# 一 摘 要

网上书店课程设计主要就其开发背景，需求分析，逻辑设计以及数据库系统的实现。在过去，人们都是通过去书店购买自己需要的书籍。但经过几年实践，我国电子商务发展迅猛，并且越来越大众化，还有就是网络的发展与应用已经蓬勃发展起来。因而设计一套适合现代人网上购书的系统如网上书店是非常必要并且可行的,并在我国已得到初步应用并有很广泛的市场需求前景。网上书店系统可以实现人们在家就能逛书店和购买图书的愿望。本系统主的功能是帮助经营实物书店的人们扩大市场和增加知名度以及方便网上用户线上购买其需要的书籍。基本功能包括用户注册会员、登录、图书展示、图书查询、在线浏览图书、远程购买、远程交易等。系统开发工具是使用VS2015及SQL Server 2008 R2数据库管理，系统是基于出版社等用户背景开发的，可以有效的实现网上图书交易。在论文中首先介绍了电子商务及网上书店的相关知识，并根据以上知识制作网站。网上图书销售系统，我们采用的.NET三层框架结构，其目的在于系统的可维护性好，结构清晰，后期开发简单。文中主要包括调研分析、可行性研究、平台论证、需求分析、网站的总体设计、详细设计、数据库设计等，并通过流程图和文字加以解释说明。

**关键字**：SQL Server 2008 R2，Visual studio 2010,.NET三层框架

**小组成员**：彭晶，黄博，姚云

# Abstract

This essay mainly introduces this project title ‘Online Bookshop's exploitation background and how to analyze designing and achieving。Now ,most people find and buy books which they are need by going to the bookshop。With several years practicing，electronic commerce has developed rapidly and the going to more and more pragmatic，Another phenomenon we should notice is that the Internet evolution and application has progressed。 A kind of 'Online bookshop 'systems which is suited for some individuals is very necessary and feasible, and this system in our country has applied and shows a very marketing trend in the future. Our system mainly helps those people who are managing a real bookshop expand their marketplace and make citizens know their business。This system's function includes user registration，log in，showing the products，searching，online-buying，online-trading and so on。 We use Asp.net and SQL Server 2008 as exploitation tools，the system 's research background is basic with publishing house and some users，and it can do Online booking shopping effectively。 At the beginning of the essay，we introduce some knowledge about electronic commerce and Online Bookshop，and make use of these knowledge for designing websites。 This essay primarily contains research analysis, feasibility researching，requirement analysis，website ,details and databank designing and uses float chart and words to explain this system.

目 录

[一 摘 要 1](#_Toc455146451)

[Abstract 2](#_Toc455146452)

[二 前 言 4](#_Toc455146453)

[三 系 统 概 述 5](#_Toc455146454)

[四 需求分析 5](#_Toc455146455)

[4.1 需求设计 5](#_Toc455146456)

[4.2使用范围 8](#_Toc455146457)

[4.3用户界面及界面 8](#_Toc455146458)

[4.4可行性分析 9](#_Toc455146459)

[4.4.1.技术可行性 9](#_Toc455146460)

[4.4.2经济可行性 9](#_Toc455146461)

[4.4.3.人员可行性 9](#_Toc455146462)

[4.4.4.影响网站开发的因素 9](#_Toc455146463)

[4.5开发和使用环境 10](#_Toc455146464)

[4.5.1开发环境 10](#_Toc455146465)

[4.5.2使用环境 10](#_Toc455146466)

[4.6目标系统的其他需求 10](#_Toc455146467)

[4.6.1可靠性 10](#_Toc455146468)

[4.6.2灵活性 10](#_Toc455146469)

[五 系统管理流程及模块功能分析 11](#_Toc455146470)

[5.1 数据库设计模块 11](#_Toc455146471)

[5.2 项目系统设计模块 16](#_Toc455146472)

[5.2.1 项目框架简介 16](#_Toc455146473)

[5.2.2 购物流程 19](#_Toc455146474)

[5.2.3 项目运行效果及测试结果 19](#_Toc455146475)

[5.3 项目系统核心代码 25](#_Toc455146476)

[六 特别说明 33](#_Toc455146478)

[6.1 网站的安全性： 33](#_Toc455146479)

[6.2 网站可维护性： 33](#_Toc455146480)

[6.3 网站的灵活性： 33](#_Toc455146481)

[6.4 故障处理： 33](#_Toc455146482)

[七 总结 34](#_Toc455146483)

# 二 前 言

随着Internet和web技术的迅速发展，电子商务已经被广大互联网用户所接受。作为图书销售和电子商务相结合的产物，网上书店以其具有可降低销售成本、交易活动不受时空限制和信息传递迅速灵活等优势，已经受到广大读者的青睐。

网上书店是以网站作为交易平台，将图书的基本信息通过网站发布到Web中。然后，客户可通过Web查看图书信息并提交订单，实现图书的在线订购。订单提交后，书店职员将订单及时处理，以保证客户能在最快时间内受到图书。一个基于B2C的网上书店系统需求描述如下。

该网上系统支持4类用户：游客，会员，职员和系统管理员。游客可以随意浏览图书及网站信息，但只有在注册为网站会员后才能在线购书。游客注册成功后即视为普通会员，当其购书金额达到一定数量时可升级为不同等级的VIP会员，以享受相应的优惠折扣。会员登录系统后，可进行的主要操作有：通过不同方式(如书名、作者、出版社等)搜索图书信息、网上订书、在线支付、订单查询与修改、发布留言等。书店工作人员以职位身份注册登录后，可进行的主要操作有：维护与发布图书信息、处理订单、安排图书配送和处理退货等。系统管理员的主要职责是维护注册会员和职员的信息。

# 三 系 统 概 述

（1）首先在网上书店系统中，数据库应当解决如下需求：

保存注册用户的个人信息，包括联系资料等。

保存图书信息，包括书名，价格，作者，图片等信息。

保存购物车的详细信息。

保存购书信息，包括用户ID，图书ID，数量，总价等。

保存订单信息，包括日期，支付方式，是否确认等。

（2）其次整个网站应实现如下的任务：

用户管理：用户管理模块包括用户注册、用户登录、修改用户个人信息等功能。

图书管理：图书管理包括允许用户浏览图书列表、查看图书商品详细信息、快速查询所需要的图书3个功能。

购物车管理：顾客可以把要买的图书放入购物车，查看购物车的内容，对自己购物情况进行修改或者确认。顾客可以删除图书、修改图书数量。顾客确认购物情况无误后可以结账。

订单管理：订单管理功能主要实现显示订单的详细信息这个功能。显示订单详细信息负责显示某个指定订单的详细信息。

# 四 需求分析

需求分析就是分析用户需求，是设计数据可的第一步。该步骤主要是通过详细调查显示世界要处理的对象，并在此基础上确定系统的功能。下面主要分析网上书店的业务需求、功能需求和业务规则等。

## 4.1 需求设计

4.1.1 业务需求及处理流程

业务需求是根据现实世界对象需求，描述应用的具体业务处理流程，并分析哪些业务是计算机可以完成的，而哪些业务是不能由计算机完成的。

网上书店主要业务包括：图书信息发布与查询，订购图书、处理订单并通知配送公司送书等。

4.1.2 功能需求及数据需求分析

功能需求分析是描述系统应提供的功能和服务。根据上述需求描述和业务流程，通过与网上书店人员的沟通与交流，网上书店主要功能和数据需求包括：

① 注册管理

* 会员注册。会员注册时要求填写基本信息，包括姓名、登录密码、性别、出生年月、地址、邮政编码、电话、电子邮箱等信息，系统检查所有信息填写正确后提示会员注册成功并返回会员编号。
* 职员注册。书店工作人员以职员身份注册并填写基本信息，包括姓名、登录密码、性别、出生年月、部分、薪水、住址、电话和电子邮箱等信息。系统检查所有信息填写正确后提示注册成功并返回职员编号。

② 图书管理

* 增加图书信息。当所有新书发布时，书店职员负责添加和发布图书信息，包括ISBN、书名、作者、版次、类别、出版社、定价、售价、内容简介和目录等。
* 图书信息查询。网站需提供多种方便快捷的方式进行图书检索，如既可以输入指定关键词进行简单查询，也可以根据ISBN、书名、作者、出版社、出版年份等单一或组合条件进行查询。
* 图书信息更新及删除。图书信息发布后职员可以随时更新和删除图书信息。

③ 在线订书

会员登录网站后，将需要订购的图书放入购物车中并填写购买数量。购物车内的图书可以随意添加、删除和修改数量，并能即使统计购物车内的图书总价格。

选书完成后，会员还需填写配送信息、发票单位及选择支付方式（在线支付或上门付款）。配送信息默认为会员注册时填写的基本信息，也可填写新的配送信息，包括收货人、送货地址、邮政编码及联系电话、付款方式、订单状态、订单明细（包括书号、书名、数量、价格）和发票单位等。如果选择了在线支付方式，则还需进行网上结算。余额不足，则取消订单。

④ 订单管理

* 订单查询。订单提交后，会员可随时查询订单的最新状态以及全部历史订单。
* 订单取消及更新。订单未审核前，允许会员取消订单及更新订单信息。
* 订单受理。订单生成后职员对订单进行审核。如发现订单信息填写不正确，则退回客户重新填写。如正确无误，则安排配送。

⑤ 配送管理

一张订单所订购的图书可拆分成不同的配送发货。每张配送单包括配送订单号、收货人、送货地址、邮政编码、联系电话、送书明细（包括书名及数量）并填写一张发票，发票内容包括发票单位和总金额等信息。

⑥出版社管理

网上书店直接从出版社采购图书。为方便查询出版社信息，要求保存和维护出版社信息，包括出版社编号，出版社名称，出版社地址，邮政编码，联系人，联系电话，传真和电子邮箱等属性。

⑦ 配送公司管理

网上书店通过配送公司将图书送到会员手中。为方便查询配送公司信息，要求保存和维护配送公司信息，包括公司编号，公司名称，公司地址，邮政编码，联系人，联系电话，传真和电子邮箱等属性。

⑧ 留言管理

* 发布留言。会员可在网站发布留言或评论。留言需要记录留言人，留言内容，发布时间等信息。
* 回复留言。书店职员可回复留言，并记录回复人和回复时间及回复内容等。

⑨ 用户管理

* 会员升级。系统可对会员进行分级，即当会员订书金额达到一定数额后成为不同级别的用户，以享受相应的优惠折扣。
* 会员信息维护。系统管理员及会员可修改、删除和更新会员信息。
* 职员信息维护。系统管理员及职员可修改、删除和更新职员信息。

4.1.3 业务规则分析

业务规则分析主要是分析数据之间的约束以及数据库的约束。基于上述功能需求，通过进一步了解，网上书店业务规则如下:

（1） 所以用户均可搜索图书信息，但只有注册会员才能在网上提交订单；只有注册职员才能维护图书信息及受理订单。

（2） 每位会员由会员编号唯一标识，会员编号由系统按时间顺序生成。

(3) 每位职员由职员编号唯一标识，职员编号由系统按时间顺序生成。

(4) 当普通会员购书总额达到10000元，升级为三级VIP会员，享受售价9.5折优惠；购书达到20000，升级为二级VIP会员，享受售价9折优惠；购书总额达到30000元，升级为一级VIP会员，享受售价8.5折优惠。

(5) ISBN是图书的唯一标识。系统需记录每种图书当前库存数量，当库存量低于某一阈值时，则通知补货。

(6) 选购的图书必须放入购物车后才能生成订单。

(7)每个订单用订单编号唯一标识。订单编号由系统按时间顺序生成，后提交的订单具有更大的订单号。

(8 )订单需记录当前状态，包括未审核、退回、已审核和已处理结束等状态。

(9 )同一订单可订购多种图书，且订购数量可以不同。因此，一张订单可包括多个书目明细，包括ISBN、图书名称、订购数量、订购价格。订单中的每种图书需记录其转态，包括未送货、已送货和已送到等状态。

(10) 订单受理前允许会员删除所选图书，修改购书数量、配送信息和发票单位、甚至取消订单，但是订单审核通过后，则不允许再做任何修改。

(11) 订单中的图书采取先到先发货的原则。若一订单中的图书未同时有货，可拆分成不同配送单发货；但是，一订单中的某种图书职员库存有足够存书时候才能安排配送。

(12) 配送单有配送单编号标识。每个订单的配送编号是由订单编号加上系统按时间顺序生成的流水号组成。

(13)每张配送单对应一张发票。发票用发票的实际编号唯一标识。

(14) 当订单中的某种图书送到后，则更新该书的状态为“已送到”。当订单内全部图书状态为“已送到”时，则更新订单状态为“已处理结束”。

(15) 一种图书只由一个出版社出版，而一个出版社可以出版多种图书。

(16) 一个会员可发表多条留言。一个职员可回复多条留言。

完成需求分析后，接下来的任务就是根据上述分析结果设计数据库的概念模型，即E-R模型，包括确定实体集，联系集及属性。

## 4.2使用范围

网上书店相对于实体店：

（1） 营业时间不受限制，与传统的8小时营业时间不同，借助互联网，网上书店可以24小时全天候营业。这种不间断的服务方式对于巩固和扩大读者群、培育潜在的顾客具有重大意义。

（2） 不受营业场地限制。因为网上书店是虚拟书店，所以它无需门市，只要维持面积有限的库房即可正常运转，并以最低的成本经营最多的品种。

（3）供需双方之间信息交流的广度、深度和速度有了质的飞跃。可提供的图书信息与用户需求信息的相互沟通及匹配一直是制约图书销量增长的瓶颈，网上书店以其直观的界面、丰富的信息、灵活的检索方式和个性化的定制服务，成功地解决了这一难题。

（4）经营管理更加科学。现代信息技术的大量运用使得网上书店能够快捷地对业务数据进行采集、统计、分析和应用，这有助于克服传统营销模式中的主观性和盲目性，对于提高经营管理水平大有裨益。

网上书店的用户群:

(1) 主流人群为经常上网的读书爱好者，拥有能够上网的条件，以青年和中年人为主。

(2) 部分用户持有信用卡，可在网上直接付款；无信用卡的用户可以汇款进行交易。

(3) 从职业划分来看，一部分用户是高校学生，追求时尚快捷的购物方式，购买力有限；另一部分则为工作人群，追求高效经济的购物方式，购买力较强。

## 4.3用户界面及界面

网站的各大页面要求要简洁，方便，友好，易懂，便于浏览者和管理者的浏览和进行相关操作。主要分一下两块：

一、前台界面，通过用户注册和用户登录进入购物车，实现网上图书的购买。

二、管理员界面，只有专门的管理员才能登录，并对有关信息进行修改和删除等操作。主要对一下六大块进行管理：订单管理、管理员管理、会员管理、物流管理、图书管理和图片管理。

输入设备：键盘，鼠标。

输出设备：显示器。

显示风格：一般浏览器都可以浏览。

输出格式：网页方式。

## 4.4可行性分析

### 4.4.1.技术可行性

网站的设计用了ASP.NE技术，用c#语言编写后台，数据管理则用到SQL Server2008,开发环境是Microsoft Visual Studio 2010. 用ASP.NET编写的页面非常简单、灵活；有现成的封装控件，开发周期较短。

### 4.4.2经济可行性

现代社会很多网上资源都是免费的，人们在网上交易不仅省去很多时间，而且更加方便快捷，只要注册就可以成为会员，就可以使用本站的一些资源，从而形成交易欲望，公司就可以获得利润。另一方面，网站的开发和管理费用低，回报远远高于费用，所以店主根本不用担心费用问题。

### 4.4.3.人员可行性

因为网站是针对大部分的客户，所以使用本网站的人不需要具备很高的计算机技术就可以使用本网站，只要懂得上网就可以了。管理本网站的管理员则要求比较高，要懂得一定的数据库知识，并且熟悉各种操作。

### 4.4.4.影响网站开发的因素

影响网站开发的因素主要是技术的不断更新，网站要很好地保留下去的话就必须跟着更新，要不然就会因为技术落后而失去大量客户。

## 4.5开发和使用环境

### 4.5.1开发环境

Visual Studio 2010自带编译器和Web服务器进行调试，在部署网站之前，所有工作都可以在Visual Studio 2010完成。数据库管理用的是SQL Server2008.

### 4.5.2使用环境

（1）设施，如多媒体计算机、多媒体教室网络、校园网络、因特网等现已经盛行，网站的宣传效果可以到达。

（2）资源，为顾客及潜在顾客提供全部图书的信息及其他相关知识。

（3）平台，本网站主要以VS2010， SQL Server2010数据库工具实现本网站，而现计算机技术已经能为该网站提供一个很好的运行平台。

（4）通讯，现在网络通信发达，能够实现与客户远程沟通与交流。

（5）硬件环境

用户端:因为用户端主要用于浏览和操作数据，所以对客户端的硬件要求不高，不过现在的电脑有很高的性价比，因此普通配置即可，不做要求。

## 4.6目标系统的其他需求

### 4.6.1可靠性

使网站管理人员和用户访问网站时都能正常操作。

### 4.6.2灵活性

支持多种付款方式、多种货物搜索方式以及多种送货方式。网站支持后续更新。

# 五 系统管理流程及模块功能分析

## 5.1 数据库设计模块

本课设为数据库原理与技术课程设计，很自然数据库的设计非常重要。

根据需求原则，各实体集设计如下：

1. 职员（Employee）的实体集。其属性有：职员编号（employeeNo）、登录密码（empPassword）姓名（empName）、性别（sex）、出生年月（birthday）、部门（department）、职务（title）薪水（salary）、住址（address）、电话（telephone）和电子邮箱（email）等。

**表名：Employee**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **序号** | **列名** | **数据类型** | **长度** | **小数位** | **标识** | **主键** | **外键** | **允许空** | **默认值** | **说明** |
| 1 | employeeNo | char | 8 | 0 |  | 是 |  | 否 |  | 员工编号 |
| 2 | empPassword | varchar | 10 | 0 |  |  |  | 是 |  | 登录密码 |
| 3 | empName | varchar | 12 | 0 |  |  |  | 是 |  | 员工姓名 |
| 4 | sex | char | 1 | 0 |  |  |  | 是 |  | 员工性别 |
| 5 | birthday | datetime | 8 | 3 |  |  |  | 是 |  | 出生日期 |
| 6 | department | varchar | 30 | 0 |  |  |  | 是 |  | 所属部门 |
| 7 | title | varchar | 10 | 0 |  |  |  | 是 |  | 职务 |
| 8 | salary | numeric | 9 | 0 |  |  |  | 否 |  | 薪水 |
| 9 | address | varchar | 40 | 0 |  |  |  | 是 |  | 员工住址 |
| 10 | telephone | varchar | 15 | 0 |  |  |  | 是 |  | 员工电话 |
| 11 | email | varchar | 20 | 0 |  |  |  | 是 |  | 员工邮箱 |

(2)会员（Member）实体集。其属性集一：会员编号（memberNo）、登录密码（memPassword）、姓名（MEMName）、性别（sex）、出生年月（birthday）、电话（telephone）、电子邮箱（email）、邮编（zipCode）、购书总额（totalAmount）、会员等级（memLevel）和享受折扣（discount）等。

**表名：Member**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **序号** | **列名** | **数据类型** | **长度** | **小数位** | **标识** | **主键** | **外键** | **允许空** | **默认值** | **说明** |
| 1 | memberNo | char | 9 | 0 |  | 是 |  | 否 |  | 会员编号 |
| 2 | memPassword | varchar | 10 | 0 |  |  |  | 是 |  | 登录密码 |
| 3 | memName | varchar | 12 | 0 |  |  |  | 是 |  | 会员姓名 |
| 4 | sex | char | 1 | 0 |  |  |  | 是 |  | 会员性别 |
| 5 | birthday | datetime | 8 | 3 |  |  |  | 是 |  | 出生日期 |
| 6 | telephone | varchar | 15 | 0 |  |  |  | 是 |  | 会员电话 |
| 7 | email | varchar | 20 | 0 |  |  |  | 是 |  | 会员邮箱 |
| 8 | address | varchar | 40 | 0 |  |  |  | 是 |  | 会员住址 |
| 9 | zipCode | char | 6 | 0 |  |  |  | 是 |  | 邮政编码 |
| 10 | totalAmount | numeric | 9 | 0 |  |  |  | 是 |  | 购书总额 |
| 11 | memLevel | char | 1 | 0 |  |  |  | 是 |  | 会员等级 |
| 12 | discount | float | 8 | 0 |  |  |  | 是 |  | 享受折扣 |

(3)图书（book）实体集。其属性有：书号（ISBN）、书名（bookTitle）、作者(author)、出版年份(publishYear)、版次(version)、类别（category）、库存数量（stockNumber）、单价（price）、售价（salePrice）、内容简介（introduction）和目录（catalog）等。

1. **表名：Book**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **序号** | **列名** | **数据类型** | **长度** | **小数位** | **标识** | **主键** | **外键** | **允许空** | **默认值** | **说明** |
| 1 | ISBN | nvarchar | 50 | 0 |  | 是 |  | 否 |  | 书号 |
| 2 | bookTitle | nvarchar | 200 | 0 |  |  |  | 否 |  | 书名 |
| 3 | author | nvarchar | 200 | 0 |  |  |  | 否 |  | 作者 |
| 4 | publishDate | datetime | 8 | 3 |  |  |  | 是 |  | 出版年份 |
| 5 | version | int | 4 | 0 |  |  |  | 是 |  | 版次 |
| 6 | stockNumber | int | 4 | 0 |  |  |  | 是 |  | 库存数量 |
| 7 | price | money | 8 | 4 |  |  |  | 否 |  | 单价 |
| 8 | salePrice | money | 8 | 4 |  |  |  | 否 |  | 售价 |
| 10 | introduction | nvarchar | 0 | 0 |  |  |  | 是 |  | 分类号 |
| 11 | catalog | nvarchar | 0 | 0 |  |  |  | 是 |  | 内容简介 |
| 13 | categoryId | int | 4 | 0 |  |  | 是 | 是 |  | 目录 |
| 14 | pressNo | int | 4 | 0 |  |  |  | 是 |  | 出版社编号 |
| 15 | BookUrl | nvarchar | 50 | 0 |  |  |  | 是 |  |  |

（4）出版社（Press）实体集。其属性有：出版社编号（pressNo）、出版社名称（pressTitle）、地址（address）、邮编（ZipCode）、联系人（contactPerson）、联系电话（telephone）、传真（fax）和电子邮箱（email）等。

**表名：Press**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **序号** | **列名** | **数据类型** | **长度** | **小数位** | **标识** | **主键** | **外键** | **允许空** | **默认值** | **说明** |
| 1 | pressNo | int | 4 | 0 |  | 是 |  | 否 |  | 出版社编号 |
| 2 | pressName | varchar | 200 | 0 |  |  |  | 是 |  | 出版社名称 |
| 3 | adress | varchar | 40 | 0 |  |  |  | 是 |  | 出版社地址 |
| 4 | zipCode | char | 6 | 0 |  |  |  | 是 |  | 邮政编码 |
| 5 | contactPerson | varchar | 12 | 0 |  |  |  | 是 |  | 联系人 |
| 6 | telephone | varchar | 15 | 0 |  |  |  | 是 |  | 联系电话 |
| 7 | fax | varchar | 15 | 0 |  |  |  | 是 |  | 传真 |
| 8 | email | varchar | 20 | 0 |  |  |  | 是 |  | 电子邮箱 |

（5）配送公司（Company）实体集。其属性有：公司编号（companyNo）、公司名称（companyTitle）、公司地址（address）、邮编（zipCode）、联系人（contactPerson）、电话（telephone）、传真（fax）和电子邮箱（email）等。

**表名：Company**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **序号** | **列名** | **数据类型** | **长度** | **小数位** | **标识** | **主键** | **外键** | **允许空** | **默认值** | **说明** |
| 1 | compantNo | char | 12 | 0 |  | 是 |  | 否 |  | 公司编号 |
| 2 | companyName | varchar | 20 | 0 |  |  |  | 是 |  | 公司名称 |
| 3 | address | varchar | 40 | 0 |  |  |  | 是 |  | 公司地址 |
| 4 | zipCode | char | 6 | 0 |  |  |  | 是 |  | 邮政编码 |
| 5 | contactPerson | varchar | 12 | 0 |  |  |  | 是 |  | 联系人 |
| 6 | telephone | varchar | 15 | 0 |  |  |  | 是 |  | 联系电话 |
| 7 | fax | varchar | 20 | 0 |  |  |  | 是 |  | 传真 |
| 8 | email | varchar | 20 | 0 |  |  |  | 是 |  | 电子邮箱 |

（6）留言（Message）实体集。其属性有：留言编号(messageNo)、内容（messagecontent）和发布时间（releaseDate）等。

**表名：Message**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **序号** | **列名** | **数据类型** | **长度** | **小数位** | **标识** | **主键** | **外键** | **允许空** | **默认值** | **说明** |
| 1 | messageNo | char | 10 | 0 |  | 是 |  | 否 |  | 留言编号 |
| 2 | employeeNo | char | 8 | 0 |  |  | 是 | 否 |  | 回复职员编号 |
| 3 | memberNo | char | 9 | 0 |  |  |  | 是 |  | 发布者编号 |
| 4 | releaseDate | datetime | 8 | 3 |  |  |  | 是 |  | 发布时间 |
| 5 | messageContent | varchar | 100 | 0 |  |  |  | 是 |  | 发布内容 |
| 6 | replyContent | varchar | 100 | 0 |  |  |  | 是 |  | 回复内容 |
| 7 | replyDate | datetime | 8 | 3 |  |  |  | 是 |  | 回复时间 |

(7)发票（invoice）实体集。其属性有：发票号（invoiceNo）、发票单位（invoiceUnit）和发票金额（invoiceSum）等。

**表名：Invoice**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **序号** | **列名** | **数据类型** | **长度** | **小数位** | **标识** | | **主键** | | **外键** | **允许空** | **默认值** | **说明** |
| 1 | invoiceNo | char | 10 | 0 | |  | | 是 |  | 否 |  | 发票编号 |
| 2 | invoiceUnit | varchar | 40 | 0 | |  | |  |  | 是 |  | 发票单位 |
| 3 | invoiceSum | numeric | 9 | 0 | |  | |  |  | 是 |  | 发票金额 |

整体E-R模型：

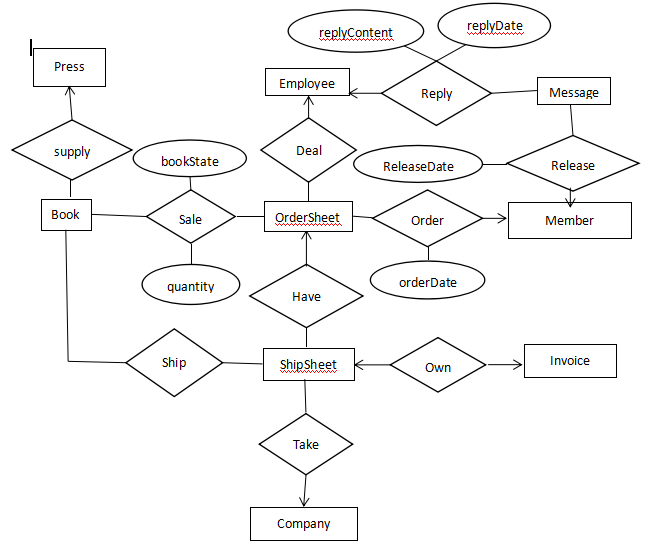


图1-1

## 5.2 项目系统设计模块

### 5.2.1 项目框架简介

本系统采用.NET三层框架来实现。ASP.NET的三层架构(DAL,BLL,UI),

BLL是业务逻辑层(Business Logic Layer),DAL是数据访问层(Data Access Layer),UI 是用户界面接口 (User Interface)。图形表示三层结构如下：

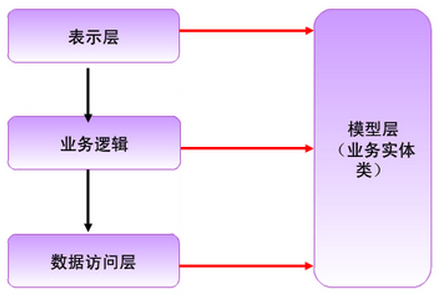


图1-2

（1）三层体系架构  
①表示层(USL)：主要表示WEB方式，也可以表示成WINFORM方式。如果逻辑层相当强大和完善，无论表现层如何定义和更改，逻辑层都能完善

地提供服务。  
②业务逻辑层(BLL)：主要是针对具体的问题的操作，也可以理解成对数据层的操作，对数据业务逻辑处理。如果说数据层是积木，那逻辑层就

是对这些 积木的搭建。  
③数据访问层(DAL)：主要是对原始数据([数据库](http://lib.csdn.net/base/14)或者文本文件等存放数据的形式)的操作层，而不是指原始数据，也就是说，是对数据的操作，

而不是数据库，具体为业务逻辑层或表示层提供数据服务.  
（2）具体区分  
①表示层：主要对用户的请求接受，以及数据的返回，为客户端提供应用程序的访问。  
②业务逻辑层：主要负责对数据层的操作，也就是说把一些数据层的操作进行组合。  
③数据访问层：主要看你的数据层里面有没有包含逻辑处理，实际上他的各个函数主要完成各个对数据文件的操作，而不必管其他操作。  
（3）总结  
三层结构是一种严格分层方法，即数据访问层(DAL)只能被业务逻辑层(BLL)访问，业务逻辑层只能被表示层(USL)访问，用户通过表示层将请求传送给业务逻辑层，业务逻辑层完成相关业务规则和逻辑，并通过数据访问层访问数据库获得数据，然后按照相反的顺序依次返回将数据显示

在表示层。有的三层结构还加了Factory、Model等其他层，实际都是在这三层基础上的一种扩展和应用.一个简单的三层结构程序一般包括DAL BLL WEB Model几个项目，它们的相互引用关系如下：  
①WEB引用 BLL,Model  
②BLL引用 DAL,Model  
③DAL引用Model  
④Model无引用

5.2.2 项目框架具体实现

1.按照三层框架模式搭建框架，每一张表都有对应的实体模型，数据库操作管理及业务管理如下：

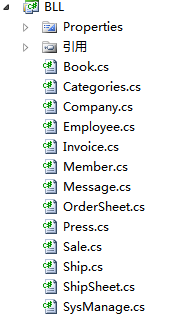
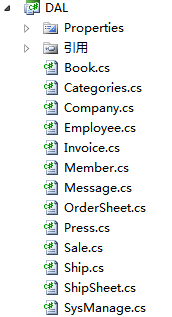


图1-3

2.整体项目结构如下：

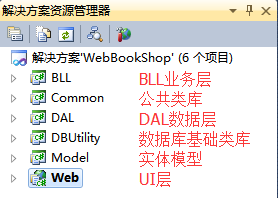


图1-4

### 5.2.2 购物流程

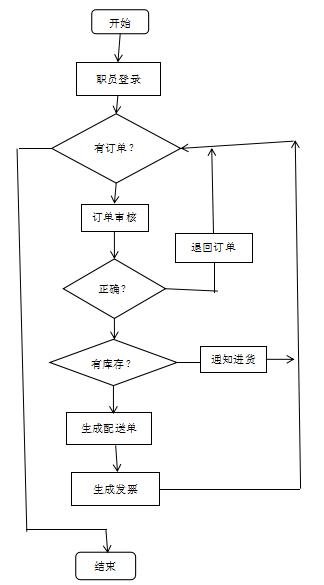
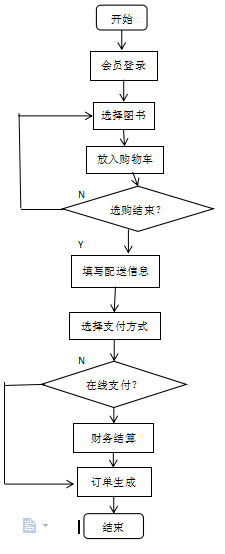


图1-5

### 5.2.3 项目运行效果及测试结果

1.主页：



图1-6

2.会员注册：

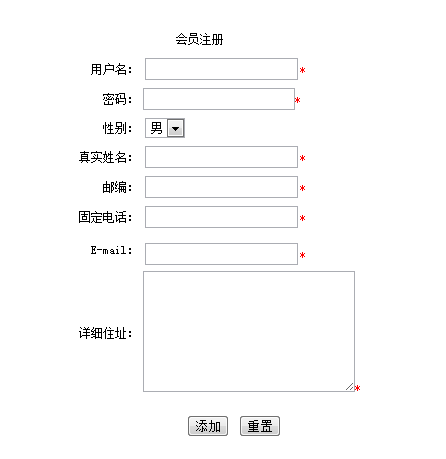


图1-7

3.会员登录：



图1-8

4购物车：

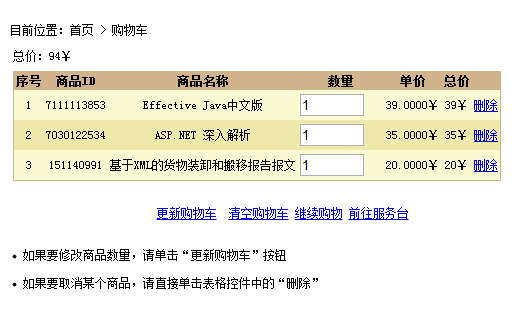


图1-9

5.服务台：

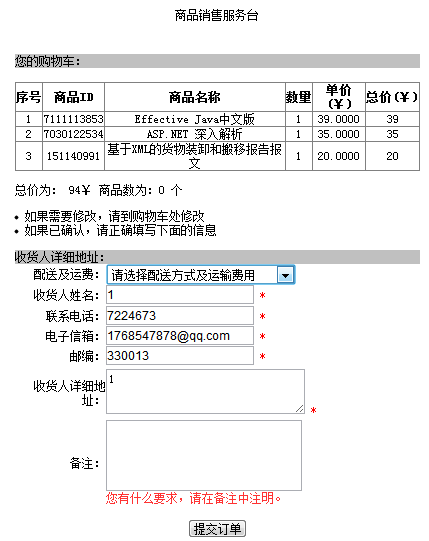


图1-10

6.选择支付方式：



图1-11

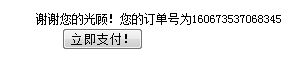




图1-12

7.后台登陆：



图1-13

8.后台界面：



图1-14

## 5.3 项目系统核心代码

#region 执行简单SQL语句

/// <summary>

/// 执行SQL语句，返回影响的记录数

/// </summary>

/// <param name="SQLString">SQL语句</param>

/// <returns>影响的记录数</returns>

public static int ExecuteSql(string SQLString)

{

using (SqlConnection connection = new SqlConnection(connectionString))

{

using (SqlCommand cmd = new SqlCommand(SQLString, connection))

{

try

{

connection.Open();

int rows = cmd.ExecuteNonQuery();

return rows;

}

catch (System.Data.SqlClient.SqlException e)

{

connection.Close();

throw e;

}

}

}

}

public static int ExecuteSqlByTime(string SQLString, int Times)

{

using (SqlConnection connection = new SqlConnection(connectionString))

{

using (SqlCommand cmd = new SqlCommand(SQLString, connection))

{

try

{

connection.Open();

cmd.CommandTimeout = Times;

int rows = cmd.ExecuteNonQuery();

return rows;

}

catch (System.Data.SqlClient.SqlException e)

{

connection.Close();

### throw e;

}

}

}

}

/// <summary>

/// 执行带一个存储过程参数的的SQL语句。

/// </summary>

/// <param name="SQLString">SQL语句</param>

/// <param name="content">参数内容,比如一个字段是格式复杂的文章，有特殊符号，可以通过这个方式添加</param>

/// <returns>影响的记录数</returns>

public static int ExecuteSql(string SQLString, string content)

{

using (SqlConnection connection = new SqlConnection(connectionString))

{

SqlCommand cmd = new SqlCommand(SQLString, connection);

System.Data.SqlClient.SqlParameter myParameter = new System.Data.SqlClient.SqlParameter("@content", SqlDbType.NText);

myParameter.Value = content;

cmd.Parameters.Add(myParameter);

try

{

connection.Open();

int rows = cmd.ExecuteNonQuery();

return rows;

}

catch (System.Data.SqlClient.SqlException e)

{

throw e;

}

finally

{

cmd.Dispose();

connection.Close();

}

}

}

/// <summary>

/// 执行带一个存储过程参数的的SQL语句。

/// </summary>

/// <param name="SQLString">SQL语句</param>

/// <param name="content">参数内容,比如一个字段是格式复杂的文章，有特殊符号，可以通过这个方式添加</param>

/// <returns>影响的记录数</returns>

public static object ExecuteSqlGet(string SQLString, string content)

{

using (SqlConnection connection = new SqlConnection(connectionString))

{

SqlCommand cmd = new SqlCommand(SQLString, connection);

System.Data.SqlClient.SqlParameter myParameter = new System.Data.SqlClient.SqlParameter("@content", SqlDbType.NText);

myParameter.Value = content;

cmd.Parameters.Add(myParameter);

try

{

connection.Open();

object obj = cmd.ExecuteScalar();

if ((Object.Equals(obj, null)) || (Object.Equals(obj, System.DBNull.Value)))

{

return null;

}

else

{

return obj;

}

}

catch (System.Data.SqlClient.SqlException e)

{

throw e;

}

finally

{

cmd.Dispose();

connection.Close();

}

}

}

public static DataSet ExecuteProc(string strProcName, string paramemterName, string strKeyWord)

{

using (SqlConnection connection = new SqlConnection(connectionString))

{

SqlCommand myCmd = new SqlCommand();

myCmd.Connection = connection;

myCmd.CommandText = strProcName;

myCmd.CommandType = CommandType.StoredProcedure;

SqlParameter key = new SqlParameter("@" + paramemterName.ToString(), SqlDbType.VarChar, 50);//添加参数

key.Value = strKeyWord;

myCmd.Parameters.Add(key);

SqlDataAdapter adapt;

DataSet ds = new DataSet();

try

{

if (myCmd.Connection.State != ConnectionState.Open)

{

myCmd.Connection.Open();

}

adapt = new SqlDataAdapter(myCmd);

adapt.Fill(ds);

return ds;

}

catch (Exception ex)

{

throw new Exception(ex.Message, ex);

}

finally

{

if (myCmd.Connection.State == ConnectionState.Open)

{

myCmd.Connection.Close();

}

}

return ds;

}

}

/// <summary>

/// 执行一条计算查询结果语句，返回查询结果（object）。

/// </summary>

/// <param name="SQLString">计算查询结果语句</param>

/// <returns>查询结果（object）</returns>

public static object GetSingle(string SQLString)

{

using (SqlConnection connection = new SqlConnection(connectionString))

{

using (SqlCommand cmd = new SqlCommand(SQLString, connection))

{

try

{

connection.Open();

object obj = cmd.ExecuteScalar();

if ((Object.Equals(obj, null)) || (Object.Equals(obj, System.DBNull.Value)))

{

return null;

}

else

{

return obj;

}

}

catch (System.Data.SqlClient.SqlException e)

{

connection.Close();

throw e;

}

}

}

}

public static object GetSingle(string SQLString, int Times)

{

using (SqlConnection connection = new SqlConnection(connectionString))

{

using (SqlCommand cmd = new SqlCommand(SQLString, connection))

{

try

{

connection.Open();

cmd.CommandTimeout = Times;

object obj = cmd.ExecuteScalar();

if ((Object.Equals(obj, null)) || (Object.Equals(obj, System.DBNull.Value)))

{

return null;

}

else

{

return obj;

}

}

catch (System.Data.SqlClient.SqlException e)

{

connection.Close();

throw e;

}

}

}

}

/// <summary>

/// 执行查询语句，返回SqlDataReader ( 注意：调用该方法后，一定要对SqlDataReader进行Close )

/// </summary>

/// <param name="strSQL">查询语句</param>

/// <returns>SqlDataReader</returns>

public static SqlDataReader ExecuteReader(string strSQL)

{

SqlConnection connection = new SqlConnection(connectionString);

SqlCommand cmd = new SqlCommand(strSQL, connection);

try

{

connection.Open();

SqlDataReader myReader = cmd.ExecuteReader(CommandBehavior.CloseConnection);

return myReader;

}

catch (System.Data.SqlClient.SqlException e)

{

throw e;

}

}

/// <summary>

/// 执行查询语句，返回DataSet

/// </summary>

/// <param name="SQLString">查询语句</param>

/// <returns>DataSet</returns>

public static DataSet Query(string SQLString)

{

using (SqlConnection connection = new SqlConnection(connectionString))

{

DataSet ds = new DataSet();

try

{

connection.Open();

SqlDataAdapter command = new SqlDataAdapter(SQLString, connection);

command.Fill(ds, "ds");

}

catch (System.Data.SqlClient.SqlException ex)

{

throw new Exception(ex.Message);

}

return ds;

}

}

public static DataSet Query(string SQLString, int Times)

{

using (SqlConnection connection = new SqlConnection(connectionString))

{

DataSet ds = new DataSet();

try{

connection.Open();

SqlDataAdapter command = new SqlDataAdapter(SQLString, connection);

command.SelectCommand.CommandTimeout = Times;

command.Fill(ds, "ds")}

catch (System.Data.SqlClient.SqlException ex)

{throw new Exception(ex.Message); }

return ds;

}

}

#endregion

# 六 特别说明

## 6.1 网站的安全性：

保证管理者和注册用户的密码安全，分权限管理，数据库访问控制；管理员应具有一定网络安全及防黑知识。

## 6.2 网站可维护性：

网站管理者须懂得一定的服务器应用、ACCESS数据库应用、硬件维护、IIS配置等方面的技能。

## 6.3 网站的灵活性：

系统应该具有良好的功能可扩充性，以应对未来用户的更高的要求。

## 6.4 故障处理：

系统运行中难免出现一些故障，对此我们提出以下建议和要求：

* 对用户提交的重要资料及时备份。（如：当用户修改注册资料时要及时更新系统资料备份，以便于系统崩溃后资料的正确恢复。）
* 做好数据库和服务器的日常维护工作，出现故障时可与我们联系由我们帮助解决。

# 七 总结

本次课程设计时间有点赶，因为我做的项目有点大，东西有点多，连续几天熬夜做这个课设，从需求分析，数据库设计，业务逻辑和规则，到UI设计等等一系列的问题分析及解决，让我迅速的的掌握整个网站开发流程。我以前没有独立做过一个项目，这次课程设计给了我这个机会，做一个网站原来没有想象的那么困难，只要思路清晰，技术上的问题可以慢慢来，关键是思想，我对我的队员也常说结果不重要，关键是过程，享受这个课设的过程最重要。

此次课设我们做的是网上图书销售系统，数据库设计我们才用教材上的数据设计，并在此基础上做了微妙的改进，在系统上我们采用的是.NET三层框架结构去搭建整个体系结构，这样可以快速的搭建出项目骨架，并快速的完善项目，我觉得采用三层框架是一个明智的选择，让我可以快速的把项目骨架搭建好。缩短了开发周期，节约了时间。

此次的课程设计感触良多，但是最重要的是自己要实践，去自己做何自己想。感谢自己的队友陪我一起熬夜，一起奋斗。

参考文献

[1] 万峰科技.ASP.NET 网站开发四[酷]全书[J].图书情报知识,2005,(5).

[2] 郭瑞军.ASP.NET 数据库开发实例精粹[M].北京：中国电子出版社,2002.

[3] 邓文渊.ASP.NET 网络数据库技术[M].西安：中国铁道出版社,2005.

[4] 刁仁宏.网络数据库原理及应用[J].情报理论与实践,2004,(1).

[5] 张莉,王强.SQL Server 数据库原理及应用教程[M].清华：清华大学出版社出版,2003.

[6] 郭瑞军,李杰,初晓璐. ASP.NET 数据库开发实例精粹[M].西安：电子工业出 版社出版,2003.

[7] 宋昕.ASP.NET 网络开发技术实用教程入门与提高[J].情报杂志,2005,(7).

[8] 顼宇峰.ASP.NET+SQL Server 典型网站建设案例[M].清华：清华大学出版社出版,2006.