Take-Home Task Brief: AI-Powered Speaker Notes Spell Checker

Overview

Design and build a proof-of-concept spell checker for speaker notes in a presentation application. This task evaluates your ability to integrate AI thoughtfully into a product workflow while considering both user experience and technical implementation.

Context

Presenters often write speaker notes quickly while preparing their decks. These notes need to be accurate and professional, but traditional spell checkers miss context-specific terminology (brand names, technical terms, industry jargon) that might be correct within a presentation's context.

Your Task

Build a working prototype that demonstrates how you would approach an AI-enhanced spell checking system for speaker notes.

Core Requirements

1. Functional Prototype

- Create a simple interface where users can input/edit speaker notes
- Implement spell checking that flags potential errors
- Provide correction suggestions
- Show how you'd handle context-aware checking (considering slide content, deck terminology, etc.)

2. Technical Implementation

- Use any language/framework you're comfortable with (bonus points for functional programming approaches)
- Integrate with at least one AI API (OpenAI, Claude, or similar)
- Handle API calls efficiently (consider rate limits, costs, latency)
- Include basic error handling

3. Product Thinking

- How does your solution improve on traditional spell checkers?
- When should AI be invoked vs. using traditional methods?
- What's the user experience for accepting/rejecting suggestions?

What to Deliver

1. **Working Code** (2-4 hours recommended)
    - A functional prototype (doesn't need to be production-ready)
    - README with setup instructions
    - Brief architecture notes
2. **Written Component** (~1 page)
    - Your approach and key design decisions
    - Trade-offs you considered (accuracy vs. speed, cost vs. quality, etc.)
    - How you'd extend this for production (scalability, privacy, offline support)
    - 2-3 ideas for future AI enhancements to speaker notes
3. **Evaluation Strategy**
    - How would you test/evaluate this feature's effectiveness?
    - What metrics matter?

Evaluation Criteria

- **AI Integration**: Thoughtful use of AI where it adds value
- **Product Sense**: Understanding when/how AI enhances the user workflow

- **Code Quality**: Clean, understandable implementation
- **Pragmatism**: Practical solutions over over-engineering
- **Communication**: Clear explanation of decisions and trade-offs

Constraints

- Time: Aim for 2-4 hours
- Scope: This is a proof-of-concept, not production code
- No need for: Authentication, database, full UI polish
- Focus on: The AI/ML integration and product thinking

Bonus Points

- Consideration of presentation-specific context (industry terms, brand names)
- Interesting approaches to prompt engineering
- Thoughts on privacy/data handling
- Creative ideas for extending the concept

Submission

Please submit:

- GitHub repository or zip file with code
- README with setup instructions and your written responses
- Any additional documentation you think is relevant

Questions? Feel free to reach out - we're happy to clarify scope or requirements.

Note: We're more interested in your thinking process and how you approach AI product problems than in pixel-perfect execution. Show us how you work.