青芽医疗管理系统

课程目标

1. 能够说出系统的业务架构
2. 能够说出系统的技术架构
3. 能够说出系统的系统架构
4. 熟悉数据库表ER图和相关数据表
5. 能完成系统的搭建
6. 实现项目的登入业务
7. 项目综合概述
8. 项目背景

医院信息系统（Hospital Information System简称HIS）是一门集医学、信息、管理、计算机等多种学科为一体的边缘科学，在发达国家已经得到了广泛的应用，并创造了良好的社会效益和经济效益。HIS是现代化医院运营的必要技术支撑和基础设施，实现HIS的目的就是为了以更现代化、科学化、规范化的手段来加强医院的管理，提高医院的工作效率，改进医疗质量，从而树立现代医院的新形象，这也是未来医院发展的必然方向。

为了满足新时代医院发展的需要，为了使医院能真正的进行信息化管理，宏鹏软件外包公司集中了大量的人力和物力，开发了《青芽医疗管理系统》。本系统严格遵照卫生部颁发的《医院信息系统基本功能规范》及国家相关政策，且充分考虑医院实际信息化管理的需要，实现整个医院的人、财、物等各种信息的顺畅流通和高度共享，并结合医院管理制度和先进的管理思想，为医院管理的信息化、高效化提供强大的管理平台。

1. 业务架构
   1. 业务模块

重要说明：

1）讲解前，请先演示项目页面效果，增加对于功能的了解和认识。

2）项目可以从提供的源码中导入，请授课前提前部署、测试

《青芽医疗管理系统》整个业务模块，分为5个模块，如图所示：

图片包含 图示

描述已自动生成

功能模块说明

|  |  |  |
| --- | --- | --- |
| 序号 | 功能模块 | 功能描述 |
| 1 | 用户管理 | 1、用户登录(token方式) 2、用户图像(OSS方法)和密码修改 3、查看用户列表 3、创建用户，同时上传用户图像 |
| 2 | 挂号管理 | 1、创建就诊卡 2、门诊挂号 3、查看挂号信息列表 4、查看挂号详情 |
| 3 | 门诊管理 | 1、查看挂号单信息列表 2、查看挂号详情 3、疾病诊断-添加医嘱 4、疾病诊断-添加药方（本次不开发，扩展内容） |
| 4 | 收费管理 | 本次不开发，自行扩展 |
| 5 | 药房管理 | 本次不开发，自行扩展 |

2.2 业务流程图

图示, 示意图

描述已自动生成

1. 技术架构
   1. 技术架构介绍

对应本项目，由于只是一个后台管理系统，不面向病人。我们分5个层次来介绍技术选型和架构。

• 数据库采用MySQL进行数据存储

• 数据库交互技术我们选择MyBatis+C3P0数据源

• 业务层就是Service，文件的上传采用阿里OSS技术

• 采用SpringMVC作为控制器，它是前后端交互的接口。

• Spring作为基础框架，用于整合SSM

•前端展示使用Vue和Element UI。

图形用户界面

中度可信度描述已自动生成

* 1. 开发环境说明

程序架构:三层架架、前后台分离架构

后台技术:Spring+Springmvc+Mybatis+MySql+…

前端技术:html、js、css、Ajax、vue、elementui

其它:oss文件服务器…

1. 系统架构

所谓系统架构意思是应该采用怎样的一种结构进行项目的搭建。我们这里还是采用三层架构。三层架构的形式开发，分为表现、业务、数据。表现层负责数据的收集以及回显，业务层负责业务逻辑处理，数据层负责与数据库打交道。

图示

描述已自动生成

1. 项目数据库设计
2. 理解数据库的E-R图

图示

描述已自动生成

1. 理解常用数据表

**sys\_users用户表**图形用户界面, 文本

描述已自动生成**sys\_roles角色表**图片包含 图形用户界面

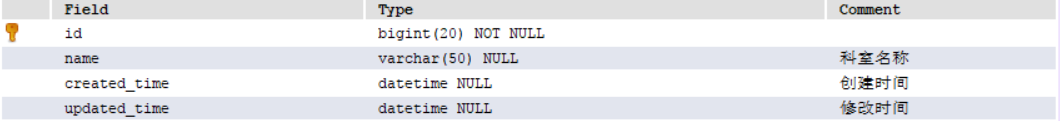
描述已自动生成**patient病人表（就诊卡表）**图形用户界面

中度可信度描述已自动生成**registry挂号单**图形用户界面

描述已自动生成**prescription药方表**图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成**room就诊房间表**图形用户界面, 应用程序

描述已自动生成**schedule医生排期表**图形用户界面

低可信度描述已自动生成**department就诊科室表****medical\_advice医嘱表**图形用户界面, 应用程序

描述已自动生成**sys\_menus**图片包含 图形用户界面

描述已自动生成**sys\_roles\_menus角色菜单表**图形用户界面

描述已自动生成

1. 搭建项目的开发环境
2. 创建工程

创建web工程>组织包的结构>添加ssm开发环境>反向工程>运行项目

1. 添加前端项目资源
2. 还原数据
3. 统一响应体解决方案
4. 什么时统一响应体？

在目前的前后端分离架构下，后端主要是一个restful api的数据接口。但是HTTP的状态码数量有限，而随着业务的增长，HTTP状态码无法很好地表示业务中遇到的异常情况。那么可以通过修改响应返回的JSON数据，让其带上一些固有的字段，例如以下这样的格式：

图片包含 形状

描述已自动生成

其中关键属性的用途如下：

- code:为返回结果的状态码

- msg:为返回结果的消息

- data:为返回的业务数据

1. 编写统一响应体

|  |
| --- |
| /\*\*  \* 返回信息包装类  \* @param <T>  \*/  @JsonInclude(JsonInclude.Include.NON\_NULL)  public class Result<T> implements Serializable {  public String code;  public String msg;  private T data;  /\*\*  \* 无参构造  \*/  public Result() {}  /\*\*  \* 根据code，msg创建一个Resutl  \* @param code  \* @param msg  \*/  public Result(String code, String msg) {  this.code = code;  this.msg = msg;  }  /\*\*  \* 根据code，msg创建一个Resutl  \* @param code  \* @param data  \*/  public Result(String code, T data) {  this.code = code;  this.data = data;  }  /\*\*  \* 根据code，msg，data创建一个Resutl  \* @param code  \* @param msg  \* @param data  \*/  public Result(String code, String msg, T data) {  this.code = code;  this.msg = msg;  this.data = data;  }  Setter和getter方法略  } |

1. 实现登入业务
2. 编写后台接口代码
   1. dao层：略
   2. 编写业务

|  |
| --- |
| 接口  public interface UserService {  //实现登入  public SysUsers login(String username, String password);  }  实现:  @Service  public class UserServiceImpl implements UserService {  @Autowired  private SysUsersMapper sysUsersMapper;  @Override  public SysUsers login(String username, String password) {  //调用dao层  SysUsersExample example=new SysUsersExample();  SysUsersExample.Criteria criteria = example.createCriteria();  //添加条件  criteria.andAccountEqualTo(username);  criteria.andPasswordEqualTo(MD5Utils.md5Encrypt(password));//将密码进行加密后进行比较  List<SysUsers> list=sysUsersMapper.selectByExample(example);  if(list!=null && list.size()==1){  return list.get(0);  }  else  return null;  }  } |

* 1. 编写控制器

|  |
| --- |
| @Controller  @RequestMapping("/user/") //将将请分模块  public class UserController {  @Autowired  private UserService userService;  // user/login?username=?&password=?  @RequestMapping("login")  @ResponseBody  public Result<SysUsers> login(String username, String password, HttpSession session){  //调用业务  SysUsers sysUsers=userService.login(username,password) ;  if(sysUsers==null){  return new Result<>("1001","登入失败"); //表示登入失败  }else{  //将用户信息保存到session  session.setAttribute("userInfo",sysUsers);  return new Result<SysUsers>("1000","登入成功",sysUsers); //表示登入成功  }  }  } |

1. 编写前端页面代码

|  |
| --- |
| <script type="application/javascript">  new Vue({  el:'#div',  data:{  loginForm: {  account: '',  password: ''  }  },  methods:{  //登入的函数  submitForm:function(){  //1.获取用户名和密码  //console.log(this.loginForm)  //2.调用后台接口实现登入----使用axios  axios.get('user/login',{  params:{ //params表示？号请求参数  username:this.loginForm.account,  password:this.loginForm.password  }  }).then((ret)=>{  //console.log(ret.data);  if(ret.data.code==1000){ //1000表示成功 1001 表示失败  //跳转页面  location.href='index.html';  }else{  //提示:  //alert("用户名密码错误，登入失败");  this.$alert('用户名密码错误', '登入提示', {confirmButtonText: '确定'});  }  }).catch((error)=>{  console.log("调用接口失败！"+error)  });  }  }  });  </script> |