**4.3多环境配置·自定义开发、测试和生产环境配置**

# 重构config.py

import os  
from urllib import parse  
  
  
class BaseConfig(ｏｂｊｅｃｔ):  
 """  
 基础配置  
 """  
 DEBUG = True # 调试模式  
 APP\_AUTHOR = "mldong" # 作者  
 WHITE\_LIST = ['/user/get', '/user/list', '/ex/test', '/ex/other', "/db/test", "/config/test"] # 权限白名单  
 DB\_HOST = os.getenv("DB\_HOST", "localhost") # 数据库ip  
 DB\_PORT = os.getenv("DB\_PORT", "3306") # 数据库端口  
 DB\_NAME = os.getenv("DB\_NAME", "flaskapp") # 数据库名称  
 DB\_USER = os.getenv("DB\_USER", "root") # 数据库用户  
 DB\_PASSWORD = os.getenv("DB\_PASSWORD", "root") # 数据库密码  
 # 数据库相关配置开始  
 # 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'  
 SQLALCHEMY\_TRACK\_MODIFICATIONS = True  
 SQLALCHEMY\_RECORD\_QUERIES = True  
 SQLALCHEMY\_ENGINE\_OPTIONS = {  
 'pool\_size': 5,  
 'pool\_timeout': 90,  
 'pool\_recycle': 7200,  
 'max\_overflow': 1024  
 }  
   
 # 数据库相关配置结束  
 JSON\_AS\_ASCII = False # 禁止中文转义  
  
  
class Development(BaseConfig):  
 PORT = 5000  
 DEBUG = True  
 ENV = "dev"  
 TESTING = True  
 SQLALCHEMY\_ECHO = True # 打印SQL  
  
  
class Test(BaseConfig):  
 PORT = 5000  
 DEBUG = True  
 ENV = "test"  
 TESTING = True  
  
  
class Production(BaseConfig):  
 PORT = 5000  
 DEBUG = False  
 ENV = "prod"  
 TESTING = False  
  
  
def getConfig():  
 # 从环境变量中加载ENV  
 env = os.getenv("ENV", "dev")  
 if env == "test":  
 return Test  
 elif env == "prod" or env == "production":  
 return Production  
 else:  
 return Development

# 修改app.py文件

from flask import Flask, request  
  
from config import getConfig  
from controllers.role\_controller import role  
from controllers.user\_controller import user  
from models import db  
  
app = Flask(\_\_name\_\_)  
  
# 注册用户模块  
app.register\_blueprint(user)  
# 注册角色模块  
app.register\_blueprint(role)  
# 从配置对象中加载  
app.config.from\_ｏｂｊｅｃｔ(getConfig())  
  
# 初始化db  
db.init\_app(app)  
  
  
@app.before\_request  
def auth():  
  
 # 简单处理一下，非白名单路由，提示登录  
 if request.path not in app.config['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("/config/test")  
def config\_test():  
 """  
 配置测试  
 :return:  
 """  
 return f"APP\_AUTHOR:{app.config['APP\_AUTHOR']},ENV:{app.config['ENV']}"  
  
  
@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")

# 运行Flask服务

flask run

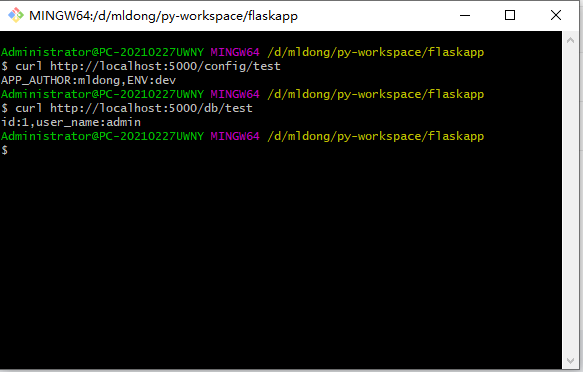
# 使用curl命令访问

## 访问/config/test

curl http://localhost:5000/config/test

## 访问/db/test

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



# 修改ENV环境为生产环境

export ENV=production

# 再次重启Flask

flask run

# 再次使用curl命令访问

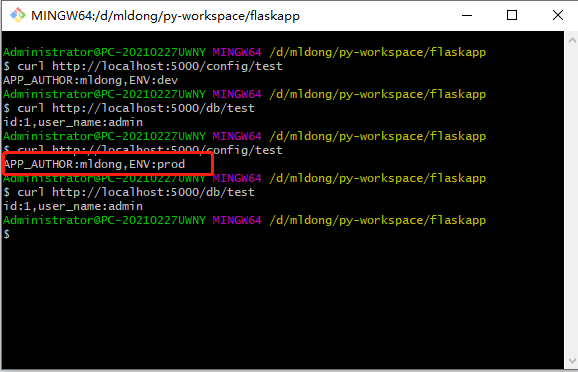
## 访问/config/test

curl http://localhost:5000/config/test

## 访问/db/test

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

结果如下：



# 测试环境也同上

略