详细设计

1. 界面设计
   1. 用户登录注册模块

图形用户界面, 应用程序

描述已自动生成

图 用户登录界面

图形用户界面, 应用程序

描述已自动生成

图 用户注册界面

1.2 商品管理模块

图形用户界面, 应用程序

描述已自动生成

图 新增商品界面

图形用户界面, 文本, 应用程序

描述已自动生成

图 商品分类查看界面

图形用户界面, 应用程序, Word

描述已自动生成

图 商品查询界面

1.3 库存管理模块

表格

描述已自动生成

图 库存查看界面

1.4 订单管理模块

图形用户界面, 应用程序

描述已自动生成

图 订单提交界面

表格

描述已自动生成

图 订单历史查询界面

1.5 供应商管理模块

表格

描述已自动生成

图 供应商添加与查看界面

1. 主要模块分析

2.1 商品信息管理模块的PAD图及伪码算法

图片包含 信件

描述已自动生成

图 商品信息管理模块

|  |
| --- |
| // 商品信息结构体定义  struct Commodity {  int categoryID;  char categoryName[20];  // 其他商品信息字段  };  // 函数：商品信息管理模块  void manageCommodityInfo(struct Commodity \*commodity) {  // 输入：用户输入  // （假设用户已经通过其他方式输入了商品信息）  // 处理：  // 1. 从用户输入中获取商品信息  // （假设用户输入已存储在commodity结构体中）  // 2. 操作数据库，插入或更新商品信息  // （假设已有数据库操作函数dbInsertOrUpdateCommodity）  dbInsertOrUpdateCommodity(commodity);  // 输出：商品基本信息  // （这里可以选择返回商品信息或进行其他操作）  } |

2.2

文本, 信件

描述已自动生成

图 库存管理模块

|  |
| --- |
| // 库存信息结构体定义  struct Inventory {  int inventoryProductID;  int categoryID;  int stockQuantity;  char stockStatus[10];  // 其他库存信息字段  };  // 函数：库存管理模块  void manageInventory(struct Commodity \*commodity, struct Inventory \*inventory) {  // 输入：商品信息管理模块输出的商品基本信息  // （假设商品信息已经存储在commodity结构体中）  // 处理：  // 1. 更新库存信息，根据商品信息更新库存数量和状态  // （这里可以根据实际情况定义库存的更新逻辑）  inventory->categoryID = commodity->categoryID;  // 更新其他库存信息字段...  // 2. 操作数据库，更新库存信息  // （假设已有数据库操作函数dbUpdateInventory）  dbUpdateInventory(inventory);  // 输出：商品库存信息  // （这里可以选择返回库存信息或进行其他操作）  } |

2.3

文本, 信件

描述已自动生成

图 订单管理模块

|  |
| --- |
| // 订单信息结构体定义  struct Order {  int orderID;  int userID;  double totalAmount;  // 其他订单信息字段  };  // 函数：订单管理模块  void manageOrder(struct Commodity \*commodity, struct Order \*order) {  // 输入：商品信息管理模块输出的商品基本信息  // （假设商品信息已经存储在commodity结构体中）  // 处理：  // 1. 处理订单信息，根据商品信息生成订单  // （这里可以根据实际情况定义订单的生成逻辑）  order->userID = getUserID(); // 假设有获取用户ID的函数  order->totalAmount = calculateTotalAmount(commodity); // 假设有计算总金额的函数  // 其他订单信息生成逻辑...  // 2. 操作数据库，插入订单信息  // （假设已有数据库操作函数dbInsertOrder）  dbInsertOrder(order);  // 输出：订单信息  // （这里可以选择返回订单信息或进行其他操作）  } |