Social IQ Chatbot - Project Documentation

Introduction

The **Social IQ Chatbot** is an intelligent conversational system designed to assist users with queries related to the Social IQ application. Built using advanced AI technologies, this chatbot leverages **Retrieval-Augmented Generation (RAG)** architecture to provide contextually relevant responses by combining document-based knowledge with conversational AI capabilities.

The system processes PDF documents to create a searchable knowledge base and maintains conversation history for personalized user interactions. It serves as a comprehensive support system that can answer questions about system functionality, technology stack, and provide contextual assistance based on uploaded documentation.

Project Objectives

- Provide intelligent, context-aware responses to user queries
- Maintain conversation history for personalized experiences
- Process and utilize PDF documentation for knowledge retrieval
- Offer real-time chat functionality with quick suggestion features
- Create a scalable, maintainable chatbot infrastructure

Technology Stack

Backend Technologies

- **Python 3.x** - Core programming language
- **Flask** - Lightweight web framework for API development
- **Flask-CORS** - Cross-Origin Resource Sharing support

AI & Machine Learning

- **Google Generative AI (Gemini)** - Primary LLM for response generation
    - Gemini-1.5-Flash (Primary model)
    - Gemini-2.0-Flash (Fallback model)
- **LangChain** - Framework for building LLM applications
- **GoogleGenerativeAIEmbeddings** - Text embedding generation

Vector Database & Search

- **FAISS (Facebook AI Similarity Search)** - Vector database for similarity search
- **RecursiveCharacterTextSplitter** - Document chunking for optimal processing

Database & Storage

- **MongoDB** - NoSQL database for conversation storage

- **PyMongo** - MongoDB driver for Python

Document Processing

- **PyPDFDirectoryLoader** - PDF document loading and processing
- **LangChain Document Loaders** - Unified document handling

Development & Deployment

- **python-dotenv** - Environment variable management
- **Threading** - Concurrent processing support
- **Webbrowser** - Automated browser launching

Frontend Integration

- **Flutter** - Mobile/web frontend framework
- **RESTful APIs** - Communication between frontend and backend

Key Features

1. Intelligent Document Processing

- Automatic PDF loading from designated directories
- Intelligent text chunking with overlap for context preservation
- Vector embedding generation for semantic search capabilities

2. Advanced RAG Architecture

- Context retrieval from processed documents using similarity search
- Contextual response generation combining retrieved knowledge with AI
- Real-time document querying for up-to-date information

3. Conversation Management

- Persistent conversation history storage in MongoDB
- Email-based user identification and session management
- Last 5 conversations context for continuity

4. Quick Suggestions System

- AI-generated follow-up questions based on responses
- Dynamic suggestion generation for improved user engagement
- Intelligent question formatting and cleaning

5. Robust Error Handling

- Fallback AI models for reliability
- Comprehensive exception handling

- Graceful degradation for service continuity

6. RESTful API Architecture

- Clean API endpoints for chat functionality
- Health check monitoring
- Chat history retrieval capabilities

7. Real-time Response Generation

- Timestamp tracking for all interactions
- Immediate response processing
- Concurrent request handling

Technology Justification

Why Python?

- **Rich AI/ML Ecosystem**: Extensive libraries for AI development (LangChain, FAISS, etc.)
- **Rapid Prototyping**: Quick development and testing capabilities
- **Community Support**: Large community and extensive documentation
- **Integration Friendly**: Easy integration with various APIs and services

Why Flask?

- **Lightweight**: Minimal overhead for API-focused applications
- **Flexibility**: High customization capability for specific needs
- **RESTful Design**: Natural fit for API-driven architectures
- **Development Speed**: Quick setup and deployment

Why Google Gemini API?

- **Advanced Capabilities**: State-of-the-art language understanding and generation
- **Cost-Effective**: Competitive pricing for high-quality AI responses
- **Reliability**: Stable API with good uptime and performance
- **Multi-modal Support**: Potential for future image/video processing

Why MongoDB?

- **Document-Based**: Natural fit for conversation data storage
- **Scalability**: Horizontal scaling capabilities for growing user base
- **Flexibility**: Schema-less design for evolving data structures
- **JSON Compatibility**: Seamless integration with Python and web APIs

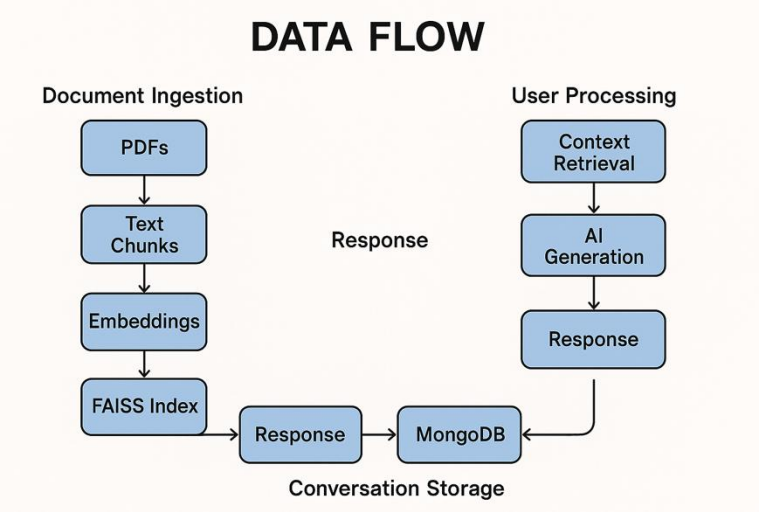
Why FAISS?

- **Performance**: Extremely fast similarity search even with large datasets
- **Memory Efficient**: Optimized memory usage for vector operations
- **Scalability**: Handles millions of vectors efficiently
- **Facebook Backing**: Robust, battle-tested technology

Why html Frontend?

- **Cross-Platform**: Single codebase for mobile and web applications
- **Performance**: Near-native performance on mobile devices
- **UI Consistency**: Uniform experience across platforms
- **Google Support**: Strong backing and continuous development

Data Flow:



Document Processing Pipeline

PDFs → PyPDFDirectoryLoader → RecursiveCharacterTextSplitter →

GoogleGenerativeAIEmbeddings → FAISS Index → Local Storage

Response Generation Process

1. **Context Retrieval**: Query similarity search against FAISS index
2. **History Integration**: Last 5 conversations for context continuity
3. **Prompt Construction**: System prompt + context + history + user query
4. **AI Generation**: Google Gemini API processing
5. **Suggestion Generation**: Follow-up questions using secondary AI call

Future Enhancements

Planned Features

1. **Multi-language Support**: Expand to support multiple languages
2. **Voice Integration**: Add voice input/output capabilities

Conclusion

The Social IQ Chatbot represents a sophisticated implementation of modern AI technologies, combining retrieval-augmented generation with robust conversation management. The system successfully addresses the need for intelligent, context-aware user assistance while maintaining scalability and performance.

Key Achievements

- **Advanced AI Integration**: Successfully implemented Google Gemini for high-quality responses
- **Robust Architecture**: Built scalable, maintainable system architecture
- **User Experience**: Created seamless, conversational user interface
- **Knowledge Management**: Effective PDF processing and retrieval system
- **Reliability**: Comprehensive error handling and fallback mechanisms