**Teraleads Prompt Engineer Assessment**

**Overview**

This structure lets you retain full task integrity with deliverables suitable for a real-world assessment while showing:

Prompt design skills

Scripting proficiency

Integration and automation thinking

Awareness of voice tech nuances

**Submission Guidelines**

* **Push your code to a private GitHub repo.**
* **Share the link of the docs and make sure it’s viewable by everyone, and access to app@teraleads.com**
* **Deadline: Friday 18, 2025**
* **Check our website for any information, teraleads.com**

**Section 1: Voice Prompt Engineering**

**Do:**

Write one concise voice prompt covering name, contact, and appointment preference.

Add notations for tone and pauses (keep it minimal but meaningful).

Provide a short sample conversation (5–6 exchanges max).

Write a 200-word explanation, focused on tone, clarity, and recovery.

**Keep focus on:**

Natural phrasing

Voice UX (pace, pause, re-prompts)

Basic error handling (“Sorry, I didn’t catch that…”)

**Section 2: Python & JSON Integration**

**Do:**

Write a simple Python script:

Hardcode a small JSON string (2–3 appointments).

Parse and generate 2–3 simulated TTS messages.

Annotate with comments: “<friendly tone>”, “<pause>”, etc.

Add a short paragraph on how this could plug into TTS systems.

**Focus on:**

JSON parsing

Modularity (use a generate\_reminder() function)

Readable message output with tone guidance

**Section 3: Automation Workflow Design**

**Do:**

Create a basic flowchart using Draw.io, Whimsical, or even hand-drawn (scan it).

Trigger: Booking confirmation

Steps: Data to JSON → Create voice prompt → Send via API → Log outcome

Add a 250-word description explaining:

Each step

Integration with TTS

Fallbacks (e.g., if no answer, send SMS or retry)

**Focus on:**

Simplicity

Clear data movement

Voice-specific handling (TTS, delays, retries)

**Section 4:**

**Do:**

Use a standard RAG diagram template with:

User query → Retriever → Vector DB → Language Model → Voice Output

Write a 300–350-word explanation:

Focus on voice UX (clarity, truncation, session memory)

Simulate one voice FAQ use case (e.g., “What are dental implants?”)