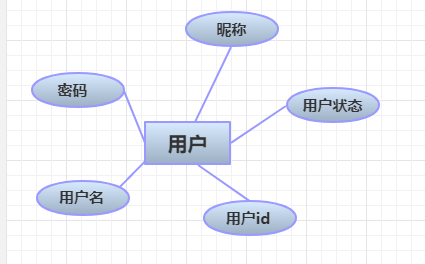
**数据库设计：**在此项目中我们只设计了一个用于存储用户信息的数据库，用户id设计为主键递增，昵称表示用户取的显示名字，用户名和密码用来账号登录，而状态表示用户是否准备好开始比赛。



**单元测试设计：**该项目的单元测试部分主要是针对常规登陆方式（即账号密码登陆）和人脸登录方法进行测试，这里在不改动原有的常规登陆和人脸登录方法代码的基础上，分别编写这两个方法对应的拥有相同逻辑代码的测试方法，具体测试代码如下：

常规登陆方法

①测试方法：

/\*

\* 常规登录测试方法

\* \*/

public void usualLogin(String account, String password) {

String str\_act = account;

String str\_pwd = password;

//Message msg = new Message();

if(str\_act.length() == 0 || str\_act == null) {

testResult = "账号不能为空！";

}

else if(str\_pwd.length()==0 || str\_act == null){

testResult = "密码不能为空！";

}

new Thread(()->{

try

{

thebin.socket = new Socket(thebin.ser\_ip,thebin.ser\_login\_port);

ips = new CppInputStream(thebin.socket.getInputStream());

ops = new CppOutputStream(thebin.socket.getOutputStream());

data\_common.user\_login ul =new data\_common().new user\_login();

stringToCharArray.stringToCharAry(ul.account,str\_act);

stringToCharArray.stringToCharAry(ul.password,str\_pwd);

data.sendPackHeader(data.login\_status.READY,data.pk\_type\_ready.NORM\_LOGIN,data.return\_type.NOTHING,0,ops);

ul.send(ops);

data.packHeader ph2 = new data().new packHeader();

ph2.recv(ips);

if(ph2.stat == data.login\_status.READY) {

if(ph2.type == data.pk\_type\_ready.NORM\_LOGIN){

if(ph2.sub\_type == data.return\_type.SUCCESS){

testFlag = 1;

//testResult = "登陆成功！";

}

else if(ph2.sub\_type == data.return\_type.INPUT\_ERROR){

testFlag = 2;

//testResult = "密码不能为空！";

}

else if(ph2.sub\_type == data.return\_type.FAILED){

testFlag = 3;

//testResult = "登陆失败,账号可能已经在线";

}

}

}

ips.close();

ops.close();

thebin.socket.close();

}

catch (Exception e)

{

testFlag = 4;

}

},"thread1").start();

if (testFlag == 1) {

testResult = "登陆成功！";

}

if (testFlag == 2) {

testResult = "账号或者密码错误！";

}

if (testFlag == 3) {

testResult = "登陆失败,账号可能已经在线";

}

if (testFlag == 4) {

testResult = "网络异常！";

}

}

②单元测试类中对应的测试代码：

/\*

\* 常规登录测试

\* \*/

@Test

public void testUsualLogin() {

try {

//创建登录类对象

LoginActivity loginActivity = new LoginActivity();

//账号密码都为空

loginActivity.usualLogin("", "");

//账号为空，密码不为空的情况

//loginActivity.usualLogin("", "1");

//账号不为空，密码为空的情况

//loginActivity.usualLogin("1", "");

//账号密码都不为空且账号不正确，但密码正确的情况

//loginActivity.usualLogin("2", "1");

//账号密码都不为空且账号正确，但密码不正确的情况

//loginActivity.usualLogin("1", "2");

//账号密码都不为空且都正确的情况

//loginActivity.usualLogin("1", "1");

String testResult = loginActivity.testResult;

if (testResult != null) {

System.out.println("测试通过！测试结果为：" + testResult);

}

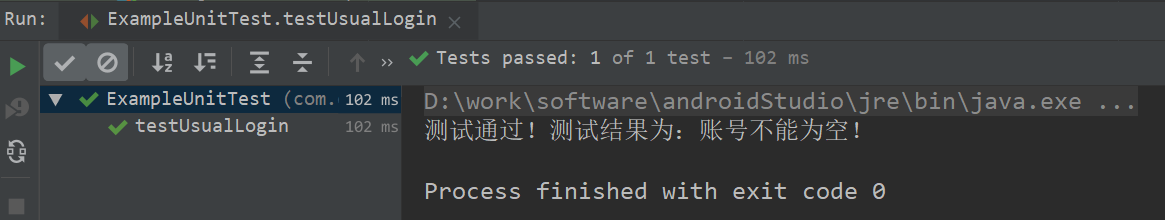
} catch (Exception e) {

System.out.println("测试不通过！");

}

}

③测试结果：部分的测试结果如下图所示。



人脸登录方法

①测试方法：

/\*

\* 人脸登录测试方法

\* \*/

public void testFaceLogin(InputStream inputStream) {

if (inputStream == null) {

testFlag = 5;

} else {

new Thread(()->{

try {

thebin.socket = new Socket(thebin.ser\_ip,thebin.ser\_login\_port);

ips = new CppInputStream(thebin.socket.getInputStream());

ops = new CppOutputStream(thebin.socket.getOutputStream());

data.sendPackHeader(data.login\_status.READY, data.pk\_type\_ready.FACE\_LOGIN, data.return\_type.NOTHING,0,ops);

compressBitmapAndUpload1(inputStream,ops);

data.packHeader ph = new data().new packHeader();

ph.recv(ips);

if(ph.stat == data.login\_status.READY) {

if (ph.type == data.pk\_type\_ready.FACE\_LOGIN) {

if (ph.sub\_type == data.return\_type.SUCCESS) {

//登录成功

testFlag = 1;

} else if (ph.sub\_type == data.return\_type.FAILED) {

testFlag = 2;

} else if (ph.sub\_type == data.return\_type.INPUT\_ERROR) {

testFlag = 3;

}

}

}

ips.close();

ops.close();

thebin.socket.close();

}catch (Exception e){

testFlag = 4;

e.printStackTrace();

}

}).start();

}

/\*

\* 根据不同情况来给出测试结果

\*/

if (testFlag == 0) {

testResult = "测试出错！";

} else if (testFlag == 1) {

testResult = "人脸验证通过，登陆成功！";

} else if (testFlag == 2) {

testResult = "您已登录或该人脸还未绑定任何账号！";

} else if (testFlag == 3) {

testResult = "未检测到人脸或人脸检测失败！";

} else if (testFlag == 4) {

testResult = "网络出现异常！";

} else if (testFlag == 5) {

testResult = "还未上传照片！";

}

}

②单元测试类中对应的测试代码：

/\*

\* 人脸登录测试

\* \*/

@Test

public void testFaceLogin() {

try {

//创建人脸登录类对象

MainActivity mainActivity = new MainActivity();

//未上传照片

InputStream inputStream1 = null;

mainActivity.testFaceLogin(inputStream1);

//上传的照片中有人脸且已绑定账号

/\*InputStream inputStream2 = new FileInputStream("C:\\Users\\ASUS\\Desktop\\images\\1.jpg");

mainActivity.testFaceLogin(inputStream2);\*/

//上传的照片中有人脸但未绑定账号

/\*InputStream inputStream3 = new FileInputStream("C:\\Users\\ASUS\\Desktop\\images\\2.jpg");

mainActivity.testFaceLogin(inputStream3);\*/

//上传的照片中没有人脸

/\*InputStream inputStream4 = new FileInputStream("C:\\Users\\ASUS\\Desktop\\images\\3.jpg");

mainActivity.testFaceLogin(inputStream4);\*/

String testResult = mainActivity.testResult;

if (testResult != null) {

System.out.println("测试通过！测试结果为：" + testResult);

} else {

System.out.println("测试出错！");

}

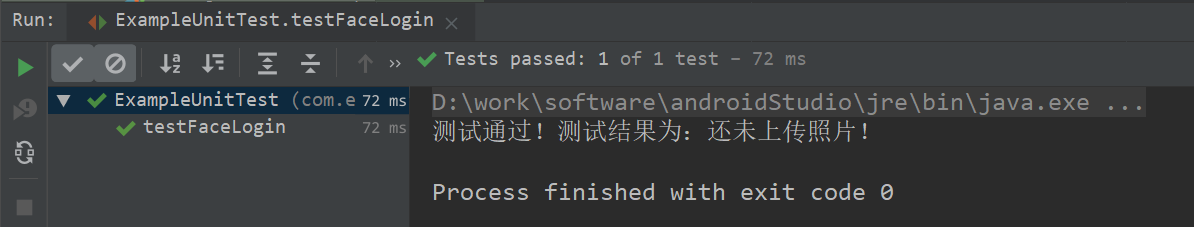
} catch (Exception e) {

System.out.println("测试出现异常");

}

}

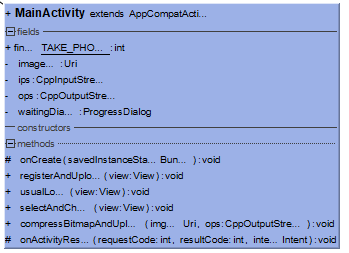
③测试结果：部分的测试结果如下图所示。



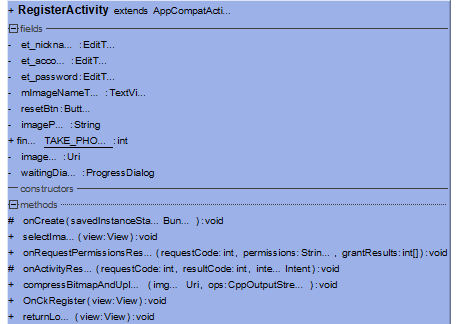
**类设计：**

**客户端(java)**

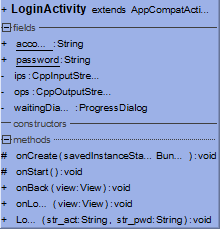
**MainActivity：**主界面类



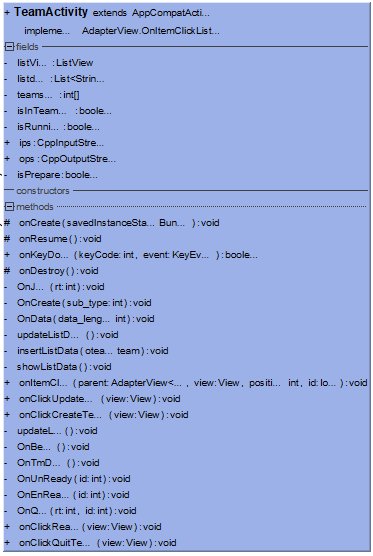
**RegisterActivity：**注册界面类



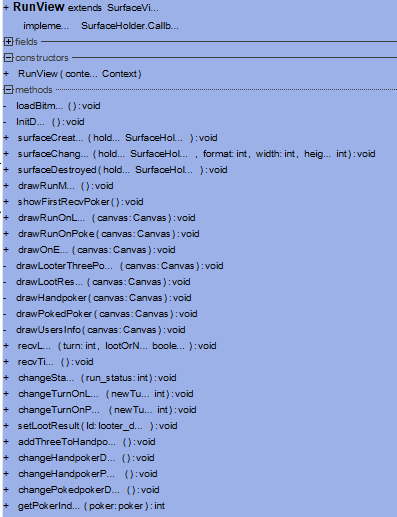
**LoginActivity：**登录界面类



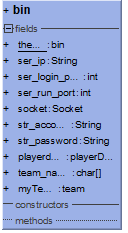
**TeamActivity：**游戏大厅与队伍类



**RunView：**游戏画面和控制类



**Bin：**公共数据类



**服务器(C++)**

**DdzRun:**游戏逻辑类

**TeamManager:**游戏队伍管理类

**本地接口设计：**

该项目用到的接口主要是百度大脑的人脸识别模块相关的API，包括人脸检测、人脸搜索以及人脸注册。以上三个API接口在本地项目中的设计如下：

人脸检测

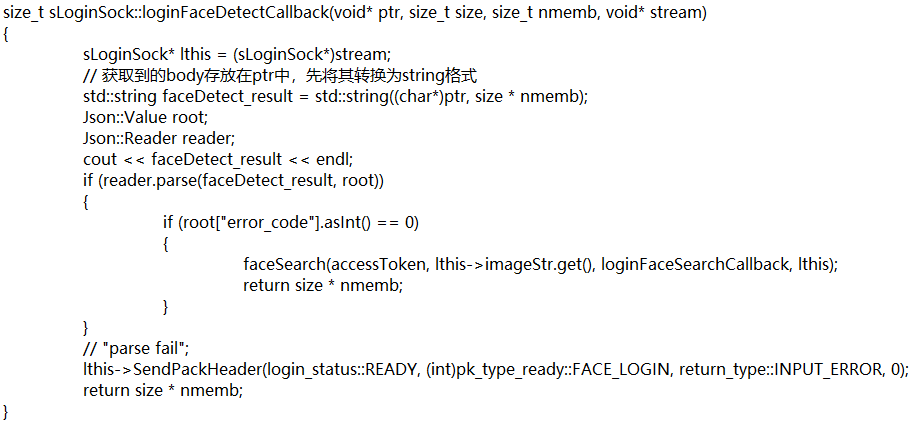
①作用：主要用于检测拍照上传的照片中人脸并标记处位置信息

②实现：该接口在face.cpp中实现，代码如下图所示。



③回调函数：主要在sock.cpp中人脸登录和注册方法实现过程中调用，代码如下图所示。

//人脸登录中人脸检测方法的回调函数



//注册中人脸检测方法的回调函数



人脸搜索

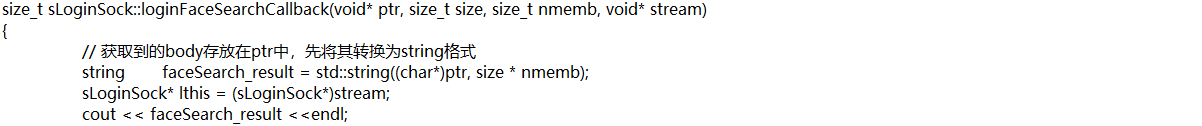
①作用：主要用于在搭建好的百度人脸库中找到最相似的人脸。

②实现：该接口在face.cpp中实现，代码如下图所示。



③回调函数：主要在sock.cpp中人脸登录和注册方法实现过程中调用，代码如下图所示。

//人脸登录中的人脸搜索回调函数



**软件安装及使用说明：**该软件须在安卓手机上安装，下载好安装包后根据系统提示安装即可，安装完成后打开软件进入主界面，首先点击注册按钮，填写昵称，账号和密码，然后点击拍照上传，录入你的脸部信息，填写完成后点击注册按钮。注册完成后你可以选择人脸登录或者账号密码登录，点击人脸登录会打开相机，对自己脸部拍照，若你的照片已经存在照片库中，你则可以顺利登录，否则失败。如果选择账号和密码登录，点击后进入该界面，需要你输入账号及密码后点击登录，若匹配成功，则可以登录成功，否则失败。之后你可以创建自己的房间，也可以加入现有的房间列表，当房间人满三个并且大家都准备好后，则游戏开始，系统开始发牌，之后抢地主，游戏就此开始啦！