# Master Blueprint Integration Index for DynamicPromptEnhancer Feature

## Overview

This document provides detailed instructions for integrating the DynamicPromptEnhancer Feature into the Setarcos App Master Technical Blueprint. It ensures proper indexing, cross-referencing, and consistency with the existing blueprint structure.

## Primary Integration Point

### Section 9.3.1 DynamicPromptEnhancer

Add the DynamicPromptEnhancer feature as a new subsection 9.3.1 under the existing Context-Specific Tone Prompt Workflow section (9.3). This placement ensures logical flow from the tone prompt workflow to the dynamic enhancement system.

```

Location: Section 9.3.1

Action: Add new subsection

File: dynamic\_prompt\_enhancer\_feature.txt

```

## Cross-Reference Updates

### Table of Contents

Update the Table of Contents to reflect the new subsection:

```

\* [9.3 Context-Specific Tone Prompt Workflow](#93-context-specific-tone-prompt-workflow)

\* [9.3.1 DynamicPromptEnhancer](#931-dynamicpromptenhancer)

\* [9.3.1.1 Feature Goal](#feature-goal)

\* [9.3.1.2 System Architecture](#system-architecture)

\* [9.3.1.3 Database Models](#database-models)

\* [9.3.1.4 API Endpoints](#api-endpoints)

\* [9.3.1.5 User Experience Flow](#user-experience-flow)

\* [9.3.1.6 Advanced Enhancement Techniques](#advanced-enhancement-techniques)

\* [9.3.1.7 Implementation Considerations](#implementation-considerations)

\* [9.3.1.8 Implementation Phases](#implementation-phases)

```

### Section 5.1 Ask Feature

Update references to prompt enhancement in the Ask Feature section:

```

Location: Section 5.1

Action: Add references to DynamicPromptEnhancer

Details:

- Update the AI Router Integration section to reference the DynamicPromptEnhancer

- Add details about how the Ask feature leverages dynamic prompt enhancement

- Ensure consistency with the user experience flow

```

### Section 9.1 AI Router Design

Update AI Router references to include DynamicPromptEnhancer integration:

```

Location: Section 9.1

Action: Add integration point with DynamicPromptEnhancer

Details:

- Add reference to the prompt enhancement step in the model selection process

- Ensure the router passes necessary context to the enhancer

- Document the flow of enhanced prompts through the router

```

### Section 7.2 Table Definitions

Add the following database tables:

```

Location: Section 7.2

Action: Add table definitions

Tables:

- prompt\_templates

- enhancement\_rules

- prompt\_enhancement\_metrics

- prompt\_cache

- user\_prompt\_preferences

```

### Section 8.3 Endpoint Specifications

Add the following API endpoints:

```

Location: Section 8.3

Action: Add endpoint specifications

Endpoints:

- POST /api/v1/prompts/enhance

- GET /api/v1/prompts/templates/tones

- GET /api/v1/prompts/templates/{template\_id}

- POST /api/v1/prompts/rules

- GET /api/v1/prompts/metrics

```

### Section 10.3 Task Definitions

Add Celery tasks for the DynamicPromptEnhancer:

```

Location: Section 10.3

Action: Add task definitions

Tasks:

- process\_prompt\_enhancement

- update\_enhancement\_metrics

- cache\_prompt\_templates

- analyze\_enhancement\_effectiveness

```

## Integration with Other Features

### Section 5.9 Concepts System

Update concept integration references:

```

Location: Section 5.9

Action: Add references to concept integration in prompts

Details: Document how the Concepts system provides data to the DynamicPromptEnhancer for incorporating relevant philosophical concepts into prompts

```

### Section 6.2 User Progression System

Update user level integration:

```

Location: Section 6.2

Action: Add references to user level adaptation

Details: Document how user progression levels influence prompt complexity and content through the DynamicPromptEnhancer

```

## Component File References

Ensure the following component files are properly referenced throughout the blueprint:

### Frontend Components

- PromptEnhancementProvider.tsx

- TonePreviewGenerator.tsx

- ContextualPromptBuilder.tsx

- UserContextCollector.tsx

- QuestionAnalyzer.tsx

- ConceptIntegrator.tsx

- PromptTemplateRenderer.tsx

- EnhancementSettingsPanel.tsx

- PromptDebugConsole.tsx

- EnhancementMetricsDisplay.tsx

### Backend Components

- dynamic\_prompt\_service.py

- tone\_template\_manager.py

- context\_analyzer.py

- question\_classifier.py

- concept\_integrator.py

- user\_level\_adapter.py

- prompt\_metrics\_service.py

- template\_renderer.py

- enhancement\_rules\_service.py

- prompt\_cache\_service.py

## Implementation Timeline Alignment

Ensure the DynamicPromptEnhancer implementation phases align with the overall project timeline:

```

Location: Section 14.1 Environment Strategy

Action: Align DynamicPromptEnhancer phases with deployment milestones

Details:

- Phase 1 (Core Enhancement Framework): Weeks 1-2

- Phase 2 (Advanced Context Integration): Weeks 3-4

- Phase 3 (Personalization & Learning): Weeks 5-6

- Phase 4 (Optimization & Analytics): Weeks 7-8

```

## Final Verification Steps

1. Verify all section numbers and cross-references are correct

2. Ensure consistent terminology throughout the blueprint

3. Validate database schema compatibility with existing tables

4. Confirm API endpoint naming conventions match the rest of the blueprint

5. Verify integration with AI Router preserves existing functionality

6. Confirm template structure is compatible with existing tone prompt workflow

7. Verify all component file names follow project conventions

8. Ensure the enhancement rules system doesn't conflict with existing rule engines