# 项目管理计划书

项目名称：GSP 药店管理系统

所在学院：软件学院

所在专业：软件工程

所在班级：软工 212

项目成员：

- 201 周新斌
- 203 纪润泽
- 210 代萌
- 219 薛薇
- 230 郑语晖

# 目录

# 一、背景描述

## 1.1 背景描述

GSP 是英文 Good Supply Practice 的缩写，意即产品供应规范，是控制医药商品流通环节所有可能发生质量事故的因素从而防止质量事故发生的一整套管理程序，医药商品在其生产、经营和销售的全过程中，由于内外因素作用，随时都有可能发生质量问题，必须在所有这些环节上采取严格措施，才能从根本上保证医药商品质量。

GSP 医药管理系统是专为医药管理行业设计的一种医药药店管理软件产品和零售管理系统，它结合了制药企业的管理实践和行业的发展趋势。系统集采购、销售、库存、财务、业务分析、GSP 管理于一体，对流通领域药品全面质量管理的各个方面进行记录和管理。**有效帮助制药企业建立科学、规范、高效的管理模式，有效提高企业的市场竞争力和效率。**

通过调研发现，药店广泛应用的系统有新海 GSP 药店管理系统、以大药店管理系统、管家婆千方百济医药管理系统。本系统按照药监部门 GSP 标准进行设计与实现。与此同时，本系统界面友好、易于学习、可靠性高、功能齐全。

## 1.2 预期目标

本项目设定的目标如下：

1. 系统能够提供友好的用户界面，使操作人员的工作量最大限度的减少；
2. 系统具有良好的运行效率，能够得到提高生产率的目的；
3. 系统应有良好的可扩充性，可以容易的加入其它系统的应用；
4. 平台的设计具有一定的超前性，灵活性，能够适应企业生产配置的变化；
5. 通过这个项目可以锻炼队伍，提高团队的开发能力和项目管理能力。

# 二、软件工程方法和模型

对于 GSP 药店管理系统，因为涉及到药品管理、销售管理、库存管理、会员管理等核心功能，需求明确、变更可能会比较少；项目的目标和规格在项目启动阶段被明确定义，不太容易跟随时间发生重大调整，故开发过程选用瀑布模型。

瀑布模型分为**制定计划、需求分析、软件设计、程序编写、软件测试和运行维护**六个阶段。

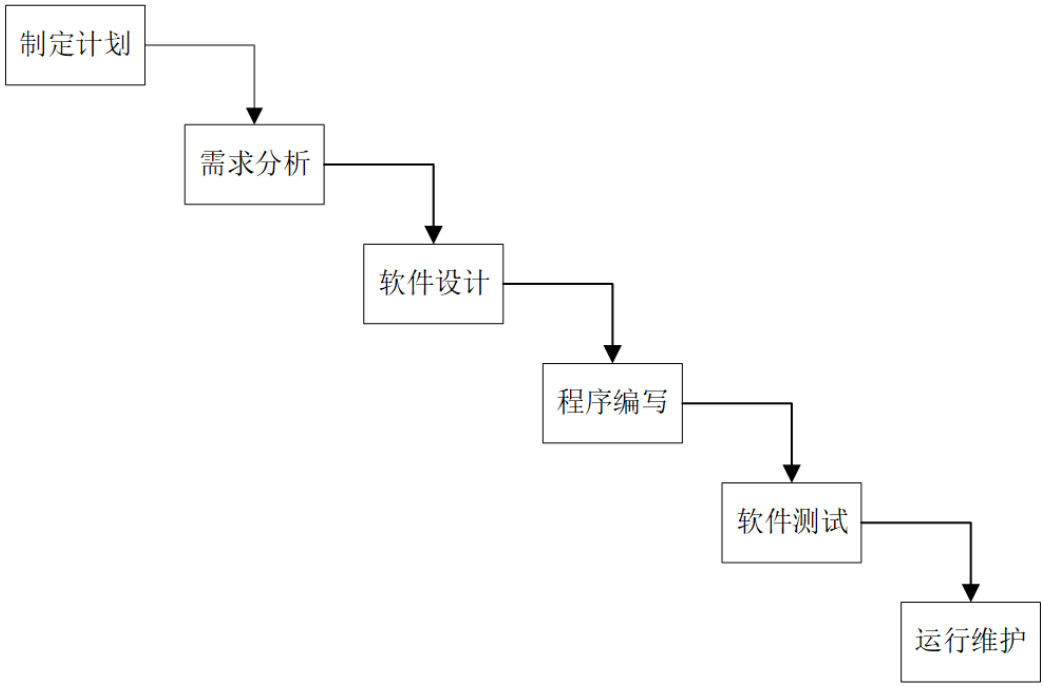



图 2.1 软件过程模型

# 三、进度管理计划

表 3.1 进度管理计划表

| 任务名称 | 开始时间 | 持续时间 | 完成时间 |
|---|---|---|---|
| 制定计划 | 4 月 1 日 | 10 | 4 月 10 日 |
| 需求分析 | 4 月 11 日 | 10 | 4 月 20 日 |
| 系统设计 | 4 月 21 日 | 20 | 5 月 10 日 |
| 程序编写 | 5 月 11 日 | 30 | 6 月 8 日 |
| 软件测试 | 6 月 9 日 | 20 | 6 月 28 日 |
| 运行维护 | 6 月 29 日 | 11 | 7 月 10 日 |

制定计划：根据对未来的项目决策，选择制定包括项目目标、工程标准、项目预算、实施程序及实施方案等的活动。

需求分析：对需求方提出的所有需求，进行详细的分析。这个阶段一般需要和客户反复确认，以保证能充分理解客户需求。最终会形成需求分析文档。

系统设计：根据系统分析的结果，作用系统科学的思想和方法，设计出能最大限度满足所要求的目标（或目的）的新系统的过程。

程序编写：信息从一种形式或格式转换为另一种形式的过程，也称为计算机编程语言的代码简称编码。

软件测试：用精心选择的测试数据进行的机器运行。运行结果与已知结果不一致的，查明原因后加以纠正。

运行维护：一个系统必然会存在一些缺陷和漏洞，把这些漏洞和缺陷之处补上，把一些 Bug 弥补上就相当于系统完善。

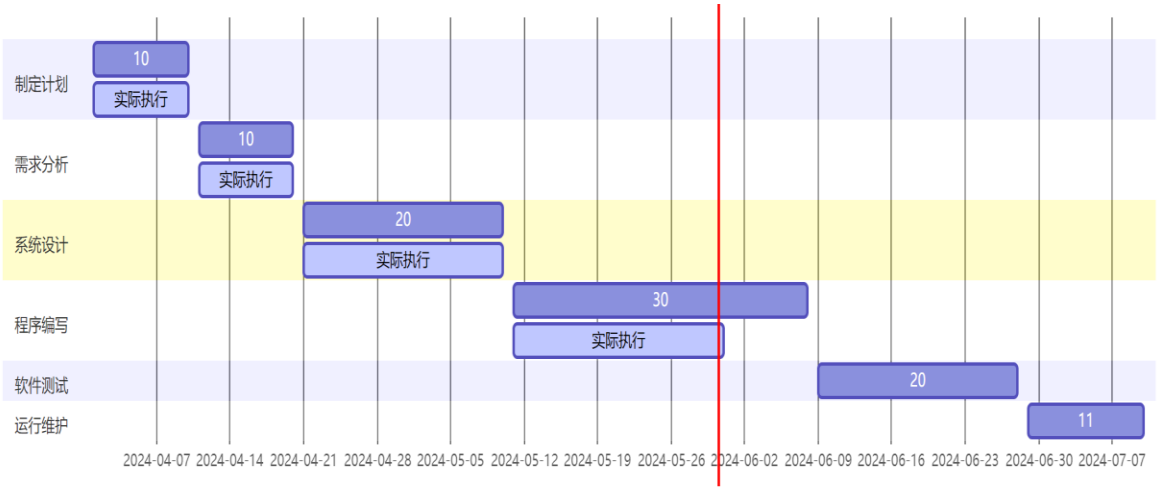



图 3.2 进度管理计划甘特图

# 四、人力资源管理计划

表 4.1 人力资源管理计划表

| 人员 / 工作任务 | 周新斌 | 纪润泽 | 薛薇 | 代萌 | 郑语晖 |
|---|---|---|---|---|---|
| 制定计划 | C | A/R | | | |
| 需求分析 | | | C | A/R | |
| 软件设计 | | | A | C | R |
| 程序编写 | A/R | I | I | C | I |
| 软件测试 | A | R | | | I |
| 运行维护 | A/R | C | I | I | I |

- 人员的专业能力和经验：

纪润泽作为项目计划制定的负责人，有丰富的项目管理经验和优秀的组织和协调能力，能够确保项目进度和质量的同时，有效地管理团队资源。

代萌作为需求分析的负责人，具备深入的业务理解和分析能力，能够为项目提供远见和决策支持。

软件设计不仅需要更为明确的逻辑关系，而且界面的美观程度也在一定程度上具有十分重要的作用，所以该部分由更为细心的女士负责，责任人则由具备了更良好的沟通能力的薛薇担任。

周新斌在编码方面具有专业的技术能力和丰富的经验，能够确保系统设计的合理性和编码的高质量。

测试运行阶段和运行维护的工作则由参与编写代码的主力周新斌负责，因为此项阶段更需要一个对本系统的代码熟悉的人去进行，为系统的稳定性和功能完善性做出贡献。

郑语晖作为具有最多“I”身份的人，在项目中的知情度、沟通和参与是非常重要的，可以在需要时提供反馈和意见，最大限度上避免决策延迟和决策上的障碍。

- 团队协作和沟通能力：

项目团队成员具备良好的团队合作能力和卓越的沟通技巧，能够顺利沟通和协调各个阶段的工作，确保项目各项工作的顺利进行。

具有高度的责任心和积极的工作态度，能够更好地应对项目中的挑战和困难，并及时解决和调整。

# 五、成本管理计划

## 5.1 成本构成

### ➢ 管理成本

表 5.1 管理成本表　　单位：元

| 项目名称 | 单价 | 数量 | 合计 |
|---|---|---|---|
| 工商注册 | 2000 | - | 2000 |
| 水电费 | 500 / 月 | 12 个月 | 6000 |
| 场地租金 | 2000/ 月 | 12 个月 | 24000 |
| 市场推广 | 20000 | - | 20000 |

### ➢ 人力成本

表 5.2 人力成本表　　单位：元

| 项目名称 | 月薪 | 数量 | 月数 | 合计 |
|---|---|---|---|---|
| 开发人员 | 6000 | 2 | 12 | 144000 |
| 测试人员 | 6000 | 1 | 12 | 72000 |
| 管理人员 | 6000 | 2 | 12 | 144000 |

## 5.2 成本基线与监控点

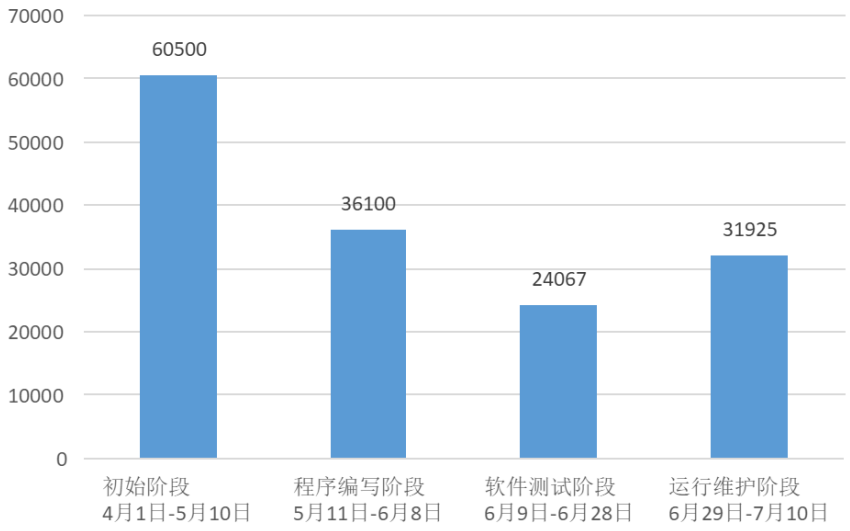



图 5.3 成本基线与监控点　　单位：元

- **启动阶段（4 月 1 日 - 5 月 10 日）**
  - 人力成本：6000 * 5 * 2 * 40 / 60 = 54000 元；
  - 管理成本：2000 + (2000 + 500) * 2 * 40 / 60 = 6500 元；
  - 阶段合计：60500 元。
- **开发阶段（5 月 11 日 - 6 月 8 日）**
  - 人力成本：6000 * 5 * 1 + 6000 * 3 * 1 * 20% = 33600 元；
  - 管理成本：(2000 + 500) * 1 = 2500 元；
  - 阶段合计：36100 元。
- **测试与完善阶段（6 月 9 日 - 6 月 28 日）**
  - 人力成本：6000 * 5 * 1 * 20 / 30 = 20000 元；
  - 管理成本：(2000 + 500) * 1 * 20 / 30= 1667 元；
  - 加班成本：6000 * 3 * 1 * 20 / 30 * 20% = 2400 元；
  - 阶段合计：24067 元。
- **上线阶段（6 月 29 日 - 7 月 10 日）**
  - 市场推广：20000 元；
  - 人力成本：6000 * 5 * 11 / 30 = 11000 元；
  - 管理成本：(2000 + 500) * 11 / 30 = 925 元；
  - 阶段合计：31925 元。

# 六、需求矩阵

需求跟踪矩阵（**RTM**）

项目名称：GSP药店管理系统　　PM：周新斌　　担当：代萌

| No. | 大分類（模块） | 中分類（子模块） | 小分類（功能点） | 详细说明 | 完成情况跟踪 | | | | | | | 担当者 | 责任者 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | SD | PD | DD | COD | UT | IT | ST | | |
| 1 | 药店管理系统 | 管理员管理 | 添加管理员 | 系统生成管理员唯一ID，填写管理员基本信息(职位、姓名、性别、电话号等) | | | | | | | | 代萌 | 周新斌 |
| 2 | | | 删除管理员 | 删除指定管理员 | | | | | | | | 代萌 | 周新斌 |
| 3 | | | 修改管理员信息 | 对指定管理员的基本信息进行修改 | | | | | | | | 代萌 | 周新斌 |
| 4 | | | 查询管理员信息 | 可以查询药店的全部管理员信息和按条件查询管理员 | | | | | | | | 代萌 | 周新斌 |
| 5 | | 药品管理 | 增加 | 进入药品添加功能，系统生成药品内部唯一ID，填写药品基本信息(名称、厂家、规格、产地、类别等) | | | | | | | | 纪润泽 | 周新斌 |
| 6 | | | 修改 | 进入药品信息修改功能，选择药品编号进行信息修改 | | | | | | | | 纪润泽 | 周新斌 |
| 7 | | | 删除 | 进入药品信息删除功能，输入具体药品编号进行删除 | | | | | | | | 纪润泽 | 周新斌 |
| 8 | | | 查询 | 进入查询药品信息功能，可以进行全部查询和条件查询 | | | | | | | | 纪润泽 | 周新斌 |
| 9 | | 供货商管理 | 添加供货商 | 进入添加供货商功能，系统生成供货商唯一ID，填写供货商基本信息(名称、地址、业务员、联系方式等) | | | | | | | | 郑语晖 | 周新斌 |
| 10 | | | 删除供货商 | 进入删除供货商功能，删除指定供货商ID | | | | | | | | 郑语晖 | 周新斌 |
| 11 | | | 修改供货商信息 | 进入更改供货商信息功能，选择特定供货商ID进行信息修改 | | | | | | | | 郑语晖 | 周新斌 |
| 12 | | | 查询供货商信息 | 进入查询供货商信息功能，可以查询全部供货商和按条件查询 | | | | | | | | 郑语晖 | 周新斌 |
| 13 | | 库存管理 | 药品采购 | 药品采购：系统生成唯一采购ID，填写采购基本信息(供货商、药品信息、产品批号、国药准字、生产日期、保质期、采购时间、采购人等) | | | | | | | | 薛薇 | 周新斌 |
| 14 | | | 药品入库 | 药品入库：系统生成唯一入库ID，填写入库基本信息(供货商、药品信息、产品批号、国药准字、生产日期、保质期、入库时间、入库人等) | | | | | | | | 薛薇 | 周新斌 |
| 15 | | | 药品出售 | 出售药品：系统生成唯一出售ID，填写出售基本信息(药品基本信息、产品批号、国药准字、生产日期、保质期、出售时间、出售人、顾客信息等) | | | | | | | | 薛薇 | 周新斌 |
| 16 | | | 库存盘点 | 可以盘点全部药品库存，也可以按条件盘点药品，比如名称、类型、厂家、产地、保质期等 | | | | | | | | 薛薇 | 周新斌 |
| 17 | | | 出售记录查询 | 可以查询全部出售记录，也可以按出售信息的各项元素进行条件查询 | | | | | | | | 薛薇 | 周新斌 |
| 18 | | | 采购记录查询 | 可以查询全部采购记录，也可以按采购信息的各项元素进行条件查询 | | | | | | | | 薛薇 | 周新斌 |
| 19 | | 顾客管理 | 添加顾客 | 填写顾客基本信息(姓名、性别、联系方式等)，系统唯顾客生成唯一顾客ID，系统添加顾客信息 | | | | | | | | 周新斌 | 周新斌 |
| 20 | | | 删除顾客 | 根据顾客ID删除指定顾客的信息 | | | | | | | | 周新斌 | 周新斌 |
| 21 | | | 修改顾客信息 | 根据顾客ID进行顾客信息修改 | | | | | | | | 周新斌 | 周新斌 |
| 22 | | | 查询顾客信息 | 可以查询全部顾客信息，也可以按顾客信息的各项元素进行条件查询 | | | | | | | | 周新斌 | 周新斌 |

图 6.1 需求矩阵

# 七、系统流程图

## 7.1 管理员管理

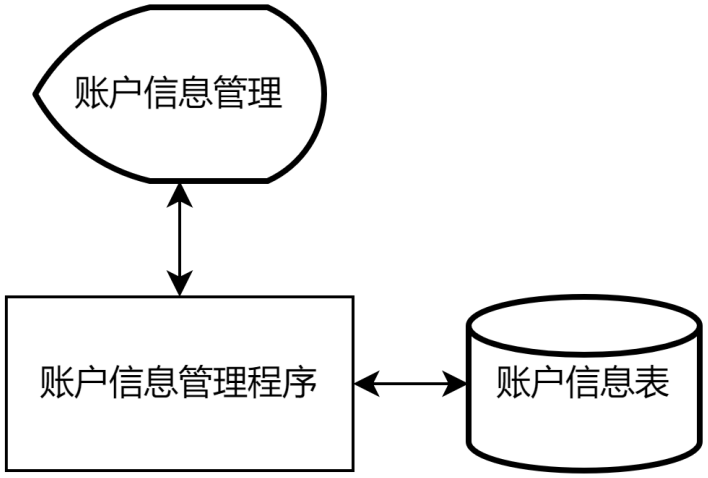
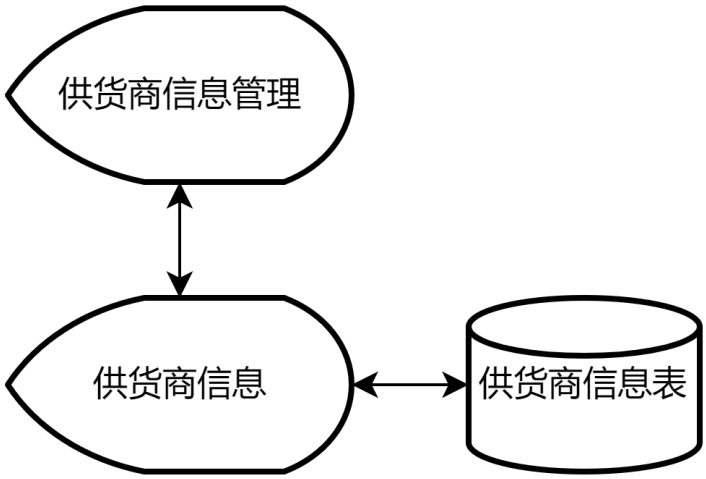



图 7.1 账户信息管理

## 7.2 供货商管理




7.2 供货商管理

## 7.3 药品信息管理

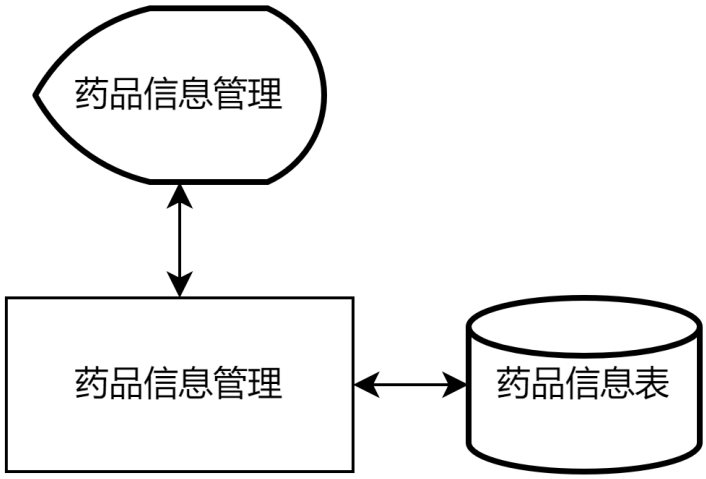



图 7.3 药品信息管理

## 7.4 库存管理

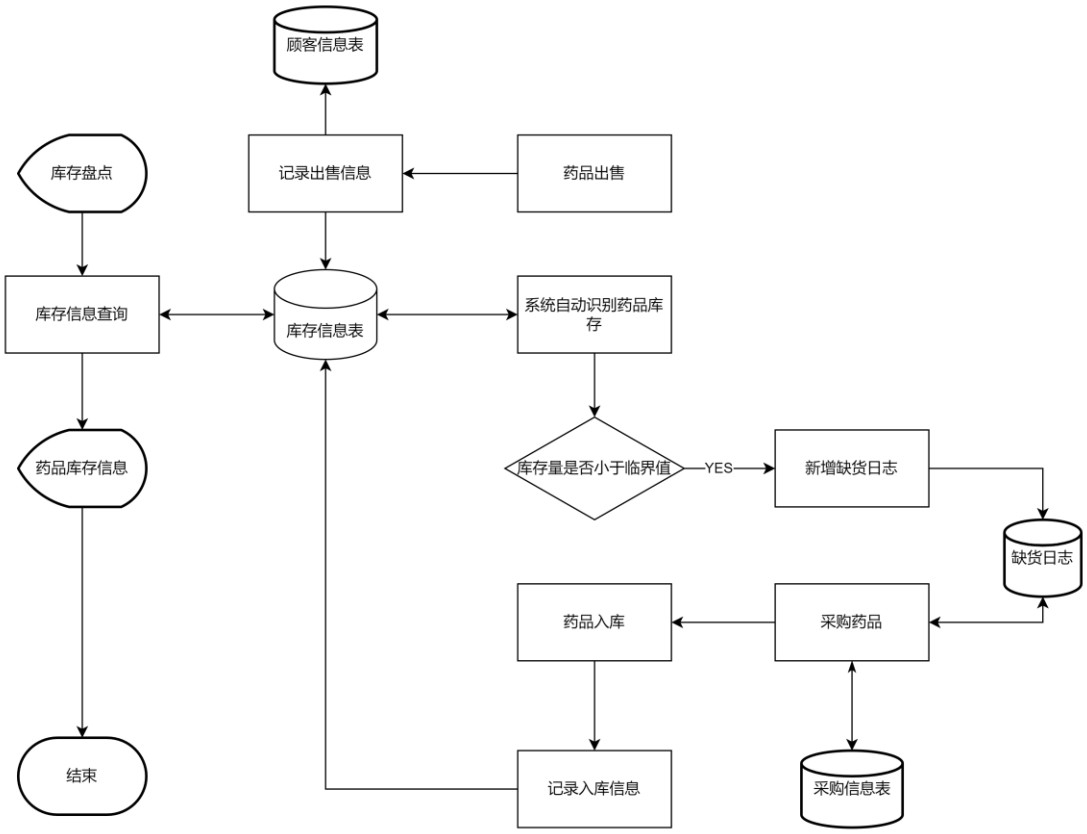



图 7.4 库存管理

## 7.5 顾客管理

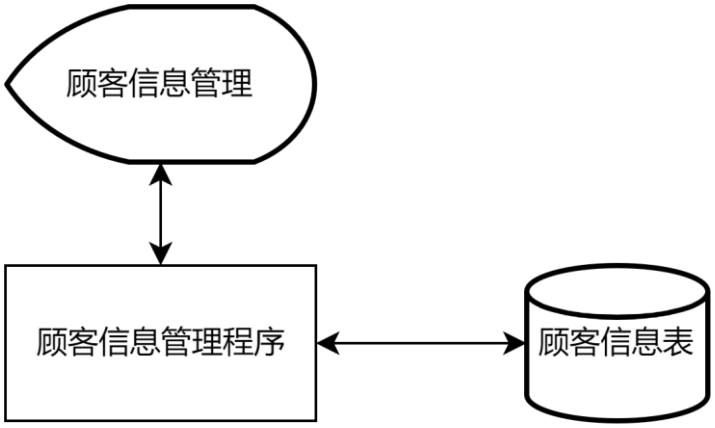



图 7.5 顾客管理

# 八、数据字典

## 8.1 管理员数据字典

表名：员工信息表

| 字段标识 | 字段名 | 类型 | 递增 | 约束 |
|---|---|---|---|---|
| admin_id | 员工 ID | smallint | 是 | |
| admin_name | 员工名称 | varchar(16) | 否 | |
| admin_type | 员工类别 | varchar(12) | 否 | ^(质量负责人\|销售管理员\|库存管理员)$ |
| admin_sex | 员工性别 | varchar(36) | 否 | ^(男\|女)$ |
| admin_phone | 员工手机 | varchar(12) | 否 | ^1[3-9]\d{9}$ |
| admin_pwd | 员工密码 | varchar(36) | 否 | ^(?=.*[A-Za-z])(?=.*\d)[A-Za-z\d]{8,}$ |

## 8.2 供货商数据字典

表名：供货商信息表

| 字段标识 | 字段名 | 类型 | 递增 | 约束 |
|---|---|---|---|---|
| supplier_id | 供货商 ID | smallint | 是 | |
| supplier_name | 供货商名 | varchar(16) | 否 | |
| supplier_address | 供货商地址 | varchar(64) | 否 | |
| supplier_phone | 供货商电话 | varchar(16) | 否 | [0-9]{4}-[0-9]{7} or [0-9]{3}-[0-9]{8} |

## 8.3 药品数据字典

表名：药品信息表

| 字段标识 | 字段名 | 类型 | 递增 | 约束 |
|---|---|---|---|---|
| drug_id | 药品 ID | smallint | 是 | |
| drug_name | 药品名称 | varchar(16) | 否 | |
| drug_maker | 药品厂家 | varchar(36) | 否 | |
| drug_expiration | 药品保质期 | varchar(12) | 否 | \[[1-9][0-9]{0,1}个月\] |
| drug_type | 药品类别 | varchar(12) | 否 | |
| production_batch | 生产批号 | varchar(10) | 否 | |
| drug_spec | 药品规格 | varchar(12) | 否 | |

## 8.4 库存数据字典

表名：库存信息表

| 字段标识 | 字段名 | 类型 | 递增 | 约束 |
|---|---|---|---|---|
| stock_id | 库存 ID | smallint | 是 | |
| drug_id | 药品 ID | smallint | | 外键 |
| drug_name | 药品名称 | varchar(16) | | |
| drug_maker | 药品厂家 | varchar(36) | | |
| drug_expiration | 药品保质期 | varchar(12) | | \[[1-9][0-9]{0,1}个月\] |
| production_data | 生产日期 | date | | |
| production_batch | 生产批号 | varchar(10) | | |
| stock_purchase_price | 采购价格 | decimal(9,2) | | |
| stock_sale_price | 出售价格 | decimal(9,2) | | |
| stock_purchase_quantity | 入库数量 | smallint | | ^[1-9]\d*$ |
| stock_date | 入库日期 | date | | |

## 8.5 顾客数据字典

表名：顾客信息表

| 字段标识 | 字段名 | 类型 | 递增 | 约束 |
|---|---|---|---|---|
| customer_id | 顾客 ID | smallint | 是 | |
| customer_name | 顾客名称 | varchar(16) | 否 | |
| customer_reward | 顾客积分 | decimal(9,2) | 否 | |
| customer_sex | 顾客性别 | varchar(36) | 否 | ^(男\|女)$ |
| customer_phone | 顾客手机 | varchar(12) | 否 | ^1[3-9]\d{9}$ |

## 8.6 采购数据字典

表名：采购信息表

| 字段标识 | 字段名 | 类型 | 递增 | 约束 |
|---|---|---|---|---|
| purchase_id | 库存 ID | smallint | 是 | |
| supplier_id | 供货商 ID | smallint | | 外键 |
| drug_id | 药品 ID | smallint | | 外键 |
| drug_name | 药品名称 | varchar(16) | | |
| drug_maker | 药品厂家 | varchar(36) | | |
| purchase_price | 采购价格 | decimal(9,2) | | |
| purchase_sale_price | 出售价格 | decimal(9,2) | | |
| purchase_quantity | 采购数量 | smallint | | ^[1-9]\d*$ |
| purchase_date | 采购日期 | date | | |

# 九、系统建模

## 9.1 活动图

### 9.1.1 售药活动图

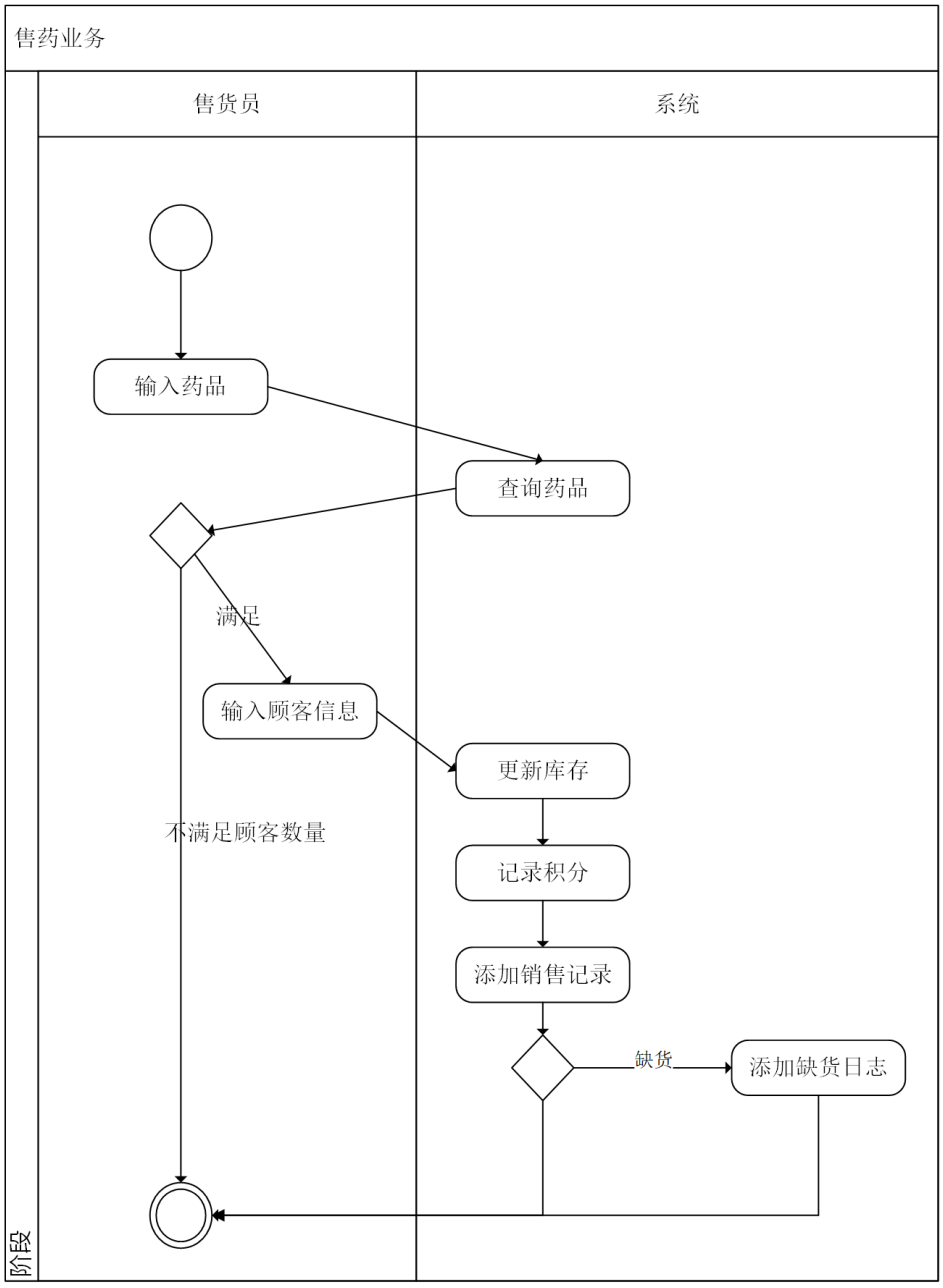



图 9.1 售药活动图

### 9.1.2 采购活动图

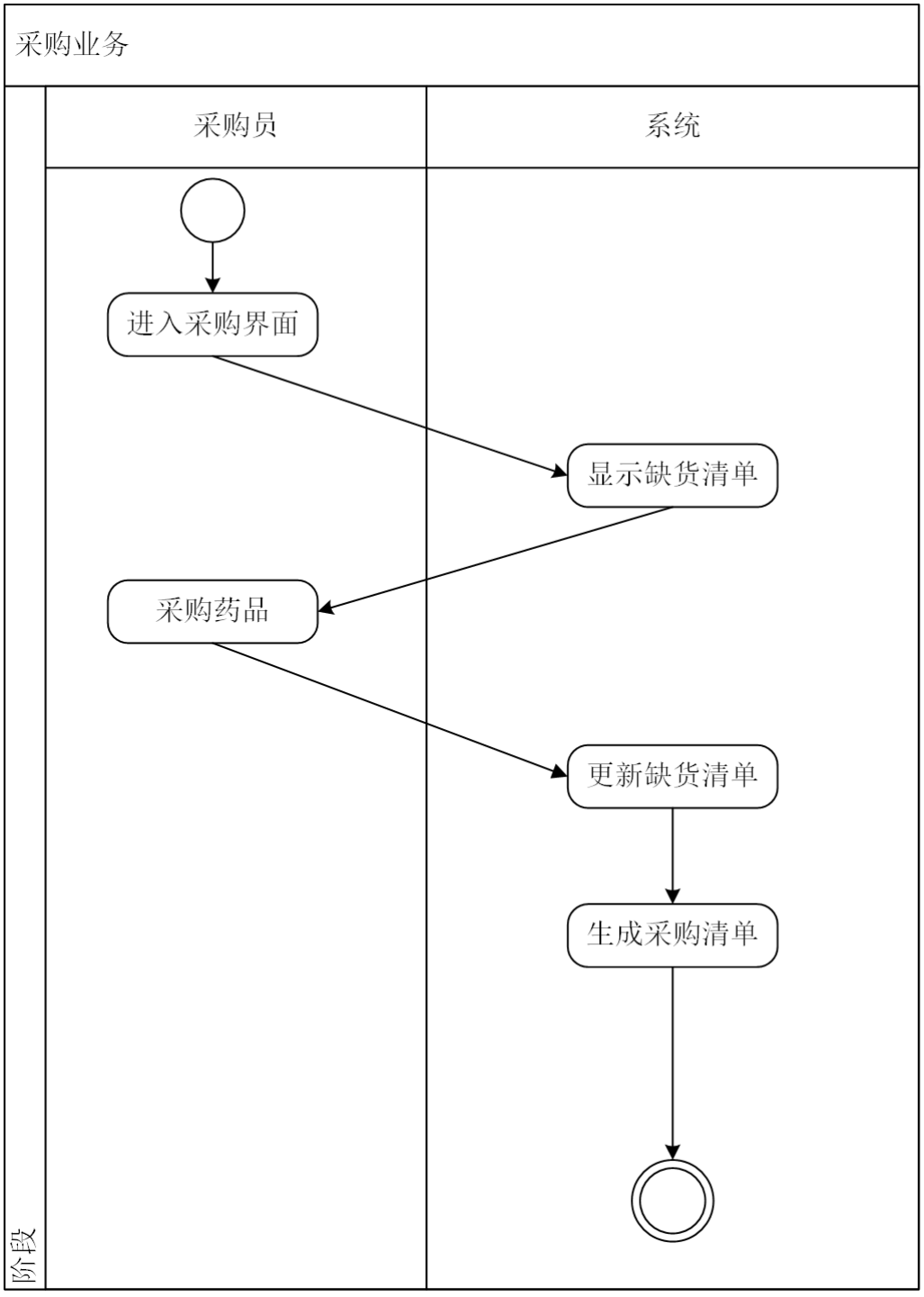



图 9.2 采购活动图

## 9.2 时序图

### 9.2.1 售药

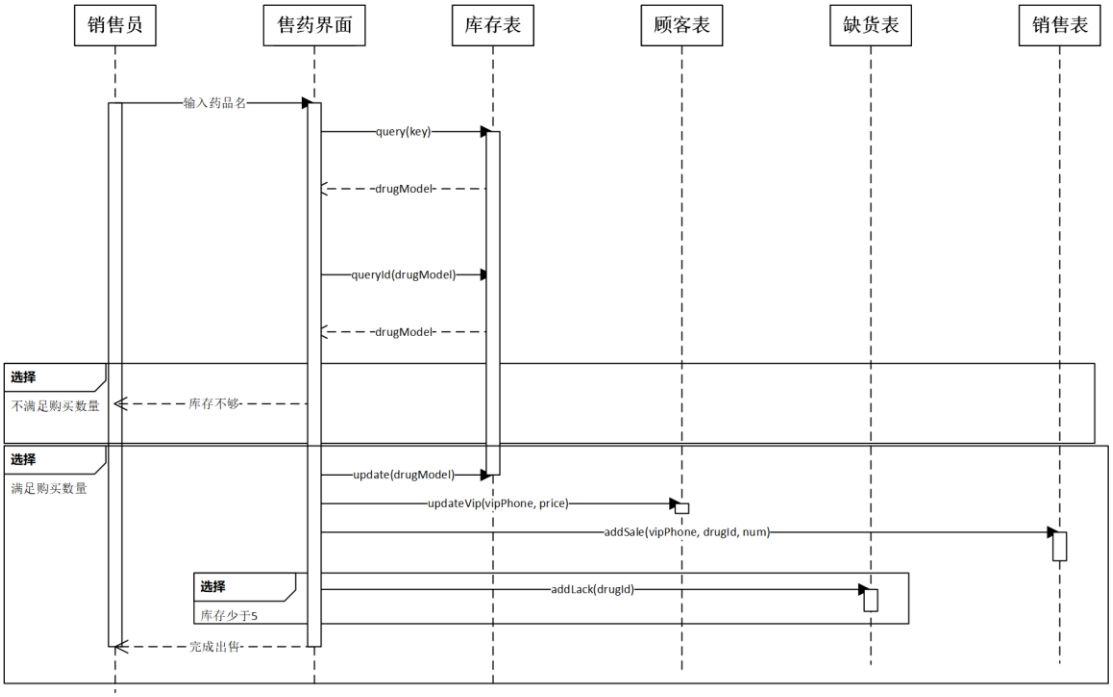
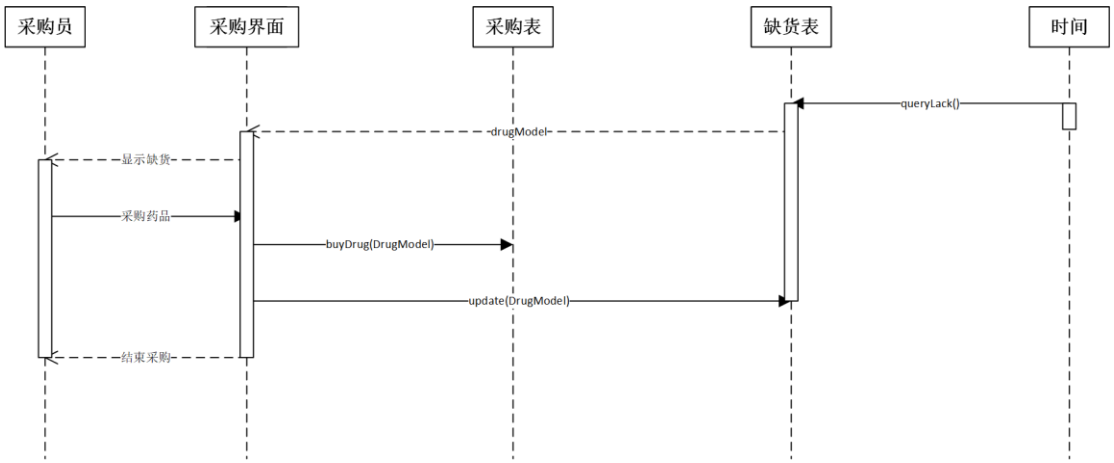



图 9.3 售药时序图

## 9.2.2 采购入库




9.4 采购入库时序图

## 9.3 类图

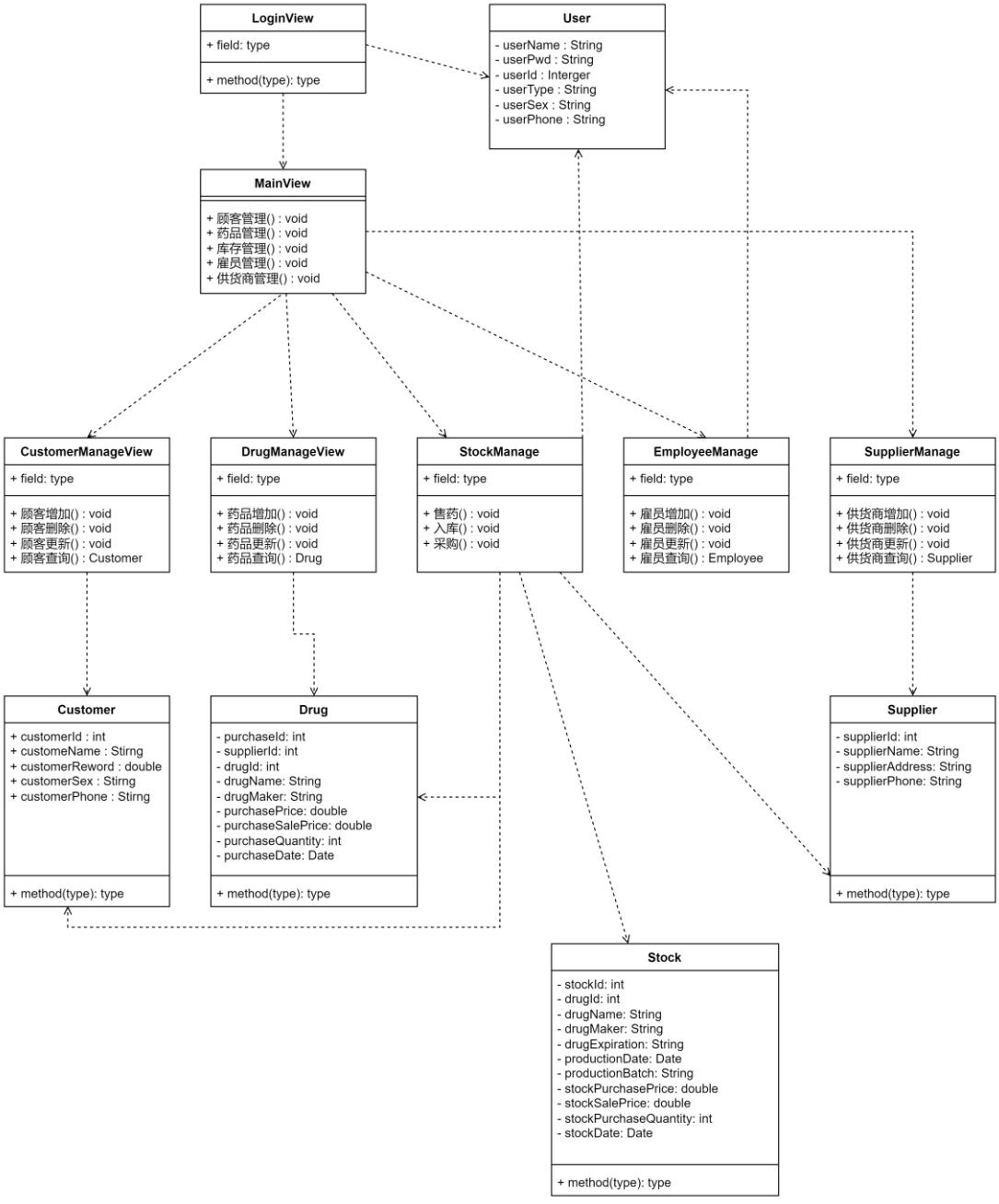

LoginView
+ field: type
+ method(type): type
User
- userName : String
- userPwd : String
- userId : Interger
- userType : String
- userSex : String
- userPhone : String
MainView
+ 顾客管理() : void
+ 药品管理() : void
+ 库存管理() : void
+ 雇员管理() : void
+ 供货商管理() : void
CustomerManageView
+ field: type
+ 顾客增加() : void
+ 顾客删除() : void
+ 顾客更新() : void
+ 顾客查询() : Customer
DrugManageView
+ field: type
+ 药品增加() : void
+ 药品删除() : void
+ 药品更新() : void
+ 药品查询() : Drug
StockManage
+ field: type
+ 售药() : void
+ 入库() : void
+ 采购() : void
EmployeeManage
+ field: type
+ 雇员增加() : void
+ 雇员删除() : void
+ 雇员更新() : void
+ 雇员查询() : Employee
SupplierManage
+ field: type
+ 供货商增加() : void
+ 供货商删除() : void
+ 供货商更新() : void
+ 供货商查询() : Supplier
Customer
+ customerId : int
+ customeName : Stirng
+ customerReword : double
+ customerSex : Stirng
+ customerPhone : Stirng
+ method(type): type
Drug
- purchaseId: int
- supplierId: int
- drugId: int
- drugName: String
- drugMaker: String
- purchasePrice: double
- purchaseSalePrice: double
- purchaseQuantity: int
- purchaseDate: Date
+ method(type): type
Supplier
- supplierId: int
- supplierName: String
- supplierAddress: String
- supplierPhone: String
+ method(type): type
Stock
- stockId: int
- drugId: int
- drugName: String
- drugMaker: String
- drugExpiration: String
- productionDate: Date
- productionBatch: String
- stockPurchasePrice: double
- stockSalePrice: double
- stockPurchaseQuantity: int
- stockDate: Date
+ method(type): type

图 9.5 类图

# 9.4 状态图

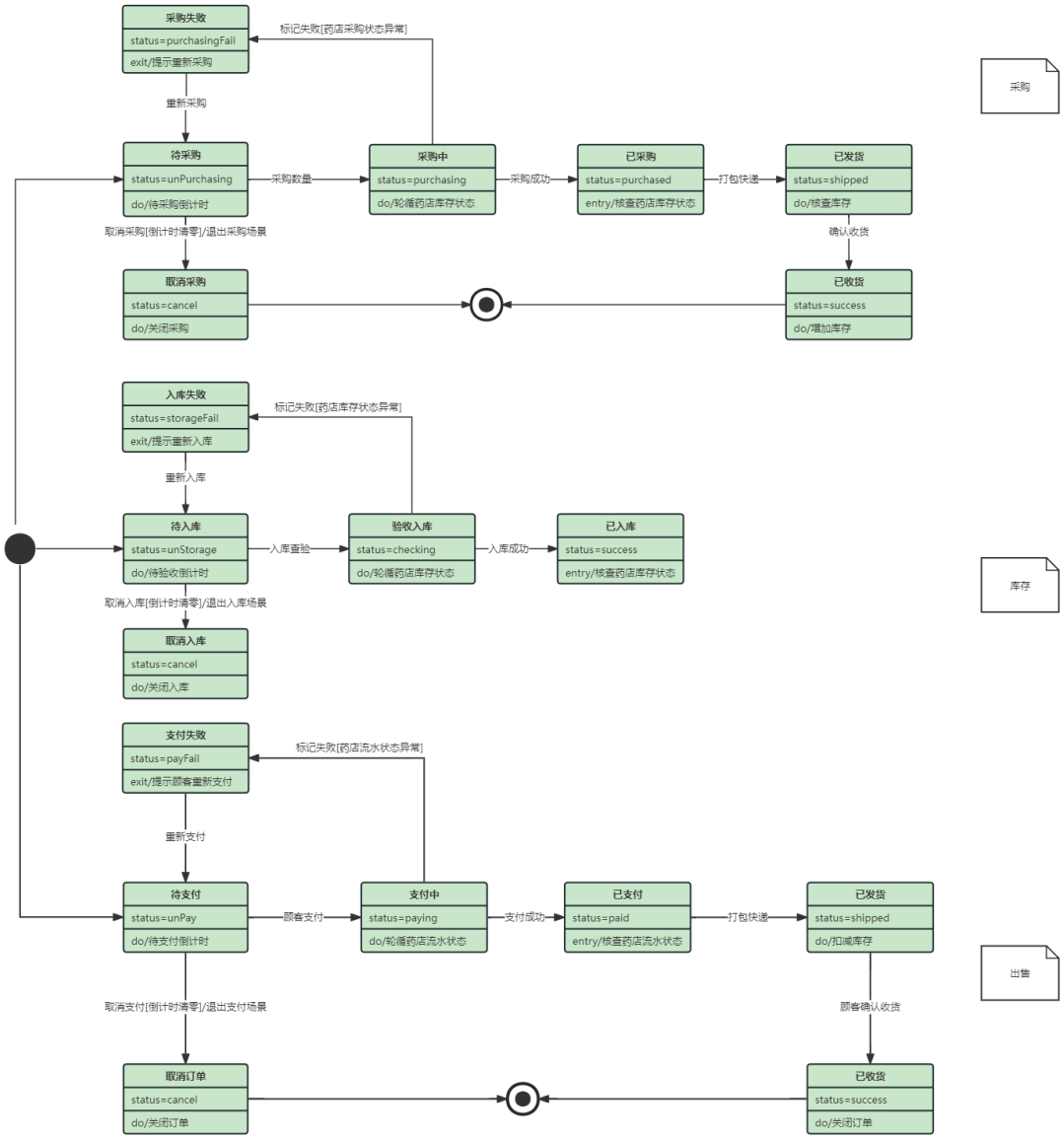



图 9.6 状态图

# 十、系统体系架构

## 10.1 体系架构图

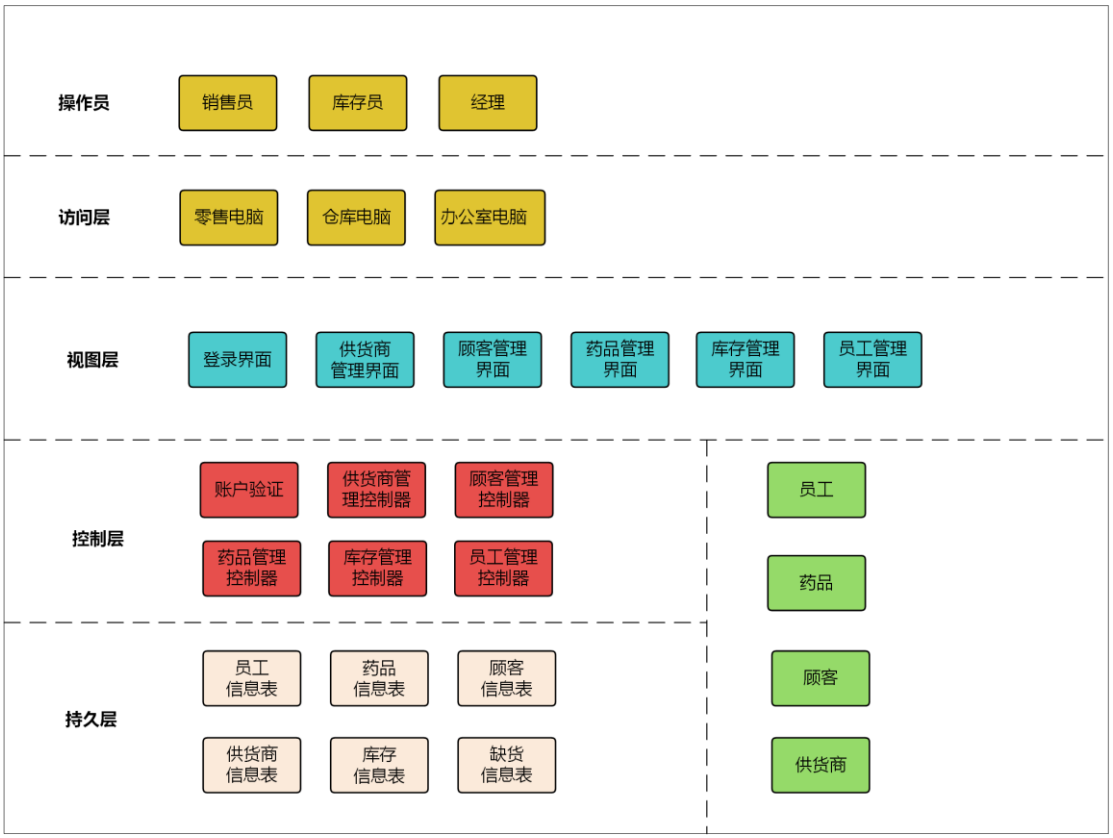



图 10.1 系统体系架构图

1. 整体采用 2 层 C/S 架构。GSP 药店管理系统业务简单，业务范围集中在某个药店内部，故采用 2 层 C/S 架构。业务操作在本地客户端执行，药店可以根据具体的本地需求定制和优化其界面和功能。通过向云端数据库服务器发送 SQL 语句进行读写操作。业务数据存放在云服务器可提高数据安全性和可恢复性。
2. 软件内部采用 MVC 架构。MVC 架构通过分离数据模型（Model）、用户界面（View）和控制逻辑（Controller），实现了关注点的清晰划分。这种分离有助于减少组件间的依赖，使得开发、测试和维护各个部分变得更加简单和高效。
3. 数据库映射在药店管理系统中使用**直接在线映射**实现，这种系统需求明确且功能相对简单，可以通过 SQL 语句直接与数据库进行交互。

## 10.2 核心代码实现

### 10.2.1 DBUtil

在多个地方创建和配置数据库连接会增加系统的复杂性和运行开销。单例模式避免了这种情况，因为整个应用程序生命周期中只创建一个 `DBUtil` 实例。这减少了重复代码和对象创建所带来的开销，也使得对数据库工具类的修改更加集中和可控。

```
public class DBUtil {
    private static DBUtil instance = new DBUtil();
    private static String url = "jdbc:mysql://127.0.0.1:3306/javalab";
    private static String username = "root";
    private static String password = "";
    // 私有构造函数防止其他类实例化
    private DBUtil() {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
        } catch (ClassNotFoundException e) {
            System.err.println("未找到驱动: " + e.getMessage());
            System.exit(1);
        }
    }
    // 公共静态方法获取实例
    public static DBUtil getInstance() {
        return instance;
    }
    // 获取数据库连接的方法
    public Connection getConnection() {
        try {
            return DriverManager.getConnection(url, username, password);
        } catch (SQLException e) {
            System.err.println("SQL 错误: " + e.getMessage());
            System.exit(1);
            return null; // 此行代码永远不会执行，但为了编译需要保留。
        }
    }
```

```
    // 静态方法关闭连接
    public static void closeConnection(Connection connection) {
        if (connection != null) {
            try {
                connection.close();
            } catch (SQLException e) {
                System.err.println("关闭连接失败: " + e.getMessage());
                System.exit(1);
            }
        }
    }
    // 静态方法关闭 PreparedStatement
    public static void closePs(PreparedStatement ps) {
        if (ps != null) {
            try {
                ps.close();
            } catch (SQLException e) {
                System.err.println("关闭 PreparedStatement 失败: " +
e.getMessage());
                System.exit(1);
            }
        }
    }

    // 静态方法关闭 ResultSet
    public static void closeResultSet(ResultSet resultSet) {
        if (resultSet != null) {
            try {
                resultSet.close();
            } catch (SQLException e) {
                System.err.println("关闭 ResultSet 失败: " + e.getMessage());
                System.exit(1);
            }
        }
    }
}
```

## 10.2.2 SalePartListener

```
public class SalePartListener implements ActionListener {
    private SalePartView salePartView;
    private String key;

    // 构造函数，初始化销售部分视图
    public SalePartListener(SalePartView salePartView) {
        this.salePartView = salePartView;
    }

    // 实现 ActionListener 接口的方法，根据用户操作触发相应的处理
    @Override
    public void actionPerformed(ActionEvent e) {
        if(e.getSource() == salePartView.getQueryProductButton()) {
            performProductQuery();  // 执行产品查询
        } else if(e.getSource() == salePartView.getSettlementButton()) {
            handleProductSettlement();  // 处理产品结算
        }
    }

    // 执行产品查询，显示查询结果
    private void performProductQuery() {
        key = salePartView.getQueryProductText().getText().trim();
        if(key.isEmpty()) {
            return;
        }
        MyTableModel  myTableModel  =  SalePartController.query(key,  new
MyTableModel());
        salePartView.getMyJTable().setMyTableModel(myTableModel);
    }

    // 处理产品结算，进行库存检查和计价
    private void handleProductSettlement() {
        String idStr = salePartView.getProductIDText().getText().trim();
        String numStr = salePartView.getSealNumberText().getText().trim();
```

```
        String vipPhone = salePartView.getVipPhoneText().getText.trim();

        if (idStr.isEmpty() || numStr.isEmpty()) {
            MyJOptionPane.showMessageDialog(null, "请输入商品 ID 和数量", "
错误");
            return;
        }

        try {
            int id = Integer.parseInt(idStr);
            int num = Integer.parseInt(numStr);
            settleTransaction(id, num);  // 结算交易
        } catch (NumberFormatException ex) {
            MyJOptionPane.showMessageDialog(null,"商品ID和数量必须是有效数
字", "错误");
        }
    }

    // 结算交易，更新库存和显示新价格
    private void settleTransaction(int id, int num) {
        ProductionModel temp = new ProductionModel();
        temp.setId(id);
        ProductionModel productionModel = SalePartController.idQuery(temp);
        if (productionModel == null) {
            MyJOptionPane.showMessageDialog(null, "未找到指定的商品", "错误
");
            return;
        }

        if (productionModel.getPurchaseQuantity() < num) {
            MyJOptionPane.showMessageDialog(null, "库存不足", "提示");
            return;
        }

        double totalPrice = num * productionModel.getSalePrice();
        salePartView.getTotalPriceText().setText("  应  收  :  "  +
```

```
formatPrice(totalPrice));
        // 更新库存

productionModel.setPurchaseQuantity(productionModel.getPurchaseQuantity
() - num);
        SalePartController.update(productionModel);
        // 添加销售记录
        SalePartControllere.addSale(id, num, vipPhone);
        // 更新积分
        SalePartController.update(vipPhone, totalPrice);
        // 更新查询列表
        refreshQuery();
        // 添加缺货记录
        if (productionModel.getPurchaseQuantity() - num < 5) {
            SalePartController.addLack(id);  // 添加缺货记录
        }
    }

    // 格式化价格显示
    private String formatPrice(double price) {
        return new DecimalFormat("#.##").format(price);
    }

    // 刷新查询显示，用于更新界面上的表格数据
    private void refreshQuery() {
        MyTableModel  myTableModel  =  SalePartController.query(key,  new
MyTableModel());
        salePartView.getMyJTable().setMyTableModel(myTableModel);
    }
}
```

## 10.2.3 SalePartContriller

```
public class SalePartController {
    //模糊查询关键词，不显示进价
    public static MyTableModel query(String key, MyTableModel myTableModel)
{
        String sql = "SELECT id, name, factory, address, productionDate, "
+
                "expirationDate, purchaseQuantity, salePrice" +
                " FROM product WHERE " +
                "name LIKE '%" + key + "%' OR " +
                "factory LIKE '%" + key + "%' OR " +
                "address LIKE '%" + key + "%'";

        Connection conn = null;
        PreparedStatement ps = null;
        ResultSet rs = null;
        Vector<Vector<Object>> data = new Vector<>();

        try {
            conn = DBUtil.getInstance().getConnection();
            ps = conn.prepareStatement(sql);
            rs = ps.executeQuery();
            while(rs.next()) {
                Vector<Object> dt = new Vector<>();
                for (int i = 1; i <= 8; i++) {
                    dt.addElement(rs.getObject(i));
                }
                data.addElement(dt);
            }
            //向实体对象添加数据
            myTableModel.setDate(data);
            //向实体对象添加表头
            Vector<Object> columns = new Vector<>();
            columns.addElement("id");
```

```
            columns.addElement("药品名称");
            columns.addElement("生产厂家");
            columns.addElement("生产地址");
            columns.addElement("生产日期");
            columns.addElement("有效期");
            columns.addElement("库存数量");
            columns.addElement("售价");
            myTableModel.setColumns(columns);
            //表格实体设置表头和数据
            myTableModel.setDataVector(data, columns);
            return myTableModel;
        } catch (SQLException e) {
            e.printStackTrace();
        } finally {
            DBUtil.closeResultSet(rs);
            DBUtil.closePs(ps);
            DBUtil.closeConnection(conn);
        }
        return null;
    }

    //根据 id 查询商品，返回该 id 商品的库存和售价，返回类型为 Product
    public static ProductionModel idQuery(ProductionModel productionModel)
{
        int qid = productionModel.getId();
        String sql = "select purchaseQuantity, salePrice from product where
" +
                "id = " + qid + "";
        Connection conn = null;
        PreparedStatement ps = null;
        ResultSet rs = null;

        try {
            conn = DBUtil.getInstance().getConnection();
            ps = conn.prepareStatement(sql);
            rs = ps.executeQuery();
```

```
        while(rs.next())
        {
            productionModel.setPurchaseQuantity(rs.getInt(1));
            productionModel.setSalePrice(rs.getDouble(2));
        }
        return productionModel;
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        DBUtil.closeResultSet(rs);
        DBUtil.closePs(ps);
        DBUtil.closeConnection(conn);
    }
    return null;
}
//更新库存
public static void update(ProductionModel productionModel) {
    String sql = "update product set purchaseQuantity = " +
            productionModel.getPurchaseQuantity() + " " +
            "where id = " + productionModel.getId() + "";
    Connection conn = null;
    PreparedStatement ps = null;

    try {
        conn = DBUtil.getInstance().getConnection();
        ps = conn.prepareStatement(sql);
        ps.executeUpdate();
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        DBUtil.closePs(ps);
        DBUtil.closeConnection(conn);
    }
}
```

```
    // 添加缺货日志
    public static void addLack(int id)
    {
        String sql = "insert into lackLog (lackId, lackDate) values (?, ?)";
        Connection conn = null;
        PreparedStatement ps = null;
        try {

            conn = DBUtil.getInstance().getConnection();
            ps = conn.prepareStatement(sql);
            ps.setInt(1, id);
            ps.setDate(2, new Date(System.currentTimeMillis()));
            ps.executeUpdate();
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            DBUtil.closePs(ps);
            DBUtil.closeConnection(conn);
        }
    }
}
```

# 十一、参考文献

[1]陈志坚.新版 GSP 认证下药品零售企业的发展趋势探讨[J].中文科技期刊数据库(全文版)经济管理, 2022(4):3.

[2]夏婧,牛强强.基于 VB 的进销存管理系统设计和实现[J]. 2022(11).

[3]舒炼, 祝悦, 张嘉杨. 药事管理与法规. 重庆大学电子音像出版社有限公司; 2021 Jul 1.

[4]张守钗. 单体药店两轮 GSP 认证检查缺陷项目比较分析. 中国药事. 2021 Nov 8;35(8):915-22.

[5]符光量, 宋荣斌, 张维安, & 李卓襄. (2022). 广东省药品经营企业 GSP 认证现场检查缺陷项目分析. *中国药事*, *35*(1), 164-169.