



# 物流管理系统

# 设计报告

学院：软件学院

专业：软件工程

姓名：张启洋

学号：20301090

创建时间：2023/6/20

---

# 目录

1 概要设计 .....	3
1.1 功能设计 .....	3
1.2 系统架构 .....	4
2 接口设计 .....	5
2.1 基础配置 .....	5
(1) 注册账户 .....	5
(2) 登录账号 .....	6
(3) 登出 .....	7
(4) 查询员工 .....	8
(4) 查询用户 .....	9
(5) 修改用户信息 .....	11
(6) 修改员工信息 .....	12
(7) 批量删除用户 .....	14
(8) 增加普通用户 .....	15
(9) 增加员工用户 .....	16
(10) 查找车辆列表 .....	17
(11) 新增车辆 .....	19
(12) 修改车辆 .....	20
(13) 删除车辆 .....	21
2.2 订单管理 .....	22
(1) 创建订单 .....	22
(2) 计算价格，为订单分配车辆和驾驶人 .....	24
(3) 改变订单状态 .....	25
(4) 添加订单中转站 .....	27
(5) 查找快递员负责的订单 .....	28
(6) 根据 ID 查找订单 .....	30

---

(7) 获取订单列表 .....	32
(8) 查询订单运输中转信息 .....	33
3 页面设计 .....	35
4 单元测试 .....	44
4.1 创建订单 .....	44
4.2 更新订单状态 .....	45
4.3 添加转运信息 .....	46

# 1 概要设计

## 1.1 功能设计

本系统是一个简易的物流系统，系统设置了三个角色——用户、承运人员和管理员，根据角色对页面访问权限进行控制。系统实现了用户管理、车辆管理、订单管理、员工管理。

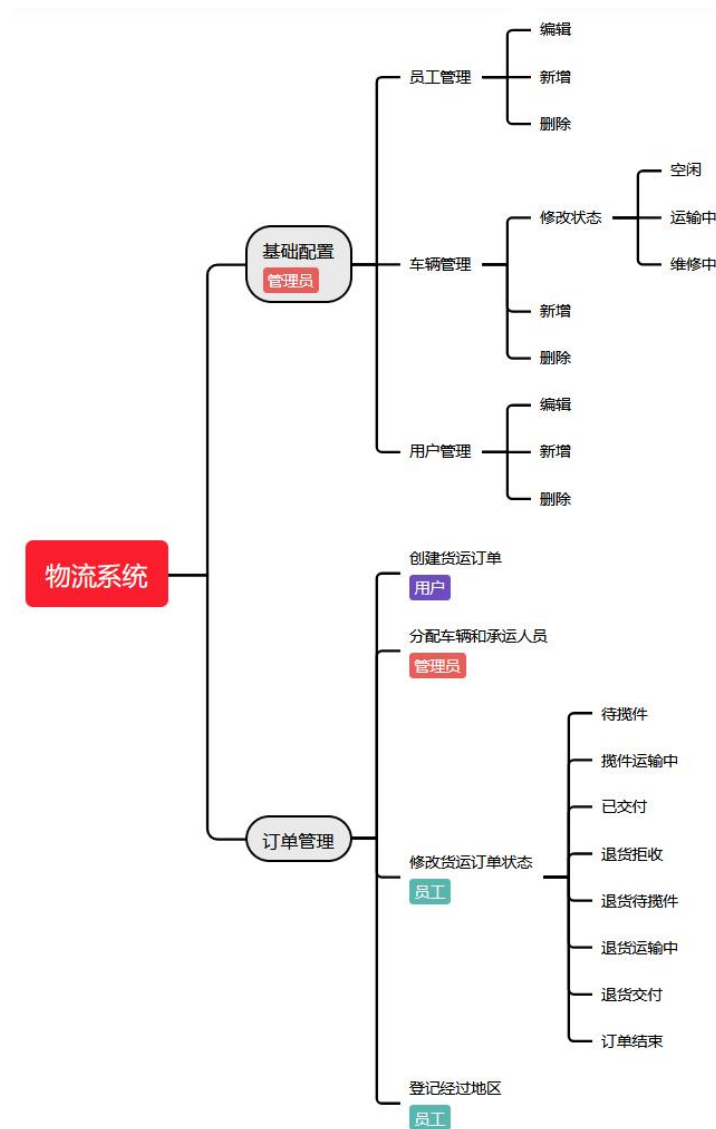
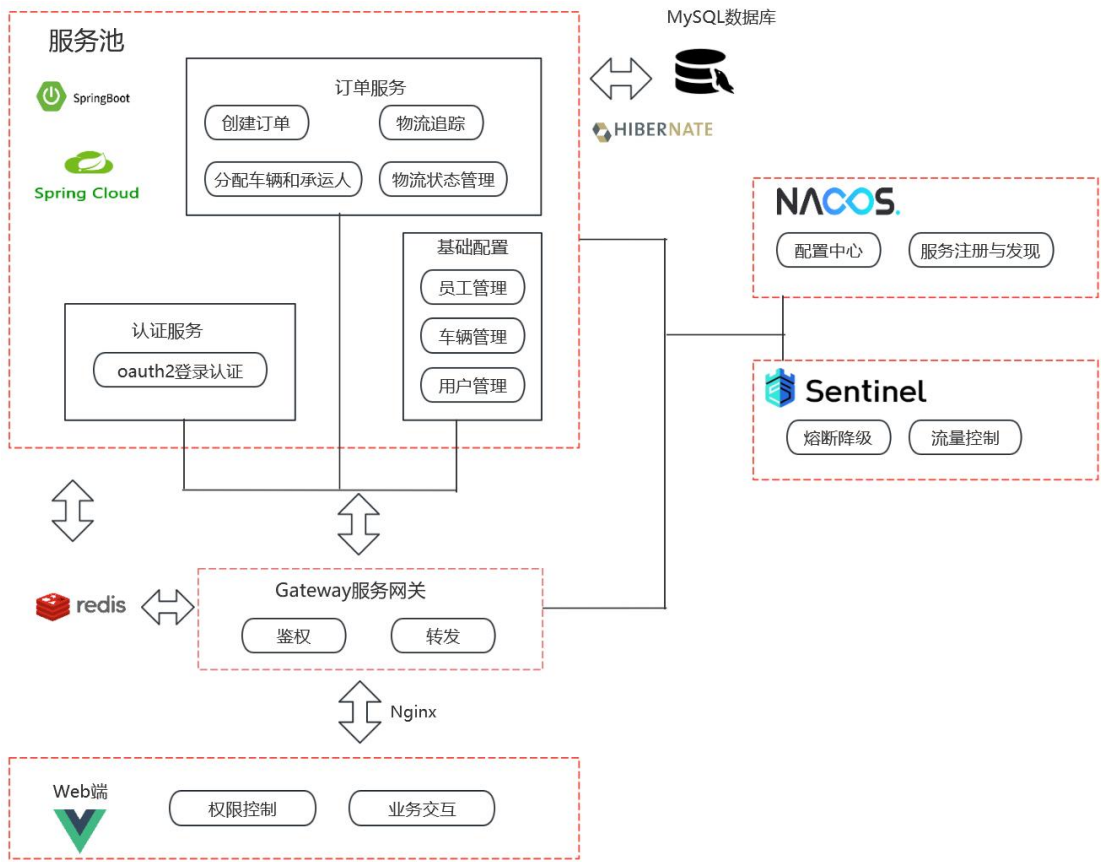


图 1 功能设计图

## 1.2 系统架构

本系统使用springboot+mybatis-plus+hibernate+vue进行Web应用程序开发，使用Restful API和Ajax设计开发系统控制层。在此基础上进行微服务拆分，使用nacos注册中心实现服务的注册与发现，并通过nacos-config实现配置中心；使用ouath2 实现单点登录与授权；通过gateway网关对外部客户端进行请求转发与权限校验。系统微服务包括服务网关，认证服务器和外部服务、库管服务两个资源服务，架构图如下所示。



该项目选择IDEA作为开发工具，数据库使用MySQL进行搭建，并利用Redis进行访问权限资源缓存。

## 2 接口设计

### 2.1 基础配置

#### (1) 注册账户

创建个人账户，可以选择注册成为哪种角色。

URL	http://localhost:8080/auth/signup
请求方式	POST

请求参数：

参数名	含义	规则说明	是否必需	缺省值
username	用户名	无	是	无
phone	电话号码	11 位数字串	是	无
password	密码	至少为六位字符串	是	无
role	角色	数组类型，可选值 0 或 1。 1：员工 0：普通用户	是	无
realName	真实姓名	中文字符	否，员工必需	无
sex	性别	数字，可选值 1 或 0。 1：女 0：男	否，员工必需	无
entryTime	入职时间	格 式 ： “YYYY-MM-DD”	否，员工必需	无
birthday	生日	格 式 ： “YYYY-MM-DD”	否，员工必需	无

返回结果：

名称	类型	说明
code	String	访问状态码
message	String	状态码说明
data	Object	返回的用户数据
roles	Array	返回用户的角色信息
role	String	用户角色名
name	String	用户名
Phone	String	电话号码

截图：

POST

http://localhost:8080/auth/signup

H 请求头

参数

☐ x-www-form-urlencoded;charset=UTF-8

☒ 自定义格式

application/json

```
{
  "userName": "xxx",
  "phone": "12312344321",
  "password": "123456",
  "realName": "小测试",
  "role": [1],
  "sex": 0,
  "entryTime": "2023-06-12",
  "birthday": "1999-09-09"
}
```

☒ 响应体

H 响应头

Cookie

原始数据

格式化

JSON: 

全部展开

展开一级

展开二级

展开三级

展开四级

自动换行

帮助文档

```
{
  code: 200,
  message: "Success",
  - data: {
    phone: "12312344321",
    - roles: [
      - {
        role: "ROLE_CARRIER"
      }
    ],
    name: "xxx"
  }
}
```

(2) 登录账号

登录个人账号。

登录个人账号 API 服务地址：

URL	http://localhost:8080/auth/login
请求方式	POST

请求参数：

参数名	含义	规则说明	是否必须	缺省值
username	用户名	无	是	无





code	String	访问状态码
message	String	状态码说明

截图：

POST

http://localhost:8080/auth/logout

H 请求头

参数

☐ x-www-form-urlencoded; charset=UTF-8

☒ 自定义格式

application

☒ 响应体

H 响应头

Cookie

原始数据

格式化

JSON: 

全部展开

展开一级

展开二级

展开三级

```
{
  code: 200,
  message: "登出成功",
  data: {}
}
```

(4) 查询员工

根据用户名和真实姓名模糊查询普通用户。

登出个人账号 API 服务地址：

URL	http://localhost:8080/user/carrierList
请求方式	GET

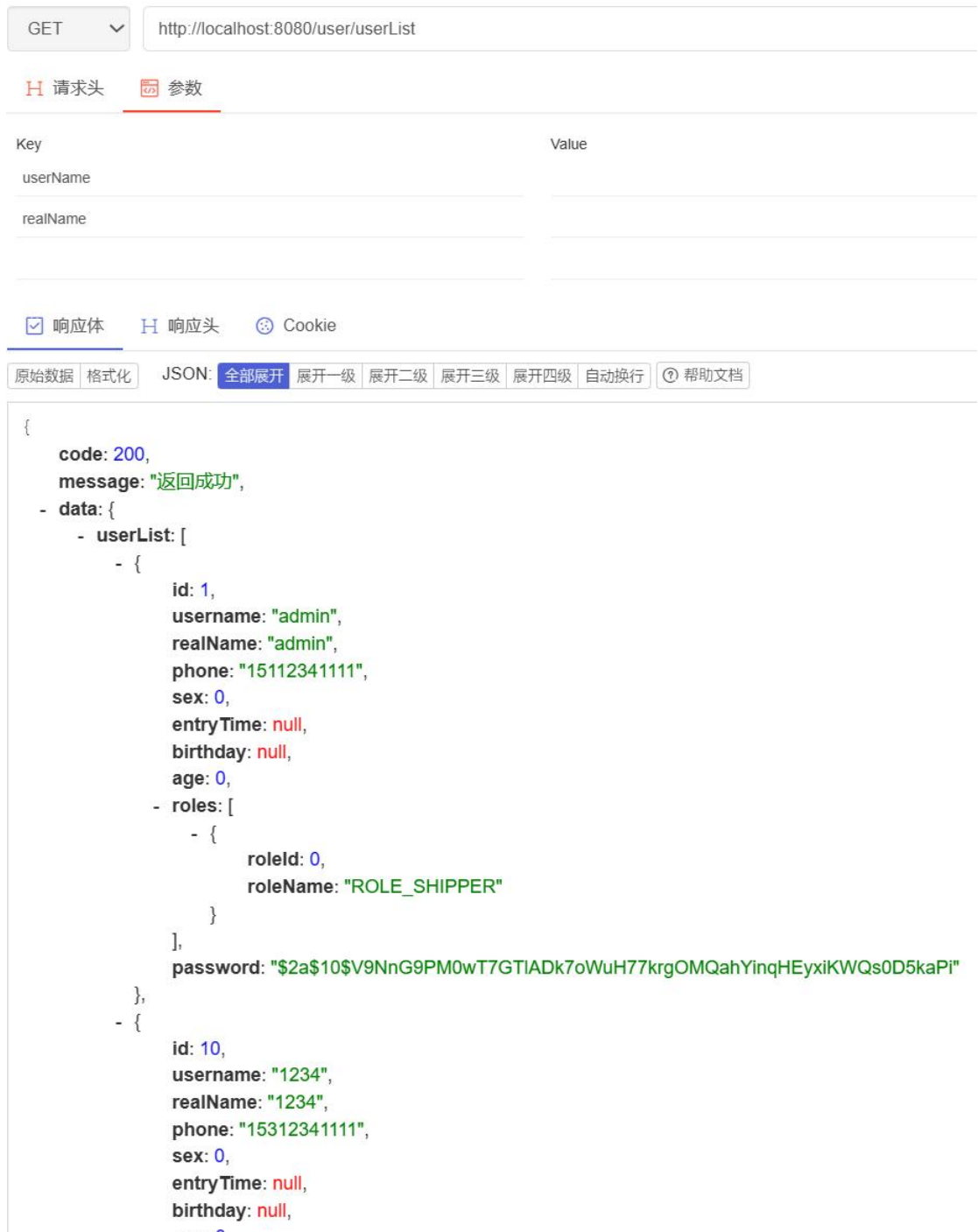
请求参数：

参数名	含义	规则说明	是否必须	缺省值
userName	用户名	无	否	无
realName	真实姓名	中文字符串	否	无

返回结果：

名称	类型	说明
code	String	访问状态码
message	String	状态码说明
data	Object	返回的用户数据
carrierList	Array	返回员工列表

截图：



#### (4) 查询用户

根据用户名和真实姓名模糊查询普通用户。

登出个人账号 API 服务地址：

URL	http://localhost:8080/user/userList
请求方式	GET

---

请求参数:

参数名	含义	规则说明	是否必须	缺省值
userName	用户名	无	否	无
realName	真实姓名	中文字符串	否	无

返回结果:

名称	类型	说明
code	String	访问状态码
message	String	状态码说明
data	Object	返回的用户数据
userList	Array	返回用户的角色信息

截图:

GET

http://localhost:8080/user/userList

H 请求头

参数

Key	Value
userName	
realName	

☒ 响应体

H 响应头

Cookie

原始数据

格式化

JSON: 全部展开 展开一级 展开二级 展开三级 展开四级 自动换行 帮助文档

```

{
  code: 200,
  message: "返回成功",
  data: {
    userList: [
      {
        id: 1,
        username: "admin",
        realName: "admin",
        phone: "15112341111",
        sex: 0,
        entryTime: null,
        birthday: null,
        age: 0,
        roles: [
          {
            roleId: 0,
            roleName: "ROLE_SHIPPER"
          }
        ],
        password: "$2a$10$V9NnG9PM0wT7GTIADk7oWuH77krgOMQahYinqHEyxikWQs0D5kaPI"
      },
      {
        id: 10,
        username: "1234",
        realName: "1234",
        phone: "15312341111",
        sex: 0,
        entryTime: null,
        birthday: null,
        age: 0,
        roles: [
          {
            roleId: 0,
            roleName: "ROLE_SHIPPER"
          }
        ],
        password: "$2a$10$V9NnG9PM0wT7GTIADk7oWuH77krgOMQahYinqHEyxikWQs0D5kaPI"
      }
    ]
  }
}

```

## （5）修改用户信息

修改普通用户信息。

修改用户信息 API 服务地址：

URL	http://localhost:8080/user/modifyUser
请求方式	POST

请求参数：

参数名	含义	规则说明	是否必须	缺省值
id	用户 ID	无	是	无
username	用户名	无	是	无

realName	真实姓名	中文字符串	是	无
phone	电话号码	11 位数字串	是	无
password	密码	不少于 6 位字符串	是	无

返回结果:

名称	类型	说明
code	String	访问状态码
message	String	状态码说明
data	Object	

截图:

POST

http://localhost:8080/user/modifyUser

H 请求头

参数

☐ x-www-form-urlencoded; charset=UTF-8

☒ 自定义格式

applicati

```
{
  "id": 1,
  "username": "modify",
  "realName": "改名了",
  "phone": "2222222222",
  "password": "123456"
}
```

☒ 响应体

H 响应头

☐ Cookie

原始数据

格式化

JSON: 

全部展开

展开一级

展开二级

展开三

```
{
  code: 200,
  message: "用户信息更新成功",
  data: {}
}
```

(6) 修改员工信息

修改员工信息。

修改员工信息 API 服务地址:

URL	http://localhost:8080/user/modifyCarrier
请求方式	POST

请求参数:

参数名	含义	规则说明	是否必须	缺省值
-----	----	------	------	-----

id	用户 ID	无	是	无
username	用户名	无	是	无
realName	真实姓名	中文字符串	是	无
phone	电话号码	11 位数字串	是	无
password	密码	不少于 6 位字符串	是	无
entryTime	入职时间	格 式 : “YYYY-MM-DD”	是	无
birthday	生日	格 式 : “YYYY-MM-DD”	是	无
sex	性别	数字	是	无

返回结果:

名称	类型	说明
code	String	访问状态码
message	String	状态码说明
data	Object	

截图:

POST
http://localhost:8080/user/modifyCarrier

H 请求头
参数

☐ x-www-form-urlencoded; charset=UTF-8
☒ 自定义格式
application/json

```

{
  "id": 49,
  "username": "modify2",
  "realName": "改名了",
  "phone": "2222222211",
  "password": "123456",
  "entryTime": "2022-09-09",
  "birthday": "1998-09-09",
  "sex": 1
}

```

☒ 响应体
H 响应头
Cookie

原始数据
格式化
JSON:
全部展开
展开一级
展开二级
展开三级
展开四级

```

{
  code: 200,
  message: "員工信息更新成功",
  data: {}
}

```

---

(7) 批量删除用户

批量删除用户，可删除普通用户和员工。

批量删除用户 API 服务地址：

URL	http://localhost:8080/user/deleteUsers
请求方式	DELETE

请求参数：

参数名	含义	规则说明	是否必须	缺省值
ids	需要删除的用户 ID	数组	是	无

返回结果：

名称	类型	说明
code	String	访问状态码
message	String	状态码说明
data	Object	

截图：

DELETE
▼
http://localhost:8080/user/deleteUsers

H 请求头
参数

☐ x-www-form-urlencoded;charset=UTF-8
☒ 自定义格式
application/

```

{
  "ids": [55]
}

```

☒ 响应体
H 响应头
Cookie

原始数据
格式化
JSON:
全部展开
展开一级
展开二级
展开三级

```

{
  code: 200,
  message: "删除成功",
  data: {}
}

```

#### (8) 增加普通用户

增加普通用户。

增加普通用户 API 服务地址：

URL	http://localhost:8080/user/addUser
请求方式	POST

请求参数：

参数名	含义	规则说明	是否必须	缺省值
username	用户名	无	是	无
phone	电话号码	11 位数字串	是	无
password	密码	至少为六位字符串	是	无

返回结果：

名称	类型	说明
code	String	访问状态码



message	String	状态码说明
data	Object	

截图：

The screenshot shows a REST client interface. At the top, the method is set to POST and the URL is http://localhost:8080/user/addUser. Below this, there are tabs for '请求头' (Headers) and '参数' (Parameters). The '请求头' tab is selected, showing 'x-www-form-urlencoded;charset=UTF-8'. The '参数' tab is also visible. Below the tabs, there is a large text area for the request body, containing a JSON object: { "username": "modify", "phone": "98778997899", "password": "123456" }. At the bottom, there are tabs for '响应体' (Response Body), '响应头' (Response Headers), and 'Cookie'. The '响应体' tab is selected, showing a JSON object: { "code": 103, "message": "该用户名已存在", "data": {} }. The 'code' is highlighted in blue, and the 'message' is highlighted in green.

#### (9) 增加员工用户

增加员工用户。

增加员工用户 API 服务地址：

URL	http://localhost:8080/user/addCarrier
请求方式	POST

请求参数：

参数名	含义	规则说明	是否必须	缺省值
id	用户 ID	无	是	无
username	用户名	无	是	无
realName	真实姓名	中文字符串	是	无
phone	电话号码	11 位数字串	是	无

password	密码	不少于 6 位字符串	是	无
entryTime	入职时间	格 式 : “YYYY-MM-DD”	是	无
birthday	生日	格 式 : “YYYY-MM-DD”	是	无

返回结果:

名称	类型	说明
code	String	访问状态码
message	String	状态码说明
data	Object	

截图:

POST
▼
http://localhost:8080/user/addUser

H 请求头
参数

☐ x-www-form-urlencoded;charset=UTF-8
☒ 自定义格式
a

```
{
  "username": "modify",
  "phone": "98778997899",
  "password": "123456"
}
```

☒ 响应体
H 响应头
Cookie

原始数据
格式化
JSON:
全部展开
展开一级
展开二级

```
{
  code: 103,
  message: "该用户名已存在",
  data: {}
}
```

(10) 查找车辆列表  
模糊查询车辆列表。

---

查找车辆列表 API 服务地址:

URL	http://localhost:8080/van/vanList
请求方式	GET

请求参数:

参数名	含义	规则说明	是否必须	缺省值
license	牌照	无	否	无
state	状态	数字 0: 空闲 1: 使用中 2: 损坏	否	无

返回结果:

名称	类型	说明
code	String	访问状态码
message	String	状态码说明
data	Object	
vanList	Array	返回车辆信息

截图:

GET

http://localhost:8080/van/vanList

H 请求头

参数

Key

Value

license

state

☒ 响应体

H 响应头

Cookie

原始数据

格式化

JSON: 全部展开 展开一级 展开二级 展开三级

```

{
  code: 200,
  message: "返回成功",
  data: {
    vanList: [
      {
        id: 24,
        maxLoad: 2000,
        license: "京ASDFG",
        state: 1,
        info: "野马汽车"
      },
      {
        id: 28,
        maxLoad: 54645,
        license: "京ASDFA",
        state: 1,
        info: "342432"
      },
      + { ... },
      + { ... }
    ]
  }
}

```

(11) 新增车辆

新增车辆。

新增车辆 API 服务地址：

URL	http://localhost:8080/van/addVan
请求方式	POST

请求参数：

参数名	含义	规则说明	是否必须	缺省值
maxLoad	最大载重	数字	是	无
license	牌照	字符串	是	无
state	状态	数字 0：空闲 1：使用中 2：损坏	是	无

info	备注	字符串	否	无
------	----	-----	---	---

返回结果:

名称	类型	说明
code	String	访问状态码
message	String	状态码说明
data	Object	

截图:



## (12) 修改车辆

修改车辆。

修改车辆 API 服务地址:

URL	http://localhost:8080/van/modifyVan
请求方式	POST

请求参数:

参数名	含义	规则说明	是否必须	缺省值
maxLoad	最大载重	数字	是	无
license	牌照	字符串	是	无
state	状态	数字 0: 空闲 1: 使用中	是	无

		2: 损坏		
info	备注	字符串	否	无
id	车辆 ID	表 vans 中已有记录的 ID	是	无

返回结果:

名称	类型	说明
code	String	访问状态码
message	String	状态码说明
data	Object	

截图:

The screenshot shows a REST client interface. At the top, a POST request is configured to `http://localhost:8080/van/modifyVan`. Below the URL bar, there are tabs for '请求头' (Headers) and '参数' (Parameters). The '请求头' tab is selected, showing the content type `application/json`. The request body is a JSON object: `{ "id": 59, "maxLoad": 3000, "state": 0, "license": "京BJ13D", "info": "金云货车" }`. Below the request, there are tabs for '响应体' (Response Body), '响应头' (Response Headers), and 'Cookie'. The '响应体' tab is selected, showing the response in JSON format: `{ "code": 200, "message": "车辆信息更新成功", "data": {} }`. The response is displayed in a code editor with syntax highlighting.

### (13) 删除车辆

根据汽车 ID 删除车辆。

删除车辆 API 服务地址:

URL	<code>http://localhost:8080/van/deleteVans</code>
请求方式	DELETE

请求参数:

参数名	含义	规则说明	是否必须	缺省值
ids	车辆 ID	数组	是	无

返回结果:

名称	类型	说明
code	String	访问状态码
message	String	状态码说明
data	Object	

截图:

DELETE

http://localhost:8080/van/deleteVans

H 请求头

参数

☐ x-www-form-urlencoded;charset=UTF-8

☒ 自定义格式

application/json

```
{
  "ids": [59]
}
```

☒ 响应体

H 响应头

☐ Cookie

原始数据

格式化

JSON: 

全部展开

展开一级

展开二级

展开三级

展开四级

```
{
  code: 200,
  message: "删除车辆成功",
  data: {}
}
```

## 2.2 订单管理

### (1) 创建订单

用户创建订单。

创建订单 API 服务地址:

URL	http://localhost:8080/order/create
-----	------------------------------------

请求方式	POST
------	------

请求参数：

参数名	含义	规则说明	是否必须	缺省值
shipperId	创建订单用户 ID	必须是 users 表中已有的记录 id	是	无
start	出发地	无	是	无
destination	目的地	无	是	无
receiver	收件人	无	是	无
weight	重量	数字，单位 kg	是	无
info	信息	无	是	无

返回结果：

名称	类型	说明
code	String	访问状态码
message	String	状态码说明
data	Object	

截图：



POST

http://localhost:8080/order/create

H 请求头

参数

☐ x-www-form-urlencoded; charset=UTF-8
 ☒ 自定义格式
 

ap

```

{
  "shipperId": "10",
  "start": "北京xx中心",
  "destination": "浙江xx仓库",
  "receiver": "刘xx",
  "weight": 4000,
  "info": "建筑钢材"
}

```

☒ 响应体
 

H 响应头

Cookie

原始数据 格式化

JSON: 全部展开 展开一级 展开二级

```

{
  code: 200,
  message: "Success",
  data: {}
}

```

(2) 计算价格，为订单分配车辆和驾驶人

为用户创建的订单分配车辆和驾驶人，统计所需运费。

添加取件人 API 服务地址：

URL	http://localhost:8080/order/update
请求方式	POST

请求参数：

参数名	含义	规则说明	是否必须	缺省值
id	订单 ID	必须是 orders 表中已有的记录 id	是	无
carrierIds	驾驶员 ID	必须是 users 表中已有的记录 id 且角色为 carrier	是	无

money	运费	数字	是	无
vanIds	货车 ID	必须是 vans 表中已有记录 id	是	无

返回结果：

名称	类型	说明
code	String	访问状态码
message	String	状态码说明
data	Object	

截图：

POST http://localhost:8080/order/update

**H** 请求头 **参数**

☐ x-www-form-urlencoded; charset=UTF-8 ☒ 自定义格式 **ap**

```
{
  "id": 51,
  "carrierIds": [7],
  "money": 5000,
  "vanIds": [29]
}
```

☒ 响应体 **H** 响应头 ☐ Cookie

原始数据 格式化 **JSON:** 全部展开 展开一级 展开二级

```
{
  code: 200,
  message: "Update successfully!",
  data: {}
}
```

### (3) 改变订单状态

改变订单的状态。

创建订单 API 服务地址：

URL	http://localhost:8080/order/changeState
请求方式	POST

请求参数：

参数名	含义	规则说明	是否必须	缺省值
-----	----	------	------	-----

id	订单 ID	必须是 orders 表中已有的记录 id	是	无
state	状态值	可选值如下： 1: 已付款待揽件 2: 揽件运输中 3: 已交付 4: 退货拒收 5: 订单结束 40: 退货待揽件 41: 退货运输中 42: 退货交付	是	无

返回结果：

名称	类型	说明
code	String	访问状态码
message	String	状态码说明
data	Object	

截图：

POST
▼
http://localhost:8080/order/changeState

H 请求头
参数

☐ x-www-form-urlencoded; charset=UTF-8
☒ 自定义格式
applica

```
{
  "id": 51,
  "state": 1
}
```

☒ 响应体
H 响应头
Cookie

原始数据
格式化
JSON:
全部展开
展开一级
展开二级
展开三

```
{
  code: 200,
  message: "Update successfully!",
  data: {}
}
```

#### (4) 添加订单中转站

为运输中的订单添加其到达的中转站。

添加订单中转站 API 服务地址：

URL	http://localhost:8080/order/transfer
请求方式	POST

请求参数：

参数名	含义	规则说明	是否必须	缺省值
Id	订单 ID	必须是 orders 表中已有的记录 id	是	无
loca	中转站地名	无	是	无

返回结果：

名称	类型	说明
code	String	访问状态码
message	String	状态码说明
data	Object	

截图：

POST ☐ http://localhost:8080/order/transfer

**H** 请求头 **参数**

☐ x-www-form-urlencoded; charset=UTF-8 ☒ 自定义格式 **appl**

```
{
  "id": 51,
  "loca": "河北"
}
```

☒ 响应体 **H** 响应头 **Cookie**

原始数据 格式化 **JSON:** **全部展开** 展开一级 展开二级 展

```
{
  code: 200,
  message: "Update successfully!",
  data: {}
}
```

(5) 查找快递员负责的订单

根据快递员 id 查找快递员负责的订单记录。

查找快递员负责的订单 API 服务地址：

URL	http://localhost:8080/api/order/searchCarrier/{carrierId}
请求方式	GET

请求参数：

参数名	含义	规则说明	是否必须	缺省值
carrierId	驾驶人 ID	必须是 users 表中已有的记录 id 且角色为 carrier	是	无

返回结果：

名称	类型	说明
code	String	访问状态码
message	String	状态码说明
data	Object	返回的用户数据
totalMoney	String	总金额
orders	Object	返回运输订单信息
money	String	用户角色名
from	String	订单出发地
to	String	订单目的地
id	String	订单 ID

截图：

GET
http://localhost:8080/order/searchCarrier

请求头

参数

Key	Value
carrierId	7

响应体

响应头

Cookie

原始数据

格式化

JSON:

全部展开

展开一级

展开二级

展开三级

展开四级

自动换行

帮助文档

(6) 根据 ID 查找订单

根据订单 id 查找订单。

查找订单 API 服务地址：

URL	http://localhost:8080/api/order/searchId/{orderId}
请求方式	GET

请求参数：

参数名	含义	规则说明	是否必须	缺省值
orderId	订单 ID	必须是 orders 表中已有的记录 id	是	无

返回结果：

名称	类型	说明
code	String	访问状态码
message	String	状态码说明
data	Object	返回的用户数据
totalMoney	String	总金额
orders	Object	返回运输订单信息
money	String	用户角色名
from	String	订单出发地
to	String	订单目的地
id	String	订单 ID

截图：

GET
http://localhost:8080/order/searchId

请求头
参数

Key	Value
orderId	51

响应体
响应头
Cookie

原始数据
格式化
JSON:
全部展开
展开一级
展开二级
展开三级
展开四级
自动换行
帮助文档

```

{
  code: 200,
  message: "Success",
  data: {
    order: {
      id: 51,
      createDate: "2023-08-21",
      shipper: {
        id: 10,
        username: "1234",
        realName: "1234",
        phone: "15312341111",
        sex: 0,
        entryTime: null,
        birthday: null,
        age: 0,
        roles: [
          {
            roleId: 0,
            roleName: "ROLE_SHIPPER"
          }
        ],
        password: "$2a$10$.MZxArc0iXdDWWKkNsR.JlujLqLjP.Vykifu.J8wblFNu.dl/C5Lji"
      },
      carriers: [
        {
          id: 7,
          username: "jun",
          realName: "肖俊",
          phone: "15112333323",
          sex: 0,
          entryTime: "2023-05-21",
          birthday: "1999-08-08",
          age: 0,
          roles: [
            {
              roleId: 1,
              roleName: "ROLE_CARRIER"
            }
          ],
          password: "$2a$10$DRrCqleYTwIAloBKW6iQ2.mzVrhZ4VU8FY8LGyZqb4UzRICBajacy"
        }
      ],
      vans: [
        {
          id: 20,
          maxLoad: 4000,
          license: "京ASDF1",
          state: 1,
          info: "大货车"
        }
      ],
      transfers: [],
      destination: "浙江xx仓库",
      state: 1,
      info: "建筑材料",
      money: 5000,
      start: "北京xx中心",
      weight: 4000,
      receiver: "文xx"
    }
  }
}

```



(7) 获取订单列表

根据发货人用户名、驾驶人用户名、汽车牌照模糊查询订单。

查询订单运输中转信息 API 服务地址:

URL	http://localhost:8080/api/order/orderList
请求方式	GET

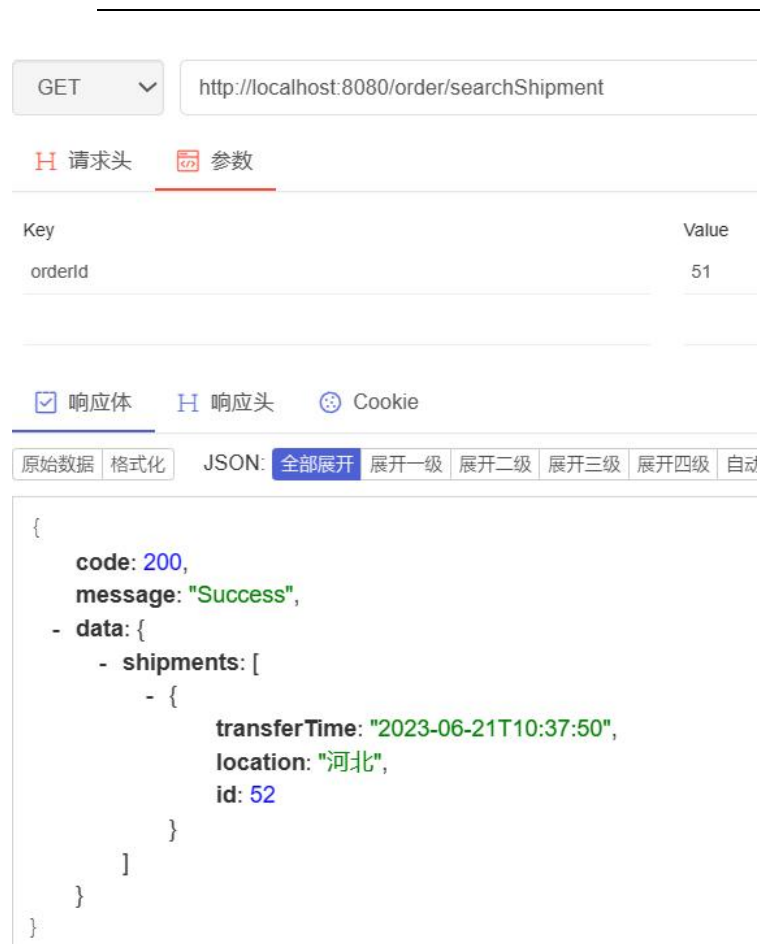
请求参数:

参数名	含义	规则说明	是否必须	缺省值
shipperName	发货人用户名	必须是 users 表中已有的记录 id 且角色为 shipper	否	无
carrierName	驾驶人用户名	必须是 users 表中已有的记录 id 且角色为 carrier	否	无
vanLicense	汽车牌照	必须是 vans 表中已有的记录 license	否	无

返回结果:

名称	类型	说明
code	String	访问状态码
message	String	状态码说明
data	Object	返回的用户数据
shipments	Object	返回运输订单中转信息
transferTime	String	中转时间
location	String	中转地名
id	String	中转记录 ID

截图:



#### (8) 查询订单运输中转信息

根据订单 ID 查询订单的运输中转信息。

查询订单运输中转信息 API 服务地址：

URL	http://localhost:8080/api/order/searchShipment/{orderId}
请求方式	GET

请求参数：

参数名	含义	规则说明	是否必须	缺省值
orderId	订单 ID	必须是 orders 表中已有的记录 id	是	无

返回结果：

名称	类型	说明
code	String	访问状态码
message	String	状态码说明
data	Object	返回的用户数据
shipments	Object	返回运输订单中转信息
transferTime	String	中转时间

	location	String	中转地名
	id	String	中转记录 ID

截图：

GET

http://localhost:8080/order/orderList

H 请求头

参数

Key	Value
shipperName	
carrierName	
vanLicense	

☒ 响应体

H 响应头

Cookie

原始数据

格式化

JSON:

全部展开

展开一级

展开二级

展开三级

展开四级

自动换行

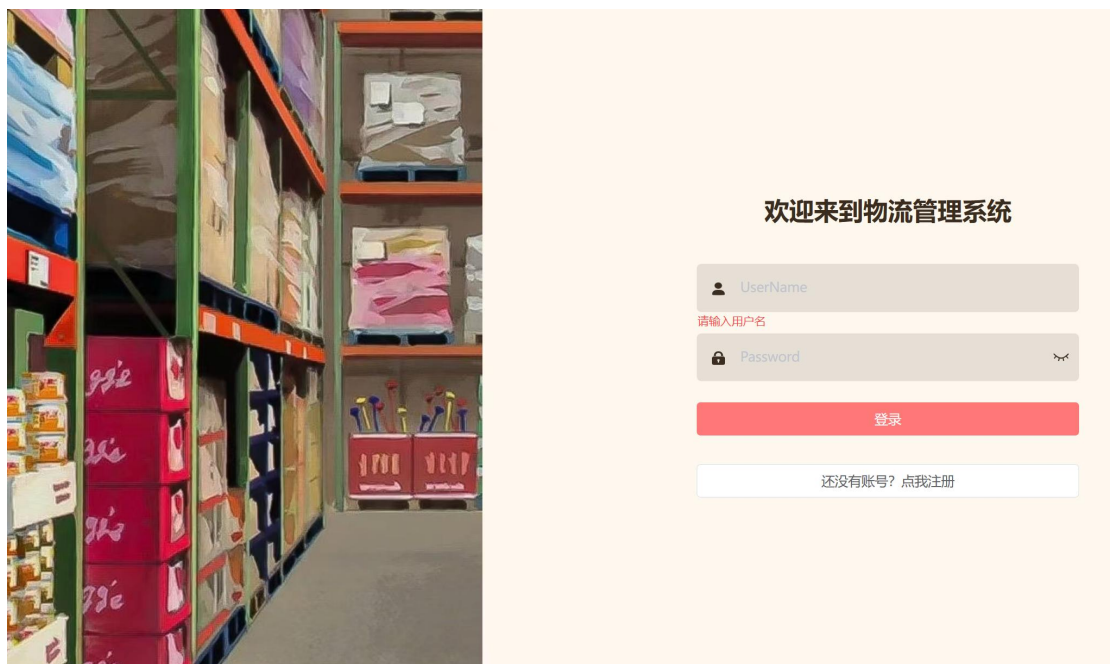
```
{
  code: 200,
  message: "返回成功",
  - data: {
    - orderList: [
      - {
        id: 31,
        createDate: "2023-05-25",
        + shipper: { ... },
        + carriers: [ ... ],
        + vans: [ ... ],
        transfers: [ ],
        destination: "浙江xx仓库",
        state: 2,
        info: null,
        money: 50000,
        start: "北京xx中心",
        weight: 2000,
        receiver: "刘xx"
      },
      - {
        id: 32,
        createDate: "2023-05-25",
        - shipper: {
          id: 10,
          username: "1234",
          realName: "1234",
          phone: "15312341111",
        }
      }
    ]
  }
}
```

### 3 页面设计

主要设计了登录注册页面、首页、用户管理、员工管理、车辆管理、全部订单、承运员订单、用户订单页面。

#### (1) 登录页面：


员工、用户、管理员可从此登录进入系统。



#### (2) 注册页面：

普通用户能在此页面进行注册。

已有账号?



用户名

真实姓名

手机号

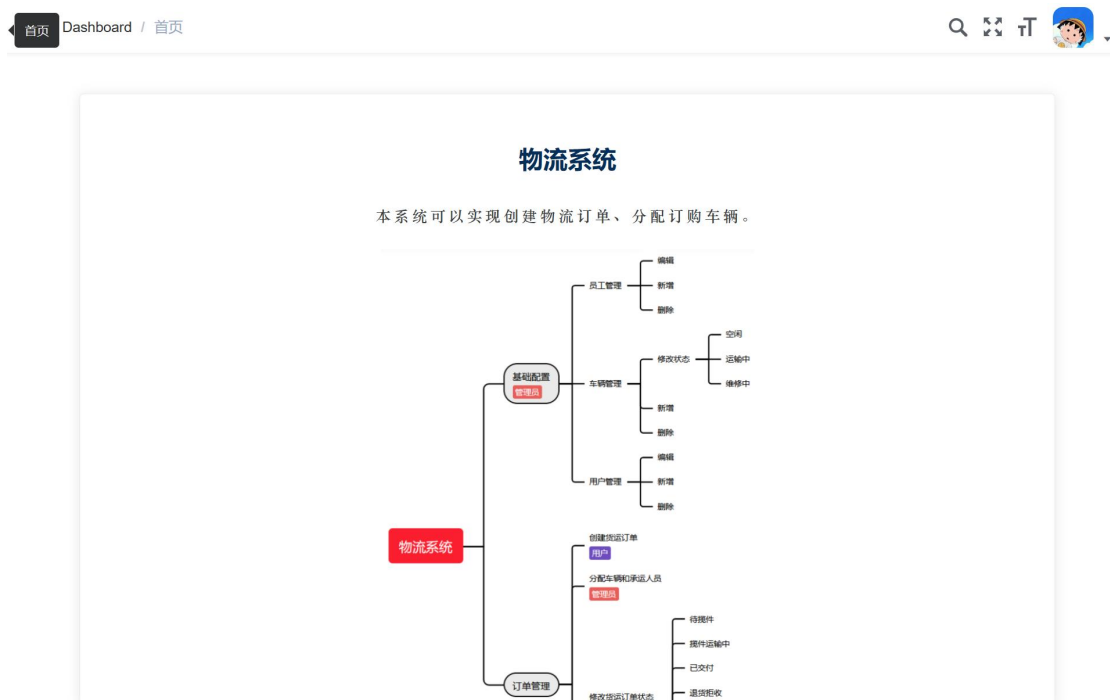
密码

确认密码

注册

(3) 首页:

该系统的简单介绍。



(4) 用户管理:

管理员可在此页面进行用户的增删查改。

Dashboard / 基础配置 / 用户管理

搜索

刷新

全屏

头像

用户名

真实姓名

查找

新增

批量删除

取消选择

<input type="checkbox"/>	账号名称	ID	真实姓名	联系方式	操作
<input type="checkbox"/>	modify	1	改名了	2222222222	编辑
<input type="checkbox"/>	1234	10	1234	15312341111	编辑
<input type="checkbox"/>	aaa	12	啊啊啊	15312341112	编辑
<input type="checkbox"/>	ttt	15	啊啊啊	12341234567	编辑

Total 4

10/page

<1>Go to1

Dashboard / 基础配置 / 用户管理

搜索

刷新

全屏

头像

用户名

真实姓名

查找

新增

批量删除

取消选择

<input type="checkbox"/>	账号名称	ID	真实姓名	联系方式	操作
<input type="checkbox"/>	modify	1	改名了	2222222222	编辑
<input type="checkbox"/>	1234	10	1234	15312341111	编辑
<input type="checkbox"/>	aaa	12	啊啊啊	15312341112	编辑
<input type="checkbox"/>	ttt	15	啊啊啊	12341234567	编辑

Total 4

10/page

<1>Go to1

id1

账号名称modify

真实姓名改名了

\*密码请输入密码

联系方式2222222222

取消确定

(5) 员工管理：

管理员可在此页面进行员工用户的增删查改。

Dashboard / 基础配置 / 员工管理

用户名

真实姓名

查找

新增

批量删除

取消选择

号名称	ID	真实姓名	性别	入职时间	生日	联系方式	操作
ten	3	李永钦	男	2023-05-21	1996-02-27	15112341123	编辑
yang	4	刘扬扬	男	2023-05-21	2000-10-10	15112342123	编辑
kun	5	钱琨	男	2023-05-21	1996-01-01	15112342323	编辑
win	6	董思成	男	2023-05-21	1997-10-28	15112332323	编辑
jun	7	肖俊	男	2023-05-21	1999-08-08	15112333323	编辑
endery	8	黄冠亨	男	2023-05-21	1999-09-28	15142333323	编辑
odify2	49	改名了	女	2022-09-09	1998-09-09	2222222211	编辑

Total 710/page1Go to1

Dashboard / 基础配置 / 员工管理

用户名

真实姓名

查找

新增

批量删除

取消选择

号名称	ID	真实姓名	性别	入职时间	生日	联系方式	操作
ten	3	李永钦	男	2023-05-21	1996-02-27	15112341123	编辑
yang	4	刘扬扬	男	2023-05-21	2000-10-10	15112342123	编辑
kun	5	钱琨	男	2023-05-21	1996-01-01	15112342323	编辑
win	6	董思成	男	2023-05-21	1997-10-28	15112332323	编辑
jun	7	肖俊	男	2023-05-21	1999-08-08	15112333323	编辑
endery	8	黄冠亨	男	2023-05-21	1999-09-28	15142333323	编辑
odify2	49	改名了	女	2022-09-09	1998-09-09	22222211	编辑

Total 710/page1Go to1

id

3

\* 账号名称

ten

\* 真实姓名

李永钦

\* 密码

请输入密码

\* 联系方式

15112341123

\* 性别

男

女

\* 入职日期

2023-05-21

\* 出生日期

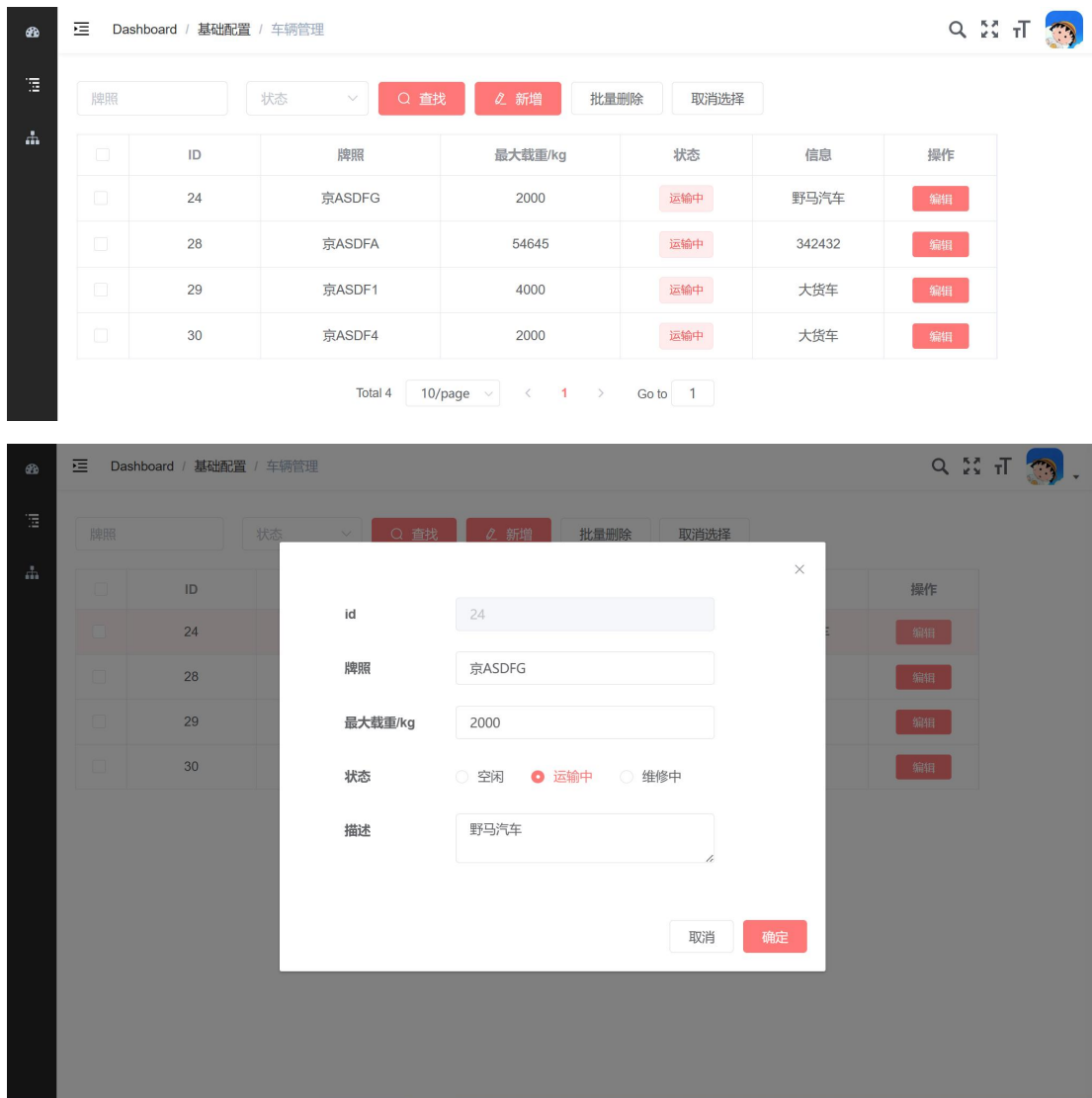
1996-02-27

取消

确定

(6) 车辆管理：

管理员可在此页面进行车辆管理。



### (7) 全部订单:

管理员可在此页面给订单分配车辆和承运员、确定运费，分配车辆的数量不能多于承运人数量，车辆的总载重不能小于货物总重量。管理员还可在次页面查看订单详情，包括订单的运输记录。



Dashboard / 订单管理 / 全部订单

发货人用户名

运输员用户名

车辆牌照

Q 查找

	起点	终点	信息	重量	运费/元	状态		操作
	北京xx中心	浙江xx仓库		2000	50000	揽件运输中	查看详情	分配车辆和运货员
	北京xx中心	浙江xx仓库		2000	434535	待揽件	查看详情	分配车辆和运货员
	北京xx中心	浙江xx仓库		2000	44444	待揽件	查看详情	分配车辆和运货员
	北京xx中心	浙江xx仓库		2000	40000	订单结束	查看详情	分配车辆和运货员
	北京xx中心	浙江xx仓库	建筑钢材	2000	40000	订单结束	查看详情	分配车辆和运货员
	北京xx中心	浙江xx仓库	建筑钢材	2000	40000	订单结束	查看详情	分配车辆和运货员
	北京xx中心	浙江xx仓库	建筑钢材	2000	40000	订单结束	查看详情	分配车辆和运货员
	北京xx中心	浙江xx仓库	建筑钢材	4000	0	订单结束	查看详情	分配车辆和运货员
	北京	新疆	装修钢材	0	0	订单结束	查看详情	分配车辆和运货员

Dashboard / 订单管理 / 全部订单

发货人用户名

运输员用户名

车辆牌照

Q 查找

详情

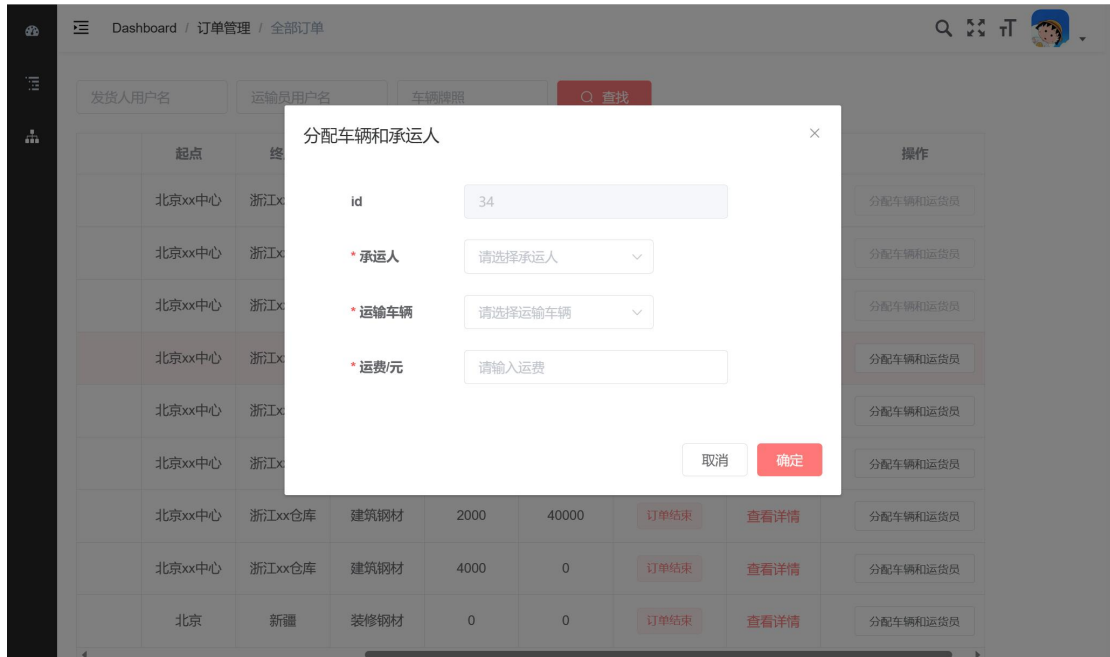
ID	创建日期	发件人	发件人电话号码
31	2023-05-25	modify	22222222222

用户名	真实姓名	联系电话
ten	李永钦	15112341123

牌照	最大载重
京ASDF4	2000

时间	地点
2023-05-27T19:48:46	山东

确定



#### (8) 承运员订单:

员工可在此页面对自己负责的订单的地点进行登记, 更新订单的状态, 未分配车辆和承运人的订单无法进行地点登记和状态更新, 处于“待揽件”状态的订单无法进行地点登记。员工还可在此页面查看自己负责订单的总运费金额。



Dashboard / 订单管理 / 承运员订单

总金额: 528979元

起点

北京xx中心

北京xx中心

北京xx中心

详情

ID	创建日期	发件人	发件人电话号码
31	2023-05-25	modify	2222222222

用户名	真实姓名	联系电话
ten	李永钦	15112341123
ten	李永钦	15112341123

牌照	最大载重
京ASDFA	54645
京ASDF4	2000

时间	地点
2023-05-27T19:48:46	山东

Dashboard / 订单管理 / 承运员订单

总金额: 528979元

起点

北京xx中心

北京xx中心

北京xx中心

终点

浙江xx仓库

浙江xx仓库

浙江xx仓库

操作

登记

更新状态

登记

更新状态

登记

更新状态

运输地点登记

id

31

\* 地点

取消

确定

Dashboard / 订单管理 / 承运员订单

总金额: 528979元

起点

北京xx中心

北京xx中心

北京xx中心

终点

浙江xx仓库

浙江xx仓库

浙江xx仓库

操作

登记

更新状态

登记

更新状态

登记

更新状态

订单状态修改

id

31

\* 订单状态

☐ 待揽件

☒ 揽件运输中

☐ 已交付

☐ 退货拒收

☐ 退货待揽件

☐ 退货运输中

☐ 退货交付

☐ 订单结束

取消

确定

## (9) 用户订单:

用户可在此页面创建订单，查看自己订单的物流记录。

Dashboard / 订单管理 / 用户订单

新增

发件人	收件人	起点	终点	信息	重量	运费/元	状态	
modify	刘xx	北京xx中心	浙江xx仓库		2000	50000	揽件运输中	查看详情

Total 110/page1Go to 1

Dashboard / 订单管理 / 用户订单

新增

创建订单

\* 发货地

\* 目的地

\* 收件人

\* 重量/kg

\* 信息

取消确定

Dashboard / 订单管理 / 用户订单

新增

详情

ID	创建日期	发件人	发件人电话号码
31	2023-05-25	modify	2222222222

用户名	真实姓名	联系电话
ten	李永钦	15112341123

牌照	最大载重
京ASDF4	2000

时间	地点
2023-05-27T19:48:46	山东

---

## 4 单元测试

主要测试了订单模块 Controller 层的创建订单、更新订单状态、添加转运信息的功能。单元测试采用 JUnit+ Mockito 实现，生成代码测试覆盖分析报告。

### 4.1 创建订单

测试用例编号	Order-01	用例类型	Controller 层
用例目的	测试创建订单的功能		
测试步骤			
<p>创建一个模拟的用户对象 shipper，并使用 Mockito 配置 userRepository.findById() 方法，以便在调用时返回这个用户对象。</p> <p>创建一个 OrderRequest 对象，该对象包含创建订单所需的信息。</p> <p>配置 orderRepository.save() 方法，以便在调用时返回一个新的 Order 对象，并将该对象的 ID 设置为 1L。</p> <p>使用 MockMvc 发送 POST 请求，将 OrderRequest 对象作为 JSON 内容发送到 /order/create 路径。</p> <p>使用 MockMvcResultMatchers 验证响应中的状态码和其他属性，例如响应体中的 code 字段是否为 200。</p> <p>验证订单是否已创建并保存到数据库中。为此，创建一个 Order 对象，然后将其 ID 设置为 1L，并使用 Mockito 配置 orderRepository.findById() 方法，以便在调用时返回该对象。最后，使用 orderRepository.findAll() 方法获取数据库中的所有订单，并验证其长度是否为 1。</p>			
代码覆盖分析			

```

@PostMapping("/create")
public ResponseEntity<?> createOrder(@Valid @RequestBody OrderRequest orderRequest) {
    Map<String, Object> response = new HashMap<>();
    User shipper = userRepository.findById(orderRequest.getShipperId()).get();
    if (shipper == null) {
        return ResponseEntity
            .badRequest()
            .body(new Result(StatusCode.INFOERR, "Shipper not found!", response));
    }
    int state=0;
    LocalDateTime createDate= LocalDateTime.now();
    System.out.println(orderRequest.getStart());
    Order order = new Order(shipper,
        orderRequest.getStart(),
        orderRequest.getDestination(),
        orderRequest.getReceiver(),
        orderRequest.getInfo(),
        orderRequest.getWeight(),
        state,
        createDate);

    System.out.println(order);
    orderRepository.save(order);

    return ResponseEntity.ok(new Result(StatusCode.SUCCESS, "Success", response));
}

```

## 4.2 更新订单状态

测试用例编号	Order-02	用例类型	Controller 层
用例目的	测试更新订单状态的功能		
测试步骤			
<p>创建一个模拟的用户对象 shipper，并使用 Mockito 配置 userRepository.findById() 方法，以便在调用时返回这个用户对象。</p> <p>创建一个订单对象，并使用 Mockito 配置 orderRepository.findById() 方法，以便在调用时返回该订单对象。</p> <p>创建一个 OrderRequest 对象，用于指定要更改的订单的 ID 和状态。</p> <p>使用 MockMvc 发送 POST 请求，将 OrderRequest 对象作为 JSON 内容发送到 /order/changeState 路径。</p> <p>使用 MockMvcResultMatchers 验证响应中的状态码和其他属性，例如响应体中的 code 字段是否为 200。</p> <p>验证订单状态是否已更改但未保存到数据库中。为此，创建一个 Order 对象，该对象包含新状态，然后使用 Mockito 配置 orderRepository.findById() 方法，以便在调用时返回该对象。</p>			
代码覆盖分析			

```

@PostMapping("/changeState")
public ResponseEntity<?> changeState(@Valid @RequestBody OrderRequest orderRequest) {
    Order order = orderRepository.findById(orderRequest.getId()).orElse(null);
    Map<String, Object> response = new HashMap<>();
    if (order == null) {
        return ResponseEntity.badRequest().body(new Result(ResultCode.INFOERR, "Order not found!", response));
    }
    //1- 已付款待揽件, 2- 揽件运输中, 3- 已交付, 4- 退货拒收 40- 退货待揽件 41- 退货运输中 42- 退货交付 5- 订单结束
    int state = orderRequest.getState();
    order.setState(state);
    orderRepository.save(order);
    return ResponseEntity.ok(new Result(ResultCode.SUCCESS, "Update successfully!", response));
}

```

## 4.3 添加转运信息

测试用例编号	Order-03	用例类型	Controller 层
用例目的	测试添加转运信息的功能		
测试步骤			
<p>创建一个模拟的用户对象 shipper，并使用 Mockito 配置 userRepository.findById() 方法，以便在调用时返回这个用户对象。</p> <p>创建一个模拟的用户对象 carrier，并使用 Mockito 配置 userRepository.findById() 方法，以便在调用时返回这个用户对象。</p> <p>创建一个模拟的 Van 对象，并使用 Mockito 配置 vanRepository.findById() 方法，以便在调用时返回这个 Van 对象。</p> <p>创建一个订单对象，并使用 Mockito 配置 orderRepository.findById() 方法，以便在调用时返回该订单对象。</p> <p>创建一个 OrderRequest 对象，用于指定要转移的订单的 ID、运输公司 ID 和货车 ID。</p> <p>使用 Mockito 配置 transferRepository.save() 方法，以便在调用时返回一个新的 Transfer 对象。</p> <p>使用 MockMvc 发送 POST 请求，将 OrderRequest 对象作为 JSON 内容发送到 /order/transfer 路径。</p> <p>使用 MockMvcResultMatchers 验证响应中的属性，例如响应体中的 code 字段是否为 102。</p>			
代码覆盖分析			

---

```
@PostMapping("/transfer")
public ResponseEntity<?> transfer(@Valid @RequestBody OrderRequest orderRequest) {
    Order order = orderRepository.findById(orderRequest.getId()).orElse(null);
    Map<String, Object> response = new HashMap<>();
    if (order == null) {
        return ResponseEntity.badRequest().body(new Result(ResultCode.INFOERR, "Order not found!", response));
    }
    if (order.getState() == 0) {
        return ResponseEntity.badRequest().body(new Result(ResultCode.INFOERR, "State of order wrong!", response));
    }

    String loca = orderRequest.getLoca();
    LocalDateTime time = LocalDateTime.now();

    Transfer transfer = new Transfer(order, loca, time);
    transferRepository.save(transfer);

    return ResponseEntity.ok(new Result(ResultCode.SUCCESS, "Update successfully!", response));
}
```