Customer FAQ System - Homework Project

Project Overview

Build a customer FAQ system that integrates AI with knowledge base support and ticket management functionality. The system should provide a chat-based UI for users to interact with an AI agent, with fallback task creation when the AI cannot provide adequate responses.
Note: Feel free to use AI coding tools (such as Cursor, etc) for this assignment.

Core Requirements

1. Knowledge Base Integration

- Implement a knowledge base system that can be accessible by AI
- AI should attempt to answer user questions using the knowledge base
- Knowledge base can contain FAQs, product information, etc, **expected to be small text file** (<500k bytes), so database may not be required
- You can find a few pages (say 10 URLs) and save to a text file as knowledge base

2. AI Chat Interface

- Create a chat-based UI similar to existing customer service chat products
- AI responds based on knowledge base content
- Chat history should be maintained during the session
- Any LLM is totally fine (such as ChatGPT API, or local solution), **the criteria is the capability of LLM integration**, not the LLM quality itself.

3. Ticket Generation

- When AI cannot provide a satisfactory answer, automatically generate a ticket
- Tasks should be stored in the database with relevant information (user question, timestamp, user contact, etc.)
- Basic page to display generated tickets in DB, no need fancy UI

Technical Stack Suggestions

- Frontend: Vue.js or React
- Backend: Python FastAPI or Go
- Database: Any relational DB (PostgreSQL, MySQL) or key-value store (Redis, etc.)

Note: These are suggestions only. Use any technology stack you're already familiar with.

Testing Requirements

- End-to-end testing for core business logic
- Test the complete flow: user question → AI response → task generation (if needed)
- Unit tests for core functions are enough, no need high coverage
- Simulation testing if more relevant than traditional testing

Deliverables

Code Submission

- Check in complete code to GitHub repository
- Repository access to be shared with reviewer gensparkreviewer@gmail.com
- README.md with setup instructions in local run (recommended), no need cloud deploy
- Screenshots or screen recording (optional but appreciated)

Extra Bonus

- Feel free to add any interesting features which you think would help in the real scenario.

Review Process

Review Meeting Preparation

- Prepare bullet points explaining your technical decisions and architecture choices
- Be ready to demonstrate the application's core functionality, and testing results
- No comprehensive documentation required - focus on key decision rationale

Evaluation Criteria

- Functionality: All core features working as specified
- Testing: Adequate test coverage of critical functionality
- Technical Decisions: Ability to explain implementation choices and code