BBS 系统制作文档

姓名：朱一曾雄　　学号：2017210218031

一、 项目需求

1）登陆识别身份
2）浏览所有根问题（帖子）
3）浏览对应问题的回复
4）回复问题
5）删除自己的留言
6）新建问题

二、 功能分析

1、登陆识别身份。通过用户输入的 userid 和 password 对云数据库中 data 项的进行搜索，如果不存在则无法识别，如果存在则登录成功，并且该页面支持注册功能。

2、浏览所有根问题。对所有已存在的问题进行遍历输出

3、浏览对应问题的回复。该功能在用户点击根问题后跳转页面，在跳转页面下显示该问题的所有回复。

4、回复问题。跳转页面中不仅包括已存在的回复，还可以进一步添加回复。

5、删除自己的留言。该功能仅支持对自己回复的留言进行删除。

6、新建问题。新建根问题。

三、 总体设计

BBS 系统主要是为了实现一下内容：1：登陆识别身份；2：浏览所有根问题（帖子）；3：浏览对应问题的回复；4：回复问题；5：删除自己的留言；6：新建问题。一个用户可以写多个问题，一个问题可以有多个留言，实现 DB-API-MP 之间的联系，通过 login、browseroot、browereplybyid、reply、deleteproblembyid、createproblem 函数实现以上内容。该系统是实现小程序中使用 API 连接云数据库并对数据库进行处理，包括增删查功能，考察小程序的运用。

四、 实现原理

通过对云数据库中的 data 数据进行编辑，包括 userid（用户名），password（密码），id（该数据条目 id），problem（根问题），rapid（回复），pid（回复对应的根问题）等。增删查改都是基于对数据集内容的编辑。微信小程序通过 url 对云数据库进行操作最终实现以上需求。

五、 模块划分

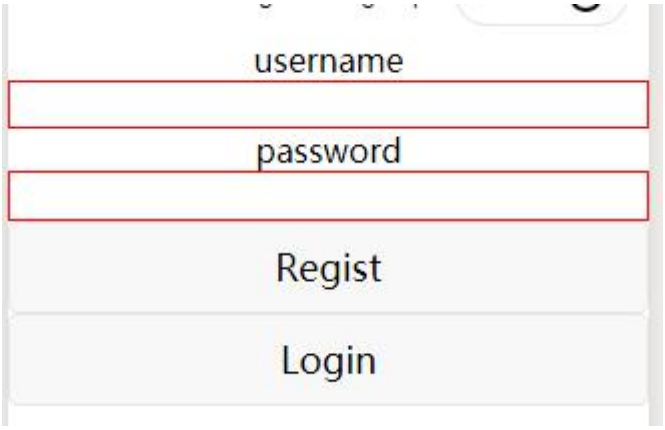
分为 1、登录界面，2、功能选择界面，3、创建问题，4、查询问题，5、查看问题，6、问题详情，7、回复页面。

六、 各模块实现过程

a) 模块一：登录界面

1、设计 UI

分析：在界面中我设计了两个输入框分别用于输入 userid 和 password，以及两个摁钮，分别实现登陆与注册。



2、代码实现

在 js 文件中

```
//记录用户名
bindusername: function (e) {
console.log(e.detail.value)
this.setData({
username: e.detail.value
})
},
//记录用户密码
bindpassword: function (e) {
console.log(e.detail.value)
this.setData({
password: e.detail.value
})
},
//注册
clicktoreg: function () {
var that = this;
wx.request({
//用户信息添加：实现对数据表 reg 的数据增加
url: 'http://172.22.130.33/index.php/Ajaxapi/Ajaxapi/reg',
data: {
username: that.data.username,
password: that.data.password
},
//提示信息框的设置
success: function (res) {
wx.showToast({
title: 'Successfully Regist! You can Login now',
icon: 'none',
duration: 10000
})
```

```
}
})
},
//登录
clicktolog: function () {
var that = this;
var id = that.data.authorid;
wx.request({
//用户信息的查询匹配：对数据表 login 进行查询
url: 'http://172.22.130.33/index.php/Ajaxapi/Ajaxapi/login',
data: {
username: that.data.username,
password: that.data.password
},
success: function (res) {
console.log(res.data.data)
//查询用户信息是否存在 login 数据表中
//若存在，则跳转页面
if (res.data.data != null) {
id = res.data.data.username
console.log(id)
that.setData({
authorid: res.data.data.id
})
wx.setStorage({
key: 'id',
data: id,
})
wx.navigateTo({
url: '/pages/initial/initial',
})

}
//否则提示
else {
wx.showToast({
title: 'The information entered is incorrect',
icon: 'none',
duration: 10000
})
}
},
fail: function () {
console.log("fail")
```

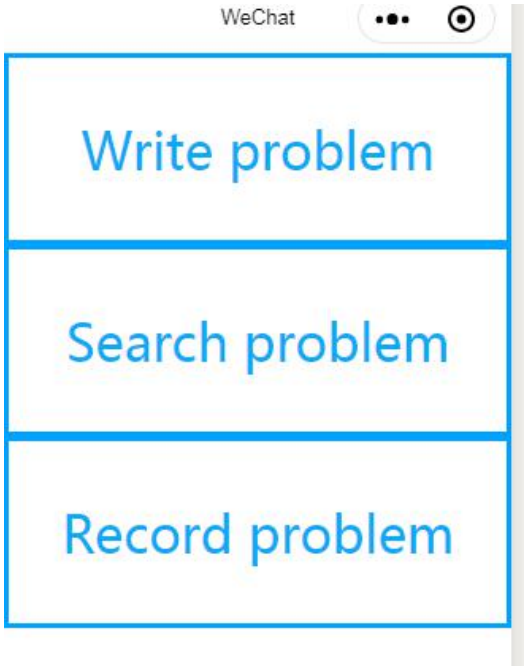
```
}
})
},
```

在 wxml 中

```
<!--pages/result/result.wxml-->

<view class="page-section">
<view class="weui-cells__title">username</view>
<!--用户名输入框设置-->
<view>
<input style="border:1px solid red" maxlength="10" bindinput="bindusername" />
</view>
<view class="weui-cells__title">password</view>
<view>
<!--用户密码输入框设置-->
<input password="true" style="border:1px solid red" class="weui-input" maxlength="10"
bindinput="bindpassword" />
</view>
<!--按钮设置-->
<button type="default" bindtap="clicktoreg" > Regist </button>
<button type="default" bindtap="clicktolog" > Login </button>
</view>
```

b) 功能选择界面：

1、设计 UI



分析：在该界面中我设计了三个按钮，分别实现新建问题（write problem），查询问题（search problem）和查看所有根问题（record problem）。

2、各类实现算法

2.1、新建问题（write problem）

Js 中通过转页到新建问题页面

```
//编写日记
write: function () {
wx.navigateTo({
url: '/pages/problem/problem',
})
},
```

Wxml 中

```
<!--write-->
<view class="image-parent" bindtap="write">
<text style="font-size:2rem;color:#0AAAF6">Write problem</text>
</view>
```

2.2、查询问题（search problem）

Js 中，通过转页转到查询问题页面

```
search: function () {
wx.navigateTo({
url: '/pages/search/search',
```

```
})
},
```

Wxml 中

```
<!--search-->
<view class="image-parent" bindtap="search">
<text style="font-size:2rem;color:#0AAAF6">Search problem</text>
</view>
```

2.3、查看所有根问题（record problem）

Js 中，通过对云数据库的查询，将查询得到的数据储存在缓存中，并跳转到查看页面读取缓存并显示。

```
//查看问题
record: function () {

//获取当前用户所有的信息
var str = this.data.str
wx.request({
url: 'http://172.22.130.33/index.php/Ajaxapi/Ajaxapi/browseroot',
data: {
authorid: this.data.pid,
},
success: res => {
this.setData({
str: res.data.data
})
console.log(this.data.str)
wx.setStorage({
key: 'strr',
data: res.data.data,
})
wx.navigateTo({
url: '/pages/record/record',
})
}

})

},
```

Wxml 中

```
<!--record-->
<view class="image-parent" bindtap="record">
<text style="font-size:2rem;color:#0AAAF6">Record problem</text>
```

```
</view>
```

c) 新建问题

1、设计 UI

分析：在新建问题的页面，我设置了 Title 输入框，用于输入问题的标题，以及文本输入框，用于输入问题的主要内容。我还设置了 Save 按钮用于将输入的内容存入云数据库。

2、各类实现算法

2.1、title 输入框

Js 中

```
bindTitle: function (e) {
this.data.bindTitle = e.detail.value;
},
```

Wxml 中

```
<view class="parent">
<view class="test-bg">
<!--表头信息-->
<view class="title">
<input style="border:1px solid gray;top:-10rpx;width:40rem;margin-left:20rpx;
font-size: 20px; height: 3rem;min-height: 0.8rem" maxlength="20" bindinput="bindTitle"
placeholder="Title" placeholder-style="margin-left:150rpx"/>
</view>
</view>
</view>
```

分析：Js 中将 title 框内内容存入 data 数据。

2.2、文本输入

Js 中

```
//问题内容存储
bind: function (e) {
var that = this
var reason_input = that.data.reason_input
reason_input = e.detail.value

wx.setStorage({
key: 'input',
data: reason_input,
})
this.setData({
reason_input: reason_input
});

},
```

Wxml 中

```
<!--文本框-->
<view class='row' id="textareawrap" catchtap="onFocus">
<textarea bindinput="bind" fixed="true" class='text' maxlength="5000000"
focus="true" name="content" placeholder="点击添加文本" >
</textarea>
</view>
```

分析：Js 中通过缓存将文本内容存入存入缓存，在需要使用的时候取出

2.3、save 按钮

Js 中

```
//问题内容保存
save: function (e) {
var that = this
```

```
console.log(e)
wx.request({
url: 'http://172.22.130.33/index.php/Ajaxapi/Ajaxapi/createproblem',
data: {
userid: that.data.userid,
problem: that.data.reason_input,
},
success: function (res) {
console.log(res.data)
var pages = getCurrentPages();
var prevPage = pages[pages.length - 2];
wx.navigateBack({
delta: 1
})
},
})
},
```

Wxml 中

```
<text bindtap="save"
style="font-size:1.2rem;align-items:right;margin-left:5rpx">Save</text>
```

分析：Js 中通过使用 url，向其中输入 userid 和 problem 内容

D）查询问题

1、设计 UI

分析：在查询页面，设计了关键词输入框以及查询按钮。对云数据库的进行关键词搜索查询所有包括关键词的 data 数据

2、各类实现算法

2.1、关键词输入框

Js 中

```
//存储关键词
bindsearch: function (e) {
console.log(e.detail.value)

this.setData({
searchstr: e.detail.value
})
},
```

Wxml 中

```
<!--搜索栏-->
<view class="parent">
<image bindtap="search" src="/images/sousuo.png"
style="width:100rpx;height:100rpx"></image>
<input style="border:1px solid gray" class="weui-input" maxlength="100"
placeholder="Please enter keywords" bindinput="bindsearch"/>
</view>
```

分析：在 js 中通过对文本框输入值的存储到 data 中。

2.2、关键词搜索实现

Js 中

```
//根据关键词进行搜索
search: function () {
var that = this;
var str = that.data.searchstr;

wx.request({
//用户信息的查询匹配：对数据表 login 进行查询
url: 'http://172.22.130.33/index.php/Ajaxapi/Ajaxapi/read',
data: {
authorid: that.data.id,
searchstr: that.data.searchstr
},
success: function (res) {
```

```
console.log(res.data.data)
//查询用户信息是否存在 login 数据表中
//若存在，则跳转页面
if (res.data.data.length != 0) {
str = res.data.data
that.setData({
str1: str,
})
}
//否则显示提示语
else {
wx.showToast({
title: 'There are not any diaries include these contents',
icon: 'none',
duration: 3000
})
}


},
fail: function () {
console.log("fail")
}
})
},
```

Wxml 中

```
<!--结果栏-->
<view wx:for='{{str1}}' >
<view id='{{item.id}}' bindtap="clickme" class="image-parent" >
<text style="font-size:1rem;color:#0AAAF6" >{{item.sj}}</text>
<text style="font-size:1rem;color:#0AAAF6;margin-left:15rpx">{{item.nr[0]}}</text>
<image src="/images/mood/{{item.xq}}.png"
style="height:100rpx;width:100rpx;margin-left:15rpx"></image>
</view>
</view>
```

分析：

Js 中通过获取 url 向其中赋值 authorid 和 searchstr，当获取成功时读

取其中的 data 数据，如果读取不成功则提示相应文本框。

Wxml 中通过对 item.sj 和 item.nr 的应用，输出数据项的数据和内容。

E）查看问题

1、设计 UI

分析：在该界面内设计了一个根问题的输出，使得所有问题能够一条一条的输出在该界面中

2、各类实现算法

Js 中

```
onLoad: function (options) {
wx.getStorage({
key: 'strr',
success: res => {
this.setData({
str: res.data
});
console.log(res.data)
},
})
wx.getStorage({
key: 'id',
success: res => {
this.setData({
id: res.data
});
console.log(this.data.id)
},
})

},


//查看详情
clickme: function (e) {
```

```
var that = this
var index = that.data.index
index = e.currentTarget.id;
that.setData({
index: index
})
wx.setStorage({
key: 'index',
data: index,
})
//根据用户 id 筛选信息
wx.request({
url: 'http://172.22.130.33/index.php/Ajaxapi/Ajaxapi/read',
data: {
authorid: that.data.id,
},
success: function (res) {
var array = res.data.data
var diary = that.data.diary
//当触发开始
array.forEach((item, index) => {
console.log(item.id)
console.log(that.data.index)
//筛选出用户选中的日记
if (item.id == that.data.index) {
diary = item
that.setData({
diary: diary
})
};
})
wx.setStorage({
key: 'diary',
data: diary,
})
wx.navigateTo({
url: '/pages/recordDiary/recordDiary',
})
},
})
},
```

Wxml 中

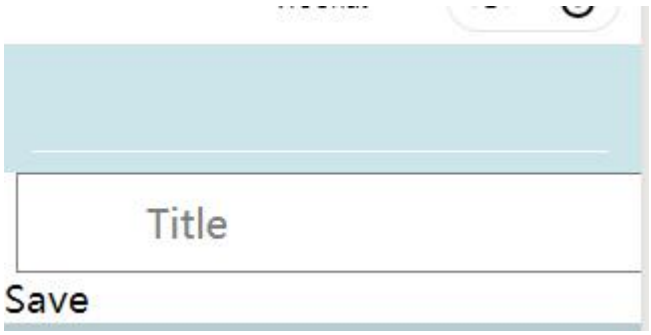
```
<view wx:for='{{str}}'>
<view id='{{item.id}}' class="image-parent" bindtap="clickme">
<text style="font-size:1rem;color:#0AAAF6" >{{item.sj}}</text>

<text
style="font-size:1rem;color:#0AAAF6;margin-left:15rpx">{{item.nr[0]}}{{item.nr[1]}}
{{item.nr[2]}}{{item.nr[3]}}{{item.nr[4]}}{{item.nr[5]}}{{item.nr[6]}}</text>
</view>
</view>
```

分析：在 js 文件中设计了 onload 和 clickme 按钮用于显示根问题条目和触发搜索详细查看选项，即点击界面内条目显示具体内容，详细查看会跳转到新的界面 recordDiary，在该界面可以实现留言和留言的查看，留言的删除。

Wxml 中同理通过对数据条目 item.sj 和 item.nr 对条目的数据和内容进行读取和显示。

F）问题详情

1、设计 UI



分析：在该界面中有根问题显示框，位于回复框上方，回复框和保存按钮。

2、各类实现算法

Js 中

```
onShow: function () {
wx.getStorage({
key: 'id',
success: res => {
this.setData({
id: res.data
});
}})
var that = this;
wx.getStorage({
key: 'index',
success: res => {
that.setData({
index: res.data
});
console.log(res.data)
},
})


},
clickme: function (e) {
var that = this
var index = that.data.index
index = e.currentTarget.id;
that.setData({
index: index
})
wx.setStorage({
key: 'index',
data: index,
})
//根据用户 id 筛选信息
wx.request({
url: 'http://172.22.130.33/index.php/Ajaxapi/Ajaxapi/browsereplybyid',
data: {
pid: that.data.id
},
success: function (res) {
var array = res.data.data
var diary = that.data.diary
//当触发开始
array.forEach((item, index) => {
```

```
console.log(item.id)
console.log(that.data.index)
//筛选出用户选中的日记
if (item.id == that.data.index) {
diary = item
that.setData({
diary: diary
})
};
})
wx.setStorage({
key: 'rapid',
data: diary,
})
wx.navigateTo({
url: '/pages/recordrapid/recordrapid',
})
},
})
},
```

Wxml 中

```
<view class="parent">
<view class="test-bg" style="opacity:{{0.6}}">

<view class='title'>


</view>
</view>
</view>
```

分析：在 js 中通过对缓存文件的读取来获取根问题的详细内容，并对云数据库的 pid 进行搜索，将 pid=id 的回复进行输出。

2.2、添加回复

Js 中

```
//增加回复
rapid: function (e) {
var that = this
var reason_input = that.data.reason_input
reason_input = e.detail.value
wx.setStorage({
key: 'input',
data: reason_input,
})
this.setData({
reason_input: reason_input
})
},

save: function (e) {
var that = this
console.log(e)
wx.request({
url: 'http://172.22.130.33/index.php/Ajaxapi/Ajaxapi/createproblem',
data: {
authorid: that.data.pid
},
success: function (res) {
console.log(res.data)
var pages = getCurrentPages();
var prevPage = pages[pages.length - 2];
wx.navigateBack({
delta: 1
})
},
})
},
```

Wxml 中

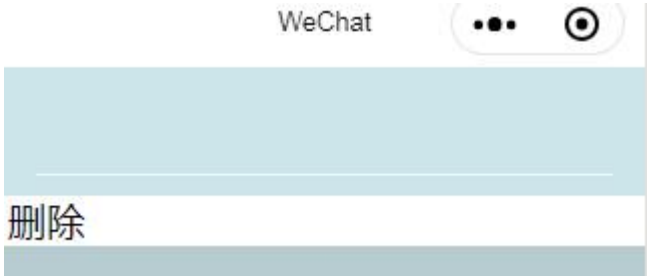
```
<view>
<input style="border:1px solid gray;top:-10rpx;width:40rem;margin-left:20rpx;
font-size: 20px; height: 3rem;min-height: 0.8rem" maxlength="20" bindinput="rapid"
placeholder="Title" placeholder-style="margin-left:150rpx"/>
<text bindtap="save"
style="font-size:1.2rem;align-items:right;margin-left:5rpx">Save</text>
</view>

<view class="image-parent">
```

```
<view class="test-bg" style="opacity:{{0.6}}">
<image src="/images/p1.jpeg" class="bg-image" mode="scaleToFill"></image>
</view>

<view class="name1">{{diary.nr}}</view>
</view>
```

分析：在 js 中，我通过 title 框获取回复内容，通过 save 按钮中的 URL 方法进行存储。

G）回复页面。

1、设计 UI



分析：在该界面中，我设计了回复查看狂，置于删除键上方，以及删除键，对于点击的回复再点击删除实现删除操作。

2、各类实现算法

2.1、回复显示

Js 中

```
onLoad: function (options) {
var that = this;
wx.getStorage({
key: 'index',
success: res => {
that.setData({
index: res.data
```

```
});
console.log(res.data)
},
})
},
```

Wxml 中

```
<view class="parent">
<view class="test-bg" style="opacity:{{0.6}}">

<view class='title'>

<image wx:if='{{diary.xq}}' src="/images/mood/{{diary.xq}}.png"
style="width:100rpx;height:100rpx;margin-left:15rpx"></image>

<image src="/images/rili.png"
style="width:100rpx;height:100rpx;margin-left:15rpx"></image>

</view>
</view>
</view>
```

分析：在 js 中我读取了上一界面的缓存进行对该界面 data 赋值，用于输出显示根问题回复

2.2、删除回复

Js 中

```
Delet:function(e){
var that = this
console.log(e)
wx.request({
url: 'http://172.22.130.33/index.php/Ajaxapi/Ajaxapi/deleteproblembyid',
data: {
userid: that.data.userid,
pid: that.data.pid
},
success: function (res) {
console.log("删除成功")
```

```
},
})
},
```

Wxml 中

```
<view>
<text bindtap="Delet" style="font-size:1.2rem;align-items:right;margin-left:5rpx">删除</text>
</view>
<view class="image-parent">

<view class="test-bg" style="opacity:{{0.6}}">
<image src="/images/p1.jpeg" class="bg-image" mode="scaleToFill"></image>
</view>

<view class="name1">{{diary.nr}}</view>
</view>
```

分析：在 js 中我设计了 delet 函数用于触发删除操作，通过 URL 对云数据库赋值 userid 和 pid 进行匹配点击的回复进行删除操作。