Hướng dẫn chi tiết FastAPI NL2SQL Assistant

Author: Nguyễn Phương Anh Tú (Alex Dev)

1. Cài đặt môi trường

{ Alex Dev }

```
# Tạo môi trường ảo
python -m venv venv

# Kích hoạt môi trường ảo
# Windows
venv\Scripts\activate
# Linux/Mac
source venv/bin/activate

# Cài đặt các thư viện cần thiết
pip install fastapi uvicorn langchain ollama pydantic python-dotenv
```

2. Cấu trúc project

{ Alex Dev }

```
fastapi/
├── app.py                # FastAPI application chính
├── db.py                 # Xử lý database
```

```
├── libs/                # Thư viện tùy chỉnh
│   └── langchain.py     # Cấu hình LangChain
├── utils.py             # Các hàm tiện ích
└── .env                 # Biến môi trường
```

3. Cấu hình FastAPI Application

3.1 Khởi tạo FastAPI và các route cơ bản

{ Alex Dev }

```python
from fastapi import FastAPI
from fastapi.middleware.cors import CORSMiddleware

app = FastAPI()

# Cấu hình CORS
app.add_middleware(
    CORSMiddleware,
    allow_origins=["*"],
    allow_credentials=True,
    allow_methods=["*"],
    allow_headers=["*"],
)
```

3.2 Định nghĩa model input

{ Alex Dev }

```python
from pydantic import BaseModel, Field
```

```
class ChatInput(BaseModel):
    context: str = Field(
        default="",
        description="Thông tin các bảng cơ sở dữ liệu"
    )
    file_path: str = Field(
        default="",
        description="Đường dẫn file database"
    )
    prompt: str = Field(
        description="Câu hỏi truy vấn",
        min_length=1,
        max_length=1000
    )
    # Các tham số cho model
    temperature: float = 0.1
    top_k: int = 10
    top_p: float = 0.95
    num_ctx: int = 2048
    repeat_penalty: float = 1.15
```

3.3 Cấu hình LangChain và Ollama

{ Alex Dev }

```
from langchain.llms import Ollama
from langchain.prompts import PromptTemplate
from langchain.cache import InMemoryCache

# Khởi tạo LLM
llm = Ollama(
    base_url="http://localhost:11434",
    model="qwen2.5-coder:0.5b",
    temperature=0.1,
    # Các tham số khác...
)
```

```
# Cấu hình cache
cache = InMemoryCache()
set_llm_cache(cache)

# Template prompt
template = PromptTemplate.from_template(
    """
    {context}
    Tạo câu truy vấn SQL từ câu hỏi:
    {question}
    """
)
```

3.4 API Endpoint xử lý chat

{ Alex Dev }

```
@app.post('/ask')
def ask(chat_input: ChatInput):
    # Khởi tạo database connection
    db = SQLDatabase.from_uri(f"sqlite:///{chat_input.file_path}")

    # Cấu hình model parameters
    config = {
        "temperature": chat_input.temperature,
        "top_k": chat_input.top_k,
        "top_p": chat_input.top_p,
        "num_ctx": chat_input.num_ctx,
        "repeat_penalty": chat_input.repeat_penalty
    }

    # Gọi LLM chain với config
    response = llm_chain.with_config(**config).invoke({
        "context": chat_input.context or f"Tables:
{db.get_table_info()}",
```

```
        "question": chat_input.prompt
    })

    # Thực thi SQL query
    try:
        data_db = db.run(get_sql_from_answer_llm(response))
    except Exception as e:
        data_db = str(e)

    return {
        "answer": response,
        "data_db": data_db
    }
```

4. Chạy ứng dụng

```
{ Alex Dev }

# Development
uvicorn app:app --reload --host 0.0.0.0 --port 8000

# Production
uvicorn app:app --host 0.0.0.0 --port 8000 --workers 4
```

5. API Documentation

- Swagger UI: http://localhost:8000/docs
- ReDoc: http://localhost:8000/redoc

6. Lưu ý quan trọng

1. Đảm bảo Ollama server đang chạy và model đã được tải về
2. Kiểm tra quyền truy cập file database

3. Xử lý lỗi và validation đầu vào
4. Cấu hình CORS phù hợp với môi trường production
5. Tối ưu cache để tăng hiệu năng
6. Xử lý timeout cho các request dài

7. Tham khảo

- FastAPI Documentation: https://fastapi.tiangolo.com/
- LangChain Documentation: https://python.langchain.com/
- Ollama Documentation: https://ollama.ai/docs