```
import requests
from bs4 import BeautifulSoup
import json
import time
import re
from datetime import datetime
```

```
import requests
from bs4 import BeautifulSoup
import json
import time
import re
from datetime import datetime
import sys

def scrape_bbc_ukraine_war_news(max_pages=1112):
    """
    Scrape BBC news articles about the Ukraine war by iterating through page numbers.

    Args:
        max_pages: Maximum number of pages to scrape (default: 1112 as specified)

    Returns:
        Dictionary with headlines as keys and [date, url] as values
    """
    # Dictionary to store all headlines
    all_headlines = {}

    # Base URL for the search
    base_url = "https://www.bbc.com"

    # Headers to mimic a browser request
    headers = {
        'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36',
        'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8',
        'Accept-Language': 'en-US,en;q=0.5',
        'Connection': 'keep-alive',
        'Upgrade-Insecure-Requests': '1',
        'Cache-Control': 'max-age=0'
    }

    # Iterate through pages directly
    for page in range(max_pages + 1):  # +1 to include page 0
        print(f"Scraping page {page}")

        # Construct the URL for the current page
        # Note: We're not using the provided token as it's page-specific and likely expires
        # Instead, we'll use a clean URL and rely on the server to redirect/authenticate
        url = f"{base_url}/search?q=ukraine+war&page={page}"

        try:
            # Make the request to the website
            response = requests.get(url, headers=headers)

            # Check if the page exists
            if response.status_code != 200:
                print(f"Received status code {response.status_code} for page {page}. Stopping.")
                break

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

            # Find all headlines using the specific CSS selector
            headline_elements = soup.select('#main-content > div.sc-32f23d22-0.heeShB > div > div.sc-32f23d22-2.iumrhG > div > div > div > a

            # As a fallback, also try to find elements by data-testid
            if not headline_elements:
                headline_elements = soup.find_all('h2', {'data-testid': 'card-headline'})

            # Check if we found any headlines
            if not headline_elements:
                print(f"No articles found on page {page}. This page might not exist.")

                # If we hit 3 empty pages in a row, assume we've reached the end
                if page > 2:
```

```
                    break
                else:
                    continue

            print(f"Found {len(headline_elements)} headlines on page {page}")

            # Extract information from each headline
            for headline_element in headline_elements:
                # Get the headline text
                headline = headline_element.get_text(strip=True)

                # Find the parent article container (going up the DOM tree)
                card = headline_element
                for _ in range(6):  # Go up a few levels to find the container with the link
                    if card.parent:
                        card = card.parent
                    else:
                        break

                # Extract URL
                # Look for an anchor tag within the card element
                link_element = card.find('a')
                if not link_element:
                    continue  # Skip if no link found

                article_url = link_element.get('href')
                if article_url.startswith('/'):
                    article_url = base_url + article_url

                # Extract publication date
                # First try with CSS selector for the date
                date_element = card.select_one('span[data-testid="card-metadata-lastupdated"]')
                if not date_element:
                    # Try alternative selector if needed
                    date_element = card.select_one('.sc-6fba5bd4-1')

                if not date_element:
                    publication_date = "Date not available"
                else:
                    date_text = date_element.get_text(strip=True)
                    # Try to parse and format the date
                    try:
                        # Example formats: "16 Feb 2025", "2 hours ago", "Yesterday"
                        if re.match(r'\d+ \w+ \d{4}', date_text):  # Format: "16 Feb 2025"
                            date_obj = datetime.strptime(date_text, '%d %b %Y')
                            publication_date = date_obj.strftime('%m-%d-%Y')
                        else:
                            publication_date = date_text  # Keep original text for relative dates
                    except Exception as e:
                        publication_date = date_text  # Keep original text if parsing fails

                # Add to our dictionary
                all_headlines[headline] = [publication_date, article_url]

            # Be polite and not overload the server
            time.sleep(2)

        except Exception as e:
            print(f"Error on page {page}: {str(e)}")
            # Continue with the next page instead of breaking
            continue

    return all_headlines

def main():
    print("Starting BBC Ukraine War news scraper...")

    # Set recursion limit higher for complex HTML
    sys.setrecursionlimit(10000)

    # Scrape with the maximum page number specified
    headlines = scrape_bbc_ukraine_war_news(max_pages=1112)

    # Save to JSON file
    with open('bbc_ukraine_war_headlines.json', 'w', encoding='utf-8') as f:
        json.dump(headlines, f, ensure_ascii=False, indent=4)

    print(f"Scraping complete. Found {len(headlines)} headlines.")
```

```
    print("Data saved to bbc_ukraine_war_headlines.json")

    # Print a sample of the results
    print("\nResults (first 5 entries):")
    result_sample = dict(list(headlines.items())[:5])
    print(json.dumps(result_sample, ensure_ascii=False, indent=4))
    print(f"... and {len(headlines) - 5} more entries")

if __name__ == "__main__":
    main()
```

```
Starting BBC Ukraine War news scraper...
Scraping page 0
Found 9 headlines on page 0
Scraping page 1
Found 9 headlines on page 1
Scraping page 2
Found 9 headlines on page 2
Scraping page 3
Found 9 headlines on page 3
Scraping page 4
Found 9 headlines on page 4
Scraping page 5
Found 9 headlines on page 5
Scraping page 6
Found 9 headlines on page 6
Scraping page 7
Found 9 headlines on page 7
Scraping page 8
Found 9 headlines on page 8
Scraping page 9
Found 9 headlines on page 9
Scraping page 10
Found 9 headlines on page 10
Scraping page 11
Found 9 headlines on page 11
Scraping page 12
Found 9 headlines on page 12
Scraping page 13
Found 9 headlines on page 13
Scraping page 14
Found 9 headlines on page 14
Scraping page 15
Found 9 headlines on page 15
Scraping page 16
Found 9 headlines on page 16
Scraping page 17
Found 9 headlines on page 17
Scraping page 18
Found 9 headlines on page 18
Scraping page 19
Found 9 headlines on page 19
Scraping page 20
Found 9 headlines on page 20
Scraping page 21
Found 9 headlines on page 21
Scraping page 22
Found 9 headlines on page 22
Scraping page 23
Found 9 headlines on page 23
Scraping page 24
Found 9 headlines on page 24
Scraping page 25
Found 9 headlines on page 25
Scraping page 26
Found 9 headlines on page 26
Scraping page 27
Found 9 headlines on page 27
```

```
import json
import pandas as pd
import csv
from google.colab import files
import os

# First, let's make sure the JSON file exists
try:
    # Open the JSON file
    with open('bbc_ukraine_war_headlines.json', 'r', encoding='utf-8') as file:
        data = json.load(file)
```

```
    print(f"Successfully loaded JSON file with {len(data)} headlines")

    # Convert the JSON structure to a format suitable for CSV
    # The JSON is in format: {"headline": ["date", "url"], ...}
    csv_data = []
    for headline, details in data.items():
        # Extract date and URL from the details array
        if len(details) >= 2:
            date, url = details[0], details[1]
        else:
            # Handle cases where the details array might not have both elements
            date = details[0] if len(details) > 0 else "No date"
            url = details[1] if len(details) > 1 else "No URL"

        # Add to the list of rows
        csv_data.append({
            "Headline": headline,
            "Date": date,
            "URL": url
        })

    # Create a DataFrame
    df = pd.DataFrame(csv_data)

    # Save to CSV
    csv_filename = 'bbc_ukraine_war_headlines.csv'
    df.to_csv(csv_filename, index=False, encoding='utf-8-sig')  # utf-8-sig for Excel compatibility

    print(f"Successfully converted to CSV with {len(df)} rows")

    # Download both files
    print("Downloading files...")
    files.download('bbc_ukraine_war_headlines.json')
    files.download(csv_filename)

    print("Files downloaded successfully!")

    # Print a sample of the data
    print("\nSample of the CSV data (first 5 rows):")
    print(df.head())

except FileNotFoundError:
    print("Error: The file 'bbc_ukraine_war_headlines.json' was not found.")
    print("Make sure you ran the scraper first and the file was created successfully.")

except Exception as e:
    print(f"An error occurred: {str(e)}")
```

```
Successfully loaded JSON file with 8388 headlines
Successfully converted to CSV with 8388 rows
Downloading files...
Files downloaded successfully!

Sample of the CSV data (first 5 rows):
                                            Headline        Date  \
0              Hundreds gather at Ukraine war vigil  7 days ago
1                Ukraine war: US-Russia peace talks  02-19-2025
2  Performance marks third anniversary of Ukraine...  5 days ago
3         Diplomacy gathers pace over war in Ukraine  02-16-2025
4    Fact-checking Trump claims about war in Ukraine  02-19-2025

                                                 URL
0  https://www.bbc.com/news/articles/cqlyrkgkde5o
1         https://www.bbc.com/audio/play/p0ks6jlm
2  https://www.bbc.com/news/articles/c778jm8pm4eo
3         https://www.bbc.com/audio/play/p0krp07v
4  https://www.bbc.com/news/articles/c9814k2jlxko
```