Project Spec

LexLearner Project Spec

Note: Gemini used to assist with gaps in technical implementation knowledge

1. Introduction

- **Project Overview**: LexLearner is a website that enables users to track and learn about legislation in the U.S. at both the local and national levels.
- **Goals**:
  - Increase public access and education.
  - Create a source of truth for difficult-to-track U.S. legislation.
  - Make fact-checking online quick and easy for news and social media users.
- **Target Audience**:
  - Concerned citizens
  - Educators
  - Students
  - Lawyers/Policy advocates
- **Problem Statement**: Legislative information, including bills proposed, their supporters, their implications, etc, is often enshrouded in uncertainty and technical legal language. Further, social media is frequently referenced as a source of news, which does not always portray the process and progress of law-making accurately.
- **Solution Overview**: LexLearner will provide a source of truth for U.S. legislation, empowering everyday people to follow and understand the process, the content, and the potential impacts of new and emerging legislation.
- **External Mechanisms**: The project will primarily use external APIs for data acquisition. Major packages will likely include Python's `requests` module for API

calls and potentially `pandas` for data handling.

- **Minimal CLI Functionality (V 1)**: For initial version, LexLearner could minimally run as a command-line interface. Users could type commands to:
  - `search_bill <keyword>`: Return a list of bill titles and their current status.
  - `get_bill_details <bill_id>`: Display a simplified summary, sponsors, and current status for a specific bill.
  - `get_latest_executive_orders`: List the most recent executive orders by title and date.
    This would involve using Python scripts to fetch, process, and display data directly in the console.
- **Ideal GUI**: Ideally, LexLearner will use a web-based GUI. The frontend is envisioned to be built with React.

2. Functional Requirements

- **Bill & Executive Order Search & Browse**:
  - Keyword Search
  - Advanced Filters
  - Categorization/Topics
- **Bill Detail Pages**:
  - Bill Title & Number
  - Sponsors/Co-sponsors
  - Opponents (if available)
  - Status & Timeline
  - Simple Summary
  - Full Bill Text or link to text
  - Current Status & Legislative History
  - Key Dates
  - Voting Records (if voted on)

    - Related Legislation (by topic)
    - Estimated Impact:
        - Groups
        - If criminal, punishment
        - Other considerations
- **Executive Order Details**:
    - Simple Summary
    - Full Text or link to text
    - Related Legislation (by topic)
    - Estimated Impact:
        - Groups
        - If criminal, punishment
        - Other considerations
- **Local Resources (if APIs available)**:
    - Upcoming Vote Information
    - Lawmakers:
        - Contact Information
        - History
        - Party Involvement
- **Educational Resources**:
- **Data Acquisition & Update**:
    - Automated API Integration
    - Error Handling

3. Non-Functional Requirements

- **Performance**:

    - Fast Load Times (Less than 5s/pg)
    - Responsive Design (mobile, tablet, app)
- **Security**:
    - API Key Management
- **Usability**:
    - Intuitive User Interface
    - Clear Language
- **Accessibility**:
    - WCAG compliance
    - Nice to have: screen reader and dark mode enabled
- **Scalability**:
    - Data Handling
    - User Base (stress testing)
- **Maintainability**:
    - Modular Codebase
    - API Monitoring

4. Tech Architecture (High-Level)

- **Frontend**: React
- **Backend**: Python
- **Database**: PostgreSQL or MongoDB
- **Deployment**: PythonAnywhere

5. Data Model (High-Level)

- **Bill**:
    - `bill_id` (Primary Key)
    - `title`

    - `bill_number`
    - `summary`
    - `official_text_url`
    - `legislative_body` (Federal, State, Local)
    - `status`
    - `introduction_date`
    - `last_action_date`
    - `sponsors`
    - `topics`
    - `simplified_explanation` (Text)
- **Sponsor**:
    - `sponsor_id` (Primary Key)
    - `name`
    - `party`
    - `chamber` (House/Senate/State Legislature)
    - `contact_info` (Link to official page)
- **Vote**:
    - `vote_id` (Primary Key)
    - `bill_id` (Foreign Key)
    - `date`
    - `chamber`
    - `result`
    - `yes_votes` (Count)
    - `no_votes` (Count)
    - `abstain_votes` (Count)
    - `topics`

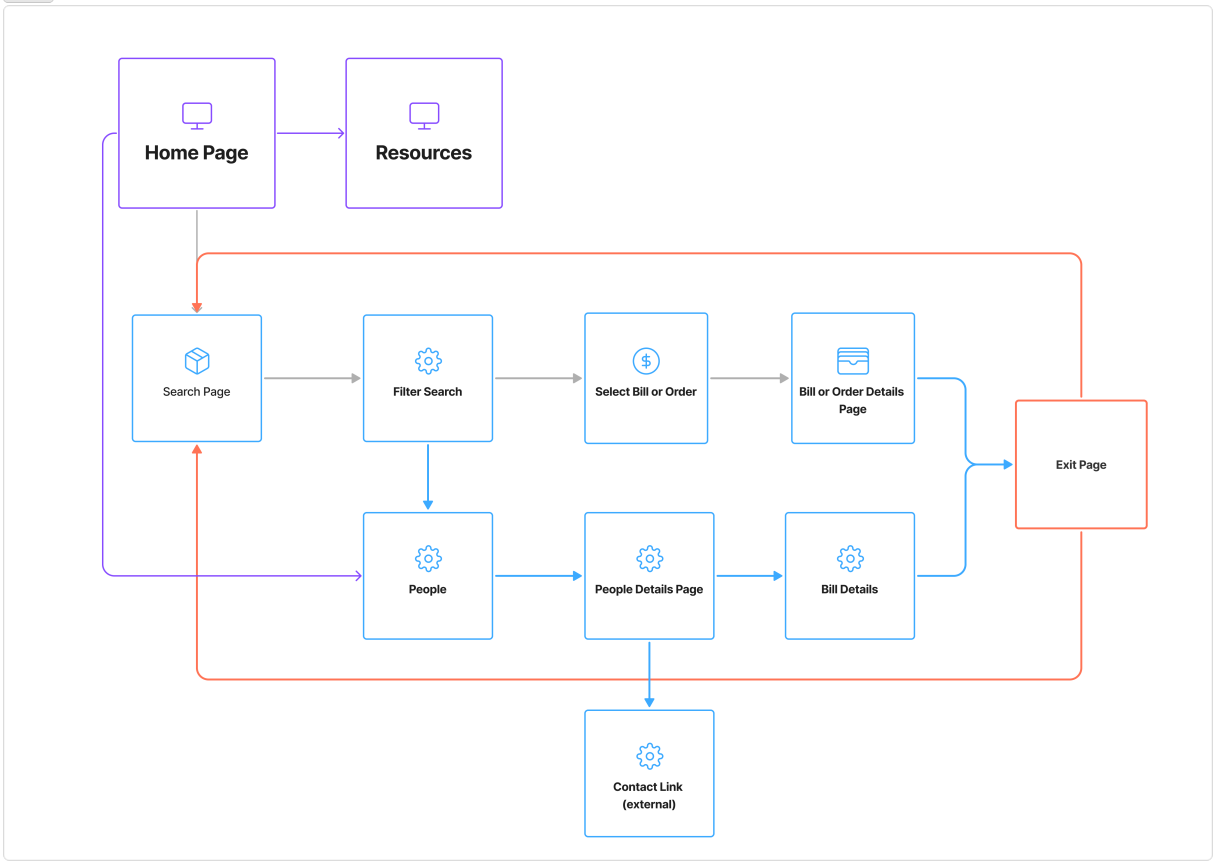
- **Executive Order**:
  - `eo_id` (Primary Key)
  - `title`
  - `summary`
  - `official_text_url`
  - `issuing_authority`
  - `effective_date`
  - `topics`

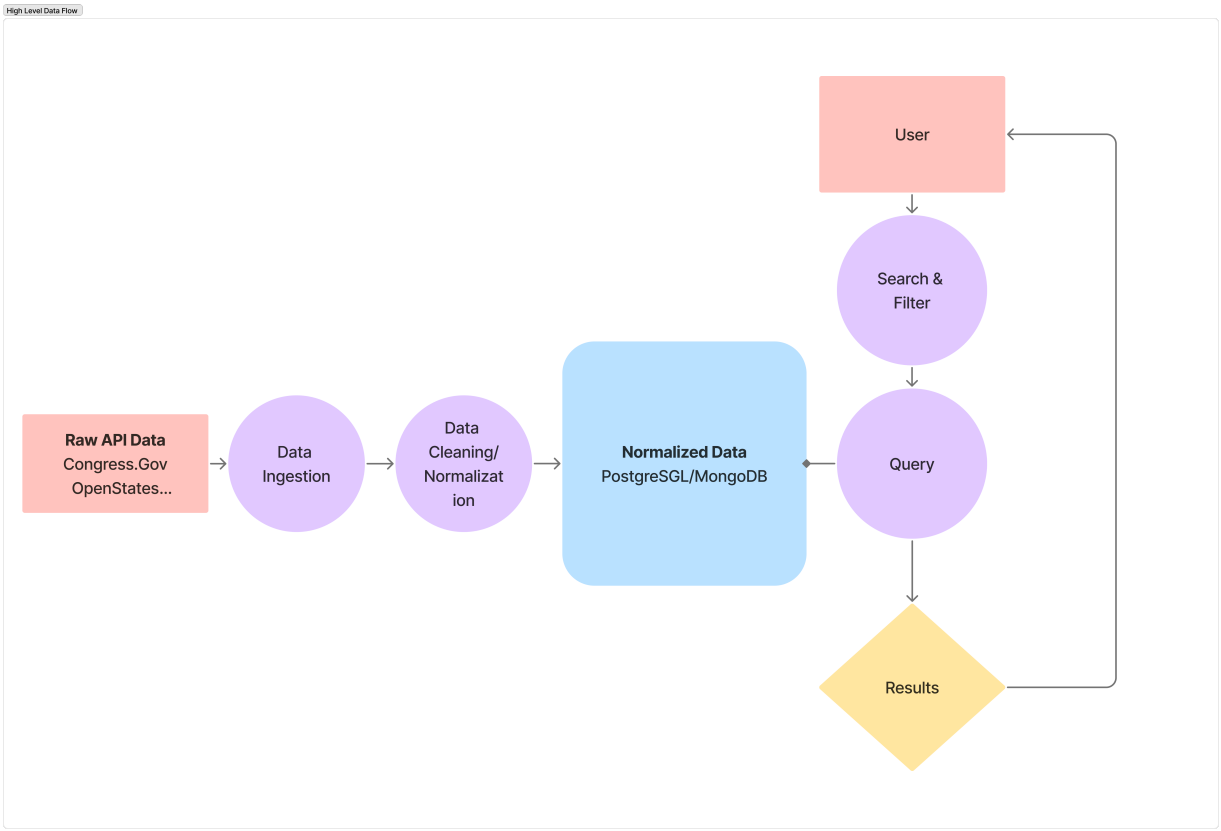
6. Free APIs

- **Federal (Primary Source)**:
  - Congress.gov API
  - api.govinfo.gov/docs/
  - GPO (Government Publishing Office) API
- **State**:
  - OpenStates API
- **Additional Free**:
  - Sunlight Labs Congress API
  - ProPublica Congress API
  - Google Civic Information API

7. Figma Flows/Diagrams/Visuals

User Flow


Home Page
Resources
Search Page
Filter Search
Select Bill or Order
Bill or Order Details Page
Exit Page
People
People Details Page
Bill Details
Contact Link (external)



8. Task Vignettes (User activity "flow")

As the user (a concerned citizen), I want to track a bill from its introduction to its final status and understand its potential impact.

Vignette 1: Tracking a Bill

As a concerned citizen, I want to find out the current status of the "Clean Air Act Amendment of 2025" and who supported it. I will navigate to the LexLearner homepage and use the search bar. After typing in "Clean Air Act Amendment 2025" and hitting enter, I expect to see a search results page. From this list, I'll click on the specific bill I'm looking for. On the bill's detail page, I should immediately see its current legislative status ("Passed House, Awaiting Senate Vote") and a list of its primary sponsors.

- **Technical Details**:
  - User input: String for search query.
    - Filters

    - Backend processing: Keyword search against bill titles, summaries, and perhaps bill numbers.
    - API call: Likely to Congress.gov or ProPublica Congress API.
    - Data presented: Bill title, number, summary, current status, and sponsor names.
    - Database interaction: Query the `Bill` and `Sponsor` tables in the `Normalized Data` store.

Vignette 2: Understanding a Recently Passed Executive Order

As a student researching government actions, I heard about a recent executive order on climate change and want to understand what it actually does. I will go to the LexLearner homepage and find a section for "Recently Issued Executive Orders." Clicking on the latest executive order related to climate, I expect to be taken to its detail page. Here, I hope to find a simplified summary of the order's content and its effective date, so I don't have to wade through dense legal text.

- **Technical Details**:
    - User navigation: Clicking on a pre-defined page for executive orders.
    - API call: Likely to api.govinfo.gov/docs/ for executive order details.
    - Data presented: Executive order title, issuing authority, effective date, and a simplified summary.
    - Database interaction: Retrieve data from the `Executive Order` table in the `Normalized Data` store.

Vignette 3: Finding and Contacting a Local Representative

As an engaged citizen, I want to know who my state senator is and how to contact them about a specific state bill. I will navigate to the "Local Resources" or "Lawmakers" section of LexLearner. I will search by state. The system should then display my elected officials at the federal and state levels. Clicking on my state senator's name, I expect to see their contact information (e.g., office address, phone number, website link) and their voting history on key bills. From there, I will click on the external link to their contact form.

- **Technical Details**:

- User input: State search.
- API call: Google Civic Information API for elected officials based on address.
- Data presented: Legislator name, party, chamber, and contact details.
- Database interaction: Possibly store cached lawmaker data, otherwise direct API lookup for specific contact info.
- External navigation: User is redirected to an external website.

9. Future Enhancements (Out of Scope)

- User Alerts/Notifications based on followed bills and laws.
- Machine Learning for complex aggregated analysis.
- Integration with Advocacy Tools or local resource sites.
- Report building and export.
- Data prior to 2010 (or initial year starting current site).
- User profiles.

10. Development Phases (High-Level)

- **Phase 1: Research & Planning**:
  - Python refresh
  - Project Spec
- **Phase 2: Core Development**:
  - Data flow implementation
  - Coding core features (API ingestion, data cleaning, basic search, bill/EO detail display).
- **Phase 3: Enhanced Features & UX**:
  - Enhance UX/UI experience (implementing responsive design, improving visual appeal).
  - Implement advanced filtering and related legislation features.

- **Phase 4: Testing & Deployment**:
  - Ensure responsiveness.
  - Functionality testing.
  - WCAG standards compliance.
  - Implement feedback.
- **Phase 5: Iteration & Maintenance**:
  - Ongoing API monitoring and data updates.
  - Bug fixes and minor feature improvements.
  - Future enhancements (local data integration beyond current APIs).

Self Evaluation

Unexpected Change: how much I needed to limit my initial idea to make the project technically feasible. Removed user profile and login.

Confidence in Implementation: I feel a little hesitant due to my humanities background, but with the help of modern online resources and the help of the professor, I feel confident I can create a great project.

Biggest Problem to Solve: Data accuracy and API calls. Since data is the core of the functionality, it must be accurate and up to date at all times, and ensure error handling for downtime and other challenges.

Biggest Challenge: technical implementation. While I understand conceptually what I need to do, actually implementing it seems daunting. Feedback will be essential to ensure I'm on the right track. Additionally, where should I host the data and deployment for free?