# 组织架构管理系统加分项

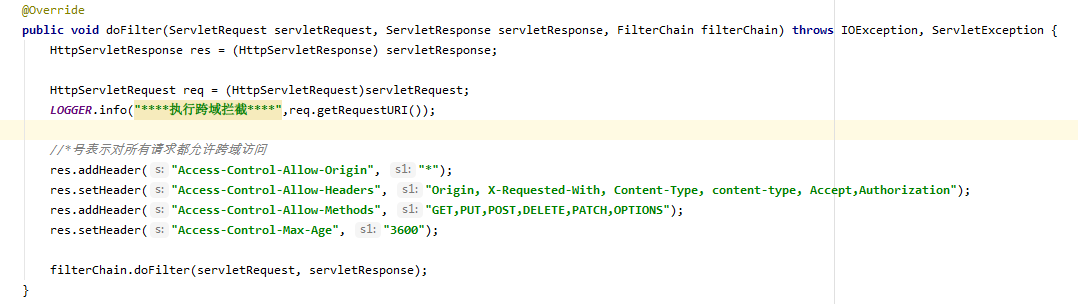
1. **加分项清单**

|  |  |  |
| --- | --- | --- |
| 编号 | 功能 | 说明 |
| 1 | 前后端分离 | 后台错误不会直接反映到页面，前端JS可以做很大部分的数据处理工作，对服务器的压力减小到最小，前端负责页面跳转，后端负责数据数据处理，前后台各尽其职可以最大程度的减少开发难度。 |
| 2 | 防止sql注入 | SQL注入，就是通过把SQL命令插入到Web表单递交或输入域名或页面请求的查询字符串，最终达到欺骗服务器执行恶意的SQL命令 |
| 3 | 记录系统操作痕迹 | 当系统数据出现意外情况时，可以根据操作记录，还原数据，以及操作人员对数据违规操作时，可以根据系统操作记录，进行操作人员追责等 |
|  |  |  |
|  |  |  |

1. **加分项实现详细说明**
2. **前后端分离实现**

后端（Java）：

1. 后端无任何静态文件资源
2. 跨域处理，设置后端允许跨域请求



1. 每个cotroller都采用Ajax数据响应（@RestController）

@RestController

**public class** JobInfoController {}

@RestController

**public class** LogInfoController {}

@RestController

**public class** OrgInfoController{}

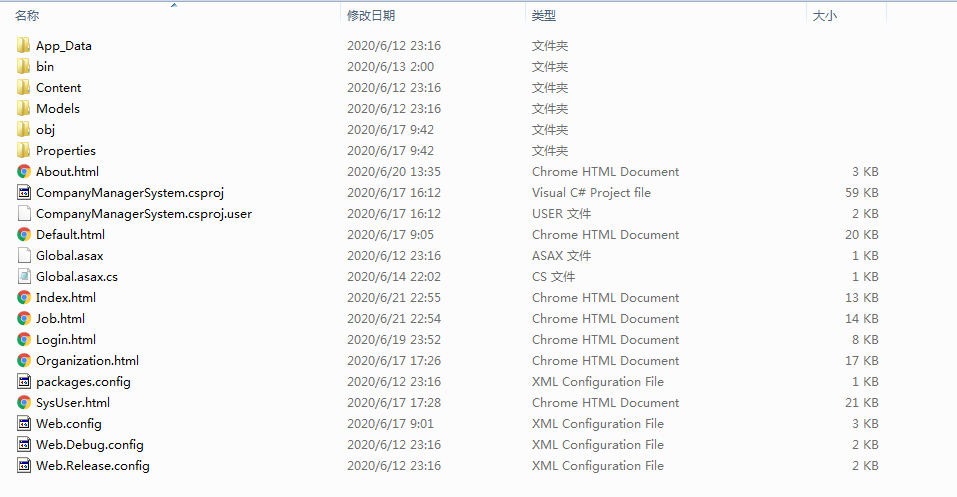
@RestController

**public class** UserInfoController{}

前端（.Net）：

Jquery+Ajax请求方式获取数据，如下示例（用户登录）：

|  |
| --- |
| var data = JSON.stringify({  "account": $.trim($username.val()),  "password": $.trim($password.val())  });  $.ajax({  url: baseUrl() + "userInfo/login",  data: data,  type: "post",  contentType: "application/json;charset=utf-8",  dataType: "json",  success: function (data) {  if (data.success == true) {  setCookie('user', $.trim($username.val()), 60 \* 60 \* 60, '/');  setCookie('password', $.trim($password.val()), 60 \* 60 \* 60, '/');  setCookie('realname', data.data.userName, 60 \* 60 \* 60, '/');  $("#login\_button").find('span').html("登录成功，正在跳转...");  window.setTimeout(function () {  window.location.href = "Index.html";  }, 500);  } else {  $("#login\_button").removeAttr('disabled').find('span').html("登录");  $.login.formMessage(data.msg);  }  }  }); |



1. **防止sql注入**

防止sql注入，可以在控制层检查前端页面穿入的参数，对请求数据进行校验，减少sql注入风险，也可以通过PreparedStatement对象对sql参数进行设值操作。该比赛中，我们小组使用的mybatis数据持久层框架，在sql参数设值，使用#{参数}方式，规避sql注入，如果必须使用${参数}方式设值，则需要提前检查和检验参数，来规避sql注入。具体操作如下示例：



1. **记录系统操作痕迹**

首先需要新建一张操作日志记录表，然后在每一个操作方法后往操作日志记录表中插入一条系统操作记录数据，一个系统的操作是非常多的，显然在每个方法后，去写插入操作记录数据是非常繁琐的，而本次比赛中，我们采用Spring中Aop技术，实现操作日志的记录，使操作记录与业务代码进行分离，具体实现如下：

|  |
| --- |
| @Aspect @Component **public class** LogAspect {   @Autowired  **private** LogInfoService **logInfoService**;   @Around(**"execution(public \* com.zuxia.controller.\*.\*(..))"**)  **public** Object around(ProceedingJoinPoint joinPoint) **throws** Throwable {  RequestAttributes requestAttributes = RequestContextHolder.*getRequestAttributes*();  *//从获取RequestAttributes中获取HttpServletRequest的信息* HttpServletRequest request = (HttpServletRequest) requestAttributes.resolveReference(RequestAttributes.***REFERENCE\_REQUEST***);   String requestURI = request.getRequestURI();  String params=argsFormat(joinPoint.getArgs());  String clsMethod = joinPoint.getSignature().toShortString();  Object proceed = joinPoint.proceed();  OptResult ort = (OptResult)proceed;   LogInfo logInfo=**new** LogInfo();  logInfo.setCreateTime(**new** Date());  logInfo.setIsDelete(Final.***IS\_NOT\_DELETE***);  logInfo.setOptIp(request.getRemoteAddr());  **if**(requestURI.contains(**"login"**)&&ort.getSuccess()){  UserInfo user = (UserInfo) ort.getData();  logInfo.setUserId(user.getId());  }**else**{  logInfo.setUserId(1);  *//UserInfo userInfo = (UserInfo) request.getSession().getAttribute("userInfo");  //logInfo.setUserId(userInfo.getId());* }  logInfo.setOptType(ort.getMsg());  logInfo.setRemark(strFormat(requestURI, params, clsMethod));  **logInfoService**.insert(logInfo);   **return** proceed;  }   **private** String strFormat(String uri, String params, String method){  **return "{\"requestUrl\":\""** + uri + **"\",\"params\":\""** + params + **"\",\"clsMethod\":\""** + method +**"\"}"**;  }   **private** String argsFormat(Object args[]) **throws** IllegalAccessException {  **if**(args==**null**||args.**length**<1){  **return "[]"**;  }**else**{  Object obj=args[0];  **if**(obj **instanceof** UserInfo || obj **instanceof** JobInfo || obj **instanceof** LogInfo || obj **instanceof** OrgInfo){  String params=**"["**+obj.getClass().getSimpleName()+**"{"**;  Field[] fields = obj.getClass().getDeclaredFields();  **for** (Field field:fields) {  field.setAccessible(**true**);  Object o = field.get(obj);  String name = field.getName();  **if**(o != **null**&&!**"serialVersionUID"**.equals(name)){  params+=name+**"='"**+o+**"',"**;  }  }  **return** params.substring(0, params.length()-1)+**"}]"**;  }  **if**(obj **instanceof** Integer[]||obj **instanceof int**[]||obj **instanceof** Array){  **return** CollectionUtils.*arrayToList*(obj).toString();  }  **return** JSONObject.*fromObject*(obj).toString();  }  } |