基于大数据的手工分享平台源码清单

**学 校：** 肇庆学院

**学 院：**计算机科学与软件学院、大数据学院

**项目 名称 ：** 基于大数据的手工分享平台源码清单

**指导 老师 ：** 岑宇森

**组长(学号)：** 张 楠201624092153

**成员(学号)：** 冯志航201624133213

**成员(学号)：** 伍世活201624133222

目录

[前台代码（app端） 3](#_Toc10761812)

[MainActivity.java 3](#_Toc10761813)

[Get\_Home.java 5](#_Toc10761814)

[Home\_List\_Fragment.java 8](#_Toc10761815)

[CommunityFragment.java 12](#_Toc10761816)

[Community\_all.java 14](#_Toc10761817)

[Community\_fullActivity.java 20](#_Toc10761818)

[后台代码 28](#_Toc10761819)

[DynamicMapper.java 28](#_Toc10761820)

[CustomerMapper.java 28](#_Toc10761821)

[BigDataService.java 32](#_Toc10761822)

[DynamicService.java 32](#_Toc10761823)

[BigDataServiceImpl.java 32](#_Toc10761824)

[DynamicServiceImpl.java 34](#_Toc10761825)

[BigDataHandler.java 34](#_Toc10761826)

[CustomerHandler.java 36](#_Toc10761827)

# 前台代码（app端）

## MainActivity.java

package com.example.manual;

import android.os.Bundle;

import android.support.design.widget.BottomNavigationView;

import android.support.v4.app.Fragment;

import android.support.v4.app.FragmentManager;

import android.support.v4.app.FragmentTransaction;

import android.support.v7.app.ActionBar;

import android.support.v7.app.AppCompatActivity;

import android.support.annotation.NonNull;

import android.view.MenuItem;

import com.example.manual.Chat.Chat\_List\_Fragment;

import com.example.manual.Community.CommunityFragment;

import com.example.manual.Home.Home\_List\_Fragment;

import com.example.manual.Mall.avtivity.MallHomeFragment;

import com.example.manual.Mine.MyFragment;

public class MainActivity extends AppCompatActivity {

Fragment f = null;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

ActionBar actionBar = getSupportActionBar();

actionBar.hide();

FragmentManager fm = getSupportFragmentManager();

FragmentTransaction ft = fm.beginTransaction();

ft.replace(R.id.fragment,new Home\_List\_Fragment());

ft.commit();

BottomNavigationView navView = findViewById(R.id.nav\_view);

navView.setOnNavigationItemSelectedListener(mOnNavigationItemSelectedListener);

}

private BottomNavigationView.OnNavigationItemSelectedListener mOnNavigationItemSelectedListener

= new BottomNavigationView.OnNavigationItemSelectedListener() {

@Override

public boolean onNavigationItemSelected(@NonNull MenuItem item) {

FragmentManager fm = getSupportFragmentManager();

FragmentTransaction ft = fm.beginTransaction();

switch (item.getItemId()) {

case R.id.navigation\_home:

f = new Home\_List\_Fragment();

break;

case R.id.navigation\_message:

f = new Chat\_List\_Fragment();

break;

case R.id.navigation\_community:

f = new CommunityFragment();

break;

case R.id.navigation\_shop:

f = new MallHomeFragment();

break;

case R.id.navigation\_myself:

f = new MyFragment();

break;

default:

break;

}

ft.replace(R.id.fragment,f);

ft.commit();

return true;

}

};

}

## Get\_Home.java

package com.example.manual.Home;

import android.os.Bundle;

import android.os.Handler;

import android.os.Message;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.Map;

import tool.JDBCutils;

public class Get\_Home extends Thread{

private Handler handler;

private String u\_id;

private String t\_id;

//h负责传输数据

//在数据库中搜索涉及u（u\_id)的聊天记录

public Get\_Home(Handler h, String u) {

this.handler = h;

this.u\_id = u;

}

@Override

public void run() {

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Here is Get\_Home\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

Statement stmt=null;

Connection conn=null;

try {

//1.注册驱动程序，2.获取连接对象

conn = JDBCutils.getConnection();

//3.创建Statement

stmt = conn.createStatement();

int[] column\_num = new int[1000];

int i = 0;

int max\_num = 0;

String max\_category = null;

//4.准备sql，获取favorite的列名对应数据

sql ="select pri\_total\_like from favorite where u\_id = '"+u\_id+"'";

System.out.println(sql);

//5.发送sql语句，执行sql语句,得到返回结果

rs = stmt.executeQuery(sql);

if (rs.next()) {

column\_num[i] = rs.getInt("pri\_total\_like");

i++;

}

System.out.println("column\_num.length = "+column\_num.length);

System.out.println("max\_num = "+max\_num);

System.out.println("max\_category = "+max\_category);

for(int ii=0; ii<i; ++ii) {

System.out.println("ii = "+ii+" , column\_num[ii] = "+column\_num[ii]);

}

String[] username = new String[1000];

String[] title = new String[1000];

String[] count = new String[1000];

String[] time = new String[1000];

String[] content = new String[1000];

String[] u\_id1 = new String[1000];

String[] d\_id = new String[1000];

String[] headpic = new String[1000];

String[] type = new String[1000];

String[] category = new String[1000];

int category\_num = (int) ((Math.random()\*49 + 1)%7+1);

sql = "select goods\_category.`ca\_name` from goods\_category "+

"where goods\_category.`ca\_id` = '"+category\_num+"'";

rs = stmt.executeQuery(sql);

String mul\_category = null;

while(rs.next()) {

mul\_category = rs.getString("ca\_name");

}

//获取dynamic表的信息

/\*sql ="select dynamic.\*,customer.name,customer.headPicPath from dynamic,customer where category = '"+max\_category+"' "+

"order by total\_like+total\_benefit desc";\*/

// sql = " select dynamic.\*,customer.`name`,customer.`headPicPath` from dynamic,customer where " +

// "dynamic.`u\_id`= customer.`u\_id` " +

// "and category = '"+max\_category+"' ";

sql = " SELECT dynamic.\*,customer.`name`,customer.`headPicPath` FROM DYNAMIC,customer WHERE" +

" dynamic.`u\_id`= customer.`u\_id` AND (dynamic.category = '"+max\_category+"' OR dynamic.category = '"+mul\_category+"')" +

" ORDER BY total\_like+total\_benefit DESC LIMIT 0,5";

System.out.println(sql);

//5.发送sql语句，执行sql语句,得到返回结果

rs = stmt.executeQuery(sql);

ArrayList<Map<String, Object>> listitem = new ArrayList<>();

while (rs.next()) {

username[i] = rs.getString("name");

title[i] = rs.getString("title");

time[i] = rs.getString("release\_date");

content[i] = rs.getString("content");

u\_id1[i] = rs.getString("u\_id");

d\_id[i] = rs.getString("d\_id");

headpic[i] = rs.getString("headPicPath");

/\* type[i] = rs.getString("type");\*/

category[i] = rs.getString("category");

sql = "select count(\*) from comment where d\_id = '"+rs.getString("d\_id")+"'";

Statement statment2 = conn.createStatement();

ResultSet resultSet = statment2.executeQuery(sql);

resultSet.next();

//获取评论数

int comment = Integer.parseInt(resultSet.getString(1));

//获取点赞数

sql = "select count(\*) from liked where d\_id = '"+rs.getString("d\_id")+"'";

Statement statment3 = conn.createStatement();

ResultSet resultSet2 = statment3.executeQuery(sql);

resultSet2.next();

int like = Integer.parseInt(resultSet2.getString(1));

sql = "update dynamic set total\_comment="+comment+",total\_like="+like+" where d\_id = '"+rs.getString("d\_id")+"'";

Statement statment4 = conn.createStatement();

statment4.execute(sql);

count[i] = "评论："+comment+" 点赞："+like+"";

Map<String, Object> map = new HashMap<String, Object>();

map.put("username", username[i]);

map.put("title", title[i]);

map.put("count", count[i]);

map.put("release\_date" , time[i]);

map.put("content" , content[i]);

map.put("d\_id",d\_id[i]);

map.put("u\_id",u\_id1[i]);

map.put("headpic",headpic[i]);

map.put("category",category[i]);

map.put("total\_like",like);

map.put("total\_comment",comment);

i++;

listitem.add(map);

}

Message message = handler.obtainMessage();

Bundle bundle = new Bundle();

bundle.putSerializable("listitem",listitem);

message.what=0x003;

message.setData(bundle);

handler.sendMessage(message);

} catch (SQLException e1) {

e1.printStackTrace();

} catch (ClassNotFoundException e1) {

e1.printStackTrace();

}

}

}

## Home\_List\_Fragment.java

package com.example.manual.Home;

import android.annotation.SuppressLint;

import android.content.Context;

import android.content.Intent;

import android.content.SharedPreferences;

import android.os.Bundle;

import android.os.Handler;

import android.os.Message;

import android.support.annotation.NonNull;

import android.support.annotation.Nullable;

import android.support.v4.app.Fragment;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.AdapterView;

import android.widget.ListView;

import android.widget.SimpleAdapter;

import com.example.manual.Community.Community\_fullActivity;

import com.example.manual.R;

import com.google.gson.Gson;

import com.google.gson.reflect.TypeToken;

import java.util.ArrayList;

import java.util.List;

import java.util.Map;

import java.util.concurrent.ConcurrentHashMap;

import entity.Customer;

import entity.Dynamic;

public class Home\_List\_Fragment extends Fragment {

private String u\_id;//自己的ID

//通过Get\_Home更新聊天列表

@SuppressLint("HandlerLeak")

private Handler handler = new Handler() {

@Override

public void handleMessage(Message msg) {

ListView listView = null;

switch (msg.what){

case 0x003:

if(getActivity() == null) break;

else {

listView = getActivity().findViewById(R.id.home\_list);

Bundle bundle = msg.getData();

List<ConcurrentHashMap<String, Object>> listitem =

(List<ConcurrentHashMap<String, Object>>) bundle.getSerializable("listitem");

SimpleAdapter adapter=new SimpleAdapter(getContext(), listitem, R.layout.home\_list\_fragment\_item,

new String[]{"username", "title","count","release\_date"}, new int[]{R.id.username, R.id.title,R.id.count,R.id.time});

listView.setAdapter(adapter);

}

break;

default:

break;

}

}

};

@Nullable

@Override

public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {

//获取自己的ID

//创建一个customer.xml存储数据，并设置为私人模式

SharedPreferences sharedPreferences = getActivity().getSharedPreferences("customer", Context.MODE\_PRIVATE);

String customerJson = sharedPreferences.getString("customerJson","");

Gson gson = new Gson();

Customer customer = gson.fromJson(customerJson, Customer.class);

u\_id = customer.getU\_id(); //赋值自己的ID

//启动一个子线程，刷新一次Home推荐列表

Get\_Home gh = new Get\_Home(handler, u\_id);

gh.start();

View view = inflater.inflate(R.layout.home\_list\_fragment,null);

ListView listView=view.findViewById(R.id.home\_list);

listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {

@Override

public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

Map<String, Object> map= (Map<String, Object>) parent.getItemAtPosition(position);

String name=map.get("username").toString();

String title=map.get("title").toString();

String time = map.get("release\_date").toString();

String content = map.get("content").toString();

String u\_id = map.get("u\_id").toString();

String d\_id = map.get("d\_id").toString();

String headpic = map.get("headpic").toString();

String category = map.get("category").toString();

int total\_like = (int)map.get("total\_like");

int total\_comment = (int) map.get("total\_comment");

Dynamic cd = new Dynamic();

cd.setTotal\_like(total\_like);

cd.setTotal\_comment(total\_comment);

cd.setTitle(title);

cd.setD\_id(d\_id);

cd.setContent(content);

cd.setRelease\_date(time);

/\* cd.setType(type);\*/

cd.setCategory(category);

Bundle bundle =new Bundle();

bundle.putString("u\_id",u\_id);

bundle.putString("hostName",name);

bundle.putString("hostHeadPic",headpic);

bundle.putSerializable("dynamic",cd);

//存放到历史记录

SharedPreferences sharedPreferences = getActivity().getSharedPreferences("customer", Context.MODE\_PRIVATE);

String str = sharedPreferences.getString("customerJson","");

Gson gson = new Gson();

Customer customer = gson.fromJson(str,Customer.class);

SharedPreferences history = getActivity().getSharedPreferences(customer.getU\_id(), Context.MODE\_PRIVATE);

String his = history.getString("historyList","");

List<String> historyList = gson.fromJson(his,new TypeToken<List<String>>() {

}.getType());

if(historyList==null){

historyList = new ArrayList<>();

}

//不存在，才加进去

if(!historyList.contains(cd.getD\_id())){

historyList.add(cd.getD\_id());

}

his = gson.toJson(historyList);

SharedPreferences.Editor edit = history.edit();

edit.putString("historyList",his);

edit.apply();

edit.commit();

System.out.println(his);

Intent intent = new Intent();

intent.setClass(getContext(),Community\_fullActivity.class);

intent.putExtra("data",bundle);

startActivity(intent);

}

});

return view;

}

}

## CommunityFragment.java

package com.example.manual.Community;

import android.content.Intent;

import android.os.Bundle;

import android.support.annotation.NonNull;

import android.support.annotation.Nullable;

import android.support.v4.app.Fragment;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.AdapterView;

import android.widget.ListView;

import android.widget.SimpleAdapter;

import com.example.manual.R;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

public class CommunityFragment extends Fragment {

//各个社区的介绍

//等有了数据库，可以上数据库查

String fabric = "布艺社区是一个非常受欢迎的社区++++++++....00";

String leather = "皮艺社区是一个非常受欢迎的社区++++++++....00";

String paper = "纸艺社区是一个非常受欢迎的社区++++++++....00";

String weave = "编织社区是一个非常受欢迎的社区++++++++....00";

String ornaments = "饰品社区是一个非常受欢迎的社区++++++++....00";

String embroidery = "刺绣社区是一个非常受欢迎的社区++++++++....00";

@Nullable

@Override

public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {

View view = inflater.inflate(R.layout.community,null);

//图片资源数组

int[] imageid=new int[] {R.mipmap.fabric,R.mipmap.leather,R.mipmap.paper,

R.mipmap.weave,R.mipmap.ornaments,R.mipmap.embroidery};

String[] title=new String[] {"布艺", "皮艺", "纸艺",

"编织", "饰品", "刺绣" };

//英文title数组

String[] egtitle=new String[] {"fabric", "leather", "paper",

"weave", "ornaments", "embroidery" };

String[] introduction= new String[]{fabric,leather ,paper,

weave ,ornaments,embroidery

};

List<Map<String, Object>> listitem=new ArrayList<Map<String, Object>>();

for(int i=0; i<imageid.length; i++) {

Map<String, Object> map=new HashMap<String, Object>();

map.put("image", imageid[i]);

map.put("name", title[i]);

map.put("egtitle",egtitle[i]);

map.put("introduction",introduction[i]);

listitem.add(map);

}

SimpleAdapter adapter=new SimpleAdapter(getActivity(), listitem, R.layout.community\_main,

new String[]{"name", "image","introduction"}, new int[]{R.id.title, R.id.image,R.id.introduction});

ListView listView=view.findViewById(R.id.listview);

listView.setAdapter(adapter);

listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {

@Override

public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

Map<String, Object> map= (Map<String, Object>) parent.getItemAtPosition(position);

String name=map.get("name").toString();

String egtitle = map.get("egtitle").toString();

Bundle bundle =new Bundle();

bundle.putString("name",name);

bundle.putString("egtitle",egtitle);

Intent intent = new Intent();

intent.setClass(getActivity(),Community\_all.class);

intent.putExtra("title",bundle);

startActivity(intent);

}

});

return view;

}

}

## Community\_all.java

package com.example.manual.Community;

import android.annotation.SuppressLint;

import android.content.Context;

import android.content.Intent;

import android.content.SharedPreferences;

import android.os.Bundle;

import android.os.Handler;

import android.os.Message;

import android.support.annotation.Nullable;

import android.support.v4.view.MenuItemCompat;

import android.support.v7.app.ActionBar;

import android.support.v7.app.AppCompatActivity;

import android.support.v7.widget.SearchView;

import android.util.Log;

import android.view.Menu;

import android.view.MenuInflater;

import android.view.MenuItem;

import android.view.View;

import android.widget.AdapterView;

import android.widget.ImageView;

import android.widget.ListView;

import android.widget.SimpleAdapter;

import android.widget.TabHost;

import com.example.manual.R;

import com.google.gson.Gson;

import com.google.gson.reflect.TypeToken;

import java.io.Serializable;

import java.util.ArrayList;

import java.util.List;

import java.util.Map;

import entity.Dynamic;

import entity.Customer;

import tool.GetDataThread;

public class Community\_all extends AppCompatActivity implements TabHost.OnTabChangeListener, AdapterView.OnItemClickListener {

String introduction = null;

String type = null;//存储当前的选项卡的类型

String category = null;//记录当前所在的社区

private List<Map<String, Object>> listitem=new ArrayList<Map<String, Object>>();

//接收子线程返回回来的数据

@SuppressLint("HandlerLeak")

private Handler handler = new Handler(){

@Override

public void handleMessage(Message msg) {

//0最新回复，1最新发布，2热门

ListView listView = null;

switch (msg.what){

case 0:

type = "latestresp";

listView = findViewById(R.id.lastestresponse);

break;

case 1:

type = "latestpub";

listView = findViewById(R.id.lastestpublish);

break;

case 2:

type = "host";

listView = findViewById(R.id.host);

break;

default:

break;

}

Bundle bundle = msg.getData();

List<Map<String,Object>> listitem = (List<Map<String, Object>>) bundle.getSerializable("listitem");

SimpleAdapter adapter=new SimpleAdapter(getApplicationContext(), listitem, R.layout.community\_all\_main,

new String[]{"name", "title","total\_comment","total\_like","release\_date"}, new int[]{R.id.username, R.id.title,R.id.comment,R.id.like,R.id.time});

listView.setAdapter(adapter);

}

};

@Override

protected void onCreate(@Nullable Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.community\_all);

//点击发布文章监听事件

ImageView imageView = findViewById(R.id.headpic);

imageView.bringToFront();

imageView.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

SharedPreferences sharedPreferences = getSharedPreferences("customer",MODE\_PRIVATE);

String customerJson = sharedPreferences.getString("customerJson","");

Log.i("js",customerJson);

Gson gson = new Gson();

Customer customer = gson.fromJson(customerJson, Customer.class);

if(customer !=null){

Intent intent = new Intent();

Bundle bundle = new Bundle();

bundle.putSerializable("customer", (Serializable) customer);

bundle.putString("category",category);

bundle.putString("type",type);

intent.putExtra("b",bundle);

intent.setClass(Community\_all.this,Community\_publish.class);

startActivity(intent);

}

}

});

ActionBar actionBar = getSupportActionBar();

Intent intent = getIntent();

//从intent取出bundle

Bundle bundle = intent.getBundleExtra("title");

// 设置返回按钮

actionBar.setDisplayHomeAsUpEnabled(true);

category = bundle.getString("egtitle");

actionBar.setTitle(bundle.getString("name")+"社区");

//设置选项卡导航栏

TabHost tabhost = findViewById(R.id.tabhost);

tabhost.setup();

//添加选项

tabhost.addTab(tabhost.newTabSpec("latestresp")

.setIndicator("最新回复")

.setContent(R.id.lastestresponse)

);

//添加选项

tabhost.addTab(tabhost.newTabSpec("latestpublish")

.setIndicator("最新发布")

.setContent(R.id.lastestpublish)

);

//添加选项

tabhost.addTab(tabhost.newTabSpec("host")

.setIndicator("热门")

.setContent(R.id.host)

);

//为三个listview设置监听器

ListView listView1 = findViewById(R.id.lastestresponse);

listView1.setOnItemClickListener(this);

ListView listView2 = findViewById(R.id.lastestpublish);

listView2.setOnItemClickListener(this);

ListView listView3 = findViewById(R.id.host);

listView3.setOnItemClickListener(this);

tabhost.setOnTabChangedListener(this);

//第一次显示最新回复的页面数据

GetDataThread getDataThread = new GetDataThread(handler,"latestresp",category);

getDataThread.start();

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

MenuInflater menuInflater = getMenuInflater();

menuInflater.inflate(R.menu.community\_topmenu,menu);

final MenuItem item = menu.findItem(R.id.search);

SearchView mSearchView = (SearchView) MenuItemCompat.getActionView(item);

mSearchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() { @Override

public boolean onQueryTextSubmit(String query) {

/\* doSearch();\*/

GetDataThread getDataThread = new GetDataThread(handler, type,category);

getDataThread.setTitle(query);

getDataThread.start();

return false;

}

@Override

public boolean onQueryTextChange(String newText) {

/\* doSearch();\*/

return true;

}

});

return super.onCreateOptionsMenu(menu);

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

switch (item.getItemId()) {

case android.R.id.home: //返回键的id

this.finish();

return false;

case R.id.search:

default:

return super.onOptionsItemSelected(item);

}

}

//切换选项卡时，更新数据

public void onTabChanged(String tabId){

GetDataThread getDataThread = null;

if(tabId.equals("latestresp")){

Log.i("sss","1111");

getDataThread = new GetDataThread(handler,"latestresp",category);

}

if(tabId.equals("latestpublish")){

Log.i("112","嘎嘎嘎");

getDataThread = new GetDataThread(handler,"latestpub",category);

}

if(tabId.equals("host")){

Log.i("44","aa");

getDataThread = new GetDataThread(handler,"host",category);

}

getDataThread.start();

}

@Override

public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

Map<String, Object> map= (Map<String, Object>) parent.getItemAtPosition(position);

String name=map.get("name").toString();

String title=map.get("title").toString();

String time = map.get("release\_date").toString();

String content = map.get("content").toString();

String u\_id = map.get("u\_id").toString();

String d\_id = map.get("d\_id").toString();

String headpic = map.get("headPicPath").toString();

/\*Double d1 = new Double((Double)map.get("total\_like"));\*/

int total\_like = Integer.parseInt(map.get("total\_like").toString());

/\* Double d2 = new Double((Double)map.get("total\_comment"));\*/

int total\_comment = Integer.parseInt(map.get("total\_comment").toString());

Log.i("sssssss",total\_like+","+total\_comment);

Dynamic cd = new Dynamic();

cd.setTitle(title);

cd.setD\_id(d\_id);

cd.setContent(content);

cd.setRelease\_date(time);

/\* cd.setType(type);\*/

cd.setCategory(category);

cd.setTotal\_like(total\_like);

cd.setTotal\_comment(total\_comment);

Bundle bundle =new Bundle();

bundle.putString("u\_id",u\_id);

bundle.putString("hostName",name);

bundle.putString("hostHeadPic",headpic);

bundle.putSerializable("dynamic",cd);

//存放到历史记录

SharedPreferences sharedPreferences = getSharedPreferences("customer", Context.MODE\_PRIVATE);

String str = sharedPreferences.getString("customerJson","");

Gson gson = new Gson();

Customer customer = gson.fromJson(str,Customer.class);

SharedPreferences history = getSharedPreferences(customer.getU\_id(), Context.MODE\_PRIVATE);

String his = history.getString("historyList","");

List<String> historyList = gson.fromJson(his,new TypeToken<List<String>>() {

}.getType());

if(historyList==null){

historyList = new ArrayList<>();

}

//不存在，才加进去

if(!historyList.contains(cd.getD\_id())){

historyList.add(cd.getD\_id());

}

his = gson.toJson(historyList);

SharedPreferences.Editor edit = history.edit();

edit.putString("historyList",his);

edit.commit();

System.out.println(his);

Intent intent = new Intent();

intent.setClass(this,Community\_fullActivity.class);

intent.putExtra("data",bundle);

startActivity(intent);

}

}

## Community\_fullActivity.java

package com.example.manual.Community;

import android.annotation.SuppressLint;

import android.content.Context;

import android.content.Intent;

import android.content.SharedPreferences;

import android.os.Bundle;

import android.os.Handler;

import android.os.Message;

import android.support.v7.app.ActionBar;

import android.support.v7.app.AppCompatActivity;

import android.util.Log;

import android.view.Menu;

import android.view.MenuInflater;

import android.view.MenuItem;

import android.view.View;

import android.view.ViewGroup;

import android.widget.Button;

import android.widget.EditText;

import android.widget.ImageView;

import android.widget.ListAdapter;

import android.widget.ListView;

import android.widget.TextView;

import android.widget.Toast;

import com.example.manual.Community.Adapter.CommentAdapter;

import com.example.manual.Community.MyThread.GetCommentThread;

import com.example.manual.Mall.avtivity.MallGoodsDetailActivity;

import com.example.manual.Mall.netUtil.ImageLoad;

import com.example.manual.Mine.GetDataThread.GetLikelThread;

import com.example.manual.Mine.activity.UserHomePage;

import com.example.manual.R;

import com.google.gson.Gson;

import com.google.gson.reflect.TypeToken;

import com.zhuang.likeviewlibrary.LikeView;

import java.text.SimpleDateFormat;

import java.util.List;

import entity.Comment;

import entity.Customer;

import entity.CustomerComment;

import entity.Dynamic;

import tool.MysqlUtil;

import tool.SaveCommentThread;

public class Community\_fullActivity extends AppCompatActivity {

Customer customer;

Dynamic cd;

EditText etcomment;

ListView commentListView;

LikeView like;

@SuppressLint("HandlerLeak")

Handler handler = new Handler(){

@Override

public void handleMessage(Message msg) {

if(msg.what==111) {

Bundle data = msg.getData();

String json = data.getString("comment");

List<CustomerComment> commentList = new Gson().fromJson(json, new TypeToken<List<CustomerComment>>() {

}.getType());

CommentAdapter commentAdapter = new CommentAdapter(commentList, Community\_fullActivity.this);

commentListView = findViewById(R.id.commentList);

commentListView.setAdapter(commentAdapter);

setListViewHeightBasedOnChildren(commentListView);

}else if(msg.what==222){

String path = getString(R.string.server\_projectpath)+"selectCommentByD\_id.action?d\_id="+cd.getD\_id();

GetCommentThread getCommentThread = new GetCommentThread(handler,path);

getCommentThread.start();

}else if(msg.what==333){

Bundle data = msg.getData();

int count = data.getInt("count");

if(count==0){

like.setLikeCount(cd.getTotal\_like()+1);

String sql = "update dynamic set total\_like = total\_like+1 where d\_id='"+cd.getD\_id()+"'";

MysqlUtil mysqlUtil = new MysqlUtil(sql);

mysqlUtil.excute();

like.callOnClick();

}else{

}

}else if(msg.what==444) {

/\*\*

\* 这里小心用Log调试，如果返回为null，会报错闪退

\*/

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*In 444");

Bundle data = msg.getData();

cd.setGoods\_id(data.getString("goods\_id"));

Context context = getApplicationContext();

CharSequence text = null;

int duration = Toast.LENGTH\_SHORT;

Toast toast = null;;

if(cd.getGoods\_id() == null) {

text = "抱歉，本动态无对应商品!";

toast = Toast.makeText(context, text, duration);

toast.show();

} else {

text = "正在进入商城!";

toast = Toast.makeText(context, text, duration);

toast.show();

Log.i("cd.getGoods\_id() ", cd.getGoods\_id());

Bundle bundle = new Bundle();

bundle.putString("goods\_id", cd.getGoods\_id());

// Intent intent = new Intent(Community\_fullActivity.this, MallGoodsDetailActivity.class);

Intent intent = new Intent();

intent.setClass(Community\_fullActivity.this, MallGoodsDetailActivity.class);

intent.putExtras(bundle);

startActivity(intent);

}

}

}

};

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.community\_full);

ActionBar actionBar = getSupportActionBar();

actionBar.setDisplayHomeAsUpEnabled(true);

actionBar.setTitle("");

SharedPreferences sp = getSharedPreferences("customer",MODE\_PRIVATE);

String customerJson = sp.getString("customerJson","");

Gson gson = new Gson();

customer = gson.fromJson(customerJson,Customer.class);

Bundle bundle = getIntent().getBundleExtra("data");

String hostName = bundle.getString("hostName");

String hostHeadPic = bundle.getString("hostHeadPic");

final String hostU\_id = bundle.getString("u\_id");

ImageView headpic = findViewById(R.id.headpic);

headpic.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent();

intent.setClass(Community\_fullActivity.this, UserHomePage.class);

intent.putExtra("u\_id",hostU\_id);

startActivity(intent);

}

});

ImageLoad imageLoad = new ImageLoad(this);

imageLoad.loadImage(headpic,getString(R.string.server\_projectpath)+hostHeadPic);

cd = (Dynamic) bundle.getSerializable("dynamic");

TextView textView1 = findViewById(R.id.title);

TextView textView2 = findViewById(R.id.name);

TextView textView3 = findViewById(R.id.publishtime);

TextView textView4 = findViewById(R.id.content);

textView1.setText(cd.getTitle());

textView2.setText(hostName);

textView3.setText(cd.getRelease\_date());

textView4.setText(cd.getContent());

//首次进入，加载评论

String path = getString(R.string.server\_projectpath)+"selectCommentByD\_id.action?d\_id="+cd.getD\_id();

GetCommentThread getCommentThread = new GetCommentThread(handler,path);

getCommentThread.start();

//发表评论

Button send = findViewById(R.id.sendcomment);

etcomment = findViewById(R.id.etcomment);

send.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

String content = etcomment.getText().toString();

etcomment.setText("");

Comment comment = new Comment();

comment.setC\_id(System.currentTimeMillis()+""+Math.random()\*10);

comment.setContent(content);

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm+:ss");

String date = sdf.format(System.currentTimeMillis());

comment.setTime(date);

comment.setD\_id(cd.getD\_id());

comment.setU\_id(customer.getU\_id());

SaveCommentThread saveCommentThread = new SaveCommentThread(handler,comment);

saveCommentThread.start();

}

});

//给文章点赞

like = findViewById(R.id.likeView);

/\*\*

\* 这个控件在handler中用like.setHasLike()方法就会报错;

\* 只能先设置默认为点赞状态，如果是没点赞的就调用callonclick方法

\* 取消点赞，如果动态点赞为0的，就设置为1，避免出错

\*/

if(cd.getTotal\_like()==0){

like.setLikeCount(1);

}else {

like.setLikeCount(cd.getTotal\_like());

}

like.setHasLike(true);

GetLikelThread getLikelThread = new GetLikelThread(handler,cd.getD\_id(),customer.getU\_id());

getLikelThread.start();

like.setOnLikeListeners(new LikeView.OnLikeListeners() {

@Override

public void like(boolean isCancel) {

//isCancel

//如果设置了like\_canCancel为false，则isCancel可以不管，此时表示likeview被点击了

//如果设置了like\_canCancel为true,表示可以取消点赞，那么isCancel为true时，表示取消点赞，为false时，表示点赞

if(isCancel){

//取消点赞

String sql ="delete from liked where d\_id='"+cd.getD\_id()+"' and u\_id='"+customer.getU\_id()+"'";

MysqlUtil mysqlUtil = new MysqlUtil(sql);

mysqlUtil.excute();

sql = "update dynamic set total\_like = total\_like-1 where d\_id='"+cd.getD\_id()+"'";

mysqlUtil.setSql(sql);

mysqlUtil.excute();

}else{

//点赞

/\*try{\*/

String sql ="insert into liked value('"+cd.getD\_id()+"','"+customer.getU\_id()+"')";

MysqlUtil mysqlUtil = new MysqlUtil(sql);

mysqlUtil.excute();

sql = "update dynamic set total\_like = total\_like+1 where d\_id='"+cd.getD\_id()+"'";

mysqlUtil.setSql(sql);

mysqlUtil.excute();

/\*}catch(Exception e){

Toast.makeText(this,"你已经点过赞了")

}\*/

}

}

});

/\*\*

\* 监听商城链接的Button

\*/

Button button = (Button) findViewById(R.id.shopping);

button.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

/\*\*

\* 调用去对应商品的子线程

\*/

Get\_goods\_id gdi = new Get\_goods\_id(handler, cd.getD\_id());

gdi.start();

}

});

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

MenuInflater menuInflater = getMenuInflater();

menuInflater.inflate(R.menu.community\_topmenu,menu);

return super.onCreateOptionsMenu(menu);

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

switch (item.getItemId()) {

case android.R.id.home: //返回键的id

Bundle bundle =new Bundle();

String category = cd.getCategory();

bundle.putString("egtitle",category);

String name = "";

if(category.equals("fabric")){

name = "布艺";

}else if(category.equals("leather")){

name ="皮艺";

}else if(category.equals("paper")){

name = "纸艺";

}else if(category.equals("weave")){

name ="编织";

}else if(category.equals("ornaments")){

name ="饰品";

}else if(category.equals("embroidery")){

name ="刺绣";

}

bundle.putString("name",name);

Intent intent = new Intent();

intent.setClass(Community\_fullActivity.this,Community\_all.class);

intent.putExtra("title",bundle);

startActivity(intent);

return true;

default:

return super.onOptionsItemSelected(item);

}

}

/\*\*

\* 解决scrollview中嵌套listview只显示一行的问题

\* @param listView

\*/

public void setListViewHeightBasedOnChildren(ListView listView) {

// 获取ListView对应的Adapter

ListAdapter listAdapter = listView.getAdapter();

if (listAdapter == null) {

return;

}

int totalHeight = 0;

for (int i = 0, len = listAdapter.getCount(); i < len; i++) {

// listAdapter.getCount()返回数据项的数目

View listItem = listAdapter.getView(i, null, listView);

// 计算子项View 的宽高

listItem.measure(0, 0);

// 统计所有子项的总高度

totalHeight += listItem.getMeasuredHeight();

Log.e("hhhhhh",totalHeight+"");

}

ViewGroup.LayoutParams params = listView.getLayoutParams();

params.height = totalHeight+ (listView.getDividerHeight() \* (listAdapter.getCount() - 1));

// listView.getDividerHeight()获取子项间分隔符占用的高度

// params.height最后得到整个ListView完整显示需要的高度

listView.setLayoutParams(params);

}

}

# 后台代码

## DynamicMapper.java

package ssm.mapper.mineMapper;

import java.util.List;

import java.util.Map;

import org.apache.ibatis.annotations.Param;

import ssm.pojo.Mine.Customer;

import ssm.pojo.Mine.CustomerExample;

import ssm.pojo.Mine.Dynamic;

public interface DynamicMapper {

List<Dynamic> selectMyDynamic(String u\_id);

int selectDynamicLikeCount(String d\_id);

int selectDynamicCommentCount(String d\_id);

List<Dynamic> ProductUpSuggestion();

List<Dynamic> ProductDownSuggestion();

List<Map<String,Object>> findLatestPubDynamic(String category);

List<Map<String, Object>> findLatestRespDynamic(String category);

List<Map<String, Object>> findHostDynamic(String category);

}

## CustomerMapper.java

package ssm.mapper.mineMapper;

import java.util.List;

import org.apache.ibatis.annotations.Param;

import ssm.pojo.Mine.Customer;

import ssm.pojo.Mine.CustomerExample;

public interface CustomerMapper {

/\*\*

\* This method was generated by MyBatis Generator.

\* This method corresponds to the database table customer

\*

\* @mbggenerated Thu Mar 28 10:49:02 CST 2019

\*/

int countByExample(CustomerExample example);

Customer selectByUsernameAndPassword(Customer customer);

List<Customer> selectFollowDetail(String u\_id);

void updateCustomer(Customer customer);

/\*\*

\* This method was generated by MyBatis Generator.

\* This method corresponds to the database table customer

\*

\* @mbggenerated Thu Mar 28 10:49:02 CST 2019

\*/

int deleteByExample(CustomerExample example);

/\*\*

\* This method was generated by MyBatis Generator.

\* This method corresponds to the database table customer

\*

\* @mbggenerated Thu Mar 28 10:49:02 CST 2019

\*/

int deleteByPrimaryKey(String uId);

/\*\*

\* This method was generated by MyBatis Generator.

\* This method corresponds to the database table customer

\*

\* @mbggenerated Thu Mar 28 10:49:02 CST 2019

\*/

int insert(Customer record);

/\*\*

\* This method was generated by MyBatis Generator.

\* This method corresponds to the database table customer

\*

\* @mbggenerated Thu Mar 28 10:49:02 CST 2019

\*/

int insertSelective(Customer record);

/\*\*

\* This method was generated by MyBatis Generator.

\* This method corresponds to the database table customer

\*

\* @mbggenerated Thu Mar 28 10:49:02 CST 2019

\*/

List<Customer> selectByExample(CustomerExample example);

/\*\*

\* This method was generated by MyBatis Generator.

\* This method corresponds to the database table customer

\*

\* @mbggenerated Thu Mar 28 10:49:02 CST 2019

\*/

Customer selectByPrimaryKey(String uId);

/\*\*

\* This method was generated by MyBatis Generator.

\* This method corresponds to the database table customer

\*

\* @mbggenerated Thu Mar 28 10:49:02 CST 2019

\*/

int updateByExampleSelective(@Param("record") Customer record, @Param("example") CustomerExample example);

/\*\*

\* This method was generated by MyBatis Generator.

\* This method corresponds to the database table customer

\*

\* @mbggenerated Thu Mar 28 10:49:02 CST 2019

\*/

int updateByExample(@Param("record") Customer record, @Param("example") CustomerExample example);

/\*\*

\* This method was generated by MyBatis Generator.

\* This method corresponds to the database table customer

\*

\* @mbggenerated Thu Mar 28 10:49:02 CST 2019

\*/

int updateByPrimaryKeySelective(Customer record);

/\*\*

\* This method was generated by MyBatis Generator.

\* This method corresponds to the database table customer

\*

\* @mbggenerated Thu Mar 28 10:49:02 CST 2019

\*/

int updateByPrimaryKey(Customer record);

int findDynamicCount(String u\_id);

int findfollowCount(String u\_id);

int findfanCount(String u\_id);

List<Customer> selectFanDetail(String u\_id);

//后台管理

List<Customer> selectallCustomer();

Customer selectByU\_id(String u\_id);

void editCustomer(Customer customer);

void deleteCustomer(String u\_id);

void addCustomer(Customer customer);

List<Customer> SelectHighRisk();

public void updataCustomerCancellationCount(Customer customer);

public Customer findCustomerById(String u\_id);

/\*\*

\* 修改个人密码

\* @param customer

\*/

public void updateCustomerPwd(Customer customer);

/\*\*

\* 修改个人信息

\* @param customer

\*/

public void updateCustomerInfo(Customer customer);

}

## BigDataService.java

package ssm.service;

import java.util.List;

import ssm.pojo.Mine.Dynamic;

public interface BigDataService {

public List<Dynamic> ProductDownSuggestion();

public List<Dynamic> ProductUpSuggestion();

}

## DynamicService.java

package ssm.service;

import java.util.List;

import java.util.Map;

import ssm.pojo.Mine.Dynamic;

public interface DynamicService {

List<Map<String,Object>> findCommuityDynamic(String type, String category);

}

## BigDataServiceImpl.java

package ssm.service.impl;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import ssm.mapper.commentMapper.CommentMapper;

import ssm.mapper.mineMapper.CustomerMapper;

import ssm.mapper.mineMapper.DynamicMapper;

import ssm.pojo.Comment.CustomerComment;

import ssm.pojo.Mine.Customer;

import ssm.pojo.Mine.Dynamic;

import ssm.service.BigDataService;

import ssm.service.CommentService;

import ssm.service.CustomerService;

@Service

public class BigDataServiceImpl implements BigDataService {

@Autowired

DynamicMapper dynamicMapper;

@Override

public List<Dynamic> ProductDownSuggestion() {

// TODO Auto-generated method stub

List<Dynamic> DownSuggestionList = dynamicMapper.ProductDownSuggestion();

return DownSuggestionList;

}

@Override

public List<Dynamic> ProductUpSuggestion() {

// TODO Auto-generated method stub

List<Dynamic> UpSuggestionList = dynamicMapper.ProductUpSuggestion();

return UpSuggestionList;

}

public DynamicMapper getDynamicMapper() {

return dynamicMapper;

}

public void setDynamicMapper(DynamicMapper dynamicMapper) {

this.dynamicMapper = dynamicMapper;

}

}

## DynamicServiceImpl.java

package ssm.service.impl;

import java.util.List;

import java.util.Map;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import ssm.mapper.mineMapper.DynamicMapper;

import ssm.pojo.Mine.Dynamic;

import ssm.service.DynamicService;

@Service

public class DynamicServiceImpl implements DynamicService {

@Autowired

DynamicMapper dynamicMapper;

@Override

public List<Map<String,Object>> findCommuityDynamic(String type, String category) {

List<Map<String,Object>> list = null;

//判断type的类型

if(type.equals("latestpub")) {

list = dynamicMapper.findLatestPubDynamic(category);

}else if(type.equals("latestresp")) {

list = dynamicMapper.findLatestRespDynamic(category);

}else if(type.equals("host")){

list = dynamicMapper.findHostDynamic(category);

}

return list;

}

}

## BigDataHandler.java

package ssm.controller;

import java.io.IOException;

import java.io.InputStream;

import java.io.OutputStream;

import java.io.UnsupportedEncodingException;

import java.util.List;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.servlet.ModelAndView;

import com.google.gson.Gson;

import ssm.pojo.Mine.Customer;

import ssm.pojo.Mine.Dynamic;

import ssm.service.\*;

import ssm.service.impl.\*;

@Controller

public class BigDataHandler {

@Autowired

private BigDataService bigDataService;

public BigDataService getBigDataService() {

return bigDataService;

}

public void setBigDataService(BigDataService bigDataService) {

this.bigDataService = bigDataService;

}

@RequestMapping("/ProductDownSuggestion.action")

@ResponseBody

public ModelAndView ProductDownSuggestion() {

ModelAndView modelAndView = new ModelAndView();

List<Dynamic> dynamicList = bigDataService.ProductDownSuggestion();

modelAndView.addObject("dynamicList",dynamicList);

for (Dynamic dynamic : dynamicList) {

System.out.println(dynamic);

}

modelAndView.setViewName("jsps/ProductDownSuggestion.jsp");

return modelAndView;

}

@RequestMapping("/ProductUpSuggestion.action")

public ModelAndView ProductUpSuggestion() {

ModelAndView modelAndView = new ModelAndView();

List<Dynamic> dynamicList = bigDataService.ProductUpSuggestion();

modelAndView.addObject("dynamicList",dynamicList);

for (Dynamic dynamic : dynamicList) {

System.out.println(dynamic);

}

modelAndView.setViewName("jsps/ProductUpSuggestion.jsp");

return modelAndView;

}

}

## CustomerHandler.java

package ssm.controller;

import java.io.ByteArrayOutputStream;

import java.io.IOException;

import java.io.InputStream;

import java.io.OutputStream;

import java.io.UnsupportedEncodingException;

import java.util.List;

import javax.servlet.ServletInputStream;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.servlet.ModelAndView;

import com.google.gson.Gson;

import ssm.pojo.Mine.Customer;

import ssm.pojo.Mine.Dynamic;

import ssm.service.CustomerService;

import ssm.service.impl.CustomerServiceImpl;

@Controller

public class CustomerHandler {

@Autowired

private CustomerService customerService;

@RequestMapping("/login.action")

@ResponseBody

public void login(HttpServletRequest req,HttpServletResponse resp) throws IOException {

resp.setContentType("text/html;charset=utf-8");

InputStream is = req.getInputStream();

byte[] b = new byte[1024];

int len = is.read(b);

String str = new String(b,0,len);

System.out.println(str);

Gson gson = new Gson();

Customer customer = gson.fromJson(str, Customer.class);

System.out.println(customer);

customer = customerService.selectByUsernameAndPassword(customer);

System.out.println(customer);

if(customer!=null) {

OutputStream os = resp.getOutputStream();

String json = gson.toJson(customer);

System.out.println(json);

os.write(json.getBytes("UTF-8"));

os.flush();

}else {

OutputStream os = resp.getOutputStream();

os.write("".getBytes());

}

return;

}

@RequestMapping("/findFollow\_Fan.action")

@ResponseBody

public void findFollow\_Fan(HttpServletRequest req,HttpServletResponse resp) throws IOException {

resp.setContentType("text/html;charset=utf-8");

String u\_id = req.getParameter("u\_id");

String data = customerService.findFollow\_Fan(u\_id);

OutputStream os = resp.getOutputStream();

os.write(data.getBytes("UTF-8"));

os.flush();

return;

}

@RequestMapping("/findFollowDetail.action")

@ResponseBody

public void findFollowDetail(HttpServletRequest req,HttpServletResponse resp) throws IOException {

resp.setContentType("text/html;charset=utf-8");

String u\_id = req.getParameter("u\_id");

List<Customer> list = customerService.selectFollowDetail(u\_id);

Gson gson = new Gson();

String data = gson.toJson(list);

OutputStream os = resp.getOutputStream();

//处理乱码问题

os.write(data.getBytes("UTF-8"));

os.flush();

return ;

}

@RequestMapping("/findFanDetail.action")

@ResponseBody

public void findFanDetail(HttpServletRequest req,HttpServletResponse resp) throws IOException {

resp.setContentType("text/html;charset=utf-8");

String u\_id = req.getParameter("u\_id");

List<Customer> list = customerService.selectFanDetail(u\_id);

for(Customer c: list) {

System.out.println(c);

}

Gson gson = new Gson();

String data = gson.toJson(list);

OutputStream os = resp.getOutputStream();

//处理乱码问题

os.write(data.getBytes("UTF-8"));

os.flush();

return ;

}

@RequestMapping("/updateCustomerPwd.action")

@ResponseBody

public String updateCustomerPwd(HttpServletRequest request,HttpServletResponse response) throws IOException {

ServletInputStream is = request.getInputStream();

ByteArrayOutputStream baos = new ByteArrayOutputStream();

byte[] buffer = new byte[1024];

int len = -1;

while ((len = is.read(buffer)) != -1) {

System.out.println("len=" + len);

baos.write(buffer, 0, len);

}

String json = baos.toString();

Customer customer = new Gson().fromJson(json, Customer.class);

System.out.println("customer = " + customer);

customerService.updateCustomerPwd(customer);

String res = "success";

try {

OutputStream os = response.getOutputStream();

os.write(res.getBytes("utf-8"));

os.flush();

os.close();

} catch (Exception e) {

e.printStackTrace();

}

return res;

}

@RequestMapping("/updateCustomerInfo.action")

@ResponseBody

public String updateCustomerInfo(HttpServletRequest request,HttpServletResponse response) throws IOException {

ServletInputStream is = request.getInputStream();

ByteArrayOutputStream baos = new ByteArrayOutputStream();

byte[] buffer = new byte[1024];

int len = -1;

while ((len = is.read(buffer)) != -1) {

System.out.println("len=" + len);

baos.write(buffer, 0, len);

}

String json = baos.toString();

Customer customer = new Gson().fromJson(json, Customer.class);

System.out.println("customer = " + customer);

customerService.updateCustomerInfo(customer);

String res = "success";

try {

OutputStream os = response.getOutputStream();

os.write(res.getBytes("utf-8"));

os.flush();

os.close();

} catch (Exception e) {

e.printStackTrace();

}

return res;

}

//后台管理

@RequestMapping("/findallCustomer.action")

public ModelAndView findallCustomer() {

ModelAndView modelAndView = new ModelAndView();

List<Customer> customerList = customerService.findallCustomer();

modelAndView.addObject("customerList",customerList);

for (Customer customer : customerList) {

System.out.println(customer);

}

modelAndView.setViewName("jsps/customer.jsp");

return modelAndView;

}

@RequestMapping("/editCustomerPre.action")

public ModelAndView editCustomerPre(String u\_id) {

ModelAndView modelAndView = new ModelAndView();

Customer customer = customerService.findCustomer(u\_id);

modelAndView.addObject(customer);

System.out.println(customer);

modelAndView.setViewName("jsps/editCustomer.jsp");

return modelAndView;

}

@RequestMapping("/editCustomer.action")

public ModelAndView editCustomer(Customer customer,HttpServletRequest req,HttpServletResponse resp) throws UnsupportedEncodingException {

req.setCharacterEncoding("UTF-8");

resp.setContentType("text/html;charset=utf-8");

System.out.println(customer);

ModelAndView modelAndView = new ModelAndView();

customerService.editCustomer(customer);

modelAndView.setViewName("findallCustomer.action");

return modelAndView;

}

@RequestMapping("/deleteCustomer.action")

public ModelAndView deleteCustomer(String u\_id,HttpServletRequest req,HttpServletResponse resp) throws UnsupportedEncodingException {

req.setCharacterEncoding("UTF-8");

resp.setContentType("text/html;charset=utf-8");

ModelAndView modelAndView = new ModelAndView();

customerService.deleteCustomer(u\_id);

modelAndView.setViewName("findallCustomer.action");

return modelAndView;

}

@RequestMapping("/addCustomer.action")

public ModelAndView addCustomer(Customer customer,HttpServletRequest req,HttpServletResponse resp) throws UnsupportedEncodingException {

req.setCharacterEncoding("UTF-8");

resp.setContentType("text/html;charset=utf-8");

ModelAndView modelAndView = new ModelAndView();

customer.setU\_id(System.currentTimeMillis()+"");

customerService.addCustomer(customer);

modelAndView.setViewName("findallCustomer.action");

return modelAndView;

}

@RequestMapping("/SelectHighRisk.action")

public ModelAndView SelectHighRisk() {

ModelAndView modelAndView = new ModelAndView();

List<Customer> customerList = customerService.SelectHighRisk();

modelAndView.addObject("customerList",customerList);

for (Customer customer : customerList) {

System.out.println(customer);

}

modelAndView.setViewName("jsps/SelectHighRisk.jsp");

return modelAndView;

}

}