Content audit to promote reuse

Overview

The purpose of this audit is to analyze, optimize, and maintain content reuse across SOPs and blurbs.

Expected outcomes

- Standardize content reuse across documents.
- Sustainable system for managing reusable content.
- Bi-annually improvement in reuse metrics.

This process aims to improve and manage content reuse, but it's important to note that not all content is suitable for reuse. Some content may be unique by necessity, and that is perfectly acceptable. These steps may not apply equally to all functions or content types.

Note: Complete the steps separately for SOP and blurb content.

Phase 1: Initial setup

1. Copy the following code and paste it into notepad.

```
import re
import os
from bs4 import BeautifulSoup

missing_files_dict = {}

def extract_word_count_from_html(html_content):

    # Create BeautifulSoup object
    soup = BeautifulSoup(html_content, 'html.parser')

    # Remove script and style elements
    for script in soup(["script", "style"]):
        script.decompose()

    # Get visible text
    text = soup.get_text(separator=' ', strip=True)

    # Clean up whitespace
    text = re.sub(r'\s+', ' ', text).strip()
    return len(text.split(" "))


def get_src_contents(text):
    block_pattern = r'<MadCap:snippetBlock\s+src="([^"]+)"'
    block_matches = re.findall(block_pattern, text)
    text_pattern = r'<MadCap:snippetText\s+src="([^"]+)"'
    text_matches = re.findall(text_pattern, text)
    block_matches.extend(text_matches)
    return block_matches

def count_words_in_file(file_path, source_file_path):
    global missing_files_dict
    try:
        with open(file_path, 'r', encoding='utf-8') as file:
            file_contents = file.read()
    except Exception as e:
        if missing_files_dict.get(source_file_path):
            missing_files_dict.get(source_file_path).append(file_path)
        else:
            missing_files_dict[source_file_path] = [file_path]
        print(f"Error reading file {file_path}: {e}")
        return 0
    return extract_word_count_from_html(file_contents)

def process_file(file_path, root_folder):
    try:
        with open(file_path, 'r', encoding='utf-8') as file:
            content = file.read()

        main_word_count = extract_word_count_from_html(content)
        src_list = get_src_contents(content)

        src_word_count = 0
        for src in src_list:
            full_src_path = os.path.join(root_folder, src)
            src_word_count += count_words_in_file(full_src_path, file_path)

        total_words = main_word_count + src_word_count
        src_percentage = (src_word_count / total_words) * 100 if total_words
> 0 else 0
```

```
        return main_word_count, src_word_count, len(src_list),
src_percentage

    except Exception as e:
        print(f"Error processing file {file_path}: {e}")
        return 0, 0, 0, 0

def process_folder(folder_path):
    total_main_words = 0
    total_src_words = 0
    total_src_files = 0
    processed_files = 0
    total_percentage = 0

    for filename in os.listdir(folder_path):
        if "SOPs" in folder_path:
            if filename.endswith(".htm") or filename.endswith(".html"):
                file_path = os.path.join(folder_path, filename)
                main_words, src_words, src_count, src_percentage =
process_file(file_path, folder_path)

                total_main_words += main_words
                total_src_words += src_words
                total_src_files += src_count
                total_percentage += src_percentage
                processed_files += 1

                print(f"Processed: {filename}")
                print(f"  Main file words: {main_words}")
                print(f"  Snippet files words: {src_words}")
                print(f"  Number of snippet files used: {src_count}")
                print(f"  Percentage of content from src files:
{src_percentage:.2f}%")
                print()
        else:
            if filename.endswith(".htm") or filename.endswith(".html"):
                file_path = os.path.join(folder_path, filename)
                main_words, src_words, src_count, src_percentage =
process_file(file_path, folder_path)

                total_main_words += main_words
                total_src_words += src_words
                total_src_files += src_count
                total_percentage += src_percentage
                processed_files += 1

                print(f"Processed: {filename}")
                print(f"  Main file words: {main_words}")
                print(f"  Snippet files words: {src_words}")
                print(f"  Number of snippet files used: {src_count}")
                print(f"  Percentage of content from src files:
{src_percentage:.2f}%")

    overall_total_words = total_main_words + total_src_words
    overall_src_percentage = (total_src_words / overall_total_words) * 100
if overall_total_words > 0 else 0
    average_src_percentage = total_percentage / processed_files if
processed_files > 0 else 0

    print(f"Total processed files: {processed_files}")
    print(f"Total words in main files: {total_main_words}")
    print(f"Total words in snippets files: {total_src_words}")
    print(f"Total snippets referenced: {total_src_files}")
    print(f"Grand total of words: {overall_total_words}")
    print(f"Overall percentage of content from src files:
{overall_src_percentage:.2f}%")

# Specify the folder path
folder_path = input("Enter the full content folder path: ").replace("\\", "\\\\")

# Process the folder
process_folder(folder_path)
for source_file, missing_list in missing_files_dict.items():
    print(f"\n\n{source_file} relies on missing file(s):")
    for item in missing_list:
        print(f"\t{item}")
input()
```

2. Save the file as "count_snippet_percentage_2.py" to your desktop.
3. Navigate to your functions content folder.
4. Create a new folder called "reuse_script", copy and paste all of your SOPs or blurb content into this folder.

Phase 2: Preparation download python

Ensure Python is downloaded to your machine, if yes, continue to Phase 2. If it is not complete the following steps.

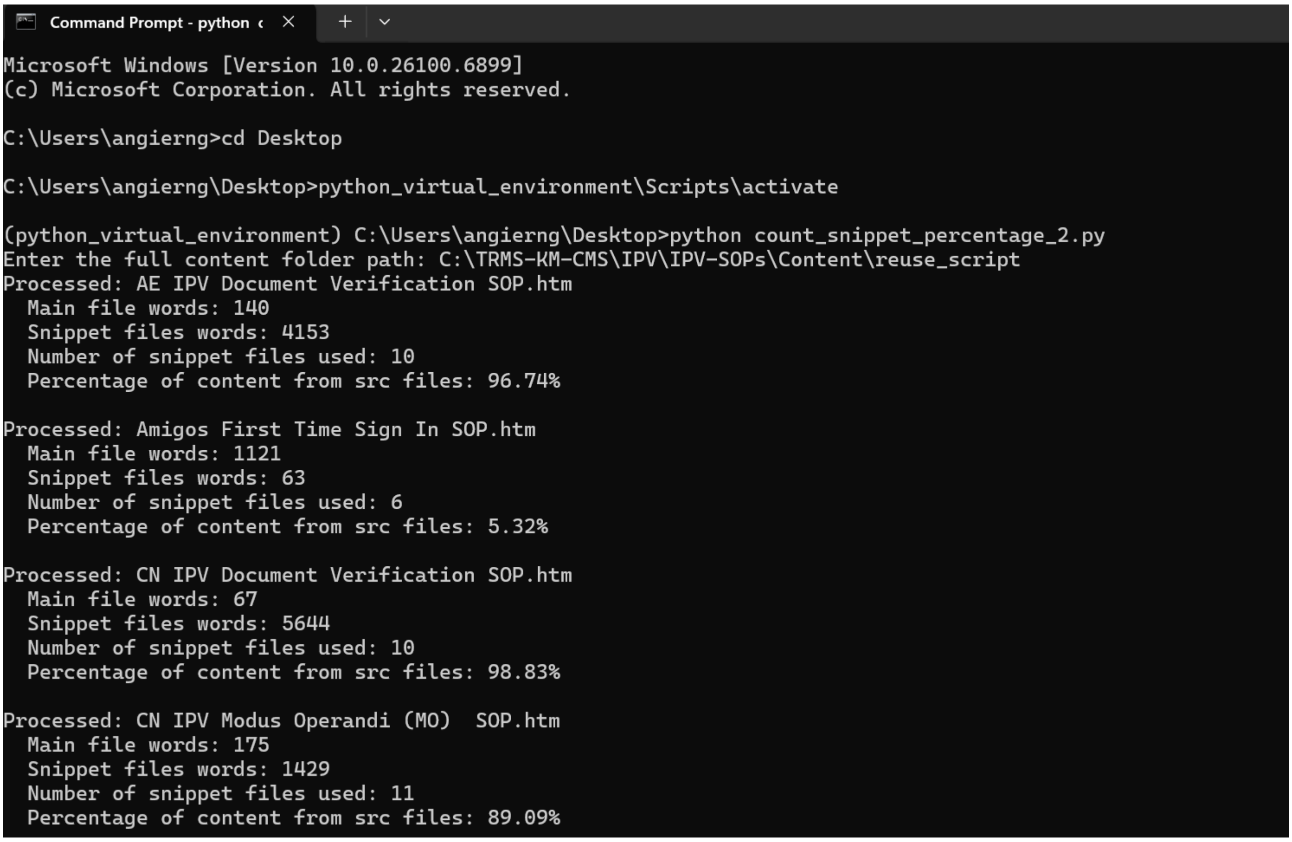
1. From the search field on your task bar, type **Software Center**, then press **Enter**.
2. In Software Center, search for Python. Click the result, then click **Install**.
3. Once Python is installed, open the Command prompt.
4. Type "cd Desktop" then press Enter.
5. Type "python -m venv python_virtual_environment" to create the virtual environment.
6. Type "python_virtual_environment\Scripts\activate" to activate the virtual environment.
7. Type "pip install beautifulsoup4" to install the HTML parser library.
8. Once installed go to the location of the count_snippet_percentage_2.py file.

Phase 3: Run the script to calculate the percentage of reuse

Now that the virtual environment is set up, follow these steps to run the script.

The virtual environment only needs to be set up once. For all future script runs start from here.

1. Open the command prompt if it is not already.
2. Type "cd Desktop" then press **Enter**.
3. Type "python_virtual_environment\Scripts\activate" then press **Enter**.
4. Type "python count_snippet_percentage_2.py" then press **Enter**.
5. Copy and paste one of the following file paths to check the percentage of reuse for either SOP or blurb content. Press **Enter,** the script will run and display the percentage of reuse for each document and the overall percentage.



```
Microsoft Windows [Version 10.0.26100.6899]
(c) Microsoft Corporation. All rights reserved.

C:\Users\angierng>cd Desktop

C:\Users\angierng\Desktop>python_virtual_environment\Scripts\activate

(python_virtual_environment) C:\Users\angierng\Desktop>python count_snippet_percentage_2.py
Enter the full content folder path: C:\TRMS-KM-CMS\IPV\IPV-SOPs\Content\reuse_script
Processed: AE IPV Document Verification SOP.htm
  Main file words: 140
  Snippet files words: 4153
  Number of snippet files used: 10
  Percentage of content from src files: 96.74%

Processed: Amigos First Time Sign In SOP.htm
  Main file words: 1121
  Snippet files words: 63
  Number of snippet files used: 6
  Percentage of content from src files: 5.32%

Processed: CN IPV Document Verification SOP.htm
  Main file words: 67
  Snippet files words: 5644
  Number of snippet files used: 10
  Percentage of content from src files: 98.83%

Processed: CN IPV Modus Operandi (MO)  SOP.htm
  Main file words: 175
  Snippet files words: 1429
  Number of snippet files used: 11
  Percentage of content from src files: 89.09%
```

Phase 4: Create a tracking table

1. Create an analysis tracking table with the following columns,

- **Document name**
- **Current reuse percentage**
- **Priority level** (this will depend on the level of reuse per document, start with the documents which have the lowest percentage of reuse)
  - High priority : 0%-30% (The document contains little to no reuse, there may be an opportunity to increase reuse)
  - Medium priority : 31%-60% (The document contains some reuse, there may be an opportunity to increase reuse)
  - Low priority : 61% - 90%+ (The document contains a high level of reuse, there still may be an opportunity to increase reuse)

Note: Low reuse percentages don't always indicate a problem. Some documents may contain highly specific or unique content that can't or shouldn't be reused. The goal is to

identify opportunities for reuse, not to force reuse where it doesn't make sense.

- **Date**
- **Document type**
- **Overall reuse percentage**

2. Choose one of the high priority documents and complete the following steps

Phase 5: Setup the MadCap environment to promote reuse

MadCap analysis setup

1. Navigate to **File** > **Options** > **Project Analysis** > **Advanced Scan Options.**
2. Check **Collect Segments** and **Collect Snippet Prospects** to enable the similar segments and snippet prospects analysis
3. Run Similar Segments analysis.
    - Optional: this may flag the same content which is written slightly differently.
    - Optional: create snippets with the overlapped content if it makes sense.

MadCap auto snippet setup

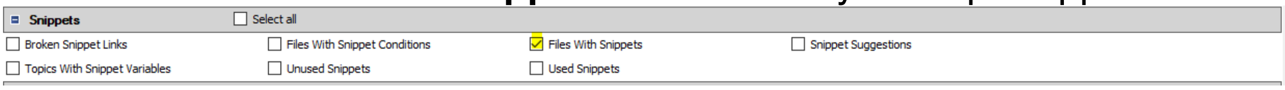
1. Navigate to **File** > **Options** > **Auto Suggestion** tab.
2. Check **Enable Snippet Auto Suggestion.**
3. Search **Snippet Suggestion** to open the window.

- Optional: this may flag content which can be snippetized.
- Optional: create snippets with the suggested content if it makes sense to do so.

Phase 6: Assess existing content

Run a report on existing snippets

1. Use dynamic reports to view files with snippets.
2. Open the **Project Organizer.**
3. Right click on **Reports**, click **Add report file.**
4. Name the report **snippets**.
5. Filter for **Files with Snippets.** Uncheck any other pre-applied filters.



⊟ Snippets □ Select all

□ Broken Snippet Links □ Files With Snippet Conditions ☑ Files With Snippets □ Snippet Suggestions

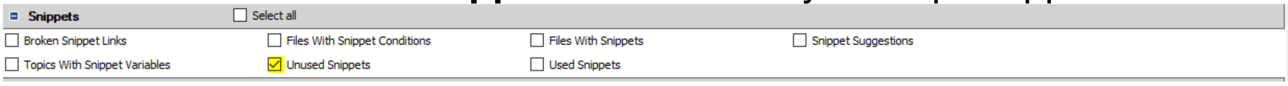
□ Topics With Snippet Variables □ Unused Snippets □ Used Snippets

6. Save the file and then click **Generate.**
7. Once the report is generated, click **Open in Browser**.

8. This displays all the documents and the snippets within them, locate the SOP you are currently auditing, locate each snippet and ensure there is common naming conventions for example **[ContentType]** ***[Category]*****[SpecificIdentifier]**.

Run a report on unused snippets

1. Use dynamic reports to view files with snippets.
2. Open the **Project Organizer.**
3. Right click on **Reports**, click **Add report file.**
4. Name the report **snippets**.
5. Filter for **Unused Snippets**. Uncheck any other pre-applied filters.



⊟ **Snippets** ☐ Select all

☐ Broken Snippet Links ☐ Files With Snippet Conditions ☐ Files With Snippets ☐ Snippet Suggestions

☐ Topics With Snippet Variables ☑ Unused Snippets ☐ Used Snippets

6. Save the file and then click **Generate.**
7. Once the report is generated, click **Open in Browser**.
8. Go through all of the unused snippets and clean up this list, decide to either use, merge, or delete the snippet.

Is there an opportunity for inline snippets?

1. Look through the document and check for content which can be turned into inline snippets.
2. Consider creating inline snippets for the following content,

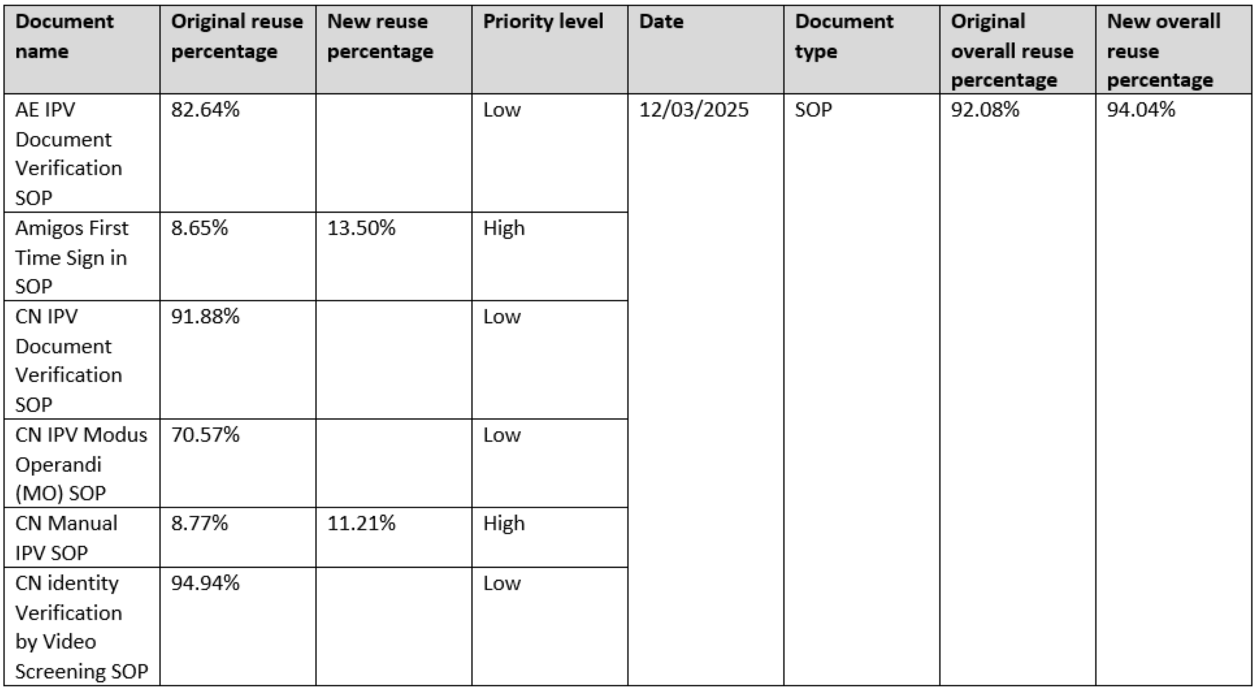
- Product names
- Legal text
- Standard phrases
- Warning, notes, or caution statements

Note: Inline snippets reduce typing and formatting time and promote consistency in naming conventions and wording across content.

Phase 7: Maintenance

Rerun the count_snippet_percentage.py file

1. Follow the steps in Phase 3: Run the script to calculate the percentage of reuse **.**
2. Compare the original reuse score with the new one.
3. Add an additional column to your table.



Document name	Original reuse percentage	New reuse percentage	Priority level	Date	Document type	Original overall reuse percentage	New overall reuse percentage
AE IPV Document Verification SOP	82.64%		Low	12/03/2025	SOP	92.08%	94.04%
Amigos First Time Sign in SOP	8.65%	13.50%	High				
CN IPV Document Verification SOP	91.88%		Low				
CN IPV Modus Operandi (MO) SOP	70.57%		Low				
CN Manual IPV SOP	8.77%	11.21%	High				
CN identity Verification by Video Screening SOP	94.94%		Low				

Update your functions configuration document

1. Add the table to your functions configuration document.
2. Complete the reuse audit bi-annually.