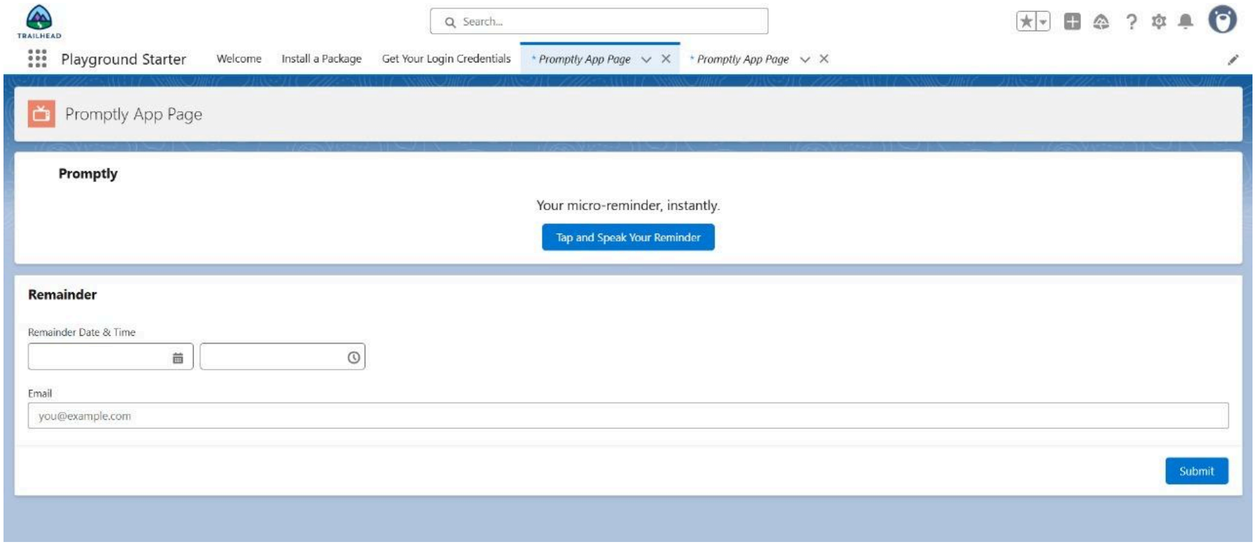
PROMPTLY

PHASE 7: INTEGRATION & EXTERNAL ACCESS

1. Project Context: Stateless Architecture and Integration Strategy

The core architectural constraint of the **"Promptly"** application is its design as a **stateless, self-contained, client-side application**. This means the application's entire runtime environment—from listening to the final notification—operates exclusively within the user's web browser tab.

The integration strategy is based on **zero-dependency Native APIs**. This approach inherently eliminates the need for managing API keys, server-side logic (like Apex), external authentication, or subscription costs, focusing on maximum speed and privacy.



Your micro-reminder, instantly.

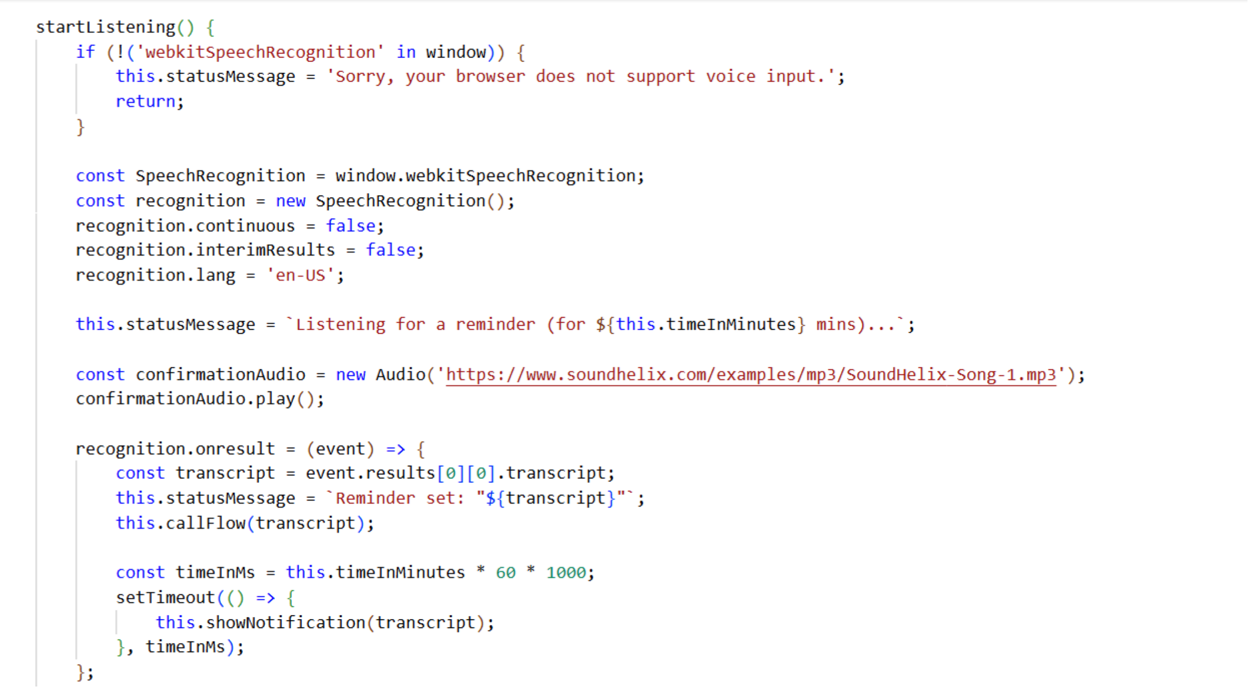
---

2. Integration with Native Browser APIs

The application integrates with two essential native APIs to automate the reminder process:

Web Speech API (`webkitSpeechRecognition`)

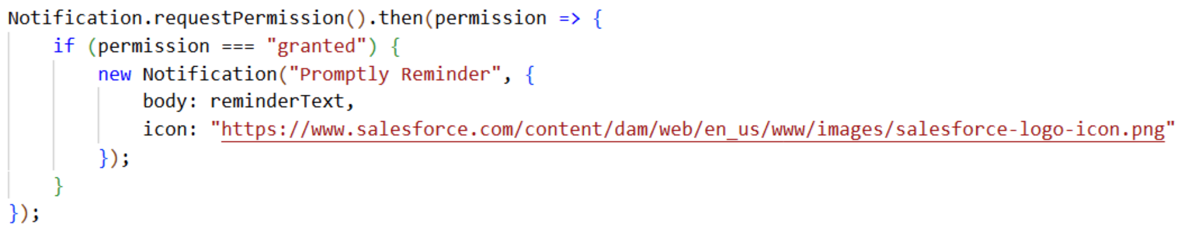
- **Function**: Provides the crucial input mechanism by accessing the device's microphone and automatically converting the spoken audio into a reminder text string.
- **Integration Detail**: The application specifically utilizes the `webkitSpeechRecognition` object, which is directly instantiated and configured within the main JavaScript file (`promptly.js`). It is set to run in a non-continuous mode to capture a single, complete thought efficiently.
- **External Access**: **None**. The voice-to-text processing occurs locally on the user's device, ensuring the spoken reminder content remains private and is never transmitted over a network.



```
startListening() {
    if (!('webkitSpeechRecognition' in window)) {
        this.statusMessage = 'Sorry, your browser does not support voice input.';
        return;
    }

    const SpeechRecognition = window.webkitSpeechRecognition;
    const recognition = new SpeechRecognition();
    recognition.continuous = false;
    recognition.interimResults = false;
    recognition.lang = 'en-US';

    this.statusMessage = `Listening for a reminder (for ${this.timeInMinutes} mins)...`;

    const confirmationAudio = new Audio('https://www.soundhelix.com/examples/mp3/SoundHelix-Song-1.mp3');
    confirmationAudio.play();

    recognition.onresult = (event) => {
        const transcript = event.results[0][0].transcript;
        this.statusMessage = `Reminder set: "${transcript}"`;
        this.callFlow(transcript);

        const timeInMs = this.timeInMinutes * 60 * 1000;
        setTimeout(() => {
            this.showNotification(transcript);
        }, timeInMs);
    };
```

Web Notifications API

- **Function**: Serves as the final output mechanism, pushing the time-based reminder message to the user's operating system (desktop or mobile) even if the browser tab is minimized or out of focus.
- **Integration Detail**: The application calls `Notification.requestPermission()` to ensure user consent, and then uses the `new Notification()` constructor to display the final reminder text after the timer expires.



```
Notification.requestPermission().then(permission => {
    if (permission === "granted") {
        new Notification("Promptly Reminder", {
            body: reminderText,
            icon: "https://www.salesforce.com/content/dam/web/en_us/www/images/salesforce-logo-icon.png"
        });
    }
});
```

- **External Access**: **None**. The notification is managed by the local operating system and browser, ensuring the output is also self-contained.

---

3. External Access & Data Management Policy

Area	Status	Technical Policy
Data Access & Storage	**Stateless**	Reminder text is held only in the browser's temporary memory for the 3-minute duration. There is **no persistent storage** (e.g., LocalStorage, IndexedDB, or external database connection).

Network Communication	**None**	After the initial download of the `index.html` file, the application makes **zero API calls** to any external service, database, or server for its core functionality.
Third-Party Dependencies	**None**	The codebase contains no calls to external SDKs, libraries, or APIs (e.g., Google, Amazon, geolocation services), simplifying maintenance and security auditing.

---

4. Security and Privacy Implications

- **Data Security**: By eliminating all network communication and persistent storage, the risk of data interception (Man-in-the-Middle attacks) or data breaches is **completely mitigated**.
- **Privacy Assurance**: The architecture guarantees that the user's spoken thoughts and reminder contents are processed and destroyed locally, adhering to a strict **privacy-by-design** principle.
- **Governor Limits**: Since no server-side resources or database transactions are involved, the application is **immune** to Salesforce Governor Limits or any similar operational quotas.