AI Context Vault - Profile Manager Documentation

Overview

The Profile Manager feature in AI Context Vault allows users to create and manage global, AI-agnostic profiles. These profiles can be injected into any AI chat interface, ensuring consistent context for various scenarios (e.g., developer or business contexts).

Storage Schema

Profiles are stored in `chrome.storage.local` with keys formatted as:

`ctx_profiles_{ALIAS_NAME_WITH_UNDERSCORES}`

Profile Data Structure

```
interface ProfileData {
  id: string;
  alias: string; // e.g., "Developer", "Business"
  category: "developer" | "business" | "custom";
  summary: string;
  rules: string; // CSV list of selected frameworks/languages or
        business rules
  active: boolean;
  created: number;
  lastModified: number;
}
```

UI Flow

Inject Overlay (`inject.js`)

A new tab named **Profiles**:

- Lists all saved profiles by their alias.
- Inline editing capability similar to bookmarks/context.
- Star icon/button to set the profile as active for injection.
- **[+ New Profile]** button redirects users to the form on `options.html`.

Options Page (`options.html`)

- **Form Page**:
  - Dropdown: Select profile type (`Developer`, `Business`, `Custom`).
  - Text Input: Alias (prepopulated based on dropdown choice; editable).

    - Textarea: Summary
    - Checkboxes: Frameworks/Languages for Developer; Business Attributes for Business
    - Large Textarea: Generated CSV of selected checkboxes

Validation:

- Prevents creating profiles with duplicate aliases.

Checkbox Fields for Developer Profile

Checkbox selections update a CSV-formatted list.

Programming Languages

- JavaScript
- Python
- TypeScript
- Java
- C#
- Go
- Ruby
- PHP
- Swift
- Kotlin
- Rust
- C++
- SQL
- HTML/CSS

Frameworks & Libraries

- React
- Angular
- Vue
- Svelte
- Next.js
- Nuxt.js
- Django
- Flask
- Express.js
- Ruby on Rails
- .NET
- Spring
- Laravel
- Flutter
- React Native

Example CSV Result

Selecting React, Next.js, TypeScript:

```
React, Next.js, TypeScript
```

Checkbox Fields for Business Profile

Checkbox selections update a CSV-formatted list of business attributes.

- Marketing Strategy
- Target Audience
- SEO Priorities
- Sales Goals
- Industry
- Communication Tone (Formal, Casual, Technical, Friendly)

Example CSV Result

Selecting Marketing Strategy, Sales Goals, Formal:

```
Marketing Strategy, Sales Goals, Formal
```

Global Context Introduction Templates

These intros precede the injected context:

Developer Template

```
‼️ GLOBAL PROFILE: Developer

📌 SUMMARY:
This profile guides the AI coding assistant to adhere strictly to
the user's programming languages, frameworks, coding standards,
and best practices as listed.

🔪 SELECTED TECHNOLOGIES:
{Comma-separated CSV of selected languages/frameworks}
```

Business Template

```
‼️ GLOBAL PROFILE: Business

📌 SUMMARY:
This profile provides critical business context, guiding the AI
to consistently respect business strategies, communication style,
and core objectives.

🔪 BUSINESS ATTRIBUTES:
{Comma-separated CSV of selected attributes}
```

Custom Template

```
‼️ GLOBAL PROFILE: {Alias}

📌 SUMMARY:
{User-generated summary}

🔪 SPECIFIC RULES:
{User-generated rules or CSV content}
```

Active Profile Mechanism

- Only one profile can be “active” at any given time.
- Users toggle the active profile via a star icon/button in the overlay UI.
- The active profile auto-injects at the top of every prompt, clearly delineated from other context entries.

GitHub Gist Sync

Profiles are synced to GitHub Gist alongside bookmarks and contexts:

- Extend `gatherAllContextData()` to include keys beginning with `ctx_profiles_`.
- Maintain existing sync logic (race handling, conflict resolution).

Implementation Tasks (For Cursor AI)

Inject.js

- Create Profile tab UI
- Implement inline edit/delete
- Active star toggle logic
- Redirect for new profile form to `options.html`

Options.js

- Form UI with dropdowns/checkboxes
- Alias validation to avoid duplicates
- Save logic
- CSV generation from checkboxes

ContextStorage.js

- Implement `saveProfile()`, `getProfiles()`, `updateProfile()`, `deleteProfile()`
- Ensure key structure aligns with existing context/bookmark pattern

---

Additional Notes

- Maintain UI consistency with existing bookmarks/context sections.
- Ensure all text and interaction patterns match established UX in AI Context Vault.

This comprehensive guide outlines clear data structures, UI flow, and implementation steps, facilitating efficient development and integration of the Profile Manager.