# 后端设计

数据库设计：

SHIPPER船员表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| shipper\_id(主键) | shipper\_name | email | passward | shipper\_address |
| 1001 | John | john@example.com | 123 | 北京海淀 |

ORDER订单表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| shipment\_id(主键) | consignee\_name | consignee\_address | description | weigth |
| 100001 | 王先生 | 北京海淀区 | 十箱木材 | 100kg |
| 100002 | 刘先生 | 上海浦东 | 十箱玻璃 | 50KG |

Billings费用表

|  |  |  |  |
| --- | --- | --- | --- |
| shipment\_id（主键） | amout | duedate | status |
| 100001 | 100$ | 20230524 | pending |
| 100002 | 50$ | 20230527 | paid |

Track货物跟踪表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| shipper\_id | shipment\_id | ship\_date | delivery\_date | status |
| 1001 | 100001 | 20230510 | 20230517 | unshipped |

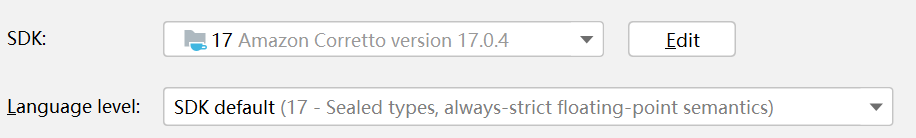
TrackInfo表

|  |  |  |  |
| --- | --- | --- | --- |
| shipment\_id（主键） | track\_datetime (主键) | location | status |
|  |  |  |  |

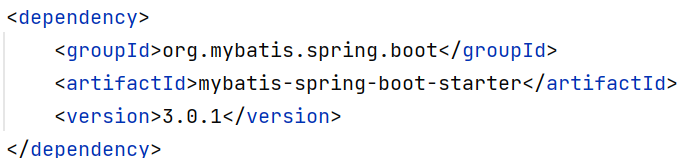
**后端技术方案：**

采用Java Springboot Mybatis3 Mysql8.0的方案，以下是各版本号

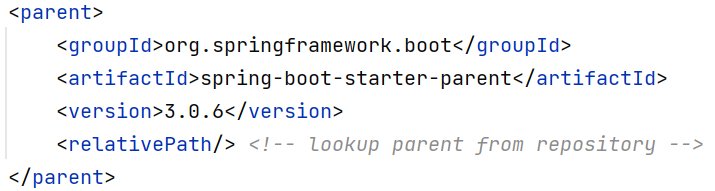
java



Maven

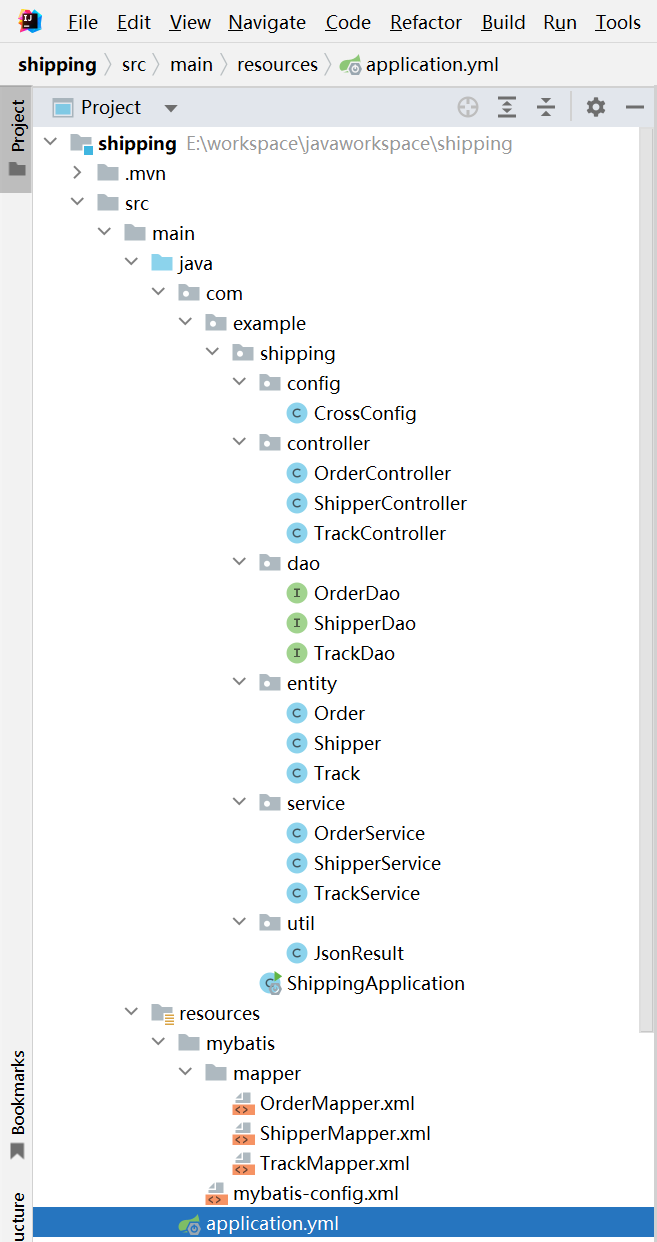


Springboot

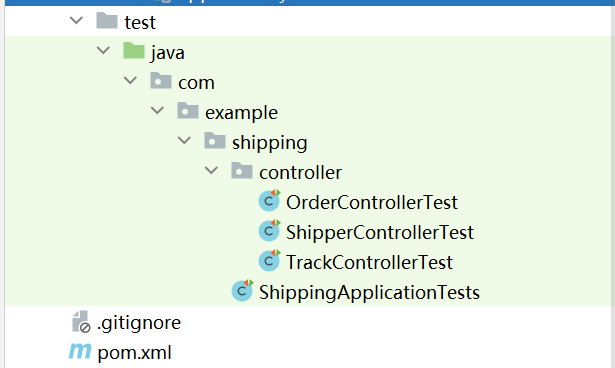


**后端项目预览：**

以下是后端项目代码结构预览：



在Test文件夹中完成Server层代码的单元测试：



**后端介绍：**

后端采用典型的MVC架构，分别设计Controller Service Dao Entity Mapper等架构，在Controller层中设计对外接口，在Service层中实现基本业务，在Entity中设计实体类，在Dao层中实现数据持久化接口设计，在Mapper文件 中编写SQL语句。

# 前端设计：

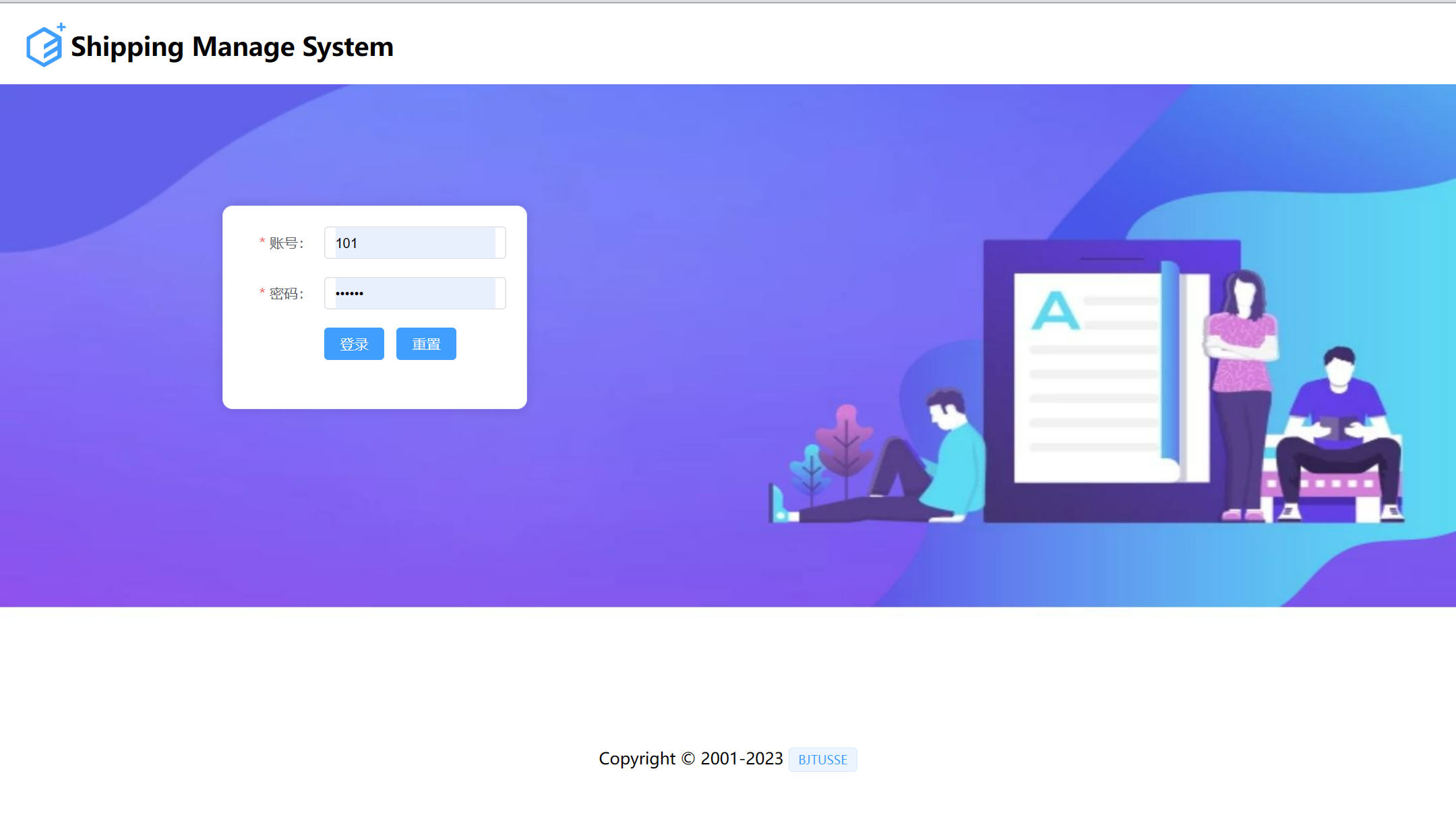
前端使用Vue3 + Vite3 + node.js18搭建项目

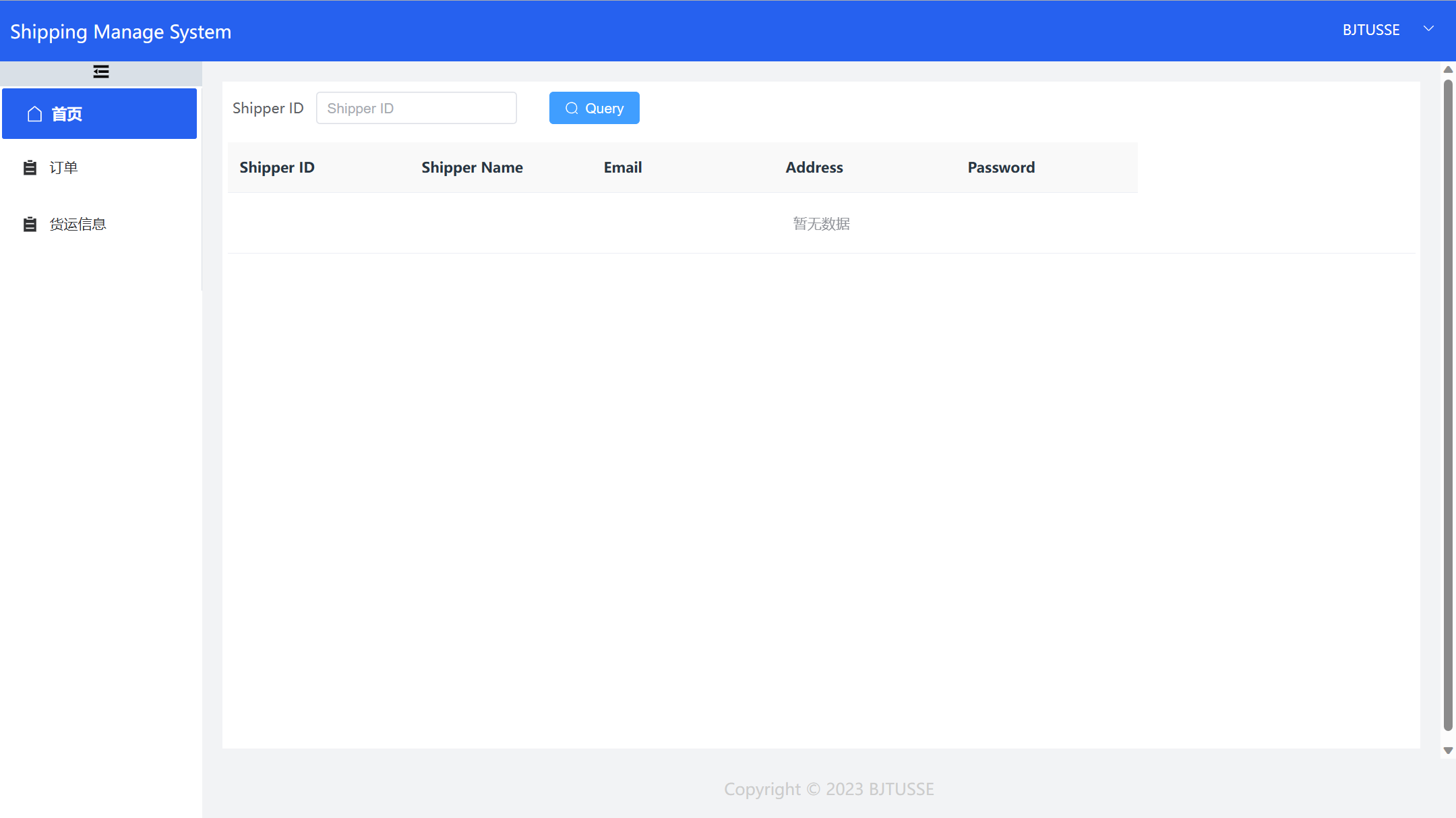
使用：

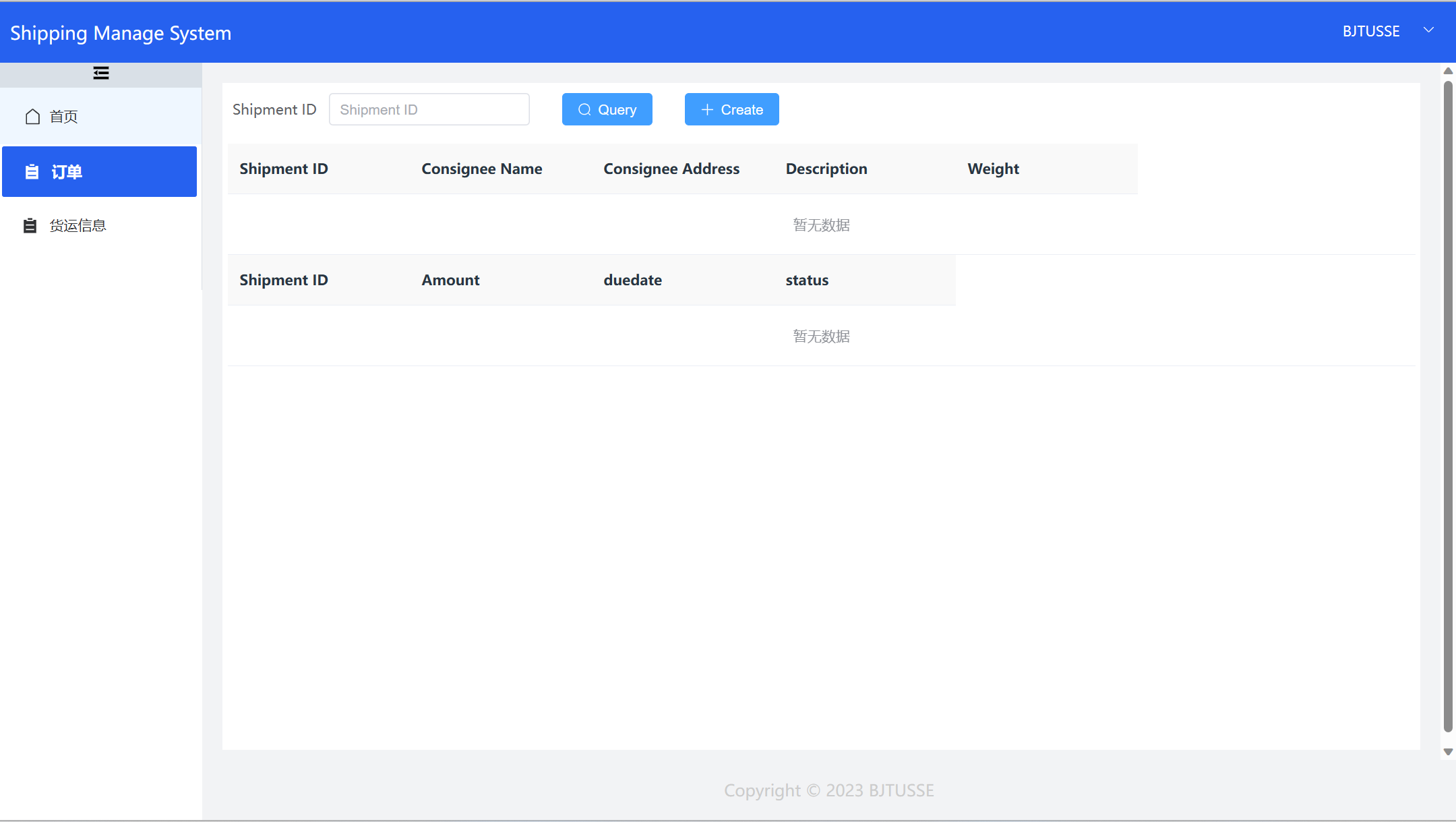
-npm install

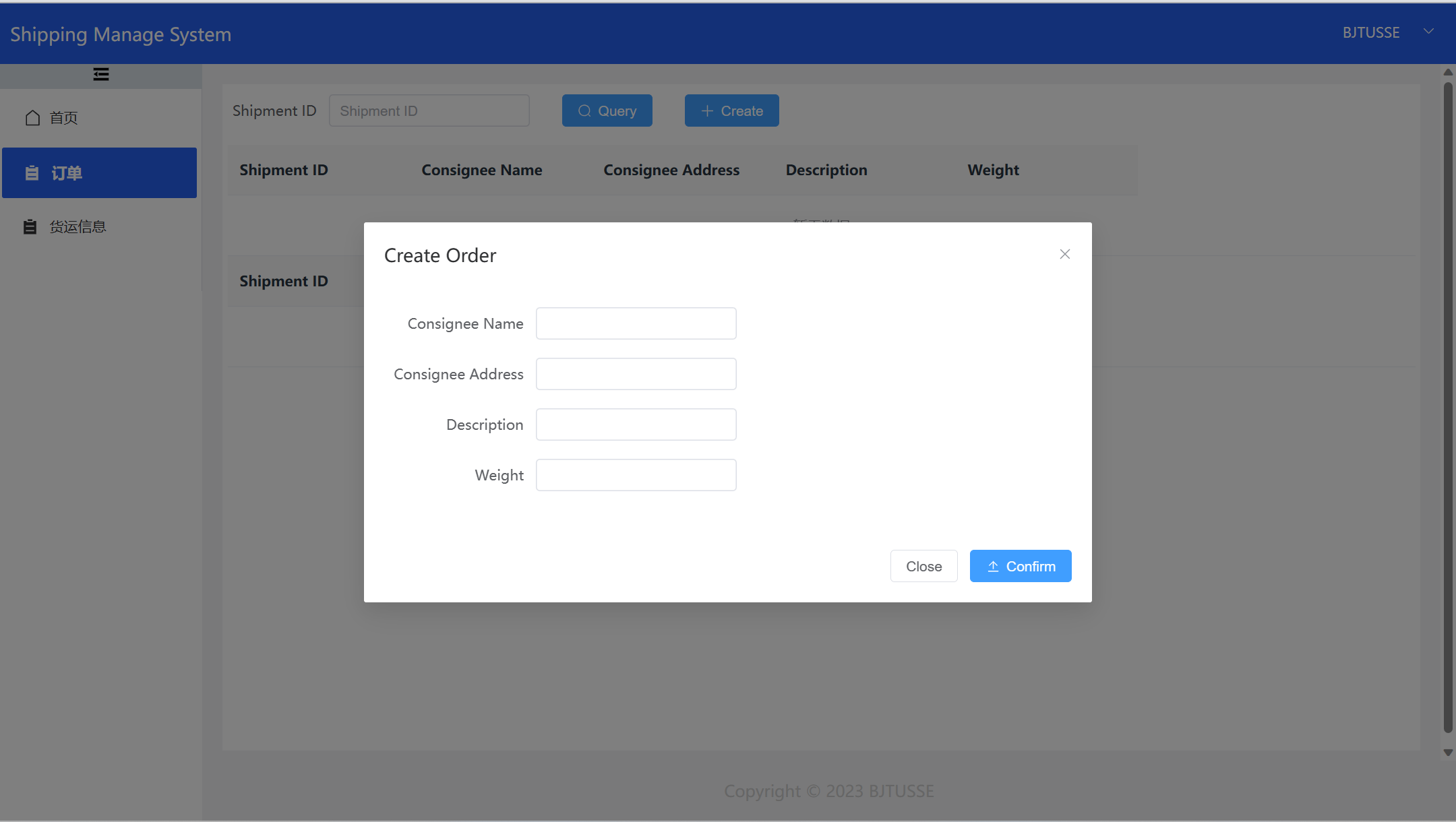
-npm run dev

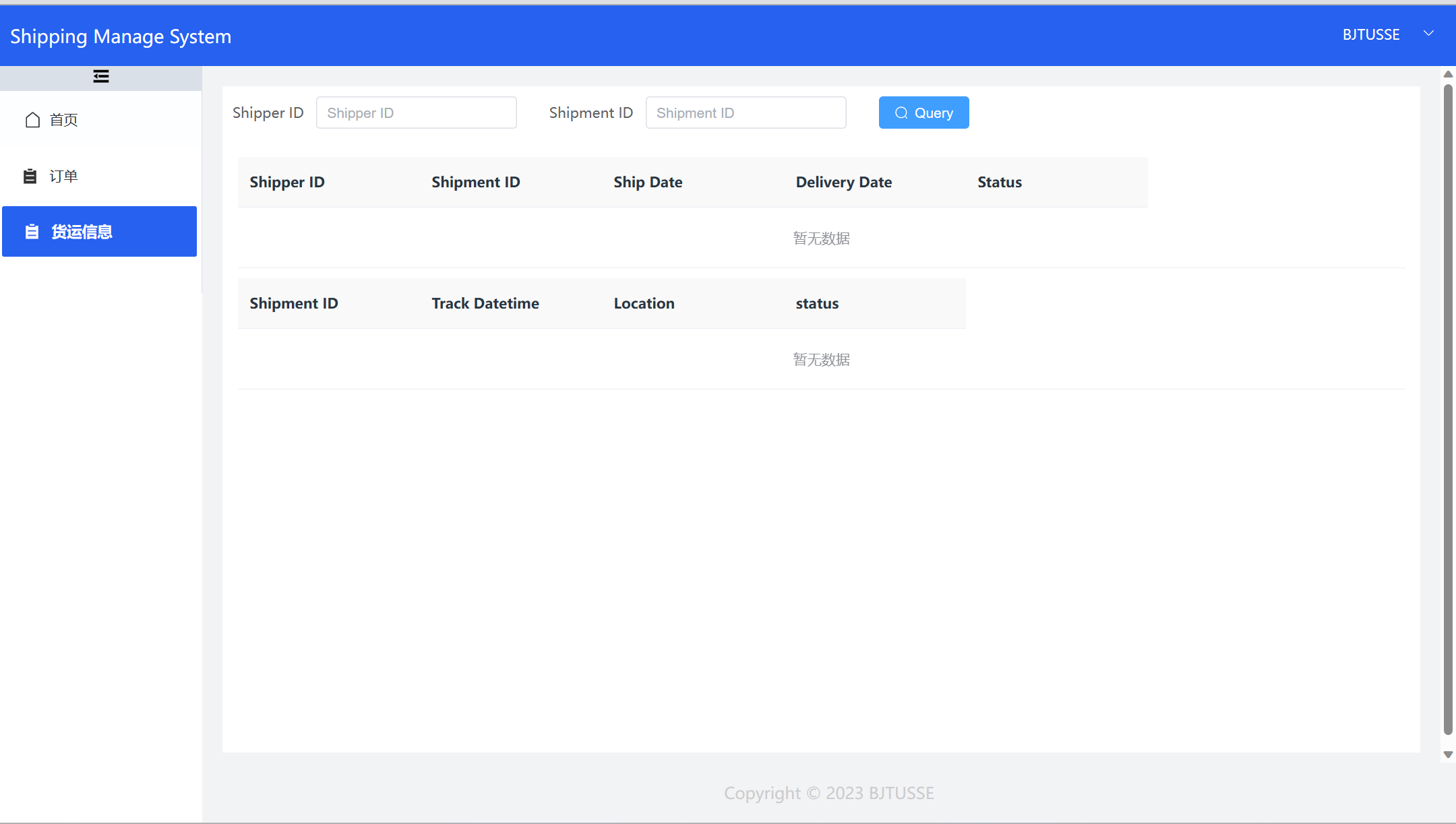
**前端界面预览：**











**前端界面介绍：**

前端界面共设计四个主界面分别是：登录、首页、订单、货运信息。

登录界面：

提供管理员登录功能

首页：

提供根据船员ID - Shipper ID 查询船员信息。

订单：

根据订单ID - Shipment ID 查询订单信息 。

创建订单 在表单中填写基本信息即可创建订单。

货运信息：

在货运信息界面填写Shipper ID和Shipment ID后可查询船员所负责的订单信息以及该订单对应的货运信息。

Shipment ID是必填字段，若只填写Shipment ID则只会显示该订单的货运信息。

# 测试：

**单元测试：**

在Springboot项目中的Test文件夹使用Junit测试框架结合@SpringbootTest和@Test直接实现Service层全部业务的单元测试。

**接口（集成测试）：**

使用ApiPost7对Controller层所暴露的外部接口进行API测试。

测试预览如下：

