Assessment Submission: eCFR Deregulation Dashboard

Developer: Mohammad Khan

Date: February 27th, 2026

1. Feedback on the Assignment & Expertise Fit

I really enjoyed this assessment. Developing a small full stack website that takes data from an eCFR website with API, store it into our local postgreSQL database and finally creating a frontend for users to digest that information.

My background in full-stack development allowed me to approach this methodically. Rather than building a frontend-only app that would crash the browser with massive fetch requests, I implemented a robust Node.js/PostgreSQL backend ingestion engine. The tech stack I went with was the following: PostgreSQL, Express JS, React, Node JS (PERN).

A few notes:

- The eCFR API does not publish data on weekends or federal holidays. I had to put in a logic to make sure we fetch on workdays to avoid crashing the script. Finally, to stay strictly under the 1,200-line code limit, I used simple React and express js on top of Node JS.

2. Duration

This assessment took me about **6-7 hours** to complete.

3. Application Access & Setup Instructions

Because this application processes real data and requires a backend database, it is intended to be run locally as I did. Please follow these steps to initialize the application on your machine:

Prerequisites: Ensure you have PostgreSQL and Node.js installed.

Step 1: Database Configuration

1. Open the `ecfr-backend` folder.
2. Inside the `.env file`, update the `DB_PASSWORD` to match your local PostgreSQL password.

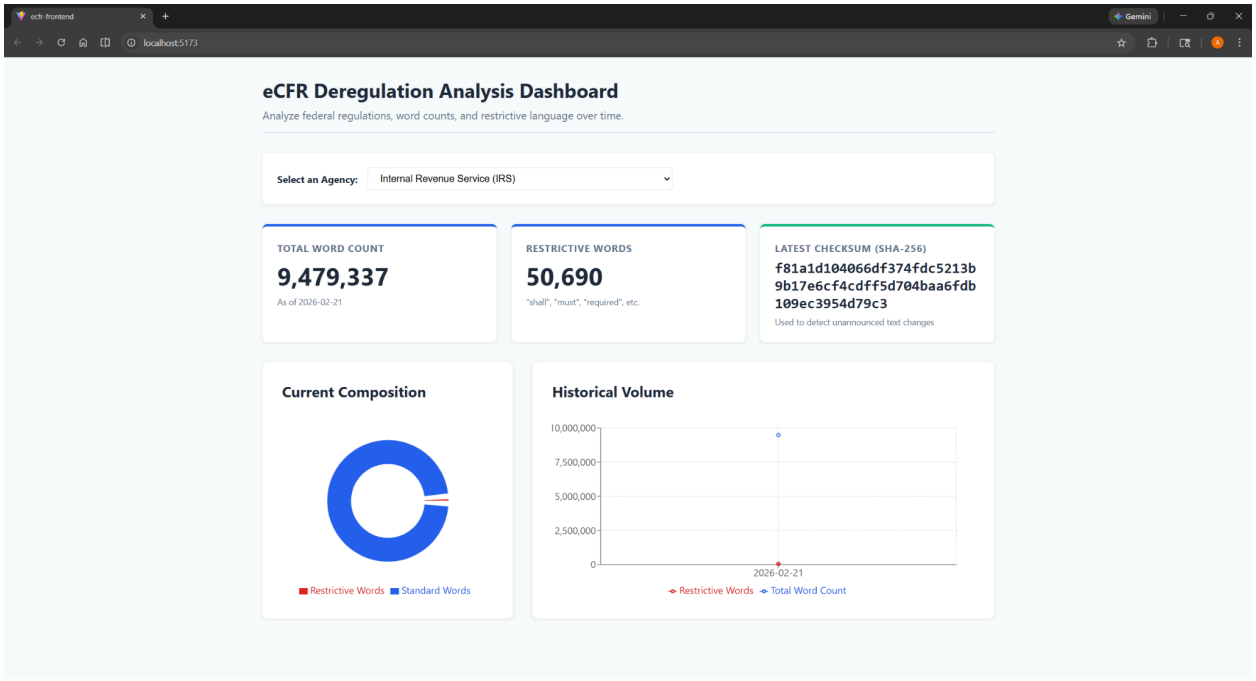
Step 2: Backend & Database Initialization

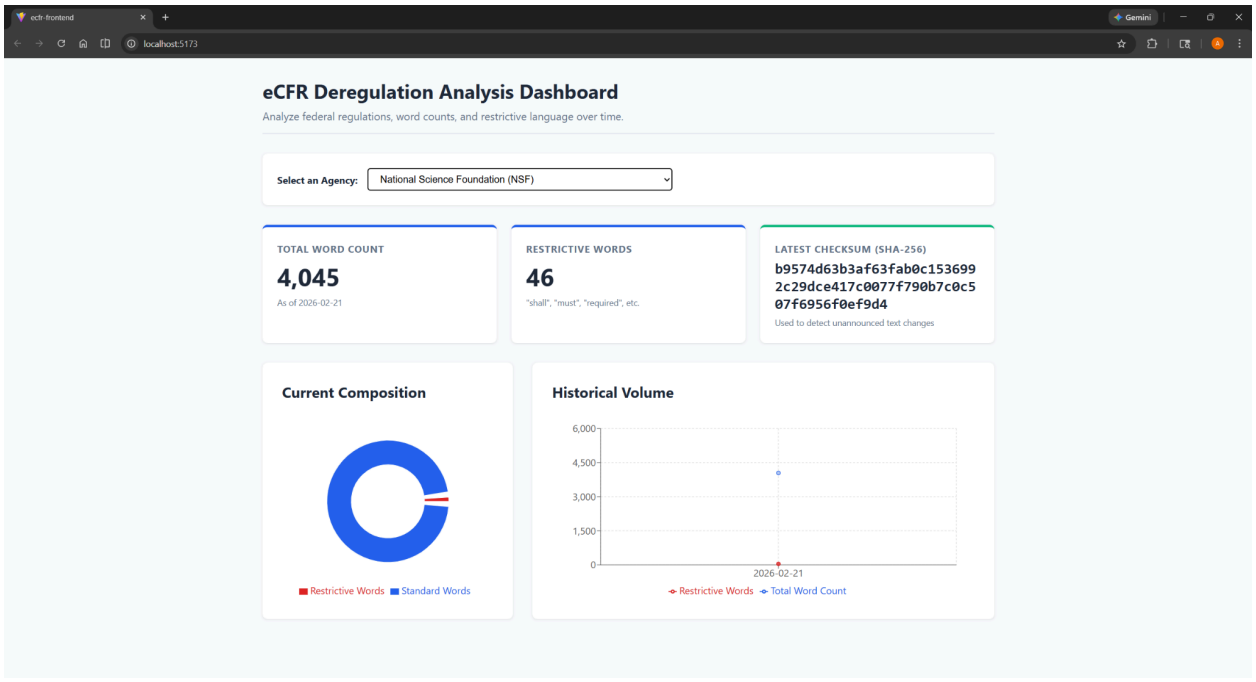
1. Open a terminal and navigate to `ecfr-backend`.
2. Run `npm install`.
3. Run `node setup.js`. *This setup js creates ecfr_db database inside postgres and creates tables that we later need for frontend fetching*
4. Run `node server.js` to start the backend API

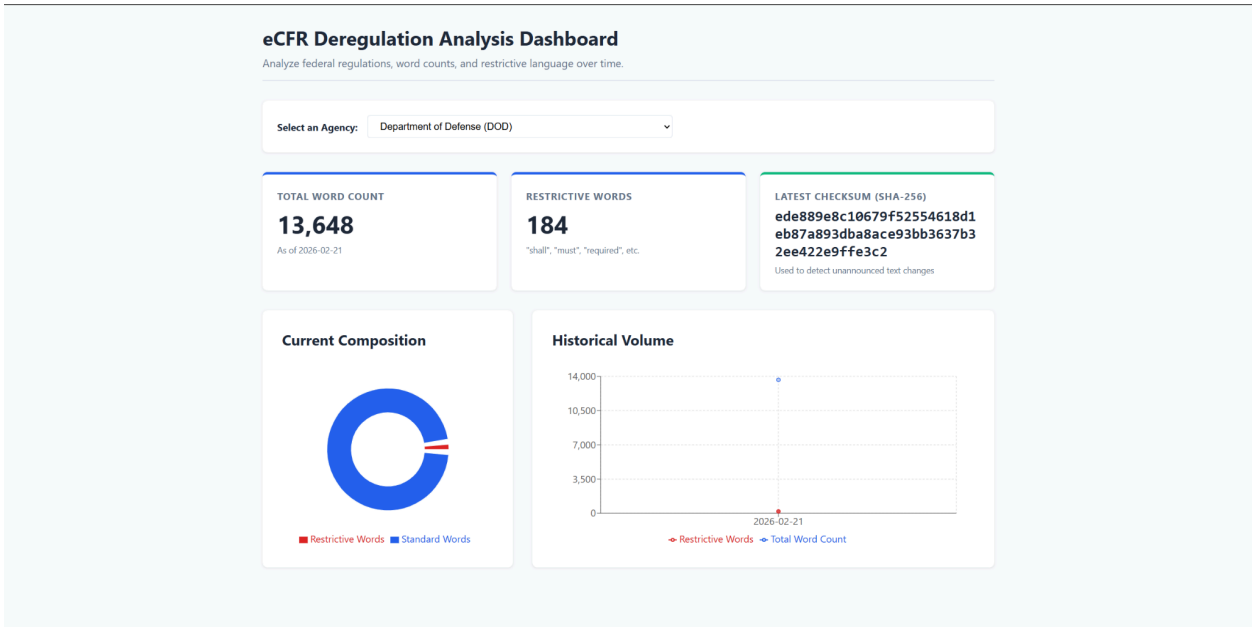
Step 3: Frontend Initialization

1. Open a second terminal window and navigate to `ecfr-frontend`.
2. Run `npm install`.
3. Run `npm run dev`.
4. Open your browser and navigate to the local link provided (usually `http://localhost:5173`).

4. Application Screenshots




ecfr-frontend
localhost:5173
Gemini
eCFR Deregulation Analysis Dashboard
Analyze federal regulations, word counts, and restrictive language over time.
Select an Agency:
National Science Foundation (NSF)
TOTAL WORD COUNT
4,045
As of 2026-02-21
RESTRICTIVE WORDS
46
"shall", "must", "required", etc.
LATEST CHECKSUM (SHA-256)
b9574d63b3af63fab0c1536992c29dce417c0077f790b7c0c507f6956f0ef9d4
Used to detect unannounced text changes
Current Composition
Restrictive Words
Standard Words
Historical Volume
6,000
4,500
3,000
1,500
0
2026-02-21
Restrictive Words
Total Word Count


eCFR Deregulation Analysis Dashboard
Analyze federal regulations, word counts, and restrictive language over time.
Select an Agency:
Department of Defense (DOD)
TOTAL WORD COUNT
13,648
As of 2026-02-21
RESTRICTIVE WORDS
184
"shall", "must", "required", etc.
LATEST CHECKSUM (SHA-256)
ede889e8c10679f52554618d1eb87a893dba8ace93bb3637b32ee422e9ffe3c2
Used to detect unannounced text changes
Current Composition
Restrictive Words
Standard Words
Historical Volume
14,000
10,500
7,000
3,500
0
2026-02-21
Restrictive Words
Total Word Count