

**构件与中间件技术**

课程实验报告

实验名称：Java Web工程构建（二）

实验次数：第三次

任课教师：徐悦甡

课程班级：Web工程方向

学号姓名：21009201175 霍志杰

提交日期：2024年5月23日

1. **实验名称**

第三次实验：JavaWeb工程构建（二）

1. **实验日期**

2024年5月23日

1. **实验学生**

21009201175霍志杰

1. **实验目的**

本实验的主要目标是通过实际操作，深入理解并掌握如何运用Java Web技术（特别是JSP、Servlet以及Tomcat服务器）来构建一个具备实际功能的Web应用——一个简单的学生信息查询系统。该系统允许用户通过输入学生的学号，查询并展示该学生的相关信息，如姓名、班级等。实验旨在强化学生对前后端交互、数据库操作及Web服务器配置的理解与实践能力。

1. **实验内容**

**技术栈：**JSP (Java Server Pages), Tomcat (服务器), Servlet (Java 后端控制，MySQL数据库

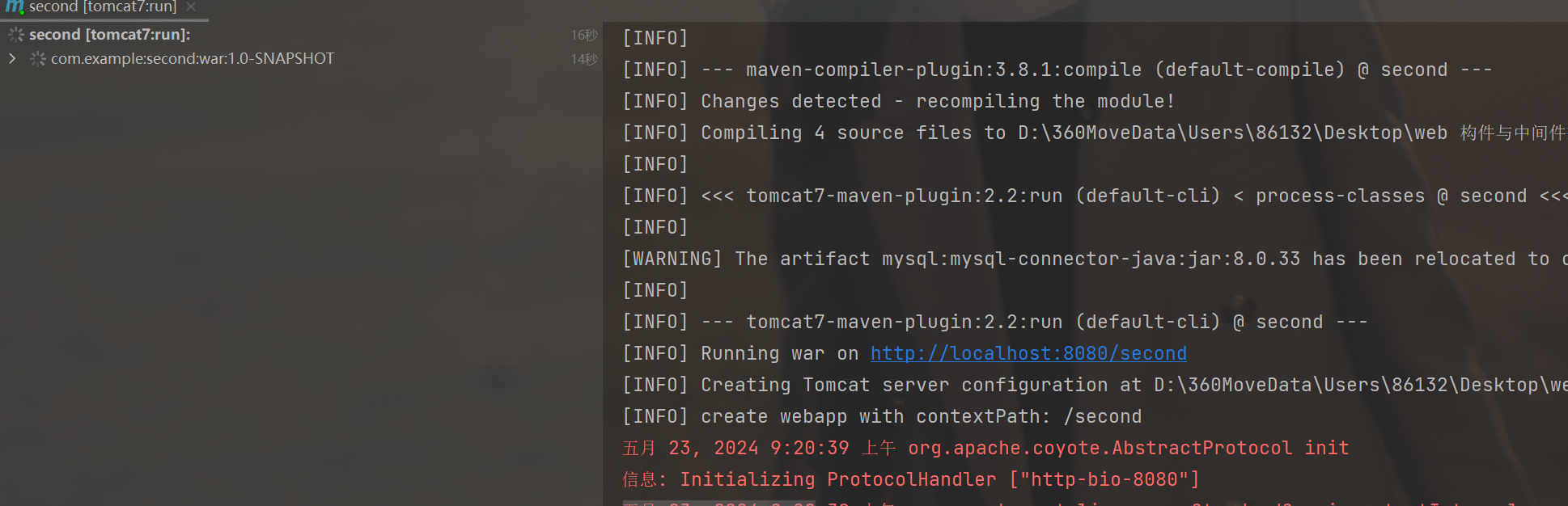
**功能实现：**创建一个Web工程，允许用户通过输入学生的学号，查询并展示该学生的相关信息，通过JSP页面展示结果。

**环境配置：**Win10 + IntelliJ IDEA 2021；本地Java开发环境和Tomcat服务器；MySQL。

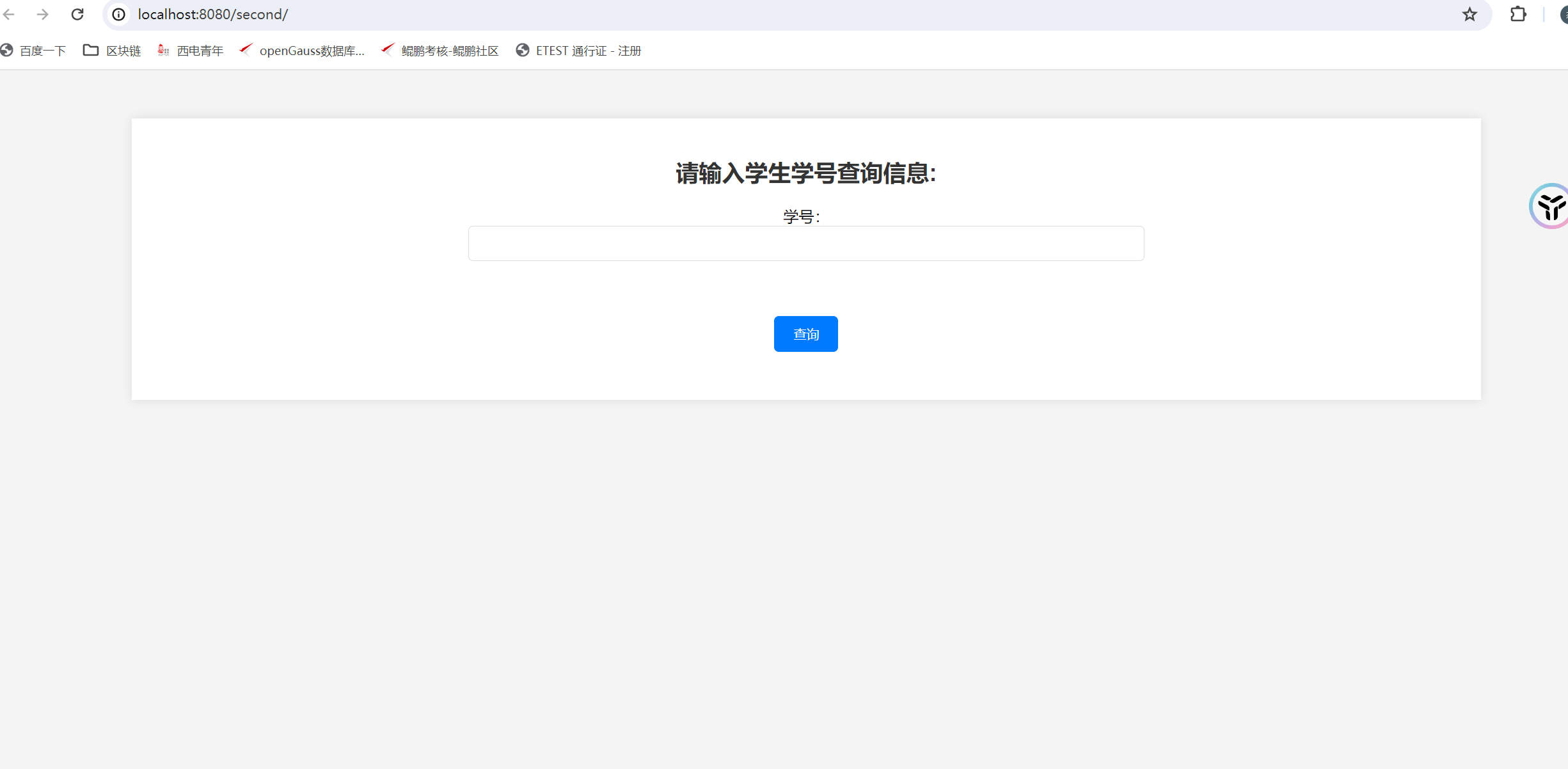
1. **实验过程**
2. **项目初始化** ：首先，在IntelliJ IDEA中创建一个新的Java Web项目，并确保项目配置支持Servlet和JSP。同时，设置项目连接本地的MySQL数据库。
3. **数据库设计** ：在MySQL中构建学生信息表`students`，至少包含`id`(学号, 主键), `name`(姓名), `classname`(班级)字段。
4. **Servlet编写** ：编写核心Servlet类`StuInfoServlet`，继承自`HttpServlet`。通过`doGet`方法处理来自用户的查询请求，获取学号参数，调用`StuInfoService`服务层接口查询学生信息。
5. **服务层实现** ：实现`StuInfoService`接口，定义`selectById`方法，使用JDBC连接数据库，执行SQL查询语句获取指定学号的学生信息。
6. **JSP设计：**创建`index.jsp`作为主界面，包含一个表单用于输入学生学号和提交按钮。当表单提交后，通过`action`属性指向`StuInfoServlet`进行处理。

**代码示例：**

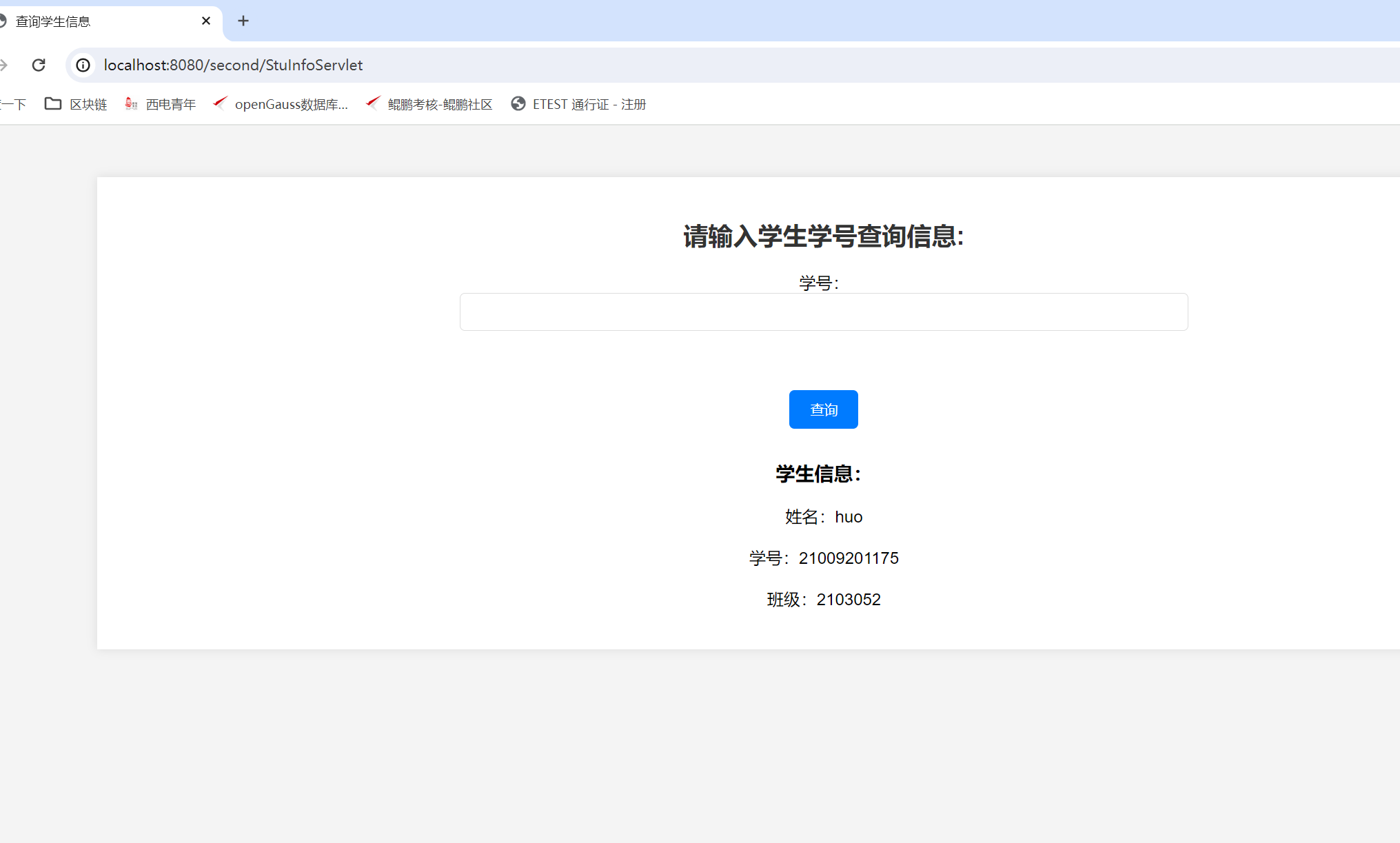
1. @WebServlet("/StuInfoServlet")
2. public class StuInfoServlet extends HttpServlet {
3. protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
4. doGet(request, response);
5. }
6. protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
7. String studentId = request.getParameter("studentId");
8. System.out.println("接收参数："+ studentId);
9. StuInfoService service = new StuInfoService();
10. Stu studentInfo = null;
11. try {
12. studentInfo = service.selectById(studentId);
13. } catch (SQLException e) {
14. e.printStackTrace();
15. }
16. RequestDispatcher dispatcher;
17. if (studentInfo.getName() != null) {
18. *// 查询成功，将学生信息设置为request属性，转发回JSP页面显示*
19. request.setAttribute("studentName", studentInfo.getName());
20. request.setAttribute("studentId", studentId);
21. request.setAttribute("studentClassName", studentInfo.getClassname());
22. System.out.println("成功查询"+ studentInfo.getName());
23. *// 添加更多属性以展示更多信息*
24. dispatcher = request.getRequestDispatcher("index.jsp");
25. dispatcher.forward(request, response);
26. } else {
27. *// 查询失败，可以设置错误消息并转发回JSP页面显示*
28. request.setAttribute("errorMessage", "未找到该学号的学生信息！");
29. dispatcher = request.getRequestDispatcher("index.jsp");
30. dispatcher.forward(request, response);
31. }
32. }
33. }
34. **实验结果**
35. **启动后端：**



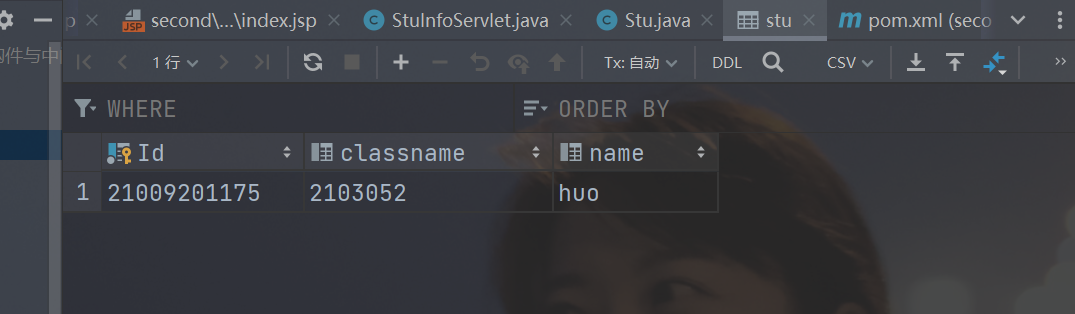
1. **访问项目**



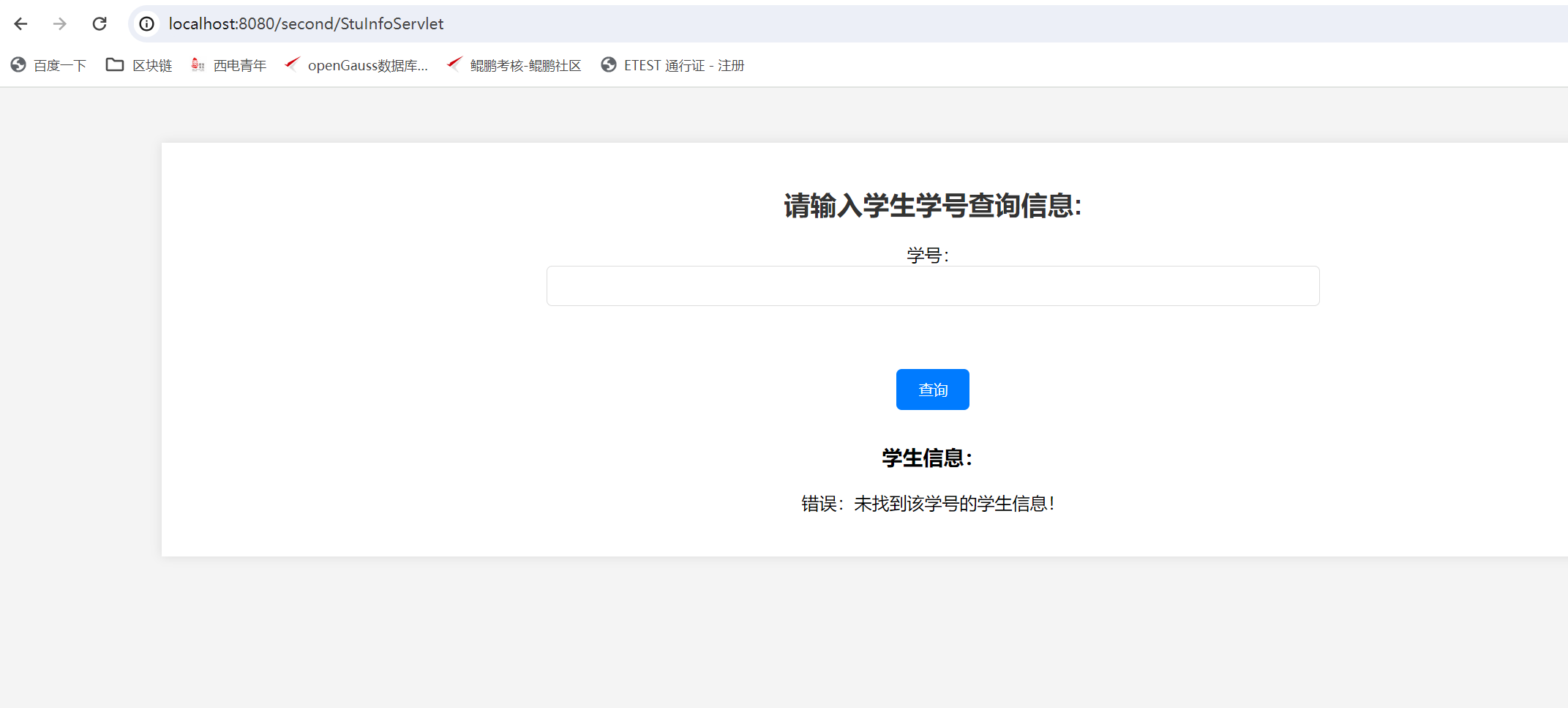
1. **测试功能**



1. **数据库**



1. **错误处理**



1. **总结建议**

通过本次实验，我们成功地实现了一个简易的学生信息查询系统，不仅巩固了Java Web开发的基础知识，还加深了对数据库操作及前后端交互流程的理解。实验过程中，我们亲身体验了从需求分析、设计数据库、编写后端逻辑到前端页面展示的完整Web开发流程，为后续复杂Web应用的开发打下了坚实的基础。