HTML Scraper Prompt

I need you to write a Chrome Extension (Manifest V3) that scrapes a website based on its sitemap, downloads its media, and packages everything into a Confluence-ready HTML ZIP file.

Because of Manifest V3 limitations, you MUST use the Offscreen API architecture. Do not attempt to use DOMParser, XMLHttpRequest, or the JSZip library inside the background Service Worker.

Here are the exact requirements and architecture:

Architecture

- manifest.json: Needs offscreen, downloads, and host permissions for <all_urls>.
- background.js: Acts purely as the orchestrator. It fetches the sitemap text, manages the fetch loop (with a 300ms delay between pages to avoid rate limits), tracks duplicate titles, and updates the UI.
- offscreen.html **&** offscreen.js: This hidden document will load the jszip.min.js library natively. It is responsible for parsing XML/HTML via DOMParser, fetching media blobs over the network, building the JSZip instance, and generating the final Blob URL for download.
- popup.html **&** popup.js: A UI that drops down from the extension icon. It needs a "Start" button, a progress bar, status text, and a counter ("Processing X out of Y pages"). It must resume its visual state if closed and reopened during a scrape.

Sitemap Logic

- Fetch /sitemap.xml at the root domain.
- Pass the XML text to the offscreen document. If the offscreen document detects a <sitemapindex>, the background script must recursively fetch all nested sitemaps to build a single, flat, alphabetically sorted array of all URL strings.

HTML Extraction & Formatting

- Fetch each HTML page and pass the text to the offscreen document.
- Extract the <title> and the <main> tag content (fallback to <body> if <main> is missing).
- Wrap the extracted content in a standard HTML skeleton (<!DOCTYPE html><html><head><title>...</title></head><body><h1>Title</h1>[Content]</body></html>) so Confluence accepts it.

Media Handling

- While parsing the HTML in the offscreen document, find all <img>, <video>, <audio>, and <source> tags.
- Ignore data: URI base64 strings.
- Resolve relative URLs to absolute URLs.
- Fetch the media files natively inside the offscreen document, save them to a media/ folder inside the JSZip instance with a unique filename, and rewrite the src attributes in the HTML to point to media/filename.ext.

Duplicate Title Handling

- Confluence requires unique page titles. The background script must track titles.
- If a duplicate is found, prepend an incrementing number (e.g., "2 - Overview").
- At the end of the scrape, if duplicates were found, generate an artificial HTML page called "Duplicate Titles Report" listing the renamed pages and their original source URLs, and add it to the ZIP.

UI Updates

- The background script should update the popup's progress bar.
- The background script should also use chrome.action.setBadgeText to show the percentage complete (e.g., "45%"), or "RUN", "DONE", and "ERR" on the browser toolbar icon.

Please provide the complete, functional code for manifest.json, background.js, offscreen.html, offscreen.js, popup.html, and popup.js. Assume I will download jszip.min.js myself and place it in the folder.