

## Consideraciones de la solución.

### Justificación técnica Google Cloud TTS y STT vs WebKit Speech API

Cuando se empezó el desarrollo del sistema conversacional del chatbot BOTE3T se implementó usando las Web Speech APIs integradas en navegadores basado en Chromium. Sin embargo, al final se optó por integrar los servicios pagos de Google Cloud. A continuación, se presentan las razones técnicas y operativas que motivaron la decisión final.

#### Comparación técnica

Criterio	WebKit Speech API	Google Cloud TTS / STT
Compatibilidad	Solo en Chrome y parcialmente en Android.	Multiplataforma (Procesamiento en backend)
Calidad de reconocimiento	Aceptable, pero limitado en precisión ya que no está diseñado para captar oraciones largas	Alta precisión
Calidad de voz sintetizada	Voz robótica y poco natural	Voces naturales con control de entonación y ritmo
Configuración de idioma/voz	Opciones limitadas y no estandarizadas	Amplitud de idiomas y control total (pitch, rate)
Acceso offline	Limitado según navegador	No disponible
Privacidad / Seguridad	Ejecutado del lado del cliente	Procesamiento en backend
Control de errores y logs	Difícil de auditar o capturar errores	Integrado en backend, permite trazabilidad completa
Integración con Flask	Complejo, dependiente del navegador	Totalmente integrable como microservicio Flask
Limitaciones por navegador	Muy dependiente de políticas de Chrome	Independiente de navegador, estable y escalable

Como dificultad inicial de la implementación del WebKit se tuvo la longitud de reconocimiento, ya que por cuestiones de seguridad la grabación de audio tiene un tiempo máximo, luego de que se pasa este al primer silencio la transcripción para. Para que el usuario mantuviera abierto el micrófono la cantidad de tiempo que necesitara se tuvo que implementar un ciclo el cual volvía a lanzar la función de reconocimiento una y otra vez hasta que el usuario presionara el botón para terminar la “grabación”. Todo este sistema se manejaba a través del front-end por medio de JavaScript. Después de superar esta limitación se pudieron hacer pruebas de calidad sobre la transcripción, se encontró que la transcripción era mejor en smartphones (posiblemente porque poseen un mejor sistema de micrófonos) que en computadores. La generación de voz fue muy

simple de implementar, el único problema que podíamos observar era la calidad y, nuevamente, la variabilidad entre plataformas siendo Android la preferida.

La facilidad de implementar un servicio para ambas funcionalidades (STT y TTS) iba en línea con la naturaleza modular que se usó para el proyecto. Se consideró como primera alternativa los servicios de Google Cloud ya que conocíamos la experticia de Google al momento de generación y transcripción de voz. Después de implementar el módulo y hacer pruebas pudimos entonces llegar a las siguientes ventajas y desventajas frente al sistema anterior.

#### Ventajas encontradas

- **Consistencia en entornos controlados:** En un entorno educativo, donde los dispositivos pueden ser diversos, el uso de Google Cloud garantiza uniformidad, sin depender del navegador.
- **Mejor experiencia para niños:** La voz generada por Google TTS es mucho más clara, agradable y menos robótica, lo cual mejora la experiencia de niños en etapas de aprendizaje temprano.
- **Procesamiento del lado servidor:** Permite registrar transcripciones, sintetizar respuestas y controlar tiempos de expiración o limpieza de audios (como se implementó en `google_tts_stt.py`).
- **Facilidad de mantenimiento:** Cualquier error puede ser trazado mediante logs del servidor, algo imposible con APIs web del navegador.

#### Desventajas encontradas

- **Costo por uso:** Google Cloud tiene un costo por cada segundo de audio procesado, lo cual debe ser monitoreado con límites o alertas.
- **Requiere conexión a internet:** Al depender de APIs externas, se necesita conexión estable, lo cual puede ser un reto en zonas rurales. Este sistema es el que más información mueve ya que usa archivos de audio y no solo texto.
- **Tiempos de respuesta:** Aunque mínimos, los tiempos de respuesta son ligeramente mayores que el TTS local.

#### Conclusión

La decisión de usar Google TTS y STT responde a la necesidad de robustez, calidad y escalabilidad en un contexto educativo diverso. A pesar de los costos y dependencia de la nube, los beneficios en precisión, experiencia del usuario y facilidad de integración justifican plenamente la elección frente al uso de soluciones nativas del navegador como `webkitSpeechRecognition`.

## Justificación Técnica. Almacenamiento de Archivos en el VPS vs Integración con Google Drive

Durante el diseño de la herramienta BOTE3T, uno de los primeros módulos que se creó fue el de subir documentos a la base de datos vectorial ubicada en Pinecone. Por simplicidad, al inicio del proyecto se decidió no usar la opción empleada en GPTE3T (el chatbot anterior que sirvió como punto de partida), la cual consistía en un script ejecutado manualmente en Google Colab. Esta plataforma, aunque útil para correr código en Python, no permite automatizar ejecuciones, por lo que todo el proceso debía ser iniciado deliberadamente por el equipo técnico.

Inicialmente, se implementó una versión que obtenía los archivos desde una carpeta localizada dentro del servidor (VPS), y tras configurar una tarea cron para que este script se ejecutara cada 24 horas, se logró establecer un sistema confiable y funcional. Sin embargo, la principal dificultad fue encontrar una forma sencilla de modificar los archivos alojados en esa carpeta del VPS, especialmente pensando en docentes sin conocimientos técnicos. No se halló una solución accesible para este reto.

Buscando alternativas, el director del proyecto recomendó emplear un WebService de su autoría que ya había usado anteriormente para propósitos similares. Este servicio permite obtener archivos directamente desde una carpeta compartida de Google Drive, lo que resuelve el problema de accesibilidad y facilita la interacción para usuarios no técnicos. Al integrar este WebService con un script programado para ejecutarse cada 24 horas en el servidor, se logró un sistema simple, automatizado y escalable para mantener actualizada la base de datos vectorial de Pinecone.

Después de tener ambas implementaciones comparamos ambas soluciones, estas fueron las ventajas y desventajas que encontramos:

Criterio	Almacenamiento en VPS	Google Drive como fuente externa
Facilidad de uso para docentes	Requiere acceso SSH o FTP	Subida desde interfaz web de Google Drive
Automatización de carga	Totalmente automatizado vía tarea programada	Totalmente automatizado vía WebService y tarea programada
Límites técnicos	Acepta casi todos los formatos disponibles. Depende del espacio disponible en el servidor	Solo acepta PDF, DOCX y TXT. Límite de 100 páginas o 50MB por archivo
Costo en espacio y mantenimiento	Aumenta consumo del disco del servidor. Uso alto de CPU, para convertir entre formatos	Cero uso de disco del VPS. Uso mínimo de CPU
Escalabilidad y portabilidad	Portátil si se tiene el espacio disponible	Portátil, accesible desde cualquier ubicación
Privacidad y control interno	100% bajo control del servidor propio	Riesgo si permisos no son bien configurados
Interacción con el sistema	Interno, más rápido y flexible	Basado en una API de Google Apps Script

<b>Complejidad técnica de configuración</b>	Alta para usuarios finales	Solo requiere compartir carpeta con su ID público
---	----------------------------	---

## Conclusión

Aunque almacenar archivos directamente en el VPS puede ofrecer más control y flexibilidad técnica, la integración con Google Drive fue seleccionada como solución final debido a su facilidad de uso, portabilidad, escalabilidad y mínima intervención técnica requerida. Esta elección permite que cualquier docente o colaborador cargue nuevos documentos sin depender del equipo de desarrollo, democratizando así el proceso de gestión del conocimiento y facilitando la expansión del sistema a nuevas instituciones o contextos educativos.

No obstante, se considera que ambos sistemas son suficientemente robustos, por lo que se ha decidido mantener ambos códigos disponibles dentro del proyecto. El sistema basado en carpetas locales se preserva como opción viable en caso de que el chatbot deba ser instalado en un servidor local o en entornos sin acceso a Google Drive, asegurando así mayor flexibilidad y adaptabilidad del sistema a distintos contextos técnicos o institucionales.