

****

**本科毕业论文**

|  |
| --- |
| **基于Restful方式的在线学生信息管理系统** |

**全程**

**201231000826**

|  |  |
| --- | --- |
| 指导教师 | **张义青 讲师** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 学院名称 |  | **数学与信息学院** | 专业名称 |  | **软件工程** |
| 论文提交日期 |  |  | 论文答辩日期 |  |  |

答辩委员会主席 \_\_\_\_\_\_\_\_\_\_\_\_

评 阅 人 \_\_\_\_\_\_\_\_\_\_\_\_

摘 要

关键词：.NET MVC AJAX 停车场

目 录

目录

[1. 引言 3](#_Toc446973150)

[1.1. 选题背景 3](#_Toc446973151)

[1.2. 论文组织 3](#_Toc446973152)

[2. 相关技术与工具介绍 3](#_Toc446973153)

[2.1. REST 3](#_Toc446973154)

[2.2. SpringMVC 4](#_Toc446973155)

[2.3. HttpClient 4](#_Toc446973156)

[2.4. C/S模式 4](#_Toc446973157)

[2.5. fastjson 5](#_Toc446973158)

[3. 基于Restful方式的在线学生信息管理系统总体设计 5](#_Toc446973159)

[3.1. 系统概述 5](#_Toc446973160)

[3.2. 系统设计思想 5](#_Toc446973161)

[3.3. 系统功能模块设计 5](#_Toc446973162)

[3.3.1. 客户端： 5](#_Toc446973163)

[3.3.2. 服务器端： 6](#_Toc446973164)

[3.4. 数据库设计 6](#_Toc446973165)

[3.5. 系统架构设计 6](#_Toc446973166)

[4. 系统实现 8](#_Toc446973167)

[4.1. 后台资源服务实现 8](#_Toc446973168)

[4.1.1. SpringMVC和Hibernate搭建Service框架 8](#_Toc446973169)

[4.1.2. HTTP标准方法和JSON实现数据传输 9](#_Toc446973170)

[4.2. 数据管理软件实现 9](#_Toc446973171)

[4.2.1. 可视化界面实现 9](#_Toc446973172)

[4.2.2. 远程数据库数据管理 9](#_Toc446973173)

[4.3. 安卓客户端实现 10](#_Toc446973174)

[4.3.1. 可视化界面实现 10](#_Toc446973175)

[4.3.2. 数据获取及保存实现 10](#_Toc446973176)

[5. 系统运行环境 10](#_Toc446973177)

[5.1. 基于Restful方式的在线学生信息管理系统运行与评估 10](#_Toc446973178)

[5.2. 数据管理软件运行与评估 10](#_Toc446973179)

[5.3. 安卓客户端运行与评估 10](#_Toc446973180)

[6. 总结与展望 10](#_Toc446973181)

[6.1. 总结 10](#_Toc446973182)

[6.2. 展望 10](#_Toc446973183)

[参 考 文 献 10](#_Toc446973184)

[致 谢 10](#_Toc446973185)

# 引言

## 选题背景

随着社会的发展，用手机查询学生个人信息的需求不断地增强，对于现在的情况，一些学校的教务网站开始将信息查询的功能转移到Android系列手机的客户端上。拥有了这种功能的Android软件使学生能够方便地查询到学生的个人信息。

从学生角度分析，学生查询校方公告、考试成绩时往往需要使用个人电脑来完成，这会造成信息无法即时获取，也可能因为服务器端出现问题导致无法查询。从校方角度分析，发现部分信息无法及时到达学生手中，同时己方服务器由于学生查询成绩等原因奔溃。

如今安卓系统技术已相当成熟，可以完美地完成信息交流过程中的运算和存储，缓解服务器端压力，同时其高普及率保证大部分学生能够及时获取相关信息。

本课题的研究意义在于通过以上提出智能手机接收信息、信息查询即时话等思路，设计并实现一个可行的在线学生信息管理系统，缓解学生不能即时获取校方信息的问题。

## 论文组织

本文将用六个章节阐述、设计并实现系统的过程、思想、技术，各章如下内容：

第一章：前言，介绍系统设计与实现的背景，展现文章的组织结构。

第二章：使用工具和关键技术介绍，对在系统设计和实现过程中使用到工具以及关键技术进行描述和分析。

第三章：系统设计与实现，对系统进行模块功能设计，数据库设计，系统架构设计，分析实现系统总体的技术要点与过程。

第四章：系统实现的功能的关键技术思路与实现，关键代码分析。

第五章：系统最终运行效果展示，图文展示验收最终系统实现运行起来的效果。

第六章：总结与展望。

# 相关技术与工具介绍

## REST

Representational State Transfer，是一种针对网络应用的设计风格。REST指定了一组架构约束条件和原则，满足这些要求的系统可以称为RESTful。

REST主要原则包括：对资源使用一致的命名规则，即使用URI标识所有需要标识的事物，特别是应用中需要使用到的资源；链接所有资源事物，系统中的资源都可以使用特定的URI获取；使用HTTP标准方法，使客户端程序能与系统资源相互协作，使用POST、DELETE、PUT、GET完成数据管理中的增删改查；资源多重表述，为同一资源提供多种表述形式，提高系统重用性和可扩展性；无状态通信，通信状态应该由客户端负责维护，服务端应该返回无状态的资源信息。

RESTful架构约束主要有使用CS模式，通信只由客户端发起，服务端表现为根据请求进行响应的模式；内容缓存，响应内容或者资源可以在通信链进行缓存，可以改善网络效率，在本系统的表现为数据可以在安卓手机离线保存，同时利用安卓手机减轻服务器运算负担；分层系统，通过规定组件的行为，使其只能接受与其交互的紧邻层的信息，达到将服务分解为若干层的效果；以及统一接口、无状态和按需编码。

## SpringMVC

Spring MVC属于SpringFrameWork的后续产品，提供了构建 Web 应用程序的全功能 MVC 模块。框架主要由DispatcherServlet、HandlerMapping、Controller、ViewResoler、视图组成。



组件细节：DispatcherServlet是SpringMVC的核心，其本质是一个HttpServlet，通过配置文件中的<servlet-mapping>完成对request的监听。当DispatcherServlet接收到请求时，会根据HandlerMapping的内容，调用Controller对请求进行处理。在本系统的表现形式为，系统接收到资源请求，会调用对应的Controller返回相关资源。

## HttpClient

HttpClient 是一个可以用来提供高效的、最新的、功能丰富的支持 HTTP 协议的客户端编程工具包，并且支持 HTTP 协议最新的版本和建议。HttpClient实现了所有HTTP的方法，可以方便地配合REST架构约束中的使用HTTP标准方法的要求，同时支持HTTPS协议。使用HttpClient需要以下步骤：首先创建HttpClient实例，接着创建某种连接方法的实例，这里包括HttpGet、HttpPut、HttpPost、HttpDelete等，HttpClient中的execute方法可以调用以上实例，完成HTTP中的方法。接着读取HTTP方法的response，释放连接，最后对得到的内容进行处理。

HttpClient同时支持多种运用场景，同时能够加快系统效率以及开发效率，本系统中的数据管理软件以及安卓客户端均使用HttpClient完成HTTP请求。

## C/S模式

Client-Server是为了配合REST技术以及安卓开发所选用的系统结构。通过它可以充分利用两端硬件环境的优势，将任务合理分配到Client端和Server端来实现，降低了系统的通讯开销。根据功能分布的原则，客户端完成数据处理，数据表示以及用户接口功能；服务器端完成数据库管理系统的核心功能。

C/S结构优点为充分发挥客户端处理能力，将部分业务分配到客户端进行处理，客户端可以存储部分数据，保证系统响应速度，同时减轻网络负担。其缺点

## fastjson

# 基于Restful方式的在线学生信息管理系统总体设计

## 系统概述

本系统设计成一个可以让学生查询校方信息的平台，可以根据自身学号获取本人课程的相关信息，当前类似四六级等级考试的专业技能等级考试信息以及校方发布的公告。而校方可以远程管理托管在阿里云的学生数据，进行管理学生账号、发布公告和学生成绩等操作。

## 系统设计思想

根据REST技术的特点，本系统采取版本迭代的方式，不断推出新版本，在系统框架确定的前提下不断为系统添加新功能，让系统功能逐步完善。

## 系统功能模块设计

通过本系统，在移动端实现学生的登入登出、个人信息以及学院公告的查询，在SERVICE端实现学院信息、学生个人信息以及课程信息的管理，并通过数据库保证数据同步。图1为本系统功能模块图。



图1 本系统功能模块图

### 客户端：

客户端功能如下：

登入：学生根据系统中预设的账户进行登入操作，可以根据账户查看不同学生的信息。

登出：已登入账户可以进行登出操作，同时删除相关的学生信息。

更新相关课程信息：根据登入的学生账户，可以更新并离线保存该学生当前的课程信息、成绩等。

更新学院发布的公告：已登入的学生可以获取学院发布的公告，相同学院的学生会收到同一份公告。

更新技能等级考试信息：根据登入的学生账户，可以更新并离线保存该学生当前的技能等级考试信息。

数据显示：学生在已登入的情况下可以直接查看离线保存的数据。

### 服务器端：

学生账户管理：可以管理学生账户信息，包括密码、学院分组等。

课程管理：可以进行建立课程，更新考试成绩等一系列操作。

学院数据管理：可以对学校当前学院及其下属教师的信息进行数据管理。

技能等级考试管理：可以根据已有的学生账户，管理个人技能等级考试信息。

## 数据库设计

本系统使用MySQL5.6以及SQLite数据库，设计的数据库有6张表，其中包括notice表、academy表、teacher表、student表、course表、technologicalexam表。

其中，表与表之间关系如下所述：

academy与notice是一对多关系。

academy与teacher是一对多关系。

academy与student是一对多关系。

student与technologicalexam是一对多关系。

student与course是一对多关系。

teacher与course是一对多关系。

主要的表设计与关系如图2所示：



图2基于Restful方式的在线学生信息管理系统数据库设计

## 系统架构设计

为配合REST核心技术以及Android系统特性，本系统采用C/S结构，资源接口编程设计、MVC架构，提高了系统的重用、可扩展性、并发性、运行速率。图3为本系统的架构设计:



图3基于Restful方式的在线学生信息管理系统架构设计

后台采用MVC思想，用Hibernate技术奖表映射出Entities，将常用的数据库操作代码设计为Data Access Object，用SpringMVC将具体业务逻辑运算的代码封装在Service层，Resource Interface层暴露出REST的API，使其他程序能使用HTTP标准方法读取和管理MySQL中的数据。前端分别使用Android开发技术和Intellij IDEA集成的JAVA swing开发技术实现一个适用对象为学生的Client端和个人电脑端的ServiceGUI端，能够遵循RESTful标准管理远程后台的相关信息。

根据RESTful标准，后台搭建HTTP标准方法的资源接口，即使用HttpClient技术实现GET方法获取数据，DELETE方法删除数据，POST方法新增数据，PUT方法更新数据的ＨＴＴＰ标准方法。数据传输方面，本系统使用Alibaba的fastjson技术完成数据的封装，用Google的gson技术完成解读，同时使用URLEncoder和URLDecoder实现跨平台数据传输。这些技术保证了系统的重用性，可扩展性和足够高的运行效率。

安卓Client端使用Google官方Android Design Support Library中的Tablayout和Viewpage完成可视化组件，构建出类似新浪微博手机版的界面，可以划屏切换学院公告、课程信息和技能等级考试的不同信息，同时使用HttpClient发送请求和接受Json数据。接收到数据后，使用SQLite保存数据，使学生能够离线查看自身信息。数据以远程数据库为标准，每次更新信息都会获取远程数据库的相关数据保存到本地。

服务器GUI的界面部分使用Intellij IDEA集成的Java Swing开发工具，使用xml文件对Java可视化组件进行组建和一些Windows平面风格的改进，提高了Java可视化组件的用户体验。数据传输部分使用即时更新的方式管理远程数据库数据，不保留本地数据。即每次修改保存都会直接更新到数据库，以数据库为基准值保持数据同步。

在实际应用场景中，ServiceGUI不会直接影响Android Client，校方只能通过ServiceGUI更新信息到远程数据库，学生使用安卓手机获取远程数据库当前的信息。

# 系统实现

## 后台资源服务实现

### SpringMVC和Hibernate搭建Service框架



图4基于Restful方式的在线学生信息管理系统后台框架

如图4所示，后台使用SpringMVC和hibernate搭建从MySQL数据库到Resource Interface的框架。首先使用hibernate生成集成了JDBC的DAO和与MySQL相匹配的Entities，这使得系统能从代码角度修改MySQL的数据。其次，使用SpringMVC管理系统内的组件，即使用Spring注解实现以springframework中的工厂为管理器，DAO、Entities、Service和Resource Interface为工厂中的bean为被管理对象，自动实现beans的生成与使用。Service和Resource Interface的实现使用了Spring工厂生成的beans，其中Service处理具体的业务逻辑计算，Resource Interface用于暴露RESTful的数据处理接口。当系统监听到外部通过Resource Interface发送请求时，会调用Service进行相关操作，Service则调用DAO进行具体操作或者获取相关数据并通过Resource Interface返回到外部。

### HTTP标准方法和JSON实现数据传输



根据REST技术，Service端对资源使用一致的命名规则，外部通过HTTP标准方法调用URL，即可获得对应资源。本系统Service端挂载在阿里云服务器中，资源服务器URL为http://115.28.167.136:8080/RESTSIQS/\*，个人电脑及安卓手机会根据HTTP标准方法向服务器发送请求，若使用GET方法则会返回服务器当前数据，若使用DELETE、POST和PUT方法则会返回数据更新后的新数据。

## 数据管理软件实现

### 可视化界面实现

数据管理软件是原生JAVA组件构建的JAR可运行程序。不同的是可视化界面部分由Intellij IDEA集成的工具完成。系统每个窗口都是Intellij IDEA定义的GUI form，每个GUI form由一个java文件和一个form文件组成，并由form文件决定组件布局，java文件完成业务逻辑计算。

具体实现过程为：使用MainView.form完成数据管理软件中AcademyPanel、CoursePanel、NoticePanel和StudentPanel的布局，在自定义四个Panel的具体业务逻辑，搭配相关数据的管理Dialogs完成整个可视化界面。AcademyPanel中以双重列表完成数据显示，左侧为学校当前的学院列表，右侧为被选中学院中的教师列表，通过双击等操作即可管理远程数据库中的数据。CoursePanel为关联教师与学生之间课程信息的界面，会以多对多的形式显示两者之间的关联。StudentPanel显示当前学生账号信息，双击一个学生账号时可以管理该账号对应的基本信息、技能等级考试信息和该账号对应的课程信息。NoticePanel为学院查看和管理当前公告的界面，不同的学院发布的公告会被对应学院的学生接收到。

### 远程数据库数据管理

实际应用场景中，用户在程序界面完成数据更新后，会即时保存在远程数据库中并更新本地程序的数据。本系统数据管理软件会向Service发送与Resource Interface相对应的请求，并获取新的数据，即在使用GET以外的方法后，程序都会调用一次GET方法获取新的数据。

以更新数据为例，用户更新完某段数据后，程序首先将新数据封装成一个pojo对象，其后转化为JSON数据，通过HttpClient中的HttpPut将请求发送到数据库，而后程序会调用一次HttpGET方法获得新的数据，并显示在可视化界面中。

public static void put(String URL, Object object) throws ConnectException {  
 try {  
 String jsonObject = JSONObject.*toJSONString*(object);  
 jsonObject = URLAccepter.*encrypt*(jsonObject);HttpClient httpclient = new DefaultHttpClient();  
 HttpPut request = new HttpPut(URL + jsonObject);  
 httpclient.execute(request);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }

## 安卓客户端实现

### 可视化界面实现

### 数据获取及保存实现

# 系统运行环境

## 基于Restful方式的在线学生信息管理系统运行与评估

## 数据管理软件运行与评估

## 安卓客户端运行与评估

# 总结与展望

## 总结

## 展望

参 考 文 献

致 谢