

后端设计

数据库设计：

SHIPPER 船员表

shipper_id(主键)	shipper_name	email	password	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	weight
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)	amount	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

SDK:

 17 Amazon Corretto version 17.0.4

Edit

Language level:

SDK default (17 - Sealed types, always-strict floating-point semantics)

Maven

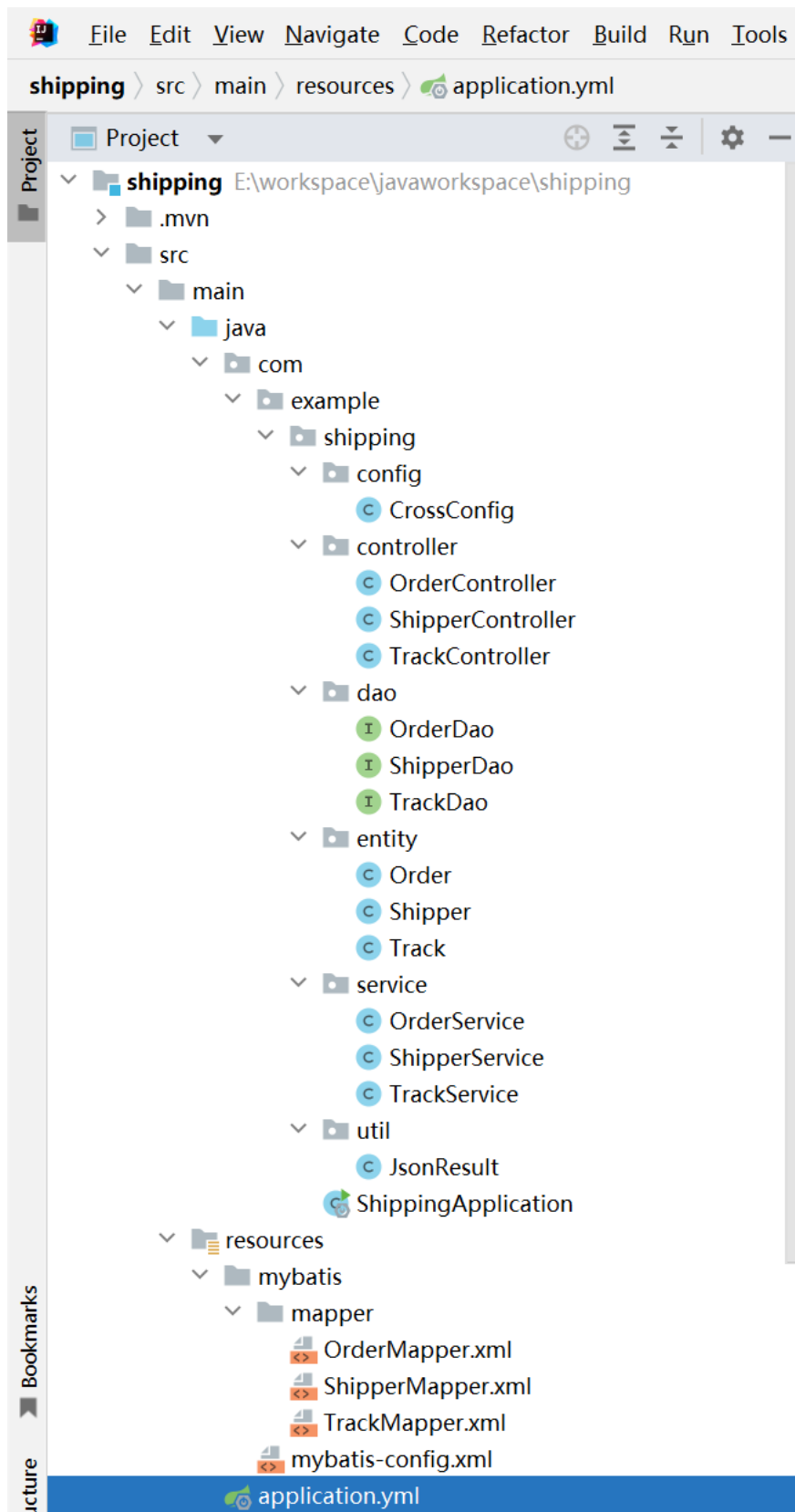
```
<dependency>
  <groupId>org.mybatis.spring.boot</groupId>
  <artifactId>mybatis-spring-boot-starter</artifactId>
  <version>3.0.1</version>
</dependency>
```

Springboot

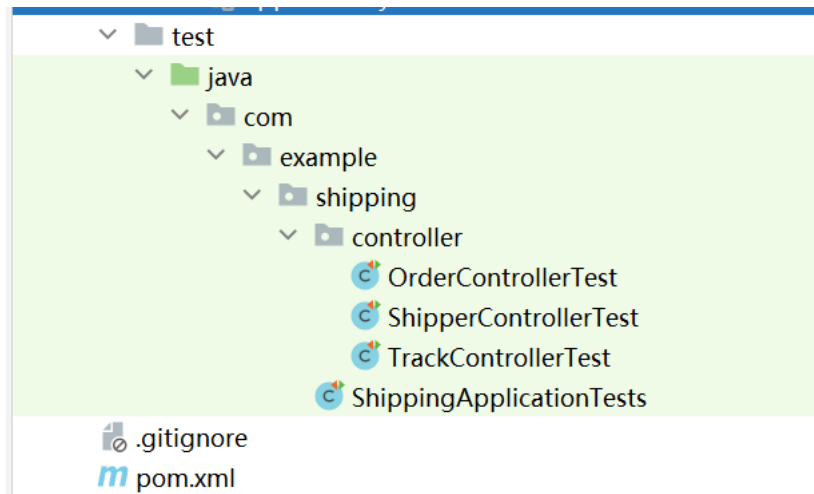
```
<parent>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-parent</artifactId>
  <version>3.0.6</version>
  <relativePath/> <!-- lookup parent from repository -->
</parent>
```

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：

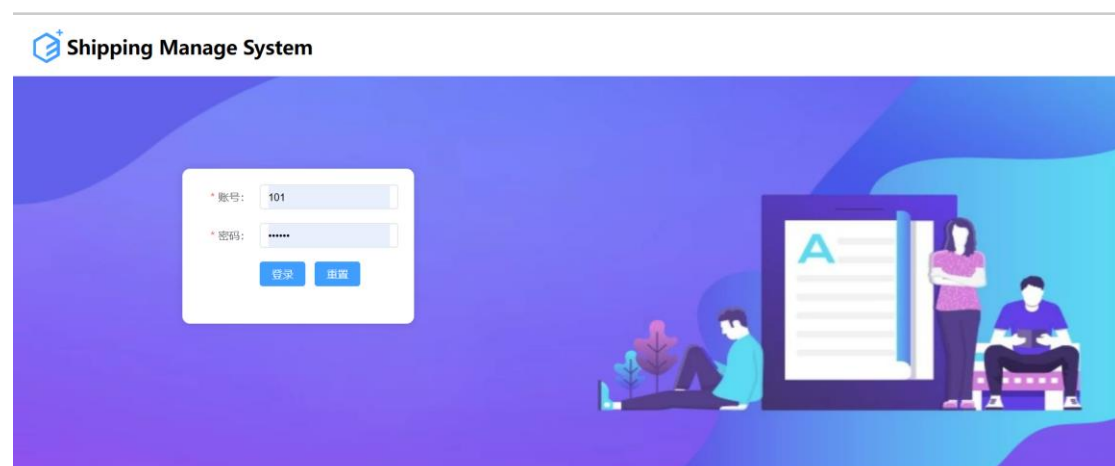
后端采用典型的 MVC 架构，分别设计 Controller Service Dao Entity Mapper 等架构，在 Controller 层中设计对外接口，在 Service 层中实现基本业务，在 Entity 中设计实体类，在 Dao 层中实现数据持久化接口设计，在 Mapper 文件 中编写 SQL 语句。

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：



Shipping Manage System

BITUSSE

首页

订单

货运信息

Shipper ID

Shipper ID

Query

Shipper ID	Shipper Name	Email	Address	Password
暂无数据				

Copyright © 2023 BITUSSE

Shipping Manage System

BITUSSE

首页

订单

货运信息

Shipment ID

Shipment ID

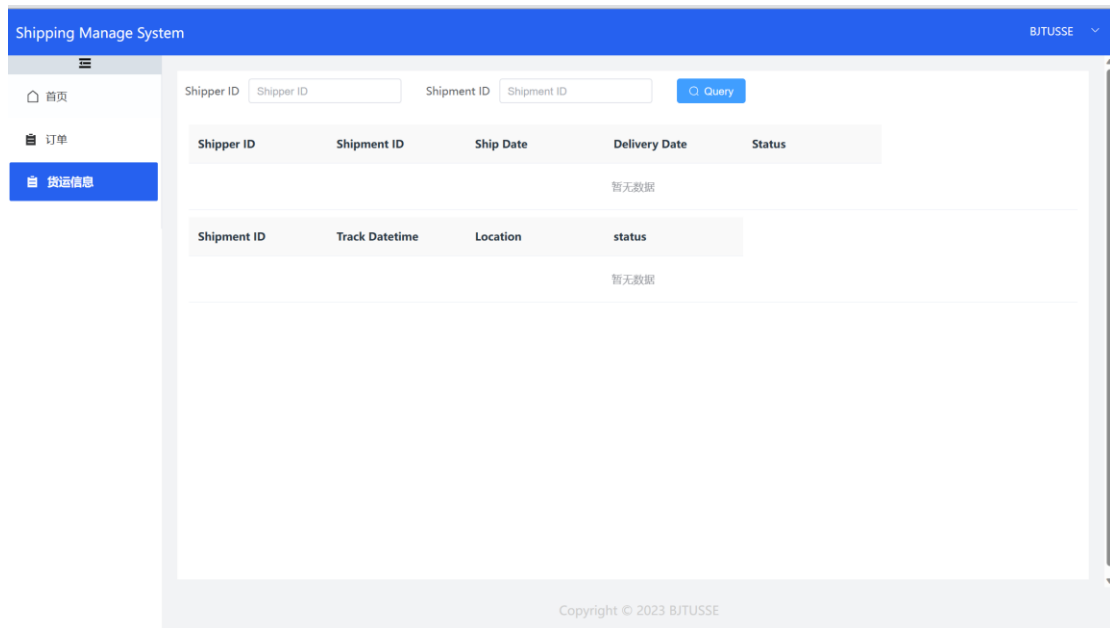
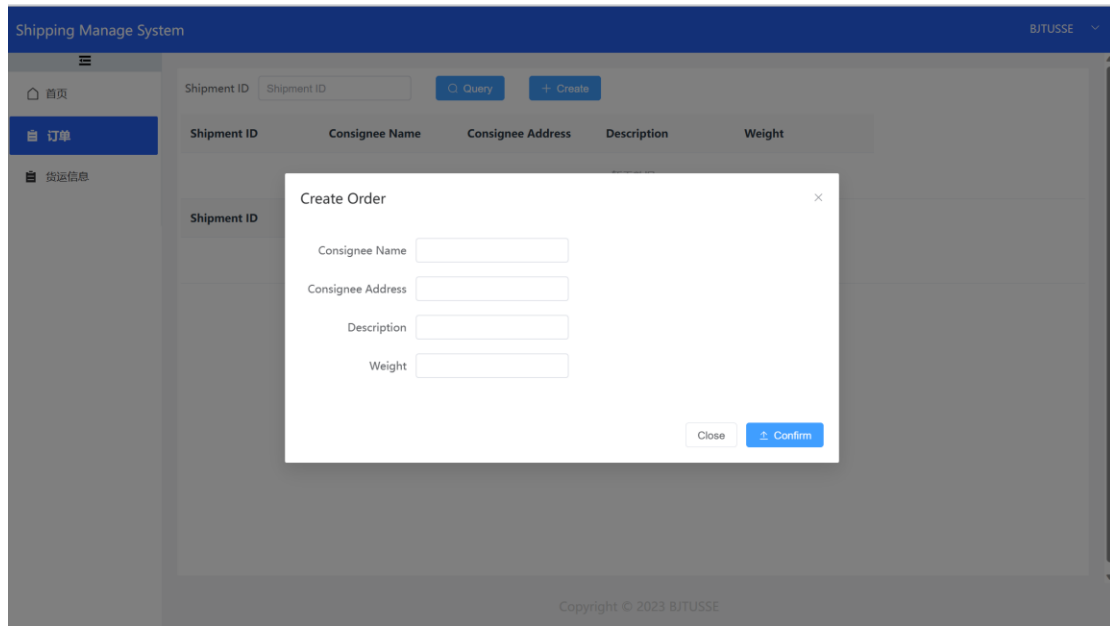
Query

Create

Shipment ID	Consignee Name	Consignee Address	Description	Weight
暂无数据				

Shipment ID	Amount	duedate	status
暂无数据			

Copyright © 2023 BITUSSE



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。

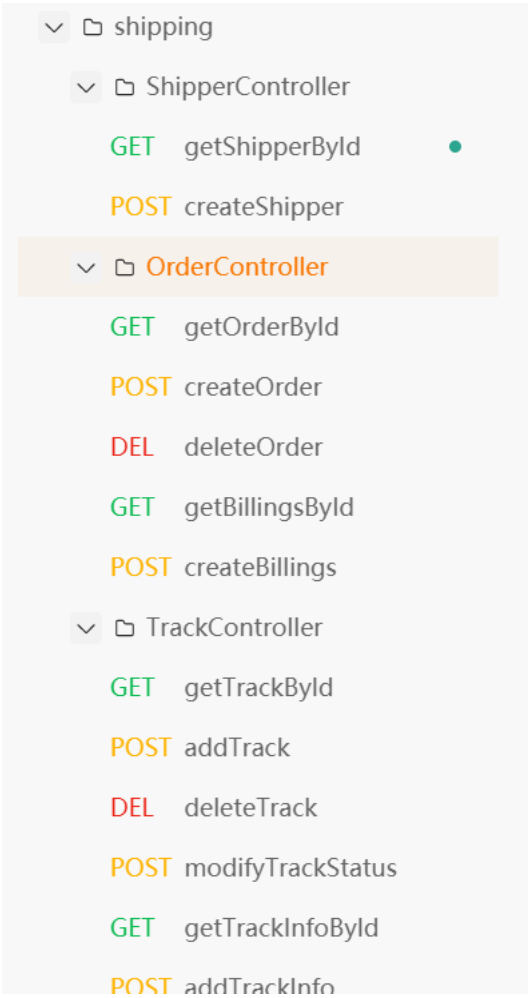
测试：

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：



● 已完成

getShipperByld

分享文档

默认环境

GET

http://localhost:8181/api/getShipperByld

发送

保存

HeaderQueryBody认证预执行脚本后执行脚本一键压测NEW

none

form-data

x-www-form-urlencoded

raw

json

可视化结构原生模式

园 点击更新

提取字段和描述美化三简化

1 {

2 "shipperId": "1001"

3 }

« 字段描述