**单位代码： 10605**

**密 级： 公 开**



本科生毕业论文（设计）

题 目： 火车票查询系统 \_\_

学 院： 计算机与信息工程学院\_\_\_\_\_

专 业： 软件工程\_\_\_\_\_\_\_\_\_\_\_

学生姓名： 樊礼嶢\_\_\_\_\_\_\_\_\_\_\_\_

学 号： 2014107303\_\_\_\_\_\_\_\_\_\_

指导教师： 韦 静\_\_\_\_\_\_\_\_\_\_\_\_

职 称： 助 教\_\_\_\_\_\_\_\_\_\_\_\_

2017年12月25日

目录

[[摘 要] 4](#_Toc515224485)

[[关键词] 4](#_Toc515224486)

[1. 系统概述 4](#_Toc515224487)

[1.1 系统开发背景 4](#_Toc515224488)

[1.2 系统开发意义 5](#_Toc515224489)

[1.3 相关技术介绍 5](#_Toc515224490)

[1.3.1 Nginx简介 5](#_Toc515224491)

[1.3.2 Nodejs简介 5](#_Toc515224492)

[1.3.3 MongoDB简介 5](#_Toc515224493)

[1.3.4 SSL简介 5](#_Toc515224494)

[1.3.5 其他相关简介 6](#_Toc515224495)

[2. 系统分析 7](#_Toc515224496)

[2.1. 可行性分析 7](#_Toc515224497)

[2.2. 需求分析 8](#_Toc515224498)

[2.3. 运行环境及开发工具 8](#_Toc515224499)

[3. 系统总体设计 9](#_Toc515224500)

[3.1. 购买云服务器 9](#_Toc515224501)

[3.2. 配置云服务器 9](#_Toc515224502)

[3.3. 安装Nodej开发环境 10](#_Toc515224503)

[3.4. Nginx 服务器 10](#_Toc515224504)

[3.5. 配置网站域名 11](#_Toc515224505)

[3.6. 配置SSL 11](#_Toc515224506)

[3.7. 抓取车站、车次数据 11](#_Toc515224507)

[3.8. 后台分析整理 12](#_Toc515224508)

[3.9. 编写页面模板 13](#_Toc515224509)

[3.10. 前台功能设计 14](#_Toc515224510)

[3.11. 后台功能设计 15](#_Toc515224511)

[3.11.1. 爬虫管理模块 15](#_Toc515224512)

[3.11.2. 车站、车次信息管理模块 15](#_Toc515224513)

[4. 数据库设计 16](#_Toc515224514)

[4.1 MongoDB数据类型 16](#_Toc515224518)

[4.2 MongoDB数据库建模代码 16](#_Toc515224519)

[4.3 建模视图 19](#_Toc515224520)

[5. 系统运行界面 21](#_Toc515224521)

[5.1 桌面端 22](#_Toc515224523)

[5.2 移动端 22](#_Toc515224524)

[6. 系统测试与维护 22](#_Toc515224525)

[6.1 正确流程测试 23](#_Toc515224527)

[6.2 错误操作测试 26](#_Toc515224528)

[6.3 测试简报 28](#_Toc515224529)

[7. 结论与体会 28](#_Toc515224530)

**火车票查询系统**

软件工程专业 樊礼嶢 指导教师：韦 静

[摘 要] 在互联网兴起之前，火车票信息只能到火车站相应服务窗口询问才得知，旅客想获取某一车次或车站详细信息的方式比较单一，且困难、麻烦。这在一定程度上给旅客的出行构成很大的不便。随着社会的发展，计算机软件及硬件的不断更新迭代发展，互联网的兴起与普及，以及电脑的普及率不断上升、手机等移动设备的普及，从网络获取信息成了人们日常生活的首选，几乎替代了旧时代人们获取信息的所有方式。本系统正是基于这种趋势，参考现有的火车票查询网站，取其长，去其短而开发出来的在线实时获取火车票信息、以及查看各省份车站车次信息详情的车票查询系统。本系统基于Nodejs创建，使用快速的轻量级的Express Web框架进行开发，并基于HTTPS安全安全加密传输链接的构建起来的轻量级火车票查询系统，部署和运行于云服务器上，可通过浏览器或移动端进行访问。

[关键词] 火车票 站点 查询；Nodejs；Express；HTTPS

引言

伴随着互联网的快速兴起和软件编程开发技术的不断成熟，越来越多的企业或单位紧跟互联网发展的大潮，除线下提供相关信息获取渠道外，也在大力发展线上信息渠道。而同样地，计算机技术方面的高速发展，让大范围使用计算机来为人类工作已经变为可能。借助于计算机优越的运算能力，能够完成很多人工难以完成的、工作量巨大的工作，鉴于计算机出色的计算性能和高效的办公效率，计算机被成功地应用到各个领域来协助人类工作。本火车票查询系统可以协助我们来处理大量的车站、车次以及其他相关的信息。借助于计算机超强的处理能力来获取车站、车次的相关数据，并整理保存至数据库中，能让旅客查询火车票及相关站点信息时更简单易用。

1. 系统概述

## 系统开发背景

当下正处于一个互连网高速发展的阶段，由此产生的网络时刻在影响和改变着我们的生活方式，各式各样的电脑软件应用于各行各业以及生活中的方方面面，接入互联网早已成为人们日常生活中的一项不可或缺的事情。而在另一方面，互联网的便利性也大大地改变了人们获取信息的方式，从传统的大字报、墙报、报刊报纸过渡到现在的网页新闻、微博、知乎等，网络信息来源已成为人们日常生活的重要消息渠道。同样地，火车票信息亦是如此，从以前的窗口查问票，电话询问和人工打印报表等，前进到现在的网页12306客运服务中心以及其他的一系列查票抢票的网站，互联网对我们的生活方式产生了不可言喻的影响。所以，在网络获取信息也就成为最通用的渠道了。而火车票查询系统正是基于在线网络来运行的，正好符合这一大趋势。

此外，随着信息化社会步伐的不断前进，网络成为了人们日常生活中获取信息或资源的最重要途径之一。人类在生活中对网络的依赖程度也接近直线上升，而几乎所有的互联网查询网站也正好捉住此契机，在线为人们提供查询各类车票信息。与此同时，固然很多的网站都提供了丰富的车票查询的服务，但是都有着自己的盈利模式，都以自我的网站为中心，不单给用户提供了基础的查询服务，而且额外增添了其他的功能，以此来达到吸取流量和用户的运营目的。但这样的模式，不免让一部分用户反感，他们需要的是一个轻量级的网站应用，操作简易，功能实用且专一的，而同时不以盈利为目的的 Web 应用，因此，这一个轻量级的、只为用户提供所需的精确信息的火车票查询系统正是部分用户所期待的。

## 系统开发意义

既然网络已经登上了信息传播渠道的主流之座，那么，一个在线的，简洁的，实时查询火车票的查票系统肯定是必不可少的了。火车票查询系统摒弃了所有不必要的信息，目标是只为用户呈现最简洁，最有用的车票信息。火车票查询系统只专注于为用户提供一个可在线的、实时的火车票查询环境，并且提供各省车站车次详情的查看，内陆大部分车次的详细信息的查看，包括起止站、发车/到达时间、列车全程、列车运行时间、列车票价和途径车站等。火车票查询系统大大方便了出行的旅客，给用户提供了一个简洁、轻量、准确的车票查询系统。这不仅节约了用户的时间，而且给用户带来的是一种轻松、简洁和舒适的体验感，很大程度上降低了用户的信息疲劳度，并让用户对自己所关注的更聚焦，从而能让用户更专注于自己的行程路线及行程规划。

## 相关技术介绍

### Nginx简介

Nginx是一个HTTP服务器和反向代理，同时也是一个邮件协议代理服务器。Nginx是免费的，开源的，同时更是高性能的服务器。Nginx 闻名于世界是由于其高性能和高稳定性，还有多样的功能集插件，以及配置简易和低消耗。Nginx 不是基于线程驱动的，而是基于事件驱动。这使得 Nginx小巧而灵活，无论是小的VPS，还是大的服务器集群，Nginx 都可以轻松应对。故Nginx 逐渐成为主流服务器之一。

### Nodejs简介

Nodejs准确来说不是一门编程语言，它只是一个 JavaScript 运行时。Nodejs是基于谷歌的V8引擎来构建并运行的。 Nodejs使用的是非阻塞型IO模型，它基于事件驱动。Nodejs 的生态社区完善且成熟。

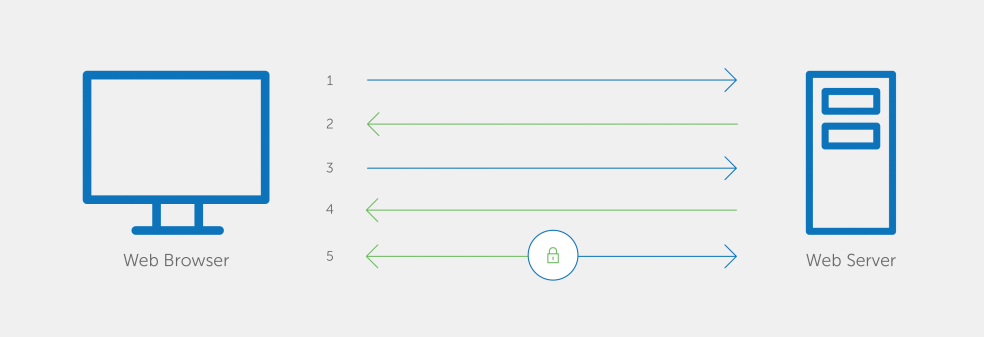
### MongoDB简介

MongoDB是一个非关系数据库。特点是免费、开源和跨平台。正是由于此项优点，MongoDB在网络实时方面的应用愈来愈广泛。其虽分类为非关系型数据库，但却是公认的比较类似于关系型数据的非关系型数据库，故“非关系”的另一种释义为“不仅仅是 SQL”。

### SSL简介

SSL 是安全套接字层（Secure Socket Layer），它属于安全技术的一种标准，主要用于在服务器和客户端（通常是Web服务器（网站）和浏览器，或邮件服务器和邮件客户端（例如Outlook））之间建立起加密链接。更为具体来说，SSL是一种安全协议。这种协议用来描述如何使用算法加密在网络上传输的数据。在这种情况下，SSL协议确定链路和正在传输的数据的加密变量。

“SSL握手”过程（请参见下图）



浏览器连接到使用SSL（https）保护的Web服务器（网站）。

服务器发送自己的加密证书的副本，包括用于加密的公钥。

浏览器根据可信的CA列表检查服务器所传输过来的证书，验证其证书是否过期，是否被撤销，并且其公共名称对于正在连接的网站是否有效。 如果浏览器信任证书，它将使用服务器的公钥创建，加密并发回对称会话密钥。

服务器使用其私钥解密对称会话密钥，并发回用会话密钥加密的确认以启动加密会话。

服务端和客户端“握手”完成后使用会话密钥加密所有传输的数据。

### 其他相关简介

* Express简介

Express是一个轻量级的网络应用框架，基于Nodejs构建，具有简约、快速和非选择性的特点。

* Mongoose简介

Mongoose模块主要用于操作MongoDB数据库。其简单的、基于模式的解决方案为数据库的建模提供了极大的便利条件。

* C9 IDE简介

Cloud9 IDE是一个在线的集成开发环境，从3.0版开放源码发布。它支持数百种编程语言，包括但不限于C，C ++，Ruby，Perl，Python，PHP，并且带有Nodejs的JavaScript和Go。它使开发人员能够立即开始使用预先配置的工作区进行编码，使用协作编码功能与同行进行协作，以及实时预览和浏览器兼容性测试等Web开发功能。

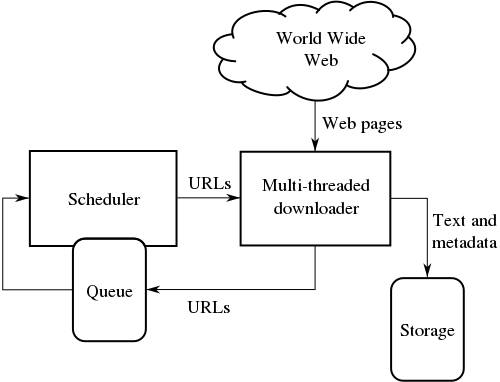
* Cheerio简介

Cheerio 模块是专门为服务器而生的jQuery，不仅快速，而且灵活。相当于运行于服务器端的功能强悍的 jQuery。

* 爬虫简介

网络爬虫，有时被也称为蜘蛛（spider），是一种互联网机器人程序。是对所有互联网的网站内容或其他网站的网页内容的索引。网络爬虫复制页面以供搜索引擎处理，搜索引擎保存索引下载的页面，以方便用户使用。爬虫在爬取资源时很消耗对方主机的系统资源，并经常未经批准地去访问网站。不过，时间表、负载和“礼貌”问题在大量访问页面时发挥重要作用。对于不希望被抓取的公共站点可以向爬虫代理公开这些机制。例如，包含一个robots.txt文件可以要求机器人只索引一个网站的一部分，或者什么也不做。。在今天，搜索引擎几乎可以把相关问题的结果立即给出。

爬虫爬取原理图



1. 系统分析

## 可行性分析

* + 1. 技术可行性

Nodejs是当前越来越热门的编程语言之一，以及互联网和微机技术的高速发展，Nodejs基于Chrome的V8引擎，在编程语言中的地位也愈来愈重要。但起初的Nodejs因各种问题而被开发人员所诟病，然而经过一次次的更新迭代，Nodejs现在已经是前后端通吃，不仅可服务于后端服务器的开发，而且同样可以用于前端的开发。

NPM是用来管理Nodejs软件包的系统，是目前已存在的世界上最大的开源库。也就是说，通过Nodejs和NPM，就可以找到任何一个技术比较新的、可用于你的项目的软件包，并且NPM提供完善的依赖关系处理，开发者完全不用担心软件包之间的依赖关系管理问题，从而更专注于项目代码的开发，把更多的精力投入到开发中。

MongoDB 是一个非关系数据库，功能最为丰富，但却又最类似关系型数据库的。MongoDB 支持非常松散的数据结构，即类似 json 的 bjson，从而被赋予了存储复杂数据的能力。MongoDB支持强大的类似SQL的查询语言，并且其语法类似面向对象的查询语言。此外，MongoDB还支持对所存储的数据进行索引，这样更进一步地加快了信息的查询速度。

由于上述的 Nodejs、MongoDB数据库 等技术早已趋于成熟，相关的社区比较活跃，知识点比较全面，网络上教程相对完整，故本系统在技术上完全可行。

* + 1. 经济可行性

本系统是一个轻量级的火车票查询系统，主要为用户提供实时的火车票信息查询功能，暂不向用户提供一键购票、退票等功能。故此系统的开发周期相对较短，部署运行环境所需成本低，而且此项目可运行于较低配置的系统之上，基本不用担心配置问题，后期所需维护成本低，也不必担心过多的经济问题，在经济方面是非常可行的。

* + 1. 操作可行性

本系统采用B/S架构，仅需使用浏览器打开网址即可使用，系统操作简单、符合用户操作习惯，功能完备，不仅用户体验较好，而且系统运行流畅，相应速度快。除上述点以外，不仅可以实时查询火车票信息，而且可以展示给用户友好的界面，为用户出行提供了更便捷的信息获取方式。相对于用户来说，上手简易，操作容易，相对于系统管理员来说，功能模块划分良好，各功能模块之间的耦合度低，操作功能易于读懂，维护管理工作简单，非必需专业的技术技能，只需一定的讲解即可上手。无论是在用户层面，还是在管理员层面，本系统的所需操作合乎常理，并易于实现，故具有操作可行性。

* + 1. 安全性

本系统中所有涉及的数据都使用HTTPS进行加密传输，在传输过程中所有数据加密，很大程度上提高信息传输的安全行，并且在建立传输会话会进行单向检验，即客户端会校验服务器的证书、签名等是否合法、有效等，这在很大程度上避免了网络中间人的窥视。而后台管理功能则需要提供管理员的账号密码并且校验合法时才可登录，从而大大地降低了系统被入侵的危险，保证了能让用户与系统之间多一层安全的保护层。故此系统的安全性目前是毋庸置疑的。

## 需求分析

火车票查询系统所提供的功能是在线实时查询车票、车次等信息。为实现此目标，需要完成的工作有如下：

首先，本系统的运行需依赖于网络连接。由于此系统提供在线查询的功能，故需要搭建或者将此项目挂载于 HTTP 服务器之上，并且保持运行；其次，此系统应可以接受用户合法的输入，并且处理不合法的输入；然后，当用户的输入合法时，应当返回用户所搜索的车票或车站、车次信息，当用户输入不合法时，应当用返回合理的提示；再次，当用户没有输入时，点击相应的车站、车次信息入口应当返回正确的车站、车次信息，并且最终渲染在浏览器上，以友好的界面显示结果给用户；最后，需具有爬虫爬取、整理数据模块。此模块应具有的功能有：爬虫先爬取国内所有的车站、车次信息，当抓取完成后，需要分析、筛选数据，并且还有验证数据的有效性；通过校验的数据应当在数据库中存储，以用于后面的查询。

## 运行环境及开发工具

硬件平台：

CPU：1GHz

内存：512M以上

硬盘：2G以上

软件平台：

操作系统：Ubuntu 16.04

数据库：MongoDB

Web服务器：Nodejs、Nginx反向代理服务器

开发语言：Nodejs、MongoDB

开发工具：C9 IDE (CLOUD 9 IDE)

1. 系统总体设计
   1. 购买云服务器

需要自行购买一台云服务器，服务器操作系统以 Linux 主流发行版较好，例如： CentOS、Redhat、Ubuntu 、Scientific Linux、Debian、Fedora、Suse等。

最低配置需求：

CPU：1 GHz

内存：512 M

硬盘：2 G

当然，配置越高越好。在此项目中，采用的是 Ubuntu 16.04 LTS Server 版，操作系统为64位，以下所用到的命令皆基于此环境下运行。

* 1. 配置云服务器
* 登录系统

用服务器提供商提供的账号密码登录远程的系统，登陆命令示例如下：

ssh username@host

# username服务器提供商提供的用户名

# host 服务器的地址

* 新增用户

添加一个普通用户，添加的同时创建用户的家目录，并且把此username用户添加进sudo组，以方便此用户后面用来登录并配置项目相关运行环境

# 添加用户

useradd -m username

# 设置用户密码

passwd username

# 把用户加进 sudo 组

usermod -aG sudo username

* 配置SSH连接

用其它电脑生成一对密钥

ssh-keygen -t rsa

把生成的公钥上传至服务器

ssh-copy-id user@host

修改 /etc/ssh/sshd\_config文件，禁止 root 用户远程登录，禁止使用密码登录

sudo nano /etc/ssh/sshd\_config

* 重启 ssh 服务

sudo systemctl reload sshd.service

配置项目运行环境

* 1. 安装Nodej开发环境

curl -sL https://deb.nodesource.com/setup\_8.x | sudo -E bash –

sudo apt-get install -y nodejs

下载相应系统的安装文件

测试是否安装成功

nodejs -v

* 1. Nginx 服务器

安装

# 更新系统源

sudo apt-get update

# 安装 nginx

sudo apt-get install nginx

配置防火墙

sudo ufw allow 'Nginx HTTP'

启动 Nginx

sudo systemctl start nginx

* 1. 配置网站域名

首先在域名提供商购买一个或多个域名

购买成功后登录域名管理控制台，添加新的域名解析，通过域名解析把域名和服务器的 IP 关联，此后即可通过域名来访问服务器，访问部署上去的项目

* 1. 配置SSL

安装配置 SSL 所需的软件

$ sudo apt-get update

$ sudo apt-get install software-properties-common

$ sudo add-apt-repository ppa:certbot/certbot

$ sudo apt-get update

$ sudo apt-get install python-certbot-nginx

开始配置 SSL

$ sudo certbot –nginx

设置自动更新证书

$ sudo certbot renew --dry-run

* 1. 抓取车站、车次数据

配置爬虫开发环境

安装 cheerio、request等编写爬虫所需的 nodejs 模块

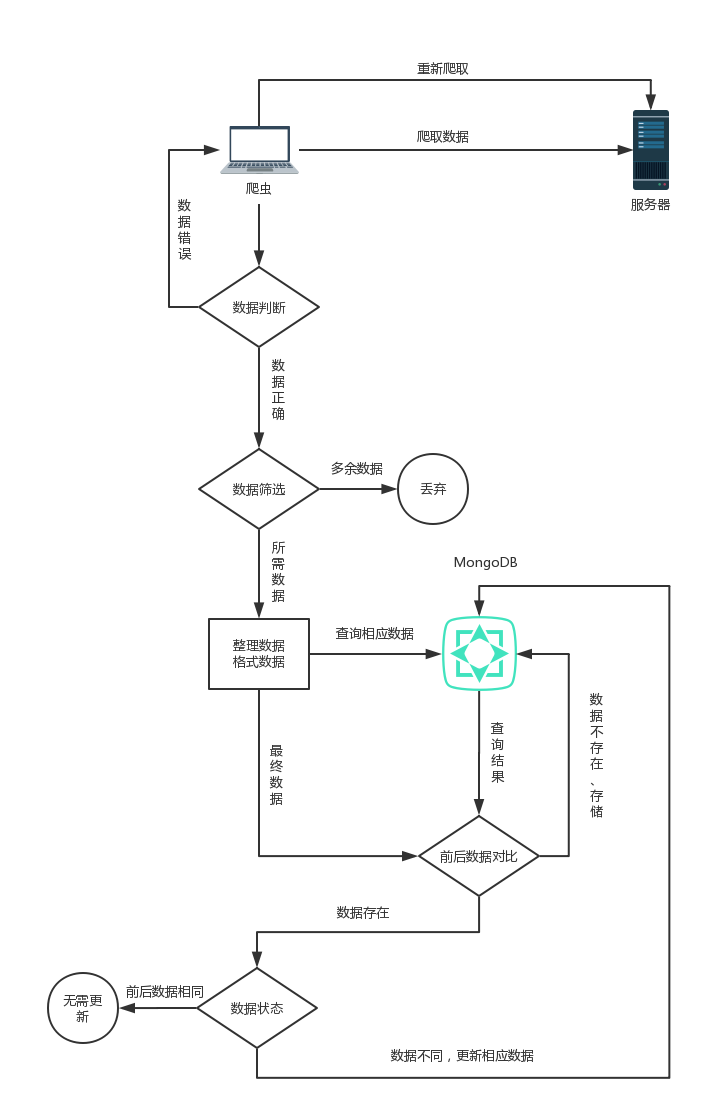
npm install –save cheerio request

爬虫相关模块安装完成后即可编写爬虫测试代码

待爬虫测试代码成功爬取数据后，即进行多线程的信息爬取进程。在爬取时，若数据出错，则重新开始爬取当前项，若3次尝试后仍失败，则跳过此项；若爬取成功，则对所爬取回来的数据进行筛选，保留项目所需数据。

使用 MongoDB 存储筛选后的数据，以便后续的使用

爬虫爬取流程图



* 1. 后台分析整理

分析此前存储的数据

把存储于数据库的数据再次取出来，对此进行分析。当数据重复或者是无效时，则删除此项数据，否则，按照省、市等级别来进行分类存储，或存储相应的关联关系。

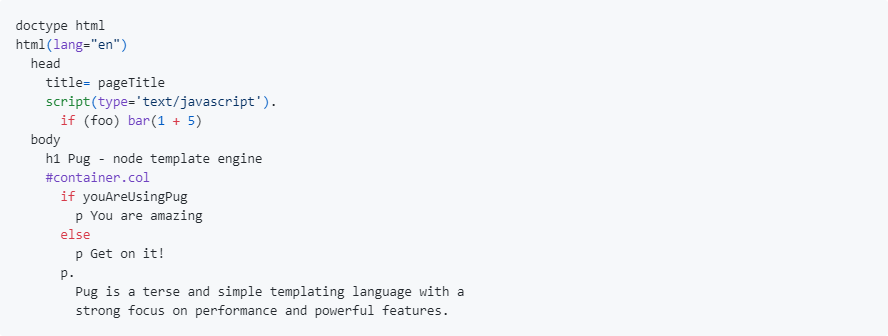
* 1. 编写页面模板

页面模板使用Pug 模板引擎

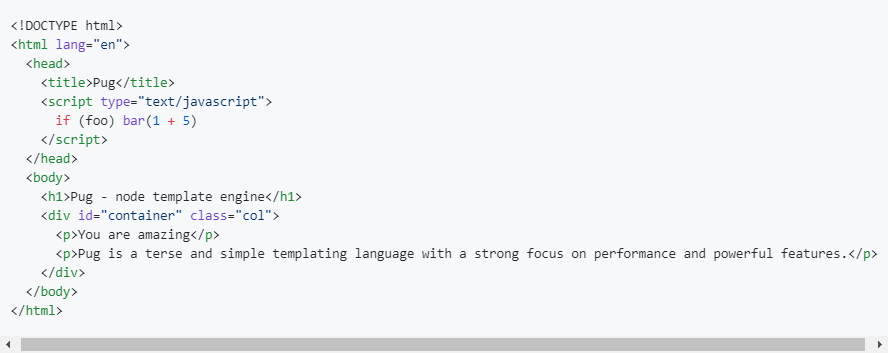
Pug引擎再结合 Bootstrap CSS v4 框架构建页面的基础框架，此后的页面基于此基础框架构建。

Pugjs 使用示例

Pugjs 源码



Pugjs 编译后代码



( 1 ) 从 MongoDB 获取数据，经路由再次整理分析，传入页面框架分别从 MongoDB 获取相对应的数据，分析、整合，最后通过相应的路由传送到相应的页面模板上进行解析

接收数据，解析数据并整合至框架分门别类来接收并判断来源数据，不同的数据归为不一样的地方，不同的页面模板解析不同的数据内容，最后整合至同一类的页面模板引擎处

( 2 ) 调整并展示页面视图

( 3 ) 根据CSS框架来对页面布局进行微调

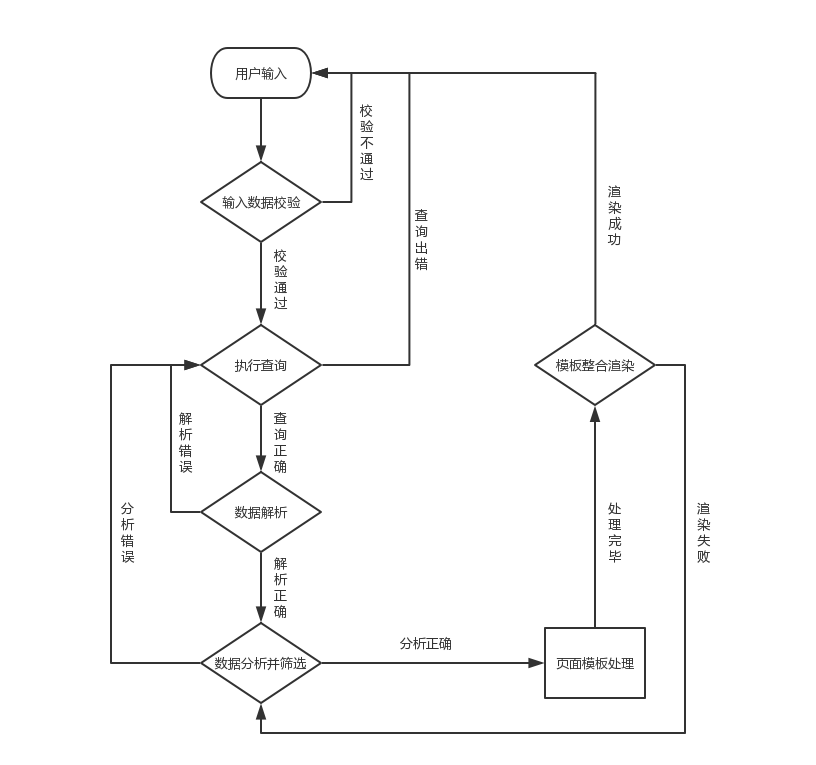
( 4 ) 颜色搭配等细节处理

( 5 ) 整合并编写前台展示页面

* 1. 前台功能设计

实时车票信息查询

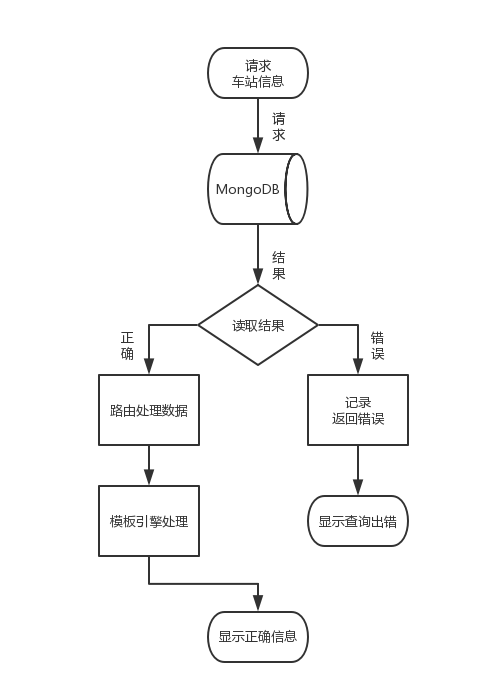
可以实时查询国内所有可在12306客运服务中心查到的车票信息，并以友好的界面返回给用户，以便于用户查看

用户查询流程图如下：

提供国内大陆范围内所有省份车站信息的查看

经数据库及路由的整理，分别显示各个省份，各省份的车站总数，每个车站的链接网址

省份车站信息流程图如下：



* 1. 后台功能设计
     1. 爬虫管理模块

此模块用于管理爬虫的启动、爬取数据等功能。当启动爬虫时，必需安装所提示的步骤一步步进行爬取，否则不能爬取到所需的车站、车次数据，因为每一项后面的数据抓取操作都取决于前一项的结果。只有当前一项（第一项除外）有并且是正确的数据时，才能保证下一项的爬取可以成功运行。

* + 1. 车站、车次信息管理模块

此模块用于管理数据库中的车站、车次信息。模块的内容主要是用于管理员对车站、车次信息的微改，即只能对数据库进行少量的更新操作，当车站、车次信息临时有变时可用。此外，此模块还担当了清空数据库省份、车站、车次等信息的重任，当需要全部更新所有数据时需要此模块来先清空之前的旧数据，再存储新的数据，以防止数据污染。

1. 数据库设计

## MongoDB数据类型

String 字符型，用于存储字符串的数据类型。只能存储有效的UTF-8字符。

Integer 整数型，用于存储整数。

Boolean 用于存储一个布尔（真/假）值。

Double 浮点型，用于存储double 型数据。

Min/ Max keys 是用来比较一个元素与最低和最高的 BSON 值。

Arrays 数组型，存储数组。

Timestamp ctimestamp。 记录文档被修改或添加的时间。

Object 对象型，存储一个Object对象。

Null 用于存储一个空值。

Symbol字符串类型。 用来报仇特定符号。

Date 时间类型，存储以UNIX时间格式的当前日期或时间。

Object ID 一份文档的唯一标识符。

Binary data 存储二进制数据。

Code 存储JavaScript代码。

Regular expression 存储Reg表达式。

## MongoDB数据库建模代码

CodeSchema

Model Code 详解

code: 存储列车代码（编号），类型为 String

Status: 存储列车的代码（编号）的状态，1 为默认值，默认为此条记录有效，0 表示此条记录无效，类型为 Boolean

codeheader: 内嵌型 document，里面分别存储了以下内容：

ctype: 列车的类型，类型为 String

bstation: 列车起始站，类型为 String

dstation: 列车终点站，类型为 String

ltime: 列车发车时间，类型为 String

atime: 列车到达时间，类型为 String

dist: 列车运行全程，类型为 String

ctime: 列车运行时间，类型为String

price: 列车票价，分别存储无座，硬座，硬卧，软卧，高级软卧，二等，一等，商务等座次的信息，类型为 Array

station: 列车途经站点，存储当前列车起始站与终点站之间的车站，类型为 Array

let CodeSchema = new Schema({

code: String,

status: {

type: Boolean,

default: 1

},

codeheader: {

ctype: String,

bstation: String,

dstation: String,

ltime: String,

atime: String,

dist: String,

ctime: String,

price: Array

},

station: Array

})

车次信息 (code collection)

|  |  |  |
| --- | --- | --- |
| fieId | 数据类型 | 备注 |
| code | String | 车次代码 |
| status | Boolean | 车次状态 |
| codeheader.ctype | String | 车次类型 |
| codeheader.bstation | String | 始发站 |
| codeheader.dstation | String | 终点站 |
| codeheader.ltime | String | 发车时间 |
| codeheader.atime | String | 到达时间 |
| codeheader.dist | String | 列车全程 |
| codeheader.ctime | String | 运行时间 |
| codeheader.price | String | 车票价格 |
| station | Array | 途径车站 |

StationNameSchema

Model StationName 详解

name: 车站名称，类型为 String

aliascode: 车站特有的唯一的编码，类型为 String

aliasname: 车站别名（即车站的拼音字符串），类型为 String

let StationSchema = new Schema({

name: String,

aliascode: String,

aliasname: String

})

车站名 (stationname collection)

|  |  |  |
| --- | --- | --- |
| fieId | 数据类型 | 备注 |
| name | String | 车站名称 |
| aliascode | String | 车站代码 |
| aliasname | String | 车站别名 |

ProvinceSchema

Model Province 详解

name: 省份名称，类型为 String

station: 存储该省的车站名数组，类型为 Array

let ProvinceSchema = new Schema({

name: String,

station: Array

})

省份车站 (province collection)

|  |  |  |
| --- | --- | --- |
| fieId | 数据类型 | 备注 |
| name | String | 省份名称 |
| station | Array | 省份车站 |

ProvinceDetailSchema

Model ProvinceDetail 详解

station: 车站名称，类型为 String

status: 车站状态，1 为默认值，默认此条记录有效，0 表示此条记录无效，类型为 Boolean

code: 存储经过此车站的所有车次（不包括重复车次），类型为 Array

let ProvinceDetailSchema = new Schema({

station: String,

status: {

type: Boolean,

default: 1

},

code: Array

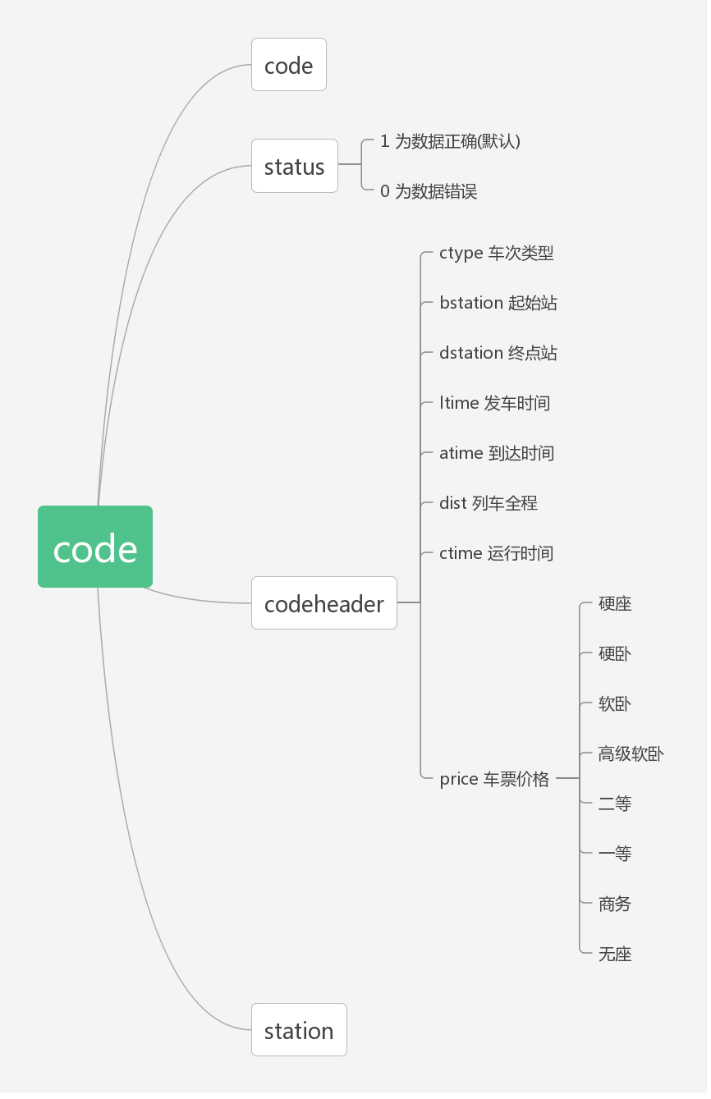
})

车站详情 (provincedetail collection)

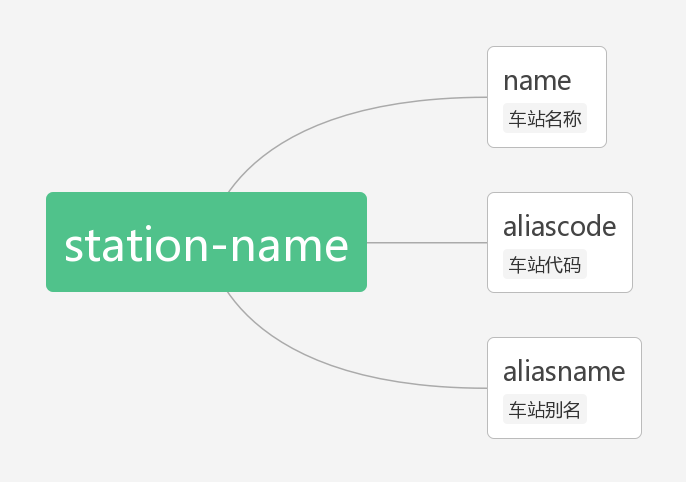
|  |  |  |
| --- | --- | --- |
| fieId | 数据类型 | 备注 |
| station | String | 车站名称 |
| status | Boolean | 车站状态 |
| code | Array | 车站车次 |

## 建模视图

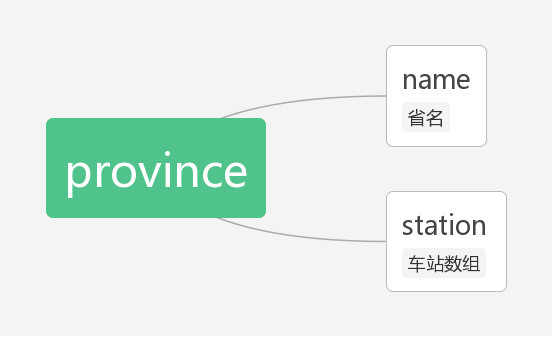
车次信息



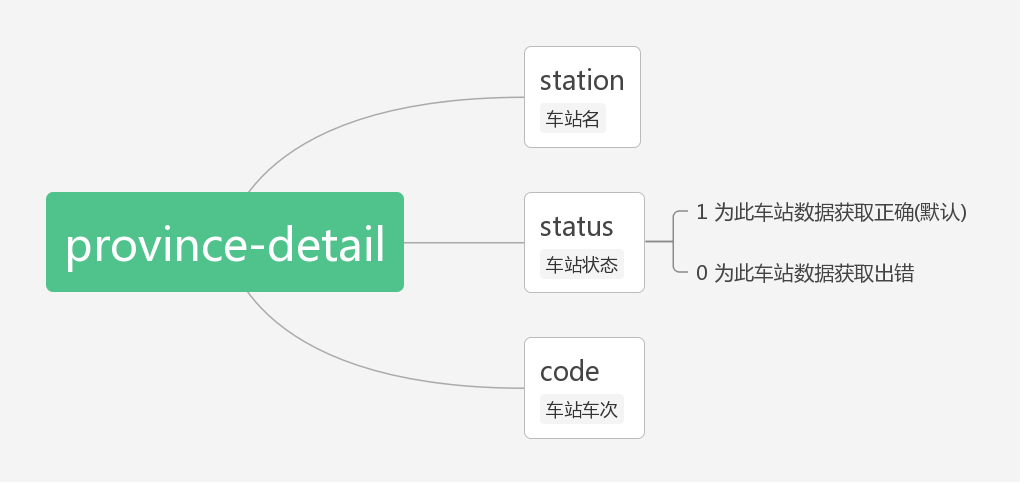
车站名称



省份车站



车站详情



1. 系统运行界面

## 桌面端



## 移动端



1. 系统测试与维护

## 正确流程测试

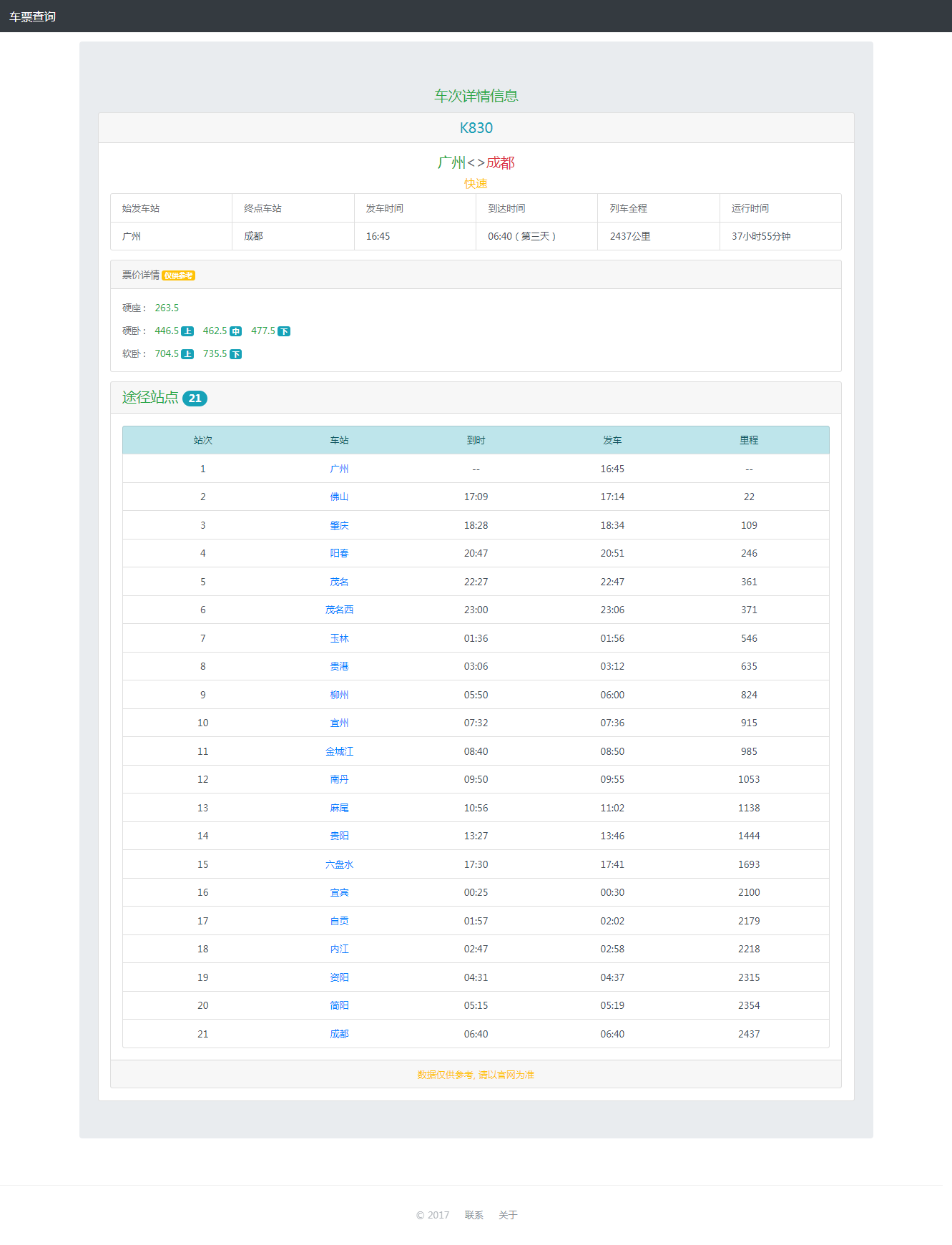
首页查询



查询结果



车次详细信息



车站详情



爬虫控制台



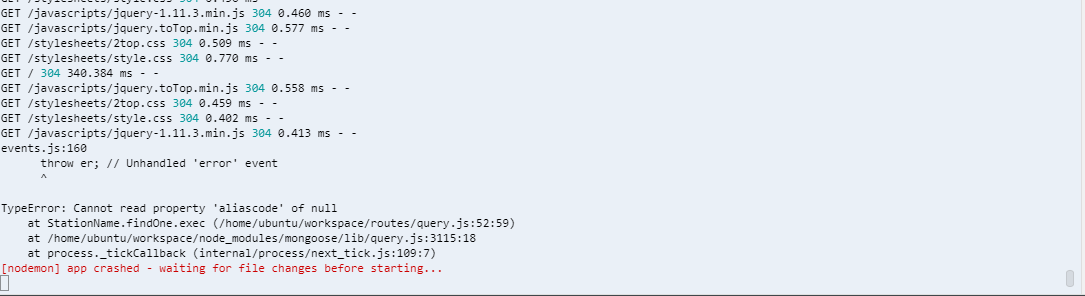
爬虫高级操作



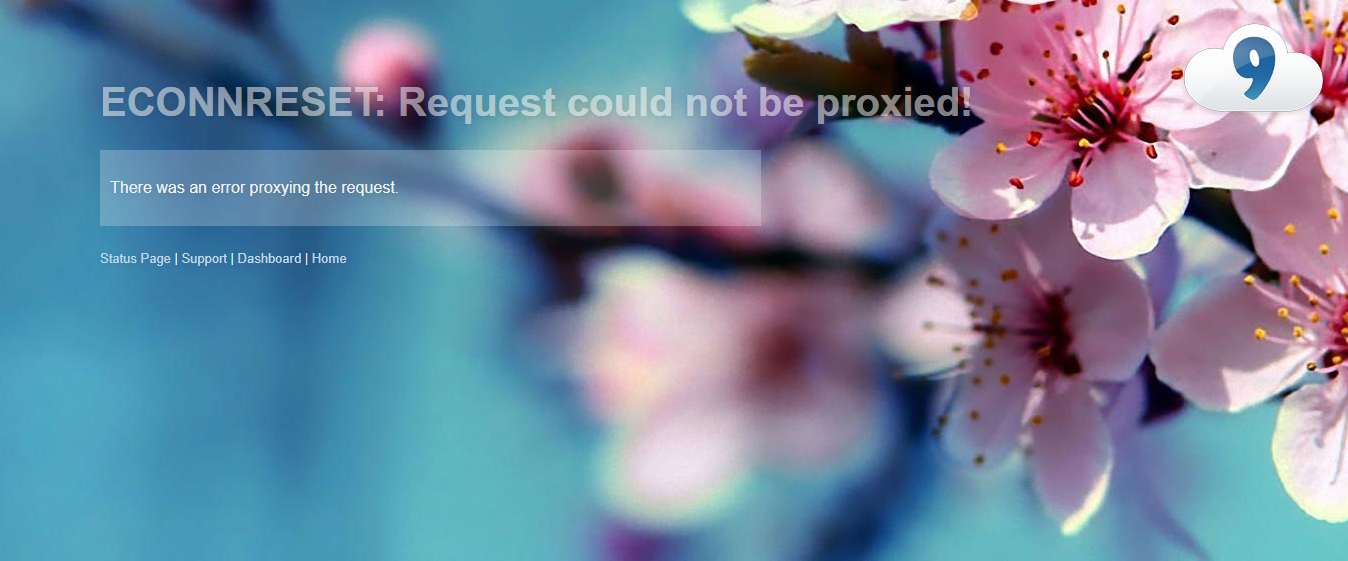
经以上测试，所有正确的输入都会得到正确的输出，正确的信息反馈。证明 了此系统的正确性。

## 错误操作测试

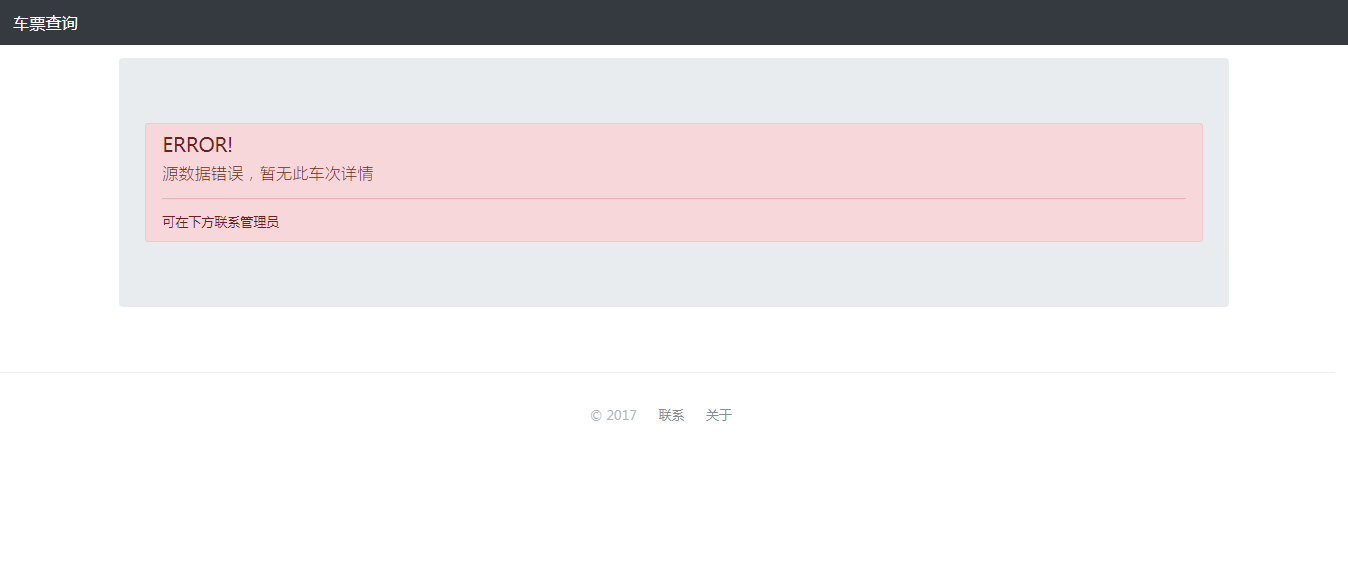
错误输入



页面报错



查询出错



数据不存在



## 测试简报

本项目所有模块皆通过了测试，期间先是进行了每个模块的单元测试，确认单元测试无误后，进行下一步的模块衔接测试。在测试中，对于正确与不正确的输入都有相应的处理模块，适应了不同的环境需求，测试结果也在预期当中。当每一次正确的查询都会返回给用户正确的信息，并且渲染于浏览器内；每一次成功的操作都有相应的成功的提示，不一样的操作则有不一样的返回提示。与此同时，每一次的错误的操作都反馈回错误信息，并打印在控制台用户或管理员可根据提示信息来进行操作调整，从而达到了引导用户或者管理员正确操作的目标。

1. 结论与体会

本项目已达成可以在线实时查询火车票余票的信息的功能，对于所提出的需求基本已经满足。但是，虽然项目已完成，但项目还有很多的需要继续进行，比如：优化网站的响应速度；客户端缓存静态文件以降低网络请求，从而达到减缓服务器的压力以及优化数据的存储结构等。

通过本次的项目设计，我不仅学会了如何分析软件的项目需求，而且也学会了如何设计项目流程，并且如何让代码更优美，后期更容易维护。同时也相当于有了一次完整项目的开发经验，为以后的开发提供了一层铺垫。在此次项目中，不仅让我了解到了计算机编程世界的多姿多彩，而且也了解到具体的代码的具体作用、涵义。

1. 致谢