USDS Coding Task- Project Report

Author: Jackson A. Henry

Date: 5/11/2025

Assignment Title: eCFR Dashboard Parsing and Visualization

Duration to Complete: Approximately 4 hours

**Overview**

This coding assignment involved building a dashboard to visualize and analyze parsed data from a set of HTML and XML documents representing titles of the Electronic Code of Federal Regulations (eCFR). The project involved writing several Python scripts to extract content and complete summary metrics. Once that was done, those metrics were displayed in a user-friendly dashboard.

The final project consisted of:

* **download\_ecfr.py** script which downloaded all of the files from the eCFR site.
* **parse\_ecfr.py** script which parsed the files.
* **create\_summary.py** script which created the summary statistics including
* title
* agency
* wordcount (number of words)
* checksum
* **index.html-** This is the Main UI template in the project.
* **app.py-** Backend file serving the ‘index.html’ file. This file handles API endpoints.

**Assignment Feedback**

This assignment was very thoughtfully designed and offered a good balance between backend development and frontend display. It was great to demonstrate:

* Data processing using these Python frameworks
* BeautifulSoup- Used for parsing
* Hashlib
* JSON
* Organized file and directory handling
* Web application development using Flask and HTML (would have used CSS and JavaScript if I were to take this project further).

While the scope of the project was manageable, I had to carefully debug filename parsing logic, which was a strong test of attention to detail (very important in software development) and writing robust code.

Although I have a strong background in data analytics and automation, I had never used Flask before. I have used Python to code, but have never used it in this sense.

In addition, I took a Software Development bootcamp at my local university, but it had been several years since I took this camp.

Furthermore, I had started working through the freeCodeCamp curriculum but had not gotten very far (this project inspired me to work on it again).

This project inspired me to want to continue learning more about software development. FreeCodeCamp is a great resource to do that, and they have a series of 12 courses (which are free), which go through skills like HTML, CSS, JavaScript, jQuery, etc.

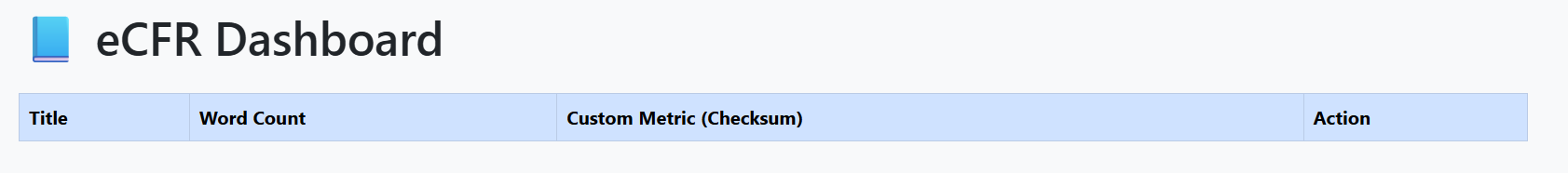
**Challenges in the Assignment** (am hoping to fix before the interview)

**Note:** While I completed most of the project, there was a final issue which was getting the metric dashboard to correctly display parsed rows.

I have documented the file structure and parsing logic, and I plan to walk through the debugging and design decisions during the interview.

I used ChatGPT to assist with understanding libraries and for help with troubleshooting specific areas in the code, similar to how someone would use StackOverflow of documentation. The goal was to learn and apply the concepts independently. **I will explain all aspects of the code in detail during the interview.**

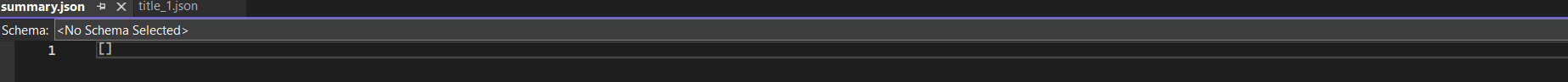
**eCFR Dashboard is blank**

****

This dashboard is needing to have values. The HTML code is working like it should, but none of the rows in the backend were displaying values.

To fix this, I had checked through the parse\_data.py file to ensure that the files were being properly parsed. I made a couple of adjustments to this file, and am happy to note that this file is correctly extracting the HTML text, calculating word counts, and saving the parsed data as a JSON file.

**Summary file is also blank**

****

After doing further investigation, I realized that the summary.json file is empty.

All of the files were properly being generated in the parsed folder (title\_1 is an example).

A screenshot of a computer

AI-generated content may be incorrect.

To fix this, I am going to write another file that will read in all the .json files in the parsed folder. This file will aggregate them into a single list of dictionaries, and will write that list into summary.json. The dashboard on the front-end will read it.

**Further Improvements I would make on the project**

* **CSS Implementation-** Although HTML gives the basic “skeleton” of the dashboard, CSS will make the dashboard look a little bit nicer.
* **JavaScript-** Would use JavaScript for additional functionality.

In addition, I would explore other metrics that could make decision-making more effective.