Manual de Usuario

ChatBot en Java Swing usando la API de Ollama

Estudiante:
Ivan Peña

Código Estudiantil:
200212253

Manual de Usuario del ChatBot en Java Swing usando la API de Ollama

Índice

---

1. Introducción

Este manual describe detalladamente cómo instalar, usar y colaborar en el desarrollo del proyecto ChatBot en Java Swing, con conexión a la API de Ollama. Incluye instrucciones paso a paso, explicaciones del funcionamiento del código y una guía para resolver errores comunes.

---

2. Instalación del Programa

Requisitos Previos

Antes de instalar el programa, asegúrate de tener lo siguiente:

- **Java Development Kit (JDK)** versión 8 o superior.
- **Apache Maven** (opcional, para gestionar dependencias).
- **IDE recomendado:** NetBeans, Eclipse o IntelliJ IDEA.
- **API de Ollama:** Ejecutándose en http://localhost:11434.

Pasos para la Instalación

1. **Clonar el repositorio desde GitHub:**
    - Abre una terminal y ejecuta el comando:
    - git clone https://github.com/IvMilo/Lab2IvanCamilo-AyG---II
    - Esto descargará los archivos necesarios.
2. **Abrir el proyecto en tu IDE:**
    - NetBeans: Selecciona "Abrir Proyecto" y navega hasta la carpeta descargada.
    - Eclipse o IntelliJ IDEA: Importa el proyecto como un proyecto estándar de Java o Maven.
3. **Ejecutar la aplicación:**
    - Corre la clase principal ChatUi. (también puedes correr el main).

---

3. Cómo Usar el ChatBot

Descripción de la Interfaz

La aplicación cuenta con las siguientes secciones:

1. **Campo de Entrada de Texto:** Permite al usuario escribir preguntas para enviarlas al chatbot.
2. **Botón "Enviar":** Envía la pregunta al chatbot y muestra la respuesta.
3. **Área de Conversación:** Muestra todas las preguntas y respuestas de la conversación actual.
4. **Botón "NewChat":** Inicia una nueva conversación y guarda la actual en el historial.
5. **Área de Historial:** Lista donde se muestran las conversaciones previas.
6. **Botón "Limpiar":** Borra todo el historial almacenado.

Flujo de Uso

Enviar una Pregunta

1. Escribe tu pregunta en el campo de texto.
2. Haz clic en el botón "Enviar".
3. La respuesta aparecerá en el área de conversación.

Consultar el Historial

1. Haz clic en una conversación del historial.
2. La conversación se cargará para que puedas revisarla o continuarla.

Iniciar un Nuevo Chat

1. Haz clic en el botón "NewChat".
2. Esto limpiará la conversación actual y la guardará automáticamente en el historial.

Limpiar el Historial

1. Haz clic en el botón "Limpiar".
2. Todo el historial se borrará de la interfaz.

4. Cómo Funciona el Código

Campo de Entrada y Botón "Enviar"

- **Campo de Texto:** Captura la pregunta ingresada:
- String pregunta = Pregunta.getText().trim();
- **Botón "Enviar":** Envía la pregunta a la API y muestra la respuesta:
- private void EnviarMouseClicked(java.awt.event.MouseEvent evt) {
- enviarPregunta();
- }

Manejo de Conversaciones

- Las conversaciones se almacenan en ArrayList<String>:
- private final ArrayList<String> conversacionActual = new ArrayList<>();
- Los mensajes se agregan dinámicamente al área de conversación:
- private void agregarConversacion(String texto) {
- conversacionActual.add(texto);
- modeloConversacion.addElement(texto);
- }

Manejo del Historial

- **Almacenar conversaciones:**
- private final ArrayList<String> historialConversaciones = new ArrayList<>();
- **Cargar conversaciones previas:**
- private void mostrarConversacionHistorial() {
- int indiceSeleccionado = Historial.getSelectedIndex();
- if (indiceSeleccionado != -1) {
- guardarCambiosConversacionActual();
- limpiarConversacion();
- indiceConversacionActual = indiceSeleccionado;

- String conversacionSeleccionada = historialConversaciones.get(indiceSeleccionado);
- DefaultListModel<String> modeloConversacion = (DefaultListModel<String>) Conversacion.getModel();
- modeloConversacion.clear();
-
- // Dividir la conversación seleccionada y mostrarla en la interfaz
- modeloConversacion.clear();
- conversacionActual.clear();
- String[] lineasConversacion = conversacionSeleccionada.split("\n");
- for (String linea : lineasConversacion) {
- modeloConversacion.addElement(linea);
- conversacionActual.add(linea);
- }
- Conversacion.setModel(modeloConversacion);
- }
- **Borrar el historial:**
- private void LimpiarMouseClicked(java.awt.event.MouseEvent evt) {
- historialConversaciones.clear();
- }

---

5. Resolución de Errores Comunes

Error de Conexión con la API

- **Problema:** La aplicación no puede conectarse a la API.
- **Solución:**
    - Verifica que la API esté ejecutándose en http://localhost:11434.
    - Asegúrate de tener conexión a Internet.

Mensajes No Guardados en el Historial

- **Problema:** Los mensajes de una conversación no se almacenan en el historial.
- **Solución:**
    - Usa el botón "NewChat" para guardar automáticamente la conversación actual antes de iniciar una nueva.

Mensajes Duplicados

- **Problema:** Los mensajes aparecen duplicados en el área de conversación.
- **Solución:**
    - Asegúrate de que no estés enviando mensajes repetidamente.

---

6. Guía para Colaborar en el Proyecto Usando GitHub

Pasos para Contribuir

1. **Clonar el repositorio:**
2. git clone https://github.com/IvMilo/Lab2IvanCamilo-AyG---II
3. **Crear una nueva rama:**
4. git checkout -b nueva-funcionalidad
5. **Realizar cambios y hacer commits:**
6. git add .
7. git commit -m "Descripción de los cambios realizados"
8. **Subir la rama a GitHub:**

9. git push origin nueva-funcionalidad
10. **Abrir un Pull Request:**
    - Ve al repositorio en GitHub y selecciona tu rama.
    - Haz clic en "Pull Request" y describe los cambios realizados.

Resolución de Conflictos

1. Abre los archivos en conflicto en tu IDE.
2. Realiza los ajustes necesarios para resolver los conflictos.
3. Guarda los cambios y crea un nuevo commit:
4. git add .
5. git commit -m "Conflictos resueltos"
6. Sube los cambios y actualiza el Pull Request.
