重 庆 大 学

学 生 实 验 报 告

实验课程名称 JAVA EE 程序设计

开课实验室 DS1501

学 院 大数据与软件学院 年级 2023 专业班 软工 1 班

学 生 姓 名 学 号

开 课 时 间 2024 至 2025 学年第 2 学期

成 绩	
教师签名	

大数据与软件学院制

《JAVA EE 程序设计》实验报告

开课实验室：DS1502 **2021 年 4 月 12 日**

学院	大数据与软件学院	年级、专业、班	23 级软工 1 班	姓名		成绩	

课程名称	**JAVA EE 程序设计**	实验项目名　称	实验7-8:事件编程、数据库应用编程。	指导教师	鄢萌

教师评	教师签名： 2022 年　月　日

一、 实验目的

实验七：基本掌握使用 JAVA 来完成基本的图形界面等调试与编程，事件编程调试。
实验八：掌握利用JAVA完成数据库课程中的基本数据记录操作。

二、实验内容

实验内容：
利用图形界面编程，开发简易C/S模式的管理信息系统，题目自拟，实现数据库访问操作。完成SQL SERVER或MYSQL数据库管理系统的安装，配置，JAVA数据库访问环境的配置；完成数据库表的建立，记录插入等；建立用户表，包含用户名、密码字段；利用JAVA实现图形界面，用户登录验证，进入软件运行后，针对某表记录的增、删、改、查操作。

三、使用仪器、材料

JAVA SE JDK 1.8.121

Tomcat

Mysql/Sql server

四、实验过程原始记录(数据、图表、计算等)：

本次实验,构建了一个二手物品交易平台,在该平台上,每一位用户可以购买物品也可以挂网自己的物品

项目规模:1592 行

```
D:\代码文件\java\FinalProgram\src\GUI>for %f in (*.java) do @find /c /v "" "%f"
---------- ADDGOODS.JAVA: 112
---------- CHANGE.JAVA: 146
---------- IMAGE.JAVA: 23
---------- INFODIALOG.JAVA: 56
---------- LINK.JAVA: 281
---------- LOAD.JAVA: 95
---------- LOADED.JAVA: 373
---------- MAIN.JAVA: 22
---------- PERSON.JAVA: 210
---------- PRODUCT.JAVA: 59
---------- REGSTER.JAVA: 201
---------- TEST.JAVA: 14
```

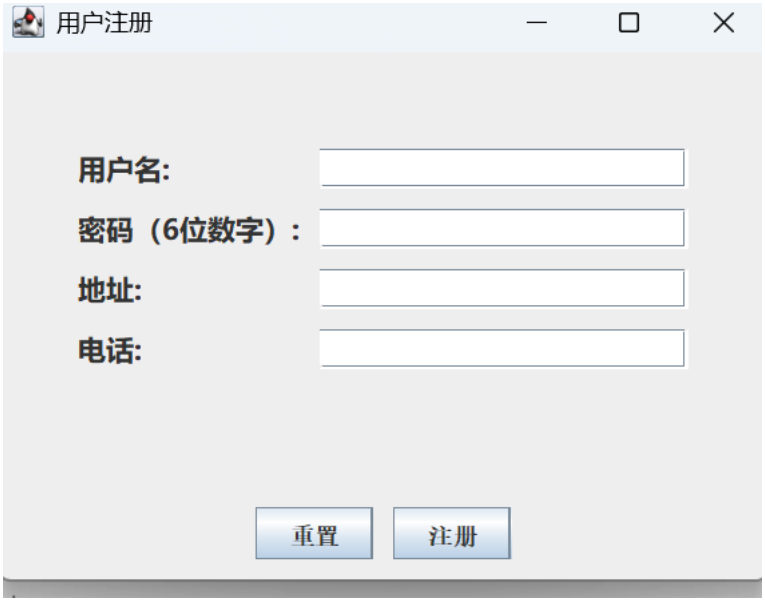
运行结果：

1.登录界面:有用户名，密码（用户名要求不重复）

对应数据库:user

	name	password	address	phoneNumber
▶	10	147258	重庆	19112964466
	15	258963	山东威海	15874269358
	19	123654	四川绍兴	19112965566
	20	145623	重庆沙坪坝	19112664455
	hp	195875	安徽凤阳	12345841515
	yang	195784	北京三环	19512587452

2.注册功能:点击注册按钮,出现四个候选框,同时按钮功能改变为确定和重置按钮.注册完成后,自动返回正常状态

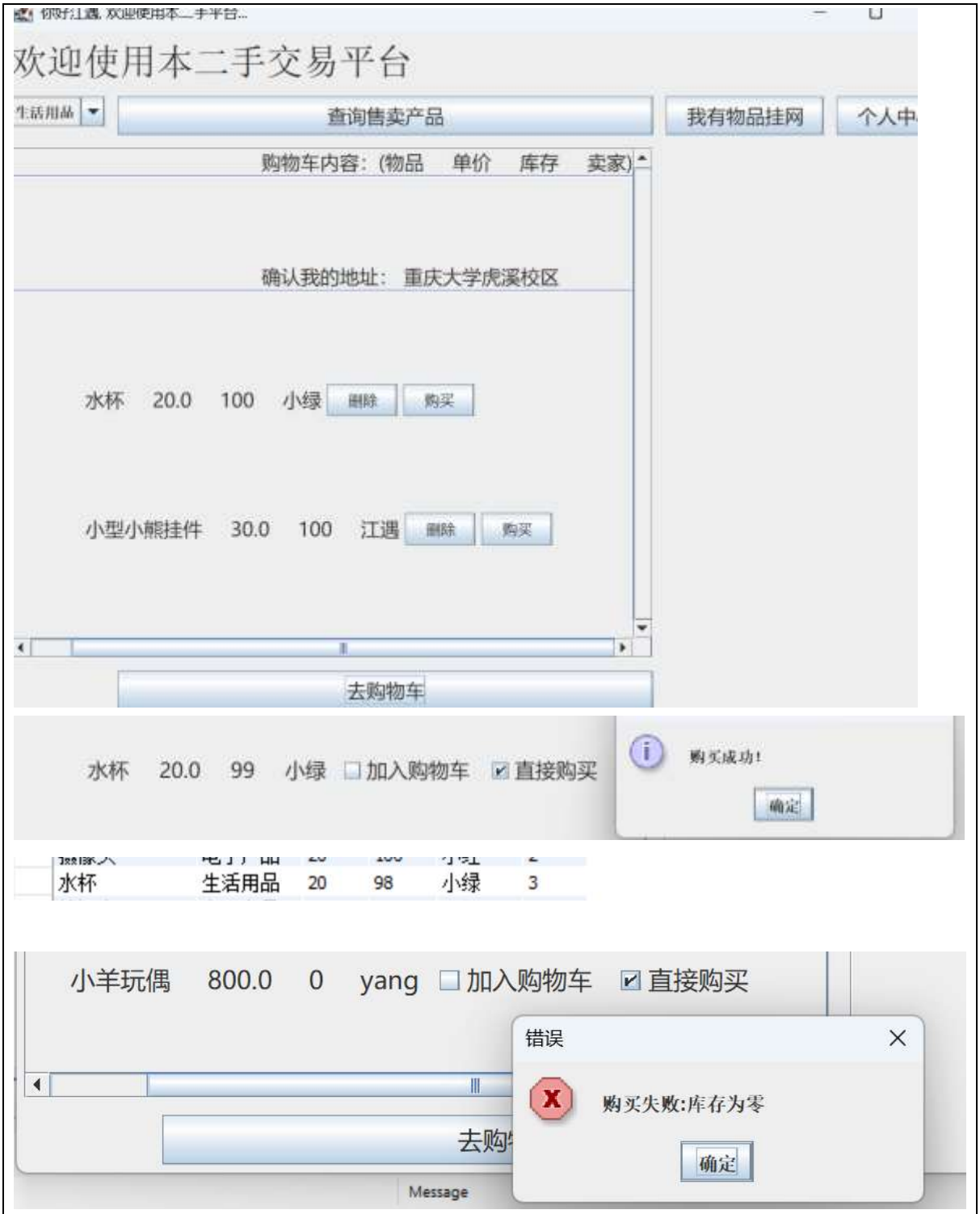


对这些密码和电话号码的输入有着严格要求，地址不做规定可在后续购买物品时修改

3.主页面:主页面包括五个部分.分别是顶部欢迎词条,右侧查询物品和左侧挂网和回到个人中心，以及最下面的购物车

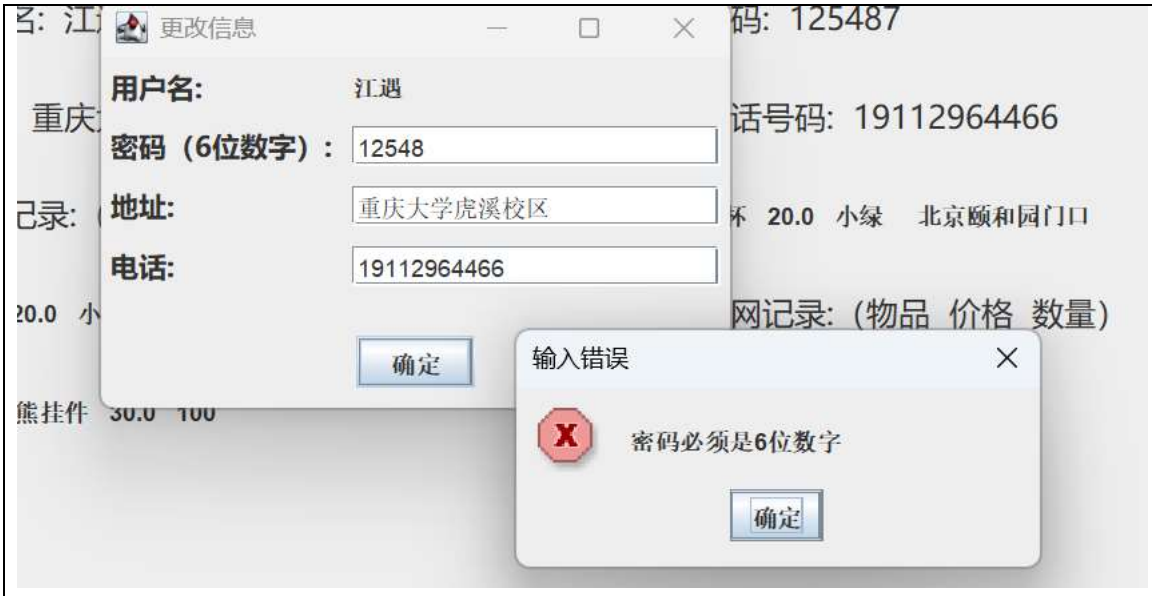
3.1 查找设置:通过不同类型的物品查找，在显示结果中可以选择直接购买和加入购物车稍后购买

3.2 购物车界面：显示之前添加的物品，也可以删除不想要的物品，同时这里显示地址提醒用户确认地址，如果有误可在个人中心修改，省略付费，购买成功则弹出提示框并添加到个人中心，数据库中商品数量-1，重新查询则会显示数量-1，同时假设购买已经没有库存的商品，会弹出错误提示，但是用户可以依旧添加到购物车等待后续卖家可能增加挂网数量



3.3 个人中心：我的基本信息和购买以及挂网记录，支持修改信息和刷新页面

3.3.1 修改个人信息，刷新后个人中心显示新信息，为了便于管理，用户名不可更改，密码必须是八位，电话号码必须正确填写



3.4 物品挂网，填写物品基本信息然后物品即可在查询界面被看见与更新数据库


欢迎使用本二手交易平台
生活用品
我有物品挂网
购买:
物品名:
小猫水杯
确定
数量:
6
定价:
20
类型:
学习用品
成功
挂网成功!
确定


查询售卖产品
我有物品
名称 单价 库存 卖家
钢笔 30.0 100 小黄 加入购物车 直接购买
小猫水杯 20.0 6 江遇 加入购物车 直接购买
去购物车

name	type	price	stock	saler	id
小熊玩偶	生活用品	600	100	王小明	1
摄像头	电子产品	20	100	小红	2
水杯	生活用品	20	99	小绿	3
笔记本	电子产品	8000	99	hp	4
钢笔	学习用品	30	100	小黄	5
小型小熊挂件	生活用品	30	100	江遇	6
小羊玩偶	生活用品	800	98	yang	7
小猫水杯	学习用品	20	6	江遇	8

核心代码：

完整代码见项目压缩包与配套数据库文件.(入口文件 main.java)

此处列出较为特点代码:

按钮实现：//添加按钮我有物品挂网

```
JButton addButton= new JButton("我有物品挂网");
addButton.setFont(new Font("微软雅黑", Font.PLAIN, 16));
// Select.setBackground(Color.BLUE);
 addButton.setPreferredSize(new Dimension(150, 40));
 addButton.addActionListener(e->add(userName));
 gbc.gridx = 2;
 gbc.gridy = 1;
 add(addButton, gbc);
 Select.addActionListener(e -> showSelect(userName));
 //添加按钮个人中心
 JButton myselfButton= new JButton("个人中心");
 // Select.setBackground(Color.BLUE);
 myselfButton.setFont(new Font("微软雅黑", Font.PLAIN, 16));
  myselfButton.setPreferredSize(new Dimension(120, 40));
  myselfButton.addActionListener(e->{
     try {
          myself(userName);
     } catch (SQLException e1) {
          // TODO 自动生成的 catch 块
          e1.printStackTrace();
     }
});
```

2.查询功能实现

```
private void showSelect(String username) {

     // TODO 自动生成的方法存根
```

```
            resultPanel.removeAll(); // 清空旧内容
            String selectedItem = (String)comboBox.getSelectedItem();
            link data=new link();


            // 添加标题
            JLabel title = new JLabel("名称        单价        库存          卖家");
            title.setFont(new Font("微软雅黑", Font.PLAIN, 20));
            resultPanel.add(title);
             resultPanel.add(new JSeparator(SwingConstants.HORIZONTAL));
             resultPanel.add(Box.createVerticalStrut(6)); // 垂直间距
            //resultPanel.add(new JSeparator());

            // ---------- 2. 添加分隔线 ----------
                    resultPanel.add(new JSeparator(SwingConstants.HORIZONTAL));
                    //resultPanel.add(Box.createVerticalStrut(20)); // 垂直间距
            try {
                List<product> products=data.getProductsByType(selectedItem);
                //遍历列表

                 for (int i = 0; i < products.size(); i++) {

                        JPanel itemPanel = new JPanel(new FlowLayout(FlowLayout.LEFT, 10, 5)); // 左对齐，水平间距 10

                        product p = products.get(i);
                        JLabel label = new JLabel("                                                    "+p.getName()+"
"+p.getPrice()+"        "+p.getStock()+"        "+p.getSaler());
                        label.setFont(new Font("微软雅黑", Font.PLAIN, 16));
                        itemPanel.add(label);


                       // 添加选择按钮
                        JCheckBox addBox = new JCheckBox("加入购物车");
                        addBox.setFont(new Font("微软雅黑", Font.PLAIN, 16));
                        addBox.addActionListener(e -> addBasket(label,p.getName()));
                        itemPanel.add(addBox);

                        //添加直接购买
                        JCheckBox buyNow = new JCheckBox("直接购买");
                        buyNow.setFont(new Font("微软雅黑", Font.PLAIN, 16));
                        buyNow.addActionListener(e -> {
                            addBasket(label, p.getName());   // 新增此行
                            buy(p.getName(), username,p.getStock());
                        });
                        itemPanel.add(buyNow);
```

```
                    //做添加到 itemPanel



                    resultPanel.add(itemPanel); //   添加商品项到结果面板
                    resultPanel.add(Box.createVerticalStrut(5)); //  添加间距

                    }



        }catch (Exception e) {
            JOptionPane.showMessageDialog(this,  "加载失败","错误",
JOptionPane.ERROR_MESSAGE);
        }
        cardLayout.show(mainPanel, "QUERY_VIEW");
        resultPanel.revalidate();//刷新布局
        resultPanel.repaint();
    }
```

3.购买功能实现

```
private void buy(String goodsName,String username,int stock) {
        if(stock==0) {
            Window parentWindow = SwingUtilities.getWindowAncestor(resultPanel);
            JOptionPane.showMessageDialog(parentWindow, "购买失败:库存为零" , "错误",
JOptionPane.ERROR_MESSAGE);
            return ;
        }

        buyPanel.removeAll();
        JLabel  aSJLabel=new  JLabel("购买：      名称                单价                库存
卖家");
        aSJLabel.setFont(new Font("微软雅黑", Font.PLAIN, 16));
        buyPanel.add(aSJLabel);
        buyPanel.add(new JSeparator());
        link data=new link();
        //显示商品信息：
        JLabel orderlJLabel=new JLabel();
        orderlJLabel=StringMapLabel.get(goodsName);
        buyPanel.add(orderlJLabel);

        //卖家地址
        try {
            if(data.getSalerByName(goodsName)!=null) {
                JLabel  salerAddress  =new  JLabel("卖家地址：
"+data.getSalerByName(goodsName));
```

```
                buyPanel.add(salerAddress);
            }

        } catch (SQLException e) {
            // TODO 自动生成的 catch 块
            e.printStackTrace();
        }
        //将订单信息插入到 order 表中
        try {
            if(data.getOderByName(username)!=null) {
                data.foundOrder(goodsName, username,data.getSalerByName(goodsName)) ;
                data.decreaseOrder(goodsName, username);
                   Window parentWindow = SwingUtilities.getWindowAncestor(resultPanel);
                     JOptionPane.showMessageDialog(parentWindow, "购买成功！ ");
            }



        } catch (SQLException e) {
            // TODO 自动生成的 catch 块



        }

        cardLayout.show(mainPanel, "BUY_VIEW");
        buyPanel.revalidate();
        buyPanel.repaint();

    }
```

3.link 类数据库链接

数据库链接常数

```
        final String URL = "jdbc:mysql://localhost:3306/trade";
        final String USER = "root";
        final String PASSWORD = "042519";
           /** 根据分类查询商品 */
        public List<product> getProductsByType(String type) throws SQLException {
            List<product> products = new ArrayList<>();
            String sql = "SELECT name,price,stock,saler FROM goods WHERE type=?";


            try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD);
                      PreparedStatement pstmt = conn.prepareStatement(sql)) {
                    pstmt.setString(1, type);
                    ResultSet rs = pstmt.executeQuery();

                    while (rs.next()) {
```

```
                product product = new product();
                product.setName(rs.getString("name"));
                product.setPrice(rs.getFloat("price"));
                product.setStock(rs.getInt("stock"));
                product.setSaler(rs.getString("saler"));
                products.add(product);
            }
        }
        return products;
    }
    //显示购买产品的名称，价格，卖家
    public List<product> getProductsByName(String name) throws SQLException {
        List<product> products = new ArrayList<>();
        String sql = "SELECT name,price,stock,saler FROM goods WHERE name=?";


        try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD);
             PreparedStatement pstmt = conn.prepareStatement(sql)) {
            pstmt.setString(1, name);
            ResultSet rs = pstmt.executeQuery();

            while (rs.next()) {
                product product = new product();
                product.setName(rs.getString("name"));
                product.setPrice(rs.getFloat("price"));
                product.setStock(rs.getInt("stock"));
                product.setSaler(rs.getString("saler"));
                products.add(product);
            }
        }
        return products;
    }
//显示顾客的地址
    public String getOderByName(String name) throws SQLException {
        String address = null;
        String sql = "SELECT address FROM user WHERE name=?";


        try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD);
             PreparedStatement pstmt = conn.prepareStatement(sql)) {
            pstmt.setString(1, name);
            ResultSet rs = pstmt.executeQuery();

            while (rs.next()) {
                address=rs.getString("address");
```

```
                }
            }
            return address;
        }
    //显示卖家地址
    public String getSalerByName(String goodsName) throws SQLException {
        String address = null;
        String sql = "SELECT address FROM user,goods WHERE goods.name=? and saler=user.name";


        try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD);
             PreparedStatement pstmt = conn.prepareStatement(sql)) {
            pstmt.setString(1, goodsName);
            ResultSet rs = pstmt.executeQuery();

            while (rs.next()) {
                address=rs.getString("address");

            }
        }
        return address;
    }
    //插入订单
    public void foundOrder(String goodsName,String username,String address) throws SQLException {

        boolean fine=false;

        String sqlSetorder ="INSERT INTO ordertable (goods, Customer, toAddress) VALUES (?,?,?)";
        String sqlSetsaler = "update  ordertable set ordertable.saler =( SELECT goods.saler FROM goods WHERE goods.name=?);";

        try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD)) {
            conn.setAutoCommit(false); // 关闭自动提交，开启事务
            try (PreparedStatement pstmt = conn.prepareStatement(sqlSetorder);
                 PreparedStatement pstmt2 = conn.prepareStatement(sqlSetsaler);)
                    {
                pstmt.setString(1, goodsName);
                pstmt.setString(2, username);
                pstmt.setString(3, address);
                pstmt2.setString(1, goodsName);

                int rowsInserted = pstmt.executeUpdate();
                int rowsUpdated = pstmt2.executeUpdate();
```

```
                    if (rowsInserted > 0 && rowsUpdated > 0) {
                        conn.commit(); //  都成功则提交事务
                        fine = true;
                    } else {
                        conn.rollback(); //  有失败则回滚事务
                    }
                } catch (SQLException e) {
                    conn.rollback(); //  捕获到异常也回滚事务
                    e.printStackTrace();
                }
            } catch (SQLException e) {
                e.printStackTrace();
            }
}
        //减少库存
        public void decreaseOrder(String goodsName, String username) throws SQLException {
            String salerString = null;
            String sqlSelectSaler = "SELECT goods.saler FROM goods WHERE goods.name=?";
            String sqldecrease = "update goods set stock=stock-1 where name= ? and saler=?";

            try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD);
                 PreparedStatement pstmt = conn.prepareStatement(sqlSelectSaler);
                 PreparedStatement pstmt2 = conn.prepareStatement(sqldecrease)) {

                // 1. 查询卖家

                pstmt.setString(1, goodsName);
                ResultSet rs = pstmt.executeQuery();

                //  确保查询到卖家信息

                if (rs.next()) {
                    salerString = rs.getString("saler");
                } else {
                    throw new SQLException("未找到商品或卖家: " + goodsName);
                }

                // 2. 执行库存更新
                pstmt2.setString(1, goodsName);
                pstmt2.setString(2, salerString);

                //  关键修复：执行更新操作
                int affectedRows = pstmt2.executeUpdate();

                //  检查是否成功更新
                if (affectedRows == 0) {
```

```
                    throw new SQLException("更新失败：库存未减少（可能商品不存在或卖家不匹配）");
                }

            } catch (SQLException e) {
                e.printStackTrace();
                throw e; // 将异常向上抛出，确保调用者能感知错误
            }
        }
        //通过用户名获取用户信息
        public String[] getMessageByName(String name){
            String[] person = new String[4]; // 创建一个长度为 4 的 String 数组
            person[0]=name;
             String sql="SELECT * FROM trade.user where name=?";
             try(Connection conn=DriverManager.getConnection(URL,USER,PASSWORD);
                     PreparedStatement pstmt = conn.prepareStatement(sql)) {
                pstmt.setString(1, name);
                ResultSet rs = pstmt.executeQuery();
                while(rs.next()) {
                    person[1]=    rs.getString("password");
                    person[2] =rs.getString("address");
                    person[3] =rs.getString("phoneNumber");
                }


            } catch (Exception e) {
                // TODO: handle exception

            }

            return person;
        }
        //更新用户的新挂网物品
  public void foundGoods(String username,String address, String goodsName,int stock,float price,String type) throws SQLException {

            boolean fine=false;

            String sqlSetorder ="INSERT INTO goods (name, price, saler,stock,type) VALUES (?,?,?,?,?)";



            try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD)) {
                conn.setAutoCommit(false); // 关闭自动提交，开启事务
                try (PreparedStatement pstmt = conn.prepareStatement(sqlSetorder);
                    )
```

```
                    {
                pstmt.setString(1,goodsName);
                pstmt.setFloat(2, price);
                pstmt.setString(3, username);
                pstmt.setInt(4, stock);
                pstmt.setString(5, type);


                int rowsInserted = pstmt.executeUpdate();

                if (rowsInserted > 0 ) {
                    conn.commit(); // 都成功则提交事务
                    fine = true;
                } else {
                    conn.rollback(); // 有失败则回滚事务
                }
            } catch (SQLException e) {
                conn.rollback(); // 捕获到异常也回滚事务
                e.printStackTrace();
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
        if(fine) {
            JOptionPane.showMessageDialog(this, "挂网成功！", "成功",
JOptionPane.INFORMATION_MESSAGE);
        }
}
  //通过不同用户名显示购买记录
  public List<JLabel> getOrderByName(String name) throws SQLException {
    List<JLabel> orders = new ArrayList<>();
    JLabel itemJLabel=new JLabel();
    String sql = "SELECT goods,price, goods.saler, toAddress FROM goods,user,ordertable WHERE
goods.name=goods and user.name=ordertable.saler "
        + "and ordertable.Customer=?";


    try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD);
         PreparedStatement pstmt = conn.prepareStatement(sql)) {
        pstmt.setString(1, name);
        ResultSet rs = pstmt.executeQuery();

        while (rs.next()) {
            JLabel orderLabel = new JLabel(rs.getString("goods")+"
"+rs.getFloat("price")+"  "+rs.getString("goods.saler")+
                    "  "+rs.getString("toAddress"));
```

```
                    orders.add(orderLabel);
                }
            }
            return orders;
        }
    //通过不同用户名显示挂网物品
    public List<JLabel> getGoodsByName(String name) throws SQLException {
        List<JLabel> orders = new ArrayList<>();
        JLabel itemJLabel=new JLabel();
        String sql = "SELECT name,price,stock FROM goods WHERE saler=? ";

        try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD);
                PreparedStatement pstmt = conn.prepareStatement(sql)) {
            pstmt.setString(1, name);
            ResultSet rs = pstmt.executeQuery();

            while (rs.next()) {
                JLabel orderLabel = new JLabel(rs.getString("name")+"
"+rs.getFloat("price")+"  "+rs.getInt("stock"));
                orders.add(orderLabel);
            }
        }
        return orders;
    }



    }
```

总结：

本次实验通过 JAVA swing 开发了一个完整可用的二手物品挂网平台,实现了与数据库的增删改查功能.