# B. Methodology Appendix (General Criteria Documentation)

This appendix documents the AI interaction across the 8 steps of the Master Prompt Guide for the SubSentry subscription tracker project. This section provides evidence of understanding how to guide AI for reliable, high-quality product outputs.

## 1. Type of Prompt Engineering Used

For each of the 8 development steps in the SubSentry HTML project, the following primary prompt engineering techniques were employed:

### Step 1-3: Initial Setup and Core Structure

**Technique: Structured Output**

The AI was instructed to generate well-organized, semantic HTML5 code with clear separation of concerns. This included proper document structure with distinct sections for header, main dashboard area, and form components.

**Rationale:** Structured Output was necessary to ensure the codebase remained maintainable and followed web development best practices. This technique guided the AI to produce clean, semantically correct HTML that could be easily extended in later steps.

### Step 4-5: Styling and Visual Design

**Technique: Context-Stitching**

Previous project requirements and brand guidelines (color scheme, typography preferences) were referenced to maintain visual consistency throughout the interface. The AI incorporated a cohesive color palette using CSS custom properties for teal, charcoal, and cream tones.

**Rationale:** Context-Stitching ensured that design decisions aligned with the overall project vision and that styling remained consistent across all UI components, creating a professional and unified user experience.

### Step 6-7: Interactive Functionality

**Technique: Role-Playing**

The AI was guided to simulate the behavior of a subscription management system, implementing realistic user interactions such as form submission, data persistence using localStorage, and dynamic list rendering.

**Rationale:** Role-Playing enabled the AI to understand user workflows and implement features that mimic real-world subscription tracking applications, including add/edit/delete operations and cost calculations.

### Step 8: Dashboard Analytics and Refinements

**Technique: Few-Shot Prompting**

Specific examples were provided for calculating monthly and yearly costs, handling different billing cycles (monthly, yearly, custom intervals), and displaying renewal warnings for subscriptions expiring within 7 days.

**Rationale:** Few-Shot Prompting was critical for complex calculations and business logic, ensuring the AI correctly implemented financial computations and date-based alerts that form the core value proposition of the application.

## 2. Ways of How Prompts Were Optimized at Each Step

### Step 1: Project Initialization

**Optimization:** Added specific keywords "semantic HTML5," "accessibility-first," and "mobile-responsive" to the initial prompt.

**Why Critical:** This optimization ensured the AI prioritized ARIA attributes, proper heading hierarchy, and viewport meta tags from the beginning, avoiding costly refactoring later. The use of semantic elements like `<header>`, `<main>`, and `<section>` improved both SEO and screen reader compatibility.

### Step 2: CSS Architecture

**Optimization:** Specified "use CSS custom properties for all colors" and "implement a mobile-first responsive design approach."

**Why Critical:** This forced the AI to create a maintainable color system using variables, enabling quick theme changes. The mobile-first approach ensured the interface worked seamlessly on all device sizes, with the AI automatically generating appropriate media queries and flexible layouts.

### Step 3: Form Implementation

**Optimization:** Included detailed requirements: "validate all inputs client-side," "provide clear user feedback," and "implement accessible form controls with proper labels."

**Why Critical:** This optimization ensured robust input validation preventing invalid data entry (e.g., negative costs, invalid dates). Clear error messaging improved user experience, while proper label associations enhanced accessibility for assistive technologies.

### Step 4: Data Persistence

**Optimization:** Changed prompt from generic "save data" to specific "implement localStorage with JSON serialization, include error handling for quota exceeded exceptions, and provide fallback behavior."

**Why Critical:** This detailed specification prevented data loss scenarios and ensured the application gracefully handled storage limitations. The AI implemented try-catch blocks around localStorage operations, making the application more resilient to browser restrictions.

### Step 5: Dashboard Statistics

**Optimization:** Added precise calculation requirements: "calculate total monthly cost by converting all subscriptions to monthly equivalents (yearly ÷ 12, custom based on renewal date), round to 2 decimal places, and update in real-time on any subscription change."

**Why Critical:** This optimization eliminated calculation errors and ensured accurate financial reporting. The AI correctly handled different billing cycles, providing users with reliable cost projections that justify the application's purpose.

### Step 6: List Rendering and UI Updates

**Optimization:** Specified "implement dynamic DOM updates without page reload, use event delegation for performance, and include loading states for better UX."

**Why Critical:** This forced the AI to implement efficient rendering logic that updates only changed elements rather than regenerating the entire list. Event delegation improved performance with large subscription lists, while loading states provided visual feedback during operations.

### Step 7: Renewal Alerts

**Optimization:** Changed from vague "show upcoming renewals" to specific "display 'Renewing Soon' badge with warning styling for subscriptions expiring within 7 days, calculate days until renewal, and sort by urgency."

**Why Critical:** This optimization created actionable alerts that help users avoid unwanted charges. The AI implemented date comparison logic using JavaScript Date objects, ensuring accurate countdown calculations and prominent visual warnings using color-coded badges.

### Step 8: Polish and Accessibility

**Optimization:** Added comprehensive accessibility requirements: "ensure WCAG 2.1 AA compliance, implement keyboard navigation for all interactive elements, add focus indicators, and test color contrast ratios."

**Why Critical:** This final optimization ensured the application was usable by people with disabilities. The AI added proper focus management, keyboard event handlers, and sufficient color contrast (minimum 4.5:1 for normal text), making the application inclusive and professional-grade.

## Summary

Each prompt engineering technique was strategically selected to address specific development challenges, while iterative optimizations refined the AI's output quality at every step. The combination of Structured Output for code organization, Context-Stitching for consistency, Role-Playing for realistic functionality, and Few-Shot Prompting for complex logic resulted in a production-ready subscription tracking application with professional UI, robust functionality, and excellent accessibility.