

PROMPTLY

PHASE 8: DATA MANAGEMENT & DEPLOYMENT

1. Data Management Strategy (The Ephemeral Model)

The core data management strategy is based on the principle of **ephemeral, in-memory data storage**. This completely bypasses the security and complexity associated with traditional data persistence.

Aspect	Technical Rationale	Operational Policy
Data Persistence	The application is stateless , meaning it intentionally avoids all forms of permanent storage. The absence of a server or database eliminates the requirement for data schema design or SQL integration.	Zero Data Storage. Reminder text is never written to a server, external database, or client-side permanent storage (e.g., Local Storage).
Data Flow & Storage	Reminder text is captured from the Web Speech API and lives only as a JavaScript variable within the browser's volatile memory (RAM).	Data is strictly in-memory . The variable is only accessible within the scope of the function that executes the timer.
Data Retention	Data retention is tied directly to the core functionality via the setTimeout() function. Once the timer completes and the notification is pushed, the variable reference is lost.	Ephemeral Lifespan. The maximum lifespan of the data is the pre-set reminder time (e.g., 3 minutes). The data is destroyed automatically upon task completion or tab closure.

Backup Strategy	Since the application is stateless and holds no historical user data or custom configurations, a data backup strategy is unnecessary and irrelevant to the architecture.	Not Applicable. The source code itself is protected through Git version control.
------------------------	--	---

2. Deployment Architecture: Minimalist and Globally Distributed

The project utilizes a **minimalist, distributed deployment model** designed for zero cost, maximum global availability, and high maintainability.

Element	Artifact & Location	Technical Rationale
Project Artifact	A single, self-contained index.html file.	This artifact encapsulates all necessary code (HTML structure, CSS styling, and JavaScript logic), eliminating the need for complex asset management, build scripts, or dependency resolution
Source Control	GitHub Repository (Public)	Provides robust version control, collaboration history, and a readily available source for deployment.
Hosting Platform	GitHub Pages	Serves the single index.html file globally. This platform provides free hosting and leverages a global CDN (Content Delivery Network) , ensuring rapid load times worldwide.

Server Requirements	Zero Backend	The application runs exclusively in the client's web browser, requiring no server-side technologies (e.g., Python, Node.js, PHP) or expensive cloud infrastructure (AWS/Azure/GCP). This is the key enabler of the project's zero-cost operational model .
----------------------------	---------------------	---

3. Deployment Steps

1. **Commit Code:** Finalize the `promptly.html` file and commit it to the main branch of the public GitHub repository.
2. **Enable GitHub Pages:** In the repository settings, configure **GitHub Pages** to serve the content from the main branch.
3. **URL Activation:** The application becomes instantly available and globally accessible via its public GitHub Pages URL.