# 后端连接数据库

## 0.数据准备

MySQl数据库启动，mysql -u root -p进入数据库：

create user dbuser identified by ‘dbuser1234’;

create database newdb;

grant all privileges on newdb.\* to dbuser;

mysql -u dbuser -p 进入数据库

use newdb;

create table users(id int primary key,name varchar(100),age int,city varchar(100));

insert into users values(1,’zhangsan’，18,’Tianjin’);

数据库ip：127.0.0.1，端口号：3306，用户名：dbuser，密码dbuser1234，数据库名称：newdb

## 1. 安装依赖

首先，确保你的 Python 环境中已经安装了 FastAPI、uvicorn、SQLAlchemy 和 pymysql。如果尚未安装，可以通过以下命令安装：

pip install fastapi uvicorn sqlalchemy pymysql

## **2. 创建项目文件夹结构**

创建一个简单的项目文件夹结构，例如：

fastapi/

├── \_\_init\_\_.py

├── main.py

├── database.py

├── models.py

├── schemas.py

└── crud.py

## **3. 配置数据库连接（database.py）**

from sqlalchemy import create\_engine

from sqlalchemy.ext.declarative import declarative\_base

from sqlalchemy.orm import sessionmaker

SQLALCHEMY\_DATABASE\_URL = "mysql+pymysql://用户名:密码@主机地址:端口号/数据库名"

engine = create\_engine(

SQLALCHEMY\_DATABASE\_URL,

pool\_size=5, # 连接池的大小

max\_overflow=10, # 超出连接池大小后，最多可以创建的连接数

pool\_timeout=30, # 连接池中没有连接时，等待连接的时间

pool\_recycle=3600, # 连接的回收时间

echo=True, # 显示SQL语句

pool\_pre\_ping=True, # 连接之前ping一下

connect\_args={

"charset": "utf8mb4", # 设置编码

}

)

SessionLocal = sessionmaker(autocommit=False, autoflush=False, bind=engine)

Base = declarative\_base()

## **4. 定义数据模型（models.py）**

from sqlalchemy import Column, Integer, String

from database import Base

class User(Base):

\_\_tablename\_\_ = "users"

id = Column(Integer, primary\_key=True, index=True)

name = Column(String(100), index=True)

age = Column(String(50), index=True)

city = Column(String(100), index=True)

**5. 定义 Pydantic 模型（schemas.py）**

from pydantic import BaseModel

class UserBase(BaseModel):

name: str

age:int

city: str

class UserCreate(UserBase):

pass

class User(UserBase):

id: int

class Config:

orm\_mode = True

**6. 创建 CRUD 操作（crud.py）**

from sqlalchemy.orm import Session

import models, schemas

def get\_user(db: Session, user\_id: int):

return db.query(models.User).filter(models.User.id == user\_id).first()

def create\_user(db: Session, user: schemas.UserCreate):

db\_user = models.User(name=user.name, age=user.age,city=user.city)

db.add(db\_user)

db.commit()

db.refresh(db\_user)

return db\_user

**7. 创建 FastAPI 应用（main.py）**

from fastapi import FastAPI, Depends, HTTPException

from sqlalchemy.orm import Session

import crud, models, schemas

from database import SessionLocal, engine

models.Base.metadata.create\_all(bind=engine)

app = FastAPI()

def get\_db():

db = SessionLocal()

try:

yield db

finally:

db.close()

@app.post("/users/", response\_model=schemas.User)

def create\_user(user: schemas.UserCreate, db: Session = Depends(get\_db)):

db\_user = crud.get\_user(db, user\_id=user.id)

if db\_user:

raise HTTPException(status\_code=400, detail="User already registered")

return crud.create\_user(db=db, user=user)

@app.get("/users/{user\_id}", response\_model=schemas.User)

def read\_user(user\_id: int, db: Session = Depends(get\_db)):

db\_user = crud.get\_user(db, user\_id=user\_id)

if db\_user is None:

raise HTTPException(status\_code=404, detail="User not found")

return db\_user

**8. 启动 FastAPI 服务器**

在项目根目录下运行以下命令启动 FastAPI 服务器：

uvicorn main:app --reload

现在，你可以通过访问 http://127.0.0.1:8000/docs 或 http://127.0.0.1:8000/redoc 来查看和测试你的 API。

**注意事项：**

* 确保 MySQL 数据库已经创建，并且 SQLALCHEMY\_DATABASE\_URL 中的用户名、密码、主机地址、端口号和数据库名正确无误。
* 如果你使用的是 Docker 来运行 MySQL，可以参考以下命令：

docker pull mysql:8.0.25

docker run --name mysql --restart=always -e MYSQL\_ROOT\_PASSWORD=ygg123456 -p 3306:3306 -d mysql:8.0.25

docker exec -it mysql mysql -uroot -p

create database test charset utf8mb4;

* 如果需要执行数据库迁移，可以使用 Alembic 等工具。