Practical Exercise: Web Translator using Frontend + Backend

Objective

Learners will practice prompt engineering with an LLM to develop a web-based translator application. The app should include both frontend and backend components, support language selection, detect source language, translate text, display results, and optionally provide read-aloud functionality.

Instructions for Learners

Part 1: Program Requirements

- Input: User enters text in the frontend.
- Language Selection: User selects a target language.
- Backend: Receives the text and target language, detects the source language, translates the text, and returns source text, detected language, and translated text.
- Frontend: Displays the source and translated text, includes a read-aloud option.

Part 2: Backend Instructions

- Task: Write an LLM prompt to generate a Python Flask backend server.
- Requirements:
  - Use the googletrans library for translation.
  - Ensure the correct version of googletrans is used to avoid errors:
    pip uninstall googletrans
    pip install googletrans==4.0.0-rc1
  - Handle POST requests for translation.
  - Return JSON containing source text, detected source language, translated text, and target language.

Part 3: Frontend Instructions

- Task: Write an LLM prompt to generate HTML + JavaScript frontend.
- Requirements:
  - Text input box for user entry.
  - Dropdown to select target language.
  - Button to send text to backend and receive translation.
  - Display source and translated text.
  - Include a read-aloud button to speak the translated text.

Part 4: Testing and Integration

- Start the backend Flask server.
- Open the frontend in a browser.
- Enter text, select a target language, click translate.
- Verify source and translated text display correctly.
- Test the read-aloud feature.

Reflection Questions

- How does the frontend communicate with the backend?
- How does the backend detect the source language?
- What improvements can be made to the UI/UX or functionality?