成 绩:\_\_\_\_\_\_\_\_\_\_\_\_

教师签名:\_\_\_\_\_\_\_\_\_\_\_\_



**需求分析说明书**

**项目名称:** 火车车次信息管理

教 师: 胡峡进

姓 名: 唐可寅

学 号: 201710253110

班 级: 17gb计算3班

日 期: 2020年07月06日

目录

[一、概述 1](#_Toc1080766628)

[1、摘要 1](#_Toc1028030470)

[2、系统的应用背景 1](#_Toc1602169175)

[3、系统的开发意义 1](#_Toc359874492)

[二、需求分析 2](#_Toc1096637092)

[1、可行性研究 2](#_Toc1474946690)

[2、 系统需求分析 2](#_Toc1025281509)

[三、系统总体设计 3](#_Toc497538235)

[1、系统设计 3](#_Toc1971277874)

[项目环境和依赖 3](#_Toc2037006049)

[分层分析 4](#_Toc776365069)

[2、系统详细设计 9](#_Toc257075511)

[前言 9](#_Toc2078499260)

[概要设计 9](#_Toc220577071)

[详细设计 9](#_Toc682057575)

[四、系统的编码、测试和总结 12](#_Toc73955339)

[1、系统代码 12](#_Toc1721834607)

[2、系统的测试 15](#_Toc1532096524)

[五、总结 17](#_Toc1617351338)

# 一、概述

## 1、摘要

随着高铁在国内铺设的越来越广泛，高铁出行也逐渐成为人们的一种主要长途出行方式。在线火车车次查询系统在此环境下应运而生。火车车次查询是一项进行信息获取的软件，应用广泛，并且是实时性非常强的一个非常典型的管理信息系统。人们使用本系统，从本系统中获取所需信息，而它的信息来源依赖车站工作人员对火车运行次序对排期。

在此背景下，本此课程设计首先考虑了我国交通的现状和不同乘客的实际要求，分析了此系统应该具有的功能；其次对系统的主要功能模块进行了详细地描述，其中主要包括各列车查询模块和列车信息管理更新模块，最后结合系统开发阶段和调试阶段发现的问题，提出了系统需要完善的地方，总结了开发此系统所取得的经验和体会。

火车车次信息管理系统：是为广大群众开发的，帮助人们查询车次信息，大大节约了人们的时间，是铁路营运公司所需要的全部功能的一个综合的管理系统。

本次设计首先对系统建设对各个层面进行了基本定义，并在其基础上进行了简单分析，阐述中给出了部分UML图以说明系统结构。本次设计以基于文本信息传送和接收功能的聊天室系统为基准实行此次系统的开发。

## 2、系统的应用背景

开发的软件系统的名称：火车车次信息系统

本软件产品是一项独立的软件，而且全部内容自含。实现网络化铁路售票管理。一般用户、售票员、系统管理员等用户都可以通过上网来进行铁路车票业务的管理，实现网络化售票业务。构建标准的铁路车票，火车管理基础数据库。构建起火车信息，车票信息，特殊用户信息等基础数据库。

实现列车及车票信息查询、登录系统及信息管理、车票的销售与退票列车及车票管理等子系统的流程化管理。

## 3、系统的开发意义

本火车车次管理信息系统设计的目的是为了方便乘客可以更好的找到自己所需要的列车，使用者通过本系统查询出发地与目的地之间可搭乘的列车车次信息，这样可以在最短的时间里帮助使用者找到自己应搭乘的列车，

通过该系统的查询主要是目的地查询和乘客所在地的时刻的查询，这样在最短的时间里让乘客找到自己所要的列车，也方便了盲目乘车的问题，最大可能的减少了列车的拥堵的现象。

# 二、需求分析

## 1、可行性研究

（1）管理可行性

随着时代的发展，人员素质己经逐步提高，不论是对于计算机系统的基本操作还是对于系统的维护都有了一定的基础。同时还可以配置专业的计算机维护人员来维护电脑，不必担心计算机故障问题。

1. 技术可行性

随着国内软件开发的日益发展壮大，网站开发的软件层出不穷，诸如Dreamweaver, Freehand, Fireworks, Flash 等工具都是网站开发的理想工具。本系统是在IntelliJ IDEA中进行代码设计与Java编程的，它属于一个 Web 数据库应用类的系统，其基本操作是建立 Web 界面和对数据库中信息进行录入、删除、修改等操作。从技术结构上来说是可行的。

## 系统需求分析

系统的工作总体规划由该系统管理人员在系统中完成对各种所需的基本数据的维护，包括相关列车信息的增加、修改以及对各项信息的变动都将在这进行操作。

后台管理主要由系统建设人员和系统管理员进行操作。他们可以进行各种列车信息以及各城市车站信息的录入、修改和删除等。

前台主要是让旅客了解各种列车信息以及车站最新动态情况，还有实现对旅客所需信息的査询和最新的新闻动态等功能。

本系统包含如下基本功能：

1. 浏览器端页面
   1. 提供车次列表查询功能；
   2. 提供车次详细信息查询功能；
   3. 提供基于搜索获取车次信息的功能；
   4. 提供友好的用户界面；
   5. 提供车次信息删除的功能；
2. 服务端
   1. 提供向所有需求用户提供车次列表的能力；
   2. 提供车次信息删除的功能。

# 三、系统总体设计

## 1、系统设计

### **项目环境和依赖**

* 开发环境
  + 开发语言：JDK 1.8 (1.8.0\_221)
  + 开发工具：IntelliJ IDEA 2020.1.2
  + 数据库：MySQL 8.0.20
  + Spring Boot：2.0.6.RELEASE
* 环境配置
  + 访问路径：http://127.0.0.1:8095/train/
  + 服务端口号：8095
* 第三方库依赖
  + javax.servlet: javax.servlet-api 3.1.0
  + javax.servlet.jsp: jsp-api 2.2
  + javax.servlet: jstl 1.2
  + org.apache.tomcat.embed: tomcat-embed-jasper 8.5.34
  + org.springframework.boot: spring-boot-starter-tomcat 2.0.6.RELEASE
  + log4j: log4j 1.2.17
  + org.springframework.boot: spring-boot-starter-test 2.0.6.RELEASE
  + org.springframework.boot: spring-boot-starter-web 2.0.6.RELEASE
  + org.springframework.boot: spring-boot-devtools 2.0.9.RELEASE
  + org.springframework.boot: spring-boot-configuration-processor 2.0.6.RELEASE
  + org.springframework.boot: mybatis-spring-boot-starter 1.3.1
  + mysql: mysql-connector-java 8.0.20
  + org.projectlombok: lombok 1.18.12
  + junit: junit 4.13

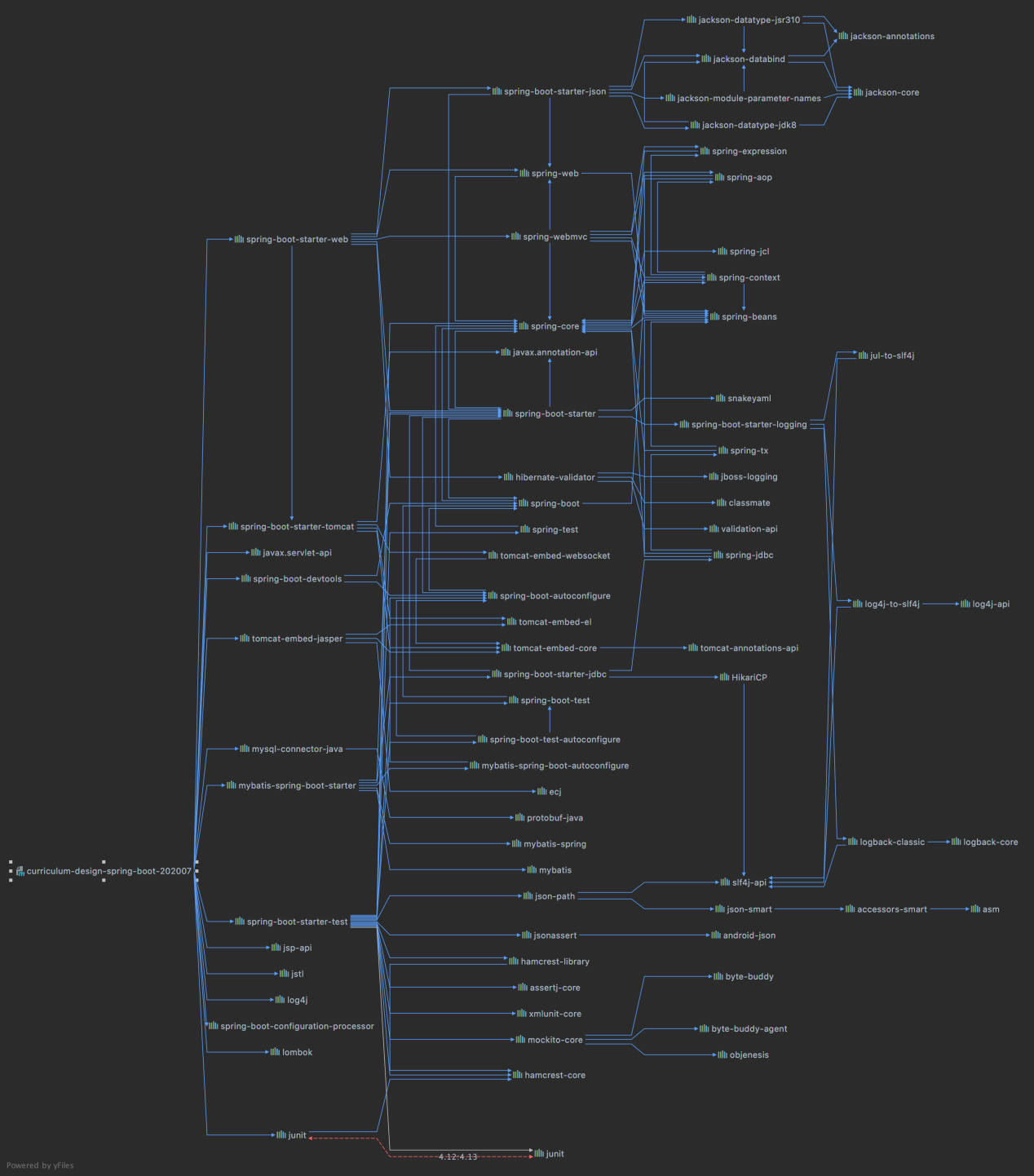


图 1第三方库依赖

### 分层分析

1. 驱动和中间介层
   1. 数据库驱动-JDBC

JDBC（Java Database Connectivity，Java 数据库连接）是一种用于执行 SQL 语句的 Java API 可以为多种关系数据库提供统一访问，它由一组用 Java 语言编写的类和接口组成。JDBC 提供了一种基准，据此可以构建更高级的工具和接口，使数据库开发人员能够编写数据库应用程序。

JDBC 是个“低级”接口，也就是说，它用于直接调用 SQL 命令。在这方面它的功能极佳，并比其它的数据库连接 API 易于使用，但它同时也被设计为一种基础接口，在它之上可以建立高级接口和工具。高级接口是“对用户友好的”接口，它使用的是一种更易理解和更为方便的 API，这种 API 在幕后被转换为诸如 JDBC这样的低级接口。

在关系数据库的“对象/关系”映射中，表中的每行对应于类的一个实例，而每列的值对应于该实例的一个属性。于是，程序员可直接对 Java 对象进行操作；存取数据所需的 SQL 调用将在“掩盖下”自动生成。此外还可提供更复杂的映射，例如将多个表中的行结合进一个 Java 类中。

随着人们对 JDBC 的兴趣日益増涨，越来越多的开发人员一直在使用基于 JDBC 的工具，以使程序的编写更加容易。程序员也一直在编写力图使最终用户对数据库的访问变得更为简单的应用程序。例如，应用程序可提供一个选择数据库任务的菜单。任务被选定后，应用程序将给出提示及空白供填写执行选定任务所需的信息。所需信息输入应用程序将自动调用所需的 SQL 命令。在这样一种程序的协助下，即使用户根本不懂 SQL 的语法，也可以执行数据库任务。

JavaSoft 提供三种 JDBC 产品组件，它们是 Java 开发工具包（JDK）的组成部份：JDBC 驱动程管理器、JDBC 驱动程序测试工具包和 JDBC-ODBC 桥。

JDBC 驱动程序管理器是 JDBC 体系结构的支柱。它实际上很小，也很简单；其主要作用是把 Java 应用程序连接到正确的 JDBC 驱动程序上，然后即退出。

JDBC 驱动程序測试工具包为使 JDBC 驱动程序运行您的程序提供一定的可信度。只有通过 JDBC 驱动程序测试的驱动程序才被认为是符合 JDBC 标准 TM 的。

JDBC-ODBC 桥使 ODBC 驱动程序可被用作 JDBC 驱动程序。它的实现为 JDBC 的快速发展供了一条途径，其长远目标提供一种访问某些不常见的 DBMS（如果对这些不常见的 DBMS 未实现 JDBC）的方法。

使用的时候，需要首先下载相应的驱动，然后配置 Java 工程里面的环境，再书写代码。因为 JDBC 是一个开放的标准，所以在使用的时候。需要先用反射加载驱动。可以使用 ClassLoader，也可以直接用 Class.forName 进行反射，反射的时候会自动运行驱动程序类的静态代码块，在静态代码块里面会进行数据库连接的初始化。同时因为这个关系，如果静态代码块初始出现异常，数据库功能将不能使用，除非重启程序。

1. Spring Boot框架层

Spring Boot 是所有基于 Spring 开发的项目的起点。Spring Boot 的设计是为了让你尽可能快的跑起来 Spring 应用程序并且尽可能减少你的配置文件。简单来说就是Spring Boot其实不是什么新的框架，它默认配置了很多框架的使用方式，就像Maven整合了所有的jar包，Spring Boot整合了所有的框架。

Spring Boot四个主要特性：

1. Spring Boot Starter：他将常用的依赖分组进行了整合，将其合并到一个依赖中，这样就可以一次性添加到项目的Maven或Gradle构建中；
2. 自动配置：Spring Boot的自动配置特性利用了Spring 4对条件化配置的支持，合理地推测应用所需的bean并自动化配置他们；
3. 命令行接口：（Command-line-interface, CLI）：Spring Boot的CLI发挥了Groovy编程语言的优势，并结合自动配置进一步简化Spring应用的开发；
4. Actuatir：它为Spring Boot应用的所有特性构建一个小型的应用程序。但首先，我们快速了解每项特性，更好的体验他们如何简化Spring编程模型。
5. Model和View层
   1. Model和View层的关系

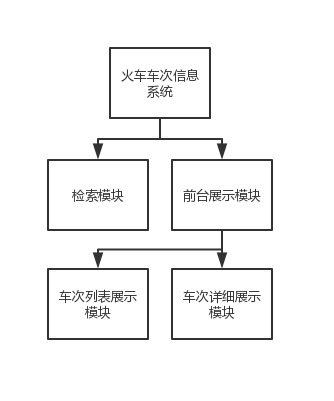


图 2 Model和View层的关系E-R图

1. Constant层

该层根据阿里巴巴Java代码设计在应用模块包下建立constant包，建立ConstConfig程序类存放应用级共享常量，这将可以消除编码过程中可能出现的绝大多数魔法值，使应用返回信息统一，实现一处修改多处同步，使程序复杂度降低。

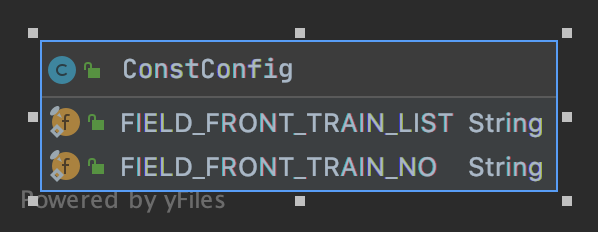


图 3 Constant层ConstConfig类图

1. DAO层

火车信息表：用来存储列车车次基本信息，如车次编号、始发站、终点站、出发时间、到达时间、车次类型、运行时间、里程等。

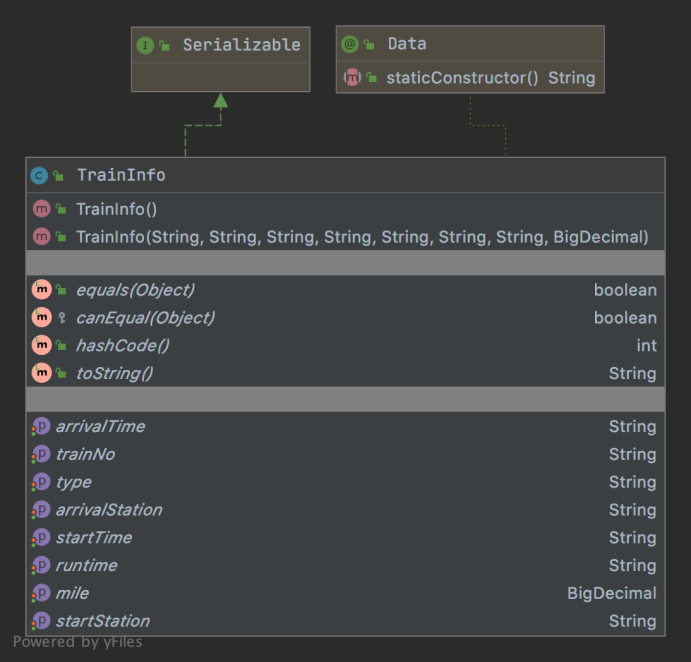


图 4 TrainInfo实体类

1. E-R图

以上表都是独立的实体，这些实体的 E-R 图设计为：

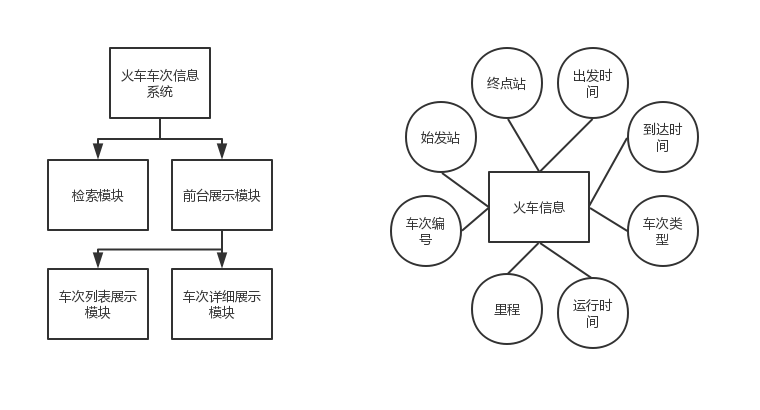


图 5实体E-R图

1. Service层

该层包含的主要是对 DAO 层的操作。这里提供了针对DAO层TrainInfo表的操作方法。根据以上实体之间的关系，设计出本层的UML类图。从图中我们可以清楚得看出类间依赖关系。

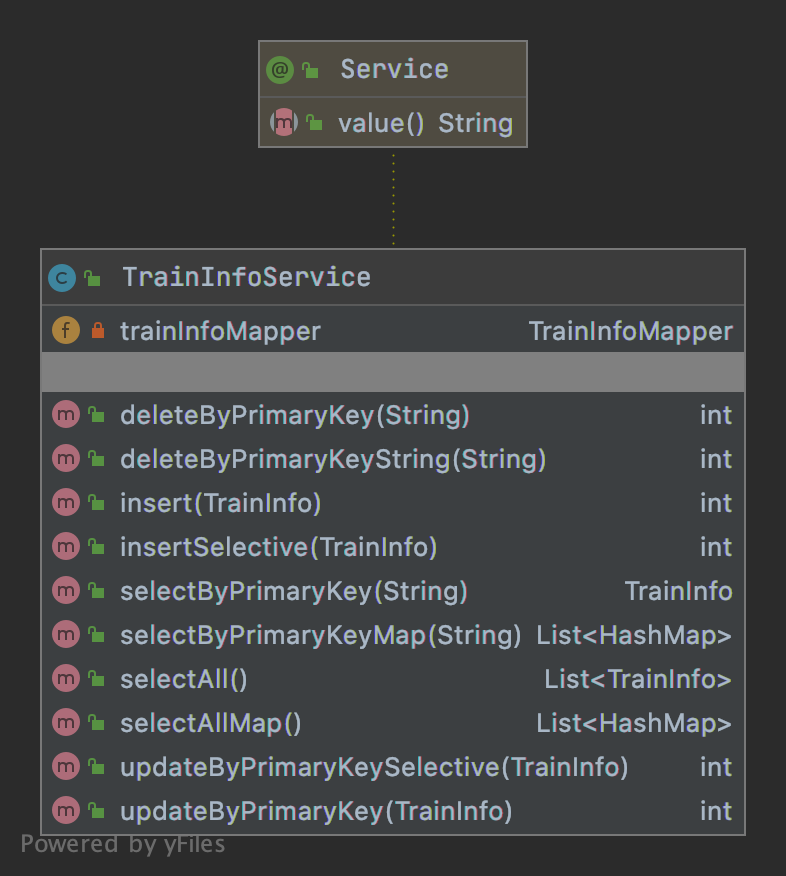


图 6 Service层E-R图

1. Controller 层
   1. IndexController

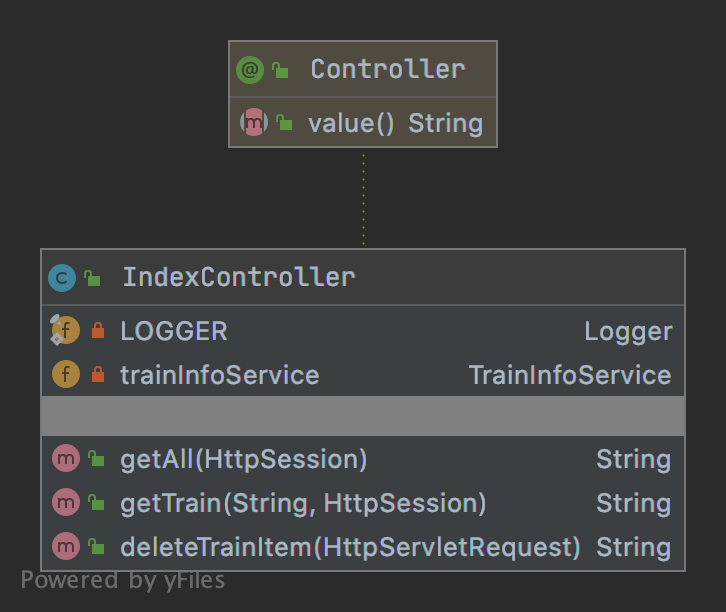


图 7IndexController.java UML类图

1. UML

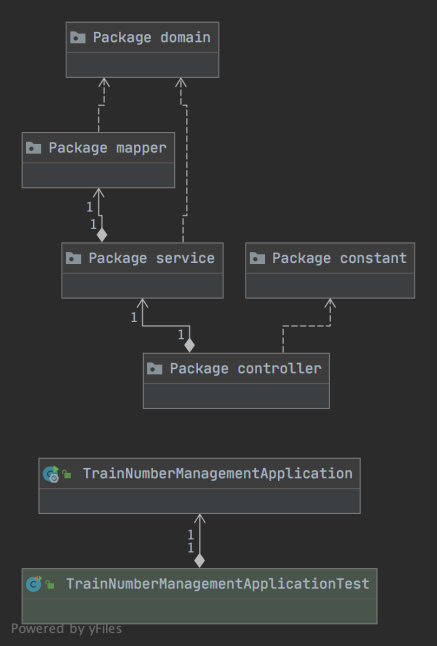


图 8程序包依赖图

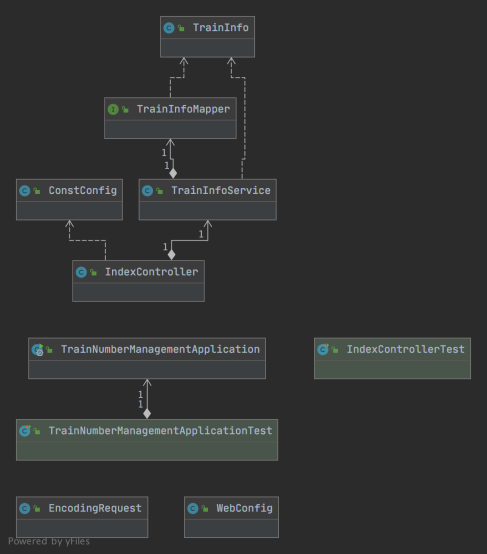


图 9 UML 类图

## 2、系统详细设计

### 前言

随着高铁在国内铺设的越来越广泛，高铁出行也逐渐成为人们的一种主要长途出行方式。在线火车车次查询系统在此环境下应运而生。火车车次查询是一项进行信息获取的软件，应用广泛，并且是实时性非常强的一个非常典型的管理信息系统。本系统正是基于这样一种环境下应运而生。本系统为前台部分，主要实现用户的网上列车车次业务流程、用户登录注册的功能。

本系统主要利用 Spring Boot + JSP 技术进行开发，有很强的逻辑性、可扩展性，便于维护。火车车次查询系统设计主要使用了CSS进行聊天界面的美化。其次本系统设计还使用了 Spring Boot、JSP 、JDBC 、JavaBean 、CSS、JavaScript、jQuery、AJAX等技术。

### 概要设计

1. 火车车次查询系统简要功能介绍

本次课程设计提供的简易火车车次查询系统是为需要长途出行且需要搭乘火车的用户开发设计的，和其他大型的列车车次查询系统相比该火车车次查询系统功能比较简单，但能实现基本的列车车次信息查询和删除功能。

1. 车次列表查询功能

功能描述：向浏览器端返回列车车次基本信息的列表；

1. 车次详细信息查询功能

功能描述：向浏览器端返回用户选定的列车车次详细信息；

1. 车次信息删除功能

功能描述：提供列车车次信息批量删除的能力。

### 详细设计

1. 服务端设计
   1. 应用级常量
      1. ConstConfig.java
         1. 前台回传/请求数据字段（input标签name属性名）名

|  |  |
| --- | --- |
| 常量名 | 常量值 |
| FIELD\_FRONT\_TRAIN\_LIST | "trainList" |
| FIELD\_FRONT\_TRAIN\_NO | "trainNo" |

* 1. Path resources
     1. MyBatis
        1. TrainInfoMapper.xml
           1. 配置文件路径：

src/main/resources/mybatis/mapper/TrainInfoMapper.xml

* + - * 1. 功能描述

此配置文件包含对MySQL数据库中train\_info表的SQL操纵语句。其返回结果均为java.util.Hashmap类型，这种数据类型能够帮助开发人员更加高效的进行程序逻辑开发，可以省略对实体类的依赖，降低程序耦合度。

* + 1. Spring Boot
       1. application.yaml
          1. 配置文件路径：

src/main/resources/application.yaml

* + - * 1. 功能描述

此配置文件包含对Spring Boot的相关配置，本系统在其中配置了JDBC、MyBatis、Server等属性信息。

* 1. Package Service
     1. TrainInfoService.java
        1. 程序包路径：

com.kevinkda.univ.curriculum.design.b202007.spring.boot.train.number.manager.service

* + - 1. 功能描述

此程序类依赖于TrainInfoMapper.java接口，它完成了对MyBatis Mapper的操纵功能，能帮助开发人员快速的扩展接口能力，完成对数据库的增删改查。

* 1. Package domain
     1. TrainInfo.java
        1. 程序包路径：

com.kevinkda.univ.curriculum.design.b202007.spring.boot.train.number.manager.domain

* + - 1. 功能描述

此程序类是火车车次信息的实体类，用于处理对火车车次信息实体的操作。

此程序类包含两个字段：trainNo（车次编号）、startStation（始发站）、arrivalStation（终点站）、startTime（出发时间）、arrivalTime（到达时间）、type（车次类型）、runtime（运行时间）、mile（里程）。

此程序类包含重写的toString()方法和两个构造方法，以及对每个字段的Getter & Setter方法。

* 1. Package controller
     1. IndexController.java
        1. 程序包路径：

com.kevinkda.univ.curriculum.design.b202007.spring.boot.train.number.manager.controller

* + - 1. 功能描述

此程序类3项方法：getAll()、getTrain()和deleteTrainItem()。

* + - * 1. getAll()

此方法向用户提供完整的列车车次基本信息。

* + - * 1. getTrain()

此方法向用户提供用户指定列车车次完整信息。

* + - * 1. deleteTrainItem()

此方法向用户提供批量删除用户指定的列车车次信息。

* 1. Main 程序入口 TrainNumberManagementApplication.java
     1. 程序包路径：

com.kevinkda.univ.curriculum.design.b202007.spring.boot.train.number.manager

* + 1. 功能描述

此程序类是本系统的入口程序。

* 1. Package com.kevinkda.core.util.web.support
     1. EncodingRequest.java
        1. 程序包路径：

com.kevinkda.core.util.web.support

* + - 1. 功能描述

此程序类继承自HttpServletRequestWrapper类，通过覆写父类方法以将Filter中的原有ServletRequest对象进行替换，得到新的HttpServletRequest对象。通过此方法可以对Request对象进行更多功能的扩充。

此程序类在此处的方法是将用户从前台提交的ISO-8859-1字符编码转换为UTF-8字符编码。

1. 前台页面设计
   1. JavaScript

本系统中使用jQuery技术完成JS的快速开发。其版本使用jQuery 1.12.4，本系统开发过程中采用离线依赖，文件存储于web/js/目录下。

* 1. JSP
     1. 根目录
        1. getAll.jsp

此页面是项目首页，此页面向用户提供完整的列车车次基本信息。

* + - 1. getTrain.jsp

此页面是进行查询用户指定列车车次完整信息的详细信息页。

# 四、系统的编码、测试和总结

## 1、系统代码

系统配置文件（application.yaml）代码：

server:

port: 8095

servlet:

context-path: /train

mybatis:

#type-aliases-package: com.chinasoft.springboot\_mybatis.pojo

mapper-locations: classpath:/mybatis/mapper/\*.xml

spring:

thymeleaf:

enabled: true #开启thymeleaf视图解析

encoding: utf-8 #编码

prefix: classpath:/templates/ #前缀

cache: false #是否使用缓存

mode: HTML #严格的HTML语法模式

suffix: .html #后缀名

resources:

static-locations: classpath:/META-INF/resources/,classpath:/resources/,classpath:/static/,classpath:/public/,classpath:/upload/

mvc:

static-path-pattern: /\*\*

view:

# prefix: /static/jsp/

prefix: /WEB-INF/jsp/

suffix: .jsp

datasource:

driver-class-name: com.mysql.cj.jdbc.Driver

url: jdbc:mysql://bj.kevinkda.cn:3306/TemplateData?characterEncoding=UTF-8&rewriteBatchedStatements=true

username: kevin

password: kevin

jdbc:

username: kevin

password: kevin

url: jdbc:mysql://bj.kevinkda.cn:3306/test?serverTimezone=UTC&useSSL=false&useUnicode=true&characterEncoding=utf8

driverClassName: com.mysql.cj.jdbc.Driver

Controller代码：

/\*\*

\* 访问路径控制器

\*

\* @author Kevin KDA on 2020/7/6 17:55

\* @version 1.0.0

\* @project curriculum-design-spring-boot-202007

\* @package com.kevinkda.univ.curriculum.design.b202007.spring.boot.train.number.manager.controller

\* @classname IndexController

\* @apiNote <p></p>

\* @since 1.0.0

\*/

@Controller

public class IndexController {

private static final Logger LOGGER = LoggerFactory.getLogger(IndexController.class);

/\*\*

\* 服务层实例

\*

\* @since 1.0.0

\*/

@Autowired

private TrainInfoService trainInfoService;

/\*\*

\* 项目首页访问路径，查看所有车次信息

\*

\* @param session 传入{@link HttpSession}，将程序处理得到的数据存入Session

\* @return java.lang.String 页面跳转路径

\* @author Kevin KDA on 2020/7/7 21:54

\* @description IndexController / getAll

\* @version 1.0.0

\* @apiNote <p></p>

\* @since 1.0.0

\*/

@RequestMapping({"/", "/index", "/getAll"})

public String getAll(HttpSession session) {

// 用户访问地址

LOGGER.debug("getAll.jsp");

// 获得所有火车车次信息

List<HashMap> trainList = trainInfoService.selectAllMap();

// 向前台回传数据

session.setAttribute(ConstConfig.FIELD\_FRONT\_TRAIN\_LIST, trainList);

// 日志：记录数据记录

LOGGER.info(trainList.toString());

// 跳转车次列表页

return "getAll";

}

/\*\*

\* 查看车次详细信息

\*

\* @param trainNo 传入请求的车次编号

\* @param session 传入{@link HttpSession}，将程序处理得到的数据存入Session

\* @return java.lang.String 页面跳转路径

\* @author Kevin KDA on 2020/7/7 21:56

\* @description IndexController / getTrain

\* @version 1.0.0

\* @apiNote <p></p>

\* @since 1.0.0

\*/

@RequestMapping({"/getTrain"})

public String getTrain(@RequestParam(ConstConfig.FIELD\_FRONT\_TRAIN\_NO) String trainNo,

HttpSession session) {

// 用户访问地址

LOGGER.debug("getTrain.jsp");

// 获得请求的火车车次信息

HashMap trainList = trainInfoService.selectByPrimaryKeyMap(trainNo).get(0);

// 向前台回传数据

session.setAttribute(ConstConfig.FIELD\_FRONT\_TRAIN\_LIST, trainList);

// 日志：记录数据记录

LOGGER.info(trainList.toString());

// 跳转车次详细页

return "getTrain";

}

/\*\*

\* 删除车次信息

\*

\* @param request 传入{@link HttpServletRequest}，获得用户传入的请求参数组

\* @return java.lang.String 页面跳转路径

\* @author Kevin KDA on 2020/7/7 22:00

\* @description IndexController / deleteTrainItem

\* @version 1.0.0

\* @apiNote <p></p>

\* @since 1.0.0

\*/

@RequestMapping({"/deleteTrainItem"})

public String deleteTrainItem(HttpServletRequest request) {

// 用户访问地址

LOGGER.debug("deleteTrainItem API");

// 获得需要进行删除的车次编号

String[] no = request.getParameterValues(ConstConfig.FIELD\_FRONT\_TRAIN\_NO);

// 核验前台传入参数是否为空，为空即结束操作

if (no != null && no.length != 0) {

// 日志：记录删除数据

LOGGER.info("delete trainNo: " + Arrays.toString(no));

// 循环遍历删除数据

for (String s :

no) {

trainInfoService.deleteByPrimaryKey(s);

}

}

// 重定向跳转至车次详细页

return "redirect:/getAll";

}

}

## 2、系统的测试

软件测试的目的和原则

在开发软件系统的过程中，我们需要面对错综复杂的问题，不可避免的在每个生产阶段都会产生一些错误，我们力求进行严格的技术审查在每个阶段任务结束前尽可能多的发现和纠正软件中的错误。而软件测试的目的就是在软件投入生产环境前尽可能的修正软件中的错误，目前软件测试仍然是软件质量控制的最后防线。

软件测试分以下三种类型：

1. 单元测试：单元测试的用例从单元详细设计中导出。在单元测试中可以采用功能性测试和结构性测试两种；
2. 集成测试：在这个测试步骤中所发现的往往是概要设计的错误；
3. 系统测试：在这个测试步骤中所发现的往往是需求规格说明的错误。一般来说，系统测试是功能性测试，不是结构性测试。

本次设计进行了如下调试和测试，并提出相应的解决方案：

1. 访问失败

检查浏览器访问的路径是否为 http://127.0.0.1:8095/train/ ；

1. 前端脚本失效

检查是不是有语法错误导致后面的脚本全被屏蔽了；

1. 数据库连接出错

检查 JDBC 版本；检査数据库服务监听是否成功开启；检查数据库 URL、用户名密码等是否正确；

1. 配置文件加载失败

为 XML 的配置文件有些是 XSD 验证的，所以可能需要联网下载验证规则，此时应该保证网络畅通。其他就是可能配置文件书写的时候出现了语法错误；

1. Jdbctype 和 javatype 以及数据库 Type 问题

需要查询这三种 type 的对应关系，印象最深的就是不同数据库和 jdbctype 对应关系不一样。

# 五、总结

这次 Spring Boot课程设计是一门非常不错的课程，在此之前我写服务器基本都是自己造轮子自己写，但是这次课设使得我有机会使用现在还在使用Spring Boot + JSP架构进行开发，同时，这次实验对我来说是一个巨大的挑战。因为之前只是对 Spring Boot的快捷开发有所耳闻，原先用 Servlet的时候也是自己从GET和 POST处理函数为入口自己写处理，所以一开始对于 Spring Boot的写法还不是很熟练。但是，经过持续约一个月坚持不懈的努力，我终于完成了本次的实验任务。

当然在开发的过程中也遇到了很多问题。老师曾说，对于我们现在接触到的困难，之前肯定有人接触过，所以要好好利用互联网。课程设计，从安装 MySQL，到配置数据库，到使用JDBC进行数据的增删改查，每步都曾遇到很棘手的问题，多亏了老师的帮助，也感谢在社区和论坛上帮我解决问题的朋友们。

采用B/S结构与JDBC框架。动态 WEB 中，程序依然使用客户端和服务端，客户端依然使用浏览器（Chrome、Safari、Firefox 等），通过网络（Network）连接到服务器上，使用 HTTP 协议发起请求（Request），现在的所有请求都先经过一个 WEB Server Plugin（服务器插件）来处理，此插件用于区分是请求的是静态资源（\*.htm 或者是 \*.html）还是动态资源。

当然，这次实验的另外一个体会就是，时间安排非常重要。老师在很早的时候就把任务书发了下来，但是当时我没在意，所以在之后做得非常紧。由此了解到，做好一个项目，每一阶段的时间都必须规划好。因为我们学生现阶段采用开发模式基本都是瀑布开发模式，就必须保证每一步的正确性和及时性。总之，本次设计锻炼了实践能力，増长了我发现问题解决问题的能力，大大丰富了自己的专业知识和技能。由于各种原因，系统的功能还没有得到进一步的完善，以后会继续努力，在今后的学习过程中，不断地提高自己。