WebSocket-Based Chat Feedback System in Golang

Objective

Build a Golang WebSocket chatbot to handle customer conversations and feedback collection. The chatbot should store messages and feedback in a database. A CLI-based interface should be used for demonstration. Candidates are encouraged to leverage AI tools to accelerate development, improve chatbot logic, and automate workflows.

Requirements

1. WebSocket Chatbot API (Golang)

- Accepts customer messages and replies in real time.
- Triggers feedback requests when keywords like "feedback" or "review" are detected.
- Stores chats and feedback in a database.
- Uses AI-assisted tooling to improve efficiency in coding, testing, and documentation.

2. WebSocket API Endpoints

Endpoint	Description
`ws://chat`	WebSocket connection for real-time chat.
`GET /message/list`	Retrieves chat history for a customer.

3. Database Schema (SQLite/ Postgres/ Mysql / In-Memory DB)

Tables:

- Customers → Stores customer details.
- Messages → Stores chat messages.
- Feedback → Saves ratings and comments.

Example:

customer_id	message	sender	timestamp
123	"I want to leave a review"	customer	2025-03-12 12:00:00
123	"Please rate your experience from 1-5"	bot	2025-03-12 12:00:01

CLI Demonstration

- Send messages via CLI and receive bot responses.
- Trigger and store feedback.
- Retrieve conversation history.

Bonus Features

- Sentiment Analysis → Classify feedback as positive, neutral, or negative using AI.
- Workflow Automation → AI-powered chatbot logic to enhance responses.
- Config-Based Chat Logic → Store chatbot rules in JSON/YAML files for easy updates.

Submission Requirements

- Golang WebSocket chatbot with database integration.
- CLI-based interaction demo instead of a front-end.
- Clear documentation on setup and usage.
- Estimated time: 3-4 hours.

After submission, a live review session will be scheduled to discuss your design and extend functionality.