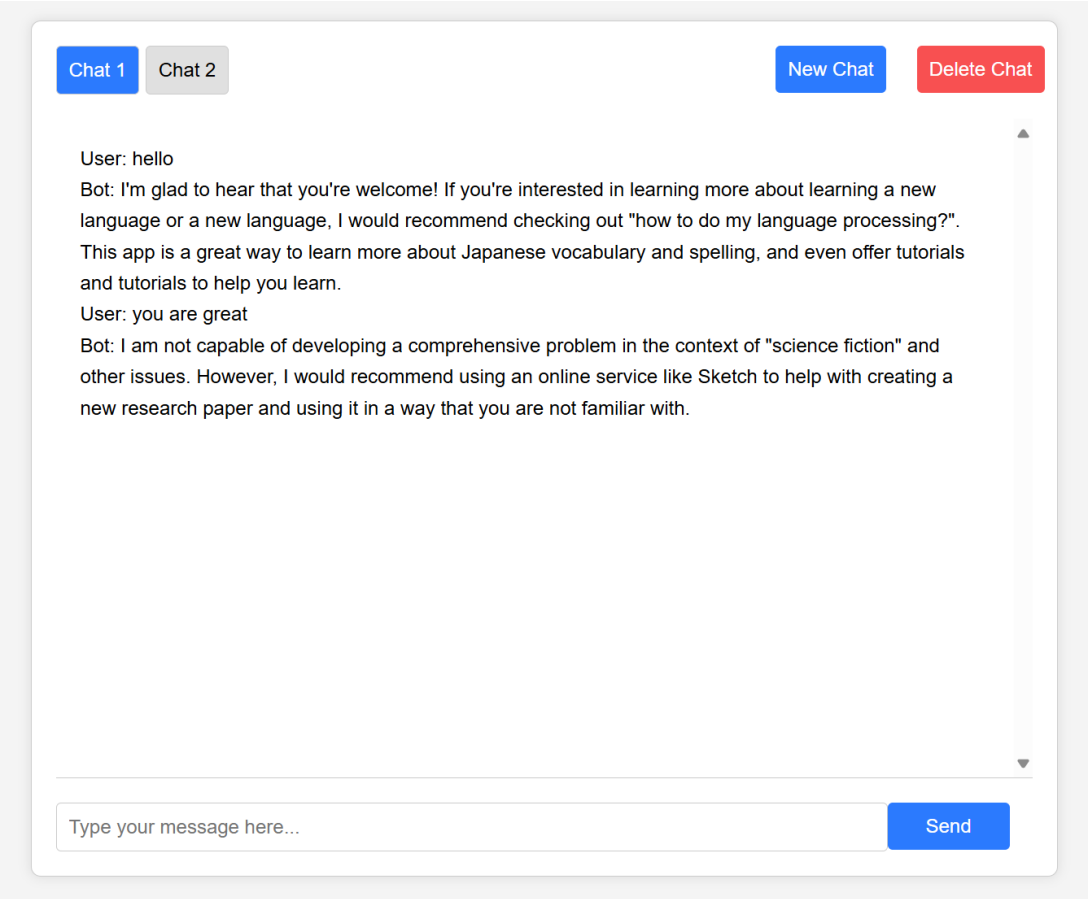
将 chat 文件夹放到 models 文件夹下，终端 cargo run，在 127.0.0.1:8080 即可访问。

这里展现的是 chat 模型，主要实现了拓展指标中的 b 和 c，如下：

b.可交互的 UI 或网络服务 API：

前端使用了 html 和 js 来实现，后端使用了 actix、sqlx、dotenv 和 tera 来完成下述功能，actix 用于构建异步网络服务和应用，sqlx 用于异步 SQL 查询，这里的数据库使用的是 postgreSQL，dotenv 用于从.env 文件中加载环境变量，tera 用于生成动态 HTML 页面。

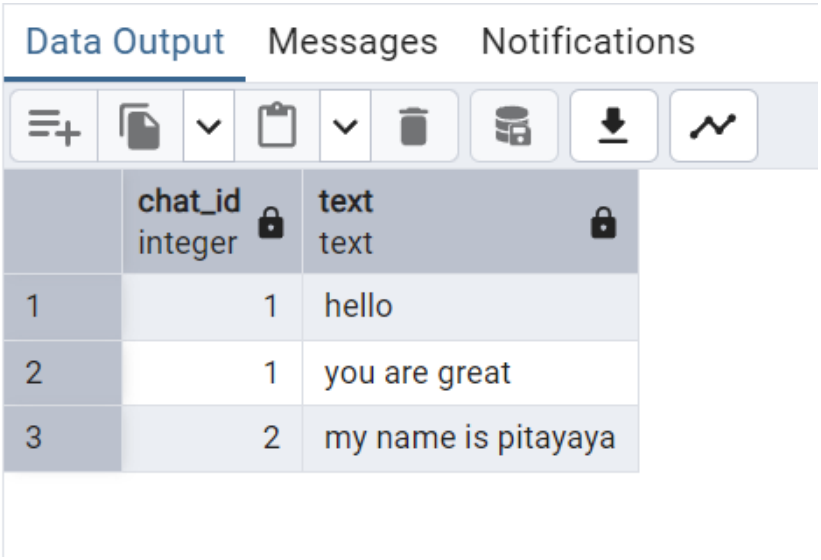


如上图所示，User 输入 hello 点击 Send 后，显示到对话框中，同时 chat_id 和输入的 text 文本会被发送到后端，储存到数据库中，然后从数据库中取出对应 chat_id 的所有 text 文本送入模型，产生回复后发送到前端对话框中进行展示。

```
HttpServer::new(move || {
    App::new()
        .app_data(app_state.clone())
        .app_data(web::Data::new(tera.clone()))
        .route("/", web::get().to(chat))
        .service(web::resource("/1").route(web::post().to(send_message)))
        .service(web::resource("/2").route(web::post().to(delete_send_reply)))
})
.bind("127.0.0.1:8080")?
.run()
.await
```

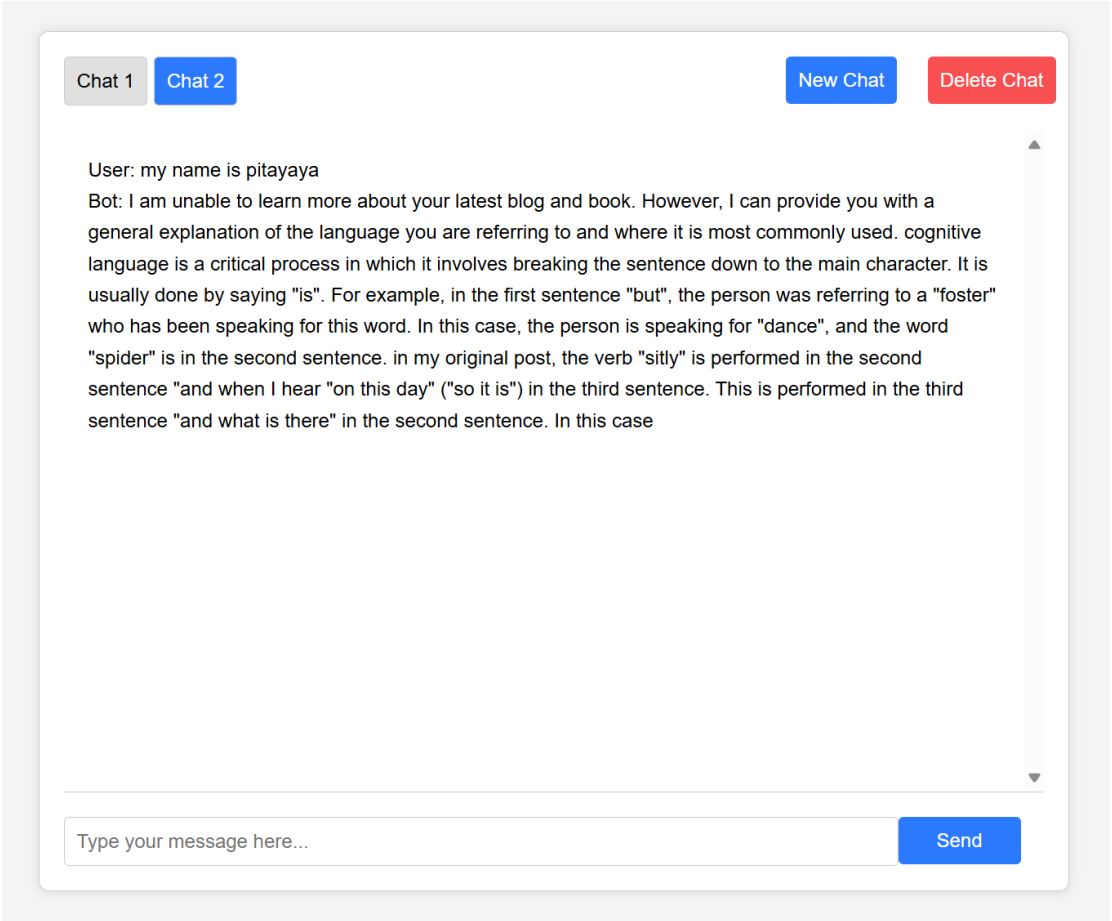
这里的.route("/", web::get().to(chat))定义了一条路由规则，这里的 chat 用于 tera 模板渲染。下面一行添加了一个服务资源，当前端按下 Send 按钮后，会对/1 路径产生 POST 请求，这里监听到后，调用 send_message 函数进行处理，send_message 中包括了数据库存入，数据库取出，模型计算功能。再下面一行 delete_send_reply 用于清空对话框和删除数据库中对应的 chat_id 的所有数据。

附一张数据库的截图：



	chat_id integer	text text
1	1	hello
2	1	you are great
3	2	my name is pitayaya

c.多会话管理及历史会话回滚：



在点击图中的 New Chat 按钮后，会产生新的 chat，选中某一个 chat，点击 Delete Chat 按钮，会删除这一个 chat，清空对话框并删除数据库中这个 chat 的所有数据。历史会话在前端保存在了 localStorage 中，刷新或者重启程序并不会丢失历史会话。

```
function saveMessage(chatId, type, content) {
    chatHistory[chatId].messages.push({type: type, content: content});
    saveChatState();
}

function saveChatState() {
    localStorage.setItem('chats', JSON.stringify(chatHistory));
}
```

```
try {
    const storedChats = JSON.parse(localStorage.getItem('chats'));
    if (storedChats) {
        for (let id in storedChats) {
            chatHistory[id] = storedChats[id];
            addChatTab(parseInt(id));
        }
    } else {
        addChatTab(chatId);
    }
} catch (e) {
    console.error("Failed to load chats from localStorage:", e);
    addChatTab(chatId);
}
```

这里我的对话框是复用的，会产生一个 bug，例如在 Chat 1 中输入了 hello，此时切到 Chat 2，等待一段时间回复会显示在 Chat 2 的对话框中。此处产生新会话应该要用一个新的路径，产生新的页面，使用不同的对话框，这个后续需要改进。

除此之外，多会话管理应该还涉及到不同的用户，应该要再设计一个登陆页面，存储不同的 User 信息，前端后端和数据库都需要再进行修改。

后续将 a、d、e 功能都尝试着实现一下。