FlashGenius — Complete Testing Scripts (Paragraph Format)

---

1. Prompt Engineering Testing

We tested FlashGenius' ability to generate flashcards based on user topics and selected difficulty levels.
When users entered a simple topic like "Gravity," the system generated 3–5 basic flashcards containing clear, introductory information.
When users selected "Intermediate" or "Advanced" difficulty levels, the flashcards became more detailed and technical, adjusting content complexity appropriately.
If users entered no topic or an invalid difficulty level, the system displayed a friendly error:
"Please enter a valid topic and select a supported difficulty level (Basic, Intermediate, or Advanced)."
In case the OpenAI API returned improperly formatted responses, the app's fallback logic detected the issue and still produced usable flashcards using flexible parsing.

Example Output for Intermediate Difficulty:

```
Flashcard 1:
Q: What is gravity?
A: Gravity is the force that attracts two bodies toward each other,
depending on their masses and the distance between them.

Flashcard 2:
Q: Who formulated the law of universal gravitation?
A: Sir Isaac Newton proposed that every particle attracts every other
particle with a force proportional to their masses and inversely
proportional to the square of the distance between them.
```

---

2. Retrieval-Augmented Generation (RAG) Testing — Text Files Only

We tested the system's RAG feature by uploading `.txt` files as user knowledge sources.
When valid text files like "PhysicsNotes.txt" were uploaded, the system parsed and embedded them client-side successfully.
Flashcard generation based on document content worked reliably, with citations such as *(Source: PhysicsNotes.txt)*automatically appended.
If a user attempted to upload unsupported file types such as PDFs, DOCX, or images, the system rejected the upload and displayed an error:
"Only .txt files are supported. Please upload a valid text file."
Empty or corrupted `.txt` files triggered another error: *"The uploaded file is empty or could not be processed."*
In cases where no matching content was found within the uploaded documents, the system either generated fallback flashcards from general knowledge or politely indicated that no reliable source material was available.

Example Output After Uploading "PhysicsNotes.txt":

```
Flashcard:
Q: What is Newton's Third Law of Motion?
A: For every action, there is an equal and opposite reaction. (Source:
PhysicsNotes.txt)
```

Additionally, if users uploaded multiple text files, the knowledge library displayed all files separately, and flashcards referenced the correct document. Duplicate file uploads were handled gracefully by either overwriting or informing users that the file already existed.

---

3. Multimodal Integration Testing (Speech-to-Text and Text-to-Speech)

We validated FlashGenius' multimodal capabilities through speech input and audio playback features.
Users could speak any topic clearly into their microphone (e.g., "Explain Quantum Computing"), and the system successfully recognized the speech, converted it into text, and generated flashcards accordingly.
When users denied microphone access or spoke unclearly, the system detected the issue and displayed appropriate messages like:
"Microphone access is needed for voice input," or
"Sorry, we couldn't understand that. Please try again."

For audio playback, users could click a play button next to any flashcard, and the content was read aloud using ElevenLabs text-to-speech (TTS) with natural-sounding voices.
If ElevenLabs API was unavailable, the Web Speech API fallback ensured that users could still hear the flashcard content.
Even during slow network conditions, the system showed a loading spinner to indicate TTS processing, ensuring users weren't confused by delays.

Example TTS Output for a Flashcard: *(spoken aloud)*

> "Flashcard 1: What is Quantum Computing? It is a type of computing where information is processed using quantum bits instead of classical bits."

---

4. Synthetic Data Generation Testing

We tested FlashGenius' ability to automatically generate real-world examples and alternate perspectives for each flashcard topic.
When the Synthetic Data Generation toggle was enabled, flashcards included a supplementary section called "Real-World Example" or "Application" under the answer.
For instance, after generating flashcards about "Artificial Intelligence," the system added examples like self-driving cars, recommendation systems, and medical imaging diagnostics.

If users disabled the synthetic examples feature, the flashcards were generated without any additional context.
In cases where the API call to generate examples failed (e.g., timeout, error), the system showed a graceful fallback message:
"Example unavailable. Please try regenerating later."

The system also demonstrated the ability to generate multiple domain-specific examples (e.g., healthcare, education, technology) when prompted for broader perspectives.

Example Flashcard with Synthetic Example:

```
Flashcard:
Q: What is Artificial Intelligence?
A: Artificial Intelligence is the simulation of human intelligence in
machines that are programmed to think and learn.

Real-World Example:
Self-driving cars use AI algorithms to interpret sensor data and
navigate safely without human intervention.
```

✅ Overall Testing Conclusion

FlashGenius successfully passed functional testing for all major features:

- Prompt Engineering adjusts output based on topic and difficulty levels, with strong error handling.
- RAG (text files only) processes `.txt` documents correctly, cites sources properly, and blocks unsupported formats.
- Multimodal Integration enables smooth voice input and natural TTS playback, even under slow networks.
- Synthetic Data Generation enriches flashcards with real-world examples and handles fallback conditions gracefully.

The system maintains a **responsive**, **privacy-conscious**, and **user-friendly** experience across different use cases.