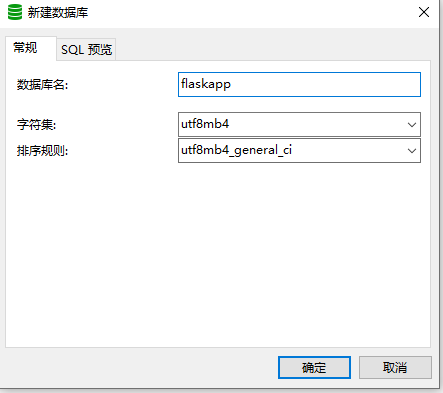
**2.8Flask基础知识·Flask连接Mysql数据库**

做后端服务，操作数据库是必不可少的环节，这里简单介绍一下Flask是如何连接Mysql数据库的。本文默认Mysql已经安装好的前提下进行的。

# 新建数据库

略



# 新建表

CREATE TABLE `sys\_user` (  
 `id` bigint(20) NOT NULL COMMENT '用户ID',  
 `user\_name` varchar(32) NOT NULL COMMENT '用户名',  
 `real\_name` varchar(32) NOT NULL COMMENT '姓名',  
 `nick\_name` varchar(32) DEFAULT NULL COMMENT '昵称',  
 `avatar` varchar(100) DEFAULT NULL COMMENT '用户头像',  
 `password` varchar(64) DEFAULT NULL COMMENT '用户密码',  
 `salt` varchar(10) DEFAULT NULL COMMENT '密码加盐',  
 `mobile\_phone` varchar(11) NOT NULL COMMENT '手机号',  
 `tel` varchar(20) DEFAULT NULL COMMENT '联系电话',  
 `email` varchar(32) DEFAULT NULL COMMENT '邮箱',  
 `admin\_type` int(11) DEFAULT NULL COMMENT '管理员类型<sys\_admin\_type>',  
 `sex` int(11) DEFAULT NULL COMMENT '性别<sys\_sex>',  
 `is\_locked` tinyint(1) DEFAULT '0' COMMENT '是否锁定',  
 `dept\_id` bigint(20) DEFAULT NULL COMMENT '所属部门',  
 `post\_id` bigint(20) DEFAULT NULL COMMENT '所属岗位',  
 `remark` varchar(255) DEFAULT NULL COMMENT '备注',  
 `create\_time` datetime(3) DEFAULT NULL COMMENT '创建时间',  
 `create\_user` bigint(20) DEFAULT NULL COMMENT '创建用户',  
 `update\_time` datetime(3) DEFAULT NULL COMMENT '更新时间',  
 `update\_user` bigint(20) DEFAULT NULL COMMENT '更新用户',  
 `is\_deleted` tinyint(1) DEFAULT '0' COMMENT '是否删除',  
 PRIMARY KEY (`id`) USING BTREE,  
 KEY `idx\_sys\_user\_user\_name` (`user\_name`) USING BTREE,  
 KEY `idx\_sys\_user\_real\_name` (`real\_name`) USING BTREE,  
 KEY `idx\_sys\_user\_mobile\_phone` (`mobile\_phone`) USING BTREE  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='用户';

# 插入一条记录

INSERT INTO `flaskapp`.`sys\_user`(`id`, `user\_name`, `real\_name`, `nick\_name`, `avatar`, `password`, `salt`, `mobile\_phone`, `tel`, `email`, `admin\_type`, `sex`, `is\_locked`, `dept\_id`, `post\_id`, `remark`, `create\_time`, `create\_user`, `update\_time`, `update\_user`, `is\_deleted`) VALUES (1567762634947375105, 'admin', '普通管理员', NULL, NULL, 'e82098b6d649e98d3c4bc7731baab8c8', 'az3xjbk0', '18600000001', '19833173224', 't.eeq@kcshc.nc', 2, 1, 0, 1575377096640569346, 1575379311115640834, NULL, '2022-09-08 14:31:54.450', 0, '2023-02-10 14:22:38.455', 1567738052492341249, 0);

# 激活虚拟环境

source ./venv/ｓｃｒｉｐｔs/activate

# 安装SQLAlchemy

# 安装最新版本  
pip3 install SQLAlchemy  
# 指定安装版本  
pip3 install SQLAlchemy==1.4.26

# IMG_257

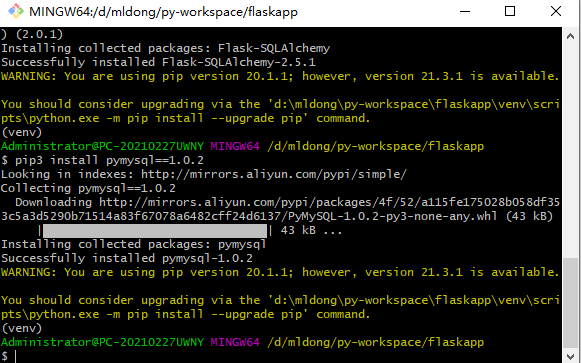
# 安装Flask-SQLAlchemy

# 安装最新版本  
pip3 install Flask-SQLAlchemy  
# 指定安装版本  
pip3 install Flask-SQLAlchemy==2.5.1

# IMG_258

# 安装pymysql

# 安装最新版本  
pip3 install pymysql  
# 指定安装版本  
pip3 install pymysql==1.0.2



# 创建模型目录

mkdir models

# 新增models/\_\_init\_\_.py文件

from flask\_sqlalchemy import SQLAlchemy  
  
db = SQLAlchemy()

# 修改app.py文件

from urllib import parse  
  
from flask import Flask, request  
  
from controllers.role\_controller import role  
from controllers.user\_controller import user  
from modeles import db  
  
DB\_HOST = "localhost"  
DB\_PORT = "3306"  
DB\_NAME = "flaskapp"  
DB\_USER = "root"  
DB\_PASSWORD = "root"  
# 防止密码中有特殊字符，需要使用parse.quote\_plus进行转义  
SQLALCHEMY\_DATABASE\_URI = f'mysql+pymysql://{DB\_USER}:{parse.quote\_plus(DB\_PASSWORD)}@{DB\_HOST}:{DB\_PORT}/{DB\_NAME}?charset=utf8mb4'  
  
app = Flask(\_\_name\_\_)  
  
# 注册用户模块  
app.register\_blueprint(user)  
# 注册角色模块  
app.register\_blueprint(role)  
# Flask中新增数据库配置  
app.config['SQLALCHEMY\_DATABASE\_URI'] = SQLALCHEMY\_DATABASE\_URI  
app.config['SQLALCHEMY\_TRACK\_MODIFICATIONS'] = True  
app.config['SQLALCHEMY\_RECORD\_QUERIES'] = True  
app.config['SQLALCHEMY\_ENGINE\_OPTIONS'] = {  
 'pool\_size': 5,  
 'pool\_timeout': 90,  
 'pool\_recycle': 7200,  
 'max\_overflow': 1024  
 }  
  
# 初始化db  
db.init\_app(app)  
  
  
@app.before\_request  
def auth():  
 WHITE\_LIST = ['/sys/user/detail', '/sys/user/page', '/ex/test', '/db/test']  
 # 简单处理一下，非白名单路由，提示登录  
 if request.path not in WHITE\_LIST:  
 return "请先登录"  
  
  
@app.route("/db/test")  
def db\_test():  
 cursor = db.session.execute('select \* from t\_user')  
 result = cursor.fetchall()  
 if len(result) > 0:  
 u = result[0]  
 return f"id:{u.id},user\_name:{u.user\_name}"  
 return "无记录"  
  
  
@app.route("/ex/test")  
def ex\_test():  
 """  
 测试其他异常  
 :return:  
 """  
 a = 3 / 0  
 return a  
  
  
@app.errorhandler(404)  
def error(e):  
 """  
 404异常处理  
 :param e:  
 :return:  
 """  
 return "请求地址不存在"  
  
  
@app.errorhandler(Exception)  
def error(e):  
 """  
 其他异常处理  
 :param e: 异常  
 :return:  
 """  
 return str(e)  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 app.run(host="0.0.0.0", port=5000)

# ****运行Flask服务****

flask run

# 使用curl命令访问

## 访问/db/test

curl http://localhost:5000/db/test

# 使用Apifox/Postman接口测试工具访问

略