# 网上书店应用解决方案

**组员**
葛剑航 关宏朗 韩龙粤 马博轩 张楠 杨振杰

**分工一览**

| | |
|---|---|
| 关宏朗，杨振杰，韩龙粤 | 编写一个设计方案，包括相关的数据库、url-handler清单（含装饰函数），cookie中用户名和session的管理. |
| 葛剑航，马博轩，张楠 | 用 UML 顺序图 描述用户第一次进入网站、将书加入购物车、login的业务过程， 关注 cookie 和 购物车 的变化. |

---

## 1. 数据库部署

- 条目类型：用户
- 条目内容：`{"username": string, "password": string, "shopping cart": list ……}`

用户名和密码均以字符串储存（密码经`MD5`处理），购物车以列表储存商品ID

## 2. URL-handlers清单

### BaseHandler

```
class BaseHandler(tornado.web.RequestHandler):
    def get_current_user(self):
        # cookie: save the basic information of current user
        # if not log in, cookie doesn't save username.
        return self.get_secure_cookie("username")
```

### BookHandler

```
class BookHandler(BaseHandler):
```

```
    @roles(['guest', 'user'])
    def get(self):
        if not self.current_user
            self.render('book.html', user = "guest")
        else
            self.render('book.html', user = self.current_user)
            # update the shopping cart according to the session_id
            session_id = self.get_secure_cookie("session_id")
            currentShoppingCart = get_session(session_id)
            user = db.users.find_one({"username" : current_user})
            userShoppingCart = user['shoppingCart']
            newShopingCart = currentShoppingCart join userShoppingCart
            db.users.update({"username" : current_user},
                {"shoppingCart" : newShopingCart})
```

### LoginHandler

```
class LoginHandler(tornado.web.RequestHandler):
    # login
    def get(self):
        self.render("login.html")

    def post(self):
        username = self.get_argument('username')
        self.set_secure_cookie(username)
        if client has made a order form:
            self.redirect("/buy")
        else
            self.redirect("/")
```

### ShoppingCartHandler

```
class ShoppingCartHandler(BaseHandler):
    @roles(['guest', 'user'])
    def get(self):
        # if has log in
        if self.current_user
            user = db.users.find_one({"username" : current_user})
            currentShoppingCart = user['shoppingCart']
        else
            # session is used for guest to save the shoping cart
            # session is saved in the data server
            # it's id is saved in cookie
```

```
            session_id = self.get_secure_cookie("session_id")
            if not session_id:
                # if no session for current guest, create one
                session_id = create_session(newShopingCart)
                self.set_secure_cookie("session_id", session_id)
            else:
                currentShoppingCart = get_session(session_id)

        self.render('shoppingCart.html', user = self.current_user,
            shoppingCart = currentShoppingCart)

    @roles(['guest', 'user'])
    def post(self, newShopingCart):
        if self.current_user
             # update the shoppingcart in the user
            db.users.update({"username" : current_user},
                {"shoppingCart" : newShopingCart})
        else
            # update the shoppingcart in the session
            if self.get_secure_cookie("session_id")
                db.session.update({"session_id" : session_id},
                    {"shoppingCart" : newShopingCart})
```

### BuyHandler

```
class BuyHandler(BaseHandler):
    @roles(['user'])
    def post(self, shopingCart):
        # buy things
        # update the shoppingcart
        if not self.current_user:
            self.redirect("/login")
        else
            pay for books
            clean the order form and update the shopping cart
        self.redirect("/")
```

### application

```
application = tornado.web.Application([
        (r'/', BookHandler),
        (r'/', LoginHandler),
        (r'/shoppingCart', ShoppingCartHandler),
        (r'/buy', BuyHandler),
```

```
        (r'/logout', LogoutHandler)
    ])
```

## 3. 用户名和`session`的管理

- 当用户成功登录，则`cookie`用于保存当前用户主要信息；
- 当匿名用户将货物加入购物车时，`session`用于储存购物车内商品ID，在匿名用户提交订单时登录账户，再把`session`内购物车信息更新到已登录用户的购物车信息当中，实现临时数据的数据迁移.

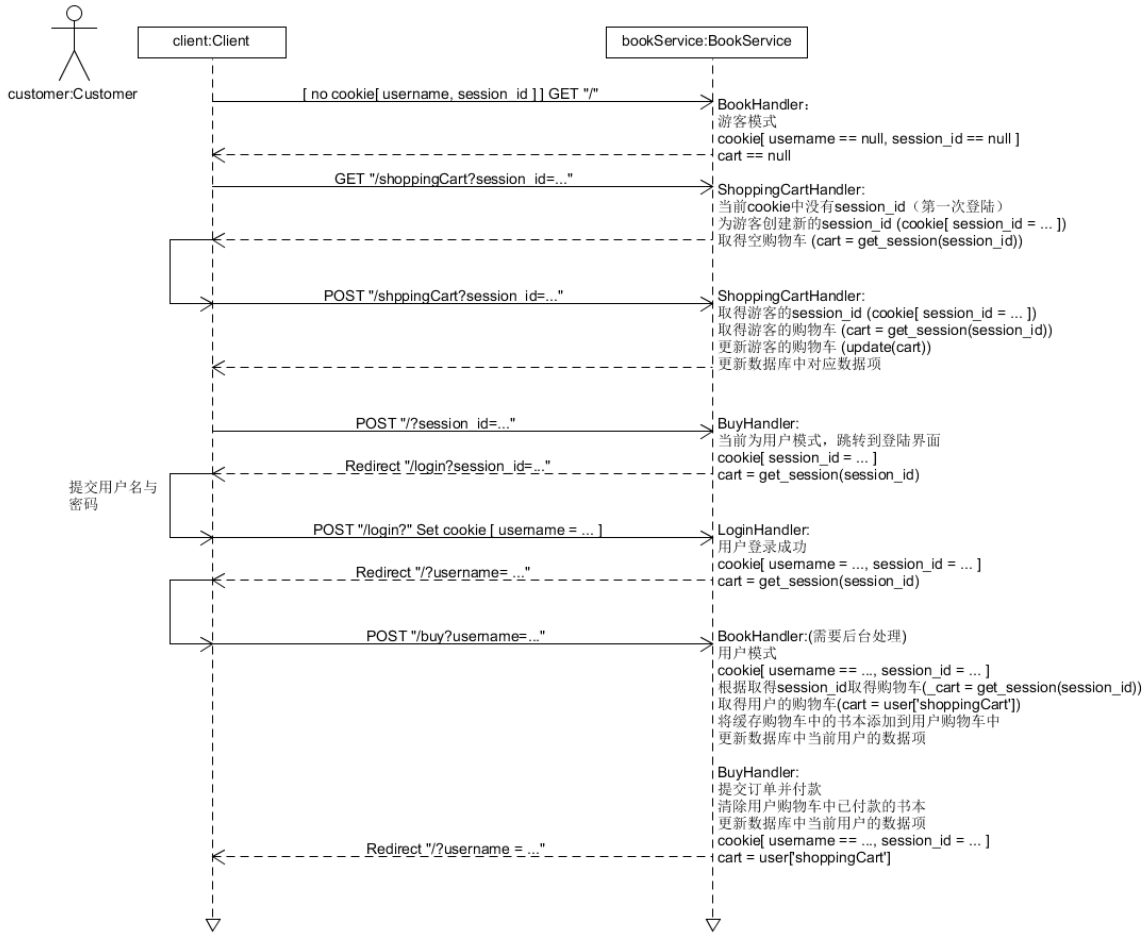
## 4. 用户事件

**场景一**
用户先进行身份验证并成功登陆，挑选货物到购物车中，提交订单成功购买

**场景二**
用户没有进行登录，处于匿名状态，能随意挑选货物到购物车中，当匿名用户提交了订单，服务器弹出用户登录界面，用户成功通过验证，但随即取消了订单，先前匿名状态下的购物车内容将增添到用户账号里的购物车中

**场景三**
用户没有进行登录，处于匿名状态，能随意挑选货物到购物车中，当匿名用户提交了订单，服务器弹出用户登录界面，用户成功通过验证，随即进行付款，愉快地进行了购物过程.

## 5. UML顺序图


customer:Customer
client:Client
bookService:BookService
[ no cookie[ username, session_id ] ] GET "/"
BookHandler：
游客模式
cookie[ username == null, session_id == null ]
cart == null
GET "/shoppingCart?session_id=..."
ShoppingCartHandler:
当前cookie中没有session_id（第一次登陆）
为游客创建新的session_id (cookie[ session_id = ... ])
取得空购物车 (cart = get_session(session_id))
POST "/shppingCart?session_id=..."
ShoppingCartHandler:
取得游客的session_id (cookie[ session_id = ... ])
取得游客的购物车 (cart = get_session(session_id))
更新游客的购物车 (update(cart))
更新数据库中对应数据项
POST "/?session_id=..."
BuyHandler:
当前为用户模式，跳转到登陆界面
cookie[ session_id = ... ]
cart = get_session(session_id)
Redirect "/login?session_id=..."
提交用户名与密码
POST "/login?" Set cookie [ username = ... ]
LoginHandler:
用户登录成功
cookie[ username = ..., session_id = ... ]
cart = get_session(session_id)
Redirect "/?username= ..."
POST "/buy?username=..."
BookHandler:(需要后台处理)
用户模式
cookie[ username == ..., session_id = ... ]
根据取得session_id取得购物车(_cart = get_session(session_id))
取得用户的购物车(cart = user['shoppingCart'])
将缓存购物车中的书本添加到用户购物车中
更新数据库中当前用户的数据项
BuyHandler:
提交订单并付款
清除用户购物车中已付款的书本
更新数据库中当前用户的数据项
cookie[ username == ..., session_id = ... ]
cart = user['shoppingCart']
Redirect "/?username = ..."