": "\"Not)A;Brand\";v=\"8\", \"Chromium\";v=\"138\", \"Google Chrome\";v=\"138\"",
    "sec-ch-ua-mobile": "?1",
    "sec-ch-ua-platform": "\"Android\"",
    "sec-fetch-dest": "document",
    "sec-fetch-mode": "navigate",
    "sec-fetch-site": "same-origin",
    "sec-fetch-user": "?1",
    "upgrade-insecure-requests": "1",
    "user-agent": "Mozilla/5.0 (Linux; Android 6.0; Nexus 5 Build/MRA58N) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/138.0.0.0
}


    response = requests.get(url, headers=headers)
    print(f"Page {page} status:", response.status_code)

    soup = BeautifulSoup(response.content, "html.parser")
    products = soup.select('a.ProductBoxLinkHandler_productBoxLink__FPhjp')
```

```
    if not products:
        print(f"No products found on page {page}")
        break

    for product in products:
        # Title
        title_tag = product.select_one("h2.ProductDetailsSection_title__JorAV")
        title = title_tag.text.strip() if title_tag else "N/A"
        titles.append(title)

        # Price
        price_tag = product.select_one("strong.Price_amount__2sXa7")
        price = price_tag.text.strip() if price_tag else "N/A"
        prices.append(price)

        # Rating
        rating_tag = product.select_one("div.RatingPreviewStar_textCtr__sfsJG")
        rating = rating_tag.text.strip() if rating_tag else "N/A"
        ratings.append(rating)

        # Product Link
        href = product.get('href')
        full_link = "https://www.noon.com" + href if href else "N/A"
        product_links.append(full_link)

        # Image Link
        img_tag = product.select_one("img.ProductImageCarousel_productImage__jtsOn")
        img_src = img_tag['src'] if img_tag else "N/A"
        image_links.append(img_src)

    print(f"✅ Scraped page {page} with {len(products)} products.")
    time.sleep(1)  # Be polite


# Create DataFrame
df = pd.DataFrame({
    "Product_name": titles,
    "Rating": ratings,
    "Price": prices,
    "Product_link": product_links,
    "Image_link": image_links
})

# Output
print("\nDataFrame shape:", df.shape)
print("Number of items scraped:", len(df))
```

```python
df.to_csv("noon_gaming_laptops_v2.csv", index=False, encoding='utf-8')
print("DataFrame saved successfully as 'noon_gaming_laptops_v2.csv'")
```

```
Page 1 status: 200
✅ Scraped page 1 with 50 products.
Page 2 status: 200
✅ Scraped page 2 with 50 products.
Page 3 status: 200
✅ Scraped page 3 with 50 products.
Page 4 status: 200
✅ Scraped page 4 with 50 products.
Page 5 status: 200
✅ Scraped page 5 with 50 products.

DataFrame shape: (250, 5)
Number of items scraped: 250
DataFrame saved successfully as 'noon_gaming_laptops_v2.csv'
```