**“任务管理系统”技术文档**

**目录**

[1 系统说明 3](#_Toc528084928)

[2 实现方法 3](#_Toc528084929)

[2.1 前端 3](#_Toc528084930)

[（1）系统注册页面sign\_up.html 3](#_Toc528084931)

[（2）系统登录页面sign\_in.html 4](#_Toc528084932)

[（3）所有项目展示页面project.html 5](#_Toc528084933)

[（4）创建项目界面projectCreate.html 5](#_Toc528084934)

[（5）项目展示页面projectDetails.html 6](#_Toc528084935)

[（6）创建任务界面createTask.html 7](#_Toc528084936)

[（7）查看任务界面taskDetails.html 9](#_Toc528084937)

[（8）编辑任务界面updateTask.html 9](#_Toc528084938)

[（9）个人任务查看界面userInfo.html 9](#_Toc528084939)

[（10）项目成员展示界面projectTeam.html 10](#_Toc528084940)

[（11）项目成员编辑页面projectTeamUpdate.html 10](#_Toc528084941)

[2.2 前后端交互 10](#_Toc528084942)

[2.3 后台 11](#_Toc528084943)

[2.3.1 数据库设计 11](#_Toc528084944)

[2.3.2 后台架构 12](#_Toc528084945)

[2.3.3 后台启动 15](#_Toc528084946)

[3 文件夹说明 15](#_Toc528084947)

[4 各部分负责人 16](#_Toc528084948)

# 1 系统说明

本系统实现了一个任务管理系统，实现了对项目、任务、用户的增删改查的基本操作。前端使用Bootstrap框架，后台使用Spring Boot + MongoDB，实现了基本的前后端交互。

# 2 实现方法

## 2.1 前端

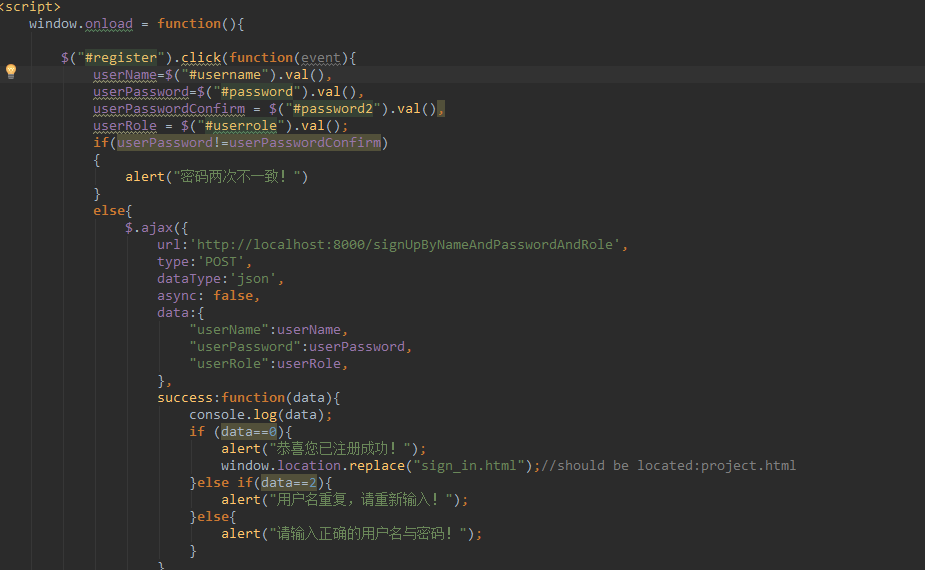
前端：使用ajax，传递json对象。数据交互格式严格参照如下的jQuery ajax语法：



### （1）系统注册页面sign\_up.html

对应js：页面内（规范起见应当单独抽出为sign\_up.js）

传参过程：页面使用ajax传值。首先获取四个输入框的值，在前端就判断两次密码是否输入的一致，不一致将弹出提示框。使用ajax传值，url如下，与后台提供的接口一致，使用post方式传输，传递userName，userPassword，userRole三个参数。如果注册成功，则跳转到登录页面，若注册的用户名在数据库已存在，则提示用户重新注册。



### （2）系统登录页面sign\_in.html

对应js：index.js（规范起见应当改为sign\_in.js与html命名一致）

传参过程：页面使用ajax传值。首先获取两个输入框的数据，然后给登录button添加点击事件。使用ajax将两个输入框的数据传给后台，这里的url是为了统一格式使用变量获取，实际上为http://localhost:8000/ signInByNameAndPassword。传输方式为post。若成功，则跳转到project.html，若失败则显示不正确。为了后面页面的使用，将userName值存在session里。



### （3）所有项目展示页面project.html

对应js：zjc.js

传参过程：传参函数getAllProjectName以POST方式向urllist.allProjectName.projectName发起请求，参数为string类型的userName，后台返回如下所示的json对象，前端接收后对json对象进行解析，显示出用户所有项目的名称、起止时间以及项目所有人。

{

\_id: "5bcd259ea482cc1d6837abc7"

​​ projectDesp: "efdqwq"

​​ projectEndtime: "20181022"

​​ projectName: "test"

​​ projectOwner: "baijiameng"

​​ projectStartTime: "20181022"

​​ projectState: "开发中"

​​ projectTeam: (4) […]

​​​ 0: "baijiameng"

​​​ 1: "baijiameng"

​​​ 2: "liuweile"

​​​ 3: "liubaoping"

​​​ length: 4

}

### （4）创建项目界面projectCreate.html

对应js：zjc.js、time-zjc.js

① 传参函数：initNewProject

传参过程：以POST方式向urllist.allNames.getAlUser发起请求，返回json对象如下所示，其中userRole被解析为int型数据，为0则表示用户角色为普通用户。

{

Object { userRole: "0", userPassword: "123456", userID: "baijiameng" }

Object { userRole: "0", userPassword: "123456", userID: "liuweile" }

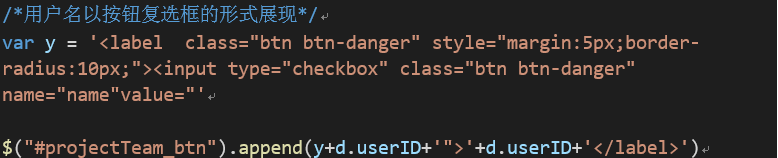
Object { userRole: "0", userPassword: "123456", userID: "zhangjiacheng" }

Object { userRole: "0", userPassword: "123456", userID: "liubaoping" }

Object { userRole: "0", userPassword: "123456", userID: "xufeifei" }

Object { userRole: "0", userPassword: "123456", userID: "bjm" }

}



② 传参函数：createNewProject

传参过程：以POST方式向urllist.newProject.createNewProject发起请求，提交的参数即为用户输入的项目信息，具体如下表所示，项目创建成功后，跳转到所有项目展示页面project.html。

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| projectName | 是 | string | 项目名 |
| projectDesp | 否 | string | 项目描述 |
| projectStartTime | 是 | string | 开始时间 |
| projectEndTime | 是 | string | 结束时间 |
| projectOwner | 是 | string | 项目创建人 |
| projectTeam | 是 | string | 项目成员 |

### （5）项目展示页面projectDetails.html

对应js：showtask.js

对应css：style5.css

页面设计：

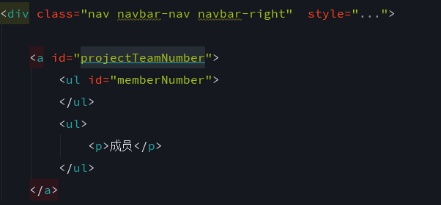
① 页面初始化函数为initProjectDesp，利用ajax函数进行传值，传参方法为POST，传入参数projectId，从后台数据库获取该projectId下所包含的项目名称（projectName）、项目简介（projectDespContent）、任务Id（taskId）、项目成员（projectTeam）等参数。



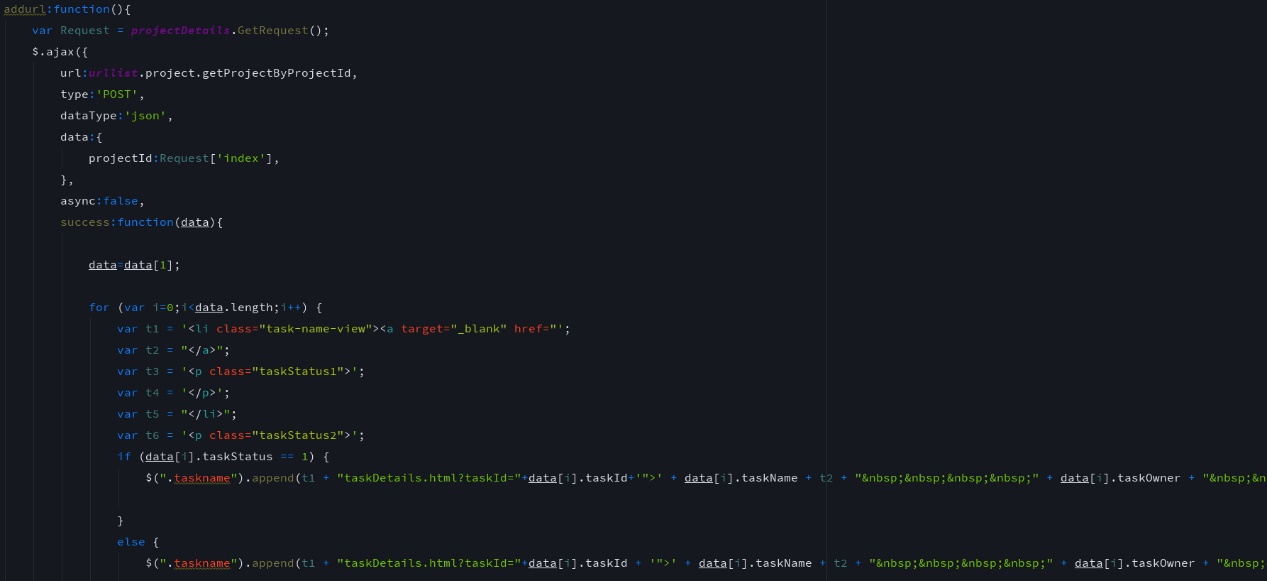
② 获取的projectName和projectDespContent直接显示在页面的左上方，对应的html中的containter下的id为header的div。



③ 页面右上角显示该项目下的成员数，在initProjectDesp函数中取projectTeam的数组的长度，并为该元素前添加点击跳转到到项目成员展示界面（projeTeam.html）的链接,之后再将其append到id为projectTeamNumber的div中。



④ 根据taskId获取到该任务下的任务名称（taskName）、任务指派人（taskOwner）、任务开始时间（taskStartTime）、任务截止时间（taskEndTime）、当前任务状态（已完成和未完成）等数据，并通过函数addurl为taskName添加跳转到到查看任务界面（taskDetails.html）的链接；通过for循环把项目下的所有任务信息逐条通过append的方式添加到class为taskname的div中。其中任务状态通过判断taskStatus的值来进行显示，已完成标为绿色，未完成标为红色。



⑤ 项目删除通过一个模态框来实现，点击删除项目的button弹出模态框，删除按钮的点击事件为确认删除项目并跳转到所有项目展示界面（project.html），点击取消按钮或者页面其他任意位置返回当前界面。删除项目的函数为deleteProject，同样运用ajax通过POST的方式进行传值，传入参数projectId，并用session中的userName判断当前的登录用户是否有权限删除此项目。



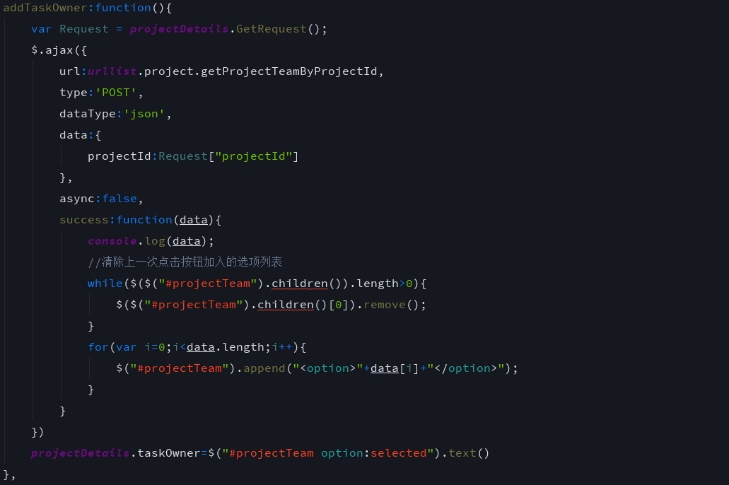
### （6）创建任务界面createTask.html

关联js：showtask.js

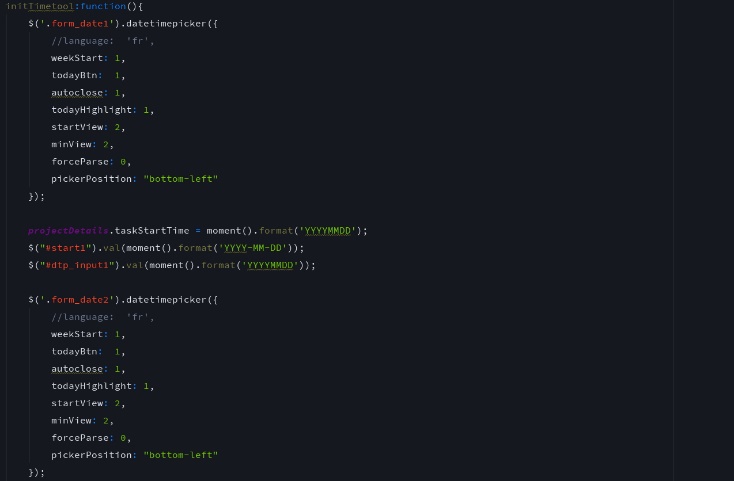
关联css：style5.css

页面设计：

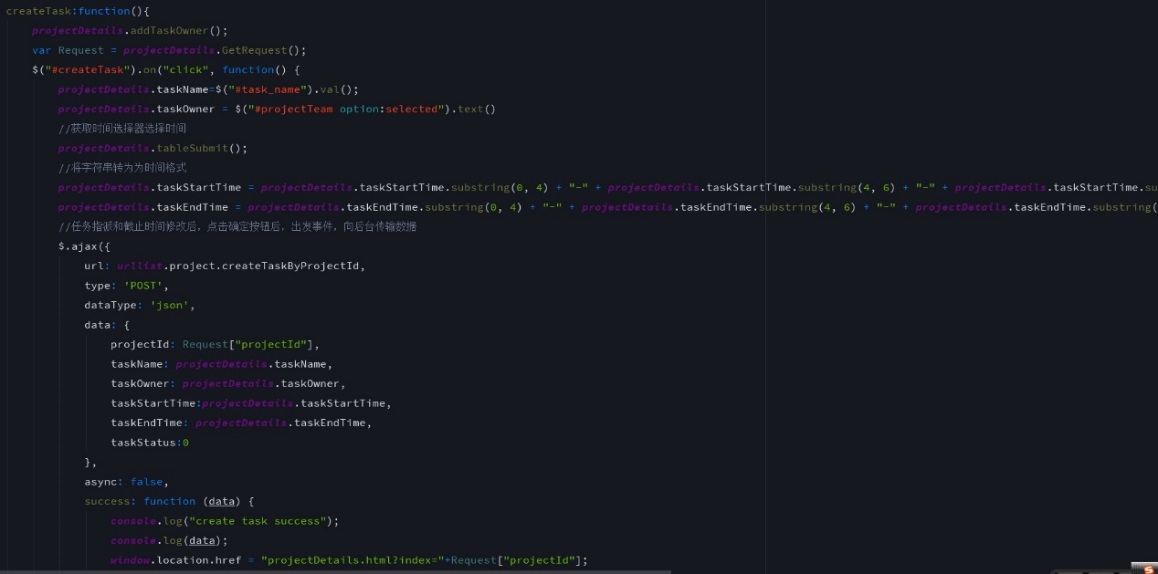
① 下拉单选框：用来选择项目指派人。下拉栏中的人员是该任务所在项目下的所有成员名称。获取成员对象对应的函数是addTaskOwner，函数通过ajax函数传入参数projectId获取projectTeam数组，考虑到之前的点击按钮可能已有选项列表被加入，所以这里通过一个while函数来判断html中的id为projectTeam的div中是否存在children，已存在则删除。之后通过一个for循环将后台数据projectTeam数组中的元素append到div中。函数的最后一行用来获取下拉框的选择的任务人员。



② 开始时间和截止时间的时间选择器：该时间选择器模块运用的是师兄项目中的模板，可以直接移植使用。initTimetool函数用于时间选择器的初始化，tableSubmit函数获取输入的任务开始时间和截止时间。



③ 添加和取消按钮：为添加和取消按钮设置点击事件，js函数名称为createTask。点击添加按钮时，首先将从输入框获取的任务名称以及从下拉框获取的任务指派人的值分别赋给projectDetails.taskName和projectDetails.taskOwner，将时间选择器获取的时间转化为时间的格式并赋给变量projectDetails.taskStartTime和projectDetTails.EndTime，利用ajax的POST方式把projectId、taskName、taskOwner、taskStartTime和taskEndTime传给后台。初始的taskStatus设为0，即未完成状态。传参完成后跳转回任务展示界面。点击取消按钮直接跳转回项目展示界面。



### （7）查看任务界面taskDetails.html

关联js：taskDetails.js

传参过程：从projectDetails.html页面使用链接传参，格式为taskDetails.html?taskId=value,到taskDetails页面通过解析链接获取taskId的值，调用getTaskByTaskId接口，获取后台数据信息，在前端展示。并把taskId保存在本地session中（sessionStorage.setItem("taskId",Request["taskId"])）。

### （8）编辑任务界面updateTask.html

关联js：taskDetails.js

传参过程：点击修改，进入updateTask.html页面，taskId:sessionStorage.getItem("taskId"),获取taskId，获取任务信息，修改任务描述后，保存，返回 taskDetails.html。注：页面传值可以不适用sessionStorage传值，但是使用sessionStorage可以使的代码简单，逻辑相对清晰。

### （9）个人任务查看界面userInfo.html

关联js：userInfo.html

传参过程：以POST方式向后台请求某用户对应的任务列表，传递参数为储存在sessionStorage中的userName，后台返回该userName对应的任务列表后前端添加标签进行展示。

### （10）项目成员展示界面projectTeam.html

关联js：projectTeam.html

传参过程：以POST方式向后台请求项目包含的成员列表，传递参数projectID，后台返回该项目包含的所有成员，前端解析json对象并进行展示。

### （11）项目成员编辑页面projectTeamUpdate.html

关联js：zjc.js

① 传参函数：listMember

传参过程：该函数包含两个POST请求，第一个请求向urllist.allNames.getProjectTeamByProjectId请求获得某个项目内的所有成员，传递参数为string类型的projectID，通过项目ID获得项目成员；第二个请求向urllist.allNames.getAlUser请求获得所有项目成员，并执行比较函数，最后只展示不在该项目内的成员。

② 传参函数：updateProjectTeam

传参过程：以POST方式向urllist.allNames.updateProjectTeam请求更新项目成员。传递参数包括string类型的projectTeam，存放修改后的项目成员，以及string类型的projectID即项目ID。

## 2.2 前后端交互

为了实现系统的基本功能，共设计前后端交互函数16个。Spring Boot实现相应的函数功能，向前端返回json对象，前端接收结果进行展示或处理。

16个函数的名称及对应的页面如下表所示，具体功能及输入输出在2.3节详细介绍。

|  |  |
| --- | --- |
| **函数名称** | **对应页面** |
| signUpByNameAndPasswordAndRole | sign\_up.html |
| signInByNameAndPassword | index.html |
| getAllProjectName | project.html |
| createNewProject | projectCreate.html |
| getProjectByProjectId | projectDetails.html |
| createTaskByProjectId | createTask.html |
| changeTaskStatus |
| deleteProjectByProjectId |
| getProjectTeamByProjectId |
| getTaskByTaskId | taskDetails.html |
| deleteTaskByTaskId |
| getProjectMemberByTaskId |
| updateTaskByTaskId | updateTask.html |
| getTaskByUserName | userInfo.html |
| getProjectTeamByProjectId | projectTeam.html |
| updateProjectTeam | projectTeamUpdate.html |

## 2.3 后台

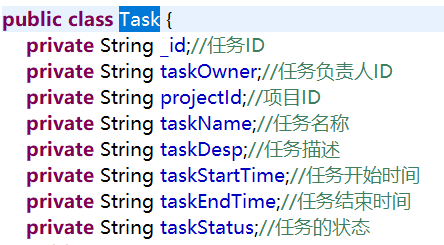
### 2.3.1 数据库设计

数据库名为ProjectManagement，其中包含的collections如下：

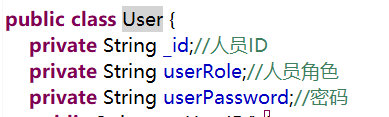
Project->项目表：



Task->任务表：



User->用户表：

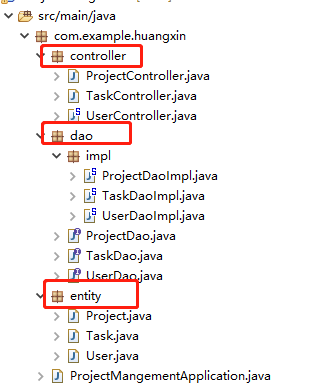


注意事项：

* 建这些类的时候注意Task和Project的主键均为\_id，mongodb插入数据的时候会自动往数据里面添加唯一标识符\_id，如果输入的时候指定了，则不会插入，若未指定，则会插入到\_id中，可以通过get方法来获得新插入的主键。
* 项目的状态目前只分为了“完成”与“未完成”两个选项，以后拓展时可以修改，但是只需要在前台修改即可，与后台无太大关系。
* 目前在Project中的projectTeam默认为projectOwner也在其中。
* 目前由于一些细节并未明确，尚未对Task的删除做权限处理。
* 原则上需要对每一个项目及任务的负责人做身份认证，判断其是不是合法用户，通用方法是前台不能随便输入，而是获取所有用户之后自行选择。
* 设置StartTime和EndTime时注意大小关系。

### 2.3.2 后台架构

后台设计主要采用三层的设计架构，结构如下：



entity层：实体层，包含Project，Task，User三个实体类。

dao层：主要是数据库的增删改查操作。外层的ProjectDao定义接口，内层的ProjectDaoImpl是其具体实现。其他的与之类似。

controller：负责与前端的交互以及调用后台的数据。

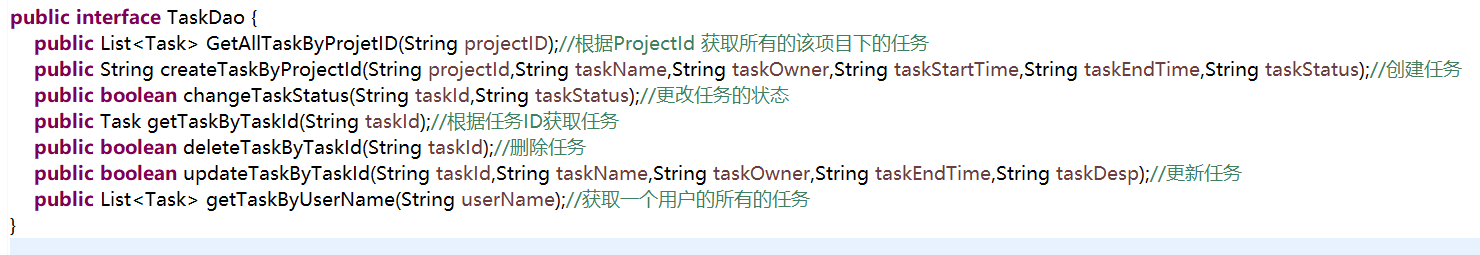
每一个层中都包含了Project，Task，User三个相关的文件，与Project相关的方法基本上都在Project相关的类中。

**（1）Dao层**

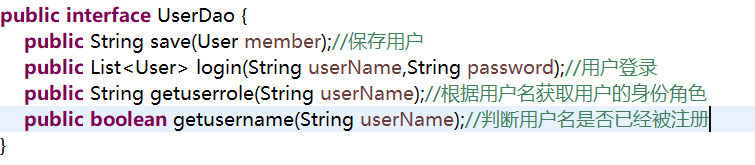
ProjectDao：



TaskDao：



UserDao:



然后各个Dao层的impl文件是这些接口的具体实现，主要是java-mongodb的一些增删改查操作。

**（2）Controller层**

Controller层负责与前端进行交互并调用Dao层的实现方法进行后台数据的处理。主要依据对象主体的不同划分为ProjectController、TaskController和UserController三类：

**ProjectController：**

**getAllProjectName**(@RequestParam String userName)：

输入：用户名

输出：List<Project>

功能：获取关于该用户的所有项目，包括该用户自己创建的以及参与的所有项目。

**createNewProject**():

输入：一个JsonStr字符串

输出：是否创建成功

功能：创建一个新的项目，注意查看一些基本的信息是否完整，同时需要注意ProjectOwner需要在ProjectTeam中。

**getProjectByProjectId**()：

输入：projectId

输出：Project

功能：根据项目ID来获得project的信息。

**deleteProjectByProjectId**()：

输入：projectId，userName

输出：是否删除成功。

功能：根据userName判断是否有删除的权限，在判断是否删除项目。只有ProjectOwner和教师才有资格删除项目。

**getProjectTeamByProjectId**():

输入：projectId

输出：项目参与人员

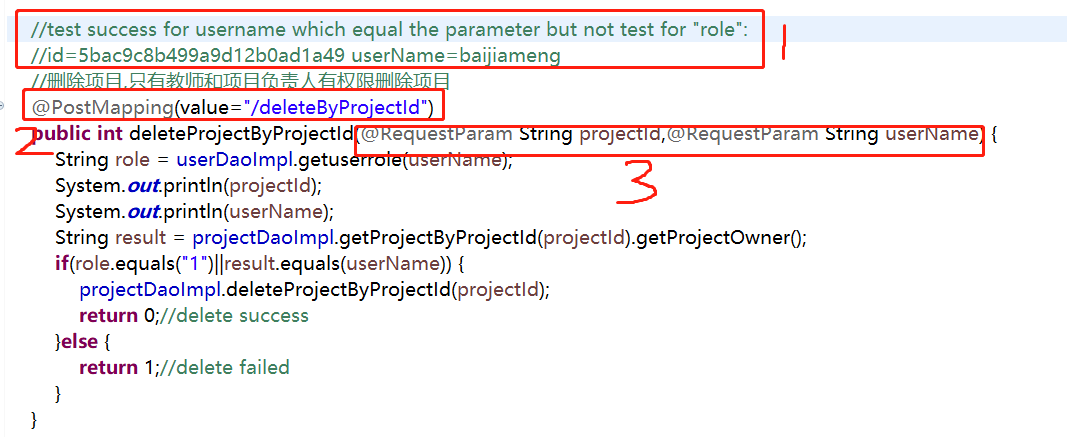
功能：根据项目ID获取该项目的所有参与成员。

**updateProjectTeam**():

输入：jsonStr

输出：是否更新成功

功能：更新项目的成员



以deleteProjectByProjectId为例，第1部分为使用Postman测试的结果以及需要post的数据，注意，发送的方法均为post，其中第二行为发送的参数。第2部分为前端请求该方法时需要请求的url后加的方法，意思是请求这个路径下的这个方法。第3部分用于接收从前台传过来的数据。其余部分做下一步的数据处理，其他的所有方法与此类似。

**TaskController：**

**createTaskByProjectId**():

输入：projectId，taskName，taskOwner，taskStartTime，taskEndTime，taskStatus

输出：返回taskID

功能：创建一个项目下的任务，该任务只属于此项目。

**changeTaskStatus** ():

输入：taskId， taskStatus

输出：成功返回0，失败返回1

功能：改变任务的状态，目前只设了完成，未完成两种状态。

**deleteTaskByTaskId** ():

输入：taskId

输出：成功返回0，失败返回1

功能：删除任务，暂未做权限控制

**getTaskByTaskId** ():

输入：taskId

输出：Task类

功能：根据taskID获取具体的任务信息。

**getProjectMemberByTaskId** ():

输入：taskId

输出：该任务所属项目的成员，List<String>

功能：根据taskID获取具体的任务所属项目的成员。

**updateTaskByTaskId** ():

输入：taskId, taskName, taskOwner, taskEndTime, taskDesp

输出：成功返回0，失败返回1

功能：更新Task。

**getTaskByUserName** ():

输入：userName

输出：List<Task>

功能：获取该用户的所有相关任务。

**UserController：**

**signUpByNameAndPasswordAndRole** ():

输入：userName，userPassword，userRole

输出：成功返回0，失败返回1

功能：用户注册

**signInByNameAndPassword** ():

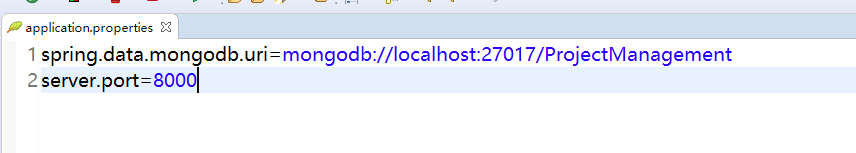
输入：userName， userPassword

输出：成功返回0，失败返回1

功能：用户登录

### 2.3.3 后台启动

在application.properties中，配置启动信息：

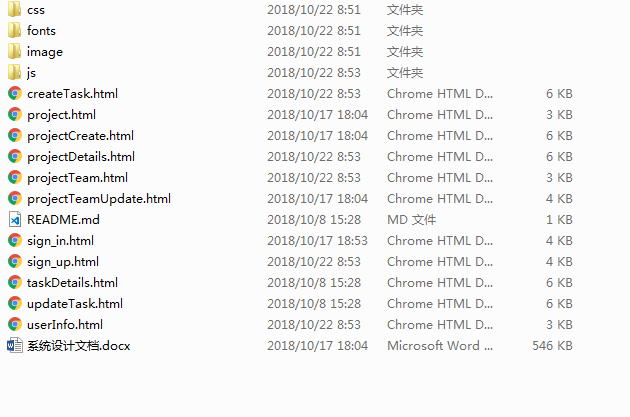


第一行代表mongodb的连接信息，使用的是本地的mongodb，端口号为默认端口：27017。第二行代表使用的端口号为8000，如果不设则默认的端口号为8080端口。在此需要注意，若电脑本身装了Tomcat，且设置的默认端口，则会与Springboot中的内置服务器默认端口冲突，故在此设为8000。

# 3 文件夹说明

本文档所在文件夹共包含6个文件/夹，其中“原始题目”是张逸飞师兄出题的文档，开发计划为刘伟乐同学写的项目规划，需求文档介绍了实现系统的基本功能和前端页面样式，技术文档中介绍了前后端交互的ajax传值函数以及后台的构建架构和搜索函数。

managementsyatem为前端代码包，结构如下：



① css文件夹：放入所有css样式文件，包括每个人负责页面的css文件以及引用库的css文件。

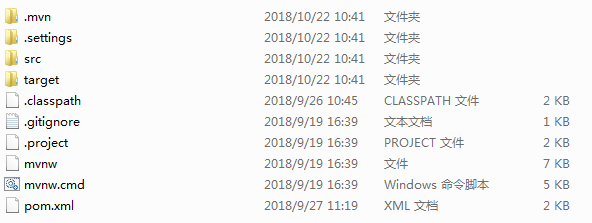
② fronts文件夹：放入所有bootstrap中提供的字体库文件，保证页面的图标效果。

③ image文件夹：存放所有使用的图片

③ js文件夹：存放每个人负责的js文件夹以及引用库的js文件。

④ 项目根目录：所有页面html页面放在根目录里面，代码中使用相对路径进行引用。

ProjectManagement为后台代码包，使用eclipse打开，在eclipse插件库中下载springboot。文件结构如下：



① src/main文件夹：项目代码目录

② pom.xml:项目配置文件

③ src/test:测试代码

# 4 各部分负责人

|  |  |
| --- | --- |
| **页面** | **负责人** |
| 系统注册页面sign\_up.html | 白嘉萌 |
| 系统登录页面sign\_in.html | 白嘉萌 |
| 所有项目展示页面project.html | 张佳程 |
| 创建项目界面projectCreate.html | 张佳程 |
| 项目展示界面projectDetails.html | 寇英帅 |
| 创建任务界面createTask.html | 寇英帅 |
| 查看任务界面taskDetails.html | 刘伟乐 |
| 编辑任务界面updateTask.html | 刘伟乐 |
| 个人任务查看界面userInfo.html | 冯文英 |
| 项目成员展示界面projectTeam.html | 冯文英 |
| 项目成员编辑页面projectTeamUpdate.html | 张佳程 |
| 数据库构建、后台查询函数编写 | 黄鑫、白嘉萌 |