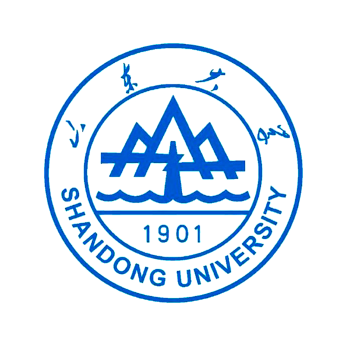
****

**Web技术大作业**

**学生信息管理系统**

**实验报告**

****

**组员学号姓名：202100800521安竟豪**

**202100800570林正阳**

**202100800609孙义捷**

**202100800646于瑞东**

**指导教师：刘猛**

**编写日期：2023-11-25**

1. 实验内容

理解并掌握动态网页制作相关概念和技术，综合利用所学的 Web 技术，掌

握基于 Flask 开发 B/S 架构系统的方法。首先重现课堂演示系统，并在此基础上根据功能要求改进和优化课堂演示系统，并测试通过。其中实现功能如下：

1、 增加数据合法性验证功能，包括客户端验证和服务端验证。

2、 实现操作结果显示功能，比如添加数据时提示添加成功/添加失败、删除

时提示删除成功等，实现方式可以通过显示页面元素内容提示或客户端 JS

弹出对话框方式。

3、 实现批量删除功能。

4、 实现年龄的批量修改功能。

5、 实现数据显示分页功能。

6、 为登录页面添加验证码功能。

7、 改进用户密码存储方式，防止密码泄露。

8、 适当页面美化工作（选做）。

2. 实验分工

202100800521安竟豪：功能一和功能二的实现。

202100800570林正阳：功能三和功能四的实现。

202100800609孙义捷：功能五和功能六的实现。

202100800646于瑞东：功能七和功能八的实现。

3. 功能的具体实现

3.1 功能一设计与实现

**问题：** 增加数据合法性验证功能，包括客户端验证和服务端验证。

**思路：** 数据合法性验证即验证数据的合理性，在这里实现了以下三点：

（1）增加登录表单验证：用户名和密码不能为空

（2）添加学生时会有主键约束，否则会弹出js对话框。表单验证：学号9位，姓名汉字且不超过五个字，年龄大于17小于25，否则会给出提示

（3）修改学生信息同理需要满足上述条件，否则表单不能提交

采用HTML的pattern属性实现上述要求。在HTML表单元素中，pattern属性用于指定一个正则表达式，该表达式定义了输入字段的合法值模式。在用户名和密码的input元素上，使用pattern=".{1,}"来指定一个正则表达式，要求输入至少包含一个字符，即用户名和密码不能为空。

对于第二点，学号：pattern="\d{9}"：这个正则表达式要求学号必须为9位数字。

姓名：pattern="^[\u4e00-\u9fa5]{1,5}$"：这个正则表达式要求姓名必须为汉字，且不超过五个字。

年龄：pattern="1[7-9]|2[0-5]"：这个正则表达式要求年龄必须大于等于17且小于等于25。

同时完善添加学生的代码，使其可以捕获并返回错误信息，当主键冲突时，可以弹出js对话框（对话框的详细实现在功能二设计与实现中说明）。修改学生信息同理采用上述方法，为避免冲突，修改不能修改学生学号。关键代码，相关实现截图见下方。

**结果：**

**登录表单验证代码：**

<div class="form-group">

<label class="control-label col-sm-4">用户名：</label>

<div class="col-sm-8">

<input class="form-control" name="username" type="text" id="username" pattern=".{1,}" required>

</div>

</div>

<div class="form-group">

<label class="control-label col-sm-4">密码：</label>

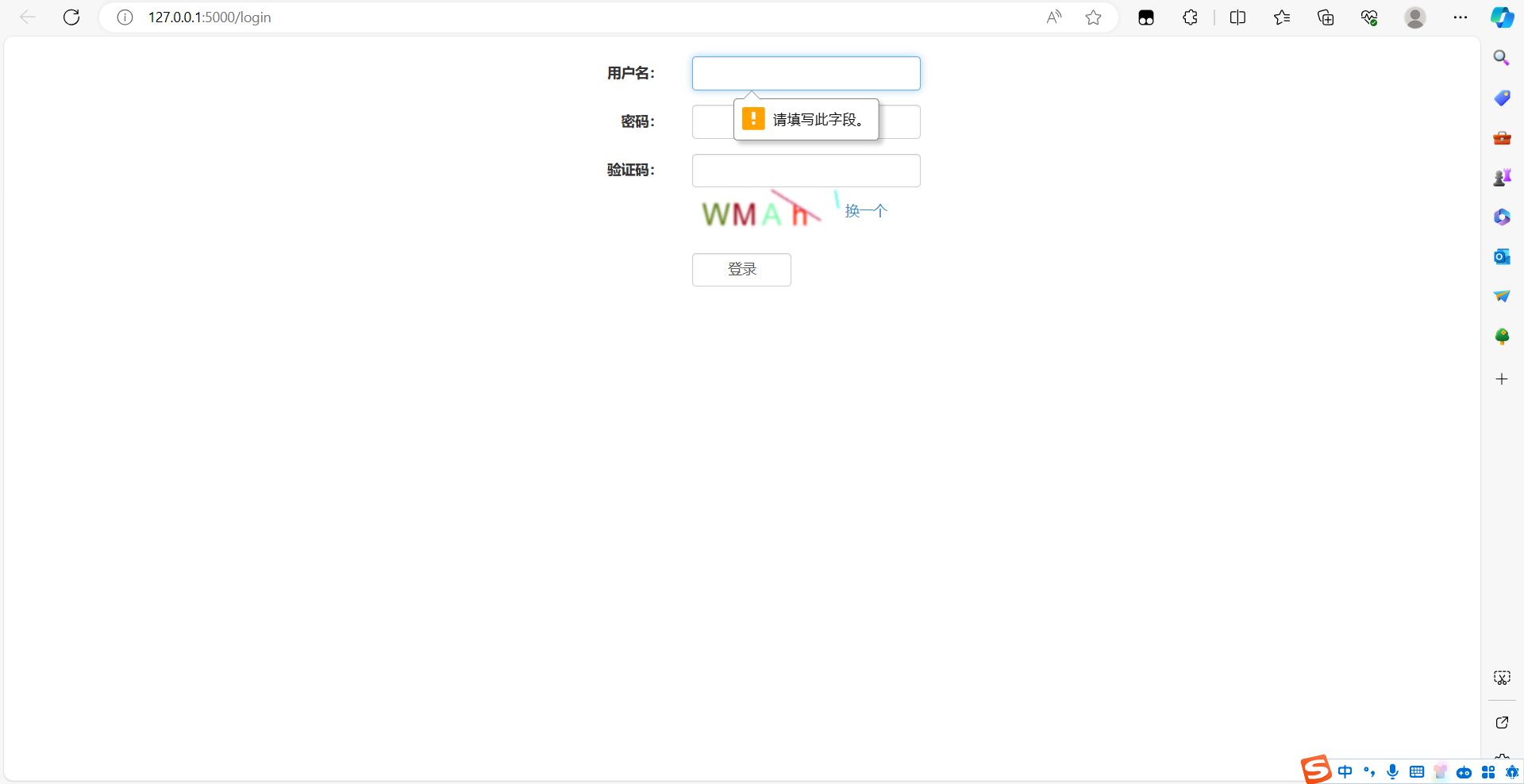
<div class="col-sm-8">

<input class="form-control" name="pwd" id="pwd" type="password" pattern=".{1,}" required>

</div>

</div>

登录如果输入了空内容会有提示：



**添加学生信息表单关键代码：**

<form id="form1" name="form1" method="post" action="/add">

<label>学号： </label>

<input type="text" name="stu\_id" pattern="\d{9}" title="学号必须为9位数字" required />

<p>

<label>姓名： </label>

<input type="text" name="stu\_name" pattern="^[\u4e00-\u9fa5]{1,5}$" title="姓名必须为汉字，且不超过五个字"

required />

</p>

<p>

<label>性别：</label>

<input name="stu\_sex" type="radio" value="男" checked="checked" />

男

<input type="radio" name="stu\_sex" value="女" />

女

<br />

</p>

<p>

<label>年龄：

<input type="text" name="stu\_age" pattern="1[7-9]|2[0-5]" title="年龄必须大于等于17且小于等于25" required/>

</label>

</p>

<p>

<label>籍贯：</label>

<input type="text" name="stu\_origin" />

</p>

<p>

<label>专业：</label>

<select name="stu\_profession">

{% for pro in datas -%}

<option value='{{pro[0]}}'>{{pro[1]}}</option>

{%endfor-%}

</select>

</p>

<p>

<input class="btn btn-primary" type="submit" name="button" value="提交" />

<input type="reset" name="button2" value="重置" />

<br />

</p>

</form>

**修改的添加函数代码：**

def InsertData(data, tablename):

conn = OpenDb()

values = []

cursor = conn.cursor()

fieldNames = list(data)

for v in fieldNames:

values.append(data[v])

try:

sql = "insert into %s (%s) values(%s)" % (tablename, ",".join(fieldNames), ",".join(["?"] \* len(fieldNames)))

cursor.execute(sql, values)

conn.commit()

CloseDb(conn)

return True, None # 返回成功的标志和无错误消息

except sqlite3.IntegrityError as e:

if "UNIQUE constraint failed" in str(e):

# 处理主键冲突异常

CloseDb(conn)

return False, "学号已存在，请使用其他学号"

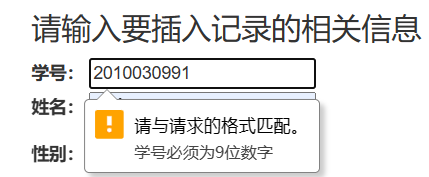
else:

conn.rollback()

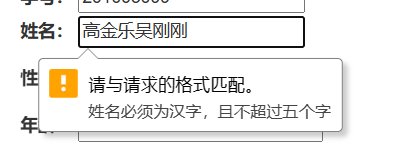
CloseDb(conn)

return False, str(e) # 返回失败的标志和错误消息

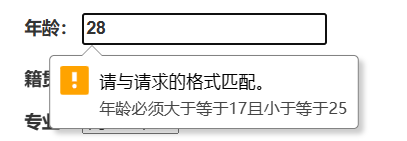
未按要求添加信息相关截图如下，学号位数不对：



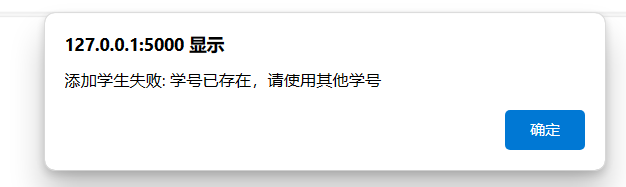
姓名未按规定填写：



年龄超出范围：



主键约束未满足，弹出对话框：



**修改学生信息表单关键代码：**

<form id="form1" name="form1" method="post" action="/update">

<label>学号：{{data[0]}}</label>

<input type="hidden" name="stu\_id" value='{{data[0]}}'/>

<p>

<label>姓名： </label>

<input type="text" name="stu\_name" value='{{data[1]}}' pattern="[\u4e00-\u9fa5]{1,5}" title="姓名必须为汉字，且不超过五个字" required />

</p>

<p>

<label>性别：</label>

{% if data[2]=="男" %}

<input name="stu\_sex" type="radio" value="男" checked="checked"/>

男

<input type="radio" name="stu\_sex" value="女"/>

女

{% else %}

<input name="stu\_sex" type="radio" value="男"/>

男

<input type="radio" name="stu\_sex" value="女" checked="checked"/>

女

{% endif %}

<br/>

</p>

<p>

<label>年龄： </label>

<input type="text" name="stu\_age" value='{{data[3]}}' pattern="(?:1[7-9]|20|21|22|23|24|25)" title="年龄必须在17到25之间" required />

</p>

<p>

<label>籍贯：</label>

<input type="text" name="stu\_origin" value='{{data[4]}}'/>

</p>

<p>

<label>专业：</label>

<select name="stu\_profession">

{% for pro in datas %}

<option value='{{pro[0]}}' {{'selected' if pro[0]==data[5] }}>{{pro[1]}}</option>

{% endfor %}

</select>

</p>

<p>

<input type="submit" name="button" value="提交"/>

<input type="reset" name="button2" value="重置"/>

<br/>

</p>

</form>

**修改学生信息函数关键代码：**

def UpdateData(data, tablename):

conn = OpenDb()

values = []

cusor = conn.cursor()

idName = list(data)[0]

try:

for v in list(data)[1:]:

values.append("%s='%s'" % (v, data[v]))

sql = "update %s set %s where %s='%s'" % (tablename, ",".join(values), idName, data[idName])

cusor.execute(sql)

conn.commit()

CloseDb(conn)

return True, "更新成功"

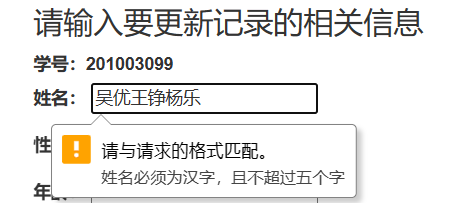
except Exception as e:

# 如果发生异常，返回更新失败信息

CloseDb(conn)

return False, "更新失败: {}".format(str(e))

未按照要求修改学生信息，结果和上面的截图相似：





3.2 功能二设计与实现

**问题：** 实现操作结果显示功能，比如添加数据时提示添加成功/添加失败、删除时提示删除成功等，实现方式可以通过显示页面元素内容提示或客户端 JS弹出对话框方式。

**思路：**添加以下对话框，以增强交互感:

（1）修改成功/失败弹出对话框

（2）添加成功/失败弹出对话框

（3）删除成功/失败弹出对话框

（4）查询失败弹出对话框

在Flask中，flash消息用于在一个请求中传递一次性的消息给用户，通常用于显示成功信息、错误信息或其他通知。对话框就是基于此实现的。在main.py的特定路由的视图函数中，设置flash消息，然后在模板中，通过调用get\_flashed\_messages()来获取Flash消息，从而弹出对话框。

**结果：**

**Show1.html中添加获取flash消息并弹出对话框的代码：**

<!-- 显示添加成功的消息 -->

<div id="flash-messages">

{% with messages = get\_flashed\_messages() %}

{% if messages %}

<script>

// 使用 JavaScript 弹出消息对话框

alert('{{ messages[0] }}');

</script>

{% endif %}

{% endwith %}

</div>

**修改学生信息函数代码（返回成功或者失败消息）：**

def UpdateData(data, tablename):

conn = OpenDb()

values = []

cusor = conn.cursor()

idName = list(data)[0]

try:

for v in list(data)[1:]:

values.append("%s='%s'" % (v, data[v]))

sql = "update %s set %s where %s='%s'" % (tablename, ",".join(values), idName, data[idName])

cusor.execute(sql)

conn.commit()

CloseDb(conn)

return True, "更新成功"

except Exception as e:

# 如果发生异常，返回更新失败信息

CloseDb(conn)

return False, "更新失败: {}".format(str(e))

**修改学生信息路由的视图函数代码：**

@app.route('/update', methods=['GET', 'post'])

def update():

if not CheckLogin():

return redirect(url\_for('login'))

if request.method == "GET":

id = request.args['id']

result, \_ = GetSql2("select \* from student\_info where stu\_id='%s'" % id)

print(result[0])

print(type(result[0]))

datas, \_ = GetSql2("select \* from stu\_profession")

return render\_template('update.html', data=result[0], datas=datas)

else:

data = dict(

stu\_id=request.form['stu\_id'],

stu\_name=request.form['stu\_name'],

stu\_sex=request.form['stu\_sex'],

stu\_age=request.form['stu\_age'],

stu\_origin=request.form['stu\_origin'],

stu\_profession\_id=request.form['stu\_profession']

)

success, error\_message = UpdateData(data, "student\_info")

if success:

flash('修改学生成功: {}'.format(request.form['stu\_name']), 'success')

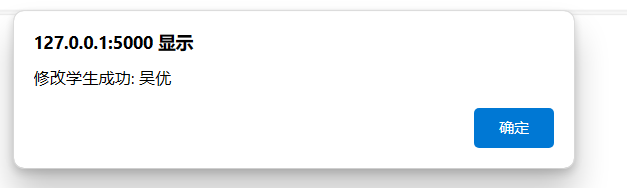
else:

flash('修改学生失败: {}'.format(error\_message), 'error')

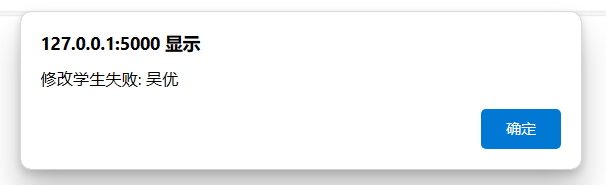
return redirect(url\_for('index'))

修改成功的截图：

修改成功，显示修改的学生名字，并显示成功



修改失败截图是把成功改为失败，但是由于表单验证，因此失败情况很少见，在特定情况下，也会弹出以下对话框：



**添加学生函数代码：**

def InsertData(data, tablename):

conn = OpenDb()

values = []

cursor = conn.cursor()

fieldNames = list(data)

for v in fieldNames:

values.append(data[v])

try:

sql = "insert into %s (%s) values(%s)" % (tablename, ",".join(fieldNames), ",".join(["?"] \* len(fieldNames)))

cursor.execute(sql, values)

conn.commit()

CloseDb(conn)

return True, None # 返回成功的标志和无错误消息

except sqlite3.IntegrityError as e:

if "UNIQUE constraint failed" in str(e):

# 处理主键冲突异常

CloseDb(conn)

return False, "学号已存在，请使用其他学号"

else:

conn.rollback()

CloseDb(conn)

return False, str(e) # 返回失败的标志和错误消息

**添加学生路由的视图函数代码：**

@app.route('/add', methods=['GET', 'post'])

def add():

if not CheckLogin():

return redirect(url\_for('login'))

if request.method == "GET":

datas, \_ = GetSql2("select \* from stu\_profession")

return render\_template('add.html', datas=datas)

else:

data = dict(

stu\_id=request.form['stu\_id'],

stu\_name=request.form['stu\_name'],

stu\_sex=request.form['stu\_sex'],

stu\_age=request.form['stu\_age'],

stu\_origin=request.form['stu\_origin'],

stu\_profession\_id=request.form['stu\_profession']

)

# 调用 InsertData 函数插入数据

success, error\_message = InsertData(data, "student\_info")

if success:

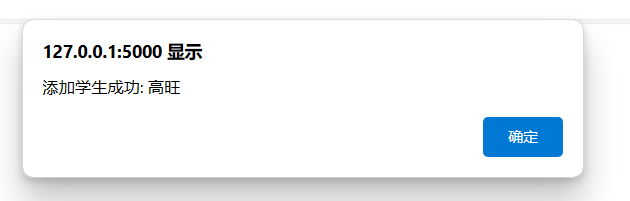
flash('添加学生成功: {}'.format(request.form['stu\_name']), 'success')

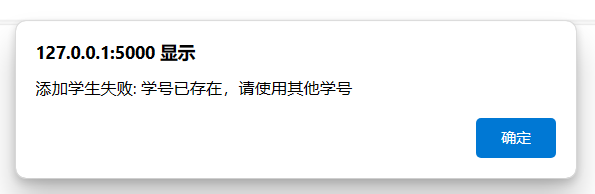
else:

flash('添加学生失败: {}'.format(error\_message), 'error')

return redirect(url\_for('index', id=index\_page))

**添加成功/失败截图：**





**删除学生函数代码：**

def DelDataById(id, value, tablename):

conn = OpenDb()

values = []

cusor = conn.cursor()

sql = "delete from %s where %s=?" % (tablename, id)

# print (sql)

cusor.execute(sql,(value,))

conn.commit()

CloseDb(conn)

**根据学生学号获取姓名函数代码：**

def GetStudentNameById(student\_id):

conn = OpenDb()

cursor = conn.cursor()

try:

cursor.execute("SELECT stu\_name FROM student\_info WHERE stu\_id = ?", (student\_id,))

result = cursor.fetchone()

if result:

return result[0]

else:

return None

finally:

CloseDb(conn)

**删除学生路由的视图函数代码：**

@app.route('/del2/<idi>', methods=['GET'])

def delete2(idi):

if not CheckLogin():

return redirect(url\_for('login'))

student\_name = GetStudentNameById(idi) # 获取学生姓名

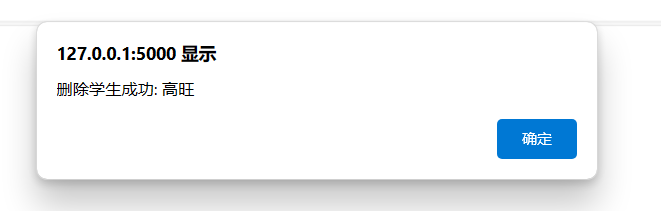
DelDataById("stu\_id", idi, "student\_info")

# 在删除成功后，使用 flash 传递消息

flash('删除学生成功: {}'.format(student\_name), 'success')

return redirect(url\_for('index', id=index\_page))

**添加成功截图：**



**查询学生信息关键代码：**

sql = f"SELECT s.\*, p.stu\_profession FROM student\_info s INNER JOIN stu\_profession p " \

f"on s.stu\_profession\_id=p.stu\_profession\_id LIMIT {start\_index}, {items\_per\_page}"

strWhere = []

if "name" in request.args:

name = request.args["name"]

if name != "":

strWhere.append(f"stu\_name LIKE '%%%s%%'" % name)

if "stuno" in request.args:

stuno = request.args["stuno"]

if stuno != "":

strWhere.append(f"stu\_id = '%s'" % stuno)

if len(strWhere) > 0:

sql = "SELECT s.\*, p.stu\_profession FROM student\_info s INNER JOIN stu\_profession p " \

"on s.stu\_profession\_id=p.stu\_profession\_id"

sql = sql + " WHERE " + " AND ".join(strWhere)

total\_pages=1

print(sql)

result, fields = GetSql2(sql)

if result:

pass

else:

flash('找不到结果', 'error')

fields = ["学号", "姓名", "性别", "年龄", "籍贯", "专业号", "专业"]

return render\_template('show1.html', datas=result, fields=fields, index\_page=index\_page, total\_pages=total\_pages)

**查询学生信息函数：**

def GetSql2(sql):

conn = OpenDb()

result,fields = GetSql(conn, sql)

CloseDb(conn)

return result,fields

def GetSql(conn, sql):

cur = conn.cursor()

cur.execute(sql)

fields=[]

for field in cur.description:

fields.append(field[0])

result = cur.fetchall()

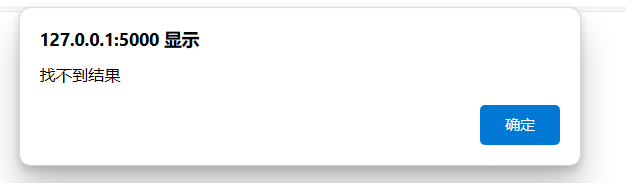
# for item in result:

# print(item)

cur.close()

return result,fields

查询成功会直接显示信息，查询失败则会弹出对话框：



3. 实验结论（实验过程遇到的难题及解决方法、实验心得）

这次的Web大作业是我们小组合作的一次很棒的经历。在项目中，我们面临了许多挑战，但也收获了许多宝贵的经验。

对于功能一和功能二的实现，我查阅了资料，了解到对话框的调出可以采用flash库的方式，并学习了相关代码，运用到了项目中，可以正常实现相关功能。

4. 参考说明

1. [flash消息分类与美化 - 知乎 (zhihu.com)](https://zhuanlan.zhihu.com/p/23374605)

2. [Flask 基础教程 - Flask教程 - 菜鸟教程 (cainiaojc.com)](https://www.cainiaojc.com/flask/)