《数据库系统原理》课程设计

系统实现报告

题目名称： “学院路”外卖订购配送系统

学号及姓名：\_\_\_\_\_14061043\_\_\_冯岩\_\_\_\_\_\_\_\_\_

\_ 14061056\_\_\_杨子琛\_\_\_\_\_\_\_\_

2016年 11 月 26 日

**总目录**

[一．系统功能需求分析 5](#_Toc21165)

[二．系统功能结构设计 6](#_Toc20120)

[注册验证与权限管理 7](#_Toc13617)

[商品的选购与订单提交 7](#_Toc17742)

[商家后台管理 7](#_Toc4206)

[快递员后台管理 8](#_Toc10007)

[反馈评分管理 8](#_Toc31666)

[三、数据库基本表的定义 8](#_Toc29198)

[用户表(users) 9](#_Toc14502)

[订单表(orders) 10](#_Toc5446)

[订单详细表(orderitems) 10](#_Toc9689)

[优惠券表(coupons) 11](#_Toc6588)

[商家表(shops) 11](#_Toc17007)

[食品表(cuisine) 12](#_Toc7086)

[快递员表(deliverymen) 13](#_Toc13467)

[四、触发器的定义与实现 13](#_Toc24522)

[用户信息同步触发器(sync\_userInfo\_trigger) 13](#_Toc4232)

[订单完结信息同步触发器(sync\_order\_complete\_info\_trigger) 14](#_Toc27358)

[用户头像信息同步触发器(sync\_userPhoto\_trigger) 14](#_Toc19417)

[用户权限验证触发器(check\_user\_trigger) 15](#_Toc30695)

[订单状态控制触发器(check\_orderStatus\_trigger) 15](#_Toc27118)

[订单总价同步(插入操作)触发器(sync\_order\_insert) 15](#_Toc3983)

[订单总价同步(更新操作)触发器(sync\_order\_update) 16](#_Toc24172)

[订单总价同步(删除操作)触发器(sync\_order\_delete) 16](#_Toc24748)

[五、存储过程的定义与实现 17](#_Toc9941)

[添加食品(addFood()) 17](#_Toc24910)

[删除食品(delFood()) 17](#_Toc28663)

[六、系统实现结果 18](#_Toc6899)

[注册验证与权限管理 18](#_Toc8581)

[商品的选购与订单提交 20](#_Toc4588)

[商家后台管理 22](#_Toc9301)

[快递员后台管理 24](#_Toc12909)

[反馈评分管理 24](#_Toc10134)

[七、总结 25](#_Toc23418)

**图目录**

[二．系统功能结构设计 6](#_Toc5373)

[图2.1 系统功能结构设计图 6](#_Toc18572)

[六、系统实现结果 18](#_Toc17315)

[注册验证与权限管理 18](#_Toc25763)

[图6.1 用户注册界面 19](#_Toc10625)

[图6.2 用户注册界面（点击注册并验证后） 19](#_Toc12669)

[图6.3 验证邮箱连接 20](#_Toc27444)

[商品的选购与订单提交 20](#_Toc2234)

[图6.4 首页的店铺推荐 20](#_Toc20759)

[图6.5 商家列表 20](#_Toc21094)

[图6.6 商品信息展示 21](#_Toc18354)

[图6.7 点击加号将商品加入购物车 21](#_Toc8913)

[图6.8 购物车界面 21](#_Toc11359)

[商家后台管理 22](#_Toc6951)

[图6.9 新增菜品界面 22](#_Toc22499)

[图6.10 删除按钮 22](#_Toc17445)

[图6.11 在下架状态 23](#_Toc7921)

[图6.12 点击上架按钮后变为上架状态 23](#_Toc23306)

[图6.13 订单条目 23](#_Toc32696)

[图6.14 点击详细后可看到详细信息 23](#_Toc2930)

[图6.15 当顾客付款后可点击确认发货 24](#_Toc11496)

[快递员后台管理 24](#_Toc9416)

[图6.16 当快递完成后可点击确认发货 24](#_Toc27089)

[反馈评分管理 24](#_Toc4198)

[图6.17 订单完成后，用户可对订单进行评价 25](#_Toc2186)

**表目录**

[三、数据库基本表的定义 8](#_Toc2409)

[用户表(users) 9](#_Toc15398)

[表3.1 用户表(users) 9](#_Toc20866)

[订单表(orders) 10](#_Toc29351)

[表3.2 订单表(orders) 10](#_Toc28088)

[订单详细表(orderitems) 10](#_Toc6775)

[表3.3 订单详细表(orderitems) 11](#_Toc19989)

[优惠券表(coupons) 11](#_Toc2710)

[表3.4 优惠券表(coupons) 11](#_Toc53)

[商家表(shops) 11](#_Toc19045)

[表3.5 商家表(shops) 12](#_Toc9561)

[食品表(cuisine) 12](#_Toc30691)

[表3.6 食品表(cuisine) 12](#_Toc26179)

[快递员表(deliverymen) 13](#_Toc30876)

[表3.7 快递员表(deliverymen) 13](#_Toc15062)

**一．系统功能需求分析**

“学院路”外卖订购配送系统的设计目标是：方便顾客在网站上选购食品，并享受食品配送到家服务；为商家提供在线的食品展示、推广与销售渠道；为快递员分配订单配送任务。为实现此目标，系统应具备以下功能:

1. 用户的注册、验证与权限管理：

用户可以以顾客、商家、快递员三个不同身份在系统进行注册，注册时系统需要记录用户名、密码、邮箱等信息，在验证了邮箱地址后可以正式使用系统的服务。注册完成后，用户可以使用相应的用户名与密码信息进行登录。同时，每个身份的用户属于不同的角色组，具有不同的操作权限，因而系统具有权限管理功能，在操作前先进行相应的权限检查。

1. 商品的选购与订单的提交：

顾客可以从商家列表中选择商家，从而查看该商家的菜品信息。每条菜品信息包括菜名、价格等基础信息与图片、文字介绍等详细信息。通过点击菜品图片下加号，用户可以并选购相应的菜品。用户所选的菜品会先加入购物车，以方便用户增删菜品。当菜品选购完成后，用户可以在填写地址等信息后选择提交订单，以将订单发送给对应商家。

1. 商家后台管理：

商家用户可以向系统提交新增的菜品，并配以菜名、价格等基础信息与图片、文字介绍等详细信息。对于不再销售的菜品，商家可以选择删除菜品。同时对于一些暂时断货、短期无法销售的菜品，商家可以选择暂时下架，并在销售恢复后进行上架。

商家可在后台接受用户订单，以及在菜品完成后通知用户。

1. 快递员的后台管理功能：

快递员可在后台查看被分配的订单信息，并在订单送达后，选择订单完成，更改订单状态。

1. 评价反馈功能：

订单完成后，用户可以对商家的菜品，以及快递员的配送服务进行评价。评价较高的商家会被推荐到首页，获得更多的点击量；评价较高的快递员会获得更多的订单配送任务，以得到更高报酬。

## 二．系统功能结构设计

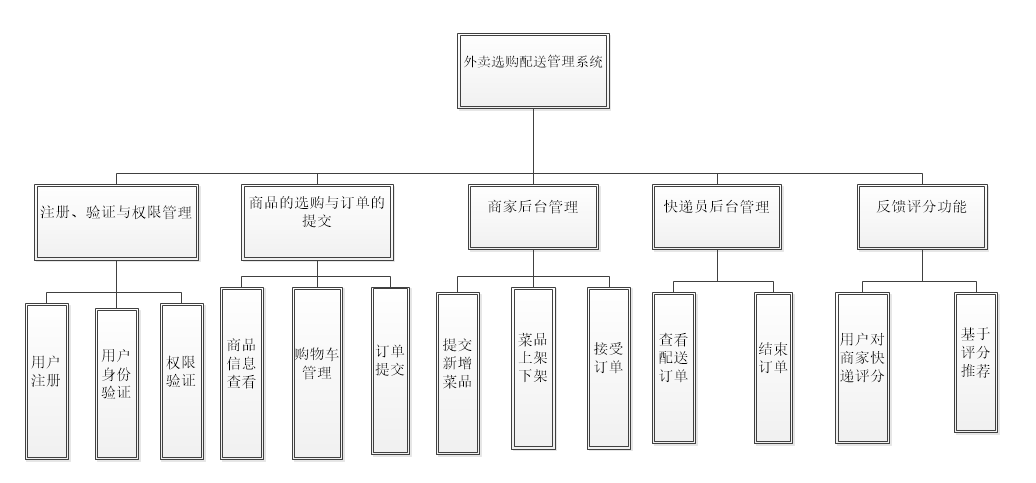
****

图2.1 系统功能结构设计图

系统结构功能设计图如上，下面分条进行详述。

**注册验证与权限管理**

**用户注册与验证**

用户在正式使用系统服务前必须进行注册。用户注册时，需要提交用户名、密码、邮箱、电话号码等信息。如果身份是顾客与商家，需要提交地址信息。在注册信息提交完成后，为防止恶意注册，还需验证用户信息真实性。系统向用户邮箱发送包含验证链接的邮件，用户点击链接可进行信息验证。

**用户权限管理**

用户具有顾客、商家、快递员等不同的角色，每种角色具有不同的操作权限。因而系统在进行每项操作前，会首先进行权限检查。如果检查发现当前登录用户不具有相应权限，系统则终止操作。

**商品的选购与订单提交**

**商品的查看与选购**

用户可在首页推荐或商家列表中选择商家，从而查看该商家提供的商品信息。每则商品信息包括商品名、价格、照片、详细说明等信息项。如果看到心仪的商品，用户可进行选购从而将该商品加入购物车。

**购物车管理与订单提交**

用户可以通过点击购物车，查看其所选择的商品详细信息。同时，通过点击购物车上的加减按钮，用户可以方便地增减选购商品的数量。当用户完成选购自己所要的商品后，可以点击提交按钮对于订单进行提交。

**商家后台管理**

**菜品的新增与删除**

商家可以通过后台管理，为自己的店铺新增菜品。在提交新菜品时，商家需要提供菜品的菜品名，价格，展示图片，文字介绍等信息。菜品提交后，其信息即可被顾客查看。如果某种菜品，商家已经不再销售，则可以通过在后台点击菜品图片下方的删除按钮，从而对菜品进行删除操作。

**菜品的下架与上架**

商家在提交菜品后，为给商家准备时间，其默认处于未上架状态，在未上架状态下，商品只可被顾客查看，但不能进行选购。待商家准备完毕，能够提供相应菜品后，在后台点击菜品图片下方的上架按钮，则可以简单地对商品进行上架操作，菜品上架之后，顾客则可对其进行选购。如果某种菜品暂时无法供应，商家可以在后台点击其图片下方的下架按钮，从而对其进行下架操作。

**订单的查看与接受**

商家可以在后台查看当前待处理订单与历史订单。每项订单信息条目包括菜品名、菜品数量与顾客地址、电话号码等信息。通过点击待处理订单上的接受订单按钮，商家可以通知顾客订单已被接受。当商家准备好订单上的菜品后，可以点击开始配送按钮，从而使得系统可以分配快递员，以及通知顾客菜品已准备完成。

**快递员后台管理**

**查看并接受配送任务**

快递员可以通过快递员后台，查看分配给自己的配送任务。每项配送任务信息条目包括商家地址、商家电话、顾客地址、顾客电话等信息。快递员可以通过点击配送信息旁边的接受任务按钮，通知系统改变快递员状态（空闲->正在配送），并开始配送任务。

**完成订单**

快递将订单送达后，可以通过后台点击相应配送任务旁边的完成订单按钮，通知系统改变订单状态。

**反馈评分管理**

**用户对商家与快递员进行评价**

在订单完成后，用户可以对此次服务进行评价。用户可以根据菜品价格、菜品口味等信息对于商家进行评价；同时也可根据配送速度，配送员服务态度等信息对于配送员进行评价。每次评价的结果将计入数据库。评价较高的商家，将获得首页推荐的机会；评价较高的配送员将获得更多的订单分配，从而获得更多报酬。

## 三、数据库基本表的定义

根据之前的设计分析，数据库主要使用一下七张基本表：用户表（users），订单表（orders），订单详细表（orderitems），优惠券表（coupons），商家表（shops），食品表（cuisine），快递员表（deliverymen）。下面对于每个基本表进行定义解释。

### 用户表(users)

**表结构**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 类型 | 允许为空 | 默认值 | 额外 |
| username | varchar(10) | 否 | 无 | 不可重复 |
| password | varchar(40) | 否 | 无 | 无 |
| userid | int(11) | 否 | 无 | 主码，自动增加 |
| role | Int(11) | 否 | 0 | 无 |
| address | text | 是 | NULL | 无 |
| verified | tinyint(1) | 否 | 0 | 无 |
| token | varchar(40) | 否 | 无 | 无 |
| email | varchar(40) | 否 | 无 | 不可重复 |
| phone | varchar(15) | 是 | 无 | 无 |
| photo | varchar(256) | 是 | NULL | 无 |

表3.1 用户表(users)

**表项说明**

username : 表示用户的用户名，登录时使用，不可重复

password : 表示用户的密码

userid: 表示用户的id号

role: 表示用户的角色，主要分为游客（注册但并未验证），顾客，商家，快递员

address: 表示用户的家庭住址

verified: 用户是否已通过邮件验证

token: 验证时所要用的token

email: 用户注册使用的邮箱

phone: 用户的手机号

photo: 用户上传头像的存储路径

### 订单表(orders)

**表结构**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 类型 | 允许为空 | 默认值 | 额外 |
| orderid | int(11) | 否 | 无 | 主码，自动增加 |
| userid | int(11) | 否 | 无 | 外码 |
| status | int(11) | 是 | NULL | 无 |
| total | int(11) | 否 | 0 | 无 |
| shopid | int(11) | 否 | 无 | 外码 |
| address | text | 否 | 无 | 无 |
| deliveryid | int(11) | 是 | 无 | 外码 |

表3.2 订单表(orders)

**表项说明**

orderid: 表示订单的id号码

userid : 表示点餐顾客的id号码

status: 表示订单的状态，共分为：未提交（1），已提交（2），已付款（3），商家已接单（4）， 快递员开始配送（5），订单送达（6），用户提交评价（7）。

total: 表示订单的总价

shopid: 表示订单对应的商家id

address: 表示顾客的家庭住址

deliveryid: 接受订单的快递员id

### 订单详细表(orderitems)

**表结构**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 类型 | 允许为空 | 默认值 | 额外 |
| userid | int(11) | 否 | 无 | 外码 |
| orderid | int(11) | 否 | 无 | 主码 |
| itemid | int(11) | 否 | 无 | 主码 |
| amount | int(11) | 否 | 无 | 无 |

表3.3 订单详细表(orderitems)

**表项说明**

orderid: 表示订单的id号码

userid : 表示点餐顾客的id号码

itemid: 表示该项食品的id号码

amount: 表示顾客所点该项食品的id号码

### 优惠券表(coupons)

**表结构**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 类型 | 允许为空 | 默认值 | 额外 |
| shopid | int(11) | 否 | 无 | 主码 |
| money | int(11) | 否 | 无 | 主码 |
| downmoney | int(11) | 否 | 无 | 无 |

表3.4 优惠券表(coupons)

**表项说明**

shopid: 表示优惠券所在的商家id号码

money : 表示优惠券适用的最低价格

downmoney: 表示该优惠券所允许的折扣

### 商家表(shops)

**表结构**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 类型 | 允许为空 | 默认值 | 额外 |
| id | int(11) | 否 | 无 | 主码 |
| credit | int(11) | 否 | 0 | 无 |
| income | int(11) | 否 | 0 | 无 |
| address | text | 否 | 无 | 无 |
| photo | varchar(256) | 是 | 无 | 无 |

表3.5 商家表(shops)

**表项说明**

id: 表示商家的id号码

credit : 表示商家的总信用积分

income: 表示商家通过平台获得的总收入

address: 表示商家的地址

photo: 表示商家上传的招牌照片存储地址

### 食品表(cuisine)

**表结构**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列命 | 类型 | 允许为空 | 默认值 | 额外 |
| cid | int(11) | 否 | 无 | 主码，自动增加 |
| shopid | int(11) | 否 | 无 | 外码 |
| name | varchar(256) | 否 | 无 | 不可重复 |
| pic | varchar(256) | 否 | 无 | 无 |
| price | decimal(10.2) | 否 | 无 | 无 |
| info | varchar(1024) | 否 | 无 | 无 |
| status | tinyint(1) | 否 | 0 | 无 |

表3.6 食品表(cuisine)

**表项说明**

cid: 表示食品的id号码

shopid : 表示食品所在的商家id号码

name: 表示食品名称

pic: 表示食品图片的存储地址

price: 表示食品的价格

info: 表示食品的详细介绍信息

status: 表示食品是否上架

### 快递员表(deliverymen)

**表结构**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列命 | 类型 | 允许为空 | 默认值 | 额外 |
| deliveryid | int(11) | 否 | 无 | 主码 |
| phone | int(11) | 否 | 无 | 无 |
| credit | int(11) | 否 | 0 | 无 |
| status | tinyint(1) | 否 | 0 | 无 |

表3.7 快递员表(deliverymen)

**表项说明**

deliveryid: 表示快递员的id号码

phone : 表示快递员的电话号码

credit: 表示快递员的信用评分

status: 表示快递员的当前状态（空闲或者正在配送）

**四、触发器的定义与实现**

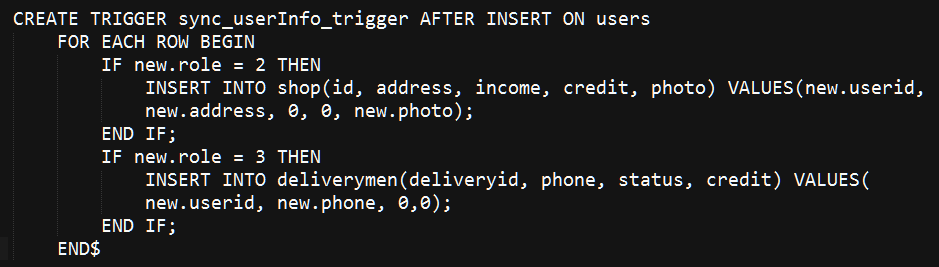
### 用户信息同步触发器(sync\_userInfo\_trigger)

**作用对象：**用户表（users）

**作用时机：**插入操作之后

**触发器功能：**用户有三种身份可以选择：顾客，商家，快递员。而商家表与快递员表作为用户表的子表存在，用于存储其所需的一些详细信息。因而，将用户信息插入用户表时，如果其角色为商家或者快递员，还需要使用同一id号码，将部分信息同步到商家表或快递员表中，以保证数据的一致性。

**具体实现：**



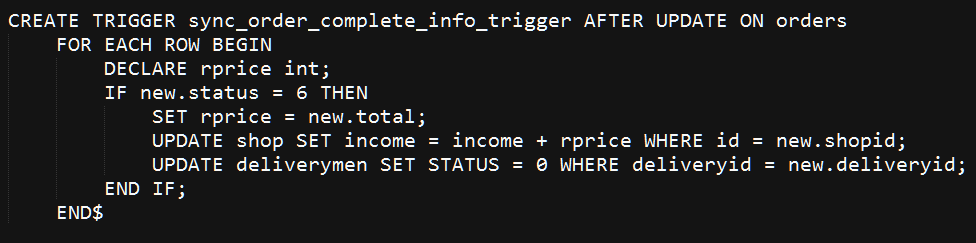
### 订单完结信息同步触发器(sync\_order\_complete\_info\_trigger)

**作用对象：**订单表（orders）

**作用时机：**更新操作之后

**触发器功能：**该触发器用于同步订单的完结信息。当一份订单完结时，除了需要改变订单状态，同时还需要给店家打入相应的款项，以及将快递员的状态设为空闲。

**具体实现**：



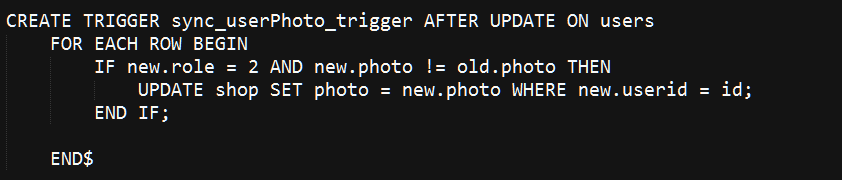
### 用户头像信息同步触发器(sync\_userPhoto\_trigger)

**作用对象：**用户表（users）

**作用时机：**更新操作之后

**触发器功能：**此触发器用于同步商家表与用户表中的信息。商家用户的头像即为其店铺的照片。因此当商家用户在users表中头像改变时，其在shops表中的头像也应相应改变

**具体实现：**



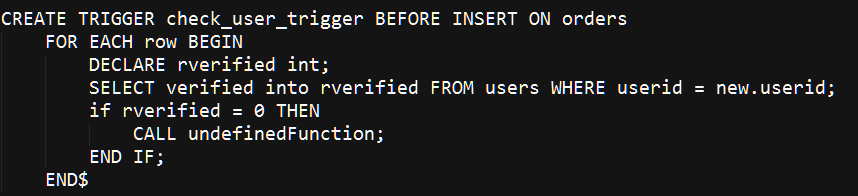
### 用户权限验证触发器(check\_user\_trigger)

**作用对象：**订单表（orders）

**作用时机：**插入操作之前

**触发器功能：**此触发器用于判断插入订单项的用户是否具有权限（是否为已验证用户），如果没有权限则抛出异常，终止插入操作。

**具体实现：**

****

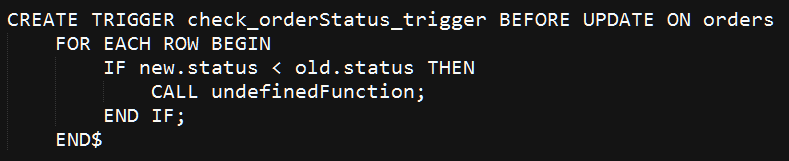
### 订单状态控制触发器(check\_orderStatus\_trigger)

**作用对象：**订单表（orders）

**作用时机：**更新操作之前

**触发器功能：**此触发器用于判断订单状态的变化方向。由于订单状态无法回退，因此一旦数据库用户尝试进行相应的操作，则抛出异常，终止操作。

**具体实现：**

****

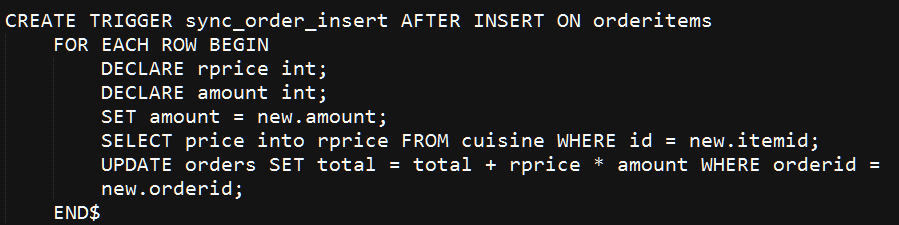
### 订单总价同步(插入操作)触发器(sync\_order\_insert)

**作用对象：**订单详细表（orderitems）

**作用时机：**插入操作之后

**触发器功能：**此触发器用于在向订单详细表中插入新的订单详细项时，根据其价格与数量，对于订单表(orders)中的总价数据进行相应的更新。

**具体实现：**

****

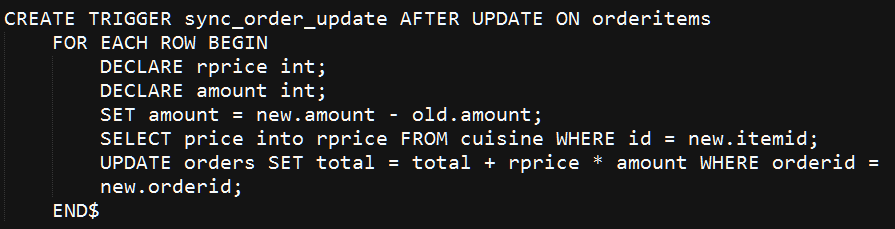
### 订单总价同步(更新操作)触发器(sync\_order\_update)

**作用对象：**订单详细表（orderitems）

**作用时机：**更新操作之后

**触发器功能：**此触发器用于在向订单详细表更新订单详细项时，根据其价格与数量，对于订单表(orders)中的总价数据进行相应的更新。

**具体实现：**

****

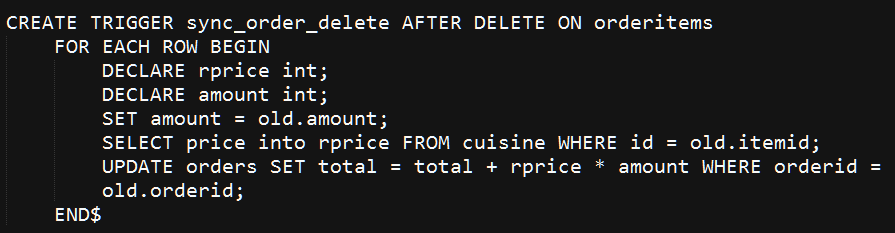
### 订单总价同步(删除操作)触发器(sync\_order\_delete)

**作用对象：**订单详细表（orderitems）

**作用时机：**删除操作之后

**触发器功能：**此触发器用于在从订单详细表中删除订单详细项时，根据其价格与数量，对于订单表(orders)中的总价数据进行相应的更新。

**具体实现**

****

**五、存储过程的定义与实现**

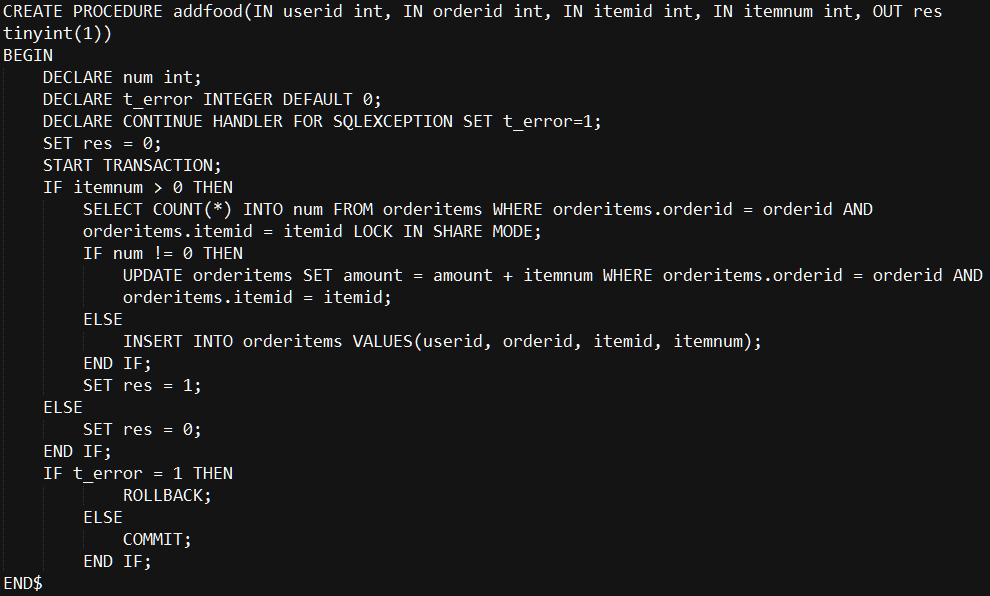
### 添加食品(addFood())

**过程输入：**用户id，订单id，食品id，食品数量

**过程输出：**添加食品操作是否成功

**功能：**首先判断食品数量是否为正，如果不是，数据不合法，返回0（操作失败）。否则查找订单详细表看该订单中是否已有该食品项。如果没有，根据用户id，订单id，食品id及数量插入新的食品项；否则直接在原食品项上更新食品数量，返回1（操作成功）。如果过程中出现了异常，则直接事务回滚，返回0。

**存储过程实现：**



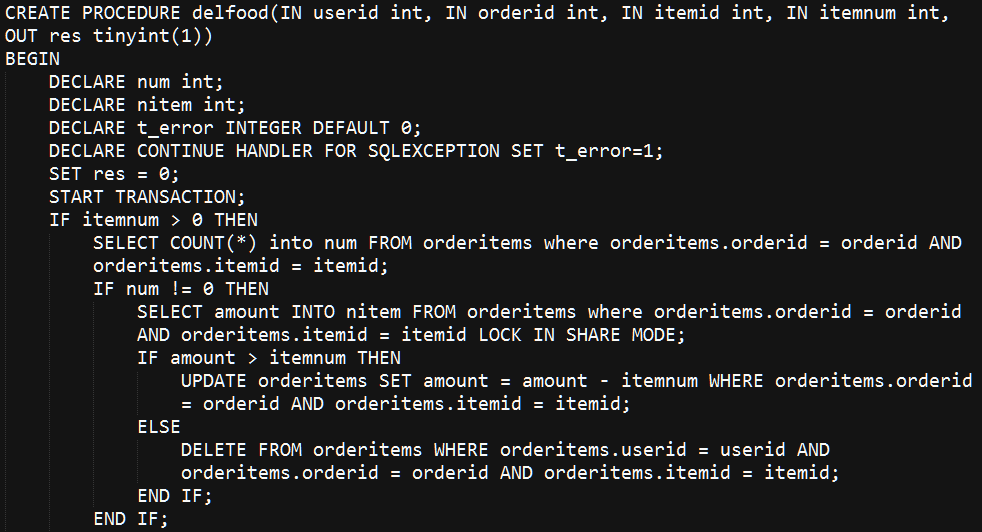
### 删除食品(delFood())

**过程输入：**用户id，订单id，食品id，食品数量

**过程输出：**删除食品操作是否成功

**功能：**首先判断食品数量是否为正，如果不是，数据不合法，返回0（操作失败）。否则查找订单详细表看该订单中是否已有该食品项。如果没有，则无需进行删除操作；否则判断原选择的食品数量与要删除的食品数量大小，如果要删除的数量多于已有的数量，则直接将食品项整个删除，否则仅需减少食品数量，返回1（操作成功）。如果过程中出现了异常，则直接事务回滚，返回0。

**存储过程实现：**

****

****

**六、系统实现结果**

**注册验证与权限管理**

**用户注册与验证**

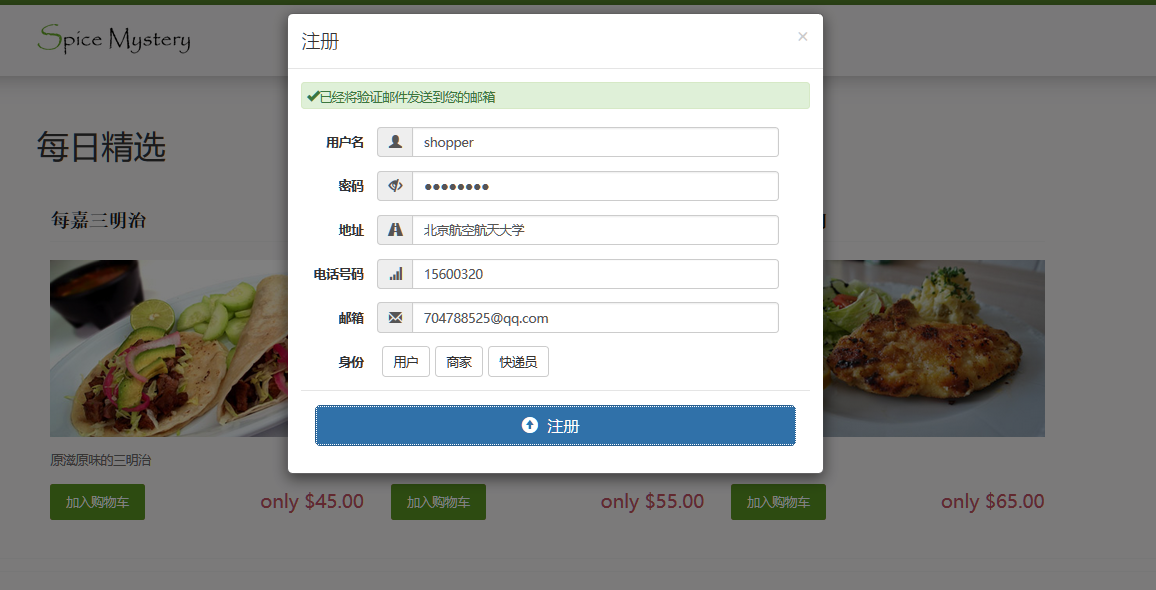


图6.1 用户注册界面

用户注册界面直接以弹出框的形式实现，需要填写用户名、密码、地址、电话号码、邮箱、身份等信息。



图6.2 用户注册界面（点击注册并验证后）

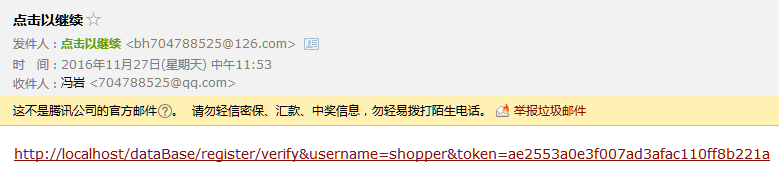


图6.3 验证邮箱连接

如果用户信息正确，系统将会发送验证邮件到邮箱中去。用户点击链接即可进行验证。

**商品的选购与订单提交**

**商品的查看与选购**



图6.4 首页的店铺推荐

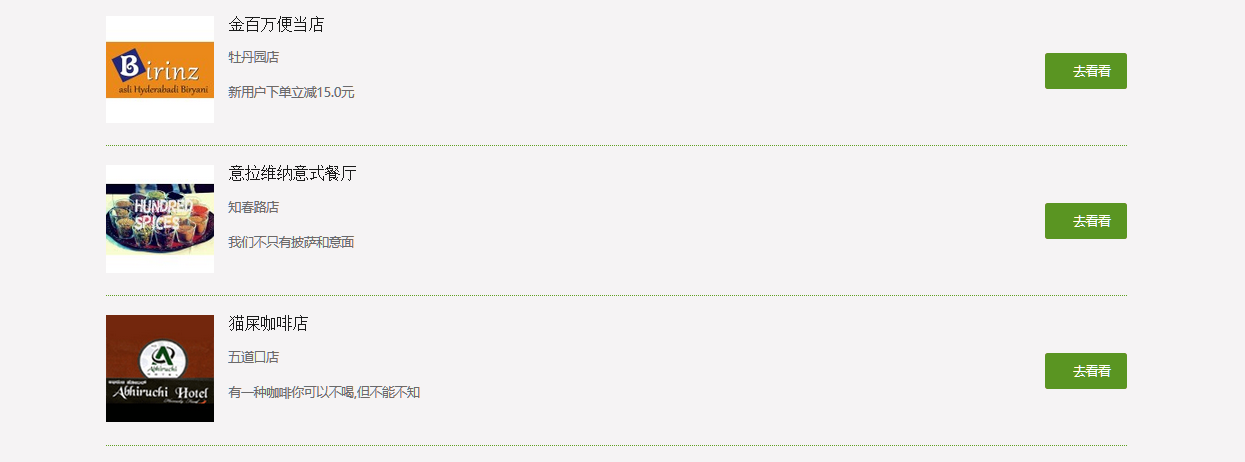


图6.5 商家列表

用户可在首页推荐或商家列表中选择商家，从而查看该商家提供的商品信息。



图6.6 商品信息展示

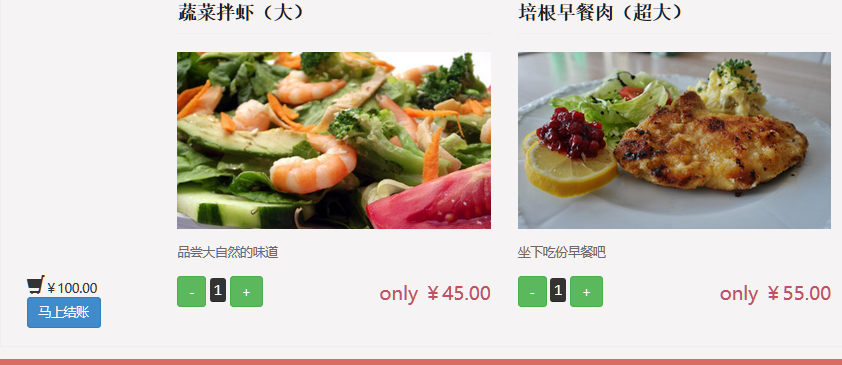


图6.7 点击加号将商品加入购物车

如果看到心仪的商品，用户可进行选购从而将该商品加入购物车。

**购物车管理与订单提交**

****

图6.8 购物车界面

用户可以通过点击购物车的图标，查看其所选择的商品详细信息。购物车以弹出界面的形式实现。当用户完成选购自己所要的商品后，可以点击确认提交按钮对于订单进行提交。

**商家后台管理**

**菜品的新增与删除**

****

图6.9 新增菜品界面

商家可以通过后台管理，为自己的店铺新增菜品。在提交新菜品时，商家需要提供菜品的菜品名，价格，展示图片，文字介绍等信息。菜品提交后，其信息即可被顾客查看。



图6.10 删除按钮

如果某种菜品，商家已经不再销售，则可以通过在后台点击菜品图片下方的删除按钮，从而对菜品进行删除操作。

**菜品的下架与上架**



图6.11 在下架状态



图6.12 点击上架按钮后变为上架状态

商家在提交菜品后，其默认处于未上架状态。待商家准备完毕，能够提供相应菜品后，在后台点击菜品图片下方的上架按钮，则可以简单地对商品进行上架操作。果某种菜品暂时无法供应，商家可以在后台点击其图片下方的下架按钮，从而对其进行下架操作。

**订单的查看与接受**

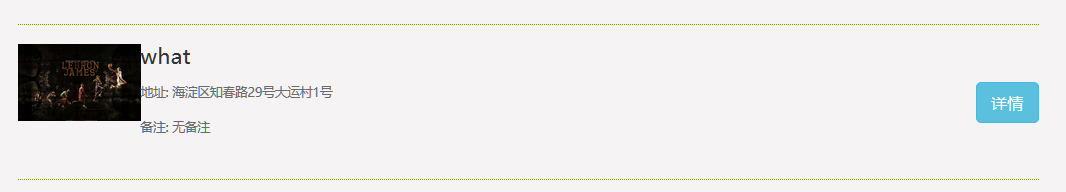
****

图6.13 订单条目

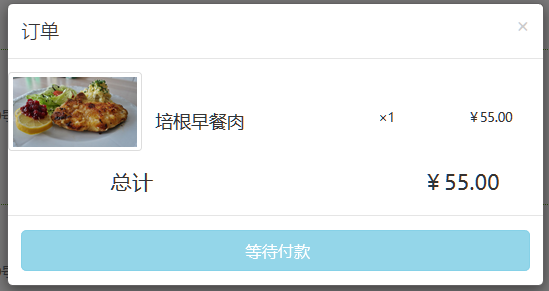


图6.14 点击详细后可看到详细信息



图6.15 当顾客付款后可点击确认发货

**快递员后台管理**

**完成订单**

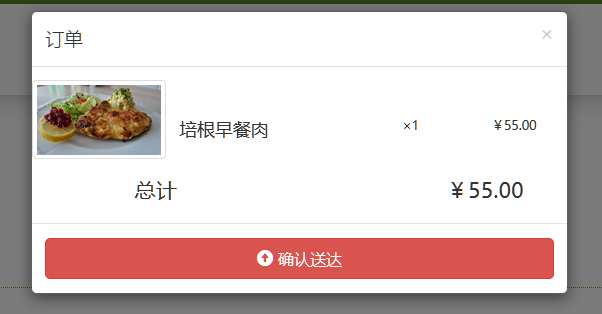


图6.16 当快递完成后可点击确认发货

快递将订单送达后，可以通过后台点击相应配送任务旁边的完成订单按钮，通知系统改变订单状态。

**反馈评分管理**

**用户对商家与快递员进行评价**

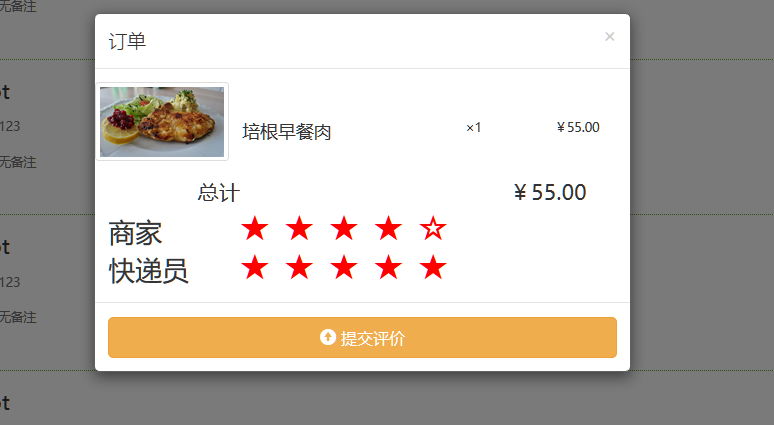
****

图6.17 订单完成后，用户可对订单进行评价

**七、总结**

总的来说，此数据库系统的实现，基本符合了预先对于外卖订购与配送系统的设计，在实现过程中我们使用了事务处理，视图，索引，触发器，存储过程等机制，对于课上所学的内容进行了实践，同时也对于网站设计实现的前后端知识进行了进一步的学习与实践。

客观的说，此次系统设计实现还是有一定难度的，实现的过程也不总是一帆风顺。我们在最初设计数据库时，少考虑了一些数据项；我们在一些技术难点上遇到了障碍；我们组内成员在实现时由于交流不到位，使得设计接口时出现不兼容；由于前端设计经验不足，在实际运行时不是很稳定。但我们队友之间相互信任，相互鼓励，还是跨过了一个又一个的坎，最终也坚持了下来，将整个系统比较完整地实现了出来。

通过这次实践，我们首先对于课上知识有了更为深刻的理解。很多知识都是纸上得来终觉浅，真正实践时才意识到这些知识的重要性。就以事务处理中的并发控制为例。在刚学完这节课后，我觉得课上的东西还是比较偏理论，在实践中无需如此严格保证并发的正确性。然而我们真正在实现功能时，我突然意识到对于购物网站这样高并发的平台来说，事务执行的正确性是相当关键的一个特性。因此我们便结合课上所学的两段锁协议，与mysql本身提供的事务管理机制，在每个事务中的读操作前先获取对象的S锁，在写操作前先获取X锁，并在事务commit或rollback时一次释放所有锁，从而保证并发事务间的正确合理调度。

除此之外，我们队友之前也学会了如何更有效地去合作。刚开始时，我们两人都还是习惯于一直以来的单人编程模式，每人都只负责自己的模块。然而这就导致我们之间交流不足，有很多接口两人的理解有偏差，从而使得最终对接的时候不很顺利。因而到了后期，我们提高了交流的频率与效率，每次在实现一个用于对接的模块前，先讨论好对方需要的接口，这也使得后期完成的代码对接时更加顺利稳定。同时，在交流的过程中，通过不同想法的碰撞，我们也不断地发现自己想法的局限性，从而使得系统的设计更加完善。

总而言之，通过这次系统设计实现过程，我们对于课上知识有了更加深刻的理解，也学会了如何高效合作；同时又跨过了很多技术难点，较为完整地做出了一个有一定实用价值的数据库系统。在编程能力提升的同时，也对自己有了更多信心。因此，我们感觉在这过程中所付出的努力，所熬的夜，都是值得的。