



专业综合扩展报告

(仓库管理系统详细设计与实现)

学 院 计算机学院

专 业 信息安全

年级班别 2018级（2）班

学 号 3118005434

学生姓名 钟景文

指导教师 黄益民

**2021年10月**

目录

[1 绪论 1](#_Toc30024)

[1.1 系统背景 1](#_Toc11295)

[1.2 系统任务 1](#_Toc20087)

[1.3 系统环境 1](#_Toc24137)

[2 系统详细设计 2](#_Toc10226)

[2.1 API接口 2](#_Toc1679)

[2.1.1 HTTP 请求 2](#_Toc795)

[2.1.2 HTTP响应 2](#_Toc8890)

[2.1.3 HTTP 状态码 2](#_Toc10442)

[2.1.4 URI 细节 4](#_Toc14789)

[2.2 静态模型 17](#_Toc26493)

[2.3 数据库设计 18](#_Toc7627)

[2.3.1 Warehourse表 18](#_Toc22583)

[2.3.2 Cabinet表 19](#_Toc32222)

[2.3.3 Goods表 19](#_Toc1874)

[2.3.4 GoodsType表 20](#_Toc20520)

[2.3.5 Detail表 20](#_Toc2245)

[2.3.6 InOrder表 21](#_Toc7557)

[2.3.7 InItem表 21](#_Toc31659)

[2.3.8 OutOrder表 22](#_Toc17268)

[2.3.9 OutItem表 22](#_Toc24347)

[2.3.10 GetOrder表 23](#_Toc9750)

[2.3.11 GetItem表 23](#_Toc27267)

[3 系统实现 25](#_Toc23437)

[3.1 Spring SpringMVC 配置 25](#_Toc28205)

[3.1.1 Servlet容器 25](#_Toc25347)

[3.1.2 Servlet 初始化器 26](#_Toc9364)

[3.1.3 Spring 配置 27](#_Toc19730)

[3.1.4 Spring MVC 配置 27](#_Toc25522)

[3.2 登录实现 28](#_Toc10358)

[3.2.1 登录界面 28](#_Toc21872)

[3.2.2 登记接口 28](#_Toc23843)

[3.3 数据库实现 29](#_Toc22745)

[3.4 DAO层实现 36](#_Toc12600)

[3.4.1 MyBatis配置文件 36](#_Toc31180)

[3.4.2 MyBatis映射文件 37](#_Toc20072)

[3.4.3 MyBatis映射器接口 38](#_Toc6967)

[3.5 POJO 层实现 39](#_Toc5073)

[3.6 Service层实现 40](#_Toc4127)

[3.6.1 Service层接口 40](#_Toc8021)

[3.6.2 Service 层接口实现 41](#_Toc516)

[3.7 Controller 层实现 42](#_Toc4849)

[总结与体会 43](#_Toc14188)

绪论

系统背景

随着社会的发展，物质资料的极大丰富，物资的储蓄与流动管理越来越来重要，任务越来越频繁。依靠人工管理物资不仅效率低，劳动强度大，且容易出错，因此越来越迫切地需要一个简便搞笑的仓库管理系统。

系统任务

库存管理：增删查改仓库，增删查改货物。

入库管理：输入货物编号，登记入库，预分配储存位置，生成入库表。

出库管理：收集取货单信息，分配出货仓库，生成出库信息。定时生成入库表。

取货管理：增删查改取货单，增删查改取货单项。

系统环境

开发平台：JSE8

后端技术：Spring,SpringMVC,MyBatis

前端技术：JSP

数据库：MySQL

其他：.....

系统详细设计

* 1. API接口
     1. HTTP 请求

请求头部必须包含以下字段和值，其他字段后续详细定义。

表 1 HTTP 请求头公共字段

|  |  |  |
| --- | --- | --- |
| 头部字段 | 值 | 解释 |
| Content-Type | application/json | 表示请求体内容是Json |

* + 1. HTTP响应

响应头部必须包含以下字段和值，其他字段后续详细定义。

表 2 HTTP 响应头公共字段

|  |  |  |
| --- | --- | --- |
| 头部字段 | 值 | 解释 |
| Content-Type | application/json | 表示响应体内容是Json |

响应体是JSON数据，请求失败时，返回失败信息，如表所示：

表 3 HTTP 响应体

|  |  |  |
| --- | --- | --- |
| 数据类型 | 内容 | 说明 |
| JSON | 响应结果 | 请求成功 |
| JSON | {“error”:“~”,“descript”:“~”} | 请求失败 |

* + 1. HTTP 状态码

2xx状态码：表示操作成功，不同的请求方法成功返回不同的状态，如表所示：

表 4 HTTP 2xx 状态码

|  |  |  |
| --- | --- | --- |
| HTTP 方法 | 状态码 | 短语 |
| GET | 200 | OK |
| POST | 201 | Created |
| PATCH | 200 | OK |
| DELETE | 204 | NO Content |

3xx状态码：表示资源的URL被转移，主要用于重定向，详细如表所示：

表 5 HTTP 3xx 状态码

|  |  |  |
| --- | --- | --- |
| 状态码 | 短语 | 描述 |
| 301 | Moved Permanently | 资源被永久移动到新URI |
| 302 | Found | 资源被临时移动到新URI |
| 303 | See Other | 由用户决定是否重定向 |

为了告诉客户端资源转移的URL，需要增加如下响应字段：

表 6 HTTP 3xx 响应头

|  |  |  |
| --- | --- | --- |
| 头部字段 | 值 | 解释 |
| Location | 资源的URI | 返回的URI |

4xx状态码：表示客户端请求有错误：

表 7 HTTP 4xx 状态码

|  |  |  |
| --- | --- | --- |
| 状态码 | 短语 | 解释 |
| 400 | Bad Request | 服务器不理解客户端的请求，未做任何处理。 |
| 401 | Unauthorized | 用户未提供身份验证凭据或没有通过身份验证。 |
| 403 | Forbidden | 用户通过身份验证，但是不具访问资源所需的权限。 |
| 404 | Not Found | 所请求的资源不存在，或不可用。 |
| 405 | Method Not Allowed | 用户通过身份验证，但无权限用这个HTTP 方法。 |
| 410 | Gone | 所请求的资源已从这个地址转移，不再可用。 |
| 415 | Unsupported Media Type | 客户端要求的返回格式不支持。 |
| 422 | Unprocessable Entity | 客户端上传的附件无法处理，导致请求失败 |
| 429 | Too Many Requests | 客户端的请求次数超过限额。 |

5xx状态码：表示服务端错误，一般来说，不会向用户透露服务器的详细信息。

表 8 HTTP 5xx 状态码

|  |  |  |
| --- | --- | --- |
| 状态码 | 短语 | 解释 |
| 500 | Internal Server Error | 客户端请求有效，服务器处理时发生了意外。 |
| 503 | Service Unavailable | 服务器无法处理请求，一般用于网站维护状态。 |

* + 1. URI 细节
* 仓库管理
  + 请求URI：GET /warehouses
    - 说明：列出仓库
    - 请求参数：
      * offset：类型int, 可选，指定返回记录的开始位置。
      * limit：类型int, 可选，返回记录的限制数量。
      * status：类型String，可选，筛选仓库的状态。
      * address：类型String, 可选，筛选仓库地址。
    - 响应数据：List<warehourse>
  + 请求 URI：GET /warehourses/<id>
    - 说明：获取指定仓库信息。
    - 请求参数：无
    - 响应数据：Warehourse
  + 请求 URI：POST /warehourses
    - 说明：创建一个新的数据库。
    - 请求参数：置于请求体JSON中。
      * name：类型String，必选。
      * address：类型String，必选。
      * descript：类型String，可选。
      * length：类型int，可选。
      * width：类型int，可选
      * height：类型int，可选。
    - 响应数据：无。
    - 响应头部：Location字段：新数据库的URI。
  + 请求 URI：PATCH /warehourses/<id>
    - 说明：修改一个仓库的信息。
    - 请求参数：置于请求体JSON中。
      * status：类型string，可选。
      * name：类型String, 可选。
      * address：类型String，可选。
      * description：类型String，可选。
      * length：类型int，可选。
      * width：类型int，可选。
      * height：类型int，可选。
    - 响应数据：无。
  + 请求URL：DELETE /warehourse/<id>
    - 说明：删除仓库。
    - 请求参数：无
    - 响应数据：无
* 储柜管理：
  + 请求URI：GET /warehourse/<id>/cabinets
    - 说明：列出储柜信息
    - 请求参数：
      * offset：类型int，可选。
      * limit：类型int，可选。
      * status：类型int，可选。
      * time：类型LocalDate，可选，对应status。
      * before：类型LocalDate，可选，对应status。
      * after：类型LocalDate，可选，对应status。
      * goods\_type：类型int，可选，筛选储藏货物类型。
      * cabinet\_type：类型int，可选，筛选储柜类型。
      * free\_stroage\_size：类型，可选，储藏柜以货物为单位的剩余储存空间。
    - 响应数据：List<Cabinet>。
  + 请求URI：GET /warehourse/<id>/cabinets/<id>
    - 说明：获取指定储柜信息
      * 请求参数：无
      * 响应数据：Cabinet
  + 请求URI：POST /warehourse/<id>/cabinets
    - 说明：增加一个储柜
    - 请求参数：置于请求体JSON中。
      * goods\_id：类型int，必选，储柜也是一种货物。
      * storage\_type：类型int，可选，储柜储存的货物类型。
    - 响应数据：无。
    - 响应字段：Location：新储柜的URI。
  + 请求URI：PATCH /warehourse/<id>/cabinets/<id>
    - 说明：修改储柜信息
    - 请求参数：置于请求体JSON中。
      * status：类型String，可选。
      * postion：类型String，可选。
      * goods\_type：类型int，可选。
    - 响应数据：无。
  + 请求URI：DELETE /warehourse/<id>/cabinets/<id>
    - 说明：删除储柜，仅仅是删除储柜功能，实际上还作为货物储存在仓库。
    - 请求参数：无
    - 响应数据：无
* 货物管理
  + 请求URI：GET /goodstypes
    - 说明：列出货物类型
    - 请求参数：
      * name：类型String,可选。
      * detail\_name：类型String，详细项名
      * detail\_value：类型String，详细项值
    - 响应数据：List<GoodsType>
  + 请求URI：GET /goodstypes/<id>
    - 说明：获取指定货物类型
    - 请求参数：无
    - 响应数据：GoodsType
  + 请求URI：POST /goodstypes
    - 说明：增加货物类型
    - 请求参数：置于请求体JSON中。
      * name：类型String，必选。
      * length：类型int，必选。
      * width：类型int，必选。
      * height：类型int，必选。
      * descript：类型String，必选。
      * details：类型List<Detail>，可选。
    - 响应数据：无
    - 响应头部：Location字段：新货物类型的URL
  + 请求URI：POST /goodstypes/<id>/detail
    - 说明：增加货物详细信息
    - 请求参数：至于请求体JSON中。
      * details：类型List<Detail>
    - 响应数据：无
  + 请求URI：PATCH /goodstypes/<id>
    - 说明：修改货物类型信息
    - 请求参数：置于请求体JSON中。
      * name：类型String，可选。
      * descript：类型String，可选。
      * length：类型int，可选。
      * width：类型int，可选。
      * height：类型int，可选。
    - 响应数据：无
  + 请求URI：DELETE /goodstypes/<id>
    - 说明：删除货物类型
    - 请求参数：无
    - 响应数据：无
  + 请求URI：DELETE /goodstypes/<id>/details
    - 说明删除详细信息
    - 请求参数：
      * details：类型List<Detail>
    - 响应数据：无
  + 请求URI：GET /warehourses/<id>/goodses
    - 说明：列出仓库货物信息
    - 请求参数：
      * offset：类型int，可选。
      * limit：类型int，可选。
      * status：类型String，可选。
      * time：类型LocalDate，可选，对应status。
      * before：类型LocalDate，可选，对应status。
      * after：类型LocalDate，可选，对应status。
      * type：类型int，可选。
      * name：类型String，可选。
      * detail\_name：类型String，详细项名
      * detail\_value：类型String，详细项值
      * 响应数据：List<Goods>
  + 请求URI：GET /warehourse/<id>/goodses/<id>
    - 说明：获取货物信息
    - 请求参数：无
    - 响应数据：Goods
  + 请求URI：POST /warehourse/<id>/goodses
    - 说明：增加一个货物
    - 请求参数：
      * type：类型int，必选。
      * status：类型Sring，可选。
    - 响应数据：无
    - 响应头部：Location字段：新货物的URL
  + 请求URI：PATCH /warehourse/<id>/goodses/<id>
    - 说明：修改货物信息
    - 请求参数：
      * status：类型String，可选。
      * cabinet：类型int，可选。
    - 响应数据：无
  + 请求URI：DELETE /warehourse/<id>/goodses/<id>
    - 说明：删除货物信息
    - 请求参数：无
    - 响应数据：无。
* 入库管理
  + 请求URI：GET /warehourse/<id>/inOrder
    - 说明：列出入库单
    - 请求参数：
      * offset：类型int，可选。
      * limit：类型int，可选。
      * status：类型String，可选。
      * time：类型LocalDate，可选，对应status。
      * before：类型LocalDate，可选，与status对应。
      * after：类型LocalDate，可选，与status对应。
    - 响应数据：List<InOrder>
  + 请求URI：GET /warehourse/<id>/inOrder/<id>
    - 说明：获取指定入库单
    - 请求参数：无
    - 响应数据：InOrder
  + 请求URI：POST /warehourse/<id>/inOrder
    - 说明：创建一个入库单
    - 请求参数：无
    - 响应数据：无
    - 响应头部：Location字段：新入库单的URL。
  + 请求URI：PATCH /warehourse/<id>/inOrder/<id>
    - 说明：更新入库单
    - 请求参数：
      * status：类型int
    - 响应数据：无
  + 请求URI：DELETE /warehourse/<id>/inOrder/<id>
    - 说明：删除入库单
    - 请求参数：无
    - 响应数据：无
  + 请求URI：GET /warehourse/<id>/inOrder/<id>/item
    - 说明：列出入库项
    - 请求参数：
      * offset：类型int，可选。
      * limit：类型int，可选。
      * status：类型String，可选。
      * time：类型LocalDate，可选，对应status。
      * before：类型LocalDate，可选，对应status。
      * after：类型LocalDate，可选，对应status。
      * goods\_type：类型int，可选。
      * cabinet\_type：类型int，可选。
      * goods\_id：类型int，可选。
      * cabinet\_id：类型int，可选。
    - 响应数据：List<InItem>
  + 请求URI：GET /warehourse/<id>/inOrder/<id>/item/<id>
    - 说明：获取指定入库项
    - 请求参数：无
    - 响应数据：Intem
  + 请求URI：GET /warehouse/<id>/inOrder/<id>/item/<id>
    - 说明：创建一项入库项
    - 请求参数：
      * goods\_id：类型int，必选。
      * cabinet\_id：类型int，可选。
    - 响应数据：无
    - 响应头部：Location字段：新入库项的URL。
  + 请求URI：PATCH /warehourse/<id>/inOrder/<id>/item/<id>
    - 说明：更新入库项
    - 请求参数：
      * status：类型String，可选。
      * goods\_id：类型int，可选。
      * cabinet\_id：类型int，可选。
  + 请求URI：DELETE /warehourse/<id>/inOrder/<id>/item/<id>
    - 说明：删除入库项目
    - 请求参数：无
    - 响应数据：无
* 出库管理
  + 请求URI：GET /warehourse/<id>/outOrder
    - 说明：列出入库单
    - 请求参数：
      * offset：类型int，可选。
      * limit：类型int，可选。
      * status：类型String，可选。
      * time：类型LocalDate，可选，与status对应。
      * before：类型LocalDate，可选，与status对应。
      * after：类型LocalDate，可选，与status对应。
    - 响应数据：List<outOrder>
  + 请求URI：GET /warehourse/<id>/outOrder/<id>
    - 说明：列出指定出库单
    - 请求参数：无
    - 响应数据：OutOrder
  + 请求URI：POST /warehouse/<id>/outOrder
    - 说明：创建出库单。
    - 请求参数：
      * limit：类型int，可选，限制此批出库货物数量。
    - 响应数据：无
    - 响应头部：Location字段：新出库单的URL。
  + 请求URI：PATCH /warehourse/<id>/outOrder/<id>
    - 说明：更新出库单
    - 请求参数：
      * status：类型int
    - 响应数据：无
  + 请求URI：DELETE /warhourse/<id>/outOrder/<id>
    - 说明：删除出库单
    - 请求参数：
    - 响应数据：无
  + 请求URI：GET /warehourse/<id>/outOrder/<id>/items
    - 说明：列出出库项
    - 请求参数：
      * offset：类型int，可选。
      * limit：类型int，可选。
      * status：类型String，可选。
      * time：类型LocaDate，可选，与status对应。
      * before：类型LocalDate，可选，与status对应。
      * after：类型LocalDate，可选，与status对应。
      * goods\_type：类型int，可选。
      * cabinet\_type：类型int，可选。
      * goods\_id：类型int，可选。
      * cabinet\_id：类型int，可选。
    - 响应数据：List<OutItem>
  + 请求URI：GET /warehourse/<id>/outOrder/<id>/items/<id>
    - 说明：选择特定出库项目
    - 请求参数：无
    - 响应数据：OutItem
  + 请求URI：POST /warehourse/<id>/outOrder/<id>/items
    - 说明：创建出库项
    - 请求参数：
      * goods：类型int，必选，货物id。
      * getitems：类型int，可选，GetItemid。
    - 响应数据：无
    - 响应头部：Location字段：新资源的URL。
  + 请求URI：PATCH /warehourse/<id>/outOrder/<id>/items/<id>
    - 说明：更新出库项目
    - 请求参数：
      * status：类型int，可选。
      * goods：类型int，可选，货物id。
      * cabinet：类型int，可选，储柜id。
    - 响应数据：无
  + 请求URI：DELETE /warehourse/<id>/outOrder/<id>/items/<id>
    - 说明：删除出库项
    - 请求参数：
    - 响应数据：无
* 取货单管理
  + 请求URI：GET /getOrder
    - 说明：列出取货单
    - 请求参数：
      * status：类型String，可选
      * time：类型LocalDate，可选，对应status。
      * before：类型LocalDate，可选，对应status。
      * after：类型LocalDate，可选，对应status。
      * address：类型String。
      * reason：类型String。
    - 响应数据：List<Order>
  + 请求URI：GET /getOrder/<id>
    - 说明：选择指定取货单
    - 请求参数：无
    - 响应数据：Order
  + 请求URI：POST /getOrder
    - 说明：创建取货单
    - 请求参数：
      * address：类型String，必选。
      * reason：类型String，必选。
    - 响应数据：无
    - 响应头部：Location字段：新资源的URL。
  + 请求URI：PATCH /getOrder/<id>
    - 说明：修改取货单
    - 请求参数：
      * status：类型int，可选。
      * address：类型String，可选。
      * reason：类型String，可选。
    - 响应数据：无
  + 请求URI：DELETE /getOrder/<id>
    - 说明删除取货单
    - 请求参数：无
    - 响应数据：无
  + 请求URI：GET /getOrder/<id>/items
    - 说明：列出取货项目
    - 请求参数：
      * status：类型int,可选。
      * time：类型LocalDate，可选，对应status。
      * before：类型LocalDate，可选，对应status。
      * after：类型LocalDate，可选，对应status。
      * warehouse：类型int，可选，仓库id。
      * goods\_type：类型int，可选，货物类型。
      * goods\_id：类型int，可选，货物id。
      * goods\_count：类型int，可选，货物数量。
    - 响应数据：List<OrderItem>
  + 请求URI：GET /getOrder/<id>/items/<id>
    - 说明：选择指定取货项目
    - 请求参数：无
    - 响应数据：GetItem
  + 请求URI：POST /getOrder/<id>/items
    - 说明：创建取货单项目
    - 请求参数：
      * goods\_type：类型int,必选
      * good\_count：类型int,必选
      * warehourse：类型int，可选。
      * goods\_id：类型int，可选。
    - 响应数据：无
    - 响应头部：Location字段：新资源的URL。
  + 请求URI：PATCH /getOrder/<id>/items/<id>
    - 说明：修改取货单项目
    - 请求参数：
      * status：类型String，可选。
      * good\_count：类型int，可选。
      * warehourse：类型int，可选。
      * goods\_id：类型int，可选，0值表示删除。
    - 响应数据：无
  + 请求URI：DELETE /getOrder/<id>/items/<id>
    - 说明：删除取货项
    - 请求参数：无
    - 响应数据：无
  1. 静态模型

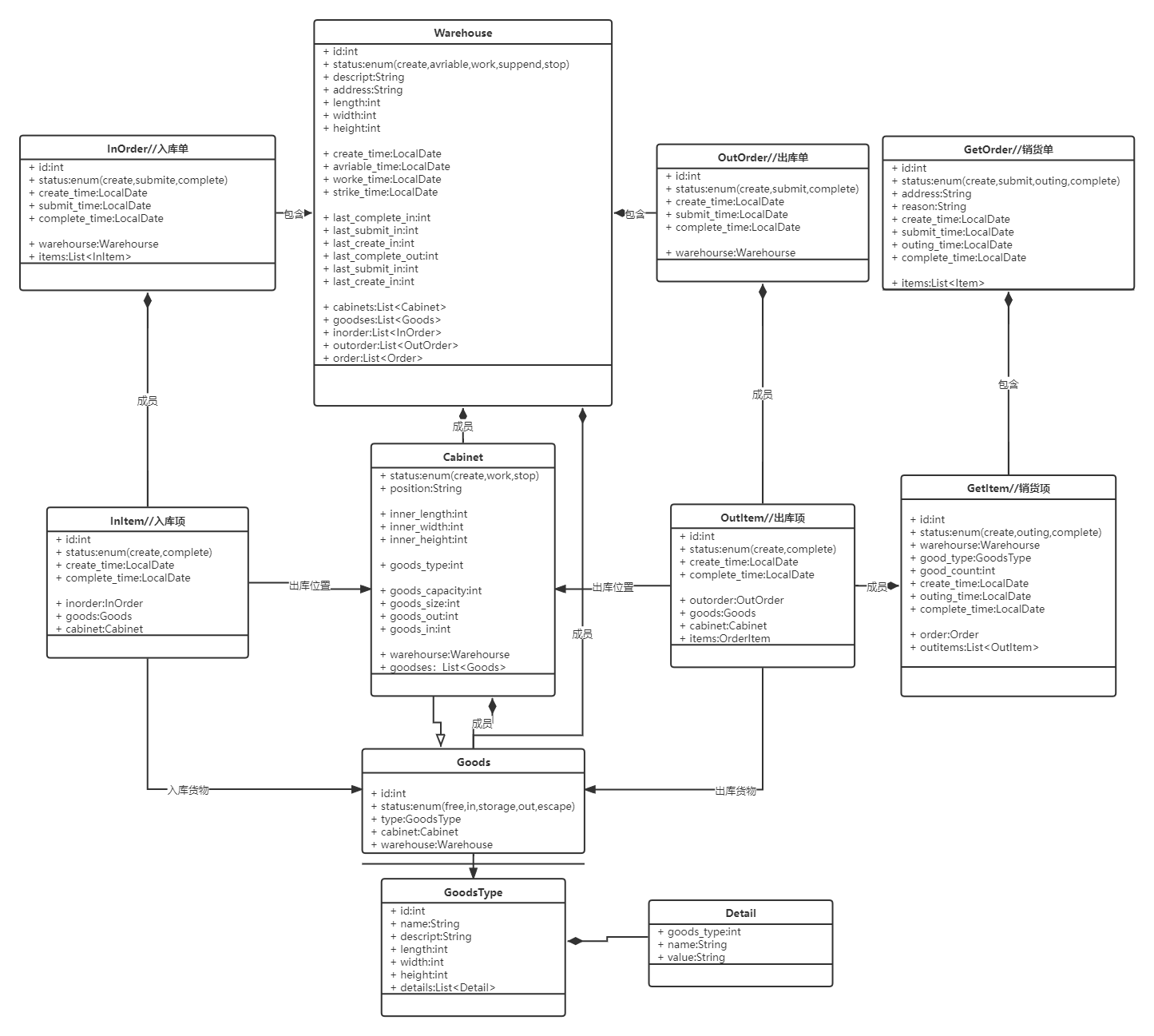


图 1 软件数据及其关联图

* 1. 数据库设计
     1. Warehourse表

仓库表记录了一些有关于仓库的信息。

create状态：表示新建仓库，但是仓库还不能工作。

available状态：表示该仓库已经准备好了，随时可以投入工作。

work状态：表示仓库正式开始工作，可以存取货物了。

suspend状态：表示仓库因为某种原因暂停工作，将来还有继续工作。

stop状态：表示仓库被废弃了，废弃的仓库以后都不能工作，只是保留存档而已。

表 9 仓库表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 取值 | 默认值 | null值 | 主键 | 外键 | 其他 |
| id | INT | 无 | 否 | 是 | 否 | 自增 |
| status | ENUM(create,available,work,suspend,stop) | create | 否 | 否 | 否 |  |
| description | CHAR(125) | 无 | 是 | 否 | 否 |  |
| name | CHAR(125) | 无 | 否 | 否 | 否 |  |
| address | CHAR(125) | 无 | 否 | 否 | 否 |  |
| length | INT，单位m | 无 | 是 | 否 | 否 |  |
| width | INT，单位m | 无 | 是 | 否 | 否 |  |
| height | INT，单位m | 无 | 是 | 否 | 否 |  |
| create\_time | DATETIME | 无 | 是 | 否 | 否 |  |
| avriable\_time | DATETIME | 无 | 是 | 否 | 否 |  |
| work\_time | DATETIME | 无 | 是 | 否 | 否 |  |
| suspend\_time | DATETIME | 无 | 是 | 否 | 否 |  |
| stop\_time | DATETIME | 无 | 是 | 否 | 否 |  |
| last\_complete\_in | INT | 无 | 是 | 否 | InOrder(id) |  |
| last\_submit\_in | INT | 无 | 是 | 否 | InOrder(id) |  |
| last\_create-in | INT | 无 | 是 | 否 | InOrder(id) |  |
| last\_complete\_out | INT | 无 | 是 | 否 | OutOrder(id) |  |
| last\_submit\_out | INT | 无 | 是 | 否 | OutOrder(id) |  |
| last\_complete\_out | INT | 无 | 是 | 否 | OutOrder(id) |  |

* + 1. Cabinet表

每个仓库包含若干个储藏柜，一个储藏柜储藏若干同型货物，储藏柜也是仓库的一个货物，只是这个货物不是被储藏的对象，而是用来储藏货物。

表 10 储藏柜表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 取值 | 默认值 | null值 | 主键 | 外键 | 其他 |
| id | INT | 无 | 否 | 是 | Goods(id) |  |
| status | enum(create,work,stop) | create | 否 | 否 | 否 |  |
| position | CHAR(125) | 无 | 是 | 否 | 否 |  |
| inner\_length | INT，单位m | 无 | 否 | 否 | 否 |  |
| inner\_width | INT，单位m | 无 | 否 | 否 | 否 |  |
| inner\_height | INT，单位m | 无 | 否 | 否 | 否 |  |
| goods\_type | INT | 无 | 是 | 否 | GoodsType(id) |  |
| goods\_capacity | INT | 无 | 是 | 否 | 否 |  |
| goods\_size | INT | 无 | 是 | 否 | 否 |  |
| goods\_out | INT | 无 | 是 | 否 | 否 |  |
| goods\_in | INT | 无 | 是 | 否 | 否 |  |

* + 1. Goods表

一个货物是指一个具体的货物，储藏在一个具体的储藏柜中。

free状态表示货物处于自由，意思是没有状态，这意味货物是直接放在仓库某处，例如作为储藏柜时，显然没有入库、出库以及储藏概念。

in状态表示货物被登记入库，当未真正储藏到合适的位置，可能只是在分拣区。

stroage状态表示货物已经储藏在储藏柜，入库工作已经全部完毕。

out状态表示货物已经被离开储藏的位置，但是尚未离开仓库。

escape状态寓意逃脱，值货物已经离开仓库了，所以逃离仓库的管理。

表 11 货物表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 取值 | 默认值 | null值 | 主键 | 外键 | 其他 |
| id | INT | 无 | 否 | 是 | 否 | 自增 |
| status | ENUM(free,in,storage,out,escape) | free | 否 | 否 | 否 |  |
| type | INT | 无 | 是 | 否 | GoodsType(id) |  |
| cabinet | INT | 无 | 是 | 否 | Cabinet(id) |  |
| warehouse | INT | 无 | 是 | 否 | Warehouse(id) |  |

* + 1. GoodsType表

货物类型表：描述同一类货物，比如小米手机和华为手机是两个不同类型货物。

表 12 货物类型表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 取值 | 默认值 | null值 | 主键 | 外键 | 其他 |
| id | INT | 无 | 否 | 是 | 否 | 自增 |
| name | CHAR(125) | 无 | 否 | 否 | 否 |  |
| descript | CHAR(125) | 无 | 否 | 否 | 否 |  |
| length | INT，单位cm | 无 | 否 | 否 | 否 |  |
| width | INT，单位cm | 无 | 否 | 否 | 否 |  |
| height | INT，单位cm | 无 | 否 | 否 | 否 |  |

* + 1. Detail表

GoodType表示只描述了货物类型中有利仓库管理的共同信息，不同类型货物需要不尽相同的描述。Detail描述了一个货物类型的其他信息，name是一个信息名，value则是对应值。如分辨率1092×200可表述为：name=“分辨率”，value=”1092,200”

表 13 货物类型详细信息表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 取值 | 默认值 | null值 | 主键 | 外键 | 其他 |
| goods\_type | INT | 否 | 否 | 否 | GoodsType(id) |  |
| name | CHAR(125) | 否 | 否 | 否 | 否 |  |
| value | CHAR(125) | 否 | 否 | 否 | 否 |  |

* + 1. InOrder表

每一个入库单代表一个仓库中一次入库的一批货物。

create状态表示还在不断登记入库货物。

submit状态表示提交这一批货物，接下来就是将货物放置到具体的储藏柜中。

complete状态表示这一个批货物已经全部安置妥当，货物的入库工作已经完成。

表 14 入库单表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 取值 | 默认值 | null值 | 主键 | 外键 | 其他 |
| id | INT | 无 | 否 | 是 | 否 |  |
| status | ENUM(create,submit,complete) | create | 否 | 否 | 否 |  |
| create\_time | DATETIME | 无 | 是 | 否 | 否 |  |
| submit\_time | DATETIME | 无 | 是 | 否 | 否 |  |
| complete\_time | DATETIME | 无 | 是 | 否 | 否 |  |
| warehourse | INT | 无 | 是 | 否 | Warehourse(id) |  |

* + 1. InItem表

每一个入库项是一个入库单的详细条目。

create状态表示入库项还在建立，此时入库单状态可能为create和submit。

complete状态表示该入库项的入库工作已经完成，即货物的状态为storage。

表 15 入库项表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 取值 | 默认值 | null值 | 主键 | 外键 | 其他 |
| id | INT | 无 | 否 | 是 | 否 |  |
| status | ENUM(create,complete) | create | 否 | 否 | 否 |  |
| create\_time | DATETIME | 无 | 是 | 否 | 否 |  |
| complete\_time | DATETIME | 无 | 是 | 否 | 否 |  |
| inorder | INT | 无 | 是 | 否 | InOrder(id) |  |
| goods | INT | 无 | 是 | 否 | Goods(id) |  |
| cabinet | INT | 无 | 是 | 否 | Cabinet(id) |  |

* + 1. OutOrder表

create状态表示还在不断登记出库货物。

submit状态表示提交这一批货物，接下来就是取出具体货物。

complete状态表示这一个批货物已经离库，货物的出库工作已经完成。

表 16 出库单表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 取值 | 默认值 | null值 | 主键 | 外键 | 其他 |
| id | INT | 无 | 否 | 是 | 否 |  |
| status | ENUM(create,submit,complete) | create | 否 | 否 | 否 |  |
| create\_time | DATETIME | 无 | 是 | 否 | 否 |  |
| submit\_time | DATETIME | 无 | 是 | 否 | 否 |  |
| complete\_time | DATETIME | 无 | 是 | 否 | 否 |  |
| warehourse | INT | 无 | 是 | 否 | Warehourse(id) |  |

* + 1. OutItem表

每一个出库项是一个出库单的详细条目。与一个取货项关联，表示其的一个货物。

create状态表示出库项还在建立，此时出库单状态可能为create和submit。

complete状态表示该出库项的出库工作已经完成，即货物的状态为escape。

表 17 出库项表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 取值 | 默认值 | null值 | 主键 | 外键 | 其他 |
| id | INT | 无 | 否 | 是 | 否 |  |
| status | ENUM(create,complete) | create | 否 | 否 | 否 |  |
| create\_time | DATETIME | 无 | 是 | 否 | 否 |  |
| complete\_time | DATETIME | 无 | 是 | 否 | 否 |  |
| outorder | INT | 无 | 是 | 否 | OutOrder(id) |  |
| goods | INT | 无 | 是 | 否 | Goods(id) |  |
| cabinet | INT | 无 | 是 | 否 | Cabinet(id) |  |
| getitem | INT | 无 | 是 | 否 | GetItem(id) |  |

* + 1. GetOrder表

一个取货单可能取货于多个仓库，货物可能分几个包裹发送。

create状态表示取货单还来构建中。

submit状态表示取货单已经提交，不再修改。

outing\_time状态表示取货单正在出库，送货中。

complete\_item状态表示取货单已经完成，收货人已经收到货物。

表 18 取货单表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 取值 | 默认值 | null值 | 主键 | 外键 | 其他 |
| id | INT | 无 | 否 | 是 | 否 |  |
| status | ENUM(create,submit,out,complete) | 无 | 否 | 否 | 否 |  |
| address | CHAR(125) | 无 | 否 | 否 | 否 |  |
| reason | CHAR(125) | 无 | 否 | 否 | 否 |  |
| create\_time | DATETIME | 无 | 是 | 否 | 否 |  |
| submit\_time | DATETIME | 无 | 是 | 否 | 否 |  |
| outing\_time | DATETIME | 无 | 是 | 否 | 否 |  |
| complete\_time | DATETIME | 无 | 是 | 否 | 否 |  |

* + 1. GetItem表

一个取货项包含若干个同类货物，具体货物由关联的出库项表示。

一个取货项的货物必须是同一个包裹发送，不能拆分。

create状态表示取货项创建中或已提交（提交取货单即提交取货项），故此时取货单的状态为create或submit。

outing状态表示取货项出库，送货中。

complete状态表示此项货物已经收获。

表 19 取货项表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 取值 | 默认值 | null值 | 主键 | 外键 | 其他 |
| id | INT | 无 | 否 | 是 | 否 |  |
| status | ENUM(create,outing,complete) | create | 是 | 否 | 否 |  |
| create\_time | DATETIME | 无 | 是 | 否 | 否 |  |
| outing\_time | DATETIME | 无 | 是 | 否 | 否 |  |
| complete\_time | DATETIME | 无 | 是 | 否 | 否 |  |
| warehourse | INT | 无 | 是 | 否 | Warehourse(id) |  |
| goods\_type | INT | 无 | 是 | 否 | GoodsType(id) |  |
| goods\_count | INT | 无 | 是 | 否 | 否 |  |
| getorder | INT | 无 | 是 | 否 | GetOrder(id) |  |

系统实现

Spring SpringMVC 配置

需要适当的配置，才能使用Spring的IOC与AOP服务，以及SpringMVC功能。

Servlet容器

根据JEE标准，需要一个Servlet容器才能使用Java Web 开发。Apache Tomcat是一个开源的Servlet容器并附带一个小型的HTTP服务。可以通过配置来使用服务器中安装的Apache Tomcat，但是为了方便，这里直接使用嵌入式的Tomcat。

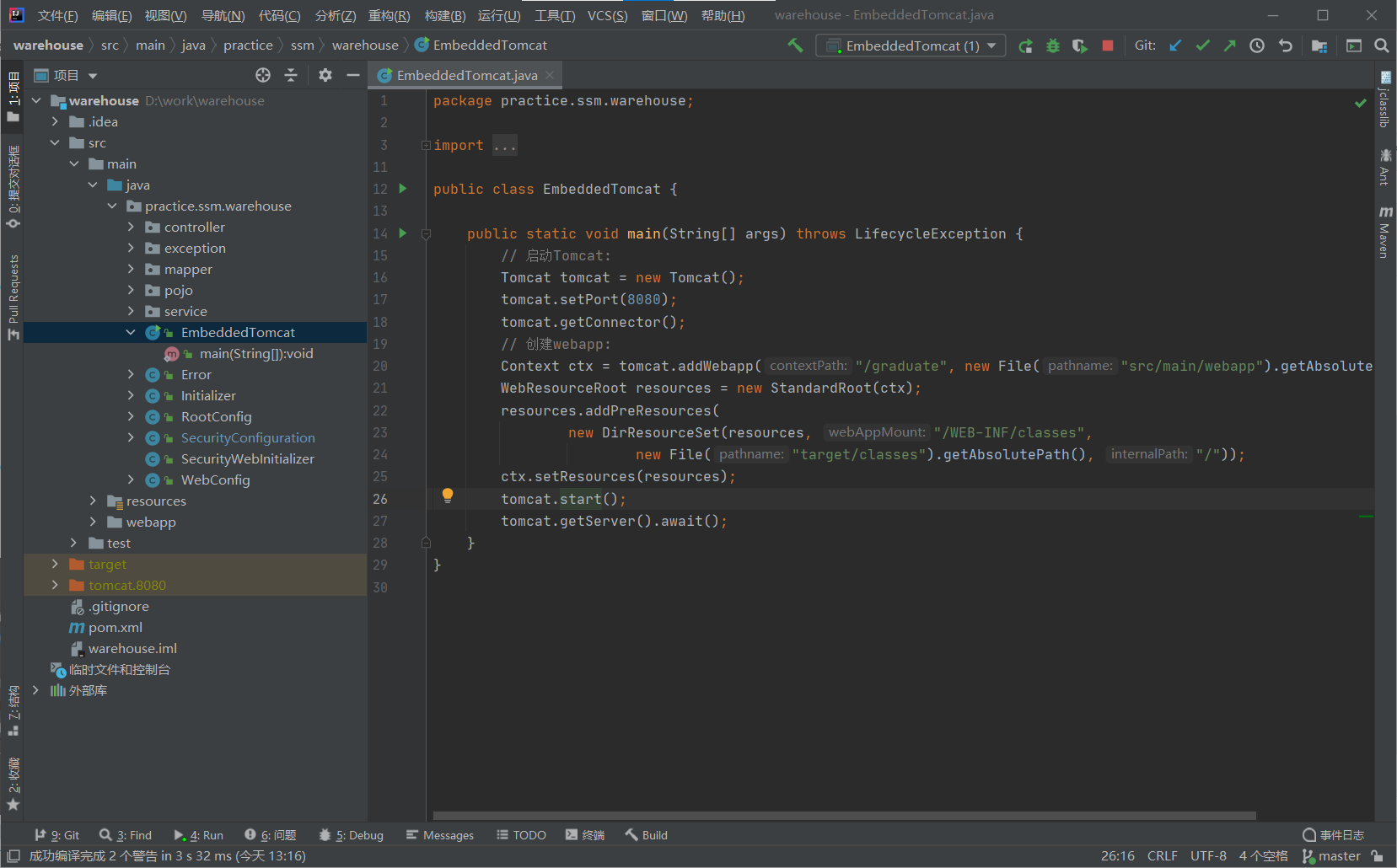


图 2 嵌入式Tomcat

Servlet 初始化器

如果Servlet容器检测到 Servlet初始化器的实现，那么Servlet容器就会自动使用Servlet初始化器。利用这个特点，可以实现Servlet 初始化器，然后就可以在Servlet容器启动后，指定Spring,SpringMVC的初始化与配置工作。

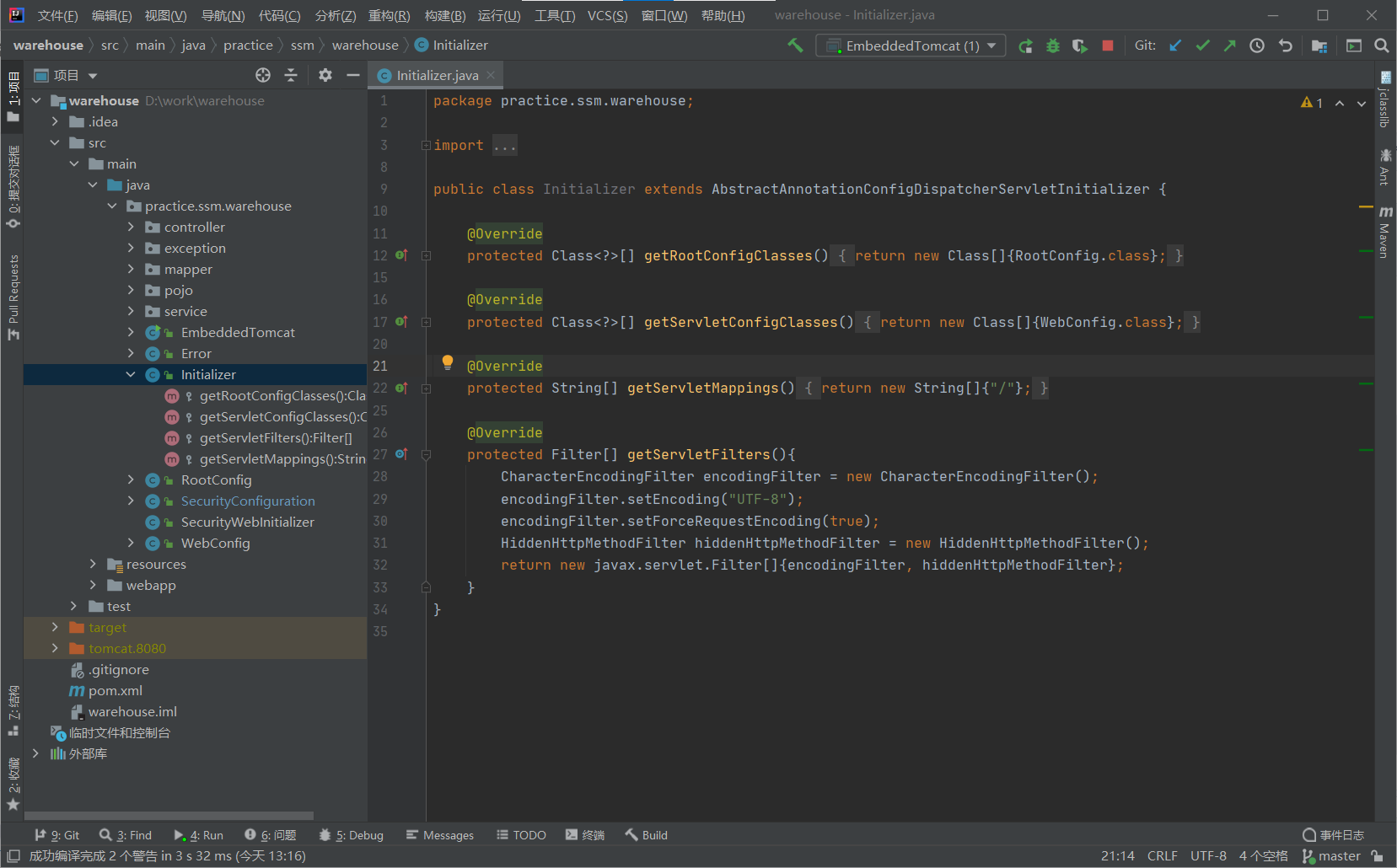


图 3 Servlet 初始化器

Spring 配置

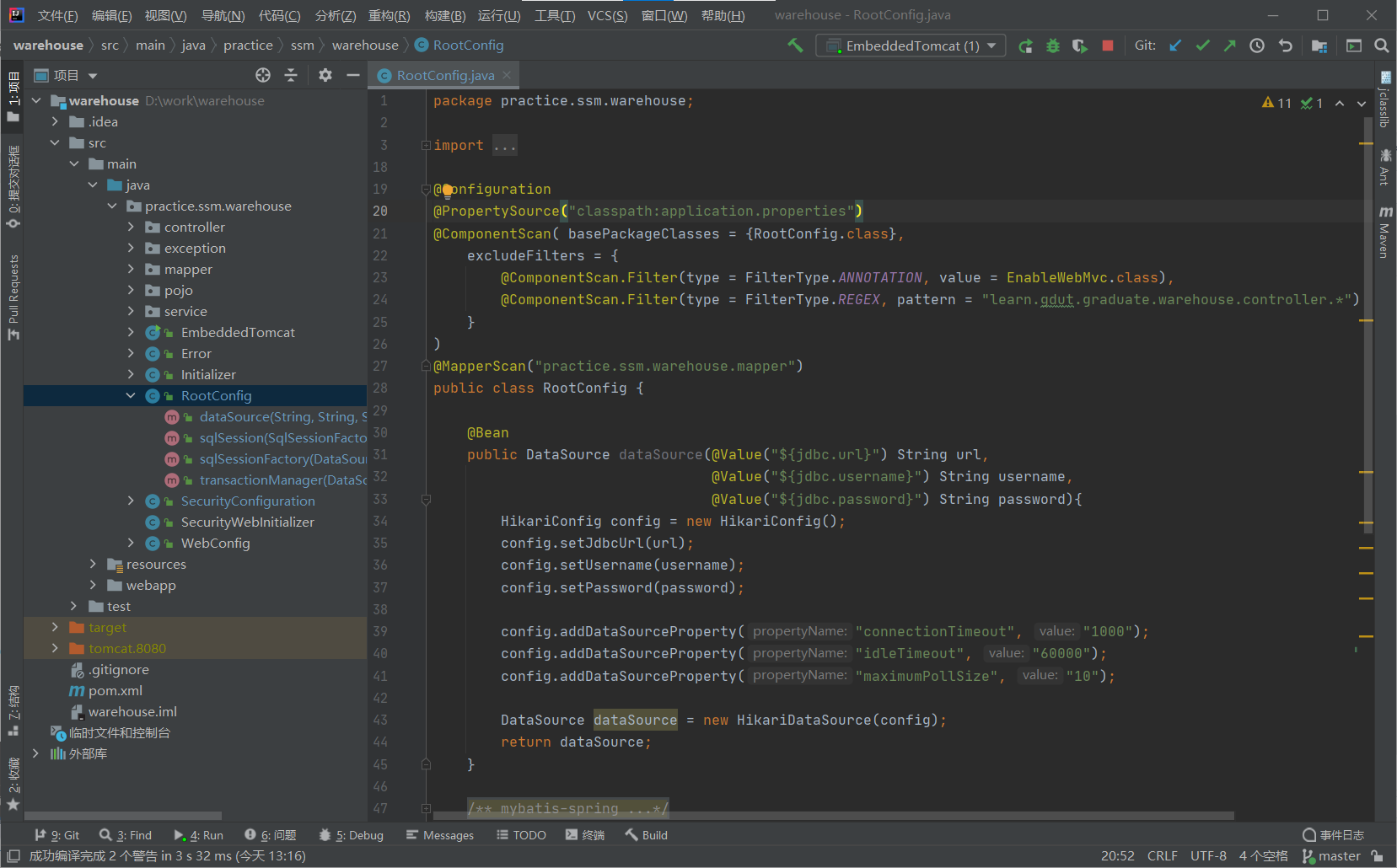


图 4 Spring 配置

Spring MVC 配置

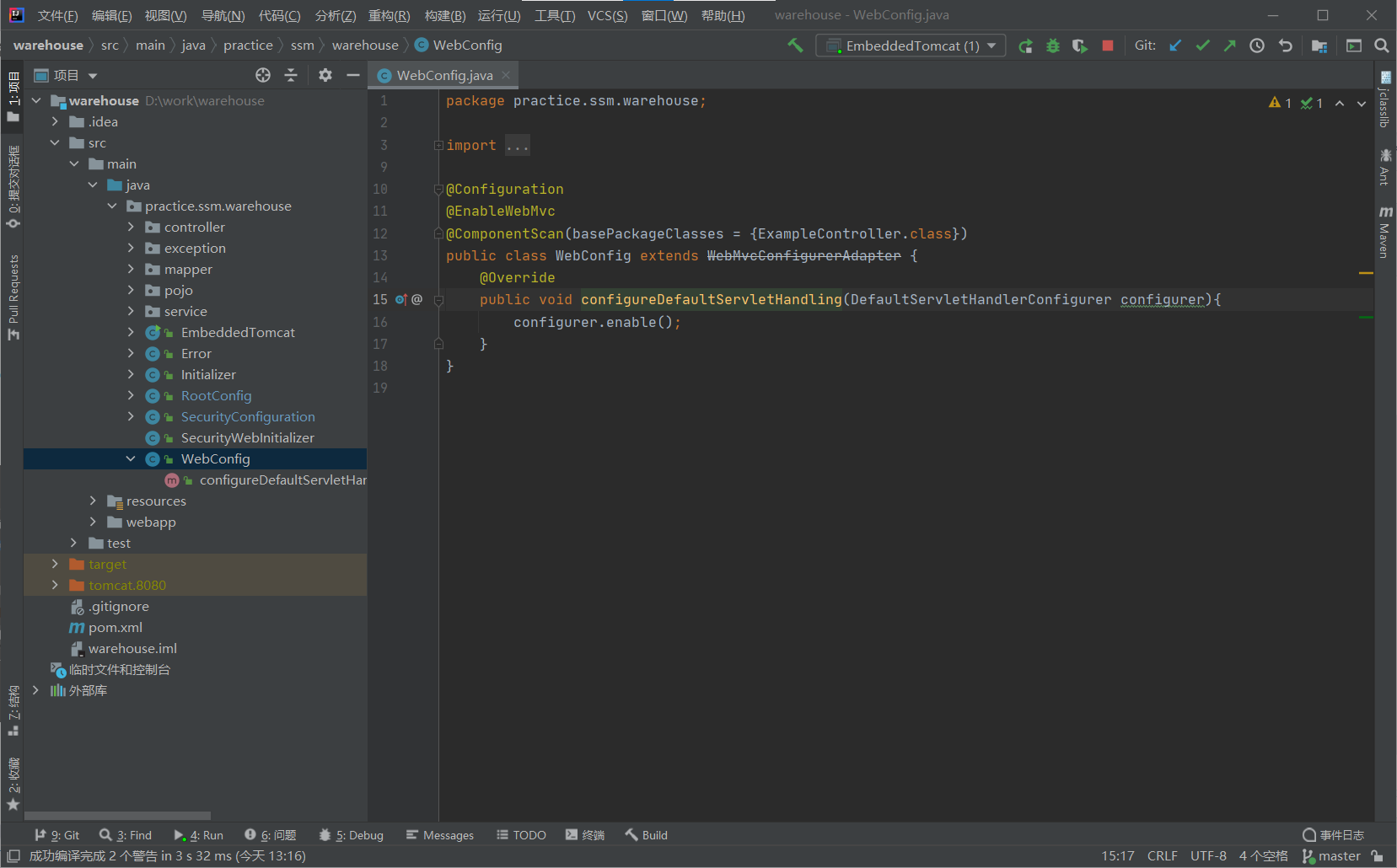


图 5 Spring MVC 配置

登录实现

登录界面

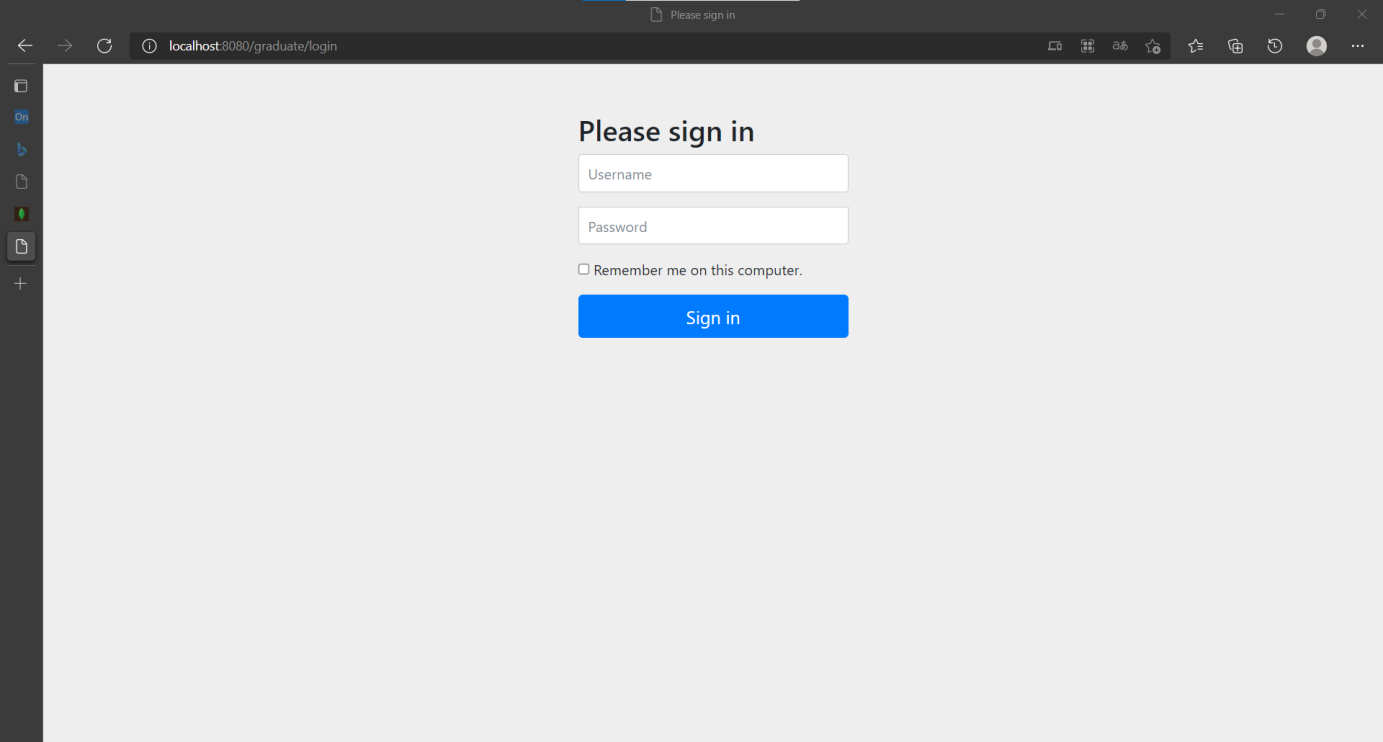


图 6 登录界面

登记接口

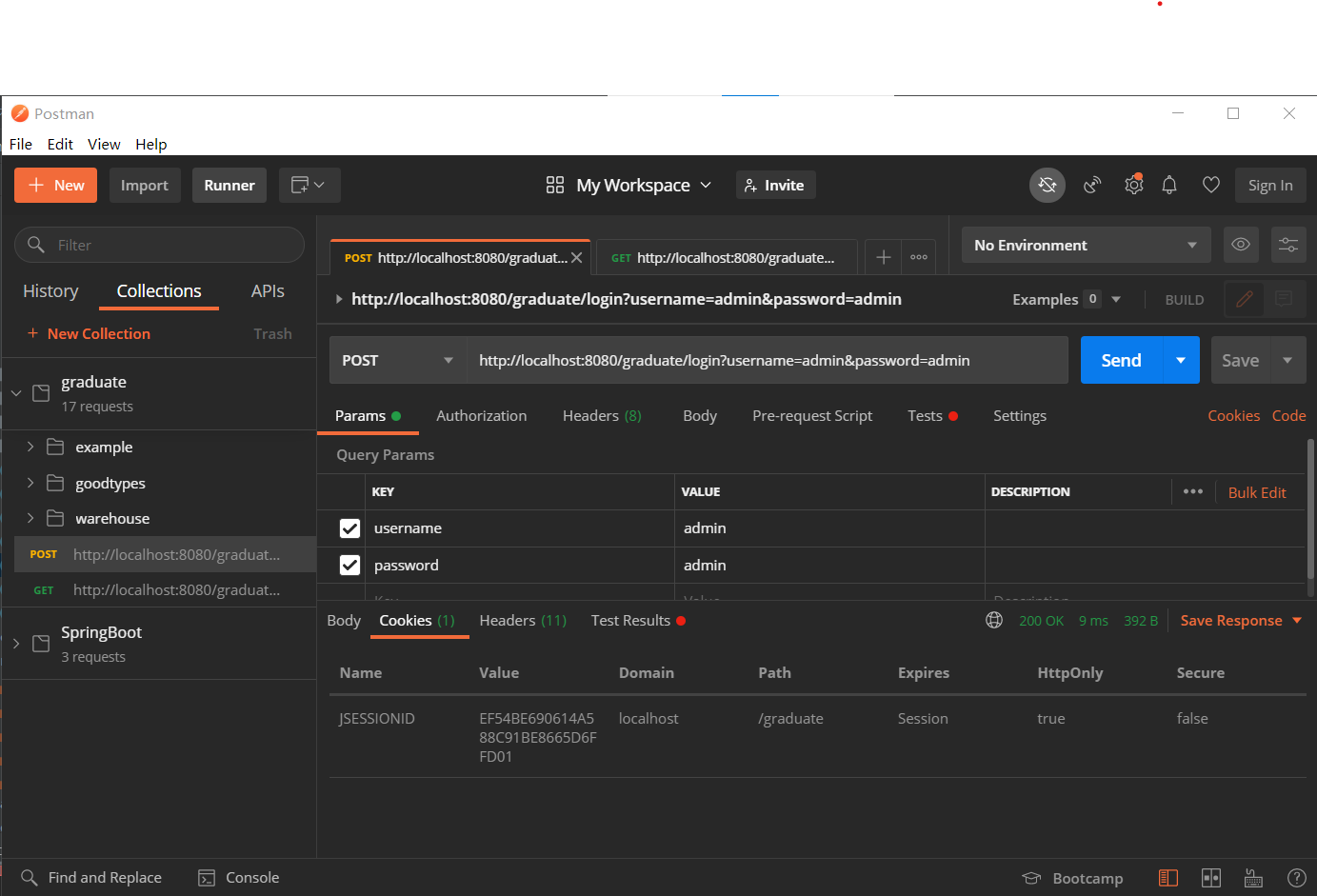


图 7 登录接口

数据库实现

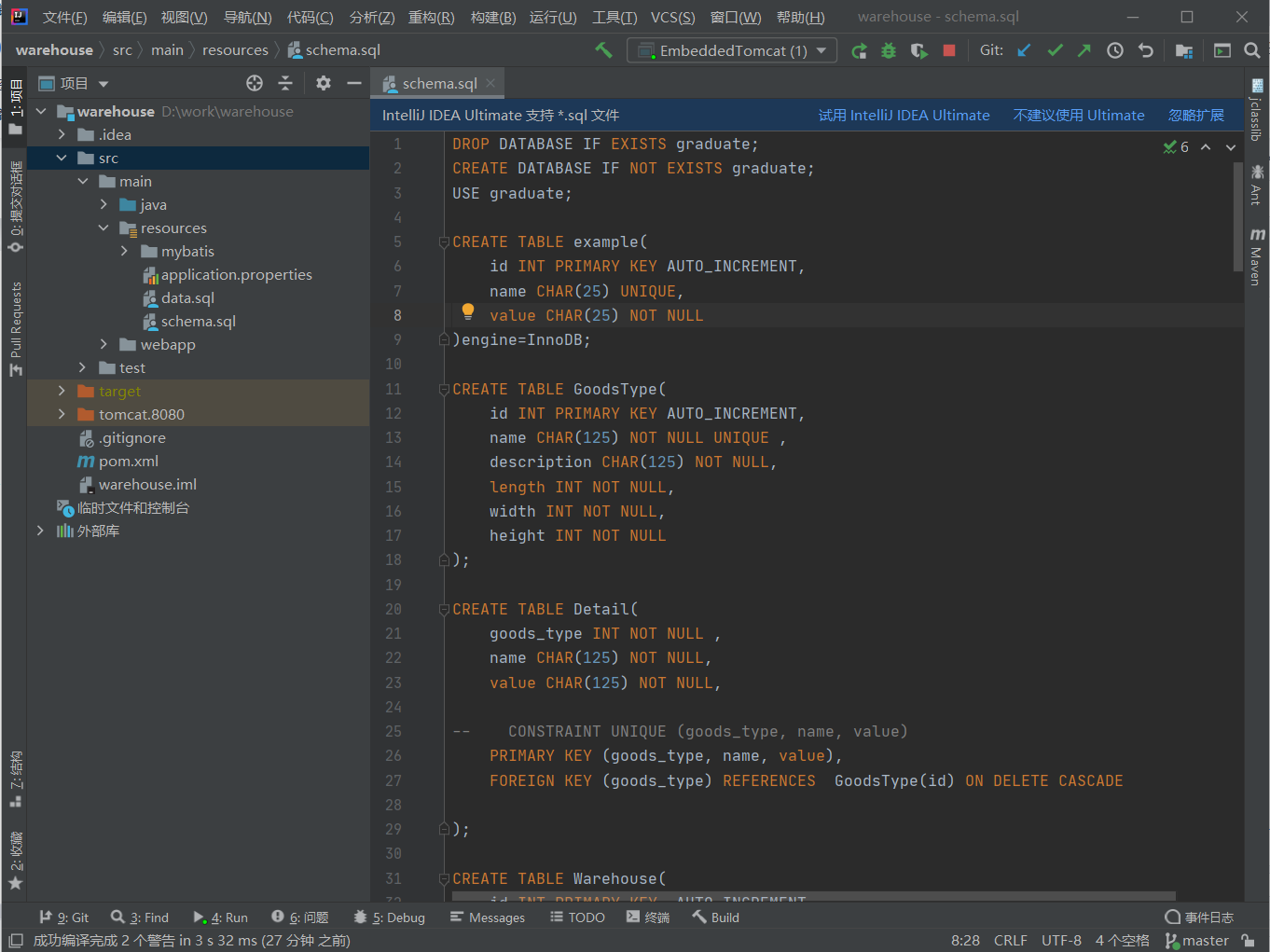


图 8.1 数据库实现

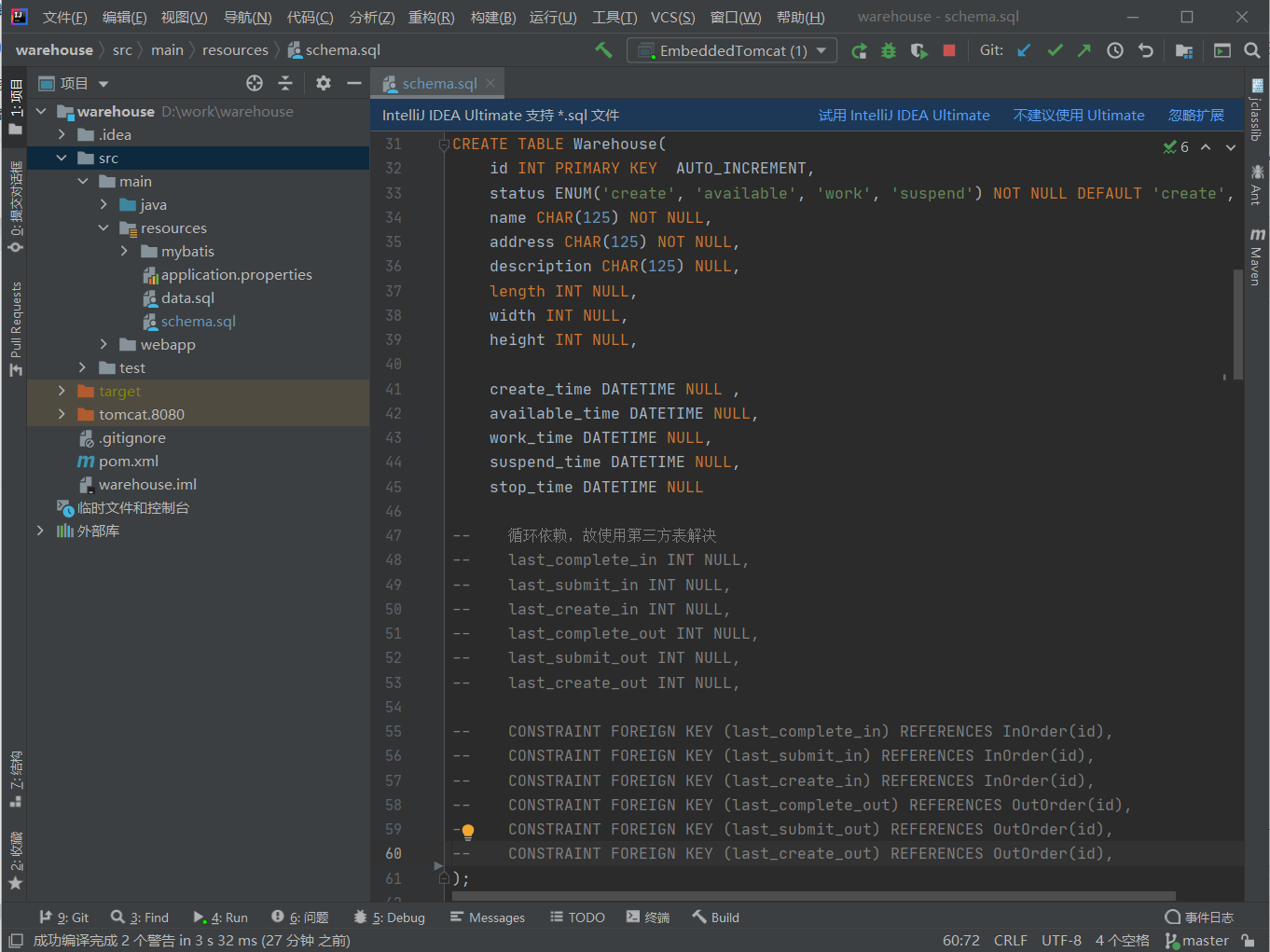


图 4.2 数据库实现

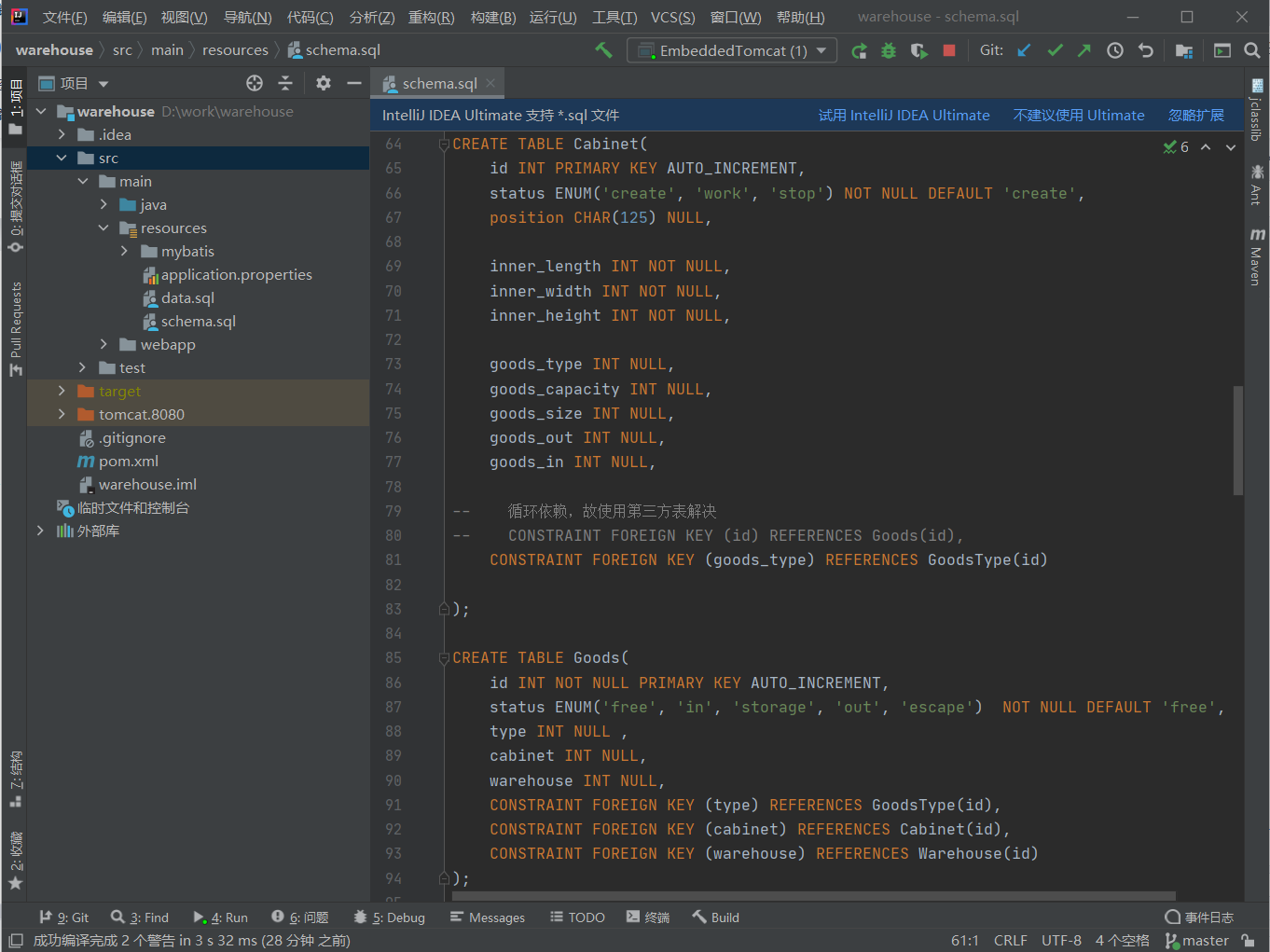


图 4.3 数据库实现

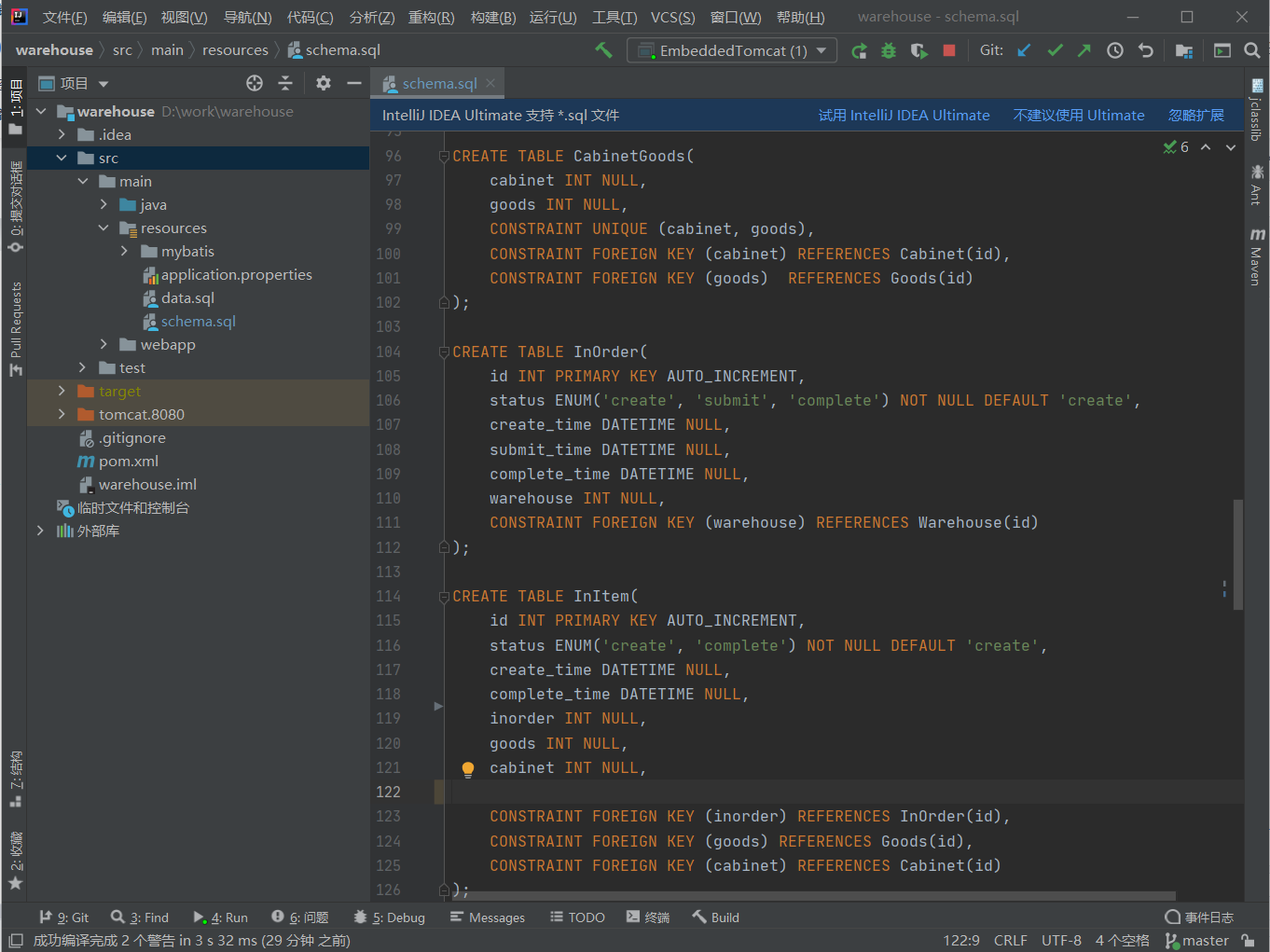


图 4.4 数据库实现

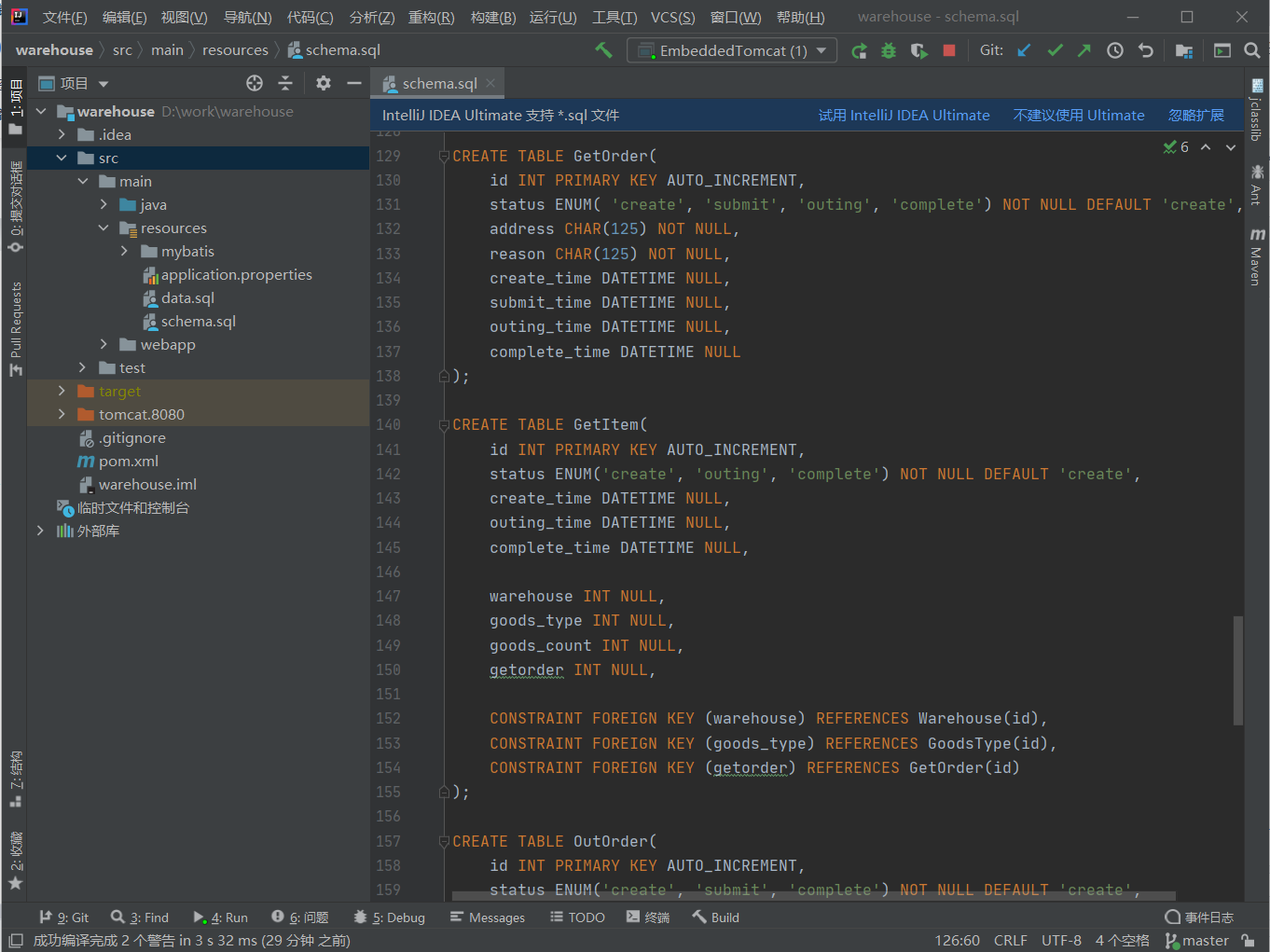


图 4.5 数据库实现

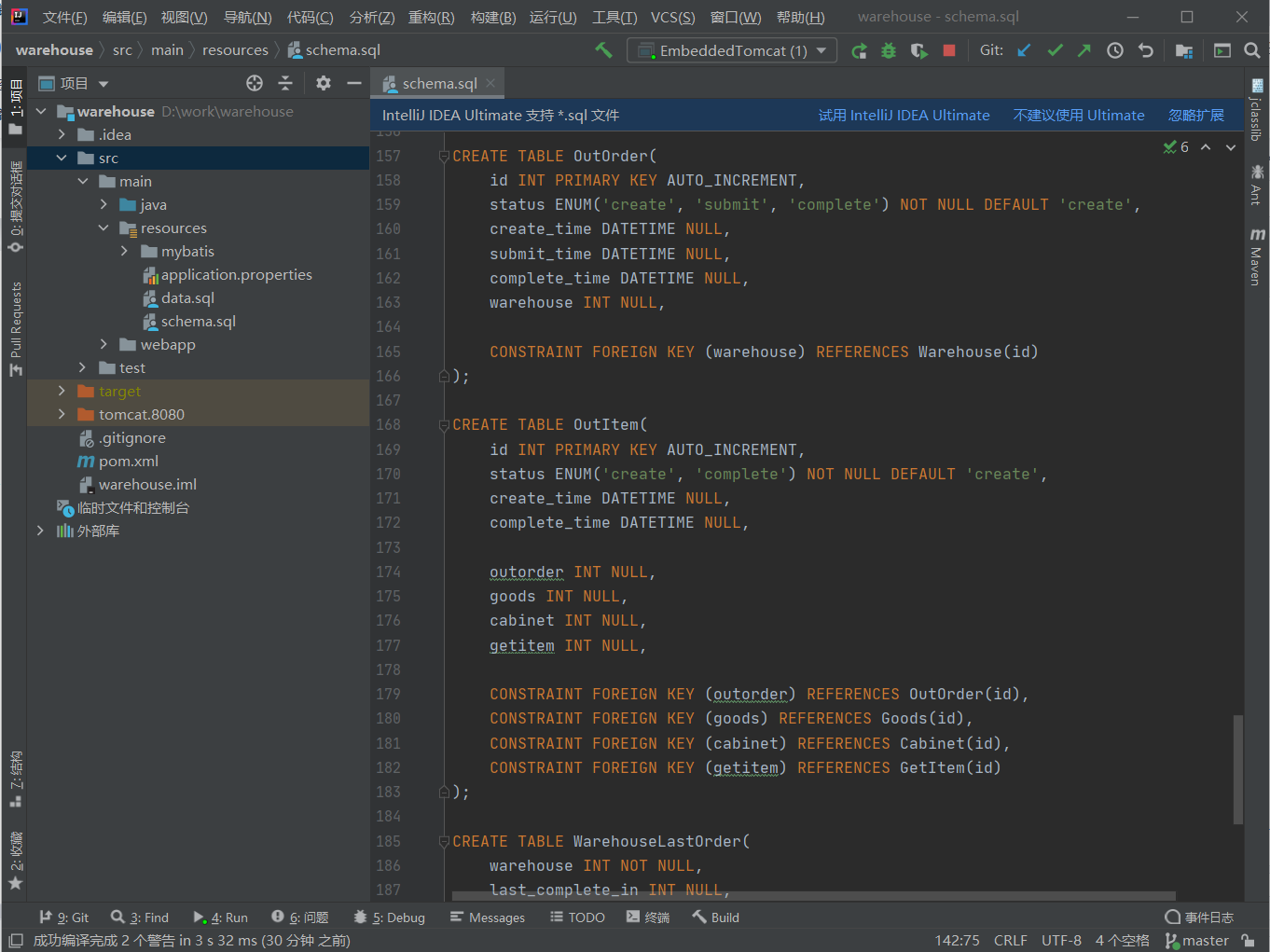


图 4.6 数据库实现

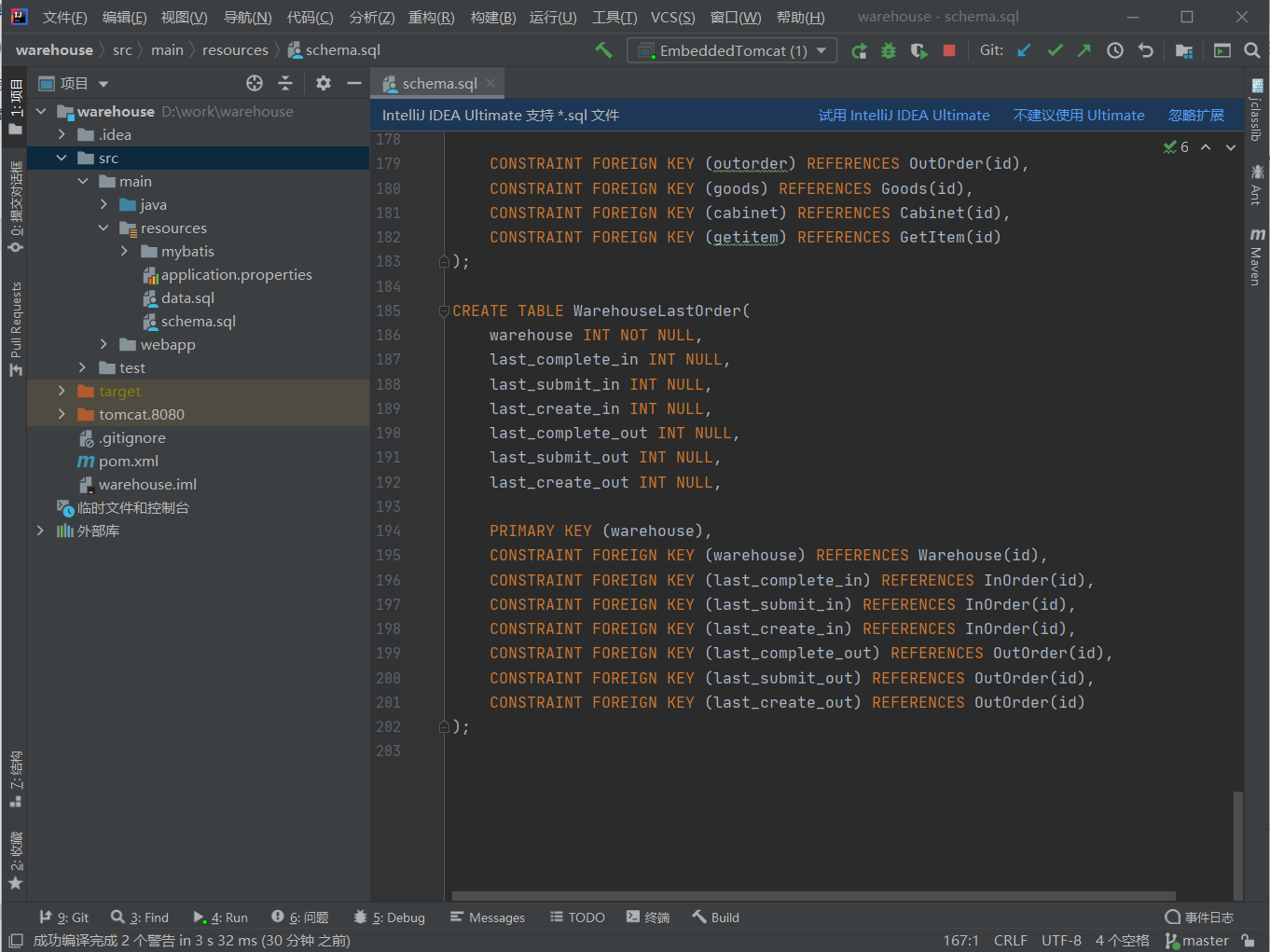


图 4.7 数据库实现

DAO层实现

MyBatis配置文件

MyBatis是一个Java开源的ORM框架，封装SQL查询语句，在程序实体类和数据库表之间自动映射。

配置文件则配置了Mybatis的工作配置，这将影响到Mybatis操作数据的工作。

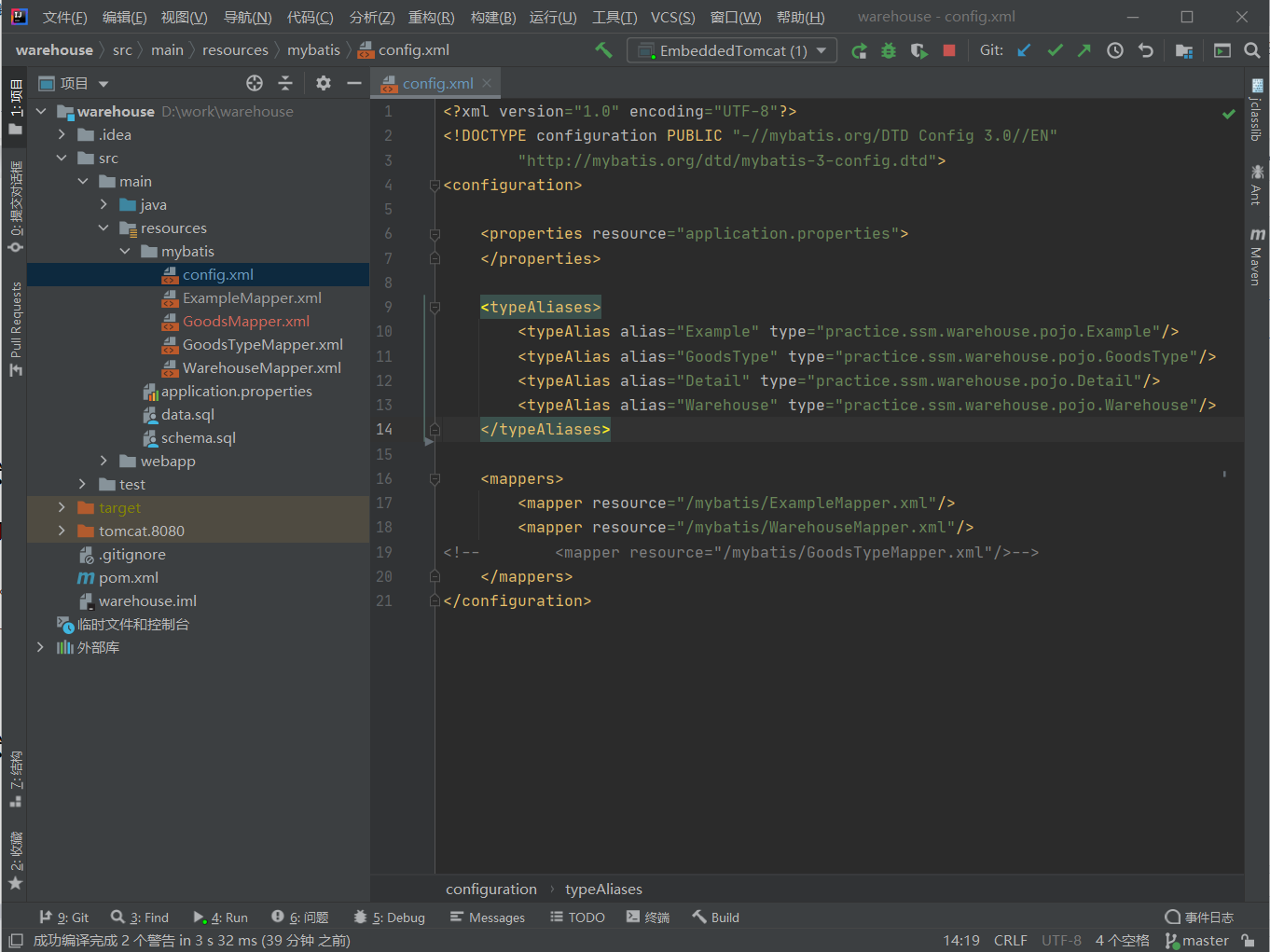


图 9 MyBatis配置文件

MyBatis映射文件

MyBatis映射文件定义了具体数据库表与程序实体类（即POJO层类）的映射关系，并封装SQL语句。

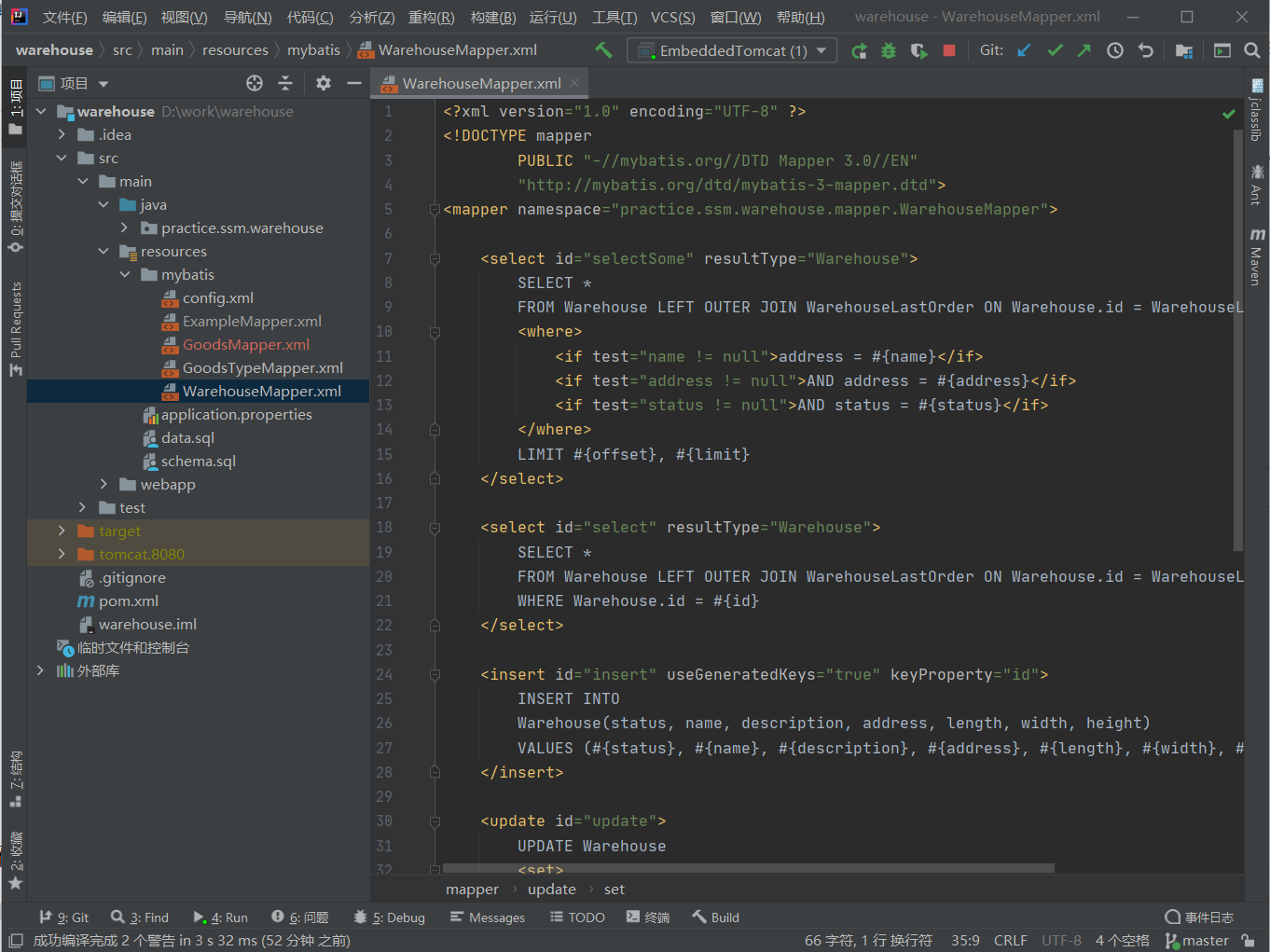


图 10 MyBatis映射文件

MyBatis映射器接口

为MyBatis提供一写接口，并在接口中提供符合映射规则的方法，并设置适当的配置。则这些接口会被MyBatis解释为映射器接口，并自动生成接口实现。通过调用接口实现的方法，就会执行对应的在映射文件中的SQL语句，并自动将程序数据和数据库数据直接转换。

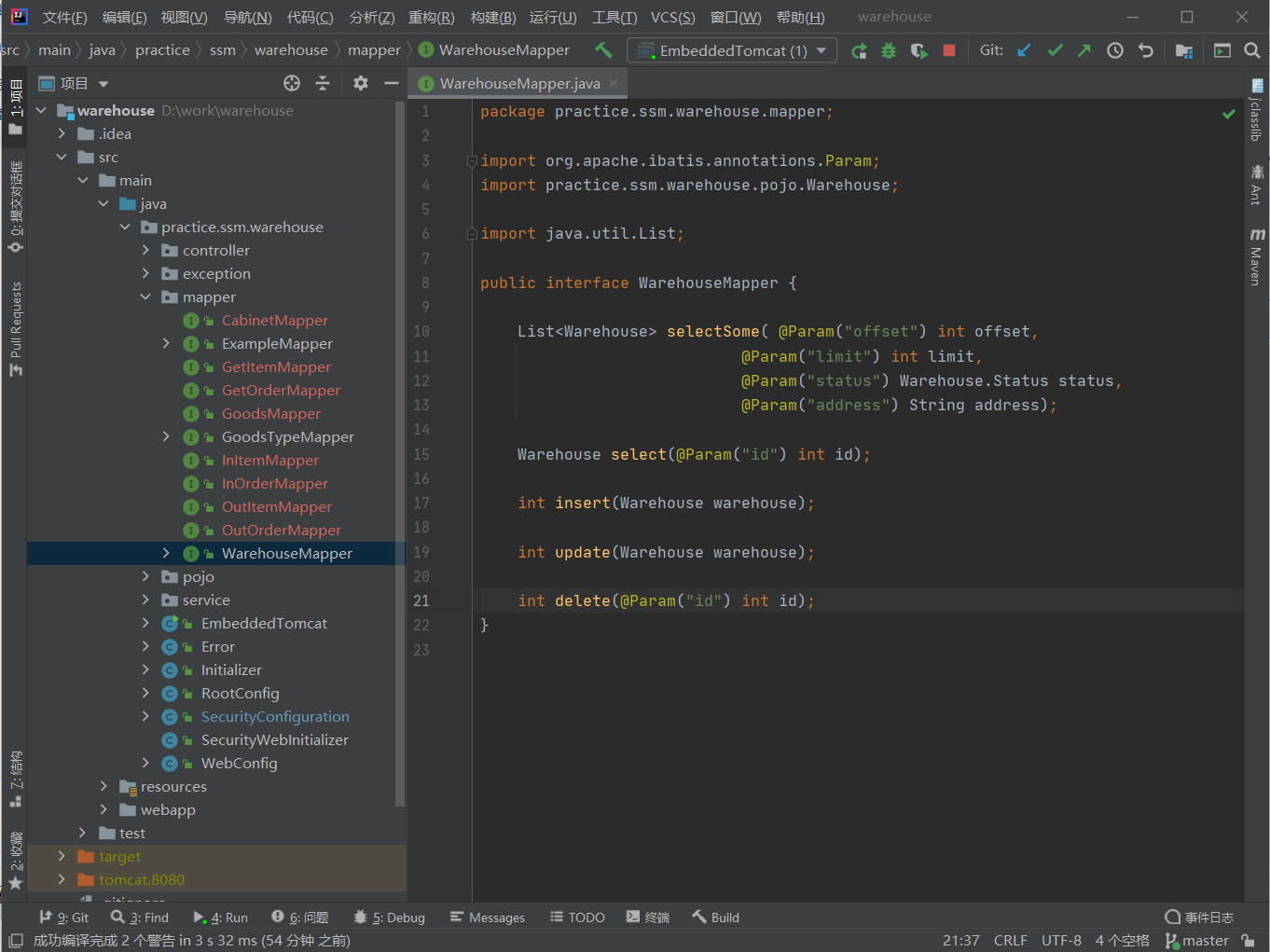


图 11 MyBatis映射器接口

POJO 层实现

为了避免重复繁琐的编码工作，对于POJO类这种只有简单的Getter和Setter方法的类。可以再Ideal中使用lombak插件，然后通过lombok注解指示如何自动生成相关方法。这些方法的代码将自动生成并插入字节码文件，源码文件就可简洁到只剩下域定义。

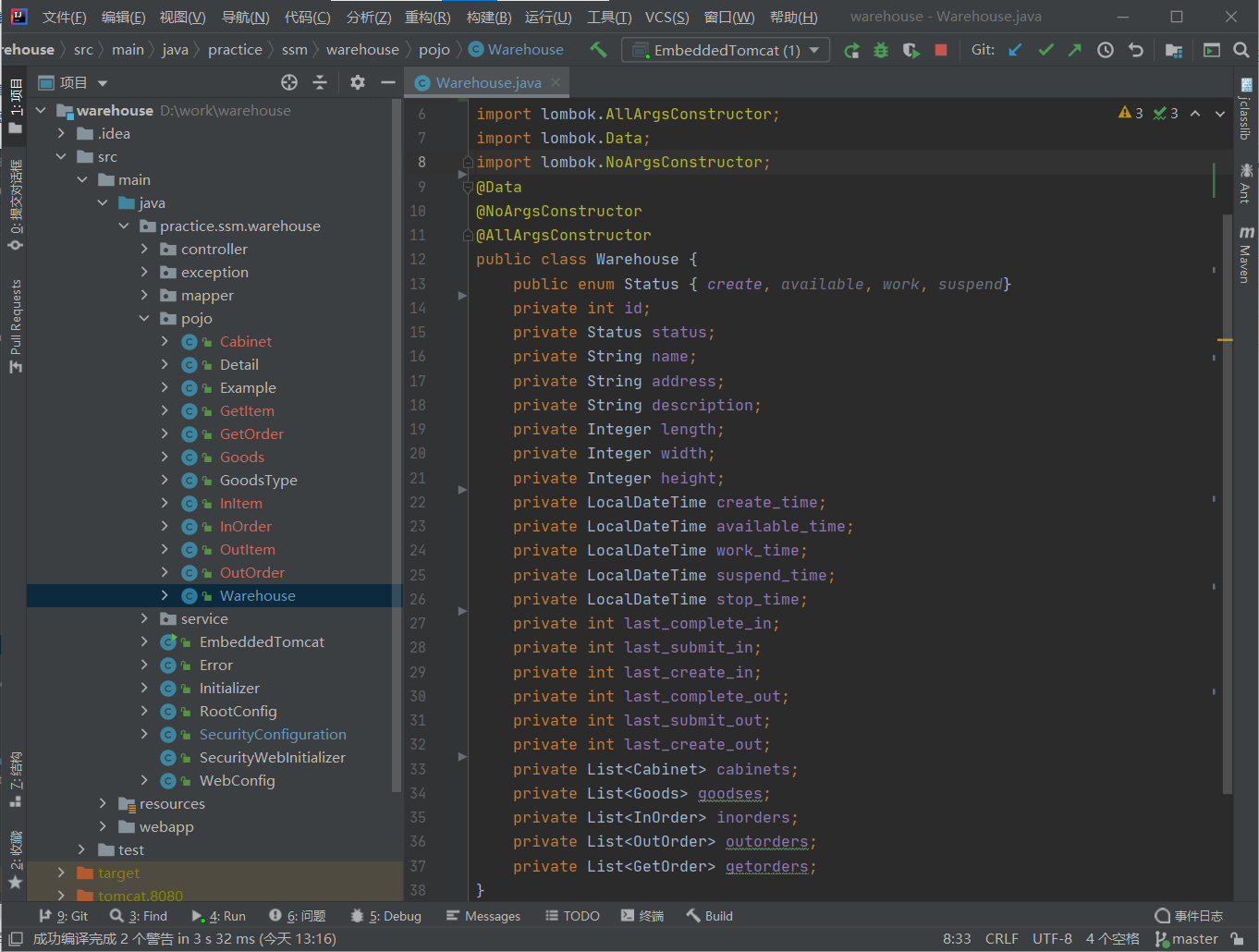


图 12 POJO 实现

Service层实现

Service层接口

为了降低程序代码的耦合度，先设计接口，然后再设计其实现。这样子设计，即便更换不同实现，也能保证统一的接口，而不会影响Controller层代码。

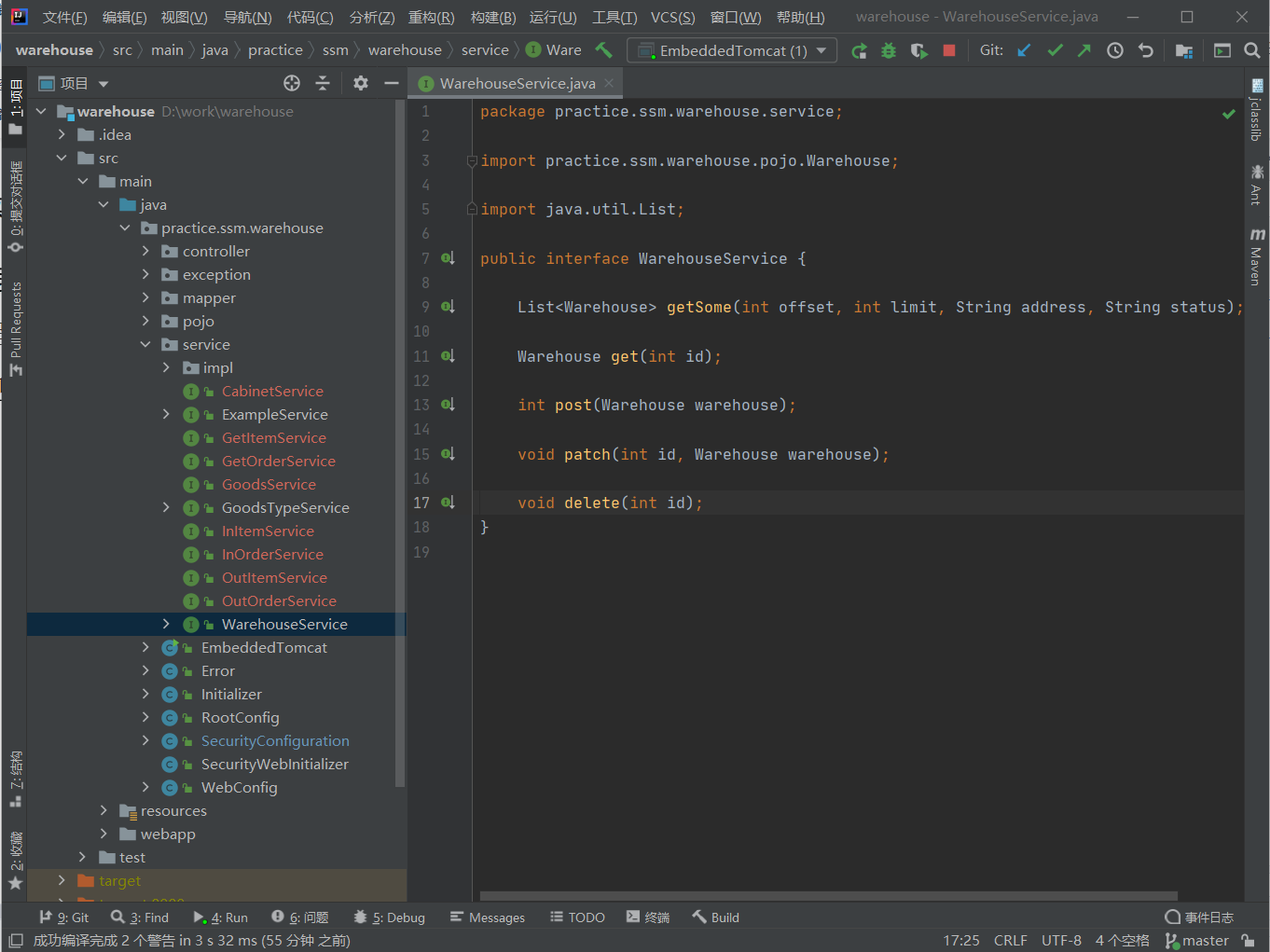


图 13 Service 层接口

Service 层接口实现

Service 层接口调用DAO层的数据访问服务，实现具体的服务逻辑，为Controller层提供服务。

@Service注解，是Spring的@Component接口的变体，用于自动发现类并实例化。

@Autowirde注解，是Spring的依赖注入注解，Spring会查找合适的类实例，并自动添加依赖。

@Transactional注解，是Spring Data 的事务管理注解，被注解的类的方法，将被Spring 代理，然后管理它的数据库事务。

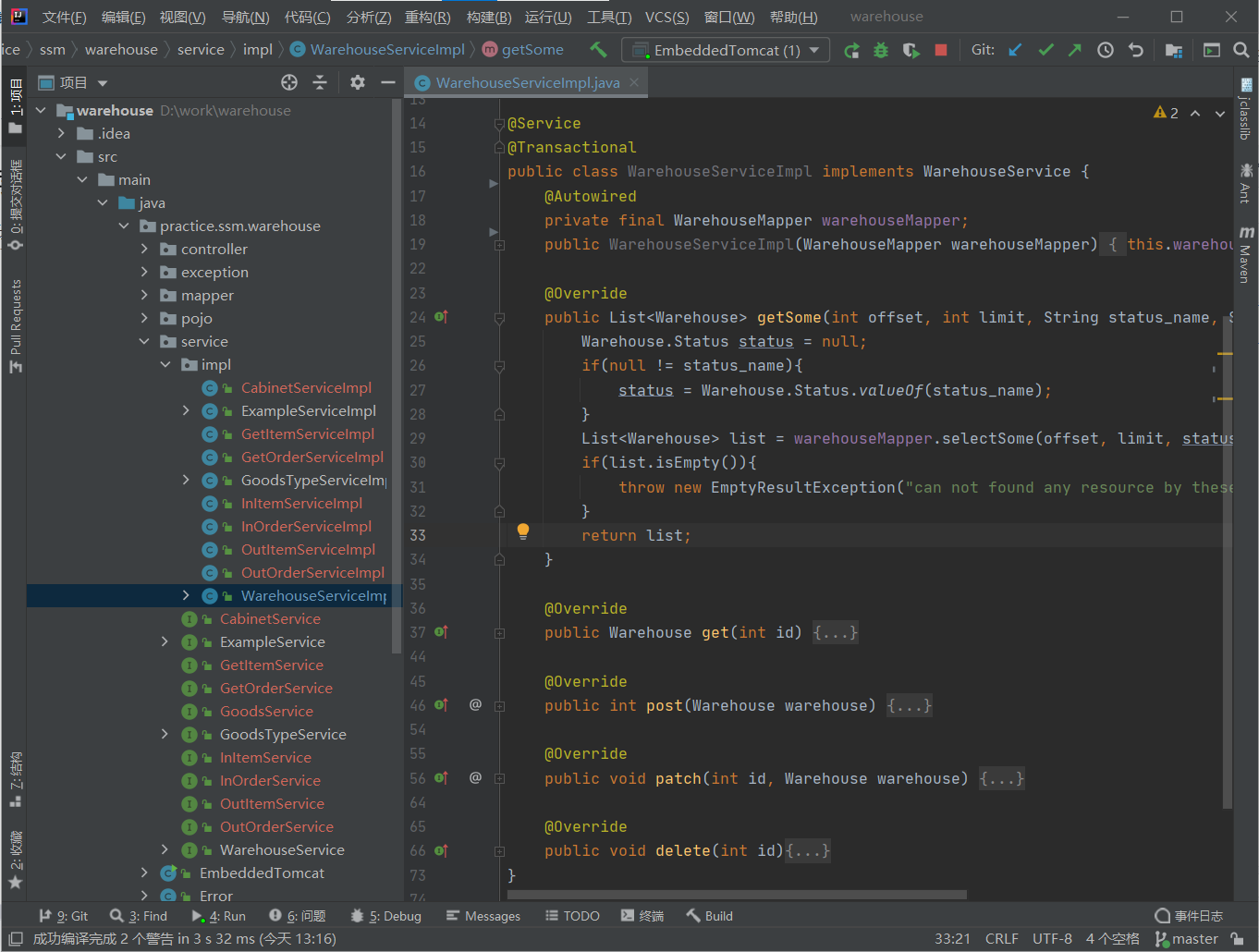


图 14 Service 层接口实现

Controller 层实现

Controller 层实现网络功能，控制业务流程。通过如图所示的一系列注解，控制HTTP 请求与响应。

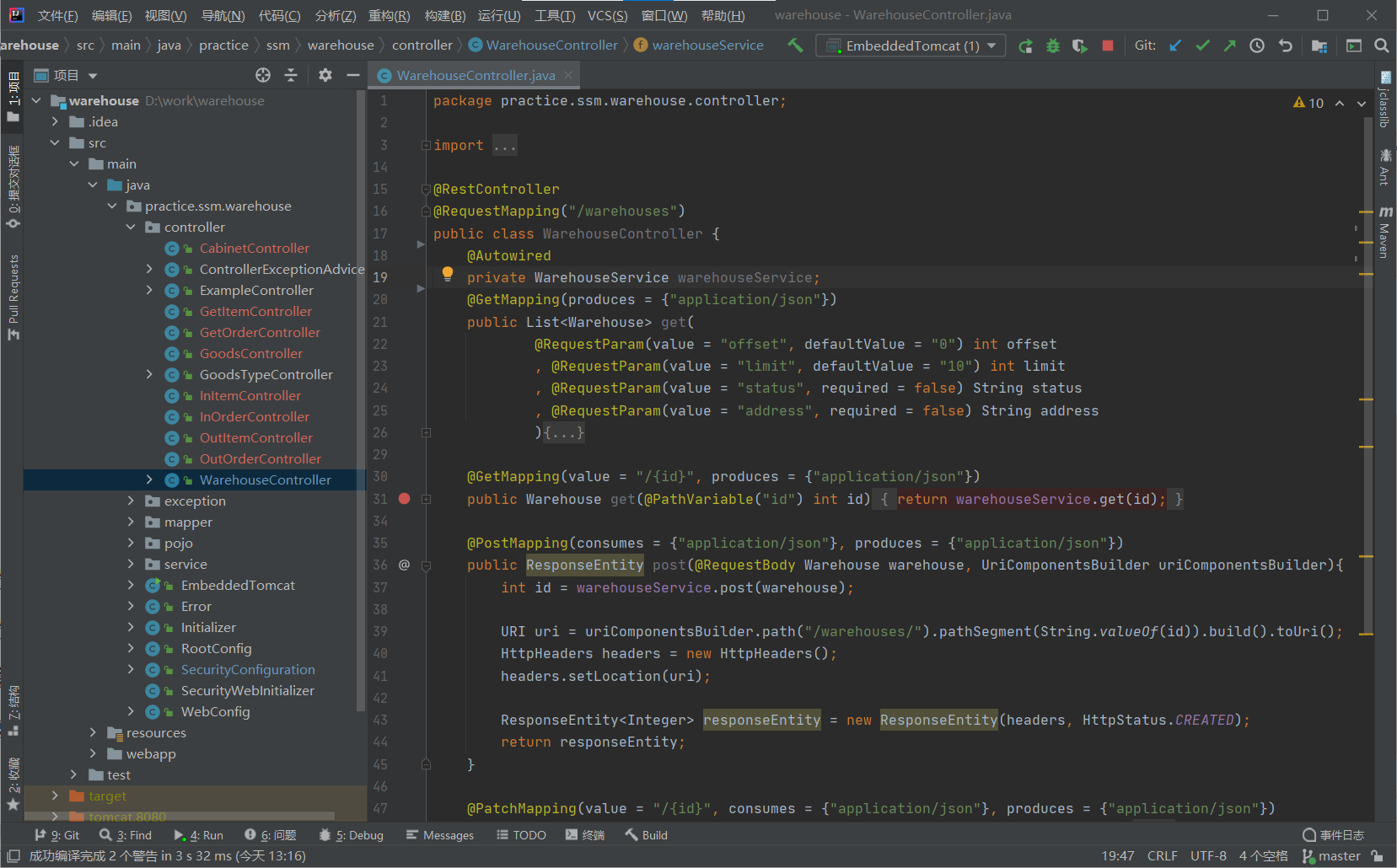


图 15 Controller 层实现

总结与体会

通过本次专业综合扩展，结合软件工程“生命周期方法学”与“面向对象方法学”，设计了详细设计阶段的面向对象的模型，深入理解了“面向对象分析”，“面向对象设计”与“面向对象实现”。面向对象方法学三个阶段之间的界限远比想象中模糊得多。尤其是“面向对象设计”和“面向对象实现”，基本上完成了“面向对象设计”，就可以找本宣科完成“面向对象实现”。