AI Sutra - Project Documentation

Executive Summary

AI Sutra is a dual-modal content platform that combines intelligent feed curation with progressive learning experiences. The platform leverages Claude AI to deliver personalized content streams and structured educational curricula, transforming how users consume information and acquire new skills.

Project Name: AI Sutra

Rationale behind the Name

"Sutra" (Sanskrit: सूत्र) translates to "thread" or "string of knowledge" - ancient Indian texts that distilled complex wisdom into concise, memorable verses. Just as classical sutras were compact carriers of profound knowledge, **AI Sutra** weaves together personalized information streams and learning paths tailored to each user's needs.

The name embodies:

- **Thread of Knowledge**: Continuous, connected learning experiences
- **Condensed Wisdom**: Curated, relevant content filtered from vast information streams
- **Personalized Guidance**: AI-driven customization similar to how sutras were interpreted for different students
- **Progressive Learning**: Building knowledge systematically, thread by thread

---

Purpose & Objectives

Core Purpose

AI Sutra addresses the modern challenge of **information overload** by providing intelligent, context-aware content delivery and structured learning experiences.

Key Objectives

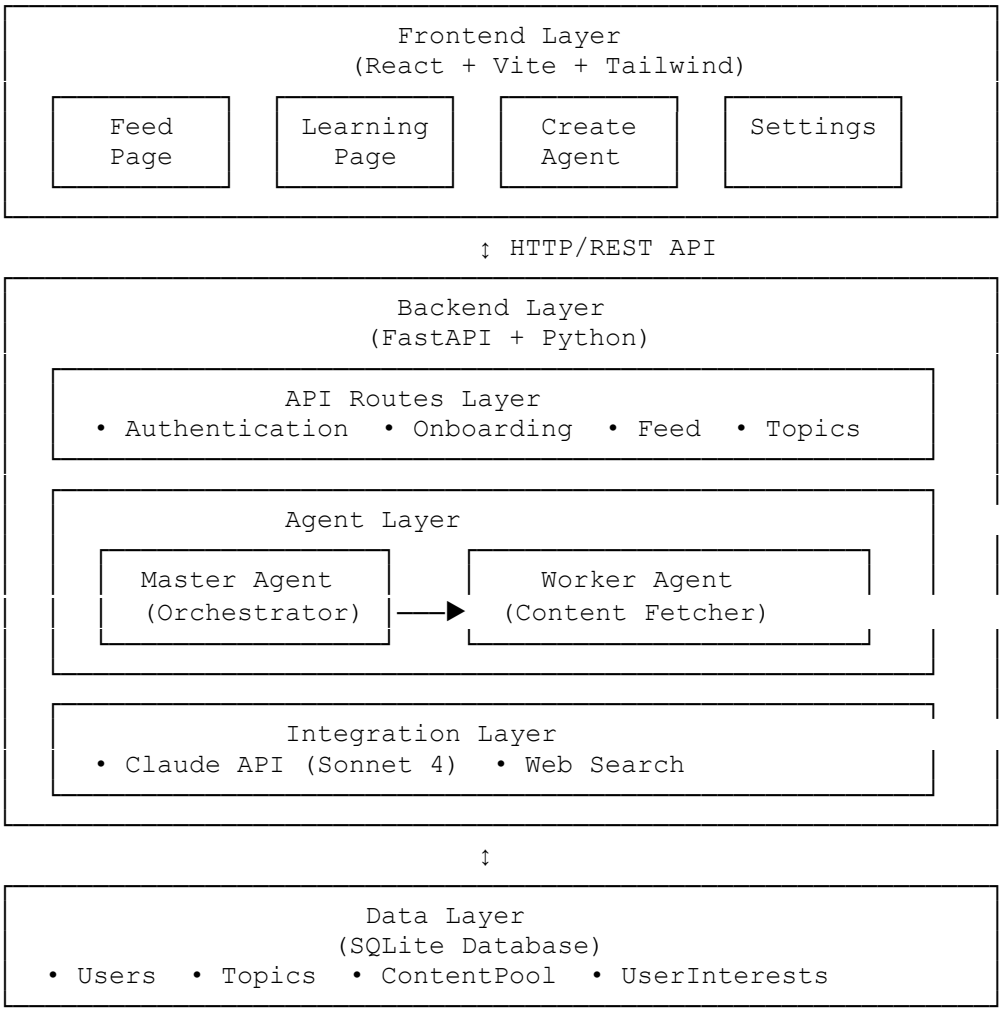
1. **Ease Information Consumption**
    - Filter relevant content from overwhelming internet noise
    - Deliver timely, personalized news and updates
    - Organize content by topics users care about
    - Provide both real-time feeds and AI-synthesized insights
2. **Create Personalized Training Modules**
    - Generate custom learning curricula based on user goals
    - Structure knowledge progression over defined time periods

    - Build context-aware lessons that adapt to previous learning
    - Track progress through multi-day learning journeys
3. **Dual-Modal Content Delivery**
    - **Feed Mode**: Real-time content aggregation for staying informed
    - **Learning Mode**: Structured, progressive educational content
    - Support both internet-sourced articles and AI-generated content
4. **Intelligent Automation**
    - Scheduled content fetching based on user preferences
    - Automated curriculum generation with day-by-day progression
    - Context-aware content that builds on previous interactions

---

System Architecture

High-Level Architecture



Component Roles

Frontend Components

1. Feed Page

- **Purpose**: Display real-time content feeds for user topics
- **Features**:
  - Topic sidebar for navigation
  - Content cards with summaries and source links
  - "Fetch Now" button for manual content refresh
  - Bookmark/save functionality
- **Content Types**: Internet articles (with URLs) or AI-generated insights

2. Learning Page

- **Purpose**: Display structured learning curricula
- **Features**:
  - Learning topic sidebar with progress tracking
  - Full lesson content display with theory, examples, exercises
  - Day-by-day progression (Day X of Y)
  - Progress bar visualization
  - Completion status tracking
- **Content Type**: AI-generated educational content only

3. Create Agent Page

- **Purpose**: Create and manage Feed and Learning agents
- **Features**:
  - Dual creation modes (Feed Agent / Learning Agent)
  - Topic configuration (name, details, language, schedule)
  - Feed Source selection (Internet / AI) for Feed agents
  - Learning Period specification for Learning agents
  - Edit and delete functionality

4. Settings Page

- **Purpose**: User preferences and account management
- **Features**: (Future implementation)

Backend Components

1. Master Agent (Orchestrator)

Location: `backend/app/agents/master_agent.py`

Role: Handles user onboarding and topic creation

Responsibilities:

- Parse natural language interest input
- Extract structured topics using Claude API
- Create topic records in database
- Link topics to users
- Generate default agent configurations

Example Flow:

```
User Input: "I want Python news and cricket updates"
    ↓
Master Agent → Claude API → Parse interests
    ↓
Create Topics: ["Python News", "Cricket Updates"]
    ↓
Link to User → Store in Database
```

---

2. Worker Agent (Content Fetcher)

Location: `backend/app/agents/worker_agent.py`

Role: Fetches and generates content for topics

Responsibilities:

- Route to appropriate content method based on topic type
- Fetch internet articles using Claude web search
- Generate AI content for analysis/insights
- Generate progressive learning curricula
- Store content in ContentPool
- Track learning progress and completion

Content Modes:

A) Internet Content (Feed)

- Uses Claude web search to find articles
- Returns 15 articles with URLs, titles, summaries
- Best for: News, Sports, Politics, Trending Topics

B) AI-Generated Content (Feed)

- Claude generates unified, comprehensive responses
- No URLs, single synthesized content piece
- Best for: Astrology, Market Analysis, Personalized Advice

C) Learning Content (Learning)

- Generates structured day-by-day lessons
- Progressive curriculum with theory, examples, exercises
- Context-aware (builds on previous lessons)
- Tracks current_day and marks completion

Example Flow:

```
Scheduled Trigger → Worker Agent
    ↓
Check Topic Type & Feed Source
    ↓
    ├─→ Internet: Fetch articles via Claude web search
    ├─→ AI: Generate synthesized content
    └─→ Learning: Generate Day N lesson (context-aware)
    ↓
Store in ContentPool → Update last_fetched
```

---

3. Claude Client

Location: `backend/app/utils/claude_client.py`

Role: Interface with Claude API for AI capabilities

Methods:

- `parse_interests()`: Extract topics from natural language
- `fetch_content_for_topic()`: Web search for articles
- `generate_ai_content()`: Create AI-synthesized content
- `generate_learning_content()`: Create structured lessons

API Usage:

- Model: Claude Sonnet 4 (`claude-sonnet-4-20250514`)
- Features: Web search, content generation, structured outputs

---

4. API Routes

Authentication Routes (`/api/auth`)

- User login and registration
- Username/password authentication

Onboarding Routes (`/api/onboarding`)

- Process user interests → Create topics
- Get user's topics

Feed Routes (`/api/feed`)

- Get user feed (all topics' content)
- Refresh all topics for user
- Refresh specific topic

Topic Routes (`/api/topics`)

- Update topic details
- Delete topic (cascade delete content)

Database Schema

Core Tables

1. Users

- Authentication credentials
- Profile information
- Created timestamp

2. Topics

```
- id, topic_name, description
- feed_source: "internet" | "ai"
- topic_type: "feed" | "learning"
- learning_period_days: Total days for learning plan
- current_day: Progress tracker (1, 2, 3...)
- is_completed: Boolean for learning completion
- agent_config: JSON configuration
- last_fetched: Timestamp
```

3. ContentPool

```
- id, topic_id (FK)
- title, summary, content
- url (nullable for AI content)
- source, fetched_at
```

4. **user_topics** (Junction Table)

- Many-to-many relationship
- Users ↔ Topics

5. UserInterests

- Raw natural language inputs
- Historical record of user preferences

Key Features

1. Dual Agent System

Feed Agents

- Real-time content aggregation
- 15 articles per fetch
- Two modes: Internet (articles with URLs) or AI (synthesized insights)
- Scheduled or manual fetching
- 7-day retention period

Learning Agents

- Structured curricula (e.g., 7-day, 30-day plans)
- Progressive day-by-day lessons
- Context-aware content generation
- Automatic completion tracking
- Always uses AI generation

2. Intelligent Content Generation

Context-Aware Learning

- Each lesson builds on previous days
- AI receives summary of last 3 lessons
- Progressive difficulty and complexity
- Comprehensive structure: objectives, theory, examples, exercises

Time-Aware Feed Content

- Considers schedule frequency (daily/weekly/monthly)
- Includes current date context for AI generation
- Relevant to user's specified time period

3. User Experience

Feed UI

- Topic sidebar navigation
- Content cards with summaries
- "Read more" links for internet sources
- "AI Generated" label for AI content
- Bookmark/save functionality

- Bigger cards for better readability

Learning UI

- Progress tracking (Day X of Y)
- Visual progress bar
- Full lesson display (not truncated)
- Completion badges
- All content remains accessible after completion

Technology Stack

Frontend

- **Framework**: React 18
- **Build Tool**: Vite
- **Styling**: Tailwind CSS
- **Icons**: Heroicons
- **HTTP Client**: Axios
- **Routing**: React Router DOM

Backend

- **Framework**: FastAPI (Python)
- **Database**: SQLite with SQLAlchemy ORM
- **AI Integration**: Anthropic Claude API (Sonnet 4)
- **Async Support**: asyncio
- **Scheduling**: APScheduler

AI Services

- **Model**: Claude Sonnet 4 (`claude-sonnet-4-20250514`)
- **Capabilities**: Web search, content generation, structured outputs

---

Data Flow Examples

Example 1: Creating a Feed Agent

```
User → Create Agent Page
    ↓
Fills Form: "Cricket News", "Internet" source, "Daily" schedule
    ↓
Frontend → POST /api/onboarding
    ↓
Master Agent → Parse input → Create Topic
    ↓
Topic stored with: topic_type="feed", feed_source="internet"
    ↓
Response → Frontend → Show success
```

Example 2: Creating a Learning Agent

```
User → Create Agent Page
    ↓
Fills Form: "Python Basics", Learning period: 7 days
    ↓
Frontend → POST /api/onboarding
    ↓
Master Agent → Create Topic
    ↓
Topic stored with: topic_type="learning", learning_period_days=7,
current_day=1
    ↓
Response → Frontend → Show success
```

Example 3: Fetching Learning Content (Day 1)

```
User clicks "Fetch Now" on Learning Page
    ↓
Frontend → POST /api/feed/refresh/{user_id}/topic/{topic_id}
    ↓
Worker Agent → Check: current_day=1, total_days=7
    ↓
Worker Agent → Claude API:
    - Generate Day 1 lesson
    - Include: objectives, theory, examples, exercises
    ↓
Store in ContentPool:
    - title: "Day 1: Python Basics"
    - source: "Learning Day 1/7"
    - content: Full structured lesson
    ↓
Update Topic: current_day=2
    ↓
Response → Frontend → Display lesson
```

Example 4: Fetching Learning Content (Day 7)

```
User clicks "Fetch Now" on Day 7
    ↓
Worker Agent → Check: current_day=7, total_days=7
    ↓
Worker Agent → Get previous 3 days context
    ↓
Claude API: Generate Day 7 lesson (building on Days 4-6)
    ↓
Store lesson in ContentPool
    ↓
Update Topic: current_day=8, is_completed=True
    ↓
Response → Frontend → Display lesson + "Completed" badge
```

Future Enhancements

Planned Features

1. **Feed Enhancements**
    - Time-grouped collapsible sections (by fetch time)
    - Advanced filtering and search
    - Content recommendations
2. **Learning Enhancements**
    - Quiz and assessment integration
    - Certificate generation
    - Download curriculum as PDF
    - Restart completed plans
3. **User Experience**
    - Mobile application
    - Dark mode
    - Notification system
    - Social sharing
4. **Analytics**
    - Learning progress dashboards
    - Reading time tracking
    - Topic engagement metrics

Conclusion

AI Sutra represents a paradigm shift in how users consume information and learn new skills. By combining intelligent content curation with structured learning paths, the platform transforms overwhelming information streams into personalized, actionable knowledge threads - staying true to the ancient concept of sutras as condensed, progressive wisdom.

The dual-modal architecture ensures users can both stay informed (Feed) and systematically acquire new skills (Learning), all powered by Claude AI's advanced capabilities in web search, content generation, and context-aware learning.

Document Version: 1.0
Last Updated: February 15, 2026
Project Repository: AI Sutra
Technology: React + FastAPI + Claude Sonnet 4