序号___05__

华南理工大学广州学院
2019-2020学年度第1学期

课程名称：JavaEE框架应用开发项目实践

题　　目：校园帮

分　　工：前端开发、前后端接口设计、需求分析、界面设计

专业班级：软件工程四班

年　　级：2017级

姓　　名：刘健婷

学　　号：201710098026

成绩：

评语：

目录

一、引言

我们队名叫咸鱼队，选择的项目是校园帮——这是一个为了给在校学生的一个向别人求助，帮助别人的平台。而我在其中主要负责前端开发。

我们的组员有：

项目经理：关小梅（17 软 3）；

测试：翁春怡（17 软 3）；

后端：林怡伉（17 软 3）、 刘铁梅（17 软 4）；

前端：刘健婷（17 软 4）；

我们的项目选择于上学期的期末，确立于这学期的开始。由于课程和自身的原因，实际花费时间大约两个星期左右。

首先我想肯定一下我的组员，很庆幸，我们有一个很良好的学习工作氛围。我们组员的积极沟通、善解人意确实是为这个项目进程和组员各自的情绪带来了巨大的好处。正是我们这 5 个人的付出和努力，才有这个颇有凝聚力的咸鱼队。

但同时也由于组员内都是很平凡普通的学生，这也意味着我们要完成好一个项目，就比不得缺少努力专研的精神和坚持不懈的努力，很庆幸，我们的组员都做得还不错。尽管到最后，项目没有达到理想之中的美好，但我说实话，这对一开始的我们来说，已经是一个很大的进步了。

当然，这个项目还有很大的能改进的空间，但由于时间和自身原因，这个项目不能说是很满意，但我个人觉得，这个项目带来的，已经满满的超过一个项目的呈现结果了，更多的是项目之外带来的收获。因此，我很感谢这个项目。

最后，也谢谢老师们对我们项目的评定。

二、原型设计、页面设计

1、原型设计

welcome 页面：一个大 div 介绍校园帮的使用方法，页面包含登录注册按钮。

登录注册页面：是两个分开的页面，登录页面包括返回 welcom 页面和注册页面链接。

首页页面：包含一个菜单栏，其中包括：个人姓名显示，账号信息页面链接，退出登录链接（即返回到 welcome 页面并退出），发布任务模态框，用户所接受的任务链接，用户所发布的任务链接和一个“关于我们”的一个介绍。主面板的内容则是放整个所有任务的列表，其可根据任务类型、关键字查询，并可进行查看详情任务和接取任务。每个具有能看到任务信息的页面都具有分页功能，每个具有能看到任务信息的页面都具有根据关键字查询的功能。

账号信息页面：显示除了密码的所有用户个人信息。

发布任务模态框：点击菜单栏则会在当前页面弹出一个模态框，给用户提供简便的发布任务功能。里面包含两个输入框分别是任务标题和任务内容，和一个类型筛选下拉框，一个发布按钮和取消按钮。

任务详情内容模态框：点击接取首页的某个任务，即会弹出接取任务的模态框，里面会

有任务的详细内容。

我接受的任务页面：包含三个页面，以任务状态来区分去查看相对应的任务。分别是“已接取”、“待被确认完成”、“任务已结束”。这三个页面都是查看相对应状态的任务的。而其中的“已接取”页面有一个“确认完成”的按钮，给用户提供确认完成所接取的任务。

我发布的任务页面：包含四个页面，以任务状态来区分区查看相对应的任务。分别是“待被接取”、“已被接取”、“也被完成”、“任务结束”。而“取消任务”页面提供“取消任务”按钮给用户提供取消用户的功能，“已被完成”页面提供“确认完成”按钮给用户提供确认任务完成使任务最终结束的功能。

每个页面都具有相同的菜单栏，在菜单栏的底部即每个页面的左下角都有一个关于我们的字样。该菜单栏可以伸缩，停留在不同的页面菜单栏具有不同的变化。

下图为此项目的功能分解的原型设计图（见图 1）

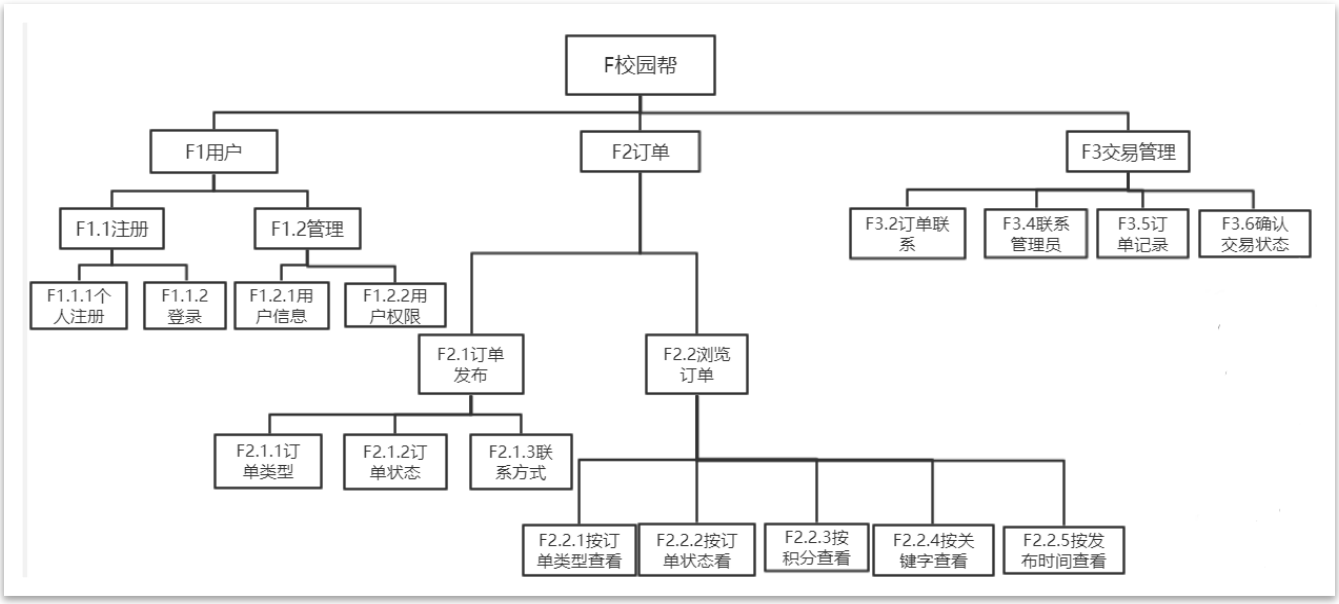


图 1：功能分解的原型设计图

2、页面设计

下面展示几个页面的原型设计图：

（见图 1：首页原型设计图（1.0）、 图 2 首页原型设计图（2.0））

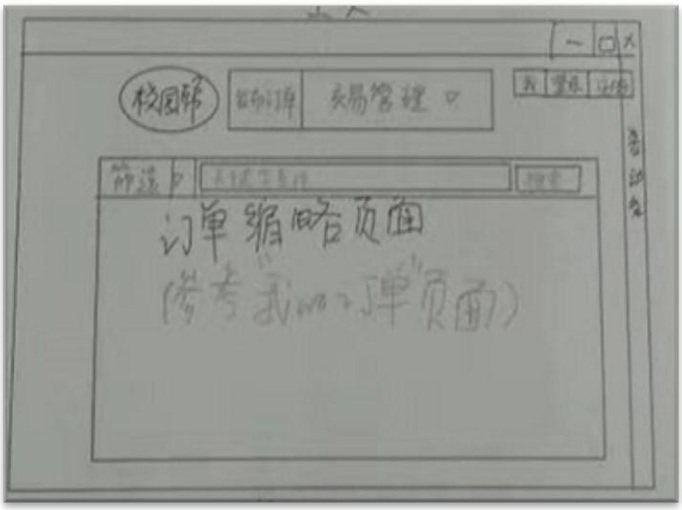


图 1：首页原型设计图（1.0）

图 2：首页原型设计图（2.0）

下面展示几个成果页面：

由于考虑到篇幅长度，所以只是截取了部分的页面。具体页面可移步到浏览器搜索 liouer.top 浏览。由于后台没有上传到服务器，所以暂时只能浏览一个没有数据和部分功能也不可用的页面。

除了首页，其他能看到任务的页面都采用了 table 来存放任务，一是为了更方便的查看任务，二是为了方便的操作，当然界面则会略低于首页，但个人觉得还可以更美化。

其次结合了实际使用，进而使用了模态框来显示发布任务和接取任务，这使得用户界面内更加的简单且人性化，提升用户体验感。

（见图 3：welcome 页面、图 4：登录页面、图 5：首页、图:6：发布功能的模态框、图 7：“我发布任务中”的“待被接取”页面）

图 3：welcome 页面

图 4：登录页面

图 5：首页

图 6：发布模态框

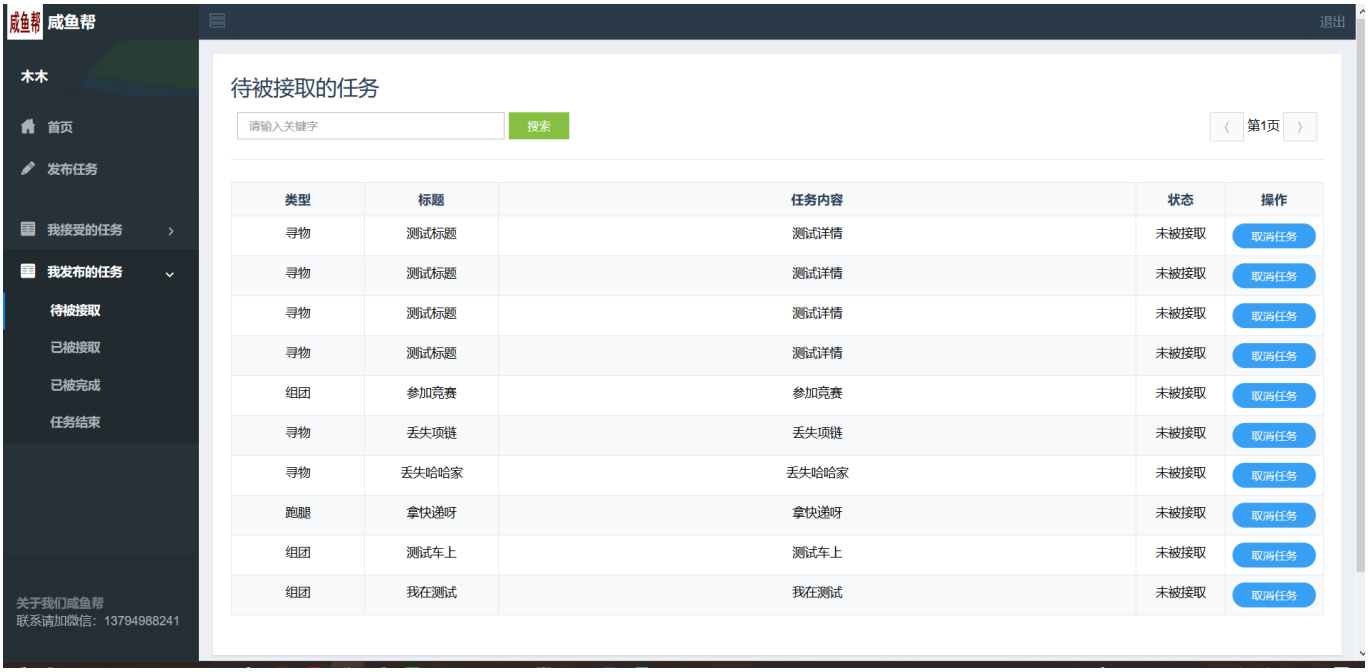


图 7：我发布任务中的待被接取页面

三、前后端交互接口设计

在课堂上，我们组内成员一起讨论过前后盾交互接口并进行了设计。但是由于没接触过过项目开发，组员们都对前后端交互接口的设计理解有偏差，所以没多大用得上。

随着开发的逐渐深入，经过前后端水深火热的讨论之后，最终得出此前后端交互接口的设计，但由林怡伉负责编写：

1.登陆接口

1. 接口描述

输入账号密码，查询 user 表中是否存在该用户，是则返回成功，否则返回失败

请求方式	GET
请求 url	http://localhost:8081/schoolhelp/userController/login.do
具体功能	获取账号密码

2. 参数说明

参数名	是否必传	参数描述	枚举值列表
uName	是	用户名	

uPassword	是	密码	

3. 返回结构

```
"user": {
  "uName": "201710098000"
  “name”：“莲藕”
}
```

2.注册接口

1. 接口描述

输入账号，密码，学院，专业，班级，姓名，联系方式注册，查询 user 表中是否存在，不存在则注册成功

请求方式	GET
请求 url	http://localhost:8081/schoolhelp/userController/addUser.do
具体功能	接受用户基本信息

2. 参数说明

参数名	是否必传	参数描述	枚举值列表
uName	是	学号	
uPssword	是	密码	
academy	是	学院	
major	是	专业	
grade	是	班级	
name	是	姓名	
tel	是	电话号码	

3. 返回结构

```
{
  "status": “200”,
  "message": "success",
}
```

3.获取用户信息接口

1. 接口描述

前端传入账号，查询 user 表中的信息并返回一个 User 类型的数据

请求方式	Post
请求 url	http://localhost:8081/schoolhelp/userController/queryUser.do
具体功能	获取用户基本信息

2. 参数说明

参数名	是否必传	参数描述	枚举值列表
uName	是	学号	

3. 返回结构

```
"user": {
        "uName": "201710098000",
        “name”：“刘备”，
        “academy”：“计算机专业”，
        “major”：“软件工程”
        “grade”：“3 班”
        “tel”：“15816100000”
}
```

4.分页根据类型查询任务接口

1. 接口描述

前端传入页码，任务类型，关键字查询 mission 表中相关信息并返回一个 PageBean<Mission>类型的数据

请求方式	Get
请求 url	http://localhost:8081/schoolhelp/missionController/queryType.do
具体功能	分页获取各种类型的任务信息以及关键字搜索

2. 参数说明

参数名	是否必传	参数描述	枚举值列表
pageNo	否	页码，默认 1	

missionType	否	任务类型，默认全部	全部/组团/寻物/闲置/跑腿/资源共享
Word	否	关键字	

3. 返回结构

```
“pageBean”：{
    “mission”:[{
      “missionNo”：1
      “missionTitle”:急急急！
      “uName”:“201710098000”
      “missionType”：“寻物”
      “missionDetail”：“在三饭丢了饭卡，捡到请联系我：xxx”
      }]
  “totalCount（总任务数）”:”1000”
  “totalPage（总页数）”:100
}
```

5.分页查询个人所发布过的任务接口

1. 接口描述

前端传入页码，账号，任务状态查询 mission 表中相关信息并返回一个 PageBean<Mission> 类型的数据

请求方式	Get
请求 url	http://localhost:8081/schoolhelp/missionController/querySelf.do
具体功能	查询个人发布未被接取的任务/查询个人发布已被接取的任务/查询个人发布已被完成的任务

2. 参数说明

参数名	是否必传	参数描述	枚举值列表
pageNo	否	页码	
uName	是	发布者（用户名）	
missionStatus	是	发布者所能看到的任务状态	未被接取/已被接取/已被完成/结束任务

3. 返回结构

{
“pageBean”：{
“mission”:[{
“missionNo”：1
“missionTitle”:急急急！
“uName”:“201710098000”
“missionType”：“寻物”
“missionDetail”：“在三饭丢了饭卡，捡到请联系我：xxx”
}]
“totalCount（总任务数）”:”1000”
“totalPage（总页数）”:100
}

6.分页查询个人接取过的任务接口

1. 接口描述

前端传入页码，账号，任务状态查询 mission 表中相关信息并返回一个 PageBean<Mission>类型的数据

请求方式	Get
请求 url	http://localhost:8081/schoolhelp/missionController/querySelfNOrYReceive.do
具体功能	查询个人已接取的任务/查询个人已完成的任务/查询个人已被确认的任务

2. 参数说明

参数名	是否必传	参数描述	枚举值列表
pageNo	否，默认 1	页码	
uName	是	接取者（用户名）	
missionStatus	是	接取者所能看到的任务状态	已接取/已完成/已被确认

3. 返回结构

“pageBean”：[{
“mission”:[{
“missionNo”：1
“missionTitle”:急急急！
“uName”:“201710098000”
“missionType”：“寻物”

“missionDetail”：“在三饭丢了饭卡，捡到请联系我：xxx”
}]
“totalCount（总任务数）”:”1000”
“totalPage（总页数）”:100
}

7.发布任务接口

1. 接口描述

前端传入账号，任务标题，任务类型，任务详情，任务类型，任务人数，对 mission 表中添加一条数据

请求方式	Get
请求 url	http://localhost:8081/schoolhelp/missionController/addMission.do
具体功能	删除未被接取的任务

2. 参数说明

参数名	是否必传	参数描述	枚举值列表
uName	是	学号	
missionTitle	是	任务标题	
missionDetail	是	任务详情	
missionType	是	任务类型	寻物/组团/跑腿/闲置/资源共享/其他
count	否	任务人数	

3. 返回结构

成功

{
“message”：“success”
“status”：“200”
}

失败

{
“message”：“faile”
“status”：“201”
}

8.接取任务接口

1. 接口描述

前端传入任务号，账号判断是否为自己发布自己接取，已接取过，任务人数是否足够，是则对 mission 表中符合条件的任务进行修改，其状态由未被接取改为已被接取。对 receive 表中添加一条数据

请求方式	Get
请求 url	http://localhost:8081/schoolhelp/missionController/receiveMission.do
具体功能	判断该任务人数是否足够，该用户是否接过该任务，是则接取任务

2. 参数说明

参数名	是否必传	参数描述	枚举值列表
missionNo	是	任务号	
uName	是	接取者（用户名）	

3. 返回结构

成功

```
{
    “message”：“success”
    “status”：“200”
}
```

失败

```
{
    “message”：“faile”
    “status”：“201”
}
```

9.完成任务接口

1. 接口描述

前端传入任务号，账号并查询 receive 表中数据，对 receive 表中将符合条件的数据进行修改，其状态由已接取改为已完成。如果 mission 表中的 count 等于 receive 表中符合条件的数据，则中对 mission 表中符合条件的任务进行修改，其状态由已被接取改为已被完成。

请求方式	Get

请求 url	http://localhost:8081/schoolhelp/missionController/completeMission.do
具体功能	修改接取者任务状态，并判断该任务完成人数是否足够，是则修改发布者状态已被完成

2. 参数说明

参数名	是否必传	参数描述	枚举值列表
uName	是	接取者（用户名）	
missionNo	是	任务号	

3. 返回结构

成功

```
{
    “message”：“success”
    “status”：“200”
}
```

失败

```
{
    “message”：“faile”
    “status”：“201”
}
```

10.结束任务接口

1. 接口描述

前端传入任务号，对 misison 表中符合条件的数据进行修改，其状态由已被完成改为结束任务

请求方式	Get
请求 url	http://localhost:8081/schoolhelp/missionController/missionProfile.do
具体功能	发布者确认任务已完成，状态改为任务结束 接取者任务状态改为已被确认

2. 参数说明

参数名	是否必传	参数描述	枚举值列表

missionNo	是	任务号	

3. 返回结构

成功

```
{
    “message”：“success”
    “status”：“200”
}
```

失败

```
{
    “message”：“faile”
    “status”：“201”
}
```

11.删除任务接口

1. 接口描述

前端传入任务号，对 mission 表中符合条件且该任务状态为未被接取的数据进行删除

请求方式	Get
请求 url	http://localhost:8081/schoolhelp/missionController/deleteMission.do
具体功能	删除未被接取的任务，包括图片和其他信息

2. 参数说明

参数名	是否必传	参数描述	枚举值列表
missionNo	是	任务号	

3. 返回结构

成功

```
{
    “message”：“success”
    “status”：“200”
}
```

失败

```
{
    “message”：“faile”
    “status”：“201”
}
```

四、前端工作

1、使用技术

开发语言：HTML、css、JavaScript、jQuery；
开发环境：Window；
编译器：WebStorm；
使用框架：Bootstrap；
使用前后端分离技术；
使用阿里云服务器，将前端的文件上传到阿里云服务器。

2、功能模块规划

网页 1.0 版本
1. 用户可以注册账号
2. 用户可以根据已注册的账号登陆
3. 用户可以登陆后查看自己的账号信息
4. 用户可以根据类型查询任务
5. 用户可以查询自己接取未完成的任务
6. 用户可以查询自己已完成的任务
7. 用户可以查询自己发布过未被接取的任务
8. 用户可以查询自己发布过已被接取的任务
9. 用户可以查询自己发布过已被完成的任务

网页 2.0 版本
1. 用户可以注册账号
2. 用户登录或者注册前可以看到 welcome 页面
2. 用户可以根据已注册的账号登陆
3. 用户可以登陆后查看自己的账号信息
4. 用户可以分页根据类型查询任务
5. 用户可以分页查询自己接取未完成的任务
6. 用户可以分页查询自己接取已完成的任务
7. 用户可以分页查询自己接取已被发布者确认任务结束的任务
8. 用户可以分页查询自己发布过未被接取的任务
9. 用户可以分页查询自己发布过已被接取的任务
10. 用户可以分页查询自己发布过已被完成的任务
11. 用户可以分页查询自己发布过已结束的任务
12. 用户可以接取任务
13. 用户可以在接取过未完成的页面中点击按钮完成任务
14. 用户可以发布任务

15. 用户可以在删除还没被接取的任务
16. 用户可以查看任务详情
17. 用户可以再发布已被完成的任务页面中点击按钮结束任务
18. 用户可以退出登录返回到 welcome 页面

3、代码目录/层次结构

见下图 7：代码目录、图 8：html 文件夹下的 html 文件、图 9 ：js 文件夹下的 js 文件

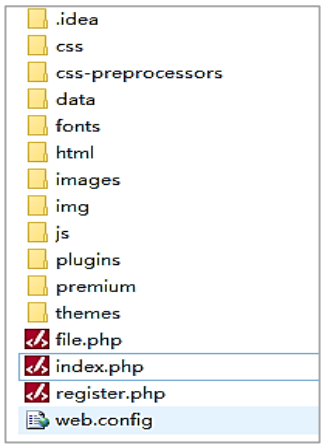


图 7：代码目录

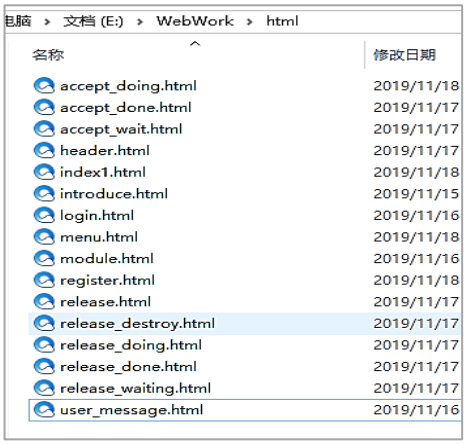


图 8：html 文件夹下的 html 文件

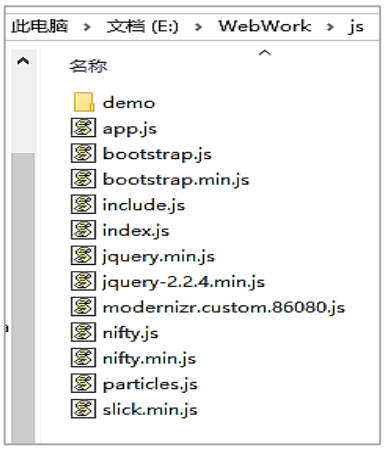


图 9：js 文件夹下的 js 文件

html 文件夹放的是项目的所有 html 文件；其余都是框架的所需要的文件夹，里面包含很多 css、js 文件。下面举几个例子：

首先是引用文件，里面包含了前端开发所需要的所以外部和内部引用的文件，其中有包括 js 文件夹里的各种 js 文件（图 10）和 css 文件夹里的各种样式文件（图 11），里面还包含了自己所写的 css（图 12）。

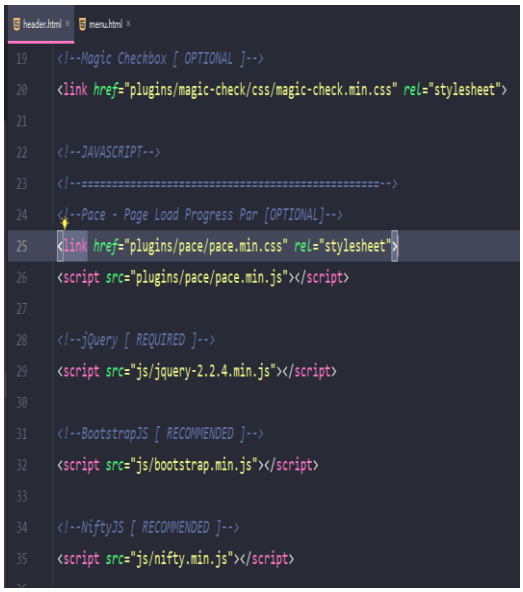


图 10：引用 js 文件

图 11：引用 css 文件

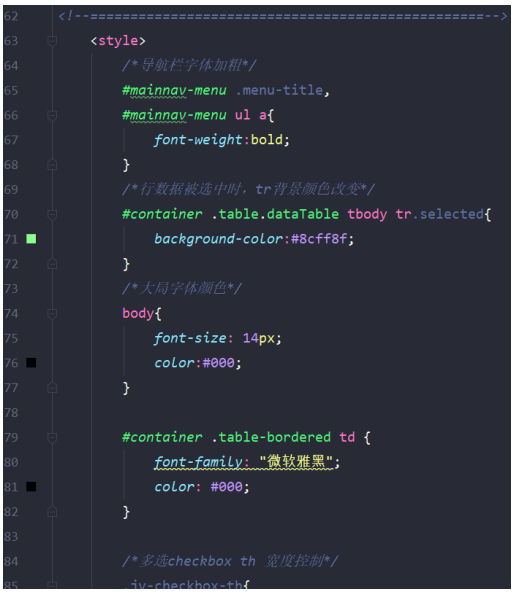


图 12：编写 css

img 文件夹放项目所需用到的图片素材（见图 13）：

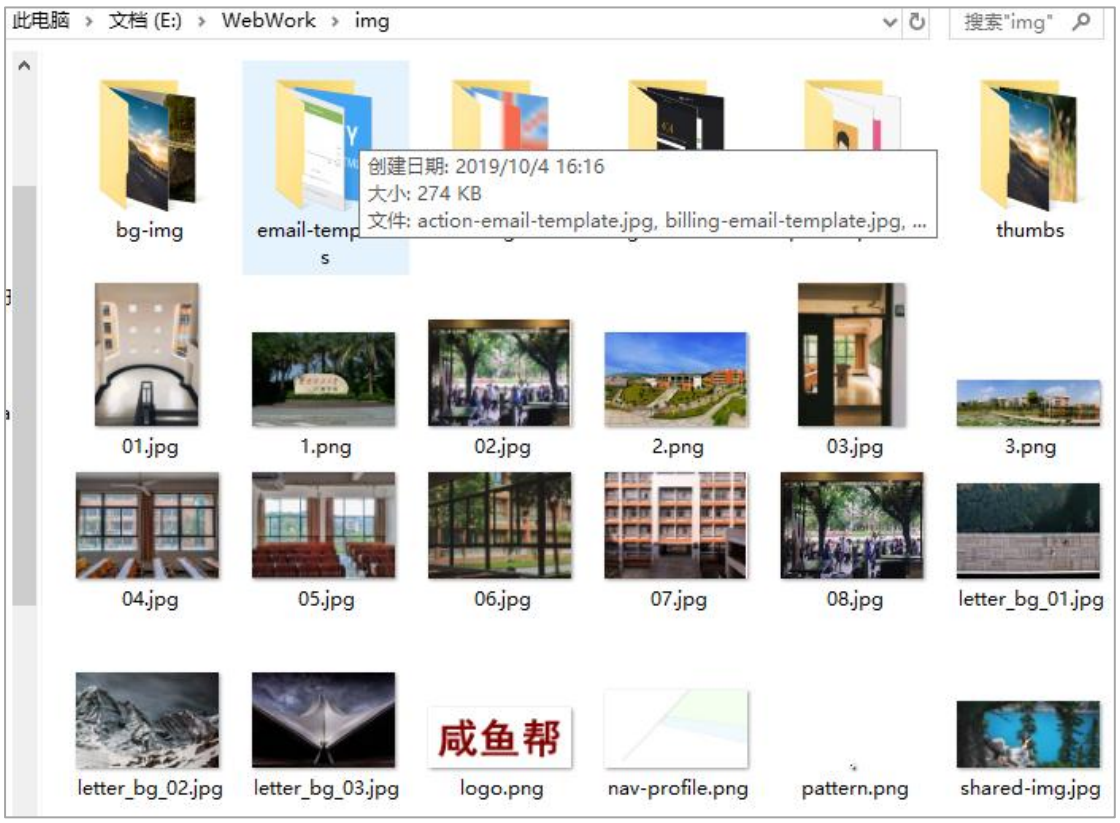


图 13：img 文件夹下的素材

html 文件下的 html 文件：

html 文件目录的解释（目录文件见图 8）,由于考虑到篇幅，我就不一一把每个界面都截图了，详情可到 liouer.top 上浏览，但后台没在服务器上，因此只能浏览一个没有数据的页面。

accept_doing：我接受的任务:已接取状态的任务列表页面

accept_done:我接受的任务：任务已结束的任务列表页面

accept_wait:我接受的任务：待被确认完成的任务列表页面

index1.heml:首页

introduce.html:welcome 页面

login.html:登录页面

menu.html:菜单

module.html:为这个项目做的一个页面 module

register.html:注册页面

release_destroy.html:我发布的任务：任务结束的任务列表页面

release_doing.html:我发布的任务：已被接取的任务列表页面

release_done.html:我发布的任务：已被完成的任务列表页面

release_waiting.html:我发布的任务：待被接取的任务列表页面

user_message.html:用户账号信息页面

这个项目我采用的是写一个 menu.html，其在页面上是充当每个页面的菜单（如图 14），所以每个页面都会引用这个 menu.html 文件。拿两个功能分析一下，一个是伸缩菜单的按钮，是引用了 css 文件的 nifty 的某个类（如图 15），一个是菜单的高亮显示（如图 16）。因为发布功能是在每个页面都能使用的模态框，所以把它也放在了 menu.html 里（代码如图 17）。

图 14：阴影部分为菜单部分

```
<ul class="nav navbar-top-links pull-left">
    <!--伸缩导航栏按钮-->
    <!--~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~-->
    <li class="tgl-menu-btn">
        <a class="mainnav-toggle" href="#">
            <i class="demo-pli-view-list"></i>
        </a>
    </li>
    <!--~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~-->
    <!--End 伸缩导航栏按钮-->
</ul>
```

图 15：伸缩栏按钮的代码

```
<!-- 菜单区别 -->
<script>
    var s_url=window.location.pathname;
    //根据文件夹路径来改
    //s_url = s_url.replace("/WebWork/","");                    //在webstorm里打开运行
    s_url = s_url.replace("/","");                              //在服务器上运行
    //alert(s_url);
    var now_url = '';
    for(var i = 0;i<$("#mainnav-menu li").length;i++){
        now_url = $("#mainnav-menu li a").eq(i).attr("href");
        if(now_url == s_url){
            $("#mainnav-menu a").eq(i).parent().addClass("active-link");
            $("#mainnav-menu a").eq(i).parent().parent().parent().addClass("active-sub");
            $("#mainnav-menu a").eq(i).parent().parent().parent().parent().parent().addClass("active");
            $("#mainnav-menu a").eq(i).parent().parent().addClass("in");
        }else{
            $("#mainnav-menu a").eq(i).parent().removeClass("active-link");
        }
    }
</script>
```

图 16：菜单选中高亮的代码

再分析一下分页功能的代码，这里拿取 accept_doing.html（我接受的任务：已接取的页面）文件来举例（如图 18:），#tou 是上一页的控件，点击会去判断如果当前页为 1，则会将页码原本要变成 0 改成 1，数据不变：

```
//获取当前页
var getPageNo = $('.ye').text();
var userId = localStorage.getItem("userLocalId");

//点击页码处理
$('#tou').click(function () {
    getPageNo = $('.ye').text() - 1;
    console.log("当前页：",getPageNo);
    if (getPageNo<1){
        getPageNo = 1;
        console.log("<1当前页：",getPageNo);
    }else {
        $('.ye').text(getPageNo);
        getAcceptMission(getPageNo);
    }
});
```

图 18：“上一页”点击事件代码

紧接着，我们分析一下模态框的使用，下面这个是任务详情内容的模态框的 body 代码，它有模态框的 header，body，footer，这里只截关键部分 body，在模态框里用 form 展示数据。这是利用了 bootstrap 框架里面的模态框，个人感觉非常好用，并且看起来很大方美观（如图 19）。

```
<!--Modal body-->
<div class="modal-body">
    <div class="panel-body">
        <form method="post" id="detail_form" class="form-horizontal" enctype="multipart/form-data">       <!--enctype:
            <input type="hidden" name="missionDetailNo">      <!--enctype：对表单数据进行编码-->
            <!-- <input type="hidden" name="orderNo"> -->
            <div class="form-group">
                <div class="col-md-7 mar-btm">任务标题
                    <input  id="title" name="title" type="text" class="form-control"disabled="">
                </div>
                <div class="col-md-3 mar-btm">任务类型
                    <input id="orderType" name="orderType"  class="form-control"  disabled="">
                </div>
                <div class="col-md-2 mar-btm" style="...">已接人数
                    <input  id="userCount" name="userCount" type="text" class="form-control" disabled="">
                </div>
                <div class="pad-ver bord-ver">
                    <p style="..." id="missionDetail"></p>
                </div>
            </div>
        </form>
    </div>
</div>
```

图 19：任务详情模态框 body 代码

最后，看一下利用 ajax 传递数据的代码，下面举一个获取任务详情接口代码（图 20）：

```
var ddd;
// 详细任务显示和接口
function showDetail(missionNo) {
    ddd = missionNo;           //把后台的order_id赋值给orderNo之后再赋值给这个放在接单模态框的已经hidden的名为orderNo的input里
    var url = "http://localhost:8081/schoolhelp/missionController/query.do";
    console.log("模态框里的missionNo:",missionNo);
    $.ajax({
        url:url,
        type:"get",
        dataType:"json",
        data:{
            missionNo:missionNo
        },
        success:function (data) {
            console.log("data:",data);
            var jie = data['countReceive'];
            var gong = data['count'];
            $('input[name=\'title\']').val(data.missionTitle);
            $('input[name=\'orderType\']').val(data.missionType);
            $('input[name=\'userCount\']').val(jie + '/' + gong);
            document.getElementById("missionDetail").innerHTML = data.missionDetail;
            $('#detail_window').modal("show");
        },
        error:function (data) {
            var jie = data['countReceive'];
            var gong = data['count'];
            console.log("打开详细任务失败的data",data);
            console.log("标题",data.missionTitle);
            console.log("类型",data.missionType);
            console.log("人数",jie + '/' + gong);
            console.log("内容",data.missionDetail);
            alert("打开信息任务失败");
        }
    });
}
```

图 20：获取任务详情接口代码

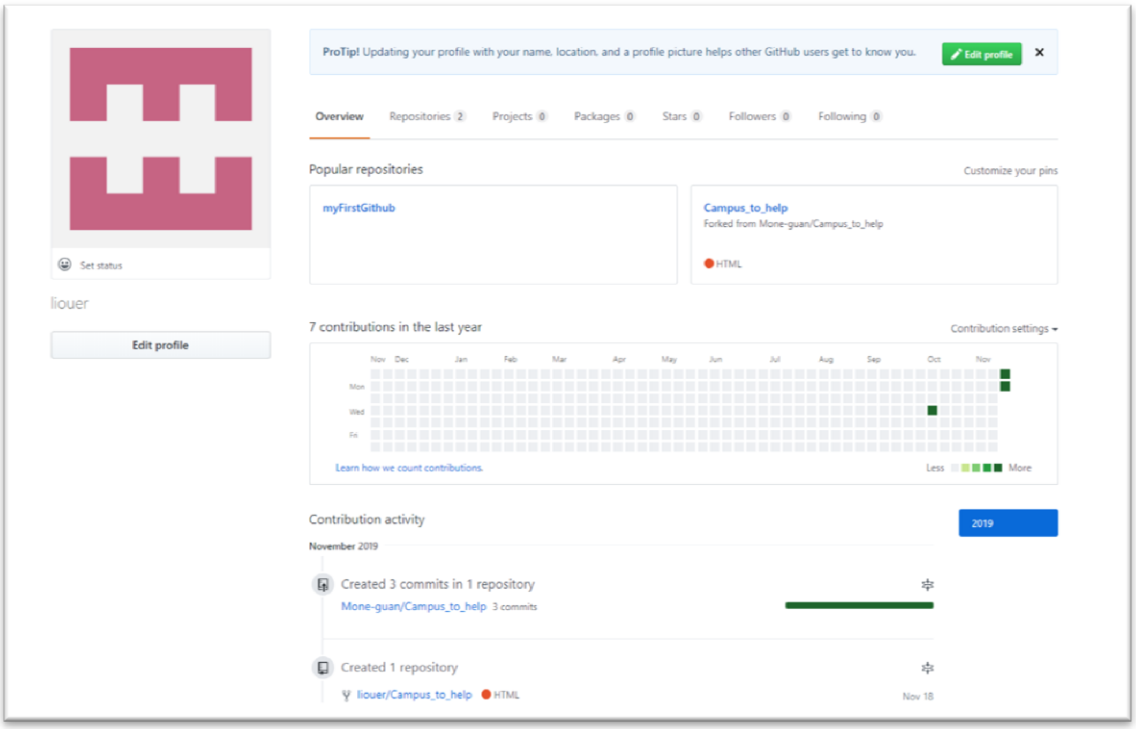
五、管理工作

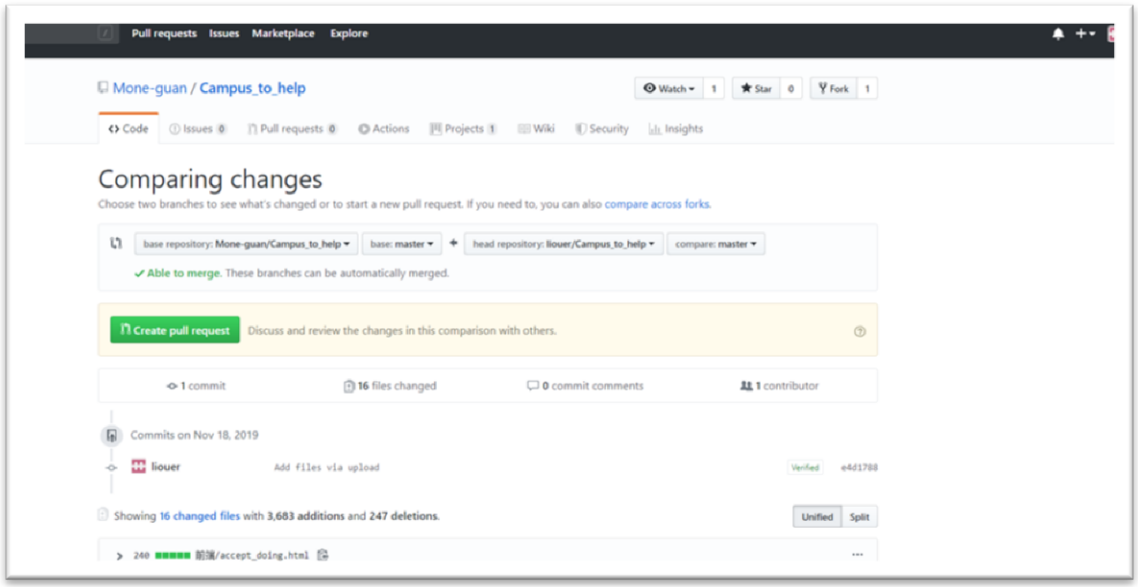
当天工作完的文件都会被保存好，并上传到阿里云服务器。

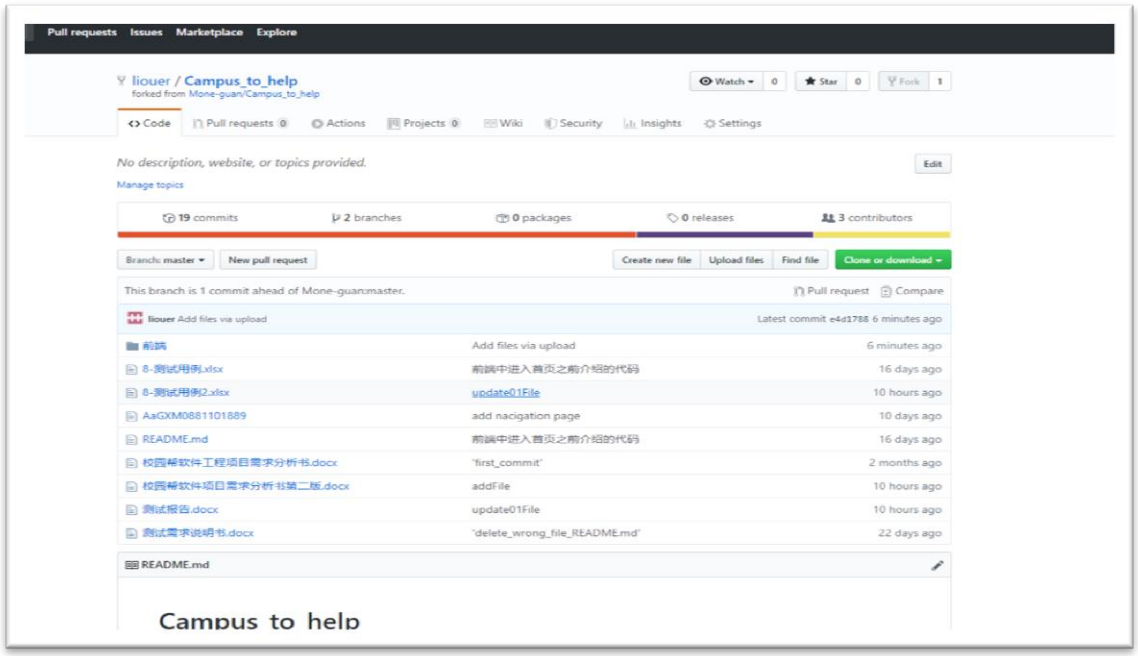
在进行工作的时候，会计划好当前需要解决什么问题，以及解决了一个问题之后及时记录。

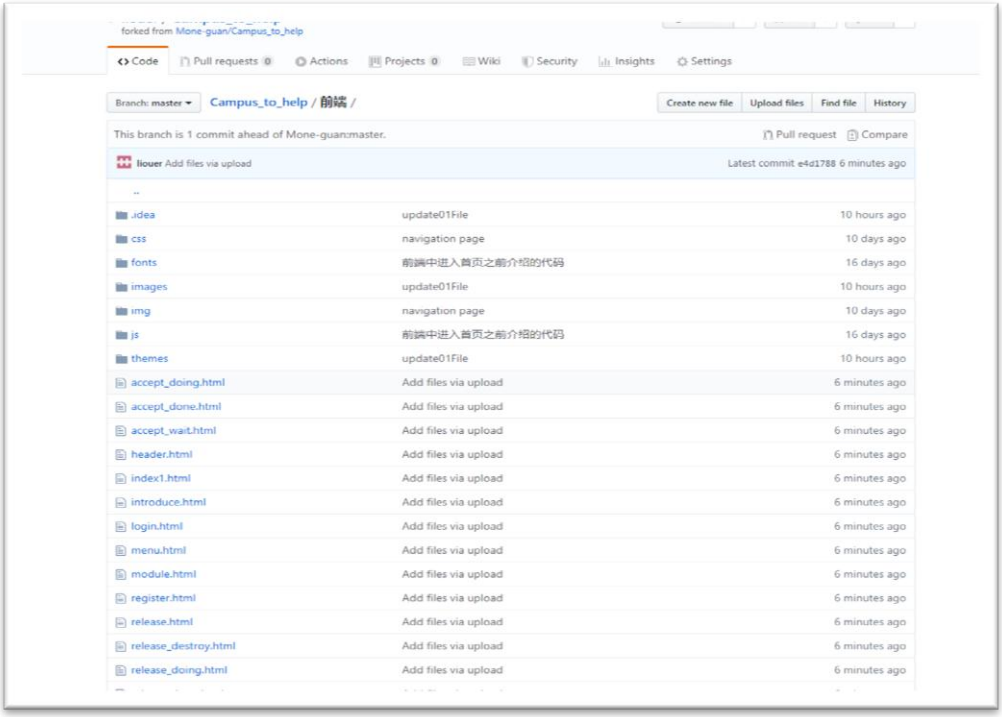
六、版本控制报告

专业	软件工程	年级	2017	班级	4	小组编号	05
项目名称-项目经理	校园帮-关小梅						
项目成员	刘健婷						
承担角色	前端开发、需求分析						
成员学号	201710098026						
成员班级	4 班						

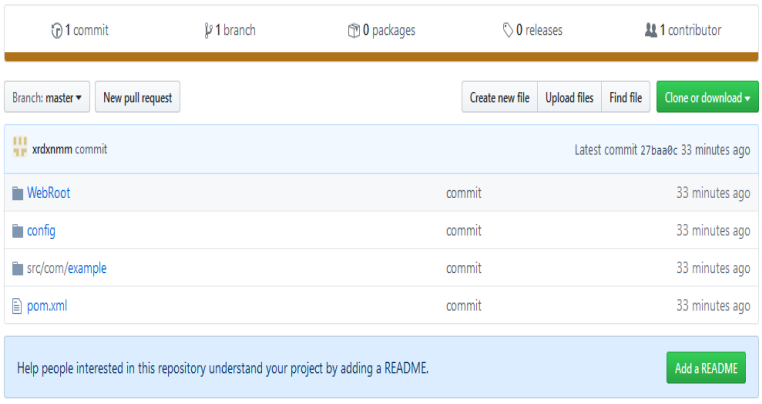
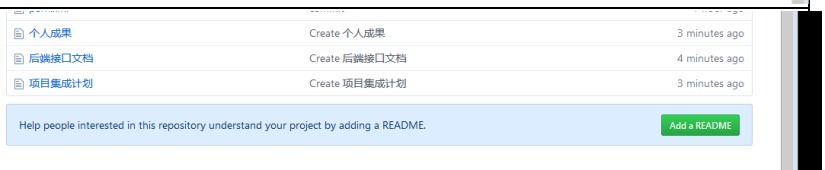



Pull requests Issues Marketplace Explore
Mone-guan / Campus_to_help
Watch 1 Star 0 Fork 1
Code Issues 0 Pull requests 0 Actions Projects 1 Wiki Security Insights
Comparing changes
Choose two branches to see what's changed or to start a new pull request. If you need to, you can also compare across forks.
base repository: Mone-guan/Campus_to_help base: master head repository: liouer/Campus_to_help compare: master
Able to merge. These branches can be automatically merged.
Create pull request
Discuss and review the changes in this comparison with others.
1 commit
16 files changed
0 commit comments
1 contributor
Commits on Nov 18, 2019
liouer
Add files via upload
Verified
e4d1788
Showing 16 changed files with 3,683 additions and 247 deletions.
Unified
Split
240 前端/accept_doing.html


Pull requests Issues Marketplace Explore
liouer / Campus_to_help
forked from Mone-guan/Campus_to_help
Watch 0 Star 0 Fork 1
Code Pull requests 0 Actions Projects 0 Wiki Security Insights Settings
No description, website, or topics provided.
Edit
Manage topics
19 commits
2 branches
0 packages
0 releases
3 contributors
Branch: master
New pull request
Create new file
Upload files
Find file
Clone or download
This branch is 1 commit ahead of Mone-guan:master.
Pull request
Compare
liouer Add files via upload
Latest commit e4d1788 6 minutes ago
前端 Add files via upload 6 minutes ago
8-测试用例.xlsx 前端中进入首页之前介绍的代码 16 days ago
8-测试用例2.xlsx update01File 10 hours ago
AaGXM0881101889 add nacigation page 10 days ago
README.md 前端中进入首页之前介绍的代码 16 days ago
校园帮软件工程项目需求分析书.docx 'first_commit' 2 months ago
校园帮软件项目需求分析书第二版.docx addFile 10 hours ago
测试报告.docx update01File 10 hours ago
测试需求说明书.docx 'delete_wrong_file_README.md' 22 days ago
README.md
Campus to help



项目小组完成情况：

Day	Task	截图
2019-11-11	创建后端代码仓库，将后端项目代码纳入版本控制； 项目已完成用户模块的功能	本阶段性任务完成后 GitLab 上的截图： 
2019-11-17	成功创建文档管理仓库；将原型/需求文档/用户故事列表等纳入版本控制；	

七、经验教训总结

我在项目中负责前端开发部分和需求分析。从这个项目中，我学习到了以下：

对一个产品的形成加深了了解。认识到项目管理的重要性。给我新添了很多概念和思想，包括但不限于一个产品的完成里面要考虑到产品与社会的关系，产品与用户的关系，产品与产家的关系等，涉及到金钱，社会，文化等方面。

了解到需求分析＞码字。需求分析真的很重要！这可能是在这个项目感受最深的了。最开始的需求分析由于是大家一起商量的，但都只是商量了一个大概，大家都凭着自己对项目，对需求分析的理解各抒己见，于是等到自己真正在开发的时候，对需求分析的不明百姓，不确定性，不理解性，不支持性，不可行性等等等等，我先不说后端，先说自己的前端，为了需求分析，多次停下开发的脚步去研究需求分析，这无疑是给开发的进程的一个严重的延误。特别是到开发后期，竟然还在对本应该就确定好的需求分析进行解构然后重构，真的很浪费时间。所以前期一定要做好需求分析！

对前端有了更深的认识。比如说接触了一些框架，比如说 Bootstrap，还有前后端数据的传递，对 css、js 等有了更近一步的了解。了解了 cookie 和 session 等存储方式，了解了前端安全防范问题，了解了 ajax 和 json 之间的关系，学会了怎么调试、接接口、使用 f12。加强了自己的检索能力等等。一定要多调试，调试的时候要多打印，才能更加清楚和方便的知道问题到底出现在哪里。对自己的变量没有一个统一的管理，经常写错或者找不到或忘记变量。

其实一开始开发的时候，找了 bootstrap 的 nifty 模板，当时看了这个模板感觉大概的页面已经有了，然后就拿着这个模板开发。但开发越进行到后面，越看多了别人的页面之后，就会发现，其实，这个模板并不是最合适的，应该找更合适的比如说商城的模板。现在的模板更偏向于管理系统后台管理那一方面。应该要早一点做准备，做事还是欠考虑周全。

一个团队合作之间的关系一定要融洽。要有一个可以抉择的人。也不要吝啬分享，特别是对于我们这样什么都不懂得团队。我们团队我绝对最严重的错误就是错在大家分工之后都是自己学自己，不懂得分享自己所用所学的东西，好比一个链接也好。以至于我不懂你，你不懂我，明明说的是同一个东西，却各有各的一套说我说一套，很难搞。

逐渐明确了自己要攻城的方向。认识到自己更加不适合什么，可能适合什么。在开发的过程中，说实话，我曾无数次怀疑自己是不是选错了专业，特别是遇到一直卡着的问题的时候，真的很让人怀疑人生。当中，我怀疑过自己，质疑过自己，同时也鼓励过自己，肯定过自己，崩溃过也兴奋过，失败过也成功过。一直在两面之间徘徊。但就是这么个徘徊的过程，才让自己发现了自己的另一面。更加认识到了自己的性格、习惯、以及在面对一些事情的反应和面对方式。

通过这个项目，我深深地了解到自己还要学习很多内容，这只是对现在的我来说是一个起点，但我相信，通过这一次项目带来的经历，以后自己能更加的珍惜时间，对时间的观念性，对自己发展的方向，对自己追求的目标有一定的明确和增强。

学习都是从模仿开始的，当自己按着 f12 去看各个控件，当自己趴在框架的模板上找控件，当在 Demo 里面找样式的时候，那个样子真的像足了在街边贴膜的，但那个很认真，很投入的自己真的很让人着迷。希望自己可以改掉一些坏毛病，要沉得住气，更加耐心地去对待 bug，更加专心的去学习，更加热爱自己的专业，去发现更好的自己。谢谢。