摘　要

汽车租赁是近年来一个新兴行业。为规范管理和经营行为，减少经营成本，提高工作效率。开发汽车租赁管理系统软件十分必要。

使用 SQL Server 数据库和 Visual Studio 系列工具实现了一个基于. net 的汽车租赁管理系统。其开发主要包括前端应用程序的开发以及后台数据库的建立和维护两个方面。前台操作包括：浏览查询车辆信息，驾驶员信息，网上公告，进行汽车租赁、还车和订单取消。后台发布新车辆信息，管理修改车辆信息，添加、管理驾驶员信息，添加、管理公告信息，订单管理，用户管理，车辆归还审核等。

该系统以网络为平台，界面简洁，操作简单，易于掌握，简化租赁活动记录手续，提高了租赁周转效率。

关键字：汽车租赁；.net；租赁管理

Abstract

Renting Car is an emerging industry in recent years. In order to regulate the management and working actions, reduce operating costs and improve efficiency. It's so necessary to develop of Renting Car Management system software.

With the use the SQL Server database and Visual Studio family of tools to achieve a .NET Car Rental Management System. Includes the development of front-end application development as well as and establishment and maintenance of two aspects and back-end database. Front desk operations including: browse and query vehicle information, driver information, online bulletin, car rental, return the car and order cancellation. Background release of new vehicle information, manage and modify vehicle information, add and manage driver information, add and manage bulletin information, Order form management, user management, and auditing the vehicles.

The system to the network as a platform, simple interface, simple operation, easy to grasp, to simplify the record formalities for leasing activities, improved the efficiency of rental turnover.

Key Words：Renting Cars ； .net ； Lease management

目　录

第 1 章　引言

伴随着 Internet 在中国的高速发展， 人们广泛地使用计算机技术为自己的学习、工作、娱乐服务，同样， 网上管理也成为了其中的重要部分。同时经济的发展，生活水平的提高，使人们对汽车的需求越来越大。随着生活逐渐富裕，人们已经有能力租车消费，但中国现有汽车保有量难以充分满足需求。其次，购买私家车还不能普及，从个人来说，租车是最好的，购买私家车一次性投入大，购车的手续多，养车费用高，而私家车的利用率一般不高，闲置时间较长。而租车则有很大的灵活性，既不会占用大量资金，也不会出现闲置，车况有保证，出差到外地也可以驾驶车辆。另外不是所有的人都会开车，所以我们提供了驾驶员供人们选择，不会开车的人也可以租车，这就在租车时是否需要驾驶员、要什么样的驾驶员有了选择。而且租赁车车型可选择，客户既能承受，又能满足多样化的需求。而且，从社会角度看，租赁车辆属于公共用车的范畴，它既缓解了现阶段财政控购与企业单位用车之间的矛盾，提高了资金利用率，同时也符合社会车辆总量控制原则，可在一定程度上缓解交通拥挤；从发展角度看，汽车租赁业的发展不仅可以带动中国的新车销售，同时还可以推动中国二手车的经营，为旧车交易注入新的内容；汽车租赁的特殊作用，可以带动多种相关行业的发展，融合产业间的联系，成为第二产业与第三产业间的联系纽带。

得益于以上几个方面的原因，汽车租赁业在我国迅速崛起，业务量也是越来越多，对信息的处理要求也是越来高，传统的汽车管理人员已不满足汽车租赁业务的需求发展。租赁管理就是对车辆信息、驾驶员信息和客户信息的管理，主要包括车辆的基本信息、驾驶员基本信息、定单的管理等。由于这些过程间的关系复杂，有一对一的关系，一对多的关系，也有多对多的关系，所有这些工作使管理工作变得量大而又复杂，以前包括现在还有很多租赁公司采用人工管理，直接导致了错误的产生，服务质量的低下。租赁管理系统的开发使得这一状况得到了根本的改善。

因此我们将发挥计算的庞大的存储空间，高性能的处理效率，高度可靠的数据安全，清晰的可视化数据等这些资源的合理利用，真正达到减少劳动力提高劳动质量的目的。根据目前的情况我设计了一套具有网络功能的汽车租赁管理系统，客户可以通过网络查阅相关的汽车租赁情况，企业的管理人员可以根据客户的实际情况进行管理，具有很强的实用性，即方便又提高了工作效率。

1.1 选题依据及意义

适应市场要求，开发符合企业要求的功能齐全的基于.net 的车辆租赁管理系统并不断加紧完善迫在眉睫，保证企业能通过使用该系统，能够方便、高效地在网上管理车辆。在该基于.net 平台的车辆租赁管理系统下，企业车辆信息的传递和处理、调查和信息收集都将变得更加快捷，也使客户能方便的与企业进行信息交流，及时了解车辆信息、驾驶员信息。同时站在客户的角度，为客户提供了更便捷的车辆租赁环境，人们不受时间的限制，不受空间的限制，不受传统租赁的诸多限制，可以随时随地在网上租赁申请、审签、查询。

选择本题为毕业设计，不仅因为它有着作为一个社会生产项目的实际功用。而且在开发过程中，在指导老师的帮助下，相信能使自己更多的体会开发实践项目的过程，更加深刻掌握本科四年所学习的专业知识，达到到将知识化作生产力的真正目的。

1.2 研究现状及发展态势

1.2.1 国际汽车租赁业务的发展及主要企业的经营模式

自 1908 年福特推出了低价位的 T 型车，使汽车开始进入普通家庭，美国的汽车租赁业开始发展。至今已从八十年代末的 150 亿美元，扩张为 1997 年的 480 亿美元，到 2000 年超过了千亿美元，约占全球租赁业总额的 1/5 左右。从事汽车租赁业务的公司数量也已经达到了 5 千多家。

1.2.1.1 国际汽车租赁业的发展背景

生活环境及工作方式变化：随着世界经济形势的变化和西方国家产业结构的变化，人们的生活方式也在改变。越来越多的家庭、个人选择租赁汽车，认为这样可以节省大笔投资。

生活及工作中个性化需求日益突出：由于工作上的特殊要求和业余生活的丰富多样化，人们在不同时期对于汽车的要求也不尽相同。这时，租赁公司就可以根据顾客的特定需求，为其提供专用的汽车。

汽车作为经营辅助手段的观念成为时代潮流：中小企业利用租赁汽车来完成其经营活动是当代汽车租赁市场呈现出的一个明显的发展趋势。其增长速度以及所占比例，都显示出了汽车租赁在中小企业经营辅助中不可或缺的地位。

1.2.1.2 国际汽车租赁业的经营规模

当前，全球汽车租赁业的运营车辆年需求总数约在 200 万辆左右。主要汽车租赁公司的运营车辆都保持在数十万辆左右，管理着多达数千个遍布全球的租赁站点。在千亿美元的汽车租赁业务中，以欧美国家的租赁市场发展最为成熟。在美国，以租赁形式销售的新汽车占该国汽车总销售量的三分之一左右，并且大部分车为长期租赁，而旧车的租赁业务约为 40 万辆；德国汽车租赁业的运营车辆总数为 250 万辆左右；法国 1997 年以租赁方式使用汽车的人有 300 多万，占法国总人口的 7%；而日本每年的汽车租赁销售规模也达到 200 多万辆，约占全国新汽车销售量的 15%，该比例有不断提高趋势。

1.2.1.3 经营、运作模式

当前在汽车租赁业通行的经营方式是特许经营方式。特许经营是汽车租赁公司授予某人特许经营权，使其加入租赁公司的服务网络，使用租赁公司的品牌和标识，按照租赁公司的统一规范进行业务运作。租赁公司对特许经营点的经营进行监督和指导，并收取特许经营权使用费。

1.2.2 国内汽车租赁

1.2.2.1 发展背景和过程

我国汽车租赁业在 1989 年起源于北京，为了迎合 1990 年在北京举行的亚运会上，国外记者及相关人士在华工作中对交通的便捷、机动、私密性的需求，建立了第一家汽车租赁公司——北京福斯特汽车租赁公司。随后，又分别成立了北京首汽租赁公司、上海安吉租赁公司等。经过 10 多年的发展，国内汽车租赁行业有了长足的发展，从原有仅限在北京、上海、广州等大型城市的汽车租赁业务，发展到了中小城市、乃至县镇。

1.2.2.2 国内汽车租赁业发展特点

(1)国内汽车租赁业正处于起步阶段

国内汽车租赁公司并不具备规模经营的竞争优势：我国汽车租赁公司虽然数量众多，但大多数公司的经营规模小、实力弱，难以抵御市场风险和竞争。

国内汽车租赁服务网络体系没有建立，客户对汽车租赁的认知程度不高：由于目前国内汽车租赁业务仍然采取单点或小范围的经营模式，加之汽车租赁企业自身管理和服务项目等方面的缺陷，使客户对汽车租赁的认知程度普遍不高。

国内汽车租赁企业的管理技术和服务水平与国际先进企业有很大的差距：由于企业的经营规模及资金的限制，使其无法采用国际上通用的一些卓有成效的高

新技术。国际上成熟的多种经营模式也因为各种原因无法推广。而管理技术水平的落后也直接导致了国内汽车租赁企业服务水平的低下。

(2)国内汽车租赁业有巨大的发展潜力

随着中国经济的发展和人民生活水平的提高，人们对汽车消费需求与日俱增，而汽车租赁业也有着良好的外部环境，这些都表明中国已经具备开展大规模汽车租赁业务的条件，汽车租赁业将迎来重大的发展机遇。

总结：不论是国内还是国外，汽车租赁这个行业有着很大的发展潜力，在现在这个信息飞速发展的年代，传统的手工管理方式不但浪费人力、时间，而且管理复杂，易出差错。基于计算机技术，汽车租赁管理系统把一些繁琐的数据计算、信息处理化作为简单的指令操作。完全实现数据信息的电算化管理，彻底把人从手工管理中解放出来。只有使用汽车租赁系统才有利于提高汽车租赁公司的劳动生产率，节约生产成本，增加经济效益。 目前，国内外汽车租赁公司早已认识到这个问题的重要性，早已不满足传统的管理方式，都在使用汽车租赁管理信息系统，来提高工作效率和经济效益。因此，国内外都对汽车租赁管理信息系统进行了更深入的研究，提高改善汽车租赁系统，使之使用起来更加便捷，更符合实用性。

1.3 研究目标及意义

使用SQL Server数据库和Visual Studio系列工具设计完成的汽车租赁管理系统，主要好处是一方面可以方便租赁车辆信息共享，管理员上传租赁车辆的描述信息，图片，用户就可以在线浏览，对所属的租赁车辆有更感官的了解。同时管理员也可以发布一系列的公告来告知用户车辆信息。另一方面就是通过 Internet 网广泛平台，以及计算机庞大的存储空间，高性能的处理效率，高度可靠的数据安全等优点，准确的记录租赁信息，显示公告和租赁情况，生成表给管理者直观的显示。将本系统应用到租赁汽车过程中，会大大增加租赁会员的数量，及时得到更新的租赁信息，第一时间了解车辆信息等信息。方便管理员记录租赁情况，减少人工统计的时间，节省成本，加快汽车租赁的周期。

1.4 主要理论

系统采用了B/S(浏览器，服务器)架构作为信息的共享模式，以TCP/IP协议集作为网络平台基础，以．NET Framework辅以ASP．NET、C#等技术作为网络活动应用的具体开发手段，以SQL技术作为数据库连接方法。这一方案的选择，不但管理方便，而且有利于系统功能的扩展

第 2 章 系统分析

2.1 系统开发环境概述

本系统以 ASP.NET 技术为基础、C#为开发语言、Microsoft Visual 2010 为开发环境、Microsoft SQL Server 2008 为数据存储。运行环境的配置包括 Microsoft .NET Framework，Internet 信息服务版本 IIS, Internet Explorer 9.0。

2.1.1 .NET Framework 简介

.NET Framework 是支持生成和运行下一代应用程序和 XML Web services 的内部 Windows 组件。.NET Framework 旨在实现下列目标：

提供一个一致的面向对象的编程环境，而无论对象代码是在本地存储和执行，还是在本地执行但在Internet上分布，或者是在远程执行的。

提供一个将软件部署和版本控制冲突最小化的代码执行环境。

提供一个可提高代码(包括由未知的或不完全受信任的第三方创建的代码)执行安全性的代码执行环境。

提供一个可消除脚本环境或解释环境的性能问题的代码执行环境。

使开发人员的经验在面对类型大不相同的应用程序(如基于Windows的应用程序和基于web的应用程序)时保持一致.

按照工业标准生成所有通信，以确保基于．NET Framework的代码可与任何其他代码集成。

.NET Framework具有两个主要组件：公共语言运行库和．NET Framework类库。公共语言运行库(CLR)正是．NET Framework的核心所在。顾名思义，CLR就是一个运行期环境，使用不同语言编写的应用程序都可以在这里运行并且互不干扰——即所谓的“跨语言互用”(cross-language interoperability)。.NET Framework的另一个主要组件是类库，它是一个综合性的面向对象的可重用类型集合，可以使用它开发多种应用程序，这些应用车厢包括传统的命令行或图形用户界面（GUI），也包括基于ASP.NET所提供的最新创新的应用程序（如 Web窗体和 XML Web services）。

.NET Framework可由非托管组件承载，这些组件将公共语言运行库加载到它们的进程中并启动托管代码的执行，从而创建一个可以同时利用托管和非托管功能的软件环境。.NET Framework不但提供若干个运行库宿主，而且还支持第三方运行库宿主的开发

2.1.2ASP.NET 简介

ASP.net 是 Microsoft.net 的一部分，作为战略产品，不仅仅是 Active Server Page (ASP) 的下一个版本；它还提供了一个统一的 Web 开发模型，其中包括开发人员生成企业级 Web 应用程序所需的各种服务。ASP.NET 的语法在很大程度上与 ASP 兼容，同时它还提供一种新的编程模型和结构，可生成伸缩性和稳定性更好的应用程序，并提供更好的安全保护。可以通过在现有 ASP 应用程序中逐渐添加 ASP.NET 功能，随时增强 ASP 应用程序的功能。

ASP.NET 是一个已编译的、基于 .NET 的环境，可以用任何与 .NET 兼容的语言（包括 Visual Basic .NET、C# 和 JScript .NET.）创作应用程序。另外，任何 ASP.NET 应用程序都可以使用整个 .NET Framework。开发人员可以方便地获得这些技术的优点，其中包括托管的公共语言运行库环境、类型安全、继承等等。

ASP.net 提供了稳定的性能、优秀的升级性、更快速的开发、更简便的管理、全新的语言以及网络服务。贯穿整个 ASP.net 的主题就是系统帮用户做了大部分不重要的琐碎的工作。

新的 ASP.net 引入受管代码(Managed Code)这样一个全新概念，横贯整个视窗开发平台。受管代码在 NGWS Runtime 下运行，而 NGWS Runtime 是一个时间运行环境，它管理代码的执行，使程序设计更为简便。

2.1.3 C#简介

C#语言出人意表的简单，只有大约80个关键字和12种内置数据类型，但C#在实现现代编程概念方面却给人留下了深刻印象。C#是建立在C++和JAVA这样优秀语言的基础上的，它涵盖了对现代语言所能纳入的结构式、基于组件式、面向对象式编程的支持。

任何面向对象语言的核心在于支持对类的定义和处理。类定义了新的类型，可以扩展语言以创造更适于解决具体问题的模型。C#中有声明新的类及其方法和性质的关键字，还含有实现面向对象编程三大支柱：封装、继承和多态的关键字。

在C#中，与类的定义有关的一切都可在声明本身中找到。C#的类定义并不需要独立的头文件或IDL(接口定义语言)文件。而且，C#支持新的XML风格的内嵌文档，大大简化了软件的在线和印刷品参考文档的制作工作。

C#支持接口(interface)，一种与其所指定的服务的类订立合同(contract)的方式。在C#中，类只能从一个父类继承，但可以实现多个接口。在实现接口时，C#类实际上也继承了要承诺了要提供接口所规定的功能。

C#语言从C和C++演变而来，它是给那些愿意牺牲C++一点底层功能，以获得更方便和更产品化的企业开发人员而创造的。C#现代、简单、面向对象和类型安全。尽管它借鉴了C和C++的许多东西，但是在一些诸如名字空间、类、方法和异常处理等特定领域，它们之间还存在着巨大的差异。

2.1.4 数据库技术

自从数据库系统出现以来，人们从来没有停止过对数据库访问的要求。而随着Internet动态技术的发展，人们又提出了在网络环境下使用数据库的问题。许多数据库语言对诸如C语言或Fortran语言都提供了开发接口，这使得用户便于使用。

2.1.4.1 数据库系统的结构

数据库系统的简单结构如图2-1所示。图中的数据库是数据的汇集，它以一定的组织形式存于存储介质上。数据库管理系统(DBMS)是管理数据库的软件，它实现数据库系统的各种功能。应用指以数据库为基础的各种应用程序，应用程序必须通过DBMS访问数据库．数据库是共享的，需要有人进行数据库的规划、设计、协调、维护和管理等工作。应用程序、数据库管理系统、数据库和数据库管理员构成了数据库系统

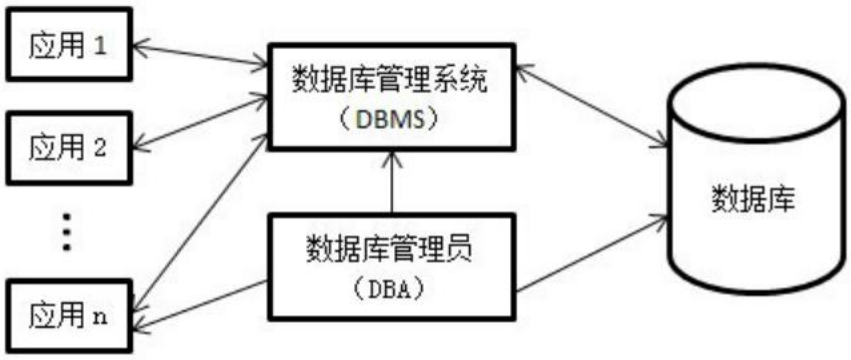

图2-1数据库系统

2.1.4.2 数据库管理系统

数据库管理系统(DBMS)是管理数据库中数据的应用软件，负责在逻辑视图和物理视图之间进行转换。主要包括整体组织结构、存储方式、查询方法、安全管理以及数据的完整性。

2.1.4.3 关系型数据库

数据模型是用来描述数据的一组概念和定义，它包括两个方面：

(1)数据的静态特性：指数据的基本结构、数据间的联系和数据中的约束。

(2)数据的动态特性：指定义在数据上的操作。

传统的数据模型包括层次数据模型、网状数据模型和关系数据模型。在层次数据模型和网状数据模型中，应用的主要数据结构是树结构和系结构，由于这些结构难以掌握和运用，这些模型的软件开发效率低。在关系模型中，信息被组织成若干张二维表的结构，每一张二维表称为一个关系，每个表中的信息用来描述客观世界中的一件事情。关系模型中的一些基本概念包括：

表(Table)，也称关系，由表名、列名及若干行组成。

列(Field)，也称字段、域或属性。表中的每一列都包含一类信息。表中列的顺序与要表达的信息无必要的联系，因此列是无序的。

行(Row)，也称纪录．表中每一行由若干字段组成，描述一个对象的信息。每个字段描述了该对象的某种性质或属性。

码(Key)，也称关键字．表中的某个属性组，它们的组唯一地标识一行。

值域，属性的取值范围。

(3)E．R数据模型

传统数据模型的基本结构是纪录，而人们对现实世界的认识往往以某个事、物或概念为单位。这些可以相互区别的事、物或概念统称为实体(Entity)。实体所具有的特征称之为属性。实体与实体之间的关系抽象为联系(Relationship)。这种面向现实世界，以实体、属性、联系为其抽象概念的模型就是E—R数据模型。

2.3.4.4 SQLSever2008 简介

SQL Server 2008 是一个重大的产品版本，它推出了许多新的特性和关键的改进，使得它成为至今为止的最强大和最全面的 SQL Server 版本。

与 SQL Server 2005 相比，SQL Server 2008 可以对整个数据库、数据文件和日志文件进行加密，而不需要改动应用程序。进行加密使公司可以满足遵守规范和及其关注数据隐私的要求。简单的数据加密的好处包括使用任何范围或模糊查询搜索加密的数据、加强数据安全性以防止未授权的用户访问、还有数据加密。这些可以在不改变已有的应用程序的情况下进行。

SQL Server 2008 基于 SQL Server 2005，并提供了更可靠的加强了数据库镜像的平台。新的特性包括：页面自动修复，SQL Server 2008 通过请求获得一个从镜像合作机器上得到的出错页面的重新拷贝，使主要的和镜像的计算机可以透明的修复数据页面上的 823 和 824 错误；提高了性能，SQL

Server 2008 压缩了输出的日志流，以便使数据库镜像所要求的网络带宽达到最小。

SQL Server 2008 由一系列相互协作的组件组成，能满足最大的 Web 站点和企业数据处理系统存储和分析数据的需要。

SQL Server 2008 支持 Web，通过 Web 可以查询、分析和处理数据。从浏览器通过防火墙可方便而安全地访问数据，并可对有格式文档执行快速全文检索。分析和链接联机分析处理(OLAP)多维数据集，即使在 Web 上也是如此。执行点击流分析，以了解 Web 用户的情况。

2.1.4.5 结构化查询语句 SQL 简介

SQL（Structured Query Language）语言是一种介于关系代数与关系验算之间的结构化查询语句，其功能并不仅仅是查询。SQL 是一个通用的、功能极强的关系数据库语言。SQL 语言集数据查询（Data Query）、数据操纵（Data Manipulation）、数据定义（Data Definition）和数据控制（Data Control）功能于一体，主要特点包括：

1．综合统一

2．高度非过程化

3．面向集合的操作方式

4．以同一种语法结构提供两种使用方式

5．语言简介，易学易用

数据库模式定义语言 DDL (Data Definition Language）用于定义和管理对象，例如数据库、数据表以及视图。DDL 语句通常包括每个对象的 CREATE、ALTER 以及 DROP 命令。

CREATE TABLE 语句用来定义一个基本表；ALTER TABLE 语句用来修改数据表的定义与属性；DROP TABLE 语句用来删除数据表定义以及所有的数据、索引、触发程序、条件约束以及数据表的权限。

数据操作语言 DML(Data Manipulation Language) 利用 INSERT、SELECT、UPDATE 及 DELETE 等语句来操作数据库对象所包含的数据。

INSERT 语句用来在数据表或视图中插入一行数据；UPDATE 语句用来更新或修改一行或多行中的；DELETE 语句用来删除数据表中一行或多行的数据，您也可以删除资料表中的所有数据行。

数据控制语言 DCL（Data Control Language）用于控制对数据库对象操作的权限，它使用 GRANT 和 REVOKE 语句对用户或用户组授予或回收数据库对象的权限。

SELECT 语句用来检索数据表中的数据，而哪些数据被检索由列出的数据行与语句中的 WHERE 子句决定。

2.2 可行性分析

2.2.1 技术可行性分析

本系统采用 Visual Studio 2010作为开发工具，SQL Server 2008作为数据库。Visual Studio 2010 组件包括 Visual Basic .NET 2010、Visual C++ .NET 2010、Visual C# .NET 2010和 Visual F# .NET 2010。Visual Studio 2010 提供了高级开发工具、调试功能、数据库功能和创新功能，帮助在各种平台上快速创建当前最先进的应用程序。为了帮助开发人员迅速创建先进的软件，Visual Studio 2010 提供了改进的语言和数据功能，例如语言集成的查询 (LINQ)，各个编程人员可以利用这些功能更轻松地构建解决方案以分析和处理信息。

Visual Studio 2010 还使开发人员能够从同一开发环境内创建面向多.NET Framework 版本的应用程序。开发人员能够构建面向 .NET Framework 2.0、3.0 或 4.0 的应用程序，意味他们可以在同一环境中支持各种各样的项目。SQL Server 2008使用了先进的数据库结构，可以为大型的 Web 站点和企业应用提供优良的扩展性和可靠的保证。同时，SQL Server 2008还为用户提供了重要的安全性功能，为用户的数据安全提供了可靠的保证。可信任的——使得公司可以以很高的安全性、可靠性和可扩展性来运行他们最关键任务的应用程序。高效的——使得公司可以降低开发和管理他们的数据基础设施的时间和成本。智能的——提供了一个全面的平台，可以在你的用户需要的时候给他发送观察和信息。

使用C#语言编写代码，C#语言出人意表的简单，只有大约80个关键字和12种内置数据类型，但C#在实现现代编程概念方面却给人留下了深刻印象。C#是建立在C++和JAVA这样优秀语言的基础上的，它涵盖了对现代语言所能纳入的结构式、基于组件式、面向对象式编程的支持。

在C#中，与类的定义有关的一切都可在声明本身中找到。C#的类定义并不需要独立的头文件或IDL(接口定义语言)文件。而且，C#支持新的XML风格的内嵌文档，大大简化了软件的在线和印刷品参考文档的制作工作。

C#支持接口(interface)，一种与其所指定的服务的类订立合同(contract)的方式。在C#中，类只能从一个父类继承，但可以实现多个接口。在实现接口时，C#类实际上也继承了要承诺了要提供接口所规定的功能。

C#语言从 C 和 C++演变而来，它是给那些愿意牺牲 C++一点底层功能，以获得更方便和更产品化的企业开发人员而创造的。C#现代、简单、面向对象和类型安全。

2.2.2 需求上可行性分析

经济的发展，生活水平的提高，使人们对汽车的需求越来越大。随着生活逐渐富裕，人们已经有能力租车消费，但中国现有汽车保有量难以充分满足需求。其次，购买私车还不能普及，从个人来说，租车是最好的，买车一次性投入大，购车的手续多，养车费用高，而私车的利用率一般不高，闲置时间较长；出现交通事故后，处理手续太烦琐。而租车则有很大的灵活性，既不会占用大量资金，也不会出现闲置，车况有保证，出差到外地也可以驾驶车辆。而且租赁车车型可选择，客户既能承受，又能满足多样化的需求。而且，从社会角度看，租赁车辆属于公共用车的范畴，它既缓解了现阶段财政控购与企业单位用车之间的矛盾，提高了资金利用率，同时也符合社会车辆总量控制原 则，可在一定程度上缓解交通拥挤；从发展角度看，汽车租赁业的发展不仅可以带动中国的新车销售，同时还可以推动中国二手车的经营，为旧车交易注入新的内容；汽车租赁的特殊作用，可以带动多种相关行业的发展，融合产业间的联系，成为第二产业与第三产业间的联系纽带。

同时伴随着 Internet 在中国的高速发展， 人们广泛地使用计算机技术为自己的学习、工作、娱乐服务，同样， 网上管理也成为了其中的重要部分。传统的车辆租赁管理已不能满足企业和客户的需求，企业的管理人员和客服可能不能及时到场，客户也不能及时的了解车辆的信息、驾驶员的信息，这样不仅浪费时间，而且也有可能直接影响到公司的利益。如何建立一个符合自身的车辆租赁管理系统，在市场经济激烈的竞争中占有一席之地，并拓宽车辆租赁渠道、增加企业效益，成为许多企业的当务之急。于是，开发基于.net 平台的车辆租赁管理系统迫在眉睫。

2.3 小结

在本章中讲了关于系统开发的一些环境，包括了.NET Framework、ASP.NET、C#和数据库。.NET Framework 是支持生成和运行下一代应用程序和 XML Web services 的内部 Windows 组件，ASP.NET 是一个已编译的、基于 .NET 的环境，C#是建立在 C++和 JAVA 这样优秀语言的基础上的，它涵盖了对现代语言所能纳入的结构式、基于组件式、面向对象式编程的支持，数据库是按照数据结构来组织、存储和管理数据的仓库。

本章后半部分还分析了系统的可行性，从技术上来讲现在编程语言众多，运用 Visual Studio 2010 系列工具，使用 C#完全能够实现系统。需求上可行性上来讲也是目前社会所需要的。

第 3 章 系统设计

3.1 概要设计

概要设计中包括系统设计原则、系统模块设计、数据库概要设计等几点介绍。

3.1.1 设计原则

本系统遵循软件工程规定的设计方法和步骤，对系统进行细致地分析研究后，确定了以下基本设计原则：

(1) 实用性：尽量选择成熟实用的技术，使得整个系统有一个安全、稳定的运行环境。

(2) 安全性：为了保障系统平稳正常的运行，以及数据的完整性，整个系统必须有很好的安全性，必须加强数据库的安全保密设计。

(3) 开放性：该系统在建设上必须加强标准化及采用统一的技术规范，以实现网络互联，资源共享，高效运行和科学管理。

3.1.2 系统设计方案

B/S（Browser/Server，浏览器、服务器）模式又称 B/S 结构，它是随着 Internet 技术的兴起，对 C/S 模式应用的扩展。在这种结构下，用户工作界面是通过 IE 浏览器来实现。B/S 模式最大的好处是运行维护比较简便，能实现不同的人员，从不同的地方，以不同的链入方式（比如 LAN，WAN，Internet/Internet 等）访问和操作共同数据。B/S 结构，主要是利用不断成熟的 WWW 浏览器技术，结合浏览器的多种 Script 语言和 ActiveX 技术，用通用浏览器就实现了原来需要复杂专用软件才能实现的强大功能，并节约了开发成本，是一种全新的软件系统构造技术。比起 C/S 模式，该结构系统扩充性高，便于维护和修改。采用 B/S 结构，在软件开发工作中可以主要集中于服务器端的应用程序，节约对客户端的应用程序进行开发所花费的时间和麻烦

Browser/Server的基本结构要实现一个完整的Browser/Server应用系统需要由Brower，Web Server，DB Server三个部分组成。

(1)Browser

浏览器是B/S结构中与用户交互的界面，用于向服务器发送特定的数据或请求，以及接收从服务器发送来的数据。

(2)Server

Server是Web系统中的第三层，在该层中存储了系统中所有需要发布的数据信息，因此为了保证Web站点的快速，高效，一般需要把Server放在硬件配置较好的机器上，它可以和Web Server在同一台计算机上，也可以位于两台、甚至是多台计算机上。

(3)Web Server

B/S结构在客户端(Browse)与服务器(Sewer)之间加了一个Web服务器，Web服务器是实现B/S结构的关键。Web服务器的引入，使得通过浏览器来访问数据

库服务器成为可能，从而免去了开发与维护客户端界面的大量工作。分散在各地的用户。只要安装了浏览器软件，都可以访问数据库服务器。Web服务器作为一种应用服务器，可以将原来分布于客户端或服务器端的应用集中在一起，使系统的结构更见清晰和精细，有利于系统的扩展。Web服务器作为客户端和服务器端的中介，起着沟通与协调二者的作用。

Web Server的作用是接受浏览器的页面请求找到正确的页面并将其回传给浏览器。随着Internet的发展以及用户对Web的要求的提高，Web Serve 的涵义覆盖的范围也有很大的扩展。它需要接受浏览器的页面请求，对其中的非Web页面制作语言的部分解释执行，承担着浏览器与数据库的接口作用。不同的系统平台提供的Web Server是不同的，但是基本功能没有太大差别，只是其实现的功能的方法不一样。浏览器通过将URL发送给Web服务器请求信息，服务器通过返回超文本标记语言(HTML)进行页面响应，页面可以是已经格式化并存储在Web节点中的静态页面，也可能是服务器动态创建以响应用户所提供信息的页面，或者是列出在Web节点上可用的文件和文件夹的页面

3.1.3 系统模块设计

从整体角度出发，本系统主要包括前台和后台 2 个模块。前台分为游客功能模块和会员功能模块，后台分为管理员功能模块和第三方功能模块。

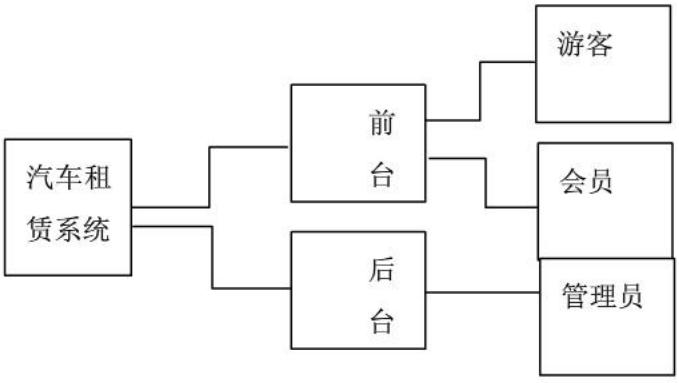

图 3-1 汽车租赁管理系统模块图

3.2 详细设计

3.2.1 模块详细设计

前台有 2 种权限用户（会员与游客）参与，它们分别参与了前台功能的某些模块，下面加以详述模块详细设计分别叙述 3 种不同权限用户的功能。后台有管理员，详细功能描述如下。

游客模块主要功能：

车辆信息查看：游客也可以查看车辆的一些信息。

驾驶员信息查看：游客可以查看所有驾驶员普通信息。

新闻公告查看：游客可以查看管理员发布的新闻公告。

功能结构图如图 3-2 所示。

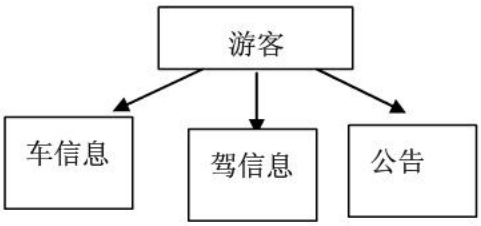

图 3-2 游客模块功能结构图

会员模块主要功能：

车辆信息查看：可以查看车辆的一些信息。

车辆详细信息查看：会员可以查看车辆的详细信息。

驾驶员信息查看：可以查看所有驾驶员信息。

新闻公告查看：可以查看管理员发布的新闻公告。

单独车辆租赁：会员可以单独租赁车辆，提交订单。

带驾驶员的车辆租赁：会员可以租赁车辆并附带租驾驶员，提交订单。

租车订单查看：会员可以查看自己的租车订单。

订单取消：会员可以取消还未通过的订单。

还车申请：会员可通过提交还车让管理员审核。

功能结构图如图 3-3 所示。

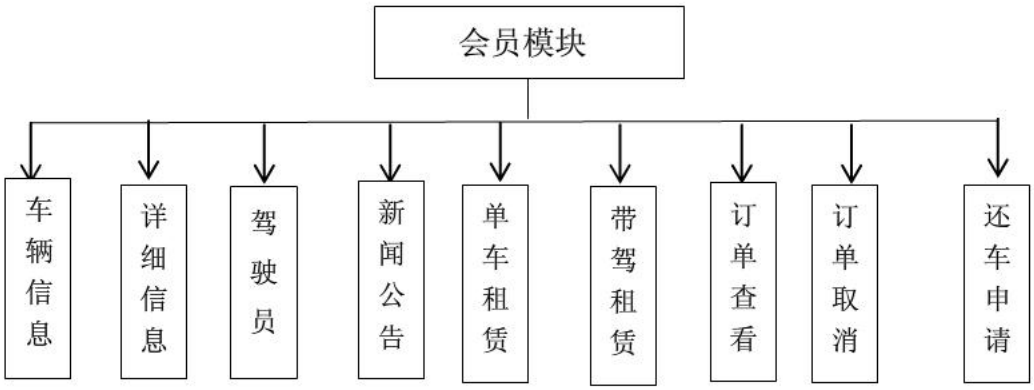

图 3-3 会员模块功能结构图

管理员模块主要功能：

车辆信息管理：主要实现对车辆信息的添加、编辑与删除。

驾驶员信息管理：主要实现驾驶员信息的添加、编辑与删除。

新闻公告管理：主要实现新闻公告的添加、编辑与删除。

用户资料管理：主要实现用户信息资料的编辑与删除，用户权限提升。

审签管理：主要实现订单的审签审核，订单的修改删除等。

归还审核管理：主要实现会员还车的审核。

功能结构图如图 3-4 所示。

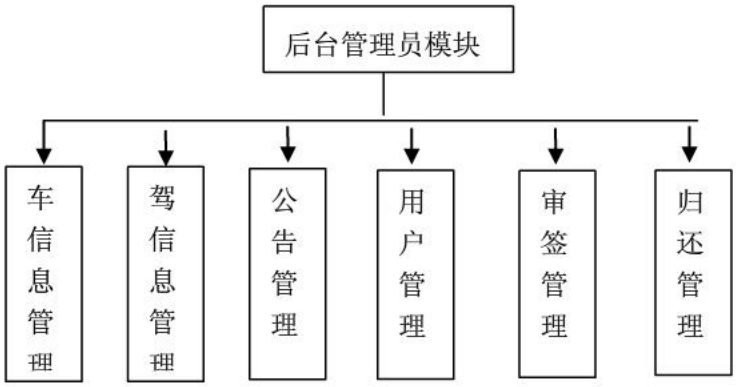

图 3-4 后台管理功能结构图

3.2.2 数据库详细设计

根据系统模块概要设计结果，计划设计以下表，分别为用户管理员信息表、车辆信息表、驾驶员信息表、订单信息表、新闻公告信息表具体说明如下：

用户管理员信息表：用来存放用户和管理员信息。

汽车信息表：用来存放汽车信息。

驾驶员信息表：用来存驾驶员信息。

订单信息表：用来存放租赁订单信息。

新闻公告信息表：用来存放新闻公告的信息。

表的数据结构：

表 3-1 用户信息表（User）

字段名	数据类型	长度	描述
UserID	int	4	标识、主键
UserName	varchar	50	用户 ID
UserPwd	varchar	50	用户密码
Email	varchar	50	用户邮箱
Phone	varchar	20	用户联系电话
Name	varchar	10	用户真实姓名
IDCard	varchar	20	用户身份证
UserSex	varchar	2	用户性别
Grade	varchar	10	用户权限

表 3-2 汽车信息表（car）

字段名	数据类型	长度	描述
CarID	int	4	标识、主键
CarName	varchar	50	汽车名称
Leased	int	4	已租车数量
Remnants	int	4	剩余车辆数量
Stock	int	4	车辆总量
Price	int	4	日租价
Description	varchar	max	车辆描述
ImageUrl	varchar	20	车辆图片名

表 3-3 驾驶员信息表（Driver）

字段名	数据类型	长度	描述
DriverID	int	4	标识、主键
DriverName	varchar	10	驾驶员姓名
DriverAge	int	4	驾驶员年龄
DriverPhone	varchar	20	驾驶员电话
DriverEmail	varchar	50	驾驶员 Email
DriverSex	varchar	2	驾驶员性别
DriverAddress	varchar	50	驾驶员住址
Experience	float	10	驾龄
DriverImage	varchar	50	驾驶员头像名
Idle	varchar	10	空闲状态

表 3-4 新闻公告信息表（News）

字段名	数据类型	长度	描述
NewsID	in	4	标识、主键
NewsName	varchar	50	新闻公告名
Newsneirong	varchar	max	新闻公告内容
Newsclick	int	11	新闻公告点击量
Newsdatatime	varchar	50	新闻公告发布日期

表 3-5 租赁订单信息表（Order）

字段名	数据类型	长度	描述
OrderID	int	4	标识、主键
CarID	int	4	车辆信息表内对应的 CarID
DriverID	int	4	驾驶员信息边对应的 DriverID
OrderNUM	float	50	订单号
UserName	varchar	50	用户登录名
Name	varchar	10	用户真实姓名
CarName	varchar	50	车辆名字
LseaeDataTime	varchar	50	租车日期
LeaseDay	int	4	租车天数

ReturnDataTime	varchar	50	车辆应还日期
DriverName	varchar	10	驾驶员姓名
CarImage	varchar	50	汽车图片名
Returned	varchar	10	是否归还车辆
Price	varchar	10	车辆日租价
Adopt	varchar	10	审签是否通过

3.3 小结

本章讲述了车辆租赁系统的设计，包括了概要设计和详细设计。在概要设计中设计原则包括了实用性、安全性和开放性。系统设计方案用了 B/S（Browser/Server，浏览器、服务器）模式，B/S 模式最大的好处是运行维护比较简便，能实现不同的人员，从不同的地方，以不同的连入方式访问和操作共同数据。系统设计模块则包含了前台游客和注册会员，后台的管理员模块。在详细设计中详细地描述了游客模块、注册用户模块和管理员模块的功能和对应的用户的权限问题，并详细地设计了数据库用户管理员信息表、汽车信息表、驾驶员信息表、订单信息表和新闻公告信息表。

第 4 章 系统实现和问题解决

4.1 游客功能实现

无任何权限用户（游客）进入系统能进行注册、车辆查看、驾驶员信息查看和新闻公告查看。

图 4-1 游客模块实现图

图 4-2 游客注册模块实现图

图 4-3 游客和会员信息查看实现图

4.2 会员功能模块实现

4.2.1 会员登录

会员进入系统需要进行身份验证，在系统登陆界面输入输入用户名和密码与数据库中的注册信息进行比较。若通过验证则将进入Default.aspx主页面，如果失败则重新回到登录页面，并且显示错误提示信息。

图 4-4 登录失败显示

```
SqlCommand command = new SqlCommand("select * from [User] where UserName=@UserName and UserPwd=@UserPwd", connection);
                command.Parameters.Add(new SqlParameter("@UserName", userName.Text));
                command.Parameters.Add(new SqlParameter("@UserPwd", userPwd.Text));
```

如果用户名或密码错误则：

```
Response.Write("<script>alert('登录失败，用户名或密码错误！')</Script>");
```

登陆成功：

```
string UserName = userName.Text;//取UserName
FormsAuthenticationTicket t = new FormsAuthenticationTicket(1, UserName,DateTime.Now,DateTime.Now.AddMonths(3), false,UserName,FormsAuthentication.FormsCookiePath);
string encTicket = FormsAuthentication.Encrypt(t);
HttpCookie c = new HttpCookie(FormsAuthentication.FormsCookieName, encTicket);
HttpContext.Current.Response.Cookies.Add(c);
Session[Constants.Sessionuser] = UserName;
Response.Redirect("Default.aspx");
```

产生cookies并写入Session记录登陆信息。

成功登录本系统后，比游客多了租赁汽车，订单查看、订单取消和还车功能。

4.2.2 信息查看模块实现

注册用户可以查看车辆详细信息、驾驶员信息、新闻公告等。这里详细信息查看时通过网页间传参实现指定信息查看。

4.2.2.1 注册用户车辆详细信息查看

注册用户拥有游客没有的车辆详细信息查看功能，可以查看车辆的详细信息。

图 4-5 车辆信息详细查看

4.2.2.2 注册用户驾驶员信息查看

注册用户和游客一样拥有查看驾驶员信息的功能，可以查看全部驾驶员的信息。驾驶员信息包括了驾驶员的编号、姓名、年龄、驾龄、性别、空闲状态和头像。

4.2.2.3 注册用户新闻公告查看

注册用户和游客一样拥有查看新闻公告的功能，可以查看全部新闻公告。新闻公告是有管理员发布的消息，新闻公告可以通过点击新闻标题查看新闻公告的具体内容。

4.2.3 租赁模块实现

当确认租赁车辆后，租赁的记录会出现在租赁管理中，并可以进行订单查询和订单取消的管理。

4.2.3.1 租赁车辆

车辆租赁是会员用户的特有权限，允许注册会员租赁车辆，租赁车辆有 2 中情况，一种是只租赁车辆而不需要驾驶员的，另外一种是既要租车也要配属驾驶员的。

图 4-6 会员租赁单独车辆模块实现图

单车辆租赁不配属司机则在 Order 表写入时把关于 Driver 的写为“无”，租车时间由系统获取当前本地时间。

配属司机RadioButton当为否时，不显示司机表，当选择是的时候，显示司机表并可以选择司机。

选择配属司机时在写入 Order 表时获取用户选择的司机信息，在 Order 表里填入对应信息，如：DriverID、DriverName 等。

图 4-7 会员租赁车辆带司机功能实现图

带驾驶员的车辆租赁部分代码：

首先要将 Driver 表的驾驶员的空闲状态改为忙碌

```
string Driversql = "update Driver set  Idle=@Idle"
                 + " where DriverID=@DriverID";
//构建sql语句
                 SqlHelper dbDriver = new SqlHelper();
                 DbCommand commandD =
dbDriver.GetSqlStringCommond(Driversql);
                 commandD.Parameters.Add(new SqlParameter("@Idle", "忙|
碌"));
```

其次是把 car 表里的余量和已租量改变

```
string carsql = "update car set  Remnants=@Remnants, Leased=@Leased"
                 + " where CarID=@CarID";
                 SqlHelper db = new SqlHelper();
                 DbCommand command1 = db.GetSqlStringCommond(carsql);
                 command1.Parameters.Add(new SqlParameter("@Remnants",
```

```
Remnants));
                command1.Parameters.Add(new SqlParameter("@Leased",
Leased));
```

最后才是把这些收集到的信息写入带 Order 表里

```
string connStr =
ConfigurationManager.ConnectionStrings["db1"].ConnectionString;
                string sql = "insert into
[Order](OrderNUM,UserName,Name,CarName,CarNUM,LeaseDataTime,LeaseDay,
ReturnDataTime,DriverName,Adopt,CarImage,Returned,Price,CarID,DriverI
D)"
                                        + "values
(@OrderNUM,@UserName,@Name,@CarName,@CarNUM,@LeaseDataTime,@LeaseDay,
@ReturnDataTime,@DriverName,@Adopt,@CarImage,@Returned,@Price,@CarID,
@DriverID)";
                SqlConnection connection = new SqlConnection(connStr);
                SqlCommand command = new SqlCommand(sql, connection);
                command.Parameters.Add(new SqlParameter("@CarImage",
CarImage));
                command.Parameters.Add(new SqlParameter("@UserName",
UserName));
                command.Parameters.Add(new SqlParameter("@Price",
Price));
                command.Parameters.Add(new SqlParameter("@CarName",
CarName));
                command.Parameters.Add(new SqlParameter("@Name",
UserName));
                command.Parameters.Add(new SqlParameter("@CarNUM", "
无"));
                command.Parameters.Add(new SqlParameter("@OrderNUM",
OrderNUM));
                command.Parameters.Add(new
SqlParameter("@LeaseDataTime", TimeNow));
                command.Parameters.Add(new SqlParameter("@LeaseDay",
LeasedayTextBox1.Text));
                command.Parameters.Add(new
SqlParameter("@ReturnDataTime", ReturnDataTime));
                command.Parameters.Add(new SqlParameter("@DriverName",
RadioButton2.Checked ? "无" : DriverName));
                command.Parameters.Add(new SqlParameter("@Adopt", "未
通过"));
                command.Parameters.Add(new SqlParameter("@Returned", "
未还"));
                command.Parameters.Add(new SqlParameter("@CarID",
CarID));
```

```
                command.Parameters.Add(new SqlParameter("@DriverID",
DriverID));
                connection.Open();
                command.ExecuteNonQuery();
                connection.Close();
                Response.Write("<script>alert('租车成功!
')</Script>");
```

会员登陆本系统后，租赁车辆，页面上显示你要租车辆的信息，车辆图片，名称，日租价，配司机状况，配置描述。需要填写的信息只有租车天数，其他信息都会从会员注册信息里调取填入。最后提交后还要经由管理员进行审签。审签通过后才完成全部租车，否则租车无效。

4.2.3.2 还车

还车时，系统会告知你还车的类别是正常还车还是提前还车，总的租金，并提交一个还车申请给管理员。管理员根据用户还车情况修改、保留或通过申请，以确认用户租还车是否完成。

用户还车

编号	订单号	用户	姓名	车名	租天	驾驶员	租车时间	还车时间	还车	审核	车照片	
25	2012510103435513	5550818	5550818	测试001	10	测试002	2012/5/10/10:34:35	2012/5/20 10:34:35	未还	通过		还车申请

图 4-8 会员还车功能实现图

4.2.3.3 用户订单查询

注册会员用户还能查看自己租车的订单，这些订单里包括了以前租车通过和未通过，通过后还车了的和未还车的详细信息。

```
String UserName= Convert.ToString(Session[Constants.Sessionuser]);
SqlHelper db = new SqlHelper();
DbCommand command = db.GetSqlStringCommond( “select * from [Order]
where UserName=@UserName” );
```

```
command.Parameters.Add(new SqlParameter("@UserName", UserName));
DataTable table = db.ExecuteDataTable(command);
GridViewAudit.DataSource = table;
GridViewAudit.DataBind();
```

订单查看时取出用户名通过用户名想数据库请求 Order 表关于用户的订单数据，并绑定到 Gridview 上。

图 4-9 会员租赁后订单模块实现图

4.2.3.4 用户订单取消

注册用户有权取消那些还未被管理员通过审签的订单来取消自己租车的失误。

图 4-10 会员订单取消模块实现图

4.2.3.5 会员还车提交

当注册会员租车时间到了并归还了车辆，可以再会员还车里申请还车，等待管理员的审核，如果通过审核，那么这次车辆租赁就完成了。否则则需要管理员和注册会员联系。

图 4-11 会员还车模块实现图

4.3 管理员管理模块实现

4.3.1 发布新车辆实现

填写车辆名称，租车价格，库存，已租，日租价，基本配置，上传车辆图片。添加成功，车辆会出现在车辆信息页面。

图 4-12 发布新车辆实现图

图片处理部分代码：

```
string fullfileName = this.FileUpload1.PostedFile.FileName;
            string fileName =
fullfileName.Substring(fullfileName.LastIndexOf("\\") + 1);
            string type =
fullfileName.Substring(fullfileName.LastIndexOf(".") + 1);
            if (type == "jpg" || type == "gif" || type == "bmp" || type
```

```
== "jpeg")
            {

this.FileUpload1.PostedFile.SaveAs(Server.MapPath("~/up") + "\\" +
fileName);
                this.Image1.ImageUrl = "~/up/" + fileName;
            }
            else
            {
                Response.Write("<script language='javascript'>alert('
图片格式错误！')</Script>");
            }
```

4.3.2 车辆管理实现

读取车辆的信息，对车辆进行更新和修改的操作，说明维修的状况。

图 4-13 车辆管理实现

删除功能部分代码：

```
string id = e.Keys[0].ToString();//获得主键列
            string sql = "delete from car where CarID=" + id;
            SqlHelper db = new SqlHelper();
            DbCommand command = db.GetSqlStringCommond(sql);
            db.ExecuteNonQuery(command);
            bindGrid();
```

编辑部分代码：

```
string CarID = e.Keys[0].ToString();                    //获得主键列
            string CarName = e.NewValues["CarName"].ToString();
//获得编辑后名称
            string Price = e.NewValues["Price"].ToString();
            string Stock = e.NewValues["Stock"].ToString();
            string Description =
e.NewValues["Description"].ToString();
            string ImageUrl = e.NewValues["ImageUrl"].ToString();
```

```
            string Leased = e.NewValues["Leased"].ToString();
            string Remnants = e.NewValues["Remnants"].ToString();
            string sql = "update car set CarName=@CarName, Price=@Price,
Remnants=@Remnants, "
                + " Stock=@Stock, Description=@Description ,
ImageUrl=@ImageUrl , Leased=@Leased"
                + " where CarID=@CarID";
//构建sql语句
            SqlHelper db = new SqlHelper();
            DbCommand command = db.GetSqlStringCommond(sql);
            db.AddInParameter(command, "@ImageUrl", DbType.String,
ImageUrl); //添加参数
            db.AddInParameter(command, "@CarName", DbType.String,
CarName);
            db.AddInParameter(command, "@Price", DbType.Double,
Price);
            db.AddInParameter(command, "@Stock", DbType.Double,
Stock);
            db.AddInParameter(command, "@Description", DbType.String,
Description);
            db.AddInParameter(command, "@CarID", DbType.String,
CarID);
            db.AddInParameter(command, "@Leased", DbType.String,
Leased);
            db.AddInParameter(command, "@Remnants", DbType.String,
Remnants);
            db.ExecuteNonQuery(command);
            GridView1.EditIndex = -1;
//取消编辑状态
            bindGrid();
```

4.3.3 驾驶员管理实现

驾驶员信息上传

上传新的驾驶员基本信息，包括驾驶员姓名、年龄、性别、电话、Email、住址、驾龄和头像照片。

图 4-14 驾驶员信息上传实现

4.3.4 订单管理实现

根据用户提交的租赁订单，审签，通过审批租赁成功，否则不予通过。

另外还提供了对订单进行修改和删除的功能，使得一些非正常的订单不能影响租赁。

图 4-15 订单管理实现图

4.3.4 新闻公告管理实现

新闻公告管理模块式管理员对新闻公告编辑和删除的模块，包括了对新闻公告标题、内容和事件的修改，并且还有新闻公告上传功能，上传新闻公告事件是获取系统当前时间。

图 4-16 新闻公告上传实现图

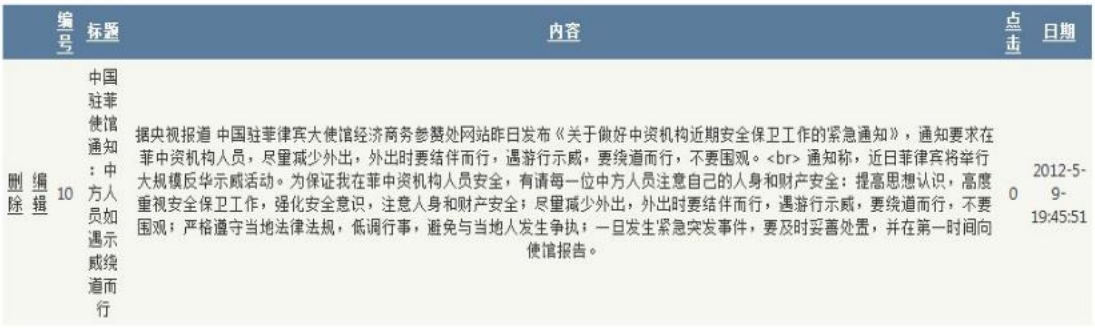

图 4-17 新闻公告编辑、删除实现图

4.3.5 用户信息管理实现

用户信息管理也是非常有必要的，管理员可以通过用户信息管理用户注册，修改用户的权限，及时删除一些非正常用户等。

图 4-18 用户信息编辑、删除实现图

4.4 问题解决

4.4.1 技术问题

4.4.1.1 注册用户

用户注册时需要判断该用户名是否被注册，并把信息反馈给用户。这里我使用了Visval Studio 2010中的验证模块CustomValidator自定义验证，该验证要访问数据库。另外多个验证要验证一个文本框时一般把验证控件的Display属性改为Dynamic让他不显示的时候不占位置。

用户名存在性验证部分代码：

```
string UserName = args.Value;
            string s =
ConfigurationManager.ConnectionStrings["db1"].ConnectionString;
            //创建连接对象
            SqlConnection connection = new SqlConnection(s);
            connection.Open();
            SqlCommand cmd = new SqlCommand("select Count(*) from [User]
where UserName='" + UserName + "'", connection);
            count = Convert.ToInt32(cmd.ExecuteScalar());
            connection.Close();
            if (count > 0)
            {
                args.IsValid = false;
            }
            else
            {
                args.IsValid = true;
            }
```

邮箱及身份证格式也有硬性要求，我使用格式验证，邮箱格式为\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*，身份证格式为\d{17}[\d|X]|\d{15}。

图 4-19 用户注册实现图

4.4.1.2 上传图片

在上传汽车信息和驾驶员信息时要上传图片，图片有格式要求，在这里我使用了系统时间来重命名图片并保存在指定文件夹下，而数据库则保存该图片的相对地址，以便取用。

```
try
        {
string filePath = null;
                        //以当前时间修改图片的名字或创建文件夹的名字
string modifyFileName = DateTime.Now.Year.ToString() +
DateTime.Now.Month.ToString() + DateTime.Now.Day.ToString() +
DateTime.Now.Hour.ToString() + DateTime.Now.Minute.ToString() +
DateTime.Now.Second.ToString() + DateTime.Now.Millisecond.ToString();
            //获得站点的物理路径
            string uploadFilePath = null;
            uploadFilePath =
System.Web.HttpContext.Current.Server.MapPath(".") + path;
            //获得文件的上传的路径
            string sourcePath = PathToName(PostedFile.FileName);
            //判断上传文件是否为空
            if (sourcePath == "" || sourcePath == null)
            {
                //message("您没有上传图片！");
```

```
            return null;
        }
        //获得文件扩展名
        string tFileType =
sourcePath.Substring(sourcePath.LastIndexOf(".") + 1);
        //获得上传文件的大小
        long strLen = PostedFile.ContentLength;
        //分解允许上传文件的格式
        string[] temp = fileType.Split('|');
        //设置上传的文件是否是允许的格式
        bool flag = false;
        //判断上传文件大小
        if (strLen >= sizes)
        {

            message("上传的文件不能大于" + sizes + "KB");
            return null;
        }
        //判断上传的文件是否是允许的格式
        foreach (string data in temp)
        {
            if (data == tFileType)
            {
                flag = true;
                break;
            }
        }
        //如果为真允许上传，为假则不允许上传
        if (!flag)
        {
            message("目前本系统支持的格式为:" + fileType);
            message("文件上传不成功!");
            return null;
        }
        System.IO.DirectoryInfo dir = new
System.IO.DirectoryInfo(uploadFilePath);
        //判断文件夹否存在,不存在则创建
        if (!dir.Exists)
        {
            dir.Create();
        }
        filePath = uploadFilePath + modifyFileName + "." + tFileType;

        PostedFile.SaveAs(filePath);
```

```
            filePath =  modifyFileName + "." + tFileType;

            return filePath;

        }
        catch
        {
            //异常
            message("出现未知错误！");
            return null;
        }
```

4.4.1.3Gridview 控件中编辑、删除、排序和翻页等

Gridview 控件是我的程序中使用最多的控件，而且在使用中遇到问题最多的控件，该控件有很多功能，其中编辑、删除、排序和翻页是常用的功能。

Gridview控件中编辑有编辑项属性，我在属性中双击RowEditing出现编辑触发事件，同理删除和排序也是一样。

Gridview控件中还有翻页功能，一页中显示太多的内容回到石页面过大，适当的选择每页显示信息条数是非常有必要的。这里要求激活PageIndexChanging属性，并在Gridview属性里添加按钮。具体翻页的代码如下：

```
<PagerTemplate>
                当前第:
                <asp:Label ID="LabelCurrentPage" runat="server"
Text="<%# ((GridView)Container.NamingContainer).PageIndex +
1 %>"></asp:Label>
                页/共:
                <asp:Label ID="LabelPageCount" runat="server"
Text="<%#
((GridView)Container.NamingContainer).PageCount %>"></asp:Label>
                页
                <asp:LinkButton ID="LinkButtonFirstPage"
runat="server" CommandArgument="First" CommandName="Page"
ForeColor="White"

Visible='<%#((GridView)Container.NamingContainer).PageIndex != 0 %>'>
首页</asp:LinkButton>
                <asp:LinkButton ID="LinkButtonPreviousPage"
runat="server" CommandArgument="Prev" ForeColor="White"
                    CommandName="Page" Visible='<%#
((GridView)Container.NamingContainer).PageIndex != 0 %>'>上一页
</asp:LinkButton>
                <asp:LinkButton ID="LinkButtonNextPage" runat="server"
CommandArgument="Next" CommandName="Page" ForeColor="White"
```

```
                    Visible='<%#
((GridView)Container.NamingContainer).PageIndex !=
((GridView)Container.NamingContainer).PageCount - 1 %>'>下一页
</asp:LinkButton>
                <asp:LinkButton ID="LinkButtonLastPage" runat="server"
CommandArgument="Last" CommandName="Page" ForeColor="White"
                    Visible='<%#
((GridView)Container.NamingContainer).PageIndex !=
((GridView)Container.NamingContainer).PageCount - 1 %>'>尾页
</asp:LinkButton>
                转到第
                <asp:TextBox ID="txtNewPageIndex" runat="server"
Width="20px" Text='<%# ((GridView)Container.Parent.Parent).PageIndex +
1 %>' />页
                <asp:LinkButton ID="btnGo" runat="server"
CausesValidation="False" CommandArgument="-2"
                    CommandName="Page" Text="GO" ForeColor="White" />
            </PagerTemplate>
```

PageIndexChanging触发事件中代码：

```
  // 得到该控件t
            GridView theGrid = sender as GridView;
            int newPageIndex = 0;
            if (e.NewPageIndex == -3)
            {
                //点击了Go按钮
                TextBox txtNewPageIndex = null;
                GridViewRow pagerRow = theGrid.BottomPagerRow;

                if (pagerRow != null)
                {
                    //得到text控件
                    txtNewPageIndex =
pagerRow.FindControl("txtNewPageIndex") as TextBox;
                }
                if (txtNewPageIndex != null)
                {
                    //得到索引
                    newPageIndex = int.Parse(txtNewPageIndex.Text) - 1;
                }
            }
            else
            {
```

```
            //点击了其他的按钮
            newPageIndex = e.NewPageIndex;
        }
        //防止新索引溢出
        newPageIndex = newPageIndex < 0 ? 0 : newPageIndex;
        newPageIndex = newPageIndex >= theGrid.PageCount ?
theGrid.PageCount - 1 : newPageIndex;
        //得到新的值
        theGrid.PageIndex = newPageIndex;
        //重新绑定
        bindGrid();
```

4.4.1.4 简单搜索功能

在设计时发现如果车辆上架太多用户要租赁某一类的车时往往会花费很多时间，所以制作了一个简单搜索功能，实现对车辆名字的模糊简单搜索，该搜索也要对数据库进行访问，在查询数据库时用“Like”而不是“=”来查询就会得到模糊搜索功效。其中也用到了页面间传参。具体用法如下：

在 textbox 框里输入模糊字点击搜索按钮触发如下

```
string varrid = null ;
        try
        {
            varrid = TextBox3.Text.ToString();

        }
        catch
        {
            Response.Write("输入车名有误请返回！");
            Response.End();
        }
        Response.Redirect("carmodify.aspx?CarName=" + varrid);
```

在carmodify.aspx页面里我们string key = Request.QueryString["CarName"].ToString();来获得刚才页面传过来的参数，实现页面间传参。

4.4.1.5 游客、注册用户和管理员身份判断及访问权限

由于有注册用户必须的信息才能租车和管理员后台管理，所以身份判定和访问权限是必要的，游客身份判定是未登录用户即为游客，而登陆前台的用户为注册用户，前台登陆不需要用户有权限。管理员身份判定是有后台登陆获得，后台并非所有用户都能登陆，只有在数据库里用户信息里的权限项里有权限的用户财位管理员。

具体做法是：用户前台登陆判断用户名和密码，正确则记录
Session[Constants.Sessionuser] = UserName;
否则Session[Constants.Sessionuser] = Null;
后台管理员登陆要判断用户名和密码并且还要判断用户的权限是否时管理权限才能登陆，登陆成功后记录Session[Constants.SessionAdmin] = userNameTextBox.Text;否则Session[Constants.SessionAdmin] = Null;

访问权限的设置是访问后台在admin/路径下的页面时在Page_Load里要先检测。

```
if (Session[Constants.SessionAdmin] == null)
            {
                Response.Redirect("../Default.aspx");
            }
```

如果Session[Constants.SessionAdmin] == null;则跳转到前台主页面

车辆租赁是也是这样的判断。

```
if (Session[Constants.Sessionuser] == null)
            {
                Response.Redirect("login.aspx");
                Response.Write("<script>alert('请先登录')</Script>");
             }
```

判断用户是否登陆，如果没有登陆则跳转到登陆页面并提醒用户登陆后在操作。

4.4.2 逻辑处理

车辆租赁是有注册用户提交汽车租赁订单来确定自己所租赁车辆和是否有驾驶员，驾驶员是谁，租车几天等信息。这份订单保存在服务器数据库里等待管理员审签。管理员通过查询得到注册用户提交的订单，通过判断来确认是否通过审签并派发车辆。车辆租赁给注册用户。当注册用户需要还车时并实际还了车，那么在网站上提交还车申请，等待管理员审核，若审核通过则这次车辆租赁完成，若审核未通过则需要管理员联系注册用户协商并完成车辆租赁。

4.5 小结

本章讲了汽车租赁系统的实现和一些问题的解决。系统实现了关于游客功能、注册会员功能和管理员功能模块的具体实现。问题解决中主要解决了用户注册、图片上传、Gridview 使用、简单搜索和权限方面的技术问题，另外也解决了汽车租赁过程的逻辑问题，使得系统逻辑清晰。

第 5 章　系统测试

5.1 测试目的

软件测试是软件代码生成后必不可少的一步，软件测试包括功能代码的测试、系统功能的完整性测试、性能测试、安全性测试、数据库的一致性测试等，测试的目的是尽可能多的发现软件制作过程中的错误，通过测试使软件的错误减少，使系统的可靠性进一步提高。

在该阶段主要是对应用程序的编码进行调试，排除存在的程序逻辑错误。并且往数据库中装入测试用数据。由于数据库中通常包含着一个组织内多个部门的数据，这些数据的格式、规格都可能不同，所以在加载之前要对这些数据作严格地检验、整理，将数据装入数据库。

5.2 测试内容

根据需求分析划定系统测试的功能范围，即需要的功能是否已经在系统中得到充分体现。测试包括软件的界面要求、功能体现、性能要求、稳定性要求、是否达到操作便捷、灵活性要求、安全性要求；各模块之间的关系是否与需求分析中的一致，测试整个系统的功能和性能，检验其是否满足的需求。

5.3 具体测试

软件测试是软件开发过程的重要组成部分，为了发现错误而执行程序的过程。软件测试在软件生存周期中横跨两个阶段：通常在编写出每一个模块之后就对它做必要的测试，称为单体测试。编码和单体测试属于软件生存期的同一个阶段。下边介绍系统具体测试用例：

5.3.1 游客功能模块测试

1、测试目的：区别游客和会员、管理员的功能，游客浏览关键页面时需要有一定的权限，使用某项功能时需要登录成为会员。

2、预期的结果：没有注册为本站会员的游客，不享有租车，还车，订单查询和订单取消的功能，也不能进入后台管理员界面。

3、具体测试方法：不登陆状态下访问系统所有页面，包括手动输入地址和使用页面自有地址。

4、实际的结果：游客只可以浏览，查看车辆信息、驾驶员信息和新闻公告

的功能，其他页面会提示游客登录，并且不能在未登陆状态下进入后台管理员界面。达到了预期结果。

5.3.2 会员功能模块测试

1、测试目的：区别会员和管理员的功能，会员浏览管理员页面时需要有一定的权限，使用某项功能时需要管理员身份。

2、预期的结果：注册会员只享有前台查看车辆信息、驾驶员信息、新闻公告、租车、还车、订单查询和订单取消的功能，不能进入后台管理员界面。

3、具体测试方法：会员登陆状态下访问系统所有页面，包括手动输入地址和使用页面自有地址。

4、实际的结果：会员只享有前台所有功能，可以浏览，查看车辆信息、驾驶员信息、新闻公告、租车、还车、订单查询和订单取消的功能，其他页面会提示需要管理员身份，并且不能在非管理员身份状态下进入后台管理员界面。达到了预期结果。

5.3.3 管理员后台功能模块测试

1、测试目的：测试管理员的功能，管理员应享有全部系统的管理权限。

2、预期的结果：管理员享有前台查看车辆信息、驾驶员信息、新闻公告、租车、还车、订单查询和订单取消的功能，并且能进入后台管理员界面，享有管理员所有功能，包括车辆信息发布和管理、驾驶员信息发布和管理、新闻公告发布和管理、订单管理、用户信息管理。

3、具体测试方法：管理员登陆状态下访问系统所有页面，包括手动输入地址和使用页面自有地址。

4、实际的结果：管理员享有前台和后台所有功能。达到了预期结果。

5.4 小结

本章讲了汽车租赁系统的测试问题，具体测试包括了：功能体现、性能要求、稳定性要求、是否达到操作便捷、灵活性要求、安全性要求。通过测试，对数据库的操作正常，系统功能设计满足需求，基本实现了汽车租赁的需求，各功能模块运行正常。

第 6 章　结束语

在本文的最后，对本系统进行简单概括，对系统的开发过程进行总结，并描述本系统的不足和待优化之处。

6.1 系统总结

通过本次设计我体会到了做软件开发工作首先要根据软件工程来制定出一套适合自己的软件开发时间和进程，严格按照时间和进程来进行操作，然后是一定要做好系统分析，系统开发的关键在于对系统进行需求分析，在开发一个系统的初期，首先要对系统的整个工作流程，需要实现的功能，系统需求等进行充分的分析。只有在做好需求分析的前提下才能顺利的完成整个系统的开发。其次是开发数据库应用系统一定要做好数据库的分析与设计，完全整理通顺系统中的数据流图，理解数据库的逻辑结构设计，在数据库关系表的设计中要注意运用数据库规范化理论，并对数据库关系表进行规范化使其满足高的范式要求。

经过几个多月的设计和开发，汽车租赁管理系统已经开发完毕，其功能符合基本需求：管理员管理汽车租赁订单信息、管理车辆信息、管理驾驶员信息和用户信息。会员在线浏览租赁信息资源，查看车辆信息、驾驶员信息和新闻公告，搜索车辆、租赁车辆、订单取消等。

6.2 系统不足

6.2.1 租赁身份验证

租赁车辆时的身份验证情况，现实生活中，通常以用户的户口本或者身份证作为抵押，必须涉及人工的方式，本系统简化了这个过程，由会员租赁车辆时自己填写身份证信息，相当于身份证抵押的情况，缴纳押金后进行租赁。

6.2.2 时间限制问题

(1) 会员租赁车辆填写的预租信息，缴纳押金通过后台管理员审批方能租赁。在此期间可能出现只是填写租赁信息，不缴纳押金的状况。这样会耽误车辆的租赁情况，应该对填写订单至缴纳押金的时间做一个限制。

(2) 对车辆租赁中超期租赁未设任何提醒和惩罚，未对超期的日期进行限制，应该对超期天数有一个限制。

6.3 系统改进思想

6.3.1 时间限制

(1) 填写订单至缴纳押金的时间做一个时间限制，超过时间则租车订单取消。

(2) 车辆租赁给登录用户提供一个还车提醒功能，并设置租赁车辆超期天数有时间限制，超期处罚等。

6.3.2 人性化细节功能

添加一些细节功能，比如管理员结算，对数据库进行备份，添加租赁车辆的方法和租赁过程中的小常识。

致谢

为期四年的大学本科的学习生活即将结束，在此，感谢我的家人，是他们的支持和鼓励使我有了四年美好的大学时光。我要感谢所有曾经教导过我的老师和关心过我的同学，感谢他们在我学习成长过程中所给予我的帮助。

这次毕业设计能成功完成，要特别感谢我的指导老师惠孛老师，感谢惠孛老师对我的作品提出的建议和指导。在整个设计阶段，惠孛老师在我有困难的时候细心为我指导，给了我很大的帮助。在论文完成之际，谨向我尊敬的导师表示诚挚的谢意，感谢惠孛老师对我的关心和帮助。在整个系统的开发过程中，同学和朋友给了我很大的帮助。感谢给予我帮助的同学和朋友们，他们在我的整个设计过程中给我提供了大量的技术指导和实践机会。

衷心的感谢你们！

参考文献

[1]《asp.net 技术方案宝典》出版社：人民邮电出版社 作者：明日科技 张跃廷 张宏宇 出版日期：2007 年 2 月

[2]《asp.net 网络编程自学手册》出版社：人民邮电出版社 作者：张跃廷 房大伟 苏宇 出版日期：2007 年 3 月

[3]《asp.net 从入门到精通》出版社：机器工业出版社 作者：张昌龙 辛永平 出版日期：2007 年 3 月

[4]《asp.net 范例完全自学手册》 出版社：人民邮电出版社 作者：张跃廷 房大伟 梁冰 出版日期：2007 年 3 月

[5]《asp.net 网络数据库开发》 出版社：电子工业出版社 作者：武新华 孙健 肖庆和 出版日期：2007 年 2 月

[6]《asp.net 动态网站开发》 出版社：电子工业出版社 作者：马军 王灏 出版日期：2007 年 1 月

[7]《asp.net 典型模块大全》 出版社：人民邮电出版社 作者：明日科技 张跃廷 张宏宇 出版日期：2007 年 1

[8]《asp.net 网络编程》 出版社：清华大学出版社 作者：孙继磊 出版日期：2006 年 11 月

外文文献翻译

英文原文

Basic Security Practices for Web Applications

Even if you have limited experience with and knowledge of application security, there are basic measures that you should take to help protect your Web applications. The following sections in this topic provide minimum-security guidelines that apply to all Web applications.General Web Application Security Recommendations;Run Applications with Minimum Privileges ;Know Your Users; Guard Against Malicious User Input;Access Databases Securely;Create Safe Error Messages;Keep Sensitive Information Safely;Use Cookies Securely;Guard Against Denial-of-Service Threats.

1. General Web Application Security Recommendations

Even the most elaborate application security can fail if a malicious user can use simple ways to gain access to your computers. General Web application security recommendations include the following: Back up data often and keep your backups physically secure.Keep your Web server physically secure so that unauthorized users cannot gain access to it, turn it off, physically steal it, and so on.Use the Windows NTFS file system, not FAT32. NTFS offers substantially more security than FAT32. Protect the Web server and all of the computers on the same network with strong passwords.Follow best practices for securing Internet Information Services (IIS). Close any unused ports and turn off unused services.Run a virus checker that monitors site traffic.Use a firewall.Learn about and install the latest security updates from Microsoft and other vendors.Use Windows event logging and examine the logs frequently for suspicious activity. This includes repeated attempts to log on to your system and excessive requests against your Web server.

2. Run Applications with Minimum Privileges

When your application runs, it runs within a context that has specific privileges on the local computer and potentially on remote computers. For information about configuring application identity, see Configuring ASP.NET Process Identity.To run with the minimum number of privileges needed, follow these guidelines: Do not run your application with the identity of a system user (administrator).Run the application in the context of a user with the minimum practical privileges. Set permissions (ACLs, or Access Control Lists) on all the resources required for your application. Use the most restrictive setting. For example, if practical in your application, set files to be read-only. For a list of the minimum ACL permissions required for the identity of your ASP.NET application, see ASP.NET Required Access Control Lists (ACLs).Keep files for your Web application in a folder below the application root. Do not allow users the option of specifying a path for any file access in your application. This helps prevent users from getting access to the root of your server.

3. Know Your Users

In many applications, it is possible for users to access the site without having to provide

credentials. If so, your application accesses resources by running in the context of a predefined user. By default, this context is the local ASPNET user (Windows 2000 or Windows XP) or NETWORK SERVICE user (Windows Server 2003) on the Web server.To restrict access to users who are authenticated, follow these guidelines: If your application is an intranet application, configure it to use Windows Integrated security. This way, the user's login credentials can be used to access resources.If you need to gather credentials from the user, use one of the ASP.NET authentication strategies. For an example, see the ASP.NET Forms Authentication Overview.

4. Guard against Malicious User Input

As a general rule, never assume that input you get from users is safe. It is easy for malicious users to send potentially dangerous information from the client to your application. To help guard against malicious input, follow these guidelines: In forms, filter user input to check for HTML tags, which might contain script. For details, see How to: Protect Against Script Exploits in a Web Application by Applying HTML Encoding to Strings.Never echo (display) unfiltered user input. Before displaying untrusted information, encode HTML to turn potentially harmful script into display strings.Similarly, never store unfiltered user input in a database.If you want to accept some HTML from a user, filter it manually. In your filter, explicitly define what you will accept. Do not create a filter that tries to filter out malicious input; it is very difficult to anticipate all possible malicious input.Do not assume that information you get from the header (usually via the Request object) is safe. Use safeguards for query strings, cookies, and so on. Be aware that information that the browser reports to the server (user agent information) can be spoofed, in case that is important in your application.If possible, do not store sensitive information in a place that is accessible from the browser, such as hidden fields or cookies.

5. Access Databases Securely

Databases typically have their own security. An important aspect Web application security is designing a way for the application to access the database securely. Follow these guidelines: Use the inherent security of your database to limit who can access database resources. The exact strategy depends on your database and your application: If practical in your application, use Windows Integrated security so that only Windows-authenticated users can access the database. Integrated security is more secure than using SQL Server standard security.If your application uses anonymous access, create a single user with very limited permissions, and perform queries by connecting as this user.Do not create SQL statements by concatenating strings that involve user input. Instead, create a parameterized query and use user input to set parameter values. If you must store a user name and password somewhere to use as the database login credential, store them securely. If practical, encrypt or hash them. For details, see Encrypting and Decrypting Data.

6. Create Safe Error Messages

If you are not careful, a malicious user can deduce important information about your

application from the error messages it displays. Follow these guidelines: Do not write error messages that echo information that might be useful to malicious users, such as a user name.Configure the application not to show detailed errors to users. If you want to display detailed error messages for debugging, check first that the user is local to the Web server. For details, see How to: Display Safe Error Messages.Use the customErrorsconfiguration element to control who can view exceptions from the server.Create custom error handling for situations that are prone to error, such as database access.

7. Keep Sensitive Information Safely

Sensitive information is any information that you need to keep private. A typical piece of sensitive information is a password or an encryption key. If a malicious user can get to the sensitive information, then the data protected by the secret is compromised. Follow these guidelines: If your application transmits sensitive information between the browser and the server, consider using Secure Sockets Layer (SSL). Use Protected Configuration to secure sensitive information in configuration files such as the Web.config or Machine.config files. For more information, see Encrypting Configuration Information Using Protected Configuration.If you must store sensitive information, do not keep it in a Web page, even in a form that you think people will not be able to view (such as in server code).Use the strong encryption algorithms supplied in the System Security Cryptography namespace.

8 . Use Cookies Securely

Cookies are an easy and useful way to keep user-specific information available. However, because cookies are sent to the browser's computer, they are vulnerable to spoofing or other malicious use. Follow these guidelines: Do not store any critical information in cookies. For example, do not store a user's password in a cookie, even temporarily. As a rule, do not store any sensitive information in a cookie that. Instead, keep a reference in the cookie to a location on the server where the information is located. Set expiration dates on cookies to the shortest practical time you can. Avoid permanent cookies if possible.Consider encrypting information in cookies. Consider setting the Secure and HttpOnly properties on your cookies to true.

9. Guard against Denial-of-Service Threats

An indirect way that a malicious user can compromise your application is by making it unavailable. The malicious user can keep the application too busy to service other users, or if nothing else can simply crash the application. Follow these guidelines: Close or release any resource you use. For example, always close data connections and data readers, and always close files when you are done using them.Use error handling (for example, try/catch blocks). Include a finally block in which you release resources in case of failure.Configure IIS to use throttling, which prevents an application from using a disproportionate amount of CPU. Test size limits of user input before using or storing it.Put size safeguards on database queries to help guard against large queries using up system resources.You can also use the RequestLengthDiskThreshold property in to reduce the memory overhead of large uploads and form posts.

外文译文

Web 应用程序的基本安全做法

即使您对应用程序安全性的体验和了解非常有限，也应采取一些基本措施来保护您的 Web 应用程序。以下各部分提供了适用于所有 Web 应用程序的最低安全性准则。常规 Web 应用程序安全性建议；使用最少特权运行应用程序；了解您的用户；防止恶意用户的输入；安全地访问数据库；创建安全的错误消息；保证敏感信息的安全；安全地使用 Cookie；防止拒绝服务威胁。

1.常规 Web 应用程序安全性建议；

如果恶意用户可以使用简单方法进入您的计算机，即使是最精心设计的应用程序安全性也会失败。常规 Web 应用程序安全性建议包括以下内容：经常备份数据，并将备份存放在安全的场所；将您的 Web 服务器放置在安全的场所，使未经授权的用户无法访问它、关闭它、带走它，等等。使用 Windows NTFS 文件系统，不使用 FAT32。NTFS 的安全性比 FAT32 高得多。使用不易破解的密码，保护 Web 服务器和同一网络上的所有计算机的安全。遵循用于确保 Internet 信息服务 (IIS) 安全的最佳做法。关闭任何不使用的端口并关闭不使用的服务。运行监视网站通信量的病毒检查程序。使用防火墙。了解和安装来自 Microsoft 和其他供应商的最新安全更新。使用 Windows 事件日志记录，并且经常检查这些日志，以查找可疑活动。这样的活动包括：反复尝试登录您的系统，以及向您的 Web 服务器发出数量巨大的请求。

2.使用最少特权运行应用程序

当您的应用程序运行时，它运行在一个具有本地计算机（还可能是远程计算机）的特定特权的上下文中。有关配置应用程序标识的信息，请参见 配置 ASP.NET 进程标识。若要以最少特权运行，请遵循以下准则： 不要以系统用户（管理员）身份运行应用程序。在具有最少实用特权的用户上下文中运行应用程序。 设置应用程序所需的所有资源上的权限（ACL 或访问控制列表）。使用最严格的设置。例如，如果在您的应用程序中是可行的，则将文件设置为只读。有关 ASP.NET 应用程序标识所需的最少 ACL 权限的列表，请参见 ASP.NET 必需的访问控制列表 (ACL)。将您的 Web 应用程序的文件保存在应用程序根目录下的一个文件夹中。不要让用户指定在应用程序中进行文件访问的路径。这样有助于防止用户访问服务器的根目录。

3.了解您的用户

在许多应用程序中，用户有可能不必提供凭据即可访问网站。如果是这样，则您的应用程序通过在预定义用户的上下文中运行即可访问资源。默认情况下，此上下文是 Web 服务器上的本地 ASPNET 用户（Windows 2000 或 Windows XP）或 NETWORK SERVICE 用户 (Windows Server 2003)。若要仅允许已授权用户进行访问，请遵循以下准则： 如果您的应用程序是 Intranet 应用程序，则将其配置为使用 Windows 集成安

全性。这样，用户的登录凭据就可以用于访问资源。如果您需要从用户收集凭据，则使用其中一种 ASP.NET 身份验证策略。有关示例，请参见 ASP.NET Forms 身份验证概述。

4.防止恶意用户的输入

通常，决不假定从用户获得的输入是安全的。对恶意用户来说，从客户端向您的应用程序发送潜在危险的信息是很容易的。若要帮助防止恶意输入，请遵循以下准则： 在窗体中，筛选用户输入以查找 HTML 标记，其中可能包含脚本。有关详细信息，请参见 如何：通过对字符串应用 HTML 编码在 Web 应用程序中防止脚本侵入。决不回显（显示）未经筛选的用户输入。在显示不受信任的信息之前，对 HTML 进行编码以将潜在有害的脚本转换为显示字符串。类似地，决不将未经筛选的用户输入存储在数据库中。如果要接受来自用户的一些 HTML，则手动筛选它。在您的筛选器中，显式定义将要接受的内容。不要创建一个试图筛选出恶意输入的筛选器；因为预料到所有可能的恶意输入是非常困难的。不要假定您从标头（通常通过 Request 对象）获得的信息是安全的。对查询字符串、Cookie 等采取安全措施。注意，浏览器向服务器报告的信息（用户代理信息）可以被假冒（如果此信息在您的应用程序中相当重要）。如有可能，不要将敏感信息（如隐藏字段或 Cookie）存储在可从浏览器访问的位置。例如，不要将密码存储在 Cookie 中。

5.安全地访问数据库

数据库通常具有它们自己的安全性。Web 应用程序安全性的一个重要方面是设计一种应用程序安全地访问数据库的方式。请遵循这些指导： 使用数据库的内在安全性来限制可以访问数据库资源的人员。确切的策略取决于您的数据库和应用程序： 如果在您的应用程序中切实可行，请使用 Windows 集成安全性以便只有 Windows 授权的用户才能访问数据库。集成安全性比使用 SQL Server 标准安全性更安全。如果您的应用程序使用匿名访问，请创建具有非常有限的权限的单个用户，并以此用户身份连接来执行查询。不要通过串联涉及用户输入的字符串创建 SQL 语句。相反，创建参数化查询并使用用户输入设置参数值。 如果您必须将用户名和密码存储在某个位置以用作数据库登录凭据，请安全地存储它们。如果可行，请对它们进行加密或计算哈希值。

6.创建安全的错误消息

如果您不小心，恶意用户就可以从应用程序显示的错误消息推断出有关您的应用程序的重要信息。请遵循这些指导： 不要编写会回显可能对恶意用户有用的信息（例如用户名）的错误消息。将应用程序配置为不向用户显示详细错误。如果为进行调试而要显示详细错误消息，请先检查该用户是否为 Web 服务器的本地用户。使用 customErrors 配置元素控制谁可以查看服务器发出的异常。对于容易发生错误的情况（如数据库访问）创建自定义错误处理方式。

7.保证敏感信息的安全

“敏感信息”是需要保密的任意信息。密码或加密密钥即是典型的敏感信息。如果恶意用户可以获得敏感信息，则该信息保护的数据将受到威胁。请遵循这些指导： 如果您的应用程序在浏览器和服务器之间传输敏感信息，请考虑使用安全套接字层(SSL)。使用受保护的配置来确保配置文件（如 Web.config 或 Machine.config 文件）中敏感信息的安全。如果您必须存储敏感信息，即使是以您认为人们将无法看到它的形式（如在服务器代码中）进行保存，也不要将它保存在网页中。使用 System.Security.Cryptography 命名空间中提供的强加密算法。

8.安全地使用 Cookie

为了让用户特定的信息保持可用，Cookie 是一种容易而有用的方法。但是，由于 Cookie 会被发送到浏览器所在的计算机，因此它们容易被假冒或用于其他恶意用途。请遵循这些指导： 不要将任何关键信息存储在 Cookie 中。例如，不要将用户的密码存储在 Cookie 中，即使是暂时存储也不要这样做。作为一项原则，不要在 Cookie 中存储任何敏感信息。而是在 Cookie 中保存对信息在服务器上的位置的引用。将 Cookie 的过期日期设置为可以设置的最短时间。尽可能避免使用永久的 Cookie。考虑对 Cookie 中的信息加密。考虑将 Cookie 的 Secure 和 HttpOnly 属性设置为 true。

9.防止拒绝服务威胁

恶意用户危害您的应用程序的一种间接方式是使其不可用。恶意用户可以使应用程序太忙而无法为其他用户提供服务，或者仅仅使应用程序出现故障。请遵循这些指导：关闭或释放您使用的任何资源。例如，在使用完毕后，始终关闭数据连接和数据读取器，而且始终关闭文件。使用错误处理机制（例如，try/catch 块）。包含 finally 块，以便万一失败就可以在其中释放资源。将 IIS 配置为使用调节，这样可以防止应用程序消耗过多的 CPU。 在使用或存储用户输入之前，测试它的大小限制。对数据库查询设置大小保护措施，以防止大型查询耗尽系统资源。 如果文件上载是您的应用程序的一部分，则对它们的大小加以限制。还可以使用 RequestLengthDiskThreshold 属性来减少大型上载和窗体发布所需的内存开销。