项目需求规格分析说明书

# 一、引言

## 1.1 编写目的

本软件需求说明书旨在分析学生信息管理系统的基本信息，并了解系统中各功能及性能等方面的要求，使开发人员能尽快掌握并及时开发。该系统的使用者为管理员，并完成相关的基本操作。

## 1.2 背景

### 1.2.1 背景概述

学生信息管理系统，是针对学校人事处的大量业务处理工作而开发的管理软件，是典型的管理信息系统。它是一个教育单位不可缺少的部分,它的内容对于学校的决策者和管理者来说都至关重要,它能够为用户提供充足的信息和快捷的查询手段。能有效的帮助学校和老师掌握学生的情况，为学生提供成绩跟课程方面的查询。在传统模式下利用人工进行学生信息管理，存在着较多的缺点，如：效率底，保密性差，时间一长将产生大量的文件和数据，更不便于查找，更新，维护等。诸如这些情况，令学校管理者对学生的信息管理带来了很大困难，严重影响了教育工作者的工作效率。随着科学技术的不断提高,计算机科学日渐成熟, 使用日趋成熟的计算机技术来代替传统的人工模式，来实现学生信息的现代化管理，其强大的功能已为人们深刻认识,它已进入人类社会的各个领域并发挥着越来越重要的作用。作为计算机应用的一部分,使用计算机对学生信息进行管理,具有着手工管理所无法比拟的优点.例如:检索迅速、查找方便、易修改、可靠性高、存储量大、数据处理快捷、保密性好、寿命长、成本低、便于打印等。这些优点能够极大地提高学生信息管理的效率,也是学校的科学化、正规化管理,与世界接轨的重要条件。用计算机制作的学生信息管理系统还可以通过功能强大的Internet网及时的向学生的家长传递该生在校的最新成绩，有助于学校与家长的信息互动，为更好地把握学校的教育方针一定的促进作用！因此，开发这样一套管理软件成为很有必要的事情。

### 1.2.2 系统名称

学生信息管理系统

### 1.2.3 任务提出者

上海电力学院计算机科学与技术学院

### 1.2.4 任务开发者

凤翔、陈龙龙

### 1.2.5 项目使用者

管理员

### 1.2.6 项目与其它软件、系统的关系

系统：Windows7

数据库：MongoDB

IDE：Webstorm8

版本控制软件：Git  
编辑器：Sublime Text 3

Google查资料用代理工具：Goagent

## 1.3定义

学生信息管理系统（Student Information Management System）：为用户提供充足的信息和快捷的查询手段。能有效的帮助学校和老师掌握学生的情况，为学生提供成绩跟课程方面的查询。

Node.js： 是一个基于Chrome JavaScript 运行时建立的一个平台， 用来方便地搭建快速的， 易于扩展的网络应用· Node.js 借助[事件驱动](http://baike.baidu.com/view/536048.htm" \t "_blank)， 非阻塞[I/O](http://baike.baidu.com/subview/300881/11169495.htm" \t "_blank) 模型变得轻量和高效， 非常适合 运行在分布式设备 的 数据密集型 的实时应用

MongoDB：一个基于分布式文件存储的数据库。由C++语言编写。旨在为WEB应用提供可扩展的高性能数据存储解决方案。

CoffeeScript：CoffeeScript是一套JavaScript的转译语言。受到Ruby、Python与Haskell等语言的启发，CoffeeScript增强了JavaScript的简洁性与可读性。此外，CoffeeScript也新增了更复杂的功能，例如列表内涵（List comprehension）、模式匹配（Pattern matching）等。

## 1.4参考资料

[Node.js v0.10.29 Manual & Documentation](http://nodejs.org/api/)

[The MongoDB 2.6 Manual](http://docs.mongodb.org/manual/)

[Mongoose Schemas v3.8.12](http://mongoosejs.com/docs/guide.html)

[NODE.JS UNIT TESTING BY @朴灵](http://html5ify.com/unittesting/slides/index.html#/)

[Express 4.x API Reference](http://expressjs.com/4x/api.html)

[How to use templates with EJS]( https://code.google.com/p/embeddedjavascript/wiki/Templates)

[Bootstrap 中文文档](http://v3.bootcss.com/)

[LESS 语法](http://www.bootcss.com/p/lesscss/)

[Pro Git](http://git-scm.com/book/zh/)

[初识 mocha in NodeJS](http://cnodejs.org/topic/516526766d38277306c7d277)

[Mocha - the fun, simple, flexible JavaScript test framework](http://visionmedia.github.io/mocha/)

[Wikipedia](http://zh.wikipedia.org/)

[Stackoverflow](http://stackoverflow.com/)

# 二、任务概述

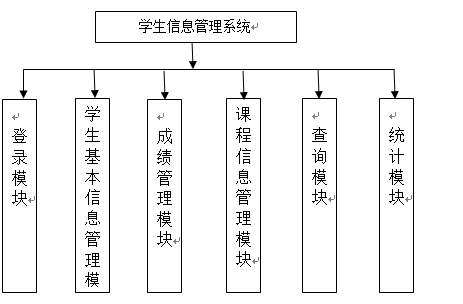
## 2.1 目标

### 2.1.1 开发目标及作用范围

对于开发的学生信息管理系统，主要实现以下功能：对学生的基本信息、课程安排、成绩、选课、密码，进行基本的管理。包括：学生的基本信息和学科成绩的存储、输入、删除、修改、查询。老师可以录入成绩、查询学生课程信息及修改密码、编辑学生的信息。管理员：课程管理、学生管理、修改密码及查询。

### 2.1.2 系统基本情况及模块介绍

本学生信息管理系统共包含6个模块，分别为登录模块、学生基本信息管理模块、成绩管理模块、课程信息管理模块、查询模块、统计模块，模块结构图如下图所示。



每一个模块都有其固有的功能，下面我将为大家一一介绍每个模块的基本功能，介绍如下：

（1）学生基本信息管理模块：对学生的基本信息进行综合管理，可以添加、修改及删除学生的基本信息。

（2）成绩管理模块：对学生所选课程的成绩信息进行综合管理，可以添加、修改及删除基本信息。

（3）课程信息管理模块：对课程信息进行综合管理，可以添加、修改及删除课程的基本信息。

（4）查询模块：

1）学生基本信息的查询：根据学生的已知条件来查询学生的详细信息，对姓名、学号、班级、系名等支持模糊查询。

2）课程基本信息的查询：根据课程的信息来查询课程的详细信息。

3）查询学生的选课情况、查询学生所选课程的成绩。

（5）统计模块：根据不同课程对学生成绩进行统计，求平均分、总分等；根据不同的分数区间进行人数统计等。

## 2.2 用户特点

### 2.2.1 用户范围

本系统分为3种用户，分别为管理员、教师和学生，每一种用户的功能各不相同，下面我将从用户的角度出发为大家介绍用户基本功能情况。

### 2.2.2 用户功能介绍

* 管理员



## 2.3 假定和约束

本系统从开始到结束预计花费九天时间，以下是本系统每天的预期工程进度表。

|  |  |
| --- | --- |
| 时间 | 预期工程进度 |
| 第一天 | 可行性分析、需求分析 |
| 第二天 | 数据库设计 |
| 第三天 | 详细设计 |
| 第四天 | 编码 |
| 第五天 | 编码 |
| 第六天 | 编码 |
| 第七天 | 测试与运维 |
| 第八天 | 测试与运维 |
| 第九天 | 撰写总结报告 |

由上表可知，本系统的实现受时间约束的影响可能完成并不是很完善，但是除时间因素的影响，例如经费等因素均不在约束考虑范围之内。

# 三、需求规定

## 3.1 对功能的规定

每一个模块中的用例均有其特有的功能描述，因时间有限，下面我将以学生基本信息管理模块为例，为大家进行用例分析。

|  |  |
| --- | --- |
| 用例名称 | 添加学生信息 |
| 用例编号 | 1 |
| 执行者 | 管理员 |
| 用例简述 | 该用例描述了管理员如何为系统添加学生的过程 |
| 前置条件 | 1. 管理员登录系统成功 2. 输入正确的学生信息 |
| 后置条件 | 学生添加成功 |
| 基本流程 | 1. 管理员登录 2. 系统显示登录后首页 3. 管理员点击【学生信息维护】按钮 4. 系统显示学生列表页面 5. 单击【添加】按钮，系统显示学生添加界面 6. 输入正确的学生信息后，点击【确认】 7. 系统提示学生添加成功 8. 系统返回学生列表界面 |
| 替代流程 | 任何时刻发生以下状况，添加用户信息用例将失败：   1. 管理员未输入其中任一一项学生信息 2. 系统提示“XXX信息未输入，请返回重新输入” 3. 管理员输入错误的用户信息 4. 系统提示“XXX信息输入有误，请重新输入” |

|  |  |
| --- | --- |
| 用例名称 | 查看学生信息 |
| 用例编号 | 2 |
| 执行者 | 管理员 |
| 用例简述 | 该用例描述了管理员如何查看学生信息的过程 |
| 前置条件 | 1. 管理员登录系统成功 |
| 后置条件 | 学生信息查看成功 |
| 基本流程 | 1. 管理员登录 2. 系统显示登录后首页 3. 管理员点击【学生信息维护】按钮 4. 系统显示学生列表页面 5. 单击任意一名学生，系统显示学生信息界面 |
| 替代流程 | 任何时刻发生以下状况，查看学生信息用例将失败：   1. 管理员点击错误的按钮 2. 系统停留在原界面不跳转 3. 系统跳转至其它页面 |

|  |  |
| --- | --- |
| 用例名称 | 修改学生信息 |
| 用例编号 | 3 |
| 执行者 | 管理员 |
| 用例简述 | 该用例描述了管理员如何为修改学生信息的过程 |
| 前置条件 | 1. 管理员登录系统成功 2. 输入正确的学生信息 |
| 后置条件 | 学生信息修改成功 |
| 基本流程 | 1. 管理员登录 2. 系统显示登录后首页 3. 管理员点击【学生信息维护】按钮 4. 系统显示学生列表页面 5. 单击任意一名学生，系统显示该学生信息界面 6. 单击【编辑】按钮，系统进入编辑界面 7. 输入正确的学生信息后，点击【确认】 8. 系统提示学生信息修改成功 9. 系统返回学生列表界面 |
| 替代流程 | 任何时刻发生以下状况，修改学生信息用例将失败：   1. 管理员输入错误的学生信息 2. 系统提示“XXX信息输入有误，请重新输入” |

|  |  |
| --- | --- |
| 用例名称 | 删除学生 |
| 用例编号 | 5 |
| 执行者 | 管理员 |
| 用例简述 | 该用例描述了管理员如何删除学生的过程 |
| 前置条件 | 1. 管理员登录系统成功 |
| 后置条件 | 学生删除成功 |
| 基本流程 | 1. 管理员登录 2. 系统显示登录后首页 3. 管理员点击【学生信息维护】按钮 4. 系统显示学生列表页面 5. 单击任意一名学生后的【删除】按钮 6. 系统提示学生删除成功 7. 系统返回学生列表界面 |
| 替代流程 | 无 |

## 3.2 对性能的规定

从实用、好用的角度出发开发学生信息管理系统，在设计过程中主要考虑以下原则。

### 3.2.1 可操作性

原始信息皆由管理员录入，系统应尽量减少管理员的数据录入量，录入数据尽量通过设计下拉列表框来选择录入，这样的处理同时也避免了许多录入异常现象的发生。数据输入的格式应符合业务习惯，并且直观、方便。要求系统处理的数据能准确无误，同时输出信息要求直观、简洁。

### 3.2.2 可靠性

系统运行具有较高的可靠性，提供严格的并发控制，确保数据的一致性和正确性。

### 3.2.3实用性

从用户的实际需要出发进行系统开发，不盲目追求高新技术的应用。

### 3.2.4 安全性

系统安全措施可靠、高效、可维护性好，有权限控制，其中口令录入界面便于系统识别登录用户。

### 3.2.5 可维护性

为了保证系统的可维护性，要求具有详细的文档资料，同时，要求系统在功能设计上考虑可扩展性，以满足业务变动的需求。

### 3.2.6 可移植性

系统开发完成后，要能运行于任何由Window操作系统所构成的计算机网络环境下。

# 四、系统开发环境分析

## 4.1 系统开发技术

本系统全部使用开源技术，可以在Windows、OS X、Linux全平台通用。  
服务器环境：node.js 0.10.29  
服务器语言：CoffeeScript  
MVC框架：express 4  
数据库：MongoDB  
ORM框架：mongoose  
模板引擎：ejs  
单元测试框架：Mocha  
前端页面语言：HTML5  
层叠样式表：CSS3、LESS  
页面脚本语言：Javascript、CoffeeScript  
页面开源框架：JQuary、Bootstrap  
版本控制软件：Git  
版本控制服务：Github  
数据库托管平台：Mongohq  
IDE：JetBrain Webstorm  
编辑器：Sublime Text 3  
Google查资料用代理工具：Goagent