# AI Router Model and Tier 3 AI Benefits

## Overview

This document outlines the refined AI router model and premium AI benefits for Tier 3 ("Philosophical Sage") users, providing them with exclusive access to top AI models, advanced customization options, and an enhanced philosophical dialogue experience.

## AI Router Architecture for Tier 3 Users

### Core Architecture Enhancements

#### Model Selection Interface

* **Implementation**: Dropdown selector in the Ask feature UI for Tier 3 users
* **Available Models**:
  + Grok (optimized for creative philosophical exploration)
  + Claude (optimized for nuanced ethical reasoning)
  + Gemini (optimized for conceptual connections and visualization)
  + GPT (optimized for diverse philosophical perspectives)
* **Default Setting**: "AI Router Choice" (automatic selection based on question type)
* **UI Integration**: Non-intrusive model selector appears below tone selection
* **Model Information**: Expandable cards with model strengths and optimal use cases

#### User Preference System

* **Global Preferences**: Default model selection for all questions
* **Context-Specific Preferences**: Model preferences by philosophical area
  + Ethics & Moral Philosophy → Claude
  + Metaphysics & Ontology → Grok
  + Epistemology → Gemini
  + Political Philosophy → GPT
  + etc.
* **Preference Storage**: User preferences stored in user\_ai\_preferences table
* **Preference Learning**: System learns from user model switching patterns

#### Enhanced Router Logic

* **Base Logic**: Extends existing AI Router system from Section 9.1
* **Tier-Specific Logic**:
  + Free/Tier 1/Tier 2: Original router logic (cost-optimized, quality-focused)
  + Tier 3: User selection overrides automatic routing when specified
* **Fallback Mechanism**: If selected model is unavailable, system falls back to next best option with notification
* **Performance Monitoring**: Tracks response quality by model for continuous improvement

### Technical Implementation

# src/services/ai\_router\_service.py

class EnhancedAIRouter:

def \_\_init\_\_(self, db, user\_service):

self.db = db

self.user\_service = user\_service

self.base\_router = AIRouter(db) # Original router from Section 9.1

async def route\_request(self, user\_id, question, tone\_id, context=None):

"""Enhanced router with Tier 3 model selection support"""

# Get user tier

user = await self.user\_service.get\_user(user\_id)

user\_tier = user.get('subscription\_tier', 0)

# For non-Tier 3 users, use base router

if user\_tier < 3:

return await self.base\_router.route\_request(user\_id, question, tone\_id, context)

# For Tier 3 users, check for model preference

user\_prefs = await self.db.user\_ai\_preferences.find\_one({"user\_id": user\_id})

# Get selected model if specified

selected\_model = context.get('selected\_model') if context else None

# If no explicit selection for this question, check user preferences

if not selected\_model and user\_prefs:

# Check for philosophical area specific preference

area = self.\_detect\_philosophical\_area(question)

if area and area in user\_prefs.get('area\_preferences', {}):

selected\_model = user\_prefs['area\_preferences'][area]

# Fall back to global preference

elif user\_prefs.get('default\_model'):

selected\_model = user\_prefs['default\_model']

# If model selected (explicitly or via preferences), use it

if selected\_model:

try:

response = await self.\_generate\_response\_with\_model(

selected\_model, user\_id, question, tone\_id, context

)

# Track successful usage

await self.\_track\_model\_usage(user\_id, selected\_model, tone\_id)

return response

except Exception as e:

# Log error and fall back to base router

logger.error(f"Error using selected model {selected\_model}: {str(e)}")

# Notify user of fallback

context = context or {}

context['model\_fallback'] = True

context['original\_model'] = selected\_model

# Fall back to base router logic

return await self.base\_router.route\_request(user\_id, question, tone\_id, context)

async def \_generate\_response\_with\_model(self, model\_name, user\_id, question, tone\_id, context):

"""Generate response using specified model"""

# Get tone prompt

tone\_service = ToneService()

tone\_prompt = await tone\_service.get\_tone\_prompt(tone\_id)

# Apply user customizations if available

if context and context.get('customizations'):

tone\_prompt = self.\_apply\_customizations(tone\_prompt, context['customizations'])

# Select appropriate model service

if model\_name == "grok":

model\_service = GrokService()

elif model\_name == "claude":

model\_service = ClaudeService()

elif model\_name == "gemini":

model\_service = GeminiService()

elif model\_name == "gpt":

model\_service = GPTService()

else:

raise ValueError(f"Unknown model: {model\_name}")

# Generate response

response = await model\_service.generate\_response(question, tone\_prompt, context)

# Enhance response with model-specific metadata

response['model\_used'] = model\_name

return response

def \_detect\_philosophical\_area(self, question):

"""Detect philosophical area of question"""

# Implementation using NLP classification or keyword matching

# Returns area like "ethics", "metaphysics", "epistemology", etc.

# For now, a simplified implementation

keywords = {

"ethics": ["ethics", "moral", "right", "wrong", "good", "bad", "virtue", "vice"],

"metaphysics": ["reality", "existence", "being", "ontology", "metaphysics"],

"epistemology": ["knowledge", "belief", "truth", "justification", "epistemology"],

"logic": ["logic", "reasoning", "argument", "fallacy", "valid", "sound"],

"aesthetics": ["beauty", "art", "aesthetic", "taste", "judgment"],

"political": ["politics", "justice", "rights", "freedom", "equality", "state"]

}

question\_lower = question.lower()

for area, terms in keywords.items():

if any(term in question\_lower for term in terms):

return area

return None

async def \_track\_model\_usage(self, user\_id, model\_name, tone\_id):

"""Track model usage for analytics and personalization"""

await self.db.model\_usage\_stats.update\_one(

{"user\_id": user\_id, "model": model\_name, "tone\_id": tone\_id},

{"$inc": {"usage\_count": 1}, "$set": {"last\_used": datetime.now()}},

upsert=True

)

def \_apply\_customizations(self, tone\_prompt, customizations):

"""Apply user customizations to tone prompt"""

modified\_prompt = tone\_prompt

if 'depth' in customizations:

depth\_instructions = {

'concise': "Provide concise, focused responses.",

'balanced': "Balance depth and brevity in responses.",

'comprehensive': "Provide comprehensive, in-depth responses."

}

modified\_prompt += f"\n\n{depth\_instructions.get(customizations['depth'], '')}"

if 'complexity' in customizations:

complexity\_instructions = {

'beginner': "Use accessible language suitable for philosophical beginners.",

'intermediate': "Use moderately complex philosophical language.",

'advanced': "Use sophisticated philosophical language and concepts."

}

modified\_prompt += f"\n\n{complexity\_instructions.get(customizations['complexity'], '')}"

if 'citations' in customizations and customizations['citations']:

modified\_prompt += "\n\nInclude references to relevant philosophical works and thinkers."

return modified\_prompt

## Premium AI Customization Options for Tier 3

### Response Customization Controls

#### Philosophical Depth Adjustment

* **Implementation**: Three-level slider in Ask interface
* **Options**:
  + Concise: Brief, focused responses
  + Balanced: Moderate depth (default)
  + Comprehensive: In-depth, thorough exploration
* **Technical Approach**: Modifies prompt instructions and token limits
* **Use Case**: Allows users to tailor response length to their current needs

#### Complexity Level Control

* **Implementation**: Three-level selector in Ask interface
* **Options**:
  + Beginner: Accessible language, fundamental concepts
  + Intermediate: Moderate complexity (default)
  + Advanced: Sophisticated philosophical language and concepts
* **Technical Approach**: Adjusts prompt instructions for language complexity
* **Use Case**: Adapts content to user's philosophical background

#### Citation Preferences

* **Implementation**: Toggle switch in Ask interface
* **Options**:
  + Include citations to philosophical works
  + Omit citations for cleaner responses
* **Technical Approach**: Adds citation instructions to prompt
* **Use Case**: Supports academic or learning-focused exploration

#### Conversation Memory Depth

* **Implementation**: Slider in conversation settings
* **Options**:
  + Recent (last 1-2 exchanges)
  + Extended (last 3-5 exchanges)
  + Complete (entire conversation)
* **Technical Approach**: Adjusts context window utilization
* **Use Case**: Balances coherence with exploration of new ideas

### Advanced AI Features for Tier 3

#### Philosophical Tone Blending

* **Implementation**: Experimental feature allowing combination of two tones
* **Interface**: Dual tone selector with blend ratio slider
* **Technical Approach**: Interpolates prompt instructions from two tones
* **Use Case**: Creates unique philosophical perspectives

#### Custom Tone Creation (Beta)

* **Implementation**: Guided wizard for creating personal philosophical tones
* **Components**:
  + Tone naming and description
  + Philosophical influence selection
  + Communication style preferences
  + Example response curation
* **Technical Approach**: Generates custom prompt template from user inputs
* **Use Case**: Allows for highly personalized philosophical exploration

#### Comparative Analysis

* **Implementation**: Option to receive multiple AI perspectives on same question
* **Interface**: "Compare Perspectives" button after receiving initial response
* **Technical Approach**: Routes same question to different models/tones
* **Use Case**: Explores philosophical diversity and contrasting viewpoints

#### Socratic Dialogue Mode

* **Implementation**: Extended conversation mode with Socratic questioning
* **Interface**: "Socratic Exploration" toggle in conversation settings
* **Technical Approach**: Modifies AI instructions to use Socratic method
* **Use Case**: Deeper philosophical exploration through guided questioning

## User Interface Design

### Model Selection UI

#### Primary Selection Interface

* **Location**: Below tone selection in Ask screen
* **Default State**: Collapsed to "AI Router Choice" (automatic)
* **Expanded State**: Horizontal scrollable cards for each model
* **Visual Design**:
  + Distinctive icon and color for each model
  + Brief strength description visible at a glance
  + Currently selected model clearly highlighted
* **Accessibility**: Full keyboard navigation and screen reader support

#### Model Information Cards

* **Content Per Model**:
  + Name and developer
  + Key philosophical strengths
  + Optimal use cases
  + Sample response style
* **Interaction**: Tap model card to select, tap info icon for detailed information
* **Visual Design**: Clean cards with subtle branding elements from each model

### Customization Controls UI

#### Settings Panel

* **Access**: "Customize" button near model selection
* **Layout**: Collapsible sections for different customization categories
* **State Persistence**: Remembers user's preferred settings
* **Visual Design**: Slider and toggle controls with philosophical aesthetic

#### Preset Configurations

* **Implementation**: Quick-select presets for common use cases
* **Options**:
  + Academic Research (citations on, comprehensive, advanced)
  + Casual Exploration (citations off, balanced, intermediate)
  + Learning Mode (citations on, comprehensive, beginner)
* **Visual Design**: Preset chips with descriptive icons

### Response Attribution

#### Model Attribution

* **Implementation**: Subtle footer on AI responses
* **Content**: "Response by [Model Name]" with model icon
* **Interaction**: Tappable to view model details
* **Visual Design**: Non-intrusive but clearly visible attribution

#### Customization Indicator

* **Implementation**: Small icon showing active customizations
* **Interaction**: Tappable to view applied customization settings
* **Visual Design**: Unobtrusive indicator that doesn't distract from content

## Technical Considerations

### Performance Optimization

#### Response Time Management

* **Challenge**: Different models have varying response times
* **Solution**:
  + Implement adaptive loading indicators
  + Use streaming responses where supported
  + Pre-warm connections to all models
* **User Experience**: Set appropriate expectations for each model

#### Quota Management

* **Challenge**: Premium models have higher API costs
* **Solution**:
  + Implement soft usage limits even for Tier 3
  + Transparent usage dashboard
  + Predictive notifications before limits
* **User Experience**: Avoid unexpected cutoffs while maintaining sustainability

#### Fallback Mechanisms

* **Challenge**: Model availability and errors
* **Solution**:
  + Robust error handling
  + Graceful fallback to alternative models
  + Clear communication about fallbacks
* **User Experience**: Maintain reliability even when preferred model is unavailable

### Data Privacy and Security

#### Model-Specific Privacy

* **Challenge**: Different models have varying privacy policies
* **Solution**:
  + Clear privacy indicators for each model
  + Option to exclude sensitive questions from specific models
  + Transparent data retention policies
* **User Experience**: Informed choice about privacy implications

#### Secure Credential Management

* **Challenge**: Managing multiple API credentials securely
* **Solution**:
  + Encrypted credential storage
  + Rotating API keys
  + Minimal permission scopes
* **Technical Implementation**: Secure vault for API credentials

### Analytics and Improvement

#### Model Performance Tracking

* **Implementation**: Track quality metrics by model
  + User ratings via ContemplativeOrb
  + Response time
  + Usage patterns
* **Application**: Continuous improvement of routing algorithm
* **User Benefit**: Increasingly accurate model recommendations

#### Personalization Engine

* **Implementation**: Learn from user preferences and feedback
  + Track which models users manually select for which topics
  + Analyze rating patterns by model and topic
* **Application**: Increasingly personalized default selections
* **User Benefit**: More relevant automatic model selection

## Implementation Roadmap

### Phase 1: Core Model Selection (Weeks 1-2)

* Implement basic model selection UI
* Extend AI Router to support user model selection
* Add model information cards
* Implement usage tracking and analytics

### Phase 2: Basic Customization (Weeks 3-4)

* Add depth and complexity controls
* Implement citation preferences
* Create customization presets
* Develop fallback mechanisms

### Phase 3: Advanced Features (Weeks 5-6)

* Implement conversation memory controls
* Add Socratic dialogue mode
* Develop comparative analysis feature
* Create personalization engine

### Phase 4: Custom Tones (Weeks 7-8)

* Implement tone blending
* Develop custom tone creation wizard
* Add tone management interface
* Create tone sharing capabilities (future)

## User Education and Onboarding

### Tier 3 Feature Introduction

* **Implementation**: Interactive tutorial upon upgrading to Tier 3
* **Content**:
  + Model selection guide
  + Customization options overview
  + Use case recommendations
  + Best practices for optimal results
* **Format**: Interactive walkthrough with practice opportunities

### Contextual Guidance

* **Implementation**: Subtle tooltips and hints in UI
* **Trigger**: First use of features or detected confusion
* **Content**: Just-in-time guidance on feature usage
* **Format**: Unobtrusive hints that don't interrupt workflow

### Advanced Techniques Documentation

* **Implementation**: In-app "Sage Techniques" guide
* **Content**:
  + Model selection strategies
  + Customization combinations for different goals
  + Advanced philosophical exploration techniques
* **Format**: Searchable knowledge base with examples

## Value Communication

### Tier 3 Value Reinforcement

* **Implementation**: "Sage Benefits" section in profile
* **Content**:
  + Usage statistics across models
  + Customization usage
  + Exclusive feature utilization
* **Format**: Visual dashboard with insights

### Comparative Examples

* **Implementation**: "Why Sage Matters" showcase
* **Content**: Side-by-side comparisons of:
  + Basic vs. premium model responses
  + Standard vs. customized interactions
  + Simple vs. advanced philosophical explorations
* **Format**: Interactive examples with clear value demonstration