# 《闲云》成果交付

# 1、系统概述

**1.1、系统简介**

闲云是一款面向校园用户的二手信息发布平台，基于微信小程序实现。一方面校园用户在日常学习生活中难免会产生一些闲置物品，另一方面有的用户对相关物品有需求可是没有必要或不愿购买全新的商品。有关的供求信息往往不平衡，这时一个二手信息发布平台呼之欲出。闲云可以为校园用户提供二手商品信息交换的平台，这样用户物品的闲置率将大大降低，资源会得到有效的再利用，减少资源浪费。

**1.2、系统运行环境**

硬件平台：安装有微信app的智能手机

操作系统：所有支持微信app的手机操作系统（包括但不限于Android，iOS）

数据库系统：微信小程序云开发提供的基于JavaScript的云数据库

编程平台：Windows操作系统

网络协议：https协议

**1.3、开发环境**

工程工具：微信web开发者工具 版本Stable v1.02.1904090

Notepad++ 版本v7.5.9

Microsoft Visio 2016

开发语言：JavaScript，HTML，CSS

# 2、系统说明

**2.1、整体说明**

在该部分简要介绍系统的整体功能，并给出系统对于的流程图。对于一些模块间合作完成或是系统整体合作完成的功能，在该部分应向用户解释该如何输入，又应得到什么输出。

**2.2、index模块**

2.2.1、功能描述

小程序主页面，提供向多个子页面跳转的接口，包括：搜索页面，商品分类页面，发布页面。在五种分类的按钮下边是数据库中所有商品信息列表显示。

2.2.2、使用说明

页面下半部分显示goods表中所有记录，每条记录对应一个信息框，显示商品的图片、名称、类别和价格。

**2.3、add模块和modify模块**

2.3.1、功能描述

add模块实现用户输入商品名称、类别、价格和商品描述，选择商品图片，和提交商品信息到数据库、把图片文件上传到云端的功能。其中上传图片部分支持上传不超过3张图片。

modify模块实现修改商品详细信息的功能，页面中用户空缺的项保持原值不变。

2.3.2、使用说明

用户输入商品名称、价格、描述，选择商品类别和商品图片。设置商品名称、价格、图片为必填项。如果用户未填写完整信息则e.detail.value的相应属性为空字符串；或者用户没有选择图片，则列表this.data.images的长度为0.因为现实中两个用户可能会发布相似的商品，使用了相同的商品名称，所以系统不会检测新发布的商品的信息与数据库已有商品信息是否有重复。即使出现冗余数据，5天自动下架和10天自动删除可以清理这些冗余数据，数据库中的冗余数据不会过多。

如果用户填写的商品信息完整，在点击提交后商品信息对象会上传到云数据库，用户选择的图片会上传到云存储空间。页面提示“提交成功”告知用户商品信息提交成功，之后返回小程序首页。

2.3.3、重要代码分析

结合算法，对模块中的重要代码进行分析。

//表单提交函数

formSubmit(e) {

console.log(e);

var information = e.detail.value;

//数据有效性、完整性检验

if (information.name == ""){ //商品名称未填

wx.showModal({

title: '提示',

content: '请输入商品名称！'

})

}

else if (information.phone == ""){ //价格未填

wx.showModal({

title: '提示',

content: '请输入商品价格！'

})

}

else if(this.data.images.length==0){ //没有选择图片

wx.showModal({

title: '提示',

content: '请选择照片！'

})

}

else{

var that = this;

wx.showLoading({

title: '上传中...',

})

const filePath = this.data.images, cloudPath = [];

filePath.forEach((item, i) => {

cloudPath.push(util.vcode(new Date()) + '\_' + i + filePath[i].match(/\.[^.]+?$/)[0])

})

console.log(cloudPath)

filePath.forEach((item, i) => { //上传图片

wx.cloud.uploadFile({

cloudPath: cloudPath[i],

filePath: filePath[i],

success: res => {

console.log('[上传图片] 成功:', cloudPath[i], res)

that.data.fileID.push(res.fileID)

app.globalData.cloudPath = cloudPath

app.globalData.imagePath = filePath

if (i === filePath.length - 1) { //当所有图片上传完成后，上传数据

var toSubmit = { //嵌套if实现主要是为了获取文件在云端的存储路径数组this.data.fileID

name: information.name,

price: information.phone,

category: that.data.array[that.data.index],

time:new Date(),

collect:false,

info: information.info,

image: that.data.fileID

};

console.log(toSubmit);

db.collection('goods').add({ //上传记录到goods表

data: toSubmit,

success: res => { //提交成功信息提醒

wx.showToast({

title: '提交成功',

icon: 'success'

})

console.log('[数据库] [新增记录] 成功，记录 \_id: ', res.\_id)

setTimeout(() => { //跳转到首页

wx.switchTab({

url: '../index/index',

})

}, 200)

},

fail: err => {

wx.showToast({ //提交失败信息提醒

icon: 'none',

title: '提交失败'

})

console.error('[数据库] [新增记录] 失败：', err)

}

})

}

},

fail: e => { //提交失败信息提醒

console.error('[上传文件] 失败：', e)

wx.showToast({

title: '图片上传失败',

icon: 'none'

})

}

})

})

wx.hideLoading()

}

modalHidden: true;

}

**2.4、search模块**

2.4.1、功能描述

该模块实现搜索商品的功能。该页面显示搜索历史记录，点击相应的记录可以完成对该内容的再一次搜索。页面上有清空历史按钮，点击即可清空搜索历史记录。

2.4.2、使用说明

介绍该模型的输入（输入数据、有效性检测规则等），输出信息（输出信息、表现形式、信息意义等）。

输入：

用户在搜索框输入要搜索的商品的名称。

和历史纪录有关的操作是从history表获取数据，查看历史纪录的条件是：

\_openid: this.data.openid,

\_id: 'history' + this.data.openid，

输出：

产生的数据有inputValue和historyList，inputValue是用户的输入内容，传给list模块进行搜索。HistoryList是用户的搜索历史纪录，在wxml中遍历列表将所有内容显示在“搜索历史“的下方。

**2.5、list模块**

2.5.1、功能描述

本页面实现搜索商品并将搜索结果列表显示的功能，搜索内容为search页传递进来的字符串。设置每页显示10条记录。页面具有上拉加载功能。

2.5.2、使用说明

输入：从goods表中获取数据，查询操作的条件是：

name: {

$regex: '.\*' + content,

$options: 'i'

}

输出：LoadList函数查询到数据后将数据保存在页面全局变量this.data.list，wxml中遍历list逐条显示数据项。数据是所有符合查询条件的数据项。

**2.6、detail模块**

2.6.1、功能描述

该模块用于向用户展示商品详细信息，显示商品图片、名称、类别、价格和说明，图片以轮播图的形式展现数据库中该商品的所有图片。页面底部提供收藏、聊天、转发按钮。点击收藏则商品信息进入用户收藏夹，点击聊天则进入与卖家聊天的聊天室，点击转发用户可以和好友分享商品信息。

2.6.2、使用说明

输入：LoadDetail函数从数据库goods表中获取该商品的信息，查询条件：\_id: \_.eq(productId)。GetCollect函数从数据库goods表中查询该商品信息，条件：\_id: productid。

输出：模块通过查询数据库得到商品图片、名称、类别、价格、描述，将这些数据呈现在页面中。

2.6.3、重要代码分析

getToken() { //获取token作为用户登陆环信服务器的唯一标识符

wx.request({

url: "http://a1.easemob.com/1109190602090245/xianyun/token",

method: 'POST',

header: {

'content-type': 'application/json',

}, data: {

"grant\_type": "client\_credentials",

"client\_id": "YXA6W936UIT8EemXqp-Dn7BjCw",

"client\_secret": "YXA6bb0r3sk69p56AnTTFGu0imljW4s"

},

success(res) {

console.log(res)

token = res.data.access\_token

console.log(token)

},

fail(res) {

console.log("添加好友失败")

}

})

},

add(u1, u2, pram){ //利用上一步获得的token使用服务器的rest 接口将买家和卖家免验证强制添加好友

wx.request({

url: "https://a1.easemob.com/1109190602090245/xianyun/users/" + u1 + "/contacts/users/" + u2,

method: 'POST',

header: {

'content-type': 'application/json',

'Authorization': 'Bearer '+pram ,

},

success(res) {

console.log(res)

console.log("添加好友成功")

},

fail(res) {

console.log("添加好友失败")

}

})

}

**2.7、chat模块**

2.7.1、功能描述

模块实现买家和卖家之间的聊天，支持发送文字和图片。

2.7.2、使用说明

介绍该模型的输入（输入数据、有效性检测规则等），输出信息（输出信息、表现形式、信息意义等）。

输入：用户通过手机键盘输入要发送的文字消息。输入框支持从手机相册中选择图片发送或者打开相机拍摄照片并发送。

模块聊天功能的实现主要是调用环信云的接口，从服务器中获取聊天记录数据。使用相关的查询语句就可以获取这些数据。

输出：用户发送的文字消息和图片在页面中以对话框的形式呈现。对话框上方标注消息的发送时间。

**2.8、books, clothes, others, phones, snacks模块**

这五个模块的功能和实现方式相似，下文以books为代表进行分析。

2.8.1、功能描述

books模块实现书籍类商品记录的列表显示。

2.8.2、使用说明

介绍该模型的输入（输入数据、有效性检测规则等），输出信息（输出信息、表现形式、信息意义等）。

输入：该模块从数据库goods表中获取category（类别）为“书籍”的所有数据项。查询条件：category:”书籍”。

输出：模块查询到goods表的所有category为商品的数据项将这些数据项储存在this.data.list中，在books.wxml中以list的index为wx:key，wx:for遍历list将所有记录的名称、类别、价格、图片封装成一个信息框，列表显示所有书籍类商品。

**2.9、index1模块**

2.9.1、功能描述

本模块显示用户曾经发过消息的对话框，对话框显示用户和卖家交流过的最后一条消息。点击对话框用户可以进入与这个卖家聊天的界面。用户向左滑动对话框可以删除这个对话框，同时可以删除和这个卖家曾经聊天的记录。

2.9.2、使用说明

输出：本模块基于环信聊天sdk开发，模块通过环信云服务器获取用户聊天历史数据，sdk在获取到数据后在聊天界面中将历史对话框展示。

**2.10、index2模块**

2.10.1、功能描述

该页面具有获取用户信息的功能，用户点击头像则对小程序进行授权，小程序可以获取用户的公开信息。用户头像的下方有两个按钮：我发布的和我喜欢的，点击这两个按钮可分别跳转到用户发布过的商品列表页和用户收藏夹。

2.10.2、使用说明

介绍该模型的输入（输入数据、有效性检测规则等），输出信息（输出信息、表现形式、信息意义等）。

输入：用户点击获取头像昵称键小程序自动获取微信用户公开信息。

输出：小程序得到用户授权后在原来“获取头像昵称”键的位置显示用户的头像和昵称。

2.10.3、重要代码分析

onLoad: function () {

console.log("app.globalData.OpenId:" + app.globalData.OpenId)

if (app.globalData.userInfo) {

this.setData({

userInfo: app.globalData.userInfo,

hasUserInfo: true

})

} else if (this.data.canIUse) {

// 由于 getUserInfo 是网络请求，可能会在 Page.onLoad 之后才返回

// 所以此处加入 callback 以防止这种情况

app.userInfoReadyCallback = res => {

this.setData({

userInfo: res.userInfo,

hasUserInfo: true

})

}

} else {

// 在没有 open-type=getUserInfo 版本的兼容处理

wx.getUserInfo({

success: res => {

app.globalData.userInfo = res.userInfo

this.setData({

userInfo: res.userInfo,

hasUserInfo: true

})

}

})

}

}

**2.11、like模块**

2.11.1、功能描述

和list界面基本一致，只是用户收藏夹为空时显示“暂无此类商品”。

2.11.2、使用说明

输入：从goods表中获取collect属性值为true的所有数据项。

输出：getdata函数查询到数据后将数据保存在页面全局变量this.data.list，wxml中遍历list逐条显示数据项。

**2.12、release模块**

2.12.1、功能描述

显示用户发布过的商品列表，用户可以查看、修改发布过的商品信息。如果商品发布5天后被暂时下架，用户需要在这里确认商品状态是继续在售还是永久下架。

2.12.2、使用说明

输入：从数据库goods表中查询\_openid== wx.getStorageSync("myUsername")的所有数据项，将其写入this.data.list。

# 3、功能测试

该部分针对前一部分的使用说明中提到的各功能进行测试。

**3.1、搜索功能**

3.1.1、测试用例

|  |  |  |
| --- | --- | --- |
| 编号 | 测试用例内容 | 预期结果 |
| 1 | 空字符串”” | 小程序不跳转到pages/list/list |
| 2 | “开发板” | 小程序跳转到pages/list/list?content=${“开发板”}，页面中有商品名称为“开发板”的一条记录 |
| 3 | “衬衫” | 小程序跳转到pages/list/list?content=${“衬衫”}，页面中没有数据记录 |
| 4 | 点击“清空历史”键 | 历史记录清空 |

3.1.2、测试结果

根据测试样例，运行软件，得到测试结果。

|  |  |
| --- | --- |
| 编号 | 测试结果 |
| 1 | 页面不跳转 |
| 2 | 页面中只有一条商品名称为”开发板”的记录，如图1所示。 |
| 3 | 页面中没有数据记录，显示“---我是有底线的---”，如图2所示。 |
| 4 | 提示“清空历史”，搜索历史部分变为“搜索历史为空” |

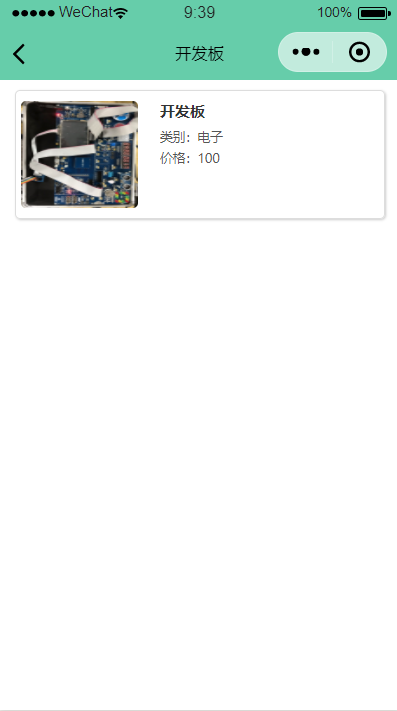
 

图3.1.1 搜索“开发板” 图3.1.2 搜索“衬衫”

图3.1.3 清空历史前 图3.1.4 清空历史后

3.1.3、结果分析

经过测试，结果和预期符合得很好，实现了预期功能。

**3.2、商品信息分类显示功能**

3.2.1、测试用例

|  |  |  |
| --- | --- | --- |
| 编号 | 测试用例 | 预期结果 |
| 1 | 用户点击“书籍”按钮 | 跳转到  pages/classify/books/books,页面中显示商品类别为“书籍”的所有记录。若没有该类别商品显示“暂无此类商品” |
| 2 | 用户点击“服装”按钮 | 跳转到  pages/classify/clothes/clothes,页面中显示商品类别为“服装”的所有记录。若没有该类别商品显示“暂无此类商品” |
| 3 | 用户点击“电子”按钮 | 跳转到  pages/classify/phones/phones,页面中显示商品类别为“电子”的所有记录。若没有该类别商品显示“暂无此类商品” |
| 4 | 用户点击“零食”按钮 | 跳转到  pages/classify/snacks/snacks,页面中显示商品类别为“零食”的所有记录。若没有该类别商品显示“暂无此类商品” |
| 5 | 用户点击“其他”按钮 | 跳转到  pages/classify/others/others,页面中显示商品类别为“其他”的所有记录。若没有该类别商品显示“暂无此类商品” |

3.2.2、测试结果

根据测试样例，运行软件，得到测试结果。

|  |  |
| --- | --- |
| 编号 | 测试结果 |
| 1 | 页面中显示goods表中所有category==”书籍”的记录 |
| 2 | 页面显示“暂无此类商品” |
| 3 | 页面中显示goods表中所有category==”电子”的记录 |
| 4 | 页面显示“暂无此类商品” |
| 5 | 页面中显示goods表中所有category==”其他”的记录 |

限于篇幅，只贴出书籍类商品分类显示页面如图3.2所示。

图3.2

测试结果与预期结果相符。

**3.3、发布功能**

3.3.1、测试用例

测试用例：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号 | 商品名称 | 类别 | 价格 | 描述 | 图片 |
| 1 | （空） | 书籍 | 15 | （空） | IMG\_5230.jpg |
| 2 | 程序设计实验 | 书籍 | （空） | 好书，值得一读。 | IMG\_5230.jpg |
| 3 | 程序设计实验 | 书籍 | 15 | （空） | （空） |
| 4 | 程序设计实验 | 书籍 | 15 | 好书，值得一读。 | IMG\_5230.jpg |

预期结果：

|  |  |  |
| --- | --- | --- |
| 编号 | 测试用例 | 预期结果 |
| 1 | 1 | 出现模态框提示“请输入商品名称！” |
| 2 | 2 | 模态框提示“请输入商品价格！” |
| 3 | 3 | 模态框提示“请选择照片！” |
| 4 | 4 | 提示“提交成功！”后跳转到小程序首页 |

3.3.2、测试结果

信息未填写完整会弹出模态框提示补全信息，如图3.3.1~3.3.3所示。

由于提交成功后立刻跳转，所以没办法截图展示相关结果。如图3.3.4所示。

图3.3.1 图3.3.2

图3.3.3 图3.3.4

测试结果与预期结果相符。

**3.4、详细信息展示功能**

3.4.1、测试用例

|  |  |  |
| --- | --- | --- |
| 编号 | 测试用例 | 预期结果 |
| 1 | 开发板 | 跳转到“开发板”详细信息页面 |

3.4.2、测试结果



图3.4

测试结果与预期相符。

**3.5、聊天功能**

3.5.1、测试用例

本模块使用“开发板”商品跳转的聊天页面进行测试。

3.5.2、测试结果

测试结果截图如下：

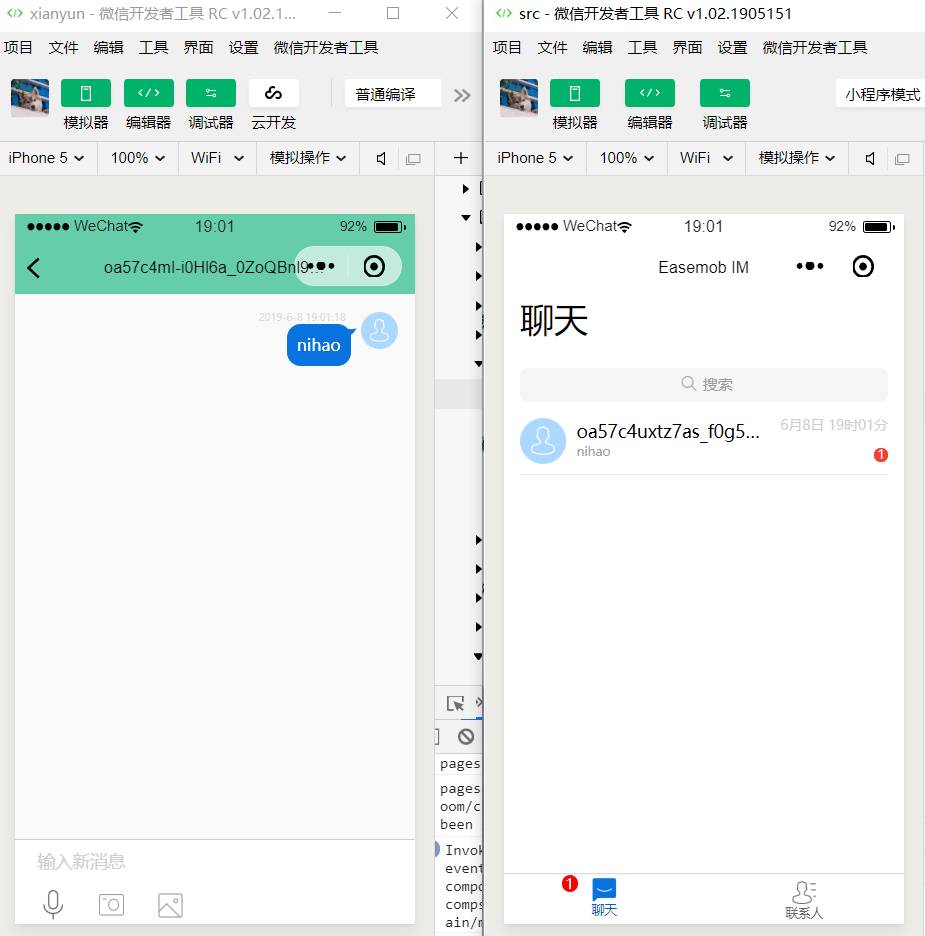
****

图3.5.1

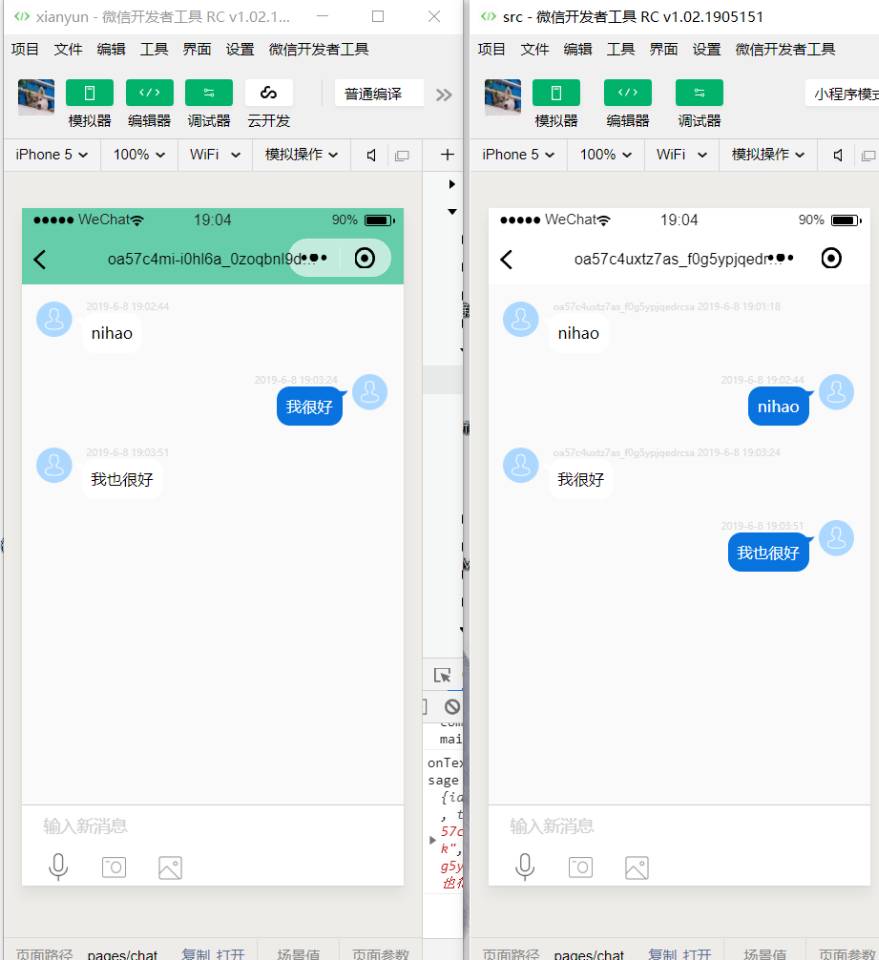


图3.5.2

3.5.3、结果分析

测试结果和预期结果相符。

**3.6、收藏模块**

3.6.1、测试用例

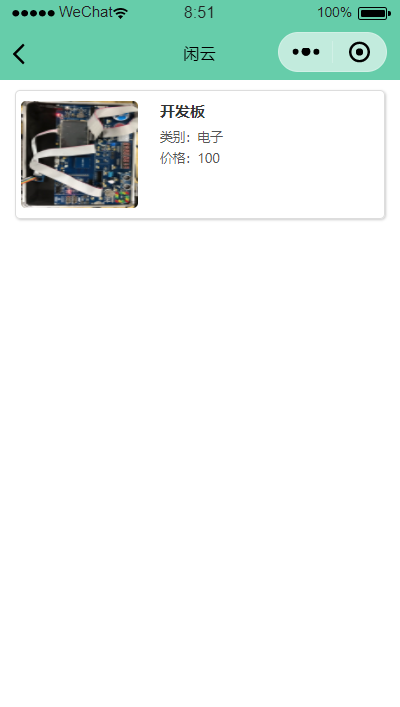
所使用的测试用例如下：

|  |  |  |
| --- | --- | --- |
| 编号 | 测试用例 | 预期结果 |
| 1 | 在“python程序设计”商品详细信息页点击收藏按钮 | 按钮变亮，收藏夹中出现该商品 |
| 2 | 在“开发板”详细信息页点击已经变亮的收藏按钮 | 按钮变灰，收藏夹中该商品信息消失 |

3.6.2、测试结果

根据测试样例，运行软件，得到测试结果。

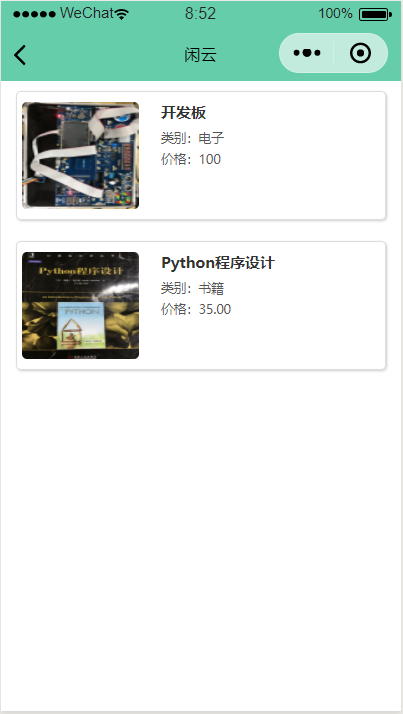
开始时用户收藏夹中只有“开发板”商品，如图3.6.1所示。

图3.6.1

在“python程序设计”商品详细信息页中点击灰色的收藏按钮，按钮变亮，提示“收藏成功”，如图3.6.2所示。

图3.6.2

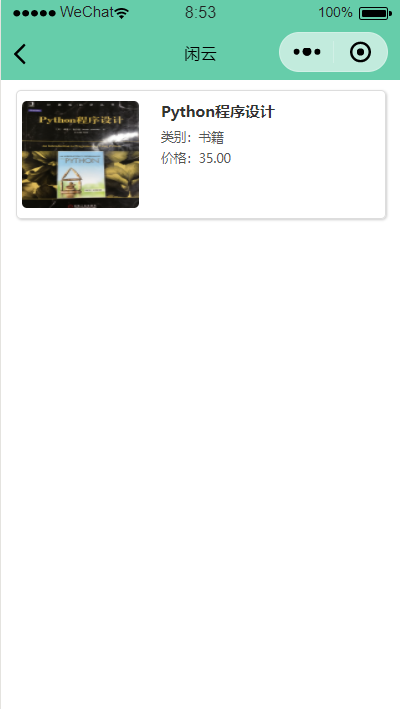
这时收藏夹中共有两条商品信息， “开发板”和“python程序设计”，如图3.6.3所示。

图3.6.3

这时进入“开发板”商品详细信息页，点击收藏按钮，按钮变灰，提示“取消收藏成功”，如图3.6.4所示。

图3.6.4

这时收藏夹中只有一件商品“python程序设计”，如图3.6.5所示。可以看到 “开发板”商品已经从用户收藏夹中移除了。

图3.6.5

可以看到测试结果与预期结果相符。

**3.7、我的发布模块**

3.7.1、测试用例

简要罗列使用的测试用例，或介绍测试用例的设计方法（如边界值、错误推测等）。对于每个测试用例，都应写出预期结果。

|  |  |  |
| --- | --- | --- |
| 编号 | 测试用例 | 预期结果 |
| 1 | 点击“计算机原理”商品信息框 | 跳转到“计算机原理”商品详细信息页 |
| 2 | 点击“python程序设计”信息框的“修改按钮”，修改商品价格 | 数据库中商品价格属性值改变，信息框显示的商品价格改变 |
| 3 | 点击“计原课本”信息框的“删除”按钮 | 该数据项在发布列表、收藏夹和数据库中都被删除 |

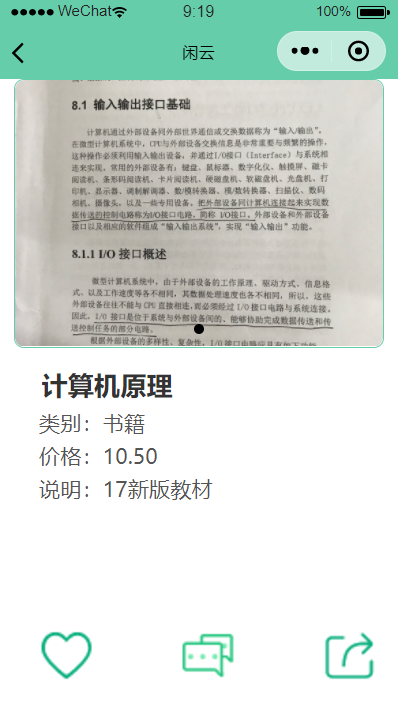
3.7.2、测试结果

根据测试样例，运行软件，得到测试结果。

“我发布的”页面如图3.7.1所示。

图 3.7.1

点击“计算机原理”，跳转到该商品详细信息页，如图3.7.2所示。

图3.7.2

返回发布列表后点击“python程序设计”的“修改按钮”后，进入修改页面，如图3.7.4.



图3.7.3 修改前商品的价格 图3.7.4 修改价格

点击“修改”后提示“修改成功”，再次查看商品价格已变为“25.00”。

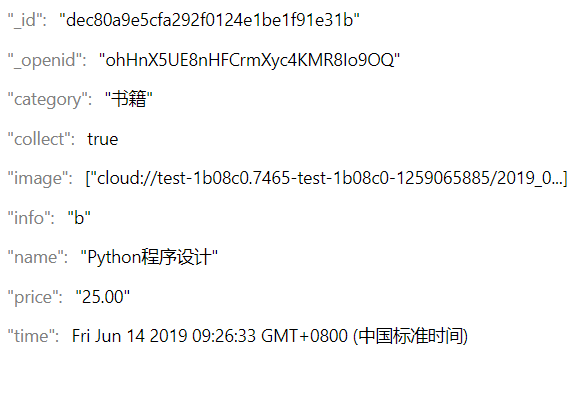


图3.7.5 修改后商品的价格 图3.7.6 数据库中商品价格已被修改

现在点击“计原课本”信息框的删除按钮，可以看到发布列表中该商品信息已被删除。如图3.7.7、3.7.8所示。

图3.7.7 删除“计原课本”前发布列表 图3.7.8 删除“计原课本”后发布列表

3.7.3、结果分析

测试结果与预计结果相符。

# 4、性能测试

4.1、精度分析

系统各模块的输出均满足需求的精度要求。

4.2、时间特性分析

系统各模块反应时间如下：

小程序启动耗时：18ms

帧率：60fps

|  |  |  |  |
| --- | --- | --- | --- |
| 模块 | 切页面耗时 | 初次渲染耗时 | 再次渲染耗时 |
| index | 300ms | 8ms | 9ms |
| search | 122ms | 30ms | 25ms |
| list | 97ms | 14ms | 11ms |
| Books | 108ms | 11ms | 32ms |
| Clothes | 103ms | 11ms | 32ms |
| phones | 79ms | 8ms | 31ms |
| Snacks | 106ms | 9ms | 31ms |
| others | 105ms | 9ms | 24ms |
| detail | 135ms | 42ms | 35ms |
| chat | 191ms | 92ms | 0ms |
| add | 108ms | 25ms | 6ms |
| Index1 | 108ms | 28ms | 5ms |
| Index2 | 12ms | 27ms | 20ms |
| release | 148ms | 7ms | 39ms |
| modify | 116ms | 29ms | 8ms |
| like | 127ms | 8ms | 14ms |

由于需求分析对时间特性没有特别要求，系统各模块反应时间均满足需求分析提出的时间要求。

4.3、灵活性分析

由于小程序具有不依赖运行环境的特点，只要是能运行微信app的智能手机均能支持闲云系统的运行。系统对运行环境、操作方式和输入精度的变化均具有良好的适应能力。

4.4、测试结论和建议

系统可以通过设计的所有性能测试和功能测试，但在并发处理方面仍有待改进。