Script Workflow

1. **Fetching Webpage Content**:
    - The script sends an HTTP GET request to the Flipkart search URL using the `requests` library.
    - It includes a `User-Agent` header to mimic a browser request and avoid anti-scraping blocks.
    - It then waits for 2 seconds between requests using `time.sleep(2)` to avoid overwhelming the server and bypassing anti scraping mechanisms.

2. **Parsing and Extracting Product Details**:

- The script uses the `BeautifulSoup` library to parse the HTML content.
- It identifies product containers using class names and extracts:
    - **Product Name**: Located in a `div` with the class `KzDlHZ`
    - **Price**: Located in a `div` with the class `Nx9bqj _4b5DiR`
    - **Rating**: Located in a `div` with the class `XQDdHH`
- Each product's details are stored in a list of dictionaries.

3. **Saving Extracted Data to CSV**:

- The script writes the extracted product details to a CSV file.
- The file includes headers: **Name**, **Price**, and **Rating**.

Customization Options

1. **Change Search URL**:

    Modify the `url` variable in the `main()` function to scrape other products. Replace laptops with the product you want. For example: `url = "https://www.flipkart.com/search?q=mobiles"`

    You can add sorting to your query by modifying the URL. For example:

    Sort by **Price** (ascending): `url = "https://www.flipkart.com/search?q=laptops&sort=price_asc"`

    Sort by **Rating** (descending): `url = "https://www.flipkart.com/search?q=laptops&sort=rating_desc"`

2. **Extract Additional Data**:

    Update the `extract` function to include other details like **discounts** or **product links** by inspecting the webpage's HTML.

3. **Output File Name**:

    Change the default CSV file name in the `save` function: