FoodieLine Introduction

FoodieLine is an intelligent chatbot integrated with the **LINE Messaging API**, designed to provide personalized *restaurant recommendations* based on *user interactions* and *location data*. The tool leverages **Large Language Models (LLMs)**, **Google Maps API**, and a **SQLite database** to deliver dynamic and context-aware responses to users.

Table of Contents

Features

- **Natural Language Conversations**: Engage users in intuitive and meaningful dialogues using **LLMs** integrated via **LangChain**.
- **Location Message Handling**: *Collect* and *store* user location to provide tailored recommendations.
- **Restaurant Recommendations**: Deliver personalized dining suggestions based on user location, preferred cuisine, and search radius using the **Google Maps API**.
- **SQLite Database**: Efficiently manage and retrieve user location with a lightweight **SQLite** database.

Project Structure

```
FoodieLine
├── main.py
├── module
│   ├── model.py
│   ├── user_locations.py
│   └── google_map.py
├── locations.db
├── requirements.txt
└── README.md
```

Implementation

1. **main.py**:
    - **Application Initialization**: Runs the **Flask** app, initializes the **database**.
    - **Chat Bot Webhook Handling**: Manages webhook requests from **LINE**.
    - **Message Processing**:
        - **Text Messages**: Responds to user text inputs using **LLM**.
        - **Location Message**: Records *user location data* in the **database** for personalized recommendations.

2. **model.py**:
    - **LLM Integration with LangChain**: Uses **LangChain** to bind **Large Language Models (LLMs)** with **custom tools**.
    - **Custom Tools Development**:
        - **Restaurant Recommendations**: Suggests suitable dining options according to user *preferences* and *location*.

3. **google_map.py**:
    - **Google Maps API interactions**:
        - Search restarants according to preferred *location*, *keyword* and *distance*

4. **user_locations.py**:
    - **Database Implementation**:
        - *Add* and *update* user location
        - *Check* if user location is stored in database

Technologies Used

- **Flask**: Serves as the web framework to handle HTTP requests and manage the application's routing.
- **LINE Messaging API**: Powers the chatbot's communication capabilities within the LINE platform.
- **LangChain**: A platform that can integrates Large Language Models (LLMs) with custom tools, enhancing the chatbot's conversational intelligence and functionality.
- **Google Maps API**: Provides real-time data for restaurant searches based on user-defined criteria.
- **SQLite**: Offers a lightweight and efficient database solution for managing user location data.