Product Specification: Prompt Builder

This document provides detailed specifications, user stories, and acceptance criteria for the Prompt Builder application. It is intended to guide the design, development, and testing phases of the project.

1. Product Charter

Product: Prompt Builder

Vision: To deliver an intelligent and dynamic ecosystem for prompts that guides users in crafting effective prompts and allows teams to seamlessly Create, share, discover, and continuously improve them through collaboration.

Core Problem: The effectiveness of powerful AI tools is gated by a user's ability to write good prompts, creating a significant barrier for non-technical users. Valuable prompts are frequently lost in scattered notes and chats, preventing reuse and team collaboration. This isolation means teams cannot build on collective knowledge, and Users lack clear guidance on leveraging AI effectively, leading to inconsistent and suboptimal outcomes.

Scope (Initial Version):

- A standalone web application.
- User authentication and management with role-based access control.
- Private and Shared team prompt libraries.
- AI-assisted prompt creation workflow.
- Mandatory metadata (tags) for categorization and structure for prompts.
- Advanced search, filtering, and Library Prompt recommendation.
- Collaborative features: version history, upvotes, and usage performance metrics.
- Administrative features for managing LLM versions.

Out of Scope (Initial Version):

- A publicly consumable API for third-party integrations.
- Automated prompt execution and formal evaluation systems.
- Complex version management features (e.g., branching, merging). A simple linear history will be maintained.
- Bookmarking or "favorites" functionality for prompts.

2. User Roles

The application will support two distinct user roles with different levels of permissions:

1. User (Default Role):

- **Prompt Creation:** Can create new prompts and save them as Private or Public.

- **Editing & Collaboration:** Can directly edit any prompt they own (both Private and Public). For Public prompts created by other users, they cannot edit directly but can contribute by creating a new, improved version.
- **Access:** Cannot access any system-level administrative settings.

2. Admin:

- **Inherited Permissions:** Has all the permissions of the 'User' role.
- **Global Editing:** Can directly edit *any* prompt in the system, including Public prompts they do not own.
- **Global Management:** Can archive or unarchive any prompt.
- **System Administration:** Can access the Admin Panel to manage system-level settings, such as LLM versions and user roles.

3. Data Models

This section defines the core data structures for the application.

3.1 Prompt Object

Field	Type	Description
promptId	UUID	Primary key for the prompt.
title	String	Unique, user-defined title for the prompt.
promptText	Text	The full content of the prompt.
ownerId	UUID	Foreign key referencing the User who owns it.
isPublic	Boolean	true for shared library, false for private.
targetLLM	String	The specific LLM version this prompt is for (e.g., "Gemini 2.6 Pro").
discoveryTags	Array	Array of foreign keys referencing DiscoveryTag objects.
versionHistory	Array	An array of version objects,

		each with versionNumber, promptText, note, authorId, and timestamp.
upvotes	Integer	The total number of upvotes.
downvotes	Integer	The total number of downvotes.
usageCount	Integer	How many times the prompt has been used.
status	Enum	ACTIVE, ARCHIVED, NEEDS_VALIDATION.
createdAt	Timestamp	Timestamp of initial creation.
updatedAt	Timestamp	Timestamp of the last update.
lastUpdatedBy	UUID	Foreign key referencing the User who made the last update.

3.2 User Object

Field	**Type**	**Description**
userId	UUID	Primary key for the user.
email	String	User's email, from Google SSO.
displayName	String	User's name, from Google SSO.
role	Enum	USER, ADMIN.

3.3 DiscoveryTag Object

Field	Type	Description
tagId	UUID	Primary key for the tag.
tagName	String	The name of the tag (e.g., "Dept" “Project”).
tagValues	String	The name of the tag (e.g., "QA" “Product”).
createdBy	UUID	Foreign key referencing the Admin User who created it.
Mandatory	Boolean	true for Mandatory, false for Non-Mandatory.
Status	Boolean	true for Active, false for Inactive

3.4 RequestTag Object

Field	Type	Description
tagId	UUID	Primary key for the tag.
tagName	String	The name of the tag (e.g., “Persona”).
tagValues	String	The name of the tag (e.g., "ProductOwner" “Dev Team Lead”).
createdBy	UUID	Foreign key referencing the Admin User who created it.
Mandatory	Boolean	true for Mandatory,, false for Non-Mandatory.
Status	Boolean	true for Active, false for Inactive

4. UI/UX Guidelines

4.1 General Philosophy

The user interface should be minimalist, clean, and intuitive, prioritizing ease of use for non-technical users. The color palette will be based on "shuttle colors"—a professional theme using light grays, off-white backgrounds, dark text for high contrast, and a single, muted accent color (e.g., a calm blue) for buttons and interactive elements.

4.2 Mockup Wireframes

The following HTML Mockup link provide conceptual wireframes for the key application screens:

- https://drive.google.com/file/d/1P1g1EYysMXsujunPIoYF8solzTxNdwaN/view?usp=drive_link

4.3 Component-Specific Behavior

- **Prompt Recommendations (Journey 2):** When a user starts typing, a dropdown list will appear directly below the main input field, showing the top matching prompts from the library.
- **Prompt Detail Page:** This is a dedicated view for a single prompt. It will display all relevant details: Title, Owner's Name, Creation Date, Last Updated Date/Time, Last Updated By, Status, Usage Count, Upvote/Downvote counts, and a list of all assigned tags. A tabbed interface will be used to show the full prompt text in one tab and the complete version history in another. An "Edit" button will allow authorized users (owner or Admin) to modify these details.
- **Admin Panel Layout:** The "Administration" section will be a distinct area of the application. It will contain three primary tabs: "User Management" for role changes and deletions, "LLM Management" for handling retired versions, and "Tag Management" for creating and maintaining the list of Discovery Tags.

5. System Concepts & Error Handling

5.1 Tag Management & Notifications

Tag Management

The system utilizes two distinct types of tags to aid in prompt creation and discovery:

- **Request Prompt Tags:** These tags are used exclusively during the initial AI-assisted prompt generation phase. Users select from a list of contextual tags (e.g., Project, User Persona, Desired Output Format) to provide the LLM with better guidance for crafting a high-quality initial draft. These tags are temporary and are **not** stored with the final prompt record.

- **Discovery Prompt Tags:** These are permanent tags stored with every prompt record and are essential for the application's ecosystem. They are used for searching, filtering the library, and powering the recommendation engine. To maintain consistency and prevent duplication, the list of available Discovery Tags is managed exclusively by users with the 'Admin' role.

Notifications

Key system events will trigger automated email notifications to relevant users to keep them informed. Notifications will be sent for events such as when a prompt's underlying LLM is retired, notifying the owner (or Admins for orphaned prompts) of the need for re-validation.

5.2 Error Handling

The system will provide clear, user-friendly error messages, typically as inline text below the relevant field or as a temporary toast notification.

- **Non-Unique Title:** When a user tries to save a prompt with a title that is already taken, an inline error message will appear under the title field stating: "A prompt with this title already exists. Please choose a unique title."
- **AI Generation Failure:** If the backend call for AI-assisted prompt generation fails, a toast notification will appear with a message like: "Sorry, the AI assistant is currently unavailable. You can still write your prompt manually." The user can then proceed without interruption.
- **Permission Denied:** If a user with the User role attempts to access an Admin-only URL, they will be redirected to a simple page with the message: "Access Denied. You do not have the necessary permissions to view this page."

6. User Stories & Acceptance Criteria

User Journey 1: Create and Add New Prompt in Library

Description: A user creates a new prompt from scratch, optionally using AI assistance, and saves it to either their private library or the shared team library.

User Story 1.1: User Authentication

> As a new or returning user, I want to log in to the system using my Google SSO, so that I can securely access my prompts.

Acceptance Criteria:

- **Given** I am on the login page, **then** I see a "Sign in with Google" button.
- **Given** I click the button and successfully authenticate with Google, **then** I am redirected to the application's main dashboard.

User Story 1.2: AI-Assisted Prompt Generation

> As a user, I want to provide initial context using 'Request Prompt Tags' (like use case, project, persona), so that the system can generate a high-quality draft prompt for me to refine.

Acceptance Criteria:

- **Given** I am creating a new prompt, **then** I am presented with fields to input 'Request Prompt Tags'.
- **Given** I provide the context and click "Generate", **then** the system generates a draft prompt in the editor.
- **Given** the draft is generated, **then** I am able to freely edit or completely rewrite the text in the editor.
- **Given** the AI generation fails, **then** I see a user-friendly error message, and the editor remains available for me to write the prompt manually.

User Story 1.3: Saving and Categorizing a New Prompt

> As a user, after finalizing my prompt, I want to save it with a unique title, define its type (Private or Public), select the target LLM, and add mandatory 'Discovery Prompt Tags' so it can be easily found later.

Acceptance Criteria:

- **Given** I am ready to save a prompt, **then** I must provide a title that is unique across the entire system.
- **Given** I enter a title that already exists and try to save, **then** I see an inline error message "A prompt with this title already exists. Please choose a unique title." and the save is prevented.
- **Given** I try to save, **then** the system validates that all mandatory fields (Title, Type, LLM, Discovery Tags) are complete.
- **Given** all fields are valid, **when** I click "Save", **then** the prompt is successfully saved to the appropriate library (Private or Public).
- **Given** the prompt is saved as 'Public', **then** it is immediately visible to all team members in the shared library.

User Journey 2: System Recommendations for Best Prompt

Description: A user begins typing a prompt and the system proactively suggests relevant, high-quality prompts from the shared library.

User Story 2.1: Real-time Prompt Recommendation

> As a user, when I start typing in the prompt input field, I want the system to suggest existing public prompts based on matching keywords, tags, upvotes, and usage count, so I can leverage proven solutions.

Acceptance Criteria:

- **Given** I start typing in the main input field, **then** a list of recommended prompts appears in a non-intrusive UI element.
- **Given** recommendations are displayed, **then** they are ranked in order of relevance based on a weighted algorithm: Keywords > Tags > (Upvotes - Downvotes) > Usage Count.
- **Given** I select a recommended prompt, **then** its full text is inserted into the input field, replacing what I had typed.
- **Given** a prompt is inserted, **then** I can edit it before execution.

User Journey 3: Use an Existing Prompt from the Library

Description: A user browses the library to find and use a pre-existing prompt.

User Story 3.1: Browsing and Filtering the Library

> As a user, I want to access the prompt library and be able to search and filter by tags, so I can quickly find the exact prompt I need.

Acceptance Criteria:

- **Given** I navigate to the Library, **then** I see a list of all Public prompts and my own Private prompts.
- **Given** I am viewing the library, **then** I can use a search bar to find prompts by title or content.
- **Given** I am viewing the library, **then** I can filter the list by one or more 'Discovery Prompt Tags'.

User Story 3.2: Using a Prompt from the Library

> As a user, once I find a prompt in the library, I want to click a button to instantly insert its full text into my input window, so I can use or edit it immediately.

Acceptance Criteria:

- **Given** I have found a prompt in the library list, **then** there is a "Use This Prompt" button associated with it.
- **Given** I click the button, **then** I am taken back to the main interface and the full text of the selected prompt is populated in the input field.

User Journey 4: Iterate and Improve a Shared Prompt

Description: A user improves upon a shared prompt, creating a new version while preserving the history of the original.

User Story 4.1: Creating a New Version of a Prompt

> As a user, when I find a public prompt that could be better, I want to create a new

> version of it, edit the text, and add a "Version Note" explaining my improvement, so the team's collective knowledge can evolve.

Acceptance Criteria:

- **Given** I am viewing a public prompt's detail page, **then** I see an "Improve" or "Create New Version" button.
- **Given** I click the button, **then** I am taken to the prompt editor, pre-populated with the content of the version I'm improving.
- **Given** I am in the editor, **then** there is a mandatory "Version Note" field where I must explain my changes.
- **Given** I save the new version, **then** it immediately becomes the default version of that prompt displayed in the library.
- **Given** the new version is saved, **then** its performance metrics (upvotes, usage count) are carried over from the previous version.
- **Given** I am viewing the prompt's detail page, **then** I can access a history of all previous versions and their associated version notes.

User Journey 5: Review Prompt Performance and Manage Contributions

Description: A user tracks the impact and value of the prompts they have shared with the team.

User Story 5.1: Tracking Contribution Metrics

> As a user, I want to see a "My Contributions" dashboard that shows the usage count and upvotes for each prompt I've shared, so I can understand which of my contributions are most valuable to the team.

Acceptance Criteria:

- **Given** I navigate to my dashboard, **then** I see a "My Contributions" section.
- **Given** I am in this section, **then** I see a list of all the prompts I have created.
- **Given** I am viewing the list, **then** each prompt displays its total usage count and current upvote count.
- **Given** a user clicks "Use This Prompt" from a recommendation or the library, **then** the prompt's usage count is incremented by one.

User Story 5.2: Filtering Performance Data

> As a contributor, I want to filter my prompt performance metrics by different time periods (e.g., last 30 days, lifetime), so I can track trends in their usage.

Acceptance Criteria:

- **Given** I am on the "My Contributions" dashboard, **then** the metrics displayed are for

"Lifetime" by default.
- **Given** I am on the dashboard, **then** there is a dropdown filter with options: "Last 30 Days", "This Quarter", "Last 6 Months", "Last Year", and "Lifetime".
- **Given** I select a time period, **then** the usage and upvote metrics in the list update to reflect only that period.

User Journey 6: Provide Feedback on a Shared Prompt

Description: A user provides feedback on a public prompt to help signal its quality to the rest of the team.

User Story 6.1: Upvoting a Prompt

> As a user, after using a public prompt I found valuable, I want to upvote it, so that its quality is recognized and it becomes more discoverable for others.

Acceptance Criteria:

- **Given** I am on the detail page of a public prompt, **then** I see an "upvote" button (e.g., a thumbs-up icon) with the current count.
- **Given** I have not previously upvoted this prompt, **when** I click the button, **then** the count instantly increments by one, and the button appears "active".
- **Given** I have previously upvoted this prompt, **when** I click the active button, **then** my upvote is removed, the count decrements, and the button returns to a "neutral" state.

User Journey 7: Archive an Outdated Shared Prompt

Description: An author or admin archives an outdated prompt to keep the library clean and relevant.

User Story 7.1: Archiving a Prompt and Notifying Active Viewers

> As a prompt owner or an admin, I want to archive an outdated prompt, so that it no longer clutters the active library. As a user viewing a prompt, I want to be notified if it is archived in real-time.

Acceptance Criteria:

- **Given** I am the owner of a shared prompt or an Admin, **when** I am on the prompt's detail page, **then** I see an "Archive" button.
- **Given** I click "Archive", **then** a confirmation message appears explaining the consequences (removed from search/recommendations, but not deleted).
- **Given** I confirm the action, **then** the prompt is flagged as "Archived" and is removed from the main library view, search results, and recommendations.
- **Given** another user is actively viewing the detail page of that same prompt, **when** I confirm the archival, **then** an in-page notification banner immediately appears for that user, stating "This prompt has been archived and is no longer recommended for active

use."
- **Given** I am the author, **then** I can still find the archived prompt in a dedicated "Archived" section of my "My Contributions" dashboard.

User Journey 8: Manage a Retired LLM Version

Description: An administrator manages the lifecycle of an LLM version, and the system handles the impact on existing prompts, including those with inactive owners.

User Story 8.1: Admin Manages LLM Lifecycle

> As an Admin, I want to configure fallback LLMs and mark an existing LLM version as discontinued, so that the system can gracefully handle technology transitions.

Acceptance Criteria:

- **Given** I am in the Admin Panel, **then** I see a section for "LLM Management".
- **Given** I am in LLM Management, **when** I select an active LLM, **then** I have the option to configure a "Fallback LLM" from the list of other active LLMs.
- **Given** I have configured a fallback, **when** I mark the original LLM as "Discontinued", **then** the system triggers the re-validation process for all prompts using that LLM.

User Story 8.2: System Handles Re-validation and Orphaned Prompts

> As the system, when an LLM is retired, I must notify active prompt owners to re-validate their work and reassign orphaned prompts to Admins, so that library quality is maintained.

Acceptance Criteria:

- **Given** an Admin discontinues an LLM, **then** the system identifies all prompts that were using it.
- **Given** an affected prompt has an **active owner**, **then** an email notification is sent to that owner, informing them of the change and the need for re-validation.
- **Given** an affected prompt has an **inactive or deleted owner** (is orphaned), **then** an email notification is sent to all users with the 'Admin' role.
- **Given** a prompt is orphaned, **then** its ownership is automatically transferred to the 'Admin' group, making it manageable by any Admin.
- **Given** any affected prompt (owned or orphaned), **then** it is flagged with a "Needs Validation" status and is deprioritized in search results until it is re-validated and saved.

User Journey 9: Admin User Role Management

Description: An administrator manages user accounts, including their roles and what happens to their contributions when they are removed from the system.

User Story 9.1: Changing a User's Role

> As an Admin, I want to change a user's role between 'User' and 'Admin', so that I can manage system permissions.

Acceptance Criteria:

- **Given** I am in the "User Management" section of the Admin Panel, **then** I see a list of all users and their current roles.
- **Given** I select a user, **when** I change their role from 'User' to 'Admin' (or vice-versa) and save, **then** their permissions are immediately updated.

User Story 9.2: Deleting a User and Reassigning Their Prompts

> As an Admin, I want to delete a user account and reassign ownership of their public prompts, so that I can manage team membership while preserving valuable shared assets.

Acceptance Criteria:

- **Given** I am in the "User Management" section, **when** I select a user and click "Delete", **then** a confirmation modal appears.
- **Given** the user owns Public prompts, **then** the modal informs me that their Private prompts will be deleted, but their Public prompts must be reassigned.
- **Given** the reassignment modal is shown, **then** I am presented with an option to reassign the prompts to another active user (via a searchable dropdown) or to the default 'Admin' group.
- **Given** I select a new owner and confirm the deletion, **then** the user's account and their Private prompts are permanently deleted.
- **Given** the deletion is confirmed, **then** ownership of all their Public prompts is transferred to the new owner I selected.