GeoNotes:
A Location-Aware Note-Taking Web Application Using Browser APIs

Aditya Saxena
Course: Modern Web Development
Department of Computer Science
`aditya.saxena@torontomu.ca`

May 30, 2025

Abstract

GeoNotes is a browser-based location-aware note-taking application designed to demonstrate the practical integration of multiple browser APIs. This tutorial walks students through the full-stack development of a modern client-side web app using Geolocation, LocalStorage, Notifications, and Clipboard APIs. The project emphasizes progressive enhancement, offline storage, and user-centric interaction.

1 Introduction

Modern web browsers offer a rich set of JavaScript APIs that empower developers to create applications with native-like functionality. GeoNotes provides an applied context for students to learn about these APIs by building a lightweight Progressive Web App (PWA) that tags notes with geographical metadata, stores them locally, and notifies users contextually.

2 System Architecture

The application architecture is entirely front-end based, built using HTML5, CSS3, and vanilla JavaScript. Fig. 1 outlines the high-level architecture.

3 Browser APIs Used

This section provides a comprehensive overview of the browser APIs integrated into GeoNotes. Each API plays a unique role in enabling interactive, persistent, and context-aware features essential for a modern web application.

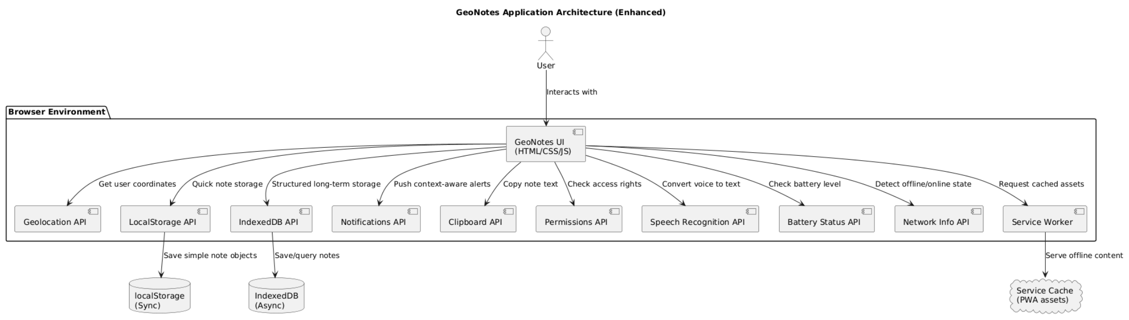

Figure 1: Enhanced system architecture of the GeoNotes web application.

3.1 Geolocation API

The Geolocation API allows the application to access the user's physical location, typically as latitude and longitude coordinates. This is used in GeoNotes to tag each note with its geographical context, enabling features like location-based reminders.

3.2 LocalStorage API

LocalStorage is a synchronous key-value storage mechanism provided by the browser. It is ideal for small-scale persistence such as user preferences or recent notes. GeoNotes uses it for quick storage and retrieval of note objects using unique timestamp-based keys.

3.3 IndexedDB API

IndexedDB is an asynchronous, NoSQL-like database that stores complex structured data. Unlike LocalStorage, it allows indexing, querying, and managing large volumes of notes efficiently. GeoNotes leverages it for scalability, enabling advanced features like search, categorization, and offline-first behavior.

3.4 Notifications API

This API enables the app to deliver system-level notifications even when the tab is inactive. In GeoNotes, it is used to notify users when they revisit a location where a note was previously saved, enhancing engagement through contextual prompts.

3.5 Clipboard API

The Clipboard API provides programmatic access to the system clipboard. It allows users to copy content from a note with a single click. This improves usability, especially when notes need to be reused in other applications.

3.6 Permissions API

The Permissions API checks and manages the user's consent for accessing sensitive features such as location and notifications. GeoNotes uses it to gracefully handle denied permissions and provide fallback UI or instructional feedback.

3.7 Network Information API

This API provides information about the user's current network status. GeoNotes uses it to detect whether the user is offline or has limited connectivity and responds by warning the user or switching to cached content to maintain functionality.

3.8 Speech Recognition API

Part of the Web Speech API, the Speech Recognition interface enables voice-to-text input. In GeoNotes, it allows users to dictate notes by voice, enhancing accessibility and hands-free usage for mobile or multitasking users.

3.9 Battery Status API

This optional API retrieves the current status of the device's battery. GeoNotes can use this to avoid initiating write-heavy operations when the battery is low, ensuring that critical user data is not lost due to unexpected power loss.

3.10 Service Worker API

The Service Worker API enables the app to cache assets and resources, providing offline support and faster load times. GeoNotes registers a service worker to allow users to access and interact with their notes even without an internet connection.

4 Implementation Steps

4.1 HTML Structure

The HTML file contains a minimal form-based layout that captures user input and organizes saved notes in a visually accessible format. Key elements include:

- An input field for the note title
- A textarea for the note content
- A button to save the note
- A section to dynamically display saved notes

This structure ensures semantic clarity and smooth integration with event-driven JavaScript.

4.2 JavaScript Logic

GeoNotes follows a modular, event-driven architecture. The application's core logic is explained step by step:

1. **Capture Geolocation:** When a user clicks the save button, the app requests their current location using `navigator.geolocation.getCurrentPosition()`. This position data is attached to each note.

2. **Serialize and Store in LocalStorage:** A note object is constructed from user input and location data, serialized using `JSON.stringify()`, and saved in `localStorage` with a timestamp-based key.

3. **Optionally Store in IndexedDB:** For enhanced functionality, the same note is asynchronously saved in `IndexedDB`. This enables efficient querying and long-term scalability.

4. **Use Permissions API:** Before accessing features like notifications or geolocation, the app queries `navigator.permissions` to determine the current state (granted, denied, prompt). This ensures respectful and secure access control.

5. **Detect Network State:** The `navigator.onLine` flag is used to determine connectivity. The app may warn the user or defer sync operations if offline.

6. **Enable Voice Input:** Using the `SpeechRecognition` API, users can dictate note content, which is transcribed and inserted into the note field. This supports accessibility and hands-free operation.

7. **Service Worker Registration:** Upon loading, the app registers a service worker. This worker caches assets and serves them when offline, supporting Progressive Web App (PWA) capabilities.

8. **Trigger Notification:** Upon matching a previously saved location with the user's current location, a push-style notification is triggered using the `Notification` API.

9. **Copy Note Text:** Each displayed note has an associated copy button. Clicking it triggers the `Clipboard API` to copy the note text into the system clipboard, enhancing portability.

This logical flow ensures a progressive enhancement approach that accommodates both basic use cases and advanced functionality depending on the user's device and permissions.

5 Educational Outcomes

Students will:

- Understand and implement multiple browser APIs

- Use `Permissions API` to manage user consent workflows
- Detect and respond to network and power state using contextual APIs
- Implement both voice input and clipboard-based extraction
- Configure service workers for caching and offline support

6 Conclusion

GeoNotes offers a rich, hands-on opportunity to master modern browser capabilities. By incorporating advanced APIs like IndexedDB, SpeechRecognition, and Service Workers, learners build a deeper understanding of scalable, offline-capable, and user-friendly web applications. This project not only reinforces foundational front-end skills but also equips learners for real-world progressive web app development.

7 References

1. MDN Web Docs. ”Using the Geolocation API.” [Online]. Available: `https://developer.mozilla.org/en-US/docs/Web/API/Geolocation_API`
2. MDN Web Docs. ”Using the Web Storage API.” [Online]. Available: `https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage`
3. MDN Web Docs. ”IndexedDB API.” [Online]. Available: `https://developer.mozilla.org/en-US/docs/Web/API/IndexedDB_API`
4. MDN Web Docs. ”Notifications API.” [Online]. Available: `https://developer.mozilla.org/en-US/docs/Web/API/Notifications_API`
5. MDN Web Docs. ”Clipboard API.” [Online]. Available: `https://developer.mozilla.org/en-US/docs/Web/API/Clipboard_API`
6. MDN Web Docs. ”Permissions API.” [Online]. Available: `https://developer.mozilla.org/en-US/docs/Web/API/Permissions_API`
7. MDN Web Docs. ”Network Information API.” [Online]. Available: `https://developer.mozilla.org/en-US/docs/Web/API/Network_Information_API`
8. MDN Web Docs. ”Web Speech API.” [Online]. Available: `https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API`
9. MDN Web Docs. ”Battery Status API.” [Online]. Available: `https://developer.mozilla.org/en-US/docs/Web/API/Battery_Status_API`
10. MDN Web Docs. ”Service Workers.” [Online]. Available: `https://developer.mozilla.org/en-US/docs/Web/API/Service_Worker_API`