厚朴诊所信息系统（前端文档）

1. 整体设计

1. 技术选型

- 系统采取单页面应用（*single page application*）模式。
- 前端模块加载方式使用异步加载（*AMD*）规范的 *Require.js* 。
- 前端开发架构采用MVC模式的 *Backbone.js*。
- UI框架采用 *Amaze UI v2.4.2* 。
- 压缩、打包js文件使用 *Require.js* 的插件 *r.js*。
- 系统整体采用前后端分离的 *Restful*架构，前端向后端发起的请求均为 *ajax* 请求。

2. 文件结构

- **css**（包含项目所有css文件）
- **dist**（文件压缩打包后的输出目录，网站实际的文件地址）
- **imgs**（包含项目所有的图片，图标文件）
- **js**（包含项目所有js文件）
  - **libs**（包含项目引入的基础包，如query，backbone，amazeui）

    - **modules**（包含系统子功能的文件夹，子文件夹对应菜单，如Index包含首页所有页面逻辑，Bill包含收费所有页面逻辑等）
    - **plugins**（包含项目引入的基础包上的插件，如amazeui-chosen）
    - almond.js（多文件模块化加载入口，简化的requirejs）
    - config.js （全局配置文件，预留，没有实际逻辑）
    - jethisMain.js（require.js设置的主文件，同时有菜单权限和用户信息的逻辑）
    - jethisRouter.js （backbone路由设置文件，包含页面通用逻辑）
    - libs.js （方便打包基础包的文件）
- build.js（r.js 打包配置文件）
- jethis.html（主页面，所有路由导向的页面只是更新这个页面）
- r.js（打包工具）
- regist.html（注册页面）
- restPwd.html（重设密码页面）
- SignIn.html（登录页面，*www.jethis.cn* 即指向这个页面 ）

3. 页面结构

由于是单页面应用，页面结构基本一致，由 **jethis.html** 决定。**jethis.html** 的结构为上－中－下。

- 上方显示logo，一级菜单，用户名，科室，未读消息数，以及退出按钮。
- 下方显示系统名称以及客服电话。
- 中间又分为左右两个部分，左侧为二级菜单，右侧为菜单对应的内容。

其中，左侧菜单通用模板位于**/js/modules/Common/sidebar.html**，在登录获取到用户的权限并保存到本地sessionStorage [1]以后，由jethisMain.js [2]中对数权限数据进行处理，即可组装用户拥有的菜单数据，通过handlebars引擎渲染到DOM上。

4. jethisRouter.js和jethisMain.js中的全局逻辑

- **jethisRouter.js** 中除了设置路由，跳转页面逻辑以外，还设置了每个页面公用的逻辑：
  - ajax全局设置，error回调，timeout超时。
  - 导出excel按钮全局事件绑定，点击按钮导出所在表的Excel。
  - 页面载入时的焦点聚焦到含 “first-focus” 类名的元素上。
  - 含有start类名的datepicker选择的日期必须在end类名的datepicker选择的日期之前。
  - 请求用户实时的未读消息数量。
- **jethisMain.js** 中除了定义文件路径以外，还设置了页面初次载入的逻辑：
  - 开启路由。

  window.router = new Router(); Backbone.history.start();
  - 显示右上角用户信息，包含用户名，科室（如果没有则不显示），绑定退出按钮的点击事件，包括清除缓存等。
  - 根据权限显示一级菜单和二级菜单。
  - 轮询（原用作保持登录状态，后来后端换了一种实现方式，此处预留，将来需要长连接即可修改此处）
  - 禁用点击backspace事件，防止引发的意外浏览器回退现象。

---

1 /js/modules/viewModule/signin/login.js 中160-172行

2 /js/jethisMain.js 79-111行即是根据权限生成菜单的逻辑

2. 库、组件

1.Backbone.js 1.2.3

- **router**包含前端路由，以及每个页面载入前的逻辑，已在上文全局逻辑中说明，此处不赘述。**jethisRouter.js**中引用了各个一级菜单的view，在routes属性中配置了各个一级菜单的路由。路由的可选部分放在括号中**(/:optional)**，当点击二级菜单时，即传递子菜单的submenu到optional中（例:#bill/charge",传递bill作为**changePage**的参数），由于一级菜单一般没有自己的html页面，所以每次都会传递submenu到**changePage**中，随后渲染页面(**$('.admin-main').html($(view.el));**)，再进行一系列的全局初始dom操作。
- **view**里包含了页面的全部逻辑，**lisentTo**方法与**on**方法为view与其他model、view通信的方式，写在**initialize**方法里。对dom元素的操作（包括**bootstrapTable**或者**datepicker**这样的组件渲染dom）都写在**render**方法里。初次请求写在**render**方法里。所有的事件写在**events**里（必须是**view.el**里面的元素上绑定的事件）
- **model**包含所有与后端交互的连接，基本都是**ajax**。如果方式为**get**，则按照常用的方式传递参数，附在链接后面即可，实现方式为**$.ajax({ type:'get' , data:{})**。如果方式为**post**或者**patch**，传递的form表单必须转为JSON字符串，**$.ajax({ type:'post' , data:{JSON.stringify(param)})**。
- **model**与**view**通信的方式为trigger events。model内部设置一个result的对象，在success和error的情况下触发相同的事件(eventName)，返回不同的结果。注意:在开发中期，为了统一请求错误的表现，舍弃了done/fail的回调方式，并且在**jethisRouter.js**中通过ajaxSetup全局设置了error的回调，所以很多model的方法中只写了success回调。但是到这个迭代完成，

后端url并没有统一的错误标识码规范，因此很多方法加上error回调也是很有必要的，后续迭代可以完善这个方面。

2.Amaze UI v2.4.2

- 页面整体布局采用了amazeui的样式及inputgroup，datepicker，tabs，panel，collapse等组件。
- 修改了amazeui，修改的地方具有注释（for jethis）
- 由于amazeui与bootstrap组件上均绑定在**$.fn**上且同名，因此组件在调用时往往会发生冲突。当前解决方案是，在 **/js/libs/bootstrap/bootstrap.js** 中把相同名字的组件修改名称。（bootstrap.js 713-726行，修改collapse为Btcollapse；bootstrap.js 2345-2347行，修改modal为bootstrapModal；）

3.bootstrap-table 1.11.0

- 几乎全部的数据呈现都采用了bootstrap-table。常用的方法是**load**加载数据，onClickRow单行点击事件，pagination分页等。
- bootstrapTable加载即**sidePagination**属性有两种方式，client和server。client就是载入本地数据，在数据量小的情况下，可以使用client模式，直接load数据，绝大多数加载均采用了这个模式。server模式是在后端已经完成分页算法情况下的载入方式，url设置为请求的url，使用queryParams传递参数，responseHandler处理返回的数据，点击页码时触发onPageChange方法向后端请求对应的页码。**注意**：server模式下，数据源必须为**{rows:[],total:20}** 这种格式。

- 由于对插件认识的不完善，项目中没有采用server模式。**/js/libs/bootstrap-table/bootstrap-table.js** 中增加了jethis，修改的代码为1304-1305，1600-1603，2359-2361行。这种情况下数据源必须为**{rows:[],dataNum:20}** 这种格式。

4.tableExport

- jethisMain.js 224-236行给全局.excel_tool 绑定了导出excel的方法。
- 导出excel中的单元格中数字过长，会采用科学计数法。解决方案是 **/js/plugins/tableExport/tableExport.js** 修改了348-349行，给td加上了style样式: **mso-number-format:/@;** 。
- 所有**.table_panel**的标题必须加上**panel_title**的className。

5.chosen

- amzeui样式的select插件。
- 每次赋值后必须**$elem.trigger("chosen:updated")**。
- 多选时，用数组**[a1,a2,a3]**赋值，val()返回**"a1,a2,a3"**

6.r.js

- 压缩打包工具，配置文件在**build.js**。
- 由于逻辑较多，打包成单个文件体积太大， 因此分为多个模块导出。

7.almond.js

- require.js 的精简版。
- 可以不设置data-main，自动解析define方法定义的模块。

3. 注册登录

- 注册
  - 页面为**regist.html**。
  - 逻辑在 **js/modules/viewModule/signup/signup.js**，**js/modules/viewModule/signup/baseInfo.js**。
  - 注册提交方式为post，数据格式为**formdata**。
- 登录
  - 页面为**SignIn.html**，逻辑在 **js/modules/viewModule/signin/login.js**。
  - 登录采取了aes加密。
  - 登录成功以后保存了基础信息，包括医院，医生，用户，当前对**sessionStorage**存取没有加密，后续可以考虑加上加密/解密操作。
- 重设密码
  - 页面为**resetPwd.html**，逻辑在 **js/modules/viewModule/resetPwd/resetPwd.js**

4. 功能模块

- 由于页面逻辑过多且重复，文档中逐一叙述于查询无益，此处均简述。主要概括了各个菜单的功能，以及不同模块之间的联系。
- 所有一级菜单的文件均有一个同名的View（如/modules/Bill文件夹中的billView.js）,负责接收由jethisRouter分发过来的子菜单id，也即submenu，随后选择相应的子view，赋值给这个view的el属性。以下只描述各二级菜单的功能及逻辑。
- p.s. curd 表示增加(Create)、读取查询(Retrieve)、更新(Update)和删除(Delete)

1.首页 (/js/modules/Index)

- 通知
  - 请求的方法为 **IndexModel.getTodaynews**，对应的URI为 **/jethis/news/allNewsRecord/1**
  - 显示当前收到的通知标题及时间。通知可以在**办公-推送通知**中由管添加，同时可以在**办公-我的通知**中查看详情。
  - 点击首页的通知也可以跳转到**办公-我的通知**页面。
- 动态
  - 请求的方法为 **IndexModel.getDynamicNews**，对应的URI为 **/jethis/News/GetNewsRecord**
  - 显示当前收到的新闻标题及时间。通知可以在**办公-发布动态**中由管理员添加，同时可以在**办公-全部动态**中查看详情。
  - 点击首页的新闻也可以跳转到**办公-全部动态**页面。
  - **后续可以加入图片，轮播展示。**
- 今日待办事项

- 请求的方法为 **IndexView.changeSelect**，调用通用查询接口[3]
- 当前登录用户为医生，则显示当天医生的排班，点击日期可以进行查询。
- 当前用户没有医生角色，则无功能。

- 今日概况
  - 请求的方法为 **IndexModel.getHospitalCondition**，对应的URI为 **/common/HomePageInfo**
  - 根据角色显示统计内容，显示的columns位于IndexView 19-35行，例：医生角色显示挂号人数，就诊人数，中药处方数，西药处方数。

2.挂号 (/js/modules/Regist)

- 新开就诊**(./register)**
  - **挂号与患者注册功能。**
  - 患者查询的方法为：通过id查询 **/js/modules/Common/patientModel.searchPatientById**，通过患者姓名查询 **/js/modules/Common/patientModel.searchPatientByName**，通过id获取患者时只有一条记录，直接对患者详情进行填充，通过姓名获取患者时如果有一条记录则直接填充患者详情，如果有多条记录，会在患者姓名查询框下生成表格，点击一行选择患者并填充患者详情。
  - 挂号的号池可以在**办公-排班**设置。查询当前号池的方法是 **registerModel.getDoctor**，URI是 **/jethis/registeration/**

[3] 位于/js/modules/Common/commonModel.js 11-37行，URI是/jethis/query/get

get_registeration_data，此处会**根据当前时间来获取**，早上可以看到所有班别的医生，下午就只能看到由下午班和夜班的医生。

- 挂号的方法是**registerModel.regist**,URI是 **/jethis/registeration/patientregisteration**
- 目前挂号仅支持现金收费。
- 患者注册。方法是**/js/modules/Common/patientModel.addPatient**，URI是 **/jethis/registeration/addpatientinfo**。

- 挂号管理**(./searchRegister)**
  - **查询所有挂号情况**。
  - 对于未收费的挂号（多为医生加号）可以进行收费，对于未就诊的挂号进行补打印。
  - 查询挂号记录的方法是**searchRegisterModel.getRegInfo**,对应的URI是 **/jethis/Patient/RegisterRecord**。
  - 补缴挂号费的方法是**searchRegisterModel.chargeRecord**,对应的URI是 **/jethis/registeration/NewChargeRecord**。
  - 打印挂号费的方法是**searchRegisterView.printReg**。
- 挂号统计**(./totalRegister)**
  - 默认显示根据科室统计的挂号数据，点击科室统计表格的任意行，弹框显示该科室内不同医生的挂号数据。
  - 查询挂号记录的方法是**searchRegisterModel.getDepartment**,对应的URI是 **/jethis/registeration/statisticsbydepartment**。
  - 补缴挂号费的方法是**totalRegisterModel.getdoctotal**,对应的URI是 **/jethis/registeration/statisticsbydoctor**。

3.诊疗 (/js/modules/Diagnose)

- 就诊队列**(regPatsView)**
  - 获取当前医生的待诊、已诊患者。
  - 点击患者获取详细信息。
  - 双击患者或者点击 开始就诊 按钮，现诊显示患者详细信息以及历史就诊记录。
  - 点击就诊记录显示记录详情，可以补打印病历。
- 处方界面**(recipeView)**
  - 初始化中间处方部分的dom结构，包括各个panel以及处方药物的表格。
- 药品处置**(treatmentsView)**
  - 初始化药品及处置的表格，并请求相应的数据。
  - 点击药物时，触发事件。430行，**that.trigger("addTreatments", [row]);**
- 诊疗主界面**(diagnoseView)**
  - 将上述三个view的内容填充进diagnose的左中右分栏。
  - 监听子view的事件。如，54行，**this.treatments.on("addTreatments", this.addTreatments, this);**，执行addTreatments方法，添加药物进处方界面。
  - 点击panel上的panel_tool，弹出相应的弹窗，例如 体格检查，既往史等。

4.财务 (/js/modules/Bill)

- 收费 **#bill/charge**

    - 查询患者的诊疗记录，单击记录显示明细，**可以修改明细数量进行收费**，如果是会员会返回打折后返回的总费用。
- 退费 **#bill/refund**
    - 当前版本对单次收费记录只能全部退费。
- 收费记录 **#bill/searchCharge**
    - 查询收费记录。
- 退费记录 **#bill/searchRefund**
    - 查询退费记录。
- 处方发药 **#bill/dispense**
    - 对已收费的处方进行发药
- 对账 **#bill/settle**
    - 对收费员实收的总费用与明细核对。
- 对账纪录 **#bill/searchSettle**
    - 查询对账记录
- 零售收费 **#bill/retailPricing**
    - 可以是系统内的患者也可以自定义输入患者，零售药品
- 零售记录 **#bill/retailDrugRecord**
    - 查询零售记录。

5.药品 (/js/modules/Medicine)

- 库存查询 **#medicine/mcurStorage**
    - 查询当前库存，药物以批次+药品编码为基本单位。
- 药品预警 **#medicine/mwarning**
    - 查询库存不足或者即将过期的药品。
    - 可以设置药品预警提示的时间，频率以及条件（阈值），**仍需调试**，URI调试完成了但是后端短信没发。

- 药品信息管理 **#medicine/mdictionary**
  - 药品基础库，查询药物信息及说明书。
- 供应商 **#medicine/msupplier**
  - curd供应商。
- 采购入库 **#medicine/minStorage**
  - 可以扫码或者查询选择药物。
- 入库审核 **#medicine/minCheck**
  - 对入库的项目进行审核。
- 库存盘点 **#medicine/mverify**
  - 顾名思义，盘点药物的数量变化。
- 盘点日志 **#medicine/mverifyLog**
  - 查询盘点数量的变化。
- 库存变动查询 **#medicine/mnumChange**
  - 宏观监测药品的数量变化，数量增加包括入库，盘点增加等，出库包括处方，零售，盘点减少等。
- 药品调价 **#medicine/mpriceChange**
  - 对药品进行调价。
- 调价记录 **#medicine/drugPriceAudit**

6.患者 (/js/modules/Patient)

- 我的患者 **#patient/patientClass**
  - 医生就诊过的所有患者。
  - 可以对患者进行分组，设置分组条件后，患者自动归类到该组。
- 医院患者 **#patient/Hospitalpatients**
  - 显示医院患者，并可以对患者信息进行编辑。可以查看患者的历次就诊信息。

- 科室患者 **#patient/Patientsdepartment**
  - 显示当前登录医生所在科室就诊的全部患者，可以根据条件查询，当前版本不可用。
- 添加患者 **#patient/addpatient**
  - 与挂号处添加患者的逻辑相同。

7. 办公 (/js/modules/Office)

- 坐诊计划 **#office/schedule**
  - 显示医生的坐诊计划。
- 排班 **#office/arrange**
  - 管理员对医生进行排版。
  - 点击编辑后，点击表格单元格进行编辑。
  - 点击确认后即提交。
- 推送通知 **#office/sendInfo**
  - 推送短通知
- 我的通知 **#office_historyMessage**
  - 查询自己收到的所有通知。
- 全部通知 **#office/allNews**
  - 仅管理员可见，查询所有发出的通知。
- 发布动态 **#office/postDynamicNews**
  - 仅管理员可见，发布医疗、平台动态。
- 全部动态 **#office/allDynamicNews**
  - 看到全部的动态。

8. 会员 (/js/modules/Member)

- 会员列表**#member/memberList**
  - curd会员信息，对会员进行充值。
- 会员统计**#member/memberStatistics**
  - 对会员充值消费的积分信息进行统计，**当前版本没有调试，这个菜单暂时禁用，用户无法看到**。
- 会员等级**#member/memberSetting**
  - curd会员等级
- 会员充值记录**#member/memberRecord**
  - 记录、查询充值目录。

9. 统计 (/js/modules/Stastitical)

- 药品流水统计**#statistical/InventoryContrast**
  - 记录每一次药品的销售。
- 药品销售统计**#statistical/drugsTo**
  - 记录每一次药品的销售。
- 处方统计**#statistical/prescriptionStatistics**
  - 记录医生开具处方数量，以及中西医处方的比例。
- 收入/消费统计**#statistical/yearReport**
  - 统计诊所一段时间的收入，以及患者产生的消费。
- 药品采购统计**#statistical/drugPurchase**
  - 统计诊所采购药物。
- 医生订单统计**#statistical/presItemByDoctor**
  - 统计医生每次订单的费用，记录医生开具订单流水。

10.设置 (/js/modules/Setting)

- 我的诊所 **#setting/myClinic**
  - 显示当前诊所的注册信息。
- 功能设置 **#setting/functionSetting**
  - 对诊所功能的基础设置，目前可以进行四舍五入的设置，待扩展。
- 修改个人信息 **#setting/changePersonal**
  - 修改用户的基本信息，密码，头像，绑定手机。
- 科室管理 **#setting/departMange**
  - 增加，修改，编辑科室。
  - 科室属于基础设置，没有科室或者删除科室会对用户管理，开处方等均产生影响。
- 用户管理 **#setting/userMange**
  - 添加用户，编辑用户信息，修改用户权限。
- 角色权限设置 **#setting/menuSet**
  - crud角色信息，给角色设置相应的权限。属于管理后台功能，**仅仅为了开发方便，才将此菜单放在当前版本中，后续应该转移到厚朴诊所信息系统的后台管理系统**。
- 挂号费用设置 **#setting/registerFeeSet**
  - 设置医院各个级别医生 普通号、专家号的挂号费用。
- 诊疗模板设置 **#setting/recipeMaintain**
  - curd诊断模板，医嘱模板，处方模板。
- 检查项目设置 **#setting/checkSet**
  - curd诊疗项目
- 字典表维护 **#setting/dicTableMaintain**
  - 设置药品单位的字典。由平台维护，**与生成处方关联**，只有字典表中有相应的单位，保存处方才能成功。