江南大学

课 程 设 计 报 告

题 目： 车辆租赁管理子系统

院（系）： 人工智能与计算机学院

专 业: 计算机科学与技术

班 级： 1806

学 号: 1033180624

姓 名： 周余民

指导老师： 钱雪忠

2020年7月6日～2020年7月24日

成绩：

目录

[1 开发环境与开发工具 4](#_Toc32578)

[2 系统需求分析 4](#_Toc15185)

[2.1 系统数据流图 5](#_Toc23234)

[2.2 系统数据字典 5](#_Toc18434)

[3 系统功能分析 7](#_Toc2615)

[4 系统设计 8](#_Toc8103)

[4.1 数据库概念结构设计 8](#_Toc7517)

[4.1.1系统概念模块设计 8](#_Toc13215)

[4.1.2E-R图 9](#_Toc9817)

[4.2 系统功能模块设计 11](#_Toc17049)

[4.2.1模块图： 11](#_Toc4542)

[4.2.2各子模块功能描述 11](#_Toc29969)

[4.3 数据库逻辑结构设计 13](#_Toc25474)

[4.4 数据库物理结构设计及其实现 13](#_Toc28393)

[4.4.1创建数据库及表的SQL命令 13](#_Toc10420)

[4.4.2 表数据的初始化（INSERT命令） 16](#_Toc1728)

[4.4.3 数据库表关系图 19](#_Toc11269)

[5 系统实现 20](#_Toc6377)

[5.1说明 20](#_Toc11094)

[5.2数据库操作通用api 20](#_Toc23346)

[5.3验证登入与判断是否登入 21](#_Toc32486)

[5.3 前端登入页面 22](#_Toc32737)

[5.3.1 HTML代码 22](#_Toc26334)

[5.3.2 JavaScript代码 23](#_Toc10634)

[5.3.3 CSS代码 24](#_Toc6730)

[5.4系统主界面功能菜单 26](#_Toc32433)

[5.4.1 HTML代码 26](#_Toc19131)

[5.4.2 CSS代码 29](#_Toc25423)

[5.5 查询操作 32](#_Toc10796)

[5.5.1 JavaScript代码 33](#_Toc20407)

[5.5.2后端实现查询代码 34](#_Toc11489)

[5.6 添加操作 34](#_Toc14191)

[5.6.1 JavaScript代码 35](#_Toc20306)

[5.6.2后端实现添加操作 36](#_Toc18419)

[5.7 修改操作 36](#_Toc10340)

[5.7.1 JavaScript代码 37](#_Toc18052)

[5.7.2后端实现修改操作 38](#_Toc4735)

[5.8 删除操作 38](#_Toc19817)

[5.8.1 JavaScript代码 39](#_Toc19013)

[5.8.2后端实现删除操作 39](#_Toc27733)

[5.9预约车辆 40](#_Toc6123)

[5.9.1 JavaScript代码 40](#_Toc3271)

[5.9.2后端实现预约操作 41](#_Toc140)

[5.10租用车辆 42](#_Toc17056)

[5.10.1 JavaScript代码 42](#_Toc9968)

[5.10.2后端实现租用操作 43](#_Toc12872)

[5.11 租金统计 44](#_Toc32662)

[5.11.1 JavaScript代码 44](#_Toc16663)

[5.11.2 后端实现租金查询操作 47](#_Toc23046)

[5.12通用CSS 47](#_Toc10712)

[5.13 其他系统功能 49](#_Toc4529)

[5.13.1获取当天日期 49](#_Toc7848)

[5.13.2租金数字格式转换 49](#_Toc2801)

[6 系统测试 50](#_Toc10873)

[7 体会与自我评价 50](#_Toc1254)

# 1 开发环境与开发工具

我已经拥有一台阿里云云服务器，并且安装好了MySQL数据库，也开启了远程连接，可以作为车辆租赁管理子系统所需的数据库，同时也有远程连接MySQL所需的软件Navicat，Navicat是一套快速、可靠并价格相当便宜的[数据库管理](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%AE%A1%E7%90%86/10509024" \t "https://baike.baidu.com/item/navicat/_blank)工具，专为简化数据库的管理及降低系统管理成本而设。它的设计符合[数据库管理员](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%AE%A1%E7%90%86%E5%91%98/1216449" \t "https://baike.baidu.com/item/navicat/_blank)、开发人员及中小企业的需要。Navicat 是以直觉化的[图形用户界面](https://baike.baidu.com/item/%E5%9B%BE%E5%BD%A2%E7%94%A8%E6%88%B7%E7%95%8C%E9%9D%A2/3352324" \t "https://baike.baidu.com/item/navicat/_blank)而建的，可以以安全并且简单的方式创建、组织、访问并共用信息。

为此，基于上述条件可以开发出支持B/S模式的车辆租赁管理子系统，本系统使用Navicat建立和管理所需数据库，使用NodeJS作为后端语言，前端则采用HTML+JavaScript+CSS写好的静态页面展现，在数据渲染方面则采取前端渲染方式，使用Ajax发送请求，获得数据，然后渲染前端静态页面，代码编写则使用Visual Studio Code,这是一款轻量级编辑器，安装好插件能方便快捷有效地开发出车辆租赁管理子系统。

# 2 系统需求分析

随着车辆工业的发展和汽车的普及，车辆租赁成为近年来兴起的一个新兴行业，使用车辆租赁管理系统可以规范企业的管理和经营行为，减少企业的经营成本，提高工作效率。另外，随着电脑的普及与使用，现在的管理也提升了一个档次，渐渐实现了无纸化办公，即从原来的人工记录管理模式转变为电脑一体化管理。原来的数据处理手工操作，工作量大，出错率高，出错后不易更改。租车公司采取手工方式对车辆租借情况进行人工管理，由于信息比较多，信息的管理工作混乱而又复杂，如再要进行查询，就得在众多的资料中翻阅、查找了，造成查询费时、费力。所以对于车辆租赁公司来说就需要一种车辆租赁管理系统。车辆租赁管理系统多是针对车辆租赁的业务处理的一种系统，它主要依赖于数据库，管理层面一般包括基本设置、信息管理、业务处理、查询报表、统计租金以及系统管理等模块，优点在于业务处理模块结合了实际，所以对于日常的信息和车辆出租的相关业务都能做到很好的管理。而车辆租赁信息管理系统提高车辆租赁管理工作效率，做到信息的规范管理，科学统计和快速查询，为车辆租赁公司的管理解决了一大难题。

## 2.1 系统数据流图



## 2.2 系统数据字典

数据流图表达了数据和处理的关系，数据字典则是系统中各类数据描述的集合，是进行详细的数据收集和数据分析所获得的主要成果。数据字典通常包括数据项、数据结构、数据流、数据存储和处理过程五个部分。以下数据字典卡片的形式来举例说明。

1. **“租赁信息”数据结构：**

名称：租赁信息

描述：记录公司的各种租赁信息

定义：租赁信息=客户信息+车辆信息+司机信息+租赁时间+租金

1. **“租赁信息”数据结构之数据项：**

名称：客户信息

描述：记录客户的基本信息

组成：客户编号、身份证号、姓名、性别、年龄、电话、地址

名称：车辆信息

描述：记录出租车辆的基本信息

组成：车辆编号、车型、车名、购买时间、租金标准、租用状态

名称：司机信息

描述：记录司机的基本信息

组成：司机编号、身份证号、姓名、性别、年龄、电话、地址、工作状态

1. **数据流**

数据流是数据结构在系统内传输的路径。前面已画出的数据流图能较好地反映出数据的前后流动关系，除此外还能描述为（以“租赁信息数据流”来说明）：

数据流名：租赁信息数据流

说明：“租赁信息”数据结构在系统内的流向

数据流来源：车辆租赁公司新产生的订单

数据流去向：数据库

平均流量：每天几百次

高峰期流量：每天上千次

1. **数据存储**

数据存储是数据结构停留或保存的地方，也是数据流的来源和去向之一。它可以是手工文档或手工凭单，也可以是计算机文档。对数据存储的描述通常包括（以租赁信息表数据存储来说明）：

数据存储名：租赁信息表

说明：租赁数据，作为原始数据需要保存与备查

编号：订单号为唯一标识，顺序整数，自动增长

输入的数据流：租赁信息数据流，来自新产生的订单

输出的数据流：订单查看数据流，用于信息管理人员查看

# 3 系统功能分析

B/S模式实现车辆租赁信息管理，具体功能如下

**车辆租赁信息管理需要完成的功能：**

* 信息管理：
* 客户管理：车辆租赁公司管理员可以查看客户的信息，当有新的客户时可以添加录入客户的信息，也可以修改客户的信息，但涉及到对历史订单的追溯不能将客户信息进行删除。
* 车辆信息：车辆租赁公司管理员可以查看车辆信息，并能修改车辆信息（调整租金标准，和更改车辆状况），添加车辆信息。同时也涉及到对历史订单的追溯不能将车辆信息删除。
* 司机信息：车辆租赁公司管理员可以查看司机信息，方便公司充分了解工作人员信息，并能修改司机信息（主要是调整工作状态），同时也涉及到对历史订单的追溯不能将司机信息删除。
* 记录管理：
* 预约记录：车辆租赁公司管理员可以管理预约记录，操作有履行预约，修改预约的信息，以及取消预约。
* 租用记录：车辆租赁公司管理员可以操作租用记录，当租赁快到期时能对客户进行催车或续租，当客户还车时能结束租用订单进行还车操作。
* 业务管理：
* 预约车辆：车辆租赁公司管理员可以根据客户需要提前添加预约记录，达成预约业务办理。
* 出租车辆：当有客户租车时车辆租赁公司管理员可以通过该功能添加租用记录，达成出租车辆的业务办理。
* 租金统计：车辆租赁公司管理员可以查看当日租金情况，或者选定某一段时间查看某一时间段的租金统计情况，结果会以图表的形式展示。

# 4 系统设计

## 4.1 数据库概念结构设计

### 4.1.1系统概念模块设计

车辆租赁公司信息管理员能对人员信息，车辆信息进行添加、查询和修改，能对租赁订单进行修改、查询、和管理，可以办理相关租赁业务



### 4.1.2E-R图

#### 4.1.2.1客户信息



#### 4.1.2.2车辆信息



#### 4.1.2.3司机信息



#### 4.1.2.4全局E-R图



## 4.2 系统功能模块设计

### 4.2.1模块图：



### 4.2.2各子模块功能描述

#### 4.2.2.1信息管理模块



**客户管理：**登记公司内所有客户详细信息，如身份证号、姓名、性别、年龄、电话、地址，信息管理员可以对信息进行添加、查询和修改。

**车辆管理：**登记公司所有车辆信息，如车型、车名、购买时间、租金标准、租用状态，信息管理员能对信息进行添加、查询和修改。

**司机管理：**登记公司内所有司机详细信息，如身份证号、姓名、性别、年龄、电话、地址、工作状态，信息管理员可以对信息进行添加、查询和修改。

#### 4.2.2.2记录管理模块



**预约记录：**记录所有预约信息，包括客户编号、车辆编号、租赁模式、预约租赁时间、预约时间等，信息管理员能进行查看详情、履行预约、修改预约、取消预约、查询等操作。

**租用记录：**记录所有租用信息，包括客户编号、车辆编号、司机编号、租赁时间等，信息管理员能进行查看详情、还车、续租、催车、查询等操作。

**还车记录：**记录所有还车信息，包括客户编号、车辆编号、司机编号、租赁时间、还车信息、租金结算等，信息管理人员能进行查看详情、查询等操作。

**驾驶记录：**记录所有司机的驾驶信息，包括司机编号、车辆编号、开始时间，结束时间，信息管理人员能进行查看详情等操作。

#### 4.2.2.3业务管理模块

#### 

**预约车辆：**信息管理员能进行车辆预约的操作，登记相关预约信息，包括，客户编号、车辆编号、租赁模式、预约时间等。

**租用车辆：**信息管理员能进行车辆租赁的操作，登记相关租赁信息，包括，客户编号、车辆编号、司机编号、租赁模式、租赁时间等。

#### 4.2.2.4租金统计模块



**当日租金：**统计当日租金。

**某一时间段租金：**统计某一时间段租金，结果以图表形式展示。

## 4.3 数据库逻辑结构设计

按照E-R图到逻辑关系模式的转换规则，可得到系统如下7个关系。

1. 客户（客户编号，身份证号，姓名，性别，年龄，电话，地址）
2. 司机（司机编号，身份证号，姓名，性别，年龄，电话，地址，状态）
3. 车辆（车辆编号，车型，车名，购买时间，租金标准，租用状态）
4. 预约（订单编号，客户编号，车辆编号，租赁模式，预约租赁开始时间，预约租赁结束时间，预约时间）
5. 租用（订单编号，客户编号，车辆编号，司机编号，租赁模式，租赁开始时间，租赁结束时间）
6. 还车（订单编号，客户编号，车辆编号，司机编号，租赁模式，租赁开始时间，租赁结束时间，租赁时长，还车时间，租金）
7. 驾驶（司机编号，车辆编号，开始时间，结束时间）

其中带下划线的为关系关键字（即主码）

## 4.4 数据库物理结构设计及其实现

### 4.4.1创建数据库及表的SQL命令

-- 创建数据库

drop database if exists vehicle\_rental\_management\_system;

create database vehicle\_rental\_management\_system;

-- 创建管理员表

drop table if exists Admin;

create table Admin (

username varchar(20) primary key not null,

passwd varchar(20) not null

);

-- 创建车辆信息表

drop table if exists carMessage;

create table carMessage (

carId bigint primary key not null auto\_increment,

carType varchar(10) not null,

carName varchar(10) not null,

carBuyTime date not null,

carRentStandard decimal(10,2) not null,

carState varchar(10) not null

);

-- 创建客户信息表

drop table if exists clientMessage;

create table clientMessage (

clientId bigint primary key not null auto\_increment,

clientIdCard varchar(20) not null,

clientName varchar(10) not null,

clientSex varchar(5) not null,

clientAge int,

clientPhone varchar(20) not null,

clientAddress varchar(50) not null

);

-- 创建司机信息表

drop table if exists driverMessage;

create table driverMessage (

driverId bigint primary key not null auto\_increment,

driverIdCard varchar(20) not null,

driverName varchar(10) not null,

driverSex varchar(5) not null,

driverAge int not null,

driverPhone varchar(20) not null,

driverAddress varchar(50) not null,

driverState varchar(10) not null

);

-- 创建租用记录表

drop table if exists rentRecord;

create table rentRecord (

rentId bigint primary key not null auto\_increment,

clientId bigint not null,

carId bigint not null,

driverId bigint,

rentType varchar(5) not null,

rentFromTime date not null,

rentToTime date not null,

foreign key (clientId) references clientMessage(clientId),

foreign key (carId) references carMessage(carId),

foreign key (driverId) references driverMessage(driverId)

);

-- 创建还车记录表

drop table if exists returnRecord;

create table returnRecord (

rentId bigint primary key not null auto\_increment,

clientId bigint not null,

carId bigint not null,

driverId bigint,

rentType varchar(5) not null,

rentFromTime date not null,

rentToTime date not null,

rentTime int not null,

returnTime date not null,

rentMoney decimal(10,2) not null,

foreign key (clientId) references clientMessage(clientId),

foreign key (carId) references carMessage(carId),

foreign key (driverId) references driverMessage(driverId)

);

-- 创建驾驶记录表

drop table if exists drive;

create table drive (

driverId bigint,

carId bigint not null,

driveFromTime date not null,

driveToTime date not null,

foreign key (driverId) references driverMessage(driverId),

foreign key (carId) references carMessage(carId)

);

-- 创建预约记录表

drop table if exists bookRecord;

create table bookRecord (

rentId bigint primary key not null auto\_increment,

clientId bigint not null,

carId bigint not null,

rentType varchar(5) not null,

rentFromTime date not null,

rentToTime date not null,

bookTime date not null,

foreign key (clientId) references clientMessage(clientId),

foreign key (carId) references carMessage(carId)

);

### 4.4.2 **表数据的初始化（INSERT命令）**

-- 管理员信息

insert into Admin values ('admin', 'admin123');

insert into Admin values ('test', 'test');

-- 车辆信息

insert into carMessage values ('11001', '中型车', '奥迪', '2018-10-15', 100, '可出租');

insert into carMessage values ('11002', '小型车', '大众', '2019-10-15', 80, '可出租');

insert into carMessage values ('11003', '大型车', '五菱宏光', '2019-08-12', 120, '可出租');

insert into carMessage values ('11004', '紧凑型车', '丰田', '2019-09-12', 110, '可出租');

insert into carMessage values ('11005', '大型车', '马自达', '2019-11-12', 120, '可出租');

insert into carMessage values ('11006', '小型车', '雪佛兰', '2020-05-13', 110, '已租用');

insert into carMessage values ('11007', '商务车', '现代', '2020-05-11', 100, '已租用');

insert into carMessage values ('11008', '商务车', '别克', '2020-05-09', 115, '已租用');

insert into carMessage values ('11009', '小型车', '奔驰', '2020-05-01', 100, '已租用');

insert into carMessage values ('11010', '中型车', '路虎', '2020-05-13', 90, '已租用');

-- 客户信息

insert into clientMessage values ('10001', '320100\*\*\*\*\*\*\*\*\*\*\*\*', '赵峰', '男', 26, '133\*\*\*\*0510', '江苏南京');

insert into clientMessage values ('10002', '320200\*\*\*\*\*\*\*\*\*\*\*\*', '钱林', '男', 27, '135\*\*\*\*0517', '江苏无锡');

insert into clientMessage values ('10003', '320300\*\*\*\*\*\*\*\*\*\*\*\*', '孙淼', '男', 28, '172\*\*\*\*7745', '江苏徐州');

insert into clientMessage values ('10004', '320400\*\*\*\*\*\*\*\*\*\*\*\*', '李鑫', '男', 29, '199\*\*\*\*8976', '江苏常州');

insert into clientMessage values ('10005', '320500\*\*\*\*\*\*\*\*\*\*\*\*', '周欣', '女', 27, '178\*\*\*\*0796', '江苏苏州');

-- 司机信息

insert into driverMessage values ('1111', '320100\*\*\*\*\*\*\*\*\*\*\*\*', '王明', '男', 26, '133\*\*\*\*0510', '江苏南京','未接单');

insert into driverMessage values ('1112', '320200\*\*\*\*\*\*\*\*\*\*\*\*', '黄浩', '男', 27, '135\*\*\*\*0517', '江苏无锡','未接单');

insert into driverMessage values ('1113', '320300\*\*\*\*\*\*\*\*\*\*\*\*', '李雷', '男', 28, '172\*\*\*\*7745', '江苏徐州','未接单');

insert into driverMessage values ('1114', '320400\*\*\*\*\*\*\*\*\*\*\*\*', '杨星', '男', 29, '199\*\*\*\*8976', '江苏常州','未接单');

insert into driverMessage values ('1115', '320500\*\*\*\*\*\*\*\*\*\*\*\*', '张泽', '女', 27, '178\*\*\*\*0796', '江苏苏州','未接单');

-- 租用记录

insert into rentRecord values('11006','10001','11006','1111','日租','2020-6-16','2020-6-20');

insert into rentRecord values('11007','10002','11007','1112','日租','2020-07-01','2020-07-05');

insert into rentRecord values('11008','10003','11008','1113','年租','2019-08-16','2020-08-16');

insert into rentRecord values('11009','10004','11009','1114','月租','2019-12-10','2020-06-10');

insert into rentRecord values('11010','10005','11010','1115','月租','2020-06-16','2020-07-16');

-- 归还记录

insert into returnRecord values('11001','10001','11001','1111','日租','2018-10-16','2018-10-20',4,'2018-10-20',400.00);

insert into returnRecord values('11002','10002','11002','1112','日租','2019-10-16','2019-10-20',4,'2019-10-20',320.00);

insert into returnRecord values('11003','10003','11003','1113','年租','2019-08-16','2020-08-16',366,'2019-08-16',43920.00);

insert into returnRecord values('11004','10004','11004','1114','月租','2019-09-10','2019-11-10',61,'2019-11-10',6710.00);

insert into returnRecord values('11005','10005','11005','1115','月租','2019-11-16','2019-12-16',30,'2019-12-16',3600.00);

-- 驾驶记录

insert into drive values('1111','11001','2018-10-16','2018-10-20');

insert into drive values('1112','11002','2019-10-16','2019-10-20');

insert into drive values('1113','11003','2019-08-16','2020-08-16');

insert into drive values('1114','11004','2019-09-10','2019-11-10');

insert into drive values('1115','11005','2019-11-16','2019-12-16');

-- 预约记录

insert into bookRecord values('11011','10001','11001','日租','2020-8-16','2020-8-20','2020-07-05');

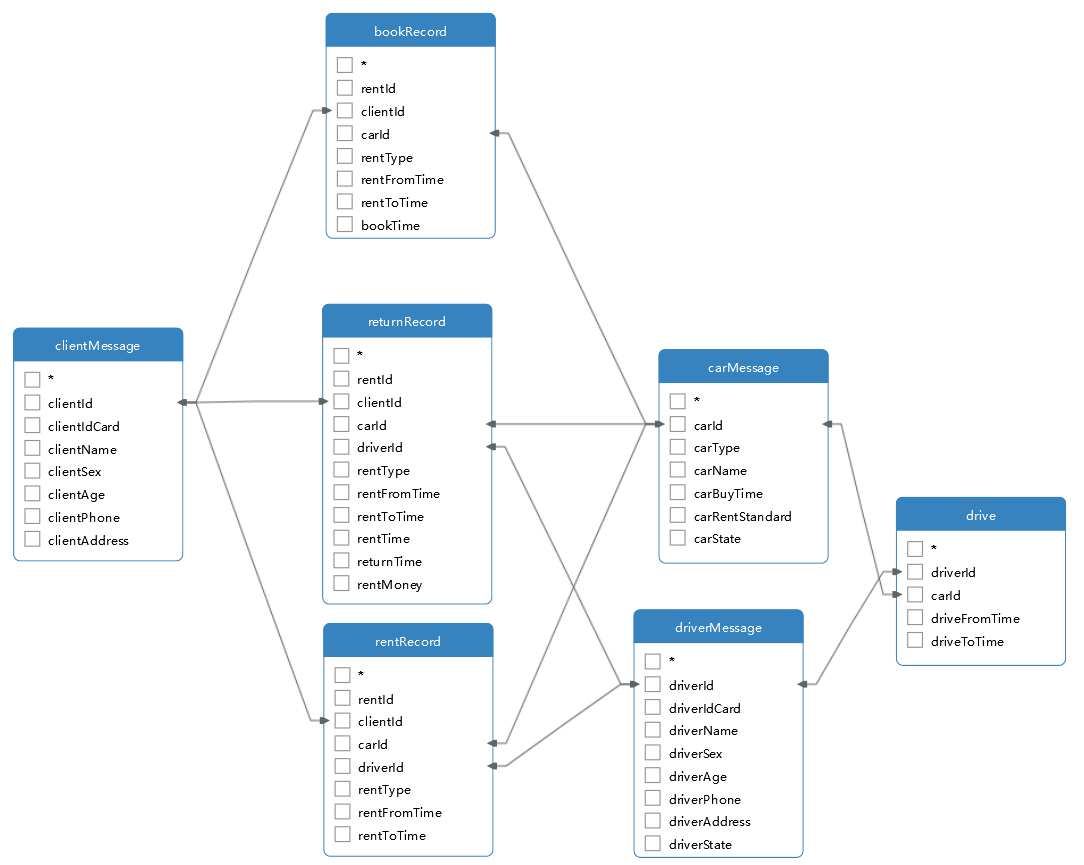
insert into bookRecord values('11012','10002','11002','日租','2020-08-01','2020-08-05','2020-05-01');

insert into bookRecord values('11013','10003','11003','年租','2020-08-16','2021-08-16','2020-06-29');

insert into bookRecord values('11014','10004','11004','月租','2020-8-10','2020-010-10','2020-07-01');

insert into bookRecord values('11015','10005','11005','月租','2020-09-16','2022-09-16','2020-06-24');

### 4.4.3 数据库表关系图



# 5 系统实现

## 5.1说明

该系统后端使用NodeJS编写，前端使用HTML+CSS+JavaScript,通过Ajax实现后台与服务器的数据交换

## 5.2数据库操作通用api

/\*

    封装操作数据库的通用api

\*/

const mysql = require('mysql'); //引入mysql模块

exports.base = (sql, data, callback) => {

    // 创建数据库连接

    const connection = mysql.createConnection({

        host: '121.199.32.198', // 数据库所在的服务器的域名或者IP地址

        user: 'root', // 登录数据库的账号

        password: 'WSad13579/', // 登录数据库的密码

        database: 'vehicle\_rental\_management\_system', // 数据库名称

        dateStrings : true, //解决时间格式

        multipleStatements:true //可一次性执行多条语句

    });

    // 执行连接操作

connection.connect();

    // 操作数据库(数据库操作也是异步的)

    connection.query(sql, data, function (error, results, fields) {

        try {

            if (error) throw error;

        } catch (err) {

           results = {affecterRows:-1,error:err}

        }

        callback(results);

    });

    // 关闭数据库

    connection.end();

}

## 5.3验证登入与判断是否登入

通过查找Admin表来验收用户名和密码是否正确

通过验证session来验证用户是否已经登入，否则重定向跳转到登入页面

router.use(session({

    secret: 'secret', // 对session id 相关的cookie 进行签名

    resave: true,

    saveUninitialized: false, // 是否保存未初始化的会话

    cookie: {

        maxAge: 1000 \* 60 \* 60, // 设置 session 的有效时间，单位毫秒

    }

}))

// router.use(bodyparser.json()) // 使用bodyparder中间件，

router.use(bodyparser.urlencoded({ extended: true }))

// 登入验证

exports.checkLogin = (req, res) => {

    const sql = 'select \* from Admin where username = ? and passwd = ?'

    const info = req.body

    const data = [info.username, info.password]

    db.base(sql, data, (result) => {

        if (result.length) { // 登录成功

            req.session.userName = info.username

        }

        res.json(result[0])

    })

}

// 验证session

exports.checkSession = (req, res, next) => {

    url = req.url

    if (url == '/login.html' || req.url.startsWith('/css/') || req.url.startsWith('/images/') || req.url.startsWith('/js/'))// 如果是登入页面就跳过

        next()

    else if (req.session.userName) {  //判断session 状态，如果有效，则返回主页，否则转到登录页面

        next()

    }

    else {

        res.redirect('login.html')

    }

}

## 5.3 前端登入页面



### 5.3.1 HTML代码

<!DOCTYPE html>

<html lang="zh-CN">

<head>

    <meta charset="UTF-8">

    <title>用户登入界面</title>

    <link rel="icon" type="image/png" href="images/icon.png">

    <link rel="stylesheet" type="text/css" href="css/login.css">

    <script type="text/javascript" src="js/login.js"></script>

    <script type="text/javascript" src="js/jquery.js"></script>

    <script type="text/javascript" src="js/dialog.js"></script>

</head>

<body>

    <div class="main">

        <div class="sign-in">

            <h1>登入</h1>

            <form class="login">

                <input type="text" name="username" placeholder="用户名：">

                <br><br>

                <input type="password" name="password" placeholder="密码："><br>

                <a href="javascript:alert('联系管理员重置密码')">忘记密码?</a><br>

                <button type="button" onclick="login()">登入</button>

                <button type="button" onclick="register()">注册</button>

            </form>

        </div>

        <div class="about">

            <h1>说明</h1>

            <h2>一、系统简介</h2>

            <p>

                此系统为车辆租赁管理子系统<br>

                供车辆租赁公司信息管理员使用

            </p>

            <h2>二、系统使用</h2>

            <p>

                登入后按照需求和界面提示进行操作

            </p>

            <h2>三、系统维护</h2>

            <p>

                请联系公司系统管理员及开发人员

            </p>

            <h2>四、联系作者</h2>

            <p>

                请去GitHub提issue <a href="https://github.com/zhouyumin/vehicle-rental-management-system"

                    target="\_blank">[跳转链接]</a>

            </p>

        </div>

    </div>

</body>

</html>

### 5.3.2 JavaScript代码

function login() {

    var form = $('form.login')

    $.ajax({

        type: 'post',

        url: '/login',

        data: form.serialize(),

        dataType: 'json',

        success: function (data) {

        window.open('/','\_self')

        },

        error: function(){

            alert('登入失败，请重新登入')

            form.find('input[name=username]').val('')

            form.find('input[name=password]').val('')

        }

    })

}

function register(){

    alert('请联系数据库管理员注册账号')

}

### 5.3.3 CSS代码

html {

    height: 100%;

}

body {

    background-image: linear-gradient(-20deg, #b721ff 0%, #21d4fd 100%);

}

.main {

    user-select: none;

    background-color: rgba(255,255,255,0.6);

    border-radius: 10px;

    box-shadow: 0 14px 28px rgba(0,0,0,0.25), 0 10px 10px rgba(0,0,0,0.22);

    position: absolute;

    overflow: hidden;

    left: 50%;

    top: 50%;

    width: 700px;

    height: 480px;

    margin-left: -350px;

    margin-top: -240px;

}

.sign-in{

    text-align: center;

    position: absolute;

    left: 0px;

    top: 15%;

    width: 50%;

}

.sign-in a {

    display: inline-block;

    color: #333;

    margin: 10px 0px 10px 0px;

}

.about {

    text-align: center;

    position: absolute;

    right: 0px;

    width: 50%;

    height: 100%;

    background-color: rgba(244,244,244,0.2);

}

.login {

    position: relative;

    top: 20%;

}

.login input {

    background-color: #eee;

    outline: none;

    border: none;

    border-radius: 20px;

    padding: 12px 15px;

    width: 80%;

}

button {

    width: 90px;

    height: 40px;

    margin-left: 10px;

    margin-right: 20px;

    border-radius: 20px;

    border: none;

    background: linear-gradient(90deg, #9090ff, #7070ff);

    outline: none;

    cursor: pointer;

    transition-duration: 100ms;

}

button:hover {

    background: linear-gradient(90deg, #7070ff, #9090ff);

    color: #b721ff;

}

## 5.4系统主界面功能菜单



### 5.4.1 HTML代码

<!DOCTYPE html>

<html lang="zh-CN">

<head>

    <meta charset="UTF-8">

    <title>车辆租赁管理子系统</title>

    <link rel="icon" type="image/png" href="images/icon.png">

    <link rel="stylesheet" type="text/css" href="css/index.css">

    <script type="text/javascript" src="./js/jquery.js"></script>

    <script type="text/javascript" src="./js/today.js"></script>

    <script type="text/javascript" src="./js/dialog.js"></script>

    <script type="text/javascript" src="./js/manage/book.js"></script>

    <script type="text/javascript" src="./js/manage/rent.js"></script>

    <script type="text/javascript" src="./js/manage/charge.js"></script>

    <script type="text/javascript" src="./js/Chart.js"></script>

</head>

<body>

    <div class="head">

        <h1>车辆租赁管理子系统</h1>

    </div>

    <div class="main">

        <div class="menu menu-left">

            <h2>信息管理</h2>

            <div onclick="window.open('./info/client.html')">

                客户信息

            </div>

            <div onclick="window.open('./info/car.html')">

                车辆信息

            </div>

            <div onclick="window.open('./info/driver.html')">

                司机信息

            </div>

        </div>

        <div class="menu menu-middle">

            <h2>记录管理</h2>

            <div onclick="window.open('./record/book.html')">

                预约记录

            </div>

            <div onclick="window.open('./record/rent.html')">

                租用记录

            </div>

            <div onclick="window.open('./record/return.html')">

                还车记录

            </div>

        </div>

        <div class="menu menu-right">

            <h2>业务管理</h2>

            <div id="book">

                预约车辆

            </div>

            <div id="rent">

                出租车辆

            </div>

            <div id="charge">

                租金统计

            </div>

        </div>

    </div>

    <form class="form" id="bookForm">

        <label>客户编号：</label><input type="text" name="clientId"><br>

        <label>车辆编号：</label><input type="text" name="carId"><br>

        <label>租赁模式：</label>

        <select name="rentType" style="margin-top: 6px; width: 166px; height: 24px; margin-left: -5px;">

            <option>日租</option>

            <option>月租</option>

            <option>年租</option>

        </select><br>

        <label>预约租赁开始时间：</label><input type="date" name="rentFromTime" style="width: 160px;"><br>

        <label>预约租赁结束时间：</label><input type="date" name="rentToTime" style="width: 160px;"><br>

        <input type="text" hidden name="bookTime">

        <input type="button" value="提交" style="margin-left: 145.67px; margin-top: 20px;">

    </form>

    <form class="form" id="rentForm">

        <label>客户编号：</label><input type="text" name="clientId"><br>

        <label>车辆编号：</label><input type="text" name="carId"><br>

        <label>司机编号：</label><input type="text" name="driverId" placeholder="不安排司机可不填"><br>

        <label>租赁模式：</label>

        <select name="rentType" style="margin-top: 6px; width: 166px; height: 24px; margin-left: -5px;">

            <option>日租</option>

            <option>月租</option>

            <option>年租</option>

        </select><br>

        <label>租赁开始时间：</label><input type="date" name="rentFromTime" style="width: 160px;"><br>

        <label>租赁结束时间：</label><input type="date" name="rentToTime" style="width: 160px;"><br>

        <input type="button" value="提交" style="margin-left: 129.67px; margin-top: 20px;">

    </form>

    <div class="container">

        <br>

        <div class="todayMoney">

            <label>&nbsp;&nbsp;今日租金：</label>

            <label id="rentMoney"></label>

        </div>

        <br>

        <form id="timeInput">

            <label>&nbsp;&nbsp;从</label> <input type="date" name="fromTime">

            <label>到</label> <input type="date" name="toTime">

            <label>总租金：</label><label id="sumMoney"></label>

        </form>

        <br>

        <div class="graphic">

            <canvas id='chart'></canvas>

        </div>

    </div>

    <footer>

        <li>

            作者：ZYM

        </li>

        <li>

            <a href="https://github.com/zhouyumin/vehicle\_rental\_management\_system">网站源代码</a>

        </li>

    </footer>

</body>

</html>

### 5.4.2 CSS代码

html {

    height: 100%;

}

body {

    background-image: url("../images/background.jpg");

    background-repeat: no-repeat;

    background-attachment: fixed;

    background-position: 50% 50%;

    background-size: 100% 100%;

    margin: 0px;

}

.head {

    text-align: center;

    background-color: #006179;

    height: 80px;

    top: 0px;

    left: 0px;

    user-select: none;

}

.head h1{

    color: #f4f4f4;

    margin: 0px;

    padding-top: 19.2px;

}

.main {

    position: absolute;

    width: 70%;

    height: 450px;

    top: 50%;

    margin-top: -225px;

    left:15%;

    border-radius: 30px;

    background-color: rgba(0, 0, 0, 0.66);

    backdrop-filter: blur(6px);

}

.menu{

    position: relative;

    text-align: center;

    width: 20%;

    top: 35px;

    height: 440px;

    user-select: none;

}

.menu h2{

    color: #00f4f4cc;

    margin: 0px;

    width: 100%;

    padding-top: 20px;

}

.menu-left {

    left: 10%;

}

.menu-middle {

    margin-top: -440px;

    left: 40%;

}

.menu-right {

    margin-top: -440px;

    left: 70%;

}

.menu-left div, .menu-middle div, .menu-right div {

    position: relative;

    background-color: #2ed573;

    color: #fff;

    margin-top: 20px;

    height: 80px;

    line-height: 80px;

    width: 100px;

    left: 50%;

    margin-left: -50px;

    border-radius: 50%;

    transition-duration: 100ms;

}

.menu-left div:hover, .menu-middle div:hover, .menu-right div:hover {

    cursor: pointer;

    background-color: #2ed5;

    color: #2ed573;

}

.form {

    display: none;

    position: absolute;

    left: 50%;

    top: 50%;

}

.form label, .form input {

    margin-top: 6px;

}

.form label {

    display: inline-block;

    text-align: center;

}

#bookForm {

    margin-left: -157.87px;

    margin-top: -79.5px;

}

#bookForm label {

    width: 144px;

}

#rentForm {

    margin-left: -138.87px;

    margin-top: -94.4px;

}

#rentForm label {

    width: 112px;

}

.container{

    display: none;

}

footer {

    position: fixed;

    color: lightgray;

    font-size: 24px;

    left: 50%;

    margin-left: -79.215px;

    bottom: 0px;

}

footer a{

    color: lightgray;

}

## 5.5 查询操作

以查询客户信息为例

输入客户姓名点击查询就能实现查找



### 5.5.1 JavaScript代码

function initList(url) {

    $.ajax({

        type: 'get',

        url: url,

        dataType: 'json',

        success: function (data) {

            if (data == '') {

                alert('客户信息为空')

            }

            // 渲染数据列表

            var html = template('template', { list: data })

            //渲染模板

            $('#dataList').html(html)

            //绑定操作事件

            $('#dataList').find('tr').each(function (index, element) {

                const td = $(element).find('td:eq(7)')

                //绑定修改客户信息操作

                td.find('a:eq(0)').click(function () {

                    editClient(element)

                })

                // //绑定删除客户信息操作

                // td.find('a:eq(1)').click(function () {

                //     deleteClient(clientId)

                // })

            })

            //绑定添加客户信息操作

            addClient()

        }

    })

}

//绑定查询操作

$('#search').click(search)

var clientNameInput = $('div.operation').find('input[name=clientName]')

clientNameInput.keyup(function (e) {

    if (e.keyCode == 13) {

        search()

    }

})

function search() {

    var clientName = clientNameInput.val()

    if (clientName == '') {

        alert('姓名未输入')

    }

    else {

        initList('/clients/clientName/' + clientName)

    }

}

### 5.5.2后端实现查询代码

//查询客户信息

exports.getClientByName=(req,res)=>{

    let data = req.params.clientName

    let sql = 'select \* from clientMessage where clientName = ?'

    db.base(sql,data,(result)=>{

        res.json(result)

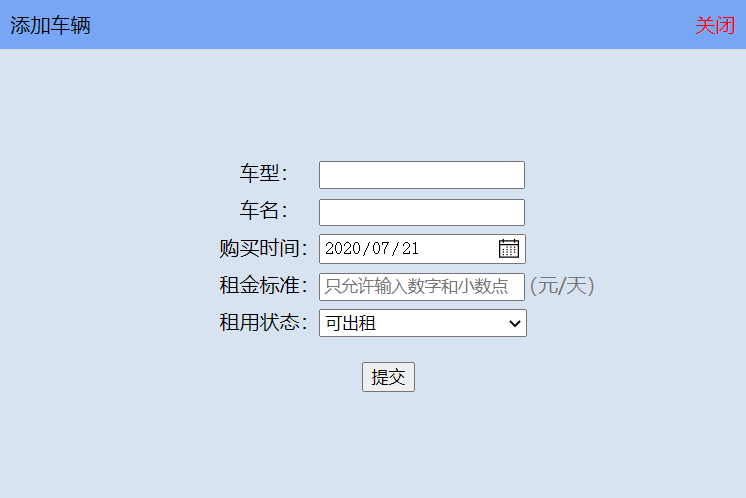
    })

}

## 5.6 添加操作

以添加车辆信息为例

输入相关信息点击提交就能实现添加车辆



### 5.6.1 JavaScript代码

//添加车辆信息

function addcar() {

    $('#addCar').click(function () {

        //重置表单

        clearForm()

        var form = $('#addCarForm')

        form.find('input[name=carBuyTime]').val(today())

        //初始化弹窗

        var mark = new MarkBox(600, 400, '添加车辆', form.get(0))

        mark.init()

        //绑定添加事件

        form.find('input[type=button]').unbind('click').click(function () {

            $.ajax({

                type: 'post',

                url: '/cars/car',

                data: form.serialize(),

                dataType: 'json',

                success: function (data) {

                    if (data.flag == 1) {

                        mark.close()

                        initList('/cars')

                        alert('添加成功')

                    }

                    else {

                        mark.close()

                        alert('[error]:' + data.sqlMessage)

                    }

                }

            })

        })

    })

}

//重置表单

function clearForm() {

    var form = $('#addCarForm')

    form.get(0).reset()

}

### 5.6.2后端实现添加操作

//添加车辆信息

exports.addCar = (req,res)=>{

    let info = req.body

    let sql = 'insert into carMessage set ?'

    db.base(sql , info,(result)=>{

        if(result.affectedRows == 1){

            res.json({flag:1})

        }

        else{

            res.json(result.error)

        }

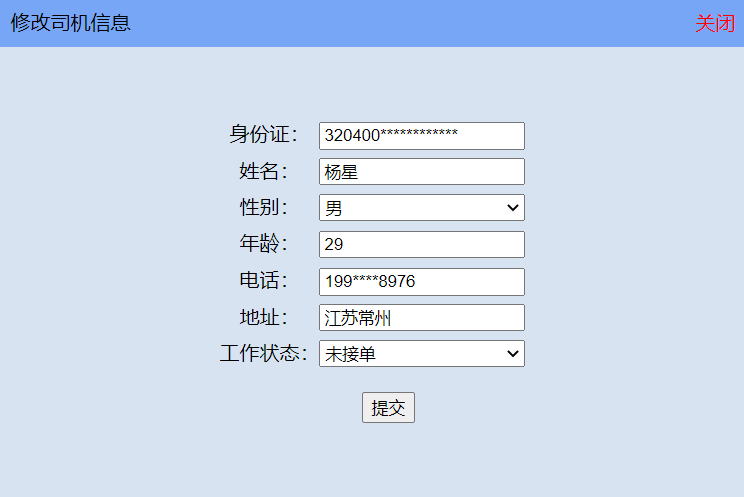
    })

}

## 5.7 修改操作

以修改司机信息为例

修改完信息后点击提交就能实现修改



### 5.7.1 JavaScript代码

//编辑司机信息

function editDriver(element) {

    //重置表单

    clearForm()

    var form = $('#addDriverForm')

    //初始化弹窗

    var mark = new MarkBox(600, 400, '修改司机信息', form.get(0))

    mark.init()

    // $('#dialogTitle').text('修改司机信息')

    //填充表单

    var data = $(element).find('td')

    const driverId = $(data.get(0)).text()

    form.find('input[name=driverIdCard]').val($(data.get(1)).text())

    form.find('input[name=driverName]').val($(data.get(2)).text())

    form.find('select[name=driverSex]').val($(data.get(3)).text())

    form.find('input[name=driverAge]').val($(data.get(4)).text())

    form.find('input[name=driverPhone]').val($(data.get(5)).text())

    form.find('input[name=driverAddress]').val($(data.get(6)).text())

    form.find('select[name=driverState]').val($(data.get(7)).text())

    //绑定提交表单数据

    form.find('input[type=button]').unbind('click').click(function () {

        $.ajax({

            type: 'put',

            url: '/drivers/driver',

            data: form.serialize()+'&driverId='+driverId,

            dataType: 'json',

            success: function (data) {

                if (data.flag == 1) {

                    mark.close()

                    initList('/drivers')

                    alert('修改成功')

                }

                else {

                    mark.close()

                    alert('[error]:' + data.sqlMessage)

                }

            }

        })

    })

}

//重置表单

function clearForm() {

    var form = $('#addDriverForm')

    form.get(0).reset()

}

### 5.7.2后端实现修改操作

//修改司机信息

exports.editDriver = (req,res)=>{

    let info = req.body

    let sql = 'update driverMessage set ? where driverId=?'

    db.base(sql , [info,info.driverId],(result)=>{

        if(result.affectedRows == 1){

            res.json({flag:1})

        }

        else{

            res.json(result.error)

        }

    })

}

## 5.8 删除操作

以删除预约记录为例

点击取消就会询问是否删除

点击确定后就会从数据库中删除该记录



### 5.8.1 JavaScript代码

//取消操作

function cancel(rentId, type) {

    console.log(type)

    if (type == 'cancel') {

        var res = confirm("是否取消该订单？")

        if (res == false) {

            return

        }

    }

    $.ajax({

        type: 'delete',

        url: '/books/book/' + rentId,

        dataType: 'json',

        success: function (result) {

            if (result.flag == 1) {

                initList('/books')

                if (type == 'cancel')

                    alert('取消成功')

                else if (type == 'delete') {

                    alert('履约成功')

                }

            }

            else {

                alert('[error]:' + result.sqlMessage)

            }

        },

        error: function () {

            alert('error')

        }

    })

}

### 5.8.2后端实现删除操作

//删除预约记录

exports.deleteBook=(req,res)=>{

    let rentId = req.params.rentId

    let sql = 'delete from bookRecord where rentId = ?'

    db.base(sql ,rentId,(result)=>{

        if(result.affectedRows == 1){

            res.json({flag:1})

        }

        else{

            res.json(result.error)

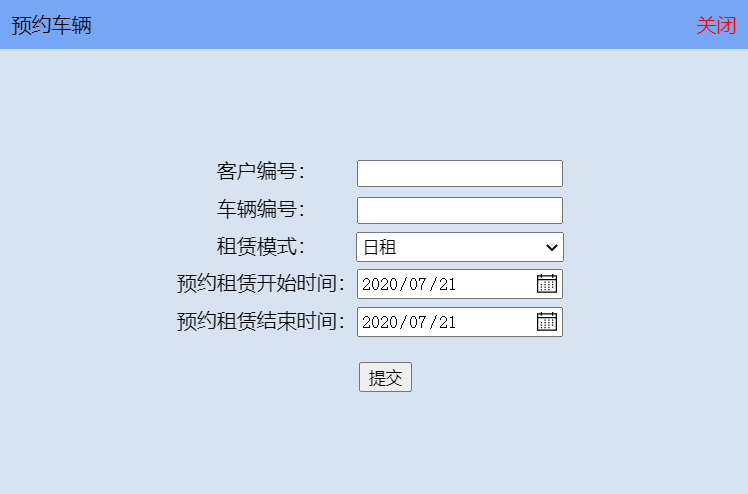
        }

    })

}

## 5.9预约车辆

填写好表单信息提交后就能预约成功



### 5.9.1 JavaScript代码

$(function () {

    const form = $('#bookForm')

    //绑定预约

    $('#book').click(function(){

        var mark = new MarkBox(600,400,'预约车辆',form.get(0))

        form.get(0).reset()

        form.find('input[name=rentFromTime]').val(today())

        form.find('input[name=rentToTime]').val(today())

        form.find('input[name=bookTime]').val(today())

        mark.init()

})

    //绑定提交

    form.find('input[type=button]').click(function () {

        const clientId = form.find('input[name=clientId]').val()

        const carId = form.find('input[name=carId]').val()

        if (clientId == '' || carId == '') {

            alert('客户编号或车辆编号为空')

            return

        }

        $.ajax({

            type: 'post',

            url: '/books/book',

            data: form.serialize(),

            dataType: 'json',

            success: function (data) {

                if (data.flag == 1) {

                    alert('预约成功')

                }

                else {

                    alert('[error]:' + data.sqlMessage)

                }

            },

            error: function(){

                alert('发生错误')

            }

        })

    })

})

### 5.9.2后端实现预约操作

//添加预约记录

exports.addBook=(req,res)=>{

    let info = req.body

    let sql = 'insert into bookRecord set ?'

    db.base(sql , info,(result)=>{

        if(result.affectedRows == 1){

            res.json({flag:1})

        }

        else{

            res.json(result.error)

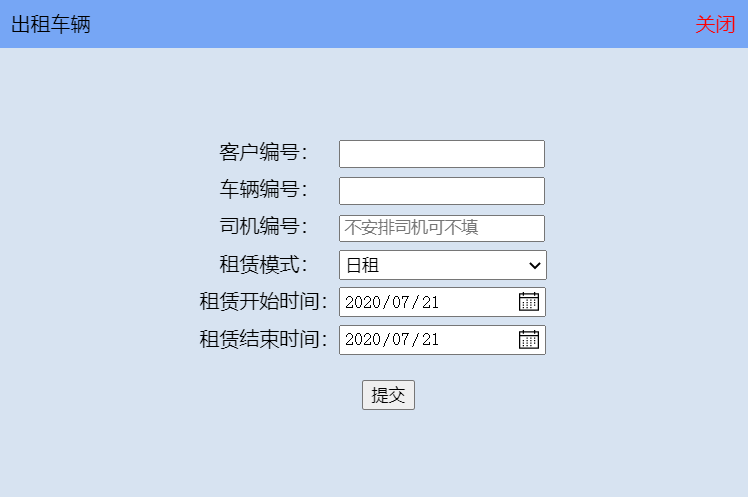
        }

    })

}

## 5.10租用车辆

填写好信息提交就能租用车辆



### 5.10.1 JavaScript代码

$(function () {

    const form = $('#rentForm')

    //绑定出租车辆

    $('#rent').click(function(){

        var mark = new MarkBox(600,400,'出租车辆',form.get(0))

        form.get(0).reset()

        form.find('input[name=rentFromTime]').val(today())

        form.find('input[name=rentToTime]').val(today())

        mark.init()

    })

    //绑定提交

    form.find('input[type=button]').click(function () {

        const clientId = form.find('input[name=clientId]').val()

        const carId = form.find('input[name=carId]').val()

        if (clientId == '' || carId == '') {

            alert('客户编号或车辆编号为空')

            return

        }

        $.ajax({

            type: 'post',

            url: '/records/record',

            data: form.serialize(),

            dataType: 'json',

            success: function (data) {

                if (data.flag == 1) {

                    alert('租车成功')

                }

                else {

                    alert('[error]:' + data.sqlMessage)

                }

            },

            error: function(){

                alert('发生错误')

            }

        })

    })

})

### 5.10.2后端实现租用操作

//添加订单

exports.addRecord=(req,res)=>{

    let info = req.body

    if (info.driverId==''){

        info.driverId=null

    }

    let sql = `insert into rentRecord set ?;update carMessage set carState='已租用' where carId=?;\

    update driverMessage set driverState='已接单' where driverId=?;`

    db.base(sql , [info,info.carId,info.driverId],(result)=>{

        if(result[0].affectedRows == 1){

            res.json({flag:1})

        }

        else{

            res.json(result[0].error)

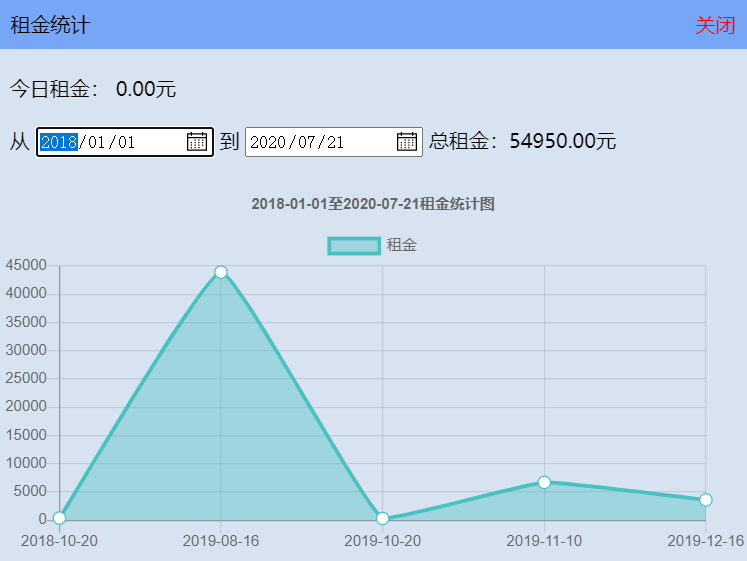
        }

    })

}

## 5.11 租金统计

可查看当日租金和一段时间内的租金



### 5.11.1 JavaScript代码

$(function () {

    //获得表单

    var form = $('#timeInput')

    var fromTime = form.find('input[name=fromTime]')

    var toTime = form.find('input[name=toTime]')

    //绑定租金统计

    $('#charge').click(function () {

        var mark = new MarkBox(600, 450, '租金统计', $('div.container').get(0))

        mark.init()

        fromTime.val(today())

        toTime.val(today())

        //查询今日租金

        $.ajax({

            type: 'get',

            url: '/returns/money/' + fromTime.val() + '/' + toTime.val(),

            data: form.serialize(),

            dataType: 'json',

            success: function (result) {

                if (result.length != 0) {

                    $('#rentMoney').text(result[0].rentMoney+'元')

                }

                else {

                    $('#rentMoney').text('0.00元')

                    $('#sumMoney').text('0.00元')

                }

                updateChart(result)

            },

            error: function () {

                alert('error')

            }

        })

    })

    //时间改变，刷新租金统计

    fromTime.change(function () {

        search()

    })

    toTime.change(function () {

        search()

    })

    //查询时间段内的租金

    function search() {

        $.ajax({

            type: 'get',

            url: '/returns/money/' + fromTime.val() + '/' + toTime.val(),

            data: form.serialize(),

            dataType: 'json',

            success: function (result) {

                updateChart(result)

            },

            error: function () {

                alert('error')

            }

        })

    }

    function updateChart(results) {

        var labels = []

        var data = []

        var sum = 0

        results.forEach(element => {

            labels.push(element.returnTime)

            data.push(element.rentMoney)

            sum += parseFloat(element.rentMoney)

        });

        //获得保留两位小数

        var getFloat = (num) => {

            num += ''

            num = num.replace(/[^0-9|\.]/g, '') //清除字符串中的非数字非.字符

            if (/^0+/) //清除字符串开头的0

                num = num.replace(/^0+/, '')

            if (!/\./.test(num)) //为整数字符串在末尾添加.00

                num += '.00'

            if (/^\./.test(num)) //字符以.开头时,在开头添加0

                num = '0' + num

            num += '00'        //在字符串末尾补零

            num = num.match(/\d+\.\d{2}/)[0]

            return num

        }

        sum = getFloat(sum)

        $('#sumMoney').text(sum+'元')

        var lineChartData = {

            labels: labels,

            datasets: [

                {

                    label: '租金',

                    fill: true,

                    lineTension: 0.1,

                    backgroundColor: "rgba(75,192,192,0.4)",

                    borderColor: "rgba(75,192,192,1)",

                    borderCapStyle: 'butt',

                    borderDash: [],

                    borderDashOffset: 0.0,

                    borderJoinStyle: 'miter',

                    pointBorderColor: "rgba(75,192,192,1)",

                    pointBackgroundColor: "#fff",

                    pointBorderWidth: 1,

                    pointHoverRadius: 5,

                    pointHoverBackgroundColor: "rgba(75,192,192,1)",

                    pointHoverBorderColor: "rgba(220,220,220,1)",

                    pointHoverBorderWidth: 2,

                    pointRadius: 5,

                    pointHitRadius: 10,

                    data: data,

                    spanGaps: false,

                }

            ]

        };

        var ctx = $("#chart").get(0).getContext("2d");

        var LineChart = new Chart(ctx, {

            type: 'line',

            data: lineChartData,

            responsive: true,

            bezierCurve: false,

            options: {

                title: {

                  display: true,

                  text: fromTime.val() + '至' + toTime.val() + "租金统计图",

                  position: 'top'

                }

              }

        });

    }

})

### 5.11.2 后端实现租金查询操作

//查询租金

exports.getMoney = (req, res) => {

    let info = req.params

    const sql = 'select returnTime,sum(rentMoney) as rentMoney from returnRecord\

    where returnTime between ? and ? group by returnTime order by returnTime'

    db.base(sql, [info.rentFromTime,info.rentToTime], (result) => {

        result.forEach(element => {

            element.rentMoney = toFloat.getFloatStr(element.rentMoney)

        });

        res.json(result)

    })

}

## 5.12通用CSS

body{

    background-color: lightblue;

}

.title {

    width: 100%;

    text-align: center;

    background-color: #006179;

    height: 50px;

    top: 10px;

    margin-top: 0px;

    line-height: 50px;

}

.title h1{

    position: absolute;

    left: 50%;

    margin-left: -62.5px;

    width: 125px;

    margin-top: auto;

    font-size: 28px;

    color: #fff;

}

.operation{

    position: absolute;

    right: 20px;

    top: 21.1px;

}

.operation button{

    padding-top: 0px;

    margin-right: 20px;

    border: none;

}

.operation button:hover{

    cursor: pointer;

}

.operation input{

    line-height: 18.4px;

    border: none;

    width: 100px;

}

table {

    width: 100%;

    text-align: center;

    border-bottom: #2ed573 1px solid;

    border-right: #2ed573 1px solid;

}

td, th {

    height: 40px;

    line-height: 40px;

    border-top: #2ed573 1px solid;

    border-left: #2ed573 1px solid;

}

.form {

    display: none;

    position: absolute;

    left: 50%;

    top: 50%;

}

.form label, .form input{

    margin-top: 6px;

}

## 5.13 其他系统功能

### 5.13.1获取当天日期

//获取当天日期

function today() {

    var time = new Date()

    var day = ("0" + time.getDate()).slice(-2)

    var month = ("0" + (time.getMonth() + 1)).slice(-2)

    return time.getFullYear() + "-" + (month) + "-" + (day)

}

### 5.13.2租金数字格式转换

//转保留两位小数的数

exports.getFloatStr = (num) => {

    num += ''

    num = num.replace(/[^0-9|\.]/g, '') //清除字符串中的非数字非.字符

    if (/^0+/) //清除字符串开头的0

        num = num.replace(/^0+/, '')

    if (!/\./.test(num)) //为整数字符串在末尾添加.00

        num += '.00'

    if (/^\./.test(num)) //字符以.开头时,在开头添加0

        num = '0' + num

    num += '00'        //在字符串末尾补零

    num = num.match(/\d+\.\d{2}/)[0]

    return num

}

# 6 系统测试

经测试，该系统后端代码在服务器运行良好。前端页面在不同浏览器的界面会有些差距，其中原因是浏览器的内核不同，对各种CSS样式的解释也不同，对样式属性的默认值设置不同，另外浏览器对CSS样式的支持也不是全面的，各有各的不同，同一浏览器的不同版本对CSS的支持也是不一样的。

在后端执行的过程中暂未发现异常错误，另外使用pm2管理NodeJS进程能保证后端代码在服务器上稳定运行，不用担心进程退出。

# 7 体会与自我评价

自评得分：96

在有一点前端知识的基础上用短短三周时间内学会了使用NodeJS开发数据库管理系统，并能按时完成课设，个人有了极大收获，在面对数据库管理系统的开发上有了一定的经验和能力，在对数据库的建立和维护方面有了更深入的体会。