perplexity

Code for Collecting Hiking Safety Tips & Wilderness Guides

Here's comprehensive code to gather safety content from authoritative Canadian sources:[1] [2] [3]

Complete Safety Content Scraper

```
import requests
from bs4 import BeautifulSoup
import time
from pathlib import Path
import json
import re

# Create directory structure
Path('data/safety/adventuresmart').mkdir(parents=True, exist_ok=True)
Path('data/safety/parks_canada').mkdir(parents=True, exist_ok=True)
Path('data/safety/provincial').mkdir(parents=True, exist_ok=True)
Path('data/safety/leave_no_trace').mkdir(parents=True, exist_ok=True)
Path('data/safety/wildlife').mkdir(parents=True, exist_ok=True)

def scrape_page_content(url, headers=None):
    """Generic function to scrape main content from a page"""
    if headers is None:
        headers = {
            'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36'
        }

    try:
        response = requests.get(url, headers=headers, timeout=30)
        response.raise_for_status()
        soup = BeautifulSoup(response.content, 'html.parser')

        # Remove script, style, nav, footer elements
        for element in soup(['script', 'style', 'nav', 'footer', 'header']):
            element.decompose()

        # Try to find main content area (common selectors)
        main_content = (
            soup.find('main') or
            soup.find('article') or
            soup.find('div', class_=re.compile('content|main|article', re.I)) or
            soup.find('body')
        )

        if main_content:
            # Extract text with structure
```

```
            text_parts = []

            for element in main_content.find_all(['h1', 'h2', 'h3', 'h4', 'p', 'li', 'blo
                text = element.get_text(strip=True)
                if text and len(text) > 10:  # Filter out very short texts
                    # Format headers differently
                    if element.name in ['h1', 'h2', 'h3', 'h4']:
                        text_parts.append(f"\n{'#' * int(element.name[^1])} {text}\n")
                    elif element.name == 'li':
                        text_parts.append(f"- {text}")
                    else:
                        text_parts.append(text)

            return '\n\n'.join(text_parts)

        return None

    except Exception as e:
        print(f"Error scraping {url}: {str(e)}")
        return None


# 1. AdventureSmart Canada - Comprehensive Safety Guide
def scrape_adventuresmart():
    """Scrape AdventureSmart Canada safety resources"""

    print("\n=== Scraping AdventureSmart Canada ===")

    adventuresmart_pages = {
        'hiking_safety': 'https://www.adventuresmart.ca/hiking/',
        'health_safety': 'https://www.adventuresmart.ca/health-safety/',
        'trip_planning': 'https://www.adventuresmart.ca/land/trip-planning/',
        'taking_essentials': 'https://www.adventuresmart.ca/land/essentials/',
        'training': 'https://www.adventuresmart.ca/land/training/',
        'weather_safety': 'https://www.adventuresmart.ca/land/weather/',
        'navigation': 'https://www.adventuresmart.ca/land/navigation/',
        'communications': 'https://www.adventuresmart.ca/land/communications/',
        'first_aid': 'https://www.adventuresmart.ca/land/first-aid/',
        'survival_skills': 'https://www.adventuresmart.ca/land/survival-skills/',
    }

    all_content = []

    for topic, url in adventuresmart_pages.items():
        print(f"  Scraping: {topic}...")
        content = scrape_page_content(url)

        if content:
            document = f"""
{'='*60}
TOPIC: {topic.replace('_', ' ').upper()}
SOURCE: AdventureSmart Canada
URL: {url}
{'='*60}

{content}
```

```
"""
            all_content.append(document)

            # Save individual file
            filename = f'data/safety/adventuresmart/{topic}.txt'
            with open(filename, 'w', encoding='utf-8') as f:
                f.write(document)

            print(f"    ✓ Saved {len(content)} characters")
        else:
            print(f"    ✗ Failed to extract content")

        time.sleep(1)

    # Save combined corpus
    with open('data/safety/adventuresmart_full_guide.txt', 'w', encoding='utf-8') as f:
        f.write('\n\n'.join(all_content))

    print(f"✓ AdventureSmart: {len(all_content)} sections scraped")
    return all_content


# 2. Parks Canada Safety Guidelines
def scrape_parks_canada_safety():
    """Scrape Parks Canada safety and visitor guidelines"""

    print("\n=== Scraping Parks Canada Safety ===")

    safety_pages = {
        'wilderness_safety': 'https://parks.canada.ca/voyage-travel/conseils-tips',
        'bear_safety': 'https://parks.canada.ca/nature/faune-wildlife/ours-bears/securite
        'backcountry_camping': 'https://parks.canada.ca/voyage-travel/conseils-tips/arrie
        'winter_safety': 'https://parks.canada.ca/voyage-travel/conseils-tips/hiver-winte
        'water_safety': 'https://parks.canada.ca/voyage-travel/conseils-tips/eau-water',
        'avalanche_safety': 'https://parks.canada.ca/voyage-travel/conseils-tips/avalanch
        'weather_preparation': 'https://parks.canada.ca/voyage-travel/conseils-tips/meteo
    }

    all_content = []

    for topic, url in safety_pages.items():
        print(f"  Scraping: {topic}...")
        content = scrape_page_content(url)

        if content:
            document = f"""
{'='*60}
TOPIC: {topic.replace('_', ' ').upper()}
SOURCE: Parks Canada
URL: {url}
{'='*60}

{content}
"""
            all_content.append(document)
```

```
            filename = f'data/safety/parks_canada/{topic}.txt'
            with open(filename, 'w', encoding='utf-8') as f:
                f.write(document)

            print(f"    ✓ Saved {len(content)} characters")
        else:
            print(f"    ✗ Failed")

        time.sleep(1)

    with open('data/safety/parks_canada_safety_guide.txt', 'w', encoding='utf-8') as f:
        f.write('\n\n'.join(all_content))

    print(f"✓ Parks Canada: {len(all_content)} sections scraped")
    return all_content


# 3. Provincial Parks Safety Guidelines
def scrape_provincial_safety():
    """Scrape provincial park safety guidelines"""

    print("\n=== Scraping Provincial Safety Guidelines ===")

    provincial_pages = {
        'bc_parks_safety': 'https://bcparks.ca/plan-your-trip/visit-responsibly/staying-s
        'bc_wildlife': 'https://bcparks.ca/plan-your-trip/visit-responsibly/wildlife-safe
        'alberta_hiking_safety': 'https://www.albertaparks.ca/parks/kananaskis/kananaskis
        'alberta_bear_safety': 'https://www.albertaparks.ca/parks/kananaskis/kananaskis-c
        'ontario_safety': 'https://www.ontarioparks.com/parksblog/safety-tips/',
    }

    all_content = []

    for topic, url in provincial_pages.items():
        print(f"  Scraping: {topic}...")
        content = scrape_page_content(url)

        if content:
            document = f"""
{'='*60}
TOPIC: {topic.replace('_', ' ').upper()}
SOURCE: {topic.split('_')[^0].upper()} Provincial Parks
URL: {url}
{'='*60}

{content}
"""
            all_content.append(document)

            filename = f'data/safety/provincial/{topic}.txt'
            with open(filename, 'w', encoding='utf-8') as f:
                f.write(document)

            print(f"    ✓ Saved")
        else:
            print(f"    ✗ Failed")
```

```
        time.sleep(1)

    with open('data/safety/provincial_safety_guide.txt', 'w', encoding='utf-8') as f:
        f.write('\n\n'.join(all_content))

    print(f"✓ Provincial: {len(all_content)} sections scraped")
    return all_content


# 4. Leave No Trace Principles
def scrape_leave_no_trace():
    """Scrape Leave No Trace principles and ethics"""

    print("\n=== Scraping Leave No Trace ===")

    lnt_pages = {
        'seven_principles': 'https://leavenotrace.ca/seven-principles',
        'plan_prepare': 'https://leavenotrace.ca/principle-1',
        'travel_camp': 'https://leavenotrace.ca/principle-2',
        'dispose_waste': 'https://leavenotrace.ca/principle-3',
        'leave_what_you_find': 'https://leavenotrace.ca/principle-4',
        'minimize_campfire': 'https://leavenotrace.ca/principle-5',
        'respect_wildlife': 'https://leavenotrace.ca/principle-6',
        'be_considerate': 'https://leavenotrace.ca/principle-7',
    }

    all_content = []

    for topic, url in lnt_pages.items():
        print(f"  Scraping: {topic}...")
        content = scrape_page_content(url)

        if content:
            document = f"""
{'='*60}
TOPIC: {topic.replace('_', ' ').upper()}
SOURCE: Leave No Trace Canada
URL: {url}
{'='*60}

{content}
"""
            all_content.append(document)

            filename = f'data/safety/leave_no_trace/{topic}.txt'
            with open(filename, 'w', encoding='utf-8') as f:
                f.write(document)

            print(f"    ✓ Saved")
        else:
            print(f"    ✗ Failed")

        time.sleep(1)

    with open('data/safety/leave_no_trace_guide.txt', 'w', encoding='utf-8') as f:
```

```
        f.write('\n\n'.join(all_content))

    print(f"✓ Leave No Trace: {len(all_content)} sections scraped")
    return all_content


# 5. Wildlife Safety (Bears, Cougars, etc.)
def scrape_wildlife_safety():
    """Scrape wildlife encounter safety guidelines"""

    print("\n=== Scraping Wildlife Safety ===")

    wildlife_pages = {
        'bear_safety_basics': 'https://www.pc.gc.ca/en/pn-np/mtn/ours-bears/sec',
        'bear_spray_use': 'https://www.pc.gc.ca/en/pn-np/mtn/ours-bears/generaux-general/
        'wildlife_viewing': 'https://www.pc.gc.ca/en/nature/faune-wildlife/observer',
        'cougar_safety': 'https://www.env.gov.bc.ca/fw/wildlife/living-with-wildlife/coug
        'moose_safety': 'https://www.gov.nl.ca/ecc/files/wildlife-moose-safety-tips-moose
    }

    all_content = []

    for topic, url in wildlife_pages.items():
        print(f"  Scraping: {topic}...")

        # Skip PDF for now (moose safety)
        if url.endswith('.pdf'):
            print(f"    ⚠ Skipping PDF")
            continue

        content = scrape_page_content(url)

        if content:
            document = f"""
{'='*60}
TOPIC: {topic.replace('_', ' ').upper()}
SOURCE: Canadian Wildlife Safety Resources
URL: {url}
{'='*60}

{content}
"""
            all_content.append(document)

            filename = f'data/safety/wildlife/{topic}.txt'
            with open(filename, 'w', encoding='utf-8') as f:
                f.write(document)

            print(f"    ✓ Saved")
        else:
            print(f"    ✗ Failed")

        time.sleep(1)

    with open('data/safety/wildlife_safety_guide.txt', 'w', encoding='utf-8') as f:
        f.write('\n\n'.join(all_content))
```

```
    print(f"✓ Wildlife: {len(all_content)} sections scraped")
    return all_content


# 6. Create Structured Safety Tips Dataset
def create_structured_safety_dataset():
    """
    Create a structured JSON dataset of safety tips by category
    Useful for RAG retrieval by topic
    """

    print("\n=== Creating Structured Safety Dataset ===")

    safety_categories = {
        'preparation': [
            "Check weather forecast before departure",
            "Create a trip plan and share it with someone reliable",
            "Pack the 10 essentials: navigation, sun protection, insulation, illumination
            "Bring extra food and water beyond your planned needs",
            "Wear appropriate footwear with good ankle support",
            "Dress in layers for changing weather conditions",
            "Carry a physical map and compass (don't rely solely on GPS)",
            "Ensure your phone is fully charged and bring a portable battery",
            "Download offline maps of your trail area",
            "Tell someone your expected return time"
        ],
        'navigation': [
            "Stay on marked trails to prevent getting lost",
            "If lost, use STOP: Stop, Think, Observe, Plan",
            "Use landmarks and your map to track your location",
            "Learn to use a compass before your trip",
            "Mark your route with cairns or ribbons in unmarked areas (remove when return
            "Travel during daylight hours when possible",
            "Turn back if weather deteriorates or you're running late",
            "Follow trail blazes: blue for main trails, yellow for side trails",
            "Don't shortcut switchbacks - it causes erosion"
        ],
        'weather': [
            "Check UV index and apply sunscreen liberally",
            "Avoid hiking between 11 AM and 3 PM during high heat",
            "Watch for signs of hypothermia: shivering, confusion, slurred speech",
            "Recognize heat exhaustion: pale skin, weakness, nausea, rapid pulse",
            "Seek shelter immediately if thunderstorms approach",
            "Lightning safety: avoid peaks, ridges, and solitary trees",
            "In winter, cover all exposed skin to prevent frostbite",
            "Windburn and sunburn can occur even on cloudy days",
            "Carry rain gear regardless of forecast"
        ],
        'hydration_nutrition': [
            "Drink water before you feel thirsty - thirst means you're already dehydrated
            "Ration your sweat, not your water",
            "Drink 2-3 liters of water per day minimum",
            "Never drink untreated water from streams or lakes",
            "Boil water for 2 minutes or use purification tablets",
            "Eat small snacks throughout the day to maintain energy",
```

```
            "High-energy foods: nuts, dried fruit, energy bars, jerky",
            "Don't eat snow - it lowers your core body temperature",
            "Avoid alcohol - it dehydrates you and impairs judgment",
            "Dark yellow urine indicates dehydration"
        ],
        'wildlife': [
            "Make noise while hiking - sing, talk, use bear bells",
            "Carry bear spray and keep it accessible (not in your pack)",
            "If you see a bear, back away slowly while facing it",
            "Never run from wildlife - it triggers chase instinct",
            "Keep food in bear-proof containers at least 200m from camp",
            "Never feed wildlife - a fed bear is a dead bear",
            "Hike in groups when possible for added safety",
            "Store scented items (toothpaste, deodorant) with food",
            "Report all wildlife sightings to park staff",
            "For cougars: make yourself look large, maintain eye contact, fight back if a
        ],
        'emergency': [
            "Call 911 for emergencies - rescue is free in Canada",
            "Use three whistle blasts to signal distress",
            "Signal SOS: 3 short, 3 long, 3 short (lights or whistle)",
            "If injured, stay put and make noise to attract searchers",
            "Build a fire for warmth and as a signal (where permitted)",
            "Use a space blanket or emergency shelter to prevent hypothermia",
            "Stay visible - use bright clothing, reflective items",
            "If bivouacking, insulate yourself from the ground",
            "Conserve phone battery - use airplane mode except when signaling",
            "Know basic first aid: CPR, treating fractures, stopping bleeding"
        ],
        'leave_no_trace': [
            "Pack out all trash - leave no trace of your visit",
            "Bury human waste 15-20cm deep, 70m from water sources",
            "Stay on established trails to prevent erosion",
            "Don't pick flowers, plants, or collect rocks/artifacts",
            "Camp on durable surfaces - established sites or rock/gravel",
            "Keep campsites small and away from water",
            "Use biodegradable soap 70m from water sources",
            "Respect quiet hours and other trail users",
            "Don't build new fire rings - use existing ones",
            "Take photos, not souvenirs"
        ],
        'fitness_health': [
            "Start with easy trails if you're new to hiking",
            "Build endurance gradually over the season",
            "Stretch before and after hiking to prevent injury",
            "Know your physical limits and respect them",
            "Take breaks every 30-60 minutes",
            "Use trekking poles to reduce knee strain",
            "Treat blisters immediately with moleskin",
            "Check for ticks after hiking in grassy areas",
            "Remove ticks with tweezers, pulling straight out",
            "Update tetanus vaccination before backcountry trips"
        ]
    }

    # Save as JSON
```

```
    with open('data/safety/safety_tips_structured.json', 'w', encoding='utf-8') as f:
        json.dump(safety_categories, f, indent=2, ensure_ascii=False)

    # Save as formatted text
    text_output = []
    for category, tips in safety_categories.items():
        text_output.append(f"\n{'='*60}")
        text_output.append(f"CATEGORY: {category.upper().replace('_', ' ')}")
        text_output.append('='*60)
        for i, tip in enumerate(tips, 1):
            text_output.append(f"{i}. {tip}")

    with open('data/safety/safety_tips_formatted.txt', 'w', encoding='utf-8') as f:
        f.write('\n'.join(text_output))

    total_tips = sum(len(tips) for tips in safety_categories.values())
    print(f"✓ Created structured dataset: {len(safety_categories)} categories, {total_ti

    return safety_categories


# 7. Main Execution
def main():
    """Main function to collect all safety content"""

    print("="*70)
    print(" CANADIAN WILDERNESS SAFETY CONTENT COLLECTION")
    print("="*70)

    # Run all scrapers
    adventuresmart_content = scrape_adventuresmart()
    parks_canada_content = scrape_parks_canada_safety()
    provincial_content = scrape_provincial_safety()
    lnt_content = scrape_leave_no_trace()
    wildlife_content = scrape_wildlife_safety()
    structured_tips = create_structured_safety_dataset()

    # Create master corpus
    print("\n=== Creating Master Safety Corpus ===")

    all_scraped = (
        adventuresmart_content +
        parks_canada_content +
        provincial_content +
        lnt_content +
        wildlife_content
    )

    with open('data/safety/MASTER_SAFETY_CORPUS.txt', 'w', encoding='utf-8') as f:
        f.write('\n\n'.join(all_scraped))

    # Statistics
    print("\n" + "="*70)
    print(" COLLECTION COMPLETE")
    print("="*70)
    print(f"Total Documents Scraped: {len(all_scraped)}")
```

```
    print(f"Structured Safety Tips: {sum(len(tips) for tips in structured_tips.values())}
    print("\nFiles created:")
    print("  - data/safety/MASTER_SAFETY_CORPUS.txt (full text corpus)")
    print("  - data/safety/safety_tips_structured.json (structured tips)")
    print("  - data/safety/safety_tips_formatted.txt (quick reference)")
    print("  - Individual files in subdirectories by source")

    # File size summary
    total_size = 0
    for root, dirs, files in os.walk('data/safety/'):
        for file in files:
            if file.endswith('.txt') or file.endswith('.json'):
                filepath = os.path.join(root, file)
                total_size += os.path.getsize(filepath)

    print(f"\nTotal Content Size: {total_size / 1024:.2f} KB")
    print("="*70)


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

Quick Run Script

```
# run_safety_scraper.py
"""
Quick script to collect just the essential safety content
"""

import requests
from bs4 import BeautifulSoup
from pathlib import Path

def quick_safety_collection():
    """Collect essential safety tips from key sources"""

    Path('data/safety').mkdir(parents=True, exist_ok=True)

    # Essential URLs
    urls = {
        'AdventureSmart Hiking': 'https://www.adventuresmart.ca/hiking/',
        'AdventureSmart Health': 'https://www.adventuresmart.ca/health-safety/',
        'BC Parks Safety': 'https://bcparks.ca/plan-your-trip/visit-responsibly/staying-s
        'Alberta Hiking Safety': 'https://www.albertaparks.ca/parks/kananaskis/kananaskis
    }

    headers = {'User-Agent': 'Mozilla/5.0'}
    all_content = []

    for title, url in urls.items():
        print(f"Fetching: {title}...")
        try:
            response = requests.get(url, headers=headers, timeout=30)
            soup = BeautifulSoup(response.content, 'html.parser')
```

```
            # Get main content
            main = soup.find('main') or soup.find('article') or soup.find('body')
            text = main.get_text(separator='\n\n', strip=True) if main else ""

            doc = f"""
{'='*60}
{title.upper()}
Source: {url}
{'='*60}

{text}
"""
            all_content.append(doc)
            print(f"  ✓ Collected {len(text)} characters")

        except Exception as e:
            print(f"  ✗ Error: {str(e)}")

    # Save combined
    output_file = 'data/safety/essential_safety_guide.txt'
    with open(output_file, 'w', encoding='utf-8') as f:
        f.write('\n\n'.join(all_content))

    print(f"\n✓ Saved to: {output_file}")
    print(f"Total sections: {len(all_content)}")

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

Installation & Usage

```
# Install dependencies
pip install requests beautifulsoup4

# Run full collection (10-15 minutes)
python collect_safety_content.py

# OR run quick collection (2-3 minutes)
python run_safety_scraper.py
```

Expected Output

After running, you'll have:

Structured by source:

- `data/safety/adventuresmart/` - 10+ files (hiking, health, navigation, etc.)[2] [1]
- `data/safety/parks_canada/` - 7+ files (bear safety, backcountry, winter, etc.)
- `data/safety/provincial/` - 5+ files (BC, Alberta, Ontario safety)[4] [5]
- `data/safety/leave_no_trace/` - 8 files (7 principles + overview)[6]

- `data/safety/wildlife/` - 4+ files (bears, cougars, viewing guidelines)

Master files:

- `MASTER_SAFETY_CORPUS.txt` - All scraped content combined (~200-300 KB)
- `safety_tips_structured.json` - 80+ tips organized by category[7]
- `safety_tips_formatted.txt` - Quick reference guide

Content coverage:[8] [1] [2] [4]

- ✓ The "3 Ts": Trip Planning, Training, Taking Essentials
- ✓ 10 Essential items for hiking
- ✓ Weather safety (hypothermia, heat exhaustion, frostbite)
- ✓ Wildlife encounters (bears, cougars, moose)
- ✓ Navigation and survival skills
- ✓ First aid and emergency procedures
- ✓ Leave No Trace principles
- ✓ Water purification and hydration
- ✓ Fire safety and shelter building
- ✓ Physical fitness and mental preparation

This comprehensive dataset will make your RAG system highly valuable for answering safety questions like "What should I pack for a day hike?" or "How do I handle a bear encounter?".[1] [7] [4] [8]

⁂

1. https://www.adventuresmart.ca/hiking/
2. https://www.adventuresmart.ca/health-safety/
3. https://www.adventuresmart.ca
4. https://www.albertaparks.ca/parks/kananaskis/kananaskis-country/advisories-public-safety/backcountry-safety/hiking-safety/
5. https://bcparks.ca/plan-your-trip/visit-responsibly/staying-safe/
6. https://tctrail.ca/stories/leave-no-trace/
7. https://www.whistler.com/blog/post/2023/06/06/whistler-hiking-and-backpacking-safety-tips/
8. https://www.frontenacpark.ca/safety-and-survival-skills/stay-safe-while-exploring-essential-health-protocols-for-ontario-parks/
9. https://www.nps.gov/articles/leave-no-trace-seven-principles.htm
10. https://www.adventuresmart.ca/9634/