

VCB

# 目录

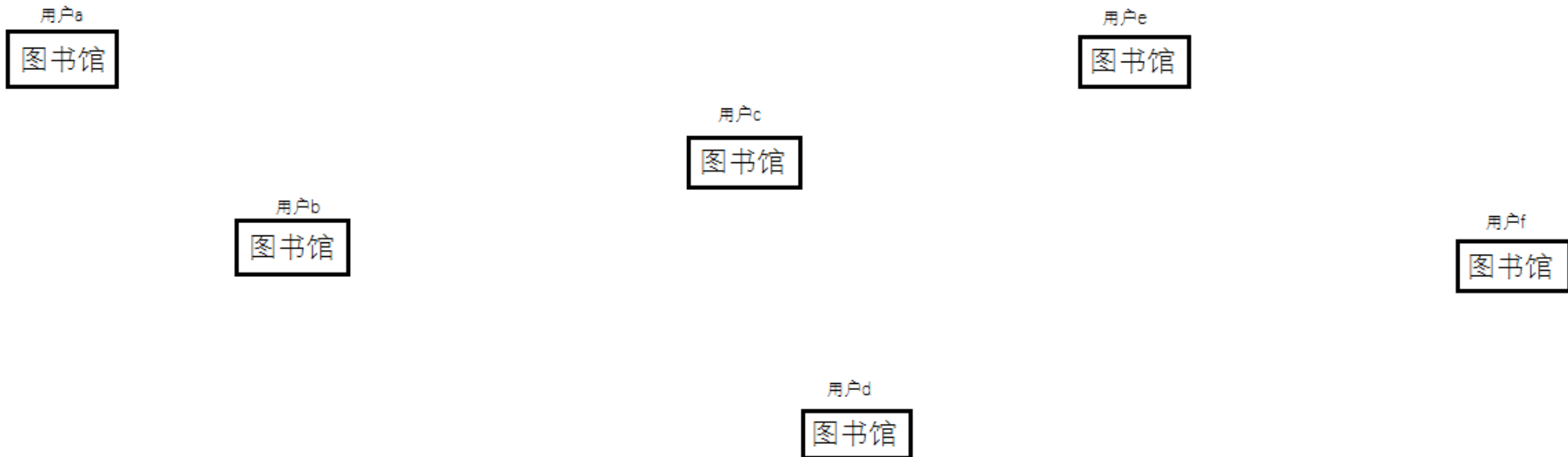
概述

显示效果

设计&细节

# 概述

人人都是图书馆，每个人都可以将自己的书分享出来借给别人，可以通过借书来认识更多人。



显示效果

# register.jsp

## 注册页

注册需填默认地点和经纬度，一个用户可以有多个地点(没有实现增加地点及设置地点为默认地点，但内部对多个地点进行了支持，一本书主要是跟地点绑定的而不是用户)。

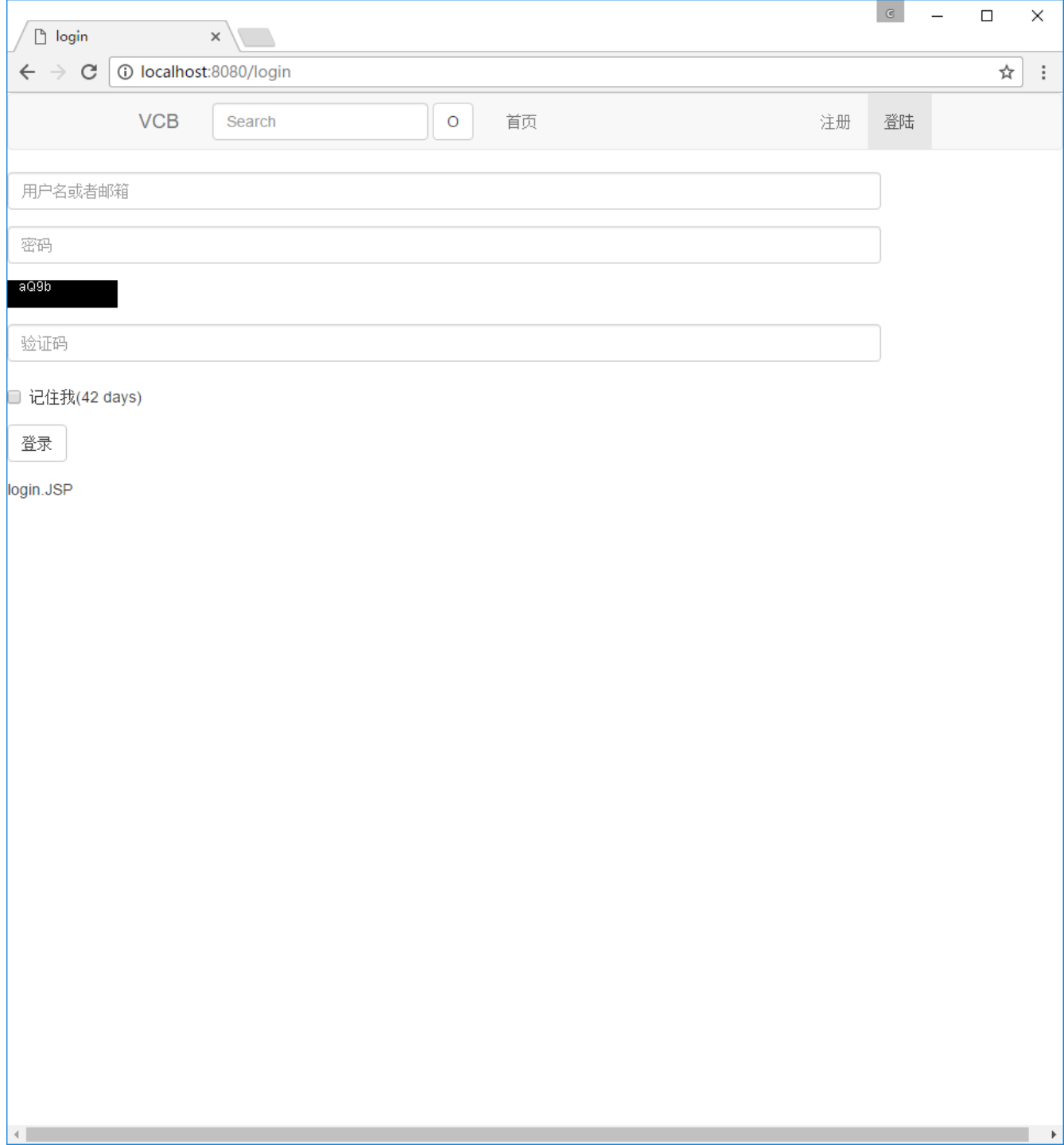
会检查数据库中是否存在表单中的用户名和密码，并在服务端检查是否符合规则，不用另外编写javascript检查代码。



The screenshot shows a web browser window with the address bar displaying 'localhost:8080/register'. The page has a header with 'VCB' and a search bar. Below the header, there are several input fields for registration: '昵称(长度:1-16)', '用户名(长度:6-16)', '邮箱(长度:5+)', '密码(长度:6-16)', '默认地点名称', '默认地点纬度(-90<x<90)', and '默认地点经度(-180<x<180)'. There is a blacked-out section labeled 'PCID'. Below these fields is a 'CAPTCHA' field and a '注册' (Register) button. The footer of the page displays 'register.JSP'.

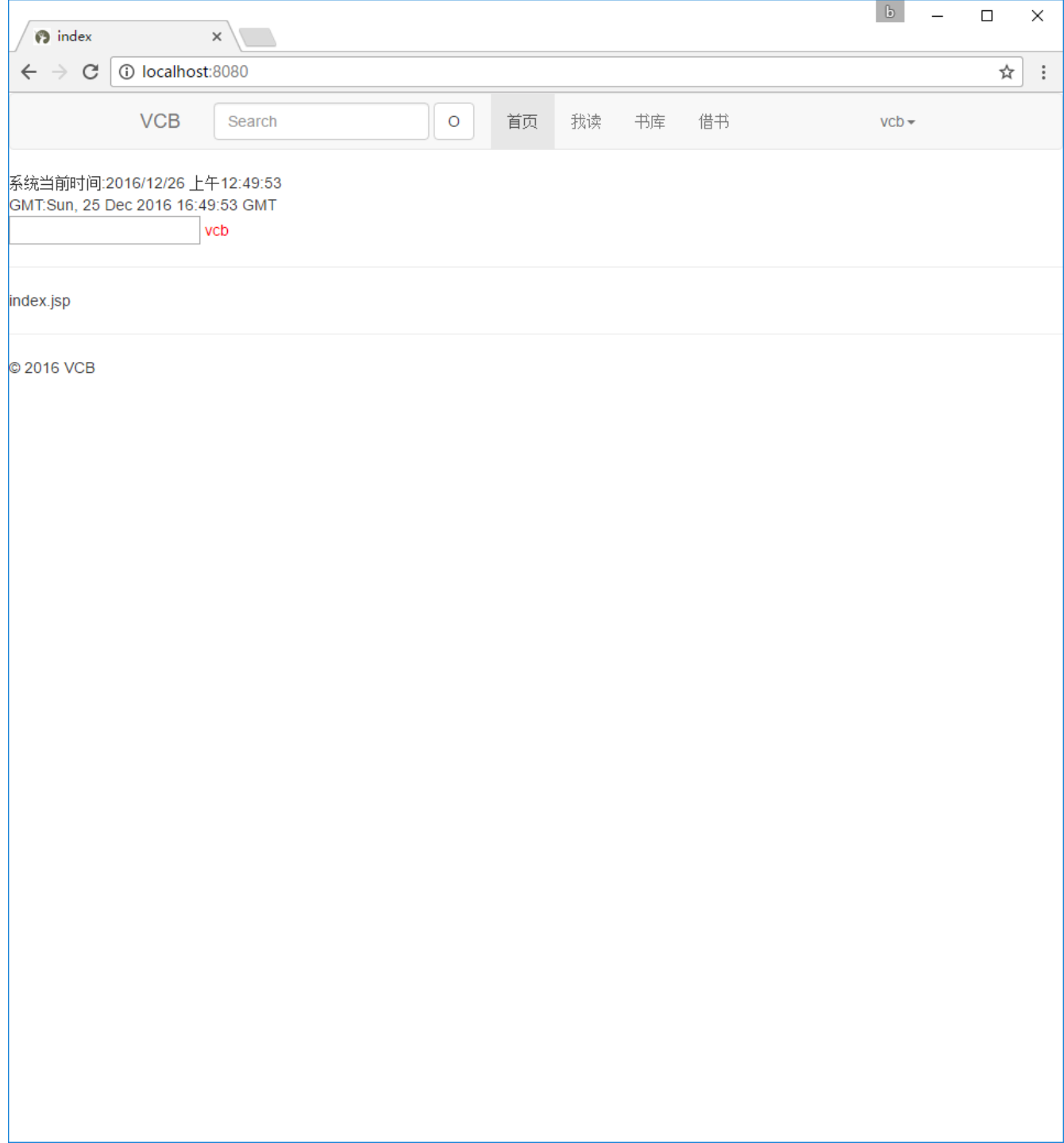
# login.jsp

是否含有@来判断用何种方式登录



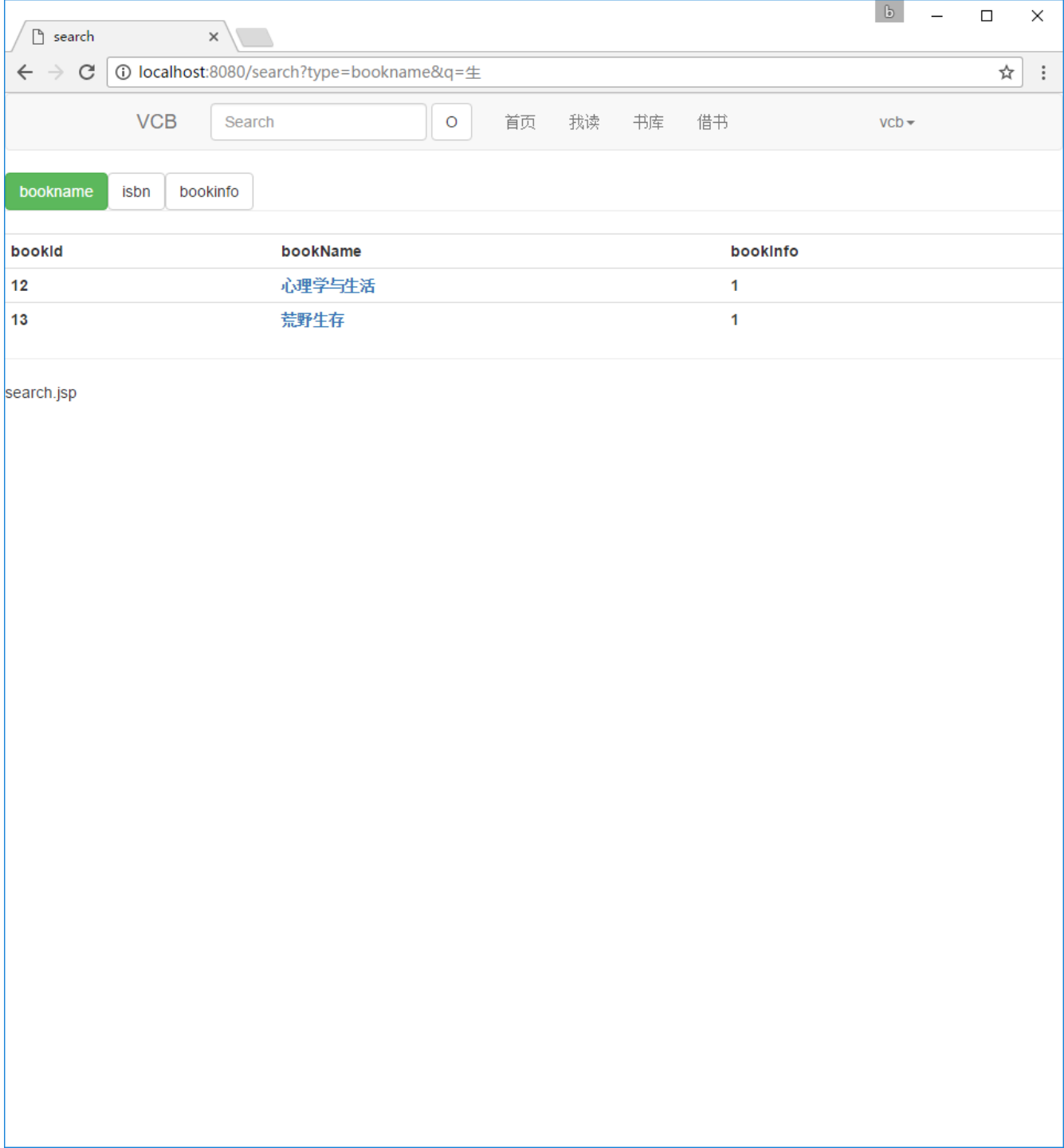
# index.jsp

首页



# search.jsp-bookname

显示含有搜索关键词的书





# book.jsp

显示book的具体信息

拥有按钮不为绿时说明用户的默认地点并没有此书，但其他地点可能有这书(一般用户只有一个地点)，当点击拥有时，会在stacks表中插入一行(绑定的是默认地点id)。如果按钮为绿，按下则在stacks表中删除一行。此按钮使用了ajax。

通过isbn借书按钮，按下会跳转到搜索页(search.jsp-isbn)

通过bookinfo借书按钮，按下会跳转到搜索页(search.jsp-bookinfo)

想读在读读过删除按钮均未实现

数据库中并没有此书的作者和译者信息所以显示为空白

此书bookinfo为1，书默认为1，所以显示为未知，为1时不能通过bookinfo借书，更多bookinfo的说明会在下面提到。



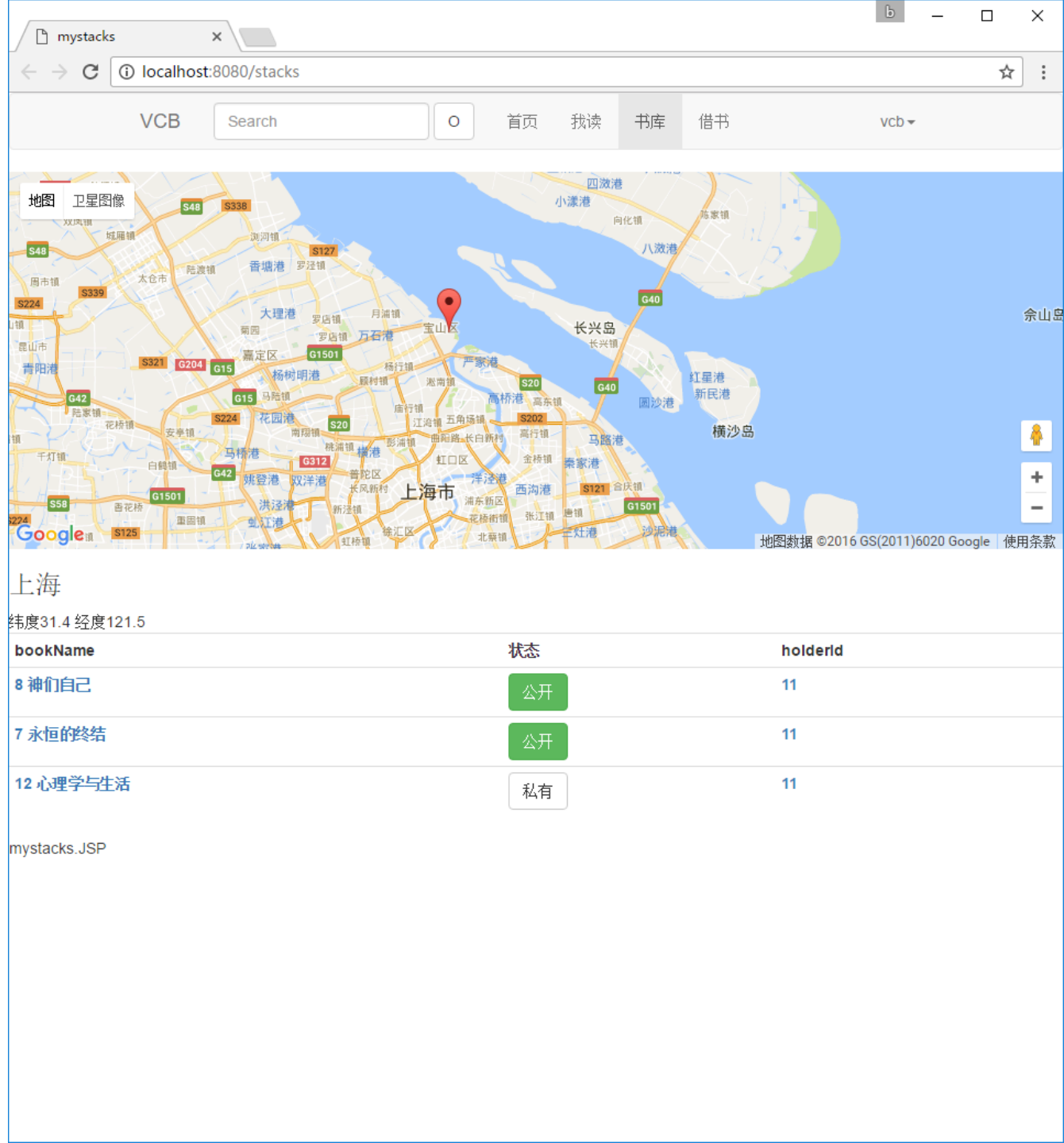
# mystacks.jsp

显示我的书库

当对那本书进行拥有操作后，我的书库就有了这本书

此页会把用户的所有地点显示出来，并可在谷歌地图(有可能要科学上网)上看到

状态显示为公开说明此书别人可以借阅，私有则别人看你的书库看不到此书，按下按钮则改变状态，修改了stacks.isLoan。这个按钮使用了ajax

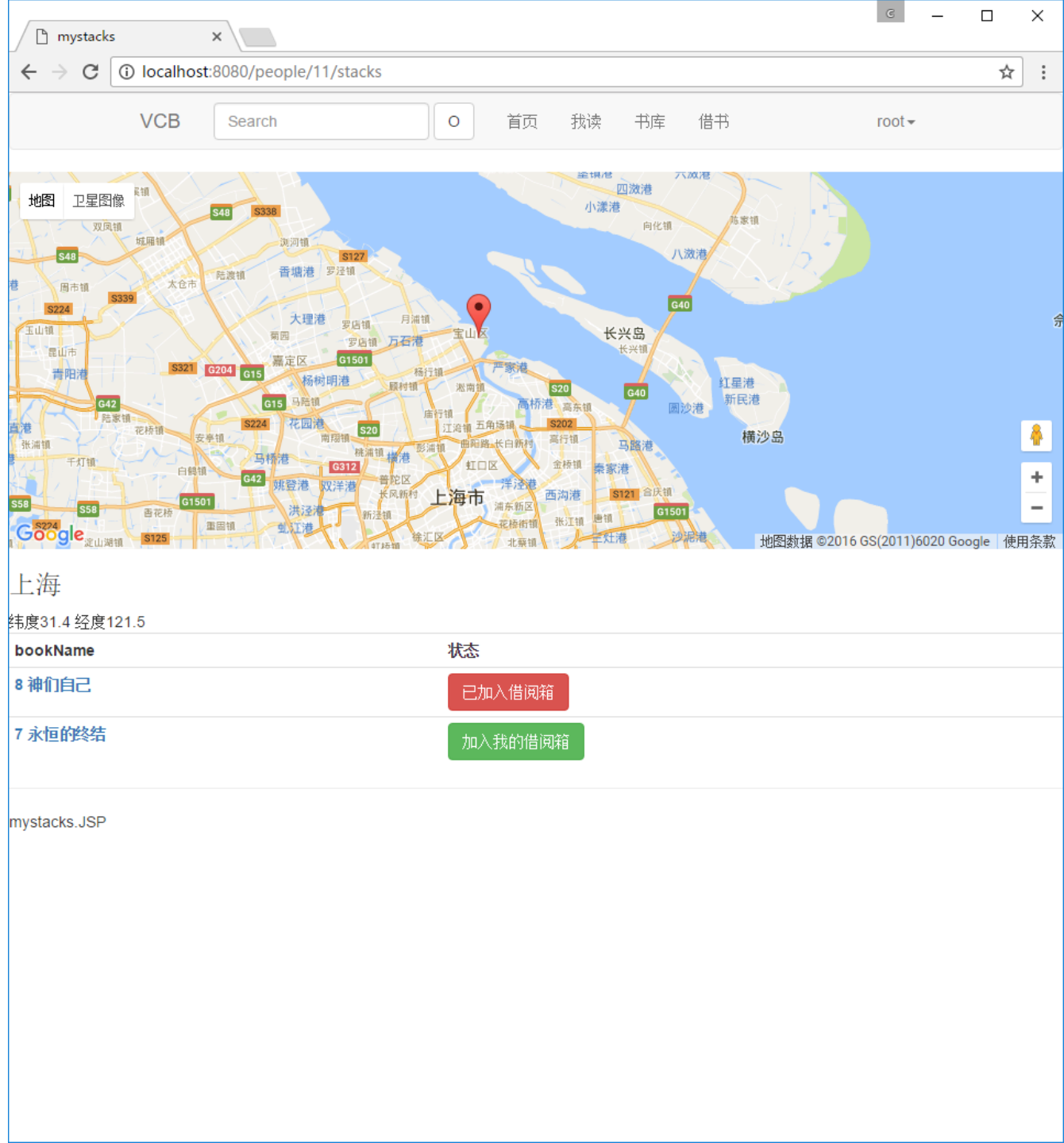


# peoplestacks.jsp

切换到root用户查看刚才用户的书库

可以看到那本状态为私有的书并没有在此页显示

按下加入我的借阅箱按钮可以将此用户的这本书加入到借阅箱(相当于购物车)内，按下已加入借阅箱按钮，则从借阅箱删除此书。这个按钮使用了ajax

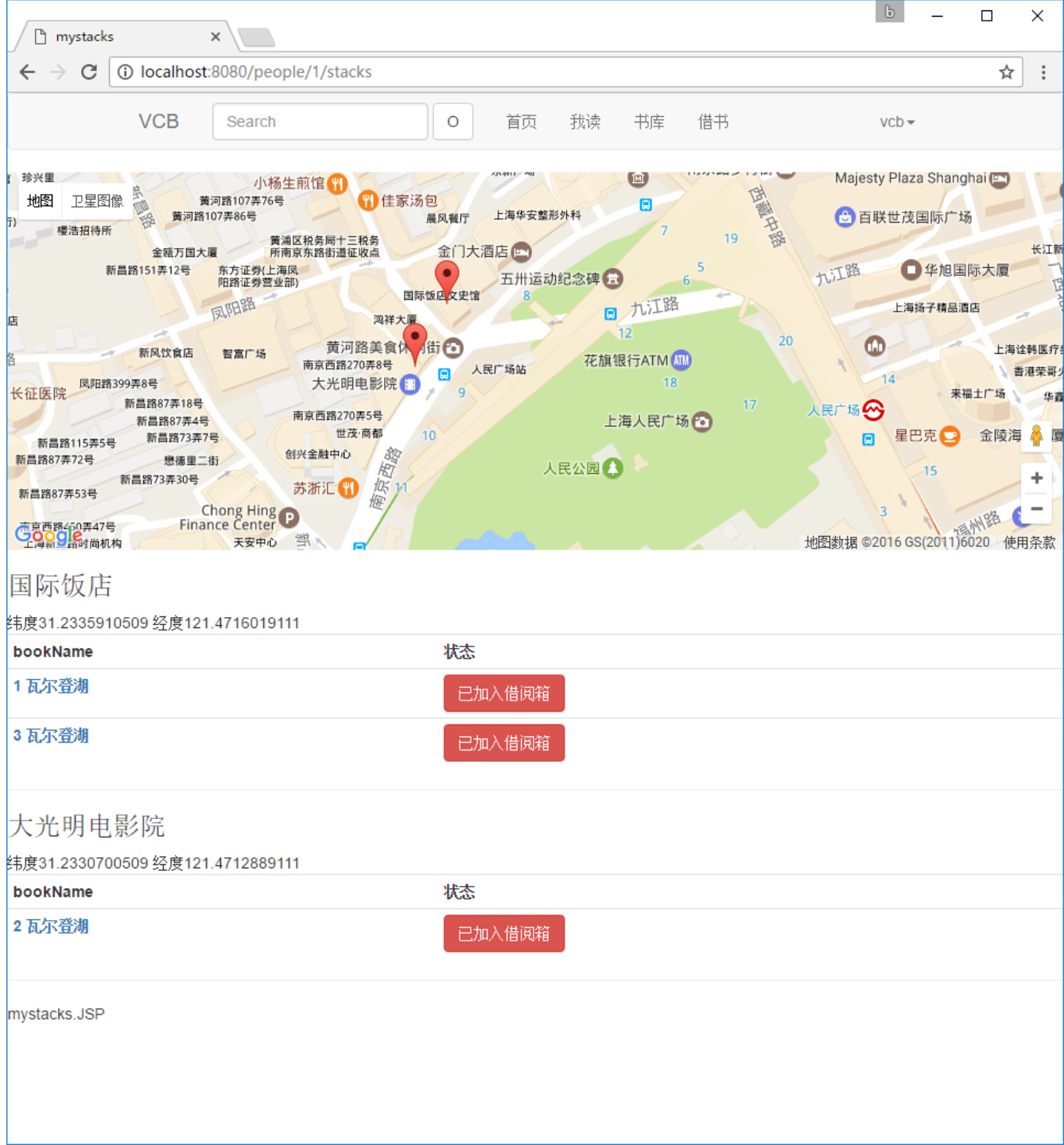


切换回原来的用户，浏览root的书库，并将书都加入到借阅箱内

userId为1的用户在数据库中有多个地点，所以在此页显示了多个地点，而增加地点和设置默认地点没有实现，所以其他用户只有一个地点，但系统是支持一个用户有多个地点的

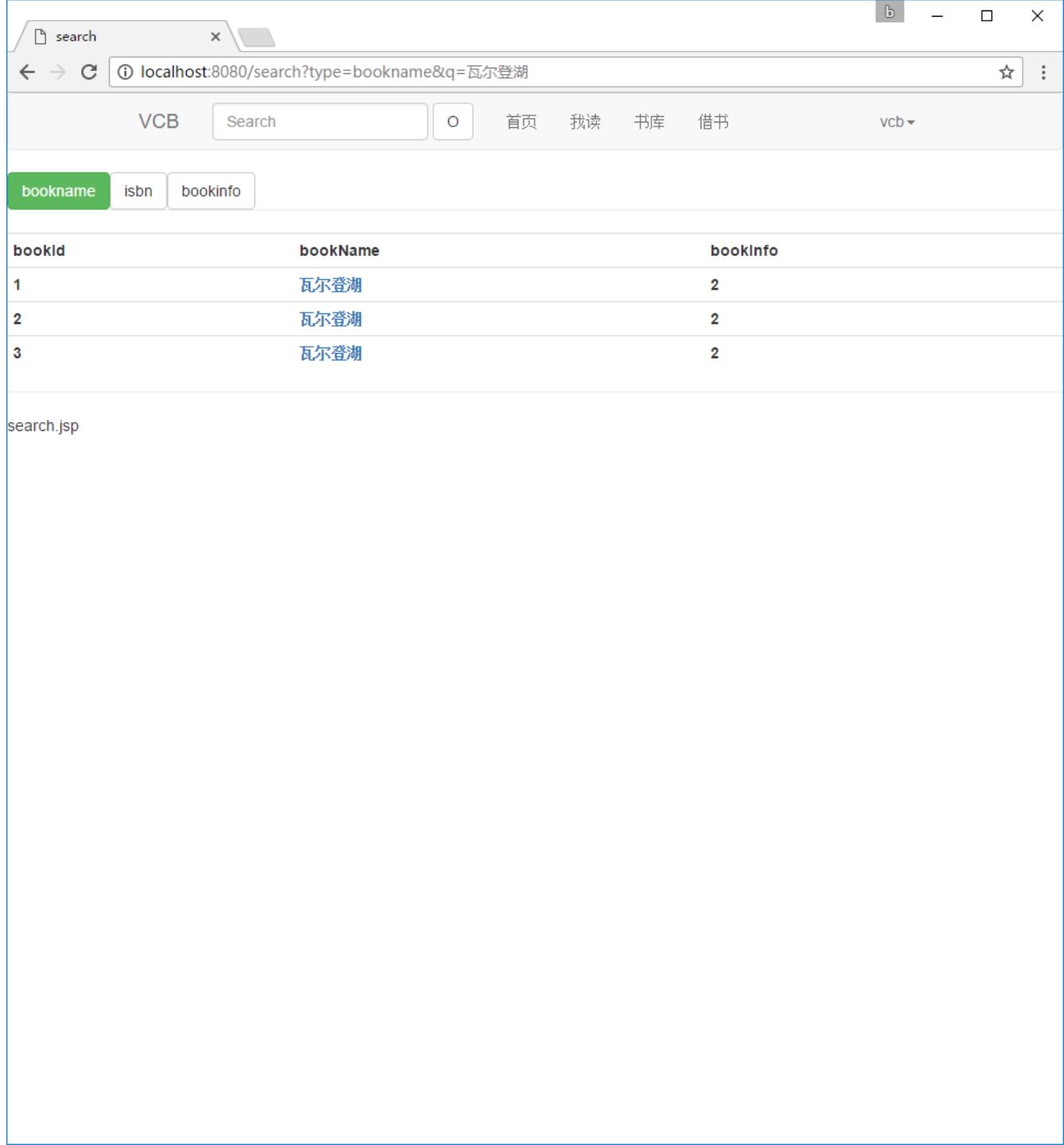
关于userId为1的用户有个很奇怪的bug，任何登录用户访问此用户这个页面，只要物品能借则状态永远是已加入借阅箱(实际你的借阅箱并没有此书)，其他用户页面不会出现此问题，问题可能出在了jstl标签库上，详见read.me

此处对按钮按了两次，是确实加入了借阅箱里



## search.jsp-bookname

使用关键词 瓦尔登湖，可以搜索到同名的书，但书的具体信息是不同的，它们的bookinfo的id均为2，因为他们的作者写出的内容是相同的，但译者不同或者译者也是一个人甚至内容一模一样但isbn一定是不同的。



# book.jsp

可以看到此用户的默认地点已经有了此书，  
按下通过isbn借书按钮链接到search.jsp-  
isbn



# search.jsp-isbn

可以看到拥有这本书并愿意将这本书借出的用户和地点等信息，并在地图上标注了地点

刚才放入借阅箱的书在此处也显示为已加入借阅箱

这里的按钮跟前面的一样也可以加入借阅箱或者从中删除

在这里我们不进行任何操作

search

localhost:8080/search?type=isbn&q=9787530209073


VCBSearchO

首页我读书库借书vcb

booknameisbnbookinfo

瓦尔登湖 [ISBN:9787530209073]

地图卫星图像



userNickName	locationName	bookName	状态
root	国际饭店	瓦尔登湖	已加入借阅箱
GHM1230	上海唐镇街道102室	瓦尔登湖	加入我的借阅箱
abc	和平饭店	瓦尔登湖	加入我的借阅箱

search.jsp



# search.jsp-bookinfo

回到刚才的book页，按下通过bookinfo借书按钮显示此页

此页显示出了所有bookinfo为2的书，并在地图上显示出了有这几本书(只要有这几本书其中一本并愿意借出)等信息

bookinfo解决了现实中一个很大的问题:如果一本书是外国人写的，这本书很有名，那很可能这本书有多个译本，如果找书只是单单用过书名搜索，那搜索效果可能并不好，而通过bookinfo可以很容易找到各个版本(数据库里有个languages表，所以可以不单单找简体中文的，也可以找到英文版的甚至藏文版的如果有的话，当然这个功能在这里并没有实现)，现实中类似这样的功能貌似并没有或者说实现的很不好(豆瓣)。一般一本书的印刷是有限的，过个几年，书就不再印刷销售就绝版了，所以一本书的数量并不会很多，而通过isbn搜索可能能借到这本书的地点离你很远，通过bookinfo可以找更多书可能借书地点也会近点如果对版本没要求的话。总之很好很强大

这里对所有书都加入借阅箱

search

localhost:8080/search?type=bookinfo&q=2

VCBSearchO

首页我读书库借书vcb


booknameisbnbookinfo

1 瓦尔登湖[ISBN:9787532739578]

2 瓦尔登湖[ISBN:9787532747979]

3 瓦尔登湖[ISBN:9787530209073]

地图卫星图像



上海市

userNickName	locationName	bookName	状态
root	国际饭店	1 瓦尔登湖	已加入借阅箱
root	大光明电影院	2 瓦尔登湖	已加入借阅箱
root	国际饭店	3 瓦尔登湖	已加入借阅箱
GHM1230	上海唐镇街道102室	3 瓦尔登湖	加入我的借阅箱
abc	和平饭店	2 瓦尔登湖	加入我的借阅箱
abc	和平饭店	3 瓦尔登湖	加入我的借阅箱

search.jsp



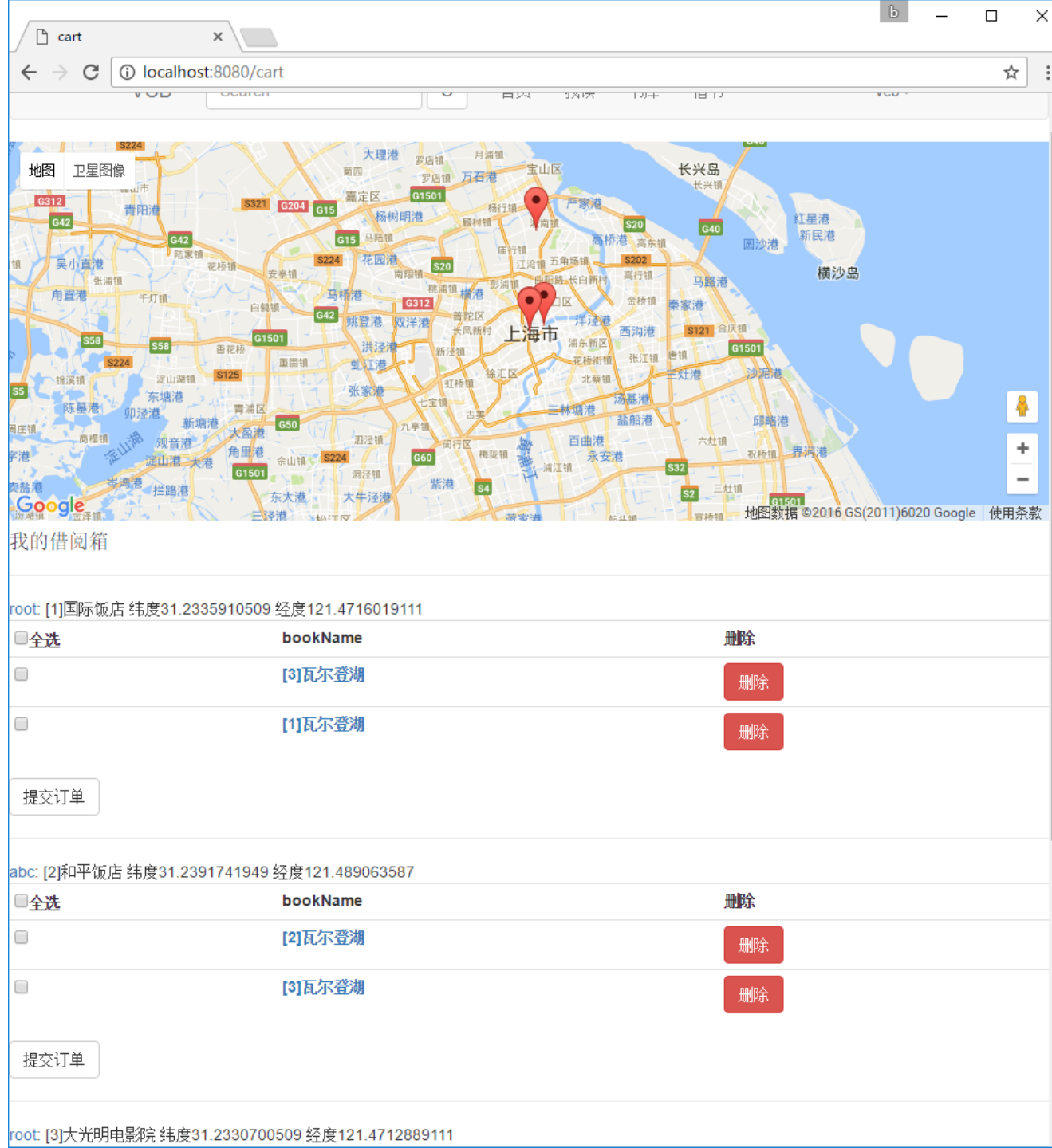
# cart.jsp

显示你的借阅箱

这里有四个地点，但因为国际饭店离大光明电影院很近，所以看上去只有三个地点，这里也可以看到同一个用户不同地点是不能在一起创建同一个订单的

删除按钮可以将书从借阅箱删除，同样使用了ajax

在这里提交了root的国际饭店订单，提交订单同样使用了ajax，成功创建订单会弹出提示框告知并跳转到借入订单管理页



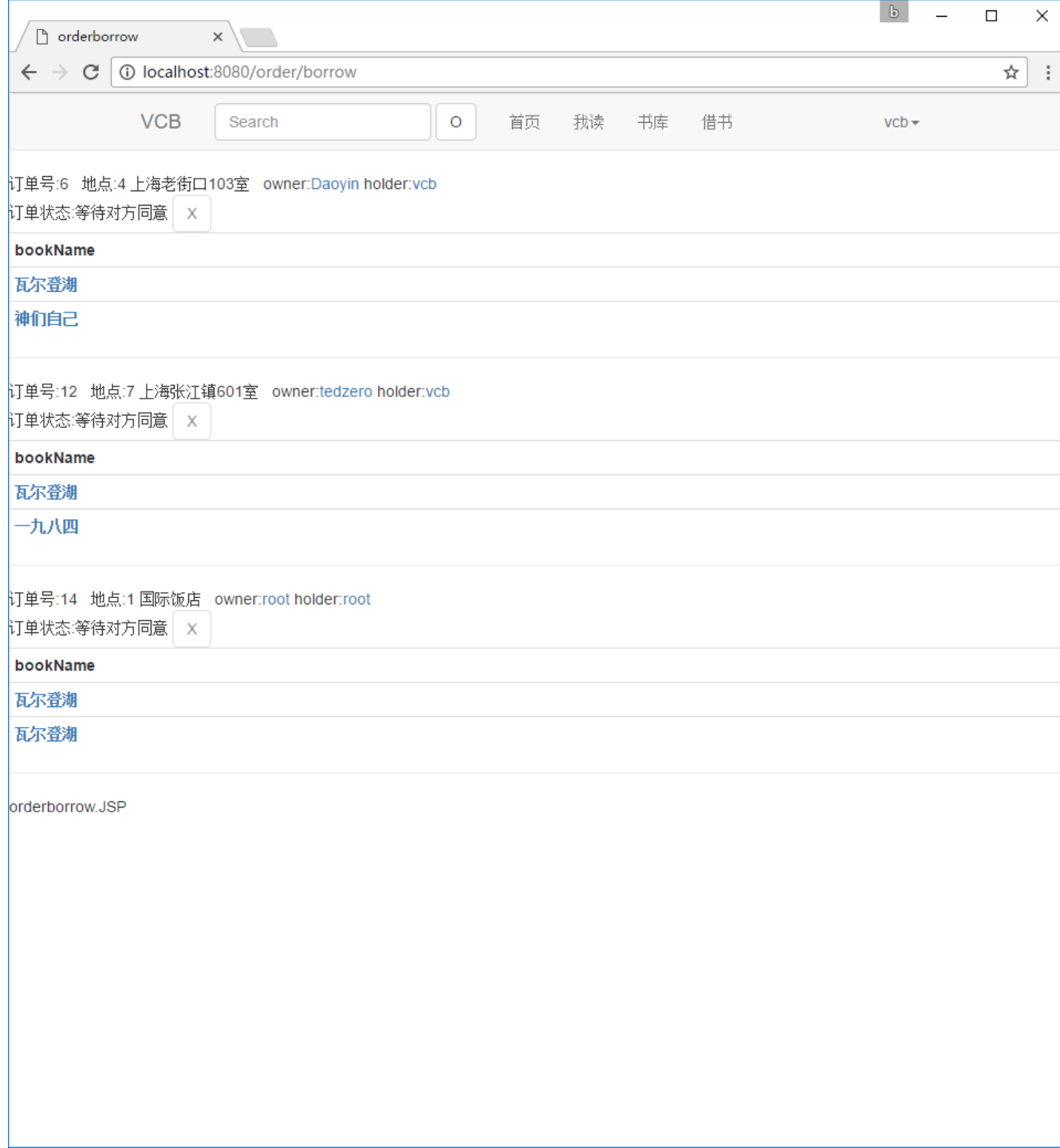
# orderborrow.jsp

显示用户借入的订单(包括历史订单)

vcb用户

可以看到刚才创建的订单号为14,订单状态为等待对方同意

在这里可以看到有关于瓦尔登湖的其他订单，而在刚才搜索页并没有出现这些书，这是因为订单创建了就会改书的状态，而不是要等对方同意了才改状态



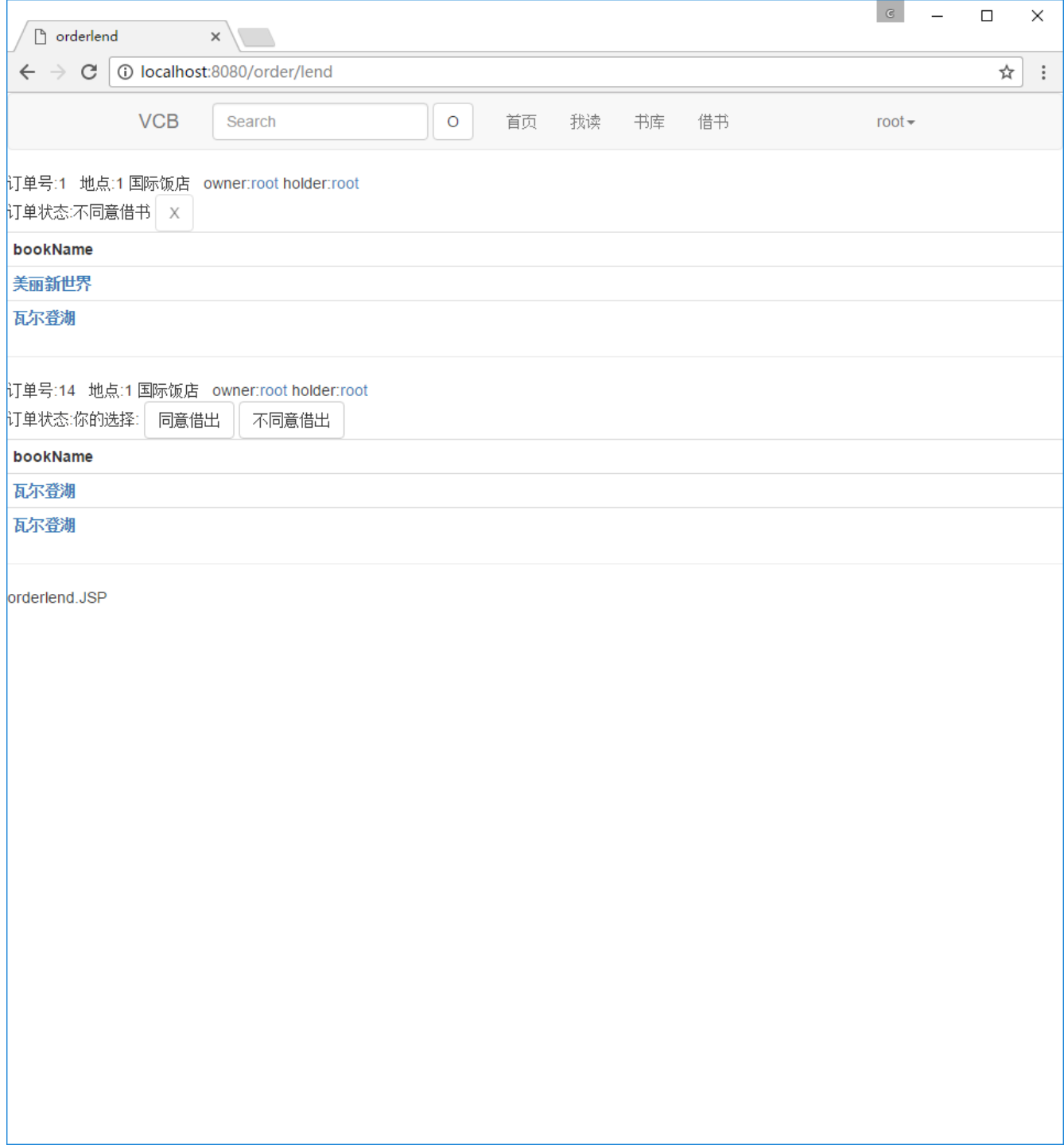
# orderlend.jsp

显示用户借出的订单(包括历史订单)

root用户

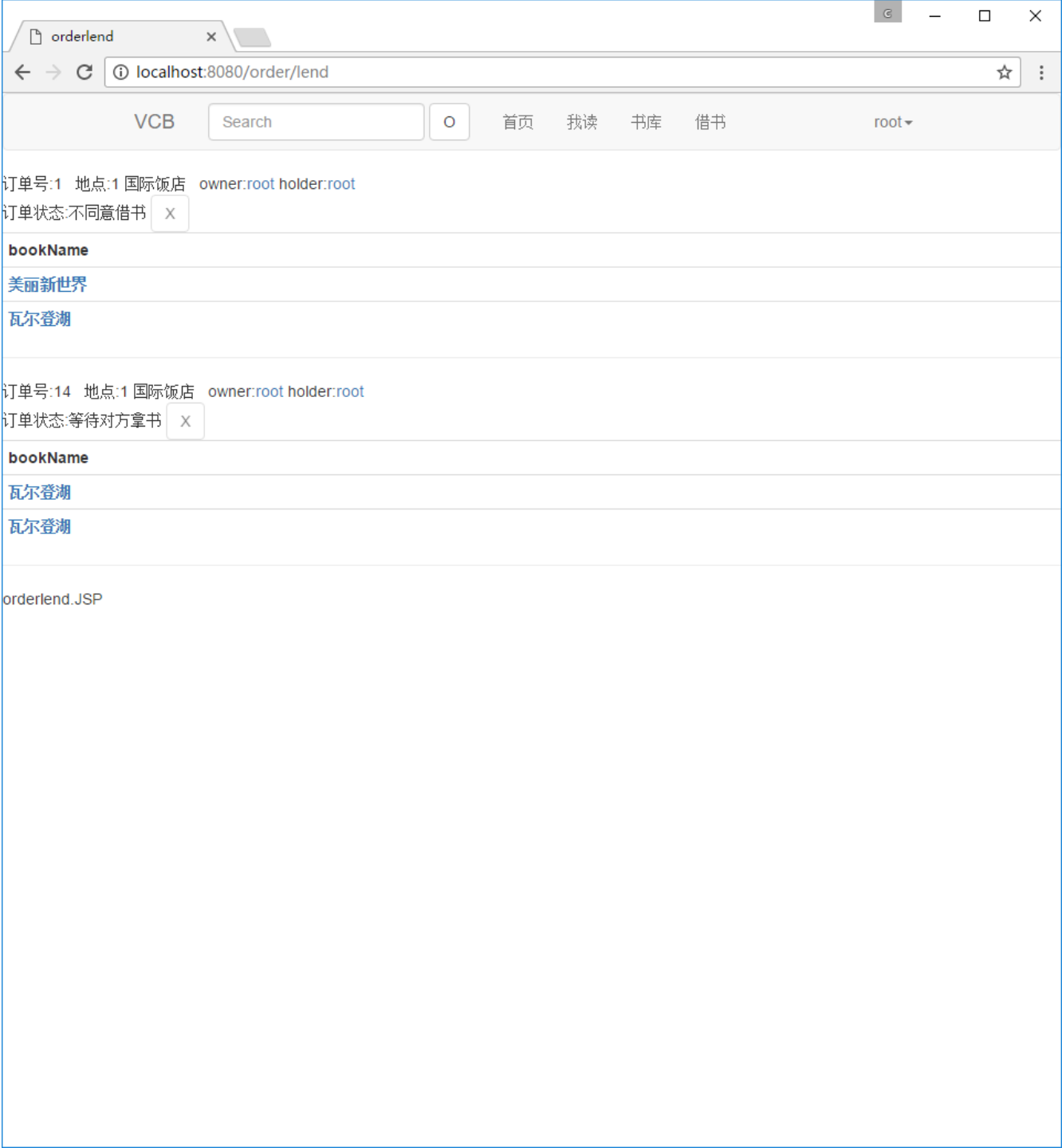
切换到root用户，在这里可以看到vcb用户提交的订单，有同意借出和不同意按钮，不同意则马上结束订单，并将书的状态从借出改为公开状态，同意则进入下一步。这两个按钮同样使用了ajax

这里点同意，这时root用户看到的订单状态会改为等待对方拿书



orderlend.jsp

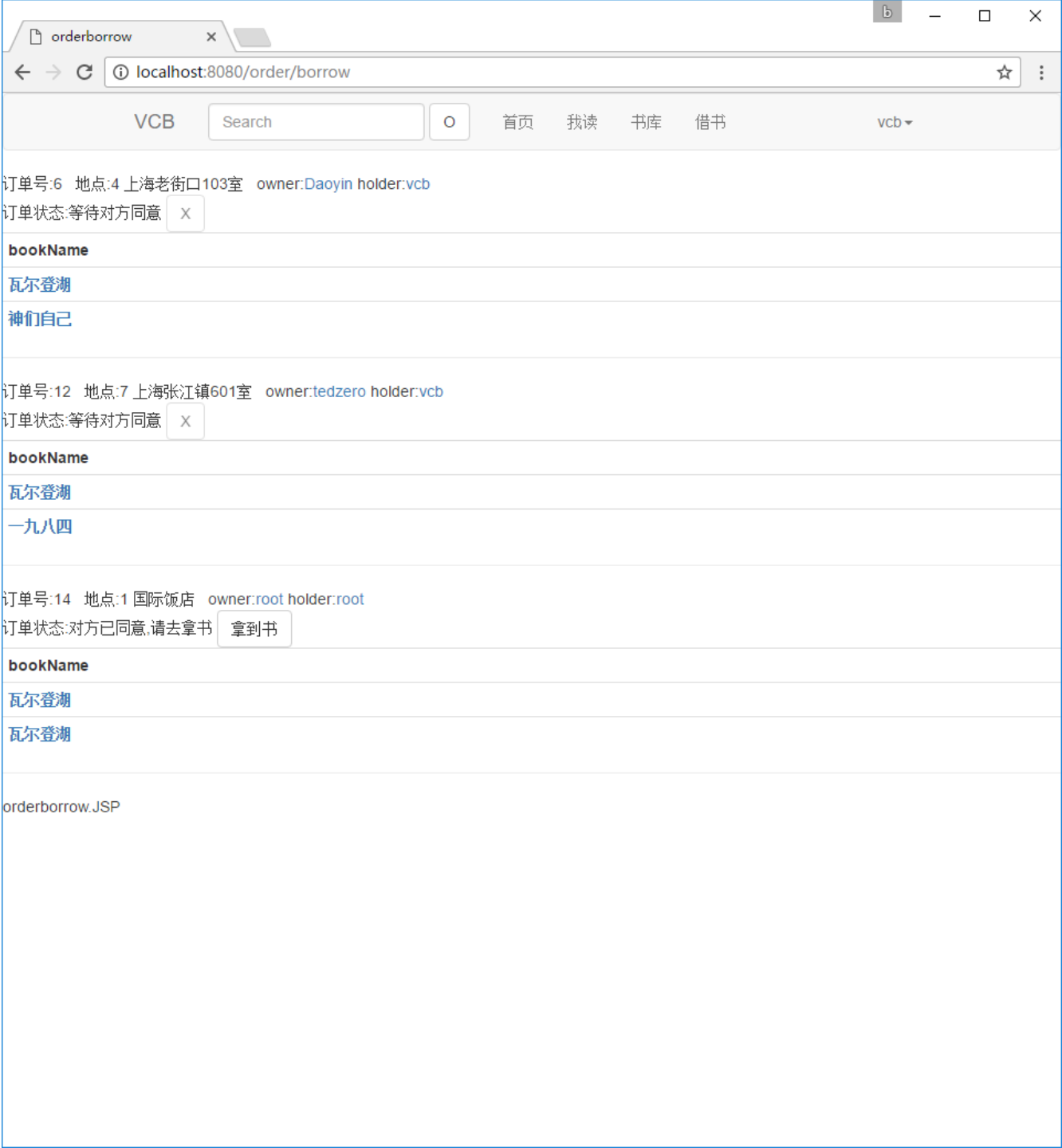
root用户



# orderborrow.jsp

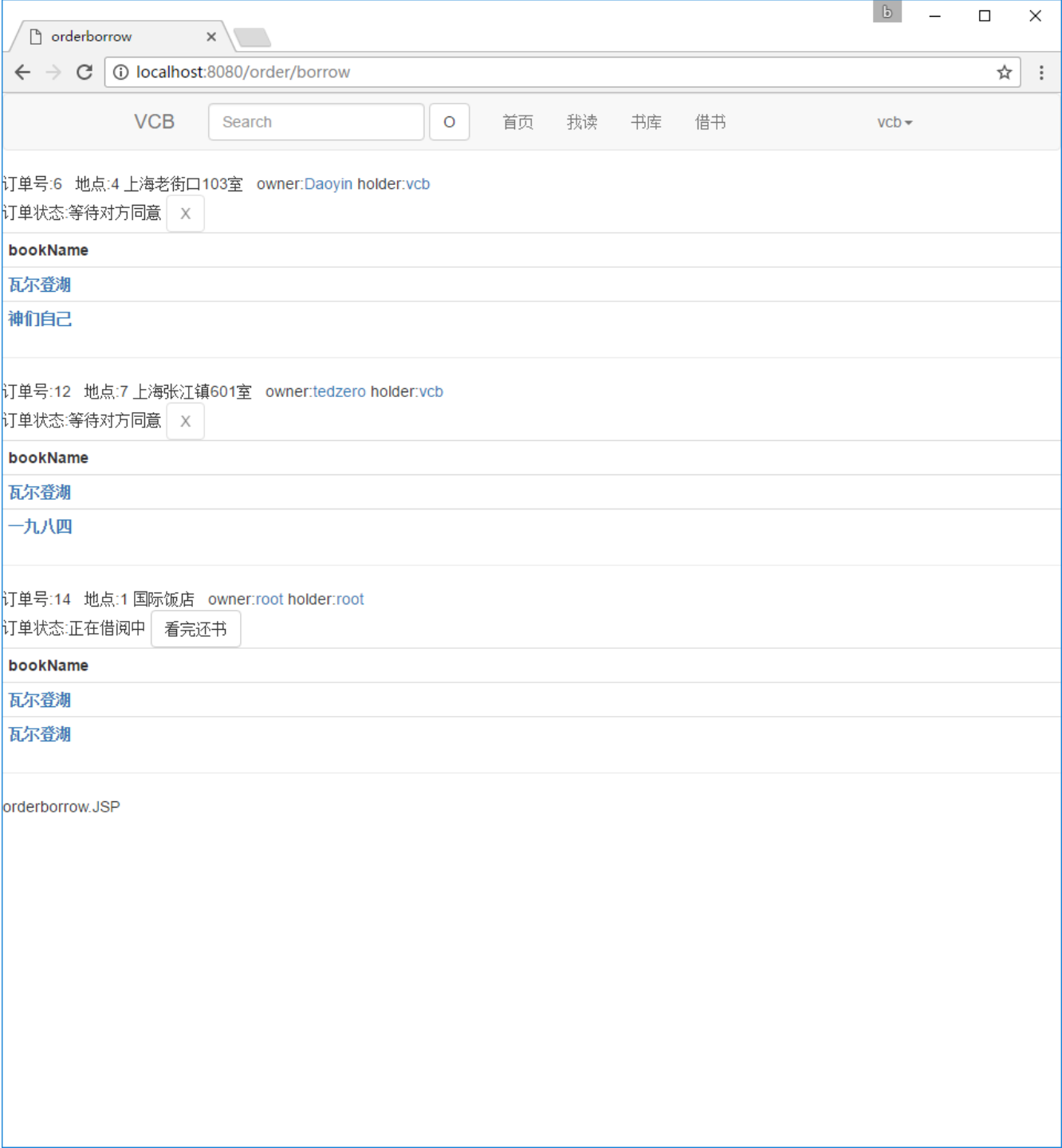
vcb用户

点拿到书，订单状态改为正在借阅中，并  
有个看完还书的按钮



# orderborrow.jsp

vcb用户

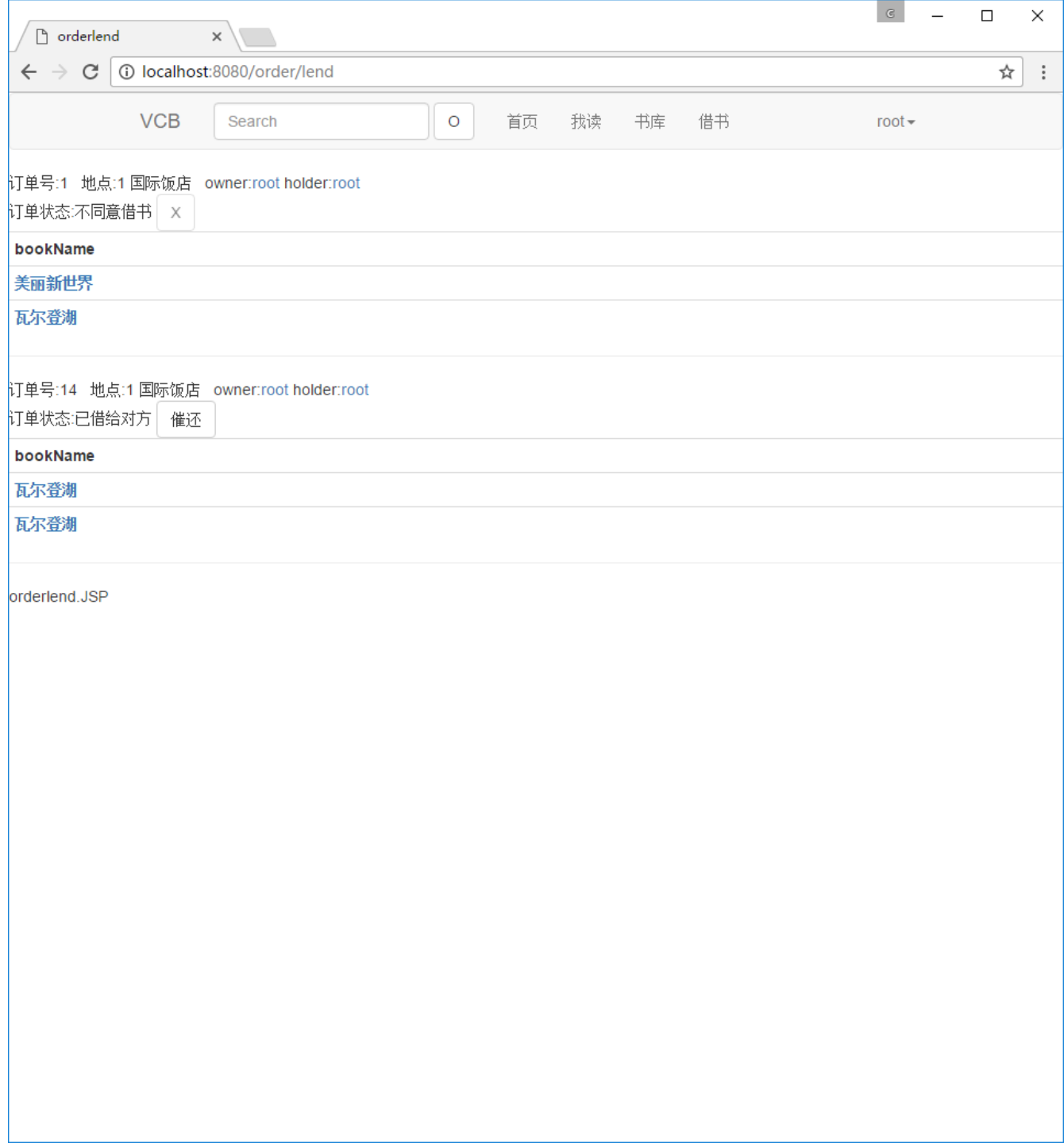


# orderlend.jsp

root用户

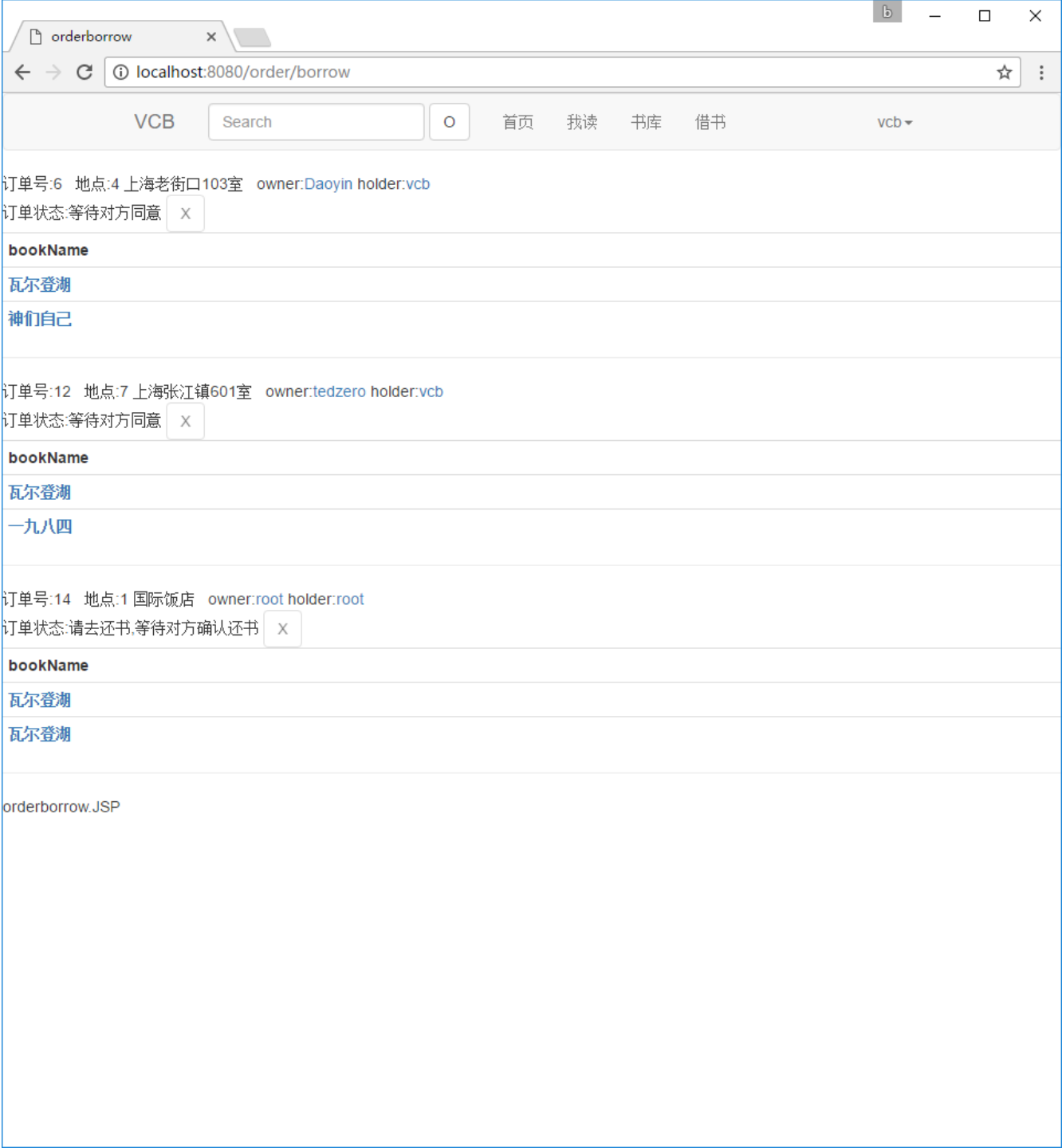
这里的催还按钮效果跟看完还书按钮一样，  
两个用户都能操作到下一步

点催还



# orderborrow.jsp

vcb用户

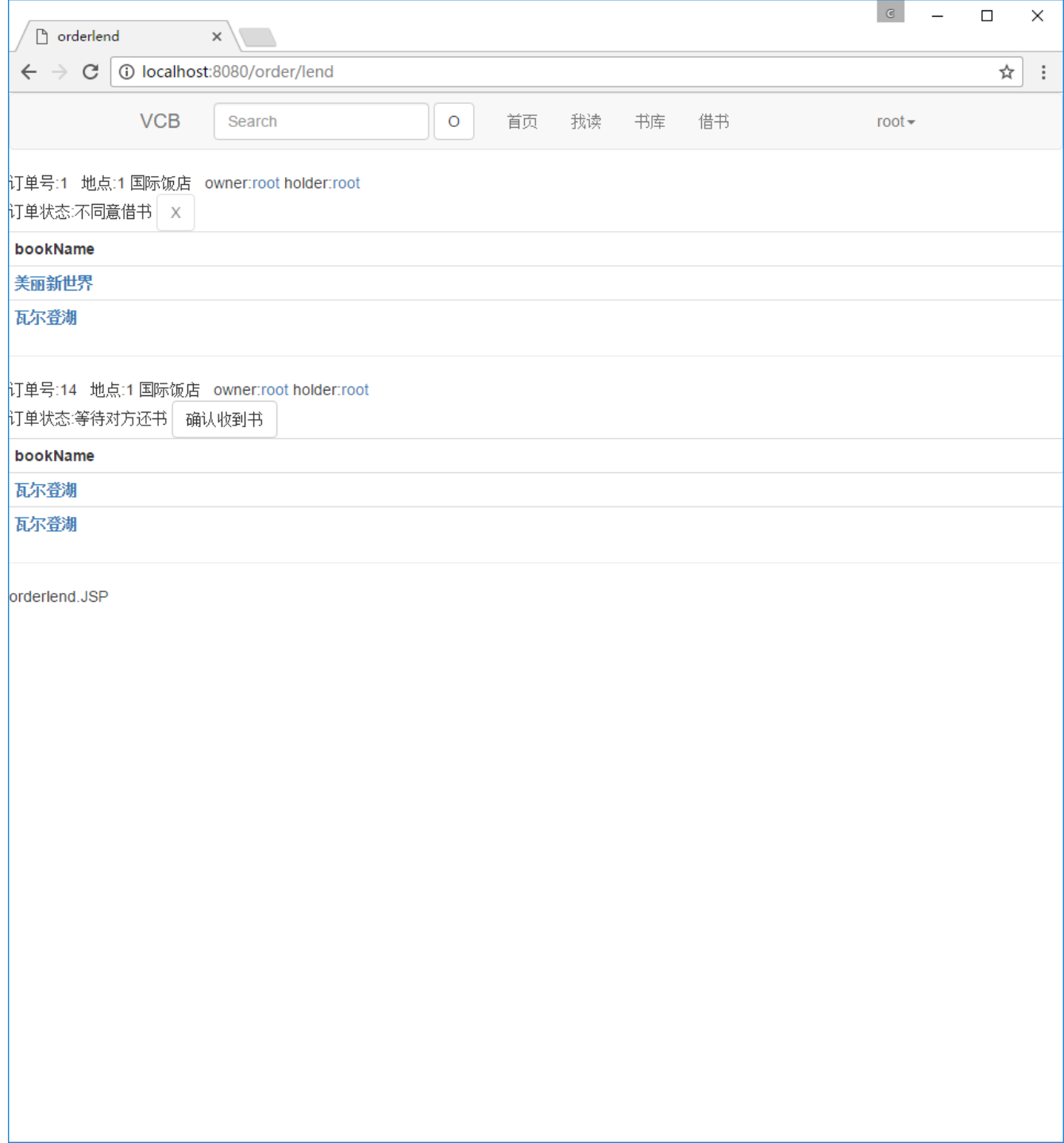




# orderlend.jsp

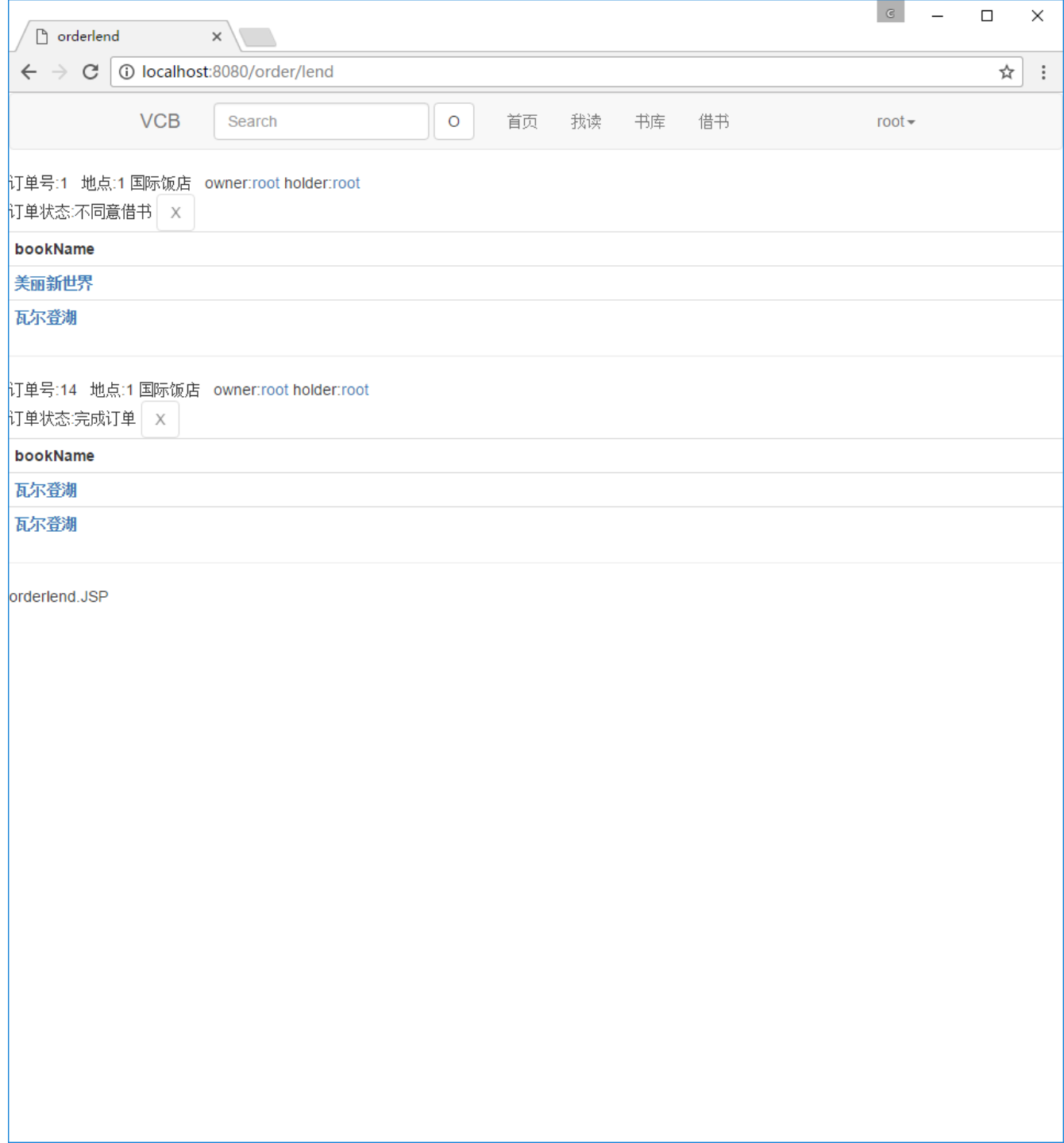
root用户

点确认收到书，完成订单



orderlend.jsp

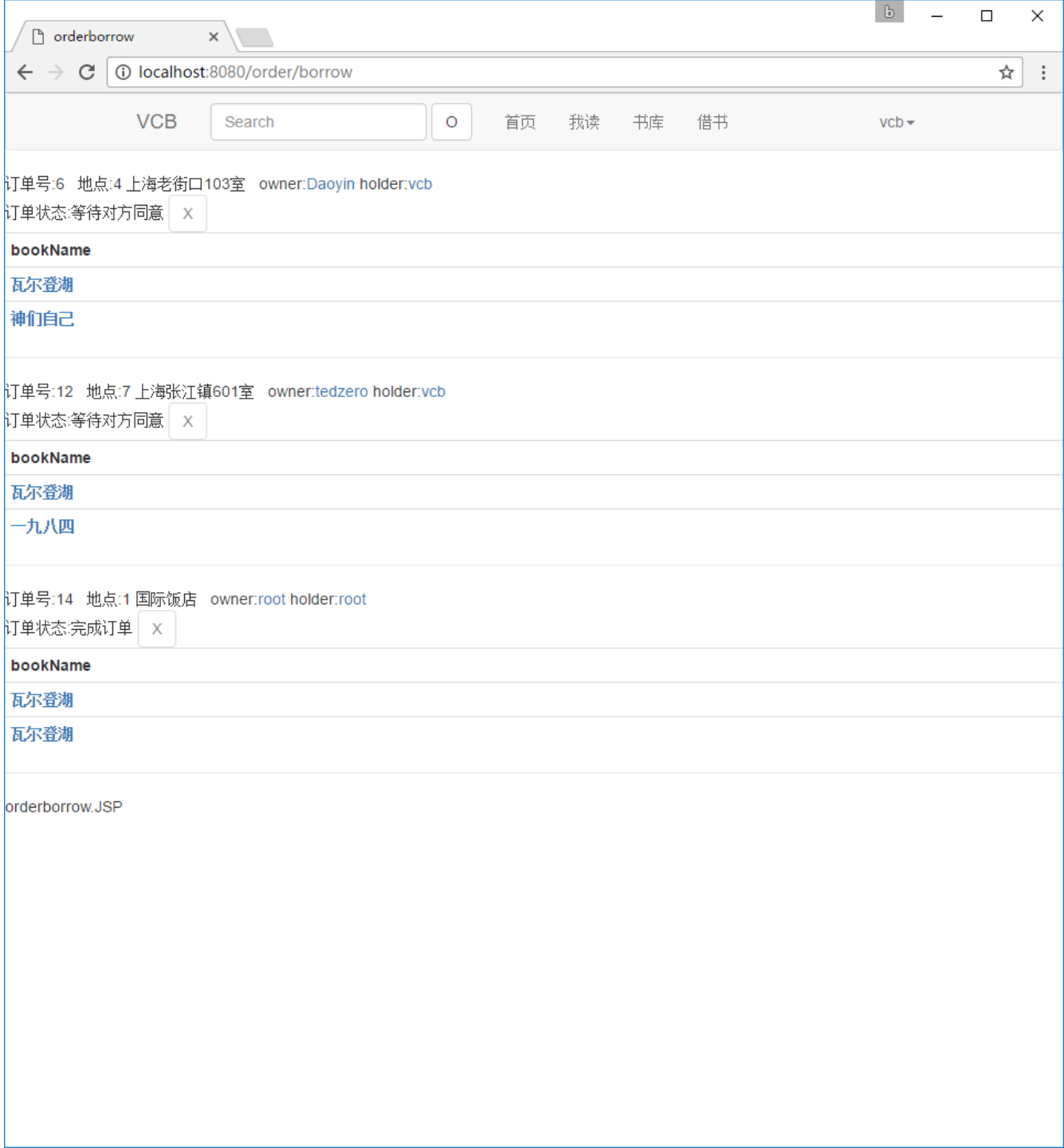
root用户



# orderborrow.jsp

vcb用户

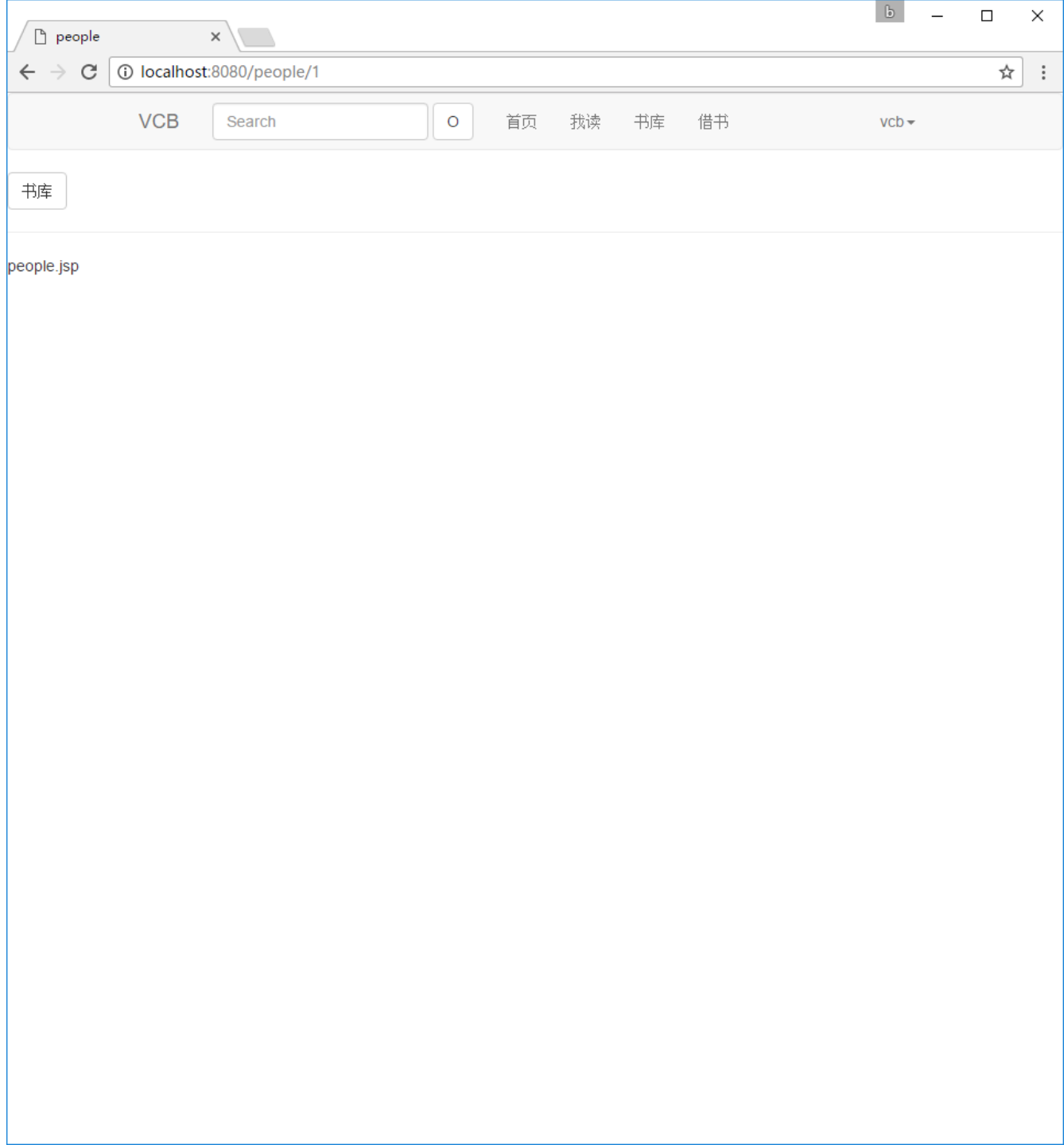
完成订单，书的持有人变回为root



# people.jsp

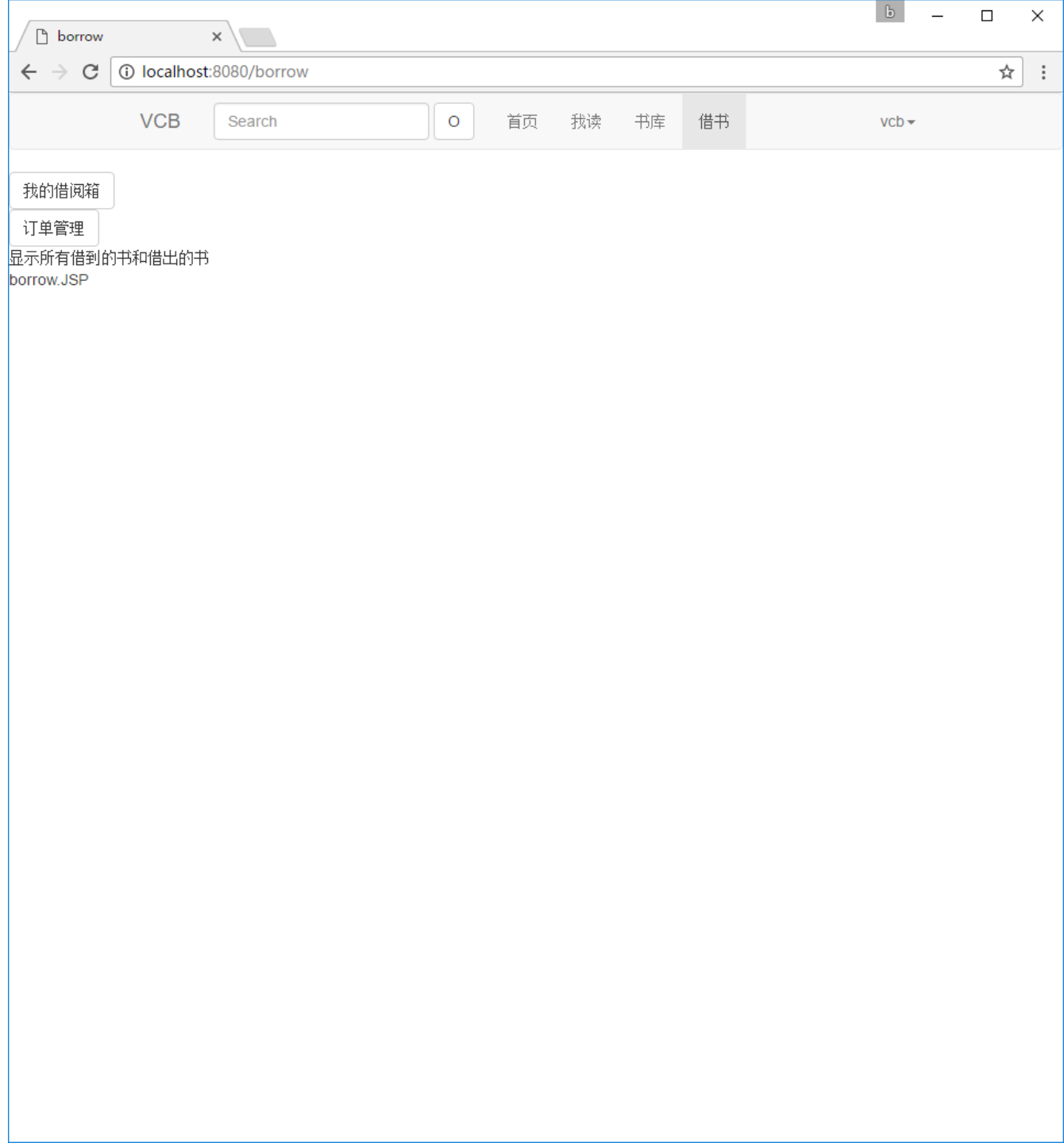
书库按钮链接到别人的书库

显示这个用户在读，读过，想读的书(未实现)



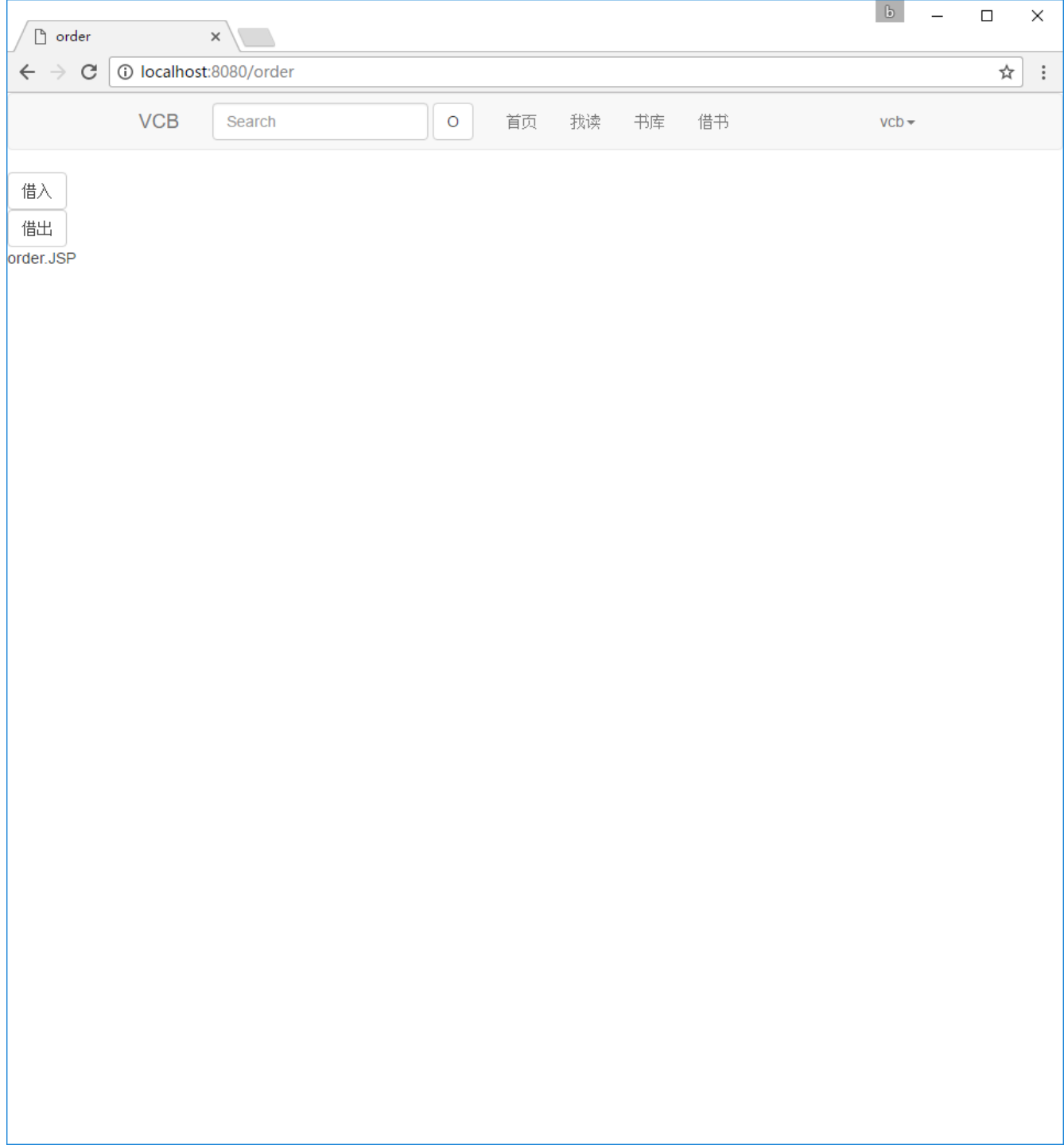
# borrow.jsp

显示所有借到的书和借出的书(未实现),  
两个按钮分别链接到cart.jsp, order.jsp



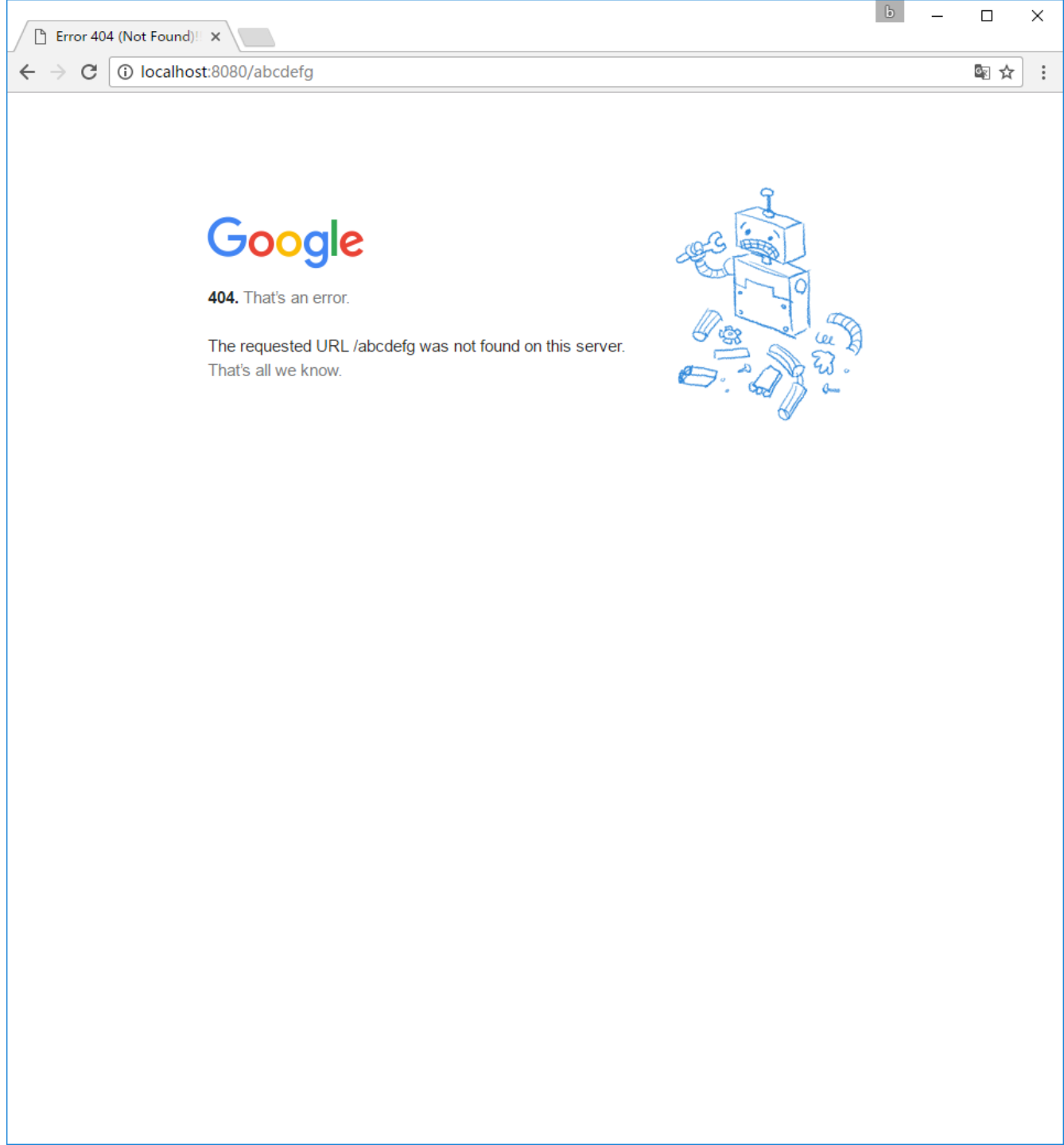
# order.jsp

两个按钮分别链接到orderborrow.jsp,  
orderlend.jsp



# 404.jsp

仿照Google的404页



设计&细节
















# 设计过程

- 大概功能在脑中确定
- 在纸上画出界面草图
- 确定大概要用到的技术
- 确定需要的表及字段
- 确定大体的类包结构和目录结构

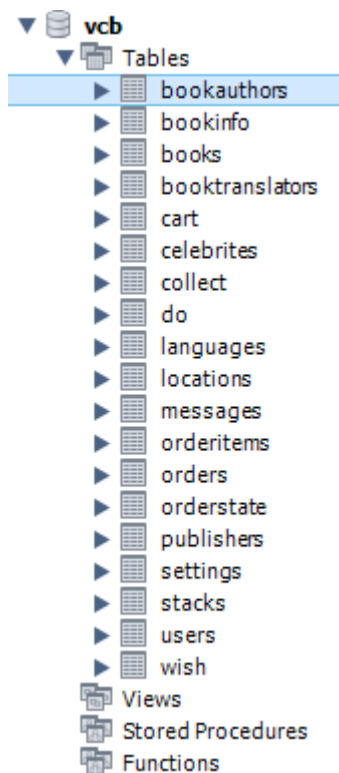
# 环境

- JDK 1.8
- Tomcat 8.0
- MySQL 5.7
- Struts2 2.3.31
- Bootstrap 3.3.7
- jQuery 3.1.1

- jar包

-  commons-fileupload-1.3.2.jar
-  commons-io-2.2.jar
-  commons-lang3-3.2.jar
-  freemarker-2.3.22.jar
-  javassist-3.11.0.GA.jar
-  mysql-connector-java-5.1.40-bin.jar
-  ognl-3.0.19.jar
-  struts2-core-2.3.31.jar
-  struts2-json-plugin-2.3.31.jar
-  taglibs-standard-impl-1.2.5.jar
-  taglibs-standard-jstlel-1.2.5.jar
-  taglibs-standard-spec-1.2.5.jar
-  xwork-core-2.3.31.jar

# 数据库



bookauthors:书的作者celebrityId

bookinfo:书的信息, book可以通过bookinfo查找其他标有相同bookinfo的book

books:书

booktranslators:书的译者celebrityId

cart:借阅箱(购物车)

celebrities:存放作者和译者的具体信息

collect:记录用户读过的书

do:记录用户在读的书

languages:语言

locations:用户地点

message:用户发送消息

orderitems:订单物品

orders:订单

orderstate:记录订单状态时间

publishers:书的出版社














settings:用户设置表

stacks:书库表, 记录用户所属书

users:用户信息

wish:记录用户想读的书

# 结构

- >  dao
- >  entity
- >  interceptor
- >  rule
- >  service
- >  servlet
- >  show
- >  struts
- >  test
- >  util
  -  db.properties
  -  read.me
  -  struts.xml

dao:dao层

entity:对应表字段

interceptor:拦截器

rule:规则

service: service层

servlet:弃用

show:jsp显示的数据-model层

struts:所有实现action类和配置文件

test:弃用

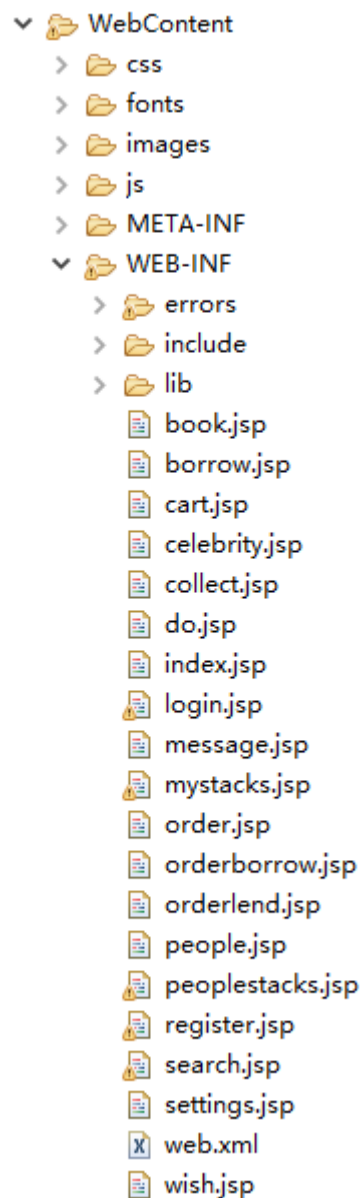
util:工具包

db.properties:数据库配置文件

read.me:未实现功能和bug记录

struts.xml:struts2配置文件

# 结构



WEB-INF

errors:错误页，只有404页

include:页头(导航条，除了json几乎所有Action包含)，页尾(只有index包含了)

book.jsp:book的显示页

borrow.jsp:借到的书(未实现)，链接到我的借阅箱和订单管理

cart.jsp:我的借阅箱

celebrity.jsp:名人的具体信息(未实现)

collect.jsp:已读的书(未实现)

do.jsp:在读的书(未实现)

index.jsp:首页

login.jsp:登录页

message.jsp:消息(聊天)页(未实现)

mystacks.jsp:我的书库

order.jsp:链接到借入，借出

orderborrow.jsp:借入订单操作

orderlend.jsp:借出订单操作

people.jsp:用户页(未实现)

peoplestacks.jsp:别人的书库(如果看自己的书库会转到我的书库页)

register.jsp:注册页

search.jsp:搜索页

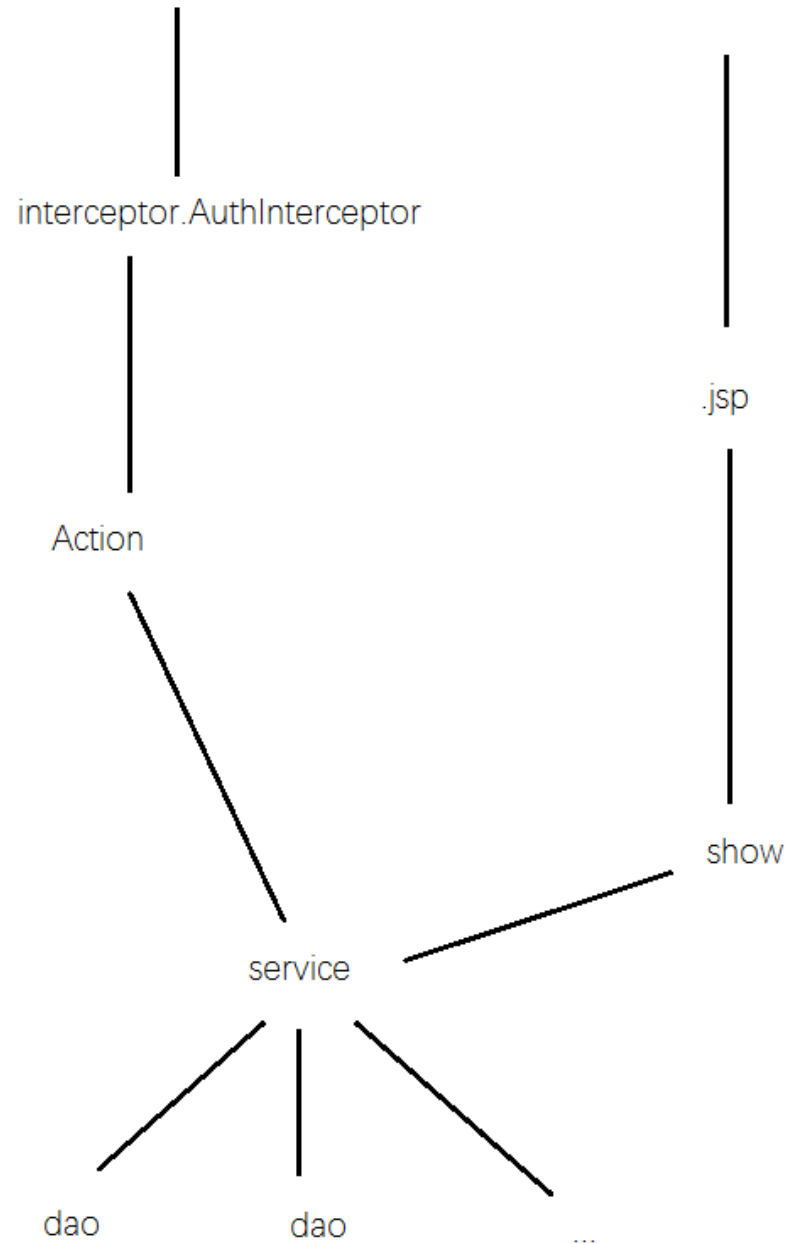
settings.jsp:用户设置页(未实现)

web.xml:配置文件

wish.jsp:想读的书(未实现)

# 流程

service将数据库中的数据整合存入show中，show相当于容器，jsp从中读数据并显示



# 设计缺陷

同一地点只能拥有相同isbn的书一本，对普通用户可能影响不大，但对类似公共图书馆的用户影响很大(相同isbn只能有一本)

# cookie

nickname	118&97&116&99&111&114&101&
user_id	11
verify	D57679842251C95B22501B887621BFD6

nickname:

用户的昵称，每个整数对应字符，&为分隔符

user\_id:

用户的id

verify:

userId+一串字符串通过MD5(32位大写)加密后的字符串



# interceptor.AuthInterceptor

interceptor包下只有此类。此类的作用为读取cookie，并将四个变量放入request域。几乎所有的Action都通过此拦截器。

isLogin:

将userId按规则加密，与读取的verify对比判断用户是否登录

```
<package name="myDefault" extends="struts-default" abstract="true">
  <interceptors>
    <interceptor name="auth" class="interceptor.AuthInterceptor"> </interceptor>
    <interceptor-stack name="myDefaultStack">
      <interceptor-ref name="auth"/>
      <interceptor-ref name="defaultStack"/>
    </interceptor-stack>
  </interceptors>
  <default-interceptor-ref name="myDefaultStack"> </default-interceptor-ref>
  <global-results>
    <result name="404">/WEB-INF/errors/404.jsp</result>
    <result name="verifyFail">/logout</result>
    <result name="needToLogin">/WEB-INF/login.jsp</result>
  </global-results>
</package>

<package name="myJsonDefault" extends="json-default" abstract="true">
  <interceptors>
    <interceptor name="auth" class="interceptor.AuthInterceptor"> </interceptor>
    <interceptor name="json" class="org.apache.struts2.json.JSONInterceptor"> </interceptor>
    <interceptor-stack name="myDefaultStack">
      <interceptor-ref name="auth"/>
      <interceptor-ref name="json"/>
      <interceptor-ref name="defaultStack"/>
    </interceptor-stack>
  </interceptors>
  <default-interceptor-ref name="myDefaultStack"> </default-interceptor-ref>
  <global-results>
    <result name="404">/WEB-INF/errors/404.jsp</result>
    <result name="verifyFail">/logout</result>
    <result name="needToLogin">/WEB-INF/login.jsp</result>
  </global-results>
</package>
```

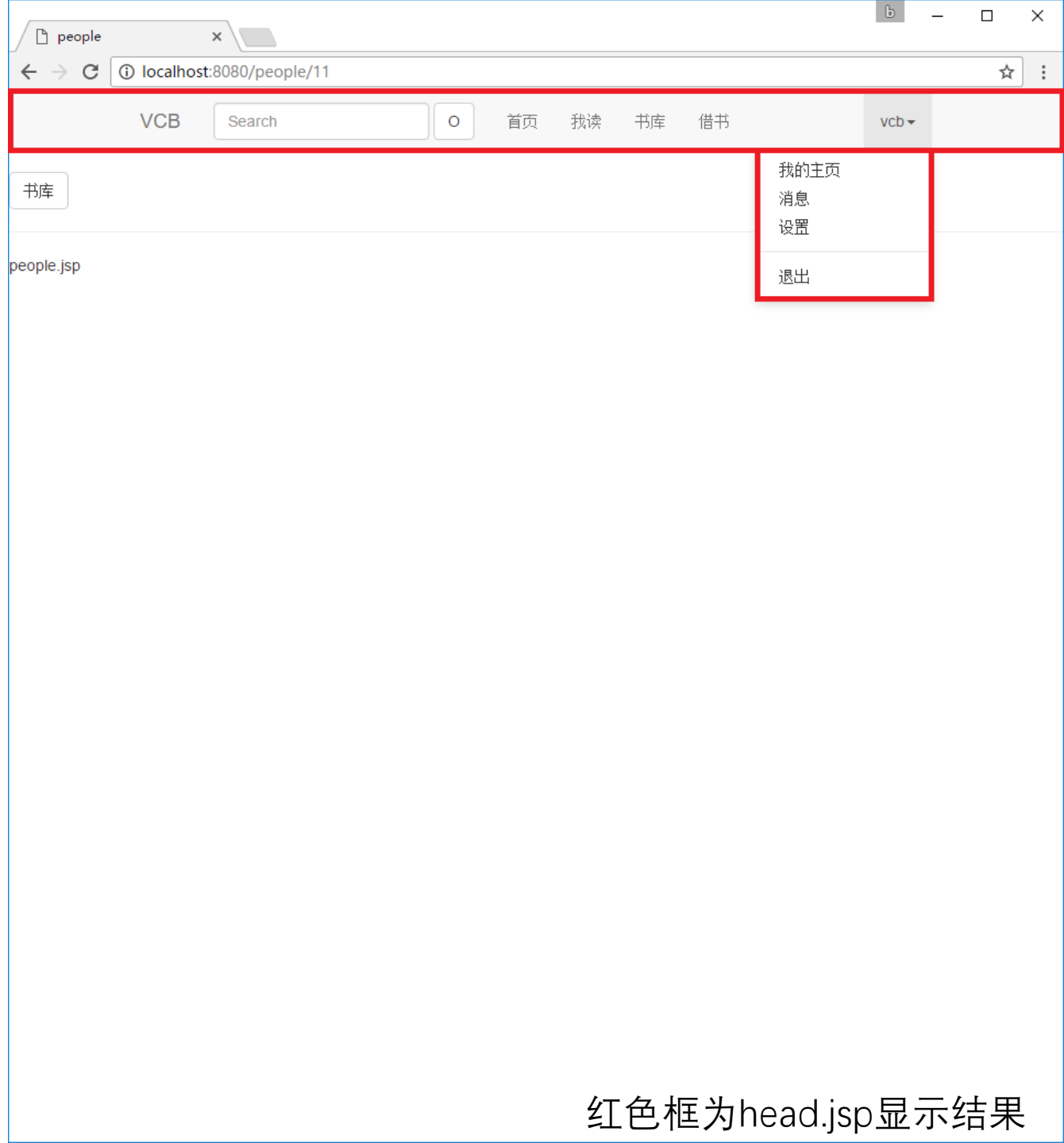
struts.xml

# head.jsp

导航条

几乎所有的jsp会有一行代码:

```
<jsp:include  
page="include/head.jsp"></jsp:include>
```



红色框为head.jsp显示结果

# 地图

```
<script type="text/javascript">
  var map;
  function initMap() {
    var shanghai = {lat: 31.2335, lng: 121.4716};
    var map = new google.maps.Map(document.getElementById('map'), {
      center: shanghai,
      zoom: 10
    });
    <c:forEach items="${searchbookinfo.show.locationIdMap}" var="l">
      var location = {lat: ${l.value.n}, lng: ${l.value.e}};
      var marker = new google.maps.Marker({
        position: location,
        map: map,
        title: '${l.value.name}'
      });
    </c:forEach>
  }
</script>
```

search.jsp-bookinfo

使用谷歌地图api

用jstl标签库生成javascript代码，这样就可以在地图上显示多个地点

# MyStacksShow, PeopleStacksShow

.jsp	MyStacksShow	PeopleStacksShow
mystacks.jsp	✓	
peoplestacks.jsp	✓	✓
search.jsp-isbn		✓
search.jsp-bookinfo		✓

MyStacksShow:存放用户书的信息

PeopleStacksShow:存放用户借阅箱里书的  
itemId, 一般作用为检查书是否已在用户  
借阅箱子内

本来设计为mystacks.jsp和peoplestacks.jsp  
一一对应各自的show, 但后来发现完全可以复用

# CartShow

存放借阅箱里书的信息

## **void initOwnerLocationIdMap()**

根据OwnerLocationId(书的所有者的地点id)将书(item)的信息放入map中(对借阅箱里的书根据地点分类)，这样就可以显示每个地点下的书，而不是杂乱无章

# OrderBorrowShow, OrderLendShow

orderborrow.jsp和orderlend.jsp均使用  
OrderBorrowShow, OrderLendShow弃用

OrderBorrowShow含有List<OrderVat>,  
存有指定用户的所有订单

OrderVat存放着某订单下的书(item)的信息

## orderstate

state	owner提示信息	操作按钮	holder提示信息	操作按钮
-99	不同意借书	X	对方不同意借书	X
0	未知错误	X	未知错误	X
1	你的选择	同意借出[2],不同意借出[-99]	等待对方同意	X
2	等待对方拿书	X	对方已同意,请去拿书	拿到书[3]
3	已借给对方	催还[4]	正在借阅中	看完还书[4]
4	等待对方还书	确认收到书[5]	请去还书,等待对方确认还书	X
5	完成订单	X	完成订单	X

注:

0.owner:书的所有者,holder:借书人

1.操作按钮为X说明用户不可操作

2."[]"内的数字为按下按钮后的state值