lab3　MVC 设计模式与视图技术 EL、JSTL

学号：_____　　姓名：_____

专业：_____　　年级：_____

实验目的

（1）理解 MVC 设计模式的数据流程

（2）理解请求转发页面数据的共享

（3）掌握视图层技术 EL、JSTL 的基本使用

（4）掌握基于 MVC 模式的系统开发

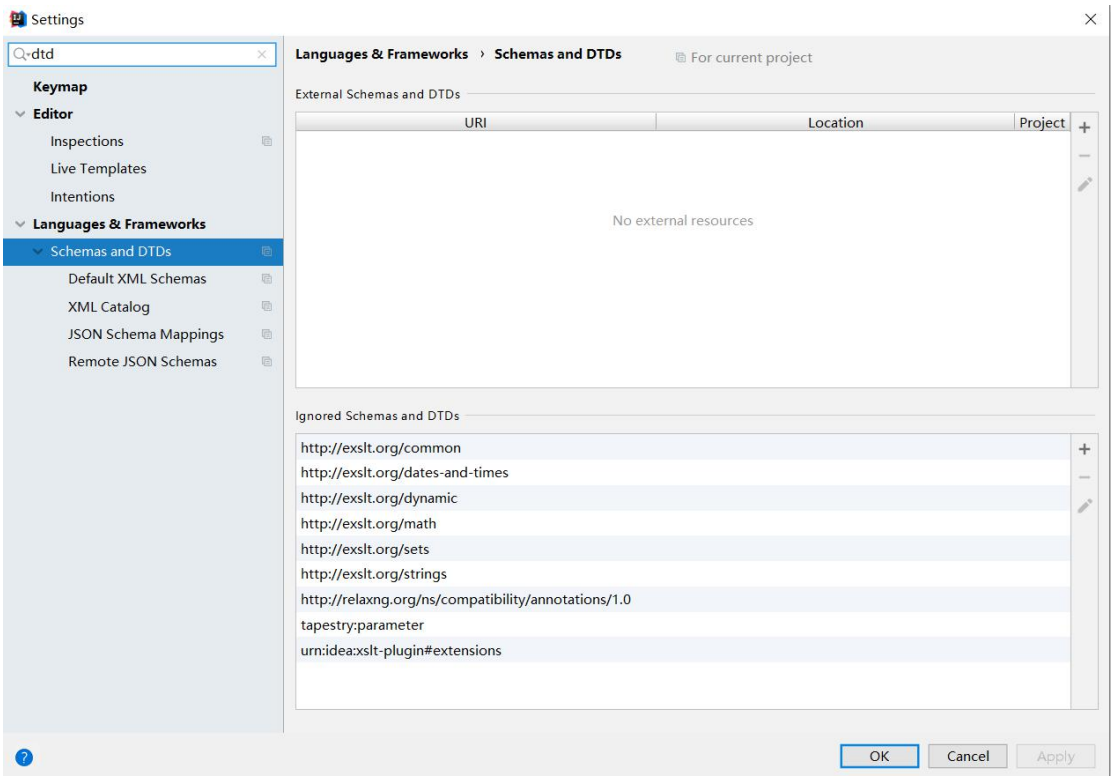
实验内容

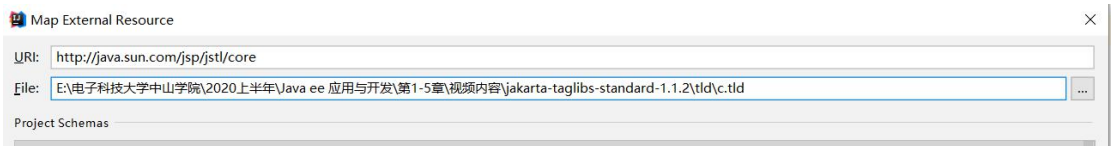
Part 1　票据管理 TicketAppDemo 的功能实现(完成时间为____分钟)

背景分析：票据系统是模拟了一个跨国公司（中山领航科技制造公司）的客户服务系统，公司客户遍布全球，目前需要开发一个票据系统，使客户可以提交票据的相关信息，并能够查看票据列表、票据详情等功能。

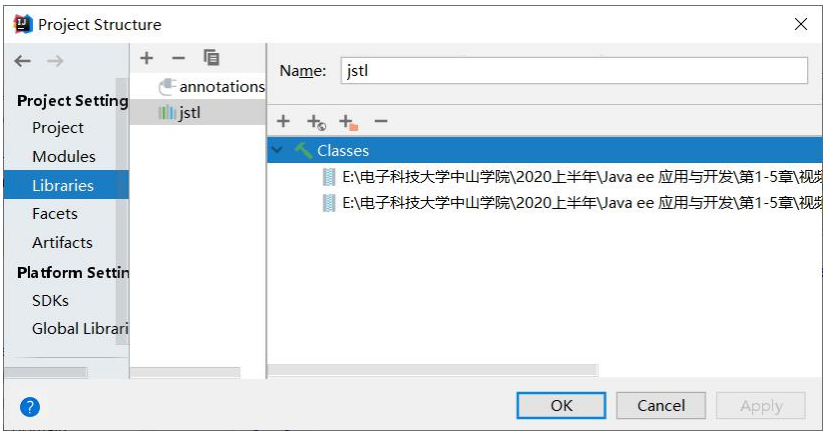
步骤 1：导入工程 TicketAppDemo，修改工程的 lib 配置，根据视频完成 JSTL 配置、tomcat 服务器配置。

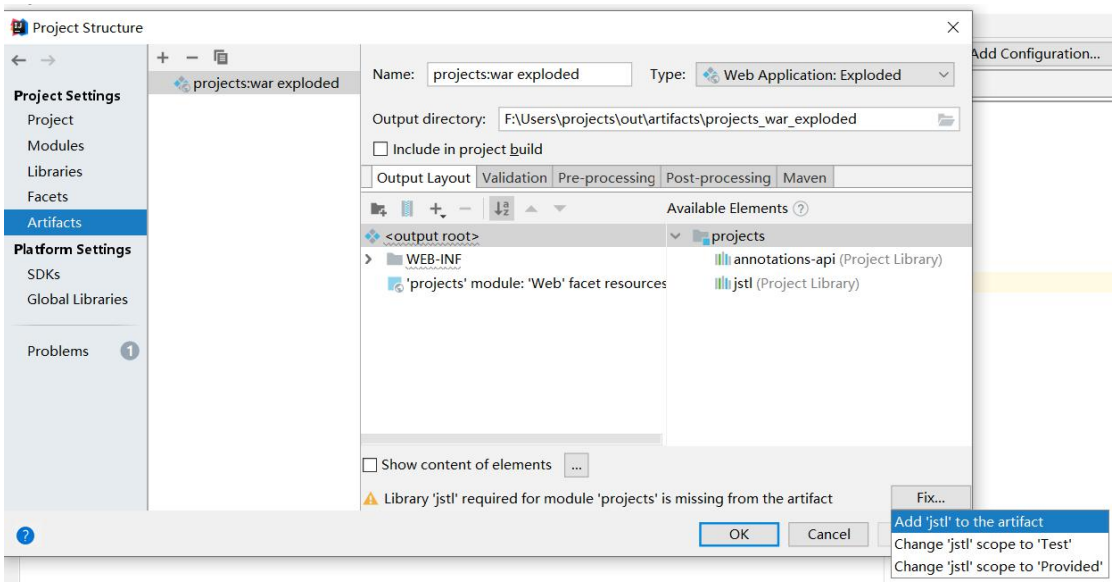
1）在 IDEA 中配置 JSTL 的 tld 模板，http://java.sun.com/jsp/jstl/core。



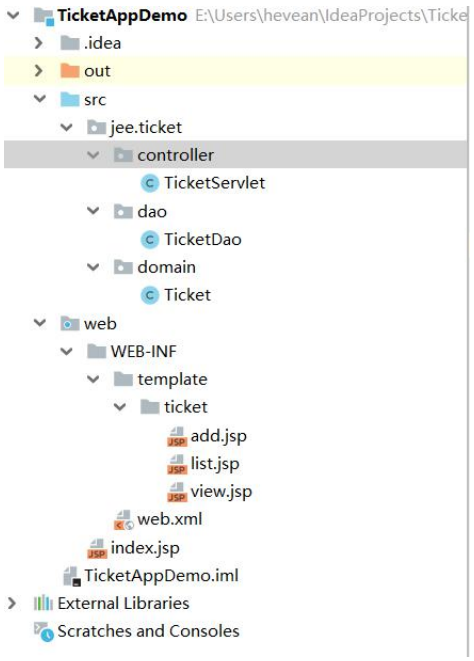


步骤 2：将 jstl 相应的 jar 包添加到工程 lib 目录下，修复 artifact 的错误。

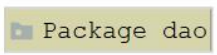




步骤 2：查看项目结构，如下图所示。

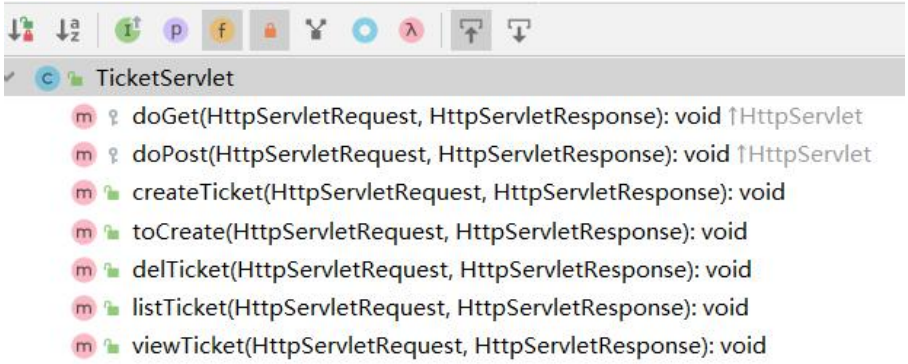


步骤 3：项目包的结构如下：



步骤 4：理解主要类的相关接口及其功能。（完成时间为____分钟）

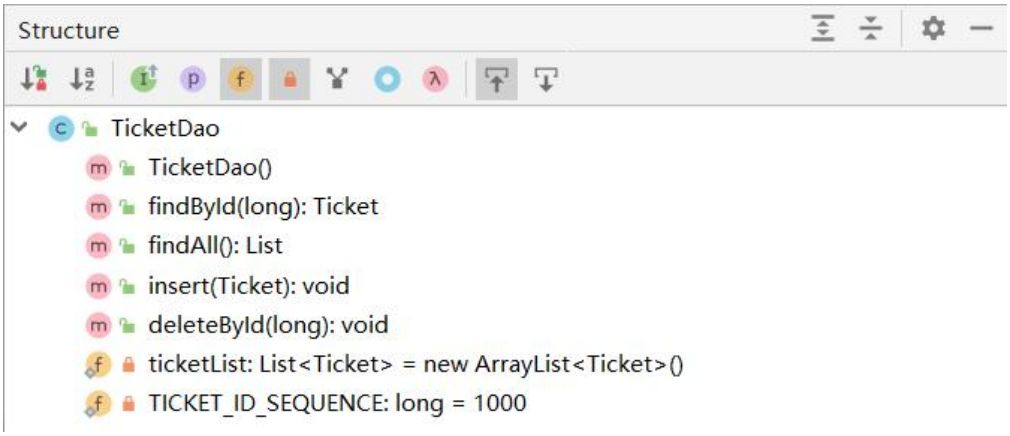
TicketServlet 主要方法列表如下图所示（使用 Alt+7 可以查看类的结构图）：



从方法名字上写出相关方法的主要功能(补充完成)：

方法名	功能说明
doGet()	

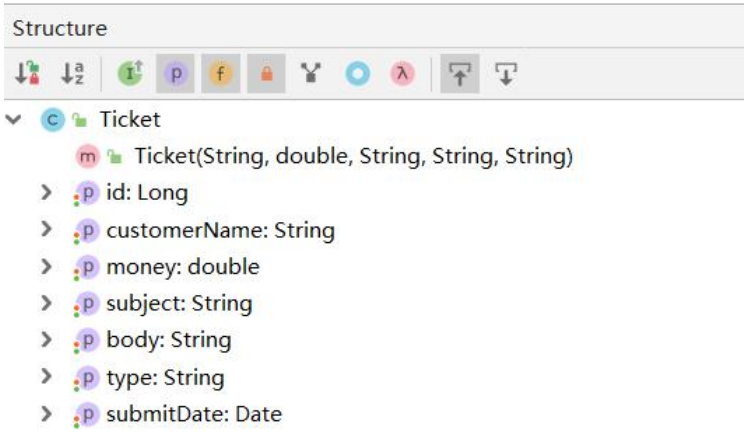
TicketDao 主要方法列表如下图所示（使用 Alt+7 可以查看类的结构图）：



从方法名字上写出相关方法的主要功能(补充完成)：

方法名	功能说明

Ticket 类为实体类，主要属性如下图所示（使用 Alt+7 可以查看类的结构图）：

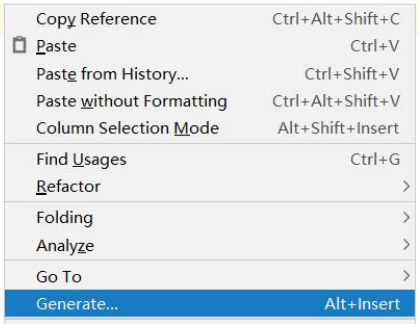


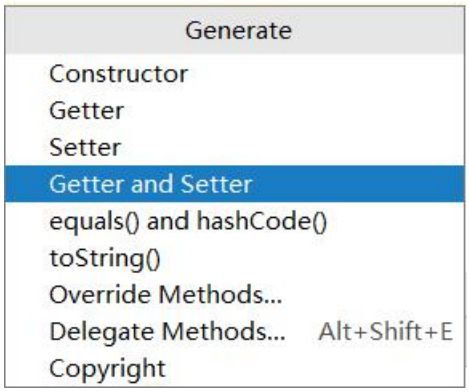
步骤 5：分别补充完成 Ticket、TicketDao 和 TicketServlet 的相关代码。

（1）在 Ticket 类中添加 setter 和 getter 方法。（完成时间为____分钟）

单击鼠标右键。

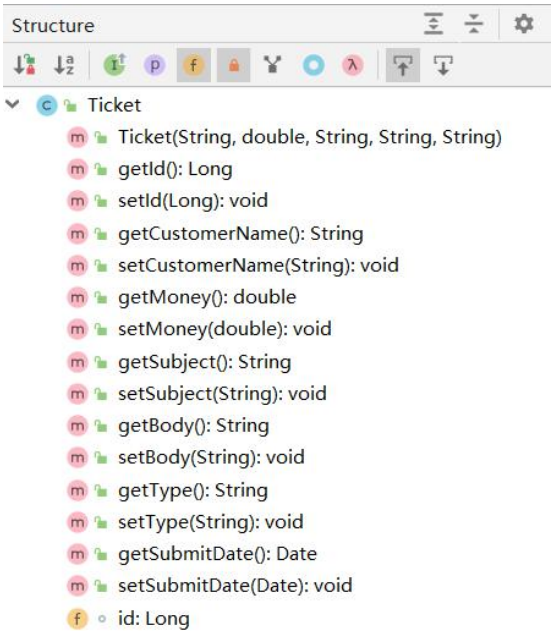
```
public class Ticket {
    Long id;            //票据id
    String customerName; //客户名称
    double money;       //票据金额
    String subject;     //票据主题
    String body;        //票据主要内容
    String type;
```





按住 shift 键，选择全部字段，单击确定，生成所有字段的 setter 和 getter 方法。

完成后代码界面如下图所示：



（2）补充完成 TicketDao 代码。（完成时间为____分钟）

要点注意：

（1）TicketList 用于保持票据列表，相当于一个简单的内存数据库，需要使用 static 关键词，设置为静态变量。

(2) TICKET_ID_SEQUENCE 用于保存当前最大的票据序号，每当插入一个新的票据，该变量也增加 1，需要设置为 static 静态变量。

(3) 新增票据函数使用关键词 synchronized，防止多个并发用户同时添加票据，序号出现错误。

```
public class TicketDao {
    //使用静态变量，只保存一份副本
    private  static  List<Ticket> ticketList  = new ArrayList<~>();;
    //静态变量，  票据ID号，全局唯一
    private static  volatile long TICKET_ID_SEQUENCE = 1000;
    //构造函数
    public  TicketDao(){
        //只初始化1次
        if(ticketList.size()==0) {
            //初始化数据
            for (int i = 0; i < 10; ++i) {
                Ticket ticket = new Ticket("中山领航科技公司", 5000 * i, "办公器材" + i,
                        "2020年新购置办公设备、电脑设备" + i, "银行汇票");

                ticket.setId(this.TICKET_ID_SEQUENCE);
                TICKET_ID_SEQUENCE++;
                ticket.setSubmitDate(new Date());
                ticketList.add(ticket);
            }
        }
    }
```

```
    //根据id查找ticket对象
    public Ticket findById(long id){
        for (Ticket ticket: ticketList) {
            if (ticket.getId() == id)
                return ticket;
        }
        return  null;
    }
    //查找所有的ticket对象
    public List findAll() { return  ticketList; }

    // 添加票据,使用同步加锁
    public synchronized void insert(Ticket ticket){
        //生成票据id
        TICKET_ID_SEQUENCE++;
        ticket.setId(TICKET_ID_SEQUENCE);
        ticketList.add(ticket);
        System.out.println(ticket+";" +ticketList.size());
    }
    //删除票据
    public void deleteById(long id) {
        for (Ticket ticket: ticketList) {
            if (ticket.getId() == id) {
                ticketList.remove(ticket);
                return;
            }
        }
    }
}
```

步骤 6：补充完成 TicketServlet 代码。（完成时间为____分钟）

要点注意：

（1）TicketServlet 入口函数为 doPost()和 doGet()方法。

（2）使用参数 action 保持用户的操作类型。action 的数据包括：create（新增票据），del（删除），view（查看详情），list（票据列表）。其中 view、list、del、create 以 url 形式传递参数，而 create 以表单的 POST 方法传递参数。

其中当 action=create 时，如果 HTTP 为 GET 请求时，表示跳转到添加票据的页面；如果 HTTP 为 POST 请求，则表示添加票据。

（3）理解操作完成后的页面跳转。

TicketServlet 代码如下：

```java
package jee.ticket.controller;

import ...

@WebServlet("/ticket")
public class TicketServlet extends HttpServlet {
    //处理GET请求
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
        req.setCharacterEncoding("UTF-8");
        String action = req.getParameter("action");
        if(action==null)
            action="list";
        switch (action) {
            case "view":
                this.viewTicket(req, resp);
                break;
            case "del":
                this.delTicket(req, resp);
                break;
            case "list":
                this.listTicket(req, resp);
                break;
            case "create":
                this.toCreate(req, resp);
                break;
            default:
                break;
        }
    }
```

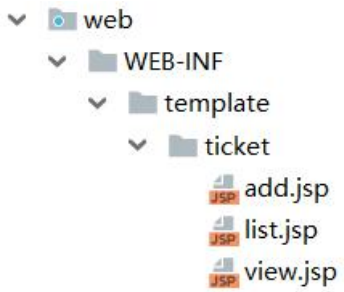
```java
    //处理POST请求
    @Override
    protected void doPost(HttpServletRequest req, HttpServletResponse resp)
            throws ServletException, IOException {
        req.setCharacterEncoding("UTF-8");
        String action = req.getParameter("action");
        switch (action) {
            case "create":
                this.createTicket(req, resp);
                break;
            default:
                break;
        }
    }

    //添加票据
    public void createTicket(HttpServletRequest req, HttpServletResponse resp)
            throws ServletException, IOException {
        TicketDao ticketDao = new TicketDao();
        //构成ticket对象
        String customerName = req.getParameter("customerName");
        double money = Double.parseDouble(req.getParameter("money"));
        String type = req.getParameter("type");
        String subject = req.getParameter("subject");
        String body = req.getParameter("body");
        Date submitDate = new Date();

        Ticket ticket = new Ticket(customerName, money, subject, body, type);
        ticket.setSubmitDate(submitDate);
        ticketDao.insert(ticket);
        //重定向到list URL
        resp.sendRedirect("/ticketAppDemo/ticket");
    }
```

```
//跳转到添加票据界面
public void toCreate(HttpServletRequest req,
                     HttpServletResponse resp) throws ServletException, IOException {
    req.getRequestDispatcher("/WEB-INF/template/ticket/add.jsp").forward(req, resp);
}

//删除票据
public void delTicket(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException {
    TicketDao ticketDao = new TicketDao();
    //获取票据id
    Long id = Long.parseLong(req.getParameter("id"));
    ticketDao.deleteById(id);
    //重定向到list URL
    resp.sendRedirect("/ticketAppDemo/ticket");
}

//查看票据列表
public void listTicket(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException {
    //获取票据列表
    TicketDao ticketDao = new TicketDao();
    List list = ticketDao.findAll();
    System.out.println(list);
    req.setAttribute("ticketList", list);
    req.getRequestDispatcher("/WEB-INF/template/ticket/list.jsp").forward(req, resp);
}

    //查看详情
    public void viewTicket(HttpServletRequest req, HttpServletResponse resp)
            throws ServletException, IOException {
        //获取票据id
        Long id = Long.parseLong(req.getParameter("id"));
        //调用dao的方法
        TicketDao ticketDao = new TicketDao();
        Ticket ticket = ticketDao.findById(id);
        //
        req.setAttribute("ticket", ticket);
        req.getRequestDispatcher("/WEB-INF/template/ticket/view.jsp").forward(req, resp);
    }

}
```

问题 1：当 action 参数为空时，默认执行什么动作？

问题 2：哪些操作是需要使用 doPost 来完成的？

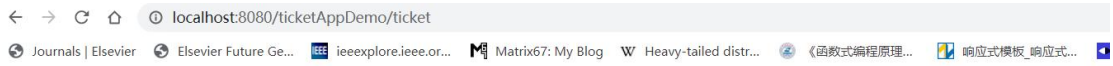
步骤 7：完成视图层代码(完成时间为____分钟)



（1）list.jsp 代码如下：

```jsp
<%@ page import="jee.ticket.domain.Ticket" %>
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>中山领航科技有限公司客服系统</title>
</head>
<body>

<h2>票据列表</h2>
<a href="/ticketAppDemo/ticket?action=create">添加新票据</a><br /><br />
<hr>
<table border="1" cellspacing="0">
    <tr>
        <th>序號</th>
        <th>票据Id</th>
        <th>企业名</th>
        <th>总金额</th>
        <th>主题</th>
        <th>类型</th>
        <th>提交时间</th>
        <th>主要内容</th>
        <th>操作</th>
    </tr>

<c:forEach items="${ticketList}" var="ticket" varStatus="s">
    <tr>
        <td>${s.count}</td>
        <td>${ticket.id}</td>
        <td>${ticket.customerName}</td>
        <td>${ticket.money}</td>
        <td>${ticket.subject}</td>
        <td>${ticket.type}</td>
        <td>${ticket.submitDate}</td>
        <td>${ticket.body}</td>
        <td>
            <a href="/ticketAppDemo/ticket?action=view&id=${ticket.id}">详情</a>
            <a href="/ticketAppDemo/ticket?action=del&id=${ticket.id}">删除</a>
            <a href="/ticketAppDemo/ticket?action=edit&id=${ticket.id}">编辑</a>
            <a href="/ticketAppDemo/ticket?action=audit&id=${ticket.id}">审核</a>
        </td>

    </tr>
</c:forEach>
</table>

<hr>@zsc 电子科技大学中山学院 计算机学院 班级:xxx, 学号:xx, 姓名:xxx  <p></p>
</body>
</html>
```

注意：上述代码中 foreach 标签的使用。完成代码后测试界面如下所示：



票据列表

添加新票据

序號	票据Id	企业名	总金额	主题	类型	提交时间	主要内容	操作
1	1000	中山领航科技公司	0.0	办公器材0	银行汇票	Tue Mar 24 10:45:09 CST 2020	2020年新购置办公设备、电脑设备0	详情 删除 编辑 审核
2	1001	中山领航科技公司	5000.0	办公器材1	银行汇票	Tue Mar 24 10:45:09 CST 2020	2020年新购置办公设备、电脑设备1	详情 删除 编辑 审核
3	1002	中山领航科技公司	10000.0	办公器材2	银行汇票	Tue Mar 24 10:45:09 CST 2020	2020年新购置办公设备、电脑设备2	详情 删除 编辑 审核
4	1003	中山领航科技公司	15000.0	办公器材3	银行汇票	Tue Mar 24 10:45:09 CST 2020	2020年新购置办公设备、电脑设备3	详情 删除 编辑 审核
5	1004	中山领航科技公司	20000.0	办公器材4	银行汇票	Tue Mar 24 10:45:09 CST 2020	2020年新购置办公设备、电脑设备4	详情 删除 编辑 审核
6	1005	中山领航科技公司	25000.0	办公器材5	银行汇票	Tue Mar 24 10:45:09 CST 2020	2020年新购置办公设备、电脑设备5	详情 删除 编辑 审核
7	1006	中山领航科技公司	30000.0	办公器材6	银行汇票	Tue Mar 24 10:45:09 CST 2020	2020年新购置办公设备、电脑设备6	详情 删除 编辑 审核
8	1007	中山领航科技公司	35000.0	办公器材7	银行汇票	Tue Mar 24 10:45:09 CST 2020	2020年新购置办公设备、电脑设备7	详情 删除 编辑 审核
9	1008	中山领航科技公司	40000.0	办公器材8	银行汇票	Tue Mar 24 10:45:09 CST 2020	2020年新购置办公设备、电脑设备8	详情 删除 编辑 审核
10	1009	中山领航科技公司	45000.0	办公器材9	银行汇票	Tue Mar 24 10:45:09 CST 2020	2020年新购置办公设备、电脑设备9	详情 删除 编辑 审核

@zsc 电子科技大学中山学院 计算机学院 班级:xxx, 学号:xx, 姓名:xxx

（2）add. jsp 代码如下：

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>添加票据</title>
</head>
<body>
<form method="POST" action="/ticketAppDemo/ticket ">
    <h2>添加新票据</h2>
    <!--用隐藏字段传递action的类型-->
    <input type="hidden" name="action" value="create">
    客户名称<br/>
    <input type="text" name="customerName" value="电子科技大学中山学院"><br/><br/>
    票据主题<br/>
    <input type="text" name="subject" value="2018年设备采购"><br/><br/>
    总金额<br/>
    <input type="text" name="money" value="20000"><br/><br/>
    类型<br/>
    <select  name="type">
        <option value="银行汇票" selected>银行汇票</option>
        <option value="商业汇票">商业汇票</option>
        <option value="商业本票">商业本票</option>
        <option value="银行本票">银行本票</option>
        <option value="转帐支票">转帐支票</option>
    </select>
    <br/><br/>
    票据内容<br/>
    <textarea name="body" rows="5" cols="30">购买办公用品、服务器1批，共10000元...
            </textarea><br/><br/>
    <input type="submit" value="提交"/>
</form>

<p>
    <a href="/ticketAppDemo/ticket">返回票据列表</a><br /><br />
<hr>@zsc 电子科技大学中山学院 计算机学院 班级:xxx, 学号:xx, 姓名:xxx <p></p>
</body>
</html>
```

（3）view.jsp 代码如下：

```
<%@ page import="jee.ticket.domain.Ticket" %>
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>中山领航科技有限公司客服系统</title>
</head>
<body>


<h2>票据详情</h2><hr>
    票据Id: ${ticket.id}<p>
    企业名字: ${ticket.customerName}  <p>
    票据总金额: ${ticket.money} <p>
    主题:   ${ticket.subject}  <p>
    主要内容:  ${ticket.body} <p>

    <a href="/ticketAppDemo/ticket">返回票据列表</a><br /><br />

<hr>@zsc 电子科技大学中山学院 计算机学院 班级:xxx, 学号:xx, 姓名:xxx  <p></p>
</body>
</html>
```

测试界面如下：

要求：修改页尾，显示自己的学号和姓名。

课后思考题：

1. 思考如何实现票据编辑操作？
2. 为什么重启服务器所有票据数据都会消失？如何实现数据持久化？