Overview

Tuesday, October 28, 2025 4:00 PM

This project is meant to make the information on politicians more accessible from the various sources on the internet. Users should be able to view the top donors, PACs, external sources of income, etc., and compare those to their apparent policies. I will also include OpenAI/LLM chatbots to further aid the user in analyzing the Politicians' public info.

This is a non-partisan tool, and it does not represent any political party or ideology. It is meant to be used for people to educate and inform themselves.

Tech Stack

- **Backend:** FastAPI
    - Why?
        - **Modern & Fast:** It's a modern Python framework that is incredibly fast.
        - **Built-in Data Validation:** Uses Python type hints (Pydantic) to validate incoming data, which is perfect for an API.
        - **Automatic Docs:** Automatically generates interactive API documentation (Swagger UI)
- **Database:** PostgreSQL
    - Why?
        - Perfect for structured data (politicians, donors, votes, statements).
        - It's a standard in the industry and great for practicing complex SQL queries.
- **Frontend:** React
    - Why?
        - **Interactive & Popular:** You need a JavaScript framework to build the "interactive" part.
        - React is the most popular in the job market
- **Data Visualization:** Plotly.js
    - Why?
        - **Visuals:** Fetch data from your API and feed it into a JS-based charting library on the frontend.
        - Plotly is more powerful for complex, scientific, and interactive charts.
- **AI / Data:** Pandas, OpenAI, LangChain
    - Why?
        - **The Engine:** These are the necessary Python libraries.
        - **Pandas** to clean data
        - **OpenAI** library for summaries
        - **LangChain** to manage the prompts and context for your chatbot
- **DB Connector:** SQLAlchemy (Python)
    - Why?
        - **The "Glue":** This is the Python library used to connect the FastAPI backend to the PostgreSQL database.
        - ORM (Object-Relational Mapper)
            - the standard way to handle databases in an application

Workflow

1. **Data Ingestion**
    a. Python scripts (using `Pandas`, `Requests`, etc.) will run on a schedule (or manually) to fetch data from sources
2. **Store & Transform**
    a. Scripts will clean and transform data to store neatly in PostgreSQL database
3. **Backend API**
    a. **FastAPI** application waits for requests. When a user on your website wants to see a politician's donors, the **React** frontend sends a request
4. **Backend Logic**
    a. **Backend Logic:** FastAPI receives this, uses **SQLAlchemy** to query PostgreSQL database
    b. Sends it to the **OpenAI** API for a quick summary
    c. Sends all of it back to the frontend as a JSON object.
5. **Frontend UI**
    a. **React** app receives the JSON data and uses **Plotly.js**
        i. beautiful, interactive charts and summaries

Data Sources

- **Campaign Finance (Donors, PACs)**
    - This data comes from one primary source, but there are two main ways to get it.
        - FEC (Federal Election Commission)
            - **What it is:** The official U.S. government agency that registers all campaign finance data for federal elections (President, Senate, House).
                - **Pros:** the **primary, unadulterated, 100% non-partisan raw data**
                    - **OpenFEC API**
- **Voting Records & Legislation**
    - This is about what politicians *do* in office—what they sponsor, and how they vote.
        - Congress.gov
            - **What it is:** The official website for U.S. federal legislative information, run by the Library of Congress.
                - **Pros:** It's the official record. It has all bills, all amendments, all co-sponsors, and every single "roll call" (recorded) vote from every member. (Raw Data)
                    - API: https://gpo.congress.gov/
        - Congress Project
            - **What it is:** A community-run project to develop Python tools to collect data about the bills, amendments, roll call votes, and other core data about the U.S. Congress into simple-to-use structured data files.
                - **Pros:** It's the official record. It has all bills, all amendments, all co-sponsors, and every single "roll call" (recorded) vote from every member. (Raw Data)
                    - Link: https://github.com/unitedstates/congress
- **External Income & Financial Disclosures**
    - This is what politicians own, where they get money from *outside* of their government salary (stocks, book deals, etc.).
        - ProPublica Data Store
            - **What it is:** ProPublica is a non-profit, non-partisan investigative journalism organization
            - **Pros:** They have datasets on personal financial disclosures that they have painstakingly digitized from PDFs.
                - They also maintain datasets on things like political appointees and other key data.
            - **Cons:** These are often static files (CSVs) you must download and import, not a live API
- **Past Statements & Public Behavior**
    - **Most tedious** part of the project because the data is unstructured
        - ProPublica's "Politwoops" Database
            - **What it is:** A project that archives tweets deleted by public officials.
            - **Pros:** objective record of public statements that were later removed.
            - **Cons:** It's limited to Twitter and may not be comprehensive.
        - Other Possible Sources:
            - Factcheck.org
            - PolitiFact