# Alignment Analysis: Existing Components vs. Refined Ask Feature

## Overview

This document analyzes the alignment between the existing Context-Specific Tone Prompt Workflow and AI Router Design components and the refined Ask Feature, identifying integration points, potential conflicts, and necessary updates.

## Key Alignment Points

### 1. Philosophical Tone System

\*\*Existing Implementation:\*\*

- Tone-specific prompt files stored in version-controlled repository

- Each file corresponds to a specific philosopher/tone (e.g., `prompts/socrates\_tone.txt`)

- AI models have specific philosophical tones they excel at (e.g., Grok: Socratic, Existential, Pragmatic)

- Tone selection maps to specific AI models via the AI Router

\*\*Refined Ask Feature:\*\*

- Introduces 10 philosophical tones with tiered access (Free, Tier 1, Tier 2)

- Includes tone preview system with sample responses

- Provides guided tone recommendations based on question type

- Implements tone exploration and comparison mode

\*\*Alignment Analysis:\*\*

- The refined Ask feature builds upon the existing tone system but expands it significantly

- The existing prompt storage and tone-to-model mapping can be leveraged

- Need to ensure the 10 refined tones map correctly to the 4 AI models in the router

### 2. AI Router Integration

\*\*Existing Implementation:\*\*

- Model Registry with defined strengths, philosophical tones, and costs

- Selection logic based on tone, question complexity, user tier, and cost

- Fallback mechanisms for reliability

- Dynamic prompt construction combining tone instructions and user question

\*\*Refined Ask Feature:\*\*

- Relies on AI Router for model selection based on tone

- Introduces new user flows that may impact router selection criteria

- Adds tone preview functionality requiring sample responses

\*\*Alignment Analysis:\*\*

- The refined Ask feature depends on the existing AI Router architecture

- Need to update the router's philosophical\_tones mappings to include all 10 tones

- The existing dynamic prompt construction can be leveraged for the refined feature

### 3. User Experience Flow

\*\*Existing Implementation:\*\*

- Simple flow: Philosopher Selection → Question Input → API Call → Response

- Backend handles tone prompt retrieval and model selection

- Limited frontend interaction options

\*\*Refined Ask Feature:\*\*

- More complex flow with reflection prompts, smart suggestions, and progressive disclosure

- Mobile-optimized response cards with concept badges and action buttons

- Personalized experience for returning users

- Tone exploration and comparison mode

\*\*Alignment Analysis:\*\*

- The refined Ask feature significantly enhances the user experience

- Backend processing flow remains largely compatible

- Frontend components need substantial updates to support new interactions

### 4. Response Handling

\*\*Existing Implementation:\*\*

- Raw AI response returned to frontend

- Limited post-processing mentioned (potential for concept extraction)

\*\*Refined Ask Feature:\*\*

- Structured response cards with concept badges

- Multiple action buttons (Save, Expand, Clarity, Share)

- Integration with Journal, Seek Clarity, and social sharing

\*\*Alignment Analysis:\*\*

- Need to implement additional post-processing for concept extraction

- Response formatting needs to be standardized for mobile display

- Cross-feature integration points need to be implemented

### 5. Prompt Construction

\*\*Existing Implementation:\*\*

- Dynamic prompt construction combining:

- Retrieved tone-specific prompt

- User question

- Context from previous turns

- System-level instructions

\*\*Refined Ask Feature:\*\*

- Similar dynamic construction but adds:

- Pre-question reflection prompts

- Personalization based on user history

- Concept-aware prompting

\*\*Alignment Analysis:\*\*

- The existing dynamic prompt construction can be extended

- Need to ensure prompt templates support all 10 philosophical tones

- User context integration needs to be enhanced

## Potential Conflicts

1. \*\*Tone-to-Model Mapping\*\*

- The existing system maps philosophical tones to specific AI models

- The refined feature introduces 10 tones that need to be mapped to only 4 models

- Some tones may not align perfectly with any single model's strengths

2. \*\*Response Format Standardization\*\*

- The existing system returns raw AI responses

- The refined feature requires structured responses for concept extraction and mobile display

- Need to ensure consistent formatting across all models

3. \*\*User Context Integration\*\*

- The existing system has basic user context integration

- The refined feature requires more sophisticated personalization

- Need to ensure compatibility with the existing UserContextService

4. \*\*Performance Considerations\*\*

- The tone preview system may increase API calls

- The concept extraction adds processing overhead

- Need to ensure compatibility with existing cost optimization strategies

## Integration Recommendations

1. \*\*Update Philosophical Tone Mappings\*\*

- Extend the MODELS dictionary in the AI Router to include all 10 tones

- Create appropriate mappings for each tone to the optimal AI model

2. \*\*Enhance Prompt Templates\*\*

- Create or update prompt templates for all 10 philosophical tones

- Ensure templates are optimized for each AI model

3. \*\*Implement Response Post-Processing\*\*

- Add concept extraction to the response handling pipeline

- Standardize response formatting for mobile display

4. \*\*Extend User Context Integration\*\*

- Update the UserContextService to track tone preferences and usage

- Implement personalized suggestions based on user history

5. \*\*Optimize Performance\*\*

- Implement caching for tone previews

- Consider batch processing for concept extraction

- Ensure compatibility with existing cost optimization strategies