Eat Safe API

Eat Safe API is a Flask-based web service that helps users get concise summaries and important keywords from Chicago restaurant health inspection reports using an AI language model (FLAN-T5). It fetches inspection data from the City of Chicago, summarizes violations, and extracts keywords to help users quickly understand restaurant safety.

Table of Contents

Features

- Fetches real-time restaurant inspection data from the City of Chicago.
- Uses a fine-tuned FLAN-T5 AI model to summarize violation reports.
- Extracts the top 5 most important keywords from each violation.
- Simple REST API for easy integration.

How It Works

1. **User sends a request** with a restaurant name or address.
2. **API fetches inspection data** from the City of Chicago's open data portal.
3. **The most recent violation text** is sent to a fine-tuned FLAN-T5 model.
4. **The model generates** a short summary and 5 keywords.
5. **API returns** the summary and keywords as a JSON response.

Project Structure

```
final_project/
│
├── main.py                  # App entry point, starts Flask server
├── app/
│   ├── __init__.py
```

```
│   ├── routes.py            # Defines API endpoints
│   ├── models/
│   │   └── model.py         # Loads and runs the FLAN-T5 model
│   ├── services/
│   │   └── inspections.py   # Handles Chicago API queries
│   └── utils/
│       └── logger.py        # Logging setup
├── flan_t5_finetuned_final/ # Fine-tuned model and adapter files
│   └── ...                  # Model weights, tokenizer, config, etc.
└── .env                     # (You must create) Stores API keys
```

Setup Instructions

1. Clone the Repository

```
git clone <your-repo-url>
cd final_project
```

2. Install Python Dependencies

Make sure you have Python 3.8+ and pip installed.

```
pip install flask flask-cors python-dotenv sodapy torch transformers peft
```

3. Get a Chicago API Key

- Go to Chicago Data Portal
- Sign up and create an API key.

4. Create a `.env` File

In the project root, create a file named `.env`:

```
CHICAGO_API_KEY=your_chicago_api_key_here
```

5. Download/Place the Model

Ensure the `flan_t5_finetuned_final/` folder contains the fine-tuned model files: - `adapter_model.safetensors` - `adapter_config.json` - Tokenizer files, etc.

If you don't have these, you need to train or obtain the model.

6. Run the API

```
python main.py
```

The server will start at `http://0.0.0.0:5000`.

API Usage

Endpoint

`POST /api/restaurant-info`

Request Body

```
{
  "query": "Giordano's"
}
```

- `query`: Restaurant name or part of the address.

Example Response

```
{
  "summary": "Food not protected from contamination during storage",
  "keywords": ["contamination", "storage", "food", "protection", "safety"]
}
```

Error Responses

- `400`: Missing or invalid query.
- `404`: No inspection data or violations found.

Model Details

- **Model**: FLAN-T5 (fine-tuned for summarizing health violations)
- **Adapter**: Loaded via PEFT (Parameter-Efficient Fine-Tuning)
- **Input**: Violation text from inspection report
- **Output**: One-sentence summary and 5 keywords

Logging

- Logs are printed to the console with timestamps and log levels.
- All major actions (API calls, model loading, errors) are logged.

Troubleshooting

- **No API key error**: Make sure `.env` exists and contains your `CHICAGO_API_KEY`.
- **Model not found**: Ensure `flan_t5_finetuned_final/` contains all required files.
- **CUDA errors**: If you don't have a GPU, the model will run on CPU (slower).
- **peft not installed**: The model will run without adapter weights, but results may be less accurate.

Credits

- City of Chicago Data Portal for inspection data.

- HuggingFace Transformers and PEFT for model loading.
- FLAN-T5 model by Google.

Feel free to ask for more details or help!