**哈尔滨工业大学计算机科学与技术学院**

**2016年秋季学期《软件工程》**

**Lab 2：MVC编程与云平台部署**

|  |  |  |
| --- | --- | --- |
| **姓名** | **学号** | **联系方式** |
| 张翔熙 | 6142910605 | 15645160847 |
| 实验结果在SAE的访问地址（URL） | | http://libuzi.applinzi.com/ |

**目 录**

[1 实验要求 1](#_Toc387752290)

[2 开发环境配置 1](#_Toc387752291)

[2.1 在Eclipse中配置Struts2 1](#_Toc387752292)

[2.2 在Eclipse中配置MySQL 1](#_Toc387752293)

[2.3 在Eclipse中配置Tomcat 1](#_Toc387752294)

[3 图书SaaS设计 1](#_Toc387752295)

[3.1 Web.xml 1](#_Toc387752296)

[3.2 Struts.xml 1](#_Toc387752297)

[3.3 Action类 2](#_Toc387752298)

[3.4 辅助类 2](#_Toc387752299)

[3.5 JSP页面 2](#_Toc387752300)

[3.6 各Action/JSP之间的调用和消息传递关系 2](#_Toc387752301)

[4 图书SaaS核心代码 2](#_Toc387752302)

[4.1 按作者查询 2](#_Toc387752303)

[4.2 展示图书详细信息 3](#_Toc387752304)

[4.3 删除图书 3](#_Toc387752305)

[4.4 新增图书/作者(可选) 3](#_Toc387752306)

[4.5 更新图书信息(可选) 3](#_Toc387752307)

[4.6 数据库连接与访问 3](#_Toc387752308)

[5 图书SaaS的SAE部署 3](#_Toc387752309)

[5.1 步骤 3](#_Toc387752310)

[5.2 外在访问结果 3](#_Toc387752311)

[6 计划与实际进度 3](#_Toc387752312)

[7 小结 4](#_Toc387752313)

[文档全部完成之后，请更新上述区域]

# 实验要求

使用MySQL建立一个“图书数据库”BookDB，包含两张表：

Book{ISBN(PK), Title, AuhorID(FK), Publisher, PublishDate, Price};

Author{AuthorID(PK), Name, Age, Country};

网站满足要求：

输入作者名字，查询该作者的全部书目；

当用户点击某本图书的题目时，展示图书详细信息和作者详细信息；

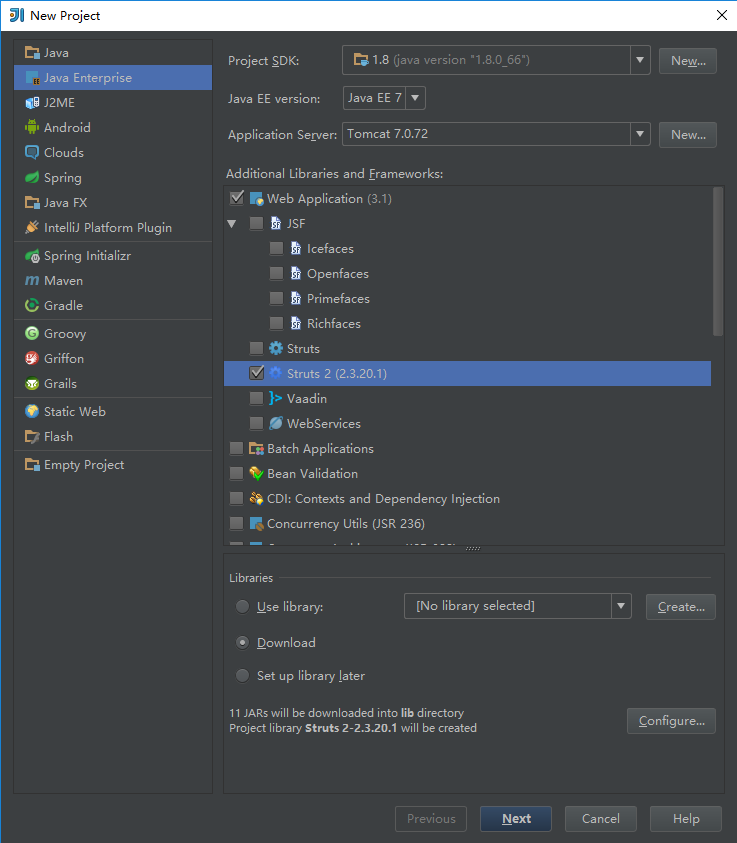
当用户点击“删除”按钮时，将对应行的图书从数据表中删除。

网站需要在SAE上上线。

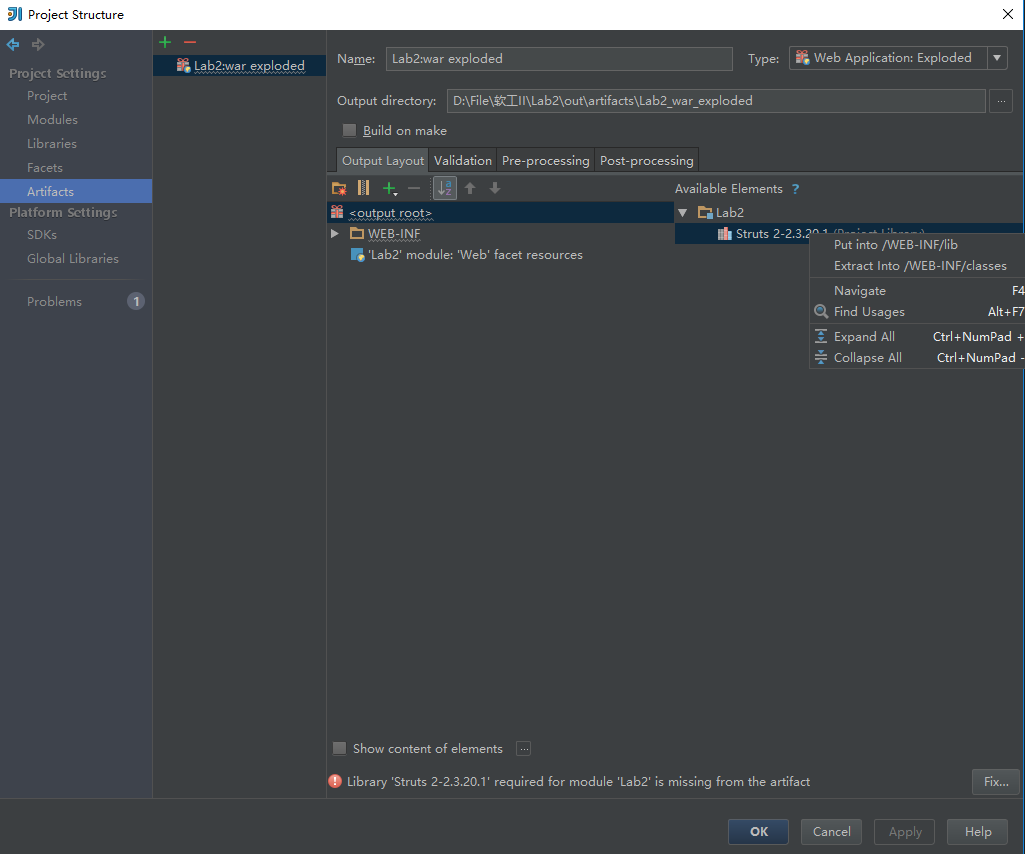
# 开发环境配置

采用屏幕截图加文字说明的方式给出你自己在Eclipse中配置MySQL、Tomcat、Stuts2的各步骤。

## 在Eclipse中配置Struts2



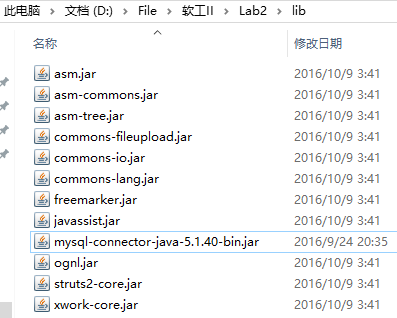
在IDEA创建项目时，选择Java Enterprise下的Struts 2框架，在下设选项中选择Download，即可自动下载struts2框架并集成在项目中。



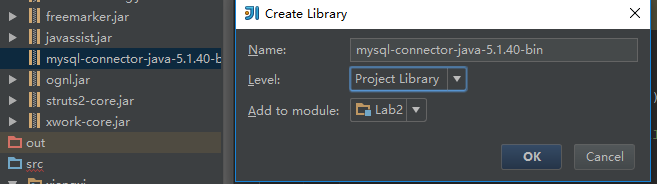
随后，要在ProjectStructure中的Artifacts选项卡下将Struts2右键点选put into /Web-inf/lib，将库文件移动至合适的位置。

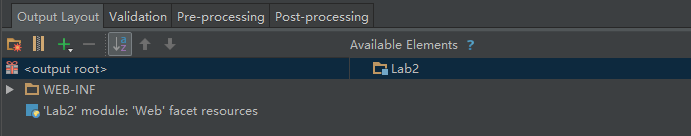
## 在Eclipse中配置MySQL

在官网下载JDBCconnector组件后，将其中的jar文件拷贝到lib路径下：

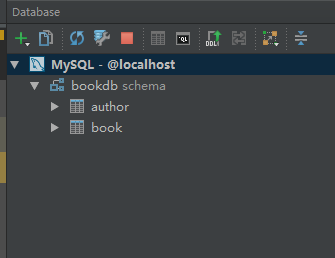


然后在项目中把这个jar文件作为库文件导入，并在project structure中将其植入WEB/INF中，完成JDBC的配置：

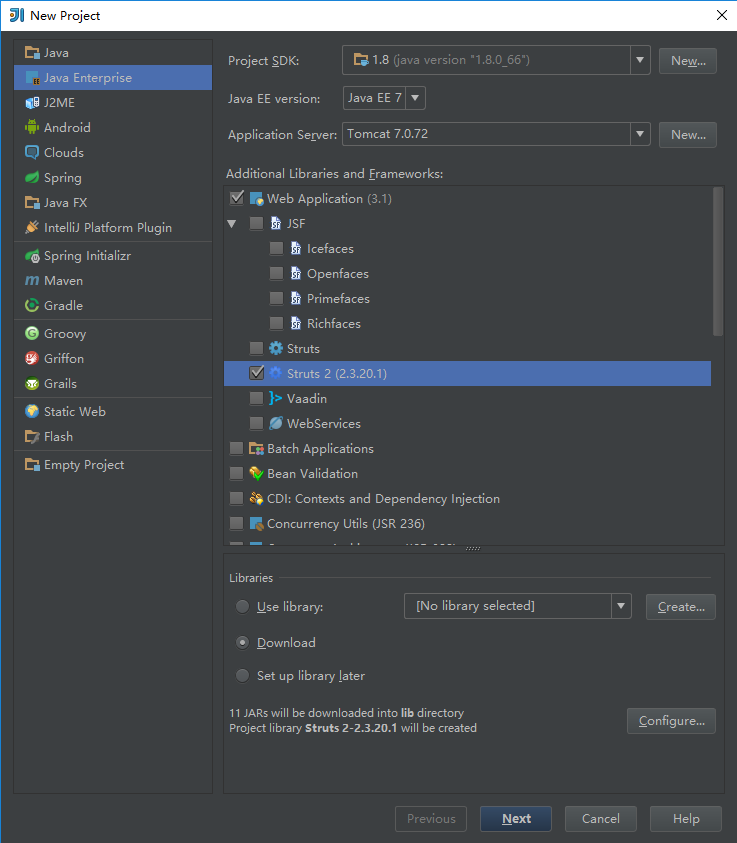




然后在IDEA的Database选项框中点击连接到new database，选择mysql，输入用户名密码后即可成功：



## 在Eclipse中配置Tomcat



在创建项目时，在Application Server中选择Tomcat的安装目录，即可将Tomcat完成配置。

# 图书SaaS设计

## Web.xml

<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app\_3\_1.xsd"

version="3.1">

<filter>

<filter-name>struts2</filter-name>

<filter-class>org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter</filter-class>

</filter>

<filter-mapping>

<filter-name>struts2</filter-name>

<url-pattern>/\*</url-pattern>

</filter-mapping>

</web-app>

Web.xml是项目对外的接口，尤其要注意其中的url-pattern部分，决定了框架接收何种url格式。

## Struts.xml

1. <?xml version="1.0" encoding="UTF-8"?>  
     
   <!DOCTYPE struts PUBLIC  
    "-//Apache Software Foundation//DTD Struts Configuration 2.3//EN"  
    "http://struts.apache.org/dtds/struts-2.3.dtd">  
     
   <struts>  
    <package name="default" extends="struts-default">  
    <action name="search" class="xiangxi.action.Search">  
    <result name="success">/WEB-INF/searchResults.jsp</result>  
    </action>  
    <action name="detailedInfo" class="xiangxi.action.DetailedInfo">  
    <result name="success">  
    /WEB-INF/detailedInfo.jsp  
    </result>  
    </action>  
    <action name="DeleteBook" class="xiangxi.action.DeleteBook">  
    <result name="success">  
    /WEB-INF/seccessfulDeletion.jsp  
    </result>  
    </action>  
    </package>  
   </struts>

这个xml中定义了三个action。第一个action用来完成根据作者名的查询，第二个action用来完成对详细信息的显示，第三个action用来完成对图书进行删除的操作。每个action都分别对应到了它们的action类与结果JSP页面。

## Action类

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | Action类名 | 作用 | 操作列表 | 操作说明 |
| 1 | Search | 完成根据作者名对书籍的搜索 | execute | 调用另一个方法，成功执行后返回success |
| searchByAuthorName | 根据作者名进行sql搜索，返回一组book列表 |
| 2 | DetailedInfo | 完成根据ISBN码对作者信息与书籍信息的调用 | execute | 根据ISBN，进行一个sql查询获得书籍信息，再根据authorID进行查询获得作者信息，成功后返回success |
| 3 | DeleteBook | 完成根据ISBN码对书籍的删除 | execute | 根据ISBN码在数据库中删除书目信息，成功后返回success |

## 辅助类

辅助类为DatabaseConnection，类的作用为根据url、username和password，完成对数据库的链接，对外提供一个Connection类的对象用于完成数据库访问。

## JSP页面

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 页面名 | 作用 | 页面核心元素(form) | Form对应的action name | Form中提交的数据项 |
| 1 | index | 欢迎界面，完成查询 | 书名搜索输入框 | search | authorName |
| 2 | detailedInfo | 显示书目详细信息 | 无form | 无 | 无 |
| 3 | searchResults | 显示书目列表 | 无form，通过url传递信息 | 无 | 无 |
| 4 | successfulDeletion | 显示书目删除信息 | 无form | 无 | 无 |

## 各Action/JSP之间的调用和消息传递关系

# 图书SaaS核心代码

针对下列功能，分别给出Action类和JSP页面中的核心代码。

## 按作者查询

动作类如下：

public class Search extends ActionSupport{  
 private String authorName;  
 private ArrayList<Book> searchResults;  
  
 public String getAuthorName() {  
 return authorName;  
 }  
  
 public void setAuthorName(String authorName) {  
 this.authorName = authorName;  
 }  
  
 public ArrayList<Book> getSearchResults() {  
 return searchResults;  
 }  
  
 public void setSearchResults(ArrayList<Book> searchResults) {  
 this.searchResults = searchResults;  
 }  
 public String execute() {  
 searchByAuthorName();  
 return *SUCCESS*;  
 }  
 private void searchByAuthorName() {  
 Connection connection = null;  
 DatabaseConnection databaseConnection = new DatabaseConnection();  
 try {  
 connection = databaseConnection.*getConnection*();  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 String query = "select \* from book where **AuthorID** in (select **AuthorID** from author where **Name** = '"+authorName+"');";  
 try {  
 PreparedStatement statement = connection.prepareStatement(query);  
 ResultSet resultSet = statement.executeQuery();  
 this.searchResults = new ArrayList<>();  
 while(resultSet.next()) {  
 String ISBN = resultSet.getString(1);  
 String Title = resultSet.getString(2);  
 Integer AuthorID = resultSet.getInt(3);  
 String Publisher = resultSet.getString(4);  
 String PublishDate = resultSet.getString(5);  
 String Price = resultSet.getString(6);  
 searchResults.add(new Book(ISBN, Title, AuthorID, Publisher, PublishDate, Price));  
 }  
 } catch(SQLException e) {  
 e.printStackTrace();  
 }  
 }  
}

对应JSP页面有两个，搜索框页面与搜索结果，搜索框页面如下：

<html>  
 <head>  
 <title>IZU图书管理系统</title>  
 </head>  
 <body>  
 </div>  
 <center>  
 <h1 style="text-align: center">欢迎来到IZU图书管理系统</h1>  
 <div style="margin-left:auto;margin-right:auto">  
 <**s:form** action="search" cssStyle="align-content: center">  
 <**s:textfield** name="authorName" label="作者名" cssStyle="align-self: center"></**s:textfield**>  
 <**s:submit** value="搜索" cssStyle="align-self: center"></**s:submit**>  
 </**s:form**>  
 </div>  
 </center>  
 </body>  
</html>

搜索结果如下：

<head>  
 <title>  
 IZU图书管理系统搜索结果  
 </title>  
</head>  
<body>  
 <center>  
 <div>  
 作者：<**s:property** value="authorName" default="xiangxi"/>  
 </div>  
 <**s:if** test="searchResults.size()==0">  
 没有此作者的相关书目。  
 </**s:if**>  
 <**s:iterator** value="searchResults">  
 <ul>  
 <li>  
 <div>  
 <h4>  
 <**s:property** value="Title"></**s:property**>  
 <a href="detailedInfo.action?ISBN=<**s:property** value='ISBN'/>" target="\_blank">  
 显示更多信息  
 </a>  
 </h4>  
 </div>  
 </li>  
 </ul>  
 </**s:iterator**>  
 <li>  
 <**s:a** href="/">返回首页</**s:a**>  
 </li>  
 </center>  
</body>

## 展示图书详细信息

在搜索结果中，点击任何一本书都会进入到数目的详细信息中。Action类如下：

public class DetailedInfo extends ActionSupport{  
 private String ISBN;  
 private String Title;  
  
 public String getTitle() {  
 return Title;  
 }  
  
 public void setTitle(String title) {  
 Title = title;  
 }  
  
 public String getCountry() {  
 return Country;  
 }  
  
 public void setCountry(String country) {  
 Country = country;  
 }  
  
 public Integer getAge() {  
 return Age;  
 }  
  
 public void setAge(Integer age) {  
 Age = age;  
 }  
  
 public String getName() {  
 return Name;  
 }  
  
 public void setName(String name) {  
 Name = name;  
 }  
  
 public Integer getAuthorID() {  
 return AuthorID;  
 }  
  
 public void setAuthorID(Integer authorID) {  
 AuthorID = authorID;  
 }  
  
 private Integer AuthorID;  
 private String Publisher;  
 private String PublishDate;  
 private String Price;  
 private String Name;  
 private Integer Age;  
 private String Country;  
  
 public String getISBN() {  
 return ISBN;  
 }  
  
 public void setISBN(String ISBN) {  
 this.ISBN = ISBN;  
 }  
  
 public String getPublisher() {  
 return Publisher;  
 }  
  
 public void setPublisher(String publisher) {  
 Publisher = publisher;  
 }  
  
 public String getPublishDate() {  
 return PublishDate;  
 }  
  
 public void setPublishDate(String publishDate) {  
 PublishDate = publishDate;  
 }  
  
 public String getPrice() {  
 return Price;  
 }  
  
 public void setPrice(String price) {  
 Price = price;  
 }  
  
 public String execute() {  
 Connection connection = null;  
 DatabaseConnection databaseConnection = new DatabaseConnection();  
 try {  
 connection = databaseConnection.*getConnection*();  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 String query = "select \* from book where **ISBN** = '"+ISBN+"';";  
 try {  
 PreparedStatement statement = connection.prepareStatement(query);  
 ResultSet resultSet = statement.executeQuery();  
 if(resultSet.next()) {  
 Title = resultSet.getString(2);  
 AuthorID = resultSet.getInt(3);  
 Publisher = resultSet.getString(4);  
 PublishDate = resultSet.getString(5);  
 Price = resultSet.getString(6);  
 }  
 System.*out*.println(AuthorID);  
 query = "select \* from author WHERE **AuthorID** = '" + Integer.*toString*(AuthorID) + "';";  
 System.*out*.println(query);  
 statement = connection.prepareStatement(query);  
 resultSet = statement.executeQuery();  
 if (resultSet.next()) {  
 Name = resultSet.getString(2);  
 Age = resultSet.getInt(3);  
 Country = resultSet.getString(4);  
 }  
 } catch(SQLException e) {  
 e.printStackTrace();  
 }  
 return *SUCCESS*;  
 }  
}

在类中完成了对数据库的访问查询工作。对应JSP页面如下：

<body>  
<center>  
 <div style="margin-left: auto;margin-right: auto;">  
 <ul>  
 <li>  
 书名：<**s:property** value="Title"></**s:property**>  
 </li>  
 <li>  
 作者编号：<**s:property** value="AuthorID"></**s:property**>  
 </li>  
 <li>  
 作者：<**s:property** value="Name"></**s:property**>  
 </li>  
 <li>  
 年龄：<**s:property** value="Age"></**s:property**>  
 </li>  
 <li>  
 国籍：<**s:property** value="Country"></**s:property**>  
 </li>  
 <li>  
 书号：<**s:property** value="ISBN"></**s:property**>  
 </li>  
 <li>  
 出版社：<**s:property** value="Publisher"></**s:property**>  
 </li>  
 <li>  
 出版日期：<**s:property** value="PublishDate"></**s:property**>  
 </li>  
 <li>  
 价格：<**s:property** value="Price"></**s:property**>  
 </li>  
 <div style="margin-right: auto;margin-right: auto">  
 <li>  
 <**s:a** href="/">返回首页</**s:a**>  
 </li>  
 <li>  
 <a href="/DeleteBook.action?ISBN=<**s:property** value='ISBN'/>">  
 删除本书  
 </a>  
 </li>  
 </div>  
 </ul>  
 </div>  
 </center>  
</body>

## 删除图书

在Action类中完成了对数据库的修改，代码如下：

public class DeleteBook extends ActionSupport{  
 private String ISBN;  
  
 public String getISBN() {  
 return ISBN;  
 }  
  
 public void setISBN(String ISBN) {  
 this.ISBN = ISBN;  
 }  
 public String execute() {  
 Connection connection = null;  
 DatabaseConnection databaseConnection = new DatabaseConnection();  
 try {  
 connection = databaseConnection.*getConnection*();  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 String deletion = "DELETE FROM book WHERE **ISBN** = '" + ISBN + "';";  
 try {  
 Statement statement = connection.createStatement();  
  
 statement.execute(deletion);  
 } catch(SQLException e) {  
 e.printStackTrace();  
 }  
 return *SUCCESS*;  
 }  
}

而对应显示了一个删除成功的JSP页面，如下：

<html>  
<head>  
 <title>IZU图书管理系统</title>  
</head>  
<body>  
<center>  
 <li>  
 <h2 style="align-left:auto; align-right: auto;">  
 已成功删除！  
 </h2>  
 </li>  
 <li>  
 <h4>  
 <**s:a** href="/">  
 返回首页  
 </**s:a**>  
 </h4>  
 </li>  
 </center>  
</body>  
</html>

## 数据库连接与访问

数据库通过JDBC进行联接，对应的类如下：

public class DatabaseConnection {  
 private static Connection *connection*;  
 public static Connection getConnection() throws SQLException{  
 try {  
 String url = "jdbc:mysql://w.rdc.sae.sina.com.cn:3307/app\_libuzi";  
 String username = "\*\*\*\*\*\*\*\*\*";  
 String password = "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";  
 Driver foo = new Driver();  
 *connection* = DriverManager.*getConnection*(url, username, password);  
 System.*out*.println("Successfully connected to the database");  
 } catch (Exception e) {  
 System.*out*.println("Failed to connect the database");  
 e.printStackTrace();  
 }  
 return *connection*;  
 }  
}

（其中的用户名与密码在项目中为SAE提供的用户名密码，此处未写出）

# 图书SaaS的SAE部署

## 步骤

1. 

在新浪云页面右上角，选择用微博登录，即完成了用户注册。

2.



在左上方，选择控制台中的云应用；

3.



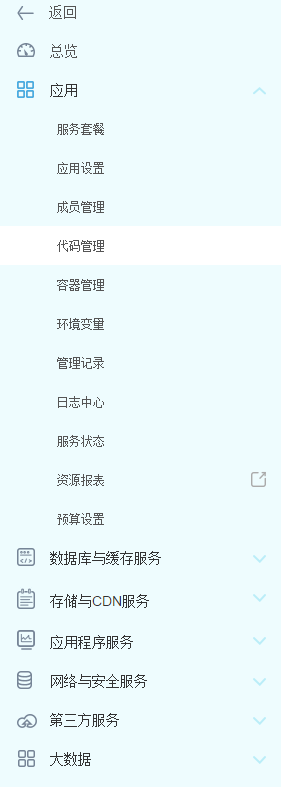
在应用管理处，选择创建新应用。

4.



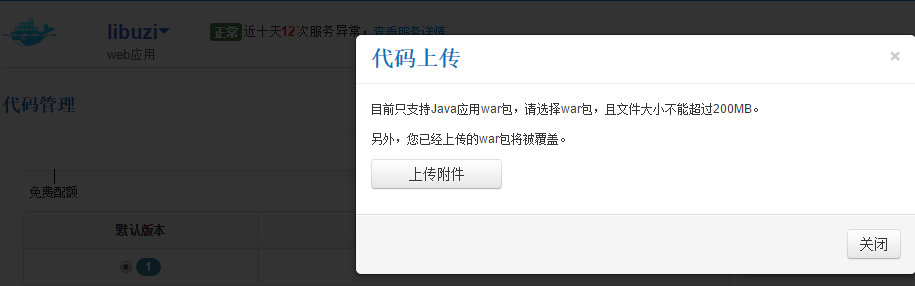
选择JAVA开发语言以及其他选项，点击右侧的创建应用。

5.



随后进入云应用控制台中的代码管理部分。

6.



在代码管理中上传war包，网站即可上线运行。

7.



在数据库中选择共享型MySQL，创建数据库与数据表，即可上线工作。

## 外在访问结果

给出访问SAE上的图书SaaS的屏幕截图结果，要展示出核心功能。截图要带浏览器标题栏和地址栏。

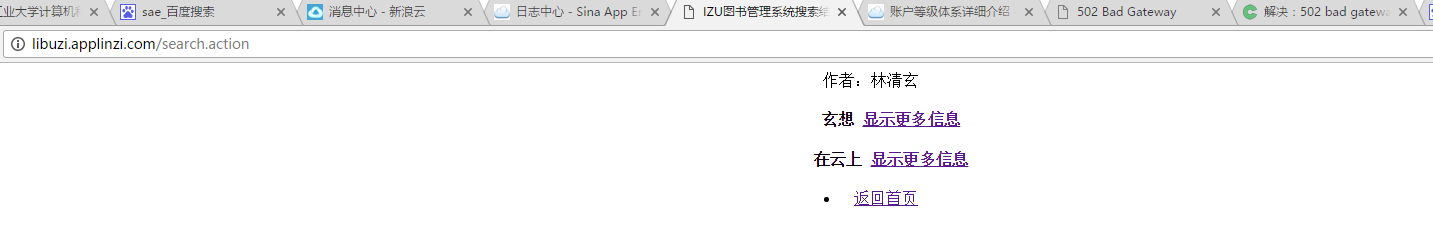
1. 首页

首页提供了按照作者名对图书的搜索：

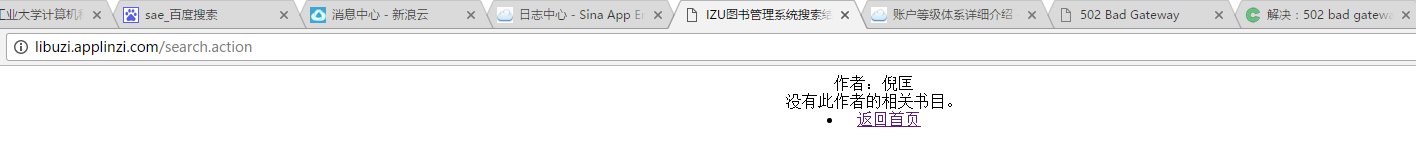


2. 搜索结果

对于作者的每一本书，都列出了书名与显示详细信息的链接与返回首页的链接。

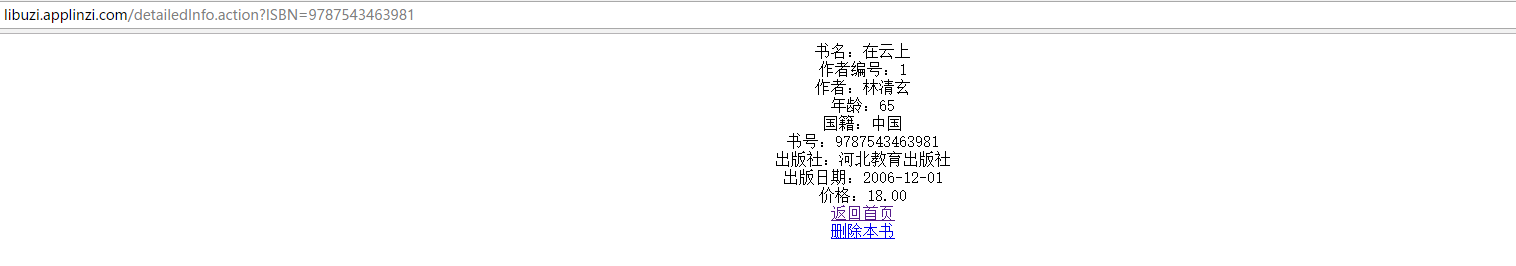


对于书目中没有的作者，给出正确的提示信息。



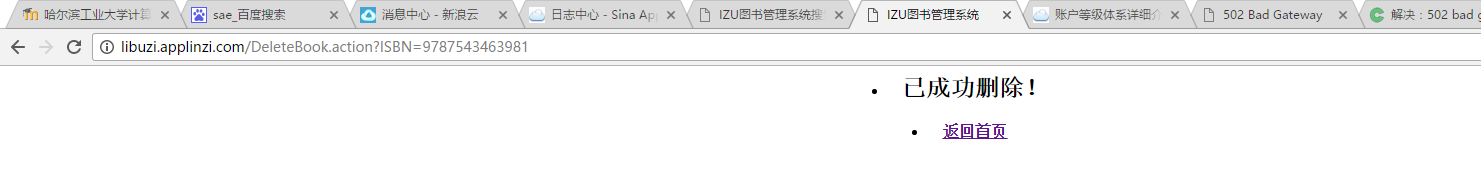
3.

对于每本书可以给出详细信息与删除选项。



4.

成功删除后，给出反馈信息



返回首页再次查询，可见该书目已被删除。



# 计划与实际进度

|  |  |  |  |
| --- | --- | --- | --- |
| 任务编号 | 计划时间长度（分钟） | 实际耗费时间（分钟） | 提前或延期的原因分析 |
| 1 | 40 | 30 | 由于使用了IDEA，项目的配置比预想简单 |
| 2 | 120 | 150 | 对MySQL与JDBC不熟悉，在Struts联接MySQL时出现了困难 |
| 3 | 90 | 120 | 对于在Struts2中使用url进行页面跳转不熟悉，走了一些弯路 |
| 4 | 30 | 50 | 上线SAE时遇到了困难，原因是JDBC版本与SAE的Tomcat版本不兼容 |

# 小结

本次实验中，我完成了一个简单的JSP网站的编写与上线运行。在此前，我没有过任何Web开发的经历与经验，所以这次实验对于我来说是一个比较大的挑战。为了完成此次实验，我详细地学习了Struts2的工作原理与使用方法，在没有经验的情况下自己解决了诸如项目路径配置错误、SQL联接错误等问题，其中的过程相当艰辛。在后台方面，我对不同Action之间的数据传递方式缺少了解，走了很多弯路，最后通过url的方式解决了在不同action之间传递信息的问题；在前端方面，我自学HTML与CSS，完成了页面的设计。

尽管最后的成品看起来效果比较简陋，但是也完成了项目的所有基础要求，对于初次接触WEB开发的我而言已经是很满意了。有了这一次的经验，以后应当会有更好的表现。