****

**上海交通大学本科毕业论文**

**基于电子芯片行业的ERP系统的设计与实现**

**学 生： 张二斤**

**学 号：719101010086**

**专 业：计算机科学与技术**

**导 师：李发致**

**学校代码：10248**

**上海交通大学继续教育学院**

**二Ｏ二一 年 二 月**

**毕业论文声明**

本人郑重声明：

1、此毕业论文是本人在指导教师指导下独立进行研究取得的成果。除了特别加以标注和致谢的地方外，本文不包含其他人或其它机构已经发表或撰写过的研究成果。对本文研究做出重要贡献的个人与集体均已在文中作了明确标明。本人完全意识到本声明的法律结果由本人承担。

2、本人完全了解学校、学院有关保留、使用学位论文的规定，同意学校与学院保留并向国家有关部门或机构送交此论文的复印件和电子版，允许此文被查阅和借阅。本人授权上海交通大学继续教育学院可以将此文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本文。

3、若在上海交通大学继续教育学院毕业论文审查小组复审中，发现本文有抄袭，一切后果均由本人承担（包括接受毕业论文成绩不及格、缴纳毕业论文重新写作费、重新写作毕业论文、不能按时获得毕业证书等），与毕业论文指导老师无关。

**作者签名**： 张二斤 **日期**： 2021年02月25日

基于电子芯片行业的ERP系统的设计与实现

摘 要

本系统专注于IC集成电路设计行业信息化整体解决方案，IC集成电路设计行业的物资供应管理通常是非常复杂的，生产委外进度、交期无法及时掌握，发料收货频繁，与上游委外工厂数据对接难，造成信息不对称不及时，严重影响了工作效率。

系统利用先进的信息化技术，帮助用户通过信息化的手段使得业务数据的一致性和准确性，实现数据的标准化和规范化，使用先进的数据对接模式，实现与委外工厂的数据同步，可实时掌握外协厂商的的库存情况，使得所有工作都透明化、数字化，通过系统实时库存和销售参考价，各销售员的业绩一目了然。

应用开发软件Microsoft Visual Studio 2019和数据库管理软件Microsoft SQL Server 2017， 以及一系列网站设计软件，使得开发更加的得心应手。本系统的开发基于仓储模式以及MVC，使得整个系统层次更加分明，展示层、业务层面和数据层面完全分割，代码逻辑十分清晰，利用依赖注入，减少各层之间的相互引用，更加利于后期业务扩展及维护。

关键词：仓储模式，MVC，数据库，ERP进销存系统，IC集成电路设计行业

**DESIGN AND IMPLEMENTATION OF ERP SYSTEM BASED ON ELECTRONIC CHIP INDUSTRY**

**ABSTRACT**

This system focuses on the overall solution of information technology in IC design industry. The material supply management of IC design industry is usually very complex. The schedule and delivery date of production outsourcing can not be mastered in time, the materials are issued and received frequently, and it is difficult to connect with the upstream outsourcing factory data, resulting in information asymmetry and not in time, which seriously affects the work efficiency.

The system uses advanced information technology to help users achieve the consistency and accuracy of business data by means of information technology, realize the standardization and normalization of data, use the advanced data docking mode, realize the data synchronization with the outsourcing factory, master the inventory situation of outsourcing manufacturers in real time, and make all work transparent and digital Deposit and sales reference price, the performance of the salesperson at a glance.

Application development software Microsoft Visual Studio 2019 and database management software Microsoft SQL Server 2017, as well as a series of website design software, make the development more handy. The development of the system is based on the warehousing mode and MVC, which makes the whole system more clear-cut, the display layer, business layer and data layer are completely separated, the code logic is very clear, and the use of dependency injection reduces the mutual reference between the layers, which is more conducive to the later business expansion and maintenance.

**Key word：**Storage Mode, MVC, Datebase, ERP Invoicing System, IC Design Industry

目录

[第一章 绪论 8](#_Toc67227859)

[1.1 系统开发背景 8](#_Toc67227860)

[1.2 系统开发的目的和意义 8](#_Toc67227861)

[1.3 可行性分析 9](#_Toc67227862)

[1.4 开发工具简介 9](#_Toc67227863)

[1.5 数据库简介 9](#_Toc67227864)

[第二章 系统需求分析 10](#_Toc67227865)

[2.1 主要流程分析 10](#_Toc67227866)

[2.2 系统主要需求分析 10](#_Toc67227867)

[2.2.1 登录 10](#_Toc67227868)

[2.2.2 物料管理 10](#_Toc67227869)

[2.2.3 采购管理 11](#_Toc67227870)

[2.2.4 生产管理 11](#_Toc67227871)

[2.2.5 仓库管理 11](#_Toc67227872)

[2.2.6 销售管理 12](#_Toc67227873)

[2.2.7 系统管理 12](#_Toc67227874)

[第三章 系统设计 13](#_Toc67227875)

[3.1 系统功能模块分析 13](#_Toc67227876)

[3.1.1 登录模块 13](#_Toc67227877)

[3.1.2 物料管理模块 14](#_Toc67227878)

[3.1.3 采购管理模块 14](#_Toc67227879)

[3.1.4 生产管理模块 15](#_Toc67227880)

[3.1.5 仓库管理模块 16](#_Toc67227881)

[3.1.6 销售管理模块 17](#_Toc67227882)

[3.1.7 系统管理模块 17](#_Toc67227883)

[3.2 数据库设计 18](#_Toc67227884)

[3.2.1 主要E-R图 18](#_Toc67227885)

[3.2.2 关键实体属性图 21](#_Toc67227886)

[第四章 系统实现 26](#_Toc67227887)

[4.1 用户登录及退出 26](#_Toc67227888)

[4.2 物料管理 27](#_Toc67227889)

[4.3 采购管理 29](#_Toc67227890)

[4.3.1 晶圆报价单管理 29](#_Toc67227891)

[4.3.2 晶圆订单管理 30](#_Toc67227892)

[4.3.3 采购订单管理 31](#_Toc67227893)

[4.4 生产管理 32](#_Toc67227894)

[4.4.1 晶圆测试订单管理 32](#_Toc67227895)

[4.4.2 划片订单管理 33](#_Toc67227896)

[4.4.3 封装订单管理 34](#_Toc67227897)

[4.4.4 芯片测试订单管理 35](#_Toc67227898)

[4.5 仓库管理 36](#_Toc67227899)

[4.5.1 收货通知单管理 36](#_Toc67227900)

[4.5.2 发料通知单管理 37](#_Toc67227901)

[4.5.3 入库单管理 38](#_Toc67227902)

[4.5.4 出库单管理 39](#_Toc67227903)

[4.5.5 库存查询 40](#_Toc67227904)

[4.5.6 仓库设置管理 40](#_Toc67227905)

[4.6 销售管理 40](#_Toc67227906)

[4.6.1 客户管理 40](#_Toc67227907)

[4.6.2 销售报价单管理 42](#_Toc67227908)

[4.6.3 销售单管理 44](#_Toc67227909)

[4.6.4 送样需求单管理 45](#_Toc67227910)

[4.7 系统管理 46](#_Toc67227911)

[4.7.1 公司管理 46](#_Toc67227912)

[4.7.2 部门管理 47](#_Toc67227913)

[4.7.3 职员管理 48](#_Toc67227914)

[4.7.4 系统用户管理 49](#_Toc67227915)

[第五章 系统测试 50](#_Toc67227916)

[5.1 白盒测试法 50](#_Toc67227917)

[5.2 黑盒测试法 51](#_Toc67227918)

[第六章 结论 53](#_Toc67227919)

[参考文献 54](#_Toc67227920)

[致谢 55](#_Toc67227921)

# 第一章 绪论

## 1.1 系统开发背景

随着经济的发展，ERP系统已成为现代企业管理的重要手段。但是针对于电子芯片设计行业来说，电子芯片设计行业的复杂程序远超寻常行业，使之库存记录、销售的数据难以记录和统计，使得数据杂乱无章。

产成品料号不确定、产品更新换代速度快，Lot批次片号、芯片批号追溯难，生产全部外协，交期无法及时掌握，不方便统一记录和管理，为此迫切研发一套针对IC设计行业的ERP系统。

## 1.2 系统开发的目的和意义

为了解决不断增加的库存、销售数据的规范化、合理化、标准化、科学化的高要求。

本系统的主要特点包括：

1. 通过信息化的手段和对接模式，实现与委外工厂的数据同步，可实时掌握外协厂商的生产进度以及库存状况。
2. 代理商通过平台实时在线协同作业，提高代理商的管理水平以及代理商粘性，提高整体运营效率。
3. 形成数据闭环，及时了解生产进度及在制品所在仓库和状态。
4. 委外订单、物流、资金流全程可视化跟踪。
5. 划片损耗、封装不良品等实际成本的核算。
6. 交期管理、交期提醒。
7. 订单与OA审批流紧密集成。
8. 强大的自定义字段功能，使得订单更加详细。

## 1.3 可行性分析

从20世纪50年代中期以来随着计算机的出现 投入使用，信息化方面获得了巨大的突破，这对于ERP所采用的方法产生了深远的影响。60年代中期，美国IBM公司的管理专家约瑟夫·奥利弗博士首先提出独立需求和相关需求的概念。

先进的管理理念一出现便立刻与先进的信息技术相结合，大约在1960年，计算机首次在库存管理获得了应用。在后面的ERP的形成发展过程中，先进的管理理念和先进的信息技术一直相互促进、相互发展直至今日。

## 1.4 开发工具简介

Microsoft Visual Studio 2019（简称VS 2019）是美国微软公司的开发工具包系列产品。VS 2019是一个基本完整的开发工具集，它包括了整个软件生命周期中所需要的大部分工具，如UML工具、代码管控工具、集成开发环境（IDE）等等。所写的目标代码适用于微软支持的所有平台。

## 1.5 数据库简介

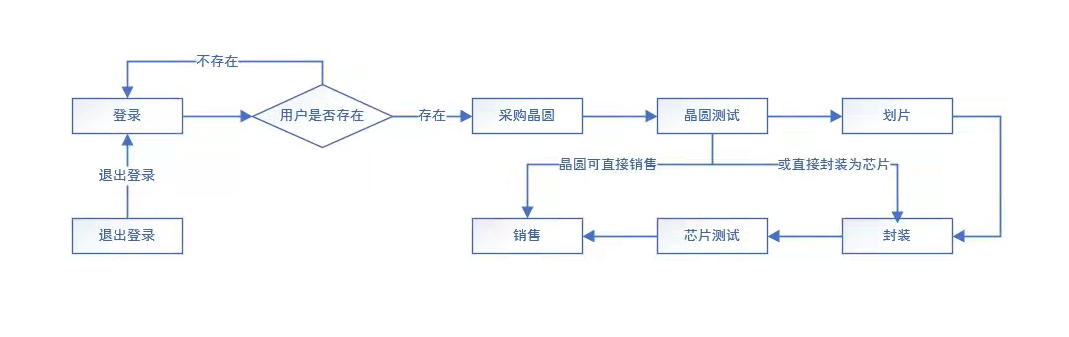
SQL Server是一个全面的数据库平台，使用集成的商业智能工具提供了企业级的数据管理。SQL Server数据库引擎为关系型数据和结构化数据提供了更安全可靠的存储功能，使您可以构建和管理用于业务的高可用和高性能的数据应用程序。SQL Server不仅可以有效地执行大规模联机事务处理，而且可以完成数据仓库和管理系统等许多具有挑战性的工作。具有使用方便可伸缩性好与相关软件集成高等优点。

# 第二章 系统需求分析

## 2.1 主要流程分析

系统登录后方可进行操作，登录判断用户是否存在，存在即可登陆成功。

系统流程为采购晶圆，然后将采购的晶圆进行晶圆测试，测试后的晶圆可进行划片、封装或直接售卖，晶圆或划片后的角片进行封装，封装后的芯片进行销售（如图2.1所示）。



**图2.1 系统主要流程**

## 2.2 系统主要需求分析

### 2.2.1 登录

该系统完全属于内部系统，则必须对系统的数据具有隐秘性和不可公开性。也为了追溯操作者，所以需要登录，且系统用户保持唯一性，一个系统用户只可对应一个职员。

### 2.2.2 物料管理

系统中便于用户选择和确保系统的一致性、正确性，所采用物料管理。

为符合IC芯片设计行业需要，物料中晶圆可按片或者颗进行售卖，则对物料中的单位进行自定义设置，更加灵活多变。为符合多种类型芯片的不同用处，包装方式及封装形式也进行自定义设置。

为方便管理各种物料的类型、组别、产品种类，进行物料组和产品种类的管理。

### 2.2.3 采购管理

企业管理中经常需要采购物料，常规来说，比如办公所用的电脑、文具及其他所需要的物品。而针对IC芯片设计行业来说，最常规的属于采购晶圆物料。

采购晶圆时由于供应商的数量以及供应商提供的工程种类不同，且分别对应不同的价格，为了便于用户选择未过期的报价单，所设立供应商报价单。

采购的物料进行芯片生产或办公使用，则需记录物料的详细信息及来源，如物料出现问题则需按照对应订单的供应商来进行退货，则进行采购订单的管理。

### 2.2.4 生产管理

采购而来的晶圆经过一系列的加工最终成为芯片，生产类型分为晶圆测试、划片、封装、芯片测试，生产的流程依次为晶圆测试、划片、封装、芯片测试。

采购而来的晶圆需经过测试，确定其良率、可用品以及Good Die数量。

划片是将客户所需的晶圆划成为不同规格的角片，以便于对于需求量不大的用户进行销售。

封装是将一个晶圆或角片制作成为芯片，根据供应商返回的进行将芯片入库存储，用于售卖。根据封装所发货的晶圆或角片大致计算出能封装回货的芯片颗数，预判可以销售的芯片数量，以便于对客户所需要的芯片的预定。

芯片测试是将封装完成的芯片进行测试，以防对客户发出的芯片有质量问题。

所进行的生产为预防质量或其他问题，需要对每批货物进行追溯，能够明确物料的来源以及去向。

### 2.2.5 仓库管理

对于生产中的发原料以及收货，为了数据的正确性和完整性，涉及收货和发货，则涉及库存，需要经过多次的检查，以免发错或缺漏。

库存则需要记录当前物料的库存及库存金额，便于数据统计库存量，不可过多，也不可过少。

有时候需要进行仓库调拨或者当仓库中的物料自然挥发时，需要进行库存的调整，用于仓库数据的完整性。

### 2.2.6 销售管理

生产后的芯片或晶圆向客户进行销售服务，记录客户信息，可按照物料的不同单位进行销售，便于向客户进行收款动作。

创建销售订单时，为了方便录入订单信息，做客户管理，也便于对客户进行回访，客户为了方便数据统计和用户类型的销售额的差别，客户需具备不同的类别、行业、标签、区域等的属性。

也会有赠送、试用产品给客户，也需进行记录跟踪，需要获取送样的回馈，以便进行产品的改善。

### 2.2.7 系统管理

为方便集团的子公司或者其他地区的办事处的工作衔接和配合，需要支持多公司的集团化应用，公司下需包含多个部门，部门下需包含多个职员，且职员唯一。

每一个用户对应一个职员，职员的职位不同，分工不同，建立角色权限模块。一个角色可对应多个用户，一个用户也可对应多个角色。职员公司不同，所可查看的仓库不同，则也需要设计仓库权限。

为确保生产订单的创建便捷性和唯一性，订单的编号可自动生成。

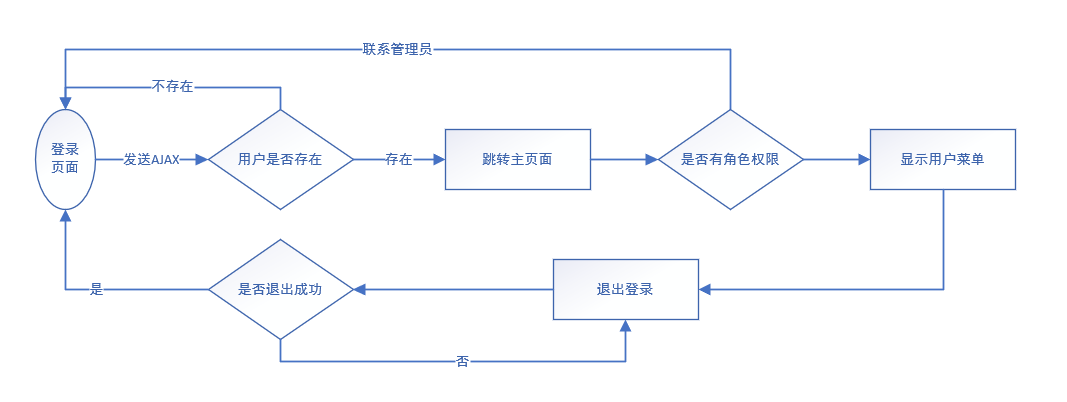
为追溯操作异常，或进行错误处理，则需设计审计日志，包括登录日志、操作日志、账密更改日志。

# 第三章 系统设计

## 3.1 系统功能模块分析

本系统主要包括登录模块、物料管理模块、采购管理模块、生产管理模块、仓库管理模块、销售管理模块、系统管理模块。

### 3.1.1 登录模块

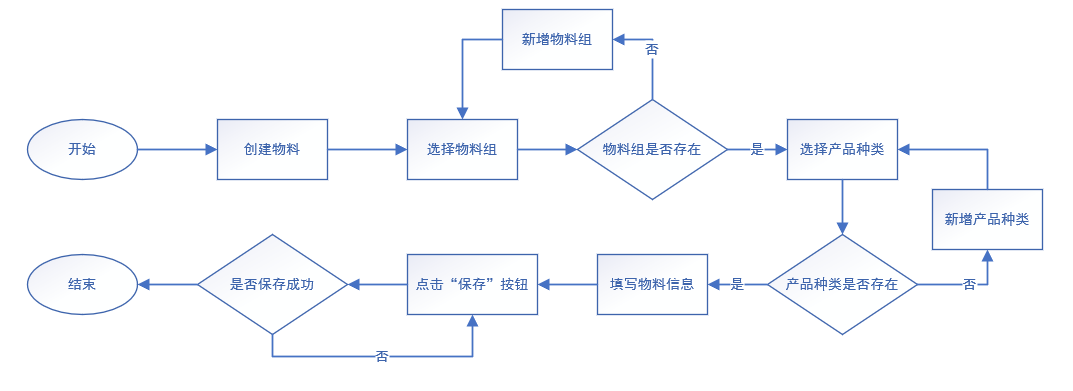


**图3.1 登录流程图**

登录大致流程如图3.1所示，详细步骤如下：

1. 判断用户输入的用户名和密码是否为空，然后使用JavaScript获取用户输入的用户名及密码，当用户点击“登录”按钮或敲击键盘回车时利用AJAX进行接口访问。
2. 后端程序进行匹配数据库，后台通过Eentity Framework进行数据库匹配，如果用户名和加密转换后的密码输入正确，并正确查询到数据，则记录当前登录的用户信息。
3. 跳转主页面后，获取当前登录用户的唯一标识和权限标识，通过JSON格式返回数据，根据获取的权限标识，显示该用户所拥有的菜单列表。
4. 点击“退出登录”按钮时，清除程序中所记录的当前登录用户信息，并返回登录页面。

### 3.1.2 物料管理模块



**图3.2 创建物料流程图**

创建物料的流程如图3.2所示，详细步骤如下：

1. 判断用户输入的物料描述、规格型号等必要信息是否为空，然后使用JavaScript获取所输入的所有信息，利用AJAX进行接口访问，后端程序判断用户是否输入物料编码，输入时则判断该编码是否重复；不输入时，匹配对应的生成规则，进行生成。
2. 然后将数据保存到数据库中。通过JSON格式返回数据，提示用户操作成功，并返回物料列表页面，使用AJAX进行刷新整个列表。
3. 如遇保存失败问题，则提示用户具体的错误信息，重新输入并提交。

物料列表的每行数据都会有删除、详情按钮，删除时通过AJAX访问接口进行删除；点击详情，则进行物料详情页面，通过MVC特有的参数接收和传递，编辑需要更改的信息，点击保存，即可编辑该物料信息。

### 3.1.3 采购管理模块



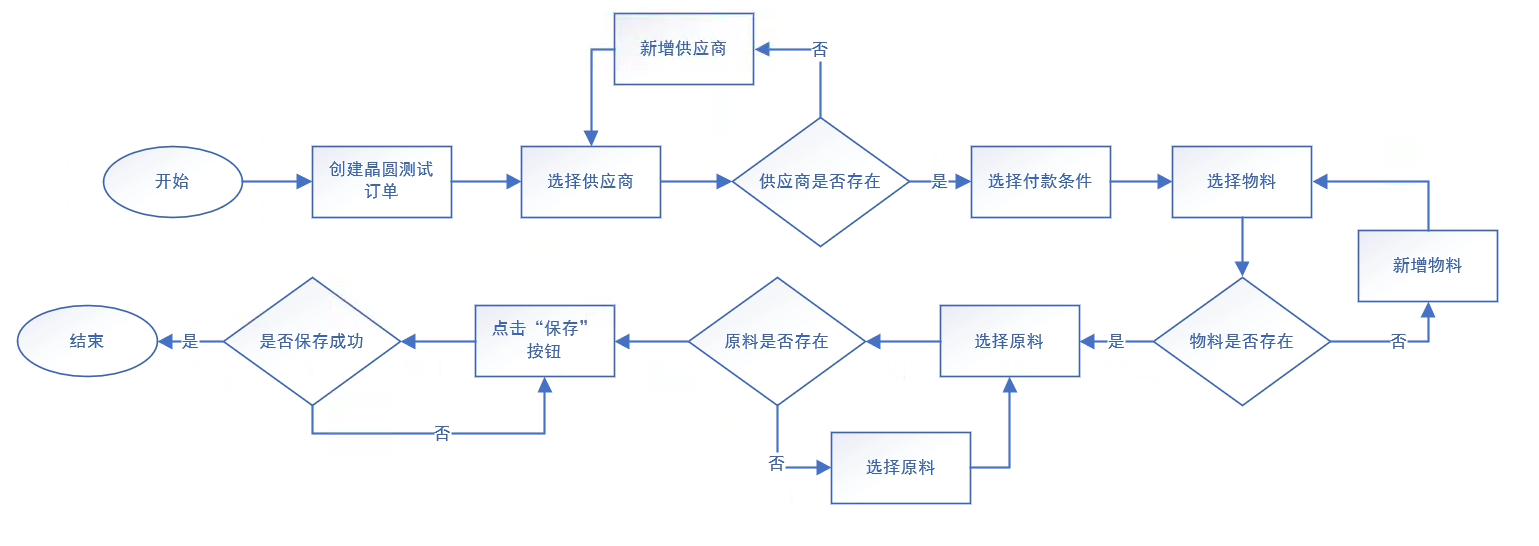
**图3.3 创建采购晶圆订单流程图**

创建采购晶圆订单的大致流程如图3.3所示，详细步骤如下：

1. 判断用户输入的必要信息是否为空，获取所输入的所有信息，进行接口访问，后端程序判断用户是否输入订单编号，输入时则判断该编码是否重复；不输入时，匹配对应的生成规则，进行生成。
2. 将输入的数据保存在数据库中，通过JSON格式返回数据，提示用户操作成功，并返回常规物料订单列表页面，使用AJAX进行刷新整个列表。
3. 如遇保存失败问题，则提示用户具体的错误信息，重新输入并提交。

常规采购订单、晶圆采购订单的每行数据都存在删除、详情按钮，通过AJAX访问接口进行删除；点击详情，则进行订单详情页面，编辑需要更改的信息，点击保存，即可编辑该订单信息。

### 3.1.4 生产管理模块



**图3.4 创建晶圆测试订单流程图**

创建晶圆测试订单大致流程如图3.4所示，详细步骤如下：

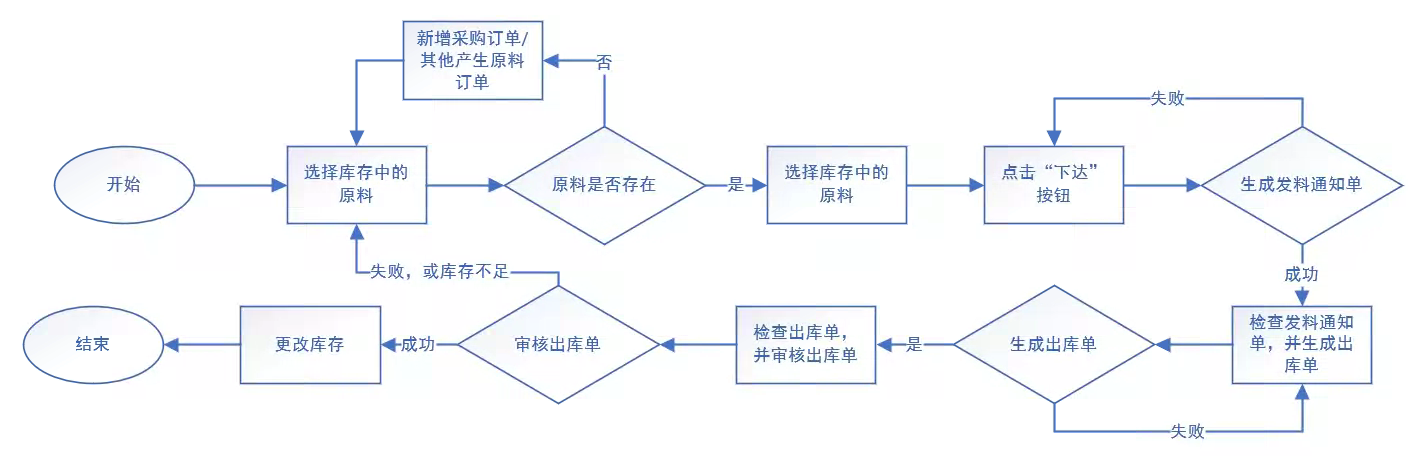
1. 判断用户输入的必要信息是否为空，获取所输入的所有信息，进行接口访问，后端程序判断用户是否输入订单编号 ，输入时则判断该编码是否重复；不输入时，匹配对应的生成规则，进行生成。
2. 将输入的数据保存到数据库中，通过JSON格式返回数据，提示用户操作成功，并返回晶圆测试订单列表页面，使用AJAX进行刷新整个列表。
3. 如遇保存失败问题，则提示用户具体的错误信息，重新输入并提交。

创建划片订单、封装订单、芯片测试订单，则同晶圆测试订单操作基本相同。

晶圆测试订单、划片订单、封装订单、芯片测试订单每行都会有删除、详情按钮，删除时通过AJAX访问接口进行删除；点击详情，则进行订单详情页面，编辑需要更改的信息，点击保存，即可编辑该订单信息。

订单中发料、收货将在章节3.1.5中讲述，发料、收货请查看章节3.1.5。

### 仓库管理模块

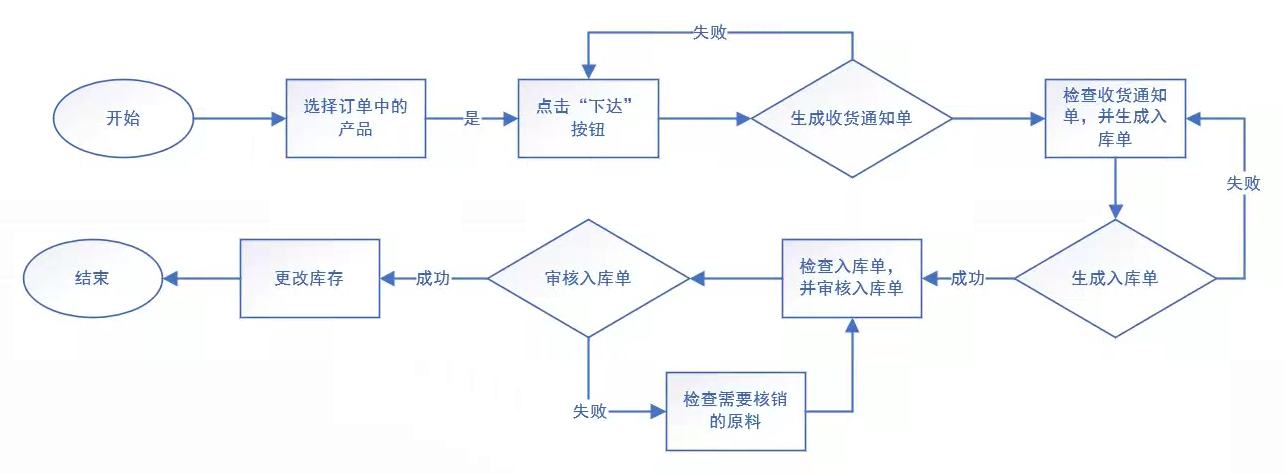


**图3.5 发料流程图**

进行生产订单时，会有出库或者入库的业务产生。

发料大致流程如图3.5所示，详细步骤如下：

1. 晶圆测试详情页面中，使用发货功能进行出库业务，进行下达后则生成发料通知单。
2. 检查发料通知单，数据无误，则生成出库单。
3. 检查出库单，数据无误后审核出库单，进行库存的更改。



**图3.6 收货流程图**

入库大致流程如图3.6所示，详细步骤如下：

1. 晶圆测试详情页面中，使用收货功能进行入库业务，进行下达后生成入库通知单。
2. 检查收货通知单，数据无误，则生成入库单。
3. 检查入库单，数据无误后审核入库单，进行库存的更改。

发料、收货的动作均会在库存中体现，可根据输入的条件进行库存的查询。

划片订单、封装订单、芯片测试订单的发料、收货流程同晶圆测试订单大致相同，可参考图3.5和图3.6,。

### 3.1.6 销售管理模块



**图3.7 创建销售订单流程**

创建销售订单大致流程如图3.7所示，详细步骤如下：

1. 判断用户输入的必要信息是否为空，获取所输入的所有信息，进行接口访问，后端程序判断用户是否输入客户编码 ，输入时则判断该编码是否重复；不输入时，匹配对应的生成规则，进行生成。
2. 将输入的数据保存到数据库中，通过JSON格式返回数据，提示用户操作成功，并返回客户列表页面，使用AJAX进行刷新整个列表。
3. 如遇保存失败问题，则提示用户具体的错误信息，重新输入并提交。

销售报价单、送样需求订单，则同新增销售操作基本相同。

新增客户、销售报价单、销售订单、送样需求订单每行都会有删除、详情按钮，删除时通过AJAX访问接口进行删除；点击详情，则进行客户或订单详情页面，编辑需要更改的信息，点击保存，即可编辑该客户或订单信息。

销售订单、送样需求订单中发料流程请参考3.1.5中的发料流程。

### 3.1.7 系统管理模块

新增公司详细步骤如下：

1. 判断用户输入的必要信息是否为空，获取所输入的所有信息，进行接口访问，后端程序判断用户是否输入公司编码 ，输入时则判断该编码是否重复；不输入时，匹配对应的生成规则，进行生成。
2. 将输入的信息保存到数据库中，通过JSON格式返回数据，提示用户操作成功，并返回公司列表页面，使用AJAX进行刷新整个列表。
3. 如遇保存失败问题，则提示用户具体的错误信息，重新输入并提交。

新增部门、职员、角色，则同公司操作基本相同。

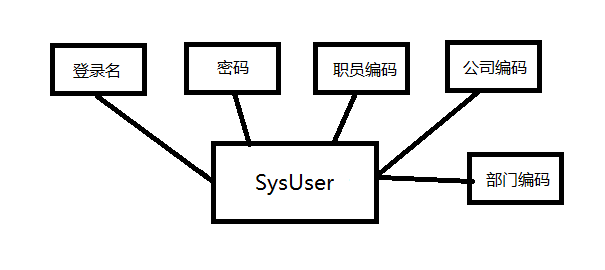
新增系统用户时，需选择所属公司、所属部门、对应职员和仓库权限，如未选择，则进行提示，并不能保存。

配置角色权限时，选择该角色所拥有的菜单权限及按钮权限。

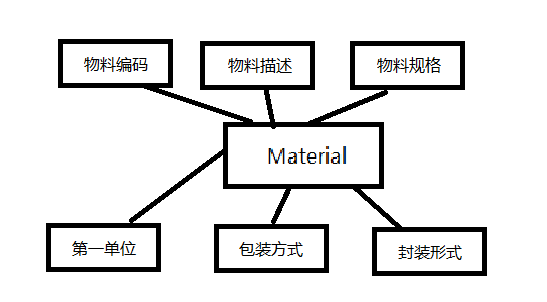
公司管理、部门管理、职员管理、系统用户管理、角色管理每行都会有删除、详情按钮，删除时通过AJAX访问接口进行删除；点击详情，则进行详情页面，编辑需要更改的信息，点击保存，即可编辑该数据信息。

## 3.2 数据库设计

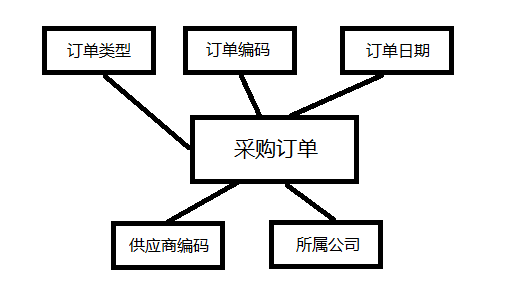
### 3.2.1 主要E-R图



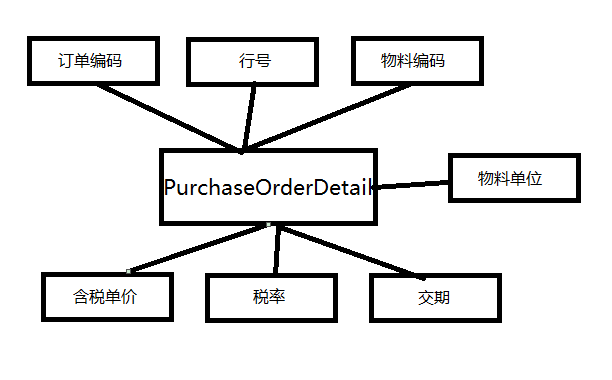
**图3.8 系统用户信息表E-R图**



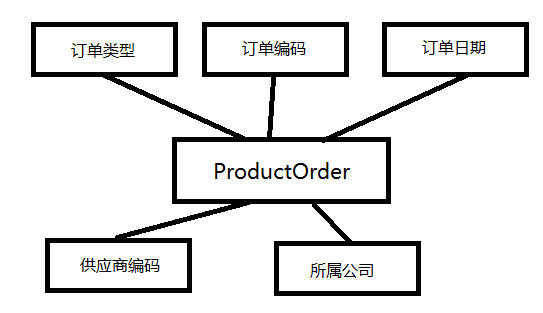
**图3.9物料信息表E-R图**



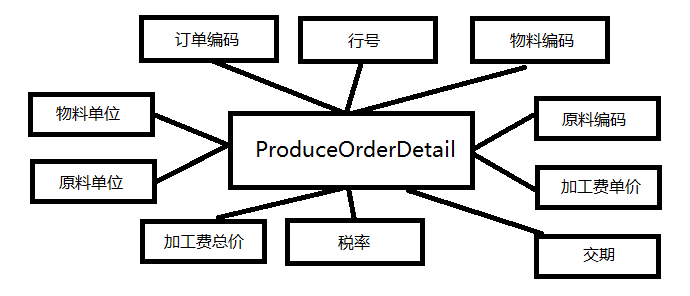
**图3.10采购订单信息表E-R图**



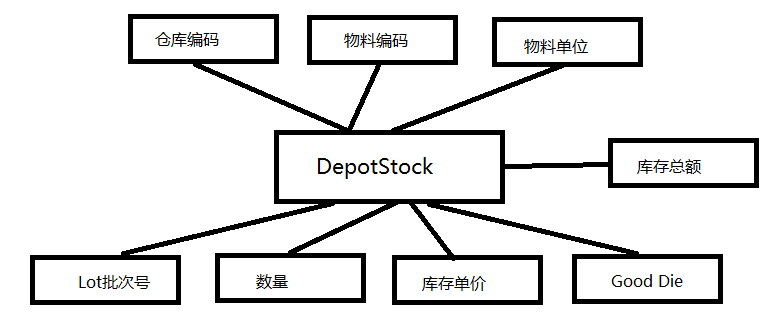
**图3.11 采购订单明细信息表E-R图**



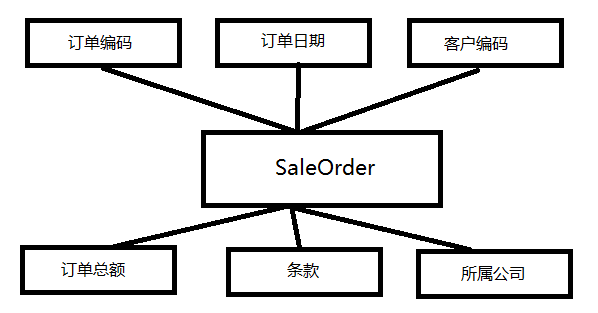
**图3.12 生产订单信息表E-R图**



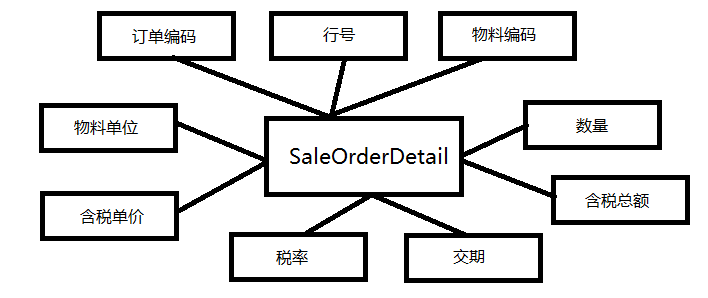
**图3.13 生产订单明细信息表E-R图**



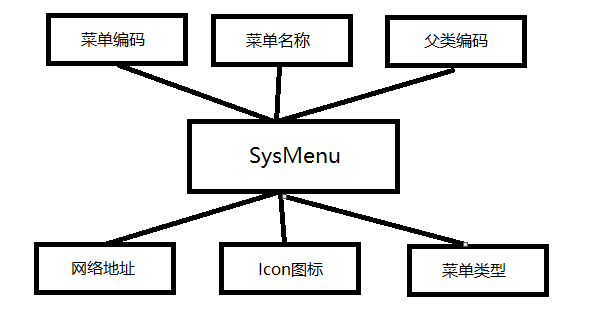
**图3.14 库存信息表E-R图**



**图3.15 销售订单信息表E-R图**



**图3.16 销售订单明细信息表E-R图**



**图3.17 系统菜单信息表E-R图**

### 3.2.2 关键实体属性图

以下是对应3.2.1节所述关键实体属性图。

**表3.1 系统用户信息表（SysUser）如图3.8所示**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **中文名称** | **英文代码** | **类型** | **长度** | **空** |
| 登录名 | LoginName | 字符型 | 50 | 否 |
| 密码 | Password | 字符型 | 100 | 否 |
| 职员编码 | StaffCode | 字符型 | 20 | 否 |
| 所属公司编码 | CompCode | 字符型 | 20 | 否 |
| 所属部门编码 | DeptCode | 字符型 | 20 | 否 |

**表3.2 物料信息表（Material）如图3.9所示**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **中文名称** | **英文代码** | **类型** | **长度** | **空** |
| 物料编码 | Code | 字符型 | 50 | 否 |
| 物料描述 | Name | 字符型 | 100 | 否 |
| 物料规格 | Model | 字符型 | 100 | 否 |
| 第一单位 | Unit | 字符型 | 20 | 否 |
| 包装方式 | Packing | 字符型 | 20 | 是 |
| 封装形式 | PackageForm | 字符型 | 20 | 是 |

**表3.3 采购订单信息表（PurchaseOrder）如图3.10所示**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **中文名称** | **英文代码** | **类型** | **长度** | **空** |
| 订单类型标识 | Flag | 整数型 |  | 否 |
| 订单编码 | Code | 字符型 | 100 | 否 |
| 订单日期 | OrderDate | 日期型 |  | 否 |
| 供应商编码 | SupplierCode | 字符型 | 20 | 否 |
| 所属公司 | CompCode | 字符型 | 50 | 否 |

**表3.4 采购订单明细信息表（PurchaseOrderDetail）如图3.11所示**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **中文名称** | **英文代码** | **类型** | **长度** | **空** |
| 订单编码 | Code | 字符型 | 100 | 否 |
| 行号 | LineNum | 整数型 |  | 否 |
| 物料编码 | MaterialCode | 字符型 | 50 | 否 |
| 物料单位 | MaterialUnit | 字符型 | 50 | 否 |
| 含税单价 | Price | 数字型 |  | 否 |
| 税率 | TaxRate | 字符型 | 50 | 否 |
| 交期 | DeliveryDate | 日期型 |  | 否 |

**表3.5 生产订单信息表（ProduceOrder）如图3.12所示**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **中文名称** | **英文代码** | **类型** | **长度** | **空** |
| 订单类型标识 | Flag | 整数型 |  | 否 |
| 订单编码 | Code | 字符型 | 100 | 否 |
| 订单日期 | OrderDate | 日期型 |  | 否 |
| 供应商编码 | SupplierCode | 字符型 | 20 | 否 |
| 所属公司 | CompCode | 字符型 | 50 | 否 |

**表3.6 生产订单明细信息表（ProduceOrderDetail）如图3.13所示**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **中文名称** | **英文代码** | **类型** | **长度** | **空** |
| 订单编码 | Code | 字符型 | 100 | 否 |
| 行号 | LineNum | 整数型 |  | 否 |
| 物料编码 | MaterialCode | 字符型 | 50 | 否 |
| 物料单位 | MaterialUnit | 字符型 | 50 | 否 |
| 原料编码 | LinkMaterialCode | 字符型 | 50 | 否 |
| 原料单位 | LinkMaterialUnit | 字符型 | 50 | 否 |
| 加工费含税单价 | Price | 数字型 |  | 否 |
| 加工费含税总价 | TotalProcessFee | 数字型 |  | 否 |
| 税率 | TaxRate | 字符型 | 50 | 否 |
| 交期 | DeliveryDate | 日期型 |  | 否 |

**表3.7 库存信息表（DepotStock）如图3.14所示**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **中文名称** | **英文代码** | **类型** | **长度** | **空** |
| 仓库编码 | DepotCode | 字符型 | 100 | 否 |
| 物料编码 | MaterialCode | 字符型 | 50 | 否 |
| 物料单位 | MaterialUnit | 字符型 | 50 | 否 |
| Lot批次号 | LotBatch | 字符型 | 50 | 否 |
| 数量 | Quantity | 整数型 |  | 否 |
| 库存单价 | GoodsPrice | 数字型 |  | 否 |

续表3.7

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **中文名称** | **英文代码** | **类型** | **长度** | **空** |
| Good Die | GoodDie | 整数型 |  | 否 |
| 库存总额 | GoodsAmount | 数字型 |  | 否 |

**表3.8 销售订单信息表（SaleOrder）如图3.15所示**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **中文名称** | **英文代码** | **类型** | **长度** | **空** |
| 订单编码 | Code | 字符型 | 100 | 否 |
| 订单日期 | OrderDate | 日期型 |  | 否 |
| 客户编码 | ClientCode | 字符型 | 20 | 否 |
| 所属公司 | CompCode | 字符型 | 50 | 否 |
| 订单总额 | TotalAmount | 数字型 |  | 否 |
| 条款 | TotalAmount | 字符型 | 8000 | 否 |

**表3.9 销售订单明细信息表（SaleOrderDetail）如图3.16所示**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **中文名称** | **英文代码** | **类型** | **长度** | **空** |
| 订单编码 | Code | 字符型 | 100 | 否 |
| 行号 | LineNum | 整数型 |  | 否 |
| 物料编码 | MaterialCode | 字符型 | 50 | 否 |
| 物料单位 | MaterialUnit | 字符型 | 50 | 否 |
| 数量 | Quantity | 整数型 |  | 否 |
| 含税单价 | Price | 数字型 |  | 否 |
| 含税总额 | TotalAmount | 数字型 |  | 否 |
| 税率 | TaxRate | 数字型 |  | 否 |
| 交期 | DeliveryDate | 日期型 |  | 否 |

**表3.10 系统菜单信息表（SysMenu）如图3.17所示**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **中文名称** | **英文代码** | **类型** | **长度** | **空** |
| 菜单编码 | Code | 字符型 | 100 | 否 |

续表3.10

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **中文名称** | **英文代码** | **类型** | **长度** | **空** |
| 菜单名称 | Name | 字符型 | 100 | 否 |
| 父类编码 | ParentCode | 字符型 | 100 | 否 |
| 网络地址 | Url | 字符型 | 100 | 否 |
| Icon图标 | IconClass | 字符型 | 100 | 否 |
| 菜单类型 | IsButton | 数字型 |  | 否 |

# 第四章 系统实现

## 4.1 用户登录及退出

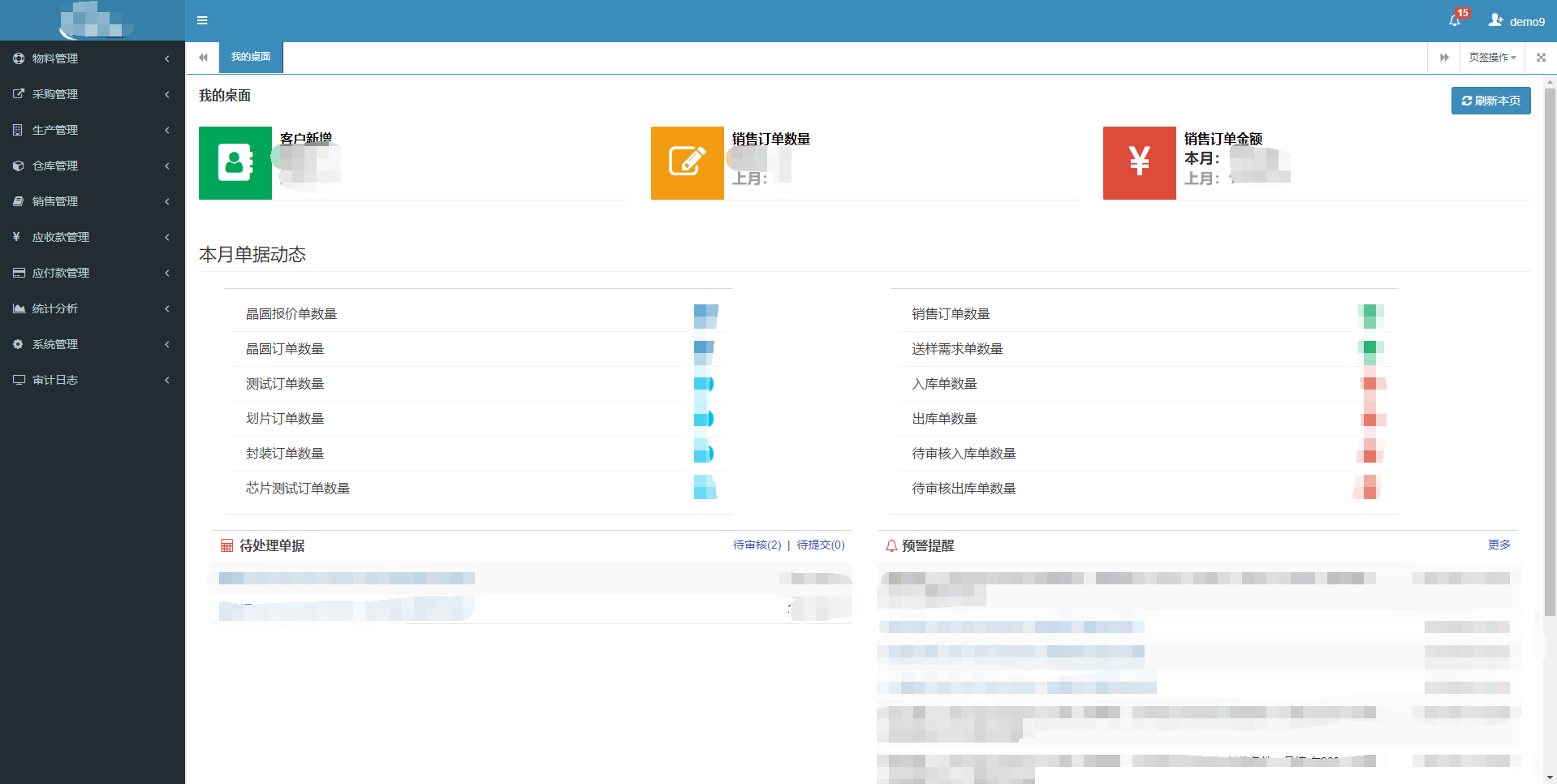


**图4.1 系统登录页面**



**图 4.2 用户登录实体类**

用户输入用户名以及密码，判断输入的用户名和密码是否为空，使用Trim方法去除用户名以及密码的前后空格，并使用AJAX Post类型访问接口，Post方式类型保密性更强，使得用户信息更加安全，登录页面如图4.1所示。后端使用实体类进行接收所传入的参数（实体类参数如图4.2所示），这样的做法使得程序看上去更加简洁、更加便于管理，如果有其他参数，则直接在本实体类进行增加即可。后端使用ORM框架中EF Core通过用户名和密码进行数据库匹配，匹配成功，则跳转主页面（如图4.3所示）；匹配不成功，则提示用户信息不正确或其他不确定的错误信息。



**图4.3 系统主页面**

## 4.2 物料管理



**图4.4 新增物料页面**

输入物料描述等必要条件，将用户输入的信息使用form表单提交（form表单所提交的内容如图4.4所示），form表单提交的好处是可以整个表单中的数据进行提交，而不用一个一个的去获取参数，简洁方便。

 **图 4.5 物料实体类**

用户输入的数据传递到接口，后端使用实体类进行接收（实体类参数如图4.5所示）。物料编码的生成规则由“物料编码前缀”+“序列号”生成，“序列号”是顺序的，由缓存Redis来完成，Redis具有高可用、读取速度快的特性，很适合用来做这样的功能。

生成后物料编码的数据实体类通过EF Core框架进行数据的保存。为方便追溯操作记录，此时也需要将当前登录用户的信息进行保存；保存失败时抛出异常提醒，便于用户及时反馈整理问题，并重新提交。

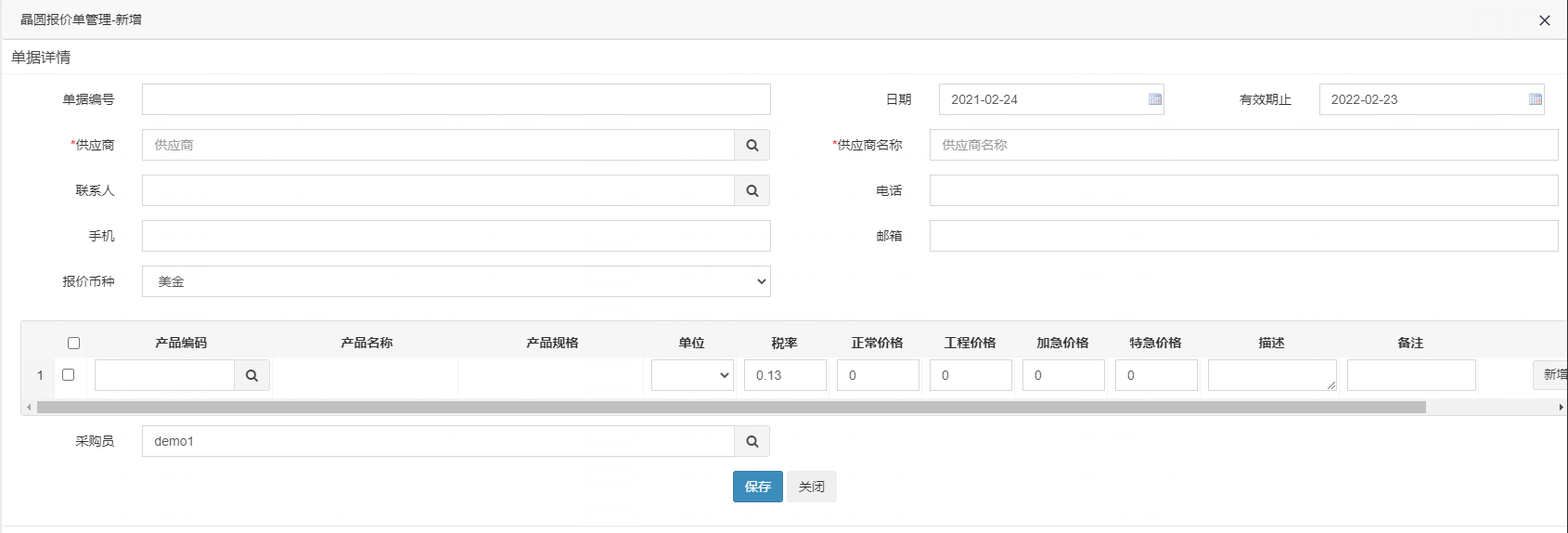
保存成功后便于用户良好的体验感，则返回物料列表页面（如图4.6所示）。



**图4.6 物料列表页面**

## 4.3 采购管理

### 4.3.1 晶圆报价单管理



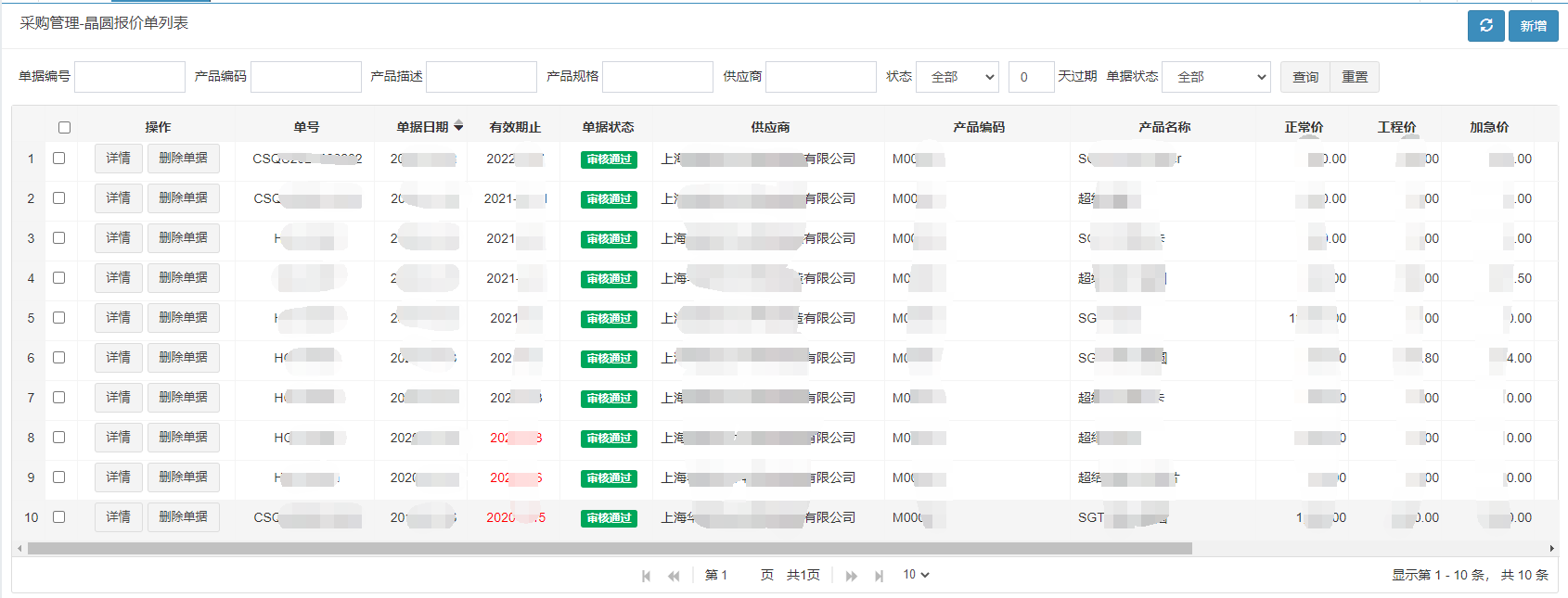
**图 4.7 新增晶圆报价单页面**

选择供应商等必要条件，将用户输入的信息使用AJAX传递到接口，因为具有明细列表，不可通过form表单来进行提交，所以要通过Jquery获取列表的内容形成JS 数组传递到后端进行处理。

后端使用数组类型的参数用来接收明细的内容，其余主要的信息使用实体类来接收，报价单编号在不输入的情况下采取按规则进行生成，报价单的规则由用户设置的“报价单编码”前缀+“日期”字符串+“序列号”生成，序列号同样由Redis缓存进行存储，并在每天的0点清除，确保每天的序列号由1开始。

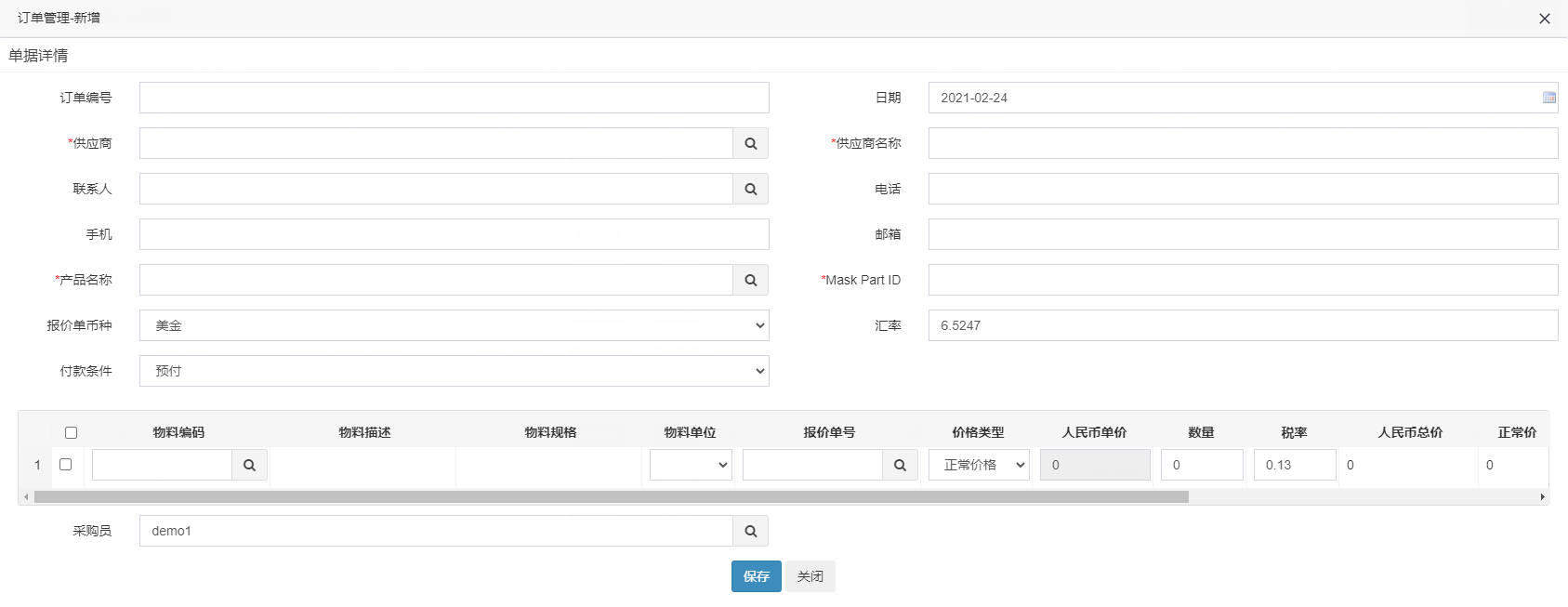
生成后的数据通过EF Core框架进行数据库保存，为方便追溯操作记录，此时也需要将当前登录用户的信息进行保存；保存失败时抛出异常提醒，便于用户及时反馈整理问题，并重新提交。

保存成功后便于用户良好的体验感，则返回晶圆报价单列表页面（如图4.8所示）。



**图4.8 晶圆报价单列表页面**

### 4.3.2 晶圆订单管理



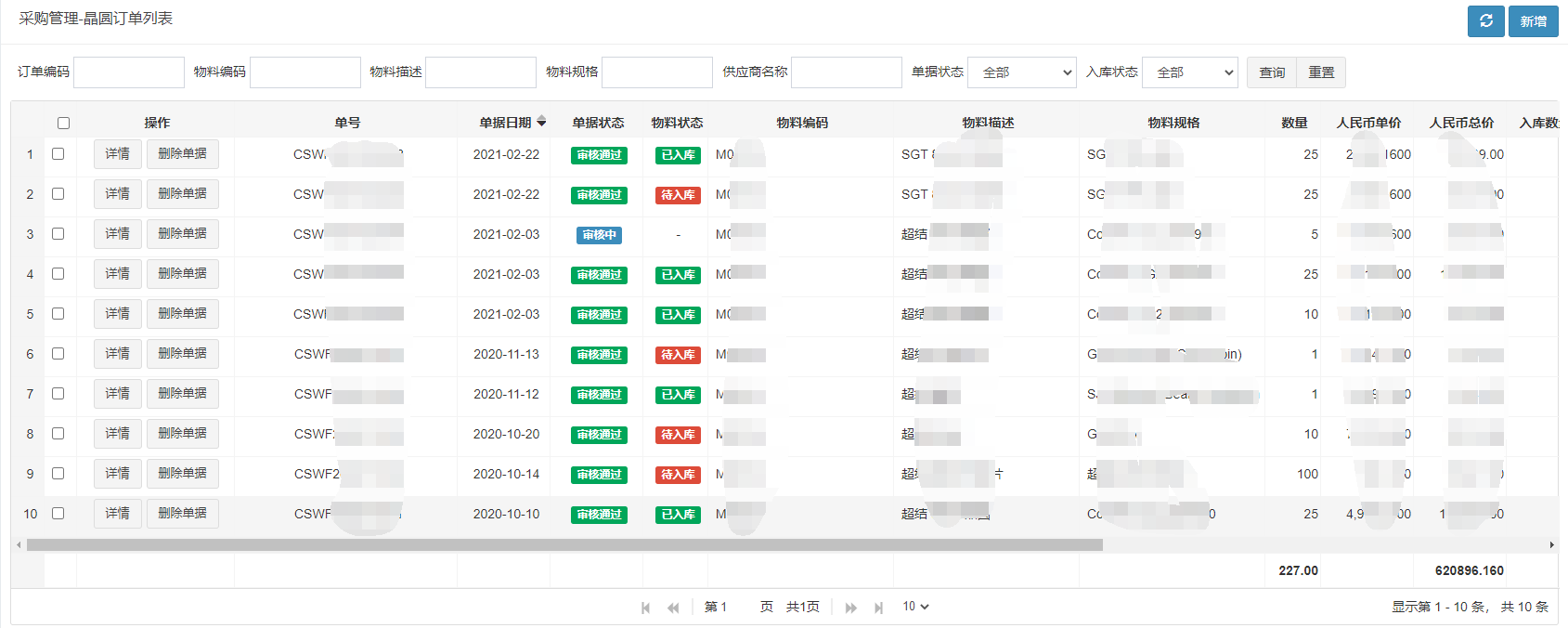
**图 4.9 新增晶圆订单页面**

选择供应商等必要条件，必要条件防止数据错乱、缺失，引起系统所不必要的BUG，将用户输入的明细数据获取成为数组，除明细外的主要信息，使用AJAX传递到后端实体类。

接口接收传过来的数据后，按照规则生成订单编号，订单编号由用户设置的“晶圆订单编码前缀” +“日期”字符串+“序列号”生成，序列号同样由Redis缓存进行存储，并在每天的0点清除，确保每天的序列号由1开始。

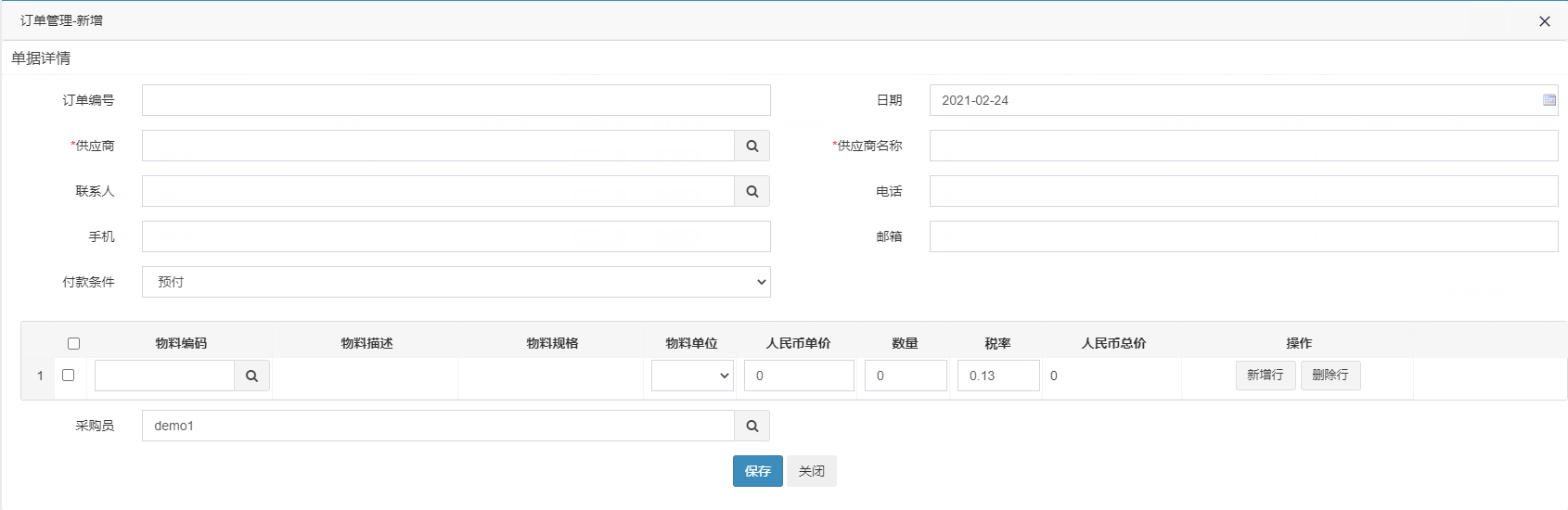
生成后的数据通过EF Core框架进行数据库保存，为方便追溯操作记录，此时也需要将当前登录用户的信息进行保存；保存失败时抛出异常提醒，便于用户及时反馈整理问题，并重新提交。

保存成功后便于用户良好的体验感，应关闭当前页面，返回晶圆订单列表页面（如图4.10所示），刷新列表，获取用户刚添加的数据。



**图 4.10 晶圆订单列表页面**

### 4.3.3 采购订单管理



**图 4.11 新增采购订单页面**

选择供应商等必要条件，用户输入的数据内容如图4.11所示。将用户输入的明细数据获取成为数组，除明细外的主要信息，使用AJAX传递到后端实体类。

接口接收传过来的数据后，按照规则生成订单编号，订单编号由用户设置的“采购订单编码前缀” +“日期”字符串+“序列号”生成，序列号同样由Redis缓存进行存储，并在每天的0点清除，确保每天的序列号由1开始。

生成后的数据通过EF Core框架进行数据库保存，为方便追溯操作记录，此时也需要将当前登录用户的信息进行保存；保存失败时抛出异常提醒，便于用户及时反馈整理问题，并重新提交。

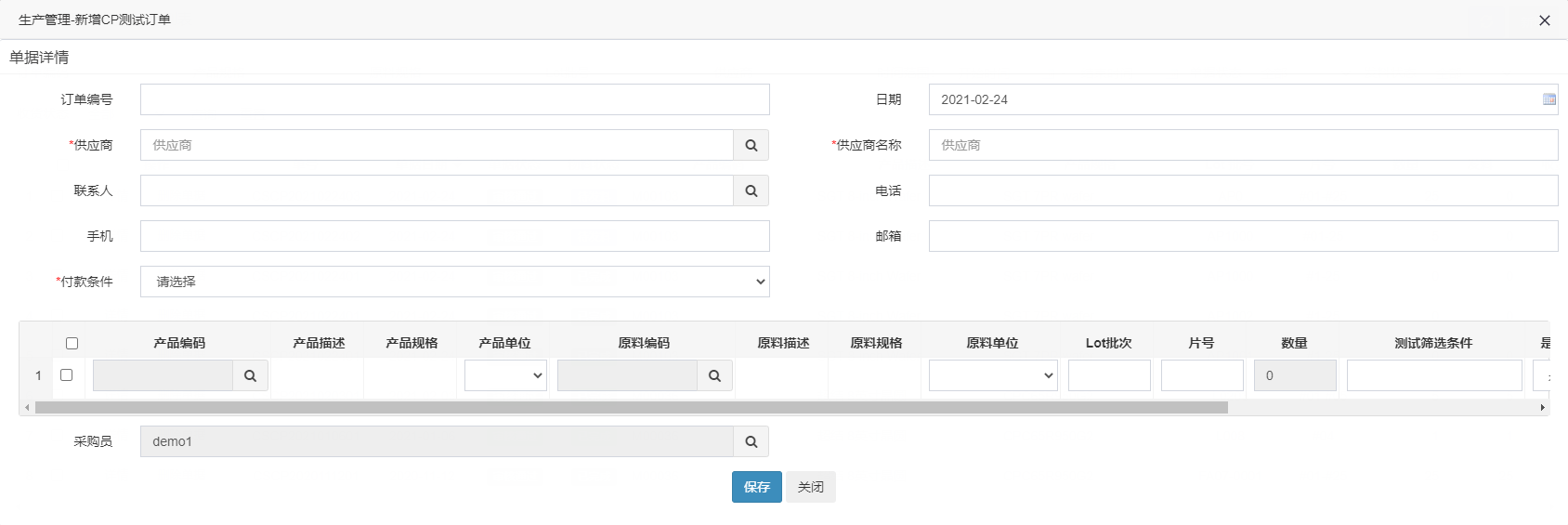
保存成功后便于用户良好的体验感，应关闭当前页面，返回采购订单列表页面（如图4.12所示），刷新列表，获取用户刚添加的数据。



**图4.12 采购订单列表页面**

## 4.4 生产管理

### 4.4.1 晶圆测试订单管理



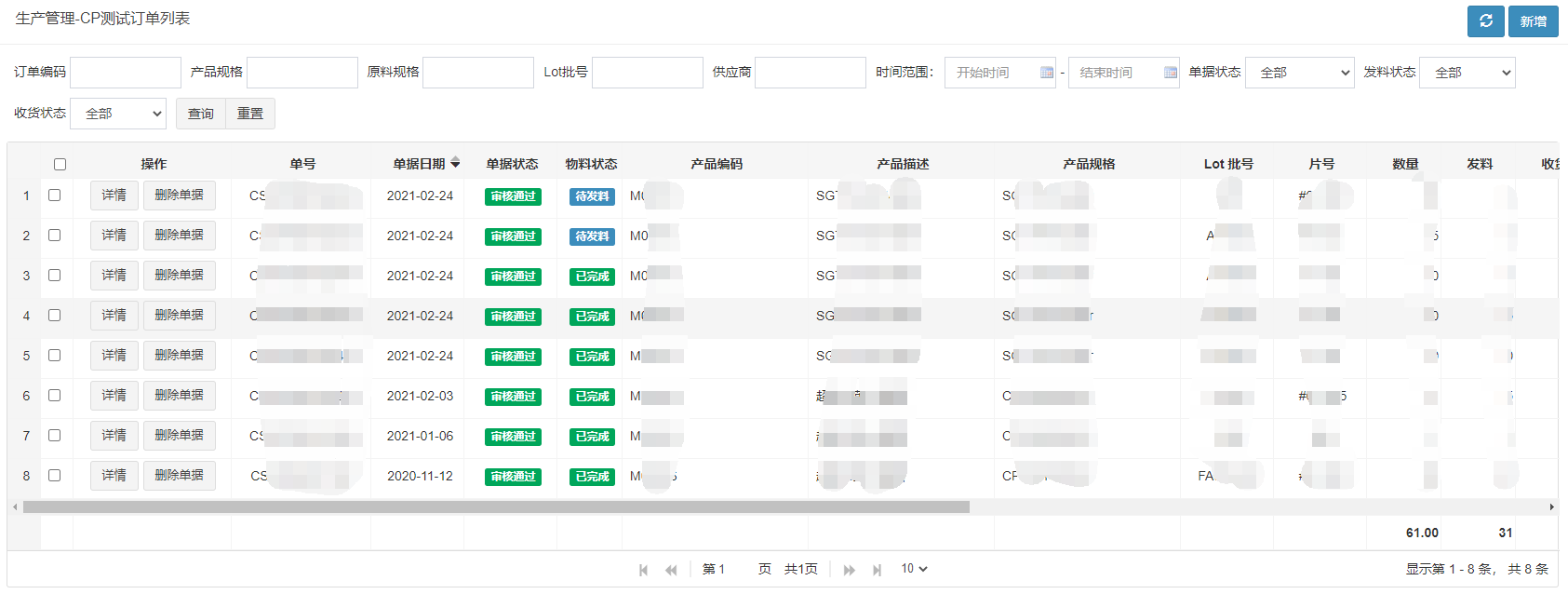
**图4.13 新增晶圆订单页面**

选择供应商等必要条件，用户输入的数据如图4.13所示，将用户输入的明细数据获取成为数组，除明细外的主要信息，使用AJAX传递到后端实体类。

接口接收传过来的数据后，按照规则生成订单编号，订单编号由用户设置的“晶圆订单编码前缀” +“日期”字符串+“序列号”生成，序列号同样由Redis缓存进行存储，并在每天的0点清除，确保每天的序列号由1开始。

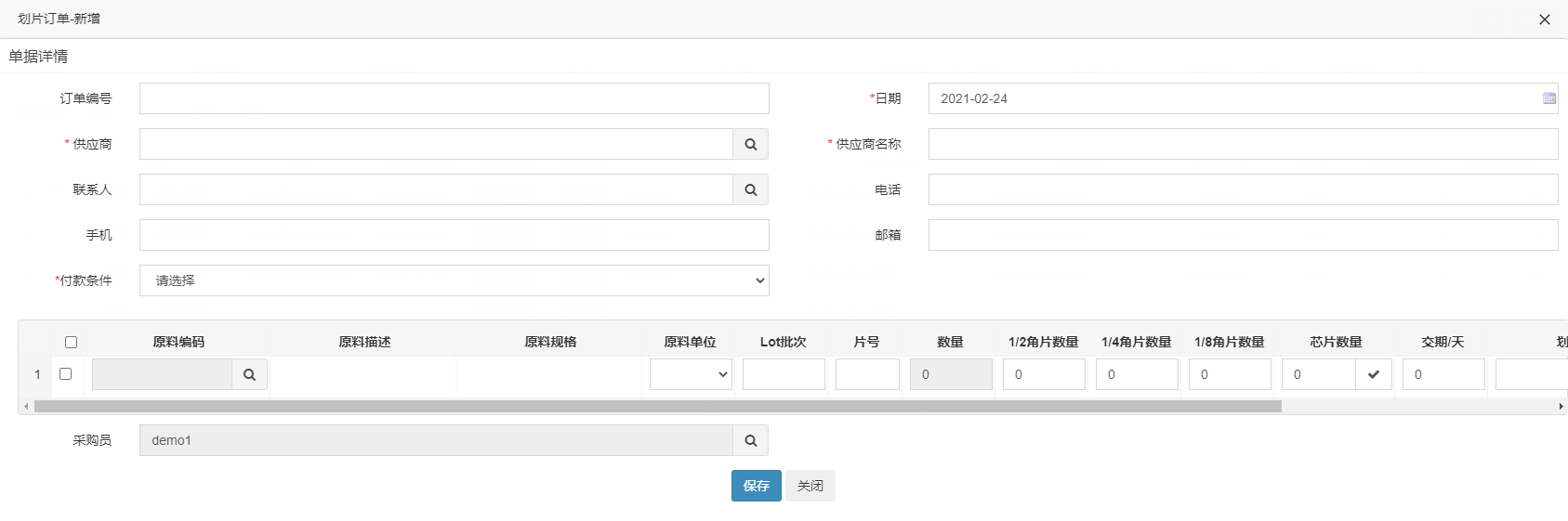
生成后的数据通过EF Core框架进行数据库保存，为方便追溯操作记录，此时也需要将当前登录用户的信息进行保存；保存失败时抛出异常提醒，便于用户及时反馈整理问题，并重新提交。

保存成功后便于用户良好的体验感，应关闭当前页面，返回晶圆订单列表页面（如图4.14所示），刷新列表，获取用户刚添加的数据。



**图4.14 晶圆测试订单列表页面**

### 4.4.2 划片订单管理



**图 4.15 新增划片订单页面**

选择供应商等必要条件，用户输入的数据如图4.15所示，将用户输入的明细数据获取成为数组，除明细外的主要信息，使用AJAX传递到后端实体类。

接口接收传过来的数据后，按照规则生成订单编号，订单编号由用户设置的“划片订单编码前缀” +“日期”字符串+“序列号”生成，序列号同样由Redis缓存进行存储，并在每天的0点清除，确保每天的序列号由1开始。

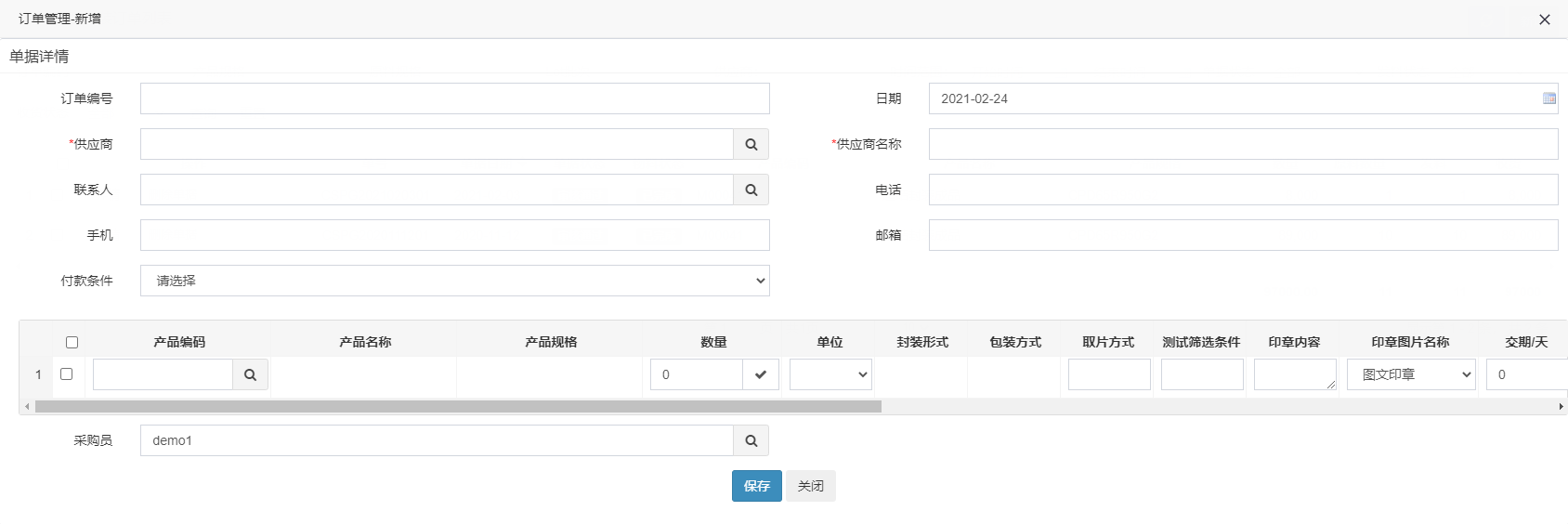
生成后的数据通过EF Core框架进行数据库保存，为方便追溯操作记录，此时也需要将当前登录用户的信息进行保存；保存失败时抛出异常提醒，便于用户及时反馈整理问题，并重新提交。

保存成功后便于用户良好的体验感，应关闭当前页面，返回划片订单列表页面（如图4.16所示），刷新列表，获取用户刚添加的数据。



**图4.16 划片订单列表页面**

### 4.4.3 封装订单管理



**图 4.17 新增封装订单页面**

选择供应商等必要条件，用户输入的数据如图4.17所示，将用户输入的明细数据获取成为数组，除明细外的主要信息，使用AJAX传递到后端实体类。

接口接收传过来的数据后，按照规则生成订单编号，订单编号由用户设置的“划片订单编码前缀” +“日期”字符串+“序列号”生成，序列号同样由Redis缓存进行存储，并在每天的0点清除，确保每天的序列号由1开始。

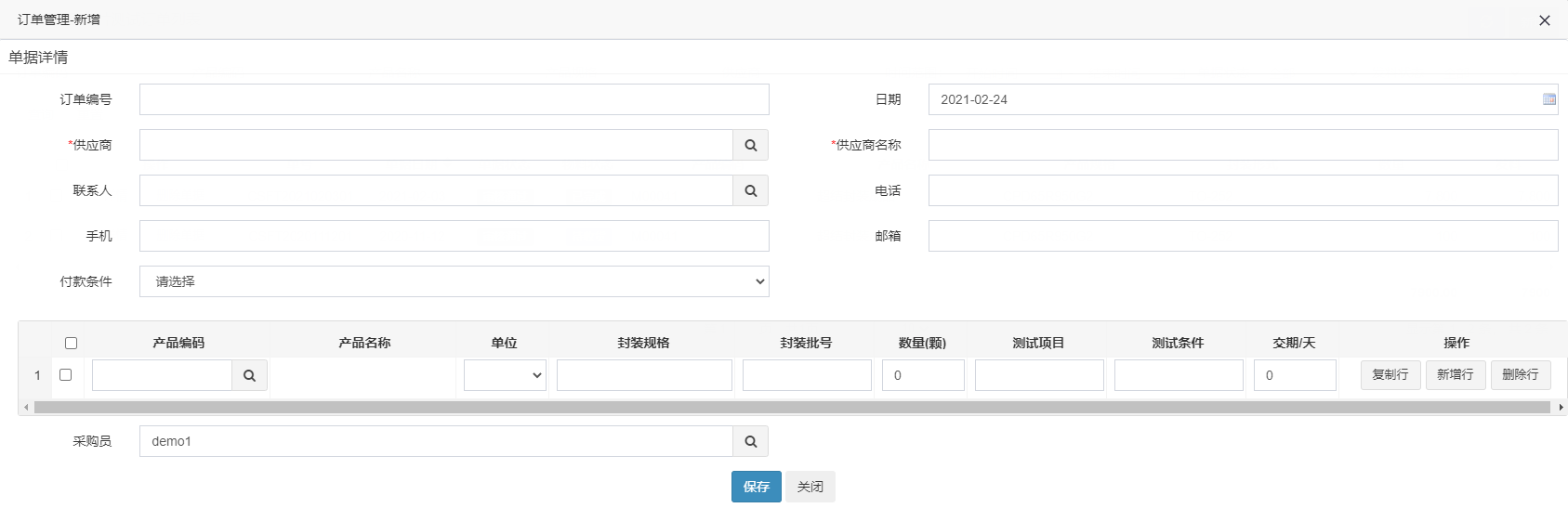
生成后的数据通过EF Core框架进行数据库保存，为方便追溯操作记录，此时也需要将当前登录用户的信息进行保存；保存失败时抛出异常提醒，便于用户及时反馈整理问题，并重新提交。

保存成功后便于用户良好的体验感，应关闭当前页面，返回划片订单列表页面（如图4.18所示），刷新列表，获取用户刚添加的数据。



**图4.18 封装订单列表页面**

### 4.4.4 芯片测试订单管理



**图 4.19 新增芯片测试订单页面**

选择供应商等必要条件，用户输入的数据如图4.19所示，将用户输入的明细数据获取成为数组，除明细外的主要信息，使用AJAX传递到后端实体类。

接口接收传过来的数据后，按照规则生成订单编号，订单编号由用户设置的“芯片测试订单编码前缀” +“日期”字符串+“序列号”生成，序列号同样由Redis缓存进行存储，并在每天的0点清除，确保每天的序列号由1开始。

生成后的数据通过EF Core框架进行数据库保存，为方便追溯操作记录，此时也需要将当前登录用户的信息进行保存；保存失败时抛出异常提醒，便于用户及时反馈整理问题，并重新提交。

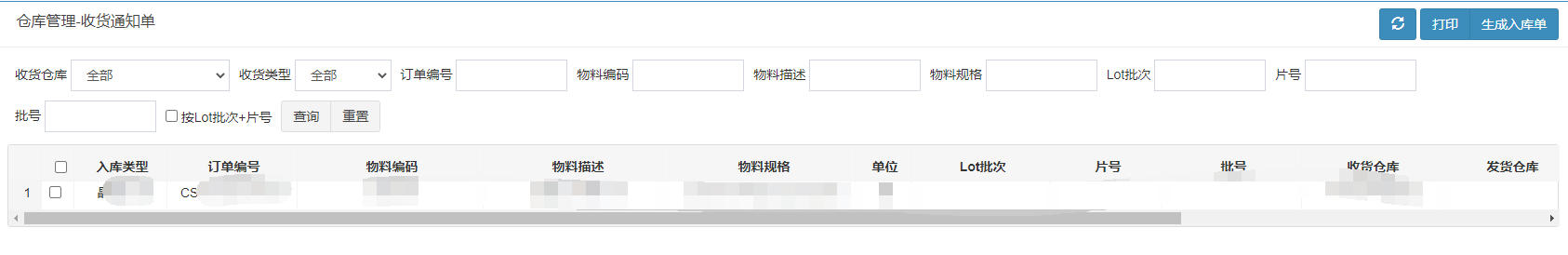
保存成功后便于用户良好的体验感，应关闭当前页面，返回芯片测试订单列表页面（如图4.20所示），刷新列表，获取用户刚添加的数据。



**图 4.20 芯片测试订单页面**

## 4.5 仓库管理

### 4.5.1 收货通知单管理

****

**图 4.21 收货通知单列表页面**

选择需要生成入库单的数据记录，将所选中的收货通知单的唯一标识，并组织为数组类型，访问接口将收货通知单的唯一标识所组成的数组传递到接口当中，后端使用数据进行接收，使用Redis缓存记录该数组并为该数组生成唯一标识。

接口访问成功并正确将数组记录到缓存当中后，跳转到生成入库单界面并将数组的唯一标识传递到该页面中（生成入库单页面如图4.22所示），加载界面成功后优先访问数据列表接口，将唯一标识传递到后端接口，获取之前保存在缓存中的收货通知单明细列表。

检查数据无误后，点击保存按钮，将用户输入的明细数据获取成为数组，除明细外的主要信息，使用AJAX传递到后端实体类。

接口接收传过来的数据后，按照规则生成订单编号，订单编号由用户设置的“入库订单编码前缀” +“日期”字符串+“序列号”生成，序列号同样由Redis缓存进行存储，并在每天的0点清除，确保每天的序列号由1开始。

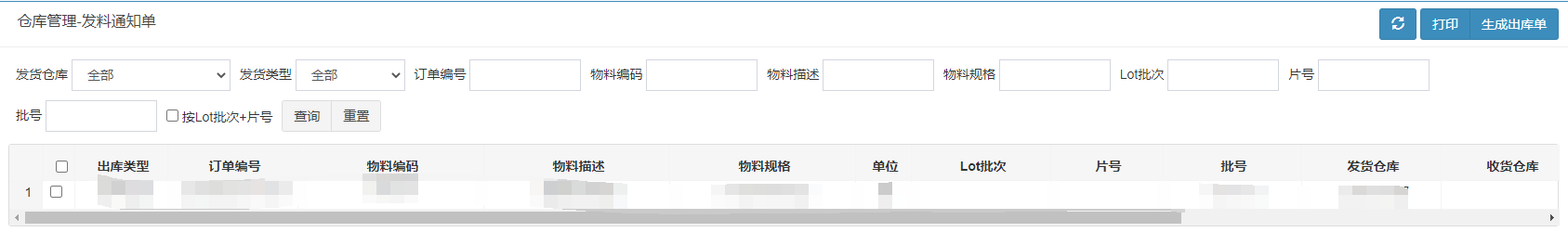
生成后的数据通过EF Core框架进行数据库保存，为方便追溯操作记录，此时也需要将当前登录用户的信息进行保存；保存失败时抛出异常提醒，便于用户及时反馈整理问题，并重新提交。

保存成功后便于用户良好的体验感，应关闭当前页面，返回收货通知单列表页面。



**图 4.22 生成收货单页面**

### 4.5.2 发料通知单管理



**图 4.23 发料通知单列表页面**

选择需要生成出库单的数据记录，将所选中的发料通知单的唯一标识，并组织为数组类型，访问接口将发料通知单的唯一标识所组成的数组传递到接口当中，后端使用数据进行接收，使用Redis缓存记录该数组并为该数组生成唯一标识。

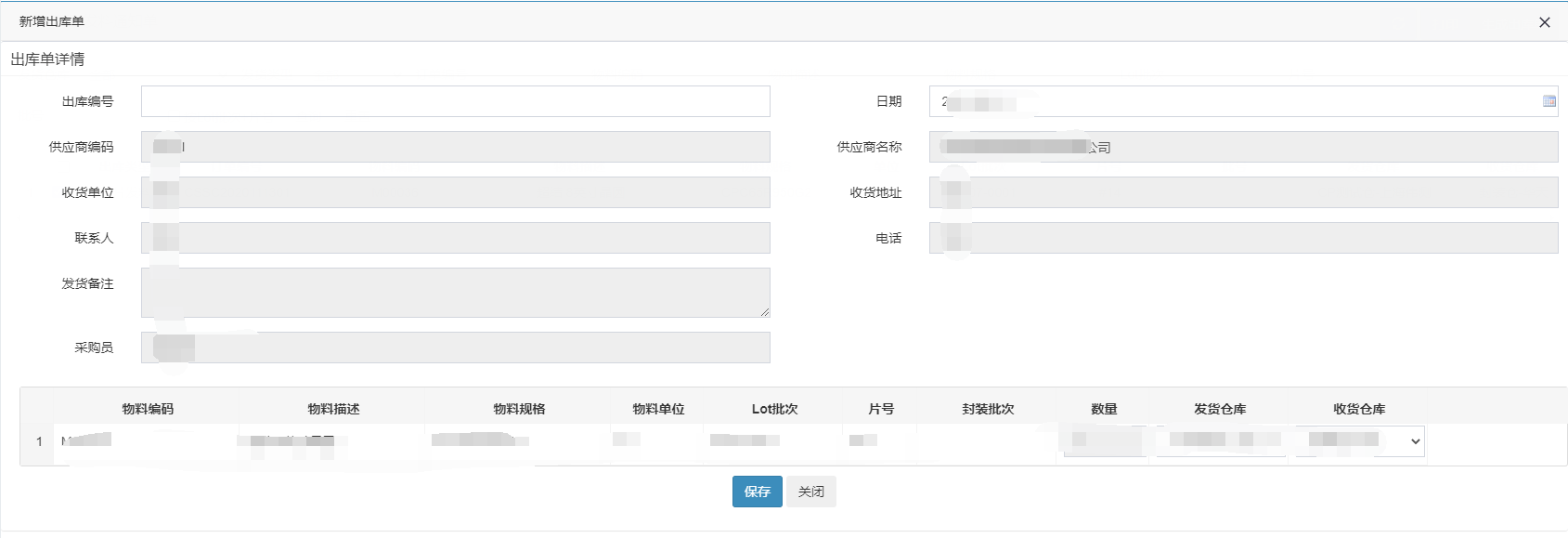
接口访问成功并正确将数组记录到缓存当中后，跳转到生成入库单界面并将数组的唯一标识传递到该页面中（生成出库单页面如图4.24所示），加载界面成功自动访问数据列表接口，将唯一标识传递到后端接口，获取之前保存在缓存中的发料通知单明细列表。

检查数据无误后，点击保存按钮，将用户输入的明细数据获取成为数组，除明细外的主要信息，使用AJAX传递到后端实体类。

接口接收传过来的数据后，按照规则生成订单编号，订单编号由用户设置的“出库订单编码前缀” +“日期”字符串+“序列号”生成，序列号同样由Redis缓存进行存储，并在每天的0点清除，确保每天的序列号由1开始。

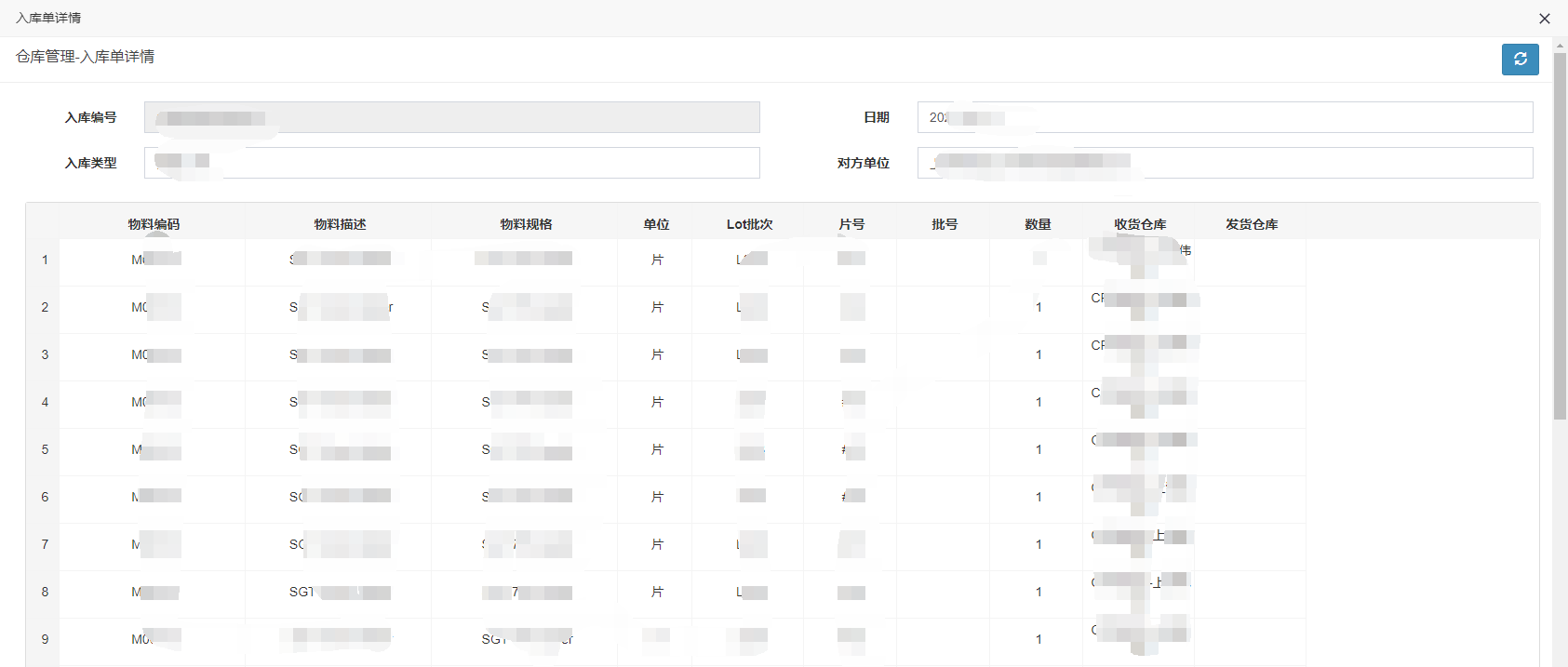
生成后的数据通过EF Core框架进行数据库保存，为方便追溯操作记录，此时也需要将当前登录用户的信息进行保存；保存失败时抛出异常提醒，便于用户及时反馈整理问题，并重新提交。

保存成功后便于用户良好的体验感，应关闭当前页面，返回发料通知单列表页面。



**图4.24 生成发货单页面**

### 入库单管理



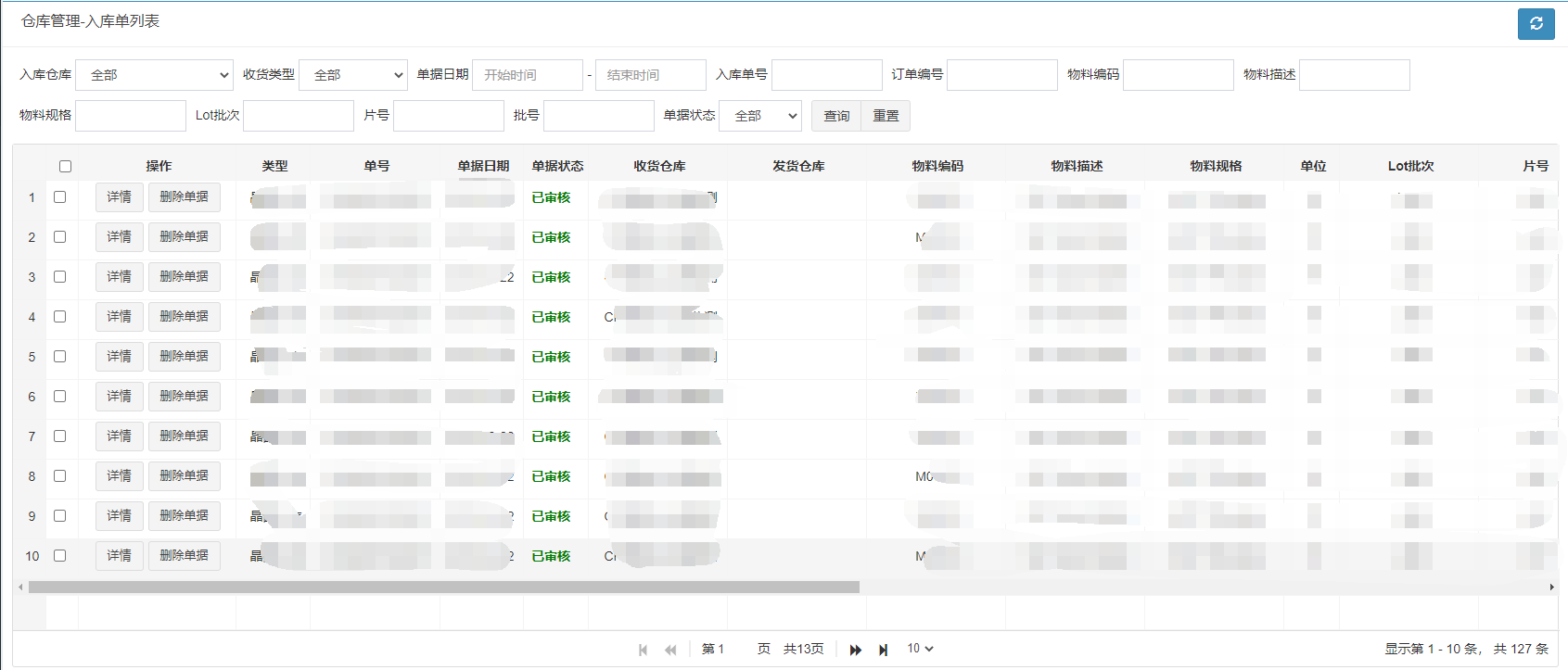
**图4.25 入库单详情页面**

点击“详情”按钮，进入入库单详情页面（入库单详情页面如图4.25所示），检查数据无误后审核该入库单。

审核入库单时只需传递该入库单编号即可，后端采用参数接收，根据入库单所对应的入库类型标识，进行核销供应商仓库的原料，并且入库供应商所发回的产品。

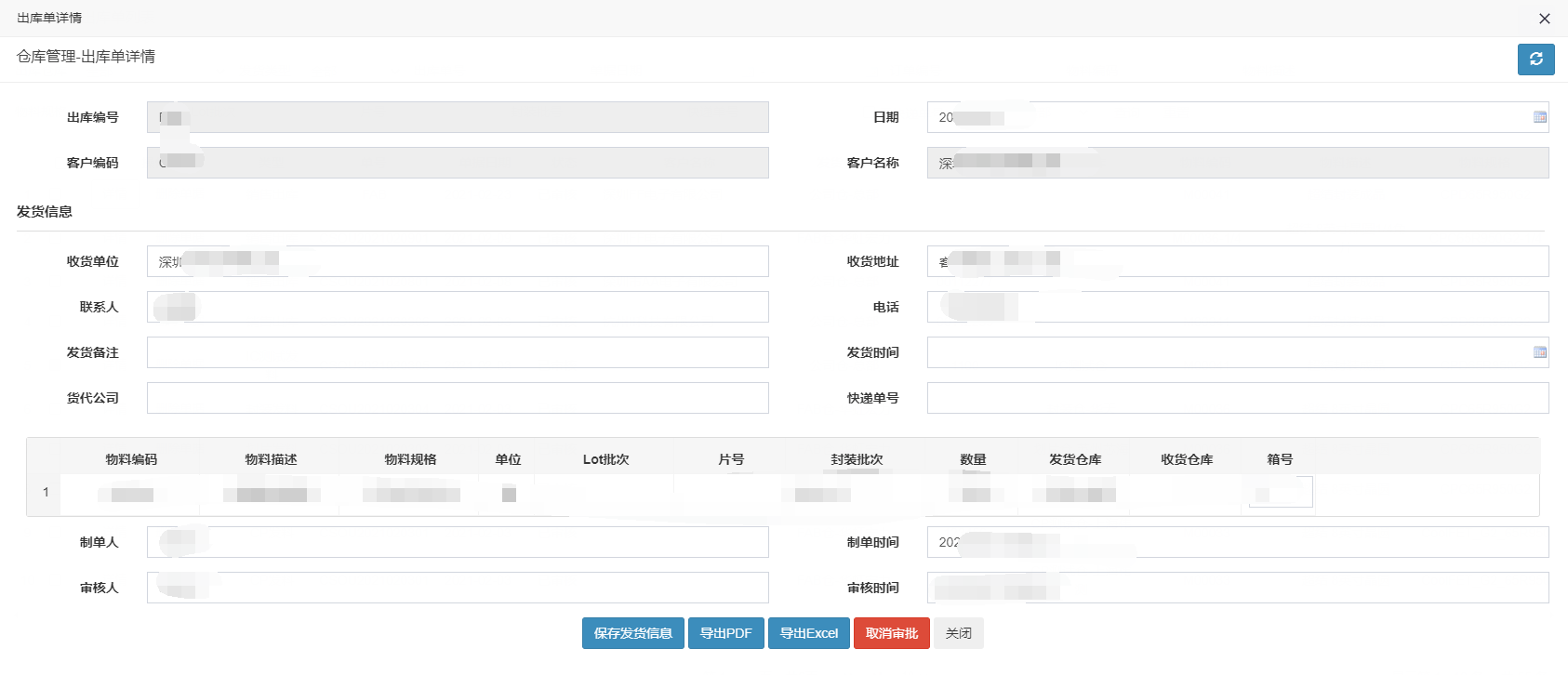
入库单审核入库后，使用EF Core框架查询该入库单所对应的生产订单或采购订单，更改生产订单或采购订单的入库数量。为方便追溯操作记录，此时也需要将当前登录用户的信息进行保存；保存失败时抛出异常提醒，便于用户及时反馈整理问题，并重新提交。

审核入库单成功后便于用户良好的体验感，应关闭当前页面，返回入库单列表页面（如图4.26所示）。



**图 4.26 入库单列表页面**

### 4.5.4 出库单管理

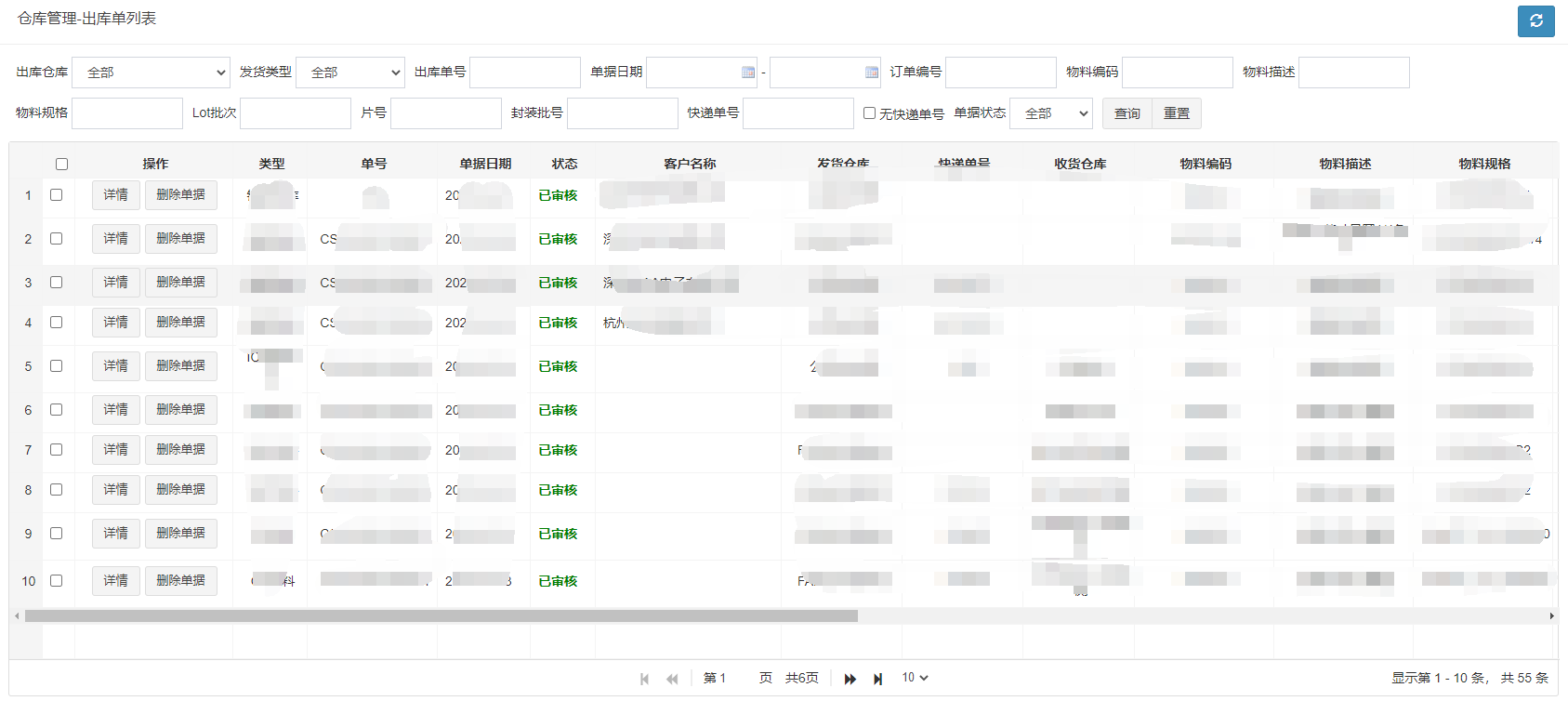


**图4.27 出库单详情页面**

点击“详情”按钮，进入出库单详情页面（出库单详情页面如图4.27所示），检查数据无误后审核该出库单。

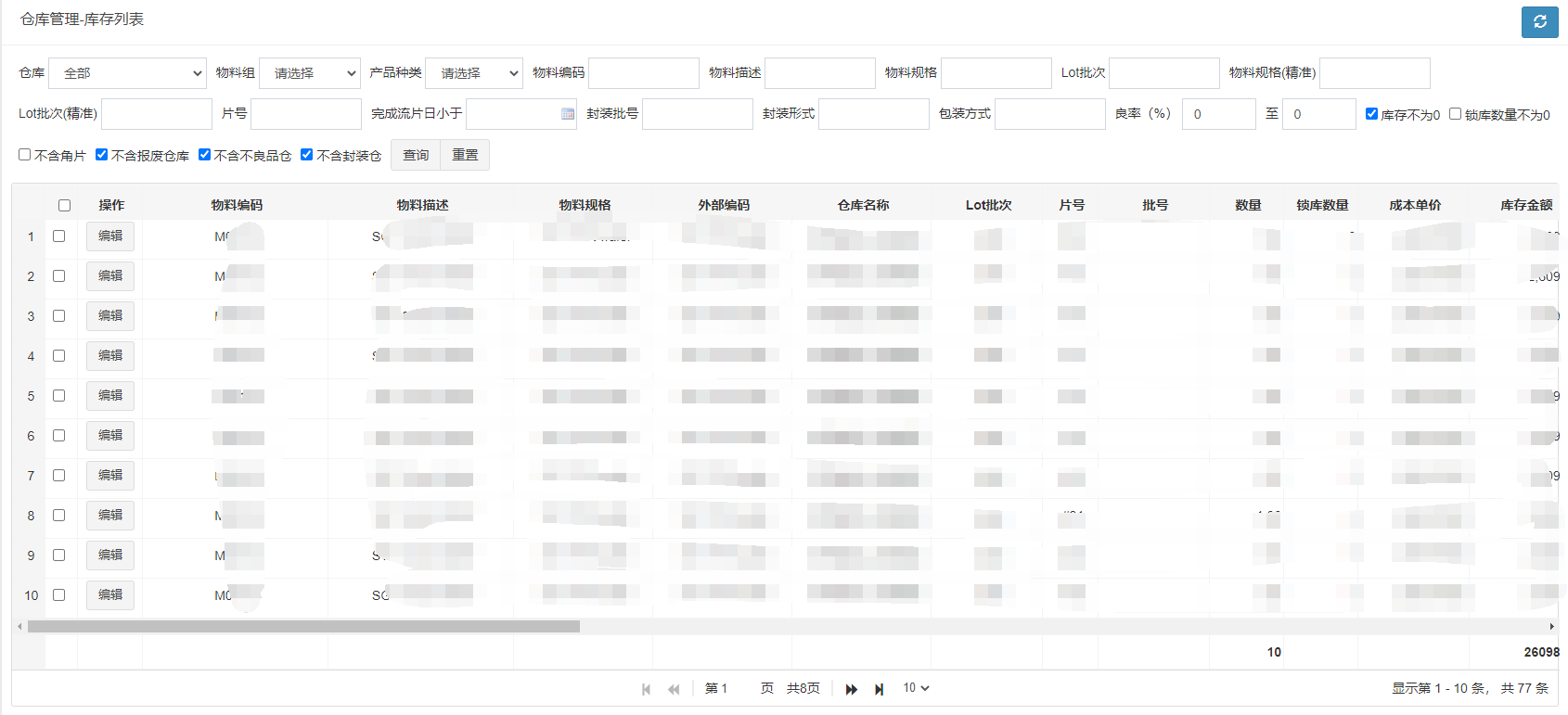
审核出库单时只需传递该出库单编号即可，后端采用参数接收，根据出库单所对应的出库类型标识，出库供应商所要生产的原料。出库单审核出库后，使用EF Core框架查询该出库单所对应的生产订单或销售订单，更改生产订单或销售订单的出库数量。为方便追溯操作记录，此时也需要将当前登录用户的信息进行保存；保存失败时抛出异常提醒，便于用户及时反馈整理问题，并重新提交。

审核出库单成功后便于用户良好的体验感，应关闭当前页面，返回出库单列表页面（如图4.26所示）。



**图 4.28 出库单列表页面**

### 4.5.5 库存查询



**图4.29 库存列表页面**

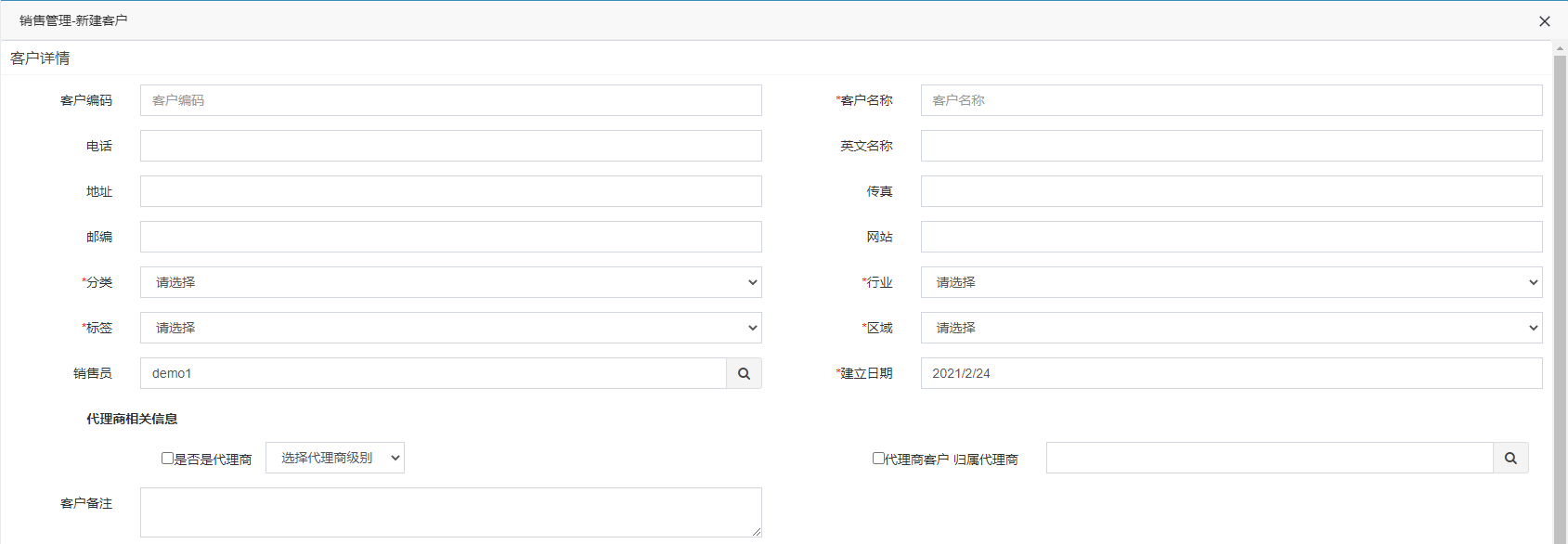
可根据条件进行查询，使用form表单或Jquery获取用户输入的查询条件（查询条件内容如图4.29所示），将其组合成为JSON格式的数据，传递给后端。

后端则采用实体类来接收查询条件，对于不需要转换类型的查询条件，可使用三元运算符来判断传入的参数是否为空，使代码比较简洁，更加通俗易懂。

后端将查询到的数据通过JSON格式返回，前端使用Jquery进行处理，从而展示用户所查询出的所有库存情况。

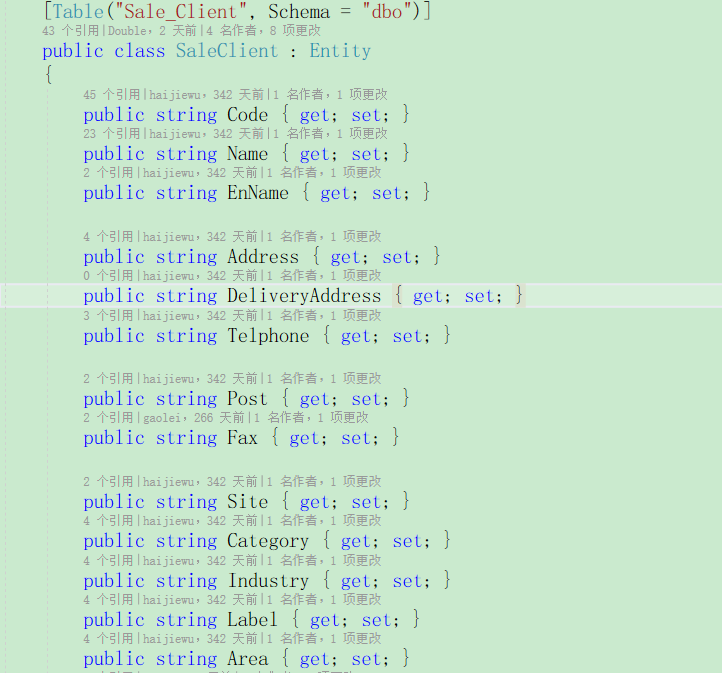
## 4.6 销售管理

### 4.6.1 客户管理



**图 4.30 新增客户页面**

填写用户名称等必要条件，将用户输入的信息使用form表单提交（form表单所提交的内容如图4.30所示），form表单提交的好处是可以整个表单中的数据进行提交，而不用一个一个的去获取参数，简洁方便。



**图4.31 客户实体类**

用户输入的数据传递到接口，后端使用实体类进行接收（实体类参数如图4.31所示）。物料编码的生成规则由“客户编码前缀”+“序列号”生成，“序列号”是顺序的，由缓存Redis来完成，Redis具有高可用、读取速度快的特性，很适合用来做这样的功能。

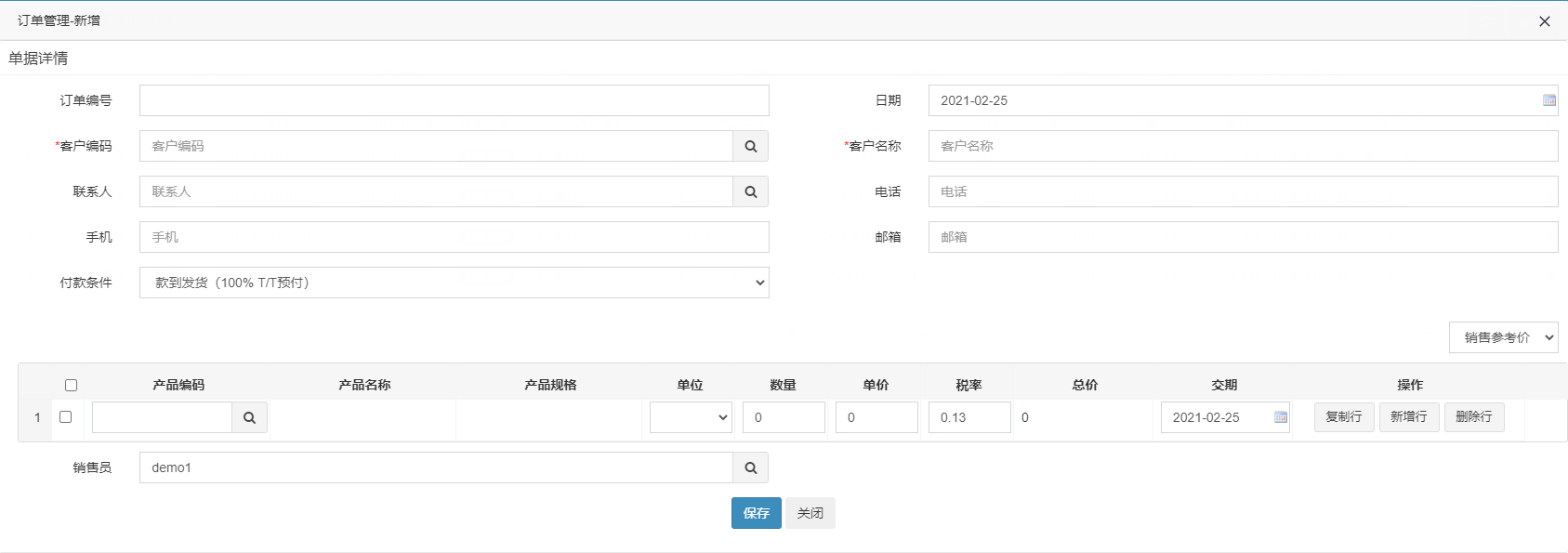
生成后物料编码的数据实体类通过EF Core框架进行数据的保存。为方便追溯操作记录，此时也需要将当前登录用户的信息进行保存；保存失败时抛出异常提醒，便于用户及时反馈整理问题，并重新提交。

保存成功后便于用户良好的体验感，则返回客户列表页面（如图4.32所示）。



**图 4.32 客户列表页面**

### 4.6.2 销售报价单管理



**图 4.33 新增销售报价单页面**

选择客户等必要条件，将用户输入的信息使用AJAX传递到接口，因为具有明细列表，不可通过form表单来进行提交，所以要通过Jquery获取列表的内容形成JS 数组传递到后端进行处理。



**图4.34 销售报价单实体类**

后端使用数组类型的参数用来接收明细的内容，其余主要的信息使用实体类来接收（实体类如图4.34所示），报价单编号在不输入的情况下采取按规则进行生成，报价单的规则由用户设置的“销售报价单编码”前缀+“日期”字符串+“序列号”生成，序列号同样由Redis缓存进行存储，并在每天的0点清除，确保每天的序列号由1开始。

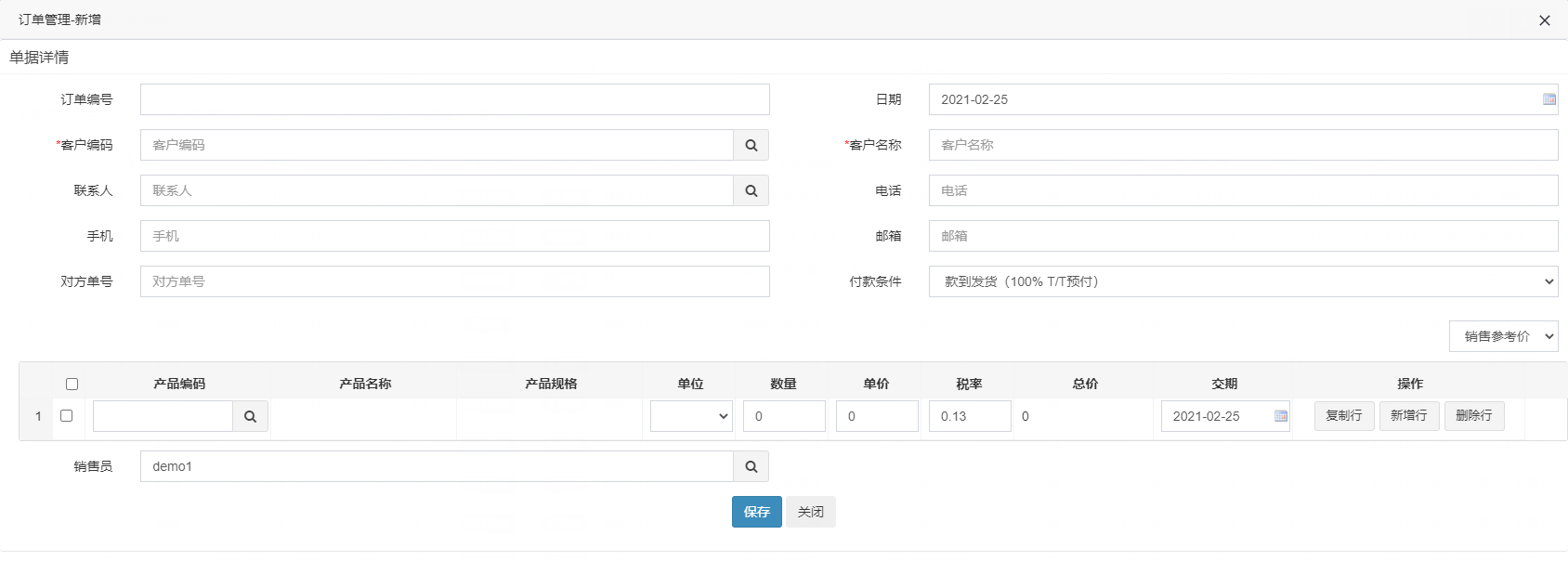
生成后的数据通过EF Core框架进行数据库保存，为方便追溯操作记录，此时也需要将当前登录用户的信息进行保存；保存失败时抛出异常提醒，便于用户及时反馈整理问题，并重新提交。

保存成功后便于用户良好的体验感，则返回销售报价单列表页面（如图4.35所示）。



**图4.35 销售报价单列表页面**

### 4.6.3 销售单管理



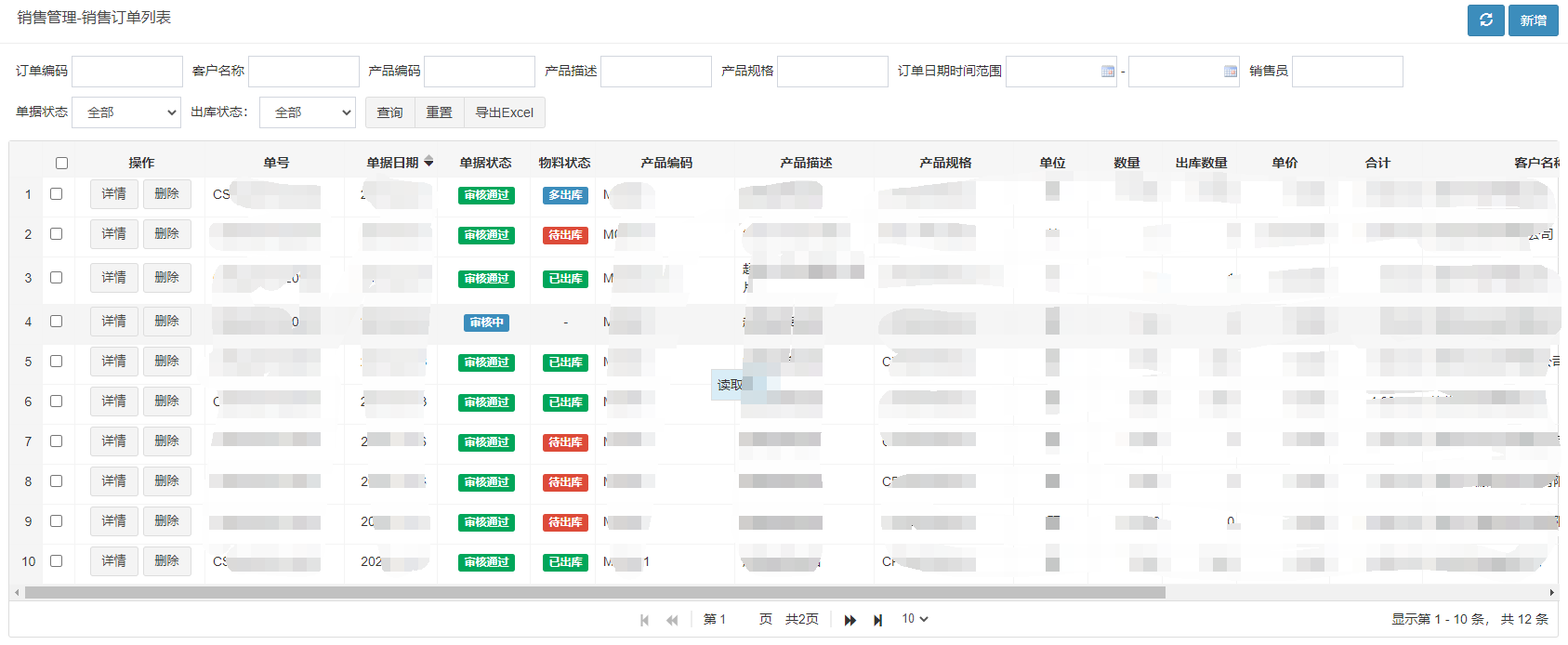
**图 4.36 新增销售订单页面**

选择客户等必要条件，将用户输入的信息使用AJAX传递到接口，因为具有明细列表，不可通过form表单来进行提交，所以要通过Jquery获取列表的内容形成JS 数组传递到后端进行处理。

后端使用数组类型的参数用来接收明细的内容，其余主要的信息使用实体类来接收，销售单编号在不输入的情况下采取按规则进行生成，销售订单的规则由用户设置的“销售订单编码”前缀+“日期”字符串+“序列号”生成，序列号同样由Redis缓存进行存储，并在每天的0点清除，确保每天的序列号由1开始。

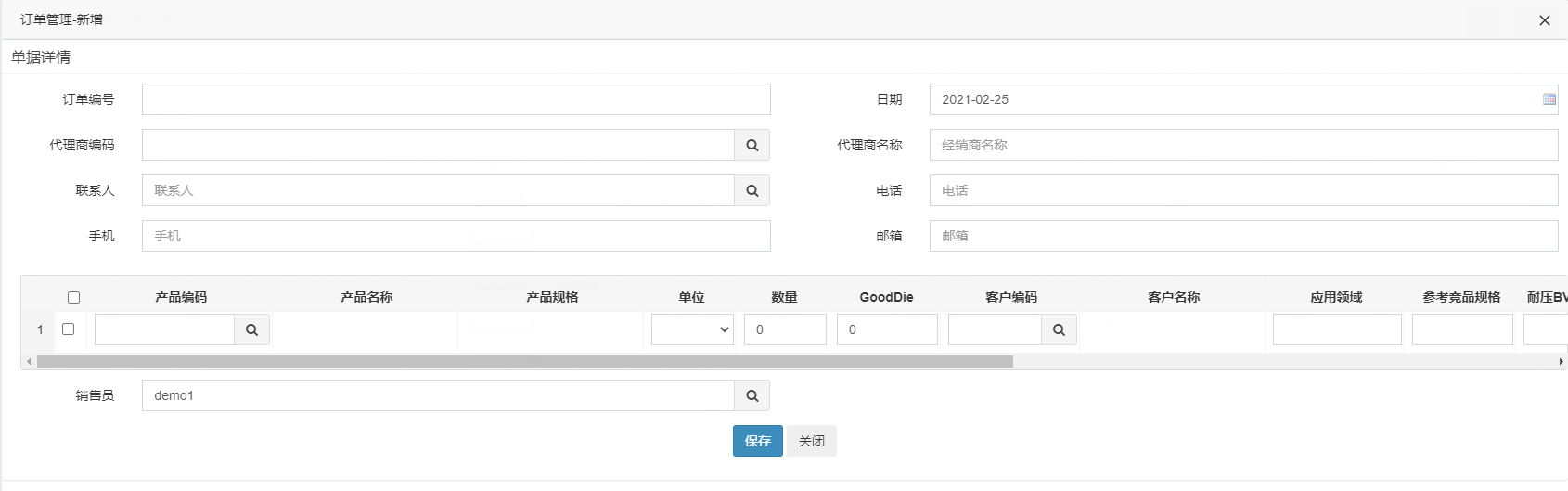
生成后的数据通过EF Core框架进行数据库保存，为方便追溯操作记录，此时也需要将当前登录用户的信息进行保存；保存失败时抛出异常提醒，便于用户及时反馈整理问题，并重新提交。

保存成功后便于用户良好的体验感，则返回销售订单列表页面（如图4.37所示）。



**图4.37 销售订单列表页面**

### 4.6.4 送样需求单管理



**图 4.38 新增送样需求单页面**

选择客户等必要条件，将用户输入的信息使用AJAX传递到接口，因为具有明细列表，不可通过form表单来进行提交，所以要通过Jquery获取列表的内容形成JS 数组传递到后端进行处理。

后端使用数组类型的参数用来接收明细的内容，其余主要的信息使用实体类来接收，送样单编号在不输入的情况下采取按规则进行生成，销售订单的规则由用户设置的“送样订单编码”前缀+“日期”字符串+“序列号”生成，序列号同样由Redis缓存进行存储，并在每天的0点清除，确保每天的序列号由1开始。

生成后的数据通过EF Core框架进行数据库保存，为方便追溯操作记录，此时也需要将当前登录用户的信息进行保存；保存失败时抛出异常提醒，便于用户及时反馈整理问题，并重新提交。

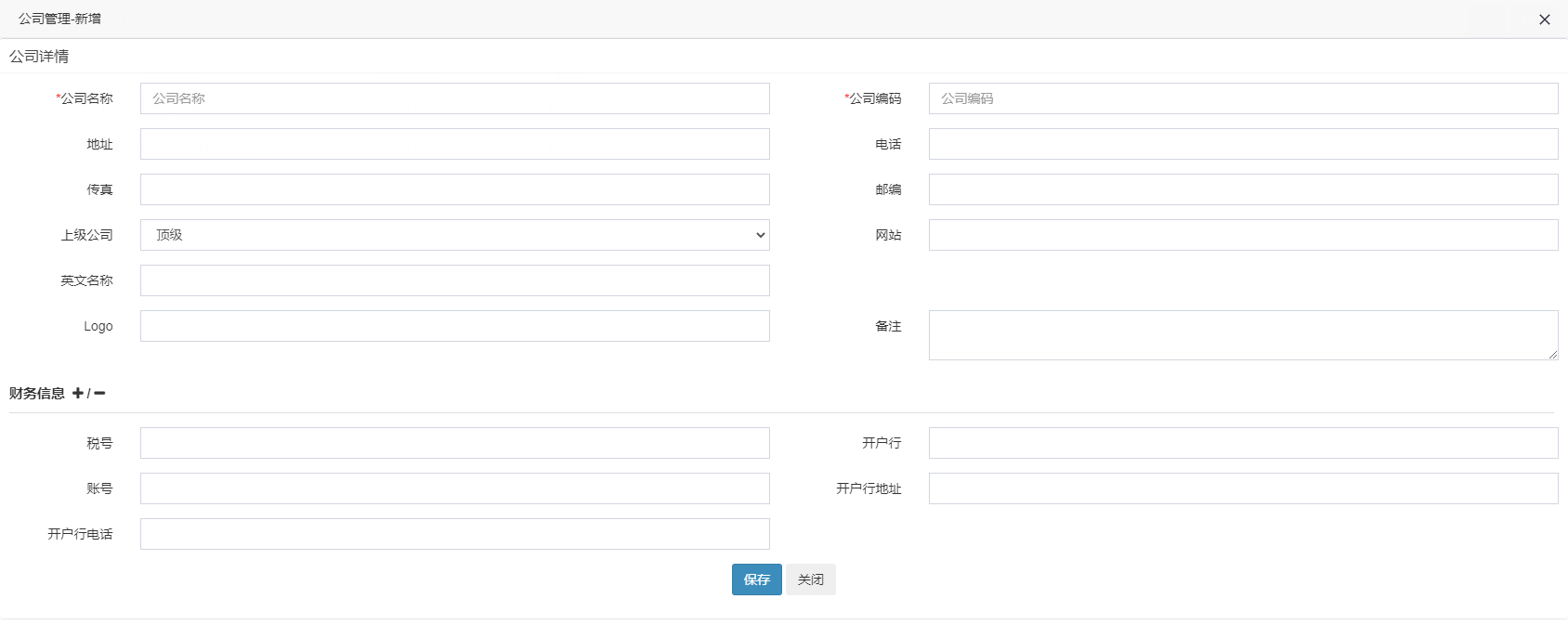
保存成功后便于用户良好的体验感，则返回送样订单列表页面（如图4.39所示）。



**图4.39 送样需求单列表页面**

## 4.7 系统管理

### 4.7.1 公司管理



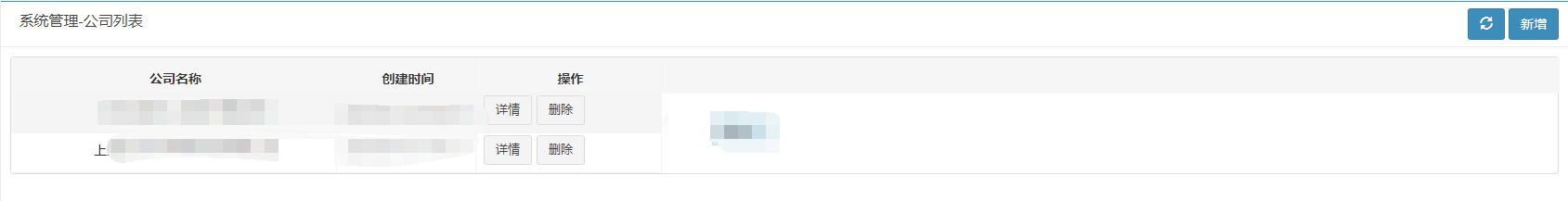
**图 4.40 新增公司页面**

填写公司名称等必要条件，将用户输入的信息使用form表单提交（form表单所提交的内容如图4.40所示），form表单提交的好处是可以整个表单中的数据进行提交，而不用一个一个的去获取参数，简洁方便。

用户输入的数据传递到接口，后端使用实体类进行接收。公司编码的生成规则由“公司编码前缀”+“序列号”生成，“序列号”是顺序的，由缓存Redis来完成，Redis具有高可用、读取速度快的特性，很适合用来做这样的功能。

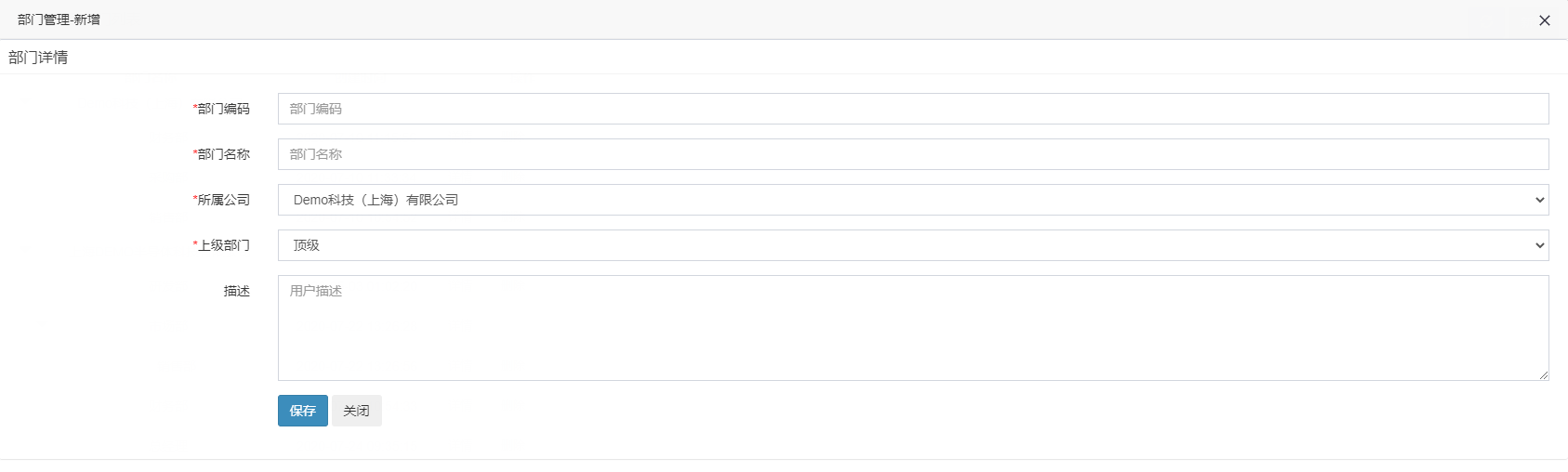
生成后公司编码的数据实体类通过EF Core框架进行数据的保存。为方便追溯操作记录，此时也需要将当前登录用户的信息进行保存；保存失败时抛出异常提醒，便于用户及时反馈整理问题，并重新提交。

保存成功后便于用户良好的体验感，则返回公司列表页面（如图4.41所示）。



**图 4.41 公司列表页面**

### 4.7.2 部门管理



**图 4.42 新增部门页面**

填写部门名称等必要条件，将用户输入的信息使用form表单提交（form表单所提交的内容如图4.42所示），form表单提交的好处是可以整个表单中的数据进行提交，而不用一个一个的去获取参数，简洁方便。

用户输入的数据传递到接口，后端使用实体类进行接收。部门编码的生成规则由“部门编码前缀”+“序列号”生成，“序列号”是顺序的，由缓存Redis来完成，Redis具有高可用、读取速度快的特性，很适合用来做这样的功能。

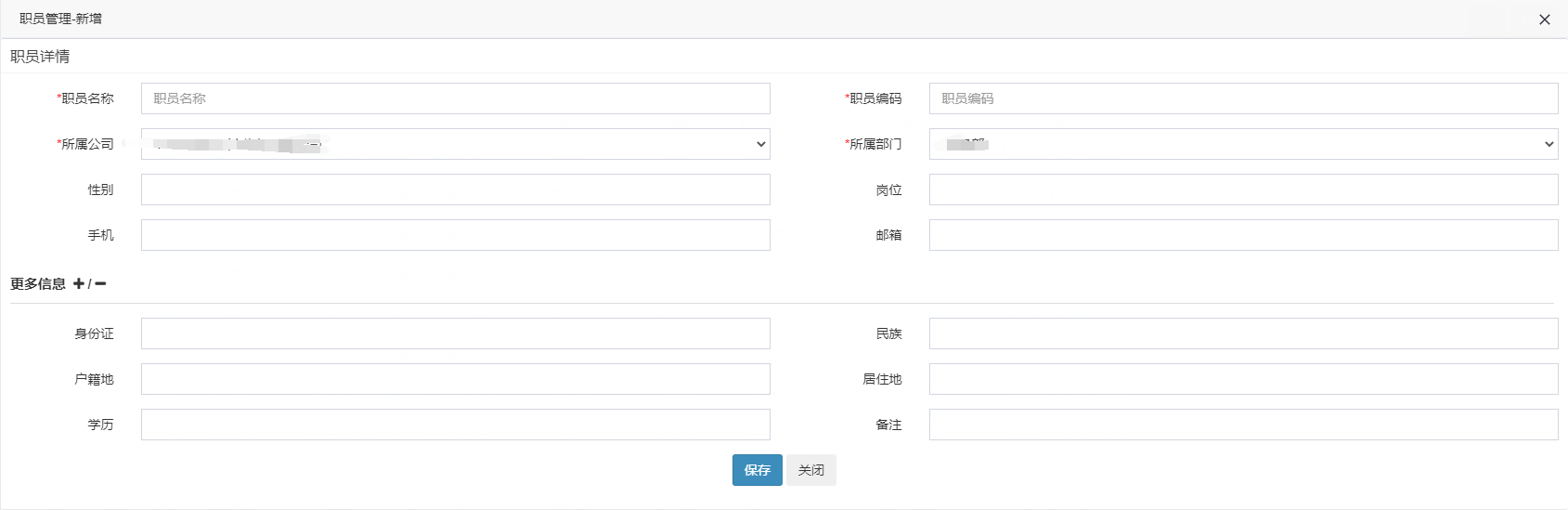
生成后部门编码的数据实体类通过EF Core框架进行数据的保存。为方便追溯操作记录，此时也需要将当前登录用户的信息进行保存；保存失败时抛出异常提醒，便于用户及时反馈整理问题，并重新提交。

保存成功后便于用户良好的体验感，则返回部门列表页面（如图4.43所示）。



**图 4.43 部门列表页面**

### 4.7.3 职员管理



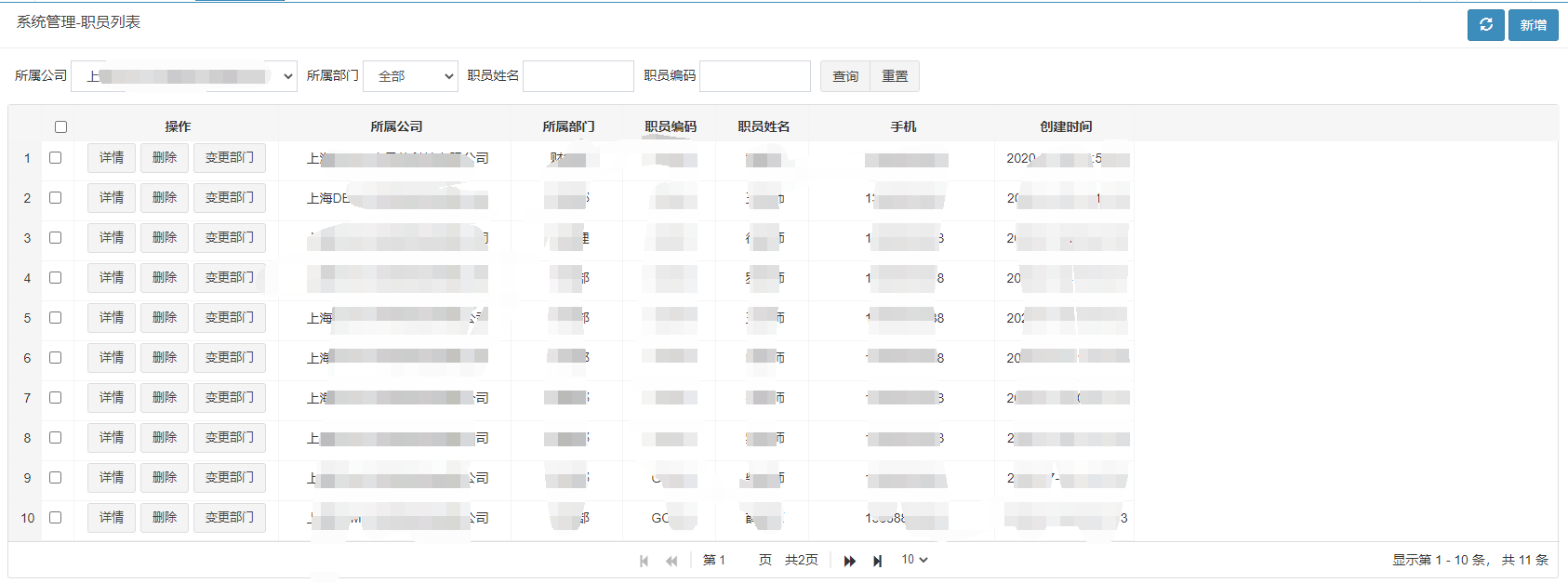
**图 4.44 新增职员页面**

填写职员名称等必要条件，将用户输入的信息使用form表单提交（form表单所提交的内容如图4.44所示），form表单提交的好处是可以整个表单中的数据进行提交，而不用一个一个的去获取参数，简洁方便。

用户输入的数据传递到接口，后端使用实体类进行接收。部门编码的生成规则由“职员编码前缀”+“序列号”生成，“序列号”是顺序的，由缓存Redis来完成，Redis具有高可用、读取速度快的特性，很适合用来做这样的功能。

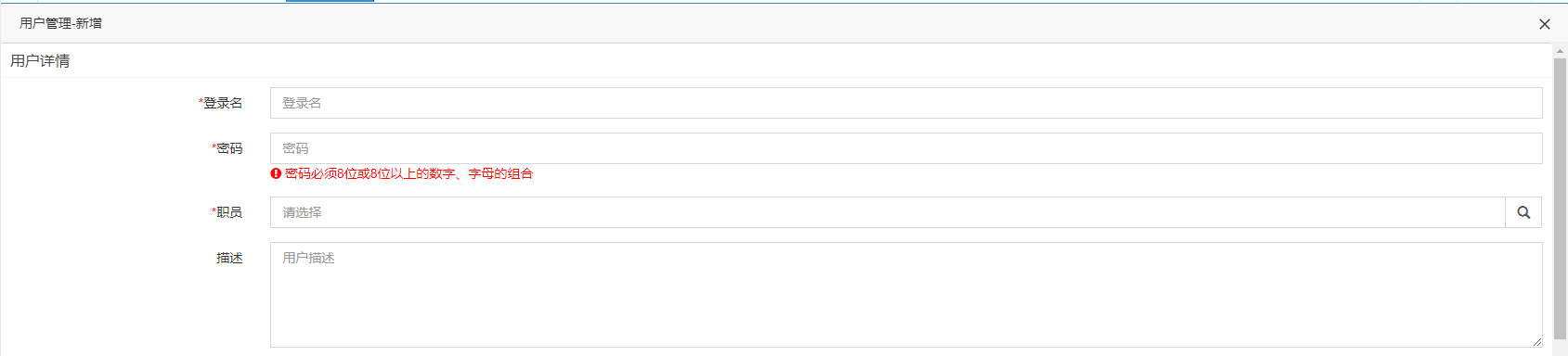
生成后职员编码的数据实体类通过EF Core框架进行数据的保存。为方便追溯操作记录，此时也需要将当前登录用户的信息进行保存；保存失败时抛出异常提醒，便于用户及时反馈整理问题，并重新提交。

保存成功后便于用户良好的体验感，则返回职员列表页面（如图4.45所示）。



**图 4.45 职员列表页面**

### 4.7.4 系统用户管理



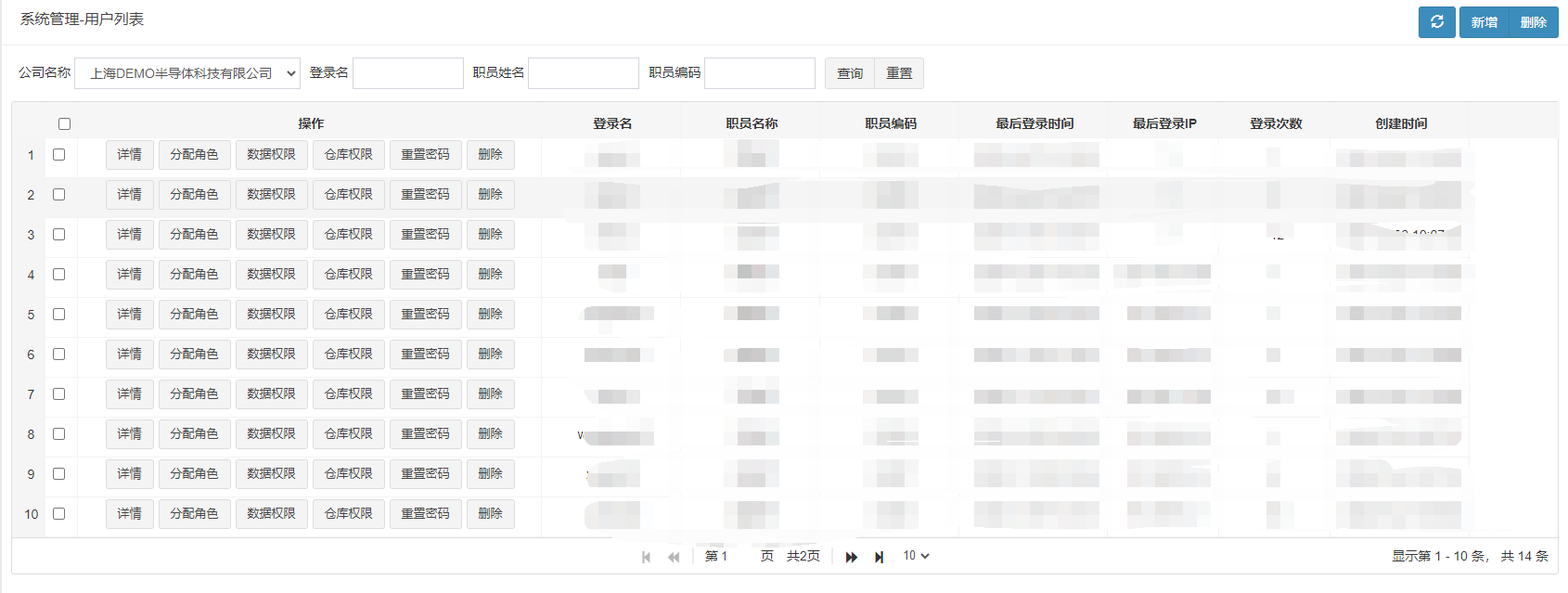
**图4.46 新增系统用户页面**

填写登录名等必要条件，将用户输入的信息使用form表单提交（form表单所提交的内容如图4.46所示）。

用户输入的数据传递到接口，后端使用实体类进行接收。

数据实体类通过EF Core框架进行数据的保存。为方便追溯操作记录，此时也需要将当前登录用户的信息进行保存；保存失败时抛出异常提醒，便于用户及时反馈整理问题，并重新提交。

保存成功后便于用户良好的体验感，则返回用户列表页面（如图4.47所示）。



**图4.47 系统用户列表页面**

# 第五章 系统测试

## 5.1 白盒测试法

白盒测试法是按照程序的执行逻辑路径进行检测，检查是不是有存在引起程序出现错误的语句。在进行白盒测试的时候，使用调试的工具进行流程跟踪，确保操作经过每个流程时的流畅性。

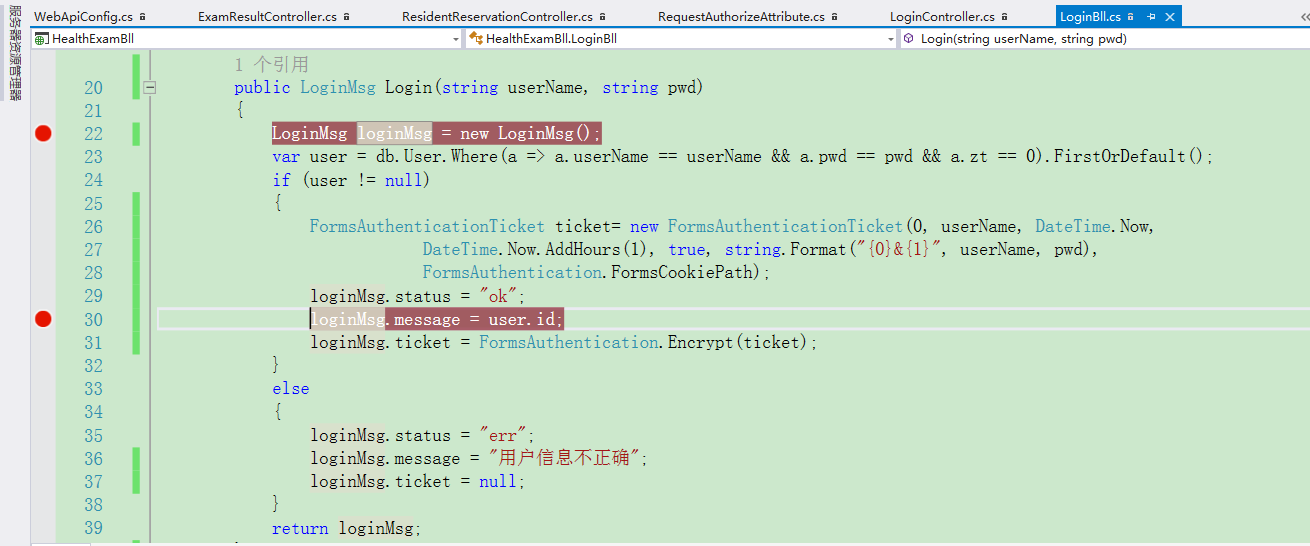
在测试的时候发现的软件错误必须立即改正，并且类似过程进行异常处理，这就是大概调试的任务。为了纠正错误，首先必须对bug进行定位，同时尽量考虑全面以避免在调试过程中引起新的作物。必须使用Debug调试器就很容易地定位到错误并进行纠正。

下面以登录方法为例，以此来说明调试的过程和方法：

第一步：打开Debug画板；

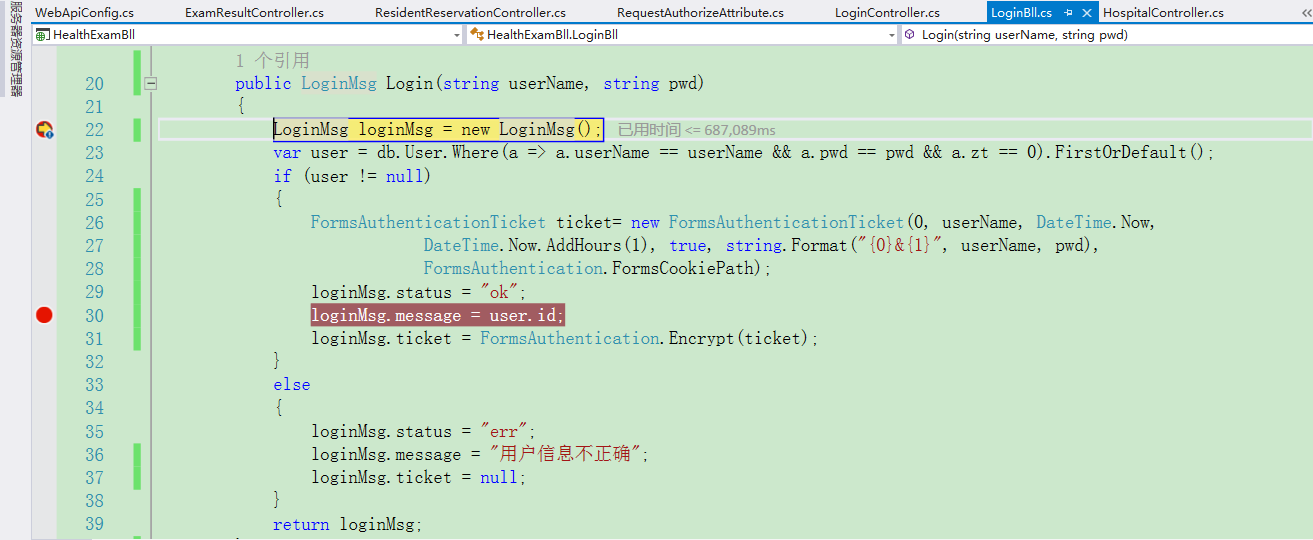
第二步：在程序中设置断点；

或者两步颠倒顺序也是可以的（如图5.1所示）。



**图5.1 Debug画板**

第三步：在调试跟踪的方式下运行应用程序（如图5.2所示）；



**图5.2 运行应用程序**

第四步：当应用程序处在断点处执行时，可以监控到变量所对应的值的变化，检查内存中对象的属性或改变对应变量的值。

第五步：依次调试应用程序，可以根据适当情况在程序中增加新的断点或取消已存在的断点。

第六步：根据第五步中找到问题的所在，修改问题代码并重新运行该应用程序。

通过调试会发现一般错误都出现在马虎或考虑不全面，比如数据类型不符合，非空验证等。再有就是对数据库的数据类型使用不恰当。但经过最后认真分析、查阅资料，这些问题都得以解决。

## 5.2 黑盒测试法

黑盒测试是以用户的角度，从[输入](https://baike.baidu.com/item/%E8%BE%93%E5%85%A5/32696)数据与[输出](https://baike.baidu.com/item/%E8%BE%93%E5%87%BA/11056752)数据的对应关系出发进行测试的。很明显，如果外部特性本身设计有问题或规格说明的规定有误，用黑盒测试方法是发现不了的。

黑盒测试的优点：适用于功能测试、可用性测试及可接受性测试;对照说明书测试程序功能;可测试长的、复杂的程序的工作逻辑，易被理解。黑盒测试的缺点：不可能进行完全的、毫无遗漏的输入测试，有一些软件Bug或人为设置的故障通过黑盒测试是无法检测出来的。正是因为黑盒测试的测试数据来自规格说明书，这一方法的主要缺点是它依赖于规格说明书的正确性。实际上，人们并不能保证规格说明书完全正确。如在规格说明书中规定了多余的功能，或是漏掉了某些功能，这对于黑盒测试来说是完全无能为力的。

# 第六章 结论

系统开发过程中，使用仓储模式和MVC的设计模式，使得系统层次的分工更加明确，也便于以后的业务扩展。使用ORM框架EF Core更加的方便对数据库进行操作，极大的方便了开发的便捷程度和访问数据库的正确操作。

当然其中也会有自己“偷懒”的部分，直接在展示层面写了业务的代码，虽然其中逻辑无误，但是从代码设计和系统层面考虑，这都是不太正确的做法。由于对程序的底层了解较少，所以一些代码会占用更多的服务器硬件资源，这使得增加了服务器的负担，可能会使系统运行变慢等。

业务处理中，代码完全按照自己的理解进行编辑，对于一些复杂的业务，应使用更加便捷、快速的算法或者更加简洁的逻辑进行处理。

接下来在系统开发的过程中，不应“偷懒”，应遵循系统开发规范，将代码置于正确的位置，以保持代码的整洁程度。对于业务的处理，应更加了解业务的流程，想着在业务逻辑的正确性之下保证代码的运行效率。

应加强对程序底层的研究和算法的研究，使得在工作中更加的得心应手，而不是写出一堆“烂代码”。

# 参考文献

[1] 郑阿奇,刘启芬，顾韵华.SQL Server教程.北京：清华大学出版社，2015:75-75.

[2]罗鸿，王忠民.ERP原理、设计和实施[M].北京：电子工业出版社，2003：68-73.

[3]RonaldJN.面向对象系统分析与设计[M].周之英，等，译.北京：清华大学出版社，2000：89-102.

[4]陈启申.ERP——从内部集成起步[M].电子工业出版社，2004.

[5]朱三妹.计算机与电子信息技术在工程管理的应用[J/OL]集成电路应用，2019(12)108-109.

[6]王志铭.数据时代下计算机技术的应用[J/OL].电子技术与软件工程,2019(22):122-123.

[7]戴昀.计算机数据库技术在信息管理中的应用现状与途径.2019(22):146-147.

[8]张信敏.ERP系统在制造企业成本管理中的应用[J].中小企业管理与科技(中旬刊),2018(02):9-10

[9]胡宗标,胡晓宇.公司全面预算管理中的ERP系统的应用研究[J].西安文理学院学报(自然科学版),2018,21(02):15-19

[10]朱金祥,孙立民,黄烈生.ERP系统中客户订单变更的研究与设计[J].软件导利,2018,17(04):164-168.

# 致谢

虽然毕业论文内容繁多，过程繁琐但我的收获却更加丰富。对开发语言的熟悉度，开发环境的熟练运用，对项目架构的了解，我都是随着代码的不断编写而不断熟悉并学会如何进行高效的使用。和论文指导老师的沟通交流使我对系统设计方面和工作量的评估有了更加清晰的认识；和公司行业顾问的沟通交流让我更加了解芯片设计行业的专业知识。通过这次毕业设计让我提前了解了这些知识，这是很珍贵的。

非常感谢在毕业论文设计中所帮助到我的人，再次感谢。