

**实 训 报 告**

**实训名称：** 工程与技能实训

**院 系：** 计算机科学与工程学院

**专 业：** 计算机科学与技术

**班　　级：**

**学 号：**

**姓 名：**

**指导教师：**

**开课时间：** 2018 至 2019 年第 一 学期

常熟理工学院计算机科学与工程学院 制

# 第一章 引言

## 1.1 研究背景及意义

随着当今社会的科学技术和社会经济的飞速发展，计算机也从以前的鲜为人知变成如今的大众化、普遍化。当人们学会了使用计算机来开发程序，开发软件，人们竟可以将一些需要人力去完成的事情交给计算机来做，这样提高了生产力，节约了人力资源。短信验证码是通过发送验证码到手机的一种有效的验证码系统。无论是大型网站尤其是购物网站，都提供有手机短信验证码功能，可以比较准确和安全地保证购物的安全性，验证用户的正确性。某些验证码接入商提供手机短信验证码服务，各网站通过接口发送请求到接入商的服务器，服务器发送随机数字或字母到手机中，由接入商的服务器统一做验证码的验证。目前使用的最普遍的有各大银行[网上银行](https://baike.baidu.com/item/%E7%BD%91%E4%B8%8A%E9%93%B6%E8%A1%8C" \t "https://baike.baidu.com/item/%E7%9F%AD%E4%BF%A1%E9%AA%8C%E8%AF%81%E7%A0%81/_blank)、[网上商城](https://baike.baidu.com/item/%E7%BD%91%E4%B8%8A%E5%95%86%E5%9F%8E" \t "https://baike.baidu.com/item/%E7%9F%AD%E4%BF%A1%E9%AA%8C%E8%AF%81%E7%A0%81/_blank)、团购网站、票务公司等。短信验证码利用短信验证码来注册会员，大大降低了非法注册，烂注册的数据。而且目前用户人均手机数目较多，多部手机接收验证码不方便人们的生活需求，我们通过自动截取信息中的验证码集中处理，可以在一部手机上同时获取多部手机的验证码，方便人们的生活。

计算机现在已经渗透了各个领域，成为各种行业必不可少的用品，随着Internet技术的发展，网路信息大道也越来越宽广，随着时代信息化的推进，大量的数据信息是必然的，所以对数据库更进一步的研究和运用也是必不可少的。

## 1.2 技术的介绍

该项目主要运用了Java技术和MYSQL数据库技术的相结合。Java语言的优点就是跨平台性强、面向对象，它可以在很多的操作系统上运行，可移植性非常高，而且有简单，安全可靠，多线程等特点，并且还是动态语言。Java的跨平台性的意思是：Java写的程序不需要再去修改，就可以在不同的系统上运行。比如：在windows XP系统上写的程序，不用修改就可以在Linux系统上面运行，Java之所以有这么强大的跨平台性完全是因为Java虚拟机（JVM）。JVM具体可以说成是一个抽象机器，它直接装在操作系统上面，它有着自己一套完整的操作指令，有自己的存储器等，因此它就能实现Java的跨平台性，但JVM一般都在软件上面实现。Java的面向对象就是将现实生活中的实物，实例所具有的属性，行为映射到计算机中去， 然后对这些属性，行为进行处理。Java中的封装就是把对象的属性，行为联合在一起形成整体。Java的继承就是说一个对象可以直接使用另一个对象的属性和方法。Java的多态就是具有多种表现形式。并且Java的多线程更是独具一格，多线程提高了程序的运行速率，并且方便开发人员开发多线程程序，只需要调用Java自身提供的Thread类就可以，它可以启动运动和终止线程还可以检查线程的状态。

MYSQL服务器具有快速，多线程，多用户的特点。它能在不同的操作系统运行。MYSQL查询功能方便，而且还支持ODBC for Windows，支持大数据库，并且还有一个灵活、安全的权限和口令系统，MYSQL还有快速且稳定的基于线程的内存分配系统。综上所述，所以在这次开发实践中选择了Java技术和MYSQL技术的相结合运用。

## 1.3 系统实现技术的简介

在本系统中主要运用Andoid,tomcat和MYSQL数据库服务技术。

Java技术中,主要运用JDBC连接数据库，servlet技术连接web前端。

MYSQL数据库的操作：首先连接数据库，然后才能使用sql语句。我们在仓库管理系统中建一个编辑数据库的类，就可以直接对数据库操作，进行我们需求的增加，删除，修改，查询等操作。

# 第二章 理论背景

## 2.1 servlet技术

Servlet（Server Applet）是[Java](https://baike.baidu.com/item/Java/85979" \t "https://baike.baidu.com/item/servlet/_blank) Servlet的简称，称为小服务程序或服务连接器，用Java编写的[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8/100571" \t "https://baike.baidu.com/item/servlet/_blank)端程序，具有独立于平台和[协议](https://baike.baidu.com/item/%E5%8D%8F%E8%AE%AE/13020269" \t "https://baike.baidu.com/item/servlet/_blank)的特性，主要功能在于交互式地浏览和生成数据，生成动态[Web](https://baike.baidu.com/item/Web/150564" \t "https://baike.baidu.com/item/servlet/_blank)内容。

Servlet 的主要功能在于交互式地浏览和修改数据，生成动态 Web 内容。这个过程为：

1、客户端发送请求至服务器端；

2、服务器将请求信息发送至 Servlet；

3、Servlet 生成响应内容并将其传给[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "https://baike.baidu.com/item/servlet/_blank)。响应内容动态生成，通常取决于客户端的请求；

4、服务器将响应返回给客户端。

## 2.2 Tomcat的简介

大家都知道Servlet，那么Tomcat就是Servlet的运行环境也可以说是Servlet的容器，它是在JSWOK基础上才发展起来的，是JSP和Servlet的规范实现。Servlet可以在Java编写的服务器上进行直接运行，它和普通Java类的不同就在于它是运行在服务器端的，使用Servlet可以扩展Java网络服务器的功能，可以为用户提供一个安全性高，移植性强的动态网页。

# 第三章 需求的分析

## 3.1 可行性

可行性其实说白了就是达到用最小的代价怎么才能换取最大的利益。其中包含经济可行性与技术可行性还有操作可行性。

### 3.1.1 技术的可行性

该系统的前台数据库采用的是MYSQL数据库，而MYSQL数据库在安全性还有准确性上相比较其它数据库占有非常大的优势的，而且运行速度也快，还能同时处理大量数据，效率高。

该管理系统的后台Microsoft公司发行的MyEclise8.6作为整个系统的开发平台，这个平台中有一系列已经很完善的指令控制语句、封装了丰富的类，对象和数据类型，让开发变得更加方便，高效。

综上所述，该管理系统的开发和设计在技术上和设备上的条件都是很理想的。所以技术上是可行的。

### 3.1.3 操作的可行性

该软件是根据用户需求，结合实际工作要求开发的，对手机的要求不是很高，符合对应部门实际工作的环境。

从上面来看，技术可行性是完全符合的，实用性高，经济性强。在该系统中对用户没有什么要求，用户只需要在手机端登陆，就能使用，可以满足用户的需求，到达开发该软件的目的。所以，该软件在操作运行上是可行的。

## .2 验证码管理系统的分析

### 3.2.1 验证码管理系统的需求分析

验证码管理系统的需求分析是对该系统的全面检查，在开发中尤为重要，对于系统需求分析就是：系统要做什么，达到什么目的。

### 3.2.2 验证码管理系统的功能分析

系统功能就是客户的需求，客户需要什么功能就做什么功能，而在这个仓库管理系统中，通过分析需求说明，总结出以下功能：

1. 用户信息管理
2. 用户手机短信管理
3. 自动获取手机短信验证码

这三个模块就是该系统要实现的所有需求功能。

用图片的展示如下：

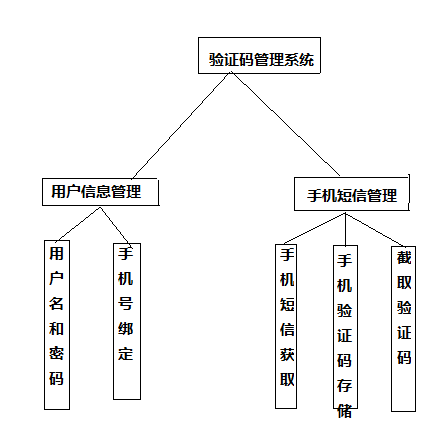


图 3-1 系统功能图

### 3.2.3 用户的登录流程示意图

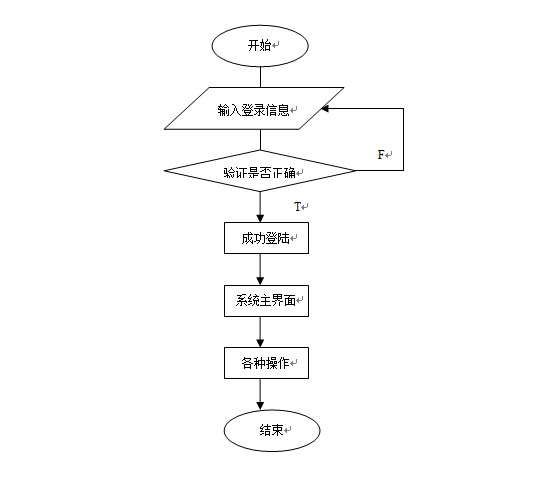


图3-5 登陆流程图

# **第四章 数据库的设计模块**

## 4.1 数据模型的分析和设计

数据库模型主要的功能就是描述数据库中数据的结构化和操纵数据的方法。其中数据库的模型的结构部分规范了数据如何被读取，比如树和表等

数据库模型的操做部分规定了对数据的增添、删除、自动排序、修改和查看等操作。

数据库模型的主要类别有分层模型、关系模型、网络模型和对象模型等四种基本模型。

而在该管理系统中运用的是关系模型，关系模型是当前编程中非常重要的一种模型。在关系数据库的系统中采用了关系模型作为数据的基本管理方式。

以下几种优点也是我们选择关系数据模型的关键：

A .关系模型和别的数据模型有很大的区别，它有非常严格的数学概念作为基础，能够很大的提高准确性。

B. 关系模型只有单一的概念。不管是实体或是实体之间的相互联系都能用关系模型表示出来。对数据的检查、搜索结构也是一种关系，也就是表。综上所述，关系模型的数据结构简单、一目了然，用户体验更加方便简洁。

C.关系模型的存储、读取路径对完全对用户开放，这样是的数据库具有良好的数据独立性和数据安全性，提高了数据的保密性，也方便了开发者的工作效率，开发者更能快速有效的进行开发。

正式因为上面众多的优点，关系数据模型在被发行后迅速的发展起来，被广泛的运用，深受广大用户的喜爱，而这也正是该管理系统选择使用关系数据模型的关键。

当然了，金无足赤、人无完人，什么东西都是这样的，关系数据模型也有其自身存在的缺点，关系数据模型中最大的缺点就是存储和读取的路径对用户完全开放，因此查询得速度很多时候没有非关系数据模型的速度快。由于存在这个缺点，为了提高系统的性能，在设计用户查询请求这一块是进行了很多的优化，提高了数据库管理方面开发的难度。

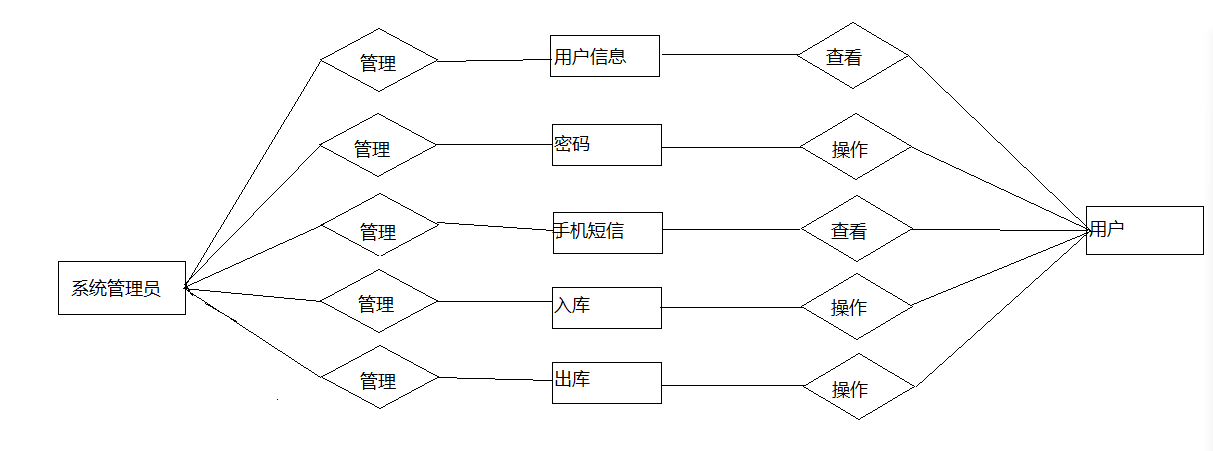


图 4-1 数据关系模型图

该系统中的关系数据模型的数据结构具体用下面的几张表来表示：

## 4.2 数据库表的结构

userm表，表格结构userid用的INT类型，username，password用的varchar类型。如表4-1。

表 4-1 userm

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 数据类型 | 主键 | NULL |
| userid | int(10) | 是 | 不是 |
| username | varchar(255) | 不是 | 是 |
| password | varchar(255) | 不是 | 不是 |

message表，表结构message\_id用的int类型，phone,message用的varchar类型.如表4-2。

表 4-2 message

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 数据类型 | 主键 | NULL |
| message\_id | int(10) | 是 | 不是 |
| phone | varchar(255) | 不是 | 是 |
| message | varchar(255) | 不是 | 不是 |

um表，表结构userid,message\_id用的int类型，如表4-3。

表 4-3 um

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 数据类型 | 主键 | NULL | 外键 |
| userid | int(10) | 是 | 不是 | 是 |
| message\_id | int(10) | 是 | 不是 | 是 |

## 4.3 详细设计

本模块具体介绍了短信插入时，代码及用户页面的实现。

短信插入模块是系统登陆成功进入的主操作页面的一个功能模块。

在本系统中，短信插入读取是相联合的。

**public** Message ReadMessage(String phone) {  
 Message message=**new** Message();  
 String sql=**"select** *\** **from message where phone=?;"**;  
 Connection connection= JDBCUtil.*getConnection*();  
 PreparedStatement preparedStatement;  
 ResultSet resultSet;  
 **try** {  
 preparedStatement=connection.prepareStatement(sql);  
 preparedStatement.setString(1,phone);  
 resultSet=preparedStatement.executeQuery();  
 **while** (resultSet.next()){  
 String id=resultSet.getString(1);  
 String ph=resultSet.getString(2);  
 String msg=resultSet.getString(3);message.setMid(id);  
 message.setPhone(ph);  
 message.setMessage(msg);  
 message.setCode(200);  
 }  
 } **catch** (Exception e) {  
 message.setCode(100);  
 e.printStackTrace();  
 }  
 **return** message;  
}

以上为短信插入部分的代码，通过连接数据库。

### 4.3.1 短信插入流程

进入短信插入界面，点击确认，会出现主页面，然后填写信息，填写符合要求，会弹出成功页面，不符合则不能插入，提示错误信息，以下是短信插入流程图：

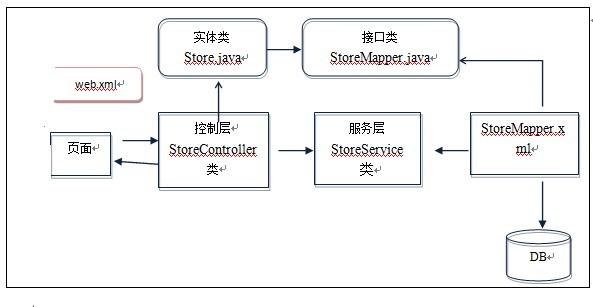


图 4-2 短信入库流程图

# 软件的设计

验证码管理系统是通过自动获取手机短信并且通过算法截取其中的验证码统一集中到用户随身携带的手机上，方便用户及时查收验证码。

1. 硬件设备环境

* CPU：P4 3.0G以上；
* 内存：1GB以上；
* 硬盘：100G以上可用存储空间；

1. 软件设备环境

* 数据库：MYSQL5.5以上；
* 应用服务器：Tomcat6.0以上；
* JDK版本：1.7及以上；
* 操作系统：Windows 7及以上版本。

# 第六章 系统的实现

## 6.1 用户登录简介

### 6.1.1登录界面

登陆界面的主要就是进行用户验证，用户登陆时页面会发送请求，从后台查看信息，如果数据库有该条信息则登陆成功，若没有则登陆失败。

下图为登陆界面：

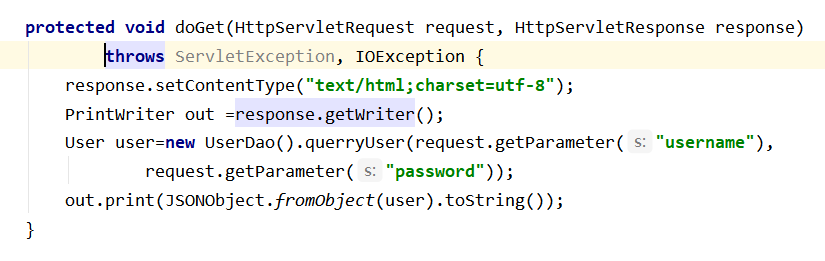


图 6-1 登陆界面图

以下是用户登陆页面的客户端实现代码：

private void loginTX(final String username, String password) {  
 final String url = getString(R.string.ip\_config) + "/MessageBox\_Server\_beta/userLogin?username=" + username + "&password=" + password;  
 new Thread(new Runnable() {  
 @Override  
 public void run() {  
  
 HttpUtil.getUrlConnectJson(LoginActivity.this, url, new OnHttpRepsonLinstener() {  
 @Override  
 public void onGetString(String json) {  
 try {  
 //Toast.makeText(LoginActivity.this, "URL=" + url, Toast.LENGTH\_LONG).show();  
 JSONObject jsonObject = new JSONObject(json);  
 int code = jsonObject.getInt("code");  
 //String id=jsonObject.getString("id");  
 if (code == 200) {  
 Toast.makeText(LoginActivity.this, "登录成功", Toast.LENGTH\_LONG).show();  
 //跳转到主界面  
 Intent intent = new Intent(LoginActivity.this, MainActivity.class);  
 intent.putExtra("username", username);  
 intent.putExtra("msgArray",MsgArray);  
 startActivityForResult(intent, 1000);  
 finish();  
 } else {  
 Toast.makeText(LoginActivity.this, "用户名或者密码错误", Toast.LENGTH\_LONG).show();  
 }  
 } catch (JSONException e) {  
  
 e.printStackTrace();  
 }  
 }  
 }  
 );  
 }  
 }

以下是用户登陆页面的服务器实现代码：



以上代码是程序开始运行代码，代码运行时首先要获取从页面上输入的用户名和密码，系统会自动调用userDao.querryUser这个方法，从已经建立连接的数据库中查询出这条记录，若是数据库中存在这条记录，客户端会提示“登陆成功”弹窗；若查到的记录为空，依旧停在登录页面，会有弹窗提示用户“账号或密码错误”。

### 6.1.1注册界面

注册界面的主要就是进行用户注册，用户注册时页面会发送请求，将账号面膜添加到后台，如果数据库有不存在重复的用户名则允许写入，若没有则注册失败。



下面是注册端的主要逻辑代码

private void registTX(String username, String password) {  
 //final String url = getString(R.string.ip\_config)+"/message/uRegistServlet?username="+username+"&password="+password;  
 final String url = getString(R.string.ip\_config) + "/MessageBox\_Server\_beta/userRegist?username=" + username + "&password=" + password;  
 //Toast.makeText(LoginActivity.this,"URL="+url,Toast.LENGTH\_LONG).show();  
 new Thread(new Runnable() {  
 @Override  
 public void run() {  
  
 HttpUtil.getUrlConnectJson(RegistActivity.this, url, new OnHttpRepsonLinstener() {  
 @Override  
 public void onGetString(String json) {  
 try {  
 //Toast.makeText(LoginActivity.this, "URL=" + url, Toast.LENGTH\_LONG).show();  
 JSONObject jsonObject = new JSONObject(json);  
 int code = jsonObject.getInt("code");  
 if (code == 200) {  
 Toast.makeText(RegistActivity.this, "注册成功", Toast.LENGTH\_LONG).show();  
 } else {  
 Toast.makeText(RegistActivity.this, "注册失败，用户名已经被注册过了", Toast.LENGTH\_LONG).show();  
 }  
 } catch (JSONException e) {  
  
 e.printStackTrace();  
 }  
 }  
 }  
 );  
 }  
 }  
  
  
 ).start();

使用注册界面注册成功之后，就可以享受远程登录、短信云管理等服务。

## 6.2 主操作界面

### 6.2.1 主操作界面

该系统登陆进去的第一个页面。

下图为系统主界面图：

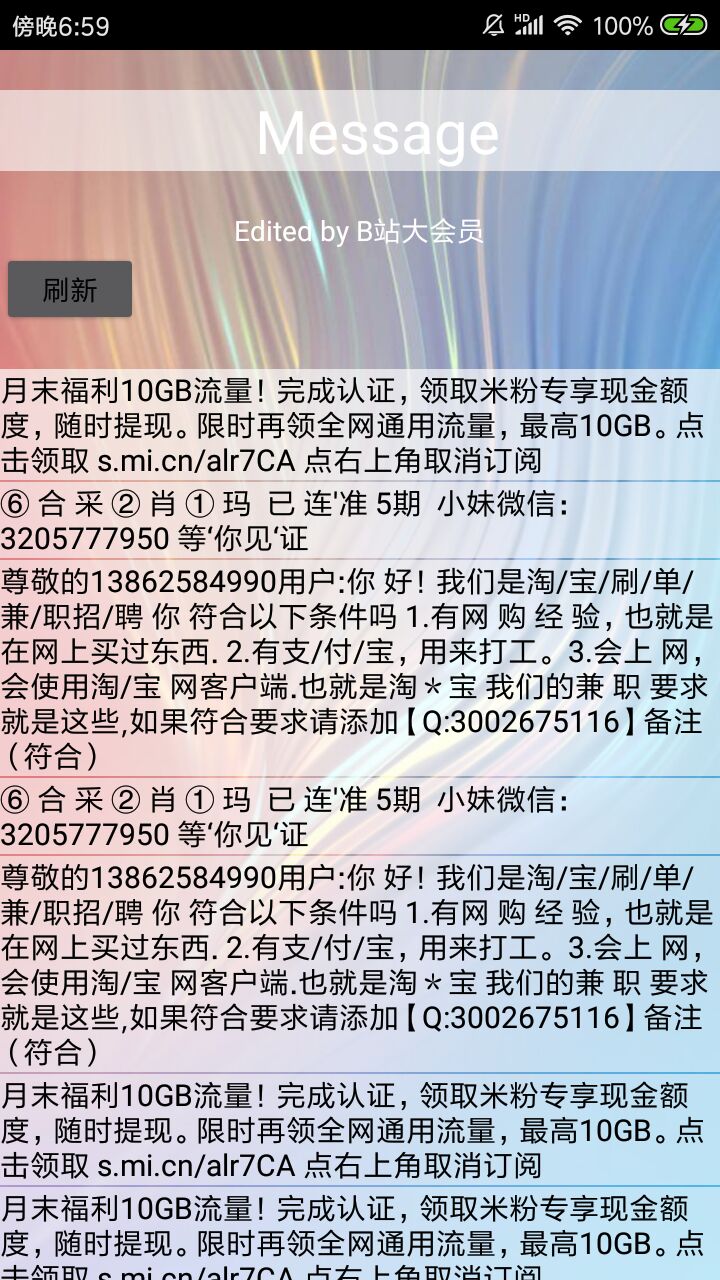
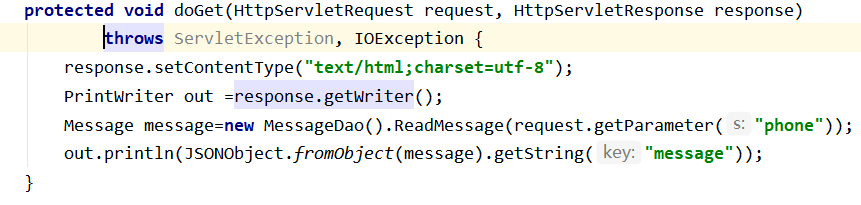


图 6-2 系统主界面图

以下是服务器反馈主页面的实现代码



以上代码是程序开始运行代码，代码运行时首先要获取从页面上输入的用户的手机号，系统会自动调用messageDao.ReadMessage()这个方法，从已经建立连接的数据库中查询出这条记录，若是数据库中存在这条记录，客户端会提示“信息“查询成功”弹窗；若查到的记录为空，依旧停在登录页面，会有弹窗提示用户“失败”。

### 6.2.2 信息插入界面

信息插入功能部分代码如下：



如上代码所示，这个模块主要是管理手机短信插入

手机端自动执行截取短信验证码并且发送，下面是客户端的实例代码。

public void getSmsFromPhone() {  
 new Thread(new Runnable() {  
 @Override  
 public void run() {  
 ContentResolver cr = getContentResolver();  
 String[] projection = new String[]{"body"};//"\_id", "address", "person",, "date", "type  
 //String where = " address = '10690630746185017590' AND date > " + (System.currentTimeMillis() - 10 \* 60 \* 1000);  
 //String where = " address = '10690630746185017590'";  
 //Cursor cur = cr.query(SMS\_INBOX, projection, where, null, "date desc");  
 Cursor cur = cr.query(SMS\_INBOX, projection, null, null, "date desc");  
 Log.e("getSmsFromPhone!","sdfasdfas");  
 if (null == cur)  
 return;  
 while(cur.moveToNext()) {  
 //String number = cur.getString(cur.getColumnIndex("address"));//手机号  
 //String name = cur.getString(cur.getColumnIndex("person"));//联系人姓名列表  
 String body = cur.getString(cur.getColumnIndex("body"));  
  
 //System.out.println("Body="+body);  
 //这里我是要获取自己短信服务号码中的验证码~~  
 // Pattern pattern = Pattern.compile(" [a-zA-Z0-9]{10}");  
 Pattern pattern=Pattern.compile("[验证]");  
 Matcher matcher = pattern.matcher(body);  
 System.out.println("find match="+body);  
 if (matcher.find()) {  
 System.out.println("find match="+body);  
 MsgArray.add(body);  
 //String res = matcher.group(1);  
 //mobileText.setText(res);  
 //Log.e(  
 // "GetMsg=" , res);  
 }  
  
 }  
 System.out.print("123");  
  
 }  
 }

### 6.3.3 信息读取界面

如图所示，读取的部分实现代码如下：:

**protected void** doGet(HttpServletRequest request, HttpServletRespone response)  
 **throws** ServletException, IOException {  
 response.setContentType(**"text/html;charset=utf-8"**);  
 PrintWriter out =response.getWriter();  
 Message message=**new** MessageDao().ReadMessage(request.getParameter(**"phone"**));  
 out.println(JSONObject.*fromObject*(message).getString(**"message"**));  
}

以下是客户端主页的主要逻辑代码

//读取Message

List<String> listdata = new ArrayList<String>();

private void readMessageTX(String username)

{

//创建ListView

listview = (ListView)findViewById(R.id.message\_list);

final ArrayAdapter<String> arrayAdapter = new ArrayAdapter<String>(this, R.layout.list\_item, listdata);//listdata和str均可

final String url = getString(R.string.ip\_config)+"/MessageBox\_Server\_beta/readMessage?username="+username;

new Thread(new Runnable() {

@Override

public void run() {

HttpUtil.getUrlConnectJson(MainActivity.this, url, new OnHttpRepsonLinstener() {

@Override

public void onGetString(String json) {

try {

//Toast.makeText(LoginActivity.this, "URL=" + url, Toast.LENGTH\_LONG).show();

JSONObject jsonObject = new JSONObject(json);

//int code=jsonObject.getInt("code");

JSONArray message = jsonObject.getJSONArray("package");

JSONObject packageObject=message.getJSONObject(0);

int code=packageObject.getInt("code");

if (code == 200) {

// Toast.makeText(MainActivity.this, "读取数据库中", Toast.LENGTH\_LONG).show();

for (int i=0;i<100;i++)

{

try {

JSONObject packageObjectx=message.getJSONObject(i);

if(packageObject!=null)

listdata.add(packageObjectx.getString("message"));}

catch (JSONException e){e.printStackTrace();}

//System.out.println("message="+packageObjectx.getString("message"));

//Toast.makeText(MainActivity.this, "message="+packageObjectx.getString("message"), Toast.LENGTH\_LONG).show();

}

//显示ListView

arrayAdapter.notifyDataSetChanged();

listview.setAdapter(arrayAdapter);

} else {

Toast.makeText(MainActivity.this, "用户还没有上传数据", Toast.LENGTH\_LONG).show();

}

} catch (JSONException e) {

e.printStackTrace();

}

}

}

);

}

}

这个模块主要就是管理已在库中的短信，并且读取信息到手机端，通过手机端的验证码截取功能成功获取验证码。

第七章 测试模块

## 7.1 系统测试

系统测试的目的是为了证明这个系统是非常正确的，没有任何错误，弱势这样考虑，那在开发过程中就可以设计一些比较稳定的方案，不用去检查，检测系统的隐患就行了。但是作为一个开发者是不可能这样去考虑问题的，开发者要要反复的测试系统的漏洞和错误程序，保证程序的完美运行。

在一个开发者眼中，测试的目的有：

1. 检查程序中因为错误而执行的程序；
2. 检测当中可能会发现未知的错误；
3. 完善测试方案。

## 7.2 软件测试

软件测试的目的就是确保软件的质量，使软件在运行时出错的几率降到最小，从而达到测试的目的。

7.2.1 SQL注入测试

测试方式:针对不同的接口进行黑盒测试

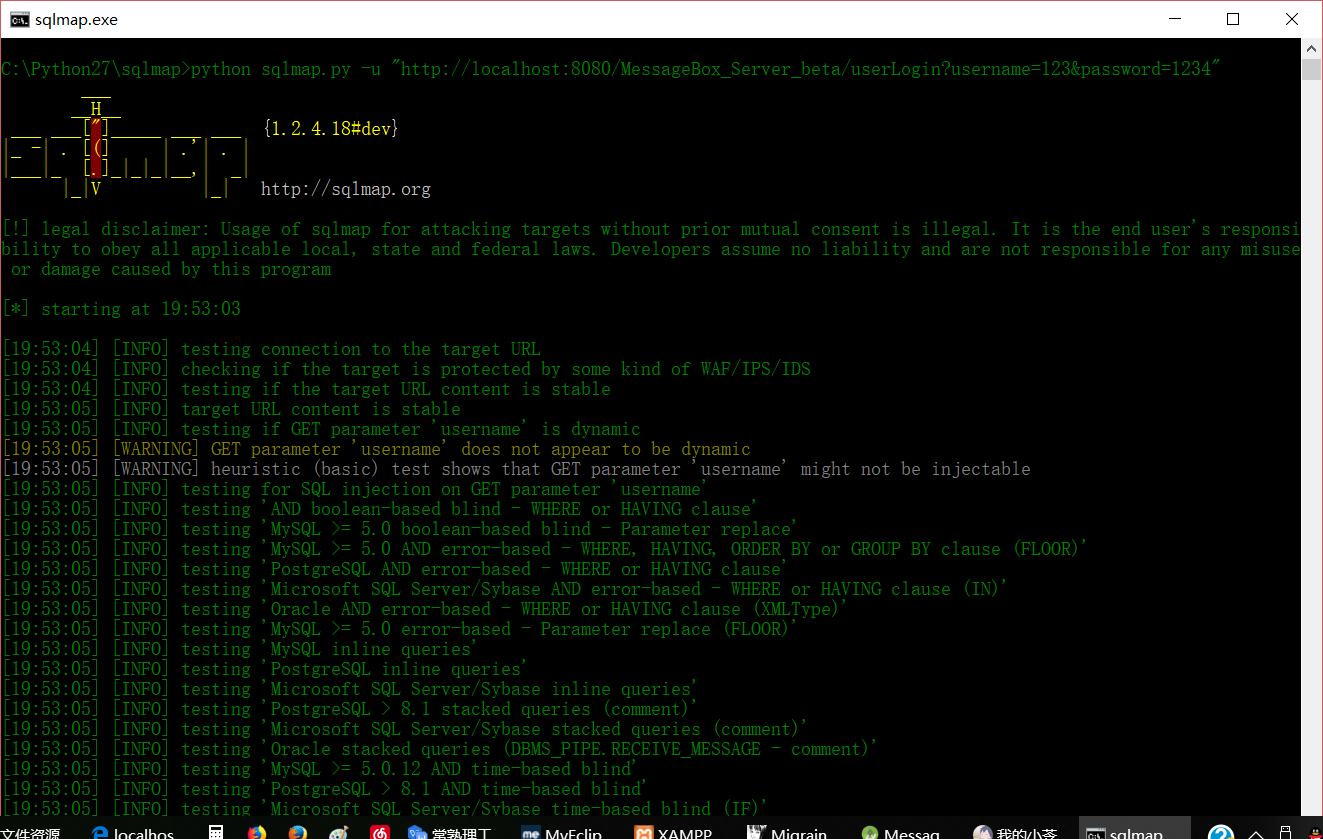
测试类型:安全性测试

测试工具:SQLMAP

（1）登录接口测试

测试URL:

<http://localhost:8080/MessageBox_Server_beta/userLogin?username=123&password=1234>



Username和password都不存在注入点

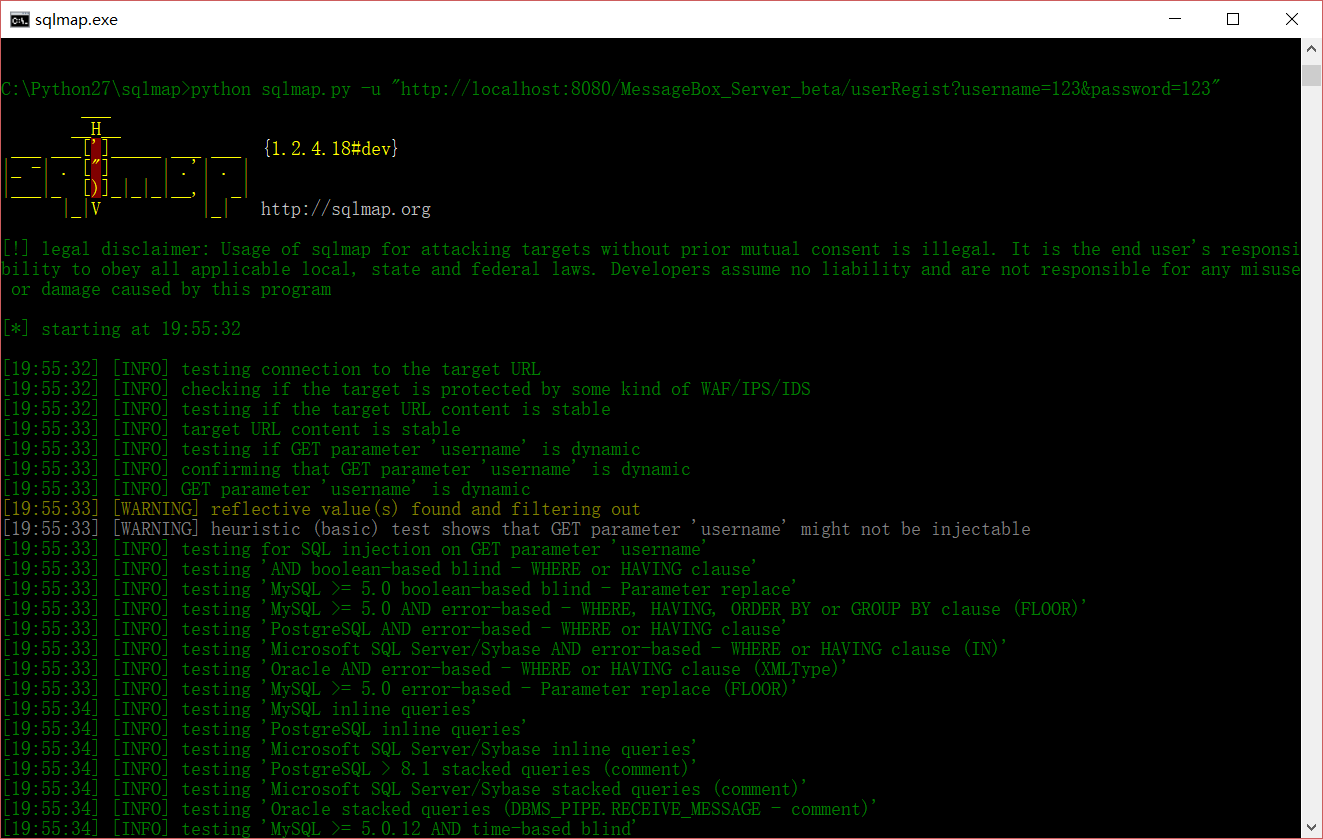
[WARNING] GET parameter 'username' does not seem to be injectable

[WARNING] GET parameter 'password' does not seem to be injectable

(2)注册接口测试

测试URL:

<http://localhost:8080/MessageBox_Server_beta/userRegist?username=123&password=123>



Username和password都不存在注入点

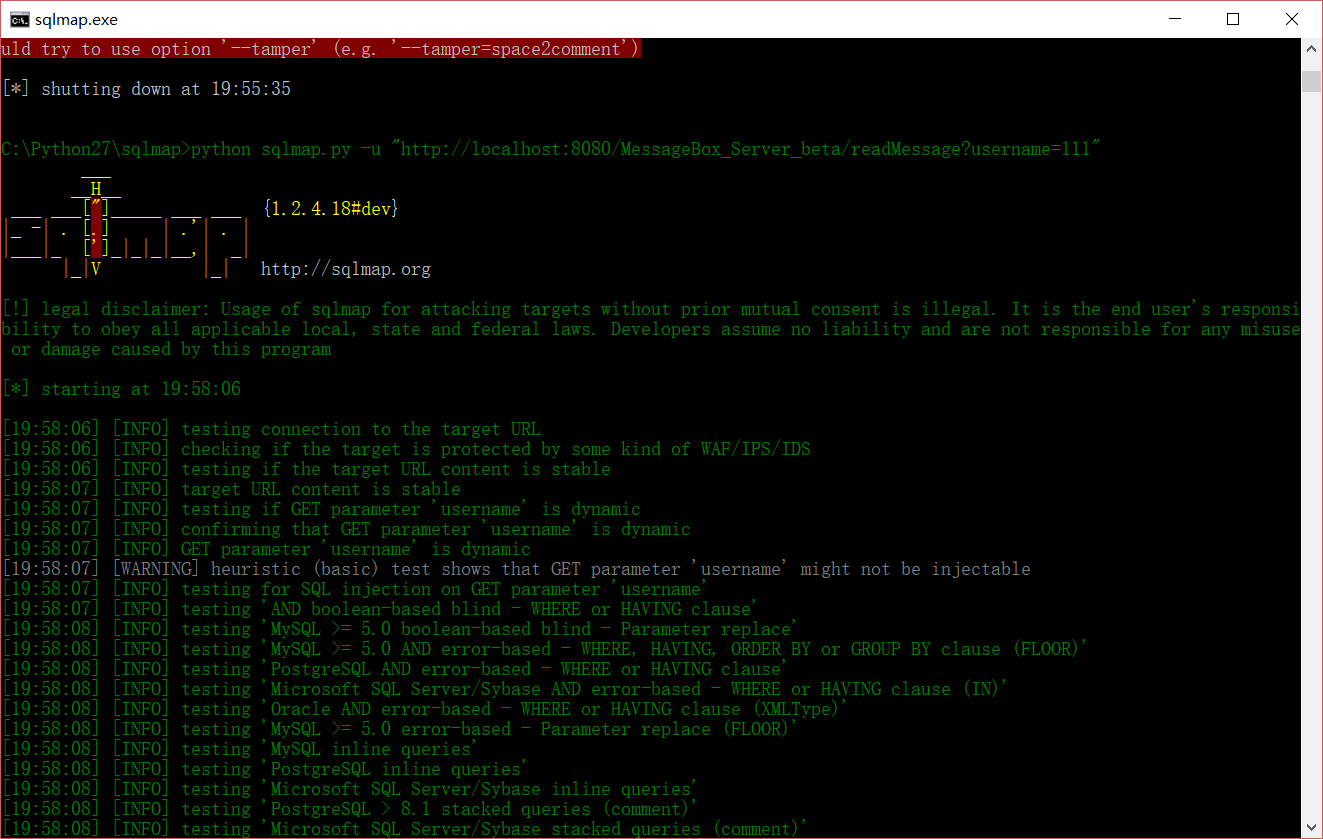
[WARNING] GET parameter 'username' does not seem to be injectable

[WARNING] GET parameter 'password' does not seem to be injectable

1. 读取短信接口测试

测试URL:

<http://localhost:8080/MessageBox_Server_beta/readMessage?username=111>



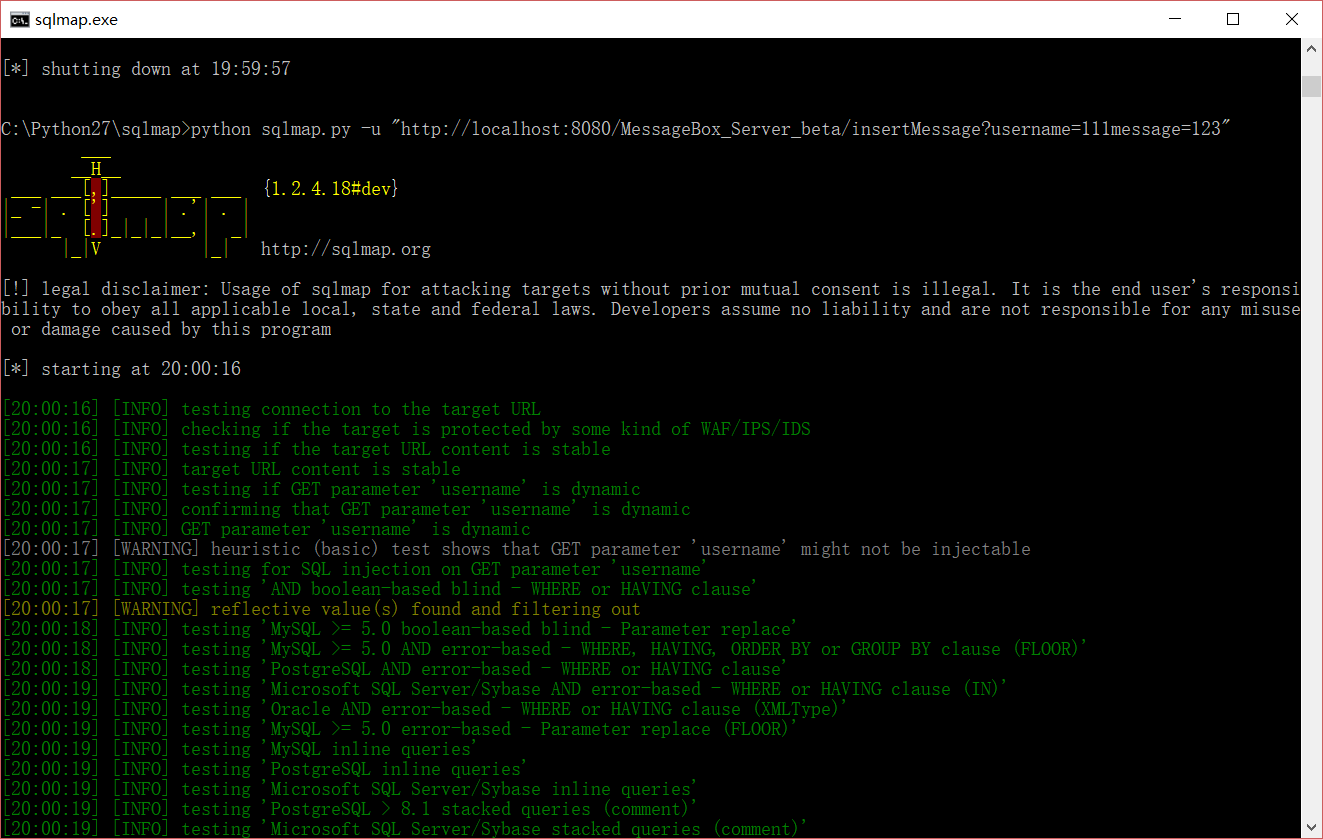
Username不存在注入点

[WARNING] heuristic (basic) test shows that GET parameter 'username' might not be injectable

1. 读取短信接口测试

测试URL：

<http://localhost:8080/MessageBox_Server_beta/insertMessage?username=111message=123>



Username和Message都不可能被注入

[20:01:31] [WARNING] GET parameter 'username' does not seem to be injectable

[20:01:31] [WARNING] heuristic (basic) test shows that GET parameter 'message' might not be injectable

结论：因为后端操作没有采用代码拼接的方式，有效避免了SQL注入等针对WEB端的攻击。结果证明服务端不存在SQL漏洞以及被利用漏洞造成更大的损失，证明了这个软件系统的安全性。