专业技能

★ 熟练掌握HTML/CSS构建符合W3C标准的页面；

★ 熟悉 CSS3，掌握常见布局、盒模型、浮动、定位等概念，可准确还原设计稿；

★ 熟悉ES6语法、事件模型、闭包、 原型链、面向对象等知识，了解正则表达式基本语法；

★ 熟悉Ajax/jQuery/Javascript/Bootstrap等技术，对React框架有基本认识，对数据驱动视图有一定了解；

★ 了解页面加载和渲染机制和网页性能优化方案，能解决常见跨域问题，对前后端联合开发的技术原理有基本认识；

★ 熟练使用Sublime、WebStorm，版本管理工具Git，代理工具Charles，构建工具Webpack, 切图工具Photoshop；

★ 熟悉Linux 操作系统和常用的操作命令，掌握HTTP、TCP/IP相关协议，了解数据库基本知识。

项目经验

* 网上购物商城 Web前端开发

项目介绍：

该电商平台类似于京东商城、淘宝商城，主要包含用户模块、商品模块、购物车模块、订单模块和支付模块。

前后端完全分离式开发，前端技术为html、css、js 、jQuery(用到了它的Ajax请求和DOM操作)，采用分层架构设计，其中工具层封装了一些通用js操作，数据层定义了各模块会用到的接口操作，逻辑层完成业务逻辑的实现。采用Webpack构建项目，使用CommonJS模块化方案。

项目职责：

1. 分析项目需求，确定模块功能划分；
2. 合理使用webpack实现前端开发的自动化，使用loaders加载相应模块，正确使用插件提高开发效率；
3. 对常用的js操作做了封装，包括网络请求、获取url参数、字段验证、渲染html模板等；用HTML、CSS完成了多个模块的页面设计，用jQuey主要完成了以下任务：表单同步/异步验证，小型SPA开发，模态框组件封装，多级联动下拉菜单，复杂表单的回填，独立组件（轮播图和分页）的抽离等。
4. 完全通过接口和后端做数据交互，通过ajax向服务端发送请求并返回json数据，再对返回数据做处理。
5. **小组分配**：和同学做的项目，2个前端，2个后端，年初做的，断断续续4个月左右。
6. **选用jquery原因**：求稳定，用户类型多样/考虑兼容性，有SEO要求，多页应用。
7. **前后端完全分离**：主要关注纯静态html，完全通过接口和后端做数据交互，优：完全脱离后端模板，系统复杂度低，缺点：不太利于SEO，优化方案：server render/蜘蛛定制页面。
8. **AMD（**异步模块定义，是RequireJS在推广过程中对模块定义的规范化产出,提前执行）和CMD（通用模块定义，是SeaJS在…，延迟执行）区别是加载和运行的时机不太一样，不太喜欢他俩的原因是将模块化代码和业务代码掺杂在一起了。CommonJS与nodejs模块化方案相一致，模块化语法比较少，并且与业务代码也是分开的；ES6模块化方案比较新，组织方式与commonjs有点相近，但对旧版本的浏览器的兼容性不是很好，所以课程选用CommonJS模块化方案。

**AMD与CMD区别：1.**对于依赖的模块，AMD 是提前执行，CMD 是延迟执行。不过 RequireJS 从 2.0 开始，也改成可以延迟执行（根据写法不同，处理方式不同）。CMD 推崇 as lazy as possible. **2.** CMD 推崇依赖就近，AMD 推崇依赖前置。看代码：

1. //CMD
2. define(function(require, exports, module) {
3. var a = require('./a')
4. a.doSomething()
5. // 此处略去 100 行
6. var b = require('./b') // **依赖可以就近书写**
7. b.doSomething()
8. // ...
9. })
10. // AMD 默认推荐的是
11. define(['./a', './b'], function(a, b) { // **依赖必须一开始就写好,** **相对我们定 义模块的函数 function, AMD 加载器是提前执行所有依赖**
12. a.doSomething()
13. // 此处略去 100 行
14. b.doSomething()
15. ...
16. })
17. **目录结构**：src存放所有代码源文件：逻辑层：page+view；数据层：service；工具层：util ；需要用到的图片：image。
18. **Webpack相关，本项目用的1.15.0版本**：（选择天生支持commonjs的webpack，grunt太过庞大）

设计思想：把你的项目当做一个整体，通过一个给定的主文件（如：index.js），Webpack将从这个文件开始找到项目的所有依赖文件，使用loaders处理它们，最后打包为一个（或多个）浏览器可识别的JavaScript文件。**Loader作用**：使得webpack可以对各种格式的文件进行处理，比如说分析json文件并转换为js文件，把es6文件转换为浏览器可识别的js文件，如react中jsx转换为js文件。

（**对比**：Gulp/Grunt是一种能够优化前端开发流程的工具，而WebPack是一种模块化的解决方案，不过Webpack的优点使得Webpack在很多场景下可以替代Gulp/Grunt类的工具。Grunt和Gulp的工作方式是：在一个配置文件中，指明对某些文件进行类似编译，组合，压缩等任务的具体步骤，工具之后可以自动替你完成这些任务。）

**Webpack.config.js**

entry：js的入口文件 externals：外部依赖的声明 output：目标文件

resolve：配置别名 module：各种文件，各种loader plugins：插件

**loaders**

html:html-webpack-plugin/html-loader0

js:babel-loader + babel-preset-se2015

css:style-loader + css-loader

image+font:url-loader

**webpack-dev-server**实现更新代码后自动更新浏览器

webpack打包的css如何独立成单独的css文件？安装插件extract-text-webpack-plugin

1. **工具层封装的通用js操作：**

包括网络请求（封装了jquey的ajax方法）、

获取url参数：

function getUrlParam (name){

var reg = new RegExp('(^|&)' + name + '=([^&]\*)(&|$)');

var result = window.location.search.substr(1).match(reg);

return result ? decodeURIComponent(result[2]) : null;

}

（如果 regexp 没有标志 g，那么 match() 方法就只能在 stringObject 中执行一次匹配。如果没有找到任何匹配的文本， match() 将返回 null。否则，它将返回一个数组，其中存放了与它找到的匹配文本有关的信息。该数组的第 0 个元素存放的是匹配文本，而其余的元素存放的是与正则表达式的子表达式匹配的文本。

正则表达式的子表达式匹配的文本就是被括号括起来的部分。）

字段验证（非空、手机邮箱格式判断）、统一跳转（如跳到首页、登录界面）、渲染html模板（用到Hogan）、获取服务器地址、通用提示(只是简单的用alert框来提示成功和失败)

1. **用户模块（登录、注册、找回密码、个人中心、修改密码）**
2. 用户登录模块逻辑：点击登录或按下回车—>获取用户名和密码进行表单字段的验证（用户名和密码不能为空）—>验证成功就通过\_user.login()接口，验证失败就显示错误提示。
3. 用户注册模块：类似用户登录模块。首先用户名输入框失去焦点时，如用户名不为空会做一个**异步验证**\_user.checkUsername()—>点击注册按钮或按下回车后，前端进行表单字段验证，比如不为空、格式正确等—>验证成功后，利用接口­ \_user.register（）发给后台。
4. 找回密码模块（这个模块有3个接口，注意每次发送的字段要依赖于上一个接口的返回值）：一个简单的SPA开发，把找回密码的三个步骤放在了一个页面里来实现。输入用户名点击下一步，用的\_user.getQuestion（）发送用户名成功后返回了提示问题并加载了提示问题页面—>点击输入密码提示问题答案中的下一步，用的是\_user.checkAnswer（）发送用户名、提示问题和输入的问题答案，成功后返回了token，并加载输入新密码的页面—>点击输入新密码后的按钮，用了\_user.resetPassword（）接口，发送了用户名、新密码、token，成功后显示操作成功的页面。
5. 个人中心包括了用户信息展示和信息修改两部分。信息展示部分，利用\_user.getUserInfo（）接口获取个人信息，然后用\_mm.renderHtml()实现动态数据的渲染，以及事件的委托。信息修改部分，对输入框内容做字段验证，验证成功后用\_user.updateUserInfo（）提交修改信息。
6. 修改密码模块：点击提交按钮后，前端做字段验证，成功后交给\_user.updatePassword（）接口。
7. **商品模块开发（网站首页 、商品列表页/这块最复杂、商品详情页）**
8. 网站首页的页面开发：左侧关键字导航使用ul来做，F楼层界面开发，首页banner处用了jquery轮播图插件unslider，其中前一张和后一张操作的事件绑定参考官网说明。
9. 商品列表页：点击排序按钮后，置当前页为1—>再判断排序是默认排序，还是按照价格升降排序，对应每一种排序，添加和删除对应的样式—>通过\_product.getProductList（）发送keyword、categoryId、orderBy、pageNum、pageSize成功返回后去渲染列表页面，然后加载分页信息（用到了分页技术，待会来看）
10. 商品详情页（主要3个逻辑：图片预览，count操作，加入购物车）：首先获取url参数productId，如果没有传productId, 自动跳回首页，否则渲染出详情页面。图片预览，触发mouseenter时，将小图的src赋值给大图；count操作：要注意数量最小为1，最大不超过库存数；加入购物车：点击加入购物车按钮后，将productId和数量提交\_cart.addToCart（），操作成功后显示操作成功页面。
11. **购物车模块(涉及非选中、选中、非全选、全选、数量增加、数量减少、删除操作、删除选中、去结算这几个事件)**

刚加载页面时，用\_cart.getCartList（）返回数据渲染出购物车页面，并通知导航的购物车更新数量—>用事件委托处理了商品的选择 / 取消选择, 商品的全选 / 取消全选, 删除单个商品, 删除选中商品, 提交购物车这几个事件。

1. **订单模块**

（一）添加新地址和修改已有地址共用一个弹窗（模态框），主要写下灰色阴影效果的底部如下，

/\* modal部分 \*/

.modal{

position: fixed;

top: 0;

bottom: 0;

left: 0;

right: 0;

background:#eee;

background: rgba(0,0,0,.4);

z-index: 1000;

overflow-x: hidden;

overflow-y: auto;

}

其中，background做个兼容性处理，还有左右不可滚动，垂直方向如果溢出，则提供滚动机制。

（二）省份和城市的二级联动。可以选择加载在本地也可以是从服务器读取，由于我们只有两级联动，选择加载在本地。将网上找到的省份和城市对应的json放入到cityInfo中，分别获取所有省份和所有城市，都已数组形式保存。省份变化后，获取对应城市数组，遍历该城市数组，选用带选项的选择列表<select><option></option></select>来作为城市的html。

（三）获取表单里收件人的信息，并做表单的验证（比较简单了）。使用新地址且验证通过则提交\_address.save（）接口。

★ 人工智能信用评级管理系统 Web前端开发

项目介绍：

用于贵阳财政厅进行人工智能信用评级的系统，能够对地方级各商户的信用评级做预测，为单位自身提供资信信息和决策参考。系统主要包括模型管理、算法管理、数据配置、指标库管理、信用评级、企业信息、算法管理、模型训练8 个模块。

项目职责：

1. 在UI设计师的配合下，完成主页面和各模块页面的样式优化；（不要说是bootstrap，说是公司自己的样式）
2. 项目呈现方式以表格为主，涉及的主要js操作包括：表单的同步/异步验证，数据安全性处理，城市、区域、行业级联操作，复杂表单回填等。
3. 在团队合作下按预期完成了该项目的第一个版本。
4. **团队组成：3个前端，2个UI，3~4个后端。去时项目已完成了一半以上。然后遇到一个展览会耽搁了一段日子，后辞职了。**
5. **公司如何管理代码的？**

* 基于物联网的实验设备智能管理系统 Web前端开发

项目介绍：

     底层使用蓝牙、RFID等物理技术实现对高校实验室内所有设备的智能感知，从而数据库存有所有有效设备的信息。用户通过浏览器登录界面登录，后台判断出用户等级从而呈现出具有不同功能模块的网页界面。主要面向3类用户：普通师生可以查询设备、借用设备、申请维修设备，设备管理员负责跟踪维修进展、申购设备，后台管理页面能够查看所有用户的设备使用情况。前后端部分分离式开发，利用ajax+jsp+servlet+MySQL等技术完成网站的功能实现。

项目职责：

1. 规划网站的基本架构，整理所需素材，针对普通师生用户的页面完成了设备查询和设备预约等功能模块，主要包括表单数据的读取、验证和提交，使用qunee组件展现网络设备间的拓扑关系。
2. 小组队长，负责项目阶段性任务总结，跟进项目进展。

**团队组成：1个底层实现，2个网页开发，2个后台、数据库的开发。**

1. 现在师妹师弟们，已经在实现用二维码识别设备，从手机终端上操作借用设备等的操作了。
2. **部分分离开发方式（缺点），**后端负责页面模板（jsp）,本地开发环境搭建成本较高，更新页面模板仍需后端协助，效率不够高，需要前后端同时发布

* 基于移动无线Mesh网络的智能小车应急通信系统 软件开发

项目介绍：

以团队名义参加竞赛，该项目针对灾后救援人类无法进入、通信不佳的场景，利用多个智能车进入救援区后自组织搭建的Mesh 网络提高网络覆盖范围，将智能车上搭载摄像头采集到的视频传输到指定服务器，供救援人员了解灾后情况并作对应部署。项目主要分为几个模块：基于Andriod平台开发智能小车路径规划系统实现小车按照APP中规划的路径移动、基于OpenWrt开发视频传输系统实现搭载摄像头采集视频，以及多智能小车联结形成的Mesh网络为视频传输提供良好的传输通道。

项目职责：

1. 负责基于智能小车上的无线Mesh节点开发能够根据所在不同环境情况进行最优路径传输的智能组网系统。开发板采用OpenWrt系统，将原始AODV路由协议的跳数最少路由策略，变为由控制器统一分配路由。具体地，分别对AODV协议的HELLO机制、RREQ机制、RREP机制等进行改进，使它适用于带有集中式控制器的Mesh网络系统。
2. 利用Visual Studio与GCC等开发环境开发节点和控制器程序，实现系统组网，资源存储，路由决策等功能。
3. 合并本模块和其他小组成员的模块，共同完成了整个系统的搭建、测试和分析。
4. **开发板是基于OpenWrt系统**，可后刷（可以被描述为一个嵌入式的 Linux 发行版，主流路由器固件有 dd-wrt,tomato,openwrt三类）
5. **为什么用到一个协调节点？**由于想要提升视频传输质量，时延更小，就想到用一个协调节点（控制中心节点）作为总的决策者，能够收集节点和网络状态信息，它能做综合判断该选择哪些节点进行传输（便于理解，可以提到集中式和分布式的概念）
6. **如何让协调节点收集到各种网络信息？（即改进HELLO机制、RREQ机制、RREP机制等）**

（一）**HELLO报文原始**作用主要是周期性地广播报文，用来探测并维护邻居节点关系，通过接收邻居节点周期性发送过来的HELLO报文来来判断该节点与邻居节点这条链路通与断。

**现利用HEELO的特性来现获取节点间时延和丢包率：时延方法**：终端甲将报文发送给终端乙，报文中带有终端甲发送报文时的时间戳，终端乙收到该报文后，立刻反馈该报文给终端甲，终端甲收到报文后，比较接收反馈报文时间戳和报文中记录的发送时间戳，便可以获取到终端甲到终端乙的时延，该方法不要求两个节点在时间上严格同步。**两个终端间的丢包率方法：**终端甲向终端乙周期性地发送报文，终端甲每发送一次报文，就把报文ID加一，报文被发送的时候，会携带报文ID，一起发送给终端乙；终端乙在收到终端甲发送过来的报文后，先记录该报文的报文ID，然后统计在过去的一段时间内，报文丢失的数量。考虑到存储空间的利用率，可以用一个循环数组来存储这些报文信息。将最大的报文ID减去最小的报文ID，然后除以中间报文的丢失数量，便可以获取到终端甲到终端乙的丢包率。

（二）**RREQ（路由请求报文）**：当节点要发送一个数据包时，在查找路由表之后，并没有找到到达目的地址的有效路由，此时就会触发RREQ机制，进行路由发现请求，即广播RREQ报文去查找路由。本文将RREQ机制中的路由请求分为两种：一种为目的地址为网络控制器的路由请求；另一种为目的地址为其他普通节点路由请求。目的地址为网络控制器的路由请求，还是使用原来的RREQ机制来发现路由，即广播RREQ报文。而目的地址为普通节点的路由请求，则节点将路由请求发送给网络控制器，由网络控制器来对该路由请求做出应答。

（三）**RREP（路由请求应答报文）：**在原始的AODV路由协议中，RREP机制主要用来回应源节点广播的路由请求。在改进后的RREP机制中，仍然保留了这个功能，用来回应目的地址为网络控制器的路由请求。在这个基础上，本文在改进后的RREP机制中添加了构建路由的功能。当源节点收到网络控制器返回的路由请求应答，便开始了从源节点到目的节点路由的构建。源节点会根据收到的路由请求应答报文，将该报文传递给到目的节点的路径上的所有中间节点。目的节点收到该报文后，就会返回一个确认报文给源节点。源节点收到这个确认报文后，就开始将缓存的数据包发送给目的节点。

自我评价

★ 热爱前端，善于记录工作中遇到的零碎问题，定时总结归纳记录于个人技术博客，博客正不断完善中；

★ 主动性强， 逻辑思维能力和理解能力较好，注重细节；

★ 为人诚恳，乐观，善于和同事沟通，营造和谐的工作氛围；

★ 此外业余爱好广泛，有个人特长，二胡九级。

**描述一下自己在校研究方向：**偏理论研究，主要是基于一种数学理论（SNC）对物联网（提倡智能化，不需要人为干涉，比如智能抄表，带无线模块的水电气表通过网络传输给数据管理中心，这是物联网中的一种业务；家庭防盗系统，通过传感器监测到威胁而自动报警，这是另一种业务；又例如，小黄车也属于物联网范畴，它现在的智能锁用的是NB-IOT技术，能提高信号覆盖范围，比4G的范围还广，其他的共享单车可能还在用2G网络，当前蜂窝比如用户锁车后由于网络覆盖不足使得车辆状态无法反馈到云端进而迟迟无法结束计费和释放该车辆）环境下的业务流进行分析，并且针对具体的网络场景分析业务流穿过网络需要的时延等信息。为其他人研究如何改进网络传输技术提供依据。

**前端遇到最难的问题：**页面设计以及兼容性这些问题都是有通性的，一般在网上都能找到解决方法。难的问题是思考交互逻辑怎么写，思路要清晰，以及怎样优化自己的代码，不至于写出冗余的代码，以最少的代码写出逻辑。