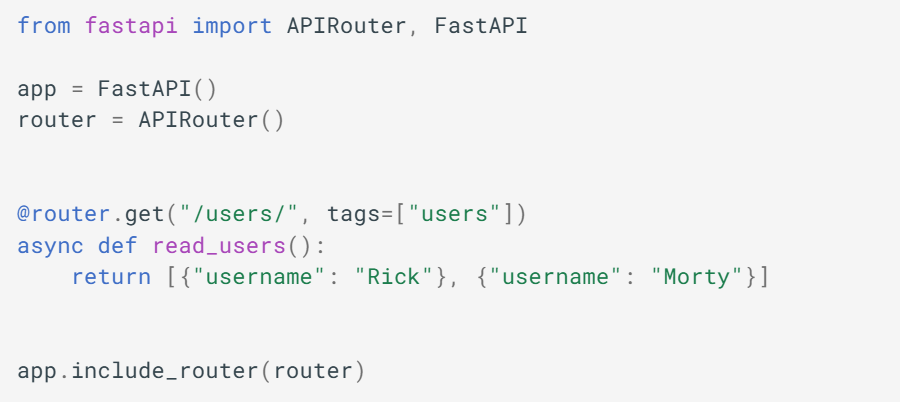
# FastAPI学习

## APIRouter的使用

APIRouter 类，用于对 *路径操作* 进行分组，例如将应用程序结构化为多个文件。然后可以将其包含在 FastAPI 应用程序中，或者包含在另一个 APIRouter 中（最终都包含在应用程序中）。



## Ref的使用

ref 的作用是**创建一个响应式的引用（reactive reference）**，使得当这个引用的值发生变化时，Vue 能够自动追踪并更新相关的视图或副作用。

ref() 让我们能创造一种对任意值的 “引用”

# Sqlite学习

SQLite 是一个 嵌入式数据库，不同于 MySQL、PostgreSQL 等 客户端-服务器型数据库（如需提供 host、port、username、password 等连接信息），它直接以 文件形式 存储在本地磁盘中，所以连接时只需提供 数据库文件的路径 即可。

.db 文件是 SQLite 的“容器”，里面可以包含任意数量的表、视图、索引等数据库对象。相当于时mysql中的数据库。

## 连接sqlite数据库

可以使用sqlalchemy工具

from sqlalchemy import create\_engine

engine = create\_engine('sqlite:///server.db')

上面的代码就创建并连接了sqlite的server数据库了

## 创建表

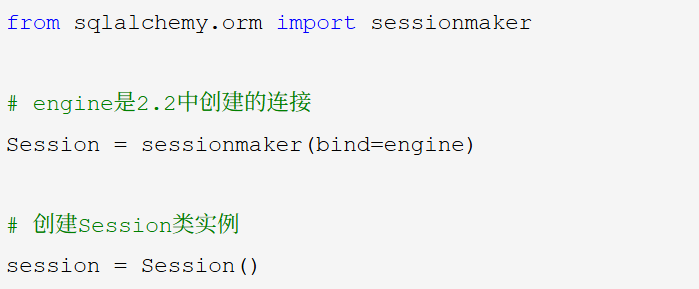
from sqlalchemy.ext.declarative import declarative\_base  
Base = declarative\_base()

Base.metadata.create\_all(engine)

在engine指定的server数据库中创建所有的表，扫描所有继承自 Base 的模型类，根据这些类的定义，自动在数据库中创建对应的表，如果表已经存在，则不会重复创建

## 建立会话

要对表进行增删改查就得通过会话



## 增

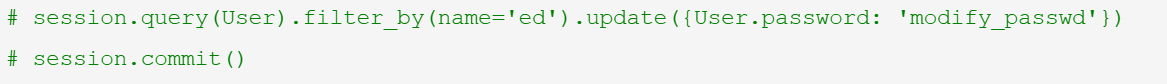


## 改



上边的操作，先查询再修改相当于执行了两条语句，和我们印象中的update不一致

# 可直接使用下边的写法，传给服务端的就是update语句



# 系统功能

## 登录功能

首先会先判断是否是第一次启动系统（通过查询数据库的user表，如果记录数count==0，说明是第一次），如果是第一次，那么会要求创建超级管理员账户

正常登录（不是第一次）

从请求中获取用户名和密码，查询数据库中的用户记录，验证密码是否匹配，匹配则更新user表对应记录的 last\_login 时间。生成 JWT access\_token。将登录日志保存到operation\_logs表中。返回 access\_token、用户信息等。

# 项目启动

## 前端

npm install pnpm

pnpm install

pnpm dev

# 项目简历

1.使用JWT进行登录校验